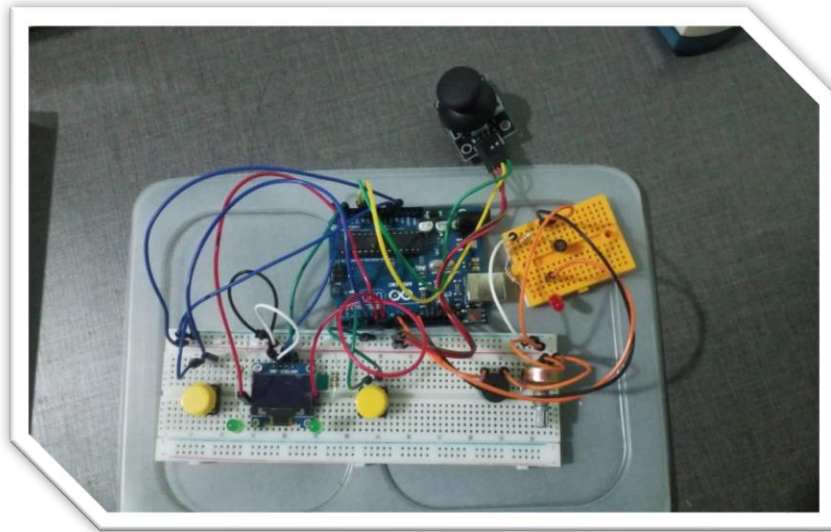


Arduino Project



Game on Arduino Uno R3
Using OLED 0.96" screen as display

May 2021

อุปกรณ์

ARDUINO UNO R3

การแสดงผลของเกมส์จะแสดงออกมาทั้งรูปภาพและเสียง โดยอุปกรณ์ต่อพ่วงสองตัวได้แก่

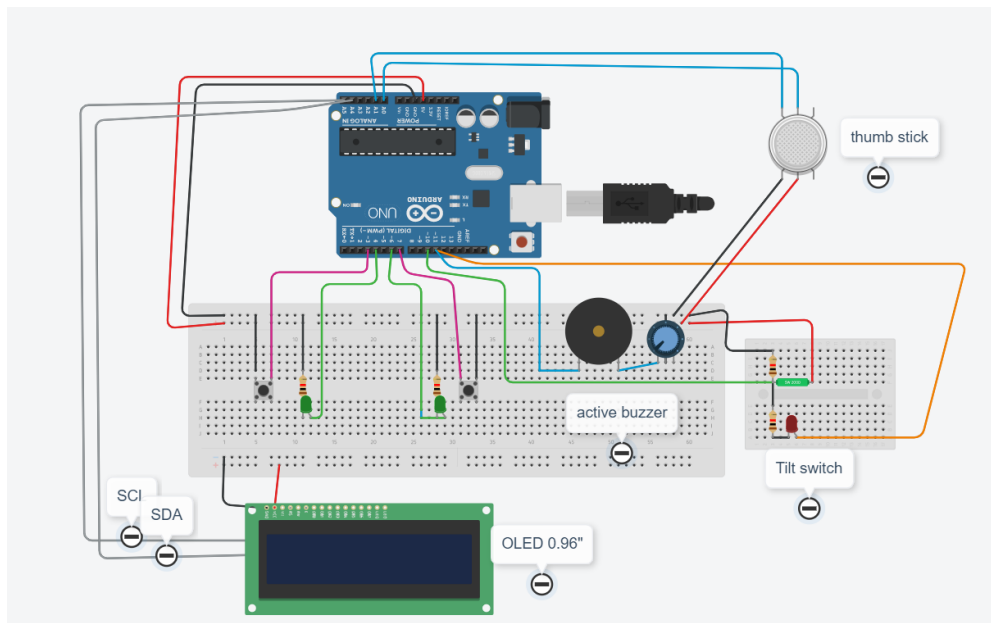
1. จอ OLED 0.96" (I2C) สีฟ้า
2. Active buzzer (ฮอด)

ผู้เล่นสามารถควบคุมเกมส์ได้โดยผ่านอุปกรณ์ต่อพ่วงสามตัวได้แก่

1. สวิตช์ 2 ตัว
2. คันโยก อนาล็อก
3. Tilt switch

อุปกรณ์อื่นๆ

1. หลอดไฟ LED สีเขียว x2
2. หลอดไฟ LED สีแดง x1
3. ฝาพลาสติกสีเหลือง x2
4. ตัวต้านทาน 1K x3
5. ตัวต้านทานปรับค่าได้

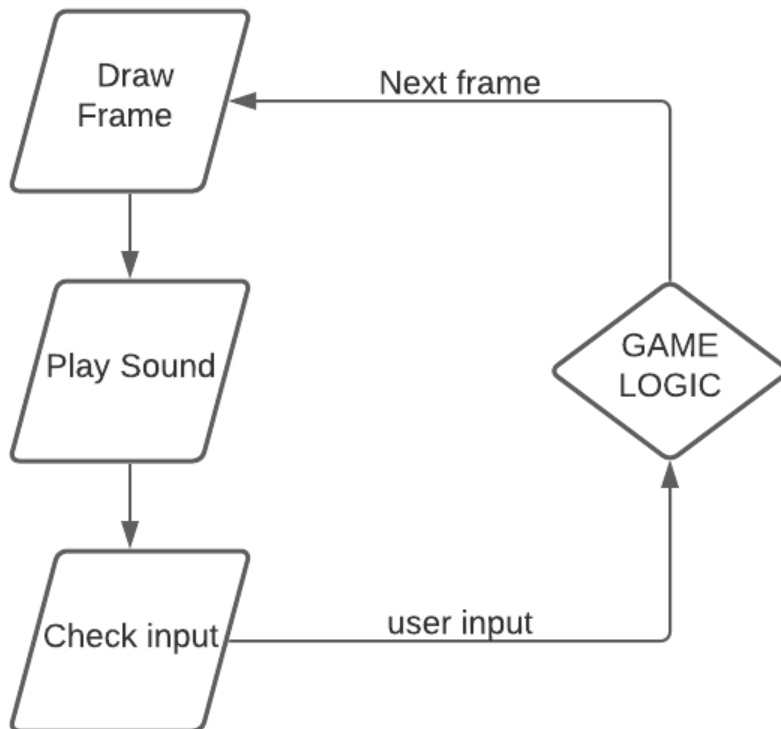


หลักการทำงาน

DEMO VIDEO 1/2 <https://youtu.be/Du07v2zZGrY>

DEMO VIDEO 2/2 <https://youtu.be/qAGQX1hmQ1A>

หลักการทำงานของเกมนั้นจะเป็นไปตาม flow chart ที่แสดงในรูปข้างล่าง เราจะวาดรูปลงจอ เล่นเสียง แล้วก็เช็คสถานะของอุปกรณ์ต่อพ่วงว่ากดอะไรอยู่รึเปล่า ตัว game logic (code) จะคำนวณว่าเฟรมต่อไปคืออะไรแล้วก็นวนลูอย่างนี้ไปจนจบเกมส์ ตัว game logic (code) จะถูกอธิบายไว้อย่างคร่าวๆ ในภาคผนวก



เริ่มเกมส์

1. หลังจากเปิดเครื่อง ให้ผู้เล่นทำการกดปุ่มซ้ายเพื่อเริ่มเกมส์
2. โยก tilt switch 90 องศา เพื่อเริ่มด่าน (ไม่เอียงส่งสัญญาณ 0 , เอียงส่งสัญญาณ 1)

