PROGRAMMING

Lecture 04

Hanbat National University
Dept. of Computer Engineering
Changbeom Choi

OUTLINE

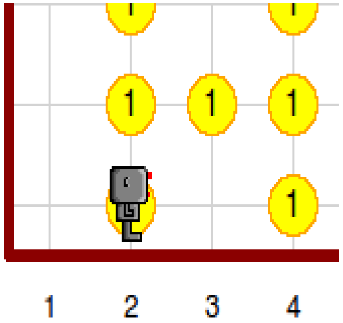
Conditionals

While-loops

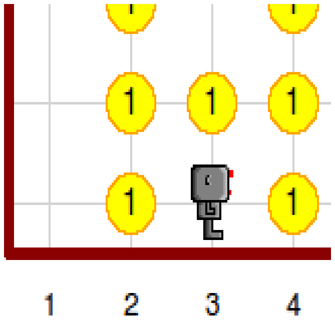
Practice

To check if there is a beeper at the current position

hubo.on_beeper()



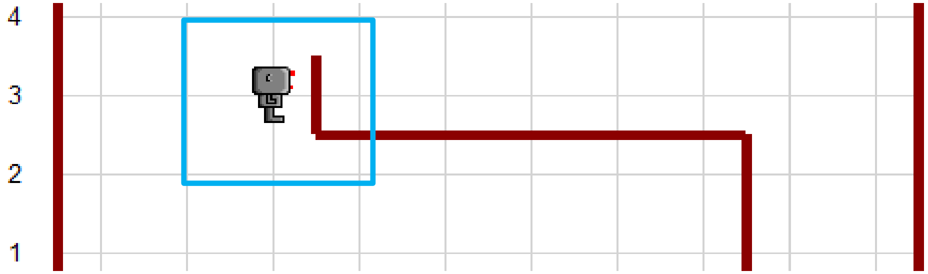
```
hubo.on_beeper()
   True
```



```
hubo.on_beeper()
   False
```

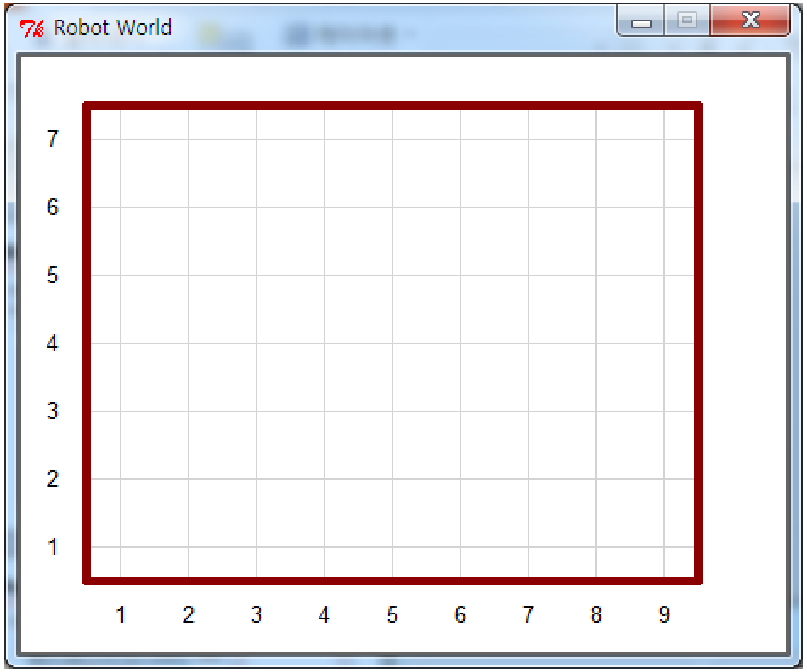
To check if there is a wall on the left, right, or front.

```
hubo.front_is_clear()  False
hubo.left_is_clear()    True
hubo.right_is_clear()  True
```



To create a grid

```
create_world(avenues = 9, streets = 7)
```



PRACTICE

Read sections 10~13 in the robot note to do the following tasks:

