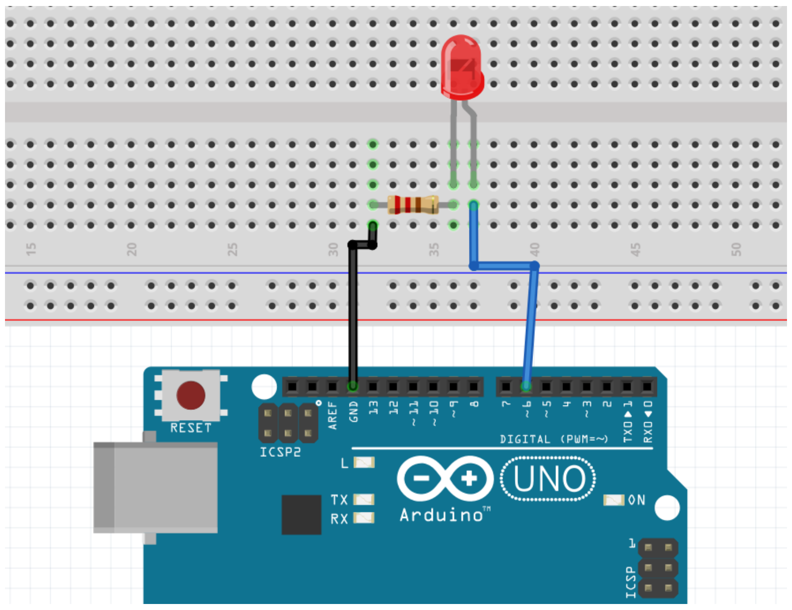
Chapter 13
Controlling Brightness of a LED

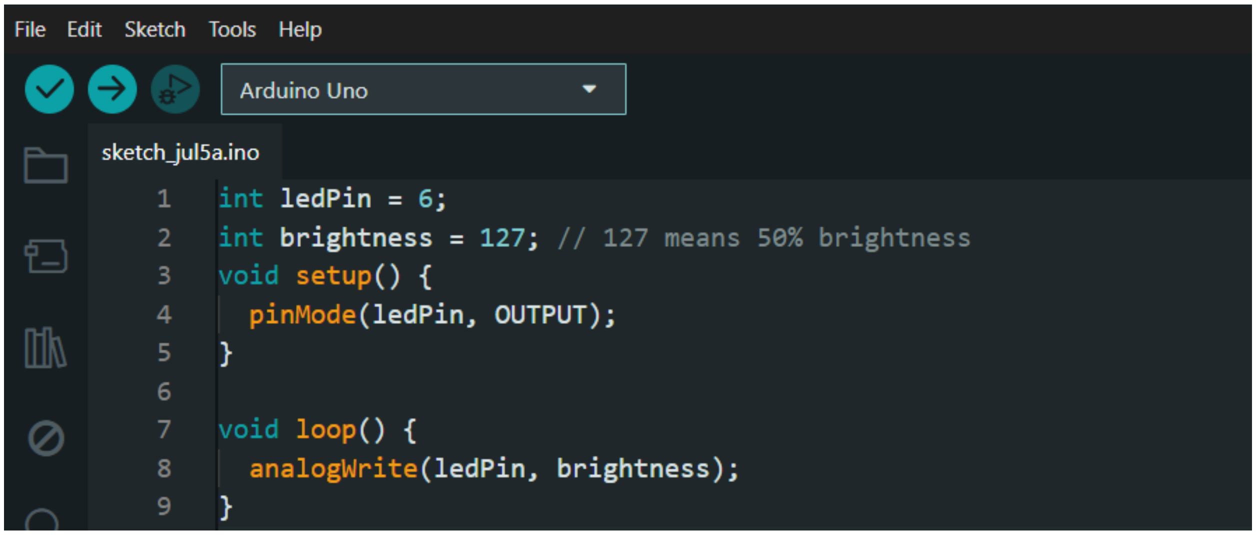
So, far we have learned to turn ON and OFF a LED. The function used to do so is digitalWrite(). This word digital means that this type of control has only two states: ON and OFF. Just like any digital systems. Note: HIGH/LOW or 1/0 or ON/OFF are actually same things. ON means 100% brightness and OFF means 0% brightness.

