CS101 Homework #1

Hubo consuming fuel

Read the homework description carefully and follow the instructions. Please be fully aware that this homework is an **individual task**; you can discuss the problem with your friends, but you must not implement your idea together. **You will fail the entire course (that is, your CS101 grade will be an F) if you are found to be involved in any attempt of plagiarism**.

Preliminaries

The `cs1robots` library used in Lab 1-3 is used. Specifically, you may need to use the following methods:

```
drop_beeper()
pick_beeper()
on_beeper()
carries_beeper()
move()
turn_left()
load_world()
front_is_clear()
left_is_clear()
right_is_clear()
```

Requirements

(Task 1. 10 points) Implement a function `goUntilRunOut()`

In the Task 1, a straight-shaped world is given. The length of streets is 1 and the length of avenues is an unknown number from 1 to 100. Write a function `goUntilRunOut()` that makes Hubo move until the fuel runs out. **The number of beepers Hubo starts with is a random integer greater than or equal to 0.** When Hubo can't move any further, Hubo must drop all beepers on the spot.

Note: Hubo named `hubo` is already created at (1, 1), heading East. You must not create a robot again!

Note: You must use `hubo` to move Hubo.

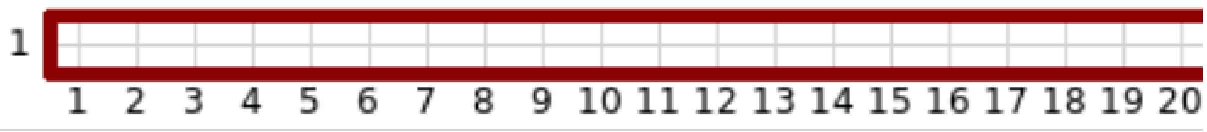
Note: There are example code lines to help you. You may start the task by erasing them first.

Note: You can write any additional functions you would like.

Note: Hubo should be at the precise position as the result of moves. Also the numbers of beepers laid over the world must be accurate. The direction of the Hubo is not checked.

Example

Input 1

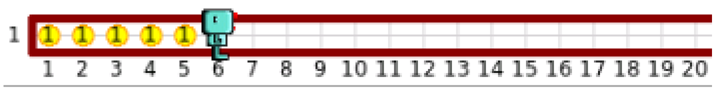


`ex1.wld`

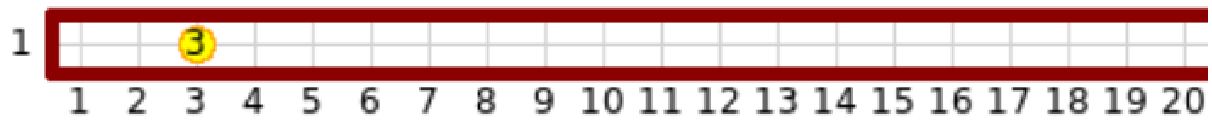
world size = (100, 1), No beepers

`hubo = Robot(beepers=5)`

Output 1



Input 2

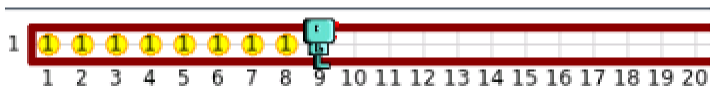


`ex2.wld`

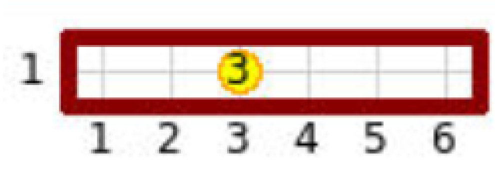
world size = (100, 1), 3 beepers on (3,1)

`hubo = Robot(beepers=5)`

Output 2



Input 3

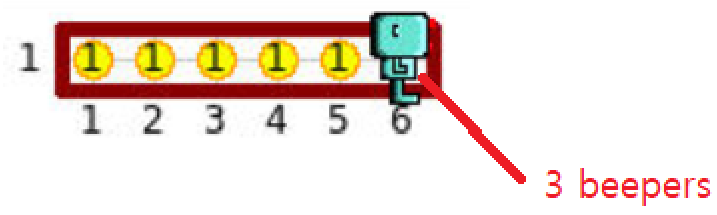


`ex3.wld`

world size = (6, 1), 3 beepers on (3,1)

`hubo = Robot(beepers=5)`

Output 3



(Task 2. 25 points) Implement a function `goUntilRunOut2()`

you can use your codes in Labs, Task 1

In the Task 2, the world is given as an one-way maze, rather than a straight-shaped world. In the maze, Hubo can only move in two directions at all positions except a starting point (1, 1) and an end point. Hubo can always move in at least one direction (It menas that a single-size maze is not considered).

