Girls into Electronics
Further Activities

This guide follows on from the Student Guide to make more advanced programs on the Grove Beginner Kit.

Contents

Introduction..... 2
Tutorial - Serial Communication and Talking to your Computer 2
Exercise 5 – The Accelerometer 5
Exercise 6 – Combining it all 6
About the UK Electronics Skills Foundation..... 10

Introduction

The introduction to the Grove Beginner Kit is in the Student Guide. It is assumed you have worked through that previous guide.

Tutorial - Serial Communication and Talking to your Computer

Serial communication works by converting information to a stream of bits, which are then sent between two devices over one or more wires, allowing interaction with more complex devices such as a laptop or sensors such as a digital accelerometer.

Serial protocol functions :-

Serial.begin(9600)- Initialises different peripherals and communication protocols. 9600 is the 'baud rate' of the connection. It defines how fast the data is to be sent.

Serial.print() – Sends the text in the parentheses to the computer.

Serial.println() – Sends the text in the parentheses to the computer and also starts a new line. To

test this functionality, type the below code into a new Duino project

```
void setup() {
  Serial.begin(9600);               // Begin the serial communication.
  Serial.println("Hello, World!");  // Send a message to the computer.
  //You can add more print statements here
}

void loop() {} // For now, our loop() function does nothing.
```

Plug in the Grove kit, then compile and upload the code.

To see the serial monitor click on "**MONITOR**" and change the baud to *9600 baud* as in Figure 1.

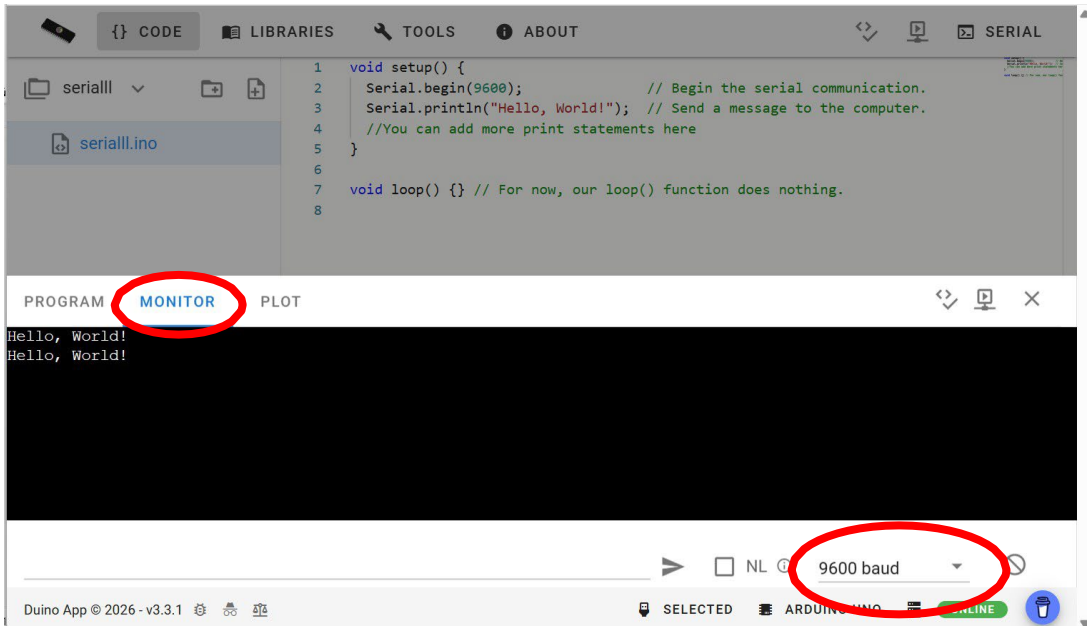


Figure 1 A preview of the Serial Monitor tool showing the "Hello, World!" message that was sent over USB from the Arduino

You should see the "Hello World" message. Try sending different messages using both Serial.print() and Serial.println().

Installing the UKESF Sixth Formers Library for Duino

This guide comes with an Arduino library, called *UKESF Sixth-Formers*. A library contains a set of functions that allow you to easily do more with an Arduino that is available out of the box. To do this click on the library icon, shown in Figure 2.

