```
#include "pitches.h"

int solo[] = {  NOTE_DS5, NOTE_C5,NOTE_DS5,NOTE_C6,NOTE_FS5,0};

int soloDurations[] = {
 4, 4, 4, 8, 6, 4 };

int riff[] = {
 NOTE_F2, NOTE_F2, NOTE_G2, NOTE_F2, NOTE_GS2, NOTE_F2, NOTE_AS2,
NOTE_A2
  };

// note durations: 4 = quarter note, 8 = eighth note, etc.:
int riffDurations[] = {
 4, 4, 4, 4, 4, 4, 4, 4
};

float transpose[] = {0.5,1.0,2,1.4};

const int button1Pin = 2;
int button1State = 0;
const int speakPin = 11;    // new position for speaker

#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>


#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 32 // OLED display height, in pixels

// Declaration for an SSD1306 display connected to I2C (SDA, SCL pins)
// The pins for I2C are defined by the Wire-library.
// On an arduino UNO:       A4(SDA), A5(SCL)
// On an arduino MEGA 2560: 20(SDA), 21(SCL)
// On an arduino LEONARDO:   2(SDA),  3(SCL), ...
#define OLED_RESET     6 // Reset pin # (or -1 if sharing Arduino reset pin)
#define SCREEN_ADDRESS 0x3C ///< See datasheet for Address; 0x3D for 128x64,
0x3C for 128x32
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

#define NUMFLAKES     10 // Number of snowflakes in the animation example

#define LOGO_HEIGHT   16
#define LOGO_WIDTH    16
```

```
static const unsigned char PROGMEM logo_bmp[] =
{ B00000000, B11000000,
  B00000001, B11000000,
  B00000001, B11000000,
  B00000011, B11100000,
  B11110011, B11100000,
  B11111110, B11111000,
  B01111110, B11111111,
  B00110011, B10011111,
  B00011111, B11111100,
  B00001101, B01110000,
  B00011011, B10100000,
  B00111111, B11100000,
  B00111111, B11110000,
  B01111100, B11110000,
  B01110000, B01110000,
  B00000000, B00110000 };

  // constants won't change. They're used here to set pin numbers:
int buttonPin = 2;     // the number of the pushbutton pin
int ledPin =  13;      // the number of the LED pin
int repCounter = 0;
// variables will change:
int buttonState = 0;         // variable for reading the pushbutton status

//Adafruit_MPU6050 mpu
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>

#include <Adafruit_NeoPixel.h>
#ifdef __AVR__
  #include <avr/power.h>
#endif
#include <Wire.h>

Adafruit_MPU6050 mpu;

#define PIN           6
#define NUMPIXELS      4

Adafruit_NeoPixel pixels = Adafruit_NeoPixel(NUMPIXELS, PIN, NEO_GRB +
NEO_KHZ800);//ALITOVE 100pcs WS2812B
```

```
void setup() {
 Serial.begin(115200);

  while (!Serial) {
    delay(10); // will pause Zero, Leonardo, etc until serial console opens
  }

  // Try to initialize!
  if (!mpu.begin()) {
    Serial.println("Failed to find MPU6050 chip");
    while (1) {
      delay(10);
    }
  }

  mpu.setAccelerometerRange(MPU6050_RANGE_16_G);
  mpu.setGyroRange(MPU6050_RANGE_250_DEG);
  mpu.setFilterBandwidth(MPU6050_BAND_21_HZ);
  Serial.println("");

   pixels.begin(); // This initializes the NeoPixel library.
   pixels.clear(); // reset pixels
   pixels.setPixelColor(0, pixels.Color(70,70,70));
   pixels.show();

   delay(50);


    pixels.setPixelColor(0, pixels.Color(0,0,0));

   pixels.show();



  delay(100);

  // SSD1306_SWITCHCAPVCC = generate display voltage from 3.3V internally
  if(!display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS)) {
    Serial.println(F("SSD1306 allocation failed"));
    for(;;); // Don't proceed, loop forever

 // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT);
}
```

```
 // Show initial display buffer contents on the screen --
  // the library initializes this with an Adafruit splash screen.
  display.display();
  delay(300); // Pause for 5/10 second

  // Clear the buffer
  display.clearDisplay();

  // Draw a single pixel in white
//  display.drawPixel(10, 10, SSD1306_WHITE);

  // Show the display buffer on the screen. You MUST call display() after
  // drawing commands to make them visible on screen!
  display.display();
  delay(1000);
  // display.display() is NOT necessary after every single drawing command,
  // unless that's what you want...rather, you can batch up a bunch of
  // drawing operations and then update the screen all at once by calling
  // display.display(). These examples demonstrate both approaches...
