



Fundusze Europejskie
dla Podkarpacia



Rzeczpospolita
Polska

Dofinansowane przez
Unię Europejską



PODKARPACKIE
przestrzeń otwarta

Paweł Dymora

Miroslaw Mazurek

Mariusz Nycz

Joanna Rysz

Mateusz Salach

Bartosz Kowal

Karol Siwiec

Język maszyn

Od odkrywcy do inżyniera



RaP

STEAM

Informatyka dla klasy 7 szkoły podstawowej

Autorzy:

dr inż. Paweł Dymora

dr inż. Mirosław Mazurek

dr inż. Mariusz Nycz

mgr inż. Joanna Rysz

mgr inż. Mateusz Salach

mgr inż. Bartosz Kowal

mgr inż. Karol Siwiec

Podręcznik został opracowany w ramach projektu „RaP STEAM - robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego” współfinansowanego ze środków Europejskiego Funduszu Społecznego Plus w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021–2027, realizowanego przez Województwo Podkarpackie - Podkarpacki Zespół Placówek Wojewódzkich w Rzeszowie we współpracy z Partnerami projektu:

- Politechniką Rzeszowską im. Ignacego Łukasiewicza,
- Gminą Grodzisko Dolne - Zespołem Szkół im. prof. Franciszka Leji w Grodzisku Górnym,
- Wojewódzkim Domem Kultury w Rzeszowie - Podkarpackim Centrum Nauki „Łukasiewicz”.

Rzeszów 2026

Spis treści

Moduł 1 - Graficzna alchemia przyszłości

Lekcja nr 1 – Przygotowanie rysunku nt. „Księżycowa wyprawa. Roboty budują bazę dla przyszłych misji!”.	5
Lekcja nr 2 – Eksport gotowego rysunku do formatu graficznego.	13
Lekcja nr 3 – Import rysunku do programu graficznego w celu edycji.	20
Lekcja nr 4 – Przygotowanie plakat nt. „Nasze roboty na Księżycu!”.	29

Moduł 2 - Odkrywamy tajemnice Kodu... Spotkanie z robotem

Lekcja nr 5 - Wprowadzenie do języka Python. Różnice i podobieństwa języka graficznego i tekstowego.	39
Lekcja nr 6 - Programowanie w języku tekstowym. Od czego zacząć?	55
Lekcja nr 7 - Kodujemy ruch naszego robota. Jak korzystać z bibliotek kodu?	81
Lekcja nr 8 - Czy można w języku Python programować szkolne roboty?	103
Lekcja nr 9 - Odważnie naprzód! Najlepiej slalomem - zadanie dotyczące mobilności robota. Program w języku Python.	133

Moduł 3 - Globalna wspólnota Cyfrowego Świata

Lekcja nr 10 – Budowa sieci komputerowej.	162
Lekcja nr 11 – Wpływ systemów komputerowych na życie człowieka.	174
Lekcja nr 12 – Sieci rozległe jako kluczowy element globalnej komunikacji.	188

Moduł 4 - Biurowa moc TechOffice

Lekcja nr 13 - Jak wykonać dokumentację oprogramowania?	198
Lekcja nr 14 - Automatyczne generatory spisów ułatwiają pracę.	214
Lekcja nr 15 - Tabele i funkcje porządkujemy dane.	237
Lekcja nr 16 - Test sensorów i wykres w arkuszu kalkulacyjnym.	255

Moduł 5 - Odkrywamy tajemnice Kodu... Robot eksplorator

Lekcja nr 17 - Potęga eksperymentu. Testujemy roboty dla uzyskania najlepszych efektów podczas ruchu.	273
---	-----

Lekcja nr 18 - Robot dostarcza autonomicznie różne elementy niezbędne ludziom w pracy. Analiza problemu i tworzenie planu kodu źródłowego.	296
Lekcja nr 19 - Robot dostawca – realizacja programu i testy praktyczne.	322
Lekcja nr 20 - Kontrola dodatkowych silników. Przenoszenie ładunków na różne wysokości. Analiza problemu.	343
Lekcja nr 21 - Kontrola dodatkowych silników - realizacja zadania.	362
Lekcja nr 22 - Usprawniamy kod źródłowy za pomocą funkcji.	385
Lekcja nr 23 - Przenoszenie ładunków – analiza kodu pod kątem wykorzystania funkcji.	408
Lekcja nr 24 - Przenoszenie ładunków - realizacja zadania z użyciem funkcji.	425

Moduł 6 - PixelPerfect Studio

Lekcja nr 25 - Łączenie klipów filmowych z grafikami wytworzonymi w dziale „Graficzna alchemia przyszłości”.	450
Lekcja nr 26 - Jak zarządzać wieloma ścieżkami na osi czasu projektu filmowego?	461
Lekcja nr 27 - Dodawanie efektów wideo - filtry, korekcja kolorów oraz efekty specjalne?	472
Lekcja nr 28 - Tworzenie animowanych przejść pomiędzy klipami.	485
Lekcja nr 29 - Edycja dźwięku - korekcja, dodawanie ścieżek dźwiękowych i innych efektów.	499
Lekcja nr 30 - Wybieranie ustawień eksportu i renderowania filmu dla różnych platform.	510

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 1: Graficzna alchemia przyszłości

Temat lekcji nr 1 – Przygotowanie rysunku nt. „Księżycowa wyprawa. Roboty budują bazę dla przyszłych misji!”

Treści nauczania (wymagania szczegółowe - podstawa programowa) – II. 3) a); III. 2); III. 3);

Wymagania wstępne (wiadomości i umiejętności uczniów):

Uczeń przed rozpoczęciem lekcji (umiejętności zdobyte w klasie VI):

- zna interfejs programu Krita (obszar roboczy, paski narzędzi, paleta warstw),
- potrafi:
 - uruchomić program Krita i utworzyć nowy dokument,
 - zapisać plik w formacie .kra oraz .png,
 - korzystać z podstawowych narzędzi rysunkowych (pędzel, gumka),
 - korzystać z warstw

Cel lekcji:

- Uczniowie, na podstawie przygotowanych przez nauczyciela materiałów graficznych, projektują rysunek dotyczący eksploracji Księżyca przez roboty.
- Dla potrzeb lekcji wykorzystano edytor graficzny Krita. Jest to bezpłatny program graficzny open-source.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- Uczeń korzystając z aplikacji komputerowych, przygotowuje dokumenty i prezentacje, także w chmurze, na pożytek rozwiązywanych problemów i własnych prac z różnych dziedzin (przedmiotów), dostosowuje format i wygląd opracowań do ich treści i przeznaczenia, wykazując się przy tym umiejętnościami:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- tworzenia i podstawowej obróbki cyfrowej plików multimedialnych (zdjęć, filmów).
- Uczeń rozwija umiejętności korzystania z różnych urządzeń do tworzenia elektronicznych wersji tekstów, obrazów, dźwięków, filmów i animacji;
- Uczeń poprawnie posługuje się terminologią związaną z informatyką i technologią.

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści programowe będą realizowane za pomocą uniwersalnych narzędzi dostępnych w systemach operacyjnych, przede wszystkim menedżerach okien. W miarę możliwości, spowodowanej dostępnością usługi zapisu danych w chmurze, uczniowie będą korzystać z zapisu danych w środowiskach wirtualnych.
- Realizacja za pomocą uniwersalnych narzędzi, w które wyposażone są wszystkie wykorzystywane w szkołach systemy operacyjne (np. eksploratory plików).
- Treści programowe realizowane będą w programie Krita. Jest to bezpłatny program graficzny open-source.

Uczeń:

- projektuje rysunek na podstawie danych materiałów,

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna,
- praca na komputerze,
- pogadanka.

Środki dydaktyczne

- zestaw PC, laptop, komputer All in One lub stacja robocza z dostępem do Internetu, zainstalowaną przeglądarką i ustawieniami regionalnymi dla języka polskiego, zainstalowany program Krita
- rzutnik multimedialny lub tablica multimedialna,
- e-podręcznik.

Formy pracy

- Praca indywidualna.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przebieg lekcji:

Faza I. Wprowadzenie

Nauczyciel podaje temat lekcji, tłumaczy uczniom cel lekcji: zaprojektowanie w programie Krita rysunku, który na kolejnych lekcjach będzie wyeksportowany do programu GIMP, żeby utworzyć plakat.

Faza II. Część zasadnicza - realizacja

1. Nauczyciel wykorzystując e-platformę demonstruje na tablicy multimedialnej część teoretyczną uruchamiając odpowiedni moduł i materiał multimedialny.
2. Uczniowie logują się na swoich kontach na platformie edukacyjnej, gdzie otwierają lekcję w podanym przez nauczyciela module.
3. Prezentowany materiał multimedialny, zrealizowany w oparciu o szczegółowy scenariusz (załącznik nr 1 do lekcji), oprócz waloru teoretycznego stanowi również podstawę przebiegu części praktycznej lekcji. W szczególności, uczniowie:
 - a. uruchamiają program graficzny Krita i tworzą nowy plik graficzny;
 - b. tworzą projekt, zapisują go we wskazanym folderze (nauczyciel wskazuje właściwą lokalizację; może to być dysk lokalny lub zasób w chmurze);
4. Nauczyciel objaśnia sposób pracy na lekcji – we wskazanym przez niego momencie uczniowie będą wykonywać kolejne etapy prowadzące do rozwiązania zadania. Zdolniejsi uczniowie spróbują wykonać je samodzielnie. Słabsi wykonają kolejne kroki razem z nauczycielem, w oparciu o przygotowaną do zajęć instrukcję dostępną w e-podręczniku lub materiał multimedialny. W razie potrzeby będą mogli dopytać nauczyciela. Nauczyciel na bieżąco sprawdza postępy uczniów.

Faza III. Podsumowanie

Uczniowie prezentują swoje obrazy, nauczyciel omawia je i ocenia. Kryteria oceniania:

Ocena dopuszczająca:

– uczeń tworzy nowy obraz w Krita, wkleja tło z materiałów od nauczyciela.

Ocena dostateczna:

– uczeń tworzy obraz na zadany temat wykorzystując grafiki od nauczyciela.

Ocena dobra:

– uczeń potrafi zastosować różne funkcje programu Krita.

Ocena bardzo dobra:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

– uczeń świadomie stosuje narzędzia i funkcje programu graficznego, żeby uzyskać oczekiwany efekt.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, dostępnych również na platformie edukacyjnej w formie testu dla uczniów, np.:

1. Jakiego formatu tworzymy obraz?
 - a. A4
 - b. B5
 - c. E4
 - d. F1
2. Gdzie roboty mają budować bazę, jakie to miejsce?
 - a. Klasa lekcyjna
 - b. Ziemia
 - c. Mars
 - d. Księżyc
3. W jakim formacie zapisujemy projekt?
 - a. .kra
 - b. .png
 - c. .bmp
 - d. .txt

Faza V. Zadanie do samodzielnego wykonania.

- **Dodatkowe grafiki.**
 - Znajdź samodzielnie w Internecie obrazy kuli ziemskiej lub/i kraterów księżycowych, które możesz wkomponować w pracy.

Zadanie do samodzielnego wykonania – Sztuczna Inteligencja *.

- **Wygeneruj grafikę.**
 - Wykorzystując programy generatywnej AI (do generowania obrazów – Canva, <https://raphael.app/pl> i inne) utwórz grafikę tła – strukturę powierzchni księżyca, którą dodasz do projektu. Napisz jakie prompty używasz.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 1 – Przygotowanie rysunku nt.

„Księżycowa wyprawa. Roboty budują bazę dla przyszłych misji!”

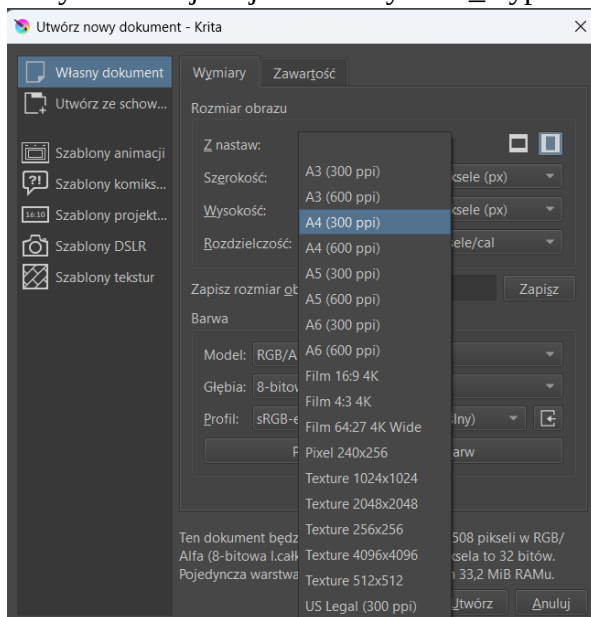
Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
- Przedstawienie uczniom celu i zakresu tematycznego lekcji.
- Uruchomienie na platformie RaP STEAM odpowiedniej e-lekcji.
- Indywidualna praca uczniów na komputerach.
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów (test na e-platformie).
- Zadanie pracy do samodzielnego wykonania – dostępne również na e-platformie (opcjonalnie).

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 2 – Realizacja zadania głównego podczas lekcji nr 1 – Przygotowanie rysunku nt. „Księżycowa wyprawa. Roboty budują bazę dla przyszłych misji!”

1. W programie Krita utwórz nowy obraz o wymiarach A4 (300 ppi). Zapisz obraz we wskazanym przez nauczyciela miejscu jako Ksiezykowa_wyprawa.kra

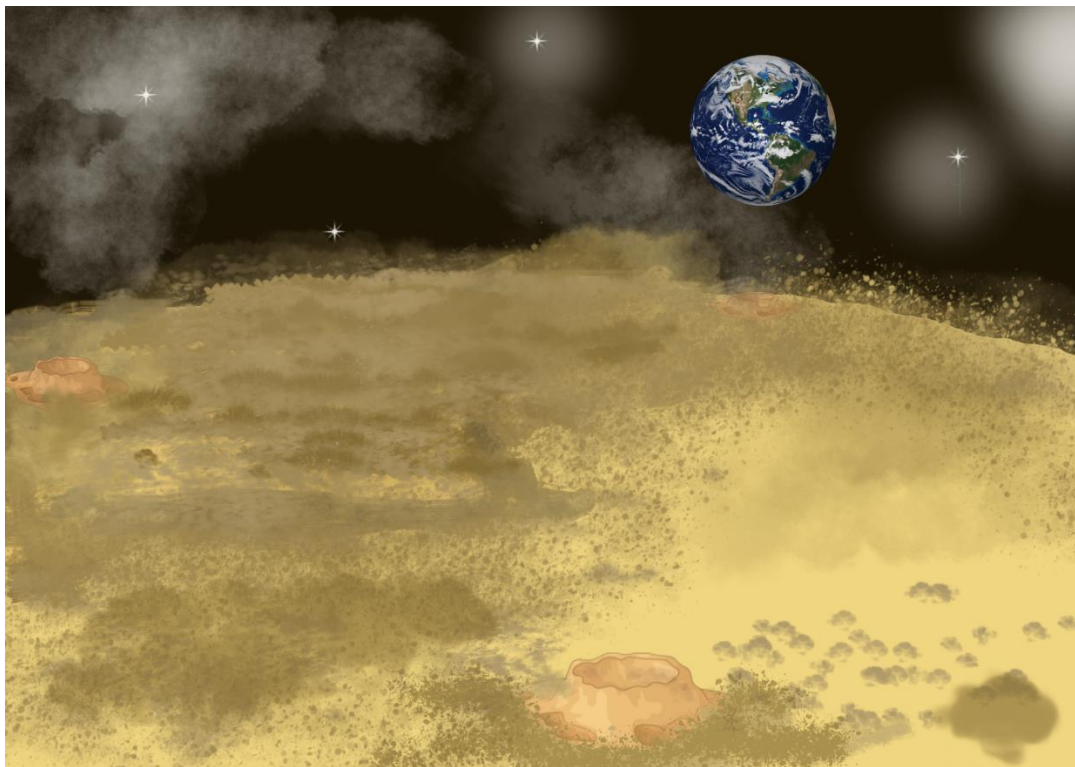


2. Używając materiałów graficznych udostępnionych przez nauczyciela na e-platformie oraz własnej kreatywności stwórz tło do plakatu nt. „Księżycowa wyprawa. Roboty budują bazę dla przyszłych misji!”. Pamiętaj o regularnym zapisywaniu swojej pracy. Warto potestować i użyć różnych pędzli i stempli:



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

3. Przykładowe tło (użyto gotowej grafiki kuli ziemskiej i krateru):



Kolejne przykładowe tło (użyto grafiki powierzchni księżyca i kuli ziemskiej):



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie do samodzielnego wykonania.

- **Baza.**
 - Dodaj do obrazu bazę, którą będą budować roboty. Możesz wykorzystać gotowe grafiki od nauczyciela, albo namalować samemu.

Zadanie do samodzielnego wykonania – Sztuczna Inteligencja *.

- **Wygeneruj grafikę.**
 - Wykorzystując programy generatywnej AI (do generowania obrazów – Canva, <https://raphael.app/pl> i inne) utwórz grafikę tła – strukturę powierzchni księżyca, którą dodasz do projektu albo projekt bazy. Napisz jakie prompty używasz.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 1: Graficzna alchemia przyszłości

Temat lekcji nr 2 – Eksport gotowego rysunku do formatu graficznego.

Treści nauczania (wymagania szczegółowe - podstawa programowa) – II. 4).

Cel lekcji:

- Uczniowie eksportują gotowe projekty graficzne do formatów, które umożliwiają dalszą edycję.
- Dla potrzeb lekcji wykorzystano edytor graficzny Krita. Jest to bezpłatny program graficzny open-source.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- Uczeń gromadzi, porządkuje i selekcjonuje efekty swojej pracy oraz potrzebne zasoby w komputerze lub w innych urządzeniach, a także w środowiskach wirtualnych (w chmurze).

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści programowe będą realizowane za pomocą uniwersalnych narzędzi dostępnych w systemach operacyjnych, przede wszystkim menedżerach okien. W miarę możliwości, spowodowanej dostępnością usługi zapisu danych w chmurze, uczniowie będą korzystać z zapisu danych w środowiskach wirtualnych.
- Realizacja za pomocą uniwersalnych narzędzi, w które wyposażone są wszystkie wykorzystywane w szkołach systemy operacyjne (np. eksploratory plików).
- Treści programowe realizowane będą w programie Krita. Jest to bezpłatny program graficzny open-source.

Uczeń:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- potrafi wyeksportować rysunek do danego formatu.

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna,
- praca na komputerze,
- pogadanka.

Środki dydaktyczne

- zestaw PC, laptop, komputer All in One lub stacja robocza z dostępem do Internetu, zainstalowaną przeglądarką i ustawieniami regionalnymi dla języka polskiego, zainstalowany program Krita
- rzutnik multimedialny lub tablica multimedialna,
- e-podręcznik.

Formy pracy

- Praca indywidualna.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przebieg lekcji:

Faza I. Wprowadzenie

Nauczyciel przypomina zadanie z poprzednich zajęć, a następnie podaje temat aktualnej lekcji i wyjaśnia po co należy wyeksportować gotowe projekty.

Faza II. Część zasadnicza - realizacja

1. Nauczyciel wykorzystując e-platformę demonstruje na tablicy multimedialnej część teoretyczną uruchamiając odpowiedni moduł i materiał multimedialny.
2. Uczniowie logują się na swoich kontach na platformie edukacyjnej, gdzie otwierają lekcję w podanym przez nauczyciela module.
3. Prezentowany materiał multimedialny, zrealizowany w oparciu o szczegółowy scenariusz (załącznik nr 1 do lekcji), oprócz waloru teoretycznego stanowi również podstawę przebiegu części praktycznej lekcji. W szczególności, uczniowie:
 - a. uruchamiają program graficzny Krita, otwierają swój projekt, kończą go;
 - b. eksportują projekt do podanego przez nauczyciela formatu, zapisują go we wskazanym folderze (nauczyciel wskazuje właściwą lokalizację; może to być dysk lokalny lub zasób w chmurze);
4. Nauczyciel objaśnia sposób pracy na lekcji – we wskazanym przez niego momencie uczniowie będą wykonywać kolejne etapy prowadzące do rozwiązania zadania. Zdolniejsi uczniowie spróbują wykonać je samodzielnie. Słabsi wykonają kolejne kroki razem z nauczycielem, w oparciu o przygotowaną do zajęć instrukcję dostępną w e-podręczniku lub materiał multimedialny. W razie potrzeby będą mogli dopytać nauczyciela. Nauczyciel na bieżąco sprawdza postępy uczniów.

Faza III. Podsumowanie

Uczniowie prezentują swoje obrazy, nauczyciel omawia je i ocenia. Kryteria oceniania:

Ocena dopuszczająca:

- uczeń otwiera swój projekt w Krita.

Ocena dostateczna:

- uczeń kończy projekt na zadany temat.

Ocena dobra:

- uczeń potrafi wyeksportować plik do zadanego formatu.

Ocena bardzo dobra:

- uczeń samodzielnie dobiera format eksportu.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, dostępnych również na platformie edukacyjnej w formie testu dla uczniów, np.:

1. Do jakich formatów możemy wyeksportować plik w programie Krita?
 - a. **wszystkich podanych**
 - b. .ora
 - c. .png
 - d. .jpg
2. Czy eksport to to samo co zapisz plik?
 - a. tak
 - b. **nie**
3. Dlaczego eksportujemy plik do innego formatu?
 - a. **Żeby móc go edytować w innym programie.**
 - b. Żeby lepiej wyglądał.
 - c. Żeby był mniejszy.
 - d. Wszystkie odpowiedzi są poprawne.

Faza V. Zadanie do samodzielnego wykonania.

- **Inny format.**
 - Wyeksportuj plik do formatu .ora, Zastanów się, który format będzie dla Ciebie lepszy, uzasadnij wybór.

Zadanie do samodzielnego wykonania – Sztuczna Inteligencja *.

- **Wykorzystaj AI.**
 - Wykorzystując sztuczną inteligencję (z wyszukiwarki Google lub innego źródła) napisz czym różnią się dostępne do eksportu rozszerzenia w programie Krita. Które rozszerzenie będzie najlepsze do dalszej edycji w programie Gimp i dlaczego?

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 2 – Eksport gotowego rysunku do formatu graficznego.

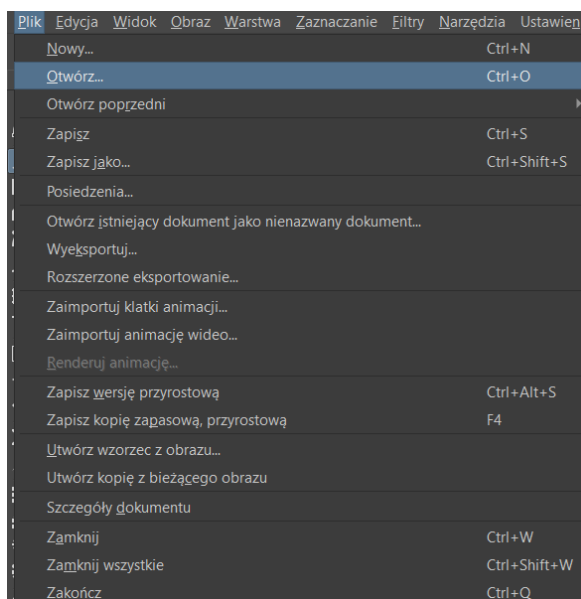
Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
- Przedstawienie uczniom celu i zakresu tematycznego lekcji.
- Uruchomienie na platformie RaP STEAM odpowiedniej e-lekcji.
- Indywidualna praca uczniów na komputerach.
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów (test na e-platformie).
- Zadanie pracy do samodzielnego wykonania – dostępne również na e-platformie (opcjonalnie).

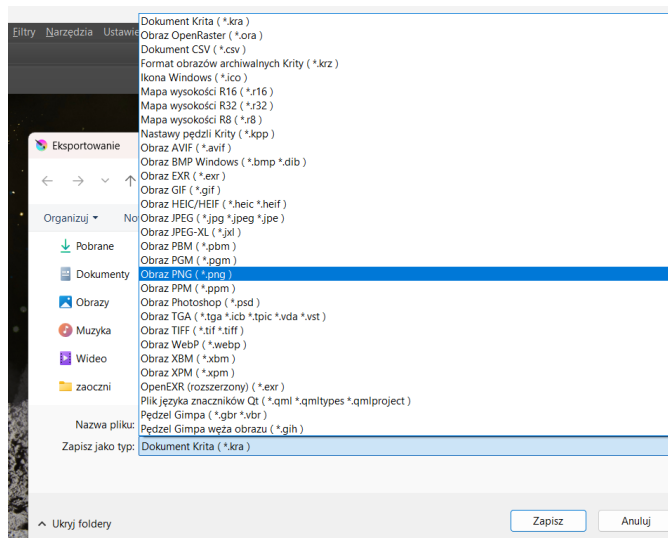
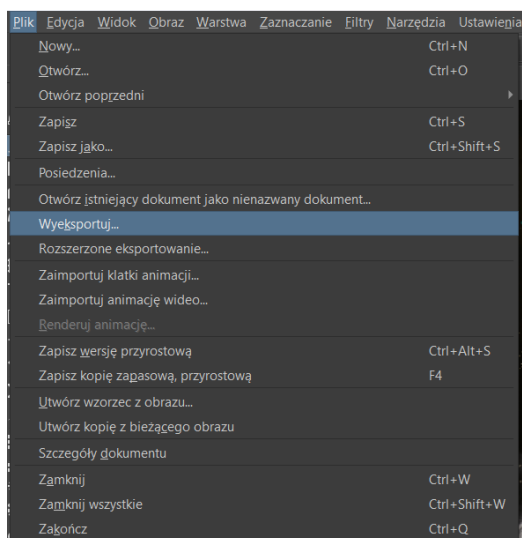
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 2 – Realizacja zadania głównego podczas lekcji nr 2 – Eksport gotowego rysunku do formatu graficznego.

1. W programie Krita otwórz projekt z poprzedniej lekcji.



2. Dokończ projekt rysunku nt. „Księżycowa wyprawa. Roboty budują bazę dla przyszłych misji!”. Zapisz zmiany (Ctrl + S).
3. Wyeksportuj obraz do formatu .png. Zapisz plik w miejscu wyznaczonym przez nauczyciela.



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie do samodzielnego wykonania.

- **Inny format.**
 - Wyeksportuj plik do formatu .ora, Zastanów się, który format będzie dla Ciebie lepszy, uzasadnij wybór.

Zadanie do samodzielnego wykonania – Sztuczna Inteligencja *.

- **Wykorzystaj AI.**
 - Wykorzystując sztuczną inteligencję (z wyszukiwarki Google lub innego źródła) napisz czym różnią się dostępne do eksportu rozszerzenia w programie Krita. Które rozszerzenie będzie najlepsze do dalszej edycji w programie Gimp i dlaczego?

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 1: Graficzna alchemia przyszłości

Temat lekcji nr 3 – Import rysunku do programu graficznego w celu edycji.

Treści nauczania (wymagania szczegółowe - podstawa programowa) – II. 4); II. 3) a); III. 2); III. 3);

Cel lekcji:

- Uczniowie importują rysunki stworzone na poprzedniej lekcji do programu, który umożliwia dalszą edycję.
- Dla potrzeb lekcji wykorzystano edytor graficzny Gimp. Jest to bezpłatny program graficzny open-source.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- Uczeń korzystając z aplikacji komputerowych, przygotowuje dokumenty i prezentacje, także w chmurze, na potrzeby rozwiązywanych problemów i własnych prac z różnych dziedzin (przedmiotów), dostosowuje format i wygląd opracowań do ich treści i przeznaczenia, wykazując się przy tym umiejętnościami: tworzenia i podstawowej obróbki cyfrowej plików multimedialnych (zdjęć, filmów);
- Uczeń zapisuje efekty swojej pracy w różnych formatach i przygotowuje wydruki;
- Uczeń rozwija umiejętności korzystania z różnych urządzeń do tworzenia elektronicznych wersji tekstów, obrazów, dźwięków, filmów i animacji;
- Uczeń poprawnie posługuje się terminologią związaną z informatyką i technologią.

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści programowe będą realizowane za pomocą uniwersalnych narzędzi dostępnych w systemach operacyjnych, przede wszystkim menedżerach okien. W miarę

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

możliwości, spowodowanej dostępnością usługi zapisu danych w chmurze, uczniowie będą korzystać z zapisu danych w środowiskach wirtualnych.

- Realizacja za pomocą uniwersalnych narzędzi, w które wyposażone są wszystkie wykorzystywane w szkołach systemy operacyjne (np. eksploratory plików).
- Treści programowe realizowane będą w programie Gimp. Jest to bezpłatny program graficzny open-source.

Uczeń:

- potrafi zaimportować gotowy rysunek do danego programu.

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna,
- praca na komputerze,
- pogadanka.

Środki dydaktyczne

- zestaw PC, laptop, komputer All in One lub stacja robocza z dostępem do Internetu, zainstalowaną przeglądarką i ustawieniami regionalnymi dla języka polskiego, zainstalowany program Gimp
- rzutnik multimedialny lub tablica multimedialna,
- e-podręcznik.

Formy pracy

- Praca indywidualna.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przebieg lekcji:

Faza I. Wprowadzenie

Nauczyciel podaje temat lekcji, przypomina gdzie na ostatniej lekcji zostały zapisane wyeksportowane pliki i w jakim formacie. Wskazuje program, w którym uczniowie będą edytować obraz: program Gimp i wyjaśnia jakie narzędzia zostaną użyte: kolory i filtry.

Faza II. Część zasadnicza - realizacja

1. Nauczyciel wykorzystując e-platformę demonstruje na tablicy multimedialnej część teoretyczną uruchamiając odpowiedni moduł i materiał multimedialny.
2. Uczniowie logują się na swoich kontach na platformie edukacyjnej, gdzie otwierają lekcję w podanym przez nauczyciela module.
3. Prezentowany materiał multimedialny, zrealizowany w oparciu o szczegółowy scenariusz (załącznik nr 1 do lekcji), oprócz waloru teoretycznego stanowi również podstawę przebiegu części praktycznej lekcji. W szczególności, uczniowie:
 - a. uruchamiają program do edycji grafiki Gimp;
 - b. importują zapisany na poprzedniej lekcji projekt i edytują go w programie, wykorzystują filtry i dostosowanie kolorów;
4. Nauczyciel objaśnia sposób pracy na lekcji – we wskazanym przez niego momencie uczniowie będą wykonywać kolejne etapy prowadzące do rozwiązania zadania. Zdolniejsi uczniowie spróbują wykonać je samodzielnie. Słabsi wykonają kolejne kroki razem z nauczycielem, w oparciu o przygotowaną do zajęć instrukcję dostępną w e-podręczniku lub materiał multimedialny. W razie potrzeby będą mogli dopytać nauczyciela. Nauczyciel na bieżąco sprawdza postępy uczniów.

Faza III. Podsumowanie

Uczniowie prezentują swoje obrazy, nauczyciel omawia je i ocenia. Kryteria oceniania:

Ocena dopuszczająca:

– uczeń importuje projekt do programu Gimp.

Ocena dostateczna:

– uczeń wykorzystał filtr albo zmianę kolorów.

Ocena dobra:

– uczeń zastosował dwa różne narzędzia/filtry/zmiany kolorów do edycji grafiki.

Ocena bardzo dobra:

– uczeń wykorzystuje i samodzielnie dobiera różne narzędzia do edycji.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, dostępnych również na platformie edukacyjnej w formie testu dla uczniów, np.:

1. Jaki format pliku importujemy do Gimpa?
 - a. **.png albo .ora**
 - b. .txt
 - c. .kra
 - d. .pdf
2. Jak importujemy obrazek w Gimpie?
 - a. przez „Zapisz”
 - b. przez „Utwórz”
 - c. przez „Eksportuj”
 - d. **przez „Otwórz”**
3. Jakie narzędzia były wykorzystane w edycji obrazu w Gimpie?
 - a. **filtry, kolory**
 - b. wypełnianie kubelkiem
 - c. tekst
 - d. ścieżki

Faza V. Zadanie do samodzielnego wykonania.

- **Dodatkowe efekty.**
 - Użyj dodatkowych filtrów i pędzli w Gimpie.

Zadanie do samodzielnego wykonania – Sztuczna Inteligencja *.

- **Wykorzystaj AI.**
 - Wykorzystując sztuczną inteligencję (z wyszukiwarki Google lub innego źródła) znajdź w jakim rozszerzeniu najlepiej zapisać plakat w Gimpie do druku. Uzasadnij. Sprawdź w programie, czy jest to możliwe i przygotuj odpowiedni plik.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 3 – Import rysunku do programu graficznego w celu edycji.

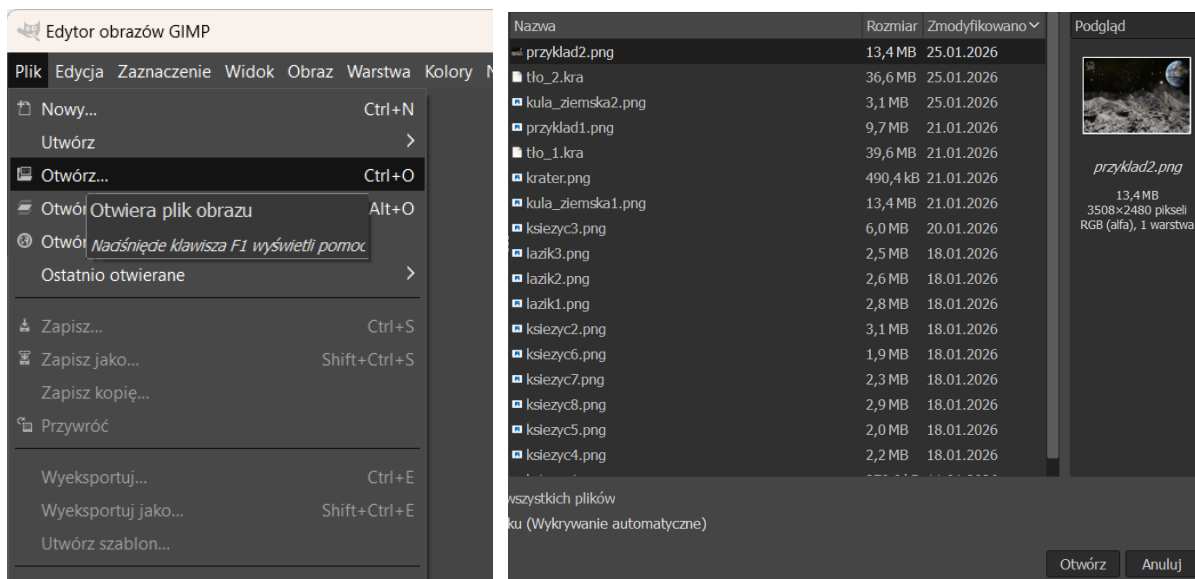
Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
- Przedstawienie uczniom celu i zakresu tematycznego lekcji.
- Uruchomienie na platformie RaP STEAM odpowiedniej e-lekcji.
- Indywidualna praca uczniów na komputerach.
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów (test na e-platformie).
- Zadanie pracy do samodzielnego wykonania – dostępne również na e-platformie (opcjonalnie).

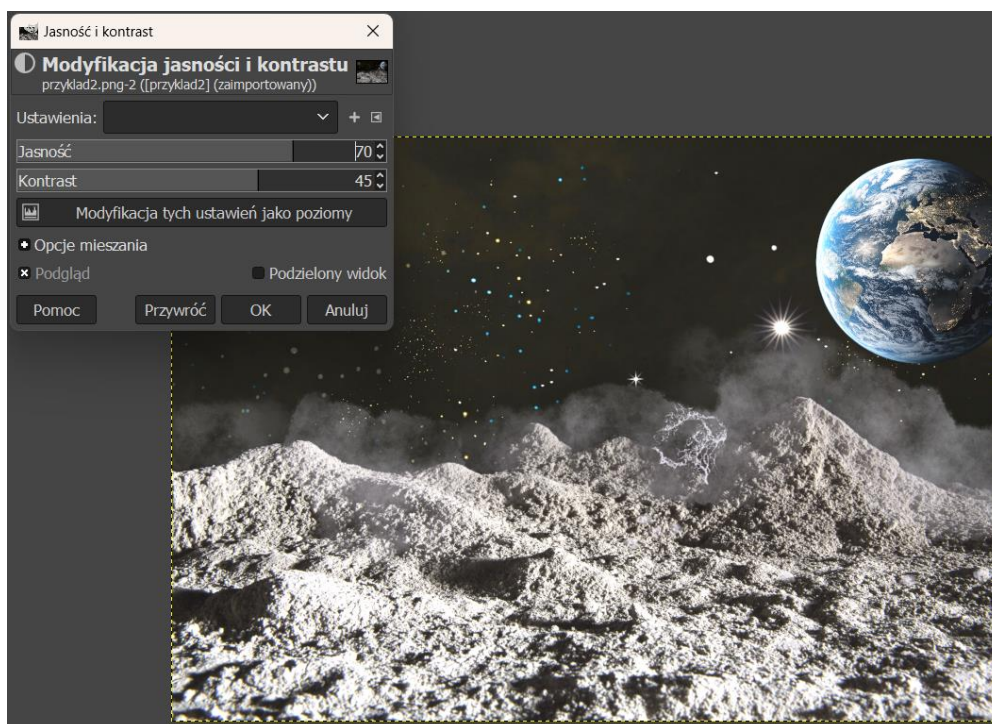
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 2 – Realizacja zadania głównego podczas lekcji nr 3 – Import rysunku do programu graficznego w celu edycji..

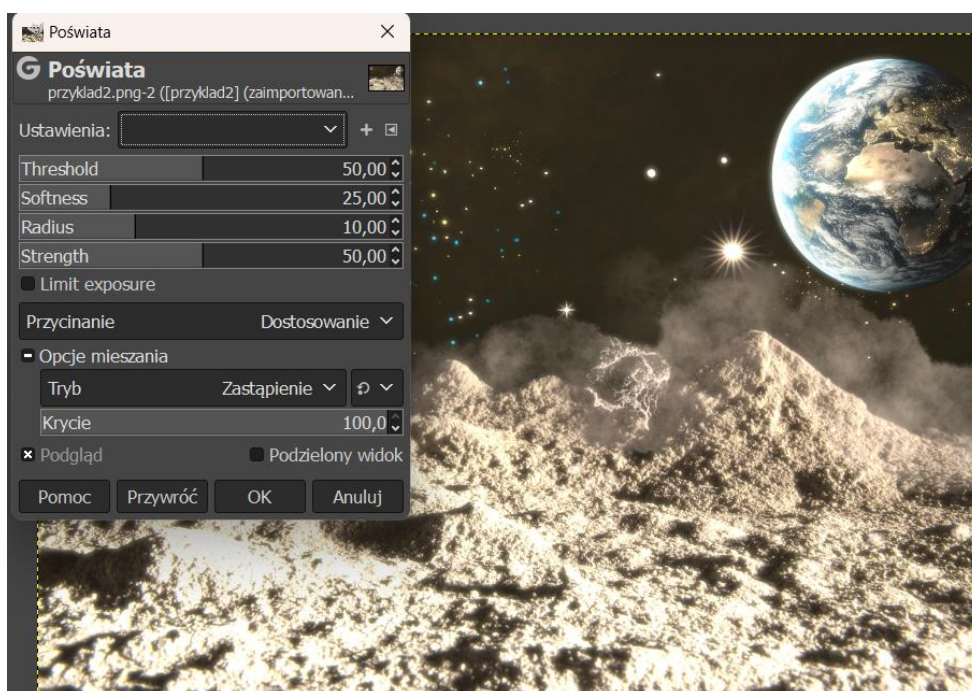
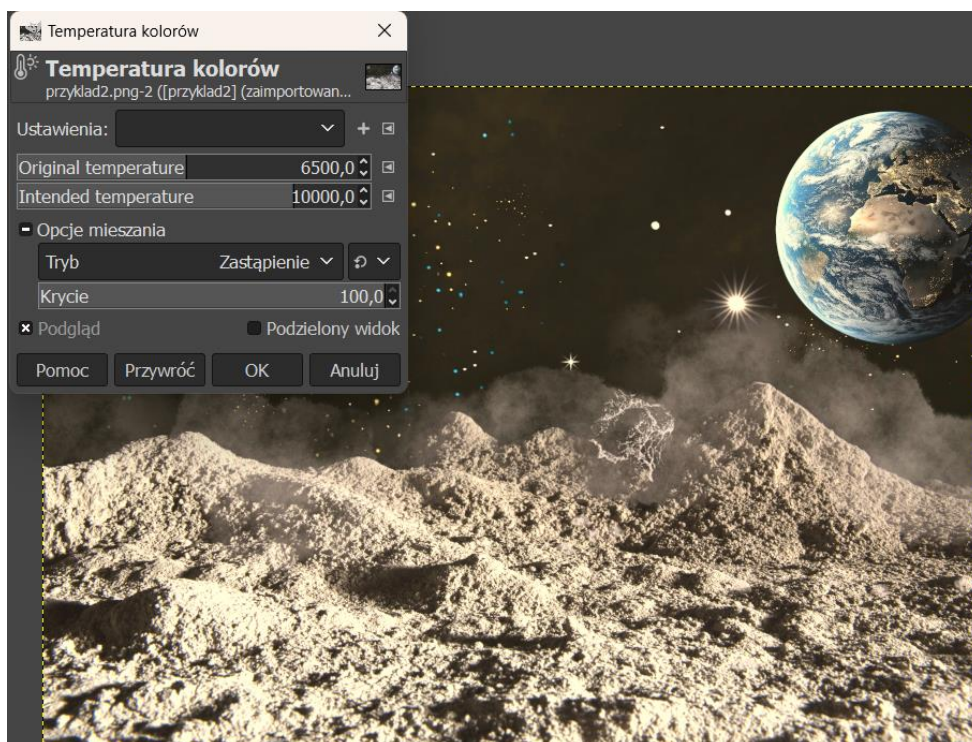
1. Otwórz program GIMP. Zimportuj rysunek z poprzedniej lekcji przez Plik->Otwórz. Pamiętaj, że GIMP nie przeczyta rozszerzenia programu Krita .kra.



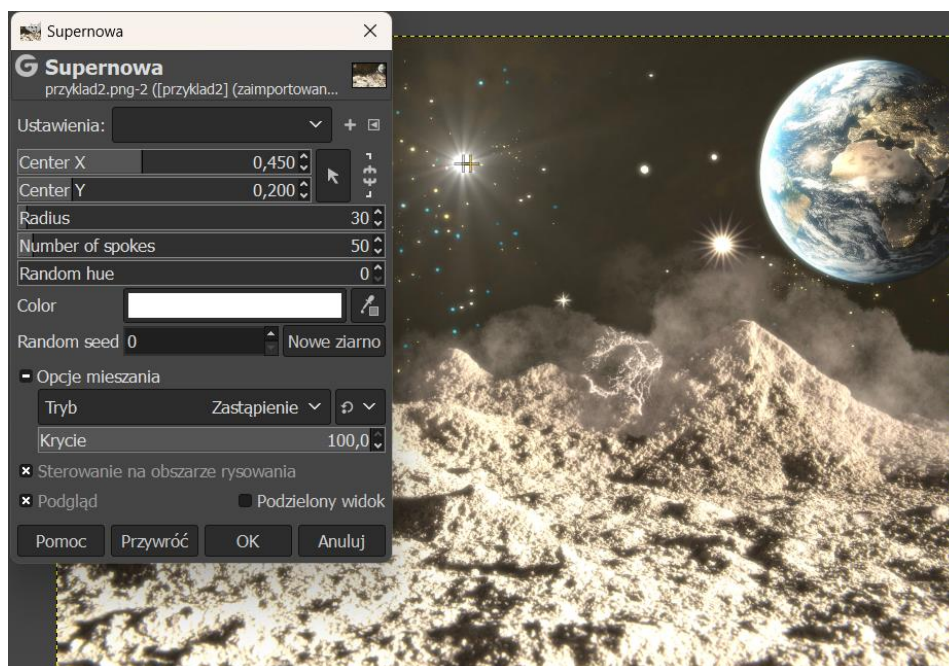
2. Zastosuj do swojego rysunku zmianę kolorów albo filtry, które pomogą uatrakcyjnić tło plakatu. Wybierz dwa różne. Przykładowe pomysły na zastosowanie filtrów:



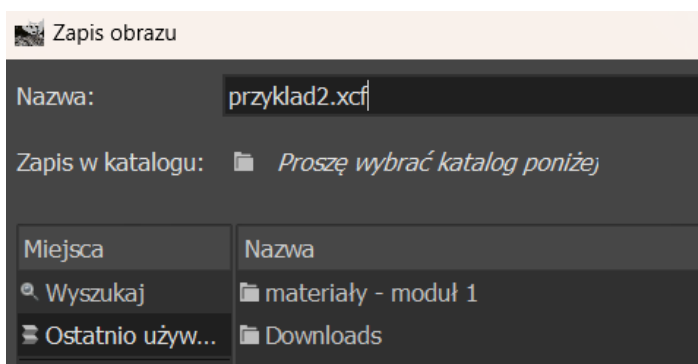
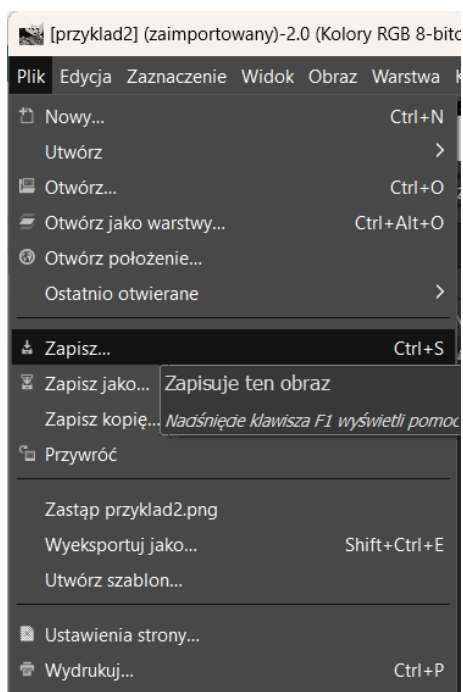
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



3. Zapisz gotowy obraz w formacie gimpa .xcf w ustalonym przez nauczyciela miejscu na dysku.



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie do samodzielnego wykonania.

- **Dodatkowe efekty.**
 - Użyj dodatkowych filtrów i pędzli w Gimpie.

Zadanie do samodzielnego wykonania – Sztuczna Inteligencja *.

- **Wykorzystaj AI.**
 - Wykorzystując sztuczną inteligencję (z wyszukiwarki Google lub innego źródła) znajdź w jakim rozszerzeniu najlepiej zapisać plakat w Gimpie do druku. Uzasadnij. Sprawdź w programie, czy jest to możliwe i przygotuj odpowiedni plik.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 1: Graficzna alchemia przyszłości

Temat lekcji nr 4 – Przygotowanie plakat nt. „Nasze roboty na Księżycu!”.

Treści nauczania (wymagania szczegółowe - podstawa programowa) – II. 3) a); II. 3) b); III. 2); III. 3);

Cel lekcji:

- Uczniowie przygotowują plakat na zadany temat.
- Dla potrzeb lekcji wykorzystano edytor graficzny Gimp. Jest to bezpłatny program graficzny open-source.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- Uczeń korzystając z aplikacji komputerowych, przygotowuje dokumenty i prezentacje, także w chmurze, na pożytek rozwiązywanych problemów i własnych prac z różnych dziedzin (przedmiotów), dostosowuje format i wygląd opracowań do ich treści i przeznaczenia, wykazując się przy tym umiejętnościami:
 - tworzenia i podstawowej obróbki cyfrowej plików multimedialnych (zdjęć, filmów),
 - tworzenia różnych dokumentów: formatuje teksty, wstawia symbole, obrazy, tabele, dłuższe dokumenty dzieli na strony;
- Uczeń rozwija umiejętności korzystania z różnych urządzeń do tworzenia elektronicznych wersji tekstów, obrazów, dźwięków, filmów i animacji;
- Uczeń poprawnie posługuje się terminologią związaną z informatyką i technologią.

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści programowe będą realizowane za pomocą uniwersalnych narzędzi dostępnych w systemach operacyjnych, przede wszystkim menedżerach okien. W miarę

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

możliwości, spowodowanej dostępnością usługi zapisu danych w chmurze, uczniowie będą korzystać z zapisu danych w środowiskach wirtualnych.

- Realizacja za pomocą uniwersalnych narzędzi, w które wyposażone są wszystkie wykorzystywane w szkołach systemy operacyjne (np. eksploratory plików).
- Treści programowe realizowane będą w programie Gimp. Jest to bezpłatny program graficzny open-source.

Uczeń:

- potrafi wstawić nowe grafiki i pola tekstowe do obrazu;
- wie jak korzystać z warstw, narzędzi i filtrów w programie graficznym.

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna,
- praca na komputerze,
- pogadanka.

Środki dydaktyczne

- zestaw PC, laptop, komputer All in One lub stacja robocza z dostępem do Internetu, zainstalowaną przeglądarką i ustawieniami regionalnymi dla języka polskiego, zainstalowany program Gimp
- rzutnik multimedialny lub tablica multimedialna,
- e-podręcznik.

Formy pracy

- Praca indywidualna.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przebieg lekcji:

Faza I. Wprowadzenie

Nauczyciel wprowadza uczniów w temat lekcji: kończymy pracę nad plakatem nt. „Nasze roboty na Księżycu”, do utworzonego wcześniej tła dodamy dzisiaj grafiki robotów i napisy. Przygotowane materiały online mają służyć jako przykładowe edycje plakatów.

Faza II. Część zasadnicza - realizacja

1. Nauczyciel wykorzystując e-platformę demonstruje na tablicy multimedialnej część teoretyczną uruchamiając odpowiedni moduł i materiał multimedialny.
2. Uczniowie logują się na swoich kontach na platformie edukacyjnej, gdzie otwierają lekcję w podanym przez nauczyciela module.
3. Prezentowany materiał multimedialny, zrealizowany w oparciu o szczegółowy scenariusz (załącznik nr 1 do lekcji), oprócz waloru teoretycznego stanowi również podstawę przebiegu części praktycznej lekcji. W szczególności, uczniowie:
 - a. edytują obraz z poprzedniej lekcji: do tła dodają warstwy z robotami oraz pola tekstowe;
 - b. wykorzystują filtry i narzędzie do edycji plakatu;
4. Nauczyciel objaśnia sposób pracy na lekcji – we wskazanym przez niego momencie uczniowie będą wykonywać kolejne etapy prowadzące do rozwiązania zadania. Zdolniejsi uczniowie spróbują wykonać je samodzielnie. Słabsi wykonają kolejne kroki razem z nauczycielem, w oparciu o przygotowaną do zajęć instrukcję dostępną w e-podręczniku lub materiał multimedialny. W razie potrzeby będą mogli dopytać nauczyciela. Nauczyciel na bieżąco sprawdza postępy uczniów.

Faza III. Podsumowanie

Uczniowie prezentują swoje obrazy, nauczyciel omawia je i ocenia. Kryteria oceniania:

Ocena dopuszczająca:

– uczeń dodaje do plakatu grafikę robota.

Ocena dostateczna:

– uczeń eksportuje plakat do wydruku (format .pdf).

Ocena dobra:

– uczeń dodaje pola tekstowe i używa filtrów.

Ocena bardzo dobra:

– uczeń wykorzystuje i samodzielnie dobiera różne narzędzia edytora do stworzenia plakatu.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, dostępnych również na platformie edukacyjnej w formie testu dla uczniów, np.:

1. W jakim formacie jest plik Gimp'a z poprzednich zajęć?
 - a. **.xcf**
 - b. .pdf
 - c. .txt
 - d. .doc
2. Jaką grafikę dodaliśmy na dzisiejszej lekcji do plakatu?
 - a. Księżyc
 - b. Kulę Ziemską
 - c. bazę
 - d. **robota**
3. Jakie narzędzia były wykorzystane do dodania napisów?
 - a. **tekst**
 - b. wypełnianie kubelkiem
 - c. filtry, kolory
 - d. ścieżki

Faza V. Zadanie do samodzielnego wykonania.

- **Własny robot.**
 - Znajdź w Internecie inną grafikę robota albo łożnika. Może to być robot, którego budujecie na lekcjach informatyki. Umieść go na plakacie.

Zadanie do samodzielnego wykonania – Sztuczna Inteligencja *.

- **Wykorzystaj AI.**
 - Zrób zdjęcie robota, którego programujecie na lekcjach i z pomocą sztucznej inteligencji usuń tło. Wkomponuj robota w grafikę plakatu.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 4 – Przygotowanie plakat nt. „Nasze roboty na Księżycu!”.

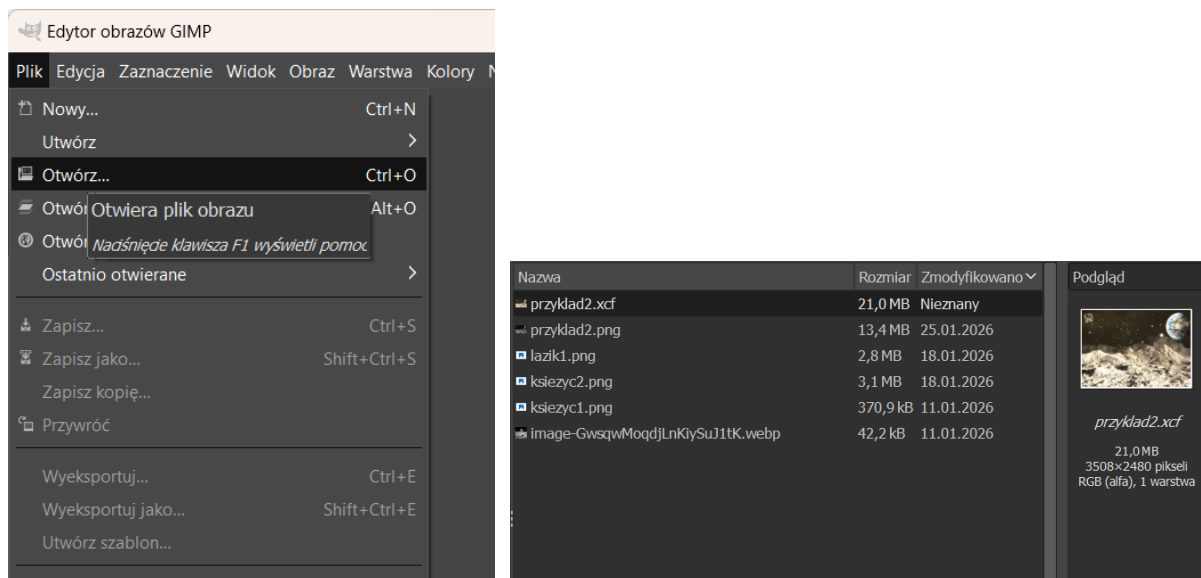
Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
- Przedstawienie uczniom celu i zakresu tematycznego lekcji.
- Uruchomienie na platformie RaP STEAM odpowiedniej e-lekcji.
- Indywidualna praca uczniów na komputerach.
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów (test na e-platformie).
- Zadanie pracy do samodzielnego wykonania – dostępne również na e-platformie (opcjonalnie).

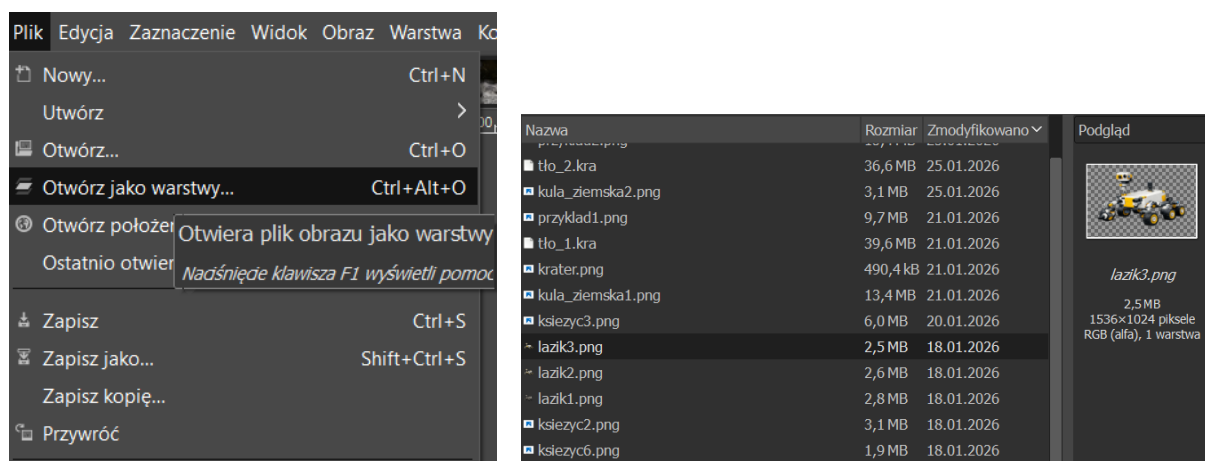
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 2 – Realizacja zadania głównego podczas lekcji nr 4 – Przygotowanie plakatu nt. „Nasze roboty na Księżycu!”.

1. Otwórz obraz zapisany na poprzedniej lekcji z rozszerzeniem .xcf.

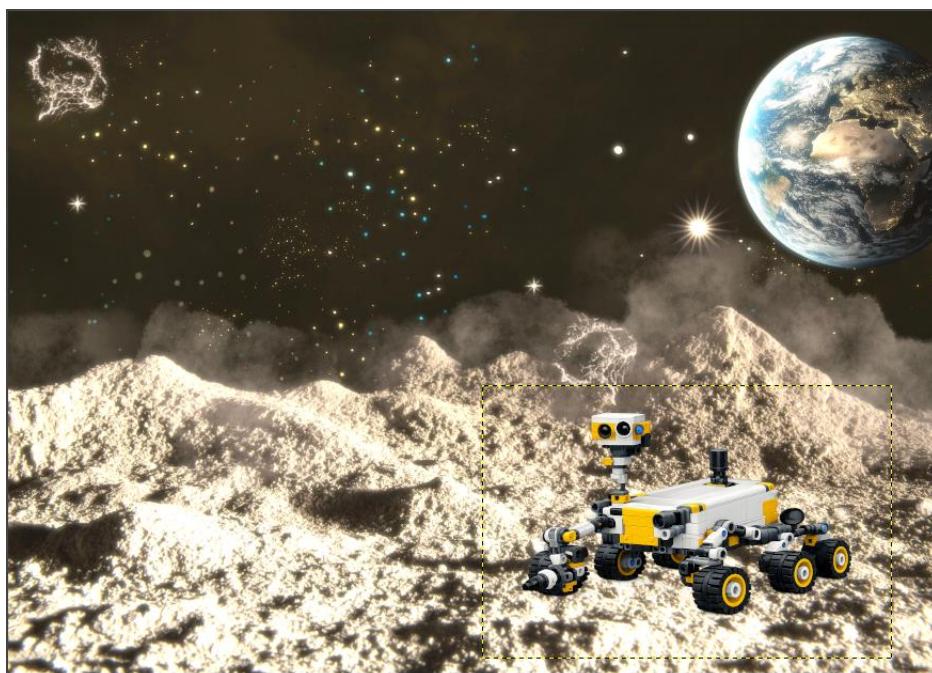


2. Wykorzystując materiały od nauczyciela dodaj do obrazu grafikę robota albo łazika marsjańskiego jako kolejną warstwę.

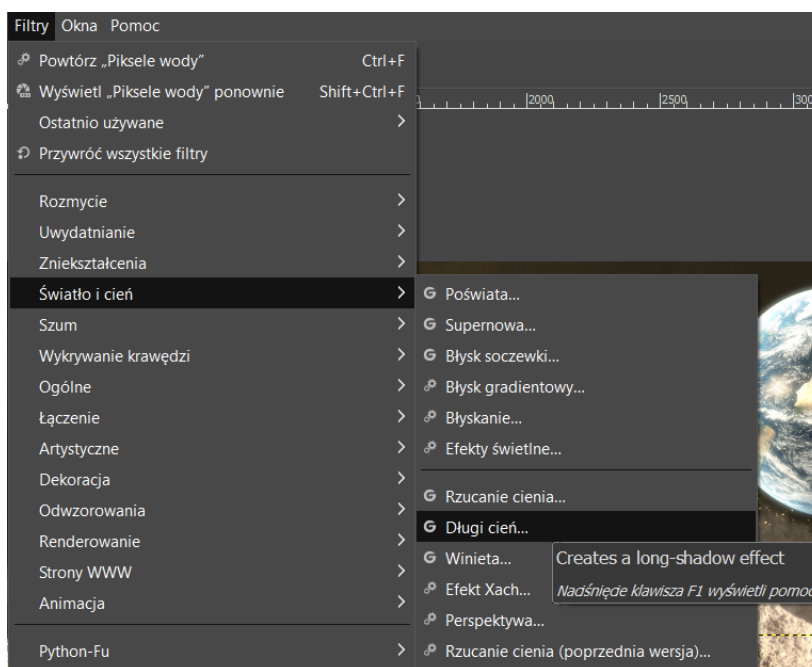


3. Przenieś łazika/robota w odpowiednie miejsce.

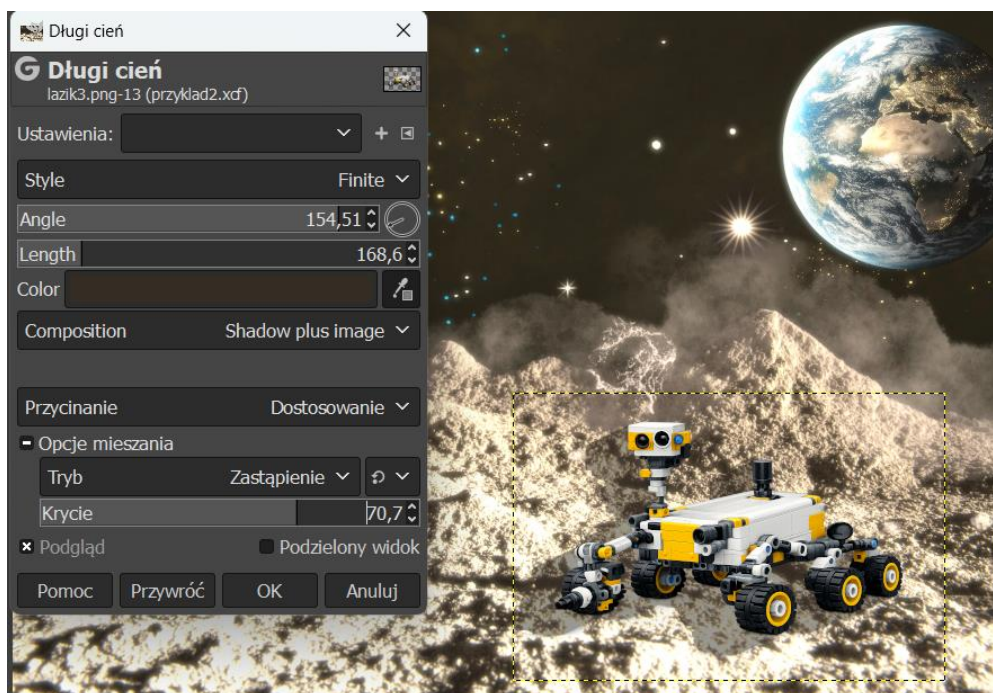
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



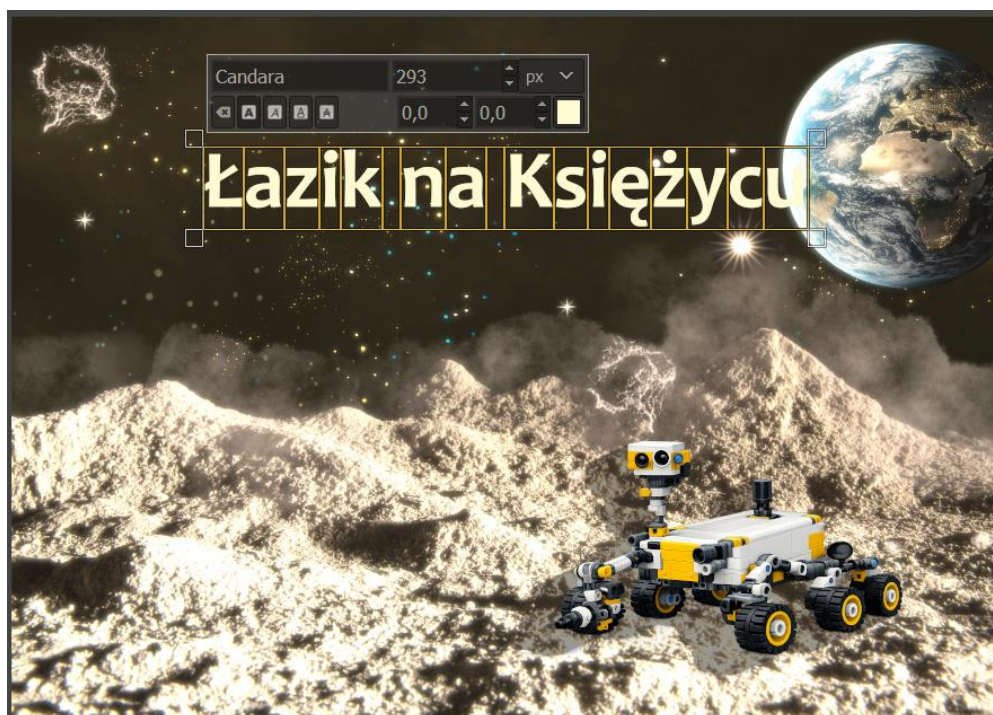
4. Aby nadać robot nie wyglądał jak naklejony, dodaj do tej warstwy Filtr -> Światło i cień -> Długi cień



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



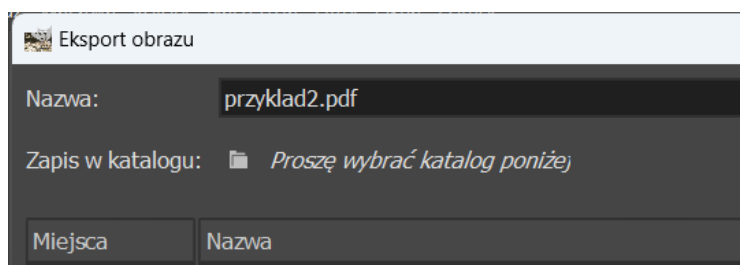
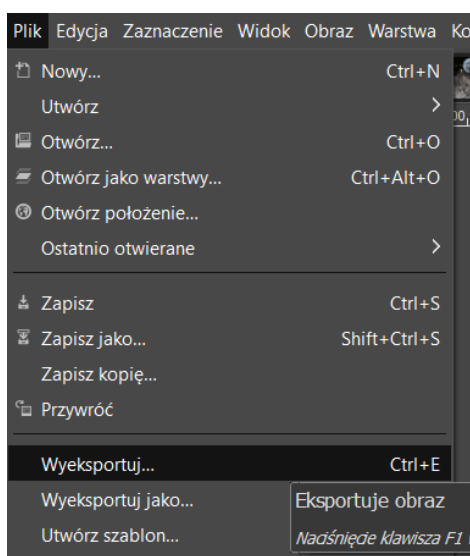
5. Dodaj pole tekstowe, w którym wpiszesz nazwę plakatu. Oraz drugie pole tekstowe, w którym podpiszesz się jako autor. Przykładowe teksty:



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



6. Zapisz gotowy plakat w formacie do wydruku .pdf w ustalonym przez nauczyciela miejscu na dysku.



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie do samodzielnego wykonania.

- **Własny robot.**
 - Znajdź w Internecie inną grafikę robota albo łożnika. Może to być robot, którego budujecie na lekcjach informatyki. Umieść go na plakacie.

Zadanie do samodzielnego wykonania – Sztuczna Inteligencja *.

- **Wykorzystaj AI.**
 - Zrób zdjęcie robota, którego programujecie na lekcjach i z pomocą sztucznej inteligencji usuń tło. Wkomponuj robota w grafikę plakatu.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 2: Odkrywamy tajemnice Kodu... Roboty w uścisku Pythona

Temat lekcji nr 5 - Wprowadzenie do języka Python. Różnice i podobieństwa języka graficznego i tekstowego

Treści nauczania (wymagania szczegółowe - podstawa programowa) – I. 1);

Cel lekcji:

- **Wprowadzenie do języka Python** - uczniowie poznają podstawowe elementy programowania tekstowego oraz zauważają podobieństwa i różnice pomiędzy językiem graficznym a tekstowym.
- **Analiza działania programu** - uczniowie analizują proste fragmenty programu zapisane w języku Python oraz określają, w jaki sposób poszczególne polecenia wpływają na działanie robota.
- **Tworzenie prostych algorytmów w języku Python** - uczniowie opracowują i zapisują proste programy sterujące ruchem robota oraz wykonaniem określonych działań.
- **Testowanie i poprawianie programu** - uczniowie uruchamiają program, obserwują działanie robota oraz wprowadzają poprawki w celu uzyskania poprawnego działania programu.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- **Uczeń dostrzega, że działanie programu i robota zależy od spełnienia określonych warunków:**
 - rozumie, że robot wykonuje polecenia zgodnie z zapisanym programem,
 - porównuje działanie robota programowanego w języku graficznym i tekstowym (Python),
 - obserwuje działanie robota i formułuje wnioski dotyczące poprawności wykonywania poleceń.
- **Uczeń analizuje działanie programu i jego efekty:**
 - sprawdza poprawność działania programu napisanego w języku Python,
 - identyfikuje błędy i modyfikuje program poprzez zmianę poleceń lub ich kolejności,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- dostrzega podobieństwa i różnice pomiędzy programowaniem blokowym a tekstowym.
- **Uczeń rozwija umiejętność testowania i doskonalenia rozwiązań:**
 - wykonuje próby działania programu i analizuje efekty pracy robota,
 - wprowadza poprawki w celu uzyskania poprawnego działania programu,
 - doskonalą program poprzez testowanie różnych rozwiązań w języku Python.

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści kształcenia będą realizowane za pomocą robotyki edukacyjnej LEGO® Education SPIKE™ Prime.

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna RaP STEAM,
- praca na komputerze,
- praca przy zestawie robotyki edukacyjnej,
- pogadanka.

Środki dydaktyczne

- zestaw PC z dostępem do Internetu,
- zestaw robotyki edukacyjnej LEGO® Education SPIKE™ Prime oraz zestaw rozszerzony LEGO® Education SPIKE™ Prime Expansion Kit,
- rzutnik multimedialny,
- e-podręcznik.

Formy pracy

- Praca indywidualna lub grupowa.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przebieg lekcji:

Faza I. Wprowadzenie

Na początku lekcji nauczyciel wprowadza uczniów w zagadnienia związane z **tekstowym językiem programowania Python**, wyjaśniając, że na wcześniejszych zajęciach uczniowie korzystali z języka graficznego Scratch, który pomagał rozwijać logiczne myślenie i rozumienie działania programu jako listy kolejnych instrukcji. Nauczyciel podkreśla, że Python pozwala zapisywać podobne polecenia w postaci tekstu, tak jak robią to programiści tworzący gry, aplikacje lub systemy komputerowe.

Następnie nauczyciel odnosi się do zadania lekcyjnego, polegającego na **poznaniu różnic i podobieństw między programowaniem graficznym a tekstowym oraz wykonaniu pierwszych prostych poleceń w języku Python w środowisku LEGO SPIKE App**.

Nauczyciel przedstawia cel i zakres zadania. Prezentuje uczniom interfejs **SPIKE App przeznaczony do pracy z Pythonem oraz porównuje go z interfejsem znanym z programowania blokowego**. Wyjaśnia, że podczas zajęć uczniowie będą korzystać z angielskiej wersji aplikacji, aby uniknąć błędów wynikających z niepełnego tłumaczenia programu.

Uczniowie zapoznają się z podstawowymi informacjami dotyczącymi języka Python, w szczególności z tym, że:

- program w Pythonie zapisuje się tekstowo, a nie za pomocą blozków,
- Python wymaga poprawnej składni, czyli dokładnego zapisu poleceń,
- małe i wielkie litery mają znaczenie, np. print i Print to różne zapisy,
- każda linijka kodu jest zazwyczaj osobnym poleceniem,
- nawiasy, cudzysłowy, dwukropki i wcięcia mają konkretne znaczenie.

Następnie nauczyciel omawia pierwsze polecenie w języku Python – funkcję print(), służącą do wyświetlania informacji w konsoli. Porównuje ją z blokiem „powiedz” znanym ze Scratcha i wyjaśnia, że tekst przeznaczony do wyświetlenia należy umieścić w nawiasie oraz w cudzysłowie, np. print("Witaj, świecie").

Następnie uczniowie zdobywają umiejętności obejmujące:

- określenie problemu: jak zapisać znane ze Scratcha polecenia w języku Python,
- porównanie programowania blokowego i tekstowego,
- analizę prostych poleceń zapisanych w Pythonie,
- uruchamianie programu i obserwowanie wyniku w konsoli,
- rozpoznawanie prostych błędów składniowych, np. braku cudzysłowu, nawiasu lub literówki,
- poprawianie kodu tak, aby program działał prawidłowo.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Na zakończenie tej fazy lekcji nauczyciel podaje temat do zapisania w zeszycie i przechodzi do fazy II – realizacji zadania głównego dotyczącego pracy z tekstowym językiem programowania Python.

Faza II. Część zasadnicza - realizacja

W tej fazie lekcji nauczyciel zapewnia uczniom praktyczne ćwiczenia, które pozwolą eksperymentować z koncepcjami programowania.

1. Nauczyciel objaśnia sposób pracy na lekcji – we wskazanym przez niego momencie uczniowie będą wykonywać kolejne etapy prowadzące do rozwiązania zadania. Zdolniejsi uczniowie spróbują wykonać je samodzielnie. Słabsi wykonają kolejne kroki razem z nauczycielem, w oparciu o przygotowaną do zajęć instrukcję dostępną w e-podręczniku lub materiał multimedialny. W razie potrzeby będą mogli dopytać nauczyciela. Nauczyciel na bieżąco sprawdza postępy uczniów.
2. Nauczyciel, wykorzystując e-platformę, demonstruje na tablicy multimedialnej działanie programu lub tłumaczy przykład.
3. Prezentowany materiał multimedialny oprócz waloru teoretycznego stanowi również podstawę przebiegu części praktycznej lekcji. W szczególności, uczniowie:
 - a. tworzą listę czynności niezbędnych do zrealizowania zadania,
 - b. na podstawie listy czynności realizują zadania przygotowane dla nich w danej lekcji;

Faza III. Podsumowanie

Uczniowie analizują przedstawione przykłady prostych programów zapisanych w języku Python, omawiają ich działanie oraz porównują sposób zapisu poleceń w języku graficznym i tekstowym. Nauczyciel ocenia stopień zrozumienia zależności między zapisanym kodem a działaniem programu po jego uruchomieniu. Kryteria oceniania:

Ocena dopuszczająca:

- uczeń rozpoznaje podstawowe elementy programu w języku Python,
- potrafi wskazać polecenie służące do wyświetlania informacji w konsoli,
- odróżnia programowanie blokowe od tekstowego.

Ocena dostateczna:

- uczeń analizuje prosty program zapisany w języku Python i opisuje jego działanie,
- rozumie zależność między zapisanym kodem a wynikiem działania programu,
- wskazuje podstawowe błędy składniowe w kodzie.

Ocena dobra:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- uczeń samodzielnie modyfikuje prosty program w języku Python,
- poprawia błędy związane z nawiasami, cudzysłowami lub wielkością liter,
- porównuje sposób zapisu poleceń w Scratchu i Pythonie.

Ocena bardzo dobra:

- uczeń samodzielnie tworzy poprawny program w języku Python,
- wyjaśnia działanie poszczególnych elementów programu,
- proponuje własne rozwiązania i potrafi uzasadnić sposób działania programu,
- samodzielnie analizuje i poprawia błędy w kodzie.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, np.:

1. Czym różni się Python od Scratcha?
 - a. **Python jest językiem tekstowym**
 - b. Python jest językiem graficznym
 - c. Python nie służy do programowania
 - d. Python nie używa instrukcji
2. Co się stanie, gdy w Pythonie popełnimy błąd w składni?
 - a. **Program nie uruchomi się i pojawi się błąd**
 - b. Robot będzie działał szybciej
 - c. Program się poprawi automatycznie
 - d. Nic się nie stanie
3. Czy Python rozróżnia wielkie i małe litery?
 - a. **Tak**
 - b. Nie
 - c. Tylko w niektórych przypadkach
 - d. Tylko przy liczbach
4. Co jest charakterystyczne dla języka Python w porównaniu do innych języków?
 - a. Kolorowe bloczki
 - b. **Wcięcia w kodzie**
 - c. Brak składni
 - d. Brak błędów
5. Do czego służy funkcja print() w Pythonie?
 - a. Do zatrzymywania robota
 - b. Do budowania robota
 - c. **Do wyświetlania tekstu**
 - d. Do zmiany koloru robota

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Faza V. Dodatkowe, praca domowa (dla chętnych).

Zadania dodatkowe (dla chętnych).

Zadanie 1: Detektyw błędów w Pythonie

Przeanalizuj poniższy program i znajdź wszystkie błędy składniowe. Następnie popraw kod tak, aby działał prawidłowo.

- `Print("Witaj")`
- `print("Ucze sie Python)`
- `print(W programowaniu wazna jest dokladnosc")`

Po poprawieniu programu odpowiedz:

- które błędy udało Ci się znaleźć,
- dlaczego program nie chciał się uruchomić,
- co należy zrobić, aby unikać takich błędów w przyszłości.

Opcjonalnie możesz rozbudować zadanie tak, aby:

- dopisać własne linijki kodu z funkcją `print()`,
- samodzielnie stworzyć program zawierający błędy dla kolegi lub koleżanki,
- porównać błędy, które mogą pojawić się w Scratchu i Pythonie,
- wyjaśnić, dlaczego Python jest bardziej wymagający pod względem składni niż programowanie blokowe.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 5 - *Wprowadzenie do języka Python. Różnice i podobieństwa języka graficznego i tekstowego.*

Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie):
 - Nauczyciel omawia **przykładowe programy zapisane w języku Python oraz nawiązuje do wcześniej poznanego środowiska Scratch**:
 - wyjaśnia, że **program w Pythonie jest zapisywany tekstowo**, a nie za pomocą blozków,
 - przedstawia przykłady prostych poleceń, takich jak **`print()`**,
 - pokazuje różnice między programowaniem graficznym i tekstowym.
 - Wprowadzenie do zadania i przypomnienie podstaw programowania w LEGO SPIKE Prime:
 - **omówienie środowiska programowania Python w aplikacji LEGO SPIKE App**,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- przedstawienie podstawowych zasad składni języka Python,
- omówienie znaczenia nawiasów, cudzysłowów, dwukropków oraz wielkości liter,
- przedstawienie zależności: polecenie w kodzie >> uruchomienie programu >> wynik działania programu.
- Analiza przykładowych programów:
 - analiza prostych poleceń wyświetlających tekst w konsoli,
 - wskazywanie błędów składniowych w kodzie,
 - porównanie przykładów zapisanych w Scratchu i Pythonie.
- Praca uczniów:
 - analiza przedstawionych przykładów programów w języku Python,
 - omawianie działania poszczególnych linijek kodu,
 - poprawianie błędów w kodzie,
 - proponowanie własnych przykładów komunikatów wyświetlanych za pomocą funkcji print().
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów (**analiza poprawności zapisu programu, rozpoznawanie błędów składniowych oraz poprawność działania programu po uruchomieniu**).
- Zadanie pracy domowej (opcjonalnie) – **napisanie prostego programu w języku Python wyświetlającego wybrane komunikaty w konsoli** oraz poprawienie przykładowego programu zawierającego błędy składniowe.

Załącznik nr 2 – Realizacja zadania głównego podczas lekcji nr 5 - Wprowadzenie do języka Python. Różnice i podobieństwa języka graficznego i tekstowego.

Zadanie 1. Przygotowanie do realizacji zadania głównego

Uczniowie, pracując indywidualnie lub w zespołach, realizują zadanie polegające na **poznaniu podstaw programowania tekstowego w języku Python oraz porównaniu go z programowaniem blokowym znanym ze Scratcha**. Szczególny nacisk położony jest na poprawny zapis poleceń, rozumienie składni oraz analizę działania programu po jego uruchomieniu.

Uczniowie opracowują i zapisują rozwiązanie problemu w postaci **prostych programów tekstowych, które umożliwią wyświetlanie komunikatów w konsoli** oraz analizowanie poprawności działania kodu. W tym celu:

- analizują przykładowe programy zapisane w języku Python,
- określają różnice pomiędzy programowaniem graficznym i tekstowym,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- określają cel programu oraz przewidują możliwe błędy (np. brak nawiasu, cudzysłowu lub użycie niepoprawnej wielkości liter),
- planują kolejność działań programu,
- opracowują algorytm w uporządkowanej formie,
- zapisują go jako prosty kod w języku Python,
- testują działanie programu i wprowadzają poprawki,
- doprecyzowują rozwiązanie tak, aby kod był czytelny i poprawny składniowo.

Uczniowie wykorzystują środowisko **LEGO SPIKE App** w trybie Python, zwracając uwagę na zgodność między zapisanym kodem a wynikiem działania programu w konsoli. Szczególną uwagę poświęcają poprawności składni, wykorzystaniu funkcji **print()** oraz analizie komunikatów o błędach pojawiających się po uruchomieniu programu.

Zadanie 2. Realizacja zadania głównego

Wprowadzenie teoretyczne:

OD KŁOCKÓW DO KODU

Twoja droga od Scratcha do Pythona

**NAJWAŻNIEJSZE**

Scratch nauczył Cię logicznego myślenia.

Program to lista instrukcji wykonywanych po kolei.

Python to język profesjonalistów!

**DZIŚ NAUCZYSZ SIĘ, JAK TŁUMACZYĆ ZNANE KONCEPCJE NA JĘZYK PYTHONA!**

Przez ostatnie lata na zajęciach z robotyki wykorzystywany był Scratch. To świetne narzędzie, które nauczyło Cię najważniejszej umiejętności programisty: logicznego myślenia. Wiesz, że program to po prostu lista instrukcji, które komputer wykonuje jedna po drugiej. Jednak profesjonalne programowanie, z którym spotykają się twórcy gier wideo, aplikacji mobilnych czy systemów bankowych, wygląda nieco inaczej. Zamiast kolorowych, gotowych klocków, używa się tekstu. Wyobraź sobie, że Scratch jest jak budowanie z gotowych paneli ściennych – szybko widać efekt, ale trudno zmienić ich kształt. Python natomiast jest jak cegły a nawet jak LEGO. Możesz z nich zbudować dokładnie to, co chcesz, bez żadnych ograniczeń narzuconych przez producenta klocków (poza kształtem). Dzisiaj nauczysz się, jak tłumaczyć znane Ci już koncepcje na język, który rozumieją dorośli programiści.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

SCRATCH – KOLOROWE KLOCKI

Uczy logicznego myślenia i podstaw programowania. Program tworzysz z gotowych bloczków.

Jak gotowe panele ściennie: szybko widać efekt, ale trudno zmienić ich kształt.

POMYŚL → **KLOCKI** → **DZIAŁA!**

PYTHON – TEKSTOWY KOD

Używany przez profesjonalistów w grach, aplikacjach, bankach i wielu innych dziedzinach. Zamiast klocków piszesz kod – masz pełną swobodę!

```
main.py
1 from spike import PrimeHub
2 from hub import runloop
3
4 async def main():
5     hub = PrimeHub()
6     print("Cześć! Jestem w Pythonie!")
7     await hub.speaker.beep()
8     await hub.motor_pair.move(10)
9
10 runloop.run(main())
```

Jak cegły lub LEGO: zbudujesz dokładnie to, co chcesz, bez ograniczeń (poza wyobraźnią).

POMYŚL → **PISZESZ KOD** → **DZIAŁA!**

KOLEJNY KROK

CO ZYSKUJESZ?
Logiczne myślenie + precyzja + konsekwencja = prawdziwy programista!

DZIŚ UCZYSZ SIĘ TŁUMACZYĆ SVOJE POMYSŁY NA JĘZYK, KTÓRY ROZUMIEJĄ DOROŚLI PROGRAMIŚCI!

W Scratchu największą zaletą było to, że bloczki do siebie „pasowały”. Nie dało się włożyć bloczka ruchu tam, gdzie powinien być bloczek dźwięku. W Pythonie masz pełną swobodę, co niesie ze sobą nowe wyzwanie: składnię. Składnia to zestaw zasad, które mówią, jak budować poprawne zdania w języku programowania. Jeśli w Scratchu zapomniano o jakimś elemencie, program po prostu działał inaczej. W Pythonie, jeśli zapomnisz o nawiasie, przecinku lub zrobisz literówkę w nazwie polecenia, komputer wyświetli komunikat o błędzie i nie uruchomi kodu.

Kolejną ważną różnicą jest to, że **Python odróżnia małe litery od wielkich**. Polecenie **print** to nie to samo co **Print**. Dla komputera to dwa zupełnie inne słowa. W Scratchu wybierano polecenia z listy, tutaj musisz stać się precyzyjny jak chirurg. Każda kropka i każdy cudzysłów mają swoje miejsce i znaczenie.

PYTHON ODRÓŻNIA MAŁE LITERY OD WIELKICH

Polecenie **print** to nie to samo co **Print**. Dla komputera to dwa zupełnie różne słowa.

W Scratchu wybierano polecenia z listy, tutaj musisz być precyzyjny jak chirurg.

Każda kropka i każdy cudzysłów mają swoje miejsce i znaczenie.

POPRAWNIE ✓

```
print("Witaj!")
```

▶ Witaj!

BŁĄD ✗

```
Print("Witaj!")
```

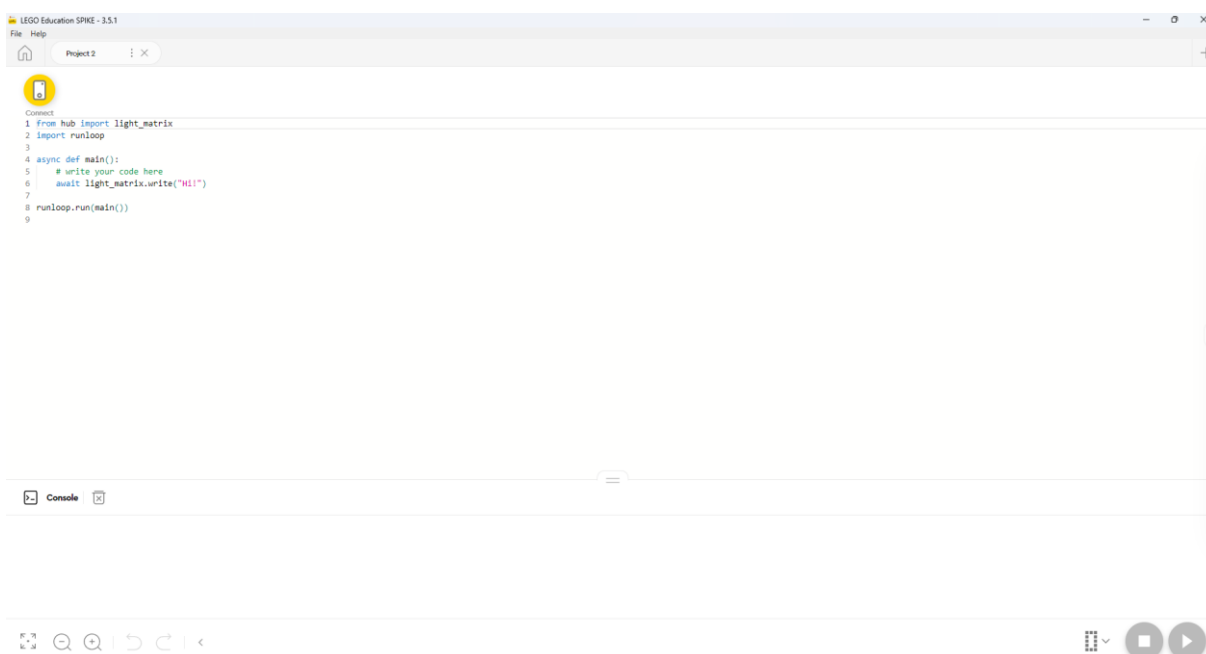
▶ NameError: name 'Print' is not defined

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Od teraz na kolejnych lekcjach będziesz używać Pythona. Interfejs aplikacji **Spike App dla Scratcha** wyglądał tak:



Dla Pythona wygląda tak:



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zauważ, że interfejs jest po angielsku. Na wszystkich lekcjach związanych z Python będziesz używać języka angielskiego! Dlaczego tak? Jest to związane z pewnym błędem w aplikacji. Nie wszystko zostało dobrze przetłumaczone i dodane, dlatego w języku polskim pojawiają się błędy, których nie powinno być. Aby uniknąć tego problemu – i nie stresować się błędami, które nie istnieją – będziesz pracować w języku angielskim.



PODSUMOWANIE: W Scratchu wybierasz bloczki z listy.
W Pythonie piszesz tekst – musisz być dokładny i znać zasady składni.



**ĆWICZ, TESTUJ, POPRAWIAJ –
TAK ZOSTAJESZ PROGRAMISTĄ!**

Język Python – ważne informacje

W języku polskim kropka kończy zdanie, a wielka litera je zaczyna. W Pythonie też mamy taką „gramatykę”, którą nazywamy składnią. Jest ona jednak znacznie bardziej rygorystyczna. Najważniejszym elementem, który odróżnia Pythona od innych programowania, są wcięcia. W innych językach programiści używają mnóstwa nawiasów klamrowych {}, aby pokazać, gdzie zaczyna się i kończy jakaś część kodu. Python stawia na przejrzystość. Jeśli chcesz, aby kilka komend wykonało się wewnątrz pętli, musisz je wszystkie przesunąć o taką samą liczbę spacji w prawo. To wymusza na programiście dbanie o porządek. Kod w Pythonie po prostu musi wyglądać estetycznie, żeby w ogóle zadziałał. Dzięki temu, patrząc na ekran, od razu widzisz strukturę programu, co w Scratchu było widoczne dzięki kształtom bloczków.



SKŁADNIA PYTHONA – MOC WCIĘĆ

W języku polskim kropka kończy zdanie, a wielka litera je zaczyna.
W Pythonie też mamy taką „gramatykę”, którą nazywamy składnią.
Python jest bardzo rygorystyczny – **najważniejsze są wcięcia**.



Kod w Pythonie musi wyglądać estetycznie, żeby w ogóle zadziałał.
Dzięki temu od razu widać strukturę programu!

W SCRATCHU

Struktura programu jest widoczna dzięki kształtom bloczków.



W PYTHONIE – WCIĘCIA ZAMIAST NAWIASÓW

ZAMIAST NAWIASÓW KLAMROWYCH {}

```
for i in range(3) {  
    move(10)  
    say("Cześć!")  
    play("bęben")  
}
```

UŻYWAMY WCIĘĆ (SPACJI)

```
for i in range(3):  
    move(10)  
    say("Cześć!")  
    play("bęben")
```

Wszystkie komendy wewnątrz pętli muszą być przesunięte o tę samą liczbę spacji w prawo.

```
1  > for i in range(3):  
2  >     move(10)  
3  >     say("Cześć!")  
4  >     play("bęben")  
5  > 4 spacje
```

DLACZEGO WCIĘCIA SĄ WAŻNE?

- ✓ **WYMÓG PYTHONA**
Bez poprawnych wcięć program się nie uruchomi.
- ✓ **PORZĄDEK W KODZIE**
Wcięcia wymuszają logiczny i czytelny kod.
- ✓ **ŁATWOŚĆ CZYTANIA**
Od razu widzisz, które linie należą do tej samej części kodu.

BŁĄD W WCIĘCIACH = BŁĄD SKŁADNI

```
for i in range(3):  
    move(10)  
    say("Cześć!")  
    play("bęben")
```

IndentationError: expected an indented block



ZAPAMIĘTAJ!

W Scratchu strukturę widzisz dzięki kształtom klocków.
W Pythonie strukturę pokazują wcięcia.
To podstawa profesjonalnego programowania!

PYTHON = PRECYZJA



Wymaga
dokładności



Docenia
porządek



Uczy dobrych
nawyków



WCIĘCIA TO KLUCZ DO CZYTELNEGO I DZIAŁAJĄCEGO KODU W PYTHONIE!

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Ważna uwaga: W Pythonie każda linijka to zazwyczaj jedno polecenie. Nie musisz stawiać średników na końcu linii (co jest złą praktyką w innych językach), **ale musisz pamiętać o dwukropkach po instrukcjach warunkowych i pętlach**. To sygnał dla komputera: „Uwaga, pod spodem będzie blok kodu, który do mnie należy!”.

Zacznij od najprostszej rzeczy – wyświetlania informacji. W Scratchu używaliśmy bloku „powiedz”. W Pythonie służy do tego funkcja **print()**. Tekst, który ma się pojawić na ekranie, zawsze musimy umieścić w nawiasach i w cudzysłowie. Dzięki temu komputer wie, że ma po prostu przepisać te słowa, a nie szukać w nich ukrytego polecenia.

Prosty przykład:



Connect

```
1 from hub import light_matrix
2 import runloop
3
4 async def main():
5     # write your code here
6     print("Witaj Swiecie")
7
8 runloop.run(main())
9
```

Zobacz, co tu się dzieje. Pojawiają się jakieś słowa typu from hub, import runloop, jakiś async def. To wszystko jest niezbędne do działania programu. Jak to działa i co to robi? Dowiesz się na kolejnych lekcjach, stosując w praktyce. Dziś skupiasz się na komendzie print. Zobacz, jak jest zbudowana:

```
print("Witaj Swiecie")
```

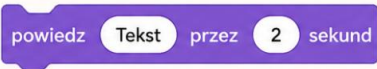
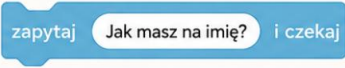
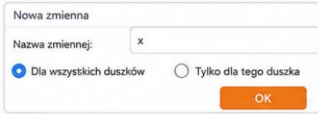
Najpierw pojawia się słowo **print**. Python rozpoznaje je jako słowo swojego języka. Oznacza to, że dla interpretera (tak nazywa się program, który zamienia kod na działanie) słowo **print** ma ważne znaczenie i nie jest tylko słowem. Po słowie print otwierany jest nawias. W jego miejscu należy coś wpisać. Jeśli nie, to nawias się zamyka.

Nawias jest wymagany i niezbędny – musi być, inaczej nie działa. Spotkasz się z takimi przypadkami.

W tym wypadku w nawiasie należy wpisać tekst, który chcesz, aby pojawił się na ekranie. Używając znaków cudzysłowu (""), podajesz tekst, który ma pojawić się w konsoli na dole w programie. Po uruchomieniu programu pojawi się tekst.

Znane Ci są już zmienne, pętle i warunki – ze Scratcha. W Pythonie poznasz je na dalszej lekcji i wtedy skupimy się na ich dokładnym wytłumaczeniu. W tabeli poniżej masz porównanie – czym różni się Scratch od Pythona.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

 Funkcja w programowaniu	 Jak to robiłeś w Scratchu?	 Jak to robisz w Pythonie?
 Wyjście danych Wyświetlanie informacji na ekranie	 Bloczek „Powiedz [Tekst]”	<pre>print("Tekst")</pre> print("Tekst")
 Pobieranie danych Pobieranie informacji od użytkownika	 Bloczek „Zapytaj i czekaj”	<pre>imie = input()</pre> imie = input()
 Zmienne Przechowywanie danych w zmiennych	 Okno tworzenia zmiennej	<pre>x = 5</pre> x = 5
 Pętla powtórz Wielokrotne wykonywanie kodu	 Bloczek „C” otaczający kod	<pre>for i in range(5):</pre> # wcięty kod for i in range(5): (plus wcięcie)
 Warunki Wykonywanie kodu, jeśli warunek jest spełniony	 Bloczek „Jeżeli... to”	<pre>if x > 5:</pre> # wcięty kod if x > 5: (plus wcięcie)

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Dlaczego Python jest dobrym wyborem?

Być może zastanawiasz się, po co uczyć się pisać, skoro układanie bloczków było prostsze. Odpowiedź tkwi w potęgze tego języka. Python jest obecnie najpopularniejszym językiem na świecie z kilku powodów. Po pierwsze, jest uniwersalny – od tworzenia prostych kalkulatorów, przez analizę zdjęć z teleskopów, aż po tworzenie zaawansowanych algorytmów sztucznej inteligencji. Po drugie, Python ma ogromną społeczność. Jeśli masz jakiś problem, miliony ludzi przed Tobą już go rozwiązało i opisało w Internecie.

Nauka Pythona to nauka porozumiewania się z maszynami w sposób bezpośredni. To, czego nauczyliście się w Scratchu, było treningiem Waszego mózgu. Teraz dostajesz do ręki prawdziwe narzędzie pracy. Od następnej lekcji zaczniesz pisać pierwsze skrypty, które nie tylko będą wyświetlać napisy, ale zaczną reagować na dane z zewnątrz i wykonywać skomplikowane obliczenia w ułamku sekundy.



DLACZEGO WARTO UCZYĆ SIĘ PYTHONA?

Od bloczków do kodu – czas na **prawdziwe możliwości!**

POTĘGA PYTHONA

Python to **najpopularniejszy** język na świecie.
Dlaczego?

1 UNIWERSALNY

Od prostych programów po sztuczną inteligencję – używany wszędzie!



kalkulatory



analiza zdjęć z teleskopów



sztuczna inteligencja

2 OGROMNA SPOŁECZNOŚĆ

Masz problem? Miliony programistów już go rozwiązało i dzieli się wiedzą w Internecie!



miliony użytkowników i gotowych rozwiązań

CO DAJE CI NAUKA PYTHONA?



UCZYSZ SIĘ KOMUNIKACJI Z MASZYNAMI

Piszesz kod, który komputer wykonuje dokładnie tak, jak mu każesz.



TRENING DLA MÓZGU (SCRATCH)

Logiczne myślenie, planowanie i rozwiązywanie problemów – to Twój fundament!



PRAWDZIWE NARZĘDZIE PRACY

Od teraz kod, który napiszesz, może reagować na dane, wykonywać obliczenia i realizować zaawansowane zadania!

CO CIĘ CZEKA?

- ✓ Pierwsze skrypty
- ✓ Reakcja na dane z zewnątrz
- ✓ Skomplikowane obliczenia
- ✓ Nowe możliwości!



PAMIĘTAJ:

Scratch dał Ci sposób myślenia. Python daje Ci moc tworzenia!
TERAZ ZACZYNA SIĘ PRAWDZIWE PROGRAMOWANIE.



SCRATCH
Trening mózgu



PYTHON
Narzędzie pracy



TWOJE
MOŻLIWOŚCI



CEL: PISAĆ KOD, KTÓRY ROZWIĄDUJE PRAWDZIWE PROBLEMY!

Gotowy na kolejny krok?
Zaczynamy przygodę z Pythonem!

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przykład realizacji zadania

Zadanie:

Znajdź błąd w poniższym kodzie. Co się stanie, kiedy go uruchomimy bez zmian?



Connect

```
1 print("Hej, co u ciebie?")
2
```

Odpowiedź:

W powyższym kodzie brakuje cudzysłowu kończącego wypisywany tekst, prawidłowo powinno wyglądać to następująco:



Connect

```
1 print("Hej, co u ciebie?")
2
```

Uruchamiając poprawiony kod uzyskamy następujący rezultat widoczny w konsoli:



Console



2:59:42 PM: Compiled

Hej, co u ciebie?

W przypadku, gdybyśmy spróbowali uruchomić kod bez poprawienia błędu to otrzymalibyśmy informację o wystąpieniu błędu, która wygląda następująco:



Console



Traceback (most recent call last):

File "Przykład", line 1

SyntaxError: invalid syntax

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie do samodzielnego rozwiązania – 01

Napisz program w Pythonie, który wyświetli twoje imię i nazwisko w konsoli. Opcjonalnie możesz dodać drugą linijkę kodu, która wyświetli w nowej linii twój ulubiony kolor.

Zadanie do samodzielnego rozwiązania – 02

Porównaj dwa poniższe programy. Czym się różnią i co zostanie wyświetlone w konsoli dla każdego z nich? Czy będą to jednakowe wartości?



Connect

```
1 print("2+8")
2
```



Connect

```
1 print(2+8)
2
```

Zadanie do samodzielnego rozwiązania – 03

Przeanalizuj poniższy kod. Opisz, co zostanie wyświetlone i w jakiej kolejności. Następnie zmień kod tak, aby komunikaty były w poprawnej kolejności logicznej.



Connect

```
1 print("Koniec")
2 print("Start")
3 print("Robot działa")
4
```

Zadanie do samodzielnego rozwiązania – 04

Przeanalizuj poniższy kod, wskaż wszystkie błędy (minimum 3), a następnie popraw kod tak, aby działał poprawnie. Wyjaśnij, dlaczego każda poprawka była konieczna.



Connect

```
1 Print("Witaj świecie")
2 print(Witaj uczniu)
3 print("Dzis uczymy sie Python)
4
```

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 2: Odkrywamy tajemnice Kodu... Roboty w uścisku Pythona

Temat lekcji nr 6 - Programowanie w języku tekstowym. Od czego zacząć?

Treści nauczania (wymagania szczegółowe - podstawa programowa) – I. 1)

Cel lekcji:

- **Wprowadzenie do programowania tekstowego** – uczniowie poznają podstawowe zasady tworzenia programów w języku Python oraz zauważają różnice między programowaniem blokowym a tekstowym.
- **Analiza działania programu** – uczniowie analizują proste fragmenty kodu w języku Python oraz określają, w jaki sposób poszczególne polecenia wpływają na działanie programu.
- **Tworzenie prostych algorytmów i programów** – uczniowie opracowują algorytmy oraz zapisują je w postaci prostych programów tekstowych z wykorzystaniem podstawowych poleceń języka Python.
- **Testowanie działania programu** – uczniowie uruchamiają program, obserwują wyniki działania w konsoli oraz wprowadzają poprawki w celu uzyskania poprawnego działania programu.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- **Uczeń dostrzega, że działanie programu zależy od poprawności zapisanych poleceń:**
 - rozumie, że komputer wykonuje instrukcje zgodnie z zapisanym kodem programu,
 - obserwuje działanie programu napisanego w języku Python i formułuje wnioski dotyczące poprawności działania poleceń.
- **Uczeń analizuje działanie programu i jego efekty:**
 - sprawdza poprawność działania prostego programu w języku Python,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- identyfikuje błędy składniowe i modyfikuje program poprzez poprawę zapisu poleceń lub ich kolejności.
- **Uczeń rozwija umiejętność testowania i doskonalenia rozwiązań:**
 - wykonuje próby działania programu i analizuje wyniki jego działania,
 - wprowadza poprawki w celu uzyskania poprawnego działania programu,
 - porównuje programowanie graficzne i tekstowe oraz dostrzega ich podobieństwa i różnice.

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści kształcenia będą realizowane za pomocą robotyki edukacyjnej LEGO® Education SPIKE™ Prime.

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna RaP STEAM,
- praca na komputerze,
- praca przy zestawie robotyki edukacyjnej,
- pogadanka.

Środki dydaktyczne

- zestaw PC z dostępem do Internetu,
- zestaw robotyki edukacyjnej LEGO® Education SPIKE™ Prime oraz zestaw rozszerzony LEGO® Education SPIKE™ Prime Expansion Kit,
- rzutnik multimedialny,
- e-podręcznik.

Formy pracy

- Praca indywidualna lub grupowa.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przebieg lekcji:

Faza I. Wprowadzenie

Na początku lekcji nauczyciel wprowadza uczniów w zagadnienia związane z **programowaniem tekstowym w języku Python**, wyjaśniając, że program komputerowy jest zapisanym zestawem instrukcji wykonywanych przez komputer w określonej kolejności. Następnie odnosi się do zadania lekcyjnego, polegającego na poznaniu podstaw języka Python oraz zapisaniu pierwszych prostych poleceń w środowisku LEGO SPIKE Prime.

Nauczyciel przedstawia cel i zakres zadania. Prezentuje uczniom **środowisko programowania w języku Python w aplikacji LEGO SPIKE App** oraz omawia podstawowe zasady składni języka, takie jak poprawny zapis poleceń, znaczenie nawiasów, cudzysłowów oraz wielkości liter. Następnie przekazuje uczniom niezbędne informacje do realizacji lekcji.

Uczniowie zapoznają się ze **sposobami zapisu prostych poleceń w języku Python oraz obserwują, jak program po uruchomieniu wykonuje kolejne instrukcje i wyświetla wyniki działania w konsoli.**

Następnie uczniowie zdobywają umiejętności obejmujące:

- określenie problemu (w jaki sposób zapisać proste polecenia w języku Python),
- analizę działania programu tekstowego,
- porównanie programowania blokowego i tekstowego,
- obserwację wyników działania programu po jego uruchomieniu,
- testowanie programu i wprowadzanie prostych poprawek związanych ze składnią kodu.

Na zakończenie tej fazy lekcji nauczyciel podaje temat do zapisania w zeszycie i przechodzi do fazy II – realizacji praktycznej.

Faza II. Część zasadnicza - realizacja

W tej fazie lekcji nauczyciel zapewnia uczniom praktyczne ćwiczenia, które pozwolą eksperymentować z koncepcjami programowania. W szczególności obejmować to będzie używanie blokowych języków programowania oraz zestawu do robotyki.

1. Nauczyciel objaśnia sposób pracy na lekcji – we wskazanym przez niego momencie uczniowie będą wykonywać kolejne etapy prowadzące do rozwiązania zadania. Zdolniejsi uczniowie spróbują wykonać je samodzielnie. Słabsi wykonają kolejne kroki razem z nauczycielem, w oparciu o przygotowaną do zajęć instrukcję dostępną w e-podręczniku lub materiał multimedialny. W razie potrzeby będą mogli dopytać nauczyciela. Nauczyciel na bieżąco sprawdza postępy uczniów.
2. Bazując na posiadanych zasobach, zapewnij każdemu uczniowi lub zespołowi uczniów kompletny zestaw do robotyki (tj. LEGO® Education SPIKE™).

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

3. Zestawy te pozwalają uczniom budować własne roboty i programować je do wykonywania określonych zadań. Nauczyciel zaczyna od prostych projektów, takich jak zbudowanie i zaprogramowanie robota, który może wykonać proste czynności, które wprawia robota w ruch.
4. Uczniowie budują bazowego robota według załączonej instrukcji (załącznik **Budowa_bazy_robota.pdf** do lekcji).
5. Uczniowie włączają program **SPIKE**, poprzez który będą programować robota.
6. Nauczyciel, wykorzystując e-platformę, demonstruje na tablicy multimedialnej działanie programu i robota, uruchamiając odpowiedni moduł i materiał multimedialny.
7. Prezentowany materiał multimedialny, zrealizowany w oparciu o szczegółowy scenariusz (załącznik do lekcji), oprócz waloru teoretycznego stanowi również podstawę przebiegu części praktycznej lekcji. W szczególności, uczniowie:
 - a. budują prosty model bazowego robota w oparciu o zestaw do robotyki (tj. LEGO® Education SPIKE™),
 - b. tworzą listę czynności niezbędnych do wprawienia robota w ruch,
 - c. na podstawie listy czynności przygotowują program, zapisują go we wskazanym folderze (nauczyciel wskazuje właściwą lokalizację; może to być dysk lokalny lub zasób w chmurze) i potrafią go uruchomić;
 - d. wgrywają program do zbudowanego modelu robota, uruchamiają, obserwują i testują działanie robota. Wprowadzają modyfikacje w programie, poprawiają ewentualne błędy.

Faza III. Podsumowanie

Uczniowie analizują przedstawione przykłady **prostych programów zapisanych w języku Python oraz omawiają ich działanie**. Nauczyciel ocenia stopień zrozumienia zależności między zapisanym kodem a wynikiem działania programu po jego uruchomieniu.

Uczniowie wspólnie wskazują **najważniejsze elementy składni języka Python oraz omawiają najczęściej pojawiające się błędy** w zapisie kodu i sposoby ich poprawiania.

Kryteria oceniania:

Ocena dopuszczająca:

- uczeń rozpoznaje podstawowe elementy programu w języku Python,
- potrafi wskazać polecenie odpowiedzialne za wyświetlanie informacji w konsoli.

Ocena dostateczna:

- uczeń analizuje przykładowy program i opisuje jego działanie,
- rozumie zależność między zapisanym kodem a działaniem programu,
- wskazuje podstawowe błędy składniowe.

Ocena dobra:

- uczeń samodzielnie modyfikuje przykładowy program,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- poprawia błędy związane z zapisem poleceń, nawiasami lub cudzysłowami,
- testuje działanie programu po wprowadzonych zmianach.

Ocena bardzo dobra:

- uczeń samodzielnie tworzy poprawny program w języku Python,
- wyjaśnia działanie poszczególnych elementów programu,
- proponuje własne rozwiązania i potrafi je uzasadnić,
- samodzielnie analizuje i poprawia błędy w kodzie.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, np.:

1. Co należy zrobić na początku programu, aby móc korzystać z funkcji robota?
 - a. **Zaimportować odpowiednie biblioteki**
 - b. Uruchomić robota
 - c. Zmienić kolor obudowy
 - d. Włączyć dźwięk
2. Co oznacza słowo await w programie?
 - a. Program się zatrzymuje na zawsze
 - b. **Program czeka na zakończenie danej operacji**
 - c. Program przyspiesza działanie
 - d. Program ignoruje polecenie
3. Jakie wartości mogą przyjmować współrzędne pikseli w matrycy?
 - a. Od 1 do 10
 - b. Od 0 do 10
 - c. **Od 0 do 4**
 - d. Dowolne liczby

Faza V. Dodatkowe, praca domowa (dla chętnych).

Zadania dodatkowe (dla chętnych).

Zadanie 1: Mój pierwszy mini-program w Pythonie

Utwórz prosty program w języku Python, który wyświetli w konsoli kilka informacji o Tobie lub o wybranym temacie.

Program powinien:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- zawierać co najmniej 3 polecenia **print()**,
- wyświetlać tekst zapisany w cudzysłowie,
- być zapisany poprawnie pod względem nawiasów i cudzysłowów,
- mieć logiczną kolejność wyświetlanych komunikatów.

Opcjonalnie możesz rozbudować program tak, aby:

- zawierał 5 lub więcej linijek kodu,
- wyświetlał krótką instrukcję dla użytkownika,
- zawierał komunikat powitalny i końcowy,
- był estetycznie zapisany i łatwy do odczytania.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 6 - *Programowanie w języku tekstowym.*

Od czego zacząć?

Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
 - Nauczyciel omawia podstawowe **zagadnienia związane z programowaniem tekstowym w języku Python:**
 - wyjaśnia, że program komputerowy jest zestawem instrukcji wykonywanych przez komputer w określonej kolejności,
 - przedstawia przykłady prostych poleceń wykorzystywanych w języku Python,
 - omawia znaczenie poprawnej składni programu.
 - Wprowadzenie do zadania i przypomnienie podstaw pracy w LEGO SPIKE Prime:
 - omówienie środowiska programowania Python w aplikacji LEGO SPIKE App,
 - przedstawienie podstawowych zasad składni języka Python (nawiasy, cudzysłowy, wielkość liter, wcięcia),
 - przedstawienie zależności: zapis kodu >> uruchomienie programu >> wynik działania programu,
 - analiza przykładowych programów wykorzystujących funkcję print().
 - Praca uczniów:
 - analiza przedstawionych przykładów programów w języku Python,
 - omawianie działania poszczególnych linijek kodu,
 - wyszukiwanie i poprawianie błędów składniowych,
 - proponowanie własnych przykładów komunikatów wyświetlanych w konsoli.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów (**analiza poprawności zapisu programu, działania kodu oraz umiejętności rozpoznawania błędów składniowych**).
- Zadanie pracy domowej (opcjonalnie) – **napisanie prostego programu w języku Python wyświetlającego komunikaty w konsoli oraz poprawienie przykładowego kodu zawierającego błędy składniowe**.

Załącznik nr 2 – Realizacja zadania głównego podczas lekcji nr 6 - Programowanie w języku tekstowym. Od czego zacząć?

Zadanie 1. Przygotowanie do realizacji zadania głównego

Uczniowie, pracując w zespołach lub indywidualnie, realizują zadanie **polegające na poznaniu podstaw programowania tekstowego w języku Python oraz zapisaniu pierwszych prostych programów w środowisku LEGO SPIKE Prime**. Szczególny nacisk położony jest na poprawny zapis poleceń, rozumienie składni oraz analizę działania programu po jego uruchomieniu.

Uczniowie **opracowują i zapisują rozwiązanie problemu w postaci prostych programów tekstowych**, które umożliwią wyświetlanie komunikatów w konsoli oraz analizowanie poprawności działania kodu. W tym celu:

- analizują przykładowe programy zapisane w języku Python,
- określają cel programu oraz przewidują możliwe błędy składniowe (np. brak nawiasu, cudzysłowu lub błędny zapis polecenia),
- planują kolejność wykonywania instrukcji w programie,
- opracowują algorytm w uporządkowanej formie,
- zapisują go jako prosty program w języku Python,
- testują działanie programu i wprowadzają poprawki,
- doprecyzowują rozwiązanie tak, aby kod był czytelny i poprawny składniowo.

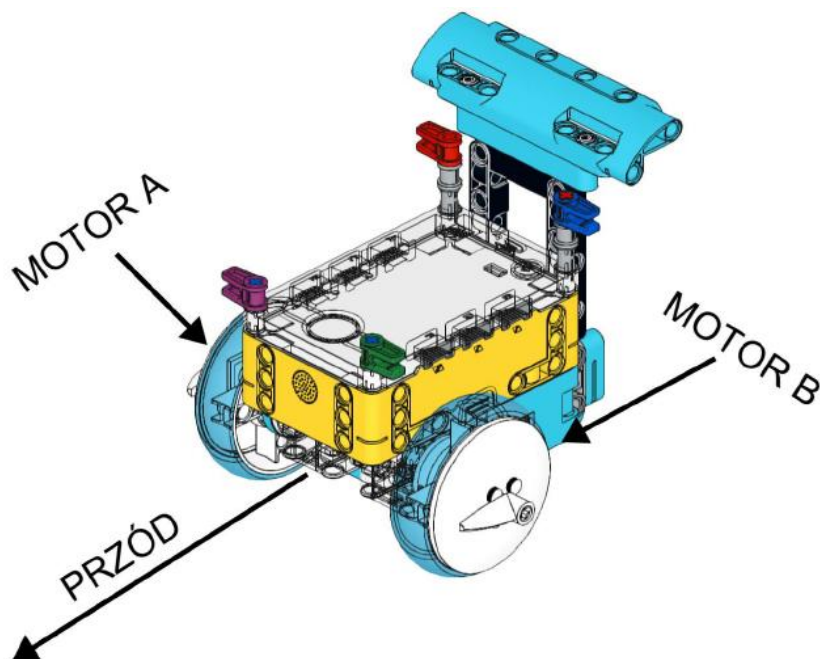
Uczniowie wykorzystują środowisko LEGO SPIKE Prime w trybie Python, zwracając uwagę na zgodność między zapisanym kodem a wynikiem działania programu widocznym w konsoli. Szczególną uwagę poświęcają poprawności składni, wykorzystaniu funkcji **print()** oraz analizie komunikatów o błędach pojawiających się po uruchomieniu programu.

Jeśli jest taka potrzeba, wskazana przez nauczyciela, to uczniowie zbudują robota od podstaw zgodnie z poniższą instrukcją.

Budowa bazowego robota

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Uruchom plik **Budowa bazy robota.pdf** znajdujący się na platformie i postępuj wg podanych w pliku kroków. Efektem końcowym będzie fizyczny robot gotowy do programowania. **Jeśli konstrukcja jest gotowa – idź dalej.**



Pudełka niezbędne do realizacji lekcji: **Brak**



Dodatkowe instrukcje niezbędne do realizacji lekcji:

- **Brak**

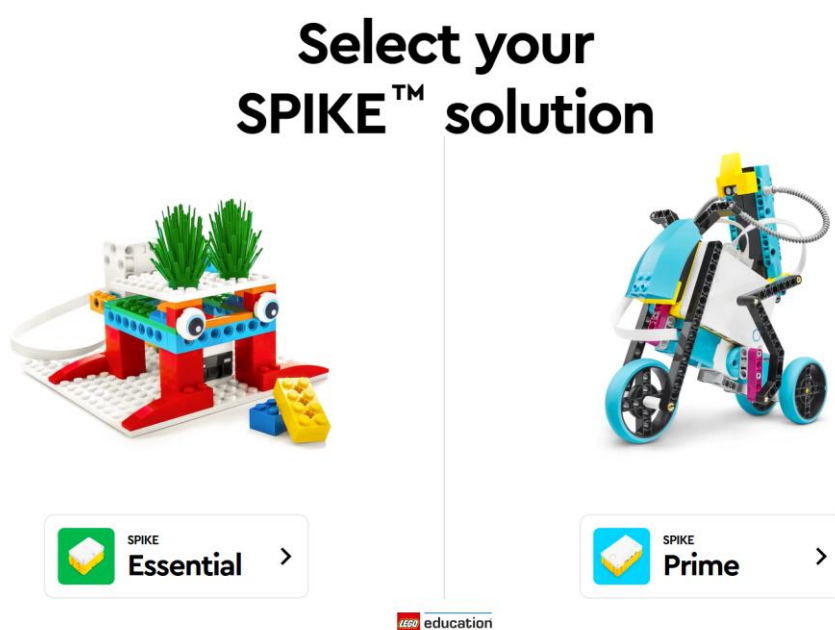
Do realizacji scenariusza zajęć będziesz potrzebował również matę:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 2. Konfiguracja środowiska.

Uruchamianie aplikacji LEGO Spike

1. Włącz przeglądarkę internetową dostępną na twoim urządzeniu
2. Wpisz w przeglądarce adres: <https://spike.legoeducation.com/> Pojawi się okno wyboru wersji LEGO SPIKE. Wybierz Spike Prime

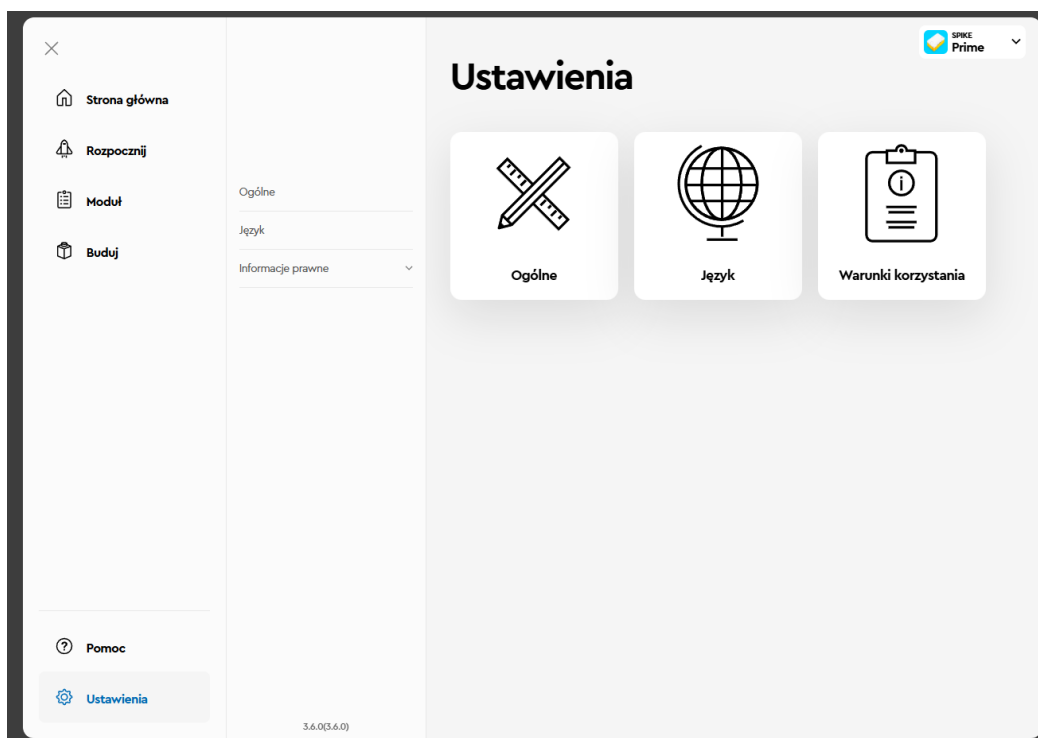


3. Po wybraniu opcji SPIKE Prime pojawia się menu główne. Jeśli występuje język polski, należy zmienić go na język angielski.

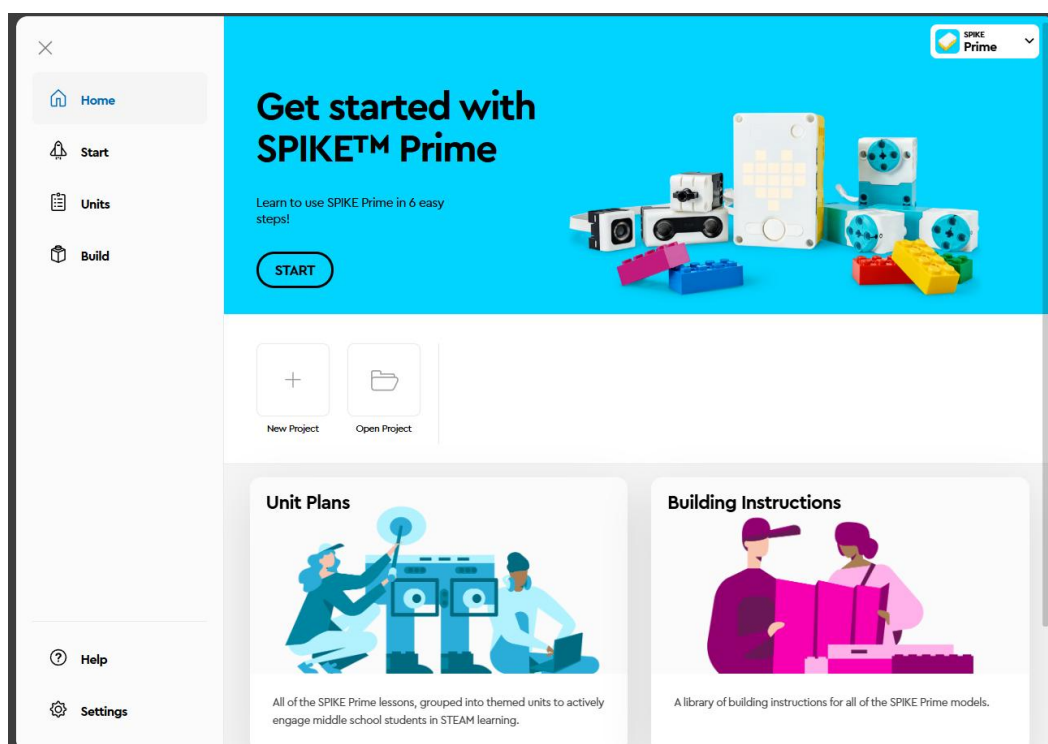
Jeżeli zaś wszystko jest w języku angielskim, to przejdź do kroku 6.

Aby zmienić język, w lewym dolnym rogu menu głównego kliknij przycisk **Ustawienia**.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

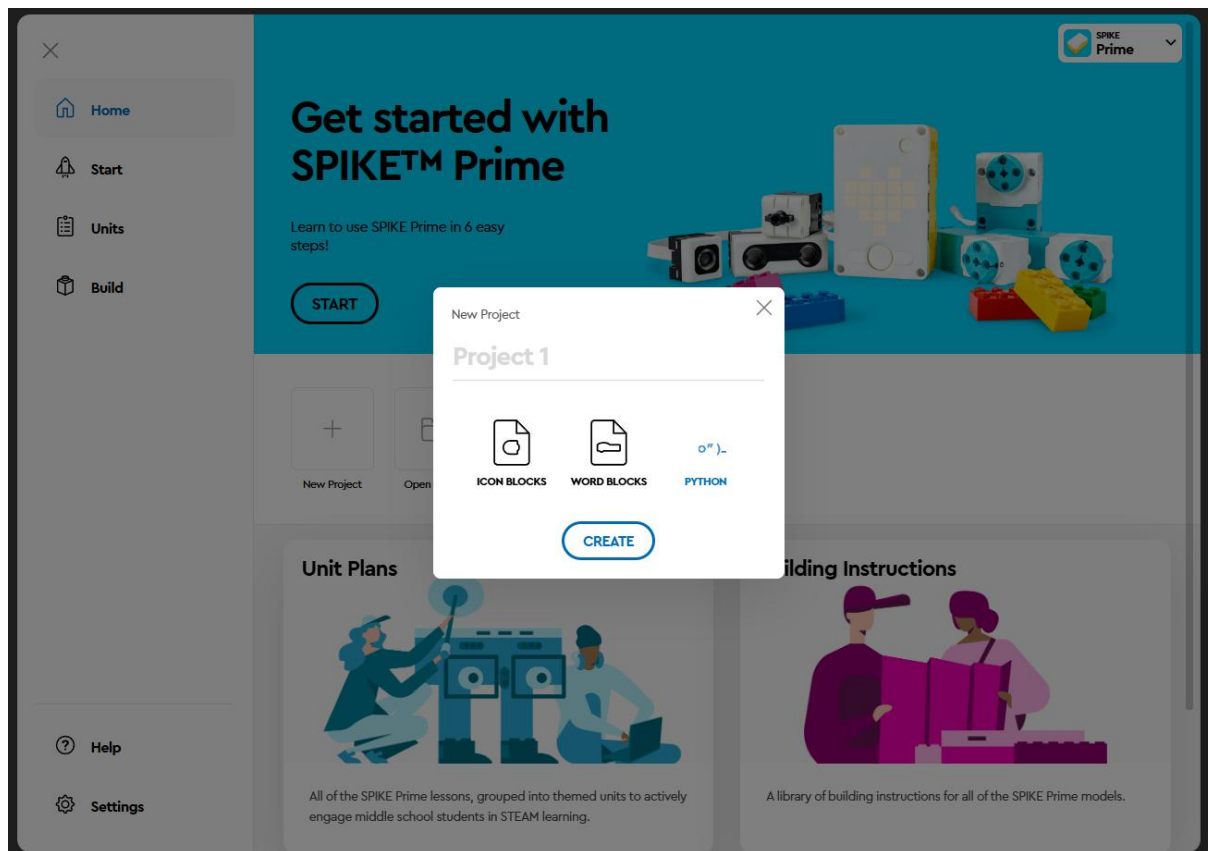


4. W nowym oknie kliknij **globus** podpisany „Language”.
5. **Wybierz język: United States (English)**. Strona przeładuje się i język angielski będzie dostępny.
6. Kliknij po lewej przycisk **Home**. Pojawi się okno jak poniżej

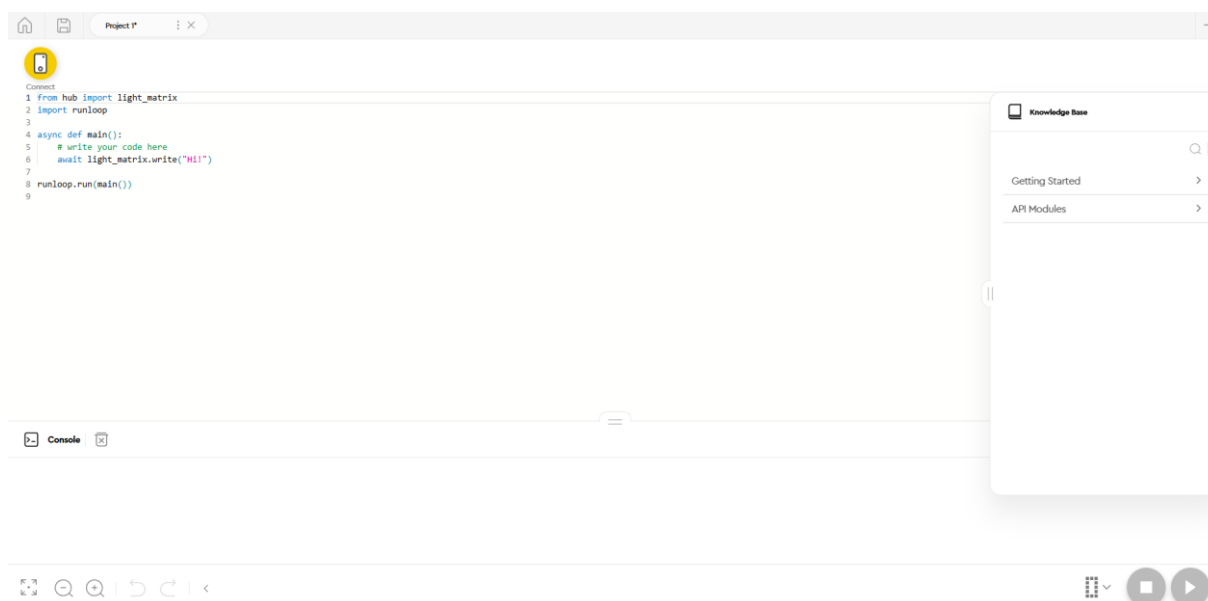


Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

7. Kliknij na środku przycisk **New Project**. Pojawi się nowe okno jak poniżej



8. Wybierz opcję **Python** oraz podaj nazwę projektu. Kiedy wszystko zostanie poprawnie wybrane, wciśnij przycisk **Create**.

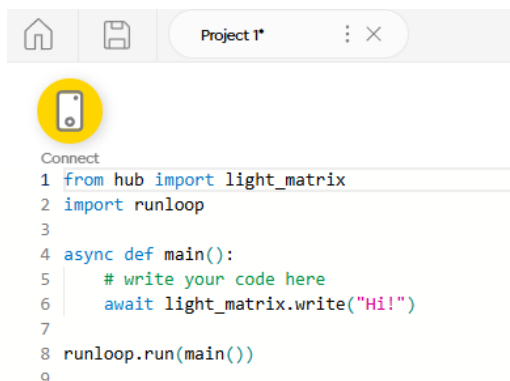


Wszystko jest gotowe do programowania. Pora na podłączenie kontrolera Spike Hub.

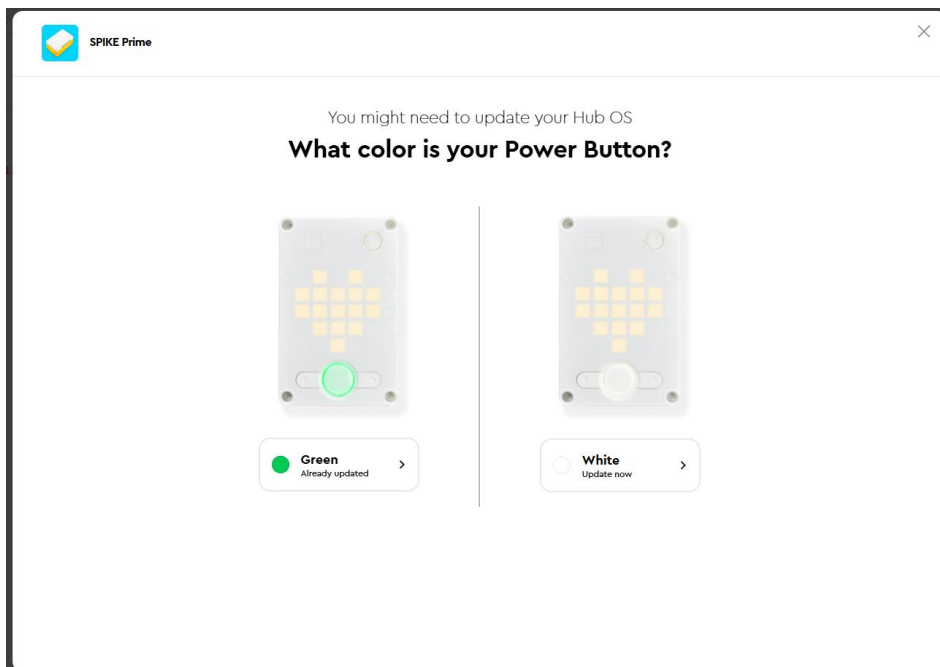
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 3. Podłączenie kontrolera Spike Hub

1. W głównym programie kliknij przycisk **Connect** znajdujący się w górnym rogu, tuż pod nazwą twojego projektu. Patrz poniżej.

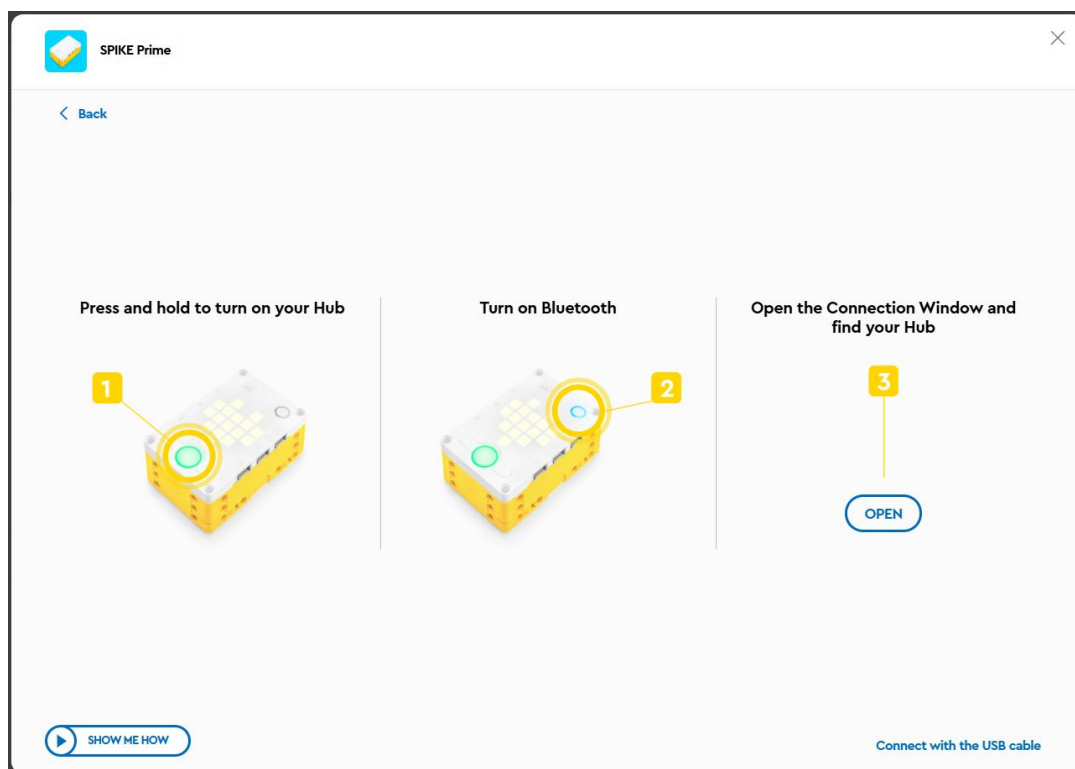


2. Pojawi się nowe okno. **Włącz kontroler Spike Hub** klikając **duży biały przycisk** i poczekaj aż się uruchomi. Następnie wybierz jedną z opcji odpowiadającą stanowi kontrolera Spike Hub – zawsze będzie to **Green – Already updated**.

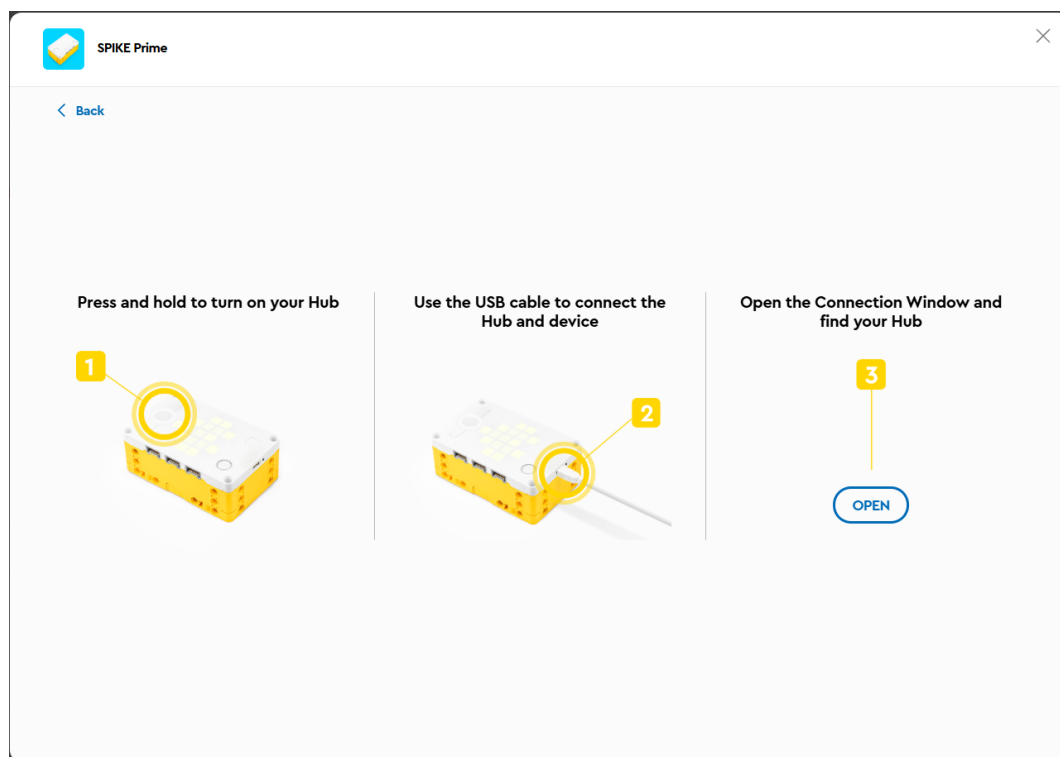


3. Pojawi się nowe okno (patrz poniżej). Wybierz opcję **Connect with USB cable** znajdującą się w prawym dolnym rogu okna.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

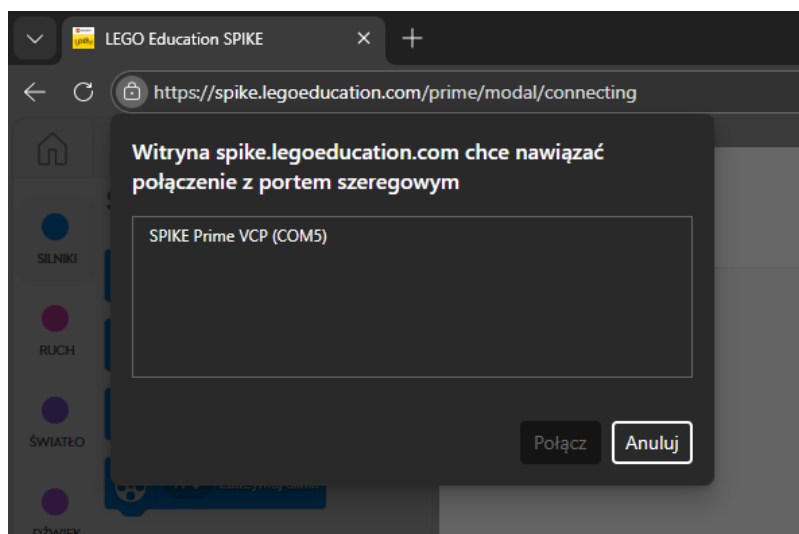


4. Postępuj zgodnie z instrukcją na ekranie. Po przeczytaniu kliknij przycisk **Open**.



5. Pojawi się nowe okno (patrz poniżej)

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



6. Wybierz opcję **SPIKE Prime VCP** i kliknij **Połącz**.

UWAGA – u Ciebie nazwa może nie być identyczna jak na rysunku powyżej. Wybierz tę opcję, która ma w nazwie Spike Prime. Innymi rzeczami się nie przejmuj.

Jeśli wszystko zostanie poprawnie połączone, to okna zostaną automatycznie zamknięte, a ikona kontrolera w programie zmieni się na zieloną.



```
1 from hub import light_matrix
2 import runloop
3
4 async def main():
5     # write your code here
6     await light_matrix.write("Hi!")
7
8 runloop.run(main())
9
```

Gratulacje. Twój Spike Hub jest podłączony do komputera. Możesz przystąpić do programowania robota.

W przypadku problemów postępuj wg kroków:

1. Sprawdź, czy kontroler Spike Hub jest włączony i świeci na zielono
 - a. Jeśli kontroler świeci na biało zgłoś to nauczycielowi
 - b. Jeśli kontroler świeci na zielono to podążaj dalej wg instrukcji łączenia powyżej
2. Sprawdź czy kabel USB jest podłączony (Spike Hub <-> Komputer)
3. Sprawdź, czy wybrano odpowiednią nazwę urządzenia – w oknie połączenia z portem szeregowym należy wybrać urządzenie, które w nazwie ma SPIKE Prime.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 4. Realizacja zadania głównego

Dzisiejsze zajęcia wprowadzają do programowania w Pythonie dla LEGO Spike. Celem lekcji jest poznanie programu oraz nauczenie się obsługi kontrolera od strony programistycznej.

- Zaprogramuj kontroler Spike Hub, aby wykonywał następujące zadania:
- Zamiana koloru przycisku zasilania na Spike Hub
- Zmiana koloru przycisku połączenia na Spike Hub
- Wyświetl tekst na kontrolerze Spike Hub
- Włącz odpowiednie piksele (światła) na wyświetlaczu Spike Hub
- Uruchom dźwięki na kontrolerze Spike Hub

Przykład realizacji zadania – wprowadzenie bibliotek

UWAGA – w materiałach stosowane są komentarze. Na zrzutach ekranu jest to tekst oznaczony jako zielony. Tego tekstu nie przepisuj! To jest tylko wytłumaczenie dla Ciebie, co dzieje się w danej części programu. Przepisuj tylko i wyłącznie tekst, który nie jest zielony. Korzystaj z tabeli. Pamiętaj, że wielkość liter MA ZNACZENIE!

Na początku programu należy dodać odpowiednie odwołania (biblioteki) do modułów kontrolera Spike Hub i czujników. Dodaj moduł:

- light,
- sound,
- color,
- time.

Aby móc skorzystać z wyświetlacza kontrolera Spike Hub, niezbędne jest wykorzystanie modułu **light_matrix** znajdującego się w **bibliotece hub**. Ta część kodu jest od razu wpisana w programie tuż po uruchomieniu. Nie należy jej zmieniać. W celu uruchomienia dźwięku na kontrolerze, a nie w komputerze, potrzebny jest moduł **sound** z **biblioteki hub**. Moduł **light** w **bibliotece hub** jest wymagany m.in. do zmiany kolorów na kontrolerze dla przycisku zasilania oraz przycisku połączenia.

Biblioteka color daje dostęp do gotowych kolorów, które są zainstalowane w ramach biblioteki do kontrolera Spike Hub. Możliwe jest wtedy ustawienie koloru wpisując tylko jego nazwę. **Biblioteka time** pozwala na stosowanie opóźnienia, czyli **bloku Czekaj** (Scratch), który poznano w poprzednich klasach.

Pozostałe elementy należy pozostawić niezmiennione. Ostatecznie górna część kodu powinna wyglądać następująco:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zdjęcie	Kod (tekst)
 <pre> 1 from hub import light_matrix, light 2 from hub import sound 3 import color 4 import runloop 5 import time </pre>	<pre> from hub import light_matrix, light from hub import sound import color import runloop import time </pre>

W tej chwili cały program powinien wyglądać następująco:

```

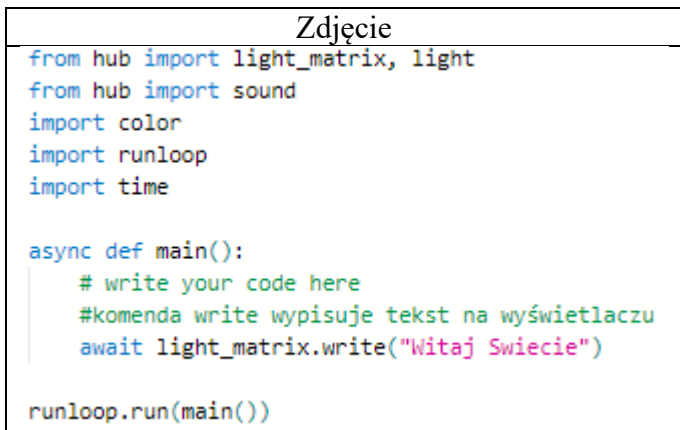
1 from hub import light_matrix, light
2 from hub import sound
3 import color
4 import runloop
5 import time
6
7 async def main():
8     # write your code here
9     #komenda write wypisuje tekst na wyświetlaczu
10    await light_matrix.write("Hi!")
11
12 runloop.run(main())

```

Moduł Matrycy

Zacznij lekcję od uruchomienia całego programu. Kliknij przycisk Play (strzałka w prawo) w prawym dolnym rogu, aby wgrać program do robota i zobaczyć co się wydarzy. Na ekranie Spike Hub po uruchomieniu powinien pojawić się przesuwany tekst: **Hi!** Gdy tekst zakończy się wyświetlać wcisnij przycisk Stop w prawym dolnym rogu.