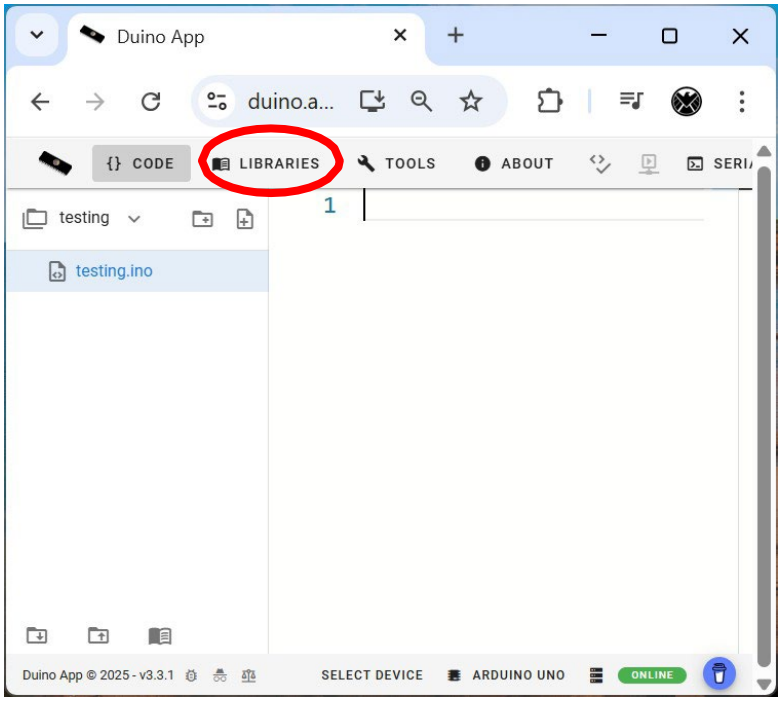


Figure 2 Duino App- Library icon

In the search box in the top right corner type in “ukesf”. This should come up with one library called “UKESF Sixth-Formers” as shown in Figure .

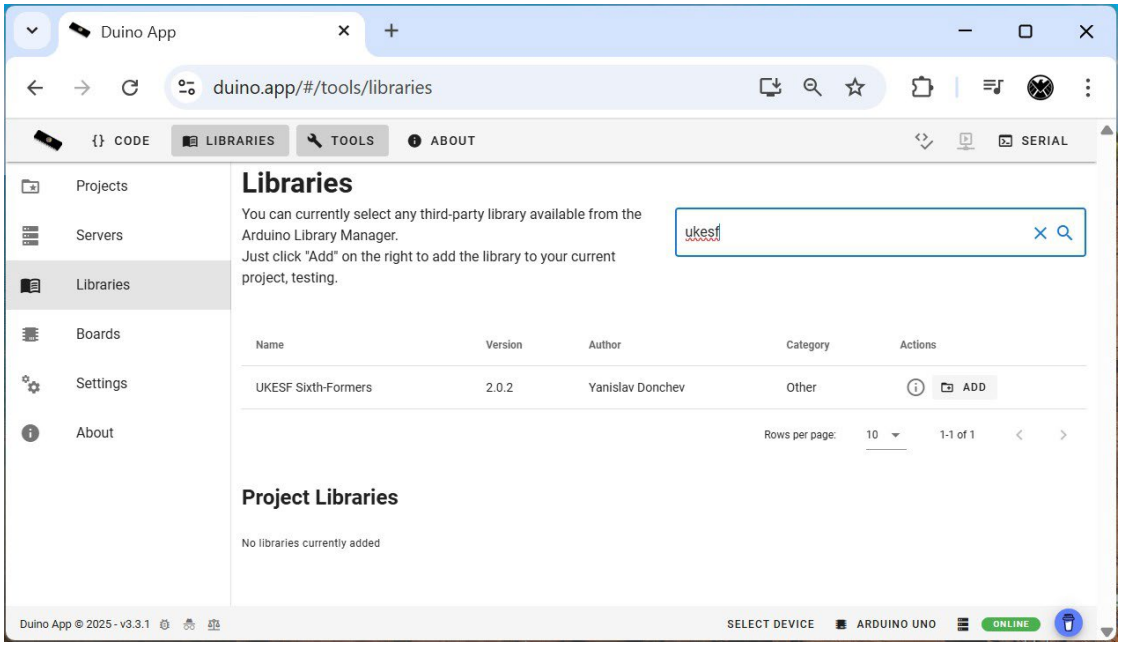


Figure 3 Duino App- Libraries tab

Click on the add button to add this library to your project. 

