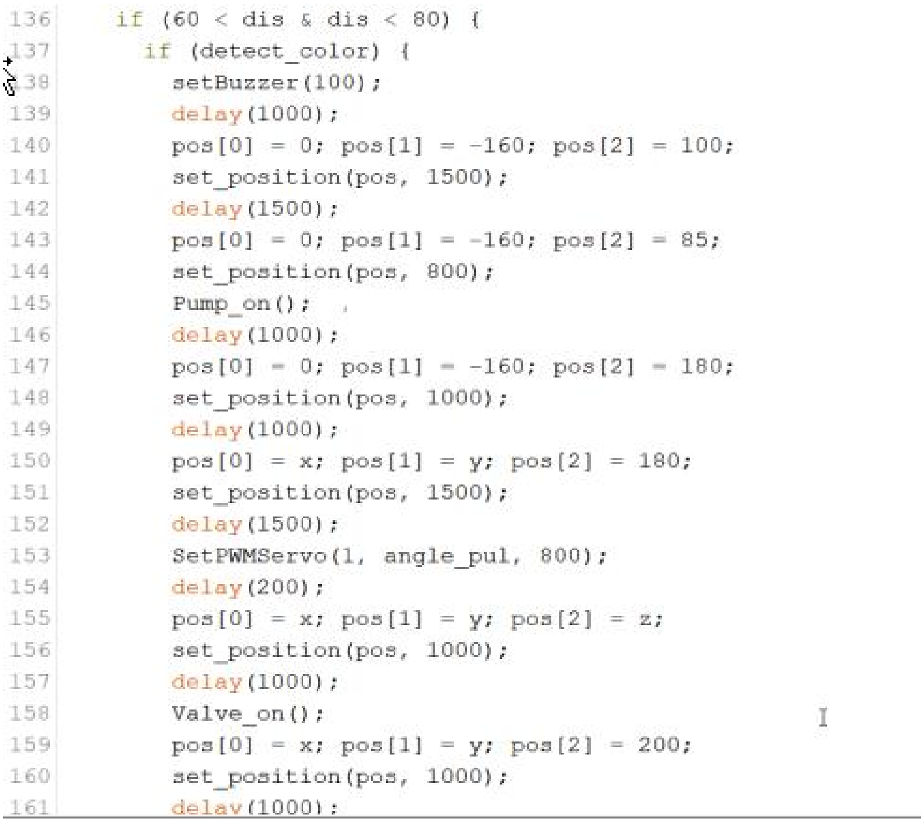
Lesson 2 Color Sorting

1. Working Principle

The color sensor is a sensor with various functions such as recognizing the color of objects, detecting the brightness of the surrounding environment, realizing object proximity detection and non-contact gesture detection, etc.

This lesson will use the color sensor for recognizing and comparing the RGB value of object. These sensors generally determine whether the detected color is consistent with the set one by using RGB (red, green, and blue) LED light sources and the ratio of R, G, and B wavelengths in the reflected light of an object.

The path of the program file: “6.Secondary Development/Sensor-extension Game/Arduino Development/Program Files/ Color Sorting/ Color_Sorting /Color_Sorting.ino”



```
if (60 < dis & dis < 80) {
  if (detect_color) {
    setBuzzer(100);
    delay(1000);
    pos[0] = 0; pos[1] = -160; pos[2] = 100;
    set_position(pos, 1500);
    delay(1500);
    pos[0] = 0; pos[1] = -160; pos[2] = 85;
    set_position(pos, 800);
    Pump_on();
    delay(1000);
    pos[0] = 0; pos[1] = -160; pos[2] = 180;
    set_position(pos, 1000);
    delay(1000);
    pos[0] = x; pos[1] = y; pos[2] = 180;
    set_position(pos, 1500);
    delay(1500);
    SetPWMServo(1, angle_pul, 800);
    delay(200);
    pos[0] = x; pos[1] = y; pos[2] = z;
    set_position(pos, 1000);
    delay(1000);
    Valve_on();
    pos[0] = x; pos[1] = y; pos[2] = 200;
    set_position(pos, 1000);
    delay(1000);
```

Firstly, import the corresponding libraries and initialize ultrasonic sensor and color sensor.

Then read and calculate the RGB value of the object, and print out the color result.

Finally, the ultrasonic sensor emits corresponding light, and the functions for controlling servo and air pump are executed. MaxArm will suck and place the object to the corresponding position according to the color.

