Konstruktorzy gier

Autorzy: Grzegorz Zawistowski, Maciej Wojnicki

Lekcja 11 i 12:
Projekt: gra "strzelnica"

W czasie tej lekcji uczniowie zbudują własne urządzenie do gry oraz stworzą do niego oprogramowanie.

Cele lekcji:
Uczeń porafi:
- Zbudować urządzenie elektro-mechaniczne korzystając z instrukcji i gotowych podzespołów.
- Prawidłowo podłączyć urządzenia wejścia wyjścia (czujnik, dioda, serwomotor, wyświetlacz) do sterowania Arduino za pośrednictwem adaptera LOFI brain.
- Wyjaśnić, na czym polega istota zaplanowanej gry.
- Określić cele gry, zmienne i warunki, jakie muszą znaleźć się w programie.
- Zaplanować szkic programu w Arduino IDE służący osiągnięciu konkretnego celu.
- Napisać program będący w pełni funkcjonalną grą.
- Zapisywać, weryfikować, wgrywać, testować i poprawiać program.
- Wyjaśnić zasadę działania programu, analizując poszczególne linie kodu.
- Rozbudować swoją grę o nowe funkcje.

Materiały pomocnicze:
- zestaw LOFI Robot CODEBOX Starter z rozszerzeniem CODEBOX Tv,
- komputery stacjonarne lub przenośne z zainstalowanym Arduino IDE
- komputer nauczyciela z zainstalowanym Arduino IDE, projektor, tablica projekcyjna

Pojęcia kluczowe:
→ dioda RGB → projekt → wyświetlacz RGB 8×8 → czujnik natężenia światła → sterownik Arduino / płytka → adapter LOFI Brain → funkcje (setup, loop, write, delay, read, buzzer, distance,-servo, display.begin, display.show, displayClear, displayNumber)

Czas realizacji: 90 min. lub 135 min. (2-3 godziny lekcyjne)

Metody pracy:
- wykład problemowy,
- dyskusja dydaktyczna związana z wykładem,
- pokaz,
- projekt.

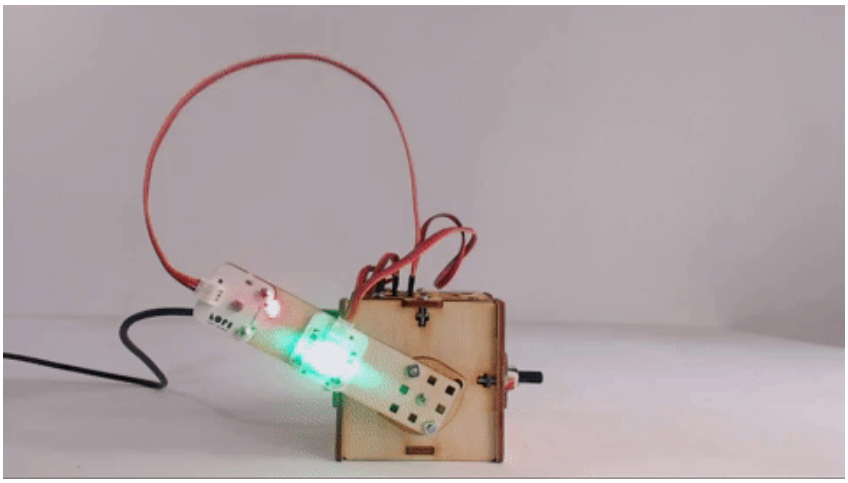
Treści programowe:
Podstawa programowa kształcenia ogólnego dla szkół podstawowych – II etap edukacyjny – klasy VII-VIII, informatyka:
I. Rozumienie, analizowanie i rozwiązywanie problemów. Uczeń:
1) formułuje problem w postaci specyfikacji (czyli opisuje dane i wyniki) i wyróżnia kroki w algorytmicznym rozwiązywaniu problemów. Stosuje różne sposoby przedstawiania algorytmów, w tym w języku naturalnym, w postaci schematów blokowych, listy kroków;
2) stosuje przy rozwiązywaniu problemów podstawowe algorytmy:
a) na liczbach naturalnych: bada podzielność liczb, wyodrębnia cyfry danej liczby, przedstawia działanie algorytmu Euklidesa w obu wersjach iteracyjnych (z odejmowaniem i z resztą z dzielenia),
4) rozwija znajomość algorytmów i wykonuje eksperymenty z algorytmami, korzystając z pomocy dydaktycznych lub dostępnego oprogramowania do demonstracji działania algorytmów;
II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera i innych urządzeń cyfrowych. Uczeń:
1) projektuje, tworzy i testuje programy w procesie rozwiązywania problemów. W programach stosuje: instrukcje wejścia/wyjścia, wyrażenia arytmetyczne i logiczne, instrukcje warunkowe, instrukcje iteracyjne, funkcje oraz zmienne i tablice.
2) projektuje, tworzy i testuje oprogramowanie sterujące robotem lub innym obiektem na ekranie lub w rzeczywistości;
5) wyszukuje w sieci informacje potrzebne do realizacji wykonywanego zadania,stosując złożone postaci zapytań i korzysta z zaawansowanych możliwości wyszukiwarek.
III. Posługiwanie się komputerem, urządzeniami cyfrowymi i sieciami komputerowymi. Uczeń:
3) poprawnie posługuje się terminologią związaną z informatyką i technologią.
IV. Rozwijanie kompetencji społecznych. Uczeń:
1) bierze udział w różnych formach współpracy, jak: programowanie w parach lub w zespole, realizacja projektów, uczestnictwo w zorganizowanej grupie uczących się, projektuje, tworzy i prezentuje efekty wspólnej pracy;

