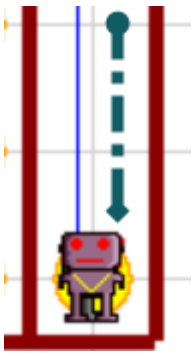
# CS101 HW#1 Report

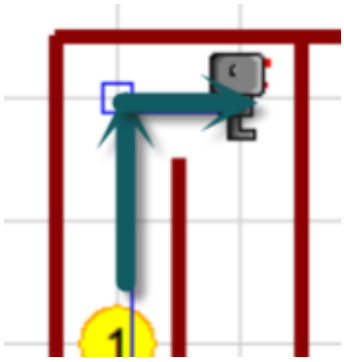
지준섭 Ji Junseop

## 1. Algorithm

### 1.1. To make Hubo get to Ami

First, we need to make Hubo reach Ami. To reach Ami, we make Hubo follow the right wall. Hubo should move to front if his right is not clear (There is wall at his right side), and should turn right if his right is clear.

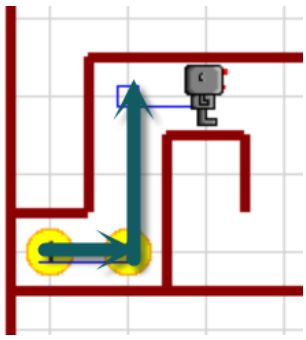




So we can write Python code to simulate this:

<code>
```
if Hubo.right_is_clear():
    Hubo_turn_right()
Hubo.move()
```
</code>

But probably there can be a wall located at the front of Hubo, so we need to avoid the wall by turning left while Hubo's front is not clear. After we do this, Hubo's front will be clear and Hubo is okay to move front.



So we can add a simple code to do this function:

<code>
```
if Hubo.right_is_clear():
    Hubo_turn_right()
while not Hubo.front_is_clear():
```
</code>

```
        Hubo.turn_left()
Hubo.move()
```
</code>

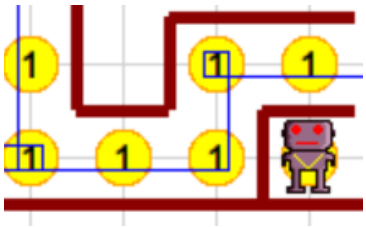
### 1.2. To make trace of Hubo for going back

To bring back Ami to place where Hubo started to move, we are recommended to make trace of the Hubo. But be careful of that the whole trace Hubo made has low efficiency because of the follow-right-wall algorithm. While Hubo is doing follow-right-wall algorithm, Hubo can meet blocked way, and he will turn around and go back. This is not efficient trace. So we should 'erase' the trace that is not really useful.

We just need simple code to make the whole trace:

<code>
```
num_of_drop = 0
if Hubo.front_is_clear():
        num_of_drop = num_of_drop + 1
if Hubo.right_is_clear():
        num_of_drop = num_of_drop + 1
if Hubo.left_is_clear():
        num_of_drop = num_of_drop + 1

for i in range(num_of_drop):
        Hubo.drop_beeper()
```
</code>

In this code, we made a variable num_of_drop to count the number of directions which Hubo can move to. After counting, Hubo drops the beepers following to num_of_drop to make trace.

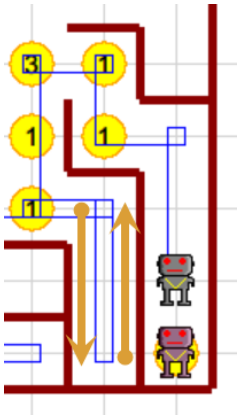


Then we should add a piece of code to 'erase' the useless trace.

<code>
```
num_of_drop = 0
if Hubo.front_is_clear():
        num_of_drop = num_of_drop + 1
if Hubo.right_is_clear():
        num_of_drop = num_of_drop + 1
if Hubo.left_is_clear():
        num_of_drop = num_of_drop + 1
```
</code>

```
if not Hubo.on_beeper():
        for i in range(num_of_drop):
                Hubo.drop_beeper()
else:
        Hubo.pick_beeper()

if num_of_drop == 0:
        Hubo_turn_around()
</code>
```

By this Python code, Hubo would drop beepers only if he is not on beeper. If he is on beeper, it means that he is now 'going back'. Then we need to erase the trace, so we make Hubo pick up the beeper.




