PROGRAMMING

Lecture 06

Hanbat National University
Dept. of Computer Engineering
Changbeom Choi

OUTLINE

Case study: Photo processing

Color conversion

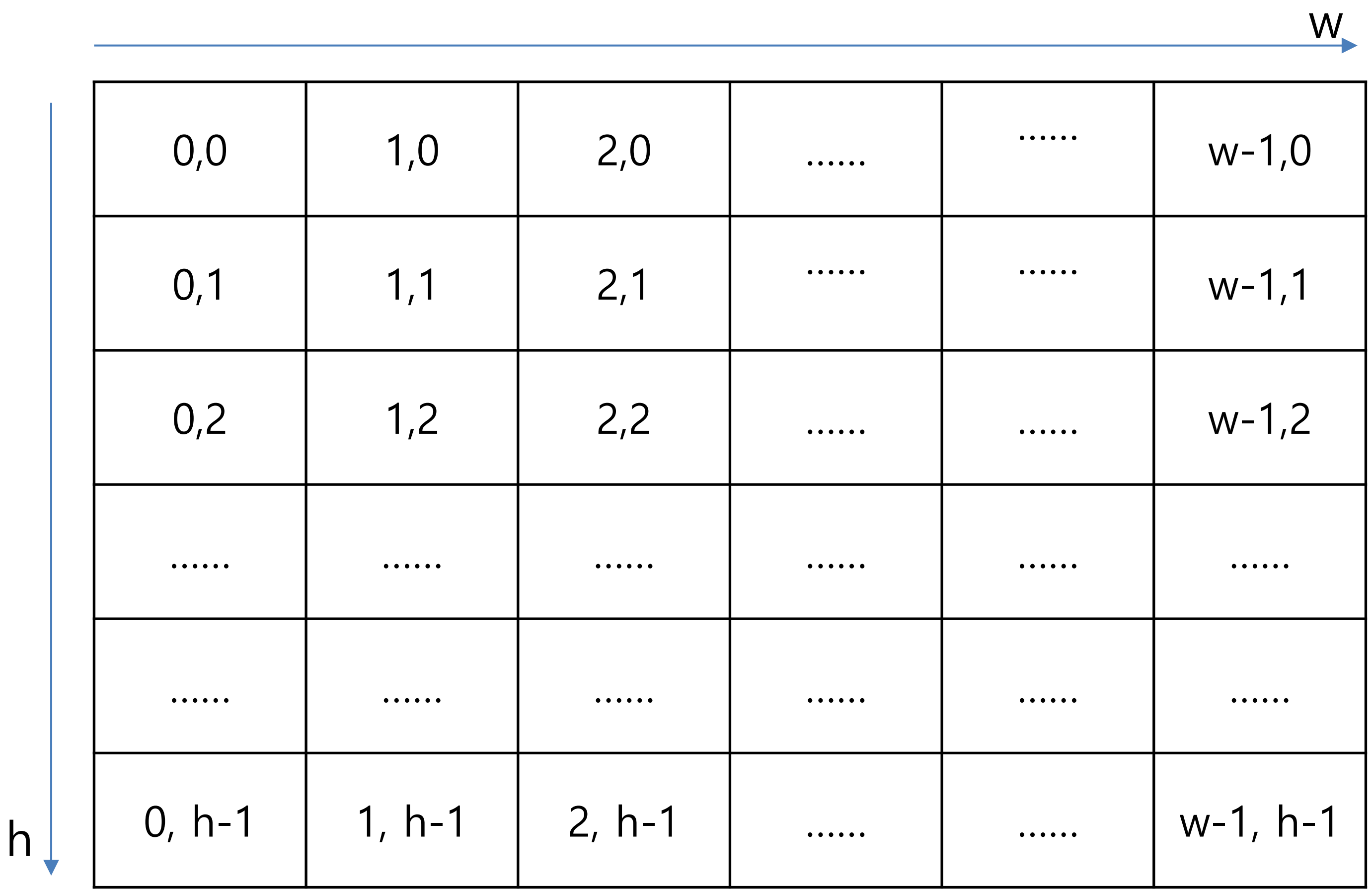
More practice through 2D robot control

- Conditional + while-loop

CASE STUDY: PHOTO PROCESSING

Reference : Otfried Cheong, photo processing with cs1media

pixel coordinates (x, y), $0 \leq x < w$, $0 \leq y < h$



0,0	1,0	2,0	……	……	w-1,0
0,1	1,1	2,1	……	……	w-1,1
0,2	1,2	2,2	……	……	w-1,2
……	……	……	……	……	……
……	……	……	……	……	……
0, h-1	1, h-1	2, h-1	……	……	w-1, h-1

