Lesson 3 Waste Sorting

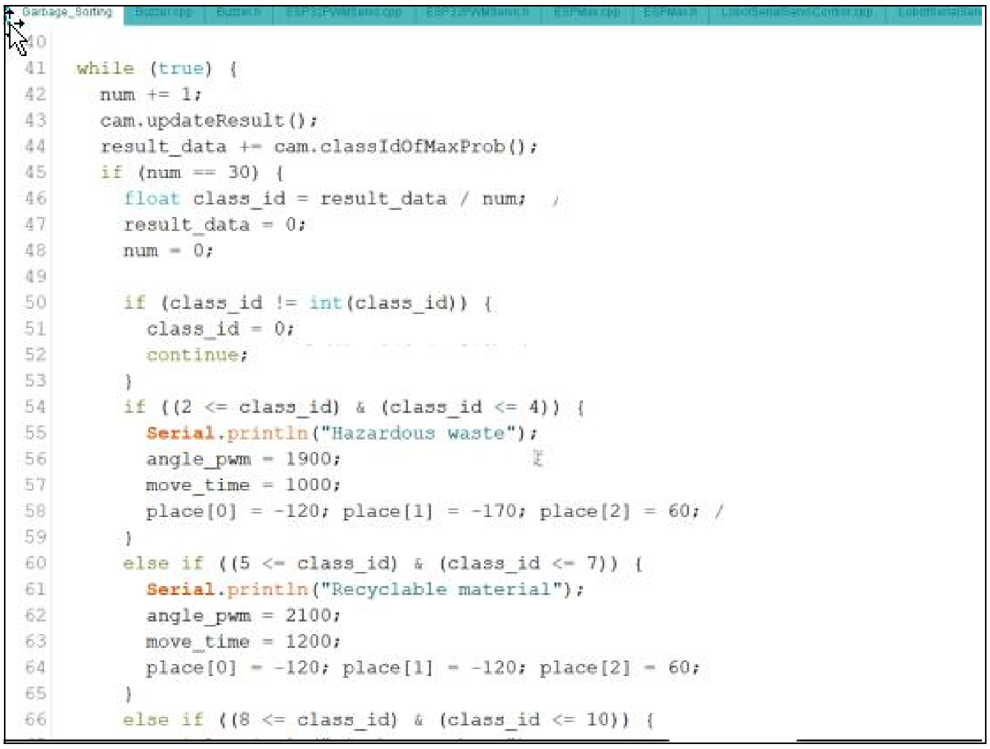
Note: If the mask recognition models have been burnt, please re-download the firmware of WonderCam vision module before starting this game. The specific operation refers to “WonderCam Firmware (Installation method included)”.

1. Project Principle

Using image classification function, robot arm is able to recognize waste cards, and then sort them into the corresponding position.

Waste classification is mainly implemented by the trained waste models, and the sorting function is implemented by inverse kinematics function to control robotic arm. The specific control method refers to “5. Program Instruction” in this lesson.

The path of the program file: “7. AI Vision Game/ Arduino Development/ AI Vision Game/ Program Files/ Waste Sorting/Garbage_Sorting/ Garbage_Sorting.ino”



```
while (true) {
  num += 1;
  cam.updateResult();
  result_data += cam.classIdOfMaxProb();
  if (num == 30) {
    float class_id = result_data / num;
    result_data = 0;
    num = 0;

    if (class_id != int(class_id)) {
      class_id = 0;
      continue;
    }
    if ((2 <= class_id) & (class_id <= 4)) {
      Serial.println("Hazardous waste");
      angle_pwm = 1900;
      move_time = 1000;
      place[0] = -120; place[1] = -170; place[2] = 60;
    }
    else if ((5 <= class_id) & (class_id <= 7)) {
      Serial.println("Recyclable material");
      angle_pwm = 2100;
      move_time = 1200;
      place[0] = -120; place[1] = -120; place[2] = 60;
    }
    else if ((8 <= class_id) & (class_id <= 10)) {
```

2. Preparation

2.1 Hardware

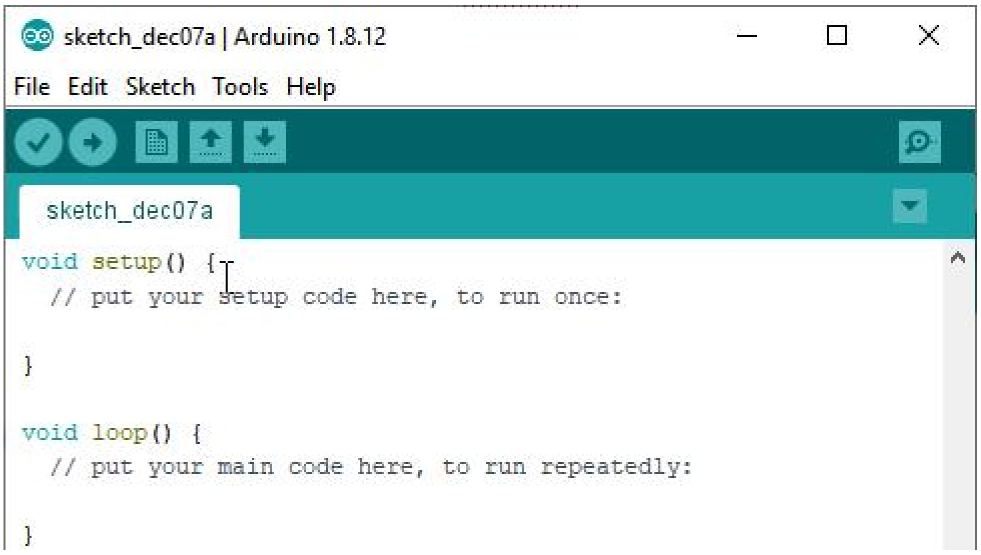
Please refer to "7.AI Vision Game/ WonderCam AI Vision Module (must read!)/ Lesson 3 Aassembly" to assemble WonderCam module to the corresponding position on MaxArm

2.2 Software

Please refer to "4. Underlying Program Learning/ Arduino Development/ Lesson 1 Set Development Environment" to connect MaxArm to Arduino.

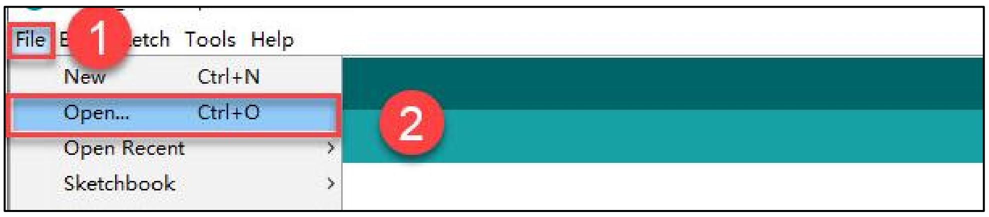
3. Program Download

1) Double click to open Arduino IDE .