//
//  testdrawline();      // Draw many lines
//  testscrolltext();    // Draw scrolling text
//  testgoodbyetext();    // Draw goodbye text
//  testanimate(logo_bmp, LOGO_WIDTH, LOGO_HEIGHT); // Animate bitmaps
}




void loop() {





// read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);
// delay (2000);

  // check if the pushbutton is pressed. If it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
```

```
    digitalWrite(ledPin, HIGH);
//    testdrawline();      // Draw many lines
//    testdrawstyles();    // Draw 'stylized' characters
//    testscrolltext();    // Draw scrolling text

      /* Get new sensor events with the readings */
  sensors_event_t a, g, temp;
  mpu.getEvent(&a, &g, &temp);

  float ax = a.acceleration.x;

  /* Print out the values */
//  Serial.print(ax);
//  Serial.print(",");
//  Serial.print(a.acceleration.y);
//  Serial.print(",");
//  Serial.print(a.acceleration.z);
//  Serial.print(", ");
//  Serial.print(g.gyro.x);
//  Serial.print(",");
//  Serial.print(g.gyro.y);
//  Serial.print(",");
//  Serial.print(g.gyro.z);
//   Serial.print(",");
   ax = abs(ax);
   Serial.print(ax);
   Serial.print(",");
   float axt;
   if (ax > 15 ) {
   axt = ax;
   testscrolltext();    // Draw scrolling text
repCounter++;

if (repCounter < 2 ) {
  goText1();
  }

if (repCounter > 2 && repCounter < 12) {
  goText2();
  }

if (repCounter > 4 && repCounter < 12) {
  goText3();

  }
if (repCounter > 5 && repCounter < 12) {
  goText4();

  }

if (repCounter > 6 && repCounter < 12) {
  goText5();

  }

  if (repCounter > 8 && repCounter < 12) {
  goText6();

  }

  if (repCounter > 10 && repCounter < 12) {
  goText7();

  }

if (repCounter == 13) {

goTextdone();

  }




    Serial.print("                              ");
    Serial.print(repCounter);
    Serial.print("------------------------------ ");
   Serial.print(axt);
   } else {
    axt = 0;
   Serial.print(axt);
   }
   Serial.println(" ");

   pixels.setPixelColor(0, pixels.Color(axt*10,12,220));
     pixels.setPixelColor(1, pixels.Color(axt*10,62,22));
       pixels.setPixelColor(2, pixels.Color(axt*10,175,98));
   pixels.show();
  delay(50);
  } else {

    testgoodbyetext();    // Draw goodbye text
//    testanimate(logo_bmp, LOGO_WIDTH, LOGO_HEIGHT); // Animate bitmaps

 noTone(speakPin);

// turn LED off:
    digitalWrite(ledPin, LOW);
 display.clearDisplay();
  }



}
void testdrawline() {
  int16_t i;

  display.clearDisplay(); // Clear display buffer

  for(i=0; i<display.width(); i+=4) {
    display.drawLine(0, 0, i, display.height()-1, SSD1306_WHITE);
    display.display(); // Update screen with each newly-drawn line
    delay(1);
  }
  for(i=0; i<display.height(); i+=4) {
    display.drawLine(0, 0, display.width()-1, i, SSD1306_WHITE);
    display.display();
    delay(1);
  }
  delay(250);

  display.clearDisplay();

  for(i=0; i<display.width(); i+=4) {
    display.drawLine(0, display.height()-1, i, 0, SSD1306_WHITE);
    display.display();
    delay(1);
  }
  for(i=display.height()-1; i>=0; i-=4) {
    display.drawLine(0, display.height()-1, display.width()-1, i, SSD1306_WHITE);
    display.display();
    delay(1);
  }
  delay(250);
  display.clearDisplay();

  for(i=display.width()-1; i>=0; i-=4) {
    display.drawLine(display.width()-1, display.height()-1, i, 0, SSD1306_WHITE);
    display.display();
    delay(1);
  }
  for(i=display.height()-1; i>=0; i-=4) {
    display.drawLine(display.width()-1, display.height()-1, 0, i, SSD1306_WHITE);
    display.display();
    delay(1);
  }
  delay(500);

  display.clearDisplay();
}

void testdrawstyles(void) {
  display.clearDisplay();

  display.setTextSize(1);             // Normal 1:1 pixel scale
  display.setTextColor(SSD1306_WHITE);        // Draw white text
  display.setCursor(0,0);             // Start at top-left corner