2. Preparation

2.1 Hardware

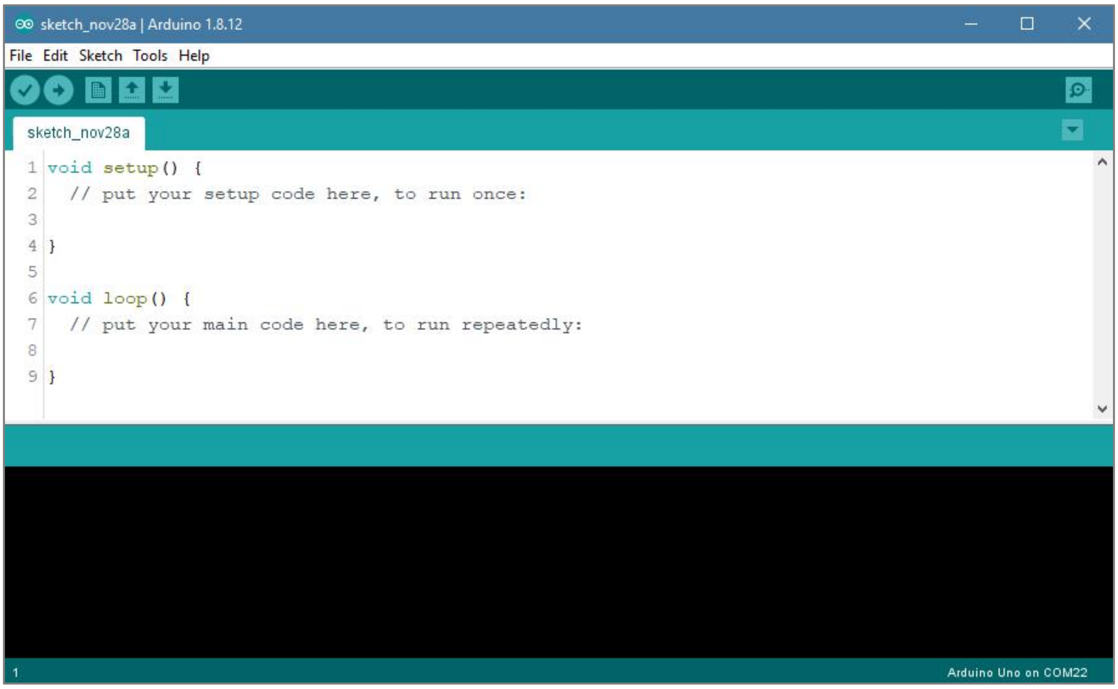
Please assemble the color sensor and the ultrasonic sensor to the corresponding position on MaxArm according to the tutorial in folder “Lesson 1 Infrared Sensor Assembly” under the same directory.

2.2 Software

Please connect MaxArm to Arduino editor according to the tutorial in folder “4. Underlying Program Learning/Arduino Development/Lesson 1 Set Development Environment”.

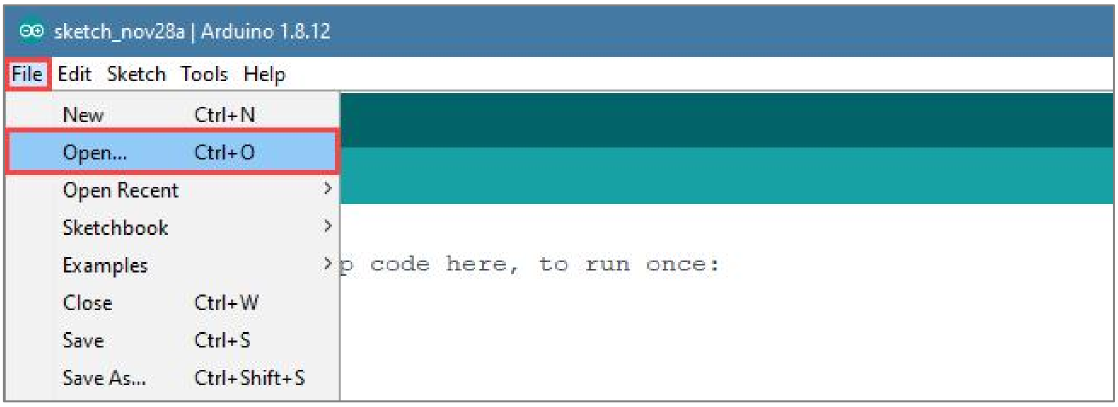
3. Program Download

1) Click on arduino icon to open Arduino IDE.

