The prelab portion of this lab has to be completed at home and presented to a TA at the BEGINNING of your lab section. The prelab may contain written assignments and/or assignments to implement on your own Arduino kit. Before starting each lab, present your work papers and/or your working implementations to a TA and get their signature and keep this sheet for your records. Prelabs are INDIVIDUAL assignments. The lab portions may be worked on in groups of two.

Lab #5 – Prelab/Home Portion

Accelerometer is a MEMS device (Microelectromechanical Systems) that measures the instantaneous acceleration or rate of change of velocity for bodies. For example an accelerometer which is resting on a surface will measure the earth gravity acceleration which is 9.81 m/s² or 32.19 ft/ s². That means when an accelerometer free falls towards the earth it is going to measure zero. Most of modern accelerometer have 3 measuring axes (X, Y and Z) in a single chip. For more illustration see the below video about accelerometers and how they work:

https://www.youtube.com/watch?v=i2U49usFo10

IMPORTANT WARNING: THE ACCELEROMETER DOES NOT SUPPORT FIVE VOLT POWER! ATTACHING THE V_{CC} PIN TO FIVE VOLTS MAY CAUSE IRREPARABLE DAMAGE TO THE ACCELEROMETER WHICH YOU ARE RESPONSIBLE FOR.

1) Acknowledge that you have read and understood the warning concerning damage to your accelerometer.

2) Connect your accelerometer to your Arduino according to the tables below:

Table 1

MPU-6050 / GY-521	Arduino
V_{CC}	3.3V
GND	GND
SCL	A5
SDA	A4

Table 2

ADXL335	Arduino
V_{CC}	3.3V
GND	GND
X	A0
Y	A1
Z	A2

3) If you are using the ADXL335 add this line to your `setup()` function:
`analogReference(EXTERNAL);`
And connect your Arduino's A_{REF} pin to the 3.3V pin, this will change the analog reference from 5.0V to the same 3.3V used by the accelerometer and should give you more accurate results. Make sure that you don't accidentally connect the 3.3V pin to GND, doing so will damage your Arduino.

4) Write a program that prints the X, Y, and Z accelerations to the serial monitor every 500ms (See programming hints and codes in pages 2 - 4)

5) Complete the following table

Table 3

Axis	Raw Value
X+ (X = -1g)	
X- (X = +1g)	
Y+ (Y = -1g)	
Y- (Y = +1g)	
Z+ (Z = -1g)	
Z- (Z = +1g)	

MPU-6050/GY-521 and ADXL335 Accelerometers

MPU-6050/GY-521 accelerometers are more advanced than the ADXL335 accelerometers, so instead of reading an analog voltage and converting that value to an acceleration, it uses a serial communication to get the values directly. It's important to note that the values may be very different from the values provided by the ADXL335. The connections for the MPU-6050 and GY-521 are included below.

Programming:

The following codes can be used to help complete the prelab and the lab. Calculations for the scale and modifying the code will still need to be done by you to perform the desired way for this lab.

Code for MPU-6050/GY-521:

The **`xAccel`**, **`yAccel`**, and **`zAccel`** variables contain the acceleration values you will need for this lab exercise – the other values contain the values from the thermometer and gyroscope, which are not used in this lab.
Acceleration in gees can be calculated by the formula $Acceleration\ (g) = \frac{Raw\ Value}{16{,}384}$
This is because the sensitivity for the accelerometer for MPU-6050/GY-521 is 16,384 with the default range of +/- 2g.

```
#include <Wire.h>
// i2c address of the mpu-6050
const int MPU_ADDR = 0x68;
// variables
int16_t xAccel, yAccel, zAccel, temp, xGyro, yGyro, zGyro;

void setup()
{
    // initialize the i2c library
    Wire.begin();
    // begin communication with the mpu-6050
    Wire.beginTransmission(MPU_ADDR);
```

```
        // write to the power management register
        Wire.write(0x6B);
        // wake up the mpu-6050
        Wire.write(0);
        // end communication with the mpu-6050
        Wire.endTransmission(true);
        // initialize the serial port
        Serial.begin(9600);
    }
    void loop()
    {
        // read the mpu data
        readMpuData();
        /*
           * put your code for the digital letter scale here
           */
        // delay
        delay(500);
    }

    void readMpuData()
    {
        // begin communication with the mpu-6050
        Wire.beginTransmission(MPU_ADDR);
        // write to the first data register
        Wire.write(0x3B);
        // end communication
        Wire.endTransmission(false);
        // request fourteen register reads from the mpu-6050
        Wire.requestFrom(MPU_ADDR, 14, true);
        // read registers 0x3B (ACCEL_XOUT_H) and 0x3C (ACCEL_XOUT_L)
        xAccel  = Wire.read() << 8 | Wire.read();
        // read registers 0x3D (ACCEL_YOUT_H) and 0x3E (ACCEL_YOUT_L)
        yAccel  = Wire.read() << 8 | Wire.read();
        // read registers 0x3F (ACCEL_ZOUT_H) and 0x40 (ACCEL_ZOUT_L)
        zAccel  = Wire.read() << 8 | Wire.read();
        // read registers 0x41 (TEMP_OUT_H) and 0x42 (TEMP_OUT_L)
        temp   = Wire.read() << 8 | Wire.read();
        // convert the temperature to celsius
        temp   = ((temp / 340.00) + 36.53);
        // read registers 0x43 (GYRO_XOUT_H) and 0x44 (GYRO_XOUT_L)
        xGyro = Wire.read() << 8 | Wire.read();
        // read registers 0x45 (GYRO_YOUT_H) and 0x46 (GYRO_YOUT_L)
        yGyro = Wire.read() << 8 | Wire.read();
        // read registers 0x47 (GYRO_ZOUT_H) and 0x48 (GYRO_ZOUT_L)
        zGyro = Wire.read() << 8 | Wire.read();
        // print out the results
        Serial.print("AX = "); Serial.print(xAccel); // X Raw Value
        Serial.print(" | AY = "); Serial.print(yAccel); // Y Raw Value
        Serial.print(" | AZ = "); Serial.print(zAccel); // Z Raw Value
        Serial.print(" |  C = "); Serial.print(temp);
        Serial.print(" | GX = "); Serial.print(xGyro);
        Serial.print(" | GY = "); Serial.print(yGyro);
        Serial.print(" | GZ = "); Serial.println(zGyro);
    }
```

Code for ADXL 335:

The `xAccel`, `yAccel`, and `zAccel` variables contain the acceleration values you will need for this lab exercise. Be noted that this accelerometer produces a fair amount of noise so taking an average of multiple samples could help with filtering this noise.

```
double xAccel = 0.00;
double yAccel = 0.00;
double zAccel = 0.00;
void setup()
{
// for you to do...
}
void loop()
{
// update the acceleration values
readAcceleration();
// for you to do...
}
void readAcceleration()
{
// an index variable
int index = 0;
// reset the values
xAccel = 0;
yAccel = 0;
zAccel = 0;
// loop through the values
for (index = 0; index < 16; index++)
{
// read a sample
xAccel += analogRead(A0);
yAccel += analogRead(A1);
zAccel += analogRead(A2);
delay(10);
}
// calculate the average
xAccel /= (16.00);
yAccel /= (16.00);
zAccel /= (16.00);
}
```

Lab #5 Assignment

Part 1.Liquid Crystal Display

You will be using the LCD module in your kit to display the deflection of the scale and the weight applied. If your LCD module has not been soldered, the TAs in your lab will solder the header pins onto the display for you.

The connections between the Arduino and the LCD are given in the following table:

LCD	ARDUINO	PURPOSE
RS	DI/O pin 12	Register select
EN	DI/O pin 11	Enable
RW	GND	Read/Write (selected as Write)
D4	DI/O pin 5	Data
D5	DI/O pin 4	Data
D6	DI/O pin 3	Data
D7	DI/O pin 2	Data
V_{DD} or V_{CC}	5V	Power
V_{SS} or GND	GND	Ground

You will also need to add a 10kΩ potentiometer and a 220Ω resistor to control the contrast and power the backlight; respectively.