  display.println(F("   TIME TO WORKOUT!"));

//  display.setTextColor(SSD1306_BLACK, SSD1306_WHITE); // Draw 'inverse' text
//  display.println(3.141592);

  display.setTextSize(2);             // Draw 2X-scale text
  display.setTextColor(SSD1306_WHITE);
  display.print(F("NO EXCUSES"));

  display.display();
  delay(4000);
}

void testscrolltext(void) {
  display.clearDisplay();

  display.setTextSize(2); // Draw 2X-scale text
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(10, 0);
  display.println(F("BEASTMODE"));
  display.display();      // Show initial text
  delay(50);
  // Scroll in various directions, pausing in-between:
```

```
  display.startscrollright(0x00, 0x0F);
  delay(50);
//  display.stopscroll();
//  delay(2000);
  display.startscrollleft(0x00, 0x0F);
  delay(50);
  display.stopscroll();
  delay(50);
  display.startscrolldiagright(0x00, 0x07);
  delay(50);
  display.startscrolldiagleft(0x00, 0x07);
  delay(50);
  display.stopscroll();
  delay(50);
}

void goText1(void) {
  display.clearDisplay();

  display.setTextSize(2); // Draw 2X-scale text
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(10, 0);
  display.println(F("LET'S GO"));
  display.display();      // Show initial text
  delay(50);
  // Scroll in various directions, pausing in-between:
  display.startscrollright(0x00, 0x0F);
  delay(50);
//  display.stopscroll();
//  delay(2000);
  display.startscrollleft(0x00, 0x0F);
  delay(50);
  display.stopscroll();
  delay(50);
  display.startscrolldiagright(0x00, 0x07);
  delay(50);
  display.startscrolldiagleft(0x00, 0x07);
  delay(50);
  display.stopscroll();
  delay(50);
}

void goText2(void) {
  display.clearDisplay();

  display.setTextSize(2); // Draw 2X-scale text
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(10, 0);
  display.println(F("ONE MORE!"));
  display.display();      // Show initial text
  delay(50);
  // Scroll in various directions, pausing in-between:
  display.startscrollright(0x00, 0x0F);
  delay(50);
  display.startscrollleft(0x00, 0x0F);
  delay(50);
  display.stopscroll();
  delay(50);
  display.startscrolldiagright(0x00, 0x07);
  delay(50);
  display.startscrolldiagleft(0x00, 0x07);
  delay(50);
  display.stopscroll();
  delay(50);
}


void goText3(void) {
  display.clearDisplay();

  display.setTextSize(2); // Draw 2X-scale text
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(10, 0);
  display.println(F("KEEP       PUSHING!"));
  display.display();      // Show initial text
  delay(50);
  // Scroll in various directions, pausing in-between:
  display.startscrollright(0x00, 0x0F);
  delay(50);
  display.startscrollleft(0x00, 0x0F);
  delay(50);
  display.stopscroll();
  delay(50);
  display.startscrolldiagright(0x00, 0x07);
  delay(50);
  display.startscrolldiagleft(0x00, 0x07);
  delay(50);
  display.stopscroll();
  delay(50);
}

void goText4(void) {
  display.clearDisplay();

  display.setTextSize(2); // Draw 2X-scale text
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(10, 0);
  display.println(F("HECK       YEAH!"));
  display.display();      // Show initial text
  delay(50);
  // Scroll in various directions, pausing in-between:
  display.startscrollright(0x00, 0x0F);
  delay(50);
  display.startscrollleft(0x00, 0x0F);
  delay(50);
  display.stopscroll();
  delay(50);
  display.startscrolldiagright(0x00, 0x07);
  delay(50);
  display.startscrolldiagleft(0x00, 0x07);
  delay(50);
  display.stopscroll();
  delay(50);
}

void goText5(void) {
  display.clearDisplay();

  display.setTextSize(2); // Draw 2X-scale text
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(10, 0);
  display.println(F("MAGNIFICO"));
  display.display();      // Show initial text
  delay(50);
  // Scroll in various directions, pausing in-between:
  display.startscrollright(0x00, 0x0F);
  delay(50);
  display.startscrollleft(0x00, 0x0F);
  delay(50);
  display.stopscroll();
  delay(50);
  display.startscrolldiagright(0x00, 0x07);
  delay(50);
  display.startscrolldiagleft(0x00, 0x07);
  delay(50);
  display.stopscroll();
  delay(50);
}
```

```
void goText6(void) {
  display.clearDisplay();

  display.setTextSize(2); // Draw 2X-scale text
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(10, 0);
  display.println(F("WAY TO GO!"));
  display.display();      // Show initial text
  delay(50);
  // Scroll in various directions, pausing in-between:
  display.startscrollright(0x00, 0x0F);
  delay(50);
  display.startscrollleft(0x00, 0x0F);
  delay(50);
  display.stopscroll();
  delay(50);
  display.startscrolldiagright(0x00, 0x07);
  delay(50);
  display.startscrolldiagleft(0x00, 0x07);
  delay(50);
  display.stopscroll();
  delay(50);
}

void goText7(void) {
  display.clearDisplay();

  display.setTextSize(2); // Draw 2X-scale text
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(10, 0);
  display.println(F("OMFG LETS GO!"));
  display.display();      // Show initial text
  delay(50);
  // Scroll in various directions, pausing in-between:
  display.startscrollright(0x00, 0x0F);
  delay(50);
  display.startscrollleft(0x00, 0x0F);
  delay(50);
  display.stopscroll();
  delay(50);
  display.startscrolldiagright(0x00, 0x07);
  delay(50);
  display.startscrolldiagleft(0x00, 0x07);
  delay(50);
  display.stopscroll();
  delay(50);
}

void goTextdone(void) {
  display.clearDisplay();

  display.setTextSize(2); // Draw 2X-scale text
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(10, 0);
  display.println(F("ALL DONE"));
  display.display();      // Show initial text
  delay(50);
  // Scroll in various directions, pausing in-between:
  display.startscrollright(0x00, 0x0F);
  delay(50);
//  display.stopscroll();
//  delay(2000);
  display.startscrollleft(0x00, 0x0F);
  delay(50);
  display.stopscroll();
  delay(50);
  display.startscrolldiagright(0x00, 0x07);
  delay(50);
  display.startscrolldiagleft(0x00, 0x07);
  delay(50);
  display.stopscroll();
  delay(50);
}

void testgoodbyetext(void) {
  display.clearDisplay();

  display.setTextSize(2); // Draw 2X-scale text
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(10, 0);
  display.println(F(" GOODBYE!"));
  display.display();      // Show initial text
  delay(5000);


//  // Scroll in various directions, pausing in-between:
//  display.startscrollright(0x00, 0x0F);
//  delay(200);
//  display.stopscroll();
//  delay(200);
//  display.startscrollleft(0x00, 0x0F);
//  delay(200);
//  display.stopscroll();
//  delay(200);
//  display.startscrolldiagright(0x00, 0x07);
//  delay(200);
//  display.startscrolldiagleft(0x00, 0x07);
//  delay(200);
//  display.stopscroll();
//  delay(200);
//  display.clearDisplay();
}

void testdrawbitmap(void) {
  display.clearDisplay();

  display.drawBitmap(
    (display.width()  - LOGO_WIDTH ) / 2,
    (display.height() - LOGO_HEIGHT) / 2,
    logo_bmp, LOGO_WIDTH, LOGO_HEIGHT, 1);
  display.display();
  delay(1000);
}

//#define XPOS   0 // Indexes into the 'icons' array in function below
//#define YPOS   1
//#define DELTAY 2
//
//void testanimate(const uint8_t *bitmap, uint8_t w, uint8_t h) {
//  int8_t f, icons[NUMFLAKES][3];
//
//  // Initialize 'snowflake' positions
//  for(f=0; f< NUMFLAKES; f++) {
//    icons[f][XPOS]   = random(1 - LOGO_WIDTH, display.width());
//    icons[f][YPOS]   = -LOGO_HEIGHT;
//    icons[f][DELTAY] = random(1, 6);
//    Serial.print(F("x: "));
//    Serial.print(icons[f][XPOS], DEC);
//    Serial.print(F(" y: "));
//    Serial.print(icons[f][YPOS], DEC);
//    Serial.print(F(" dy: "));
//    Serial.println(icons[f][DELTAY], DEC);
//
//  }
//
//  for(;;) { // Loop forever...
