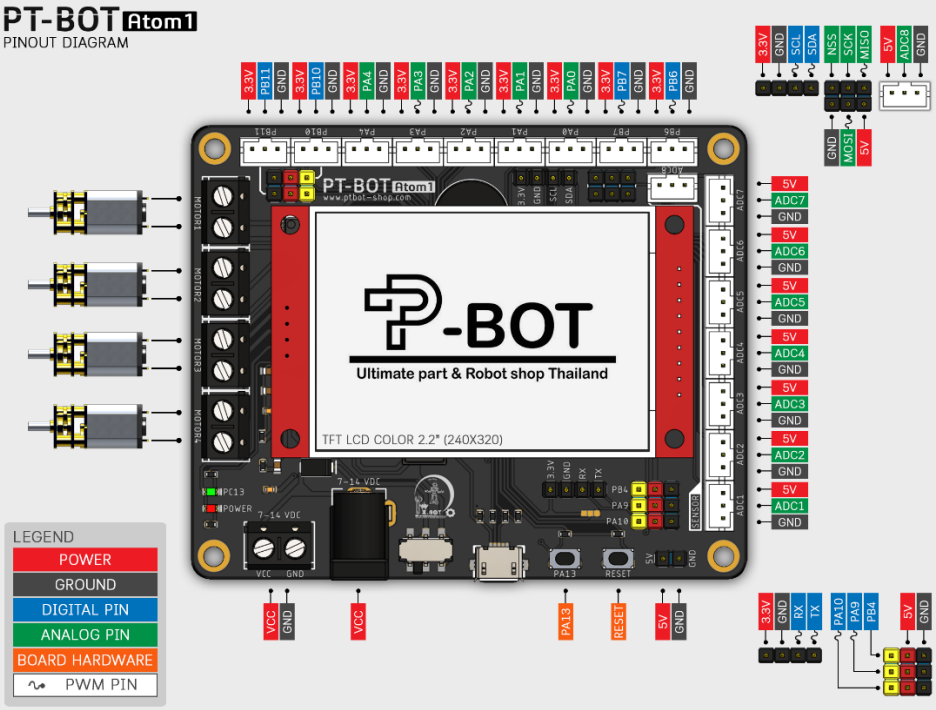
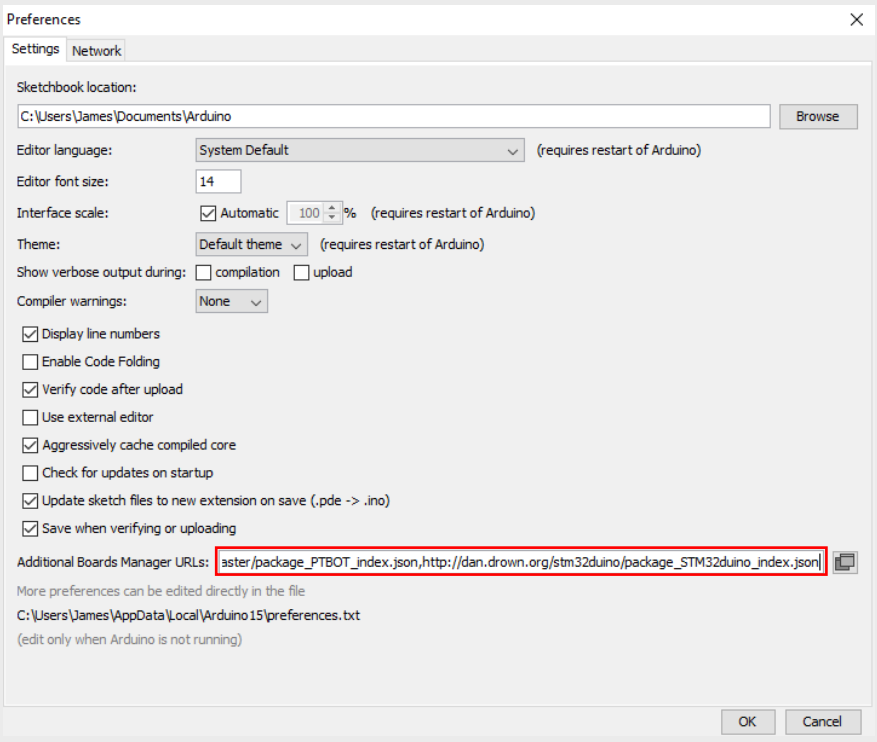


