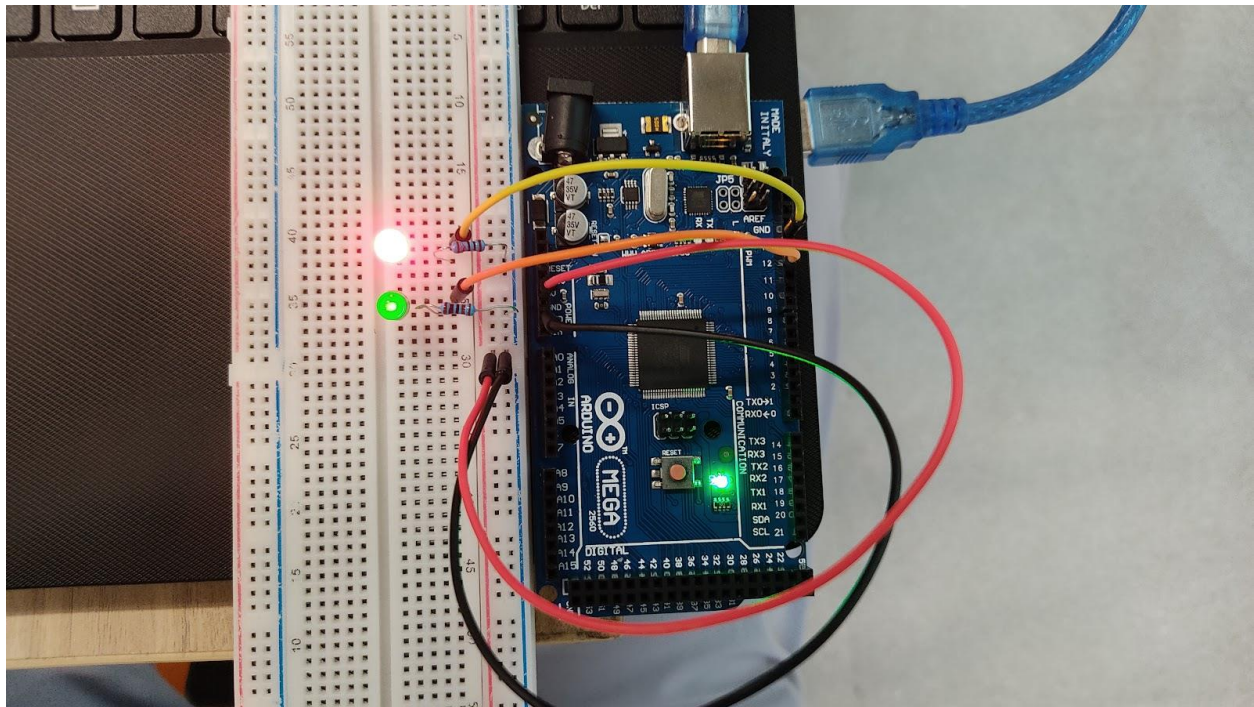
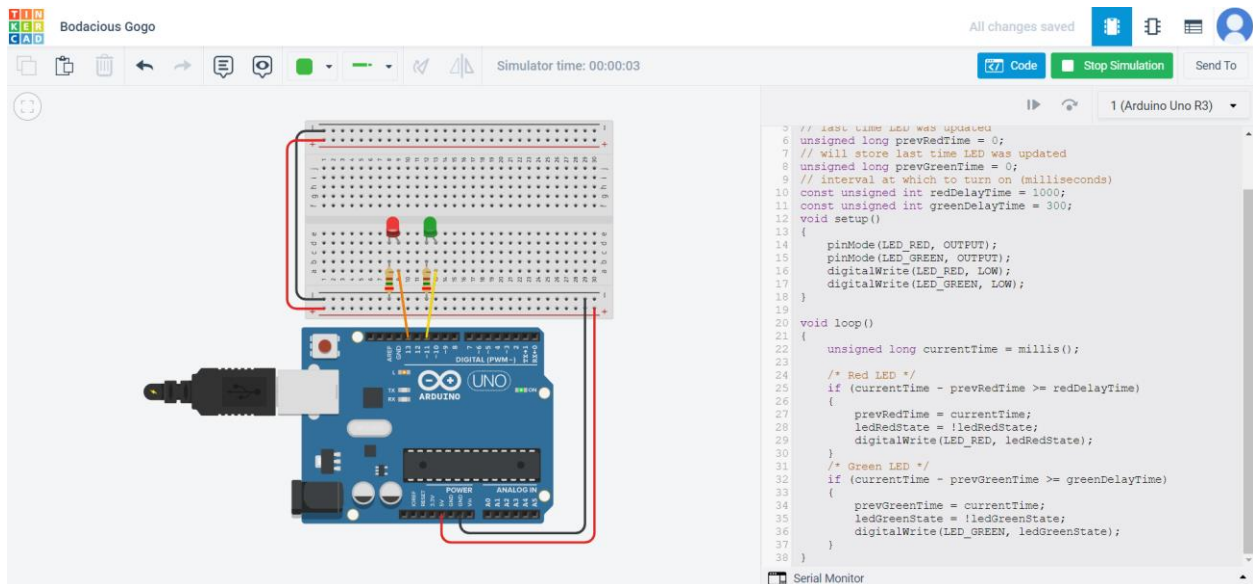


