Basic Electronics

COSC 1010

Electrical Currents

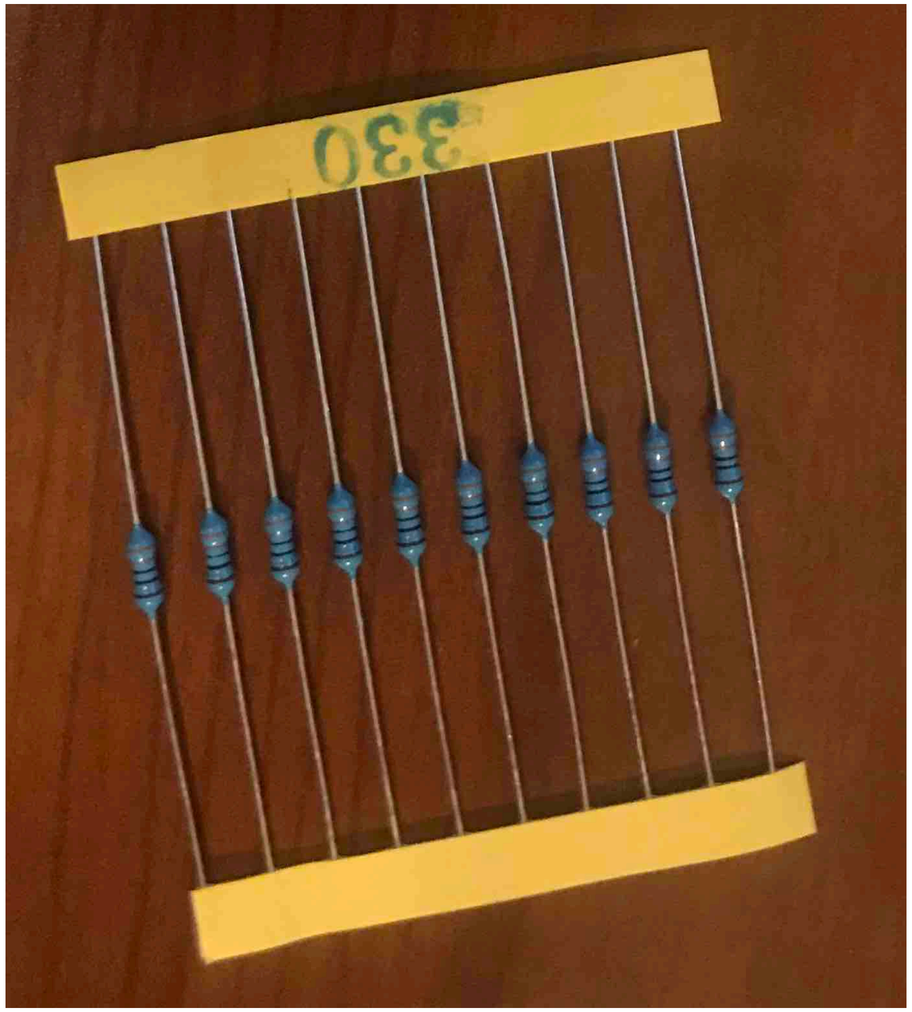
- Electronics uses electrical currents
  - Flow from positive to negative
    - Even though electrons flow in the opposite direction
  - First component:
    - A power source
    - A resistor
    - A Light Emitting Diode

Electrical Currents

- Current is determined by voltage and resistance in the circuit
    - Ohm's Law:
        - Current I in Ampere
        - Voltage U in Volt
        - Resistance R in Ohm
        - $I = \frac{U}{R}$
    - In general, we do not want very high currents