Pinout Diagram PT-BOT Atom (ATOM128)

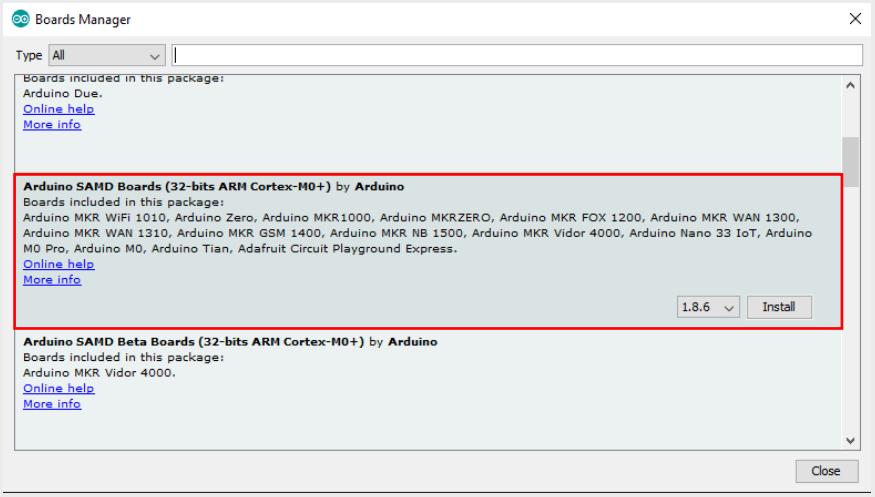
ขั้นตอนการติดตั้งบอร์ดลง Arduino IDE

เปิด Arduino IDE ขึ้นมา ทำการเพิ่ม Additional Boards Manager URL โดยเข้าไปที่ File > Preferences และ Copy URL ด้านล่างไปใส่ในช่อง

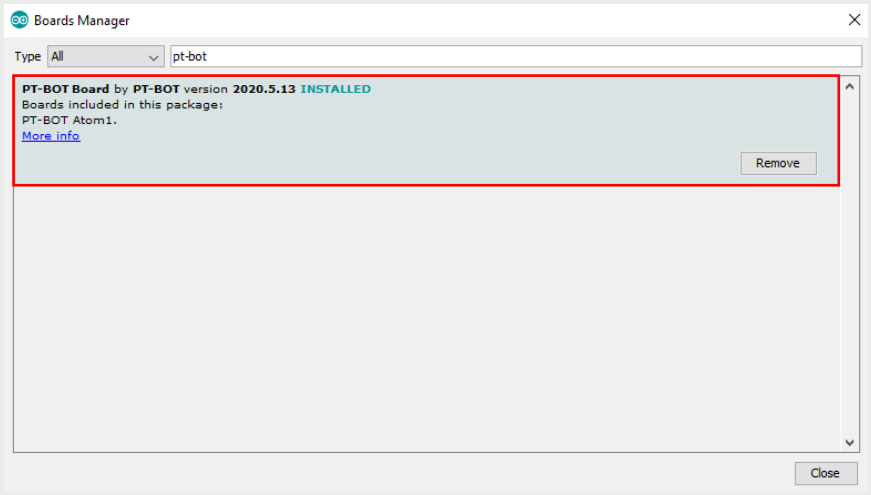
https://raw.githubusercontent.com/iBuilds/PT-BOT/master/package_PTBOT_index.json