## Group Work 1 &amp; 2; Real Circuit Picture



## Group Work 1

## Tinkercad Picture




Code:

```
#define LED_RED 13
#define LED_GREEN 11
int ledRedState = LOW;
int ledGreenState = LOW;
// last time LED was updated
unsigned long prevRedTime = 0;
// will store last time LED was updated
unsigned long prevGreenTime = 0;
// interval at which to turn on (milliseconds)
const unsigned int redDelayTime = 1000;
const unsigned int greenDelayTime = 300;
void setup()
{
    pinMode(LED_RED, OUTPUT);
    pinMode(LED_GREEN, OUTPUT);
    digitalWrite(LED_RED, LOW);
    digitalWrite(LED_GREEN, LOW);
}

void loop()
{
    unsigned long currentTime = millis();

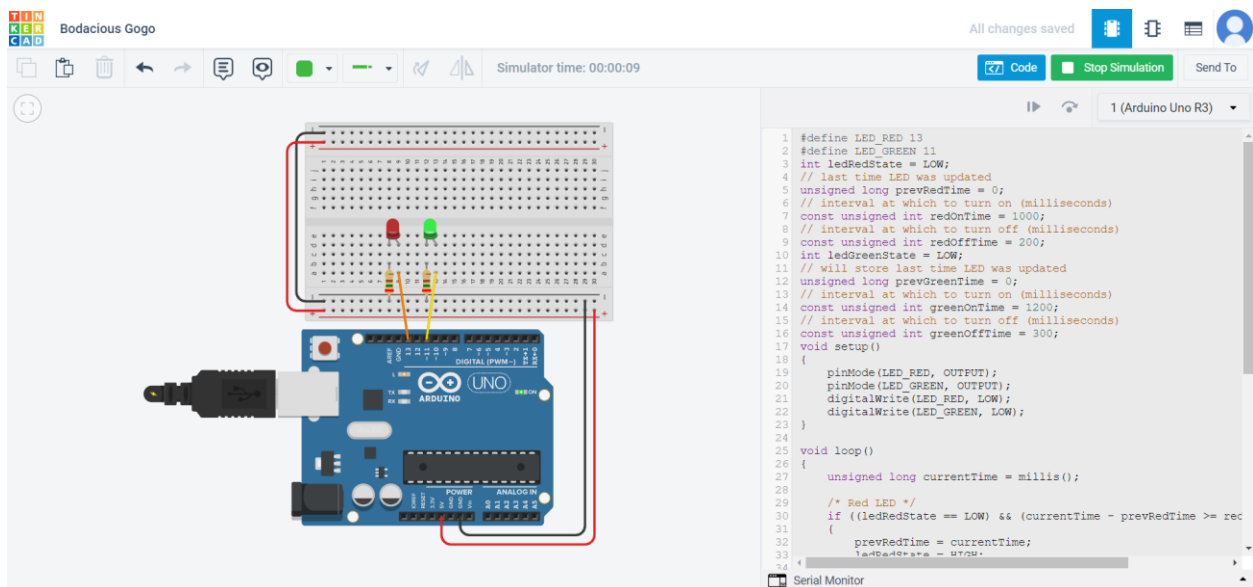
    /* Red LED */
    if (currentTime - prevRedTime >= redDelayTime)
    {
        prevRedTime = currentTime;
        ledRedState = !ledRedState;
        digitalWrite(LED_RED, ledRedState);
    }
    /* Green LED */
    if (currentTime - prevGreenTime >= greenDelayTime)
    {
        prevGreenTime = currentTime;
        ledGreenState = !ledGreenState;
        digitalWrite(LED_GREEN, ledGreenState);
    }
}
```



```
1  #define LED_RED 13
2  #define LED_GREEN 11
3  int ledRedState = LOW;
4  int ledGreenState = LOW;
5  // last time LED was updated
6  unsigned long prevRedTime = 0;
7  // will store last time LED was updated
8  unsigned long prevGreenTime = 0;
9  // interval at which to turn on (milliseconds)
10 const unsigned int redDelayTime = 1000;
11 const unsigned int greenDelayTime = 300;
12 void setup()
13 {
14     pinMode(LED_RED, OUTPUT);
15     pinMode(LED_GREEN, OUTPUT);
16     digitalWrite(LED_RED, LOW);
17     digitalWrite(LED_GREEN, LOW);
18 }
19
20 void loop()
21 {
22     unsigned long currentTime = millis();
23
24     /* Red LED */
25     if (currentTime - prevRedTime ≥ redDelayTime)
26     {
27         prevRedTime = currentTime;
28         ledRedState = !ledRedState;
29         digitalWrite(LED_RED, ledRedState);
30     }
31     /* Green LED */
32     if (currentTime - prevGreenTime ≥ greenDelayTime)
33     {
34         prevGreenTime = currentTime;
35         ledGreenState = !ledGreenState;
36         digitalWrite(LED_GREEN, ledGreenState);
37     }
38 }
```

