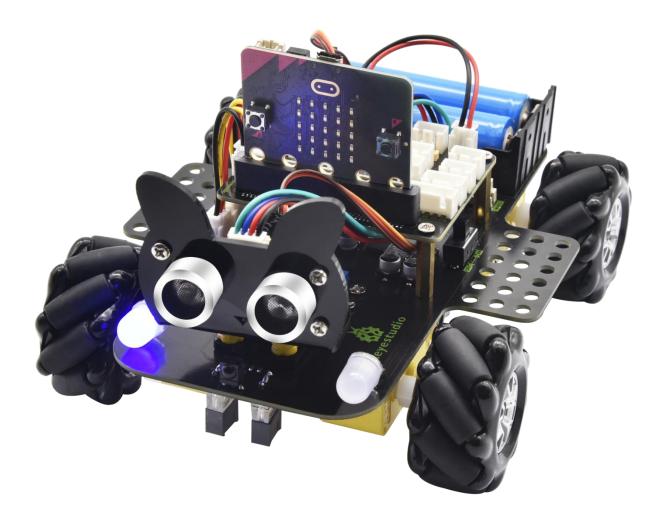


# Keyestudio 4WD Mecanum Robot Car

# (Python)





# Contents

| Keyestudio 4WD Mecanum Robot Car1                                  |
|--|
| 1. Introduction 4  |
| 2. Description 6   |
| 3.Parameters 7   |
| 4.Kit List······7  |
| 5.Preparations: 13   |
| 5.1Background Information about Micro:bit                          |
| (1)What is Micro:bit?13  |
| (2) Comparison between V2.0 & V1.5                                 |
| (3) Pinout18   |
| (4)Notes for the application of Micro:bit main board20             |
| 5.2.Install Micro:bit driver21                                     |
| 6.keyestudio 4WD Mecanum Robot Car22                               |
| 6.1.Basic Information about Keyestudio 4WD Mecanum Robot Car··· 22 |
| 6.2. The Installation of Keyestudio 4WD Mecanum Robot Car24        |
| 7. Python46  |
| 8.Projects 56  |
| Project 1: Heartbeat 57  |
| Project 2: Light A Single LED 70                                   |
| Project 3: LED Dot Matrix76  |
| Project 4: Programmable Buttons86                                  |



|      | Project 6: Geomagnetic Sensor                       | 99  |
|------|---|-----|
|      | Project 5: Temperature Detection                    | 99  |
|      | Project 7: Accelerometer                            | 120 |
|      | Project 8: Light Detection                          | 133 |
|      | Project 9: Speaker                                  | 139 |
|      | Project 10: Touch-sensitive Logo                    | 143 |
|      | Project 11: Microphone                              | 149 |
|      | Project 12: Touch-sensitive Logo Controlled Speaker | 159 |
|      | Project 13:Colorful Lights                          | 165 |
|      | Project 14:WS2812 RGB LEDs                          | 176 |
|      | Project 15:Servo                                    | 195 |
|      | Project 16:Motor                                    | 204 |
|      | Project 17: Line Tracking Sensor                    | 226 |
|      | 17.1: Detect Line Tracking Sensor                   | 226 |
|      | 17.2: Line Tracking Smart Car                       | 238 |
|      | Project 18: Ultrasonic Following Smart Car          | 250 |
|      | 18.1: Ultrasonic Ranging                            | 250 |
|      | 18.2: Ultrasonic Avoidance Car                      | 260 |
|      | 18.3: Ultrasonic Following Smart Car                | 274 |
| 9. R | Resources   | 283 |



## 1. Introduction

Have you wondered to learn programming or have your own programming robot? Nowadays, programming has developed to a lower age group, and it will be a trend for everyone to be able to program thanks to the spread of simple graphical programming platforms, from micro:bit to Arduino and Raspberry Pi. Maybe you haven't heard of them before. It doesn't matter because with the help of this product and tutorial, you can easily install a multi-functional programming car and experience the fun of being a maker.

Micro:bit is a highly integrated microcontroller of powerful functions and small size. It is very suitable to be applied in STEAM education for it functions to make robots, wearable devices and electronic interactive games via the combination of code programming and graphical programming.

This Keyestudio 4WD Mecanum Robot Car is a smart DIY car specially designed for micro:bit. The smart car kit consists of a car body with extended functions, a PCB base plate with integrated motor drive sensors, 4 decelerating DC motors, Mecanum wheels, various modules and sensors



and acrylic boards. Therefore, you can easily assemble a cool Mecanum wheel 4WD smart car by yourself.

This tutorial programs in MicroPython language which is the Micro:bit version of Python language. It will guide you to use software Mu to write MicroPython language for Micro:bit main board to control the smart home system. In this process, not only can you enhance your ability to make stuffs but also learn the skills of programming.

Python is one of the most popular programming language especially in machine learning for its availability and accessibility have brought huge convenience to this field. However, MicroPython is a lean and efficient implementation of the Python programming language for microcontrollers and embedded systems.

and then use Microsoft's online graphical programming platform Make Code to program the micro:bit control board to control the car. In the process, you can not only experience the fun of creation but enhance hands-on ability and learn programming skills as well.

This tutorial is a Python tutorial for 4WD Mecanum Robot Car. If you



haven't learned the basic tutorial (Makecode version of Tutorial), we strongly recommend you to learn it first. Because the basic one is programmed using graphical blocks, which is easier to understand and start.

For your convenience, source code written in Python has been provided in every project, as well as code programming steps and code explanation in details. Hope you can better understand them.

#### 2. Description

This product is a smart car based on Micro:bit. It boasts multiply functions including ultrasonic sound following, line tracking, infrared control and Bluetooth control. It comes with a passive buzzer which is able to play music, 4 WS2812RGB LEDs to display different colors, 2 colorful lights to make direction lights for the car. This product uses two 18650 lithium batteries for power supply.

When installing and disassembling the battery, please pay attention to the positive and negative poles of the battery, and be sure not to reverse the them. By the way, the motor speed of this product is adjustable.



In order to provide you with better experience, corresponding documents about installation and test code are also provided.

#### **3.Parameters**

- Connector port input: DC 6V---9V
- Operating voltage of drive board system: 5V
- Standard operating power consumption: about 2.2W
- Maximum power: Maximum output power is 12W
- Motor speed: 200RPM/1min
- ◆ Working temperature range: 0-50°C
- Size: 120\*120\*120mm
- Environmental protection attributes: ROHS

Note: working voltage of micro:bit is 3.3V, driver shield integrates 3.3V/5V communication conversion circuit.

#### 4.Kit List

| # | Picture | Components | Quanti |
|---|---------|------------|--------|
|   |         |            | ty     |



| 1 |           | KS0511 Acrylic Board T=3mm             | 1 |
|---|-----------|--|---|
| 2 |           | Acrylic Board with Lego Holes<br>T=3mm | 1 |
| 3 |           | 4.5V Motor                             | 4 |
| 4 | • • • • • | 23*15*5MM Fixing Board                 | 4 |
| 5 |           | Servo                                  | 1 |



| 6 |           | Mecanum Wheels   | 4 |
|---|-----------|--|---|
| 7 |           | Keyestudio Micro:bit IO Port<br>Expansion Sensor Shield With<br>Level Conversion       | 1 |
| 8 | micro:bit | Micro:bit Main Board V2.0 with<br>Package for KS4031<br>Micro:bit Main Board V2.0 with | 1 |
|   |           | Package for KS4032   | 0 |
| 9 |           | Keyestudio Driver Board  | 1 |



| 10 | M3*20MM Dual-pass Copper<br>Pillar | 4  |
|----|------------------------------------|----|
| 11 | 4265c Lego Part                    | 4  |
| 12 | 43093 Lego Part                    | 4  |
| 13 | Acrylic Gasket Six in One Pack     | 1  |
| 14 | M3*6MM Round Head Screw            | 18 |



| 15 | keyestudio  | Keyestudio Ultrasonic Module          | 1  |
|----|---|---------------------------------------|----|
| 16 |   | M3 Nickle-plated Nut                  | 14 |
| 17 |   | M3*30MM Round Head Screw              | 9  |
| 18 | 000   | M2 Nickle-plated Nut                  | 3  |
| 19 |   | M2*8MM Round Head Screw               | 3  |
| 20 |   | M3*8MM Round Head Screw               | 5  |
| 21 | <ul> <li>A</li> <li>C</li> <li>C</li></ul> | Remote Control (without<br>batteries) | 1  |
| 22 |   | Plastic String 3*100mm                | 5  |



| 23 |  | USB Cable                                 | 1 |
|----|--|---|---|
| 24 |  | HX-2.54 2P DuPont Wire<br>100mm           | 1 |
| 25 |  | HX-2.54 4P DuPont Wire<br>50mm            | 2 |
| 26 |  | XH2.54 4P DuPont Wire<br>160mm            | 1 |
| 27 |  | XH2.54 3P DuPont Wire 50mm                | 2 |
| 28 |  | 3*40mm Screwdriver                        | 1 |
| 29 | Onteentities Onteentities<br>Onteentities Onteentities | M1.2*5mm Round Head<br>Self-tapping Screw | 6 |



#### **5.Preparations:**

#### 5.1Background Information about Micro:bit

#### (1)What is Micro:bit?

Micro:bit is an open source hardware platform based on the ARM architecture launched by British Broadcasting Corporation (BBC) together with ARM, Barclays, element14, Microsoft and other institutions. The core device is a 32-bit Arm Cortex-M4 with FPU micro-processing.

Though it is just the size of a credit card, the Micro:bit main board is equipped with loads of components, including a 5\*5 LED dot matrix, 2 programmable buttons, an accelerometer, a compass, a thermometer, a touch-sensitive logo and a MEMS microphone, a Bluetooth module of low energy, and a buzzer and others. Thus, it also boasts multiple functions.

The buzzer built in the other side of the board makes playing all kinds of sound possible without any external equipment. The golden fingers and gears added provide a better fixing of crocodile clips. Moreover, this board has a sleeping mode to lower power consumption of batteries and it can be entered if users long press the Reset & Power button on the back of it. It is capable of reading the data of sensors, controlling servos and RGB lights and attaching with a shield so as to connect with various sensors. It also supports a variety of codes and graphical programming platforms, and is compatible with almost all PCs and mobile devices. It has no need to install drivers. It is of high integration of electronic modules, and has a serial port

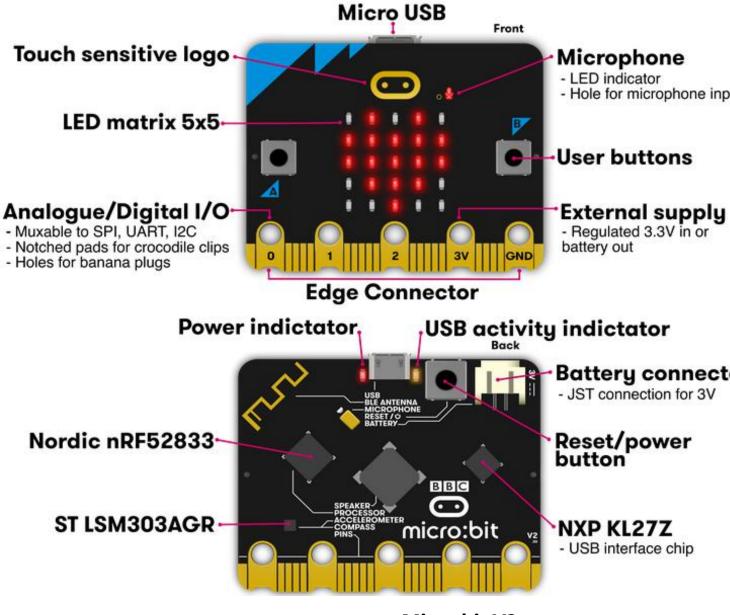


monitoring function for easy debugging.

The board has found wide applications. It can be applied in programming video games, making interactions between light and sound, controlling a robot, conducting scientific experiments, developing wearable devices and make some cool inventions like robots and musical instruments, basically everything imaginable.

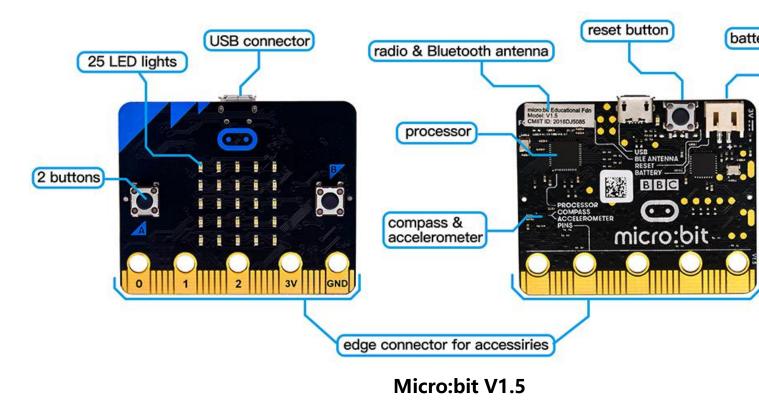
## (2)Layout





Micro:bit V2





## (2) Comparison between V2.0 & V1.5





|                         | V1.5  | V2  |  |  |
|-------------------------|---|---|--|--|
| PROCESSOR               | Nordic Semiconductor<br>nRF51822                              | Nordic Semiconductor<br>nRF52833  |  |  |
| MEMORY                  | 256KB Flash, 16KB RAM   | 512KB Flash, 128KB RAM  |  |  |
| INTERFACECHIP           | NXP KL26Z, 16KB RAM   | NXP KL27Z, 32KB RAM   |  |  |
| MICROPHONE              | N/A   | MEMS microphone and<br>LED indicator  |  |  |
| SPEAKER                 | N/A   | On board speaker  |  |  |
| TOUCH                   | N/A   | Touch sensitive logo  |  |  |
| EDGE                    |   | ,SPI and Extension interface.<br>ctin crocodile clips/banana plugs.   |  |  |
| CONNECTOR               | 3 dedicated GPIO  | 4 dedicated GPIO<br>Notched for easier connection   |  |  |
| 12C                     | Shared (mux) I2C bus  | Dedicated I2C bus   |  |  |
| WIRELESS                | 2.4GHz Radio/BLE Blutooth 4.0                                 | 2.4GHz Radio/BLE Blutooth 5.0   |  |  |
| POWER                   | Micro USB 5V power supply,<br>3V port or battery power supply | Micro USB 5V power supply,<br>3V port or battery power supply<br>LED Indicator, Power off (push and<br>hold power button) |  |  |
| CURRENT<br>AVAILABLE    | 90mA  | 200mA   |  |  |
| MOTION<br>SENSOR        | ST LSM 303  |   |  |  |
| PROGRAMMING<br>SOFTWARE | C++, Makecode, Python, Scratch                                |   |  |  |
| SIZE                    | 5cm(W) x 4cm(H)   |   |  |  |

For the Micro: Bit main board V2, pressing the Reset & Power button, it will reset the Micro: Bit and rerun the program. If you hold it tight, the red LED will slowly get darker. When the power indicator flickers into darkness,



releasing the button and your Micro: Bit board will enter sleep mode for power saving .This will make your battery more durable. And you could press this button again to 'wake up' your Micro:bit.

For more information, please resort to following links:

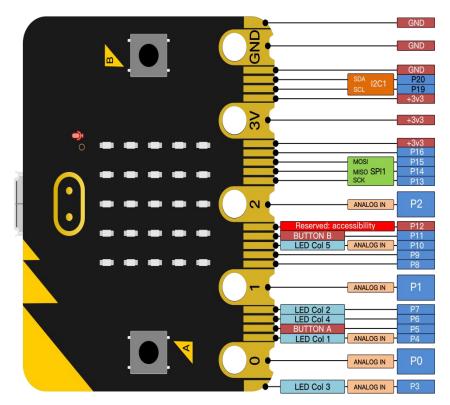
https://tech.microbit.org/hardware/

https://microbit.org/new-microbit/

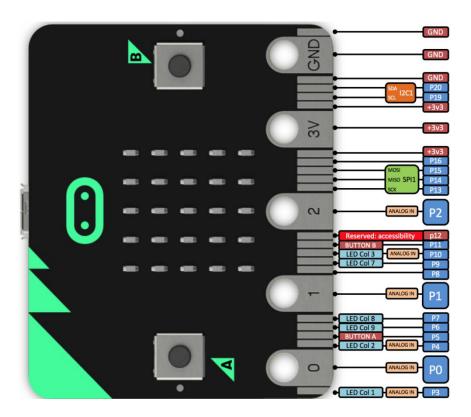
https://www.microbit.org/get-started/user-guide/overview/

https://microbit.org/get-started/user-guide/features-in-depth/

## (3) Pinout







The functions of pins:

| CDIO        | P0, P1, P2, P3, P4, P5, P6, P7, P8, P9, P10, P11, P12,  |
|-------------|---|
| GPIO        | P13, P14, P15, P16, P19, P20                            |
| ADC/DAC     | P0, P1, P2, P3, P4, P10                                 |
| IIC         | P19 (SCL) , P20 (SDA)                                   |
| SPI         | P13 (SCK) , P14 (MISO) , P15 (MOSI)                     |
| PWM (used   |   |
| frequently) | P0, P1, P2, P3, P4, P10                                 |
| PWM (not    |   |
| frequently  | P5、P6、P7、P8、P9、P11、P12、P13、P14、P15、P16、P19、             |
| used)       | P20   |
| Occupied    | P3(LED Col3), P4(LED Col1), P5(Button A), P6(LED Col4), |



# P7(LED Col2), P10(LED Col5), P11(Button B)

Browse the official website for more details: <u>https://tech.microbit.org/hardware/edgeconnector/</u> https://microbit.org/guide/hardware/pins/

## (4)Notes for the application of Micro:bit main board

a. It is recommended to cover it with a silicone protector to prevent short circuit for it has a lot of sophisticated electronic components.

b. Its IO port is very weak in driving since it can merely handle current less than 300mA. Therefore, do not connect it with devices operating in large current, such as servo MG995 and DC motor or it will get burnt.
Furthermore, you must figure out the current requirements of the devices before you use them and it is generally recommended to use the board together with a Micro:bit shield.

c. It is recommended to power the main board via the USB interface or via the battery of 3V. The IO port of this board is 3V, so it does not support sensors of 5V. If you need to connect sensors of 5 V, a Micro: Bit expansion board is required. d. When using pins(P3, P4, P6, P7 and P10)shared with the LED dot matrix, blocking them from the matrix or the LEDs may display randomly and the data about sensors connected maybe wrong.

e. Pin 19 and 20 can not be used as IO ports though the Makecode shows they can. They can only be used as I2C communication.

f. The battery port of 3V cannot be connected with battery more than 3.3V or the main board will be damaged.

g. Forbid to operate it on metal products to avoid short circuit.

To put it simple, Micro:bit V2 main board is like a microcomputer which has made programming at our fingertips and enhanced digital innovation. And as for programming environment, BBC provides a website:

https://microbit.org/code/, which has a graphical MakeCode program easy for use.

# 5.2.Install Micro:bit driver

Micro:bit is free of driver installation. However, in case your computer fail to recognize the main board, you can install the diver too.

Just enter the link <u>https://fs.keyestudio.com/KS4031-4032</u>



to download the driver file **e** mbed\_usb\_2020\_x64\_1212.exe of micro:bit in file folder **1**. Install Microbit Driver

## 6.keyestudio 4WD Mecanum Robot Car

This chapter will introduce the function and structure of keyestudio 4WD Mecanum Robot Car. It is a programmable car based on BBC micro:bit. Driven by motors, it boasts a line tracking sensor and an infrared receiver integrated into the bottom plate, an ultrasonic sensor, servos ,2 colorful lights, 4 WS2812 RGB lights. The wiring is not complicated and it has Lego jacks to facilitate connection with other peripheral devices. Abundant hardware resources will enable you to master more knowledge and skills, so that you can use your imagination to create more technological inventions.

#### 6.1.Basic Information about Keyestudio 4WD Mecanum Robot Car

This car can help you to better learn to use Micro:bit and obtain electronic knowledge.

**Components:**an ultrasonic sensor, servos ,2 colorful lights, 4 WS2812 RGB lights 4 decelerating DC motors, Mecanum wheels,



| Sensor | Colorful | Decelerat | Servo | Ultrasonic | Line     | Infrared | WS2812 | Power  |
|--------|----------|-----------|-------|------------|----------|----------|--------|--------|
|        | light    | ing DC    |       | sensor     | Tracking | Receive  | RGB    | switch |
|        |          | motor     |       |            | Sensor   | r        | light  |        |
|        |          |           |       |            |          |          |        |        |
| #      | 2        | 4         | 1     | 1          | 1        | 1        | 4      | 1      |

Note: the line tracking sensor, WS2812 RGB lights and infrared receiver

#### servo are integrated in the base.

#### Pins:

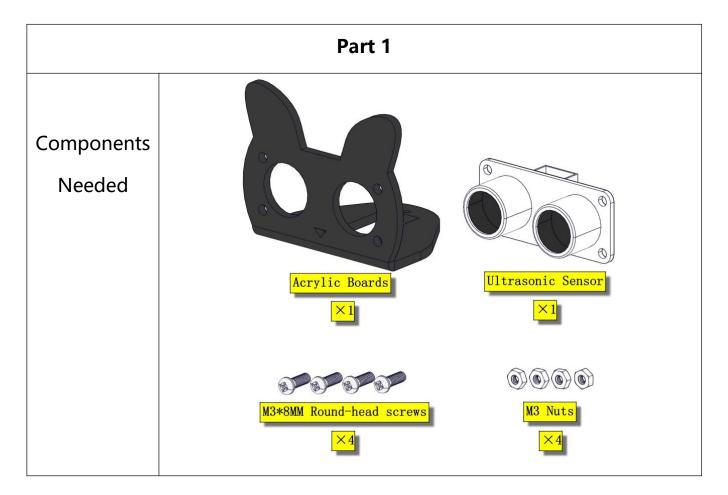
| Pin on Micro:bit | Sensors of the keyestudio |
|------------------|---------------------------|
|                  | 4WD Mecanum Robot Car     |
| P1 P2            | Line Tracking Sensor      |
| P14              | Servo                     |
| P8               | 4个WS2812RGB Lights        |
| Р9               | Infrared Receiver         |
| P15P16           | Ultrasonic Sensor         |

#### Power supply and Battery

The keyestudio 4WD Mecanum Robot Car is powered by two 18650 batteries. The battery holder of the car is compatible with any type of 18650 lithium battery (rechargeable). You can use a universal battery charger to charge the 18650 lithium battery.

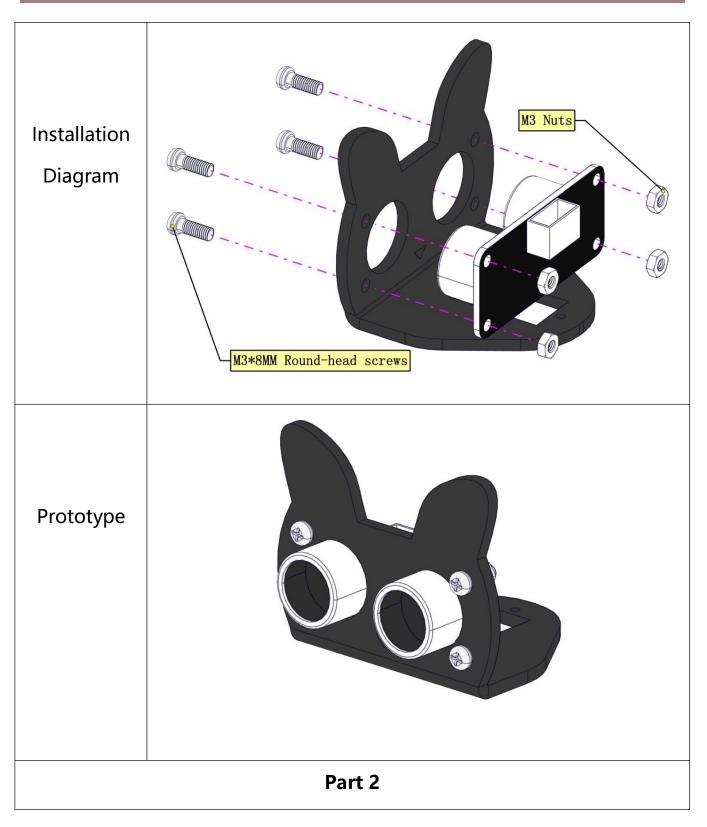


Please note: This product does not contain batteries.

