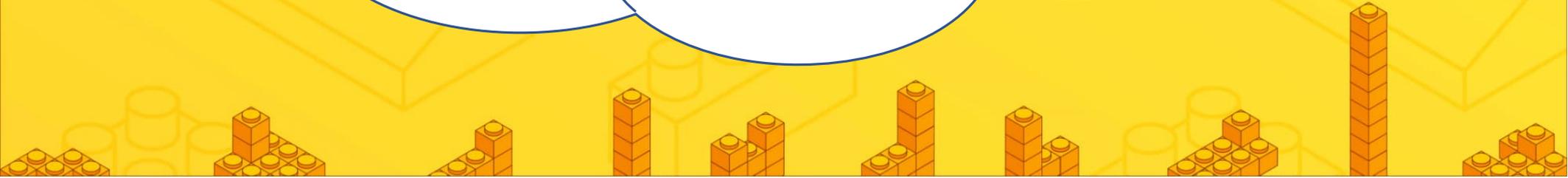




Track Trolley (1)





Target

- Learn to Use a Grayscale Sensor for Basic Line Following
- Master the Use of Conditional Structures in Programming
- Complete Learning How to Exit Loops



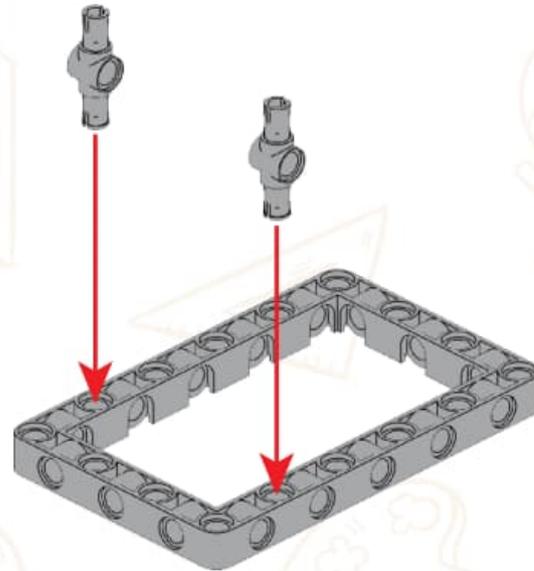
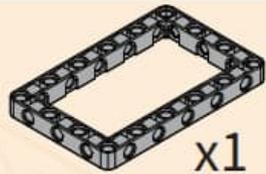


