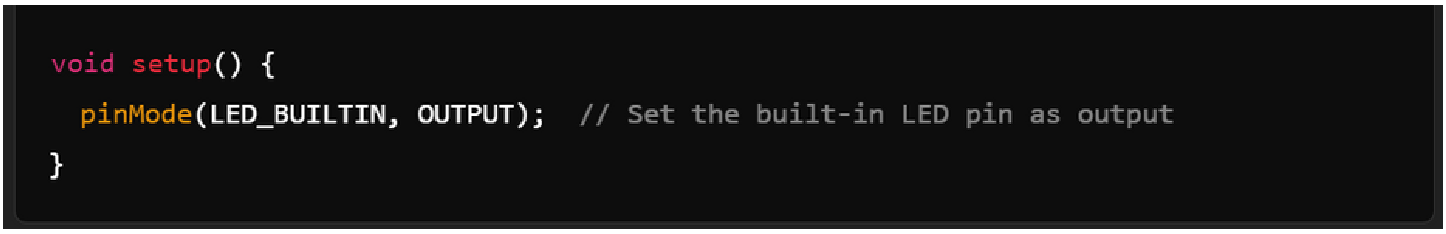
Introduction to Arduino Programming

Arduino programming revolves around writing code (sketches) in a simplified version of C++ using the Arduino IDE (Integrated Development Environment). Arduino programs are structured to interact with physical components such as sensors, motors, and LEDs. By writing code, you control these components to perform specific tasks.

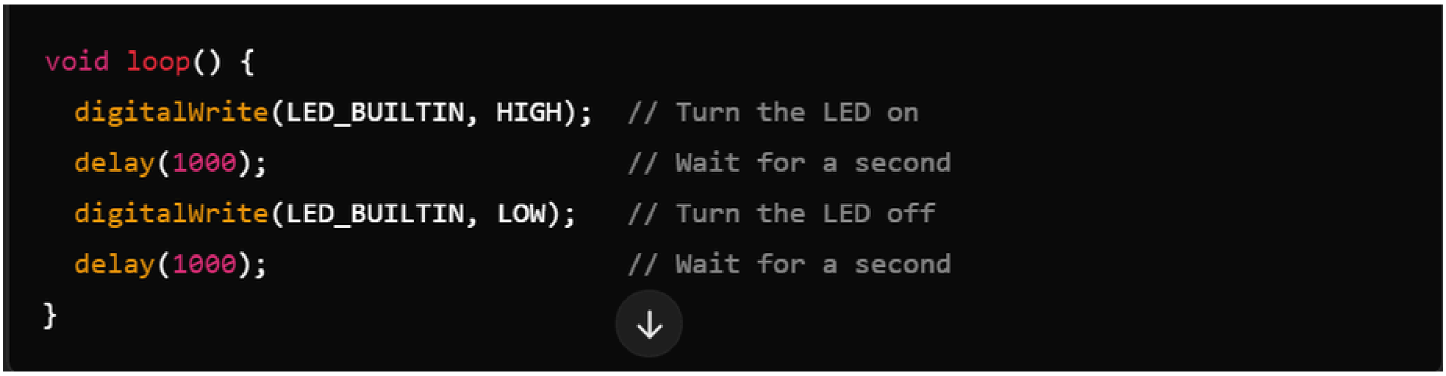
Key Components of an Arduino Program:

Setup function (void setup()): This function runs once when the Arduino is powered on or reset. You use it to initialize variables, pin modes (whether a pin is an input or output), libraries, etc.



```
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);  // Set the built-in LED pin as output
}
```

Loop function (void loop()): This function continuously executes after setup(). It contains the main logic of the program that you want to repeat.



```
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);  // Turn the LED on
  delay(1000);                      // Wait for a second
  digitalWrite(LED_BUILTIN, LOW);   // Turn the LED off
  delay(1000);                      // Wait for a second
}
```

Pin Modes: Arduino pins can be configured as inputs or outputs:

- pinMode(pin, INPUT); for reading data (e.g., from a sensor)
- pinMode(pin, OUTPUT); for sending signals (e.g., turning on an LED)