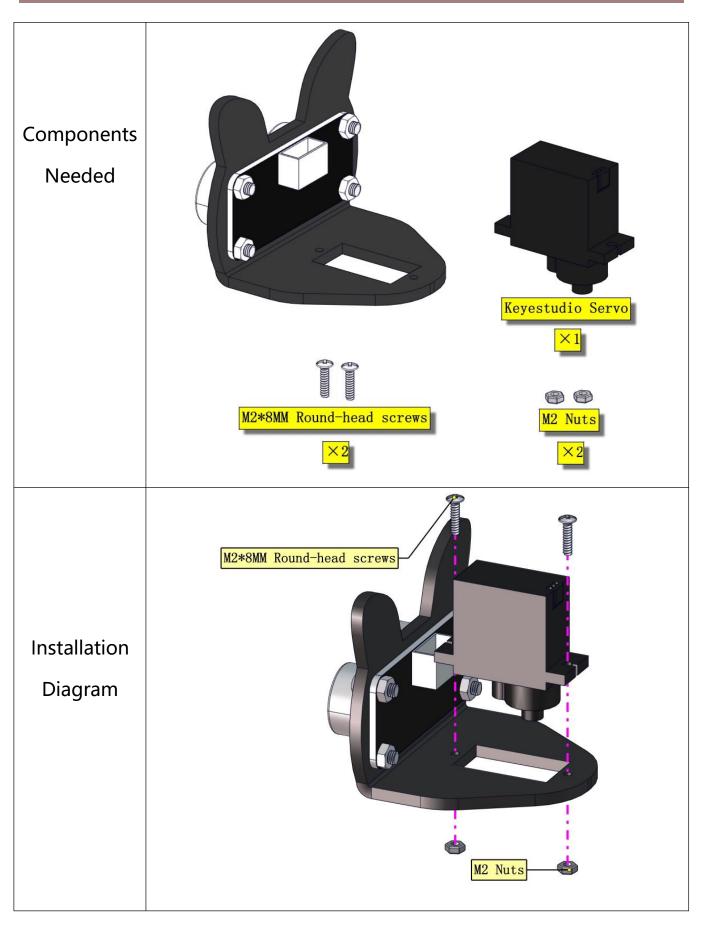


## 6.2. The Installation of Keyestudio 4WD Mecanum Robot Car

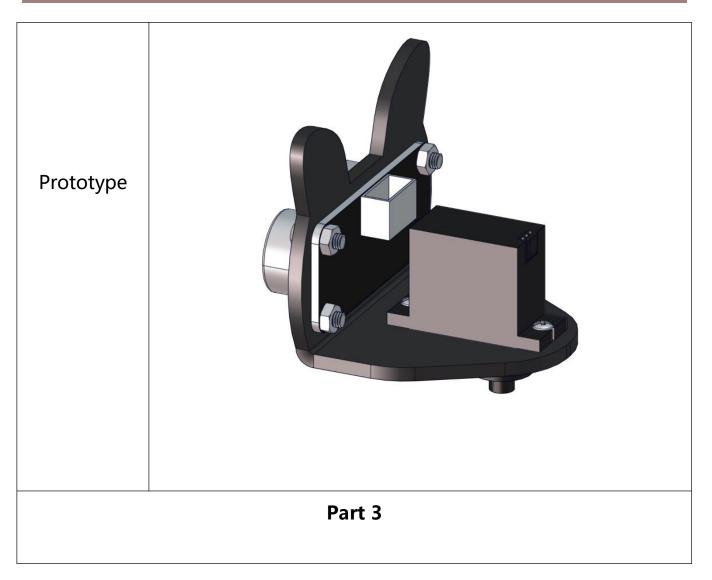




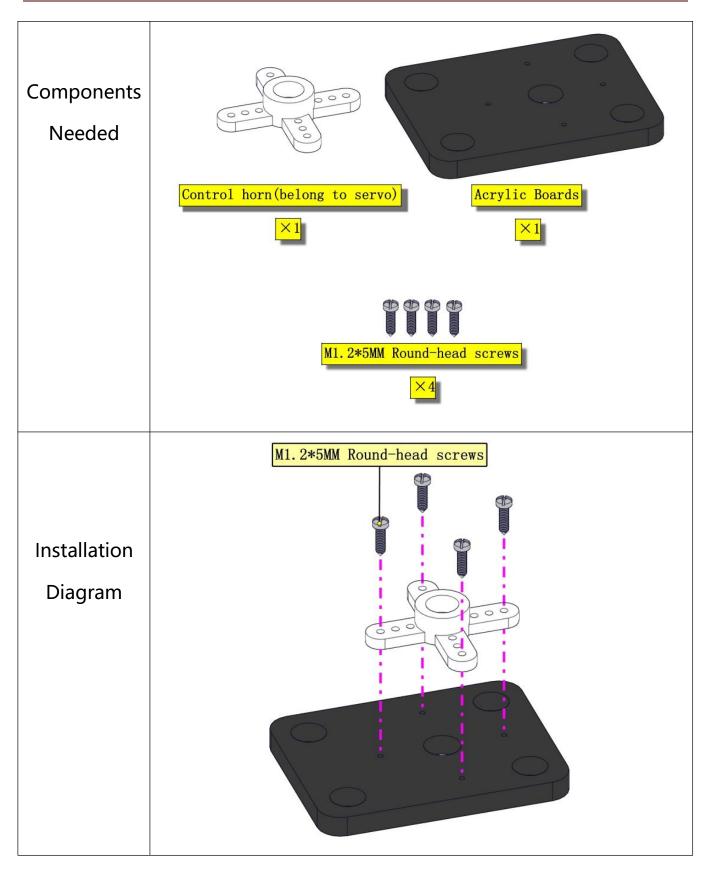




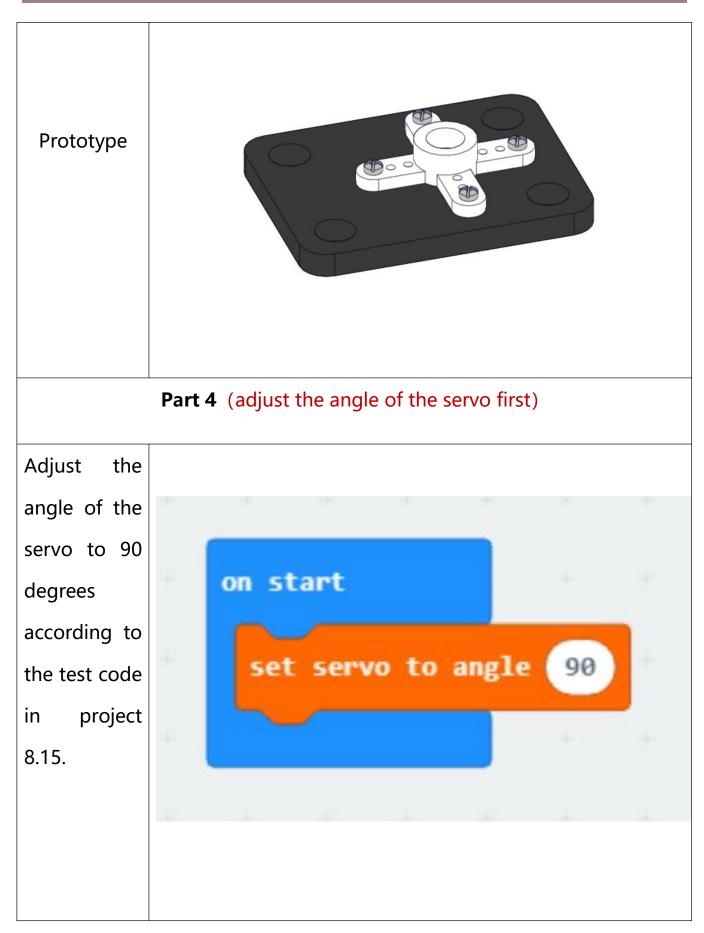




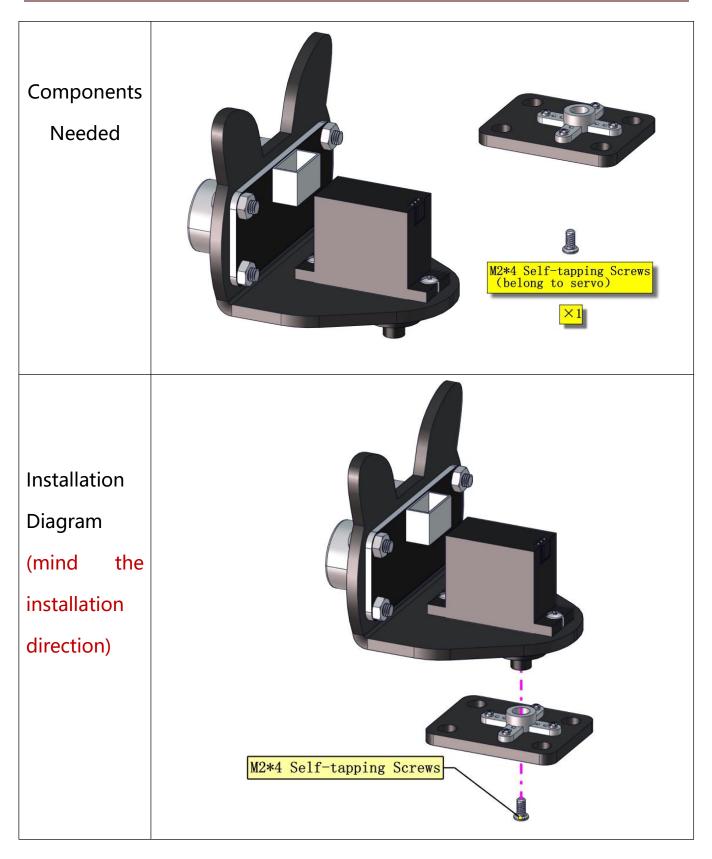




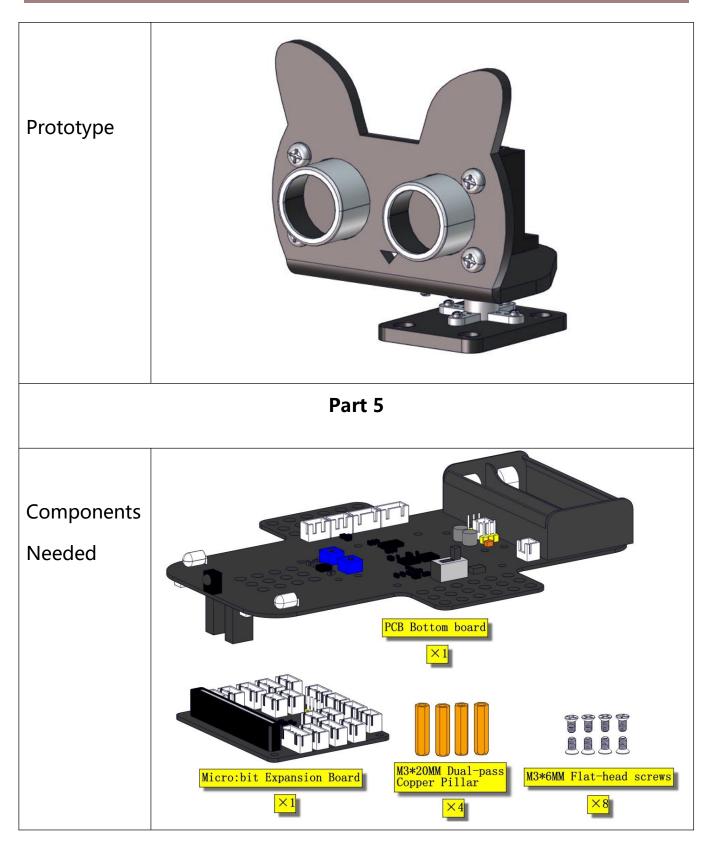




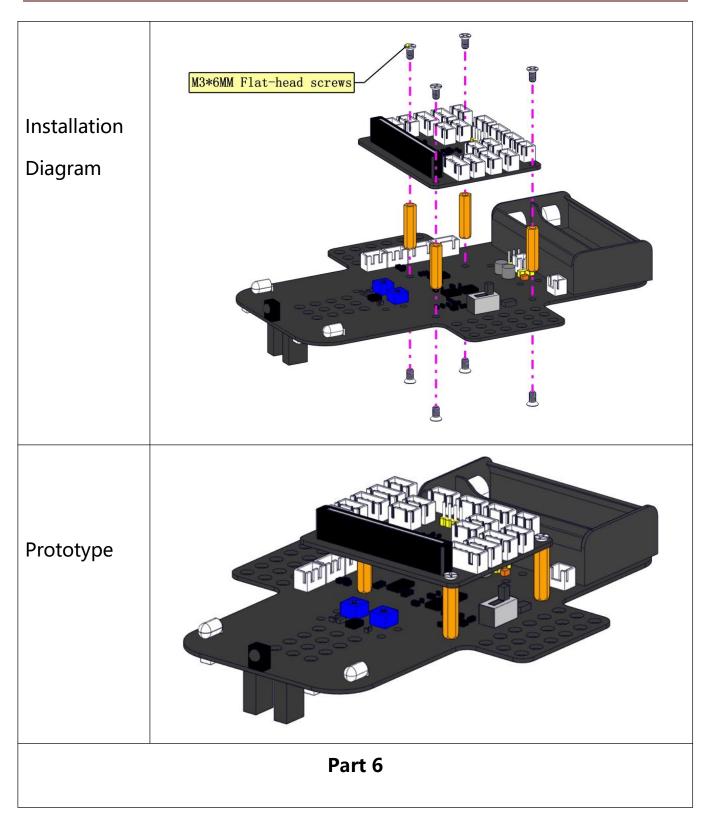




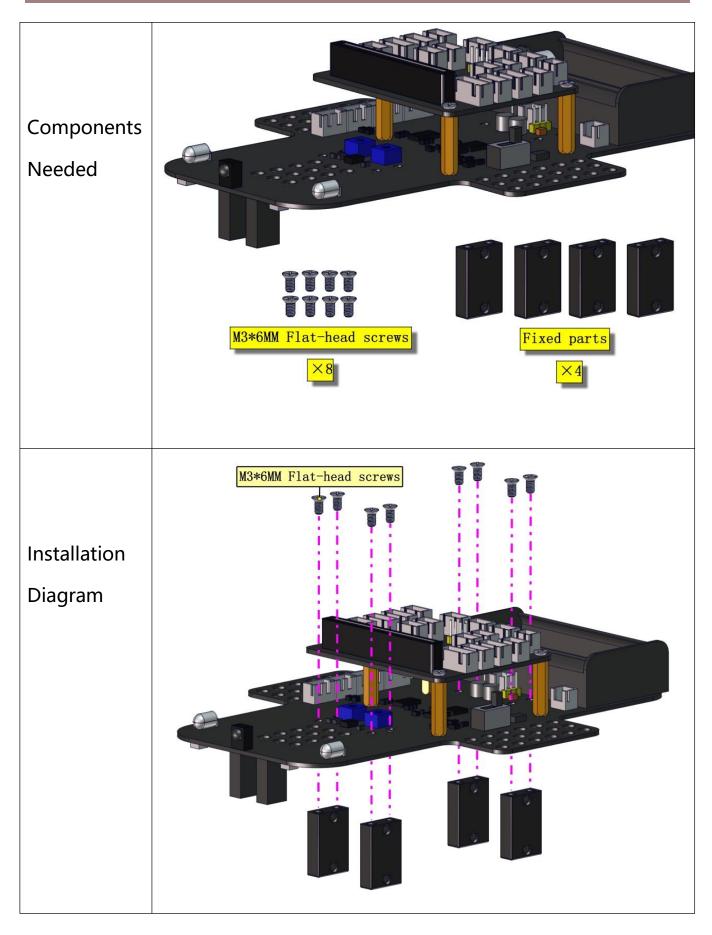




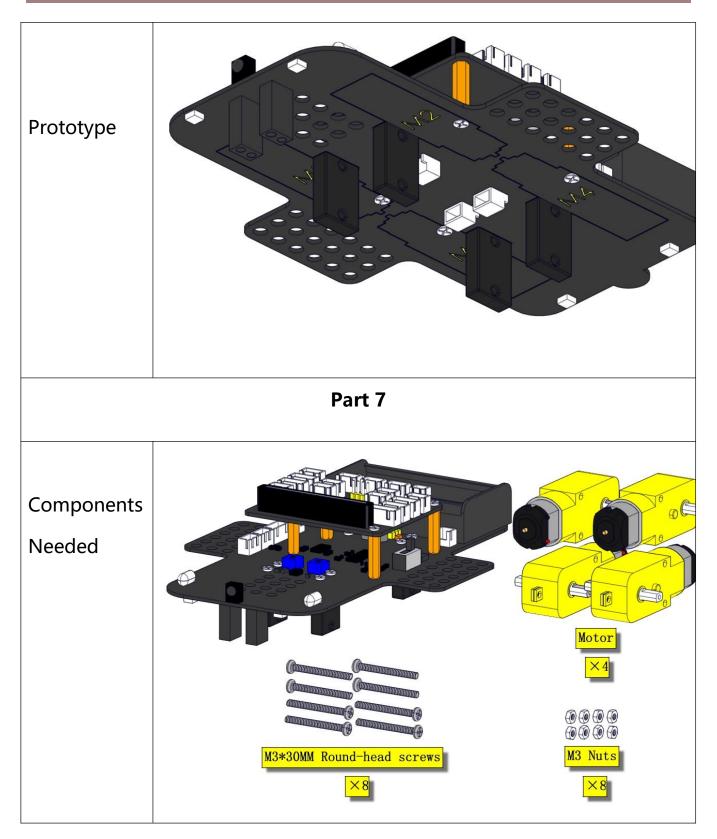




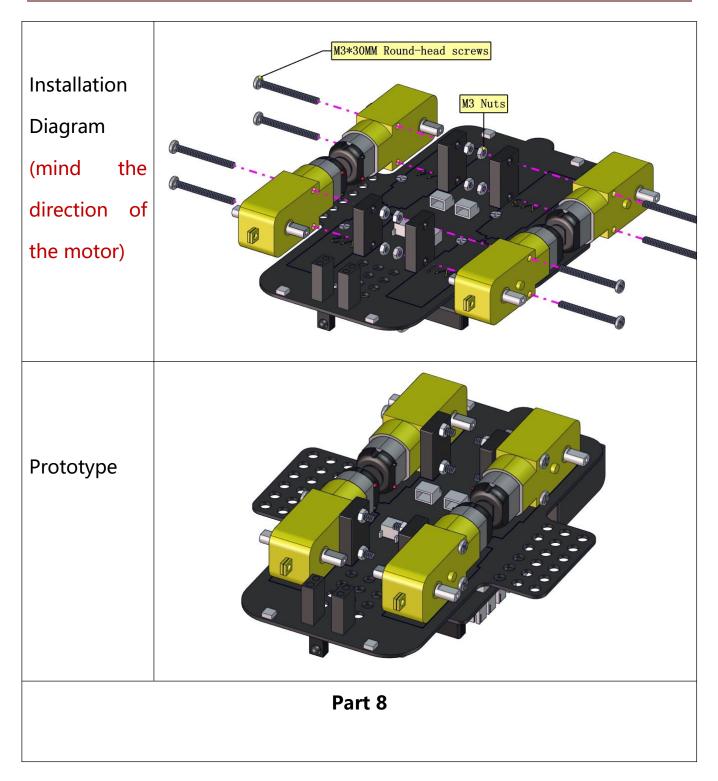




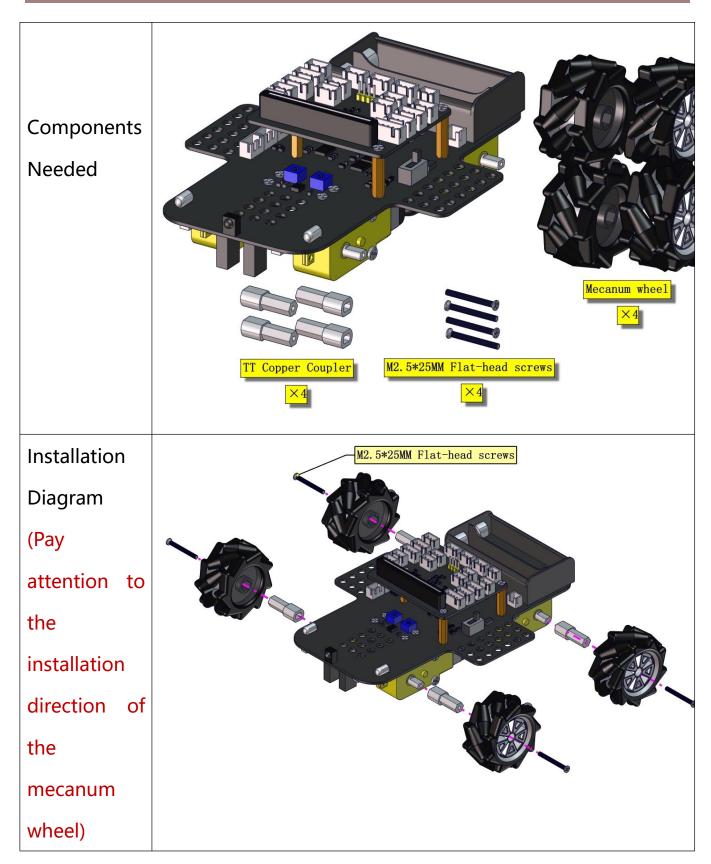




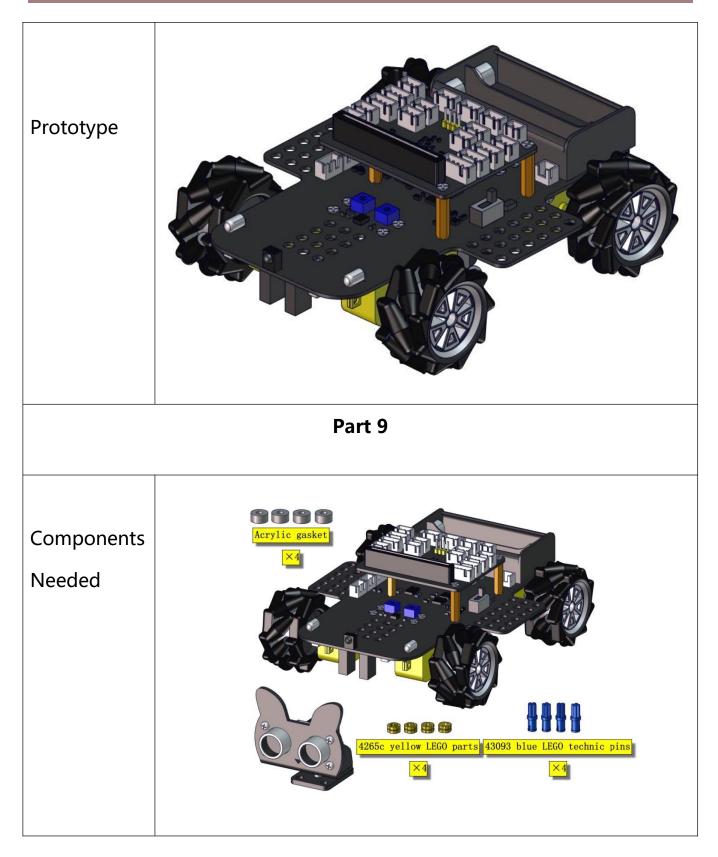




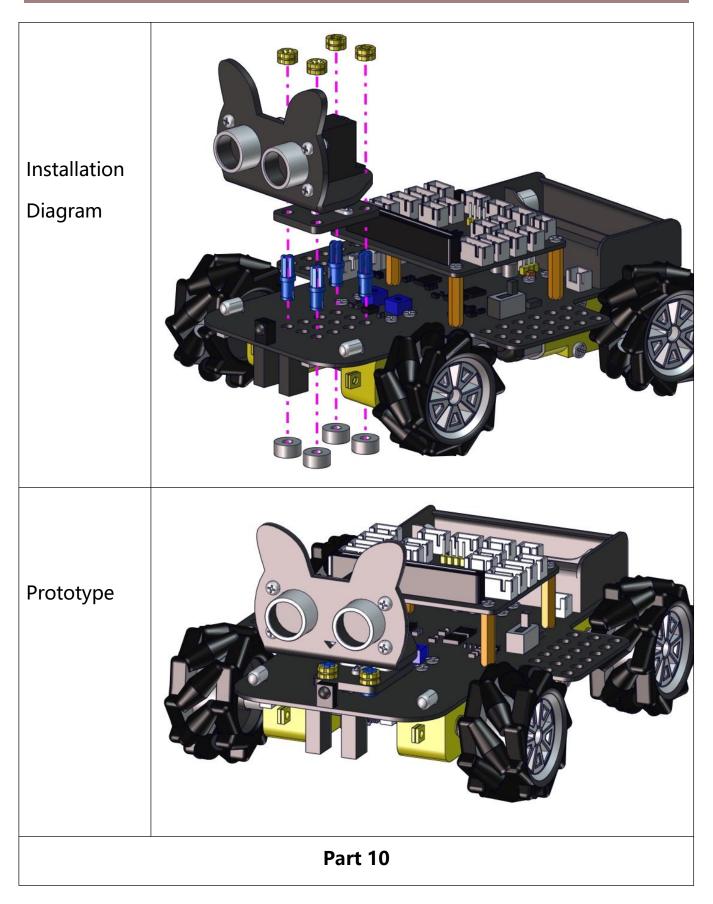




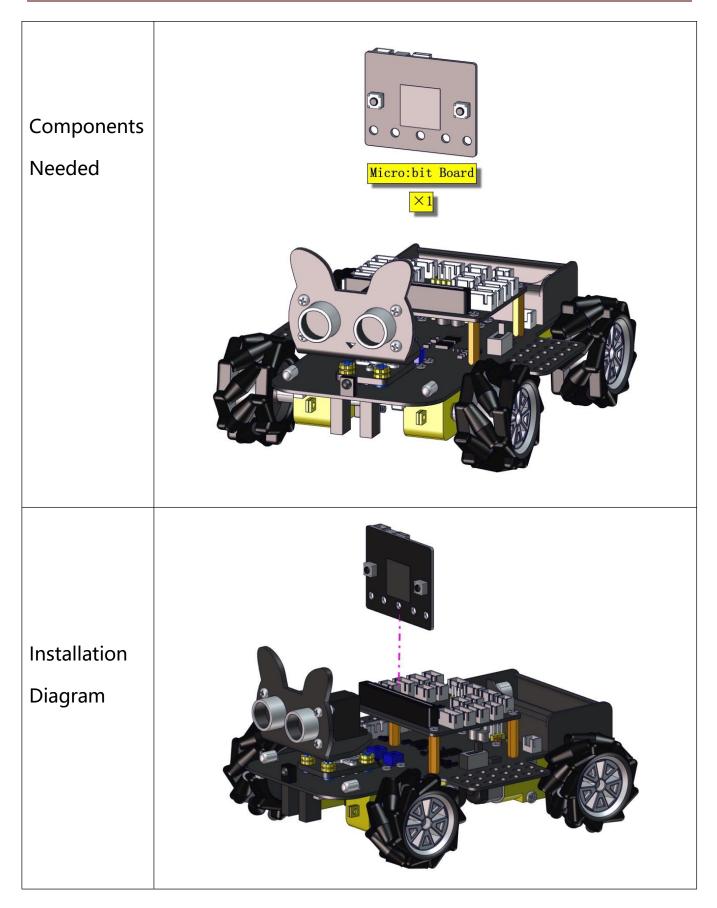




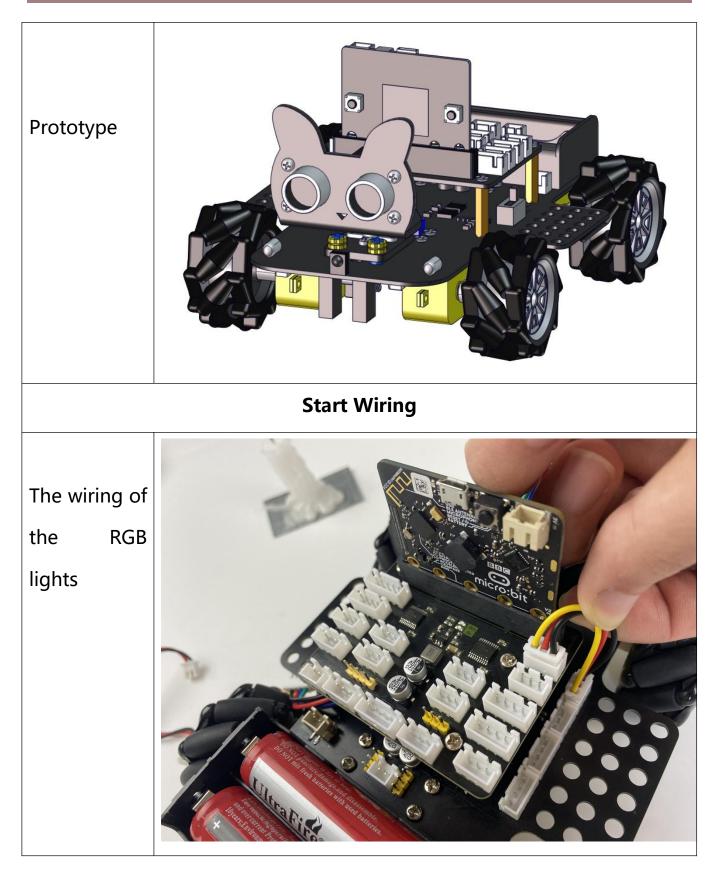




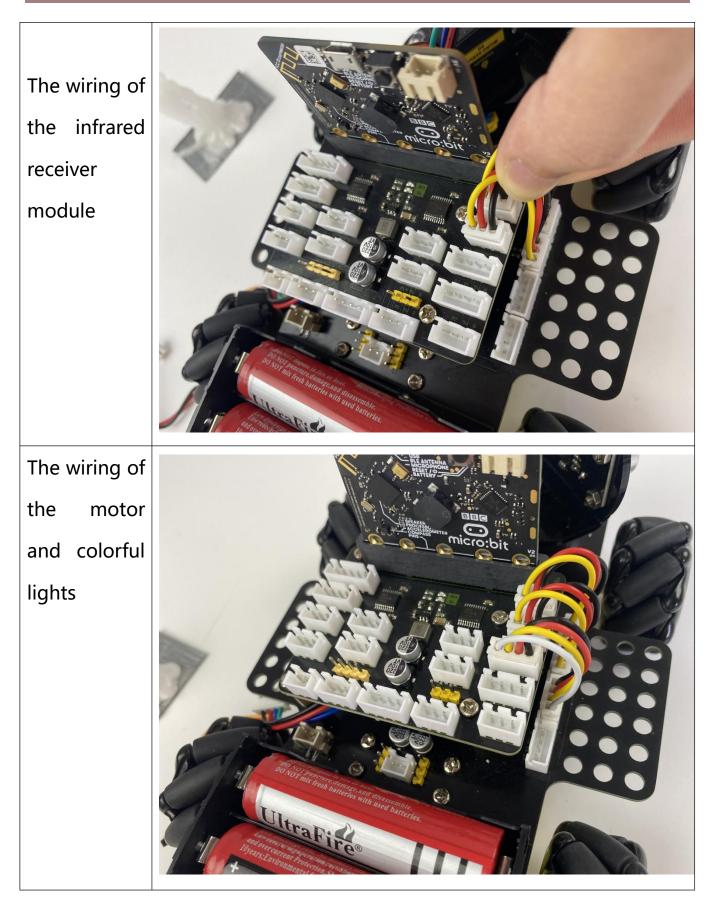




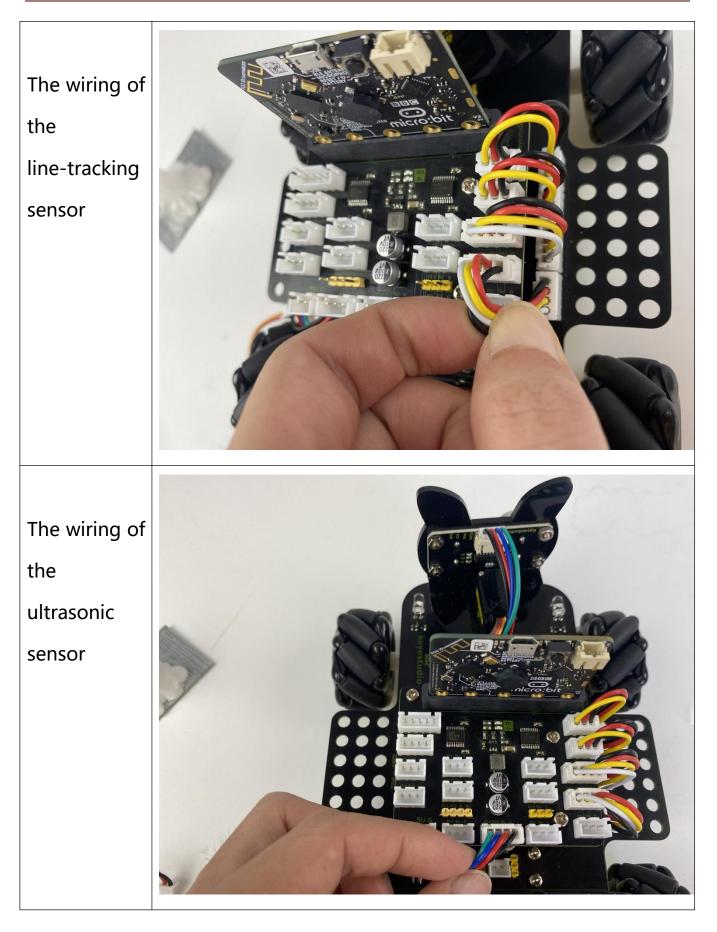




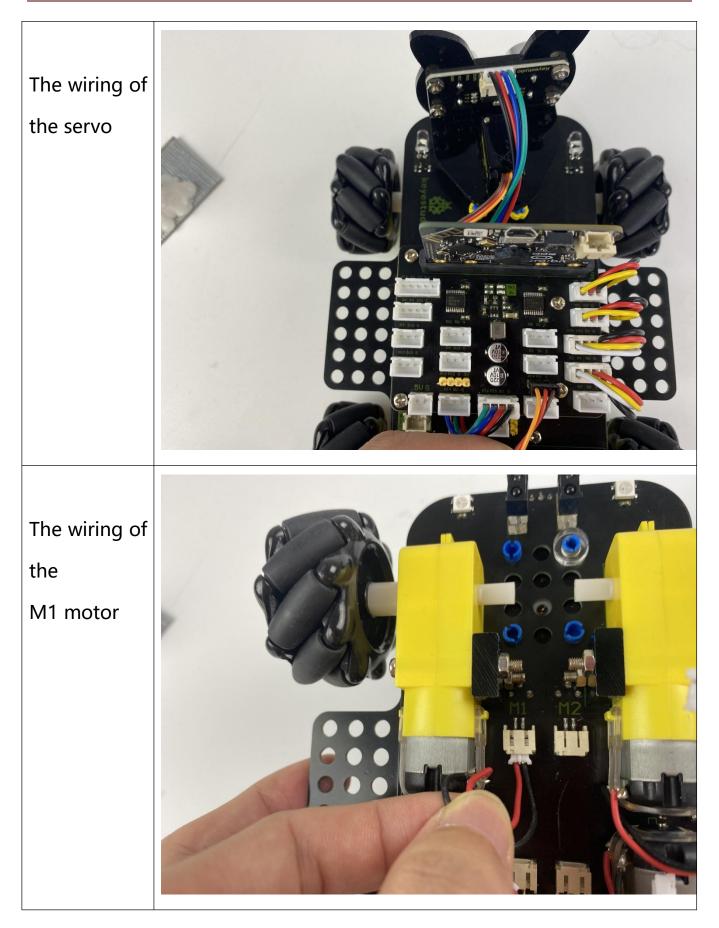




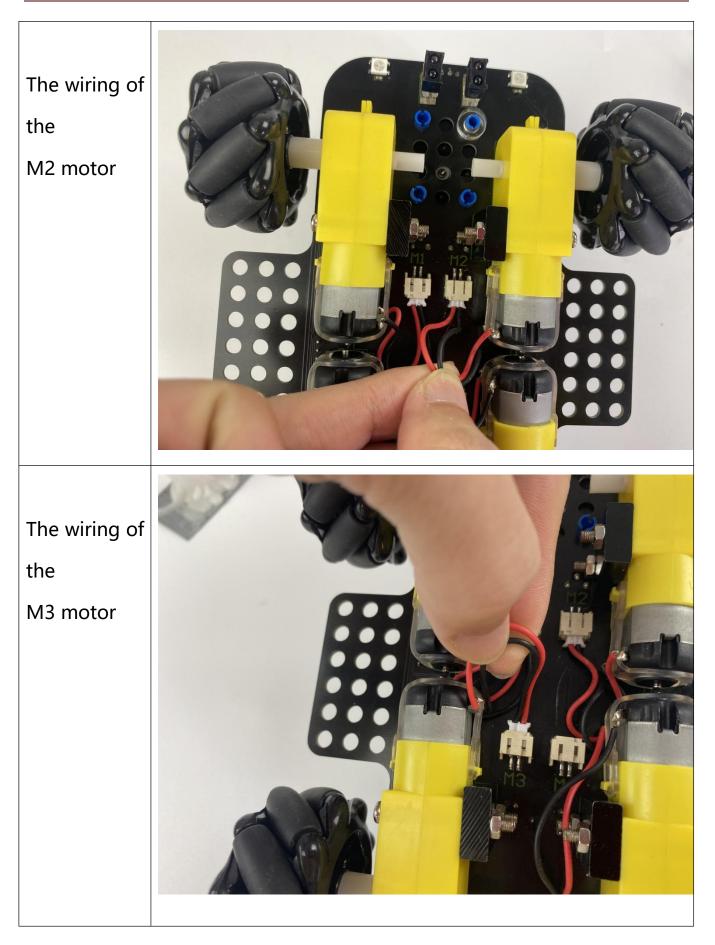




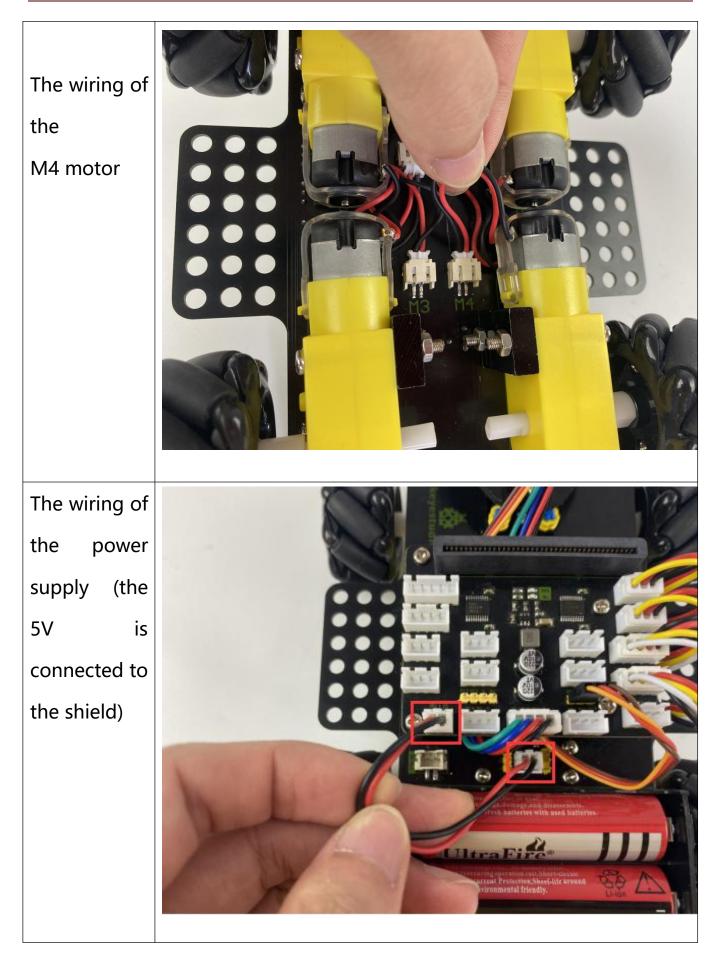














## 7. Python

This tutorial is written for Python language. If you want to use graphical code programming, please refer to the manual "Makecode Tutorial.pdf". In the root directory of the resource you downloaded, there is a folder named "Python tutorial", which stores all the Python code of Micro:bit 4WD Mecanum Robot Car. The Python code file is a file ending with ".py".

### What is MicroPython?

MicroPython is a tiny open source Python programming language interpreter that runs on small embedded development boards. With MicroPython you can write clean and simple Python code to control hardware instead of having to use complex low-level languages like C or C++ (what Arduino uses for programming).

The simplicity of the Python programming language makes MicroPython an excellent choice for beginners who are new to programming and hardware. However MicroPython is also quite full-featured and supports most of Python's syntax so even seasoned Python veterans will find MicroPython familiar and fun to use.



More details please log in official micro:bit website:

https://microbit-micropython.readthedocs.io/en/latest/index.html https://microbit-micropython.readthedocs.io/en/latest/tutorials/introducti on.html

## Python has two types of editors (web version and offline version)

1. Web version: <u>https://python.microbit.org/v/1.1</u>



2. The other one is the offline compiler tool -----Mu

(Download Mu: <a href="https://codewith.mu/en/download">https://codewith.mu/en/download</a>)

Mu

Official Website: <u>https://codewith.mu/</u>

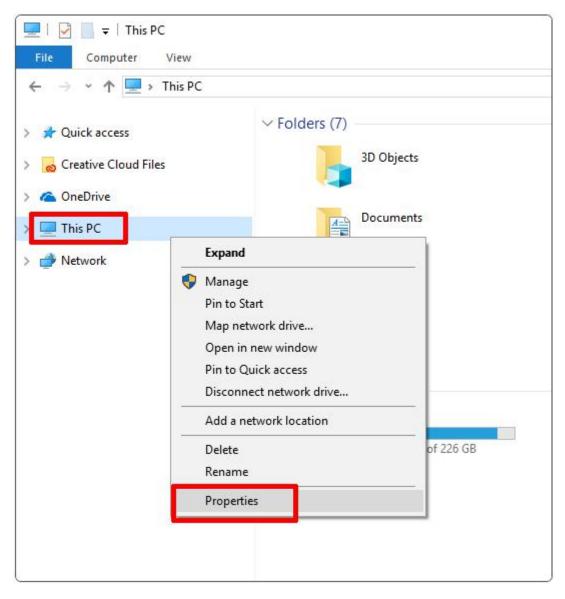
Mu, a Python code editor, is suitable for starters.



Mu doesn' t support 32-bit Windows. The latest version is Mu 1.1.0-beta 2

### 1. Download Mu

Click "This PC" and right- click to select Properties to check the version of your computer.



Below is shown system type of your computer.



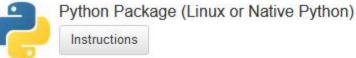
| ontrol Panel Home       | View basic information   | about your computer  |
|-------------------------|--------------------------|--|
| evice Manager           | Windows edition          |  |
| emote settings          | Windows 10 Home          |  |
| ystem protection        | © 2018 Microsoft Corpora | tion. All rights reserved.   |
| dvanced system settings |                          | and a second |
|                         | System                   |  |
|                         | Processor:               | Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz 2.70 GHz  |
|                         | Installed memory (RAM):  | 8.00 GB (7.73 GB usable)   |
|                         | System type:             | 64-bit Op rating System, x64-based processor   |
|                         | Pen and Touch:           | No Pen or Touch Input is available for this Display  |
|                         |                          |  |

# Download Mu

The simplest and easiest way to get Mu is via the official installer for Windows or Mac OSX (we no longer support 32bit Windows)

The current recommended version is Mu 1.1.0-beta-2. We advise people to update to this version via the links for each suppor system:







# Enter link: https://codewith.mu/en/download to download the

# corresponding version of Mu.

| → * ↑ ↓  | > This | PC > Windows10 1909 (C:) > Users > | Administrator > Downloads | 5 V           | , S    |
|--|--------|------------------------------------|---------------------------|---------------|--------|
|  | ^      | Name                               | Date modified             | Туре          | Siz    |
| 🖈 Quick access   |        | T- J (1)                           |                           |               |        |
| E Desktop  | *      | ∨ Today (1)                        |                           |               |        |
| 🕹 Downloads  | 1      | 🛃 Mu-Editor-Win64-1.1.0b2          | 3/24/2021 2:22 PM         | Windows Insta | ller 1 |
| Downloads  | 14     |                                    |                           |               |        |
| Documents  | *      |                                    |                           |               |        |
|  |        |                                    |                           |               |        |
| Documents  | *      |                                    |                           |               |        |
| <ul> <li>Documents</li> <li>Pictures</li> </ul>  | *      |                                    |                           |               |        |
| Documents <ul> <li>Documents</li> <li>Pictures</li> </ul> This PC  | *      |                                    |                           |               |        |
| <ul> <li>Documents</li> <li>Pictures</li> <li>This PC</li> <li>3D Objects</li> </ul>                                     | *      |                                    |                           |               |        |
| <ul> <li>Documents</li> <li>Pictures</li> <li>This PC</li> <li>3D Objects</li> <li>Desktop</li> </ul>                    | *      |                                    |                           |               |        |
| <ul> <li>Documents</li> <li>Pictures</li> <li>This PC</li> <li>3D Objects</li> <li>Desktop</li> <li>Documents</li> </ul> | *      |                                    |                           |               |        |

## Mac OSX: <a href="https://codewith.mu/en/howto/1.1/install\_macos">https://codewith.mu/en/howto/1.1/install\_macos</a>

Windows 10

You will view the page pop up, then click More info



| Windows protected your PC  | ^ |
|--|---|
| Windows Defender SmartScreen prevented an unrecognized app from<br>starting Running this app might put your PC at risk.<br>More info |   |
| Don't run  |   |

Then click "Run anyway" .



| Wind     | dows protected your PC   | × |
|----------|--|---|
|          | s Defender SmartScreen prevented an unrecognized app from<br>Running this app might put your PC at risk. |   |
| App:     | mu_2018-06-19_10_25_master_a132d40_64bit.e<br>xe   |   |
| Publishe | r: Unknown publisher   |   |
|          |  |   |
|          |  |   |
|          |  |   |
|          |  |   |
|          |  |   |
|          |  |   |
|          | Run anyway Don't run   |   |

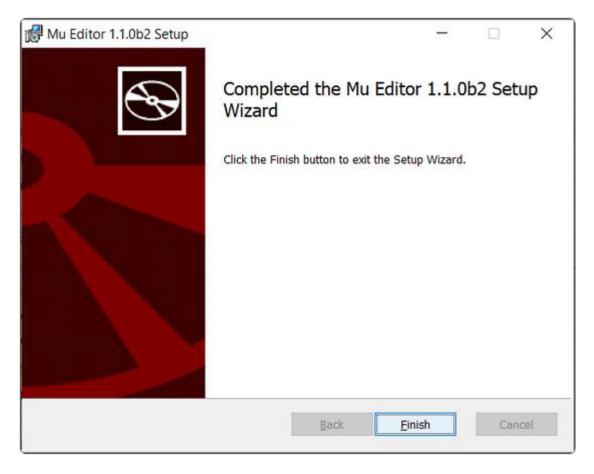
After it is installed, click "finish" .



| Mu Editor 1.1.0b2 Setup       |   |   |            | ×  |
|-------------------------------|---|---|------------|----|
|                               | Please read the Mu Editor<br>Agreement  | 1.1.0b2 Lice                                  | ense       |    |
|                               | GNU GENERA<br>Version 3, 29   |   | ICENSE     | ^  |
|                               | Copyright (C) 2007 Free<br>Foundation, Inc. <http: f<br="">Everyone is permitted t<br/>distribute verbatim copi<br/>of this license documen<br/>not allowed.<br/>Preamble</http:> | sf.org/><br>o copy and<br>es<br>it, but chang | jing it is |    |
|                               | The GNU General Publi   | ic License is                                 | a free,    | ~  |
|                               | I accept the terms in the Licen   |   |            |    |
| Print                         | Back  | nstall  | Cance      | el |
| 🛃 Mu Editor 1.1.0b2 Setup     |   | <u></u>                                       |            | ×  |
| Installing Mu Editor 1.1      | .0b2  |   | e          | Ð  |
| Please wait while the Setup W | izard installs Mu Editor 1.1.0b2.   |   |            |    |
| Status: Copying new file      | s   |   |            | _  |
|                               |   |   |            |    |
|                               |   |   |            |    |
|                               |   |   |            |    |
|                               |   |   |            |    |
|                               | Back  | Next  | Cance      | el |



## After it is installed, click "finish" .



Start Mu

Next, find it according to the following picture



| ≡. | Recently added                   | Productivity                       |
|----|----------------------------------|------------------------------------|
|    | C Mu Editor                      | न्द्र भग भग                        |
|    | #                                |                                    |
|    | 3D Viewer                        | Office S Mail                      |
|    | A                                |                                    |
|    | Alarms & Clock                   | ○                                  |
|    | Audacity                         | Microsoft Edge Photos Microsoft To |
|    | c                                | Explore                            |
|    | Calculator                       | Partiy Sunny                       |
|    | Calendar                         | 9° <sup>12°</sup><br>6° <b>T</b>   |
|    | Camera                           | Microsoft Store London             |
| ۲  | Candy Crush Friends              |                                    |
| ß  | Connect                          | NETFLIX 🌌 🏘 🙀                      |
|    | E                                | Solitaire Play                     |
|    | eLicenser                        |                                    |
| ŝ  | F                                |                                    |
| ¢  | Farm Heroes Saga                 |                                    |
|    | Feedback Hub                     |                                    |
| -  | $\mathcal P$ Type here to search | O Ħ 🔚                              |

Its main interface is shown as follows:



| P Mu 1.1.0.beta. | 1.5 - untitled   |
|------------------|--|
|                  | Image: Save     Imag |
| untitled 🗙       |  |
| 1 # V<br>2       | Write your code here :-)   |
| Changed to BBC   | C micro:bit mode.  |

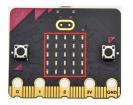
# <mark>8.Projects</mark>

(Note: project 1 to 12 will be conducted with the built-in sensors and LED

dot matrix of the Micro:bit main board V2)



## **Project 1: Heartbeat**



### (1)Project Introduction

This project is easy to conduct with a micro:bit main board, a Micro USB cable and a computer. This experiment serves as a starter for your entry to the magical programming world of Micro:bit.

#### (2) Preparations:

A. Attach the Micro:bit main board to your computer via the USB cable;

B.Open the offline version of Mu.

(3)Test Code:



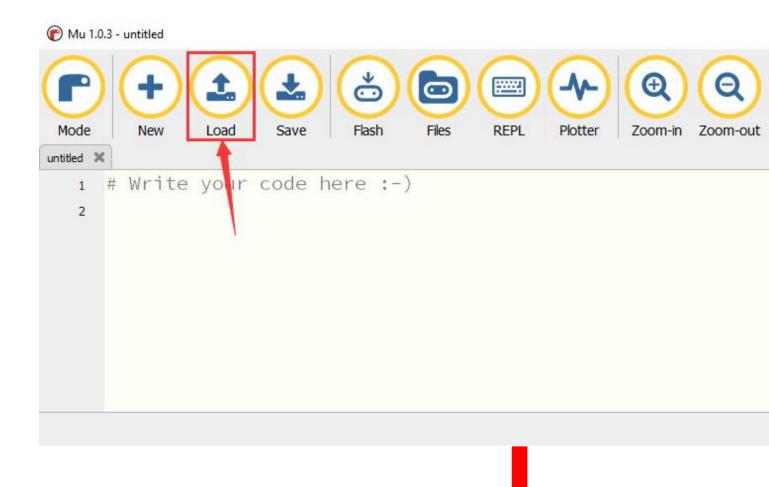
# Open Mu software, click Mode, then click "BBC micro: bit" and

"OK" :

| Mu 1.0.3 - untitled | Save Flash Files REPL Plotter Com-in Zoom-out   |
|---------------------|---|
| # Write your        | Please select the desired mode then click "OK". Otherwise, click "Cancel".  Adafruit CircuitPython Use CircuitPython on Adafruit's line of boards.  BBC micro:bit |
|                     | Write MicroPython for the BBC micro:bit.<br>Pygame Zero<br>Make games with Pygame Zero.<br>Python 3<br>Create code using standard Python 3.                       |
|                     | Change mode at any time by clicking the "Mode" button containing Mu's logo.   |

Tap "Load", select "Project 1: Heartbeat.py" file and click "open":

| File   | Route                     | File Name    |
|--------|---------------------------|--------------|
| Туре   |                           |              |
| Python | KS4031(KS4032)            | Project 1 :  |
| file   | folder/Python             | Heartbeat.py |
|        | Tutorial/Python           |              |
|        | Code/Project 1: Heartbeat |              |





|              | S. Python Tutorial > Python Code |                  |                    |      |
|--------------|----------------------------------|------------------|--------------------|------|
| ganize 🔻 🛛 N | ew folder                        |                  | EEE 🔻              | Π    |
| Desktop      | * ^ Name                         | Date modified    | Туре               | Siz  |
| - Downloads  | 🖈 💽 microbit-Heartbeat           | 9/3/2020 2:14 PM | Python Source File |      |
| Documents    | *                                | 0                |                    |      |
| Pictures     | *                                |                  |                    |      |
| This PC      |                                  |                  |                    |      |
| 3D Objects   |                                  |                  |                    |      |
| Desktop      |                                  |                  | 2                  |      |
| Documents    | v K                              |                  |                    |      |
|              | File name:                       | ~                | *.py               | tool |

There is another way to import code. Open Mu software and drag file" Project1:Heartbeat.py" into it.



| Mu 1.1.0.beta.2 - untitled<br>Mode New Load Save  | Flash Files REFL Plotter    | Q<br>Zoom-in Zoom-ou | t Theme Check | (I)<br>Tidy | -<br>Help      | Quit    | ×             |
|---|-----------------------------|----------------------|---------------|-------------|----------------|---------|---------------|
| untitled 🗙  |                             |                      |               |             |                |         |               |
| i # Write your code   | e here :-)                  |                      |               |             |                |         |               |
|   |                             | + Copy               |               |             |                |         |               |
|   |                             | drag it              | to here       |             |                |         |               |
|   |                             | drag it              |               | BBC mi      | cro:bit        |         | Ø             |
| 8.1: Heartbeat  |                             | drag it              |               | BBC mi      | cro:bit<br>—   |         | <b>0</b><br>× |
|   | thon Tutorial → Python Code |                      |               | BBC mi      | _              |         |               |
|   | thon Tutorial > Python Code | 8.1: Heartbeat       |               |             | _<br>          |         |               |
| ← → ~ ↑ <mark>-</mark> « 5. Py  | Name                        | 8.1: Heartbeat       |               |             | _<br>ر<br>Type | ) Searc | :h 8<br>S     |
| $\leftarrow \rightarrow \checkmark \uparrow \square \ll 5. Py$ $\blacksquare Documents                                    $   | ~                           | 8.1: Heartbeat       | Date modified |             | _<br>ر<br>Type |         | :h 8<br>S     |
| ← → ~ ↑ <mark>····</mark> ≪ 5. Pyr  | Name                        | 8.1: Heartbeat       | Date modified |             | _<br>ر<br>Type | ) Searc | :h 8<br>S     |
| <ul> <li>← → × ↑</li> <li>Que of the second secon</li></ul> | Name<br>microbit-Heartbeat  | 8.1: Heartbeat       | Date modified |             | _<br>ر<br>Type | ) Searc | :h 8<br>S     |

You can also input code in the edit window yourself.

(note:all English words and symbols must be written in English.)



P Mu 1.0.3 - microbit-Heartbeat.py

| • |  |     |
|---|--|-----|
| e | ew Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check Help Quit |     |
|   |  | _   |
| 1 | from microbit import *   |     |
| 2 |  |     |
| 3 | while True:  |     |
| 4 | display.show(Image.HEART)  |     |
| 5 | sleep(500)   |     |
| 6 | display.show(Image.HEART_SMALL)  |     |
| 7 | sleep(500)   |     |
| 8 |  |     |
|   |  |     |
|   |  |     |
|   |  |     |
|   |  |     |
|   |  |     |
|   |  |     |
|   |  | - N |

The following is a list of built-in images:

- Image.HEART
- Image.HEART\_SMALL
- Image.HAPPY
- Image.SMILE
- Image.SAD
- Image.CONFUSED
- Image.ANGRY
- Image.ASLEEP
- Image.SURPRISED

N



- Image.SILLY
- Image.FABULOUS
- Image.MEH
- Image.YES
- Image.NO
- Image.CLOCK12, Image.CLOCK11, Image.CLOCK10, Image.CLOCK9,

Image.CLOCK8, Image.CLOCK7, Image.CLOCK6, Image.CLOCK5,

Image.CLOCK4, Image.CLOCK3, Image.CLOCK2, Image.CLOCK1

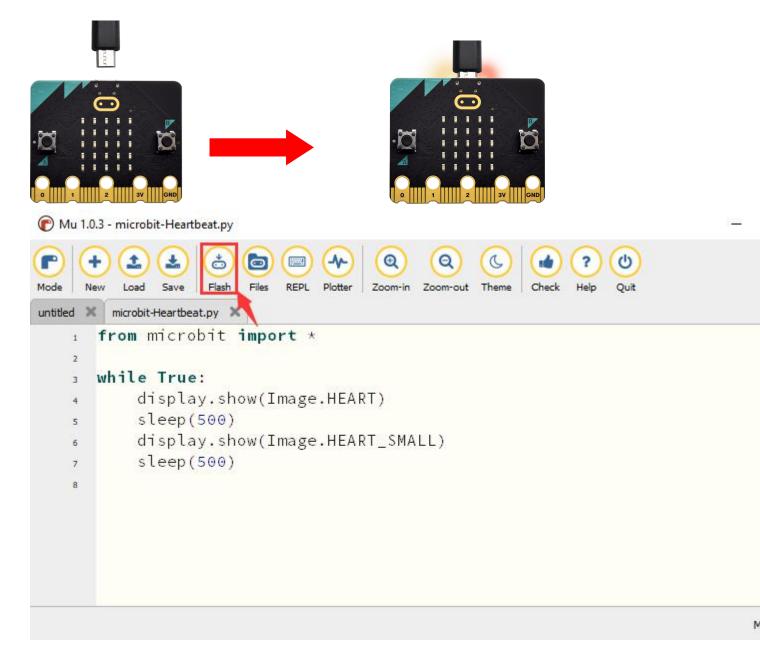
- Image.ARROW\_N, Image.ARROW\_NE, Image.ARROW\_E, Image.ARROW\_SE, Image.ARROW\_S, Image.ARROW\_SW, Image.ARROW\_W, Image.ARROW\_NW
- Image.TRIANGLE
- Image.TRIANGLE\_LEFT
- Image.CHESSBOARD
- Image.DIAMOND
- Image.DIAMOND\_SMALL
- Image.SQUARE
- Image.SQUARE\_SMALL
- Image.RABBIT
- Image.COW
- Image.MUSIC\_CROTCHET
- Image.MUSIC\_QUAVER



- Image.MUSIC\_QUAVERS
- Image.PITCHFORK
- Image.PACMAN
- Image.TARGET
- Image.TSHIRT
- Image.ROLLERSKATE
- Image.DUCK
- Image.HOUSE
- Image.TORTOISE
- Image.BUTTERFLY
- Image.STICKFIGURE
- Image.GHOST
- Image.SWORD
- Image.GIRAFFE
- Image.SKULL
- Image.UMBRELLA
- Image.SNAKE, Image.ALL\_CLOCKS, Image.ALL\_ARROWS

Connect micro:bit board to computer with USB cable, click "Flash" to download code to micro:bit board.





The code, even it is wrong, can be downloaded to micro:bit board successfully, yet not working on micro:bit board.

Click "Flash" to download code to micro:bit.



| Mode (   | .0.3 - microbit-Heartbeat.py  | <u>2008</u> |
|--|---|-------------|
| untitled<br>1<br>2<br>3<br>4<br>5<br>6<br>7<br>8 | <pre>microbitHeartbeat.py from microbit import * while True:     display.show(Image.HEART)     sleep(500)     display.show(Image.HEART_SMALL)     sleeps(500)</pre> |             |
| Copied co  | ode onto micro:bit.   |             |

Click "REPL" and press the reset button on micro:bit, the error information will be displayed on REPL window, as shown below:

N



| Mu 1.0.3 - microbit-Heartbeat.py                        |  |  |  |  |  |
|---|--|--|--|--|--|
| <u> </u>  | +<br>Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check Help Quit<br>microbit-Heartbeat.py  |  |  |  |  |
| 1   | from microbit import *   |  |  |  |  |
| 2   |  |  |  |  |  |
| 3   | while True:  |  |  |  |  |
| 4   | <pre>4 display.show(Image.HEART)</pre>   |  |  |  |  |
| 5   | s sleep(500)   |  |  |  |  |
| 6   | display.show(Image.HEART_SMALL)  |  |  |  |  |
| 7   | 7 sleeps(500)  |  |  |  |  |
| 8   |  |  |  |  |  |
| BBC mic   | ro:bit REPL  |  |  |  |  |
| Type "h<br>>>><br>>>> Tra<br>File<br>NameErr<br>MicroPy | /thon v1.9.2-34-gd64154c73 on 2017-09-01; micro:bit v1.0.1 with nRF51822<br>nelp()" for more information.<br>aceback (most recent call last):<br>"main", line 7, in <module><br/>for: name 'sleeps' is not defined<br/>/thon v1.9.2-34-gd64154c73 on 2017-09-01; micro:bit v1.0.1 with nRF51822<br/>nelp()" for more information.</module> |  |  |  |  |

N

Click "REPL" again to turn off REPL mode, then you could refresh new code.

To make sure code correct, you only need to tap "Check". The errors will be

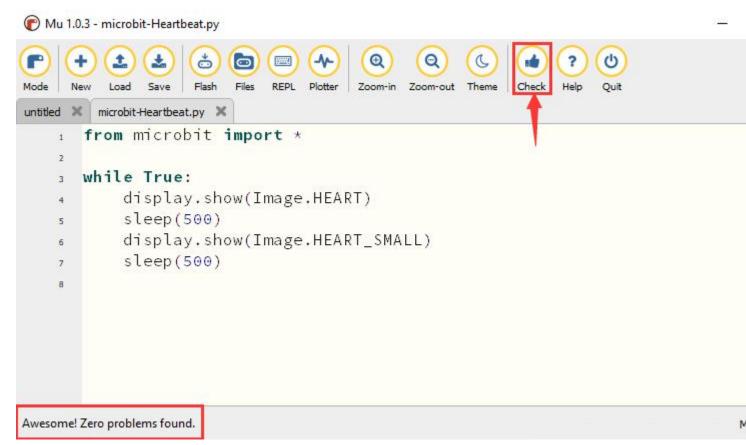
shown on the window.

>>>



Mu 1.0.3 - microbit-Heartbeat.py P + ÷ ð -1-Q Q C £ 0 1999 C C ? Flash Plotter Theme Check Mode New Load Save Files REPL Zoom-in Zoom-out Help Quit microbit-Heartbeat.py 🗙 untitled 🗶 from microbit import \* 1 2 while True: 3 display.show(Image.HEART) 4 sleep(500) 5 display.show(Image.HEART\_SMALL) 6 sleeps(500) 7 ↑ undefined name 'sleeps' 8

Modify the code according to the prompt and click "Check" .



N



More tutorials, log in website please:<u>https://codewith.mu/en/tutorials/</u>

# (4)Test Results:

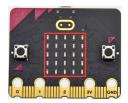
After uploading test code to micro:bit main board, clicking "Flash" again and keeping the connection with the computer to power the main board, the LED dot matrix shows pattern "" and then "" alternatively.

#### (5)Code Explanation:

| from microbit import *          | Import the library file of micro: bit |  |
|---------------------------------|---------------------------------------|--|
| while True:                     | This is a permanent loop that makes   |  |
|                                 | micro:bit execute the code of it.     |  |
| display.show(Image.HEART)       | micro: bit shows "••"                 |  |
| sleep(500)                      | Delay in 500ms                        |  |
| display.show(Image.HEART_SMALL) | micro: bit displays "🔡"               |  |

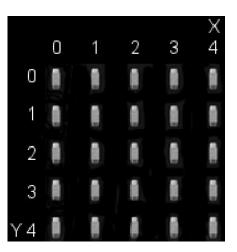


# **Project 2: Light A Single LED**



(1) Project Introduction

The LED dot matrix consists of 25 LEDs arranged in a 5 by 5 square. In order to locate these LEDs quickly, as the figure shown below, we can regarded this matrix as a coordinate system and create two aces by marking those in rows from 0 to 4 from top to bottom, and the ones in columns from 0 to 4 from the left to the right. Therefore, the LED sat in the second of the first line is (1,0) and the LED positioned in the fifth of the fourth column is (3,4) and others likewise.



#### (2) Preparations:

- A. Attach the Micro:bit main board to your computer via the USB cable;
- B.Open the offline version of Mu.



# (3)Test Code:

Enter Mu software and open the file "Project 2: Light A Single LED.py" to

import code:

| Туре   | Route                    | File Name                 |
|--------|--------------------------|---------------------------|
| Python | KS4031(KS4032)           | Project 2: Light A Single |
| file   | folder/Python            | LED.py                    |
|        | Tutorial/Python          |                           |
|        | Code/Project 2 : Light A |                           |
|        | Single LED               |                           |

You can also input code in the editing window yourself.

(Note:all English words and symbols must be written in English)



| 🕜 Mu 1.     | .0.3 - microbit-Light up an LED.py  | 13 <u>1111</u> |
|-------------|---|----------------|
| Mode (      | Image: Save       Image: Save |                |
| microbit-Li | ight up an LED.py 🔀   |                |
| 1           | from microbit import *  |                |
| 2           |   |                |
| 3           | val1 = Image("09000;""00000;""00000;""00000;""00000;")  |                |
| 4           | val2 = Image("00000:""00000:""00000:""00000:""00090:")  |                |
| 5           | val3 = Image("00000:""00000:""00000:""00000:""00000:")  |                |
| 6           |   |                |
| 7           | while True:   |                |
| 8           | display.show(val1)  |                |
| 9           | sleep(500)  |                |
| 10          | display.show(val3)  |                |
| 11          | sleep(500)  |                |
| 12          | display.show(val2)  |                |
| 13          | sleep(500)  |                |
| 14          | display.show(val3)  |                |
| 15          | sleep(500)  |                |
| 16          |   |                |
|             |   |                |

Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.

N



P Mu 1.0.3 - microbit-Light up an LED.py

|              | +<br>Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check Help Quit |  |
|--------------|--|--|
| microbit-Lig | ght up an LED.py 🗙   |  |
| 1            | from microbit import *   |  |
| 2            |  |  |
| 3            | val1 = Image("09000:""00000:""00000:""00000:""00000:")                         |  |
| 4            | val2 = Image("00000:""00000:""00000:""00000:""00090:")                         |  |
| 5            | val3 = Image("00000:""00000:""00000:""00000:""00000:")                         |  |
| 6            |  |  |
| 7            | while True:  |  |
| 8            | display.show(val1)   |  |
| 9            | sleep(500)   |  |
| 10           | display.show(val3)   |  |
| 11           | sleep(500)   |  |
| 12           | display.show(val2)   |  |
| 13           | sleep(500)   |  |
| 14           | display.show(val3)   |  |
| 15           | sleep(500)   |  |
| 16           |  |  |
|              |  |  |

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.



Mu 1.0.3 - microbit-Light up an LED.py

| Mode M | +<br>Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check Help Quit |  |
|--------|--|--|
|        | pht up an LED.py 🗶   |  |
| 1      | from microbit import *   |  |
| 2      |  |  |
| 3      | val1 = Image("09000:""00000:""00000:""00000:""00000:")                         |  |
| 4      | val2 = Image("00000:""00000:""00000:""00000:""00000:")                         |  |
| 5      | val3 = Image("00000:""00000:""00000:""00000:""00000:")                         |  |
| 6      |  |  |
| 7      | while True:  |  |
| 8      | display.show(val1)   |  |
| 9      | sleep(500)   |  |
| 10     | display.show(val3)   |  |
| 11     | sleep(500)   |  |
| 12     | display.show(val2)   |  |
| 13     | sleep(500)   |  |
| 14     | display.show(val3)   |  |
| 15     | sleep(500)   |  |
| 16     |  |  |

## (4)Test Results:

After uploading test code to micro:bit main board and powering the main

board via the USB cable, the LED in (1,0) lights up for 0.5s and the one in

(3,4) shines for 0.5s and repeat this sequence.

#### (5)Code Explanation:

| from microbit | import * | Import the library file |
|---------------|----------|-------------------------|
|               |          | of micro: bit           |



| val1 =  | Set Image() to val1  |
|---|--|
| Image("09000:""00000:""00000:""00000:""00000:") | Set pixel of LED on  |
|   | micro:bit to the value   |
|   | in 0~9   |
|   | Pixel of each LED  |
|   | on micro:bit can be set  |
| val2 =  | in one of ten values   |
| Image("00000:""00000:""00000:""00000:""00090:") | If set pixel to 0  |
| val3 =  | (zero) , which means   |
| Image("00000:""00000:""00000:""00000:""00000:") | in close state, literally,   |
|   | 0 is brightness, 9 is  |
|   | best brightness  |
|   |  |
|   | Set Image() to val2  |
|   | Set Image() to val2<br>Set Image() to val3   |
| while True:                                     |  |
| while True:                                     | Set Image() to val3  |
| while True:                                     | Set Image() to val3<br>This is a   |
| while True:                                     | Set Image() to val3<br>This is a<br>permanent loop that  |
| while True:<br>display.show(val1)               | Set Image() to val3<br>This is a<br>permanent loop that<br>makes micro:bit   |
|   | Set Image() to val3<br>This is a<br>permanent loop that<br>makes micro:bit   |
| display.show(val1)                              | Set Image() to val3<br>This is a<br>permanent loop that<br>makes micro:bit<br>execute the code of it.                        |
| display.show(val1)<br>sleep(500)                | Set Image() to val3<br>This is a<br>permanent loop that<br>makes micro:bit<br>execute the code of it.<br>LED at (1,0) blinks |



| display.show(val2) |                      |
|--------------------|----------------------|
| sleep(500)         | LED at (3,4) flashes |
| display.show(val3) | for 0.5s             |
| sleep(500)         |                      |

## (6)Reference

sleep(ms) : delay time

For more details about delay, please refer to:

https://microbit-micropython.readthedocs.io/en/latest/utime.html

#### **Project 3: LED Dot Matrix**



#### (1) **Project Introduction**

Dot matrices are very commonplace in daily life. They have found wide applications in LED advertisement screens, elevator floor display, bus stop announcement and so on.