But it is also possible to operate the brightness at 50% brightness or 60% or 70% or at any brightness between 0% to 100%. This method is known as PWM control. Just remember that this term PWM stands for Pulse Width Modulation. It has details theory that we will learn one day but not today. Only a few pins of Arduino UNO have this PWM capability. Those capable pins are marked with '~' sign. Look carefully at pins 3,5,6,9,10 and 11.

Build this circuit below:



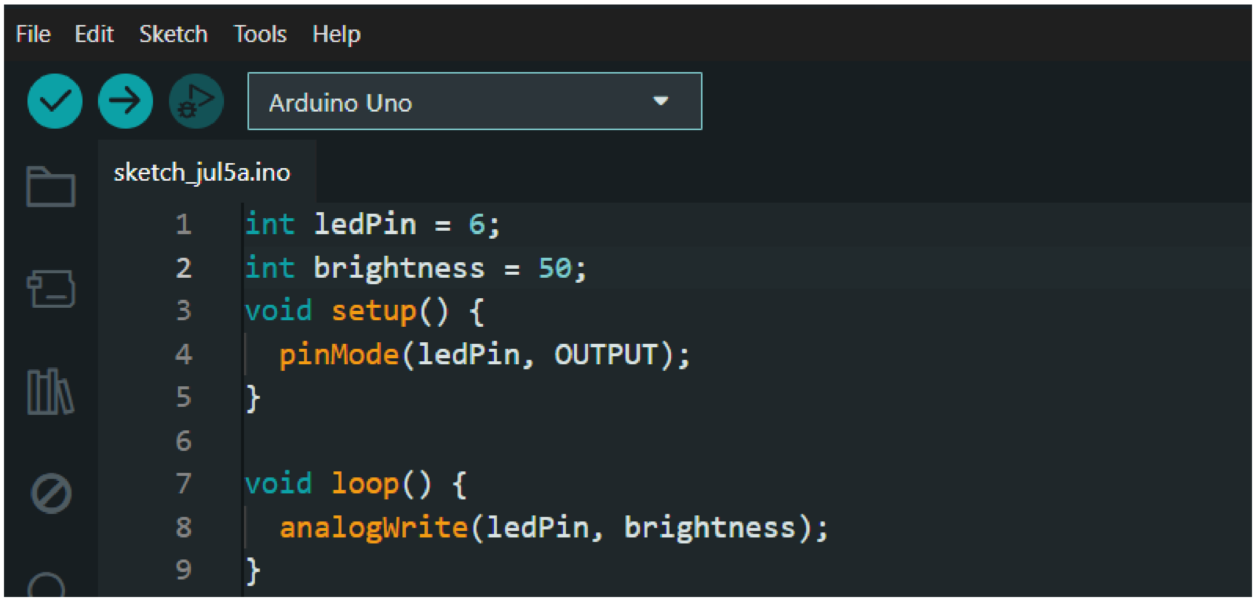
Now upload this code below:



```
int ledPin = 6;
int brightness = 127; // 127 means 50% brightness
void setup() {
  pinMode(ledPin, OUTPUT);
}

void loop() {
  analogWrite(ledPin, brightness);
}
```

