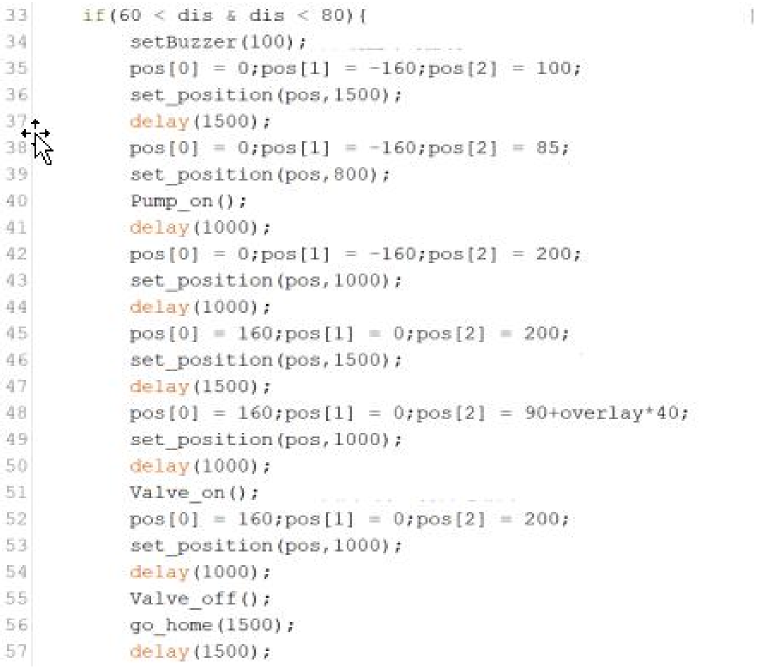
Lesson 2 Ultrasonic Detection and Stacking

1. Working Principle

Ultrasonic sensor can convert ultrasonic signals into other energy signals (usually electrical signals). There are two probes on ultrasonic sensor for receiving and transmitting ultrasound.

The path to the source code of the program is 6.Secondary Development /Arduino Development/Program Files/ Ultrasonic Detection and Stacking/ Ultrasound_palletize/ Ultrasound_palletize.ino.



```
if(60 < dis & dis < 80){
    setBuzzer(100);
    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] = 200;
    set_position(pos,1000);
    delay(1000);
    pos[0] = 160;pos[1] = 0;pos[2] = 200;
    set_position(pos,1500);
    delay(1500);
    pos[0] = 160;pos[1] = 0;pos[2] = 90+overlay*40;
    set_position(pos,1000);
    delay(1000);
    Valve_on();
    pos[0] = 160;pos[1] = 0;pos[2] = 200;
    set_position(pos,1000);
    delay(1000);
    Valve_off();
    go_home(1500);
    delay(1500);
```

Firstly, import the corresponding library and initialize ultrasonic sensor, buzzer, servo and action groups.

Next, the object is detected by ultrasonic sensor and the measured distance is read by I2C protocol. After determining the distance, MaxArm will perform the corresponding action based on the determined result.

Then, execute the functions for controlling action group, buzzer and air pump to suck the detected object to the side and stack it.

2. Preparation

2.1 Hardware

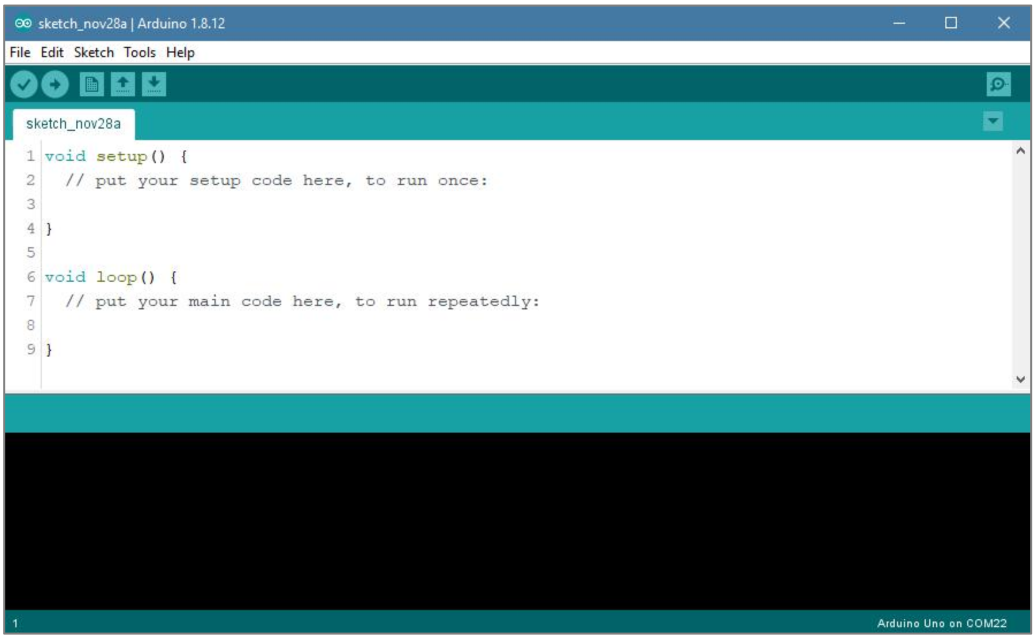
Please assemble the ultrasonic sensor to the corresponding position on MaxArm according to the tutorial in folder “Lesson 1 Sensor Assembly” under the same directory.