Harvest4 (page 6)
Plant1 (page 7)
Hurdles3 (page 8)
Hurdles4 (page 9)

conditionals

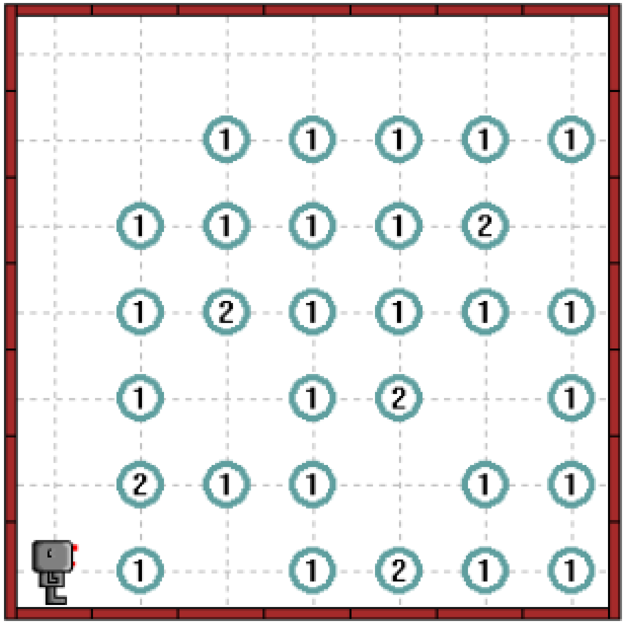
Harvest5 (page 10)
ZigZag2 (page 10)

while-loops

PRACTICE USING CONDITIONALS

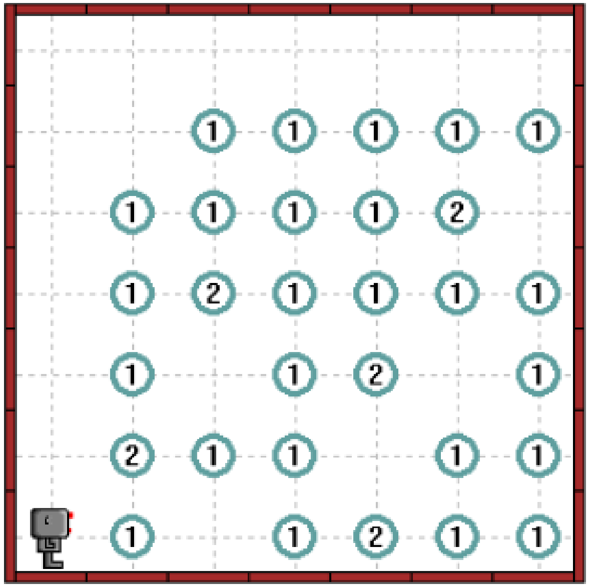
PROBLEM 1: HARVEST4

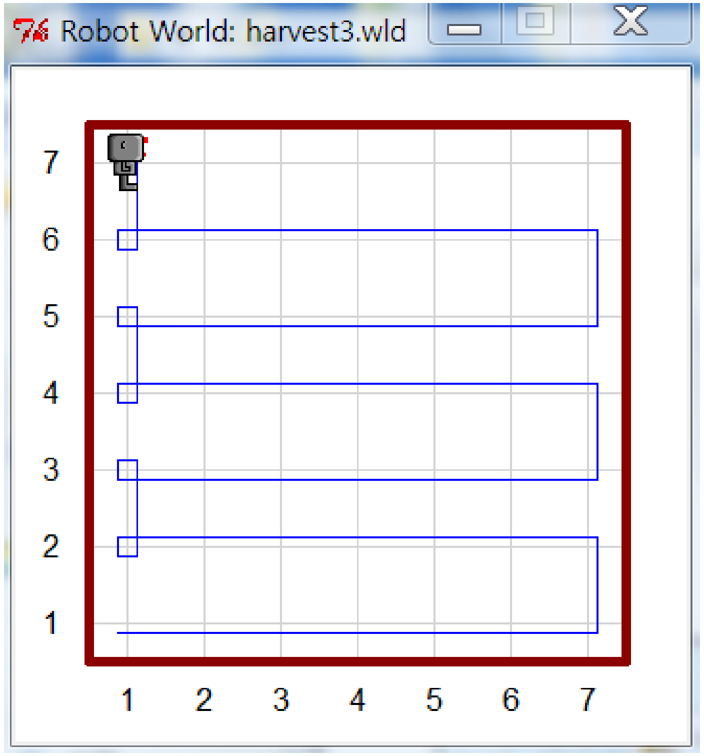
Modify your program for the Harvest2 task so that the program works for the world harvest4.wld. Note that the new program should also work for the worlds, harvest1.wld and harvest3.wld.