//    display.clearDisplay(); // Clear the display buffer
```

```
//
//    // Draw each snowflake:
//    for(f=0; f< NUMFLAKES; f++) {
//      display.drawBitmap(icons[f][XPOS], icons[f][YPOS], bitmap, w, h, 
SSD1306_WHITE);
//    }
//
//    display.display(); // Show the display buffer on the screen
//    delay(200);        // Pause for 1/10 second
//
//    // Then update coordinates of each flake...
//    for(f=0; f< NUMFLAKES; f++) {
//      icons[f][YPOS] += icons[f][DELTAY];
//      // If snowflake is off the bottom of the screen...
//      if (icons[f][YPOS] >= display.height()) {
//        // Reinitialize to a random position, just off the top
//        icons[f][XPOS]   = random(1 - LOGO_WIDTH, display.width());
//        icons[f][YPOS]   = -LOGO_HEIGHT;
//        icons[f][DELTAY] = random(1, 6);
//         display.clearDisplay();
//      }
```

 Pitches.H

```
#define NOTE_B0  31
#define NOTE_C1  33
#define NOTE_CS1 35
#define NOTE_D1  37
#define NOTE_DS1 39
#define NOTE_E1  41
#define NOTE_F1  44
#define NOTE_FS1 46
#define NOTE_G1  49
#define NOTE_GS1 52
#define NOTE_A1  55
#define NOTE_AS1 58
#define NOTE_B1  62
#define NOTE_C2  65
#define NOTE_CS2 69
#define NOTE_D2  73
#define NOTE_DS2 78
#define NOTE_E2  82
#define NOTE_F2  87
#define NOTE_FS2 93
#define NOTE_G2  98
#define NOTE_GS2 104
```

```
#define NOTE_A2  110
#define NOTE_AS2 117
#define NOTE_B2  123
#define NOTE_C3  131
#define NOTE_CS3 139
#define NOTE_D3  147
#define NOTE_DS3 156
#define NOTE_E3  165
#define NOTE_F3  175
#define NOTE_FS3 185
#define NOTE_G3  196
#define NOTE_GS3 208
#define NOTE_A3  220
#define NOTE_AS3 233
#define NOTE_B3  247
#define NOTE_C4  262
#define NOTE_CS4 277
#define NOTE_D4  294
#define NOTE_DS4 311
#define NOTE_E4  330
#define NOTE_F4  349
#define NOTE_FS4 370
#define NOTE_G4  392
#define NOTE_GS4 415
#define NOTE_A4  440
#define NOTE_AS4 466
#define NOTE_B4  494
#define NOTE_C5  523
#define NOTE_CS5 554
#define NOTE_D5  587
#define NOTE_DS5 622
#define NOTE_E5  659
#define NOTE_F5  698
#define NOTE_FS5 740
#define NOTE_G5  784
#define NOTE_GS5 831
#define NOTE_A5  880
#define NOTE_AS5 932
#define NOTE_B5  988
#define NOTE_C6  1047
#define NOTE_CS6 1109
#define NOTE_D6  1175
#define NOTE_DS6 1245
#define NOTE_E6  1319
#define NOTE_F6  1397
#define NOTE_FS6 1480
```

```
#define NOTE_G6  1568
#define NOTE_GS6 1661
#define NOTE_A6  1760
#define NOTE_AS6 1865
#define NOTE_B6  1976
#define NOTE_C7  2093
#define NOTE_CS7 2217
#define NOTE_D7  2349
#define NOTE_DS7 2489
#define NOTE_E7  2637
#define NOTE_F7  2794
#define NOTE_FS7 2960
#define NOTE_G7  3136
#define NOTE_GS7 3322
#define NOTE_A7  3520
#define NOTE_AS7 3729
#define NOTE_B7  3951
#define NOTE_C8  4186
#define NOTE_CS8 4435
#define NOTE_D8  4699
#define NOTE_DS8 4978
```