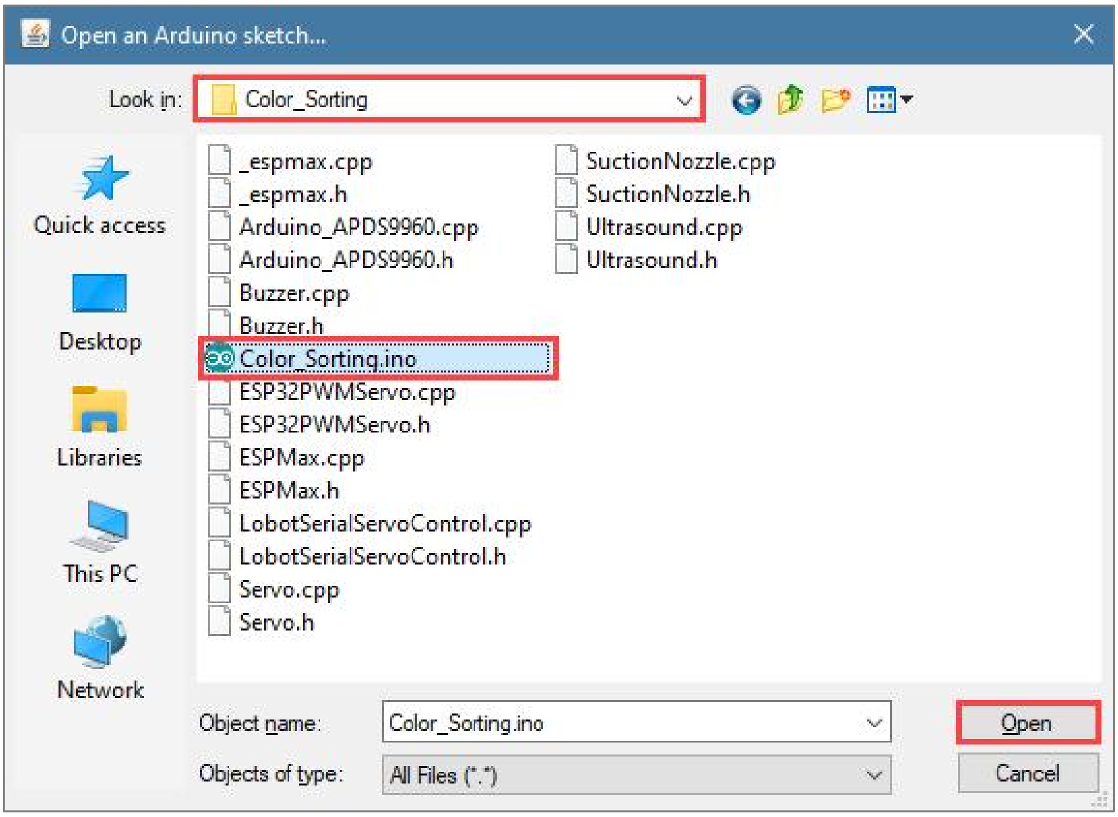


```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

2) Click “File->Open” in turn.



3) Select the program “InfraredSensor_control.ino” in the folder “6.Secondary Development /Sensor-extension Game/Arduino Development/ Program Files/Infrared Detection and Control/InfraredSensor_control”.



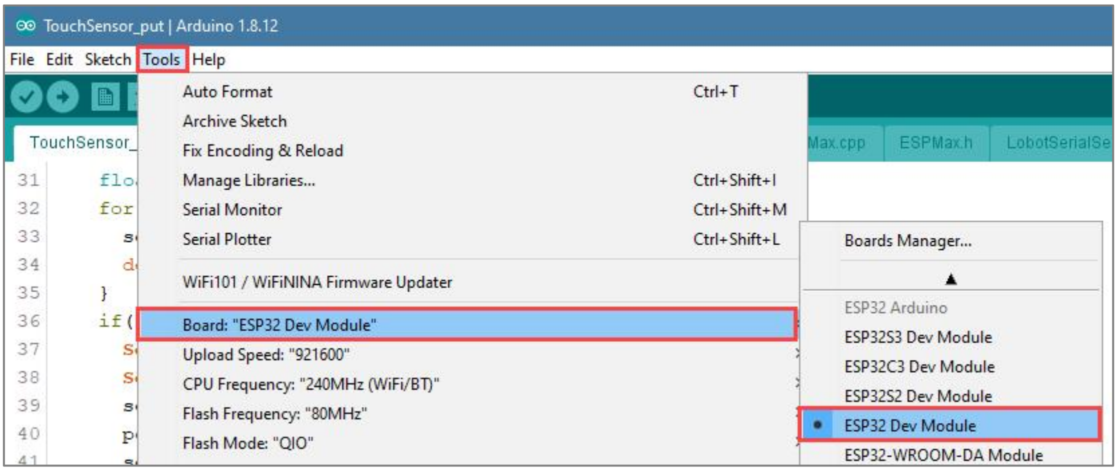
Open an Arduino sketch...

Look in: Color_Sorting

Object name: Color_Sorting.ino

Objects of type: All Files (*.*)

4) Select the model of the development board. Click “Tools-> Board” and select “ESP 32 Dev Module” (If the model of the development board has been configured when setting the development environment, you can skip this step).



TouchSensor_put | Arduino 1.8.12

