Konstruktorzy gier

Autorzy: Grzegorz Zawistowski, Maciej Wojnicki

Lekcja 3:
Poznajemy składnię kodu Arduino IDE

W czasie tej lekcji uczniowie poznają składnię kodu w środowisku Arduino IDE, nauczą się pisać proste programy, np. sterujące pracą diod, wgrywać je na płytkę Arduino i obserwować efekty.

Cele lekcji:
Uczeń porafi:
- wyjaśnić podstawowe zasady przy pisaniu kodu w Arduino IDE.
- zaimplementować bibliotekę LOFI.
- wyjaśnić, do czego służy funkcja setup() i loop();
- skorzystać z funkcji write(); oraz delay();
- przepisać program z przykładu, zapisać go do pliku, zweryfikować i wgrać na płytkę;
- przeanalizować i wyjaśnić znaczenie poszczególnych linii kodu;
- dokonać drobnych poprawek w kodzie w celu uzyskania konkretnego efektu migania 1 i 2 diod.

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

Pojęcia kluczowe:
→ Arduino IDE → zweryfikuj → wgraj → biblioteka, np. LOFI → funkcja (setup, loop, write, delay)

Czas realizacji: 45 min.

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

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?

- uruchamialiśmy Arduino IDE,
- otwieraliśmy gotowy przykładowe skrypty,
- wgrywaliśmy je na płytkę Arduino UNO,
- podłączaliśmy do płytki urządzenia wejścia i wyjścia oraz testowaliśmy przykładowe programy.

Dziś poznamy składnię kodu w środowisku Arduino IDE. Nauczymy się pisać proste programy sterujące pracą diod. Następnie będziemy wgrywać je na płytkę.

Część zasadnicza (35 min.)

Przedstawiciel każdej grupy uczniów bierze od nauczyciela przypisany danej grupie zestaw. Uczniowie siadają przy komputerach.Prosimy uczniów aby włączyli komputery, zalogowali się i uruchomili Arduino IDE. Od tej pory nauczyciel wyświetla za pomocą projektora zawartość swojego ekranu, wykonuje i omawia następujące czynności oraz prosi uczniów, aby wykonywali po kolei to samo.

Nauczyciel wyjaśnia podstawowe zasady przy pisaniu kodu w Arduino IDE:

- w ArduinoIDE, tak jak w C i C++, wielkości znaków ma znaczenie,
- słowa kluczowe takie jak **if, for, while** zawsze piszemy małymi literami,
- nie używamy polskich znaków,
- na końcu każdej linii stawiamy średnik,
- klamry **{ .. }** służą do oznaczania bloków programu, używamy ich do ograniczenia obszaru funkcji, pętli i instrukcji warunkowych,
- warto dodawać komentarze do treści programu, które nie są interpretowane przez kompilator (komputer będzie je ignorował, ale my jako programiści łatwiej będziemy mogli zrozumieć kod),
- wyróżniamy dwa rodzaje komentarzy: jednoliniowe i wieloliniowe,
- komentarze jednoliniowe rozpoczynamy znakami **//** (podwójny slash),
- komentarze wieloliniowe rozpoczynamy znakami **/*** a kończymy ***/**.
- jeżeli chcemy dołączyć do naszego programu jakąś bibliotekę (np. LOFI), korzystamy z polecenia **include**,
- poniżej w 1. linijce przykład komentarza, a w 2. – dołączenia biblioteki LOFI:

```
1  //deklaracja importowania biblioteki LOFI
2  #include <LOFI.h>
```

Każdy program, który będziemy tworzyć zawiera trzy podstawowe elementy:

- zaimportowanie biblioteki **LOFI** i jej zainicjalizowanie – inicjalizacja obiektu ROBOT będącego instancją klasy LOFI, obiekt ten zawiera metody ułatwiające korzystanie z funkcji sterownika LOFI Brain,
- **SETUP** – funkcja która uruchomi się zaraz po włączeniu sterownika i wykona tylko jeden raz,
- **LOOP** – główna funkcja programu, która wykona się po zakończeniu funkcji SETUP i będzie wykonywać w pętli w nieskończoność (jak w Scratch pętla “Zawsze”).

