Temat: Arduino - własne funkcje z argumentami

Załóżmy, że chcemy edytować naszą funkcję *zamigajLED* w taki sposób, aby podczas wywoływania możliwa była **zmiana prędkości migania.** W naszej deklaracji funkcji musimy dodać informację, że może ona przyjmować argument. Robimy to w nagłówku funkcji:

```
1 //Pierwotna wersja
2 void zamigajLED(){
3
4 //Nowa wersja
5 void zamigajLED(int czas){
```

Od tej pory kompilator wie, że w momencie napotkania wywołania funkcji w okrągłych nawiasach **ma spodziewać się liczby** (typ *int*). Liczba ta zostanie przekazana do funkcji i będzie tam występowała pod nazwą *czas.*

Będzie to zmienna widoczna tylko w wewnątrz naszej funkcji, czyli będzie to, tak zwana **zmienna lokalna.**

W praktyce wnętrze funkcji będzie wyglądało tak:

```
1 void zamigajLED(int czas){
2   digitalWrite(13, HIGH); //Włączenie diody
3   delay(czas); //Odczekanie przez jakiś czas
4   digitalWrite(13, LOW); //Wyłączenie diody
5   delay(czas); //Odczekanie przez jakiś czas
6 }
```

Pora na ostatni element, czyli nowe wywołanie funkcji. Zwyczajnie w nawiasie wpisujemy liczbę, która będzie oznaczała czas przez jaki dioda ma być włączona i wyłączona. Cały kod prezentuje się tak:

```
 1  void setup() {
 2    pinMode(13, OUTPUT); //Konfiguracja pinu 13 jako wyjście
 3  }
 4
 5  void loop() {
 6    zamigajLED(50);
 7  }
 8
 9  void zamigajLED(int czas){
10    digitalWrite(13, HIGH); //Włączenie diody
11    delay(czas); //Odczekanie jakiegoś czasu
12    digitalWrite(13, LOW); //Wyłączenie diody
13    delay(czas); //Odczekanie jakiegoś czasu
14 }
```

Sprawdź, że podmieniając wartość w nawiasach **możesz wpływać na częstotliwość migania.** Pora na kolejny przykład.

Funkcje z argumentem - przykład 2

Do tej pory ciągłe miganie diody było widoczne, ponieważ wywoływaliśmy naszą funkcję w funkcji **loop**, która jest pętlą. Gdy przeniesiemy wywołanie funkcji do sekcji **setup**, to po starcie systemu zobaczymy tylko jedno mignięcie.

Pora na przerobienie funkcji w taki sposób, aby oprócz zmiany czasu migania możliwe było również **wpływanie na ilość mignięć.** Na początku musimy zmienić deklarację funkcji:

```
1 //Wersja pierwotna
2 void zamigajLED(int czas){
3
4 //Wersja poprawiona
5 void zamigajLED(int czas, int ile){
```

Dodaliśmy argument **ile**, to on będzie odpowiadał za ilość mignięć. Jak widać, wpisany został po przecinku, poprzedzony oczywiście swoim typem (liczba całkowita, int). Parametr ten mógłby być innego typu, nie ma to żadnej różnicy. Ważne, aby deklaracja była odpowiednia.

Funkcja może przyjmować wiele argumentów o różnych typach!

Jak pewnie się domyślasz wykorzystanie drugiego argumentu jest bardzo proste. Zwyczajnie mamy do dyspozycji teraz drugą zmienną lokalną **ile.**

Mam nadzieję, że wiesz już, którą pętlę użyć, aby najłatwiej zapanować nad ilością mignięć? Jeśli nie, to odsyłam do poprzedniej części, w której omówiłem pętle for. Zakładam jednak, że materiał ten masz opanowany (lub zaraz nadrobisz braki) i przedstawiam nową funkcję **zamigajLED:**

```
1 void zamigajLED(int ile, int czas){
2   for (int i=0; i < ile; i++) {
3     digitalWrite(13, HIGH); //Włączenie diody
4     delay(czas); //Odczekanie jakiegoś czasu
5     digitalWrite(13, LOW); //Wyłączenie diody
6     delay(czas); //Odczekanie jakiegoś czasu
7   }
8 }
```

Cały kod:

```
 1  void setup() {
 2    pinMode(13, OUTPUT); //Konfiguracja pinu 8 jako wyjście
 3    zamigajLED(100, 5);
 4  }
 5
 6  void loop() {
 7  }
 8
 9  void zamigajLED(int czas, int ile){
10    for (int i=0; i < ile; i++) {
11      digitalWrite(13, HIGH); //Włączenie diody
12      delay(czas); //Odczekanie jakiegoś czasu
```

```
13    digitalWrite(13, LOW); //Wyłączenie diody
14    delay(czas); //Odczekanie jakiegoś czasu
15   }
16 }
```

Wszystko powinno działać. Teraz po starcie dioda musi zamigać dokładnie 5 razy. Jednak skąd Arduino wie jak rozróżnić informacje podane w nawiasach, co jest czasem, a co ilością powtórzeń?

Otóż zasada jest banalnie prosta - **kolejne wartości oddzielone przecinkami są przypisywane do kolejnych argumentów opisanych w deklaracji funkcji.** W powyższym przypadku pierwsza liczba będzie zawsze traktowana jako czas, a druga jako ilość powtórzeń.

Zadanie domowe

Napisz funkcję, która pozwala osobno regulować ilość mignięć, czas przez jaki dioda jest wyłączona oraz osobno, czas przez jaki dioda jest włączona.