01 Assembly

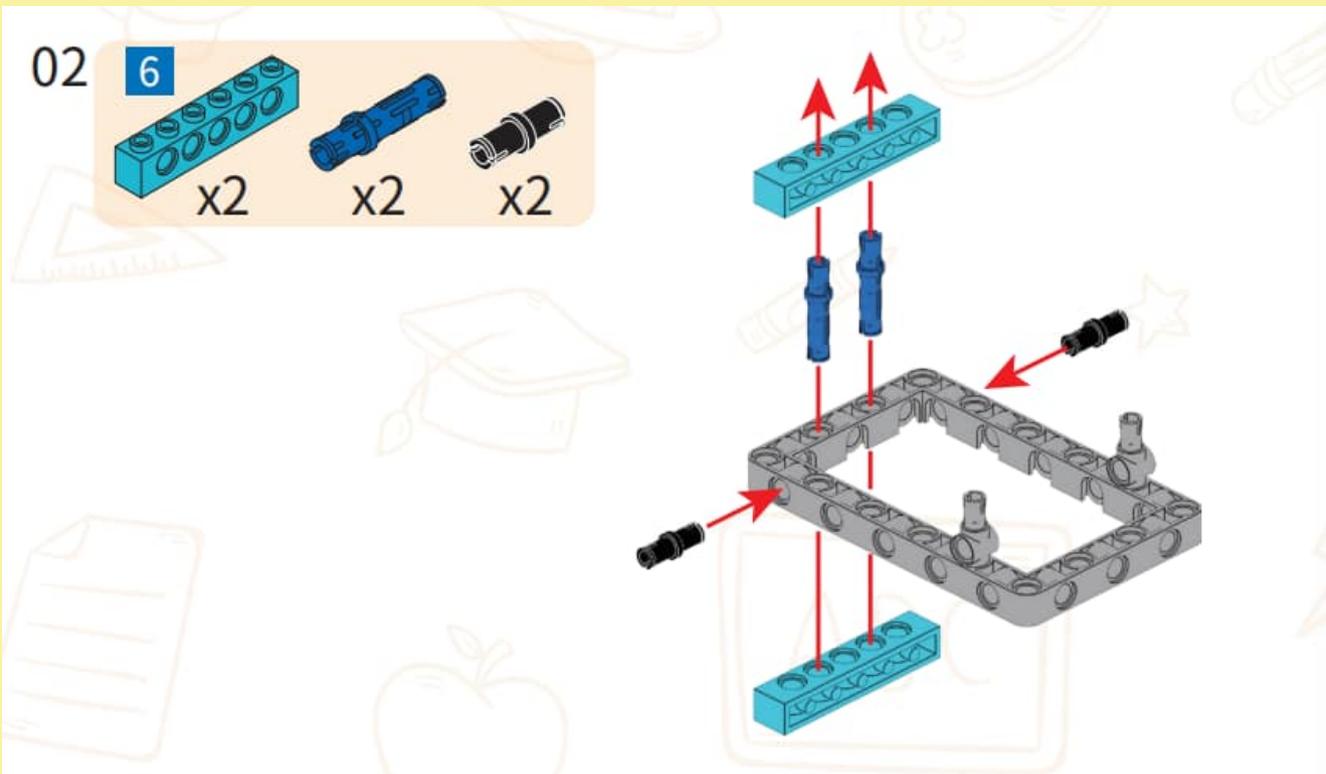


Assembly

01



Assembly



Assembly

03

3



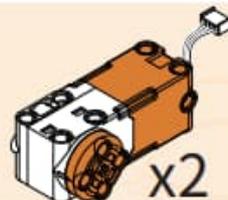
x2



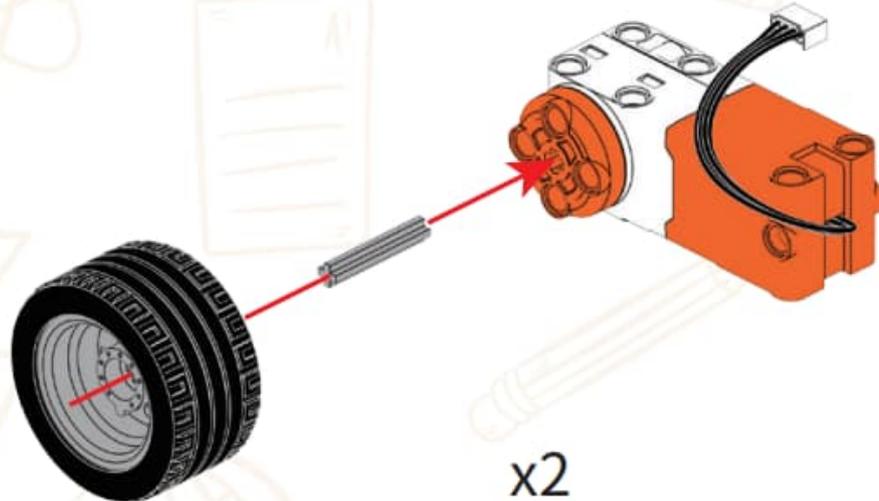
x2



x2



x2

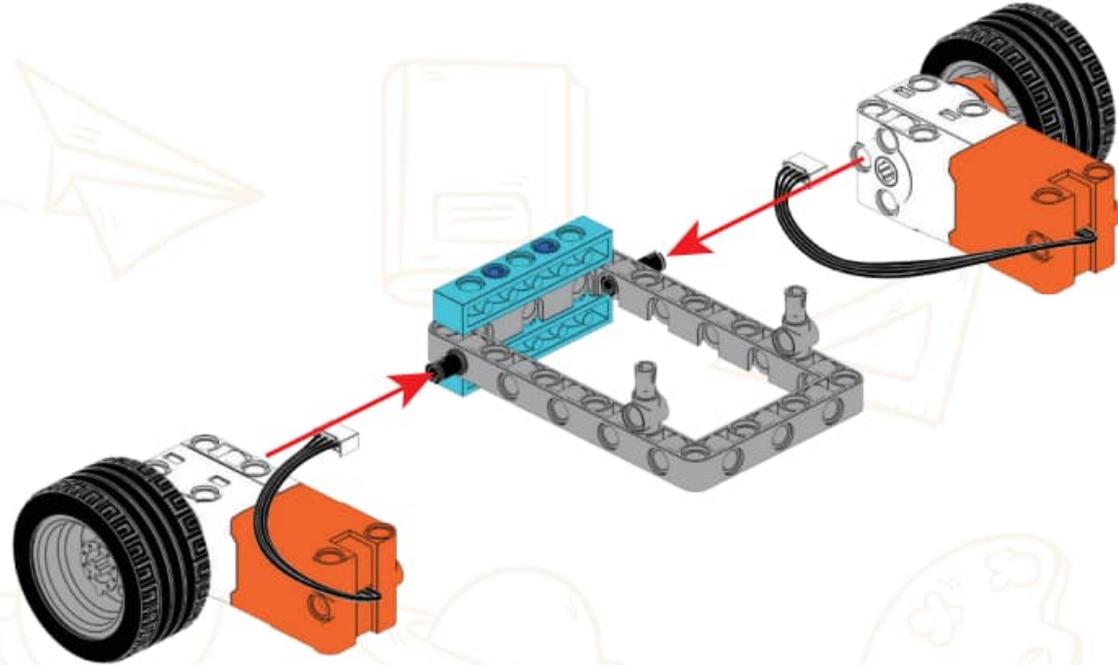


x2



Assembly

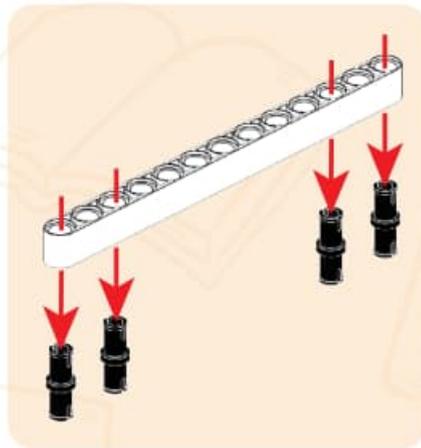
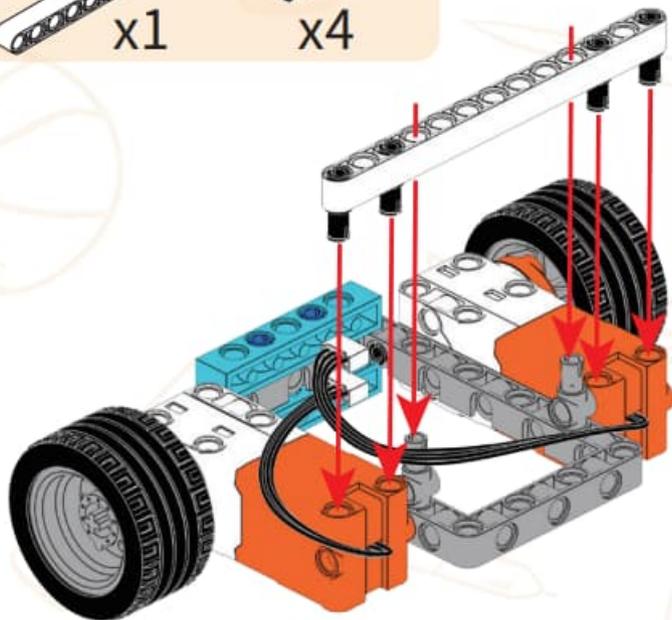
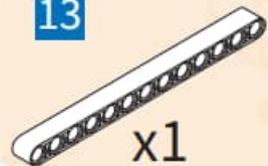
04



