Konstruktorzy gier

Autorzy: Grzegorz Zawistowski, Maciej Wojnicki

Lekcja 13 i 14:
Projekt: gra "wyścigi"

W czasie tej lekcji uczniowie zbudują kolejne urządzenie raz stworzą do niego oprogramowanie. Gra "wyścigi" dedykowana jest dla dwóch graczy i sprawdza naszą szybkość i refleks.

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 → projekt → wyświetlacz RGB 8×8 → czujnik natężenia światła → przycisk → sterownik Arduino / płytka → adapter LOFI Brain → funkcje (setup, loop, write, delay, read, buzzer, distance,servo, display.begin, display.show, displayClear, displayNumber, rectangle. color, rainbowColor) → zmienna liczbowa, zmienna boolen / bool

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

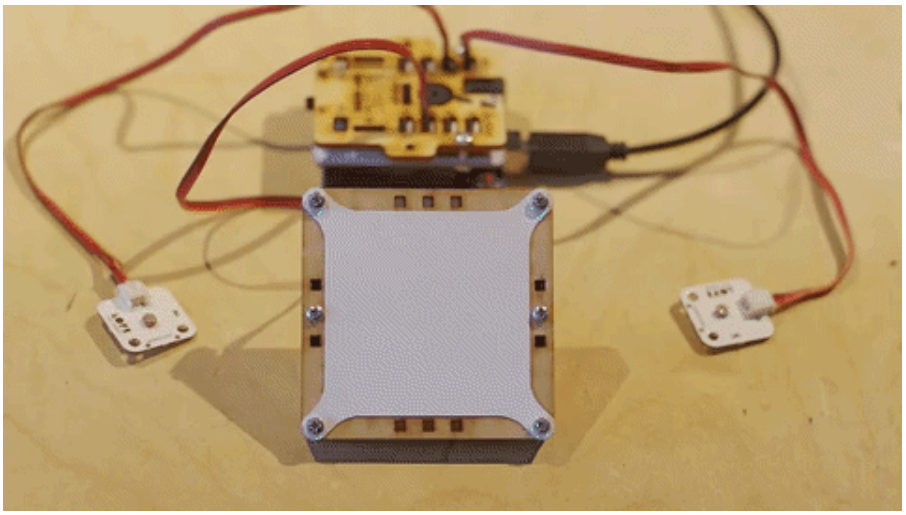
Metody pracy:
- wykład problemowy,
- dyskusja dydaktyczna związana z wykładem,
- pokaz,
- projekt.

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.)

Prosimy uczniów, aby podzielili się swoimi wrażeniami z ostatniego projektu. Co najbardziej im się podobało? Jak można byłoby udoskonalić strzelnicę? Czy mają pomysł na jakąś podobną grę, bazując na strzelnicy?

Mówimy, że podczas tej lekcji (która może potrwać od 2 do 3 godzin) zajmiemy się projektem innej gry. Stworzymy grę "wyścigi". Będzie sprawdzała szybkość i refleks. Będzie to gra dla dwóch zawodników.



Wyświetl GIF: https://gph.is/2yLqmxQ

Część zasadnicza (80 min.)

Do realizacji projektu potrzebować będziemy:

- LOFI Brain,
- wyświetlacz LED 8×8 pikseli RGB,
- przycisk,
- 2 czujniki światła,
- 2 diody LED,
- kabel USB,
- elementy konstrukcyjne z zestawu LOFI Robot CODEBOX.