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”.

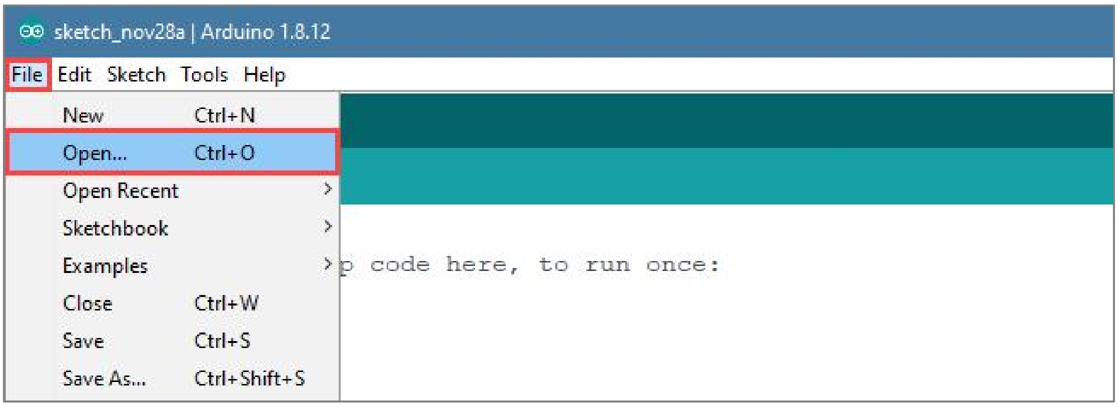
3. Program Download

1) Click on arduino icon to open Arduino IDE.



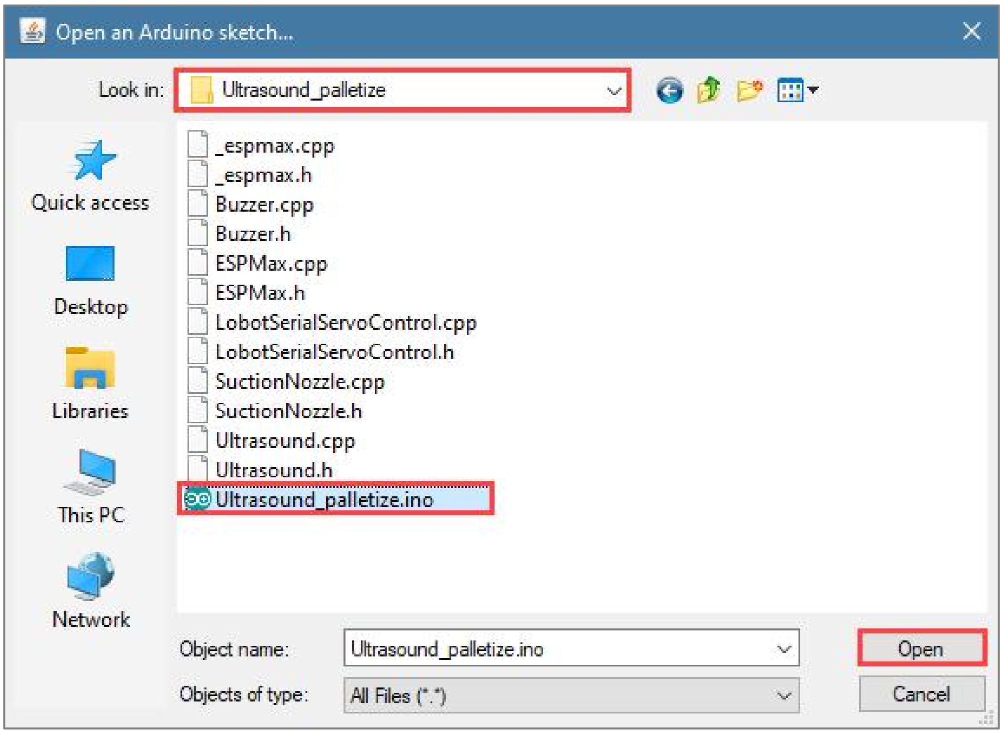
```
void setup() {
  // put your setup code here, to run once:

}

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

}
```