Board: "ESP32 Dev Module"

Upload Speed: "921600"

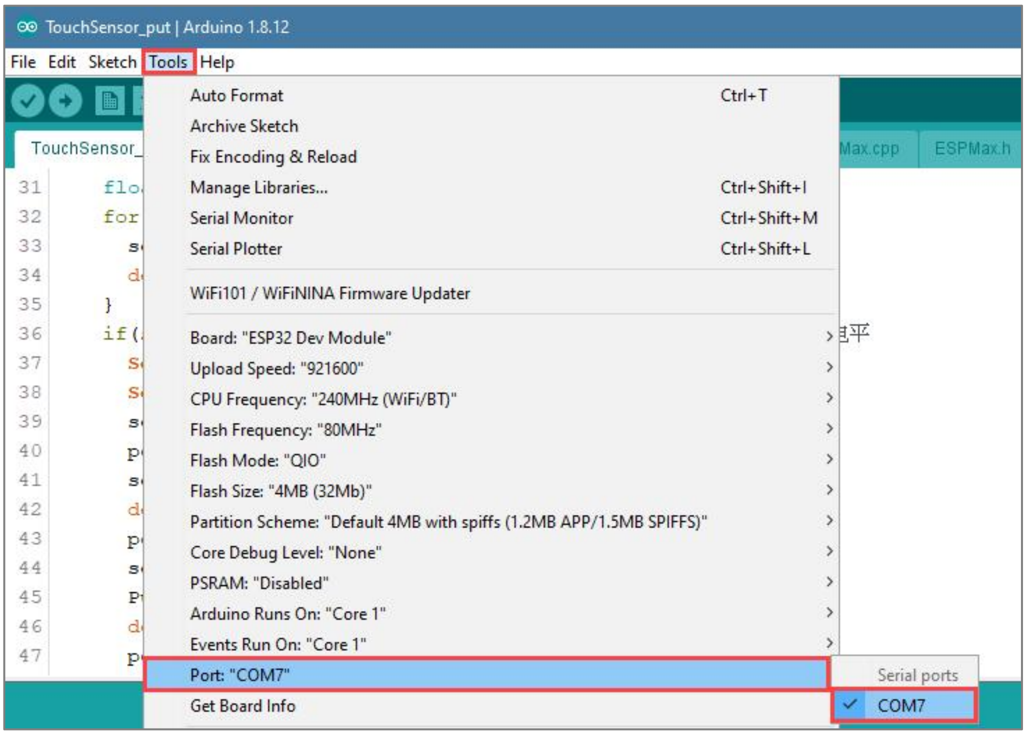
CPU Frequency: "240MHz (WiFi/BT)"

Flash Frequency: "80MHz"

Flash Mode: "QIO"

5) Select the corresponding port of Arduino controller in “Tools->Port”. (Here take the port “COM5” as example. Please select the port based on your computer. If COM1 appears, please do not select because it is the system

communication port but not the actual port of the development port.)



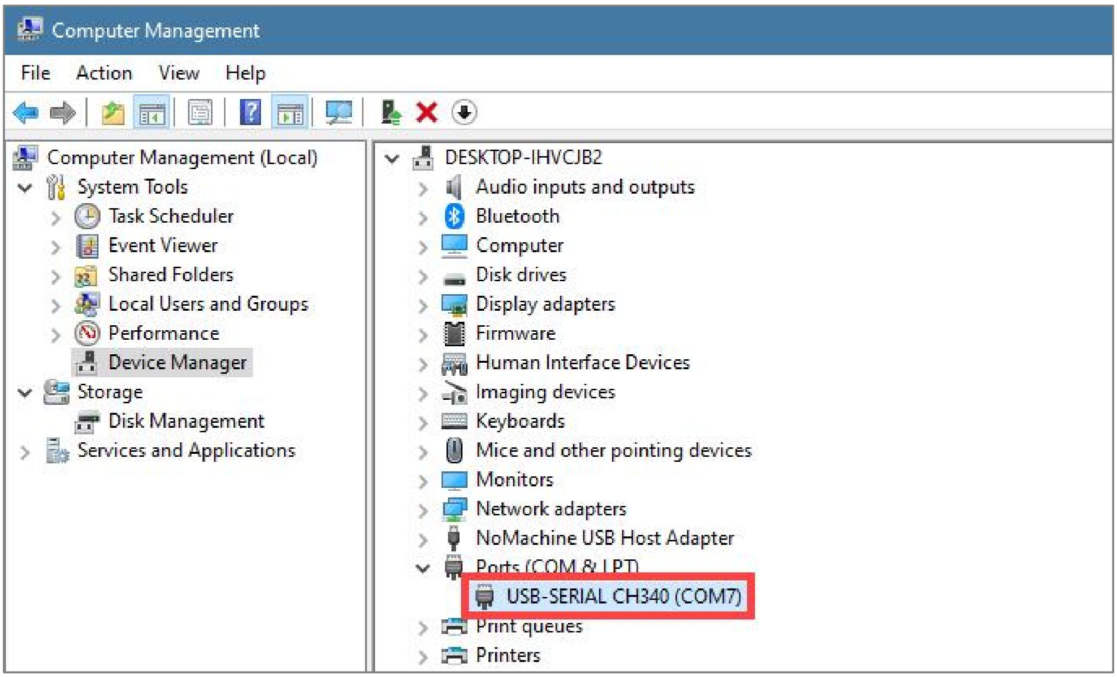
TouchSensor_put | Arduino 1.8.12

Tools

Port: "COM7"

COM7

6) If you’re not sure about the port number, please open the “This PC” and click “Properties->Device Manger” in turns to check the corresponding port number (the device is with CH340). Then select the correct port on Arduino editor.