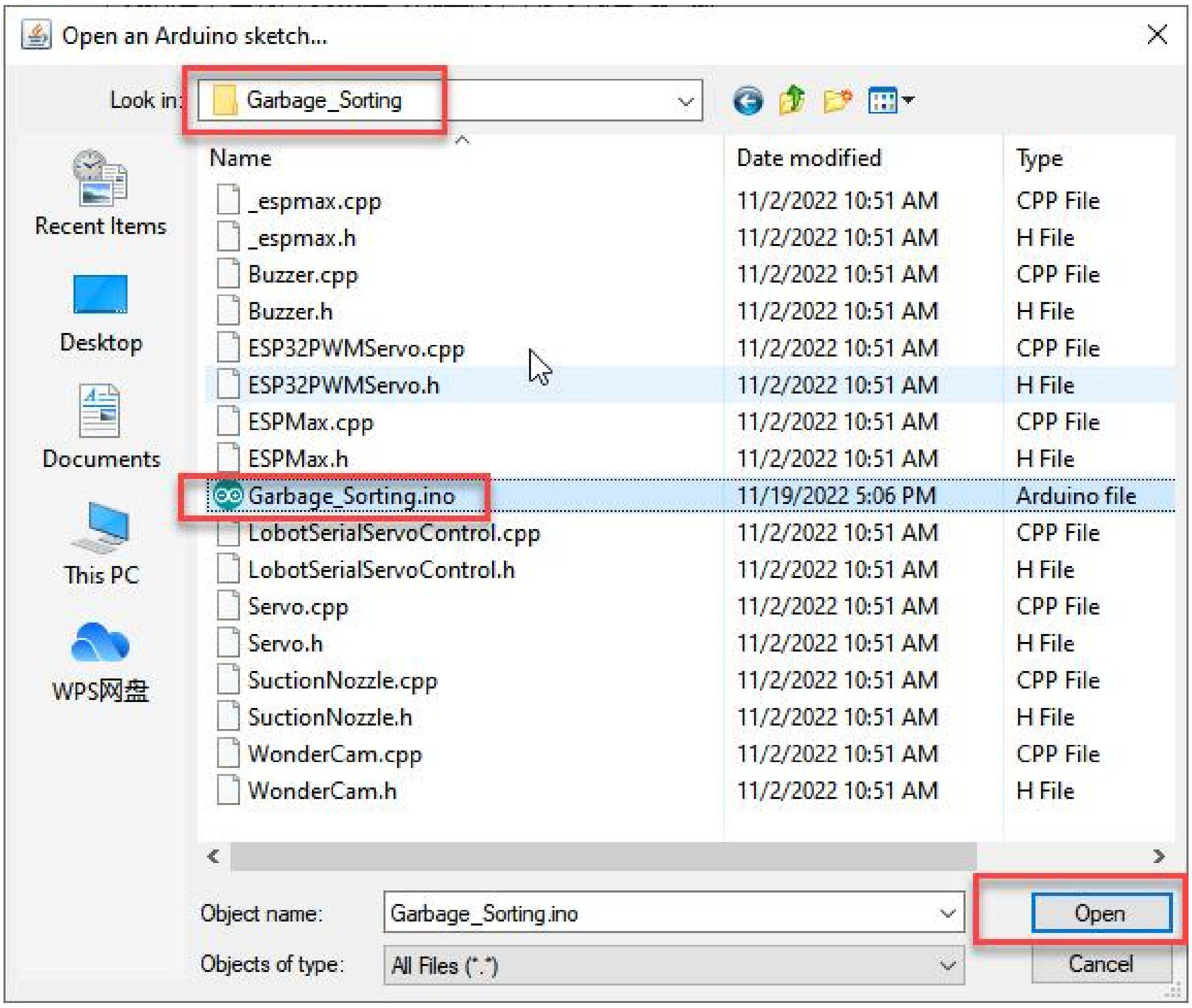


```
void setup() {
  // put your setup code here, to run once:

}

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

}
```

2) Click "File -- Open".



3) Open the program “Masks_identify.ino” in the folder “7. AI Vision Game/ Arduino Development/ Sensor-extension Game/ Program Files/ Mask Recognition/ Masks_identify.

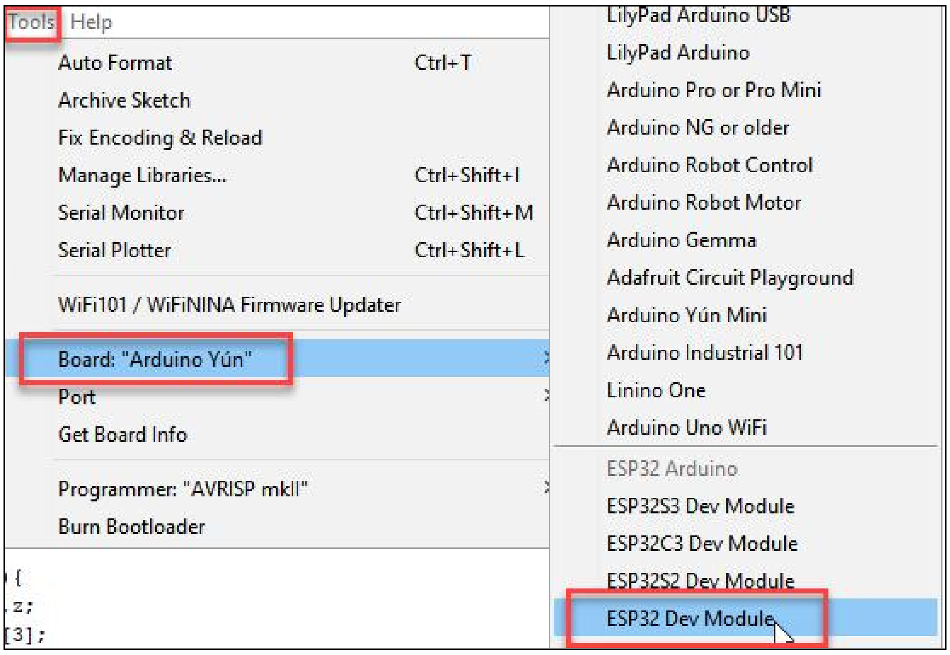


Name	Date modified	Type
_espmax.cpp	11/2/2022 10:51 AM	CPP File
_espmax.h	11/2/2022 10:51 AM	H File
Buzzer.cpp	11/2/2022 10:51 AM	CPP File
Buzzer.h	11/2/2022 10:51 AM	H File
ESP32PWMServo.cpp	11/2/2022 10:51 AM	CPP File
ESP32PWMServo.h	11/2/2022 10:51 AM	H File
ESPMax.cpp	11/2/2022 10:51 AM	CPP File
ESPMax.h	11/2/2022 10:51 AM	H File
Garbage_Sorting.ino	11/19/2022 5:06 PM	Arduino file
LobotSerialServoControl.cpp	11/2/2022 10:51 AM	CPP File
LobotSerialServoControl.h	11/2/2022 10:51 AM	H File
Servo.cpp	11/2/2022 10:51 AM	CPP File
Servo.h	11/2/2022 10:51 AM	H File
SuctionNozzle.cpp	11/2/2022 10:51 AM	CPP File
SuctionNozzle.h	11/2/2022 10:51 AM	H File
WonderCam.cpp	11/2/2022 10:51 AM	CPP File
WonderCam.h	11/2/2022 10:51 AM	H File

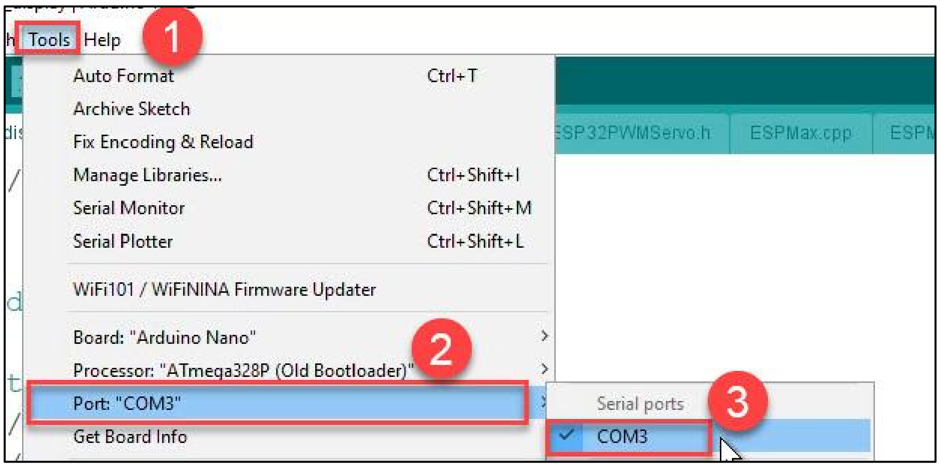
Object name: Garbage_Sorting.ino

Objects of type: All Files (*.*)

