```
//////////////////////////////////////////////
/*
 CITATION
 Code inspired by Arduino Starter Kit example
 Project 6  - Optical Theremin
 by Scott Fitzgerald
 http://arduino.cc/starterKit
 This example code is part of the public domain
*/
//////////////////////////////////////////////
/* HELEN"S PSEUDO CODE

*** SET-UP ***
Set up & establish arduino connection with all components
    - mouse eye 1, green LED, OUTPUT for low tones
    - mouse eye 2, yellow LED, OUTPUT for high tones
    - calibrator, LED, OUTPUT will display light during millis calibration count
    - photosensor, analog INPUT, reads light
    - piezo, digital OUPUT, sound

Set up variables to hold high and low
  (I found this brilliant calibration method in Scott's code. Previously, I had attempted
  sensorHigh and sensorLow based on values found in serial printouts.)

    - initialize sensorValue, which is a variable for storing the current photosensor valu

    - initialize sensorLow
        SensorLow is set to a value near the upper limit of the photosensor serial reads.
        starting the variable at a high value we can be sure to overwrite it with an appr
        value during the calibration loop.
         After calibration, this variable will hold the lower light limit for our map func

    - initialize sensorHigh to 0
         By starting at 0 we can be sure to overwrite it with an appropriate high value d
        calibration loop.
         After calibration, this variable will hold the upper light limit for our map func

    - initialize eyeDiff, which is a variable that stores the difference b/t sensorHigh &
      eyeDiff/2 will be used to as a switch b/t the mouse's green (low tone range) and yel

Run calibration for 5s using mills()
  - reset sensorHigh value
     Each time a value is recorded that is higher than the current state of sensorHigh,
     rewrite sensorHigh with the new high value.

  - reset sensorLow value
       Each time a value is recorded that is lower than the current state of sensorLow,
       rewrite sensorLow with the new low value.

  - turn off calibrator LED to signal the end of the mills calibration period
```

```
  - calculate eyeDiff
      (sensorHigh-sensorLow)/2

Check the current values of sensorHigh, sensorLow, & eyeDiff via Serial.println()


*** LOOP ***
Read the input from the photosensor and assign it to sensorValue

Output light based on sensorValue data
- green mouse eye will light when low tones are read
- yellow mouse eye will light when high tones are read

Translate the sensorValue data to tones values using map()
- store the mapped tone values in a new variable called pitch

Output sounds to piezo based on pitch values


END PSEUDO CODE */
////////////////////////////////////////////


////////////////////////////////////////////
// BEGIN PROGRAM
////////////////////////////////////////////

//declaring my globals varibles
int calibrator = 13;//LED
int sensor = 0;//photosensor
int green = 5;//digi LED on mouse
int yellow = 3;//digi LED on mouse
int piezo = 8;//digi piezo speaker

int  sensorValue;// variable photosensor value

// Variable set purposely high to be reset in the sensor calibration loop.
// After calibration, this variable will be the lower light limit.
int sensorLow = 1023;

// Variable set purposely low to be reset in the sensor calibration loop.
// After calibration, this variable will be our upper light limit.
int sensorHigh = 0;

// variable to store the difference b/t sensorHigh & sensorLow for use
//as a switch for the mouse's eyes
int   eyeDiff;

////////////////////////////////////////////
void setup() {
  Serial.begin(9600); //begin serial output
```

```
  // Make the calibration LED pin an output and turn it on
  pinMode(calibrator, OUTPUT); // LED marker for millis-based calibration time
  digitalWrite(calibrator, HIGH); //turn calibration LED on

  pinMode(sensor, INPUT); //photosensor

  pinMode(green, OUTPUT); //green LED on mouse
  digitalWrite(green, LOW); //starts in OFF state

  pinMode(yellow, OUTPUT); //yellow LED on mouse
  digitalWrite(yellow, LOW); //starts in OFF state

 // calibrate for the first 5s on Run
  while (millis() < 5000) {
   // record the maximum sensor value
     sensorValue =analogRead(sensor);
    if (sensorValue > sensorHigh) {
      sensorHigh = sensorValue;
    }

    // record the minimum sensor value
    if (sensorValue < sensorLow) {
      sensorLow = sensorValue;
    }
  }
    // turn the LED off at the end of the millis loop
   // which will signal the end of the calibration period
    digitalWrite(calibrator, LOW);
    Serial.println(sensorHigh);
    Serial.println(sensorLow);

  eyeDiff = (sensorHigh-sensorLow)/2;
  Serial.println(eyeDiff);

} //END SETUP

////////////////////////////////////////
void loop() {
  //read the input from A0 and store it in sensorValue
   sensorValue =analogRead(sensor);

  // map the sensor values to a wide range of pitches
  int pitch =map(sensorValue, sensorLow, sensorHigh, 100, 4000);

 //Based on serial readings the average difference between my sensorHigh and sensorLow see
  if(sensorValue > sensorHigh-eyeDiff) {
    digitalWrite(yellow, HIGH);
    digitalWrite(green, LOW);
  }

  else if (sensorValue < sensorHigh-eyeDiff){
    digitalWrite(yellow, LOW);
```

```
    digitalWrite(green, HIGH);
  }

  // play the tone for 20 ms
  tone(piezo, pitch, 20);

  //the aesthetic quality of the output (tones and lights) was better when I did this
  delay(50);

  //this was necessary for debugging
  Serial.println(sensorValue);

} //END LOOP
```