}

## Group Work 2

*Tinkercad Picture*

Code:

```

#define LED_RED 13
#define LED_GREEN 11
int ledRedState = LOW;
// last time LED was updated
unsigned long prevRedTime = 0;
// interval at which to turn on (milliseconds)
const unsigned int redOnTime = 1000;
// interval at which to turn off (milliseconds)
const unsigned int redOffTime = 200;
int ledGreenState = LOW;
// will store last time LED was updated
unsigned long prevGreenTime = 0;
// interval at which to turn on (milliseconds)
const unsigned int greenOnTime = 1200;
// interval at which to turn off (milliseconds)
const unsigned int greenOffTime = 300;
void setup()
{
    pinMode(LED_RED, OUTPUT);
    pinMode(LED_GREEN, OUTPUT);
    digitalWrite(LED_RED, LOW);
    digitalWrite(LED_GREEN, LOW);
}

```

```
void loop()
{
    unsigned long currentTime = millis();

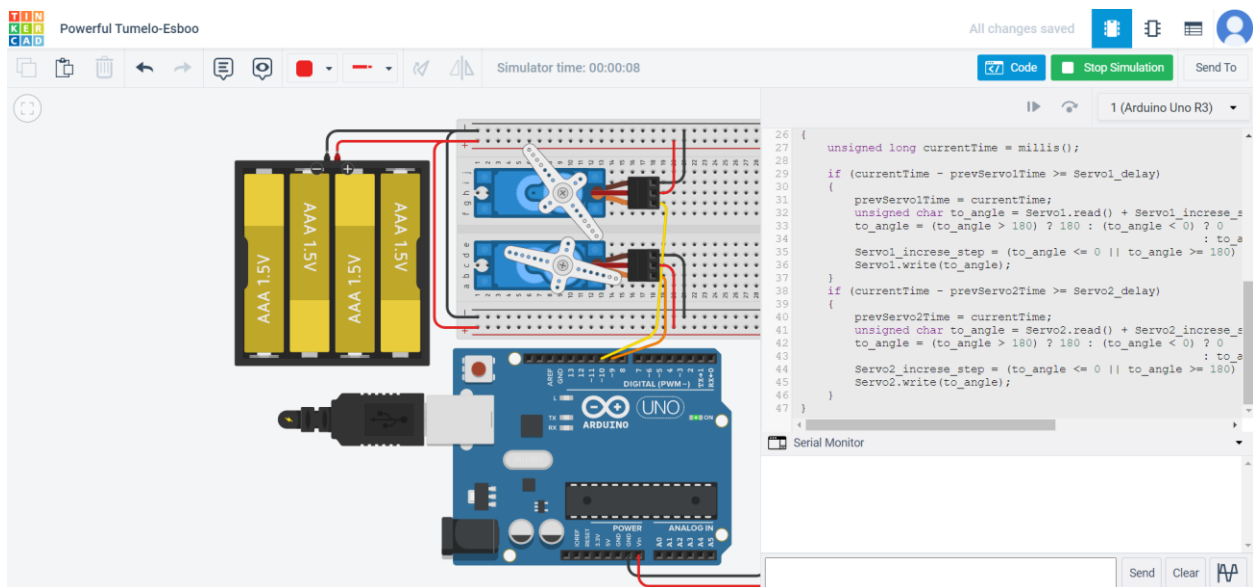
    /* Red LED */
    if ((ledRedState == LOW) && (currentTime - prevRedTime >=
redOffTime))
    {
        prevRedTime = currentTime;
        ledRedState = HIGH;
        digitalWrite(LED_RED, ledRedState);
    }
    else if ((ledRedState == HIGH) && (currentTime - prevRedTime
>= redOnTime))
    {
        prevRedTime = currentTime;
        ledRedState = LOW;
        digitalWrite(LED_RED, ledRedState);
    }

    /* Green LED */
    if ((ledGreenState == LOW) && (currentTime - prevGreenTime >=
greenOffTime))
    {
        prevGreenTime = currentTime;
        ledGreenState = HIGH;
        digitalWrite(LED_GREEN, ledGreenState);
    }
    else if ((ledGreenState == HIGH) && (currentTime -
prevGreenTime >= greenOnTime))
    {
        prevGreenTime = currentTime;
        ledGreenState = LOW;
        digitalWrite(LED_GREEN, ledGreenState);
    }
}
```

```
1  #define LED_RED 13