Colors are often represented as a tuple with three elements that specify the intensity of red, green, and blue light:

```
red = (255, 0, 0)
green = (0, 255, 0)
blue = (0, 0, 255)

white = (255, 255, 255)
black = (0, 0, 0)

yellow = (255, 255, 0)
purple = (128, 0, 128)
```

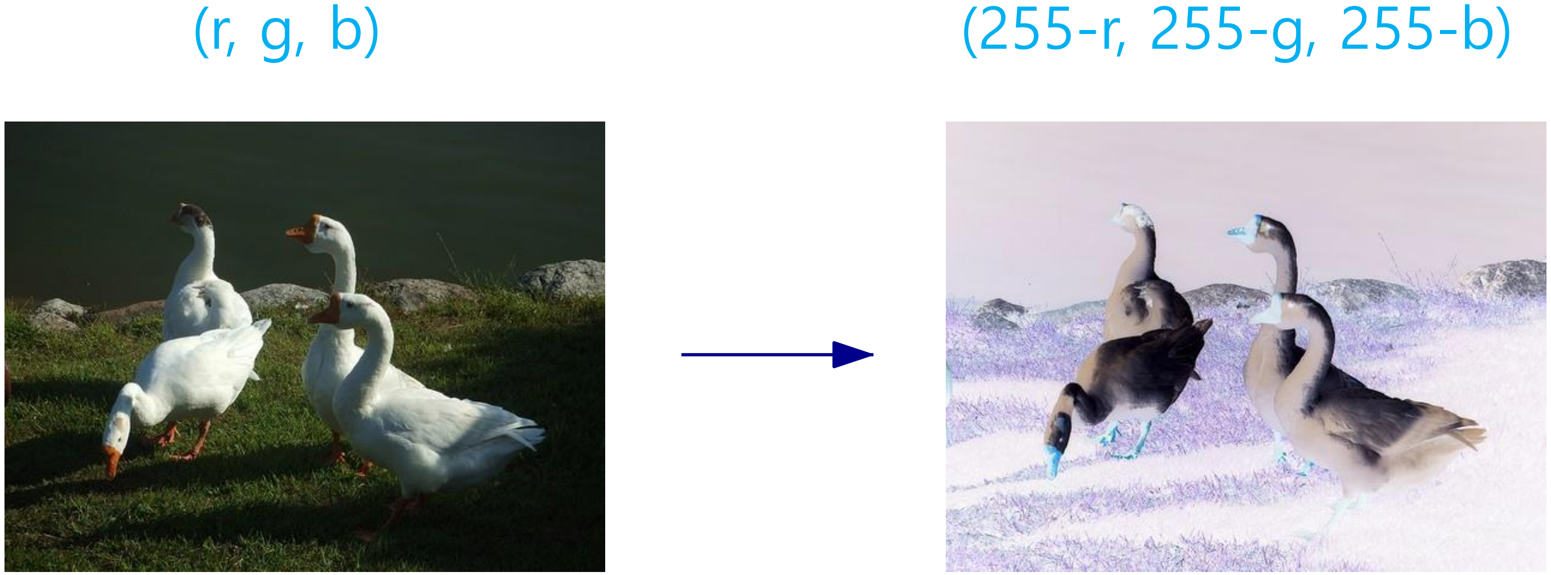
OBJECTS: VALUES AND TYPES

```
from cs1media import *
img = create_picture(300, 300, “purple”)
img.show()
img.set_pixels(“yellow”)
img.show()
>>> img.get(250, 188)
(255, 255, 0)
>>> img.set(250, 188, (255, 0, 0))
>>> img.get(250, 188)
```

```
“yellow”: (255, 255, 0)
“purple”: (128, 0, 128)
```

red, green, blue triples

NEGATIVE FILM



(r, g, b)