### 1.3. To make Hubo and Ami follow the trace

Because of the condition of while part, Hubo will stop when he meets Ami. Then we need to make Hubo and Ami follow the trace to get where Hubo started from. We will make Hubo to find where the beepers are and use variable 'Hubo_turn_count' to indicate Ami.

```
<code>
Hubo_turn_count = 0

if num_of_dir == 0:
        Hubo.pick_beeper()
        Hubo_turn_count = 4

elif num_of_dir == 1:
        if Hubo.right_is_clear():
                Hubo_turn_right()
                Hubo_turn_count = 3
        elif Hubo.left_is_clear():
                Hubo.turn_left()
                Hubo_turn_count = 1
        Hubo.move()
```

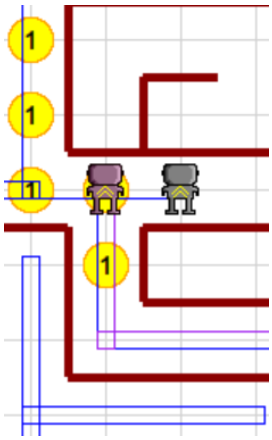
```
else:
        if Hubo.right_is_clear() and chk == 0:
                Hubo_turn_right()
                Hubo_turn_count = 3
            Hubo.move()
            chk = 1
            if not Hubo.on_beeper():
                Hubo_turn_around()
                Hubo.move()
                Hubo_turn_right()
                chk = 0

    if Hubo.front_is_clear() and chk == 0:
            Hubo_turn_count = 0
            Hubo.move()
            chk = 1
            if not Hubo.on_beeper():
                Hubo_turn_around()
                Hubo.move()
                Hubo_turn_around()
                chk = 0

        if Hubo.left_is_clear() and chk == 0:
            Hubo.turn_left()
            Hubo_turn_count = 1
            Hubo.move()
            chk = 1
            if not Hubo.on_beeper():
                Hubo_turn_around()
                Hubo.move()
                Hubo.turn_left()
                chk = 0
</code>
```

variable num_of_dir means the same thing as a num_of_drop that is the number of ways that Hubo can go to. If num_of_dir is 0, it means Hubo got to the end, so Hubo should just pick the beeper to stop the while code. Else if it is 1, Hubo will turn right or left and change the Hubo_turn_count. Else if Hubo can go to two or more ways, Hubo will check all the way where he can go. If there was no beeper, Hubo will go back and do the next instructions. Then now we will make Ami to follow Hubo by this code:

```
<code>
while Ami.on_beeper():
      Ami.pick_beeper()
```

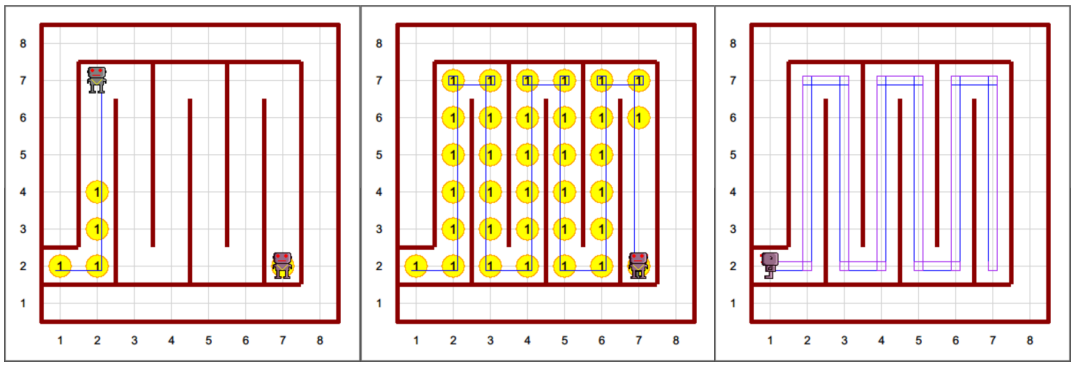
```
if Hubo_turn_count == 1:
        Ami.turn_left()
elif Hubo_turn_count == 3:
        Ami.turn_right()
if not Hubo_turn_count == 4:
        Ami.move()
</code>
```

Ami will pick beeper if there is any beeper under her. And if Hubo_turn_count is 1, it means ami to turn left. If it's 3, it means turn right. If it's 4, it means nowhere to go, so ami will move if Hubo_turn_count is not 4.