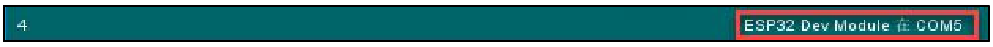
Computer Management

Device Manager

Ports (COM & LPT)

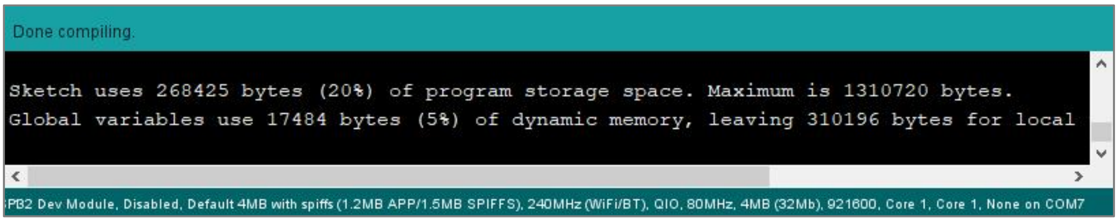
USB-SERIAL CH340 (COM7)

7) After selecting, confirm the board “ESP32 Dev Module” in the lower right corner and the port number “COM5” (it is an example here, please refer to the actual situation).



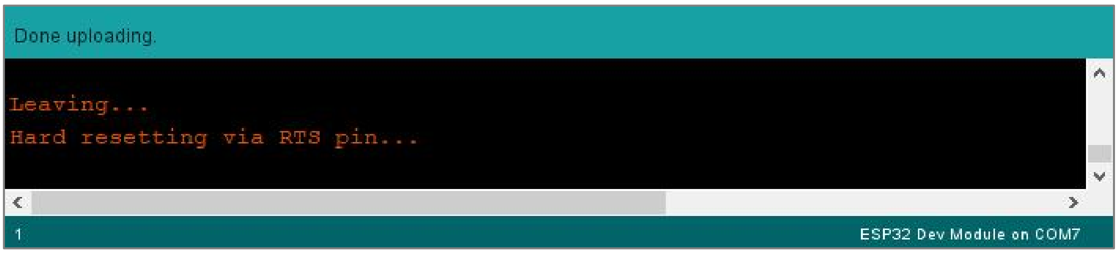
```
4                                                    ESP32 Dev Module 在 COM5
```

8) Then click on ✓ icon to verify the program. If no error, the status area will display “Compiling->Compile complete” in turn. After compiling, the information such as the current used bytes, and occupied program storage space will be displayed.