Wprowadź swój tekst. Zastąp słowo **Hi!** swoim, np. **Witaj Swiecie**

Zdjęcie	Kod (tekst)
 <pre> from hub import light_matrix, light from hub import sound import color import runloop import time async def main(): # write your code here #komenda write wypisuje tekst na wyświetlaczu await light_matrix.write("Witaj Swiecie") runloop.run(main()) </pre>	<pre> await light_matrix.write("Witaj Swiecie") </pre>

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Usuń teraz słowo `await` i uruchom program. Zobacz, co się wydarzy. Linia kodu powinna wyglądać następująco:

```
light_matrix.write("Witaj Świecie")
```

Nie powinna pojawić się żadna różnica w działaniu programu. Przejdźmy zatem dalej. Dodaj drugą linię kodu, w której wyświetlisz słowo `Text`. Twój program powinien wyglądać następująco:

```
1 from hub import light_matrix, light
2 from hub import sound
3 import color
4 import runloop
5 import time
6
7 async def main():
8     # write your code here
9     #wypisanie Witaj Świecie na ekranie Spike Hub. Co się wydarzy?
10    light_matrix.write("Witaj Świecie")
11
12    #kolejny tekst. Jak zachowa się Spike Hub z Witaj świecie kiedy dodamy mu wypisanie słowa: Text? Sprawdź
13    light_matrix.write("Text")
14
15 runloop.run(main())
```

Przywróć teraz słowo `await` przy pierwszej linii kodu (`Witaj Świecie`). Kod w linii 10 (jak powyżej) powinien wyglądać następująco:

```
1 from hub import light_matrix, light
2 from hub import sound
3 import color
4 import runloop
5 import time
6
7 async def main():
8     # write your code here
9     #wypisanie Witaj Świecie na ekranie Spike Hub. Co się wydarzy?
10    await light_matrix.write("Witaj Świecie")
11
12    #kolejny tekst. Jak zachowa się Spike Hub z Witaj świecie kiedy dodamy mu wypisanie słowa: Text? Sprawdź
13    light_matrix.write("Text")
14
15 runloop.run(main())
```

Co się wydarzyło i jak myślisz, dlaczego tak się dzieje?

Rozwiązanie jest proste. Słowo `await` oznacza wstrzymanie wykonywania dalszej części programu, dopóki kod znajdujący się w tej samej linii nie zakończy działania. Dlaczego zatem nie widzieliśmy tekstu `Witaj Świecie` kiedy nie było tego słowa?

Kod jak najbardziej się wykonał. Zdanie `Witaj Świecie` zostało wyświetlone jednak bez słowa `await` program przeszedł dalej i od razu wyświetlił słowo `Text`. Słowo `await` będzie często się powtarzać w dalszych lekcjach, dlatego warto zrozumieć jego zasadę działania. Spróbuj użyć słowa `await` w obu liniach. Zobacz co się stanie.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

```
1 from hub import light_matrix, light
2 from hub import sound
3 import color
4 import runloop
5 import time
6
7 async def main():
8     # write your code here
9
10    #oczekiwanie na zakończenie. Zobacz co się wydarzy
11    await light_matrix.write("Witaj Świecie")
12    await light_matrix.write("Text")
13
14 runloop.run(main())
```

Sterowanie oświetleniem matrycy

Spike Hub posiada matrycę pikseli o rozmiarze $5 \times 5 = 25$ punktów świetlnych – 5 wierszy i 5 kolumn. Spróbujemy teraz zaprogramować matrycę, aby uruchomić poszczególne piksele. Wykorzystaj do tego komendę **light_matrix.set_pixel**. Twój program powinien wyglądać następująco:

```
1 from hub import light_matrix, light
2 from hub import sound
3 import color
4 import runloop
5 import time
6
7 async def main():
8     # write your code here
9
10    #włączanie pixeli na matrycy. Spike Hub posiada matrycę pikseli o rozmiarze 25 punktów świetlnych.
11    #Przykład zaświecania pikseli
12    #wartosc 0 to nr kolumny, wartosc 3 to nr wiersza. 100 to moc świecenia diody - 100%
13    light_matrix.set_pixel(0,3,100)
14
15 runloop.run(main())
```

Komenda **light_matrix.set_pixel(0,3,100)** uruchamia diodę w kolumnie 0 (w programowaniu pierwszą liczbą jest 0 a nie 1) i wierszu 3 z mocą 100% światła.

Spróbuj podmienić liczby i uruchomić inne piksele. W komendzie **light_matrix.set_pixel(a,b,c)**:

- a – może być liczbą dodatnią, całkowitą z zakresu od 0 do 4
- b - może być liczbą dodatnią, całkowitą z zakresu od 0 do 4
- c - może być liczbą dodatnią, całkowitą z zakresu od 0 do 100

Przykład (uruchamia diodę w 2 kolumnie, 4 wierszu z mocą 50%):

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

```
1 from hub import light_matrix, light
2 from hub import sound
3 import color
4 import runloop
5 import time
6
7 async def main():
8     # write your code here
9
10    #włączanie pixeli na matrycy. Spike Hub posiada matrycę pikseli o rozmiarze 25 punktów świetlnych.
11    #Przykład zaświecania pikseli
12    #wartosc 0 to nr kolumny, wartosc 3 to nr wiersza. 100 to moc świecenia diody - 100%
13    light_matrix.set_pixel(0,3,100)
14
15    #zaswiecenie diody w 2 kolumnie, 4 wierszu z mocą 50%. Co się stało z poprzednim pikselem?
16    light_matrix.set_pixel(2,4,50)
17
18 runloop.run(main())
```

Co się stało z poprzednimi pikselami?

Pozostały. To, co zostało zaświecone, dopóki nie wywołana zostanie odpowiednia komenda, wciąż będzie się świecić. Aby wyłączyć wszystkie piksele, stosuje się komendę clear. Niestety, zadziała to natychmiastowo. Żeby móc zobaczyć wynik, należy wstrzymać działanie programu na parę sekund. Służy do tego funkcja **time.sleep_ms** z biblioteki **time**. Spróbuj wykonać następujący kod:

```
1 from hub import light_matrix, light
2 from hub import sound
3 import color
4 import runloop
5 import time
6
7 async def main():
8     # write your code here
9
10    #włączanie pixeli na matrycy. Spike Hub posiada matrycę pikseli o rozmiarze 25 punktów świetlnych.
11    #Przykład zaświecania pikseli
12    #wartosc 0 to nr kolumny, wartosc 3 to nr wiersza. 100 to moc świecenia diody - 100%
13    light_matrix.set_pixel(0,3,100)
14
15    #zaswiecenie diody w 2 kolumnie, 4 wierszu z mocą 50%. Co się stało z poprzednim pikselem?
16    light_matrix.set_pixel(2,4,50)
17
18    #czekanie przez 5 sekund aby zobaczyć piksele
19    time.sleep_ms(5000)
20
21    #komenda clear wyłącza piksele
22    light_matrix.clear()
23
24    #czekanie przez 1 sekundę aby zobaczyć że piksele się wyłączyły
25    time.sleep_ms(1000)
26
27 runloop.run(main())
```

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Kod w formie tekstowej:

```
from hub import light_matrix, light
from hub import sound
import color
import runloop
import time
```

```
async def main():
```

```
    # write your code here
```

```
    #włączanie pixeli na matrycy. Spike Hub posiada matrycę pikseli o rozmiarze 25 punktów
    #światlnych.
```

```
    #Przykład zaświecania pikseli
```

```
    #wartosc 0 to nr kolumny, wartosc 3 to nr wiersza. 100 to moc świecenia diody - 100%
```

```
    light_matrix.set_pixel(0,3,100)
```

```
    #zaswiecenie diody w 2 kolumnie, 4 wierszu z mocą 50%. Co się stało z poprzednim
    #pikselem?
```

```
    light_matrix.set_pixel(2,4,50)
```

```
    #czekanie przez 5 sekund aby zobaczyć piksele
```

```
    time.sleep_ms(5000)
```

```
    #komenda clear wyłącza piksele
```

```
    light_matrix.clear()
```

```
    #czekanie przez 1 sekundę aby zobaczyć że piksele się wyłączyły
```

```
    time.sleep_ms(1000)
```

```
runloop.run(main())
```

Zauważ, co się dzieje na kontrolerze Spike Hub. Wymień się informacjami z uczniem obok.

Możliwe jest uruchomienie wszystkich pikseli jednocześnie. Należy wtedy użyć funkcji show. Przepisz (lub skopiuj) kod poniżej i uruchom go. Zobacz co się wydarzy.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

```
from hub import light_matrix, light
from hub import sound
import color
import runloop
import time
```

```
async def main():
    # write your code here

    #uruchamiamy wszystkie piksele jedną komendą
    #100 - oznacza moc świecenia, 25 oznacza liczbę diod (5x5)
    pixels = [100] * 25

    #komenda show wysyła informacje do Spike Hub aby pokazać piksele
    light_matrix.show(pixels)
```

```
runloop.run(main())
```

Spike Hub umożliwia także wyświetlanie zdjęć, które posiada w programie. W tej części kodu wyświetlimy jakiś obrazek. Zamień kod. Całość powinna wyglądać jak na zdjęciu poniżej.

```
1 from hub import light_matrix, light
2 from hub import sound
3 import color
4 import runloop
5 import time
6
7 async def main():
8     # write your code here
9
10    #wyswietlanie obrazka zapisanego w Spike Hub
11    light_matrix.show_image(3)
12
13
14 runloop.run(main())
```

Co się wyświetliło?

Spróbuj wyświetlić zdjęcia, używając cyfr z zakresu od 0 do 5. Nie musisz wszystkich.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zamiana kolorów przycisków na Spike Hub

Aby zmienić kolor podświetlenia przycisków, należy wpisać kod, który znajduje się poniżej, a następnie uruchomić program. Kolory zarówno przycisku zasilania, jak i przycisku połączenia Bluetooth powinny zmienić się na te, które ustawiono w kodzie. Dzięki bibliotece **light** możliwa jest zmiana koloru, zaś dzięki bibliotece **color** możliwe jest wybranie konkretnego koloru z nazwy.

```
1 from hub import light_matrix, light
2 from hub import sound
3 import color
4 import runloop
5 import time
6
7 async def main():
8     # write your code here
9
10    #zmienia kolor przycisku zasilania na Spike Hub. Spróbuj z innymi kolorami
11    light.color(light.POWER, color.MAGENTA)
12    #zmienia kolor przycisku połączenia Bluetooth na Spike Hub. Spróbuj z innymi kolorami
13    light.color(light.CONNECT, color.GREEN)
```

Kod w formie tekstowej:

```
from hub import light_matrix, light
from hub import sound
import color
import runloop
import time
```

```
async def main():
```

```
    # write your code here
```

```
    #zmienia kolor przycisku zasilania na Spike Hub. Spróbuj z innymi kolorami
```

```
    light.color(light.POWER, color.MAGENTA)
```

```
    #zmienia kolor przycisku połączenia Bluetooth na Spike Hub. Spróbuj z innymi kolorami
```

```
    light.color(light.CONNECT, color.GREEN)
```

```
runloop.run(main())
```

Uruchom program. Kolory powinny zostać zamienione.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Obsługa głośniczka w Spike Hub

Spike Hub może wydawać dźwięk. Nie będzie to jakość, jaką prezentują dzisiejsze głośniki. Nie zagra on też muzyki z radia. Jednak dzięki zmianie krótkich dźwięków możliwe jest generowanie krótkich sygnałów dźwiękowych. Spróbuj. Wpisz do programu kod przedstawiony poniżej.

```
1 from hub import light_matrix, light
2 from hub import sound
3 import color
4 import runloop
5 import time
6
7 async def main():
8     # write your code here
9     #komenda write wypisuje tekst na wyświetlaczu
10    #await light_matrix.write("Hi!")
11
12    #uruchomienie głośnika w Spike Hub. 440 to częstotliwość dźwięku,
13    #1000 to czas w milisekundach (1000 milisekund to 1 sekunda), 100 to głośność w procentach czyli 100% głośności
14    await sound.beep(440,1000,100)
15
16 runloop.run(main())
```

Kod w formie tekstowej:

```
from hub import light_matrix, light
from hub import sound
import color
import runloop
import time
```

```
async def main():
    # write your code here
    #komenda write wypisuje tekst na wyświetlaczu
    #await light_matrix.write("Hi!")

    #uruchomienie głośnika w Spike Hub. 440 to częstotliwość dźwięku,
    #1000 to czas w milisekundach (1000 milisekund to 1 sekunda), 100 to głośność w
    procentach czyli 100% głośności
    await sound.beep(440,1000,100)
```

```
runloop.run(main())
```

Zaraz po uruchomieniu będzie słycać dźwięk, dlatego miej robota blisko siebie. Jeśli nie usłyszysz dźwięku, to włącz program jeszcze raz, będąc bardzo blisko robota.

Z dźwiękami można się bawić. Kluczowe jest tu użycie słowa `await` – znasz już jego działanie i cel, dlatego pamiętaj, aby je stosować. Poniżej masz przykład z różnymi dźwiękami. Spróbuj.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

```
1 from hub import light_matrix, light
2 from hub import sound
3 import color
4 import runloop
5 import time
6
7 async def main():
8     # write your code here
9     #komenda write wypisuje tekst na wyświetlaczu
10    #await light_matrix.write("Hi!")
11
12    #uruchomienie głośnika w Spike Hub. 440 to częstotliwość dźwięku,
13    #1000 to czas w milisekundach (1000 milisekund to 1 sekunda), 100 to głośność w procentach czyli 100% głośności
14    await sound.beep(440,1000,100)
15
16    #testy dźwięku
17    await sound.beep(200, 500, 75)
18    await sound.beep(100, 500, 75)
19    await sound.beep(400, 500, 75)
20    await sound.beep(1000, 500, 75)
21
22 runloop.run(main())
```

Kod w formie tekstu:

```
from hub import light_matrix, light
from hub import sound
import color
import runloop
import time
```

```
async def main():
    # write your code here
    #komenda write wypisuje tekst na wyświetlaczu
    #await light_matrix.write("Hi!")
```

```
#uruchomienie głośnika w Spike Hub. 440 to częstotliwość dźwięku,
#1000 to czas w milisekundach (1000 milisekund to 1 sekunda), 100 to głośność w
procentach czyli 100% głośności
await sound.beep(440,1000,100)
```

```
#testy dźwięku
await sound.beep(200, 500, 75)
await sound.beep(100, 500, 75)
await sound.beep(400, 500, 75)
await sound.beep(1000, 500, 75)
```

```
runloop.run(main())
```

Pobaw się wartościami i milej zabawy!

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Możliwe rozszerzenia zadania

1. Ustaw na wyświetlaczu inne grafiki. Sprawdź zawartość funkcji **show_image**
2. Uruchom piksele tak aby pokazywały literę X. Nie korzystaj gotowego obrazka, nie wypisuj litery X. Użyj zapalania pikseli

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 2: Odkrywamy tajemnice Kodu... Roboty w uścisku Pythona

Temat lekcji nr 7 - Kodujemy ruch naszego robota. Jak korzystać z bibliotek kodu?

Treści nauczania (wymagania szczegółowe - podstawa programowa) – I. 4); II. 2);

Cel lekcji:

- **Wprowadzenie do programowania ruchu robota** – uczniowie poznają podstawowe zasady sterowania ruchem robota w języku Python oraz **uczą się korzystania z gotowych bibliotek** kodu w LEGO SPIKE Prime.
- **Analiza działania programu** – uczniowie analizują fragmenty kodu odpowiedzialne za ruch robota oraz określają, w jaki sposób poszczególne polecenia wpływają na kierunek i sposób poruszania się robota.
- **Tworzenie prostych algorytmów i programów sterujących ruchem** – uczniowie opracowują algorytmy oraz zapisują je w postaci programów tekstowych wykorzystujących podstawowe polecenia sterujące ruchem robota i biblioteki kodu.
- **Testowanie działania programu** – uczniowie uruchamiają program, obserwują sposób poruszania się robota oraz wprowadzają poprawki w celu uzyskania poprawnego działania programu i precyzyjnego ruchu robota.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- **Uczeń dostrzega, że działanie programu i robota zależy od poprawności zapisanych poleceń:**
 - rozumie, że robot wykonuje instrukcje zgodnie z zapisanym kodem programu oraz wykorzystaniem odpowiednich bibliotek,
 - obserwuje działanie programu sterującego ruchem robota i formułuje wnioski dotyczące poprawności działania poleceń.
- **Uczeń analizuje działanie programu i jego efekty:**
 - sprawdza poprawność działania programu sterującego ruchem robota w języku Python,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- identyfikuje błędy składniowe i logiczne oraz modyfikuje program poprzez zmianę poleceń lub ich kolejności,
- **korzysta z gotowych bibliotek kodu umożliwiających sterowanie robotem.**
- **Uczeń rozwija umiejętność testowania i doskonalenia rozwiązań:**
 - wykonuje próby działania programu i analizuje sposób poruszania się robota,
 - wprowadza poprawki w celu uzyskania poprawnego działania robota,
 - **testuje różne rozwiązania związane ze sterowaniem ruchem robota i wykorzystaniem bibliotek kodu.**

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści kształcenia będą realizowane za pomocą robotyki edukacyjnej LEGO® Education SPIKE™ Prime.

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna RaP STEAM,
- praca na komputerze,
- praca przy zestawie robotyki edukacyjnej,
- pogadanka.

Środki dydaktyczne

- zestaw PC z dostępem do Internetu,
- zestaw robotyki edukacyjnej LEGO® Education SPIKE™ Prime oraz zestaw rozszerzony LEGO® Education SPIKE™ Prime Expansion Kit,
- rzutnik multimedialny,
- e-podręcznik.

Formy pracy

- Praca indywidualna lub grupowa.

Przebieg lekcji:

Faza I. Wprowadzenie

Na początku lekcji nauczyciel wprowadza uczniów w zagadnienia związane z programowaniem ruchu robota w języku Python, wyjaśniając, że robot wykonuje polecenia zapisane w programie w określonej kolejności. Następnie odnosi się do zadania lekcyjnego, polegającego na **wykorzystaniu gotowych bibliotek kodu do sterowania ruchem robota w środowisku LEGO SPIKE Prime.**

Nauczyciel przedstawia cel i zakres zadania. Prezentuje uczniom środowisko programowania w języku Python w aplikacji LEGO SPIKE App oraz **omawia podstawowe elementy bibliotek**

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

wykorzystywanych do sterowania robotem, takie jak polecenia odpowiedzialne za ruch, zatrzymanie oraz zmianę kierunku jazdy. Następnie przekazuje uczniom niezbędne informacje do realizacji lekcji.

Uczniowie zapoznają się ze sposobami zapisu poleceń sterujących ruchem robota w języku Python oraz obserwują, jak robot po uruchomieniu programu wykonuje kolejne instrukcje i porusza się zgodnie z zapisanym kodem.

Następnie uczniowie zdobywają umiejętności obejmujące:

- określenie problemu (w jaki sposób zaprogramować ruch robota w języku Python),
- analizę działania programu sterującego ruchem robota,
- wykorzystanie gotowych bibliotek kodu do sterowania robotem,
- obserwację działania robota po uruchomieniu programu,
- testowanie programu i wprowadzanie prostych poprawek związanych ze sterowaniem ruchem robota.

Na zakończenie tej fazy lekcji nauczyciel podaje temat do zapisania w zeszycie i przechodzi do fazy II – realizacji praktycznej.

Faza II. Część zasadnicza - realizacja

W tej fazie lekcji nauczyciel zapewnia uczniom praktyczne ćwiczenia, które pozwolą eksperymentować z koncepcjami programowania. W szczególności obejmować to będzie używanie blokowych języków programowania oraz zestawu do robotyki.

1. Nauczyciel objaśnia sposób pracy na lekcji – we wskazanym przez niego momencie uczniowie będą wykonywać kolejne etapy prowadzące do rozwiązania zadania. Zdolniejsi uczniowie spróbują wykonać je samodzielnie. Słabsi wykonają kolejne kroki razem z nauczycielem, w oparciu o przygotowaną do zajęć instrukcję dostępną w e-podręczniku lub materiał multimedialny. W razie potrzeby będą mogli dopytać nauczyciela. Nauczyciel na bieżąco sprawdza postępy uczniów.
2. Bazując na posiadanych zasobach, zapewni każdemu uczniowi lub zespołowi uczniów kompletny zestaw do robotyki (tj. LEGO® Education SPIKE™).
3. Zestawy te pozwalają uczniom budować własne roboty i programować je do wykonywania określonych zadań. Nauczyciel zaczyna od prostych projektów, takich jak zbudowanie i zaprogramowanie robota, który może wykonać proste czynności, które wprawiają robota w ruch.
4. Uczniowie budują bazowego robota według załączonej instrukcji (załącznik **Budowa_bazy_robota.pdf** do lekcji).
5. Uczniowie włączają program **SPIKE**, poprzez który będą programować robota.
6. Nauczyciel, wykorzystując e-platformę, demonstruje na tablicy multimedialnej działanie programu i robota, uruchamiając odpowiedni moduł i materiał multimedialny.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

7. Prezentowany materiał multimedialny, zrealizowany w oparciu o szczegółowy scenariusz (załącznik do lekcji), oprócz waloru teoretycznego stanowi również podstawę przebiegu części praktycznej lekcji. W szczególności, uczniowie:
- a. budują prosty model bazowego robota w oparciu o zestaw do robotyki (tj. LEGO® Education SPIKE™),
 - b. tworzą listę czynności niezbędnych do wprawienia robota w ruch,
 - c. na podstawie listy czynności przygotowują program, zapisują go we wskazanym folderze (nauczyciel wskazuje właściwą lokalizację; może to być dysk lokalny lub zasób w chmurze) i potrafią go uruchomić;
 - d. wgrywają program do zbudowanego modelu robota, uruchamiają, obserwują i testują działanie robota. Wprowadzają modyfikacje w programie, poprawiają ewentualne błędy.

Faza III. Podsumowanie

Uczniowie analizują przedstawione przykłady **programów sterujących ruchem robota zapisanych w języku Python** oraz omawiają ich działanie. Nauczyciel ocenia stopień zrozumienia zależności między zapisanym kodem a **sposobem poruszania się robota po uruchomieniu programu**.

Uczniowie wspólnie wskazują najważniejsze elementy składni języka Python oraz omawiają **polecenia i biblioteki wykorzystywane do sterowania ruchem robota**, a także najczęściej pojawiające się błędy w zapisie kodu i sposoby ich poprawiania. Kryteria oceniania:

Ocena dopuszczająca:

- uczeń rozpoznaje podstawowe elementy programu w języku Python,
- potrafi wskazać polecenia odpowiedzialne za ruch robota.

Ocena dostateczna:

- uczeń analizuje przykładowy program i opisuje jego działanie,
- rozumie zależność między zapisanym kodem a działaniem programu,
- rozpoznaje podstawowe polecenia wykorzystywane do sterowania robotem.

Ocena dobra:

- uczeń samodzielnie modyfikuje przykładowy program,
- poprawia błędy związane z zapisem poleceń, nawiasami lub cudzysłowami,
- testuje działanie programu sterującego ruchem robota po wprowadzonych zmianach.

Ocena bardzo dobra:

- uczeń samodzielnie tworzy poprawny program w języku Python,
- wyjaśnia działanie poszczególnych elementów programu,
- **wykorzystuje biblioteki kodu do sterowania ruchem robota,**

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- proponuje własne rozwiązania i potrafi je uzasadnić,
- samodzielnie analizuje i poprawia błędy w kodzie.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, np.:

1. Czym jest biblioteka w języku Python?
 - a. Programem do budowania robota
 - b. Zbiorem gotowych funkcji i narzędzi**
 - c. Rodzajem czujnika
 - d. Typem silnika
2. Do czego służy słowo kluczowe **import**?
 - a. Do wczytania biblioteki do programu**
 - b. Do zatrzymania robota
 - c. Do zmiany prędkości
 - d. Do wyświetlania tekstu
3. Co oznacza zapis `from hub import light_matrix`?
 - a. Import całego programu
 - b. Usunięcie biblioteki
 - c. Zatrzymanie programu
 - d. Import wybranej części biblioteki**
4. Dlaczego używa się bibliotek w programowaniu robotów?
 - a. Aby robot działał wolniej
 - b. Aby nie pisać wszystkiego od zera**
 - c. Aby usunąć kod
 - d. Aby zmienić język

Faza V. Dodatkowe, praca domowa (dla chętnych).

Zadania dodatkowe (dla chętnych).

Zadanie 1: Programujemy ruch robota

Utwórz prosty program w języku Python, który spowoduje wykonanie przez robota kilku podstawowych ruchów.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Program powinien:

- zawierać polecenia odpowiedzialne za ruch robota,
- wykorzystywać gotowe biblioteki kodu LEGO SPIKE Prime,
- być zapisany poprawnie pod względem składni,
- wykonywać ruchy w logicznej kolejności,
- umożliwiać uruchomienie i przetestowanie działania robota.

Opcjonalnie możesz rozbudować program tak, aby robot:

- zatrzymywał się po wykonaniu ruchu,
- zmieniał kierunek jazdy,
- wykonywał skręt o określony kąt,
- poruszał się z różną prędkością,
- wykonywał przygotowaną trasę przejazdu.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 7 - *Kodujemy ruch naszego robota. Jak korzystać z bibliotek kodu?*

Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
 - Nauczyciel omawia podstawowe zagadnienia związane z programowaniem ruchu robota w języku Python:
 - wyjaśnia, że robot wykonuje instrukcje zapisane w programie w określonej kolejności,
 - przedstawia przykłady poleceń wykorzystywanych do sterowania ruchem robota,
 - omawia znaczenie poprawnej składni programu oraz **wykorzystania bibliotek kodu**.
 - Wprowadzenie do zadania i przypomnienie podstaw pracy w LEGO SPIKE Prime:
 - omówienie środowiska programowania Python w aplikacji LEGO SPIKE App,
 - **przedstawienie podstawowych bibliotek wykorzystywanych do sterowania robotem,**
 - przedstawienie zależności: zapis kodu >> uruchomienie programu >> ruch robota,
 - analiza przykładowych programów odpowiedzialnych za ruch robota.
 - Praca uczniów:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- analiza przedstawionych przykładów programów sterujących ruchem robota,
 - omawianie działania poszczególnych linijek kodu,
 - testowanie działania robota po uruchomieniu programu,
 - wyszukiwanie i poprawianie błędów w kodzie,
 - proponowanie własnych rozwiązań związanych ze sterowaniem ruchem robota.
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów (**analiza poprawności działania programu, sposobu poruszania się robota oraz umiejętności wykorzystania bibliotek kodu**).
 - Zadanie pracy domowej (opcjonalnie) – **napisanie prostego programu sterującego ruchem robota oraz przetestowanie działania programu po wprowadzeniu własnych zmian**.

Załącznik nr 2 – Realizacja zadania głównego podczas lekcji nr 7 - *Kodujemy ruch naszego robota. Jak korzystać z bibliotek kodu?*

Zadanie 1. Przygotowanie do realizacji zadania głównego

Uczniowie, pracując w zespołach lub indywidualnie, realizują zadanie polegające na **poznaniu sposobu korzystania z bibliotek kodu w języku Python** oraz programowaniu podstawowych ruchów robota w środowisku LEGO SPIKE Prime. **Szczególny nacisk położony jest na poprawne wykorzystanie gotowych bibliotek, rozumienie działania importowanych modułów oraz sterowanie ruchem robota za pomocą odpowiednich poleceń.**

Uczniowie opracowują i zapisują rozwiązanie problemu w postaci prostych programów sterujących ruchem robota oraz wyświetlaniem komunikatów na kontrolerze SPIKE Hub. W tym celu:

- analizują przykładowe programy zapisane w języku Python wykorzystujące biblioteki kodu,
- określają cel programu oraz przewidują możliwe błędy składniowe i logiczne,
- planują kolejność wykonywania instrukcji sterujących robotem,
- opracowują algorytm w uporządkowanej formie,
- zapisują go jako program w języku Python z wykorzystaniem bibliotek i modułów,
- testują działanie programu i obserwują sposób poruszania się robota,
- wprowadzają poprawki związane ze sterowaniem ruchem lub poprawnością składni kodu,
- doprecyzowują rozwiązanie tak, aby program był czytelny i skuteczny w działaniu.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

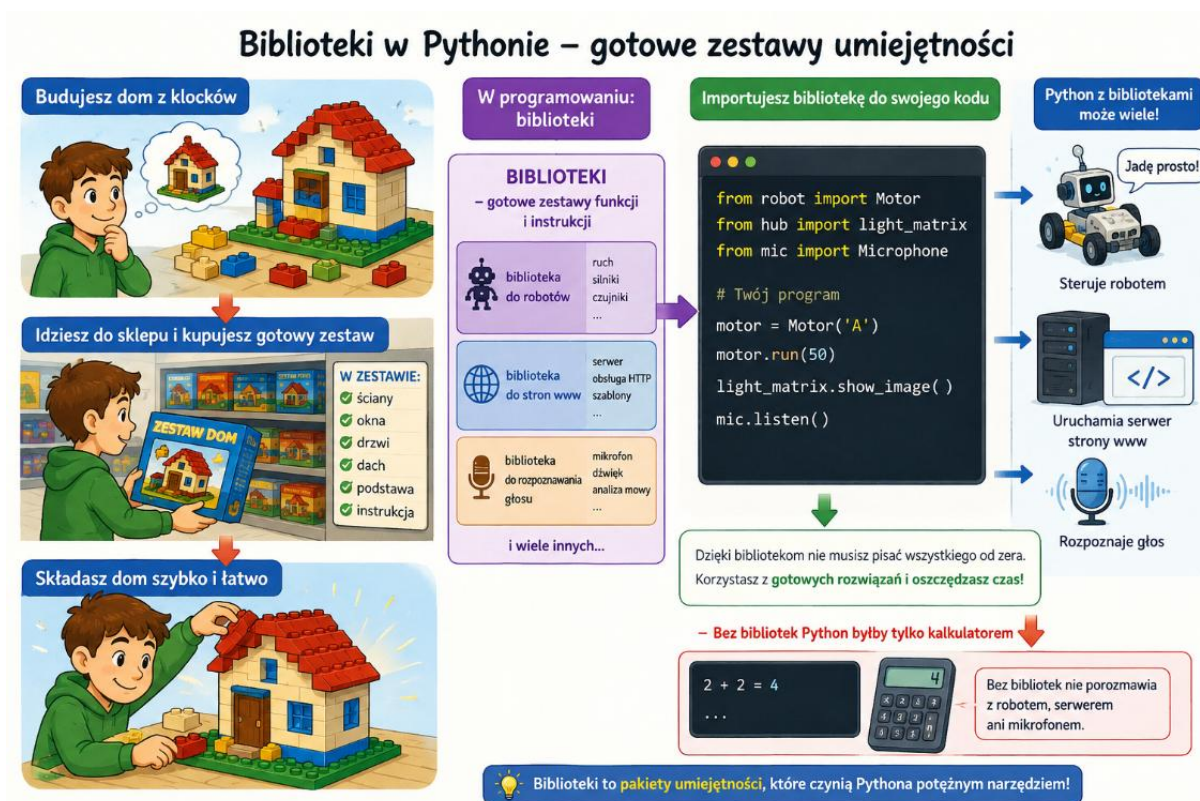
Uczniowie wykorzystują środowisko LEGO SPIKE Prime w trybie Python, zwracając uwagę na zgodność między zapisanym kodem a rzeczywistym działaniem robota. Szczególną uwagę poświęcają wykorzystaniu bibliotek i modułów, poprawnemu zapisowi poleceń oraz analizie komunikatów o błędach pojawiających się po uruchomieniu programu.

Zadanie 2. Wprowadzenie teoretyczne do lekcji

Biblioteki jako źródło informacji

Wyobraź sobie, że chcesz zbudować dom z klocków. Nie musisz samodzielnie produkować plastiku, formować każdego klocka ani projektować sposobu, w jaki będą się ze sobą łączyć. Idziesz do sklepu i kupujesz gotowy zestaw. W programowaniu takimi zestawami są właśnie biblioteki. Biblioteka to zbiór gotowych rozwiązań, funkcji i instrukcji napisanych przez innych programistów, z których możesz korzystać w swoich programach. Dzięki bibliotekom nie musisz tłumaczyć Pythonowi, jak dokładnie ma sterować napięciem w porcie USB, aby zakręcić silnikiem robota. Zamiast tego importujesz odpowiednią bibliotekę, która „uczy” program obsługi konkretnego urządzenia. Można powiedzieć, że biblioteki to takie „pakiety umiejętności”, które dołączamy do kodu na samym początku pracy. Bez nich Python byłby tylko potężnym kalkulatorem, ale dzięki nim staje się mózgiem robota, serwerem strony internetowej czy systemem rozpoznawania głosu.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Programowanie obiektowe

Nie da się mówić o bibliotekach, nie mówiąc o programowaniu obiektowym. Większość bibliotek służących do obsługi robotów (jak np. w systemie LEGO Spike czy EV3) korzysta właśnie z tak zwanego programowania obiektowego. Aby to zrozumieć, musisz poznać pojęcie klasy. Klasa to nic innego jak szczegółowy projekt lub instrukcja obsługi czegoś.

Pomyśl o klasie jak o projekcie technicznym silnika. W tym projekcie zapisane jest wszystko: co silnik potrafi robić (kręcić się, zatrzymywać, mierzyć obroty) oraz jakie ma cechy (moc, kierunek). Jednak sam projekt silnika na papierze nie zakręci kołem twojego robota. To tylko definicja, która mówi Pythonowi, czym w ogóle jest „Silnik” i jak się z nim komunikować. Klasy w bibliotekach są pisane po to, aby programista wiedział, jakie ma możliwości, zanim jeszcze fizycznie podłączy urządzenie. Klasę możesz traktować jako ogólne założenie – mówię np. **kontroler Spike Hub** (i to jest klasa), a Ty mówisz: **ten konkretny kontroler**, który trzymam w ręce – to jest **obiekt** klasy kontroler Spike Hub. Obiekt jest ściśle powiązany z klasą i o nim troszkę teraz. W programowaniu proces tworzenia obiektu na podstawie klasy nazywa się fachowo „instancjonowaniem” lub po prostu powoływaniem obiektu. Nie musisz znać tego słowa. Pewnie go nie użyjesz nigdy, ale gdzieś Ci mignęło i nikt Ci nie powie, że nie wiesz.

Dlaczego to jest takie ważne? Wyobraź sobie, że Twój robot ma dwa silniki: lewy i prawy. Oba są identyczne, czyli pochodzą z tej samej klasy (tego samego projektu). Jednak w programie

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

musisz je rozróżnić, aby móc skręcać. Dlatego stworzymy dwa oddzielne obiekty. Jednemu nadajemy nazwę **lewy_silnik**, a drugiemu **prawy_silnik**. Od tego momentu Python wie, że kiedy wydajesz polecenie obiektowi **lewy_silnik**, ma ono trafić do konkretnego urządzenia podłączonego do konkretnego portu, a nie do wszystkich silników naraz. Zrozumiesz to dokładnie w kolejnych lekcjach, gdzie pojawi się programowanie silnika lewego i prawego, a także silnika trzeciego, który najczęściej będzie służył jako manipulator.

PROGRAMOWANIE OBIEKTOWE

Klasy to projekty, obiekty to konkretne rzeczy!

1. KLASA – PROJEKT / INSTRUKCJA

Klasa to projekt techniczny silnika.
Określa, co potrafi robić i jakie ma cechy.

Silnik potrafi:

- kręcić się
- zatrzymywać się
- mierzyć obroty
- ma moc i kierunek

Klasa to tylko definicja. Sama w sobie nic nie porusza!

2. OBIEKT – KONKRETNY EGZEMPLARZ

Obiekt to rzeczywisty silnik podłączony do robota. Tworzymy go na podstawie klasy.

To jest obiekt:

lewy_silnik
podłączony do portu A

`lewy_silnik = Motor('A')`

Tworzymy obiekt (instancjonujemy klasę Motor) i przypisujemy mu nazwę.

3. DLACZEGO TO WAŻNE?

Mamy dwa silniki z tej samej klasy, ale są to dwa różne obiekty. Dzięki temu możemy sterować każdym z nich niezależnie!

lewy_silnik
port A

prawy_silnik
port B

Polecenie wysłane do lewy_silnik trafia tylko do silnika w porcie A, a nie do wszystkich!

4. JAK TO WYGLĄDA W KODZIE?

```
from spike import Motor
# Tworzymy dwa obiekty klasy Motor
lewy_silnik = Motor('A')
prawy_silnik = Motor('B')

# Używamy metod (czynności), które ma silnik
lewy_silnik.run(50) # jedź do przodu
prawy_silnik.run(50)
lewy_silnik.stop() # zatrzymaj lewy silnik
prawy_silnik.stop() # zatrzymaj prawy silnik
```

Importujemy klasę Motor z biblioteki spike.

Tworzymy dwa obiekty (egzemplarze klasy Motor). Określamy port, do którego jest podłączony silnik.

Wywołujemy metody obiektu za pomocą kropek. Każdy obiekt działa niezależnie.

METODY OBIEKTU (PRZYKŁADY)

- `run(50)` – kręć się do przodu z mocą 50
- `run(-50)` – kręć się do tyłu z mocą 50
- `stop()` – zatrzymaj się
- `get_degrees_counted()` – zmierz obroty
- `set_default_speed(30)` – ustaw domyślną moc

PODSUMOWANIE: KLASA TO OGÓLNY PROJEKT, A OBIEKT TO KONKRETNA RZECZ.

Programowanie obiektowe pozwala budować duże i uporządkowane programy, w których każdy element robota ma swoją nazwę i swoje zadania.

Jak to wygląda w kodzie?

W Pythonie korzystanie z biblioteki i tworzenie obiektu odbywają się zazwyczaj w trzech krokach. Najpierw musimy „sprowadzić” projekt do naszego programu za pomocą słów **from** oraz **import**. Mówimy wtedy Pythonowi: „Z biblioteki dla robotów weź projekt o nazwie Silnik”. Popatrz na przykład obsługi ekranu matrycy kontrolera **Spike Hub**.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Connect

```
1 from hub import light_matrix
2 import runloop
3
4 async def main():
5     # write your code here
6     await light_matrix.write("Hi!")
7
8 runloop.run(main())
9
```

Jak widzisz, pojawiają się na samej górze słowa **from** i **import**. Czym się różnią? Słowo **import** pobiera nam wszystkie dane, jakie biblioteka posiada. Akurat **runloop** jest biblioteką niezbędną do działania robota, dlatego zawsze będzie obecna w twoim programie. Użycie zdania **from hub import light_matrix** oznacza, że z baaardzo obszernej biblioteki **hub** wyciągasz tylko część. Tę część w przypadku wyżej jest obsługa matrycy na kontrolerze Spike Hub. Klasa (biblioteka) **hub** ma dużo więcej funkcjonalności, które w ramach nauki na lekcjach będziesz rozwijać i poznawać. Podczas zajęć praktycznych będziesz importować różne moduły, dodawać i odkrywać nowe.

Kolejnym krokiem jest stworzenie obiektu. Robimy to, przypisując klasę do nazwy, którą sami wymyślimy, na przykład: **moje_kolo = Silnik('A')**. Litera w nawiasie to informacja dla Pythona, do którego portu w robocie podłączone jest to konkretne urządzenie. Od tej linii kodu słowo **moje_kolo** staje się dla komputera żywym obiektem, który ma swoje metody. I tak to działa w normalnym Pythonie. W LEGO Spike nie potrzebujemy dodatkowych nazw. Używa się od razu komendy wywołującej jakiś efekt. Popatrz na rysunek wyżej. Pojawia się tam taki kod:

```
await light_matrix.write("Hi!")
```

Z modułu **light_matrix** będącego częścią biblioteki **hub** wyciągana jest funkcja **write**. Funkcja **write** odpowiada za konkretne działanie. W tym wypadku jest to wypisanie tekstu na ekranie kontrolera Spike Hub. W programowaniu znane są Ci już funkcje. Możesz się jednak spotkać ze słowem **metody**. Metody używa się właśnie w programowaniu obiektowym i, prawdę mówiąc, są to funkcje. Metody to działania, które obiekt potrafi wykonać. Wywołujemy je po kropce, na przykład: **moje_kolo.start()**. Ta kropka jest kluczowa – oznacza ona, że polecenie **start** odnosi się konkretnie do obiektu **moje_kolo**. To właśnie dzięki takiemu zapisowi programowanie tekstowe jest tak precyzyjne. Możemy mieć dziesięć różnych obiektów tej samej klasy i każdym sterować niezależnie, używając ich unikalnych nazw.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



IMPORT, OBIEKTY I METODY W PYTHONIE (LEGO SPIKE PRIME)

Jak korzystać z bibliotek i sterować robotem?



1 IMPORT – jak „sprowadzamy” bibliotekę do programu?

import – pobiera wszystko

Importuje całą bibliotekę (wszystkie moduły, klasy, funkcje).

```
import runloop  
# runloop – niezbędna do działania  
# robota, zawsze obecna
```

runloop to biblioteka systemowa w SPIKE Prime – potrzebna w każdym programie.

from ... import ... – pobiera tylko część

Importuje tylko wybrany moduł (część) z danej biblioteki.

```
from hub import light_matrix
```

Z bardzo obszernej biblioteki hub wyciągamy tylko moduł light_matrix – obsługę matrycy wyświetlacza w kontrolerze Spike Hub.

2 OBIEKTY – konkretne urządzenia w Twoim programie

Tworzymy obiekt, przypisując klasę do własnej nazwy.

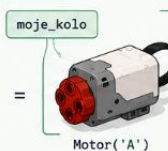
W Pythonie (ogólnie)

Tworzymy obiekt z klasy (np. Motor) i podajemy port, do którego jest podłączony.

```
from spike import Motor  
moje_kolo = Motor('A')
```

moje_kolo – nazwa obiektu (wymyślona)
'A' – port, do którego podłączony jest silnik

Co to oznacza?



Od tego momentu
moje_kolo jest żywym obiektem, który ma swoje metody (czynności).

Możemy mieć wiele obiektów tej samej klasy, np.:

```
lewy_silnik = Motor('A')  
prawy_silnik = Motor('B')
```

3 METODY – działania obiektu

Metody to funkcje przypisane do obiektu. Wywołujemy je po kropce.

Przykład w Pythonie (ogólnie)

```
moje_kolo.start(50) # rusz do przodu  
moje_kolo.stop() # zatrzymaj się  
moje_kolo.run_for_degrees(360, 50) # obróć o 360°
```

Kropka (.) mówi Pythonowi: „wykonaj tę metodę na tym konkretnym obiekcie”.

Dlaczego to ważne?
Dzięki obiektom możemy sterować każdym urządzeniem osobno.



4 W LEGO SPIKE – używamy od razu gotowych komend

Nie tworzymy własnych nazw obiektów – wywołujemy funkcję (metodę) z modułu.

Przykład z modułu light_matrix

```
await light_matrix.write("Hi!")
```

Z modułu light_matrix (części biblioteki hub) wywołujemy metodę write().
Wynik: tekst „Hi!” pojawi się na matrycy kontrolera Spike Hub.



Inne przykłady metod w LEGO SPIKE

```
light_matrix.clear() # wyczyść matrycę  
light_matrix.show_image(...) # wyświetl obraz  
speaker.play_tone(60, 1) # zagraj dźwięk  
motor.run(50) # uruchom silnik  
sensor.distance() # zmierz odległość
```

Każda metoda to konkretne działanie obiektu (modułu).

PODSUMOWANIE – najważniejsze zasady



Importujemy biblioteki
import – wszystko
from ... import ... – tylko to,
czego potrzebujemy

Tworzymy obiekty
(nazwy), jeśli chcemy
sterować urządzeniami
osobno.

Metody wywołujemy
po kropce – to działania
obektu.

W LEGO SPIKE często
używamy gotowych
komend z modułów
bez tworzenia nazw.

Obiekty = precyzja, porządek
i możliwość budowy
zaawansowanych programów
i robotów.

Dzięki obiektom
mój robot robi
dokładnie to,
co chcę!



Dlaczego obiekty są lepsze niż zwykłe polecenia?

Przejście na system klas i obiektów pozwala na budowanie bardzo skomplikowanych maszyn. W Scratchu często pojawiał się jeden zestaw bloczków do „wszystkich silników” – różowy. W Pythonie, dzięki bibliotekom i obiektom, Twój kod staje się uporządkowany. Jeśli chcesz sprawdzić, jaki kolor widzi czujnik, nie pytasz „jakiegoś” czujnika, ale pytasz konkretny obiekt: **czujnik_koloru.pobierz_kolor()**.

Zrozumienie, że programowanie to zarządzanie takimi wirtualnymi reprezentacjami fizycznych urządzeń, to klucz do zostania prawdziwym inżynierem. Na następnych lekcjach zobaczysz, że dokładnie w ten sam sposób tworzy się zaawansowane roboty wymagające bardzo specjalistycznych ruchów.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



DLACZEGO OBIEKTY SĄ LEPSZE NIŻ ZWYKŁE POLECENIA?

Obiekty pozwalają sterować konkretnymi urządzeniami – precyzyjnie, porządnie i bez zamieszania!



1. W SCRATCHU – „WSZYSTKIE SILNIKI”

Jeden zestaw bloków działa na wszystkie silniki naraz.

- Ruch
- Wygląd
- Dźwięk
- Zdarzenia
- Kontrola
- Czujniki
- Wyrażenia
- Moje bloki

ustaw silniki na 50 % mocy

uruchom silniki przez 2 sek.

zatrzymaj wszystkie silniki

ustaw kierunek silników sym samym

Działa na wszystkie silniki jednocześnie.

2. W PYTHONIE – KAŻDY OBIEKT TO KONKRETNE URZĄDZENIE

Dzięki obiektom mówimy dokładnie, któremu urządzeniu jest wydawane polecenie.

lewy_silnik

port A

```
# silnik lewy - port A
lewy_silnik.run(50)
# jedzie do przodu
lewy_silnik.stop()
```

prawy_silnik

port B

```
# silnik prawy - port B
prawy_silnik.run(-50)
# jedzie do tyłu
prawy_silnik.stop()
```

✓ Sterujesz każdym silnikiem osobno.
✓ Możesz skręcać, zawracać, wykonywać złożone ruchy.

3. PRACA Z CZUJNIKIEM – KONKRETNY OBIEKT

Nie pytamy „jakis czujnik”, tylko pytamy konkretny obiekt o konkretną informację.

czujnik_koloru

port C

```
kolor = czujnik_koloru.pobierz_kolor()
print(kolor)
```

Pytamy konkretny czujnik o to, co widzi.
Wynik jest dokładny i jednoznaczny.

4. ZALETY OBIEKTÓW

- Precyzja** – polecenie trafia dokładnie do wybranego urządzenia.
- Porządek w kodzie** – każdy obiekt ma swoją nazwę i rolę.
- Łatwe rozbudowywanie** – możesz dodać kolejne urządzenia bez zamieszania.
- Czytelność** – kod jest zrozumiały dla Ciebie i dla innych.
- Możliwości** – budujesz bardziej złożone, inteligentne roboty.

Obiekty to jak pilot, który steruje każdym urządzeniem z osobna!

5. PORÓWNIANIE W PRAKTYCE

BEZ OBIEKTÓW (mało precyzyjne)

```
# Wyobraź sobie polecenie „uruchom silniki”
```

✗ Nie wiadomo, które silniki. Trudno sterować robotem.

Z OBIEKTAMI (precyzyjne)

```
lewy_silnik.run(50)
prawy_silnik.run(-50)
czujnik_koloru.pobierz_kolor()
```

✓ Wiesz dokładnie, co i gdzie działa. Możesz tworzyć zaawansowane programy!

VS

KLUCZ DO INŻYNIERA!
Programowanie to zarządzanie wirtualnymi reprezentacjami fizycznych urządzeń. Zrozum to dobrze, a będziesz tworzyć niesamowite roboty!

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 3. Przykład praktyczny

Zadanie:

Napisz kod w Pythonie, który wyświetli na ekranie kostki Spike Hub dowolny napis. Przykładowo przywitanie, a następnie wyświetlenie dodatkowej informacji, czyli przykładowo Twój ulubiony kolor jako tekst.

Realizacja:



```

1 from hub import light_matrix
2 import runloop
3
4 async def main():
5     # write your code here
6     await light_matrix.write("Hej!")
7     await light_matrix.write("Uwielbiam kolor niebieski")
8
9 runloop.run(main())
10

```

Powyższy kod wyświetla 2 teksty, pierwszy z nich to przywitanie „Hej!”, a kolejny to informacja na temat ulubionego koloru „Uwielbiam kolor niebieski”. Kod można zweryfikować na kostce Spike Hub.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie do samodzielnego rozwiązania – 01

Przeanalizuj poniższy kod, znajdź i popraw błędy w nim istniejące, a następnie wgraj program na kostkę i zweryfikuj jego działanie.



Connect

```
1 from chub import light_matrix
2 import runloop
3
4 async def main():
5     # write your code here
6     await light_matrix.write(HELLO)
7     await light_matrix.write("START")
8
9 runloop.run(main())
10
```

Zadanie do samodzielnego rozwiązania – 02

Znajdź i podaj dokładne nazwy biblioteki i ewentualnego modułu odpowiedzialnego za:

- Czujnik odległości
- Czujnik siły
- Dźwięk

Przykład matryca/wyświetlacz kostki – biblioteka: **hub** moduł: **light_matrix**

Zadanie do samodzielnego rozwiązania – 03

Napisz kod wyświetlający na kostce obrazek o następujących numerach z bazy domyślnych obrazków i podpisz każdy z nich:

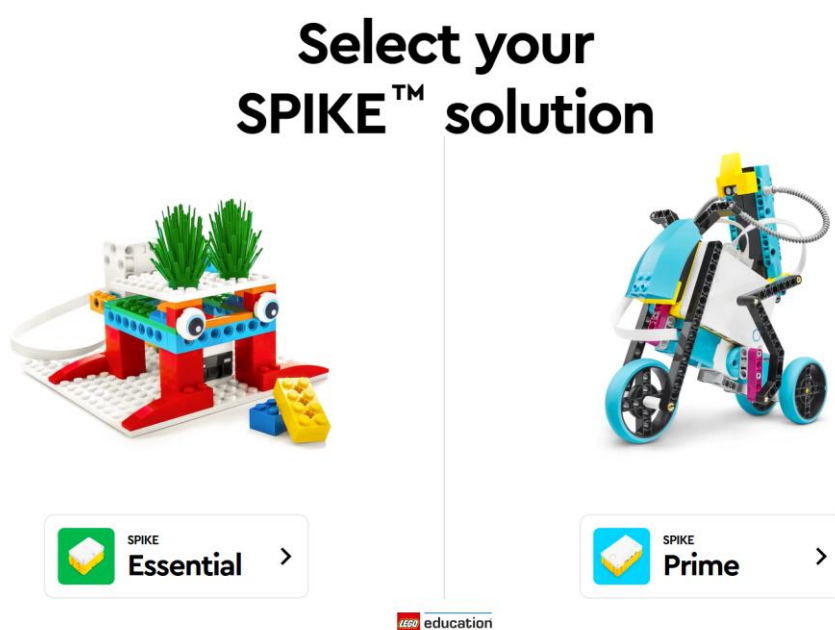
- | | |
|------|------|
| • 46 | • 63 |
| • 58 | • 66 |

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 3. Konfiguracja środowiska.

Uruchamianie aplikacji LEGO Spike

1. Włącz przeglądarkę internetową dostępną na twoim urządzeniu
2. Wpisz w przeglądarce adres: <https://spike.legoeducation.com/> Pojawi się okno wyboru wersji LEGO SPIKE. Wybierz Spike Prime

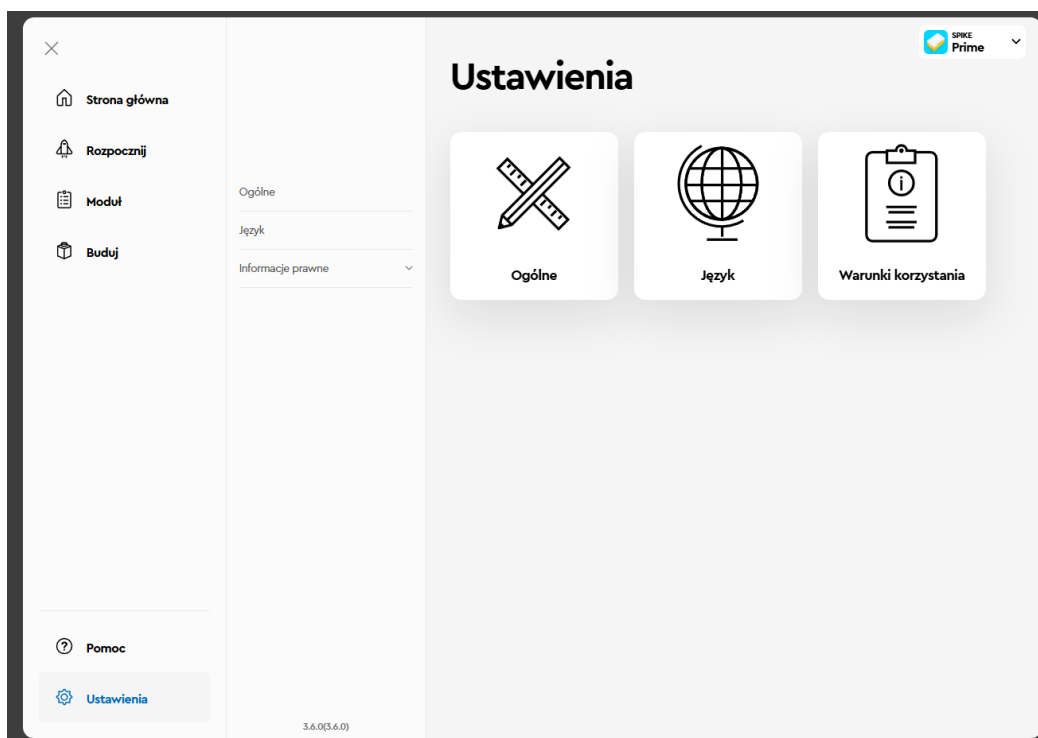


3. Po wybraniu opcji SPIKE Prime pojawia się menu główne. Jeśli występuje język polski, należy zmienić go na język angielski.

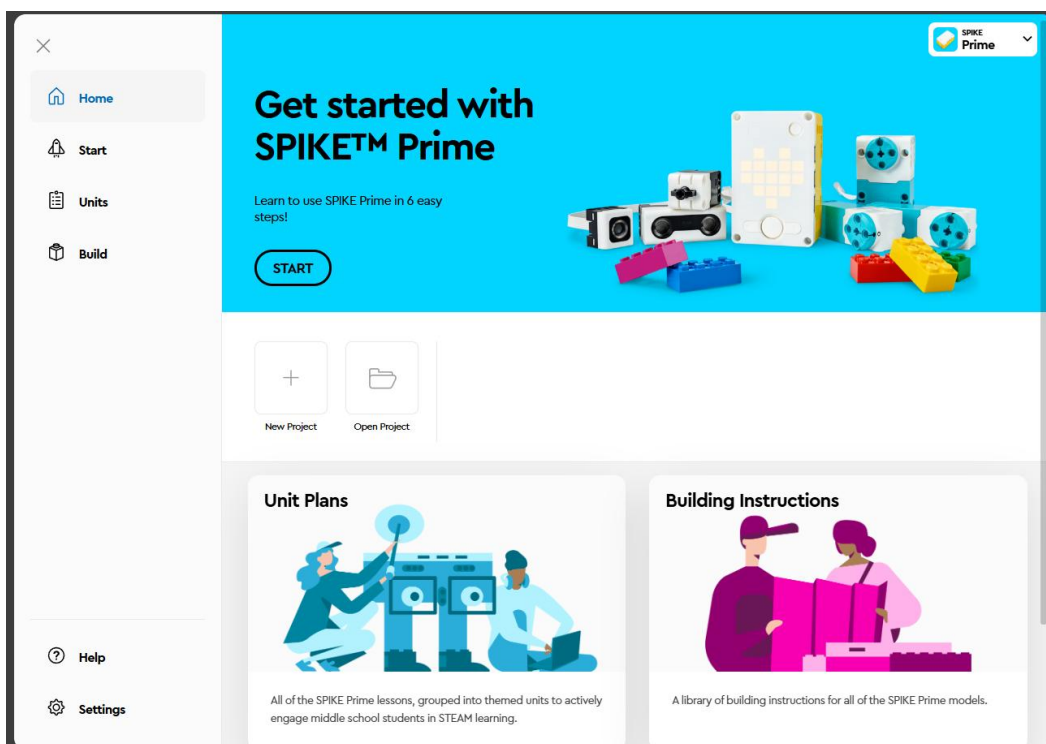
Jeżeli zaś wszystko jest w **języku angielskim**, to przejdź do kroku 6.

Aby zmienić język, w lewym dolnym rogu menu głównego kliknij przycisk **Ustawienia**.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

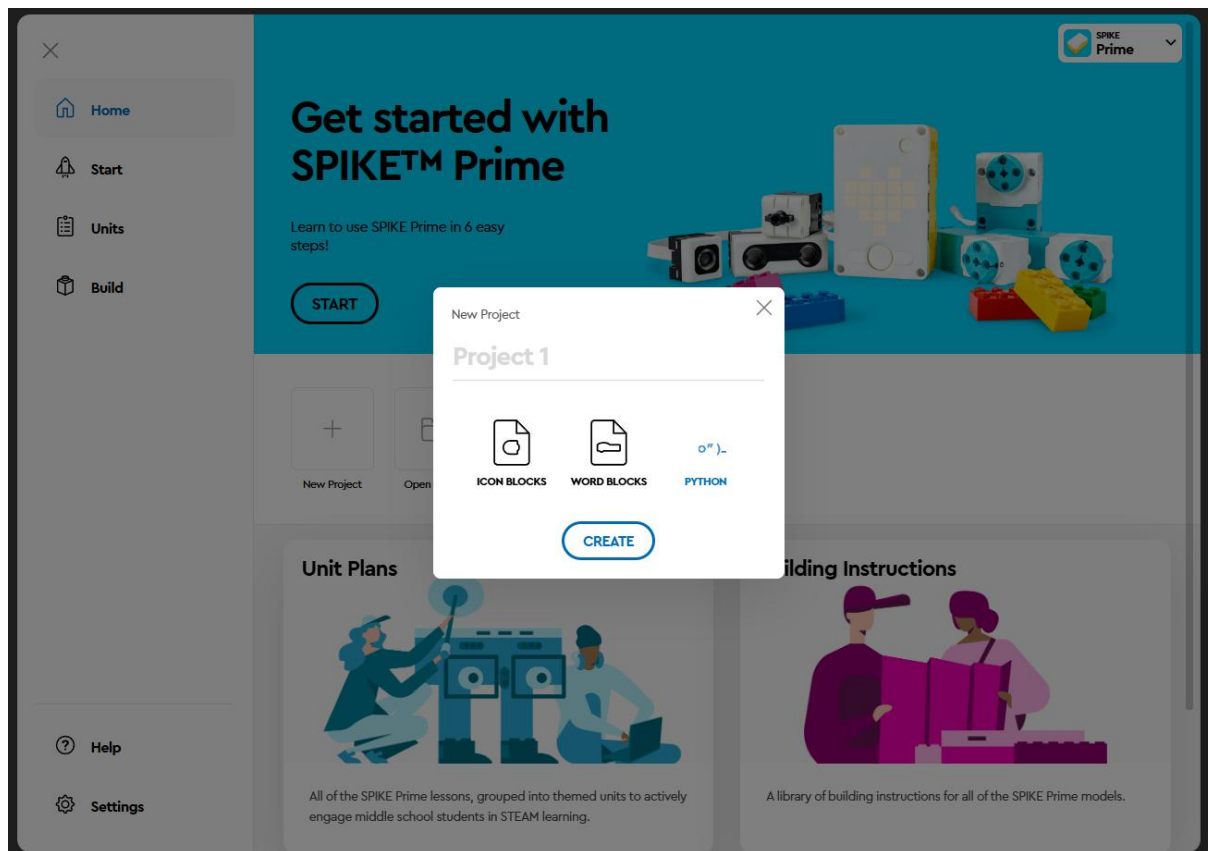


4. W nowym oknie kliknij **globus** podpisany „Language”.
5. **Wybierz język: United States (English)**. Strona przeładuje się i język angielski będzie dostępny.
6. Kliknij po lewej przycisk **Home**. Pojawi się okno jak poniżej

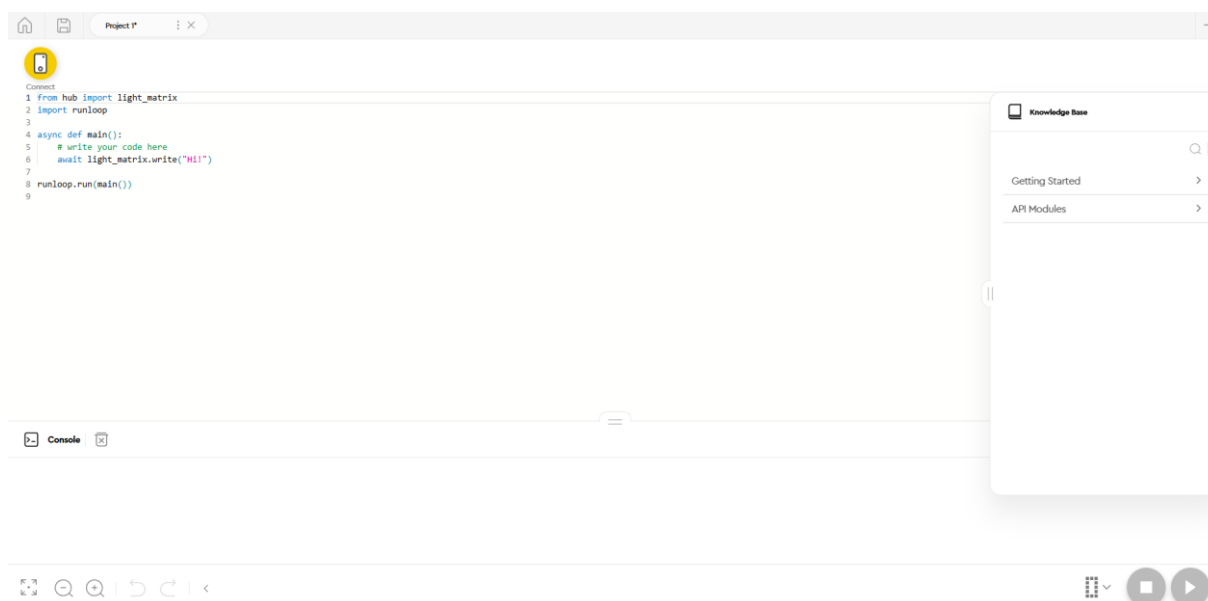


Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

7. Kliknij na środku przycisk **New Project**. Pojawi się nowe okno jak poniżej



8. Wybierz opcję **Python** oraz podaj nazwę projektu. Kiedy wszystko zostanie poprawnie wybrane, wciśnij przycisk **Create**.

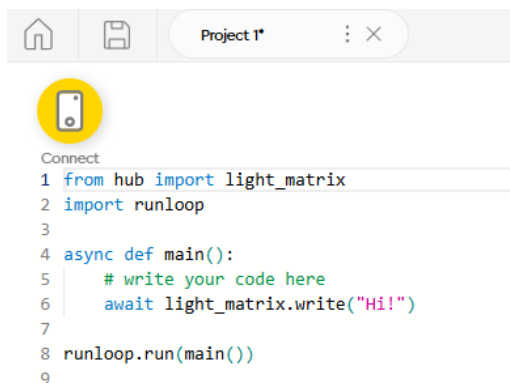


Wszystko jest gotowe do programowania. Pora na podłączenie kontrolera Spike Hub.

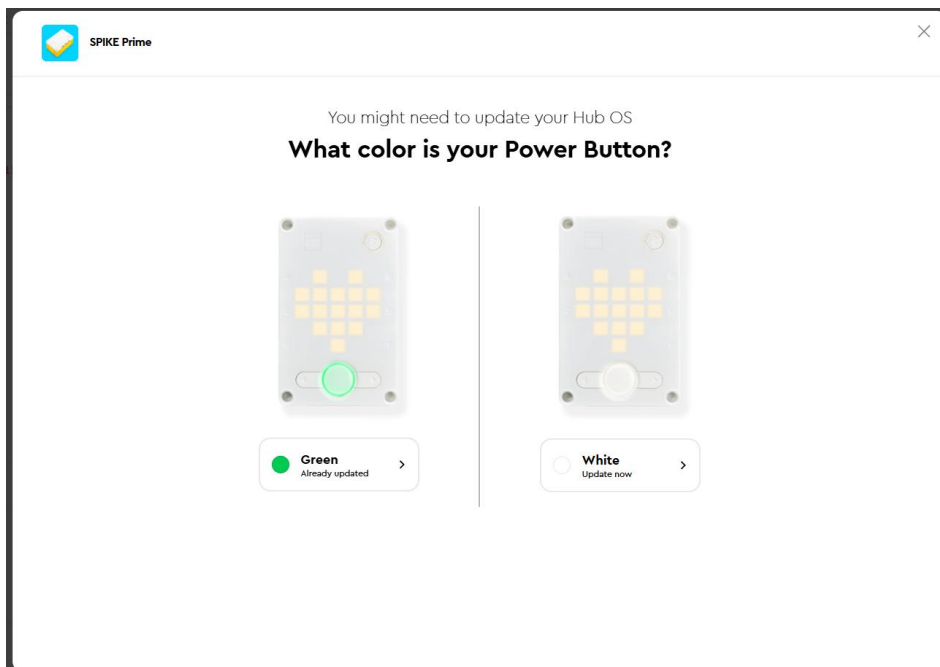
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 3. Podłączenie kontrolera Spike Hub

1. W głównym programie kliknij przycisk **Connect** znajdujący się w górnym rogu, tuż pod nazwą twojego projektu. Patrz poniżej.

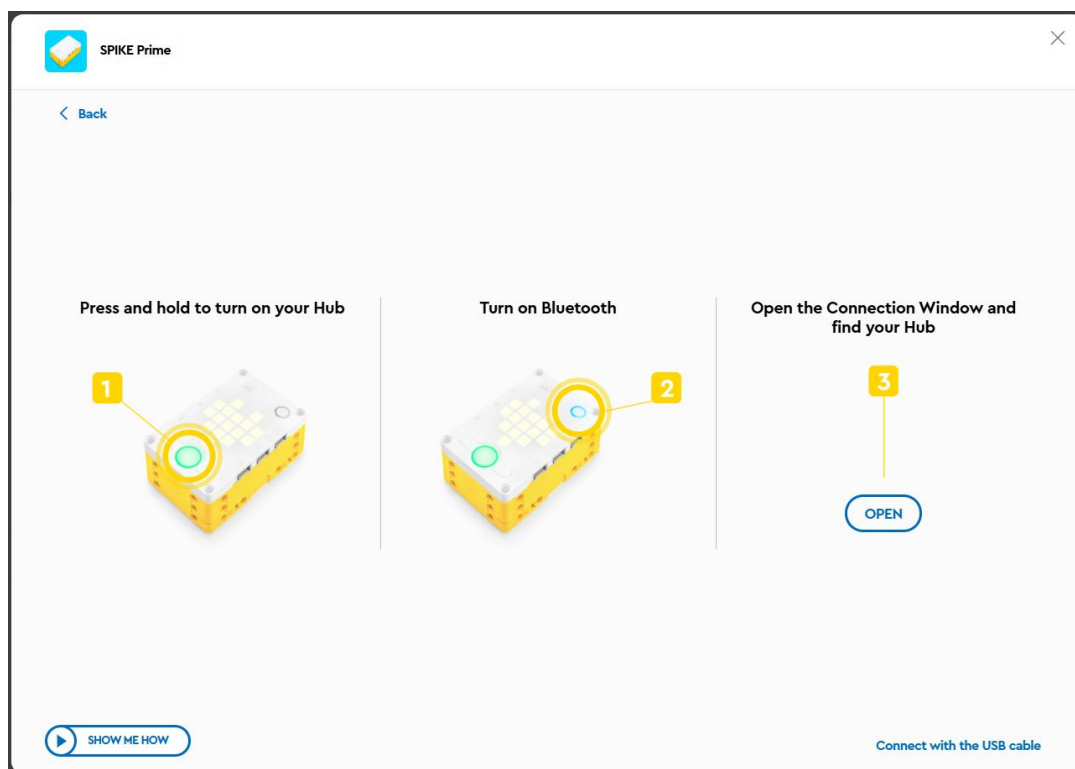


2. Pojawi się nowe okno. **Włącz kontroler Spike Hub** klikając **duży biały przycisk** i poczekaj aż się uruchomi. Następnie wybierz jedną z opcji odpowiadającą stanowi kontrolera Spike Hub – zawsze będzie to **Green – Already updated**.

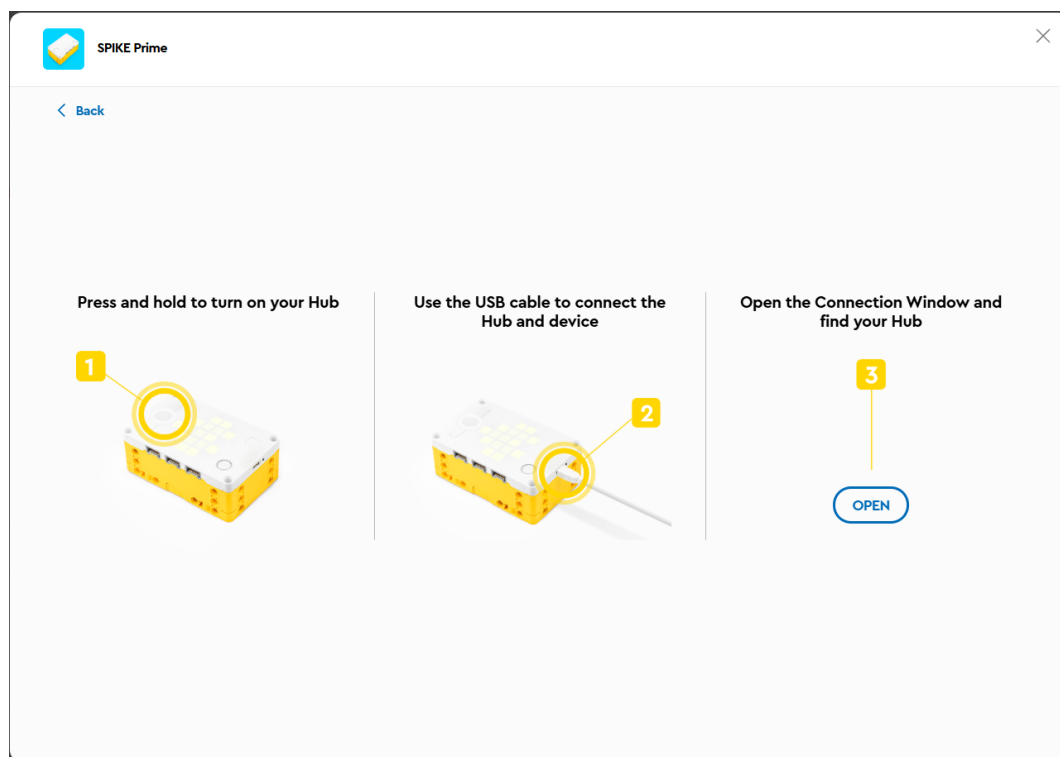


3. Pojawi się nowe okno (patrz poniżej). Wybierz opcję **Connect with USB cable** znajdującą się w prawym dolnym rogu okna.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

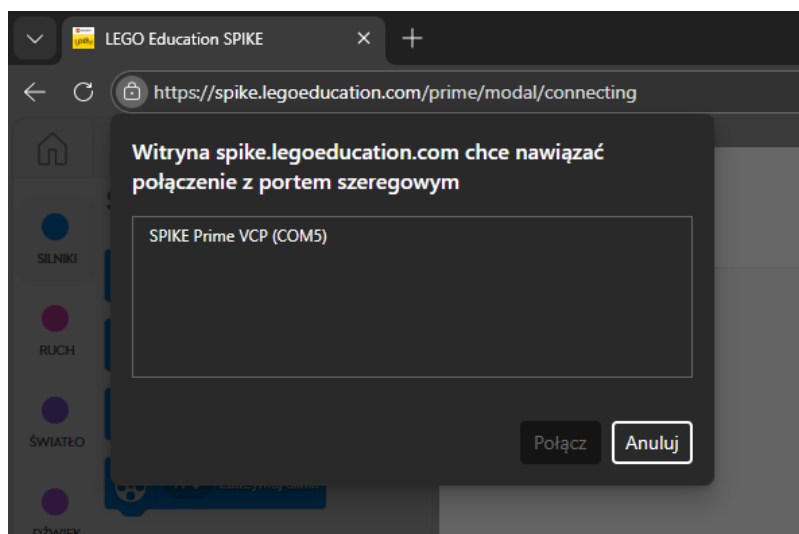


4. Postępuj zgodnie z instrukcją na ekranie. Po przeczytaniu kliknij przycisk **Open**.



5. Pojawi się nowe okno (patrz poniżej)

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



6. Wybierz opcję **SPIKE Prime VCP** i kliknij **Połącz**.

UWAGA – u Ciebie nazwa może nie być identyczna jak na rysunku powyżej. Wybierz tę opcję, która ma w nazwie Spike Prime. Innymi rzeczami się nie przejmuj.

Jeśli wszystko zostanie poprawnie połączone, to okna zostaną automatycznie zamknięte, a ikona kontrolera w programie zmieni się na zieloną.



```
1 from hub import light_matrix
2 import runloop
3
4 async def main():
5     # write your code here
6     await light_matrix.write("Hi!")
7
8 runloop.run(main())
9
```

Gratulacje. Twój Spike Hub jest podłączony do komputera. Możesz przystąpić do programowania robota.

W przypadku problemów postępuj wg kroków:

1. Sprawdź, czy kontroler Spike Hub jest włączony i świeci na zielono
 - a. Jeśli kontroler świeci na biało zgłoś to nauczycielowi
 - b. Jeśli kontroler świeci na zielono to podążaj dalej wg instrukcji łączenia powyżej
2. Sprawdź czy kabel USB jest podłączony (Spike Hub <-> Komputer)
3. Sprawdź, czy wybrano odpowiednią nazwę urządzenia – w oknie połączenia z portem szeregowym należy wybrać urządzenie, które w nazwie ma SPIKE Prime.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 2: Odkrywamy tajemnice Kodu... Roboty w uścisku Pythona

Temat lekcji nr 8 - Czy można w języku Python programować szkolne roboty?

Treści nauczania (wymagania szczegółowe - podstawa programowa) – I. 4);

Cel lekcji:

- **Wprowadzenie do programowania czujników w języku Python** – uczniowie poznają podstawowe zasady korzystania z bibliotek i modułów odpowiedzialnych za obsługę czujników w LEGO SPIKE Prime.
- **Analiza działania programu** – uczniowie analizują fragmenty kodu odpowiedzialne za odczytywanie danych z czujników oraz określają, w jaki sposób poszczególne polecenia wpływają na działanie programu i robota.
- **Tworzenie prostych algorytmów i programów wykorzystujących czujniki** – uczniowie opracowują algorytmy oraz zapisują je w postaci programów tekstowych umożliwiających pobieranie danych z czujników odległości, koloru oraz siły.
- **Testowanie działania programu** – uczniowie uruchamiają program, obserwują wyniki działania w konsoli oraz analizują dane zwracane przez czujniki, a następnie wprowadzają poprawki w celu uzyskania poprawnego działania programu.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- **Uczeń dostrzega, że działanie programu i robota zależy od poprawności zapisanych poleceń:**
 - rozumie, że robot wykonuje instrukcje zgodnie z zapisanym kodem programu oraz danymi pobieranymi z czujników,
 - obserwuje działanie programu napisanego w języku Python i analizuje informacje zwracane przez czujniki robota.
- **Uczeń analizuje działanie programu i jego efekty:**
 - sprawdza poprawność działania programu wykorzystującego czujniki robota w języku Python,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- identyfikuje błędy składniowe i logiczne oraz modyfikuje program poprzez poprawę zapisu poleceń lub zmianę sposobu odczytu danych z czujników,
- wykorzystuje biblioteki i moduły służące do obsługi czujników LEGO SPIKE Prime.
- **Uczeń rozwija umiejętność testowania i doskonalenia rozwiązań:**
 - wykonuje próby działania programu i analizuje dane zwracane przez czujniki,
 - wprowadza poprawki w celu uzyskania poprawnego działania programu i właściwego odczytu danych,
 - testuje działanie różnych czujników oraz porównuje uzyskiwane wyniki.

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści kształcenia będą realizowane za pomocą robotyki edukacyjnej LEGO® Education SPIKE™ Prime.

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna RaP STEAM,
- praca na komputerze,
- praca przy zestawie robotyki edukacyjnej,
- pogadanka.

Środki dydaktyczne

- zestaw PC z dostępem do Internetu,
- zestaw robotyki edukacyjnej LEGO® Education SPIKE™ Prime oraz zestaw rozszerzony LEGO® Education SPIKE™ Prime Expansion Kit,
- rzutnik multimedialny,
- e-podręcznik.

Formy pracy

- Praca indywidualna lub grupowa.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przebieg lekcji:

Faza I. Wprowadzenie

Na początku lekcji nauczyciel wprowadza uczniów w zagadnienia związane z programowaniem czujników w języku Python, wyjaśniając, że robot może pobierać informacje z otoczenia za pomocą odpowiednich czujników oraz wykorzystywać je podczas działania programu. Następnie odnosi się do zadania lekcyjnego, polegającego na odczytywaniu danych z czujników LEGO SPIKE Prime oraz wyświetlaniu ich w konsoli programu.

Nauczyciel przedstawia cel i zakres zadania. **Prezentuje uczniom środowisko programowania w języku Python w aplikacji LEGO SPIKE App oraz omawia sposób korzystania z bibliotek i modułów odpowiedzialnych za obsługę czujników.** Następnie przekazuje uczniom niezbędne informacje dotyczące wykorzystania czujnika odległości, czujnika koloru oraz czujnika siły.

Uczniowie zapoznają się ze sposobami zapisu poleceń **służących do pobierania danych z czujników oraz obserwują, jak program po uruchomieniu wyświetla wyniki działania w konsoli.**

Następnie uczniowie zdobywają umiejętności obejmujące:

- określenie problemu (w jaki sposób pobierać dane z czujników w języku Python),
- analizę działania programu wykorzystującego czujniki,
- wykorzystanie bibliotek i modułów służących do obsługi czujników,
- obserwację wyników działania programu po jego uruchomieniu,
- testowanie programu i analizę danych zwracanych przez czujniki,
- wprowadzanie prostych poprawek związanych z działaniem programu i odczytem danych.

Na zakończenie tej fazy lekcji nauczyciel podaje temat do zapisania w zeszycie i przechodzi do fazy II – realizacji praktycznej.

Faza II. Część zasadnicza - realizacja

W tej fazie lekcji nauczyciel zapewnia uczniom praktyczne ćwiczenia, które pozwolą eksperymentować z koncepcjami programowania. W szczególności obejmować to będzie używanie blokowych języków programowania oraz zestawu do robotyki.

1. Nauczyciel objaśnia sposób pracy na lekcji – we wskazanym przez niego momencie uczniowie będą wykonywać kolejne etapy prowadzące do rozwiązania zadania. Zdolniejsi uczniowie spróbują wykonać je samodzielnie. Słabsi wykonają kolejne kroki razem z nauczycielem, w oparciu o przygotowaną do zajęć instrukcję dostępną w e-

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- podręczniku lub materiał multimedialny. W razie potrzeby będą mogli dopytać nauczyciela. Nauczyciel na bieżąco sprawdza postępy uczniów.
2. Bazując na posiadanych zasobach, zapewnij każdemu uczniowi lub zespołowi uczniów kompletny zestaw do robotyki (tj. LEGO® Education SPIKE™).
 3. Zestawy te pozwalają uczniom budować własne roboty i programować je do wykonywania określonych zadań. Nauczyciel zaczyna od prostych projektów, takich jak zbudowanie i zaprogramowanie robota, który może wykonać proste czynności, które wprowadzą robota w ruch.
 4. Uczniowie budują bazowego robota według załączonej instrukcji (załącznik **Budowa_bazy_robota.pdf** do lekcji).
 5. Uczniowie włączają program **SPIKE**, poprzez który będą programować robota.
 6. Nauczyciel, wykorzystując e-platformę, demonstruje na tablicy multimedialnej działanie programu i robota, uruchamiając odpowiedni moduł i materiał multimedialny.
 7. Prezentowany materiał multimedialny, zrealizowany w oparciu o szczegółowy scenariusz (załącznik do lekcji), oprócz waloru teoretycznego stanowi również podstawę przebiegu części praktycznej lekcji. W szczególności, uczniowie:
 - a. budują prosty model bazowego robota w oparciu o zestaw do robotyki (tj. LEGO® Education SPIKE™),
 - b. tworzą listę czynności niezbędnych do wprowadzenia robota w ruch,
 - c. na podstawie listy czynności przygotowują program, zapisują go we wskazanym folderze (nauczyciel wskazuje właściwą lokalizację; może to być dysk lokalny lub zasób w chmurze) i potrafią go uruchomić;
 - d. wgrywają program do zbudowanego modelu robota, uruchamiają, obserwują i testują działanie robota. Wprowadzają modyfikacje w programie, poprawiają ewentualne błędy.

Faza III. Podsumowanie

Uczniowie analizują przedstawione przykłady **prostych programów zapisanych w języku Python oraz omawiają ich działanie**. Nauczyciel ocenia stopień zrozumienia **zależności między zapisanym kodem a danymi zwracanymi przez czujniki po uruchomieniu programu**.

Uczniowie wspólnie wskazują najważniejsze elementy składni języka Python oraz omawiają **sposób wykorzystania bibliotek i modułów obsługujących czujniki**, a także najczęściej pojawiające się błędy w zapisie kodu i sposoby ich poprawiania.

Kryteria oceniania:

Ocena dopuszczająca:

- uczeń rozpoznaje podstawowe elementy programu w języku Python,
- potrafi wskazać polecenia odpowiedzialne za odczytywanie danych z czujników oraz wyświetlanie informacji w konsoli.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Ocena dostateczna:

- uczeń analizuje przykładowy program i opisuje jego działanie,
- rozumie zależność między zapisanym kodem a działaniem programu,
- rozpoznaje podstawowe biblioteki i moduły wykorzystywane do obsługi czujników,
- wskazuje podstawowe błędy składniowe.

Ocena dobra:

- uczeń samodzielnie modyfikuje przykładowy program,
- poprawia błędy związane z zapisem poleceń, nawiasami lub cudzysłowami,
- testuje działanie programu i analizuje dane zwracane przez czujniki po wprowadzonych zmianach.

Ocena bardzo dobra:

- uczeń samodzielnie tworzy poprawny program w języku Python,
- wyjaśnia działanie poszczególnych elementów programu,
- wykorzystuje biblioteki i moduły do obsługi czujników LEGO SPIKE Prime,
- proponuje własne rozwiązania i potrafi je uzasadnić,
- samodzielnie analizuje i poprawia błędy w kodzie.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, np.:

1. Co należy zrobić, aby wyświetlić wynik działania czujnika w konsoli programu?
 - a. Użyć funkcji `beep()`
 - b. Użyć funkcji `print()`**
 - c. Użyć funkcji `write()`
 - d. Użyć funkcji `clear()`
2. W jakiej jednostce czujnik odległości zwraca wynik?
 - a. W milimetrach**
 - b. W centymetrach
 - c. W metrach
 - d. W procentach
3. Co zwraca czujnik koloru w trybie rozpoznawania koloru?
 - a. Nazwę koloru w języku angielskim
 - b. Wartość temperatury
 - c. Prędkość robota
 - a. Kod koloru**

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Faza V. Dodatkowe, praca domowa (dla chętnych).

Zadania dodatkowe (dla chętnych).

Zadanie 1: Odczytujemy dane z czujników

Utwórz prosty program w języku Python, który odczyta dane z wybranego czujnika LEGO SPIKE Prime i wyświetli wynik w konsoli programu.

Program powinien:

- zawierać poprawnie zaimportowane biblioteki i moduły,
- odczytywać dane z wybranego czujnika (odległości, koloru lub siły),
- wyświetlać wynik działania w konsoli za pomocą funkcji **print()**,
- być zapisany poprawnie pod względem składni,
- mieć logiczną kolejność wykonywania poleceń.

Opcjonalnie możesz rozbudować program tak, aby:

- odczytywał dane z więcej niż jednego czujnika,
- wyświetlał dodatkowe komunikaty tekstowe w konsoli,
- prezentował wartości w różnych jednostkach (np. mm i cm),
- wykorzystywał dodatkowe funkcje czujników,
- był estetycznie zapisany i łatwy do odczytania.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 8 - *Czy można w języku Python programować szkolne roboty?*

Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
 - Nauczyciel omawia podstawowe zagadnienia związane z **programowaniem czujników w języku Python**:
 - wyjaśnia, że **robot może pobierać dane z otoczenia za pomocą czujników**,
 - przedstawia przykłady **bibliotek i modułów wykorzystywanych do obsługi czujników LEGO SPIKE Prime**,
 - omawia znaczenie **poprawnego odczytywania i interpretowania danych zwracanych przez czujniki**.
 - Wprowadzenie do zadania i przypomnienie podstaw pracy w LEGO SPIKE Prime:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- omówienie środowiska programowania Python w aplikacji LEGO SPIKE App,
- przedstawienie **bibliotek i modułów odpowiedzialnych za obsługę czujników**,
- przedstawienie zależności: **zapis kodu >> uruchomienie programu >> odczyt danych z czujników**,
- analiza przykładowych programów wykorzystujących **czujnik odległości, czujnik koloru oraz czujnik siły**.
- Praca uczniów:
 - analiza przedstawionych przykładów programów **wykorzystujących czujniki**,
 - omawianie działania poszczególnych linii kodu,
 - **testowanie działania czujników po uruchomieniu programu**,
 - wyszukiwanie i poprawianie błędów w kodzie,
 - proponowanie własnych sposobów **prezentacji danych w konsoli programu**.
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów (**analiza poprawności działania programu, odczytu danych z czujników oraz umiejętności interpretowania wyników działania programu**).
- Zadanie pracy domowej (opcjonalnie) – napisanie **prostego programu odczytującego dane z wybranego czujnika oraz wyświetlającego wynik działania w konsoli programu**.

Załącznik nr 2 – Realizacja zadania głównego podczas lekcji nr 8 - *Czy można w języku Python programować szkolne roboty?*

Zadanie 1. Przygotowanie do realizacji zadania głównego

Uczniowie, pracując w zespołach lub indywidualnie, realizują zadanie polegające na **poznaniu sposobu programowania czujników w języku Python oraz odczytywaniu danych z czujników w środowisku LEGO SPIKE Prime**. Szczególny nacisk położony jest na **poprawne wykorzystanie bibliotek i modułów, analizę danych zwracanych przez czujniki oraz interpretację wyników działania programu**.

Uczniowie opracowują i zapisują rozwiązanie problemu w postaci prostych programów tekstowych, które umożliwią **odczytywanie danych z czujników oraz wyświetlanie wyników w konsoli programu**. W tym celu:

- analizują przykładowe programy zapisane w języku Python,
- określają cel programu oraz przewidują możliwe **błędy składniowe i błędy związane z odczytem danych z czujników**,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

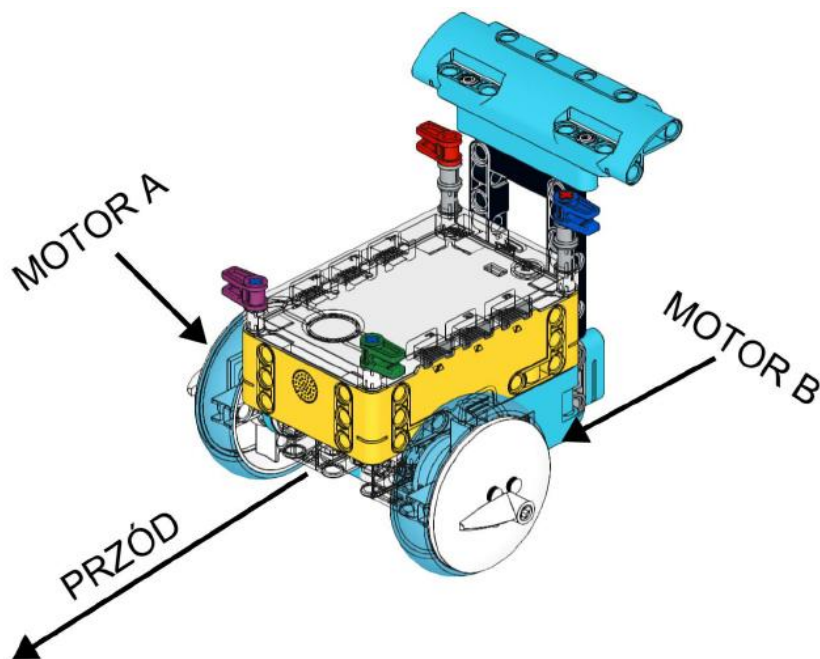
- planują kolejność wykonywania instrukcji w programie,
- opracowują algorytm w uporządkowanej formie,
- zapisują go jako prosty program w języku Python **wykorzystujący czujniki i odpowiednie biblioteki**,
- testują działanie programu i analizują dane zwracane przez czujniki,
- wprowadzają poprawki związane z działaniem programu lub odczytem danych,
- doprecyzowują rozwiązanie tak, aby kod był czytelny i poprawny składniowo.

Uczniowie wykorzystują środowisko LEGO SPIKE Prime w trybie Python, zwracając uwagę na zgodność między zapisanym kodem a **wynikami odczytu danych z czujników widocznymi w konsoli**. Szczególną uwagę poświęcają poprawności składni, wykorzystaniu funkcji **print()**, bibliotek i modułów obsługujących czujniki oraz analizie komunikatów o błędach pojawiających się po uruchomieniu programu.

Jeśli jest taka potrzeba, wskazana przez nauczyciela, to uczniowie zbudują robota od podstaw zgodnie z poniższą instrukcją.

Budowa bazowego robota

Uruchom plik **Budowa bazy robota.pdf** znajdujący się na platformie i postępuj wg podanych w pliku kroków. Efektem końcowym będzie fizyczny robot gotowy do programowania. **Jeśli konstrukcja jest gotowa – idź dalej.**



Pudełka niezbędne do realizacji lekcji: **9,11,12,13,14,15,16,17**

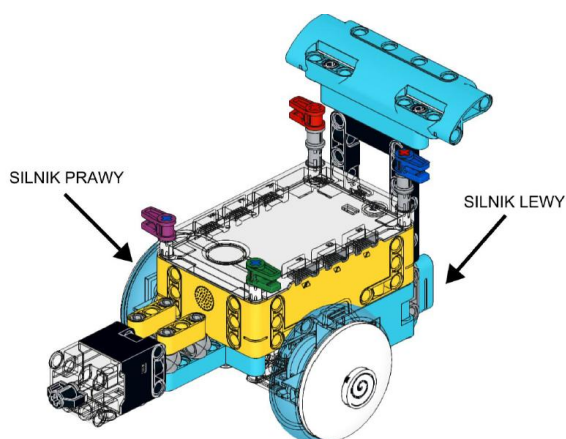
Pudełka z kolorowymi klockami niezbędnymi do realizacji lekcji: **26,27,28,29,30**

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

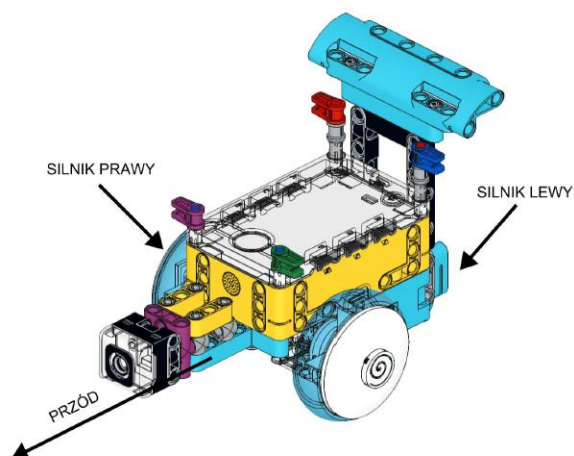


Dodatkowe instrukcje niezbędne do realizacji lekcji:

- [czujnikDotyku.pdf](#)

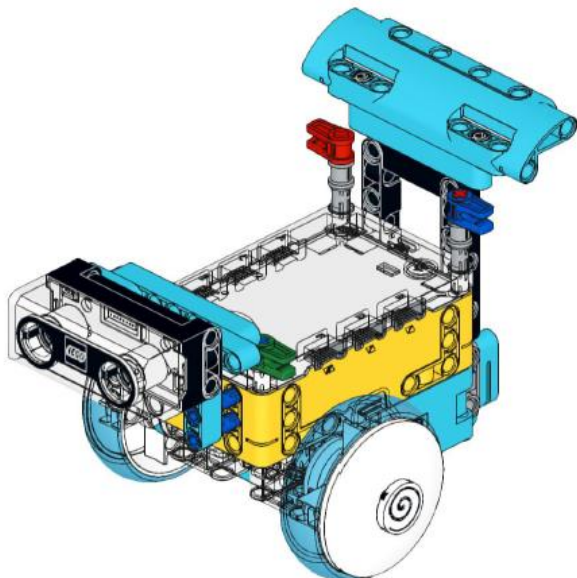


- [czujnikKoloru.pdf](#)



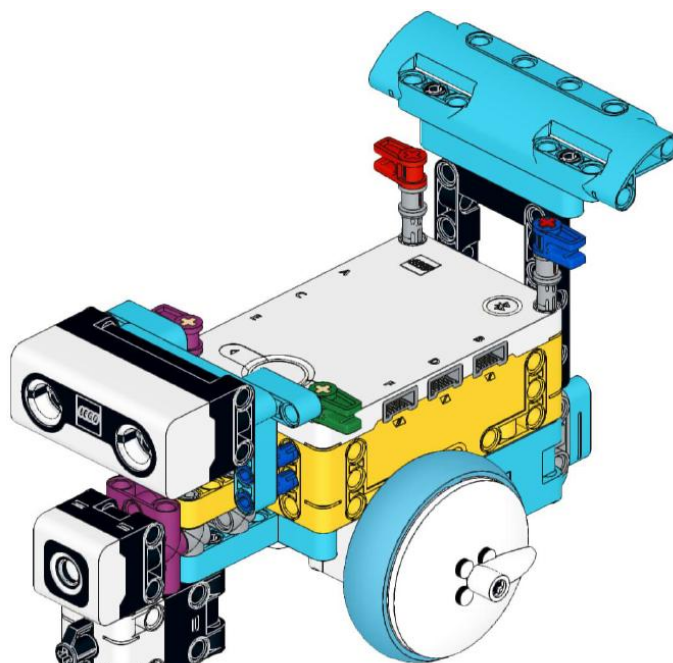
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- **[czujnikOdleglosci.pdf](#)**



Możesz także, jeżeli nauczyciel będzie tego wymagał, zbudować robota wyposażonego we wszystkie czujniki jednocześnie. W takim przypadku skorzystaj z instrukcji:

Robot z wszystkimi czujnikami ([wszystkieCzujniki.pdf](#))

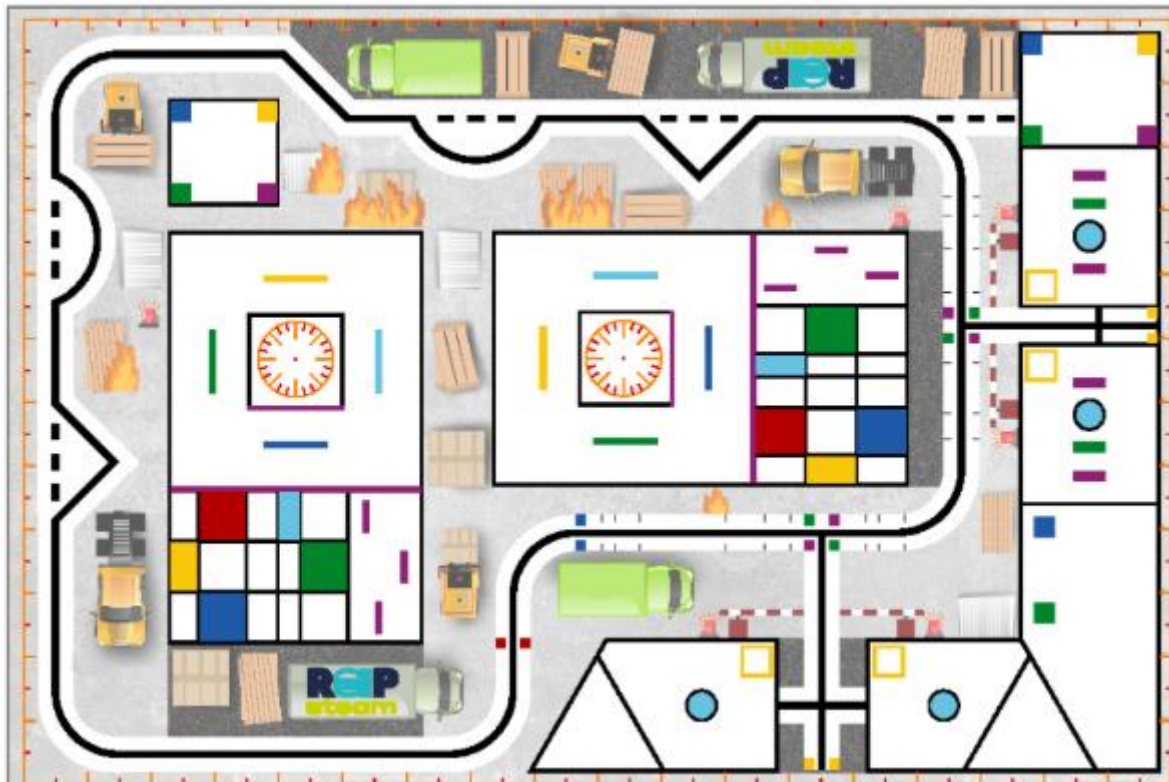


Dodatkowe instrukcje niezbędne do realizacji lekcji:

- **Brak**

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Do realizacji scenariusza zajęć będziesz potrzebował również matę:



Zadanie 2. Konfiguracja środowiska.

Uruchamianie aplikacji LEGO Spike

1. Włącz przeglądarkę internetową dostępną na twoim urządzeniu
2. Wpisz w przeglądarce adres: <https://spike.legoeducation.com/> Pojawi się okno wyboru wersji LEGO SPIKE. Wybierz Spike Prime

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Select your SPIKE™ solution



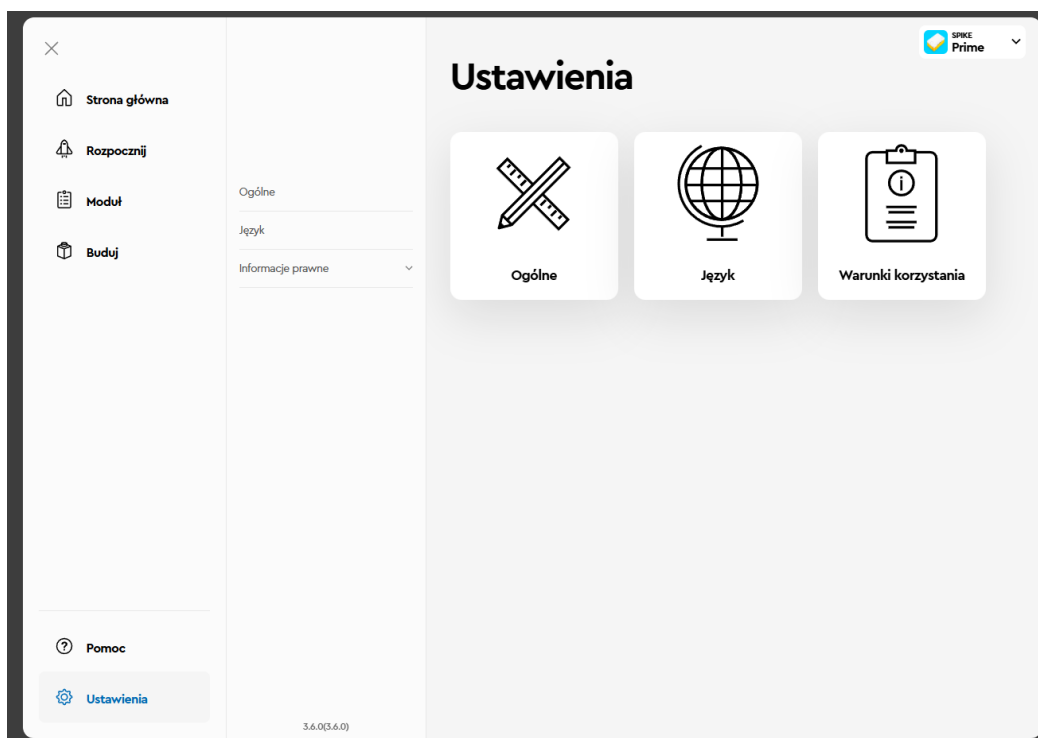
LEGO education

- Po wybraniu opcji SPIKE Prime pojawia się menu główne. Jeśli występuje język polski, należy zmienić go na język angielski.

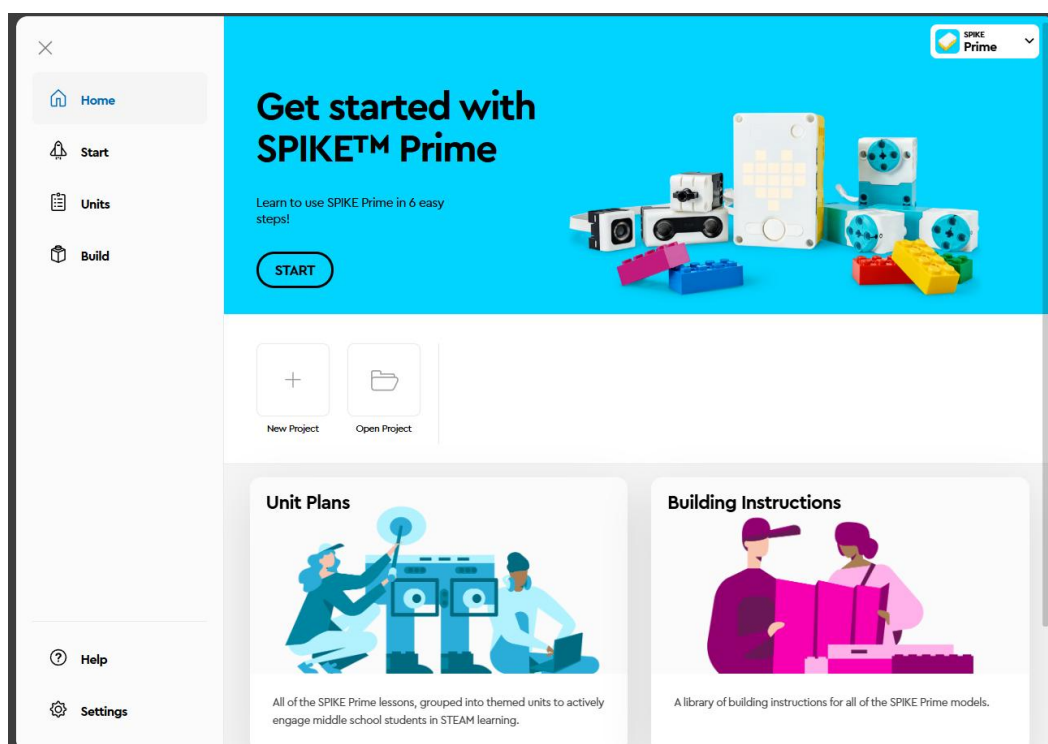
Jeżeli zaś wszystko jest w **języku angielskim**, to przejdź do kroku 6.

Aby zmienić język, w lewym dolnym rogu menu głównego kliknij przycisk **Ustawienia**.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

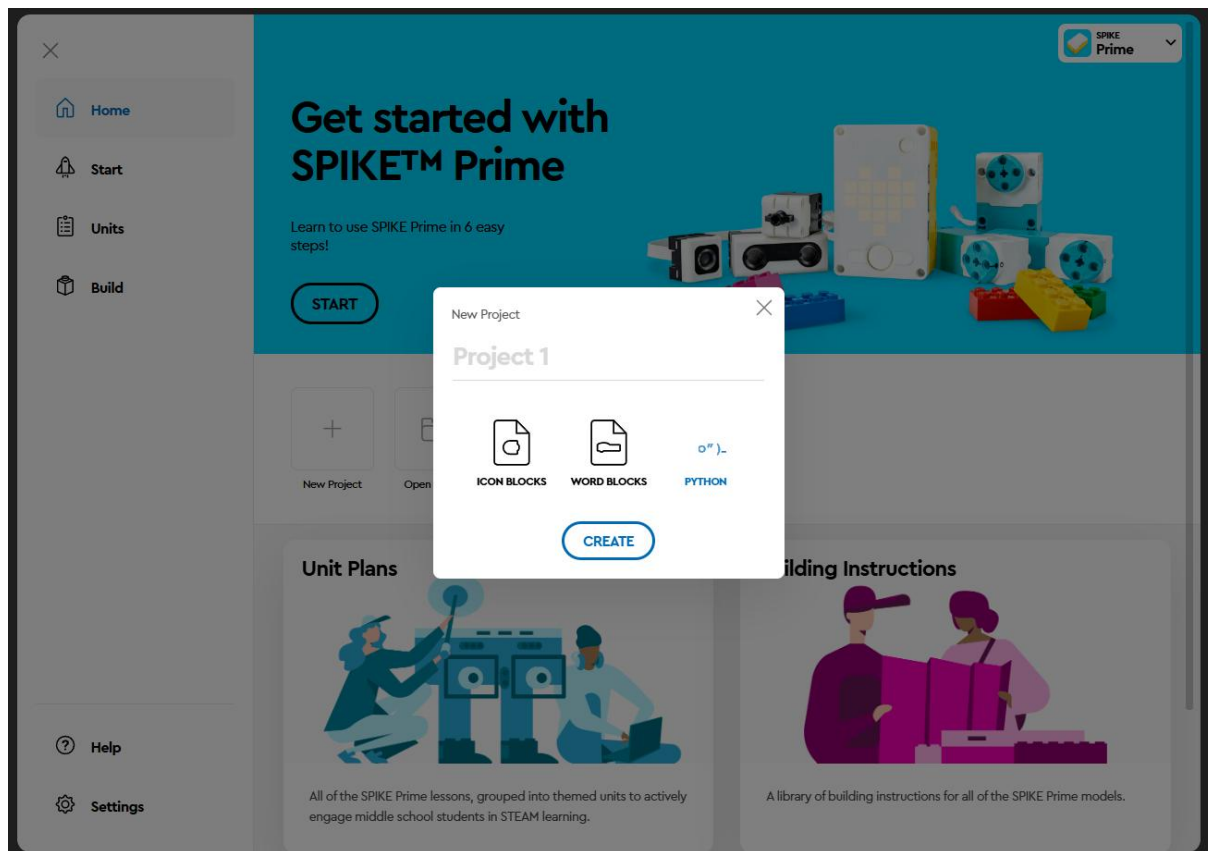


4. W nowym oknie kliknij **globus podpisany „Language”**.
5. **Wybierz język: United States (English)**. Strona przeładuje się i język angielski będzie dostępny.
6. Kliknij po lewej przycisk **Home**. Pojawi się okno jak poniżej

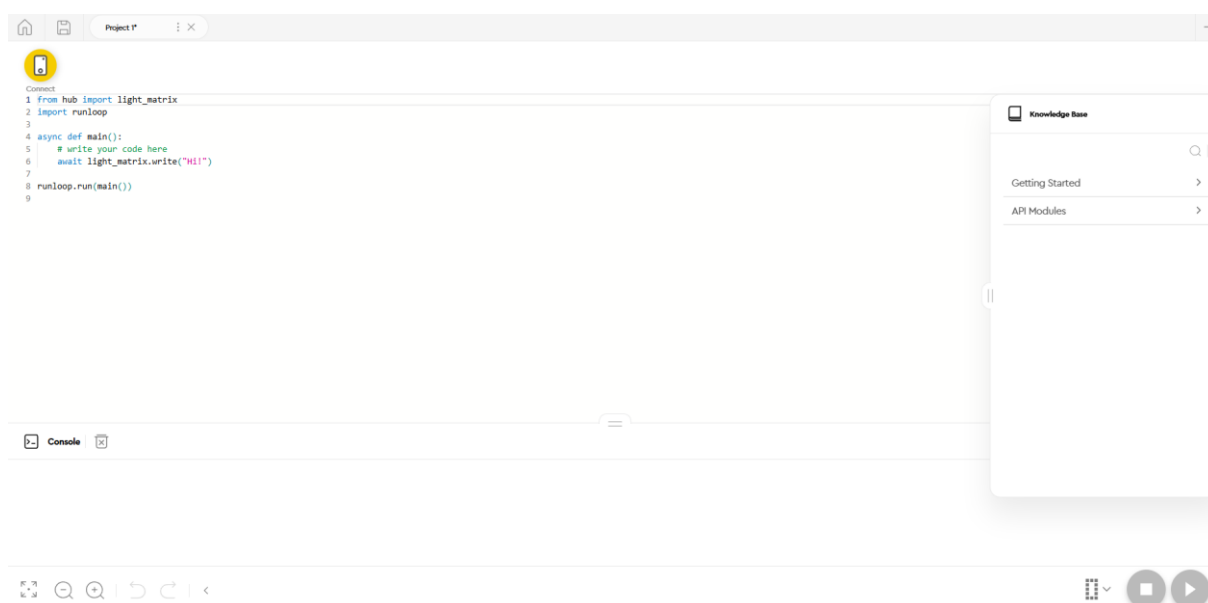


Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

7. Kliknij na środku przycisk **New Project**. Pojawi się nowe okno jak poniżej



8. Wybierz opcję **Python** oraz podaj nazwę projektu. Kiedy wszystko zostanie poprawnie wybrane, wciśnij przycisk **Create**.

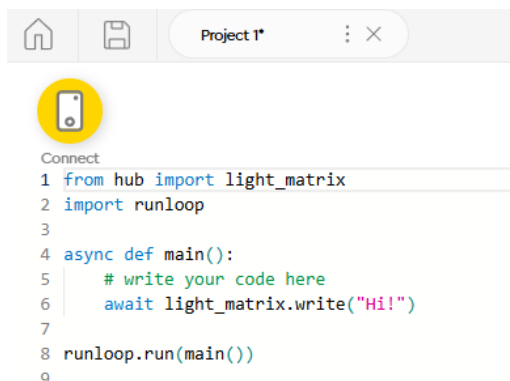


Wszystko jest gotowe do programowania. Pora na podłączenie kontrolera Spike Hub.

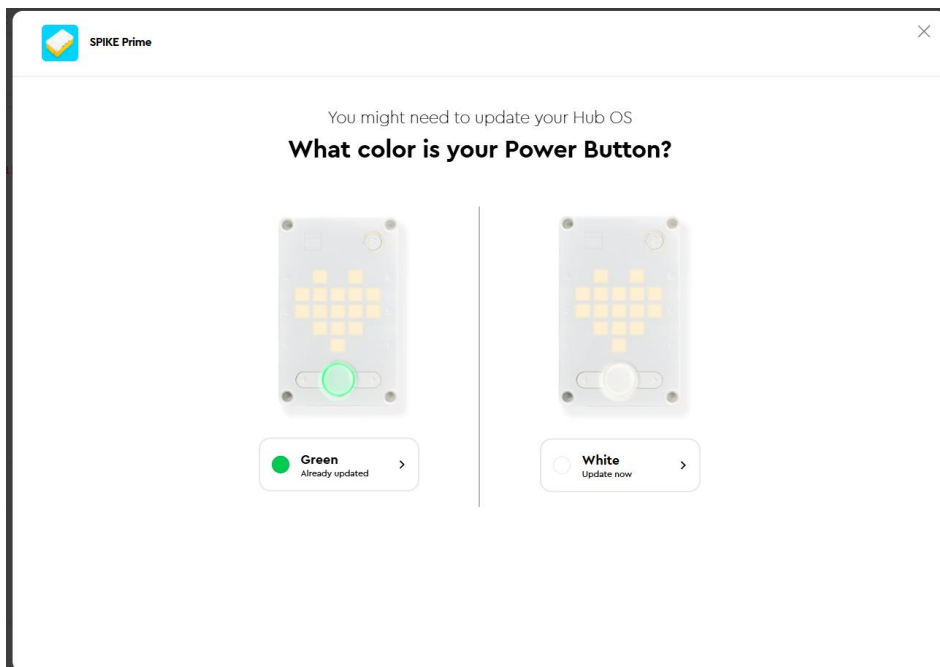
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 3. Podłączenie kontrolera Spike Hub

1. W głównym programie kliknij przycisk **Connect** znajdujący się w górnym rogu, tuż pod nazwą twojego projektu. Patrz poniżej.

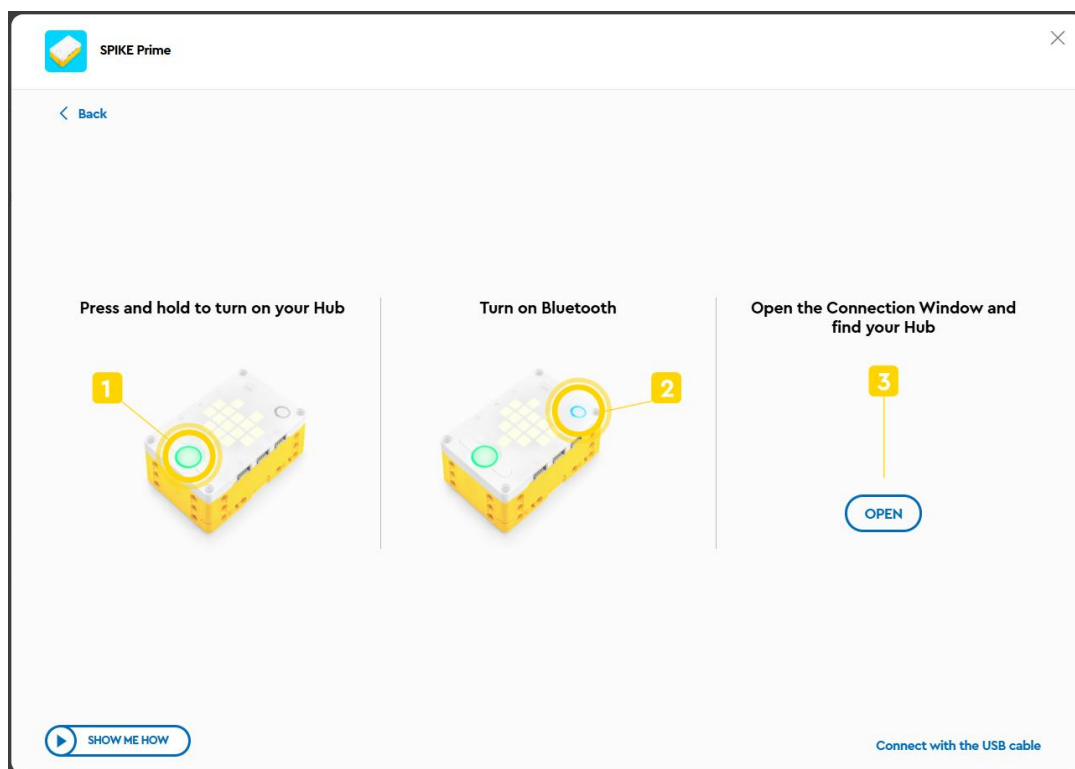


2. Pojawi się nowe okno. **Włącz kontroler Spike Hub** klikając **duży biały przycisk** i poczekaj aż się uruchomi. Następnie wybierz jedną z opcji odpowiadającą stanowi kontrolera Spike Hub – zawsze będzie to **Green – Already updated**.

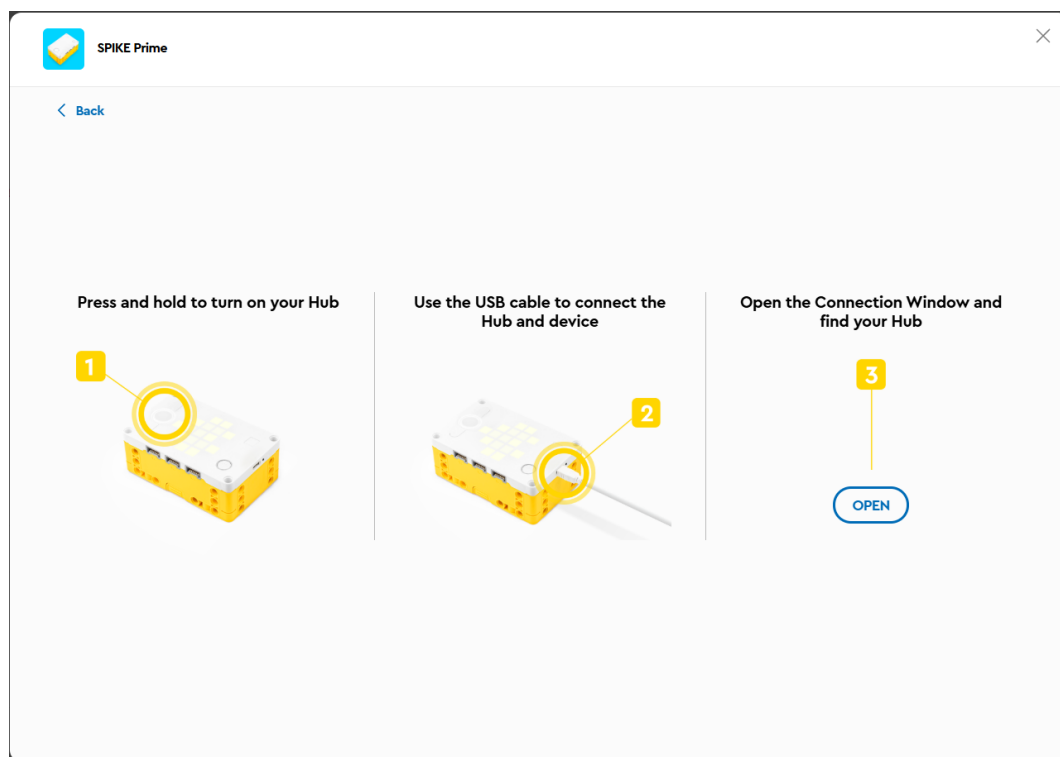


3. Pojawi się nowe okno (patrz poniżej). Wybierz opcję **Connect with USB cable** znajdującą się w prawym dolnym rogu okna.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

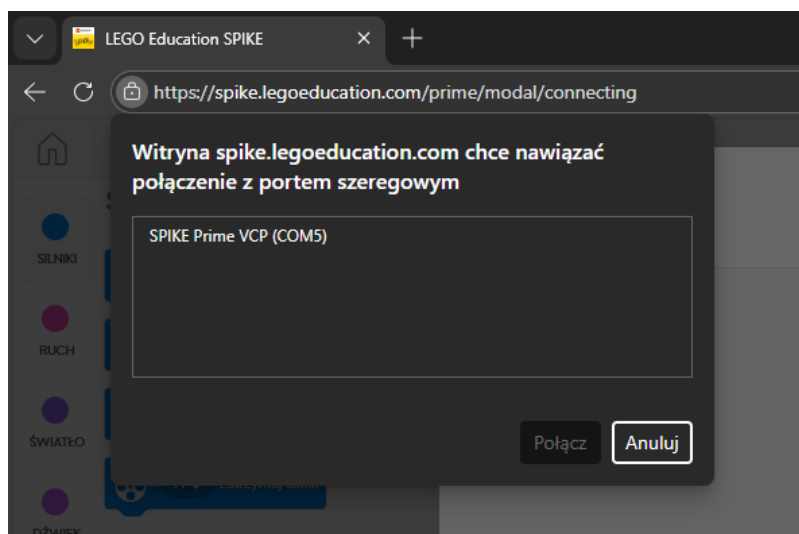


4. Postępuj zgodnie z instrukcją na ekranie. Po przeczytaniu kliknij przycisk **Open**.



5. Pojawi się nowe okno (patrz poniżej)

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



6. Wybierz opcję **SPIKE Prime VCP** i kliknij **Połącz**.

UWAGA – u Ciebie nazwa może nie być identyczna jak na rysunku powyżej. Wybierz tę opcję, która ma w nazwie **Spike Prime**. Innymi rzeczami się nie przejmuj.

Jeśli wszystko zostanie poprawnie połączone, to okna zostaną automatycznie zamknięte, a ikona kontrolera w programie zmieni się na zieloną.



```
1 from hub import light_matrix
2 import runloop
3
4 async def main():
5     # write your code here
6     await light_matrix.write("Hi!")
7
8 runloop.run(main())
9
```

Gratulacje. Twój Spike Hub jest podłączony do komputera. Możesz przystąpić do programowania robota.

W przypadku problemów postępuj wg kroków:

1. Sprawdź, czy kontroler Spike Hub jest włączony i świeci na zielono
 - a. Jeśli kontroler świeci na biało zgłoś to nauczycielowi
 - b. Jeśli kontroler świeci na zielono to podążaj dalej wg instrukcji łączenia powyżej
2. Sprawdź czy kabel USB jest podłączony (Spike Hub <-> Komputer)
3. Sprawdź, czy wybrano odpowiednią nazwę urządzenia – w oknie połączenia z portem szeregowym należy wybrać urządzenie, które w nazwie ma SPIKE Prime.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 4. Realizacja zadania głównego

Dzisiejsze zajęcia wprowadzają do programowania czujników w Pythonie dla LEGO Spike.

Celem lekcji jest poznanie programu oraz nauczenie się obsługi czujników, pobierania z nich informacji oraz prezentacji w konsoli programu od strony programistycznej.

- Zaprogramuj kontroler Spike Hub tak aby czujnik odległości zwrócił odległość w konsoli programu
- Zaprogramuj kontroler Spike Hub tak aby czujnik koloru zwrócił kod koloru oraz natężenie światła w konsoli programu
- Zaprogramuj kontroler Spike Hub tak aby czujnik nacisku (dotyku, siły) zwrócił swój aktualny stan oraz siłę nacisku w konsoli programu

Przykład realizacji zadania – wprowadzenie bibliotek

UWAGA – w materiałach stosowane są komentarze. Na zrzutach ekranu jest to tekst oznaczony jako zielony. Tego tekstu nie przepisuj! To jest tylko wytłumaczenie dla Ciebie, co dzieje się w danej części programu. Przepisuj tylko i wyłącznie tekst, który nie jest zielony. Korzystaj z tabeli. Pamiętaj, że wielkość liter MA ZNACZENIE!

Na początku programu należy dodać odpowiednie odwołania (biblioteki) do modułów kontrolera Spike Hub i czujników. Dodaj moduł, biblioteki:

- port (moduł),
- distance_sensor (biblioteka),
- color_sensor (biblioteka),
- force_sensor (biblioteka),

Aby móc skorzystać z czujnika odległości, niezbędne jest dodanie do programu **biblioteki distance_sensor**. Przechowuje ona wszystkie funkcje niezbędne do pozyskania z niego informacji. W celu pobrania danych z czujnika koloru należy zaimportować **bibliotekę color_sensor**. Daje ona dostęp do wszystkich trzech funkcjonalności czujnika koloru: wykrywania koloru, pomiaru natężenia światła oraz odbicia. Ostatnia **biblioteka** – **force_sensor** – pozwala na pozyskanie danych związanych z czujnikiem dotyku, często nazywanym czujnikiem siły czy nacisku. Wszystkie nazwy są poprawne.

Moduł **portu** z **biblioteki hub** pozwala uzyskać informacje związane z portami na kontrolerze Spike Hub – A, B, C, D, E i F. Moduł **portu** pozwala na odwołanie się do tych konkretnych portów i urządzeń do nich wpiętych. Daje to możliwość wysłania danych (np. komend do silników) lub ich pobrania z danych portów (np. odległości, jaką zwraca czujnik odległości).

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Usuń początek programu i dodaj odpowiednie odwołania. Część programu odpowiedzialna za biblioteki powinna wyglądać następująco:

Zdjęcie	Kod (tekst)
<pre>1 from hub import port 2 import distance_sensor 3 import runloop 4 import color_sensor 5 import force_sensor -</pre>	<pre>from hub import port import distance_sensor import runloop import color_sensor import force_sensor</pre>

W tej chwili cały program powinien wyglądać następująco:

```
1 from hub import port
2 import distance_sensor
3 import runloop
4 import color_sensor
5 import force_sensor
6
7
8 async def main():
9     # write your code here
10
11
12 runloop.run(main())
```

Nie przejmuj się czerwonym podkreśleniem ;) Zajmiemy się tym później...

Pomiar odległości

Uruchom instrukcję **czujnikOdleglosci.pdf**, rozbuduj robota bazowego, a następnie **podłącz czujnik odległości do portu F**.

Zacznijmy od pobrania danych z czujnika odległości. Uwaga – czujnik zwraca odległość w milimetrach! Miej to na uwadze, kiedy będziesz odczytywać z niego informacje i na nich pracować. Wartość, jaką zwróci czujnik, **zapisz w zmiennej**. Do pobrania informacji służy funkcja **distance**. W nawiasie należy podać informacje, do jakiego portu wpięty jest czujnik.

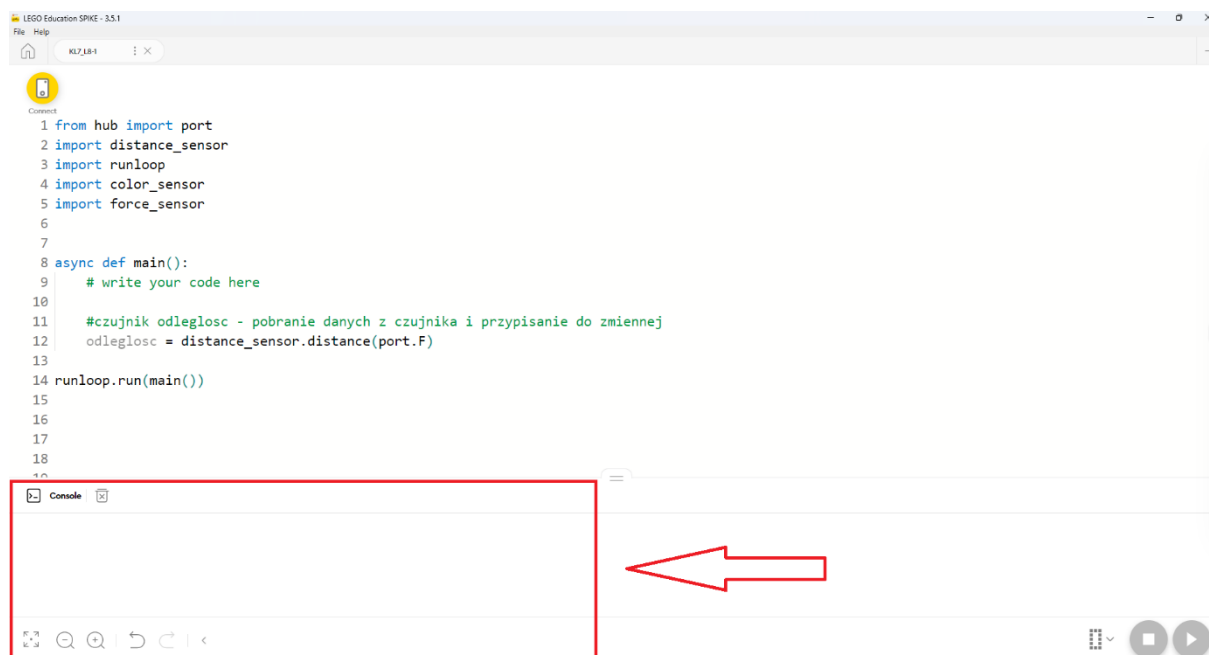
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Dzięki modułowi **port**, który został na początku zaimportowany, możliwe jest pobranie danych o tym, jakie porty posiada **Spike Hub**. Kod odpowiedzialny za pobranie wartości z czujnika prezentuje się następująco:

```
LEGO Education SPIKE - 3.5.1
File Help
K17_L8-1

1 from hub import port
2 import distance_sensor
3 import runloop
4 import color_sensor
5 import force_sensor
6
7
8 async def main():
9     # write your code here
10
11     #czujnik odleglosc - pobranie danych z czujnika i przypisanie do zmiennej
12     odleglosc = distance_sensor.distance(port.F)
13
14 runloop.run(main())
15
```

Po uruchomieniu programu zmienna odległość będzie mieć w sobie informacje związane z odległością od przeszkody, jaką podaje czujnik – w milimetrach. Teraz należy ją wyświetlić. W tej lekcji odległość i inne wartości pojawią się w aplikacji **Spike App** w **konsoli** (dolne okno, patrz niżej):



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Aby informacje zawarte w zmiennej pojawiły się w tym oknie po uruchomieniu programu, niezbędne jest użycie funkcji **print**. Funkcja **print** wypisuje tekst na ekranie aplikacji (nie na matrycy **Spike Hub**!). Spróbuj zatem. Przepisz, uruchom program i zobacz, co się stanie. Pamiętaj, aby w trakcie uruchamiania przyłożyć rękę do czujnika!

```
1 from hub import port
2 import distance_sensor
3 import runloop
4 import color_sensor
5 import force_sensor
6
7
8 async def main():
9     # write your code here
10
11     #czujnik odleglosc - pobranie danych z czujnika i przypisanie do zmiennej
12     odleglosc = distance_sensor.distance(port.F)
13
14     #wypisanie odleglosci w milimetrach - taką wartość domyślnie podaje czujnik
15     print("Odległość w mm: ", odleglosc, "mm")
16
17 runloop.run(main())
```

Wspaniale. Co tu się wydarzyło?

Wyświetlony został tekst „Odległość w mm: 56 mm”.

Dlaczego tak się stało?

Zobacz na budowę funkcji **print**. W nawiasie napisano tekst, używając cytowania (znaki: „”). Oznacza to, że wszystko, co znajduje się w znakach „”, musi być traktowane jako tekst i taki tekst ma się wyświetlić. Ponieważ nie wiemy, jaką odległość zwróci czujnik, należy wywołać zmienną, która tę wartość zawiera. Wykonuje się to, stosując przecinek – oddziela on blok tekstu od zmiennych. Użycie słowa **odleglosc** informuje funkcję **print**, że musi w programie znaleźć zmienną o nazwie **odleglosc**, pobrać z niej wartość i wypisać ją na ekranie. Po wywołaniu zmiennej możliwe jest dalsze wyświetlanie tekstu. Robi się to tak samo. Oddzielono zmienną przecinkiem, a tekst pojawił się w znakach „”. Dzięki temu pojawia się dodatkowy tekst.

Skoro program zwraca poprawną odległość, to jak zrobić, aby wartość otrzymana w konsoli pojawiła się w centymetrach, a nie milimetrach? **Należy wartość w odległości podzielić przez 10.**

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

```

1 from hub import port
2 import distance_sensor
3 import runloop
4 import color_sensor
5 import force_sensor
6
7
8 async def main():
9     # write your code here
10
11     #czujnik odleglosc - pobranie danych z czujnika i przypisanie do zmiennej
12     odleglosc = distance_sensor.distance(port.F)
13
14     #wypisanie odleglosci w milimetrach - taką wartość domyślnie podaje czujnik
15     print("Odległość w mm: ", odleglosc, "mm")
16
17     #wypisanie odleglosci w centymetrach
18     #należy zamienić milimetry na centymetry dzieląc wartość w milimetrach przez 10 (10 mm = 1 cm)
19     print("Odległość w cm: ", odleglosc/10, "cm")
20
21 runloop.run(main())

```

Działania matematyczne w funkcji **print** są w pełni dozwolone. Przejdźmy dalej. Czujnik odległości ma wbudowane diody. Możliwe jest ich uruchomienie za pomocą kodu. Diody imitują oczy. Aby uruchomić diodę, należy użyć funkcji **set_pixel**. W komendzie **set_pixel** (a,b,c,d):

- a – port, na którym podpięty jest czujnik (tutaj port.F)
- b - może być liczbą dodatnią, całkowitą z zakresu od 0 do 3 – traktuj tą wartość jako współrzędną x
- c - może być liczbą dodatnią, całkowitą z zakresu od 0 do 3 – traktuj tą wartość jako współrzędną y
- d - może być liczbą dodatnią, całkowitą z zakresu od 0 do 100 – traktuj to jako moc oświetlenia

Dodaj linie kodu do programu i zobacz, co się wydarzy. Jak tylko program się wykona, spróbuj zrobić tak, aby zapalić wszystkie 4 diody.



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

```
1 from hub import port
2 import distance_sensor
3 import runloop
4 import color_sensor
5 import force_sensor
6
7
8 async def main():
9     # write your code here
10
11     #czujnik odleglosc - pobranie danych z czujnika i przypisanie do zmiennej
12     odleglosc = distance_sensor.distance(port.F)
13
14     #wypisanie odleglosci w milimetrach - taką wartość domyślnie podaje czujnik
15     print("Odległość w mm: ", odleglosc, "mm")
16
17     #wypisanie odleglosci w centymetrach
18     #należy zamienić milimetry na centymetry dzieląc wartość w milimetrach przez 10 (10 mm = 1 cm)
19     print("Odległość w cm: ", odleglosc/10, "cm")
20
21     #zapalenie diody na czujniku odleglosci. Podpowiedź: należy zamieniać wartości 0 i 1 aby zapalić odpowiednią część.
22     distance_sensor.set_pixel(port.F,0,1,100)
23
24     #Jak zrobić aby zapalić wszystkie jednocześnie? albo tylko górne?
25
26 runloop.run(main())
```



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Czujnik koloru

Uruchom instrukcję **czujnikKoloru.pdf**, rozbuduj robota bazowego, a następnie podłącz **czujnik koloru do portu E**.

Przywróć kod do stanu początkowego. Wykasuj wszystko związane z czujnikiem odległości. Twój program powinien wyglądać następująco:

```
1 from hub import port
2 import distance_sensor
3 import runloop
4 import color_sensor
5 import force_sensor
6
7
8 async def main():
9     # write your code here
10
11
12 runloop.run(main())
```

W tej części zajmiesz się **czujnikiem koloru**. Dla Pythona posiada on dwa tryby działania:

- **Rozpoznawanie koloru** – zwraca kod (ten sam który widoczny jest w górnej części okna przy danych z czujników)
- **Odbicie światła** – wartość w procentach (niska wartość: ciemno, wysoka wartość: jasno), zakres od 0 do 100

Prace zaczniemy od odczytania kodu koloru, jaki czujnik zwraca. Najpierw, tak jak w przypadku czujnika odległości, zrób zmienną o nazwie kolor, która będzie przechowywać kod koloru, który czujnik rozpoznał. W funkcji **color()** należy w nawiasie podać port, z którego ma pobrać informacje. Następnie, używając funkcji **print()**, wyświetl ją na ekranie. Rozwiązanie wygląda następująco:

```
1 from hub import port
2 import distance_sensor
3 import runloop
4 import color_sensor
5 import force_sensor
6
7
8 async def main():
9     # write your code here
10
11     #pobranie kodu koloru z czujnika. LEGO identyfikuje kolory poprzez swoje własne kody np. -1 to brak koloru, 7 to kolor żółty
12     kolor = color_sensor.color(port.E)
13     #wypisanie kodu koloru na konsolę. W przyszłych lekcjach dowiesz się jak sprawdzać kody i kolory oraz reagować na nie
14     print("Kod koloru: ", kolor)
15
16 runloop.run(main())
```

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przyłóż do czujnika klocek z kolorem (wybrany przez Ciebie) i uruchom program. Zobacz jaki wynik otrzymasz w konsoli.

W przykładzie wykorzystano czerwony klocek z pudełka nr 27. Kod koloru zwracany przez czujnik w konsoli zgadza się z kodem, który wskazuje aplikacja w swoim module.



```
1 from hub import port
2 import distance_sensor
3 import runloop
4 import color_sensor
5 import force_sensor
6
7
8 async def main():
9     # write your code here
10
11     #pobranie kodu koloru z czujnika. LEGO identyfikuje kolory poprzez swoje własne kody np. -1 to brak koloru, 7 to kolor żółty
12     kolor = color_sensor.color(port.E)
13     #wypisanie kodu koloru na konsolę. W przyszłych lekcjach dowiesz się jak sprawdzać kody i kolory oraz reagować na nie
14     print("Kod koloru: ", kolor)
15
16 runloop.run(main())
17
18
19
20
21
```

Console

3:59:51 PM: Compiled

Kod koloru: 9

Poeksperymentuj z różnymi kolorami. Co, jeśli nie ma koloru (nic nie widzi)?

Druga część to pomiar natężenia światła. Służy do tego funkcja **reflection()**. Ona też w nawiasie potrzebuje tylko i wyłącznie informacji, na którym porcie wpięty jest czujnik. Zapisz wynik do zmiennej odbicie i wyświetl w konsoli. Zadanie realizuje następujący program:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



```

1 from hub import port
2 import distance_sensor
3 import runloop
4 import color_sensor
5 import force_sensor
6
7
8 async def main():
9     # write your code here
10
11     #pobranie kodu koloru z czujnika. LEGO identyfikuje kolory poprzez swoje własne kody np. -1 to brak koloru, 7 to kolor żółty
12     kolor = color_sensor.color(port.E)
13     #wypisanie kodu koloru na konsolę. W przyszłych lekcjach dowiesz się jak sprawdzać kody i kolory oraz reagować na nie
14     print("Kod koloru: ", kolor)
15
16     #pomiar odbicia światła od podłoża. Czujniki najczęściej używane w robotach typu line follower - jeżdżących po linii
17     odbicie = color_sensor.reflection(port.E)
18     #wypisanie wartości odbicia światła na ekran konsoli. Wartość podawana jest w procentach gdzie wartości bliskie 0 to ciemno a bliżej 100 to jasno
19     print("Poziom oświetlenia: ", odbicie, "%")
20
21 runloop.run(main())
22

```

4:04:12 PM: Compiled

Kod koloru: 9
Poziom oświetlenia: 86 %

Zwróć uwagę, że dla czerwonego klocka wartość oświetlenia wynosi 86%. To oznacza jasny obiekt. Teraz pora na testy. Przyłóż czujnik do różnych powierzchni. Jasnych, ciemnych, kolorowych i sprawdź, jakie wartości zwraca Ci w konsoli funkcja **reflection()**.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Czujnik siły (nacisku, dotyku)

Ostatnim czujnikiem na dzisiejszej lekcji jest **czujnik siły**. Uruchom instrukcję **czujnikDotyku.pdf**, rozbuduj robota bazowego, a następnie podłącz **czujnik dotyku do portu C**.

Możesz zostawić poprzedni kod albo go usunąć. Przykłady prezentowane w tej części lekcji będą bazować na podstawowym, czystym kodzie:

```
1 from hub import port
2 import distance_sensor
3 import runloop
4 import color_sensor
5 import force_sensor
6
7
8 async def main():
9     # write your code here
10
11
12 runloop.run(main())
```

Moduł ten rozpocznij od pobrania stanu przycisku. Funkcja **pressed()** zwraca informację, czy przycisk jest wciśnięty czy puszczone. Jako parametr (wartości w nawiasach) przyjmuje informacje o porcie. W tym zakresie funkcja **pressed()** nie różni się w budowie niczym od poznanych już przez Ciebie funkcji.

Po pobraniu wartości należy wypisać ją na ekran konsoli. Program wygląda następująco:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

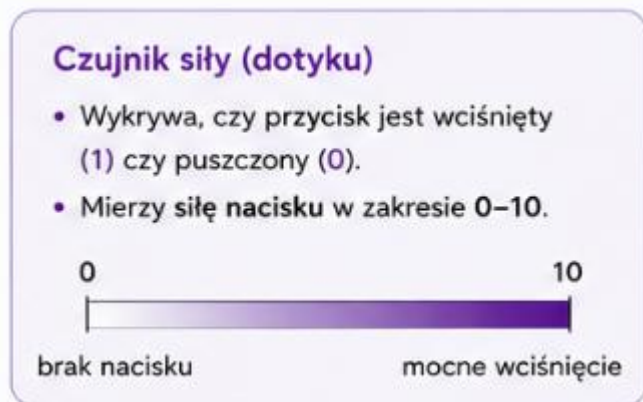


```
1 from hub import port
2 import distance_sensor
3 import runloop
4 import color_sensor
5 import force_sensor
6
7
8 async def main():
9     # write your code here
10
11     #czujnik siły/dotyku ma dwa tryby. Pomiar siły nacisku oraz status wciśnięty/puszczony. Najpierw sprawdźmy status. Pobieramy stan
12     status_czujnika = force_sensor.pressed(port.C)
13     #wypisanie stanu na ekran konsoli. Przed uruchomieniem programu trzymaj czujnik naciśnięty lub puszczony i sprawdź jakie wartości otrzymujesz. Uruchom parę razy
14     #False - przycisk puszczony, True - przycisk wciśnięty
15     print("Status czujnika dotyku: ", status_czujnika)
16
17 runloop.run(main())
18
19
20
21
22
```

4:09:46 PM: Compiled
Status czujnika dotyku: True

Czujnik może zwrócić dwa stany:

- **True** – przycisk jest wciśnięty
- **False** – przycisk jest puszczony



Uruchom program parę razy i zobacz, jakie wartości zwraca czujnik w różnych sytuacjach.

Ostatnim zadaniem jest określenie, z jaką siłą jest wciskany silnik. Pozwala to wykrywać np. to, jak mocno robot napiera na przeszkodę. Do określenia siły służy funkcja **force()**. Tak jak funkcja **pressed()** przyjmuje w nawiasie tylko port, na którym podłączony jest czujnik. Funkcja zwraca wartość od 0 do 10, z czego 0 to brak nacisku, a 10 to mocne wciśnięcie. Kod realizujący zadanie:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



```

1 from hub import port
2 import distance_sensor
3 import runloop
4 import color_sensor
5 import force_sensor
6
7
8 async def main():
9     # write your code here
10
11     #czujnik siły/dotyku ma dwa tryby. Pomiar siły nacisku oraz status wciśnięty/puszczony. Najpierw sprawdźmy status. Pobieramy stan
12     status_czujnika = force_sensor.pressed(port.C)
13     #wypisanie stanu na ekran konsoli. Przed uruchomieniem programu trzymaj czujnik naciśnięty lub puszczone i sprawdź jakie wartości otrzymujesz. Uruchom parę razy
14     #False - przycisk puszczone, True - przycisk wciśnięty
15     print("Status czujnika dotyku: ", status_czujnika)
16
17     #pomiar siły - wartość zwracana jest w zakresie od 0 (brak nacisku na silnik) do 100 (maksymalnie wciśnięty przycisk)
18     sila = force_sensor.force(port.C)
19     #wypisanie wartości siły. Przed uruchomieniem programu wciśnij czujnik mocniej lub słabiej i sprawdź wyniki. Uruchom parę razy
20     print("Siła nacisku: ", sila)
21
22 runloop.run(main())
23

```

4:14:45 PM: Compiled

Status czujnika dotyku: True
Siła nacisku: 100

Uwaga – wartość w górnej części programu może zwracać inną wartość niż ta, którą podaje funkcja **force()**. Dlaczego? Funkcja **force()** zwraca bardzo dokładne dane, zaś wynik pokazywany w górnej części ekranu w sekcji czujników jest wartością zaokrągloną. Zobacz różnicę:



```

1 from hub import port
2 import distance_sensor
3 import runloop
4 import color_sensor
5 import force_sensor
6
7
8 async def main():
9     # write your code here
10
11     #czujnik siły/dotyku ma dwa tryby. Pomiar siły nacisku oraz status wciśnięty/puszczony. Najpierw sprawdźmy status. Pobieramy stan
12     status_czujnika = force_sensor.pressed(port.C)
13     #wypisanie stanu na ekran konsoli. Przed uruchomieniem programu trzymaj czujnik naciśnięty lub puszczone i sprawdź jakie wartości
14     #False - przycisk puszczone, True - przycisk wciśnięty
15     print("Status czujnika dotyku: ", status_czujnika)
16
17     #pomiar siły - wartość zwracana jest w zakresie od 0 (brak nacisku na silnik) do 100 (maksymalnie wciśnięty przycisk)
18     sila = force_sensor.force(port.C)
19     #wypisanie wartości siły. Przed uruchomieniem programu wciśnij czujnik mocniej lub słabiej i sprawdź wyniki. Uruchom parę razy
20     print("Siła nacisku: ", sila)
21
22 runloop.run(main())
23

```

4:16:17 PM: Compiled

Status czujnika dotyku: True
Siła nacisku: 48

Program pokazuje, że czujnik zwraca wartość **5N**. Konsola jednak pokazuje, że czujnik zwraca wartość **48**. Jak można zauważyć, funkcja **force()** jest dokładniejsza i warto wykorzystywać jej dane.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

To koniec wprowadzenia do czujników. Teraz pora na Ciebie. Wykonaj testy i różne symulacje. Sprawdź, co się wydarzy, kiedy kilka czujników jest podłączonych jednocześnie.

Powodzenia!!

Możliwe rozszerzenia zadania

1. Wykorzystanie jednocześnie kilku czujników w jednym programie,
2. Wykorzystanie instrukcji warunkowych do reagowania na dane z czujników,
3. Uruchamianie określonych działań robota po wykryciu przeszkody lub koloru.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 2: Odkrywamy tajemnice Kodu... Roboty w uścisku Pythona

Temat lekcji nr 9 - Odważnie naprzód! Najlepiej slalomem - zadanie dotyczące mobilności robota. Program w języku Python

Treści nauczania (wymagania szczegółowe - podstawa programowa) II. 1) II. 2); IV. 1)

Cel lekcji:

- **Wprowadzenie do programowania ruchu robota mobilnego** – uczniowie poznają zasady sterowania ruchem robota w języku Python oraz sposoby wykorzystania bibliotek odpowiedzialnych za mobilność robota w LEGO SPIKE Prime.
- **Analiza działania programu** – uczniowie analizują fragmenty kodu odpowiedzialne za jazdę robota, skręcanie oraz omijanie przeszkód i określają, w jaki sposób poszczególne polecenia wpływają na ruch robota podczas przejazdu slalomem.
- **Tworzenie algorytmów i programów sterujących ruchem robota** – uczniowie opracowują algorytmy oraz zapisują je w postaci programów tekstowych umożliwiających robotowi przejazd slalomem pomiędzy przeszkodami.
- **Testowanie działania programu** – uczniowie uruchamiają program, obserwują sposób poruszania się robota oraz wprowadzają poprawki w celu uzyskania płynnego i poprawnego przejazdu slalomem.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- **Uczeń dostrzega, że działanie programu i ruch robota zależy od poprawności zapisanych poleceń:**
 - rozumie, że robot mobilny wykonuje instrukcje zgodnie z zapisanym kodem programu oraz realizuje zaplanowaną trasę przejazdu,
 - obserwuje działanie programu napisanego w języku Python i formułuje wnioski dotyczące poprawności ruchu robota podczas wykonywania slalomu.
- **Uczeń analizuje działanie programu i jego efekty:**

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- sprawdza poprawność działania programu sterującego ruchem robota mobilnego,
- identyfikuje błędy składniowe i logiczne oraz modyfikuje program poprzez zmianę parametrów ruchu, skrętu lub kolejności wykonywania poleceń,
- wykorzystuje biblioteki i moduły odpowiedzialne za sterowanie ruchem robota w LEGO SPIKE Prime.
- **Uczeń rozwija umiejętność testowania i doskonalenia rozwiązań:**
 - wykonuje próby działania programu i analizuje sposób poruszania się robota podczas przejazdu slalomem,
 - wprowadza poprawki w celu uzyskania płynnego i poprawnego przejazdu robota między przeszkodami,
 - testuje różne wartości parametrów ruchu i skrętu w celu poprawy dokładności działania robota.

