Route plan

Learning goal：

This lesson learns how to control car running path by Python programming.

Code:

Please use the MU software to open the **Route plan.py** file we provided.

1) Import the library used in this lesson from micro:bit, display for dot matrix display, Image calls built-in image, sleep delay time, button_a button A, button_b button B, tinybit control car;

display.show (Image.HAPPY): display a smile on micro:bit dot matrix;

show_L = Image ("90000: 90000: 90000: 90000: 99999"): set the dot matrix to display the letter 'L', each set of numbers represents a row of LEDs on the dot matrix, each number represents an LED light on the dot matrix. You can set the brightness of the LED by entering a value from 0 to 9, where 0 is the darkest (off) and 9 is the brightest;

The variable **a** is used to display path number, and the variable **b** is used to enable the car movement.

speed_run and **speed_spin** are the speed values of the car running and rotating, which can be adjusted according to the actual situation.

Code as shown below:

```
from microbit import sleep, display, Image, button_a, button_b
import tinybit

display.show(Image.HAPPY)
show_L = Image("90000:90000:90000:90000:99999")
show_O = Image("09990:90009:90009:90009:09990")
show_D = Image("99000:90900:90090:90009:99999")
show_Z = Image("99999:00090:00900:09000:99999")

a = 0
b = 0

speed_run = 100
speed_spin = 80
```

2) Customize the **run_L ()** function, running track of car is an L-shaped path. Due to the differences between the cars, there may be some differences in the experimental phenomenon. The user can adjust the speed or time of the car to achieve better results according to the actual situation;

```
def run_L():
    display.show(show_L)
    global b
    if b == 1:
        sleep(1000)
        tinybit.car_run(speed_run)
        sleep(1000)
        tinybit.car_spinleft(speed_spin)
        sleep(400)
        tinybit.car_run(speed_run)
        sleep(1000)
        tinybit.car_stop()
        global b
        b = 0
```

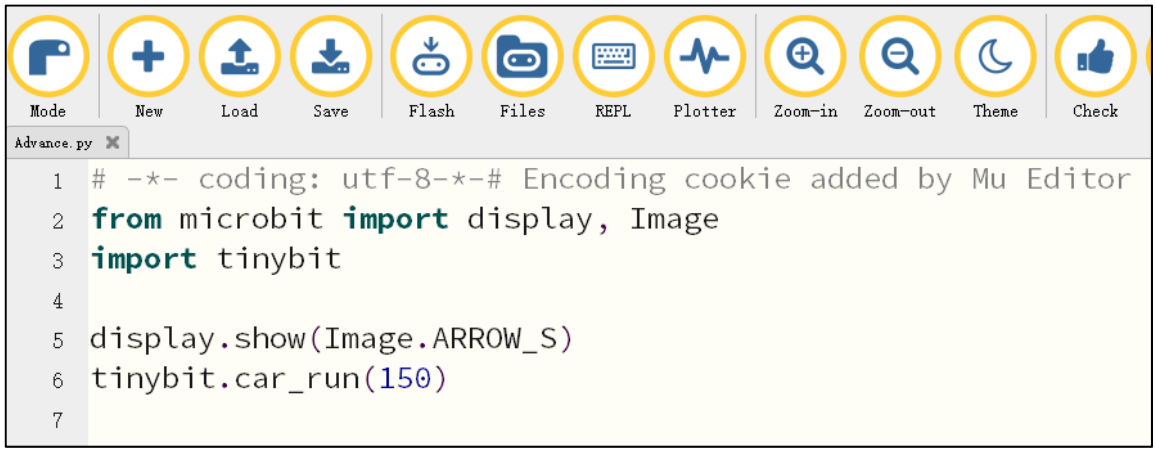
3) Detection of whether button A and button B are pressed, button A switches the planned path, and displays the corresponding path on the dot matrix, and button B starts the planned path of the car operation.

```
while True:
    if button_a.was_pressed():
        a = a + 1
        if a >= 5:
            a = 1
    if button_b.was_pressed():
        b = 1
    if a == 1:
        run_L()
    elif a == 2:
        run_O()
    elif a == 3:
        run_D()
    elif a == 4:
        run_Z()
```

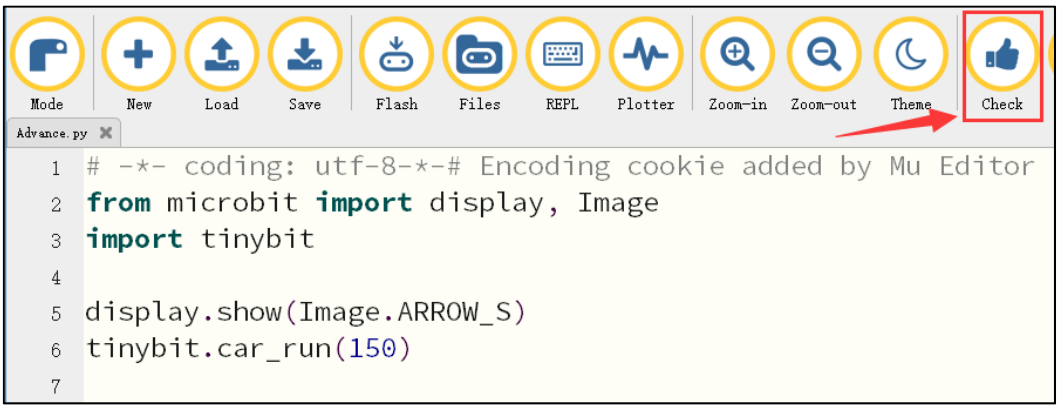
Programming and downloading：

1.You should open the Mu software, and enter the code in the edit window, , as shown below.