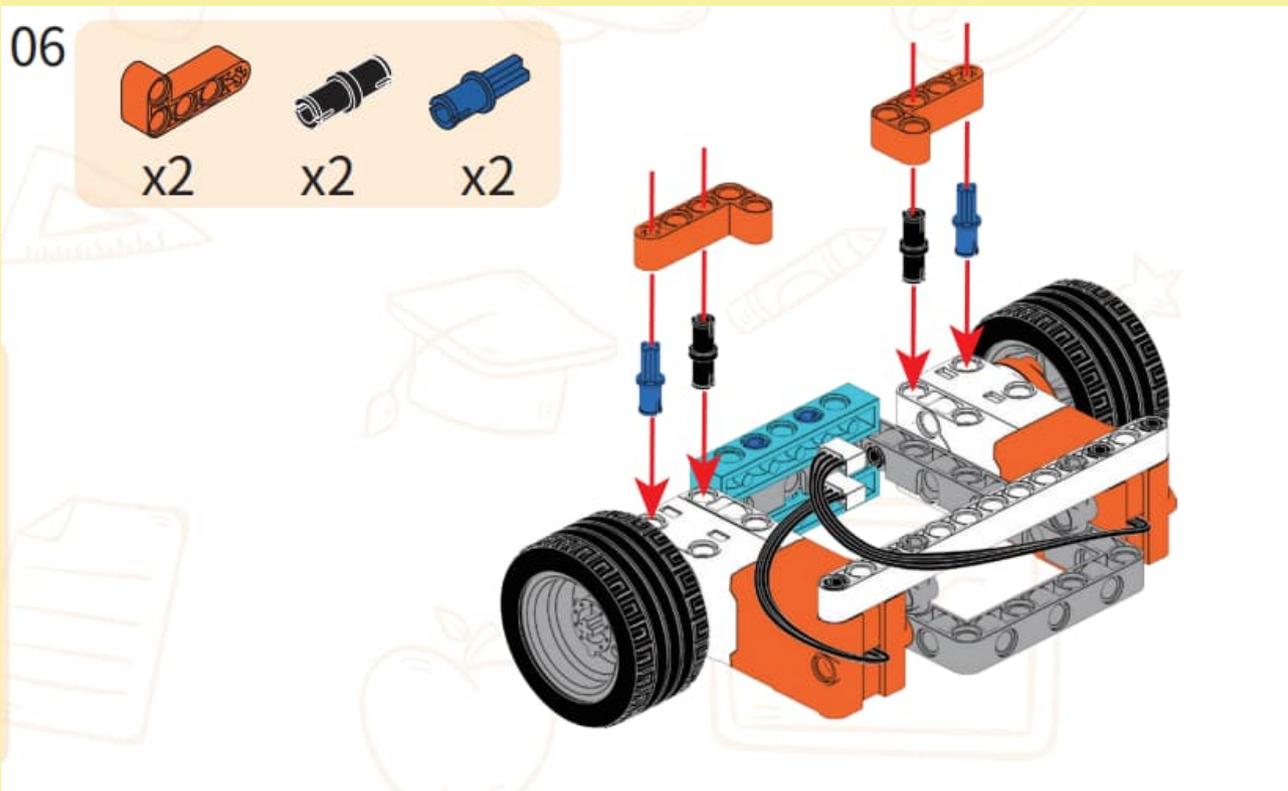
Assembly

05

13



Assembly



Assembly



07



x1



x1

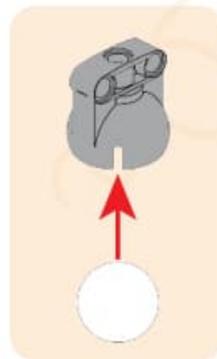
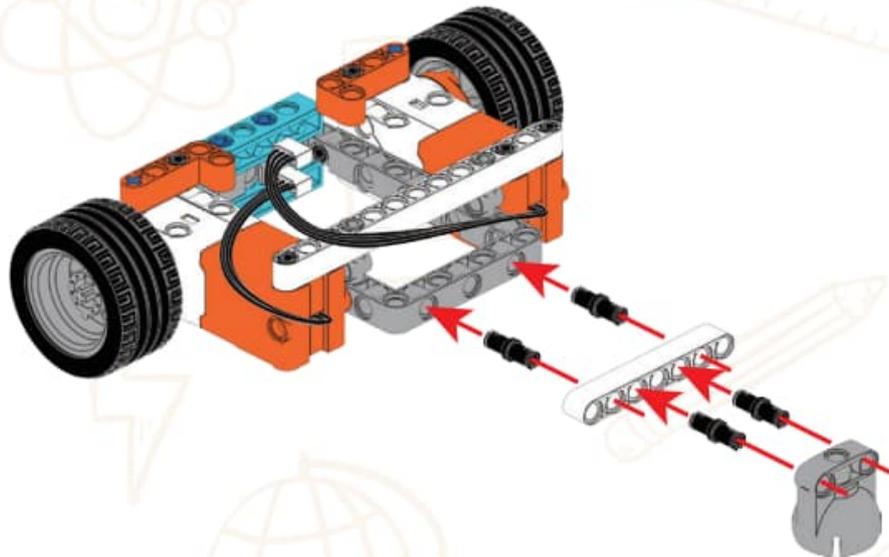