Projekt: część 1

Konstrukcja robota -czas potrzebny na wykonanie tej części to około 15 minut

Instrukcja złożenia urządzenia:

- czujnik światła → port INPUT1 – sterowanie dla gracza nr 1
- czujnik światła → port INPUT2 – sterowanie dla gracza nr 2
- przycisk → INPUT3 – do resetowaniw gry
- czerwona dioda LED → port OUTPUT1 – sygnalizacja dla gracza nr 1
- zielona dioda LED → port OUTPUT2 – sygnalizacja dla gracza nr 2
- wyświetlacz → OUTPUT3 – do obserwowania postępów
- zasilanie robota → kabel USB → wyjście USB w komputerze

Projekt: część 2

Programowanie - czas potrzebny na wykonanie tej części to około 65 minut

Opis działania programu:

Dwóch zawodników zasłania i odsłania dłońmi jak najszybciej czujnik światła. Postępy wyświetlane są na ekranie LED 8×8 przy pomocy dwóch pasków postępu (czerwony i zielony). Wygrywa ten zawodnik, który szybciej macha ręką i którego pasek pierwszy dojdzie do końca ekranu.

1. Zmienne w grze:

a) liczbowe deklarowane na początku szkicu, przed pętlą setup()

- dwie do odczytywania wartości z czujnika światła (po jednej dla każdego gracza), np. foto1, foto2;
- jedna do ustawienia poziomu "czułości", np. czulosc;
- dwie do określania stopnia postępu / poziomu (po jednej dla każdego gracza), np. level1, level2;
- jedna na przechowanie wyniku o numerze zwycięzcy, np. winner

b) inne

- dwie typu boolen do przechowywania wartości prawda/fałsz, oznaczające zakrycie lub odkrycie czujnika (przyjmijmy, że false = odkryty, true = zakryty);

bool foto1_pressed = false;

bool foto2_pressed = false;

2. Instrukcje warunkowe:

a) Zdobywanie punktów – jeżeli czujnik jest odkryty (zmienna bool = false) i gracz zakryje ręką czujnik (odczyt z czujnika < czułości) to zaświeci się jego dioda (OUTPUT1), zmienna bool ma zmienić wartość na true (oznaczającą, że czujnik został zakryty), a gracz ma zdobyć punkt.

b) Blokowanie naliczania punktów – gdy trzyma zakryty czujnik po zdobyciu 1 punktu: jeżeli czujnik jest zakryty (zmienna bool = true) i gracz trzyma rękę na czujniku (odczyt z czujnika < czułości) to dioda OUTPUT1 ma się wyłączyć i żadne inne zmienne nie zmieniają się.

c) Odblokowanie możliwości zdobycia punktu gdy gracz odkryje czujnik – jeżeli czujnik jest zakryty (zmienna bool = true) i gracz odkryje czujnik (odczyt z czujnika > czułości) to dioda OUTPUT1 ma się wyłączyć, a zmienna bool ma przyjąć wartość na false (oznaczającą odkrycie czujnika), bez dodawania punktów

d) takie same 3 instrukcje warunkowe należy napisać dla drugiego gracza.

e) Określenie zwycięzcy: jeżeli nikt jeszcze nie został zwycięzcą (winner == 0) i poziom gracza pierwszego > 32 wówczas zwycięzcą ma zostać gracz pierwszy, czyli zmiennej winner przypisujemy wartość 1.

Czemu 32? Maksymalna wartość zmiennej poziom powinna być podzielna przez 8 (bo tyle mamy kolumn na wyświetlaczu), a jednocześnie nie powinna być ani za duża, ani za mała, aby ilość zasłonięć nie była zbyt duża/mała. Np. przy 8 zasłonieniach gra byłaby zbyt krótka, a przy 100 – dłużyła by się. Przy 32, 40, czy 48 zasłonięciach prowadzących do zwycięstwa gra jest przyjemna.

f) Analogiczny warunek dla drugiego gracza.

g) Rysowanie pasków graczy na ekranie – jeśli nikt jeszcze nie został zwycięzcą if (winner == 0), to:

- ustawiamy kolor czerwony dla gracza pierwszego i rysujemy prostokąt od X=0 i Y=0, o szerokości równej ¼ aktualnego poziomu tego gracza oraz wysokości 4,
- ustawiamy kolor zielony dla gracza drugiego i rysujemy prostokąt od X=0 i Y=4, o szerokości równej ¼ aktualnego poziomu tego gracza oraz wysokości 4,