Load the file, harvest4:

```
load_world(“./worlds/harvest4.wld”)
```

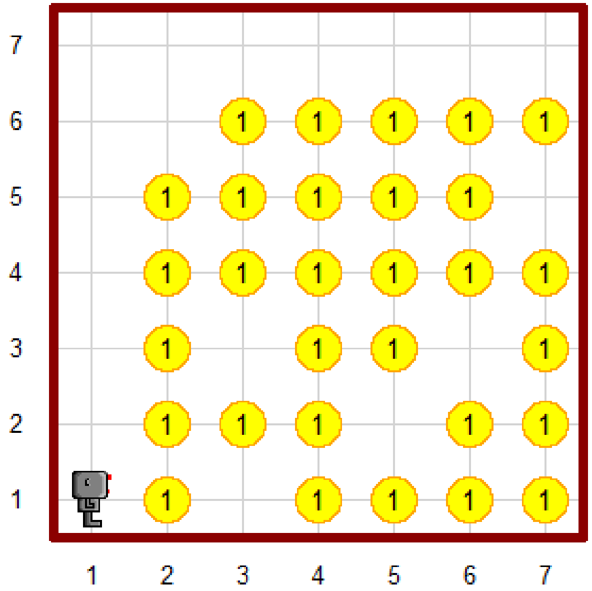



Robot World: harvest3.wld

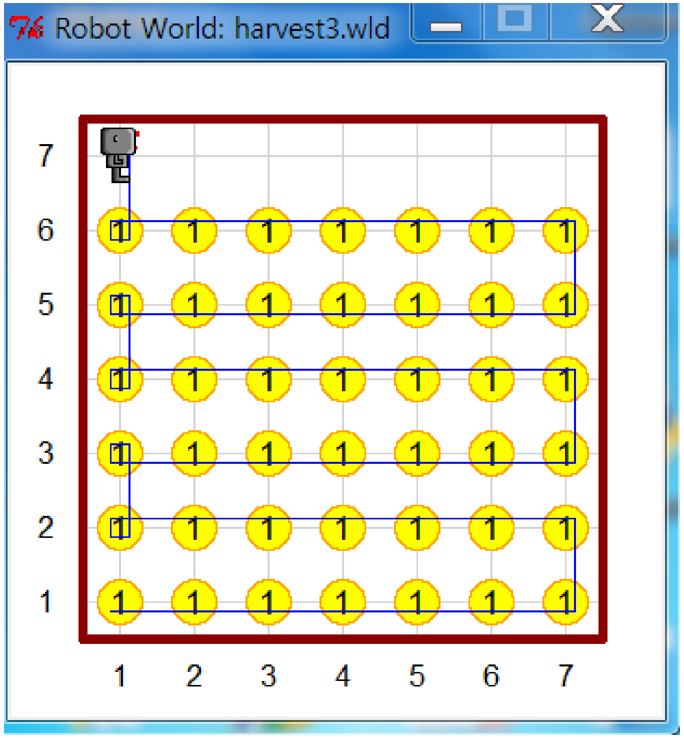
PROBLEM 2: PLANT1

Write a program that will plant a potato at every spot in the field except the top-most row at the end, as illustrated in the next slide. Hubo should skip any spot where there already is a potato. (Note that you have to create your robot with sufficiently many beepers by using hubo = Robot(beepers=36).

Try your program with an empty world and with the worlds harvest1.wld and harvest3.wld.