x4

7



x1

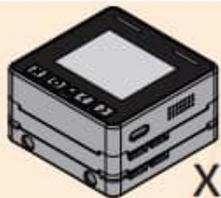


Assembly

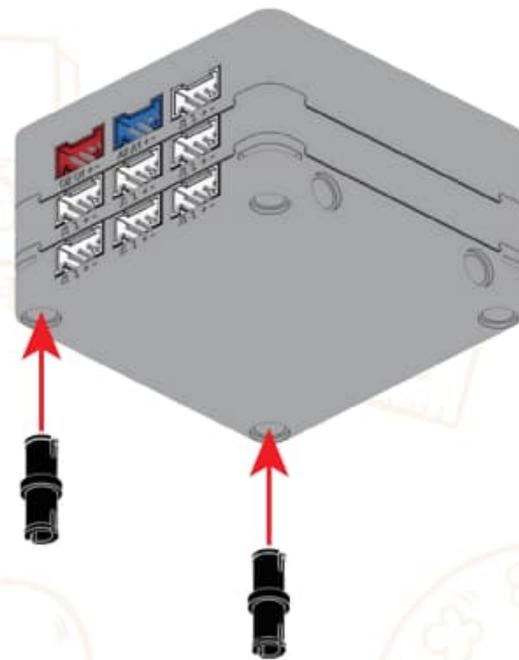
08



x2



x1



Assembly

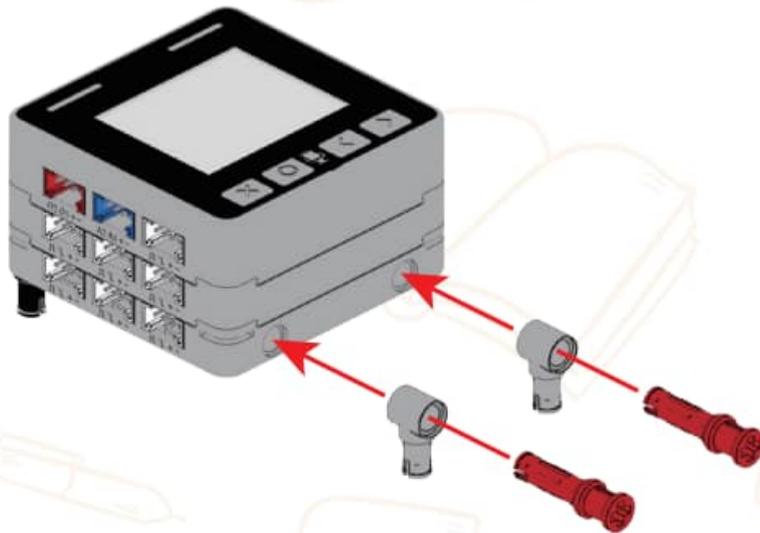
09



x2

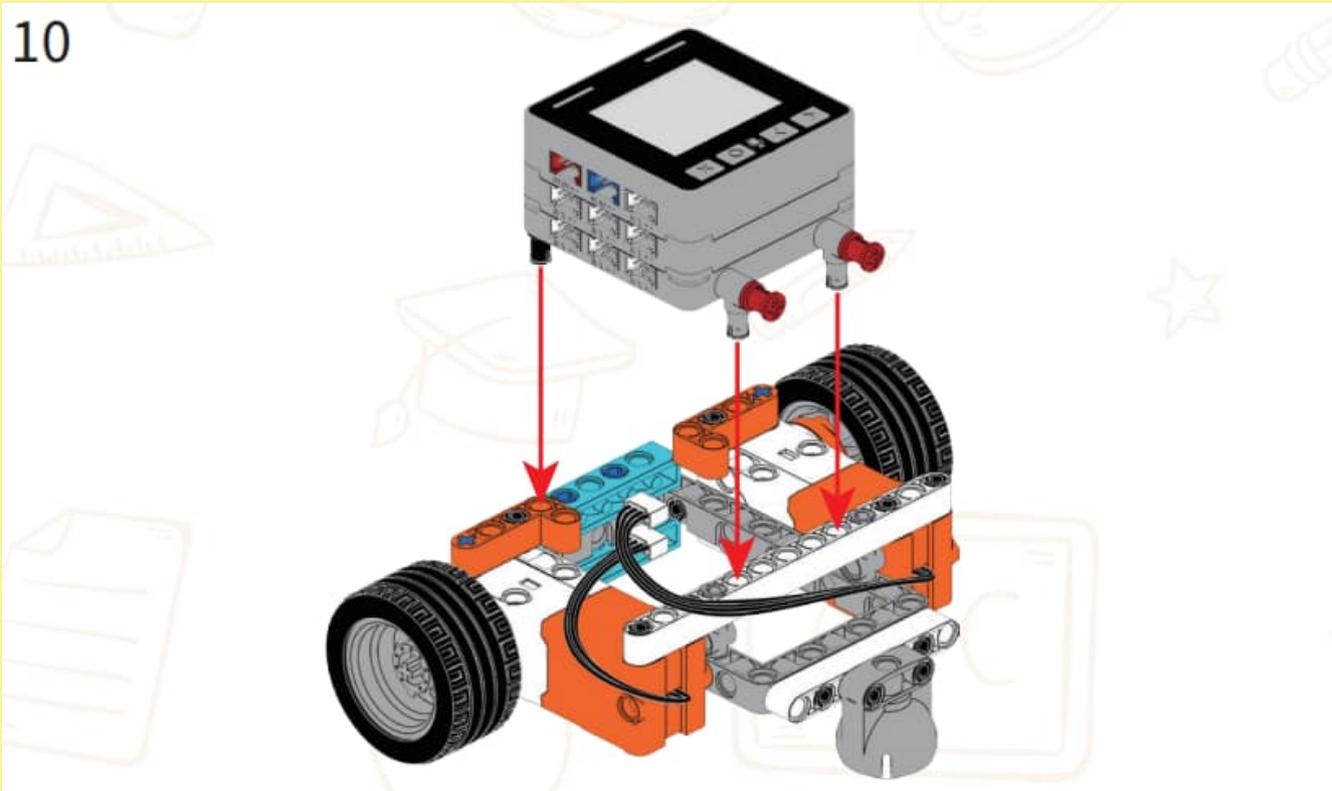