2  #define LED_GREEN 11
3  int ledRedState = LOW;
4  // last time LED was updated
5  unsigned long prevRedTime = 0;
6  // interval at which to turn on (milliseconds)
7  const unsigned int redOnTime = 1000;
8  // interval at which to turn off (milliseconds)
9  const unsigned int redOffTime = 200;
10 int ledGreenState = LOW;
11 // will store last time LED was updated
12 unsigned long prevGreenTime = 0;
13 // interval at which to turn on (milliseconds)
14 const unsigned int greenOnTime = 1200;
15 // interval at which to turn off (milliseconds)
16 const unsigned int greenOffTime = 300;
17 void setup()
18 {
19     pinMode(LED_RED, OUTPUT);
20     pinMode(LED_GREEN, OUTPUT);
21     digitalWrite(LED_RED, LOW);
22     digitalWrite(LED_GREEN, LOW);
23 }
24
25 void loop()
26 {
27     unsigned long currentTime = millis();
28
29     /* Red LED */
30     if ((ledRedState == LOW) && (currentTime - prevRedTime ≥ redOffTime))
31     {
32         prevRedTime = currentTime;
33         ledRedState = HIGH;
34         digitalWrite(LED_RED, ledRedState);
35     }
36     else if ((ledRedState == HIGH) && (currentTime - prevRedTime ≥ redOnTime))
37     {
38         prevRedTime = currentTime;
39         ledRedState = LOW;
40         digitalWrite(LED_RED, ledRedState);
41     }
42
43     /* Green LED */
44     if ((ledGreenState == LOW) && (currentTime - prevGreenTime ≥ greenOffTime))
45     {
46         prevGreenTime = currentTime;
47         ledGreenState = HIGH;
48         digitalWrite(LED_GREEN, ledGreenState);
49     }
50     else if ((ledGreenState == HIGH) && (currentTime - prevGreenTime ≥ greenOnTime))
51     {
52         prevGreenTime = currentTime;
53         ledGreenState = LOW;
54         digitalWrite(LED_GREEN, ledGreenState);
55     }
56 }
```