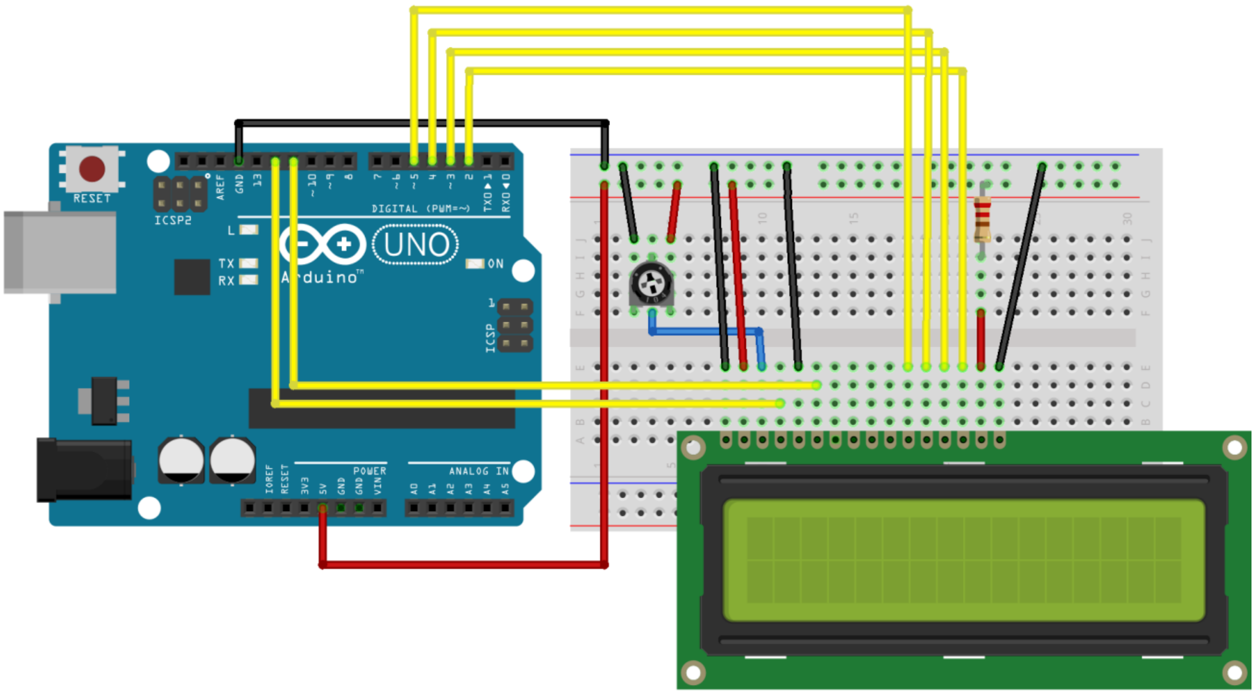

Figure 2: Diagram of the connections between the LCD and the Arduino UNO.

Hint: You will need to use the Arduino's **LiquidCrystal** library, examples can be found under the (**File → Examples → LiquidCrystal**)

Print your name on the LCD. Demonstrate this to the TA to get a signature for Part 1.

Note: The LCD will be used in Part 2 as well so do not put away your LCD.

Part 2. In this exercise, you will construct and calibrate a digital scale, suitable for measuring the weight of a letter in order to determine postage. The scale, shown in **Figure 1**, is made up of an L-shaped piece of poster board (with a hole at point A to hang the scale from), a counterweight W_2 (you will use known masses for counterweights), the ADXL335 or MPU-6050 or GY-521 accelerometer, and the weight to be measured W_1. The L-shaped piece of poster board and weights will be provided in the lab.

