



Arduino Uno



traffic.ino

```
1 int green = 10;
2 int yellow = 11;
3 int red = 12;
4
5 void setup() {
6     // put your setup code here, to run once:
7     pinMode(yellow,OUTPUT);
8     pinMode(red,OUTPUT);
9     pinMode(green,OUTPUT);
10 }
11
12 void lights(int v1,int v2,int v3){
13     digitalWrite(v1,HIGH);
14     digitalWrite(v2,LOW);
15     digitalWrite(v3,LOW);
16 }
17
18 void loop() {
19     // put your main code here, to run repeatedly:
20     lights(green,yellow,red);
21     delay(1000);
```

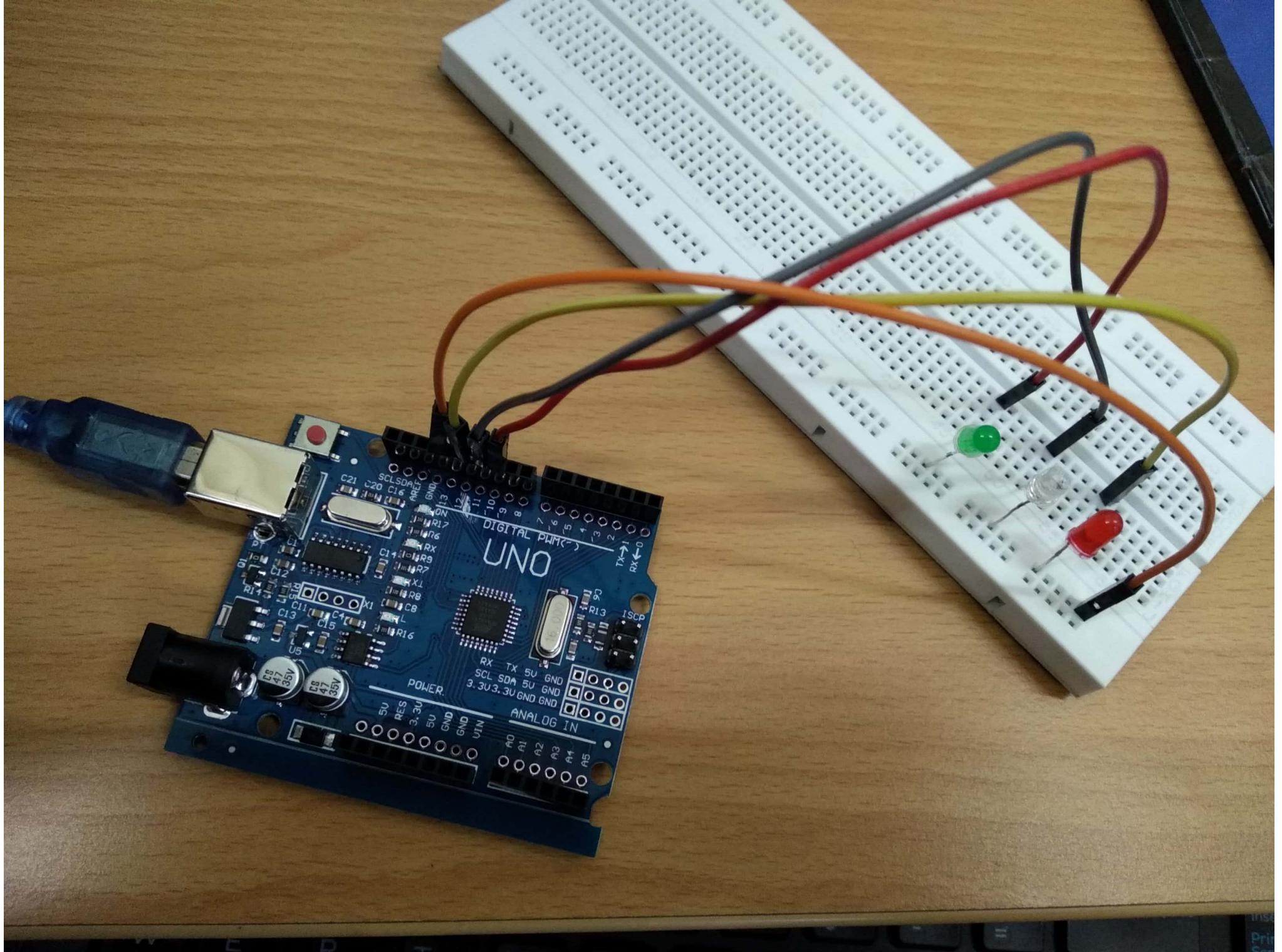
Output

Arduino Uno

traffic.ino

```
 7     pinMode(yellow,OUTPUT);
 8     pinMode(red,OUTPUT);
 9     pinMode(green,OUTPUT);
10 }
11
12 void lights(int v1,int v2,int v3){
13     digitalWrite(v1,HIGH);
14     digitalWrite(v2,LOW);
15     digitalWrite(v3,LOW);
16 }
17
18 void loop() {
19     // put your main code here, to run repeatedly:
20     lights(green,yellow,red);
21     delay(1000);
22     lights(yellow,green,red);
23     delay(1000);
24     lights(red,green,yellow);
25     delay(1000);
26 }
27
```

Output



 ⚠️ Arduino Uno at COM4

Blink.ino

∞ Blink.ino > ...