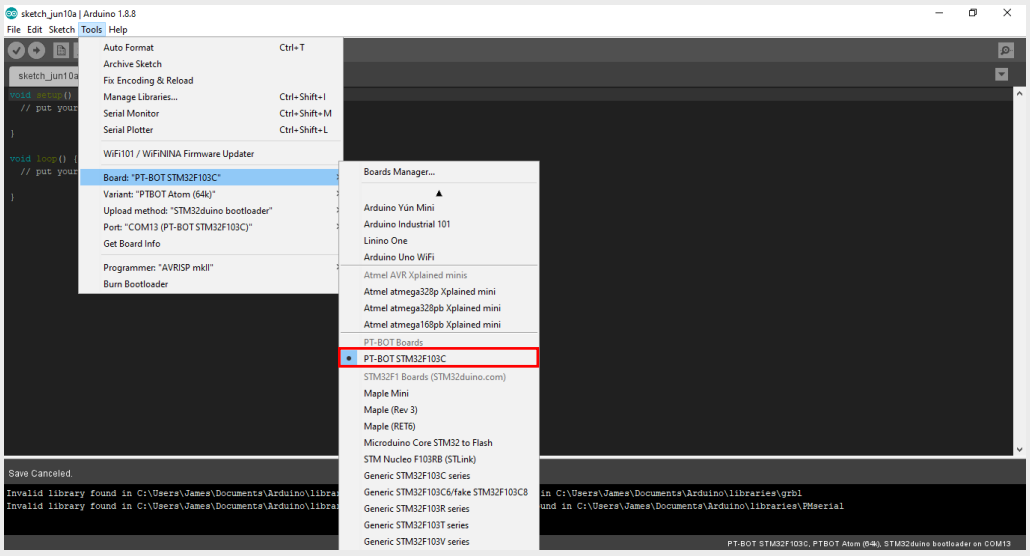
เข้าไปที่ Tools > Board > Boards Manager… และติดตั้ง Arduino SAMD Boards



ค้นหา PT-BOT และติดตั้ง PT-BOT Board



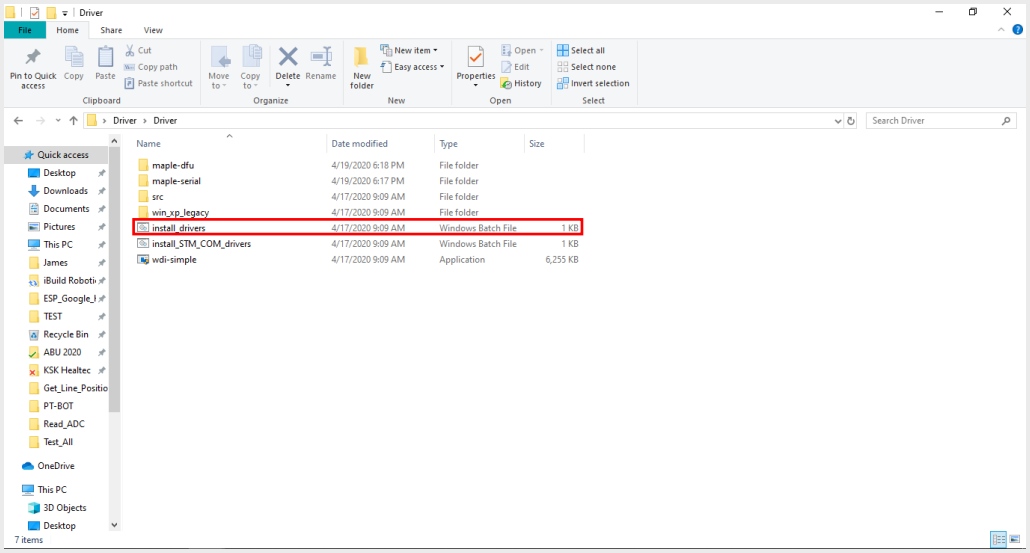
ถ้าติดตั้งสำเร็จ จะมีบอร์ดเพิ่มขึ้นมาตามรูปด้านล่าง