(255-r, 255-g, 255-b)

```
from cs1media import *
img = load_picture(“images/ninja.jpg")
w, h = img.size()
for y in range(h):
  for x in range(w):
    r, g, b = img.get(x, y)
    r, g, b = 255 - r, 255 - g, 255 - b
    img.set(x, y, (r, g, b))
img.show()
```

BLACK AND WHITE

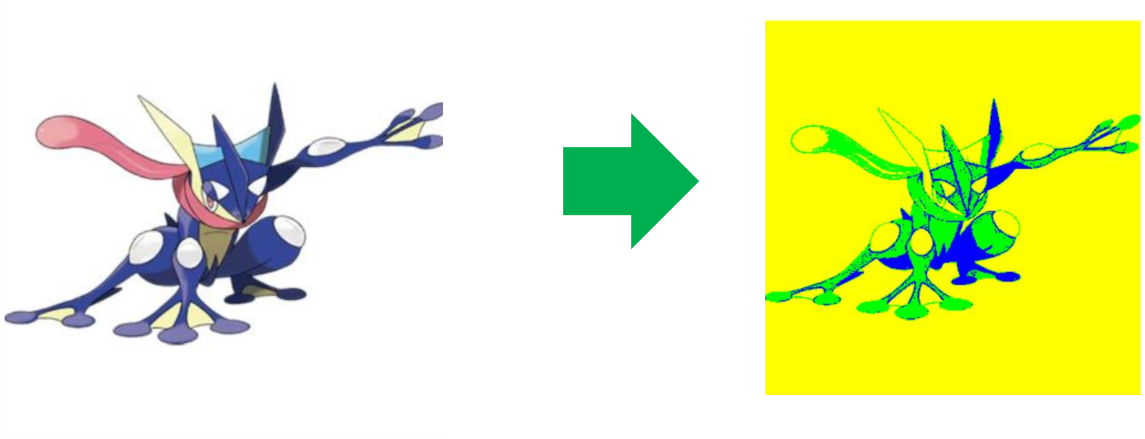
```
w, h = img.size()
for y in range(h):
 for x in range(w):
  r,g,b = img.get(x,y)
  v = (r + g + b) / 3.0
  if v > 100:
    img.set(x, y, white)
  else:
    img.set(x, y, black)
img.show()
```

threshold

$(0 \leq v \leq 255)$

```
white = (255, 255, 255)
black = (0, 0, 0)
```

COLOR COVERSION

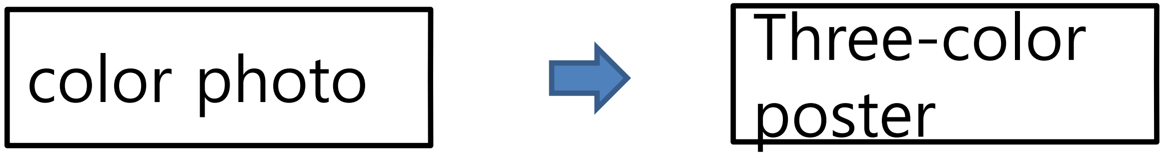


PROBLEM 1: THREE_COLOR POSTER*
In the previous lecture, you learned a program that convert a color image to a black-and-white image. Modify that program to convert a color image to a three-color poster.

Use ninja.jpg as the color image.

How to convert pixels

bright pixels	yellow
dark pixels	blue
neither bright nor dark	green.