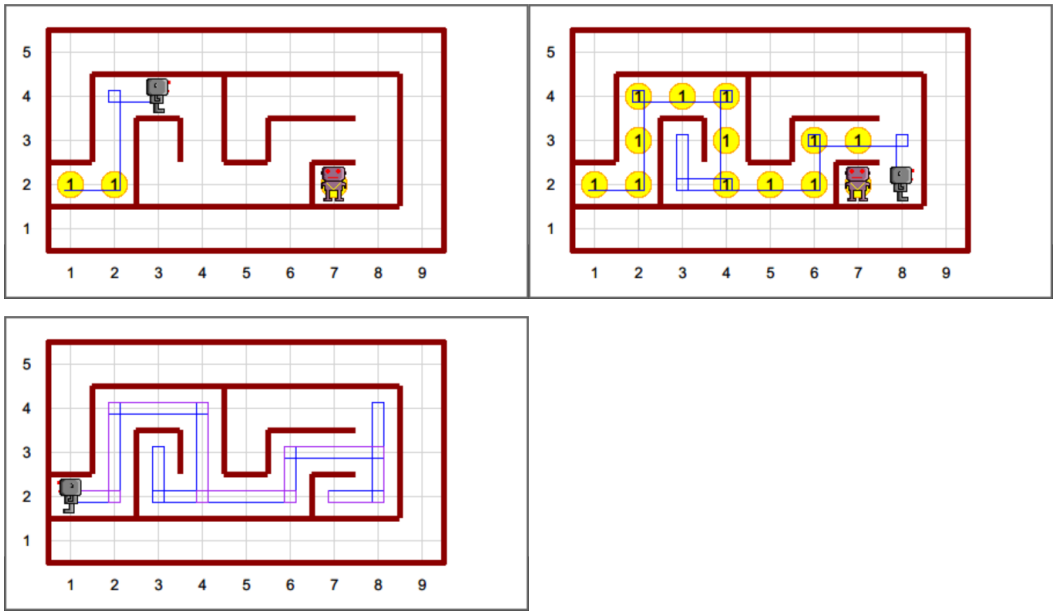


## 2. Screenshots

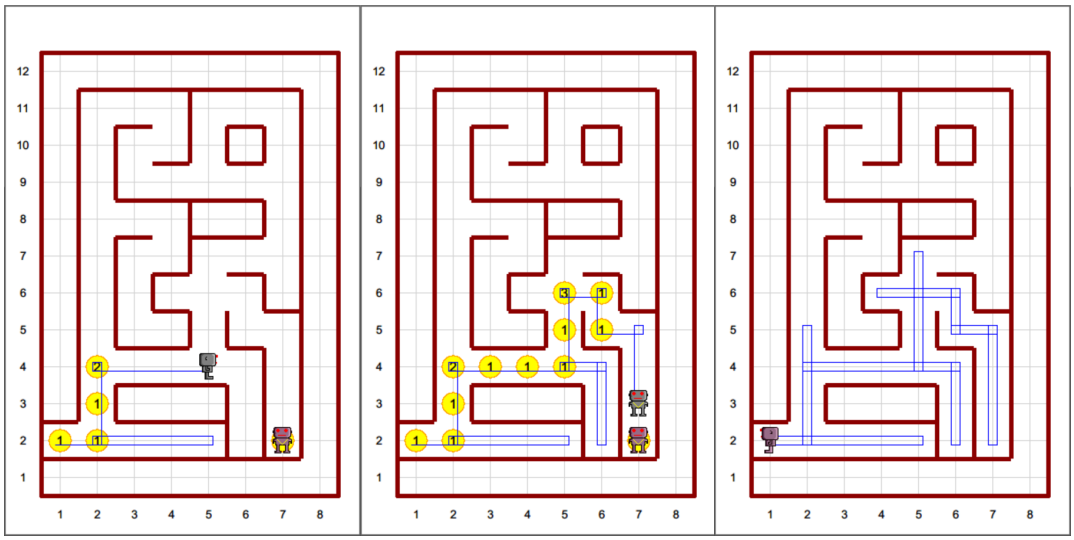
### 2.1 Maze1.wld



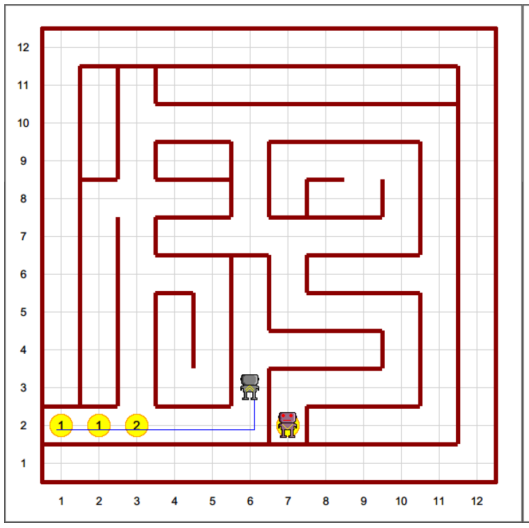
### 2.2 Maze2.wld

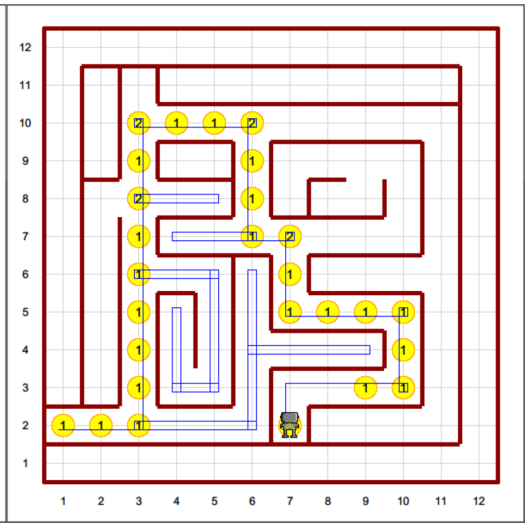


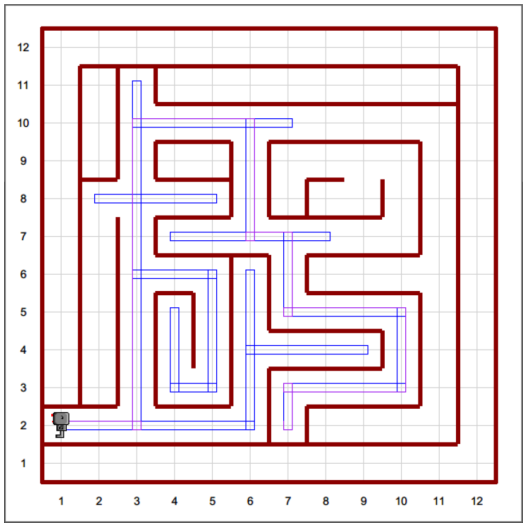
## 2.3 Maze3.wld



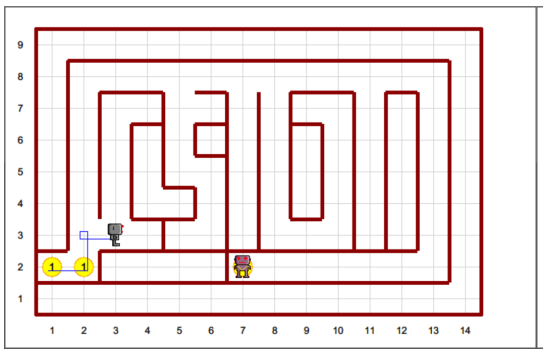
## 2.4 Maze4.wld

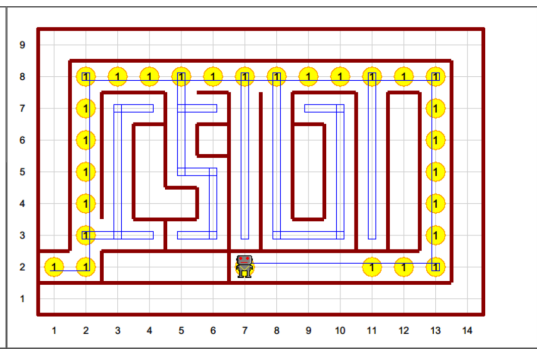


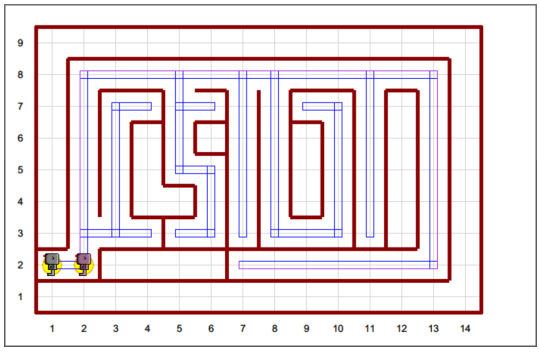




## 2.5 Maze5.wld








3. How I felt

It was interesting that I can make the Robot to find maze and go back by the trace. I think I was able to learn a little about products that runs around us. I once heard that there are not perfect algorithm for cleaning robots. Now I think I can try to make cleaning robot's algorithm more efficient.