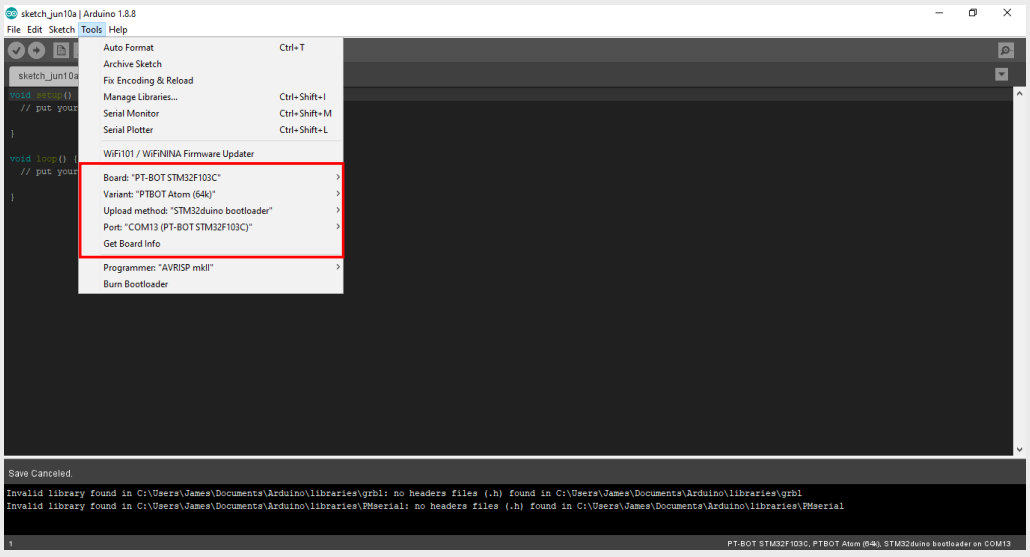
ติดตั้ง Driver สำหรับเชื่อมต่อกับบอร์ด

Link Download: https://bit.ly/2RPJeW6

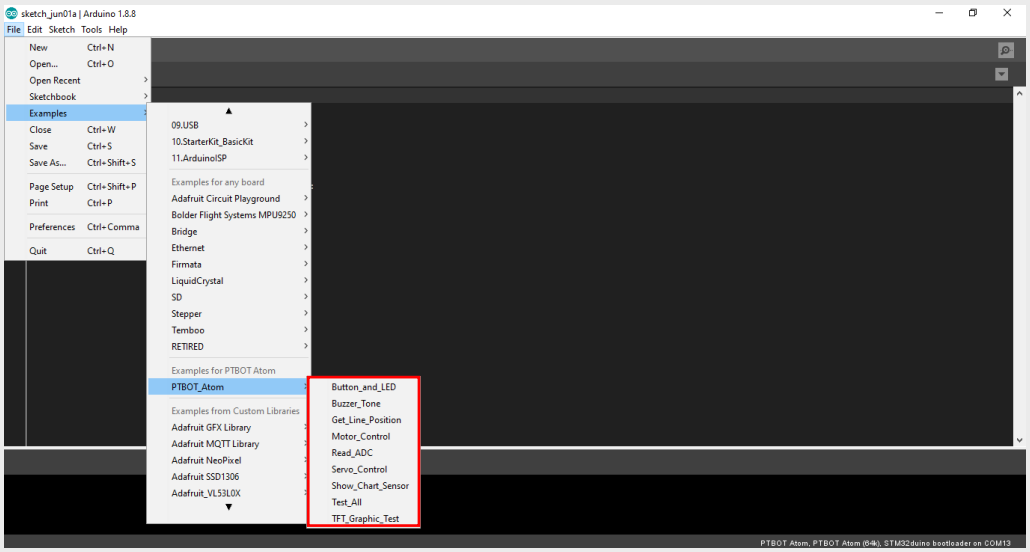
ดับเบิ้ลคลิกไฟล์ install drivers แล้วรอจนกว่าจะติดตั้งสำเร็จ