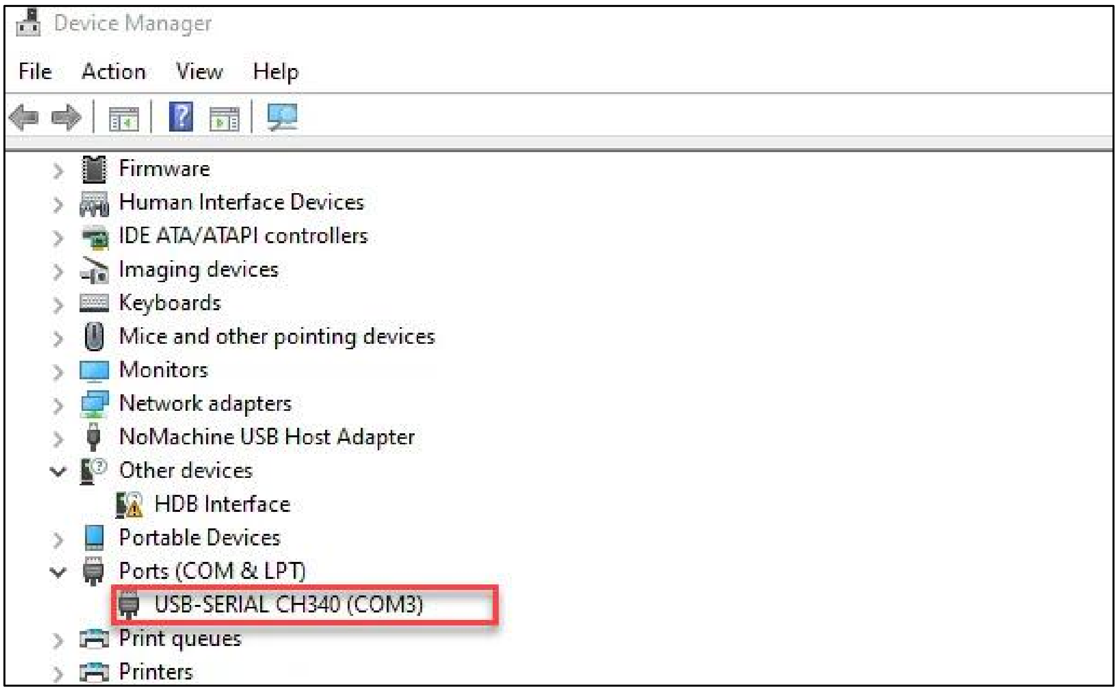
4) Select the corresponding board model. Click “Tool -- Board” and select “ESP 32 Dev Module” (If the board model has been configured when setting development environment, you can skip this step.)



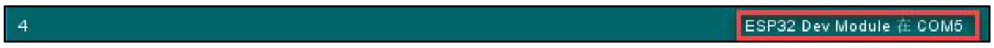
5) Select the corresponding port in “Tools->Port”. (Here take the port “COM5” as example. Please select the port based on your computer. If COM1 appears, please do not select because it is the system communication port but not the actual port of the development port.)


1
Tools
Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Board: "Arduino Nano"
2
Processor: "ATmega328P (Old Bootloader)"
Port: "COM3"
Get Board Info
ESP32PWMServo.h
ESPMax.cpp
Serial ports
3
COM3

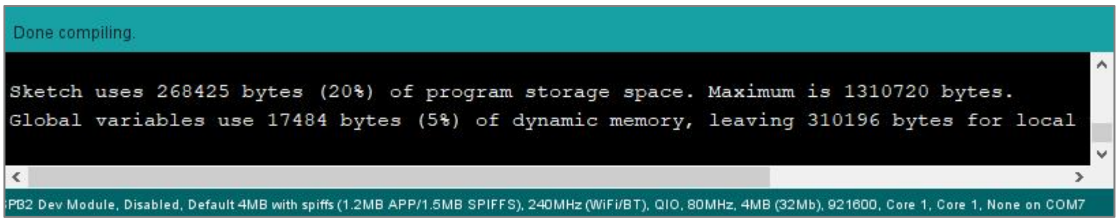
6) If you’re not sure about the port number, please open the “This PC” and click “Properties->Device Manger” in turns to check the corresponding port number (the device is with CH340).



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).



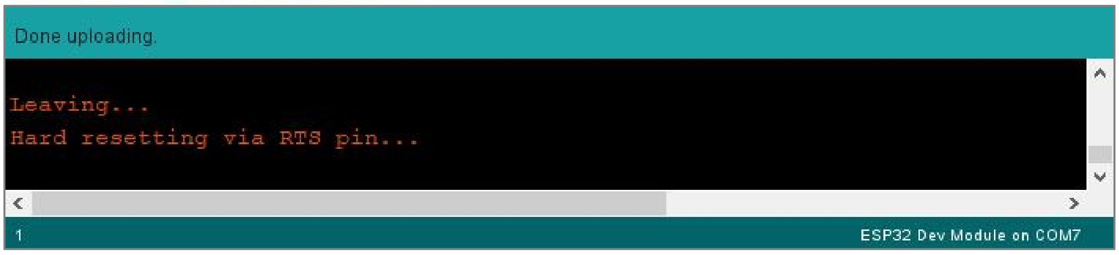
8) Then click on ✓ icon to verify the program. If no error, the status area will display “Compiling->Compile complete” in turn. After compiling, the information such as the current used bytes, and occupied program storage space will be displayed.