Electrical Currents

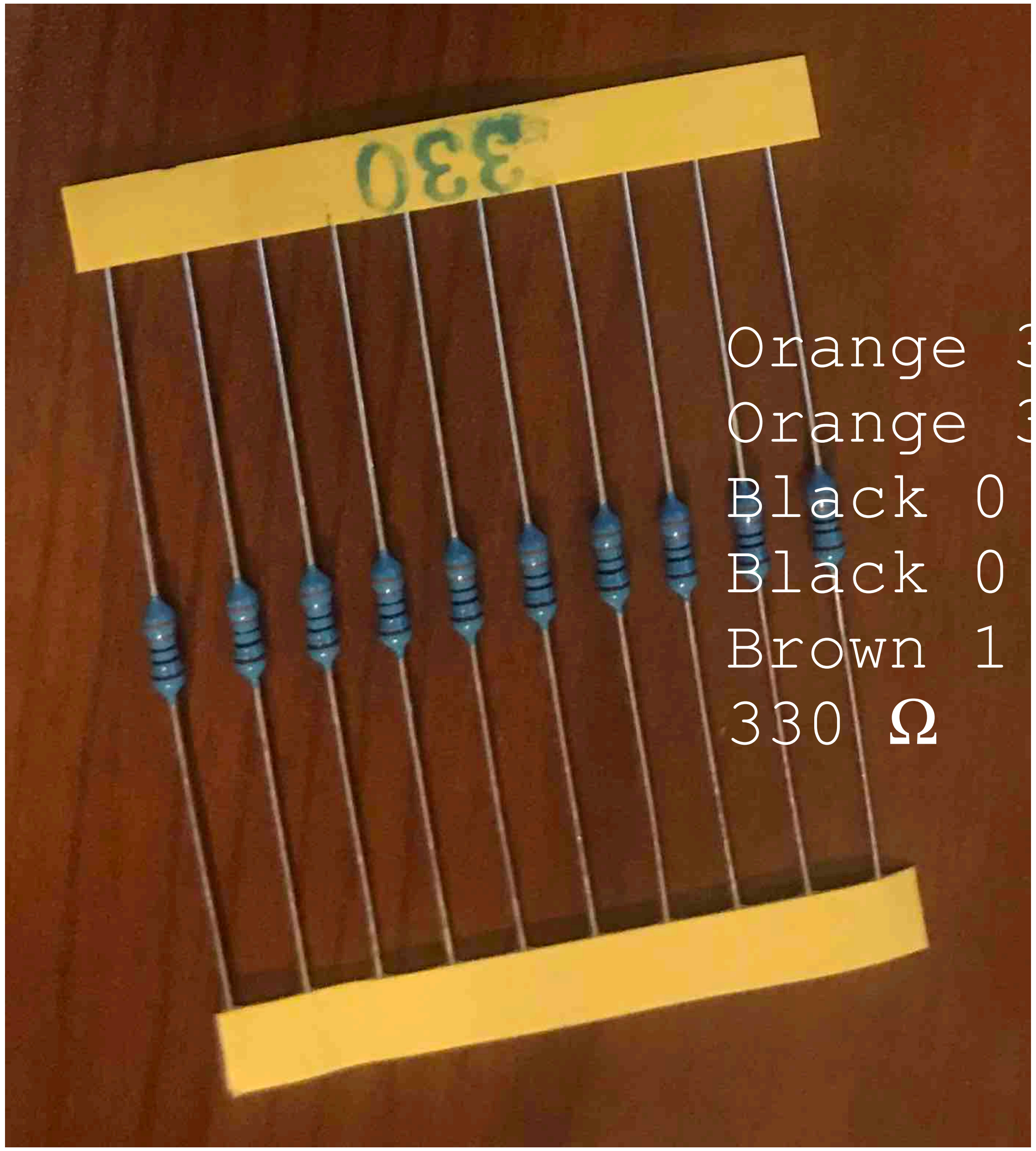
- Resistors
  - Limit current
    - Level of resistance from hundreds of Ω to $M\Omega$:
      - First band: First digit
      - Second band: Second digit
      - Third band: third digit or multiplier
      - Fourth band: Multiplier for five band resistors
      - Fifth band: Tolerance



Electrical Currents

- Color Codes
  -

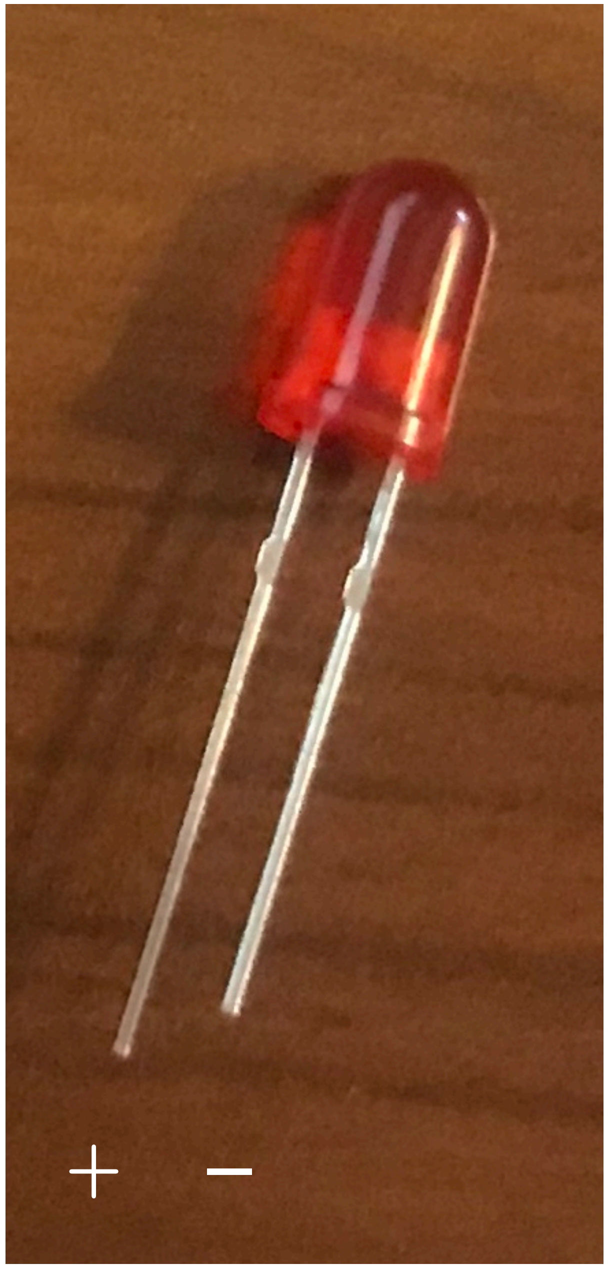
Color	Value
Black	0
Brown	1
Red	2
Orange	3
Yellow	4
Green	5
Blue	6
Violet	7
Gray	8
White	9