2) Click “File->Open” in turn.


sketch_nov28a | Arduino 1.8.12

p code here, to run once:

3) Select the program “Ultrasound_palletize.ino” in the folder “6.Secondary Development/ Arduino Development/Sensor-extension Game/Program Files/ Ultrasonic Detection and Suction/Ultrasound_clamp”.


Open an Arduino sketch...

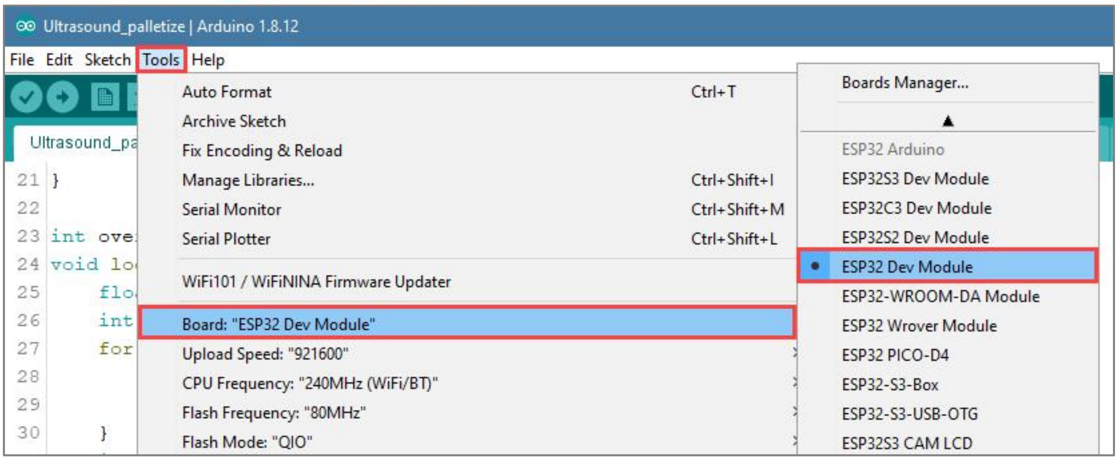
Look in: Ultrasound_palletize

_espmax.cpp
_espmax.h
Buzzer.cpp
Buzzer.h
ESPMax.cpp
ESPMax.h
LobotSerialServoControl.cpp
LobotSerialServoControl.h
SuctionNozzle.cpp
SuctionNozzle.h
Ultrasound.cpp
Ultrasound.h
Ultrasound_palletize.ino