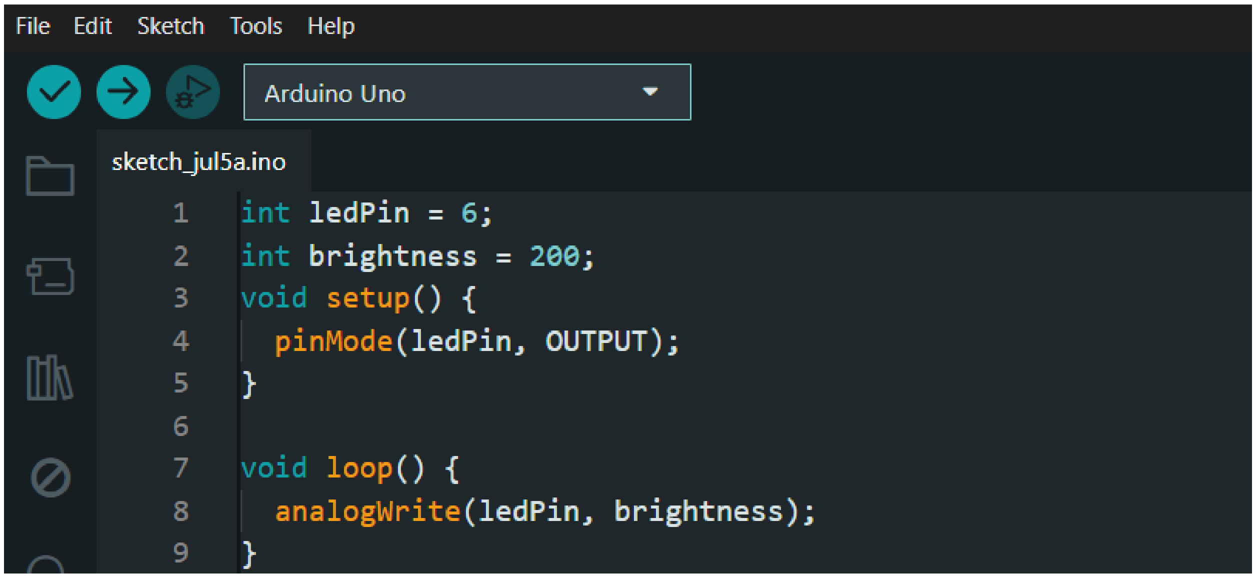
Now upload this code below and see the change:



```
int ledPin = 6;
int brightness = 50;
void setup() {
  pinMode(ledPin, OUTPUT);
}

void loop() {
  analogWrite(ledPin, brightness);
}
```

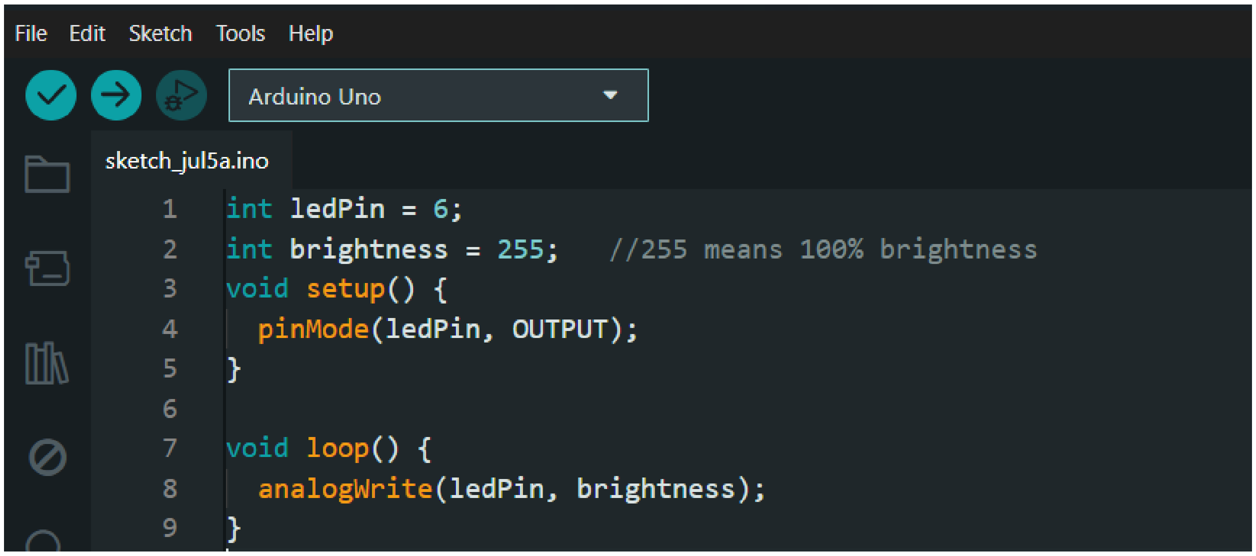
Now upload this code below and see the change:



```
int ledPin = 6;
int brightness = 200;
void setup() {
  pinMode(ledPin, OUTPUT);
}

void loop() {
  analogWrite(ledPin, brightness);
}
```