หลักการการวาดรูปลงบนจอ OLED

การวาดรูปลงบนจอ OLED นั้นมี library สำเร็จรูปให้ใช้แล้วชื่อว่า Adafruit_SSD1306

https://github.com/adafruit/Adafruit_SSD1306/

โดยเราจะใช้ function ดังนี้ (ใช้เยอะกว่าที่เขียนมา แต่เพื่อการยกตัวอย่างจึง เขียนมาแค่อันที่สำคัญ) โดยหลักการแล้วจะมี อยู่สามขั้นตอนคือล้างจอ เขียนจอ และอัปเดตจอเพื่อแสดงผล

-----clear-----

display.clearDisplay() - ล้างหน้าจอ

-----draw something-----

display.fillCircle(x,y,radius,color) – วาดวงกลมทึบ

display.drawRect(x,y,width,height,color) – วาดสี่เหลี่ยมกลวง

display.drawBitmap(x,y,picture,size_x,size_y,color) – วาด picture ที่เราเตรียมไว้

display.println(“any message”) - แสดงผลตัวอักษรลงบนหน้าจอ

-----update display-----

display.display() - อัปเดตหน้าจอ

ตัวอย่างการวาดรูปโดยใช้ `display.drawBitmap()`

สำหรับรูปที่วาดนั้น สร้างมาจากขนาด 40x40 pixel (โดยใช้โปรแกรมสร้าง pixel art ที่หาได้ทั่วไป) หลังจากนั้นใช้โปรแกรมแปลงไฟล์จาก png ให้เป็น bitmap แล้วหลังจากนั้นเราจะนำไฟล์ bmp นี้แปลงเป็น 1D array เพื่อนำไปใช้งานใน arduino

Evil.png



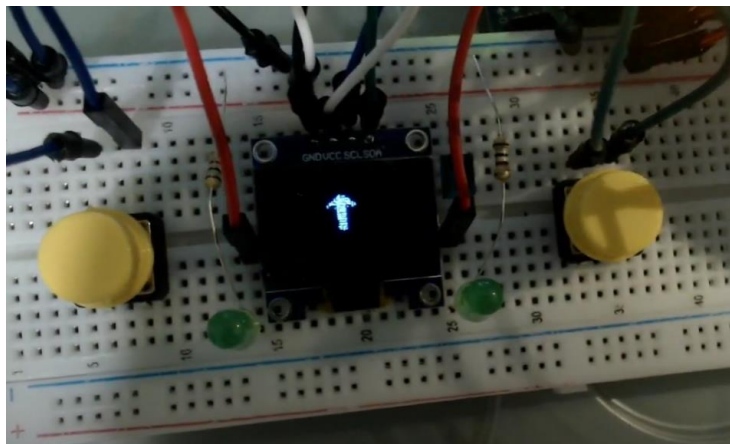
array ข้างล่างมาจากการแปลงรูป evil.png ให้เป็น evil.bmp แล้วแปลงต่อให้เป็น array evil[]

```
const unsigned char PROGMEM evil [] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x0
7, 0x80, 0x00, 0x00, 0xE0, 0x07, 0x80, 0x00, 0x00, 0xE0, 0x07, 0x80, 0x00, 0x00,
0xF0, 0x07, 0x80, 0x7F, 0xC0, 0xF0, 0x0F, 0x83, 0xFF, 0xEF, 0xF0, 0x0F, 0xFF, 0xF
F, 0xFF, 0xF0, 0x0F, 0xFF, 0x9C, 0xFF, 0xF0, 0x0F, 0xFF, 0xFF, 0xFF, 0xF0, 0x07,
0xFF, 0xFF, 0xE0, 0x00, 0x00, 0x03, 0xFF, 0xE0, 0x00, 0x00, 0x03, 0xFF, 0xE0, 0x0
0, 0x00, 0x03, 0xFF, 0xE0, 0x00, 0x00, 0x00, 0x00, 0xC0, 0xC0, 0x00, 0x00, 0x00, 0xEE,
0xC0, 0x00, 0x00, 0x00, 0xFF, 0xC0, 0x00, 0x00, 0x00, 0xFF, 0xC0, 0x00, 0x00, 0x0
0, 0x0E, 0x00, 0x00, 0x00, 0x00, 0x0E, 0x00, 0x00, 0x0F, 0xD0, 0x0E, 0x10, 0x7E,
0x0F, 0xD0, 0x0E, 0x10, 0x7E, 0x00, 0xFF, 0xFF, 0xFF, 0xE0, 0x0F, 0xC4, 0x0E, 0x0
2, 0x7E, 0x0F, 0xC4, 0x0E, 0x02, 0x7E, 0x00, 0x00, 0x0E, 0x00, 0x00, 0x00, 0x00,
0x0E, 0x00, 0x00, 0x00, 0x00, 0x0E, 0x00, 0x00, 0x00, 0x00, 0x0E, 0x00, 0x00, 0x0
0, 0x0F, 0xFF, 0xFE, 0x00, 0x00, 0x0F, 0xFF, 0xFE, 0x00, 0x00, 0x0E, 0x00, 0x0E,
0x00, 0x00, 0x0E, 0x00, 0x0E, 0x00, 0x0E, 0x00, 0x0E, 0x00, 0x0E, 0x00, 0x3E, 0x0
0, 0x0F, 0x80, 0x00, 0x3E, 0x00, 0x0F, 0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00};
```

หลังจากที่ได้ array ภาพมาแล้วก็สามารถนำมาวาดในจอ oled ได้โดยคำสั่ง

```
display.drawBitmap(x,y,evil,width,height,color)
```

ด้าน 1 เกมส์กดลูกศรตามเสียงเพลง



เมื่อเริ่มเกมส์ เสียงเพลงจะดังขึ้นมา ผู้เล่นสามารถปรับระดับความดังได้โดยให้ทำการหมุนตัวต้านทานปรับค่าได้ ทวนเข็มนาฬิกาเพื่อลดเสียง หรือตามเข็มนาฬิกาเพื่อเพิ่มเสียง เมื่อเพลงเริ่มเล่นแล้ว จะมีลูกศรปรากฏขึ้นมา ให้ผู้เล่น โยกจอยอนาล็อกไปตามทิศทางของลูกศร หากเลือกถูก จะได้คะแนน แล้วตัวเกมส์จะสุ่มลูกศรต่อไปมาให้กดต่อ (มีโอกาสสุ่มได้ 4 แบบ ได้แก่ขึ้น ลง ซ้าย ขวา) ตัวเกมส์จะเล่นเป็นเวลาทั้งหมด 30 วินาที ก่อนที่จะแสดงผลคะแนนที่ผู้เล่นกดได้ถูกต้อง

การรับค่าอนาล็อก

คันโยกอนาล็อกจะมี output สองค่าคือ Rx, Ry ซึ่งมีค่าตั้งแต่ 0-1024 , เช่นถ้าเราโยกไปมุมล่างซ้ายสุด จะอ่านค่าได้

Rx=0,Ry=0 แต่ถ้า โยกไปมุมบนขวาสุดจะอ่านค่าได้ Rx=1024,Ry=1024 , ส่วนถ้าไม่โยกเลยจะอ่านค่าได้

Rx=512,Ry=512

เพื่อความสับสนในการเล่นเกมส์ ผู้เล่นไม่จำเป็นต้องโยกไปสุดทาง แต่เพียงแคโยกเล็กน้อยก็เพียงพอ เช่น Rx>600 ก็จะสามารถได้ว่าเป็นการโยกไปทางขวา