```
Done compiling.

Sketch uses 268425 bytes (20%) of program storage space. Maximum is 1310720 bytes.
Global variables use 17484 bytes (5%) of dynamic memory, leaving 310196 bytes for local

PB2 Dev Module, Disabled, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), 240MHz (WiFi/BT), QIO, 80MHz, 4MB (32Mb), 921600, Core 1, Core 1, None on COM7
```

9) After compiling, click on ➔ icon to upload the program to the development board. The status area will display “Compiling->Uploading->Complete” in turn. After uploading, the status area will stop printing the uploading information.



```
Done uploading.

Leaving...
Hard resetting via RTS pin...

1                                                    ESP32 Dev Module on COM7
```

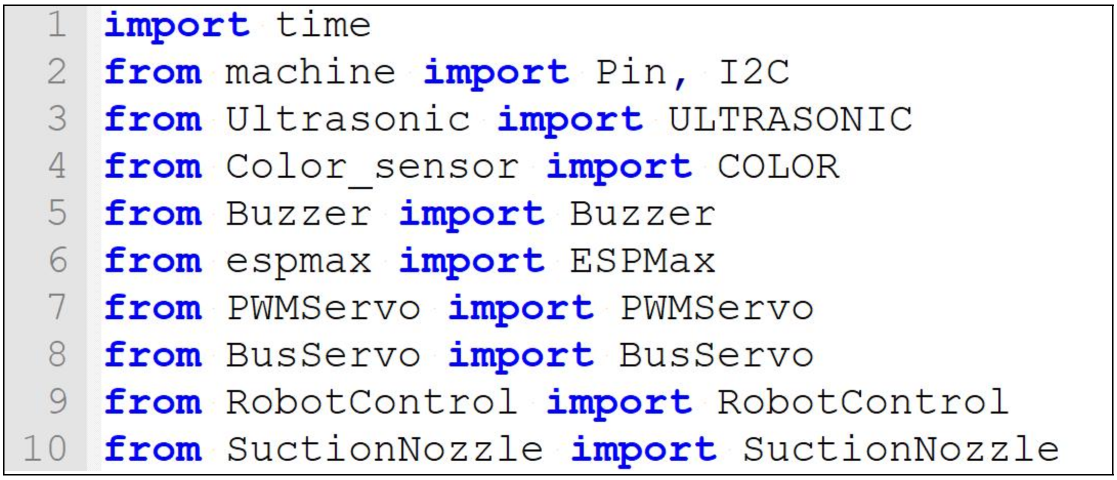
4. Project Outcome

After the color sensor recognizes the color of the block and the ultrasonic sensor detects that the block is placed in the placement area, MaxArm will sort the block based on the recognized color, and suck and transport the block to the corresponding position.

5. Program Instruction

5.1 Import Function Library

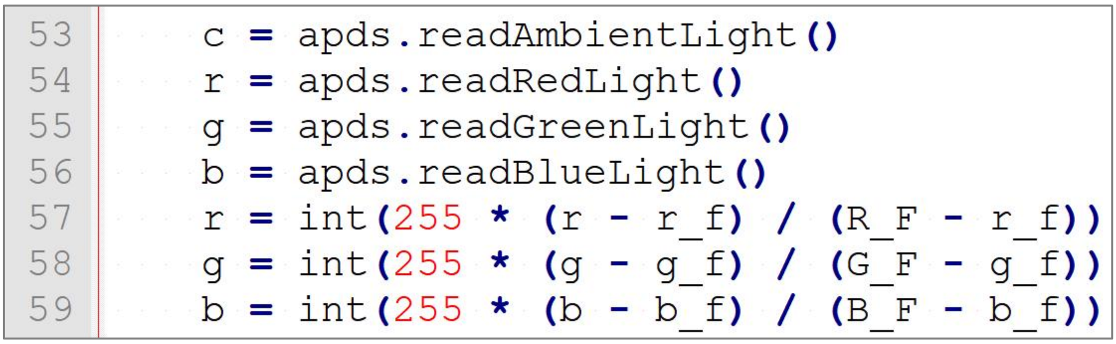
Before executing the program, the I2C protocol, ultrasonic sensor, color sensor, buzzer, PWM servo, bus servo, infrared sensor, air pump and other ralated Python function libraries need to be imported.



```
import time
from machine import Pin, I2C
from Ultrasonic import ULTRASONIC
from Color_sensor import COLOR
from Buzzer import Buzzer
from espmax import ESPMax
from PWMServo import PWMServo
from BusServo import BusServo
from RobotControl import RobotControl
from SuctionNozzle import SuctionNozzle
```

5.2 Color Detection

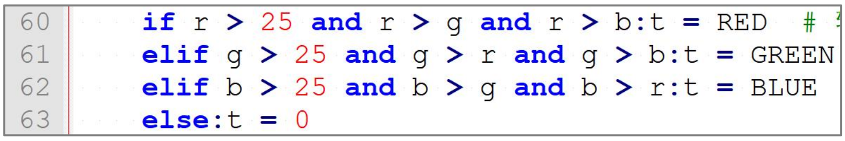
Use apds.readRedLight(), apds.readGreenLight() and apds.readBlueLight() functions of color sensor to detect the value of RGB channel of object, and then calculate the value.