The LED dot matrix of Micro: Bit main board contains 25 LEDs in a grid. Previously, we have succeeded in controlling a certain LED to light by integrating its position value into the test code. Supported by the same theory, we can turn on many LEDs at the same time to showcase patterns, digits and characters.

What's more, we can also click" show icon "to choose the pattern we like to display. Last but not the least, we can design patterns by ourselves as well.

# (2) Preparations:

A. Attach the Micro:bit main board to your computer via the USB cable;

B. Open the offline version of Mu.

# (3)Test Code:

#### Code1:

You could open "Project 3: LED Dot Matrix-1.hex "file to Import code (How

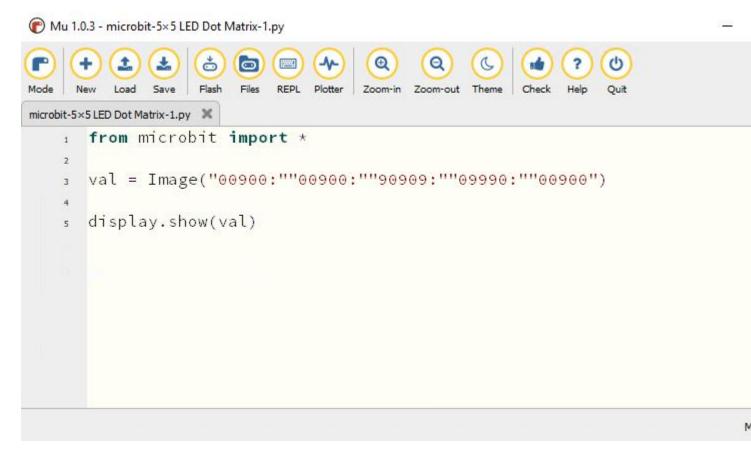
to load the project code?)

| File   | Route                   | File Name           |
|--------|-------------------------|---------------------|
| Туре   |                         |                     |
| Python | KS4031(KS4032)          | Project 3 : LED Dot |
| file   | folder/Python           | Matrix-1.hex        |
|        | Tutorial/Python         |                     |
|        | Code/Project 3: LED Dot |                     |
|        | Matrix-1.hex            |                     |



You can also input code in the editing window yourself.

## (note:all words and symbols must be written in English.)



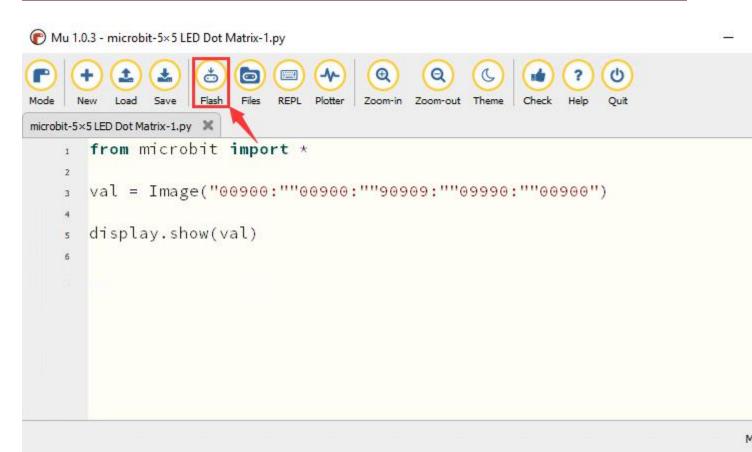
Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.



| Mu 1.0.3 - microbit-5×5 LED Dot Matrix-1.py*   —  |  |  |  |
|---|--|--|--|
| Image: Mode       Image: Mode <thimage: mode<="" th=""> <thimage: mode<="" th=""></thimage:></thimage:> |  |  |  |
| microbit-5×5 LED Dot Matrix-1.py * 🗙  |  |  |  |
| <pre>from microbit import * val = Image("00900:""00900:""00900:""00900") display.show(val) </pre>   |  |  |  |
|   |  |  |  |

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.





# Code2:

You could open "Project 3: LED Dot Matrix-2.hex "file to Import code (How

#### to load the project code?)

| File   | Route                   | File Name           |
|--------|-------------------------|---------------------|
| Туре   |                         |                     |
| Python | KS4031(KS4032)          | Project 3 : LED Dot |
| file   | folder/Python           | Matrix-2.hex        |
|        | Tutorial/Python         |                     |
|        | Code/Project 3: LED Dot |                     |
|        | Matrix-2.hex            |                     |



You can also input code in the editing window yourself.

# (note:all words and symbols must be written in English.)

P Mu 1.0.3 - microbit-5×5 LED Dot Matrix-2.py

| Mode       | New Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check |  |  |
|------------|---|--|--|
| microbit-5 | ×5 LED Dot Matrix-2.py 🗶  |  |  |
| 1          | from microbit import *  |  |  |
| 2          | val = Image("00900:""00900:""90909:""09990:""00900")                |  |  |
| 3          | display.show('1')   |  |  |
| 4          | sleep(500)  |  |  |
| 5          | display.show('2')   |  |  |
| 6          | sleep(500)  |  |  |
| 7          | display.show('3')   |  |  |
| 8          | sleep(500)  |  |  |
| 9          | display.show('4')   |  |  |
| 10         | sleep(500)  |  |  |
| 11         | display.show('5')   |  |  |
| 12         | sleep(500)  |  |  |
| 13         | display.show(val)   |  |  |
| 14         | sleep(500)  |  |  |
| 15         | display.scroll("hello!")  |  |  |
| 16         | sleep(200)  |  |  |
| 17         | display.show(Image.HEART)   |  |  |
| 18         | sleep(500)  |  |  |
| 19         | display.show(Image.ARROW_NE)  |  |  |
| 20         | sleep(500)  |  |  |
| 21         | display.show(Image.ARROW_SE)  |  |  |
| 22         | sleep(500)  |  |  |
| 23         | display.show(Image.ARROW_SW)  |  |  |
| 24         | sleep(500)  |  |  |
| 25         | display.show(Image.ARROW_NW)  |  |  |
| 26         | sleep(500)  |  |  |
| 27         | display.clear()   |  |  |
| 28         |   |  |  |



Click "Check" to examine error in the code. The program proves wrong if

underlines and cursors are shown.

| 🕐 Mu 1     | .0.3 - microbit-5×5 LED Dot Matrix-2.py —  |
|------------|--|
| Mode       | Image: New Load     Save     Files     Files     REPL     Plotter     Zoom-in     Zoom-out     Theme |
| microbit-5 | i×5 LED Dot Matrix-2.py 🕱  |
| 1          | from microbit import *   |
| 2          | val = Image("00900:""00900:""90909:""09990:""00900")   |
| 3          | display.show('1')  |
| 4          | sleep(500)   |
| 5          | display.show('2')  |
| 6          | sleep(500)   |
| 7          | display.show('3')  |
| 8          | sleep(500)   |
| 9          | display.show('4')  |
| 10         | sleep(500)   |
| 11         | display.show('5')  |
| 12         | sleep(500)   |
| 13         | display.show(val)  |
| 14         | sleep(500)   |
| 15         | display.scroll("hello!")   |
| 16         | sleep(200)   |
| 17         | display.show(Image.HEART)  |
| 18         | sleep(500)   |
| 19         | display.show(Image.ARROW_NE)   |
| 20         | sleep(500)   |
| 21         | display.show(Image.ARROW_SE)   |
| 22         | sleep(500)   |
| 23         | display.show(Image.ARROW_SW)   |
| 24         | sleep(500)   |
| 25         | display.show(Image.ARROW_NW)   |
| 26         | sleep(500)   |
| 27         | display.clear()  |
| 28         |  |



If the code is correct, connect micro:bit to computer and click "Flash" to

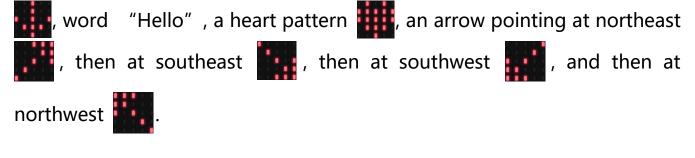
download code to micro:bit board.

| P Mu 1.0.3 - microbit-5×5 LED Dot Matrix-2.py |   |  |  |
|---|---|--|--|
|   |   |  |  |
| Mode  | New Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check |  |  |
| microbit-5                                    | <5 LED Dot Matrix-2.py 🗙  |  |  |
| 1   | from microbit import *  |  |  |
| 2   | val = Image("00900:""00900:""90909:""09990:""00900")                |  |  |
| 3   | display.show('1')   |  |  |
| 4   | sleep(500)  |  |  |
| 5   | display.show('2')   |  |  |
| 6   | sleep(500)  |  |  |
| 7   | display.show('3')   |  |  |
| 8   | sleep(500)  |  |  |
| 9   | display.show('4')   |  |  |
| 10  | sleep(500)  |  |  |
| 11  | display.show('5')   |  |  |
| 12  | sleep(500)  |  |  |
| 13  | display.show(val)   |  |  |
| 14  | sleep(500)  |  |  |
| 15  | display.scroll("hello!")  |  |  |
| 16  | sleep(200)  |  |  |
| 17  | display.show(Image.HEART)   |  |  |
| 18  | sleep(500)  |  |  |
| 19  | display.show(Image.ARROW_NE)  |  |  |
| 20  | sleep(500)  |  |  |
| 21  | display.show(Image.ARROW_SE)  |  |  |
| 22  | sleep(500)  |  |  |
| 23  | display.show(Image.ARROW_SW)  |  |  |
| 24  | sleep(500)  |  |  |
| 25  | display.show(Image.ARROW_NW)  |  |  |
| 26  | sleep(500)  |  |  |
| 27  | display.clear()   |  |  |
| 28  |   |  |  |

(4)Test Results:



After uploading test code to micro:bit main board and powering the main board via the USB cable, we find that the 5\*5 dot matrix start to show numbers 1,2,3,4 and 5, and then it alternatively shows a downward arrow



## (5)Code Explanation:

| from microbit import *                     | Import the library file of        |
|--|-----------------------------------|
|  | micro: bit                        |
| val =                                      |                                   |
| Image("09000:""00000:""00000:""00000:""000 | Set Image() to variable val       |
| 00:")                                      |                                   |
| display.show(val)                          | micro:bit shows " $\rightarrow$ " |
| display.show('1')                          | micro:bit shows "1"               |
| sleep(500)                                 | Delay in 500ms                    |
| display.scroll("hello!")                   | micro:bit scrolls to show         |
|  | "hello!"                          |
| display.show(Image.HEART)                  | micro:bit displays "••"           |
|  | pattern                           |



| display.show(Image.ARROW_NE) | micro:bit shows       |
|------------------------------|-----------------------|
| display.show(Image.ARROW_SE) | "Northeast" arrow     |
| display.show(Image.ARROW_SW) | micro:bit displays    |
| display.show(Image.ARROW_NW) | "Southeast" arrow     |
|                              | micro:bit shows       |
|                              | "Southwest" arrow     |
|                              | micro:bit displays    |
|                              | "Northwest" arrow     |
| display.clear()              | The LED dot matrix of |
|                              | micro:bit clears      |

#### (6) Reference:

display.scroll() :

The display scrolls to show the values, if it is integer or float, we use str ()

to transfer into character strings.

More details, please refer to

https://microbit-micropython.readthedocs.io/en/latest/utime.html



#### **Project 4: Programmable Buttons**



#### (1) Project Introduction

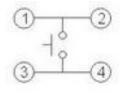
Buttons can be used to control circuits. In an integrated circuit with a push button, the circuit is connected when pressing the button and it is open the other way around.

Both ends of button river in between.



are like two mountains. There is a

The internal metal piece connect the two sides to let the current pass, just like building a bridge to connect two mountains.



Micro: Bit main board boasts three push buttons, two are programmable buttons(marked with A and B), and the one on the other side is a reset button. By pressing the two programmable buttons can input three different signals. We can press button A or B alone or press them together and the LED dot matrix shows A,B and AB respectively. Let's get started.



# (2) Preparations:

- A. Attach the Micro:bit main board to your computer via the USB cable;
- B. Open the offline version of Mu.

## (3)Test Code1:

Enter Mu software and open the file "Project 4: Code-1.py" to import code:

#### (How to load the project code?)

| File Type | Route                | File Name            |
|-----------|----------------------|----------------------|
| Python    | KS4031(KS4032)       | Project 4: Code-1.py |
| file      | folder/Python        |                      |
|           | Tutorial/Python      |                      |
|           | Code/Project 4 :     |                      |
|           | Programmable Buttons |                      |

You can also input code in the editing window yourself.

(note:all words and symbols must be written in English)



| oit-Pro | ogrammable Buttons-1.py 🔀  |
|---------|--|
| 1       | from microbit import *   |
| 2       |  |
| 3       | while True:  |
| 4       | <pre>if button_a.is_pressed():</pre>   |
| 5       | display.show("A")  |
| 6       | <pre>elif button_a.is_pressed() and button_b.is_pressed():</pre>   |
| 7       | display.scroll("AB")   |
| 8       | <pre>elif button_b.is_pressed():</pre>   |
| 9       | display.show("B")  |
| 10      | : State of the sta |
|         |  |

Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.



Mu 1.0.3 - microbit-Programmable Buttons-1.py P ÷ ð Q Q C t -C ? Flash Zoom-in Zoom-out Theme Check Help Mode New Load Save Files REPL Plotter Quit microbit-Programmable Buttons-1.py 💥 from microbit import \* 1 2 while True: 3 if button\_a.is\_pressed(): 4 display.show("A") 5 elif button\_a.is\_pressed() and button\_b.is\_pressed(): 6 display.scroll("AB") 7 elif button\_b.is\_pressed(): 8 display.show("B") 9 10

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.



Mu 1.0.3 - microbit-Programmable Buttons-1.py ð P ÷ 0 -1-Q Q ? ഗ Distance. Help Mode New Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check Quit microbit-Programmable Buttons-1.py 💥 from microbit import \* 1 2 while True: 3 if button\_a.is\_pressed(): 4 display.show("A") 5 elif button\_a.is\_pressed() and button\_b.is\_pressed(): 6 display.scroll("AB") 7 elif button\_b.is\_pressed(): 8 display.show("B") g 10

#### (4)Test Results1:

After uploading test code to micro:bit main board and powering the main board via the USB cable, the 5\*5 LED dot matrix shows "A" if button A is pressed and then released, "B" if button B pressed and released, and "AB" if button A and B pressed together and then released.

#### (5)Test Code2:

Enter Mu software and open the file "Project 4: Code-2.py

" to import code: (<u>How to load the project code</u>?)

| File Type | Route                 | File Name            |
|-----------|-----------------------|----------------------|
| Python    | KS4031(KS4032)/Python | Project 4: Code-2.py |



| file | Tutorial/Python      |  |
|------|----------------------|--|
|      | Code/Project 4:      |  |
|      | Programmable Buttons |  |

You can also input code in the editing window yourself.

(note:all English words and symbols must be written in English)



```
P Mu 1.0.3 - microbit- Programmable Buttons-2.py
                          ð
                                                   Ð
                                                                  C
                               \odot
                                                           Θ
              T
                                     :::::::
             Load
                         Flash
                                                 Zoom-in Zoom-out Theme
Mode
       New
                   Save
                                Files
                                     REPL
                                           Plotter
                                                                        Check
microbit- Programmable Buttons-2.py 💥
      from microbit import *
   4
      a = 0
   2
      b = 0
   3
      val1 = Image("00000:""00000:""00000:""00000:""00900")
   4
     val2 = Image("00000:""00000:""00000:""00900:""99999")
   5
     val3 = Image("00000:""00000:""00900:""99999:""99999")
   6
      val4 = Image("00000:""00900:""99999:""99999:""99999")
   7
      val5 = Image("00900:""99999:""99999:""99999:""99999")
   8
      9
      display.show(val1)
   10
   11
      while True:
   12
          while button_a.is_pressed() == True:
   13
               sleep(10)
   14
               if button_a.is_pressed() == False:
   15
                   a = a + 1
   16
                   if(a >= 5):
   17
                       a = 5
   18
                   break
   19
          while button_b.is_pressed() == True:
   20
               sleep(10)
   21
               if button_b.is_pressed() == False:
   22
                   a = a - 1
   23
                   if(a <= 0):
   24
                       a = 0
   25
                   break
   26
```



| 27 | if a == 0:         |
|----|--------------------|
| 28 | display.show(val1) |
| 29 | if a == 1:         |
| 30 | display.show(val2) |
| 31 | if a == 2:         |
| 32 | display.show(val3) |
| 33 | if a == 3:         |
| 34 | display.show(val4) |
| 35 | if a == 4:         |
| 36 | display.show(val5) |
| 37 | if a == 5:         |
| 38 | display.show(val6) |
| 39 |                    |

Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.



```
Mu 1.0.3 - microbit- Programmable Buttons-2.py
                          ð
                               \odot
                                                   Ð
                                                           Θ
                                                                  a
                                     ......
             Load
                         Flash
                                     REPL
                                                 Zoom-in Zoom-out Theme
Mode
       New
                   Save
                                Files
                                           Plotter
                                                                        Check
microbit- Programmable Buttons-2.py 💥
      from microbit import *
   a.
      a = 0
   2
      b = 0
   3
      val1 = Image("00000:""00000:""00000:""00000:""00900")
   4
      val2 = Image("00000:""00000:""00000:""00900:""99999")
   5
     val3 = Image("00000:""00000:""00900:""99999:""99999")
   6
      val4 = Image("00000:""00900:""99999:""99999:""99999")
   7
      val5 = Image("00900:""99999:""99999:""99999:""99999")
   8
      9
      display.show(val1)
   10
   11
      while True:
   12
          while button_a.is_pressed() == True:
   13
               sleep(10)
   14
               if button_a.is_pressed() == False:
   15
                   a = a + 1
   16
                   if(a >= 5):
   17
                       a = 5
   18
                   break
   19
          while button_b.is_pressed() == True:
   20
               sleep(10)
   21
               if button_b.is_pressed() == False:
   22
                   a = a - 1
   23
                   if(a <= 0):
   24
                       a = 0
   25
                   break
   26
```



| 27 | if a == 0:         |
|----|--------------------|
| 28 | display.show(val1) |
| 29 | if a == 1:         |
| 30 | display.show(val2) |
| 31 | if a == 2:         |
| 32 | display.show(val3) |
| 33 | if a == 3:         |
| 34 | display.show(val4) |
| 35 | if a == 4:         |
| 36 | display.show(val5) |
| 37 | if a == 5:         |
| 38 | display.show(val6) |
| 39 |                    |

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.



```
Mu 1.0.3 - microbit- Programmable Buttons-2.py
                          ð
                                                   Ð
                   +
                                \odot
                                                           Θ
                                      :::::::
                                                                   a
             Load
                   Save
                         Flash
                                Files
                                     REPL
                                                 Zoom-in Zoom-out Theme
Mode
       New
                                           Plotter
                                                                        Check
microbit- Programmable Buttons-2.py 💥
      from microbit import *
   a.
      a = 0
   2
      b = 0
   3
      val1 = Image("00000:""00000:""00000:""00000:""00900")
   4
      val2 = Image("00000:""00000:""00000:""00900:""99999")
   5
      val3 = Image("00000:""00000:""00900:""99999:""99999")
   6
      val4 = Image("00000:""00900:""99999:""99999:""99999")
   7
      val5 = Image("00900:""999999:""99999:""999999:""999999")
   8
      9
      display.show(val1)
   10
   11
      while True:
   12
          while button_a.is_pressed() == True:
   13
               sleep(10)
   14
               if button_a.is_pressed() == False:
   15
                   a = a + 1
   16
                   if(a >= 5):
   17
                       a = 5
   18
                   break
   19
          while button_b.is_pressed() == True:
   20
               sleep(10)
   21
               if button_b.is_pressed() == False:
   22
                   a = a - 1
   23
                   if(a <= 0):
   24
                       a = 0
   25
                   break
   26
```



| 27 | if a == 0:            |
|----|-----------------------|
| 28 | display.show(val1)    |
| 29 | if a == 1:            |
| 30 | display.show(val2)    |
| 31 | if a == 2:            |
| 32 | display.show(val3)    |
| 33 | if a == 3:            |
| 34 | display.show(val4)    |
| 35 | if a == 4:            |
| 36 | display.show(val5)    |
| 37 | if a == 5:            |
| 38 | display.show(val6)    |
| 39 | 57 37 551 Corte (##2) |

## (6)Test Results2:

After uploading test code to micro:bit main board and powering the main board via the USB cable, when the button A is pressed, the LEDs turning red increase while when the button B pressed, the LEDs turning red reduce.

#### (7)Code Explanation:

| from microbit import *    | Import the library file of micro: bit |
|---------------------------|---------------------------------------|
| while True:               | This is a permanent loop that         |
|                           | makes micro:bit execute the code      |
|                           | of it.                                |
| if button_a.is_pressed(): | If button A is pressed                |
| display.show("A")         | micro:bit shows "A"                   |



| elif button_a.is_pressed() and       | If button A and B are pressed at |
|--------------------------------------|----------------------------------|
| button_b.is_pressed():               | same time                        |
| display.scroll("AB")                 | micro:bit displays "AB"          |
| elif button_b.is_pressed():          | If button B is pressed           |
| display.show("B")                    | micro:bit shows "B"              |
| while button_a.is_pressed() == True: | When the button A is pressed     |
| sleep(10)                            | Delay in 10ms to eliminate the   |
| if button_a.is_pressed() == False:   | shaking of button A              |
| a = a + 1                            | when button A is released,       |
| if(a >= 5):                          | Variable a adds 1                |
| a = 5                                | If variable a≥5                  |
| break                                | Variable a=5                     |
| while button_b.is_pressed() == True: | exit the loop                    |
| sleep(10)                            | when button B is pressed         |
| if button_b.is_pressed() == False:   | Delay in 10ms to eliminate the   |
| a = a - 1                            | shaking of button B              |
| if(a <= 0):                          | When the button B is released    |
| a = 0                                | Variable a reduces 1 gradually   |
| break                                | When a≤0                         |
| if a == 0:                           | Variable a=0                     |
| display.show(val1)                   | exit the loop                    |
| if a == 1:                           | When a=0                         |



| display.show(val2) | micro:bit shows pattern val1    |
|--------------------|---------------------------------|
| if a == 2:         | When a=1                        |
| display.show(val3) | micro:bit displays pattern val2 |
| if a == 3:         | When a=2                        |
| display.show(val4) | micro:bit shows pattern val3    |
| if a == 4:         | If a=3                          |
| display.show(val5) | micro:bit displays pattern val4 |
| if a == 5:         | If a=4                          |
| display.show(val6) | micro:bit shows pattern val5    |
|                    | If a=5                          |
|                    | micro:bit displays pattern val6 |

#### **Project 5: Temperature Detection**

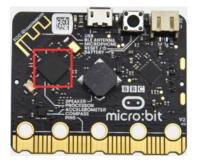
## (1) Project Introduction

The Micro:bit main board is not equipped with a temperature sensor, but uses the temperature sensor built into NFR52833 chip for temperature detection. Therefore, the detected temperature is more closer to the temperature of the chip, and there maybe deviation from the ambient temperature.



In this project, we use the sensor to test the temperature in the current environment, and display the test results in the display data (device). And then control the LED dot matrix to display different patterns by setting the temperature range detected by the sensor.

Note: the temperature sensor of Micro:bit main board is shown below:



## (2) Preparations:

- A. Attach the Micro:bit main board to your computer via the USB cable;
- B. Open the offline version of Mu.

# (3)Test Code1:

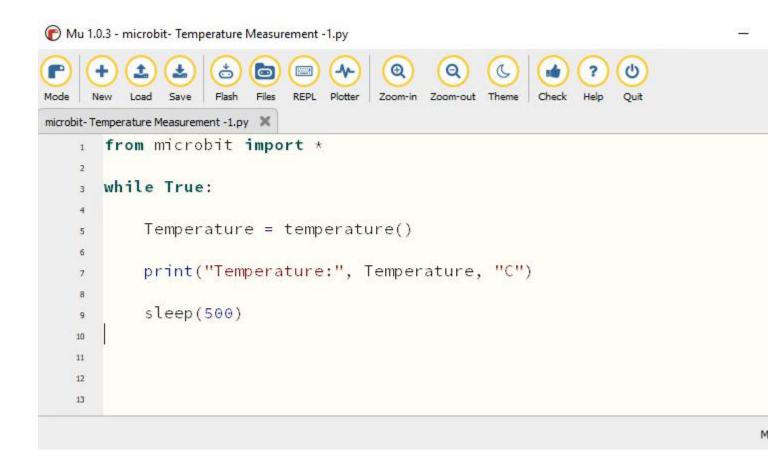
Enter Mu software and open the file "Project 5: Code-1.py" to import code:

| File Type   | Route          | File Name            |
|-------------|----------------|----------------------|
| Python file | KS4031(KS4032) | Project 5: Code-1.py |
|             | folder/Python  |                      |



| Tutorial/Python |      |     |   |
|-----------------|------|-----|---|
| Code/Project    | 5    |     | • |
| Temperature De  | tect | ior | ۱ |

You can also input code in the editing window yourself.(note:all English words and symbols must be written in English)



Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.



| 🕜 Mu 1      | 1.0.3 - microbit- Temperature Measurement -1.py    | 3 <u>958</u> |
|-------------|--|--------------|
| Mode (      | + 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2            |              |
| microbit- 1 | Temperature Measurement -1.py 🔀                    |              |
| 1           | from microbit import *                             |              |
| 2           |  |              |
| з           | while True:  |              |
| 4           |  |              |
| 5           | Temperature = temperature()                        |              |
| 6           |  |              |
| 7           | <pre>print("Temperature:", Temperature, "C")</pre> |              |
| 8           |  |              |
| 9           | sleep(500)   |              |
| 10          |  |              |
| 11          |  |              |
| 12          |  |              |
| 13          |  |              |
|             |  |              |

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.



| Mu 1.0.3 - microbit- Temperature Measurement -1.py |   |  |  |  |
|--|---|--|--|--|
| 6  | +<br>Load Save Elash Files REPL Plotter Zoom-in Zoom-out Theme Check Help Quit<br>Temperature Measurement -1.py |  |  |  |
|  | from microbit import *  |  |  |  |
| 1  | Trom interoble impore *   |  |  |  |
| 2  | while True:   |  |  |  |
|  | white frue:   |  |  |  |
| 4  | - · · · · · · · · · · · · · · · · · · ·   |  |  |  |
| 5  | Temperature = temperature()   |  |  |  |
| 6  |   |  |  |  |
| 7  | <pre>print("Temperature:", Temperature, "C")</pre>  |  |  |  |
| 8  |   |  |  |  |
| 9  | sleep(500)  |  |  |  |
| 10   |   |  |  |  |
| 11   |   |  |  |  |
| 12   |   |  |  |  |
| 13   |   |  |  |  |
|  |   |  |  |  |

#### (4)Test Results1:

After downloading test code 1 to micro:bit board, keep USB connected and click "REPL" and press the reset button on micro:bit. Then REPL window will show the ambient temperature value, as shown below:( C stands for temperature unit)

| <b>č</b>           |
|--------------------|
| www.keyestudio.com |

| Mode Ne     | ev Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check Help Quit |
|-------------|--|
| microbit-Te | mperature Measurement -1.py 🔀  |
| 1           | from microbit import * 🔪   |
| 2           |  |
| 3           | while True:  |
| 4           |  |
| 5           | Temperature = temperature()  |
|             | remperature - cemperature()  |
| 6           |  |
| 7           | <pre>print("Temperature:", Temperature, "C")</pre>                           |
| 8           |  |
| 9           | sleep(500)   |
| 10          |  |
| BBC micro   | bit REPL   |
| Temperat    | ure: 27 C  |
| Temperat    | ure: 27 C  |
| Temperat    | ure: 27 C  |
| Temperat    | cure: 27 C   |
|             | ture: 28 C   |
|             | ture: 28 C   |
|             | ure: 28 C  |
|             | ure: 28 C  |
|             | ure: 28 C  |
|             | ure: 28 C  |
| Temperat    | ure: 28 C<br>ure: 28 C   |

# (5)Test Code2:

Enter Mu software and open the file "Project 5: Code-2.py" to import code:

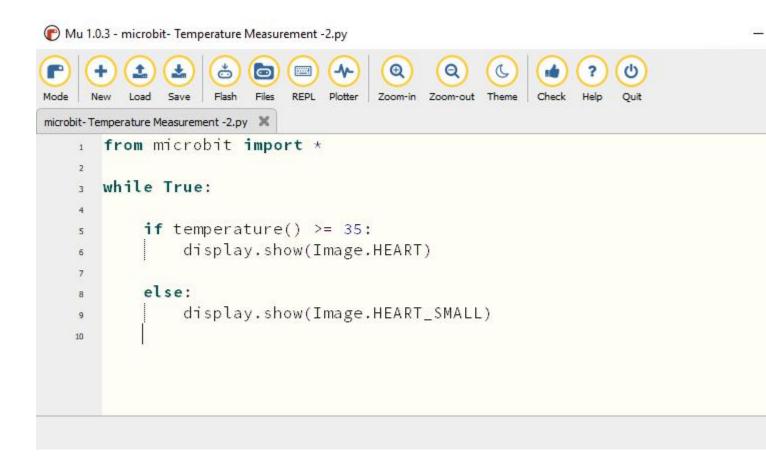
| File Type   | Route           | File Name            |
|-------------|-----------------|----------------------|
| Python file | KS4031(KS4032)  | Project 5: Code-2.py |
|             | folder/Python   |                      |
|             | Tutorial/Python |                      |



| Code/Project   | 5      | •  |
|----------------|--------|----|
| Temperature De | etecti | on |

You can also input code in the editing window yourself.(note:all English words and symbols must be written in English)

The temperature value can be set in compliance with the real temperature.



Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.



Mu 1.0.3 - microbit- Temperature Measurement -2.py Q P + ÷ ð -1-Q C t  $\odot$ 100 M C ? Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check Help Quit Mode New Load microbit-Temperature Measurement -2.py 🛛 🗙 from microbit import \* 1 2 while True: 3 4 if temperature() >= 35: 5 display.show(Image.HEART) 6 7 else: 8 display.show(Image.HEART\_SMALL) 9 10

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.



Mu 1.0.3 - microbit- Temperature Measurement -2.py P ÷ ð -11 ഗ 0 Done of Q Q a ? Flash Files REPL Plotter Mode New Load Save Zoom-in Zoom-out Theme Check Help Quit microbit- Temperature Measurement -2.py from microbit import \* 1 2 while True: 3 4 if temperature() >= 35: 5 display.show(Image.HEART) 6 7 else: 8 display.show(Image.HEART\_SMALL) 9 10

#### (6)Test Results2:

After uploading the code 2 to the board, when the ambient temperature is

less than 35 °C, the 5\*5 LED dot matrix shows



. When the



temperature is equivalent to or greater than  $35 \,^\circ\!$ C, the pattern appears.



# (7)Code Explanation:

| from microbit import *                             | Import the library file of micro: |
|--|-----------------------------------|
|  | bit                               |
| while True:  | This is a permanent loop that     |
|  | makes micro:bit execute the       |
|  | code of it.                       |
| Temperature = temperature()                        | Set temperature() to              |
|  | Temperature                       |
| <pre>print("Temperature:", Temperature, "C")</pre> | BBC micro:bit REPL prints         |
|  | temperature value                 |
| sleep(500)   | Delay in 500ms                    |
| <b>if</b> temperature() > = 35:                    | If temperature value ≥35°C        |
| display.show(Image.HEART)                          | micro:bit shows "• "              |
| else:  | If temperature value<35°C         |
| display.show(Image.HEART_SMALL)                    | micro:bit displays "🔡"            |

# Project 6: Geomagnetic Sensor



(1)Project Description



This project mainly introduces the use of the Micro:bit' s compass. In addition to detecting the strength of the magnetic field, it can also be used to determine the direction, an important part of the heading and attitude reference system (AHRS) as well.

It uses FreescaleMAG3110 three-axis magnetometer. Its I2C interface communicates with the outside, the range is  $\pm 1000\mu$ T, the maximum data update rate is 80Hz. Combined with accelerometer, it can calculate the position. Additionally, it is applied to magnetic detection and compass blocks.

Then we could read the value detected by it to determine the location. We need to calibrate the Micro:bit board when magnetic sensor works.

The correct calibration method is to rotate the Micro:bit board.

In addition, the objects nearby may affect the accuracy of readings and calibration.

o

## (2) Preparations:

A. Attach the Micro:bit main board to your computer via the USB cable;B.Open the offline version of Mu.

## (3)Test Code1::

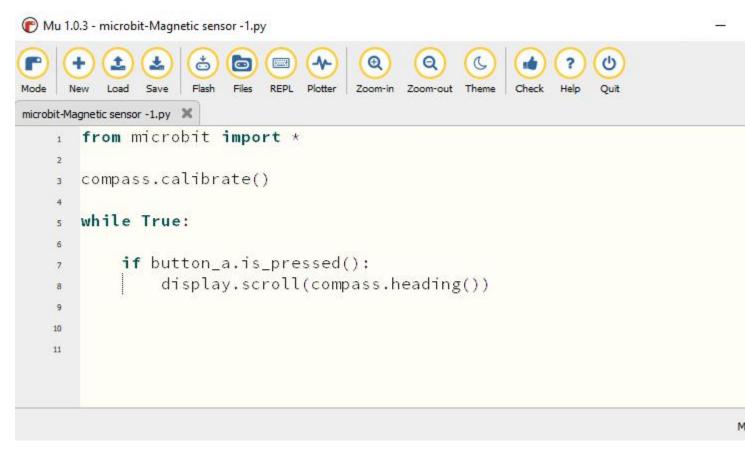
Enter Mu software and open the file "Project 6: Code-1.py" to import code:



| File Type | Route              |   | File Name            |
|-----------|--------------------|---|----------------------|
| Python    | KS4031(KS4032)     |   | Project 6: Code-1.py |
| file      | folder/Python      |   |                      |
|           | Tutorial/Python    |   |                      |
|           | Code/Project       | 6 |                      |
|           | Geomagnetic Sensor |   |                      |

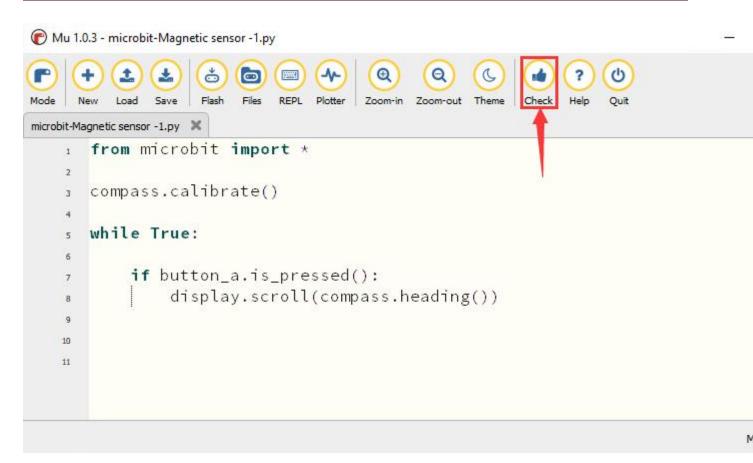
You can also input code in the editing window yourself.

(note:all English words and symbols must be written in English)



Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.





If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.

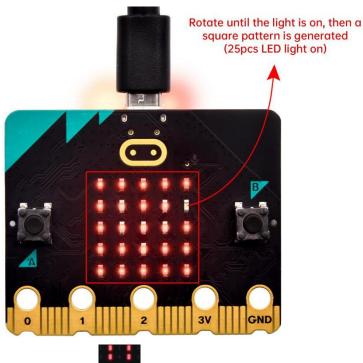


Mu 1.0.3 - microbit-Magnetic sensor -1.py ÷ ð -11-Q ഗ P 0 Q C ? Flash Files REPL Mode New Save Plotter Zoom-in Zoom-out Theme Check Help Quit Load microbit-Magnetic sensor -1.py 💥 from microbit import \* 1 2 compass.calibrate() 3 4 while True: 5 б if button\_a.is\_pressed(): 7 display.scroll(compass.heading()) 9 10 11

### (4)Test Result1:

After uploading test code1 to micro:bit main board and powering the board via the USB cable, and pressing the button A, the board asks us to calibrate compass and the LED dot matrix shows "TILT TO FILL SCREEN". Then enter the calibration page. Rotate the board until all 25 red LEDs are on as shown below.



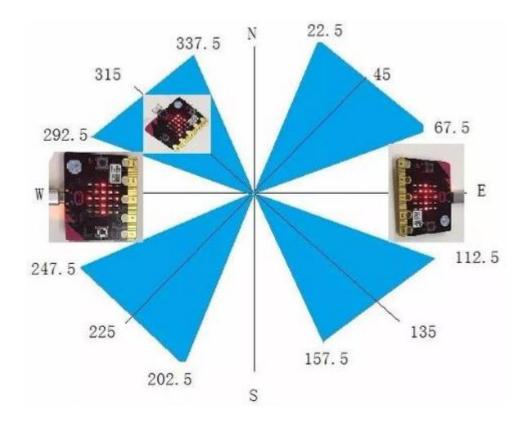


After that, a smile pattern appears, which implies the calibration is done. When the calibration process is completed, pressing the button A will make the magnetometer reading display directly on the screen. And the direction north, east, south and west correspond to 0°, 90°, 180° and 270° respectively.

### (5)Test Code2:

For the below picture, the arrow pointing to the upper right when the value ranges from 292.5 to 337.5. Because 0.5 can't be input in the code, the values we get are 293 and 338.

Then add other statements to make a set of complete code.



Enter Mu software and open the file "Project 6: Code-2.py" to import code:

| File Type | Route              |   | File Name            |
|-----------|--------------------|---|----------------------|
| Python    | KS4031(KS4032)     |   | Project 6: Code-2.py |
| file      | folder/Python      |   |                      |
|           | Tutorial/Python    |   |                      |
|           | Code/Project       | 6 |                      |
|           | Geomagnetic Sensor |   |                      |

You can also input code in the editing window yourself.(note:all English words and symbols must be written in English)



| 🕜 Mu 1     | .0.3 - microbit-Magnetic sensor -2.py -                                     | <u></u> |
|------------|---|---------|
| Mode (     | + Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check Help Quit |         |
| microbit-M | lagnetic sensor -2.py 🗶   |         |
| 1          | from microbit import *  |         |
| 2          | compass.calibrate()   |         |
| 3          | $\mathbf{x} = \mathbf{\Theta}$  |         |
| 4          | while True:   |         |
| 5          | x = compass.heading()   |         |
| 6          | if x >= 293 and x < 338:  |         |
| 7          | display.show(Image("00999:""00099:""00909:""09000:""9000                    | 0"      |
| 8          | elif $x \ge 23$ and $x \le 68$ :  |         |
| 9          | display.show(Image("99900:""99000:""90900:""00090:""00090                   | 9"      |
| 10         | elif x >= 68 and x < 113:   |         |
| 11         | display.show(Image("00900:""09000:""999999:""09000:""0090                   | 0"      |
| 12         | elif x >= 113 and x < 158:  |         |
| 13         | display.show(Image("00009:""00090:""90900:""99000:""9990                    | 0"      |
| 14         | elif x >= 158 and x < 203:  |         |
| 15         | display.show(Image("00900:""00900:""90909:""09990:""0090                    | 0"      |
| 16         | elif x >= 203 and x < 248:  |         |
| 17         | display.show(Image("90000:""09000:""00909:""00099:""00099                   | 9"      |
| 18         | elif x >= 248 and x < 293:  |         |
| 19         | display.show(Image("00900:""00090:""999999:""00090:""0090                   | 0"      |
| 20         | else:   |         |
| 21         | display.show(Image("00900:""09990:""90909:""00900:""00900                   | 0"      |
| 22         |   |         |

Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.



Mu 1.0.3 - microbit-Magnetic sensor -2.py Q Q ථ 0 personal second Theme Check Mode New Load Save Flash Files REPL Plotter Zoom-in Zoom-out Help Ouit microbit-Magnetic sensor -2.py 💥 from microbit import \* compass.calibrate() 2 x = 03 while True: 4 x = compass.heading() 5 if x >= 293 and x < 338: 6 display.show(Image("00999:""00099:""00909:""09000:""90000" 7 elif x >= 23 and x < 68: 8 display.show(Image("99900:""99000:""90900:""00090:""00009" 9 elif x >= 68 and x < 113: 10 display.show(Image("00900:""09000:""99999:""09000:""00900" 11 elif x >= 113 and x < 158: 12 display.show(Image("00009:""00090:""90900:""99000:""99900" 13 elif x >= 158 and x < 203: 14 display.show(Image("00900:""00900:""90909:""09990:""00900" 15 elif x >= 203 and x < 248: 16 display.show(Image("90000:""09000:""00909:""00099:""00999" 17 elif x >= 248 and x < 293: 18 display.show(Image("00900:""00090:""99999:""00090:""00900" 19 else: 20 display.show(Image("00900:""09990:""90909:""00900:""00900" 21 22 N

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.



```
Mu 1.0.3 - microbit-Magnetic sensor -2.py
                                              Q
P
                                       Q
                                                                  ധ
                    ð
                        0
                                 -1-
                                                    C
                            1000
                   Flash
                            REPL Plotter
Mode
     New
         Load
             Save
                        Files
                                      Zoom-in
                                            Zoom-out
                                                  Theme
                                                        Check
                                                             Help
                                                                  Ouit
microbit-Magnetic sensor -2.py 🗙
      from microbit import *
      compass.calibrate()
    2
      x = 0
    3
      while True:
    4
           x = compass.heading()
    5
           if x >= 293 and x < 338:
    6
                display.show(Image("00999:""00099:""00909:""09000:""90000"
    7
           elif x >= 23 and x < 68:
    8
                display.show(Image("99900:""99000:""90900:""00090:""00009"
    9
           elif x >= 68 and x < 113:
   10
                display.show(Image("00900:""09000:""99999:""09000:""00900"
   11
           elif x >= 113 and x < 158:
   12
                display.show(Image("00009:""00090:""90900:""99000:""99900"
   13
           elif x >= 158 and x < 203:
   14
                display.show(Image("00900:""00900:""90909:""09990:""00900"
   15
           elif x >= 203 and x < 248:
   16
                display.show(Image("90000:""09000:""00909:""00099:""00999"
   17
           elif x >= 248 and x < 293:
   18
                display.show(Image("00900:""00090:""99999:""00090:""00900"
   19
           else:
   20
                display.show(Image("00900:""09990:""90909:""00900:""00900"
   21
   22
```

#### (6)Test Results2:

Upload code 2 and plug micro:bit into power. After calibration, tilt micro:bit board, and the LED dot matrix displays the direction signs.

#### (6)Code Explanation:

| from | microbit | import * | Import the |  |
|------|----------|----------|------------|--|
|------|----------|----------|------------|--|

N



|                                   | library file of  |
|-----------------------------------|------------------|
|                                   | micro: bit       |
| compass.calibrate()               | Compass          |
|                                   | calibration      |
| while True:                       | This is a        |
|                                   | permanent        |
|                                   | loop that        |
|                                   | makes            |
|                                   | micro:bit        |
|                                   | execute the      |
|                                   | code of it.      |
| if button_a.is_pressed():         | When the         |
| display.scroll(compass.heading()) | button A is      |
|                                   | pressed          |
|                                   | Micro:bit        |
|                                   | scrolls to show  |
|                                   | the value of     |
|                                   | compass          |
| x = 0                             | Set variable     |
|                                   | x=0              |
| x = compass.heading()             | Set the value of |
|                                   | compass to       |



|  | variable x     |
|--|----------------|
| ifelifelse   | Condition      |
|  | judgement      |
|  | statement:ifel |
|  | se ifelse      |
| display.show(Image("00999:""00099:""00909:""09000:""9  | Micro:bit      |
| 0000"))  | shows the      |
| display.show(Image("99900:""99000:""90900:""00090:""0  | Northeast      |
| 0009"))  | arrow sign     |
| display.show(Image("00900:""09000:""999999:""09000:""0 | Micro:bit      |
| 0900"))  | shows the      |
| display.show(Image("00009:""00090:""90900:""99000:""9  | Northwest      |
| 9900"))  | arrow sign     |
| display.show(Image("00900:""00900:""90909:""09990:""0  | Micro:bit      |
| 0900"))  | shows the west |
| display.show(Image("90000:""09000:""00909:""00099:""0  | arrow sign     |
| 0999"))  | Micro:bit      |
| display.show(Image("00900:""00090:""999999:""00090:""0 | shows the      |
| 0900"))  | Southwest      |
| display.show(Image("00900:""09990:""90909:""00900:""0  | arrow sign     |
| 0900"))  | Micro:bit      |
|  | shows the      |



| South arrow    |
|----------------|
| sign           |
| Micro:bit      |
| shows the      |
| South arrow    |
| sign           |
| Micro:bit      |
| shows the East |
| arrow sign     |
| Micro:bit      |
| shows the      |
| North arrow    |
| sign           |
|                |

# **Project 7: Accelerometer**



(1) Project Introduction



The Micro: Bit main board V2 has a built-in LSM303AGR gravity acceleration sensor, also known as accelerometer, with a resolution of 8/10/12 bits. The code section sets the range to 1g, 2g, 4g, and 8g. We often use accelerometer to detect the status of machines. In this project, we will introduce how to measure the position of the board with the accelerometer. And then have a look at the original three-axis data output by the accelerometer.

#### (2)Preparations:

- A. Attach the Micro:bit main board to your computer via the USB cable;
- B. Open the offline version of Mu.

### (3)Test Code1:

Enter Mu software and open the file "Project 7: Accelerometer-1.py" to import code:

### (How to load the project code?)

| File Type | Route            | File Name          |
|-----------|------------------|--------------------|
| Python    | KS4031(KS4032)   | Project 7 :        |
| file      | folder/Python    | Accelerometer-1.py |
|           | Tutorial/Python  |                    |
|           | Code/Project 7 : |                    |



Accelerometer

You can also input code in the editing window yourself.(note:all English

### words and symbols must be written in English)

| 🕜 Mu 1 | .0.3 - microbit-Three-axis acceleration sensor -1.py  | 3 <u>945</u> |
|--------|---|--------------|
|        | Image: Save       Image: Save |              |
| 1      | from microbit import *  |              |
| 2      |   |              |
| 3      | while True:   |              |
| 4      | gesture = accelerometer.current_gesture()   |              |
| 5      |   |              |
| 6      | <pre>if gesture == "shake":</pre>   |              |
| 7      | display.show("1")   |              |
| 8      | <pre>if gesture == "up":</pre>  |              |
| 9      | display.show("2")   |              |
| 10     | <pre>if gesture == "down":</pre>  |              |
| 11     | display.show("3")   |              |
| 12     | <pre>if gesture == "face up":</pre>   |              |
| 13     | display.show("4")   |              |
| 14     | <pre>if gesture == "face down":</pre>   |              |
| 15     | display.show("5")   |              |
| 16     | <pre>if gesture == "left":</pre>  |              |
| 17     | display.show("6")   |              |
| 18     | <pre>if gesture == "right":</pre>   |              |
| 19     | display.show("7")   |              |
| 20     | <pre>if gesture == "freefall":</pre>  |              |
| 21     | display.show("8")   |              |
| 22     |   |              |
|        |   | N            |
|        |   |              |

Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.