Omawiamy i pokazujemy budowę podstawowego programu. Mówimy uczniom, że każdy nowy projekt przy uruchomieniu Arduino IDE zawiera już dwie funkcje **(setup()** i **loop())**. Natomiast jeśli chcemy wczytać gotowy poniższy przykład (z deklaracją importowania biblioteki i inicjowania obiektu LOFI), klikamy: **Plik > Przykłady > LOFI > 1_pusty_szablon**

```
#include <LOFI.h> //deklaracja importowania biblioteki LOFI
LOFI robot; //deklaracja obiektu robot będącego instancją klasy LOFI

void setup() {
//tutaj wpisujemy instrukcje, które mikrokontroler wykona jeden raz po uruchomieniu
}

void loop() {
/* tu wpisujemy instrukcje tworzące nasz główny program,
które będą powtarzane w nieskończoność
(dopóki nie wgramy kolejnego programu lub nie odłączymy zasilania od Arduino) */
}
```

Warto wiedzieć! Funkcje **setup()** i **loop()** to funkcje systemowe, które muszą być zdefiniowane w każdym projekcie. Nawet w sytuacji, gdy w jednej z nich nie umieścimy żadnego kodu, to muszą być zadeklarowane obie.

Zadanie 1

Polecenie:

Napisz swój pierwszy program. Jego celem będzie uzyskanie efektu świetlnego – włączanie na pół sekundy i wyłączanie jednej diody.

Rozwiązanie:

- uruchamiamy Arduino IDE,
- klikamy: Plik > Przykłady > LOFI > lofi_pusty_szablon

Wyświetlamy uczniom przykładowe rozwiązanie (bez komentarzy). Prosimy, aby przepisali kod bardzo uważnie (nie pomijając żadnej kropki, średnika czy odstępu):

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

void setup() {
}

void loop() {
robot.write(OUTPUT1, 100);
delay(500);
robot.write(OUTPUT1, 0);
delay(500);
}
```

Po napisaniu programu:

- zapisujemy szkic: **Plik > Zapisz jako >** proponujemy nazwę pliku np. “zadanie_1” (nazwa pliku bez odstępów i polskich znaków, w przeciwnym razie program sam dostosuje nazwę do tej zasady) i zapisujemy w systemowym katalogu Dokumenty w podkatalogu Arduino,
- weryfikujemy program,
- jeśli nie ma błędów, podłączamy płytkę Arduino kablem USB do komputera, upewniamy się czy w Arduino IDE wybrany jest odpowiedni typ płytki (**Narzędzia > Płytka > Arduino/Genuino UNO**) oraz port (**Narzędzia > Port**),
- do wyjścia OUTPUT1 podłączamy diodę,
- wgrywamy program na płytkę,
- obserwujemy efekt migania diody.

Obserwacja: Dioda włącza się na 0,5 sekundy i na 0,5 sekundy wyłącza. Efekt powtarzany jest dopóki płytka Arduino podłączona jest do zasilania.

Analizujemy i wyjaśniamy linijka po linijce:

```
#include <LOFI.h> //deklaracja importowania biblioteki LOFI
LOFI robot; //deklaracja obiektu robot będącego instancją klasy LOFI

void setup() {
}

//jest to jedna z dwóch obowiązkowych funkcji,
//wpisujemy w niej instrukcje, które mikrokontroler wykonać ma jeden raz po uruchomieniu
//lub po resecie Arduino, w naszym przykładzie ma pozostać pusta (ale musi być!),

void loop() {
//jest to druga obowiązkowa funkcja,
// wszystkie polecenia w niej zawarte będą powtarzane w nieskończoność
//aż do odłączenia zasilania od płytki Arduino
robot.write(OUTPUT1, 100);

//odwołanie do jednej ze zdefiniowanych w bibliotece LOFI funkcji o nazwie write,
// która przyjmuje dwa parametry: pierwszy z nich to nazwa wyjścia np. OUTPUT1, OUTPUT2,
//do którego podłączamy urządzenie (tu diodę), drugi to wartość w zakresie od 0 do 100

//w tym przypadku na wyjściu OUTPUT1 pojawić ma się sygnał elektryczny o mocy 100%
delay(500);

//polecenie czekaj, w nawiasie podajemy jeden parametr - czas w milsekundach

//1000 = 1 sekunda

//tu 500 = pół sekundy
robot.write(OUTPUT1, 0);

//funkcja write na wyjściu OUTPUT1 ma wyłączyć zasilanie = dać 0% mocy
delay(500);