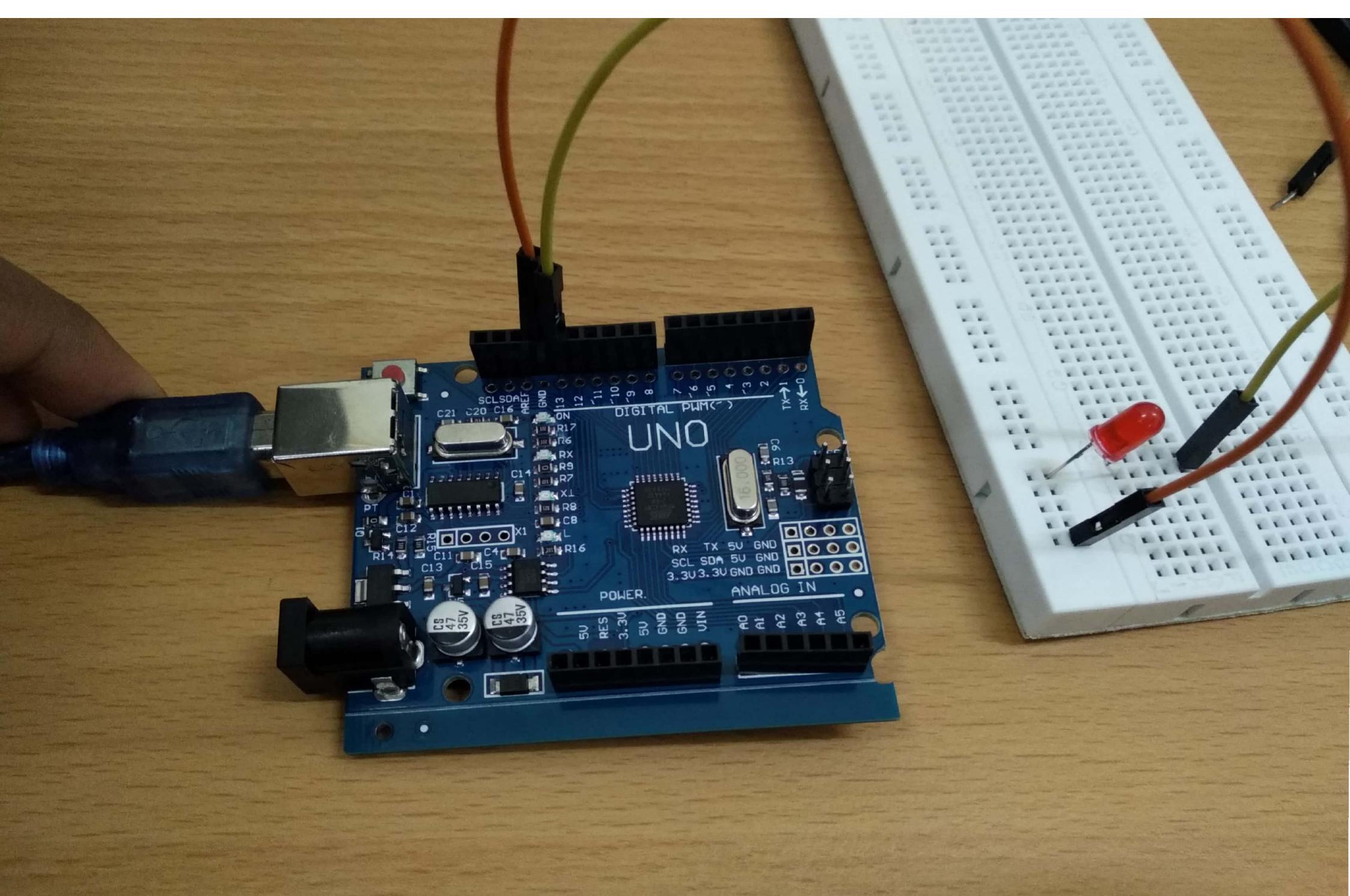
```
20  This example code is in the public domain.  
21  Follow link \(ctrl + click\)  
22  https://www.arduino.cc/en/Tutorial/BuiltInExamples/Blink  
23  */  
24  
25 // the setup function runs once when you press reset or power the board  
26 void setup() {  
27     // initialize digital pin LED_BUILTIN as an output.  
28     pinMode(LED_BUILTIN, OUTPUT);  
29 }  
30  
31 // the loop function runs over and over again forever  
32 void loop() {  
33     digitalWrite(LED_BUILTIN, HIGH);      // turn the LED on (HIGH is the voltage level)  
34     delay(1000);                      // wait for a second  
35     digitalWrite(LED_BUILTIN, LOW);       // turn the LED off by making the voltage level  
36     delay(1000);                      // wait for a second  
37 }  
38 }
```