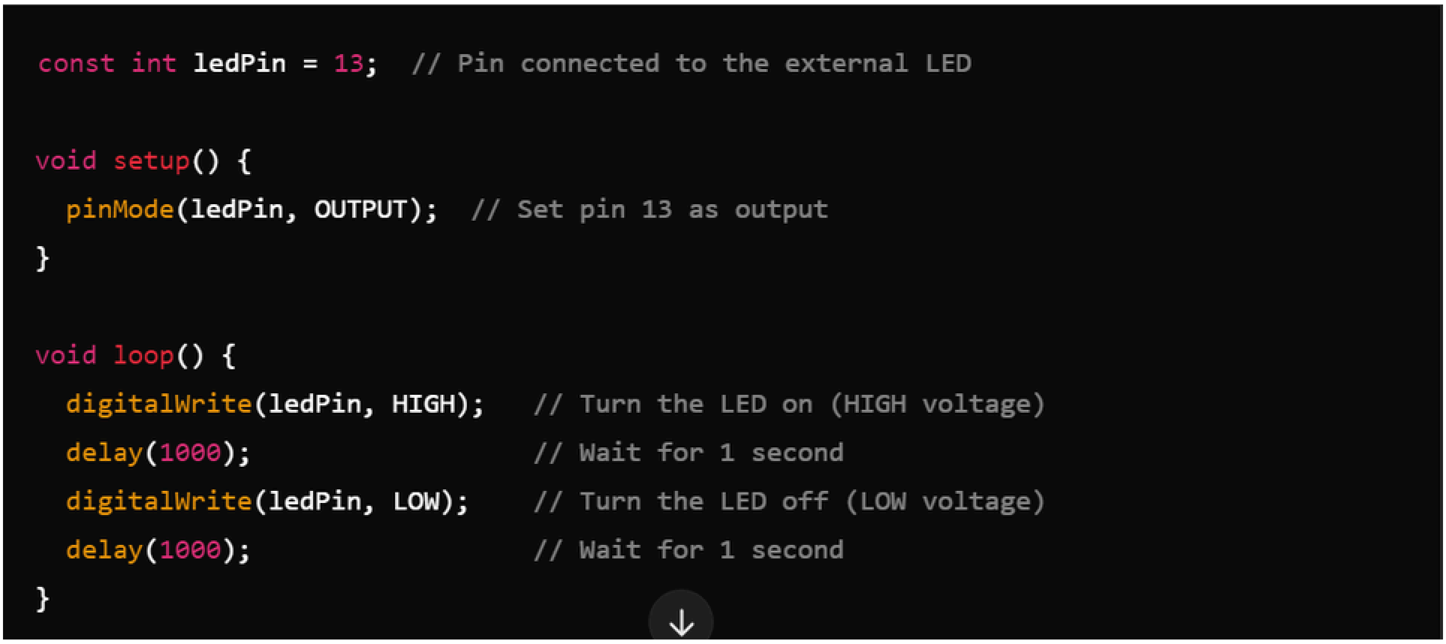
Functions:

- digitalWrite(pin, value); — Sets a pin to HIGH or LOW.
- digitalRead(pin); — Reads the value from a digital pin (HIGH or LOW).
- analogWrite(pin, value); — Writes an analog value (PWM) to a pin.
- analogRead(pin); — Reads the analog value from a pin (0 to 1023).

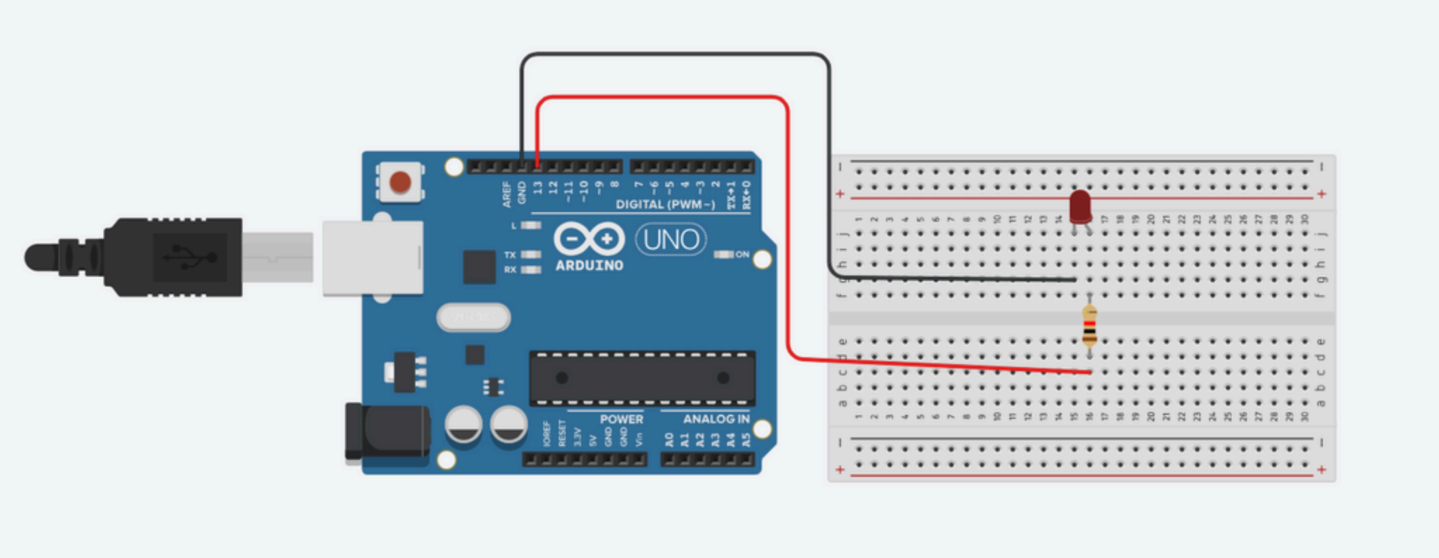
Basic Arduino Programs

1.Blinking an LED

This is the classic "Hello World" of Arduino programming. It turns an LED on and off with a 1-second delay.