Orange 3
Orange 3
Black 0
Black 0
Brown 1
330 Ω

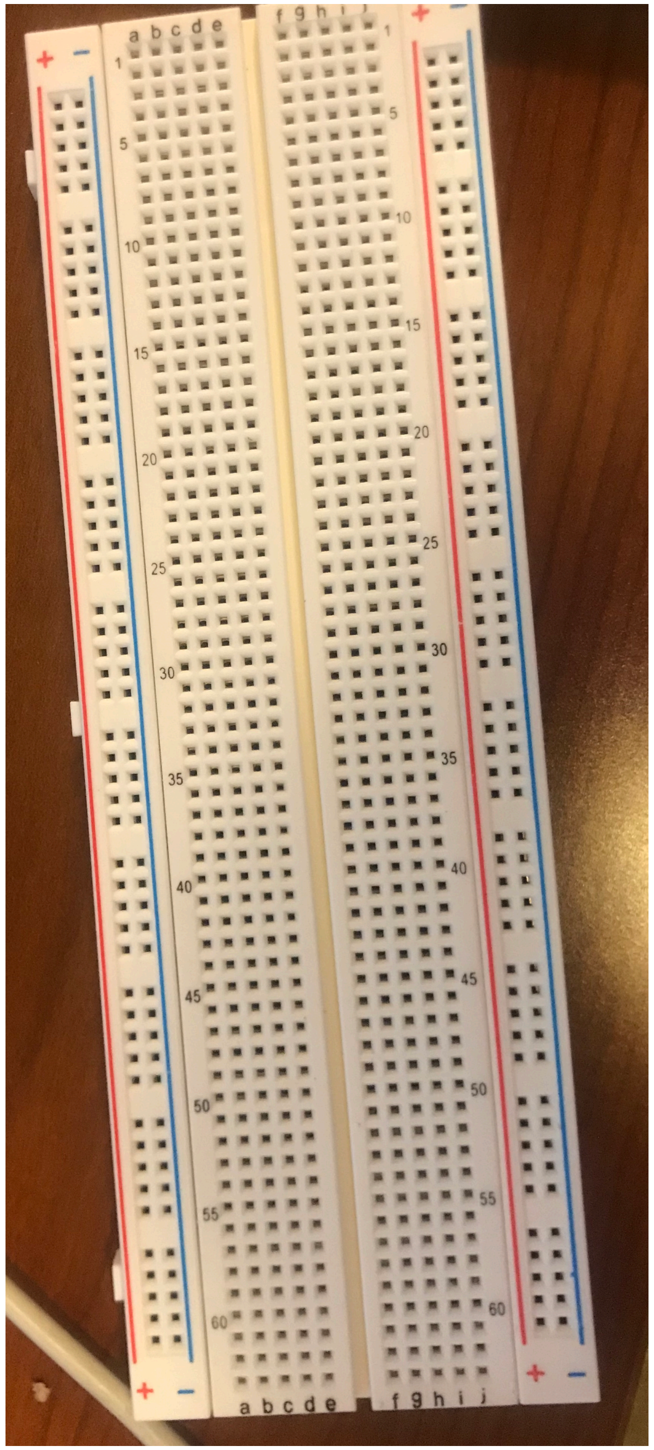
Electrical Currents

- Light Emiting Diodes
  - Diodes only allow current to pass in **one** direction
    - From Anode (+) to Cathode
    - Anode is long wire, Cathode is short wire
  - Would be destroyed without an additional resistor to limit current