เลือก Board เป็น PT-BOT STM32F103C ในช่อง Variant เลือก PTBOT Atom (64k) สำหรับรุ่น 2 Motor PTBOT Atom (128k) สำหรับรุ่น 4 Motor ช่อง Upload method เลือกเป็น STM32duino bootloader และเลือก Port ให้ถูกต้อง



เข้าไปที่ File > Example > PTBOT_Atom จะมีตัวอย่างการใช้งานต่างๆ โดยจะมีคำอธิบาย Code ในแต่ละบรรทัด



คำสั่งต่าง ๆ สำหรับการใช้งานพื้นฐาน

```
ATOM(model);
```

ฟังก์ชั่นสำหรับเรียกใช้งาน Class และกำหนดรุ่นของบอร์ด ทุกโปรแกรมควรใส่ไว้ในส่วนก่อน void setup

model = รุ่นของบอร์ดที่ใช้งาน ATOM64 (สำหรับรุ่น 2 มอเตอร์) และ ATOM128 (สำหรับรุ่น 4 มอเตอร์)

ตัวอย่างเพิ่มเติมดูได้จากตัวอย่างโปรแกรมบน Arduino IDE

```
IOSetup();
```

ฟังก์ชั่นสำหรับเรียกใช้งาน Output และ Input ทั้งหมดบนบอร์ด ควรเรียกใช้ทุกครั้งใน void setup

```
Tone(note, time);
```

ฟังก์ชั่นสำหรับใช้งาน Buzzer

note = โน้ต หรือค่าความถี่ (Hz)

time = ระยะเวลาที่ต้องการเปล่งเสียง (millisecond)

Note สำหรับการใช้งาน Buzzer

note_C	note_D
note_E	note_F
note_G	note_A
note_B	note_CC

ตัวอย่างเพิ่มเติมดูได้จากตัวอย่างโปรแกรมบน Arduino IDE ชื่อ **Buzzer_Tone**

```
ADCRead(channel);
```

ฟังก์ชั่นสำหรับอ่านค่า Analog จากช่อง ADC1 – ADC7 ความละเอียด 10 bit มีค่าตั้งแต่ 0 – 1023

channel = ช่อง ADC ที่ต้องการอ่านค่า

ตัวอย่างเพิ่มเติมดูได้จากตัวอย่างโปรแกรมบน Arduino IDE ชื่อ **Read_ADC**

```
LINESensorSET(adcchannel[], num_sensor);
```

ฟังก์ชั่นสำหรับตั้งค่าเซ็นเซอร์ตรวจจับเส้น เพื่อรับค่าตำแหน่งบนเส้น

adcchannel = ช่อง ADC ทั้งหมดที่ต่อกับเซ็นเซอร์ตรวจจับเส้น (ใส่เป็น Array)

num_sensor = จำนวนเซ็นเซอร์ตรวจจับเส้นที่ต่อกับช่อง ADC

ตัวอย่างเพิ่มเติมดูได้จากตัวอย่างโปรแกรมบน Arduino IDE ชื่อ **Get_Line_Position**

```
LINECalibrate();
```

ฟังก์ชั่นสำหรับ Calibrate เซ็นเซอร์ตรวจจับเส้น สามารถใช้ได้ทั้งเส้นขาว และเส้นดำ ก่อนใช้งานฟังก์ชั่นนี้ควรใช้งานฟังก์ชั่น LINESensorSET() ก่อน เพื่อกำหนด PIN และจำนวนของเซ็นเซอร์