//czekaj 500 milisekund czyli pół sekundy
}

// Wszystko co jest po funkcji loop() wewnątrz klamr {...} jest powtarzane, a więc wynikiem
// działania powyższego programu będzie: włączenie diody, odczekanie pół sekundy (dioda się świeci),
// wyłączenie diody, odczekanie pół sekundy (gdy dioda wyłączona) - i powtarzanie tego procesu,
// aż do wyłączenia zasilania płytki.
```

Zadanie 2

Polecenie:

Zmodyfikuj program tak, aby dioda migała trzy razy szybko, z krótkimi przerwami, następnie niech zaświeci się dwa razy na dłużej, z dłuższymi przerwami.

Przykładowe rozwiązanie:

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

void setup() {
}

void loop() {
robot.write(OUTPUT1, 100); //pierwsze włączenie diody

delay(100); //krótki czas świecenia
robot.write(OUTPUT1, 0); //wyłączenie diody
delay(100); //krótki czas przerwy

robot.write(OUTPUT1, 100); //drugie włączenie diody

delay(100);
robot.write(OUTPUT1, 0);
delay(100);

robot.write(OUTPUT1, 100); //trzecie włączenie diody

delay(100);
robot.write(OUTPUT1, 0); //wyłączenie diody
delay(500); //długi czas przerwy

robot.write(OUTPUT1, 100); //czwarte włączenie diody

delay(500); //długi czas świecenia
robot.write(OUTPUT1, 0);
delay(500); //długi czas przerwy

robot.write(OUTPUT1, 100); //piąte włączenie diody

delay(500); //długi czas świecenia
robot.write(OUTPUT1, 0);
delay(500); //długi czas przerwy

}
```

Po napisaniu programu pamiętamy, aby zapisać szkic (**Plik > Zapisz jako > “zadanie_2”)**, zweryfikować, czy nie zawiera błędów, sprawdzić czy płytka Arduino jest podłączona do komputera, upewnić się, czy wybrany jest odpowiedni typ płytki (**Narzędzia > Płytka > Arduino/Genuino UNO**) oraz port (**Narzędzia > Port**), do wyjścia OUTPUT1 podłączyć diodę, wgrać program na płytkę.

Dokonujemy obserwacji. Analizujemy. Jeśli trzeba, dokonujemy poprawek programu i ponownie wgrywamy na płytkę.

Zadanie 3

Polecenie:

Napisz program, który będzie sterował miganiem dwóch diod podłączonych do OUTPUT1 i OUTPUT2. Diody mają migać naprzemiennie (jak światła w pojeździe uprzywilejowanym).

Przykładowe rozwiązanie:

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

void setup() {
}

void loop() {
robot.write(OUTPUT1, 100); //włączenie pierwszej diody

delay(200); //czas świecenia pierwszej diody
robot.write(OUTPUT1, 0); //wyłączenie pierwszej diody
robot.write(OUTPUT2, 100); //bez odstępu czasowego włączenie drugiej diody

delay(200); //czas świecenia drugiej diody
robot.write(OUTPUT2, 0); //wyłączenie drugiej diody
//nie ma polecenia delay, ponieważ chcemy aby zaraz po wyłączeniu drugiej diody
// program się powtarzał, czyli aby natychmiast włączała się pierwsza dioda itd.
}
```

Po napisaniu programu pamiętamy, aby zapisać szkic (**Plik > Zapisz jako > “zadanie_3”**), zweryfikować czy nie zawiera błędów, do wyjścia OUTPUT1 podłączyć jedną diodę, do OUTPUT2 – drugą, wgrać program na płytkę.

Dokonujemy obserwacji. Analizujemy. Jeśli trzeba dokonujemy poprawek programu i ponownie wgrywamy na płytkę.

Zadanie 4

Polecenie:

Zmodyfikuj program sterujący miganiem dwóch diod podłączonych do OUTPUT1 i OUTPUT2 wg własnego pomysłu. Po napisaniu programu zapisz szkic jako “zadanie_4”, zweryfikuj i wgraj na płytkę.

Przykładowe rozwiązanie:

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

void setup() {
}

void loop() {
robot.write(OUTPUT1, 100);

delay(200);
robot.write(OUTPUT1, 0);
delay(200);

robot.write(OUTPUT1, 100);

delay(200);
robot.write(OUTPUT1, 0);
delay(200);

robot.write(OUTPUT2, 100);

delay(200);
robot.write(OUTPUT2, 0);

delay(200);

robot.write(OUTPUT2, 100);

delay(200);
robot.write(OUTPUT2, 0);

delay(200);
}
```

Po wykonaniu zadania 4. pozwalamy uczniom na obserwacje, poprawianie skryptów, wprowadzanie własnych innowacji. Zachęcamy do wymiany doświadczeń, obserwacji programów i efektów działania w innych grupach. W zależności od pozostałego czasu pozwalamy uczniom eksperymentować z miganiem 2 diod dłużej lub krócej.

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 pytanie: Czego nauczyliśmy się na dzisiejszej lekcji?

- umiemy uruchomić Arduino IDE,
- znamy składnię kodu w środowisku Arduino IDE,
- wykorzystujemy funkcję **write** z biblioteki LOFI,
- piszemy proste programy sterujące miganiem diod,
- umiemy wgrywać je na płytkę Arduino UNO,
- podłączamy do płytki diody.

Na zakończenie opowiadamy uczniom, co będziemy robić i czego się nauczymy podczas kolejnej lekcji: poznamy inne funkcje sterownika LOFI, będziemy deklarować zmienne i nauczymy się definiować własne proste funkcje z parametrami.