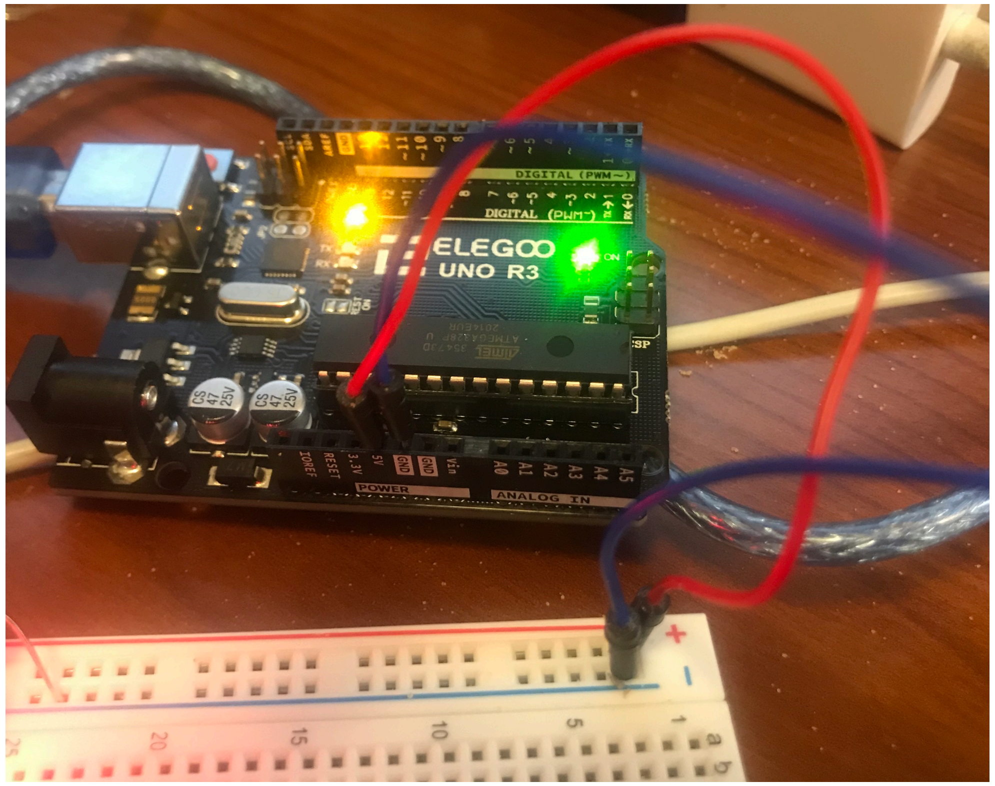
Electrical Currents

- To create circuits:
  - Use a bread-board
    - Solderless: just stick wires into the connectors
  - Red and Left Edge: Two horizontal rows for voltage (red) and ground (blue)
  - In the middle: Rows are connected