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści kształcenia będą realizowane za pomocą robotyki edukacyjnej LEGO® Education SPIKE™ Prime.

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna RaP STEAM,
- praca na komputerze,
- praca przy zestawie robotyki edukacyjnej,
- pogadanka.

Środki dydaktyczne

- zestaw PC z dostępem do Internetu,
- zestaw robotyki edukacyjnej LEGO® Education SPIKE™ Prime oraz zestaw rozszerzony LEGO® Education SPIKE™ Prime Expansion Kit,
- rzutnik multimedialny,
- e-podręcznik.

Formy pracy

- Praca indywidualna lub grupowa.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przebieg lekcji:

Faza I. Wprowadzenie

Na początku lekcji nauczyciel wprowadza uczniów w zagadnienia związane z **programowaniem mobilności robota w języku Python**, wyjaśniając, że robot wykonuje ruch zgodnie z instrukcjami zapisanymi w programie. Następnie odnosi się do zadania lekcyjnego, polegającego na zaprogramowaniu robota tak, aby przejechał slalom pomiędzy przeszkodami w środowisku LEGO SPIKE Prime.

Nauczyciel przedstawia cel i zakres zadania. Prezentuje uczniom środowisko programowania w języku Python w aplikacji LEGO SPIKE App oraz omawia podstawowe **biblioteki i polecenia odpowiedzialne za sterowanie ruchem robota, takie jak jazda do przodu, zatrzymanie oraz wykonywanie skrętów**. Następnie przekazuje uczniom niezbędne informacje do realizacji lekcji.

Uczniowie zapoznają się ze sposobami zapisu poleceń odpowiedzialnych za ruch robota oraz obserwują, **jak robot po uruchomieniu programu wykonuje kolejne instrukcje i porusza się zgodnie z przygotowaną trasą przejazdu**.

Następnie uczniowie zdobywają umiejętności obejmujące:

- określenie problemu (w jaki sposób zaprogramować robota do przejazdu slalomem),
- analizę działania programu sterującego ruchem robota,
- planowanie kolejności ruchów i skrętów robota,
- obserwację działania robota po uruchomieniu programu,
- testowanie programu i analizowanie sposobu przejazdu robota między przeszkodami,
- wprowadzanie poprawek związanych z parametrami ruchu i skrętu robota.

Na zakończenie tej fazy lekcji nauczyciel podaje temat do zapisania w zeszycie i przechodzi do fazy II – realizacji praktycznej.

Faza II. Część zasadnicza - realizacja

W tej fazie lekcji nauczyciel zapewnia uczniom praktyczne ćwiczenia, które pozwolą eksperymentować z koncepcjami programowania. W szczególności obejmować to będzie używanie blokowych języków programowania oraz zestawu do robotyki.

1. Nauczyciel objaśnia sposób pracy na lekcji – we wskazanym przez niego momencie uczniowie będą wykonywać kolejne etapy prowadzące do rozwiązania zadania. Zdolniejsi uczniowie spróbują wykonać je samodzielnie. Słabsi wykonają kolejne kroki razem z nauczycielem, w oparciu o przygotowaną do zajęć instrukcję dostępną w e-podręczniku lub materiał multimedialny. W razie potrzeby będą mogli dopytać nauczyciela. Nauczyciel na bieżąco sprawdza postępy uczniów.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

2. Bazując na posiadanych zasobach, zapewnij każdemu uczniowi lub zespołowi uczniów kompletny zestaw do robotyki (tj. LEGO® Education SPIKE™).
3. Zestawy te pozwalają uczniom budować własne roboty i programować je do wykonywania określonych zadań. Nauczyciel zaczyna od prostych projektów, takich jak zbudowanie i zaprogramowanie robota, który może wykonać proste czynności, które wprowadzą robota w ruch.
4. Uczniowie budują bazowego robota według załączonej instrukcji (załącznik **Budowa_bazy_robota.pdf** do lekcji).
5. Uczniowie włączają program **SPIKE**, poprzez który będą programować robota.
6. Nauczyciel, wykorzystując e-platformę, demonstruje na tablicy multimedialnej działanie programu i robota, uruchamiając odpowiedni moduł i materiał multimedialny.
7. Prezentowany materiał multimedialny, zrealizowany w oparciu o szczegółowy scenariusz (załącznik do lekcji), oprócz waloru teoretycznego stanowi również podstawę przebiegu części praktycznej lekcji. W szczególności, uczniowie:
 - a. budują prosty model bazowego robota w oparciu o zestaw do robotyki (tj. LEGO® Education SPIKE™),
 - b. tworzą listę czynności niezbędnych do wprowadzenia robota w ruch,
 - c. na podstawie listy czynności przygotowują program, zapisują go we wskazanym folderze (nauczyciel wskazuje właściwą lokalizację; może to być dysk lokalny lub zasób w chmurze) i potrafią go uruchomić;
 - d. wgrywają program do zbudowanego modelu robota, uruchamiają, obserwują i testują działanie robota. Wprowadzają modyfikacje w programie, poprawiają ewentualne błędy.

Faza III. Podsumowanie

Uczniowie analizują przedstawione **przykłady programów zapisanych w języku Python** oraz omawiają ich działanie. Nauczyciel ocenia stopień zrozumienia zależności między zapisanym kodem a sposobem poruszania się robota po uruchomieniu programu.

Uczniowie wspólnie wskazują najważniejsze elementy składni języka Python oraz omawiają **polecenia odpowiedzialne za sterowanie ruchem robota**, a także najczęściej pojawiające się błędy w zapisie kodu i sposoby ich poprawiania. Kryteria oceniania:

Ocena dopuszczająca:

- uczeń rozpoznaje podstawowe elementy programu w języku Python,
- potrafi wskazać polecenia odpowiedzialne za ruch robota.

Ocena dostateczna:

- uczeń analizuje przykładowy program i opisuje jego działanie,
- rozumie zależność między zapisanym kodem a działaniem programu i ruchem robota,
- wskazuje podstawowe błędy składniowe.



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Ocena dobra:

- uczeń samodzielnie modyfikuje przykładowy program,
- poprawia błędy związane z parametrami ruchu, skrętu lub zapisem poleceń,
- testuje działanie programu po wprowadzonych zmianach.

Ocena bardzo dobra:

- uczeń samodzielnie tworzy poprawny program sterujący ruchem robota,
- wyjaśnia działanie poszczególnych elementów programu,
- proponuje własne rozwiązania związane z przejazdem slalomem i potrafi je uzasadnić,
- samodzielnie analizuje i poprawia błędy w kodzie oraz optymalizuje ruch robota.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, np.:

1. Co robi funkcja **motor_pair.pair()**?
 - a. Uruchamia robota
 - b. Zatrzymuje silniki
 - c. Łączy dwa silniki w jedną parę**
 - d. Zmienia prędkość
2. Dlaczego w programie podaje się **port.B** przed **port.A** przy parowaniu silników?
 - a. Ze względu na konstrukcję robota i kierunek jazdy**
 - b. Ponieważ port A nie działa
 - c. Ponieważ Python tego wymaga
 - d. Nie ma to znaczenia
3. Dlaczego używamy słowa **await** przy funkcji **move_for_degrees()**?
 - a. Aby przyspieszyć program
 - b. Aby robot wykonał ruch przed przejściem do kolejnej instrukcji**
 - c. Aby zatrzymać robota
 - d. Aby wyłączyć silniki



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Faza V. Dodatkowe, praca domowa (dla chętnych).

Zadania dodatkowe (dla chętnych).

Zadanie 1: Slalom robota mobilnego

Utwórz program w języku Python, który umożliwi robotowi przejazd slalomem pomiędzy przeszkodami.

Program powinien:

- zawierać polecenia odpowiedzialne za jazdę robota do przodu i wykonywanie skrętów,
- wykorzystywać biblioteki i moduły sterujące ruchem robota,
- być zapisany poprawnie pod względem składni,
- umożliwiać robotowi przejazd pomiędzy wyznaczonymi przeszkodami,
- mieć logiczną kolejność wykonywania poleceń.

Opcjonalnie możesz rozbudować program tak, aby:

- robot wykonywał płynniejsze skręty,
- robot poruszał się z różną prędkością,
- robot zatrzymywał się po ukończeniu przejazdu,
- program zawierał dodatkowe komunikaty wyświetlane w konsoli,
- robot przejeżdżał bardziej skomplikowaną trasę slalomową.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 9 - *Odważnie naprzód! Najlepiej slalomem - zadanie dotyczące mobilności robota. Program w języku Python*

Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
 - Nauczyciel omawia podstawowe zagadnienia związane z **programowaniem mobilności robota w języku Python**:
 - wyjaśnia, że robot wykonuje ruch zgodnie z instrukcjami zapisanymi w programie,
 - przedstawia **przykłady poleceń wykorzystywanych do jazdy robota i wykonywania skrętów**,
 - omawia znaczenie poprawnego **doboru parametrów ruchu i kierunku jazdy**.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- Wprowadzenie do zadania i przypomnienie podstaw pracy w LEGO SPIKE Prime:
 - omówienie środowiska programowania Python w aplikacji LEGO SPIKE App,
 - **przedstawienie bibliotek i modułów odpowiedzialnych za sterowanie ruchem robota,**
 - przedstawienie zależności: **zapis kodu >> uruchomienie programu >> ruch robota po trasie slalomowej,**
 - analiza przykładowych programów sterujących jazdą robota i wykonywaniem skrętów.
- Praca uczniów:
 - **analiza przedstawionych przykładów programów sterujących ruchem robota,**
 - omawianie działania poszczególnych linijek kodu,
 - testowanie przejazdu robota pomiędzy przeszkodami,
 - wyszukiwanie i poprawianie błędów w kodzie oraz parametrach ruchu,
 - **proponowanie własnych sposobów przejazdu slalomem.**
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów (**analiza poprawności działania programu, sposobu poruszania się robota oraz skuteczności przejazdu slalomem**).
- Zadanie pracy domowej (opcjonalnie) – **napisanie programu umożliwiającego robotowi przejazd pomiędzy przeszkodami oraz przetestowanie różnych parametrów ruchu i skrętu.**

Załącznik nr 2 – Realizacja zadania głównego podczas lekcji nr 9 - *Odważnie naprzód! Najlepiej slalomem - zadanie dotyczące mobilności robota. Program w języku Python*

Zadanie 1. Przygotowanie do realizacji zadania głównego

Uczniowie, pracując w zespołach lub indywidualnie, realizują zadanie polegające na **zaprogramowaniu robota mobilnego do przejazdu slalomem pomiędzy przeszkodami w środowisku LEGO SPIKE Prime**. Szczególny nacisk położony jest na **dobór parametrów ruchu, poprawne wykonywanie skrętów** oraz testowanie skuteczności przejazdu robota.

Uczniowie opracowują i zapisują rozwiązanie problemu w postaci programu, który **umożliwi robotowi poruszanie się po wyznaczonej trasie slalomowej**. W tym celu:

- analizują przykładowe programy zapisane w języku Python,
- określają cel programu oraz przewidują możliwe błędy związane z ruchem robota, np. zbyt szeroki skręt, niewłaściwa prędkość lub niedokładny przejazd,
- planują kolejność wykonywania instrukcji w programie,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

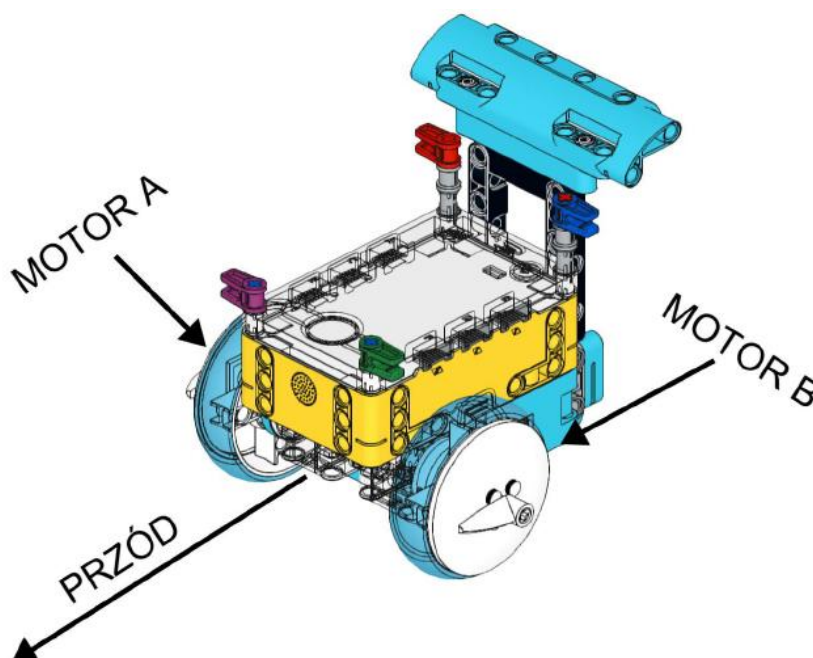
- opracowują algorytm w uporządkowanej formie,
- zapisują go jako program w języku Python sterujący ruchem robota,
- testują działanie programu i wprowadzają poprawki,
- doprecyzowują rozwiązanie tak, aby kod był czytelny, poprawny składniowo i umożliwiał płynny przejazd slalomem.

Uczniowie wykorzystują środowisko LEGO SPIKE Prime w trybie Python, zwracając uwagę na zgodność między zapisanym kodem a rzeczywistym ruchem robota po trasie slalomowej. Szczególną uwagę poświęcają poprawności składni, wykorzystaniu poleceń odpowiedzialnych za jazdę i skręcanie robota oraz analizie błędów pojawiających się podczas przejazdu.

Jeśli jest taka potrzeba, wskazana przez nauczyciela, to uczniowie zbudują robota od podstaw zgodnie z poniższą instrukcją.

Budowa bazowego robota

Uruchom plik **Budowa bazy robota.pdf** znajdujący się na platformie i postępuj wg podanych w pliku kroków. Efektem końcowym będzie fizyczny robot gotowy do programowania. **Jeśli konstrukcja jest gotowa – idź dalej.**



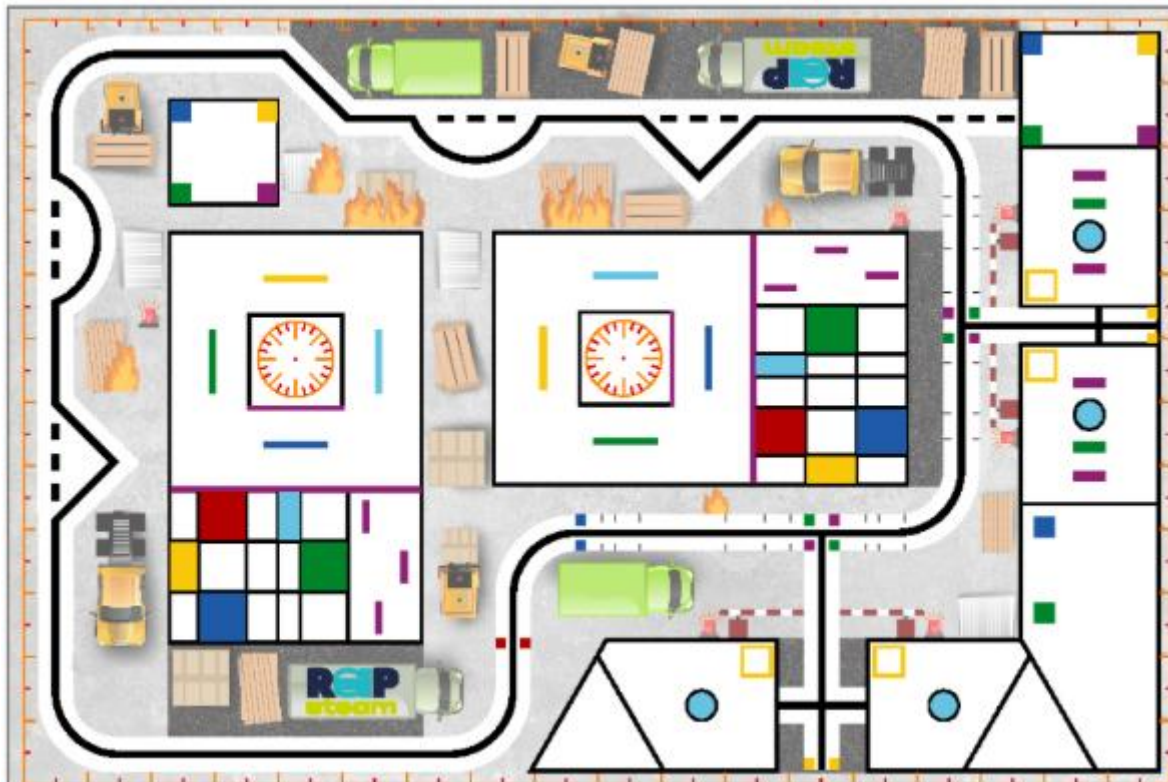
Pudełka niezbędne do realizacji lekcji: **Tylko konstrukcja bazowa**

Dodatkowe instrukcje niezbędne do realizacji lekcji:

- **Nie dotyczy**

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Do realizacji scenariusza zajęć będziesz potrzebował również matę:

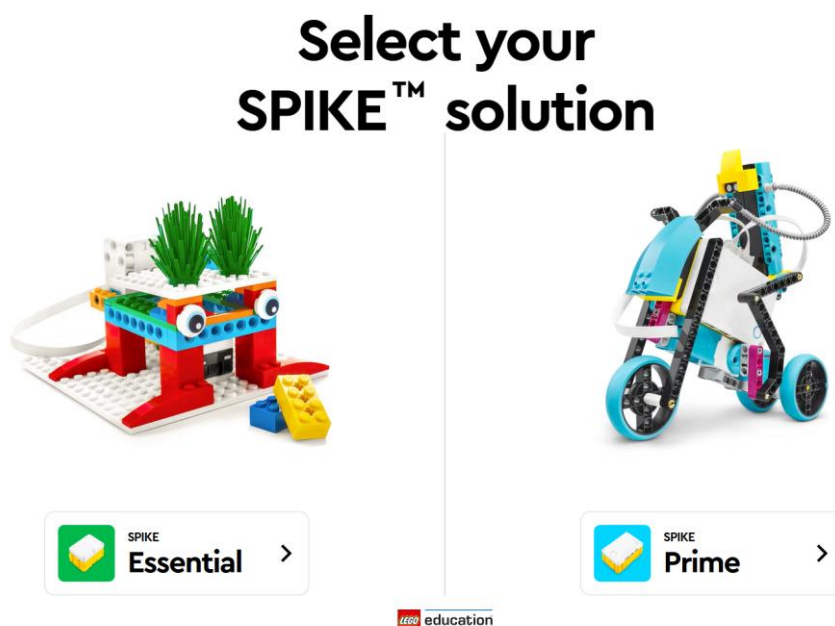


Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 2. Konfiguracja środowiska.

Uruchamianie aplikacji LEGO Spike

1. Włącz przeglądarkę internetową dostępną na twoim urządzeniu
2. Wpisz w przeglądarce adres: <https://spike.legoeducation.com/> Pojawi się okno wyboru wersji LEGO SPIKE. Wybierz Spike Prime

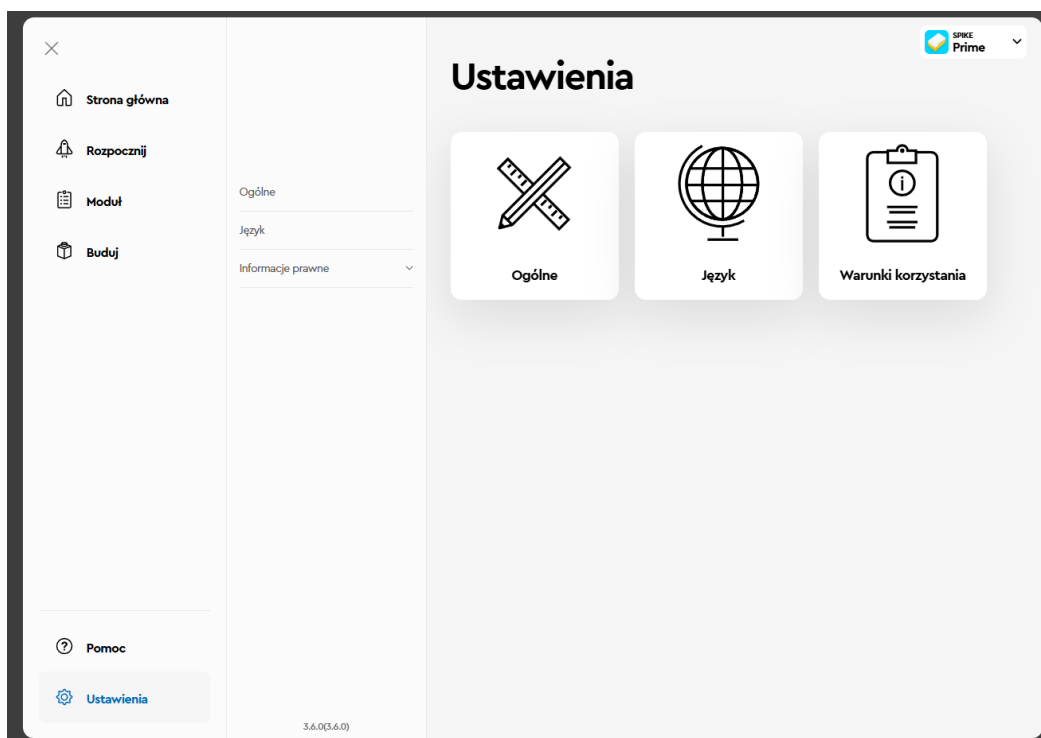


3. Po wybraniu opcji SPIKE Prime pojawia się menu główne. Jeśli występuje język polski, należy zmienić go na język angielski.

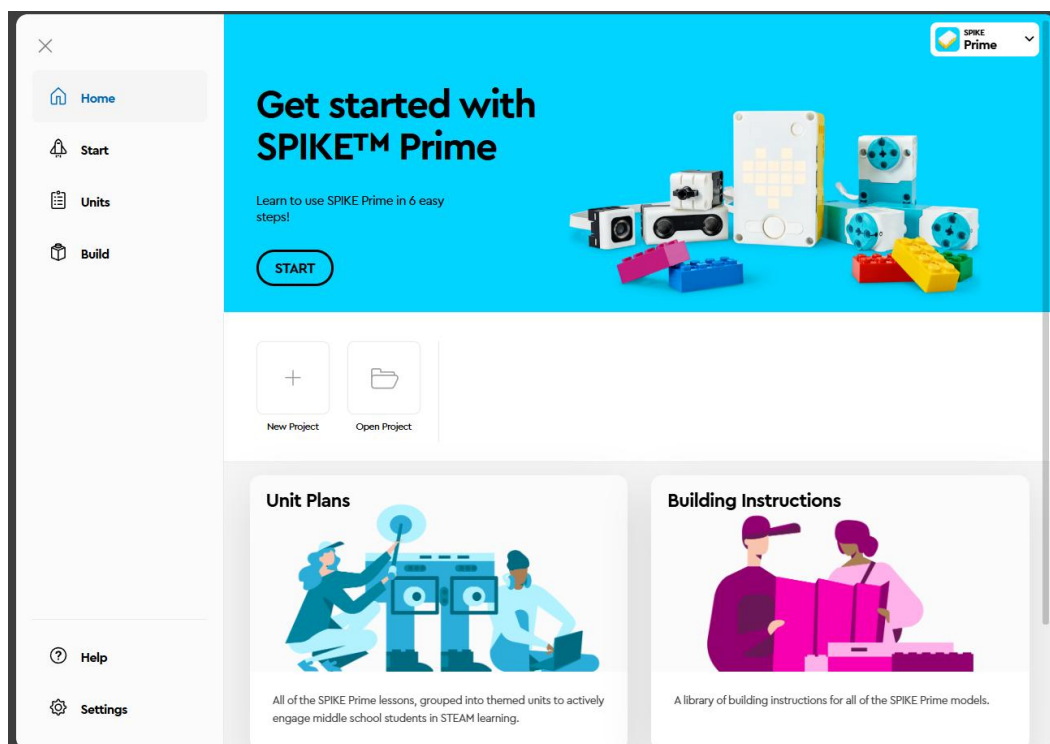
Jeżeli zaś wszystko jest w **języku angielskim**, to przejdź do kroku 6.

Aby zmienić język, w lewym dolnym rogu menu głównego kliknij przycisk **Ustawienia**.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

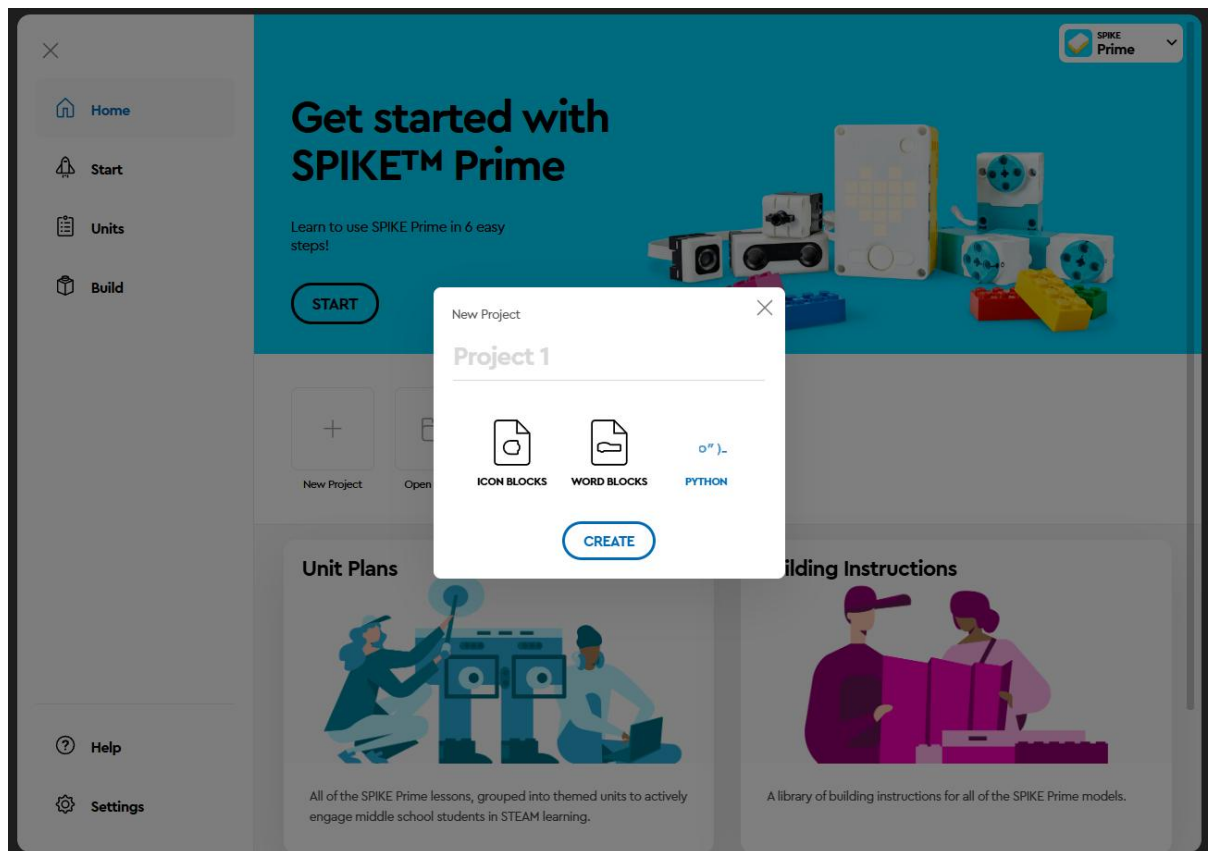


4. W nowym oknie kliknij globus podpisany „Language”.
5. Wybierz język: **United States (English)**. Strona przeładuje się i język angielski będzie dostępny.
6. Kliknij po lewej przycisk **Home**. Pojawi się okno jak poniżej

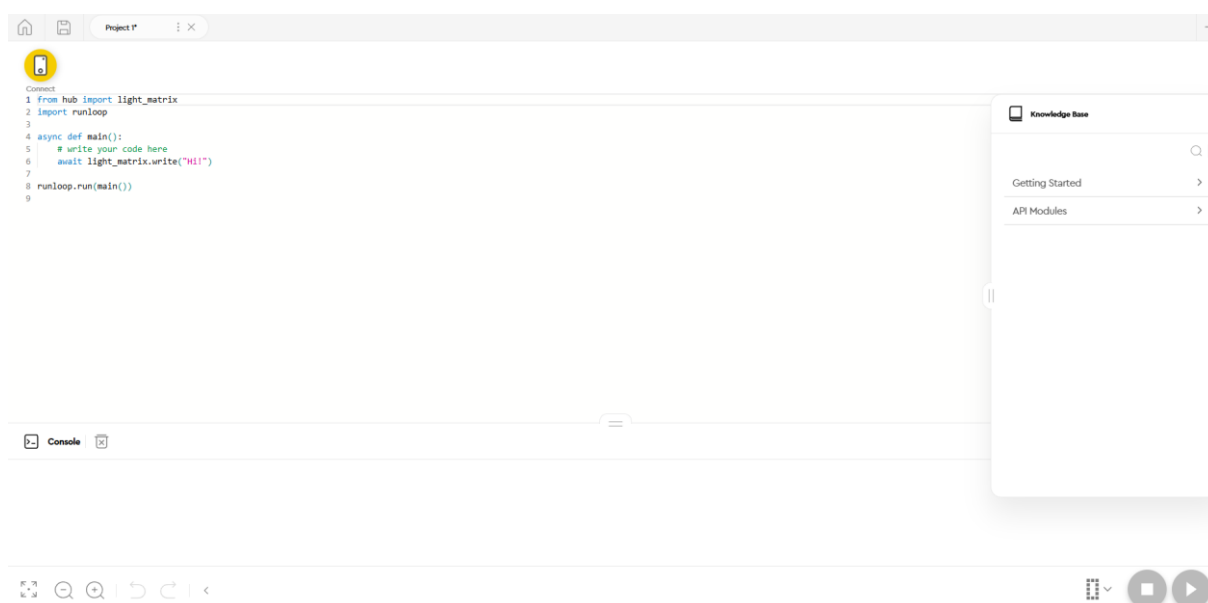


Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

7. Kliknij na środku przycisk **New Project**. Pojawi się nowe okno jak poniżej



8. Wybierz opcję **Python** oraz podaj nazwę projektu. Kiedy wszystko zostanie poprawnie wybrane, wciśnij przycisk **Create**.

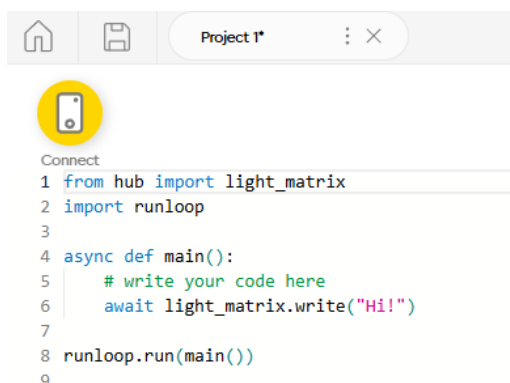


Wszystko jest gotowe do programowania. Pora na podłączenie kontrolera Spike Hub.

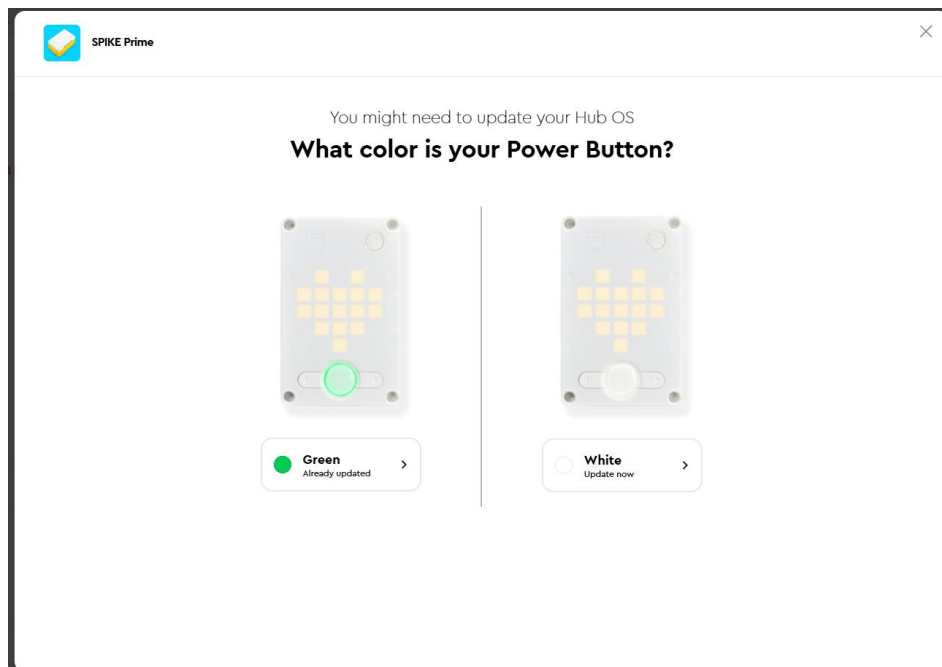
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 3. Podłączenie kontrolera Spike Hub

1. W głównym programie kliknij przycisk **Connect** znajdujący się w górnym rogu, tuż pod nazwą twojego projektu. Patrz poniżej.

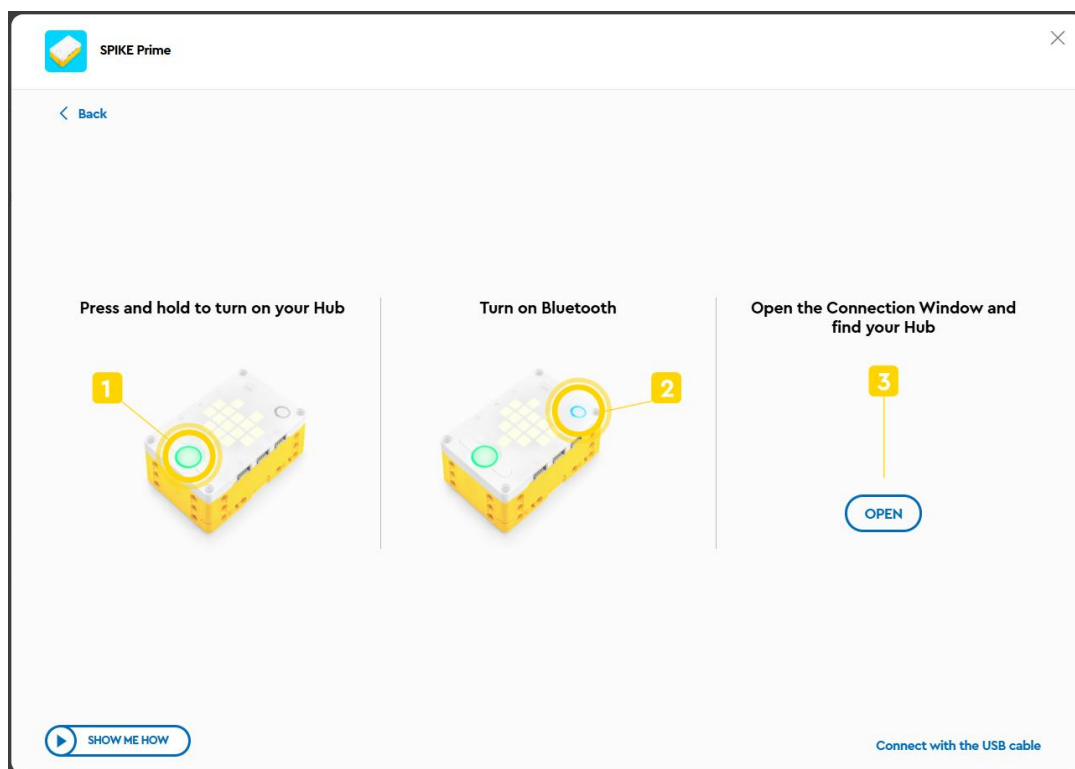


2. Pojawi się nowe okno. **Włącz kontroler Spike Hub** klikając **duży biały przycisk** i poczekaj aż się uruchomi. Następnie wybierz jedną z opcji odpowiadającą stanowi kontrolera Spike Hub – zawsze będzie to **Green – Already updated**.

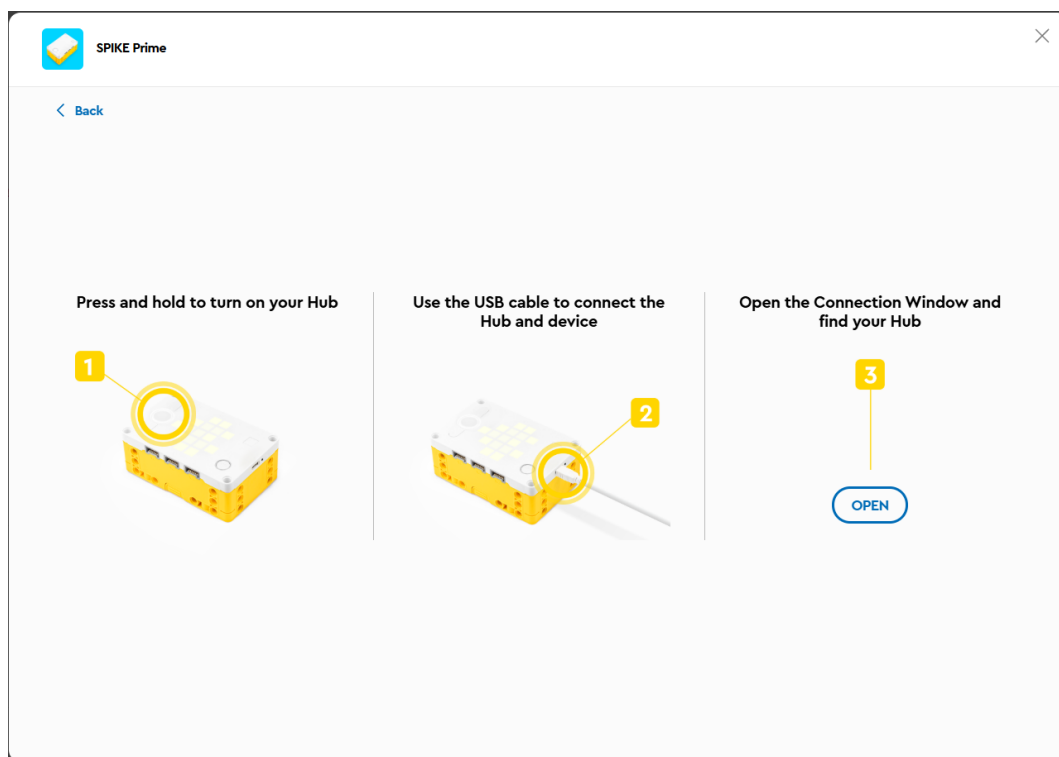


3. Pojawi się nowe okno (patrz poniżej). Wybierz opcję **Connect with USB cable** znajdującą się w prawym dolnym rogu okna.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

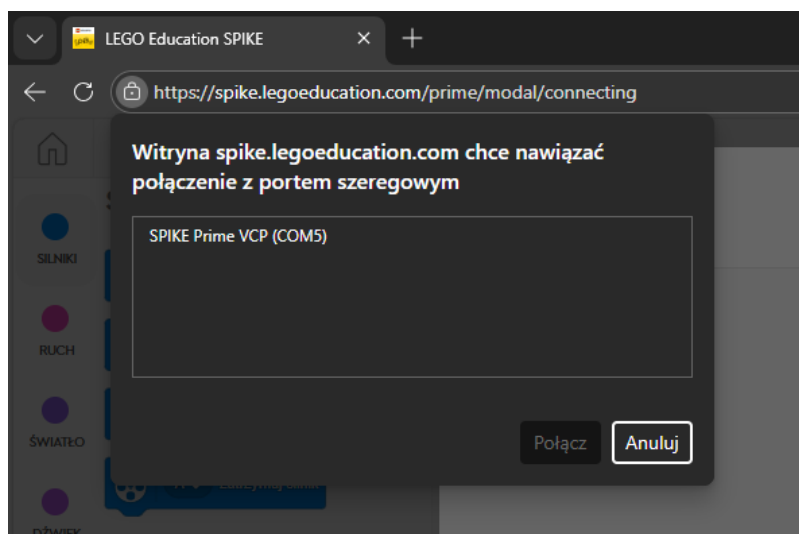


4. Postępuj zgodnie z instrukcją na ekranie. Po przeczytaniu kliknij przycisk **Open**.



5. Pojawi się nowe okno (patrz poniżej)

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



6. Wybierz opcję **SPIKE Prime VCP** i kliknij **Połącz**.

UWAGA – u Ciebie nazwa może nie być identyczna jak na rysunku powyżej. Wybierz tę opcję, która ma w nazwie Spike Prime. Innymi rzeczami się nie przejmuj.

Jeśli wszystko zostanie poprawnie połączone, to okna zostaną automatycznie zamknięte, a ikona kontrolera w programie zmieni się na zieloną.



```
1 from hub import light_matrix
2 import runloop
3
4 async def main():
5     # write your code here
6     await light_matrix.write("Hi!")
7
8 runloop.run(main())
9
```

Gratulacje. Twój Spike Hub jest podłączony do komputera. Możesz przystąpić do programowania robota.

W przypadku problemów postępuj wg kroków:

1. Sprawdź, czy kontroler Spike Hub jest włączony i świeci na zielono
 - a. Jeśli kontroler świeci na biało zgłoś to nauczycielowi
 - b. Jeśli kontroler świeci na zielono to podążaj dalej wg instrukcji łączenia powyżej
2. Sprawdź czy kabel USB jest podłączony (Spike Hub <-> Komputer)
3. Sprawdź, czy wybrano odpowiednią nazwę urządzenia – w oknie połączenia z portem szeregowym należy wybrać urządzenie, które w nazwie ma SPIKE Prime.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 4. Realizacja zadania głównego

W tej lekcji nauczysz się obsługi silników podpiętych do kontrolera Spike Hub. Poznasz różne funkcje oraz możliwości ruchu. W dalszej części wprawisz robota w ruch.



Dzisiejsze zajęcia wprowadzają do programowania silników w Pythonie dla LEGO Spike. Celem lekcji jest poznanie programu oraz nauczanie się obsługi silników, jazdy do przodu i do tyłu, skręcania oraz zmiany parametrów prędkości obrotów silników.

Przykład realizacji zadania – wprowadzenie bibliotek

UWAGA – w materiałach stosowane są komentarze. Na zrzutach ekranu jest to tekst oznaczony jako zielony. Tego tekstu nie przepisuj! To jest tylko wytłumaczenie dla Ciebie, co dzieje się w danej części programu. Przepisuj tylko i wyłącznie tekst, który nie jest zielony. Korzystaj z tabeli. Pamiętaj, że wielkość liter MA ZNACZENIE!

Na początku programu należy dodać odpowiednie odwołania (biblioteki) do modułów kontrolera Spike Hub i czujników. Dodaj moduł, biblioteki:

- port (moduł),
- light_matrix (moduł),
- motor_pair (biblioteka),
- time (biblioteka),

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Aby móc programować silniki niezbędne jest zaimportowanie biblioteki **motor_pair**. Pozwala ona, tak jak w Scratchu za pomocą różowych bloków sterować dwoma silnikami jednocześnie. Istnieje biblioteka **motor**, która odpowiada niebieskim blozkom w Scratchu (sterowanie pojedynczym silnikiem). Z tą biblioteką zapoznasz się na dalszych lekcjach. Podczas dzisiejszych zajęć wykorzystasz też moduł **port** oraz **light_matrix** (dostęp do portów oraz wyświetlanie informacji na matrycy Spike Hub), a biblioteka **time** pozwoli na stosowanie wstrzymania działania programu na określony czas.

Usuń początek programu i dodaj odpowiednie odwołania. Część programu odpowiedzialna za biblioteki powinna wyglądać następująco:

Zdjęcie	Kod (tekst)
<pre> 1 from hub import port 2 from hub import light_matrix 3 4 import motor_pair 5 import time 6 7 import runloop </pre>	<pre> from hub import port from hub import light_matrix import motor_pair import time import runloop </pre>

W tej chwili cały program powinien wyglądać następująco:

```

1 from hub import port
2 from hub import light_matrix
3 import motor_pair
4 import time
5 import runloop
6
7 async def main():
8     # write your code here
9     await light_matrix.write("Hi!")
10
11 runloop.run(main())

```

Usuń kod odpowiedzialny za wyświetlanie tekstu Hi! Nie przyda się dzisiaj. Usuń też tekst #write your code here

Ruch robotem – przód/tył

Aby móc wykonać ruch, tak samo jak w Scratch silniki potrzebują konkretnych informacji, które należy podać za pomocą kodu w języku Python. Najważniejsze informacje ograniczają

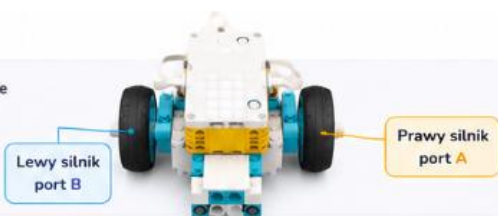
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

się do podania portów, na których wpięte są silniki. W bazowej konstrukcji przeznaczonej do **projektu RaP Steam** silniki podpięte są odpowiednio do portów A i B. W Scratch niezbędne było podanie bloków, które „ustawiały” silniki. Tutaj jest podobnie, choć za pomocą kodu robi się to przejrzysiej i szybciej.

Ruch robotem – przód/tył

Aby móc wykonać ruch, tak samo jak w Scratch silniki potrzebują konkretnych informacji, które należy podać za pomocą kodu w języku Python. Najważniejsze informacje ograniczają się do podania portów, na których wpięte są silniki.

W bazowej konstrukcji robota RaP Steam silniki podpięte są odpowiednio do portów **A i B**.



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Pierwszym etapem jest wywołanie funkcji **pair()**, która przyjmuje 3 obowiązkowe parametry (wartości w nawiasach):

W Pythonie podobnie jak w Scratch musimy „ustawić” silniki.
Pierwszym etapem jest wywołanie funkcji **pair**, która przyjmuje 3 obowiązkowe parametry:

```
motor_pair.pair(nazwa_pary, port_lewy, port_prawy)
```

- 1 **nazwa_pary** – pozwala na odwołanie się w dalszym programie do pary silników za pomocą skrótu
- 2 **port_lewy** – wartość z modułu port
- 3 **port_prawy** – wartość z modułu port

Pełna komenda wygląda następująco:

```
motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
```

1

2

3

motor_pair.PAIR_1 przypisuje do PAIR_1 silniki B (lewy) i A (prawy). Od tej pory możesz używać tej nazwy zamiast za każdym razem pisać od nowa całość.



UWAGA! Jeśli wpiszesz taki kod:

```
motor_pair.pair(motor_pair.PAIR_1, port.A, port.B)
```

to robot przy wartościach dodatnich będzie jechał do tyłu!

Jest to związane z konstrukcją i ułożeniem silników.

Aby ich nie przepinać (co skutkuje zmianą rozłożenia masy robota i wagą) podaje się wartości na odwrót.

Silnik lewy w konstrukcji robota wpięty jest do portu B dlatego podawany jest jako pierwszy.

Silnik prawy w konstrukcji robota wpięty jest do portu A dlatego podawany jest jako drugi.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

UWAGA!!! Jeśli wpiszesz taki kod: `motor_pair.pair(motor_pair.PAIR_1, port.A, port.B)`, to robot przy wartościach dodatnich będzie jechał do tyłu! Jest to związane z konstrukcją i ułożeniem silników. Aby ich nie przepinać (co skutkuje zmianą rozłożenia masy robota i wagą) podaje się wartości na odwrót.

Po zdefiniowaniu nazw wystarczy robotowi powiedzieć, aby jechał do przodu. Można to zrobić poprzez zadany dystans lub po prostu funkcją `move()`. Funkcja `move()` nie zatrzymuje robota. On po prostu jedzie. Dlatego ważne jest, aby zatrzymać robota. Służy do tego funkcja `stop()`. Oczywiście nie możemy zatrzymać robota od razu, ponieważ nigdy by nie ruszył. Nie możemy też użyć słowa `await` przy funkcji `move()`.

Dlaczego?

Skoro nie mówimy mu, kiedy skończyć (`move()` mówi jasno – jedź cały czas) to, dokąd `await` ma działać? Należy tu wykorzystać funkcje z biblioteki `time`. Użyjemy funkcji `sleep_ms()`, którą już znasz. **Pamiętaj: 1000 ms (milisekund) to 1 s (sekunda).**



Nie możemy użyć `await` przy funkcji `move`, ponieważ `move` mówi jasno – jedź cały czas. Nie wiemy, dokąd `await` ma działać.
Użyjemy funkcji z biblioteki `time` – `sleep_ms()`.
Pamiętaj: 1000 ms (milisekund) to 1 s (sekunda).

No to do dzieła. Program wygląda następująco:

```
1 from hub import port
2 from hub import light_matrix
3 import motor_pair
4 import time
5 import runloop
6
7 async def main():
8     # przypisanie portów A i B dla silników do wartości PAIR_1
9     motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
10
11     # jazda robota prosto do przodu z z prędkością 360 stopni na sekundę. PAIR_1 - porty A i B, 0 - wartość skrętu.
12     # Dla 0 robot jedzie prosto. Prędkość: 360 - ile stopni obróci się silnik w ciągu 1 sekundy
13     motor_pair.move(motor_pair.PAIR_1, 0, velocity=360)
14     # oczekiwanie 2 sekundy. W tym czasie robot jedzie komendą wyżej
15     time.sleep_ms(2000)
16
17     # zatrzymanie silników robota po 2 sekundach działania
18     motor_pair.stop(motor_pair.PAIR_1)
19
20 runloop.run(main())
```

Uruchom i zobacz co się stanie. Robot po 2 sekundach powinien się zatrzymać!

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Co, jeśli chcemy jechać przez określony dystans?

Nie ma problemu. Do tego służy funkcja `move_for_degrees`. Potrzebuje ona parę parametrów (w nawiasach):

```
motor_pair.move_for_degrees(motor_pair.PAIR_1, 720, 0)
```

Ta funkcja mówi robotowi: dla pary nr 1 (PAIR_1) uruchom silnik i niech się kręci przez 720 stopni, ale bez skrętu (0). O co chodzi ze skrętem? Możesz to określić jako skręt kierownicy w samochodzie. Jak rodzice jadą autem prosto, to kierownica też jest prosto – ma wartość 0. Tak samo jest w LEGO. Jeśli nie chcesz, aby robot skręcał, to podaj mu wartość 0.

W przypadku programowania silników **BARDZO WAŻNE** jest używanie słowa `await`. Pozwoli to robotowi na wykonanie ruchu, nim przejdzie do następnego! Stosuj `await` zawsze, jeśli określasz konkretny ruch (np. podając stopnie obrotu wału silnika).

Zrób następujący przykład. Robot jedzie prosto przez 2 sekundy. Następnie zatrzymuje się, czeka 1 sekundę, a następnie jedzie do przodu przez 720 stopni. No to do dzieła. Ze względu na dużą objętość kodu na rysunku widać tylko kod w funkcji `main()`. **Nie usuwaj odwołań do bibliotek. One się nie zmieniają, dlatego na rysunku są pomijane.**

```
1 from hub import port
2 from hub import light_matrix
3 import motor_pair
4 import time
5 import runloop
6
7 async def main():
8     # przypisanie portów A i B dla silników do wartości PAIR_1
9     motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
10
11     #jazda robota prosto do przodu z z prędkością 360 stopni na sekundę. PAIR_1 - porty A i B, 0 - wartość skrętu.
12     #Dla 0 robot jedzie prosto. Prędkość: 360 - ile stopni obróci się silnik w ciągu 1 sekundy
13     motor_pair.move(motor_pair.PAIR_1, 0, velocity=360)
14     #czekanie 2 sekundy. W tym czasie robot jedzie komendą wyżej
15     time.sleep_ms(2000)
16
17     #zatrzymanie silników robota po 2 sekundach działania
18     motor_pair.stop(motor_pair.PAIR_1)
19
20     #czekanie sekundę aby zobaczyć że robot się zatrzymał
21     time.sleep_ms(1000)
22
23     #jazda prosto o 720 stopni (2 * 360 stopni)
24     motor_pair.move_for_degrees(motor_pair.PAIR_1, 720, 0)
25
26 runloop.run(main())
```

Uruchom program i zobacz, jak się zachowuje. Robot powinien pojechać do przodu.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

A teraz do tyłu

Super, teraz pora na cofanie.

JAZDA PRZEZ OKREŚLONY DYSTANS

Do jazdy przez określony dystans służy funkcja `move_for_degrees`.

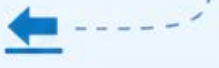
```
motor_pair.move_for_degrees(PAIR_1, stopnie, skręt, velocity=...)
```

- PAIR_1** – numer pary silników
- stopnie** – liczba stopni obrotu wałów silników (dystans)
- skręt** – kąt skrętu (0 = jazda prosto)
- velocity** – prędkość (stopnie obrotu w 1 sekundzie)

Przykład 2: jazda do tyłu

```
await motor_pair.move_for_degrees  
(motor_pair.PAIR_1, -720, 0)
```

Ujemne stopnie = robot cofa się.



Przykład 3: zmiana prędkości

```
await motor_pair.move_for_degrees  
(motor_pair.PAIR_1, 720, 0,  
velocity=720)
```

`velocity=360` – 1 obrót/sekundę
`velocity=720` – 2 obroty/sekundę
itd.



Wykorzystaj tę samą funkcję, ale z lekką modyfikacją. Jako stopnie jazdy podaj mu **wartość ujemną**. Wtedy robot będzie cofał.

```
7 async def main():
8     # przypisanie portów A i B dla silników do wartości PAIR_1
9     motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
10
11     #jazda robota prosto do przodu z z prędkością 360 stopni na sekundę. PAIR_1 - porty A i B, 0 - wartość skrętu.
12     #Dla 0 robot jedzie prosto. Prędkość: 360 - ile stopni obróci się silnik w ciągu 1 sekundy
13     motor_pair.move(motor_pair.PAIR_1, 0, velocity=360)
14     #czekanie 2 sekundy. W tym czasie robot jedzie komendą wyżej
15     time.sleep_ms(2000)
16
17     #zatrzymanie silników robota po 2 sekundach działania
18     motor_pair.stop(motor_pair.PAIR_1)
19
20     #czekanie sekundę aby zobaczyć że robot się zatrzymał
21     time.sleep_ms(1000)
22
23     #jazda prosto o 720 stopni (2 * 360 stopni)
24     motor_pair.move_for_degrees(motor_pair.PAIR_1, 720, 0)
25
26     #jazda do tyłu o -720 stopni (2 * 360 stopni) w jednej komendzie z prędkością 360 stopni na sekundę
27     motor_pair.move_for_degrees(motor_pair.PAIR_1, -720, 0, velocity=360)
28
29     runloop.run(main())
```

Jak widzisz, pojawił się nowy parametr: **velocity=360**. Oznacza on prędkość, z jaką obracają się silniki. W prostych słowach

Robot w ciągu jednej sekundy wykona 360 stopni, czyli 1 obrót

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

A co, jeśli podamy mu 720? Wtedy będzie się poruszał 2 razy szybciej – 2 obroty w ciągu jednej sekundy.

Pora na testy.

Wpisz ten parametr w linii 24 (patrz rysunek poniżej) i zobacz, co się wydarzy. Zastanów się, dlaczego.

```

7 async def main():
8     # przypisanie portów A i B dla silników do wartości PAIR_1
9     motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
10
11     # jazda robota prosto do przodu z prędkością 360 stopni na sekundę. PAIR_1 - porty A i B, 0 - wartość skreću.
12     # Dla 0 robot jedzie prosto. Prędkość: 360 - ile stopni obróci się silnik w ciągu 1 sekundy
13     motor_pair.move(motor_pair.PAIR_1, 0, velocity=360)
14     # czekanie 2 sekundy. W tym czasie robot jedzie komendą wyżej
15     time.sleep_ms(2000)
16
17     # zatrzymanie silników robota po 2 sekundach działania
18     motor_pair.stop(motor_pair.PAIR_1)
19
20     # czekanie sekundę aby zobaczyć że robot się zatrzymał
21     time.sleep_ms(1000)
22
23     # jazda prosto o 720 stopni (2 * 360 stopni) w jednej komendzie z prędkością 720 stopni na sekundę
24     motor_pair.move_for_degrees(motor_pair.PAIR_1, 720, 0, velocity=720)
25
26     # jazda do tyłu o -720 stopni (2 * 360 stopni) w jednej komendzie z prędkością 360 stopni na sekundę
27     motor_pair.move_for_degrees(motor_pair.PAIR_1, -720, 0, velocity=360)

```

Skrećanie robotem – ruch w lewo i w prawo

Skoro umiesz już poruszać się do przodu i do tyłu, to pora na skrećanie.

Jak wykonuje się skreć?

Bardzo prosto, wykorzystując tę samą funkcję. Jedyna różnica to zmiana wartości 0 na inną. W przypadku podstawowej konstrukcji przy wartościach dodatnich robot będzie skrećał w prawo, zaś jeśli podasz mu wartość ujemną, to zacznie skrećać w lewo. Przejdź od razu do testów.

Usuń kod odpowiedzialny za jazdę przód/tył i dodaj nowy:

```

1 from hub import port
2 from hub import light_matrix
3 import motor_pair
4 import time
5 import runloop
6
7 async def main():
8     # przypisanie portów A i B dla silników do wartości PAIR_1
9     motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
10
11     # skrećanie w prawo przez 130 stopni z pełnym skrećem 100 ze standardową prędkością obrotu (wartość domyślna - nie jest podana)
12     motor_pair.move_for_degrees(motor_pair.PAIR_1, 130, 100)
13
14 runloop.run(main())

```

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Robot, jak widzisz, porusza się w prawo po uruchomieniu programu. Teraz twoim zadaniem jest skręt w lewo. Ale uwaga. Zrób tak, aby robot najpierw skręcił w prawo (tak jak teraz), a następnie w lewo.

Jak to zrobić? Warto użyć jednego kluczowego słowa `await`.

Rozwiązanie

```
1 from hub import port
2 from hub import light_matrix
3 import motor_pair
4 import time
5 import runloop
6
7 async def main():
8     # przypisanie portów A i B dla silników do wartości PAIR_1
9     motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
10
11     #skręcanie w prawo przez 130 stopni z pełnym skretem 100 ze standardową prędkością obrotu (wartość domyślna - nie jest podana)
12     motor_pair.move_for_degrees(motor_pair.PAIR_1, 130, 100)
13
14     #skręcanie w lewo przez 130 stopni z pełnym skretem -100 ze standardową prędkością obrotu (wartość domyślna - nie jest podana)
15     motor_pair.move_for_degrees(motor_pair.PAIR_1, 130, -100)
16
17 runloop.run(main())
```

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przykładowa realizacja zadania głównego

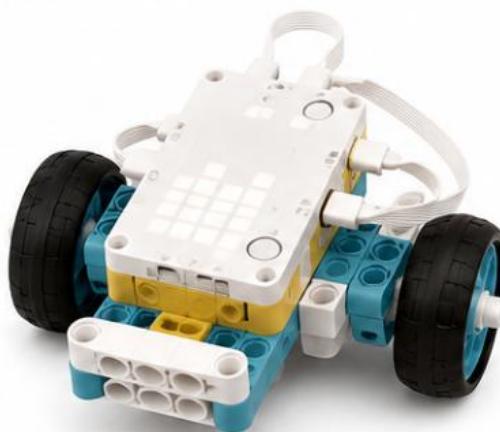
Wspaniale. Skoro umiesz już poruszać się robotem, to pora zająć się zadaniem głównym. Wyglądało ono następująco. Ostatecznym zadaniem będzie:

PRZYKŁADOWA REALIZACJA ZADANIA GŁÓWNEGO

Skoro umiesz już poruszać robotem, to pora zająć się zadaniem głównym.

Ostatecznym zadaniem będzie:

- 1 Robot jedzie do przodu przez 10 centymetrów
- 2 Robot skręca w prawo
- 3 Robot jedzie do tyłu przez 5 centymetrów
- 4 Robot skręca w lewo
- 5 Robot jedzie do przodu przez 7 centymetrów
- 6 Robot jedzie do przodu przez dodatkowe 20 centymetrów



No to do dzieła.

Usuń kod odpowiedzialny za skręcanie. Funkcja `main()` powinna być pusta

Przejdź robotem dystans 10 centymetrów. Aby to zrobić, należy podać mu liczbę stopni, jaką ma wykonać. Skorzystaj z wzoru:

Odległość w centymetrach * 20,5 = liczba stopni

Wynik mnożenia to liczba w stopniach, jaką należy wpisać. Na tym etapie możesz używać liczb całkowitych (pomiń to, co po kropce, i jej nie stosuj). Jak można policzyć, aby przebyć dystans 10 centymetrów należy podstawić:

$$10 \text{ centymetrów} * 20,5 = 205 \text{ stopni}$$

Drugi parametr, który jest wymagany, to siła skrętu.

Zadanie wykonuje komenda:

```
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 205, 0)
```



PAMIĘTAJ:

Zawsze używaj słowa `await` przy funkcjach określających konkretny ruch (np. `move_for_degrees`), dzięki temu robot wykona ruch zanim przejdzie do następnej instrukcji.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Kod odpowiadający za jazdę do przodu przez 10 centymetrów bez skręcania wygląda następująco:

```
1 from hub import port
2 from hub import light_matrix
3 import motor_pair
4 import time
5 import runloop
6
7 async def main():
8     # przypisanie portów A i B dla silników do wartości PAIR_1
9     motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
10
11     #robot jedzie do przodu na dystans 10 centymetrów (10 * 20,5 = 205). Skręt 0 (nie skręca).
12     #Słowo await aby robot wykonał ruch i dopiero przeszedł dalej
13     await motor_pair.move_for_degrees(motor_pair.PAIR_1, 205, 0)
14
15 runloop.run(main())
```

Uruchom program i zobacz, co się stanie. Twój robot powinien przejechać w przód 10 centymetrów. Po przejeździe zatrzymaj program. Dobra robota, teraz pora na skręcanie.

Jak już wiesz, skręcanie wykonywane jest tą samą funkcją. W naszych ćwiczeniach stosować będziemy wartości **100** i **-100** w zależności od tego, czy chcemy skręcać w prawo, czy w lewo. W tej lekcji masz pełną dowolność. Możesz używać innych wartości. Dodajmy zatem dalszą część kodu. Pewnie zastanawiasz się, dlaczego importujemy moduł **light_matrix**. Mamy zamiar go wykorzystać. Na wyświetlaczu **Spike Hub** dodamy strzałkę. Kiedy dojdziemy do samego końca głównego zadania, podmienimy ją na uśmiechniętą buźkę, co oznaczać będzie, że robot wykonał swoje zadanie. Na razie dodajmy strzałkę i skręt.

Kod wygląda następująco:

```
1 from hub import port
2 from hub import light_matrix
3 import motor_pair
4 import time
5 import runloop
6
7 async def main():
8     # przypisanie portów A i B dla silników do wartości PAIR_1
9     motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
10
11     light_matrix.show_image(light_matrix.IMAGE_ARROW_S)
12
13     #jazda wyznaczoną trasą. Wzór na stopnie: droga w centymetrach * 20,5.
14     #Prędkość domyślna - 360 stopni na sekundę. Wartość velocity nie będzie zmieniana
15
16     #robot jedzie do przodu na dystans 10 centymetrów (10 * 20,5 = 205). Skręt 0 (nie skręca).
17     #Słowo await aby robot wykonał ruch i dopiero przeszedł dalej
18     await motor_pair.move_for_degrees(motor_pair.PAIR_1, 205, 0)
19
20     #skręcanie w prawo przez 130 stopni z pełnym skrętem 100
21     await motor_pair.move_for_degrees(motor_pair.PAIR_1, 130, 100)
22
23 runloop.run(main())
```

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Uruchom program i sprawdź działanie programu. Pamiętaj, że po uruchomieniu robot najpierw pojedzie prosto!

Super, teraz pora na ciąg dalszy. Zobaczmy co się udało i co teraz należy zrobić:

- ~~Robot jedzie do przodu przez 10 centymetrów~~
- ~~Robot skręca w prawo~~
- Robot jedzie do tyłu przez 5 centymetrów
- Robot skręca w lewo
- Robot jedzie do przodu przez 7 centymetrów
- Robot jedzie do przodu przez dodatkowe 20 centymetrów

Pora na jazdę do tyłu przez 5 centymetrów. Już wiesz, że **aby robot jechał do tyłu**, należy podać mu ujemną liczbę stopni. Dla 5 centymetrów jest to **-125** stopni.

No to jazda:

```

1 from hub import port
2 from hub import light_matrix
3 import motor_pair
4 import time
5 import runloop
6
7 async def main():
8     # przypisanie portów A i B dla silników do wartości PAIR_1
9     motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
10
11     light_matrix.show_image(light_matrix.IMAGE_ARROW_S)
12
13     #jazda wyznaczoną trasą. Wzór na stopnie: droga w centymetrach * 20,5.
14     #Prędkość domyślna - 360 stopni na sekundę. Wartość velocity nie będzie zmieniana
15
16     #robot jedzie do przodu na dystans 10 centymetrów (10 * 20,5 = 205). Skręt 0 (nie skręca).
17     #Słowo await aby robot wykonał ruch i dopiero przeszedł dalej
18     await motor_pair.move_for_degrees(motor_pair.PAIR_1, 205, 0)
19
20     #skręcanie w prawo przez 130 stopni z pełnym skrętem 100
21     await motor_pair.move_for_degrees(motor_pair.PAIR_1, 130, 100)
22
23     #robot jedzie do tyłu na dystans -5 centymetrów (-5 * 20,5 = -125). Skręt 0 (nie skręca).
24     #Słowo await aby robot wykonał ruch i dopiero przeszedł dalej
25     await motor_pair.move_for_degrees(motor_pair.PAIR_1, -125, 0)
26
27 runloop.run(main())

```

Skoro mamy jazdę do tyłu, to przejdźmy dalej.

Pora na skręt w lewo.

Zasada działania jest taka sama. Skoro w prawo jest wartością dodatnią, to w lewo będzie ujemną. Dodaj kolejną linię kodu:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

```
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 130, -100)
```

Gdy robot skręca, to zostały dwa punkty do zrealizowania. Jazda do przodu przez 7 centymetrów i 20. Wiesz, jak to zrobić? Spróbuj samodzielnie.

Jeśli chcesz, możesz wesprzeć się ostatecznym programem:

```
1 from hub import port
2 from hub import light_matrix
3 import motor_pair
4 import time
5 import runloop
6
7 async def main():
8     # przypisanie portów A i B dla silników do wartości PAIR_1
9     motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
10
11     light_matrix.show_image(light_matrix.IMAGE_ARROW_S)
12
13     #jazda wyznaczoną trasą. Wzór na stopnie: droga w centymetrach * 20,5.
14     #Prędkość domyślna - 360 stopni na sekundę. Wartość velocity nie będzie zmieniana
15
16     #robot jedzie do przodu na dystans 10 centymetrów (10 * 20,5 = 205). Skręt 0 (nie skręca).
17     #Słowo await aby robot wykonał ruch i dopiero przeszedł dalej
18     await motor_pair.move_for_degrees(motor_pair.PAIR_1, 205, 0)
19
20     #skręcanie w prawo przez 130 stopni z pełnym skrętem 100
21     await motor_pair.move_for_degrees(motor_pair.PAIR_1, 130, 100)
22
23     #robot jedzie do tyłu na dystans -5 centymetrów (-5 * 20,5 = -125). Skręt 0 (nie skręca).
24     #Słowo await aby robot wykonał ruch i dopiero przeszedł dalej
25     await motor_pair.move_for_degrees(motor_pair.PAIR_1, -125, 0)
26
27     #skręt w lewo przez 130 stopni z pełnym skrętem 100
28     await motor_pair.move_for_degrees(motor_pair.PAIR_1, 130, -100)
29
30     #robot jedzie do przodu na dystans 7 centymetrów (7 * 20,5 = 143,5). Funkcja przyjmuje tylko wartości CAŁKOWITE.
31     #Skręt 0 (nie skręca). Słowo await aby robot wykonał ruch i dopiero przeszedł dalej
32     await motor_pair.move_for_degrees(motor_pair.PAIR_1, 143, 0)
33
34     #robot jedzie do przodu na dystans 20 centymetrów (20 * 20,5 = 410).
35     #Skręt 0 (nie skręca). Słowo await aby robot wykonał ruch i dopiero przeszedł dalej
36     await motor_pair.move_for_degrees(motor_pair.PAIR_1, 410, 0)
37
38     light_matrix.show_image(light_matrix.IMAGE_HAPPY)
39
40 runloop.run(main())
```

Na koniec wyświetlamy uśmiechniętą buźkę mówiącą nam, że robot zakończył działanie. Wspaniała robota. Wiesz już doskonale, jak działają silniki w LEGO Spike.

Eksperymentuj z wartościami. Zobacz, jak robot zachowuje się, jeśli podasz mu inne liczby. Powodzenia i do następnej lekcji!

Możliwe rozszerzenia zadania

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

1. Wykonaj obrót o 180 stopni w miejscu
2. Wykonaj obrót po łuku tak aby robot obrócił się o 180 stopni
3. Wykonaj cofanie po łuku w lewo a następnie skręć w prawo do przodu po łuku. Robot po drodze musi się zatrzymać. Zalecane są obroty o ok. 90 stopni

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 3: Globalna wspólnota Cyfrowego Świata

Temat lekcji nr 10 – Budowa sieci komputerowej.

Treści nauczania (wymagania szczegółowe - podstawa programowa) – III. 1);

Cel lekcji:

- Uczniowie zapoznają się z budową sieci komputerowej.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- Uczeń przedstawia funkcjonowanie sieci komputerowej i sieci Internet;

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści programowe będą realizowane za pomocą uniwersalnych narzędzi dostępnych w systemach operacyjnych, przede wszystkim menedżerach okien. W miarę możliwości, spowodowanej dostępnością usługi zapisu danych w chmurze, uczniowie będą korzystać z zapisu danych w środowiskach wirtualnych.
- Realizacja za pomocą uniwersalnych narzędzi, w które wyposażone są wszystkie wykorzystywane w szkołach systemy operacyjne (np. eksploratory plików).

Uczeń:

- poznaje strukturę sieci komputerowej, obejmującej system złożony z urządzeń i protokołów zapewniający komunikację pomiędzy komputerami;

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna,
- praca na komputerze,
- pogadanka.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Środki dydaktyczne

- zestaw PC, laptop, komputer All in One lub stacja robocza z dostępem do Internetu, zainstalowaną przeglądarką i ustawieniami regionalnymi dla języka polskiego,
- rzutnik multimedialny lub tablica multimedialna,
- e-podręcznik.

Formy pracy

- Praca indywidualna.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przebieg lekcji:

Faza I. Wprowadzenie

Nauczyciel pyta uczniów czy pamiętają z jakich urządzeń i łączy korzystamy, żeby połączyć się z Internetem. Podane przez uczniów nazwy wypisuje na tablicy (router, WiFi, serwer, kabel, urządzenie końcowe: laptop, komputer, komórka). Nauczyciel mówi, że dzisiaj ta wiedza będzie trochę poszerzona, poznamy jeszcze inne urządzenia i dowiemy się w jaki sposób komputery porozumiewają się między sobą. Nauczyciel podaje temat lekcji.

Faza II. Część zasadnicza - realizacja

1. Nauczyciel wykorzystując e-platformę demonstruje na tablicy multimedialnej część teoretyczną uruchamiając odpowiedni moduł i materiał multimedialny.
2. Uczniowie logują się na swoich kontach na platformie edukacyjnej, gdzie otwierają lekcję w podanym przez nauczyciela module.
3. Prezentowany materiał multimedialny, zrealizowany w oparciu o szczegółowy scenariusz (załącznik nr 1 do lekcji), oprócz waloru teoretycznego stanowi również podstawę przebiegu części praktycznej lekcji. W szczególności, uczniowie:
 - a. poznają budowę sieci komputerowej: urządzenia i protokoły;
 - b. stosują polecenia: ipconfig i ping
4. Nauczyciel objaśnia sposób pracy na lekcji – we wskazanym przez niego momencie uczniowie będą wykonywać kolejne etapy prowadzące do rozwiązania zadania. Zdolniejsi uczniowie spróbują wykonać je samodzielnie. Słabsi wykonają kolejne kroki razem z nauczycielem, w oparciu o przygotowaną do zajęć instrukcję dostępną w e-podręczniku lub materiał multimedialny. W razie potrzeby będą mogli dopytać nauczyciela. Nauczyciel na bieżąco sprawdza postępy uczniów.

Faza III. Podsumowanie

Uczniowie prezentują swoje obrazy, nauczyciel omawia ich działanie oraz ocenia. Kryteria oceniania:

Ocena dopuszczająca:

– uczeń potrafi wymienić jedno urządzenie sieciowe;

Ocena dostateczna:

– uczeń zna jeden protokół sieciowy;

Ocena dobra:

– uczeń rozróżnia protokoły i urządzenia sieciowe.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Ocena bardzo dobra:

– uczeń potrafi powiedzieć jaka jest różnica między urządzeniem sieciowym, a protokołem.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, dostępnych również na platformie edukacyjnej w formie testu dla uczniów, np.:

1. Które z podanych to protokół?
 - a. **TCP**
 - b. router
 - c. serwer
2. Jakim poleceniem można sprawdzić jak szybko odpowiada inny komputer?
 - a. ipconfig
 - b. pong
 - c. **ping**
3. Jaką funkcję ma modem?
 - a. **łączy z dostawcą Internetu**
 - b. łączy komputery w sieć lokalną
 - c. tworzy w domu sieć bezprzewodową

Faza V. Zadanie do samodzielnego wykonania (dla chętnych).

Ping

- Wybierz jakiś inny serwer (gier, portal internetowy), sprawdź przez ping jaki jest jego IP i jak szybko odpowiada.
- Zastanów się i napisz dlaczego ważne jest, żeby ping od serwera gier był jak najszybszy.

Cyberbezpieczeństwo *

- Sprawdź co to jest atak DDoS i napisz czy komenda ping może być zagrożeniem?

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 10 – Budowa sieci komputerowej.

Przebieg lekcji:

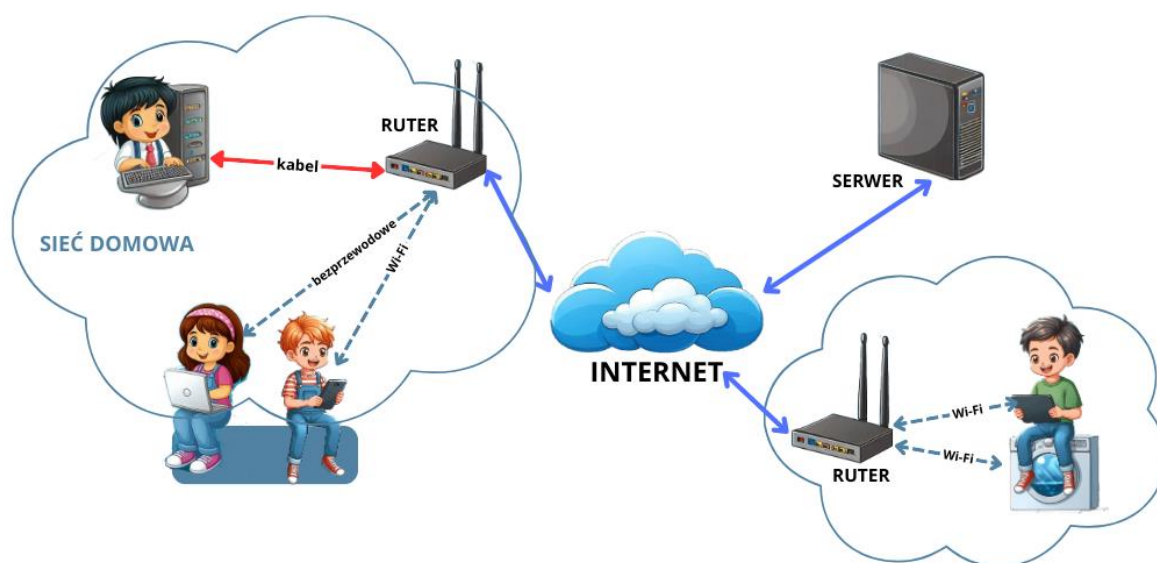
- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
- Przedstawienie uczniom celu i zakresu tematycznego lekcji.
- Uruchomienie na platformie RaP STEAM odpowiedniej e-lekcji.
- Indywidualna praca uczniów na komputerach.
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów (test na e-platformie).
- Zadanie pracy do samodzielnego wykonania – dostępne również na e-platformie (opcjonalnie).

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 2 – Realizacja zadania głównego podczas lekcji nr 10 – Budowa sieci komputerowej.

Przypominajka.

Czy wiesz jakie urządzenia należy wstawić w odpowiednie miejsce? /interaktywny moduł – przypomnienie z 4 klasy, gdzie trzeba przeciągnąć urządzenia w odpowiednie miejsce/



Z czego składa się sieć komputerowa?

Jak już wiecie, sieć komputerowa nie składa się z samych komputerów, do jej działania, czyli przesyłania informacji, potrzeba specjalnych urządzeń i protokołów.

Zapamiętaj:

Sieć komputerowa – zbiór połączonych ze sobą urządzeń, które wymieniają informacje według określonych zasad (protokołów).

Urządzenia

Poznaliście już router i najczęściej w sieciach domowych jest to właśnie główne urządzenie, na które mówimy, że tworzy sieć i daje dostęp do Internetu. Nie do końca tak jest.

Urządzeniem, które łączy komputery w sieć lokalną to switch. Modem jest bramą do

Internetu, zamienia sygnał operatora(dostawcy) na taki, który rozumie komputer. Router zaś

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

rozdziela Internet do różnych urządzeń w sieci. Najczęściej w domu nie mamy osobnego switcha, jest on wbudowany w router, który zazwyczaj ma także opcję WiFi. A często mamy wszystkie urządzenia, łącznie też z modemem, w jednym „pudełku”.



Protokoły, czyli język komunikacji

Po to, żeby ustalić metody komunikacji między komputerami w sieciach wymyślono protokoły, czyli jakby języki którymi urządzenia porozumiewają się ze sobą np.

IP – to zasada, że każde urządzenie w sieci ma swój adres

TCP - dzieli dane na mniejsze części (pakiety), numeruje je, sprawdza, czy wszystko dotarło i składa wiadomość w całość

HTTP / HTTPS - protokół, który mówi przeglądarce jak pobrać stronę.

Jedna czynność w Internecie używa kilku protokołów naraz. Na przykład, kiedy wchodzisz na stronę internetową korzystasz z:

IP → gdzie wysłać

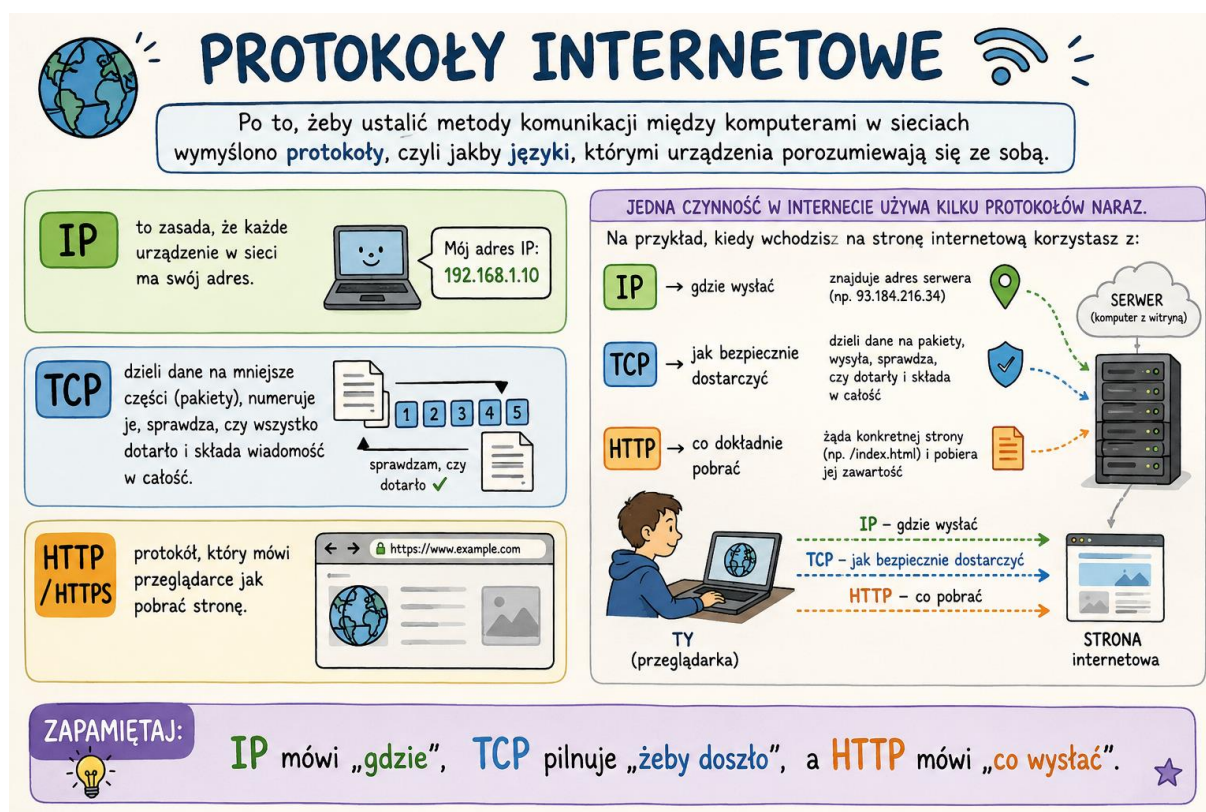
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

TCP → jak bezpiecznie dostarczyć

HTTP → co dokładnie pobrać

Zapamiętaj:

IP mówi „gdzie”, TCP pilnuje „żeby doszło”, a HTTP mówi „co wysłać”.

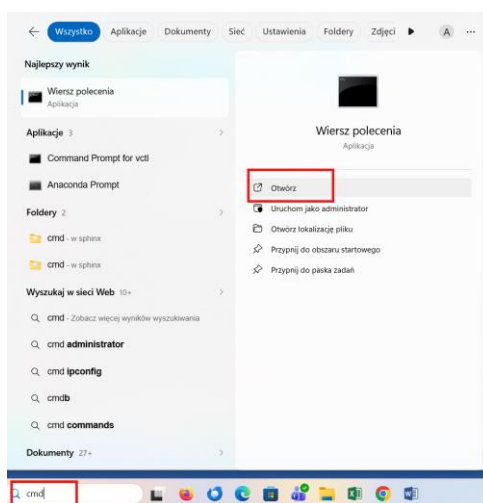


Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Sprawdź IP

Komputery w sieci mają swoje adresy, czyli IP. Jest to ciąg liczb, który identyfikuje każde urządzenie podpięte do Internetu. Mamy IP wewnętrzny, czyli adres w sieci wewnętrznej, zaś od operatora otrzymujemy IP zewnętrzne, które identyfikuje naszą sieć w Internecie.

Wpisz cmd w menu, uruchom wiersz poleceń. Wpisz **Ipconfig**



```
Wiersz polecenia
C:\Users>ipconfig

Windows IP Configuration

Ethernet adapter Ethernet:

    Connection-specific DNS Suffix  . : 
    Link-local IPv6 Address . . . . . : fe80::706b:91fc:ed58:1ed1%21
    IPv4 Address. . . . . : 172.16.2.18
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 172.16.2.1
```

Twój wewnętrzny adres IP jest podany tam gdzie jest IPv4 Address (czerwona ramka). Zapisz go i podaj nauczycielowi.

IP zewnętrzny najlepiej sprawdzić przez stronę internetową np.

<https://www.speedtest.pl/moje-ip> albo <https://whatsmydns.me/pl/whats-my-ip-address>. Strony zwracają informację z jakiego adresu łączysz się z serwerem webowym, czyli jaki jest Twój zewnętrzny IP. W pakiecie danych, które przesyłają jest zakodowane IP zewnętrzne i

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

wewnętrzne, tak, żeby informacje były dostarczone do odpowiedniego urzędu. Zapisz swój zewnętrzny IP i podaj nauczycielowi.



www.speedtest.pl/moje-ip

[Aplikacje](#) / [Moje IP - sprawdź Twój adres IP](#)

MOJE IP - SPRAWDŹ TWÓJ ADRES IP

Twoje IP to:
89.188.217.215

Na podstawie Twojego adresu IP i informacji zawartych w RIPE, za pośrednictwem zewnętrznej bazy danych określiliśmy Twoją nazwę dostawcy Internetu oraz geo-lokalizację.

Twój dostawca Internetu to:
Rzeszów University of Technology

Twoja geo-lokalizacja to:
Rzeszów

Jeśli powyższe dane są nieprawidłowe, istnieje możliwość dokonania ich korekty bezpośrednio w [formularzu](#) dostawcy bazy danych.

Jak odczytaliśmy Twój adres IP?

Twój adres IP został odczytany na serwerze SpeedTest.pl. Aby wyświetlić stronę z Twoim IP, przeglądarka utworzyła połączenie do naszego serwera, w którym jedną z informacji jest adres źródłowy nadawcy. Ten właśnie adres jest widoczny powyżej. Jeśli po drodze do Internetu masz router (np. z Wi-Fi), prezentowany tu adres będzie inny niż ustawiony na karcie sieciowej komputera lub smartfonu. Odpowiada za to mechanizm NAT.

Ping

Wyobraźcie sobie, że wasz komputer chce sprawdzić jakiś serwer albo komputer w ogóle istnieje i odpowiada. Polecenie ping właśnie tak działa, wysyła krótką informację do danego urządzenia i jeśli wszystko działa ok otrzymuje informację zrotną.

Uruchom wiersz poleceń i wpisz

```
ping google.com
```

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

```
Wiersz polecenia
C:\Users>ping google.com

Pinging google.com [142.251.98.138] with 32 bytes of data:
Reply from 142.251.98.138: bytes=32 time=20ms TTL=113
Reply from 142.251.98.138: bytes=32 time=30ms TTL=113
Reply from 142.251.98.138: bytes=32 time=17ms TTL=113
Reply from 142.251.98.138: bytes=32 time=26ms TTL=113

Ping statistics for 142.251.98.138:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 17ms, Maximum = 30ms, Average = 23ms

C:\Users>
```

W odpowiedzi dostajesz IP serwera google (w czerwonej ramce) i wyniki, jak szybko otrzymujesz odpowiedź (niebieska ramka) oraz statystyki (żółta ramka). Podaj nauczycielowi IP serwera google, najszybszy i najwolniejszy czas pakietu.

Zapytaj teraz kolegi/koleżanki obok o jego wewnętrzny adres IP. W wierszu poleceń wpisz

PING numer_ip_kolegi

Sprawdź jak szybko odpowiada.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie do samodzielnego wykonania (dla chętnych).

Ping

- Wybierz jakiś inny serwer (gier, portal internetowy), sprawdź przez ping jaki jest jego IP i jak szybko odpowiada.
- Zastanów się i napisz dlaczego ważne jest, żeby ping od serwera gier był jak najszybszy.

Cyberbezpieczeństwo *

- Sprawdź co to jest atak DDoS i napisz czy komenda ping może być zagrożeniem?

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 3: Globalna wspólnota Cyfrowego Świata

Temat lekcji nr 11 – Wpływ systemów komputerowych na życie człowieka.

Treści nauczania (wymagania szczegółowe - podstawa programowa) – IV. 2); V. 1)

Cel lekcji:

- Uczniowie rozwijają świadomość zagrożeń i umiejętności związane z korzystaniem z technologii komputerowych.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- Uczeń ocenia krytycznie informacje i ich źródła, w szczególności w sieci, pod względem rzetelności i wiarygodności w odniesieniu do rzeczywistych sytuacji, docenia znaczenie otwartych zasobów w sieci i korzysta z nich;
- Uczeń opisuje kwestie etyczne związane z wykorzystaniem komputerów i sieci komputerowych, takie jak: bezpieczeństwo, cyfrowa tożsamość, prywatność, własność intelektualna, równy dostęp do informacji i dzielenie się informacją.

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści programowe będą realizowane za pomocą uniwersalnych narzędzi dostępnych w systemach operacyjnych, przede wszystkim menedżerach okien. W miarę możliwości, spowodowanej dostępnością usługi zapisu danych w chmurze, uczniowie będą korzystać z zapisu danych w środowiskach wirtualnych.
- Realizacja za pomocą uniwersalnych narzędzi, w które wyposażone są wszystkie wykorzystywane w szkołach systemy operacyjne (np. eksploratory plików).

Uczeń:

- Rozpoznaje zalety i zagrożenia wynikające z używania technologii na co dzień;

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna,
- praca w grupach
- praca na komputerze,
- pogadanka.

Środki dydaktyczne

- zestaw PC, laptop, komputer All in One lub stacja robocza z dostępem do Internetu, zainstalowaną przeglądarką i ustawieniami regionalnymi dla języka polskiego,
- rzutnik multimedialny lub tablica multimedialna,
- e-podręcznik.

Formy pracy

- Praca w grupach i indywidualna.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przebieg lekcji:

Faza I. Wprowadzenie

Nauczyciel zadaje pytanie uczniom: „Wyobraźcie sobie: jutro znika Internet, telefony i komputery. Co przestaje działać?”. Odpowiedzi zapisuje na tablicy. Dzieli je na kategorie. Nauczyciel podaje temat lekcji.

Faza II. Część zasadnicza - realizacja

1. Nauczyciel wykorzystując e-platformę demonstruje na tablicy multimedialnej część teoretyczną uruchamiając odpowiedni moduł i materiał multimedialny.
2. Uczniowie logują się na swoich kontach na platformie edukacyjnej, gdzie otwierają lekcję w podanym przez nauczyciela module.
3. Prezentowany materiał multimedialny, zrealizowany w oparciu o szczegółowy scenariusz (załącznik nr 1 do lekcji), oprócz waloru teoretycznego stanowi również podstawę przebiegu części praktycznej lekcji. W szczególności, uczniowie:
 - a. w grupach wypisują zalety i wady technologii w poszczególnych kategoriach;
 - b. poszerzają wiedzę na temat zagrożeń związanych z używaniem technologii;
4. Nauczyciel objaśnia sposób pracy na lekcji – we wskazanym przez niego momencie uczniowie będą wykonywać kolejne etapy prowadzące do rozwiązania zadania. Zdolniejsi uczniowie spróbują wykonać je samodzielnie. Słabsi wykonają kolejne kroki razem z nauczycielem, w oparciu o przygotowaną do zajęć instrukcję dostępną w e-podręczniku lub materiał multimedialny. W razie potrzeby będą mogli dopytać nauczyciela. Nauczyciel na bieżąco sprawdza postępy uczniów.

Faza III. Podsumowanie

Uczniowie prezentują swoje obrazy, nauczyciel omawia ich działanie oraz ocenia. Kryteria oceniania:

Ocena dopuszczająca:

– uczeń potrafi opisać sytuację, w której używa technologii w życiu codziennym.

Ocena dostateczna:

– uczeń wskazuje różne przykłady użycia technologii w przynajmniej dwóch kategoriach

Ocena dobra:

– uczeń wymienia zalety i zagrożenia użycia technologii w jednej kategorii

Ocena bardzo dobra:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

– uczeń potrafi opisać zalety i zagrożenia technologiczne w kilku kategoriach

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, dostępnych również na platformie edukacyjnej w formie testu dla uczniów, np.:

- Wybierz co jest zagrożeniem w kategorii komunikacja
 - hejt**
 - szybkość komunikacji
 - dostępność
- W której sytuacji nie korzystamy z Internetu?
 - Płacimy kartą za zakupy
 - Sprawdzamy w mapach na komórce jak dojechać do kolegi
 - Gramy w piłkę nożną na boisku przy szkole**
- Co nie jest zagrożeniem:
 - komunikacja**
 - uzależnienie od technologii
 - wyciek danych osobowych

Faza V. Zadanie do samodzielnego wykonania (dla chętnych).

Tvoja opinia.

- Wybierz jedną kategorię (np. edukacja, zdrowie) i wypisz dwie korzyści i dwa zagrożenia związane z używaniem w niej technologii.

Cyberbezpieczeństwo*.

- Przypomnij sobie co to jest phishing i jakie są inne zagrożenia. Sprawdź się w teście.

test Google: <https://phishingquiz.withgoogle.com/>

Kahoot: <https://create.kahoot.it/details/5c5193b6-eb0a-44c5-ad71-ae6dcef3e86>

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 11 – Wpływ systemów komputerowych na życie człowieka.

Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
- Przedstawienie uczniom celu i zakresu tematycznego lekcji.
- Uruchomienie na platformie RaP STEAM odpowiedniej e-lekcji.
- Podział na grupy i praca w grupach bez komputerów.
- Indywidualna praca uczniów na komputerach.
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów (test na e-platformie).
- Zadanie pracy do samodzielnego wykonania – dostępne również na e-platformie (opcjonalnie).

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 2 – Realizacja zadania głównego podczas lekcji nr 11 – Wpływ systemów komputerowych na życie człowieka.

Internet, komputery, telefony....

Wyobraź sobie, że budzisz się rano... i coś jest nie tak. Sięgasz po telefon — nie ma go.

Chcesz napisać do kolegi — nie ma Messengera, nie ma SMS-ów.

Nie możesz sprawdzić planu lekcji, bo dziennik elektroniczny nie istnieje.

Wychodzisz z domu. Autobus? Nie wiadomo, kiedy przyjedzie.

Nie ma aplikacji, nie ma tablic elektronicznych. Są tylko tabliczki z rozkładem jazdy na przystankach.

W sklepie chcesz coś kupić — ale terminal nie działa.

Można płacić tylko gotówką. Nie działa bankomat, musisz iść do banku, żeby wybrać ją z konta.

Wracasz do domu. Chcesz się zrelaksować. Nie ma YouTube’a, Netflix’a, gier online.

I nagle orientujesz się...

Jak bardzo technologia jest wszędzie.

Systemy komputerowe to nie tylko komputery w klasie informatycznej.

To wszystko, co działa „w tle” naszego życia.

Kiedy wysyłasz wiadomość — działają serwery i sieci komputerowe.

Kiedy oglądasz film — dane płyną przez Internet.

Kiedy płacisz kartą — system bankowy sprawdza, czy masz pieniądze.

W szpitalach komputery pomagają lekarzom diagnozować choroby.

W samochodach pomagają w nawigacji i bezpieczeństwie.

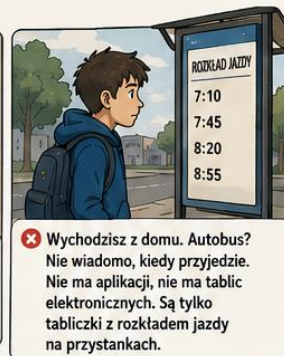
W szkołach — w nauce i komunikacji.

Technologia jest jak niewidzialna sieć, która łączy świat.

Ale jest też druga strona.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

WYOBRAŹ SOBIE, ŻE BUDZISZ SIĘ RANO... I COŚ JEST NIE TAK.



I NAGLE ORIENTUJESZ SIĘ... JAK BARDZO TECHNOLOGIA JEST WSZĘDZIE.

Systemy komputerowe to nie tylko komputery w klasie informatycznej.
To wszystko, co działa „w tle” naszego życia.

JAK TO DZIAŁA „W TLE”?



TECHNOLOGIA JEST JAK NIEWIDZIALNA SIĘĆ, KTÓRA ŁĄCZY ŚWIAT.



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zagrożenia

Systemy komputerowe ułatwiają życie — ale niosą też zagrożenia.

Dlatego tak ważne jest **cyberbezpieczeństwo**. Kiedy korzystasz z Internetu, twoje dane mogą być narażone na ataki. Może to być kradzież hasła, włamanie na konto albo próba oszustwa.

Na przykład:

- ktoś podszywa się pod znajomego i prosi o kod BLIK,
- albo wysyła link, który wygląda jak prawdziwa strona, ale nią nie jest.

Dlatego ważne jest:

- używanie silnych haseł
- nieklikanie w podejrzaną linki
- sprawdzanie, komu podajesz swoje dane

Drugim ważnym problemem jest **prywatność**. Korzystając z aplikacji i stron internetowych, zostawiasz po sobie ślad:

- zdjęcia
- wiadomości
- lokalizację

Te dane mogą być zbierane i wykorzystywane, często bez pełnej świadomości użytkownika. Dlatego warto:

- nie udostępniać wszystkiego publicznie
- sprawdzać ustawienia prywatności
- zastanowić się, zanim coś opublikujesz

Kolejnym zagrożeniem jest **uzależnienie od technologii**. Długie korzystanie z telefonu lub komputera może:

- pogarszać koncentrację
- wpływać na sen
- ograniczać kontakty z innymi ludźmi

Warto pamiętać o równowadze: czas online i offline powinien być zrównoważony.

Na koniec ważna rzecz: Technologia sama w sobie nie jest ani dobra, ani zła. To, jak wpływa na nasze życie, zależy od tego, jak z niej korzystamy.

Dlatego najważniejsze jest jedno:

Korzystaj świadomie i bezpiecznie.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

SYSTEMY KOMPUTEROWE UŁATWIAJĄ ŻYCIE – ALE NIOSĄ TEŻ ZAGROŻENIA.

Dlatego tak ważne jest **cyberbezpieczeństwo**.

Kiedy korzystasz z Internetu, twoje dane mogą być narażone na ataki.
Może to być kradzież hasła, włamanie na konto albo próba oszustwa.



NA PRZYKŁAD:

Ktoś podszywa się pod
znajomego i prosi o kod BLIK.



Cześć! To ja, mam
problem z płatnością.
Wyślij mi kod BLIK.



Jasne, zaraz
wyślę.



Albo wysyła link, który wygląda
jak prawdziwa strona, ale nią
nie jest.



<https://twoj-bank.pl> ❌



<https://bank-prawdziwy.pl> ✅



Uważaj na fałszywe strony!
Zawsze sprawdzaj adres.

DLATEGO WAŻNE JEST:

Używanie
silnych haseł



***** ✅

Nieklikanie w
podejrzane linki



Sprawdzanie, komu
podajesz swoje dane



DRUGIM WAŻNYM PROBLEMEM JEST PRYWATNOŚĆ.



Korzystając z aplikacji
i stron internetowych,
zostawiasz po sobie ślad:



zdjęcia



wiadomości



lokalizację



Te dane mogą być zbierane
i wykorzystywane, często bez
pełnej świadomości użytkownika.

DLATEGO WARTO:



Nie udostępniać
wszystkiego publicznie



Sprawdzać ustawienia
prywatności



Zastanowić się, zanim
coś opublikujesz

KOLEJNYM ZAGROŻENIEM JEST UZALEŻNIENIE OD TECHNOLOGII.

Długie korzystanie z telefonu
lub komputera może:



pogarszać koncentrację



wpływać na sen



ograniczać kontakty
z innymi ludźmi



WARTO PAMIĘTAĆ O RÓWNOWADZE:

czas online i offline powinien
być zrównoważony.



CZAS ONLINE



CZAS OFFLINE



Rób przerwy



Wyłącz
powiadomienia



Spędzaj czas
z ludźmi



Ruch i hobby
są ważne

NA KONIEC WAŻNA RZECZ:

Technologia sama w sobie
nie jest ani dobra, ani zła.



Dlatego najważniejsze
jest jedno:



TECHNOLOGIA TO POTĘŻNE NARZĘDZIE. UŻYWAJ JEJ MĄDRZE, BĄDŹ BEZPIECZNY I CIESZ SIĘ TYM,
CO NAJWAŻNIEJSZE – **PRAWDZIWYM ŻYCIEM.**

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Technologia jest wszędzie

Podzielcie się na 2,3-osobowe grupy. Nauczyciel każdej grupie przydzieli kategorię:

- komunikacja
- edukacja
- zdrowie
- rozrywka
- transport
- bankowość
- zakupy
- praca
- administracja

Przykłady dla nauczyciela:

Komunikacja: + szybki kontakt (Messenger, Teams) / – uzależnienie, hejt

Zdrowie: + szybka diagnoza, e-recepty / – błędy systemów, wyciek danych

Transport: + GPS, aplikacje typu Uber / – uzależnienie od technologii

Bankowość: + szybkie przelewy / – oszustwa internetowe

Każda grupa odpowiada na 3 pytania:

1. Gdzie tu są komputery?
2. Co to ułatwia?
3. Jakie są zagrożenia?

Przedstawcie całą klasę w jednym zdaniu:

„Komputery w [komunikacji/transporte/bankowości] pomagają w... ale problemem jest...”

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



MINIPROJEKT TECHNOLOGIA JEST WSZĘDZIE





Podzielcie się na 2,3-osobowe grupy.
Nauczyciel każdej grupy przydzieli kategorię:

KOMUNIKACJA


EDUKACJA


ZDROWIE


ROZRYWKA


TRANSPORT


BANKOWOŚĆ


ZAKUPY


PRACA


ADMINISTRACJA


1 Gdzie tu są komputery?



Wskażcie miejsca lub urządzenia, w których działają komputery i systemy komputerowe.

2 Co to ułatwia?



Jak technologia poprawia, przyspiesza lub ułatwia codzienne życie w tej kategorii?

3 Jakie są zagrożenia?



Zastanówcie się, jakie problemy lub ryzyka mogą się wiązać z używaniem technologii w tej dziedzinie.

PRZEDSTAWCIE CAŁEJ KLASIE W JEDNYM ZDANIU:



„Komputery w [komunikacji/transporte/bankowości] .. pomagają w ...
ale problemem jest ...”







Podkarpacki Zespół
Placówek Wojewódzkich
w Rzeszowie



POLITECHNIKA
RZESZOWSKA
im. IGNACEGO ŁUKASIEWICZA



ODK
WOJEWÓDZKI DOM KULTURY

184

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie do samodzielnego wykonania.

Twoja opinia.

- Wybierz jedną kategorię (np. edukacja, zdrowie) i wypisz dwie korzyści i dwa zagrożenia związane z używaniem w niej technologii.

Cyberbezpieczeństwo*.

- Przypomnij sobie co to jest phishing i jakie są inne zagrożenia. Sprawdź się w teście.

test Google: <https://phishingquiz.withgoogle.com/>

Kahoot: <https://create.kahoot.it/details/5c5193b6-eb0a-44c5-ad71-ae6dcef3e86>

Test na platformie:

1. Co to jest phishing?

A. Program do ochrony komputera

B. Próba wyludzenia danych przez podszywanie się

C. Szybkie łącze internetowe

2. Które hasło jest najbezpieczniejsze?

A. 123456

B. kasia123

C. K@t9!Lp#21

3. Co powinieneś zrobić, gdy dostaniesz podejrzany link?

A. Kliknąć szybko

B. Zignorować lub sprawdzić źródło

C. Wysłać dalej do znajomych

4. Co to jest malware?

A. Złośliwe oprogramowanie

B. Program do nauki

C. Program do gier

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

5. Co może być przykładem phishingu?

A. Wiadomość od kolegi „co tam?”

B. Mail „Twoje konto zablokowane, kliknij link”

C. Reklama butów

6. Co to jest cyberprzemoc?

A. Gra komputerowa

B. Pomoc online

C. Krzywdzenie innych w Internecie

7. Dlaczego nie należy podawać danych osobowych?

A. Bo są nudne

B. Bo mogą zostać wykorzystane przeciwko tobie

C. Bo są niepotrzebne

8. Co oznacza „uzależnienie od technologii”?

A. Korzystanie z komputera do nauki

B. Zbyt częste i niekontrolowane korzystanie

C. Granie raz dziennie

9. Co to jest fałszywa strona internetowa?

A. Strona z błędami

B. Strona udająca prawdziwą

C. Strona bez zdjęć

10. Co zrobić, gdy ktoś prosi o kod BLIK?

A. Podać od razu

B. Sprawdzić, czy to na pewno ta osoba

C. Zignorować zawsze

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

11. Każdy link w Internecie jest bezpieczny

Prawda/**Falsz**

12. Silne hasło powinno mieć różne znaki

Prawda/Falsz

13. Można podawać hasło znajomym

Prawda/**Falsz**

14. Wirusy mogą ukraść dane

Prawda/Falsz

15. Sztuczna inteligencja zawsze mówi prawdę

Prawda/**Falsz**

16. Kolega pisze: „Hej, podeślij kod BLIK, oddam za chwilę”

Co robisz?

Napisz odpowiedź.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 3: Globalna wspólnota Cyfrowego Świata

Temat lekcji nr 12 – Sieci rozległe jako kluczowy element globalnej komunikacji.

Treści nauczania (wymagania szczegółowe - podstawa programowa) – III. 1); III. 3)

Cel lekcji:

- Uczniowie poznają kluczową rolę sieci rozległych.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- Uczeń przedstawia funkcjonowanie sieci komputerowej i sieci Internet;
- Uczeń poprawnie posługuje się terminologią związaną z informatyką i technologią.

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści programowe będą realizowane za pomocą uniwersalnych narzędzi dostępnych w systemach operacyjnych, przede wszystkim menedżerach okien. W miarę możliwości, spowodowanej dostępnością usługi zapisu danych w chmurze, uczniowie będą korzystać z zapisu danych w środowiskach wirtualnych.
- Realizacja za pomocą uniwersalnych narzędzi, w które wyposażone są wszystkie wykorzystywane w szkołach systemy operacyjne (np. eksploratory plików).

Uczeń:

- Poznaje historię Internetu, jego rozwój i kluczową rolę sieci rozległych;

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna,
- praca na komputerze,
- pogadanka.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Środki dydaktyczne

- zestaw PC, laptop, komputer All in One lub stacja robocza z dostępem do Internetu, zainstalowaną przeglądarką i ustawieniami regionalnymi dla języka polskiego,
- rzutnik multimedialny lub tablica multimedialna,
- e-podręcznik.

Formy pracy

- Praca indywidualna.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przebieg lekcji:

Faza I. Wprowadzenie

Nauczyciel podaje temat lekcji i zadaje pytanie uczniom: „Jak to możliwe, że możecie napisać do kogoś na drugiej półkuli i dostać odpowiedź w sekundę?”.

Faza II. Część zasadnicza - realizacja

1. Nauczyciel wykorzystując e-platformę demonstruje na tablicy multimedialnej część teoretyczną uruchamiając odpowiedni moduł i materiał multimedialny.
2. Uczniowie logują się na swoich kontach na platformie edukacyjnej, gdzie otwierają lekcję w podanym przez nauczyciela module.
3. Prezentowany materiał multimedialny, zrealizowany w oparciu o szczegółowy scenariusz (załącznik nr 1 do lekcji), oprócz waloru teoretycznego stanowi również podstawę przebiegu części praktycznej lekcji. W szczególności, uczniowie:
 - a. poznają historię Internetu;
 - b. odkrywają kluczową rolę WAN w komunikacji;
4. Nauczyciel objaśnia sposób pracy na lekcji – we wskazanym przez niego momencie uczniowie będą wykonywać kolejne etapy prowadzące do rozwiązania zadania. Zdolniejsi uczniowie spróbują wykonać je samodzielnie. Słabsi wykonają kolejne kroki razem z nauczycielem, w oparciu o przygotowaną do zajęć instrukcję dostępną w e-podręczniku lub materiał multimedialny. W razie potrzeby będą mogli dopytać nauczyciela. Nauczyciel na bieżąco sprawdza postępy uczniów.

Faza III. Podsumowanie

Uczniowie prezentują swoje obrazy, nauczyciel omawia ich działanie oraz ocenia. Kryteria oceniania:

Ocena dopuszczająca:

– uczeń wie co to jest sieć rozległa.

Ocena dostateczna:

– uczeń rozumie znaczenie sieci rozległych w komunikacji

Ocena dobra:

– uczeń wymienia różnicę między siecią lokalną, a rozległą

Ocena bardzo dobra:

– uczeń potrafi przedstawić w skrócie historię Internetu

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, dostępnych również na platformie edukacyjnej w formie testu dla uczniów, np.:

1. Co to jest WAN?
 - a. **Sieć rozległa**
 - b. Sieć bezprzewodowa
 - c. Sieć lokalna
2. Czy Internet to sieć rozległa?
 - a. nie
 - b. tak**
3. Co było najwcześniej?
 - a. ARPANET**
 - b. Google
 - c. Youtube

Faza V. Zadanie do samodzielnego wykonania (dla chętnych).

Tracert.

- Wpisz w wiersz poleceń

tracert google.com

Sprawdź przez jakie serwery przechodzi pakiet, który jest wysyłany do Google. Czy wszystkie serwery po drodze odpowiadają?

Powtórz jeszcze raz to samo polecenie i sprawdź czy droga jest taka sama.

Powtórz to polecenie dla IP kolegi (jeśli jesteś w szkole) i dowolnego innego serwera (może być serwer gier, portal internetowy)

Cyberbezpieczeństwo*.

- Spróbuj zrobić timeline zagrożeń związanych z Internetem (np. pierwszy wirus komputerowy).

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 12 – Sieci rozległe jako kluczowy element globalnej komunikacji.

Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
- Przedstawienie uczniom celu i zakresu tematycznego lekcji.
- Uruchomienie na platformie RaP STEAM odpowiedniej e-lekcji.
- Indywidualna praca uczniów na komputerach.
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów (test na e-platformie).
- Zadanie pracy do samodzielnego wykonania – dostępne również na e-platformie (opcjonalnie).

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 2 – Realizacja zadania głównego podczas lekcji nr 12 – Sieci rozległe jako kluczowy element globalnej komunikacji.

WAN

Wyobraź sobie, że siedzisz w domu i oglądasz film w internecie.

Klikasz „play”... i po chwili wszystko działa.

Ale czy zastanawiałeś się kiedyś, jak to możliwe, że film trafia do Ciebie z drugiego końca świata?

To właśnie dzięki sieci WAN.

WAN, czyli Wide Area Network, to sieć rozległa.

Obejmuje ogromne obszary — miasta, kraje, a nawet cały świat.

To ona łączy ze sobą mniejsze sieci, takie jak te w Twoim domu czy szkole, czyli sieci LAN.

Kiedy korzystasz z internetu, Twoje urządzenie wysyła dane.

Te dane podróżują przez wiele różnych sieci, kabli i urządzeń, zanim dotrą do celu.

Czasem jest to serwer w innym mieście.

Czasem... na innym kontynencie.

Dzięki WAN możesz:

rozmawiać z kimś na drugim końcu świata,

oglądać filmy online,

grać w gry przez internet,

czy wysyłać wiadomości w kilka sekund.

Największym przykładem sieci WAN jest Internet.

To globalna sieć, która łączy miliardy urządzeń.

Można powiedzieć, że WAN to taka „autostrada informacji”,

po której dane przemieszczają się między różnymi miejscami na Ziemi.

Bez niej świat wyglądałby zupełnie inaczej.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Nie byłoby mediów społecznościowych,

filmów online,

ani szybkiego dostępu do informacji.

WAN sprawia, że świat jest połączony.

I właśnie dlatego jest tak ważna.

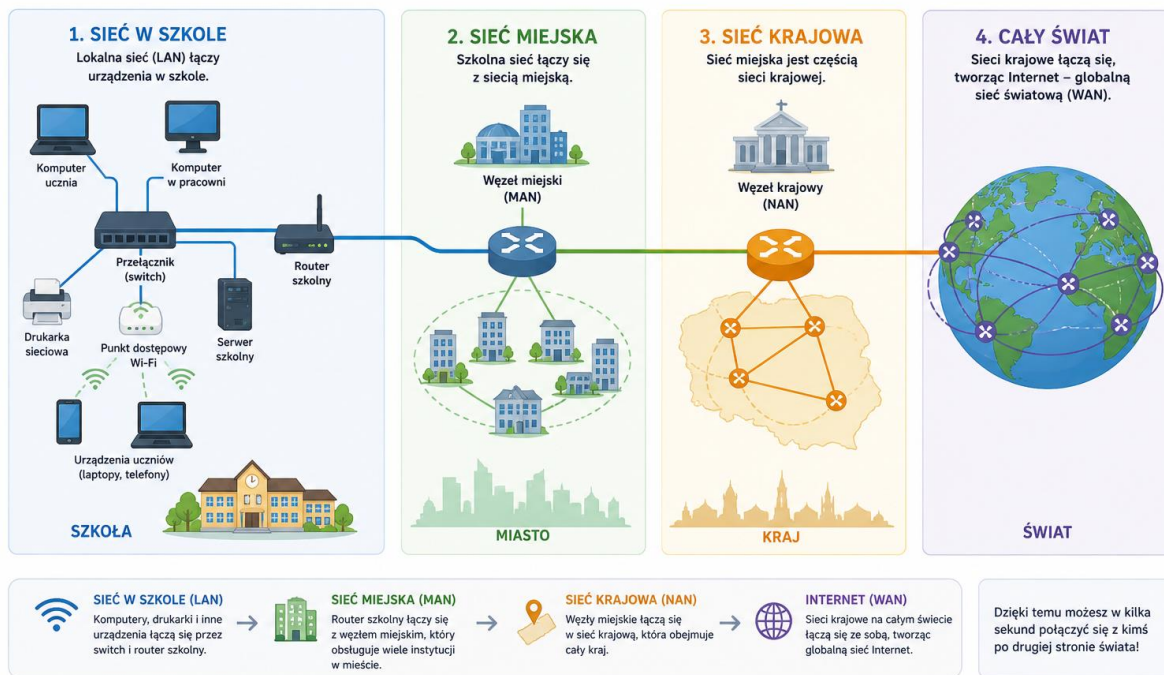
Czym się różni LAN (lokalna sieć) od WAN (sieć rozległa)?



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

JAK SIECI ŁĄCZĄ ŚWIAT?

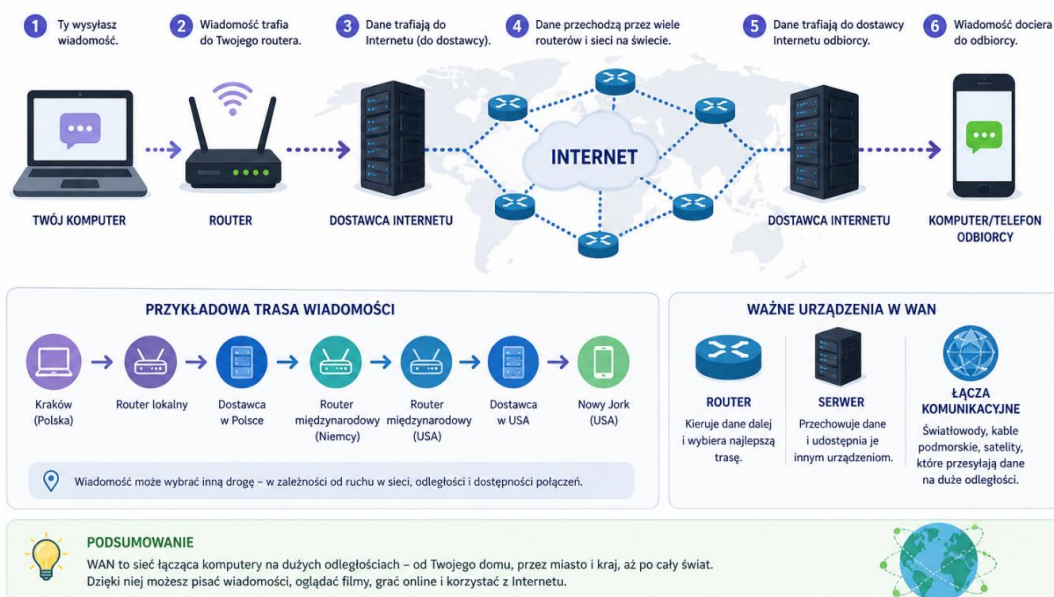
Od małej sieci w szkole do globalnej komunikacji



Jak podróżują informacje?

JAK DZIAŁA WAN?

Twoja wiadomość nie leci w linii prostej. Podróżuje przez wiele urządzeń i wybiera najlepszą trasę.



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Timeline Internetu /jako interaktywny element h5p/



Aktywność Mini quiz:

Mini quiz o historii Internetu :)

ARPANET był...

- ☐ grą komputerową
- ☐ pierwszą siecią komputerową
- ☐ telefonem
- ☐ wyszukiwarką

✓ Sprawdź



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie do samodzielnego wykonania.

Tracert.

- Wpisz w wiersz poleceń

`tracert google.com`

Sprawdź przez jakie serwery przechodzi pakiet, który jest wysyłany do Google. Czy wszystkie serwery po drodze odpowiadają?

Powtórz jeszcze raz to samo polecenie i sprawdź czy droga jest taka sama.

Powtórz to polecenie dla IP kolegi (jeśli jesteś w szkole) i dowolnego innego serwera (może być serwer gier, portal internetowy)

Cyberbezpieczeństwo*.

- Spróbuj zrobić timeline zagrożeń związanych z Internetem (np. pierwszy wirus komputerowy).

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 4: Biurowa moc: TechOffice

Temat lekcji nr 13 - Jak wykonać dokumentację oprogramowania?

Treści nauczania (wymagania szczegółowe - podstawa programowa) – II. 3) b);

Cel lekcji

Celem lekcji jest zapoznanie uczniów z zasadami tworzenia dokumentacji oprogramowania na przykładzie programu robota LEGO SPIKE. Uczniowie poznają dokumentację jako uporządkowany opis programu, który pozwala innym osobom zrozumieć jego działanie, uruchomić go, przetestować oraz w przyszłości poprawić lub rozbudować.

Podczas zajęć uczniowie uczą się przygotowywać dokumentację zawierającą tytuł i podstawowe informacje, cel projektu, opis działania programu, listę wykorzystanych elementów, instrukcję uruchomienia, opis kodu lub bloków, testowanie, wyniki oraz wnioski. Lekcja rozwija umiejętność logicznego porządkowania informacji, stosowania nagłówków, tworzenia czytelnych opisów technicznych oraz dokumentowania pracy projektowej.

Cele szczegółowe lekcji

Uczeń:

- wyjaśnia, czym jest dokumentacja oprogramowania,
- rozumie, po co tworzy się dokumentację programu,
- wskazuje podstawowe elementy dokumentacji oprogramowania,
- przygotowuje dokument tekstowy opisujący program robota,
- stosuje nagłówki do porządkowania treści dokumentu,
- tworzy stronę tytułową dokumentacji,
- zapisuje nazwę projektu, autora, datę i wersję programu,
- opisuje cel programu robota,
- opisuje działanie programu krok po kroku,
- wskazuje wykorzystane elementy LEGO SPIKE, np. hub, silniki, czujniki,
- przygotowuje instrukcję uruchomienia programu,
- opisuje kod lub bloki programu w sposób zrozumiały dla użytkownika,
- przedstawia logikę działania programu w formie tekstowej lub tabelarycznej,
- wyjaśnia znaczenie warunku, pętli i reakcji robota na odczyt z czujnika,
- opisuje testowanie programu i wyniki testów,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- formułuje wnioski oraz propozycje ulepszeń,
- rozumie znaczenie czytelności i kompletności dokumentacji,
- zapisuje efekty pracy w pliku lokalnym lub w chmurze.

Treści nauczania – zgodność z podstawą programową

Lekcja realizuje wymagania podstawy programowej poprzez praktyczne przygotowanie dokumentu tekstowego zawierającego uporządkowany opis projektu i programu robota. Uczeń posługuje się edytorem tekstu jako podstawową aplikacją komputerową, tworzy dokument z nagłówkami, opisami, listami, tabelą dokumentacyjną oraz informacjami technicznymi.

Zajęcia rozwijają umiejętność porządkowania i prezentowania informacji w dokumencie cyfrowym. Uczniowie tworzą dokumentację programu robota wykrywającego przeszkody, opisują jego działanie, warunki logiczne, użyte elementy, sposób uruchomienia, testowanie oraz wnioski. W materiale lekcyjnym podkreślono, że dokumentacja powinna być jasna, uporządkowana i zrozumiała, aby inna osoba mogła zrozumieć program, poprawić go, rozbudować lub używać bez dodatkowych wyjaśnień.

Sposób realizacji wymagań w innowacyjnym programie nauczania

Wymagania programowe będą realizowane poprzez praktyczną pracę uczniów w edytorze tekstu, np. LibreOffice Writer, Microsoft Word lub Dokumenty Google. Uczniowie przygotowują dokumentację programu robota LEGO SPIKE, wykorzystując strukturę typową dla dokumentacji technicznej: tytuł, cel projektu, opis działania, wykorzystane elementy, opis programu, instrukcję uruchomienia, testowanie i wnioski.

Lekcja łączy umiejętności informatyczne, robotyczne i komunikacyjne. Uczniowie nie tylko tworzą dokument w edytorze tekstu, ale także uczą się wyjaśniać działanie programu w sposób techniczny, logiczny i zrozumiały. Dokumentacja programu może zawierać opis kroków algorytmu, tabelę z komentarzami technicznymi, zrzuty ekranu z programu blokowego oraz wyniki testów robota.

Metody pracy

- pokaz multimedialny lub demonstracja nauczyciela,
- pogadanka kierowana,
- analiza przykładowej dokumentacji programu,
- ćwiczenia praktyczne przy komputerze,
- praca z instrukcją krok po kroku,
- samodzielne tworzenie dokumentacji,
- opis algorytmu w formie tekstowej i tabelarycznej,
- analiza działania programu robota,
- formułowanie wniosków i propozycji usprawnień.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Środki dydaktyczne

- komputer uczniowski z edytorem tekstu,
- edytor tekstu, np. LibreOffice Writer, Microsoft Word lub Dokumenty Google,
- aplikacja LEGO SPIKE,
- zestaw LEGO SPIKE z hubem, silnikami i czujnikiem odległości,
- przykładowy program robota wykrywającego przeszkody,
- przykładowa dokumentacja programu,
- zrzuty ekranu bloków programu,
- karta pracy lub e-podręcznik,
- rzutnik multimedialny lub ekran interaktywny,
- dostęp do zapisu plików lokalnie lub w chmurze.

Formy pracy

- praca indywidualna przy komputerze,
- praca pod kierunkiem nauczyciela,
- ćwiczenia samodzielne,
- elementy pracy projektowej,
- elementy pracy analitycznej podczas opisywania algorytmu,
- krótkie sprawdzenie efektów pracy ucznia.

Faza I. Wprowadzenie

Nauczyciel rozpoczyna lekcję od rozmowy z uczniami na temat tego, co powinno zostać zapisane po wykonaniu programu robota. Pyta, czy sama obecność działającego programu wystarczy, aby inna osoba mogła go zrozumieć, poprawić lub uruchomić. Następnie wyjaśnia, że dokumentacja oprogramowania jest opisem programu, jego celu, działania, sposobu uruchomienia, testowania oraz możliwych usprawnień.

Nauczyciel przedstawia przykład programu robota LEGO SPIKE, którego zadaniem jest wykrycie przeszkody i zatrzymanie się przed nią. Uczniowie omawiają, jakie informacje powinny znaleźć się w dokumentacji takiego programu: co robot robi, z jakich elementów korzysta, jak działa algorytm, kiedy robot jedzie, kiedy się zatrzymuje i jak sprawdzono poprawność programu.

Wprowadzenie kończy się zapisaniem tematu lekcji oraz określeniem zadania głównego: uczniowie przygotowują dokumentację programu robota wykrywającego przeszkody i zatrzymującego się przed nimi.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Faza II. Część zasadnicza – realizacja

W części praktycznej uczniowie uruchamiają edytor tekstu i zapisują plik pod nazwą „Dokumentacja oprogramowania robota wykrywającego przeszkody”. Następnie tworzą stronę tytułową dokumentacji, zawierającą tytuł projektu, autora, datę wykonania i szkołę. Uczniowie stosują nagłówki, aby nadać dokumentowi przejrzystą strukturę.

Kolejnym krokiem jest przygotowanie sekcji „Cel projektu”. Uczniowie zapisują, że celem programu jest zaprogramowanie robota, który zatrzymuje się przed przeszkodą dzięki wykorzystaniu czujnika odległości. Następnie tworzą sekcję „Wykorzystane elementy LEGO SPIKE”, w której wymieniają hub, dwa silniki, czujnik odległości oraz elementy konstrukcyjne pojazdu.

W sekcji „Opis działania robota” uczniowie wyjaśniają, że robot jedzie do przodu, a po wykryciu przeszkody w odległości mniejszej lub równej 15 cm zatrzymuje się, czeka chwilę i cofa. Następnie przechodzą do sekcji „Program - opis bloków”, w której przedstawiają działanie programu krok po kroku: ustawienie mocy silników, jazdę do przodu, sprawdzanie odległości w pętli, zatrzymanie silników, odczekanie jednej sekundy, cofnięcie robota i zakończenie programu.

Następnie uczniowie poznają tabelaryczną formę dokumentacji programu. Tworzą lub uzupełniają tabelę zawierającą kolumny: „Krok”, „Element programu”, „Opis działania”, „Warunek / logika” oraz „Komentarz techniczny”. Dzięki temu uczą się, że dokumentacja może mieć formę uporządkowanego zestawienia, w którym każdy fragment programu jest opisany technicznie i logicznie.

W dalszej części uczniowie przygotowują instrukcję uruchomienia robota: połączenie huba z komputerem lub tabletem, otwarcie aplikacji LEGO SPIKE, wgranie programu, ustawienie robota na równej powierzchni, ustawienie przeszkody i uruchomienie programu. Następnie opisują testowanie robota na przykładzie kilku przeszkód oraz zapisują wyniki testów.

Na zakończenie części praktycznej uczniowie formułują wnioski. Zapisują, czy program działa poprawnie, jakie ograniczenia może mieć czujnik odległości oraz jak można ulepszyć program w przyszłości, np. dodać skręcanie, aby robot omijał przeszkody zamiast tylko się zatrzymywać. W zadaniu dodatkowym uczniowie mogą przygotować podobną dokumentację dla programu robota jadącego po linii.

Faza III. Podsumowanie

Na zakończenie lekcji uczniowie prezentują przygotowaną dokumentację. Nauczyciel sprawdza, czy dokument zawiera najważniejsze elementy: tytuł, podstawowe informacje, cel projektu, opis działania robota, wykorzystane elementy, opis programu, instrukcję uruchomienia, testowanie i wnioski.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Uczniowie odpowiadają na pytania kontrolne dotyczące tego, co powinno znaleźć się w dokumentacji oprogramowania, dlaczego stosuje się nagłówki, do czego służy spis treści, kiedy robot powinien się zatrzymać oraz co dzieje się po zatrzymaniu robota. Nauczyciel podkreśla, że dobra dokumentacja jest ważna, ponieważ umożliwia zrozumienie programu, jego bezpieczne uruchomienie, testowanie i dalsze rozwijanie.

Kryteria oceniania

Ocena dopuszczająca

Uczeń z pomocą nauczyciela tworzy prosty dokument, wpisuje tytuł projektu i wybrane informacje o programie robota.

Ocena dostateczna

Uczeń samodzielnie przygotowuje podstawową dokumentację zawierającą cel projektu, wykorzystane elementy, krótki opis działania robota i instrukcję uruchomienia.

Ocena dobra

Uczeń poprawnie porządkuje dokumentację, stosuje nagłówki, opisuje działanie programu krok po kroku, przedstawia testowanie i formułuje proste wnioski.

Ocena bardzo dobra

Uczeń samodzielnie i świadomie tworzy kompletną dokumentację oprogramowania, opisuje algorytm w sposób logiczny, stosuje tabelę dokumentacyjną, uwzględnia warunki działania programu, wyniki testów oraz propozycje usprawnień.

Krótki opis problemu do rozwiązania

Uczniowie mają przygotować dokumentację programu robota LEGO SPIKE wykrywającego przeszkody. Problem polega na tym, aby nie ograniczyć się do samego programu, lecz opisać go tak, aby inna osoba mogła zrozumieć jego cel, sposób działania, użyte elementy, warunki logiczne, instrukcję uruchomienia i wyniki testów. Dzięki temu uczniowie uczą się, że dokumentacja jest ważną częścią tworzenia oprogramowania i projektów robotycznych.

Faza IV. Quiz/Test sprawdzający

1. Co powinno znaleźć się w dokumentacji oprogramowania?

A. Tylko nazwa programu

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

B. Opis działania programu i instrukcja użytkowania

C. Same zdjęcia

2. Dlaczego stosujemy nagłówki w dokumentacji?

A. Aby zmienić kolor tekstu

B. Aby uporządkować dokument i ułatwić nawigację

C. Aby zmniejszyć ilość tekstu

3. Do czego służy spis treści w dokumentacji?

A. Do ozdabiania dokumentu

B. Do usuwania błędów

C. Do szybkiego odnajdywania informacji w dokumencie

4. Kiedy robot powinien się zatrzymać?

A. Gdy odległość > 15 cm

B. Gdy odległość ≤ 15 cm

C. Zawsze

5. Co dzieje się po zatrzymaniu robota?

A. Robot skręca

B. Robot czeka 1 sekundę, a następnie cofa się

C. Robot przyspiesza

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 13 - Jak wykonać dokumentację oprogramowania?

Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
- Przedstawienie uczniom celu i zakresu tematycznego lekcji.
- Uruchomienie na e-platformie materiałów do lekcji
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów.
- Zadanie pracy domowej (opcjonalnie).

Załącznik nr 2 – Konspekt nauczyciela lekcji nr 13 - Jak wykonać dokumentację oprogramowania?

Cel lekcji:

Wprowadzenie

Tydzień 13

Co powinna zawierać dokumentacja projektu?

Cel ćwiczenia: Celem zajęć jest poznanie zasad tworzenia dokumentacji oprogramowania oraz rozwinięcie umiejętności jej przygotowywania w sposób przejrzysty i użyteczny, tak aby wspierała użytkowników w efektywnym korzystaniu z systemu i minimalizowała ryzyko błędów oraz nieporozumień.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Dokumentacja projektu budowy robota z wykorzystaniem LEGO SPIKE powinna być przejrzysta, szczegółowa i uporządkowana, aby ułatwić zarówno zrozumienie projektu, jak i jego ewentualne odtworzenie lub rozwój.

Dokumentacja powinna być **jasna, uporządkowana i zrozumiała**, tak aby ktoś inny mógł:

- zrozumieć, jak działa program,
- poprawić go lub rozbudować,
- używać go bez dodatkowych wyjaśnień.

Oto zalecane elementy dokumentacji:

Struktura dokumentacji oprogramowania

1. Tytuł i podstawowe informacje

- Nazwa projektu
- Autor
- Data stworzenia
- Wersja programu

2. Cel projektu

Krótki opis, po co powstał program, jakie zadanie wykonuje robot oraz w jakim problemie pomaga.

3. Opis działania programu

Należy wyjaśnić:

- jak robot ma się zachowywać,
- w jakiej kolejności wykonywane są działania,
- jakie czujniki i elementy są używane.

Można użyć schematu blokowego lub prostego rysunku.

4. Wymagane elementy

Lista komponentów LEGO SPIKE:

- czujniki,
- silniki,
- moduły,
- dodatkowe elementy konstrukcyjne.



Podkarpacki Zespół
Placówek Wojewódzkich
w Rzeszowie



POLITECHNIKA
RZESZOWSKA
Im. IGNACEGO ŁUKASIEWICZA



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

5. Instrukcja uruchomienia

Krok po kroku:

- jak wgrać program do robota,
- jak uruchomić program,
- co musi zrobić użytkownik przed startem (np. ustawić robota na linii).

6. Opis kodu lub bloków

Dla programu w SPIKE podaj:

- screen programu z blokami **lub**
- listę instrukcji tekstowych.

Do każdego kluczowego fragmentu dodaj krótkie wyjaśnienie.

7. Testowanie i wyniki

Opis testów:

- jak sprawdzano działanie robota,
- jakie były wyniki,
- co działa dobrze, a co wymaga poprawy.

8. Wnioski i możliwe usprawnienia

Zapisz:

- czego nauczyłeś się podczas projektu,
- jak można ulepszyć program w przyszłości.

1. Część praktyczna – Zadanie 1

1. Otwórz edytor tekstu LibreOfficeWriter. Zapisz plik pod nazwą „Dokumentacja oprogramowania robota wykrywającego przeszkody.
2. Dokumentację projektu rozpocznij od stworzenia strony tytułowej, która powinna zawierać następujące informacje: Tytuł projektu, Imiona i nazwiska autorów, Data wykonania, Szkoła.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Wpisz kolejno:

1. Tytuł projektu (styl: Nagłówek 1)

Robot wykrywający przeszkody i zatrzymujący się przed nimi (Styl: normal, pogrubiony)

Uwaga: Zastosuj te same style do pozostałych punktów.

2. Autor, Data wykonania, Szkoła (styl: Nagłówek 1)

Jan Nowak, 20.04.2026, (Szkoła Podstawowa nr 2 - klasa 7)

3. Cel projektu (styl: Nagłówek 1)

Celem programu jest zaprogramowanie robota, którego zadaniem będzie **zatrzymać się przed przeszkodą**. Dzięki czujnikowi odległości robot widzi, że coś stoi przed nim i nie wjeżdża w obiekt.

4. Wykorzystane elementy LEGO SPIKE (styl: Nagłówek 1)

- Hub SPIKE (mózg robota)
- 2 silniki (lewy i prawy)
- Czujnik odległości
- Klocki i elementy konstrukcyjne do budowy pojazdu

5. Opis działania robota (styl: Nagłówek 1)

Robot jedzie prosto do przodu. Jeśli czujnik odległości wykryje przeszkodę bliżej niż **15 cm**, robot:

1. zatrzymuje się,
2. czeka chwilę,
3. cofa się o kilka centymetrów.

Po cofaniu program się kończy. Dzięki temu robot nie wpada na przeszkodę.

6. Program – opis bloków (styl: Nagłówek 1)

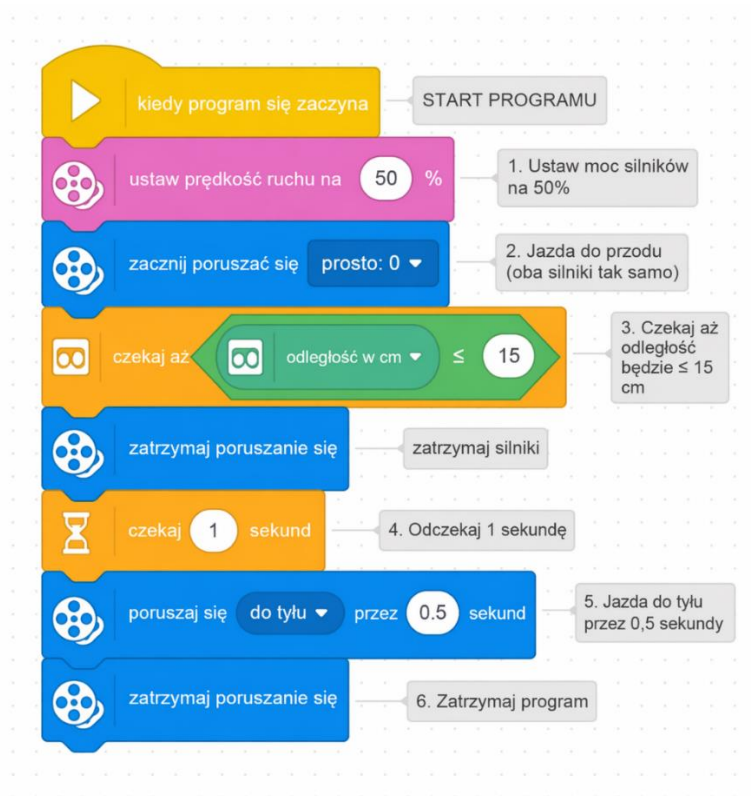
Przeanalizuj program. Program składa się z następujących kroków:

1. **Ustaw moc silników na 50%** — robot nie jedzie za szybko.
2. **Włącz jazdę do przodu** (oba silniki tak samo).

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

3. **Sprawdź czujnik odległości w pętli „zawsze”:**
 - o jeśli odległość > 15 cm $\rightarrow$ robot jedzie dalej,
 - o jeśli odległość ≤ 15 cm $\rightarrow$ zatrzymaj silniki.
4. **Odczekaj 1 sekundę.**
5. **Włącz jazdę do tyłu na 0,5 sekundy.**
6. **Zatrzymaj program.**

Prezentacja graficzna programu (Jeżeli program został już przygotowany uczniowie mogą wstawić w dokumentację zrzut ekranu z blokami).



Dokumentację programu można również przestawić w formie tabeli. Tabela pokazuje czytelny opis kroków, logikę i komentarze techniczne.

Tabela 1. Dokumentacja programu

Krok	Element programu	Opis działania	Warunek / logika	Komentarz techniczny
1	Ustawienie mocy silników	Silniki ustawiane są na 50% mocy	Brak	Ograniczenie prędkości zwiększa bezpieczeństwo i stabilność

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Krok	Element programu	Opis działania	Warunek / logika	Komentarz techniczny
				jazdy oraz poprawia reakcję na czujnik odległości.
2	Jazda do przodu	Robot porusza się do przodu, oba silniki pracują równomiernie	Brak	Równa praca silników zapewnia jazdę po linii prostej i stan bazowy programu.
3	Pętla „zawsze” (while forever)	Ciągłe sprawdzanie czujnika odległości	Warunek dynamiczny	Kluczowy element reaktywnego sterowania – robot działa w trybie ciągłego monitorowania otoczenia.
3a	Kontrola odległości	Jeśli odległość > 15 cm → jedź dalej	if distance > 15 cm	Robot może kontynuować jazdę bez przeszkód w pobliżu.
3b	Kontrola odległości	Jeśli odległość ≤ 15 cm → zatrzymaj silniki	if distance ≤ 15 cm	Zapobiega kolizji – robot natychmiast reaguje na przeszkodę.
4	Opóźnienie 1 sekundy	Program czeka 1 sekundę po zatrzymaniu	brak	Stabilizuje zachowanie robota i pozwala uniknąć natychmiastowych zmian stanu.
5	Ruch wstecz	Robot cofa się przez 0,5 sekundy	sterowanie czasowe	Mechanizm „wycofania się” od przeszkody zwiększa bezpieczeństwo i poprawia logikę unikania kolizji.
6	Zatrzymanie programu	Silniki zostają zatrzymane, program kończy działanie	brak	Zapewnia bezpieczne zakończenie pracy robota i brak dalszych niekontrolowanych ruchów.

7. Jak uruchomić robota?

1. Połącz Hub z komputerem lub tabletem.
2. Otwórz aplikację LEGO SPIKE.
3. Otwórz program o nazwie „ZatrzymajPrzedPrzeszkoda”.
4. Wgraj program do robota.
5. Postaw robota na równej powierzchni.
6. Możesz postawić przed nim pudełko lub książkę jako przeszkodę.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie dodatkowe do samodzielnego wykonania

Przykład 2 – Robot jadący po linii

Bazując na przykładzie 1 stwórz dokumentację oprogramowania zawierającą dokumentację w formie tabeli oraz formę graficzną. Program może dotyczyć robota jadącego po linii.

Dokumentacja programu robota LEGO SPIKE

1. Tytuł projektu

Robot LEGO SPIKE jadący po linii

2. Autor, Data wykonania, Szkoła

Jan Nowak, 20.04.2026, (Szkoła Podstawowa nr 2 - klasa 7)

3. Cel projektu

Celem projektu jest stworzenie programu, dzięki któremu robot LEGO SPIKE potrafi **jechać wzdłuż czarnej linii** na białej kartce lub macie. Robot wykorzystuje czujnik koloru, aby wykrywać, czy znajduje się na linii, i odpowiednio skręcać.

4. Wykorzystane elementy LEGO SPIKE

- Hub SPIKE
- 2 silniki (do napędu kół)
- 1 czujnik koloru
- Elementy konstrukcyjne LEGO (koła, osie, podstawa)

5. Opis działania robota

Robot ma jechać tak, aby **nie zjeżdżał z czarnej linii**. Program działa w następujący sposób:

1. Robot jedzie do przodu.
2. Czujnik koloru stale sprawdza, co widzi pod sobą.
3. Jeśli widzi **białe** – robot jedzie prosto.
4. Jeśli widzi **czarne** (czyli linię) – robot skręca lekko w jedną stronę.
5. Jeśli zjedzie za bardzo i widzi znów białe – skręca w drugą stronę, aby wrócić na linię.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

W ten sposób robot cały czas „poprawia” tor jazdy i trzyma się linii.

6. Program – opis bloków

Program składa się z takich kroków:

1. **Ustaw moc silników na 40%** (robot jedzie wtedy wolniej i dokładniej).
2. Włącz **pętlę „zawsze”**, aby robot stale jechał i reagował na linię.
3. W pętli:
 - o Jeśli **czujnik koloru wykrywa kolor biały**, uruchom oba silniki do przodu (jazda prosta).
 - o Jeśli **czujnik koloru widzi czarny**, to:
 - zwolnij jeden silnik lub zatrzymaj go,
 - drugi pozostaw w ruchu – robot wtedy skręca.
4. Pętla powtarza się cały czas, aż użytkownik zatrzyma program.

7. Jak uruchomić robota?

1. Podłącz Hub do komputera/tabletu i otwórz aplikację LEGO SPIKE.
2. Wgraj program „LineFollower”.
3. Umieść robota na **białej macie**, tak aby czujnik koloru znajdował się tuż nad **czarną linią**.
4. Naciśnij przycisk start na Hubie.
5. Obserwuj, czy robot poprawnie podąża za linią.

8. Testowanie robota

Robot został przetestowany na prostej linii oraz na zakrętach.
Wyniki testów:

- Na prostych odcinkach robot jechał stabilnie.
- Na zakrętach musiał częściej skręcać, ale wciąż trzymał się linii.
- Gdy jechał za szybko, zjeżdżał z trasy — dlatego moc silników ustawiono na 40%.

9. Wnioski

- Program działa poprawnie, a robot potrafi podążać za linią.
- Kluczowe jest odpowiednie umiejscowienie czujnika koloru blisko podłoża.
- Robot najlepiej działa przy mniejszej prędkości.
- W przyszłości można dodać:
 - o sygnał dźwiękowy przy skręcie,
 - o zatrzymanie po osiągnięciu mety,
 - o lepszy algorytm, który szybciej reaguje na zgubienie linii.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadania dodatkowe

1. Stwórz pełną dokumentację robota LEGO SPIKE, który reaguje na przeszkody, wydaje dźwięk i zmienia kierunek ruchu. Dokumentacja musi zawierać wszystkie elementy poznane na lekcji (cel projektu, opis działania, listę elementów, instrukcję uruchomienia, opis programu, tabelę programu, testy oraz wnioski), a także wyniki testów zapisane w arkuszu (np. Microsoft Excel lub Google Sheets) wraz z wykresem przedstawiającym zależność działania robota. Udostępnij dokumentację w środowisku chmurowym (np. Google Docs lub Microsoft Word Online), aby grupa mogła dokonać oceny, zgłosić uwagi i wprowadzić poprawki. Przeanalizuj możliwe zagrożenia (np. modyfikacja lub usunięcie pliku) i zaproponuj sposoby zabezpieczenia projektu.
2. Przeprowadź testowanie robota zgodnie z zasadami dokumentacji projektu. Zapisz wyniki testów w arkuszu kalkulacyjnym (np. Microsoft Excel lub Google Sheets), analizując zależność między odległością od przeszkody a reakcją robota. Stwórz wykres przedstawiający uzyskane dane. Porównaj różne ustawienia programu (np. różne wartości odległości zatrzymania). Sformułuj wnioski dotyczące działania robota i wskaż najlepsze parametry jego pracy.
3. Dokonaj analizy cyberzagrożeń w projekcie robota i jego dokumentacji. Określ, co może się stać w sytuacji, gdy ktoś zmodyfikuje dokumentację, usunie plik lub wprowadzi złośliwy kod. Zaproponuj sposoby zabezpieczenia projektu, takie jak tworzenie kopii zapasowych, kontrola dostępu oraz wersjonowanie dokumentów. Na tej podstawie opracuj listę zagrożeń oraz odpowiadających im metod ochrony, które zwiększą bezpieczeństwo pracy nad projektem.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 4: Biurowa moc: TechOffice

Temat lekcji nr 14 - Automatyczne generatory spisów ułatwiają pracę

Treści nauczania (wymagania szczegółowe - podstawa programowa) – II. 3) b);

Cel lekcji

Celem lekcji jest praktyczne wykorzystanie automatycznych generatorów spisów w dokumentacji oprogramowania. Uczniowie poznają automatyczny spis treści, spis tabel i spis rysunków jako narzędzia edytora tekstu, które pomagają porządkować dłuższe dokumenty, ułatwiają nawigację oraz ograniczają błędy wynikające z ręcznego wpisywania numerów stron.