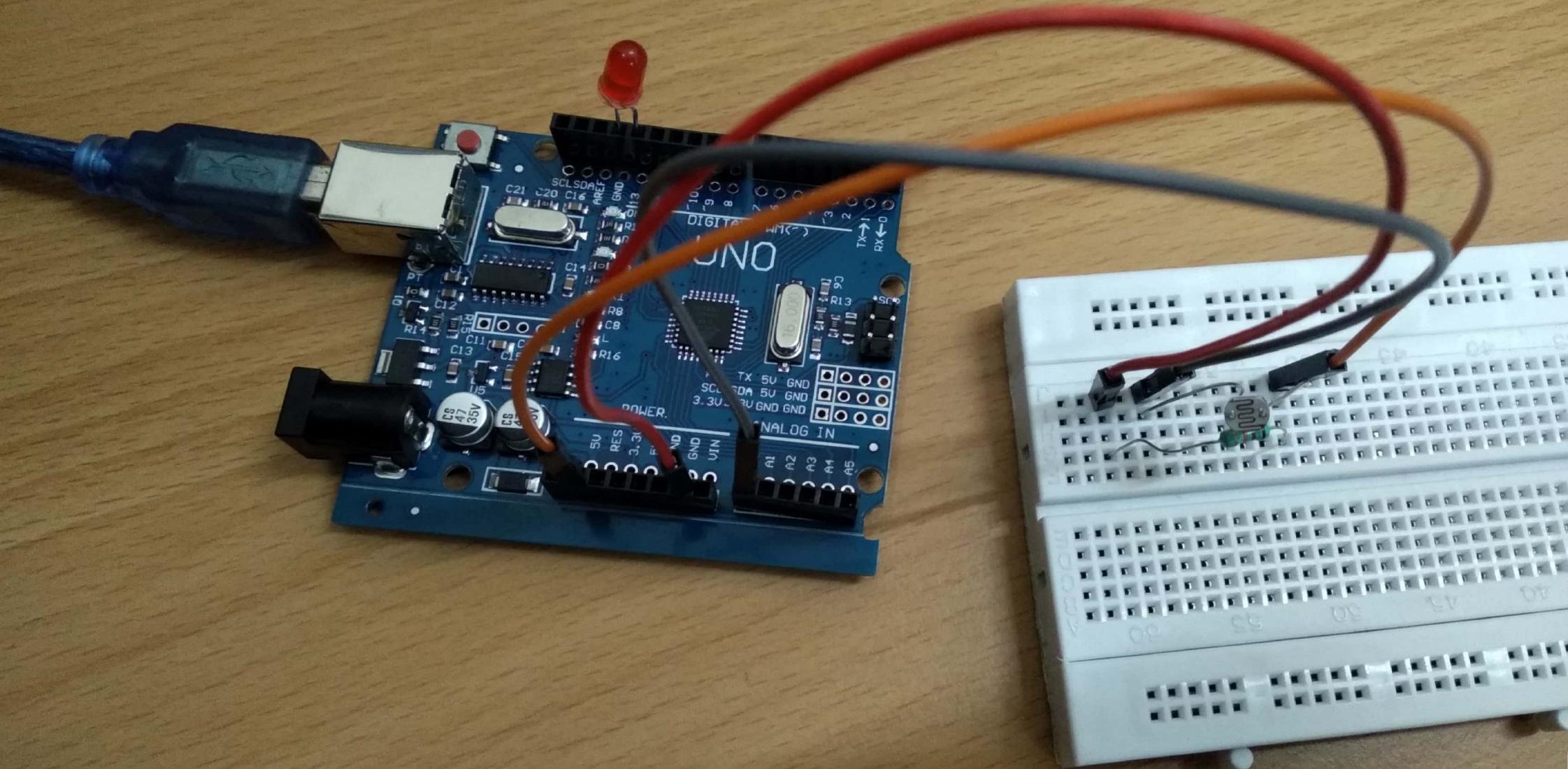
Output

Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local

Compilation complete.

upload complete.







Arduino Uno



prg3.ino



```
prg3.ino > setup
4 int potpin=0;
5 int ledpin=13;
6 int val=0;
7
8
9 void setup() {
10 // put your setup code here, to run once:
11 pinMode(ledpin,OUTPUT);
12 Serial.begin(9600);
13 }
14
15
16 void loop() {
17 // put your main code here, to run repeatedly:
18 val = analogRead(potpin);
19 Serial.println(val);
20 analogWrite(ledpin,val);
21 delay(10);
22 if(val>100)
```



Output Serial Monitor X

Not connected. Select a board and a port to connect automatically.