เสียงเพลง และ เสียงในเกมส์

เสียงในเกมส์ทั้งหมดจะถูกเล่นผ่าน active buzzer ด้วยฟังก์ชัน buzz

buzz(pin,frequency, duration)

<https://gist.github.com/lfzawacki/4149836>

เสียงในเกมส์จะแบ่งเป็นสองส่วนคือ เสียงเพลงในด้าน 1 ที่นำมาโน้ตและจังหวะมาจาก

<https://github.com/xitanggg/-Pirates-of-the-Caribbean-Theme-Song>

ส่วนเสียงส่วนที่สอง คือเสียงทั้งหมดที่ไม่ใช่เพลงในด้านแรก เสียงเหล่านี้ถูกสร้างขึ้นมาเองทั้งหมด




การสร้างอนิเมชัน เสียงเพลง และการอ่านค่าจากคั่นโยกอนาล็อค

หลักการพื้นฐานคือเราจะ วาดภาพมา 1-3 เฟรม หลังจากนั้นก็เล่นเพลงตามโน้ตมาหนึ่งจังหวะ หลังจากนั้นก็จะเห็นว่าผู้เล่น กดปุ่มอะไรอยู่ ถ้ากดถูกก็จบไปเสต็ปต่อไป แต่ถ้าไม่ถูก ก็จะมีวนอยู่ที่ลูกศรตัวเดิม เล่นโน้ตตัวต่อไปเรื่อยๆ

ด่าน 2 เกมสับสนบอส



ผู้เล่นต้องหลบการโจมตีของบอสให้ได้เป็นเวลา 1 นาที ถึงจะชนะ บอสจะมีท่าโจมตี 3 ท่าได้แก่

1. ยิงซ้าย ให้กดปุ่มขวาเพื่อโยกหลบไปขวา 
2. ยิงขวา ให้กดปุ่มซ้ายเพื่อโยกหลบไปซ้าย 
3. ท่าพิเศษ ให้โยก tilt switch ไป 90 องศาเพื่อกระโดดหลบ 
- 4.

โดยผู้เล่นจะต้องหลบให้ได้เป็นเวลา 1 นาทีถึงจะชนะ โดยที่การโจมตีของบอสจะเร็วขึ้นเรื่อยๆ หากถูกโจมตีครั้งเดียวก็จะ game over

ปัญหาที่พบบ่อยระหว่างทำ

1. Memory ไม่พอ

There are three pools of memory in the microcontroller used on avr-based Arduino boards :

- ◆ Flash memory (program space), is where the Arduino sketch is stored.
- ◆ SRAM (static random access memory) is where the sketch creates and manipulates variables when it runs.
- ◆ EEPROM is memory space that programmers can use to store long-term information.

Flash memory and EEPROM memory are non-volatile (the information persists after the power is turned off). SRAM is volatile and will be lost when the power is cycled.

The ATmega328P chip found on the Uno has the following amounts of memory:

```
Flash 32k bytes (of which .5k is used for the bootloader)
SRAM 2k bytes
EEPROM 1k byte
```

เนื่องจากการที่จะ initialize จอ oled นั้นใช้ ram ค่อนข้างเยอะ พอเพิ่มตัวโน้ตกับไฟล์ภาพเข้าไปทำให้ ram ไม่พอ และไม่สามารถใช้งานจอ oled ได้ (จอดำ) จึงต้องใช้คำสั่ง PROGMEM เพื่อนำตัวโน้ตและรูปภาพไปเก็บไว้ใน flash แทน

เช่นคำสั่ง `const int PROGMEM var [] = 10` จะเก็บตัวแปร `var=10` ไว้ใน flash แทนที่จะเก็บไว้ใน SRAM

2. ไม่สามารถเล่น ภาพและเสียงพร้อมกันได้

วิธีแก้คือ ให้เล่นภาพ เล่นเสียง สลับกัน ซึ่งมันเร็วมากจนตาและหูคนมองว่ามันเกิดขึ้นพร้อมกัน

3. สายไม่พอ

มีอุปกรณ์หลายตัวที่จะใช้ในตอนแรก เช่น 7 segment และ led matrix 8x8 แต่ว่าอุปกรณ์เหล่านี้ใช้ขาเยอะมาก

วิธีแก้คือซื้อ ตัว driver ลดขามาต่อ แต่ผมตัดปัญหาโดยการเลือกที่จะไม่ใช้อุปกรณ์เหล่านี้

4. Servo motor (MG90S) ไม่สามารถดึงไฟจาก arduino uno โดยตรงได้

ในตอนแรก ผมจะทำเกมส์ฟันดาบ โดยให้ผู้เล่นหมุนมอเตอร์ที่ติดกับใบมีดพลาสติกจำลองให้เหมือนกับเราเลือกทิศทางการฟันของดาบได้ เพื่อสู้กับศัตรูในจอ แต่ปัญหาคือ servo motor นั้นต้องการไฟ 5 V ซึ่งปกติแล้ว arduino ควรจ่ายไฟได้เพียงพอ แต่พอต่ออุปกรณ์หลายๆตัวแล้ว arduino ไม่สามารถจ่ายไฟได้เพียงพอ เช่นสั่งให้ motor หมุน 90 องศา มันก็หมุนไปได้แค่ 10-20 องศาเท่านั้น (แต่ถ้าต่อแค่ motor แค่นี้จะทำงานได้ปกติ) แล้วที่ยิ่งแปลกไปกว่านั้น คือถ้าต่อ motor กับถ่านถั่วๆ $1.5V \times 4 = 6V$ motor จะไม่สามารถหมุนได้แม้แต่นิดเดียว คาดว่าสาเหตุเกิดจากถ่านถั่วๆไม่สามารถจ่ายกระแสได้เพียงพอ ผมจึงตัดอุปกรณ์นี้ทิ้งไปเนื่องจากข้อจำกัดทางเวลา

ภาคผนวก

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

#define joyX A0
#define joyY A1

#define SCREEN_WIDTH 128 // OLED display width, in pixels
#define SCREEN_HEIGHT 64 // OLED display height, in pixels

// note เพลง และทำนอง จาก https://github.com/xitanggg/-Pirates-of-the-Caribbean-Theme-Song

#define NOTE_C4 262
#define NOTE_D4 294
#define NOTE_E4 330
#define NOTE_F4 349
#define NOTE_G4 392
#define NOTE_A4 440
#define NOTE_B4 494
#define NOTE_C5 523
#define NOTE_D5 587
#define NOTE_E5 659
#define NOTE_F5 698
#define NOTE_G5 784
#define NOTE_A5 880
#define NOTE_B5 988

const int notes[] PROGMEM = {
    NOTE_E4, NOTE_G4, NOTE_A4, NOTE_A4, 0,
    NOTE_A4, NOTE_B4, NOTE_C5, NOTE_C5, 0,
    NOTE_C5, NOTE_D5, NOTE_B4, NOTE_B4, 0,
    NOTE_A4, NOTE_G4, NOTE_A4, 0,