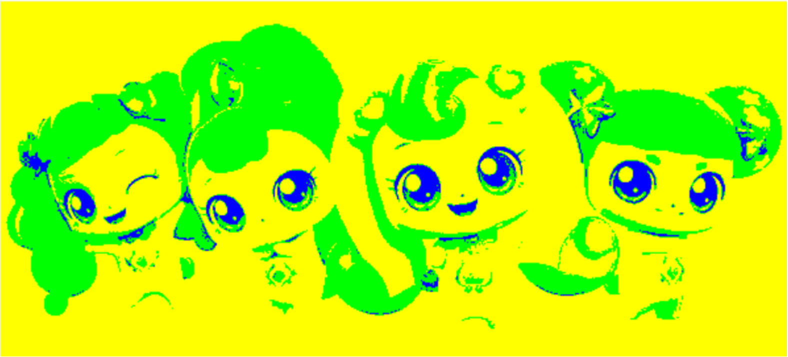


color photo

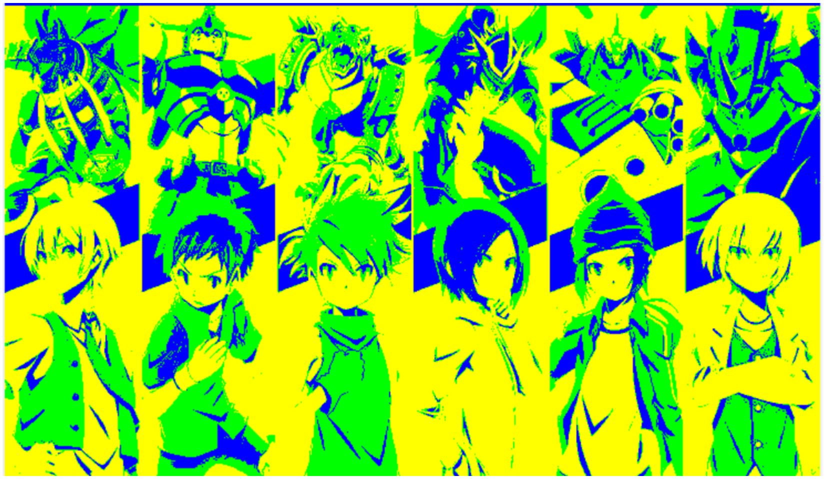
Three-color poster

```
from cs1media import *
img = load_picture( “images/ninja.jpg”)

white = (255, 255, 255)
black = (0, 0, 0)
threshold = 100    Modify here

w, h = img.size()
for y in range(h):
  for x in range(w):
     r, g, b = img.get(x, y)
     v = (r + g + b) / 3.0
     if v > threshold:
        img.set(x, y, white)           Modify here.
     else:
         img.set(x, y, black)
img.show()
```