Arduino Uno

prg3.ino

```
prg3.ino > loop
13
14 }
15
16 void loop() {
17     // put your main code here, to run repeatedly:
18     val = analogRead(potpin);
19     Serial.println(val);
20     analogWrite(ledpin,val);
21     delay(10);
22     if(val>100)
23     {
24         digitalWrite(ledpin,HIGH);
25         delay(1000);
26     }else{
27         digitalWrite(ledpin,LOW);
28         delay(1000);
29     }
30 }
31
```

Output Serial Monitor X

Not connected. Select a board and a port to connect automatically.

Arduino Uno

Button.ino

```
 20  This example code is in the public domain.
 21
 22  https://www.arduino.cc/en/Tutorial/BuiltInExamples/Button
 23  */
 24
 25 // constants won't change. They're used here to set pin numbers:
 26 const int buttonPin = 2;      // the number of the pushbutton pin
 27 const int ledPin = 13;        // the number of the LED pin
 28
 29 // variables will change:
 30 int buttonState = 0;          // variable for reading the pushbutton status
 31
 32 void setup() {
 33     // initialize the LED pin as an output:
 34     pinMode(ledPin, OUTPUT);
 35     // initialize the pushbutton pin as an input:
 36     pinMode(buttonPin, INPUT);
 37 }
 38
 39 void loop() {
 40     // read the state of the pushbutton value:
 41     buttonState = digitalRead(buttonPin);
 42
 43     // check if the pushbutton is pressed. If it is, the buttonState is HIGH:
 44     if (buttonState == HIGH) {
 45         // turn LED on:
 46         digitalWrite(ledPin, HIGH);
 47     } else {
 48         // turn LED off:
 49         digitalWrite(ledPin, LOW);
 50     }
 51 }
```



Type here to search





⚠️ Arduino Uno at COM4



Debounce.ino



```
Debounce.ino > ledState
28 */
29
30 // constants won't change. They're used here to set pin numbers:
31 const int buttonPin = 2;      // the number of the pushbutton pin
32 const int ledPin = 13;        // the number of the LED pin
33
34 // Variables will change:
35 int ledState = HIGH;         // the current state of the output pin
36 int buttonState;             // the current reading from the input pin
37 int lastButtonState = LOW;   // the previous reading from the input pin
38
39 // the following variables are unsigned longs because the time, measured in
40 // milliseconds, will quickly become a bigger number than can be stored in a
41 unsigned long lastDebounceTime = 0; // the last time the output pin was toggled
42 unsigned long debounceDelay = 50;  // the debounce time; increase if the debouncing
43
44 void setup() {
45     pinMode(buttonPin, INPUT);
46     pinMode(ledPin, OUTPUT);
```



Output

Sketch uses 1116 bytes (3%) of program storage space. Maximum is 32256 bytes.
Global variables use 19 bytes (0%) of dynamic memory, leaving 2029 bytes for local

Compilation complete.

upload complete.

Debounce.ino

```
Debounce.ino > ...
39 // the following variables are unsigned longs because the time, m
40 // milliseconds, will quickly become a bigger number than can be h
41 unsigned long lastDebounceTime = 0; // the last time the output p
42 unsigned long debounceDelay = 50; // the debounce time; increase t
43
44 void setup() {
45     pinMode(buttonPin, INPUT);
46     pinMode(ledPin, OUTPUT);
47
48     // set initial LED state
49     digitalWrite(ledPin, ledState);
50 }
51
52 void loop() {
53     // read the state of the switch into a local variable:
54     int reading = digitalRead(buttonPin);
55
56     if (reading != lastButtonState) {
57         // reset the debouncing timer
```

Output

Sketch uses 1116 bytes (3%) of program storage space. Maximum is 32256 bytes
Global variables use 19 bytes (0%) of dynamic memory, leaving 2029 bytes

Compilation complete.

upload complete.