N



| Mode N      | +<br>Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Replace Help Quit |  |
|-------------|--|--|
| microbit-Th | nree-axis acceleration sensor -1.py 🔀  |  |
| 1           | from microbit import *   |  |
| 2           |  |  |
| 3           | while True:  |  |
| 4           | gesture = accelerometer.current_gesture()  |  |
| 5           |  |  |
| 6           | <pre>if gesture == "shake":</pre>  |  |
| 7           | display.show("1")  |  |
| 8           | <pre>if gesture == "up":</pre>   |  |
| 9           | display.show("2")  |  |
| 10          | <pre>if gesture == "down":</pre>   |  |
| 11          | display.show("3")  |  |
| 12          | <pre>if gesture == "face up":</pre>  |  |
| 13          | display.show("4")  |  |
| 14          | <pre>if gesture == "face down":</pre>  |  |
| 15          | display.show("5")  |  |
| 16          | <pre>if gesture == "left":</pre>   |  |
| 17          | display.show("6")  |  |
| 18          | <pre>if gesture == "right":</pre>  |  |
| 19          | display.show("7")  |  |
| 20          | <pre>if gesture == "freefall":</pre>   |  |
| 21          | display.show("8")  |  |
| 22          |  |  |

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.

N



| 🕜 Mu 1.               | 0.3 - microbit-Three-axis acceleration sensor -1.py                            |  |
|-----------------------|--|--|
| Mode N                | +<br>Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check Help Quit |  |
| and the second second | nree-axis acceleration sensor -1.py  |  |
| 1                     | from microbit import *   |  |
| 2                     |  |  |
| 3                     | while True:  |  |
| 4                     | <pre>gesture = accelerometer.current_gesture()</pre>                           |  |
| 5                     |  |  |
| 6                     | <pre>if gesture == "shake":</pre>  |  |
| 7                     | display.show("1")  |  |
| 8                     | <pre>if gesture == "up":</pre>   |  |
| 9                     | display.show("2")  |  |
| 10                    | <pre>if gesture == "down":</pre>   |  |
| 11                    | display.show("3")  |  |
| 12                    | <pre>if gesture == "face up":</pre>  |  |
| 13                    | display.show("4")  |  |
| 14                    | <pre>if gesture == "face down":</pre>  |  |
| 15                    | display.show("5")  |  |
| 16                    | <pre>if gesture == "left":</pre>   |  |
| 17                    | display.show("6")  |  |
| 18                    | <pre>if gesture == "right":</pre>  |  |
| 19                    | display.show("7")  |  |
| 20                    | <pre>if gesture == "freefall":</pre>   |  |
| 21                    | display.show("8")  |  |
| 22                    |  |  |
|                       |  |  |

## (4)Test Results1:

After uploading the test code 1 to micro:bit main board and powering the board via the USB cable, if we shake the Micro: Bit main board, no matter at any direction, the LED dot matrix displays the digit "1".

When it is kept upright (make its logo above the LED dot matrix), the number 2 shows.





When it is kept upside down( make its logo below the LED dot matrix), it

shows as below.



When it is placed still on the desk, showing its front side, the number 4 appears.

When it is placed still on the desk, showing its back side, the number 5 exhibits.

When the board is tilted to the left, the LED dot matrix shows the number 6 as shown below.





When the board is tilted to the right, the LED dot matrix displays the number 7 as shown below:



When the board is knocked to the floor, this process can be considered as a free fall and the LED dot matrix shows the number 8. (Please note that this test is not recommended for it may damage the main board.) Attention: if you' d like to try this function, you can also set the acceleration to 3g, 6g or 8g. But still ,we do not recommend.

### (5)Test code2:

Enter Mu software and open the file "Project 7: Accelerometer-2.py" to import code:

#### (<u>How to load the project code?</u>)

| File Type | Route          | File Name          |
|-----------|----------------|--------------------|
| Python    | KS4031(KS4032) | Project 7 :        |
| file      | folder/Python  | Accelerometer-2.py |



| Tutorial/Python |   |   |
|-----------------|---|---|
| Code/Project    | 7 | : |
| Accelerometer   |   |   |

You can also input code in the editing window yourself.(note:all English

words and symbols must be written in English)

| •          | +<br>Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check Help Quit | · — |
|------------|--|-----|
| microbit-T | hree-axis acceleration sensor -2.py 🔀  |     |
| 1          | <pre>from microbit import *</pre>  |     |
| 2          |  |     |
| 3          | while True:  |     |
| 4          |  |     |
| 5          | <pre>x = accelerometer.get_x()</pre>   |     |
| 6          |  |     |
| 7          | y = accelerometer.get_y()  |     |
| 8          |  |     |
| 9          | z = accelerometer.get_z()  |     |
| 10         |  |     |
| 11         | print("x, y, z:", x, y, z)   |     |
| 12         |  |     |
| 13         | sleep(100)   |     |
| 14         |  |     |
|            |  |     |

Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.



| P (         | +<br>Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check Help Quit |  |
|-------------|--|--|
| nicrobit-Tl | nree-axis acceleration sensor -2.py 🗶  |  |
| 1           | from microbit import *   |  |
| 2           |  |  |
| 3           | while True:  |  |
| 4           |  |  |
| 5           | <pre>x = accelerometer.get_x()</pre>   |  |
| 6           |  |  |
| 7           | y = accelerometer.get_y()  |  |
| 8           |  |  |
| 9           | z = accelerometer.get_z()  |  |
| 10          |  |  |
| 11          | print("x, y, z:", x, y, z)   |  |
| 12          |  |  |
| 13          | sleep(100)   |  |
| 14          |  |  |

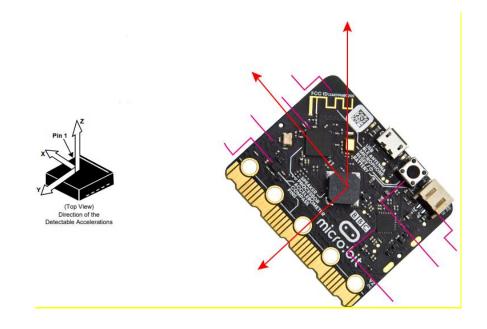
If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.



Mu 1.0.3 - microbit-Three-axis acceleration sensor -2.py P -1-÷ ð Q Q ഗ t 0 C ? 10 Mode New Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check Help Quit microbit-Three-axis acceleration sensor -2.py from microbit import \* 1 2 while True: 3 4 x = accelerometer.get\_x() 5 6 y = accelerometer.get\_y() 7 8 z = accelerometer.get\_z() 9 10 print("x, y, z:", x, y, z) 11 12 sleep(100) 13 14

After referring to the MMA8653FC data manual and the hardware schematic diagram of the Micro: Bit main board, the accelerometer coordinate of the Micro: Bit are shown in the figure below: N





#### (6)Test Results2:

Upload the test code 1 to micro:bit main board and power the board via the USB cable.

Click "REPL" and press the reset button. The value of acceleration on X axis, Y axis and Z axis are shown below:



| 🕜 Mu 1.     | 0.3 - microbit-Three-axis acceleration sensor -2.py -   |  |  |  |  |
|-------------|---|--|--|--|--|
| Mode        | Image: New Load     Image: Save     Image: Flash     Image: Flash <th< th=""></th<> |  |  |  |  |
| microbit-Th | iree-axis acceleration sensor -2.py 🔀   |  |  |  |  |
| 1           | from microbit import *  |  |  |  |  |
| 2           |   |  |  |  |  |
| 3           | while True:   |  |  |  |  |
| 4           |   |  |  |  |  |
| 5           | <pre>x = accelerometer.get_x()</pre>  |  |  |  |  |
| 6           | <pre>y = accelerometer.get_y()</pre>  |  |  |  |  |
| 7           | z = accelerometer.get_z()   |  |  |  |  |
| 8           | print("x, y, z:", x, y, z)  |  |  |  |  |
| 9           | sleep(100)  |  |  |  |  |
| 10          |   |  |  |  |  |
| BBC mici    | ro:bit REPL   |  |  |  |  |

| 1000 | and the second se | and the second se | Concernance of the local distance of the loc | COLUMN TWO IS NOT |      |
|------|---|---|--|-------------------|------|
| х,   | у,  | z:  | -12  | 732               | -788 |
| х,   | у,  | Ζ:  | -32  | 696               | -752 |
| х,   |   | z:  | -16  | 752               | -780 |
|      |   | z:  | -12  | 724               | -752 |
| х,   | у,  | z:  | -20  | 732               | -756 |
| х,   | у,  | z:  | -8   | 724 -             | -760 |
| х,   | у,  | z:  | 0 72   | 20 -1             | 772  |
| х,   | у,  | z:  | 4 72   | 24 -1             | 780  |
| х,   | у,  | z:  | -24  | 716               | -776 |
| х,   | у,  | z:  | -12  | 712               | -752 |
| х,   | у,  | z:  | -16  | 712               | -768 |
| х,   | у,  | z:  | -24  | 684               | -760 |
| х,   | у,  | z:  | -20  | 684               | -776 |
| х,   | у,  | z:  | -28  | 708               | -768 |
| х,   | у,  | z:  | -40  | 684               | -756 |
| х,   | у,  | z:  | -32  | 692               | -748 |
| х,   | у,  | z:  | -32  | 660               | -732 |
| х,   | у,  | z:  | -52  | 660               | -732 |

# (7)Code Explanation:

| from | microbit | import * | Import the library file of micro: bit |
|------|----------|----------|---------------------------------------|
|------|----------|----------|---------------------------------------|

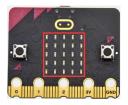


| gesture =                         | Set accelerometer.current_gesture()  |
|-----------------------------------|--------------------------------------|
| accelerometer.current_gesture()   | to gesture                           |
| while True:                       | This is a permanent loop that makes  |
|                                   | micro:bit execute the code of it.    |
| <b>if</b> gesture == "shake":     | Shaking micro:bit board, number 1    |
| display.show("1")                 | will appear                          |
| <b>if</b> gesture == "up":        | When log points to the North,        |
| display.show("2")                 | number 2 will show up.               |
| <b>if</b> gesture == "down":      | When log points to the South,        |
| display.show("3")                 | number 3 will be shown               |
| <b>if</b> gesture == "face up":   | When the LED dot matrix is upward,   |
| display.show("4")                 | the number 4 is shown.               |
| <b>if</b> gesture == "face down": | the number 5 is displayed when the   |
| display.show("5")                 | LED dot matrix is downward.          |
| <b>if</b> gesture == "left":      | When Micro:bit board is tilt to the  |
| display.show("6")                 | left, number 6 is shown.             |
| <b>if</b> gesture == "right":     | When micro:bit is tilt to the right  |
| display.show("7")                 | When Micro:bit board is inclined to  |
| <b>if</b> gesture == "freefall":  | the right, number 7 is displayed.    |
| display.show("8")                 | When it is free fall(accidentally    |
|                                   | making it fall), number 8 appears on |
|                                   | dot matrix.                          |



| x = accelerometer.get_x()  | Read the acceleration value on x       |
|----------------------------|--|
| y = accelerometer.get_y()  | axis, the return value is integer, and |
| z = accelerometer.get_z()  | set x= the read value on x axis        |
|                            | Read the acceleration value on y       |
|                            | axis, the return value is integer, and |
|                            | set y= the read value on y axis        |
|                            | Read the acceleration value on z       |
|                            | axis, the return value is integer, and |
|                            | set z= the read value on z axis        |
| print("x, y, z:", x, y, z) | The value of acceleration will be      |
|                            | shown                                  |
| sleep(100)                 | Delay in 100ms                         |

## **Project 8: Light Detection**



## (1) Project Introduction

In this project, we focus on the light detection function of the Micro: Bit main board. It is achieved by the LED dot matrix since the main board is not equipped with a photoresistor.



## (2) Preparations:

A. Attach the Micro:bit main board to your computer via the USB cable;

B. Open the offline version of Mu.

## (3)Test Code:

Enter Mu software and open the file "project 8: Light Detection.py" to import code:

(How to load the project code?)

| File Type   | Route                     | File Name                     |
|-------------|---------------------------|-------------------------------|
| Python file | KS4031(KS4032)            | project 8: Light Detection.py |
|             | folder/Python             |                               |
|             | Tutorial/Python           |                               |
|             | Code/Project Code/Project |                               |
|             | 8: Light Detection        |                               |

You can also input code in the editing window yourself.

(note:all English words and symbols must be written in English)



Mu 1.0.3 - microbit-Detect Light Intensity by Microbit .py P + ÷ ð -1-Q Q C t 0 (WWW) C ? Flash REPL Plotter Mode New Load Save Files Zoom-in Zoom-out Theme Check Help Quit microbit-Detect Light Intensity by Microbit .py from microbit import \* 1 2 while True: 3 4 Lightintensity = display.read\_light\_level() 5 6 print("Light intensity:", Lightintensity) 7 8 sleep(100) 9 10

Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.

N



Mu 1.0.3 - microbit-Detect Light Intensity by Microbit .py P + ÷ ð -1-Q Q C t 0 C ? 100 Flash REPL Plotter Zoom-in Zoom-out Theme Check Help Mode New Load Save Files Quit microbit-Detect Light Intensity by Microbit .py from microbit import \* 1 2 while True: 3 4 Lightintensity = display.read\_light\_level() 5 6 print("Light intensity:", Lightintensity) 7 8 sleep(100) 9 10

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.

N



P Mu 1.0.3 - microbit-Detect Light Intensity by Microbit .py P + ð Q Q C  $\odot$ a . ? 0000 A Flash REPL New Save Files Plotter Zoom-in Zoom-out Theme Check Help Quit Mode microbit-Detect Light Intensity by Microbit .py from microbit import \* 1 2 while True: 3 Lightintensity = display.read\_light\_level() 5 6 print("Light intensity:", Lightintensity) 7 sleep(100) 9 10

#### (4)Test Results:

Upload the test code to micro:bit main board, power the board via the USB cable and click "Show console Device".

Download code onto micro:bit board, don' t plug off USB cable. Click "REPL" and press the reset buttons, the light intensity value will be displayed, as shown below.

When the LED dot matrix is covered by hand, the light intensity showed is approximately 0; when the LED dot matrix is exposed to light, the light intensity displayed gets stronger with the light.



| P Mu 1.0.3 - microbit-Detect Light Intensity by Microbit .py  |  |  |  |  |
|---|--|--|--|--|
| Image: Control of the state       Image: Control of the state |  |  |  |  |
| 1 from microbit import *  |  |  |  |  |
| 2   |  |  |  |  |
| 3 while True:   |  |  |  |  |
| 4   |  |  |  |  |
|   |  |  |  |  |
|   |  |  |  |  |
| <pre>6 print("Light intensity:", Lightintensity)</pre>  |  |  |  |  |
| 7 sleep(100)  |  |  |  |  |
| 8   |  |  |  |  |
| BBC micro:bit REPL  |  |  |  |  |
| Light intensity: 1  |  |  |  |  |
| Light intensity: 2  |  |  |  |  |
| Light intensity: 8  |  |  |  |  |
| Light intensity: 21   |  |  |  |  |
| Light intensity: 94   |  |  |  |  |
| Light intensity: 220  |  |  |  |  |
| Light intensity: 221<br>Light intensity: 198  |  |  |  |  |
| Light intensity: 92   |  |  |  |  |
| Light intensity: 47   |  |  |  |  |
| Light intensity: 40   |  |  |  |  |
| Light intensity: 51   |  |  |  |  |
| Líght intensitý: 91   |  |  |  |  |
|   |  |  |  |  |

# (5)Code Explanation:

| from microbit import *          | Import the library file of micro: bit |
|---------------------------------|---------------------------------------|
| gesture =                       | Set accelerometer.current_gesture()   |
| accelerometer.current_gesture() | to gesture                            |
| while True:                     | This is a permanent loop that makes   |
|                                 | micro:bit execute the code of it.     |
| Lightintensity =                | Set display.read_light_level() to     |
| display.read_light_level()      | Lightintensity                        |



| print("Light intensity:", | BBC microbit REPL prints the   |
|---------------------------|--------------------------------|
| Lightintensity)           | detected light intensity value |
| sleep(100)                | Delay in 100ms                 |

**Project 9: Speaker** 



(1) **Project Introduction** 

Micro: Bit main board has an built-in speaker, which makes adding sound to the programs easier. It can also be programmed to air all kinds of tones, like playing the song *Ode to Joy*.

## (2)Preparations:

A. Attach the Micro:bit main board to your computer via the USB cable;

B.Open the offline version of Mu.

## (3)Test Code1:

Enter Mu software and open the file "Project 9: Speaker.py" to import code:

(How to load the project code?)

| File Type | Route | File Name |
|-----------|-------|-----------|
|-----------|-------|-----------|



| Python file | KS4031(KS4032)          | Project   | 9 | : |
|-------------|-------------------------|-----------|---|---|
|             | folder/Python           | Speaker.p | у |   |
|             | Tutorial/Python         |           |   |   |
|             | Code/Project 9: Speaker |           |   |   |

You can also input code in the editing window yourself.

(note:all English words and symbols must be written in English)

| lu 1.1.0.beta.2 - Speaker.py —  |
|---|
| Image: Save set of the same set |
| from microbit import *  |
|   |
| import audio  |
|   |
| while True:   |
| audio.play(Sound.GIGGLE)  |
| sleep(1000)   |
| audio.play(Sound.HAPPY)   |
| sleep(1000)   |
| audio.play(Sound.HELLO)   |
| sleep(1000)   |
| audio.play(Sound.YAWN)  |
| sleep(1000)   |
|   |
|   |
| BBC micro:bi  |
|   |

Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.



| Mode   | u 1.1.0.beta.2 - Speaker.py |      |
|--|-----------------------------|------|
| Speak<br>1<br>2<br>3<br>4<br>5<br>6<br>7<br>8<br>9<br>10<br>11<br>12<br>13<br>14 |                             |      |
|  | BBC micr                    | o:bi |

If the code is correct, connect micro:bit to computer and click "Flash" to

download code to micro:bit board.



| Mode<br>Speake        | lu 1.1.0.beta.2 - Speaker.py                                      |     |  |
|-----------------------|---|-----|--|
| ореаке<br>1<br>2<br>3 |   |     |  |
| 4                     | while True:   |     |  |
| 6<br>7                | 7 sleep(1000)   |     |  |
| 9<br>10               | audio.play(Sound.HAPPY)<br>sleep(1000)<br>audio.play(Sound.HELLO) |     |  |
| 11<br>12              | sleep(1000)<br>audio.play(Sound.YAWN)                             |     |  |
| 13<br>14              | sleep(1000)   |     |  |
|                       | BBC micro   | o:b |  |

## (4)Test Results1:

After uploading the test code1 to micro:bit main board and powering the board via the USB cable, the speaker utters sound and the LED dot matrix shows the logo of music.

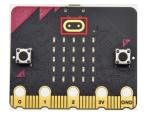
### (5)Code Explanation:

| from microbit import * | Import the library of micro: bit |  |
|------------------------|----------------------------------|--|
| import audio           | Audio library                    |  |
| while True:            | This is a permanent loop that    |  |



|                          | makes micro:bit execute the code |
|--------------------------|----------------------------------|
|                          | of it.                           |
| audio.play(Sound.GIGGLE) | Emit the "giggle" sound          |
| sleep(1000)              | delay in 1000ms                  |

## **Project 10: Touch-sensitive Logo**



## (1) **Project Introduction**

The Micro: Bit main board V2 is equipped with a golden touch-sensitive logo, which can act as an input component and function like an extra button.

It contains a capacitive touch sensor that senses small changes in the electric field when pressed (or touched), just like your phone or tablet screen do.When you press it, you can activate the program.



## (2) Preparations:

A. Attach the Micro:bit main board to your computer via the USB cable;



B.Open the offline version of Mu.

## (3)Test Code:

Enter Mu software and open the file "Project 10: Touch-sensitive Logo.py"

to import code:

(How to load the project code?)

| File Type   | Route                | File Name                   |
|-------------|----------------------|-----------------------------|
| Python file | KS4031(KS4032)       | Project 10: Touch-sensitive |
|             | folder/Python        | Logo.py                     |
|             | Tutorial/Python      |                             |
|             | Code/Project 10 :    |                             |
|             | Touch-sensitive Logo |                             |
|             |                      |                             |



You can also input code in the editing window yourself.

## (note:all English words and symbols must be written in English)

| Mu (      | 1.1.0.beta.2 - microbit-Touch Sensitive Logo.py -   | > |
|-----------|---|---|
| P)<br>ode | Image: Save       Image: Save |   |
| icrobi    | t-Touch Sensitive Logo.py 🗙   |   |
|           | rom microbit import *   |   |
|           | fme = 0   |   |
|           | tart = 0  |   |
|           | unning = False  |   |
| 5         |   |   |
| - 24 M    | hile True:  |   |
| 7         | if button a was proceed().  |   |
| 8         | <pre>if button_a.was_pressed():     running = True</pre>  |   |
| 9         | start = running_time()  |   |
| 10        | if button_b.was_pressed():  |   |
| 12        | if running:   |   |
| 13        | time += running_time() - start  |   |
| 14        | running = False   |   |
| 15        | if pin_logo.is_touched():   |   |
| 16        | if not running:   |   |
| 17        | display.scroll(int(time/1000))  |   |
| 18        |   |   |
| 19        | if running:   |   |
| 20        | display,show(Image.HEART)   |   |
| 21        | sleep(300)  |   |
| 22        | display.show(Image.HEART_SMALL)   |   |
| 23        | sleep(300)  |   |
| 24        | else:   |   |
| 25        | display.show(Image.ASLEEP)  |   |
| 26        |   |   |
|           | BBC micro:bit 🇰 🦑   | - |

### How Micro:bit works?

- A. The runtime is recorded in milliseconds(ms).
- B. When you press button A, a variable named start is set to the current running time.
- C. When you press button B, the start time will be subtracted from the new



running time to calculate how much time has passed since you started the stopwatch. This difference is added to the total time, which is stored in a variable named time.

- D. If you press the golden logo, the program will display the total elapsed time on the LED display. It converts time from milliseconds (thousandths of a second) to seconds by dividing by 1000. It uses the integer division operator to give an integer (integer) result.
- E. The program is also controlled by a Boolean variable named running. Boolean variable can only have two values: true or false. If "running" is "true", it means that the stopwatch has started. If "running" is false, it means that the stopwatch has not started or has stopped.
- F. If "running" is true, the beating heart pattern is displayed on the LED dot matrix screen.
- G. (7) If the stopwatch has stopped and the "running" is false, when you press the golden logo, it will only display the time.
- H. If the stopwatch has been started and "running" is true, it only need to ensure that the time variable will only change when button B is pressed, and the code can also prevent false readings.

Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.



| Ð             |   |
|---------------|---|
| ode<br>i croł | New Load Save   Flash Files REPL Plotter   Zoom-in Zoom-out Theme   Check Tidy Help   Quit<br>Dit-Touch Sensitive Logo.py 🗶 |
|               | from microbit import *  |
|               | time = 0  |
|               | start = 0   |
| 4             | running = False   |
| 5             | while True:   |
| - 31 - 1      | white frue.   |
| 7             | <pre>if button_a.was_pressed():</pre>   |
| 9             | running = True  |
| 10            | <pre>start = running_time()</pre>   |
| 11            | if button_b.was_pressed():  |
| 12            | if running:   |
| 13            | time += running_time() - start  |
| 14            | running = False   |
| 15            | <pre>if pin_logo.is_touched():</pre>  |
| 16            | if not running:   |
| 17            | display.scroll(int(time/1000))  |
| 18            |   |
| 19            | if running:   |
| 20            | display.show(Image.HEART)   |
| 21            | sleep(300)<br>display.show(Image.HEART_SMALL)   |
| 22            | sleep(300)  |
| 24            | else:   |
| 25            | display.show(Image.ASLEEP)  |
| 26            |   |
|               | BBC micro:bit 🗰 🐔   |

If the code is correct, connect micro:bit to computer and click "Flash" to

download code to micro:bit board.



| Ом        | u 1.1.0.beta.2 - microbit-Touch Sensitive Logo.py —  | ×      |
|-----------|--|--------|
| P)<br>ode | Image: Save     Imag | )<br>t |
|           | bit-Touch Sensitive Logo.py 💦  |        |
|           | from microbit import *   |        |
| _         | time = 0   |        |
|           | start = 0  |        |
|           | running = False  |        |
| 5         | while True:  |        |
|           | white irde.  |        |
| 7.<br>8   | <pre>if button_a.was_pressed():</pre>  |        |
| 9         | running = True   |        |
| 10        | <pre>start = running_time()</pre>  |        |
| 11        | if button_b.was_pressed():   |        |
| 12        | if running:  |        |
| 13        | time += running_time() - start   |        |
| 14        | running = False  |        |
| 15        | if pin_logo.is_touched():  |        |
| 16        | if not running:  |        |
| 17        | display.scroll(int(time/1000))   |        |
| 18        |  |        |
| 19        | if running:  |        |
| 20        | display.show(Image.HEART)  |        |
| 21        | sleep(300)   |        |
| 22        | display.show(Image.HEART_SMALL)  |        |
| 23        | sleep(300)   |        |
| 24        | else:  |        |
| 25        | display.show(Image.ASLEEP)   | 1      |
| 26        |  |        |
|           | BBC micro:bit  | O      |

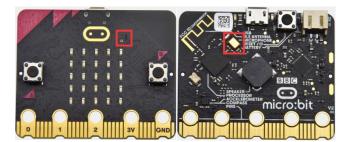
#### (4)Test Results:

Upload the test code to micro:bit main board and power the board via the USB cable, and press button A to start the stopwatch. When timing, the beating heart pattern will be displayed on the LED dot matrix screen. Press button B to stop it and you can start and stop it at any time. It will keep recording time, just like a real stopwatch. Press the golden logo on the front of the micro:bit to display the measured time in seconds. And time



can be reset to zero by pressing the reset button on the back of it.

### **Project 11: Microphone**



(1) Project Introduction

The Micro: Bit main board is built with a microphone which can test the volume of ambient environment. When you clap, the microphone LED indicator turns on. Since it can measure the intensity of sound, you can make a noise scale or disco lighting changing with music. The microphone is placed on the opposite side of the microphone LED indicator and in proximity with holes that lets sound pass. When the board detects sound, the LED indicator lights up.

#### (2) Preparations:

A. Attach the Micro:bit main board to your computer via the USB cable;B.Open the offline version of Mu.



## (3)Test Code 1:

Enter Mu software and open the file "Project 11: Microphone-1.py" to import code:

(How to load the project code?)

| File Type   | Route             | File Name       |
|-------------|-------------------|-----------------|
| Python file | KS4031(KS4032)    | Project 11 :    |
|             | folder/Python     | Microphone-1.py |
|             | Tutorial/Python   |                 |
|             | Code/Project 11 : |                 |
|             | Microphone        |                 |

You can also input code in the editing window yourself. (note:all English words and symbols must be written in English)



| Mode                  | Au 1.1.0.beta.2 - Microphone-1.py  | ?<br>Help | Quit | × |
|-----------------------|--|-----------|------|---|
| Micro                 | ophone-1.py X<br>from microbit import *  |           |      |   |
| 2                     | while True:  |           |      |   |
| 4<br>5<br>7<br>8<br>9 | <pre>if microphone.current_event() == SoundEvent.LOUD:     display.show(Image.HEART)     sleep(200) if microphone.current_event() == SoundEvent.QUIET:     display.show(Image.HEART_SMALL)</pre> |           |      |   |
|                       |  |           |      |   |
|                       |  | ro:bit    |      | * |

Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.



| om microbit import * |   |   |  |  |
|----------------------|---|---|--|--|
|                      |   |   | N  |  |
|                      |   |   |  |  |
|                      |   | nt.LOUD:  |  |  |
|                      | EART)   |   |  |  |
|                      | ant () Caundha  | st OUTET.   |  |  |
|                      |   | nt.QUIET:   |  |  |
|                      | CART_SMALL)   |   |  |  |
|                      |   |   |  |  |
|                      |   |   |  |  |
|                      |   |   |  |  |
|                      |   |   |  |  |
|                      |   |   |  |  |
|                      |   |   |  |  |
|                      |   |   |  |  |
|                      |   |   |  |  |
|                      | display.show(Image.H<br>sleep(200)<br>if microphone.current_eve | <pre>if microphone.current_event() == SoundEve<br/>display.show(Image.HEART)<br/>sleep(200)<br/>if microphone.current_event() == SoundEve<br/>display.show(Image.HEART_SMALL)</pre> | <pre>if microphone.current_event() == SoundEvent.LOUD:<br/>display.show(Image.HEART)<br/>sleep(200)<br/>if microphone.current_event() == SoundEvent.QUIET:<br/>display.show(Image.HEART_SMALL)</pre> | <pre>if microphone.current_event() == SoundEvent.LOUD;     display.show(Image.HEART)     sleep(200) if microphone.current_event() == SoundEvent.QUIET:     display.show(Image.HEART_SMALL)</pre> |

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.



| Mode   | Au 1.1.0.beta.2 - Microphone-1.py -  | Quit | × |
|--|--|------|---|
| Mi cr (<br>1<br>2<br>3<br>4<br>5<br>6<br>7<br>8<br>9 | <pre>ophone=1.py * from microbit import * while True:     if microphone.current_event() == SoundEvent.LOUD:         display.show(Image.HEART)         sleep(200)     if microphone.current_event() == SoundEvent.QUIET:         display.show(Image.HEART_SMALL) </pre> |      |   |
|  | BBC micro:bit  |      | ¢ |

### (4)Test Results1:

After uploading test code to micro:bit main board and powering the board via the USB cable, the LED dot matrix displays pattern """ when you claps and pattern 💀 when it is quiet around.

### (5)Test Code2:

Enter Mu software and open the file "Project 11: Microphone-2.py" to import code:

(How to load the project code?)

| File Type | Route | File Name |
|-----------|-------|-----------|
|-----------|-------|-----------|



| Python file | KS4031(KS4032)  |    |   | Project    | 11     | • |
|-------------|-----------------|----|---|------------|--------|---|
|             | folder/Python   |    |   | Microphone | e-2.py |   |
|             | Tutorial/Python |    |   |            |        |   |
|             | Code/Project    | 11 | • |            |        |   |
|             | Microphone      |    |   |            |        |   |

You can also input code in the editing window yourself.

### (note:all English words and symbols must be written in English)

| C N   | /u 1.1.0.beta.2 - microbit-Microphone-2.py —   |      | $\times$ |
|-------|--|------|----------|
| Mode  | Image: Market state     Image: Market st | Quit |          |
| micro | obit-Microphone-2. py 🗙  |      |          |
| T.    | from microbit import *   |      |          |
| 2     | maxSound = 0   |      |          |
| 3     | lights = Image("11111:"  |      |          |
| 4     | "11111:"   |      |          |
| 5     | "11111:"   |      |          |
| 6     | "11111:"   |      |          |
| 7     | <b>"11111"</b> )   |      |          |
| 8     | <pre># ignore first sound level reading coundlevel = microphone cound level()</pre>  |      |          |
| 9     | <pre>soundLevel = microphone.sound_level() sleep(200)</pre>  |      |          |
| 10    | steep(200)   |      |          |
| 11    | while True:  |      |          |
| 12    | <pre>if button_a.is_pressed():</pre>   |      |          |
| 14    | display.scroll(maxSound)   |      |          |
| 15    | else:  |      |          |
| 16    | soundLevel = microphone.sound_level()  |      |          |
| 17    | display.show(lights * soundLevel)  |      |          |
| 18    | if soundLevel > maxSound:  |      |          |
| 19    | maxSound = soundLevel  |      |          |
| 10    |  |      |          |
|       | BBC micro:bit  | -    | 12       |
|       |  | -    | M.       |

Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.



| ) N | /lu 1.1.0.beta.2 - microbit-Microphone-2.py -  | ×     |
|-----|--|-------|
| de  | +     + <th>2</th>   | 2     |
| cro | obit-Microphone-2.py 🗶 🕇 🕇 🕇 The second se |       |
| 2   | maxSound = 0   |       |
| 3   | lights = Image("11111:"  |       |
| 4   | "11111:"   |       |
| 5   | "11111:"   |       |
| 6   | "11111:"   |       |
| 7   | "11111")   |       |
| 8   | # ignore first sound level reading   |       |
| 8   | soundLevel = microphone.sound_level()  |       |
| 10  | sleep(200)   |       |
| 10  | 3166p(200)   |       |
| 12  | while True:  |       |
| 11  | <pre>if button_a.is_pressed():</pre>   |       |
| 14  | display.scroll(maxSound)   |       |
| 15  | else:  |       |
| 16  | soundLevel = microphone.sound_level()  |       |
| 17  | display.show(lights * soundLevel)  |       |
| 18  | if soundLevel > maxSound:  |       |
| 19  | maxSound = soundLevel  |       |
| 10  | Indx50una - 50unacevec   |       |
|     | BBC micro:bit  | - 14  |
|     | DDL micro.Dit  | F 345 |

If the code is correct, connect micro:bit to computer and click "Flash" to

download code to micro:bit board.



|              | Au 1.1.0.beta.2 - microbit-Microphone-2.py       -       □         + ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ | ×       |
|--------------|--|---------|
| lode<br>icro | New Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check Tidy Help Quit                         |         |
| 1            | from microbit import *   |         |
| 2            | maxSound = $\Theta$  |         |
| 3            | lights = Image("11111:"  |         |
| 4            | "11111:"   |         |
| 5            | "11111:"   |         |
| 6            | "11111:"   |         |
| 7            | "11111")   |         |
| 8            | # ignore first sound level reading   |         |
| 9            | soundLevel = microphone.sound_level()  |         |
| 10           | sleep(200)   |         |
| 11           |  |         |
| 12           | while True:  |         |
| 11           | <pre>if button_a.is_pressed():</pre>   |         |
| 14           | display.scroll(maxSound)   |         |
| 15           | else:  |         |
| 16           | soundLevel = microphone.sound_level()  |         |
| 17           | display.show(lights * soundLevel)  |         |
| 18           | if soundLevel > maxSound:  |         |
| 19           | maxSound = soundLevel  |         |
| 10           |  |         |
|              | BBC micro:bit 🇰 4  | A       |
|              | DDC mitro.or.  | Aug St. |

### (6)Test Results2:

Upload test code to micro:bit main board and power the board via the USB cable. When the button A is pressed, the LED dot matrix displays the value of the biggest volume( please note that the biggest volume can be reset via the Reset button on the other side of the board ) while when clapping, the LED dot matrix shows the pattern of the sound.



# (7)Code Explanation:

| from microbit import *                               | Import the library of    |
|--|--------------------------|
|  | micro: bit               |
| while True:  | This is a permanent loop |
|  | that makes micro:bit     |
|  | execute the code of it.  |
| <b>if</b> microphone.current_event() ==              | If there is a sound      |
| SoundEvent.LOUD:                                     | LED shows 🗢              |
| display.show(Image.HEART)                            | Delay in 200ms           |
| sleep(200)   | if no sound is detected  |
| <b>if</b> microphone.current_event() ==              | LED lights show          |
| SoundEvent.QUIET:                                    |                          |
| display.show(Image.HEART_SMALL)                      |                          |
| <pre>print("Light intensity:", Lightintensity)</pre> | BBC microbit REPL prints |
|  | the detected light       |
|  | intensity value          |
| maxSound = 0   | The initial value of     |
|  | maxSound is 0            |
| lights =   | Assign Image() to        |
| Image("11111:""11111:""11111:""11111:""11111         | variable lights          |
| ")   |                          |
| soundLevel = microphone.sound_level()                | Assign                   |



|                                       | microphone.sound_level     |
|---------------------------------------|----------------------------|
|                                       | () to the variable         |
|                                       | soundLevel                 |
| if button_a.is_pressed():             | if the button A is pressed |
| display.scroll(maxSound)              | LED lights show the        |
| else:                                 | sound value                |
| soundLevel = microphone.sound_level() | If not                     |
| display.show(lights * soundLevel)     | Assign                     |
| if soundLevel > maxSound:             | microphone.sound_level     |
| maxSound = soundLevel                 | () to the variable         |
|                                       | soundLevel                 |
|                                       | As the sound changes,      |
|                                       | the micro:bit will display |
|                                       | the breathing light effect |
|                                       | If the sound value is      |
|                                       | higher than its maximum    |
|                                       | value                      |
|                                       | the maximum sound          |
|                                       | value is equal to sound    |
|                                       | level value                |

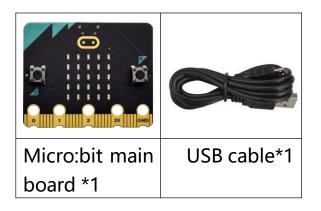


### **Project 12: Touch-sensitive Logo Controlled Speaker**

### (1) **Project Introduction**

In the previous projects, we have learned about the touch-sensitive logo and the speaker respectively. In the project, we will combine these two components to play music. That' s the logo will be applied to control the speaker to sing songs.

### (2)Components Needed:



### (3)Connection Diagram:

Attach the Micro:bit main board to your computer via the USB cable.





### (4)Test Code:

Enter Mu software and open the file "Project 12: Touch-sensitive Logo

Controlled Speaker.py" to import code:

(How to load the project code?)

| File   | Route              |      | File Name                        |
|--------|--------------------|------|----------------------------------|
| Туре   |                    |      |                                  |
| Python | KS4031(KS4032)     |      | Project 12: Touch-sensitive Logo |
| file   | folder/Python      |      | Controlled Speaker.py            |
|        | Tutorial/Python    |      |                                  |
|        | Code/Project 12    | :    |                                  |
|        | Touch-sensitive    | Logo |                                  |
|        | Controlled Speaker |      |                                  |

You can also input code in the editing window yourself.

(note:all words and symbols must be written in English)



| ode | Image: Head Save     Image: Head Save <th>Quit</th> <th></th> | Quit |  |
|-----|---|------|--|
| oje | ect 12: Touch the Logo to control the speaker py 🛛 🕱  |      |  |
| 1   | from microbit import *  |      |  |
| 2   |   |      |  |
| 3   | import Music  |      |  |
| 4   |   |      |  |
| 5   | display.show(Image.MUSIC_QUAVER)  |      |  |
| 6   |   |      |  |
| 7   | while True:   |      |  |
| 8   | if nin loca is touched().   |      |  |
| 9   | <pre>if pin_logo.is_touched():     music.play(music.BIRTHDAY)</pre>   |      |  |
| 10  |   |      |  |
|     |   |      |  |
|     |   |      |  |
|     |   |      |  |
|     |   |      |  |
|     |   |      |  |
|     |   |      |  |
|     |   |      |  |
|     |   |      |  |

Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.



| ) N     | /lu 1.1.0.beta.2 - Project 12: Touch the Logo to control the speaker.py  | 1000      |      | >   |
|---------|--|-----------|------|-----|
| de      | Image: Save     Imag | ?<br>Help | Quit |     |
| roje    | ect 12: Touch the Logo to control the speaker.py 🛛 🗶   |           |      |     |
| 1       | from microbit import *   |           |      |     |
| 2       |  |           |      |     |
| 3       | import music   |           |      |     |
| 4       | dieplay show (Image WELC OUNTED)   |           |      |     |
| 年二、     | display.show(Image.MUSIC_QUAVER)   |           |      |     |
| 6       | while True:  |           |      |     |
| 8       | wirte irwe.  |           |      |     |
| а<br>в: | <pre>if pin_logo.is_touched():</pre>   |           |      |     |
| 10      | music.play(music.BIRTHDAY)   |           |      |     |
| 11      |  |           |      |     |
|         |  |           |      |     |
|         |  |           |      |     |
|         |  |           |      |     |
|         |  |           |      |     |
|         |  |           |      |     |
|         |  |           |      |     |
|         |  |           |      |     |
|         |  |           |      |     |
|         | BBC mic  | ro hit    | E    | 275 |

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.



|     | 1u 1.1.0.beta.2 - Project 12: Touch the Logo to control the speaker.py        | 0    |    |
|-----|---|------|----|
| )   |   | 0    |    |
| le  | New Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check Tidy Help | Quit |    |
| oje | ect 12: Touch the Logo to control the speaker.py 🛛 🗶                          |      |    |
| t   | from microbit import *  |      |    |
| 2   |   |      |    |
| 1   | import MUSIC  |      |    |
| \$  |   |      |    |
| 1   | display.show(Image.MUSIC_QUAVER)  |      |    |
| 5   |   |      |    |
| F   | while True:   |      |    |
|     | <pre>if pin_logo.is_touched():</pre>  |      |    |
| 9   | music.play(music.BIRTHDAY)  |      |    |
| 1   | ind retp tay (ind ret bit (hb/t))   |      |    |
| 8   |   |      |    |
|     |   |      |    |
|     |   |      |    |
|     |   |      |    |
|     |   |      |    |
|     |   |      |    |
|     |   |      |    |
|     |   |      |    |
|     | BBC micro:bit   |      | 10 |
|     | DDC micro. Dit  | -    | 34 |

# (5)Test Results:

After uploading test code to micro:bit main board and powering the board via the USB cable, the speaker plays the song *Happy Birthday to You* when the logo is touched.

### (6)Code Explanation:

| from microbit import * | Import the library of micro: bit |
|------------------------|----------------------------------|
| while True:            | This is a permanent loop that    |
|                        | makes micro:bit execute the      |
|                        | code of it.                      |



| display.show (Image.MUSIC_QUAVER) | Music logo shows on the LED  |
|-----------------------------------|------------------------------|
|                                   | dot matrix on the micro:bit  |
| if pin_logo. is_touched( ):       | When the logo is touched, it |
|                                   | executes the following       |
|                                   | command                      |
| music.play (music.BIRTHDAY)       | The speaker plays the song"  |
|                                   | Happy Birthday to You"       |

### **Bluetooth Wireless Communication**

With 16k RAM, micro:bit owns a low-consumption Bluetooth module and support Bluetooth communication. However, BLE heap stack occupies 12K RAM, which implies that there is no enough space to run microPython. At present, microPython doesn't support Bluetooth.

https://microbit-micropython.readthedocs.io/en/latest/ble.html

The former projects are the introduction of sensors and modules. The further lessons are challenging for new starters. (Note: In order to prevent the micro:bit board from being burned,



disconnect the micro USB cable from it and turn off the power on the micro:bit motor drive board before installing it on the car expansion board and dial the POWER switch to the OFF end; likewise, before removing the the main board from the car expansion board, disconnect the micro USB cable from it and turn off the power on the micro:bit motor drive backplane.

### **Project 13:Colorful Lights**

### (1)Project Description

This module consists of a commonly used LED with 7colors but in white appearance. It can automatically flash different colors to create fantastic light effects when high level is input like a normal LED.

### (2) Experimental Preparation:

- > Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car
- > Place batteries into battery holder
- > Dial power switch to ON end
- Connect micro:bit to computer by USB cable
- > Open the offline version of Mu.



### (3)Test Code:

Enter Mu software and open the file "Project 13: Colorful Lights.py" to import code:

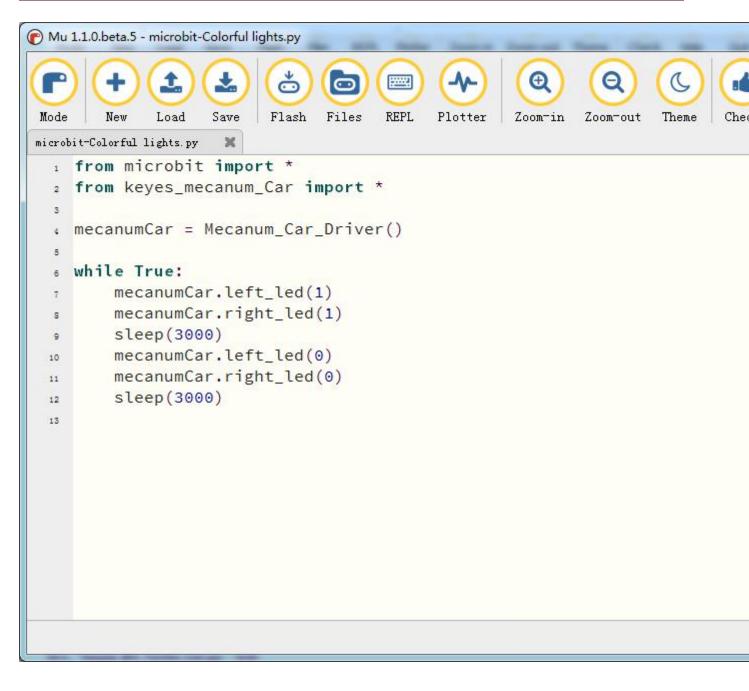
(How to load the project code?)

| File   | Route                      | File Name                      |
|--------|----------------------------|--------------------------------|
| Туре   |                            |                                |
| Python | KS4031(KS4032)             | Project 13: Colorful Lights.py |
| file   | folder/Python              |                                |
|        | Tutorial/Python            |                                |
|        | Code/Project 13 : Colorful |                                |
|        | Lights                     |                                |

You can also input code in the editing window yourself.

(note:all words and symbols must be written in English)





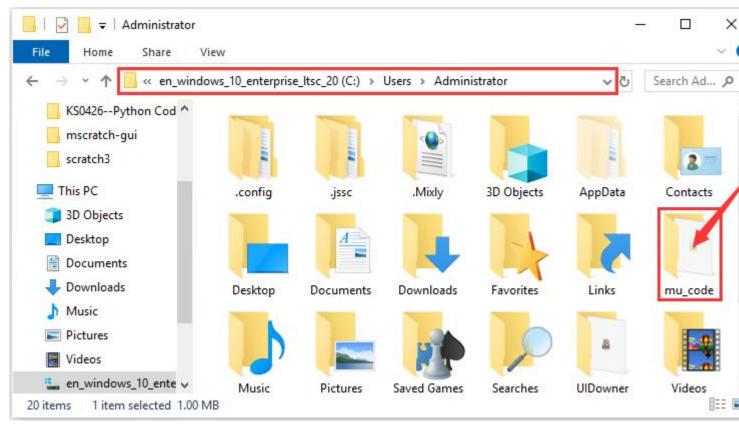
Don't click "Flash", but import the "keyes\_mecanum\_Car.py" library file into the micro:bit. This file contains the control method of the Micro:bit Mini Smart Mecanum Wheel Smart Car.



#### Import the "keyes\_mecanum\_Car.py" library file.

The default directory where Mu saves files is "Mu\_code", which is located in the root directory of the user directory. Reference link: <a href="https://codewith.mu/en/tutorials/1.0/files">https://codewith.mu/en/tutorials/1.0/files</a>

For example, in the windows system, suppose your system is installed on the C drive of the computer, and the user name is "Administrator", then the path of the "mu\_code" directory is "C:\Users\Administrator\mu\_ code". On Linux systems, the path of the "mu\_code" directory is "~/home/mu\_code".



