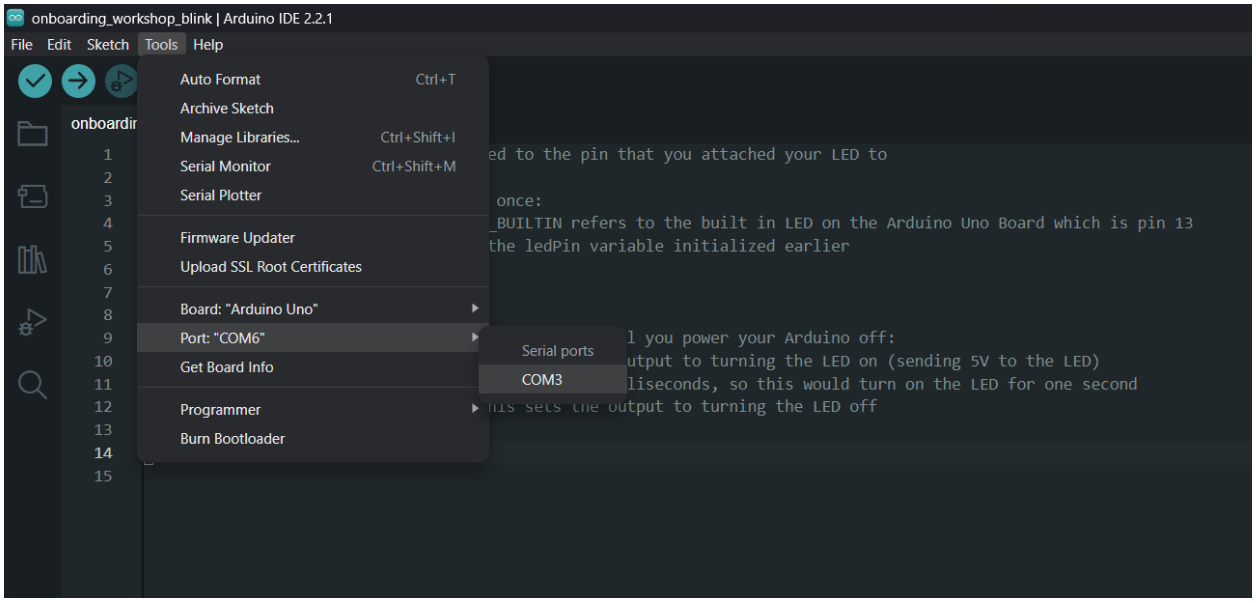
Arduino Onboarding Project:

Download the Arduino IDE for your operating system, if not installed on your computer already, using https://www.arduino.cc/en/software. After installation, connect the Arduino board via USB and select the correct COM port, shown in the image below. This COM port allows the IDE to communicate with the board, there may be multiple selections depending on how many USB connections you currently have on your laptop. Just keep trying until you are able to upload your code to the Arduino board.

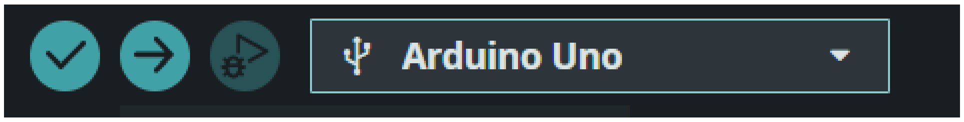


```
ed to the pin that you attached your LED to

 once:
_BUILTIN refers to the built in LED on the Arduino Uno Board which is pin 13
the ledPin variable initialized earlier

l you power your Arduino off:
utput to turning the LED on (sending 5V to the LED)
liseconds, so this would turn on the LED for one second
his sets the output to turning the LED off
```

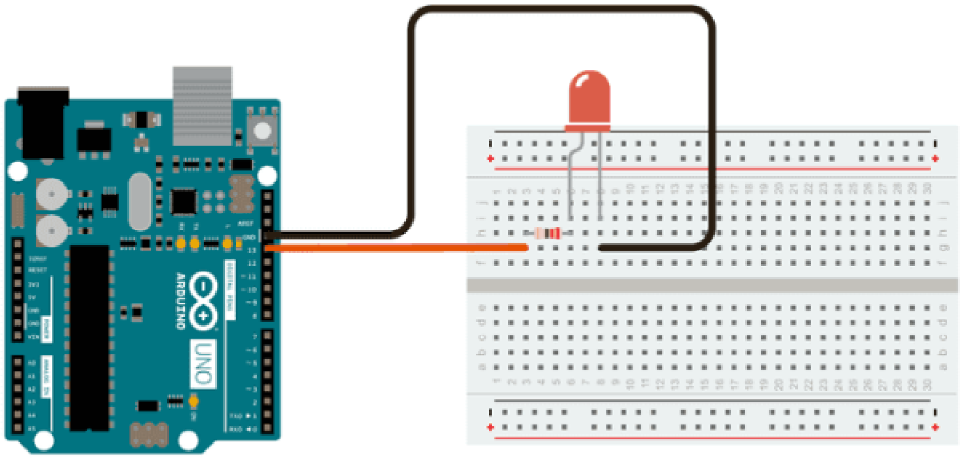
The buttons shown below are Verify and Upload. Verify makes sure your code has no errors and upload will push the code from the IDE to the board.