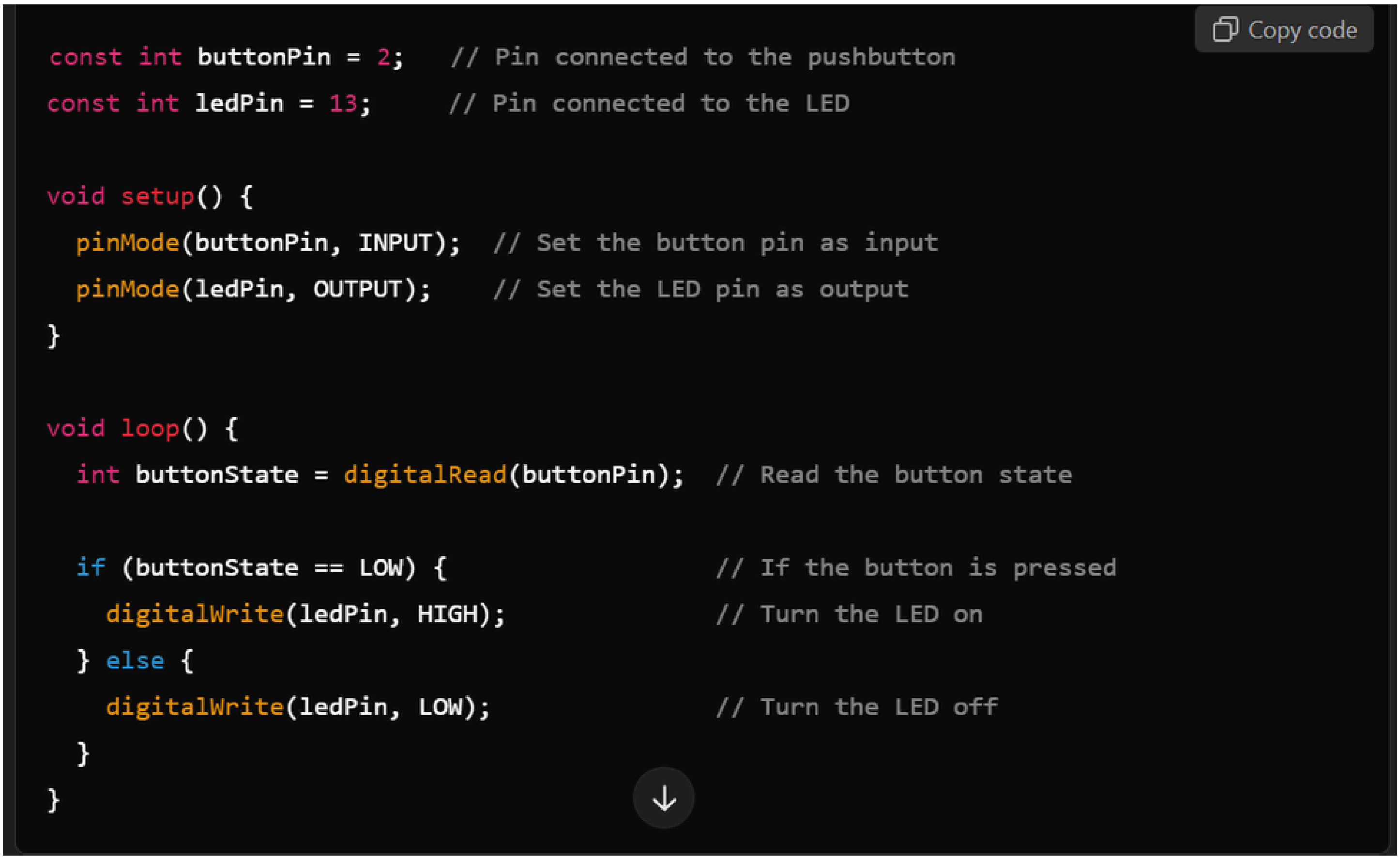


```
const int ledPin = 13;  // Pin connected to the external LED

void setup() {
  pinMode(ledPin, OUTPUT);  // Set pin 13 as output
}

void loop() {
  digitalWrite(ledPin, HIGH);   // Turn the LED on (HIGH voltage)
  delay(1000);                  // Wait for 1 second
  digitalWrite(ledPin, LOW);    // Turn the LED off (LOW voltage)
  delay(1000);                  // Wait for 1 second
}
```