#### Enter the "mu\_code" folder.

Copy "keyes\_mecanum\_Car.py "library file to folder" mu\_code " and the



path is :

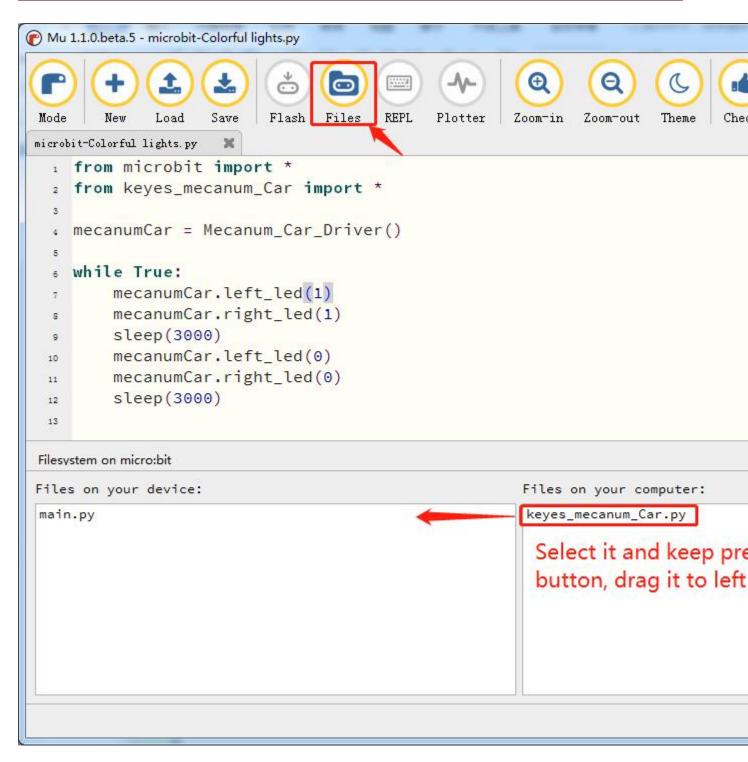
| File   | Path                                   | File name       |
|--------|--|-----------------|
| type   |  |                 |
| Pytho  | /PythonCode/LibrariesmecanumCar_python | keyes_mecanum_C |
| n file | _Libraries                             | ar.py           |

## When the copy is done, it should be look like this:

| test.txt             | 2020/8/21 14:36 | 文本文档  | 1 KE |
|----------------------|-----------------|-------|------|
| keyes_mecanum_Car.py | 2021/8/17 13:42 | PY 文件 | 5 KI |
| templates            | 2021/7/15 15:08 | 文件夹   |      |
| static               | 2021/7/15 15:08 | 文件夹   |      |
| sounds               | 2020/6/15 8:04  | 文件夹   |      |
| music                | 2020/6/15 8:04  | 文件夹   |      |
| images               | 2020/6/15 8:04  | 文件夹   |      |
| fonts                | 2020/6/15 8:04  | 文件夹   |      |
| data_capture         | 2021/8/17 15:14 | 文件夹   |      |

First open the Mu software and connect the micro:bit to your computer, then click the "Files", and then drag the "keyes\_mecanum\_Car.py" library file to micro:bit.





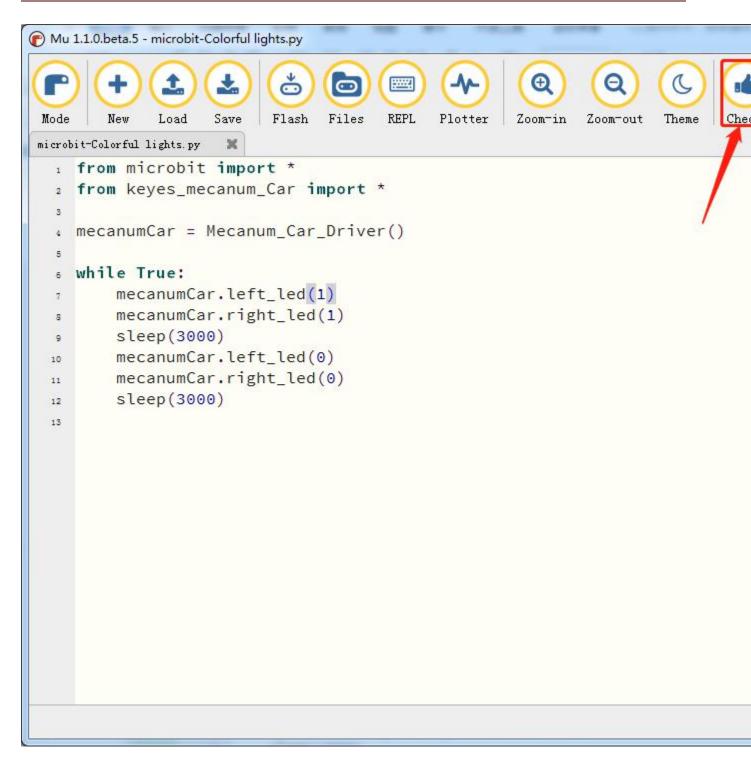
After a few seconds, the import is complete and you can see it in the box on the left.



| ilesystem on micro:bit         |                        |
|--------------------------------|------------------------|
| iles on your device:           | Files on your computer |
| eyes_mecanum_Car.py<br>main.py | keyes_mecanum_Car.py   |

After the library file is imported successfully, you also need to click the "Check" button to check the code for errors. If a cursor or an underline appears on a certain line, it indicates that there is an error in the program.





However, during this process, the following prompt will appear even if there is no error in the code. These prompts are just warnings, not code error prompts. In other words, the entire code is error-free.

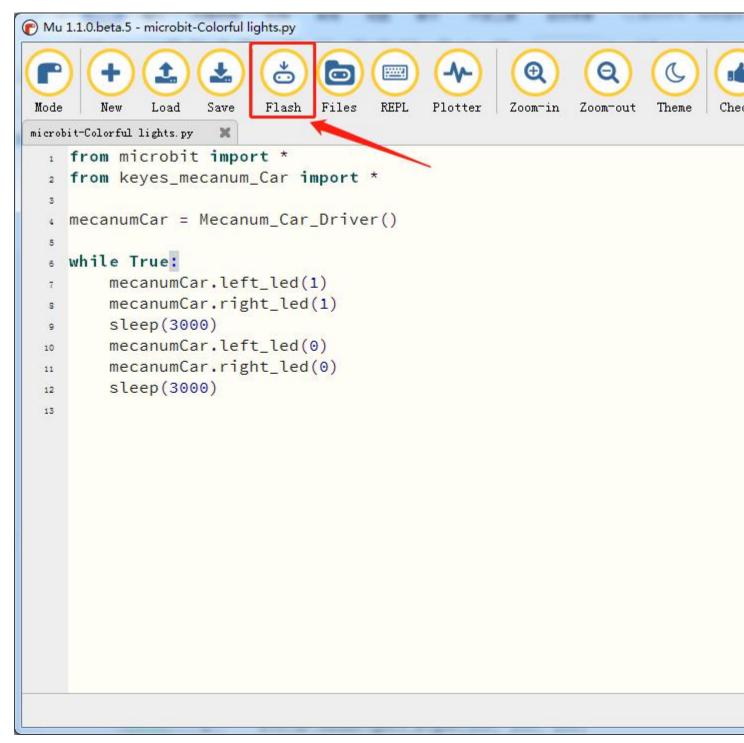
↑ 'from keyes\_mecanum\_Car import \*' used; unable to detect undefined name



↑ 'Mecanum\_Car\_Driver' may be undefined, or defined from star imports: k

If the code is correct, connect micro:bit to computer and click "Flash" to

download code to micro:bit board.



If it indicates an error after clicking the "Flash" button, please confirm whether you have imported the "keyes\_mecanum\_Car.py" library file that we provided to micro:bit.

#### Note:

Before programming with Micropython, you need to import the "keyes\_mecanum\_Car.py" library file to the micro:bit. If you program with different micro:bit, the library file"keyes\_mecanum\_Car.py" needs to be imported again to a new micro:bit.

### (4)Test Results:

Download code to micro:bit board and dial POWER switch to ON end, 2 RGB lights of smart car flash in 3s and then stop in 3s and repeat this pattern.

### (5) Code Explanation:

| from microbit import *                     | Import the library file of micro: |
|--|-----------------------------------|
|  | bit                               |
| <pre>from keyes_mecanum_Car import *</pre> | Import the library file of        |



|                                   | keyes_mecanum_Car                   |  |  |
|-----------------------------------|-------------------------------------|--|--|
| mecanumCar = Mecanum_Car_Driver() | Instantiate an object               |  |  |
|                                   | Mecanum_Car_Driver() as             |  |  |
|                                   | mecanumCar                          |  |  |
| while True:                       | This is a permanent loop that       |  |  |
|                                   | makes micro:bit execute the         |  |  |
|                                   | code of it.                         |  |  |
| mecanumCar.left led(1)            | Light up the colorful light on the  |  |  |
|                                   | left. (1 is on, 0 is off)           |  |  |
| mecanumCar.right led(1)           | Light up the colorful lights on     |  |  |
|                                   | the right. (1 is on, 0 is off)      |  |  |
| sleep(3000)                       | Delay in 3000ms                     |  |  |
| mecanumCar.left led(0)            | Turn off the colorful lights on the |  |  |
|                                   | left. (1 is on, 0 is off)           |  |  |
| mecanumCar.right led(0)           | Turn off the colorful lights on the |  |  |
|                                   | right. (1 is on, 0 is off)          |  |  |
|                                   |                                     |  |  |



### Project 14:WS2812 RGB LEDs



(1)Project Description

The driver shield cooperates 4 pcs WS2812 RGB LEDs, compatible with micro:bit board and controlled by P8. In this lesson, we will make RGB LEDs display different colors by P8. In this lesson, 3 sets of test code are provided to make the 4 WS2812 RGB LEDs display different effects.

| Sampl | Color | RGB       | Color   | Sampl | Color  | RGB       | Color    |
|-------|-------|-----------|---------|-------|--------|-----------|----------|
| e     |       | Value     | Code    | е     |        | Value     | Code     |
|       |       | (R,G,B)   | (16     |       |        | (R,G,B)   | (16      |
|       |       |           | colors) |       |        |           | colors)) |
|       | Red   | 255, 0, 0 | #FF000  |       | Orang  | 255, 165, | #FFA500  |
|       |       |           | 0       |       | е      | 0         |          |
|       | Yello | 255, 255, | #FFFF00 |       | Green  | 0, 255, 0 | #00FF00  |
|       | w     | 0         |         |       |        |           |          |
|       | Blue  | 0, 255, 0 | #0000F  |       | Indigo | 75, 0,    | #4B008   |
|       |       |           | F       |       |        | 130       | 2        |



|   | Violet | 238, 130, | #EE82EE |       | Purple | 160, 32,  | #A020F  |
|---|--------|-----------|---------|-------|--------|-----------|---------|
|   |        | 238       |         |       |        | 240       | 0       |
|   | Black  | 0, 0, 0   | #00000  |       | White  | 255, 255, | #FFFFFF |
|   |        |           | 0       |       |        | 255       |         |
| •••••   |        |           |         | ••••• |        |           |         |
| Change the value of the R,G and B to get different colors |        |           |         |       |        |           |         |

### (2) Experimental Preparation:

- > Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car
- > Place batteries into battery holder
- > Dial power switch to ON end
- Connect micro:bit to computer by USB cable
- > Open the offline version of Mu.

#### (3)Test Code:

#### Code1:

Enter Mu software and open the file "Project 14: WS2812 RGB LEDs.py"

to import code:

#### (How to load the project code?)

| File | Route | File Name |
|------|-------|-----------|
| Туре |       |           |



| Python | KS4031(KS4032)           | Code-1.py |
|--------|--------------------------|-----------|
| file   | folder/Python            |           |
|        | Tutorial/Python          |           |
|        | Code/Project 14 : WS2812 |           |
|        | RGB LEDs                 |           |

You can also input code in the editing window yourself.

### (note:all words and symbols must be written in English)

P Mu 1.0.3 - microbit-4 WS2812 RGB lights-1.py

| Mode N     | ew Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check Help Quit |
|------------|--|
| microbit-4 | VS2812 RGB lights-1.py 🐹   |
| 1          | <pre>from microbit import *</pre>  |
| 2          | import neopixel  |
| 3          | np = neopixel.NeoPixel(pin8, 4)  |
| 4          | while True:  |
| 5          | <pre>for pixel_id1 in range(0, len(np)):</pre>                               |
| 6          | np[pixel_id1] = (255, 0, 0)  |
| 7          | np.show()  |
| 8          | sleep(1000)  |
| 9          | <pre>for pixel_id2 in range(0, len(np)):</pre>                               |
| 10         | np[pixel_id2] = (255, 165, 0)  |
| 11         | np.show()  |
| 12         | sleep(1000)  |
| 13         | <pre>for pixel_id3 in range(0, len(np)):</pre>                               |
| 14         | np[pixel_id3] = (255, 255, 0)  |
| 15         | np.show()  |
| 16         | sleep(1000)  |
| 17         | <pre>for pixel_id4 in range(0, len(np)):</pre>                               |
| 18         | np[pixel_id4] = (0, 255, 0)  |
| 19         | np.show()  |
| 20         | sleep(1000)  |



```
for pixel_id5 in range(0, len(np)):
21
            np[pixel_id5] = (0, 0, 255)
22
            np.show()
23
       sleep(1000)
24
       for pixel_id6 in range(0, len(np)):
25
            np[pixel_id6] = (75, 0, 130)
26
            np.show()
27
       sleep(1000)
28
       for pixel_id7 in range(0, len(np)):
29
            np[pixel_id7] = (238, 130, 238)
30
            np.show()
31
       sleep(1000)
32
       for pixel_id8 in range(0, len(np)):
33
            np[pixel_id8] = (160, 32, 240)
34
            np.show()
35
       sleep(1000)
36
       for pixel_id9 in range(0, len(np)):
37
            np[pixel_id9] = (255, 255, 255)
38
       sleep(1000)
39
40
```

Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.

N



```
P Mu 1.0.3 - microbit-4 WS2812 RGB lights-1.py
                                         Q
P
      ÷
                     ð
                                                Q
                                                                      C
          1
               ÷
                                  -1-
                                                      C
                         ?
                    Flash
                                                           Check
Mode
     New
         Load Save
                         Files
                             REPL Plotter
                                      Zoom-in Zoom-out Theme
                                                                Help
                                                                     Quit
microbit-4 WS2812 RGB lights-1.py  🕷
      from microbit import *
    1
       import neopixel
    2
       np = neopixel.NeoPixel(pin8, 4)
    3
       while True:
    4
            for pixel_id1 in range(0, len(np)):
    5
                np[pixel_id1] = (255, 0, 0)
    6
                np.show()
    7
            sleep(1000)
    8
            for pixel_id2 in range(0, len(np)):
    9
                np[pixel_id2] = (255, 165, 0)
   10
                np.show()
   11
            sleep(1000)
   12
            for pixel_id3 in range(0, len(np)):
   13
                np[pixel_id3] = (255, 255, 0)
   14
                np.show()
   15
            sleep(1000)
   16
            for pixel_id4 in range(0, len(np)):
   17
                np[pixel_id4] = (0, 255, 0)
   18
                np.show()
   19
            sleep(1000)
   20
```



```
for pixel_id5 in range(0, len(np)):
21
            np[pixel_id5] = (0, 0, 255)
22
            np.show()
23
       sleep(1000)
24
       for pixel_id6 in range(0, len(np)):
25
            np[pixel_id6] = (75, 0, 130)
26
            np.show()
27
       sleep(1000)
28
       for pixel_id7 in range(0, len(np)):
29
            np[pixel_id7] = (238, 130, 238)
30
            np.show()
31
       sleep(1000)
32
       for pixel_id8 in range(0, len(np)):
33
            np[pixel_id8] = (160, 32, 240)
34
            np.show()
35
       sleep(1000)
36
       for pixel_id9 in range(0, len(np)):
37
            np[pixel_id9] = (255, 255, 255)
38
       sleep(1000)
39
40
```

If the code is correct, connect micro:bit to computer and click "Flash" to

N

download code to micro:bit board.



```
P Mu 1.0.3 - microbit-4 WS2812 RGB lights-1.py
                                         Q
                                                Q
      ÷
                     ð
                         0
P
          t
               ÷
                             1000
                                  -1-
                                                      C
                                                                 ?
                                                                      C
                                                            -
                    Flash
Mode
     New
         Load Save
                         Files REPL Plotter Zoom-in Zoom-out Theme
                                                          Check
                                                                Help
                                                                      Quit
microbit-4 WS2812 RGB lights-1.py  🕷
      from microbit import *
    1
       import neopixel
    2
       np = neopixel.NeoPixel(pin8, 4)
    3
       while True:
    4
            for pixel_id1 in range(0, len(np)):
    5
                np[pixel_id1] = (255, 0, 0)
    6
                np.show()
    7
            sleep(1000)
    8
            for pixel_id2 in range(0, len(np)):
    9
                np[pixel_id2] = (255, 165, 0)
   10
                np.show()
   11
            sleep(1000)
   12
            for pixel_id3 in range(0, len(np)):
   13
                np[pixel_id3] = (255, 255, 0)
   14
                np.show()
   15
            sleep(1000)
   16
            for pixel_id4 in range(0, len(np)):
   17
                np[pixel_id4] = (0, 255, 0)
   18
                np.show()
   19
            sleep(1000)
   20
```



```
for pixel_id5 in range(0, len(np)):
21
            np[pixel_id5] = (0, 0, 255)
22
            np.show()
23
       sleep(1000)
24
       for pixel_id6 in range(0, len(np)):
25
            np[pixel_id6] = (75, 0, 130)
26
            np.show()
27
       sleep(1000)
28
       for pixel_id7 in range(0, len(np)):
29
            np[pixel_id7] = (238, 130, 238)
30
            np.show()
31
       sleep(1000)
32
       for pixel_id8 in range(0, len(np)):
33
            np[pixel_id8] = (160, 32, 240)
34
            np.show()
35
       sleep(1000)
36
       for pixel_id9 in range(0, len(np)):
37
            np[pixel_id9] = (255, 255, 255)
38
       sleep(1000)
39
40
```

M

## Code2:

Enter Mu software and open the file "Project 14: WS2812 RGB LEDs.py"

to import code:

```
(How to load the project code?)
```

| File   | Route                    | File Name |
|--------|--------------------------|-----------|
| Туре   |                          |           |
| Python | KS4031(KS4032)           | Code-2.py |
| file   | folder/Python            |           |
|        | Tutorial/Python          |           |
|        | Code/Project 14 : WS2812 |           |



**RGB** LEDs

## You can also input code in the editing window yourself.

#### (note:all words and symbols must be written in English)

P Mu 1.0.3 - microbit-4 WS2812 RGB lights-2.py

| Mode       | New Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check |
|------------|---|
| microbit-4 | VS2812 RGB lights-2.py 🔀  |
| 1          | <pre>from microbit import *</pre>                                   |
| 2          | import neopixel   |
| 3          | np = neopixel.NeoPixel(pin8, 4)                                     |
| 4          | while True:   |
| 5          | for index in range(0, 4):   |
| 6          | np.clear()  |
| 7          | np[index] = (255, 0, 0)   |
| 8          | np.show()   |
| 9          | sleep(100)  |
| 10         | <pre>for index1 in range(0, 4):</pre>                               |
| 11         | np.clear()  |
| 12         | np[index1] = (255, 165, 0)  |
| 13         | np.show()   |
| 14         | sleep(100)  |
| 15         | for index2 in range(0, 4):  |
| 16         | np.clear()  |
| 17         | np[index2] = (255, 255, 0)  |
| 18         | np.show()   |
| 19         | sleep(100)  |
| 20         | for index3 in range(0, 4):  |
| 21         | np.clear()  |
| 22         | np[index3] = (0, 255, 0)  |
| 23         | np.show()   |
| 24         | sleep(100)  |



```
for index4 in range(0, 4):
25
            np.clear()
26
            np[index4] = (0, 0, 255)
27
            np.show()
28
            sleep(100)
29
        for index5 in range(0, 4):
30
            np.clear()
31
            np[index5] = (75, 0, 130)
32
            np.show()
33
            sleep(100)
34
        for index6 in range(0, 4):
35
            np.clear()
36
            np[index6] = (238, 130, 238)
37
            np.show()
38
            sleep(100)
39
        for index7 in range(0, 4):
40
            np.clear()
41
            np[index7] = (160, 32, 240)
42
            np.show()
43
            sleep(100)
44
        for index8 in range(0, 4):
45
            np.clear()
46
            np[index8] = (255, 255, 255)
47
            np.show()
48
            sleep(100)
49
50
```

Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.



| 🕜 Mu 1.    | .0.3 - microbit-4 WS2812 RGB lights-2.py   | 1    |
|------------|--|------|
| Mode       | Image: New Load     Image: Save     Image: Flash     Image: Flash <th< th=""><th>eck.</th></th<> | eck. |
| microbit-4 | WS2812 RGB lights-2.py   | 1    |
| 1          | from microbit import *   |      |
| 2          | import neopixel  | 1    |
| 3          | np = neopixel.NeoPixel(pin8, 4)  |      |
| 4          | while True:  |      |
| 5          | for index in range(0, 4):  |      |
| 6          | np.clear()   |      |
| 7          | np[index] = (255, 0, 0)  |      |
| 8          | np.show()  |      |
| 9          | sleep(100)   |      |
| 10         | for index1 in range(0, 4):   |      |
| 11         | np.clear()   |      |
| 12         | np[index1] = (255, 165, 0)   |      |
| 13         | np.show()  |      |
| 14         | sleep(100)   |      |
| 15         | for index2 in range(0, 4):   |      |
| 16         | np.clear()   |      |
| 17         | np[index2] = (255, 255, 0)   |      |
| 18         | np.show()  |      |
| 19         | sleep(100)   |      |
| 20         | for index3 in range(0, 4):   |      |
| 21         | np.clear()   |      |
| 22         | np[index3] = (0, 255, 0)   |      |
| 23         | np.show()  |      |
| 24         | sleep(100)   |      |



```
for index4 in range(0, 4):
25
            np.clear()
26
            np[index4] = (0, 0, 255)
27
            np.show()
28
            sleep(100)
29
        for index5 in range(0, 4):
30
            np.clear()
31
            np[index5] = (75, 0, 130)
32
            np.show()
33
            sleep(100)
34
        for index6 in range(0, 4):
35
            np.clear()
36
            np[index6] = (238, 130, 238)
37
            np.show()
38
            sleep(100)
39
        for index7 in range(0, 4):
40
            np.clear()
41
            np[index7] = (160, 32, 240)
42
            np.show()
43
            sleep(100)
44
        for index8 in range(0, 4):
45
            np.clear()
46
            np[index8] = (255, 255, 255)
47
            np.show()
48
            sleep(100)
49
50
```

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.



| 🕜 Mu 1.    | 0.3 - microbit-4 WS2812 RGB lights-2.py —   |
|------------|---|
| Mode       | Image: New Load     Image: Save     Image: Save |
| microbit-4 | WS2812 RGB lights-2.py 🔀  |
| 1          | from microbit import *  |
| 2          | import neopixel   |
| 3          | np = neopixel.NeoPixel(pin8, 4)   |
| 4          | while True:   |
| 5          | for index in range(0, 4):   |
| 6          | np.clear()  |
| 7          | np[index] = (255, 0, 0)   |
| 8          | np.show()   |
| 9          | sleep(100)  |
| 10         | <pre>for index1 in range(0, 4):</pre>   |
| 11         | np.clear()  |
| 12         | np[index1] = (255, 165, 0)  |
| 13         | np.show()   |
| 14         | sleep(100)  |
| 15         | for index2 in range(0, 4):  |
| 16         | np.clear()  |
| 17         | np[index2] = (255, 255, 0)  |
| 18         | np.show()   |
| 19         | sleep(100)  |
| 20         | for index3 in range(0, 4):  |
| 21         | np.clear()  |
| 22         | np[index3] = (0, 255, 0)  |
| 23         | np.show()   |
| 24         | sleep(100)  |



```
for index4 in range(0, 4):
25
            np.clear()
26
            np[index4] = (0, 0, 255)
27
            np.show()
28
            sleep(100)
29
        for index5 in range(0, 4):
30
            np.clear()
31
            np[index5] = (75, 0, 130)
32
            np.show()
33
            sleep(100)
34
        for index6 in range(0, 4):
35
            np.clear()
36
            np[index6] = (238, 130, 238)
37
            np.show()
38
            sleep(100)
39
        for index7 in range(0, 4):
40
            np.clear()
41
            np[index7] = (160, 32, 240)
42
            np.show()
43
            sleep(100)
44
        for index8 in range(0, 4):
45
            np.clear()
46
            np[index8] = (255, 255, 255)
47
            np.show()
48
            sleep(100)
49
50
```

#### Code3:

Enter Mu software and open the file "Project 14: WS2812 RGB LEDs.py"

to import code:

#### (How to load the project code?)

| File   | Route          | File Name |
|--------|----------------|-----------|
| Туре   |                |           |
| Python | KS4031(KS4032) | Code-3.py |



| file | folder/Python            |  |
|------|--------------------------|--|
|      | Tutorial/Python          |  |
|      | Code/Project 14 : WS2812 |  |
|      | RGB LEDs                 |  |

#### You can also input code in the editing window yourself.

#### (note:all words and symbols must be written in English)

P Mu 1.0.3 - microbit-4 WS2812 RGB lights-3.py

| crobit-4 | WS2812 RGB lights-3.py          |  |  |
|----------|---------------------------------|--|--|
| 1        | from microbit import *          |  |  |
| 2        | import neopixel                 |  |  |
| з        | np = neopixel.NeoPixel(pin8, 4) |  |  |
| 4        | from random import randint      |  |  |
| 5        | R = O                           |  |  |
| 6        | $G = \Theta$                    |  |  |
| 7        | $B = \Theta$                    |  |  |
| 8        | while True:                     |  |  |
| 9        | for index in range(0, 4):       |  |  |
| 10       | R = randint(10, 255)            |  |  |
| 11       | G = randint(10, 255)            |  |  |
| 12       | B = randint(10, 255)            |  |  |
| 13       | np.clear()                      |  |  |
| 14       | np[index] = (R, G, B)           |  |  |
| 15       | np.show()                       |  |  |
| 16       | sleep(500)                      |  |  |
| 17       | 3.4 dd d0 d2                    |  |  |

Click "Check" to examine error in the code. The program proves wrong if

N



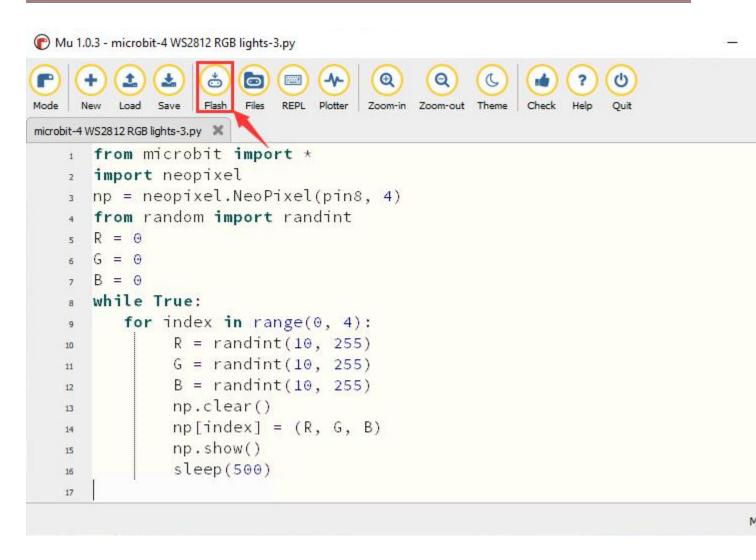
## underlines and cursors are shown.

| Mu 1.0.3 - microbit-4 WS2812 RGB lights-3.py     →                                 |
|--|
| Mode New Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check Help Quit |
| microbit-4 WS2812 RGB lights-3.py  |
| 1 from microbit import *   |
| 2 import neopixel  |
| <pre>3 np = neopixel.NeoPixel(pin8, 4)</pre>                                       |
| 4 from random import randint   |
| $s R = \Theta$   |
| 6 G = Θ  |
| 7 B = 0  |
| <pre>s while True:</pre>   |
| <pre>9 for index in range(0, 4):</pre>   |
| R = randint(10, 255)   |
| 11 G = randint(10, 255)  |
| B = randint(10, 255)   |
| <pre>13 np.clear()</pre>   |
| np[index] = (R, G, B)  |
| 15 np.show()   |
| 16 sleep(500)  |
| 17   |
|  |

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.

N





#### (4)Test Results:

Download code 1 to micro : bit, and dial POWER to ON end. All four WS2812RGB LEDs light up a different color a time cyclically.

Download code 2 to micro: bit, WS2812RGB LEDs display like flow light.

Download code 3 to micro: bit, every WS2812RGB light shows random



color one by one.

## (5)Code Explanation:

| from microbit import *                  | Import the library file of micro: bit      |
|---|--|
| import neopixel                         | Import the library file of neopixel        |
| np = neopixel.NeoPixel(pin8, 4)         | LED Set Neopixel to pin P8, and initialize |
|   | 4 LEDs                                     |
| np.clear()                              | The RGB lights on the Neopixel strip are   |
|   | all off                                    |
| while True:                             | This is a permanent loop that makes        |
|   | micro:bit execute the code of it.          |
| <b>for</b> pixel_id1 <b>in</b> range(0, | For the RGB pixels in the range of (0,     |
| len(np)):                               | len(np)), pixel_id1                        |
| <b>for</b> index <b>in</b> range(0, 4): | The RGB pixels in the range (0, 4) are     |
|   | index                                      |
| np.show()                               | Display the current pixel on the Neopixel  |
|   | strip                                      |



| np[pixel id1] = (255, 0, 0)     | Set the RGB light on the Neopixel strip to |
|---------------------------------|--|
| np[pixel_id2] = (255, 165, 0)   | pixel id1 to turn on the red light;        |
|                                 |  |
| np[pixel_id3] = (255, 255, 0)   | Set the RGB light on the Neopixel strip to |
| np[pixel_id4] = (0, 255, 0)     | pixel_id2 to turn on the orange light;     |
| np[pixel_id5] = (0, 0, 255)     | Set the RGB light on the Neopixel strip to |
| np[pixel_id6] = (75, 0, 130)    | pixel_id3 to turn on the yellow light;     |
| np[pixel_id7] = (238, 130, 238) | Set the RGB light on the Neopixel strip to |
| np[pixel_id8] = (160, 32, 240)  | pixel_id4 to turn on the green light;      |
| np[pixel_id9] = (255, 255, 255) | Set the RGB light on the Neopixel strip to |
|                                 | pixel_id5 to turn on the blue light;       |
|                                 | Set the RGB light on the Neopixel strip to |
|                                 | pixel_id6 to turn on the indigo light;     |
|                                 | Set the RGB light on the Neopixel strip to |
|                                 | pixel_id7 to turn on the violet light;     |
|                                 | Set the RGB light on the Neopixel strip to |
|                                 | pixel_id1 to turn on the purple light;     |
|                                 | Set the RGB light on the Neopixel strip to |
|                                 | pixel_id1 to turn on the white light;      |
|                                 |  |
|                                 |  |
|                                 |  |
| from random import randint      | Import randint from random variables       |



| np[pixel_id] = (R, G, B) | Set the RGB light on the Neopixel strip to pixel_id to turn on colorful light; |
|--------------------------|--|
| R = 0                    | Set the initial value of variable R to 0                                       |
| G = 0                    | Set the initial value of variable G to 0                                       |
| B = 0                    | Set the initial value of variable B to 0                                       |
| R = randint(10, 255)     | Set R=randint(10, 255)   |
| G = randint(10, 255)     | Set G=randint(10, 255)   |
| B = randint(10, 255)     | Set B=randint(10, 255)   |

#### **Project 15:Servo**



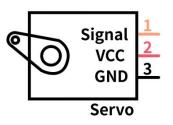
#### (1)Project Description

For those DIY smart cars, they often have the function of automatic obstacle avoidance. In the DIY process, we need a servo to control the ultrasonic module to rotate left and right, and then detect the distance between the car and the obstacle, so as to control the car to avoid the obstacle. If other microcontrollers are used to control the rotation of the servo, we need to set a certain frequency and a certain width of pulse to



control the servo angle. But if the micro:bit main board is used to control the servo angle, we only need to set the control angle in the development environment where the corresponding pulse will be automatically set to control the servo rotation. In this project, you will learn how to control the servo to rotate back and forth between 0° and 90°.

Servo motor is a position control rotary actuator. It mainly consists of housing, circuit board, core-less motor, gear and position sensor. Its working principle is that the servo receives the signal sent by MCU or receiver, and produces a reference signal with a period of 20ms and width of 1.5ms, then compares the acquired DC bias voltage to the voltage of the potentiometer and obtains the voltage difference output.



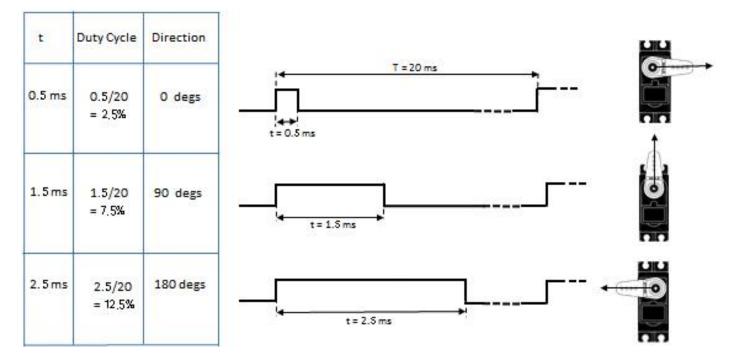
For the servo used in this project, the brown wire is the ground, the red one is the positive wire, and the orange one is the signal wire.

#### (2) Background Information of the Servo

The rotation angle of servo motor is controlled by regulating the duty cycle of PWM (Pulse-Width Modulation) signal. The standard cycle of PWM signal is 20ms (50Hz). Theoretically, the width is distributed between 1ms-2ms, but in fact, it's between 0.5ms-2.5ms. The width



corresponds to the rotation angle from 0° to 180°. But note that for different brand motor, the same signal may have different rotation angle.



After measurement, the pulse range of the servo is 0.65ms~2.5ms. For a

180 degree servo, the corresponding control relationship is as follows:

| Time on    | Angle of the | Reference Signal Cycle Time |
|------------|--------------|-----------------------------|
| High Level | Servo        | (20ms)                      |
| 0.65ms     | 0 degree     | 0.65ms high level+19.35ms   |
|            |              | low level                   |
| 1.5ms      | 90 degrees   | 1.5ms high level+18.5ms low |
|            |              | level                       |
| 2.5ms      | 180 degrees  | 2.5ms high level+17.5ms low |



| level |
|-------|
|-------|

#### (3)Parameters:

- ◆ Working voltage: DC 4.8V ~ 6V
- Operating angle range: about 180 ° (at 500  $\rightarrow$  2500 µsec)
- Pulse width range: 500  $\rightarrow$  2500 µsec
- No-load speed: 0.12 ± 0.01 sec / 60 (DC 4.8V) 0.1 ± 0.01 sec / 60 (DC 6V)
- ◆ No-load current: 200 ± 20mA (DC 4.8V) 220 ± 20mA (DC 6V)
- Stopping torque: 1.3 ± 0.01kg · cm (DC 4.8V) 1.5 ± 0.1kg · cm (DC 6V)
- Stop current:  $\leq$  850mA (DC 4.8V)  $\leq$  1000mA (DC 6V)
- Standby current:  $3 \pm 1$ mA (DC 4.8V)  $4 \pm 1$ mA (DC 6V)

It should be noted that do not use a computer for power supply, because if the current demand is greater than 500mA, the servo may be burned out. It is recommended to use an external battery for power supply.

#### (4) Experimental Preparation:

- > Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car
- > Place batteries into battery holder
- > Dial power switch to ON end



- > Connect micro:bit to computer by USB cable
- > Open the offline version of Mu.

#### (5)Test Code:

Enter Mu software and open the file "Project 15: Servo.py" to import code:

### (How to load the project code?)

| File   | Route                  | File Name         |
|--------|------------------------|-------------------|
| Туре   |                        |                   |
| Python | KS4031(KS4032)         | Project 15: Servo |
| file   | folder/Python          |                   |
|        | Tutorial/Python        |                   |
|        | Code/Project 15: Servo |                   |

You can also input code in the editing window yourself.

(note:all words and symbols must be written in English)



| ON       | Nu 1.1.0.beta.2 - Project 6: adjust the angle of a servo.py – 🛛 🛛 🛛                                       |
|----------|---|
| 2        |   |
| lode     | New Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check Tidy Help Quit                        |
| roje     | ect 6: adjust the angle of a servo.py 🗶   |
| 1        | from microbit import *  |
| 2        |   |
| 3        | class Servo:  |
| 4        | <pre>definit(self, pin, freq=50, min_us=600, max_us=2400, angle=180):</pre>                               |
| -        | <pre>self.min_us = min_us</pre>   |
| 6        | <pre>self.max_us = max_us</pre>   |
| 7        | <pre>self.us = 0</pre>  |
| 8        | <pre>self.freq = freq</pre>   |
| 9)       | <pre>self.angle = angle</pre>   |
| 10       | <pre>self.analog_period = 0</pre>   |
| 11       | self.pin = pin  |
| 12       | <pre>analog_period = round((1/self.freq) * 1000) # hertz to miliseconds</pre>                             |
| 11       | <pre>self.pin.set_analog_period(analog_period)</pre>  |
| 14       |   |
| 15.      | <pre>def write_us(self, us):</pre>  |
| 16       | us = min(self.max_us, max(self.min_us, us))   |
| 17       | duty = round(us * 1024 * <b>self</b> .freq // 1000000)  |
| 18       | <pre>self.pin.write_analog(duty) </pre>   |
| 19       | sleep(100)  |
| 20       | <pre>self.pin.write_analog(0)</pre>   |
| 21       |   |
| 22       | <pre>def write_angle(self, degrees=None):     if degrees is None;</pre>                                   |
| 23       | if degrees is None:   |
| 24       | <pre>degrees = math.degrees(radians) degrees = degrees % 360</pre>  |
| 25       | degrees = degrees % 360<br>total range = celf max us = celf min us  |
| 26       | <pre>total_range = self.max_us - self.min_us us = celf_min_us + total_range + degrees // celf_angle</pre> |
| 27       | <pre>us = self.min_us + total_range * degrees // self.angle colf write us(us)</pre>                       |
| 28       | <pre>self.write_us(us)</pre>  |
| 29       | Servo(pin8).write_angle(0)  |
| 10       | display.show(Image.HAPPY)   |
| 31       | display.snow(image.nappi)   |
| 32       | while True:   |
|          | Servo(pin8).write_angle(0)  |
| 14<br>35 | sleep(1000)   |
| 16       | Servo(pin8).write_angle(45)   |
| 37       | sleep(1000)   |
| 37       | Servo(pin8).write_angle(90)   |
|          | sleep(1000)   |
| 39       | Servo(pin8).write_angle(135)  |
|          | sleep(1000)   |
| 41       | Servo(pin8).write_angle(180)  |
| 42       | sleep(1000)   |
|          | steep(1000)   |
|          |   |

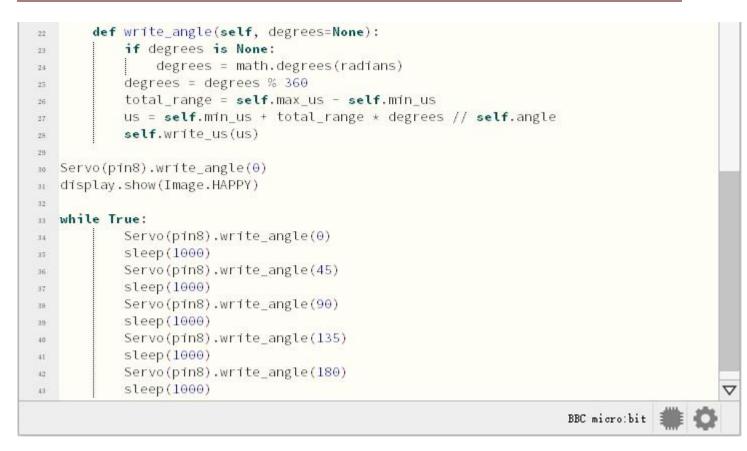


## Click "Check" to examine error in the code. The program proves wrong if

## underlines and cursors are shown.

| (P) N         | /lu 1.1.0.beta.2 - Project 6: adjust the angle of a servo.py – 🛛  | $\times$ |
|---------------|---|----------|
| Mode<br>Proje | Image: Save       Image: Save |          |
| 1             | from microbit import *  |          |
| 2             |   |          |
| 3             | class Servo:  |          |
| 4             | <pre>definit(self, pin, freq=50, min_us=600, max_us=2400, angle=180):</pre>   |          |
| 5             | <pre>self.min_us = min_us</pre>   |          |
| 6             | <pre>self.max_us = max_us</pre>   |          |
| 7             | self.us = 0   |          |
| 8             | <pre>self.freq = freq</pre>   |          |
| 9             | <pre>self.angle = angle</pre>   |          |
| 10            | <pre>self.analog_period = 0</pre>   |          |
| 11            | self.pin = pin  |          |
| 12            | analog_period = round((1/self.freq) * 1000) # hertz to miliseconds  |          |
| 13            | <pre>self.pin.set_analog_period(analog_period)</pre>  |          |
| 14            |   |          |
| 15            | <pre>def write_us(self, us):</pre>  |          |
| 16            | us = min(self.max_us, max(self.min_us, us))   |          |
| 17            | duty = round(us * 1024 * <b>self</b> .freq // 1000000)  |          |
| 18            | <pre>self.pin.write_analog(duty)</pre>  |          |
| 19            | sleep(100)  |          |
| 20            | <pre>self.pin.write_analog(0)</pre>   |          |
| 21            |   |          |





If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.



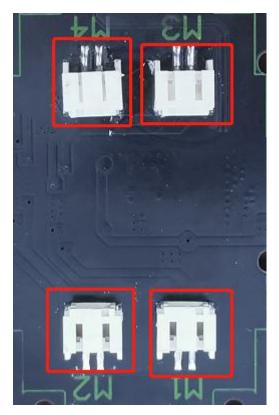
| ) N | 1u 1.1.0.beta.2 - Project 6: adjust the angle of a servo.py -   | >  | × |
|-----|---|----|---|
| )   |   |    |   |
| de  | New Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme Check Tidy Help Quit                                      |    |   |
| 20  |   |    |   |
| oje | ect 6: adjust the angle of servo.py 🗶   |    |   |
| 1   | from microbit import *  |    |   |
| 2   |   |    |   |
| 3   | class Servo:  |    | 1 |
| 4   | <pre>definit(self, pin, freq=50, min_us=600, max_us=2400, angle=180):</pre>   |    |   |
| Ŧ   | <pre>self.min_us = min_us </pre>  |    |   |
| 6   | <pre>self.max_us = max_us self.us = 0</pre>   |    |   |
| 7   |   |    |   |
| 8   | <pre>self.freq = freq calf_apgleapgle</pre>   |    | 1 |
| 9)  | <pre>self.angle = angle celf.angleg.period = 0</pre>  |    | 1 |
| 10  | <pre>self.analog_period = 0 colf pip = pip</pre>  |    |   |
| 11  | <pre>self.pin = pin applog period = round((1/celf fred) + 1000) # hertz to milicoconde</pre>                            |    | 1 |
| 12  | <pre>analog_period = round((1/self.freq) * 1000) # hertz to miliseconds self.pin.set_analog_period(analog_period)</pre> |    | 1 |
| 11  | secr.pin.sec_analog_period(analog_period)   |    |   |
| 14  | def write up (as 16 up).  |    |   |
| 15. | <pre>def write_us(self, us):</pre>  |    |   |
| 16  | <pre>us = min(self.max_us, max(self.min_us, us)) duty = round(us + 1024 + colf from (( 1000000))</pre>                  |    |   |
| 17  | <pre>duty = round(us * 1024 * self.freq // 1000000) colf pip write applog(duty)</pre>                                   |    |   |
| 18  | <pre>self.pin.write_analog(duty) sloop(100)</pre>   |    |   |
| 19  | sleep(100)  |    |   |
| 20  | <pre>self.pin.write_analog(0)</pre>   |    |   |
| 21  |   |    |   |
| 22  | <pre>def write_angle(self, degrees=None):</pre>   |    |   |
| 23  | if degrees is None:   |    |   |
| 24  | degrees = math.degrees(radians)   |    |   |
| 25  | degrees = degrees % 360   |    |   |
| 26  | total_range = <b>self.</b> max_us - <b>self.</b> min_us   |    |   |
| 27  | us = <b>self</b> .min_us + total_range * degrees // <b>self</b> .angle  |    |   |
| 28  | <pre>self.write_us(us)</pre>  |    |   |
| 29  |   |    |   |
| 10  | Servo(pin8),write_angle(0)  |    |   |
| 31  | display.show(Image.HAPPY)   |    |   |
| 32  |   |    |   |
| 33  | while True:   |    |   |
| 34  | Servo(pin8).write_angle(0)  |    |   |
| 35  | sleep(1000)   |    |   |
| 16  | Servo(pin8).write_angle(45)   |    |   |
| 37  | sleep(1000)   |    |   |
| 18  | Servo(pin8).write_angle(90)   |    |   |
| 39  | sleep(1000)   |    |   |
| 40  | Servo(pin8).write_angle(135)  |    |   |
| 41  | sleep(1000)   |    |   |
| 42  | Servo(pin8).write_angle(180)  |    |   |
| 43  | sleep(1000)   |    |   |
|     | BBC micro:bit   | hi | į |



### (6)Test Results:

After uploading the test code, dialing POWER switch to ON end and powering it by external power, the LED dot matrix shows a smiley pattern and the servo rotates in the pattern 0°~45°~90°~135°~180°~0°.

#### **Project 16:Motor**



(1)Project Description

The Keyestudio 4WD Mecanum Robot Car is equipped with 4 DC reduction motors, also called gear reduction motor, which is developed on the



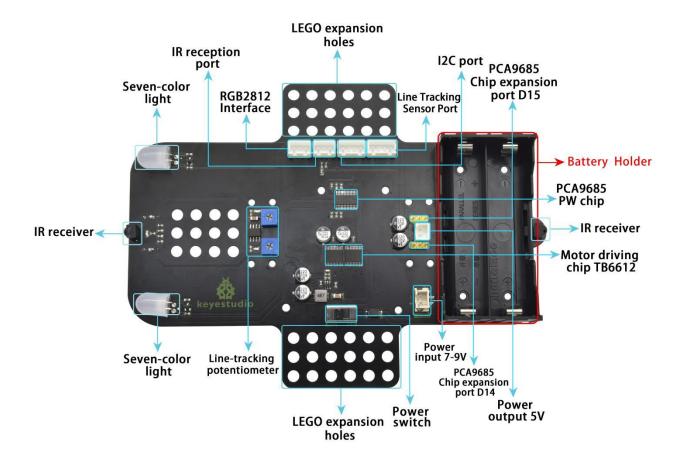
ordinary DC motor. It has a matching gear reduction box which provides a lower speed but a larger torque. Furthermore, different reduction ratios of the box can provide different speeds and torques.

Gear motor is the integration of gearmotor and motor, which is applied widely in steel and machine industry

Micro:bit motor driver shield comes with PCA9685PW and TB6612FNG chip. In order to save the IO port resource, we control the rotation direction and speed of two DC gear motors with TB6612FNG chip.