Open the Arduino IDE and copy-paste the code below.

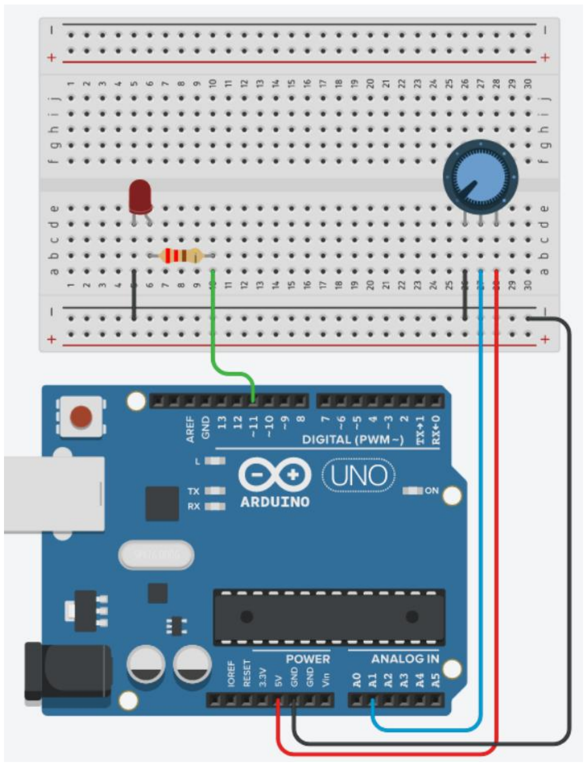
```
const int ledPin = 13; //can be changed to the pin that you attached your LED to
void setup() {
  // put your setup code here, to run once:
  pinMode(LED_BUILTIN, OUTPUT); //LED_BUILTIN refers to the built in LED on the
Arduino Uno Board which is pin 13
            //this can be changed to the ledPin variable initialized earlier, try
switching to this if you want to see how it works
}

void loop() {
  // put your main code here, to run repeatedly until you power your Arduino off:
  digitalWrite(LED_BUILTIN, HIGH); //this sets the output to turning the LED on
(sending 5V to the LED)
  delay(1000); // the number for the delay is in milliseconds, so this would turn
on the LED for one second
  digitalWrite(LED_BUILTIN, LOW); //this sets the output to turning the LED off
  delay(1000);
}
```
Set up the circuit, as shown in the image below (the resistor is 220 ohms and the pin is 13). You need to have the resistor so that the LED does not burn out by limiting the current flowing through the LED. Once correctly built, upload the code to the board. The LED should turn on for one second and turn off for one second. This ensures that your board and IDE are working properly.



Project 1: Potentiometer (Analog Input)

Set up the circuit using the schematic below (keep the 220 ohms resistor, but switch the LED pin to pin 11 and potentiometer pin at A1).



This is the code used for this project.

```
const int ledPin = 11;
const int potentiometer = A1;
void setup() {
  // put your setup code here, to run once:
  pinMode(ledPin, OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  int potentiometerValue = analogRead(potentiometer); //this value will be from
0-1023, a 10-bit number
  int brightness = map(potentiometerValue, 0, 1023, 0, 255); //to convert the 10-
bit number to a byte number
  analogWrite(ledPin, brightness); //this takes byte values or a number in range
of 0-255
}
```

Essentially, the code takes an analog input from the potentiometer dial as a 10-bit number (range of 0-1023). Then, it maps this number to the range of 0-255 so that we can use the potentiometer value in analogWrite(). This analogWrite() function allows us to control the brightness of the LED attached to this circuit.