Wprowadzenie w tematykę i integracja grupy (5 min.)

Pytamy uczniów, co robiliśmy podczas ostatniej lekcji?

- dowiedzieliśmy się czym jest serwomotor i do czego może być wykorzystany,
- nauczyliśmy się sterować serwomotorem.

Dziś skonstruujecie urządzenie elektroniczne na bazie Arduino, serwomotoru i czujnika natężenia światła oraz napiszecie program – grę na to urządzenie.



Wyświetl GIF: **https://gph.is/2Ki3PNu**

Część zasadnicza (80 min.)

Do realizacji projektu potrzebować będziemy:

- LOFI Brain,
- wyświetlacz LED 8×8 pikseli RGB,
- serwomotor,
- diodę,
- czujnik światła,
- kabel USB,
- elementy konstrukcyjne z zestawu LOFI Robot CODEBOX,
- wskaźnik laserowy (najczęściej spotykany w formie wskaźnika do prezentacji lub zabawki dla kotów), który nie wchodzi w skład zestawu CODEBOX.

Projekt: część 1

Konstrukcja robota -czas potrzebny na wykonanie tej części to około 30 minut

Instrukcja dostępna na stronie: **http://www.lofirobot.com/codebox/strzelnica-instrukcja-montazu/**

Projekt: część 2

Programowanie - czas potrzebny na wykonanie tej części to około 40 minut

Opis działania programu:

Strzelnica to bardzo prosta, ale jednocześnie dająca dużo zabawy gra. Polega na trafianiu w cel przy pomocy wskaźnika laserowego. Po trafieniu wskaźnikiem w czujnik natężenia światła, ramię robota - dzięki silnikowi serwo - obróci się do innej, losowo wybranej pozycji oraz zasygnalizuje trafienie poprzez zapalenie diody i włączenie buzzera. Jednocześnie na wyświetlaczu RGB 8×8 wyświetlany będzie wynik.

W Arduino IDE wczytaj pusty szablon: **Plik > Przykłady > LOFI > lofi_pusty_szablon**

Zanim przystąpisz do pisania kodu zapisz plik jako **“lofi_gra_strzelnica”**.