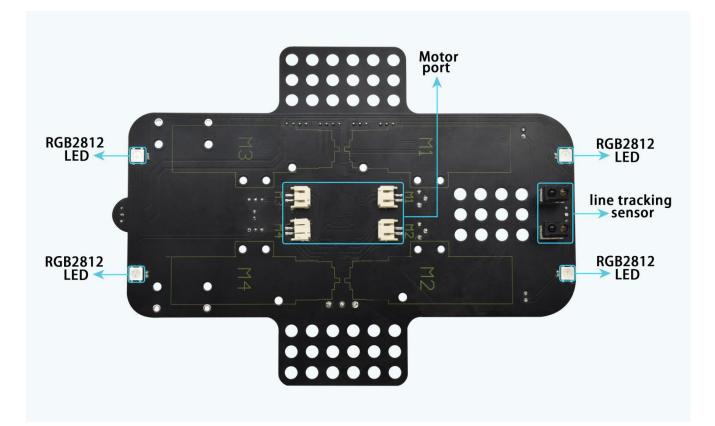
#### **Details about chips:**





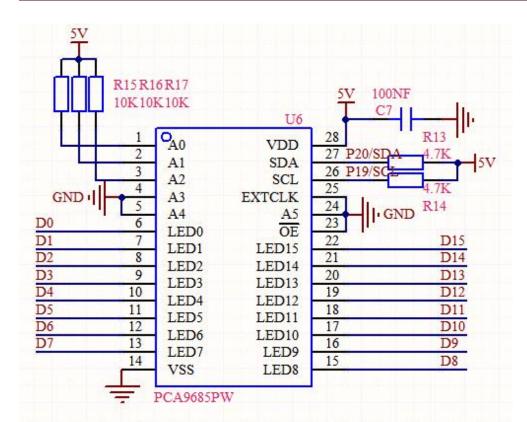
Front





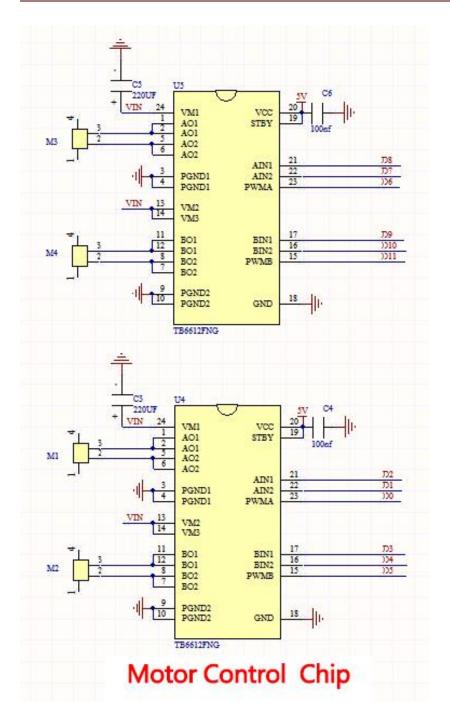
Back





# PCA9685PW Module





## (2) Experimental Preparation:

- > Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car
- Place batteries into battery holder
- > Dial power switch to ON end
- > Connect micro:bit to computer by USB cable



> Open the offline version of Mu.

## (3)Test Code:

Enter Mu software and open the file "Project 16: Motor.py" to import code:

(How to load the project code?)

| File   | Route                  | File Name |
|--------|------------------------|-----------|
| Туре   |                        |           |
| Python | KS4031(KS4032)         | Code-1.py |
| file   | folder/Python          |           |
|        | Tutorial/Python        |           |
|        | Code/Project 16: Motor |           |

You can also input code in the editing window yourself.

(note:all words and symbols must be written in English)



| (      |   |    |
|--------|---|----|
| 🕐 Mu   | 1.1.0.beta.5 - microbit-Motor Driving-1.py                      |    |
|        |   | 8  |
| Mode   | New Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme C | he |
| microb | bit-Motor Driving-1.py 🔀  |    |
| 1      | <pre>from microbit import *</pre>                               |    |
| 2      | <pre>from keyes_mecanum_Car import *</pre>                      |    |
| 3      | <pre>mecanumCar = Mecanum_Car_Driver()</pre>                    |    |
| 4      | while True:   |    |
| 5      | display.show(Image.ARROW_S)                                     |    |
| 6      | mecanumCar.Motor_Upper_L(1, 100)                                |    |
| 7      | <pre>mecanumCar.Motor_Lower_L(1, 100)</pre>                     |    |
| 8      | mecanumCar.Motor_Upper_R(1, 100)                                |    |
| 9      | <pre>mecanumCar.Motor_Lower_R(1, 100)</pre>                     |    |
| 10     | sleep(1000)   |    |
| 11     | display.show(Image.ARROW_N)                                     |    |
| 12     | mecanumCar.Motor_Upper_L(0, 100)                                |    |
| 13     | mecanumCar.Motor_Lower_L(0, 100)                                |    |
| 14     | mecanumCar.Motor_Upper_R(0, 100)                                |    |
| 15     | <pre>mecanumCar.Motor_Lower_R(0, 100)</pre>                     |    |
| 16     | sleep(1000)   |    |
| 17     | display.show(Image.ARROW_E)                                     |    |
| 18     | <pre>mecanumCar.Motor_Upper_L(0, 100)</pre>                     |    |
| 19     | mecanumCar.Motor_Lower_L(0, 100)                                |    |
| 20     | <pre>mecanumCar.Motor_Upper_R(1, 100)</pre>                     |    |
| 21     | mecanumCar.Motor_Lower_R(1, 100)                                |    |
| 22     | sleep(1000)   |    |
| 23     | display.show(Image.ARROW_W)                                     |    |
| 24     | <pre>mecanumCar.Motor_Upper_L(1, 100)</pre>                     |    |
| 25     | <pre>mecanumCar.Motor_Lower_L(1, 100)</pre>                     |    |
| 26     | <pre>mecanumCar.Motor_Upper_R(0, 100)</pre>                     |    |
| 27     | <pre>mecanumCar.Motor_Lower_R(0, 100)</pre>                     |    |
| 28     | sleep(1000)   |    |
| 29     | display.show(Image("00900:""09990:""99999:""99999:""09090"))    |    |
| 30     | <pre>mecanumCar.Motor_Upper_L(0, 0)</pre>                       |    |
| 31     | <pre>mecanumCar.Motor_Lower_L(0, 0)</pre>                       |    |
| 32     | <pre>mecanumCar.Motor_Upper_R(0, 0)</pre>                       |    |
|        | mecanum(ar Motor Lower R(0 0)                                   |    |
|        | BI  | BC |
|        |   |    |



Click "Files" to import "keyes\_mecanum\_Car.py "library file tomicro:bit (<u>How</u> <u>to import files?</u>). No need to do it again if you have imported it before. Tap "Check" button to confirm if the code has errors. The program proves wrong if there are underlines and cursors



| 🕜 Mu   | 1.1.0.beta.5 - microbit-Motor Driving-1.py                      |    |
|--------|---|----|
| ſ      |   | 1  |
| Mode   | New Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme C | he |
| microb | bit-Motor Driving-1.py 💥  |    |
| 1      | <pre>from microbit import *</pre>                               |    |
| 2      | <pre>from keyes_mecanum_Car import *</pre>                      |    |
| 3      | <pre>mecanumCar = Mecanum_Car_Driver()</pre>                    |    |
| 4      | while True:   |    |
| 5      | display.show(Image.ARROW_S)                                     |    |
| 6      | <pre>mecanumCar.Motor_Upper_L(1, 100)</pre>                     |    |
| 7      | <pre>mecanumCar.Motor_Lower_L(1, 100)</pre>                     |    |
| 8      | <pre>mecanumCar.Motor_Upper_R(1, 100)</pre>                     |    |
| 9      | <pre>mecanumCar.Motor_Lower_R(1, 100)</pre>                     |    |
| 10     | sleep(1000)   |    |
| 11     | display.show(Image.ARROW_N)                                     |    |
| 12     | <pre>mecanumCar.Motor_Upper_L(0, 100)</pre>                     |    |
| 13     | mecanumCar.Motor_Lower_L(0, 100)                                |    |
| 14     | mecanumCar.Motor_Upper_R(0, 100)                                |    |
| 15     | <pre>mecanumCar.Motor_Lower_R(0, 100)</pre>                     |    |
| 15     | sleep(1000)   |    |
| 17     | display.show(Image.ARROW_E)                                     |    |
| 18     | <pre>mecanumCar.Motor_Upper_L(0, 100)</pre>                     |    |
| 19     | <pre>mecanumCar.Motor_Lower_L(0, 100)</pre>                     |    |
| 20     | <pre>mecanumCar.Motor_Upper_R(1, 100)</pre>                     |    |
| 21     | mecanumCar.Motor_Lower_R(1, 100)                                |    |
| 22     | sleep(1000)   |    |
| 23     | display.show(Image.ARROW_W)                                     |    |
| 24     | <pre>mecanumCar.Motor_Upper_L(1, 100)</pre>                     |    |
| 25     | <pre>mecanumCar.Motor_Lower_L(1, 100)</pre>                     |    |
| 26     | <pre>mecanumCar.Motor_Upper_R(0, 100)</pre>                     |    |
| 27     | <pre>mecanumCar.Motor_Lower_R(0, 100)</pre>                     |    |
| 28     | sleep(1000)   |    |
| 29     | display.show(Image("00900:""09990:""99999:""99999:""09090"))    |    |
| 30     | <pre>mecanumCar.Motor_Upper_L(0, 0)</pre>                       |    |
| 31     | <pre>mecanumCar.Motor_Lower_L(0, 0)</pre>                       |    |
| 32     | <pre>mecanumCar.Motor_Upper_R(0, 0)</pre>                       |    |
|        | mecanumCar Motor Lower R(0 0)                                   |    |
|        | BB  | С  |
|        |   |    |

If the code is correct, connect micro:bit to computer and click "Flash" to



download code to micro:bit board.

| <pre>microbit=Motor Briving=1.py  from microbit import * from keyes_mecanum_Car import * mecanumCar = Mecanum_Car_Driver() while True:     display.show(Image.ARROW_S)     mecanumCar.Motor_Upper_L(1, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Lower_R(1, 100)     mecanumCar.Motor_Lower_R(1, 100)     mecanumCar.Motor_Lower_R(1, 100)     mecanumCar.Motor_Upper_L(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Lower_R(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Upper_R(1, 100)     sleep(1000)     display.show(Image.ARROW_W)     mecanumCar.Motor_Upper_R(0, 100)     sleep(1000)     display.show(Image.ARROW_W)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(0, 0)     mecanumCar.Motor_Upper_R(0, 0) </pre>   | <pre>Mode New Load Save Flash Files REFL Plotter Zoomrin Zoomrout Theme microbit=Moter Driving1.py W  i from microbit import * i from keyes_mecanum_Car import * mecanumCar = Mecanum_Car_Driver() while True:     display.show(Image.ARROW_S)     mecanumCar.Motor_Upper_L(1, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Lower_R(1, 100)     mecanumCar.Motor_Lower_R(1, 100)     mecanumCar.Motor_Upper_L(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Lower_R(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Upper_R(0, 0) </pre>   | ) Mu | 1.1.0.beta.5 - microbit-Motor Driving-1.py                   |    |
|--|--|------|--|----|
| <pre>microbit=Motor Briving=1.py  i from microbit import * i from keyes_mecanum_Car import * mecanumCar = Mecanum_Car_Driver() while True:     display.show(Image.ARROW_S)     mecanumCar.Motor_Upper_L(1, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Lower_R(1, 100)     mecanumCar.Motor_Lower_R(1, 100)     mecanumCar.Motor_Upper_L(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Lower_R(0, 100)     mecanumCar.Motor_Lower_R(0, 100)     mecanumCar.Motor_Lower_R(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Upper_R(1, 100)     sleep(1000)     display.show(Image.ARROW_W)     mecanumCar.Motor_Upper_R(0, 100)     sleep(1000)     display.show(Image.ARROW_W)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(0, 0)     mecanumCar.Motor_Upper_R(0, 0) </pre>  | <pre>microbitHotor Driving:Ly #  from microbit import * from keyes_mecanum_Car import * mecanumCar = Mecanum_Car_Driver() while True:     display.show(Image.ARROW_S)     mecanumCar.Motor_Upper_L(1, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Lower_R(1, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Upper_L(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_L(0, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Upper_L(1, 100)     mecanumCar.Motor_Upper_R(0, 0)     mecanumCar.Motor_Upper_R(0, 0) </pre>   | Mode |  | Ch |
| <pre>from microbit import * from keyes_mecanum_Car import * mecanumCar = Mecanum_Car_Driver() while True: display.show(Image.ARROW_S) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Lower_R(1, 100) listep(1000) display.show(Image.ARROW_N) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) listep(1000) display.show(Image.ARROW_E) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_L(0, 100) listep(1000) display.show(Image.ARROW_E) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) listep(1000) display.show(Image.ARROW_W) mecanumCar.Motor_Upper_R(1, 100) listep(1000) listep(10000) listep(10000) listep(1000000:""99999:""99999:""99999:""99999:""99999:""99999:""99999:""99999:""99999:""99999:""99999:""99999:""99999:""99999:""99999:""99999:""99990:""99999:"</pre> | <pre>from microbit import * from keyes_mecanum_Car import * mecanumCar = Mecanum_Car_Driver() while True: display.show(Image.ARROW_S) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Lower_R(1, 100) display.show(Image.ARROW_N) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Motor</pre>   |      |  |    |
| <pre>from keyes_mecanum_Car import * mecanumCar = Mecanum_Car_Driver() while True:     display.show(Image.ARROW_S)     mecanumCar.Motor_Upper_L(1, 100)     mecanumCar.Motor_Lower_L(1, 100)     mecanumCar.Motor_Lower_R(1, 100)     mecanumCar.Motor_Lower_R(1, 100)     sleep(1000)     display.show(Image.ARROW_N)     mecanumCar.Motor_Upper_L(0, 100)     mecanumCar.Motor_Lower_R(0, 100)     mecanumCar.Motor_Lower_R(0, 100)     mecanumCar.Motor_Lower_R(0, 100)     mecanumCar.Motor_Lower_R(0, 100)     mecanumCar.Motor_Lower_R(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_L(0, 100)     mecanumCar.Motor_Upper_L(0, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Lower_R(1, 100)     sleep(1000)     display.show(Image.ARROW_W)     mecanumCar.Motor_Lower_R(1, 100)     mecanumCar.Motor_Lower_R(0, 100)     mecanumCar.Motor_Lower_R(0, 100)     sleep(1000)     display.show(Image.ARROW_W)     mecanumCar.Motor_Lower_R(0, 100)     mecanumCar.Motor_Upper_L(1, 100)     mecanumCar.Motor_Upper_L(1, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Lower_R(0, 100)     mecanumCar.Motor_Upper_R(0, 0)     display.show(Image("00900:""099990:""99999:""99999:""09090"))     mecanumCar.Motor_Upper_L(0, 0)     mecanumCar.Motor_Upper_L(0, 0) </pre>   | <pre>from keyes_mecanum_Car import * mecanumCar = Mecanum_Car_Driver() while True: display.show(Image.ARROW_S) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000) display.show(Image.ARROW_N) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Upper_R(0, 100) display.show(Image.ARROW_E) mecanumCar.Motor_Upper_L(0, 100) display.show(Image.ARROW_E) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_L(0, 100) display.show(Image.ARROW_E) mecanumCar.Motor_Upper_R(1, 100) display.show(Image.ARROW_M) mecanumCar.Motor_Upper_R(1, 100) display.show(Image.ARROW_W) mecanumCar.Motor_Upper_L(1, 100) display.show(Image.ARROW_W) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_R(0, 0) mec</pre>   |      |  |    |
| <pre>mecanumCar = Mecanum_Car_Driver() while True: display.show(Image.ARROW_S) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000) display.show(Image.ARROW_N) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) sleep(1000) display.show(Image.ARROW_E) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Lower_R(0, 100) display.show(Image.ARROW_W) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Lower_R(0, 0) display.show(Image("00900:""09990:""99999:""99999:""09090")) mecanumCar.Motor_Lower_R(0, 0) display.show(Image.</pre> | <pre>mecanumCar = Mecanum_Car_Driver() while True:     display.show(Image.ARROW_S)     mecanumCar.Motor_Upper_L(1, 100)     mecanumCar.Motor_Lower_L(1, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Lower_R(1, 100)     sleep(1000)     display.show(Image.ARROW_N)     mecanumCar.Motor_Upper_L(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Lower_R(0, 100)     mecanumCar.Motor_Lower_R(0, 100)     mecanumCar.Motor_Upper_L(0, 100)     mecanumCar.Motor_Upper_L(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Upper_R(1, 100)     mecanumCar.Motor_Lower_R(1, 100)     mecanumCar.Motor_Upper_L(1, 100)     mecanumCar.Motor_Upper_L(1, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_L(1, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_R(0, 100)     mecanumCar.Motor_Upper_L(0, 0)     mecanumCar.Motor_Lower_R(0, 100)     mecanumCar.Motor_Upper_R(0, 0)     mecanumCar.Motor_Upper_R(0, 0)     mecanumCar.Motor_Upper_L(0, 0)     mecanumCar.Motor_Upper_R(0, 0) </pre>  | 2    |  |    |
| <pre>while True:<br/>display.show(Image.ARROW_S)<br/>mecanumCar.Motor_Upper_L(1, 100)<br/>mecanumCar.Motor_Upper_R(1, 100)<br/>mecanumCar.Motor_Upper_R(1, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_N)<br/>mecanumCar.Motor_Upper_L(0, 100)<br/>mecanumCar.Motor_Lower_R(0, 100)<br/>mecanumCar.Motor_Lower_R(0, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_E)<br/>mecanumCar.Motor_Upper_L(0, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_E)<br/>mecanumCar.Motor_Upper_R(1, 100)<br/>mecanumCar.Motor_Lower_R(1, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_W)<br/>mecanumCar.Motor_Lower_R(1, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_W)<br/>mecanumCar.Motor_Upper_L(1, 100)<br/>mecanumCar.Motor_Upper_R(1, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_W)<br/>mecanumCar.Motor_Lower_R(1, 100)<br/>sleep(1000)<br/>display.show(Image.MROW_W)<br/>mecanumCar.Motor_Upper_R(0, 100)<br/>sleep(1000)<br/>display.show(Image("00900:""09990:""99999:""99999:""09090"))<br/>mecanumCar.Motor_Upper_L(0, 0)<br/>mecanumCar.Motor_Upper_R(0, 0)</pre>   | <pre>while True:<br/>display.show(Image.ARROW_S)<br/>mecanumCar.Motor_Upper_L(1, 100)<br/>mecanumCar.Motor_Lower_L(1, 100)<br/>mecanumCar.Motor_Upper_R(1, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_N)<br/>mecanumCar.Motor_Upper_L(0, 100)<br/>mecanumCar.Motor_Upper_R(0, 100)<br/>mecanumCar.Motor_Lower_R(0, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_E)<br/>mecanumCar.Motor_Upper_L(0, 100)<br/>mecanumCar.Motor_Upper_L(0, 100)<br/>mecanumCar.Motor_Upper_R(1, 100)<br/>mecanumCar.Motor_Upper_R(1, 100)<br/>mecanumCar.Motor_Upper_R(1, 100)<br/>display.show(Image.ARROW_W)<br/>mecanumCar.Motor_Upper_L(1, 100)<br/>mecanumCar.Motor_Upper_R(1, 100)<br/>display.show(Image.ARROW_W)<br/>mecanumCar.Motor_Upper_L(1, 100)<br/>mecanumCar.Motor_Upper_R(0, 100)<br/>display.show(Image.ARROW_W)<br/>display.show(Image("00900:""09990:""99999:""99999:""09090"))<br/>mecanumCar.Motor_Upper_L(0, 0)<br/>mecanumCar.Motor_Upper_R(0, 0)<br/>mecanumCar</pre> |      |  |    |
| <pre>mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) display.show(Image.ARROW_N) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Upper_R(0, 0) </pre>  | <pre>mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000) display.show(Image.ARROW_N) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Lower_R(0, 0) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Lower_R(0, 0) mecanumCar.Motor_Lower_R(</pre>   | 4    |  |    |
| <pre>mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000) display.show(Image.ARROW_N) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) sleep(1000) display.show(Image.ARROW_E) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Motor_Upper_R(0, 0) </pre>  | <pre>mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Mo</pre>   | 5    | display.show(Image.ARROW_S)                                  |    |
| <pre>mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000) display.show(Image.ARROW_N) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) sleep(1000) display.show(Image.ARROW_E) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Motor_Upper_R(0, 0) </pre>  | <pre>mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Mo</pre>   | 6    |  |    |
| <pre>mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000) display.show(Image.ARROW_N) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Lower_L(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Upper_L(1, 100) sleep(1000) display.show(Image.ARROW_W) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Upper_R(0, 0) </pre>   | <pre>mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Moto</pre>   |      |  |    |
| <pre>mecanumCar.Motor_Lower_R(1, 100) sleep(1000) display.show(Image.ARROW_N) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Lower_L(0, 100) mecanumCar.Motor_Lower_R(0, 100) sleep(1000) display.show(Image.ARROW_E) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Lower_L(0, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) limecanumCar.Motor_Lower_R(1, 100) limecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Lower_R(0, 100) limecanumCar.Motor_Lower_R(0, 100) limecanumCar.Motor_Lower_R(1, 100) limecanumCar.Motor_Upper_R(1, 100) limecanumCar.Motor_Lower_R(0, 100) limecanumCar.Motor_Lower_R(0, 100) limecanumCar.Motor_Lower_R(0, 100) limecanumCar.Motor_Lower_R(0, 100) limecanumCar.Motor_Upper_R(0, 100) limecanumCar.Motor_Upper_R(0, 100) limecanumCar.Motor_Upper_R(0, 100) limecanumCar.Motor_Upper_R(0, 100) limecanumCar.Motor_Upper_R(0, 0) limecanumCar.Motor_Upper_R(0, 0) limecanumCar.Motor_Upper_R(0, 0) limecanumCar.Motor_Upper_R(0, 0) limecanumCar.Motor_Lower_R(0, 0) limecanumCar.Motor_Lower_R(0, 0) limecanumCar.Motor_Lower_R(0, 0) limecanumCar.Motor_Lower_R(0, 0) limecanumCar.Motor_Upper_R(0, 0) limecanumCar.Motor_Lower_R(0, 0) limecanumCar.Motor_Lo</pre> | <pre>mecanumCar.Motor_Lower_R(1, 100) sleep(1000) display.show(Image.ARROW_N) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Lower_L(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000) display.show(Image.ARROW_W) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Motor_Lower_R(0, 0) m</pre>   | s    |  |    |
| <pre>sleep(1000)<br/>display.show(Image.ARROW_N)<br/>mecanumCar.Motor_Upper_L(0, 100)<br/>mecanumCar.Motor_Lower_L(0, 100)<br/>mecanumCar.Motor_Upper_R(0, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_E)<br/>mecanumCar.Motor_Upper_L(0, 100)<br/>mecanumCar.Motor_Lower_L(0, 100)<br/>mecanumCar.Motor_Lower_R(1, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_W)<br/>display.show(Image.ARROW_W)<br/>mecanumCar.Motor_Lower_R(1, 100)<br/>mecanumCar.Motor_Upper_L(1, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_W)<br/>mecanumCar.Motor_Lower_L(1, 100)<br/>mecanumCar.Motor_Lower_R(0, 100)<br/>sleep(1000)<br/>display.show(Image("00900:""09990:""99999:""99999:""09090"))<br/>mecanumCar.Motor_Upper_L(0, 0)<br/>mecanumCar.Motor_Lower_L(0, 0)<br/>mecanumCar.Motor_Lower_L(0, 0)<br/>mecanumCar.Motor_Upper_R(0, 0)</pre>  | <pre>sleep(1000) display.show(Image.ARROW_N) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Lower_L(0, 100) mecanumCar.Motor_Upper_R(0, 100) sleep(1000) display.show(Image.ARROW_E) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000) display.show(Image.ARROW_W) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_R(0, 100) sleep(1000) display.show(Image.ARROW_W) mecanumCar.Motor_Lower_R(0, 100) sleep(1000) display.show(Image.ARROW_W) mecanumCar.Motor_Upper_R(0, 100) sleep(1000) display.show(Image("00900:""09990:""99999:""09090")) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Upper_R(0, 0) </pre>  | 9    |  |    |
| <pre>mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Lower_L(0, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Lower_L(0, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000) display.show(Image.ARROW_W) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Lower_R(0, 0) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Lower_R(0, 0) </pre>  | <pre>mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Lower_L(0, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000) sleep(1000) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Lower_R(0, 0) mecanumCar.Motor_Lower_R(0, 0) mecanumCar.Motor_Lower_R(0, 0) mecanumCar.Motor_Lower_R(0, 0) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Motor_Lower_R(0, 0) mecanumCar.Motor_Lower_R(0,</pre>  | 10   |  |    |
| <pre>mecanumCar.Motor_Lower_L(0, 100)<br/>mecanumCar.Motor_Upper_R(0, 100)<br/>mecanumCar.Motor_Lower_R(0, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_E)<br/>mecanumCar.Motor_Upper_L(0, 100)<br/>mecanumCar.Motor_Lower_L(0, 100)<br/>mecanumCar.Motor_Lower_R(1, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_W)<br/>display.show(Image.ARROW_W)<br/>mecanumCar.Motor_Upper_L(1, 100)<br/>mecanumCar.Motor_Lower_L(1, 100)<br/>mecanumCar.Motor_Lower_R(0, 100)<br/>sleep(1000)<br/>display.show(Image("00900:""09990:""99999:""09090"))<br/>mecanumCar.Motor_Upper_L(0, 0)<br/>mecanumCar.Motor_Lower_L(0, 0)<br/>mecanumCar.Motor_Lower_L(0, 0)<br/>mecanumCar.Motor_Lower_R(0, 0)</pre>   | <pre>mecanumCar.Motor_Lower_L(0, 100)<br/>mecanumCar.Motor_Upper_R(0, 100)<br/>mecanumCar.Motor_Lower_R(0, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_E)<br/>mecanumCar.Motor_Upper_L(0, 100)<br/>mecanumCar.Motor_Lower_L(0, 100)<br/>mecanumCar.Motor_Lower_R(1, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_W)<br/>mecanumCar.Motor_Upper_L(1, 100)<br/>mecanumCar.Motor_Lower_L(1, 100)<br/>mecanumCar.Motor_Lower_L(1, 100)<br/>mecanumCar.Motor_Lower_R(0, 100)<br/>sleep(1000)<br/>display.show(Image("00900:""09990:""99999:""09090"))<br/>mecanumCar.Motor_Lower_R(0, 0)<br/>mecanumCar.Motor_Lower_L(0, 0)<br/>mecanumCar.Motor_Lower_R(0, 0)<br/>mecanumCar.Motor_Upper_R(0, 0)<br/>mecanumCar.Motor_Upper_R(0, 0)<br/>mecanumCar.Motor_Upper_R(0, 0)<br/>mecanumCar.Motor_Lower_R(0, 0)<br/>mecanumCar.Motor_Lower_R(0, 0)<br/>mecanumCar.Motor_Lower_R(0, 0)<br/>mecanumCar.Motor_Lower_R(0, 0)<br/>mecanumCar.Motor_Lower_R(0, 0)<br/>mecanumCar.Motor_Lower_R(0, 0)</pre>  | 11   |  |    |
| <pre>mecanumCar.Motor_Lower_L(0, 100)<br/>mecanumCar.Motor_Upper_R(0, 100)<br/>mecanumCar.Motor_Lower_R(0, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_E)<br/>mecanumCar.Motor_Upper_L(0, 100)<br/>mecanumCar.Motor_Lower_L(0, 100)<br/>mecanumCar.Motor_Lower_R(1, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_W)<br/>display.show(Image.ARROW_W)<br/>mecanumCar.Motor_Upper_L(1, 100)<br/>mecanumCar.Motor_Lower_L(1, 100)<br/>mecanumCar.Motor_Lower_R(0, 100)<br/>sleep(1000)<br/>display.show(Image("00900:""09990:""99999:""09090"))<br/>mecanumCar.Motor_Upper_L(0, 0)<br/>mecanumCar.Motor_Lower_L(0, 0)<br/>mecanumCar.Motor_Lower_L(0, 0)<br/>mecanumCar.Motor_Lower_R(0, 0)</pre>   | <pre>mecanumCar.Motor_Lower_L(0, 100)<br/>mecanumCar.Motor_Upper_R(0, 100)<br/>mecanumCar.Motor_Lower_R(0, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_E)<br/>mecanumCar.Motor_Upper_L(0, 100)<br/>mecanumCar.Motor_Lower_L(0, 100)<br/>mecanumCar.Motor_Lower_R(1, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_W)<br/>mecanumCar.Motor_Upper_L(1, 100)<br/>mecanumCar.Motor_Lower_L(1, 100)<br/>mecanumCar.Motor_Lower_L(1, 100)<br/>mecanumCar.Motor_Lower_R(0, 100)<br/>sleep(1000)<br/>display.show(Image("00900:""09990:""99999:""09090"))<br/>mecanumCar.Motor_Lower_R(0, 0)<br/>mecanumCar.Motor_Lower_L(0, 0)<br/>mecanumCar.Motor_Lower_R(0, 0)<br/>mecanumCar.Motor_Upper_R(0, 0)<br/>mecanumCar.Motor_Upper_R(0, 0)<br/>mecanumCar.Motor_Upper_R(0, 0)<br/>mecanumCar.Motor_Lower_R(0, 0)<br/>mecanumCar.Motor_Lower_R(0, 0)<br/>mecanumCar.Motor_Lower_R(0, 0)<br/>mecanumCar.Motor_Lower_R(0, 0)<br/>mecanumCar.Motor_Lower_R(0, 0)<br/>mecanumCar.Motor_Lower_R(0, 0)</pre>  | 12   | mecanumCar.Motor_Upper_L(0, 100)                             |    |
| <pre>mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) sleep(1000) display.show(Image.ARROW_E) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000) display.show(Image.ARROW_W) display.show(Image.ARROW_W) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Upper_R(0, 100) sleep(1000) display.show(Image("00900:""09990:""99999:""99999:""09090")) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Upper_R(0, 0)</pre>  | <pre>mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Lower_L(0, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Lower_R(0, 0) mecanumCar.Motor_Lower_R(0, 0) mecanumCar.Motor_Lower_R(0, 0) mecanumCar.Motor_Lower_R(0, 0) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Motor_Lower_R(0, 0) me</pre>   | 13   |  |    |
| <pre>mecanumCar.Motor_Lower_R(0, 100) sleep(1000) display.show(Image.ARROW_E) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Lower_L(0, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000) display.show(Image.ARROW_W) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Lower_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) sleep(1000) display.show(Image("00900:""09990:""99999:""99999:""09090")) mecanumCar.Motor_Lower_L(0, 0) mecanumCar.Motor_Lower_L(0, 0) mecanumCar.Motor_Lower_L(0, 0)</pre>   | <pre>mecanumCar.Motor_Lower_R(0, 100) sleep(1000) display.show(Image.ARROW_E) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000) display.show(Image.ARROW_W) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) sleep(1000) display.show(Image("00900:""09990:""99999:""99999:""09090")) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Upper_R(0, 0) </pre>   | 14   |  |    |
| <pre>sleep(1000)<br/>display.show(Image.ARROW_E)<br/>mecanumCar.Motor_Upper_L(0, 100)<br/>mecanumCar.Motor_Lower_L(0, 100)<br/>mecanumCar.Motor_Upper_R(1, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_W)<br/>mecanumCar.Motor_Upper_L(1, 100)<br/>mecanumCar.Motor_Lower_L(1, 100)<br/>mecanumCar.Motor_Lower_R(0, 100)<br/>sleep(1000)<br/>display.show(Image("00900:""09990:""99999:""09090"))<br/>mecanumCar.Motor_Upper_L(0, 0)<br/>mecanumCar.Motor_Lower_L(0, 0)<br/>mecanumCar.Motor_Upper_R(0, 0)</pre>   | <pre>sleep(1000)<br/>display.show(Image.ARROW_E)<br/>mecanumCar.Motor_Upper_L(0, 100)<br/>mecanumCar.Motor_Lower_L(0, 100)<br/>mecanumCar.Motor_Upper_R(1, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_W)<br/>mecanumCar.Motor_Upper_L(1, 100)<br/>mecanumCar.Motor_Lower_L(1, 100)<br/>mecanumCar.Motor_Lower_R(0, 100)<br/>sleep(1000)<br/>display.show(Image("00900:""09990:""99999:""99999:""09090"))<br/>mecanumCar.Motor_Upper_L(0, 0)<br/>sleep(1000)<br/>sleep(1000)<br/>mecanumCar.Motor_Lower_L(0, 0)<br/>mecanumCar.Motor_Lower_L(0, 0)<br/>mecanumCar.Motor_Lower_R(0, 0)<br/>mecanumCar.Motor_Upper_R(0, 0)<br/>mecanumCar.Motor_Upper_R(0, 0)<br/>mecanumCar.Motor_Upper_R(0, 0)</pre>   | 15   |  |    |
| <pre>mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Lower_L(0, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000) display.show(Image.ARROW_W) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(0, 100) sleep(1000) display.show(Image("00900:""09990:""99999:""99999:""09090")) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Lower_L(0, 0) mecanumCar.Motor_Upper_R(0, 0)</pre>  | <pre>mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Lower_L(0, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000) display.show(Image.ARROW_W) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Lower_R(0, 100) sleep(1000) display.show(Image("00900:""09990:""99999:""99999:""09090")) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Lower_L(0, 0) mecanumCar.Motor_Lower_R(0, 0) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Motor_Upper_R(0, 0)</pre>  | 16   |  |    |
| <pre>mecanumCar.Motor_Lower_L(0, 100)<br/>mecanumCar.Motor_Upper_R(1, 100)<br/>mecanumCar.Motor_Lower_R(1, 100)<br/>sleep(1000)<br/>display.show(Image.ARROW_W)<br/>mecanumCar.Motor_Upper_L(1, 100)<br/>mecanumCar.Motor_Lower_L(1, 100)<br/>mecanumCar.Motor_Lower_R(0, 100)<br/>sleep(1000)<br/>display.show(Image("00900:""09990:""99999:""99999:""09090"))<br/>mecanumCar.Motor_Upper_L(0, 0)<br/>mecanumCar.Motor_Lower_L(0, 0)<br/>mecanumCar.Motor_Lower_L(0, 0)<br/>mecanumCar.Motor_Upper_R(0, 0)</pre>  | <pre>mecanumCar.Motor_Lower_L(0, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000) sleep(1000) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Lower_R(0, 100) sleep(1000) sleep(1000) sleep(1000) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Lower_L(0, 0) mecanumCar.Motor_Lower_R(0, 0) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Motor_Lower_R(0, 0) mecanumCar.Motor_Upper_R(0, 0)</pre>   | 17   | display.show(Image.ARROW_E)                                  |    |
| <pre>mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000) display.show(Image.ARROW_W) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(0, 100) sleep(1000) display.show(Image("00900:""09990:""99999:""99999:""09090")) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Lower_L(0, 0) mecanumCar.Motor_Lower_L(0, 0) mecanumCar.Motor_Upper_R(0, 0)</pre>   | <pre>mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000) display.show(Image.ARROW_W) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(0, 100) sleep(1000) display.show(Image("00900:""09990:""99999:""99999:""09090")) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Lower_L(0, 0) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Motor_Upper_R(0, 0)</pre>   | 18   | <pre>mecanumCar.Motor_Upper_L(0, 100)</pre>                  |    |
| <pre>mecanumCar.Motor_Lower_R(1, 100) sleep(1000) display.show(Image.ARROW_W) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(0, 100) sleep(1000) display.show(Image("00900:""09990:""99999:""99999:""09090")) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Lower_L(0, 0) mecanumCar.Motor_Lower_L(0, 0) mecanumCar.Motor_Upper_R(0, 0)</pre>  | <pre>mecanumCar.Motor_Lower_R(1, 100) sleep(1000) display.show(Image.ARROW_W) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(0, 100) sleep(1000) display.show(Image("00900:""09990:""99999:""99999:""09090")) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Lower_L(0, 0) mecanumCar.Motor_Lower_R(0, 0) mecanumCar.Motor_Lower_R(0, 0) mecanumCar.Motor_Lower_R(0, 0) </pre>  | 19   | mecanumCar.Motor_Lower_L(0, 100)                             |    |
| <pre>sleep(1000)<br/>display.show(Image.ARROW_W)<br/>mecanumCar.Motor_Upper_L(1, 100)<br/>mecanumCar.Motor_Lower_L(1, 100)<br/>mecanumCar.Motor_Upper_R(0, 100)<br/>mecanumCar.Motor_Lower_R(0, 100)<br/>sleep(1000)<br/>display.show(Image("00900:""09990:""99999:""99999:""09090"))<br/>mecanumCar.Motor_Upper_L(0, 0)<br/>mecanumCar.Motor_Lower_L(0, 0)<br/>mecanumCar.Motor_Upper_R(0, 0)</pre>   | <pre>sleep(1000)<br/>display.show(Image.ARROW_W)<br/>mecanumCar.Motor_Upper_L(1, 100)<br/>mecanumCar.Motor_Lower_L(1, 100)<br/>mecanumCar.Motor_Upper_R(0, 100)<br/>sleep(1000)<br/>display.show(Image("00900:""09990:""99999:""99999:""09090"))<br/>mecanumCar.Motor_Upper_L(0, 0)<br/>mecanumCar.Motor_Lower_L(0, 0)<br/>mecanumCar.Motor_Upper_R(0, 0)<br/>mecanumCar.Motor_Lower_R(0, 0)</pre>   | 20   | <pre>mecanumCar.Motor_Upper_R(1, 100)</pre>                  |    |
| <pre>display.show(Image.ARROW_W) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) sleep(1000) display.show(Image("00900:""09990:""99999:""99999:""09090")) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Lower_L(0, 0) mecanumCar.Motor_Upper_R(0, 0)</pre>   | <pre>display.show(Image.ARROW_W) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(0, 100) mecanumCar.Motor_Lower_R(0, 100) sleep(1000) display.show(Image("00900:""09990:""99999:""99999:""09090")) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Lower_L(0, 0) mecanumCar.Motor_Lower_R(0, 0) mecanumCar.Motor_Lower_R(0, 0) mecanumCar.Motor_Lower_R(0, 0)</pre>   | 21   | <pre>mecanumCar.Motor_Lower_R(1, 100)</pre>                  |    |
| <pre>24 mecanumCar.Motor_Upper_L(1, 100)<br/>25 mecanumCar.Motor_Lower_L(1, 100)<br/>26 mecanumCar.Motor_Upper_R(0, 100)<br/>27 mecanumCar.Motor_Lower_R(0, 100)<br/>28 sleep(1000)<br/>29 display.show(Image("009000:""099900:""999999:""09090"))<br/>30 mecanumCar.Motor_Upper_L(0, 0)<br/>31 mecanumCar.Motor_Lower_L(0, 0)<br/>32 mecanumCar.Motor_Upper_R(0, 0)</pre>   | <pre>24 mecanumCar.Motor_Upper_L(1, 100)<br/>25 mecanumCar.Motor_Lower_L(1, 100)<br/>26 mecanumCar.Motor_Upper_R(0, 100)<br/>27 mecanumCar.Motor_Lower_R(0, 100)<br/>28 sleep(1000)<br/>29 display.show(Image("00900:""09990:""999999:""999999:""09090"))<br/>30 mecanumCar.Motor_Upper_L(0, 0)<br/>31 mecanumCar.Motor_Lower_L(0, 0)<br/>32 mecanumCar.Motor_Upper_R(0, 0)<br/>33 mecanumCar.Motor_Lower_R(0, 0)<br/>34 mecanumCar.Motor_Lower_R(0, 0)<br/>35 mecanumCar.Motor_Lower_R(0, 0)<br/>36 mecanumCar.Motor_Lower_R(0, 0)<br/>37 mecanumCar.Motor_Lower_R(0, 0)<br/>38 mecanumCar.Motor_Lower_R(0, 0)</pre>  | 22   | sleep(1000)  |    |
| <pre>25 mecanumCar.Motor_Lower_L(1, 100)<br/>26 mecanumCar.Motor_Upper_R(0, 100)<br/>27 mecanumCar.Motor_Lower_R(0, 100)<br/>28 sleep(1000)<br/>29 display.show(Image("00900:""099900:""999999:""999999:""09090"))<br/>30 mecanumCar.Motor_Upper_L(0, 0)<br/>31 mecanumCar.Motor_Lower_L(0, 0)<br/>32 mecanumCar.Motor_Upper_R(0, 0)</pre>   | <pre>25 mecanumCar.Motor_Lower_L(1, 100)<br/>25 mecanumCar.Motor_Upper_R(0, 100)<br/>27 mecanumCar.Motor_Lower_R(0, 100)<br/>28 sleep(1000)<br/>29 display.show(Image("00900:""09990:""999999:""999999:""09090"))<br/>30 mecanumCar.Motor_Upper_L(0, 0)<br/>31 mecanumCar.Motor_Lower_L(0, 0)<br/>32 mecanumCar.Motor_Upper_R(0, 0)<br/>33 mecanumCar.Motor_Lower_R(0, 0)<br/>34 mecanumCar.Motor_Lower_R(0, 0)<br/>35 mecanumCar.Motor_Lower_R(0, 0)<br/>36 mecanumCar.Motor_Lower_R(0, 0)<br/>37 mecanumCar.Motor_Lower_R(0, 0)<br/>38 mecanumCar.Motor_Lower_R(0, 0)<br/>39 mecanumCar.Motor_Lower_R(0, 0)</pre>  | 23   | display.show(Image.ARROW_W)                                  |    |
| <pre>26 mecanumCar.Motor_Upper_R(0, 100)<br/>27 mecanumCar.Motor_Lower_R(0, 100)<br/>28 sleep(1000)<br/>29 display.show(Image("00900:""099990:""999999:""09090"))<br/>30 mecanumCar.Motor_Upper_L(0, 0)<br/>31 mecanumCar.Motor_Lower_L(0, 0)<br/>32 mecanumCar.Motor_Upper_R(0, 0)</pre>  | <pre>26 mecanumCar.Motor_Upper_R(0, 100)<br/>27 mecanumCar.Motor_Lower_R(0, 100)<br/>28 sleep(1000)<br/>29 display.show(Image("009000:""099990:""999999:""09090"))<br/>30 mecanumCar.Motor_Upper_L(0, 0)<br/>31 mecanumCar.Motor_Lower_L(0, 0)<br/>32 mecanumCar.Motor_Lower_R(0, 0)<br/>33 mecanumCar.Motor_Lower_R(0, 0)<br/>34 mecanumCar.Motor_Lower_R(0, 0)<br/>35 mecanumCar.Motor_Lower_R(0, 0)<br/>36 mecanumCar.Motor_Lower_R(0, 0)<br/>37 mecanumCar.Motor_Lower_R(0, 0)<br/>38 mecanumCar.Motor_Lower_R(0, 0)<br/>39 mecanumCar.Motor_Lower_R(0, 0)<br/>30 mecanumCar.Motor_Lower_R(0, 0)</pre>   | 24   | mecanumCar.Motor_Upper_L(1, 100)                             |    |
| <pre>27 mecanumCar.Motor_Lower_R(0, 100) 28 sleep(1000) 29 display.show(Image("00900:""09990:""999999:""999999:""09090")) 30 mecanumCar.Motor_Upper_L(0, 0) 31 mecanumCar.Motor_Lower_L(0, 0) 32 mecanumCar.Motor_Upper_R(0, 0)</pre>  | <pre>27 mecanumCar.Motor_Lower_R(0, 100) 28 sleep(1000) 29 display.show(Image("00900:""099990:""999999:""999999:""09090")) 30 mecanumCar.Motor_Upper_L(0, 0) 31 mecanumCar.Motor_Lower_L(0, 0) 32 mecanumCar.Motor_Upper_R(0, 0) 33 mecanumCar.Motor_Lower_R(0, 0) 34 mecanumCar.Motor_Lower_R(0, 0) 35 mecanumCar.Motor_Lower_R(0, 0) 36 mecanumCar.Motor_Lower_R(0, 0) 37 mecanumCar.Motor_Lower_R(0, 0) 38 mecanumCar.Motor_Lower_R(0, 0) 39 mecanumCar.Motor_Lower_R(0, 0) 30 mecanumCar.Motor_Lower_R(0, 0) 30 mecanumCar.Motor_Lower_R(0, 0) 31 mecanumCar.Motor_Lower_R(0, 0) 32 mecanumCar.Motor_Lower_R(0, 0) 33 mecanumCar.Motor_Lower_R(0, 0) 34 mecanumCar.Motor_Lower_R(0, 0) 35 mecanumCar.Motor_Lower_R(0, 0) 36 mecanumCar.Motor_Lower_R(0, 0) 37 mecanumCar.Motor_Lower_R(0, 0) 38 mecanumCar.Motor_Lower_R(0, 0) 39 mecanumCar.Motor_Lower_R(0, 0) 30 mecanumCar.Motor_Lower_R(0, 0) 31 mecanumCar.Motor_Lower_R(0, 0) 32 mecanumCar.Motor_Lower_R(0, 0) 33 mecanumCar.Motor_Lower_R(0, 0) 34 mecanumCar.Motor_Lower_R(0, 0) 35 mecanumCar.Motor_Lower_R(0, 0) 36 mecanumCar.Motor_Lower_R(0, 0) 37 mecanumCar.Motor_Lower_R(0, 0) 39 mecanumCar.Motor_Lower_R(0, 0) 30 mecanumCar.Motor_Lower_R(0, 0) 30 mecanumCar.Motor_Lower_R(0, 0) 31 mecanumCar.Motor_Lower_R(0, 0) 31 mecanumCar.Motor_Lower_R(0, 0) 31 mecanumCar.Motor_Lower_R(0, 0) 32 mecanumCar.Motor_Lower_R(0, 0) 33 mecanumCar.Motor_Lower_R(0, 0) 34 mecanumCar.Motor_Lower_R(0, 0) 35 mecanumCar.Motor_Lower_R(0, 0) 35 mecanumCar.Motor_Lower_R(0, 0) 36 mecanumCar.Motor_Lower_R(0, 0) 37 mecanumCar.Motor_Lower_R(0, 0) 38 mecanumCar.Motor_Lower_R(0, 0) 39 mecanumCar.Motor_Lower_R(0, 0) 30 mecanumCar.Motor_Lower_R(0, 0) 30 mecanumCar.Motor_Lower_R(0, 0) 31 mecanumCar.Motor</pre>   | 25   | <pre>mecanumCar.Motor_Lower_L(1, 100)</pre>                  |    |
| <pre>sleep(1000) display.show(Image("00900:""09990:""999999:""09090")) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Lower_L(0, 0) mecanumCar.Motor_Upper_R(0, 0)</pre>  | <pre>sleep(1000) display.show(Image("00900:""09990:""999999:""999999:""09090")) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Lower_L(0, 0) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Motor_Lower_R(0, 0)</pre>  | 26   | <pre>mecanumCar.Motor_Upper_R(0, 100)</pre>                  |    |
| <pre>29 display.show(Image("00900:""099990:""999999:""999999:""09090")) 30 mecanumCar.Motor_Upper_L(0, 0) 31 mecanumCar.Motor_Lower_L(0, 0) 32 mecanumCar.Motor_Upper_R(0, 0)</pre>  | <pre>display.show(Image("00900:""09990:""999999:""999999:""09090")) mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Lower_L(0, 0) mecanumCar.Motor_Upper_R(0, 0) mecanumCar.Motor_Lower_R(0, 0)</pre>  | 27   | <pre>mecanumCar.Motor_Lower_R(0, 100)</pre>                  |    |
| <pre>mecanumCar.Motor_Upper_L(0, 0) mecanumCar.Motor_Lower_L(0, 0) mecanumCar.Motor_Upper_R(0, 0)</pre>  | <pre>so mecanumCar.Motor_Upper_L(0, 0) s1 mecanumCar.Motor_Lower_L(0, 0) s2 mecanumCar.Motor_Upper_R(0, 0) s3 mecanumCar_Motor_Lower_R(0, 0)</pre>   | 28   | sleep(1000)  |    |
| <pre>s1 mecanumCar.Motor_Lower_L(0, 0) s2 mecanumCar.Motor_Upper_R(0, 0)</pre>   | <pre>s1 mecanumCar.Motor_Lower_L(0, 0) s2 mecanumCar.Motor_Upper_R(0, 0) s3 mecanumCar_Motor_Lower_R(0, 0)</pre>   | 29   | display.show(Image("00900:""09990:""99999:""99999:""09090")) |    |
| <pre>mecanumCar.Motor_Upper_R(0, 0)</pre>  | <pre>sz mecanumCar.Motor_Upper_R(0, 0) sz mecanumCar_Motor_Lower_R(0, 0)</pre>   | 30   | <pre>mecanumCar.Motor_Upper_L(0, 0)</pre>                    |    |
|  | mecanumCar Motor Lower R(0 0)  | 31   | <pre>mecanumCar.Motor_Lower_L(0, 0)</pre>                    |    |
| mecanum(ar Motor Lower R(0, 0)   |  | 32   | <pre>mecanumCar.Motor_Upper_R(0, 0)</pre>                    |    |
|  | В  |      | mecanumCar Motor Lower R(0 0)                                |    |



## Code 2:

Enter Mu software and open the file "Project 16: Motor.py" to import code:

```
(How to load the project code?)
```

| File   | Route                  | File Name |
|--------|------------------------|-----------|
| Туре   |                        |           |
| Python | KS4031(KS4032)         | Code-2.py |
| file   | folder/Python          |           |
|        | Tutorial/Python        |           |
|        | Code/Project 16: Motor |           |

You can also input code in the editing window yourself.