Arduino Uno at COM4

```
Debounce.ino

53 // read the state of the switch into a local variable:
54 int reading = digitalRead(buttonPin);

55

56 if (reading != lastButtonState) {
57     // reset the debouncing timer
58     lastDebounceTime = millis();
59 }

60

61 if ((millis() - lastDebounceTime) > debounceDelay) {
62     if (reading != buttonState) {
63         buttonState = reading;

64         // only toggle the LED if the new button state is HIGH
65         if (buttonState == HIGH) {
66             ledState = !ledState;
67         }
68     }
69 }
70 }

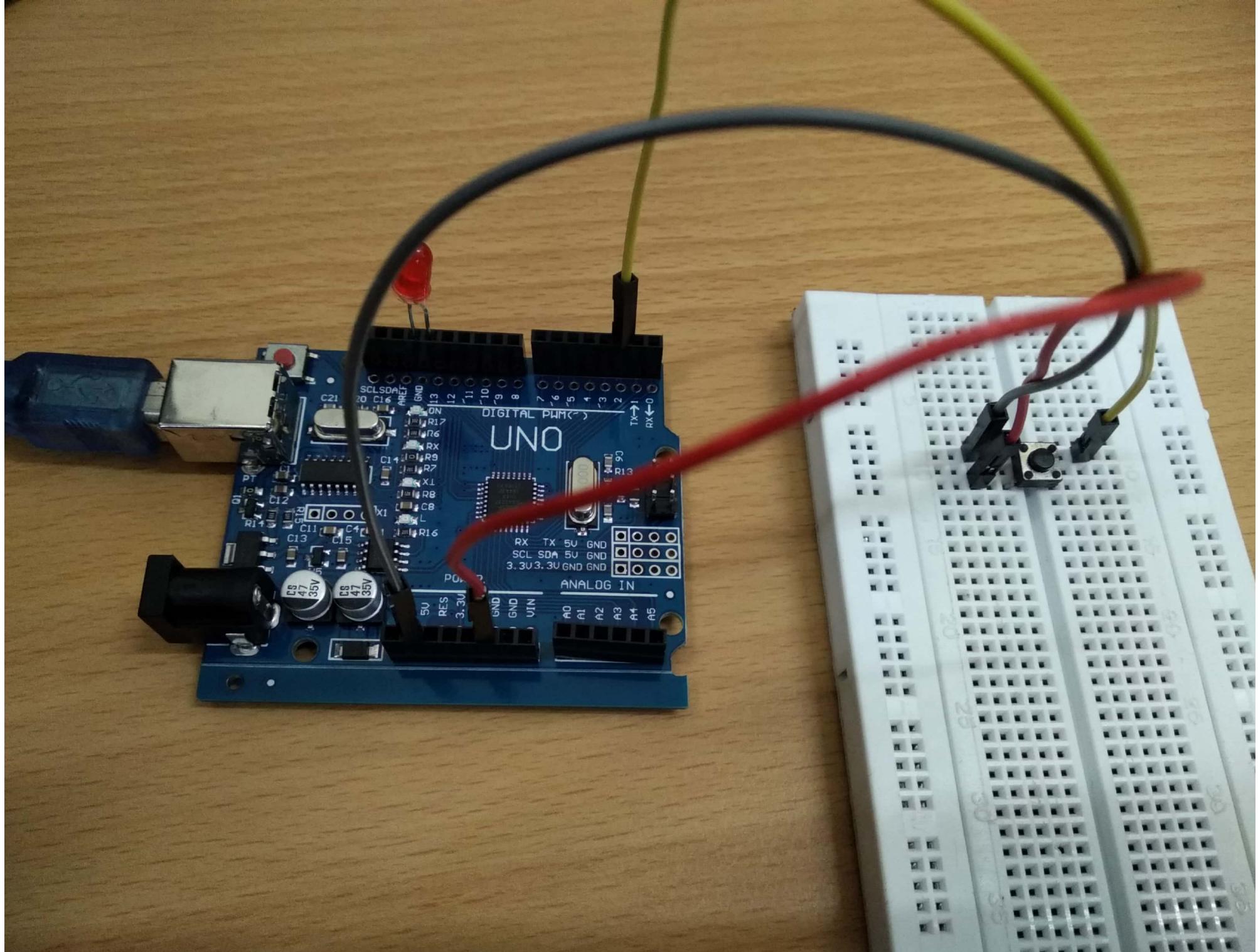
71 }
```

