Reeborg's World : Understanding Functions

1. Home 1 ~ 4

I want to go home!

Write a program that makes Reeborg go home.

> **What you need to know**
>
> - The function `move()` and `turn_left()`.
> - To make Reeborg turn right, you will have to tell it to `turn_left()` three times in a row.

More advanced

You may have noticed that your solution as some repeated patterns as you can see from the > > Reeborg's World image.

Once you know how to define functions, write a solution with the repeated code put in functions.

Using Python and the special keyword `repeat`/`for` unique to Reeborg's World.

If you know how to define functions, you should attempt to write a similar program.

2. Around 1 ~ 4

Around the world

Have Reeborg go around the world in the counter-clockwise direction once, and stop at its starting position.

Reeborg must not step on the grass.

The correct path is shown by the dashed white line.

> **What you need to know**
>
> - The functions `put()`, `move()`, and `turn_left()`.

More advanced

You can write a more general program, having Reeborg decide by itself when it had come back to its starting point.

> **What you need to know**
>
> - The test `front_is_clear()` or `wall_in_front()`, `right_is_clear()` or `wall_on_right()`, and `object_here()`.

> - How to use a `while` or a `repeat`/`for` loop and an `if elif else` statements.
> - It might be useful to know how to use the negation of a test (`not` in Python).

Ideally, you should define your own functions to make your program easier to understand.

Hint

Reeborg carries some tokens; you can only use one.

3. Center 1 ~ 2

Find the center

Help Reeborg to find the center of the world.

The height and width of the world will change each time, but they each will be an odd number, from 1 to 11.

At first, you can know the end of the world, if you move until you meet the wall.

The center is the middle point of the length of the world.

You should put down the one object at the center.

> **What you need to know**
>
> - The functions `move()`, `turn_left()` and `put()`.
> - The conditions `front_is_clear()` or `wall_in_front()`, and `object_here()`.
> - How to use `while` loops and `if` statements.

Hint

Reeborg carries many tokens; you can only use two.

> **Suggested strategy**
>
> - A possible strategy is to only use two tokens and put one at each end of the world.
> - Then, by moving them one step at a time, Reeborg could find the center of the world.

4. Harvest 1 ~ 3

Harvest time

Reeborg's garden is ready to harvest. Have Reeborg collect all the carrots in his garden.

> **What you need to know**
>
> - The functions `move()`, `turn_left()`, and `take()`.

> - The test `object_here()`.
> - How to use a `while` loop or an `if` statement.

You might find it convenient to define a function and call this function 6 times.

Planting time

Reeborg has planted some carrot seeds.

At some places no carrot has grown; at others, many carrots are growing.

Have Reeborg remove the excess carrots and plant new ones where there is none so that there is only one carrot at each location.

Reeborg already carries enough carrots (seeds) to replant the entire garden if needed.

> **What you need to know**
>
> - The functions `move()`, `turn_left()`, `take()`, and `put()`.
> - The test `object_here()`.
> - How to use a `while` loop or an `if` statement.

You might find it convenient to define a function and call this function 6 times.

5. Hurdle 1 ~ 4

Hurdles race

Reeborg has entered a hurdles race.

Make him run the course, following the path shown.

In other case, Reeborg does not know in advance how long the race is.

Also the position and the number of hurdles can be changed each time and this world is reloaded.

> **What you need to know**
>
> - The functions `move()` and `turn_left()`.
> - The condition `at_goal()`, `front_is_clear()` or `wall_in_front()` and their negation.
> - How to use a `while` loop and an `if` statement.

More advanced

You may have noticed that your solution has some repeated patterns.

If you know how to define functions, define a function and use it to simplify your program.

6. Maze

Lost in a maze

Reeborg was exploring a dark maze and the battery in its flashlight ran out.

Write a program using an `if/elif/else` statement so Reeborg can find the exit.

The secret is to have Reeborg follow along the right edge of the maze, turning right if it can, going straight ahead if it can't turn right, or turning left as a last resort.

> **What you need to know**
>
> - The functions `move()` and `turn_left()`.
> - Either the test `front_is_clear()` or `wall_in_front()`, `right_is_clear()` or `wall_on_right()`, and `at_goal()`.
> - How to use a while loop and `if/elif/else` statements.
> - It might be useful to know how to use the negation of a test (`not` in Python).