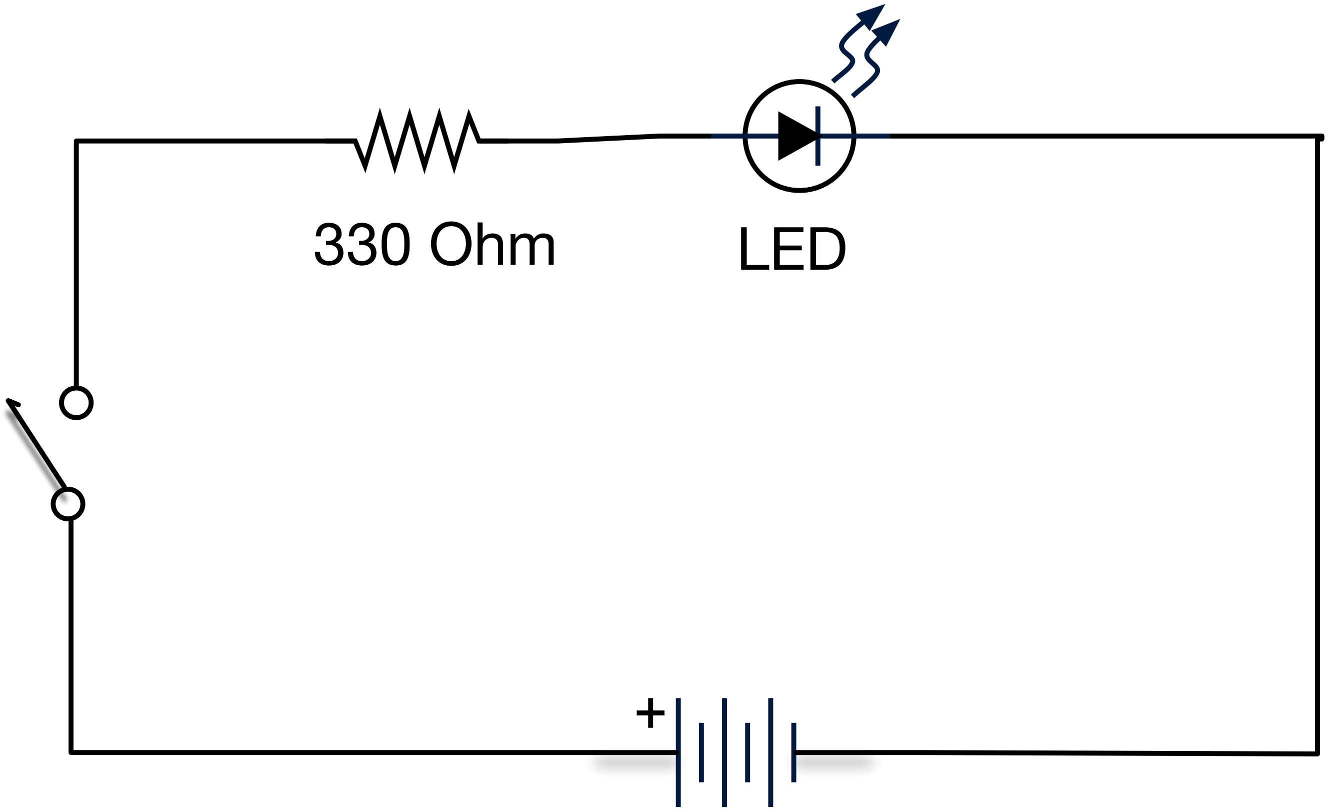
Electrical Currents

- To connect to Arduino:
    - Take a red wire and connect to 5V
    - Take a blue wire and connect to Gnd
        - Make sure you stick in the wire ends well



Electrical Currents

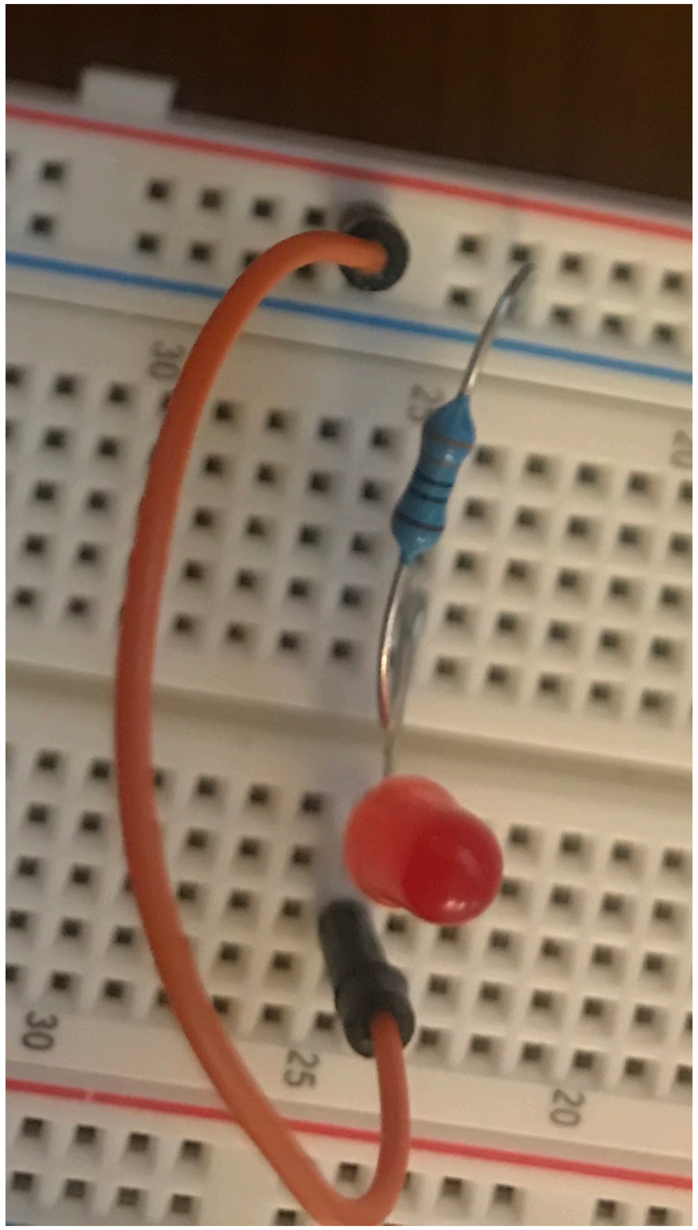
- Now we create the following circuit
  - (Without the switch)