Dopiero później wywołujemy wyświetlanie grafiki na wyświetlaczu funkcją **display.show()**

h) Reset gry – jeśli wartość odczytu z wejścia INPUT3 będzie większa od 90, to zmiennym: winner, level1 i level2 przypisuje się wartość 0, a wyświetlacz zostaje wyczyszczony funkcją displayClear();

Przykładowy kod gotowej gry

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

int foto1;
int foto2;
int czulosc = 15;

bool foto1_pressed = false;
bool foto2_pressed = false;

int level1 = 0;
int level2 = 0;
int winner = 0;

void setup() {
Serial.begin(9600);
robot.displayBegin();
robot.displayShow();
}

void loop() {
foto1 = robot.read(INPUT1);
foto2 = robot.read(INPUT2);

// pierwszy gracz:
if (foto1<czulosc && foto1_pressed == false) {
  robot.write(OUTPUT1,100);
  foto1_pressed = true;
  level1 = level1 + 1;
}

if (foto1<czulosc && foto1_pressed == true) {
  robot.write(OUTPUT1,0);
}

if (foto1>czulosc && foto1_pressed == true) {
  robot.write(OUTPUT1,0);
  foto1_pressed = false;
}

// drugi gracz:
if (foto2<czulosc && foto2_pressed == false) {
  robot.write(OUTPUT2,100);
  foto2_pressed = true;
  level2 = level2 + 1;
}

if (foto2<czulosc && foto2_pressed == true) {
  robot.write(OUTPUT2,0);
}

if (foto2>czulosc && foto2_pressed == true) {
  robot.write(OUTPUT2,0);
  foto2_pressed = false;
}

// określenie zwycięzcy
if (winner == 0 && level1>32) {
  winner = 1;
}

if (winner == 0 && level2>32) {
  winner = 2;
}

// rysowanie pasków graczy na ekranie
if (winner == 0)
robot.color(100,0,0);

robot.rectangle(0,0,level1/4,4);
robot.color(0,100,0);
robot.rectangle(0,4,level2/4,4);

if (winner == 1) {
  robot.color(100,00,0);
robot.rectangle(0,0,8,8);
}

if (winner == 2) {
robot.color(0,100,0);
robot.rectangle(0,0,8,8);
}

robot.displayShow();

//reset gry
if (robot.read(INPUT3) > 90) {
  winner = 0;
  level1 = 0;
  level2 = 0;
  robot.displayClear();
}
}
```

Po napisaniu gry, weryfikacji i wgraniu na płytkę, dajemy uczniom trochę czasu na zabawę.

Jeśli zostanie czas możemy z uczniami zastanowić się nad tym jak rozbudować projekt? Powodzenia!

Podsumowanie i ewaluacja (10 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 tym projekcie? Na jakie problemy napotkaliście podczas realizacji projektu? Co było najtrudniejsze? Z czym nie było problemów?

Podajemy przykładowe pomysły na inne gry, jakie można stworzyć dzięki zestawowi LOFI Robot CODEBOX:

- Bramka do piłki nożnej – strzelanie piłką wykrywa styk,
- Gorący ziemniak – czas ucieka trzeba podawać urządzenie z rąk do rąk,
- System alarmowy – przecinanie promieni lasera,
- Utrzymywanie równowagi – porównanie dwóch czujników światła,
- Frogger – żaba przechodzi przez ulicę,
- Liczba parzysta/nieparzysta – na wyświetlaczu pokazywana jest losowa liczba w zakresie 0-99. Użytkownik ma za zadanie jak najszybciej wybrać czy jest parzysta czy nieparzysta, naciskając jeden z dwóch przycisków (jako przyciski używamy fotoresystory). Zawodnik musi odgadnąć jak najwięcej liczb w przeciągu określonego czasu np 30 sek. Jeśli się pomyli, kończy grę przed czasem.