Week 3 Lab Activities:
Microcontrollers: Arduino Outputs

Preparation:

1. Visit the Arduino website and familiarize yourself with the Arduino Microcontroller's functionality and hardware specs.
2. Download the Arduino software on your laptop from the Arduino website.

Software setup:

When the software is downloaded, connect your Arduino board to the computer.

- **Select your board:** Open the software. Go to the top bar and select Tools > Board > Arduino uno
- **Select the Serial port that the board is connected to:**
  - Windows: Tools > Port > COM###
  - Mac: Tools > Port > /dev/tty.**usbserial**.##### (Look for the port that says usbserial)

3. Reading:
   a. Review in Programming Interactivity by Joshua Noble: Chapter 4 (p. 91-128)
   *For this week focus on pages 91-100, 102-107, 115-122, 126-128.
   b. Review in Physical Computing: Digital Output (p.87 – 89) and Analog Output (p.102 – 104)

Exercises:

What are we going to use in this workshop?

- **Solderless breadboard**
  !!! When you start to put components on your breadboard, avoid adding, removing, or changing components on a breadboard whenever the board is powered. You risk shocking yourself and damaging your components.
- **Arduino board**
- **USB cable**
- **LEDs**
- **Wires**
- **Resistors**

Circuit 1: Digital out

Make 2 LEDs blink so that when one is on the other is off. Connect each LED with a 390 Ohm resistor. Last time we used the 5V output from the Arduino board to power our electronics. The 5V output gives constant voltage. Now we want to program the behavior of our LEDs. Think about where the LED will get power from in this circuit!

Use this command:

```
digitalWrite(pinName, KEYWORD); // pinName is the variable name you choose and
KEYWORD is either HIGH or LOW
```

Task: Build the circuit as described above.
Task: Write the code that allows the LEDs to behave as we want.

Circuit 2: Analog (PWM) out

Make 2 LEDs fluctuate in brightness through dimming. The two LEDs should be in sync so that when one LED is lit, the other is completely dark. Always put a 390 Ohm resistor in series with your LEDs! Think about which pins can give analog output.

Use this command:

```
analogWrite(pinName, value); //pinName is a variable name that you choose. Think about the range of 'value', what's the value to make the LED light up completely and what's the value to make the LED be completely dark?
```

Task: Build the circuit as described above.
Task: Write the code that allows the LEDs to behave as we want.

Optional: Circuit 3: Serial Communication Output

Use keystrokes to dim/brighten an LED (Uses Serial Communication)

Use these commands:

```
Serial.begin(9600); // starts serial communication at 9600 baudrate

if ((Serial.available() > 0) { } // checks if data has been sent from the computer

Serial.read(); // read the most recent byte

Optional: Serial.print(); or Serial.println();
```

Task: Build a circuit of one LED connected to a digital pin.
Task: Write code which allows you to turn the LED on by sending 'H' to the serial monitor and turn the LED off by sending 'L' to the serial monitor.

Serial data reference:

- To start serial communication, you must open the serial port at a specific baud rate:

  Serial.begin(9600);

- To send binary data to the serial port, you use the **write()** function:

  Serial.write(data); or Serial.Writeln(data);

  *Adding '**ln**' prints each data point on a new line in the serial monitor*

- To send the characters representing the digits of a number, or a string, use the **print();** function instead.

  Serial.print(data); or Serial.println("string");

- To read data from the serial port that has been sent from an application to the Arduino you use the **read();** function:

  Serial.read();

  This will return either an integer, the first byte of serial data available, or a -1 if there is no serial data to read.

The following code will write the number 0 -255 in succession to the serial port:

```
int currentValue = 0;                          // variable to hold the analog value


void setup() {
        Serial.begin(9600);                    // open the serial port at 9600 bps:
}
void loop() {
        for(int i=0; i<256; i++){
                currentValue=i;
                Serial.println(currentValue, DEC);      //print number between 0-255
                delay(500);                             //wait ½ sec before taking next serial reading
        }
        currentValue = 0;
}
```

Notes on sending Serial Data:

From: http://itp.nyu.edu/physcomp/Labs/Serial