Do naszej gry będziemy potrzebowali 3 zmiennych: pozycja, punkty i światło. Zmienne musimy zadeklarować na samym początku programu, zaraz po deklaracji importowania biblioteki LOFI:

- ramię robota na początku programu powinno być na środku, dla tego zmiennej “pozycja” przy deklaracji na początku programu przypisujemy wartość 50:

  int pozycja = 50;

- w każdej grze, w której zdobywamy punkty, powinniśmy pamiętać, aby zmiennej “punkty” na początku po uruchomieniu programu przypisać wartość 0:

  int punkty = 0;

- potrzebna też będzie zmienna, w której będzie przechowy odczyt z czujnika natężenia światła, nie musimy jej przypisywać żadnej wartości początkowej:

  int swiatlo;

Do funkcji void setup() należy dodać następujące linijki kodu:

- funkcję inicjującą otwarcie portu szeregowego i ustawienie prędkości na 9600 bps

  Serial.begin(9600)

- dwie funkcje odpowiedzialne za inicjalizowanie wyświetlacza LOFI RGB 8×8:

  robot.displayBegin()

  robot.displayShow()

- funkcję, która jeden raz po uruchomieniu programu ustawi serwomotor w konkretnej pozycji:

  robot.servo(OUTPUT1, pozycja)

Uwzględniając powyższe założenia szkielet programu powinien wyglądać następująco:

```
#include <LOFI.h>
LOFI robot;

int pozycja = 50;
int punkty = 0;
int swiatlo;

void setup() {
robot.displayBegin();
robot.displayShow();
robot.servo(OUTPUT1, pozycja);
}

void loop() {
}
```

Teraz zastanówmy się nad działaniem programu. Podstawowym założeniem jest celowanie wskaźnikiem laserowym w czujnik natężenia światła przykręcony do ramienia serwomotoru. W głównej pętli programu – w pętli loop() musimy więc w pierwszej kolejności odczytać wartość z tego czujnika. Jakiej funkcji do tego użyjemy? Znanej nam **read(wejście);**

```
void loop() {
swiatlo = robot.read(INPUT1);
}
```

W tym momencie program odczytuje już wartość z czujnika natężenia światła podpiętego do **INPUT1** i podstawia ją na bieżąco jako zmienną **swiatlo**.

Możemy podejrzeć, jakie wartości z czujnika uzyskujemy. Do tego celu użyjemy funkcji **Serial.println(zmienna)** inicjującej wyświetlanie danych odbieranych z płytki Arduino, w oknie monitora portu szeregowego:

```
1  void loop() {
2  swiatlo = robot.read(INPUT1);
3  Serial.println(swiatlo);
4  }
```

Wgrywamy tak przygotowany program na Arduino i włączamy w Arduino IDE szeregowy monitor. Upewnijmy się, czy w oknie monitora włączona jest funkcja Autoscroll oraz wybrana jest szybkość 9600.

Jakie wyniki otrzymujemy? Czy wyniki przekraczają wartość 50? Prawdopodobnie nie.

A teraz spróbujcie zaświecić wskaźnikiem laserowym na czujnik. Jakie wyniki otrzymujemy w szeregowym monitorze? Powyżej 50? 75?

Możemy więc przyjąć, że jeśli wartość będzie mniejsza niż 50, to uznajemy że nie trafiono laserem w czujnik. I program nic nie musi wykonywać.

Instrukcja warunkowa: gdy wartość zmiennej swiatlo będzie wyższa od 50, czyli gdy trafimy strumieniem światła w czujnik, to:

1. Ramię serwomotoru powinno obrócić się do nowej, losowej pozycji.

   Trzeba skorzystać z funkcji **random(od,do)** która zwraca wartość losową z żądanego zakresu i przypisać ją do zmiennej **pozycja**. Następnie trzeba tę zmienną wykorzystać w funkcji **servo(wyjście, wartość);** jako **wartość**.