When Hubo changes its direction, one more fuel is consumed. If there is a curve where Hubo changes direction and moves, two beepers will be dropped at the location.

Write a function `goUntilRunOut2()` that makes Hubo move until the fuel runs out. **The number of beepers Hubo starts with is a random integer greater than or equal to 0.** When Hubo can't move any further (reach the end point), Hubo must drop all beepers on the spot.

Note: Hubo named `hubo` is already created at (1, 1), heading East. You must not create a robot again!

Note: You must use `hubo` to move Hubo.

Note: There are example code lines to help you. You may start the task by erasing them first.

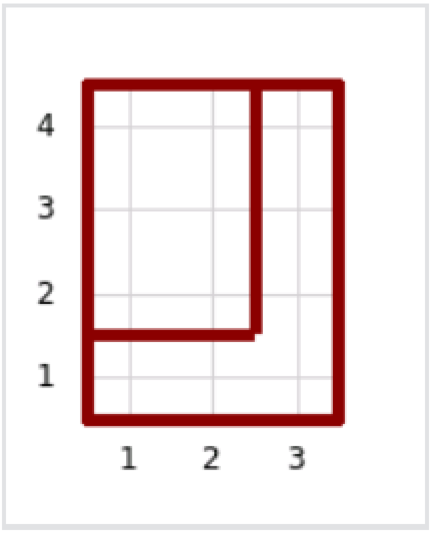
Note: You can write any additional functions you would like.

Note: Hubo should be at the precise position as the result of moves. Also the numbers of beepers laid over the world must be accurate. The direction of the Hubo is not checked.