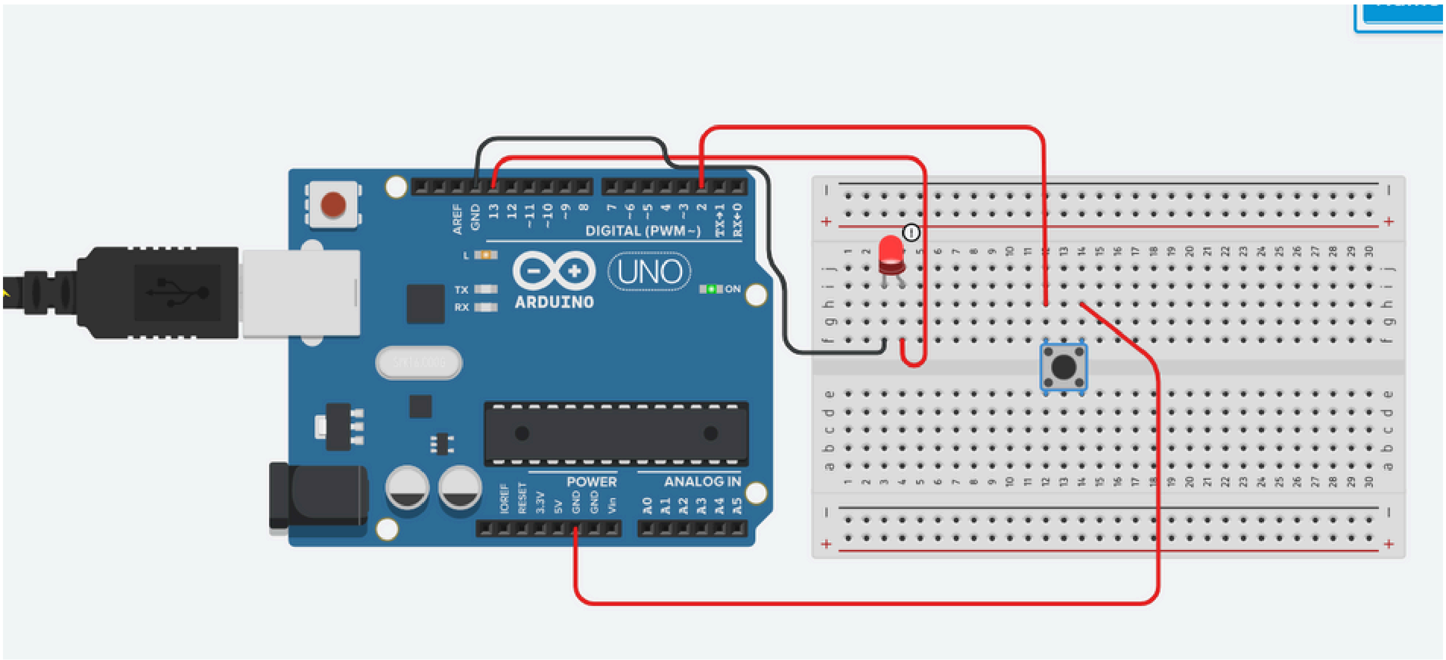


2.LED control via Pressing Push Button

This program reads the state of a pushbutton and turns an LED on or off based on whether the button is pressed.

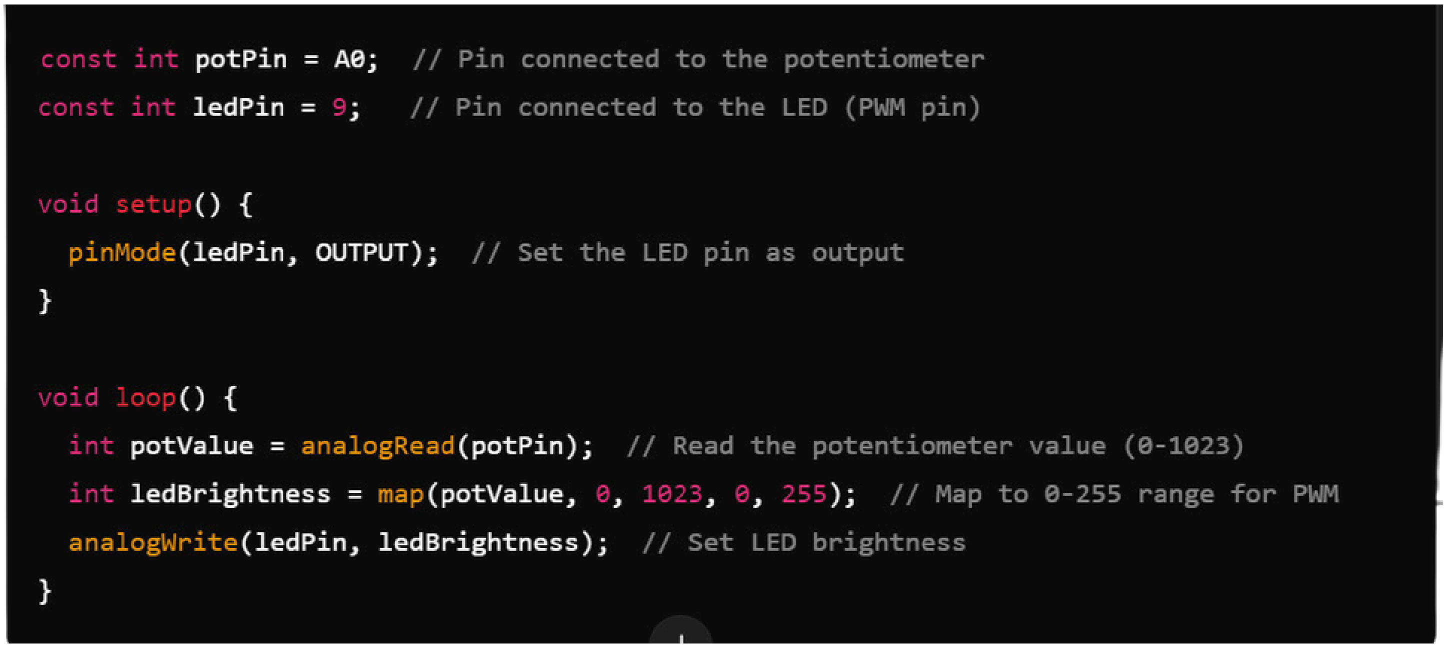


```
const int buttonPin = 2;   // Pin connected to the pushbutton
const int ledPin = 13;     // Pin connected to the LED

void setup() {
  pinMode(buttonPin, INPUT);  // Set the button pin as input
  pinMode(ledPin, OUTPUT);    // Set the LED pin as output
}

void loop() {
  int buttonState = digitalRead(buttonPin);  // Read the button state

  if (buttonState == LOW) {                  // If the button is pressed
    digitalWrite(ledPin, HIGH);              // Turn the LED on
  } else {
    digitalWrite(ledPin, LOW);               // Turn the LED off
  }
}
```