Power and Ground

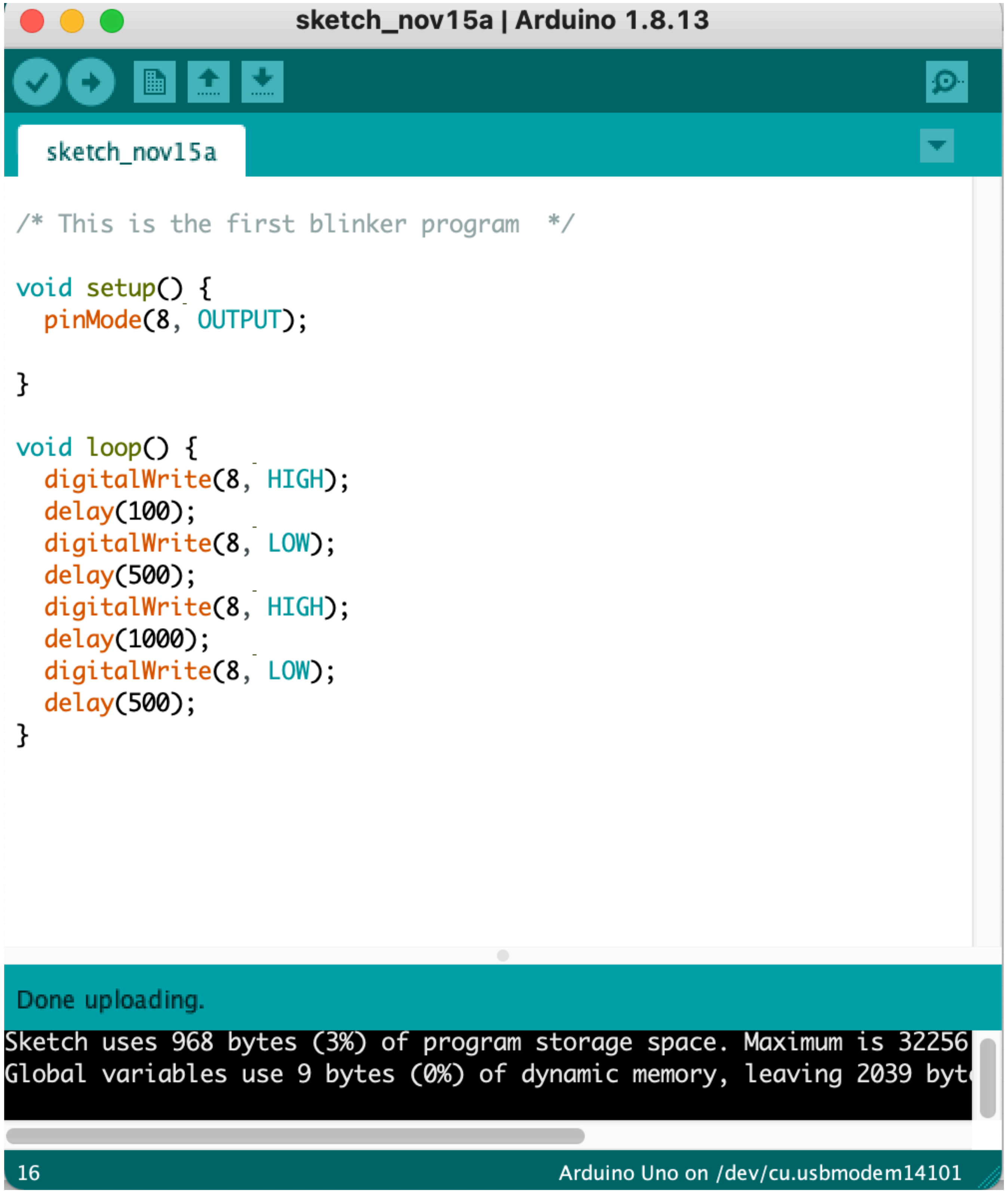
Electrical Currents

- Stick LED and resistor in the same row of the breadboard
- Connect with + and -
- Make sure that the long wire of the LED is connected to + (or nothing will happen)
- Wiggle the connections to make sure that contacts are good
- LED should shine brightly
  - (If not: bad contact or error in implementation)



Electrical Currents

- If this works, modify the previous sketch
  - Use open to find the sketch file
  - Change the name of the output to 8
  - Connect the red (plus) wire to analog pin 8 on the other side of the Arduino
- Compile and run