Podczas zajęć uczniowie uczą się stosować style nagłówków, tworzyć co najmniej dwa poziomy struktury dokumentu, generować spis treści, dodawać podpisy do tabel i rysunków, tworzyć wykazy tabel oraz rysunków, a następnie aktualizować wszystkie spisy po zmianach w treści. Lekcja rozwija umiejętność przygotowywania czytelnej, profesjonalnej dokumentacji technicznej i projektowej.

Cele szczegółowe lekcji

Uczeń:

- wyjaśnia, czym jest automatyczny spis treści,
- wyjaśnia, czym jest automatyczny spis tabel,
- wyjaśnia, czym jest automatyczny spis rysunków,
- rozumie, że automatyczny spis treści powstaje na podstawie stylów nagłówków,
- stosuje style „Nagłówek 1” i „Nagłówek 2” w dokumencie,
- tworzy logiczną strukturę dokumentacji programu,
- generuje automatyczny spis treści,
- ustawia poziomy nagłówek widoczny w spisie treści,
- sprawdza poprawność numerów stron w spisie,
- aktualizuje spis treści po zmianie nazw rozdziałów lub układu dokumentu,
- dodaje podpis do tabeli,
- wybiera właściwą kategorię podpisu, np. „Tabela”,
- tworzy automatyczny spis tabel,
- dodaje podpis do rysunku,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- wybiera właściwą kategorię podpisu, np. „Rysunek”,
- tworzy automatyczny spis rysunków,
- dodaje podziały stron dla głównych rozdziałów,
- wstawia numerację stron w stopce,
- sprawdza działanie odsyłaczy w spisie treści,
- aktualizuje spis treści, spis tabel i spis rysunków po zmianach w dokumencie,
- zapisuje uporządkowaną dokumentację w pliku lokalnym lub w chmurze.

Treści nauczania – zgodność z podstawą programową

Lekcja realizuje wymagania podstawy programowej poprzez praktyczne tworzenie i porządkowanie informacji w edytorze tekstu. Uczeń przygotowuje dokumentację programu robota LEGO SPIKE, stosuje style nagłówków, organizuje treść w rozdziały i podrozdziały oraz tworzy automatyczne spisy, które ułatwiają korzystanie z dokumentu.

Zajęcia rozwijają umiejętność posługiwania się edytorem tekstu do przygotowywania złożonych dokumentów. Uczniowie porządkują dokumentację oprogramowania, dodają spis treści, spis tabel i spis rysunków, a także aktualizują je po wprowadzeniu zmian. W materiale lekcyjnym podkreślono, że automatyczne spisy oszczędzają czas, eliminują błędy ręcznego tworzenia oraz podnoszą jakość i profesjonalizm dokumentacji.

Sposób realizacji wymagań w innowacyjnym programie nauczania

Wymagania programowe będą realizowane poprzez pracę uczniów w edytorze tekstu, np. LibreOffice Writer, Microsoft Word lub Dokumenty Google z uwzględnieniem funkcji dostępnych w danym narzędziu. Uczniowie będą pracować na dokumentacji przygotowanej na poprzedniej lekcji lub na dostarczonym dokumencie przykładowym, a następnie uporządkują go zgodnie z zasadami dokumentacji technicznej.

Lekcja łączy umiejętności cyfrowe z praktyką dokumentowania projektu programistyczno-robotycznego. Uczniowie widzą, że automatyczne spisy mają zastosowanie w pracach szkolnych, raportach technicznych, instrukcjach, dokumentacji oprogramowania i projektach LEGO SPIKE. Dzięki temu uczą się tworzyć dokumenty, które są nie tylko estetyczne, ale także łatwe w aktualizacji i wygodne dla użytkownika.

Metody pracy

- pokaz multimedialny lub demonstracja nauczyciela,
- pogadanka kierowana,
- analiza przykładowej dokumentacji programu,
- ćwiczenia praktyczne przy komputerze,
- praca z dokumentem przygotowanym na poprzednich zajęciach,
- praca z instrukcją krok po kroku,
- samodzielne generowanie i aktualizowanie spisów,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- sprawdzanie działania odsyłaczy i numerów stron,
- korekta struktury dokumentu.

Środki dydaktyczne

- komputer uczniowski z edytorem tekstu,
- edytor tekstu, np. LibreOffice Writer, Microsoft Word lub Dokumenty Google,
- dokumentacja programu robota LEGO SPIKE przygotowana na poprzedniej lekcji,
- przykładowe tabele do dokumentacji, np. „Skład grupy projektowej” i „Wykaz elementów robota”,
- przykładowe rysunki lub schematy, np. schemat blokowy programu,
- karta pracy lub e-podręcznik,
- rzutnik multimedialny lub ekran interaktywny,
- dostęp do zapisu plików lokalnie lub w chmurze.

Formy pracy

- praca indywidualna przy komputerze,
- praca pod kierunkiem nauczyciela,
- ćwiczenia samodzielne,
- elementy pracy projektowej,
- elementy pracy analitycznej podczas porządkowania struktury dokumentacji,
- krótkie sprawdzenie efektów pracy ucznia.

Faza I. Wprowadzenie

Nauczyciel rozpoczyna lekcję od rozmowy z uczniami o tym, co dzieje się z dokumentem, gdy staje się dłuższy i zawiera wiele rozdziałów, tabel oraz rysunków. Uczniowie wskazują, że trudniej wtedy szybko odnaleźć potrzebne informacje. Nauczyciel wyjaśnia, że właśnie dlatego w dokumentach technicznych i projektowych stosuje się automatyczne spisy.

Następnie nauczyciel omawia trzy typy spisów: spis treści, spis tabel i spis rysunków. Wyjaśnia, że spis treści powstaje na podstawie stylów nagłówek, natomiast spis tabel i spis rysunków powstają na podstawie podpisów dodanych do odpowiednich obiektów. Podkreśla, że automatyczne spisy można aktualizować po zmianie treści, numeracji stron lub kolejności elementów.

Wprowadzenie kończy się przedstawieniem zadania głównego: uczniowie uporządkują dokumentację programu robota LEGO SPIKE, dodając nagłówki, automatyczny spis treści, podpisy tabel i rysunków, spis tabel, spis rysunków oraz numerację stron.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Faza II. Część zasadnicza – realizacja

W części praktycznej uczniowie otwierają dokumentację oprogramowania przygotowaną na poprzedniej lekcji. W pierwszym kroku sprawdzają, czy główne części dokumentu mają zastosowany styl „Nagłówek 1”. Następnie modyfikują wybrane punkty dokumentacji, ustawiając je jako „Nagłówek 2”, aby dokument miał co najmniej dwa poziomy struktury.

Kolejnym zadaniem jest wygenerowanie automatycznego spisu treści. Uczniowie wybierają funkcję „Spis treści, indeks lub bibliografia”, ustawiają typ „Spis treści” oraz zakres obejmujący dokument do poziomu 2. Po wstawieniu spisu sprawdzają, czy wszystkie rozdziały i podrozdziały są widoczne oraz czy numery stron są poprawne.

Następnie uczniowie modyfikują nazwę jednego z rozdziałów, np. zmieniają ostatni rozdział na „Literatura”. Po tej zmianie aktualizują spis treści, klikając go prawym przyciskiem myszy i wybierając opcję aktualizacji. Dzięki temu widzą, że spis nie powinien być poprawiany ręcznie, lecz odświeżany za pomocą funkcji edytora.

W dalszej części uczniowie pracują ze spisem tabel. Dodają podpis do istniejącej tabeli dokumentacyjnej, wybierając kategorię „Tabela”. Następnie tworzą drugą tabelę, np. „Skład grupy projektowej”, formatują ją, dodają krawędzie, pogrubiają nagłówki oraz nadają podpis. Po dodaniu podpisów generują automatyczny spis tabel i umieszczają go na końcu dokumentu.

Kolejny etap dotyczy rysunków. Uczniowie dodają podpis do istniejącego rysunku, wybierając kategorię „Rysunek”. Następnie wstawiają drugi rysunek, np. schemat blokowy programu, i również dodają do niego podpis. Po wykonaniu tych czynności tworzą automatyczny spis rysunków, dbając o wybór właściwej kategorii podpisów.

W zadaniu samodzielnym uczniowie dodają kolejną tabelę, np. „Wykaz elementów robota”, uzupełniają ją danymi i nadają jej podpis. Następnie wstawiają podziały stron dla każdego głównego rozdziału, dodają nowe rozdziały „Indeks tabel” i „Wykaz rysunków”, wstawiają numerację stron w stopce oraz aktualizują spis treści, spis tabel i spis rysunków. Na końcu sprawdzają, czy numery stron i odsyłacze działają poprawnie.

Faza III. Podsumowanie

Na zakończenie lekcji uczniowie prezentują uporządkowane dokumenty. Nauczyciel sprawdza, czy dokument zawiera co najmniej dwa poziomy nagłówków, automatyczny spis treści, podpisy tabel i rysunków, spis tabel, spis rysunków oraz numerację stron.

Uczniowie odpowiadają na pytania kontrolne dotyczące tego, czym jest automatyczny spis treści, kiedy warto używać automatycznego spisu tabel oraz co należy zrobić, aby spis treści działał poprawnie. Nauczyciel podkreśla, że automatyczne generatory spisów są ważnym narzędziem w pracy z dokumentacją, ponieważ poprawiają czytelność, oszczędzają czas i ułatwiają aktualizację dokumentu po zmianach.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Kryteria oceniania

Ocena dopuszczająca

Uczeń z pomocą nauczyciela stosuje styl nagłówka w dokumencie i wstawia prosty spis treści.

Ocena dostateczna

Uczeń samodzielnie stosuje style „Nagłówek 1” i „Nagłówek 2”, tworzy automatyczny spis treści oraz aktualizuje go po zmianach w dokumencie.

Ocena dobra

Uczeń poprawnie dodaje podpisy do tabel i rysunków, tworzy spis tabel oraz spis rysunków, wstawia numerację stron i aktualizuje wszystkie spisy.

Ocena bardzo dobra

Uczeń samodzielnie i świadomie porządkuje całą dokumentację, stosuje wielopoziomą strukturę nagłówków, generuje i aktualizuje spis treści, spis tabel oraz spis rysunków, sprawdza poprawność odsyłaczy i potrafi wyjaśnić znaczenie automatycznych spisów w dokumentacji technicznej.

Krótki opis problemu do rozwiązania

Uczniowie mają uporządkować dokumentację programu robota LEGO SPIKE tak, aby była wygodna w czytaniu i łatwa w aktualizacji. Problem polega na tym, aby nie tworzyć spisów ręcznie, lecz zastosować style nagłówków, podpisy tabel i rysunków oraz automatyczne generatory spisów. Dzięki temu dokument staje się bardziej profesjonalny, a po zmianie rozdziałów, tabel, rysunków lub numerów stron można go szybko zaktualizować bez ręcznego poprawiania całego spisu.

Faza IV. Quiz/Test sprawdzający

1. Czym jest automatyczny spis treści?

A. Lista obrazków w dokumencie

B. Ręcznie wpisana lista rozdziałów

C. Automatycznie generowana lista nagłówków dokumentu

2. Kiedy warto używać automatycznego spisu tabel?

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

A. Gdy dokument zawiera dużo tabel

B. Gdy dokument ma tylko tekst

C. Tylko w prezentacjach multimedialnych

3. Co należy zrobić, aby spis treści działał poprawnie?

A. Wpisać ręcznie wszystkie tytuły

B. Użyć stylów nagłówków (np. Nagłówek 1, 2)

C. Zmienić kolor tekstu

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 14 - Automatyczne generatory spisów ułatwiają pracę

Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
- Przedstawienie uczniom celu i zakresu tematycznego lekcji.
- Uruchomienie na e-platformie materiałów do lekcji
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów.
- Zadanie pracy domowej (opcjonalnie).

Załącznik nr 2 – Konspekt nauczyciela lekcji nr 14 - Automatyczne generatory spisów ułatwiają pracę

Cel lekcji:

Celem lekcji jest praktyczne wykorzystanie generatorów spisów w dokumentacji oprogramowania.

Uczeń:

- zrozumie, czym jest automatyczny spis treści,
- potrafi zastosować style nagłówków w dokumencie,
- generuje automatyczny spis treści i spis tabel,
- wykorzystuje spis do uporządkowania dokumentacji programu,
- widzi praktyczne zastosowanie w pracy programisty.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przebieg lekcji:

Wprowadzenie

Automatyczny spis treści to funkcja edytora tekstu (np. Microsoft Word, Google Docs lub LibreOffice Writer), która **samodzielnie tworzy listę nagłówków dokumentu wraz z numerami stron** na podstawie zastosowanych stylów (Nagłówek 1, 2, 3 itd.).

Spis ten można w każdej chwili **zaktualizować**, dzięki czemu odzwierciedla aktualną strukturę dokumentu.

Automatyczny spis tabel to zestawienie wszystkich tabel w dokumencie, tworzone automatycznie na podstawie **podpisów (etykiet) tabel**, np. „Tabela 1”, „Tabela 2”.

Spis może zawierać nie tylko numer tabeli, tytuł oraz numer strony. Podobnie jak spis treści, może być automatycznie aktualizowany.

Automatyczne spisy znajdują zastosowanie w przygotowywaniu dokumentacji projektów (np. programów LEGO SPIKE), pracach szkolnych (referaty, projekty), raportach technicznych, analizach, sprawozdaniach czy instrukcjach i dokumentacjach oprogramowania.

Dzięki nim możliwe jest szybkie odnalezienie informacji, nawigacja po dokumencie, zachowanie porządku i profesjonalnego wyglądu.

Warto jest stosować gdy dokument ma **więcej niż kilka stron**, zawiera **wiele rozdziałów lub sekcji**, gdy chcemy aby dokument był **czytelny i uporządkowany** oraz w każdej **oficjalnej dokumentacji** (np. projekt, instrukcja).

Automatyczne spisy to narzędzia, które:

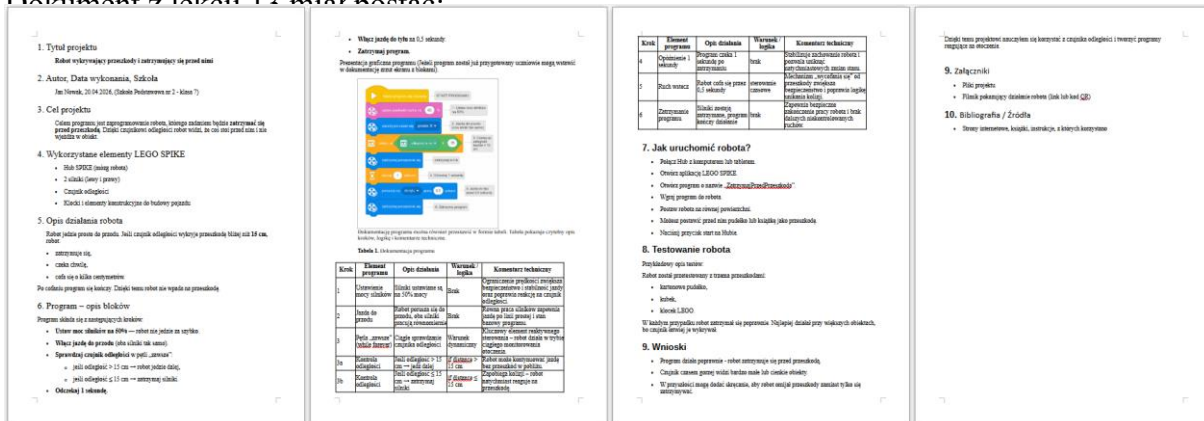
- oszczędzają czas,
- eliminują błędy ręcznego tworzenia,
- podnoszą jakość i profesjonalizm dokumentacji.
- są standardem w pracy ucznia, studenta i programisty.

Część praktyczna.

1. **Korzystając z przygotowanej na poprzedniej lekcji dokumentacji oprogramowania sformatuj ponownie dokument zgodnie z wytycznymi poniżej.**

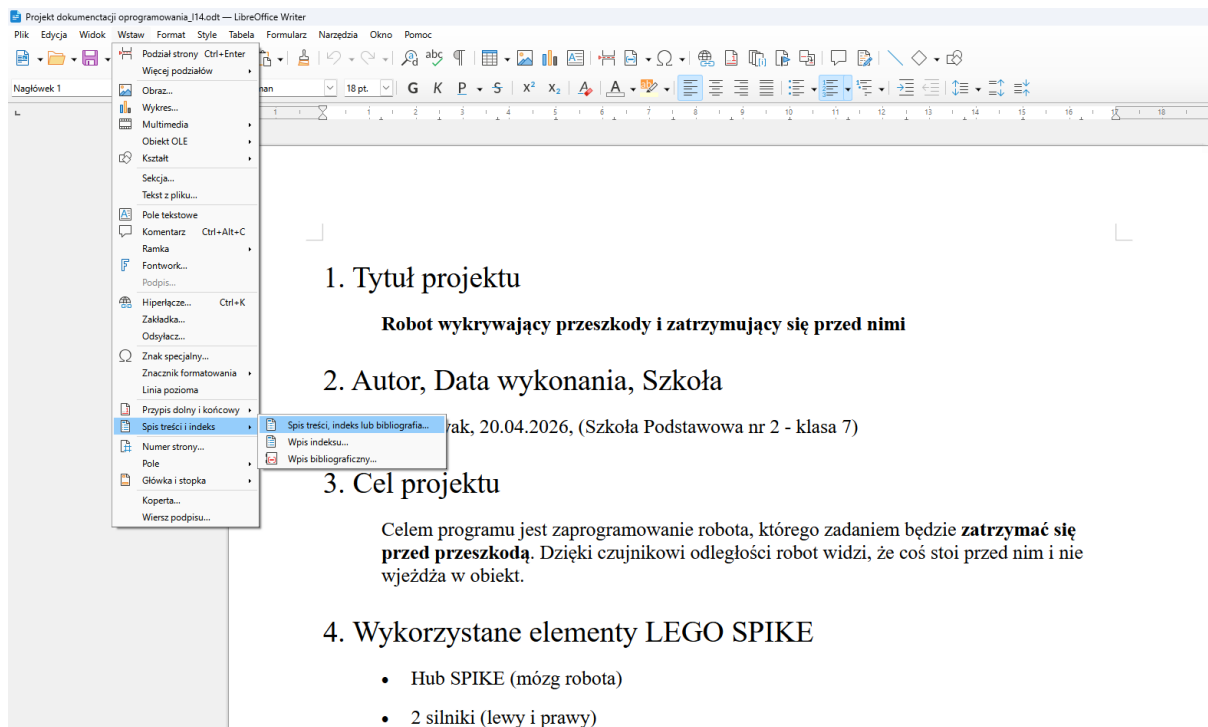
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Dokument z lekcji 12 miał nastąpić:



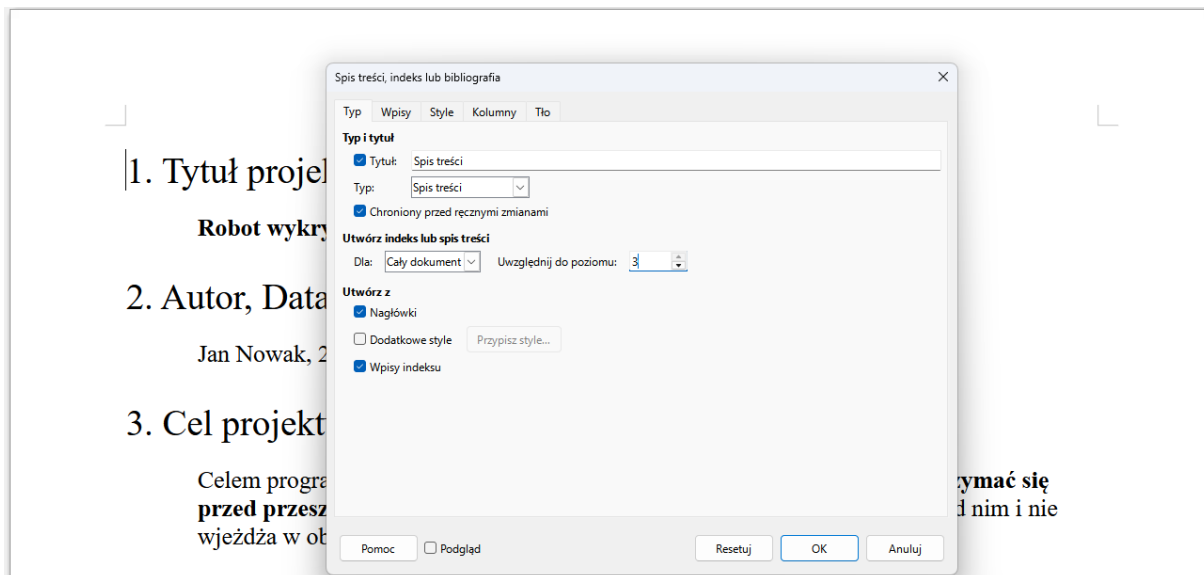
2. Wygeneruj automatyczny spis treści.

- Wybierz zakładkę „Spis treści, indeks lub bibliografia” znajdujący się na pasku głównym menu: „Wstaw” -> „Spis treści i indeks”.

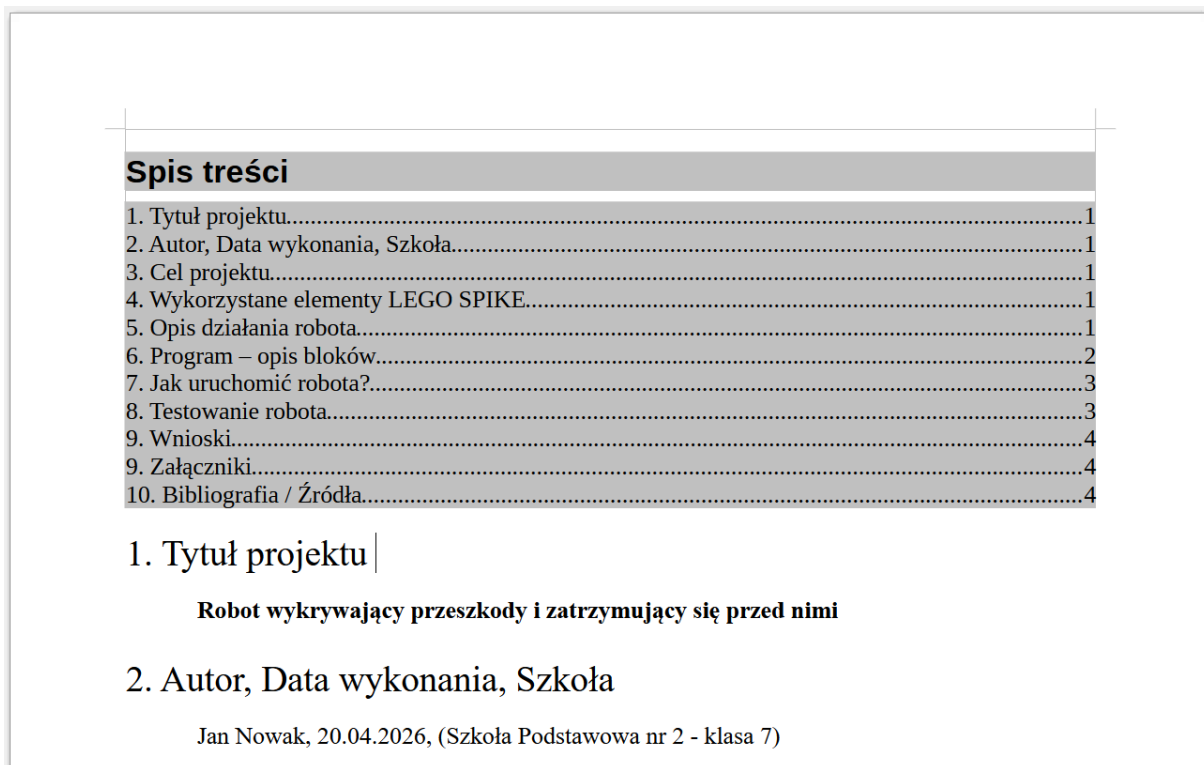


Wybierz odpowiedni typ i tytuł spisu – ustaw typ: spis treści i wybierz opcję, aby spis objął cały dokument, uwzględniając spis do poziomu 2. Zatwierdź zmiany klikając na przycisk OK.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

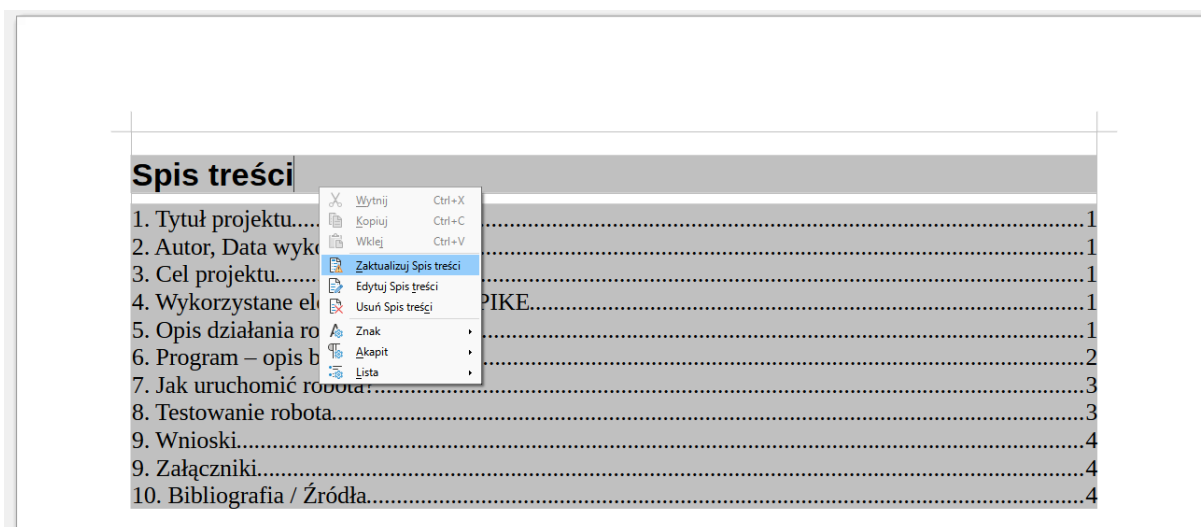


2. Sprawdź czy poprawnie dodano spis i czy numeracja stron jest poprawna.



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

3. Zmodyfikuj nazwę ostatniego rozdziału na Literatura i zaktualizuj spis (najedź kursorem myszy na spis i kliknij prawym przyciskiem myszy – pojawi się menu. Wybierz opcję *Zaktualizuj spis treści*).



Po zmianie spisu treści powinien mieć postać:

Spis treści	
1. Tytuł projektu.....	1
2. Autor, Data wykonania, Szkoła.....	1
3. Cel projektu.....	1
4. Wykorzystane elementy LEGO SPIKE.....	1
5. Opis działania robota.....	1
6. Program – opis bloków.....	2
7. Jak uruchomić robota?.....	3
8. Testowanie robota.....	3
9. Wnioski.....	4
9. Załączniki.....	4
10. Literatura.....	4

4. Uporządkuj dokument tak, aby zawierał:

- co najmniej 2 poziomy nagłówków,
- poprawny spis treści,
- logiczny podział programu.
- spis tabel i spis rysunków

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

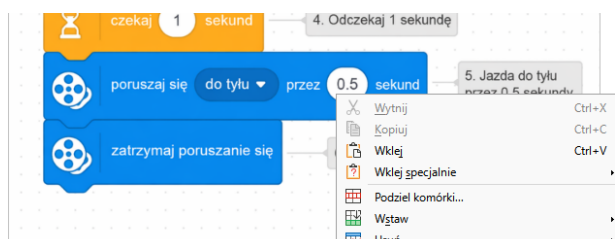
Aby dodać 2 poziom nagłówków zmodyfikuj wybrane punkty dokumentacji na styl: nagłówek 2. Proponowana nowa struktura dokumentu:

Spis treści

1. Tytuł projektu.....	1
2. Opis projektu.....	1
a) Autor, Data wykonania, Szkoła.....	1
b) Cel projektu.....	1
c) Wykorzystane elementy LEGO SPIKE.....	1
d) Opis działania robota.....	1
3. Program.....	2
a) Opis programu.....	2
b) Prezentacja graficzna programu.....	2
4. Podsumowanie.....	3
a) Jak uruchomić program.....	3
b) Testowanie robota.....	4
c) Wnioski.....	4
5. Załączniki.....	4
6. Literatura.....	4

5. Wstaw podpis tabeli i dodaj do dokumentu spis tabel.

Usuń poprzednią nazwę tabeli. Postaw kursor w dowolnym miejscu w tabeli i naciśnij lewy przycisk myszki. Wybierz opcję *Wstaw podpis*.



Dokumentację programu można również przedstawić w formie tabeli, która pokazuje czytelny opis kroków, logikę i komentarze techniczne.

Tabela 1: Dokumentacja programu

Krok	Element programu	Opis działania	Komentarz techniczny
1	Ustawienie mocy silników	Silniki ustawiane na 50% mocy	Brak
		Robot porusza się do	Wzrost prędkości zwiększa bezpieczeństwo i stabilność jazdy oraz poprawia reakcję na czujnik odległości.
			Równa praca silników zapewnia

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Dodaj drugą tabelę w rozdziale 2.a o nazwie: Skład grupy projektowej. Usuń poprzednie wpisy. Sformatuj tabelę dodając krawędzie i pogrubiając nagłówki. Dodaj nazwę tabeli – Skład grupy projektowej. Przykładowa tabela może mieć postać:

2. Opis projektu

a) Autor, Data wykonania, Szkoła

Tabela 1: Skład grupy projektowej

Grupa projektowa	Skład grupy
Grupa 1	Tomek Adam Jan

Dodaj na końcu dokumentu spis (indeks) tabel:

5. Załączniki

- Spis tabel
- Filmik pokazujący działanie robota (link lub kod QR)

6. Literatura

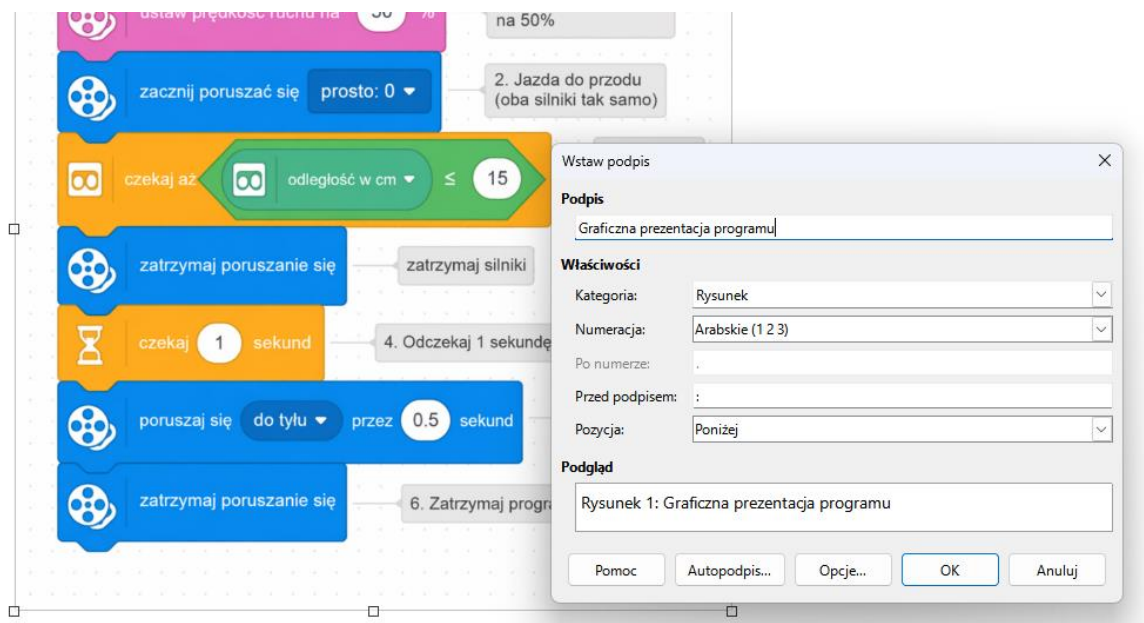
- Strony internetowe, książki, instrukcje, z których korzystano

Indeks tabel

Tabela 1: Skład grupy projektowej.....	1
Tabela 2: Dokumentacja programu.....	4

6. Wstaw podpis rysunku i dodaj do dokumentu spis rysunków.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

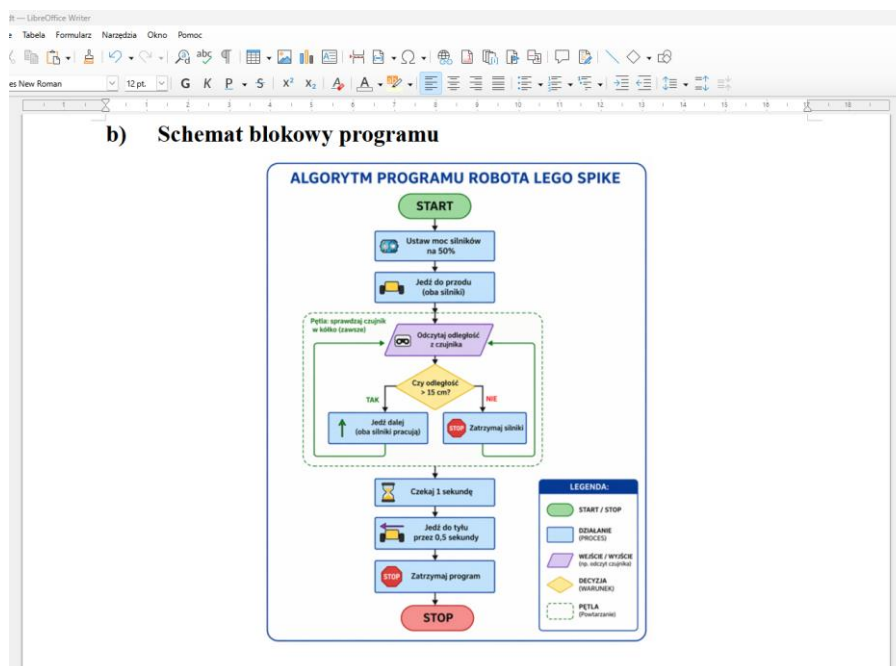


Po zaakceptowaniu pojawi się podpis rysunku:



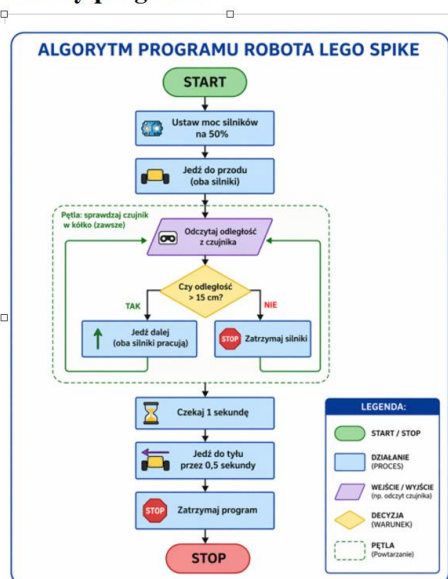
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

7. Dodaj drugi rysunek - schemat blokowy programu (lokalizację pliku wskaże nauczyciel). Rysunek drugi wstaw po punkcie 3.a dokumentacji.



Wstaw podpis rysunku.

b) Schemat blokowy programu



Rysunek 1: Schemat blokowy programu

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

8. Zaktualizuj spis treści, spis rysunków i spis tabel. Spis rysunków i spis tabel umieść na końcu dokumentu. Zwróć uwagę aby poprawnie wybrać typ spisu i kategorię podpisów (rysunek).

6. Literatura

- Strony internetowe

Indeks tabel

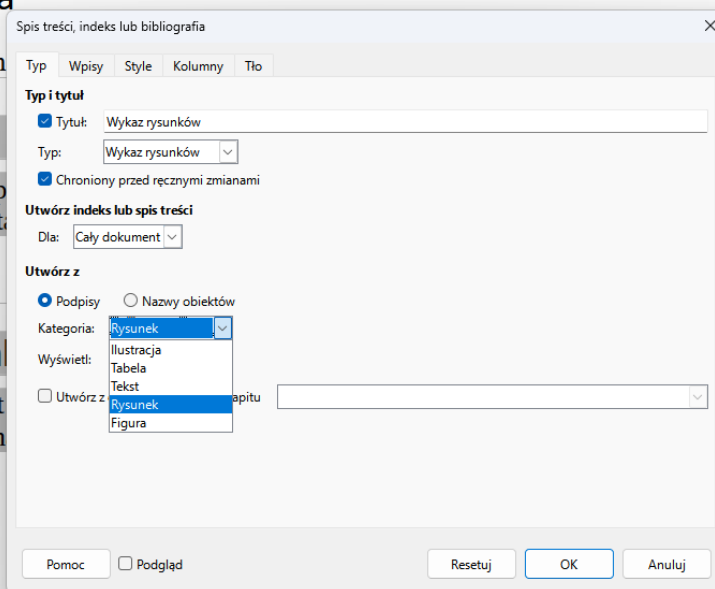
Tabela 1: Skład grupy projektowej.....1

Tabela 2: Dokumentacja programu.....4

Wykaz rysunków

Rysunek 1: Schemat blokowy programu.....3

Rysunek 2: Graficzna prezentacja programu.....4



9. Przykładowy wynik pokazano na rysunku:

Indeks tabel

Tabela 1: Skład grupy projektowej.....1

Tabela 2: Dokumentacja programu.....4

Wykaz rysunków

Rysunek 1: Schemat blokowy programu.....3

Rysunek 2: Graficzna prezentacja programu.....4

Zaktualizuj spis treści. Sprawdź czy numeracja stron uległa zmianie i jest poprawna.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Spis treści

1. Tytuł projektu.....	1
2. Opis projektu.....	1
a) Autor, Data wykonania, Szkoła.....	1
b) Cel projektu.....	1
c) Wykorzystane elementy LEGO SPIKE.....	1
d) Opis działania robota.....	2
3. Program.....	2
a) Opis programu.....	2
b) Schemat blokowy programu.....	3
c) Prezentacja graficzna programu.....	3
4. Podsumowanie.....	5
a) Jak uruchomić robota?.....	5
b) Testowanie robota.....	5
c) Wnioski.....	5
5. Załączniki.....	6
6. Literatura.....	6

Zadania do samodzielnego wykonania

1. Dodaj kolejną tabelę w rozdziale 2. C. Wstaw podpis tabeli: Wykaz elementów robota. Przykładowa tabela może mieć postać:

Tabela 2: Wykaz elementów robota

Lp.	Nazwa elementu	Liczba sztuk
1		
2		
3		
4		

2. Uzupełnij danymi tabelę.

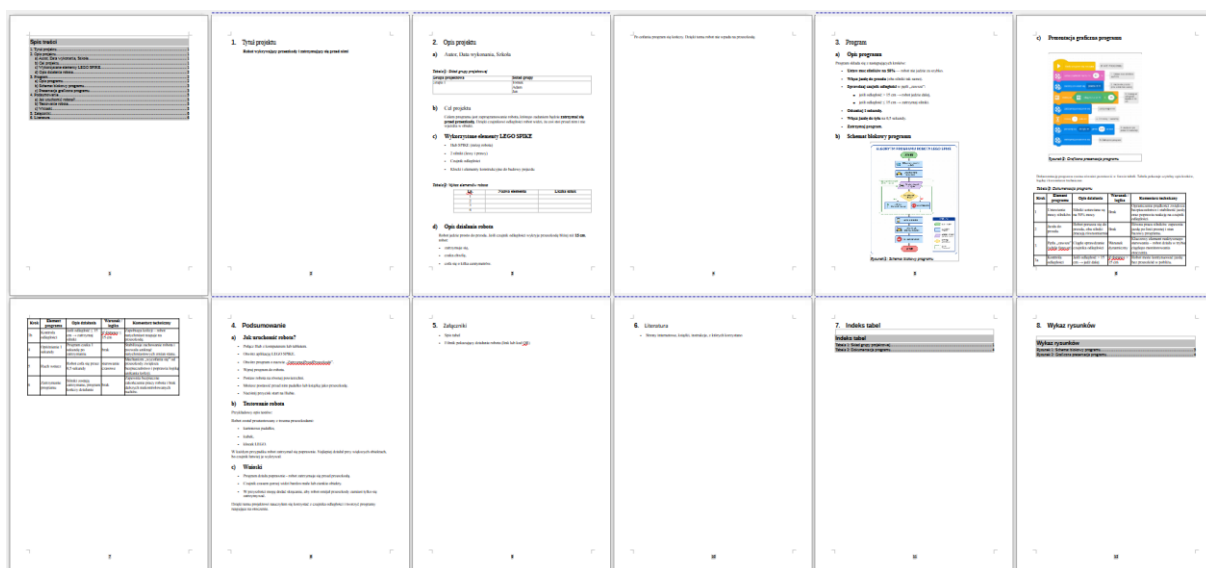
3. Wstaw znak podziału strony dla każdego rozdziału – nagłówek 1.

1. Tytuł projektu

Robot wykrywający przeszkody i zatrzymujący się przed nimi

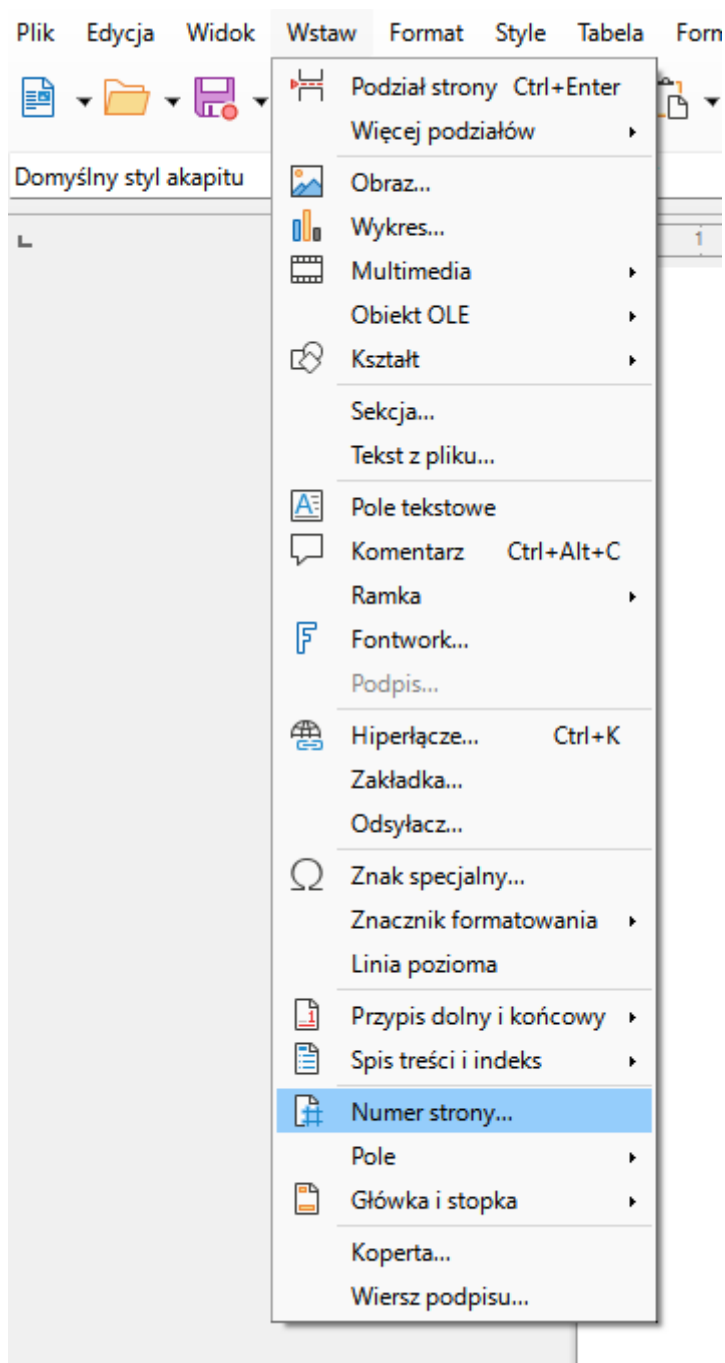
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Każdy rozdział powinien być na nowej stronie. Dodaj dwa nowe rozdziały (styl: Nagłówek 1): Indeks tabel i Wykaz rysunków. Sprawdź czy układ dokumentu jest podobny do tego na rysunku poniżej:

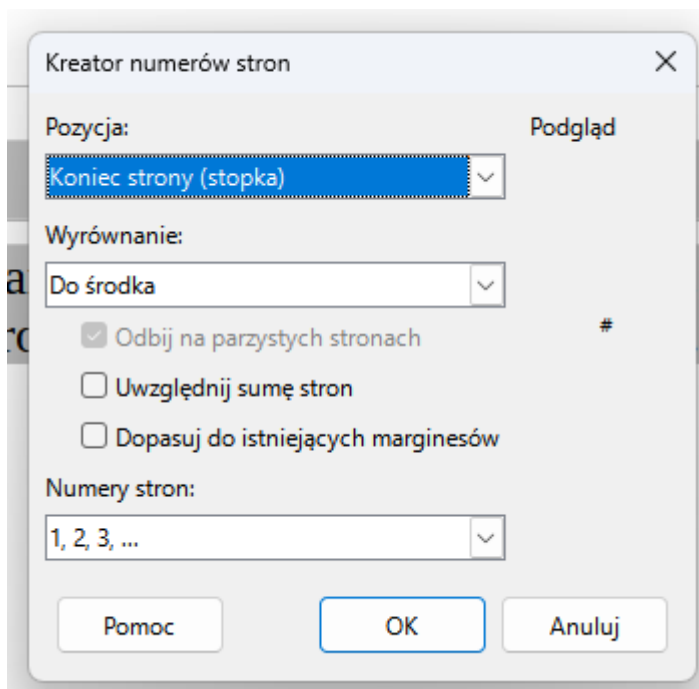


4. Wstaw numerację stron w stopce (z paska wybierz wstaw i numer strony).

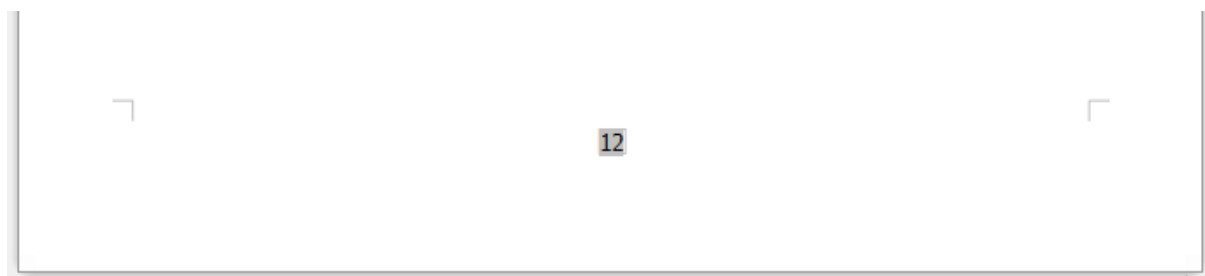
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Poprawny numer strony powinien mieć postać:



5. Zaktualizuj spis treści, indeks tabel i wykaz rysunków.

Zwróć uwagę, że przed aktualizacją spis może wyglądać następująco:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Spis treści	
1. Tytuł projektu.....	1
2. Opis projektu.....	1
a) Autor, Data wydania.....	1
b) Cel projektu.....	1
c) Wykorzystane elementy.....	1
d) Opis działania.....	2
3. Program.....	2
a) Opis programu.....	2
b) Schemat blokowy.....	3
c) Prezentacja graficzna.....	3
4. Podsumowanie.....	5
a) Jak uruchomić.....	5
b) Testowanie.....	5
c) Wnioski.....	5
5. Załączniki.....	6
6. Literatura.....	6

Po aktualizacji powinien przyjąć postać:

Spis treści	
1. Tytuł projektu.....	2
2. Opis projektu.....	3
a) Autor, Data wykonania, Szkoła.....	3
b) Cel projektu.....	3
c) Wykorzystane elementy LEGO SPIKE.....	3
d) Opis działania robota.....	3
3. Program.....	5
a) Opis programu.....	5
b) Schemat blokowy programu.....	5
c) Prezentacja graficzna programu.....	6
4. Podsumowanie.....	8
a) Jak uruchomić robota?.....	8
b) Testowanie robota.....	8
c) Wnioski.....	8
5. Załączniki.....	9
6. Literatura.....	10
7. Indeks tabel.....	11
8. Wykaz rysunków.....	12

6. Sprawdź czy numery stron są poprawne dla wybranych tabel i rozdziałów/podrozdziałów. Trzymając klawisz CTRL kliknij na wybrany wiersz i sprawdź czy zostaniesz przeniesiony na wybraną stronę.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Spis treści

1. Tytuł projektu.....	2
2. Opis projektu.....	3
a) Autor, Data wykonania, Szkoła.....	3
b) Cel projektu.....	3
c) Wykorzystane elementy LEGO SPIKE.....	3
d) Opis działania robota.....	3
3. Program.....	5
a) Opis programu.....	5
b) Schemat blokowy programu.....	5
c) Prezentacja graficzna programu.....	6
4. Podsumowanie.....	8
a) Jak uruchomić robota?.....	8
b) Testowanie robota.....	8
c) Wnioski.....	8
5. Załączniki.....	9
6. Literatura.....	10
7. Indeks tabel.....	11
8. Wykaz rysunków.....	12

Ctrl-kliknij, aby otworzyć hiperłącze: 8. Wykaz rysunków

Podsumowanie wykonania zadań:

Automatyczne generatory spisów treści znacząco ułatwiają pracę z dokumentami, szczególnie w przypadku dłuższych i bardziej złożonych opracowań, takich jak dokumentacja oprogramowania. Dzięki zastosowaniu stylów nagłówków możliwe jest szybkie i przejrzyste uporządkowanie treści oraz automatyczne utworzenie spisu treści, który pozwala łatwo odnaleźć potrzebne informacje.

Podczas ćwiczenia uczniowie przekonali się, że automatyczne spisy oszczędzają czas, eliminują błędy wynikające z ręcznego tworzenia oraz umożliwiają szybkie aktualizowanie dokumentu po wprowadzeniu zmian. Dodatkowo poprawiają czytelność i profesjonalny wygląd dokumentacji.

Wnioskiem z zajęć jest to, że umiejętność korzystania z automatycznych generatorów spisów jest bardzo przydatna w pracy z dokumentami technicznymi i stanowi ważny element kompetencji cyfrowych, szczególnie w obszarze tworzenia dokumentacji projektów i oprogramowania.

Zadania dodatkowe

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

1. Utwórz dokument pt. „Dokumentacja programu robota LEGO SPIKE” i podziel tekst programu na minimum 4 sekcje (np. Wstęp, Działanie, Algorytm, Podsumowanie), stosując nagłówki (Nagłówek 1). Pracując w parach lub grupach, podzielcie się sekcjami dokumentu, a następnie zapisz dokument w środowisku chmurowym, nadając odpowiednie uprawnienia dostępu (tylko do odczytu / edycji) i dbając o bezpieczeństwo danych (brak danych osobowych poza imieniem i inicjałem nazwiska).
2. Na podstawie dokumentu z zadania 1 wstaw automatyczny spis treści i sprawdź, czy wszystkie sekcje są w nim widoczne. Następnie udostępnij dokument członkom zespołu i poproś ich o sprawdzenie poprawności struktury, zwracając uwagę na możliwość przypadkowej edycji lub usunięcia treści oraz zaproponuj zabezpieczenia (np. wersjonowanie dokumentu, komentarze zamiast edycji).
3. Rozwiń dokument o szczegółowy opis kroków programu. Każdy krok (1–6) ustaw jako Nagłówek 2 i dodaj krótkie wyjaśnienie. Wykonaj pracę zespołową, gdzie każdy członek grupy odpowiada za inny krok programu, a następnie zaktualizuj spis treści i sprawdź historię zmian dokumentu, analizując kto i jakie modyfikacje wprowadził.
4. Dokonaj aktualizacji projektu, dodając nową sekcję „Możliwe ulepszenia robota”, w której wpiszesz minimum 2 propozycje ulepszeń. Przed dodaniem treści sprawdź, czy dokument nie został zmodyfikowany przez inne osoby bez Twojej wiedzy, a następnie zaktualizuj spis treści i zapisz nową wersję dokumentu (np. v2), stosując podstawowe zasady wersjonowania.
5. Włącz automatyczną numerację nagłówków (np. 1, 1.1, 1.2) i upewnij się, że numeracja pojawia się także w spisie treści. Porównaj dokument z wcześniejszą wersją, sprawdzając czy nie wystąpiły błędy lub utrata danych, a następnie zabezpiecz dokument poprzez zapis kopii zapasowej.
6. Stwórz kompletną dokumentację zawierającą: stronę tytułową, spis treści, opis programu, sekcję „Testowanie robota” oraz „Wnioski”. Wykonaj testy robota, zapisz wyniki w arkuszu kalkulacyjnym w chmurze, przygotuj wykres i udostępnij dokument zespołowi do oceny, zbierając komentarze oraz propozycje poprawek.
7. Stwórz nowy program i dokumentację. Zaprojektuj własną wersję robota (np. omijanie przeszkód), opisz ją w dokumencie, zastosuj nagłówki i wygeneruj spis treści. Pracuj w grupie, gdzie każdy członek odpowiada za inną część projektu (np. algorytm, testy, dokumentacja), a następnie przeanalizuj potencjalne zagrożenia (np. utrata pliku, nieautoryzowana edycja) i zaproponuj sposoby zabezpieczenia projektu.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 4: Biurowa moc: TechOffice

Temat lekcji nr 15 - Tabele i funkcje porządkujemy dane

Treści nauczania (wymagania szczegółowe - podstawa programowa) – II. 3) c);

Cel lekcji

Celem lekcji jest praktyczne wykorzystanie tabel i funkcji arkusza kalkulacyjnego do porządkowania, analizowania i interpretowania danych liczbowych oraz tekstowych. Uczniowie uczą się, jak na podstawie danych z testów robota LEGO SPIKE sprawdzić, czy robot działa zgodnie z przyjętym założeniem: powinien zatrzymać się, gdy odległość od przeszkody wynosi 15 cm lub mniej.

Podczas zajęć uczniowie wprowadzają dane do arkusza, tworzą czytelną tabelę, dodają kolumnę oceny poprawności działania robota, stosują funkcję LICZ.JEŻELI do zliczania poprawnych i błędnych reakcji, sortują dane według odległości oraz formułują wnioski. Lekcja rozwija umiejętność analizy danych, wykrywania błędów i podejmowania decyzji na podstawie uporządkowanych informacji.

Cele szczegółowe lekcji

Uczeń:

- wyjaśnia, do czego służy tabela w arkuszu kalkulacyjnym,
- wprowadza dane tekstowe i liczbowe do arkusza,
- tworzy i formatuje tabelę z danymi,
- dodaje nagłówki kolumn i krawędzie tabeli,
- analizuje dane z testów robota LEGO SPIKE,
- rozumie zasadę działania robota: odległość ≤ 15 cm oznacza zatrzymanie,
- ocenia, czy reakcja robota była poprawna,
- wpisuje wartości „TAK” i „BŁĄD” w kolumnie oceny poprawności,
- stosuje funkcję LICZ.JEŻELI do zliczania określonych wyników,
- zlicza liczbę poprawnych reakcji robota,
- zlicza liczbę błędnych reakcji robota,
- sortuje dane według wybranego kryterium, np. odległości,
- interpretuje dane po sortowaniu,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- wskazuje przypadki, w których robot popełnia błędy,
- formułuje wnioski dotyczące działania programu lub sensora,
- stosuje funkcje ŚREDNIA, MIN i MAX w zadaniu dodatkowym,
- analizuje czas przejazdu robota,
- zapisuje efekty pracy w pliku lokalnym lub w chmurze.

Treści nauczania – zgodność z podstawą programową

Lekcja realizuje wymagania podstawy programowej poprzez praktyczne wykorzystanie arkusza kalkulacyjnego do porządkowania, analizowania i prezentowania danych. Uczeń tworzy tabelę, wprowadza dane z testów robota, stosuje funkcje obliczeniowe i logiczne, sortuje informacje oraz interpretuje wyniki.

Zajęcia rozwijają umiejętność wykorzystywania arkusza kalkulacyjnego jako narzędzia wspierającego analizę danych i podejmowanie decyzji. Uczniowie nie tylko wykonują obliczenia, ale sprawdzają poprawność działania programu robota LEGO SPIKE na podstawie zebranych danych. W materiale lekcyjnym wskazano, że celem jest wykorzystanie tabeli i funkcji w procesie porządkowania oraz analizy danych, w tym stosowanie funkcji SUMA, ŚREDNIA, MIN, MAX, sortowania i analizy wyników działania robota.

Sposób realizacji wymagań w innowacyjnym programie nauczania

Wymagania programowe będą realizowane poprzez pracę uczniów w arkuszu kalkulacyjnym, np. LibreOffice Calc, Microsoft Excel lub Arkusze Google. Uczniowie będą pracować na danych dotyczących działania robota LEGO SPIKE, którego program powinien reagować na odległość od przeszkody.

Lekcja łączy kompetencje z zakresu arkusza kalkulacyjnego z analizą działania programu robotycznego. Uczniowie uczą się, że dane z testów mogą pomóc wykryć błędy w programie lub działaniu sensora. Tabela porządkuje informacje, funkcje automatyzują obliczenia, a sortowanie ułatwia zauważenie zależności. Dzięki temu arkusz kalkulacyjny staje się narzędziem wspierającym testowanie i ocenę projektów informatycznych.

Metody pracy

- pokaz multimedialny lub demonstracja nauczyciela,
- pogadanka kierowana,
- analiza problemu praktycznego,
- ćwiczenia praktyczne przy komputerze,
- praca z instrukcją krok po kroku,
- praca na danych z testów robota,
- samodzielne tworzenie tabeli i stosowanie funkcji,
- sortowanie danych,
- formułowanie wniosków na podstawie analizy.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Środki dydaktyczne

- komputer uczniowski z arkuszem kalkulacyjnym,
- arkusz kalkulacyjny, np. LibreOffice Calc, Microsoft Excel lub Arkusze Google,
- zestaw LEGO SPIKE lub dane z testów robota LEGO SPIKE,
- przykładowa tabela z odległością i reakcją robota,
- karta pracy lub e-podręcznik,
- rzutnik multimedialny lub ekran interaktywny,
- dostęp do zapisu plików lokalnie lub w chmurze.

Formy pracy

- praca indywidualna przy komputerze,
- praca pod kierunkiem nauczyciela,
- ćwiczenia samodzielne,
- elementy pracy problemowej,
- elementy analizy danych,
- elementy pracy projektowej podczas interpretowania testów robota.

Faza I. Wprowadzenie

Nauczyciel rozpoczyna lekcję od przedstawienia problemu: robot LEGO SPIKE został zaprogramowany tak, aby zatrzymywał się, gdy odległość od przeszkody wynosi 15 cm lub mniej. Podczas testów zebrano dane, ale są one nieuporządkowane i trudno na pierwszy rzut oka ocenić, czy robot działa poprawnie.

Nauczyciel pyta uczniów, jak można sprawdzić poprawność działania robota na podstawie danych. Następnie wyjaśnia, że arkusz kalkulacyjny pozwala uporządkować wyniki w tabeli, dodać kolumnę oceny, policzyć liczbę poprawnych i błędnych reakcji oraz posortować dane tak, aby łatwiej zauważyć zależności.

Wprowadzenie kończy się zapisaniem tematu lekcji oraz określeniem zadania głównego: uczniowie przygotują tabelę z wynikami testów robota, sprawdzą poprawność jego reakcji, zastosują funkcję zliczającą i sformułują wnioski dotyczące działania programu.

Faza II. Część zasadnicza – realizacja

W części praktycznej uczniowie uruchamiają arkusz kalkulacyjny i zapisują plik pod nazwą wskazaną przez nauczyciela, np. „zadanie_15”. Następnie wprowadzają dane do tabeli zawierającej kolumny: „Próba”, „Odległość (cm)” oraz „Reakcja robota”. Dane obejmują kilka prób, np. odległości 20, 12, 17, 10 i 14 cm oraz reakcje „Jechać” lub „Stop”.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Następnie uczniowie dodają nową kolumnę: „Czy reakcja robota jest poprawna?”. Analizują każdą próbę zgodnie z zasadą działania programu: jeśli odległość wynosi 15 cm lub mniej, robot powinien się zatrzymać; jeśli odległość jest większa niż 15 cm, może jechać dalej. W kolumnie oceny wpisują „TAK”, jeśli reakcja była zgodna z zasadą oraz „BŁĄD”, jeśli robot zareagował nieprawidłowo.

Kolejnym krokiem jest zastosowanie funkcji LICZ.JEŻELI. Uczniowie zliczają, ile razy robot działał poprawnie, wykorzystując zakres z kolumny oceny, np. D2:D6 oraz kryterium „tak”. Następnie modyfikują formułę, aby policzyć liczbę błędów, zmieniając kryterium na „błąd”. Dzięki temu uczą się, że funkcje mogą analizować nie tylko liczby, ale również teksty zapisane w komórkach.

W dalszej części uczniowie sortują dane według odległości rosnąco. Zaznaczają dane tabeli bez nagłówka, wybierają opcję sortowania i wskazują kolumnę z odległością jako kryterium. Po posortowaniu sprawdzają, czy łatwiej zauważyć, przy których odległościach robot powinien się zatrzymać i gdzie wystąpiły błędy.

Następnie uczniowie zapisują wnioski, odpowiadając na pytania: czy robot zawsze działał poprawnie, w których próbach popełnia błędy oraz co należałoby poprawić w programie. W przykładowej analizie błąd występuje przy odległościach 10 i 14 cm, ponieważ robot jedzie, mimo, że powinien się zatrzymać. Wniosek może dotyczyć konieczności poprawy warunku w programie lub działania sensora.

W zadaniu dodatkowym uczniowie analizują dane obejmujące odległość i czas przejazdu robota. Obliczają średnią odległość, minimalną i maksymalną odległość, najkrótszy czas, najdłuższy czas oraz średni czas przejazdu. Dodają także kolumnę „Długi przejazd”, w której oznaczają, czy czas był dłuższy niż 2 sekundy, a następnie zliczają takie przypadki funkcją LICZ.JEŻELI.

Faza III. Podsumowanie

Na zakończenie lekcji uczniowie prezentują przygotowane tabele i wyniki analizy. Nauczyciel sprawdza, czy dane zostały poprawnie uporządkowane, czy funkcje zliczające działają prawidłowo, czy sortowanie zostało wykonane według właściwego kryterium oraz czy uczniowie potrafią wskazać błędne reakcje robota.

Uczniowie odpowiadają na pytania kontrolne dotyczące funkcji SUMA, ŚREDNIA, celu stosowania tabel, warunku zatrzymania robota, sortowania danych oraz roli tabeli w arkuszu kalkulacyjnym. Nauczyciel podkreśla, że tabela, funkcje i sortowanie pomagają analizować dane szybciej, dokładniej i bardziej obiektywnie niż ręczne przeglądanie wyników.

Kryteria oceniania

Ocena dopuszczająca



Podkarpacki Zespół
Placówek Wojewódzkich
w Rzeszowie



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Uczeń z pomocą nauczyciela tworzy prostą tabelę z danymi, wpisuje podstawowe wartości i rozpoznaje, kiedy robot powinien się zatrzymać.

Ocena dostateczna

Uczeń samodzielnie uzupełnia tabelę, dodaje kolumnę oceny poprawności i wskazuje poprawne oraz błędne reakcje robota.

Ocena dobra

Uczeń poprawnie stosuje funkcję LICZ.JEŻELI, zlicza poprawne i błędne reakcje, sortuje dane według odległości oraz zapisuje wnioski z analizy.

Ocena bardzo dobra

Uczeń samodzielnie i świadomie analizuje dane z testów robota, stosuje funkcje LICZ.JEŻELI, ŚREDNIA, MIN, MAX, interpretuje wyniki, wykrywa błędy działania programu i proponuje konkretne usprawnienia.

Krótki opis problemu do rozwiązania

Uczniowie mają sprawdzić, czy robot LEGO SPIKE działa zgodnie z założeniem: powinien zatrzymywać się, gdy odległość od przeszkody wynosi 15 cm lub mniej. Problem polega na tym, że same dane z testów są trudne do oceny bez uporządkowania. Uczniowie tworzą tabelę, dodają ocenę poprawności, stosują funkcję zliczającą błędy, sortują dane i na tej podstawie formułują wnioski o działaniu programu oraz możliwych poprawkach.

Faza IV. Quiz/Test sprawdzający

1. Do czego służy funkcja SUMA?

A. Liczy średnią wartości

B. Dodaje wartości w komórkach

C. Znajduje największą wartość

2. Która funkcja oblicza średnią?

A. MIN

B. MAX

C. ŚREDNIA



Podkarpacki Zespół
Placówek Wojewódzkich
w Rzeszowie



POLITECHNIKA
RZESZOWSKA
Im. IGNACEGO ŁUKASIEWICZA



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

3. Co daje zamiana danych na tabelę?

A. Usuwa dane

B. Utrudnia pracę

C. Ułatwia porządkowanie i analizę danych

4. Kiedy robot powinien się zatrzymać?

A. Gdy odległość > 15 cm

B. Gdy odległość ≤ 15 cm

C. Zawsze

5. Do czego służy sortowanie danych?

A. Usuwa błędy

B. Porządkuje dane według wybranego kryterium

C. Zmienia wartości komórek

6. Do czego służy tabela w arkuszu kalkulacyjnym?

A. Do rysowania wykresów

B. Do porządkowania i organizowania danych

C. Do usuwania danych

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 15 - Tabele i funkcje porządkujemy dane

Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
- Przedstawienie uczniom celu i zakresu tematycznego lekcji.
- Uruchomienie na e-platformie materiałów do lekcji
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów.
- Zadanie pracy domowej (opcjonalnie).

Załącznik nr 2 – Konspekt nauczyciela lekcji nr 15 - Tabele i funkcje porządkujemy dane

Cel lekcji

Celem lekcji jest wykorzystanie tabeli oraz funkcji w procesie porządkowania i analizy danych za pomocą arkusza kalkulacyjnego.

Wprowadzenie

Uczeń wykorzystuje tabelę oraz funkcje w procesie porządkowania i analizy danych za pomocą arkusza kalkulacyjnego, ćwiczy poprawne wprowadzanie danych, tworzy i formatuje tabelę, stosuje podstawowe funkcje (SUMA, ŚREDNIA, MIN, MAX), porządkuje dane za pomocą sortowania, analizuje wyniki działania robota LEGO SPIKE oraz wyciąga wnioski na podstawie zebranych danych.

Przebieg lekcji

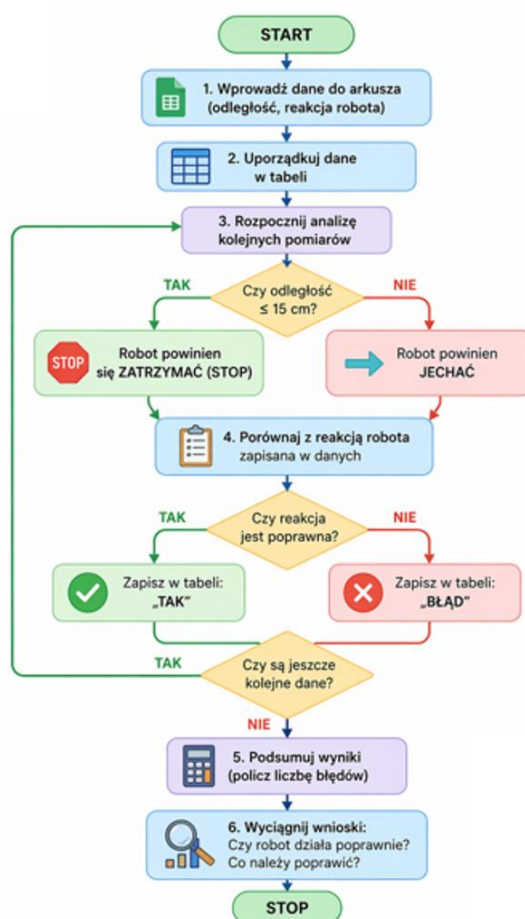
Robot LEGO SPIKE został zaprogramowany tak, aby zatrzymywał się, gdy odległość od przeszkody wynosi **15 cm lub mniej**. Podczas testów zebrano dane, ale nie są one uporządkowane i trudno ocenić, czy robot działa poprawnie.

Problem:

Czy robot działa zgodnie z założeniami? Jak uporządkować i przeanalizować dane, aby to sprawdzić?

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zapoznaj się z algorytmem działania programu i przeanalizuj przykładowe dane lub sam dokonaj pomiarów budując robota i testując sensor odległości.



Rozwiązanie problemu:

Etap 1: Uporządkowanie danych

Wprowadź dane do arkusza:

Próba	Odległość (cm)	Reakcja robota
1	20	Jechać
2	12	Stop

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Próba	Odległość (cm)	Reakcja robota
3	17	Jechać
4	10	Jechać
5	14	Jechać

Uruchom program LibreOffice Calc. Zamień dane na tabelę w arkuszu kalkulacyjnym. Wybierz dowolne komórki i arkusz. Zapisz plik we wskazanym miejscu przez nauczyciela nadając mu nazwę np. zadanie_15.

zadanie 15.ods — LibreOffice Calc

Plik Edycja Widok Wstaw Format Style Arkusz Dane Narzędzia Okno Pomoc

Liberation Sans 10 pt. G K P A Wyrównaj do lewej (Ctrl+L)

	A	B	C
1	Próba	Odległość (cm)	Reakcja robota
2	1	20	jechać
3	2	12	stop
4	3	17	jechać
5	4	10	jechać
6	5	14	jechać

Krok 2: Analiza poprawności działania

Do analizowanych danych (tabeli) dodaj kolumnę wraz z krawędziami: „Czy reakcja robota jest poprawna?”

zadanie 15.ods — LibreOffice Calc

Plik Edycja Widok Wstaw Format Style Arkusz Dane Narzędzia Okno Pomoc

Liberation Sans 10 pt. G K P A Wyrównaj do lewej (Ctrl+L)

D1:D6 Czy reakcja robota jest poprawna

	A	B	C	D
1	Próba	Odległość (cm)	Reakcja robota	Czy reakcja robota jest poprawna?
2	1	20	jechać	
3	2	12	stop	
4	3	17	jechać	
5	4	10	jechać	
6	5	14	jechać	

Krawędzie: Zewnętrzna krawędź i wszystkie linie wewnętrzne

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Wpisz:

- TAK – jeśli robot zachował się zgodnie z zasadą (≤ 15 cm $\rightarrow$ stop),
- BŁĄD – jeśli nie.

	A	B	C	D
1	Próba	Odległość (cm)	Reakcja robota	Czy reakcja robota jest poprawna
2	1	20	jechać	tak
3	2	12	stop	tak
4	3	17	jechać	tak
5	4	10	jechać	BŁĄD
6	5	14	jechać	BŁĄD

Krok 3: Zastosowanie funkcji

Policz:

- ile razy robot działał poprawnie,
- ile razy popełnił błąd.

Wykorzystaj funkcje arkusza kalkulacyjnego (np. *LICZ.JEŻELI(zakres; kryteria)*). Pomyśl jak można zliczyć liczbę wyrazów „tak” oraz „BŁĄD”. Zwróć uwagę, że dane znajdują się w komórkach od D2 do D6 czyli tworzą zakres D2:D6. Jakie może być kryterium zliczania? Wpisz „tak” lub „BŁĄD”).

Poprawna formuła powinna przyjąć postać:

	A	B	C	D	E
1	Próba	Odległość (cm)	Reakcja robota	Czy reakcja robota jest poprawna	
2	1	20	jechać	tak	
3	2	12	stop	tak	
4	3	17	jechać	tak	
5	4	10	jechać	BŁĄD	
6	5	14	jechać	BŁĄD	
7					5 W x 1 K
8					
9		Ile razy robot działał poprawnie?		=licz.jeżeli(D2:D6;"tak")	
10					
11		Ile razy popełnił błąd?			

Po zatwierdzeniu przyciskiem *Enter* formuły w komórce D9 pojawi się wynik 3. Wynik jest zgodny z tabelą.

Zmodyfikuj formułę aby obliczyć ile razy robot popełnił błąd. Możesz przepisać od początku formułę lub ją przekopiować, ale zwróć uwagę na zmianę zakresu. Zakres danych jest taki sam jak wcześniej czyli D2:D6. W formule wielkość liter nie ma znaczenia. Wyraz „błąd” możesz zapisać wielkimi lub małymi literami.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

=LICZ.JEŻELI(D2:D6;"błąd")

Sprawdź czy wynik jest poprawny i wynosi 2?

	A	B	C	D
1	Próba	Odległość (cm)	Reakcja robota	Czy reakcja robota jest poprawna
2	1	20	jechać	tak
3	2	12	stop	tak
4	3	17	jechać	tak
5	4	10	jechać	BŁĄD
6	5	14	jechać	BŁĄD
7				
8				
9		Ile razy robot działał poprawnie?		3
10				
11		Ile razy popełnił błąd?		2

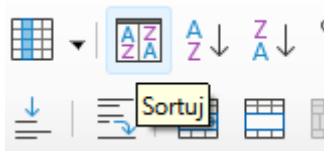
Krok 4: Sortowanie danych

Posortuj dane według odległości (rosnąco). Oceń czy po sortowaniu łatwiej zauważyć błędy?

1. Zaznacz dane tabeli bez nagłówka.

	A	B	C	D
1	Próba	Odległość (cm)	Reakcja robota	Czy reakcja robota jest poprawna
2	1	20	jechać	tak
3	2	12	stop	tak
4	3	17	jechać	tak
5	4	10	jechać	BŁĄD
6	5	14	jechać	BŁĄD

2. Kliknij kursorem na ikonę sortowania:



3. Wybierz poprawne kryterium sortowania, czyli kolumna B.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

4. Zatwierdź przyciskiem OK.
5. Sprawdź jak wygląda tabela po posortowaniu?

	A	B	C	D
1	Próba	Odległość (cm)	Reakcja robota	Czy reakcja robota jest poprawna
2	4	10	jechać	BŁĄD
3	2	12	stop	tak
4	5	14	jechać	BŁĄD
5	3	17	jechać	tak
6	1	20	jechać	tak
7				
8				
9		Ile razy robot działał poprawnie?		3
10				
11		Ile razy popełnił błąd?		2

Krok 5: Wnioski i podsumowanie analiz

Sformułuj wnioski odpowiadając na pytania:

1. Czy robot zawsze działa poprawnie?
2. W których przypadkach popełnia błędy?
3. Co należałoby poprawić w programie?

Przykładowa analiza

Błąd występuje w próbie w dwóch przypadkach – dla odległości 10 i 14 cm → **robot jedzie, a powinien się zatrzymać. Na podstawie analizy danych można wysnuć wniosek, że robot nie zawsze działa poprawnie i należy poprawić warunek w programie lub działanie sensora.**

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

1. Czy robot zawsze działa poprawnie? NIE
2. W których przypadkach popełnia błędy? Próba 4 i 5.
3. Co należałoby poprawić w programie? Poprawić np. czułość czujnika lub zmodyfikować program.

Zadania do samodzielnego wykonania

1. Zmień dane w tabeli. Dodaj kilka prób i sprawdź ponownie działanie formuł. Przykładowa nowa tabela poniżej. Samodzielnie oceń reakcję robota i zlicz ile razy robot działał poprawnie, a ile razy popełnił błąd.

	A	B	C	D
1	Próba	Odległość (cm)	Reakcja robota	Czy reakcja robota jest poprawna
2	1	20	jechać	tak
3	2	12	stop	tak
4	3	17	jechać	tak
5	4	10	jechać	BŁĄD
6	5	14	jechać	BŁĄD
7	6	17	jechać	
8	7	34	jechać	
9	8	8	stop	
10	9	32	jechać	
11	10	13	jechać	

Zadanie dodatkowe.

1. Dokonaj analizy czasu przejazdu robota i oceń czy działa poprawnie.

Wprowadź dane do arkusza:

Próba	Odległość (cm)	Czas (s)
1	22	2,5
2	14	1,2
3	30	3,1

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Próba	Odległość (cm)	Czas (s)
4	10	1,0
5	18	2,2

Podpowiedź:

	A	B	C
1	Próba	Odległość (cm)	Czas (s)
2	1	22	2,5
3	2	14	1,2
4	3	30	3,1
5	4	10	1
6	5	18	2,2

2. Dla zebranych danych oblicz: średnią odległość, najkrótszy czas, najdłuższy czas oraz średni czas.

Wskazówka:

Użyj funkcji: ŚREDNIA (AVERAGE), MIN, MAX. Zaznacz zakres komórek zamiast wpisywać liczby ręcznie.

Podpowiedź:

Średnia odległość:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

	A	B	C
1	Próba	Odległość (cm)	Czas (s)
2	1	22	2,5
3	2	14	1,2
4	3	30	3,1
5	4	10	1
6	5	18	2,2
7			5 W × 1 K
8			
9	Średnia odległość	=średnia(B2:B6)	
10	Minimalna odległość		
11	Maksymalna odległość		

Minimalna odległość:

8			
9	Średnia odległość	18,8	
10	Minimalna odległość	=min(B2:B6)	
11	Maksymalna odległość		
12			

Maksymalna odległość:

9	Średnia odległość	18,8	
10	Minimalna odległość	10	
11	Maksymalna odległość	=maks(B2:B6)	

Najkrótszy czas:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

	A	B	C	D
1	Próba	Odległość (cm)	Czas (s)	
2	1	22	2,5	
3	2	14	1,2	
4	3	30	3,1	
5	4	10	1	
6	5	18	2,2	
7				5 W x 1 K
8				
9	Średnia odległość	18,8		
10	Minimalna odległość	10		
11	Maksymalna odległość	30		
12				
13	Najkrótszy czas	=min(C2:C6)		
14	Najdłuższy czas			
15	Średni czas			

Średni czas przejazdu:

13	Najkrótszy czas	1	
14	Najdłuższy czas	3,1	
15	Średni czas	=ŚREDNIA(C2:C6)	

3. Dodaj kolumnę: „Długi przejazd” Zaimplementuj zasadę: jeśli czas był dłuższy niż 2 sekundy → powinno być „tak”, jeśli krótszy → „nie”.
4. Policz, ile razy robot jechał dłużej niż 2 sekundy (czyli był to „długi przejazd”), a ile razy krócej niż 2 sekundy?

Wskazówka:

Użyj funkcji: LICZ.JEŻELI (zakres; warunek) Podpowiedź: warunek to np. „tak”.

Podpowiedź:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

	A	B	C	D	E
1	Próba	Odległość (cm)	Czas (s)	Długi przejazd?	
2	1	22	2,5	tak	
3	2	14	1,2	nie	
4	3	30	3,1	tak	
5	4	10	1	nie	
6	5	18	2,2	tak	
7					5 W x 1 K
8					
9	Średnia odległość	18,8			
10	Minimalna odległość	10			
11	Maksymalna odległość	30			
12					
13	Najkrótszy czas	1			
14	Najdłuższy czas	3,1			
15	Średni czas	2			
16					
17					
18	Ile razy robot jechał dłużej niż 2 sekundy?			=licz.jeżeli(D2:D6;"tak")	

5. Posortuj tabelę od najmniejszej do największej odległości.

Wskazówka:

Znajdź opcję „Sortuj rosnąco”.

Zastanów się czy łatwiej teraz zauważyć, kiedy robot powinien się zatrzymać?

6. Dodaj kilka wierszy i zaktualizuj wyniki.
7. Zapisz wnioski z analizy odpowiadając na pytania:

- Przy jakich odległościach robot powinien reagować?
- Czy dane są uporządkowane i czytelne?
- Do czego przydają się funkcje?

Podsumowanie zajęć:

Wykorzystanie tabel i funkcji w arkuszu kalkulacyjnym pozwala na skuteczne porządkowanie oraz analizę danych. Uczniowie przekonali się, że odpowiednie ustrukturyzowanie informacji w formie tabeli zwiększa czytelność i ułatwia pracę z danymi, szczególnie gdy jest ich dużo.

Zastosowanie podstawowych funkcji, takich jak SUMA, ŚREDNIA, MIN i MAX, LICZ.JEŻELI umożliwia szybkie wykonywanie obliczeń oraz wyciąganie wniosków bez konieczności ręcznego liczenia. Sortowanie danych dodatkowo pozwala lepiej dostrzec zależności i błędy w działaniu analizowanego systemu, np. robota LEGO SPIKE.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Wnioskiem z ćwiczenia jest to, że arkusz kalkulacyjny stanowi ważne narzędzie wspierające analizę danych i podejmowanie decyzji. Umiejętność korzystania z tabel i funkcji jest kluczowa w pracy z danymi oraz w procesie programowania i testowania rozwiązań informatycznych.

Zadania dodatkowe

1. Zmień dane w tabeli. Dodaj kilka prób i sprawdź ponownie działanie formuł. Przykładowa nowa tabela poniżej. Samodzielnie oceń reakcję robota i zlicz ile razy robot działał poprawnie, a ile razy popełnił błąd. Następnie udostępniij dokument członkom zespołu i poproś ich o sprawdzenie poprawności struktury, zwracając uwagę na możliwość przypadkowej edycji lub usunięcia treści oraz zaproponuj zabezpieczenia (np. wersjonowanie dokumentu, komentarze zamiast edycji).
2. Zmień dane w tabeli, wprowadzając nowe wartości odległości i reakcji robota. Dodaj co najmniej 3 nowe próby i sprawdź ponownie działanie formuł (np. LICZ.JEŻELI). Samodzielnie oceń, ile razy robot działał poprawnie, a ile razy popełnił błąd. Następnie zapisz plik w chmurze i udostępniij go kolegom z klasy. Sprawdź, kto ma dostęp do edycji dokumentu. Zastanów się, jakie zagrożenia mogą wystąpić (np. przypadkowe usunięcie danych) i zaproponuj sposoby zabezpieczenia pliku (np. tylko odczyt, komentarze, historia wersji).
3. Utwórz nową tabelę, w której zapiszesz dane z dwóch różnych testów robota (np. dwa różne programy lub ustawienia sensora). Porównaj wyniki, stosując funkcje (ŚREDNIA, MIN, MAX) oraz sortowanie. Na tej podstawie oceń, który wariant działa lepiej. Następnie przygotuj krótką informację (np. w komentarzu w arkuszu lub wiadomości), w której przedstawisz wyniki swojej analizy. Zwróć uwagę na sposób komunikacji (napisz jasno, rzeczowo i bez ujawniania zbędnych danych). Zastanów się, jakie informacje można bezpiecznie udostępniać innym, a które powinny pozostać prywatne.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 4: Biurowa moc: TechOffice

Temat lekcji nr 16 - Test sensorów i wykres w arkuszu kalkulacyjnym

Treści nauczania (wymagania szczegółowe - podstawa programowa) – II. 3) c);

Cel lekcji

Celem lekcji jest rozwijanie umiejętności testowania sensorów robota LEGO SPIKE oraz analizowania zebranych danych w arkuszu kalkulacyjnym. Uczniowie poznają sposób zapisywania wyników pomiarów w tabeli, obliczania podstawowych wartości statystycznych oraz tworzenia wykresu, który pomaga ocenić, jak zmienia się odległość robota od przeszkody w czasie.

Podczas zajęć uczniowie uczą się, że dane z czujnika mogą być podstawą do podejmowania decyzji programistycznych. Analizują, czy czujnik działa poprawnie, czy pomiary są logiczne i stabilne oraz przy jakiej wartości odległości robot powinien zareagować, np. zatrzymać się przed przeszkodą.

Cele szczegółowe lekcji

Uczeń:

- wyjaśnia, po co testuje się sensory robota,
- rozumie, że czujnik odległości dostarcza danych potrzebnych do sterowania robotem,
- wprowadza dane pomiarowe do arkusza kalkulacyjnego,
- tworzy tabelę zawierającą numer pomiaru, odległość i czas,
- oblicza średnią odległość z wykorzystaniem funkcji ŚREDNIA,
- oblicza minimalną odległość z wykorzystaniem funkcji MIN,
- oblicza maksymalną odległość z wykorzystaniem funkcji MAX,
- oblicza różnicę między największą a najmniejszą wartością,
- zaokrągla wynik do dwóch miejsc po przecinku,
- interpretuje dane pomiarowe zapisane w tabeli,
- sprawdza, czy odległość maleje w czasie,
- wskazuje moment, w którym robot znajduje się najbliżej przeszkody,
- tworzy wykres punktowy XY na podstawie danych,
- ustawia oś X jako czas, a oś Y jako odległość,
- dodaje tytuł wykresu oraz opisy osi,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- odczytuje informacje z wykresu,
- wskazuje moment, w którym robot powinien się zatrzymać,
- dodaje kolumnę z oceną warunku zatrzymania robota,
- zlicza, ile razy warunek zatrzymania został spełniony,
- oblicza błąd pomiaru w zadaniu dodatkowym,
- porównuje rzeczywistą i zmierzoną odległość,
- zapisuje wnioski z analizy danych i wykresu.

Treści nauczania – zgodność z podstawą programową

Lekcja realizuje wymagania podstawy programowej poprzez praktyczne wykorzystanie arkusza kalkulacyjnego do opracowania danych pomiarowych i tworzenia wykresów. Uczeń gromadzi dane z testów sensora, zapisuje je w tabeli, stosuje funkcje obliczeniowe, tworzy wykres oraz odczytuje z niego informacje potrzebne do oceny działania robota.

Zajęcia rozwijają umiejętność wykorzystywania arkusza kalkulacyjnego jako narzędzia analitycznego w projekcie robotycznym. Uczniowie nie tylko wprowadzają dane i tworzą wykres, ale także interpretują wyniki, oceniają stabilność pomiarów i wyciągają wnioski dotyczące programowania robota. W materiale lekcyjnym wskazano, że celem lekcji jest przeprowadzenie testów sensorów robota LEGO SPIKE, zapisanie wyników w tabeli arkusza kalkulacyjnego oraz wykorzystanie funkcji i wykresów do analizy danych w celu lepszego programowania robota.

Sposób realizacji wymagań w innowacyjnym programie nauczania

Wymagania programowe będą realizowane poprzez praktyczną pracę uczniów w arkuszu kalkulacyjnym, np. LibreOffice Calc, Microsoft Excel lub Arkusze Google. Uczniowie będą analizować dane z testu czujnika odległości robota LEGO SPIKE, wykonując kolejne czynności: wpisanie danych, obliczenie wartości średniej, minimalnej i maksymalnej, wyznaczenie różnicy wartości oraz utworzenie wykresu punktowego.

Lekcja łączy pracę z arkuszem kalkulacyjnym z procesem testowania robota. Uczniowie widzą, że dane pomiarowe są potrzebne do oceny działania sensora i do podejmowania decyzji programistycznych, np. przy jakiej odległości robot powinien się zatrzymać. Dzięki temu arkusz kalkulacyjny staje się narzędziem wspierającym projektowanie, testowanie i doskonalenie programu robota.

Metody pracy

- pokaz multimedialny lub demonstracja nauczyciela,
- pogadanka kierowana,
- analiza problemu praktycznego,
- ćwiczenia praktyczne przy komputerze,
- praca z danymi pomiarowymi,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- praca z instrukcją krok po kroku,
- tworzenie tabeli i wykresu,
- interpretacja wyników,
- formułowanie wniosków na podstawie danych.

Środki dydaktyczne

- komputer uczniowski z arkuszem kalkulacyjnym,
- arkusz kalkulacyjny, np. LibreOffice Calc, Microsoft Excel lub Arkusze Google,
- zestaw LEGO SPIKE z czujnikiem odległości,
- przykładowe dane z testu czujnika,
- karta pracy lub e-podręcznik,
- rzutnik multimedialny lub ekran interaktywny,
- opcjonalnie: przeszkody do testowania czujnika, np. pudełko, klocek, kartka,
- dostęp do zapisu plików lokalnie lub w chmurze.

Formy pracy

- praca indywidualna przy komputerze,
- praca pod kierunkiem nauczyciela,
- ćwiczenia samodzielne,
- elementy pracy projektowej,
- elementy pracy badawczej podczas testowania czujnika,
- elementy analizy danych i interpretacji wykresu.

Faza I. Wprowadzenie

Nauczyciel rozpoczyna lekcję od przedstawienia problemu: robot LEGO SPIKE wykorzystuje czujnik odległości, aby reagować na przeszkody, ale przed zaprogramowaniem reakcji trzeba sprawdzić, czy czujnik działa poprawnie i czy jego wskazania można uznać za wiarygodne. Uczniowie zastanawiają się, dlaczego samo uruchomienie robota nie wystarcza do oceny działania sensora.

Nauczyciel wyjaśnia, że testowanie czujników polega na wykonywaniu pomiarów, zapisywaniu wyników i analizowaniu ich. Arkusz kalkulacyjny pozwala uporządkować dane w tabeli, obliczyć wartości charakterystyczne oraz przedstawić zmiany na wykresie. Dzięki temu łatwiej zauważyć, czy pomiary zmieniają się logicznie, czy występują błędy oraz przy jakiej odległości robot powinien zareagować.

Wprowadzenie kończy się zapisaniem tematu lekcji oraz określeniem zadania głównego: uczniowie zapiszą dane z testu czujnika odległości, obliczą podstawowe wyniki, utworzą wykres punktowy i zaproponują wartość odległości, przy której robot powinien się zatrzymać.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Faza II. Część zasadnicza – realizacja

W części praktycznej uczniowie uruchamiają arkusz kalkulacyjny i wpisują przykładowe dane z testu czujnika odległości. Tabela zawiera kolumny: „Pomiar”, „Odległość (cm)” oraz „Czas (s)”. Uczniowie wprowadzają kolejne pomiary, np. odległości 40, 35, 28, 20, 15 i 10 cm oraz odpowiadające im wartości czasu od 0,5 do 3,0 s.

Następnie uczniowie obliczają podstawowe wartości opisujące dane: średnią odległość, minimalną odległość, maksymalną odległość oraz różnicę między największą a najmniejszą wartością. Korzystają z funkcji arkusza kalkulacyjnego, takich jak ŚREDNIA, MIN i MAX, oraz tworzą prostą formułę odejmowania dla różnicy. W przypadku długiego wyniku średniej uczą się zaokrąglać wartość do dwóch miejsc po przecinku.

Kolejnym krokiem jest interpretacja danych. Uczniowie odpowiadają na pytania: czy odległość maleje w czasie, w którym pomiarze robot znajduje się najbliżej przeszkody oraz które dane są najważniejsze dla programowania reakcji robota. Nauczyciel kieruje uwagę uczniów na wartość graniczną, np. 15 cm, przy której robot powinien się zatrzymać.

W dalszej części uczniowie tworzą wykres punktowy XY. Zaznaczają dane dotyczące czasu i odległości, wybierają opcję wstawienia wykresu, a następnie typ „wykres punktowy XY”, najlepiej w wariantcie „punkty i linie”. Nauczyciel wyjaśnia, że w tym zadaniu wykres punktowy jest właściwy, ponieważ oś X powinna przedstawiać rzeczywiste wartości czasu, a nie tylko kolejne kategorie.

Uczniowie uzupełniają wykres o tytuł, np. „Zmiana odległości w czasie” oraz opisy osi: oś X - „Czas (s)”, oś Y - „Odległość (cm)”. Następnie na podstawie wykresu opisują, jak zmienia się odległość i wskazują moment, w którym robot powinien się zatrzymać, czyli gdy odległość wynosi 15 cm lub mniej.

Kolejnym etapem jest dodanie kolumny „Czy robot powinien się zatrzymać?”. Uczniowie wpisują wartości „TAK” lub „NIE” na podstawie warunku $\text{odległość} \leq 15 \text{ cm}$. Następnie zliczają, ile razy warunek został spełniony, korzystając z odpowiedniej funkcji zliczającej, np. LICZ.JEŻELI.

W zadaniu dodatkowym uczniowie rozbudowują arkusz o kolumny: „Rzeczywista odległość”, „Zmierzona odległość” oraz „Błąd pomiaru”. Obliczają błąd jako różnicę między wartością rzeczywistą a odczytem sensora, przeciągają formułę do kolejnych wierszy, obliczają średni i największy błąd pomiaru oraz tworzą wykres porównujący dwie serie danych: rzeczywistą i zmierzoną odległość.

Faza III. Podsumowanie

Na zakończenie lekcji uczniowie prezentują przygotowane tabele, wyniki obliczeń i wykresy. Nauczyciel sprawdza, czy dane zostały poprawnie wpisane, czy funkcje działają prawidłowo,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

czy wykres punktowy ma właściwe osie oraz czy uczniowie potrafią wyjaśnić, jak zmienia się odległość w czasie.

Uczniowie odpowiadają na pytania kontrolne dotyczące celu wykresu, typu wykresu najlepiej pokazującego zmiany w czasie oraz funkcji znajdującej najmniejszą wartość. Nauczyciel podkreśla, że testowanie sensorów i analiza danych pomagają lepiej programować robota, ponieważ decyzje w programie powinny opierać się na sprawdzonych i zrozumiałych pomiarach.

Kryteria oceniania

Ocena dopuszczająca

Uczeń z pomocą nauczyciela wpisuje dane pomiarowe do arkusza i tworzy prostą tabelę z pomiarami.

Ocena dostateczna

Uczeń samodzielnie oblicza podstawowe wartości, takie jak średnia, minimum i maksimum, oraz potrafi wskazać najmniejszą i największą odległość.

Ocena dobra

Uczeń poprawnie tworzy wykres punktowy XY, opisuje osie, interpretuje zmianę odległości w czasie oraz wskazuje moment, w którym robot powinien się zatrzymać.

Ocena bardzo dobra

Uczeń samodzielnie analizuje dane z testu sensora, stosuje funkcje, oblicza różnicę i błąd pomiaru, tworzy wykres z jedną lub dwiema seriami danych oraz formułuje uzasadnione wnioski dotyczące poprawy programu robota.

Krótki opis problemu do rozwiązania

Uczniowie mają sprawdzić, czy czujnik odległości robota LEGO SPIKE działa poprawnie i czy jego dane można wykorzystać do zaprogramowania reakcji robota. Problem polega na tym, aby nie oceniać działania sensora „na oko”, lecz zebrać pomiary, wpisać je do arkusza, obliczyć najważniejsze wartości, utworzyć wykres i na tej podstawie ustalić, przy jakiej odległości robot powinien się zatrzymać.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Faza IV. Quiz/Test sprawdzający

Do czego służy wykres w arkuszu kalkulacyjnym?

A. Do usuwania danych

B. Do wizualizacji danych

C. Do pisania tekstu

2. Który wykres najlepiej pokazuje zmiany w czasie?

A. Kołowy

B. Liniowy

C. Tabelaryczny

3. Która funkcja znajdzie najmniejszą wartość?

A. MAX

B. MIN

C. SUMA

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 16 - Test sensorów i wykres w arkuszu kalkulacyjnym

Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
- Przedstawienie uczniom celu i zakresu tematycznego lekcji.
- Uruchomienie na e-platformie materiałów do lekcji
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów.
- Zadanie pracy domowej (opcjonalnie).

Załącznik nr 2 – Konspekt nauczyciela lekcji nr 16 - Test sensorów i wykres w arkuszu kalkulacyjnym

Cel lekcji

Celem lekcji jest testowanie sensorów robota. Uczeń potrafi przeprowadzić test sensorów robota LEGO SPIKE, zapisać wyniki w tabeli arkusza kalkulacyjnego oraz wykorzystać funkcje i wykresy do analizy danych w celu lepszego programowania robota.

Przebieg lekcji

Robot LEGO SPIKE wykorzystuje czujnik odległości, aby reagować na przeszkody. Jednak nie wiadomo, czy czujnik działa dokładnie i czy robot podejmuje decyzje we właściwym momencie. Podczas testów zebrano dane, ale nie zostały one przeanalizowane.

Problem do rozwiązania:

Czy czujnik robota działa poprawnie i czy na jego podstawie można prawidłowo zaprogramować reakcję robota?

1. Wprowadź do arkusza przykładowe dane z testu czujnika odległości robota:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Pomiar	Odległość (cm)	Czas (s)
1	40	0,5
2	35	1,0
3	28	1,5
4	20	2,0
5	15	2,5
6	10	3,0

Po wprowadzeniu danych arkusz przyjmie postać:

	A	B	C
1	Pomiar	Czas (s)	Odległość (cm)
2	1	0,5	40
3	2	1	35
4	3	1,5	28
5	4	2	20
6	5	2,5	15
7	6	3	10

2. Oblicz:

- średnią odległość,
- minimalną odległość,
- maksymalną odległość,
- różnicę między największą a najmniejszą wartością.

Podpowiedź:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

	A	B	C
1	Pomiar	Czas (s)	Odległość (cm)
2	1	0,5	40
3	2	1	35
4	3	1,5	28
5	4	2	20
6	5	2,5	15
7	6	3	10
8			
9			
10	Średnia odległość		=ŚREDNIA(C2:C7)
11	Minimalna odległość		
12	Maksymalna odległość		
13			
14	Różnica między największą a najmniejszą wartością		

Średnia odległość wyniosła 24,666666667. Zaokrąglij wynik do dwóch miejsc po przecinku „usuwając miejsca dziesiętne”:

Plik Edycja Widok Wstaw Format Style Arkusz Dane Narzędzia Okno Pomoc

10 pt. G K P A

C10 =ŚREDNIA(C2:C7) Usun miejsce dziesiętne

	A	B	C	D
1	Pomiar	Czas (s)	Odległość (cm)	
2	1	0,5	40	
3	2	1	35	
4	3	1,5	28	
5	4	2	20	
6	5	2,5	15	
7	6	3	10	
8				
9				
10	Średnia odległość		24,66667	
11	Minimalna odległość			
12	Maksymalna odległość			
13				
14	Różnica między największą a najmniejszą wartością			

Różnicę między największą, a najmniejszą wartością należy obliczyć tworząc prostą formułę:

10	Średnia odległość	24,67
11	Minimalna odległość	10
12	Maksymalna odległość	40
13		
14	Różnica między największą a najmniejszą wartością	=C12-C11

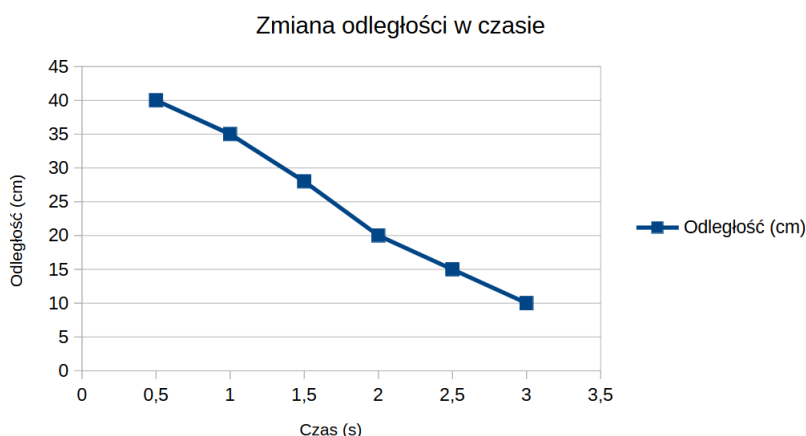
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

3. Dokonaj interpretacji danych. Odpowiedz:

- Czy odległość maleje w czasie?
- W którym momencie robot znajduje się najbliżej przeszkody?
- Jakie dane są najważniejsze dla programowania robota?

4. Na podstawie danych utwórz wykres punktowy:

- oś X → czas (s),
- oś Y → odległość (cm).



Podpowiedź:

1. Zaznacz dane

Zaznacz całą tabelę razem z nagłówkami:

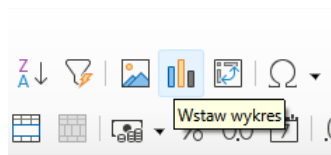
- kolumna „Czas (s)”
- kolumna „Odległość (cm)”

	A	B	C
	Pomiar	Czas (s)	Odległość (cm)
1	1	0,5	40
2	2	1	35
3	3	1,5	28
4	4	2	20
5	5	2,5	15
6	6	3	10

2. Wstaw wykres

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

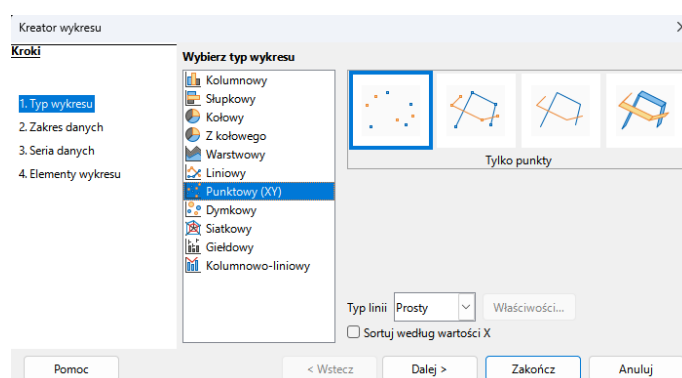
- Kliknij: **Wstaw** → **Wykres**



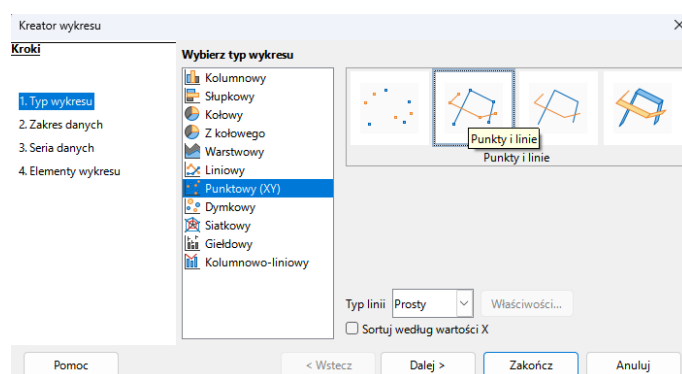
- Otworzy się kreator wykresu

3. Wybierz typ wykresu

- Wybierz: **Wykres punktowy (XY)**
NIE wybieraj zwykłego liniowego, bo wtedy oś X będzie traktowana jako kategorie



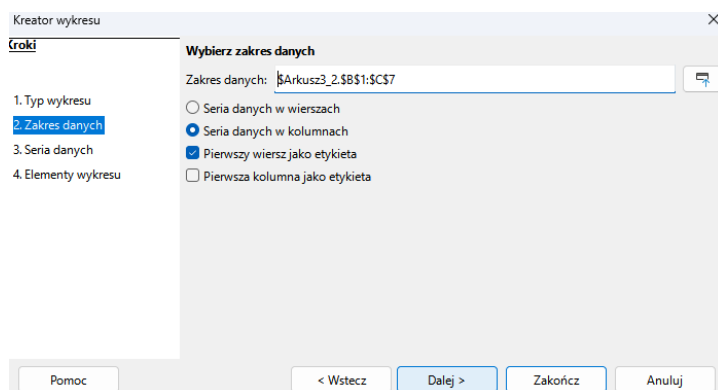
- Wybierz wariant „punkty i linie”



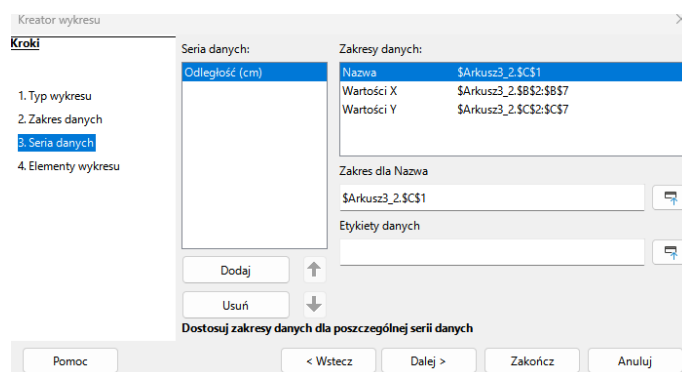
4. Ustaw dane

Przejdź dalej. W kolejnym kroku zaakceptuj **zakres danych** – dane poprawne:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

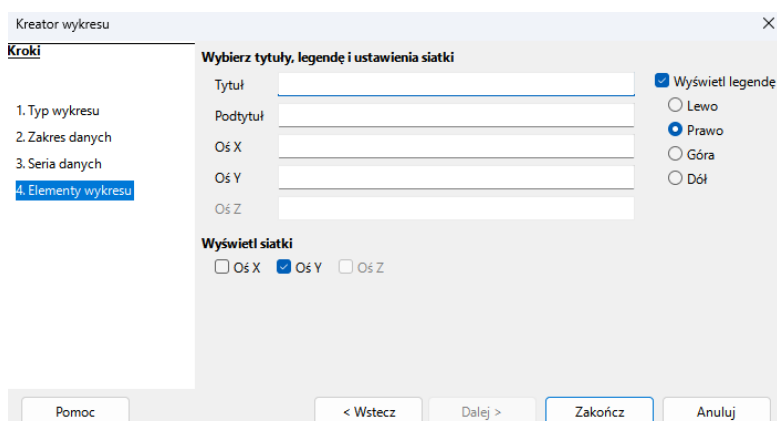


Przejdź dalej. W kolejnym kroku zaakceptuj **serię danych** – dane poprawne:



- upewnij się, że:
 - X = **Czas**
 - Y = **Odległość**

5. Elementy wykresu

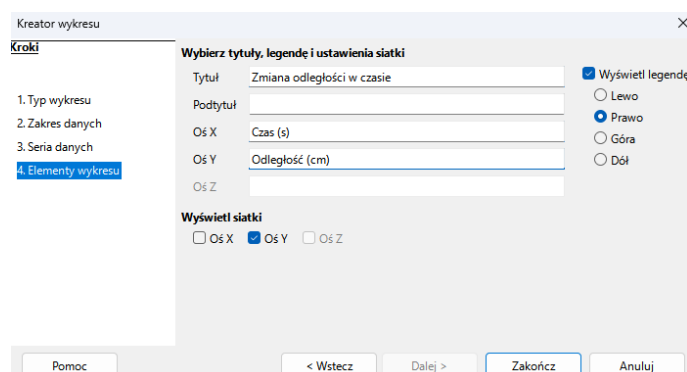


Dodaj:



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- oś X: **Czas (s)**
- oś Y: **Odległość (cm)**
- tytuł: np. *Zmiana odległości w czasie*



Zatwierdź klikając na przycisk **Zakończ**.

6. Na podstawie wykresu opisz, jak zmienia się odległość. Wskaż moment, w którym robot powinien się zatrzymać (≤ 15 cm).

7. Na podstawie danych i wykresu zaproponuj wartość odległości, przy której robot powinien reagować. Uzasadnij odpowiedź.

8. Dodaj kolumnę: „Czy robot powinien się zatrzymać?”. Wpisz TAK/NIE na podstawie warunku (≤ 15 cm). Policz, ile razy warunek został spełniony.

Podpowiedź:

	A	B	C	D	E
1	Pomiar	Czas (s)	Odległość (cm)	Czy robot powinien się zatrzymać?	
2	1	0,5	40	nie	
3	2	1	35	nie	
4	3	1,5	28	nie	
5	4	2	20	nie	
6	5	2,5	15	tak	
7	6	3	10	tak	
8					6 W x 1 K
9					
10	Średnia odległość		24,666667		
11	Minimalna odległość		10		
12	Maksymalna odległość		40		
13					
14	Różnica między największą a najmniejszą wartością		30		
15					
16	Ile razy robot powinien się zatrzymać?			=licz.jeżeli(D2:D7;"tak")	

Zadania do samodzielnego wykonania

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

1. Zmodyfikuj arkusz z pierwszego zadania, aby pojawiły się nowe kolumny o nazwie: Rzeczywista odległość, Zmierzona odległość (odczyt sensora), Błąd pomiaru.

	A	B	C	D	E
1	Pomiar	Czas (s)	Rzeczywista odległość (cm)	Zmierzona odległość (cm)	Błąd pomiaru
2	1	0,5	40	40,5	
3	2	1	35	34,5	
4	3	1,5	28	27,7	
5	4	2	20	19,8	
6	5	2,5	15	14,7	
7	6	3	10	10,3	

2. Oblicz błąd pomiaru jako: rzeczywista odległość – zmierzona odległość (odczyt sensora).

	A	B	C	D	E
1	Pomiar	Czas (s)	Rzeczywista odległość (cm)	Zmierzona odległość (cm)	Błąd pomiaru
2	1	0,5	40	40,5	=C2-D2
3	2	1	35	34,5	
4	3	1,5	28	27,7	
5	4	2	20	19,8	
6	5	2,5	15	14,7	
7	6	3	10	10,3	

Dla pozostałych pomiarów przeciągnij formułę. Zaznacz komórkę E2, następnie „złap” za niebieski kwadracik i przeciągnij do pomiaru 6 czyli komórki E7.

	E
n)	Błąd pomiaru
	-0,5

Poprawny wynik błędu dla danych wynosi:

E
Błąd pomiaru
-0,5
0,5
0,3
0,2
0,3
-0,3

4. Policz średni błąd pomiaru oraz największy błąd pomiaru.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

D	E
Zmierzona odległość (cm)	Błąd pomiaru
40,5	-0,5
34,5	0,5
27,7	0,3
19,8	0,2
14,7	0,3
10,3	-0,3
Średni błąd pomiaru	=ŚREDNIA(E2:E7)

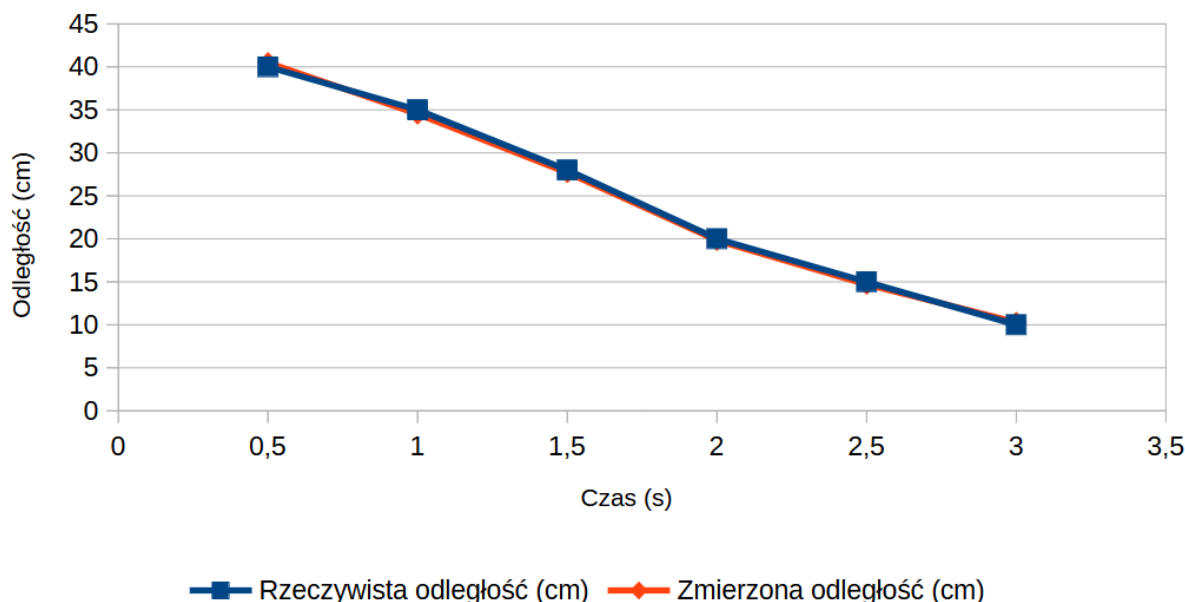
Największy błąd pomiaru:

D	E
Zmierzona odległość (cm)	Błąd pomiaru
40,5	-0,5
34,5	0,5
27,7	0,3
19,8	0,2
14,7	0,3
10,3	-0,3
Średni błąd pomiaru	0,083333333333
Największy błąd pomiaru	0,5

5. Utwórz wykres przedstawiający rzeczywistą odległość i zmierzoną odległość.
Wskazówka: na jednym wykresie dodaj dwie serie danych.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Porównanie pomiarów



Podsumowanie i wnioski z lekcji

Ćwiczenie pokazało, że testowanie sensorów robota oraz analiza zebranych danych w arkuszu kalkulacyjnym pozwala lepiej zrozumieć działanie urządzenia i jego dokładność. Uczniowie nauczyli się, jak poprawnie zbierać dane pomiarowe, zapisywać je w tabeli oraz wykorzystywać funkcje do ich analizy.

Tworzenie wykresu umożliwiło wizualne przedstawienie zmian odległości w czasie, co ułatwiło dostrzeżenie zależności i ewentualnych nieprawidłowości w działaniu sensora. Dzięki temu uczniowie mogli ocenić, czy czujnik działa stabilnie oraz czy jego wskazania są wiarygodne.

Wnioskiem z ćwiczenia jest to, że arkusz kalkulacyjny stanowi ważne narzędzie wspierające analizę danych, a wykresy pomagają w ich interpretacji. Umiejętność łączenia testów praktycznych z analizą danych jest kluczowa w procesie programowania i doskonalenia działania robota.

Zadanie domowe

1. Przeprowadź samodzielnie test czujnika:

- wykonaj kilka pomiarów odległości (np. co 0,5 sekundy lub przy różnych ustawieniach),
- zapisz wyniki.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

2. **Wprowadź dane do arkusza kalkulacyjnego.** Nazwij kolumny: pomiar, odległość (cm), czas (s).
3. Oblicz średnią odległość, minimalną wartość, maksymalną wartość.
4. Utwórz wykres, aby zaobserwować zmiany w czasie. Przyjmij, że:
 - oś X → czas,
 - oś Y → odległość.
5. Na podstawie tabeli i wykresu odpowiedz:
 - Czy pomiary są stabilne?
 - Czy wartości zmieniają się płynnie?
6. Zmodyfikuj program robota tak, aby zatrzymywał się nie przy **15 cm**, lecz przy 20 cm lub 10 cm. Następnie wykonaj ponowny test, wpisz nowe dane do tabeli i porównaj oba wykresy.
7. Jeżeli robot posiada więcej niż jeden czujnik wykonaj pomiary dwoma sensorami, zapisz wyniki w dwóch kolumnach i wykonaj jeden wspólny wykres.

Zadania dodatkowe

1. Przeprowadź samodzielnie test czujnika:
 - wykonaj kilka pomiarów odległości (np. co 0,5 sekundy lub przy różnych ustawieniach),
 - zapisz wyniki.
2. Wprowadź dane do arkusza kalkulacyjnego. Nazwij kolumny: pomiar, odległość (cm), czas (s).
3. Oblicz średnią odległość, minimalną wartość, maksymalną wartość.
4. Utwórz wykres, aby zaobserwować zmiany w czasie. Przyjmij, że:
 - oś X → czas,
 - oś Y → odległość.
5. Na podstawie tabeli i wykresu odpowiedz:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Czy pomiary są stabilne?
Czy wartości zmieniają się płynnie?

6. Zmodyfikuj program robota tak, aby zatrzymywał się nie przy 15 cm, lecz przy 20 cm lub 10 cm. Następnie wykonaj ponowny test, wpisz nowe dane do tabeli i porównaj oba wykresy.

7. Jeżeli robot posiada więcej niż jeden czujnik wykonaj pomiary dwoma sensorami, zapisz wyniki w dwóch kolumnach i wykonaj jeden wspólny wykres.

8. Zmodyfikuj swoje tabele dodając formuły, które automatycznie zweryfikują działanie robota. Dodaj nową kolumnę: Powinno być, Poprawność działania. Zastosuj zasadę:

- jeśli odległość $\leq 15 \rightarrow$ "stop"

- jeśli odległość $> 15 \rightarrow$ "jedzie"

Wykorzystaj formuły: =JEŻELI(B2<=15;"stop";"jedzie")

W kolumnie „poprawność działania” porównaj:

- „Reakcja robota”

- „Powinno być”

Wykorzystaj Formułę: =JEŻELI(C2=D2;"TAK";"BŁĄD")

9. Udostępnij dokument członkom zespołu i poproś ich o sprawdzenie poprawności struktury, zwracając uwagę na możliwość przypadkowej edycji lub usunięcia treści oraz zaproponuj zabezpieczenia (np. wersjonowanie dokumentu, komentarze zamiast edycji).



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 5: Odkrywamy tajemnice Kodu... Odczytać i zrozumieć Matrix

Temat lekcji nr 17 - Potęga eksperymentu. Testujemy roboty dla uzyskania najlepszych efektów podczas ruchu.

Treści nauczania (wymagania szczegółowe - podstawa programowa) – I. 3); I. 5); IV. 1); II. 1);

Cel lekcji:

- **Wprowadzenie do testowania ruchu robota** – uczniowie poznają znaczenie eksperymentowania z parametrami ruchu robota w języku Python oraz obserwują wpływ zmian w kodzie na zachowanie robota.
- **Analiza działania programu** – uczniowie analizują fragmenty kodu odpowiedzialne za ruch robota oraz określają, w jaki sposób zmiana parametrów prędkości, czasu ruchu i skrętu wpływa na efekty działania programu.
- **Tworzenie i modyfikowanie programów sterujących ruchem robota** – uczniowie opracowują algorytmy oraz zapisują programy umożliwiające testowanie różnych sposobów poruszania się robota i porównywanie uzyskanych efektów.
- **Testowanie działania programu** – uczniowie uruchamiają program, obserwują ruch robota, analizują wyniki eksperymentów oraz wprowadzają poprawki w celu uzyskania dokładniejszego i bardziej płynnego ruchu robota.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- **Uczeń dostrzega, że działanie programu i ruch robota zależy od poprawności zapisanych poleceń:**
 - rozumie, że robot wykonuje ruch zgodnie z zapisanym kodem programu oraz zastosowanymi parametrami ruchu,
 - obserwuje działanie programu napisanego w języku Python i formułuje wnioski dotyczące wpływu parametrów ruchu na zachowanie robota.
- **Uczeń analizuje działanie programu i jego efekty:**
 - sprawdza poprawność działania programu sterującego ruchem robota



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- identyfikuje błędy związane z parametrami ruchu, prędkością lub skretem i modyfikuje program w celu poprawy działania robota,
 - analizuje zależność między zapisanym kodem a efektem ruchu robota podczas wykonywania testów.
- **Uczeń rozwija umiejętność testowania i doskonalenia rozwiązań:**
 - wykonuje próby działania programu i porównuje wyniki ruchu robota przy różnych parametrach,
 - wprowadza poprawki w celu uzyskania płynniejszego, dokładniejszego i bardziej przewidywalnego ruchu robota,
 - testuje różne rozwiązania programistyczne oraz wyciąga wnioski z przeprowadzonych eksperymentów.

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści kształcenia będą realizowane za pomocą robotyki edukacyjnej LEGO® Education SPIKE™ Prime.

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna RaP STEAM,
- praca na komputerze,
- praca przy zestawie robotyki edukacyjnej,
- pogadanka.

Środki dydaktyczne

- zestaw PC z dostępem do Internetu,
- zestaw robotyki edukacyjnej LEGO® Education SPIKE™ Prime oraz zestaw rozszerzony LEGO® Education SPIKE™ Prime Expansion Kit,
- rzutnik multimedialny,
- e-podręcznik.

Formy pracy

- Praca indywidualna lub grupowa.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przebieg lekcji:

Faza I. Wprowadzenie

Na początku lekcji nauczyciel wprowadza uczniów w zagadnienia związane z **testowaniem programów sterujących ruchem robota w języku Python, wyjaśniając, że nawet niewielka zmiana parametrów ruchu może znacząco wpłynąć na zachowanie robota**. Następnie odnosi się do zadania lekcyjnego, polegającego na **przeprowadzaniu eksperymentów związanych z ruchem robota** oraz analizowaniu uzyskanych efektów w środowisku LEGO SPIKE Prime.

Nauczyciel przedstawia cel i zakres zadania. Prezentuje uczniom środowisko programowania w języku Python w aplikacji LEGO SPIKE App oraz omawia parametry odpowiedzialne za ruch robota, takie jak prędkość, czas ruchu, liczba stopni obrotu silników oraz siła skrętu. Następnie przekazuje uczniom niezbędne informacje do realizacji lekcji.

Uczniowie zapoznają się ze sposobami modyfikowania parametrów ruchu robota oraz **obserwują, jak zmiany w programie wpływają na sposób poruszania się robota po uruchomieniu programu**.

Następnie uczniowie zdobywają umiejętności obejmujące:

- określenie problemu (w jaki sposób dobrać odpowiednie parametry ruchu robota),
- analizę działania programu sterującego ruchem robota,
- obserwację efektów działania programu po uruchomieniu robota,
- porównywanie wyników działania robota dla różnych parametrów ruchu,
- testowanie programu i wprowadzanie poprawek związanych z ruchem robota,
- wyciąganie wniosków z przeprowadzonych eksperymentów.

Na zakończenie tej fazy lekcji nauczyciel podaje temat do zapisania w zeszycie i przechodzi do fazy II – realizacji praktycznej.

Faza II. Część zasadnicza - realizacja

W tej fazie lekcji nauczyciel zapewnia uczniom praktyczne ćwiczenia, które pozwolą eksperymentować z koncepcjami programowania. W szczególności obejmować to będzie używanie blokowych języków programowania oraz zestawu do robotyki.

1. Nauczyciel objaśnia sposób pracy na lekcji – we wskazanym przez niego momencie uczniowie będą wykonywać kolejne etapy prowadzące do rozwiązania zadania. Zdolniejsi uczniowie spróbują wykonać je samodzielnie. Słabsi wykonają kolejne kroki razem z nauczycielem, w oparciu o przygotowaną do zajęć instrukcję dostępną w e-podręczniku lub materiał multimedialny. W razie potrzeby będą mogli dopytać nauczyciela. Nauczyciel na bieżąco sprawdza postępy uczniów.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

2. Bazując na posiadanych zasobach, zapewnij każdemu uczniowi lub zespołowi uczniów kompletny zestaw do robotyki (tj. LEGO® Education SPIKE™).
3. Zestawy te pozwalają uczniom budować własne roboty i programować je do wykonywania określonych zadań. Nauczyciel zaczyna od prostych projektów, takich jak zbudowanie i zaprogramowanie robota, który może wykonać proste czynności, które wprowadzą robota w ruch.
4. Uczniowie budują bazowego robota według załączonej instrukcji (załącznik **Budowa_bazy_robota.pdf** do lekcji).
5. Uczniowie włączają program **SPIKE**, poprzez który będą programować robota.
6. Nauczyciel, wykorzystując e-platformę, demonstruje na tablicy multimedialnej działanie programu i robota, uruchamiając odpowiedni moduł i materiał multimedialny.
7. Prezentowany materiał multimedialny, zrealizowany w oparciu o szczegółowy scenariusz (załącznik do lekcji), oprócz waloru teoretycznego stanowi również podstawę przebiegu części praktycznej lekcji. W szczególności, uczniowie:
 - a. budują prosty model bazowego robota w oparciu o zestaw do robotyki (tj. LEGO® Education SPIKE™),
 - b. tworzą listę czynności niezbędnych do wprowadzenia robota w ruch,
 - c. na podstawie listy czynności przygotowują program, zapisują go we wskazanym folderze (nauczyciel wskazuje właściwą lokalizację; może to być dysk lokalny lub zasób w chmurze) i potrafią go uruchomić;
 - d. wgrywają program do zbudowanego modelu robota, uruchamiają, obserwują i testują działanie robota. Wprowadzają modyfikacje w programie, poprawiają ewentualne błędy.

Faza III. Podsumowanie

Uczniowie analizują przedstawione przykłady programów sterujących ruchem robota zapisanych w języku Python oraz omawiają ich działanie. Nauczyciel ocenia stopień zrozumienia zależności między zapisanym kodem a sposobem poruszania się robota po uruchomieniu programu.

Kryteria oceniania:

Ocena dopuszczająca:

- uczeń rozpoznaje podstawowe elementy programu w języku Python,
- potrafi wskazać polecenia odpowiedzialne za ruch robota.

Ocena dostateczna:

- uczeń analizuje przykładowy program i opisuje jego działanie,
- rozumie zależność między zapisanym kodem a działaniem programu,
- rozumie wpływ podstawowych parametrów ruchu na sposób poruszania się robota,
- wskazuje podstawowe błędy składniowe.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Ocena dobra:

- uczeń samodzielnie modyfikuje przykładowy program,
- dobiera i zmienia parametry ruchu robota w celu poprawy jego działania,
- poprawia błędy związane z zapisem poleceń, nawiasami lub cudzysłowami,
- testuje działanie programu po wprowadzonych zmianach.

Ocena bardzo dobra:

- uczeń samodzielnie tworzy poprawny program w języku Python,
- wyjaśnia działanie poszczególnych elementów programu,
- analizuje wyniki eksperymentów związanych z ruchem robota i wyciąga wnioski,
- proponuje własne rozwiązania i potrafi je uzasadnić,
- samodzielnie analizuje i poprawia błędy w kodzie.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, np.:

1. Co się stanie, gdy zwiększymy wartość **velocity**?
 - a. **Robot będzie poruszał się szybciej**
 - b. Robot zmieni kierunek
 - c. Robot zatrzyma się
 - d. Robot skręci w miejscu
2. Dlaczego stosujemy **acceleration** w programie?
 - a. Aby robot jechał szybciej
 - b. Aby robot skręcał
 - c. **Aby robot ruszał płynnie i nie ślizgał się**
 - d. Aby robot się zatrzymał
3. Co robi funkcja **move_tank_for_degrees()**?
 - a. Steruje tylko jednym silnikiem
 - b. **Steruje osobno prędkością lewego i prawego silnika**
 - c. Zatrzymuje robota
 - d. Wyświetla tekst

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Faza V. Dodatkowe, praca domowa (dla chętnych).

Zadania dodatkowe (dla chętnych).

Zadanie 1: Eksperymentujemy z ruchem robota

Utwórz program w języku Python, który pozwoli przetestować różne parametry ruchu robota LEGO SPIKE Prime.

Program powinien:

- zawierać polecenia odpowiedzialne za ruch robota do przodu, do tyłu lub wykonywanie skrętów,
- wykorzystywać różne wartości prędkości lub długości ruchu,
- być zapisany poprawnie pod względem składni,
- umożliwiać obserwację wpływu parametrów na ruch robota,
- mieć logiczną kolejność wykonywania poleceń.

Opcjonalnie możesz rozbudować program tak, aby:

- robot wykonywał kilka różnych prób ruchu,
- program porównywał różne prędkości jazdy lub wartości skrętu,
- robot zatrzymywał się pomiędzy kolejnymi testami,
- program wyświetlał komunikaty informujące o wykonywanym eksperymencie,
- kod był estetycznie zapisany i łatwy do odczytania.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 17 - *Potęga eksperymentu. Testujemy roboty dla uzyskania najlepszych efektów podczas ruchu.*

Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
 - Nauczyciel omawia podstawowe zagadnienia związane z **testowaniem programów sterujących ruchem robota w języku Python**:
 - wyjaśnia, że robot wykonuje ruch zgodnie z instrukcjami zapisanymi w programie,
 - przedstawia przykłady poleceń odpowiedzialnych za ruch robota oraz zmianę parametrów jazdy,
 - omawia znaczenie poprawnego doboru parametrów ruchu i testowania działania programu.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- Praca uczniów:
 - analiza przedstawionych przykładów programów sterujących ruchem robota,
 - omawianie działania poszczególnych linijek kodu,
 - wykonywanie prób ruchu robota przy różnych parametrach,
 - wyszukiwanie i poprawianie błędów związanych z ruchem robota oraz parametrami jazdy,
 - proponowanie własnych sposobów poprawy płynności i dokładności ruchu robota.
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów (**analiza poprawności działania programu, sposobu poruszania się robota oraz umiejętności testowania i poprawiania parametrów ruchu**).
- Zadanie pracy domowej (opcjonalnie) – **napisanie programu testującego różne parametry ruchu robota oraz opisanie wpływu zmian parametrów na sposób poruszania się robota**.

Załącznik nr 2 – Realizacja zadania głównego podczas lekcji nr 17 - *Potęga eksperymentu. Testujemy roboty dla uzyskania najlepszych efektów podczas ruchu.*

Zadanie 1. Przygotowanie do realizacji zadania głównego

Uczniowie, pracując w zespołach lub indywidualnie, realizują zadanie polegające na **testowaniu programów sterujących ruchem robota w środowisku LEGO SPIKE Prime oraz analizowaniu wpływu parametrów ruchu na zachowanie robota**. Szczególny nacisk położony jest na **poprawne dobieranie parametrów ruchu, analizę działania programu oraz wyciąganie wniosków z przeprowadzonych eksperymentów**.

Uczniowie opracowują i zapisują rozwiązanie problemu w postaci programów sterujących ruchem robota, które umożliwią **porównywanie różnych sposobów poruszania się robota i analizowanie efektów działania programu**. W tym celu:

- analizują przykładowe programy zapisane w języku Python,
- określają cel programu oraz przewidują możliwe problemy związane z ruchem robota (np. zbyt szybka jazda, niedokładny skręt lub zatrzymanie robota w niewłaściwym miejscu),
- planują kolejność wykonywania instrukcji w programie,
- opracowują algorytm w uporządkowanej formie,
- zapisują go jako program sterujący ruchem robota w języku Python,
- testują działanie programu i analizują efekty ruchu robota,
- wprowadzają poprawki związane z parametrami ruchu i działaniem programu,
- doprecyzowują rozwiązanie tak, aby ruch robota był płynny, dokładny i przewidywalny.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

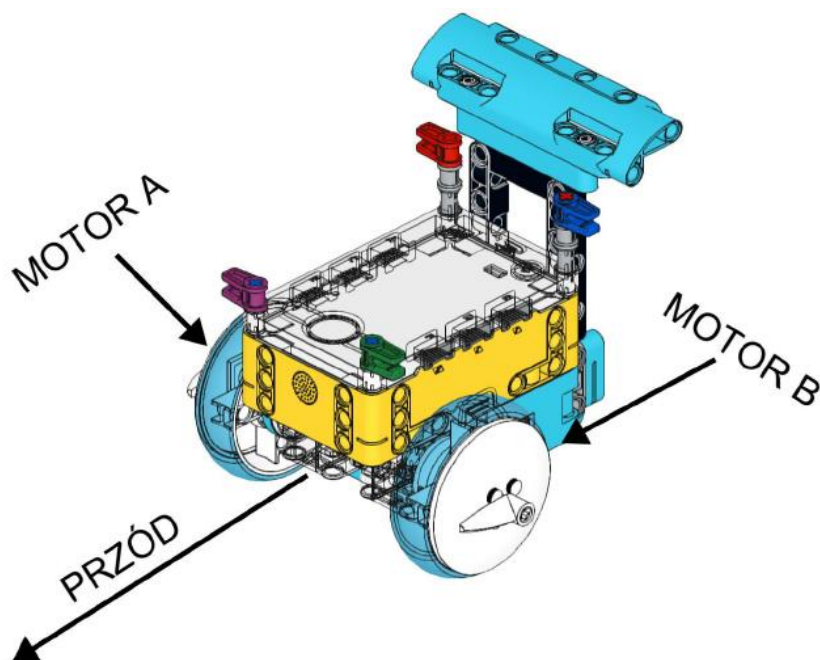
Uczniowie wykorzystują środowisko LEGO SPIKE Prime w trybie Python, zwracając uwagę na zgodność między zapisanym kodem a wynikiem działania programu widocznym w konsoli.

Jeśli jest taka potrzeba, wskazana przez nauczyciela, to uczniowie zbudują robota od podstaw zgodnie z poniższą instrukcją.

Budowa bazowego robota

Uruchom plik **Budowa bazy robota.pdf** znajdujący się na platformie i postępuj wg podanych w pliku kroków. Efektem końcowym będzie fizyczny robot gotowy do programowania. **Jeśli konstrukcja jest gotowa – idź dalej.**

Pudełka niezbędne do realizacji lekcji: **Tylko konstrukcja bazowa**



Dodatkowe instrukcje niezbędne do realizacji lekcji:

- **Nie dotyczy**
- Do realizacji scenariusza zajęć będziesz potrzebował również matę:



Fundusze Europejskie
dla Podkarpacia



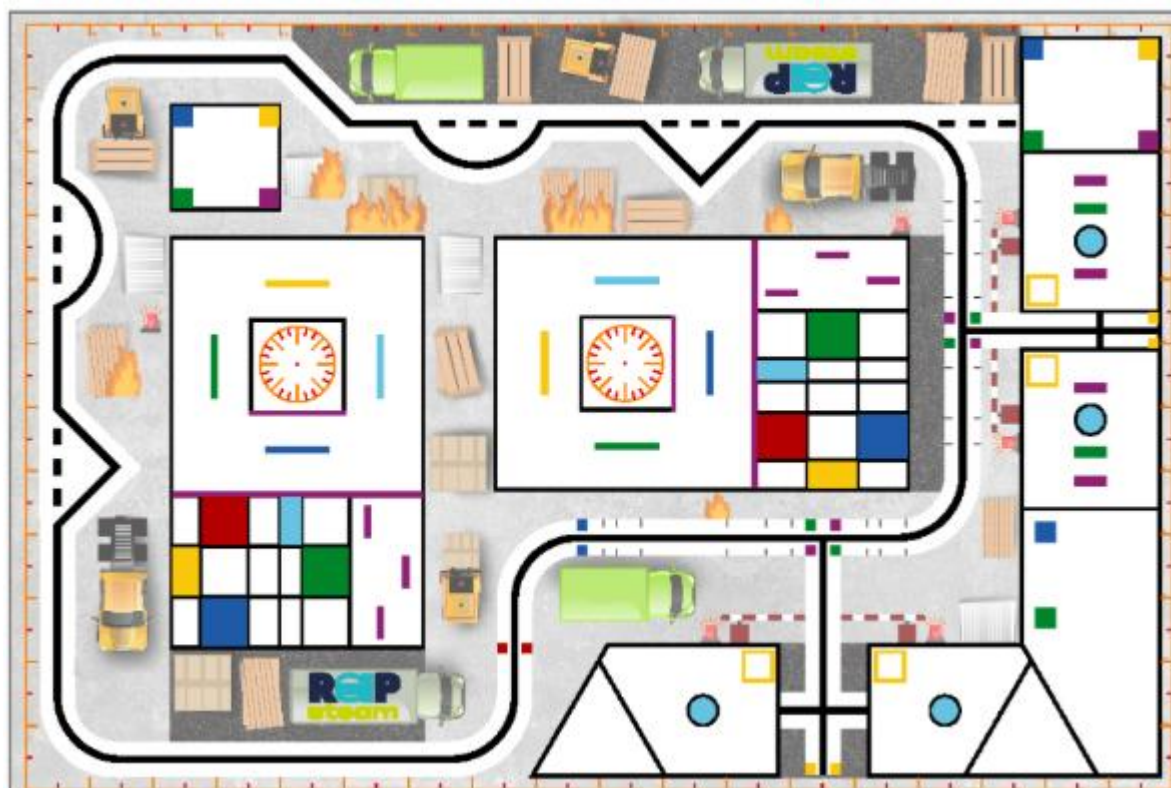
Rzeczpospolita
Polska

Dofinansowane przez
Unię Europejską



PODKARPACKIE
przestrzeń otwarta

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Podkarpacki Zespół
Placówek Wojewódzkich
w Rzeszowie



POLITECHNIKA
RZESZOWSKA
Im. IGNACEGO ŁUKASIEWICZA

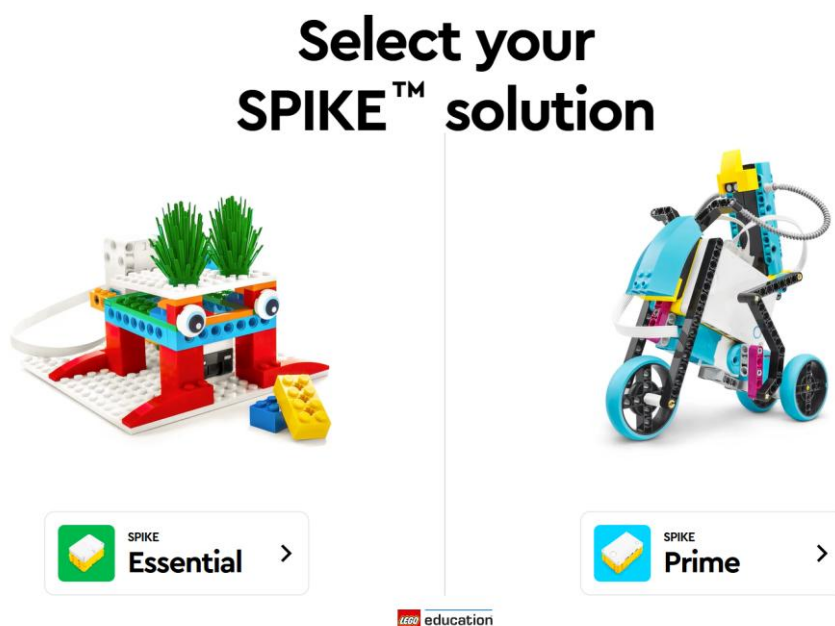


Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 2. Konfiguracja środowiska.

Uruchamianie aplikacji LEGO Spike

1. Włącz przeglądarkę internetową dostępną na twoim urządzeniu
2. Wpisz w przeglądarce adres: <https://spike.legoeducation.com/> Pojawi się okno wyboru wersji LEGO SPIKE. Wybierz Spike Prime

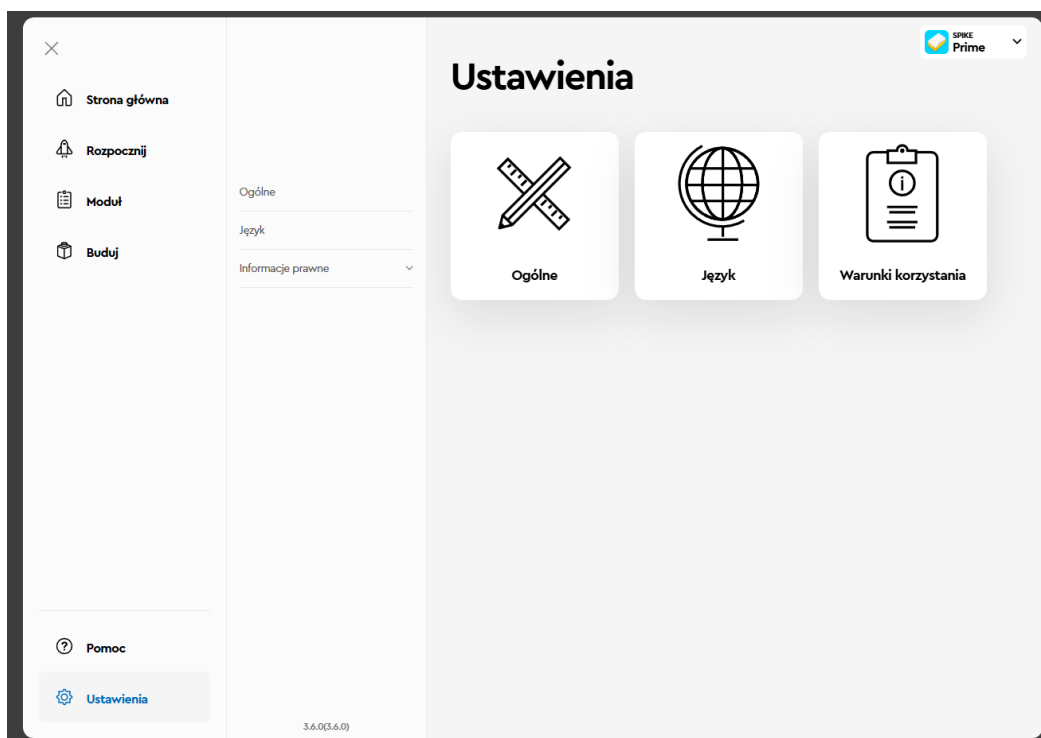


3. Po wybraniu opcji SPIKE Prime pojawia się menu główne. Jeśli występuje język polski, należy zmienić go na język angielski.

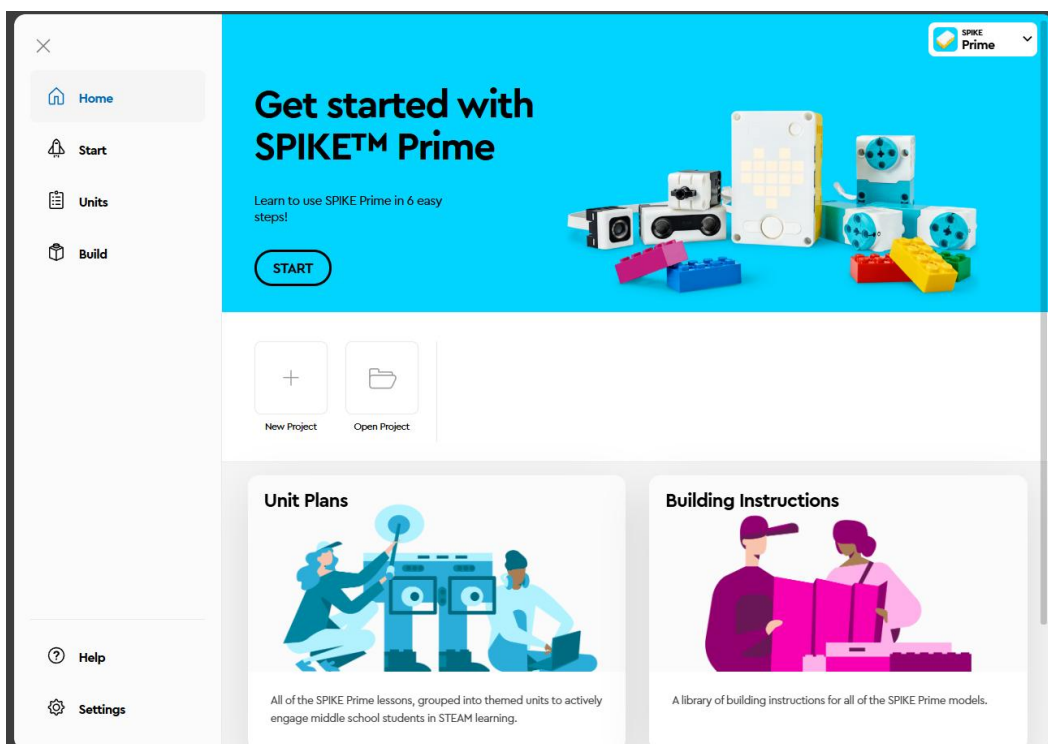
Jeżeli zaś wszystko jest w **języku angielskim**, to przejdź do kroku 6.

Aby zmienić język, w lewym dolnym rogu menu głównego kliknij przycisk **Ustawienia**.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

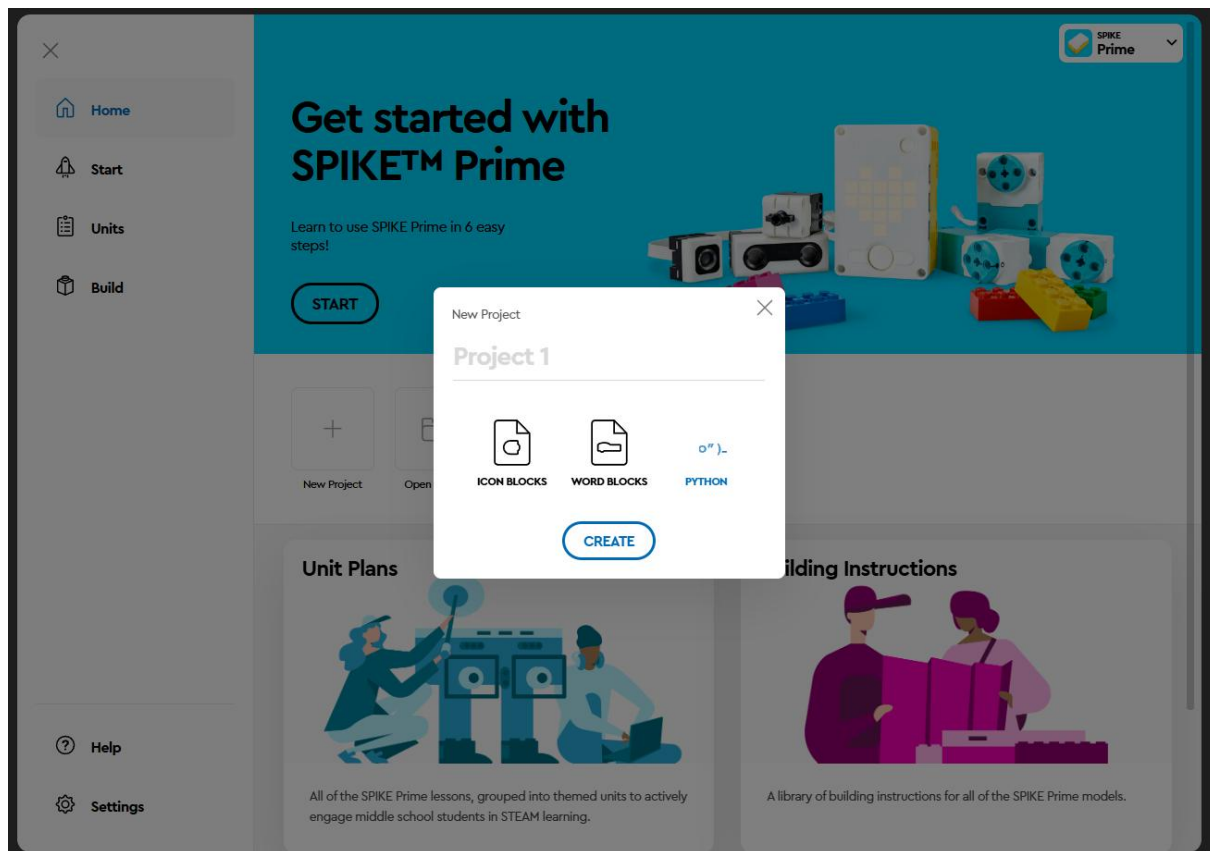


4. W nowym oknie kliknij globus podpisany „Language”.
5. Wybierz język: **United States (English)**. Strona przeładuje się i język angielski będzie dostępny.
6. Kliknij po lewej przycisk **Home**. Pojawi się okno jak poniżej

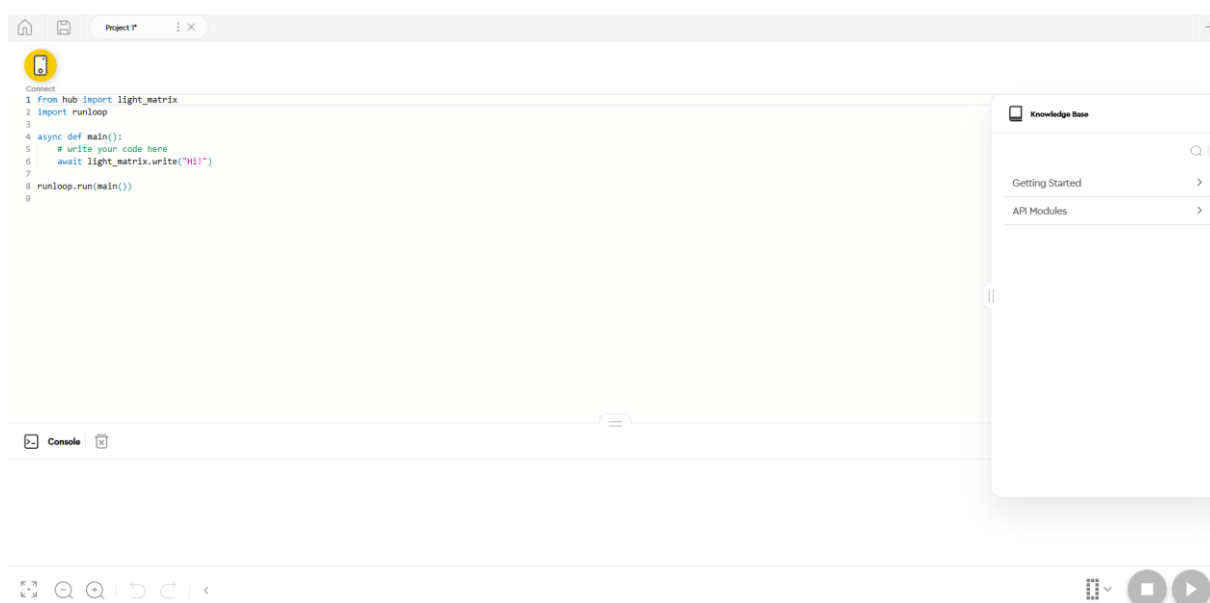


Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

7. Kliknij na środku przycisk **New Project**. Pojawi się nowe okno jak poniżej



8. Wybierz opcję **Python** oraz podaj nazwę projektu. Kiedy wszystko zostanie poprawnie wybrane, wciśnij przycisk **Create**.