ขั้นตอนการ Calibrate เซ็นเซอร์ตรวจจับเส้น

เมื่อฟังก์ชั่น LINECalibrate() เริ่มทำงาน บนหน้าจอจะมีกราฟแสดงค่า Analog ของเซ็นเซอร์แต่ละตัว และไฟ LED บนบอร์ดจะกระพริบ ให้นำเซ็นเซอร์ทุกตัววางบนเส้นแล้วกดปุ่ม PA13 บนบอร์ด 1 ครั้ง รอจนกว่าหน้าจอจะแสดงกราฟอีกครั้ง แล้วนำเซ็นเซอร์ทุกตัววางบนพื้น กดปุ่ม PA13 อีก 1 ครั้ง แล้วรอจนกว่า LED บนบอร์ดจะดับ (การ Calibrate จะถูกบันทึกค่าลงใน EEPROM จึงสามารถทำครั้งเดียวได้)

ตัวอย่างเพิ่มเติมดูได้จากตัวอย่างโปรแกรมบน Arduino IDE ชื่อ **Get_Line_Position**

```
LINEToggle();
```

ฟังก์ชั่นสำหรับสลับระหว่างสีของเส้นกับสีของพื้น ก่อนใช้งานควรใช้งานฟังก์ชั่น LINESensorSET() ก่อน เพื่อกำหนด PIN กับจำนวนของเซ็นเซอร์ และควร Calibrate เซ็นเซอร์แล้ว

ตัวอย่างเพิ่มเติมดูได้จากตัวอย่างโปรแกรมบน Arduino IDE ชื่อ **Get_Line_Position**

```
GETPosition();
```

ฟังก์ชั่นสำหรับรับค่าตำแหน่งของเซอร์ตรวจจับเส้น ก่อนใช้งานควรใช้งานฟังก์ชั่น LINESensorSET() ก่อน เพื่อกำหนด PIN กับจำนวนของเซ็นเซอร์ และควร Calibrate เซ็นเซอร์แล้ว

ตัวอย่างเพิ่มเติมดูได้จากตัวอย่างโปรแกรมบน Arduino IDE ชื่อ **Get_Line_Position**

```
LINEChartSET(adcchannel[], num_sensor);
```

ฟังก์ชั่นสำหรับตั้งค่าเซ็นเซอร์ตรวจจับเส้น ที่ต้องการให้แสดงบนหน้าจอเป็นกราฟ ถ้าไม่มีการเรียกใช้งานฟังก์ชั่นนี้ เซ็นเซอร์ทุกตัวที่ถูกใช้งานในฟังก์ชั่น LINESensorSET() จะแสดงทั้งหมด

adcchannel = ช่อง ADC ที่ต่อกับเซ็นเซอร์ตรวจจับเส้นที่ต้องการให้แสดง (ใส่เป็น Array)

num_sensor = จำนวนเซ็นเซอร์ตรวจจับเส้นที่ต้องการให้แสดง

ตัวอย่างเพิ่มเติมดูได้จากตัวอย่างโปรแกรมบน Arduino IDE ชื่อ **Show_Chart_Sensor**

```
LINEChartSHOW(x);
```

ฟังก์ชั่นสำหรับแสดงกราฟของเซ็นเซอร์บนหน้าจอ

x = ตำแหน่งในแกน X บนหน้าจอที่ต้องการให้แสดงกราฟ

ตัวอย่างเพิ่มเติมดูได้จากตัวอย่างโปรแกรมบน Arduino IDE ชื่อ **Show_Chart_Sensor**

```
servoWrite(pin, degree);
```

ฟังก์ชั่นสำหรับสั่งงานเซอร์โวมอเตอร์

pin = PIN ที่ต่อกับเซอร์โวมอเตอร์

degree = องศาที่ต้องการสั่งให้เซอร์โวมอเตอร์หมุน

ตัวอย่างเพิ่มเติมดูได้จากตัวอย่างโปรแกรมบน Arduino IDE ชื่อ **Servo_Control**

```
motorWrite(motor, speed);
```

