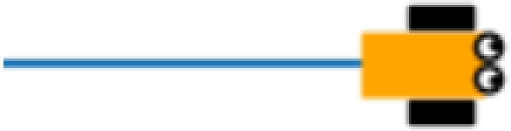


Robocar in python™

Functions

Here are the basic functions you can use, try them out.

```
robocar.drive(direction, distance)
```

direction = forward or back

distance = a number

```
robocar.turn(direction, angle)
```

direction = left or right

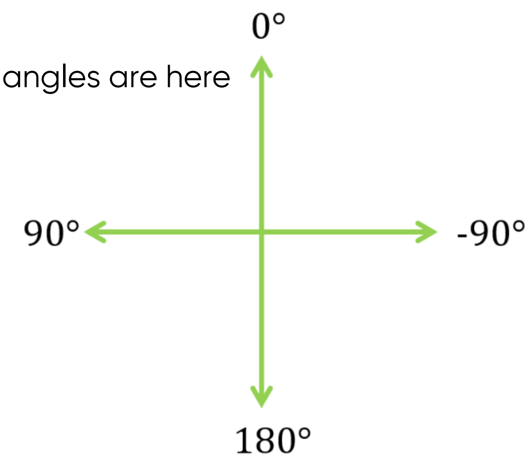
angle = a number (degrees)

```
robocar.scan_ahead()
```

tells you the distance to any obstacle in front of the car

```
robocar.get_heading()
```

tells you the direction the car is pointing in, the angles are here



```
robocar.get_position()
```

tells you the position of the car

Loops and Conditions

We can use "conditional" statements to only run some code "if" a condition is met, or "while" something isn't true.

In Python, the syntax is:

```
if condition:
    do some code once…
```

Or

```
while condition:
    repeat some code until condition is false…
```

A "condition" is something which evaluates to True or False, some examples:

`1 < 2`	Always True
`1000 < 2`	Always False
`robocar.scan_ahead() < 5`	True if something is closer than 5

Example: turn right by 5 degrees if there is an obstacle less than 5 units in front of robocar

```
if robocar.scan_ahead() < 5:
    robocar.turn(right,5)
```

Example: drive forwards until robocar is near a wall

```
while robocar.scan_ahead() > 2:
    robocar.drive(forward,1)
```

Challenges

1. Draw a square

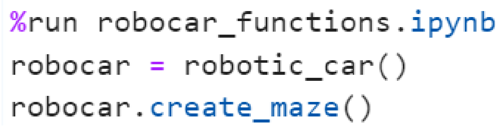
2. Drive to (5,10) and then (10,0)

2b) **(DIFFICULT)** Drive to (5,10) and then (10,0) but <u>only use robocar_drive() twice</u>

3. Draw a circle

4. Complete the maze

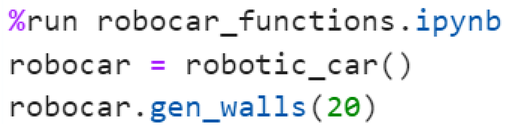
Add the line **`robocar.create_maze()`** to the top of your code and then complete the maze by reaching the green square without crashing



```
%run robocar_functions.ipynb
robocar = robotic_car()
robocar.create_maze()
```

5. Automatically avoid obstacles

Add the line **`robocar.gen_walls(20)`** to the top of your code. This will create 20 walls and place them randomly in the area. Make robocar drive around continuously, while avoiding the walls.

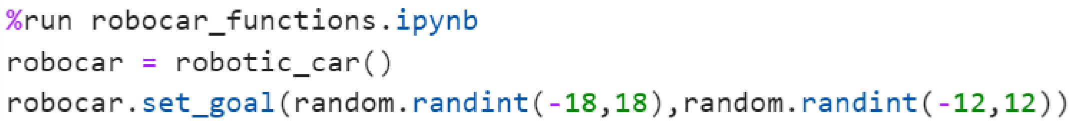


```
%run robocar_functions.ipynb
robocar = robotic_car()
robocar.gen_walls(20)
```

Hint: use the scan_ahead function, what should robocar do if scan_ahead says it is close to a wall?

6. Automatically drive to a goal

Add the below line to the top of your code:



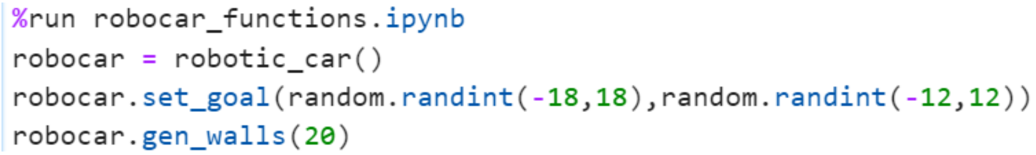
```
%run robocar_functions.ipynb
robocar = robotic_car()
robocar.set_goal(random.randint(-18,18),random.randint(-12,12))
```

This will create a goal in a random place on the map. You can get the heading of this goal (the direction robocar needs to travel to reach it), by using the function `robocar.get_goal_heading()`

Hint: compare the goal heading with robocars heading!

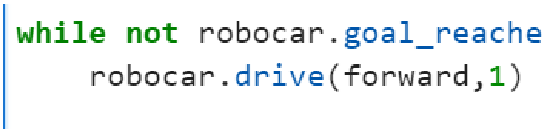
7. Search and rescue! *(Difficult!)*

Use the below setup, this will place a random goal in the map, and place random walls in the way. Can you program robocar to autonomously reach the goal without crashing, and **without using the get_goal_heading function.** This means that robocar doesn't know where the goal is, and needs to search the map without crashing.



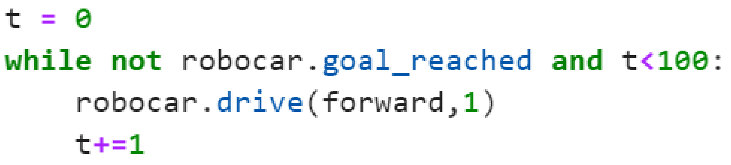
```
%run robocar_functions.ipynb
robocar = robotic_car()
robocar.set_goal(random.randint(-18,18),random.randint(-12,12))
robocar.gen_walls(20)
```

The below loop will execute some code until the goal is reached. Note that if robocar doesn't reach the goal, this code will never stop!



```
while not robocar.goal_reached:
    robocar.drive(forward,1)
```

Add a timer so that the code will stop after a set time, even if robocar doesn't reach the goal



```
t = 0
while not robocar.goal_reached and t<100:
    robocar.drive(forward,1)
    t+=1
```

Hint: because the walls and goal are placed randomly, even a well programmed car might not reach the goal on every attempt.