```
    c = apds.readAmbientLight()
    r = apds.readRedLight()
    g = apds.readGreenLight()
    b = apds.readBlueLight()
    r = int(255 * (r - r_f) / (R_F - r_f))
    g = int(255 * (g - g_f) / (G_F - g_f))
    b = int(255 * (b - b_f) / (B_F - b_f))
```

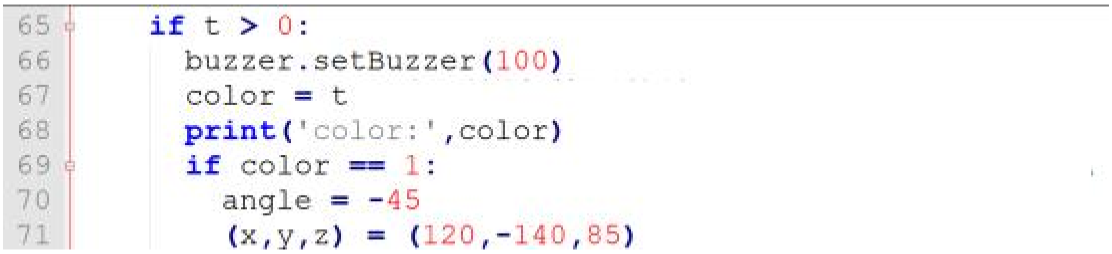
5.3 Control the LED Light of Ultrasonic Sensor

After calculating the value of RGB channel of object color, determine the color of the object.



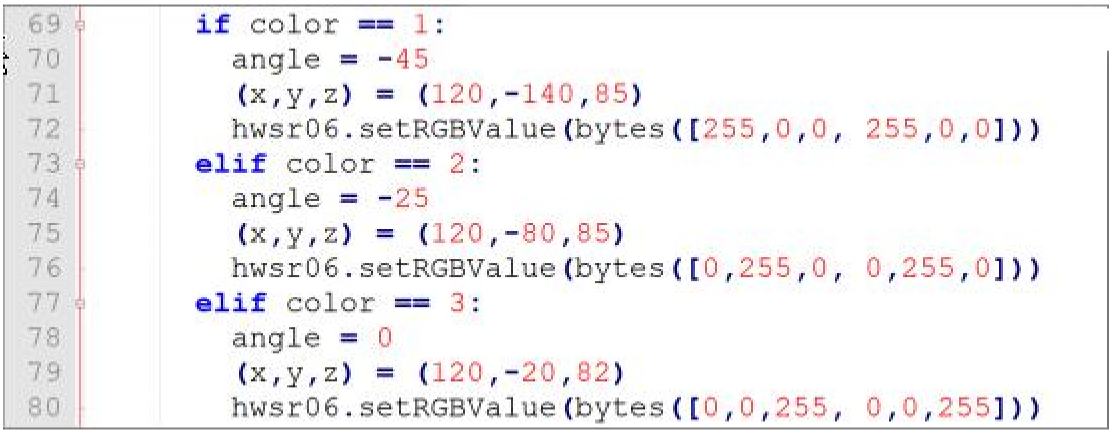
```
    if r > 25 and r > g and r > b:t = RED   #
    elif g > 25 and g > r and g > b:t = GREEN
    elif b > 25 and b > g and b > r:t = BLUE
    else:t = 0
```

And the block with different color is set to place in the different position.



```
    if t > 0:
      buzzer.setBuzzer(100)
      color = t
      print('color:',color)
      if color == 1:
        angle = -45
        (x,y,z) = (120,-140,85)
```

Then, the ultrasonic sensor emits the corresponding color light.