sketch_nov15a | Arduino 1.8.13
sketch_nov15a

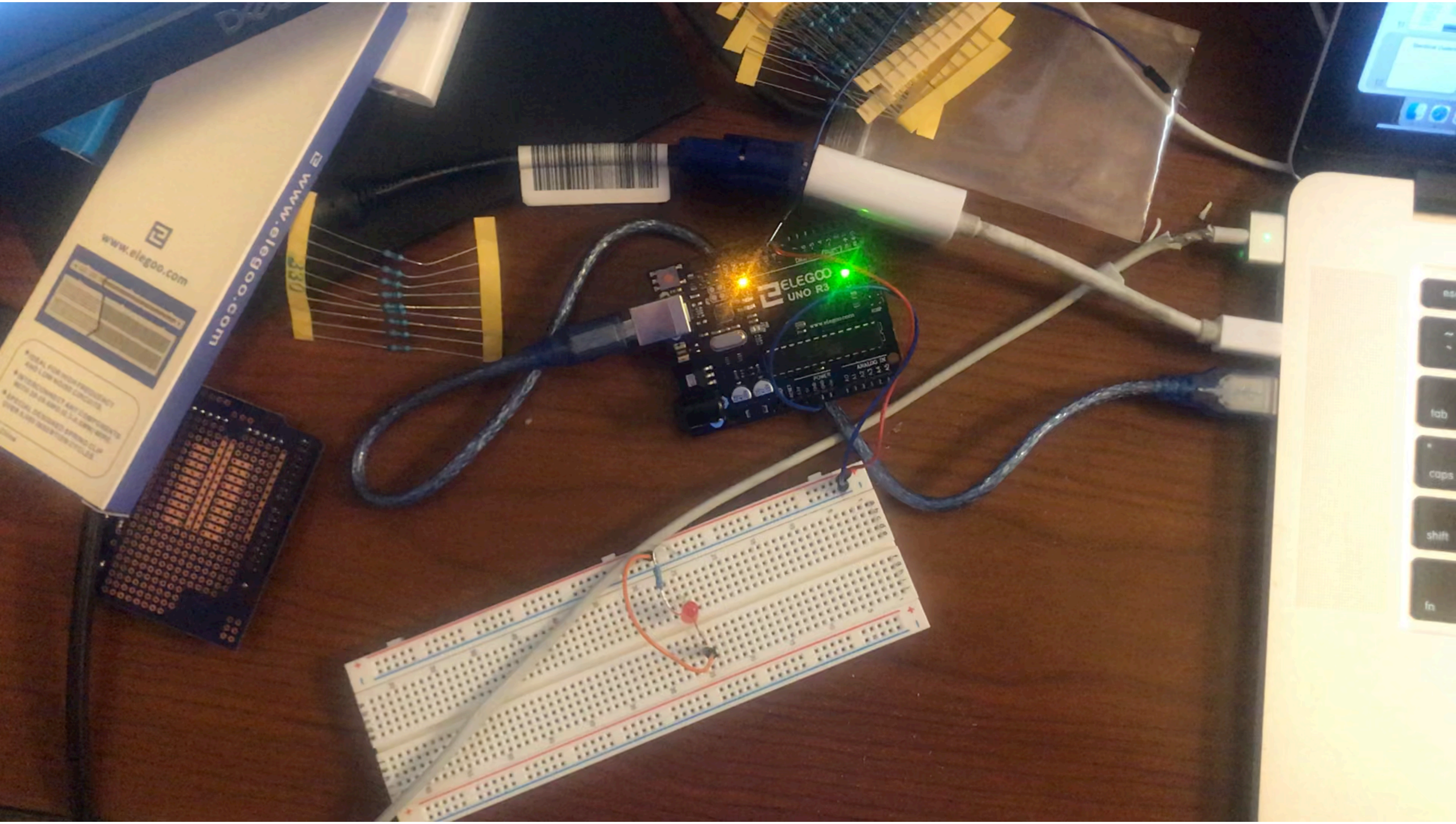
```
/* This is the first blinker program  */

void setup() {
  pinMode(8, OUTPUT);

}

void loop() {
  digitalWrite(8, HIGH);
  delay(100);
  digitalWrite(8, LOW);
  delay(500);
  digitalWrite(8, HIGH);
  delay(1000);
  digitalWrite(8, LOW);
  delay(500);
}
```

Done uploading.

```
Sketch uses 968 bytes (3%) of program storage space. Maximum is 32256
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 byt
```

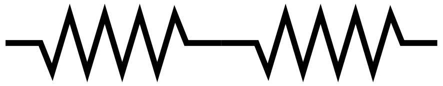
16
Arduino Uno on /dev/cu.usbmodem14101

Electrical Currents



Electrical Currents

- You can experiment by putting in a bigger resistor
  - Which you achieve by putting two resistors in series