## Group Work 3

*Tinkercad Picture*

Code:

```

#include <Servo.h>

#define Servo1_pin 9
#define Servo2_pin 10

// delay in milliseconds
const unsigned int Servo1_delay = 100;
const unsigned int Servo2_delay = 200;

int Servo1_increse_step = 1;
int Servo2_increse_step = 1;

unsigned long prevServo1Time = 0;
unsigned long prevServo2Time = 0;

Servo Servo1;
Servo Servo2;

void setup()
{
  Servo1.attach(Servo1_pin);
  Servo2.attach(Servo2_pin);
}

```

```
void loop()
{
    unsigned long currentTime = millis();

    if (currentTime - prevServo1Time >= Servo1_delay)
    {
        prevServo1Time = currentTime;
        unsigned char to_angle = Servo1.read() + Servo1_increse_step;
        to_angle = (to_angle > 180) ? 180 : (to_angle < 0) ? 0
                                                    : to_angle;
        Servo1_increse_step = (to_angle <= 0 || to_angle >= 180) ? -
Servo1_increse_step : Servo1_increse_step;
        Servo1.write(to_angle);
    }
    if (currentTime - prevServo2Time >= Servo2_delay)
    {
        prevServo2Time = currentTime;
        unsigned char to_angle = Servo2.read() + Servo2_increse_step;
        to_angle = (to_angle > 180) ? 180 : (to_angle < 0) ? 0
                                                    : to_angle;
        Servo2_increse_step = (to_angle <= 0 || to_angle >= 180) ? -
Servo2_increse_step : Servo2_increse_step;
        Servo2.write(to_angle);
    }
}
```



```
1  #include <Servo.h>
2
3  #define Servo1_pin 9
4  #define Servo2_pin 10
5
6  // delay in milliseconds
7  const unsigned int Servo1_delay = 100;
8  const unsigned int Servo2_delay = 200;
9
10 int Servo1_increase_step = 1;
11 int Servo2_increase_step = 1;
12
13 unsigned long prevServo1Time = 0;
14 unsigned long prevServo2Time = 0;
15
16 Servo Servo1;
17 Servo Servo2;
18
19 void setup()
20 {
21     Servo1.attach(Servo1_pin);
22     Servo2.attach(Servo2_pin);
23 }
24
25 void loop()
26 {
27     unsigned long currentTime = millis();
28
29     if (currentTime - prevServo1Time ≥ Servo1_delay)
30     {
31         prevServo1Time = currentTime;
32         unsigned char to_angle = Servo1.read() + Servo1_increase_step;
33         to_angle = (to_angle > 180) ? 180 : (to_angle < 0) ? 0
34                     : to_angle;
35         Servo1_increase_step = (to_angle ≤ 0 || to_angle ≥ 180) ? -Servo1_increase_step : Servo1_increase_step;
36         Servo1.write(to_angle);
37     }
38     if (currentTime - prevServo2Time ≥ Servo2_delay)
39     {
40         prevServo2Time = currentTime;
41         unsigned char to_angle = Servo2.read() + Servo2_increase_step;
42         to_angle = (to_angle > 180) ? 180 : (to_angle < 0) ? 0
43                     : to_angle;
44         Servo2_increase_step = (to_angle ≤ 0 || to_angle ≥ 180) ? -Servo2_increase_step : Servo2_increase_step;
45         Servo2.write(to_angle);
46     }
47 }
```