    NOTE_E4, NOTE_G4, NOTE_A4, NOTE_A4, 0,
    NOTE_A4, NOTE_B4, NOTE_C5, NOTE_C5, 0,
    NOTE_C5, NOTE_D5, NOTE_B4, NOTE_B4, 0,
    NOTE_A4, NOTE_G4, NOTE_A4, 0,
```

NOTE_E4, NOTE_G4, NOTE_A4, NOTE_A4, 0,
NOTE_A4, NOTE_C5, NOTE_D5, NOTE_D5, 0,
NOTE_D5, NOTE_E5, NOTE_F5, NOTE_F5, 0,
NOTE_E5, NOTE_D5, NOTE_E5, NOTE_A4, 0,

NOTE_A4, NOTE_B4, NOTE_C5, NOTE_C5, 0,
NOTE_D5, NOTE_E5, NOTE_A4, 0,
NOTE_A4, NOTE_C5, NOTE_B4, NOTE_B4, 0,
NOTE_C5, NOTE_A4, NOTE_B4, 0,
NOTE_A4, NOTE_A4,

NOTE_A4, NOTE_B4, NOTE_C5, NOTE_C5, 0,
NOTE_C5, NOTE_D5, NOTE_B4, NOTE_B4, 0,
NOTE_A4, NOTE_G4, NOTE_A4, 0,

NOTE_E4, NOTE_G4, NOTE_A4, NOTE_A4, 0,
NOTE_A4, NOTE_B4, NOTE_C5, NOTE_C5, 0,
NOTE_C5, NOTE_D5, NOTE_B4, NOTE_B4, 0,
NOTE_A4, NOTE_G4, NOTE_A4, 0,

NOTE_E4, NOTE_G4, NOTE_A4, NOTE_A4, 0,
NOTE_A4, NOTE_C5, NOTE_D5, NOTE_D5, 0,
NOTE_D5, NOTE_E5, NOTE_F5, NOTE_F5, 0,
NOTE_E5, NOTE_D5, NOTE_E5, NOTE_A4, 0,

NOTE_A4, NOTE_B4, NOTE_C5, NOTE_C5, 0,
NOTE_D5, NOTE_E5, NOTE_A4, 0,
NOTE_A4, NOTE_C5, NOTE_B4, NOTE_B4, 0,
NOTE_C5, NOTE_A4, NOTE_B4, 0,

NOTE_E5, 0, 0, NOTE_F5, 0, 0,
NOTE_E5, NOTE_E5, 0, NOTE_G5, 0, NOTE_E5, NOTE_D5, 0, 0,
NOTE_D5, 0, 0, NOTE_C5, 0, 0,
NOTE_B4, NOTE_C5, 0, NOTE_B4, 0, NOTE_A4,

NOTE_E5, 0, 0, NOTE_F5, 0, 0,
NOTE_E5, NOTE_E5, 0, NOTE_G5, 0, NOTE_E5, NOTE_D5, 0, 0,
NOTE_D5, 0, 0, NOTE_C5, 0, 0,
NOTE_B4, NOTE_C5, 0, NOTE_B4, 0, NOTE_A4};

```
const int durations[] PROGMEM = {  
    125, 125, 250, 125, 125,  
    125, 125, 250, 125, 125,  
    125, 125, 250, 125, 125,  
    125, 125, 375, 125,
```

125, 125, 250, 125, 125,
125, 125, 250, 125, 125,
125, 125, 250, 125, 125,
125, 125, 375, 125,

125, 125, 250, 125, 125,
125, 125, 250, 125, 125,
125, 125, 250, 125, 125,
125, 125, 125, 250, 125,

125, 125, 250, 125, 125,
250, 125, 250, 125,
125, 125, 250, 125, 125,
125, 125, 375, 375,

250, 125,
125, 125, 250, 125, 125,
125, 125, 250, 125, 125,
125, 125, 375, 125,

125, 125, 250, 125, 125,
125, 125, 250, 125, 125,
125, 125, 250, 125, 125,
125, 125, 375, 125,

125, 125, 250, 125, 125,
125, 125, 250, 125, 125,
125, 125, 250, 125, 125,
125, 125, 125, 250, 125,

125, 125, 250, 125, 125,
250, 125, 250, 125,
125, 125, 250, 125, 125,
125, 125, 375, 375,

250, 125, 375, 250, 125, 375,
125, 125, 125, 125, 125, 125, 125, 125, 375,
250, 125, 375, 250, 125, 375,
125, 125, 125, 125, 125, 500,

250, 125, 375, 250, 125, 375,
125, 125, 125, 125, 125, 125, 125, 125, 375,
250, 125, 375, 250, 125, 375,
125, 125, 125, 125, 125, 500};

```
const int PROGMEM note_length = sizeof(notes)/sizeof(int);
int note_index = 0;
```

// animation เพื่อแสดงผลออกทางจอ เช่น down1-down7 เป็นข้อมูลที่ไว้วาดอนิเมชั่นของลูกศรลง

[illegible][illegible]

```
const unsigned char PROGMEM down3 [] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x08, 0x00, 0x00, 0x00, 0x00, 0x6A, 0x00, 0x00, 0x
    00, 0x00, 0x28, 0x00, 0x00, 0x00, 0x00, 0x6E, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x6E, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x
    7E, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xFE, 0x00, 0x00, 0x00,
```

[illegible][illegible][illegible]

[illegible]

```
const unsigned char PROGMEM down8 [] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x2C, 0x00, 0x00, 0x00, 0x00, 0x28, 0x00, 0x00, 0x00, 0x00, 0x
2E, 0x00, 0x00, 0x00, 0x00, 0x28, 0x00, 0x00, 0x00, 0x00, 0x6A, 0x00, 0x00, 0x00,
0x00, 0x08, 0x00, 0x00, 0x00, 0x00, 0x2E, 0x00, 0x00, 0x00, 0x00, 0x28, 0x00, 0x
00, 0x00, 0x00, 0x3A, 0x00, 0x00, 0x00, 0x00, 0x28, 0x00, 0x00, 0x00, 0x00, 0x6E,
0x00, 0x00, 0x00, 0x00, 0x08, 0x00, 0x00, 0x00, 0x00, 0x6A, 0x00, 0x00, 0x00, 0x
00, 0x28, 0x00, 0x00, 0x00, 0x00, 0x3A, 0x00, 0x00, 0x00, 0x00, 0x2A, 0x00, 0x00,
0x00, 0x00, 0x2A, 0x00, 0x00, 0x00, 0x00, 0x0A, 0x00, 0x00, 0x00, 0x02, 0xAA, 0x
```

```
A0, 0x00, 0x00, 0x02, 0x22, 0xA0, 0x00, 0x00, 0x03, 0xFE, 0x80, 0x00, 0x00, 0x02,
0xA2, 0x00, 0x00, 0x00, 0x01, 0xEA, 0x80, 0x00, 0x00, 0x00, 0x02, 0x80, 0x00, 0x
00, 0x00, 0xFF, 0x00, 0x00, 0x00, 0x00, 0x82, 0x00, 0x00, 0x00, 0x00, 0x7A, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3C, 0x00, 0x00, 0x00, 0x00, 0x
00, 0x00, 0x00, 0x00, 0x00, 0x18, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00};
```

[illegible][illegible][illegible]

[illegible][illegible][illegible]

```
const unsigned char PROGMEM ar7 [] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0  
x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,  
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0  
x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,  
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,  
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,  
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,  
0x00, 0x00, 0x00, 0x76, 0x00, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x55, 0x57, 0  
xFF, 0xC0, 0x01, 0x44, 0x40, 0x11, 0x10, 0x01, 0xFF, 0xFF, 0xD5, 0x54, 0x00, 0x01
```


[illegible]