```
      if color == 1:
        angle = -45
        (x,y,z) = (120,-140,85)
        hwsr06.setRGBValue(bytes([255,0,0, 255,0,0]))
      elif color == 2:
        angle = -25
        (x,y,z) = (120,-80,85)
        hwsr06.setRGBValue(bytes([0,255,0, 0,255,0]))
      elif color == 3:
        angle = 0
        (x,y,z) = (120,-20,82)
        hwsr06.setRGBValue(bytes([0,0,255, 0,0,255]))
```

Use hwsr06.setRGBValue(bytes()) function to control the LED light of the ultrasonic sensor. Take the code “hwsr06.setRGBValue(bytes([255,0,0, 255,0,0]))” as example.

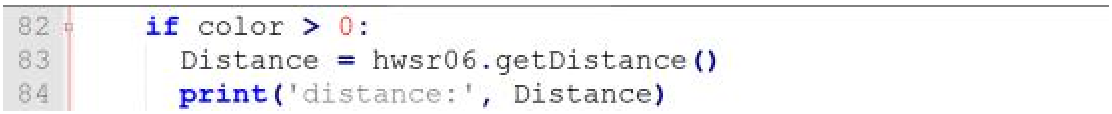
The first three parameters "255,0,0" are the color thresholds for the LED on the right side of the senor.

The last three parameters "255,0,0" are the color thresholds for the left LED of the sensor.

At this time, the LED emits red light.

5.4 Ultrasonic detection

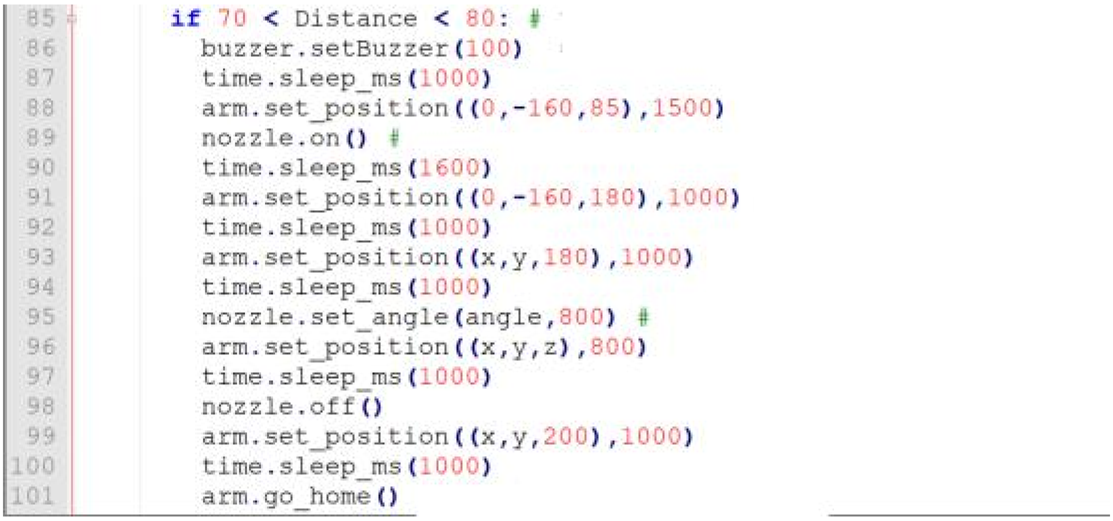
If the block is detected, use hwsr06.getDistance() function to measure distance. Then use print() function to print out the measured distance.



```
    if color > 0:
      Distance = hwsr06.getDistance()
      print('distance:', Distance)
```

5.5 Control Robotic Arm

Determine whether there are objects at the distance between 70 and 80, and then execute the corresponding action.



```
    if 70 < Distance < 80: #
      buzzer.setBuzzer(100)
      time.sleep_ms(1000)
      arm.set_position((0,-160,85),1500)
      nozzle.on() #
      time.sleep_ms(1600)
      arm.set_position((0,-160,180),1000)
      time.sleep_ms(1000)
      arm.set_position((x,y,180),1000)
      time.sleep_ms(1000)
      nozzle.set_angle(angle,800) #
      arm.set_position((x,y,z),800)
      time.sleep_ms(1000)
      nozzle.off()
      arm.set_position((x,y,200),1000)
      time.sleep_ms(1000)
      arm.go_home()
```

Use buzzer.setBuzze() function to control the buzzer. Take the code “buzzer.setBuzzer(100)” as example.

The first parameter “100” represents the sounding time of buzzer and the unit is ms.

Use arm.set_position() function to control robotic arm. Take the code “arm.set_position((0,-160,85),1500)” as example.

The first parameter “(0，-160，85)” represents is the position of the suction nozzle on x, y and z axes.

The second parameter “1500” is the running time and the unit is ms.

Use nozzle.set_angle() function to control the rotation of the suction nozzle. Take the code “nozzle.set_angle(0，800)” as example.

The first parameter “0” is the angle of PWM servo.

The second parameter “800” is the running time and the unit is ms.