Examples

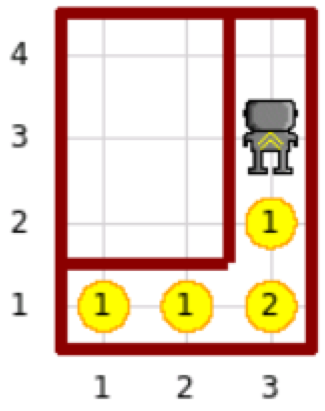
Input 1

`ex1.wld`



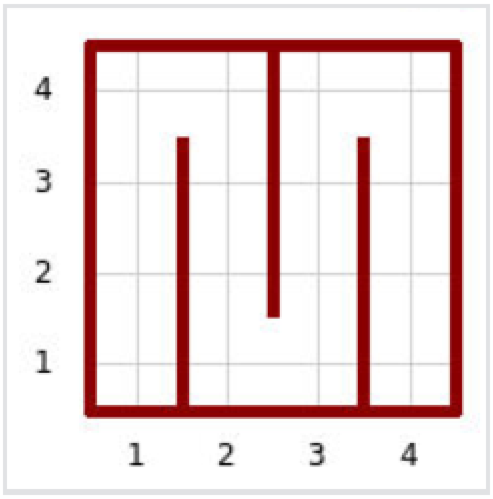
`hubo = Robot(beepers=5)`

Output 1



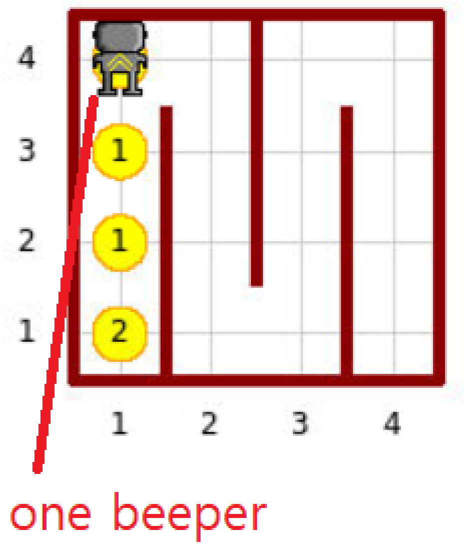
Input 2

`ex2.wld`



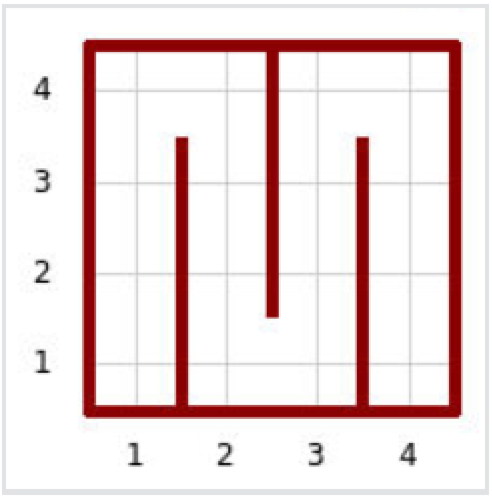
`hubo = Robot(beepers=5)`

Output 2



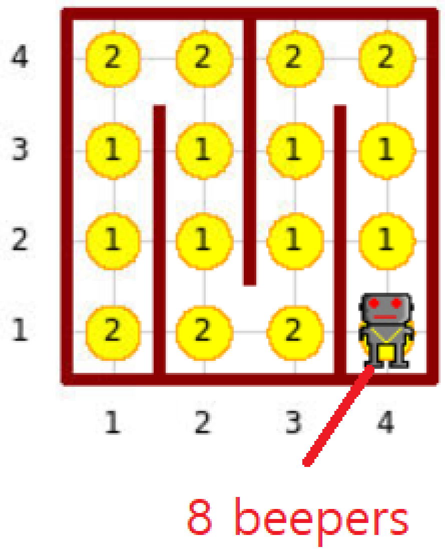
Input 3

`ex2.wld`



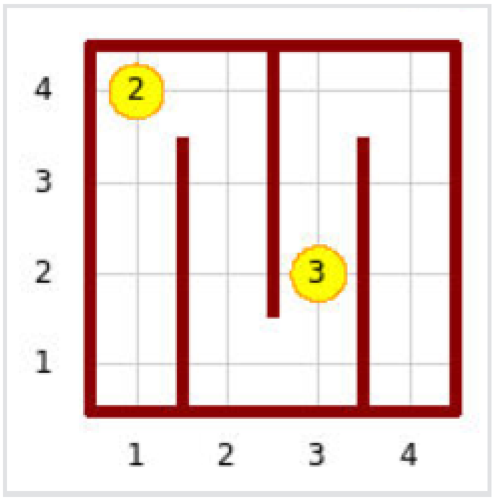
`hubo = Robot(beepers=30)`

Output 3



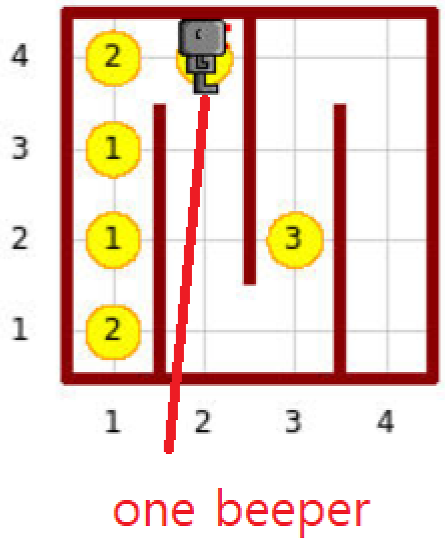
Input 4

`ex3.wld`



`hubo = Robot(beepers=5)`

Output 4



(Task 3. 15 points) Implement a function `goUntilRunOut3()`

you can use your codes in Labs, Task 1 and Task 2

In the Task 3, the rule is same as Task 2 except that Hubo tries to return to the starting point if there are fuels left. During the return, Hubo does not pick up any beepers on the ground. **At the end point, since it moves after 2 turns, it consumes 3 fuels for a total of 3 actions**. When Hubo reach the starting point during the return, Hubo stops and drops all beepers.

Write a function `goUntilRunOut3()` that makes Hubo move until the fuel runs out. **The number of beepers Hubo starts with is a random integer greater than or equal to 0.** When Hubo can't move any further (reach the starting point on a return), Hubo must drop all beepers on the spot.

Note: Hubo named `hubo` is already created at (1, 1), heading East. You must not create a robot again!

Note: You must use `hubo` to move Hubo.