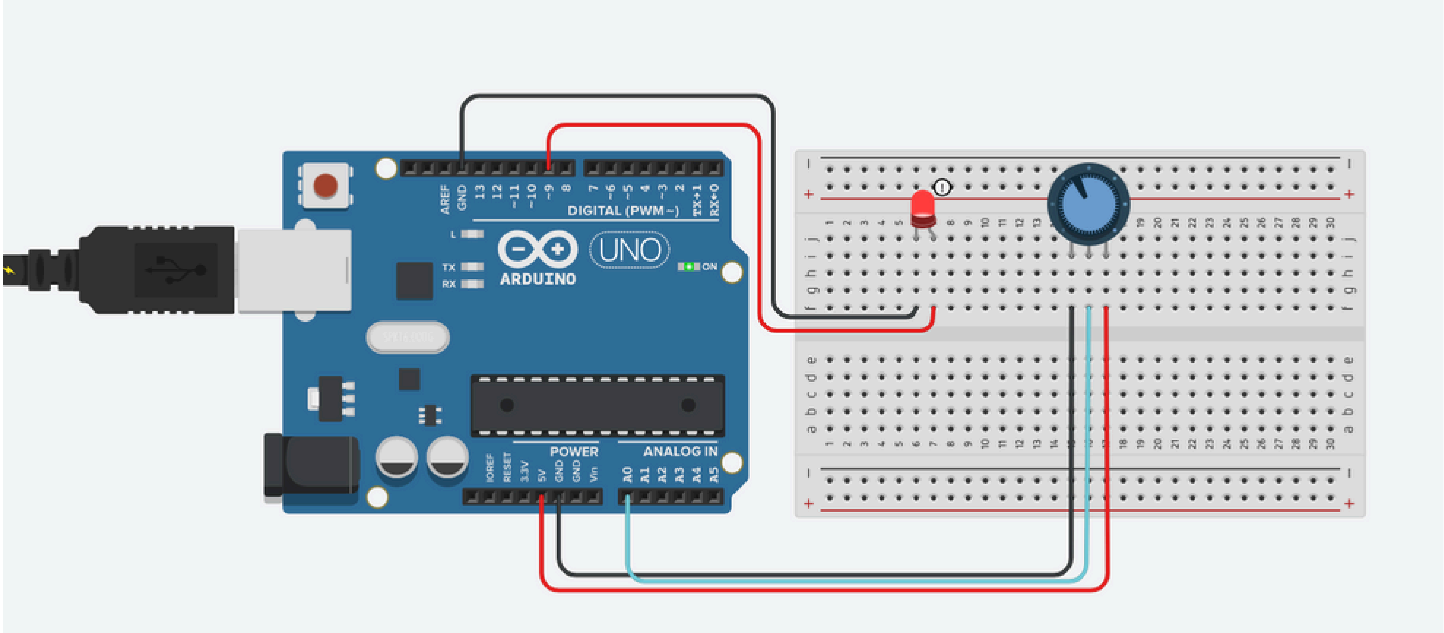


3.Reading Analog Input from a Potentiometer

This program reads the value from a potentiometer (analog input) and changes the brightness of an LED (analog output using PWM).