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
```

P32 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

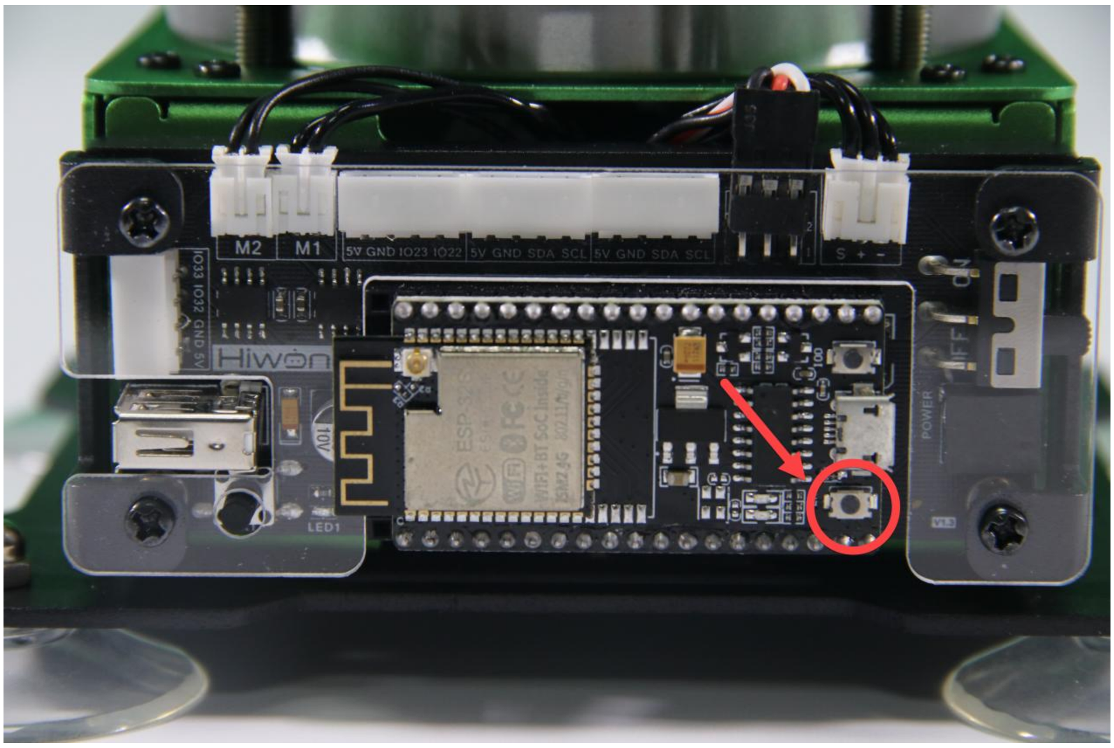
9) After compiling, click on ➜ icon to upload the program to the development board. The status area will display “Compiling->Uploading->Complete” in turn. After uploading, the status area will stop printing the uploading information.



Done uploading.

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

10) After download is completed, click “reset” or the reset button on controller to run program.



4. Project Outcome

After the program is downloaded, press reset button according to the operation steps in “3. Program Download”. Then robot arm will be in preparation for recognition.

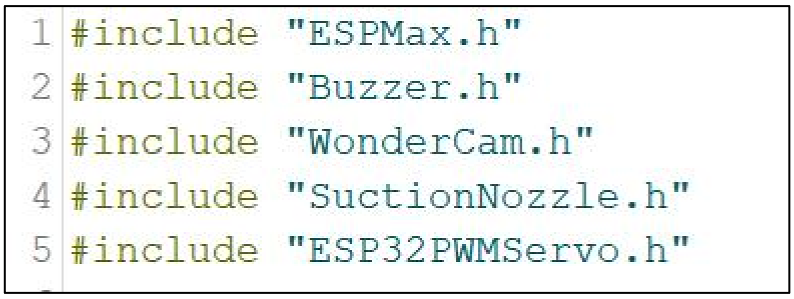
Then place a waste card to the recognition area. When the card is recognized, MaxArm will suck and transfer it to the corresponding position according to the card category.

5. Program Instruction

The path of the program file: “7. AI Vision Game/ Arduino Development/ Program Files/Garbage_Sorting/Garbage_Sorting.ino”. You can find the backup program file in Appendix if the original program file is modified.

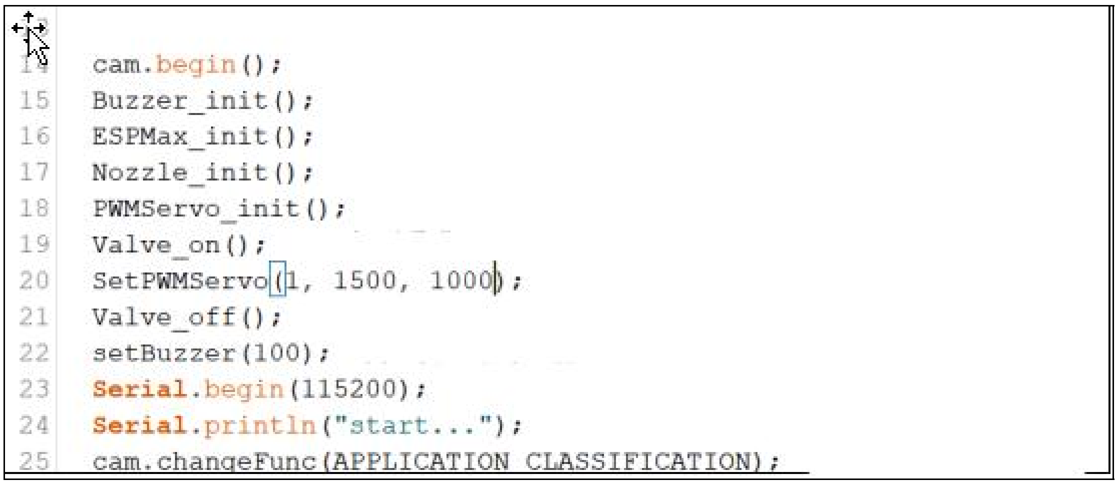
5.1 Import function library and Initialization

Firstly, import the inverse kinematics library, WonderCam driver library, buzzer library, PWM servo library, suction nozzle library and other function libraries corresponding to this game.



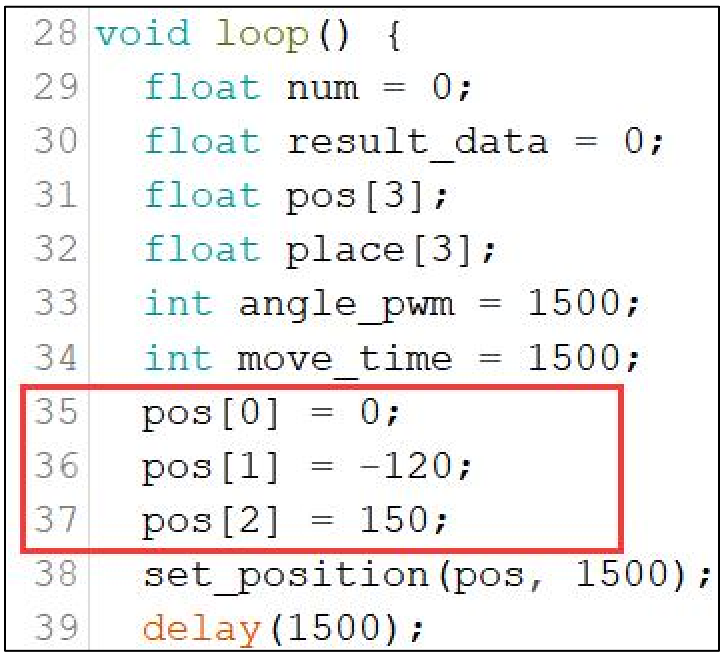
```
#include "ESPMax.h"
#include "Buzzer.h"
#include "WonderCam.h"
#include "SuctionNozzle.h"
#include "ESP32PWMServo.h"
```

Then initialize each modules, as the figure shown below:



```
cam.begin();
Buzzer_init();
ESPMax_init();
Nozzle_init();
PWMServo_init();
Valve_on();
SetPWMServo(1, 1500, 1000);
Valve_off();
setBuzzer(100);
Serial.begin(115200);
Serial.println("start...");
cam.changeFunc(APPLICATION_CLASSIFICATION);
```

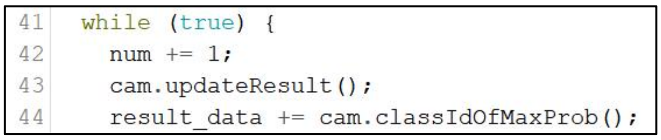
Set the coordinate of the initial position of the end effector, as the figure shown below:



```
void loop() {
  float num = 0;
  float result_data = 0;
  float pos[3];
  float place[3];
  int angle_pwm = 1500;
  int move_time = 1500;
  pos[0] = 0;
  pos[1] = -120;
  pos[2] = 150;
  set_position(pos, 1500);
  delay(1500);
```

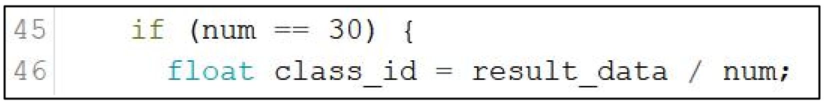
5.2 Image Classification Recognition

Call the cam.updateResult() function to update the recognized result, and call the cam.classIdOfMaxProb() function to obtain ID number the save it to result_data. (Each waste card has a specific ID number)



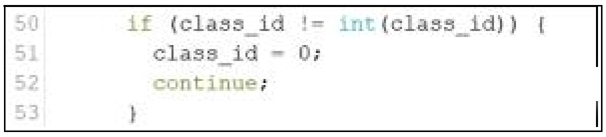
```
  while (true) {
    num += 1;
    cam.updateResult();
    result_data += cam.classIdOfMaxProb();
```

To prevent misidentification, we take the average value of 30 detections and assign it to result.



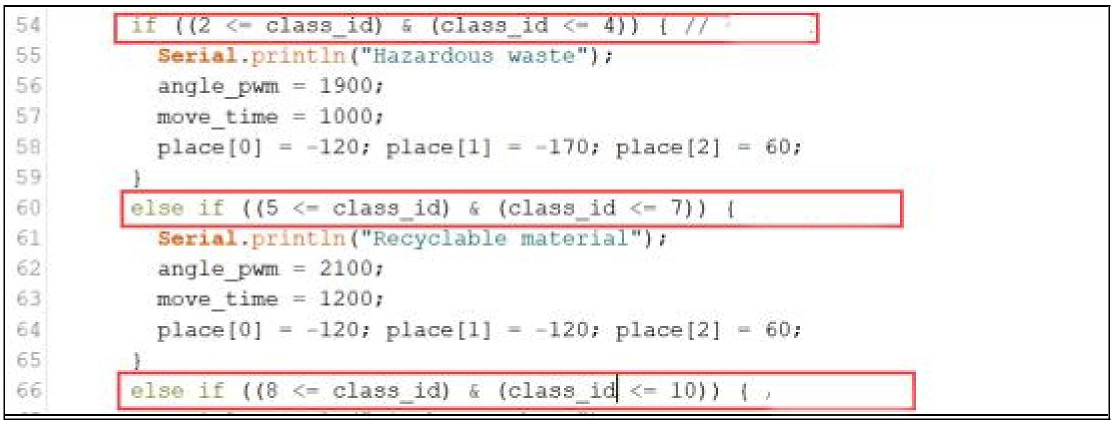
```
    if (num == 30) {
      float class_id = result_data / num;
```

If the result is not an integer value, it means the test result is unstable and the program needs to restart the next round of testing.



```
      if (class_id != int(class_id)) {
        class_id = 0;
        continue;
      }
```

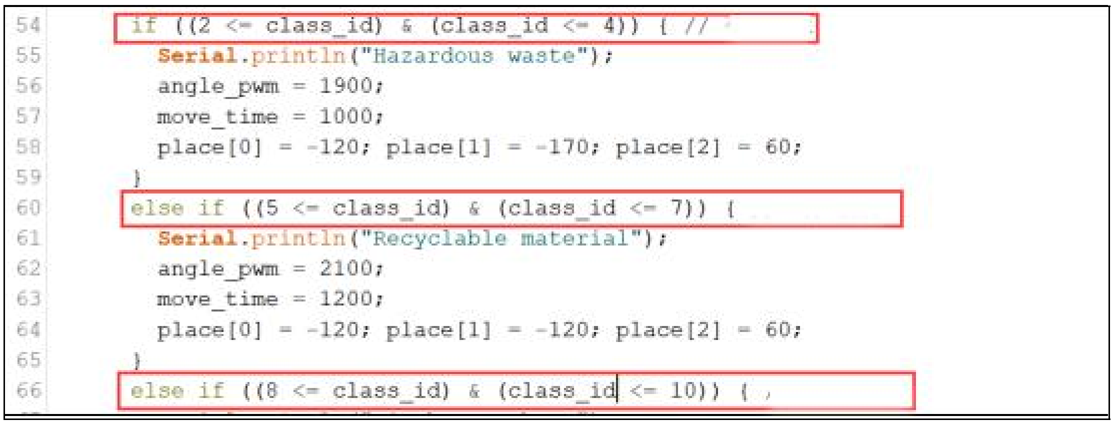
Then the card into the corresponding category according to ID result.



```
    if ((2 <= class_id) & (class_id <= 4)) { //
      Serial.println("Hazardous waste");
      angle_pwm = 1900;
      move_time = 1000;
      place[0] = -120; place[1] = -170; place[2] = 60;
    }
    else if ((5 <= class_id) & (class_id <= 7)) {
      Serial.println("Recyclable material");
      angle_pwm = 2100;
      move_time = 1200;
      place[0] = -120; place[1] = -120; place[2] = 60;
    }
    else if ((8 <= class_id) & (class_id <= 10)) {
```

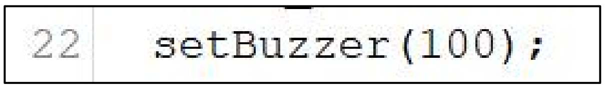
5.3 Suck Card

After the card is recognized, buzzer makes sound and MaxArm sucks and place the block to the corresponding card category, as the figure shown below:



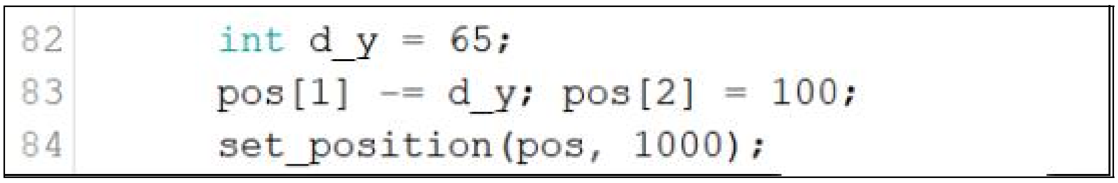
```
    if ((2 <= class_id) & (class_id <= 4)) { //
      Serial.println("Hazardous waste");
      angle_pwm = 1900;
      move_time = 1000;
      place[0] = -120; place[1] = -170; place[2] = 60;
    }
    else if ((5 <= class_id) & (class_id <= 7)) {
      Serial.println("Recyclable material");
      angle_pwm = 2100;
      move_time = 1200;
      place[0] = -120; place[1] = -120; place[2] = 60;
    }
    else if ((8 <= class_id) & (class_id <= 10)) {
```

Use the "**setBuzzer()**" function to control buzzer, as the figure shown below. Among them, the parameter 100 represents the sound duration of buzzer and its unit is ms.