x2



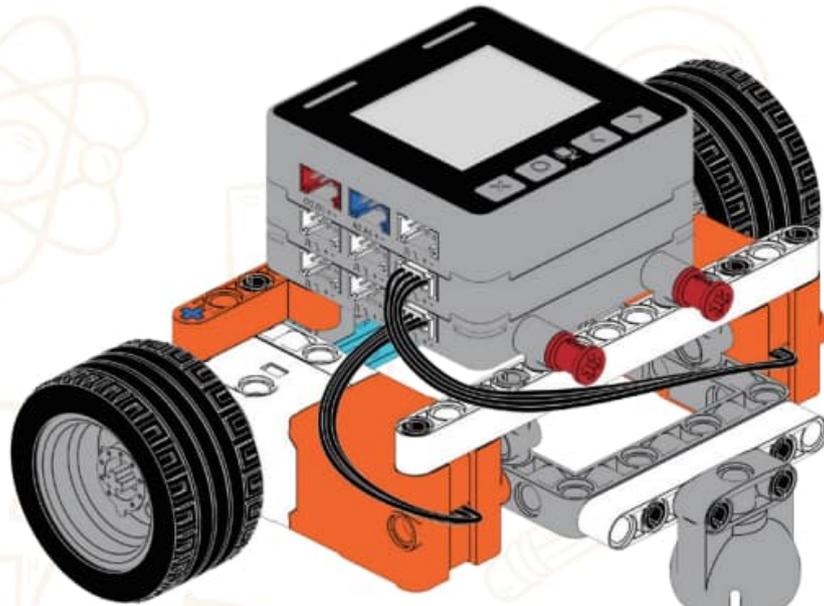
Assembly

10



Assembly

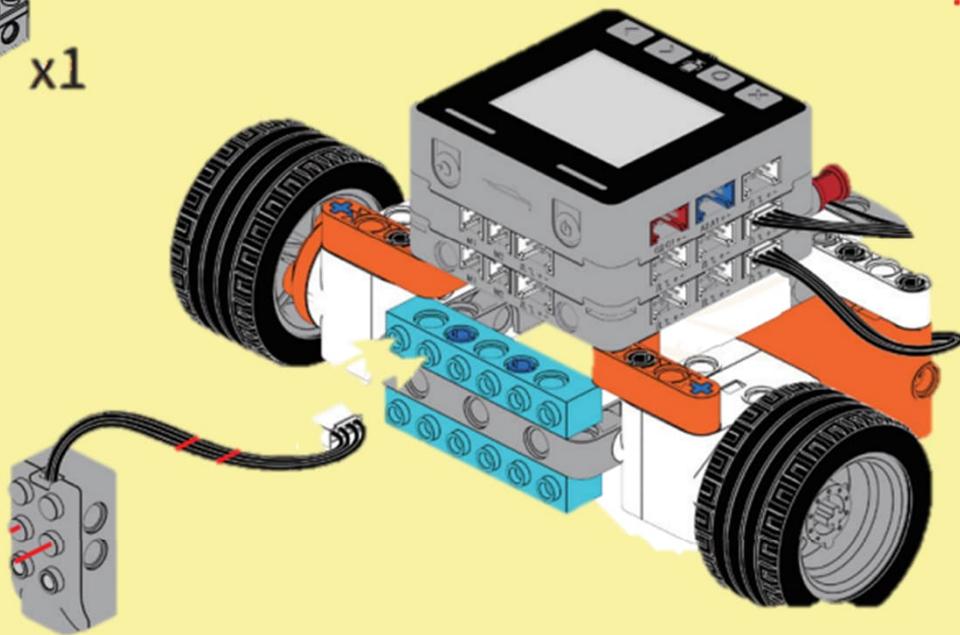
11



Assembly

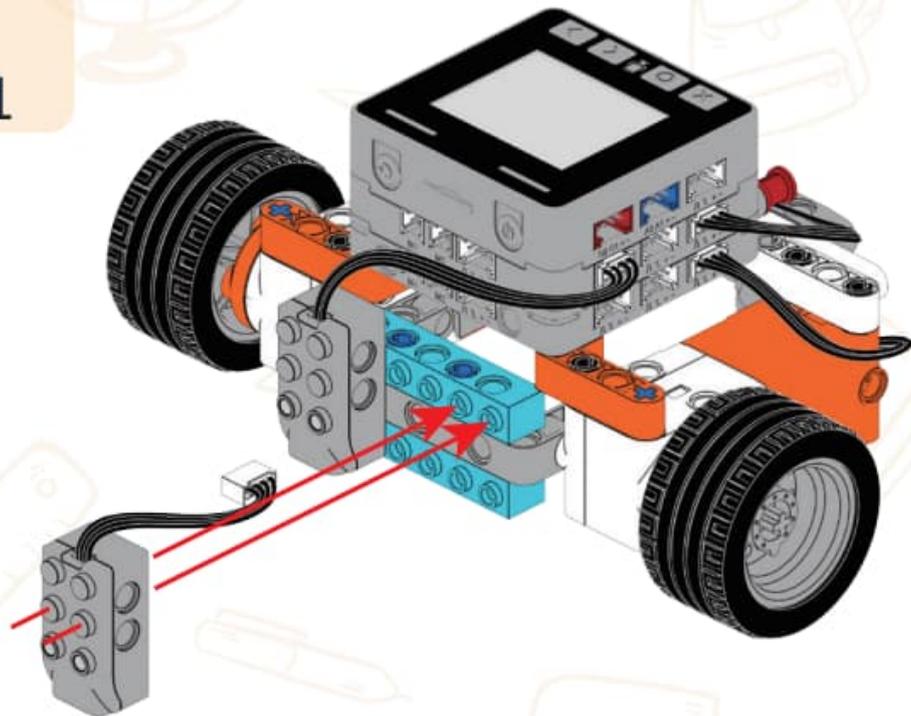
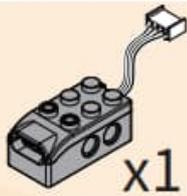