```
const unsigned char PROGMEM aL4 [] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x30, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x70, 0x00, 0x00, 0x00, 0x00, 0x20, 0x00, 0x00, 0x00, 0
x00, 0xB0, 0x00, 0x00, 0x00, 0x00, 0x80, 0x00, 0x00, 0x00, 0x01, 0xA0, 0x00, 0x00
, 0x00, 0x00, 0xA0, 0x00, 0x00, 0x00, 0x06, 0xAE, 0xAE, 0xF0, 0x00, 0x00, 0x00, 0
xA8, 0x04, 0x00, 0x0E, 0xEE, 0xBF, 0xE8, 0x00, 0x00, 0x2A, 0x08, 0xA8, 0x00, 0x0F
, 0xAA, 0xAB, 0xE8, 0x00, 0x00, 0x0A, 0xAA, 0x28, 0x00, 0x07, 0xEA, 0xAA, 0xAC, 0
x00, 0x04, 0x20, 0x00, 0x00, 0x00, 0x02, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
, 0x00, 0x00, 0x01, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x80, 0x00, 0x00, 0x00, 0x00, 0
x20, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x20, 0x00, 0x00, 0x00
, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
```

[illegible][illegible][illegible]

```
const unsigned char PROGMEM aL7 [] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,  
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,  
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0  
x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00  
, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0  
x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00  
, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0  
x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x1E, 0x00, 0x00, 0x00  
, 0x00, 0x28, 0x00, 0x00, 0x00, 0x01, 0xBA, 0x00, 0x00, 0x00, 0x02, 0xA8, 0x0A, 0
```

[illegible][illegible]

[illegible][illegible]

```
const unsigned char PROGMEM aU4 [] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x28, 0x00, 0x00, 0x00, 0x00, 0xAB, 0x00, 0x00, 0x00, 0x00, 0xAA, 0x80, 0x00
, 0x00, 0x02, 0x0A, 0x20, 0x00, 0x00, 0x0F, 0xAA, 0xB0, 0x00, 0x00, 0x20, 0x22, 0
xA0, 0x00, 0x00, 0xBB, 0xBB, 0xAA, 0x00, 0x00, 0xA8, 0x00, 0x00, 0x00, 0x00, 0x00
, 0xBE, 0x00, 0x00, 0x00, 0x00, 0xA0, 0x00, 0x00, 0x00, 0x00, 0xBE, 0x00, 0x00, 0
x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xEE, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
, 0x00, 0x00, 0x00, 0xEE, 0x00, 0x00, 0x00, 0x00, 0x20, 0x00, 0x00, 0x00, 0x00, 0
```

```
xFE, 0x00, 0x00, 0x00, 0x00, 0xA0, 0x00, 0x00, 0x00, 0x00, 0xAE, 0x00, 0x00, 0x00
, 0x00, 0x28, 0x00, 0x00, 0x00, 0x00, 0xBA, 0x00, 0x00, 0x00, 0x00, 0xA8, 0x00, 0
x00, 0x00, 0x00, 0xBE, 0x00, 0x00, 0x00, 0x00, 0x80, 0x00, 0x00, 0x00, 0x00, 0x3E
, 0x00, 0x00, 0x00, 0x00, 0x42, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x00, 0x00};
```

```
const unsigned char PROGMEM aU5 [] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
, 0x00, 0x00, 0x00, 0x00, 0x28, 0x00, 0x00, 0x00, 0x00, 0xA2, 0x00, 0x00, 0x00, 0
x01, 0xFE, 0x80, 0x00, 0x00, 0x02, 0x00, 0x80, 0x00, 0x00, 0x03, 0xFF, 0xE0, 0x00
, 0x00, 0x08, 0xA0, 0xA0, 0x00, 0x00, 0x3E, 0xFE, 0xE8, 0x00, 0x00, 0x20, 0x00, 0
x08, 0x00, 0x00, 0x6E, 0xEF, 0xFA, 0x00, 0x00, 0x00, 0x02, 0x00, 0x00, 0x00, 0x00
, 0xBE, 0x00, 0x00, 0x00, 0x00, 0xA2, 0x00, 0x00, 0x00, 0x00, 0xBA, 0x00, 0x00, 0
x00, 0x00, 0x02, 0x00, 0x00, 0x00, 0x00, 0xEE, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
, 0x00, 0x00, 0x00, 0xEE, 0x00, 0x00, 0x00, 0x00, 0x20, 0x00, 0x00, 0x00, 0x00, 0
xFE, 0x00, 0x00, 0x00, 0x00, 0x80, 0x00, 0x00, 0x00, 0xFE, 0x00, 0x00, 0x00
, 0x00, 0x80, 0x00, 0x00, 0x00, 0x00, 0x6E, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x00, 0x00, 0x6E, 0x00, 0x00, 0x00, 0x00, 0x0A, 0x00, 0x00, 0x00, 0x00, 0x38
, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x00, 0x00, 0x00};
```

```
const unsigned char PROGMEM aU6 [] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x00, 0x00, 0x00};
```

```
const unsigned char PROGMEM aU7 [] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x00, 0x00, 0x00, 0x00, 0x18, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
, 0x00, 0x3C, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x7A, 0x00, 0
x00, 0x00, 0x00, 0x82, 0x00, 0x00, 0x00, 0x00, 0xFF, 0x00, 0x00, 0x00, 0x00, 0x02
, 0x80, 0x00, 0x00, 0x01, 0xEA, 0x80, 0x00, 0x00, 0x02, 0xA2, 0x00, 0x00, 0x00, 0
```


[illegible][illegible]

```
const unsigned char PROGMEM dead [] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0, 0x3C, 0xE3, 0x87, 0x80, 0x00, 0x24, 0x86, 0x44, 0x80, 0x00, 0x24, 0x84, 0x44,
0x80, 0x00, 0x24, 0xE4, 0x44, 0x80, 0x00, 0x24, 0x87, 0xC4, 0x80, 0x00, 0x24, 0x8
4, 0x44, 0x80, 0x00, 0x3C, 0xE4, 0x47, 0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0
0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3E, 0x00, 0x00, 0x00, 0x00, 0x7F,
0x00, 0x00, 0x00, 0x00, 0xFF, 0x00, 0x00, 0x00, 0x00, 0xFF, 0x80, 0x00, 0x00, 0x0
1, 0xC7, 0x80, 0x00, 0x00, 0x01, 0xFF, 0x80, 0x00, 0x00, 0x03, 0xFF, 0xC0, 0x00,
0x00, 0x03, 0xAD, 0xC0, 0x00, 0x00, 0x03, 0xFF, 0xC0, 0x00, 0x00, 0x03, 0xFF, 0xC
0, 0x00, 0x00, 0x03, 0x5B, 0x40, 0x00, 0x00, 0x03, 0xFF, 0xC0, 0x00, 0x00, 0x07,
0xFF, 0xE0, 0x00, 0x00, 0x07, 0x76, 0xE0, 0x00, 0x00, 0x07, 0xFF, 0xE0, 0x00, 0x0
0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00};
```

