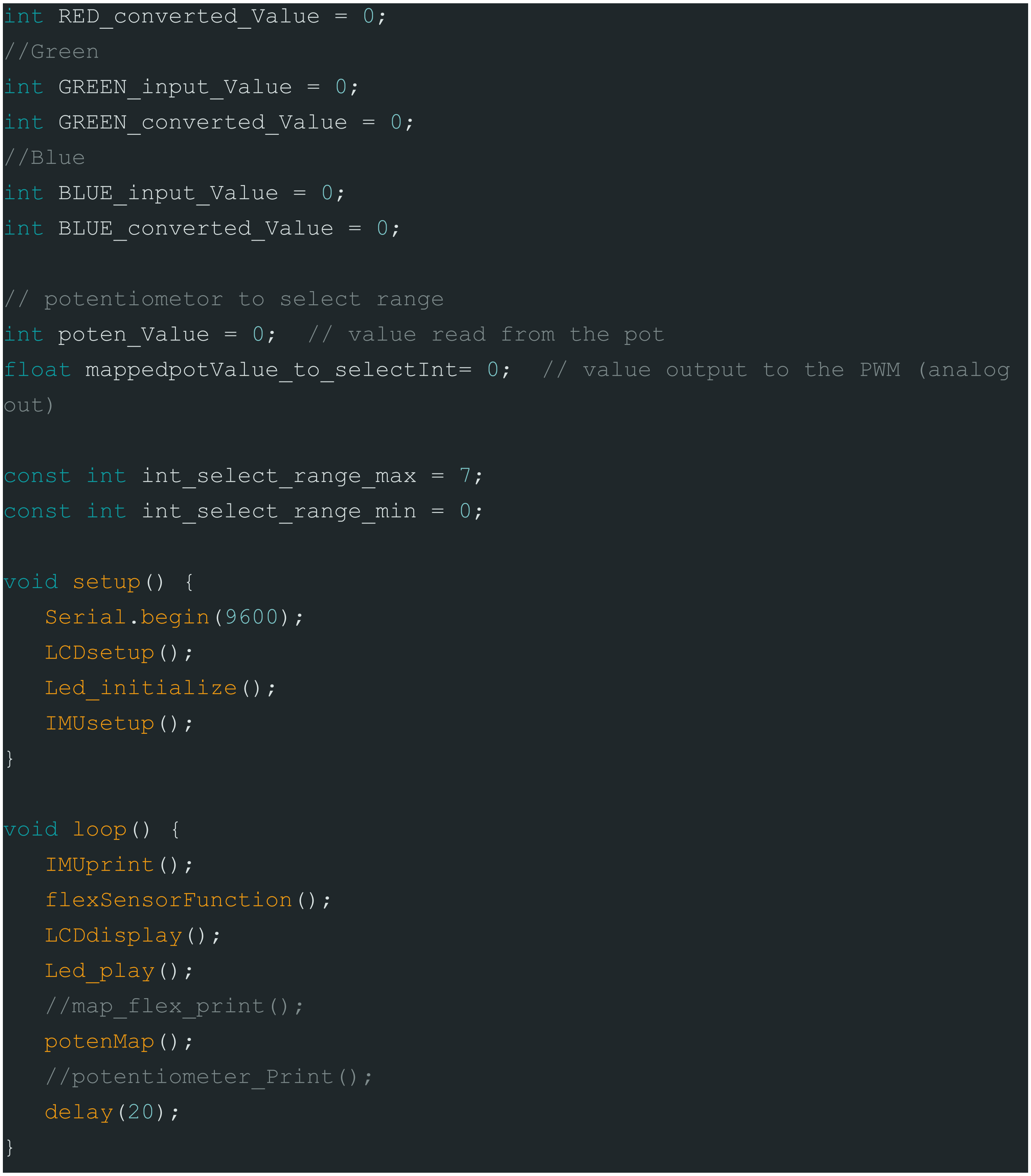
Main code

```
//#include "FlexLibrary.h"
#include <Adafruit_NeoPixel.h>
#ifdef __AVR__
#include <avr/power.h> // Required for 16 MHz Adafruit Trinket
#endif
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_NeoPixel.h>
#include <Wire.h>

Adafruit_MPU6050 mpu;

//red...........
const int RED_flexSensorPin = A1;
//green...........
const int GREEN_flexSensorPin = A2;
//blue...........
const int BLUE_flexSensorPin = A3;

const int FlexSensor_rangeLOW = 0;
const int FlexSensor_rangeHIGH = 4095;

const int colorRangeMin = 0;
const int colorRangeMax = 70;

#define Pinky_ThumbPIN        32
#include <Adafruit_NeoPixel.h>
#ifdef __AVR__
 #include <avr/power.h> // Required for 16 MHz Adafruit Trinket
#endif
#define Pinky_ThumbNUMPIXELS 1
Adafruit_NeoPixel Pinky_Thumb_pixel(Pinky_ThumbNUMPIXELS, Pinky_ThumbPIN,
NEO_GRB + NEO_KHZ800);

//red
int RED_input_Value = 0;
```



```
int RED_converted_Value = 0;
//Green
int GREEN_input_Value = 0;
int GREEN_converted_Value = 0;
//Blue
int BLUE_input_Value = 0;
int BLUE_converted_Value = 0;

// potentiometor to select range
int poten_Value = 0;  // value read from the pot
float mappedpotValue_to_selectInt= 0;  // value output to the PWM (analog out)

const int int_select_range_max = 7;
const int int_select_range_min = 0;

void setup() {
   Serial.begin(9600);
   LCDsetup();
   Led_initialize();
   IMUsetup();
}

void loop() {
   IMUprint();
   flexSensorFunction();
   LCDdisplay();
   Led_play();
   //map_flex_print();
   potenMap();
   //potentiometer_Print();
   delay(20);
}
```