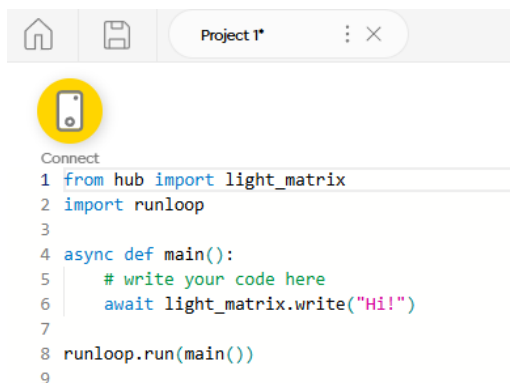


Wszystko jest gotowe do programowania. Pora na podłączenie kontrolera Spike Hub.

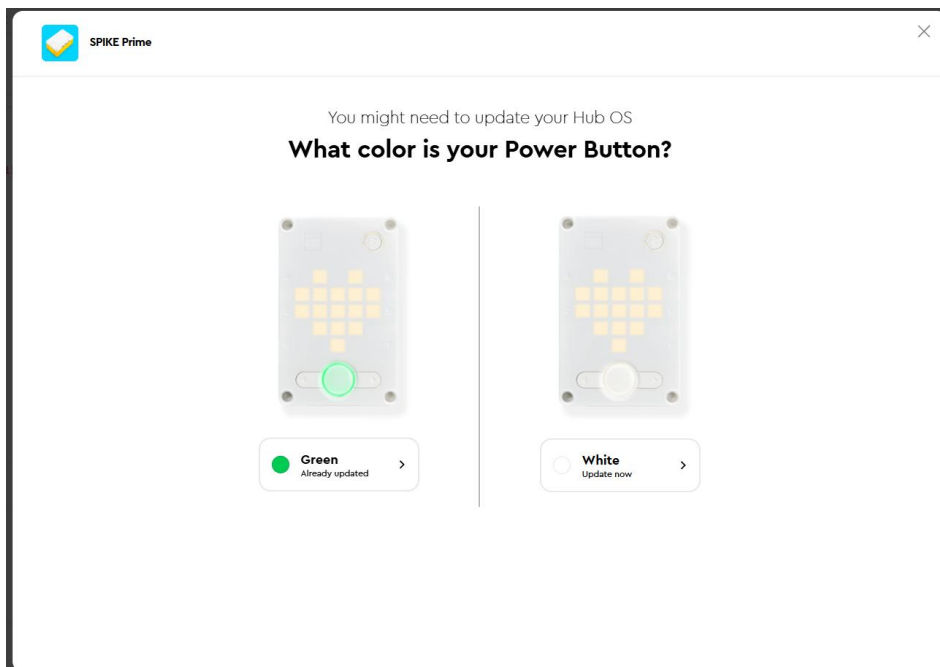
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 3. Podłączenie kontrolera Spike Hub

1. W głównym programie kliknij przycisk **Connect** znajdujący się w górnym rogu, tuż pod nazwą twojego projektu. Patrz poniżej.

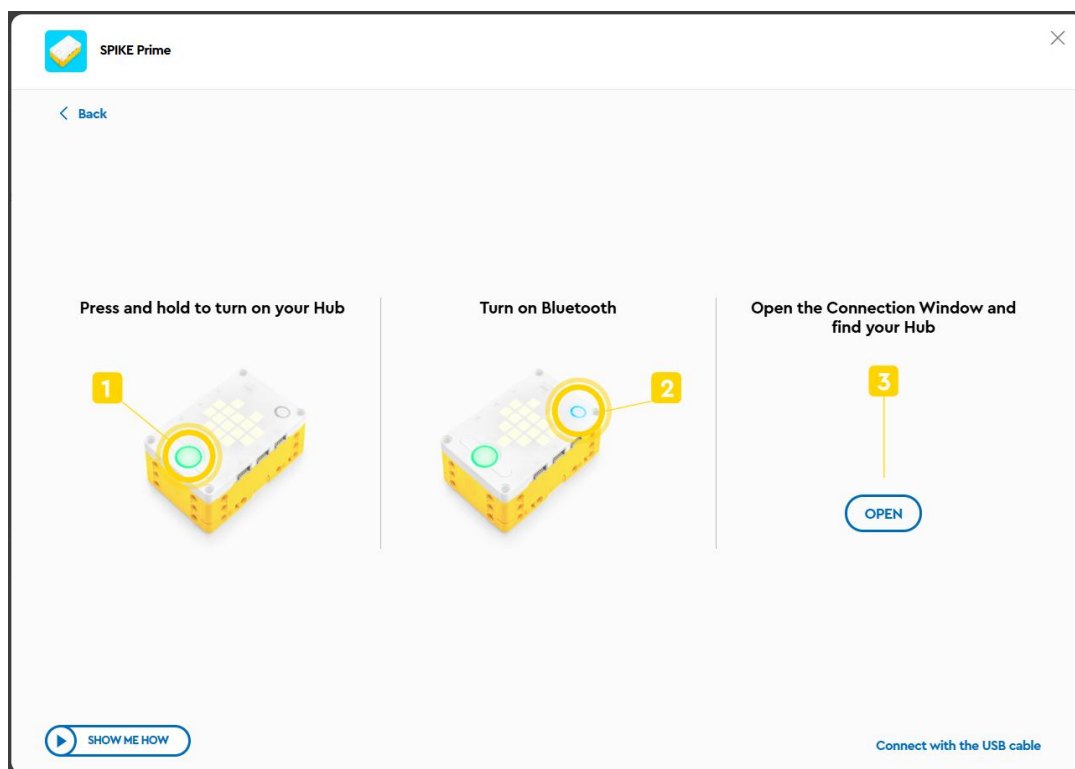


2. Pojawi się nowe okno. **Włącz kontroler Spike Hub** klikając **duży biały przycisk** i poczekaj aż się uruchomi. Następnie wybierz jedną z opcji odpowiadającą stanowi kontrolera Spike Hub – zawsze będzie to **Green – Already updated**.

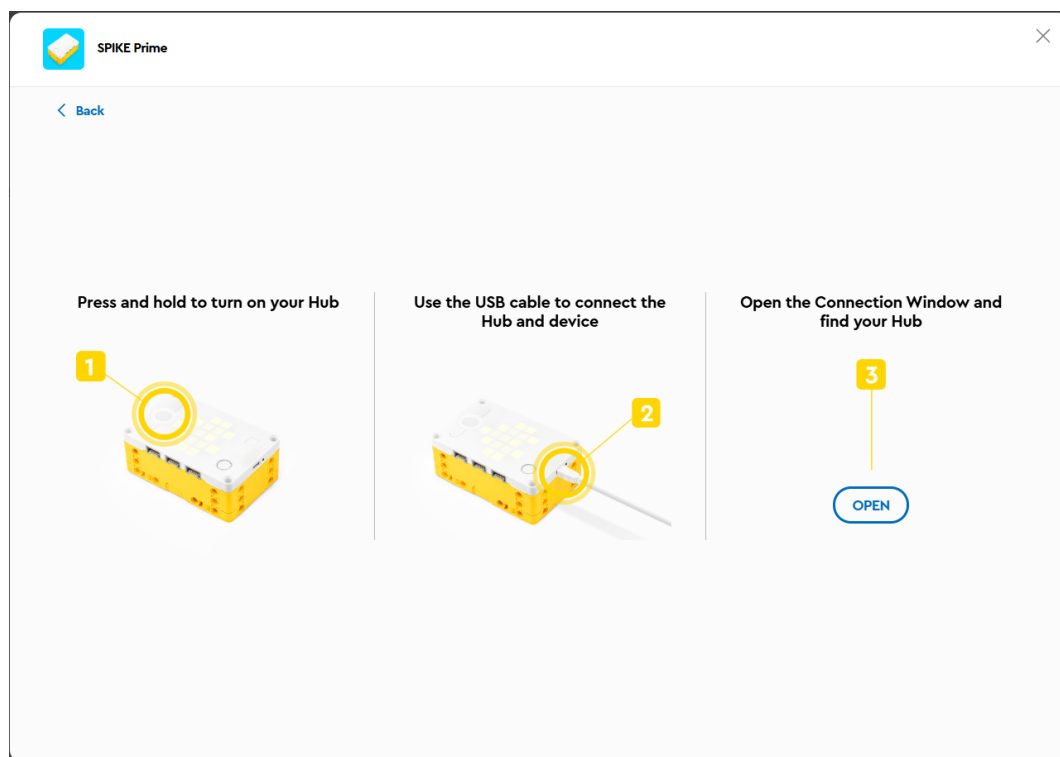


3. Pojawi się nowe okno (patrz poniżej). Wybierz opcję **Connect with USB cable** znajdującą się w prawym dolnym rogu okna.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

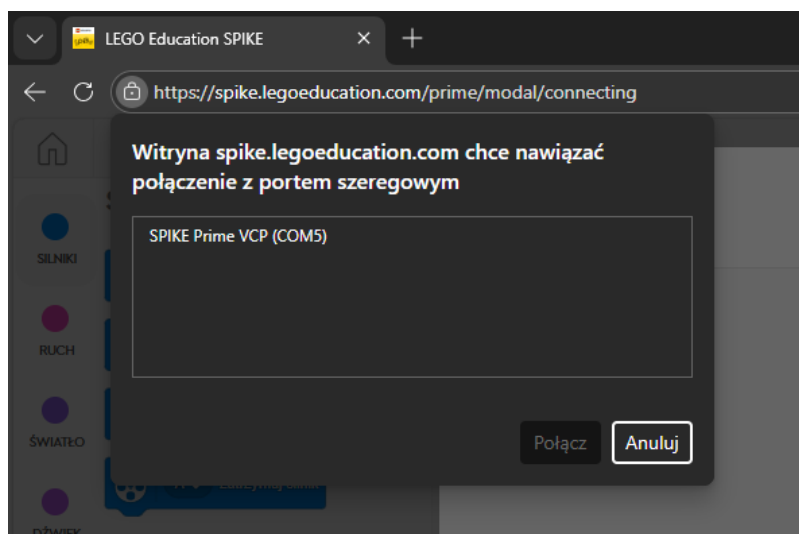


4. Postępuj zgodnie z instrukcją na ekranie. Po przeczytaniu kliknij przycisk **Open**.



5. Pojawi się nowe okno (patrz poniżej)

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



6. Wybierz opcję **SPIKE Prime VCP** i kliknij **Połącz**.

UWAGA – u Ciebie nazwa może nie być identyczna jak na rysunku powyżej. Wybierz tę opcję, która ma w nazwie Spike Prime. Innymi rzeczami się nie przejmuj.

Jeśli wszystko zostanie poprawnie połączone, to okna zostaną automatycznie zamknięte, a ikona kontrolera w programie zmieni się na zieloną.



```
1 from hub import light_matrix
2 import runloop
3
4 async def main():
5     # write your code here
6     await light_matrix.write("Hi!")
7
8 runloop.run(main())
9
```

Gratulacje. Twój Spike Hub jest podłączony do komputera. Możesz przystąpić do programowania robota.

W przypadku problemów postępuj wg kroków:

1. Sprawdź, czy kontroler Spike Hub jest włączony i świeci na zielono
 - a. Jeśli kontroler świeci na biało zgłoś to nauczycielowi
 - b. Jeśli kontroler świeci na zielono to podążaj dalej wg instrukcji łączenia powyżej
2. Sprawdź czy kabel USB jest podłączony (Spike Hub <-> Komputer)
3. Sprawdź, czy wybrano odpowiednią nazwę urządzenia – w oknie połączenia z portem szeregowym należy wybrać urządzenie, które w nazwie ma SPIKE Prime.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 4. Realizacja zadania głównego

W tej lekcji nauczysz się stosowania zmiennych parametrów podczas obsługi silników podpiętych do kontrolera Spike Hub. Poznasz metody przyspieszania, zwalniania i sterowania robotem w dwóch trybach. Ostatecznym zadaniem będzie:

- Jazda z różnymi parametrami prędkości
- Jazda w różnych kierunkach
- Jazda z różnymi parametrami przyspieszania
- Jazda z różnymi parametrami zwalniania
- Jazda w trybie normalnym i trybie czołgowym

Przykład realizacji zadania – wprowadzenie bibliotek

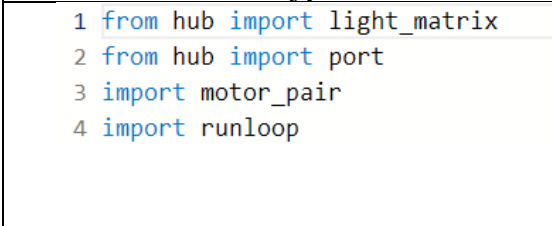
UWAGA – w materiałach stosowane są komentarze. Na zrzutach ekranu jest to tekst oznaczony jako zielony. Tego tekstu nie przepisyj! To jest tylko wytłumaczenie dla Ciebie, co dzieje się w danej części programu. Przepisuj tylko i wyłącznie tekst, który nie jest zielony. Korzystaj z tabeli. Pamiętaj, że wielkość liter MA ZNACZENIE!

Na początku programu należy dodać odpowiednie odwołania (biblioteki) do modułów kontrolera Spike Hub i czujników. Dodaj moduł, biblioteki:

- **port** (moduł),
- **light_matrix** (moduł),
- **motor_pair** (biblioteka),

Aby móc programować silniki niezbędne jest zaimportowanie biblioteki **motor_pair**. Pozwala ona, tak jak w Scratchu za pomocą różowych bloków sterować dwoma silnikami jednocześnie. Podczas dzisiejszych zajęć wykorzystasz też moduł **port** oraz **light_matrix** (dostęp do portów oraz wyświetlanie informacji na matrycy Spike Hub).

Zaktualizuj początek programu i dodaj odpowiednie odwołania/biblioteki. Część programu odpowiedzialna za biblioteki powinna wyglądać następująco:

Zdjęcie	Kod (tekst)
 <pre> 1 from hub import light_matrix 2 from hub import port 3 import motor_pair 4 import runloop </pre>	<pre> from hub import light_matrix from hub import port import motor_pair import runloop </pre>

W tej chwili cały program powinien wyglądać następująco:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

```
1 from hub import light_matrix
2 from hub import port
3 import motor_pair
4 import runloop
5
6 async def main():
7     # write your code here
8     await light_matrix.write("Hi!")
9
10 runloop.run(main())
```

Przygotowanie struktury programu

Dzisiejsze zajęcia zacznij od przygotowania podstawowych ustawień i parametrów dla kontrolera Spike Hub i silników. Dodaj komendę, która przypisze silniki podpięte do portów B i A do zmiennej PAIR_1 – tak jak na poprzedniej lekcji. Następnie uruchom silniki tak aby robot:

- Pojechał do przodu 360 stopni, bez skręcania
- Pojechał do tyłu 360 stopni, bez skręcania

Możesz zostawić komendę odpowiadającą za wyświetlanie napisu „Hi!” na kontrolerze Spike Hub. Jeśli chcesz ją usunąć, to możesz to zrobić. Nie jest ona wymagana do dzisiejszych zajęć. Program powinien wyglądać następująco:

```
1 from hub import light_matrix
2 from hub import port
3 import motor_pair
4 import runloop
5
6 async def main():
7     # write your code here
8     await light_matrix.write("Hi!")
9
10    #przypisanie portów A i B do PAIR_1
11    motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
12
13    #jazda do przodu bez skręcania (0) przez 1 obrót (360)
14    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 360, 0)
15
16    #jazda do tyłu bez skręcania (0) przez 1 obrót (360)
17    await motor_pair.move_for_degrees(motor_pair.PAIR_1, -360, 0)
18
19 runloop.run(main())
```

Uruchom program, wgrywając go do kontrolera Spike Hub, i sprawdź, czy robot wykonuje poprawnie swoje zadanie. Jeśli wszystko działa poprawnie, przejdź do dalszej części lekcji.

Faza 1 - jazda ze zmiennymi parametrami prędkości silników

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

W tej części lekcji nauczysz się sterować prędkością jazdy robota. Służy do tego parametr **velocity** każdej funkcji ruchu z biblioteki **motor_pair**. Zasada jest bardzo prosta. Im wyższa wartość parametru **velocity**, tym szybciej robot będzie się poruszał, zaś im mniejsza, tym wolniej. Parametr **velocity** ustala się poprzez użycie nazwy. Jest on parametrem opcjonalnym – to oznacza, że można go podać lub nie. Nie jest on konieczny do działania robota tak jak liczba stopni czy obrót. Jeśli **velocity** nie jest używane w kodzie, to domyślnie robot przyjmuje prędkość 1 obrotu na sekundę (360 stopni).

Pora na testy.

W dzisiejszej lekcji wykorzystasz matrycę Spike Huba do prezentacji aktualnego działania. Podświetlony piksel pozwoli Ci, patrząc na robota, określić, która część programu jest w danej chwili wykonywana. W przypadku silników zacznij od ruchu przód/tył. Wpisz następujące komendy i uruchom robota.

```
1 from hub import light_matrix
2 from hub import port
3 import motor_pair
4 import runloop
5
6 async def main():
7     # write your code here
8     await light_matrix.write("Hi!")
9
10    #przypisanie portów A i B do PAIR_1
11    motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
12
13    #jazda do przodu bez skręcania (0) przez 1 obrót (360)
14    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 360, 0)
15
16    #jazda do tyłu bez skręcania (0) przez 1 obrót (360)
17    await motor_pair.move_for_degrees(motor_pair.PAIR_1, -360, 0)
18
19    #faza 1 - testy parametrów prędkości. Dodatkowe wykorzystanie matrycy aby zrozumieć w której części kodu się jest
20    light_matrix.clear()
21    light_matrix.set_pixel(0,0,100)
22
23    #jazda do przodu bez skręcania (0) przez 3 obroty (1080 stopni) z prędkością 720
24    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1080, 0, velocity=720)
25
26    #jazda do tyłu bez skręcania (0) przez 3 obroty (-1080 stopni) z prędkością 1100
27    #zauważ że robot wraca do tej samej pozycji mimo innej prędkości
28    await motor_pair.move_for_degrees(motor_pair.PAIR_1, -1080, 0, velocity=1100)
29
30 runloop.run(main())
```

Robot najpierw wykonał pierwsze zadanie – poruszył się do przodu i do tyłu o jeden obrót ze stałą prędkością. W kolejnym kroku zapalił się pierwszy piksel na matrycy i robot zaczął się poruszać inaczej. Aby dobrze pokazać zasadę, trzymamy się w tej lekcji 3 obrotów – 1080 stopni ($360 \times 3 = 1080$). W kodzie pojawiły się parametry **velocity=720** i **velocity=1100**. Są to prędkości poruszania się. Zauważ, że robot mimo innych prędkości powrócił do tego samego miejsca, z którego startował. Dlaczego tak się dzieje? Ponieważ parametr prędkości nie wpływa na odległość. Robot przejedzie ten sam dystans (1080 stopni), jednak zrobi to szybciej lub wolniej.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Parametr **velocity** przyjmuje wartości z zakresu od -1100 do 1100 dla silników średnich oraz od -1050 do 1050 dla silników dużych. Wartość ujemna oznacza, że silnik będzie się kręcił w drugą stronę.

Zadajesz sobie pytanie, czy parametrem **velocity** można sterować kierunkiem? Odpowiedź brzmi: jak najbardziej tak. Możesz używać parametru **velocity** lub stopni, aby ustalić kierunek jazdy. Aby jednak odnaleźć się w swoim kodzie, zalecane jest, aby zmieniać tylko JEDEN parametr. W naszych lekcjach zawsze będziemy zmieniać tylko i wyłącznie stopnie bądź kierunek. Parametr **velocity** i wszystkie pozostałe zawsze będą dodatnie.

Faza 2 – testy przyspieszenia

Kolejnym parametrem, który poznasz na lekcji, jest **acceleration**, czyli przyspieszenie. W tym module zobaczysz, że robot może startować wolniej, a nawet się rozpędzać. Kiedy to się przydaje? Kiedy chcesz zachować precyzję. Robot, startując wolniej i przyspieszając w trakcie, nie poślizgnie się na początku. Działa to tak samo jak samochody w zimie. Na lodzie nigdy nie startuje się z pełnym gazem, tylko rusza powoli, aby zachować przyczepność. To samo dzieje się w tym przypadku. To do dzieła. Ustalmy wartość **velocity** na 720, dodajmy jednak dodatkowy parametr. Tak, można stosować obydwa w tej samej linii kodu. Pamiętaj o matrycy

😊 Listing poniżej przedstawia tylko kod funkcji **main()**.

NIE USUWAJ BIBLIOTEK. One zostają niezmienione, dlatego pomijamy je na listingu.

```

6 async def main():
7     # write your code here
8     await light_matrix.write("Hi!")
9
10    #przypisanie portów A i B do PAIR_1
11    motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
12
13    #jazda do przodu bez skręcania (0) przez 1 obrót (360)
14    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 360, 0)
15
16    #jazda do tyłu bez skręcania (0) przez 1 obrót (360)
17    await motor_pair.move_for_degrees(motor_pair.PAIR_1, -360, 0)
18
19    #faza 1 - testy parametrów prędkości. Dodatkowe wykorzystanie matrycy aby zrozumieć w której części kodu się jest
20    light_matrix.clear()
21    light_matrix.set_pixel(0,0,100)
22
23    #jazda do przodu bez skręcania (0) przez 3 obroty (1080 stopni) z prędkością 720
24    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1080, 0, velocity=720)
25
26    #jazda do tyłu bez skręcania (0) przez 3 obroty (-1080 stopni) z prędkością 1100
27    #zauważ że robot wraca do tej samej pozycji mimo innej prędkości
28    await motor_pair.move_for_degrees(motor_pair.PAIR_1, -1080, 0, velocity=1100)
29
30    #faza 2 - testy parametrów przyspieszenia. Dodatkowe wykorzystanie matrycy aby zrozumieć w której części kodu się jest
31    light_matrix.set_pixel(0,1,100)
32
33    #jazda do przodu bez skręcania (0) przez 3 obroty (1080 stopni) z prędkością 720, przyspieszeniem = 200
34    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1080, 0, velocity=720, acceleration=200)
35
36    #jazda do tyłu bez skręcania (0) przez 3 obroty (-1080 stopni) z prędkością 720, przyspieszeniem = 600
37    #zauważ że robot wraca do tej samej pozycji mimo innego przyspieszenia
38    await motor_pair.move_for_degrees(motor_pair.PAIR_1, -1080, 0, velocity=720, acceleration=600)
39
40    runloop.run(main())

```


Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Co się wydarzyło? Robot zaczął się rozpędzać – startować powoli. Czy jest różnica pomiędzy wartością 200 a 600? Spróbuj potestować. Parametr **acceleration** przyjmuje liczby z zakresu od 1 do 10000.

Faza 3 – testy spowolnienia

Tak samo jak przyspieszanie przy starcie, możliwe jest także zwolnienie pod koniec jazdy. Służy do tego parametr **deceleration**. Składnia jest taka sama jak **acceleration**, zakres też. Pora go wykorzystać. Zapal kolejną diodę i zastosuj spowolnienie. Ustaw mu wartość 200 i w drodze powrotnej 600. Obserwuj różnice.

```
#faza 2 - testy parametrów przyspieszenia. Dodatkowe wykorzystanie matrycy aby zrozumieć w której części kodu się jest
light_matrix.set_pixel(0,1,100)

#jazda do przodu bez skręcania (0) przez 3 obroty (1080 stopni) z prędkością 720, przyspieszeniem = 200
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1080, 0, velocity=720, acceleration=200)

#jazda do tyłu bez skręcania (0) przez 3 obroty (-1080 stopni) z prędkością 720, przyspieszeniem = 600
#zauważ że robot wraca do tej samej pozycji mimo innego przyspieszenia
await motor_pair.move_for_degrees(motor_pair.PAIR_1, -1080, 0, velocity=720, acceleration=600)

#faza 3 - testy parametrów spowalniania. Dodatkowe wykorzystanie matrycy aby zrozumieć w której części kodu się jest
light_matrix.set_pixel(0,2,100)

#jazda do przodu bez skręcania (0) przez 3 obroty (1080 stopni) z prędkością 720, spowolnienie = 200
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1080, 0, velocity=720, deceleration=200)

#jazda do tyłu bez skręcania (0) przez 3 obroty (-1080 stopni) z prędkością 720, spowolnienie = 600
#zauważ że robot wraca do tej samej pozycji mimo innego przyspieszenia
await motor_pair.move_for_degrees(motor_pair.PAIR_1, -1080, 0, velocity=720, deceleration=600)
```

Blok kodu, który jest powyżej, pokazuje fazy 2 i 3. Od teraz będziesz widzieć komendy z poprzedniego ćwiczenia (fazy) oraz aktualne. Po prostu dodawaj kod.

Twój kod powinien w tej chwili wyglądać następująco (wersja bez komentarzy):

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

```

1 from hub import light_matrix
2 from hub import port
3 import motor_pair
4 import runloop
5
6 async def main():
7     # write your code here
8     await light_matrix.write("Hi!")
9     motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
10
11     await motor_pair.move_for_degrees(motor_pair.PAIR_1, 360, 0)
12     await motor_pair.move_for_degrees(motor_pair.PAIR_1, -360, 0)
13
14     light_matrix.clear()
15     light_matrix.set_pixel(0,0,100)
16
17     await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1080, 0, velocity=720)
18     await motor_pair.move_for_degrees(motor_pair.PAIR_1, -1080, 0, velocity=1100)
19
20     light_matrix.set_pixel(0,1,100)
21     await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1080, 0, velocity=720, acceleration=200)
22     await motor_pair.move_for_degrees(motor_pair.PAIR_1, -1080, 0, velocity=720, acceleration=600)
23
24     light_matrix.set_pixel(0,2,100)
25     await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1080, 0, velocity=720, deceleration=200)
26     await motor_pair.move_for_degrees(motor_pair.PAIR_1, -1080, 0, velocity=720, deceleration=600)
27
28 runloop.run(main())

```

Faza 4 – ruch czółgowy

W ramach biblioteki **motor_pair** dostępna jest także funkcja **move_tank_for_degrees()**. Działa ona tak samo jak funkcja **move_for_degrees()** z jednym wyjątkiem.

W funkcji tank należy podać prędkość każdego silnika z osobna!

Jak wygląda funkcja? Oto ona:

```
await motor_pair.move_tank_for_degrees(motor_pair.PAIR_1, 1080, 360, 360)
```

Jakie parametry przyjmuje funkcja **move_tank_for_degrees()**? Pierwszym jest standardowo para silników, które obsługujesz – **PAIR_1**. Drugim parametrem jest liczba stopni, ile ma silnik wykonać – 1080. Kolejne dwa to właśnie prędkości silników. Pierwsze 360 to prędkość dla silnika lewego, drugie 360 to prędkość dla silnika prawego. Pora przejść do akcji:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

```

40 #faza 3 - testy parametrów spowalniania. Dodatkowe wykorzystanie matrycy aby zrozumieć w której części kodu się jest
41 light_matrix.set_pixel(0,2,100)
42
43 #jazda do przodu bez skręcania (0) przez 3 obroty (1080 stopni) z prędkością 720, spowolnienie = 200
44 await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1080, 0, velocity=720, deceleration=200)
45
46 #jazda do tyłu bez skręcania (0) przez 3 obroty (-1080 stopni) z prędkością 720, spowolnienie = 600
47 #zauważ że robot wraca do tej samej pozycji mimo innego przyspieszenia
48 await motor_pair.move_for_degrees(motor_pair.PAIR_1, -1080, 0, velocity=720, deceleration=600)
49
50 #faza 4 - testy parametrów funkcji tank: zmiana prędkości. Dodatkowe wykorzystanie matrycy aby zrozumieć w której części kodu się jest
51
52 light_matrix.set_pixel(1,0,100)
53
54 #jazda do przodu przez 3 obroty (1080 stopni), prędkość lewego silnika 360, prędkość prawego silnika 360
55 #funkcja tank pozwala na zdefiniowanie parametrów na poszczególne silniki mimo zarządzania obydwoma jednocześnie
56 await motor_pair.move_tank_for_degrees(motor_pair.PAIR_1, 1080, 360, 360)
57
58 #jazda do tyłu przez 3 obroty (-1080 stopni), prędkość lewego silnika 1100, prędkość prawego silnika 1100
59 await motor_pair.move_tank_for_degrees(motor_pair.PAIR_1, -1080, 1100, 1100)

```

Dodaj kod do swojego programu i zobacz, jak robot się zachowuje. W dwóch liniach zmieniono mu wartości prędkości silników. To tak, jakby zmieniano parametr **velocity** w funkcji **move_for_degrees** – działanie jest identyczne.

Faza 5 – przyspieszanie i zwalnianie w trybie czółgowym

Pora na ostatnią część. Tu zobaczysz, że parametry **acceleration** i **deceleration** również działają – i mają ten sam wpływ na robota. Oto kod:

```

76 #faza 4 - testy parametrów funkcji tank: zmiana prędkości. Dodatkowe wykorzystanie matrycy aby zrozumieć w której części kodu się jest
77
78 light_matrix.set_pixel(1,0,100)
79
80 #jazda do przodu przez 3 obroty (1080 stopni), prędkość lewego silnika 360, prędkość prawego silnika 360
81 #funkcja tank pozwala na zdefiniowanie parametrów na poszczególne silniki mimo zarządzania obydwoma jednocześnie
82 await motor_pair.move_tank_for_degrees(motor_pair.PAIR_1, 1080, 360, 360)
83
84 #jazda do tyłu przez 3 obroty (-1080 stopni), prędkość lewego silnika 1100, prędkość prawego silnika 1100
85 await motor_pair.move_tank_for_degrees(motor_pair.PAIR_1, -1080, 1100, 1100)
86
87 #faza 5 - testy parametrów funkcji tank: przyspieszanie i spowalnianie. Dodatkowe wykorzystanie matrycy aby zrozumieć w której części kodu się jest
88 light_matrix.set_pixel(1,1,100)
89
90 #jazda do przodu przez 3 obroty (1080 stopni), prędkość lewego silnika 360, prędkość prawego silnika 360, przyspieszenie 400
91 await motor_pair.move_tank_for_degrees(motor_pair.PAIR_1, 1080, 360, 360, acceleration=400)
92
93 #jazda do tyłu przez 3 obroty (-1080 stopni), prędkość lewego silnika 1100, prędkość prawego silnika 1100, spowolnienie 400
94 await motor_pair.move_tank_for_degrees(motor_pair.PAIR_1, 1080, -1100, -1100, deceleration=400)

```

Uruchom robota i zobacz, co się dzieje.

Jeśli potrzebujesz: oto cały kod w formie tekstowej realizujący wszystkie fazy:

```

from hub import light_matrix
from hub import port
import motor_pair
import runloop

async def main():
    # write your code here
    await light_matrix.write("Hi!")
    motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)

    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 360, 0)
    await motor_pair.move_for_degrees(motor_pair.PAIR_1, -360, 0)

    #faza 1 - testy parametrów prędkości

```

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

```
light_matrix.clear()
light_matrix.set_pixel(0,0,100)

await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1080, 0, velocity=720)
await motor_pair.move_for_degrees(motor_pair.PAIR_1, -1080, 0, velocity=1100)

#faza 2 - testy parametrów przyspieszenia
light_matrix.set_pixel(0,1,100)
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1080, 0, velocity=720, acceleration=200)
await motor_pair.move_for_degrees(motor_pair.PAIR_1, -1080, 0, velocity=720, acceleration=600)

#faza 3 - testy parametrów spowalniania
light_matrix.set_pixel(0,2,100)
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1080, 0, velocity=720, deceleration=200)
await motor_pair.move_for_degrees(motor_pair.PAIR_1, -1080, 0, velocity=720, deceleration=600)

#faza 4 - testy parametrów funkcji tank: zmiana prędkości

light_matrix.set_pixel(1,0,100)
await motor_pair.move_tank_for_degrees(motor_pair.PAIR_1, 1080, 360, 360)
await motor_pair.move_tank_for_degrees(motor_pair.PAIR_1, -1080, 1100, 1100)

#faza 5 - testy parametrów funkcji tank: przyspieszanie i spowalnianie
light_matrix.set_pixel(1,1,100)
await motor_pair.move_tank_for_degrees(motor_pair.PAIR_1, 1080, 360, 360, acceleration=400)
await motor_pair.move_tank_for_degrees(motor_pair.PAIR_1, 1080, -1100, -1100, deceleration=400)

runloop.run(main())
```

Eksperymentuj z wartościami. Zobacz, jak robot zachowuje się, jeśli podasz mu inne liczby.

Powodzenia i do następnej lekcji!

Możliwe rozszerzenia zadania

1. Wykonaj skręcanie w lewo za pomocą funkcji **move_tank_for_degrees**. Podpowiedź: sterujesz oddzielnie prędkością silników. Pozamieniaj wartości
2. Rozpędź robota jazdą do przodu o 720 stopni. Zwalniaj, zacznij skręcać po łuku w lewo i przyspiesz, wykonując ruch 1080 stopni.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 5: Odkrywamy tajemnice Kodu... Robot eksplorator

Temat lekcji nr 18 - Robot dostarcza autonomicznie różne elementy niezbędne ludziom w pracy. Analiza problemu i tworzenie planu kodu źródłowego

Treści nauczania (wymagania szczegółowe - podstawa programowa) - I. 5); IV. 4) II. 3) b); II. 3) c); I. 1)

Cel lekcji:

- **Wprowadzenie uczniów do planowania pracy** z wykorzystaniem dokumentu tekstowego oraz narzędzi TechOffice w kontekście zadania robota eksploratora.
- **Przygotowanie uczniów do tworzenia przejrzystego planu realizacji zadania** związanego z przejazdem robota i wykorzystaniem manipulatora podczas autonomicznego dostarczania elementów.
- **Rozwijanie umiejętności zapisywania kolejnych etapów pracy**, formułowania celów oraz porządkowania działań w logicznej kolejności z uwzględnieniem analizy trasy przejazdu robota i działania manipulatora.
- **Kształtowanie kompetencji pracy z edytorem tekstu**, w tym tworzenia i edytowania dokumentów, zapisywania informacji oraz ich czytelnego formatowania z wykorzystaniem narzędzi TechOffice.
- **Uświadomienie uczniom znaczenia planowania** jako etapu poprzedzającego programowanie i wpływającego na skuteczność działania robota eksploratora podczas realizacji zadania transportowego.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- **Uczeń w algorytmicznym rozwiązywaniu problemu wyróżnia podstawowe kroki:**
 - tworzy plan działania robota eksploratora z wykorzystaniem manipulatora oraz narzędzi TechOffice,
 - określa problem i cel zadania związany z autonomicznym dostarczaniem elementów,
 - opracowuje uporządkowaną listę działań prowadzących do realizacji zadania.
- **Uczeń analizuje informacje i formułuje wnioski:**

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- analizuje trasę przejazdu robota oraz działanie manipulatora,
- podejmuje decyzje dotyczące sposobu działania robota,
- wykorzystuje narzędzia cyfrowe, w tym TechOffice, do opracowania rozwiązania problemu.
- **Uczeń formułuje i zapisuje w postaci algorytmów:**
 - kolejne etapy pracy prowadzące do rozwiązania problemu,
 - logiczną sekwencję działań robota związaną z przejazdem i transportem elementów,
 - plan działań możliwy do późniejszego wykorzystania podczas programowania robota eksploratora..

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści kształcenia będą realizowane za pomocą robotyki edukacyjnej LEGO® Education SPIKE™ Prime.

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna RaP STEAM,
- praca na komputerze,
- praca przy zestawie robotyki edukacyjnej,
- pogadanka.

Środki dydaktyczne

- zestaw PC z dostępem do Internetu,
- zestaw robotyki edukacyjnej LEGO® Education SPIKE™ Prime oraz zestaw rozszerzony LEGO® Education SPIKE™ Prime Expansion Kit,
- rzutnik multimedialny,
- e-podręcznik,
- mata/tor z wyznaczoną trasą,
- kartki papieru i ołówki.

Formy pracy

- Praca indywidualna lub zespołowa.

Przebieg lekcji:

Faza I. Wprowadzenie

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Nauczyciel rozpoczyna lekcję od przedstawienia tematu i celu zajęć, którymi są analiza zadania związanego z **autonomicznym dostarczaniem elementów przez robota eksploratora** oraz **opracowanie planu działania robota w dokumencie tekstowym**. Podkreśla znaczenie planowania, logicznego myślenia oraz analizy informacji potrzebnych do realizacji zadania.

Następnie nauczyciel prezentuje sytuację zadaniową polegającą na **przejeździe robota po wyznaczonej trasie, pobraniu elementów za pomocą manipulatora oraz dostarczeniu ich do określonego miejsca**. Omawia główne elementy zadania (punkt startowy, trasa przejazdu, sposób transportu elementów, działanie manipulatora oraz miejsce zakończenia zadania). Uczniowie analizują sytuację oraz dostępne informacje, co stanowi podstawę do dalszej pracy.

Nauczyciel wskazuje, że kluczowym etapem będzie **opracowanie planu działania w dokumencie tekstowym** z wykorzystaniem narzędzi TechOffice. Uczniowie zapisują w nim:

- problem i cel działania robota,
- uporządkowaną listę kroków,
- plan przejazdu i transportu elementów,
- wstępny algorytm rozwiązania.

Następnie uczniowie pracują w zespołach, dzieląc się rolami (np. programista, analityk, tester), i wspólnie opracowują rozwiązanie.

Na zakończenie tej części nauczyciel podkreśla, że przygotowany dokument będzie podstawą do dalszego etapu – programowania i testowania działania robota eksploratora.

Faza II. Część zasadnicza - realizacja

Uczniowie przechodzą do realizacji zadania, łącząc analizę w dokumencie tekstowym z tworzeniem programu i pracą z robotem.

1. Nauczyciel wyjaśnia sposób pracy, podkreślając, że podstawą działań jest **dokument tekstowy** (np. TechOffice), w którym uczniowie zapisują plan działania robota (problem, cel, kolejne kroki).
2. Uczniowie pracują w zespołach i opracowują plan rozwiązania zadania rozpoznania sytuacji i podjęcia decyzji przez robota.
3. Nauczyciel przypomina, że analiza informacji i poprawny plan są kluczowe przed rozpoczęciem programowania.
4. Następnie uczniowie przechodzą do środowiska LEGO SPIKE Prime i tworzą program na podstawie przygotowanego planu.
5. Uczniowie:
 - a. uzupełniają dokument o szczegółowe kroki działania,
 - b. tworzą program w oparciu o schematy blokowe i/lub psuedokod,
 - c. uruchamiają go i obserwują działanie robota, (podczas następnej lekcji)
 - d. porównują efekty z planem i wprowadzają poprawki (program + dokument) (podczas następnej lekcji).

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Uczniowie uczą się, że proces ma charakter iteracyjny: analiza >> plan >> program >> test >> poprawa.

Faza III. Podsumowanie

Uczniowie prezentują swoje rozwiązania (dokument + proponowany program), a nauczyciel omawia ich działanie oraz ocenia, zwracając uwagę na analizę zadania, poprawność planu oraz zgodność działania robota z założeniami. Kryteria oceniania:

Ocena dopuszczająca:

- uczeń tworzy prosty plan działania w dokumencie lub program, jednak rozwiązanie jest niepełne i nie prowadzi do realizacji zadania.

Ocena dostateczna:

- uczeń modyfikuje algorytm, jednak działanie nie jest zgodne z założeniami zadania (np. błędna analiza lub decyzje),

Ocena dobra:

- uczeń opracowuje plan i program umożliwiające działanie robota, jednak rozwiązanie zawiera nieścisłości lub nie w pełni uwzględnia warunki zadania,

Ocena bardzo dobra:

- uczeń poprawnie analizuje zadanie, tworzy czytelny plan w dokumencie i program,
- uczeń analizuje rozwiązanie i wprowadza trafne poprawki.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, np.:

- Jaką rolę pełni manipulator w zadaniu autonomicznego dostarczania elementów?
 - umożliwia pobieranie i transportowanie elementów**
 - służy do ładowania baterii
 - odpowiada za wyświetlanie komunikatów
 - zwiększa prędkość jazdy robota
- Dlaczego robot musi poruszać się zgodnie z wcześniej zaplanowaną trasą?
 - aby program był krótszy
 - aby poprawnie dostarczyć elementy do wyznaczonego miejsca**
 - aby robot poruszał się wolniej
 - nie ma to znaczenia podczas działania robota

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

3. Jakie informacje powinny znaleźć się w planie działania robota zapisanym w dokumencie tekstowym?
- tylko nazwa robota
 - lista kolorów wykorzystanych w programie
 - kolejne etapy działania robota i sposób realizacji zadania**
 - wyłącznie długość trasy przejazdu
4. Dlaczego podczas pracy wykorzystuje się narzędzia typu TechOffice?
- aby zwiększyć prędkość działania robota
 - aby tworzyć i porządkować dokumentację zadania**
 - aby sterować silnikami robota
 - aby ładować kontroler Spike Hub

Faza V. Dodatkowe, praca domowa (dla chętnych).

Zadanie dodatkowe (dla chętnych).

Zadanie: Robot magazynowy – plan działania robota

Uczeń przygotowuje dokument tekstowy (np. w TechOffice), w którym opracuje plan działania robota eksploratora odpowiedzialnego za transport elementów pomiędzy stanowiskami pracy.

Robot ma za zadanie:

- wystartować z pola startowego,
- przejechać do wyznaczonego punktu odbioru,
- pobrać element za pomocą manipulatora,
- bezpiecznie przetransportować element do miejsca docelowego,
- wrócić na pozycję startową lub przejechać do kolejnego punktu zadania,
- zakończyć pracę po wykonaniu wszystkich etapów.

W dokumencie przygotuj:

- tytuł dokumentu,
- opis problemu i celu działania robota,
- listę kolejnych etapów pracy robota,
- plan przejazdu robota między punktami,
- pseudokod lub schemat działania robota,
- opis wykorzystania manipulatora podczas transportu elementów.

Wymagania:

- zastosuj numerację, punktory lub tabelę,
- zachowaj logiczną kolejność działań,
- uwzględnij możliwość wielokrotnego wykonywania transportu,
- przygotuj dokument w czytelnej formie z wykorzystaniem narzędzi TechOffice,



Fundusze Europejskie
dla Podkarpacia



Rzeczpospolita
Polska

Dofinansowane przez
Unię Europejską



PODKARPACKIE
przestrzeń otwarta

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- opracuj plan możliwy do późniejszego wykorzystania podczas programowania robota w LEGO SPIKE Prime.



Podkarpacki Zespół
Placówek Wojewódzkich
w Rzeszowie



POLITECHNIKA
RZESZOWSKA
Im. IGNACEGO ŁUKASIEWICZA



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1: Konspekt nauczyciela lekcji nr 18 - Robot dostarcza autonomicznie różne elementy niezbędne ludziom w pracy. Analiza problemu i tworzenie planu kodu źródłowego

Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
- Przedstawienie uczniom celu i zakresu tematycznego lekcji
 - Nauczyciel omawia główne elementy zadania:
 - jazdę do przodu i do tyłu,
 - skręt podczas jazdy (ruch po łuku),
 - sterowanie silnikami z wykorzystaniem biblioteki motor_pair,
 - możliwość wykorzystania parametrów velocity oraz deceleration.
 - Uczniowie analizują zadanie i przygotowują plan działania w dokumencie tekstowym (np. TechOffice):
 - Podział uczniów na zespoły, oraz przydzielenie ról w grupie:
 - Uruchomienie na e-platformie RaP STEAM materiałów do lekcji.
 - Praca uczniów nad dokumentem:
 - zapis problemu i celu działania robota,
 - opracowanie kolejnych etapów przejazdu robota,
 - przygotowanie planu transportu ładunku,
 - opracowanie pseudokodu lub schematu działania robota,
 - analiza wykorzystania chwytaka oraz manipulatora.
 - Praca praktyczna uczniów:
 - testowanie różnych sposobów przejazdu robota,
 - analiza skuteczności manewru cofania i skrętu,
 - dobieranie parametrów ruchu robota,
 - proponowanie własnych rozwiązań usprawniających transport ładunku.
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów (analiza poprawności planów działania oraz sposobów rozwiązania zadania).
- Zadanie pracy domowej (opcjonalnie) – opracowanie dodatkowego wariantu trasy przejazdu robota lub przygotowanie własnego planu transportu ładunku z wykorzystaniem manipulatora.

Załącznik nr 2: Realizacja zadania głównego podczas lekcji nr 18 - Robot dostarcza autonomicznie różne elementy niezbędne ludziom w pracy. Analiza problemu i tworzenie planu kodu źródłowego

Cel i wprowadzenie do lekcji:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Uczniowie, pracując w zespołach, realizują zadanie polegające na analizie sytuacji i **zaplanowaniu działania robota eksploratora odpowiedzialnego za autonomiczny transport ładunku. Robot ma bezpiecznie poruszać się po wyznaczonej trasie, pobrać element za pomocą chwytaka oraz dostarczyć go do wyznaczonego miejsca.** Szczególny nacisk położony jest na **przygotowanie planu w dokumencie tekstowym** (np. TechOffice), który stanowi podstawę do dalszego programowania.

Uczniowie mają za zadanie opracować plan działania robota, który uwzględnia **analizę otoczenia oraz podejmowanie decyzji związanych z transportem ładunku.** W tym celu uczniowie:

- analizują sytuację zadania i dostępne informacje,
- określają cel działania robota (przejazd wyznaczoną trasą, pobranie i transport ładunku z wykorzystaniem manipulatora) oraz możliwe trudności,
- planują kolejność działań robota,
- opracowują algorytm w formie uporządkowanej listy kroków,
- zapisują rozwiązanie w dokumencie tekstowym (lista czynności, pseudokod), dbając o jego czytelność,
- przygotowują plan, który będzie podstawą do późniejszego programowania robota eksploratora,
- (na kolejnej lekcji) wykorzystają plan do realizacji zadania i weryfikacji poprawności działania robota podczas transportu ładunku.

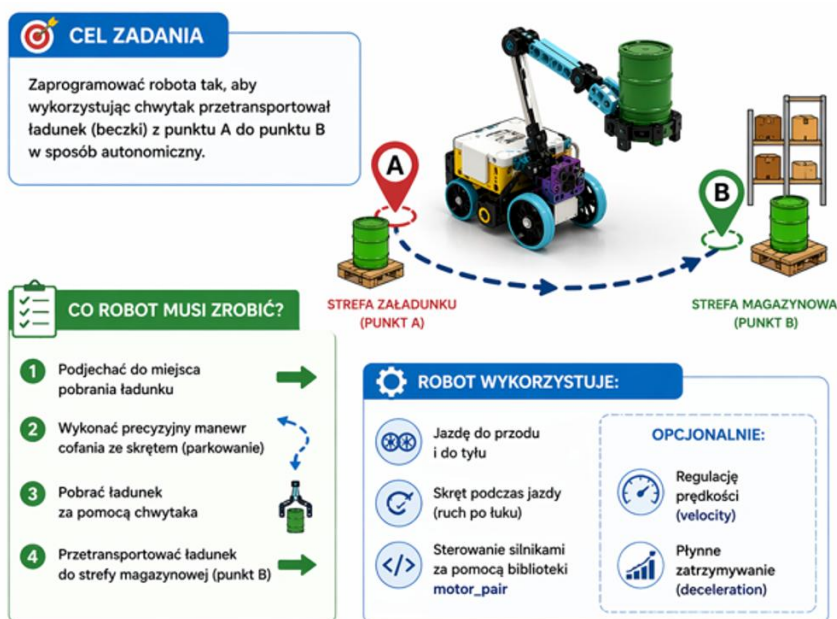
Uczniowie na lekcji skupiają się na zgodności między analizą zadania a zapisem planu w dokumencie tekstowym, który będzie wykorzystany w dalszych etapach pracy.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 1. Przygotowanie do realizacji zadania głównego

Opis zadania dla uczniów:

Celem zadania jest zaprogramowanie robota, który, wykorzystując chwytak, przetransportuje ładunek (beczki) z punktu A do punktu B. Robot znajduje się w strefie załadunku (punkt A) i musi podjechać do punktu pośredniego, wykonać precyzyjny manewr cofania ze skretem (parkowanie), pobrać ładunek oraz przetransportować go do strefy magazynowej (punkt B).



Uczniowie, pracując w zespołach, analizują trasę przejazdu robota oraz rozmieszczenie elementów zadania (punkt startowy, miejsce pobrania ładunku, przeszkody oraz punkt docelowy – magazyn), a następnie **przygotowują dokument tekstowy opisujący cały proces planowania działania robota eksploratora realizującego autonomiczne zadanie transportowe**. Na podstawie analizy sytuacji uczniowie ustalają kolejność wykonywania działań, określają miejsca zatrzymania robota oraz sposób wykorzystania chwytaka podczas pobierania i transportu ładunku.



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Na tej podstawie uczniowie tworzą dokument, w którym:

- opisują analizę zadania i dostępnych informacji (rozmieszczenie punktów oraz elementów otoczenia),
- uzasadniają wybór trasy przejazdu robota,
- określają kolejność realizacji działań robota,
- wskazują miejsca zatrzymania, cofania oraz pobrania i transportu ładunku,
- przedstawiają plan działania w formie uporządkowanej listy kroków lub pseudokodu,
- opisują proces planowania oraz ewentualne trudności i podjęte decyzje związane z ruchem robota i transportem ładunku.

Dokument powinien być czytelny, logicznie uporządkowany i stanowić podstawę do późniejszego zaprogramowania robota eksploratora.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

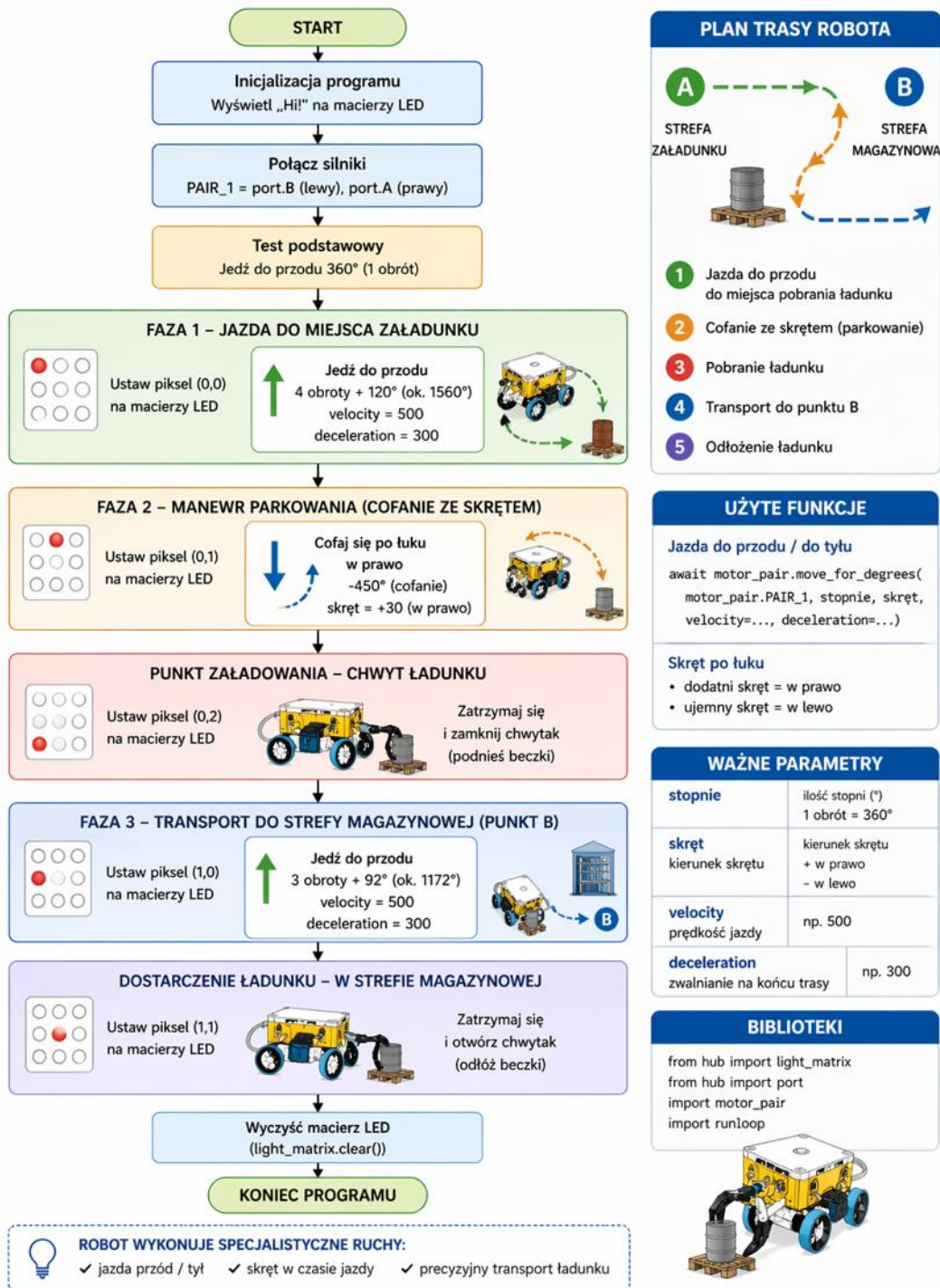
Zadanie 2. Analiza problemu i tworzenie planu kodu źródłowego dla autonomicznego robota

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

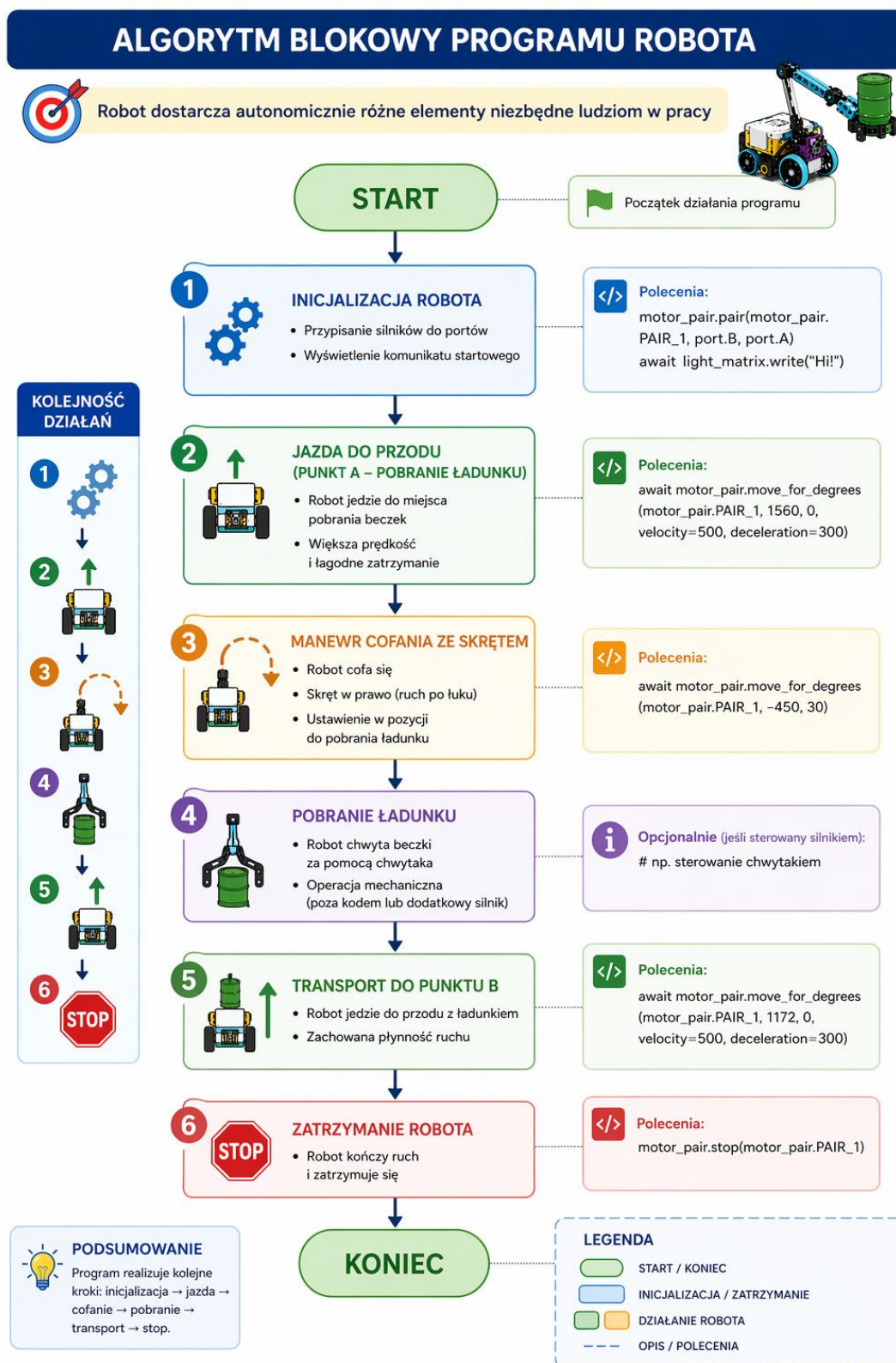
SCHEMAT BLOKOWY – TRANSPORT ŁADUNKU — LEGO SPIKE —



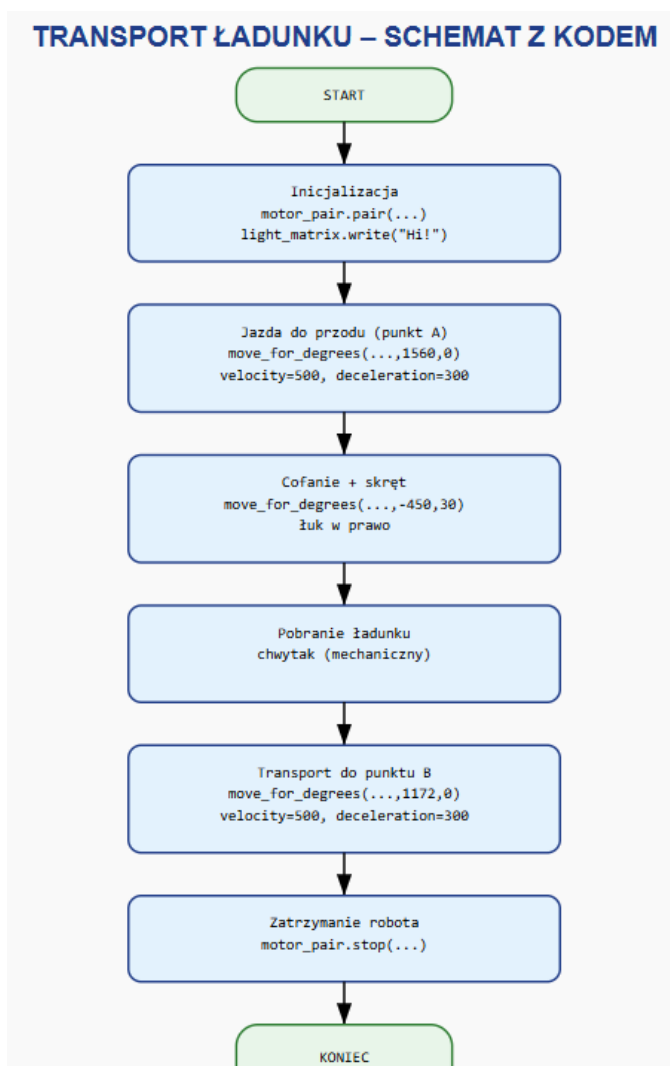
ZADANIE: Zaprogramuj robota tak, aby wykorzystując chwyt przewiózł ładunek (beczki) z punktu A (strefy załadunku) do punktu B (strefy magazynowej).



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Podsumowanie analizy problemu i tworzenie planu kodu źródłowego dla autonomicznego robota

Podczas tego etapu zajęć uczniowie realizowali zadanie polegające na zaprogramowaniu robota, który **autonomicznie transportuje ładunek z punktu A do punktu B**. W trakcie pracy robot musiał podjechać do miejsca pobrania beczek, wykonać precyzyjny manewr cofania ze skrętem, pobrać ładunek za pomocą chwytaka, a następnie bezpiecznie dostarczyć go do strefy magazynowej.

Uczniowie doskonalili umiejętność planowania działania robota krok po kroku oraz tworzenia logicznego algorytmu sterowania. Poznali zastosowanie biblioteki **motor_pair**, która umożliwia jednoczesne sterowanie dwoma silnikami odpowiedzialnymi za jazdę robota. Uczyli się także wykorzystywać parametry ruchu, takie jak prędkość (**velocity**) oraz płynne zatrzymywanie (**deceleration**), aby robot poruszał się dokładnie i stabilnie.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Lekcja pokazała, że roboty mogą wspierać ludzi w pracy, wykonując zadania transportowe szybciej, dokładniej i bezpieczniej. Skuteczne działanie robota zależy jednak od dobrze przygotowanego algorytmu, poprawnego kodu źródłowego oraz precyzyjnego testowania programu.

Teraz możemy przejść do tworzenia dokumentacji projektowej. Dokument powinien być czytelny, logicznie uporządkowany i stanowić podstawę do późniejszego zaprogramowania robota eksploratora.

Zadanie 3. Praca z edytorem tekstu

[Etap 1] Uruchomienie edytora tekstu

1. Włącz komputer.
2. Otwórz program **LibreOffice Writer**.
3. Utwórz **nowy dokument tekstowy**.
4. Zapisz plik pod nazwą: „*Dokumentacja do projektu robota – lekcja 18*” w folderze wskazanym przez nauczyciela.
5. Używaj nagłówków (Nagłówek 1, 2), dodawaj obrazy i schematy. Zapisuj plik regularnie.

[Etap 2] Stwórz dokument, który będzie miał następującą strukturę:

1. Strona tytułowa

Uzupełnij tytuł: *Transport ładunku robotem LEGO Spike*

Dodatkowo:

- imię i nazwisko
- klasa
- data

2. Wprowadzenie (opis problemu)

Opisz zadanie własnymi słowami:

Przykład:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Celem zadania jest zaprogramowanie robota LEGO Spike, który przewiezie ładunek (beczki) z punktu A do punktu B. Robot wykorzystuje chwytak oraz wykonuje ruchy do przodu, do tyłu oraz skręty podczas jazdy.

3. Założenia projektu

Wypisz wymagania:

- robot startuje w punkcie A
- musi dojechać do punktu pośredniego,
- wykonuje manewr cofania ze skrętem
- robot pobiera ładunek
- transportuje ładunek do punktu B
- zatrzymuje się i odkłada ładunek

4. Opis działania robota (kroki)

Użyj do opisu działania robota listę numerowaną, np.

1. **Jazda do przodu**
 - robot jedzie do miejsca pobrania ładunku
 - dystans: ok. **4 obroty + 120°**
2. **Manewr parkowania (cofanie ze skrętem)**
 - robot cofa się po łuku w prawo
 - ustawia się w pozycji do pobrania ładunku
3. **Pobranie ładunku**
 - chwytak łapie beczki (element mechaniczny)
4. **Transport do punktu B**
 - robot jedzie do przodu do strefy magazynowej
 - dystans: ok. **3 obroty + 92°**

5. Schemat blokowy

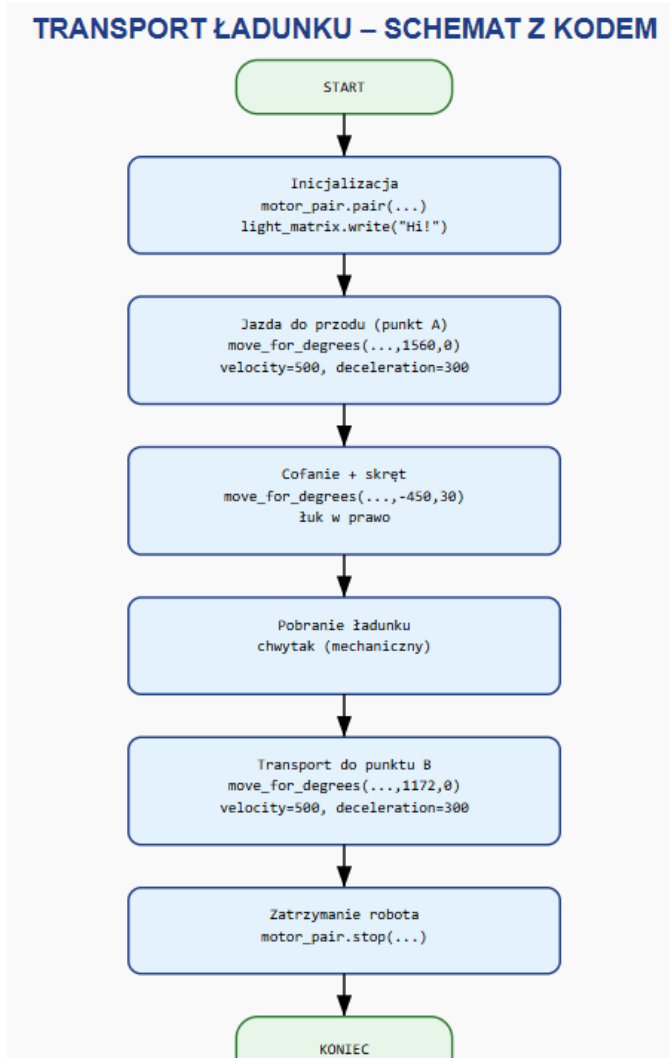
Stwórz schemat blokowy programu. Wklej gotową grafikę (jeśli ją już masz) lub narysuj schemat w LibreOffice (**Wstaw → Kształty**). Wykorzystaj poniższe bloki do stworzenie schematu:



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przykładowy schemat jaki możesz utworzyć prezentuje się jak poniżej:



6. Opis programu (kod)

Wklej fragmenty kodu i je opisz:

Przykład:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

```
76 #faza 4 - testy parametrów funkcji tank: zmiana prędkości. Dodatkowe wykorzystanie matrycy aby zrozumieć w której części kodu się jest
77
78 light_matrix.set_pixel(1,0,100)
79
80 #jazda do przodu przez 3 obroty (1080 stopni), prędkość lewego silnika 360, prędkość prawego silnika 360
81 #funkcja tank pozwala na zdefiniowanie parametrów na poszczególne silniki mimo zarządzania obydwoma jednocześnie
82 await motor_pair.move_tank_for_degrees(motor_pair.PAIR_1, 1080, 360, 360)
83
84 #jazda do tyłu przez 3 obroty (-1080 stopni), prędkość lewego silnika 1100, prędkość prawego silnika 1100
85 await motor_pair.move_tank_for_degrees(motor_pair.PAIR_1, -1080, 1100, 1100)
86
87 #faza 5 - testy parametrów funkcji tank: przyspieszanie i spowalnianie. Dodatkowe wykorzystanie matrycy aby zrozumieć w której części kodu się jest
88 light_matrix.set_pixel(1,1,100)
89
90 #jazda do przodu przez 3 obroty (1080 stopni), prędkość lewego silnika 360, prędkość prawego silnika 360, przyspieszenie 400
91 await motor_pair.move_tank_for_degrees(motor_pair.PAIR_1, 1080, 360, 360, acceleration=400)
92
93 #jazda do tyłu przez 3 obroty (-1080 stopni), prędkość lewego silnika 1100, prędkość prawego silnika 1100, spowolnienie 400
94 await motor_pair.move_tank_for_degrees(motor_pair.PAIR_1, 1080, -1100, -1100, deceleration=400)
```

7. Wykorzystane parametry

Opisz wykorzystane parametry:

- **velocity** – prędkość jazdy
- **deceleration** – zwalnianie na końcu ruchu
- **stopnie** – odległość

8. Wnioski

Podsumuj swój projekt. Odpowiedź na pytania:

- Co było trudne?
- Co działało dobrze?
- Jak można ulepszyć robota?

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przykładowe rozwiązanie zadania głównego – gotowa dokumentacja projektowa.

Strona tytułowa

Tytuł: *Transport ładunku robotem LEGO Spike*

- Tomasz Kowalski, klasa 7g, 15-05-2027

1. Wprowadzenie (opis problemu)

Celem zadania jest zaprogramowanie robota LEGO Spike, który przewiezie ładunek (beczki) z punktu A do punktu B. Robot wykorzystuje chwytak oraz wykonuje ruchy do przodu, do tyłu oraz skręty podczas jazdy.

2. Założenia projektu

Program sterujący robotem ma realizować określoną sekwencję ruchów z użyciem zmiennych parametrów (prędkości, przyspieszenia, zwalniania).

Opis działania robota:

- robot startuje w punkcie A
- robot porusza się do przodu do punktu pośredniego
- wykonuje manewr cofania ze skrętem
- robot porusza się do przodu łapiąc ładunek
- transportuje ładunek do punktu B

3. Opis działania robota (kroki)

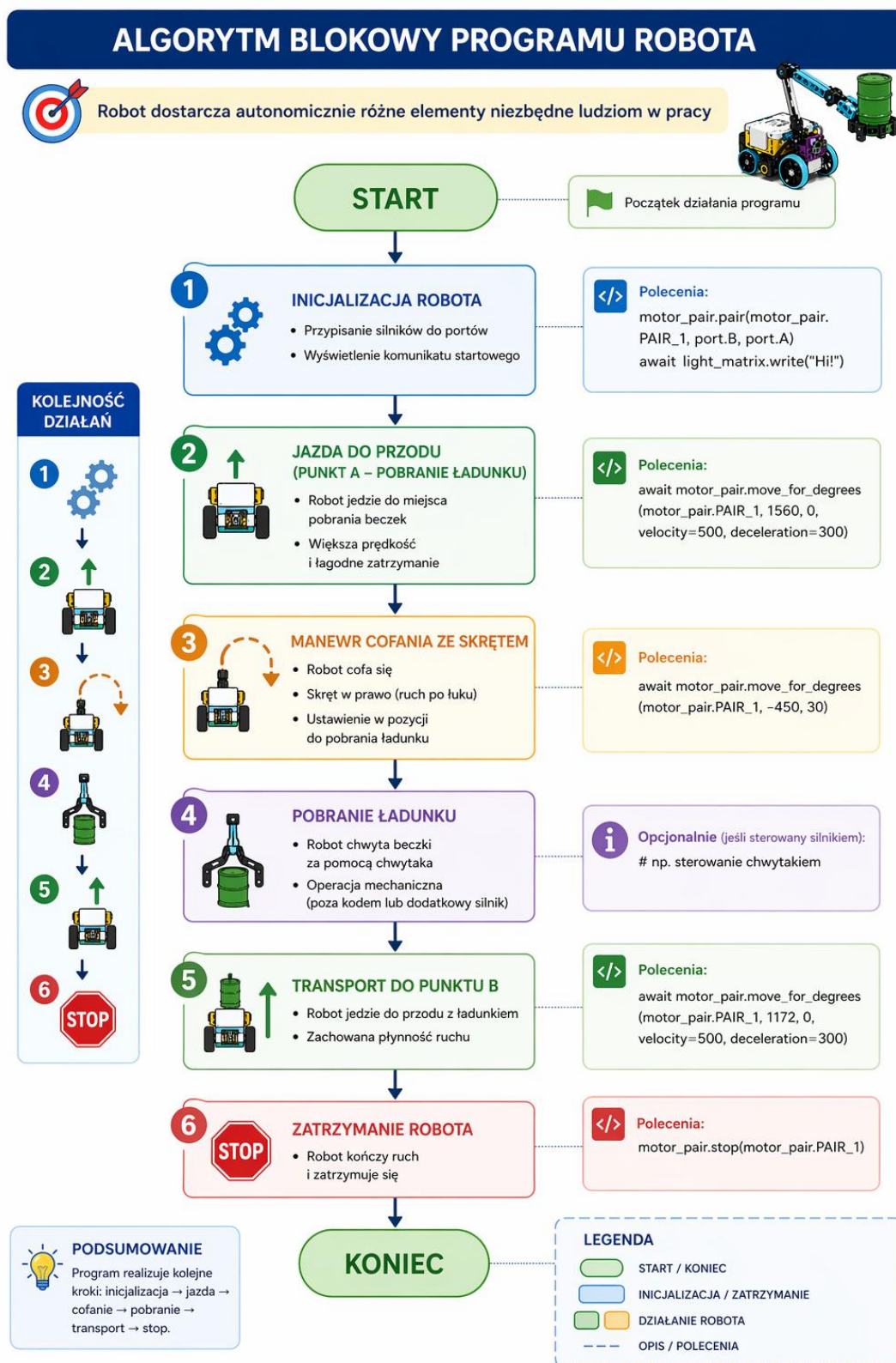
1. Jazda do przodu do miejsca załadunku
2. Cofanie ze skrętem (parkowanie)
3. Pobranie ładunku
4. Jazda do punktu B
5. Odłożenie ładunku

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

4. Schemat blokowy

Schemat blokowy programu służy do graficznego przedstawienia kolejnych kroków działania programu oraz zależności między nimi. Do rozwiązania naszego zadania proponuję:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

5. Opis programu (kod)

Plan programu jest „szkieletem”, który zostanie zamieniony na pełny kod. Składa się z kilku etapów:

1. Inicjalizacja robota

- Przypisanie silników do portów
- Wyświetlenie komunikatu startowego

Polecenia:

```
# write your code here
motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
await light_matrix.write("Hi!")
```

2. Jazda do przodu (punkt A – pobranie ładunku)

- Robot jedzie do miejsca pobrania beczek
- Zastosowano większą prędkość i łagodne zatrzymanie

Polecenia:

```
# write your code here
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1560, 0, velocity=500, deceleration=300)
```

3. Manewr cofania ze skrętem

- Robot cofa się i skręca w prawo (ruch po łuku)
- Ustawia się w pozycji do pobrania ładunku

Polecenia:

```
# write your code here
await motor_pair.move_for_degrees(motor_pair.PAIR_1, -450, 30)
```

4. Pobranie ładunku

- Robot chwytą beczki za pomocą chwytaka
- Operacja mechaniczna (poza kodem lub dodatkowy silnik)

Polecenia (opcjonalne – jeśli sterowany silnikiem):

np. sterowanie chwytakiem

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

5. Transport do punktu B

- Robot jedzie do przodu z ładunkiem
- Zachowana płynność ruchu

Polecenia:

```
# write your code here  
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1172, 0, velocity=500, deceleration=300)
```

6. Zatrzymanie robota

- Robot kończy ruch i zatrzymuje się

Polecenia:

```
# write your code here  
motor_pair.stop(motor_pair.PAIR_1)
```

Program realizuje kolejne kroki:

inicjalizacja → jazda → cofanie → pobranie → transport → stop

6. Wykorzystane parametry

W programie sterującym robotem wykorzystano kilka kluczowych parametrów, które wpływają na sposób jego poruszania się:

Stopnie (degrees)

Parametr określający, o ile stopni mają obrócić się silniki. Bezpośrednio przekłada się na pokonywaną odległość.

- większa wartość → dłuższa droga
- mniejsza wartość → krótsza droga
- wartość ujemna → jazda do tyłu

Przykład: `move_for_degrees(..., 1560, 0)`

Velocity (prędkość)

Parametr określający **szybkość ruchu robota**.

- większa wartość → szybsza jazda

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- mniejsza wartość → wolniejsza jazda

Ważne: Prędkość **nie wpływa na odległość**, tylko na czas jej pokonania.

Przykład: velocity=500

Deceleration (zwalnianie)

Parametr odpowiedzialny za **stopniowe zmniejszanie prędkości przed zatrzymaniem**.

- większa wartość → dłuższe, łagodniejsze hamowanie
- mniejsza wartość → szybsze zatrzymanie

Dzięki temu robot nie zatrzymuje się gwałtownie i porusza się bardziej precyzyjnie

Przykład: deceleration=300

8. Wnioski

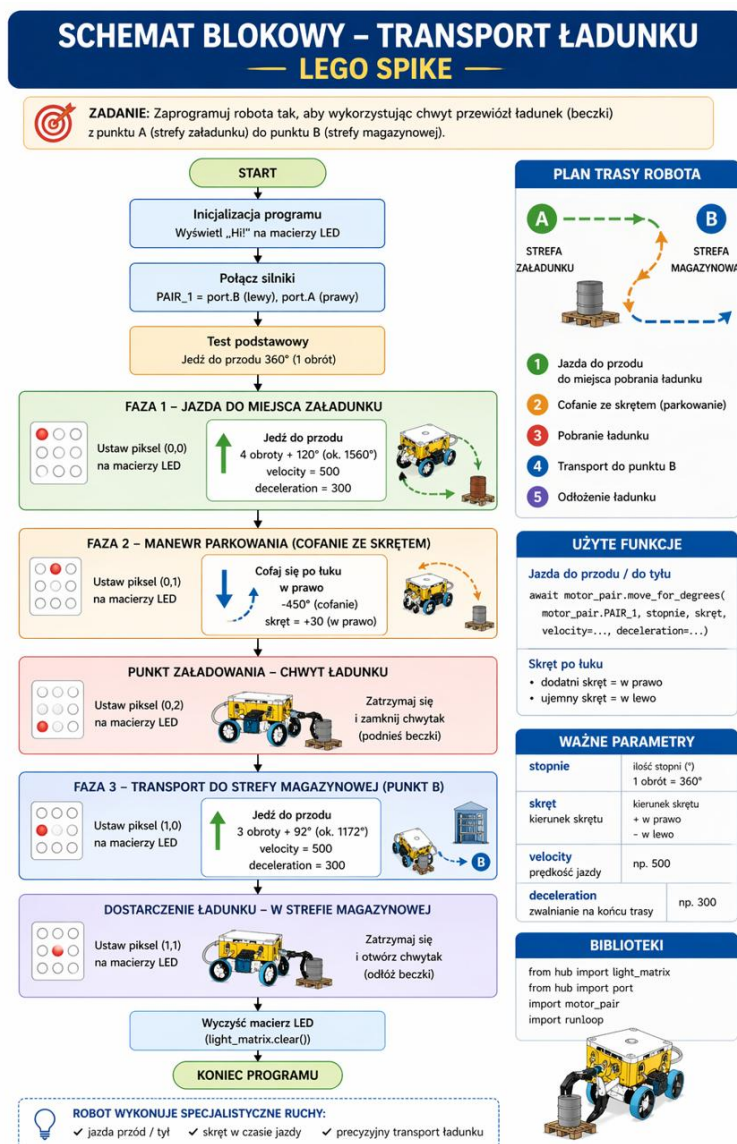
Podsumuj swój projekt. Odpowiedź na pytania:

- Co było trudne?
- Co działało dobrze?
- Jak można ulepszyć robota?

Załącznik 1

Schemat blokowy programu

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 5: Odkrywamy tajemnice Kodu... Odczytać i zrozumieć Matrix

Temat lekcji nr 19 - Robot dostawca – realizacja programu i testy praktyczne.

Treści nauczania (wymagania szczegółowe - podstawa programowa) – I. 2) a); I. 2) b); I. 4); II. 1); II. 2); IV. 1)

Cel lekcji:

- **Wprowadzenie do programowania robota dostawcy w języku Python** – uczniowie poznają podstawowe zasady sterowania robotem, manipulatorem oraz dodatkowymi silnikami w LEGO SPIKE Prime.
- **Analiza działania programu** – uczniowie analizują fragmenty kodu odpowiedzialne za sterowanie ruchem robota, transport ładunku oraz pracę manipulatora i określają, w jaki sposób poszczególne polecenia wpływają na realizację zadania.
- **Tworzenie algorytmów i programów sterujących robotem** – uczniowie opracowują algorytmy oraz zapisują je w postaci programów tekstowych umożliwiających jazdę robota, wykonywanie skrętów, podnoszenie i opuszczanie ładunku oraz transport elementów do wyznaczonego miejsca.
- **Testowanie działania programu** – uczniowie uruchamiają program, obserwują zachowanie robota podczas realizacji zadania transportowego oraz analizują poprawność działania manipulatora i ruchu robota, a następnie wprowadzają poprawki w celu uzyskania poprawnego działania programu.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- **Uczeń dostrzega, że działanie programu i robota zależy od poprawności zapisanych poleceń:**
 - rozumie, że robot wykonuje instrukcje zgodnie z zapisanym kodem programu,
 - obserwuje działanie programu napisanego w języku Python oraz analizuje wpływ poleceń na ruch robota i pracę manipulatora,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- dostrzega zależność między zapisanym programem a efektem działania robota podczas transportu ładunku.
- **Uczeń analizuje działanie programu i jego efekty:**
 - sprawdza poprawność działania programu sterującego ruchem robota i manipulatorem w języku Python,
 - identyfikuje błędy składniowe i logiczne oraz modyfikuje program poprzez poprawę zapisu poleceń lub parametrów ruchu,
 - wykorzystuje biblioteki i moduły służące do sterowania silnikami LEGO SPIKE Prime (motor_pair, motor).
- **Uczeń rozwija umiejętność testowania i doskonalenia rozwiązań:**
 - wykonuje próby działania programu i analizuje zachowanie robota podczas realizacji zadania transportowego,
 - wprowadza poprawki w celu uzyskania poprawnego działania robota oraz manipulatora,
 - testuje różne parametry ruchu (czas, stopnie obrotu, prędkość) i porównuje uzyskiwane efekty.

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści kształcenia będą realizowane za pomocą robotyki edukacyjnej LEGO® Education SPIKE™ Prime.

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna RaP STEAM,
- praca na komputerze,
- praca przy zestawie robotyki edukacyjnej,
- pogadanka.

Środki dydaktyczne

- zestaw PC z dostępem do Internetu,
- zestaw robotyki edukacyjnej LEGO® Education SPIKE™ Prime oraz zestaw rozszerzony LEGO® Education SPIKE™ Prime Expansion Kit,
- rzutnik multimedialny,
- e-podręcznik.

Formy pracy

- Praca indywidualna lub grupowa.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przebieg lekcji:

Faza I. Wprowadzenie

Na początku lekcji nauczyciel wprowadza uczniów w zagadnienia związane z **programowaniem robota dostawcy w języku Python**, wyjaśniając, że robot może wykonywać **złożone zadania transportowe dzięki odpowiedniemu sterowaniu silnikami oraz manipulatorem**. Następnie odnosi się do zadania lekcyjnego, polegającego na realizacji programu odpowiedzialnego za transport ładunku oraz przeprowadzeniu testów praktycznych działania robota.

Nauczyciel przedstawia cel i zakres zadania. Prezentuje uczniom środowisko programowania w języku Python w aplikacji LEGO SPIKE App oraz omawia sposób korzystania z bibliotek i modułów odpowiedzialnych za sterowanie robotem (**motor_pair, motor**). Następnie przekazuje uczniom niezbędne informacje dotyczące jazdy robota, wykonywania skrętów, sterowania manipulatorem oraz synchronizacji ruchu robota **z pracą dodatkowego silnika**.

Uczniowie zapoznają się ze sposobami zapisu poleceń służących do sterowania ruchem robota i manipulatora oraz obserwują, jak program po uruchomieniu realizuje kolejne etapy zadania transportowego.

Następnie uczniowie zdobywają umiejętności obejmujące:

- określenie problemu (w jaki sposób sterować robotem i manipulatorem w języku Python),
- analizę działania programu odpowiedzialnego za transport ładunku,
- wykorzystanie bibliotek i modułów służących do sterowania silnikami,
- obserwację działania robota podczas realizacji programu,
- testowanie programu i analizę poprawności wykonywanych ruchów,
- wprowadzanie prostych poprawek związanych z działaniem programu, prędkością ruchu oraz pracą manipulatora.

Na zakończenie tej fazy lekcji nauczyciel podaje temat do zapisania w zeszycie i przechodzi do fazy II – realizacji praktycznej.

Faza II. Część zasadnicza - realizacja

W tej fazie lekcji nauczyciel zapewnia uczniom praktyczne ćwiczenia, które pozwolą eksperymentować z koncepcjami programowania. W szczególności obejmować to będzie używanie blokowych języków programowania oraz zestawu do robotyki.

1. Nauczyciel objaśnia sposób pracy na lekcji – we wskazanym przez niego momencie uczniowie będą wykonywać kolejne etapy prowadzące do rozwiązania zadania. Zdolniejsi uczniowie spróbują wykonać je samodzielnie. Słabsi wykonają kolejne kroki razem z nauczycielem, w oparciu o przygotowaną do zajęć instrukcję dostępną w e-

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- podręczniku lub materiał multimedialny. W razie potrzeby będą mogli dopytać nauczyciela. Nauczyciel na bieżąco sprawdza postępy uczniów.
2. Bazując na posiadanych zasobach, zapewnij każdemu uczniowi lub zespołowi uczniów kompletny zestaw do robotyki (tj. LEGO® Education SPIKE™).
 3. Zestawy te pozwalają uczniom budować własne roboty i programować je do wykonywania określonych zadań. Nauczyciel zaczyna od prostych projektów, takich jak zbudowanie i zaprogramowanie robota, który może wykonać proste czynności, które wprowadzą robota w ruch.
 4. Uczniowie budują bazowego robota według załączonej instrukcji (załącznik **Budowa_bazy_robota.pdf** do lekcji).
 5. Uczniowie włączają program **SPIKE**, poprzez który będą programować robota.
 6. Nauczyciel, wykorzystując e-platformę, demonstruje na tablicy multimedialnej działanie programu i robota, uruchamiając odpowiedni moduł i materiał multimedialny.
 7. Prezentowany materiał multimedialny, zrealizowany w oparciu o szczegółowy scenariusz (załącznik do lekcji), oprócz waloru teoretycznego stanowi również podstawę przebiegu części praktycznej lekcji. W szczególności, uczniowie:
 - a. budują prosty model bazowego robota w oparciu o zestaw do robotyki (tj. LEGO® Education SPIKE™),
 - b. tworzą listę czynności niezbędnych do wprowadzenia robota w ruch,
 - c. na podstawie listy czynności przygotowują program, zapisują go we wskazanym folderze (nauczyciel wskazuje właściwą lokalizację; może to być dysk lokalny lub zasób w chmurze) i potrafią go uruchomić;
 - d. wgrywają program do zbudowanego modelu robota, uruchamiają, obserwują i testują działanie robota. Wprowadzają modyfikacje w programie, poprawiają ewentualne błędy.

Faza III. Podsumowanie

Uczniowie analizują przedstawione przykłady programów zapisanych w języku Python oraz omawiają ich działanie podczas **realizacji zadania transportowego**. Nauczyciel ocenia stopień zrozumienia zależności między zapisanym kodem a zachowaniem robota, pracą manipulatora oraz wykonywanymi ruchami po uruchomieniu programu.

Kryteria oceniania:

Ocena dopuszczająca:

- uczeń rozpoznaje podstawowe elementy programu w języku Python,
- potrafi wskazać polecenia odpowiedzialne za sterowanie ruchem robota oraz manipulatorem.

Ocena dostateczna:

- uczeń analizuje przykładowy program i opisuje jego działanie,
- rozumie zależność między zapisanym kodem a działaniem robota,
- rozpoznaje podstawowe biblioteki i moduły wykorzystywane do sterowania silnikami,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- wskazuje podstawowe błędy składniowe.

Ocena dobra:

- uczeń samodzielnie modyfikuje przykładowy program,
- poprawia błędy związane z zapisem poleceń oraz parametrami ruchu robota,
- testuje działanie programu i analizuje poprawność wykonywania ruchów po wprowadzonych zmianach.

Ocena bardzo dobra:

- uczeń samodzielnie tworzy poprawny program w języku Python,
- wyjaśnia działanie poszczególnych elementów programu,
- wykorzystuje biblioteki i moduły do sterowania robotem oraz manipulatorem LEGO SPIKE Prime,
- proponuje własne rozwiązania i potrafi je uzasadnić,
- samodzielnie analizuje i poprawia błędy w kodzie oraz optymalizuje działanie robota podczas realizacji zadania transportowego.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, np.:

1. Co oznacza ujemna wartość stopni w funkcji ruchu?
 - a. **Jazdę do tyłu**
 - b. Jazdę szybciej
 - c. Zatrzymanie robota
 - d. Skręt w miejscu
2. Co oznacza dodatnia wartość parametru skrętu w funkcji ruchu?
 - a. Skręt w lewo
 - b. Jazdę do tyłu
 - c. **Skręt w prawo**
 - d. Zatrzymanie robota
3. Dlaczego w programie stosuje się różne wartości **velocity**?
 - a. Aby zmienić kolor robota
 - b. **Aby dostosować prędkość robota do zadania**
 - c. Aby zatrzymać program
 - d. Aby wyłączyć silniki



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Faza V. Dodatkowe, praca domowa (dla chętnych).

Zadania dodatkowe (dla chętnych).

Zadanie 1: Magazyn autonomiczny – transport ładunku

Utwórz program w języku Python, który pozwoli robotowi przetransportować ładunek z jednego miejsca do drugiego z wykorzystaniem manipulatora.

Program powinien:

- zawierać poprawnie zaimportowane biblioteki i moduły (motor_pair, motor),
- umożliwiać jazdę robota do przodu i do tyłu,
- wykonywać co najmniej jeden skręt podczas przejazdu,
- sterować manipulatorem odpowiedzialnym za podnoszenie lub opuszczanie ładunku,
- poprawnie realizować kolejność działań robota,
- być zapisany poprawnie pod względem składni.

Opcjonalnie możesz rozbudować program tak, aby:

- robot transportował więcej niż jeden ładunek,
- wykorzystywał różne prędkości ruchu (velocity),
- wykonywał dodatkowe manewry cofania lub parkowania,
- wyświetlał komunikaty na matrycy Spike Hub,
- zawierał własne usprawnienia zwiększające precyzję działania robota.



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 19 - Robot dostawca – realizacja programu i testy praktyczne.

Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
 - Nauczyciel omawia podstawowe zagadnienia związane z programowaniem **robotu dostawcy w języku Python**:
 - wyjaśnia, że robot może wykonywać zadania transportowe dzięki odpowiedniemu sterowaniu silnikami i manipulatorem,
 - przedstawia przykłady bibliotek i modułów wykorzystywanych do sterowania robotem LEGO SPIKE Prime (**motor_pair, motor**),
 - omawia znaczenie poprawnego sterowania ruchem robota oraz synchronizacji pracy manipulatora.
 - Wprowadzenie do zadania i przypomnienie podstaw pracy w LEGO SPIKE Prime:
 - omówienie środowiska programowania Python w aplikacji LEGO SPIKE App,
 - przedstawienie bibliotek i modułów odpowiedzialnych za sterowanie silnikami oraz manipulatorem,
 - przedstawienie zależności: zapis kodu >> uruchomienie programu >> ruch robota i działanie manipulatora,
 - analiza przykładowych programów wykorzystujących jazdę robota, skręty oraz podnoszenie i transport ładunku.
 - Praca uczniów:
 - analiza przedstawionych przykładów programów sterujących robotem,
 - omawianie działania poszczególnych linii kodu,
 - testowanie działania robota i manipulatora po uruchomieniu programu,
 - wyszukiwanie i poprawianie błędów w kodzie,
 - proponowanie własnych rozwiązań usprawniających transport ładunku i precyzję ruchu robota.
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów (**analiza poprawności działania programu, ruchu robota, pracy manipulatora oraz skuteczności realizacji zadania transportowego**).
- Zadanie pracy domowej (opcjonalnie) – **napisanie prostego programu sterującego robotem dostawcą oraz manipulatorem z wykorzystaniem bibliotek motor_pair i motor**.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 2 – Realizacja zadania głównego podczas lekcji nr 19 - *Robot dostawca – realizacja programu i testy praktyczne.*

Zadanie 1. Przygotowanie do realizacji zadania głównego

Uczniowie, pracując w zespołach lub indywidualnie, realizują zadanie polegające na poznaniu sposobu programowania robota dostawcy w języku Python oraz **sterowaniu ruchem robota i manipulatora w środowisku LEGO SPIKE Prime**. Szczególny nacisk położony jest na poprawne wykorzystanie bibliotek i modułów (**motor_pair**, **motor**), analizę działania programu oraz interpretację wyników działania robota podczas realizacji zadania transportowego.

Uczniowie opracowują i zapisują rozwiązanie problemu w postaci **programów tekstowych**, które umożliwią **sterowanie ruchem robota, wykonywanie skrętów oraz obsługę manipulatora odpowiedzialnego za transport ładunku**. W tym celu:

- analizują przykładowe programy zapisane w języku Python,
- określają cel programu oraz przewidują możliwe błędy składniowe i problemy związane z ruchem robota lub pracą manipulatora,
- planują kolejność wykonywania instrukcji w programie,
- opracowują algorytm w uporządkowanej formie,
- zapisują go jako program w języku Python wykorzystujący biblioteki **motor_pair** i **motor**,
- testują działanie programu oraz analizują zachowanie robota podczas transportu ładunku,
- wprowadzają poprawki związane z działaniem programu, parametrami ruchu lub pracą manipulatora,
- doprecyzowują rozwiązanie tak, aby kod był czytelny i poprawny składniowo.

Uczniowie wykorzystują środowisko LEGO SPIKE Prime w trybie Python, zwracając uwagę na zgodność między zapisanym kodem a rzeczywistym działaniem robota. Szczególną uwagę poświęcają poprawności składni, wykorzystaniu funkcji sterujących ruchem robota i manipulatora oraz analizie komunikatów o błędach pojawiających się po uruchomieniu programu.

Jeśli jest taka potrzeba, wskazana przez nauczyciela, to uczniowie zbudują robota od podstaw zgodnie z poniższą instrukcją.

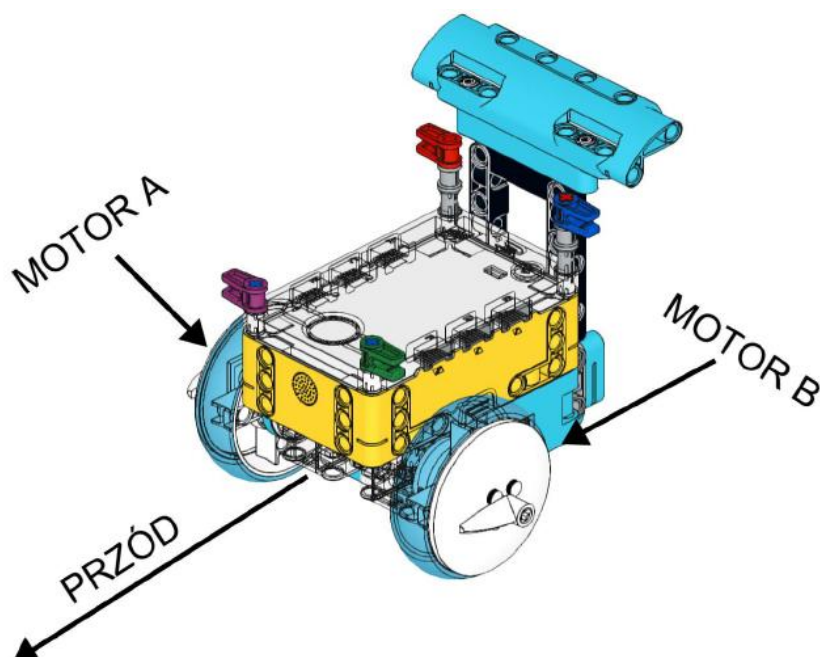
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Budowa bazowego robota

Załącznik nr 2 – Przygotowanie do realizacji zadania głównego

Budowa bazowego robota

Uruchom plik **Budowa bazy robota.pdf** znajdujący się na platformie i postępuj wg podanych w pliku kroków. Efektem końcowym będzie fizyczny robot gotowy do programowania. **Jeśli konstrukcja jest gotowa – idź dalej.**



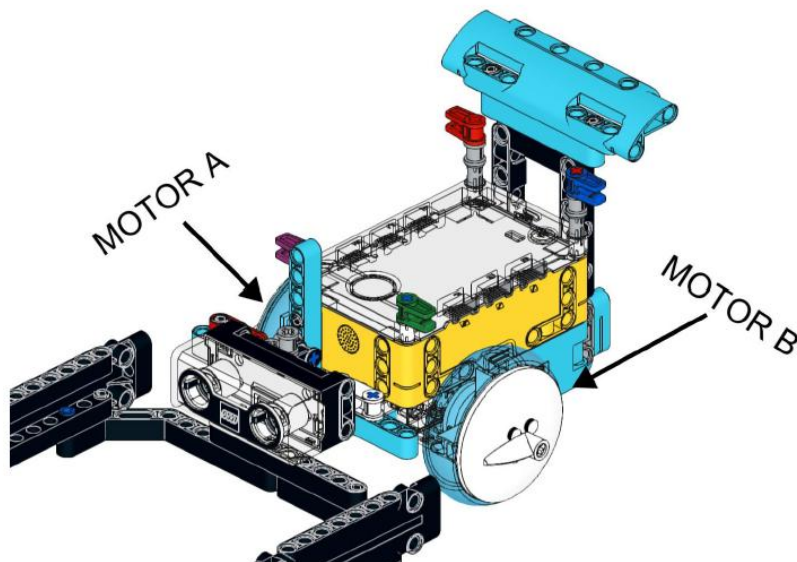
Pudełka niezbędne do realizacji lekcji: **17, 20**



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

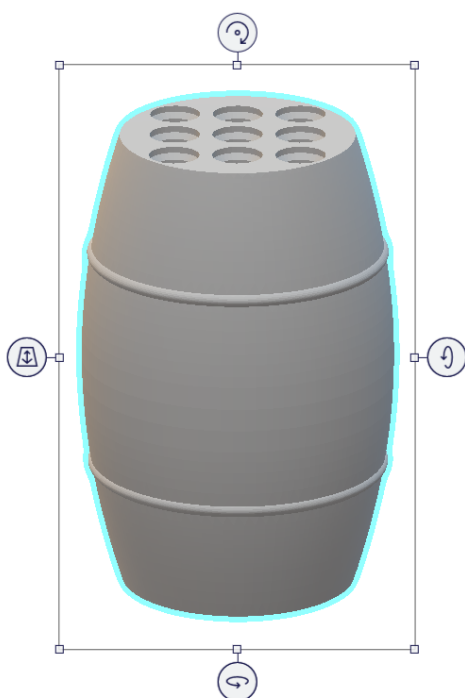
Dodatkowe instrukcje niezbędne do realizacji lekcji:

- **robotLapanie.pdf**



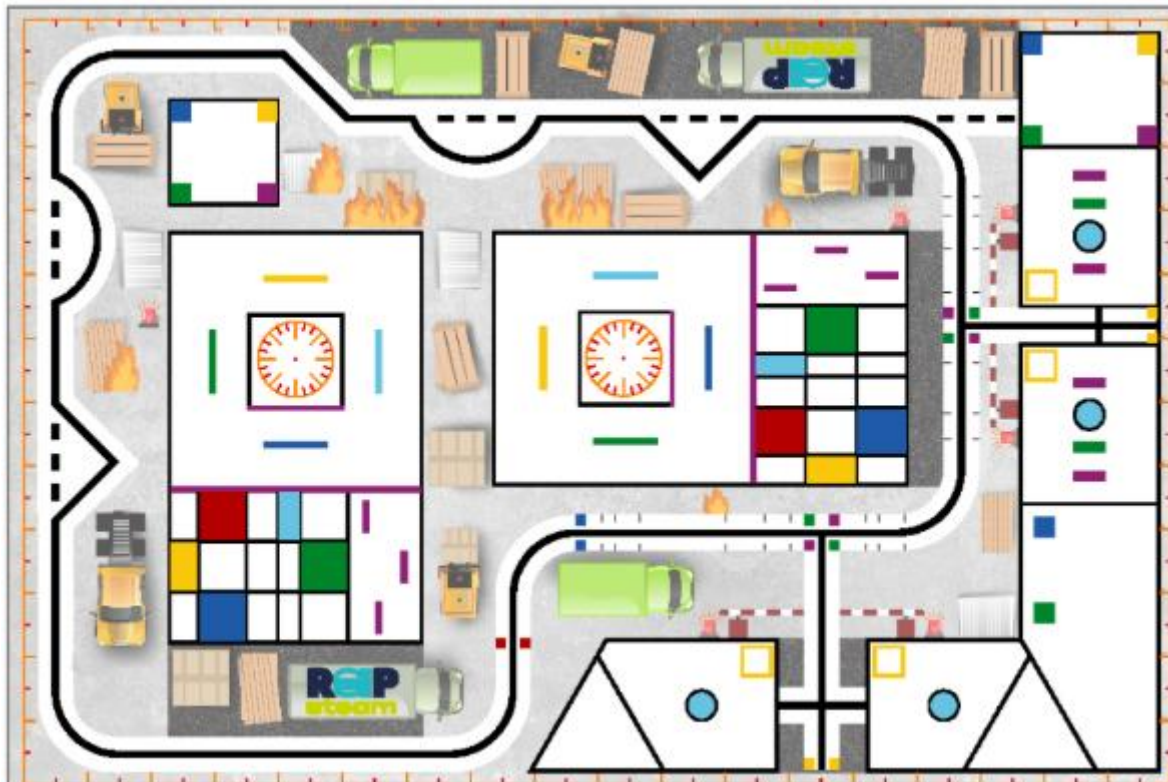
Dodatkowe elementy niezbędne do realizacji lekcji:

- **Wydruk 3D – Beczka.3mf, sztuk 1**



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Do realizacji scenariusza zajęć będziesz potrzebował również matę:

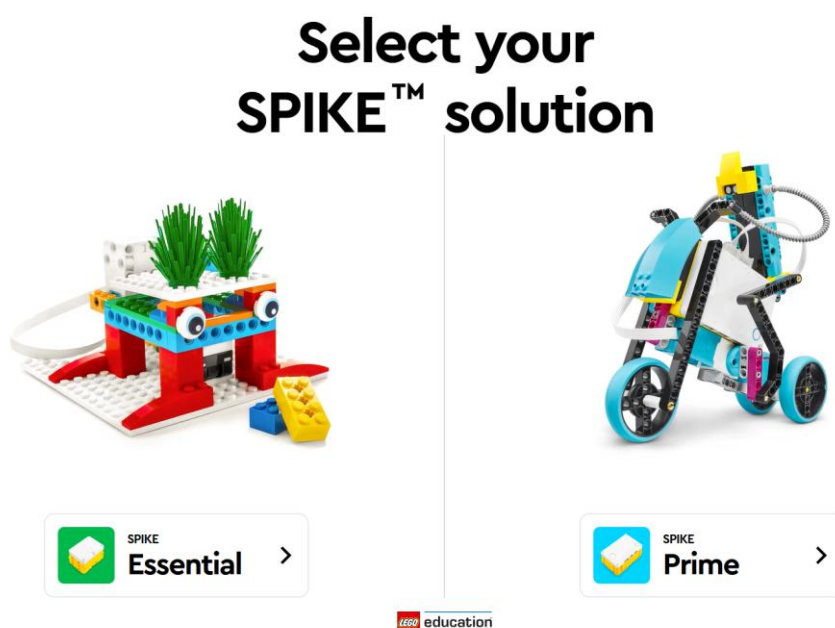


Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 2. Konfiguracja środowiska.

Uruchamianie aplikacji LEGO Spike

1. Włącz przeglądarkę internetową dostępną na twoim urządzeniu
2. Wpisz w przeglądarce adres: <https://spike.legoeducation.com/> Pojawi się okno wyboru wersji LEGO SPIKE. Wybierz Spike Prime

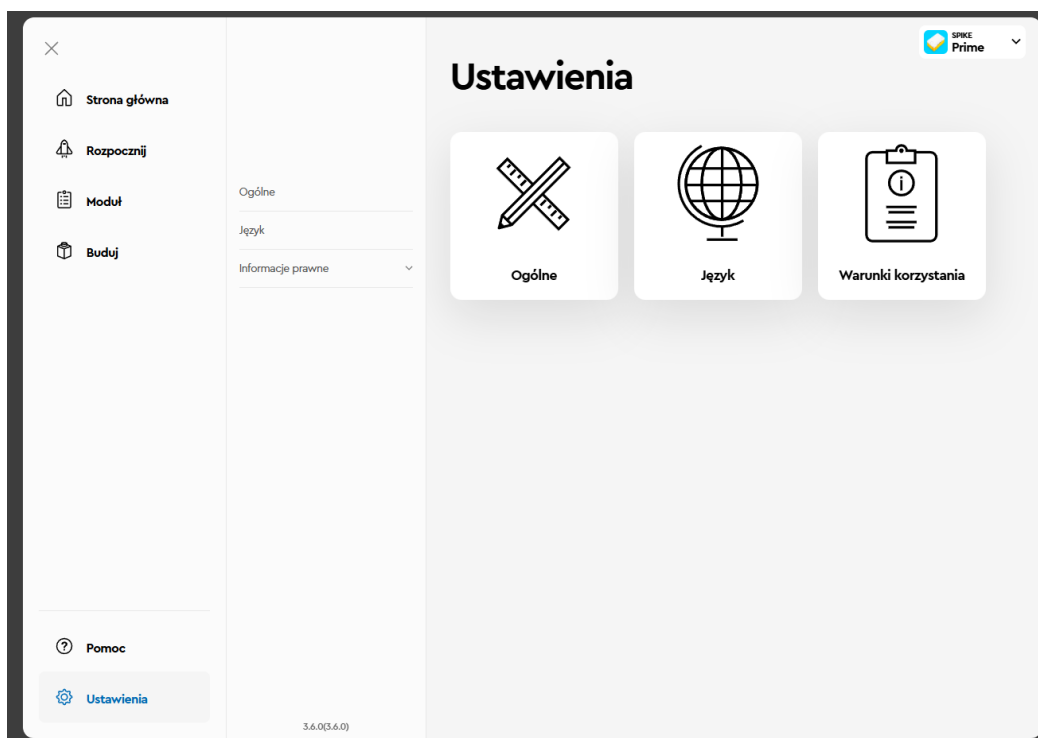


3. Po wybraniu opcji SPIKE Prime pojawia się menu główne. Jeśli występuje język polski, należy zmienić go na język angielski.

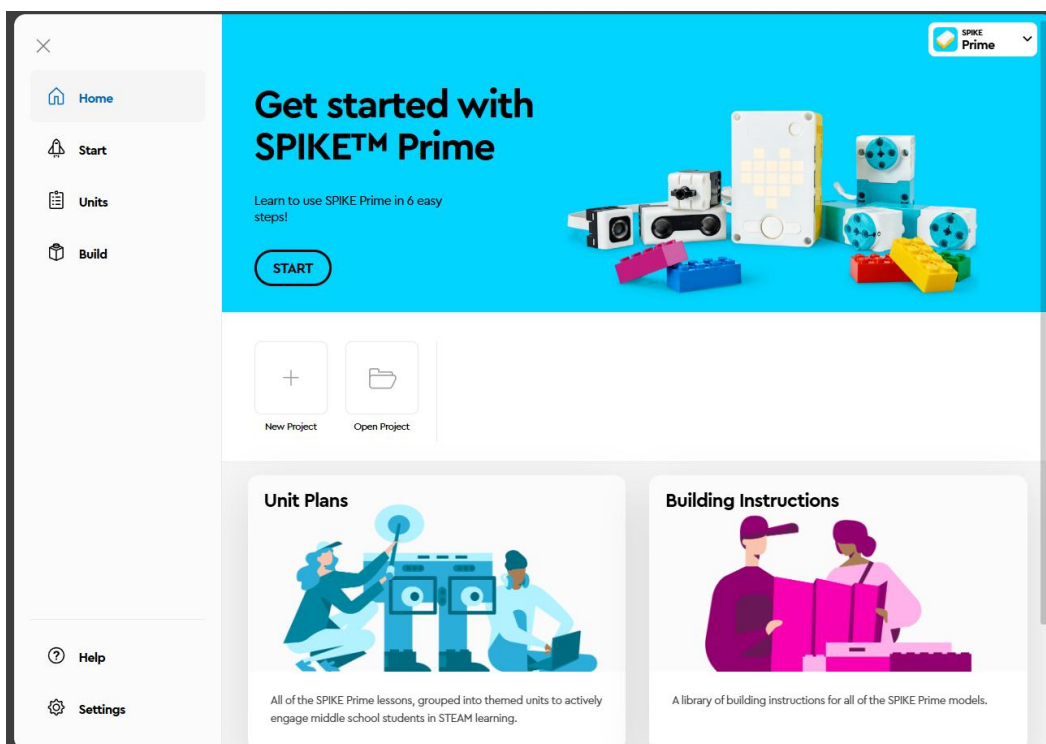
Jeżeli zaś wszystko jest w **języku angielskim**, to przejdź do kroku 6.

Aby zmienić język, w lewym dolnym rogu menu głównego kliknij przycisk **Ustawienia**.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

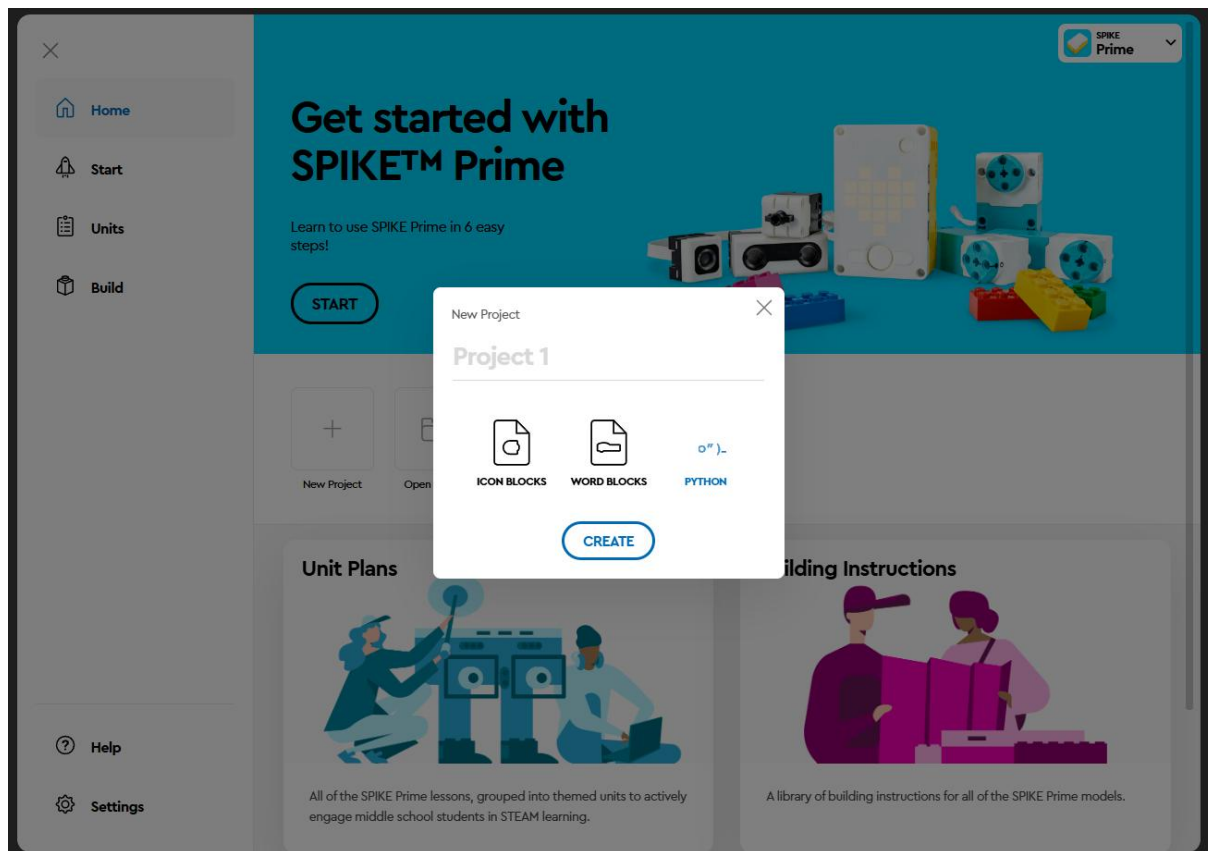


4. W nowym oknie kliknij **globus** podpisany „Language”.
5. **Wybierz język: United States (English)**. Strona przeładuje się i język angielski będzie dostępny.
6. Kliknij po lewej przycisk **Home**. Pojawi się okno jak poniżej

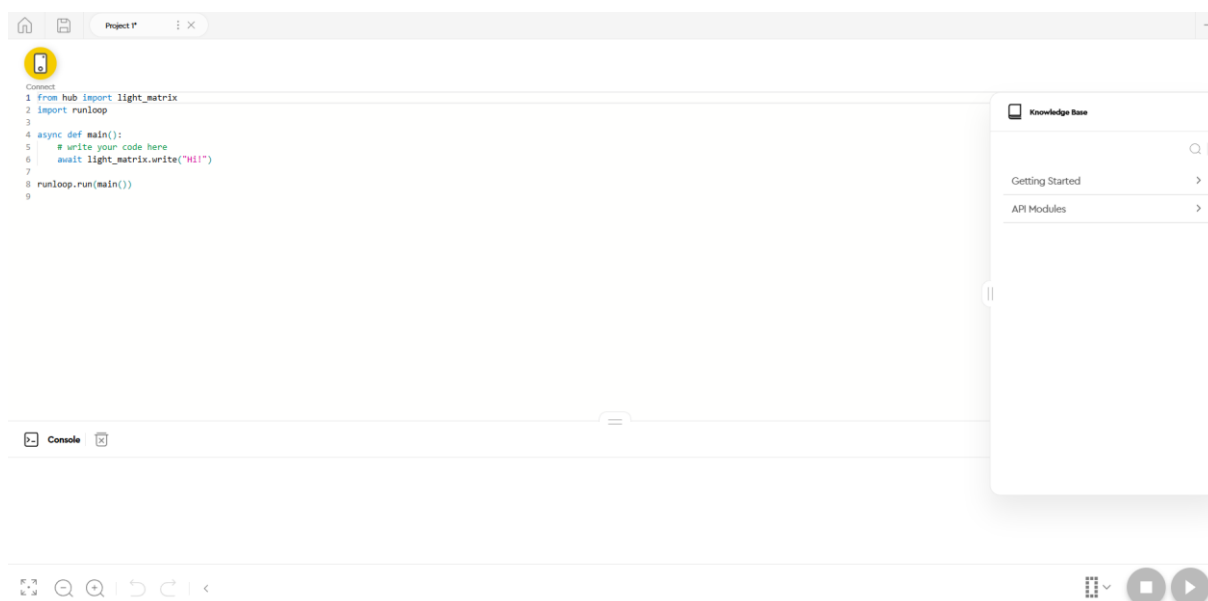


Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

7. Kliknij na środku przycisk **New Project**. Pojawi się nowe okno jak poniżej



8. Wybierz opcję **Python** oraz podaj nazwę projektu. Kiedy wszystko zostanie poprawnie wybrane, wciśnij przycisk **Create**.

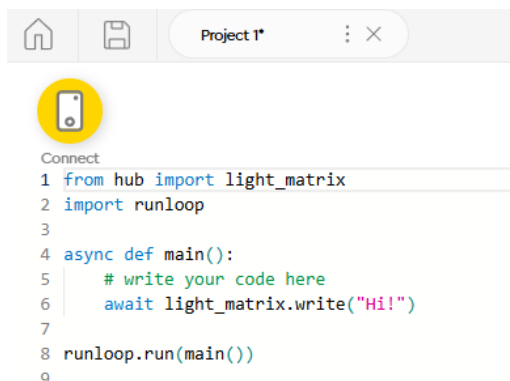


Wszystko jest gotowe do programowania. Pora na podłączenie kontrolera Spike Hub.

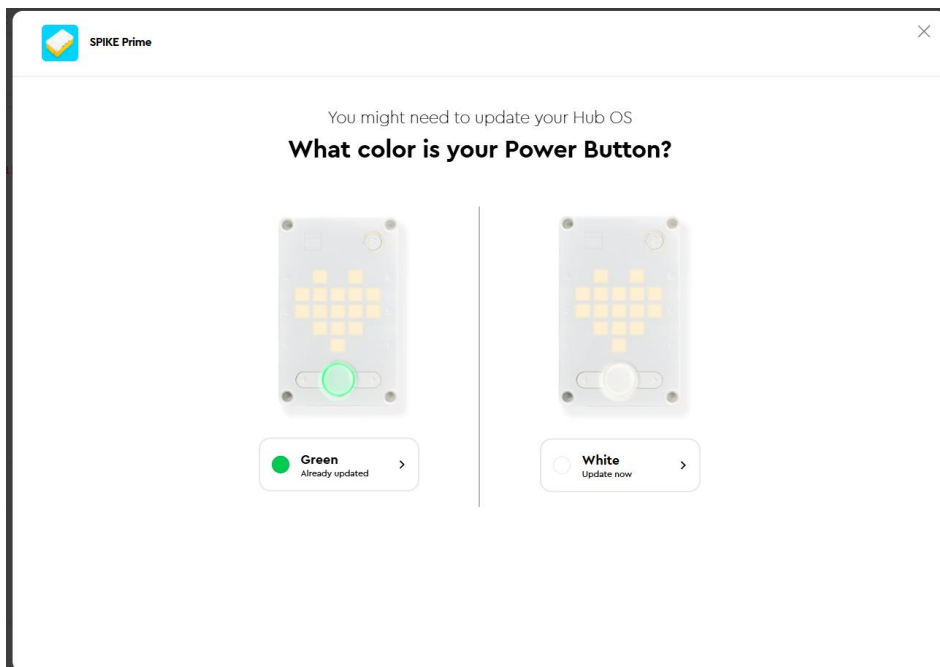
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 3. Podłączenie kontrolera Spike Hub

1. W głównym programie kliknij przycisk **Connect** znajdujący się w górnym rogu, tuż pod nazwą twojego projektu. Patrz poniżej.

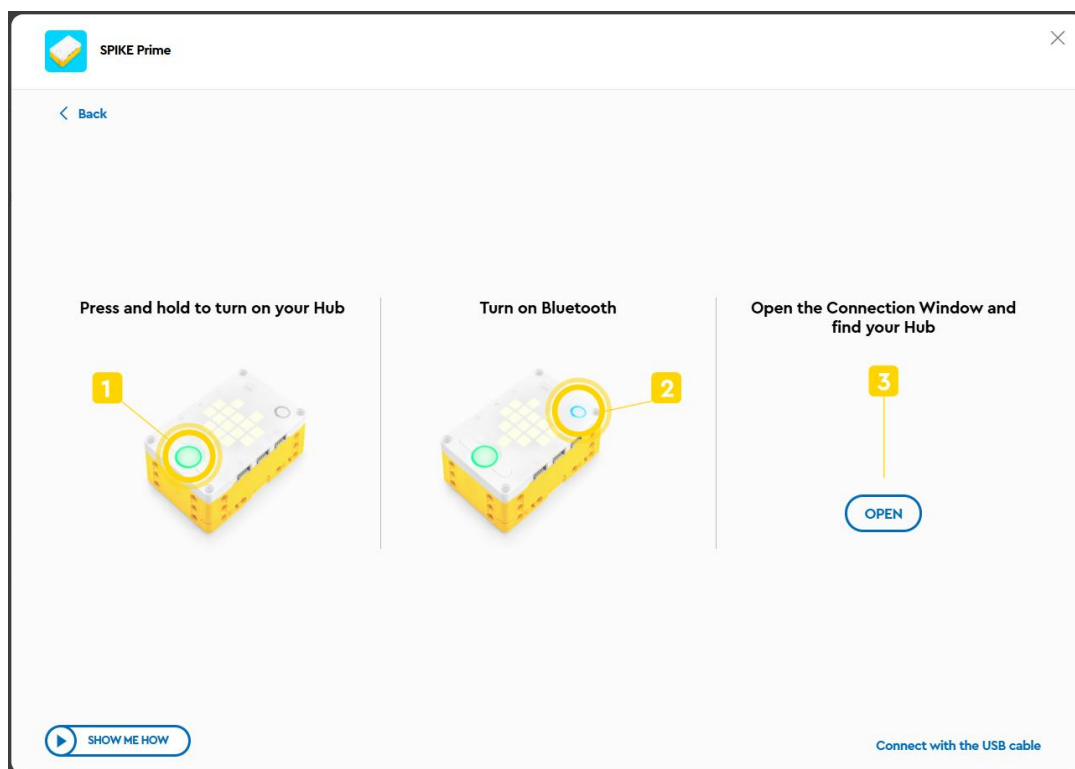


2. Pojawi się nowe okno. **Włącz kontroler Spike Hub** klikając **duży biały przycisk** i poczekaj aż się uruchomi. Następnie wybierz jedną z opcji odpowiadającą stanowi kontrolera Spike Hub – zawsze będzie to **Green – Already updated**.

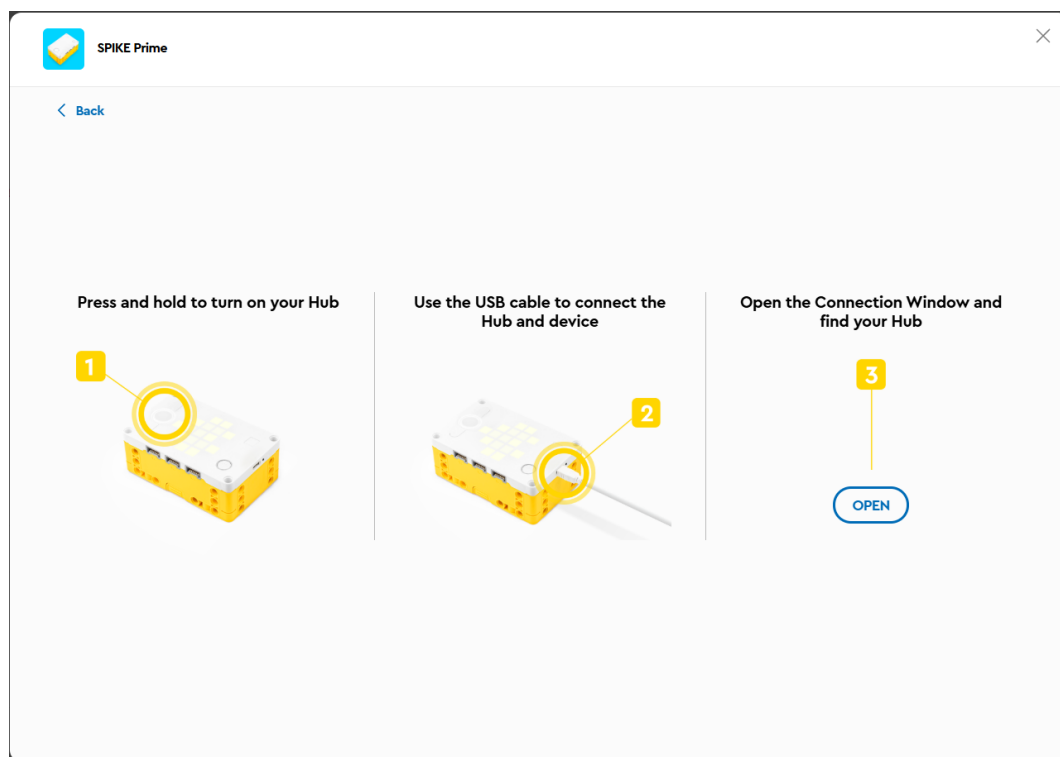


3. Pojawi się nowe okno (patrz poniżej). Wybierz opcję **Connect with USB cable** znajdującą się w prawym dolnym rogu okna.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

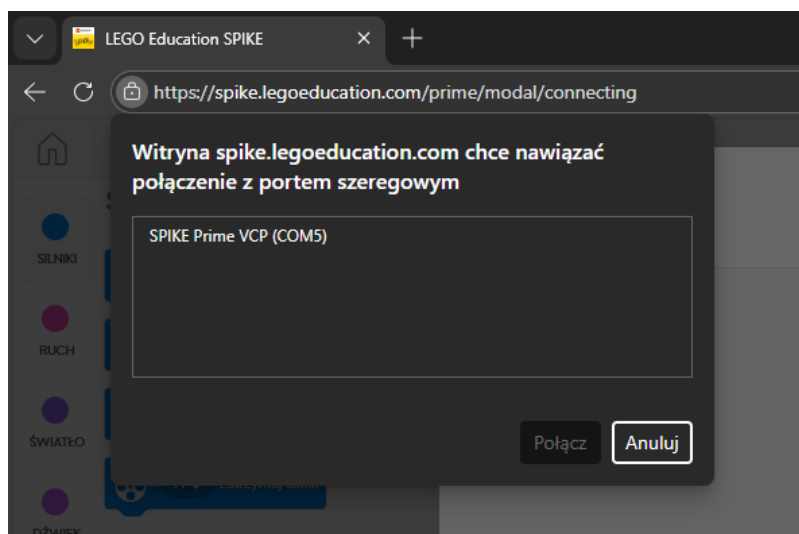


4. Postępuj zgodnie z instrukcją na ekranie. Po przeczytaniu kliknij przycisk **Open**.



5. Pojawi się nowe okno (patrz poniżej)

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



6. Wybierz opcję **SPIKE Prime VCP** i kliknij **Połącz**.

UWAGA – u Ciebie nazwa może nie być identyczna jak na rysunku powyżej. Wybierz tę opcję, która ma w nazwie **Spike Prime**. Innymi rzeczami się nie przejmuj.

Jeśli wszystko zostanie poprawnie połączone, to okna zostaną automatycznie zamknięte, a ikona kontrolera w programie zmieni się na zieloną.



```
1 from hub import light_matrix
2 import runloop
3
4 async def main():
5     # write your code here
6     await light_matrix.write("Hi!")
7
8 runloop.run(main())
9
```

Gratulacje. Twój Spike Hub jest podłączony do komputera. Możesz przystąpić do programowania robota.

W przypadku problemów postępuj wg kroków:

1. Sprawdź, czy kontroler Spike Hub jest włączony i świeci na zielono
 - a. Jeśli kontroler świeci na biało zgłoś to nauczycielowi
 - b. Jeśli kontroler świeci na zielono to podążaj dalej wg instrukcji łączenia powyżej
2. Sprawdź czy kabel USB jest podłączony (Spike Hub <-> Komputer)
3. Sprawdź, czy wybrano odpowiednią nazwę urządzenia – w oknie połączenia z portem szeregowym należy wybrać urządzenie, które w nazwie ma SPIKE Prime.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 4. Realizacja zadania głównego

W tej lekcji wykonasz zadanie, które opracowano na poprzedniej lekcji. Należy zaprogramować robota tak aby wykorzystując chwytak przewiózł ładunek (beczki) z punktu A (strefy załadunku) do punktu B (strefy magazynowej) wykonując specjalistyczne ruchy w tym:

- Jazda przód / tył
- Skręt w czasie jazdy w wybranym kierunku

Przykład realizacji zadania – wprowadzenie bibliotek

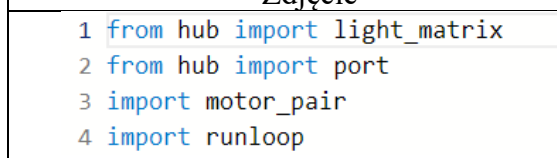
UWAGA – w materiałach stosowane są komentarze. Na zrzutach ekranu jest to tekst oznaczony jako zielony. Tego tekstu nie przepisuj! To jest tylko wytłumaczenie dla Ciebie, co dzieje się w danej części programu. Przepisuj tylko i wyłącznie tekst, który nie jest zielony. Korzystaj z tabeli. Pamiętaj, że wielkość liter MA ZNACZENIE!

Na początku programu należy dodać odpowiednie odwołania (biblioteki) do modułów kontrolera Spike Hub i czujników. Dodaj moduł, biblioteki:

- port (moduł),
- light_matrix (moduł),
- motor_pair (biblioteka),

Aby móc programować silniki, niezbędne jest zaimportowanie biblioteki **motor_pair**. Pozwala ona, tak jak w Scratchu za pomocą różowych bloków sterować dwoma silnikami jednocześnie. Podczas dzisiejszych zajęć wykorzystasz też moduł **port** oraz **light_matrix** (dostęp do portów oraz wyświetlanie informacji na matrycy Spike Hub).

Zaktualizuj początek programu i dodaj odpowiednie odwołania. Część programu odpowiedzialna za biblioteki powinna wyglądać następująco:

Zdjęcie	Kod (tekst)
 <pre> 1 from hub import light_matrix 2 from hub import port 3 import motor_pair 4 import runloop </pre>	<pre> from hub import light_matrix from hub import port import motor_pair import runloop </pre>

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

W tej chwili cały program powinien wyglądać następująco:

```
1 from hub import light_matrix
2 from hub import port
3 import motor_pair
4 import runloop
5
6 async def main():
7     # write your code here
8     await light_matrix.write("Hi!")
9
10 runloop.run(main())
```

Przygotowanie struktury programu

Dzisiejsze zajęcia zacznij od przygotowania podstawowych ustawień i parametrów dla kontrolera Spike Hub i silników. Dodaj komendę, która przypisze silniki podpięte do portów B i A do zmiennej PAIR_1 – tak jak na poprzednich lekcjach. Następnie uruchom silniki tak aby robot:

- Pojechał do przodu 360 stopni, bez skręcania

Możesz zostawić komendę odpowiadającą za wyświetlanie napisu „Hi!” na kontrolerze Spike Hub. Jeśli chcesz ją usunąć, to możesz to zrobić. Nie jest ona wymagana do dzisiejszych zajęć.

Program powinien wyglądać następująco:

```
1 from hub import light_matrix, port
2 import runloop
3 import motor_pair
4
5 async def main():
6     # write your code here
7     await light_matrix.write("Hi!")
8
9     #przypisanie portów A i B do PAIR_1
10    motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
11
12    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 360, 0)
13
14 runloop.run(main())
```

Uruchom program, wgrywając go do kontrolera Spike Hub, i sprawdź, czy robot wykonuje poprawnie swoje zadanie. Jeśli wszystko działa poprawnie, przejdź do dalszej lekcji.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Realizacja głównego zadania

Zrealizuj zadanie, którego opracowanie, dokumentację i wyliczenia przygotowano na poprzednich zajęciach. Zgłoś się do nauczyciela po odpowiednie elementy. W dzisiejszej lekcji dodatkowym modułem (poza robotem i jego modułami) będzie:

- Wydruk 3D – beczki, sztuk 1

Upewnij się, że masz go do dyspozycji. Następnie zaprogramuj robota wg ustaleń nauczyciela.

Przykład realizacji zadania

Przedstawiony poniżej przykład realizuje rozwiązanie głównego zadania. Robot wykonuje następujące zadania:

- Jedź do przodu przez 4 obroty i dodatkowo przez 120 stopni – dotrzesz do miejsca, gdzie należy się zatrzymać,
- Wykonaj obrót o ok. 90 stopni do tyłu, jednocześnie skręcając. Robot powinien wykonywać parkowanie tyłem. W przypadku przykładu będzie to skręt po łuku do tyłu w prawą stronę,
- Po osiągnięciu docelowej pozycji robot jedzie do przodu przez 3 obroty i dodatkowo 92 stopnie, po drodze łapiąc paletę z beczkami. W trakcie jazdy prosto robot zawozi paletę do miejsca docelowego (punkt B).

Do realizacji zadania wykorzystano funkcję **move_for_degrees()** podając jej odpowiednie parametry.

Aby poruszać się do przodu zastosowano następujący kod:

```
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1560, 0, velocity=500,  
deceleration=300)
```

W programie uwzględniono większą prędkość silników niż podstawową (**velocity=500**). Jest to dodatkowe działanie i nie wpływa na poprawne rozwiązanie całego zadania. Dodatkowo zastosowano parametr **deceleration** z wartością 300, aby robot, zbliżając się do końca trasy, zaczął łagodnie zwalniać. Tak samo jest to parametr dodatkowy.

Aby skręcić do tyłu po łuku w prawo zastosowano następujący kod:

```
await motor_pair.move_for_degrees(motor_pair.PAIR_1, -450, 30)
```

Robot wykonuje ruch, cofając się o 450 stopni (minus oznacza jazdę do tyłu) w prawo (30). Wartość dodatnia dla skrętu oznacza skręcanie w prawo. Ponieważ wartość skrętu jest niska, robot będzie się poruszał po łuku, wykonując łagodne cofanie.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Ostatnim kodem jest jazda do przodu:

```
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1172, 0,
deceleration=300, velocity=500)
```

W tej części kodu robot wykonuje ruch do przodu o wartości 1172 stopni, jadąc prosto, bez skręcania (0). Zastosowano dodatkowe parametry w postaci zwolnienia przy osiągnięciu mety (**deceleration=300**) oraz zwiększonej prędkości napędowej silników (**velocity=500**). Są to parametry opcjonalne – nie są wymagane do realizacji zadania, choć można je użyć.

Cały program wygląda następująco:

```
1 from hub import light_matrix, port
2 import runloop
3 import motor_pair
4
5 async def main():
6     # write your code here
7     await light_matrix.write("Hi!")
8
9     #przypisanie portów A i B do PAIR_1
10    motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
11
12    #jazda do przodu bez skręcania (0) przez ponad 4 obroty (360*4 + 120 = 1440 + 120 = 1560)
13    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1560, 0, velocity=500, deceleration=300)
14
15    #skręcanie w prawo przez -450 stopni z pełnym skretem 30 ze standardową prędkością obrotu (wartość domyślna to 360 więc nie wpisujemy)
16    await motor_pair.move_for_degrees(motor_pair.PAIR_1, -450, 30)
17
18    #jazda do przodu bez skręcania (0) przez ponad 3 obroty (3*360 + 92 = 960 + 92 = 1172)
19    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1172, 0, deceleration=300, velocity=500)
20
21 runloop.run(main())
```

Spróbuj zrealizować swoje zadanie, bazując na pozyskanej do tej pory wiedzy. Powodzenia i do następnej lekcji!

Możliwe rozszerzenia zadania

1. Ustaw beczkę w innym miejscu, zastosuj skręt w prawo.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 5: Odkrywamy tajemnice Kodu... Robot eksplorator

Temat lekcji nr 20 - Kontrola dodatkowych silników. Przenoszenie ładunków na różne wysokości. Analiza problemu

Treści nauczania (wymagania szczegółowe - podstawa programowa) - I. 5); IV. 4) II. 3) b); II. 3) c); I. 1).

Cel lekcji:

- **Wprowadzenie uczniów do planowania pracy** z wykorzystaniem dokumentu tekstowego oraz narzędzi TechOffice w kontekście zadania robota eksploratora.
- **Przygotowanie uczniów do tworzenia przejrzystego planu realizacji zadania** związanego z **kontrolą dodatkowych silników** podczas **przenoszenia ładunków na różne wysokości**.
- **Rozwijanie umiejętności zapisywania kolejnych etapów pracy**, formułowania celów oraz porządkowania działań w logicznej kolejności z uwzględnieniem **kontroli dodatkowych silników** podczas **przenoszenia ładunków na różne wysokości**.
- **Kształtowanie kompetencji pracy z edytorem tekstu**, w tym tworzenia i edytowania dokumentów, zapisywania informacji oraz ich czytelnego formatowania z wykorzystaniem narzędzi TechOffice.
- **Uświadomienie uczniom znaczenia planowania** jako etapu poprzedzającego programowanie i **wpływającego na skuteczność działania robota eksploratora** podczas realizacji zadania przenoszenia ładunków na różne wysokości.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- **Uczeń w algorytmicznym rozwiązywaniu problemu wyróżnia podstawowe kroki:**
 - tworzy plan działania robota eksploratora z wykorzystaniem manipulatora oraz narzędzi TechOffice,
 - określa problem i cel zadania związany z **kontrolą dodatkowych silników** podczas **przenoszenia ładunków na różne wysokości**,
 - opracowuje uporządkowaną listę działań prowadzących do realizacji zadania.
- **Uczeń analizuje informacje i formułuje wnioski:**

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- analizuje trasę przejazdu robota oraz działanie manipulatora podczas **przenoszenia ładunków na różne wysokości**,
- podejmuje decyzje dotyczące sposobu działania robota,
- wykorzystuje narzędzia cyfrowe, w tym TechOffice, do opracowania rozwiązania problemu.
- **Uczeń formułuje i zapisuje w postaci algorytmów:**
 - kolejne etapy pracy prowadzące do rozwiązania problemu,
 - logiczną sekwencję działań robota związaną z **kontrolą dodatkowych silników podczas przenoszenia ładunków na różne wysokości**,
 - plan działań możliwy do późniejszego wykorzystania podczas programowania robota eksploratora.

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści kształcenia będą realizowane za pomocą robotyki edukacyjnej LEGO® Education SPIKE™ Prime.

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna RaP STEAM,
- praca na komputerze,
- praca przy zestawie robotyki edukacyjnej,
- pogadanka.

Środki dydaktyczne

- zestaw PC z dostępem do Internetu,
- zestaw robotyki edukacyjnej LEGO® Education SPIKE™ Prime oraz zestaw rozszerzony LEGO® Education SPIKE™ Prime Expansion Kit,
- rzutnik multimedialny,
- e-podręcznik,
- mata/tor z wyznaczoną trasą,
- kartki papieru i ołówki.

Formy pracy

- Praca indywidualna lub zespołowa.

Przebieg lekcji:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Faza I. Wprowadzenie

Nauczyciel rozpoczyna lekcję od przedstawienia tematu i celu zajęć, którymi są analiza zadania związanego z **kontrolą dodatkowych silników podczas przenoszenia ładunków na różne wysokości**, oraz **opracowanie planu działania robota w dokumencie tekstowym**. Podkreśla znaczenie planowania, logicznego myślenia oraz analizy informacji potrzebnych do realizacji zadania.

Następnie nauczyciel prezentuje sytuację zadaniową polegającą na **przejeździe robota po wyznaczonej trasie, pobraniu elementów za pomocą manipulatora oraz podniesieniu ich do określonego miejsca**. Omawia główne elementy zadania (punkt startowy, trasa przejazdu, sposób transportu elementów, działanie manipulatora oraz miejsce zakończenia zadania). Uczniowie analizują sytuację oraz dostępne informacje, co stanowi podstawę do dalszej pracy.

Nauczyciel wskazuje, że kluczowym etapem będzie **opracowanie planu działania w dokumencie tekstowym** z wykorzystaniem narzędzi TechOffice. Uczniowie zapisują w nim:

- problem i cel działania robota,
- uporządkowaną listę kroków,
- plan przejazdu, transportu elementów, **kontrolę dodatkowych silników podczas przenoszenia ładunków na różne wysokości**,
- wstępny algorytm rozwiązania.

Następnie uczniowie pracują w zespołach, dzieląc się rolami (np. programista, analityk, tester), i wspólnie opracowują rozwiązanie.

Na zakończenie tej części nauczyciel podkreśla, że przygotowany dokument będzie podstawą do dalszego etapu – programowania i testowania działania robota eksploratora.

Faza II. Część zasadnicza - realizacja

Uczniowie przechodzą do realizacji zadania, łącząc analizę w dokumencie tekstowym z tworzeniem programu i pracą z robotem.

1. Nauczyciel wyjaśnia sposób pracy, podkreślając, że podstawą działań jest **dokument tekstowy** (np. TechOffice), w którym uczniowie zapisują plan działania robota (problem, cel, kolejne kroki).
2. Uczniowie pracują w zespołach i opracowują plan rozwiązania zadania rozpoznania sytuacji i podjęcia decyzji przez robota.
3. Nauczyciel przypomina, że analiza informacji i poprawny plan są kluczowe przed rozpoczęciem programowania.
4. Uczniowie:
 - a. uzupełniają dokument o szczegółowe kroki działania,
 - b. tworzą schematy blokowe i/lub pseudokod,
 - c. weryfikują opracowane algorytmy

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

d. porównują efekty z planem i wprowadzają poprawki (algorytm + dokument).

Uczniowie uczą się, że proces ma charakter iteracyjny: analiza >> plan >> program >> test >> poprawa.

Faza III. Podsumowanie

Uczniowie prezentują swoje rozwiązania (dokument + program), a nauczyciel omawia ich działanie oraz ocenia, zwracając uwagę na analizę zadania, poprawność planu oraz zgodność działania robota z założeniami. Kryteria oceniania:

Ocena dopuszczająca:

- uczeń tworzy prosty plan działania w dokumencie lub programie, jednak rozwiązanie jest niepełne i nie prowadzi do realizacji zadania.
- dokument zawiera podstawowe informacje, ale jest nieuporządkowany lub nieczytelny.

Ocena dostateczna:

- uczeń modyfikuje algorytm, jednak działanie nie jest zgodne z założeniami zadania (np. błędna analiza lub decyzje),
- dokumentacja zawiera opis kolejnych działań robota, jednak występują w niej braki lub nieścisłości.

Ocena dobra:

- uczeń opracowuje plan i program umożliwiające działanie robota, jednak rozwiązanie zawiera nieścisłości lub nie w pełni uwzględnia warunki zadania,
- uczeń przygotowuje czytelną dokumentację zawierającą plan działania, pseudokod oraz opis kolejnych etapów realizacji zadania.

Ocena bardzo dobra:

- uczeń poprawnie analizuje zadanie, tworzy czytelny plan w dokumencie i program,
- uczeń testuje rozwiązanie i wprowadza trafne poprawki,
- dokumentacja jest kompletna, estetyczna, logicznie uporządkowana oraz może zostać wykorzystana podczas dalszego programowania robota.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, np.:

1. Jaką rolę pełni dodatkowy silnik w zadaniu przenoszenia ładunków na różne wysokości?
 - a. umożliwia podnoszenie i opuszczanie ładunku
 - b. służy do ładowania baterii
 - c. odpowiada za wyświetlanie komunikatów

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- d. zwiększa prędkość jazdy robota
2. Jakie informacje powinny znaleźć się w planie działania robota zapisanym w dokumencie tekstowym?
- tylko nazwa robota
 - lista kolorów wykorzystanych w programie
 - kolejne etapy działania robota oraz sposób sterowania silnikami i manipulatorem**
 - wyłącznie długość trasy przejazdu
3. Dlaczego podczas pracy wykorzystuje się narzędzia typu TechOffice?
- aby zwiększyć prędkość działania robota
 - aby tworzyć i porządkować dokumentację zadania**
 - aby sterować silnikami robota
 - aby łączyć kontroler Spike Hub
4. Jakie działania powinien wykonywać robot podczas realizacji zadania transportowego?
- tylko jazdę do przodu
 - wyłącznie obracanie manipulatorem
 - jazdę, podnoszenie, opuszczanie i przenoszenie ładunku**
 - jedynie zatrzymywanie się przy przeszkodach

Faza V. Dodatkowe, praca domowa (dla chętnych).

Zadanie dodatkowe (dla chętnych).

Zadanie: Robot transportowy z podnośnikiem – plan działania robota

Uczeń przygotowuje dokument tekstowy (np. w TechOffice), w którym opracuje plan działania robota odpowiedzialnego za przenoszenie ładunków na różne wysokości.

Robot ma za zadanie:

- przejechać do punktu odbioru ładunku,
- pobrać element za pomocą manipulatora,
- podnieść ładunek z wykorzystaniem dodatkowego silnika,
- przetransportować element do miejsca docelowego,
- opuścić ładunek i zakończyć zadanie lub wrócić na pozycję startową.

W dokumencie przygotuj:

- opis problemu i celu działania robota,
- listę kolejnych etapów pracy,
- plan przejazdu robota,
- pseudokod lub schemat działania,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- opis wykorzystania manipulatora i dodatkowego silnika.