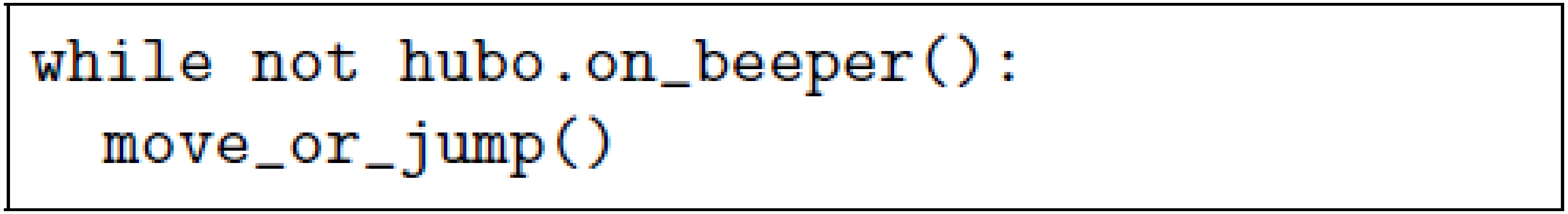


Havest3.wld


Robot World: harvest3.wld

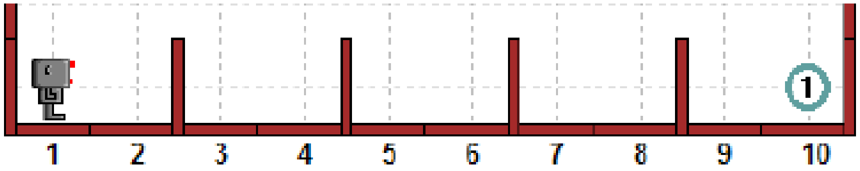
PROBLEM 3: HURDLES3

Our robot has become quite good at jumping hurdles. Hubo now enters races of different lengths: short sprints and long races. He knows that he has reached the finish line when he gets to the position with a beeper. Use a while loop so that you don't have to use a for-loop of fixed length. In other words, the core of your program should look like the following:

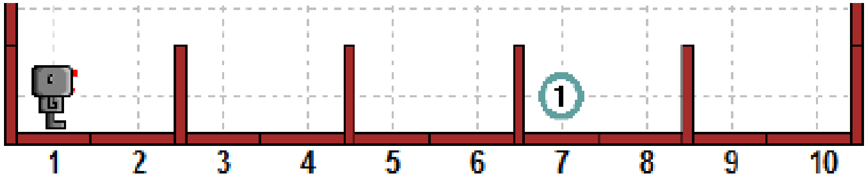


```
while not hubo.on_beeper():
  move_or_jump()
```

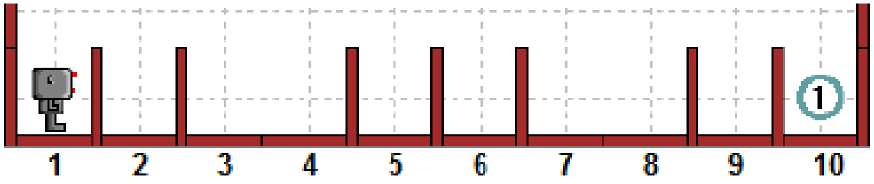
Below, you will find three such race courses; the world files are hurdles1.wld, hurdles2.wld, and hurdles3.wld :



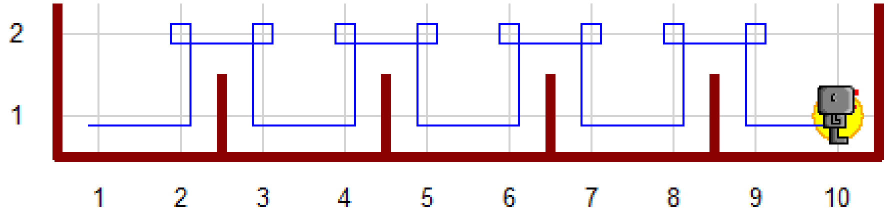
hurdles1.wld



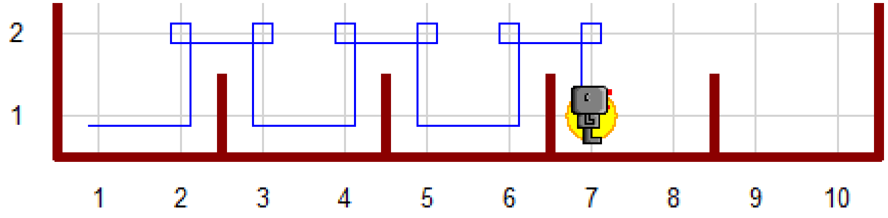
hurdles2.wld



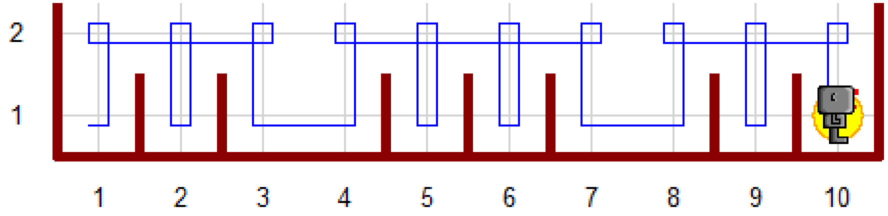
hurdles3.wld



hurdles1.wld



hurdles2.wld



hurdles3.wld

How to write a function jump_one_hurdle() for jumping over a hurdle ?