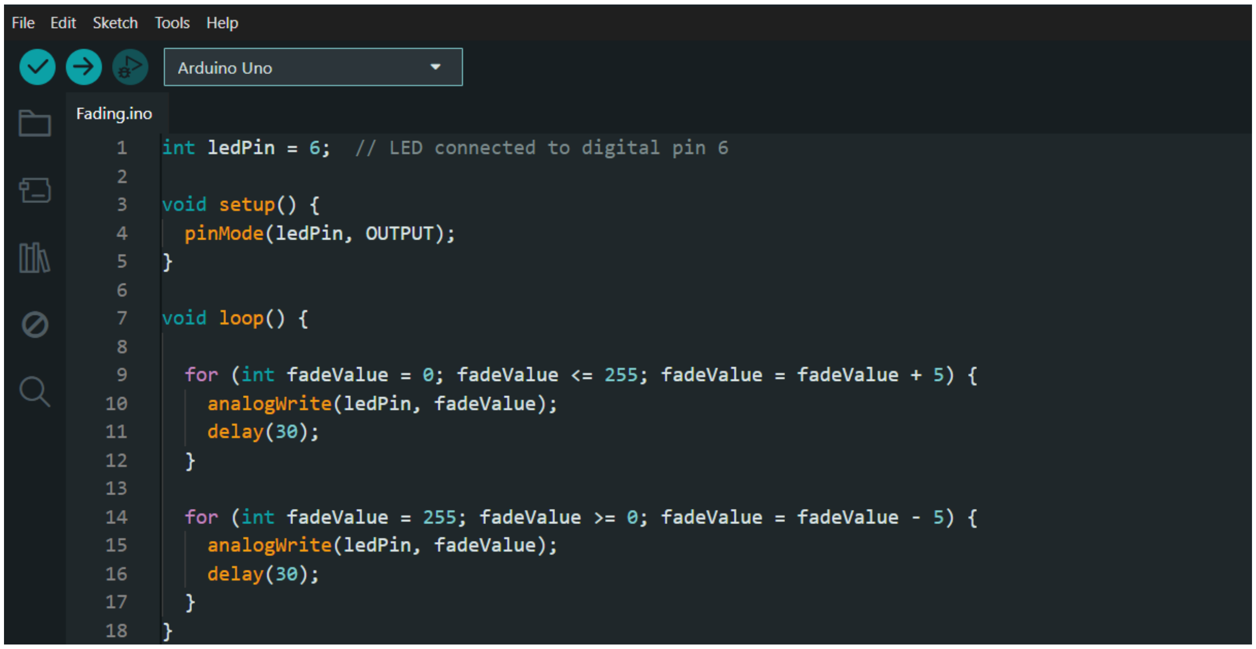
Now upload this code:



```
int ledPin = 6;
int brightness = 255;   //255 means 100% brightness
void setup() {
  pinMode(ledPin, OUTPUT);
}

void loop() {
  analogWrite(ledPin, brightness);
}
```

Hope it makes some sense. If you use brightness = 0; the LED will be turned OFF. Now upload this code below:



```
int ledPin = 6;  // LED connected to digital pin 6

void setup() {
  pinMode(ledPin, OUTPUT);
}

void loop() {

  for (int fadeValue = 0; fadeValue <= 255; fadeValue = fadeValue + 5) {
    analogWrite(ledPin, fadeValue);
    delay(30);
  }

  for (int fadeValue = 255; fadeValue >= 0; fadeValue = fadeValue - 5) {
    analogWrite(ledPin, fadeValue);
    delay(30);
  }
}
```