ฟังก์ชั่นสำหรับสั่งงานมอเตอร์

motor = ช่องที่ต่อกับมอเตอร์

speed = ความเร็วและทิศทางที่ต้องการสั่งให้มอเตอร์หมุน (-255 ถึง 255)

ตัวอย่างเพิ่มเติมดูได้จากตัวอย่างโปรแกรมบน Arduino IDE ชื่อ **Motor_Control**

ฟังก์ชั่นสำหรับใช้งานหน้าจอ ตัวอย่างเพิ่มเติมดูได้จากตัวอย่างชื่อ TFT_Graphic_Test

สีต่าง ๆบนหน้าจอที่สามารถใช้งานได้

BLACK	WHITE	GREY	LIGHT_GREY
GREEN	LIME	BLUE	RED
AQUA	YELLOW	MAGENTA	CYAN
DARK_CYAN	ORANGE	PINK	BROWN
VIOLET	SILVER	GOLD	NAVY
MAROON	PURPLE	OLIVE	

```
setRotation(rotation);
```

ฟังก์ชั่นสำหรับหมุนหน้าจอ

rotation = ทิศทางของหน้าจอ (0 - 3)

```
setTextSize(size);
```

ฟังก์ชั่นสำหรับปรับขนาดตัวอักษร

size = ขนาดของตัวอักษร

```
fillScreen(color);
```

ฟังก์ชั่นสำหรับเปลี่ยนสีพื้นหลังหน้าจอ

color = สีพื้นหลังหน้าจอ

```
setTextColor(text_color);
```

ฟังก์ชั่นสำหรับเปลี่ยนสีตัวอักษร

text_color = สีตัวอักษร

```
setTextColor(text_color, background);
```

ฟังก์ชั่นสำหรับเปลี่ยนสีตัวอักษร และสีพื้นหลังตัวอักษร

text_color = สีตัวอักษร

background = สีพื้นหลังตัวอักษร

```
setCursor(x, y);
```

ฟังก์ชั่นสำหรับเลือกตำแหน่ง Cursor บนหน้าจอ

x = ตำแหน่งในแกน X

y = ตำแหน่งในแกน Y

```
print(string, x, y, text_color);
```

ฟังก์ชั่นสำหรับพิมพ์ข้อความบนหน้าจอ แบบกำหนดตำแหน่ง และสีตัวอักษร

string = ข้อความที่ต้องการพิมพ์

x = ตำแหน่งในแกน X

y = ตำแหน่งในแกน Y

text_color = สีตัวอักษร

```
print(string, x, y, text_color, background);
```

ฟังก์ชั่นสำหรับพิมพ์ข้อความบนหน้าจอ แบบกำหนดตำแหน่ง สีตัวอักษร และสีพื้นหลังตัวอักษร

string = ข้อความที่ต้องการพิมพ์

x = ตำแหน่งในแกน X

y = ตำแหน่งในแกน Y

text_color = สีตัวอักษร

background = สีพื้นหลังตัวอักษร

```
println(string, text_color);
```

ฟังก์ชั่นสำหรับพิมพ์ข้อความบนหน้าจอ แบบขึ้นบรรทัดใหม่ และกำหนดสีตัวอักษร

string = ข้อความที่ต้องการพิมพ์

text_color = สีตัวอักษร

```
printNumber(num, x, y, number_color);
```

ฟังก์ชั่นสำหรับพิมพ์ตัวเลขบนหน้าจอ แบบกำหนดตำแหน่ง และสีของตัวเลข

num = ตัวเลขที่ต้องการพิมพ์

x = ตำแหน่งในแกน X

y = ตำแหน่งในแกน Y

number_color = สีตัวเลข

```
clearScreen();
```

ฟังก์ชั่นสำหรับล้างหน้าจอทั้งหมด

```
drawPixel(x, y, color);
```

ฟังก์ชั่นสำหรับวาดจุดบนหน้าจอ

x = ตำแหน่งในแกน X

y = ตำแหน่งในแกน Y

color = สีของจุด

```
drawRect(x, y, width, height, color);
```

ฟังก์ชั่นสำหรับวาดรูปสี่เหลี่ยมบนหน้าจอ

x = ตำแหน่งในแกน X

y = ตำแหน่งในแกน Y

width = ความกว้างของรูปสี่เหลี่ยม

height = ความสูงของรูปสี่เหลี่ยม

color = สีของเส้นที่ใช้วาดรูปสี่เหลี่ยม

```
drawFillRect(x, y, width, height, color);
```