Output

```
Sketch uses 1116 bytes (3%) of program storage space. Maximum is 32256 bytes.
Global variables use 19 bytes (0%) of dynamic memory, leaving 2029 bytes for local var

-----
Compilation complete.

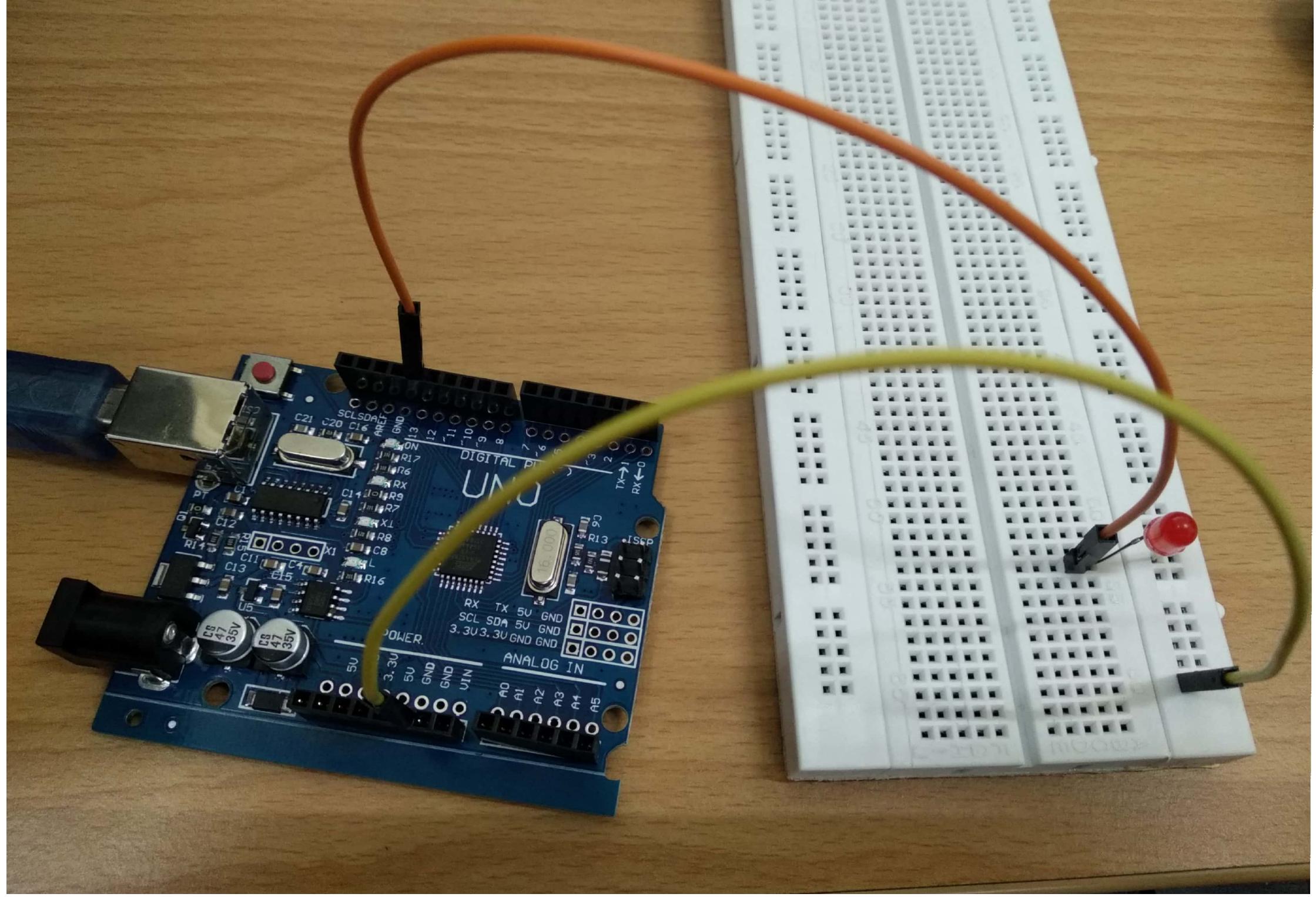
-----
upload complete.
```