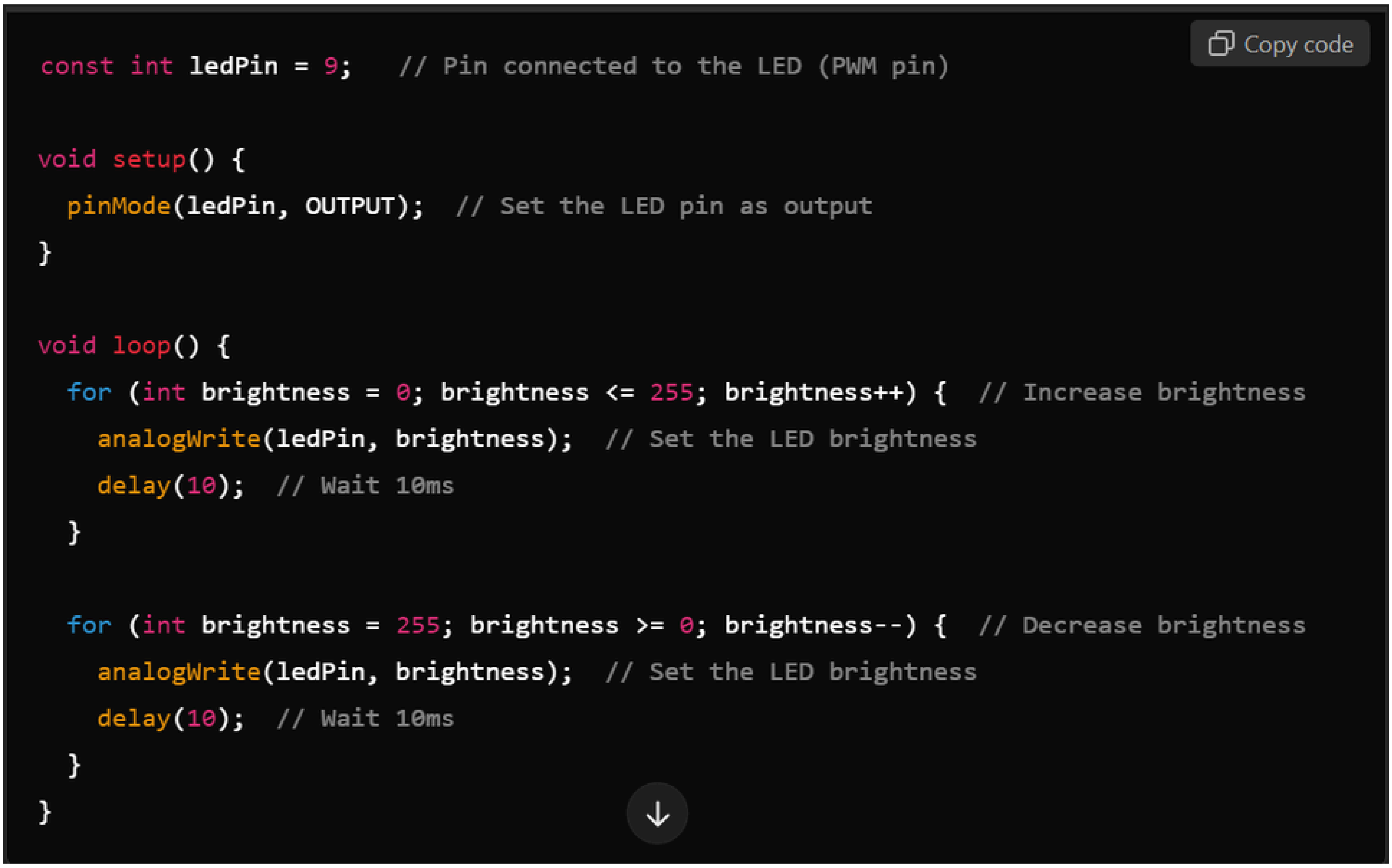


```
const int potPin = A0;  // Pin connected to the potentiometer
const int ledPin = 9;   // Pin connected to the LED (PWM pin)

void setup() {
  pinMode(ledPin, OUTPUT);  // Set the LED pin as output
}

void loop() {
  int potValue = analogRead(potPin);  // Read the potentiometer value (0-1023)
  int ledBrightness = map(potValue, 0, 1023, 0, 255);  // Map to 0-255 range for PWM
  analogWrite(ledPin, ledBrightness);  // Set LED brightness
}
```