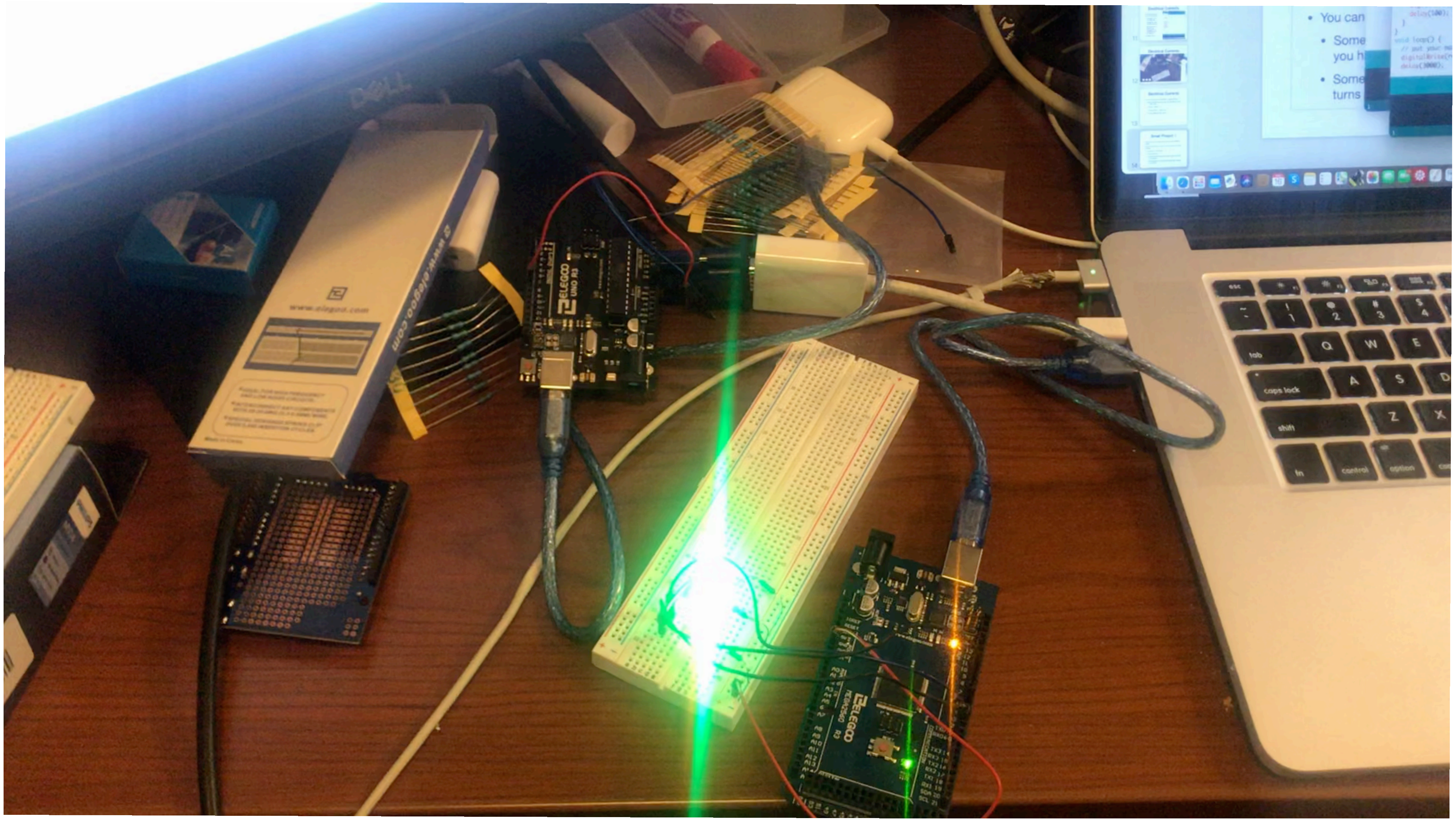
330 Ohm 200 Ohm

    - This would be ~ 550 Ohms
  - Or using different color LEDs

Small Project 1

- Reset the circuit so that you are using three different LEDs
- Connect the ground to ground and the others to Digital Outputs
- Then program a traffic light
  - You can be fancy:
    - Some countries have the red light flash just before you hit green
    - Some countries have the green light flash before it turns yellow

Example



Here I am using an Arduino Mega

Software

- Code needs to be translated into machine language
    - What the System-On-a-Chip (SOC) understands
- Human readable text is pre-processed
- The result is then compiled
    - And in real life combined with other code, but we will not do this for a while
- In the Arduino, we can now upload the machine language program
    - and the Arduino will start executing it

Software

- Arduino Sketches are written in C
    - Can (and should) write explanatory text as comments
    - Two types: /* — */ combination can span lines
        - `/*  I am a comment   */`
        - `// I am a comment that ends at the line`

Software

- Can use pre-processor commands
  - A pre-processor passes through the code before it gets compiled
  - Usually definitions and inclusions of libraries
  - Example:

```
#define red 2
#define yellow 3
#define green 4
```

- Now all references to “red” will be replaced by 2

Software

- Preprocessor commands
- We read

```
pinMode(red, OUTPUT);
```

  - is translated by the preprocessor into

```
pinMode(2, OUTPUT);
```

- Every occurrence of the string “red” is replaced with an occurrence of the string “2”.

Software

Software

Software

Software