Object name: Ultrasound_palletize.ino

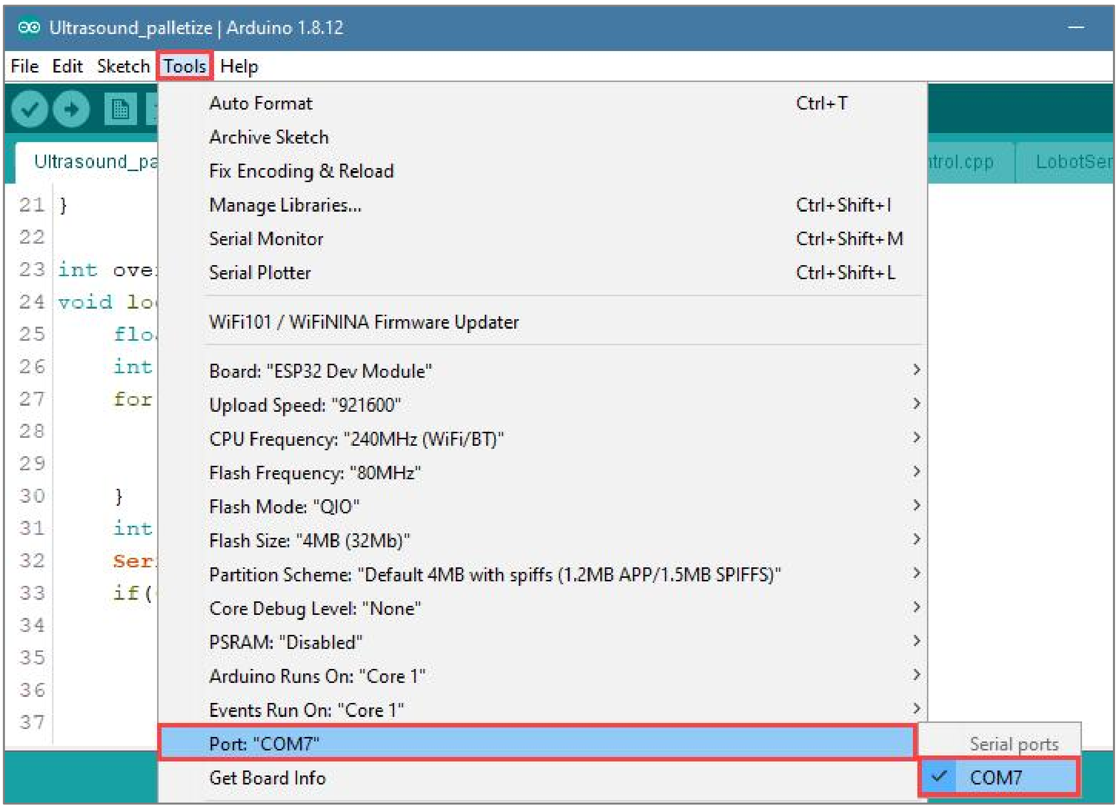
Objects of type: All Files (*.*)

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).



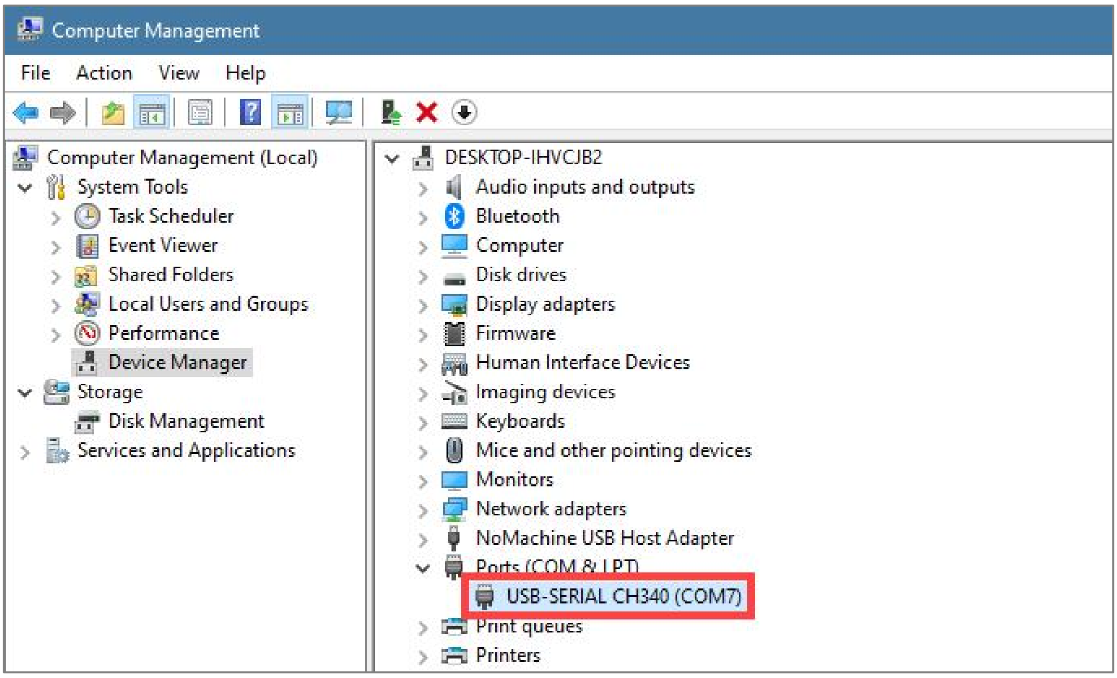
Board: "ESP32 Dev Module"
Upload Speed: "921600"
CPU Frequency: "240MHz (WiFi/BT)"
Flash Frequency: "80MHz"
Flash Mode: "QIO"

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.)