Note: There are example code lines to help you. You may start the task by erasing them first.

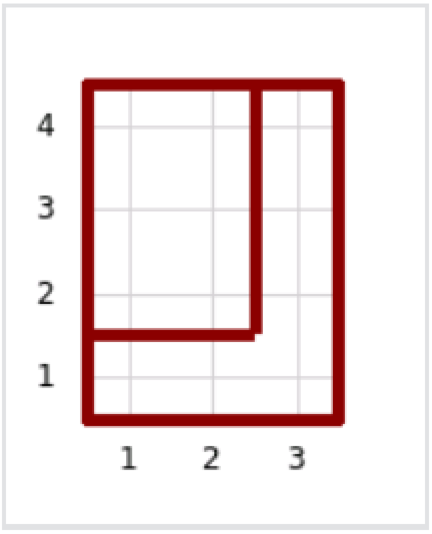
Note: You can write any additional functions you would like.

Note: Hubo should be at the precise position as the result of moves. Also the numbers of beepers laid over the world must be accurate. The direction of the Hubo is not checked.

Examples

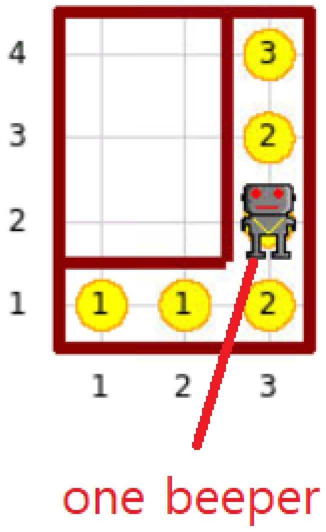
Input 1

`ex1.wld`



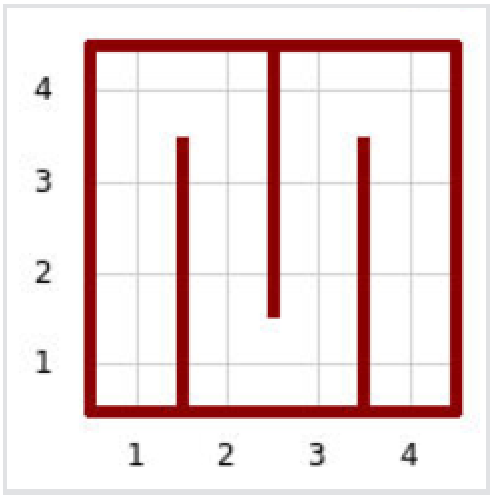