Wymagania:

- zachowaj logiczną kolejność działań,
- zastosuj numerację lub punktory,
- przygotuj czytelny dokument w TechOffice,
- opracuj plan możliwy do wykorzystania podczas programowania robota w LEGO SPIKE Prime.do późniejszego wykorzystania podczas programowania robota w LEGO SPIKE Prime.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1: Konspekt nauczyciela lekcji nr 20 - *Kontrola dodatkowych silników. Przenoszenie ładunków na różne wysokości. Analiza problemu*

Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
- Przedstawienie uczniom celu i zakresu tematycznego lekcji
 - Nauczyciel omawia główne elementy zadania:
 - jazdę robota oraz transport ładunku,
 - sterowanie dodatkowym silnikiem odpowiedzialnym za podnoszenie i opuszczanie ładunku,
 - wykorzystanie manipulatora podczas przenoszenia elementów,
 - sterowanie silnikami z wykorzystaniem biblioteki motor_pair,
 - możliwość wykorzystania parametrów velocity oraz deceleration.
 - Uczniowie analizują zadanie i przygotowują plan działania w dokumencie tekstowym (np. TechOffice):
 - Podział uczniów na zespoły, oraz przydzielenie ról w grupie:
 - Uruchomienie na e-platformie RaP STEAM materiałów do lekcji.
 - Praca uczniów nad dokumentem:
 - zapis problemu i celu działania robota,
 - opracowanie kolejnych etapów przejazdu robota,
 - przygotowanie planu transportu ładunku,
 - opracowanie pseudokodu lub schematu działania robota,
 - analiza wykorzystania chwytaka oraz manipulatora.
 - Praca praktyczna uczniów:
 - testowanie różnych sposobów przejazdu robota,
 - analiza skuteczności manewru cofania i skrętu,
 - dobieranie parametrów ruchu robota,
 - proponowanie własnych rozwiązań usprawniających transport ładunku.
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów (**analiza poprawności planów działania robota, sposobów sterowania dodatkowymi silnikami oraz wykorzystania manipulatora podczas przenoszenia ładunków**).
- Zadanie pracy domowej (opcjonalnie) – opracowanie dodatkowego **wariantu transportu ładunku na inną wysokość lub przygotowanie własnego planu działania robota z wykorzystaniem dodatkowego silnika i manipulatora**.

Załącznik nr 2: Realizacja zadania głównego podczas lekcji nr 20 - *Kontrola dodatkowych silników. Przenoszenie ładunków na różne wysokości. Analiza problemu*

Cel i wprowadzenie do lekcji:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Uczniowie, pracując w zespołach, realizują zadanie polegające na **analizie sytuacji i zaplanowaniu działania robota eksploratora odpowiedzialnego za transport oraz przenoszenie ładunków na różne wysokości**. Robot ma bezpiecznie poruszać się po wyznaczonej trasie, pobrać element za pomocą chwytaka, **podnieść go z wykorzystaniem dodatkowego silnika** oraz dostarczyć do wyznaczonego miejsca. Szczególny nacisk położony jest na przygotowanie planu w dokumencie tekstowym (np. TechOffice), który stanowi podstawę do dalszego programowania.

Uczniowie mają za zadanie opracować plan działania robota, który uwzględnia analizę otoczenia oraz podejmowanie decyzji związanych z **transportem i podnoszeniem ładunku**. W tym celu uczniowie:

- analizują sytuację zadania i dostępne informacje,
- określają cel działania robota (przejazd wyznaczoną trasą, pobranie, podniesienie i transport ładunku z wykorzystaniem manipulatora oraz dodatkowego silnika) oraz możliwe trudności,
- planują kolejność działań robota,
- opracowują algorytm w formie uporządkowanej listy kroków,
- zapisują rozwiązanie w dokumencie tekstowym (lista czynności, pseudokod), dbając o jego czytelność,
- przygotowują plan, który będzie podstawą do późniejszego programowania robota eksploratora,
- (na kolejnej lekcji) wykorzystają plan do realizacji zadania i weryfikacji poprawności działania robota podczas transportu oraz podnoszenia ładunku.

Uczniowie na lekcji skupiają się na zgodności między analizą zadania a zapisem planu w dokumencie tekstowym, który będzie wykorzystany w dalszych etapach pracy.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 1. Przygotowanie do realizacji zadania głównego

Opis zadania dla uczniów:

Celem zadania jest zaprogramowanie robota, który, wykorzystując chwytak oraz dodatkowy silnik manipulatora, przetransportuje ładunek (paletę lub beczki) z punktu A do punktu B. Robot znajduje się w strefie załadunku i musi podnieść ładunek, przemieścić się po wyznaczonej trasie, wykonać skręt w miejscu, bezpiecznie dostarczyć ładunek do punktu docelowego oraz odłożyć go we wskazanym miejscu.



Uczniowie, pracując w zespołach, analizują trasę przejazdu robota oraz rozmieszczenie elementów zadania (punkt startowy, miejsce pobrania ładunku, przeszkody oraz punkt docelowy), a następnie przygotowują dokument tekstowy opisujący cały proces planowania działania robota eksploratora realizującego autonomiczne zadanie transportowe.

Na podstawie analizy sytuacji uczniowie ustalają kolejność wykonywania działań, określają momenty wykorzystania sterowania dwoma silnikami (**motor_pair**) oraz pojedynczego silnika (**motor**), planują sposób podnoszenia i opuszczania ładunku, a także dobierają parametry ruchu robota (czas, stopnie obrotu, prędkość).

Na tej podstawie uczniowie tworzą dokument, w którym:

- opisują analizę zadania i dostępnych informacji,
- uzasadniają wybór trasy przejazdu robota,
- określają kolejność realizacji działań robota i manipulatora,
- wskazują miejsca zatrzymania, skrętu, podniesienia oraz odłożenia ładunku,
- przedstawiają plan działania w formie uporządkowanej listy kroków lub pseudokodu,
- opisują możliwe trudności (np. konieczność testowania wartości ruchu, ryzyko zablokowania silnika lub różne pozycje początkowe manipulatora),

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- wyjaśniają sposób synchronizacji ruchu robota z pracą manipulatora.

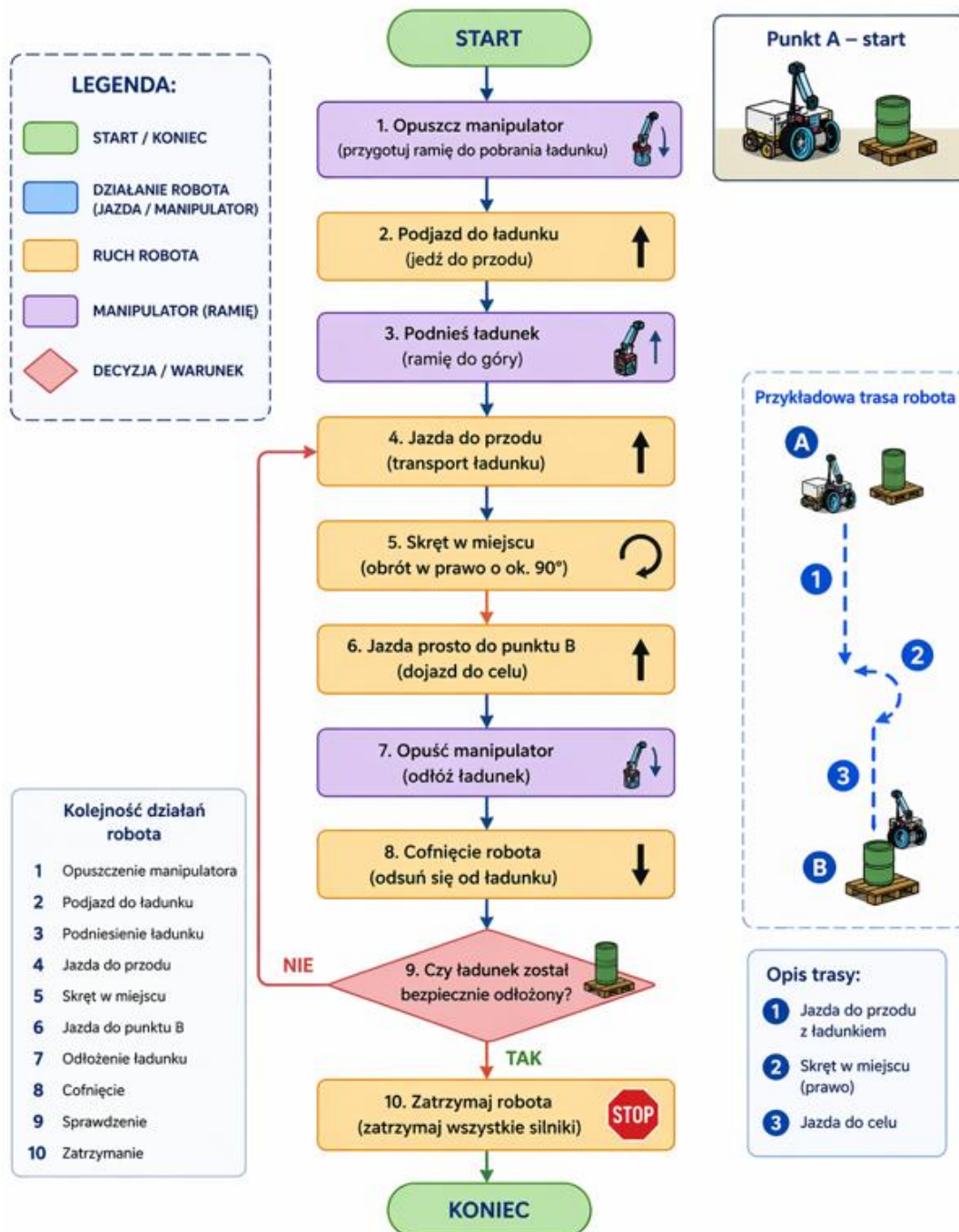
Dokument powinien być czytelny, logicznie uporządkowany i stanowić podstawę do późniejszego zaprogramowania robota eksploratora.

**Zadanie 2. Analiza problemu i tworzenie planu kodu źródłowego dla robota
przewożącego ładunki**

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

ALGORYTM – TRANSPORT ŁADUNKU ROBOTEM LEGO SPIKE

Robot przenosi ładunek z punktu A do punktu B przy użyciu manipulatora.

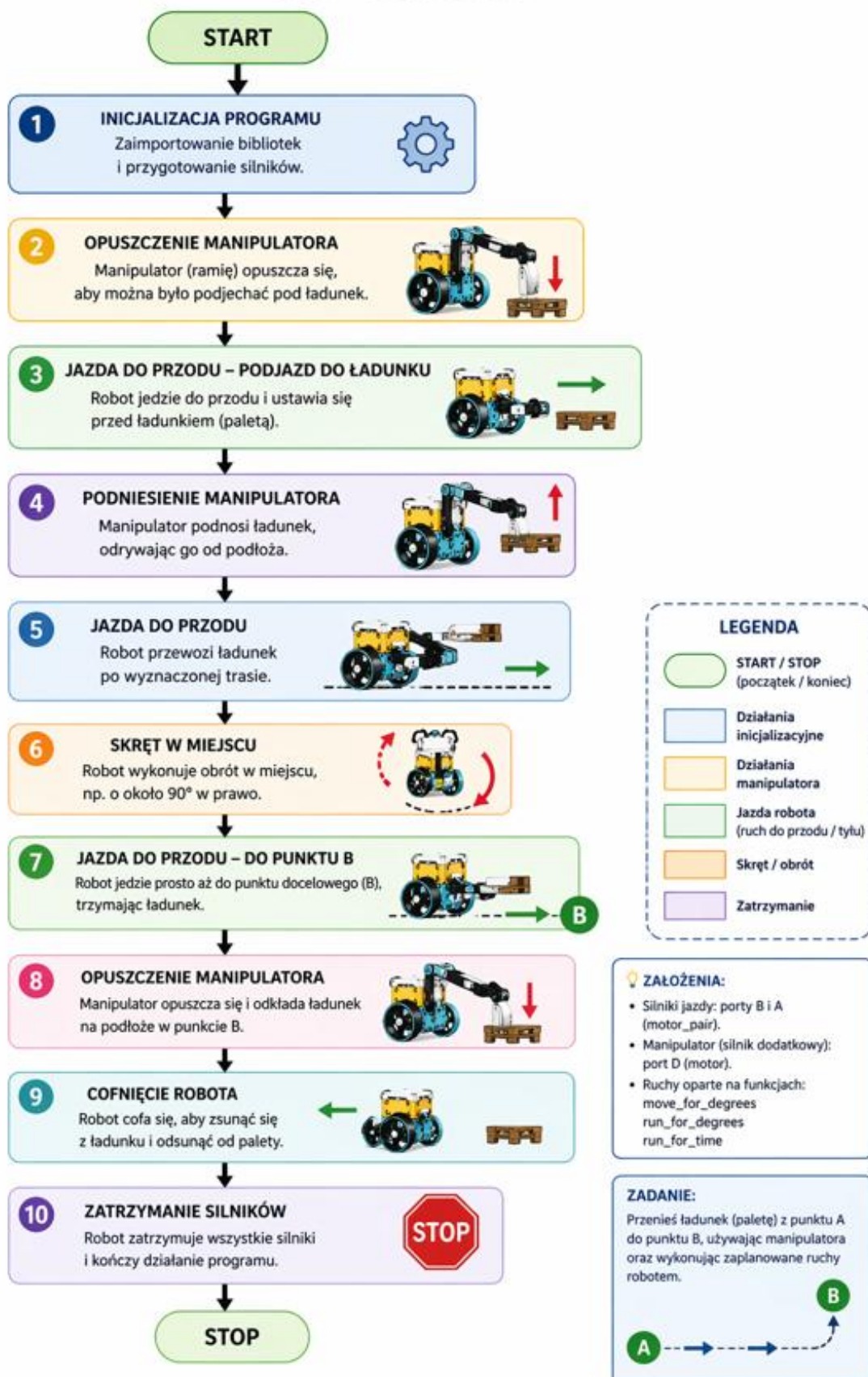


Pamiętaj: Dobierz odpowiednie parametry (stopnie, czas, prędkość), aby robot działał precyzyjnie i bezpiecznie.



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

SCHEMAT BLOKOWY – TRANSPORT ŁADUNKU LEGO® SPIKE PRIME



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Najważniejsze komendy:

- **motor_pair.move_for_degrees()** → jazda robota
- **motor.run_for_degrees()** → ruch ramienia
- **motor.run_for_time()** → bezpieczny ruch ramienia
- **motor_pair.stop()** → zatrzymanie

Wnioski:

Największą trudnością będzie dobranie odpowiednich parametrów ruchu, takich jak liczba stopni, czas działania silników i prędkość jazdy. Wyzwanie stanowi także precyzyjne sterowanie manipulatorem, aby poprawnie podnosił i odkładał ładunek. Trzeba będzie również dokładnie wykonać skręt oraz unikać blokowania programu podczas ruchu ramienia.

Jazda robota do przodu i do tyłu będzie przewidywalna oraz łatwa do zaprogramowania. Funkcja **motor_pair.move_for_degrees()** umożliwi dokładne sterowanie ruchem, a **run_for_time()** dobrze sprawdzi się przy opuszczaniu manipulatora. Dobrze zaplanowana kolejność działań pozwoli robotowi wykonać całe zadanie poprawnie.

Robota można ulepszyć poprzez dodanie czujników, np. odległości lub koloru, dzięki czemu działałaby bardziej automatycznie. Warto także zastosować warunki zamiast stałych wartości ruchu, ulepszyć konstrukcję manipulatora oraz rozbudować kod o dodatkowe funkcje. Testowanie różnych parametrów mogłoby zwiększyć szybkość i dokładność działania robota.

Podsumowanie analizy problemu i tworzenie planu kodu źródłowego dla kontroli dodatkowych silników oraz przenoszenia ładunków na różne wysokości:

Podczas tego etapu zajęć uczniowie zrealizowali zadanie polegające na zaprogramowaniu robota, który transportuje ładunek z punktu A do punktu B przy użyciu manipulatora. W trakcie pracy wykorzystali zarówno sterowanie dwoma silnikami odpowiedzialnymi za ruch robota, jak i dodatkowym silnikiem obsługującym ramię.

Uczniowie skupili się na planowaniu działania robota krok po kroku oraz na dobieraniu odpowiednich poleceń do konkretnych czynności, takich jak jazda do przodu, skręt w miejscu czy podnoszenie i opuszczanie ładunku. Poznali również różnicę między sterowaniem ruchem na określoną liczbę stopni a ruchem przez określony czas, co pozwoliło uniknąć problemów z blokowaniem programu. Ważnym elementem lekcji było testowanie i poprawianie działania robota. Uczniowie przekonali się, że poprawne działanie programu wymaga wielokrotnego sprawdzania i dostosowywania parametrów.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Lekcja pokazała, że roboty mogą wykonywać praktyczne zadania transportowe, ale ich skuteczność zależy od dobrze zaplanowanego algorytmu oraz precyzyjnego programowania.

Teraz możemy przejść do tworzenia dokumentacji projektowej. Dokument powinien być czytelny, logicznie uporządkowany i stanowić podstawę do późniejszego zaprogramowania robota eksploratora.

Zadanie 3. Praca z edytorem tekstu

[Etap 1] Uruchomienie edytora tekstu

1. Włącz komputer.
2. Otwórz program **LibreOffice Writer**.
3. Utwórz **nowy dokument tekstowy**.
4. Zapisz plik pod nazwą: „*Dokumentacja do projektu robota – lekcja 20*” w folderze wskazanym przez nauczyciela.
5. Używaj nagłówków (Nagłówek 1, 2), dodawaj obrazy i schematy. Zapisuj plik regularnie.

[Etap 2] Stwórz dokument, który będzie miał następującą strukturę:

1. Strona tytułowa

Uzupełnij tytuł: *Transport i podnoszenie ładunku robotem LEGO Spike*

Dodatkowo:

- imię i nazwisko
- klasa
- data

2. Wprowadzenie (opis problemu)

Opisz zadanie własnymi słowami:

Przykład:

Celem zadania jest zaprogramowanie robota LEGO Spike, który przewiezie ładunek (beczki) z punktu A do punktu B. Robot wykorzystuje chwytak, wykonuje ruchy do przodu, do tyłu, skręty podczas jazdy oraz podnosi i opuszcza ładunek.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

3. Założenia projektu

Wypisz wymagania:

- robot startuje w punkcie A
- musi pobrać ładunek
- wykonuje manewr cofania ze skrzętem
- transportuje ładunek do punktu B
- zatrzymuje się i odkłada ładunek

4. Opis działania robota (kroki)

Użyj do opisu działania robota listę numerowaną, np.

1. Opuszczenie manipulatora
2. Podjazd do ładunku
3. Podniesienie ładunku
4. Transport ładunku – jazda do przodu
5. Skręt w miejscu
6. Dalsza jazda do celu
7. Opuszczenie manipulatora
8. Cofnięcie robota
9. Zatrzymanie robota

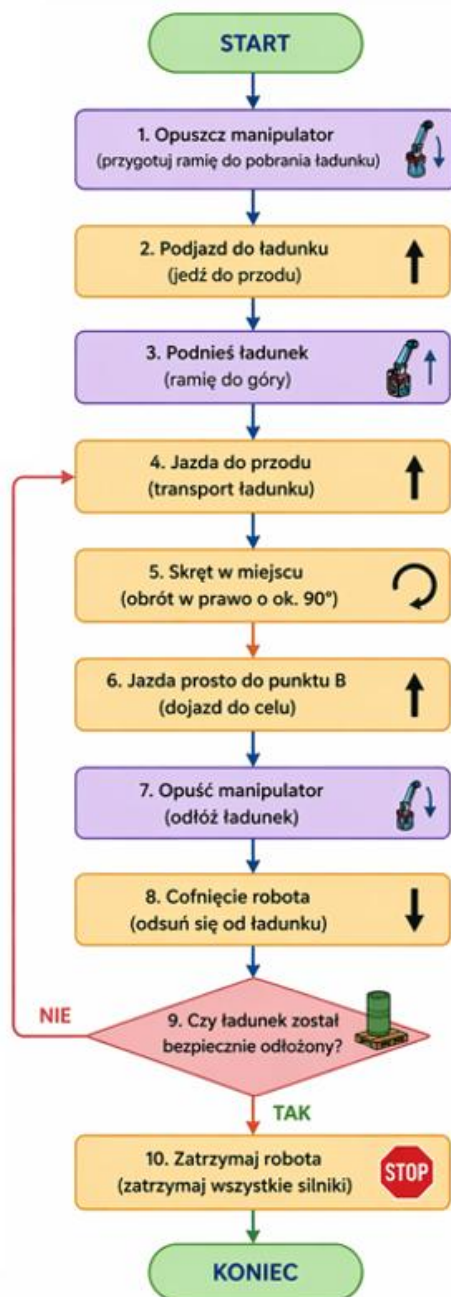
5. Schemat blokowy

Stwórz schemat blokowy programu. Wklej gotową grafikę (jeśli ją już masz) lub narysuj schemat w LibreOffice (**Wstaw** → **Kształty**). Wykorzystaj poniższe bloki do stworzenie schematu:



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przykładowy schemat jaki możesz utworzyć prezentuje się jak poniżej:



6. Opis programu (kod)

Wklej fragmenty kodu i je opisz:

Przykład:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Plan programu jest „szkieletem”, który zostanie zamieniony na pełny kod. Składa się z kilku etapów:

Krok 1: Przygotowanie programu

Na początku program:

- wczytuje biblioteki,
- przypisuje silniki do pary.

Polecenia:

```
from hub import light_matrix, port
import runloop
import motor_pair, motor

PAIR_1 = motor_pair.pair(port.B, port.A)
```

Krok 2: Start robota

Robot uruchamia program i może wyświetlić napis.

Polecenie: `await light_matrix.write("Hi!")`

Krok 3: Opuszczenie ramienia

Robot opuszcza manipulator, aby przygotować się do podniesienia ładunku.

Polecenie: `await motor.run_for_time(port.D, 300, -360)`

Ruch na czas – bezpieczny (nie zablokuje programu)

Krok 4: Podjazd do ładunku

Robot jedzie do przodu.

Polecenie: `await motor_pair.move_for_degrees(PAIR_1, 200, 0)`

Uwaga: 0 = jazda prosto

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Krok 5: Podniesienie ładunku

Manipulator podnosi paletę.

Polecenie: `await motor.run_for_degrees(port.D, 50, 360)`

Krok 6: Jazda do przodu

Robot zaczyna transport

Polecenie: `await motor_pair.move_for_degrees(PAIR_1, 450, 0)`

Krok 7: Skręt w miejscu

Robot obraca się (np. w prawo).

Polecenie: `await motor_pair.move_for_degrees(PAIR_1, 130, 100)`

Uwaga: 100 = maksymalny skręt

Krok 8: Dojazd do celu

Robot jedzie prosto do punktu B.

Polecenie: `await motor_pair.move_for_degrees(PAIR_1, 1337, 0)`

Krok 9: Odłożenie ładunku

Manipulator opuszcza paletę.

Polecenie: `motor.run_for_degrees(port.D, -50, 360)`

Uwaga: minus = ruch w dół

Krok 10: Cofnięcie robota

Robot cofa się, aby odsunąć się od ładunku.

Polecenie: `await motor_pair.move_for_degrees(PAIR_1, -360, 0)`

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Krok 11: Zatrzymanie

Robot kończy działanie.

Polecenie: `motor_pair.stop(PAIR_1)`

Krok 12: Uruchomienie programu

Na końcu program zostaje uruchomiony.

Polecenie: `runloop.run(main())`

7. Najważniejsze polecenia

- `motor_pair.move_for_degrees()` → jazda robota
- `motor.run_for_degrees()` → ruch ramienia
- `motor.run_for_time()` → bezpieczny ruch ramienia
- `motor_pair.stop()` → zatrzymanie

8. Wykorzystane parametry

Opisz wykorzystane parametry:

- `velocity` – prędkość jazdy
- `deceleration` – zwalnianie na końcu ruchu
- `stopnie` – odległość
-

9. Wnioski

Podsumuj swój projekt. Odpowiedz na pytania:

- Co było trudne?
- Co działało dobrze?
- Jak można ulepszyć robota?

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 5: Odkrywamy tajemnice Kodu... Odczytać i zrozumieć Matrix

Temat lekcji nr 21 - Kontrola dodatkowych silników - realizacja zadania.

Treści nauczania (wymagania szczegółowe - podstawa programowa) – I. 2) a); I. 2) b); I. 4); II. 1); II. 2); IV. 1)

Cel lekcji:

- **Wprowadzenie do programowania dodatkowych silników w języku Python** – uczniowie poznają podstawowe zasady korzystania z bibliotek i modułów odpowiedzialnych za sterowanie dodatkowymi silnikami oraz manipulatorem w LEGO SPIKE Prime.
- **Analiza działania programu** – uczniowie analizują fragmenty kodu odpowiedzialne za sterowanie ruchem robota, pracą dodatkowego silnika oraz manipulatora i określają, w jaki sposób poszczególne polecenia wpływają na działanie programu i robota.
- **Tworzenie algorytmów i programów wykorzystujących dodatkowe silniki** – uczniowie opracowują algorytmy oraz zapisują je w postaci programów tekstowych umożliwiających sterowanie ruchem robota, podnoszeniem i opuszczaniem ładunku oraz pracą manipulatora.
- **Testowanie działania programu** – uczniowie uruchamiają program, obserwują działanie robota podczas realizacji zadania transportowego oraz analizują poprawność pracy dodatkowych silników i manipulatora, a następnie wprowadzają poprawki w celu uzyskania poprawnego działania programu.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- **Uczeń dostrzega, że działanie programu i robota zależy od poprawności zapisanych poleceń:**
 - rozumie, że robot wykonuje instrukcje zgodnie z zapisanym kodem programu oraz poprawnym sterowaniem dodatkowymi silnikami i manipulatorem,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- obserwuje działanie programu napisanego w języku Python i analizuje wpływ poleceń na ruch dodatkowego silnika oraz transport ładunku.
- **Uczeń analizuje działanie programu i jego efekty:**
 - sprawdza poprawność działania programu sterującego dodatkowymi silnikami robota w języku Python,
 - identyfikuje błędy składniowe i logiczne oraz modyfikuje program poprzez poprawę zapisu poleceń lub zmianę parametrów ruchu silników i manipulatora,
 - wykorzystuje biblioteki i moduły służące do obsługi silników LEGO SPIKE Prime (motor_pair, motor).
- **Uczeń rozwija umiejętność testowania i doskonalenia rozwiązań:**
 - wykonuje próby działania programu i analizuje sposób działania robota oraz dodatkowych silników podczas realizacji zadania,
 - wprowadza poprawki w celu uzyskania poprawnego działania programu i właściwej synchronizacji ruchu robota z pracą manipulatora,
 - testuje działanie różnych parametrów ruchu i porównuje uzyskiwane efekty działania robota i dodatkowego silnika.

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści kształcenia będą realizowane za pomocą robotyki edukacyjnej LEGO® Education SPIKE™ Prime.

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna RaP STEAM,
- praca na komputerze,
- praca przy zestawie robotyki edukacyjnej,
- pogadanka.

Środki dydaktyczne

- zestaw PC z dostępem do Internetu,
- zestaw robotyki edukacyjnej LEGO® Education SPIKE™ Prime oraz zestaw rozszerzony LEGO® Education SPIKE™ Prime Expansion Kit,
- rzutnik multimedialny,
- e-podręcznik.

Formy pracy

- Praca indywidualna lub grupowa.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przebieg lekcji:

Faza I. Wprowadzenie

Na początku lekcji nauczyciel wprowadza uczniów w zagadnienia związane z **programowaniem dodatkowych silników w języku Python**, wyjaśniając, że robot może wykonywać złożone zadania transportowe dzięki odpowiedniemu sterowaniu ruchem robota oraz manipulatora. Następnie odnosi się do zadania lekcyjnego, polegającego na sterowaniu dodatkowymi silnikami LEGO SPIKE Prime podczas realizacji zadania transportowego.

Nauczyciel przedstawia cel i zakres zadania. Prezentuje uczniom środowisko programowania w języku Python w aplikacji LEGO SPIKE App oraz omawia **sposób korzystania z bibliotek i modułów odpowiedzialnych za sterowanie silnikami i manipulatorem**. Następnie przekazuje uczniom niezbędne informacje dotyczące wykorzystania **bibliotek motor_pair** oraz **motor do sterowania ruchem robota i pracą dodatkowego silnika**.

Uczniowie zapoznają się ze sposobami zapisu poleceń służących do sterowania ruchem robota, manipulatorem oraz dodatkowymi silnikami i obserwują, jak program po uruchomieniu realizuje kolejne etapy zadania transportowego.

Następnie uczniowie zdobywają umiejętności obejmujące:

- określenie problemu (w jaki sposób sterować robotem i dodatkowymi silnikami w języku Python),
- analizę działania programu sterującego robotem i manipulatorem,
- wykorzystanie bibliotek i modułów służących do sterowania silnikami,
- obserwację wyników działania programu po jego uruchomieniu,
- testowanie programu i analizę poprawności działania robota oraz dodatkowego silnika,
- wprowadzanie prostych poprawek związanych z działaniem programu i synchronizacją ruchów robota z pracą manipulatora.

Na zakończenie tej fazy lekcji nauczyciel podaje temat do zapisania w zeszycie i przechodzi do fazy II – realizacji praktycznej.

Faza II. Część zasadnicza - realizacja

W tej fazie lekcji nauczyciel zapewnia uczniom praktyczne ćwiczenia, które pozwolą eksperymentować z koncepcjami programowania. W szczególności obejmować to będzie używanie blokowych języków programowania oraz zestawu do robotyki.

1. Nauczyciel objaśnia sposób pracy na lekcji – we wskazanym przez niego momencie uczniowie będą wykonywać kolejne etapy prowadzące do rozwiązania zadania. Zdolniejsi uczniowie spróbują wykonać je samodzielnie. Słabsi wykonają kolejne kroki razem z nauczycielem, w oparciu o przygotowaną do zajęć instrukcję dostępną w e-

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- podręczniku lub materiał multimedialny. W razie potrzeby będą mogli dopytać nauczyciela. Nauczyciel na bieżąco sprawdza postępy uczniów.
2. Bazując na posiadanych zasobach, zapewnij każdemu uczniowi lub zespołowi uczniów kompletny zestaw do robotyki (tj. LEGO® Education SPIKE™).
 3. Zestawy te pozwalają uczniom budować własne roboty i programować je do wykonywania określonych zadań. Nauczyciel zaczyna od prostych projektów, takich jak zbudowanie i zaprogramowanie robota, który może wykonać proste czynności, które wprowadzą robota w ruch.
 4. Uczniowie budują bazowego robota według załączonej instrukcji (załącznik **Budowa_bazy_robota.pdf** do lekcji).
 5. Uczniowie włączają program **SPIKE**, poprzez który będą programować robota.
 6. Nauczyciel, wykorzystując e-platformę, demonstruje na tablicy multimedialnej działanie programu i robota, uruchamiając odpowiedni moduł i materiał multimedialny.
 7. Prezentowany materiał multimedialny, zrealizowany w oparciu o szczegółowy scenariusz (załącznik do lekcji), oprócz waloru teoretycznego stanowi również podstawę przebiegu części praktycznej lekcji. W szczególności, uczniowie:
 - a. budują prosty model bazowego robota w oparciu o zestaw do robotyki (tj. LEGO® Education SPIKE™),
 - b. tworzą listę czynności niezbędnych do wprowadzenia robota w ruch,
 - c. na podstawie listy czynności przygotowują program, zapisują go we wskazanym folderze (nauczyciel wskazuje właściwą lokalizację; może to być dysk lokalny lub zasób w chmurze) i potrafią go uruchomić;
 - d. wgrywają program do zbudowanego modelu robota, uruchamiają, obserwują i testują działanie robota. Wprowadzają modyfikacje w programie, poprawiają ewentualne błędy.

Faza III. Podsumowanie

Uczniowie analizują przedstawione przykłady programów zapisanych w języku Python oraz omawiają ich działanie. Nauczyciel ocenia stopień zrozumienia zależności między zapisanym kodem a działaniem robota, **dodatkowych silników oraz manipulatora** po uruchomieniu programu.

Kryteria oceny:

Ocena dopuszczająca:

- uczeń rozpoznaje podstawowe elementy programu w języku Python,
- potrafi wskazać polecenia odpowiedzialne za sterowanie ruchem robota oraz dodatkowym silnikiem.

Ocena dostateczna:

- uczeń analizuje przykładowy program i opisuje jego działanie,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- rozumie zależność między zapisanym kodem a działaniem programu,
- rozpoznaje podstawowe biblioteki i moduły wykorzystywane do sterowania silnikami (motor_pair, motor),
- wskazuje podstawowe błędy składniowe.

Ocena dobra:

- uczeń samodzielnie modyfikuje przykładowy program,
- poprawia błędy związane z zapisem poleceń, nawiasami lub parametrami ruchu,
- testuje działanie programu i analizuje sposób działania robota oraz manipulatora po wprowadzonych zmianach.

Ocena bardzo dobra:

- uczeń samodzielnie tworzy poprawny program w języku Python,
- wyjaśnia działanie poszczególnych elementów programu,
- wykorzystuje biblioteki i moduły do sterowania dodatkowymi silnikami LEGO SPIKE Prime,
- proponuje własne rozwiązania i potrafi je uzasadnić,
- samodzielnie analizuje i poprawia błędy w kodzie oraz optymalizuje działanie robota podczas realizacji zadania transportowego.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, np.:

1. Do czego służy biblioteka motor w programie robota?
 - a. Do sterowania dwoma silnikami
 - b. Do sterowania pojedynczym silnikiem**
 - c. Do wyświetlania tekstu
 - d. Do odczytu czujników
2. Do czego służy funkcja `run_for_time()`?
 - a. Zatrzymuje robota
 - b. Zmienia kierunek jazdy
 - c. Uruchamia silnik na określony czas**
 - d. Wyświetla tekst
3. Co może się stać przy użyciu `run_for_degrees()`, gdy silnik zostanie zablokowany?
 - a. Program może się zatrzymać i czekać**
 - b. Robot przyspieszy

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- c. Robot zmieni kierunek
- d. Nic się nie stanie

Faza V. Dodatkowe, praca domowa (dla chętnych).

Zadania dodatkowe (dla chętnych).

Zadanie 1: Robot magazynowy – podnoszenie i odkładanie ładunku

Utwórz program w języku Python, który pozwoli robotowi podnieść ładunek za pomocą dodatkowego silnika, przetransportować go w wyznaczone miejsce i bezpiecznie odłożyć.

Program powinien:

- zawierać poprawnie zaimportowane biblioteki i moduły (**motor_pair**, **motor**),
- umożliwiać jazdę robota do przodu i do tyłu,
- wykonywać co najmniej jeden skręt podczas przejazdu,
- sterować dodatkowym silnikiem odpowiedzialnym za podnoszenie i opuszczanie ładunku,
- poprawnie realizować kolejność działań robota i manipulatora,
- być zapisany poprawnie pod względem składni.

Opcjonalnie możesz rozbudować program tak, aby:

- robot wykonywał kilka transportów z rzędu,
- wykorzystywał różne prędkości ruchu i pracy silników,
- wykonywał dodatkowe manewry parkowania lub cofania,
- wyświetlał komunikaty na matrycy Spike Hub,
- zawierał własne usprawnienia zwiększające precyzję działania robota.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 21 - *Kontrola dodatkowych silników - realizacja zadania.*

Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
 - Nauczyciel omawia podstawowe zagadnienia związane z **programowaniem dodatkowych silników w języku Python**:
 - wyjaśnia, że robot może wykonywać złożone zadania transportowe dzięki sterowaniu silnikami i manipulatorem,
 - przedstawia przykłady bibliotek i modułów wykorzystywanych do sterowania silnikami LEGO SPIKE Prime (motor_pair, motor),
 - omawia znaczenie poprawnego sterowania ruchem robota oraz synchronizacji pracy dodatkowego silnika i manipulatora.
 - Wprowadzenie do zadania i przypomnienie podstaw pracy w LEGO SPIKE Prime:
 - omówienie środowiska programowania Python w aplikacji LEGO SPIKE App,
 - przedstawienie bibliotek i modułów odpowiedzialnych za sterowanie silnikami oraz manipulatorem,
 - przedstawienie zależności: zapis kodu >> uruchomienie programu >> ruch robota i działanie manipulatora,
 - analiza przykładowych programów wykorzystujących jazdę robota, skręty oraz podnoszenie i opuszczanie ładunku.
 - Praca uczniów:
 - analiza przedstawionych przykładów programów sterujących robotem i dodatkowymi silnikami,
 - omawianie działania poszczególnych linii kodu,
 - testowanie działania robota oraz manipulatora po uruchomieniu programu,
 - wyszukiwanie i poprawianie błędów w kodzie,
 - proponowanie własnych sposobów usprawnienia transportu i podnoszenia ładunku.
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów (**analiza poprawności działania programu, sterowania dodatkowymi silnikami oraz skuteczności realizacji zadania transportowego**).
- Zadanie pracy domowej (opcjonalnie) – **napisanie prostego programu sterującego dodatkowym silnikiem i manipulatorem podczas podnoszenia lub opuszczania ładunku**.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 2 – Realizacja zadania głównego podczas lekcji nr 21 - *Kontrola dodatkowych silników - realizacja zadania.*

Zadanie 1. Przygotowanie do realizacji zadania głównego

Uczniowie, pracując w zespołach lub indywidualnie, realizują zadanie polegające na poznaniu sposobu **programowania dodatkowych silników w języku Python oraz sterowaniu ruchem robota i manipulatora w środowisku LEGO SPIKE Prime**. Szczególny nacisk położony jest na poprawne wykorzystanie bibliotek i modułów (**motor_pair, motor**), analizę działania programu oraz interpretację efektów pracy robota podczas realizacji zadania transportowego.

Uczniowie opracowują i zapisują rozwiązanie problemu w postaci **programów tekstowych**, które umożliwią sterowanie ruchem robota, dodatkowymi silnikami oraz manipulatorem odpowiedzialnym za podnoszenie i transport ładunku. W tym celu:

- analizują przykładowe programy zapisane w języku Python,
- określają cel programu oraz przewidują możliwe błędy składniowe i problemy związane z pracą silników lub manipulatora,
- planują kolejność wykonywania instrukcji w programie,
- opracowują algorytm w uporządkowanej formie,
- zapisują go jako program w języku Python wykorzystujący biblioteki **motor_pair** i **motor**,
- testują działanie programu oraz analizują zachowanie robota podczas realizacji zadania transportowego,
- wprowadzają poprawki związane z działaniem programu, synchronizacją ruchów oraz parametrami pracy silników,
- doprecyzowują rozwiązanie tak, aby kod był czytelny i poprawny składniowo.

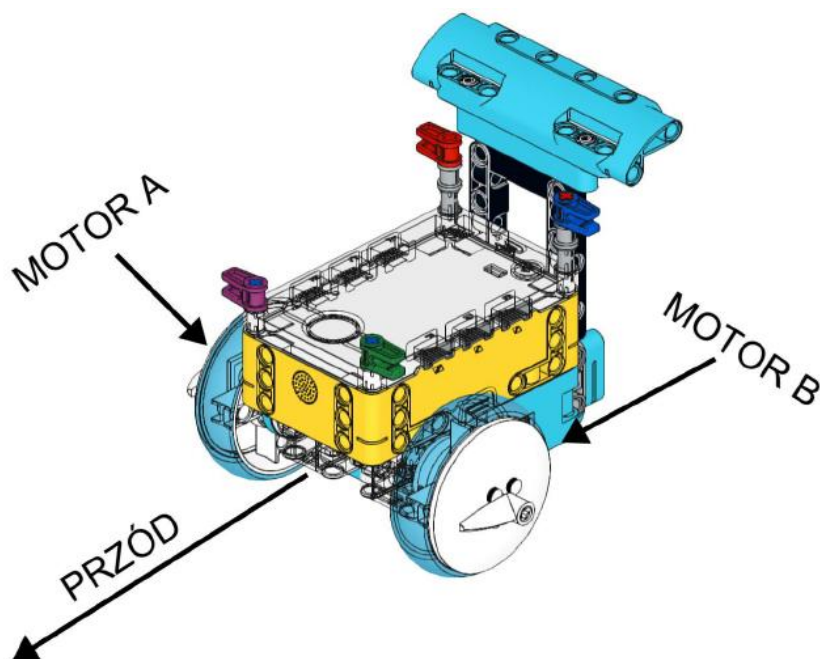
Uczniowie wykorzystują środowisko LEGO SPIKE Prime w trybie Python, zwracając uwagę na zgodność między zapisanym kodem a rzeczywistym działaniem robota i manipulatora. Szczególną uwagę poświęcają poprawności składni, **wykorzystaniu funkcji sterujących silnikami oraz analizie komunikatów o błędach** pojawiających się po uruchomieniu programu.

Jeśli jest taka potrzeba, wskazana przez nauczyciela, to uczniowie zbudują robota od podstaw zgodnie z poniższą instrukcją.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Budowa bazowego robota

Uruchom plik **Budowa bazy robota.pdf** znajdujący się na platformie i postępuj wg podanych w pliku kroków. Efektem końcowym będzie fizyczny robot gotowy do programowania. **Jeśli konstrukcja jest gotowa – idź dalej.**



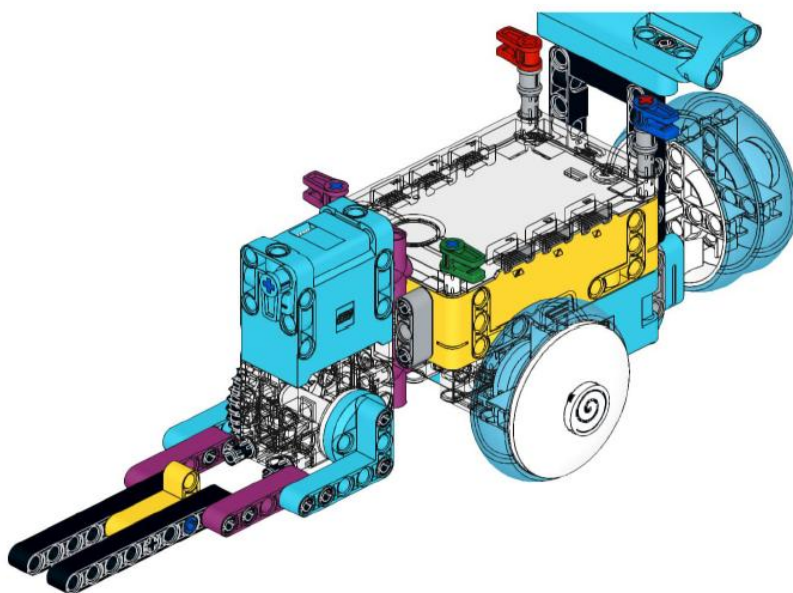
Pudełka niezbędne do realizacji lekcji: **1,2,3,4 + koła małe**



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

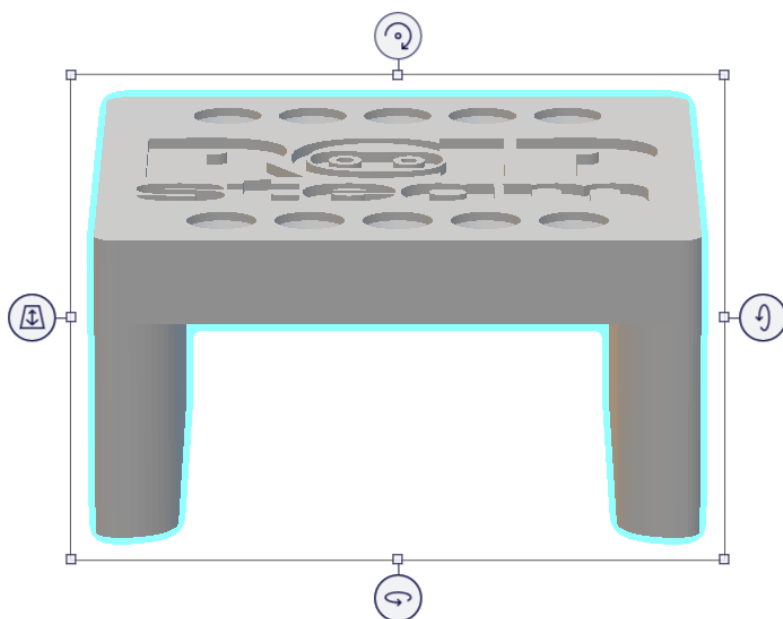
Dodatkowe instrukcje niezbędne do realizacji lekcji:

- **chwytakMaly.pdf**



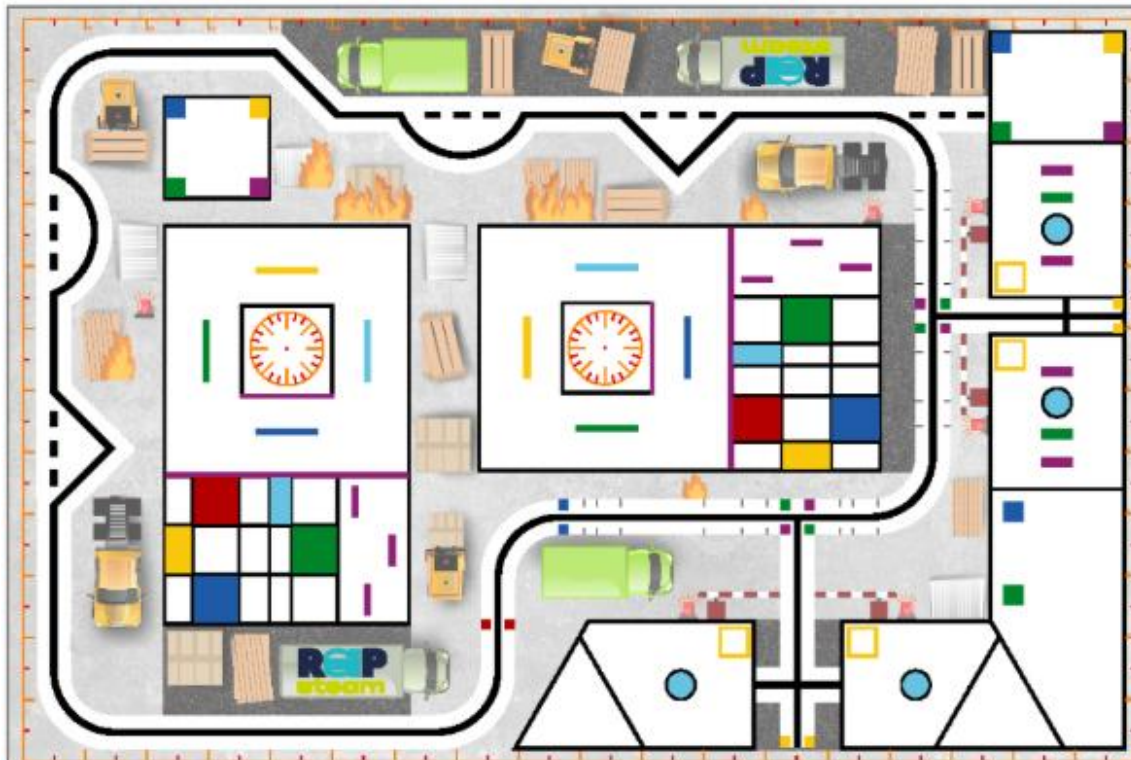
Dodatkowe elementy niezbędne do realizacji lekcji:

- Wydruk 3D – **Paleta.3mf**, sztuk 1



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Do realizacji scenariusza zajęć będziesz potrzebował również matę:

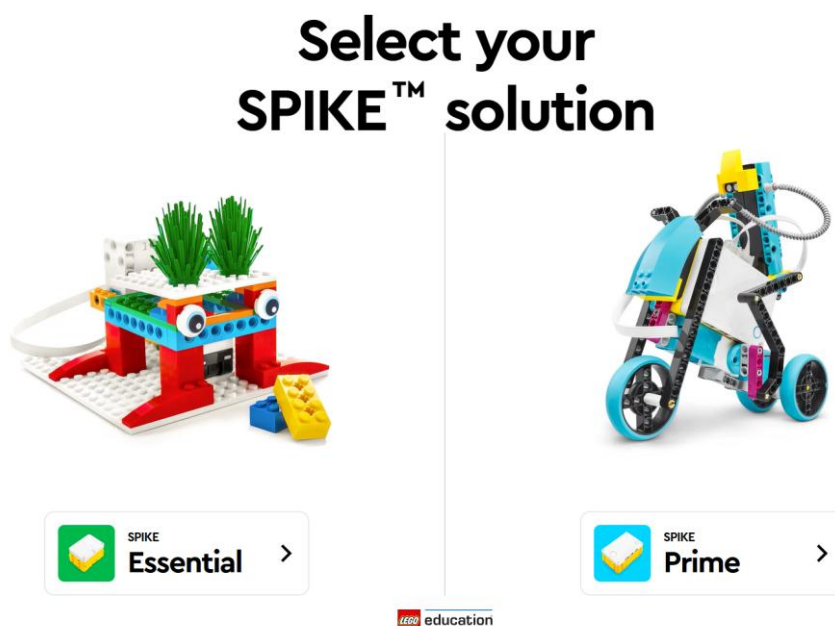


Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 2. Konfiguracja środowiska.

Uruchamianie aplikacji LEGO Spike

1. Włącz przeglądarkę internetową dostępną na twoim urządzeniu
2. Wpisz w przeglądarce adres: <https://spike.legoeducation.com/> Pojawi się okno wyboru wersji LEGO SPIKE. Wybierz Spike Prime

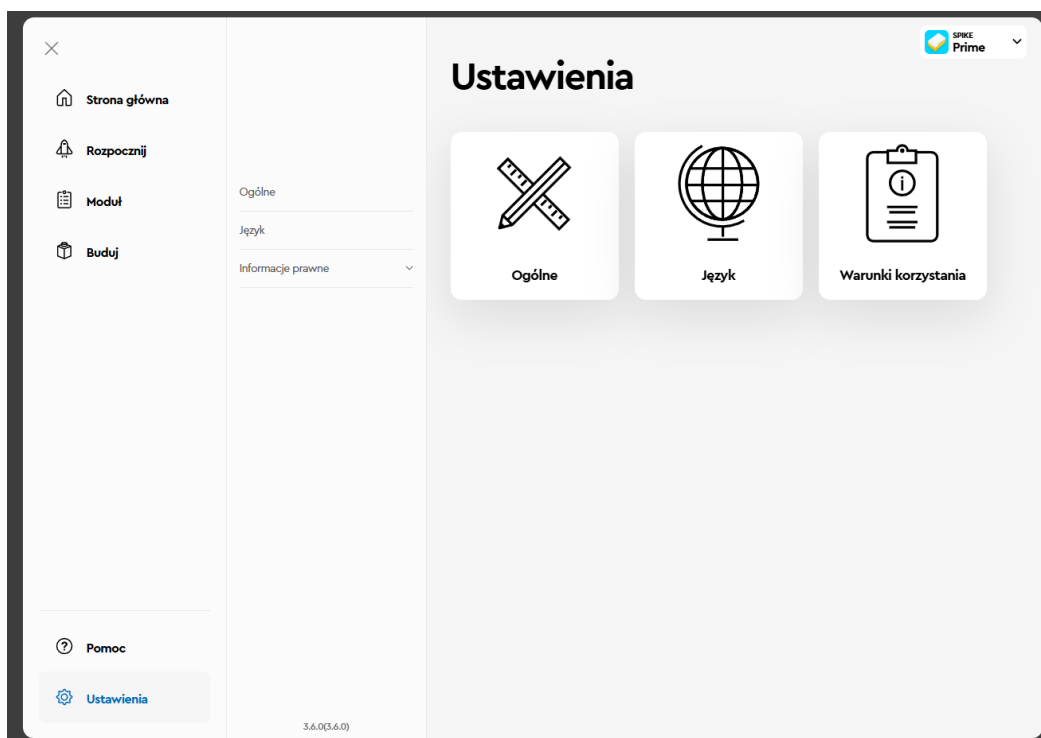


3. Po wybraniu opcji SPIKE Prime pojawia się menu główne. Jeśli występuje język polski, należy zmienić go na język angielski.

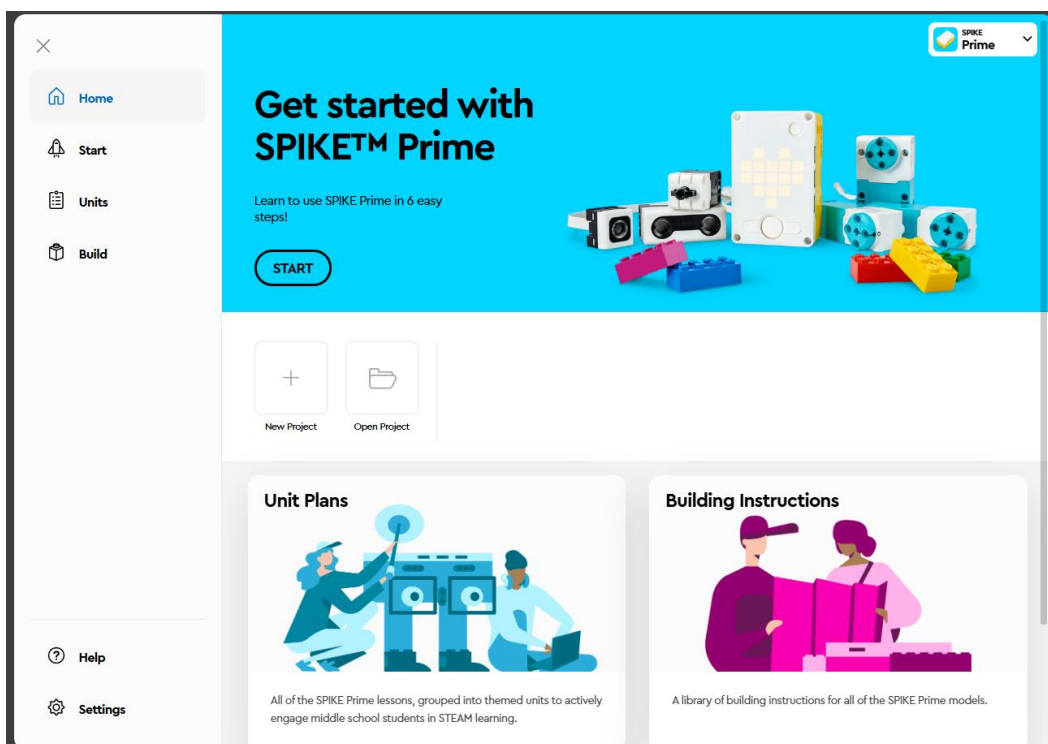
Jeżeli zaś wszystko jest w **języku angielskim**, to przejdź do kroku 6.

Aby zmienić język, w lewym dolnym rogu menu głównego kliknij przycisk **Ustawienia**.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

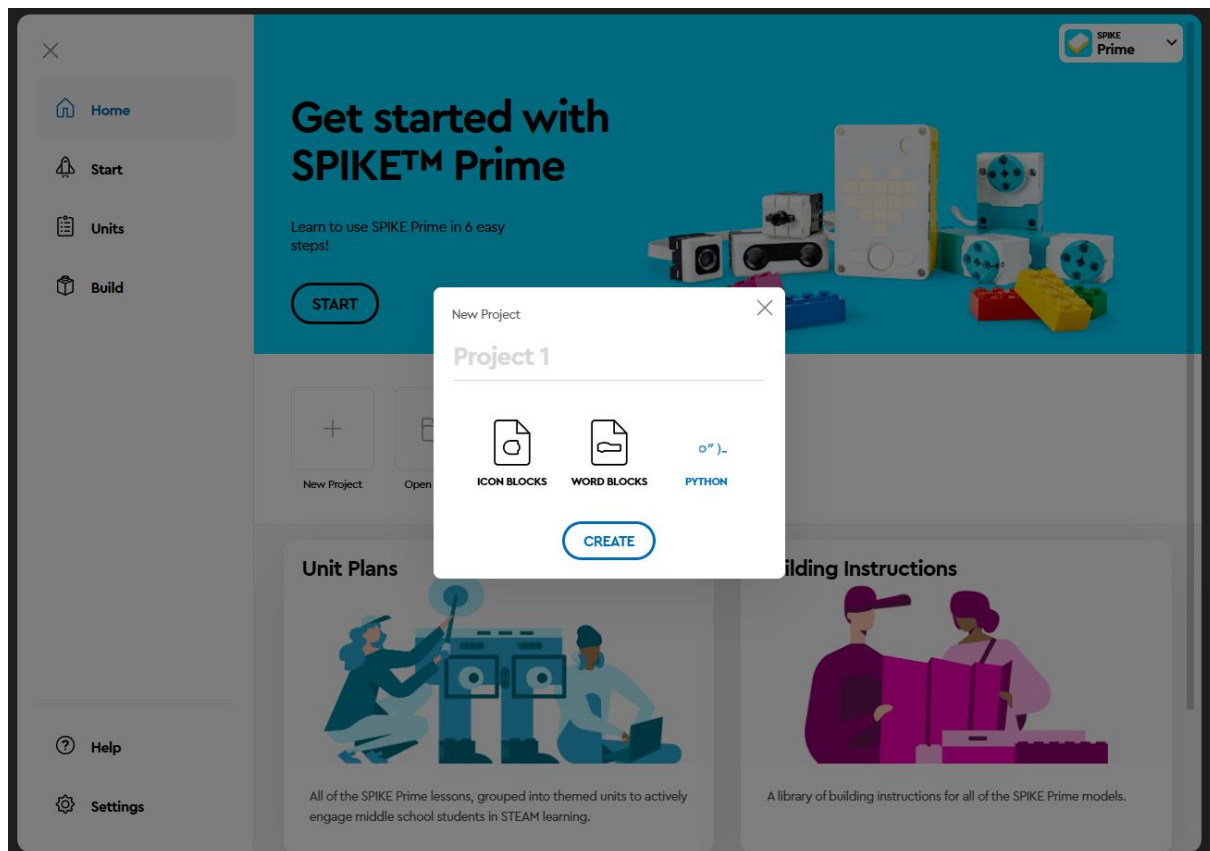


4. W nowym oknie kliknij **globus podpisany „Language”**.
5. **Wybierz język: United States (English)**. Strona przeładuje się i język angielski będzie dostępny.
6. Kliknij po lewej przycisk **Home**. Pojawi się okno jak poniżej

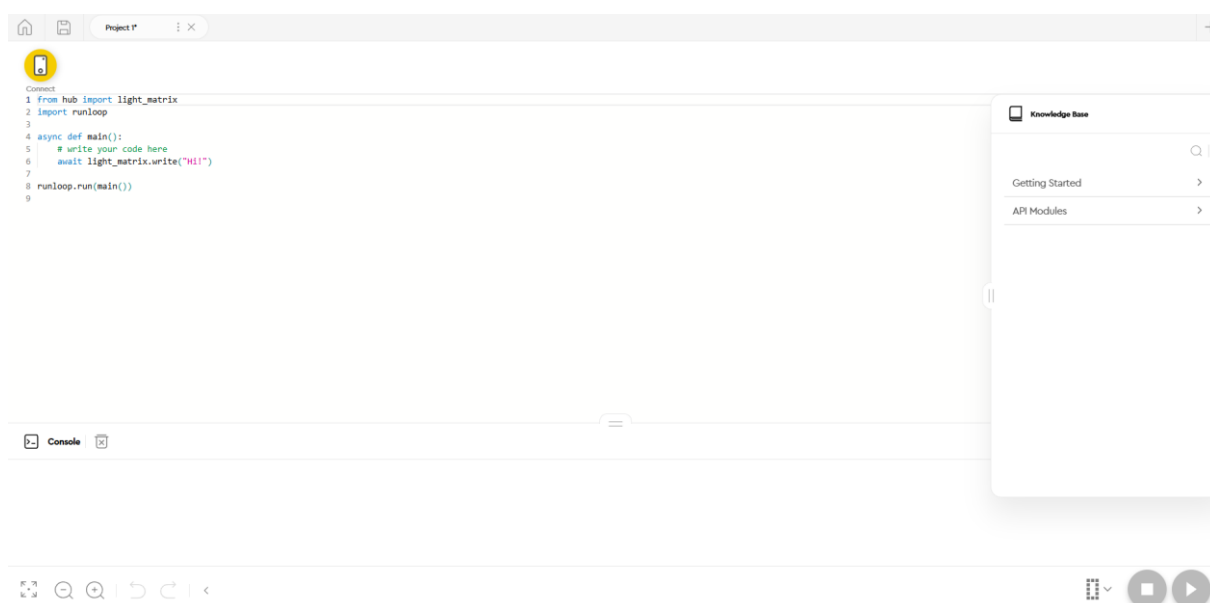


Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

7. Kliknij na środku przycisk **New Project**. Pojawi się nowe okno jak poniżej



8. Wybierz opcję **Python** oraz podaj nazwę projektu. Kiedy wszystko zostanie poprawnie wybrane, wciśnij przycisk **Create**.

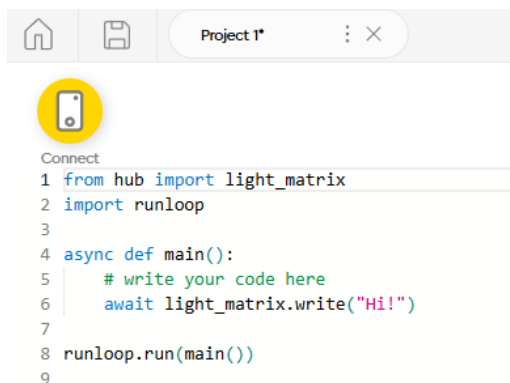


Wszystko jest gotowe do programowania. Pora na podłączenie kontrolera Spike Hub.

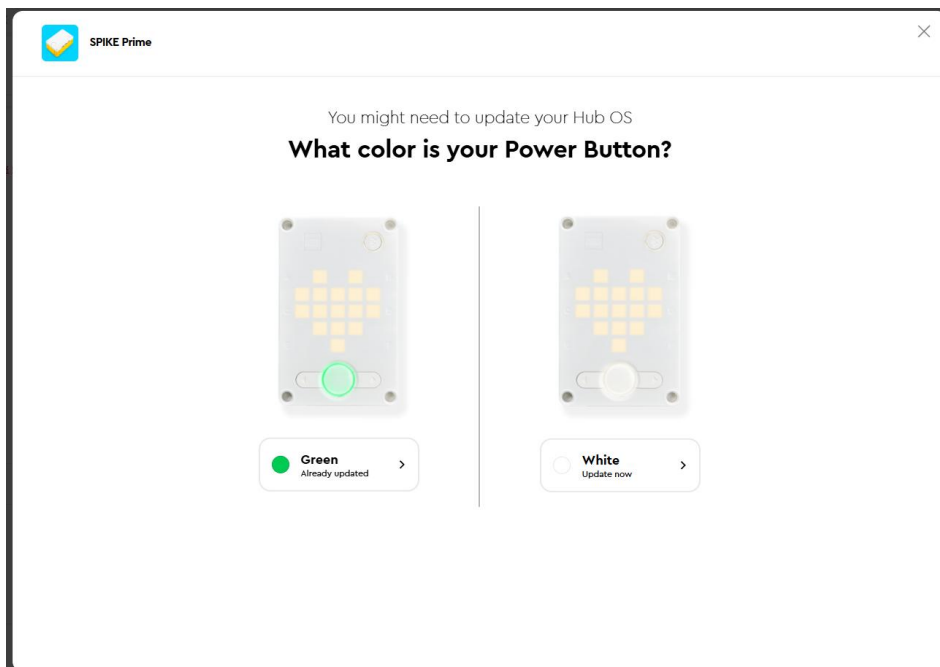
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 3. Podłączenie kontrolera Spike Hub

1. W głównym programie kliknij przycisk **Connect** znajdujący się w górnym rogu, tuż pod nazwą twojego projektu. Patrz poniżej.

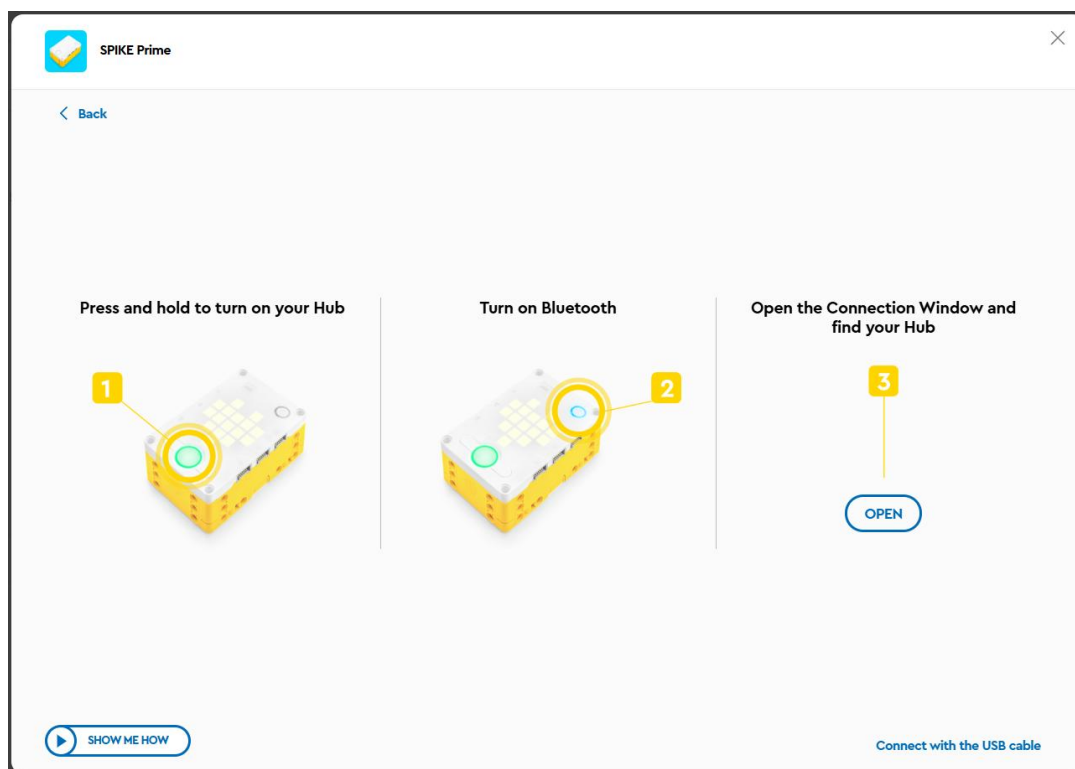


2. Pojawi się nowe okno. **Włącz kontroler Spike Hub** klikając **duży biały przycisk** i poczekaj aż się uruchomi. Następnie wybierz jedną z opcji odpowiadającą stanowi kontrolera Spike Hub – zawsze będzie to **Green – Already updated**.

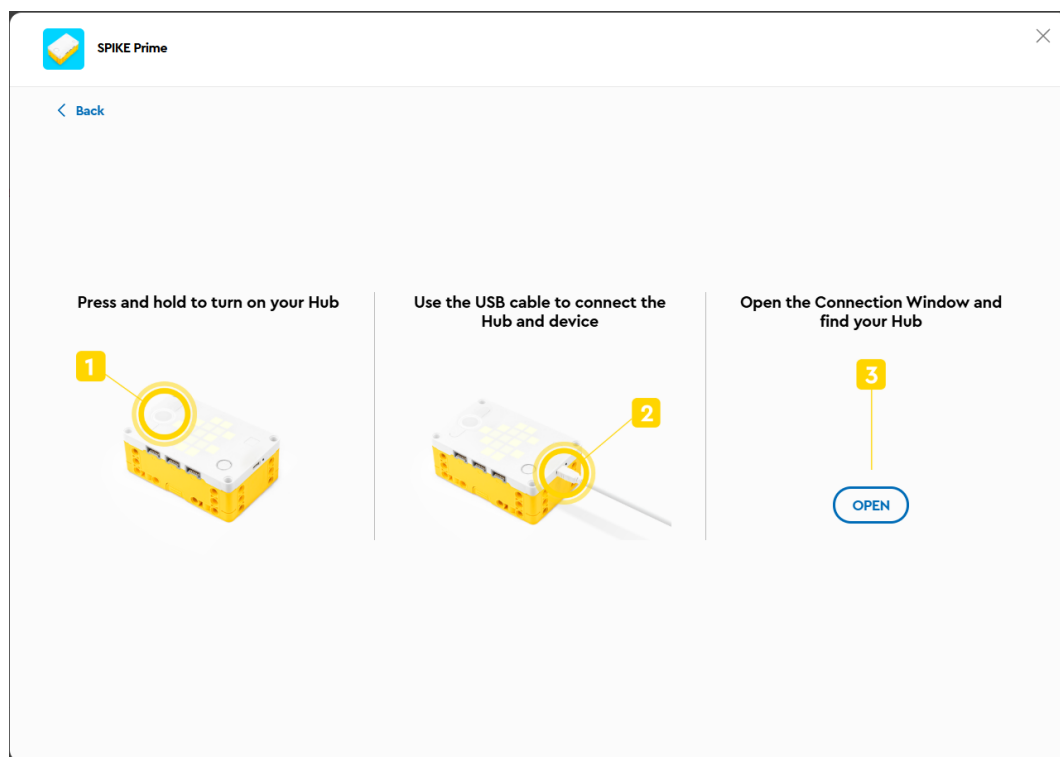


3. Pojawi się nowe okno (patrz poniżej). Wybierz opcję **Connect with USB cable** znajdującą się w prawym dolnym rogu okna.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

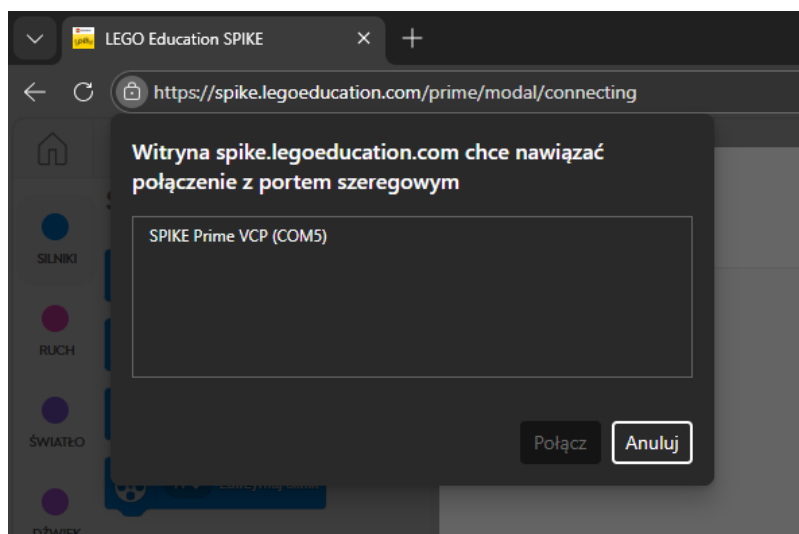


4. Postępuj zgodnie z instrukcją na ekranie. Po przeczytaniu kliknij przycisk **Open**.



5. Pojawi się nowe okno (patrz poniżej)

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



6. Wybierz opcję **SPIKE Prime VCP** i kliknij **Połącz**.

UWAGA – u Ciebie nazwa może nie być identyczna jak na rysunku powyżej. Wybierz tę opcję, która ma w nazwie **Spike Prime**. Innymi rzeczami się nie przejmuj.

Jeśli wszystko zostanie poprawnie połączone, to okna zostaną automatycznie zamknięte, a ikona kontrolera w programie zmieni się na zieloną.



```
1 from hub import light_matrix
2 import runloop
3
4 async def main():
5     # write your code here
6     await light_matrix.write("Hi!")
7
8 runloop.run(main())
9
```

Gratulacje. Twój Spike Hub jest podłączony do komputera. Możesz przystąpić do programowania robota.

W przypadku problemów postępuj wg kroków:

1. Sprawdź, czy kontroler Spike Hub jest włączony i świeci na zielono
 - a. Jeśli kontroler świeci na biało zgłoś to nauczycielowi
 - b. Jeśli kontroler świeci na zielono to podążaj dalej wg instrukcji łączenia powyżej
2. Sprawdź czy kabel USB jest podłączony (Spike Hub <-> Komputer)
3. Sprawdź, czy wybrano odpowiednią nazwę urządzenia – w oknie połączenia z portem szeregowym należy wybrać urządzenie, które w nazwie ma SPIKE Prime.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 4. Realizacja zadania głównego

W tej lekcji wykonasz zadanie, które **opracowano na poprzedniej lekcji**.

Należy zaprogramować robota tak aby wykorzystując dodatkowy manipulator (mniejszy) przewiózł ładunek z punktu A do punktu B wykonując specjalistyczne ruchy w tym:

- Jazda przód / tył
- Skręt w miejscu w wybranym kierunku
- Ruch manipulatorem góra / dół – programowanie dodatkowego silnika

Przykład realizacji zadania – wprowadzenie bibliotek

UWAGA – w materiałach stosowane są komentarze. Na zrzutach ekranu jest to tekst oznaczony jako zielony. Tego tekstu nie przepisuj! To jest tylko wytłumaczenie dla Ciebie, co dzieje się w danej części programu. Przepisuj tylko i wyłącznie tekst, który nie jest zielony. Korzystaj z tabeli. Pamiętaj, że wielkość liter MA ZNACZENIE!

Na początku programu należy dodać odpowiednie odwołania (biblioteki) do modułów kontrolera Spike Hub i czujników. Dodaj moduł, biblioteki:

- **port (moduł),**
- **light_matrix (moduł),**
- **motor_pair (biblioteka),**
- **motor (biblioteka)**

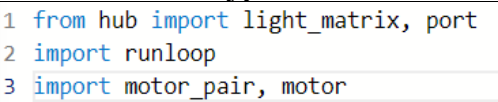
Aby móc programować silniki niezbędne jest zaimportowanie biblioteki **motor_pair**. Pozwala ona, tak jak w Scratchu za pomocą różowych bloków sterować dwoma silnikami jednocześnie.

W dzisiejszej lekcji poznasz metodę programowania dodatkowego silnika. Biblioteka **motor_pair** jest używana, gdy odbywa się sterowanie dwoma silnikami naraz. W tej lekcji trzeci silnik (silnik duży) jest pojedynczy. Wymaga to użycia biblioteki **motor**, która pozwala sterować głównie pojedynczym silnikiem. Odpowiednikiem biblioteki **motor** w Scratchu są niebieskie kafelki.

Podczas dzisiejszych zajęć wykorzystasz też moduł **port** oraz **light_matrix** (dostęp do portów oraz wyświetlanie informacji na matrycy Spike Hub).

Zaktualizuj początek programu i dodaj odpowiednie odwołania. Część programu odpowiedzialna za biblioteki powinna wyglądać następująco:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zdjęcie	Kod (tekst)
	<pre>from hub import light_matrix, port import runloop import motor_pair, motor</pre>

W tej chwili cały program powinien wyglądać następująco:

```
1 from hub import light_matrix
2 from hub import port
3 import motor_pair
4 import runloop
5
6 async def main():
7     # write your code here
8     await light_matrix.write("Hi!")
9
10 runloop.run(main())
```

Przygotowanie struktury programu

Dzisiejsze zajęcia zacznij od przygotowania podstawowych ustawień i parametrów dla kontrolera Spike Hub i silników. Dodaj komendę, która przypisze silniki podpięte do portów B i A do zmiennej **PAIR_1** – tak jak na poprzednich lekcjach. Następnie uruchom silniki tak aby robot:

- Pojechał do przodu 360 stopni, bez skręcania

Możesz zostawić komendę odpowiadającą za wyświetlanie napisu Hi! na kontrolerze Spike Hub. Jeśli chcesz ją usunąć to możesz to zrobić. Nie jest ona wymagana do dzisiejszych zajęć.

Program powinien wyglądać następująco:

```
1 from hub import light_matrix, port
2 import runloop
3 import motor_pair
4
5 async def main():
6     # write your code here
7     await light_matrix.write("Hi!")
8
9     #przypisanie portów A i B do PAIR_1
10    motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
11
12    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 360, 0)
13
14 runloop.run(main())
```

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Uruchom program, wgrywając go do kontrolera Spike Hub, i sprawdź, czy robot wykonuje poprawnie swoje zadanie. Jeśli wszystko działa poprawnie, przejdź do dalszej lekcji.

Realizacja głównego zadania

Zrealizuj zadanie, którego opracowanie, dokumentację i wyliczenia przygotowano na poprzednich zajęciach. Zgłoś się do nauczyciela po odpowiednie elementy. W dzisiejszej lekcji dodatkowym elementem (poza robotem i jego modułami) będzie:

- **Wydruk 3D – paleta, sztuk 1**

Upewnij się, że masz ją do dyspozycji. Następnie zaprogramuj robota wg ustaleń nauczyciela.

Przykład realizacji zadania

Przedstawiony przykład realizuje rozwiązanie głównego zadania. Robot wykonuje następujące zadania:

- Opuść silnik dodatkowy (manipulator) aby robot po podjechaniu do wydruku 3D (palety) mógł ją swobodnie podnieść,
- Przejdź robotem do przodu, aby nałożyć paletę na chwytak (manipulator),
- Podnieś manipulator, aby paleta nie miała kontaktu z podłożem,
- Podjedź do przodu, aby utrzymać się założonej trasy (w przypadku przykładu jazda następuje po czarnej linii bez jej wykrywania),
- Wykonaj obrót o ok. 90 stopni w prawo. Robot powinien obrócić się w miejscu,
- Jedź do przodu przez 4,5 obrotu (ok. 1337 stopni), bez skręcania. Robot dojeżdża do celu (punkt B),
- Opuść silnik dodatkowy (manipulator) aby postawić paletę na podłożu,
- Cofnij się o ok. 50 stopni, aby zsunąć paletę z manipulatora,
- Zatrzymaj robota.

Do realizacji zadania wykorzystano funkcję **move_for_degrees()** oraz **move_for_time()** podając im odpowiednie parametry.

Funkcja **move_for_time()** pozwala na uruchomienie silników przez określony czas. Będzie się on poruszać w zadanym kierunku z wybraną przez użytkownika prędkością. Gdy czas minie, silnik sam się zatrzyma – nie jest wymagane użycie funkcji **stop()**. Funkcja **move_for_time()** także wymaga użycia słowa **await**.

Funkcja **move_for_time()** zostanie wykorzystana do opuszczania manipulatora, aby dotknął podłoża. Dlaczego **move_for_time()** a nie **move_for_degree()**? Odpowiedź tkwi w zasadzie działania. W przypadku użycia funkcji **move_for_degree()** silnik MUSI obrócić się o zadaną liczbę stopni. Jeśli silnik zostanie zablokowany (przez jakiś element zewnętrzny), to cały program zostanie zablokowany. Program robota czeka cały czas, aż silnik wykona odpowiednią

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

liczbę stopni. Jeśli użyta zostanie funkcja `move_for_time()` to robot będzie obracać silnik **DO MOMENTU** aż minie czas. Jeśli silnik się zablokuje, to po zadany czasie program będzie kontynuowany. Dzięki temu możliwe jest operowanie robotami, gdy np. nieznana jest dokładna wartość obrotów silników. Takim przypadkiem może być ramię robota – manipulator. W zależności od jego ustawienia początkowego oraz wartości na silniku (wartość stopnia) nie jest możliwe ustalenie dokładnego obrotu ramienia tak, aby zawsze dotykał podłoża. Manipulator przy uruchomieniu programu nie zawsze może być w tej samej pozycji (szczególnie podczas kolejnych uruchomień). Aby uniknąć sytuacji, w której zawsze ustala się dokładnie położenie ramienia samodzielnie, można zastosować funkcję ruchu w czasie. Wygląda ona następująco:

```
await motor.run_for_time(port.D, 300, -360)
```

Funkcja `run_for_time()` wymaga podania:

- Portu, na którym podpięty jest silnik niezbędny doysterowania,
- Czas w milisekundach jak długo silnik ma się obracać,
- Prędkość i kierunek – wartość w stopniach.

W przykładzie uruchomiono silnik podpięty do portu D na 300 milisekund (0,3 sekundy) z prędkością 360 stopni na sekundę w przeciwnym kierunku.

Aby poruszać się do przodu zastosowano następujący kod:

```
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 450, 0)
```

Robot wykonuje ruch do przodu (silniki B i A – **PAIR_1**) przez 450 stopni jadąc prosto (0).

Aby podnieść manipulator należy wykonać funkcję `run_for_degrees()`. Jest podobna do funkcji `move_for_degrees()`. Jediną różnicą jest to że jako pierwszy parametr (wartość w nawiasie) przyjmuje pojedynczy port a nie zestaw portów. Zwróć uwagę, do jakiej biblioteki się odwołujesz.

- W przypadku obsługi silników do jazdy stosujesz bibliotekę **motor_pair** i funkcję `move_for_degrees()`
- W przypadku obsługi dodatkowego silnika stosujesz bibliotekę **motor** i funkcję `run_for_degrees()`

ZWRACAJ NA TO CAŁY CZAS UWAGĘ!

Zastosowano następujący kod dla manipulatora:

```
await motor.run_for_degrees(port.D, 50, 360)
```

Komenda obraca silnik z prędkością 360 stopni na sekundę o 50 stopni. Wartość została ustalona poprzez próby. W Twoim programie możesz podać inną; uwzględnij jednak, czy nie

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

zablokujesz ramienia, jeśli podasz większą, lub czy podniesiesz paletę z podłoża, jeśli podasz mniejszą.

Kolejny kod nakazuje robotowi przejechać dystans 200 stopni bez skręcania (0).

```
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 200, 0)
```

W kolejnym kroku robot wykonuje obrót w miejscu o ok. 90 stopni w prawo (130 stopni oraz skrętem o mocy 100). Prędkość pozostaje domyślna, dlatego nie jest uwzględniona.

```
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 130, 100)
```

Następnie robot jedzie do przodu przez ok. 4,5 obrotów – 1337 stopni ze standardową prędkością, cały czas pokonując drogę prosto. Robot dojeżdża do punktu docelowego po wykonaniu ok. 4,5 obrotów.

```
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1337, 0)
```

Następnym zadaniem jest opuszczenie manipulatora, aby postawić paletę na podłożu.

Wykorzystana zostanie do tego komenda **run_for_degrees()** przyjmująca przeciwną wartość do podnoszenia. Podnoszenie przyjęło parametry 50 (stopnie) i 360 (prędkość). Opuszczenie zaś przyjmuje wartości -50 (stopnie) i 360 (prędkość). Oznacza to, że program nie zostanie zablokowany, ponieważ wróci do pozycji startowej. Ponieważ nie ma dalej komendy, która nadpisze aktualną, słowo **await** nie jest konieczne. Cała komenda i tak zostanie wykonana. Jeśli chcesz, możesz użyć słowa **await** – w tym wypadku nie ma znaczenia. Kod:

```
motor.run_for_degrees(port.D, -50, 360)
```

Na koniec należy zsunąć paletę z manipulatora. Po jego opuszczeniu należy cofnąć robota. Robot wykonuje ruch do tyłu przez 360 stopni.

```
await motor_pair.move_for_degrees(motor_pair.PAIR_1, -360, 0)
```

Ostatnim kodem jest całkowite zatrzymanie silników funkcją stop:

```
motor_pair.stop(motor_pair.PAIR_1)
```

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Cały program wygląda następująco:

```
1 from hub import light_matrix, port
2 import runloop
3 import motor_pair, motor
4
5 async def main():
6
7     # write your code here
8     await light_matrix.write("Hi!")
9
10    #przypisanie portów A i B do PAIR_1
11    motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
12
13    #opuszczenie silnika dodatkowego aby łatwiej podnieść przedmiot o 300 milisekund (0,3 sekundy) na porcie D z prędkością -360 stopni na sekundę
14    #użycie funkcji run for time powoduje opuszczenie chwytaka bez jego zablokowania. Po 300 milisekundach program przejdzie dalej.
15    #w przypadku użycia funkcji run for degrees zły obrót mógłby zablokować cały program
16    await motor.run_for_time(port.D, 300, -360)
17
18    #jazda do przodu bez skręcania (0) przez ponad 1 obrót (450)
19    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 450, 0)
20
21    #podniesienie przedmiotu chwytakiem. Obróć silnik o 50 stopni (w górę) z prędkością 360 podłączonym do portu D
22    #wykorzystanie słowa await wstrzyma program do czasu wykonania ruchu - da to pewność podniesienia
23    await motor.run_for_degrees(port.D, 50, 360)
24
25    #jazda do przodu bez skręcania (0) przez 200 stopni
26    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 200, 0)
27
28    #skręcanie w prawo przez 130 stopni z pełnym skretem 100 ze standardową prędkością obrotu (wartość domyślna to 360 więc nie wpisujemy)
29    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 130, 100)
30
31    #jazda do przodu bez skręcania (0) przez 1337 stopni ~4,5 obrotów * 360 daje ~1337 stopni.
32    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1337, 0)
33
34    #opuszczenie przedmiotu chwytakiem. Obróć silnik o -50 stopni (w dół) z prędkością 360 podłączonym do portu D
35    #wykorzystanie słowa await nie jest potrzebne ponieważ nie występuje dalszy ruch tym silnikiem w programie
36    motor.run_for_degrees(port.D, -50, 360)
37
38    #jazda do tyłu bez skręcania (0) przez 360 stopni aby odjechać od skrzyni
39    await motor_pair.move_for_degrees(motor_pair.PAIR_1, -360, 0)
40
41    #zatrzymanie silników robota
42    motor_pair.stop(motor_pair.PAIR_1)
43
44 runloop.run(main())
```

Spróbuj zrealizować swoje zadanie, bazując na pozyskanej do tej pory wiedzy. Powodzenia i do następnej lekcji!

Możliwe rozszerzenia zadania

1. Transport ładunku po zmienionej trasie – uczniowie modyfikują program tak, aby robot dostarczył paletę do innego punktu docelowego, wymagającego dłuższego przejazdu lub dodatkowego skrętu.
2. Regulacja prędkości robota i manipulatora – uczniowie dodają parametr **velocity** i porównują, jak zmiana prędkości wpływa na stabilność jazdy oraz bezpieczeństwo transportu palety.
3. Zastosowanie ruchu na czas dla manipulatora – uczniowie wykorzystują funkcję **motor.run_for_time()** w sytuacji, gdy nie można dokładnie określić liczby stopni obrotu silnika.
4. Dodanie komunikatów dla użytkownika – robot wyświetla na matrycy Spike Hub lub w konsoli informacje o kolejnych etapach zadania, np. „Podnoszę ładunek”, „Transport”, „Odkładam paletę”.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 5: Odkrywamy tajemnice Kodu... Odczytać i zrozumieć Matrix

Temat lekcji nr 22 - Usprawniamy kod źródłowy za pomocą funkcji

Treści nauczania (wymagania szczegółowe - podstawa programowa) – I. 2) a); I. 2) b); I. 4); II. 1); II. 2); IV. 1)

Cel lekcji:

- **Wprowadzenie do programowania ruchu robota** – uczniowie poznają podstawowe zasady sterowania ruchem robota w języku Python oraz uczą się korzystania z gotowych bibliotek kodu w LEGO SPIKE Prime, a także tworzenia funkcji usprawniających zapis programu.
- **Analiza działania programu** – uczniowie analizują fragmenty kodu odpowiedzialne za ruch robota oraz działanie funkcji, określając, w jaki sposób poszczególne polecenia wpływają na kierunek i sposób poruszania się robota.
- **Tworzenie prostych algorytmów i programów sterujących ruchem** – uczniowie opracowują algorytmy oraz zapisują je w postaci programów tekstowych wykorzystujących podstawowe polecenia sterujące ruchem robota, biblioteki kodu oraz funkcje porządkujące wykonywanie powtarzalnych czynności.
- **Testowanie działania programu** – uczniowie uruchamiają program, obserwują sposób poruszania się robota oraz wprowadzają poprawki w celu uzyskania poprawnego działania programu, precyzyjnego ruchu robota oraz prawidłowego działania utworzonych funkcji..

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- **Uczeń dostrzega, że działanie programu i robota zależy od poprawności zapisanych poleceń:**
 - rozumie, że robot wykonuje instrukcje zgodnie z zapisanym kodem programu oraz poprawnie utworzonymi i wywoływanymi funkcjami,
 - obserwuje działanie programu sterującego ruchem robota i formułuje wnioski dotyczące poprawności działania poleceń oraz zastosowanych funkcji.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- **Uczeń analizuje działanie programu i jego efekty:**
 - sprawdza poprawność działania programu sterującego ruchem robota w języku Python,
 - identyfikuje błędy składniowe i logiczne oraz modyfikuje program poprzez zmianę poleceń, ich kolejności lub poprawę działania funkcji,
 - korzysta z gotowych bibliotek kodu umożliwiających sterowanie robotem,
 - tworzy funkcje usprawniające i porządkujące kod źródłowy programu.
- **Uczeń rozwija umiejętność testowania i doskonalenia rozwiązań:**
 - wykonuje próby działania programu i analizuje sposób poruszania się robota,
 - wprowadza poprawki w celu uzyskania poprawnego działania robota i utworzonych funkcji,
 - testuje różne rozwiązania związane ze sterowaniem ruchem robota, wykorzystaniem bibliotek kodu oraz modyfikowaniem funkcji w programie.

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści kształcenia będą realizowane za pomocą robotyki edukacyjnej LEGO® Education SPIKE™ Prime.

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna RaP STEAM,
- praca na komputerze,
- praca przy zestawie robotyki edukacyjnej,
- pogadanka.

Środki dydaktyczne

- zestaw PC z dostępem do Internetu,
- zestaw robotyki edukacyjnej LEGO® Education SPIKE™ Prime oraz zestaw rozszerzony LEGO® Education SPIKE™ Prime Expansion Kit,
- rzutnik multimedialny,
- e-podręcznik.

Formy pracy

- Praca indywidualna lub grupowa.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przebieg lekcji:

Faza I. Wprowadzenie

Na początku lekcji nauczyciel wprowadza uczniów w zagadnienia związane z programowaniem ruchu robota w języku Python, wyjaśniając, że robot wykonuje polecenia zapisane w programie w określonej kolejności oraz że kod można usprawnić dzięki **wykorzystaniu funkcji**. Następnie odnosi się do zadania lekcyjnego, polegającego na wykorzystaniu gotowych bibliotek kodu oraz **tworzeniu funkcji porządkujących kod źródłowy w środowisku LEGO SPIKE Prime**.

Nauczyciel przedstawia cel i zakres zadania. Prezentuje uczniom środowisko programowania w języku Python w aplikacji LEGO SPIKE App oraz omawia podstawowe elementy bibliotek wykorzystywanych do sterowania robotem, takie jak polecenia odpowiedzialne za ruch, zatrzymanie oraz zmianę kierunku jazdy. Następnie przekazuje uczniom informacje dotyczące **tworzenia i wywoływania funkcji** usprawniających zapis programu.

Uczniowie zapoznają się ze sposobami zapisu poleceń sterujących ruchem robota w języku Python oraz obserwują, jak robot po uruchomieniu programu wykonuje kolejne instrukcje i porusza się zgodnie z zapisanym kodem oraz **utworzonymi funkcjami**.

Następnie uczniowie zdobywają umiejętności obejmujące:

- określenie problemu (w jaki sposób wykorzystać funkcje do usprawnienia sterowania ruchem robota w języku Python),
- analizę działania programu sterującego ruchem robota,
- wykorzystanie gotowych bibliotek kodu do sterowania robotem,
- tworzenie prostych funkcji realizujących wybrane czynności robota,
- obserwację działania robota po uruchomieniu programu,
- testowanie programu i wprowadzanie prostych poprawek związanych ze sterowaniem ruchem robota oraz działaniem funkcji.

Na zakończenie tej fazy lekcji nauczyciel podaje temat do zapisania w zeszycie i przechodzi do fazy II – realizacji praktycznej.

Faza II. Część zasadnicza - realizacja

W tej fazie lekcji nauczyciel zapewnia uczniom praktyczne ćwiczenia, które pozwolą eksperymentować z koncepcjami programowania. W szczególności obejmować to będzie używanie blokowych języków programowania oraz zestawu do robotyki.

1. Nauczyciel objaśnia sposób pracy na lekcji – we wskazanym przez niego momencie uczniowie będą wykonywać kolejne etapy prowadzące do rozwiązania zadania. Zdolniejsi uczniowie spróbują wykonać je samodzielnie. Słabsi wykonają kolejne kroki

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

razem z nauczycielem, w oparciu o przygotowaną do zajęć instrukcję dostępną w e-podręczniku lub materiał multimedialny. W razie potrzeby będą mogli dopytać nauczyciela. Nauczyciel na bieżąco sprawdza postępy uczniów.

2. Bazując na posiadanych zasobach, zapewnij każdemu uczniowi lub zespołowi uczniów kompletny zestaw do robotyki (tj. LEGO® Education SPIKE™).
3. Zestawy te pozwalają uczniom budować własne roboty i programować je do wykonywania określonych zadań. Nauczyciel zaczyna od prostych projektów, takich jak zbudowanie i zaprogramowanie robota, który może wykonać proste czynności, które wprowadzą robota w ruch.
4. Uczniowie budują bazowego robota według załączonej instrukcji (załącznik **Budowa_bazy_robota.pdf** do lekcji).
5. Uczniowie włączają program **SPIKE**, poprzez który będą programować robota.
6. Nauczyciel, wykorzystując e-platformę, demonstruje na tablicy multimedialnej działanie programu i robota, uruchamiając odpowiedni moduł i materiał multimedialny.
7. Prezentowany materiał multimedialny, zrealizowany w oparciu o szczegółowy scenariusz (załącznik do lekcji), oprócz waloru teoretycznego stanowi również podstawę przebiegu części praktycznej lekcji. W szczególności, uczniowie:
 - a. budują prosty model bazowego robota w oparciu o zestaw do robotyki (tj. LEGO® Education SPIKE™),
 - b. tworzą listę czynności niezbędnych do wprowadzenia robota w ruch,
 - c. na podstawie listy czynności przygotowują program, zapisują go we wskazanym folderze (nauczyciel wskazuje właściwą lokalizację; może to być dysk lokalny lub zasób w chmurze) i potrafią go uruchomić;
 - d. wgrywają program do zbudowanego modelu robota, uruchamiają, obserwują i testują działanie robota. Wprowadzają modyfikacje w programie, poprawiają ewentualne błędy.

Faza III. Podsumowanie

Uczniowie analizują przedstawione przykłady programów sterujących ruchem robota zapisanych w języku Python oraz omawiają ich działanie. Nauczyciel ocenia stopień zrozumienia zależności między zapisanym kodem, **utworzonymi funkcjami**, a sposobem poruszania się robota po uruchomieniu programu.

Kryteria oceniania:

Ocena dopuszczająca:

- uczeń rozpoznaje podstawowe elementy programu w języku Python,
- potrafi wskazać polecenia odpowiedzialne za ruch robota,
- rozpoznaje proste funkcje zapisane w programie.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Ocena dostateczna:

- uczeń analizuje przykładowy program i opisuje jego działanie,
- rozumie zależność między zapisanym kodem, funkcjami, a działaniem programu,
- rozpoznaje podstawowe polecenia wykorzystywane do sterowania robotem,
- wyjaśnia rolę funkcji usprawniających zapis programu.

Ocena dobra:

- uczeń samodzielnie modyfikuje przykładowy program,
- poprawia błędy związane z zapisem poleceń, nawiasami lub cudzysłowami,
- testuje działanie programu sterującego ruchem robota po wprowadzonych zmianach,
- tworzy własne funkcje usprawniające wykonywanie powtarzalnych czynności robota.

Ocena bardzo dobra:

- uczeń samodzielnie tworzy poprawny program w języku Python,
- wyjaśnia działanie poszczególnych elementów programu oraz utworzonych funkcji,
- wykorzystuje biblioteki kodu do sterowania ruchem robota,
- proponuje własne rozwiązania i potrafi je uzasadnić,
- samodzielnie analizuje i poprawia błędy w kodzie,
- stosuje funkcje w celu uporządkowania i optymalizacji kodu źródłowego programu.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, np.:

1. Do czego służy funkcja w programie?
 - a. **Do wielokrotnego używania tego samego kodu**
 - b. Do wyłączania robota
 - c. Do zmiany baterii
 - d. Do przyspieszania komputera
2. Jak rozpoczynamy definiowanie funkcji w Pythonie?
 - a. Słowem start
 - b. Słowem function
 - c. **Słowem def**
 - d. Słowem begin
 - e.
3. Co musi znaleźć się po nazwie funkcji?
 - a. **Nawiasy**
 - b. Kropka
 - c. Średnik
 - d. Przecinek

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

4. Co oznacza dwukropek na końcu definicji funkcji?
- Koniec programu
 - Początek bloku kodu**
 - Błąd składni
 - Zatrzymanie robota

Faza V. Dodatkowe, praca domowa (dla chętnych).

Zadania dodatkowe (dla chętnych).

Zadanie 1: Usprawniamy ruch robota za pomocą funkcji

Utwórz prosty program w języku Python, który spowoduje wykonanie przez robota kilku podstawowych ruchów oraz będzie **wykorzystywał funkcje** usprawniające kod źródłowy.

Program powinien:

- zawierać polecenia odpowiedzialne za ruch robota,
- wykorzystywać gotowe biblioteki kodu LEGO SPIKE Prime,
- zawierać co najmniej dwie funkcje, np. `jazda_do_przodu()`, `skret()`, `stop()`,
- być zapisany poprawnie pod względem składni,
- wykonywać ruchy w logicznej kolejności,
- umożliwiać uruchomienie i przetestowanie działania robota.

Opcjonalnie możesz rozbudować program tak, aby robot:

- zatrzymywał się po wykonaniu ruchu,
- zmieniał kierunek jazdy,
- wykonywał skręt o określony kąt,
- poruszał się z różną prędkością,
- wykonywał przygotowaną trasę przejazdu,
- wykorzystywał funkcje z parametrami, np. prędkość lub czas jazdy,
- realizował kilka powtarzalnych manewrów bez powielania tych samych fragmentów kodu.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 22 - *Usprawniamy kod źródłowy za pomocą funkcji.*

Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
 - Nauczyciel omawia podstawowe zagadnienia związane z programowaniem ruchu robota w języku Python:
 - wyjaśnia, że robot wykonuje instrukcje zapisane w programie w określonej kolejności,
 - przedstawia przykłady poleceń wykorzystywanych do sterowania ruchem robota,
 - omawia znaczenie poprawnej składni programu, wykorzystania bibliotek kodu oraz funkcji usprawniających kod źródłowy.
 - Wprowadzenie do zadania i przypomnienie podstaw pracy w LEGO SPIKE Prime:
 - omówienie środowiska programowania Python w aplikacji LEGO SPIKE App,
 - przedstawienie podstawowych bibliotek wykorzystywanych do sterowania robotem,
 - przedstawienie zależności: zapis kodu >> **utworzenie funkcji** >> uruchomienie programu >> ruch robota,
 - analiza przykładowych programów odpowiedzialnych za ruch robota oraz zastosowanie funkcji w programie.
 - Praca uczniów:
 - analiza przedstawionych przykładów programów sterujących ruchem robota,
 - omawianie działania poszczególnych linii kodu oraz utworzonych funkcji,
 - testowanie działania robota po uruchomieniu programu,
 - wyszukiwanie i poprawianie błędów w kodzie oraz funkcjach,
 - proponowanie własnych rozwiązań związanych ze sterowaniem ruchem robota oraz **tworzenie funkcji usprawniających wykonywanie powtarzalnych ruchów robota.**
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów (analiza poprawności działania programu, sposobu poruszania się robota oraz umiejętności **wykorzystania bibliotek kodu i funkcji**).
- Zadanie pracy domowej (opcjonalnie) – napisanie prostego programu sterującego ruchem robota **z wykorzystaniem funkcji** oraz przetestowanie działania programu po wprowadzeniu własnych zmian.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 2 – Realizacja zadania głównego podczas lekcji nr 22 - *Usprawniamy kod źródłowy za pomocą funkcji.*

Zadanie 1. Przygotowanie do realizacji zadania głównego

Uczniowie, pracując w zespołach lub indywidualnie, realizują zadanie polegające na poznaniu sposobu korzystania z bibliotek kodu w języku Python, programowaniu podstawowych ruchów robota oraz **usprawnianiu kodu źródłowego za pomocą funkcji** w środowisku LEGO SPIKE Prime. Szczególny nacisk położony jest na poprawne wykorzystanie gotowych bibliotek, rozumienie działania importowanych modułów oraz **tworzenie funkcji porządkujących program** sterujący ruchem robota.

Uczniowie opracowują i zapisują rozwiązanie problemu w postaci prostych programów sterujących ruchem robota oraz wyświetlaniem komunikatów na kontrolerze SPIKE Hub. W tym celu:

- analizują przykładowe programy zapisane w języku Python wykorzystujące biblioteki kodu i funkcje,
- określają cel programu oraz przewidują możliwe błędy składniowe i logiczne,
- planują kolejność wykonywania instrukcji sterujących robotem,
- opracowują algorytm w uporządkowanej formie,
- tworzą funkcje odpowiedzialne za wybrane ruchy robota lub powtarzalne czynności,
- zapisują rozwiązanie jako program w języku Python z wykorzystaniem bibliotek i modułów,
- testują działanie programu i obserwują sposób poruszania się robota,
- wprowadzają poprawki związane ze sterowaniem ruchem, działaniem funkcji lub poprawnością składni kodu,
- doprecyzowują rozwiązanie tak, aby program był czytelny, uporządkowany i skuteczny w działaniu.

Uczniowie wykorzystują środowisko LEGO SPIKE Prime w trybie Python, zwracając uwagę na zgodność między zapisanym kodem, utworzonymi funkcjami a rzeczywistym działaniem robota. Szczególną uwagę poświęcają wykorzystaniu bibliotek i modułów, poprawnemu zapisowi poleceń oraz analizie komunikatów o błędach pojawiających się po uruchomieniu programu.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 2. Wprowadzenie teoretyczne do lekcji

Jak uprościć programy? Funkcjami!

Wyobraź sobie, że chcesz, aby Twój robot wykonał skomplikowany manewr: przejechał metr do przodu, obrócił się o 90 stopni, zamrugał światłami i wydał dźwięk zwycięstwa. Możesz zapisać te wszystkie instrukcje jedna po drugiej. Ale co, jeśli robot ma wykonać ten sam manewr pięć razy w różnych momentach programu? Czy za każdym razem będziesz przepisywać te same dziesięć linijek kodu? Oczywiście możesz powiedzieć, że wystarczy pętla, która wykona się pięć razy – jak najbardziej. Co, jeśli program jest na tyle duży, że ten manewr musi wykonać w różnych momentach? Przykład:

- Robot jedzie metr do przodu
- Obraca się o 90 stopni
- Wyświetla strzałkę i daje sygnał jazdy
- Jedzie po łuku w prawo przez dystans 30 centymetrów
- Obraca się o 90 stopni
- Mruga światłami i wydaje dźwięk
- Robot jedzie metr do przodu
- Obraca się o 90 stopni
- Mruga światłami i wydaje dźwięk

Czy takie zadanie da się zrobić w pętli? Tej samej? Musiałby wykonać te same zadania a tu jest problem. Raz robot jedzie do przodu, raz skręca. Skąd wiesz, kiedy? Bo jest napisane, tak. A jeśli programista nie do końca ma to wiedzieć? Czy w przypadku drobnej poprawki przepiszesz cały kod od zera? Całkowicie bez sensu. Dlatego ktoś mądry stworzył funkcje.

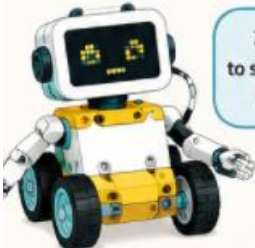
Czym są funkcje?

Funkcja to nazwany blok kodu, który wykonuje konkretne zadanie. Możesz o niej myśleć jak o „przepisie”. Zamiast za każdym razem tłumaczyć mamie, jak zrobić Twoją ulubioną kanapkę (krok po kroku: weź chleb, posmaruj masłem, połóż ser...), możesz po prostu powiedzieć: „Mamo, poproszę tę kanapkę”. Jedno hasło zastępuje całą listę instrukcji, ponieważ wszystkie instrukcje kryją się pod słowami Mamo, poproszę tę kanapkę. Proste? Oby. Zastosujmy to w praktyce. Na dzisiejszej lekcji poznasz idee funkcji. Na następnych zajęciach przebudujesz jeden z programów, aby wykorzystywał funkcje w kodzie. W kolejnej lekcji przetestujesz to na żywo!

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Podsumowanie informacji o funkcjach:

JAK UPROŚCIĆ PROGRAMY? FUNKCJAMI!



Zamiast pisać to samo wiele razy... użyj funkcji!

Wyobraź sobie, że chcesz, aby Twój robot wykonał skomplikowany manewr: przejechał metr do przodu, obrócił się o 90 stopni, zamrugał światłami i wydał dźwięk zwycięstwa.

Możesz zapisać te wszystkie instrukcje jedna po drugiej.

Ale co, jeśli robot ma wykonać ten sam manewr pięć razy w różnych momentach programu?

Czy za każdym razem będziesz przepisywać te same dziesięć linijek kodu?



PRZYKŁAD MANEWRU:

- Robot jedzie metr do przodu
- Obraca się o 90 stopni
- Wyświetla strzałkę i daje sygnał jazdy
- Jedzie po łuku w prawo przez dystans 30 cm
- Obraca się o 90 stopni
- Mrga światłami i wydaje dźwięk
- Robot jedzie metr do przodu
- Obraca się o 90 stopni
- Mrga światłami i wydaje dźwięk

CZY TO DA SIĘ ZROBIĆ W PĘTLI?

Nie zawsze! Pętla powtarza te same instrukcje, a tu robot wykonuje różne zadania w określonej kolejności. Co więcej – co jeśli programista nie powinien „wiedzieć” tego na pamięć? A co, jeśli trzeba coś zmienić?

Przepisywanie całego kodu od nowa?

To nie ma sensu!

Dlatego ktoś mądry stworzył funkcje.



CZYM SĄ FUNKCJE?

Funkcja to nazwany blok kodu, który wykonuje konkretne zadanie. Możesz o niej myśleć jak o „przepisie”.

Zamiast za każdym razem tłumaczyć mamie, jak zrobić Twoją ulubioną kanapkę (krok po kroku: weź chleb, posmaruj masłem, połóż ser...), możesz po prostu powiedzieć: „Mamo, poproszę tę kanapkę”.

Jedno hasło zastępuje całą listę instrukcji. Wszystkie instrukcje kryją się pod słowami „Mamo, poproszę tę kanapkę”. Proste? Oby.



Mamo, poproszę tę kanapkę.

BEZ FUNKCJI	VS	Z FUNKCJĄ
Powtarzasz ten sam kod w wielu miejscach.		Wywołujesz funkcję, gdy jej potrzebujesz.
<pre>... motor.run_for_degrees(...) motor.run_for_degrees(...) hub.light_matrix.write(...) hub.speaker.beep() ... # znowu to samo... motor.run_for_degrees(...) motor.run_for_degrees(...) hub.light_matrix.write(...) hub.speaker.beep() ...</pre>		<pre>... manewr() ... # w innym miejscu manewr() ... # a tutaj jeszcze raz manewr() ...</pre>
 Długi, trudny do edycji, łatwo o błędy.		 Czytelny, krótki, łatwy w zmianie i rozbudowie.

CO DAJĄ FUNKCJE?

- Mniej powtarzania kodu
- Łatwiejsze czytanie i zrozumienie
- Prostsze poprawki i modyfikacje
- Większa niezawodność programu

CO BĘDZIE DALEJ?

- Na dzisiejszej lekcji poznasz idee funkcji.
- Na następnych zajęciach przebudujesz jeden z programów, aby wykorzystywał funkcje w kodzie.
- W kolejnej lekcji przetestujesz to na żywo!

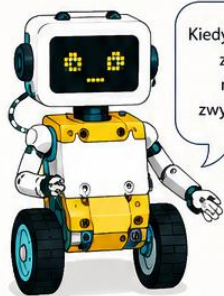


 **FUNKCJE TO TWÓJ SPRYTNY SKRÓT DO LEPSZYCH PROGRAMÓW!** 

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

POGADAJMY JESZCZE O FUNKCJACH

★ INNY PRZYKŁAD ★



Kiedy osiągam cel,
zasługuję
na taniec
zwycięstwa! 🏆

Wyobraź sobie, że chcesz, aby Twój robot
wykonał radosny taniec zwycięstwa po każdym
wykonanym zadaniu.

Taki taniec składa się z:



obrotu
360°



mignięcia
diodami



krótkiego
dźwięku

ZAMIAST TEGO...

Zamiast wpisywać te trzy polecenia
za każdym razem, gdy robot
coś osiągnie...

stwórz jedną funkcję
o nazwie
taniec_zwyciezcy

Od tego momentu, wpisując w kodzie
tylko to jedno hasło, uruchamiasz całą
zapisaną w nim procedurę.

1. BEZ FUNKCJI – DUŻO POWTÓRZEŃ

Za każdym razem, gdy robot wykona zadanie,
musisz przepisać te same polecenia.

Zadanie 1

🌀 Obrót 360°

💡 Mignięcie diodami

🎵 Dźwięk

Zadanie 3

🌀 Obrót 360°

💡 Mignięcie diodami

🎵 Dźwięk

Zadanie 3

🌀 Obrót 360°

💡 Mignięcie diodami

🎵 Dźwięk

! Dużo kodu, trudniej go czytać
i poprawiać.



2. Z FUNKCJĄ – PROSTO I PRZEJRZYŚCIE

Piszesz taniec raz, a potem wywołujesz go, kiedy
robot osiąga kolejne cele.

DEFINICJA FUNKCJI

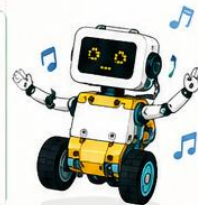
```
def taniec_zwyciezcy():  
    motor.run_for_degrees('B', 360, 50)  
    hub.light_matrix.show_image(IMAGE.HAPPY)  
    hub.speaker.beep(70, 0.3)
```

Tu opisujesz,
jak wygląda
taniec
zwycięstwa.

UŻYCIE FUNKCJI

Tutaj wywołujesz
taniec jednym
prostym hasłem.

```
# Zadanie 1  
taniec_zwyciezcy()  
# Zadanie 2  
taniec_zwyciezcy()  
# Zadanie 3  
taniec_zwyciezcy()
```



Mało kodu, łatwe zmiany, porządek w programie!

CO, GDY CHCĘ COŚ ZMIEŃĆ W TAŃCU?

BEZ FUNKCJI

Musisz znaleźć każde miejsce i zmienić wszystkie linie
z osobna.

Dużo pracy i ryzyko błęd!



VS

Z FUNKCJĄ

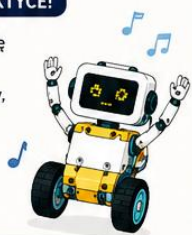
Wystarczy, że zmienisz taniec w jednym miejscu –
funkcja aktualizuje się wszędzie automatycznie!



Funkcje
oszczędzają czas,
zmniejszają błędy
i sprawiają, że
kod jest czytelny
i przyjemny
w tworzeniu. 😊

SPRÓBUJ W PRAKTYCE!

Stwórz własną funkcję
i nadaj jej nazwę,
np. **taniec_zwyciezcy**,
a potem używaj jej
za każdym razem,
gdy robot wygra!



PRZYKŁAD:

```
def taniec_zwyciezcy():  
    motor.run_for_degrees('B', 360, 50)  
    hub.light_matrix.show_image(IMAGE.HEART)  
    hub.speaker.beep(70, 0.3)  
  
# Robot wykona zadania...  
# ...a potem zatańczy!  
taniec_zwyciezcy()  
taniec_zwyciezcy()
```



FUNKCJE TO TWOJE SEKRETNE SUPERMOCE W PROGRAMOWANIU!



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Mądrzejsze funkcje – pora na funkcje z parametrami

Największą zaletą funkcji nie jest tylko to, że oszczędzają nam pisanie, ale to, że potrafią się adaptować do sytuacji. Służą do tego parametry, czyli informacje, które przekazujemy do wnętrza funkcji w nawiasach. Rozważmy funkcję odpowiedzialną za jazdę. Zamiast tworzyć osobne bloki kodu dla jazdy o dziesięć, dwadzieścia czy pięćdziesiąt centymetrów, tworzysz jedną funkcję `jedz`, która przyjmuje parametr odległości. Dzięki temu w jednym miejscu programu możesz nakazać robotowi `jedz(10)`, a w innym `jedz(50)`. To sprawia, że Twój program staje się niesamowicie elastyczny, a Ty nie musisz powtarzać tych samych instrukcji dla różnych wartości. W programowaniu robotów bardzo często jest potrzeba, aby funkcja nie tylko coś zrobiła, ale też coś nam powiedziała. Wyobraź sobie funkcję, która sprawdza odległość od ściany za pomocą czujnika ultradźwiękowego. Takie zajęcia przed nami. Taka funkcja działa jak zwiadowca: wysyła ją przodem, ona wykonuje pomiar i wraca do nas z gotową informacją. Używa się do tego słowa kluczowego **return**. Dzięki niemu można napisać program, który „pyta” funkcję o stan otoczenia i na tej podstawie podejmuje decyzję o dalszym ruchu. To właśnie dzięki takim „odpowiedziom” funkcje stają się fundamentem budowania sztucznej inteligencji robota.

Poważnym wyzwaniem dla każdego programisty jest sytuacja, w której w długim kodzie pojawia się błąd. Jeśli napisano program składający się z tysiąca linii bez użycia funkcji, znalezienie przyczyny usterki jest jak szukanie igły w stogu siana. Stosując funkcje, dzielisz program na mniejsze, logiczne moduły. Jeśli robot źle skręca, wiesz dokładnie, że problem leży w definicji funkcji odpowiedzialnej za skręcanie, a nie w całym kodzie. Naprawienie błędu w jednym miejscu automatycznie poprawia działanie robota w każdym momencie, w którym ta funkcja została użyta. To właśnie nazywamy dobrą jakością oprogramowania i efektywnym zarządzaniem projektem. Używając zestawów LEGO Spike, przyzwyczajono się do fizycznego łączenia elementów. Funkcje w Pythonie pozwalają Ci robić dokładnie to samo w świecie kodu. Możesz stworzyć bibliotekę własnych, sprawdzonych funkcji do jazdy, omijania przeszkód czy chwytania przedmiotów, a następnie składać z nich coraz bardziej skomplikowane misje. Dzięki temu Twoja praca przypomina pracę architekta, który z gotowych i przetestowanych modułów buduje nowoczesny wieżowiec.

Funkcje sprawiają, że programowanie przestaje być żmudnym wpisywaniem komend, a staje się kreatywnym projektowaniem zachowań. I tego będziesz uczyć się na kolejnych lekcjach stosując funkcje w kodzie.

Dlaczego?

Łatwość edycji błędów, brak konieczności przepisywania lub kopiowania tego samego fragmentu kodu.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Podsumowanie informacji o funkcjach z parametrami:

Jedna funkcja, wiele możliwości!

MĄDRZEJSZE FUNKCJE

FUNKCJE Z PARAMETRAMI

PARAMETR
to informacja, którą przekazujemy do funkcji.

jedz(50)

↑ nazwa funkcji ↑ wartość parametru

PRZYKŁAD: JAZDA

Zamiast wielu bloków kodu – jedna funkcja!

jedz(10) →

jedz(20) →

jedz(50) →

PRZYKŁAD 1: Funkcja z parametrem (wejście)

```
def jedz(odleglosc_cm):
    motor.run_for_degrees('B', odleglosc_cm * 3.6)
    motor.run_for_degrees('C', odleglosc_cm * 3.6)

# w jednym miejscu
jedz(10)
jedz(50)
jedz(25)
```

10 cm, 50 cm, 55 cm

PRZYKŁAD 2: Funkcja z wartością zwracaną (return)

```
def odleglosc_od_sciany():
    d = distance.get_distance_cm()
    return d

# pytamy funkcję o wynik
dystans = odleglosc_od_sciany()

if dystans < 20:
    hub.display.show_image(IMAGE.WARNING)
    hub.speaker.beep()
else:
    hub.display.show_image(IMAGE.HAPPY)
```

Mierzę... i wracam z wynikiem!

DLACZEGO FUNKCJE SĄ TAK WAŻNE?

ŁATWOŚĆ EDYCJI BŁĘDÓW

Poprawiasz raz, działa wszędzie!

OSZCZĘDNOŚĆ KODU

Nie przepisujesz tego samego!

CZYTELNOŚĆ KODU

Program jest krótki, przejrzysty i logiczny!

ELASTYCZNOŚĆ

Działa z różnymi wartościami!

MODUŁY I BUDOWANIE W ŁATWY SPOSÓB

Z funkcji tworzysz coraz większe misje!

FUNKCJE W PRAKTYCE

Przeanalizuj → Zaprojektuj funkcję → Użyj jej w kodzie → Testuj → Ulepszaj i rozwijaj

Funkcje sprawiają, że programowanie staje się kreatywne, elastyczne i przyjemne!

PRZYKŁAD: TANIEC ZWYCIĘZCY

```
def taniec_zwyciezcy(ilosc_obrotow=1):
    motor.run_for_degrees('B', 360*ilosc_obrotow)
    hub.light_matrix.show_image(IMAGE.HEART)
    hub.speaker.play_sound(Sound.FILE_FINISHED)

# wywołanie z różną liczbą obrotów
taniec_zwyciezcy(1) # 1 obrót

# w innym miejscu
taniec_zwyciezcy(3) # 3 obroty
```

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Jak wygląda robienie funkcji dla środowiska LEGO SPIKE Prime w trybie Python?

W Pythonie zarówno z wykorzystaniem LEGO jak i bez (nazywamy to normalnym Pythonem czyli nie dedykowanym stricte pod LEGO) funkcje robi się tak samo – co jest jak najbardziej naturalne. Czy wiesz, że od początku klasy 7 pracujesz z funkcjami? A przede wszystkim jedną, cały czas, od początku. Wiesz którą? Popatrz w kod startowy:



Connect

```
1 from hub import light_matrix
2 import runloop
3
4 async def main():
5     # write your code here
6     await light_matrix.write("Hi!")
7
8 runloop.run(main())
9
```

Tak, dokładnie. To funkcja **main()**. Funkcja **main()** jest najważniejszą funkcją w Pythonie. To ona „mówi” językowi, gdzie jest początek wykonywania całego programu. Tu zaczyna się cała magia. Funkcja **main()** musi zawsze występować w programie Python dla LEGO. Bez tego wszystko się wysypie. Jak wygląda definiowanie funkcji? Bardzo prosto, wystarczy jedno słowo na początku:

```
def main():
    # write your code here
    await light_matrix.write("Hi!")
```

Popatrz co się dzieje. Pojawia się funkcja **main()**. Przed nią jest słowo **def**.

def (zawsze pisane z małej litery!) oznacza definicję funkcji. Python widząc to słowo już wie czego się spodziewać. Po słowie **def** występuje tekst, w tym przypadku **main()**. Jest to nazwa funkcji. Jak wyżej użyto słowa **jedz** przy jeździe robota czy **taniec_zwyciezcy** to to jest właśnie to miejsce. Tutaj ustalasz nazwę funkcji. Będziesz ją używać w swoim kodzie, aby wykonać tą część bloku kodu. **Obowiązkowe jest dodawanie nawiasów**. Niezależnie czy funkcja będzie mieć parametry czy nie, nawiasy muszą wystąpić. Jeśli funkcja nie przyjmuje parametrów to otwierasz i zamykasz nawias. Ostatnim elementem jest dwukropek **:**. Oznacza on początek bloku kodu. W innych językach programowania używa się klamer {}, w Pythonie są to wcięcia w kodzie i dwukropek. Po dwukropku należy wcisnąć klawisz **enter**.

W Pythonie kolejna linia jest przesunięta I TAK MA BYĆ.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Python wie wtedy, że to co jest przesunięte to część funkcji. O ile przesunięte? Dokumentacja Pythona mówi o dwóch sposobach: **Jeden tabulator** lub **Cztery spacje**.

Korzystaj tylko z jednej opcji. Nie mieszaj, bo się pogubisz. Jeśli czujesz się swobodnie ze spacjami – używaj **spacje**. Jeśli lepiej dla Ciebie używać przycisku **Tab** na klawiaturze – używaj **tab**. Jeśli będziesz wymieniać **tab** i **spacje** to twój program się posypie. Dojść do błędu nie jest łatwo, dlatego warto trzymać się jednego standardu. Tyle z teorii. Spróbuj zadań praktycznych, aby poćwiczyć funkcje. Powodzenia!

Podsumowanie informacji nt. funkcji dla LEGO SPIKE Prime w trybie Python.

JAK WYGLĄDA ROBIECIE FUNKCJI W LEGO SPIKE PRIME – TRYB PYTHON?

CZY WIESZ, ŻE...?

Od początku klasy 7 pracujesz z funkcjami!
A przede wszystkim z jedną, cały czas, od początku.

Wiesz którą? Popatrz w kod startowy:

Kod startowy w LEGO SPIKE Prime (Python)

```
import hub, motor, time, math
from hub import port

async def main():
    # write your code here
    await hub.light_matrix.write("Hello!")

# Uruchomienie programu
await main()
```

Funkcje robi się w Pythonie tak samo – zarówno w LEGO SPIKE Prime, jak i w „normalnym” Pythonie!

Tak! To funkcja `main()`.
To ona mówi językowi, gdzie jest początek wykonywania całego programu. Tu zaczyna się cała magia. Bez niej wszystko się wysypie!

JAK WYGLĄDA DEFINIOWANIE FUNKCJI?

`def main():`

- 1. def** (zawsze małe litery!)
To słowo kluczowe oznacza definicję funkcji. Python wie już, czego się spodziewać.
- 2. nazwa funkcji**
Tu ustalasz nazwę funkcji. Np. jedz, skręt, taniec_zwycieczki – to miejsce, w którym nadajesz nazwę. Będziesz jej używać w swoim kodzie.
- 3. nawiasy ()**
Obowiązkowe! Niezależnie czy funkcja będzie mieć parametry czy nie – nawiasy muszą wystąpić.
- 4. dwukropek :**
Oznacza początek bloku kodu. W innych językach są tu klamry {}, w Pythonie są to wcięcia i dwukropek.

CO DALEJ?

Po dwukropku wciśnij Enter.
Kolejna linia jest przesunięta
I TAK MA BYĆ!
Python wie wtedy, że to, co jest przesunięte, to część funkcji.

```
def main():
    # write your code here
    await light_matrix.write("Hi!")
```

PRZYKŁAD W PRAKTYCE

Prosta funkcja `main()`:

```
import hub

async def main():
    # write your code here
    await hub.light_matrix.write("Hi!")
    await hub.speaker.beep()

await main()
```

- Kod wewnątrz funkcji jest wcięty.
- Funkcja `main()` jest wywoływana na końcu programu poleceniem: `await main()`

O ILE WCINAĆ?

Dokumentacja Pythona mówi o dwóch sposobach:

1 TABULATOR

```
def funkcja():
→ kod
```

4 SPACJE

```
def funkcja():
    kod
```

Korzystaj tylko z jednej opcji. Nie mieszaj – bo się pogubisz i program się posypie.

⚠ Nie mieszaj Tab i spacji! Trzymaj się jednego standardu.

PO CO UŻYWAMY FUNKCJI?

PORZĄDEK W KODZIE

Dzieliś program na mniejsze, logiczne części.

WIELOKROTNE UŻYCIE

Piszesz raz – używasz wielokrotnie.

ŁATWOŚĆ POPRAWEK

Poprawiasz błąd w jednym miejscu, a działa wszędzie, gdzie funkcja jest użyta.

ELASTYCZNOŚĆ

Dzięki parametrom funkcje potrafią się dopasować do sytuacji.

BUDOWANIE WIĘKSZYCH PROJEKTÓW

Z funkcji tworzysz „bibliotekę” swoich sprawdzonych rozwiązań.

Funkcje to potężne narzędzie, które sprawia, że programowanie staje się łatwiejsze, przyjemniejsze i bardziej profesjonalne!

NAJWAŻNIEJSZE ZASADY:

- ✓ Używaj słowa `def` na początku funkcji.
- ✓ Nadaj funkcji nazwę i dodaj nawiasy `()`.
- ✓ Zakończ definicję dwukropkiem `:` i wciśnij Enter.
- ✓ Wcięcie oznacza, co należy do funkcji.
- ✓ Nie mieszaj Tab i spacji.

Ćwiczenia czynią mistrza!
Przejdź do zadań praktycznych i poćwicz funkcje!
Powodzenia! 🚀

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 2. Przykład praktyczny

Zadanie:

Napisz dla środowiska LEGO SPIKE Prime w trybie Python funkcję **imie()**, która wyświetla twoje imię na matrycy/wyświetlaczu kostki Spike Hub.

Realizacja praktyczna:

```
1 from hub import light_matrix
2 import runloop
3
4 def imie():
5     light_matrix.write("Spike")
6
7 async def main():
8     # write your code here
9     imie()
10
11 runloop.run(main())
```

Powyższy przykład realizuje zadanie, w tym celu utworzona została funkcja „imie”, która wyświetla na kostce Spike Hub imię (w tym przypadku „Spike”). Następnie funkcja została wywołana czego rezultatem było wyświetlenie napisu na kostce Spike Hub.

Zadanie do samodzielnego rozwiązania – 01

Utwórz funkcję o nazwie **start**, która wyświetli kolejno liczby od 3 do 1 a następnie napis „START”. Następnie wywołaj funkcję i zweryfikuj jej działanie.

Zadanie do samodzielnego rozwiązania – 02

Napisz funkcję, która wyświetla wynik dodawania dwóch liczb podawanych do funkcji jako parametry. Opcjonalnie możesz zmodyfikować funkcję tak aby wypisywała w konsoli kolejno wyniki poszczególnych operacji matematycznych (dodawanie, odejmowanie, mnożenie i dzielenie).

Zadanie do samodzielnego rozwiązania – 03

Napisz funkcje, które będą wykonywały operacje matematyczne (dodawanie, odejmowanie, mnożenie i dzielenie) na dwóch liczbach podawanych do funkcji jako parametry. Wynik

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

danego działania powinien być zwracany za pomocą **return**. Wyświetl w konsoli przykłady wywołania każdej funkcji.

Zadanie do samodzielnego rozwiązania – 04

Znajdź i popraw błędy (minimum 3) w poniższym kodzie, następnie uruchom poprawiony kod i zweryfikuj jego działanie.



Connect

```

1 from hub import light_matrix
2 import runloop
3
4 def dodawanie(a,b):
5     wynik = a + c
6     return wynik
7
8
9 async def main():
10     # write your code here
11     print(dodawanie(5,8,9))
12
13 runloop.run(main())
14

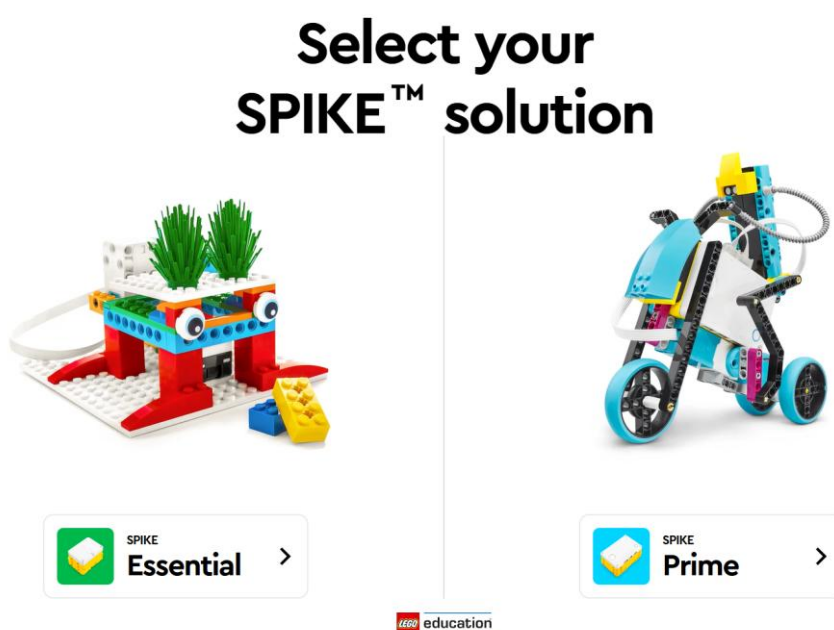
```

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 3. Konfiguracja środowiska.

Uruchamianie aplikacji LEGO Spike

1. Włącz przeglądarkę internetową dostępną na twoim urządzeniu
2. Wpisz w przeglądarce adres: <https://spike.legoeducation.com/> Pojawi się okno wyboru wersji LEGO SPIKE. Wybierz Spike Prime

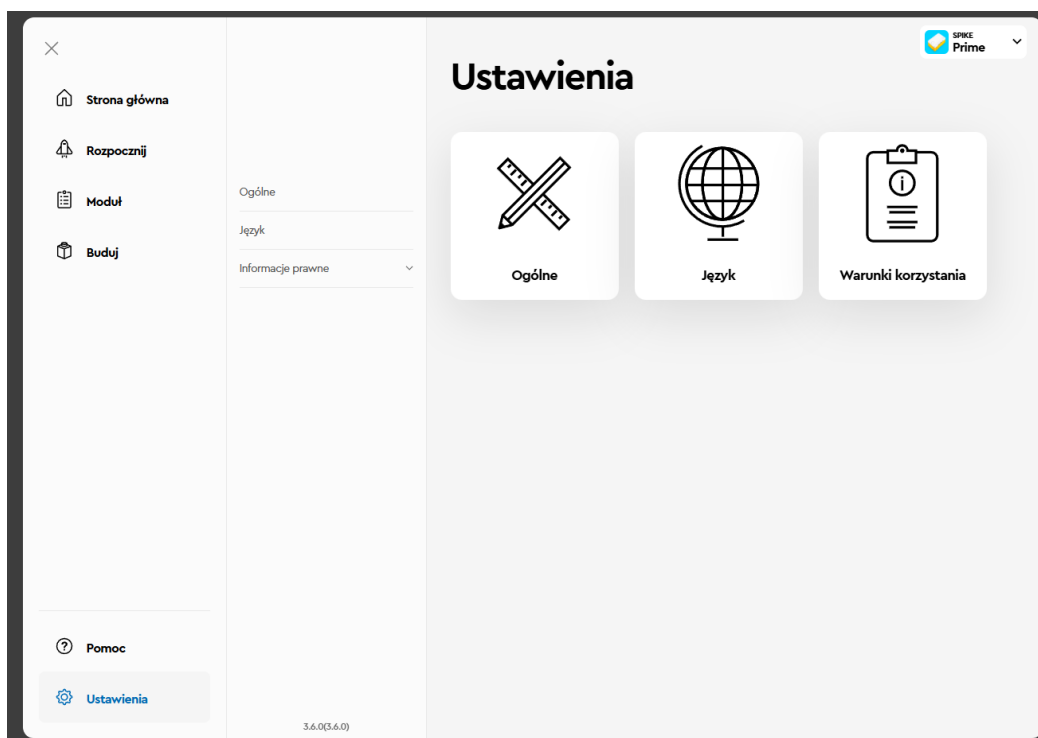


3. Po wybraniu opcji SPIKE Prime pojawia się menu główne. Jeśli występuje język polski, należy zmienić go na język angielski.

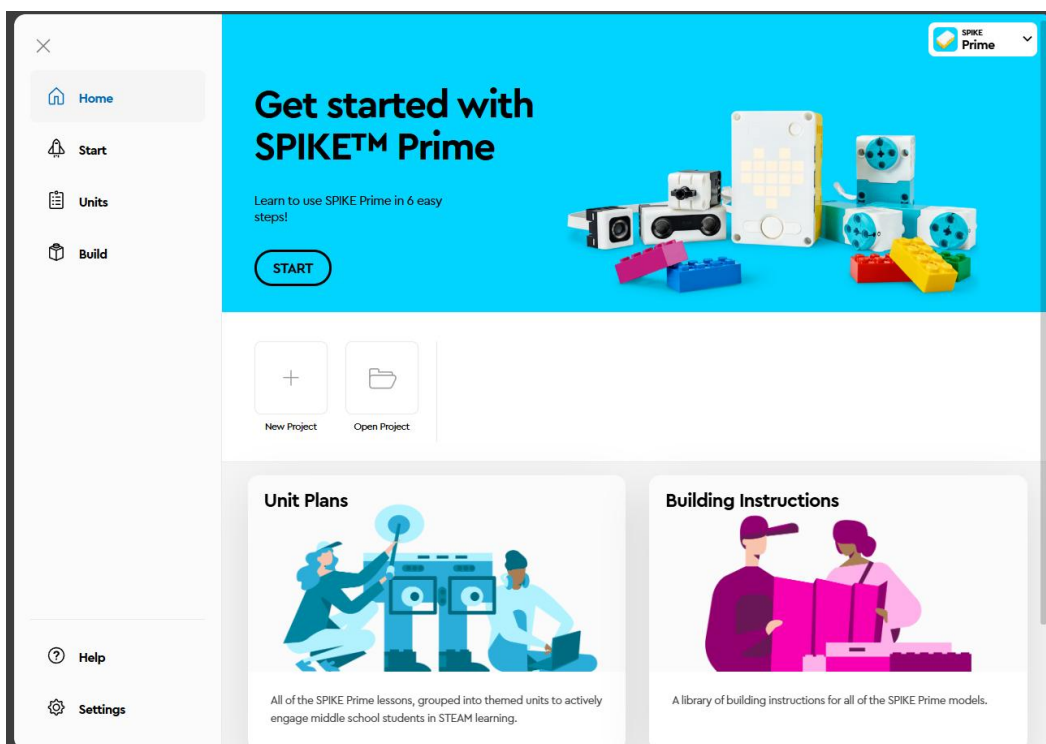
Jeżeli zaś wszystko jest w **języku angielskim**, to przejdź do kroku 6.

Aby zmienić język, w lewym dolnym rogu menu głównego kliknij przycisk **Ustawienia**.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

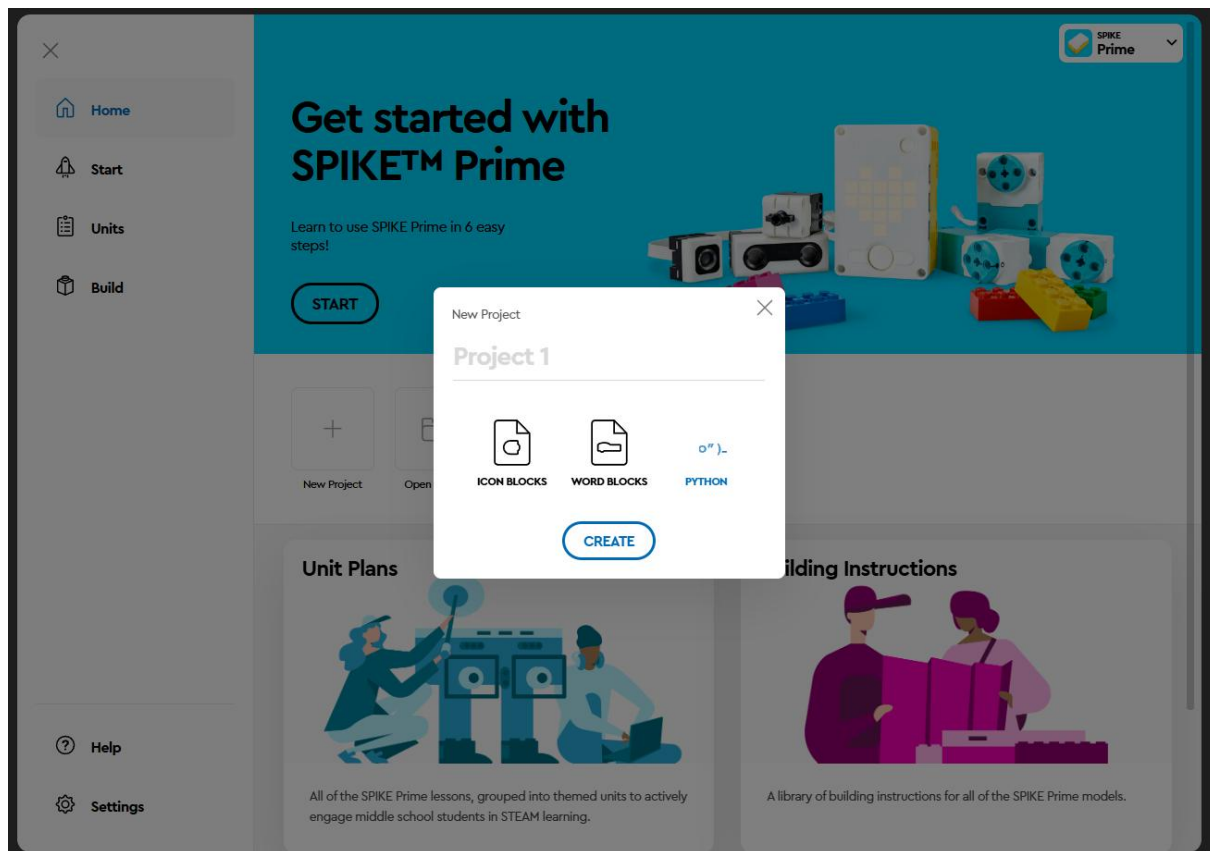


4. W nowym oknie kliknij globus podpisany „Language”.
5. Wybierz język: **United States (English)**. Strona przeładuje się i język angielski będzie dostępny.
6. Kliknij po lewej przycisk **Home**. Pojawi się okno jak poniżej

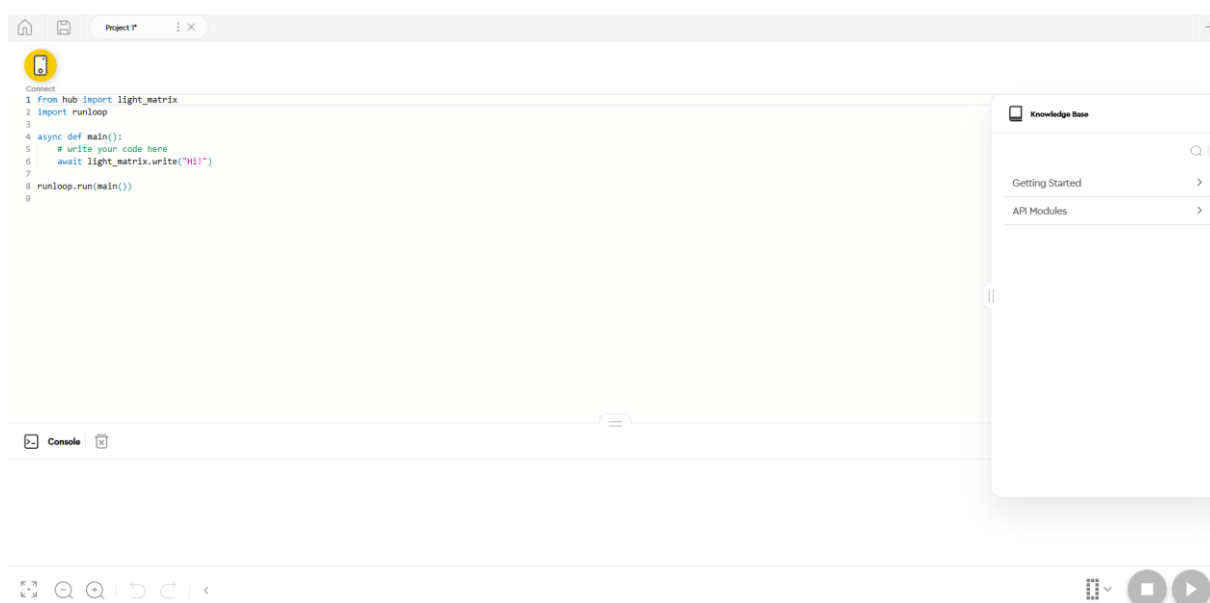


Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

7. Kliknij na środku przycisk **New Project**. Pojawi się nowe okno jak poniżej



8. Wybierz opcję **Python** oraz podaj nazwę projektu. Kiedy wszystko zostanie poprawnie wybrane, wciśnij przycisk **Create**.

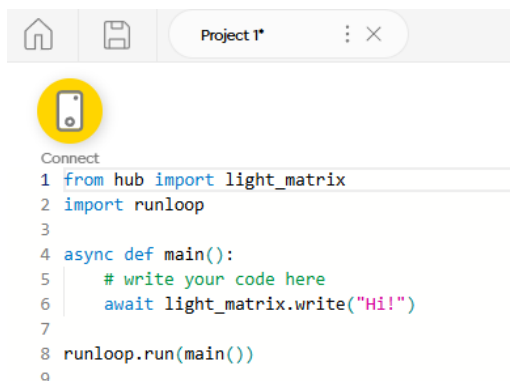


Wszystko jest gotowe do programowania. Pora na podłączenie kontrolera Spike Hub.

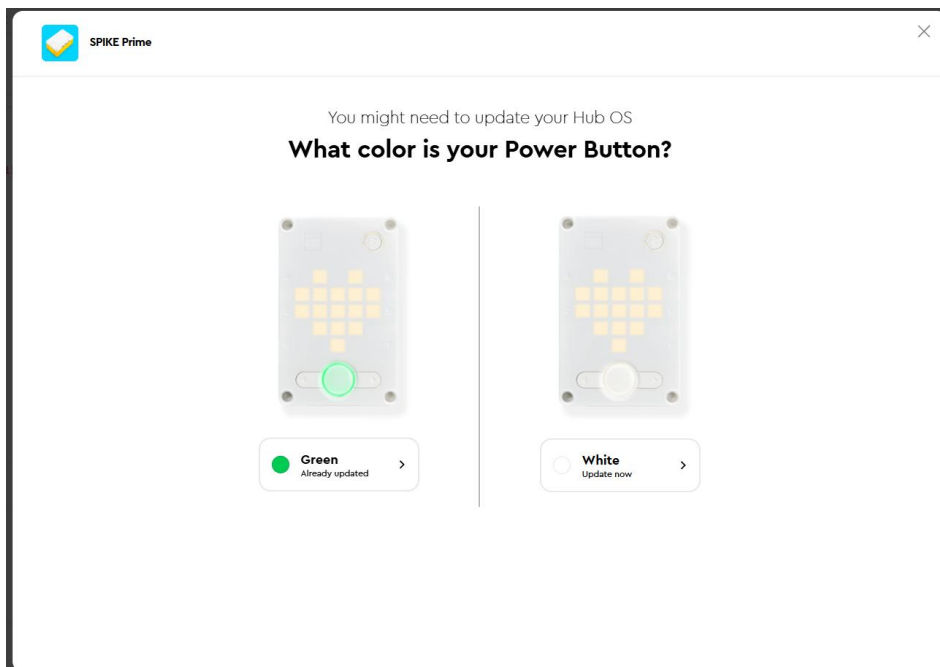
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 4. Podłączenie kontrolera Spike Hub

1. W głównym programie kliknij przycisk **Connect** znajdujący się w górnym rogu, tuż pod nazwą twojego projektu. Patrz poniżej.

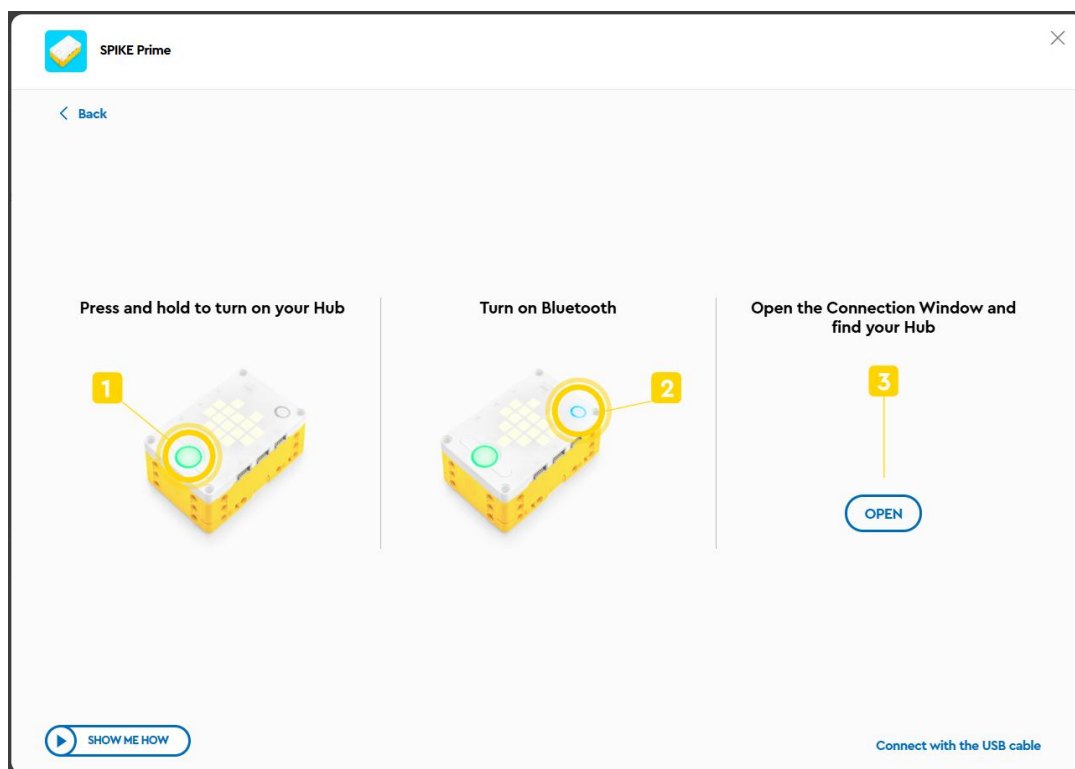


2. Pojawi się nowe okno. **Włącz kontroler Spike Hub** klikając **duży biały przycisk** i poczekaj aż się uruchomi. Następnie wybierz jedną z opcji odpowiadającą stanowi kontrolera Spike Hub – zawsze będzie to **Green – Already updated**.