You will find a message pops up saying you are missing a dependency as shown in Figure 4.

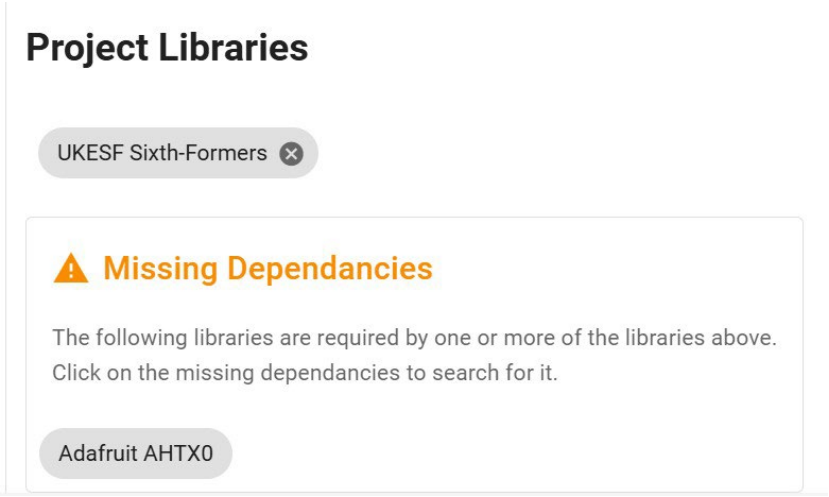


Figure 4 Duino App- Missing dependencies

Click on the name of the dependency in the grey oval (in 4this is “Adafruit AHTX0). This will make it appear above, and you can add it like you did with the UKESF library,

After this more missing dependencies will appear. Repeat the process of clicking on the name in the grey oval and then adding it. Eventually the “Missing Dependencies” box should go, and you should be left with the following libraires in your project as shown in Figure 5.

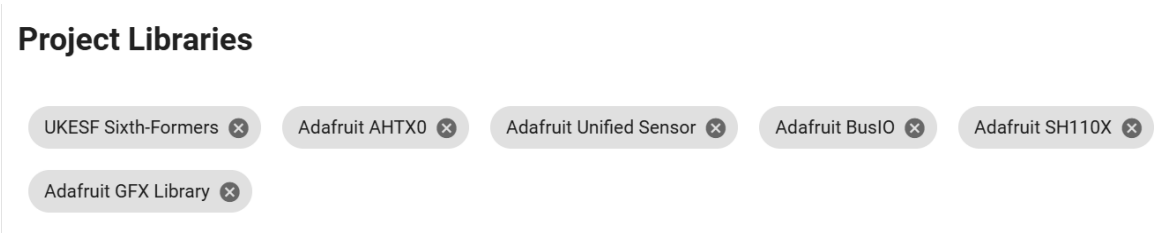


Figure 5 Duino App- Project Libraries

Click on the code icon again in the top bar to take you back to where you will write your code.

Exercise 5 – The Accelerometer

In this exercise you will learn to how to read data from a 3-axis accelerometer and plot it as a graph on the Arduino Serial Plotter. An accelerometer is a device that measures the rate of change of velocity of a body. For example, an accelerometer at rest on the surface of Earth will measure the acceleration due to Earth's gravity as 1 g (gravitational force equivalent) or approximately 9.81 m/s^2. Using the accelerometer on the Grove kit, we can measure acceleration in three axes.

Step 1

Before the setup()function, include the UKESF library (which will handle some of the accelerometer functions) and define an accelerometer variable called "myAccelerometer":

```
#include <UkesfSixthFormers.h> // Include the UKESF library of functions

Accelerometer myAccelerometer;  // Create an instance of the accelerometer
```

Step 2

Next, begin (or initialise) Serial communication and the Accelerometer:

```
void setup() {
  Serial.begin(9600);      // Begin the serial communication.
  myAccelerometer.begin();  // Begin the accelerometer.
}
```

Step 3

In the loop()function, read the accelerometer values and print them to the serial monitor. Note that floating point variables or float must be used because the acceleration is given as a decimal number.

```
void loop() {
  float x = myAccelerometer.readX(); // Read x-axis acceleration
  float y = myAccelerometer.readY(); // Read y-axis acceleration
  float z = myAccelerometer.readZ(); // Read z-axis acceleration
  Serial.print(x);
  Serial.print(" ");
  Serial.print(y);
  Serial.print(" ");
  Serial.println(z);
  delay(10); // Delay the program by 10ms for stability when looping
}
```

Step 4

There are two ways to view the data from the accelerometer. The first is using the serial monitor described above and the second is the *Serial Plotter*. You can access the by clicking on "**PLOT**" next to monitor in the output window. This is a useful tool that plots the data in graph form. Try tilting and turning your board to identify each of the three axes of the accelerometer.