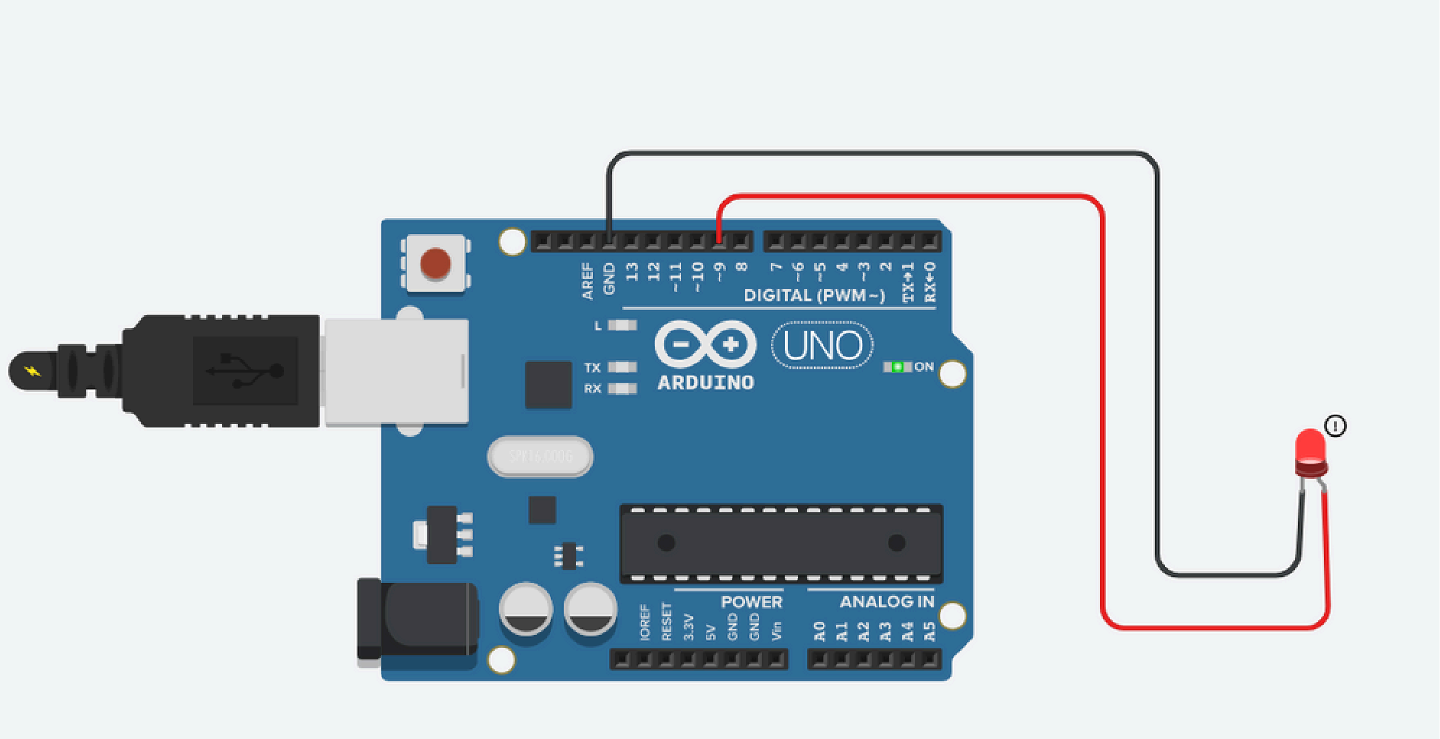


4.Fading an LED

This program gradually increases and decreases the brightness of an LED using PWM.

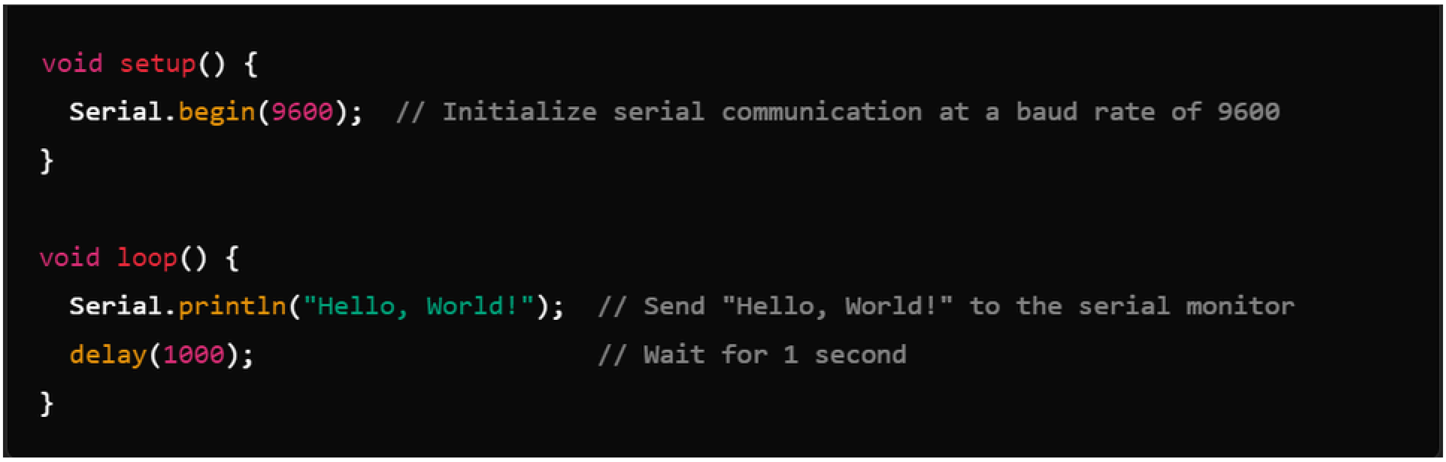


```
const int ledPin = 9;   // Pin connected to the LED (PWM pin)

void setup() {
  pinMode(ledPin, OUTPUT);  // Set the LED pin as output
}

void loop() {
  for (int brightness = 0; brightness <= 255; brightness++) {  // Increase brightness
    analogWrite(ledPin, brightness);  // Set the LED brightness
    delay(10);  // Wait 10ms
  }

  for (int brightness = 255; brightness >= 0; brightness--) {  // Decrease brightness
    analogWrite(ledPin, brightness);  // Set the LED brightness
    delay(10);  // Wait 10ms
  }
}
```