```
const unsigned char PROGMEM evil [] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x0
7, 0x80, 0x00, 0x00, 0xE0, 0x07, 0x80, 0x00, 0x00, 0xE0, 0x07, 0x80, 0x00, 0x00,
0xF0, 0x07, 0x80, 0x7F, 0xC0, 0xF0, 0x0F, 0x83, 0xFF, 0xEF, 0xF0, 0x0F, 0xFF, 0xF
F, 0xFF, 0xF0, 0x0F, 0xFF, 0x9C, 0xFF, 0xF0, 0x0F, 0xFF, 0xFF, 0xFF, 0xFF, 0xF0, 0x07,
0xFF, 0xFF, 0xE0, 0x00, 0x00, 0x03, 0xFF, 0xE0, 0x00, 0x00, 0x03, 0xFF, 0xE0, 0x0
0, 0x00, 0x03, 0xFF, 0xE0, 0x00, 0x00, 0x00, 0xC0, 0xC0, 0x00, 0x00, 0x00, 0xEE,
0xC0, 0x00, 0x00, 0x00, 0xFF, 0xC0, 0x00, 0x00, 0x00, 0xFF, 0xC0, 0x00, 0x00, 0x0
0, 0x0E, 0x00, 0x00, 0x00, 0x00, 0x0E, 0x00, 0x00, 0x0F, 0xD0, 0x0E, 0x10, 0x7E,
0x0F, 0xD0, 0x0E, 0x10, 0x7E, 0x00, 0xFF, 0xFF, 0xFF, 0xE0, 0x0F, 0xC4, 0x0E, 0x0
2, 0x7E, 0x0F, 0xC4, 0x0E, 0x02, 0x7E, 0x00, 0x00, 0x0E, 0x00, 0x00, 0x00, 0x00,
0x0E, 0x00, 0x00, 0x00, 0x00, 0x0E, 0x00, 0x00, 0x00, 0x00, 0x0E, 0x00, 0x00, 0x0
0, 0x0F, 0xFF, 0xFE, 0x00, 0x00, 0x0F, 0xFF, 0xFE, 0x00, 0x00, 0x0E, 0x00, 0x0E,
0x00, 0x00, 0x0E, 0x00, 0x0E, 0x00, 0x0E, 0x00, 0x0E, 0x00, 0x0E, 0x00, 0x0E, 0x0
0, 0x0F, 0x80, 0x00, 0x3E, 0x00, 0x0F, 0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00};
```

```
const unsigned char PROGMEM evilattack [] = {0x00, 0x46, 0x48, 0x48, 0x00, 0x0C,
0x46, 0x51, 0x8C, 0x30, 0x14, 0x56, 0x76, 0x9C, 0x26, 0x2A, 0x26, 0x52, 0x9C, 0x4
6, 0x05, 0x66, 0xD2, 0x9E, 0x9C, 0x03, 0xA7, 0xFF, 0xFF, 0x25, 0x01, 0x73, 0xEF,
0x7F, 0x2F, 0x44, 0xEF, 0xFF, 0xFE, 0x7C, 0x38, 0x71, 0xFF, 0xF2, 0xD4, 0x26, 0x6
8, 0xF7, 0xC7, 0xA8, 0x3E, 0x30, 0xDF, 0xC7, 0x30, 0x13, 0xC8, 0xFF, 0xCC, 0x60,
0x29, 0xFE, 0xFB, 0xD0, 0xE8, 0x24, 0x70, 0xFF, 0xE6, 0xE0, 0x02, 0x08, 0xFF, 0xC
5, 0xC0, 0x05, 0x66, 0xFF, 0x43, 0x08, 0x00, 0xC0, 0xF7, 0xCD, 0x40, 0x00, 0x60,
0xFF, 0xC2, 0x00, 0x00, 0x03, 0xFF, 0xC2, 0x88, 0x10, 0x17, 0xBF, 0xE2, 0x00, 0x0
0, 0x1F, 0xEF, 0xF0, 0x00, 0x10, 0x1E, 0xFF, 0xFC, 0x00, 0x04, 0x3C, 0xF3, 0xFC,
0x00, 0x00, 0x38, 0xFF, 0xDE, 0x00, 0x01, 0x30, 0xFF, 0xCE, 0x02, 0x0F, 0xB3, 0xF
F, 0xEF, 0xF0, 0x0F, 0xFF, 0xFF, 0xFF, 0xF0, 0x0F, 0xFF, 0x9C, 0xFF, 0xF0, 0x0F,
0xFF, 0xFF, 0xFF, 0xF0, 0x0F, 0xFF, 0xFF, 0xE3, 0xF0, 0x0F, 0xF3, 0xB6, 0x63, 0xF
4, 0x0F, 0xF3, 0xB6, 0x62, 0xF8, 0x4F, 0xF3, 0x80, 0x60, 0xF8, 0x0F, 0xD2, 0x80,
0x44, 0xF9, 0x1F, 0x80, 0x80, 0x45, 0xF8, 0x1F, 0x80, 0xDA, 0xC0, 0x78, 0x9F, 0xC
0, 0xFF, 0xC0, 0x78, 0x1F, 0x08, 0x0E, 0x10, 0x60, 0x00, 0x00, 0x00, 0x0A, 0x02,
0x00, 0x80, 0x80, 0x00, 0x00};
```

```
const unsigned char PROGMEM evilshoot [] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0
x00, 0x40, 0x40, 0x88, 0x10, 0x10, 0x00, 0x02, 0x00, 0x00, 0x00, 0x04, 0x00, 0x60
, 0x17, 0x82, 0x00, 0x00, 0xE0, 0x07, 0x80, 0x00, 0x30, 0xE0, 0x07, 0x80, 0x00, 0
x00, 0xF0, 0x07, 0x80, 0x7F, 0xC0, 0xF2, 0x0F, 0x83, 0xFF, 0xEF, 0xF0, 0x0F, 0xFF
, 0xFF, 0xFF, 0xF0, 0x0F, 0xFF, 0x9C, 0xFF, 0xF0, 0x0F, 0xFF, 0xFF, 0xFF, 0xF0, 0
x07, 0xFF, 0xFF, 0xE0, 0x00, 0x00, 0x03, 0x80, 0x60, 0x04, 0x00, 0x03, 0x80, 0x60
, 0x00, 0x40, 0xA3, 0x80, 0x64, 0x20, 0x00, 0x02, 0x80, 0x40, 0x00, 0x04, 0x00, 0
x80, 0x41, 0x00, 0x00, 0x00, 0xC0, 0xC0, 0x00, 0x00, 0x40, 0xFF, 0xC0, 0x08, 0x20
, 0x08, 0x0E, 0x00, 0x00, 0x00, 0x01, 0x0E, 0x08, 0x00, 0x0F, 0xD0, 0x0E, 0x80, 0
x00, 0x0F, 0xD0, 0x0E, 0x00, 0x50, 0x20, 0xFF, 0xFF, 0xFF, 0xE0, 0x0F, 0xC4, 0x0E
```

```
, 0x02, 0x00, 0x0F, 0xC4, 0x0E, 0x02, 0x00, 0x00, 0x00, 0x4E, 0x00, 0x22, 0x00, 0x00, 0x0E, 0x08, 0x00, 0x40, 0x00, 0x0E, 0x00, 0x00, 0x08, 0x80, 0x0E, 0x00, 0x08, 0x00, 0x0F, 0xFF, 0xFE, 0x00, 0x00, 0x0F, 0xFF, 0xFE, 0x00, 0x00, 0x0E, 0x00, 0x0E, 0x00, 0x00, 0x0E, 0x00, 0x02, 0x0E, 0x10, 0x8E, 0x18, 0x00, 0x3E, 0x00, 0x0F, 0x80, 0x40, 0x3E, 0x20, 0x0F, 0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00};
```