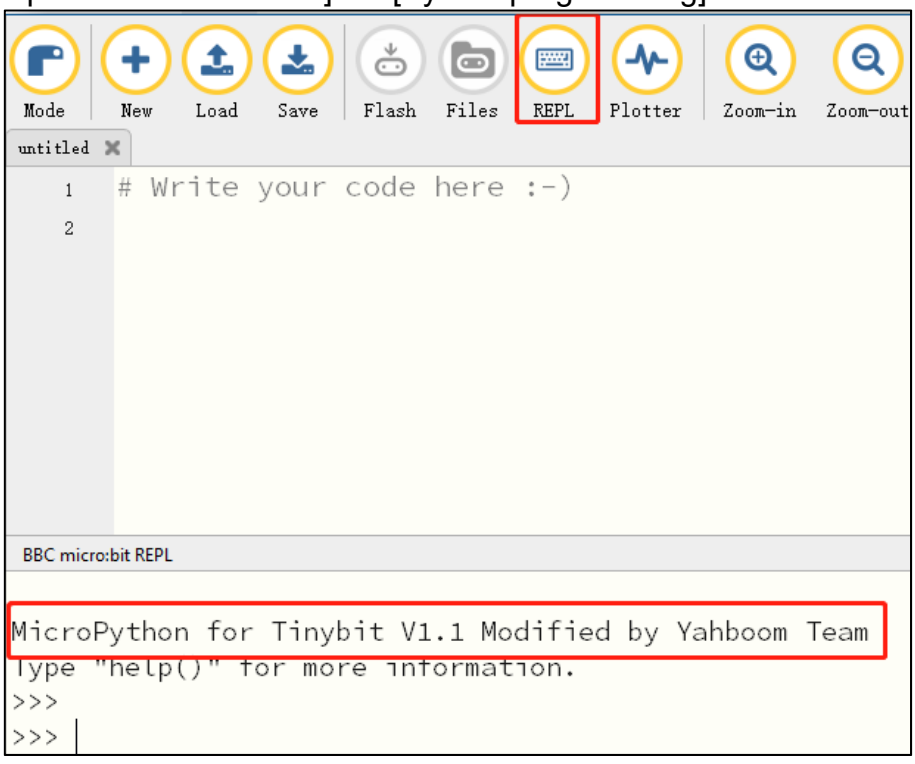
Note! All English and symbols should be entered in English, and the last line must be a space.



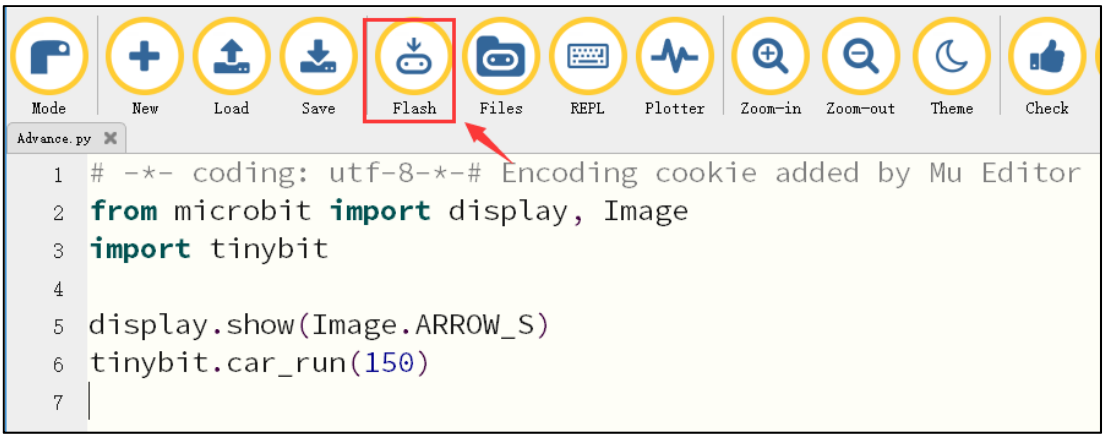
2.You can click the **"Check"** button to check if our code has an error. If a line appears with a cursor or an underscore, the program indicating this line is wrong.



3.Click **"REPL"** button,check whether the tinybit library has been downloaded. If not, please refer to the [preparation before class]---> [Python programming]



4.Click the **“Flash”** button to download the program to micro:bit board.



If the program is wrong or the experimental phenomenon is wrong after downloading, please confirm whether you have downloaded the Buildingbit libraryhex file we provided to the micro: bit board.

For the specific method of adding library files, please refer to 【1.Preparation before class】---【Python programming】

Experimental phenomena

After download is complete, open the power switch. We can see micro:bit dot matrix will display a smile. When we press A button, dot matrix will display running path icon, then we can press B button, car will be started.