Exercise 6 – Combining it all

In this exercise you will combine what you have learned in the previous exercises from the first student guide and this document into one program, implementing a tilt sensor with a warning light and sound. Specifically, the program will:

- Read data from the accelerometer,
- Calculate the pitch and roll (angles) of your board in degrees, based on that data,
- If the roll (or pitch) exceeds 20 degrees, light the LED and sound the buzzer.

The examples of code given below include only the comments of what the program should include. You will fill them in, using the syntax and pin numbers given **in the previous exercises.**

Step 1

Include the UKESF library to access the accelerometer and define the pin numbers and variables the program will use later. Fill in the missing lines of code before each comment below:

```
// Include the UKESF library of functions

// Define the name and pin number of the LED
// Define the name and pin number of the buzzer
// Frequency of tone in Hz
// Create an instance of the accelerometer
```

Step 2

Initialise the LED, buzzer, serial communication and accelerometer.

```
void setup() {
  // Initialize the ledPin as an output.
  // Initialize the buzzerPin as an output.
  // Begin the serial communication.
  // Begin the accelerometer.
}
```

Step 3

Implement the main program in the loop()function:

1) Read the accelerometer data
2) Calculate the roll and pitch
3) Light the LED and sound the buzzer IF the roll is > 20 degrees.

Approach each of these tasks in turn.

1) Read the accelerometer data

Look back to exercise 3 for the code used to read data from the accelerometer and print it to the serial monitor. Although the program does not require printing to Serial, it is a good idea to do so for debugging purposes, using the serial monitor after uploading our code to make sure the data from the accelerometer is as we would expect.

```
void loop() {
  // Read x-axis acceleration
  // Read y-axis acceleration
  // Read z-axis acceleration

  // Print the value to serial for debugging help
}
```

2) Calculating roll and pitch from accelerometer data.

Pitch, roll and yaw are rotational forces (illustrated in Figure 6) and deriving them as angles from accelerometer data is complicated. The links below give good explanations of how a 3-axis accelerometer can be used to derive pitch and roll.

- DF Robot tutorial with quick derivation: https://wiki.dfrobot.com/How_to_Use_a_Three-Axis_Accelerometer_for_Tilt_Sensing
- Application note from NXP with more rigorous derivation: https://www.nxp.com/files-static/sensors/doc/app_note/AN3461.pdf

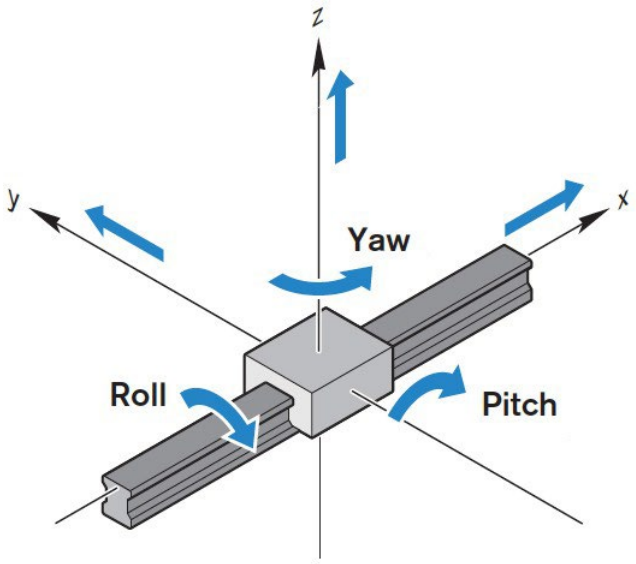


Figure 6 An illustration of pitch, roll and yaw of a linear system, in this case a rigid beam.

It is sensible to read through the information given in the provided links to get an understanding of on the derivation of roll and pitch angles. The final equations are given below and require the use of pi, which in Arduino is written as “M_PI”.