x1



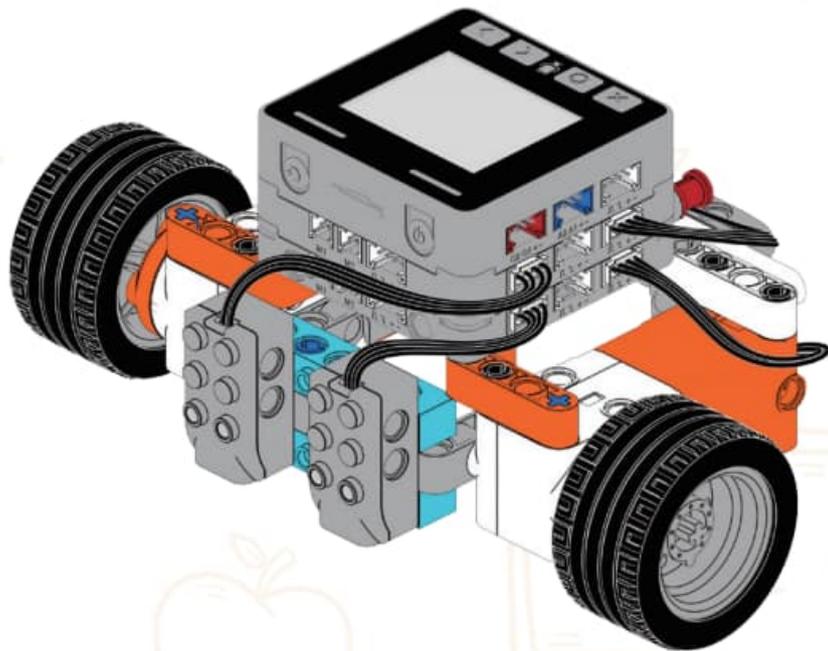
Assembly

13



Assembly

14





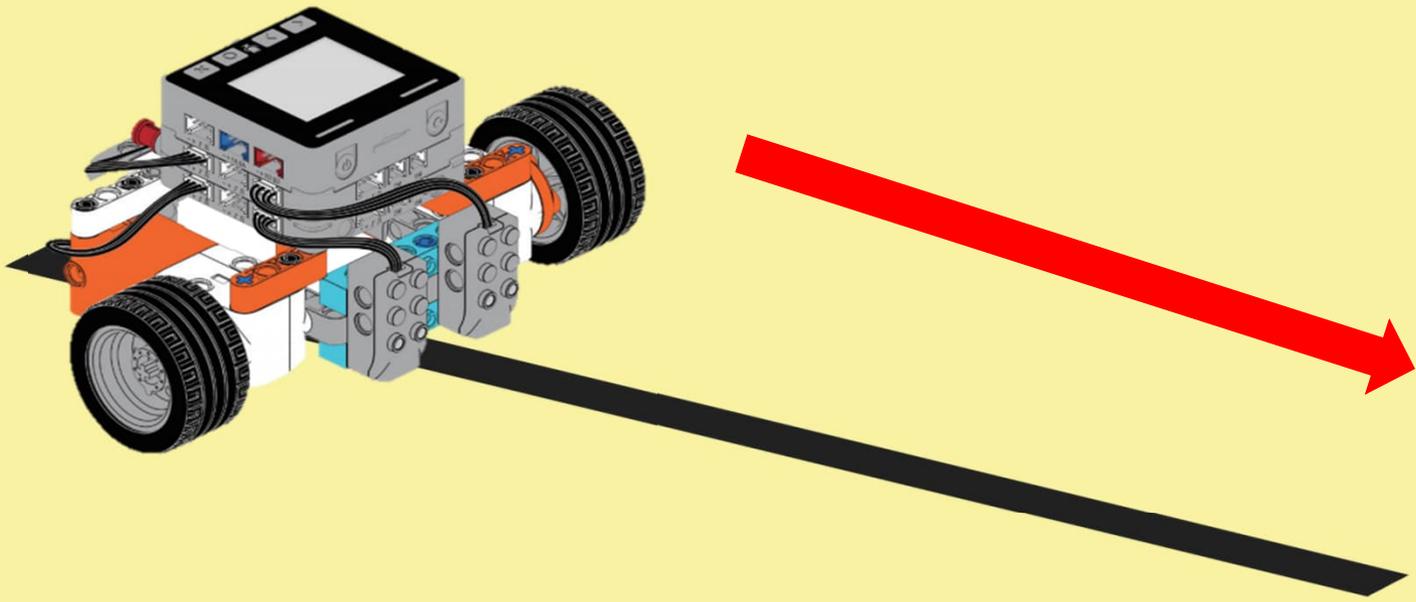
02 Task





Task

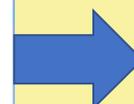
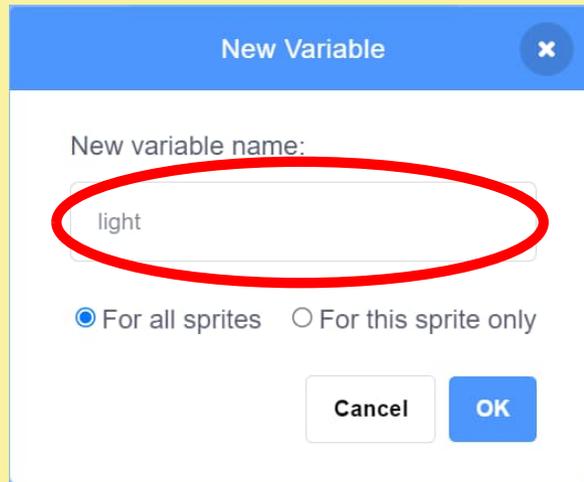
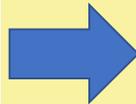
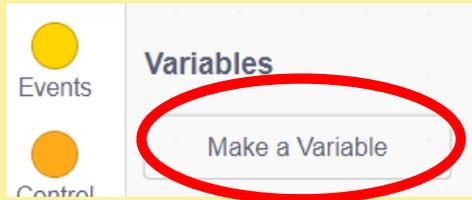
- Task 1: Use the Grayscale Sensor to Follow the Black Line



Coding Technique 1

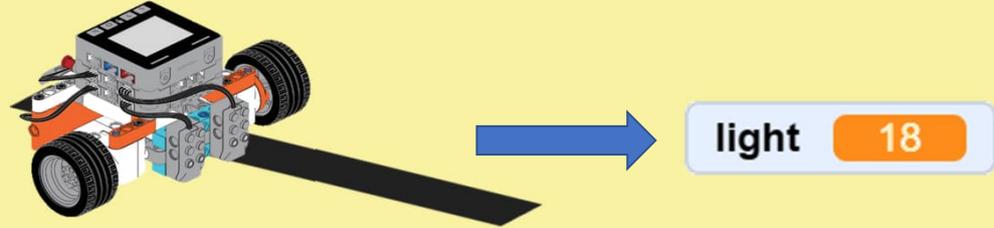
Using Variables and Viewing Sensor Values

Note: Variable names should use only English letters.