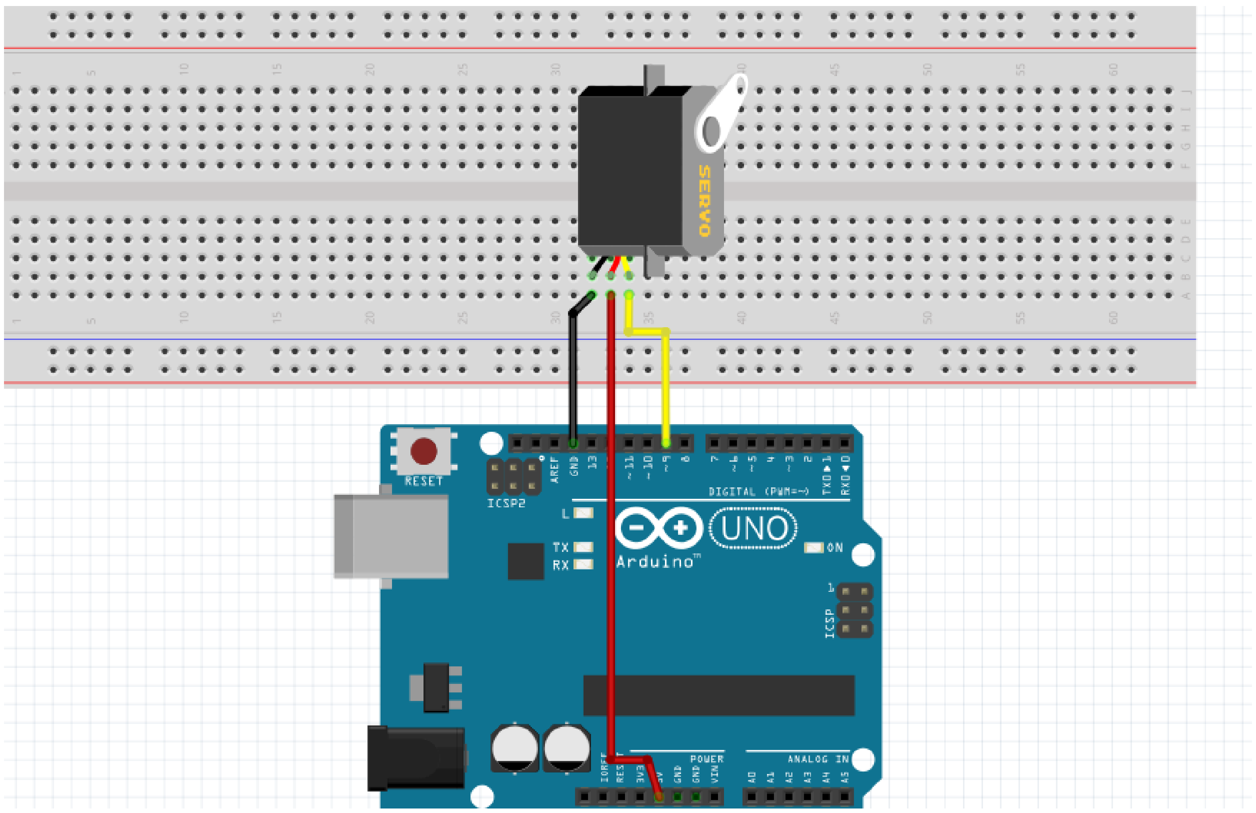
What do you see? It looks better than LED blinking. This is called LED fading. In fading instead of turning ON or OFF we go to brightness 100% or 0% slowly. Try to understand this code specially the use of for loops.

Just to mention, using this same brightness control technique we can control the speed of a motor. But the problem is Arduino can not power up a motor directly. You need an additional device named Motor driver. What this motor driver does is it works as a bridge between Arduino and Motor. The Arduino gives signal to motor driver and the motor driver gives power to the motor from the battery according to the signal given by Arduino. There are many motor drivers in the BD market. L298N, TB66, BTS79 are popular amongst all.