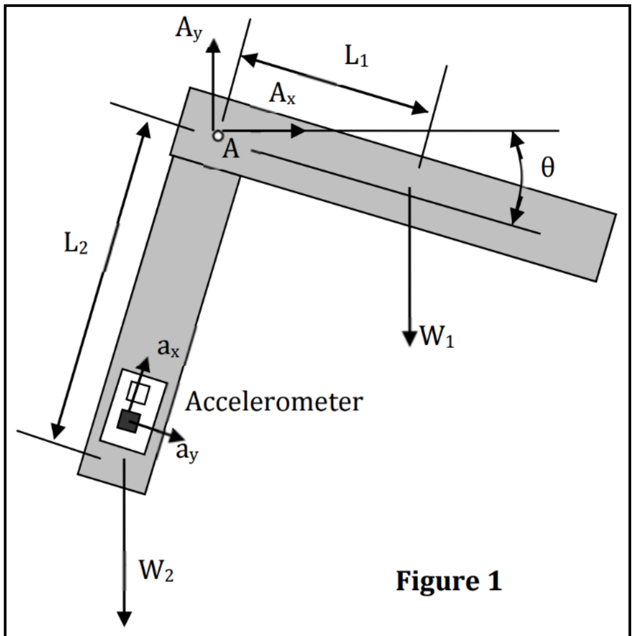

Figure 1

Calibrating the scale

To calibrate each axis of the accelerometer, use the results from the table 3 in prelab to solve the following equations:

$$R_0 = \frac{R_+ + R_-}{2}; \qquad R_R = \frac{R_+ - R_-}{2}$$

R_0 is the actual 0g point and R_R is the actual range from 0g of each axis.
R_+ is the +1g Raw Value for your axis and R_- is the -1g Raw Value.

To see how you can use the accelerometer to measure the angle of tilt, orient the x-axis to be vertical and the y and z-axes horizontal. Rotating the accelerometer about the z-axis (keeping the z-axis reading constant) results in a change in the x and y-axes readings. The change in the y-axis reading is proportional to the component of gravity acting in the y-direction. Assume that for this particular accelerometer the +1g reading is 614 and the -1g reading is 410, which gives $R_0 = 512$ and $R_R = 102$.

To calculate the angle, take the change in the y-direction $(a_y - R_0)$ and divide it by the value of R_R to find the tangent of the angle that the y-axis makes with the horizontal. Remember that for small angles (up to 15°) measured in radians, $\tan(\theta) \approx \sin(\theta) \approx \theta$.

The angle is therefore equal to:

$$\theta = \frac{(a_y - R_0)}{R_R}$$

a_y is your +1 g y-axis Raw Value.

To find the angle in degrees, multiply the angle by $\frac{180}{\pi}$:

$$degrees = 180 * \left(\frac{a_y - R_0}{R_R \pi}\right)$$

Step1: Hang the L-shaped poster board from point A so it can freely rotate. Fasten the accelerometer to the left portion of the L.

Step2: Hang a 10-gram weight from point W_1, and put enough weight on point W_2 to make the one leg of the L almost completely vertical and the other leg of the L almost horizontal. (Note: slowly increase the weight you are adding on. Do not add too much at one time. Why?)

Step3: Let the L hang freely (remove the 10-gram weight from W_1 and keep the weight at W_2) from point A and read the y-axis Raw Value, call the value $A_{y_{offset}}$.

Step4: Put a 20-gram weight on W_1. Record the new y-axis value as $A_{y_{20}}$.

Step5: Calculate the two unknowns (C and W_0) using equations 1 and 2.

$$20 \text{ grams} = \mathrm{C} * \frac{\mathrm{A}_{y_{20}} - R_0}{R_R} - W_0 \qquad [eq1]$$

$$W_0 = C * \frac{A_{y_{offset}} - R_0}{R_R} \qquad [eq2]$$

Now that we know C and W_0 we can measure the weight of an unknown mass by attaching it at W_1.

Step6: Plug equation 3 into your program to calculate the weight of an unknown mass.

$$W_1 = \left(C * \frac{a_y - R_0}{R_R}\right) - W_0 \qquad [eq3]$$

Step7: Attach an unknown mass (For Example: 20-grams, 30-grams, or 50-grams) to W_1 (do not remove the weights from W_2).

Step8: Display the y-axis acceleration A_y (in gees), the angle of deflection Θ, and the weight of the unknown at W_1 on the LCD module. While it isn't required, you may find it helpful to print the result of each calculation to the serial port while you're debugging your program. **TA's will sign off once you show them your results displayed on the LCD module.**