Result with tinyping.jpg



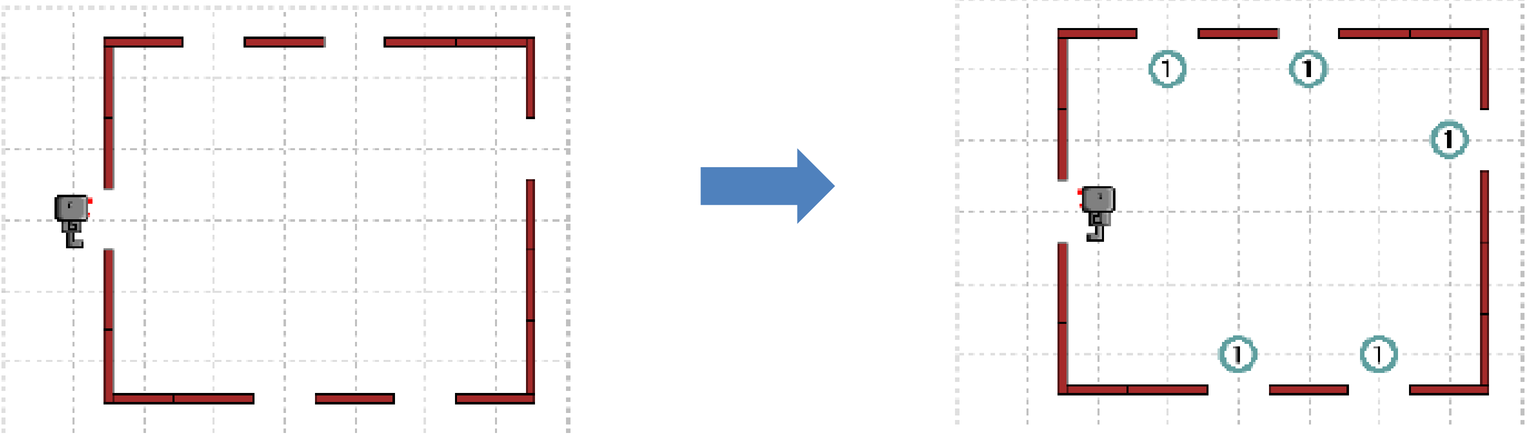
Result with digimon.jpg

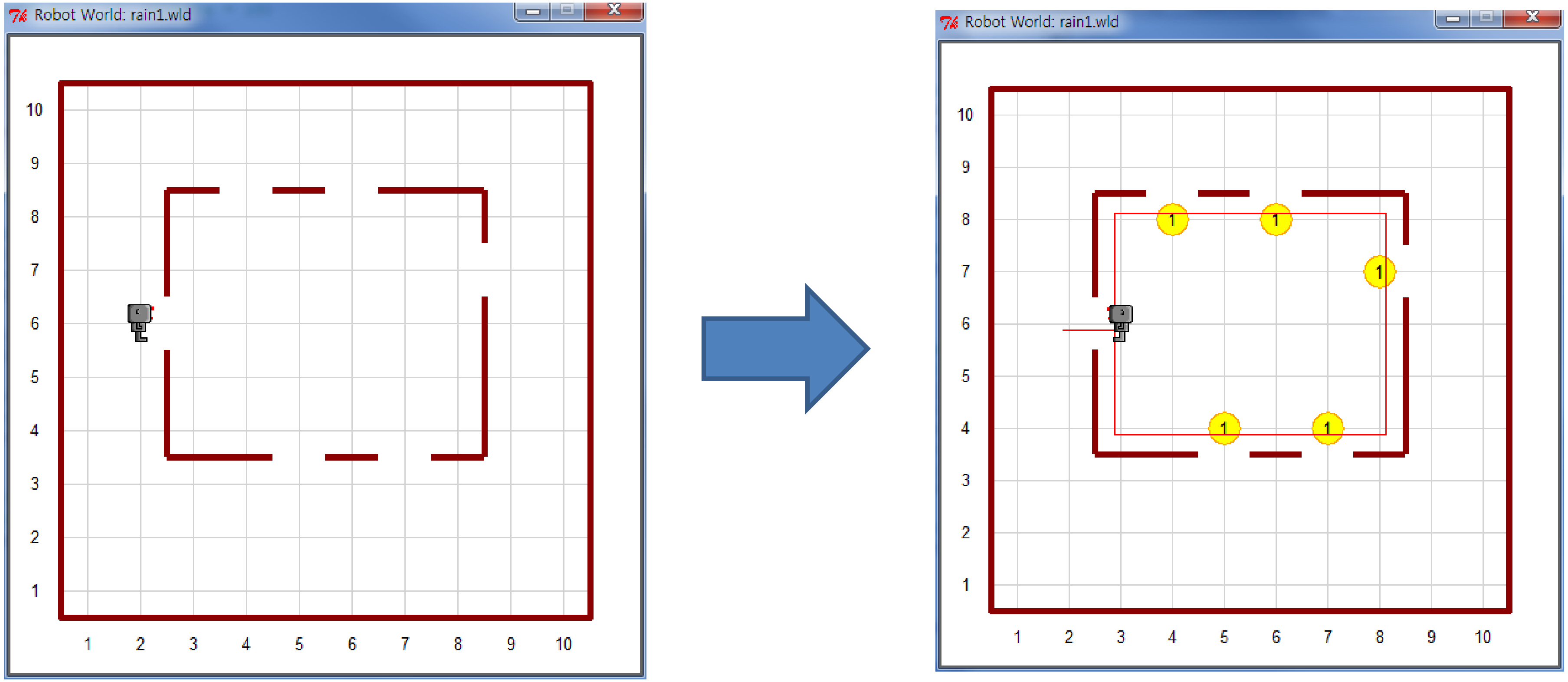


MORE PRACTICE(through 2D robot control)

PROBLEM 2-1: RAIN1

It was a beautifully sunny day. Hubo is playing outside with friends. Suddenly, it started to rain. He remembered that the windows in his house were all open. So he went back to the house and stopped in front of the door. Help Hubo close the windows of the house. A closed window has a beeper in front of it.





Starting at position (2,6) and facing east.