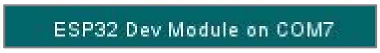

Ultrasound_palletize | Arduino 1.8.12
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Get Board Info
Serial ports
✓ COM7

Board: "ESP32 Dev Module"
Upload Speed: "921600"
CPU Frequency: "240MHz (WiFi/BT)"
Flash Frequency: "80MHz"
Flash Mode: "QIO"
Flash Size: "4MB (32Mb)"
Partition Scheme: "Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS)"
Core Debug Level: "None"
PSRAM: "Disabled"
Arduino Runs On: "Core 1"
Events Run On: "Core 1"
Port: "COM7"

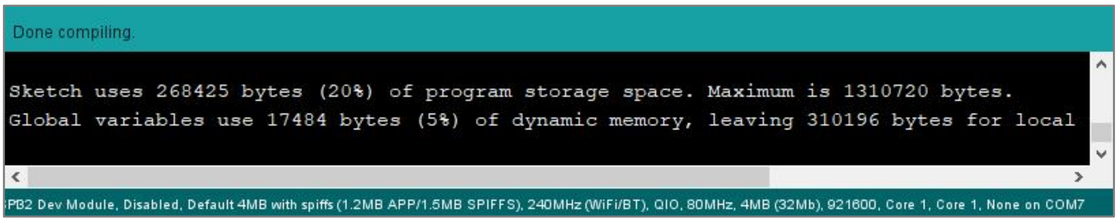
6) If you’re not sure about the port number, please open the “This PC” and click “Properties->Device Manger” in turns to check the corresponding port number (the device is with CH340). Then select the correct port on Arduino editor.



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).


ESP32 Dev Module on COM7

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.

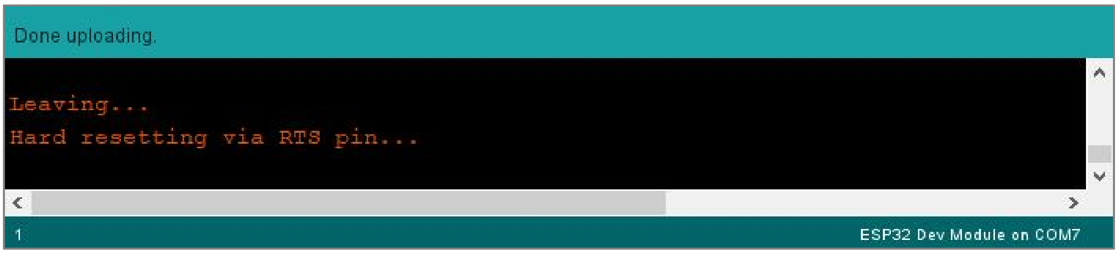

Done compiling.

```
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...
```

4. Project Outcome

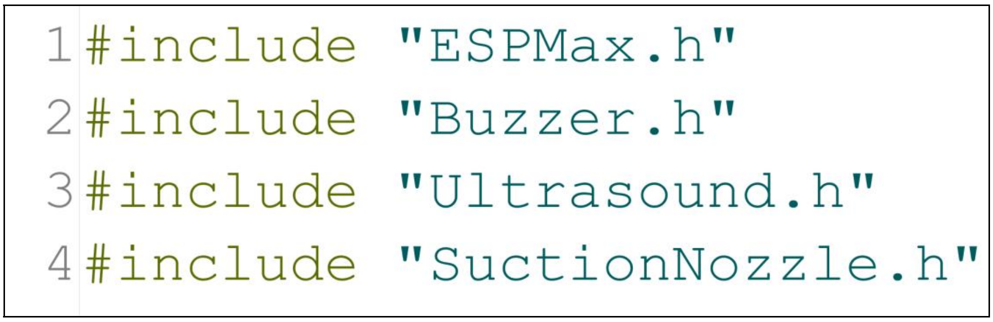
After the block is detected by the ultrasonic sensor, the buzzer will beep once. Then, the suction nozzle will move to the block and suck it after turning on the air pump, and carry the block to the stacking area for stacking. After performing the stacking action three times, MaxArm will return to the initial posture.