```
setBuzzer(100);
```

Use the "set_position()" function to control MaxArm to locate above the card, as the figure shown below:



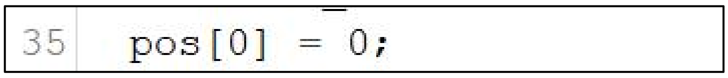
```
        int d_y = 65;
        pos[1] -= d_y; pos[2] = 100;
        set_position(pos, 1000);
```

Among them,

The “62” in “d_y=65” refers to the x-axis value of suction nozzle.

Then set the value of pos[2] as 100, i.e., the value of y-axis is 100.

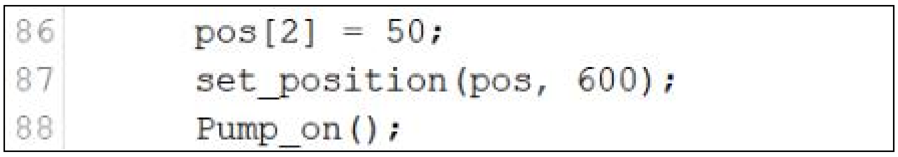
Keep the value of z-axis unchanged to keep robot in recognition posture, as the figure shown below:



```
pos[0] = 0;
```

The parameter "1000" represents the time in milliseconds it takes for the robot arm to move to this coordinate.

When the suction nozzle moves to above the card, robotic arm will bend down and ready for sucking it up, as the figure shown below:



```
        pos[2] = 50;
        set_position(pos, 600);
        Pump_on();
```

Among them,

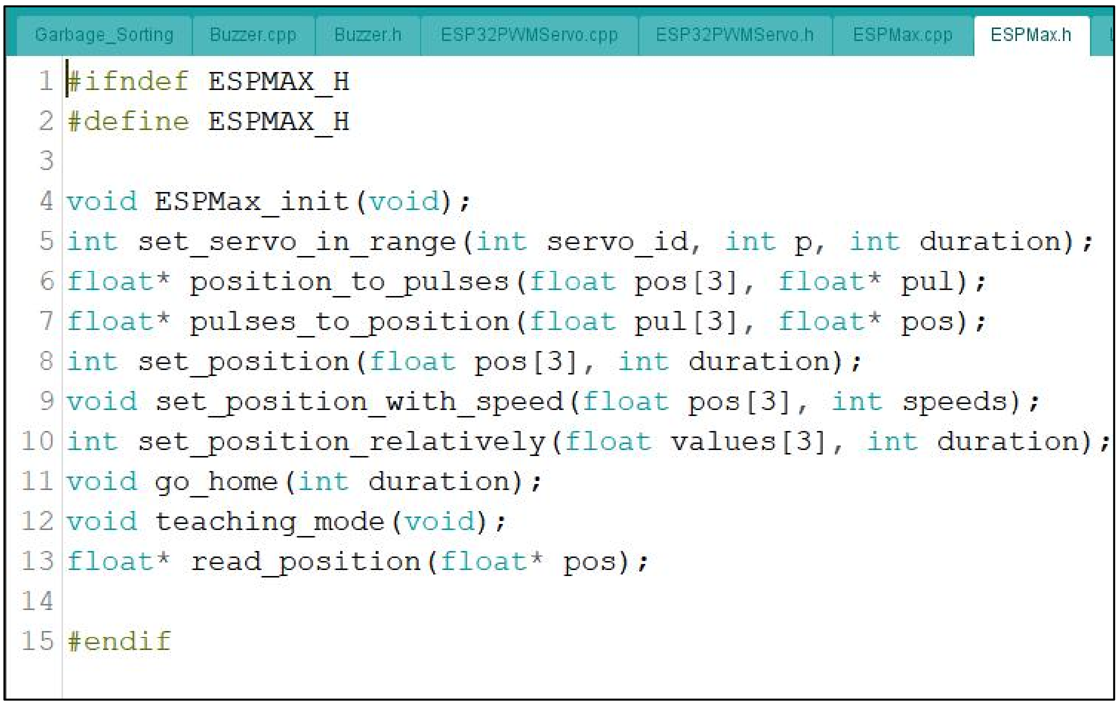
"50" in "pos[2] = 50" represents the value of the z-axis. Compared to 150, this value indicates that the robot is in a bent pose.

“**set_position(pos, 600)**” represents the coordinate of the suction nozzle, which is (0，65，50).

Then use “**Pump_on()**” function to turn on air pump to suck the card.

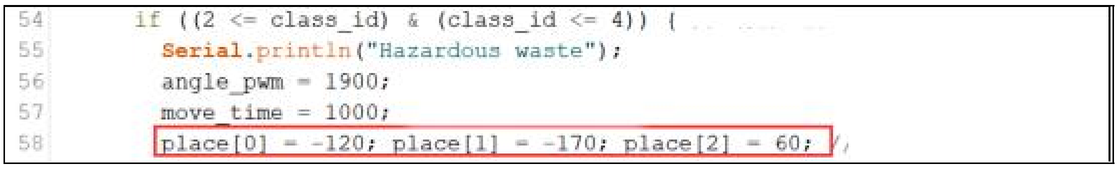
5.4 Sorting

Set up card placement areas and sort the card into the corresponding area according to the card category. However, the sorting function is implemented by the inverse kinematics function, as the figure shown below:



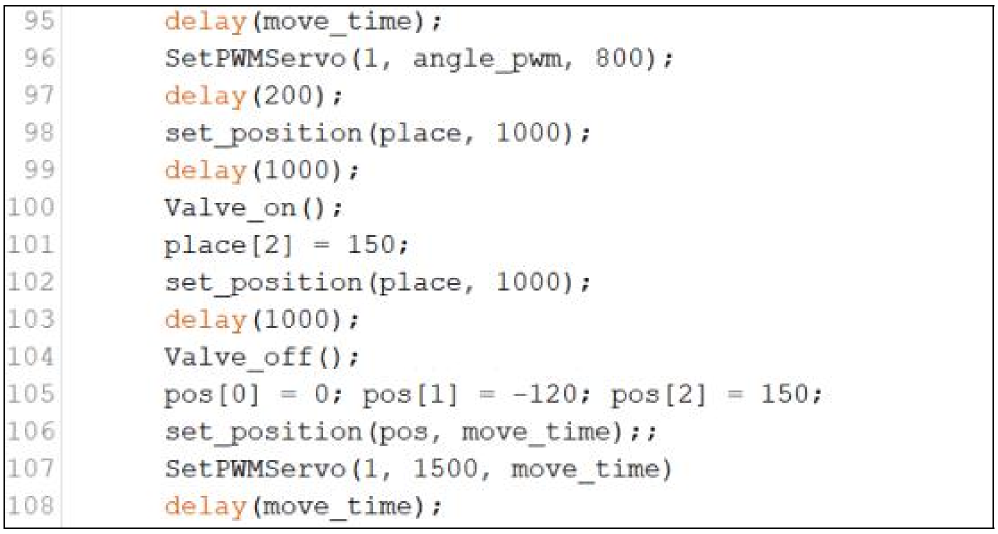
```
#ifndef ESPMAX_H
#define ESPMAX_H

void ESPMax_init(void);
int set_servo_in_range(int servo_id, int p, int duration);
float* position_to_pulses(float pos[3], float* pul);
float* pulses_to_position(float pul[3], float* pos);
int set_position(float pos[3], int duration);
void set_position_with_speed(float pos[3], int speeds);
int set_position_relatively(float values[3], int duration);
void go_home(int duration);
void teaching_mode(void);
float* read_position(float* pos);

#endif
```

Then set the coordinate parameter of the placement position (take hazardous waste for example), as the figure show below:



```
      if ((2 <= class_id) & (class_id <= 4)) {
        Serial.println("Hazardous waste");
        angle_pwm = 1900;
        move_time = 1000;
        place[0] = -120; place[1] = -170; place[2] = 60; //
```

After the block is placed, robotic arm will turn off air pump, and then return to recognition posture for next round of recognition, as the figure shown below:



```
        delay(move_time);
        SetPWMServo(1, angle_pwm, 800);
        delay(200);
        set_position(place, 1000);
        delay(1000);
        Valve_on();
        place[2] = 150;
        set_position(place, 1000);
        delay(1000);
        Valve_off();
        pos[0] = 0; pos[1] = -120; pos[2] = 150;
        set_position(pos, move_time);;
        SetPWMServo(1, 1500, move_time)
        delay(move_time);
```

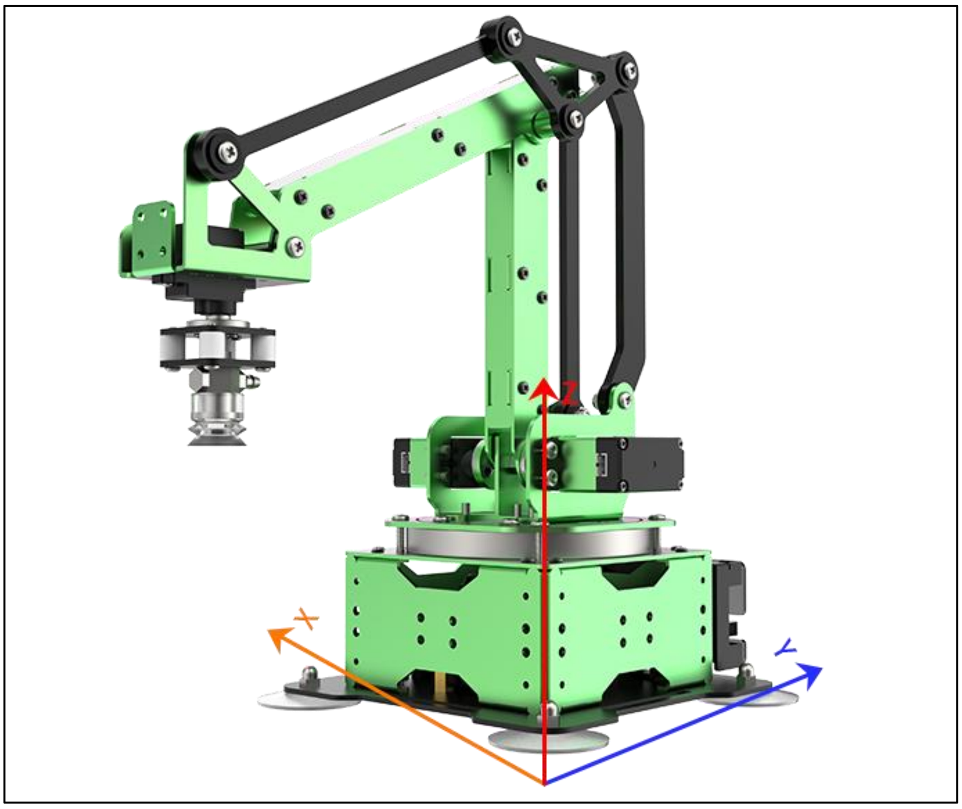
Use “Valve_on()” function to turn off air pump.

6. Function Extension

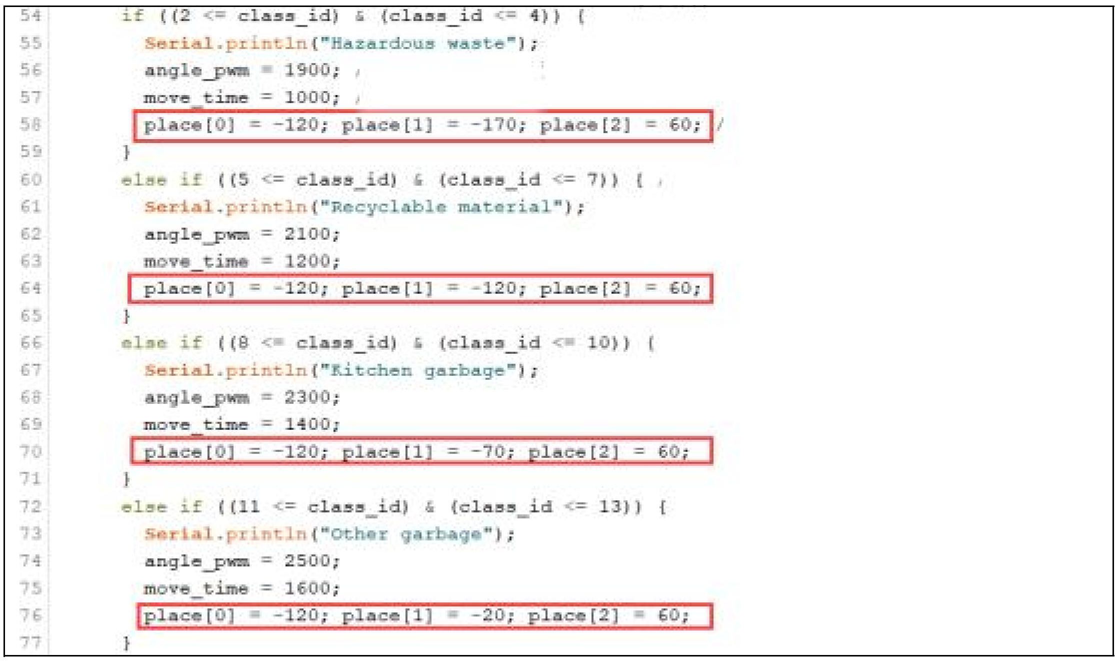
The program defaults to place the cards in a row in front of the left side of the robot arm. In this section, you can learn how to modify the placement position of cards. Take placing the cards around robotic arm for example.

Firstly, take robotic arm as the first person view. The positive direction of x axis corresponds to the right side of robotic arm; the positive direction of y axis is the rear of robotic arm; the positive direction of z axis is the top of robotic arm. All position adjustment is based on this coordinate system.

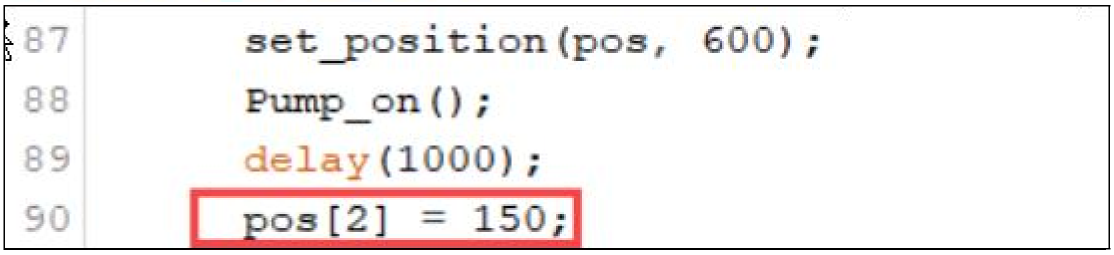
The specific coordinate system refers to the following figure:



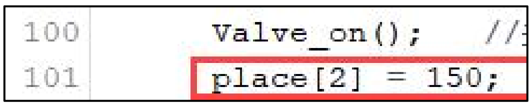
Firstly, open program file and configure port according to the steps 1-7 in "3. Program Download". Next find the following codes:



```
    if ((2 <= class_id) & (class_id <= 4)) {
      Serial.println("Hazardous waste");
      angle_pwm = 1900;
      move_time = 1000;
      place[0] = -120; place[1] = -170; place[2] = 60;
    }
    else if ((5 <= class_id) & (class_id <= 7)) {
      Serial.println("Recyclable material");
      angle_pwm = 2100;
      move_time = 1200;
      place[0] = -120; place[1] = -120; place[2] = 60;
    }
    else if ((8 <= class_id) & (class_id <= 10)) {
      Serial.println("Kitchen garbage");
      angle_pwm = 2300;
      move_time = 1400;
      place[0] = -120; place[1] = -70; place[2] = 60;
    }
    else if ((11 <= class_id) & (class_id <= 13)) {
      Serial.println("Other garbage");
      angle_pwm = 2500;
      move_time = 1600;
      place[0] = -120; place[1] = -20; place[2] = 60;
    }
```



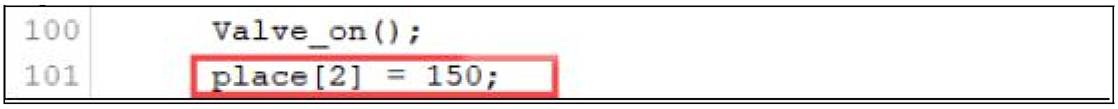
```
        set_position(pos, 600);
        Pump_on();
        delay(1000);
        pos[2] = 150;
```



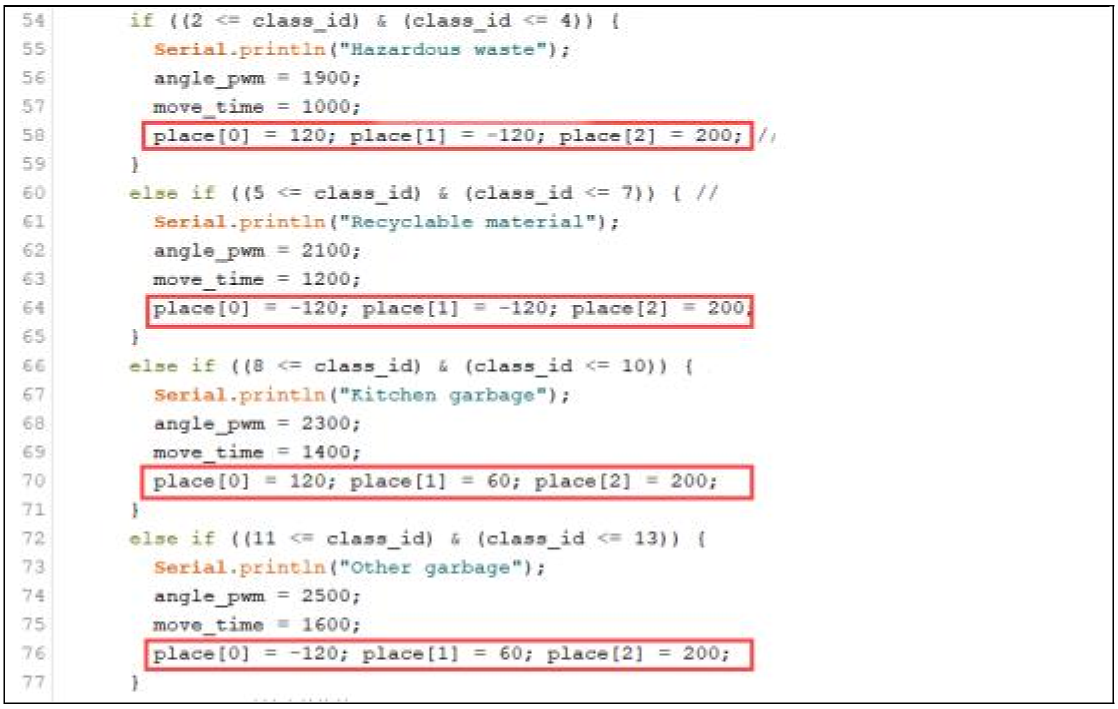
```
        Valve_on();    //
        place[2] = 150;
```

Take modifying the codes in red box as example. The parametrer place_x, place_y and place_z represents the coordinate of x, y, and z axes of the card placement position.

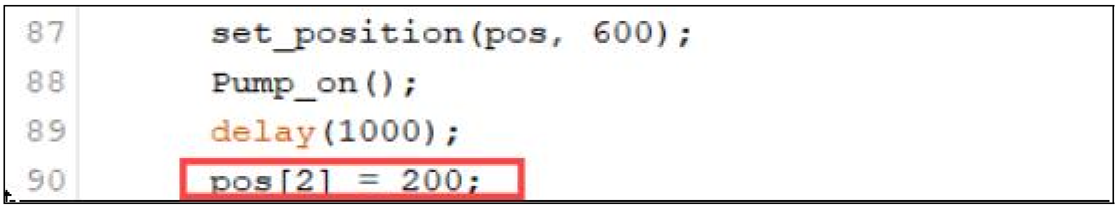
Change the codes in red box to the following codes:



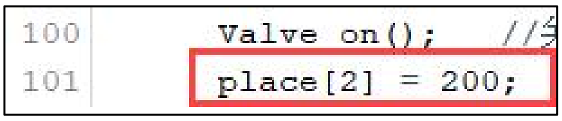
```
        Valve_on();
        place[2] = 150;
```



```
      if ((2 <= class_id) & (class_id <= 4)) {
        Serial.println("Hazardous waste");
        angle_pwm = 1900;
        move_time = 1000;
        place[0] = 120; place[1] = -120; place[2] = 200; //
      }
      else if ((5 <= class_id) & (class_id <= 7)) { //
        Serial.println("Recyclable material");
        angle_pwm = 2100;
        move_time = 1200;
        place[0] = -120; place[1] = -120; place[2] = 200;
      }
      else if ((8 <= class_id) & (class_id <= 10)) {
        Serial.println("Kitchen garbage");
        angle_pwm = 2300;
        move_time = 1400;
        place[0] = 120; place[1] = 60; place[2] = 200;
      }
      else if ((11 <= class_id) & (class_id <= 13)) {
        Serial.println("Other garbage");
        angle_pwm = 2500;
        move_time = 1600;
        place[0] = -120; place[1] = 60; place[2] = 200;
      }
```



```
        set_position(pos, 600);
        Pump_on();
        delay(1000);
        pos[2] = 200;
```



```
        Valve_on();    //
        place[2] = 200;
```

Take the code of hazardous card for example:

place[0] is modified from -120 to 120, which corresponds to the adjustment of the card position from -120 to 120 on x-axis, i.e. the card position is changed from the right to the left and the robot arm moves to the left.

place[1] is modified from -170 to -120, which corresponds to the adjustment of the card position from -170 to -120 on y-axis, i.e. the robot arm moves backwards.

place[2] is modified from 60 to 200, which corresponds to the adjustment of the card position from 60 to 200 on z-axis. Thus robotic arm will lift up.

Therefore, the position of the hazardous card will be adjusted from (-120，-170，60) to (120，-120，200).

The parameters of pos[2] and place[2] are modified from 150 to 200 which represents the height of the robot arm after it picks up the card and the height after it moves to the placement area, respectively.

If want to modify the placement position of other cards, you can refer to the following table to modify the corresponding coordinate parameters:

	Parameter Adjustment	Robot Movement Direction
X axis	Increase	left
	Decrease	Right
Y axis	Increase	Backward
	Decrease	Forward

Z axis	Increase	Upward
	Decrease	Downward

Note：Take yourself as the first person view to determine the movement direction.

After modification, please refer to the steps 8-10 in “3. Program Download” to check the effect.

After the cards are recognized, the kitchen waste will be placed to the rear of robotic arm, the other waste card will be placed to the right rear of the robot arm, the hazardous waste card will be placed to the left front of the robot arm, the recyclable waste will be placed to the right front of the robot arm. Please refer to the following image:

（The props in image just for demonstration）

The specific parameters of the codes are as follow:

Recyclable Waste：place[0] = -120; place[1] = -120; place[2] = 60;

——》 place[0] = -120; place[1] = -120; place[2] = 200;

Kitchen Waste：place[0] = -120; place[1] = -70; place[2] = 60;

——》 place[0] = 120; place[1] = 60; place[2] =200;

Other Waste：place[0] = -120; place[1] = -20; place[2] = 60;

——》 place[0] = -120; place[1] = 60; place[2] = 200;

Hazardous Waste：place[0] = -120; place[1] = -170; place[2] = 60;

——》 place[0] = 120; place[1] = -120; place[2] = 200;