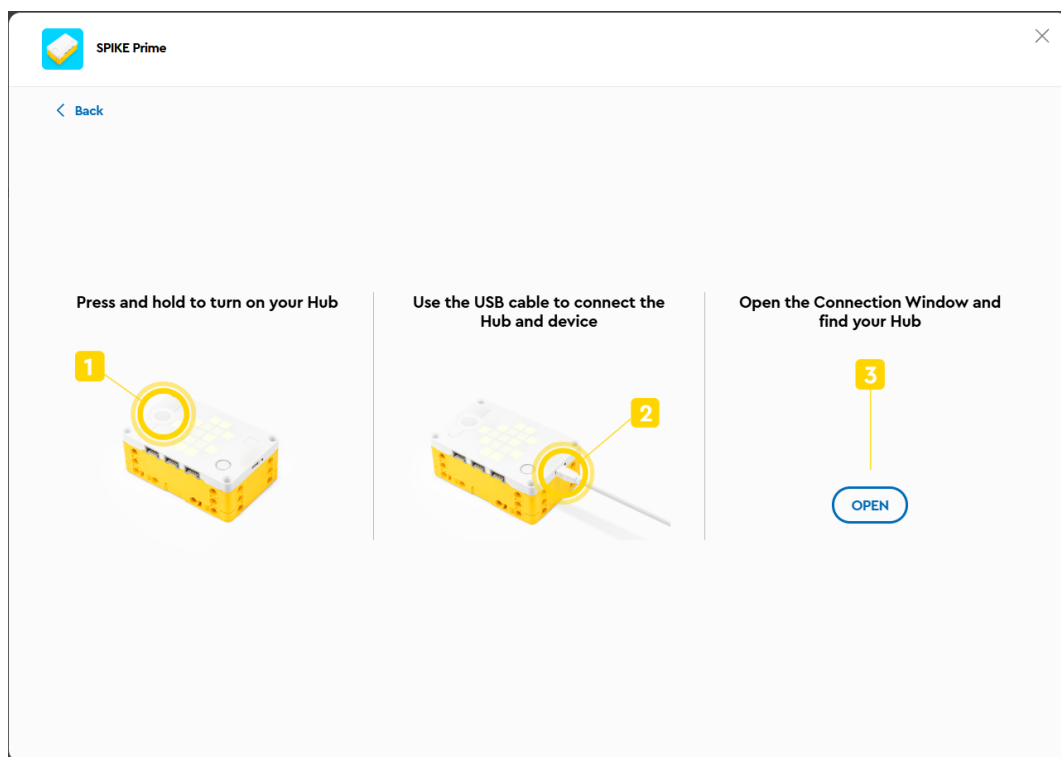


3. Pojawi się nowe okno (patrz poniżej). Wybierz opcję **Connect with USB cable** znajdującą się w prawym dolnym rogu okna.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

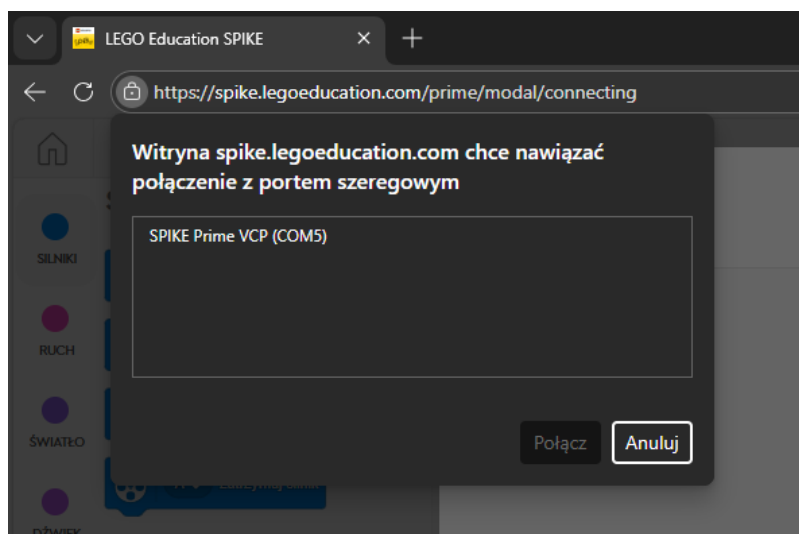


4. Postępuj zgodnie z instrukcją na ekranie. Po przeczytaniu kliknij przycisk **Open**.



5. Pojawi się nowe okno (patrz poniżej)

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



6. Wybierz opcję **SPIKE Prime VCP** i kliknij **Połącz**.

UWAGA – u Ciebie nazwa może nie być identyczna jak na rysunku powyżej. Wybierz tę opcję, która ma w nazwie Spike Prime. Innymi rzeczami się nie przejmuj.

Jeśli wszystko zostanie poprawnie połączone, to okna zostaną automatycznie zamknięte, a ikona kontrolera w programie zmieni się na zieloną.



```
1 from hub import light_matrix
2 import runloop
3
4 async def main():
5     # write your code here
6     await light_matrix.write("Hi!")
7
8 runloop.run(main())
9
```

Gratulacje. Twój Spike Hub jest podłączony do komputera. Możesz przystąpić do programowania robota.

W przypadku problemów postępuj wg kroków:

1. Sprawdź, czy kontroler Spike Hub jest włączony i świeci na zielono
 - a. Jeśli kontroler świeci na biało zgłoś to nauczycielowi
 - b. Jeśli kontroler świeci na zielono to podążaj dalej wg instrukcji łączenia powyżej
2. Sprawdź czy kabel USB jest podłączony (Spike Hub <-> Komputer)
3. Sprawdź, czy wybrano odpowiednią nazwę urządzenia – w oknie połączenia z portem szeregowym należy wybrać urządzenie, które w nazwie ma SPIKE Prime.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 5: Odkrywamy tajemnice Kodu... Odczytać i zrozumieć Matrix

Temat lekcji nr 23 - Przenoszenie ładunków – analiza kodu pod kątem wykorzystania funkcji

Treści nauczania (wymagania szczegółowe - podstawa programowa) – I. 5); IV. 4) II. 3) b); II. 3) c); I. 1);

Cel lekcji:

- **Wprowadzenie do programowania ruchu robota** – uczniowie poznają podstawowe zasady sterowania ruchem robota w języku Python oraz uczą się korzystania z gotowych bibliotek kodu w LEGO SPIKE Prime, a także tworzenia funkcji usprawniających zapis programu.
- **Analiza działania programu** – uczniowie analizują fragmenty kodu odpowiedzialne za ruch robota oraz działanie funkcji, określając, w jaki sposób poszczególne polecenia wpływają na kierunek i sposób poruszania się robota.
- **Tworzenie prostych algorytmów i programów sterujących ruchem** – uczniowie opracowują algorytmy oraz zapisują je w postaci programów tekstowych wykorzystujących podstawowe polecenia sterujące ruchem robota, biblioteki kodu oraz funkcje porządkujące wykonywanie powtarzalnych czynności.
- **Testowanie działania programu** – uczniowie uruchamiają program, obserwują sposób poruszania się robota oraz wprowadzają poprawki w celu uzyskania poprawnego działania programu, precyzyjnego ruchu robota oraz prawidłowego działania utworzonych funkcji.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- **Uczeń dostrzega, że działanie programu i robota zależy od poprawności zapisanych poleceń:**
 - rozumie, że robot wykonuje instrukcje zgodnie z zapisanym kodem programu oraz poprawnie utworzonymi i wywoływanymi funkcjami,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- obserwuje działanie programu sterującego ruchem robota i formułuje wnioski dotyczące poprawności działania poleceń oraz zastosowanych funkcji.
- **Uczeń analizuje działanie programu i jego efekty:**
 - sprawdza poprawność działania programu sterującego ruchem robota w języku Python,
 - identyfikuje błędy składniowe i logiczne oraz modyfikuje program poprzez zmianę poleceń, ich kolejności lub poprawę działania funkcji,
 - korzysta z gotowych bibliotek kodu umożliwiających sterowanie robotem,
 - tworzy funkcje usprawniające i porządkujące kod źródłowy programu.
- **Uczeń rozwija umiejętność testowania i doskonalenia rozwiązań:**
 - wykonuje próby działania programu i analizuje sposób poruszania się robota,
 - wprowadza poprawki w celu uzyskania poprawnego działania robota i utworzonych funkcji,
- testuje różne rozwiązania związane ze sterowaniem ruchem robota, wykorzystaniem bibliotek kodu oraz modyfikowaniem funkcji w programie.

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści kształcenia będą realizowane za pomocą robotyki edukacyjnej LEGO® Education SPIKE™ Prime.

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna RaP STEAM,
- praca na komputerze,
- praca przy zestawie robotyki edukacyjnej,
- pogadanka.

Środki dydaktyczne

- zestaw PC z dostępem do Internetu,
- zestaw robotyki edukacyjnej LEGO® Education SPIKE™ Prime oraz zestaw rozszerzony LEGO® Education SPIKE™ Prime Expansion Kit,
- rzutnik multimedialny,
- e-podręcznik.

Formy pracy

- Praca indywidualna lub grupowa.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przebieg lekcji:

Faza I. Wprowadzenie

Na początku lekcji nauczyciel wprowadza uczniów w zagadnienia związane z programowaniem ruchu robota w języku Python, wyjaśniając, że robot wykonuje polecenia zapisane w programie w określonej kolejności oraz że kod można usprawnić dzięki **wykorzystaniu funkcji**. Następnie odnosi się do zadania lekcyjnego, polegającego na wykorzystaniu gotowych bibliotek kodu oraz **tworzeniu funkcji porządkujących kod źródłowy w środowisku LEGO SPIKE Prime**.

Nauczyciel przedstawia cel i zakres zadania. Prezentuje uczniom środowisko programowania w języku Python w aplikacji LEGO SPIKE App oraz omawia podstawowe elementy bibliotek wykorzystywanych do sterowania robotem, takie jak polecenia odpowiedzialne za ruch, zatrzymanie oraz zmianę kierunku jazdy. Następnie przekazuje uczniom informacje dotyczące **tworzenia i wywoływania funkcji** usprawniających zapis programu.

Uczniowie zapoznają się ze sposobami zapisu poleceń sterujących ruchem robota w języku Python oraz obserwują, jak robot po uruchomieniu programu wykonuje kolejne instrukcje i porusza się zgodnie z zapisanym kodem oraz **utworzonymi funkcjami**.

Następnie uczniowie zdobywają umiejętności obejmujące:

- określenie problemu (w jaki sposób wykorzystać funkcje do usprawnienia sterowania ruchem robota w języku Python),
- analizę działania programu sterującego ruchem robota,
- wykorzystanie gotowych bibliotek kodu do sterowania robotem,
- tworzenie prostych funkcji realizujących wybrane czynności robota,
- obserwację działania robota po uruchomieniu programu,
- testowanie programu i wprowadzanie prostych poprawek związanych ze sterowaniem ruchem robota oraz działaniem funkcji.

Na zakończenie tej fazy lekcji nauczyciel podaje temat do zapisania w zeszycie i przechodzi do fazy II – realizacji praktycznej.

Faza II. Część zasadnicza - realizacja

W tej fazie lekcji nauczyciel zapewnia uczniom praktyczne ćwiczenia, które pozwolą eksperymentować z koncepcjami programowania. W szczególności obejmować to będzie używanie blokowych języków programowania oraz zestawu do robotyki.

1. Nauczyciel objaśnia sposób pracy na lekcji – we wskazanym przez niego momencie uczniowie będą wykonywać kolejne etapy prowadzące do rozwiązania zadania. Zdolniejsi uczniowie spróbują wykonać je samodzielnie. Słabsi wykonają kolejne kroki

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

razem z nauczycielem, w oparciu o przygotowaną do zajęć instrukcję dostępną w e-podręczniku lub materiał multimedialny. W razie potrzeby będą mogli dopytać nauczyciela. Nauczyciel na bieżąco sprawdza postępy uczniów.

2. Bazując na posiadanych zasobach, zapewnij każdemu uczniowi lub zespołowi uczniów kompletny zestaw do robotyki (tj. LEGO® Education SPIKE™).
3. Zestawy te pozwalają uczniom budować własne roboty i programować je do wykonywania określonych zadań. Nauczyciel zaczyna od prostych projektów, takich jak zbudowanie i zaprogramowanie robota, który może wykonać proste czynności, które wprowadzą robota w ruch.
4. Uczniowie budują bazowego robota według załączonej instrukcji (załącznik **Budowa_bazy_robota.pdf** do lekcji).
5. Uczniowie włączają program **SPIKE**, poprzez który będą programować robota.
6. Nauczyciel, wykorzystując e-platformę, demonstruje na tablicy multimedialnej działanie programu i robota, uruchamiając odpowiedni moduł i materiał multimedialny.
7. Prezentowany materiał multimedialny, zrealizowany w oparciu o szczegółowy scenariusz (załącznik do lekcji), oprócz waloru teoretycznego stanowi również podstawę przebiegu części praktycznej lekcji. W szczególności, uczniowie:
 - a. budują prosty model bazowego robota w oparciu o zestaw do robotyki (tj. LEGO® Education SPIKE™),
 - b. tworzą listę czynności niezbędnych do wprowadzenia robota w ruch,
 - c. na podstawie listy czynności przygotowują program, zapisują go we wskazanym folderze (nauczyciel wskazuje właściwą lokalizację; może to być dysk lokalny lub zasób w chmurze) i potrafią go uruchomić;
 - d. wgrywają program do zbudowanego modelu robota, uruchamiają, obserwują i testują działanie robota. Wprowadzają modyfikacje w programie, poprawiają ewentualne błędy.

Faza III. Podsumowanie

Uczniowie analizują przedstawione przykłady programów sterujących ruchem robota zapisanych w języku Python oraz omawiają ich działanie. Nauczyciel ocenia stopień zrozumienia zależności między zapisanym kodem, **utworzonymi funkcjami**, a sposobem poruszania się robota po uruchomieniu programu.

Kryteria oceniania:

Ocena dopuszczająca:

- uczeń rozpoznaje podstawowe elementy programu w języku Python,
- potrafi wskazać polecenia odpowiedzialne za ruch robota,
- rozpoznaje proste funkcje zapisane w programie.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Ocena dostateczna:

- uczeń analizuje przykładowy program i opisuje jego działanie,
- rozumie zależność między zapisanym kodem, funkcjami, a działaniem programu,
- rozpoznaje podstawowe polecenia wykorzystywane do sterowania robotem,
- wyjaśnia rolę funkcji usprawniających zapis programu.

Ocena dobra:

- uczeń samodzielnie modyfikuje przykładowy program,
- poprawia błędy związane z zapisem poleceń, nawiasami lub cudzysłowami,
- testuje działanie programu sterującego ruchem robota po wprowadzonych zmianach,
- tworzy własne funkcje usprawniające wykonywanie powtarzalnych czynności robota.

Ocena bardzo dobra:

- uczeń samodzielnie tworzy poprawny program w języku Python,
- wyjaśnia działanie poszczególnych elementów programu oraz utworzonych funkcji,
- wykorzystuje biblioteki kodu do sterowania ruchem robota,
- proponuje własne rozwiązania i potrafi je uzasadnić,
- samodzielnie analizuje i poprawia błędy w kodzie,
- stosuje funkcje w celu uporządkowania i optymalizacji kodu źródłowego programu.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, np.:

1. Jaki jest główny cel wykorzystania funkcji w programie robota?
 - a. **Uporządkowanie i uproszczenie kodu**
 - b. Przyspieszenie robota
 - c. Zmiana koloru robota
 - d. Zatrzymanie programu
2. Co oznacza parametr w funkcji (np. stopnie)?
 - a. Kolor robota
 - b. Typ silnika
 - c. **Wartość przekazywana do funkcji**
 - d. Czas programu
3. Dlaczego w funkcjach sterujących ruchem robota używamy **await**?
 - a. **Aby program czekał na wykonanie ruchu**
 - b. Aby przyspieszyć robota
 - c. Aby zmienić kierunek
 - d. Aby wyświetlić tekst

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

4. Która część programu najczęściej trafia do funkcji?
 - a. Kolor robota
 - b. Nazwy zmiennych
 - c. Komentarze
 - d. **Powtarzalne działania (np. ruch robota)**
5. Co daje użycie funkcji z parametrem w programie robota?
 - a. **Możliwość wielokrotnego użycia funkcji z różnymi wartościami**
 - b. Zmianę koloru robota
 - c. Zatrzymanie programu
 - d. Usunięcie błędów

Faza V. Dodatkowe, praca domowa (dla chętnych).

Zadania dodatkowe (dla chętnych).

Zadanie 1: Transport ładunku według instrukcji

Napisz program w Pythonie, w którym robot przewozi „ładunek” z punktu startowego do wyznaczonego miejsca. Kod powinien być podzielony na funkcje opisujące kolejne etapy przejazdu.

Program powinien:

- zawierać funkcję odpowiedzialną za start robota,
- zawierać funkcję do jazdy z ładunkiem,
- zawierać funkcję do zatrzymania się w miejscu rozładunku,
- wykorzystywać co najmniej trzy funkcje,
- unikać wielokrotnego powtarzania tych samych instrukcji,
- być czytelny i poprawny składniowo.

Opcjonalnie dodaj funkcję **dostarcz_ładunek()**, która połączy kilka mniejszych funkcji w jedną trasę przejazdu.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 23 - *Przenoszenie ładunków – analiza kodu pod kątem wykorzystania funkcji.*

Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
 - Nauczyciel omawia podstawowe zagadnienia związane z programowaniem ruchu robota przewożącego ładunek w języku Python:
 - wyjaśnia, że robot wykonuje instrukcje zapisane w programie w określonej kolejności,
 - przedstawia przykłady poleceń wykorzystywanych do sterowania ruchem robota,
 - omawia znaczenie poprawnej składni programu, wykorzystania bibliotek kodu oraz funkcji usprawniających kod źródłowy.
 - Wprowadzenie do zadania i przypomnienie podstaw pracy w LEGO SPIKE Prime:
 - omówienie środowiska programowania Python w aplikacji LEGO SPIKE App,
 - przedstawienie podstawowych bibliotek wykorzystywanych do sterowania robotem,
 - przedstawienie zależności: zapis kodu >> **utworzenie funkcji** >> uruchomienie programu >> ruch robota,
 - analiza przykładowych programów odpowiedzialnych za ruch robota oraz zastosowanie funkcji w programie.
 - Praca uczniów:
 - analiza przedstawionych przykładów programów sterujących ruchem robota,
 - omawianie działania poszczególnych linii kodu oraz utworzonych funkcji,
 - testowanie działania robota po uruchomieniu programu,
 - wyszukiwanie i poprawianie błędów w kodzie oraz funkcjach,
 - proponowanie własnych rozwiązań związanych ze sterowaniem ruchem robota oraz **tworzenie funkcji usprawniających wykonywanie powtarzalnych ruchów robota.**
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów (analiza poprawności działania programu, sposobu poruszania się robota oraz umiejętności **wykorzystania bibliotek kodu i funkcji**).
- Zadanie pracy domowej (opcjonalnie) – napisanie prostego programu sterującego ruchem robota **z wykorzystaniem funkcji** oraz przetestowanie działania programu po wprowadzeniu własnych zmian.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 2 – Realizacja zadania głównego podczas lekcji nr 23 - *Przenoszenie ładunków – analiza kodu pod kątem wykorzystania funkcji.*

Zadanie 1. Przygotowanie do realizacji zadania głównego

Uczniowie, pracując w zespołach lub indywidualnie, realizują zadanie polegające na poznaniu sposobu korzystania z bibliotek kodu w języku Python, programowaniu podstawowych ruchów robota oraz **usprawnianiu kodu źródłowego za pomocą funkcji** w środowisku LEGO SPIKE Prime. Szczególny nacisk położony jest na poprawne wykorzystanie gotowych bibliotek, rozumienie działania importowanych modułów oraz **tworzenie funkcji porządkujących program** sterujący ruchem **robota oraz robota przewożącego ładunek**.

Uczniowie opracowują i zapisują rozwiązanie problemu w postaci prostych programów sterujących ruchem **robota przewożącego ładunek** oraz wyświetlaniem komunikatów na kontrolerze SPIKE Hub. W tym celu:

- analizują przykładowe programy zapisane w języku Python wykorzystujące biblioteki kodu i funkcje,
- określają cel programu oraz przewidują możliwe błędy składniowe i logiczne,
- planują kolejność wykonywania instrukcji sterujących robotem,
- opracowują algorytm w uporządkowanej formie,
- tworzą funkcje odpowiedzialne za wybrane ruchy robota lub powtarzalne czynności,
- zapisują rozwiązanie jako program w języku Python z wykorzystaniem bibliotek i modułów,
- testują działanie programu i obserwują sposób poruszania się robota,
- wprowadzają poprawki związane ze sterowaniem ruchem, działaniem funkcji lub poprawnością składni kodu,
- doprecyzowują rozwiązanie tak, aby program był czytelny, uporządkowany i skuteczny w działaniu.

Uczniowie wykorzystują środowisko LEGO SPIKE Prime w trybie Python, zwracając uwagę na zgodność między zapisanym kodem, utworzonymi funkcjami a rzeczywistym działaniem robota. Szczególną uwagę poświęcają wykorzystaniu bibliotek i modułów, poprawnemu zapisowi poleceń oraz analizie komunikatów o błędach pojawiających się po uruchomieniu programu.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 2. Wprowadzenie teoretyczne do lekcji

Pora na funkcje!

W dzisiejszej lekcji skupisz się na opracowaniu funkcji, które ułatwią pracę przy programowaniu robota. Część programu zostanie wydzielona w blokach funkcyjnych, aby można było podać tylko ich nazwę. Zastosujesz pseudokod oraz przygotujesz blok funkcyjny. No to do dzieła.

Przykłady realizacji funkcji

Rozwiąż zadania, aby zrozumieć ideę konwersji kodu bez funkcji na kod z wykorzystaniem funkcji. Jeśli chcesz, możesz wgrać zadania do robota.

Zadanie 1: Patrol po kwadracie

Robot musi przejechać trasę w kształcie kwadratu. Każdy bok to 500 stopni obrotu kół, a po każdym boku następuje obrót o 90 stopni w miejscu. Na koniec robot zatrzymuje się.

Opis punktowy zadania:

- Przejdź do przodu o 500 stopni,
- Obróć się w prawo o 135 stopni (obróć o 90 stopni robota),
- Przejdź do przodu o 500 stopni,
- Obróć się w prawo o 135 stopni,
- Przejdź do przodu o 500 stopni,
- Obróć się w prawo o 135 stopni,
- Przejdź do przodu o 500 stopni,
- Zatrzymaj silniki.

Jak wygląda wejściowy kod realizujący zadanie? Oto on (tylko funkcja **main()**):

```
motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)

# Bok 1 i skręt
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 500, 0)
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 135, 100)

# Bok 2 i skręt
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 500, 0)
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 135, 100)
```

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Bok 3 i skręt

```
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 500, 0)
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 135, 100)
```

Bok 4

```
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 500, 0)
```

```
motor_pair.stop(motor_pair.PAIR_1)
```

Twoje zadanie:

Zaproponuj program (kod), który zrealizuje to zadanie z wykorzystaniem funkcji. Możesz korzystać z parametrów jednak nie jest to konieczne.

Zadanie 2: Trzystopniowa dostawa towaru

Robot dostarcza przesyłki do trzech punktów rozładunkowych znajdujących się w różnej odległości od startu. W każdym punkcie robot musi zasygnalizować dotarcie dźwiękiem.

Opis punktowy zadania:

- Podjedź do punktu A (300 stopni),
- Wydadź dźwięk (beeeep),
- Podjedź do punktu B (kolejne 300 stopni),
- Wydadź dźwięk (beeeep),
- Podjedź do punktu C (ostatnie 300 stopni),
- Wydadź dźwięk (beeeep),
- Cofnij się na linię startu (900 stopni).

Jak wygląda wejściowy kod realizujący zadanie? Oto on (tylko funkcja **main()**):

```
motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
```

Punkt A

```
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 300, 0)
await sound.beep(440, 500)
```

Punkt B

```
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 300, 0)
await sound.beep(440, 500)
```

Punkt C

```
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 300, 0)
await sound.beep(440, 500)
```


Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Powrót

```
await motor_pair.move_for_degrees(motor_pair.PAIR_1, -900, 0)
motor_pair.stop(motor_pair.PAIR_1)
```

Twoje zadanie:

Zaproponuj program (kod), który zrealizuje to zadanie z wykorzystaniem funkcji. Możesz korzystać z parametrów, jednak nie jest to konieczne.

Zadanie 3: Przegląd techniczny ramienia*

Zanim robot wyjedzie na trasę, musi sprawdzić sprawność swojego manipulatora. Wykonuję serię ruchów testowych ramieniem na różnych wysokościach.

Opis punktowy zadania:

- Podnieś ramię o 90 stopni,
- Opuść ramię o 90 stopni,
- Przejdź krótki dystans testowy (100 stopni),
- Podnieś ramię wysoko (180 stopni),
- Opuść ramię (180 stopni),
- Przejdź krótki dystans testowy (100 stopni),
- Podnieś ramię lekko (45 stopni),
- Opuść ramię (45 stopni).

Jak wygląda wejściowy kod realizujący zadanie? Oto on (tylko funkcja **main()**):

```
motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
```

Test 1

```
await motor.run_for_degrees(port.D, 90, 360)
await motor.run_for_degrees(port.D, -90, 360)
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 100, 0)
```

Test 2

```
await motor.run_for_degrees(port.D, 180, 360)
await motor.run_for_degrees(port.D, -180, 360)
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 100, 0)
```

Test 3

```
await motor.run_for_degrees(port.D, 45, 360)
await motor.run_for_degrees(port.D, -45, 360)
```

```
motor_pair.stop(motor_pair.PAIR_1)
```

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Twoje zadanie:

Zaproponuj program (kod), który zrealizuje to zadanie z wykorzystaniem funkcji. Możesz korzystać z parametrów, jednak nie jest to konieczne.

W razie problemów, materiały pomocnicze znajdziesz w **Dodatku A – rozwiązania zadań** (ostatnia strona lekcji online).

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 3. Realizacja zadania głównego

Dzisiejsza lekcja jest rozwinięciem lekcji 21. Wróćmy do niej. Oto zadanie, które było do zrealizowania w lekcji 21:

- Opuść silnik dodatkowy (manipulator) aby robot po podjechaniu do wydruku 3D (palety) mógł ją swobodnie podnieść,
- Przejdź robotem do przodu, aby nałożyć paletę na chwytak (manipulator),
- Podnieś manipulator, aby paleta nie miała kontaktu z podłożem,
- Podjedź do przodu, aby utrzymać się założonej trasy (w przypadku przykładu jazdy następuje po czarnej linii bez jej wykrywania),
- Wykonaj obrót o ok. 90 stopni w prawo. Robot powinien obrócić się w miejscu,
- Jedź do przodu przez 4,5 obroty (ok 1337 stopni), bez skręcania. Robot dojeżdża do celu (punkt B),
- Opuść silnik dodatkowy (manipulator) aby postawić paletę na podłożu,
- Cofnij się o ok. 50 stopni, aby zsunąć paletę z manipulatora,
- Zatrzymaj robota.

Oto kod tego realizujący te zadania:

```
from hub import light_matrix, port
import runloop
import motor_pair, motor
```

```
async def main():
```

```
    # write your code here
    await light_matrix.write("Hi!")
```

```
    #przypisanie portów A i B do PAIR_1
    motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
```

```
    #opuszczenie silnika dodatkowego aby łatwiej podnieść przedmiot o 300
    milisekund (0,3 sekundy) na porcie D z prędkością -360 stopni na sekundę
```

```
    #użycie funkcji run for time powoduje opuszczenie chwytaka bez jego
    zablokowania. Po 300 milisekundach program przejdzie dalej.
```

```
    #w przypadku użycia funkcji run for degrees zły obrót mógłby zablokować
    cały program
```

```
    await motor.run_for_time(port.D, 300, -360)
```

```
    #jazda do przodu bez skręcania (0) przez ponad 1 obrót (450)
```

```
    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 450, 0)
```

```
    #podniesienie przedmiotu chwytakiem. Obróć silnik o 50 stopni (w górę) z
    prędkością 360 podłączonym do portu D
```

```
    #wykorzystanie słowa await wstrzyma program do czasu wykonania ruchu - da
    to pewność podniesienia
```



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

```
await motor.run_for_degrees(port.D, 50, 360)

#jazda do przodu bez skręcania (0) przez 200 stopni
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 200, 0)

#skręcanie w prawo przez 130 stopni z pełnym skrętem 100 ze standardową
prędkością obrotu (wartość domyślna to 360 więc nie wpisujemy)
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 130, 100)

#jazda do przodu bez skręcania (0) przez 1337 stopni ~4,5 obrotów * 360
daje ~1337 stopni.
await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1337, 0)

#opuszczenie przedmiotu chwytakiem. Obróć silnik o -50 stopni (w dół) z
prędkością 360 podłączonym do portu D
#wykorzystanie słowa await nie jest potrzebne ponieważ nie występuje
dalszy ruch tym silnikiem w programie
motor.run_for_degrees(port.D, -50, 360)

#jazda do tyłu bez skręcania (0) przez 360 stopni aby odjechać od skrzyni
await motor_pair.move_for_degrees(motor_pair.PAIR_1, -360, 0)

#zatrzymanie silników robota
motor_pair.stop(motor_pair.PAIR_1)

runloop.run(main())
```

Pora przerobić go na funkcje. Jaka część kodu jest powtarzalna? Co warto byłoby przerzucić do funkcji? Zastanów się chwilę.



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Przykład realizacji zadania głównego

Identyfikacja funkcji sterujących

Analizując kod można zauważyć pewne powtarzalne ruchy:

- Ruch robota
- Obrót robota
- Ruch ramienia

Odpowiednie bloki kodu można przerobić na funkcje. Pozwoli to na łatwiejsze zarządzanie kodem. Czy będzie on krótszy? Niekoniecznie. Czy będzie on czytelniejszy? Na pewno. Czy możliwość poprawiania i edycji będzie dużo łatwiejsza? Jak najbardziej. Dobry kod to dobry program. Warto ułatwiać sobie życie. No to po kolei.

Funkcja 1 - Funkcja ruchu robota

Jak zrealizować ruch robota w funkcji? Zdefiniuj funkcję **jazdaRobota()** z parametrem dystans. Podczas wywołania możliwe będzie podanie, ile stopni ma przejechać robot. Daje to dużą uniwersalność funkcji. Oto przykład:

```
async def jazdaRobota(stopnie):
```

Pamiętaj o słowie **async**!

Funkcja 2 - Funkcja obrotu robota

Wspaniale, teraz zajmij się funkcją obrotu robota. Zdefiniuj funkcję **obrotRobota()** bez parametrów, który wykona obrót wg zadania:

```
async def obrotRobota():
```

Funkcja 3 - Funkcja ruchu ramienia

Pora na ostatnią funkcję – ruch ramienia. Zdefiniuj funkcję **ruchRamie()** z parametrem stopnie. Funkcja powinna odpowiadać za ruch ramieniem. Przykład:

```
def ruchRamie(stopnie):
```

Ekstra. Udało się zdefiniować funkcje. Na bazie doświadczenia spróbuj dodać odpowiednie komendy sterujące silnikami. Pomyśl, jak zrealizować swoje zadanie i do następnej lekcji!

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Dodatek A – Rozwiązania zadań

Rozwiązanie zadania 1:

```
async def bok_i_skret():
    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 500, 0)
    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 135, 100)

async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)

    await bok_i_skret()
    await bok_i_skret()
    await bok_i_skret()
    # Ostatni bok bez skrętu, więc używamy samej komendy jazdy
    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 500, 0)

    motor_pair.stop(motor_pair.PAIR_1)
```

Rozwiązanie zadania 2:

```
async def punkt_dostawy():
    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 300, 0)
    await sound.beep(440, 500)

async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)

    await punkt_dostawy() # Punkt A
    await punkt_dostawy() # Punkt B
    await punkt_dostawy() # Punkt C

    await motor_pair.move_for_degrees(motor_pair.PAIR_1, -900, 0)
    motor_pair.stop(motor_pair.PAIR_1)
```

Rozwiązanie zadania 3:

```
async def test_ramienia(stopnie):
    await motor.run_for_degrees(port.D, stopnie, 360)
    await motor.run_for_degrees(port.D, -stopnie, 360)

async def jazda_testowa():
    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 100, 0)

async def main():
    motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
```




Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

```
await test_ramienia(90)
await jazda_testowa()

await test_ramienia(180)
await jazda_testowa()

await test_ramienia(45)

motor_pair.stop(motor_pair.PAIR_1)
```



Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 5: Odkrywamy tajemnice Kodu... Odczytać i zrozumieć Matrix

Temat lekcji nr 24 - Przenoszenie ładunków - realizacja zadania z użyciem funkcji.

Treści nauczania (wymagania szczegółowe - podstawa programowa) – I. 2) a); I. 2) b); I. 4); II. 1); II. 2); IV. 1)

Cel lekcji:

- **Wprowadzenie do programowania funkcji w języku Python** – uczniowie poznają podstawowe zasady korzystania z bibliotek i modułów odpowiedzialnych za sterowanie ruchem robota oraz tworzenie funkcji wykorzystywanych podczas przenoszenia ładunków w LEGO SPIKE Prime.
- **Analiza działania programu** – uczniowie analizują fragmenty kodu odpowiedzialne za sterowanie ruchem robota oraz funkcjami realizującymi przenoszenie ładunku i określają, w jaki sposób poszczególne polecenia wpływają na działanie programu i robota.
- **Tworzenie algorytmów i programów wykorzystujących funkcje** – uczniowie opracowują algorytmy oraz zapisują je w postaci programów tekstowych umożliwiających sterowanie ruchem robota, transportem ładunku oraz wykonywaniem powtarzalnych czynności z użyciem funkcji.
- **Testowanie działania programu** – uczniowie uruchamiają program, obserwują działanie robota podczas realizacji zadania polegającego na przenoszeniu ładunków z wykorzystaniem funkcji, analizują poprawność działania programu, a następnie wprowadzają poprawki w celu uzyskania poprawnego działania robota i zapisanych funkcji.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- **Uczeń dostrzega, że działanie programu i robota zależy od poprawności zapisanych poleceń:**

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- rozumie, że robot wykonuje instrukcje zgodnie z zapisanym kodem programu oraz poprawnym sterowaniem mechanizmem odpowiedzialnym za przenoszenie ładunków,
 - obserwuje działanie programu napisanego w języku Python i analizuje wpływ poleceń na ruch robota oraz transport i przenoszenie ładunku z wykorzystaniem funkcji.
- **Uczeń analizuje działanie programu i jego efekty:**
 - sprawdza poprawność działania programu sterującego robotem oraz funkcjami odpowiedzialnymi za przenoszenie ładunku w języku Python,
 - identyfikuje błędy składniowe i logiczne oraz modyfikuje program poprzez poprawę zapisu poleceń lub zmianę parametrów ruchu robota i mechanizmu transportującego,
 - wykorzystuje biblioteki i moduły służące do obsługi silników LEGO SPIKE Prime (motor_pair, motor) oraz tworzy i wykorzystuje funkcje upraszczające zapis programu.
- **Uczeń rozwija umiejętność testowania i doskonalenia rozwiązań:**
 - wykonuje próby działania programu i analizuje sposób działania robota podczas realizacji zadania polegającego na przenoszeniu ładunków,
 - wprowadza poprawki w celu uzyskania poprawnego działania programu i właściwej synchronizacji ruchu robota z działaniem funkcji sterujących transportem ładunku,
 - testuje działanie różnych parametrów ruchu i porównuje uzyskiwane efekty działania robota oraz skuteczność wykonanych funkcji programistycznych.

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści kształcenia będą realizowane za pomocą robotyki edukacyjnej LEGO® Education SPIKE™ Prime.

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna RaP STEAM,
- praca na komputerze,
- praca przy zestawie robotyki edukacyjnej,
- pogadanka.

Środki dydaktyczne

- zestaw PC z dostępem do Internetu,
- zestaw robotyki edukacyjnej LEGO® Education SPIKE™ Prime oraz zestaw rozszerzony LEGO® Education SPIKE™ Prime Expansion Kit,
- rzutnik multimedialny,
- e-podręcznik.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Formy pracy

- Praca indywidualna lub grupowa.

Przebieg lekcji:

Faza I. Wprowadzenie

Na początku lekcji nauczyciel wprowadza uczniów w zagadnienia związane z programowaniem funkcji w języku Python, wyjaśniając, że **robot może wykonywać złożone zadania transportowe dzięki odpowiedniemu sterowaniu ruchem robota oraz zastosowaniu funkcji odpowiedzialnych za przenoszenie ładunków**. Następnie odnosi się do zadania lekcyjnego, polegającego na realizacji zadania transportowego z użyciem funkcji w LEGO SPIKE Prime.

Nauczyciel przedstawia cel i zakres zadania. Prezentuje uczniom środowisko programowania w języku Python w aplikacji LEGO SPIKE App oraz omawia sposób korzystania z bibliotek i modułów odpowiedzialnych za sterowanie silnikami. Następnie przekazuje uczniom niezbędne informacje dotyczące wykorzystania bibliotek **motor_pair** oraz **motor** do sterowania ruchem robota oraz tworzenia własnych funkcji usprawniających zapis programu.

Uczniowie zapoznają się ze sposobami zapisu poleceń służących do sterowania ruchem robota oraz **tworzenia i wywoływania funkcji odpowiedzialnych za wykonywanie określonych czynności związanych z transportem ładunku i obserwują, jak program po uruchomieniu realizuje kolejne etapy zadania transportowego**.

Następnie uczniowie zdobywają umiejętności obejmujące:

- określenie problemu (w jaki sposób wykorzystać funkcje do sterowania robotem podczas przenoszenia ładunków),
- analizę działania programu sterującego robotem oraz funkcji zapisanych w języku Python,
- wykorzystanie bibliotek i modułów służących do sterowania silnikami,
- tworzenie i stosowanie funkcji upraszczających wykonywanie powtarzalnych czynności,
- obserwację wyników działania programu po jego uruchomieniu,
- testowanie programu i analizę poprawności działania robota podczas realizacji zadania transportowego,
- wprowadzanie prostych poprawek związanych z działaniem programu, funkcji oraz synchronizacją ruchów robota.

Na zakończenie tej fazy lekcji nauczyciel podaje temat do zapamiętania w zeszycie i przechodzi do fazy II – realizacji praktycznej.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Faza II. Część zasadnicza - realizacja

W tej fazie lekcji nauczyciel zapewnia uczniom praktyczne ćwiczenia, które pozwolą eksperymentować z koncepcjami programowania. W szczególności obejmować to będzie używanie blokowych języków programowania oraz zestawu do robotyki.

1. Nauczyciel objaśnia sposób pracy na lekcji – we wskazanym przez niego momencie uczniowie będą wykonywać kolejne etapy prowadzące do rozwiązania zadania. Zdolniejsi uczniowie spróbują wykonać je samodzielnie. Słabsi wykonają kolejne kroki razem z nauczycielem, w oparciu o przygotowaną do zajęć instrukcję dostępną w e-podręczniku lub materiał multimedialny. W razie potrzeby będą mogli dopytać nauczyciela. Nauczyciel na bieżąco sprawdza postępy uczniów.
2. Bazując na posiadanych zasobach, zapewnij każdemu uczniowi lub zespołowi uczniów kompletny zestaw do robotyki (tj. LEGO® Education SPIKE™).
3. Zestawy te pozwalają uczniom budować własne roboty i programować je do wykonywania określonych zadań. Nauczyciel zaczyna od prostych projektów, takich jak zbudowanie i zaprogramowanie robota, który może wykonać proste czynności, które wprowadzą robota w ruch.
4. Uczniowie budują bazowego robota według załączonej instrukcji (załącznik **Budowa_bazy_robota.pdf** do lekcji).
5. Uczniowie włączają program **SPIKE**, poprzez który będą programować robota.
6. Nauczyciel, wykorzystując e-platformę, demonstruje na tablicy multimedialnej działanie programu i robota, uruchamiając odpowiedni moduł i materiał multimedialny.
7. Prezentowany materiał multimedialny, zrealizowany w oparciu o szczegółowy scenariusz (załącznik do lekcji), oprócz waloru teoretycznego stanowi również podstawę przebiegu części praktycznej lekcji. W szczególności, uczniowie:
 - a. budują prosty model bazowego robota w oparciu o zestaw do robotyki (tj. LEGO® Education SPIKE™),
 - b. tworzą listę czynności niezbędnych do wprowadzenia robota w ruch,
 - c. na podstawie listy czynności przygotowują program, zapisują go we wskazanym folderze (nauczyciel wskazuje właściwą lokalizację; może to być dysk lokalny lub zasób w chmurze) i potrafią go uruchomić;
 - d. wgrywają program do zbudowanego modelu robota, uruchamiają, obserwują i testują działanie robota. Wprowadzają modyfikacje w programie, poprawiają ewentualne błędy.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Faza III. Podsumowanie

Uczniowie analizują przedstawione przykłady programów zapisanych w języku Python oraz omawiają ich działanie. Nauczyciel ocenia stopień zrozumienia zależności między zapisanym kodem, zastosowanymi funkcjami, a **działaniem robota podczas przenoszenia ładunków po uruchomieniu programu.**

Kryteria oceny:

Ocena dopuszczająca:

- uczeń rozpoznaje podstawowe elementy programu w języku Python,
- potrafi wskazać polecenia odpowiedzialne za sterowanie ruchem robota oraz dodatkowym silnikiem,
- rozpoznaje proste funkcje wykorzystywane w programie.

Ocena dostateczna:

- uczeń analizuje przykładowy program i opisuje jego działanie,
- rozumie zależność między zapisanym kodem, funkcjami, a działaniem programu,
- rozpoznaje podstawowe biblioteki i moduły wykorzystywane do sterowania silnikami (motor_pair, motor),
- wskazuje podstawowe błędy składniowe,
- wyjaśnia rolę **funkcji odpowiedzialnych za wykonywanie określonych czynności robota.**

Ocena dobra:

- uczeń samodzielnie modyfikuje przykładowy program,
- poprawia błędy związane z zapisem poleceń, nawiasami lub parametrami ruchu,
- testuje działanie programu i analizuje sposób działania robota oraz manipulatora po wprowadzonych zmianach,
- **tworzy i wykorzystuje własne funkcje usprawniające realizację zadania transportowego.**

Ocena bardzo dobra:

- uczeń samodzielnie tworzy poprawny program w języku Python,
- wyjaśnia działanie poszczególnych elementów programu oraz utworzonych funkcji,
- wykorzystuje biblioteki i moduły do sterowania dodatkowymi silnikami LEGO SPIKE Prime,
- proponuje własne rozwiązania i potrafi je uzasadnić,
- **stosuje funkcje w celu uporządkowania i uproszczenia kodu programu,**
- samodzielnie analizuje i poprawia błędy w kodzie oraz optymalizuje działanie robota podczas realizacji zadania transportowego.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, np.:

1. Jak wywołać funkcję w programie?
 - a. Pisząc słowo def
 - b. Dodając print
 - c. **Podając jej nazwę z nawiasami**
 - d. Zmieniając port
2. Co oznacza zapis **await obrotRobota()**?
 - a. Zatrzymanie programu
 - b. **Wywołanie funkcji i oczekiwanie na jej wykonanie**
 - c. Zmianę prędkości
 - d. Wyłączenie robota
3. Co umożliwia wykorzystanie funkcji w programie robota?
 - a. **Uporządkowanie i skrócenie kodu**
 - b. Zwiększenie prędkości robota
 - c. Zmianę konstrukcji robota
 - d. Usunięcie czujników

Faza V. Dodatkowe, praca domowa (dla chętnych).

Zadania dodatkowe (dla chętnych).

Zadanie 1: Robot kurierski - dostarczanie przesyłki z użyciem funkcji

Utwórz program w języku Python, który pozwoli robotowi dostarczyć „przesyłkę” do wyznaczonego miejsca, a następnie wrócić do punktu startowego. Program powinien wykorzystywać **funkcje** do realizacji poszczególnych etapów zadania.

Program powinien:

- zawierać poprawnie zaimportowane biblioteki i moduły (motor_pair, motor),
- umożliwiać jazdę robota do przodu i do tyłu,
- wykonywać skręty podczas przejazdu,
- realizować przenoszenie i pozostawienie ładunku w wyznaczonym miejscu,
- zawierać co najmniej **trzy funkcje**, np.:
 - **start_trasy()**,
 - **dostarcz_przesylke()**,
 - **powrot_do_bazy()**,
- poprawnie wywoływać funkcje w odpowiedniej kolejności,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- być zapisany poprawnie pod względem składni.

Opcjonalnie możesz rozbudować program tak, aby:

- robot dostarczał przesyłki do kilku różnych punktów,
- robot zatrzymywał się na określony czas w miejscu dostawy,
- program wykorzystywał różne prędkości jazdy,
- na matrycy Spike Hub pojawiały się symbole lub komunikaty informujące o etapie dostawy,
- funkcje przyjmowały parametry, np. odległość jazdy lub prędkość ruchu robota,
- program był podzielony na większą liczbę funkcji zwiększających czytelność kodu.

Załącznik nr 1 – Konspekt nauczyciela lekcji nr 24 - *Przenoszenie ładunków - realizacja zadania z użyciem funkcji.*

Przebieg lekcji:

- Sprawdzenie obecności
- Sprawdzenie stanu wiedzy uczniów (opcjonalnie)
 - Nauczyciel omawia podstawowe zagadnienia związane z **programowaniem funkcji w języku Python**:
 - wyjaśnia, że robot może wykonywać złożone zadania transportowe dzięki odpowiednio zapisanym poleceniom oraz zastosowaniu funkcji porządkujących program,
 - przedstawia przykłady bibliotek i modułów wykorzystywanych do sterowania silnikami LEGO SPIKE Prime (motor_pair, motor),
 - omawia znaczenie poprawnego sterowania ruchem robota oraz wykorzystania funkcji podczas realizacji zadania przenoszenia ładunków.
 - Wprowadzenie do zadania i przypomnienie podstaw pracy w LEGO SPIKE Prime:
 - omówienie środowiska programowania Python w aplikacji LEGO SPIKE App,
 - przedstawienie bibliotek i modułów odpowiedzialnych za sterowanie silnikami,
 - przedstawienie zależności: zapis kodu >> utworzenie funkcji >> uruchomienie programu >> ruch robota i realizacja zadania,
 - analiza przykładowych programów wykorzystujących jazdę robota, skręty oraz funkcje odpowiedzialne za przenoszenie ładunku.
 - Praca uczniów:
 - analiza przedstawionych przykładów programów sterujących robotem,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- omawianie działania poszczególnych linijek kodu oraz utworzonych funkcji,
 - testowanie działania robota po uruchomieniu programu,
 - wyszukiwanie i poprawianie błędów w kodzie oraz funkcjach,
 - proponowanie własnych sposobów usprawnienia transportu i przenoszenia ładunku,
 - **tworzenie własnych funkcji realizujących wybrane czynności robota.**
- Podsumowanie przebiegu lekcji i ocena efektów pracy uczniów (**analiza poprawności działania programu, zastosowanych funkcji oraz skuteczności realizacji zadania przenoszenia ładunków**).
 - Zadanie pracy domowej (opcjonalnie) – **napisanie prostego programu wykorzystującego funkcje do sterowania robotem podczas przenoszenia ładunku.**

Załącznik nr 2 – Realizacja zadania głównego podczas lekcji nr 24 - *Przenoszenie ładunków - realizacja zadania z użyciem funkcji*

Zadanie 1. Przygotowanie do realizacji zadania głównego

Uczniowie, pracując w zespołach lub indywidualnie, realizują zadanie polegające na poznaniu sposobu **programowania funkcji w języku Python oraz sterowaniu ruchem robota w środowisku LEGO SPIKE Prime**. Szczególny nacisk położony jest na poprawne wykorzystanie bibliotek i modułów (**motor_pair, motor**), **tworzenie i wywoływanie funkcji, analizę działania programu oraz interpretację efektów pracy robota podczas realizacji zadania przenoszenia ładunków**.

Uczniowie opracowują i zapisują rozwiązanie problemu w postaci programów tekstowych, które umożliwią sterowanie ruchem robota oraz realizację zadania transportowego z użyciem funkcji. W tym celu:

- analizują przykładowe programy zapisane w języku Python,
- określają cel programu oraz przewidują możliwe błędy składniowe i problemy związane z działaniem robota,
- planują kolejność wykonywania instrukcji w programie,
- opracowują algorytm w uporządkowanej formie,
- tworzą funkcje odpowiedzialne za wybrane czynności robota, np. jazdę, skręt lub przeniesienie ładunku,
- zapisują rozwiązanie jako program w języku Python wykorzystujący biblioteki **motor_pair** i **motor**,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

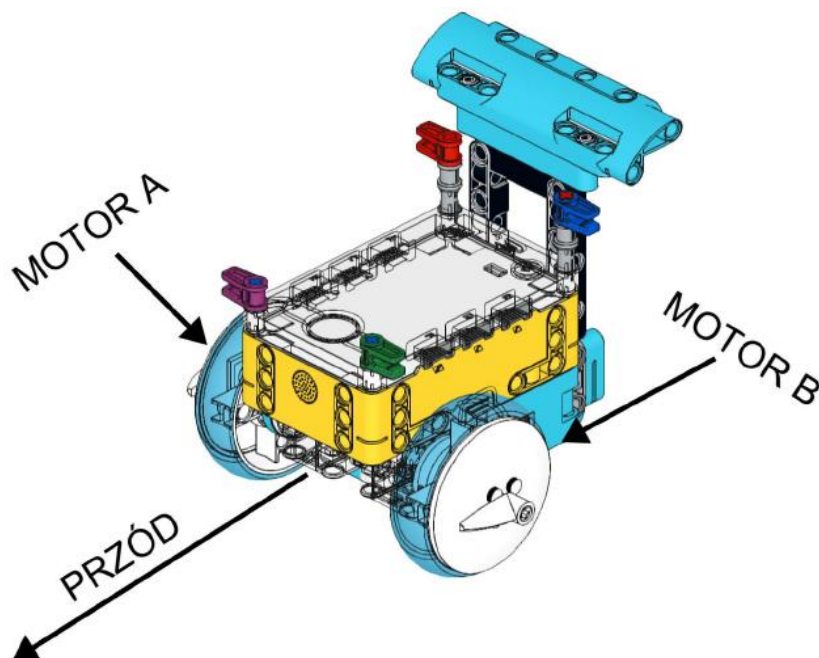
- testują działanie programu oraz analizują zachowanie robota podczas realizacji zadania przenoszenia ładunków,
- wprowadzają poprawki związane z działaniem programu, funkcji oraz parametrami ruchu robota,
- doprecyzowują rozwiązanie tak, aby kod był czytelny, uporządkowany i poprawny składniowo.

Uczniowie wykorzystują środowisko LEGO SPIKE Prime w trybie Python, zwracając uwagę na zgodność między zapisanym kodem, utworzonymi funkcjami, a rzeczywistym działaniem robota. Szczególną uwagę poświęcają poprawności składni, wykorzystaniu funkcji sterujących ruchem robota oraz analizie komunikatów o błędach pojawiających się po uruchomieniu programu.

Jeśli jest taka potrzeba, wskazana przez nauczyciela, to uczniowie zbudują robota od podstaw zgodnie z poniższą instrukcją.

Budowa bazowego robota

Uruchom plik **Budowa bazy robota.pdf** znajdujący się na platformie i postępuj wg podanych w pliku kroków. Efektem końcowym będzie fizyczny robot gotowy do programowania. **Jeśli konstrukcja jest gotowa – idź dalej.**



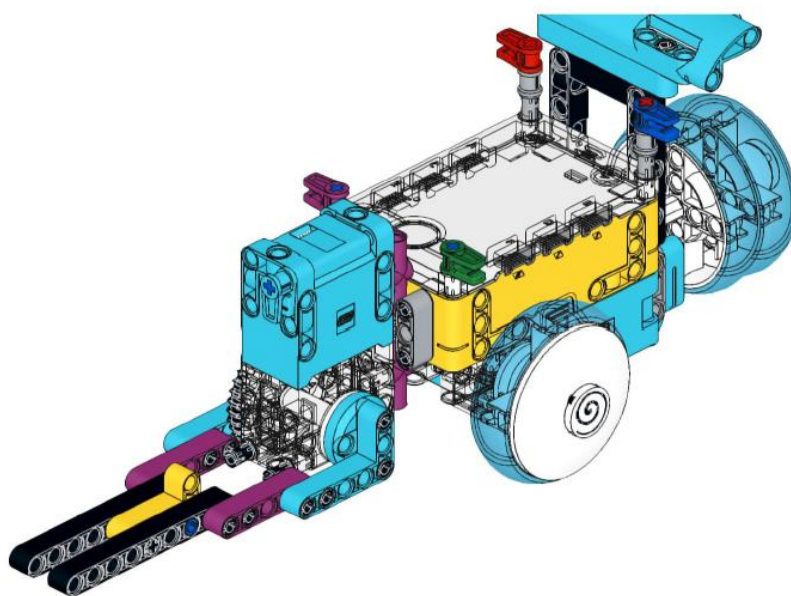
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Pudełka niezbędne do realizacji lekcji: **1,2,3,4 + koła małe**



Dodatkowe instrukcje niezbędne do realizacji lekcji:

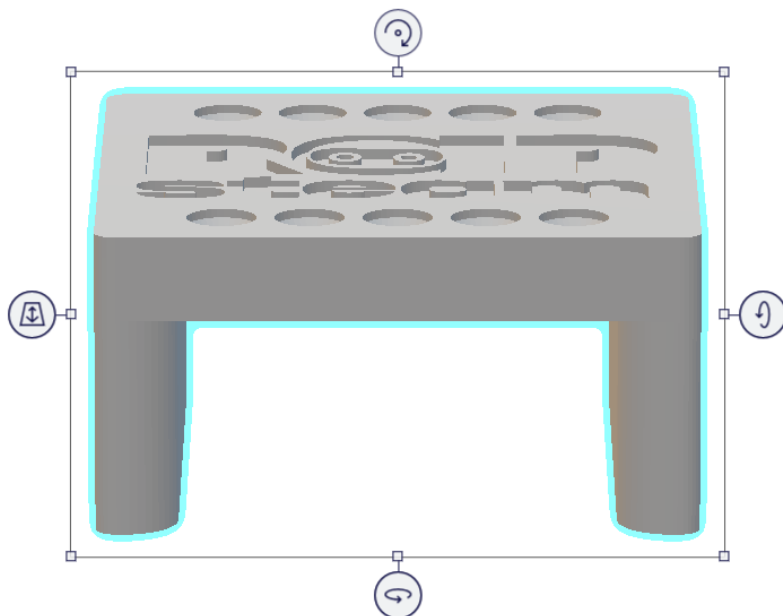
- **chwytakMały.pdf**



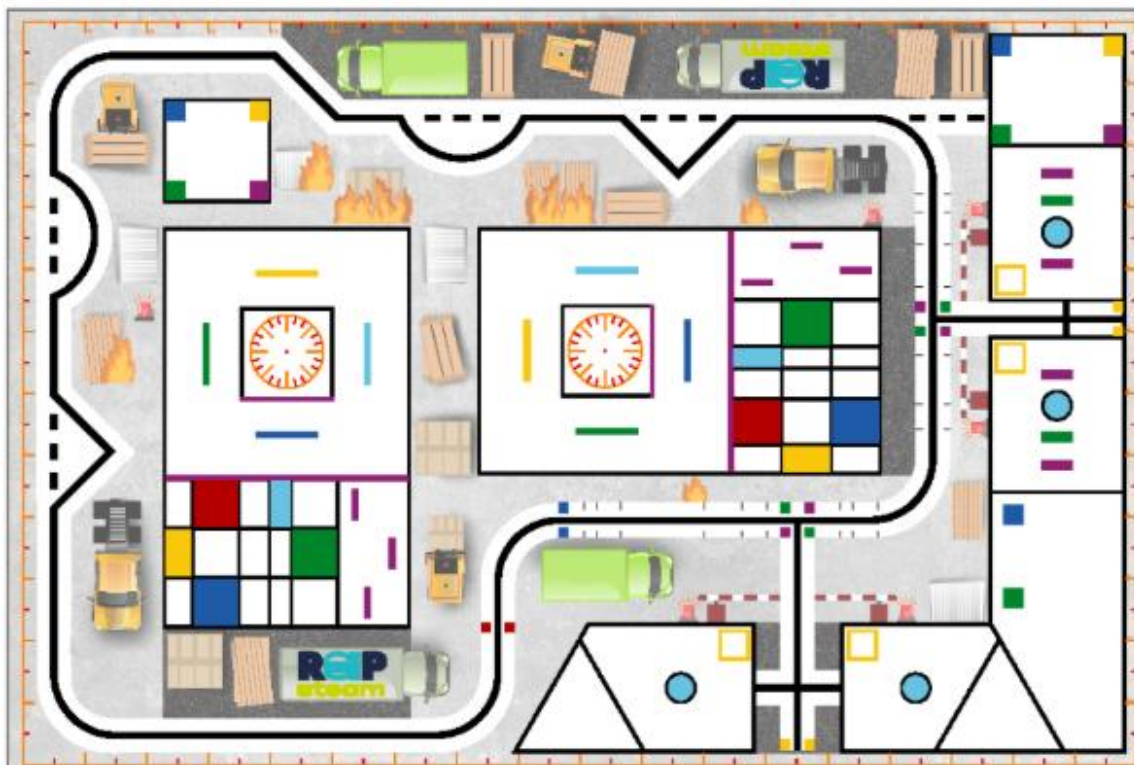
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Dodatkowe elementy niezbędne do realizacji lekcji:

- Wydruk 3D – **Paleta.3mf**, sztuk 1



Do realizacji scenariusza zajęć będziesz potrzebował również matę:

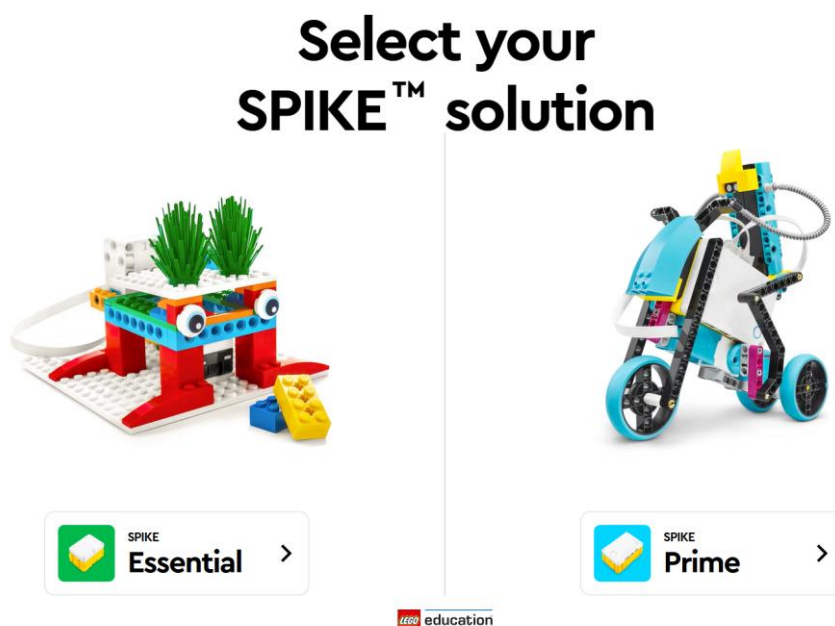


Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 2. Konfiguracja środowiska.

Uruchamianie aplikacji LEGO Spike

1. Włącz przeglądarkę internetową dostępną na twoim urządzeniu
2. Wpisz w przeglądarce adres: <https://spike.legoeducation.com/> Pojawi się okno wyboru wersji LEGO SPIKE. Wybierz Spike Prime

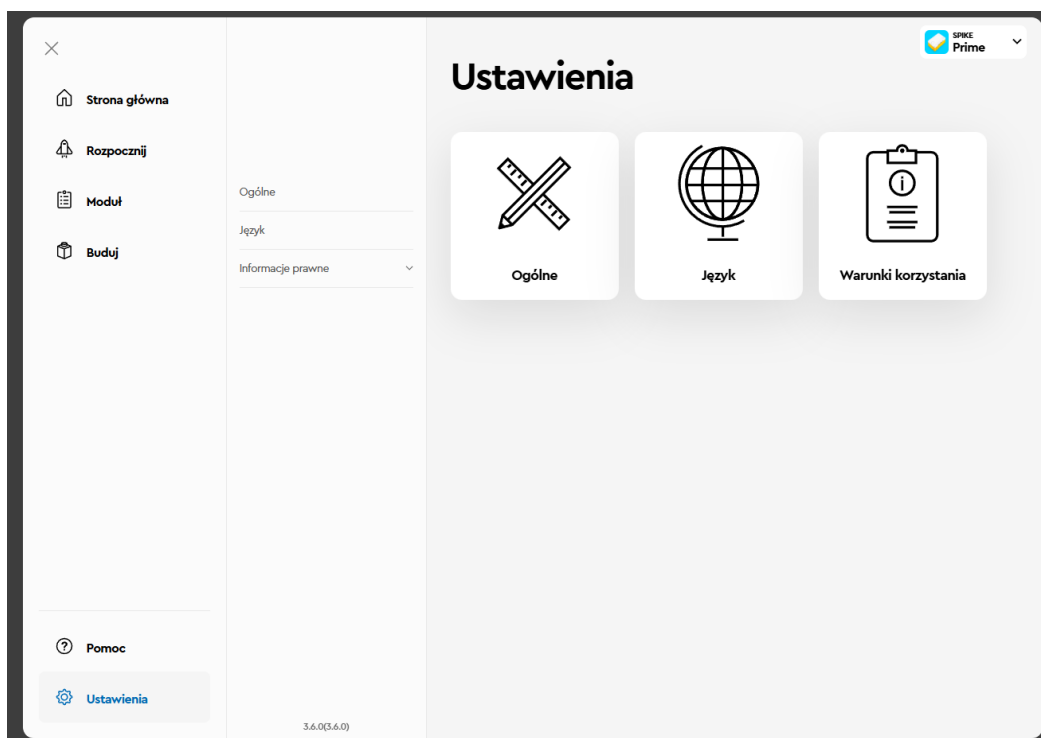


3. Po wybraniu opcji SPIKE Prime pojawia się menu główne. Jeśli występuje język polski, należy zmienić go na język angielski.

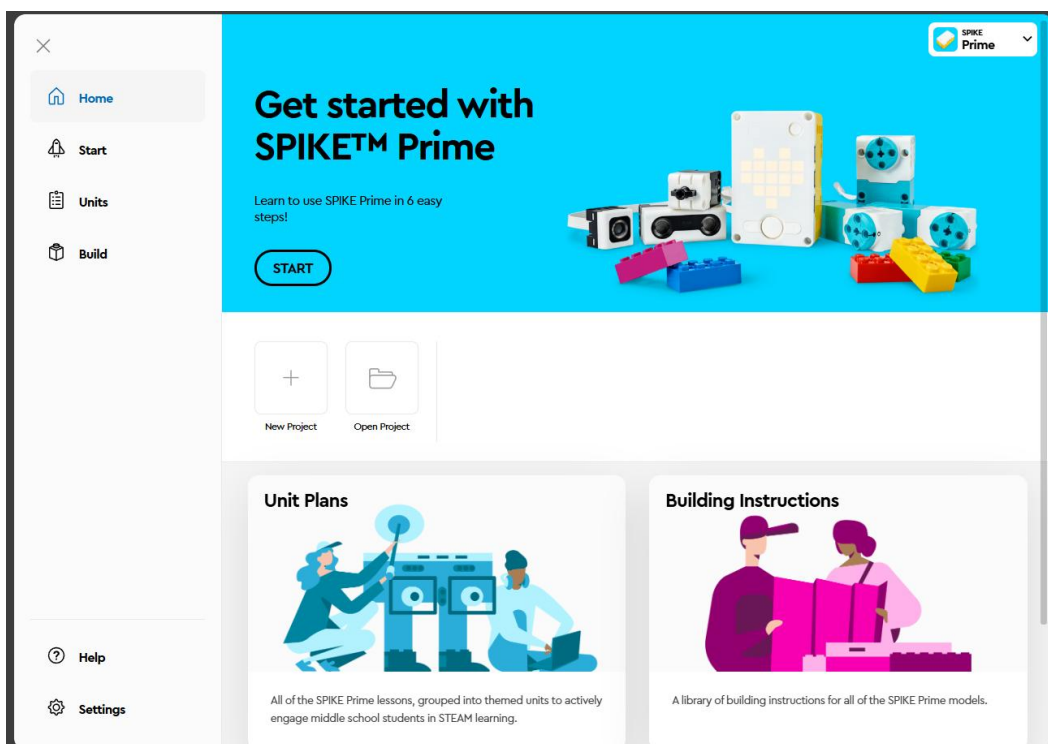
Jeżeli zaś wszystko jest w **języku angielskim**, to przejdź do kroku 6.

Aby zmienić język, w lewym dolnym rogu menu głównego kliknij przycisk **Ustawienia**.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

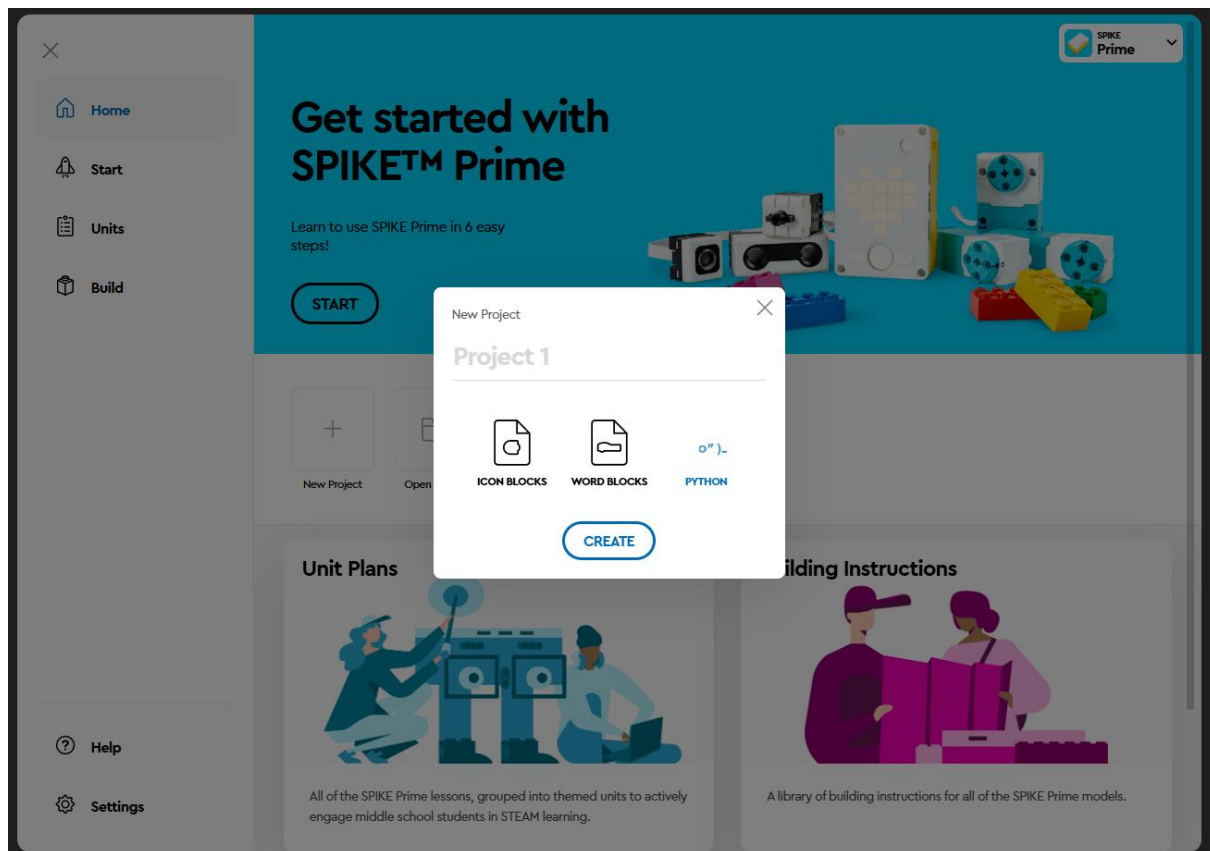


4. W nowym oknie kliknij globus podpisany „Language”.
5. Wybierz język: **United States (English)**. Strona przeładuje się i język angielski będzie dostępny.
6. Kliknij po lewej przycisk **Home**. Pojawi się okno jak poniżej

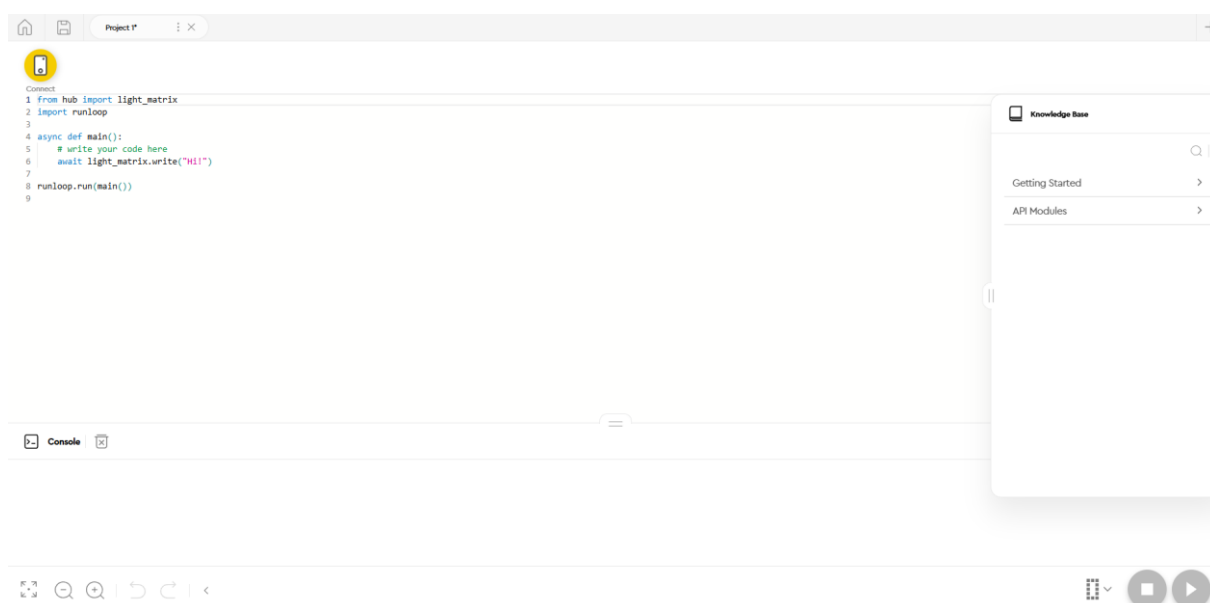


Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

7. Kliknij na środku przycisk **New Project**. Pojawi się nowe okno jak poniżej



8. Wybierz opcję **Python** oraz podaj nazwę projektu. Kiedy wszystko zostanie poprawnie wybrane, wciśnij przycisk **Create**.

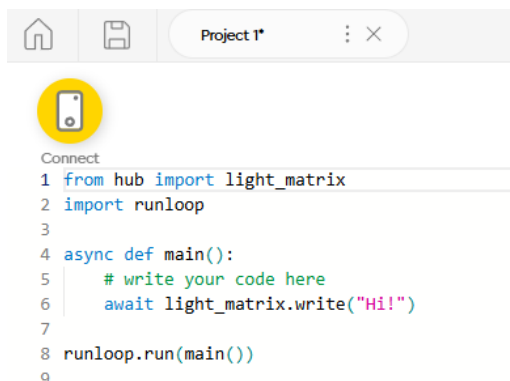


Wszystko jest gotowe do programowania. Pora na podłączenie kontrolera Spike Hub.

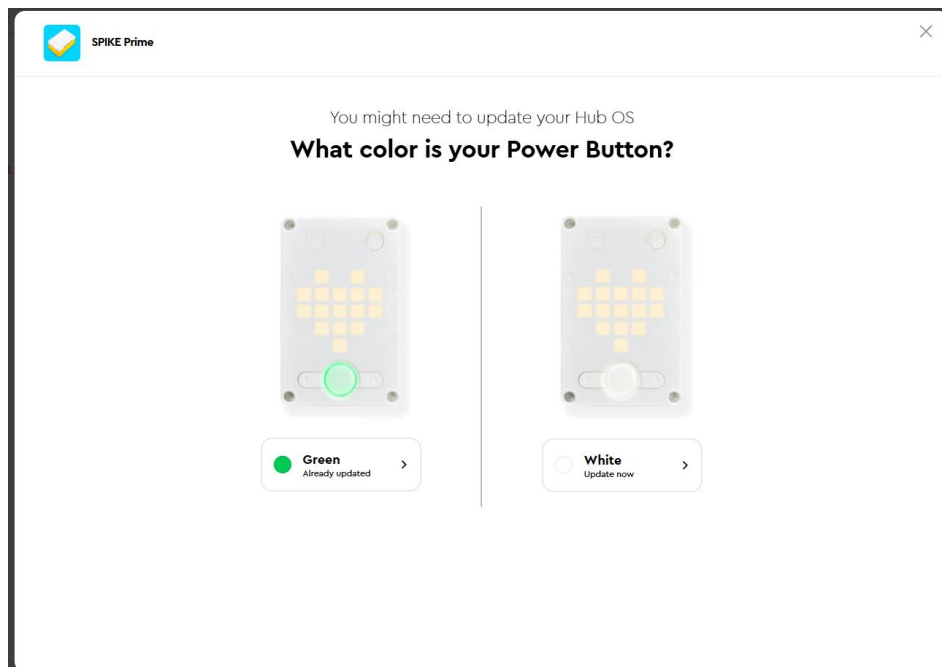
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 3. Podłączenie kontrolera Spike Hub

1. W głównym programie kliknij przycisk **Connect** znajdujący się w górnym rogu, tuż pod nazwą twojego projektu. Patrz poniżej.

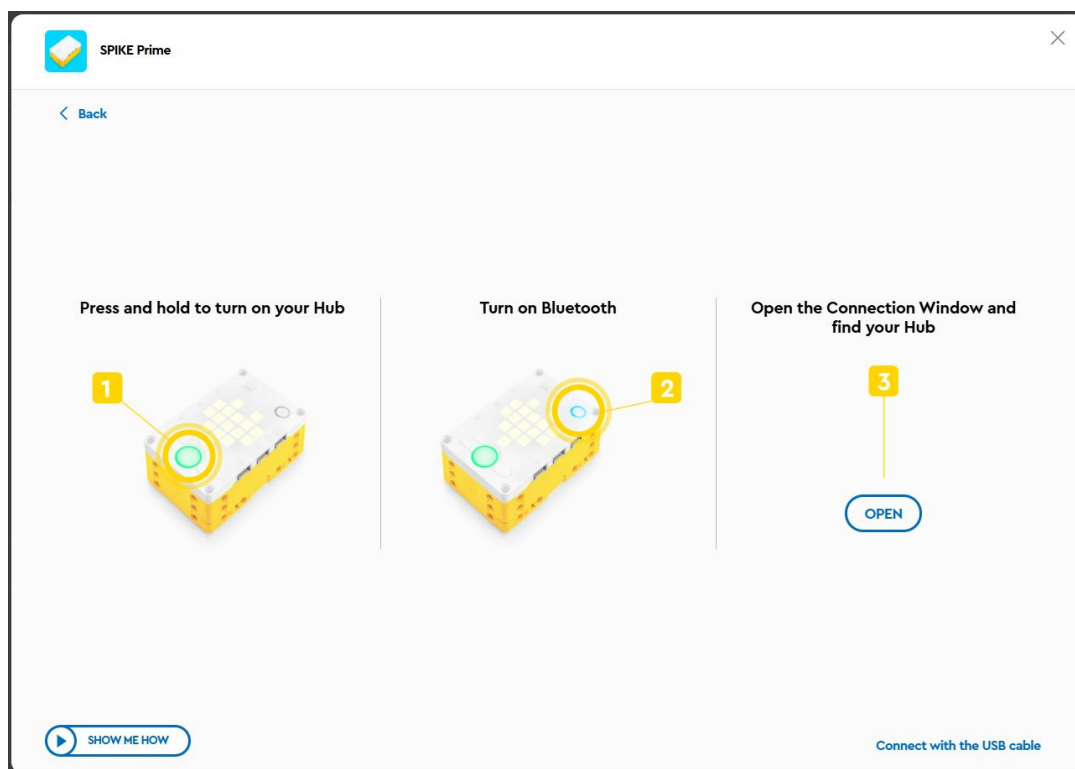


2. Pojawi się nowe okno. **Włącz kontroler Spike Hub** klikając **duży biały przycisk** i poczekaj aż się uruchomi. Następnie wybierz jedną z opcji odpowiadającą stanowi kontrolera Spike Hub – zawsze będzie to **Green – Already updated**.

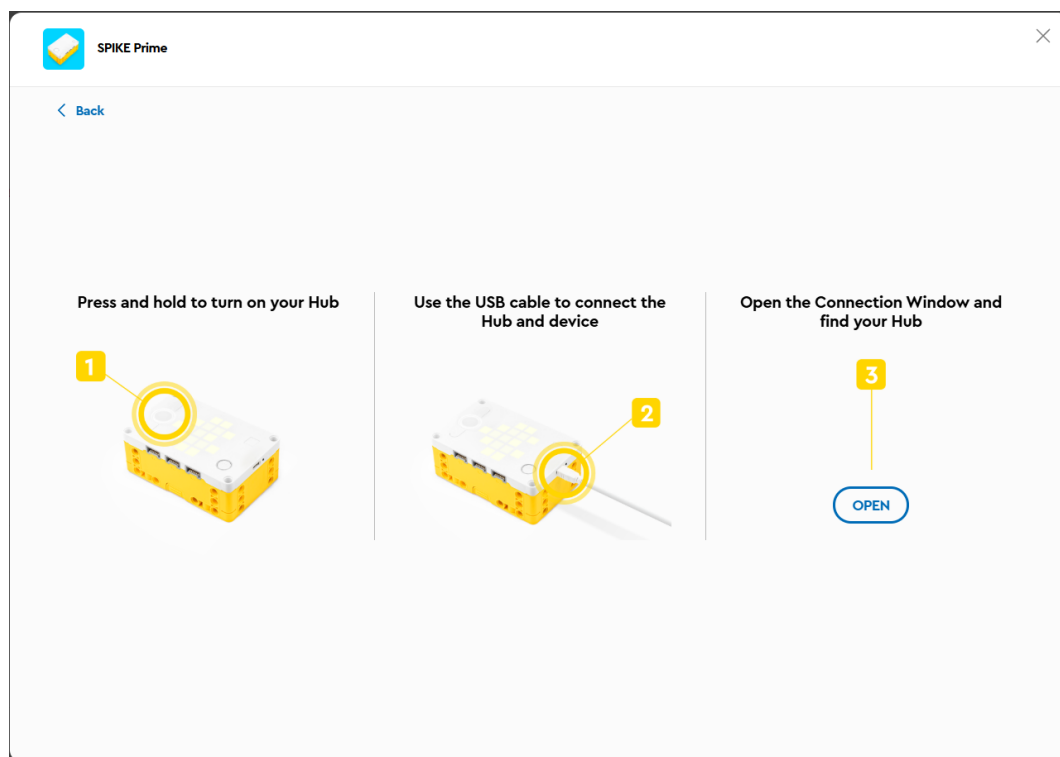


3. Pojawi się nowe okno (patrz poniżej). Wybierz opcję **Connect with USB cable** znajdującą się w prawym dolnym rogu okna.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

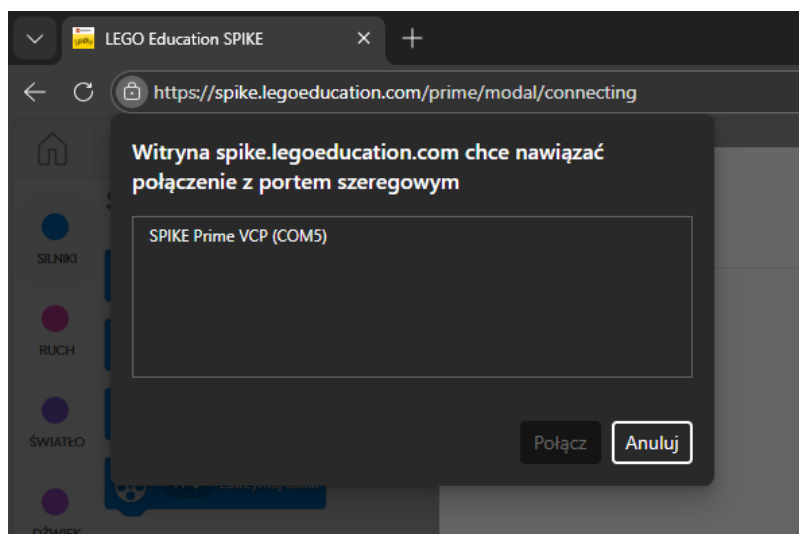


4. Postępuj zgodnie z instrukcją na ekranie. Po przeczytaniu kliknij przycisk **Open**.



5. Pojawi się nowe okno (patrz poniżej)

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



6. Wybierz opcję **SPIKE Prime VCP** i kliknij **Połącz**.

UWAGA – u Ciebie nazwa może nie być identyczna jak na rysunku powyżej. Wybierz tę opcję, która ma w nazwie **Spike Prime**. Innymi rzeczami się nie przejmuj.

Jeśli wszystko zostanie poprawnie połączone, to okna zostaną automatycznie zamknięte, a ikona kontrolera w programie zmieni się na zieloną.



```
1 from hub import light_matrix
2 import runloop
3
4 async def main():
5     # write your code here
6     await light_matrix.write("Hi!")
7
8 runloop.run(main())
9
```

Gratulacje. Twój Spike Hub jest podłączony do komputera. Możesz przystąpić do programowania robota.

W przypadku problemów postępuj wg kroków:

1. Sprawdź, czy kontroler Spike Hub jest włączony i świeci na zielono
 - a. Jeśli kontroler świeci na biało zgłoś to nauczycielowi
 - b. Jeśli kontroler świeci na zielono to podążaj dalej wg instrukcji łączenia powyżej
2. Sprawdź czy kabel USB jest podłączony (Spike Hub <-> Komputer)
3. Sprawdź, czy wybrano odpowiednią nazwę urządzenia – w oknie połączenia z portem szeregowym należy wybrać urządzenie, które w nazwie ma SPIKE Prime.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie 4. Realizacja zadania głównego

W tej lekcji **nauczysz się pisanie własnych funkcji**. Funkcje pozwalają na lepsze zarządzanie kodem, przyspieszenie części zadań i ułatwienie pisanie powtarzalnych rzeczy. Nauczysz się także pisać funkcje z parametrami. Na koniec lekcji wykonasz zadanie z poprzedniej lekcji wykorzystując funkcje.

Przykład realizacji zadania – wprowadzenie bibliotek

UWAGA – w materiałach stosowane są komentarze. Na zrzutach ekranu jest to tekst oznaczony jako zielony. Tego tekstu nie przepisyj! To jest tylko wytłumaczenie dla Ciebie, co dzieje się w danej części programu. Przepisuj tylko i wyłącznie tekst, który nie jest zielony. Korzystaj z tabeli. Pamiętaj, że wielkość liter MA ZNACZENIE!

Na początku programu należy dodać odpowiednie odwołania (biblioteki) do modułów kontrolera Spike Hub i czujników. Dodaj moduł, biblioteki:

- **port** (moduł),
- **light_matrix** (moduł),
- **motor_pair** (biblioteka),
- **motor** (biblioteka)

Aby móc programować silniki niezbędne jest zaimportowanie biblioteki **motor_pair**. Pozwala ona, tak jak w Scratchu za pomocą różnych bloków sterować dwoma silnikami jednocześnie. Zastosowana zostanie także biblioteka **motor** pozwalająca na sterowanie pojedynczym silnikiem. Odpowiednikiem biblioteki motor w Scratchu są niebieskie kafelki. Podczas dzisiejszych zajęć wykorzystasz też moduł **port** oraz **light_matrix** (dostęp do portów oraz wyświetlanie informacji na matrycy Spike Hub).

Zaktualizuj początek programu i dodaj odpowiednie odwołania. Część programu odpowiedzialna za biblioteki powinna wyglądać następująco:

Zdjęcie	Kod (tekst)
<pre>1 from hub import light_matrix, port 2 import runloop 3 import motor_pair, motor</pre>	<pre>from hub import light_matrix, port import runloop import motor_pair, motor</pre>

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

W tej chwili cały program powinien wyglądać następująco:

```
1 from hub import light_matrix
2 from hub import port
3 import motor_pair
4 import runloop
5
6 async def main():
7     # write your code here
8     await light_matrix.write("Hi!")
9
10 runloop.run(main())
```

Przygotowanie struktury programu

Dzisiejsze zajęcia zacznij od przygotowania podstawowych ustawień i parametrów dla kontrolera Spike Hub i silników. Dodaj komendę, która przypisze silniki podpięte do portów B i A do zmiennej **PAIR_1** – tak jak na poprzednich lekcjach. Następnie uruchom silniki tak aby robot:

- Pojechał do przodu 360 stopni, bez skręcania

Możesz zostawić komendę odpowiadającą za wyświetlanie napisu Hi! na kontrolerze Spike Hub. Jeśli chcesz ją usunąć to możesz to zrobić. Nie jest ona wymagana do dzisiejszych zajęć. Program powinien wyglądać następująco:

```
1 from hub import light_matrix, port
2 import runloop
3 import motor_pair
4
5 async def main():
6     # write your code here
7     await light_matrix.write("Hi!")
8
9     #przypisanie portów A i B do PAIR_1
10    motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
11
12    await motor_pair.move_for_degrees(motor_pair.PAIR_1, 360, 0)
13
14 runloop.run(main())
```

Uruchom program wgrywając go do kontrolera Spike Hub i sprawdź, czy robot wykonuje poprawnie swoje zadanie. Jeśli wszystko działa poprawnie przejdź do dalszej lekcji.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Realizacja głównego zadania

Zrealizuj zadanie, którego opracowanie, dokumentację i wyliczenia przygotowano na poprzednich zajęciach. Zgłoś się do nauczyciela po odpowiednie elementy. W dzisiejszej lekcji dodatkowym modułem (poza robotem i jego modułami) będzie:

- **Wydruk 3D – paleta, sztuk 1**

Upewnij się, że masz ją do dyspozycji. Następnie zaprogramuj robota wg ustaleń nauczyciela.

Przykład realizacji zadania

Przedstawiony przykład realizuje rozwiązanie głównego zadania. Robot wykonuje następujące zadania:

- Opuść silnik dodatkowy (manipulator) aby robot po podjechaniu do wydruku 3D (palety) mógł ją swobodnie podnieść,
- Przejedź robotem do przodu, aby nałożyć paletę na chwytak (manipulator),
- Podnieś manipulator, aby paleta nie miała kontaktu z podłożem,
- Podjedź do przodu, aby utrzymać się założonej trasy (w przypadku przykładu jazda następuje po czarnej linii bez jej wykrywania),
- Wykonaj obrót o ok. 90 stopni w prawo. Robot powinien obrócić się w miejscu,
- Jedź do przodu przez 4,5 obroty (ok 1337 stopni), bez skręcania. Robot dojeżdża do celu (punkt B),
- Opuść silnik dodatkowy (manipulator) aby postawić paletę na podłożu,
- Cofnij się o ok. 50 stopni, aby zsunąć paletę z manipulatora,
- Zatrzymaj robota.

Zadanie na pewno jest dla Ciebie znane. To zadanie było realizowane na poprzednich lekcjach.

Na dzisiejszej lekcji poznasz definiowanie funkcji, co przyspieszy twoją pracę. Od początku pracy z Pythonem korzystasz z funkcji. Funkcją jest np. **move_for_degree()**. Jak wiesz, funkcja ta przyjmuje parametry – wartości podawane w nawiasie. Mogą to być zarówno liczby jak i tekst. Wszystko zależy od samej definicji funkcji.

W Pythonie funkcje definiuje się za pomocą słowa def, tak samo jak w przypadku **main()** (który przy okazji też jest funkcją o nazwie **main()**).

Jak wygląda deklaracja funkcji?

Funkcja bez parametrów	Funkcja z parametrami
<code>def nazwaFunkcji() :</code>	<code>def nazwaFunkcji(parametr1, parametr2) :</code>

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Obie metody są poprawne. Wykorzystuje się je w zależności od tego co jest potrzebne. Jeśli funkcja ma działać sama – bez przekazywania informacji z zewnątrz, to stosuje się opcję bez parametrów. Jeśli jednak chcesz przekazać jakąś informację z zewnątrz (np. prędkość silników, położenie robota) to używa się funkcji z parametrami. Parametrów może być bardzo dużo, wszystko zależy od tego co jest potrzebne. Każdy parametr oddziela się przecinkiem.

Pora, aby zrealizować przykładową funkcję. Jak już wiesz, część kodu z poprzednich zajęć się powtarza (np. poruszanie przód / tył, ruch manipulatorem, itp.). Dzięki wykorzystaniu funkcji możliwe jest skrócenie kodu odwołując się do funkcji zamiast pisać długi tekst. Zajmij się kodem związanym z obrotem robota. Kod realizujący zadanie jest następujący:

```
motor_pair.move_for_degrees(motor_pair.PAIR_1, 135, 100)
```

Co prawda jest to tylko jedna linia kodu – ale długa. Trzeba ją przepisywać za każdym razem. Można to skrócić i ułatwić sobie pracę. Zdefiniuj funkcję o nazwie **obrotRobota** który po wywołaniu będzie wykonywał kod powyżej. Funkcję pisze się **POZA MAIN()**.

```
def obrotRobota():  
    motor_pair.move_for_degrees(motor_pair.PAIR_1, 135, 100)
```

Pamiętaj o tabulatorze. Jest on niezbędny, aby w Pythonie blok kodu wykonywał się poprawnie. Twoja funkcja powinna wyglądać następująco:

```
10 #funkcja odpowiadająca za skręcanie w prawo robotem  
11 async def obrotRobota():  
12     motor_pair.move_for_degrees(motor_pair.PAIR_1, 135, 100)
```

Zobacz, jak wygląda cały program:

```
1 from hub import light_matrix, port  
2 import runloop  
3 import motor_pair, motor  
4  
5 #funkcja odpowiadająca za skręcanie w prawo robotem  
6 async def obrotRobota():  
7     motor_pair.move_for_degrees(motor_pair.PAIR_1, 135, 100)  
8  
9 async def main():  
10     # write your code here  
11     await light_matrix.write("Hi!")
```

Jak widzisz pojawia się słowo **async**. Pewnie zastanawiasz się co ono oznacza? Jest ono ściśle powiązane ze słowem **await**, które już dobrze znasz. Aby móc użyć słowa **await**, nawet w komendzie sterowania silnikami, funkcja musi mieć słowo **async** w definicji funkcji. **Jeśli brakuje tego słowa to dostaniesz czerwone podkreślenie i błąd przy uruchomieniu.** Zobacz jak pisana jest funkcja **main()** (ta główna). Ma w sobie słowo **async**, dzięki temu **await** zawsze działał.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Aby wykonać blok kodu z funkcji **obrotRobota** przejdź do **main()** i wywołaj ją używając nazwy, o tak:

```
await obrotRobota()
```

Kod programu powinien wyglądać następująco:

```
1 from hub import light_matrix, port
2 import runloop
3 import motor_pair, motor
4
5 #funkcja odpowiadająca za skręcanie w prawo robotem
6 async def obrotRobota():
7     motor_pair.move_for_degrees(motor_pair.PAIR_1, 135, 100)
8
9 async def main():
10     # write your code here
11     await light_matrix.write("Hi!")
12
13     #przypisanie portów A i B do PAIR_1
14     motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
15
16     #wywołanie funkcji obrócenia robota o 135 stopni w prawo
17     await obrotRobota()
18
19 runloop.run(main())
```

Wspaniale. Po uruchomieniu programu robot powinien się obrócić. Jak to działa w praktyce? Jak wiesz pierwszą funkcją, która startuje w programie jest **main()**. Uruchamia ona wszystkie komendy w **main()**. Tak samo jest z obrotem. Piszac w kodzie:

```
await obrotRobota()
```

program szuka funkcji o nazwie **obrotRobota** a następnie wykonuje to co się w niej znajduje: po kolei, od początku.

Świetnie, pora na funkcję z parametrem. Wykorzystamy do tego manipulator. Napisz funkcję, która podniesie lub opuści ramię w zależności od wartości jaką mu podasz. Do dzieła, oto funkcja:

```
def ruchRamie(stopnie):
    motor.run_for_degrees(port.D, stopnie, 360)
```

Sama funkcja i kod nie powinna być dla Ciebie zaskoczeniem. To już używaliśmy. Jedyny dodatek to tekst w nawiasie. Słowo stopnie to parametr. Oznacza to że kiedy będziemy chcieli wywołać tą funkcję w kodzie to w tym przypadku w nawiasie coś musimy podać. Tak samo

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

jest z funkcją `run_for_degrees()`. Nie uruchomisz jej bez podania argumentu. Co tu jest nowego?

```
motor.run_for_degrees(port.D, stopnie, 360)
```

Zamiast wartości liczbowej pojawia się słowo – to samo, które mamy w nawiasie funkcji **ruchRamie**. Działa to bardzo prosto. Jeśli wywołasz funkcję z wartością np. 30 w **main()** to robisz to w ten sposób:

```
ruchRamie(50)
```

Co faktycznie dzieje się pod spodem? Dokładnie to:

<code>ramieRobota(30)</code>		<code>motor.run_for_degrees(port.D, 30, 360)</code>
<code>ramieRobota(stopnie)</code>		<code>motor.run_for_degrees(port.D, stopnie, 360)</code>

Tak działa przekazywanie parametrów do funkcji i używanie ich wewnątrz. Super, pora to wykorzystać. Jeśli jeszcze tego nie masz zrobione to dodaj funkcję **ramieRobota** w programie poza funkcją **main()**.

Twój program powinien wyglądać teraz tak:

```
1 from hub import light_matrix, port
2 import runloop
3 import motor_pair, motor
4
5 #funkcja odpowiadająca za sterowanie ramieniem. Użycie słowa stopnie pozwala na podanie odpowiedniej wartości w każdej chwili
6 #co czyni funkcję uniwersalną
7 def ruchRamie(stopnie):
8     motor.run_for_degrees(port.D, stopnie, 360)
9
10 #funkcja odpowiadająca za skręcanie w prawo robotem
11 async def obrotRobota():
12     motor_pair.move_for_degrees(motor_pair.PAIR_1, 135, 100)
13
14 async def main():
15
16     # write your code here
17     await light_matrix.write("Hi!")
18
19     #przypisanie portów A i B do PAIR_1
20     motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
21
22 runloop.run(main())
```

Skoro już wszystko wiesz to pora na realizację głównego programu. Wróćmy do tego co należy zrobić:

- Opuść silnik dodatkowy (manipulator) aby robot po podjechaniu do wydruku 3D (palety) mógł ją swobodnie podnieść,
- Przejdź robotem do przodu, aby nałożyć paletę na chwytak (manipulator),
- Podnieś manipulator, aby paleta nie miała kontaktu z podłożem,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- Podjedź do przodu, aby utrzymać się założonej trasy (w przypadku przykładu jazda następuje po czarnej linii bez jej wykrywania),
- Wykonaj obrót o ok. 90 stopni w prawo. Robot powinien obrócić się w miejscu,
- Jedź do przodu przez 4,5 obroty (ok 1337 stopni), bez skręcania. Robot dojeżdża do celu (punkt B),
- Opuść silnik dodatkowy (manipulator) aby postawić paletę na podłożu,
- Cofnij się o ok. 50 stopni, aby zsunąć paletę z manipulatora,
- Zatrzymaj robota.

Jednym z przykładowych rozwiązań tego zadania jest wykorzystanie następujących funkcji:

- **ruchRamie(stopnie)** – zrobione
- **obrotRobota()** – zrobione
- **jazdaRobota(stopnie)** – do zrobienia

Przykład realizacji funkcji **jazdaRobota(stopnie)** – funkcji z parametrem:

```
async def jazdaRobota(stopnie):
```

```
    motor_pair.move_for_degrees(motor_pair.PAIR_1, stopnie, 0)
```

Wykorzystane tu słowo stopnie działa dokładnie tak samo jak w poprzedniej funkcji ale nie ma z nią nic wspólnego – jest to zmienna lokalna. O tym dowiesz się dokładniej na kolejnych lekcjach. Dziś nie musi Cię to interesować.

Wszystkie funkcje są już gotowe. Pora wywołać je w programie. Oto finalny kod.

Biblioteki i twoje funkcje:

```
1 from hub import light_matrix, port
2 import runloop
3 import motor_pair, motor
4
5 #funkcja odpowiadająca za sterowanie ramieniem. Użycie słowa stopnie pozwala na podanie odpowiedniej wartości w każdej chwili
6 #co czyni funkcję uniwersalną
7 def ruchRamie(stopnie):
8     motor.run_for_degrees(port.D, stopnie, 360)
9
10 #funkcja odpowiadająca za skręcanie w prawo robotem
11 async def obrotRobota():
12     motor_pair.move_for_degrees(motor_pair.PAIR_1, 135, 100)
13
14 #funkcja odpowiadająca za sterowanie ruchem robota. Użycie słowa stopnie pozwala na podanie odpowiedniej wartości w każdej chwili
15 #co czyni funkcję uniwersalną.
16
17 #SŁOWO STOPNIE W TEJ FUNKCJI NIE MA NIC WSPÓLNEGO ZE SŁOWEM STOPNIE W FUNKCJI RUCHRAMIE
18
19 async def jazdaRobota(stopnie):
20     motor_pair.move_for_degrees(motor_pair.PAIR_1, stopnie, 0)
21
```

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Funkcja main():

```

22 async def main():
23
24     # write your code here
25     await light_matrix.write("Hi!")
26
27     #przypisanie portów A i B do PAIR_1
28     motor_pair.pair(motor_pair.PAIR_1, port.B, port.A)
29
30     #opuszczenie silnika dodatkowego aby łatwiej podnieść przedmiot o 300 milisekund (0,3 sekundy) na porcie D z prędkością -360 stopni na sekundę
31     #użycie funkcji run for time powoduje opuszczenie chwytaka bez jego zablokowania. Po 300 milisekundach program przejdzie dalej.
32     #w przypadku użycia funkcji run for degrees zły obrót mógłby zablokować cały program
33     await motor.run_for_time(port.D, 300, -360)
34
35     #wywołanie funkcji jazdy robota. Robot pojedzie do przodu przez 1 obrót (360)
36     await jazdaRobota(360)
37
38     #Wywołanie funkcji podnoszenia ramienia. Silnik obróci się w górę o 50 stopni
39     ruchRamie(50)
40
41     #wywołanie funkcji jazdy robota. Robot pojedzie do przodu przez 100 stopni
42     await jazdaRobota(100)
43
44     #wywołanie funkcji obrócenia robota o 135 stopni w prawo
45     await obrótRobota()
46
47     #jazda do przodu bez skręcania (0) przez 1337 stopni ~4,5 obrotów * 360 daje ~1337 stopni.
48     await motor_pair.move_for_degrees(motor_pair.PAIR_1, 1337, 0)
49
50     #Wywołanie funkcji opuszczenia ramienia. Silnik obróci się w dół o 50 stopni dlatego podajemy wartość ujemną
51     ruchRamie(-50)
52
53     #wywołanie funkcji jazdy robota. Robot pojedzie do tyłu przez 1 obrót (360 stopni). Aby robot jechał do tyłu należy podać wartość ujemną
54     await jazdaRobota(-360)
55
56     #zatrzymanie silników robota
57     motor_pair.stop(motor_pair.PAIR_1)
58
59 runloop.run(main())

```

Eksperymentuj z wartościami. Zobacz, jak robot zachowuje się, jeśli podasz mu inne liczby. Powodzenia i do następnej lekcji!

Możliwe rozszerzenia zadania

1. Dodanie nowych funkcji odpowiedzialnych za kolejne etapy przenoszenia ładunku.
2. Zmiana parametrów ruchu robota (prędkości, czasu jazdy, skrętów) i porównanie efektów działania programu.
3. Rozbudowanie programu tak, aby robot przenosił kilka ładunków z rzędu według określonej sekwencji funkcji.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 6: PixelPerfect Studio

Temat lekcji nr 25 - Łączenie klipów filmowych z grafikami wytworzonymi w dziale „Graficzna alchemia przyszłości”

Treści nauczania (wymagania szczegółowe - podstawa programowa) – I. 5); II. 3) a); II. 3) d);

Cel lekcji:

- Uczeń rozumie, na czym polega łączenie klipów filmowych z grafikami w celu stworzenia spójnej opowieści.
- Uczeń wyjaśnia, jak dobór grafiki i jej umiejscowienie w filmie wpływa na odbiór materiału.
- Uczeń potrafi zaplanować prostą strukturę filmu (początek, rozwinięcie, zakończenie) z wykorzystaniem wideo i grafiki w programie Shotcut.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- Uczeń przygotowuje i prezentuje rozwiązania problemów, posługując się podstawowymi aplikacjami (program do montażu wideo – Shotcut) na swoim komputerze.
- Uczeń gromadzi, porządkuje i selekcionuje materiały cyfrowe (klipy wideo oraz grafiki przygotowane wcześniej), potrzebne do stworzenia krótkiego projektu filmowego, oraz zapisuje efekty swojej pracy w postaci projektu.
- Uczeń tworzy i porządkuje informacje w postaci sekwencji na osi czasu programu, łącząc klipy wideo z grafikami, dodając elementy w odpowiednich miejscach (początek, środek, zakończenie) oraz dbając o spójność przekazu.

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści programowe realizowane będą z wykorzystaniem uniwersalnych narzędzi dostępnych w systemach operacyjnych, takich jak eksplorator plików oraz menedżery

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

okien, co umożliwi uczniom sprawne porządkowanie i wybór klipów wideo oraz grafik przygotowanych na wcześniejszych zajęciach.

- W miarę możliwości technicznych uczniowie będą korzystać z zapisu danych w środowiskach wirtualnych (chmurze) w celu gromadzenia, przechowywania oraz udostępniania materiałów graficznych i projektów filmowych.
- Realizacja treści programowych odbywać się będzie w programie Shotcut, który posłuży jako podstawowe narzędzie do łączenia klipów wideo z grafikami, planowania narracji oraz układania materiału na osi czasu z uwzględnieniem początku, rozwinięcia i zakończenia filmu.

Uczeń:

- Wyjaśnia, na czym polega łączenie klipów wideo z grafikami w celu stworzenia spójnej opowieści filmowej.
- Wskazuje, jakie elementy filmu (kolejność ujęć, miejsce wstawienia grafiki, długość wyświetlania) wpływają na odbiór materiału.
- Przedstawia znaczenie montażu oraz wykorzystania grafiki w budowaniu czytelnej i atrakcyjnej narracji.
- Potrafi zaplanować strukturę krótkiego filmu (początek – grafika tytułowa, rozwinięcie – klip wideo z elementami graficznymi, zakończenie – grafika końcowa).
- Opisuje rolę osi czasu programu Shotcut jako narzędzia do łączenia wideo i grafiki oraz porządkowania elementów filmu.

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna,
- praca na komputerze,
- pogadanka.

Środki dydaktyczne

- zestaw PC z dostępem do Internetu oraz zainstalowanym programem Shotcut,
- rzutnik multimedialny,
- e-podręcznik.

Formy pracy

- Praca indywidualna.

Przebieg lekcji:

Faza I. Wprowadzenie

Podczas lekcji uczniowie zostaną zapoznani z zasadami łączenia klipów filmowych z grafikami w celu stworzenia spójnej narracji. Omówione zostanie znaczenie odpowiedniego doboru

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

grafiki, jej umiejscowienia w filmie oraz wpływu na odbiór treści przez widza. Nauczyciel przedstawi również program Shotcut jako narzędzie do łączenia materiałów wideo i graficznych na osi czasu oraz przypomni jego podstawowe elementy: okno podglądu, listę odtwarzania i oś czasu.

1. Nauczyciel rozpoczyna lekcję, wyjaśniając, że film można wzbogacić dodatkowymi elementami, takimi jak grafiki, plansze czy napisy, które pomagają lepiej przekazać treść i przyciągnąć uwagę widza. Podkreśla, że ważne jest nie tylko samo nagranie, ale także sposób jego uzupełnienia i połączenia z innymi elementami.
2. Nauczyciel pyta uczniów, gdzie spotkali się z wykorzystaniem grafik w filmach (np. napisy w filmach, plansze tytułowe, grafiki w filmach na YouTube) oraz zadaje pytanie, czy ich zdaniem dodanie grafiki zmienia sposób odbioru filmu i pomaga lepiej zrozumieć jego treść.

Faza II. Część zasadnicza - realizacja

1. Nauczyciel wprowadza uczniów w temat łączenia filmu z grafiką, wyjaśniając, że odpowiednio dobrane obrazy (plansze, ilustracje, grafiki) mogą uzupełniać treść filmu i pomagać w budowaniu spójnej narracji. Podkreśla, że ważne jest nie tylko to, co pokazuje film, ale także kiedy i w jaki sposób pojawiają się dodatkowe elementy graficzne.
2. Dyskusja kierowana: Nauczyciel zadaje pytania, w jaki sposób grafika może zmienić odbiór filmu oraz gdzie najlepiej ją umieścić (na początku, w środku, na końcu). Uczniowie podają przykłady, np. strona tytułowa, podpisy lub grafiki wyjaśniające treść.
3. Nauczyciel omawia podstawowe zasady łączenia wideo z grafiką, zwracając uwagę na podział filmu na początek (grafika tytułowa), rozwinięcie (klip wideo z dodatkowymi elementami) oraz zakończenie (plansza końcowa). Wyjaśnia również znaczenie przejść i ich wpływ na płynność filmu.
4. Pokaz nauczyciela: Nauczyciel uruchamia program Shotcut i prezentuje, w jaki sposób dodać klip wideo oraz grafiki na oś czasu. Pokazuje, jak wstawić grafikę na początek i koniec filmu, jak rozsunąć klipy, aby zrobić miejsce, oraz jak dodać przejścia między elementami. Demonstruje także, jak zmiana miejsca grafiki wpływa na odbiór filmu w oknie podglądu.
5. Część praktyczna: Uczniowie uruchamiają program Shotcut i pod kierunkiem nauczyciela wykonują ćwiczenie polegające na dodaniu klipu wideo oraz grafik przygotowanych wcześniej. Umieszczają grafikę na początku i końcu filmu, wstawiają dodatkowy element w wybranym miejscu oraz dobierają odpowiednie przejścia, aby stworzyć spójną i czytelną formę filmową.

Faza III. Podsumowanie

Uczniowie prezentują efekty swojej pracy, omawiając sposób połączenia klipu wideo z grafikami oraz wyjaśniając, w jaki sposób dobrali ich miejsce i przejścia, aby stworzyć spójną

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

narrację. Nauczyciel podsumowuje lekcję, zwracając uwagę na znaczenie łączenia obrazu z grafiką, rolę przejść oraz wykorzystanie osi czasu w programie Shotcut do tworzenia czytelnej i atrakcyjnej formy filmowej. Następnie nauczyciel dokonuje oceny pracy uczniów zgodnie z poniższymi kryteriami:

Ocena dopuszczająca:

– uczeń potrafi wskazać podstawowe elementy interfejsu programu Shotcut (okno podglądu, lista odtwarzania, oś czasu) oraz rozumie, że służą one do odtwarzania i układania materiału filmowego i graficznego.

Ocena dostateczna:

– uczeń rozpoznaje podstawowe elementy interfejsu Shotcut i potrafi samodzielnie odtworzyć przygotowany materiał oraz dodać prostą grafikę do projektu.

Ocena dobra:

– uczeń poprawnie opisuje rolę osi czasu oraz potrafi połączyć klip wideo z grafikami, tworząc prostą i logiczną formę filmową, a także wyjaśnia, dlaczego umieścił grafikę w wybranym miejscu.

Ocena bardzo dobra:

– uczeń szczegółowo omawia znaczenie doboru grafiki i przejść w filmie, wskazuje ich wpływ na odbiór materiału oraz poprawnie wykorzystuje narzędzia programu Shotcut do tworzenia spójnej narracji.

Ocena celująca:

– uczeń samodzielnie i twórczo wykorzystuje grafiki w projekcie filmowym, potrafi uzasadnić swoje wybory dotyczące montażu i przejść oraz proponuje własne rozwiązania poprawiające jakość i atrakcyjność filmu.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, np.:

1. Co umożliwia połączenie klipu wideo z grafiką w filmie:
 - a. zmiana koloru tła
 - b. zbiorem przypadkowych nagrań
 - c. **dodanie grafiki na oś czasu**
 - d. zapisanie projektu
2. Gdzie należy umieścić grafikę, aby była widoczna w filmie:
 - a. w oknie podglądu
 - b. **na osi czasu**
 - c. w pasku menu
 - d. w liście odtwarzania

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

3. Co wpływa na płynność przejścia między klipem a grafiką
- rozmiar pliku
 - długość filmu
 - zastosowane przejście i jego ustawienia**
 - nazwa pliku

Faza V. Dodatkowe, praca domowa

Zadanie domowe (dla chętnych).

Uczeń ma za zadanie przygotować krótki opis (5–6 zdań) dotyczący pomysłu na własny film, w którym połączy klip wideo z grafikami. W pracy powinien wskazać temat filmu, opisać, co będzie przedstawiać jego początek (np. grafika tytułowa), rozwinięcie (klip wideo z dodatkowymi elementami graficznymi) oraz zakończenie (plansza końcowa). Uczeń powinien również wyjaśnić, dlaczego wybrał takie grafiki i w jaki sposób ich umiejscowienie wpływa na odbiór filmu. Może oprzeć się na karcie pracy „Plan mojego filmu” wykorzystanej na lekcji.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1 – Sztuka opowiadania historii - zasady montażu filmów dokumentalnych.

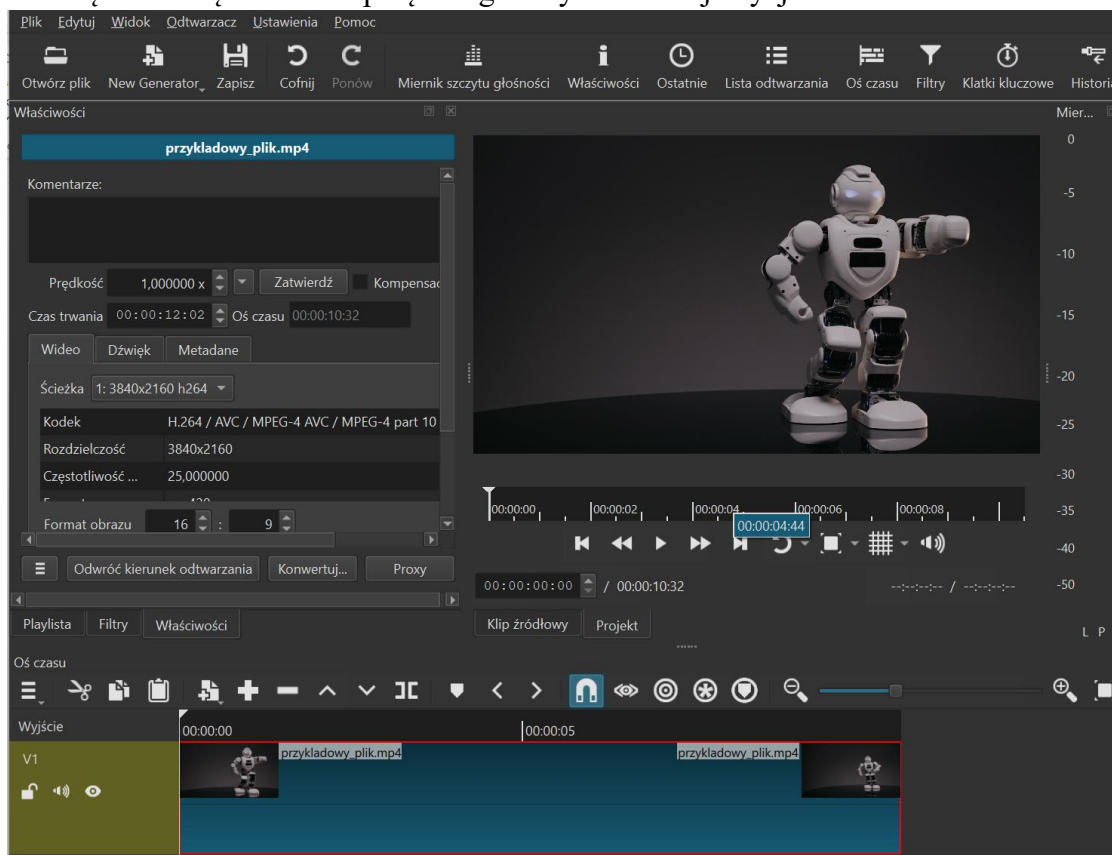
Wprowadzenie!

Dziś zajmiemy się tym, w jaki sposób połączyć film z grafiką, aby stworzyć ciekawą i spójną opowieść. Sam klip wideo często nie wystarcza, aby w pełni przekazać pomysł lub przyciągnąć uwagę widza. Dlatego wykorzystuje się dodatkowe elementy, takie jak grafiki, plansze tytułowe, które uzupełniają obraz i pomagają lepiej zrozumieć treść.

Podczas lekcji zobaczymy, jak odpowiednio dobrać grafiki do filmu oraz jak je wstawiać w taki sposób, aby pasowały do przebiegu historii. Nauczymy się również, jak przejścia między elementami wpływają na płynność i odbiór materiału. Dzięki temu przekonacie się, że dobrze zaplanowany montaż pozwala stworzyć bardziej atrakcyjny i profesjonalny film.

Scenariusz

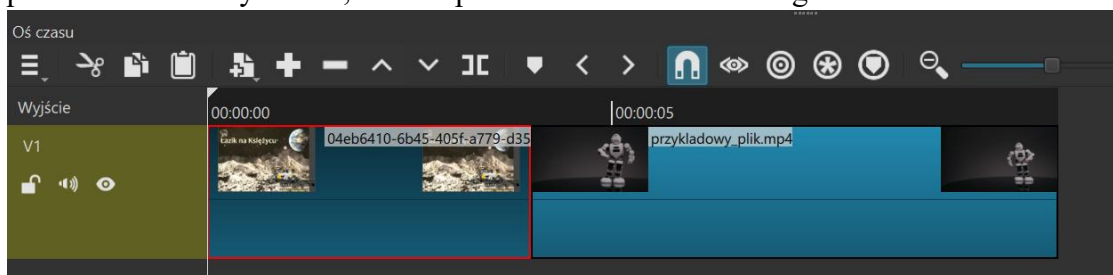
1. Otwórz program Shotcut. Następnie wybierzcie z menu *Plik* oraz *Otwórz plik...* i wczytajcie swój klip wideo. Możecie skorzystać z materiału udostępnionego przez nauczyciela. Po załadowaniu pliku przeciągnijcie go z okna podglądu na oś czasu, na ścieżkę V1. Dzięki temu klip będzie gotowy do dalszej edycji.



Rysunek 1. Widok programu z załadowanym klipem.

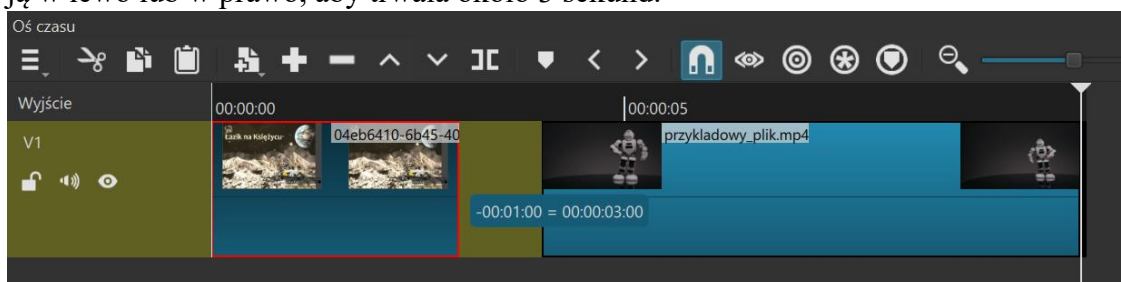
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- Odtwórzcie film przyciskiem *Play* i obejrzyjcie go uważnie. Zastanówcie się, w których momentach można dodać grafiki – na początku, w środku lub na końcu filmu.
- Teraz dodajcie pierwszą grafikę. Wybierzcie Plik i Otwórz plik..., a następnie wskażcie grafikę przygotowaną na wcześniejszych lekcjach w dziale „Graficzna alchemia przyszłości”. Jeśli nie macie swojej grafiki lub nie możecie jej znaleźć, poproście nauczyciela o pomoc. Po wczytaniu pliku przeciągnijcie go z okna podglądu na oś czasu. Umieśćcie grafikę na początku filmu, przed klipem wideo. Dzięki temu powstanie strona tytułowa, która wprowadzi widza do Waszego filmu.



Rysunek 2. Widok osi czasu z dodaną grafiką.

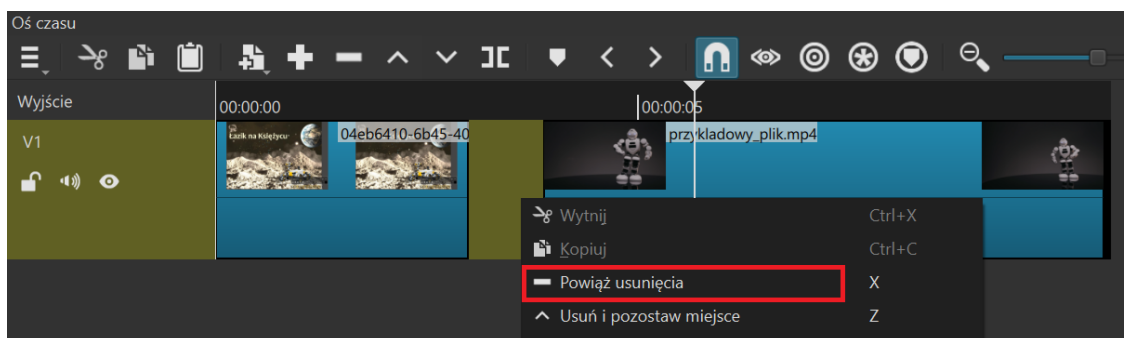
- Skróćcie czas wyświetlania grafiki. Najedźcie myszką na jej krawędź i przeciągnijcie ją w lewo lub w prawo, aby trwała około 3 sekund.



Rysunek 3. Zmniejszanie czasu wyświetlania.

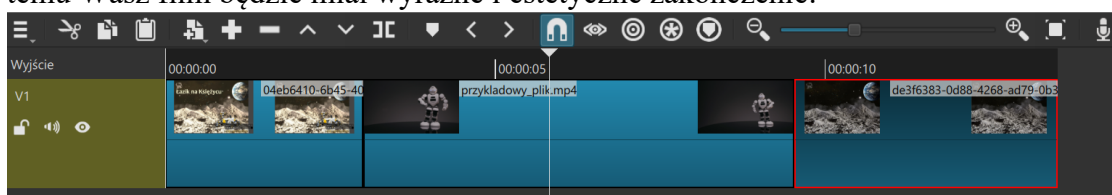
- Jeśli po dodaniu grafiki lub podziale klipu pojawi się puste miejsce na osi czasu, możecie je łatwo usunąć. Kliknijcie prawym przyciskiem myszy w pustym miejscu (przerwie), a następnie wybierzcie opcję *Powiąż usunięcia*. Dzięki temu sąsiednie klipy automatycznie dosuną się do siebie i zniknie niepotrzebna przerwa. Film będzie wyglądał płynniej i bardziej uporządkowanie.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



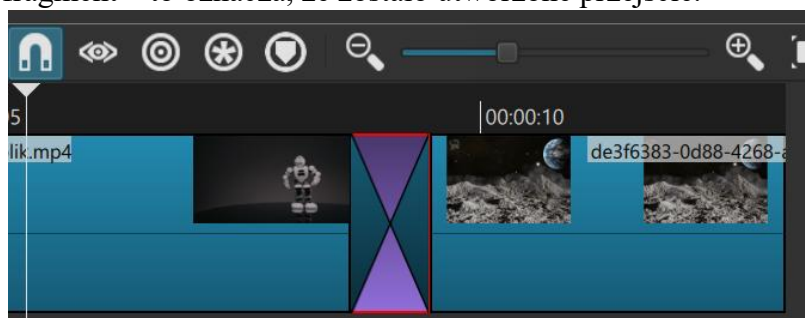
Rysunek 4. Aktywowanie opcji Powiąż usunięcia.

6. Dodajcie kolejną grafikę, która będzie pełniła rolę zakończenia filmu. Wybierzcie *Plik* oraz *Otwórz plik...*, a następnie wskażcie grafikę przygotowaną na wcześniejszych lekcjach w dziale „Graficzna alchemia przyszłości”. Jeśli nie macie swojej grafiki lub nie możecie jej znaleźć, poproście nauczyciela o pomoc. Przeciągnijcie grafikę z okna podglądu na oś czasu i umieśćcie ją na końcu filmu, za klipem wideo. W razie potrzeby możecie dopasować długość wyświetlania grafiki, przeciągając jej krawędź. Dzięki temu Wasz film będzie miał wyraźne i estetyczne zakończenie.



Rysunek 5. Oś czasu z umieszczonymi grafikami.

7. Spróbujcie zastosować przejście, aby film wyglądał bardziej płynnie. Najedźcie myszką na grafikę na osi czasu. Następnie kliknijcie i przeciągnijcie jeden element lekko na drugi (na przykład grafikę na początek lub koniec klipu wideo). Na osi czasu pojawi się zacieniony fragment – to oznacza, że zostało utworzone przejście.

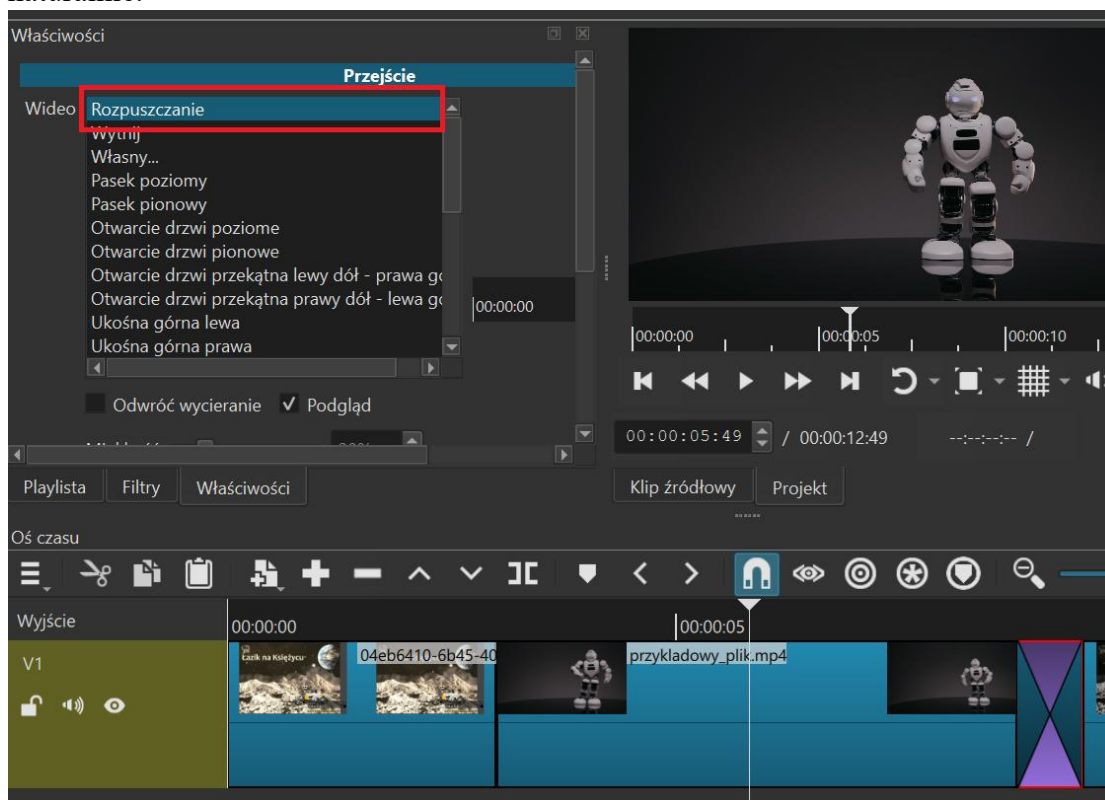


Rysunek 6. Ustawianie przejścia.

8. Teraz spróbujcie samodzielnie dodać efekt przejścia między elementami filmu. Kliknijcie przejście, aby je zaznaczyć. Następnie przejdźcie do zakładki Filtry i wybierzcie efekt przejścia np. Rozpuszczanie, który najbardziej pasuje do Waszego

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

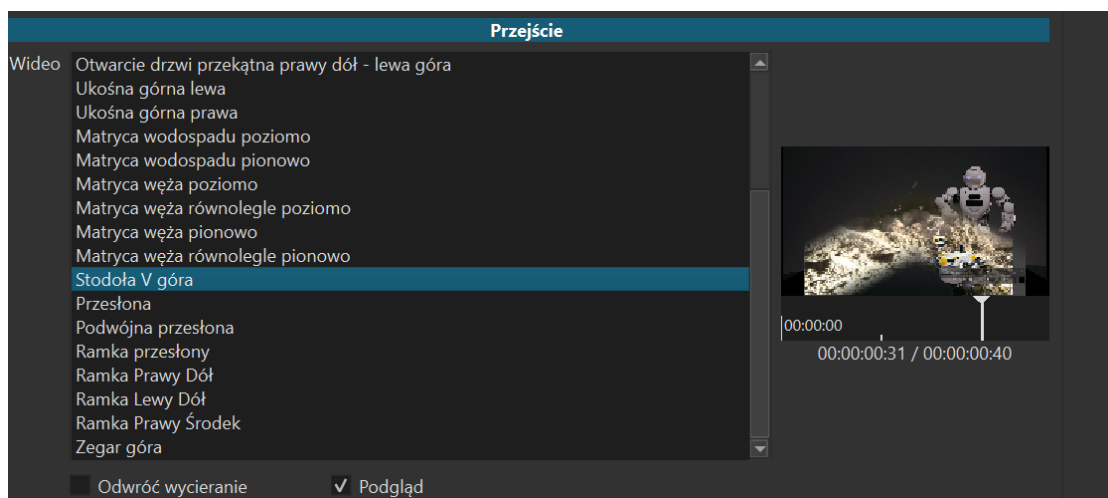
filmu. Możecie także wydłużyć lub skrócić przejście, przeciągając jego krawędzie. Sprawdźcie efekt, odtwarzając film i oceńcie, czy przejście wygląda płynnie i naturalnie.



Rysunek 7. Wybór efektów dla filmu.

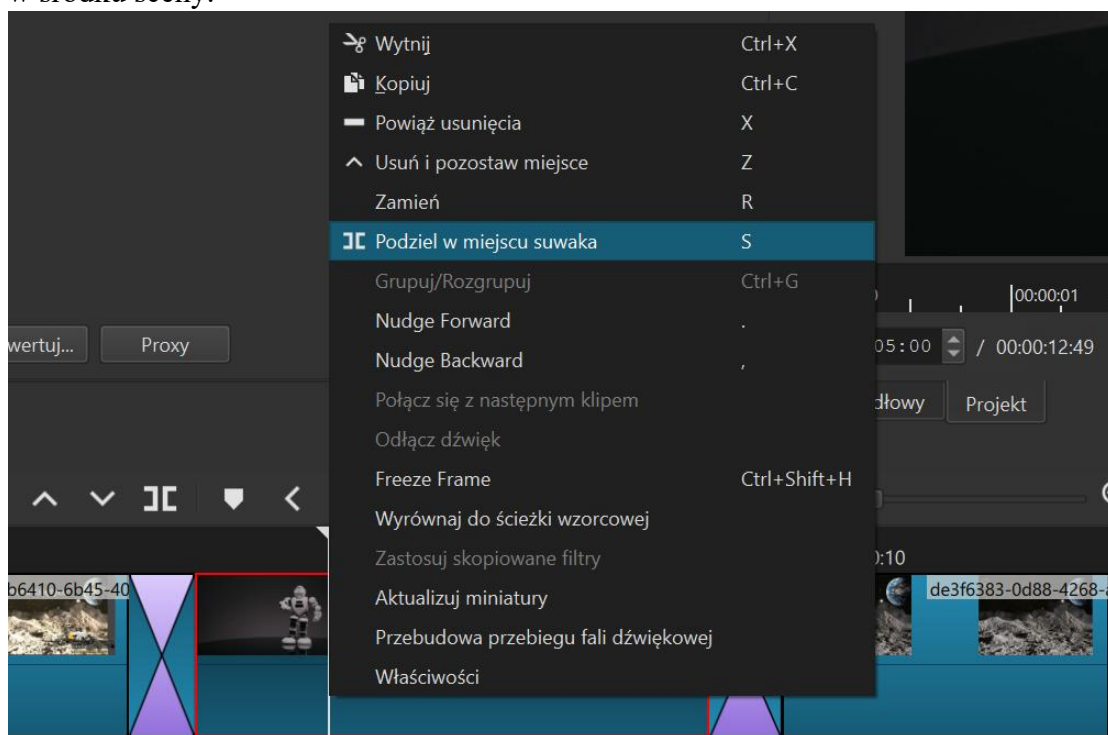
9. Teraz samodzielnie sprawdźcie różne efekty przejścia. Kliknijcie utworzone przejście na osi czasu, aby je zaznaczyć, a następnie przejdźcie do zakładki *Właściwości*, gdzie znajduje się lista dostępnych efektów przejścia. Wybierajcie różne opcje i obserwujcie, jak zmienia się wygląd przejścia. Obok, w oknie podglądu, możecie na bieżąco sprawdzać, jak dany efekt wygląda w filmie. Po każdej zmianie odtwórzcie fragment filmu przyciskiem *Play* i porównajcie efekty. Wybierzcie taki, który najlepiej pasuje do Waszego projektu i sprawia, że zmiana scen wygląda płynnie i naturalnie. Wykonajcie to samo ćwiczenie dla drugiej grafiki, aby oba przejścia w filmie były dopasowane i spójne.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Rysunek 8. Okno z efektami przejść.

10. Teraz spróbujcie podzielić klip wideo. Ustawcie suwak w wybranym miejscu i wybierzcie *Podziel w miejscu suwaka*. Dzięki temu możecie wstawić grafikę dokładnie w środku sceny.

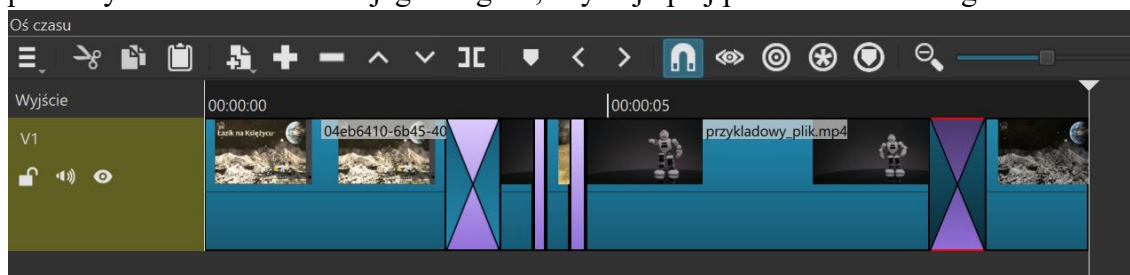


Rysunek 9. Aktywowanie opcji dzielenia filmów.

11. Aby wstawić grafikę pomiędzy podzielone fragmenty filmu, najpierw musicie zrobić dla niej miejsce. Kliknijcie jeden z podzielonych klipów na osi czasu, a następnie przeciągnijcie go myszką w prawo, aby powstała pusta przestrzeń.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

12. Wstawcie grafikę pomiędzy podzielone fragmenty filmu i zobaczcie, jak zmienia się odbiór historii, gdy dodacie obraz w odpowiednim momencie. Następnie dodajcie efekt przejścia między klipem a grafiką, przeciągając jeden element lekko na drugi, aby utworzyć przejście. Sprawdźcie w oknie podglądu, jak wygląda przejście i w razie potrzeby zmieńcie efekt lub jego długość, aby najlepiej pasował do Waszego filmu.



Rysunek 10. Oś czasu z wstawionymi grafikami.

13. Odtwórzcie cały film i sprawdźcie: czy wszystko jest w dobrej kolejności, czy grafiki pasują do treści oraz czy zastosowane przejścia są płynne i pasują do charakteru filmu.
14. Jeśli coś nie pasuje, poprawcie układ elementów – przesuwaszcie, skracajcie lub zmieniajcie ich kolejność. Pamiętajcie, że montaż polega na ciągłym ulepszaniu filmu.
15. Na zakończenie zapiszcie projekt (*Plik i Zapisz jako*), a następnie wyeksportujcie film (*Plik następnie Eksportuj i Wideo...*) i sprawdźcie gotowy efekt.

Praktyczne ćwiczenie:

Samodzielnie otwórz w programie Shotcut klip wideo udostępniony przez nauczyciela lub swój własny materiał. Dodaj go na oś czasu, a następnie wczytaj jedną dodatkową grafikę spoza wcześniejszych zajęć (np. znaną z internetu lub przygotowaną samodzielnie). Wstaw ją w wybranym miejscu filmu, które Twoim zdaniem można urozmaicić lub lepiej wyjaśnić. Dopasuj czas jej wyświetlania oraz spróbuj dobrać odpowiednie przejście, korzystając z zakładki Właściwości. Odtwórz fragment filmu i sprawdź, czy nowy element poprawia odbiór historii. Zastanów się, czy grafika pasuje do treści i czy przejście jest płynne. Jeśli efekt nie jest zadowalający, zmień miejsce wstawienia grafiki lub zastosuj inny efekt przejścia. Na koniec zapisz projekt.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 6: PixelPerfect Studio

Temat lekcji nr 26 - Jak zarządzać wieloma ścieżkami na osi czasu projektu filmowego?

Treści nauczania (wymagania szczegółowe - podstawa programowa) – I. 5); II. 3) a); II. 3) d);

Cel lekcji:

- Uczeń rozumie, na czym polega praca na wielu ścieżkach na osi czasu w programie Shotcut.
- Uczeń wyjaśnia, do czego służą różne ścieżki (video i audio) oraz jak pomagają w porządkowaniu projektu.
- Uczeń potrafi розміścić elementy filmu na odpowiednich ścieżkach (np. klip na V1, napisy lub grafika na V2, dźwięk na A1) oraz zaplanować ich układ w czasie.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- Uczeń przygotowuje i prezentuje rozwiązania problemów, posługując się podstawowymi aplikacjami (program do montażu wideo – Shotcut) na swoim komputerze.
- Uczeń gromadzi, porządkuje i selekcjonuje materiały cyfrowe (klipy wideo, grafiki, dźwięki), potrzebne do realizacji projektu filmowego, oraz zapisuje efekty swojej pracy w postaci projektu.
- Uczeń tworzy i porządkuje informacje na osi czasu programu, wykorzystując wiele ścieżek (video i audio), rozmieszczając elementy w odpowiednich miejscach oraz dbając o czytelność i porządek projektu.

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści programowe realizowane będą z wykorzystaniem uniwersalnych narzędzi dostępnych w systemach operacyjnych, takich jak eksplorator plików oraz menedżery

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

okien, co umożliwi uczniom sprawne porządkowanie i wybór materiałów (klipów wideo, grafik i dźwięków) wykorzystywanych w projekcie.

- W miarę możliwości technicznych uczniowie będą korzystać z zapisu danych w środowiskach wirtualnych (chmurze) w celu gromadzenia, przechowywania oraz udostępniania projektów filmowych.
- Realizacja treści programowych odbywać się będzie w programie Shotcut, który posłuży jako podstawowe narzędzie do pracy na wielu ścieżkach, porządkowania elementów projektu oraz układania materiału na osi czasu w sposób czytelny i uporządkowany.

Uczeń:

- Wyjaśnia, na czym polega praca na wielu ścieżkach w programie Shotcut i do czego służą ścieżki wideo oraz audio.
- Wskazuje, jakie elementy projektu (np. klip wideo, napisy, grafika, dźwięk) powinny znajdować się na odpowiednich ścieżkach.
- Przedstawia znaczenie porządkowania materiałów na osi czasu oraz wyjaśnia, dlaczego warto oddzielać elementy na różne ścieżki.
- Potrafi rozmieścić elementy projektu na ścieżkach (np. V1 – klip główny, V2 – napisy lub grafika, A1 – dźwięk) i dopasować ich czas wyświetlania.
- Opisuje rolę osi czasu jako narzędzia do zarządzania wieloma elementami filmu oraz kontrolowania ich kolejności i czasu trwania.

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna,
- praca na komputerze,
- pogadanka.

Środki dydaktyczne

- zestaw PC z dostępem do Internetu oraz zainstalowanym programem Shotcut,
- rzutnik multimedialny,
- e-podręcznik.

Formy pracy

- Praca indywidualna.

Przebieg lekcji:

Faza I. Wprowadzenie

Podczas lekcji uczniowie zostaną zapoznani z zasadami pracy na wielu ścieżkach w programie Shotcut. Omówione zostanie znaczenie porządkowania elementów filmu (wideo, grafiki,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

napisy, dźwięk) na oddzielnych ścieżkach oraz wpływ takiej organizacji na wygodę pracy i czytelność projektu. Nauczyciel przypomni również podstawowe elementy programu Shotcut: okno podglądu, listę odtwarzania i oś czasu, ze szczególnym uwzględnieniem ścieżek V1, V2 i A1.

1. Nauczyciel rozpoczyna lekcję, wyjaśniając, że w bardziej rozbudowanych projektach filmowych wykorzystuje się wiele elementów jednocześnie, dlatego ważne jest odpowiednie rozmieszczenie ich na różnych ścieżkach. Podkreśla, że dobra organizacja pracy ułatwia montaż i wprowadzanie zmian.
2. Nauczyciel pyta uczniów, jakie elementy mogą pojawić się w filmie (np. obraz, napisy, grafika, dźwięk) oraz zadaje pytanie, czy łatwiej jest pracować, gdy wszystkie elementy są uporządkowane na osobnych ścieżkach, czy gdy znajdują się w jednym miejscu.

Faza II. Część zasadnicza - realizacja

1. Nauczyciel wprowadza uczniów w temat pracy na wielu ścieżkach, wyjaśniając, że w bardziej rozbudowanych projektach filmowych różne elementy (wideo, napisy, grafika, dźwięk) umieszcza się na oddzielnych ścieżkach. Podkreśla, że takie rozwiązanie ułatwia porządkowanie projektu i wprowadzanie zmian.
2. Dyskusja kierowana: Nauczyciel zadaje pytania, jakie elementy mogą pojawić się w filmie oraz gdzie powinny być umieszczone (np. klip na V1, napisy na V2, dźwięk na A1). Uczniowie podają przykłady i zastanawiają się, dlaczego warto rozdzielać elementy na różne ścieżki.
3. Nauczyciel omawia podstawowe zasady pracy na osi czasu, zwracając uwagę na różnice między ścieżkami wideo i audio oraz ich zastosowanie. Wyjaśnia, jak rozmieszczenie elementów na ścieżkach wpływa na czytelność projektu i łatwość edycji.
4. Pokaz nauczyciela: Nauczyciel uruchamia program Shotcut i prezentuje, w jaki sposób dodać nowe ścieżki wideo i audio. Pokazuje, jak umieścić klip wideo na ścieżce V1, dodać element z napisem na V2 oraz dźwięk na A1. Demonstruje także, jak przesuwać elementy i dopasowywać ich czas trwania oraz jak wpływa to na efekt widoczny w oknie podglądu.
5. Część praktyczna: Uczniowie uruchamiają program Shotcut i pod kierunkiem nauczyciela wykonują ćwiczenie polegające na pracy na wielu ścieżkach. Dodają klip wideo na V1, element z napisem lub grafiką na V2 oraz (opcjonalnie) dźwięk na A1. Następnie porządkują elementy, dopasowują ich długość i sprawdzają w podglądzie, czy projekt jest czytelny i poprawnie ułożony..

Faza III. Podsumowanie

Uczniowie prezentują efekty swojej pracy, omawiając sposób rozmieszczenia elementów na różnych ścieżkach oraz wyjaśniając, jak uporządkowali projekt (np. klip na V1, napisy lub grafika na V2, dźwięk na A1). Nauczyciel podsumowuje lekcję, zwracając uwagę na znaczenie pracy na wielu ścieżkach, porządkowania elementów oraz wykorzystania osi czasu

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

w programie Shotcut do tworzenia czytelnego i uporządkowanego projektu filmowego.
Następnie nauczyciel dokonuje oceny pracy uczniów zgodnie z poniższymi kryteriami:

Ocena dopuszczająca:

– uczeń potrafi wskazać podstawowe elementy interfejsu programu Shotcut (okno podglądu, lista odtwarzania, oś czasu) oraz rozumie, że służą one do odtwarzania i układania materiału filmowego.

Ocena dostateczna:

– uczeń rozpoznaje podstawowe elementy interfejsu Shotcut i potrafi samodzielnie odtworzyć przygotowany materiał oraz wskazać różne ścieżki na osi czasu.

Ocena dobra:

– uczeń poprawnie opisuje rolę osi czasu oraz potrafi rozmieścić elementy na odpowiednich ścieżkach (np. V1, V2, A1), tworząc uporządkowany projekt.

Ocena bardzo dobra:

– uczeń szczegółowo omawia znaczenie pracy na wielu ścieżkach, wskazuje ich zastosowanie oraz poprawnie wykorzystuje je do organizacji projektu filmowego.

Ocena celująca:

– uczeń samodzielnie i świadomie planuje pracę na wielu ścieżkach, potrafi uzasadnić swoje decyzje dotyczące rozmieszczenia elementów oraz proponuje własne sposoby uporządkowania i ulepszenia projektu.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, np.:

1. Do czego służą różne ścieżki na osi czasu w programie Shotcut?:
 - a. do zmiany koloru filmu
 - b. do zapisywania projektu
 - c. **do porządkowania elementów (video, napisy, dźwięk)**
 - d. do zamykania programu
2. Na której ścieżce najlepiej umieścić główny klip wideo:
 - a. V2
 - b. **V1**
 - c. A1
 - d. w liście odtwarzania
3. Gdzie należy dodać napisy lub grafikę, aby były widoczne na filmie:
 - a. w oknie podglądu
 - b. w pasku menu
 - c. **na ścieżce wideo, np. V2**

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

d. w liście odtwarzania

Faza V. Dodatkowe, praca domowa

Zadanie domowe (dla chętnych).

Uczeń ma za zadanie przygotować krótki opis (5–6 zdań) dotyczący pracy nad projektem filmowym z wykorzystaniem wielu ścieżek. W pracy powinien wyjaśnić, jakie elementy umieściłby na poszczególnych ścieżkach (np. V1 – klip wideo, V2 – napisy lub grafika, A1 – dźwięk) oraz opisać, dlaczego taki podział ułatwia pracę. Uczeń powinien również wskazać, jak rozmieszczenie elementów na osi czasu wpływa na czytelność projektu i odbiór filmu. Może oprzeć się na projekcie wykonanym na lekcji.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1 – Sztuka opowiadania historii - zasady montażu filmów dokumentalnych.

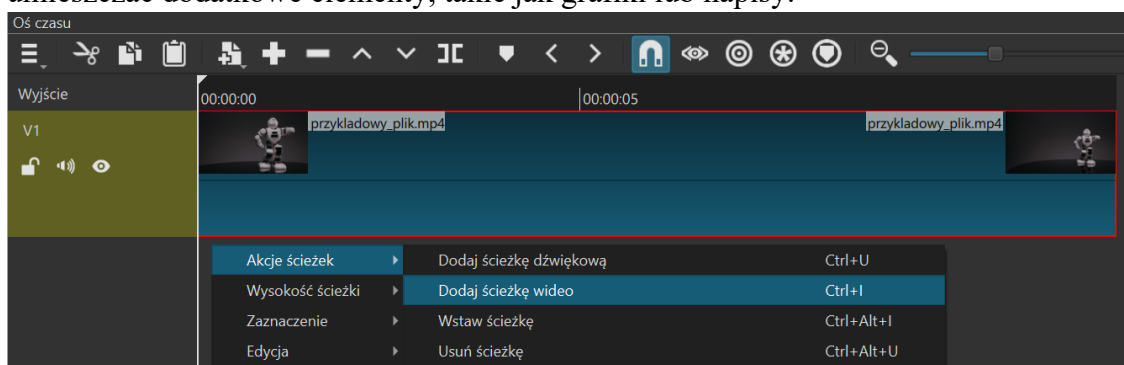
Wprowadzenie!

Dziś zajmiemy się tym, w jaki sposób uporządkować projekt filmowy, korzystając z wielu ścieżek na osi czasu. W bardziej rozbudowanych filmach pojawia się wiele elementów – klipy wideo, napisy, grafiki oraz dźwięk. Umieszczenie ich wszystkich w jednym miejscu byłoby trudne do uporządkowania, dlatego wykorzystuje się różne ścieżki.

Podczas lekcji zobaczymy, jak dodawać nowe ścieżki w programie Shotcut oraz jak przypisywać do nich odpowiednie elementy. Nauczymy się również, jak przesuwać i dopasowywać materiały, aby projekt był czytelny i łatwy do edycji. Dzięki temu przekonacie się, że dobra organizacja pracy znacznie ułatwia montaż filmu.