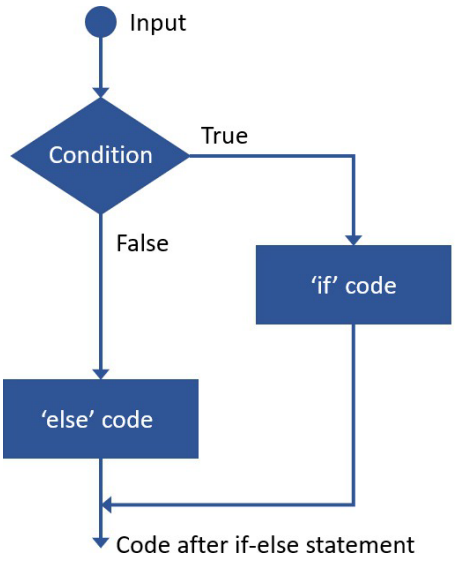
```
void loop() {
  // Read x-axis acceleration
  // Read y-axis acceleration
  // Read z-axis acceleration

  // Print the value to Serial for debugging help

  //Calculate roll and pitch based on accelerometer data
  //See provided links for derivation.
  float roll  = (atan2(-y, z)*180.0)/M_PI;
  float pitch = (atan2(x, sqrt(y*y + z*z))*180.0)/M_PI;
}
```

3) Light the LED and sound the buzzer if the roll is > 20 degrees.

To turn the LED on and sound the buzzer if the roll angle is > 20 degrees in either direction, the absolute value function abs() must be used. To do this, use a conditional if-else statement to implement the logical sequence illustrated by the flow diagram in Figure 7. The if-else statement tests an input variable against a condition (i.e., is the angle > 20 degrees?) and takes the True or False path depending on the result. When a condition is True, only the code inside the curly brackets enclosing the 'if' statement is executed and when it is False, the 'else' statement is executed.




```
if(condition is True){
 // Execute this code (condition is True)
}
else{
 // Execute this code (condition is False)
}
```

Note! The `condition` in the if statement can be any logical test, for example:

To check for negative temperatures, you could write:

```
if(temp < 0)
```

To check for an angle of 45 degrees, you could write:

```
if(angle == 45)
```

Figure 7 Flow diagram of an if-else statement in Arduino and the corresponding code. Depending on if the condition is True or False different code gets executed and the rest is skipped.

Complete the code template below, based on your previous work.

Experiment by changing your code to light the LED and sound the buzzer if the pitch is > 20 degrees

```
void loop() {
  // Read x-axis acceleration
  // Read y-axis acceleration
  // Read z-axis acceleration

  // Print the value to Serial for debugging help

  //Calculate Roll and Pitch based on accelerometer data
  //See provided links for derivation.
  float roll  = (atan2(-y, z)*180.0)/M_PI;
  float pitch = (atan2(x, sqrt(y*y + z*z))*180.0)/M_PI;

  //if-else statement to check if the roll is > 20 deg
  //Note the use of the absolute value abs()
  if(abs(roll) > 20){
    // Turn the LED on (High, 1, or 5V).
    // Play a tone of 200 Hz on the buzzer
  }
  else{
    // Turn the LED off (Low, 0, or 0V).
    // Turn the buzzer off.
    }

  delay(10); // Delay the program by 10ms for stability when looping
}
```

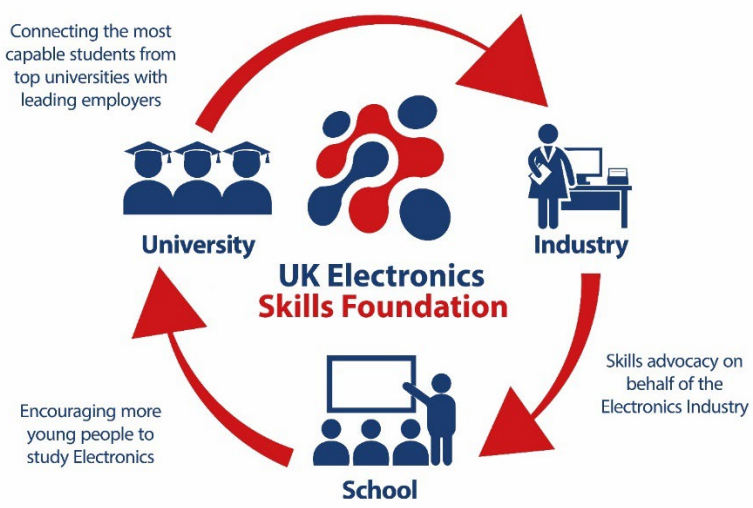
About the UK Electronics Skills Foundation

Through engagement with schools, universities and industry, it is our mission to encourage more young people to study Electronics and to pursue careers in the sector.