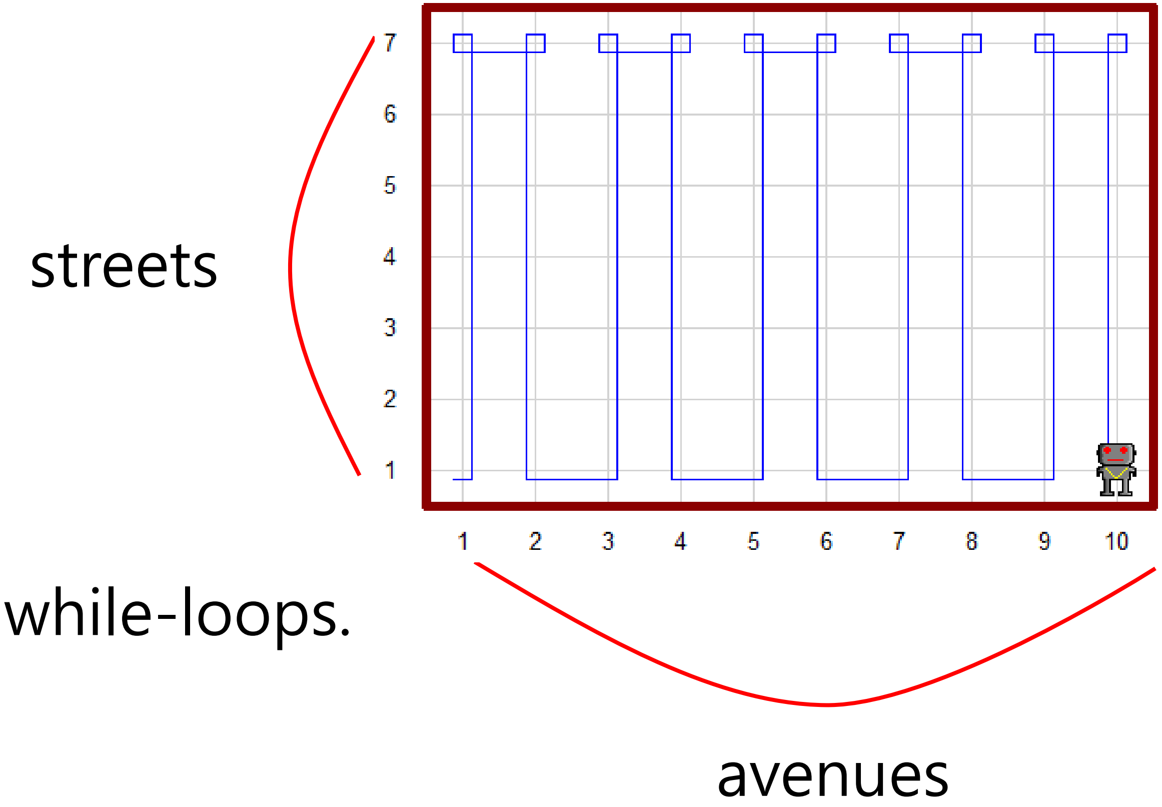
PROBEM 4: ZIGZAG2

Rewrite your program for ZigZag1 so that the robot can visit an empty world of any size in zigzag fashion. Make sure that the program works for even and odd numbers of streets and avenues. (You can assume that there are at least two streets and at least two avenues.)

Use randint() in module random to generate random numbers, e.g., random.randint(2, 20).

```
import random
x = random.randint(2, 20)
y = random.randint(2, 20)
create_world(avenues = x, streets = y)
hubo = Robot()
hubo.set_trace("blue")
hubo.set_pause(.1)
```

The number of avenues is even or odd. So is the number of streets.



Use while-loops.

Summary

Through **2D robot control** we learned:

- **conditionals**: `if, if~else, and if~elif~else`
- **iterations**
  - for-loops
  - while-loops
- **assignment**, e.g., `hubo = Robot()`
- **functions**

You have picked up the main constructs for programming.