`hubo = Robot(beepers=10)`

Output 1



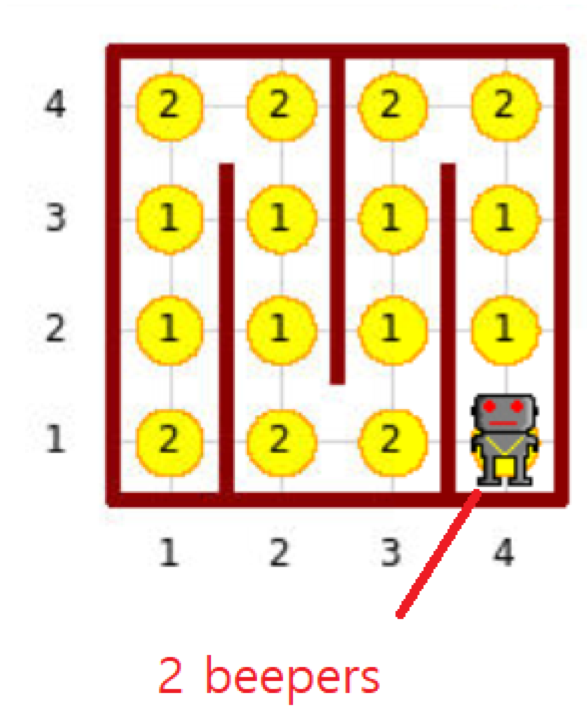
Input 2

`ex2.wld`



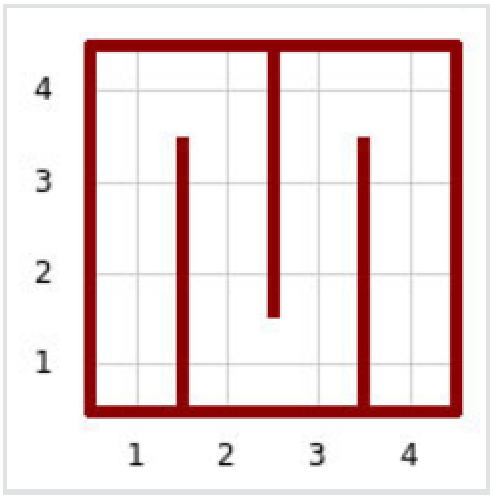
`hubo = Robot(beepers=24)`

Output 2



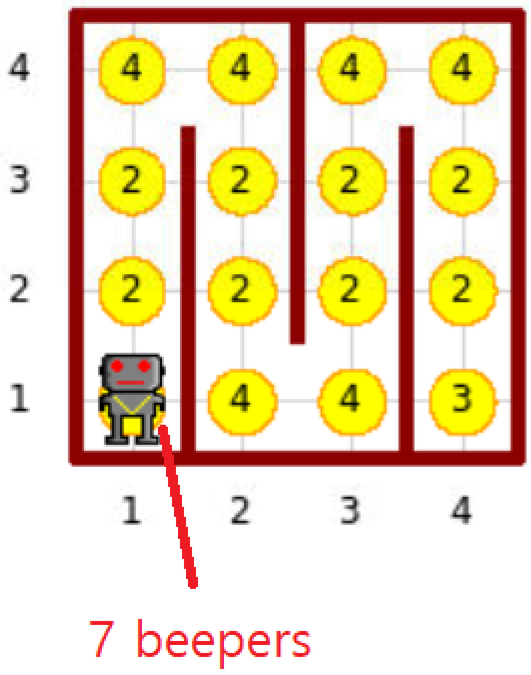
Input 3

`ex2.wld`



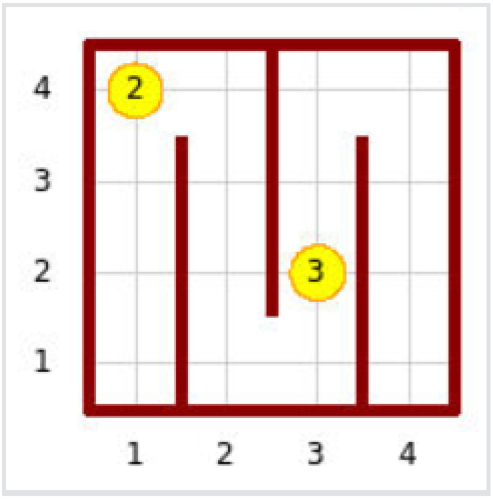
`hubo = Robot(beepers=50)`

Output 3



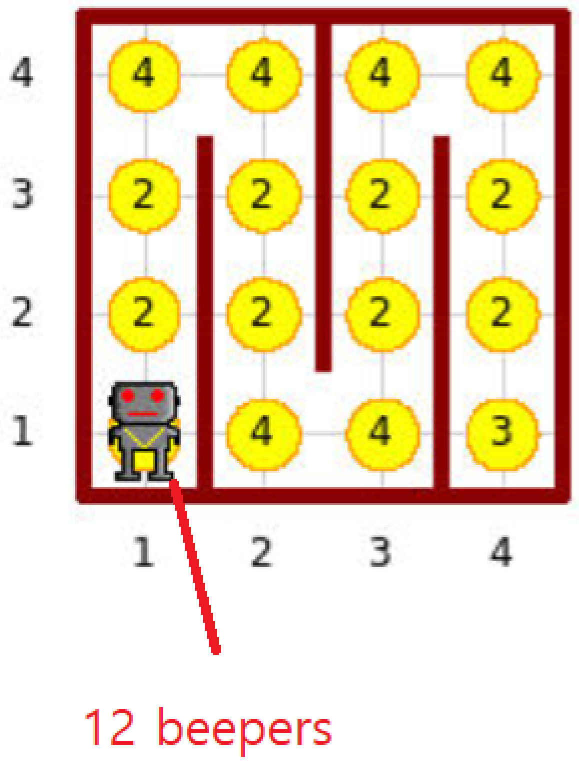
Input 4

`ex3.wld`



`hubo = Robot(beepers=50)`

Output 4



4
3
2
1
1
2
3
4
4
4
4
4
2
2
2
2
2
2
2
2
4
4
3
12 beepers