2. Dioda OUTPUT4 powinna się zaświecić.

   Do tego celu skorzystamy z funkcji **write(wyjście, wartość)**; jako wartość wpisując 100, co oznacza pełną moc świecenia diody.

3. Liczba punktów powinna się zwiększyć o 1.

   Zmiennej **punkty** należy przypisać wartość sumy jej samej i liczby 1.

4. Na wyświetlaczu powinna zostać wyświetlona liczba punktów.

   Chcąc wyświetlić aktualny wynik, najpierw trzeba wymazać poprzedni za pomocą funkcji **displayClear()**, a następnie wysłać na wyświetlacz informację o aktualnym wyniku (wartość zmiennej punkty) za pomocą funkcji **displayNumber(liczba)**; oraz wydać polecenie do pokazania danych na wyświetlaczu za pomocą funkcji **display.show()**

5. Powinniśmy dać programowi chwilę oczekiwania, aby serwomotor zdążył zmienić pozycję.

   Tu skorzystamy ze znanej nam funkcji **delay(czas w milisekundach)** dając np 1000 milisekund przerwy.

6. Dioda powinna zgasnąć.

   Jeśli dioda ma zgasnąć, to nie zapomnijmy o funkcji **write(wyjście, wartość)** jako wartość wpisując 0, co oznacza wyłączenie diody.

Po pierwsze musimy jednak dać instrukcję warunkową **if()**, w której zamieścimy wyżej opisane 6 czynności:

```
if (swiatlo > 50) {
// warunek sprawdzający kiedy zmienna swiatlo będzie większa od 50,
// czyli kiedy gracz wycelował strumieniem lasera w czujnik
pozycja = random(0,100); //Ad. 1
robot.servo(OUTPUT1, pozycja); //Ad. 1
robot.write(OUTPUT4,100); //Ad. 2
punkty = punkty + 1; //Ad. 3
robot.displayClear(); //Ad. 4
robot.displayNumber(punkty); //Ad. 4
robot.displayShow(); //Ad. 4
delay(1000); //Ad. 5
robot.write(OUTPUT4,0); //Ad. 6
}
```

Po napisaniu całego programu weryfikujemy go, zapisujemy i wgrywamy na płytkę Arduino. Bawiąc się sprawdzamy czy działa. W razie potrzeby dokonujemy poprawek i ponownie wgrywamy.

Przykładowy kod gotowej gry (bez komentarzy):

```
#include <LOFI.h>
LOFI robot;

int pozycja = 50;
int punkty = 0;
int swiatlo;

void setup() {
Serial.begin(9600);
robot.displayBegin();
robot.servo(OUTPUT1, pozycja);
}


void loop() {
  swiatlo = robot.read(INPUT1);
  Serial.println(swiatlo);

  if (swiatlo > 50) {
  pozycja = random(0,100);
  robot.servo(OUTPUT1, pozycja);
  robot.write(OUTPUT4,100);
  punkty = punkty + 1;
  robot.displayClear();
  robot.displayNumber(punkty);
  robot.displayShow();
  delay(1000);
  robot.write(OUTPUT4,0);
  }
}
```

Jeśli zostanie czas, możemy z uczniami zastanowić się, jak rozbudować projekt? Można dodać przycisk, który będzie resetował wynik itp. Powodzenia!

Podsumowanie i ewaluacja (5 min.)

Prosimy, aby uczniowie ostrożnie spakowali zestawy. Jeden przedstawiciel każdej grupy przynosi zestaw na wyznaczone przez nauczyciela miejsce w klasie.

Zadajemy uczniom pytania: Co najbardziej podobało się Wam w czasie dzisiejszej lekcji? Na jakie problemy napotkaliście podczas realizacji projektu? Co było najtrudniejsze? Z czym nie było problemów?

Na zakończenie mówimy uczniom, że podczas kolejnej lekcji zajmiemy się projektem innej gry. Stworzymy grę "wyścigi". Będzie sprawdzała szybkość i refleks. Będzie to gra dla dwóch zawodników.