Pressure Sensor to LED Output color Values

```
void flexSensorFunction() {
  REDmapFunction();
  GREENmapFunction();
  BLUEmapFunction();
}
void REDmapFunction(){
  // read the analog in value:
  RED_input_Value = analogRead(RED_flexSensorPin);
  // map it to the range of the analog out:
  RED_converted_Value = abs(map(RED_input_Value, FlexSensor_rangeLOW,
FlexSensor_rangeHIGH, colorRangeMin, colorRangeMax));
}
void GREENmapFunction(){
  GREEN_input_Value = analogRead(GREEN_flexSensorPin);
  // map it to the range of the analog out:
  GREEN_converted_Value = abs(map(GREEN_input_Value, FlexSensor_rangeLOW,
FlexSensor_rangeHIGH, colorRangeMin, colorRangeMax));
}
void BLUEmapFunction(){
  // read the analog in value:
  BLUE_input_Value = analogRead(BLUE_flexSensorPin);
  // map it to the range of the analog out:
  BLUE_converted_Value = abs(map(BLUE_input_Value, FlexSensor_rangeLOW,
FlexSensor_rangeHIGH, colorRangeMin, colorRangeMax));
}
```

LED Input Color mix Function

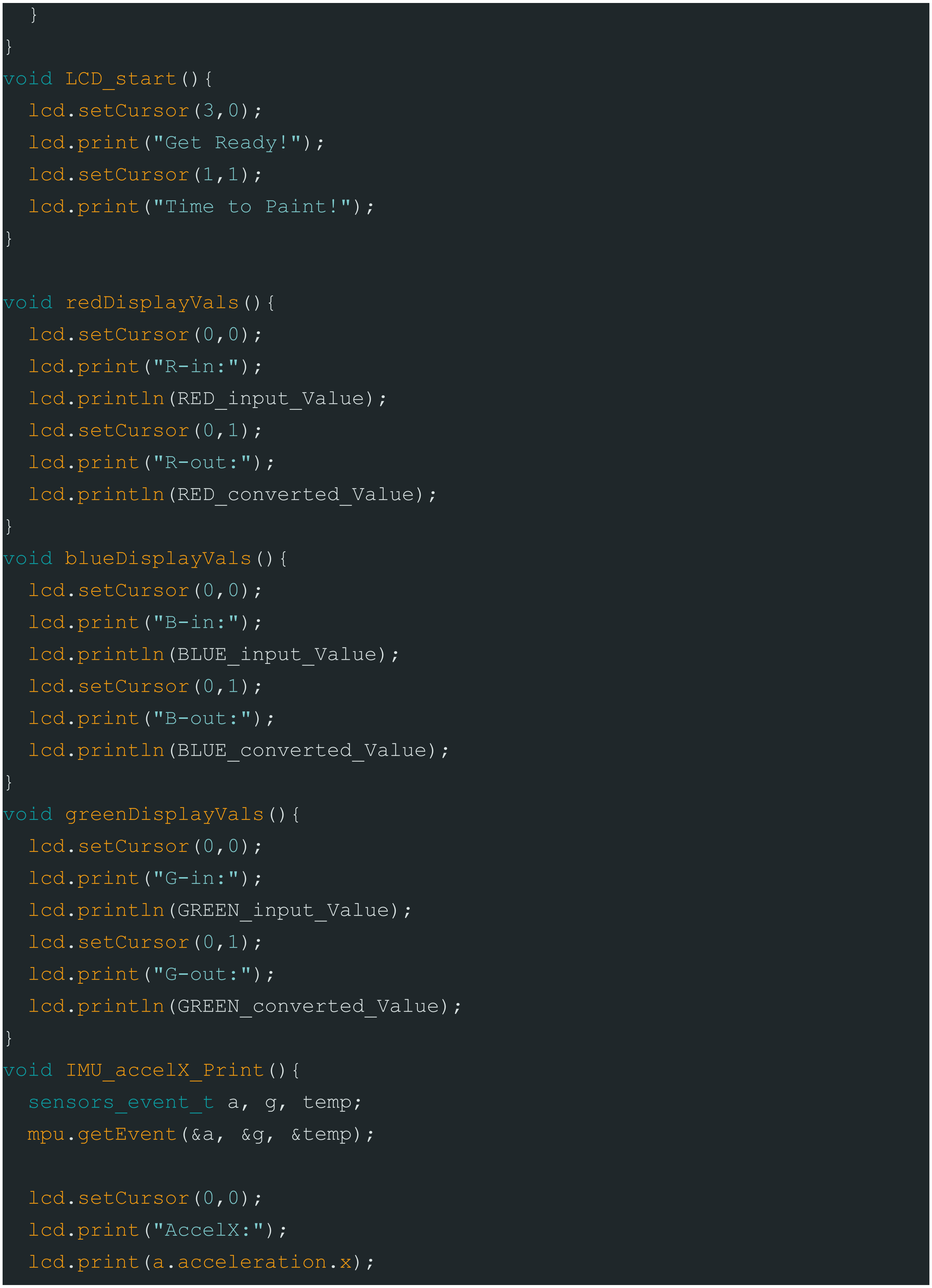
```
void Led_initialize(){
  Pinky_Thumb_pixel.begin();
  // INITIALIZE NeoPixel strip object (REQUIRED)
}
void Led_play(){
  Pinky_Thumb_pixel.setPixelColor(0,
Pinky_Thumb_pixel.Color(RED_converted_Value, GREEN_converted_Value,
BLUE_converted_Value));
  Pinky_Thumb_pixel.show();
}
```

