Blinkende LEDs mit Multithreading

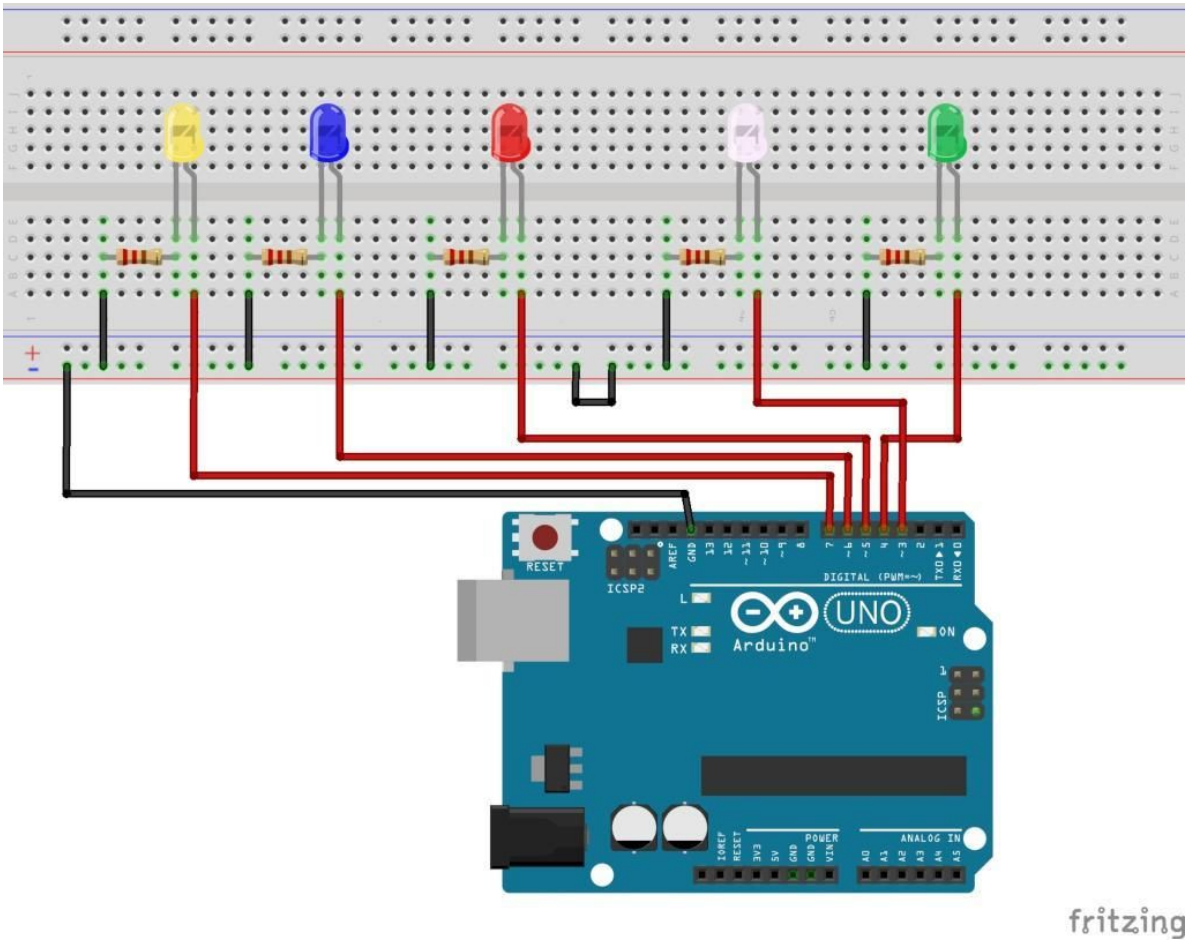
Als Multithreading wird die Möglichkeit bezeichnet, mehrere Threads (Programmteile) quasi gleichzeitig ablaufen zu lassen.

Im Programm sollen mehrere LEDs gleichzeitig und in unterschiedlichen Rhythmen blinken. Das kann natürlich mit delay nicht realisiert werden, weil delay den Programmablauf für die im Parameter bestimmte Zeit anhält.

Benötigte Bauteile:

- 5 LED
- Widerstände > 100 Ω

Baue die Schaltung auf.



Für jede der LEDs ist ein eigener Programmteil (Thread) verantwortlich, die Threads werden im loop-Teil nacheinander aufgerufen.

Mit enum werden die LEDs und ihre Ports definiert, mit Minimum und Maximum wird der Bereich des Blinkintervalls festgelegt:

```
enum LED
{
  // Startwert Pin 3
  LED_1 = 3,
  LED_2,
  LED_3,
  LED_4,
  LED_5
};

// Zufallsbereich des Blinkintervalls
# define Minimum 200
# define Maximum 2000
```

Der setup-Teil definiert die pinModes der LEDs und startet den Zufallsgenerator.

```
void setup()
{
  // pinMode festlegen: Startwert -> LED_1, Endwert  LED_5
  for (int i = LED_1; i <= LED_5; i++)
  {
    pinMode(i, OUTPUT);
  }

  // Zufallsgenerator starten
  randomSeed(analogRead(0));
}
```

Für jede LED ist ein eigener Programmteil zuständig:

```
void LED_1_Blinken()
{
  /*
    die Variablen sollen den letzten Wert des Aufrufs behalten
    -> Definition als static
  */
  static bool Zustand = false;

  /*
    der Rückgabewert von millis() ist unsigned long int
    -> WarteZeit muss ebenfalls den Typ unsigned long int haben
  */
  static unsigned long int WarteZeit = 0;
  // Blinkintervall zufällig ermitteln
  int BlinkIntervall =  random(Minimum, Maximum);

  // wenn Zustand false
```

```
  if (!Zustand)
  {
    /*
     millis() ermittelt die seit Programmstart verstrichene Zeit
     prüfen, ob die Zeit schon abgelaufen ist
     millis() - WarteZeit muss größer als das BlinkIntervall sein
    */
    if (millis() - WarteZeit >= BlinkIntervall)
    {
      digitalWrite(LED_1, HIGH);

      // Wartezeit um Blinkintervall erhöhen
      WarteZeit += BlinkIntervall;
      Zustand = !Zustand;
    }
  }

  // wenn Zustand true
  if (Zustand)
  {
    if (millis() - WarteZeit >= BlinkIntervall)
    {
      digitalWrite(LED_1, LOW);
      WarteZeit += BlinkIntervall;

      // Zustand "umdrehen" true -> false, false -> true
      Zustand = !Zustand;
    }
  }
}
```

Der Programmteil wiederholt sich für die anderen LEDs, es muss nur der Name der LED (LED_2, LED_3 …) ausgetauscht werden.

Im loop-Teil werden nacheinander die Methoden für die verschiedenen LEDs aufgerufen:

```
void loop()
{
  LED_1_Blinken();
  LED_2_Blinken();
  LED_3_Blinken();
  LED_4_Blinken();
  LED_5_Blinken();
}
```