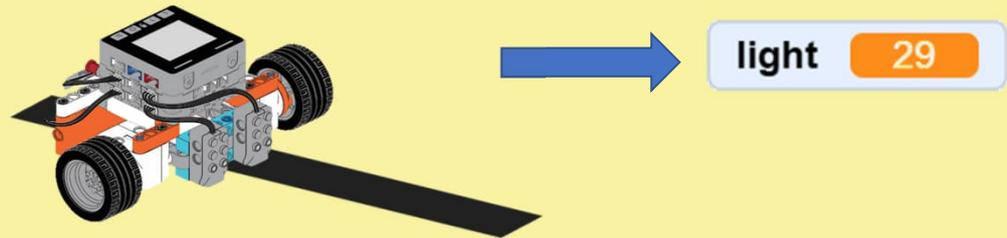


Coding Technique 1

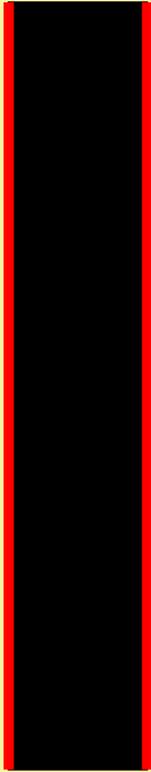
```
when green flag clicked
  forever loop
    set light sensor to single channel line tracker 1's value
```



Pay close attention to the difference between detecting the black line and not detecting it.



Coding Technique 1

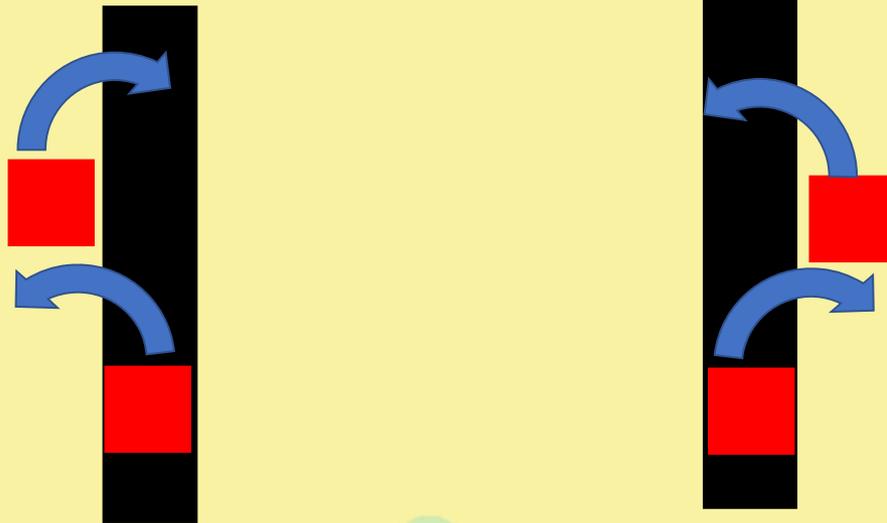


- A robot with a single grayscale sensor does not follow the black line itself.
- Instead, it follows the line where the black line meets the white area
- (the red line in the left image).
- Since there are two lines where the black line meets the white area,
- which one should the robot follow?



Coding Technique 1

- Which side to follow depends on your actions on the black line. Observe the diagram below.
- Currently, only two states are detected, so there can only be two corresponding actions.





Task

Referrable Program

(Walk along the left side of the black line)

Don't forget to upload the robot's program. Otherwise, it might not work successfully.

Threshold =

$$\frac{(black\ line\ value + white\ value)}{2}$$

2

It is recommended to switch to the power mode.

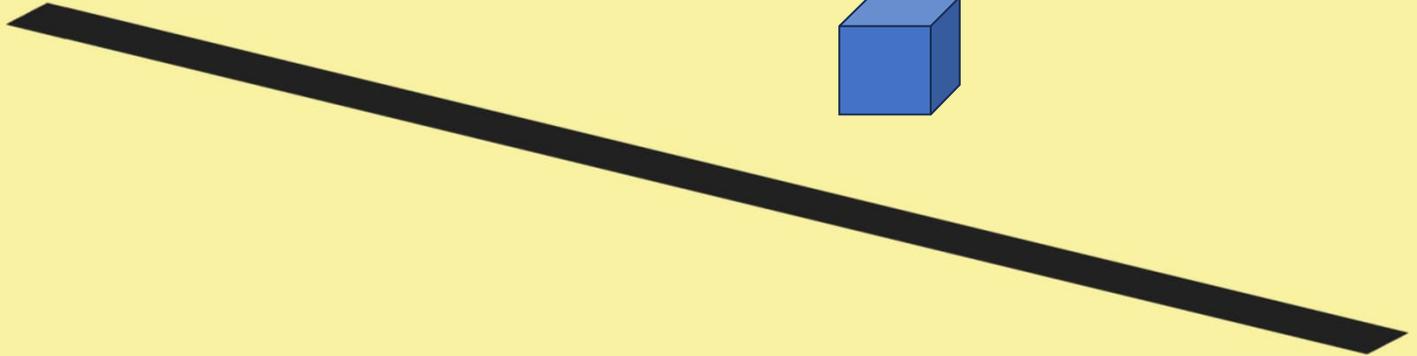
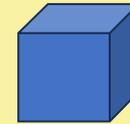
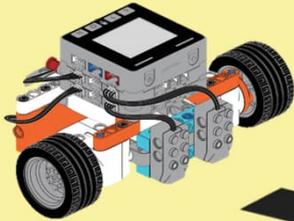
```

when flag clicked
  set 1# ext servo's origin
  set 2# ext servo's origin
  wait 0.1 seconds
  forever
    if single channel line tracker 1# 's value > 25 then
      set 1# ext servo to keep running at 0 % power on anticlockwise
      set 2# ext servo to keep running at 30 % power on clockwise
    else
      set 1# ext servo to keep running at 30 % power on anticlockwise
      set 2# ext servo to keep running at 0 % power on clockwise
  
```



Task

- Task 2: Use the Grayscale Sensor to Follow the Black Line for a Certain Distance and **Stop** Near the Block.