```
// Declaration for an SSD1306 display connected to I2C (SDA, SCL pins)
```

```
// เริ่มการทำงานของ OLED
```

```
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);
```

```
// ตั้งค่าตัวแปรที่จะใช้
```

```
int s = 0;
```

```
// เอาไว้เก็บ ค่าที่อ่านได้ของ จอยอนาล็อก joy[0] = แกน x, joy[1]= แกน y
```

```
int joy[2];
```

```
// pin 9 , ลำโพง
```

```
int buzzer = 9;
```

```
// pin 3,4 สำหรับสวิตช์ซ้ายขวา และ pin 6,7 สำหรับ led ที่ติดกับมัน
```

```
int swl=4;
```

```
int swr=3;
```

```
int led_left = 6;
```

```
int led_right = 7;
```

```
// pin 10,12 สำหรับ tilt switch กับ led ที่ติดกับมัน
```

```
int tilt = 10;
```

```
int tilt_led = 12;
```

```
// พิกัดมุมซ้ายบนของเฟรมที่จะวาดลูกศร
```

```
int xx = 40;
```

```
int yy = 10 ;
```

```
// ตัวแปรที่ใช้เก็บข้อมูลระหว่างเล่นเกมส์
```

```
bool gameover = false;
```

```
int state=0;
```

```
int level = 0;
```

```
unsigned long start_time;
```

```
// function declaration
```

```
// ให้แสดงผล ลูกศร ลง ขึ้น ซ้าย ขวา ทางหน้าจอ และเล่นเสียงเพลงที่สอดคล้อง
```

```
void playdown(void);
```

```

void playright(void);
void playup(void);
void playleft(void);
// ไว้ตรวจจับว่ากำลัง โยกอนาล็อค ไปทิศไหน
bool isdown(void);
bool isright(void);
bool isleft(void);
bool isup(void);
void readjoy(void);
// เล่นเสียงผ่านลำโพง
// buzz(pin ของลำโพง,frequency,duration) เอามา
จาก https://bgsu.instructure.com/courses/1157282/pages/week-5-day-2-sound
void buzz(int , long , long ) ;
// เล่นเสียง เมื่อผู้เล่นกดลูกศรถูก
void playsong(void);
// เช็คว่ากำลังกดปุ่ม ซ้ายหรือขวา อยู่รึเปล่า
void click(void);
int read_state(void);
// ทำของ boss ด้านสอง สามท่า อิงซ้าย อิงขวา และท่าพิเศษ รับ argument หนึ่งค่าคือความเร็วการโจมตี ซึ่งจะยกขึ้นเรื่อยๆ
void bulletleft(int);
void bulletright(int);
void specialattack(int);

void setup() {

    Serial.begin(9600);

    // กำหนด pin ต่างๆ
    pinMode(buzzer,OUTPUT);
    pinMode(sw1,INPUT_PULLUP);
    pinMode(sw2,INPUT_PULLUP);
    pinMode(tilt,INPUT);
    pinMode(motor,OUTPUT);
    pinMode(led_left,OUTPUT);
    pinMode(led_right,OUTPUT);

    // หากไม่สามารถ เริ่มใช้งาน OLED ได้จะขึ้น error
    if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) { // Address 0x3D for 128x64
        Serial.println(F("SSD1306 allocation failed"));
        for(;;);
    }
    delay(2000);
}

```

```

void loop() {

    int count = 0;
    // หน้าจอเริ่มเกมส์ จะไปต่อได้ ก็ต่อเมื่อกด ปุ่มซ้าย (sw1-switch left)
    while (digitalRead(sw1)){
        display.clearDisplay();
        display.setCursor(10,10);
        display.setTextSize(1.5);
        display.println("2110682");
        display.println("FINAL PROJECT");
        display.println("PRESS LEFT BUTTON");
        display.display();
    }
    delay(1000);

    // tilt to continue (ให้โยก tilt switch ถึงจะไปต่อได้)
    while (digitalRead(tilt)){
        digitalWrite(tilt_led,LOW);
        display.clearDisplay();
        display.setCursor(10,10);
        display.setTextSize(1);
        display.println("TILT TO CONTINUE");
        display.display();
    }
    // ถ้าโยก tilt switch หลอดไฟจะกระพริบ
    display.clearDisplay();
    digitalWrite(tilt_led,HIGH);
    delay(2000);
    digitalWrite(tilt_led,LOW);

    // คำนวณ 1 , random ลูกศร

    long start_time = millis();

    while(millis()-start_time<30000){
        int olds = s;
        switch(s) {
            case 0:
                playdown();
                break;
            case 1:
                playright();

```

```

        break;
    case 2:
        playleft();
        break;
    case 3:
        playup();
        break;
    }
    buzz(9,5000,50);
    count ++;
    while (s==olds){
        s = random(0,4);
    }
}
// แสดงคะแนนเมื่อจบด่าน 1
display.clearDisplay();
display.drawRoundRect(10, 10, 108, 44,5, WHITE);
display.setCursor(20,32);
display.display();
for (int i=0; i <50;i++){
    buzz(9,500-50*i,100);
}
display.print("SCORE : ");
display.print(count);
display.display();
delay(5000);

// อนิเมชั่นเข้าด่าน 2
display.clearDisplay();
for (int i =0; i<7;i++){
    display.drawRect(1+5*i,1+5*i,127-10*i,63-10*i,WHITE);
    display.display();
    buzz(9,100+100*i,800);
}
delay(1000);

while(digitalRead(tilt)){

    display.clearDisplay();
    display.drawBitmap(xx,yy,evil,40,40, 1);
    display.display();
    delay(1000);
    buzz(9,50,1000);
    display.setCursor(20,55);

```

```

        display.println("VS ANGRY BOSS");
        display.display();
        buzz(9,100,1000);
        buzz(9,200,1000);
    }
    // คำนวณ 2
    long stage2_time = millis();
    while(!gameover && millis()-stage2_time<60000){
        int oldss = s;
        switch (s)
        {
            case 0:
                bulletleft(level);
                if (read_state()==0){
                    gameover = true;
                    level = 0;
                }
                break;
            case 1:
                bulletright(level);
                if (read_state()==1){
                    gameover = true;
                    level = 0;
                }
                break;
            case 2:
                specialattack(level);
                if (digitalRead(tilt)){
                    gameover = true;
                    level = 0;
                }
                digitalWrite(tilt_led,HIGH);
                delay(1000);
                digitalWrite(tilt_led,LOW);
                break;
        }
        level++;
        while (oldss == s)
            s = random(0,3);
    }

    // แสดงผล dead ถ้าแพ้
    if (gameover) {
        display.clearDisplay();
        display.drawBitmap(xx,yy,dead,40,40,WHITE);
    }

```

```

        display.display();
    for (int i=0; i <50;i++){
        buzz(9,500-50*i,100);
    }
    delay(3000);
}
// แสดงผล you won ถ้านะ
else {
    display.clearDisplay();
    display.drawRoundRect(10, 10, 108, 44,5, WHITE);
    display.setCursor(20,32);
    display.display();
    for (int i=0; i <50;i++){
        buzz(9,500-50*i,100);
    }
    display.println("YOU WON!!");
    display.display();
    delay(5000);
}
gameover = false;
}

void playdown() {
    while(!isdown()){
        display.clearDisplay();
        display.drawBitmap(xx, yy,down1,40,40, 1);
        display.display ();

        display.clearDisplay();
        display.drawBitmap(xx, yy,down2,40,40, 1);
        display.display ();

        display.clearDisplay();
        display.drawBitmap(xx, yy,down3,40,40, 1);
        display.display ();
        playsong();
        display.clearDisplay();
        display.drawBitmap(xx, yy,down4,40,40, 1);
        display.display ();

        display.clearDisplay();
        display.drawBitmap(xx, yy,down5,40,40, 1);
        display.display ();

        display.clearDisplay();

```



```

        display.drawBitmap(xx, yy,down6,40,40, 1);
        display.display ();
        playsong();
        display.clearDisplay();
        display.drawBitmap(xx, yy,down7,40,40, 1);
        display.display ();

        display.clearDisplay();
        display.drawBitmap(xx, yy,down8,40,40, 1);
        display.display ();
        playsong();
    }
}

```