(note:all words and symbols must be written in English)



| 🕐 Mu :   | 1.1.0.b | eta.5 - microbit-Motor Driving-2.py                                    |   |                       |     |
|----------|---------|--|---|-----------------------|-----|
| Mode     |         |  | PL Plotter Zoom-in                      | Q C<br>Zoom-out Theme |     |
| -        | A - 220 | or Driving-2.py  |   |                       |     |
| 1        |         | n microbit import button_a, t  | utton b. display.                       | Image, sleep          |     |
|          |         | <pre>n keyes_mecanum_Car import *</pre>                                | _ , _ , , , , , , , , , , , , , , , , , |                       |     |
| 1.00     |         | anumCar = Mecanum_Car_Driver   | )                                       |                       |     |
| 4        |         |  |   |                       |     |
|          |         | v_L = Image("90000:""90000:"'  |   |                       |     |
| 6        | show    | v_0 = Image("09990:""90009:"'  | 00009:""9 <mark>0009:""</mark> 09       | 990")                 |     |
| 7        | a =     | 0  |   |                       |     |
|          | b =     |  |   |                       |     |
| 9        | def     | run_L():   |   |                       |     |
| 10       |         | global b   |   |                       |     |
| 11       |         | sleep(1000)  |   |                       |     |
| 12       |         | <pre>mecanumCar.Motor_Upper_L(1,</pre>                                 |   |                       |     |
| 13       |         | <pre>mecanumCar.Motor_Lower_L(1,</pre>                                 |   |                       |     |
| 14       |         | <pre>mecanumCar.Motor_Upper_R(1,</pre>                                 |   |                       |     |
| 15       |         | <pre>mecanumCar.Motor_Lower_R(1,</pre>                                 | 100)                                    |                       |     |
| 16       |         | sleep(1000)  |   |                       |     |
| 17       |         | <pre>mecanumCar.Motor_Upper_L(0,</pre>                                 |   |                       |     |
| 18       |         | <pre>mecanumCar.Motor_Lower_L(0,</pre>                                 |   |                       |     |
| 19       |         | <pre>mecanumCar.Motor_Upper_R(1,</pre>                                 |   |                       |     |
| 20       |         | <pre>mecanumCar.Motor_Lower_R(1,</pre>                                 | 100)                                    |                       |     |
| 21       |         | sleep(650)   |   |                       |     |
| 22       |         | <pre>mecanumCar.Motor_Upper_L(1,</pre>                                 |   |                       |     |
| 23       |         | <pre>mecanumCar.Motor_Lower_L(1,</pre>                                 |   |                       |     |
| 24       |         | <pre>mecanumCar.Motor_Upper_R(1,</pre>                                 |   |                       |     |
| 25       |         | <pre>mecanumCar.Motor_Lower_R(1,</pre>                                 | 100)                                    |                       |     |
| 26       |         | sleep(1000)  | 2)                                      |                       |     |
| 27       |         | <pre>mecanumCar.Motor_Upper_L(0,<br/>mecanumCar.Motor_Lower_L(0,</pre> | (3)                                     |                       |     |
| 28       |         |  |   |                       |     |
| 29       |         | <pre>mecanumCar.Motor_Upper_R(0,<br/>mecanumCar.Motor_Lower_R(0,</pre> |   |                       |     |
| 30       |         | b = 0  | 5)                                      |                       |     |
| 31<br>32 | def     | run_0():   |   |                       |     |
| 32       | uer     | global h   |   |                       |     |
|          |         |  |   | BBC micro:bit         | -   |
|          |         |  |   |                       | TIT |



```
grobar b
33
       sleep(1000)
34
       mecanumCar.Motor_Upper_L(1, 100)
35
       mecanumCar.Motor_Lower_L(1, 100)
36
       mecanumCar.Motor Upper R(1, 100)
37
       mecanumCar.Motor_Lower_R(1, 100)
38
       sleep(1000)
39
       mecanumCar.Motor_Upper_L(0, 100)
40
       mecanumCar.Motor_Lower_L(0, 100)
41
       mecanumCar.Motor_Upper_R(1, 100)
42
       mecanumCar.Motor_Lower_R(1, 100)
43
       sleep(620)
44
       mecanumCar.Motor_Upper_L(1, 100)
45
       mecanumCar.Motor_Lower_L(1, 100)
46
       mecanumCar.Motor_Upper_R(1, 100)
47
       mecanumCar.Motor_Lower_R(1, 100)
48
       sleep(1000)
49
       mecanumCar.Motor_Upper_L(0, 100)
50
       mecanumCar.Motor_Lower_L(0, 100)
51
       mecanumCar.Motor_Upper_R(1, 100)
52
       mecanumCar.Motor_Lower_R(1, 100)
53
       sleep(620)
54
       mecanumCar.Motor Upper L(1, 100)
55
       mecanumCar.Motor_Lower_L(1, 100)
56
       mecanumCar.Motor_Upper_R(1, 100)
57
       mecanumCar.Motor_Lower_R(1, 100)
58
       sleep(1000)
59
       mecanumCar.Motor_Upper_L(0, 100)
60
       mecanumCar.Motor_Lower_L(0, 100)
61
       mecanumCar.Motor_Upper_R(1, 100)
62
       mecanumCar.Motor_Lower_R(1, 100)
63
       sleep(620)
64
       mecanumCar Motor Unner 1 (1 100)
                                                               BBC micro:bit
```



```
mecanumcar .motor_opper_t(1, 100)
65
       mecanumCar.Motor_Lower_L(1, 100)
66
       mecanumCar.Motor_Upper_R(1, 100)
67
       mecanumCar.Motor_Lower_R(1, 100)
69
       sleep(1000)
69
       mecanumCar.Motor_Upper_L(0, 0)
70
       mecanumCar.Motor_Lower_L(0, 0)
71
       mecanumCar.Motor_Upper_R(0, 0)
72
       mecanumCar.Motor_Lower_R(0, 0)
73
       b = 0
74
  while True:
75
       if button_a.was_pressed():
76
            a = a + 1
77
            if a >= 3:
78
                a = 0
79
       if button_b.was_pressed():
80
            b = 1
81
       if (a == 1):
82
            display.show(show_L)
83
            if b == 1:
84
                run_L()
85
       elif a == 2:
86
            display.show(show_0)
87
            if b == 1:
88
                run_0()
89
90
                                                                 BBC micro:bit
```

Click "Files" to import "keyes\_mecanum\_Car.py "library file tomicro:bit (<u>How</u> <u>to import files?</u>). No need to do it again if you have imported it before.



| 1  | <pre>New Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme tor Driving=2.py sleep(1000) mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Lower_L(0, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(620) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000) mecanumCar.Motor_Lower_R(1, 100) sleep(1000) mecanumCar.Motor_Upper_L(0, 0)</pre> |
|--|---|
| 60<br>61<br>62<br>63<br>64<br>65<br>65<br>66<br>67<br>68<br>69<br>70 | <pre>mecanumCar.Motor_Upper_L(0, 100) mecanumCar.Motor_Lower_L(0, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000)</pre>  |
| 61<br>62<br>63<br>64<br>65<br>66<br>67<br>68<br>69<br>70             | <pre>mecanumCar.Motor_Lower_L(0, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(620) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(1, 100) sleep(1000)</pre>   |
| 62<br>63<br>64<br>65<br>66<br>67<br>68<br>69<br>70                   | <pre>mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(620) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(1, 100) sleep(1000)</pre>  |
| 63<br>64<br>65<br>66<br>67<br>68<br>69<br>70                         | <pre>mecanumCar.Motor_Lower_R(1, 100) sleep(620) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000)</pre>  |
| 64<br>65<br>66<br>67<br>68<br>69<br>70                               | <pre>sleep(620) mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000)</pre>   |
| 65<br>66<br>67<br>68<br>69<br>70                                     | <pre>mecanumCar.Motor_Upper_L(1, 100) mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000)</pre>  |
| 66<br>67<br>68<br>69<br>70   | <pre>mecanumCar.Motor_Lower_L(1, 100) mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000)</pre>   |
| 67<br>68<br>69<br>70   | <pre>mecanumCar.Motor_Upper_R(1, 100) mecanumCar.Motor_Lower_R(1, 100) sleep(1000)</pre>  |
| 68<br>69<br>70   | <pre>mecanumCar.Motor_Lower_R(1, 100) sleep(1000)</pre>   |
| 69<br>70   | sleep(1000)   |
| 70   |   |
|  | <pre>mecanumCar.Motor_Upper_L(0, 0)</pre>   |
| 71   |   |
|  | <pre>mecanumCar.Motor_Lower_L(0, 0)</pre>   |
| 72   | <pre>mecanumCar.Motor_Upper_R(0, 0)</pre>   |
| 73   | <pre>mecanumCar.Motor_Lower_R(0, 0)</pre>   |
| 74   | b = 0   |
| 75 whi   | le True:  |
| 76   | <pre>if button_a.was_pressed():</pre>   |
| 77   | a = a + 1   |
| 78   | if a >= 3:  |
| 79   | a = 0   |
| so   | <pre>if button_b.was_pressed():</pre>   |
| S1   | b = 1   |
| 82   | if (a == 1):  |
| 83   | display.show(show_L)  |
| S4   | if b == 1:  |
| 85   | run_L()   |
| <b>S</b> 6   | elif a == 2:  |

Tap "Check" button to confirm if the code has errors. The program proves wrong if there are underlines and cursors.



| Mode     | Image: New Load     Image: Save     Image: Save |   |
|----------|---|---|
| icrobit- | Motor Driving-2.py 🗙  | 1 |
| 59       | sleep(1000)   |   |
| 60       | mecanumCar.Motor_Upper_L(0, 100)  |   |
| 61       | mecanumCar.Motor_Lower_L(0, 100)  |   |
| 62       | <pre>mecanumCar.Motor_Upper_R(1, 100)</pre>   |   |
| 63       | mecanumCar.Motor_Lower_R(1, 100)  |   |
| 64       | sleep(620)  |   |
| 65       | mecanumCar.Motor_Upper_L(1, 100)  |   |
| 66       | mecanumCar.Motor_Lower_L(1, 100)  |   |
| 67       | <pre>mecanumCar.Motor_Upper_R(1, 100)</pre>   |   |
| 68       | <pre>mecanumCar.Motor_Lower_R(1, 100)</pre>   |   |
| 69       | sleep(1000)   |   |
| 70       | <pre>mecanumCar.Motor_Upper_L(0, 0)</pre>   |   |
| 71       | <pre>mecanumCar.Motor_Lower_L(0, 0)</pre>   |   |
| 72       | mecanumCar.Motor_Upper_R(0, 0)  |   |
| 73       | <pre>mecanumCar.Motor_Lower_R(0, 0)</pre>   |   |
| 74       | $b = \Theta$  |   |
| 75 W     | ile True:   |   |
| 76       | if button_a.was_pressed():  |   |
| 77       | a = a + 1   |   |
| 78       | if a >= 3:  |   |
| 79       |   |   |
| SO       | <pre>if button_b.was_pressed():</pre>   |   |
| 81       | b = 1   |   |
| 82       | if (a == 1):  |   |
| 83       | display.show(show_L)  |   |
| 84       | if b == 1:  |   |
| 85       | run_L()   |   |
| 86       | elif a == 2:<br>display_show(show_0)  |   |

If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.



| <b>P</b><br>Mode | Image: New Load     Image: Save     Image: Save |
|------------------|---|
| icrobi           | it-Motor Driving-2.py 🕱   |
| 59               | sleep(1000)   |
| 60               | <pre>mecanumCar.Motor_Upper_L(0, 100)</pre>   |
| 61               | mecanumCar.Motor_Lower_L(0, 100)  |
| 62               | <pre>mecanumCar.Motor_Upper_R(1, 100)</pre>   |
| 63               | <pre>mecanumCar.Motor_Lower_R(1, 100)</pre>   |
| 64               | sleep(620)  |
| 65               | <pre>mecanumCar.Motor_Upper_L(1, 100)</pre>   |
| 66               | <pre>mecanumCar.Motor_Lower_L(1, 100)</pre>   |
| 67               | <pre>mecanumCar.Motor_Upper_R(1, 100)</pre>   |
| 68               | <pre>mecanumCar.Motor_Lower_R(1, 100)</pre>   |
| 69               | sleep(1000)   |
| 70               | <pre>mecanumCar.Motor_Upper_L(0, 0)</pre>   |
| 71               | <pre>mecanumCar.Motor_Lower_L(0, 0)</pre>   |
| 72               | <pre>mecanumCar.Motor_Upper_R(0, 0)</pre>   |
| 73               | <pre>mecanumCar.Motor_Lower_R(0, 0)</pre>   |
| 74               | b = 0   |
| 75               | while True:   |
| 76               | <pre>if button_a.was_pressed():</pre>   |
| 77               | a = a + 1   |
| 78               | if a >= 3:  |
| 79               | a = 0   |
| so               | <pre>if button_b.was_pressed():</pre>   |
| S1               | b = 1   |
| 82               | if (a == 1):  |
| 83               | display.show(show_L)  |
| 84               | if b == 1:  |
| 85               | run_L()   |
| 86               | elif a == 2:  |
|                  | dicplay chow(chow 0)  |

# (4)Test Results:



Download code 1 to micro:bit, and turn on the switch on robot car. The robot car will go forward for 1s, back for 1s, turn left for 1s, right for 1s, turn anticlockwise for 1s, clockwise for 1 and stop 1s. Matrix also displays the patterns.

Download code 2 to micro:bit board, dial POWER switch to ON end.

When the button A and B are firstly pressed, micro" bit will show "L", the route of car is "L". When they are pressed again, " $\Box$ " is shown on micro:bit, and route of car is " $\Box$ ". The car repeats this pattern.

#### (5)Code Explanation:

| from   | microbit                        | import           | button_a,  | Due to insufficient memory,     |
|--------|---------------------------------|------------------|------------|---------------------------------|
| button | button_b, display, Image, sleep |                  |            | only the necessary parts such   |
|        |                                 |                  |            | as button_a, button_b, display, |
|        |                                 |                  |            | Image, sleep and so on in the   |
|        |                                 |                  |            | micro:bit library file are      |
|        |                                 |                  |            | imported here                   |
| from   | keyes_mecanu                    | ım_Car <b>ir</b> | nport *    | Import library file             |
|        |                                 |                  |            | keyes_mecanum_Car               |
| mecani | umCar =Mecai                    | num_Car_[        | Driver()() | Instantiate an object           |
|        |                                 |                  |            | Mecanum_Car_Driver()() as       |
|        |                                 |                  |            | mecanumCar                      |



| while True:                            | This is a permanent loop that  |
|--|--------------------------------|
|  | makes micro:bit execute the    |
|  | code of it.                    |
| display.show(Image.ARROW_S)            | micro:bit shows arrow pointing |
| display.show(Image.ARROW_N)            | to South                       |
| display.show(Image.ARROW_E)            | micro:bit shows arrow pointing |
| display.show(Image.ARROW_W)            | to North                       |
| display.show(Image("00900:""09990:""99 | micro:bit shows arrow pointing |
| 999:""99999:""09090"))                 | to East                        |
|  | micro:bit shows arrow pointing |
|  | to West                        |
|  | micro:bit displays "•"         |
| mecanumCar.Motor_Upper_L(1, 100)       | The left motor of car rotates  |
| mecanumCar.Motor_Lower_L(1, 100)       | clockwise at the speed of      |
| mecanumCar.Motor_Upper_R(1, 100)       | PWM100                         |
| mecanumCar.Motor_Lower_R(1, 100)       | (1: clockwise, 0:              |
|  | anticlockwise; PWM100 means    |
|  | speed (0~255) )                |
|  |                                |
|  | The rear left motor of car     |
|  | rotates clockwise at the speed |
|  | of PWM100.                     |



|                                  | The front right motor of car<br>rotates clockwise at the speed<br>of PWM100.<br>The rear right motor of car |
|----------------------------------|---|
|                                  | rotates clockwise at the speed  |
|                                  | of PWM100.  |
|                                  |   |
| mecanumCar.Motor_Upper_L(0, 100) | The front left motor of car   |
| mecanumCar.Motor_Lower_L(0, 100) | rotates anticlockwise at the  |
| mecanumCar.Motor_Upper_R(0, 100) | speed of PWM100.  |
| mecanumCar.Motor_Lower_R(0, 100) |   |
|                                  | The rear left motor of car  |
|                                  | rotates anticlockwise at the  |
|                                  | speed of PWM100.  |
|                                  | The front right motor of car  |
|                                  | rotates anticlockwise at the  |
|                                  | speed of PWM100.  |
|                                  | The rear right motor of car   |
|                                  | rotates anticlockwise at the  |



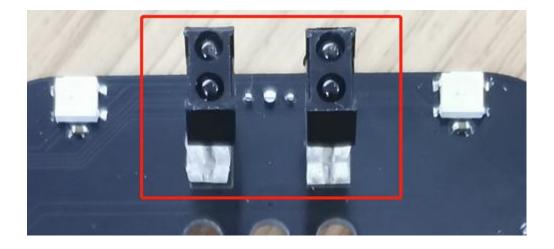
|  | speed of PWM100.                  |
|--|-----------------------------------|
| sleep(1000)                            | Delay in 1000ms                   |
| a = 0                                  | Set the initial value of variable |
| b = 0                                  | a to 0                            |
|  | Set the initial value of variable |
|  | b to 0                            |
| def run_L():                           | Define sub-function run_L()       |
| def run_O():                           | Define sub-function run_O()       |
| show_L =                               | Assign Image() to the variable    |
| Image("90000:""90000:""90000:""90000:" | show_L                            |
| "99999")                               |                                   |
| if button_a.was_pressed():             | if button A is pressed,           |
| a = a + 1                              | a = a + 1                         |
| <b>if</b> a >= 3:                      | lf a≥3                            |
| a = 0                                  | a=0                               |
| if button_b.was_pressed():             | If button B is pressed,           |
| b = 1                                  | b=1                               |
| <b>if</b> (a == 1):                    | If a=1                            |
| display.show(show_L)                   | micro:bit shows "L" pattern       |
| <b>if</b> b == 1:                      | If b=1                            |
| run_L()                                | The track of car is route L       |



| <b>elif</b> a == 2:  | If a=2                      |
|----------------------|-----------------------------|
| display.show(show_O) | micro:bit shows "O" image   |
| <b>if</b> b == 1:    | If b=1                      |
| run_O()              | The track of car is route O |

#### **Project 17: Line Tracking Sensor**

#### 17.1: Detect Line Tracking Sensor



#### (1)Project Description

The motor driving board of the Keyestudio 4WD Mecanum Robot Car comes with a dual-channel line tracking sensors which adopt TCRT5000 IR tubes and 2 potentiometers.

TCRT5000 IR tube has an IR emitting tube and a receiving tube.

Low level(0) is output when IR transmitting tube emits IR signals to



receiving tube; high level(1) will be output when smart car runs along black line.

When smart car drives on the white ground, TCRT5000 IR tube will emit IR signals which will be reflected by white ground and received by receiving tube, consequently output low level(0); on the contrary, when driving on the black surface, the high level is output.

### (2)Working Principle:

When the car runs above a white road, the infrared transmitter tube installed under the car emits infrared signals to detect the road and the receiver tube receives signals sending back. Then the output end outputs low level(0); when it detects black lines, it outputs high level(1).

The 2-way tracking sensor integrated port on the 4WD Mecanum Robot Car is connected to the collection port of G ,5V ,P1 and P2 on the micro:bit expansion board, which is controlled by the P1 and P2 of the micro:bit. The left TCRT5000 infrared pair tube on the sensor is controlled by P1, and the right one by P2.

After putting a white paper on the bottom of the 4WD Mecanum Robot Car,we rotate the two potentiometers on the 2-way tracking sensor. When the indicator light on the sensor module is on, pick up the car to make the



two wheels on the 4WD Mecanum Robot Car separate. The height of the white paper is about 1.5cm, the indicator light on the sensor module is off, and then the sensitivity is adjusted.

### (2) Experimental Preparation:

- > Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car
- > Place batteries into battery holder
- > Dial power switch to ON end
- > Connect micro:bit to computer by USB cable
- > Open the offline version of Mu.

### (4)Test Code:

Enter Mu software and open the file "Project 17: Line Tracking Sensor.py" to import code:

### (How to load the project code?)

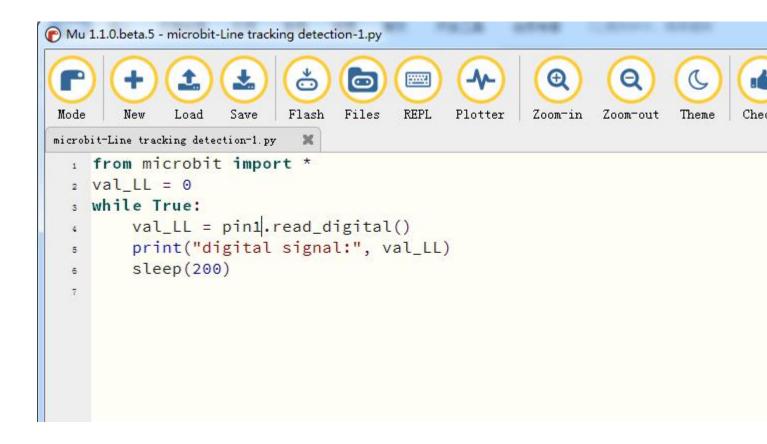
| File   | Route                  | File Name |
|--------|------------------------|-----------|
| Туре   |                        |           |
| Python | KS4031(KS4032)         | Code-1.py |
| file   | folder/Python          |           |
|        | Tutorial/Python        |           |
|        | Code/Project 17 : Line |           |



Tracking Sensor/17.1

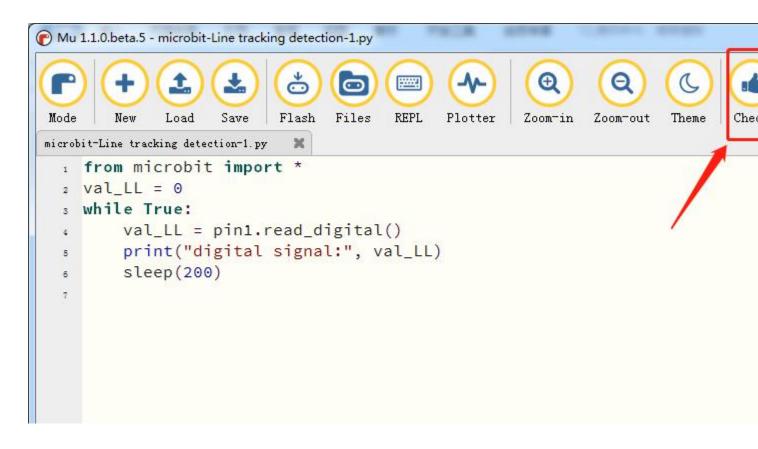
You can also input code in the editing window yourself.

(note:all words and symbols must be written in English)



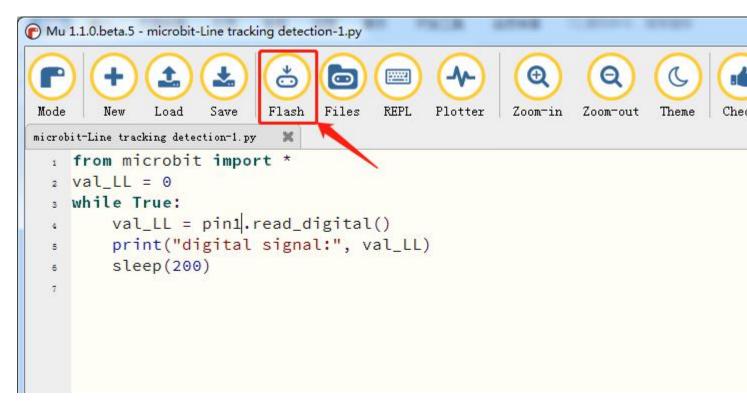
Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.





If the code is correct, connect micro:bit to computer and click "Flash" to

download code to micro:bit board.





Download code 1 onto micro:bit board, don't plug off USB cable. Click "REPL" and press the reset buttons, the readings detected by left TCRT5000 IR tube are displayed on monitor.

When the left TCRT5000 IR tube detects white object, 0 will be shown and left indicator will be on; when no white objects and only black object are detected, 1 will be displayed and indicator will be off, as shown below.

| P Mu 1.1.0.beta.5 - microbit-Line tracking detection-1.py  |
|--|
| Mode New Load Save Flash Files REPL Plotter Zoom-in Zoom-ou  |
| microbit-Line tracking detection-1.py 🞇  |
| 1 from microbit import *   |
| $_2$ val_LL = 0  |
| 3 while True:  |
| <pre>val_LL = pin1.read_digital()</pre>  |
| <pre>print("digital signal:", val_LL)</pre>  |
| <pre>6 sleep(200)</pre>  |
| 7  |
| BBC micro:bit REPL<br>digital signal: 0<br>digital signal: 0<br>digital signal: 0<br>digital signal: 1<br>digital signal: 1<br>digital signal: 1<br>digital signal: 1<br>digital signal: 1<br>digital signal: 1<br>digital signal: 1 |
| digital signal: 0  |
|  |
|  |



# Code2:

Enter Mu software and open the file "Project 17: Line Tracking Sensor.py"

to import code:

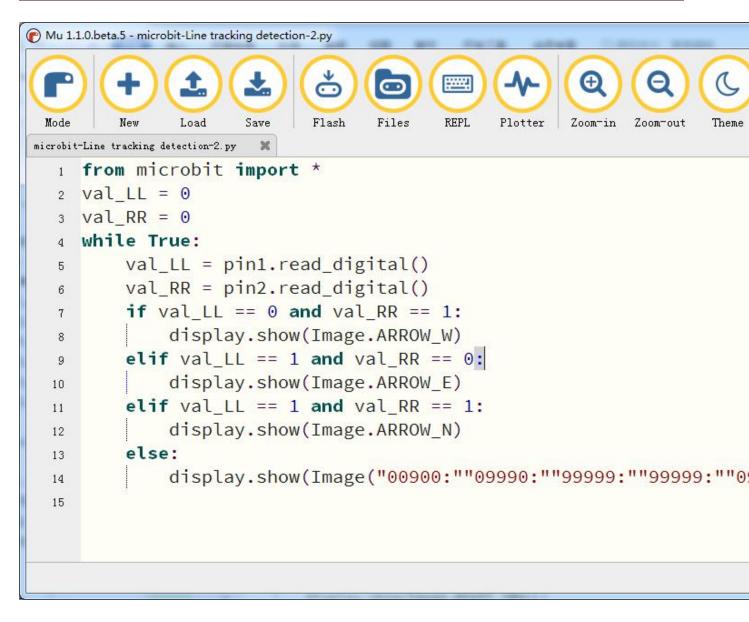
(How to load the project code?)

| File   | Route                  | File Name |
|--------|------------------------|-----------|
| Туре   |                        |           |
| Python | KS4031(KS4032)         | Code-2.py |
| file   | folder/Python          |           |
|        | Tutorial/Python        |           |
|        | Code/Project 17 : Line |           |
|        | Tracking Sensor/17.1   |           |

You can also input code in the editing window yourself.

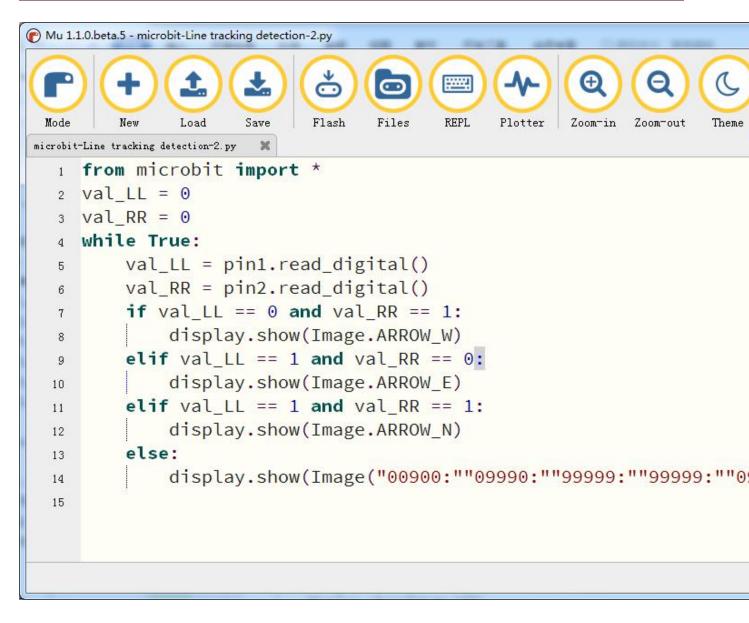
(note:all words and symbols must be written in English)





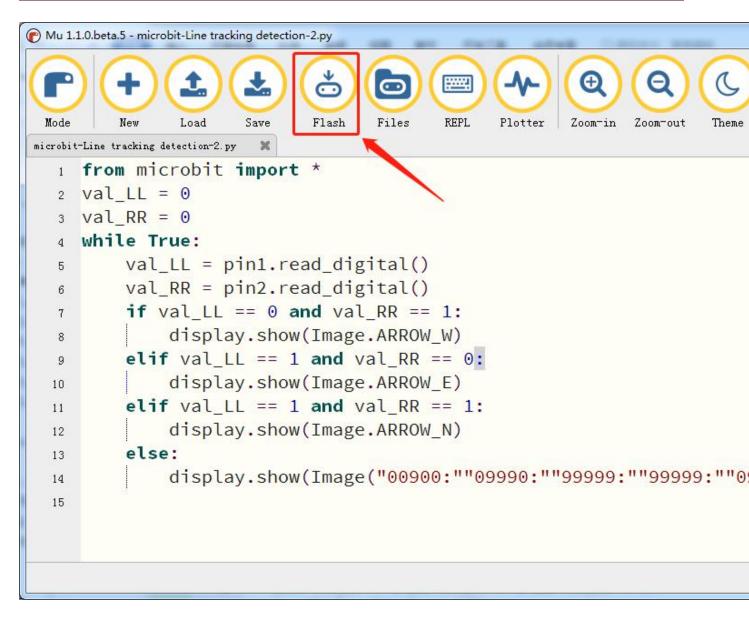
Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.





If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.





#### (5)Test Results:

Download code 2 to the micro:bit, when only the left TCRT5000 infrared pair tube on the line tracking sensor detects a white object, the micro bit LED dot matrix displays a "←" pattern, and the indicator light on the left side of the tracking sensor lights up;

When only the right TCRT5000 infrared pair tube on the sensor detects a



white object, the micro bit LED dot matrix displays a " $\rightarrow$ " pattern, and the indicator light on the right side of the tracking sensor lights up;

When both the TCRT5000 infrared pair tubes on the sensor detect a white object, the micro bit LED dot matrix displays a "♥" pattern, and the indicator light on the both sides of the tracking sensor light up;

When none of the TCRT5000 infrared pair tubes on the sensor detect a white object, the micro bit LED dot matrix displays a "1" pattern, and the indicator light on the both sides of the tracking sensor remain off;

| from microbit import *       | import the library file of micro:bit    |
|------------------------------|---|
| val_LL = 0                   | Set the initial value of val_LL to 0    |
| val_RR = 0                   | Set the initial value of val_RR to 0    |
| while True:                  | This is a permanent loop that makes     |
|                              | micro:bit execute the code of it.       |
| val_LL = pin1.read_digital() | Set the digital signal read by TCRT5000 |
|                              | IR tube connected to P1 to val_LL       |
| val_RR = pin2.read_digital() | Set the digital signal read by TCRT5000 |
|                              | IR tube connected to P2 to val_LL       |

### (6)Code Explanation:



| <b>if</b> val_LL == 0 <b>and</b> val_RR == 1: | If val_LL = 0 and val_RR = 1 is true             |
|---|--|
| display.show(Image.ARROW_W)                   | The symbol "—" is displayed on the left          |
| elif val_LL == 1 and val_RR == 0:             | of the LED dot matrix on the micro:bit;          |
| display.show(Image.ARROW_E)                   |  |
| elif val_LL == 1 and val_RR == 1:             | If val_LL =1 and val_RR = 0 is true              |
| display.show(Image.ARROW_N)                   | The symbol " $\rightarrow$ " is displayed in the |
| else:   | middle of the LED dot matrix;                    |
| display.show(Image("00900:""0                 | If val_LL =1 and val_RR = 1 is true              |
| 9990:""99999:""99999:""09090"                 | The symbol "↑" is displayed on the               |
| ))  | right of the LED dot matrix;                     |
|   | When none of the above conditions                |
|   | are met,   |
|   | the LED dot matrix displays the                  |
|   | pattern "♥".                                     |



## 17.2: Line Tracking Smart Car



### (1)Project Description

In this lesson we will combine line tracking sensors with a motor to make a line tracking smart car.

The micro:bit board will analyze the signals and control smart car to show line tracking function.

#### (2) The Working Principle

The smart car will make different moves according to the value received by the 3 channel line tracking sensor.

| Left/Right TCRT5000 |          | 4WD Mecanum Ro |
|---------------------|----------|----------------|
| IR Tunes (Level)    |          | bot Car        |
| LOW (0)             | HIGH (1) | Turn Right     |
| HIGH (1)            | LOW (0)  | Turn Left      |
| HIGH (1)            | HIGH (1) | Go forward     |
| LOW (0)             | LOW (0)  | Stop           |

The 2-way tracking sensor integrated port on the 4WD Mecanum Robot Car is connected to the collection port of G ,5V ,P1 and P2 on the micro:bit expansion board, which is controlled by the P1 and P2 of the micro:bit. The left TCRT5000 infrared pair tube on the sensor is controlled by P1, and the right one by P2.

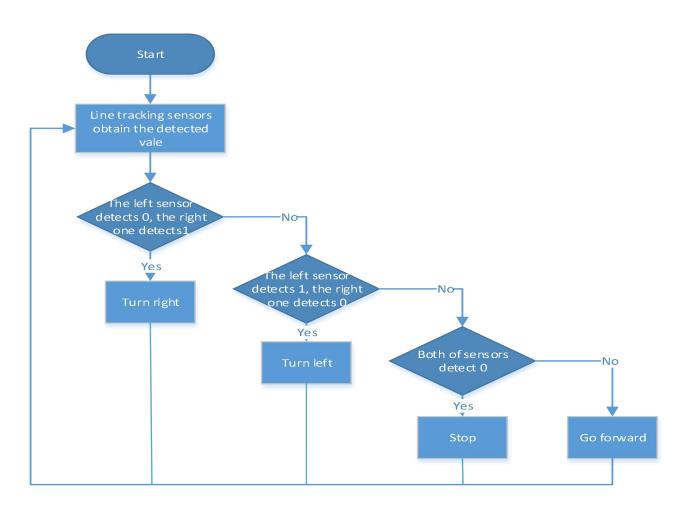
### (2) Experimental Preparation:

- > Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car
- > Place batteries into battery holder
- > Dial power switch to ON end
- Connect micro:bit to computer by USB cable
- > Open the offline version of Mu.



Warning: The 2-way tracking sensor should be used in environments without infrared interference such as sunlight. Sunlight contains a lot of invisible light, such as infrared and ultraviolet. In an environment with strong sunlight, the 2-way tracking sensor cannot work properly.

## (3)Flow Chart:



### (4)Test Code:

Enter Mu software and open the file "Project 17: Line Tracking Sensor.py"

to import code:

```
(How to load the project code?)
```



| File   | Route                  | File Name            |
|--------|------------------------|----------------------|
| Туре   |                        |                      |
| Python | KS4031(KS4032)         | Line tracking car.py |
| file   | folder/Python          |                      |
|        | Tutorial/Python        |                      |
|        | Code/Project 17 : Line |                      |
|        | Tracking Sensor/17.2   |                      |

You can also input code in the editing window yourself.

(note:all words and symbols must be written in English)



| 9 Mu 1. | 1.0.beta.5 - microbit- Line tracking car.py  |
|---------|--|
| Mode    | Image: New Load     Save     Flash     Files     REPL     Plotter     Zoom-in     Zoom-out     Theme |
|         | - Line tracking car.py   |
|         | rom microbit import *  |
|         | rom keyes_mecanum_Car import *   |
|         | ecanumCar = Mecanum_Car_Driver()   |
|         | lisplay.show(Image.HAPPY)  |
|         | al_LL = 0  |
|         | $ral_{RR} = 0$   |
|         | hile True:   |
| 8       | val_LL = pin1.read_digital()   |
| 9       | val_RR = pin2.read_digital()   |
| 10      | <pre>if val_LL == 0 and val_RR == 1:</pre>   |
| 11      | mecanumCar.Motor_Upper_L(1, 220)   |
| 12      | mecanumCar.Motor_Lower_L(1, 220)   |
| 13      | mecanumCar.Motor_Upper_R(0, 120)   |
| 14      | mecanumCar.Motor_Lower_R(0, 120)   |
| 15      | elif val_LL == 1 and val_RR == 0:  |
| 16      | mecanumCar.Motor_Upper_L(0, 120)   |
| 17      | mecanumCar.Motor_Lower_L(0, 120)   |
| 18      | <pre>mecanumCar.Motor_Upper_R(1, 220)</pre>  |
| 19      | mecanumCar.Motor_Lower_R(1, 220)   |
| 20      | <pre>elif val_LL == 0 and val_RR == 0:</pre>   |
| 21      | <pre>mecanumCar.Motor_Upper_L(0, 0)</pre>  |
| 22      | <pre>mecanumCar.Motor_Lower_L(0, 0)</pre>  |
| 23      | <pre>mecanumCar.Motor_Upper_R(0, 0)</pre>  |
| 24      | <pre>mecanumCar.Motor_Lower_R(0, 0)</pre>  |
| 25      | else:  |
| 26      | <pre>mecanumCar.Motor_Upper_L(1, 100)</pre>  |
| 27      | mecanumCar.Motor_Lower_L(1, 100)   |
| 28      | mecanumCar.Motor_Upper_R(1, 100)   |
| 29      | mecanumCar.Motor_Lower_R(1, 100)   |
| 30      |  |



Click "Files" to import "keyes\_mecanum\_Car.py "library file tomicro:bit (How

to import files? ). No need to do it again if you have imported it before.

| Mu   | 1.1.0.beta.5 - microbit- Line tracking car.py  |
|------|--|
| Mode | Image: New Load     Save     Flash     Files     REPL     Plotter     Zoom-in     Zoom-out     Theme |
|      | bit- Line tracking car. py 🗶 🔨   |
| 1    | <pre>from microbit import *</pre>  |
| 2    | from keyes_mecanum_Car import *  |
| 3    | <pre>mecanumCar = Mecanum_Car_Driver()</pre>   |
| 4    | display.show(Image.HAPPY)  |
| 5    | $val_{LL} = 0$   |
| 6    | $val_RR = 0$   |
| 7    | while True:  |
| s    | val_LL = pin1.read_digital()   |
| 9    | <pre>val_RR = pin2.read_digital()</pre>  |
| 10   | <pre>if val_LL == 0 and val_RR == 1:</pre>   |
| 11   | mecanumCar.Motor_Upper_L(1, 220)   |
| 12   | mecanumCar.Motor_Lower_L(1, 220)   |
| 13   | mecanumCar.Motor_Upper_R(0, 120)   |
| 14   | mecanumCar.Motor_Lower_R(0, 120)   |
| 15   | <pre>elif val_LL == 1 and val_RR == 0:</pre>   |
| 16   | mecanumCar.Motor_Upper_L(0, 120)   |
| 17   | mecanumCar.Motor_Lower_L(0, 120)   |
| 18   | <pre>mecanumCar.Motor_Upper_R(1, 220)</pre>  |
| 19   | mecanumCar.Motor_Lower_R(1, 220)   |
| 20   | elif val_LL == 0 and val_RR == 0:  |
| 21   | mecanumCar.Motor_Upper_L(0, 0)   |
| 22   | mecanumCar.Motor_Lower_L(0, 0)   |
| 23   | mecanumCar.Motor_Upper_R(0, 0)   |
| 24   | <pre>mecanumCar.Motor_Lower_R(0, 0) else:</pre>  |
| 25   | mecanumCar.Motor_Upper_L(1, 100)   |
| 26   | mecanumCar.Motor_Lower_L(1, 100)   |
| 27   | mecanumCar.Motor_Upper_R(1, 100)   |
| 28   | mecanumCar.Motor_Lower_R(1, 100)   |
| 29   |  |
| 30   |  |
|      |  |



L.

Click "Check" to examine error in the code. The program proves wrong if

underlines and cursors are shown.

| <b>P</b><br>Kode | Image: New Load     Image: Save     Image: Save |
|------------------|---|
| icrobi           | t- Line tracking car.py 🕱   |
| 1 1              | from microbit import *  |
| 2 1              | from keyes_mecanum_Car import *   |
| s n              | necanumCar = Mecanum_Car_Driver()   |
| 4 0              | display.show(Image.HAPPY)   |
| 5 \              | /al_LL = 0  |
| 6 <b>\</b>       | /al_RR = 0  |
| 7 N              | while True:   |
| S                | val_LL = pin1.read_digital()  |
| 9                | <pre>val_RR = pin2.read_digital()</pre>   |
| 10               | <pre>if val_LL == 0 and val_RR == 1:</pre>  |
| 11               | <pre>mecanumCar.Motor_Upper_L(1, 220)</pre>   |
| 12               | <pre>mecanumCar.Motor_Lower_L(1, 220)</pre>   |
| 13               | mecanumCar.Motor_Upper_R(0, 120)  |
| 14               | mecanumCar.Motor_Lower_R(0, 120)  |
| 15               | <pre>elif val_LL == 1 and val_RR == 0:</pre>  |
| 16               | mecanumCar.Motor_Upper_L(0, 120)  |
| 17               | mecanumCar.Motor_Lower_L(0, 120)  |
| 18               | mecanumCar.Motor_Upper_R(1, 220)  |
| 19               | mecanumCar.Motor_Lower_R(1, 220)  |
| 20               | <pre>elif val_LL == 0 and val_RR == 0:</pre>  |
| 21               | mecanumCar.Motor_Upper_L(0, 0)  |
| 22               | mecanumCar.Motor_Lower_L(0, 0)  |
| 23               | mecanumCar.Motor_Upper_R(0, 0)  |
| 24               | <pre>mecanumCar.Motor_Lower_R(0, 0)</pre>   |
| 25               | else:   |
| 26               | mecanumCar.Motor_Upper_L(1, 100)  |
| 27               | mecanumCar.Motor_Lower_L(1, 100)  |
| 28               | mecanumCar.Motor_Upper_R(1, 100)  |
| 29               | <pre>mecanumCar.Motor_Lower_R(1, 100)</pre>   |



If the code is correct, connect micro:bit to computer and click "Flash" to

download code to micro:bit board.

| C     |  |
|-------|--|
| Mode  | e New Load Save Flash Files REPL Plotter Zoom-in Zoom-out Theme C    |
| micro | bit- Line tracking car.py 🗶  |
| 1     | <pre>from microbit import *</pre>                                    |
| 2     | <pre>from keyes_mecanum_Car import *</pre>                           |
|       | <pre>mecanumCar = Mecanum_Car_Driver()</pre>                         |
| 4     | display.show(Image.HAPPY)  |
|       | -  |
|       | val_RR = 0   |
| 7     | while True:  |
| 8     | <pre>val_LL = pin1.read_digital() val_RR = pin2.read_digital()</pre> |
| 9     | if val_LL == 0 and val_RR == 1:                                      |
| 10    | mecanumCar.Motor_Upper_L(1, 220)                                     |
| 11    | mecanumCar.Motor_Lower_L(1, 220)                                     |
| 13    | mecanumCar.Motor_Upper_R(0, 120)                                     |
| 14    | mecanumCar.Motor_Lower_R(0, 120)                                     |
| 15    | <pre>elif val_LL == 1 and val_RR == 0:</pre>                         |
| 15    | mecanumCar.Motor_Upper_L(0, 120)                                     |
| 17    | mecanumCar.Motor_Lower_L(0, 120)                                     |
| 18    | <pre>mecanumCar.Motor_Upper_R(1, 220)</pre>                          |
| 19    | mecanumCar.Motor_Lower_R(1, 220)                                     |
| 20    | <pre>elif val_LL == 0 and val_RR == 0:</pre>                         |
| 21    | <pre>mecanumCar.Motor_Upper_L(0, 0)</pre>                            |
| 22    | <pre>mecanumCar.Motor_Lower_L(0, 0)</pre>                            |
| 23    | <pre>mecanumCar.Motor_Upper_R(0, 0)</pre>                            |
| 24    | <pre>mecanumCar.Motor_Lower_R(0, 0)</pre>                            |
| 25    | else:  |
| 26    | mecanumCar.Motor_Upper_L(1, 100)                                     |
| 27    | mecanumCar.Motor_Lower_L(1, 100)                                     |
| 28    | mecanumCar.Motor_Upper_R(1, 100)                                     |
| 29    | mecanumCar.Motor_Lower_R(1, 100)                                     |



### (5)Test Results:

Download code to micro:bit and dial POWER to ON end, line tacking car

goes forward along black line .

Note: turn on the switch at the back of micro:bit car.

the width of black line should be larger than the width of line tracking sensor.

Avoid to test smart car under the strong light.