ฟังก์ชั่นสำหรับวาดรูปสี่เหลี่ยมแบบทึบบนหน้าจอ

x = ตำแหน่งในแกน X

y = ตำแหน่งในแกน Y

width = ความกว้างของรูปสี่เหลี่ยม

height = ความสูงของรูปสี่เหลี่ยม

color = สีของรูปสี่เหลี่ยม

```
drawLine(x0, y0, x1, y1, color);
```

ฟังก์ชั่นสำหรับวาดเส้นบนหน้าจอ

x0 = ตำแหน่งในแกน X จุดเริ่มต้น

y0 = ตำแหน่งในแกน Y จุดเริ่มต้น

x1 = ตำแหน่งในแกน X จุดสิ้นสุด

y1 = ตำแหน่งในแกน Y จุดสิ้นสุด

color = สีของเส้น

```
drawCircle(x, y, radius, color);
```

ฟังก์ชั่นสำหรับวาดรูปวงกลมบนหน้าจอ

x = ตำแหน่งในแกน X

y = ตำแหน่งในแกน Y

radius = รัศมีของรูปวงกลม

color = สีของเส้นที่ใช้วาดรูปวงกลม

```
drawFillCircle(x, y, radius, color);
```

ฟังก์ชั่นสำหรับวาดรูปวงกลมแบบทึบบนหน้าจอ

x = ตำแหน่งในแกน X

y = ตำแหน่งในแกน Y

radius = รัศมีของรูปวงกลม

color = สีของรูปวงกลม

```
drawTriangle(x0, y0, x1, y1, x2, y2, color);
```

ฟังก์ชั่นสำหรับวาดรูปสามเหลี่ยมบนหน้าจอ

x0 = ตำแหน่งในแกน X จุดเริ่มต้น

y0 = ตำแหน่งในแกน Y จุดเริ่มต้น

x1 = ตำแหน่งในแกน X จุดที่ 2

y1 = ตำแหน่งในแกน Y จุดที่ 2

x2 = ตำแหน่งในแกน X จุดสิ้นสุด

y2 = ตำแหน่งในแกน Y จุดสิ้นสุด

color = สีของเส้นที่ใช้วาดรูปสามเหลี่ยม

```
drawFillTriangle(x0, y0, x1, y1, x2, y2, color);
```

ฟังก์ชั่นสำหรับวาดรูปสามเหลี่ยมแบบทึบบนหน้าจอ

x0 = ตำแหน่งในแกน X จุดเริ่มต้น

y0 = ตำแหน่งในแกน Y จุดเริ่มต้น

x1 = ตำแหน่งในแกน X จุดที่ 2

y1 = ตำแหน่งในแกน Y จุดที่ 2

x2 = ตำแหน่งในแกน X จุดสิ้นสุด

y2 = ตำแหน่งในแกน Y จุดสิ้นสุด

color = สีของรูปสามเหลี่ยม

การแก้ปัญหาเบื้องต้น

ถ้าขณะทำการ Upload Code ลงบอร์ด แล้วสาย USB หลุดทำให้ Upload ไม่สำเร็จ อาจทำให้เกิดปัญหาไม่เจอ Port ให้แก้โดยการเสียบสาย USB และกด Upload เมื่อขึ้นข้อความ Resetting to bootloader via DTR pulse ให้กดปุ่ม Reset บนบอร์ด 1 ครั้ง