5. Program Instruction

5.1 Import library file

The path to the source code of the program is 6.Secondary Development /Arduino Development/Sensor-extension Game/Program Files/ Ultrasonic Detection and Stacking/Ultrasound_palletize/Ultrasound_palletize.ino

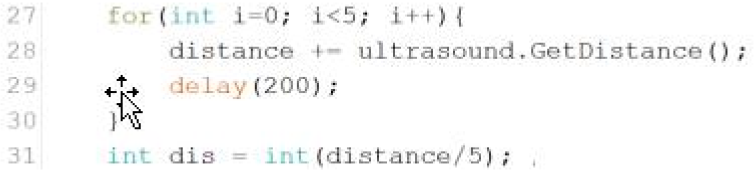
Before running program, the ultrasonic sensor, buzzer, PWM servo, bus servo and air pump and other related library files are called.



```
#include "ESPMax.h"
#include "Buzzer.h"
#include "Ultrasound.h"
#include "SuctionNozzle.h"
```

5.2 Ultrasonic Detection

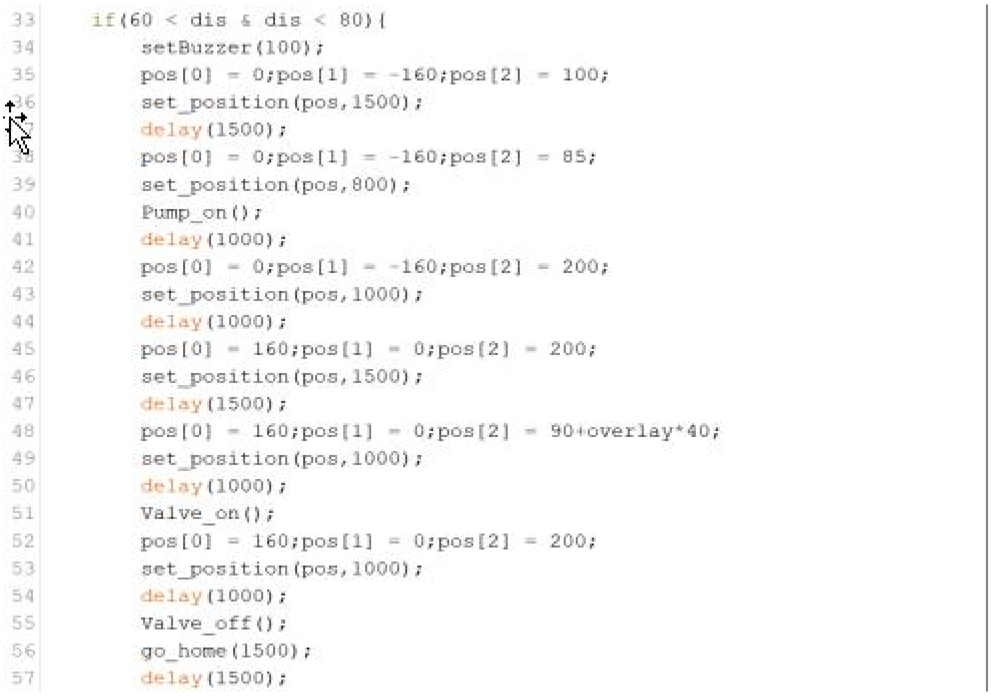
Use the ultrasound.getDistance() function to measure the distance, and then take the average of five detected distance values.



```
    for(int i=0; i<5; i++){
        distance += ultrasound.GetDistance();
        delay(200);
    }
    int dis = int(distance/5);
```

5.3 Control Robotic Arm

Determine the distance of the front object first. If the object is between 60 and 80mm, MaxArm will perform the corresponding action.



```
    if(60 < dis & dis < 80){
        setBuzzer(100);
        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] = 200;
        set_position(pos,1000);
        delay(1000);
        pos[0] = 160;pos[1] = 0;pos[2] = 200;
        set_position(pos,1500);
        delay(1500);
        pos[0] = 160;pos[1] = 0;pos[2] = 90+overlay*40;
        set_position(pos,1000);
        delay(1000);
        Valve_on();
        pos[0] = 160;pos[1] = 0;pos[2] = 200;
        set_position(pos,1000);
        delay(1000);
        Valve_off();
        go_home(1500);
        delay(1500);
```

The buzzer is controlled by using the setBuzze() function. Take the code "setBuzzer(100)" as example.

The first parameter "100" is the sounding time of buzzer and the unit is ms.

The robotic arm is controlled by set_position() function. Take the code "set_position(pos,1000)" as example.

The first parameter "pos" represents the position of the robotic arm on x, y and z axes. Among them, pos[0] represents the coordinate of x axis, pos[1] represents the coordinate of y axis, and pos[2] represents the coordinate of z-axis.

The second parameter "1000" represents the running time and its unit is ms.