#### microbit import \* from Import the library of micro:bit from keyes mecanum Car Import the library of import \* keyes mecanum Car mecanumCar = Instantiate an object Mecanum Car Driver() Mecanum Car Driver() as mecanumCar display.show(Image.HAPPY) The LED dot matrix on the micro:bit displays a "smile" pattern while True: This is a permanent loop that makes micro:bit execute the code of it. val LL = pin1.read digital()Assign the digital signal read by the TCRT5000 infrared pair tube

# (6)Code Explanation:

connected to the P1 control port to



|                              | the variable val_LL                   |
|------------------------------|---------------------------------------|
| val_RR = pin2.read_digital() | Assign the digital signal read by the |
|                              | TCRT5000 infrared pair tube           |
|                              | connected to the P2 control port to   |
|                              | the variable val_RR                   |



| if val_LL == 0 and val_RR == 1:  | if val_LL  |
|--|--|
| mecanumCar.Motor_Upper_L(1, 220)   | the front  |
| mecanumCar.Motor_Lower_L(1, 220)   | clockwise  |
| mecanumCar.Motor_Upper_R(0, 120)   | the rear l   |
| mecanumCar.Motor_Lower_R(0, 120)   | clockwise  |
| elif val_LL == 1 and val_RR == 0:  | the front  |
| mecanumCar.Motor_Upper_L(0, 120)   | anticlock  |
| mecanumCar.Motor_Lower_L(0, 120)   | PWM120   |
| mecanumCar.Motor_Upper_R(1, 220)   | the rear r   |
| mecanumCar.Motor_Lower_R(1, 220)   | anticlock  |
|  |  |
| elif val_LL == 0 and val_RR == 0:  | PWM120   |
| <pre>elif val_LL == 0 and val_RR == 0: mecanumCar.Motor_Upper_L(0, 0)</pre>  | PWM120 if val_LL   |
|  |  |
| <br>mecanumCar.Motor_Upper_L(0, 0)   | if val_LL  |
| mecanumCar.Motor_Upper_L(0, 0)<br>mecanumCar.Motor_Lower_L(0, 0)   | <b>if val_LL</b><br>the front  |
| mecanumCar.Motor_Upper_L(0, 0)<br>mecanumCar.Motor_Lower_L(0, 0)<br>mecanumCar.Motor_Upper_R(0, 0)   | <b>if val_LL</b><br>the front<br>anticlock   |
| mecanumCar.Motor_Upper_L(0, 0)<br>mecanumCar.Motor_Lower_L(0, 0)<br>mecanumCar.Motor_Upper_R(0, 0)<br>mecanumCar.Motor_Lower_R(0, 0)   | <b>if val_LL</b><br>the front<br>anticlock<br>PWM120                               |
| mecanumCar.Motor_Upper_L(0, 0)<br>mecanumCar.Motor_Lower_L(0, 0)<br>mecanumCar.Motor_Upper_R(0, 0)<br>mecanumCar.Motor_Lower_R(0, 0)<br>else:  | <b>if val_LL</b><br>the front<br>anticlock<br>PWM120<br>the rear l                 |
| mecanumCar.Motor_Upper_L(0, 0)<br>mecanumCar.Motor_Lower_L(0, 0)<br>mecanumCar.Motor_Upper_R(0, 0)<br>mecanumCar.Motor_Lower_R(0, 0)<br>else:<br>mecanumCar.Motor_Upper_L(1, 100)  | if val_LL<br>the front<br>anticlock<br>PWM120<br>the rear l<br>anticlock           |
| mecanumCar.Motor_Upper_L(0, 0)<br>mecanumCar.Motor_Lower_L(0, 0)<br>mecanumCar.Motor_Upper_R(0, 0)<br>mecanumCar.Motor_Lower_R(0, 0)<br><b>else:</b><br>mecanumCar.Motor_Upper_L(1, 100)<br>mecanumCar.Motor_Lower_L(1, 100) | if val_LL<br>the front<br>anticlock<br>PWM120<br>the rear l<br>anticlock<br>PWM120 |

= 0 and val\_RR = 1 is true, t left motor of car rotates e at the speed of PWM220; left motor of car rotates e at the speed of PWM220; t right motor of car rotates wise at the speed of ); right motor of car rotates wise at the speed of 0; = 1 and val RR = 0 is true, t left motor of car rotates wise at the speed of 0; left motor of car rotates wise at the speed of );

the front right motor of car rotates clockwise at the speed of PWM220; the rear right motor of car rotates clockwise at the speed of PWM220;

if val LL = 0 and val RR = 0 is true, the front left motor of car rotates at the speed of 0 and stops; the rear left motor of car rotates at the speed of 0 and stops; the front right motor of car rotates clockwise at the speed of 0 and stops; the rear right motor of car rotates clockwise at the speed of 0 and stops; When none of the above conditions are met,

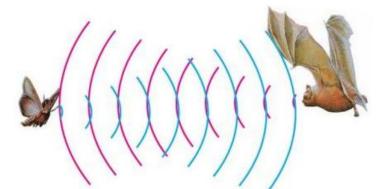
the front left motor of car rotates clockwise at the speed of PWM100; the rear left motor of car rotates clockwise at the speed of PWM100; the front right motor of car rotates clockwise at the speed of PWM100; the rear right motor of car rotates clockwise at the speed of PWM100;



### **Project 18: Ultrasonic Following Smart Car**

### 18.1: Ultrasonic Ranging

#### (1)Project Description



The ultrasonic sensor uses sonar to determine distance to an object like bats do. It offers excellent non-contact range detection with high accuracy and stable readings

in an easy-to-use package. It comes complete with ultrasonic transmitter and receiver modules.

The ultrasonic sensor is being used in a wide range of electronics projects for creating obstacle detection and distance measuring application as well as various other applications.

As the above picture shown, it is like two eyes. One is transmitting end, the other is receiving end.

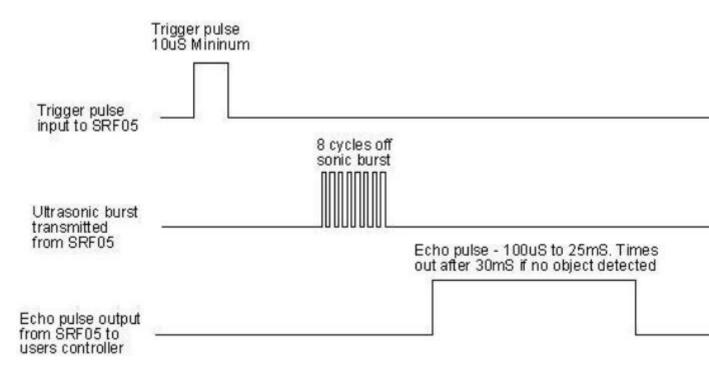
The ultrasonic module will emit the ultrasonic waves after trigger signal. When the ultrasonic waves encounter the object and are reflected back, the module outputs an echo signal, so it can determine the distance of object from the time difference between trigger signal (TRIG)and echo signal(ECHO).





According to the above wiring diagram, the integrated port of the ultrasonic sensor module is connected to the 5V G P15 P16 port on the micro:bit motor drive backplane. The Trig (T) pin is controlled by P15 of the micro:bit and the pin of Echo (E) the P16.

#### (2)Working Principle:



(1)Pull down TRIG then trigger high level signals with least 10us;

(2)After triggering, the module will automatically send eight 40KHz



ultrasonic pulses and detect whether there is a signal return;

(3)The propagation speed of sound in the air is about 340m/s, therefore, distance = speed \* time, because the ultrasonic wave emits and comes back, which is 2 times of distance, so it needs to be divided by 2, the distance measured by ultrasonic wave = (speed \* time)/2.

### (3)Parameters:

- Working voltage: 3-5.5V (DC)
- Working current: 15mA
- Working frequency: 40khz
- Maximum detection distance: about 3m
- Minimum detection distance: 2-3cm
- Precision: up to 0.2cm
- Sensing angle: less than 15 degrees
- Input trigger pulse: 10us TTL level
- Output echo signal: output TTL level signal (high), proportional to range

### (4) Experimental Preparation:

- > Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car
- > Place batteries into battery holder
- > Dial power switch to ON end
- Connect micro:bit to computer by USB cable



> Open the offline version of Mu.

## (5)Test Code:

Enter Mu software and open the file "Project 18: Ultrasonic Following Smart

Car.py" to import code:

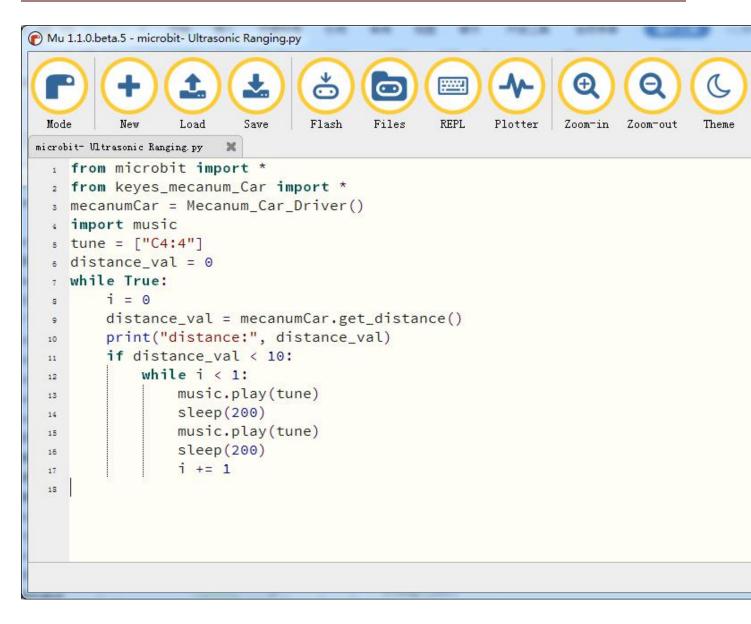
## (How to load the project code?)

| File   | Route                       | File Name             |
|--------|-----------------------------|-----------------------|
| Туре   |                             |                       |
| Python | KS4031(KS4032)              | Ultrasonic Ranging.py |
| file   | folder/Python               |                       |
|        | Tutorial/Python             |                       |
|        | Code/Project Code/Project   |                       |
|        | 18 : Ultrasonic Following   |                       |
|        | Smart Car/18.1 : Ultrasonic |                       |
|        | Ranging                     |                       |

You can also input code in the editing window yourself.

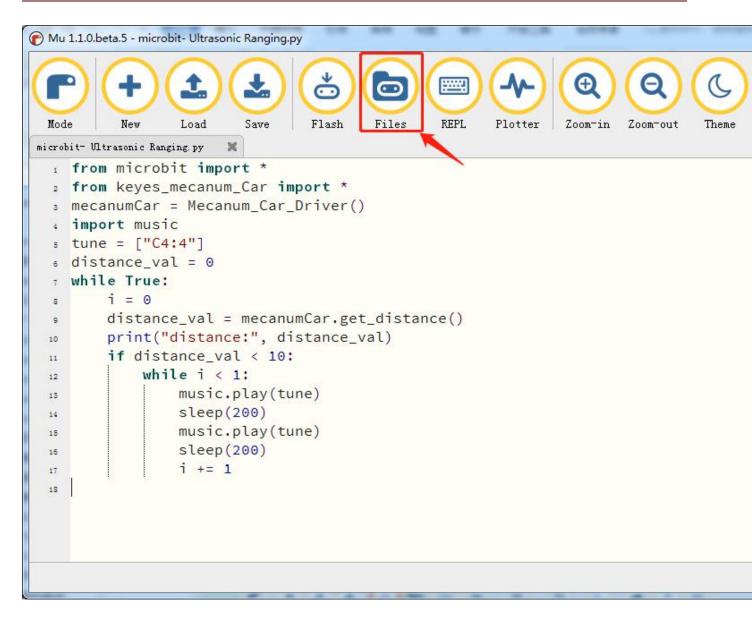
(note:all words and symbols must be written in English)





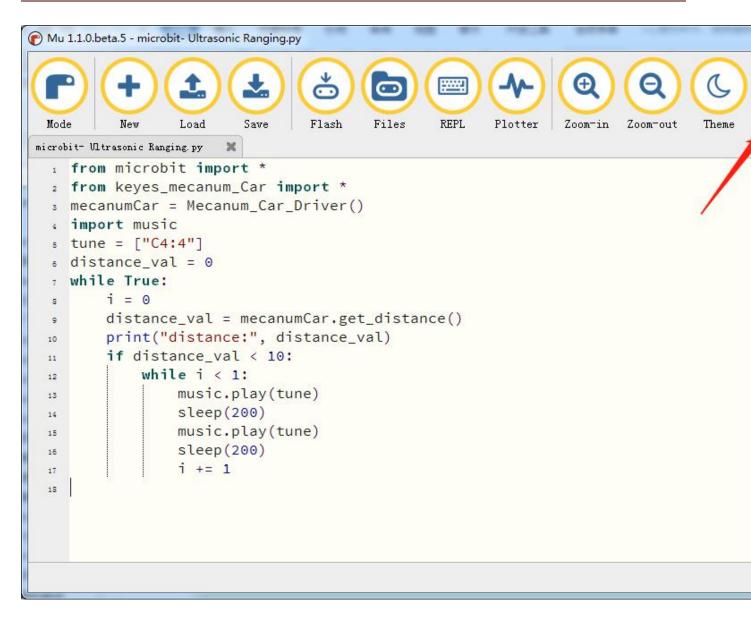
Click "Files" to import "keyes\_mecanum\_Car.py "library file tomicro:bit (<u>How</u> <u>to import files?</u>). No need to do it again if you have imported it before.





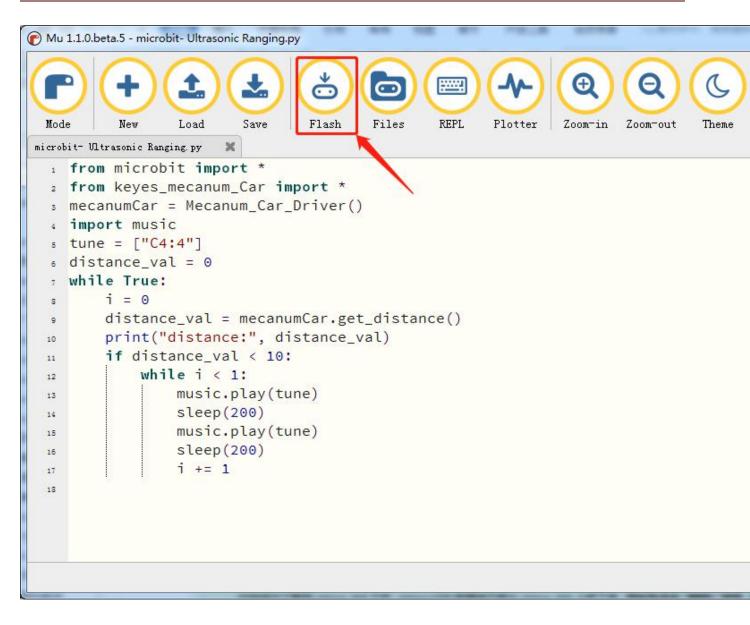
Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.





If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.



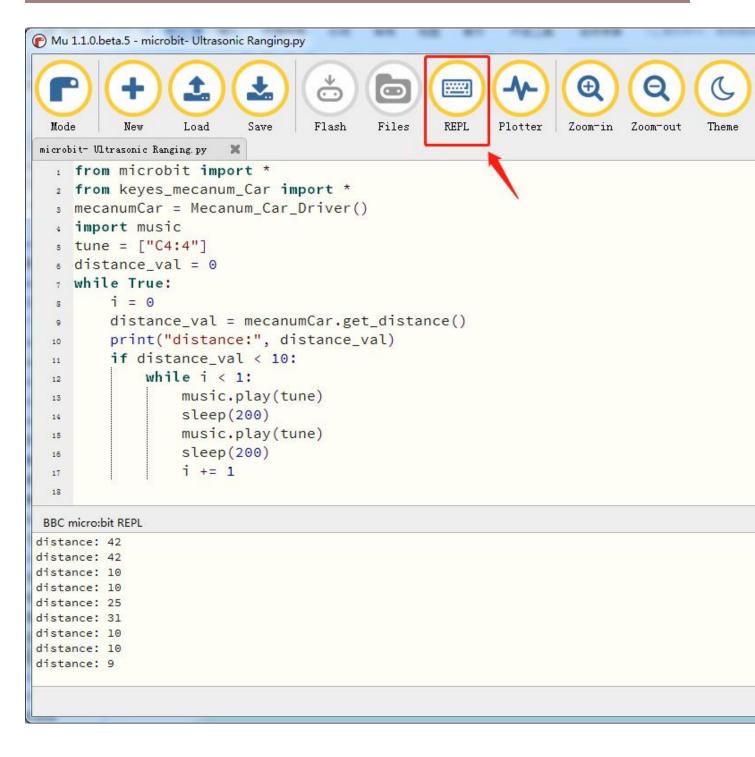


#### (6)Test Results:

Download code onto micro:bit board, don' t plug off USB cable. Click "REPL" and press the reset buttons, the distance value of obstacle will be displayed, as shown below.

When the distance is less than 10cm, the passive buzzer of smart car emits sound.





## (7)Code Explanation:

| from microbit import *        | Import the library of micro:bit |
|-------------------------------|---------------------------------|
| from keyes_mecanum_Car import | Import the library of           |



| *   |                                       |
|---|---------------------------------------|
| *   | keyes_mecanum_Car                     |
| mecanumCar =                                | instantiate Mecanum_Car_Driver()() to |
| Mecanum_Car_Driver()()                      | mecanumCar                            |
| import music                                | Import the library of music           |
| tune = ["C4:4"]                             | Create tune to save                   |
| while True:                                 | This is a permanent loop that makes   |
|   | micro:bit execute the code of it.     |
| i = 0                                       | Set variable i=0                      |
| distance_val =                              | Assign mecanumCar.get_distance()to    |
| mecanumCar.get_distance()                   | variable distance_val                 |
| <pre>print("distance:", distance_val)</pre> | BBC microbit REPL window shows the    |
|   | distance value between the ultrasonic |
|   | sensor and the obstacle               |
| if distance < 10:                           | if distance < 10                      |
| <b>while</b> i < 1:                         | When i < 1                            |
| music.play(tune)                            |                                       |
| sleep(200)                                  | Passive buzzer emits sound            |
| music.play(tune)                            |                                       |
| sleep(200)                                  |                                       |
| i += 1                                      | Variable i adds 1 gradually           |



# 18.2: Ultrasonic Avoidance Car



#### (1)Project Description

We' ve learned the knowledge of obstacle avoidance sensor. In this project, we will integrate ultrasonic sensor, and car expansion board to make an ultrasonic avoidance car.

Its principle is to detect the distance between the car and obstacle by ultrasonic sensor and control the motion of smart car.

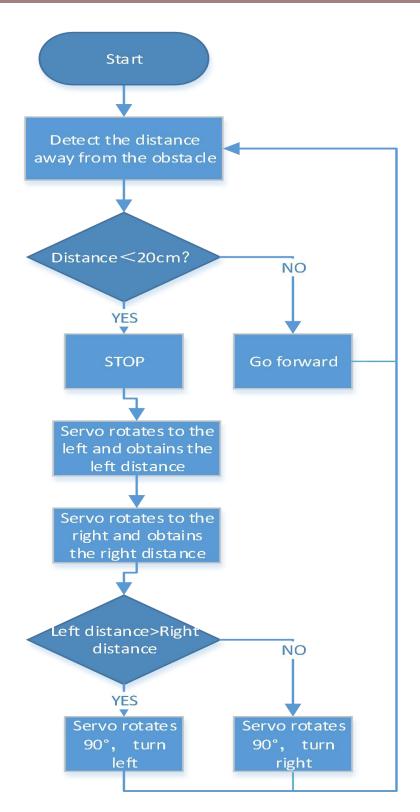


# (2) Experimental Preparation:

- > Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car
- > Place batteries into battery holder
- > Dial power switch to ON end
- Connect micro:bit to computer by USB cable
- > Open the offline version of Mu.

## (3)Flow Chart:





## (4)Test Code:

Enter Mu software and open the file "Project 18: Ultrasonic Following Smart

Car.py" to import code:



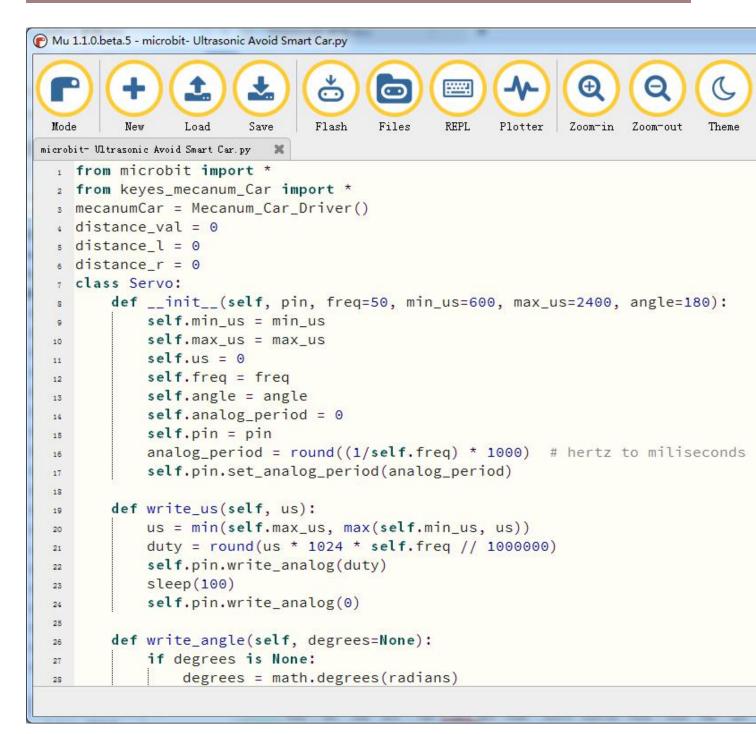
# (How to load the project code?)

| File   | Route                       | File Name                   |
|--------|-----------------------------|-----------------------------|
| Туре   |                             |                             |
| Python | KS4031(KS4032)              | Ultrasonic Avoidance Car.py |
| file   | folder/Python               |                             |
|        | Tutorial/Python             |                             |
|        | Code/Project 18: Ultrasonic |                             |
|        | Following Smart Car/18.2 :  |                             |
|        | Ultrasonic Avoidance Car    |                             |
|        |                             |                             |

You can also input code in the editing window yourself.

(note:all words and symbols must be written in English)







```
degrees = degrees % 360
29
           total_range = self.max_us - self.min_us
30
           us = self.min_us + total_range * degrees // self.angle
31
           self.write_us(us)
32
33
  Servo(pin14).write_angle(90)
34
35
  while True:
36
37
       distance_val = mecanumCar.get_distance()
38
39
       if distance val < 20:
40
           mecanumCar.Motor_Upper_L(0, 0)
41
           mecanumCar.Motor_Lower_L(0, 0)
42
           mecanumCar.Motor_Upper_R(0, 0)
43
           mecanumCar.Motor_Lower_R(0, 0)
44
           sleep(500)
45
           Servo(pin14).write_angle(180)
46
           sleep(500)
47
           distance_l = mecanumCar.get_distance()
48
           sleep(500)
49
50
           Servo(pin14).write_angle(0)
51
           sleep(500)
52
           distance_r = mecanumCar.get_distance()
53
           sleep(500)
54
55
           if distance_l > distance_r:
56
```

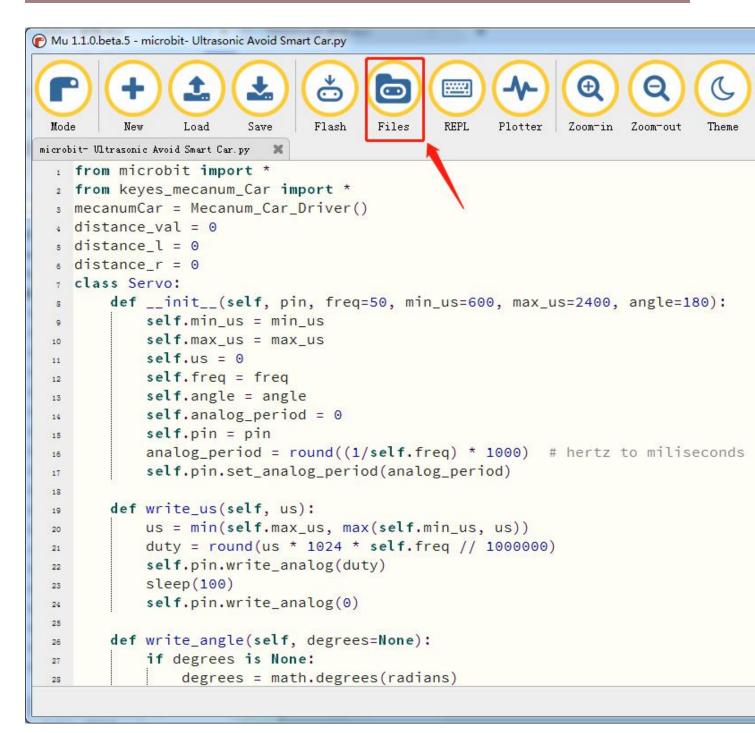


| 57 | <pre>mecanumCar.Motor_Upper_L(0, 150)</pre> |
|----|---|
| 58 | <pre>mecanumCar.Motor_Lower_L(0, 150)</pre> |
| 59 | <pre>mecanumCar.Motor_Upper_R(1, 150)</pre> |
| 50 | <pre>mecanumCar.Motor_Lower_R(1, 150)</pre> |
| 61 | Servo(pin14).write_angle(90)                |
| 62 | sleep(300)                                  |
| 63 | else:                                       |
| 64 | <pre>mecanumCar.Motor_Upper_L(1, 150)</pre> |
| 65 | <pre>mecanumCar.Motor_Lower_L(1, 150)</pre> |
| 66 | <pre>mecanumCar.Motor_Upper_R(0, 150)</pre> |
| 67 | <pre>mecanumCar.Motor_Lower_R(0, 150)</pre> |
| 68 | Servo(pin14).write_angle(90)                |
| 69 | sleep(300)                                  |
| 70 |   |
| 71 | else:                                       |
| 72 | <pre>mecanumCar.Motor_Upper_L(1, 150)</pre> |
| 73 | <pre>mecanumCar.Motor_Lower_L(1, 150)</pre> |
| 74 | <pre>mecanumCar.Motor_Upper_R(1, 150)</pre> |
| 75 | <pre>mecanumCar.Motor_Lower_R(1, 150)</pre> |
| 76 | 82  |
|    |   |

Click "Files" to import "keyes\_mecanum\_Car.py "library file tomicro:bit (How

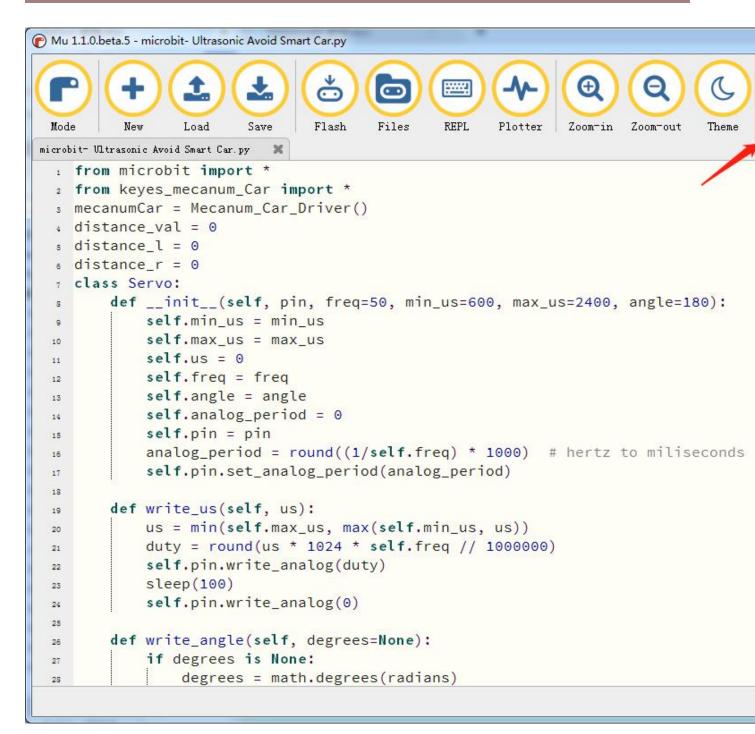
to import files? ). No need to do it again if you have imported it before.





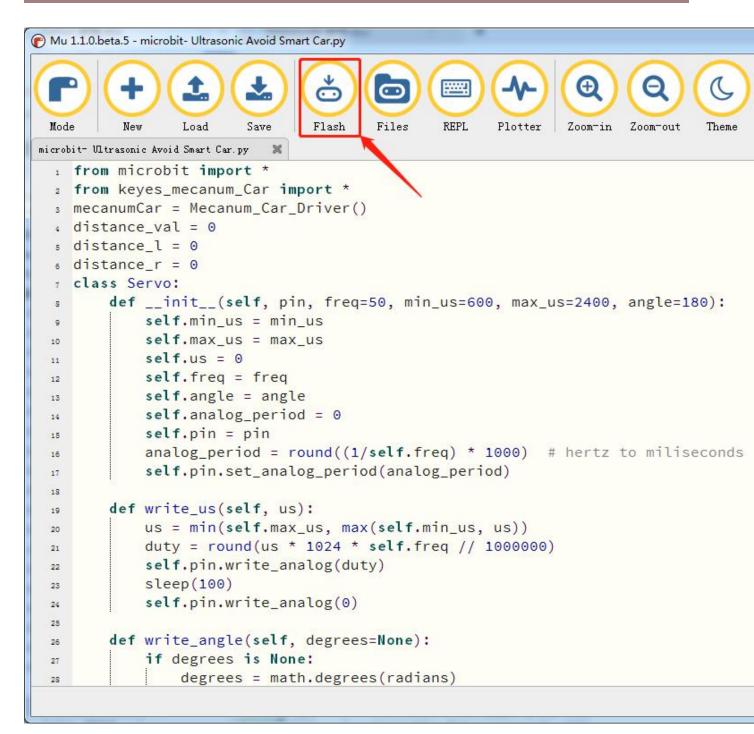
Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.





If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.





#### (5)Test Results:

Download code to micro:bit, dial to ON end, and dial POWER to ON end. When the obstacle distance is greater than 20cm, the car goes forward ; on the contrary, smart car turns left.



# (6)Code Explanation:

| from microbit import *         | Import the library of micro:bit       |
|--------------------------------|---------------------------------------|
| from keyes_mecanum_Car import  | Import the library of                 |
| *                              | keyes_mecanum_Car                     |
| mecanumCar =                   | Instantiate Mecanum_Car_Driver()()to  |
| Mecanum_Car_Driver()()         | mecanumCar                            |
| distance_val = 0               | Set the initial value of the variable |
|                                | distance_val to 0                     |
| distance_l = 0                 | Set the initial value of the variable |
|                                | distance_l to 0                       |
| distance_r = 0                 | Set the initial value of the variable |
|                                | distance_r to 0                       |
| Servo(pin14).write_angle(90)   | The steering gear is connected to     |
|                                | P14, the rotation angle is 90 degrees |
| while True:                    | This is a permanent loop that makes   |
|                                | micro:bit execute the code of it.     |
| distance_val =                 | Assign mecanumCar.get_distance()to    |
| mecanumCar.get_distance()      | variable distance_val                 |
| if distance_val < 20:          | If distance_val <20 is established    |
| mecanumCar.Motor_Upper_L(0, 0) | (stop)                                |
| mecanumCar.Motor_Lower_L(0, 0) | The left front motor of the Mecanum   |



| mecanumCar.Motor_Upper_R(0, 0) | wheel smart car stops rotating;        |
|--------------------------------|--|
| mecanumCar.Motor_Lower_R(0, 0) | The left rear motor of the Mecanum     |
| sleep(500)                     | wheel smart car stops rotating;        |
| Servo(pin14).write_angle(180)  | The right front motor of the           |
| distance_l =                   | Mecanum wheel smart car stops          |
| mecanumCar.get_distance()      | rotating;                              |
| Servo(pin14).write_angle(0)    | The right rear motor of the Mecanum    |
| distance_r =                   | wheel smart car stops rotating;        |
| mecanumCar.get_distance()      | Delay in 500ms                         |
| if distance_l > distance_r:    | The servo connected to P14 rotates     |
| mecanumCar.Motor_Upper_L(0,    | to 180 degrees;                        |
| 150)                           | Assign mecanumCar.get_distance() to    |
| mecanumCar.Motor_Lower_L(0,    | the variable distance_l;               |
| 150)                           | The servo connected to P14 turns to 0  |
| mecanumCar.Motor_Upper_R(1,    | degrees;                               |
| 150)                           | Assign mecanumCar.get_distance() to    |
| mecanumCar.Motor_Lower_R(1,    | the variable distance_r;               |
| 150)                           | If distance_l> distance_r condition is |
| Servo(pin14).write_angle(90)   | true (turn left)                       |
| sleep(300)                     | The front left motor of car rotates    |
| else:                          | clockwise at the speed of PWM150.      |
| mecanumCar.Motor_Upper_L(1,    | The rear left motor of car rotates     |



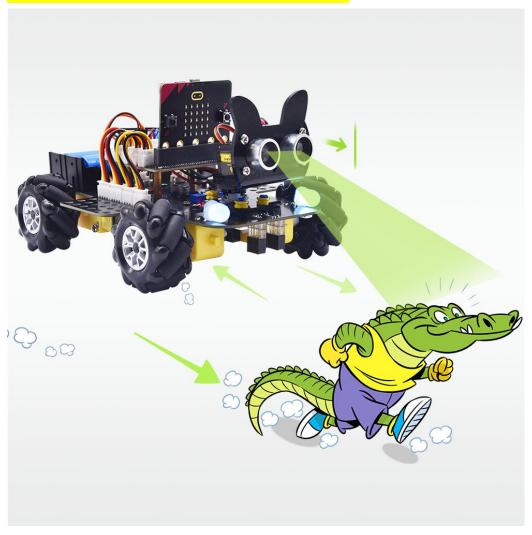
| 150)                         | clockwise at the speed of PWM150.      |
|------------------------------|--|
| mecanumCar.Motor_Lower_L(1,  | The front right motor of car rotates   |
| 150)                         | anticlockwise at the speed of          |
| mecanumCar.Motor_Upper_R(0,  | PWM150.                                |
| 150)                         | The rear right motor of car rotates    |
| mecanumCar.Motor_Lower_R(0,  | anticlockwise at the speed of          |
| 150)                         | PWM150.                                |
| Servo(pin14).write_angle(90) | The servo connected to P14 turns to    |
| sleep(300)                   | 90 degrees;                            |
| else:                        | Delay in 300ms;                        |
| mecanumCar.Motor_Upper_L(1,  | If distance_l> distance_r condition is |
| 150)                         | not true (turn right),                 |
| mecanumCar.Motor_Lower_L(1,  | The front left motor of car rotates    |
| 150)                         | anticlockwise at the speed of          |
| mecanumCar.Motor_Upper_R(1,  | PWM150.                                |
| 150)                         | The rear left motor of car rotates     |
| mecanumCar.Motor_Lower_R(1,  | anticlockwise at the speed of          |
| 150)                         | PWM150.                                |
|                              | The front right motor of car rotates   |
|                              | clockwise at the speed of PWM150.      |
|                              | The rear right motor of car rotates    |
|                              | clockwise at the speed of PWM150.      |



| The servo connected to P14 turns to  |
|--------------------------------------|
| 90 degrees;                          |
| Delay in 300ms;                      |
| If condition distance_val <20 is not |
| true (move forward)                  |
| The front left motor of car rotates  |
| clockwise at the speed of PWM150.    |
| The rear left motor of car rotates   |
| clockwise at the speed of PWM150.    |
| The front right motor of car rotates |
| clockwise at the speed of PWM150.    |
| The rear right motor of car rotates  |
| clockwise at the speed of PWM150.    |
| The servo connected to P14 turns to  |
| 90 degrees;                          |
| Delay in 300ms;                      |
|                                      |
|                                      |



# 18.3: Ultrasonic Following Smart Car



## (1) Description:

In previous lesson, we' ve learned the basic principle of line tracking sensor. Next, we will combine ultrasonic sensor with car shield to make an ultrasonic follow car.

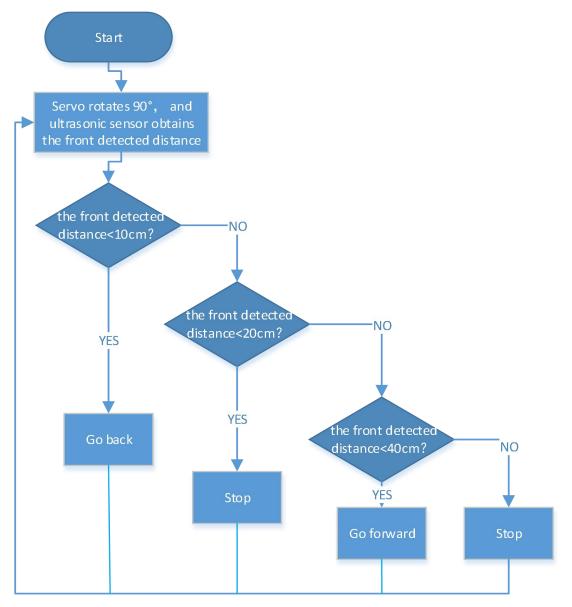
The ultrasonic sensor detects the obstacle distance and control the motion status of car.

#### (2) Experimental Preparation:



- > Insert micro:bit board into slot of keyestudio 4WD Mecanum Robot Car
- > Place batteries into battery holder
- > Dial power switch to ON end
- Connect micro:bit to computer by USB cable
- > Open the offline version of Mu.

## (3)Flow Chart:





# (4)Test Code:

Enter Mu software and open the file "18.3: Ultrasonic Follow Smart Car.py"

to import code:

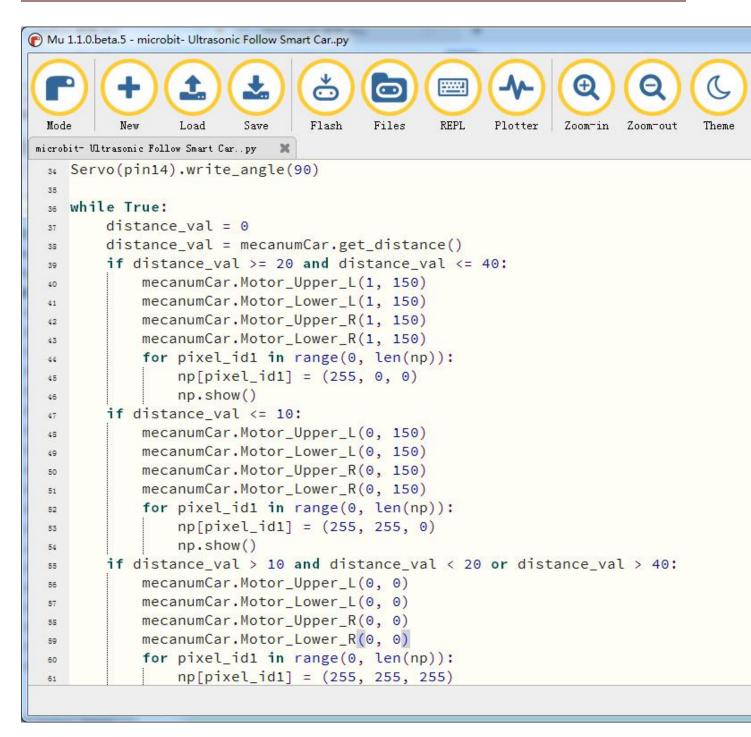
(How to load the project code?)

| File   | Route                       | File Name                       |
|--------|-----------------------------|---------------------------------|
| Туре   |                             |                                 |
| Python | KS4031(KS4032)              | Ultrasonic Follow Smart Carr.py |
| file   | folder/Python               |                                 |
|        | Tutorial/Python             |                                 |
|        | Code/Project Code/Project   |                                 |
|        | 18 : Ultrasonic Following   |                                 |
|        | Smart Car/18.3 : Ultrasonic |                                 |
|        | Follow Smart Car            |                                 |
|        |                             |                                 |

You can also input code in the editing window yourself.

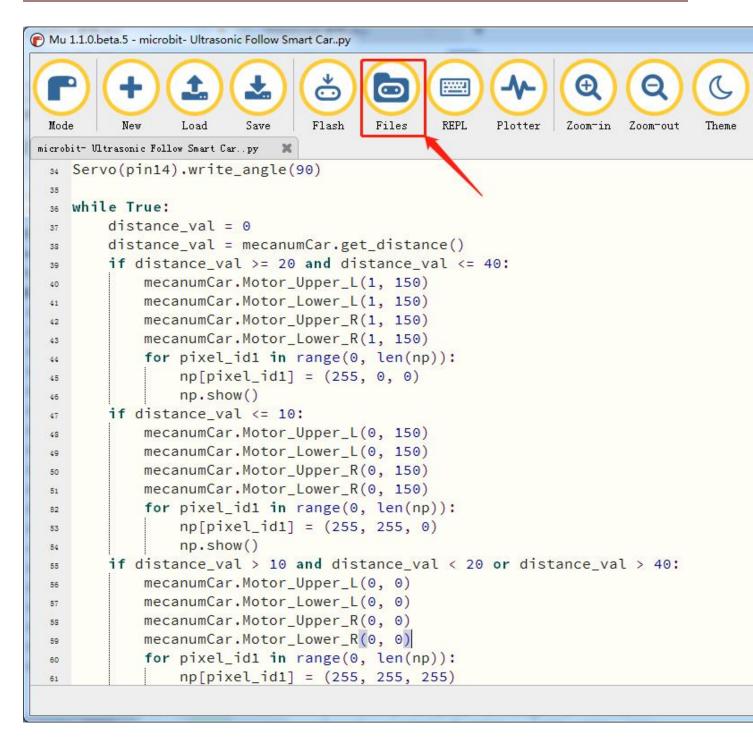
(note:all words and symbols must be written in English)





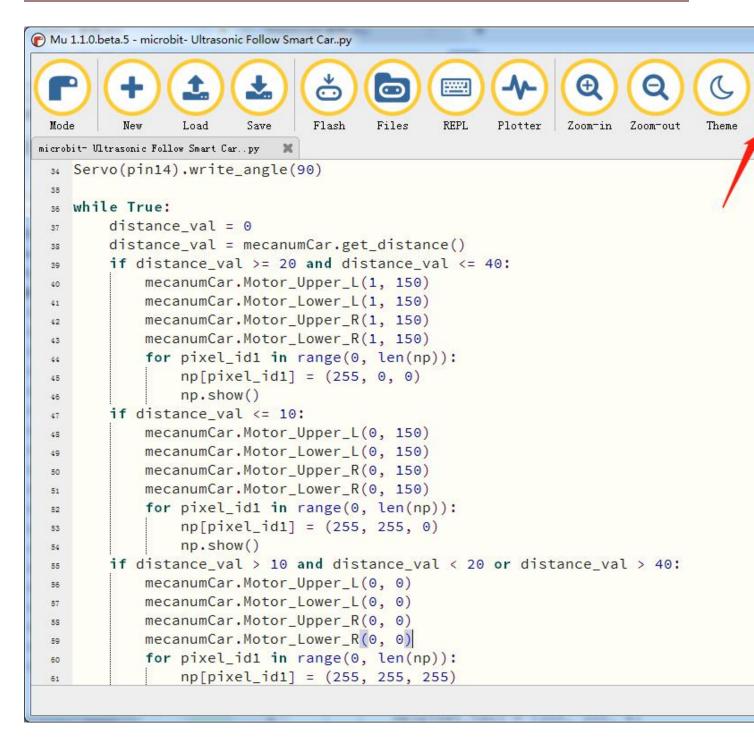
Click "Files" to import "keyes\_mecanum\_Car.py "library file tomicro:bit (<u>How</u> <u>to import files?</u>). No need to do it again if you have imported it before.





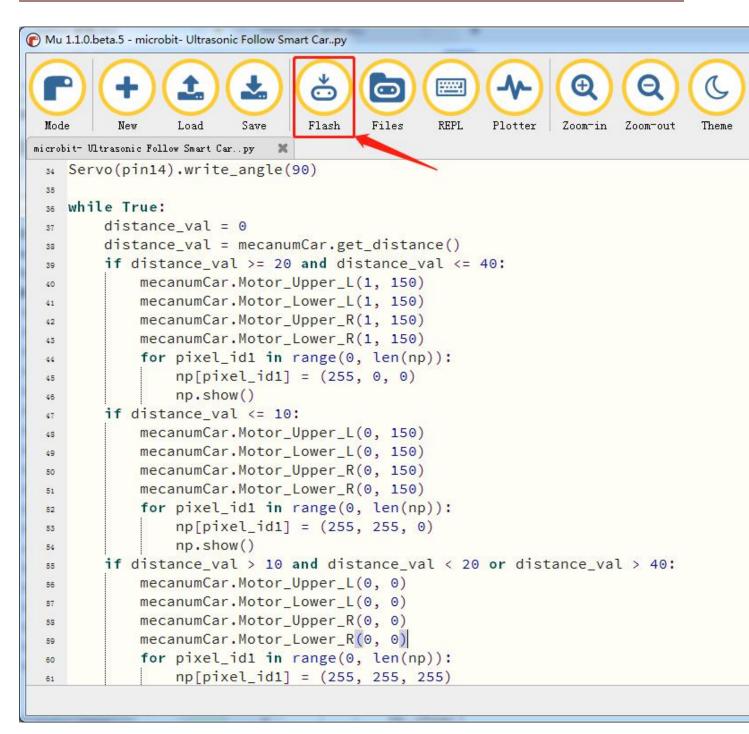
Click "Check" to examine error in the code. The program proves wrong if underlines and cursors are shown.





If the code is correct, connect micro:bit to computer and click "Flash" to download code to micro:bit board.





## (5)Test Results:

Download code to micro:bit, dial POWER switch to ON end on shield, smart car could follow the obstacle to move and WS2812 RGB lights will display different colors.



Note: the obstacle only moves in front of smart car, not turning around.

# 6.Code Explanation:

| from microbit import *                     | Import the library of micro:bit        |
|--|--|
| <pre>from keyes_mecanum_Car import *</pre> | Import the library of                  |
|  | keyes_mecanum_Car                      |
| mecanumCar =                               | Instantiate                            |
| Mecanum_Car_Driver()()                     | Mecanum_Car_Driver()()to               |
|  | mecanumCar                             |
| import neopixel                            | Import the library of neopixel         |
| np = neopixel.NeoPixel(pin8, 4)            | Set Neopixel to pin P8, and initialize |
|  | 4 LEDs                                 |
| Servo(pin14).write_angle(90)               | The servo connected to P14 pin         |
|  | turns to 90 degrees                    |
| while True:                                | This is a permanent loop that makes    |
|  | micro:bit execute the code of it.      |
| distance_val = 0                           | Set the initial value of the variable  |
|  | distance_val to 0                      |
| distance_val =                             | Assign mecanumCar.get_distance()       |
| mecanumCar.get_distance()                  | to the variable distance_val           |
| if distance_val > = 20 and                 | If distance_val ≥20 and distance_val   |
| L  |  |



| ≤ 40 are true,                       |
|--------------------------------------|
| The car moves forward.               |
|                                      |
| RGB pixels in the range of (0,       |
| len(np)) are pixel_id1               |
| Set pixel_id1 to light up red        |
| Display pixels on Neopixel strip     |
| If distance_val ≤10 holds            |
| The car moves back.                  |
|                                      |
|                                      |
| RGB pixels in the range of (0,       |
| len(np)) are pixel_id1               |
| Set pixel_id1 to bright yellow light |
| Display pixels on Neopixel strip     |
| If distance_val>10 and               |
| distance_val<20 or distance_val>40   |
| is true                              |
| The car stops.                       |
|                                      |
| RGB pixels in the range of (0,       |
| len(np)) are pixel_id1               |
|                                      |



| mecanumCar.Motor_Upper_R(0, 0)                    | Set pixel_id1 to bright white light |
|---|-------------------------------------|
| mecanumCar.Motor_Lower_R(0, 0)                    | Display pixels on Neopixel strip    |
| <b>for</b> pixel_id1 <b>in</b> range(0, len(np)): |                                     |
| np[pixel_id1] = (255, 255, 255)                   |                                     |
| np.show()   |                                     |

#### 9. Resources

Download PDF files: <u>https://fs.keyestudio.com/KS4031-4032</u>

BBC microbit MicroPython:

https://microbit-micropython.readthedocs.io/en/latest/tutorials/introducti

on.html

MicroPython:

https://docs.openmv.io/reference/index.html

ustruct library:

https://docs.openmv.io/library/ustruct.html

math library:

https://docs.openmv.io/library/math.html

utime(sleep\_us,tick\_us) library:

https://docs.openmv.io/library/utime.html#