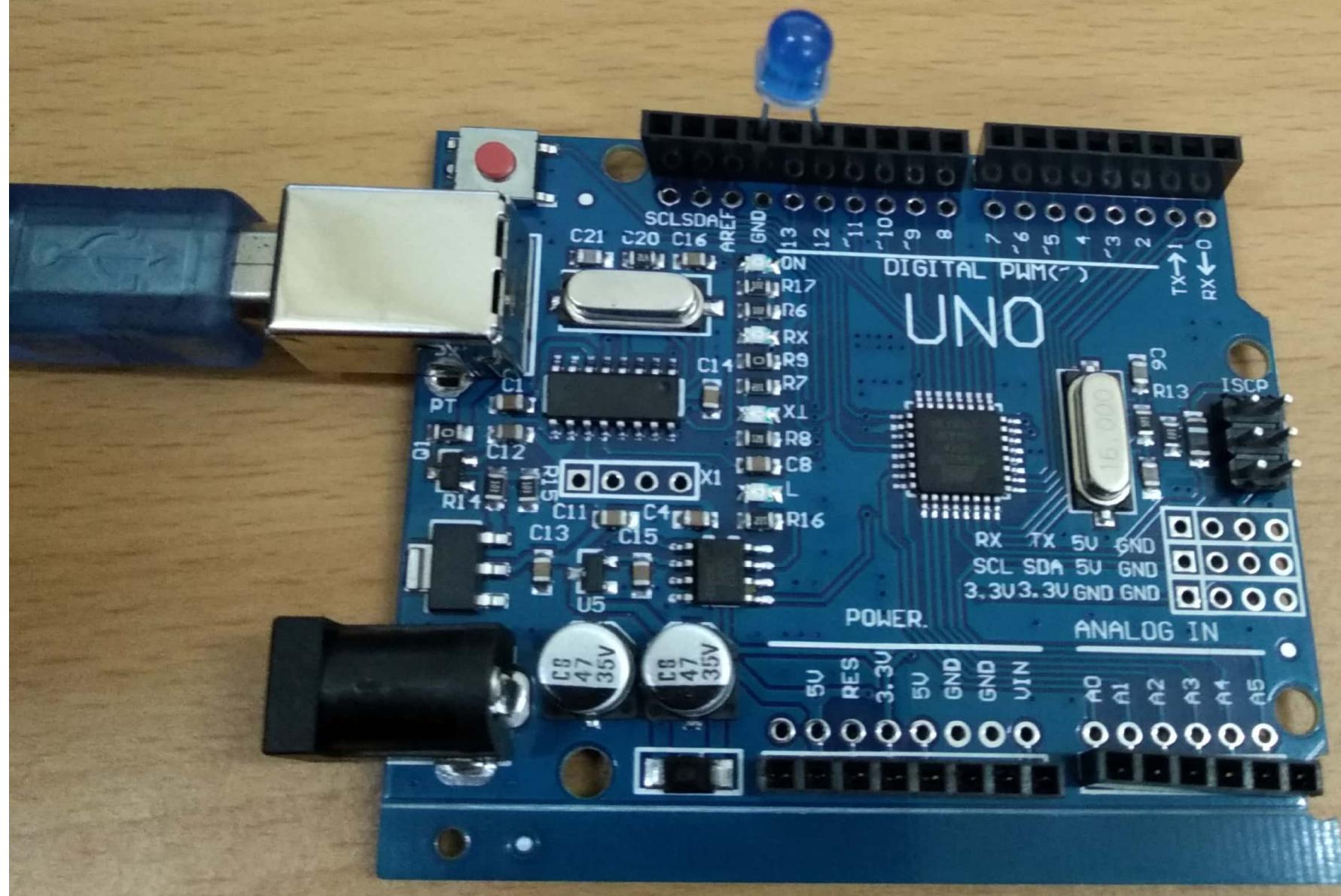


Arduino Uno

PRG5.ino

```
PRG5.ino > ...
1 int val;
2 void setup() {
3     // put your setup code here, to run once:
4     Serial.begin(9600);
5     pinMode(12,OUTPUT);
6
7 }
8
9 void loop() {
10    // put your main code here, to run repeatedly:
11    if(Serial.available()>=0){
12        val=Serial.read();
13        if(val=='1'){
14            digitalWrite(12,HIGH);
15            delay(5);
16            Serial.println("LEd is on");
17        }
18
19
20        if(val=='0'){
21            digitalWrite(12,LOW);
22            delay(5);
23            Serial.println("LEd is oFF");
24
25    }
26 }
27 }
28 }
```





```
DHT11Default
```

```
#include <SimpleDHT.h>

// for DHT11,
//      VCC: 5V or 3V
//      GND: GND
//      DATA: 2
int pinDHT11 = 2;
SimpleDHT11 dht11(pinDHT11);

void setup() {
    Serial.begin(115200);
}

void loop() {
    // start working...
    Serial.println("=====");
    Serial.println("Sample DHT11...");

    // read without samples.
    byte temperature = 0;
    byte humidity = 0;
    int err = SimpleDHTErrSuccess;
    if ((err = dht11.read(&temperature, &humidity, NULL)) != SimpleDHTErrSuccess) {
        Serial.print("Read DHT11 failed, err="); Serial.print(SimpleDHTErrCode(err));
        Serial.print(","); Serial.println(SimpleDHTErrDuration(err)); delay(1000);
        return;
    }

    Serial.print("Sample OK: ");
    Serial.print((int)temperature); Serial.print(" °C, ");
    Serial.print((int)humidity); Serial.println(" H");

    // DHT11 sampling rate is 1HZ.
    delay(1500);
}
```