Potentiometer (LCD scroll inputs)

```
const int poten_Value_Pin = A0;
const int pot_max = 4095;
const int pot_min = 20;

void potenMap(){
  poten_Value = analogRead(poten_Value_Pin);
  mappedpotValue_to_selectInt = map(poten_Value, pot_min, pot_max,
int_select_range_min, int_select_range_max);
}
```

LCD setup and Display

```
LiquidCrystal_I2C lcd(0x27,16,2);  // set the LCD address to 0x27 for a 16
chars and 2 line display

void LCDsetup()
{
  lcd.init();                      // initialize the lcd
  // Print a message to the LCD.
  lcd.backlight();
}
void LCDdisplay(){
  lcd.clear();
  if (mappedpotValue_to_selectInt  > -1  &&  mappedpotValue_to_selectInt <
1){ // if start statement
  LCD_start();
  }
  if (mappedpotValue_to_selectInt  > 0 &&  mappedpotValue_to_selectInt <
2){ //red if statement
  redDisplayVals();
  }
  if (mappedpotValue_to_selectInt  > 1  &&  mappedpotValue_to_selectInt <
3){ //green if statement
  greenDisplayVals();
  }
  if (mappedpotValue_to_selectInt  > 2  &&  mappedpotValue_to_selectInt <
4){ //blue if statement
  blueDisplayVals();
  }
  if (mappedpotValue_to_selectInt  > 3  &&  mappedpotValue_to_selectInt <
5){
    IMU_accelX_Print();
  }
  if (mappedpotValue_to_selectInt  > 4  &&  mappedpotValue_to_selectInt <
6){
    IMU_accelY_Print();
  }
  if (mappedpotValue_to_selectInt  > 5  &&  mappedpotValue_to_selectInt <
7){
    IMU_accelZ_Print();
```



```
  }
}
void LCD_start(){
  lcd.setCursor(3,0);
  lcd.print("Get Ready!");
  lcd.setCursor(1,1);
  lcd.print("Time to Paint!");
}

void redDisplayVals(){
  lcd.setCursor(0,0);
  lcd.print("R-in:");
  lcd.println(RED_input_Value);
  lcd.setCursor(0,1);
  lcd.print("R-out:");
  lcd.println(RED_converted_Value);
}
void blueDisplayVals(){
  lcd.setCursor(0,0);
  lcd.print("B-in:");
  lcd.println(BLUE_input_Value);
  lcd.setCursor(0,1);
  lcd.print("B-out:");
  lcd.println(BLUE_converted_Value);
}
void greenDisplayVals(){
  lcd.setCursor(0,0);
  lcd.print("G-in:");
  lcd.println(GREEN_input_Value);
  lcd.setCursor(0,1);
  lcd.print("G-out:");
  lcd.println(GREEN_converted_Value);
}
void IMU_accelX_Print(){
  sensors_event_t a, g, temp;
  mpu.getEvent(&a, &g, &temp);

  lcd.setCursor(0,0);
  lcd.print("AccelX:");
  lcd.print(a.acceleration.x);
```

```
}
void IMU_accelY_Print(){
  sensors_event_t a, g, temp;
  mpu.getEvent(&a, &g, &temp);

  lcd.setCursor(0,0);
  lcd.print("AccelY:");
  lcd.print(a.acceleration.y);
}
void IMU_accelZ_Print(){
  sensors_event_t a, g, temp;
  mpu.getEvent(&a, &g, &temp);

  lcd.setCursor(0,0);
  lcd.print("AccelZ:");
  lcd.print(a.acceleration.z);
}
```