Task

Referrable Program

First, walk straight to the black line.

Repeat the command until the relative angle of the motor rotating clockwise is greater than 2600, and then stop the loop. (Parameter 2600 is modified according to the actual line patrol distance)



```
when clicked
  set 1# ext servo's origin
  set 2# ext servo's origin
  wait 0.1 seconds
  set 1# ext servo to keep running at 30 % power on anticlockwise
  set 2# ext servo to keep running at 30 % power on clockwise
  wait 0.5 seconds
  repeat until abs of 1# ext servo's counted degrees > 2600
  if single channel line tracker 1# 's value > 25 then
    set 1# ext servo to keep running at 0 % power on anticlockwise
    set 2# ext servo to keep running at 30 % power on clockwise
  else
    set 1# ext servo to keep running at 30 % power on anticlockwise
    set 2# ext servo to keep running at 0 % power on clockwise
  stop all ext servo(s)
```

The image shows a Scratch script for controlling two servos. The script starts with a 'when clicked' event, followed by two 'set' blocks for servo origins, a 0.1-second wait, and two 'set' blocks for servo positions (30% anticlockwise and 30% clockwise). These two 'set' blocks are highlighted with a red box. This is followed by a 0.5-second wait, a 'repeat until' loop where the absolute value of the first servo's counted degrees is greater than 2600. Inside the loop, an 'if' statement checks if the single channel line tracker's value for servo 1 is greater than 25. If true, servo 1 is set to 0% anticlockwise and servo 2 to 30% clockwise. If false, servo 1 is set to 30% anticlockwise and servo 2 to 0% clockwise. The script ends with a 'stop all' block for the servos.

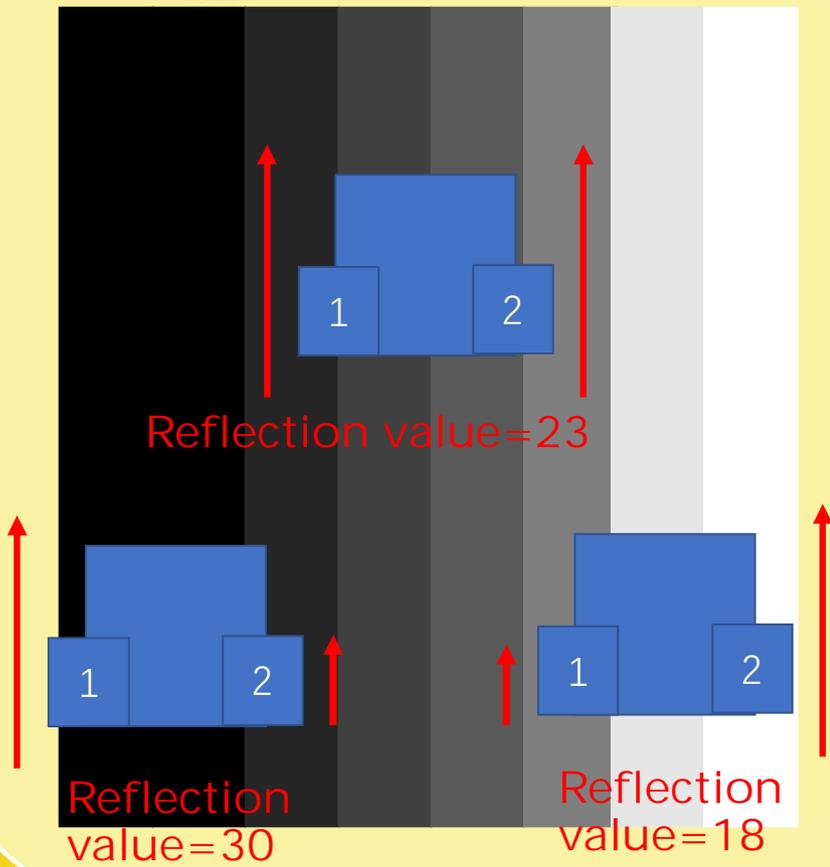
Additional Task

- Note: This task is not mandatory for learning. You only need to know which parameters to adjust.
- Additional Task: Use the proportional control mode to perform line-following **actions with a single color sensor**.





Coding Technique 2



- The boundary between black and white is a gradient pattern.
- The arrows represent the power levels.
- Have you noticed the proportional relationship between the reflection value and the motor power?



Coding Technique 2

Original Power (20~30) Threshold = $\frac{\text{black line value} + \text{white value}}{2}$

The image shows a Scratch script starting with a 'when green flag clicked' block, followed by a 'forever' loop. Inside the loop, there are two 'set ext servo to keep running at' blocks. The first block is for servo 1# and the second for servo 2#. Both blocks use a mathematical expression: $25 + 0.5 * 40 + \text{single channel line tracker } 1\# \text{ 's value} \% \text{ power on } \text{direction}$. Red boxes and arrows highlight the values 25, 0.5, and 40 in both blocks. An arrow from the text 'Original Power (20~30)' points to the 25. An arrow from the text 'Threshold = (black line value + white value) / 2' points to the 40. An arrow from the text 'The larger the proportional value, the more intense the oscillation.' points to the 0.5.

The larger the proportional value, the more intense the oscillation.