COM4

Send

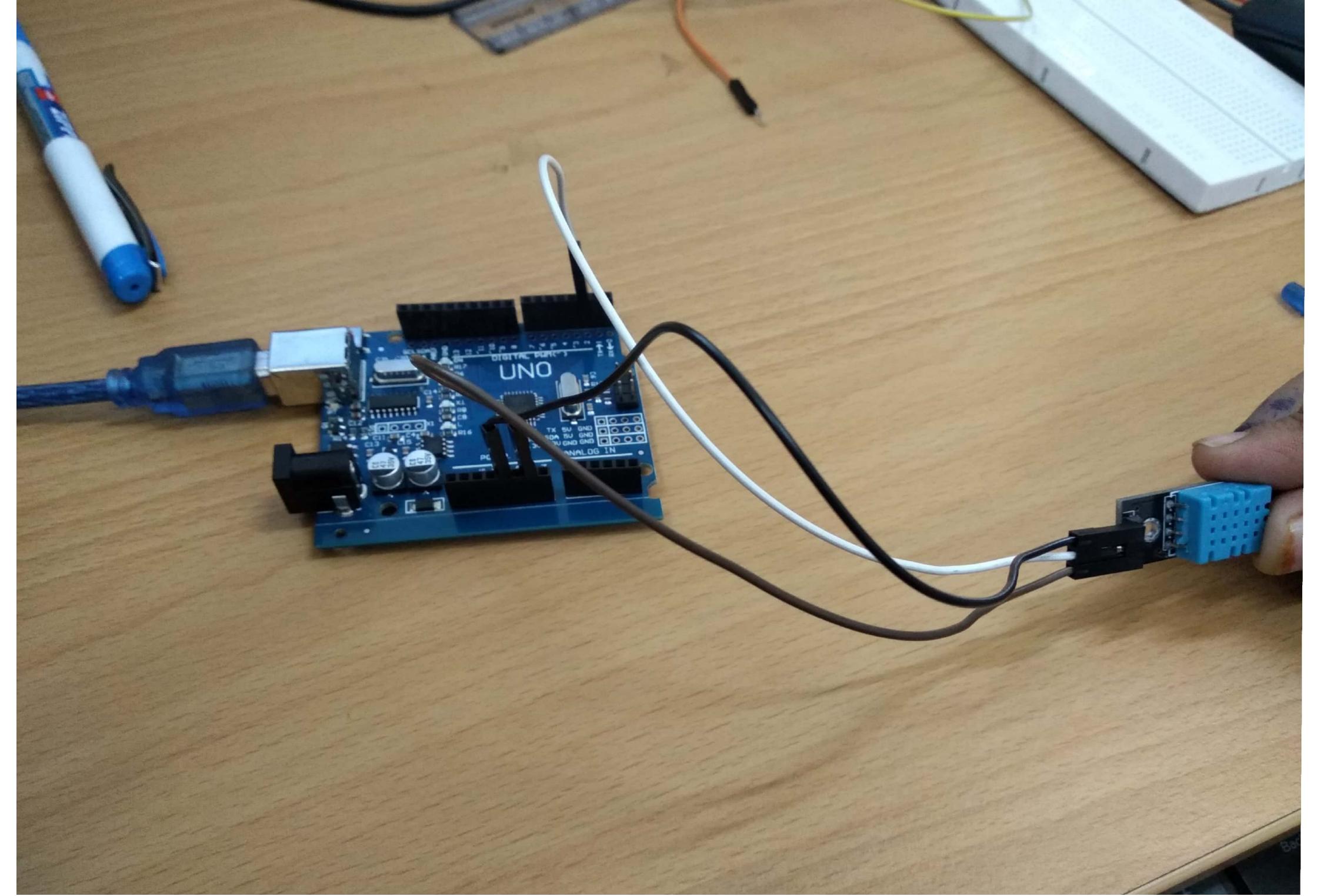
```
17:42:41.841 -> -----
17:42:41.841 -> Sample DHT11...
17:42:41.888 -> Sample OK: 29 °C, 57 H
17:42:43.365 -> -----
17:42:43.365 -> Sample DHT11...
17:42:43.412 -> Sample OK: 29 °C, 57 H
17:42:44.885 -> -----
17:42:44.885 -> Sample DHT11...
17:42:44.932 -> Sample OK: 29 °C, 57 H
17:42:46.402 -> -----
17:42:46.402 -> Sample DHT11...
17:42:46.449 -> Sample OK: 29 °C, 57 H
17:42:47.948 -> -----
17:42:47.948 -> Sample DHT11...
17:42:47.948 -> Sample OK: 29 °C, 57 H
```

Autoscroll Show timestamp

Newline

115200 baud

Clear output



eta.12

Uno at COM4

ULTRASONIC_-_PRG8.ino

```
00 ULTRASONIC_-_PRG8.ino > ...
1 int trigpin = 7;
2 int echopin =10;
3 int led=11;
4
5 void setup(){
6
7 // Serial.begin(9600);
8 pinMode(led,OUTPUT);
9 pinMode(trigpin,OUTPUT);
10 pinMode(echopin,INPUT);
11 }
12
13 void loop()
14 {
15 int duration,distance;
16 digitalWrite(trigpin,HIGH);
17 delay(1);
18 digitalWrite(trigpin,LOW);
19 //delayMicroseconds(10);
```

Output

Arduino Uno at COM4 ▾

ULTRASONIC_-_PRG8.ino

```
20 ULTRASONIC_-_PRG8.ino > ...
+-
13 void loop()
14 {
15     int duration,distance;
16     digitalWrite(trigpin,HIGH);
17     delay(1);
18     digitalWrite(trigpin,LOW);
19     //delayMicroseconds(10);
20     //digitalWrite(trigpin,LOW);
21     duration = pulseIn(echopin,HIGH);
22     distance = (duration/2)/29.1;
23     Serial.print(distance);
24     if(distance<=5 && distance>=0){
25         digitalWrite(led,HIGH);
26     }
27     else{
28         digitalWrite(led,LOW);
29     }
30
31     delay(10);
```