```

void playright(){
    while(!isright()){

        display.clearDisplay();
        display.drawBitmap(xx, yy,aR0,40,40, 1);
        display.display();

        display.clearDisplay();
        display.drawBitmap(xx, yy,aR1,40,40, 1);
        display.display();

        display.clearDisplay();
        display.drawBitmap(xx, yy,aR2,40,40, 1);
        display.display();
        playsong();
        display.clearDisplay();
        display.drawBitmap(xx, yy,aR3,40,40, 1);
        display.display();

        display.clearDisplay();
        display.drawBitmap(xx, yy,aR4,40,40, 1);
        display.display();

        display.clearDisplay();
        display.drawBitmap(xx, yy,aR5,40,40, 1);
        display.display();

        playsong();
        display.clearDisplay();
        display.drawBitmap(xx, yy,aR6,40,40, 1);
        display.display();
    }
}

```

```

        display.clearDisplay();
        display.drawBitmap(xx, yy,aR7,40,40, 1);
        display.display();
    }

}

void playleft(){
    while( !isleft()){

        display.clearDisplay();
        display.drawBitmap(xx, yy,aL0,40,40, 1);
        display.display();

        display.clearDisplay();
        display.drawBitmap(xx, yy,aL1,40,40, 1);
        display.display();

        display.clearDisplay();
        display.drawBitmap(xx, yy,aL2,40,40, 1);
        display.display();
        playsong();
        display.clearDisplay();
        display.drawBitmap(xx, yy,aL3,40,40, 1);
        display.display();

        display.clearDisplay();
        display.drawBitmap(xx, yy,aL4,40,40, 1);
        display.display();

        display.clearDisplay();
        display.drawBitmap(xx, yy,aL5,40,40, 1);
        display.display();
        playsong();
        display.clearDisplay();
        display.drawBitmap(xx, yy,aL6,40,40, 1);
        display.display();

        display.clearDisplay();
        display.drawBitmap(xx, yy,aL7,40,40, 1);
        display.display();
        playsong();
    }
}

```

```

    }

}

void playup(){
    while(!isup()){

        display.clearDisplay();
        display.drawBitmap(xx, yy,aU0,40,40, 1);
        display.display();

        display.clearDisplay();
        display.drawBitmap(xx, yy,aU1,40,40, 1);
        display.display();

        display.clearDisplay();
        display.drawBitmap(xx, yy,aU2,40,40, 1);
        display.display();
        playsong();

        display.clearDisplay();
        display.drawBitmap(xx, yy,aU3,40,40, 1);
        display.display();

        display.clearDisplay();
        display.drawBitmap(xx, yy,aU4,40,40, 1);
        display.display();

        display.clearDisplay();
        display.drawBitmap(xx, yy,aU5,40,40, 1);
        display.display();
        playsong();

        display.clearDisplay();
        display.drawBitmap(xx, yy,aU6,40,40, 1);
        display.display();

        display.clearDisplay();
        display.drawBitmap(xx, yy,aU7,40,40, 1);
        display.display();
        playsong();

    }

}

```

```
void readjoy(){
    joy[0] = analogRead(joyX);
    joy[1] = analogRead(joyY);
    /*
    Serial.print(joy[0]);
    Serial.print("\t");
    Serial.println(joy[1]);*/
}
```

```
bool isdown(){
    readjoy();
    if (joy[1]<400)
    {
        return true;
    }
    else {
        return false;
    }
}
```

```
bool isright(){
    readjoy();
    if (joy[0]>600)
    {
        return true;
    }
    else {
        return false;
    }
}
```

```
bool isleft(){
    readjoy();
    if (joy[0]<400)
    {
        return true;
    }
    else {
        return false;
    }
}
```

```
bool isup(){
    readjoy();
    if (joy[1]>600)
```

```

    {
        return true;
    }
    else {
        return false;
    }
}

void buzz(int targetPin, long frequency, long length) {
    long delayValue = 1000000/frequency/2;
    long numCycles = frequency * length/ 1000;
    for (long i=0; i < numCycles; i++){
        digitalWrite(targetPin,HIGH);
        delayMicroseconds(delayValue);
        digitalWrite(targetPin,LOW);
        delayMicroseconds(delayValue);
    }
}

void playsong(void){
    buzz(9,pgm_read_word_near(notes+note_index),pgm_read_word_near(durations+note_index)/1.5);
    note_index= (note_index+1)%note_length;
}

int read_state(){
    if (digitalRead(led_left) == HIGH) {
        return 0;
    }
    else if (digitalRead(led_right)==HIGH) {
        return 1;
    }
    else return 2;
}

void click(){
    if(!digitalRead(sw1)){
        digitalWrite(led_left,HIGH);
        digitalWrite(led_right,LOW);

    }
    if (!digitalRead(sw2)){
        digitalWrite(led_left,LOW);
        digitalWrite(led_right,HIGH);
    }
}

```

```

void bulletleft(int speed){

    display.clearDisplay();
    display.drawBitmap(xx-32,yy,evilshoot,40,40,WHITE);
    display.display();
    buzz(9,50,500-speed*10);
    click();

    display.fillCircle(32,32,5,WHITE);
    display.display();
    buzz(9,100,500-10*level);
    click();
    display.fillCircle(32,32,10,WHITE);
    display.display();
    buzz(9,500,500-10*level);
    click();
    display.fillCircle(32,32,15,WHITE);
    display.display();
    buzz(9,1000,500-10*level);
    click();
    display.fillRect(0,0,64,64,WHITE);
    display.display();
    buzz(9,5000,1000-10*level);
    click();
}

```

```

void bulletright(int speed){
    display.clearDisplay();
    display.drawBitmap(xx+32,yy,evilshoot,40,40,WHITE);
    display.display();
    buzz(9,50,500-speed*10);
    click();

    display.fillCircle(96,32,5,WHITE);
    display.display();
    buzz(9,100,500-speed*10);
    click();

    display.fillCircle(96,32,10,WHITE);
    display.display();
    buzz(9,500,500-speed*10);
    click();

    display.fillCircle(96,32,15,WHITE);
}

```

```

    display.display();
    buzz(9,1000,500-speed*10);
    display.fillRect(64,0,64,64,WHITE);
    display.display();
    click();

    buzz(9,5000,1000-speed*10);
}

void specialattack(int speed){
    display.clearDisplay();
    display.drawBitmap(xx,yy,evilattack,40,40,WHITE);
    display.display();
    buzz(9,100,500-speed*10);
    delay(700);
    for (int j=0;j<4;j++){
        display.clearDisplay();
        for (int i =0;i<4;i++){
            display.fillTriangle(32*i,16*j, 32*i+32, 16*j, 32*i+16, 16*j+16, WHITE);
            display.display();
        }
        buzz(9,50+100*j,500-speed*40);
    }
}

```