To display the analogRead() value of the potentiometer on Arduino's Serial Monitor, add this line of code to the setup section.

```
  Serial.begin(9600);
```

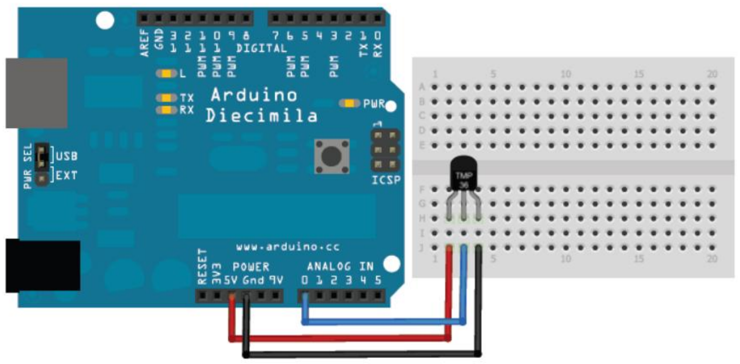
Add these lines to the loop portion, so that the printed values get updated as the potentiometer is turned. NOTE: The difference between print and println is that println will tell the monitor to start the next print on a new line. This is to make the monitor easier to read.

```
  Serial.print("Potentiometer Value: ");
  Serial.println(potentiometerValue);
  delay(100);
```

To open the Serial Monitor, go to Tools and select Serial Monitor. Make sure the Serial Monitor baud rate matches the number (9600) set in the Serial.begin() function in the bottom right of the Serial Monitor window. Open the Serial Plotter, also located under Tools. You should be able to see a live graph of the potentiometer changing.

Project 2: Temperature Sensor (TMP36)

Use the following schematic to connect the board and temperature sensor together.



This is the code used for this project.

```
const int tempPin = A0;
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
}

void loop() {
  // put your main code here, to run repeatedly:
  int sensorValue = analogRead(tempPin); //output in range of 0-1023
  float voltage = sensorValue * (5.0 / 1023.0); //converting to volts, it is 5.0
because we are using the 5V pin on the Arduino board, if using 3.3V then change
```

```
  float celsius = (voltage - 0.5) * 100.0; //converts from 10 mV per degree with
a 500 mV offset
  Serial.print("Temperature in Celsius = ");
  Serial.println(celsius);

  delay(1000);
}
```

To convert the 10-bit analog reading to a temperature, first the value has to be converted to volts. This voltage value is then placed into this formula: Centigrade temperature = [(analog voltage in V) – 0.5] * 100. Open the Serial Monitor and Serial Plotter to see the output of the sensor. Try blowing the sensor to cool it down or pinching it to heat it up.

Bonus: Convert the Celsius output to Fahrenheit and display on the Serial Monitor

Project 3: Pulse Oximeter

Open the Library Manager tab on the left-hand side panel or through the Tools dropdown. Search for MAX30105 and install the 1.1.2 version of this library.

Connect the MAX30102 to the Arduino Uno by using the following table.

MAX30102	**Arduino Uno**
VIN	5V
GND	GND
SDA	A4
SCL	A5
INT	Not connected

Find "Example4_HeartBeat_Plotter" within File → Examples → SparkFun MAX3010x Pulse and Proximity Sensor Library. This example code plots your heart beat by detecting changes in IR reflectance. Open the Serial Plotter and Serial Monitor to see your heartbeat. Make sure that the Serial Monitor baud is set to 115200. For more consistency, tape your finger to the MAX30102 board. Check to see if it lines up with your pulse on your neck, it should mimic very closely (maybe with a little delay).

Find "Example5_HeartRate" within File → Examples → SparkFun MAX3010x Pulse and Proximity Sensor Library. This example code outputs instantaneous heart rate and your BPM. Place your finger on the sensor and wait for a few seconds for readings that look accurate to appear on the Serial Monitor. Make sure that the Serial Monitor baud is set to 115200. Open the Serial Plotter and see the IR waveform which is your heartbeat.

Bonus: Take a look through https://learn.sparkfun.com/tutorials/max30105-particle-and-pulse-ox-sensor-hookup-guide/all to see all the different types of applications of the MAX30105 board.

Final Project: Multi-Sensor System

For the final project, try to combine the potentiometer, TMP36, MAX30105 and an LED to create an “integrated health monitor system.” Make it so that it can measure heart rate and report temperature. The potentiometer should be used to set a temperature threshold, which should be displayed to the Serial Monitor. The LED should blink slowly normally and faster if the heart rate is above 100 or the temperature is above the threshold. Both heart rate and temperature should come up on the Serial Monitor.