Electronics Engineers make a critical contribution to virtually all areas of life, from AI and digitalisation to green energy, healthcare, communications, manufacturing and more. The UK has a long heritage of technological innovation and has a worldclass Electronics sector. We are adept at creating and growing successful companies that design, make and sell products and services across all technological fields - and these companies will play an important role in solving some of the biggest challenges facing society today.
However, the demand for capable, employable Electronics Engineers and designers is currently outstripping supply. The only way we can sustainably support our industry to grow, is to increase the number of young people pursuing a career in the sector, and equipping them with Electronics skills for the future.
As an independent, UK based charity, we're working to address the skills gap in Electronics by:

- raising awareness,
- promoting interest in young people,
- supporting the development of those who choose Electronics, and
- building relationships to ensure a thriving sector.

Our mission can only be achieved by working collaboratively, and to date, we have worked with more than 100 employers from across the industry, 30 of the UK's leading universities, and over 900 schools.



"Moving beyond talk about the skills shortage to take positive action is what the UKESF is all about."
Stewart Edmondson, CEO, UKESF

Find out more: **ukesf.org**

Girls into Electronics
Further Activities - Code

Here are the code answers to the exercises in Further Activities document.

Exercise Solutions

Exercise 5 Solution:

```
#include <UkesfSixthFormers.h> // Include the UKESF library of functions
Accelerometer myAccelerometer;  // Create an instance of the accelerometer

void setup() {
  Serial.begin(9600);     // Begin the Serial communication.
  myAccelerometer.begin();  // Begin the Accelerometer.
}

void loop() {
  float x = myAccelerometer.readX(); // Read x-axis acceleration
  float y = myAccelerometer.readY(); // Read y-axis acceleration
  float z = myAccelerometer.readZ(); // Read z-axis acceleration
  Serial.print(x);
  Serial.print(" ");
  Serial.print(y);
  Serial.print(" ");
  Serial.println(z);
  delay(10); // Delay the program by 10ms for stability when looping
}
```

Exercise 6 Solution:

```
#include <UkesfSixthFormers.h> // Include the UKESF library of functions

const int ledPin = 4; //Define the name and pin number of the LED to blink const int buzzerPin = 5;
//Define the name and pin number of the buzzer const int toneFrequency = 200;  //Frequency of
tone in Hz
Accelerometer myAccelerometer;  // Create an instance of the accelerometer

void setup() {
   pinMode(ledPin, OUTPUT);  // Initialize the ledPin as an Output. pinMode(buzzerPin,
   OUTPUT);  // Initialize the buzzerPin as an Output. Serial.begin(9600);          // Begin the
   Serial communication. myAccelerometer.begin();  // Begin the Accelerometer.
}
void loop() {
   float x = myAccelerometer.readX();  // Read x-axis acceleration float y =
   myAccelerometer.readY();    // Read y-axis acceleration float z =
   myAccelerometer.readZ(); // Read z-axis acceleration Serial.print(x);
   Serial.print(""); Serial.print(y);
   Serial.print(""); Serial.println(z);
   //Calculate Roll and Pitch from the accelerometer data.
   //See here for rigorous derivation: https://www.nxp.com/files-
static/sensors/doc/app_note/AN3461.pdf
   //See here for quick derivation: https://theccontinuum.com/2012/09/24/arduino-
imu-pitch-roll-from-accelerometer/
   float roll  = (atan2(-y, z)*180.0)/M_PI;
   float pitch = (atan2(x, sqrt(y*y + z*z))*180.0)/M_PI;
```

Continues on next page..

```
//Use an if-else logical statement to check if the roll is > 20 deg
  // (note that we use the absolute value of the roll to deal with -ve
  //  values, i.e. tilting in the opposite direction)
  if(abs(roll) > 20){
    digitalWrite(ledPin, HIGH);  // Turn the LED on (High, 1, or 5V).
    tone(buzzerPin, toneFrequency);  // Play a tone of 200 Hz on the buzzer
  }
  else{
    digitalWrite(ledPin, LOW);  // Turn the LED off (Low, 0, or 0V).
    noTone(buzzerPin);           // Turn the buzzer off.
    }
  delay(10); // Delay the program by 10ms for stability
}
```