Scenariusz

1. Uruchomcie program Shotcut. Następnie wybierzcie z menu *Plik, Otwórz plik...* i wczytajcie klip wideo. Możecie skorzystać z materiału udostępnionego przez nauczyciela. Po załadowaniu pliku przeciągnijcie go z okna podglądu na oś czasu, na ścieżkę V1. Dzięki temu klip będzie gotowy do dalszej edycji.
2. Kliknijcie prawym przyciskiem myszy w pustym miejscu osi czasu. Następnie z wyświetlonego menu wybierzcie *Akcje ścieżek* następnie *Dodaj ścieżkę wideo* (skrót: Ctrl+I). Na osi czasu pojawi się nowa ścieżka, np. V2, na której będziecie mogli umieszczać dodatkowe elementy, takie jak grafiki lub napisy.



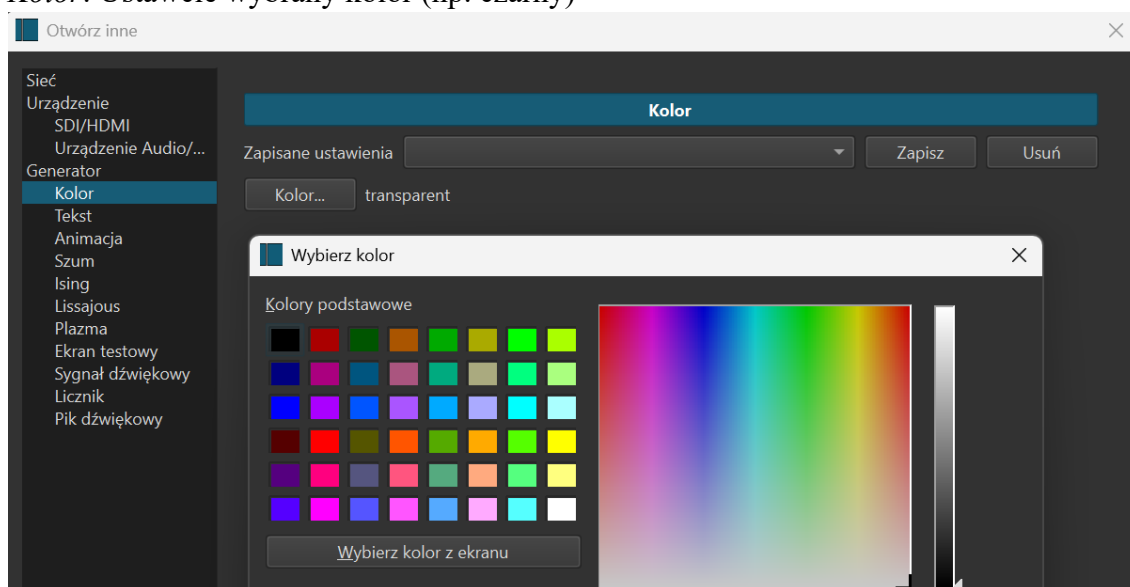
Rysunek 1. Dodawanie nowej ścieżki video.

3. Kliknijcie ponownie prawym przyciskiem myszy w pustym miejscu osi czasu. Następnie wybierzcie **Akcje ścieżek**, a potem **Dodaj ścieżkę dźwiękową** (skrót:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

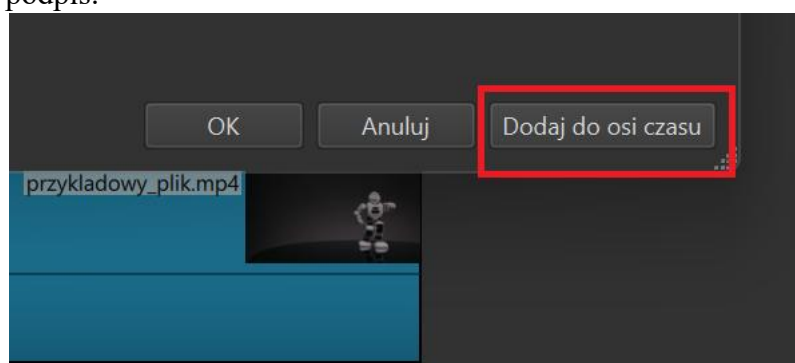
Ctrl+U). Na osi czasu pojawi się ścieżka A1, przeznaczona na muzykę, narrację lub efekty dźwiękowe.

4. Teraz dodacie element graficzny, który będzie wyświetlany na filmie jako tło pod napis. Wybierzcie z menu *Plik, Otwórz inny*, a następnie w sekcji *Generator* wybierzcie opcję *Kolor*. Ustawcie wybrany kolor (np. czarny)



Rysunek 2. Generowanie elementu graficznego.

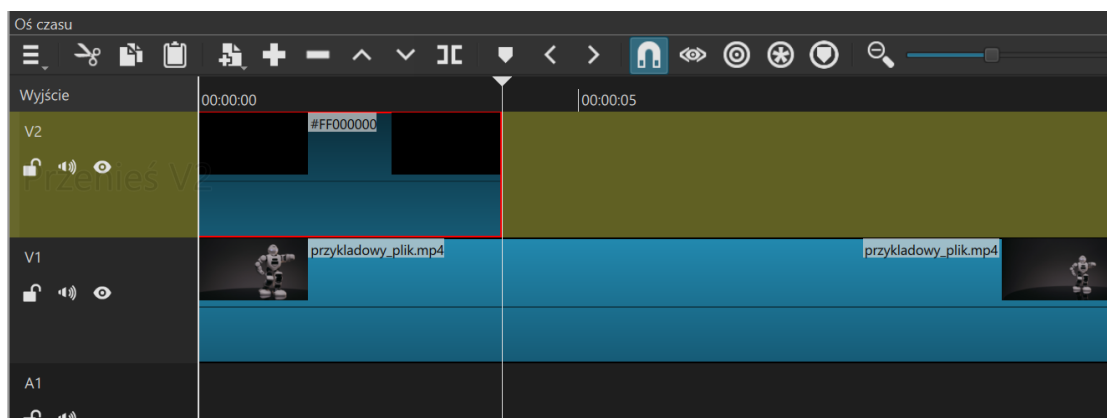
5. Po wybraniu koloru kliknijcie przycisk *Dodaj do osi czasu*. Element zostanie automatycznie umieszczony na ścieżce V2. Następnie ustawcie go w odpowiednim miejscu na osi czasu, np. na początku lub w wybranym fragmencie filmu, gdzie ma pojawić się podpis.



Rysunek 3. Dodawanie grafiku do ścieżki.

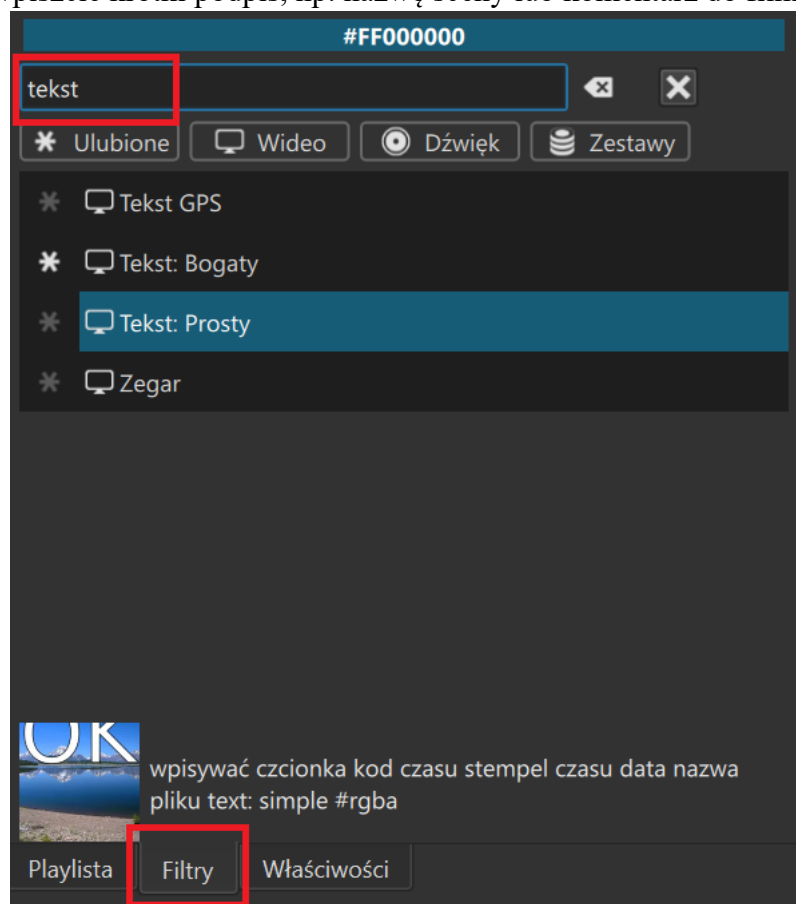
6. Spójrzcie na oś czasu i zwróćcie uwagę, jak wyglądają poszczególne ścieżki. Ścieżka V1 zawiera główny klip wideo, ścieżka V2 elementy dodatkowe (np. kolorowe tło lub napisy), a ścieżka A1 dźwięk. Dzięki temu możecie łatwo rozróżnić, które elementy odpowiadają za obraz, które za dodatki, a które za dźwięk, i wygodnie nimi zarządzać podczas montażu.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Rysunek 4. Widok ścieżek.

7. Zaznaczcie element dodany na ścieżce V2, a następnie przejdźcie do zakładki *Filtry*. Kliknijcie przycisk +, wybierzcie *Wideo*, a potem *Tekst: Prosty*. Jeśli nie widzicie tej opcji, możecie skorzystać z pola wyszukiwania w filtrach i wpisać słowo „tekst”. Następnie wpiszcie krótki podpis, np. nazwę sceny lub komentarz do filmu.



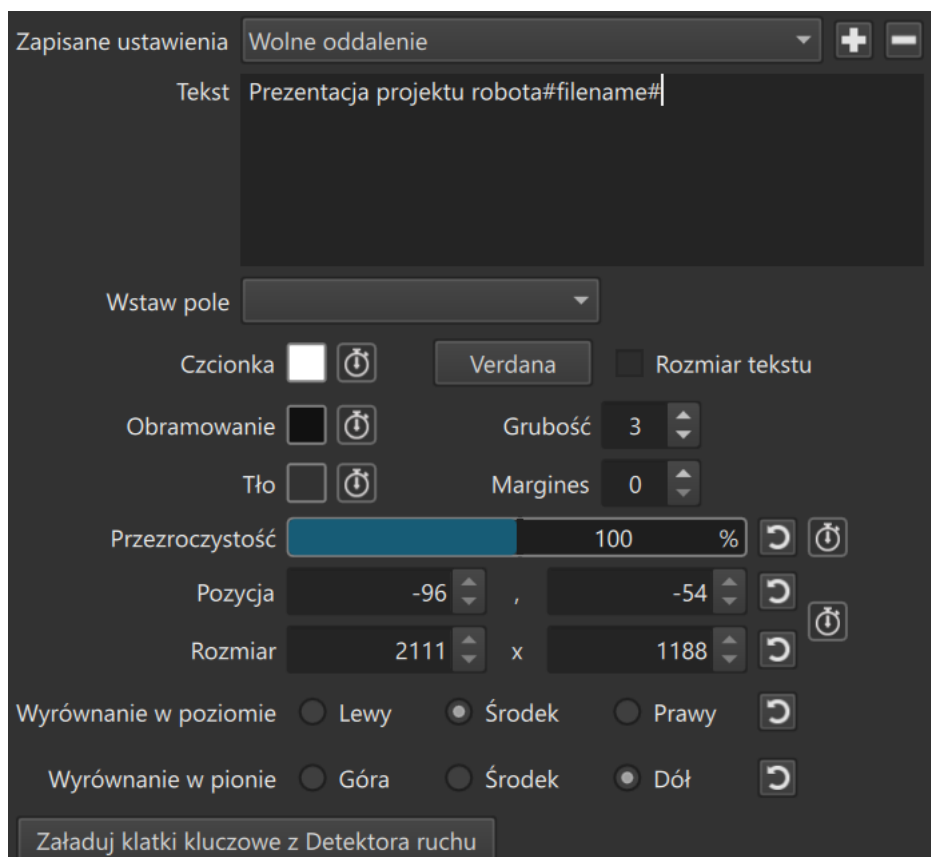
Rysunek 5. Dodawanie filtru.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

8. Dopasujecie wygląd napisu. Ustawcie jego rozmiar, kolor oraz pozycję tak, aby był czytelny i dobrze widoczny na tle filmu. Możecie wybrać większą czcionkę, kontrastowy kolor oraz umieścić napis w dolnej części ekranu, podobnie jak napisy w filmach. Wpiszcie przykładowy tekst związany z robotem, np. „Test robota”, „Mój robot w akcji” lub „Prezentacja projektu robota”. Poniżej macie opis najważniejszych opcji w filtrze *Tekst: Prosty*:

- **Zapisane ustawienia** – pozwalają wybrać gotowe style tekstu. Możecie szybko zastosować wcześniej zapisane ustawienia bez konieczności ponownego ustawiania wszystkich parametrów.
- **Tekst** – miejsce, w którym wpisujecie treść napisu wyświetlanego na filmie. Możecie wpisać własny tytuł, np. „Prezentacja projektu robota”.
- **Wstaw pole** – umożliwia dodanie automatycznych informacji, np. nazwy pliku lub czasu.
- **Czcionka** – pozwala zmienić styl liter. Możecie wybrać bardziej czytelną lub ozdobną czcionkę.
- **Rozmiar tekstu** – określa wielkość napisu. Większy rozmiar sprawia, że tekst jest bardziej widoczny.
- **Obramowanie** – dodaje obrys do liter, co może poprawić czytelność napisu na tle filmu.
- **Tło** – pozwala ustawić kolor tła pod tekstem, aby był lepiej widoczny.
- **Grubość** – określa grubość obramowania tekstu.
- **Margines** – ustawia odstęp między tekstem a jego tłem.
- **Przezroczystość** – umożliwia ustawienie stopnia widoczności napisu lub tła (im niższa wartość, tym bardziej przezroczysty element).
- **Pozycja** – określa miejsce, w którym tekst pojawi się na ekranie (możecie przesuwać go w poziomie i pionie).
- **Rozmiar (obszaru)** – ustawia wielkość pola, w którym znajduje się tekst.
- **Wyrównanie w poziomie** – pozwala ustawić tekst po lewej, na środku lub po prawej stronie.
- **Wyrównanie w pionie** – pozwala ustawić tekst na górze, środku lub dole ekranu.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Rysunek 6. Opcje konfiguracyjne filtru.

9. Dopasujcie wygląd napisu. Ustawcie jego rozmiar, kolor oraz pozycję, tak aby był czytelny i dobrze widoczny na tle filmu. Możecie umieścić go na dole ekranu, podobnie jak napisy w filmach.
10. Odtwórzcie fragment filmu przyciskiem *Play* i sprawdźcie, czy napis pojawia się w odpowiednim momencie oraz czy nie zasłania ważnych elementów obrazu.
11. Jeśli napis trwa zbyt długo lub zbyt krótko, najedźcie myszką na krawędź elementu na ścieżce **V2** i przeciągnijcie ją, aby dopasować czas wyświetlania.
12. Spróbujcie przesunąć element na osi czasu, przeciągając go w lewo lub w prawo. Dzięki temu możecie dokładnie ustawić moment, w którym pojawia się podpis.
13. Na zakończenie odtwórzcie cały film i sprawdźcie, jak współpracują ze sobą wszystkie ścieżki. Zwróćcie uwagę, czy napis poprawia zrozumienie filmu i czy cały projekt jest uporządkowany.

Praktyczne ćwiczenie:

Otwórz w programie Shotcut projekt, który wykonałeś na lekcji. Twoim zadaniem będzie jego poprawienie i uporządkowanie z wykorzystaniem kilku ścieżek. Sprawdź, czy główny film znajduje się na ścieżce V1. Następnie dodaj ścieżkę V2 i umieść na niej element z napisem,

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

który przygotowałeś wcześniej. Jeśli jeszcze go nie masz, dodaj element z Generator następnie Kolor i wykorzystaj go jako tło pod napis.

Dodaj do tego elementu tekst, korzystając z filtru Tekst: Prosty, i wpisz krótki tytuł, np. „Mój robot w akcji”. Dopasuj jego wielkość, kolor i położenie tak, aby był dobrze widoczny. Sprawdź, czy napis pojawia się w odpowiednim momencie filmu.

Odtwórz cały film i zwróć uwagę, czy wszystko jest uporządkowane: film na V1, dodatki na V2. Jeśli coś nie wygląda dobrze, przesun elementy lub zmień ich długość. Na koniec zapisz projekt.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 6: PixelPerfect Studio

Temat lekcji nr 27 - Dodawanie efektów wideo - filtry, korekcja kolorów oraz efekty specjalne?

Treści nauczania (wymagania szczegółowe - podstawa programowa) – I. 5); II. 3) a); II. 3) d);

Cel lekcji:

- Uczeń rozumie, na czym polega stosowanie filtrów i efektów wideo w programie Shotcut.
- Uczeń wyjaśnia, do czego służą podstawowe filtry (np. jasność, nasycenie, rozmycie) oraz jak wpływają na wygląd filmu.
- Uczeń potrafi zastosować wybrane efekty wideo do klipu oraz dopasować ich ustawienia, aby poprawić jakość i odbiór nagrania..

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- Uczeń przygotowuje i prezentuje rozwiązania problemów, posługując się podstawowymi aplikacjami (program do montażu wideo – Shotcut) na swoim komputerze.
- Uczeń gromadzi, porządkuje i selekcjonuje materiały cyfrowe (klipy wideo), potrzebne do realizacji projektu filmowego, oraz zapisuje efekty swojej pracy w postaci projektu.
- Uczeń modyfikuje wygląd materiału wideo, stosując podstawowe filtry i efekty (np. jasność, nasycenie, rozmycie), oraz dostosowuje ich ustawienia, dbając o czytelność i estetykę filmu..

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści programowe realizowane będą z wykorzystaniem uniwersalnych narzędzi dostępnych w systemach operacyjnych, takich jak eksplorator plików oraz menedżery

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

okien, co umożliwi uczniom sprawne zarządzanie materiałem wideo wykorzystywanym w projekcie.

- W miarę możliwości technicznych uczniowie będą korzystać z zapisu danych w środowiskach wirtualnych (chmurze) w celu gromadzenia, przechowywania oraz udostępniania projektów filmowych.
- Realizacja treści programowych odbywać się będzie w programie Shotcut, który posłuży jako podstawowe narzędzie do stosowania filtrów i efektów wideo, poprawy wyglądu obrazu oraz dostosowywania parametrów takich jak jasność, kolor czy ostrość filmu.

Uczeń:

- Wyjaśnia, na czym polega stosowanie filtrów i efektów wideo w programie Shotcut.
- Wskazuje, jakie efekty (np. jasność, nasycenie, rozmycie) wpływają na wygląd filmu i jego odbiór.
- Przedstawia znaczenie poprawy jakości obrazu oraz umiejętnego stosowania efektów wideo.
- Potrafi dodać filtr do klipu oraz dopasować jego ustawienia (np. jasność, kolory, rozmycie).
- Opisuje rolę filtrów jako narzędzi do modyfikowania obrazu i poprawy estetyki filmu..

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna,
- praca na komputerze,
- pogadanka.

Środki dydaktyczne

- zestaw PC z dostępem do Internetu oraz zainstalowanym programem Shotcut,
- rzutnik multimedialny,
- e-podręcznik.

Formy pracy

- Praca indywidualna.

Przebieg lekcji:

Faza I. Wprowadzenie

Podczas lekcji uczniowie zostaną zapoznani z zasadami stosowania filtrów i efektów wideo w programie Shotcut. Omówione zostanie znaczenie poprawy wyglądu obrazu, korekcji kolorów oraz wykorzystania efektów specjalnych w celu uatrakcyjnienia filmu. Nauczyciel przypomni

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

również podstawowe elementy programu Shotcut: okno podglądu, listę odtwarzania i oś czasu, ze szczególnym uwzględnieniem zakładki Filtry.

1. Nauczyciel rozpoczyna lekcję, wyjaśniając, że nawet dobrze nagrany film można jeszcze poprawić, zmieniając jego jasność, kolory lub dodając efekty specjalne. Podkreśla, że odpowiednio dobrane efekty mogą poprawić jakość obrazu i sprawić, że film będzie bardziej atrakcyjny.
2. Nauczyciel pyta uczniów, czy spotkali się z filmami, które mają zmienione kolory lub specjalne efekty (np. rozmycie, przyciemnienie), oraz zadaje pytanie, czy takie zmiany pomagają w odbiorze filmu, czy czasem mogą go pogorszyć.

Faza II. Część zasadnicza - realizacja

1. Nauczyciel wprowadza uczniów w temat efektów wideo, wyjaśniając, że dzięki filtrom można poprawić wygląd filmu, zmienić jego kolory, jasność oraz dodać efekty specjalne. Podkreśla, że filtry powinny być stosowane z umiarem, aby nie pogorszyć odbioru nagrania.
2. Dyskusja kierowana: Nauczyciel zadaje pytania, jakie zmiany można wprowadzić w filmie (np. rozjaśnienie, poprawa kolorów, rozmycie) oraz czy wszystkie efekty zawsze poprawiają wygląd filmu. Uczniowie podają przykłady i zastanawiają się, kiedy warto używać efektów.
3. Nauczyciel omawia podstawowe filtry wideo, takie jak Jasność, Nasycenie, Korekcja kolorów oraz Rozmycie. Wyjaśnia, do czego służą i jak wpływają na wygląd obrazu. Zwraca uwagę na konieczność delikatnego dopasowywania ustawień.
4. Pokaz nauczyciela: Nauczyciel uruchamia program Shotcut i prezentuje, w jaki sposób dodać filtr do klipu. Pokazuje, jak zaznaczyć klip, przejść do zakładki Filtry, dodać filtr Jasność oraz Nasycenie i zmienić ich ustawienia. Następnie demonstruje dodanie efektu specjalnego (np. Rozmycie: Gaussowskie) oraz pokazuje, jak zmienia się obraz w oknie podglądu.
5. Część praktyczna: Uczniowie uruchamiają program Shotcut i pod kierunkiem nauczyciela wykonują ćwiczenie polegające na dodaniu filtrów do klipu. Dodają filtr Jasność, Nasycenie oraz jeden efekt specjalny (Rozmycie), a następnie dopasowują ich ustawienia. Odtwarzają film i sprawdzają, czy zastosowane efekty poprawiają wygląd nagrania..

Faza III. Podsumowanie

Uczniowie prezentują efekty swojej pracy, omawiając zastosowane filtry oraz wyjaśniając, w jaki sposób zmienili wygląd filmu (np. poprawili jasność, kolory lub dodali efekt rozmycia). Nauczyciel podsumowuje lekcję, zwracając uwagę na znaczenie umiejętnego stosowania efektów wideo, poprawy jakości obrazu oraz korzystania z zakładki Filtry w programie Shotcut. Następnie nauczyciel dokonuje oceny pracy uczniów zgodnie z poniższymi kryteriami:

Ocena dopuszczająca:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

– uczeń potrafi wskazać podstawowe elementy interfejsu programu Shotcut (okno podglądu, lista odtwarzania, oś czasu) oraz odnaleźć zakładkę Filtry.

Ocena dostateczna:

– uczeń rozpoznaje podstawowe elementy interfejsu Shotcut i potrafi samodzielnie dodać prosty filtr do klipu (np. jasność).

Ocena dobra:

– uczeń poprawnie opisuje działanie podstawowych filtrów oraz potrafi zastosować kilka efektów (np. jasność, nasycenie) i dopasować ich ustawienia.

Ocena bardzo dobra:

– uczeń szczegółowo omawia wpływ filtrów na wygląd filmu, potrafi zastosować różne efekty (np. korekcja kolorów, rozmycie) i świadomie dopasować ich parametry.

Ocena celująca:

– uczeń samodzielnie i twórczo wykorzystuje filtry w projekcie, potrafi uzasadnić swoje decyzje dotyczące wyboru efektów oraz proponuje własne rozwiązania poprawiające jakość i estetykę filmu.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, np.:

1. Do czego służą filtry w programie Shotcut:
 - a. do zapisywania projektu
 - b. do zapisywania projektu
 - c. **do zmiany wyglądu filmu**
 - d. do zamykania programu
2. Który filtr pozwala rozjaśnić lub przyciemnić obraz:
 - a. Nasycenie
 - b. **Jasność**
 - c. Rozmycie
 - d. Tekst
3. Co robi filtr Nasycenie:
 - a. zmienia głośność dźwięku
 - b. rozmywa obraz
 - c. **zmienia intensywność kolorów**
 - d. przycina film

Faza V. Dodatkowe, praca domowa

Zadanie domowe (dla chętnych).

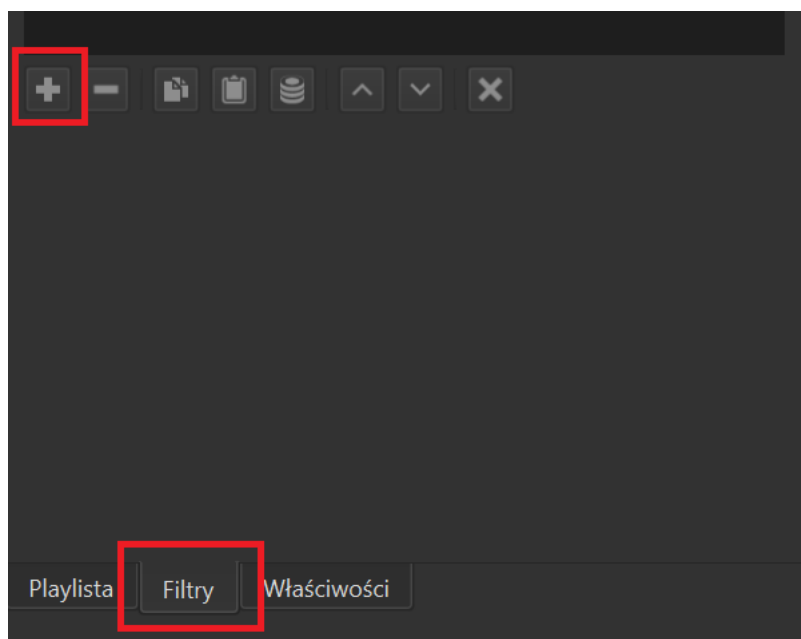
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Otwórz w programie Shotcut projekt wykonany na lekcji lub inny krótki klip wideo. Twoim zadaniem jest poprawić jego wygląd, wykorzystując poznane filtry. Dodaj filtr **Jasność** i dopasuj go tak, aby obraz był czytelny. Następnie dodaj filtr **Nasycenie** i ustaw kolory tak, aby wyglądały naturalnie.

Na końcu przetestuj różne efekty rozmycia. Dodaj filtr z grupy **Rozmycie** (np. Rozmycie: Gaussowskie) i sprawdź, jak działa. Następnie wybierz inne rozmycie z listy i porównaj efekty. Zastanów się, które rozmycie wygląda najlepiej i najlepiej pasuje do Twojego filmu.

Odtwórz film i sprawdź, czy zastosowane efekty poprawiają jego wygląd. Jeśli coś nie wygląda dobrze, zmień ustawienia filtrów. Zapisz projekt i przygotuj się do pokazania efektu na kolejnej lekcji.

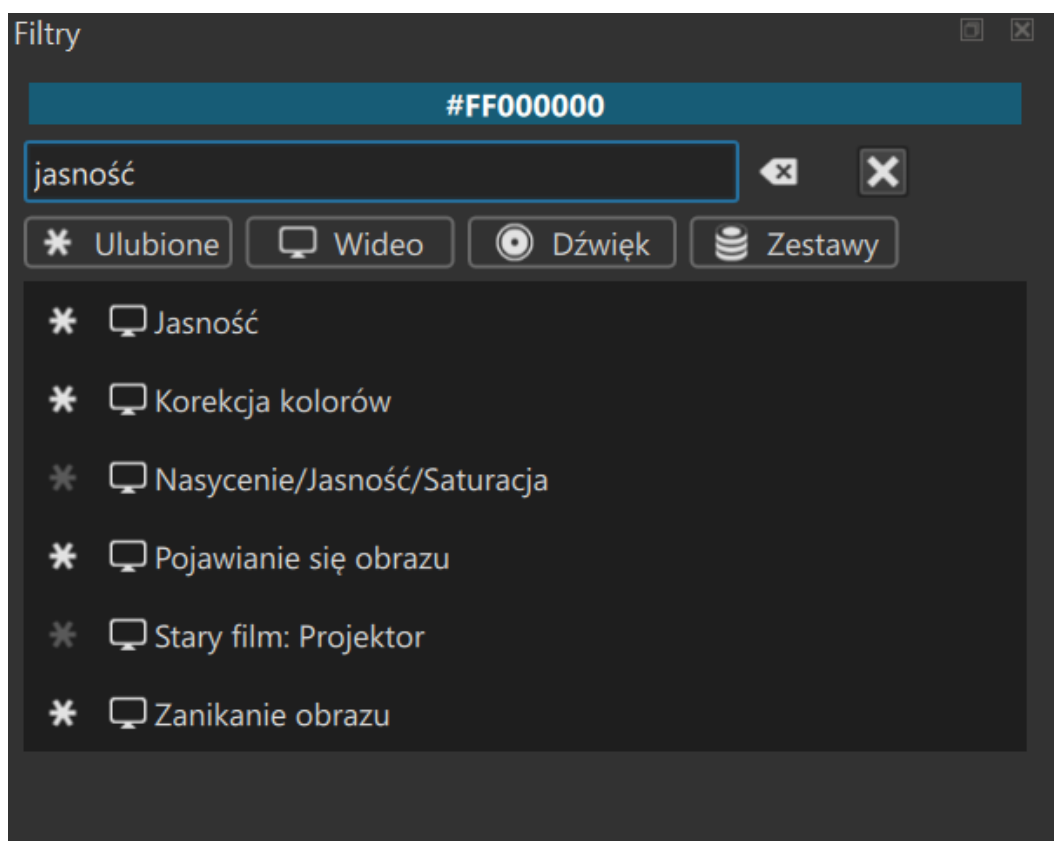
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Rysunek 2. Dodawanie nowego filtra.

4. Z listy wybierzcie kategorię *Wideo*. Jeśli nie widzicie potrzebnego efektu, możecie skorzystać z pola wyszukiwania i wpisać nazwę filtra, np. „jasność”, „kolor” albo „rozmycie”. Na początek dodajcie filtr odpowiedzialny za poprawę jasności lub kontrastu obrazu. Wybierzcie filtr związany z jasnością i kontrastem, a następnie delikatnie zmieńcie ustawienia. Sprawdźcie w oknie podglądu, jak zmienia się wygląd filmu.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

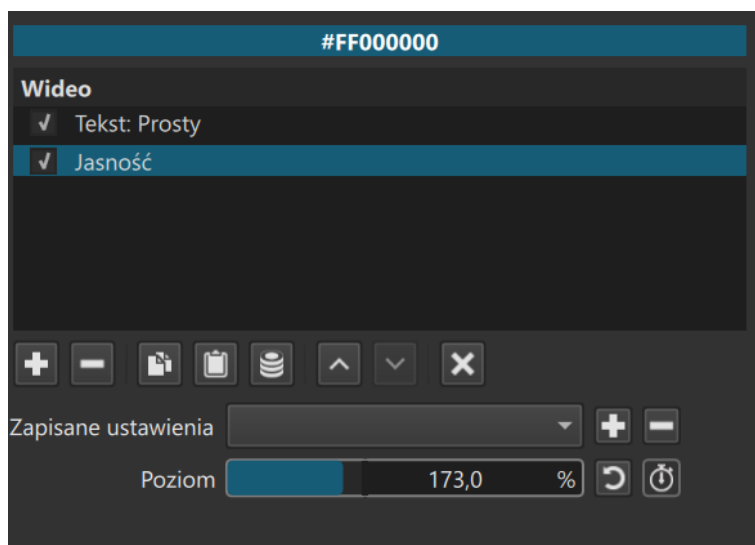


Rysunek 3. Wybór filtra w dla kategorii wideo.

5. Na początek dodajcie filtr odpowiedzialny za poprawę jasności lub kontrastu obrazu. Po dodaniu filtra *Jasność* zobaczycie panel z ustawieniami, które pozwalają zmieniać wygląd obrazu:
 - **Poziom** – to najważniejsze ustawienie. Odpowiada za jasność obrazu. Przesunięcie suwaka w prawo powoduje rozjaśnienie filmu (obraz staje się jaśniejszy). Przesunięcie w lewo powoduje przyciemnienie filmu (obraz staje się ciemniejszy).
 - **Zapisane ustawienia** – pozwalają zapisać własne ustawienia filtra lub wybrać wcześniej zapisane. Dzięki temu możecie szybko zastosować ten sam efekt w innych fragmentach filmu.
 - **Ikony obok suwaka** – umożliwiają szybkie przywrócenie ustawień domyślnych lub cofnięcie zmian.

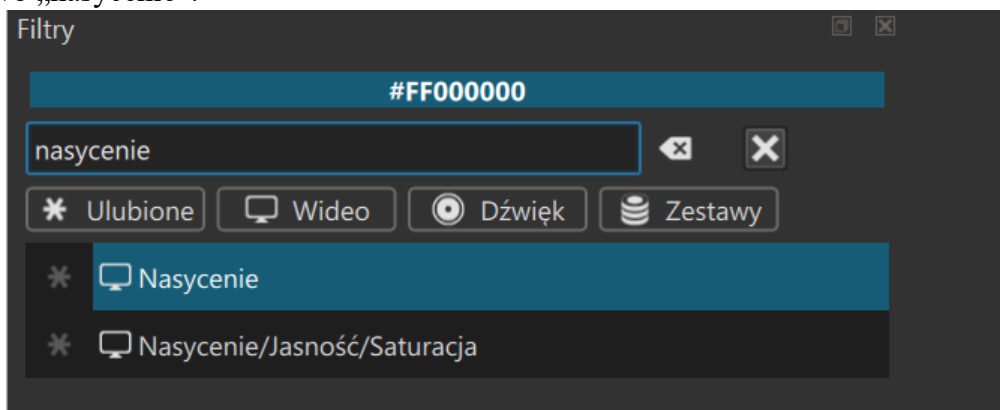
Sprawdźcie w oknie podglądu, jak zmienia się obraz podczas przesuwania suwaka, i wybierzcie takie ustawienie, które najlepiej poprawia widoczność filmu.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Rysunek 4. Menu ustawień filtra poprawiającego jasność klipu.

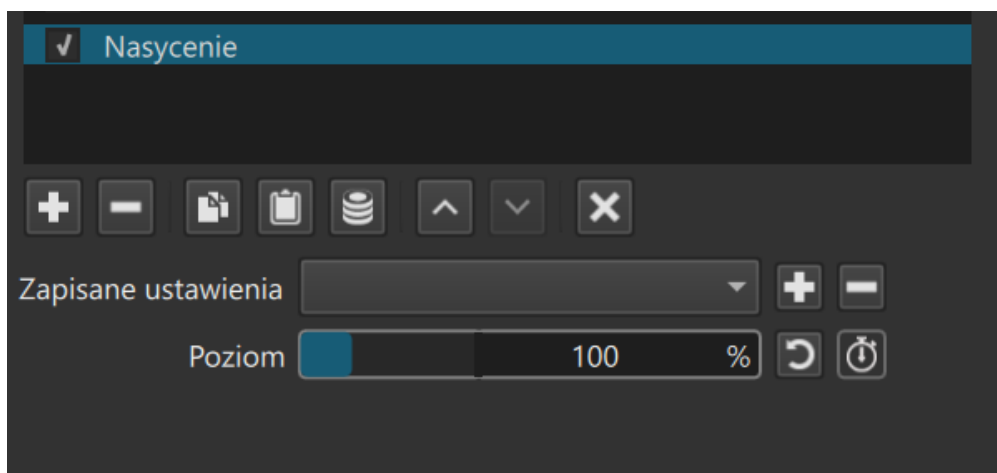
6. Teraz zajmiemy się poprawą kolorów filmu, czyli tzw. korekcją kolorów. Dzięki temu obraz może wyglądać bardziej wyraźnie i naturalnie. Czasem nagranie jest zbyt blade lub ma mało intensywne kolory – wtedy warto użyć odpowiedniego filtra. Zaznaczcie klip na osi czasu, przejdźcie do zakładki *Filtry*, kliknijcie przycisk + i w kategorii *Wideo* wybierzcie filtr *Nasycenie*. Jeśli nie widzicie tej opcji, możecie wpisać w wyszukiwarkę słowo „nasycenie”.



Rysunek 5. Wybór filtra "nasycenia".

7. Po dodaniu filtra delikatnie przesuwajcie suwak. Zwiększenie wartości sprawi, że kolory będą bardziej intensywne, a zmniejszenie – że obraz stanie się bardziej stonowany. Obserwujcie zmiany w oknie podglądu i ustawcie taką wartość, aby obraz wyglądał naturalnie i był przyjemny w odbiorze.

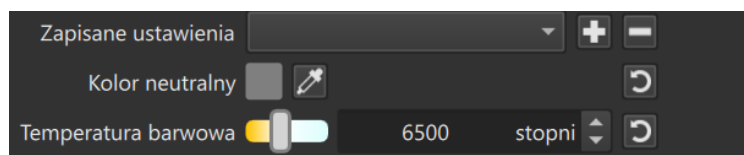
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Rysunek 6. Aktywna opcja zmiany parametrów nasycenia klipu.

8. Teraz spróbujcie poprawić wygląd obrazu, korzystając z filtra *Balans bieli*. Zaznaczcie klip na osi czasu, przejdźcie do zakładki *Filtry*, kliknijcie przycisk + i w kategorii *Wideo* wybierzcie *Balans bieli* (możecie użyć wyszukiwarki i wpisać „kolor”). Po dodaniu filtra zobaczycie kilka opcji, które pozwalają zmienić wygląd obrazu:
- **Zapisane ustawienia** – pozwalają wybrać gotowe lub wcześniej zapisane ustawienia filtra. Możecie dzięki temu szybko zastosować sprawdzony efekt.
 - **Kolor neutralny** – służy do wskazania koloru, który w rzeczywistości powinien być biały lub szary. Kliknijcie ikonę pipety i wybierzcie taki punkt na obrazie. Program automatycznie dopasuje kolory, aby wyglądały naturalnie.
 - **Temperatura barwowa** – odpowiada za „ciepło” lub „chłód” obrazu. Przesunięcie suwaka w stronę koloru żółtego sprawia, że obraz staje się cieplejszy (bardziej żółty/pomarańczowy). Przesunięcie w stronę niebieskiego powoduje, że obraz staje się chłodniejszy (bardziej niebieski).

Wartość podana w stopniach (np. 6500) oznacza temperaturę barwową światła. Zmieniajcie ją delikatnie i obserwujcie efekt w podglądzie, aby uzyskać naturalny wygląd obrazu.



Rysunek 7. Aktywne opcje filtra "Balans bieli".

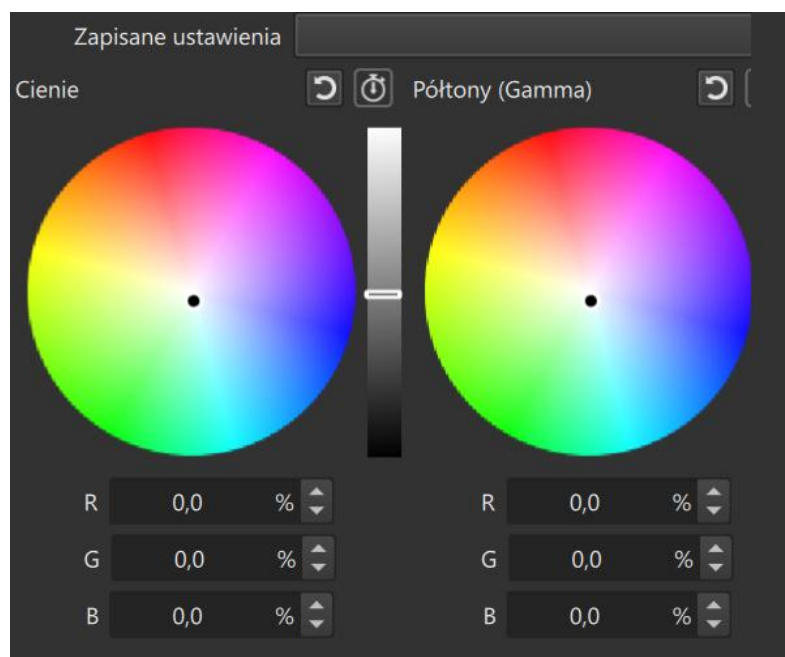
9. Teraz spróbujcie poprawić wygląd obrazu, korzystając z filtra *Korekcja kolorów*. Zaznaczcie klip na osi czasu, przejdźcie do zakładki *Filtry*, kliknijcie przycisk + i w kategorii *Wideo* wybierzcie *Korekcja kolorów* (możecie użyć wyszukiwarki i wpisać

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

„kolor”). Po dodaniu filtra zobaczycie koła kolorów oraz dodatkowe ustawienia, które pozwalają dokładniej zmieniać wygląd obrazu:

- **Cienie** – lewe koło kolorów odpowiada za najciemniejsze fragmenty obrazu. Przesuwając punkt na kole, możecie zmienić kolor cieni (np. dodać lekko niebieski lub czerwony odcień). Używajcie tej opcji delikatnie, aby nie zniekształcić obrazu.
- **Półtony (Gamma)** – środkowa część odpowiada za średnie tony obrazu, czyli większość widocznych elementów. Zmiana w tym miejscu wpływa na ogólny wygląd filmu. Możecie lekko poprawić kolory skóry, tła lub obiektów.
- **Jasność (pionowy suwak)** – pozwala rozjaśnić lub przyciemnić średnie tony obrazu. Przesunięcie w górę – jaśniejszy obraz. Przesunięcie w dół – ciemniejszy obraz.
- **R, G, B (wartości procentowe)** – umożliwiają dokładne ustawienie poziomu kolorów: R (czerwony), G (zielony), B (niebieski). Możecie ręcznie wpisać wartości lub zmieniać je strzałkami.

Podczas pracy zmieniajcie ustawienia bardzo ostrożnie i obserwujcie efekt w oknie podglądu. Celem jest poprawa kolorów i naturalny wygląd obrazu, a nie przesadne efekty.

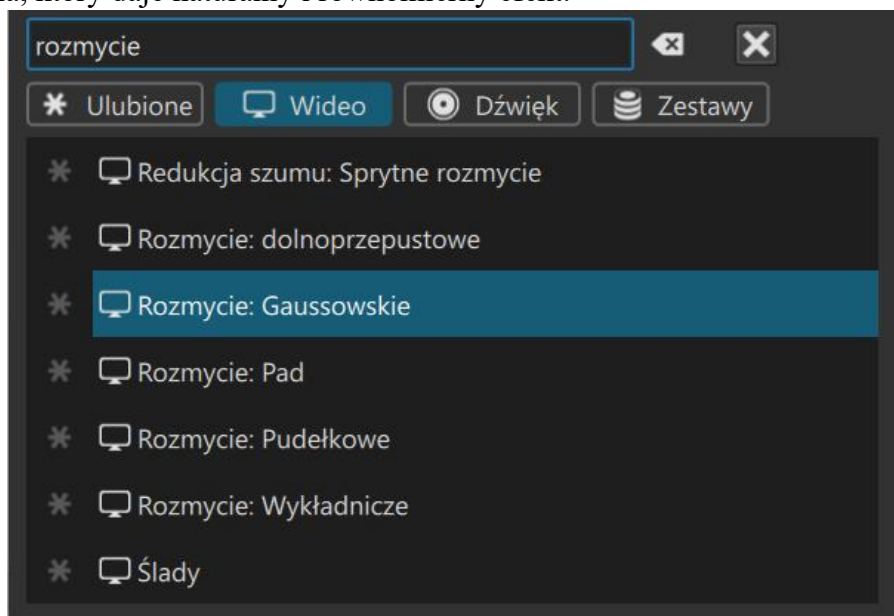


Rysunek 8. Aktywne opcje filtra "Korekcja kolorów".

10. Odtwórzcie fragment filmu przyciskiem *Play* i porównajcie efekt przed oraz po zmianach. Zastanówcie się, czy obraz jest jaśniejszy, bardziej wyraźny i przyjemniejszy w odbiorze. Modyfikujecie ustawienia tak żeby poprawić wygląd obrazu.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

11. Następnie dodajcie prosty efekt specjalny, który uatrakcyjni wygląd filmu. Zaznaczcie klip na osi czasu, przejdźcie do zakładki *Filtry*, kliknijcie przycisk + i w kategorii *Wideo* wybierzcie filtr *Rozmycie: Gaussowskie*. Jest to jeden z najczęściej używanych efektów rozmycia, który daje naturalny i równomierny efekt.



Rysunek 9. Opcje wyboru dla filtra rozmycie.

12. Po dodaniu filtra zobaczycie kilka prostych opcji, które pozwalają sterować efektem rozmycia:

- **Ilość** – to najważniejsze ustawienie. Określa siłę rozmycia obrazu. Niska wartość – lekkie, delikatne rozmycie. Wysoka wartość – mocne rozmycie, obraz staje się mniej wyraźny. Zmieniajcie tę wartość powoli i obserwujcie efekt w podglądzie.
- **Blur alpha** – odpowiada za rozmycie przezroczystości (jeśli element ją posiada). Włączona opcja – rozmywane są także krawędzie elementów z przezroczystością. Wyłączona opcja – rozmycie dotyczy tylko obrazu, bez wpływu na przezroczystość.
- **Zapisane ustawienia** – pozwalają zapisać własne ustawienia filtra lub użyć wcześniej zapisanych.

Podczas pracy starajcie się używać niewielkiego rozmycia, aby efekt był subtelny i nie utrudniał oglądania filmu.

13. Sprawdźcie, jak działa kilka filtrów jednocześnie. Możecie połączyć np. korekcję kolorów z lekkim rozjaśnieniem obrazu. Odtwórzcie film i oceńcie, czy połączenie efektów wygląda dobrze.
14. Jeśli jakiś filtr nie pasuje do filmu, usuńcie go z listy filtrów lub zmieńcie jego ustawienia. Dzięki temu nauczycie się świadomie dobierać efekty do projektu.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

15. Na zakończenie odtwórzcie cały film i sprawdźcie, czy zastosowane efekty poprawiają wygląd materiału. Zwróćcie uwagę, czy obraz jest czytelny, kolory wyglądają dobrze, a efekty specjalne pasują do treści filmu.
16. Jeśli wszystko wygląda poprawnie, zapiszcie projekt. Wybierzcie z menu *Plik*, następnie *Zapisz jako*, nadajcie projektowi nazwę i zapiszcie go w wybranym folderze.

Praktyczne ćwiczenie:

Otwórz w programie Shotcut projekt, który wykonałeś na lekcji. Twoim zadaniem będzie jego ulepszenie poprzez dodanie efektów wideo. Sprawdź, czy główny film znajduje się na ścieżce V1 oraz czy dodatkowe elementy (np. napisy) są na ścieżce V2.

Zaznacz klip wideo i przejdź do zakładki Filtry. Dodaj filtr Jasność i delikatnie popraw wygląd obrazu. Następnie dodaj filtr Nasycenie i sprawdź, jak zmienia się intensywność kolorów. Spróbuj ustawić takie wartości, aby obraz wyglądał naturalnie.

Kolejnym krokiem jest dodanie jednego efektu specjalnego. Wybierz filtr Rozmycie: Gaussowskie i ustaw jego wartość tak, aby efekt był widoczny, ale nie przeszkadzał w oglądaniu filmu.

Na koniec wybierz jeden z dodatkowych filtrów, np. Korekcja kolorów lub Balans bieli, i sprawdź, jak wpływa on na wygląd nagrania. Spróbuj różnych ustawień i wybierz takie, które najbardziej pasują do Twojego filmu.

Odtwórz cały film i sprawdź, czy efekty poprawiają jego wygląd. Jeśli coś nie wygląda dobrze, zmień ustawienia filtrów lub usuń wybrany efekt. Na koniec zapisz projekt.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 6: PixelPerfect Studio

Temat lekcji nr 28 - Tworzenie animowanych przejść pomiędzy klipami.

Treści nauczania (wymagania szczegółowe - podstawa programowa) – I. 5); II. 3) a); II. 3) d);

Cel lekcji:

- Uczeń rozumie, na czym polega wykorzystanie animowanych grafik (GIF) oraz przejść w projekcie filmowym.
- Uczeń wyjaśnia, do czego służą animowane napisy oraz efekty przejścia (np. zanikanie) i jak wpływają na odbiór filmu.
- Uczeń potrafi stworzyć prostą animację w programie *ScreenToGif*, wstawić ją do Shotcut oraz dopasować przejście między elementami filmu.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- Uczeń przygotowuje i prezentuje rozwiązania problemów, posługując się podstawowymi aplikacjami (program do tworzenia animacji – ScreenToGif oraz program do montażu wideo – Shotcut) na swoim komputerze.
- Uczeń gromadzi, porządkuje i selekcjonuje materiały cyfrowe (klipy wideo oraz animacje GIF), potrzebne do realizacji projektu filmowego, oraz zapisuje efekty swojej pracy w postaci projektu.
- Uczeń wzbogaca materiał wideo o dodatkowe elementy (animowane napisy, przejścia), wykorzystując podstawowe narzędzia oraz dostosowuje ich ustawienia, dbając o czytelność i estetykę filmu.

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści programowe realizowane będą z wykorzystaniem uniwersalnych narzędzi dostępnych w systemach operacyjnych, takich jak eksplorator plików oraz menedżery

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

okien, co umożliwi uczniom sprawne zarządzanie materiałami (klipami wideo oraz plikami GIF) wykorzystywanymi w projekcie.

- W miarę możliwości technicznych uczniowie będą korzystać z zapisu danych w środowiskach wirtualnych (chmurze) w celu gromadzenia, przechowywania oraz udostępniania projektów filmowych.
- Realizacja treści programowych odbywać się będzie w programach *ScreenToGif* oraz *Shotcut*, które posłużą jako podstawowe narzędzia do tworzenia animowanych grafik oraz ich wykorzystania w projekcie filmowym, a także do stosowania przejść i organizowania materiału na osi czasu..

Uczeń:

- Wyjaśnia, na czym polega tworzenie animowanych grafik (GIF) oraz ich wykorzystanie w projekcie filmowym.
- Wskazuje, jakie elementy (np. animowany napis, przejścia) wpływają na odbiór filmu i jego atrakcyjność.
- Przedstawia znaczenie stosowania animacji i przejść w budowaniu spójnej formy filmowej.
- Potrafi stworzyć prostą animację w programie *ScreenToGif* oraz wstawić ją do programu *Shotcut*.
- Opisuje rolę przejść (np. zanikania) jako sposobu na płynne łączenie elementów filmu.

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna,
- praca na komputerze,
- pogadanka.

Środki dydaktyczne

- zestaw PC z dostępem do Internetu oraz zainstalowanym programem *Shotcut* i *ScreenToGif*,
- rzutnik multimedialny,
- e-podręcznik.

Formy pracy

- Praca indywidualna.

Przebieg lekcji:

Faza I. Wprowadzenie

Podczas lekcji uczniowie zostaną zapoznani z zasadami tworzenia animowanych grafik oraz stosowania przejść w programie *Shotcut*. Omówione zostanie znaczenie wykorzystania animowanych napisów (GIF) jako elementów wprowadzających lub kończących film oraz

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

wpływu przejść na płynność i odbiór materiału. Nauczyciel przedstawi także program *ScreenToGif* jako narzędzie do tworzenia prostych animacji oraz przypomni podstawowe elementy programu Shotcut: okno podglądu, listę odtwarzania i oś czasu.

1. Nauczyciel rozpoczyna lekcję, wyjaśniając, że film można uatrakcyjnić, dodając animowane elementy, takie jak napisy wprowadzające lub końcowe. Podkreśla, że takie dodatki pomagają uporządkować treść i przyciągają uwagę widza.
2. Nauczyciel pyta uczniów, czy spotkali się z animowanymi napisami w filmach (np. na początku filmu lub na końcu – „Koniec”, „Dziękuję za uwagę”) oraz zadaje pytanie, czy płynne przejścia między elementami wpływają na to, jak odbieramy film.

Faza II. Część zasadnicza - realizacja

1. Nauczyciel wprowadza uczniów w temat animowanych grafik, wyjaśniając, że oprócz samego filmu można dodać elementy takie jak animowane napisy (GIF), które pełnią rolę wprowadzenia lub zakończenia. Podkreśla, że takie elementy pomagają uporządkować film i zwiększają jego atrakcyjność.
2. Dyskusja kierowana: Nauczyciel zadaje pytania, gdzie można wykorzystać animowane napisy (np. początek filmu, zakończenie, podpisy) oraz czy ich obecność wpływa na odbiór filmu. Uczniowie podają przykłady i zastanawiają się, kiedy warto je stosować.
3. Nauczyciel omawia podstawy tworzenia animacji w programie *ScreenToGif*, pokazując, jak nagrać prosty ruch (np. rysowanie napisu) oraz wyjaśnia, że animacja składa się z kolejnych klatek. Zwraca uwagę na czytelność napisu i odpowiednie tempo animacji.
4. Pokaz nauczyciela: Nauczyciel uruchamia program *ScreenToGif* i prezentuje, w jaki sposób nagrać prostą animację napisu oraz zapisać ją jako plik GIF. Następnie uruchamia program Shotcut i pokazuje, jak wczytać GIF, umieścić go na osi czasu oraz dopasować jego długość. Demonstruje także dodanie przejścia poprzez zastosowanie filtrów zanikania i pokazuje efekt w oknie podglądu.
5. Część praktyczna: Uczniowie uruchamiają program *ScreenToGif* i tworzą prostą animację napisu (np. „Mój robot” lub „Koniec filmu”). Następnie zapisują ją jako plik GIF i wstawiają do programu Shotcut. Umieszczają animację na początku lub końcu filmu, a następnie dodają przejście (zanikanie), aby połączyć ją płynnie z klipem wideo. Odtwarzają film i sprawdzają efekt końcowy..

Faza III. Podsumowanie

Uczniowie prezentują efekty swojej pracy, omawiając stworzone animacje oraz wyjaśniając, w jaki sposób wykorzystali je w filmie (np. jako stronę tytułową lub zakończenie). Opisują także, jak wykonali przejście między animacją a klipem wideo oraz czy efekt wygląda płynnie i naturalnie. Nauczyciel podsumowuje lekcję, zwracając uwagę na znaczenie stosowania animowanych elementów oraz przejść w tworzeniu spójnego i atrakcyjnego filmu. Następnie nauczyciel dokonuje oceny pracy uczniów zgodnie z poniższymi kryteriami:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Ocena dopuszczająca:

– uczeń potrafi wskazać podstawowe elementy interfejsu programu Shotcut oraz programu *ScreenToGif*.

Ocena dostateczna:

– uczeń rozpoznaje podstawowe elementy interfejsu i potrafi samodzielnie wczytać plik GIF do projektu oraz odtworzyć film.

Ocena dobra:

– uczeń poprawnie opisuje sposób tworzenia animacji oraz potrafi wstawić GIF do filmu i dopasować jego długość.

Ocena bardzo dobra:

– uczeń szczegółowo omawia proces tworzenia animacji i przejścia, potrafi zastosować efekt zanikania oraz dopasować go do filmu.

Ocena celująca:

– uczeń samodzielnie i twórczo wykorzystuje animacje w projekcie filmowym, potrafi uzasadnić swoje decyzje oraz proponuje własne rozwiązania poprawiające jakość i atrakcyjność filmu.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, np.:

1. Do czego służy program *ScreenToGif*:
 - a. do montażu filmu
 - b. do dodawania dźwięku
 - c. **do tworzenia animowanych grafik**
 - d. do zapisywania projektu w Shotcut
2. Gdzie należy umieścić plik GIF, aby był widoczny w filmie:
 - a. w oknie podglądu
 - b. w pasku menu
 - c. **na osi czasu (np. ścieżka V2)**
 - d. w ustawieniach programu
3. Co daje zastosowanie efektu zanikania:
 - a. przyspiesza film
 - b. zmienia kolor obrazu
 - c. **pozwala płynnie przejść między elementami filmu**
 - d. usuwa fragment filmu

Faza V. Dodatkowe, praca domowa

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Zadanie domowe (dla chętnych).

Otwórz w programie Shotcut projekt wykonany na lekcji lub inny krótki klip wideo. Twoim zadaniem jest urozmaicić film, dodając animowaną grafikę oraz przejście.

Przygotuj w programie *ScreenToGif* prostą animację napisu, np. „Koniec filmu”, „Dziękuję za uwagę” lub własny tytuł. Zapisz ją jako plik GIF.

Następnie wczytaj plik GIF do Shotcut i umieść go na końcu filmu (na ścieżce V2). Dopasuj jego długość, aby był widoczny przez kilka sekund.

Dodaj przejście między filmem a animacją.

Odtwórz cały film i sprawdź, czy efekt wygląda naturalnie. Jeśli coś nie wygląda dobrze, popraw ustawienia przejścia lub długość animacji. Na koniec zapisz projekt i przygotuj się do pokazania efektu na kolejnej lekcji.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1 – Tworzenie animowanych przejść pomiędzy klipami.

Wprowadzenie!

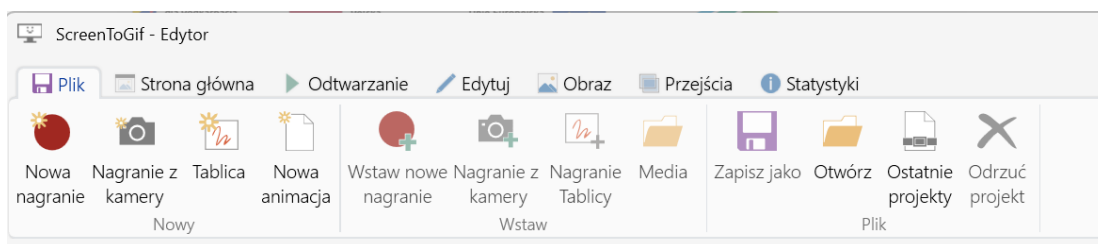
Dziś zajmiemy się tym, w jaki sposób uatrakcyjnić film, wykorzystując animowane grafiki oraz płynne przejścia między elementami. Sam klip wideo często nie wystarcza, aby przyciągnąć uwagę widza lub wprowadzić go w temat filmu. Dlatego wykorzystuje się dodatkowe elementy, takie jak animowane napisy (GIF), które mogą pełnić rolę strony tytułowej lub zakończenia.

Podczas lekcji zobaczymy, jak stworzyć prostą animację w programie *ScreenToGif*, a następnie jak wstawić ją do programu Shotcut. Nauczymy się również, jak wykonać płynne przejście między animacją a filmem, tak aby całość wyglądała spójnie i estetycznie. Dzięki temu przekonacie się, że nawet prosty efekt może znacząco poprawić odbiór filmu.

Scenariusz

1. Uruchomcie program *ScreenToGif*. Jest to narzędzie, które pozwala nagrywać fragment ekranu i tworzyć z niego animowane grafiki (GIF), które możecie później wykorzystać w filmie, np. jako stronę tytułową. Zapoznajcie się z menu programu – znajdziecie tam opcje takie jak:
 - **Nowe nagranie** – rozpoczyna nowe nagrywanie ekranu. Po kliknięciu pojawi się ramka, którą ustawicie na wybranym obszarze.
 - **Nagranie z kamery** – pozwala nagrać obraz z kamery internetowej i zapisać go jako animację.
 - **Tablica** – otwiera wirtualną tablicę, na której możecie rysować i tworzyć proste animacje (np. rysunek robota krok po kroku).
 - **Nowa animacja** – umożliwia tworzenie animacji ręcznie, klatka po klatce, bez nagrywania ekranu.
 - **Wstaw nowe nagranie** – dodaje kolejne nagranie do już istniejącego projektu.
 - **Nagranie z kamery (wstaw)** – pozwala dodać materiał z kamery do aktualnej animacji.
 - **Nagranie Tablicy** – umożliwia nagranie tego, co rysujecie na tablicy.
 - **Media** – pozwala wczytać gotowe pliki (np. obrazy lub GIF-y) do programu.
 - **Zapisz jako** – zapisuje gotową animację, najczęściej jako plik GIF.
 - **Otwórz** – otwiera wcześniej zapisany projekt lub plik.
 - **Ostatnie projekty** – pokazuje listę ostatnio otwieranych plików.
 - **Odrzuć projekt** – zamyka aktualny projekt bez zapisywania zmian.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Rysunek 1. Główne menu programu ScreenToGif

2. W programie *ScreenToGif* wybierzcie opcję *Tablica*. Otworzy się proste okno do rysowania i tworzenia animacji. To właśnie tutaj przygotujecie animowaną grafikę, która później zostanie wykorzystana w filmie jako strona tytułowa. Zanim zaczniecie tworzyć animację, zapoznajcie się z oknem *Nagrywanie tablicy*. Znajdują się w nim narzędzia:

- **Ołówek** – pozwala rysować linie i kształty odręcznie.
- **Gumka** – umożliwia usuwanie narysowanych elementów.
- **Zaznaczenie (prostokąt)** – pozwala zaznaczyć fragment rysunku.
- **Kolor** – wybór koloru rysowania (np. czarny, czerwony).
- **Grubość linii** – określa, jak gruba będzie linia (np. 10 px).
- **Dopasuj do krzywej** – wygładza rysowane linie.
- **Zakreślacz** – pozwala rysować półprzezroczyste linie, jak markerem.
- **FPS (klatki na sekundę)** – określa płynność animacji (np. 15 fps – standardowe ustawienie).
- **Rozmiar (px)** – pokazuje wielkość obszaru nagrywania (np. 900 × 261 pikseli).
- **Automatyczne nagrywanie** – po włączeniu program sam nagrywa ruchy bez konieczności ciągłego klikania.
- **Nagrywaj (czerwony przycisk)** – rozpoczyna nagrywanie animacji.
- **Stop** – zatrzymuje nagrywanie i przenosi do edytora.

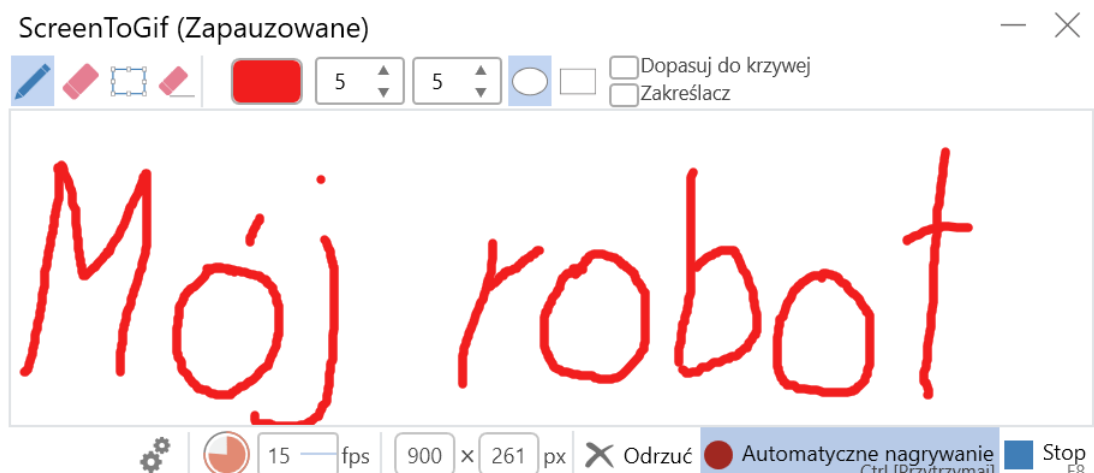
Dzięki tym opcjom możecie tworzyć proste animacje, np. pojawiający się napis lub rysunek, które później wykorzystacie w filmie.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Rysunek 2. Menu opcji Tablica.

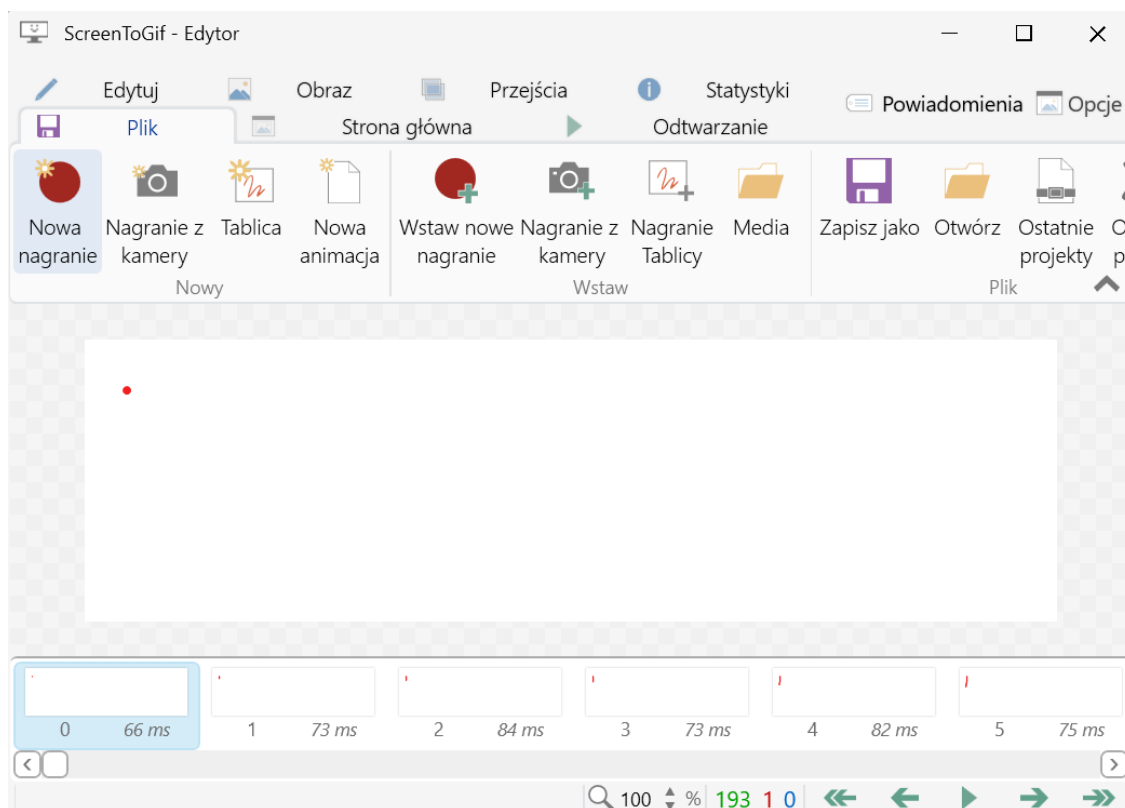
3. W oknie *Nagrywanie tablicy* wybierzcie narzędzie **Ołówek** (ikona po lewej stronie). Następnie ustawcie kolor (np. czerwony) oraz grubość linii (np. 5). W obszarze roboczym narysujcie prosty napis ręcznie, np. „Mój robot”. Jeśli chcecie, możecie dodatkowo dorysować prosty element, np. linię pod tekstem lub ramkę. Starajcie się rysować powoli – każdy ruch będzie nagrany i stanie się częścią animacji. Dzięki temu animacja będzie wyglądała tak, jakby napis pojawiał się stopniowo na ekranie. Po zakończeniu kliknijcie przycisk *Stop*.



Rysunek 3. Tworzenie animowanego napisu.

4. Po kliknięciu *Stop* zostaniecie przeniesieni do edytora animacji – dokładnie takiego jak na ekranie. Na dole widzicie małe prostokąty – to **klatki animacji**. Każda klatka pokazuje fragment tego, co narysowaliście. Im więcej klatek, tym płynniejsza animacja.

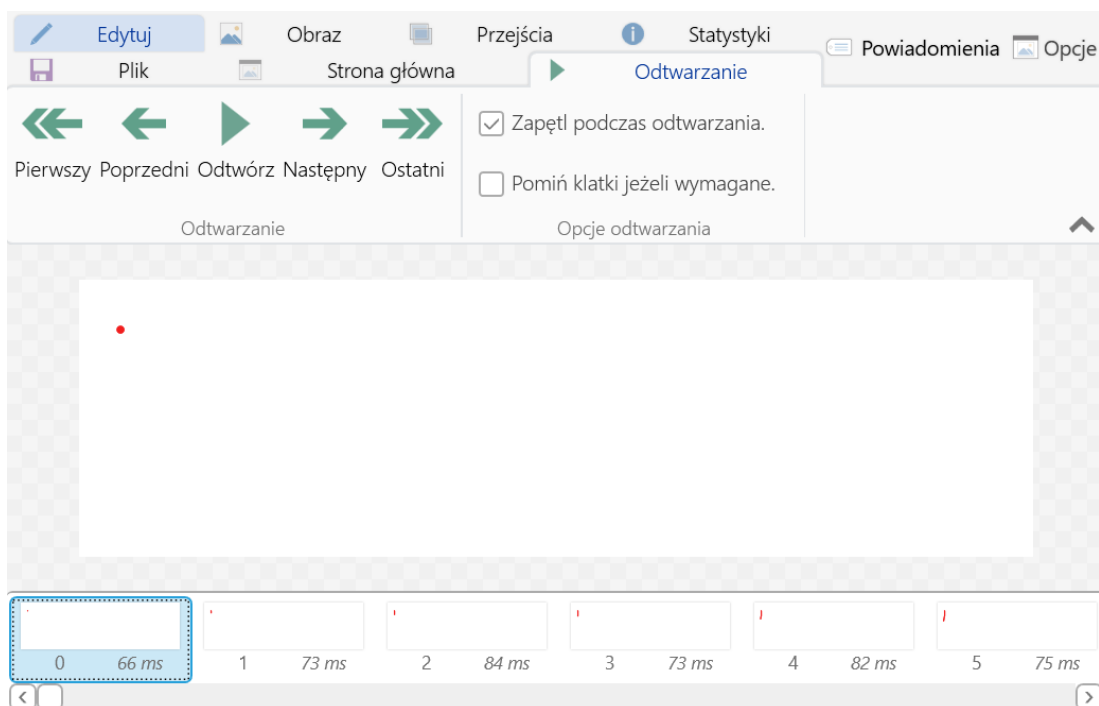
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Rysunek 4. Gotowy projekt z osią czasu animacji.

5. Odtwórzcie animację przyciskiem *Odtwarzanie* i sprawdźcie, czy napis pojawia się płynnie i jest dobrze widoczny. Jeśli coś nie wygląda dobrze, możecie poprawić rysunek lub usunąć wybrane klatki.

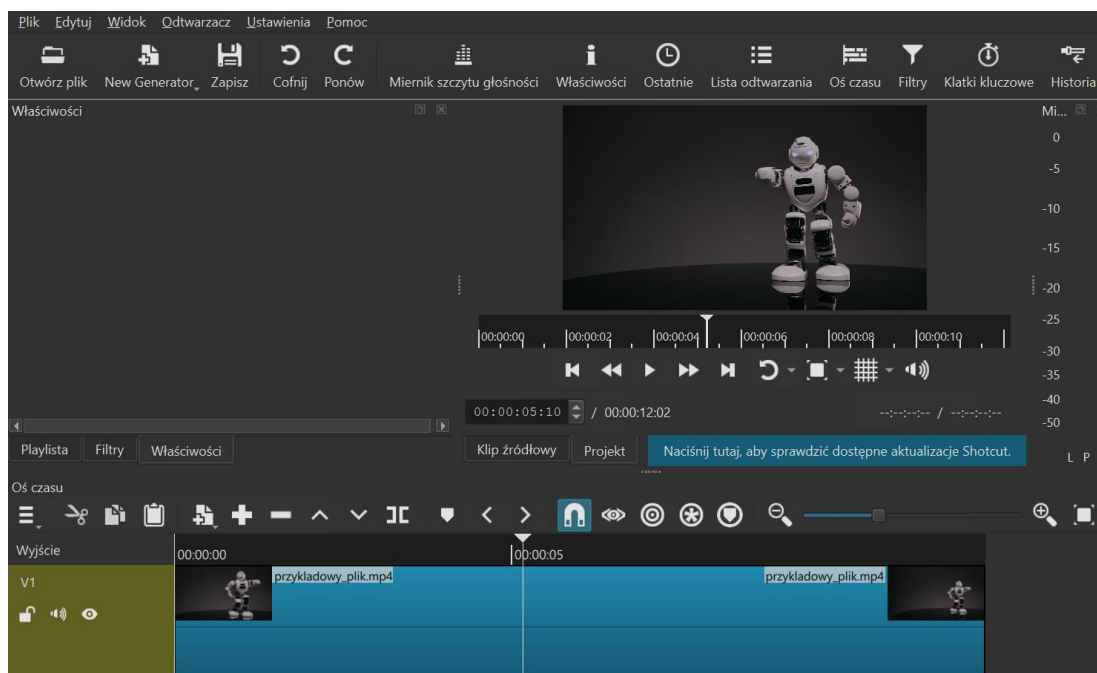
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Rysunek 5. Okno odtwarzania programu ScreenToGif.

6. Gdy animacja będzie gotowa, zapiszcie ją jako plik GIF. Wybierzcie z menu *Plik*, *Zapisz jako*, a następnie wybierzcie format GIF. Nadajcie nazwę, np. *tytul_robot.gif*, i zapiszcie plik w wybranym folderze.
7. Uruchomcie program Shotcut. Następnie wybierzcie z menu *Plik*, *Otwórz plik...* i wczytajcie klip wideo z robotem. Możecie użyć przykładowego pliku udostępnionego przez nauczyciela lub bazować na materiale utworzonym na wcześniejszych lekcjach. Przeciągnijcie klip na oś czasu, na ścieżkę V1.

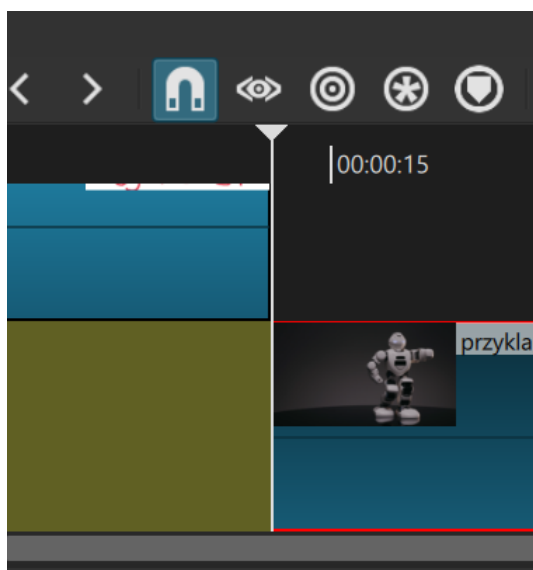
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Rysunek 6. Program Shotcut z przykładowym klipem.

8. Zanim dodacie animację, utwórzcie dodatkową ścieżkę. Kliknijcie prawym przyciskiem myszy w pustym miejscu osi czasu, wybierzcie *Akcje ścieżek*, a następnie *Dodaj ścieżkę wideo*. Na osi czasu pojawi się nowa ścieżka V2.
9. Ponownie wybierzcie *Plik, Otwórz plik...* i wczytajcie zapisany wcześniej plik GIF. Następnie przeciągnijcie go na oś czasu i umieśćcie na początku filmu jako animowaną stronę tytułową (na ścieżce V1 lub V2). Podczas dodawania pliku GIF może pojawić się okno *Konwertuj na łatwy w edycji*. Oznacza to, że program proponuje zmianę formatu pliku, aby lepiej działał na osi czasu. W tym oknie wystarczy kliknąć przycisk *OK*. Program automatycznie przekonwertuje plik i przygotuje go do pracy w projekcie.

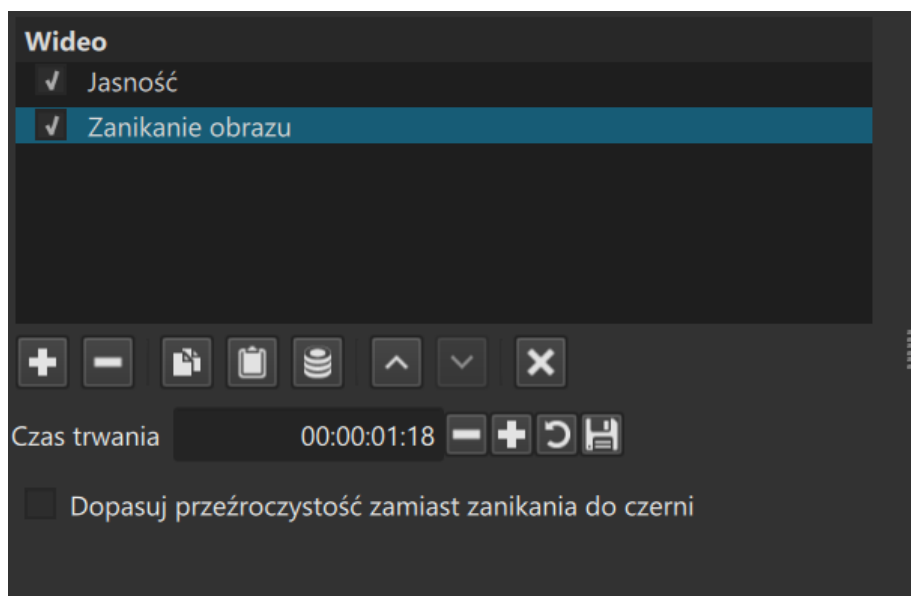
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Rysunek 9. Oś czasu z przesuniętymi klipami.

13. Możecie też dopasować długość GIF-a – skróćcie lub wydłużcie go, przeciągając jego krawędzie, aby przejście między animacją a filmem było płynne.
14. Odtwórzcie całość i sprawdźcie, czy kolejność jest poprawna: najpierw animowany tytuł, a następnie właściwy film.
15. Aby przejście między animowanym napisem a filmem wyglądało płynnie i profesjonalnie, zastosujecie efekt zanikania obrazu. Dzięki temu jeden element będzie stopniowo znikał, a drugi pojawiał się w jego miejscu, co daje naturalny i przyjemny dla oka efekt przejścia.
16. Zaznaczcie GIF na ścieżce V2. Przejdźcie do zakładki *Filtry* i kliknijcie *+*. W wyszukiwarce wpiszcie *zanikanie* lub *fade*. Wybierzcie filtr *Zanikanie obrazu*. Jeśli nie widzicie nazwy, wybierzcie filtr z ikoną przejścia (ciemniejący obraz).

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Rysunek 10. Dodawanie efektu zanikania obrazu.

17. Odtwórzcie całość i sprawdźcie efekt – animowany napis znika, a film płynnie się pojawia.
18. Na zakończenie zapiszcie projekt. Wybierzcie z menu *Plik, Zapisz jako*, nadajcie plikowi nazwę (np. *film_robot_przejście*) i zapiszcie go w wybranym folderze. Dzięki temu będziecie mogli wrócić do projektu na kolejnej lekcji.

Praktyczne ćwiczenie:

Otwórz w programie Shotcut nowy projekt lub skorzystaj z pliku przygotowanego na lekcji. Wczytaj klip wideo z robotem i przeciągnij go na oś czasu na ścieżkę V1.

Następnie wczytaj przygotowany wcześniej plik GIF (animowany napis) i przeciągnij go na ścieżkę V2. Możesz użyć GIF-a stworzonego na lekcji lub przygotować nowy w programie *ScreenToGif*, np. z napisem „Koniec filmu” lub „Dziękuję za uwagę”. Umieść go na końcu filmu jako animowane zakończenie.

Dopasuj długość GIF-a, przeciągając jego krawędzie, tak aby trwał kilka sekund. Następnie ustaw klip wideo tak, aby kończył się przed animacją.

Teraz wykonaj przejście między filmem a GIF-em. Dopasuj długość efektów tak, aby film stopniowo zniknął, a animacja pojawiała się płynnie.

Odtwórz cały film i sprawdź, czy zakończenie wygląda naturalnie. Jeśli efekt jest zbyt szybki lub zbyt wolny, zmień długość zanikania.

Na koniec zapisz projekt, wybierając *Plik, Zapisz jako*.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 6: PixelPerfect Studio

Temat lekcji nr 29 - Edycja dźwięku - korekcja, dodawanie ścieżek dźwiękowych i innych efektów.

Treści nauczania (wymagania szczegółowe - podstawa programowa) – I. 5); II. 3) a); II. 3) d);

Cel lekcji:

- Uczeń rozumie, na czym polega edycja dźwięku w projekcie filmowym oraz jakie ma znaczenie dla odbioru filmu.
- Uczeń wyjaśnia, do czego służą podstawowe efekty dźwiękowe (np. głośność, wyciszanie, normalizacja) i jak wpływają na jakość nagrania.
- Uczeń potrafi dodać ścieżkę dźwiękową w programie Shotcut, dopasować jej głośność oraz zastosować podstawowe efekty dźwiękowe.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- Uczeń przygotowuje i prezentuje rozwiązania problemów, posługując się podstawowymi aplikacjami (program do montażu wideo – Shotcut) na swoim komputerze.
- Uczeń gromadzi, porządkuje i selekcionuje materiały cyfrowe (klipy wideo oraz pliki dźwiękowe), potrzebne do realizacji projektu filmowego, oraz zapisuje efekty swojej pracy w postaci projektu.
- Uczeń wzbogaca materiał wideo o elementy dźwiękowe (muzykę, efekty), wykorzystując podstawowe narzędzia edycji oraz dostosowuje ich ustawienia (głośność, długość, efekty), dbając o czytelność i estetykę filmu.

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści programowe realizowane będą z wykorzystaniem uniwersalnych narzędzi dostępnych w systemach operacyjnych, takich jak eksplorator plików oraz menedżery

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

okien, co umożliwi uczniom sprawne zarządzanie materiałami (klipami wideo oraz plikami dźwiękowymi) wykorzystywanymi w projekcie.

- W miarę możliwości technicznych uczniowie będą korzystać z zapisu danych w środowiskach wirtualnych (chmurze) w celu gromadzenia, przechowywania oraz udostępniania projektów filmowych.
- Realizacja treści programowych odbywać się będzie w programie Shotcut, który posłuży jako podstawowe narzędzie do edycji dźwięku, dodawania ścieżek audio, stosowania efektów dźwiękowych oraz organizowania materiału na osi czasu.

Uczeń:

- Wyjaśnia, na czym polega edycja dźwięku oraz jego wykorzystanie w projekcie filmowym.
- Wskazuje, jakie elementy dźwiękowe (np. muzyka, efekty, głośność) wpływają na odbiór filmu i jego atrakcyjność.
- Przedstawia znaczenie dopasowania dźwięku do obrazu oraz stosowania efektów dźwiękowych w budowaniu spójnej formy filmowej.
- Potrafi dodać ścieżkę dźwiękową w programie Shotcut oraz dopasować jej długość i głośność.
- Opisuje rolę efektów dźwiękowych (np. wyciszanie, normalizacja) jako sposobu na poprawę jakości i płynności dźwięku w filmie..

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna,
- praca na komputerze,
- pogadanka.

Środki dydaktyczne

- zestaw PC z dostępem do Internetu oraz zainstalowanym programem Shotcut,
- rzutnik multimedialny,
- e-podręcznik.

Formy pracy

- Praca indywidualna.

Przebieg lekcji:

Faza I. Wprowadzenie

Podczas lekcji uczniowie zostaną zapoznani z zasadami edycji dźwięku w programie Shotcut. Omówione zostanie znaczenie dodawania ścieżek dźwiękowych (muzyki, efektów), dopasowania ich głośności oraz stosowania podstawowych efektów dźwiękowych. Nauczyciel

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

przypomni również podstawowe elementy programu Shotcut: okno podglądu, listę odtwarzania i oś czasu, ze szczególnym uwzględnieniem ścieżek audio (A1, A2).

1. Nauczyciel rozpoczyna lekcję, wyjaśniając, że dźwięk jest bardzo ważnym elementem filmu – odpowiednio dobrana muzyka lub efekty mogą poprawić jego odbiór i nadać mu odpowiedni nastrój. Podkreśla, że ważne jest nie tylko dodanie dźwięku, ale także jego właściwe dopasowanie.
2. Nauczyciel pyta uczniów, czy zwracają uwagę na dźwięk podczas oglądania filmów (np. muzykę w tle, efekty dźwiękowe) oraz zadaje pytanie, czy zbyt głośny lub źle dopasowany dźwięk może utrudniać odbiór filmu.

Faza II. Część zasadnicza - realizacja

1. Nauczyciel wprowadza uczniów w temat edycji dźwięku, wyjaśniając, że oprócz obrazu bardzo ważnym elementem filmu jest dźwięk (muzyka, efekty). Podkreśla, że odpowiednio dobrany i dopasowany dźwięk może poprawić odbiór filmu i nadać mu odpowiedni nastrój.
2. Dyskusja kierowana: Nauczyciel zadaje pytania, gdzie można wykorzystać dźwięk w filmie (np. muzyka w tle, efekty dźwiękowe, nagrany głos) oraz czy dźwięk zawsze poprawia odbiór filmu. Uczniowie podają przykłady i zastanawiają się, kiedy dźwięk może przeszkadzać (np. gdy jest za głośny).
3. Nauczyciel omawia podstawy pracy z dźwiękiem w programie Shotcut, pokazując, jak dodać ścieżkę audio, dopasować jej długość oraz zmienić głośność. Wyjaśnia również, do czego służą podstawowe filtry dźwiękowe (np. głośność, wyciszanie, normalizacja).
4. Pokaz nauczyciela: Nauczyciel uruchamia program Shotcut i prezentuje, w jaki sposób dodać plik dźwiękowy do projektu oraz umieścić go na ścieżce A1. Następnie pokazuje, jak zmienić głośność dźwięku oraz dodać efekty, takie jak wyciszanie na początku i końcu nagrania. Demonstruje także zastosowanie filtra *Normalizuj* i pokazuje efekt w oknie podglądu.
5. Część praktyczna: Uczniowie uruchamiają program Shotcut i dodają ścieżkę dźwiękową do swojego projektu. Dopasowują jej długość, ustawiają głośność oraz dodają efekty (np. wyciszanie, normalizacja). Następnie odtwarzają film i sprawdzają, czy dźwięk pasuje do obrazu oraz czy brzmi naturalnie.

Faza III. Podsumowanie

Uczniowie prezentują efekty swojej pracy, omawiając sposób dodania i edycji dźwięku w swoim projekcie. Wyjaśniają, jak dopasowali głośność, długość ścieżki oraz jakie efekty zastosowali (np. wyciszanie, normalizacja, redukcja szumu). Zwracają uwagę, czy dźwięk pasuje do obrazu i czy brzmi naturalnie. Nauczyciel podsumowuje lekcję, podkreślając znaczenie odpowiedniego dopasowania dźwięku oraz wykorzystania efektów audio w tworzeniu spójnego i atrakcyjnego filmu. Następnie nauczyciel dokonuje oceny pracy uczniów zgodnie z poniższymi kryteriami:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Ocena dopuszczająca:

– uczeń potrafi wskazać podstawowe elementy interfejsu programu Shotcut oraz rozpoznaje ścieżki audio.

Ocena dostateczna:

– uczeń rozpoznaje podstawowe elementy interfejsu i potrafi samodzielnie wczytać plik GIF do projektu oraz odtworzyć film.

Ocena dobra:

– uczeń poprawnie opisuje sposób dodania i edycji dźwięku oraz potrafi dopasować jego długość i głośność.

Ocena bardzo dobra:

– uczeń szczegółowo omawia zastosowane efekty dźwiękowe (np. wyciszanie, normalizacja) i potrafi je poprawnie wykorzystać w projekcie.

Ocena celująca:

– uczeń samodzielnie i świadomie wykorzystuje różne efekty dźwiękowe, potrafi uzasadnić swoje decyzje oraz proponuje własne rozwiązania poprawiające jakość i odbiór filmu.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, np.:

1. Jak można zmienić głośność dźwięku w projekcie:
 - a. przeciągając film na osi czasu
 - b. zapisując projekt
 - c. **dodając filtr Wzmocnienie/Głośność**
 - d. zmieniając nazwę pliku
2. Co daje zastosowanie efektu wyciszania dźwięku:
 - a. usuwa dźwięk całkowicie
 - b. przyspiesza film
 - c. **sprawia, że dźwięk stopniowo się pojawia lub zanika**
 - d. zmienia kolor obrazu
3. Do czego służy filtr Normalizuj:
 - a. przycina film
 - b. dodaje napisy
 - c. **wyrównuje poziom głośności dźwięku**
 - d. zmienia kolejność klipów

Faza V. Dodatkowe, praca domowa

Zadanie domowe (dla chętnych).

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Otwórz w programie Shotcut projekt wykonany na lekcji lub inny krótki klip wideo. Twoim zadaniem jest poprawić i urozmaicić film, wykorzystując dźwięk.

Dodaj do projektu ścieżkę dźwiękową (np. muzykę lub efekt dźwiękowy) i umieść ją na ścieżce A1. Dopasuj jej długość tak, aby kończyła się razem z filmem.

Następnie ustaw odpowiednią głośność, korzystając z filtra *Głośność/Wzmocnienie*. Spróbuj dobrać taki poziom, aby dźwięk był dobrze słyszalny, ale nie zagłuszał filmu.

Dodaj efekt płynnego pojawiania się dźwięku na początku oraz jego wyciszenia na końcu.

Jeśli chcesz, dodaj drugą ścieżkę dźwiękową (np. krótki efekt) i dopasuj ją do wybranego fragmentu filmu. Spróbuj wyrównać głośność obu ścieżek, korzystając z filtra *Normalizuj*.

Odtwórz cały film i sprawdź, czy dźwięk pasuje do obrazu oraz czy jest płynny i czytelny. Jeśli coś nie wygląda dobrze, popraw ustawienia lub długość ścieżek. Na koniec zapisz projekt i przygotuj się do pokazania efektu na kolejnej lekcji.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1 – Edycja dźwięku - korekcja, dodawanie ścieżek dźwiękowych i innych efektów.

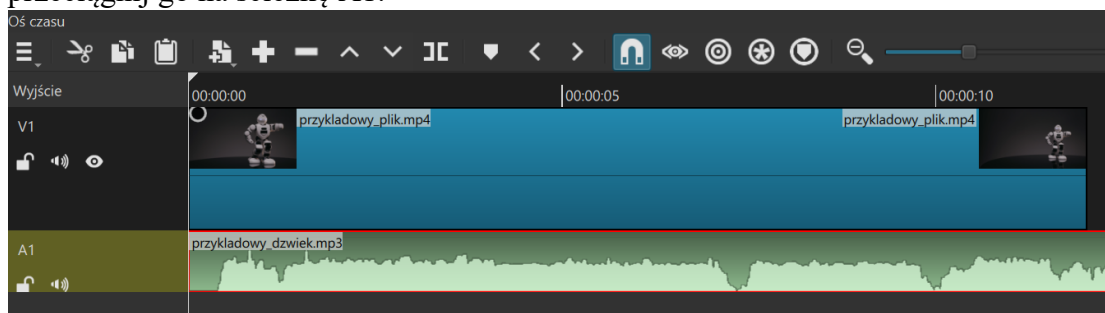
Wprowadzenie!

Dziś zajmiemy się tym, w jaki sposób poprawić i uatrakcyjnić film, wykorzystując dźwięk. Sam obraz często nie wystarcza – odpowiednio dobrana muzyka lub efekty dźwiękowe mogą całkowicie zmienić odbiór filmu i sprawić, że stanie się on bardziej interesujący.

Podczas lekcji zobaczymy, jak dodawać ścieżki dźwiękowe w programie Shotcut oraz jak dopasować ich głośność i długość do filmu. Nauczymy się również, jak stosować podstawowe efekty dźwiękowe, takie jak wyciszanie, normalizacja czy redukcja szumu. Dzięki temu przekonacie się, że dobrze dobrany i poprawnie ustawiony dźwięk sprawia, że film wygląda bardziej profesjonalnie i przyjemniej się go ogląda.

Scenariusz

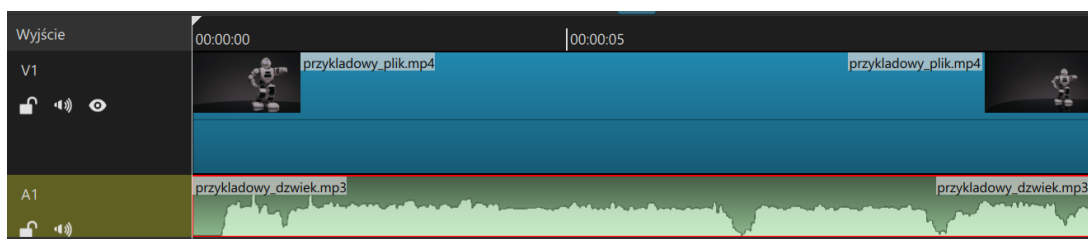
1. Uruchomcie program Shotcut. Następnie wybierzcie z menu *Plik, Otwórz plik...* i wczytajcie przykładowy klip wideo udostępniony przez nauczyciela. Po załadowaniu przeciągnijcie go z okna podglądu na oś czasu, na ścieżkę V1.
2. Sprawdź, czy na osi czasu masz już dodany klip wideo (V1). Następnie dodaj ścieżkę dźwiękową – kliknij prawym przyciskiem myszy w pustym miejscu osi czasu i wybierz *Dodaj ścieżkę audio*. Na osi czasu pojawi się ścieżka A1. Wybierz z menu *Plik, Otwórz plik...* i wczytaj plik dźwiękowy udostępniony przez nauczyciela. Po załadowaniu przeciągnij go na ścieżkę A1.



Rysunek 1. Oś czasu z dodanym klipem i dźwiękiem.

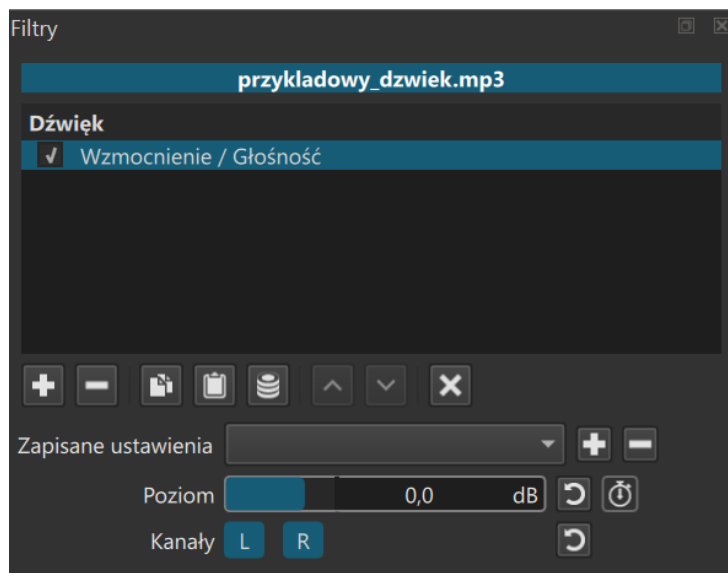
3. Odtwórz film przyciskiem *Play* i sprawdź, jak dźwięk pasuje do obrazu. Zwróć uwagę, czy jest dobrze słyszalny i czy nie jest zbyt głośny lub zbyt cichy.
4. Sprawdź, czy ścieżka dźwiękowa nie jest dłuższa niż film. Jeśli dźwięk wychodzi poza klip wideo (tak jak na obrazku), najedź myszką na prawą krawędź pliku audio na ścieżce A1, aż pojawi się strzałka. Kliknij i przeciągnij krawędź w lewo, aby skrócić dźwięk i dopasować go do długości filmu. Najlepiej ustaw go tak, aby kończył się dokładnie w tym samym momencie co klip wideo.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Rysunek 2. Oś czasu z synchronizowanymi ścieżkami.

5. Odtwórz film i sprawdź, czy obraz i dźwięk kończą się razem – dzięki temu projekt będzie wyglądał bardziej profesjonalnie.
6. Zaznaczcie ścieżkę audio i przejdźcie do zakładki *Filtry*. Kliknijcie +, wybierzcie kategorię *Dźwięk* i dodajcie filtr *Wzmocnienie/Głośność*. Zapoznajcie się z filtrem *Głośność/Wzmocnienie*. W jego ustawieniach znajdziecie:
 - *Poziom (dB)* – suwak do zmiany głośności (wartości ujemne ściszą, dodatnie pogłaśniają, 0,0 dB to brak zmian),
 - *Kanały (L / R)* – określają, czy zmiana dotyczy lewego i prawego kanału (najczęściej zaznaczone oba),
 - *Zapisane ustawienia* – możliwość zapisania lub wczytania ustawień,
 - przyciski resetu – pozwalają przywrócić domyślne wartości. Spróbujcie zmieniać poziom głośności i obserwujcie, jak wpływa to na dźwięk w filmie.

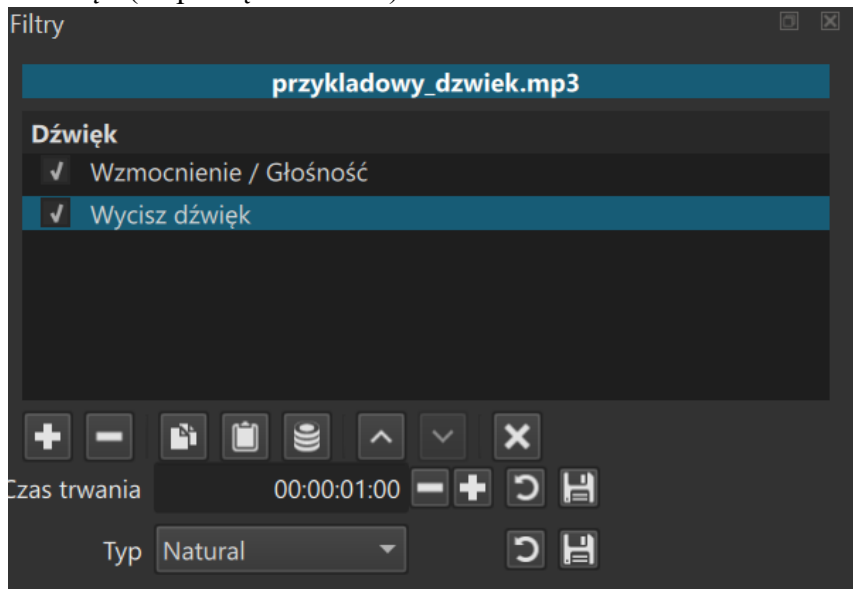


Rysunek 3. Opcje filtru *Wzmocnienie/Głośność*.

7. Za pomocą suwaka dopasujcie głośność dźwięku. Spróbujcie ustawić ją tak, aby była wyraźna, ale nie za głośna (np. około -10 dB do -20 dB).
8. Aby dźwięk nie zaczynał się nagle, tylko pojawiał się stopniowo, zastosujcie efekt płynnego wejścia. Dzięki temu początek filmu będzie brzmiał bardziej naturalnie i

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

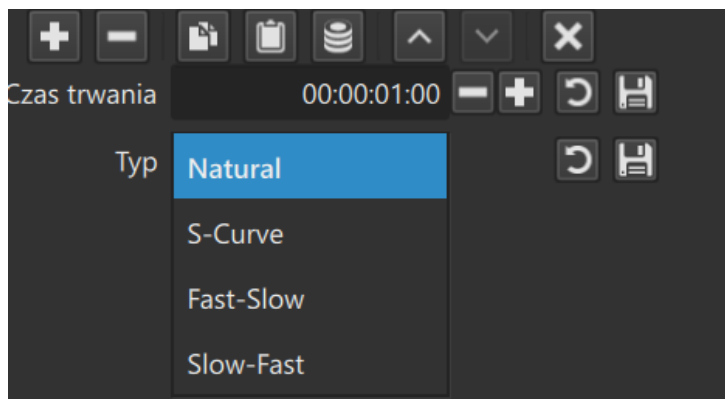
profesjonalnie. W zakładce *Filtry* kliknijcie +, wybierzcie kategorię *Dźwięk* i dodajcie filtr *Wycisz dźwięk* (na początku ścieżki).



Rysunek 4. Aktywowanie filtru wycisz dźwięk.

9. Po dodaniu filtra możecie dopasować jego ustawienia. Zwróćcie uwagę na:
- *Czas trwania* – określa, jak długo dźwięk będzie się „rozkręcał” od cisy do pełnej głośności (np. 1 sekunda),
 - *Typ* – sposób, w jaki dźwięk się pojawia:
 - i. *Natural* – równomierne, naturalne zwiększanie głośności (najczęściej używane),
 - ii. *S-Curve* – łagodny początek i koniec, bardziej płynne przejście,
 - iii. *Fast-Slow* – szybki początek, potem wolniejsze zwiększanie,
 - iv. *Slow-Fast* – wolny początek, potem szybkie zwiększenie głośności.

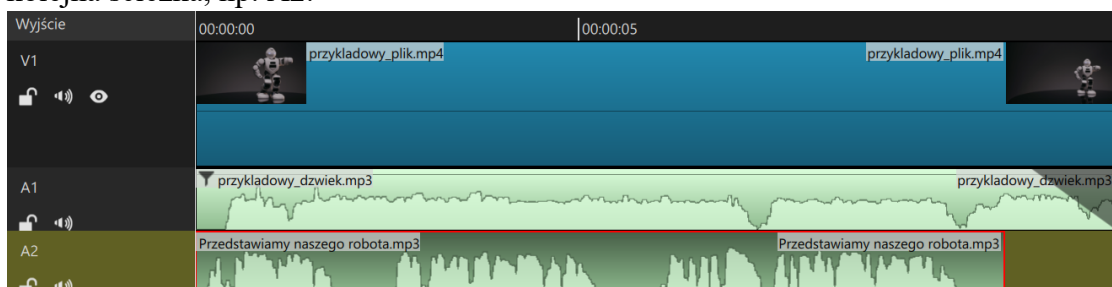
Spróbujcie zmienić typ oraz czas trwania i sprawdźcie, który efekt najlepiej pasuje do Waszego filmu.



Rysunek 5. Opcje filtru Wycisz dźwięk.

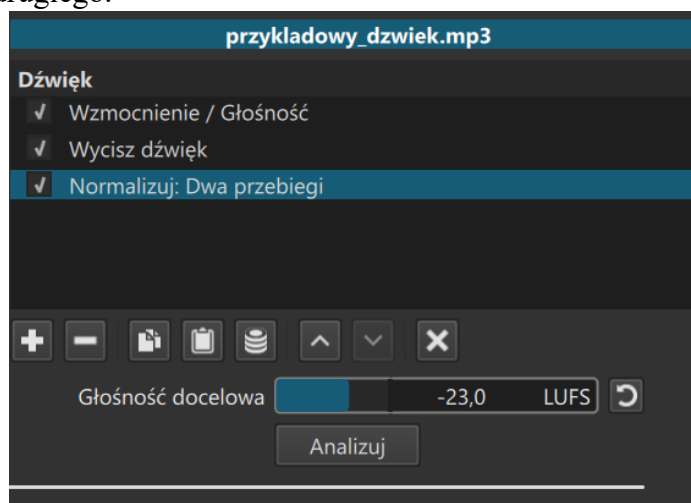
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

10. Zwróćcie uwagę, że różne ustawienia mogą dawać zupełnie inny efekt. Spróbujcie porównać kilka opcji – odtwórzcie fragment filmu dla każdego typu (*Natural*, *S-Curve*, *Fast-Slow*, *Slow-Fast*) i zdecydujcie, który brzmi najlepiej.
11. Pamiętajcie, że efekt nie powinien być zbyt długi – jeśli dźwięk pojawia się zbyt wolno, widz może odnieść wrażenie, że coś jest nie tak z nagraniem. Najczęściej wystarczy 1–2 sekundy.
12. Pamiętajcie, że efekt nie powinien być zbyt długi – jeśli dźwięk pojawia się zbyt wolno, widz może odnieść wrażenie, że coś jest nie tak z nagraniem. Najczęściej wystarczy 1–2 sekundy.
13. Spróbujcie dodać drugą ścieżkę dźwiękową. Kliknijcie prawym przyciskiem myszy w pustym miejscu osi czasu, wybierzcie *Dodaj ścieżkę audio*. Na osi czasu pojawi się kolejna ścieżka, np. A2.



Rysunek 6. Oś czasu z dodaną drugą ścieżką dźwiękową.

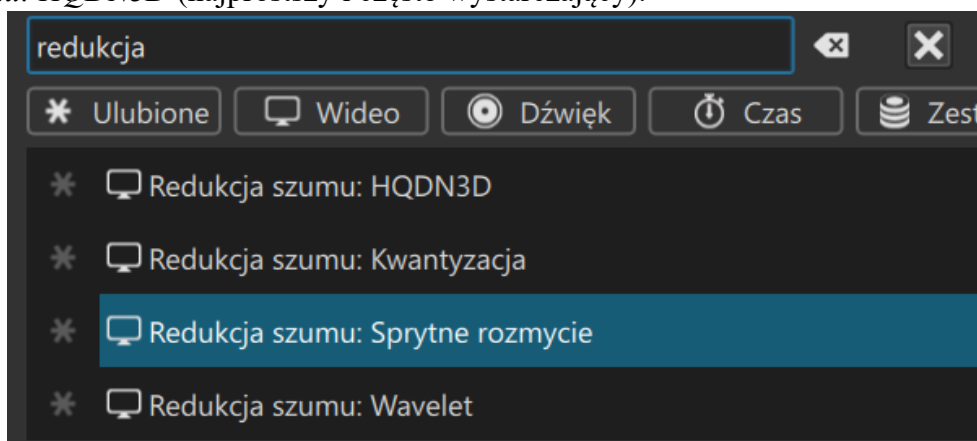
14. Odtwórzcie całość i sprawdźcie efekt – czy dźwięki są dobrze dopasowane i nie zagłuszają się nawzajem.
15. Jeśli używacie dwóch ścieżek dźwiękowych (np. muzyki i efektu), możecie zastosować filtr *Normalizuj*: *Dwa przebiegi*, aby uporządkować ich głośność. Zaznaczcie każdą ścieżkę osobno i dodajcie filtr *Normalizuj*. Filtr ten sprawi, że oba dźwięki będą miały bardziej wyrównany poziom głośności, dzięki czemu jeden nie będzie nagle dużo głośniejszy od drugiego.



Rysunek 7. Filtr Normalizuj: dwa przebiegi.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

16. Zapoznajcie się z filtrem *Normalizuj: Dwa przebiegi*. Jest to bardziej zaawansowany filtr, który wyrównuje głośność dźwięku w całym nagraniu. W ustawieniach znajdziecie *Głośność docelową (LUFS)* – określa ona, jak głośny ma być dźwięk (np. –23 LUFS to standardowy poziom, a –16 LUFS będzie głośniejszy). Aby filtr działał poprawnie, kliknijcie przycisk *Analizuj* – program sprawdzi cały dźwięk i dopasuje jego poziom. Efekt działania polega na tym, że ciche fragmenty zostają podgłośnione, a głośne wyrównane, dzięki czemu całość brzmi bardziej spójnie i profesjonalnie.
17. Po zastosowaniu filtra odtwórzcie film i sprawdźcie, czy dźwięk jest bardziej wyrównany. Jeśli nadal jest zbyt głośny lub cichy, zmieńcie wartość *LUFS* i ponownie kliknijcie *Analizuj*.
18. Teraz zastosujcie filtr *Normalizuj: Dwa przebiegi* do drugiej ścieżki dźwiękowej.
19. Na koniec spróbujcie dodać jeszcze jeden efekt poprawiający jakość dźwięku. Zaznaczcie ścieżkę audio i przejdźcie do zakładki *Filtry*, a następnie kliknijcie + i wybierzcie kategorię *Dźwięk*. Dodajcie filtr *Redukcja szumów*. W programie dostępnych jest kilka filtrów redukcji szumu. Wybierzcie jeden z nich, np. *Redukcja szumu: HQDN3D* (najprostszy i często wystarczający).



Rysunek 8. Dostępne filtry dla redukcji szumu.

Filtr ten zmniejsza niepożądane dźwięki w tle, takie jak szum czy lekkie zakłócenia, dzięki czemu nagranie brzmi czyściej.

20. Jeszcze, możecie przetestować inne opcje, np.:
 - *Kwantyzacja* – mocniej upraszcza dźwięk, może dawać mniej naturalny efekt,
 - *Sprytne rozmycie* – delikatnie wygładza dźwięk,
 - *Wavelet* – bardziej zaawansowane oczyszczanie dźwięku.
21. Odtwórzcie fragment filmu i porównajcie efekty. Zastanówcie się, który filtr daje najlepszy rezultat i nie pogarsza jakości nagrania.
22. Jeśli wszystko jest poprawne, zapiszcie projekt. Wybierzcie z menu *Plik, Zapisz jako*, nadajcie nazwę (np. *film_robot_dzwiek*) i zapiszcie plik w wybranym folderze.

Praktyczne ćwiczenie:

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Otwórz w programie Shotcut nowy projekt lub skorzystaj z pliku przygotowanego na lekcji. Wczytaj klip wideo z robotem i przeciągnij go na oś czasu na ścieżkę V1.

Następnie dodaj ścieżkę dźwiękową – kliknij prawym przyciskiem myszy w pustym miejscu osi czasu i wybierz *Dodaj ścieżkę audio*. Wczytaj plik dźwiękowy (np. muzykę) i przeciągnij go na ścieżkę A1.

Dopasuj długość dźwięku, przeciągając jego krawędzie tak, aby kończył się razem z filmem. Następnie ustaw jego głośność, dodając filtr *Głośność/Wzmocnienie*. Spróbuj dobrać taki poziom, aby dźwięk był wyraźny, ale nie za głośny.

Dodaj efekt płynnego pojawiania się dźwięku na początku oraz jego wyciszenia na końcu, korzystając z filtrów dźwięku. Sprawdź, czy dźwięk nie zaczyna się i nie kończy nagle.

Następnie dodaj drugą ścieżkę dźwiękową (A2), np. krótki efekt lub inną muzykę. Dopasuj jej długość i głośność tak, aby oba dźwięki dobrze do siebie pasowały.

Zastosuj filtr *Normalizuj: Dwa przebiegi* do obu ścieżek, aby wyrównać ich poziom głośności. Jeśli trzeba, użyj dodatkowo filtra *Głośność/Wzmocnienie*.

Na koniec spróbuj dodać filtr redukcji szumu do jednej ze ścieżek i sprawdź, czy poprawia on jakość dźwięku.

Odtwórz cały film i sprawdź, czy dźwięk pasuje do obrazu oraz czy jest płynny i dobrze słyszalny. Jeśli coś nie wygląda dobrze, popraw ustawienia filtrów lub długość ścieżek.

Na koniec zapisz projekt, wybierając *Plik, Zapisz jako*.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Scenariusz lekcji z informatyki w oparciu o robotykę edukacyjną

Ilość godzin lekcyjnych – 1h

Klasa VII

Moduł 6: PixelPerfect Studio

Temat lekcji nr 30 - Wybieranie ustawień eksportu i renderowania filmu dla różnych platform.

Treści nauczania (wymagania szczegółowe - podstawa programowa) – I. 5); II. 3) a); II. 3) d);

Cel lekcji:

- Uczeń rozumie, na czym polega eksport i renderowanie filmu oraz dlaczego ustawienia eksportu są ważne dla jakości gotowego materiału.
- Uczeń wyjaśnia, do czego służą ustawienia takie jak rozdzielczość, liczba klatek na sekundę (fps), proporcje obrazu oraz jakość dźwięku.
- Uczeń potrafi dobrać odpowiednie ustawienia eksportu w programie Shotcut oraz przygotować film do różnych platform, np. YouTube, TikTok lub internetu.

Wymagania szczegółowe określone podstawą programową z informatyki dla II etapu kształcenia:

- Uczeń przygotowuje i prezentuje rozwiązania problemów, posługując się podstawowymi aplikacjami (program do montażu wideo – Shotcut) na swoim komputerze.
- Uczeń gromadzi, porządkuje i selekcjonuje materiały cyfrowe (klipy wideo, ścieżki dźwiękowe, napisy), potrzebne do realizacji projektu filmowego, oraz zapisuje efekty swojej pracy w postaci gotowego pliku wideo.
- Uczeń dobiera i modyfikuje ustawienia eksportu filmu (np. rozdzielczość, proporcje obrazu, liczba klatek, jakość dźwięku), dostosowując materiał do różnych platform i sposobów publikacji.

Sposób realizacji wymagań w innowacyjnym programie nauczania:

- Treści programowe realizowane będą z wykorzystaniem uniwersalnych narzędzi dostępnych w systemach operacyjnych, takich jak eksplorator plików oraz menedżery

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

okien, co umożliwi uczniom sprawne zarządzanie materiałami wykorzystywanymi w projekcie filmowym oraz gotowymi plikami eksportu.

- W miarę możliwości technicznych uczniowie będą korzystać z zapisu danych w środowiskach wirtualnych (chmurze) w celu gromadzenia, przechowywania oraz udostępniania projektów i gotowych filmów.
- Realizacja treści programowych odbywać się będzie w programie Shotcut, który posłuży jako podstawowe narzędzie do eksportu i renderowania filmu, dostosowywania ustawień jakości obrazu i dźwięku oraz przygotowywania materiałów do różnych platform, takich jak YouTube, TikTok czy internetowe komunikatory.

Uczeń:

- Wyjaśnia, na czym polega eksport i renderowanie filmu oraz dlaczego ustawienia eksportu są ważne dla jakości gotowego materiału.
- Wskazuje, jakie elementy ustawień eksportu (np. rozdzielczość, fps, proporcje obrazu, jakość dźwięku) wpływają na wygląd i wielkość filmu.
- Przedstawia znaczenie dostosowania ustawień eksportu do różnych platform, takich jak YouTube, TikTok czy internetowe komunikatory.
- Potrafi przygotować film do eksportu w programie Shotcut oraz dobrać odpowiednie ustawienia jakości obrazu i dźwięku.
- Opisuje rolę renderowania jako procesu tworzenia gotowego pliku wideo przeznaczonego do odtwarzania lub publikacji.

Metody pracy:

- pokaz multimedialny – e-platforma edukacyjna,
- praca na komputerze,
- pogadanka.

Środki dydaktyczne

- zestaw PC z dostępem do Internetu oraz zainstalowanym programem Shotcut,
- rzutnik multimedialny,
- e-podręcznik.

Formy pracy

- Praca indywidualna.

Przebieg lekcji:

Faza I. Wprowadzenie

Podczas lekcji uczniowie zostaną zapoznani z zasadami eksportu i renderowania filmu w programie Shotcut. Omówione zostanie znaczenie odpowiedniego doboru ustawień eksportu, takich jak rozdzielczość, proporcje obrazu, liczba klatek na sekundę oraz jakość dźwięku.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Nauczyciel przypomni również podstawowe elementy programu Shotcut związane z eksportem filmu, takie jak zakładka Eksportuj, ustawienia Wideo, Kodek oraz Dźwięk.

1. Nauczyciel rozpoczyna lekcję, wyjaśniając, że gotowy film powinien być odpowiednio przygotowany do miejsca, w którym będzie oglądany lub publikowany. Podkreśla, że inne ustawienia będą odpowiednie dla YouTube, a inne dla TikToka lub przesyłania filmu przez internet.
2. Nauczyciel pyta uczniów, czy zauważyli, że filmy publikowane w różnych serwisach mają inne proporcje obrazu lub jakość. Następnie zadaje pytanie, dlaczego nie każdy film powinien być eksportowany w maksymalnej jakości i jak może to wpływać na wielkość pliku oraz szybkość przesyłania filmu.

Faza II. Część zasadnicza - realizacja

1. Nauczyciel wprowadza uczniów w temat eksportu i renderowania filmu, wyjaśniając, że po zakończeniu montażu konieczne jest zapisanie projektu jako gotowego pliku wideo. Podkreśla, że odpowiednie ustawienia eksportu wpływają na jakość obrazu, dźwięku oraz wielkość pliku.
2. Dyskusja kierowana: Nauczyciel zadaje pytania, gdzie uczniowie najczęściej oglądają filmy (np. YouTube, TikTok, telewizor, telefon) oraz czy wszystkie filmy mają takie same proporcje i jakość obrazu. Uczniowie podają przykłady i zastanawiają się, dlaczego różne platformy wymagają innych ustawień eksportu.
3. Nauczyciel omawia podstawowe ustawienia eksportu w programie Shotcut, takie jak rozdzielczość, liczba klatek na sekundę (fps), proporcje obrazu, jakość filmu oraz ustawienia dźwięku. Wyjaśnia, jak zmiana tych parametrów wpływa na wygląd filmu oraz wielkość gotowego pliku.
4. Pokaz nauczyciela: Nauczyciel uruchamia program Shotcut i prezentuje zakładkę Eksportuj. Pokazuje gotowe profile eksportu oraz omawia ustawienia w sekcjach Wideo, Kodek i Dźwięk. Demonstruje zmianę rozdzielczości, proporcji obrazu oraz jakości eksportu, a następnie pokazuje, jak rozpocząć renderowanie filmu i obserwować postęp eksportu.
5. Część praktyczna: Uczniowie uruchamiają program Shotcut i otwierają przygotowany wcześniej projekt filmu. Następnie przygotowują różne ustawienia eksportu – np. wersję Full HD do YouTube oraz pionową wersję do TikToka lub Shorts. Eksportują filmy, porównują ich jakość, proporcje obrazu, wielkość plików oraz czas renderowania.

Faza III. Podsumowanie

Uczniowie prezentują efekty swojej pracy, omawiając przygotowane wersje filmu oraz wyjaśniając, jakie ustawienia eksportu zastosowali dla różnych platform (np. YouTube, TikTok, internet). Opisują również, jak zmiana rozdzielczości, liczby klatek na sekundę czy jakości eksportu wpłynęła na wygląd filmu, płynność obrazu oraz wielkość pliku. Nauczyciel podsumowuje lekcję, zwracając uwagę na znaczenie odpowiedniego doboru ustawień eksportu

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

i renderowania w tworzeniu filmów przeznaczonych do różnych zastosowań. Następnie nauczyciel dokonuje oceny pracy uczniów zgodnie z poniższymi kryteriami:

Ocena dopuszczająca:

- uczeń potrafi wskazać podstawowe elementy zakładki Eksportuj w programie Shotcut.

Ocena dostateczna:

- uczeń rozpoznaje podstawowe ustawienia eksportu i potrafi samodzielnie wyeksportować film do pliku MP4.

Ocena dobra:

- uczeń poprawnie opisuje ustawienia eksportu oraz potrafi dobrać odpowiednią rozdzielczość i proporcje obrazu do wybranej platformy.

Ocena bardzo dobra:

- uczeń szczegółowo omawia wpływ ustawień eksportu (np. fps, jakość, bitrate) na wygląd i wielkość filmu oraz potrafi świadomie je dopasować.

Ocena celująca:

- uczeń samodzielnie i świadomie przygotowuje różne wersje eksportu filmu, potrafi uzasadnić wybór ustawień oraz proponuje własne rozwiązania poprawiające jakość i wygodę publikacji materiału wideo.

Faza IV. Quiz/Test sprawdzający

Ocena nabytej wiedzy uczniów na podstawie pytań sprawdzających, np.:

1. Do czego służy proces renderowania filmu:
 - a. do rysowania grafiki
 - b. do usuwania filmu
 - c. **do tworzenia gotowego pliku wideo**
 - d. do nagrywania dźwięku
2. Które proporcje obrazu najczęściej wykorzystuje się w filmach na YouTube:
 - a. 1:1
 - b. 9:16
 - c. **16:9**
 - d. 4:3
3. Dlaczego filmy eksportowane w wyższej jakości zajmują więcej miejsca na dysku:
 - a. ponieważ mają mniej klatek
 - b. ponieważ są krótsze
 - c. **ponieważ zapisują więcej szczegółów obrazu i dźwięku**
 - d. ponieważ nie mają dźwięku

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Faza V. Dodatkowe, praca domowa

Zadanie domowe (dla chętnych).

Otwórz w programie Shotcut projekt wykonany na lekcji lub inny krótki klip wideo. Twoim zadaniem jest przygotować różne wersje eksportu filmu, dopasowane do różnych platform.

Najpierw przygotuj wersję filmu przeznaczoną do publikacji na YouTube. Ustaw rozdzielczość 1920×1080 (Full HD), proporcje 16:9 oraz wysoką jakość eksportu. Następnie wyeksportuj film i zapisz go jako Film_YouTube.mp4.

Kolejnym krokiem będzie przygotowanie wersji pionowej do TikToka lub Shorts. Zmień proporcje obrazu na 9:16 oraz rozdzielczość na 1080×1920 . Wyeksportuj plik pod nazwą Film_TikTok.mp4.

Na końcu przygotuj trzecią wersję filmu o mniejszym rozmiarze pliku, przeznaczoną do przesłania przez internet lub komunikator. Zmniejsz jakość eksportu oraz rozdzielczość, a następnie zapisz plik jako Film_Internet.mp4.

Otwórz wszystkie trzy filmy i porównaj ich wygląd, jakość obrazu, płynność ruchu oraz wielkość plików. Zastanów się, które ustawienia najlepiej pasują do różnych zastosowań. Na koniec zapisz projekt i przygotuj się do pokazania efektów na kolejnej lekcji.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

Załącznik nr 1 – Wybieranie ustawień eksportu i renderowania filmu dla różnych platform.

Wprowadzenie!

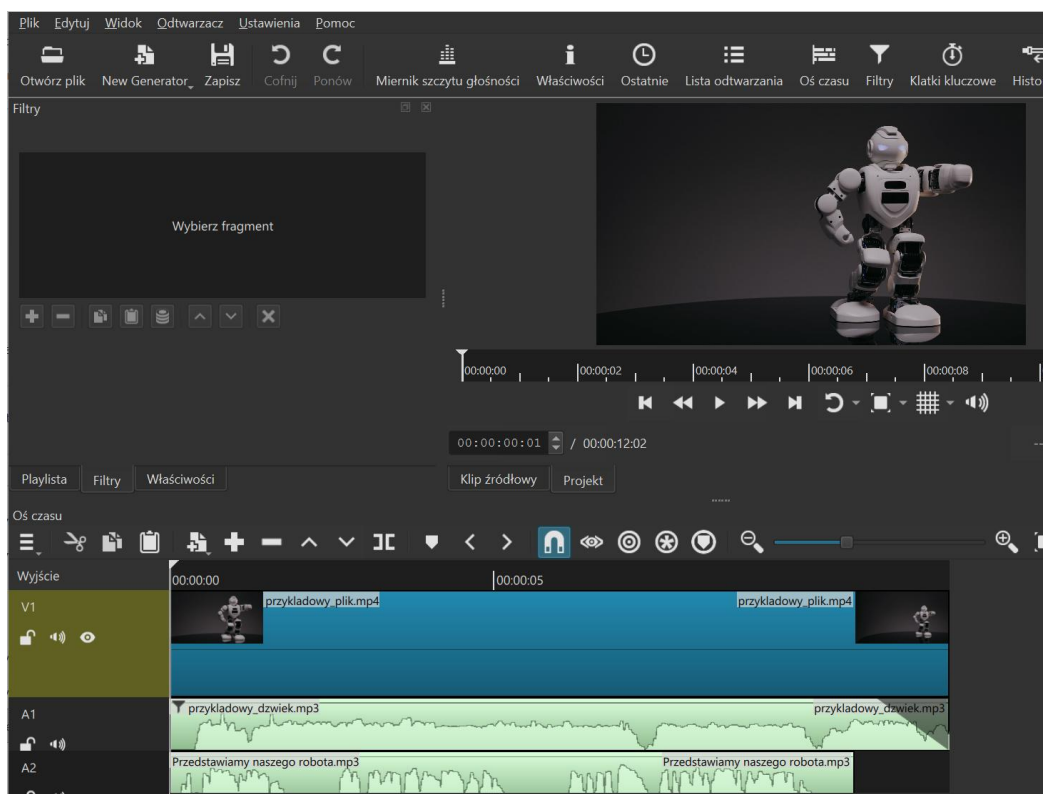
Podczas tej lekcji zajmiemy się przygotowaniem gotowego filmu do publikacji i odtwarzania na różnych urządzeniach oraz platformach internetowych. Samo zmontowanie filmu to jeszcze nie koniec pracy – bardzo ważne jest także odpowiednie dobranie ustawień eksportu i renderowania. Od nich zależy jakość obrazu, płynność ruchu, wielkość pliku oraz to, czy film będzie poprawnie działał na YouTube, TikToku, komputerze lub smartfonie.

W trakcie zajęć zobaczycie, jak korzystać z ustawień eksportu w programie Shotcut oraz jak zmieniać parametry takie jak rozdzielczość, liczba klatek na sekundę czy jakość obrazu i dźwięku. Nauczycie się również przygotowywać różne wersje filmu dopasowane do konkretnych platform. Dzięki temu przekonacie się, że odpowiednie ustawienia eksportu mają duży wpływ na wygląd i wygodę oglądania gotowego filmu.

Scenariusz

1. Uruchomcie program Shotcut i otwórzcie projekt wykonany na wcześniejszych lekcjach. Możecie wykorzystać film z robotem zawierający dźwięk oraz efekty.

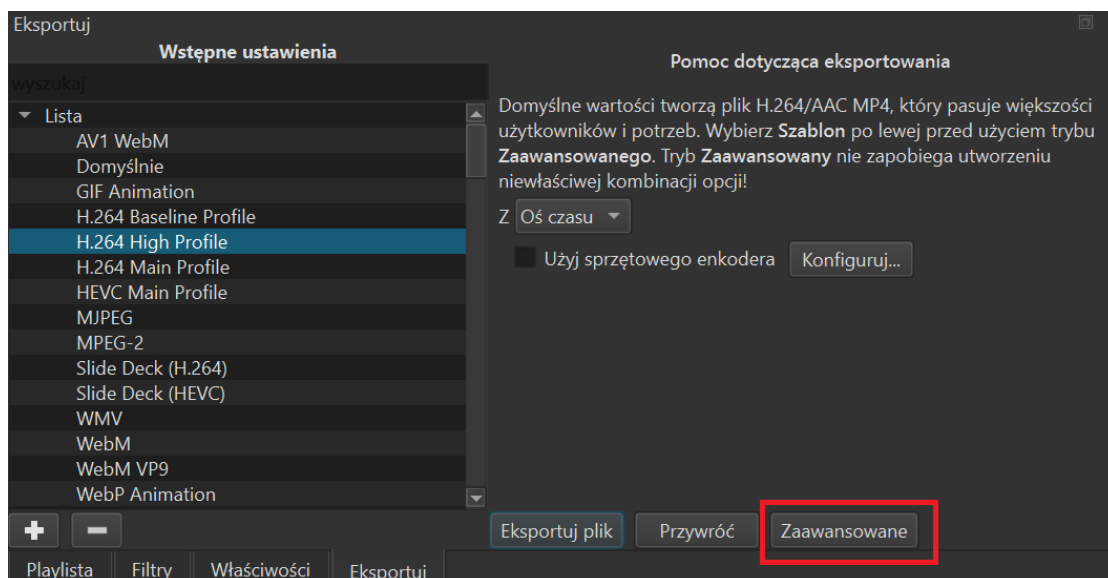
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Rysunek 1. Przykładowy projekt.

2. Odtwórzcie fragment filmu i zastanówcie się, gdzie taki materiał mógłby zostać opublikowany: YouTube, TikTok lub Shorts, komputer lub telewizor, komunikator lub strona internetowa. Zwróćcie uwagę, że każda z tych platform może wymagać innych ustawień eksportu filmu, np. innych proporcji obrazu, jakości, rozdzielczości lub rozmiaru pliku. Dlatego przed renderowaniem warto dopasować ustawienia do miejsca, w którym film będzie publikowany.
3. Po kliknięciu zakładki *Eksportuj* zobaczycie panel *Wstępne ustawienia*. Znajdują się tam gotowe profile eksportu przygotowane do różnych zastosowań i formatów plików. Dzięki nim nie trzeba samodzielnie ustawiać wszystkich parametrów – program automatycznie dobiera odpowiednie ustawienia jakości, kodeka i rozdzielczości.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

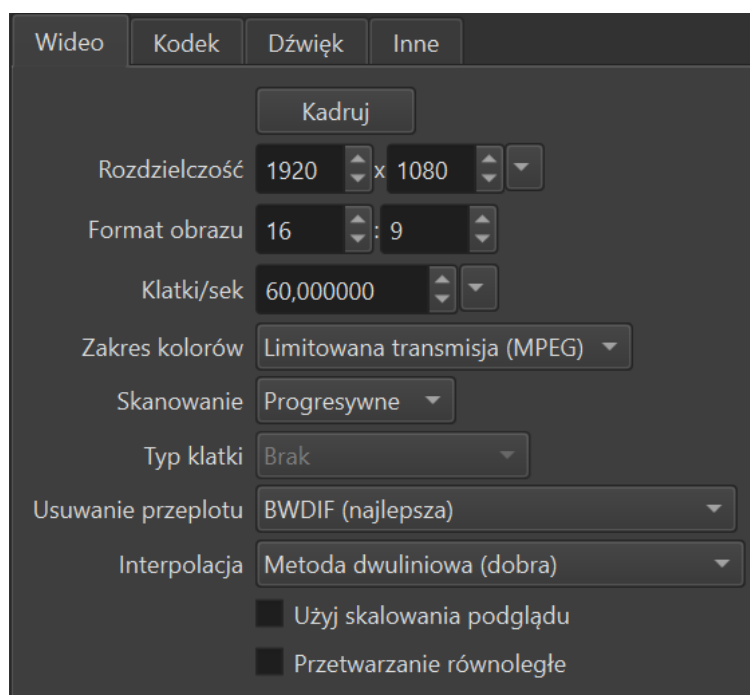


Rysunek 4. Wybór opcji zaawansowanej dla eksportu.

7. Znajdziecie tam najważniejsze ustawienia eksportu filmu dla sekcji wideo:

- *Rozdzielczość* – określa wielkość obrazu, np. 1920 × 1080 oznacza Full HD,
- *Format obrazu* – proporcje filmu, np. 16:9 dla YouTube lub 9:16 dla TikToka,
- *Klatki/sek (fps)* – liczba klatek wyświetlanych w ciągu sekundy; większa wartość daje płynniejszy ruch,
- *Zakres kolorów* – sposób zapisu kolorów w filmie; najczęściej pozostawia się ustawienie *domyślne*,
- *Skanowanie* – sposób wyświetlania obrazu; ustawienie Progresywne jest najlepsze dla nowoczesnych ekranów i internetu,
- *Typ klatki* – określa sposób zapisu klatek filmu; zazwyczaj pozostawia się ustawienie domyślne,
- *Usuwanie przeplotu* – poprawia wygląd starszych nagrań z przeplotem; opcja BWDIF daje dobrą jakość obrazu,
- *Interpolacja* – sposób skalowania obrazu podczas zmiany rozdzielczości; ustawienie Metoda dwuliniowa jest szybkie i daje dobrą jakość,
- *Użyj skalowania podglądu* – pozwala wykorzystać ustawienia podglądu podczas eksportu,
- *Przetwarzanie równoległe* – przyspiesza renderowanie, wykorzystując kilka rdzeni procesora jednocześnie.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

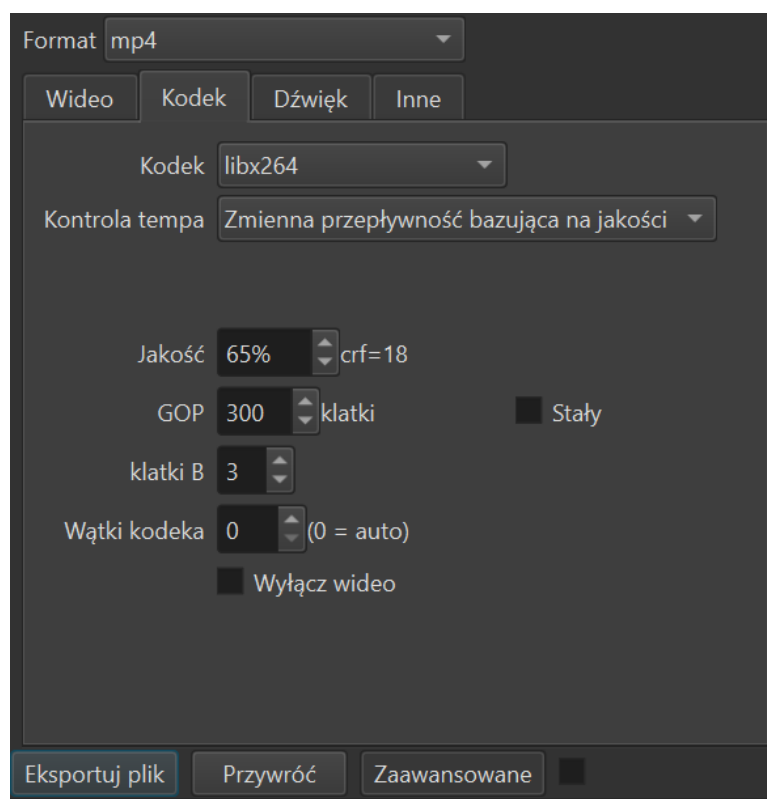


Rysunek 5. Opcje zaawansowane eksportowania.

8. Przejdźcie teraz do zakładki *Kodek*. Znajdują się tam ustawienia odpowiedzialne za sposób kompresji filmu, czyli za to, jak program zapisuje obraz do gotowego pliku. To właśnie te opcje wpływają na jakość filmu, wielkość pliku oraz szybkość renderowania. W tej sekcji znajdziecie m.in.:

- *Kodek* – określa sposób kodowania filmu. Najczęściej używany jest *libx264*, ponieważ zapewnia dobrą jakość obrazu i działa na większości urządzeń oraz platform internetowych,
- *Kontrola tempa* – sposób zarządzania jakością i wielkością pliku. Ustawienie *Zmienna przepływność bazująca na jakości* automatycznie dopasowuje ilość danych do scen filmu, dzięki czemu obraz wygląda lepiej,
- *Jakość* – wpływa na wygląd filmu i rozmiar pliku. Wyższa wartość daje lepszą jakość obrazu, ale zwiększa wielkość filmu i wydłuża renderowanie,
- *GOP* – liczba klatek pomiędzy klatkami kluczowymi. Mniejsze wartości mogą poprawić płynność przewijania filmu, ale zwiększają rozmiar pliku,
- *Klatki B* – dodatkowe klatki pomagające lepiej kompresować film. Większa liczba może zmniejszyć rozmiar pliku bez dużej utraty jakości,
- *Wątki kodeka* – liczba rdzeni procesora używanych podczas renderowania. Ustawienie *0 (auto)* pozwala programowi automatycznie dobrać najlepszą wartość,
- *Wyłącz wideo* – opcja eksportu samego dźwięku bez obrazu.

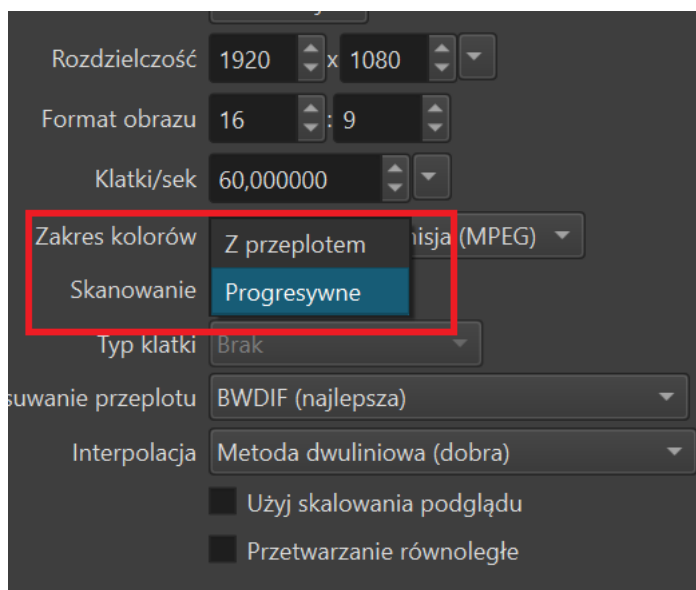
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Rysunek 6. Opcje zakładki kodek.

9. Odszukajcie parametr *Jakość* i spróbujcie zmienić jego wartość. Zwróćcie uwagę, że wyższa jakość daje lepszy obraz, ale zwiększa rozmiar pliku oraz wydłuża renderowanie. Niższa jakość zmniejsza wielkość filmu, ale może powodować utratę szczegółów i gorszy wygląd obrazu. Ustawcie najpierw bardzo wysoką jakość eksportu i sprawdźcie przewidywany rozmiar pliku, a następnie zmniejszcie jakość i porównajcie wyniki.
10. Wróćcie do sekcji *Wideo* i zwróćcie uwagę na parametr *Skanowanie*. Możecie tam wybrać m.in. ustawienie *Progresywne* – nowoczesny sposób wyświetlania obrazu używany na większości współczesnych ekranów – oraz *Przeplot*, czyli starszą metodę stosowaną dawniej w telewizji. Wyeksportujcie krótki fragment filmu w obu ustawieniach, a następnie otwórzcie gotowe pliki i porównajcie obraz. Zwróćcie uwagę, które ustawienie wygląda lepiej na nowoczesnym monitorze i dlaczego najczęściej używa się trybu *Progresywnego* podczas publikacji filmów w internecie.

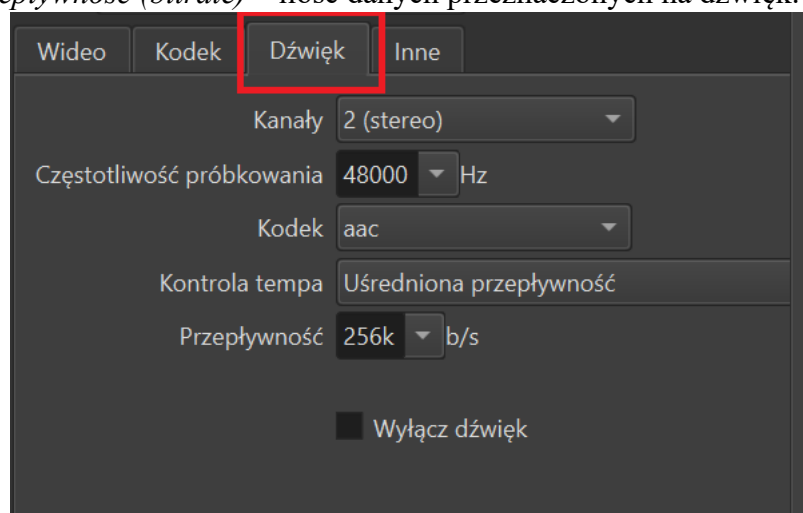
Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Rysunek 7. Ustawienia skanowania obrazu podczas eksportu filmu.

11. Następnie przejdźcie do zakładki *Dźwięk*. Znajdują się tam ustawienia odpowiedzialne za jakość eksportowanego audio. Możecie zmienić m.in.:

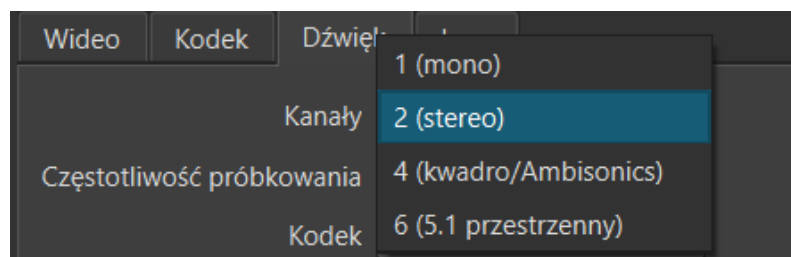
- *Kanały* – wybór dźwięku mono lub stereo,
- *Częstotliwość próbkowania* – wpływa na jakość dźwięku (najczęściej używa się 48000 Hz),
- *Kodek* – sposób kompresji dźwięku, np. AAC,
- *Kontrola tempa* – sposób zarządzania jakością audio,
- *Przepływność (bitrate)* – ilość danych przeznaczonych na dźwięk.



Rysunek 8. Ustawienia eksportu dźwięku w programie Shotcut.

12. W zakładce *Dźwięk* odszukajcie parametr *Kanały*. Sprawdźcie różnicę pomiędzy ustawieniem *Mono* a *Stereo*. Dźwięk mono odtwarzany jest jednakowo w obu głośnikach, natomiast stereo pozwala uzyskać bardziej przestrzenny efekt audio.

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.



Rysunek 9. Ustawienia kanałów dźwięku podczas eksportu filmu.

13. W zakładce *Dźwięk* odszukajcie parametr *Kanały*. Określa on, w jaki sposób będzie odtwarzany dźwięk w gotowym filmie:

- *1 (mono)* – ten sam dźwięk odtwarzany jest w obu głośnikach,
- *2 (stereo)* – dźwięk może być różny w lewym i prawym kanale, co daje bardziej przestrzenny efekt,
- *4 (kwadro/Ambisonics)* – dźwięk wielokanałowy używany w bardziej zaawansowanych nagraniach,
- *6 (5.1 przestrzenny)* – dźwięk przestrzenny stosowany np. w kinie domowym.

Najczęściej w filmach internetowych używa się ustawienia *Stereo*. Wyeksportujcie krótki fragment filmu raz w trybie *Mono*, a następnie w trybie *Stereo*. Otwórzcie oba pliki i porównajcie brzmienie dźwięku, najlepiej używając słuchawek. Wyeksportujcie krótki fragment filmu raz w trybie *Mono*, a następnie w trybie *Stereo*. Otwórzcie oba pliki i porównajcie brzmienie dźwięku, najlepiej używając słuchawek.

14. Następnie zwróćcie uwagę na parametr *Częstotliwość próbkowania*. Najczęściej używa się wartości *48000 Hz*, ponieważ zapewnia dobrą jakość dźwięku w filmach. Spróbujcie ustawić niższą wartość i sprawdźcie, czy zauważacie różnicę w jakości audio.

15. Wróćcie do głównego panelu eksportu i przygotujcie trzy różne wersje filmu i przygotujcie pierwszą wersję filmu przeznaczoną do publikacji na *YouTube*. Ustawcie:

- rozdzielczość *1920 × 1080 (Full HD)*,
- proporcje obrazu *16:9*,
- *60 fps* dla płynniejszego ruchu,
- wysoką jakość eksportu (np. około *70–80%*),
- dźwięk *Stereo* z przepływnością *256 kb/s*.

Taka konfiguracja zapewni dobrą jakość obrazu i dźwięku podczas oglądania filmu na komputerach, telewizorach oraz w serwisie *YouTube*. Kliknijcie *Eksportuj plik* i zapiszcie pierwszą wersję filmu przeznaczoną na *YouTube*, nadając jej nazwę, np. *Film_YouTube.mp4*.

16. Teraz przygotujcie drugą wersję filmu przeznaczoną do *TikToka* lub *YouTube Shorts*. Zmieńcie ustawienia na:

- rozdzielczość *1080 × 1920*,
- proporcje obrazu *9:16* (film pionowy),
- *30 fps* lub *60 fps*,
- średnią jakość eksportu (np. *60–70%*),

Projekt pn. „Rap STEAM — robotyka i programowanie w szkołach podstawowych z terenu województwa podkarpackiego”, realizowanego w ramach programu regionalnego Fundusze Europejskie dla Podkarpacia 2021-2027, współfinansowanego ze środków Unii Europejskiej, z Europejskiego Funduszu Społecznego Plus.

- dźwięk *Stereo*.

Po zmianie ustawień sprawdźcie w podglądzie, czy najważniejsze elementy filmu mieszczą się poprawnie w pionowym układzie obrazu. Następnie wykonajcie eksport drugiej wersji filmu przeznaczonej do TikToka lub Shorts i zapiszcie ją pod nazwą, np. *Film_TikTok.mp4*.

17. Poczekajcie na zakończenie renderowania obu plików i obserwujcie pasek postępu oraz czas potrzebny do eksportu filmu. Sprawdźcie, czy wyższa jakość obrazu lub większa liczba klatek na sekundę wpływa na długość renderowania. Następnie otwórzcie oba gotowe filmy i porównajcie ich proporcje obrazu, płynność ruchu, jakość obrazu i dźwięku oraz wielkość pliku. Zastanówcie się, dlaczego różne platformy wymagają innych ustawień eksportu i czy zawsze warto używać najwyższej możliwej jakości. Na zakończenie zapiszcie projekt wybierając *Plik, Zapisz jako*.

Praktyczne ćwiczenie:

Otwórz w programie Shotcut projekt wykonany na lekcji lub skorzystaj z przykładowego filmu udostępnionego przez nauczyciela. Sprawdź, czy projekt zawiera klip wideo, napisy oraz ścieżkę dźwiękową.

Przejdź do zakładki *Eksportuj* i przygotuj pierwszą wersję filmu przeznaczoną do publikacji na YouTube. Ustaw rozdzielczość 1920×1080 (*Full HD*), proporcje $16:9$, wysoką jakość obrazu oraz dźwięk stereo. Następnie wyeksportuj film i zapisz go pod nazwą *Film_YouTube.mp4*.

Teraz przygotuj drugą wersję filmu przeznaczoną do TikToka lub Shorts. Zmień proporcje obrazu na $9:16$ oraz rozdzielczość na 1080×1920 . Sprawdź w podglądzie, czy najważniejsze elementy filmu są dobrze widoczne w pionowym układzie. Następnie wyeksportuj film pod nazwą *Film_TikTok.mp4*.

W kolejnym kroku przygotuj trzecią wersję filmu o mniejszym rozmiarze pliku, przeznaczoną do przesłania przez internet lub komunikator. Zmniejsz rozdzielczość do 1280×720 (*HD*) oraz obniż jakość eksportu i przepływność dźwięku. Wyeksportuj plik jako *Film_Internet.mp4*.

Po zakończeniu renderowania otwórz wszystkie trzy filmy i porównaj ich wygląd. Zwróć uwagę na jakość obrazu, płynność ruchu, jakość dźwięku oraz wielkość plików. Zastanów się, które ustawienia najlepiej sprawdzają się dla różnych platform i dlaczego nie zawsze warto używać maksymalnej jakości eksportu.

Na koniec zapisz projekt, wybierając *Plik, Zapisz jako*.