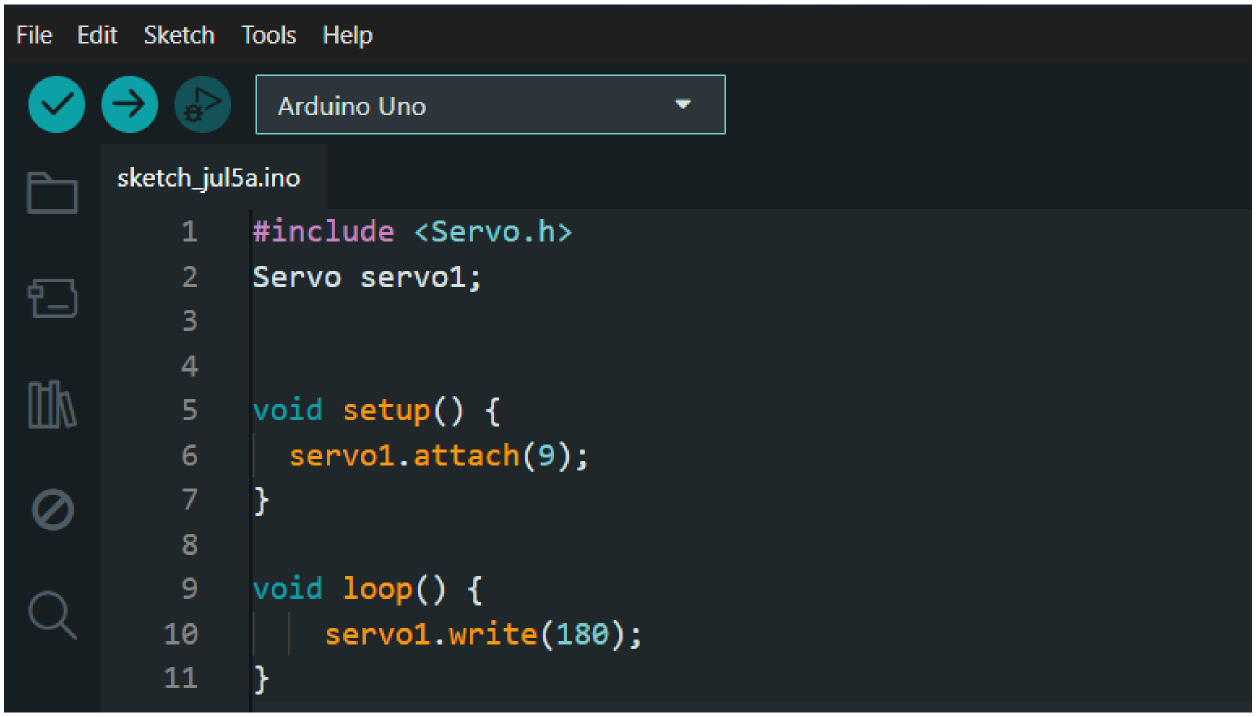
Chapter 14
Controlling a Servo Motor

So far whatever we have learned none of them can move. If a robot cannot move it will never be interesting. Motors are the way to make your robot do some movements. There are different motors in the market. Each of them has different types of work. Such as the DC motor as seen the toys. The servo motor is a special type of motor which can be controlled very precisely. It can be controlled degree by degree.

Build this circuit below: (The yellow pin might be a PWM pin.)



Now upload this code below:

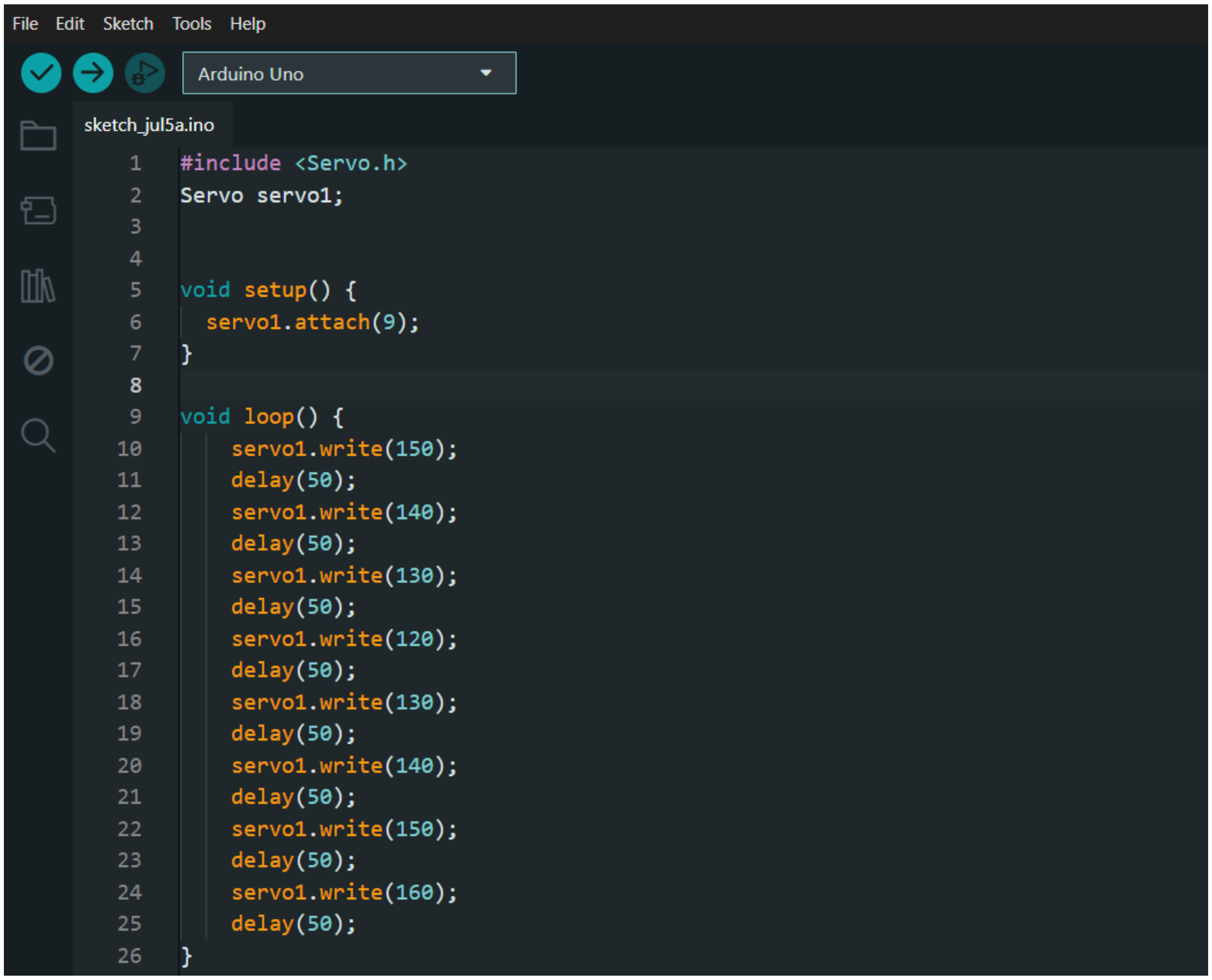


```
#include <Servo.h>
Servo servo1;


void setup() {
  servo1.attach(9);
}

void loop() {
    servo1.write(180);
}
```

Now upload this code below:

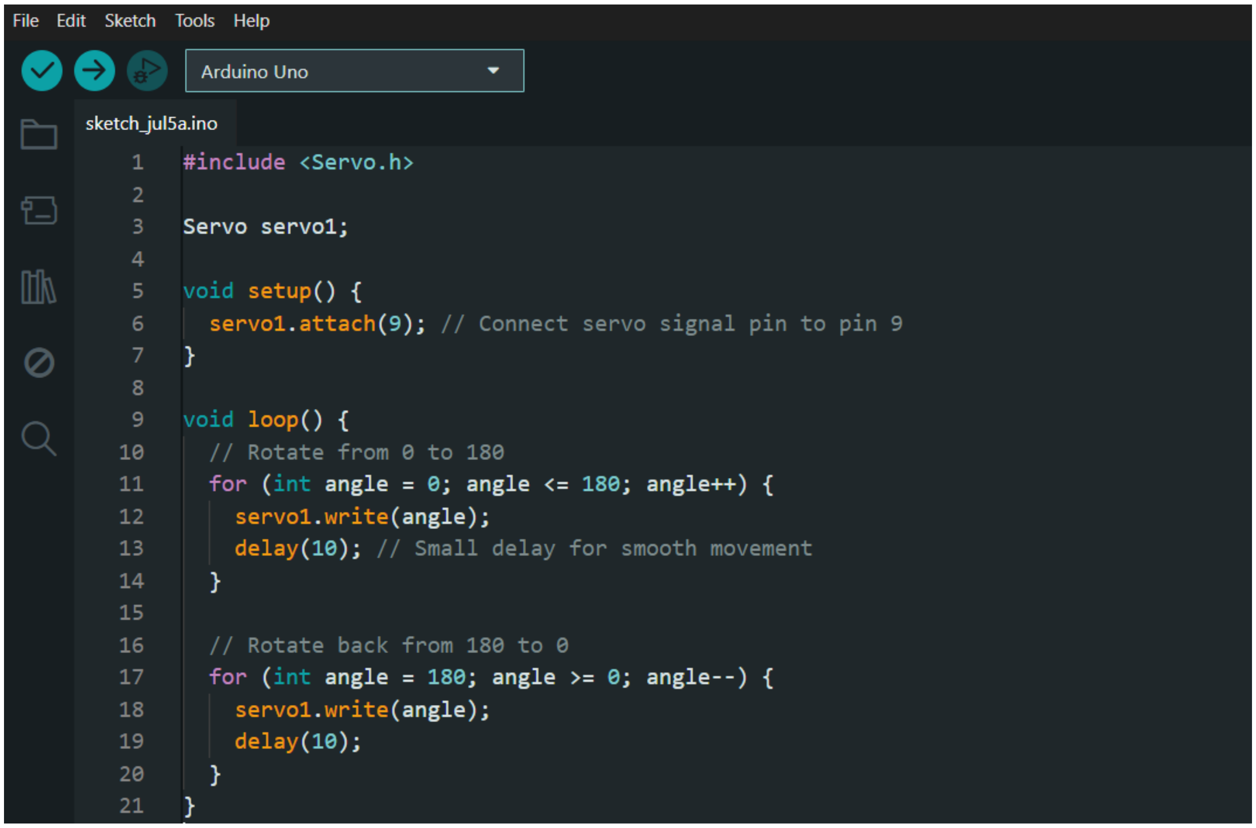


```
#include <Servo.h>
Servo servo1;


void setup() {
  servo1.attach(9);
}

void loop() {
    servo1.write(150);
    delay(50);
    servo1.write(140);
    delay(50);
    servo1.write(130);
    delay(50);
    servo1.write(120);
    delay(50);
    servo1.write(130);
    delay(50);
    servo1.write(140);
    delay(50);
    servo1.write(150);
    delay(50);
    servo1.write(160);
    delay(50);
}
```

Now upload this code and try to understand:



```
#include <Servo.h>

Servo servo1;

void setup() {
  servo1.attach(9); // Connect servo signal pin to pin 9
}

void loop() {
  // Rotate from 0 to 180
  for (int angle = 0; angle <= 180; angle++) {
    servo1.write(angle);
    delay(10); // Small delay for smooth movement
  }

  // Rotate back from 180 to 0
  for (int angle = 180; angle >= 0; angle--) {
    servo1.write(angle);
    delay(10);
  }
}
```

Spend some time experimenting with different servo movements.

Assignment 06

1. Control the brightness of a LED using the potentiometer. Note that the potentiometer gives reading in the range of 0-1023 but brightness value can be only in the range of 0-255.
2. Control the position of a servo using the potentiometer. Note that the potentiometer gives reading in the range of 0-1023 but servo position can be only in the range of 0-180 degree.
3. Control the intensity of buzzer's sound using this same brightness control technique. (If it is not working try adding a 220ohm resistor in series with the buzzer).
4. Make a project with IR obstacle and servo. You have to imitate an incident like this if anything comes in front of IR sensor the servo will open a door. Else the door will stay closed.
5. Hard one: Connect two pushbuttons and a servo to Arduino. If you press one button the servo will rotate clockwise 10 degrees for each press (but not more than 180 degree). And if you press the other button the servo will rotate counter clockwise 10 degree for each press (but not less than 0 degree).
6. What is the difference between analogWrite() and digitalWrite()?
7. What is the difference between analgRead() and digitalRead() ?