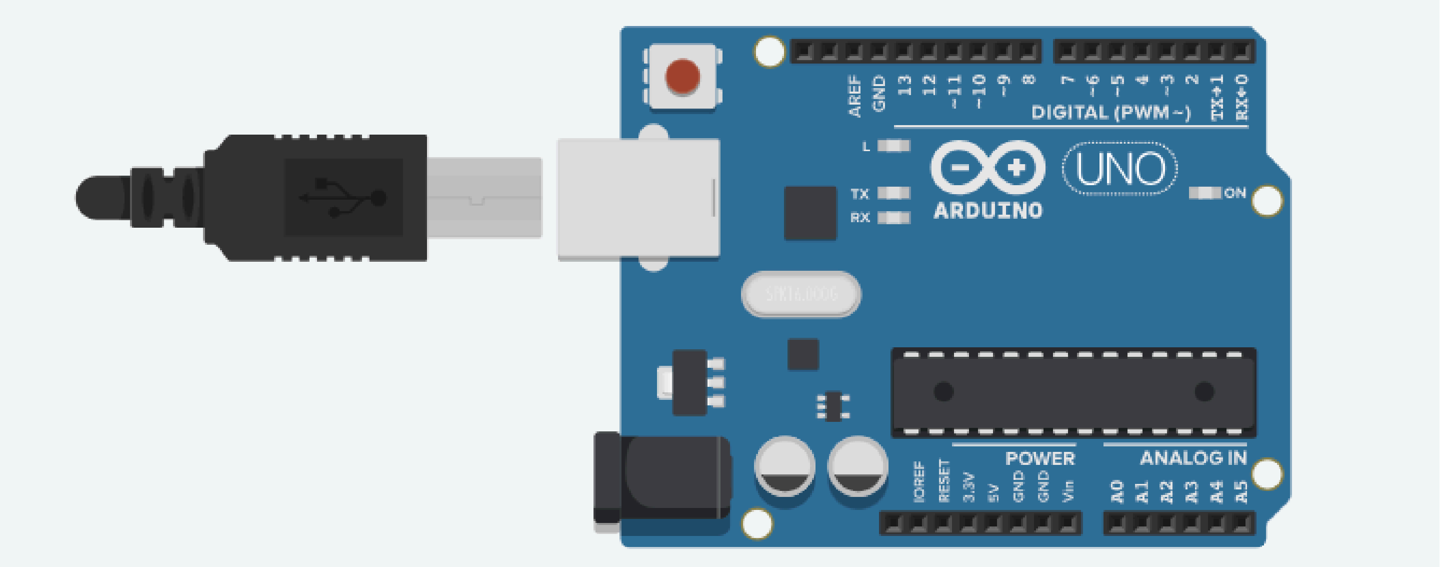


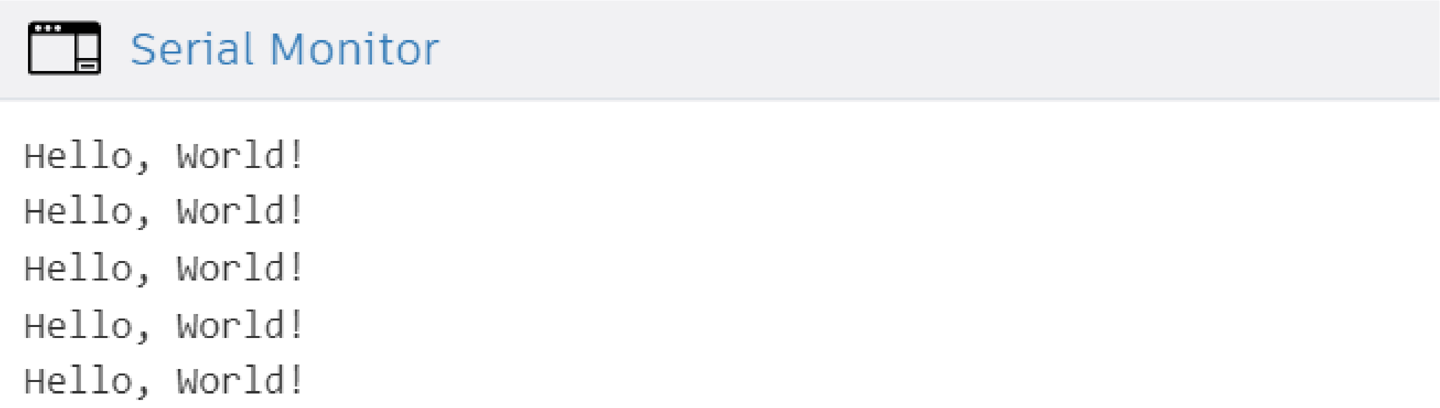
5.Serial Communication (Sending Data to the Computer)

This program demonstrates how to send data from the Arduino to the computer using the serial monitor.



```
void setup() {
  Serial.begin(9600);  // Initialize serial communication at a baud rate of 9600
}

void loop() {
  Serial.println("Hello, World!");  // Send "Hello, World!" to the serial monitor
  delay(1000);                      // Wait for 1 second
}
```





```
Hello, World!
Hello, World!
Hello, World!
Hello, World!
Hello, World!
```