Dropping a beeper in order to close a window

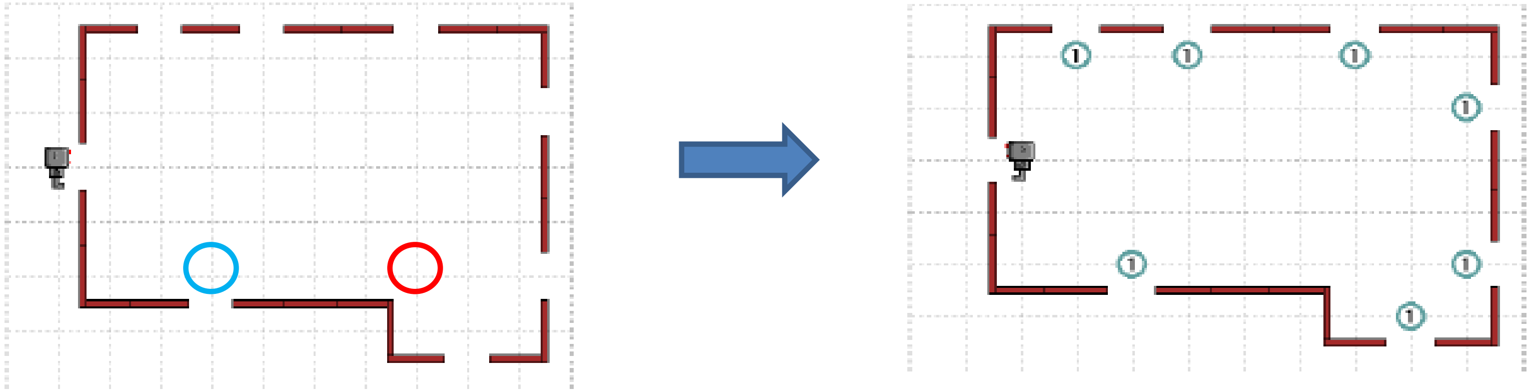
Use rain1.wld as the world file.

Pseudo code

```
1. mark the starting point and move forward:
      Move forward and mark the starting point.
      Turn right and move forward.
2. Close all windows:
      While not returning to the starting point:
          If there is a window, close it.    How?
          If the front is clear, move forward.
          Otherwise turn left.
3. Unmark the starting point and finish.
                                            turn to the west
```

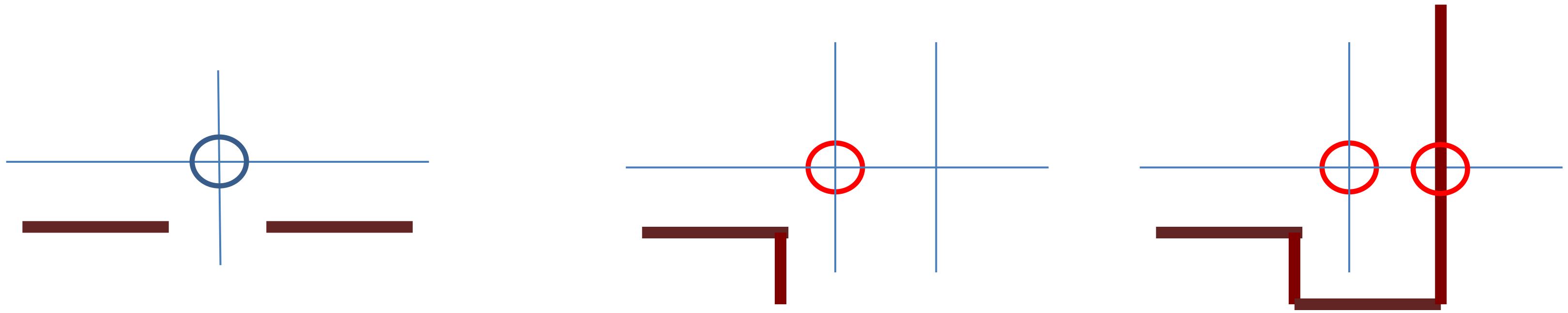
PROBLEM 2-2: RAIN2

Ami, Hubo's friend, lives in a bigger house. Ami was playing outside with Hubo when it started raining. Help Ami close the windows in her house.



How to check a window ?

How to find a window

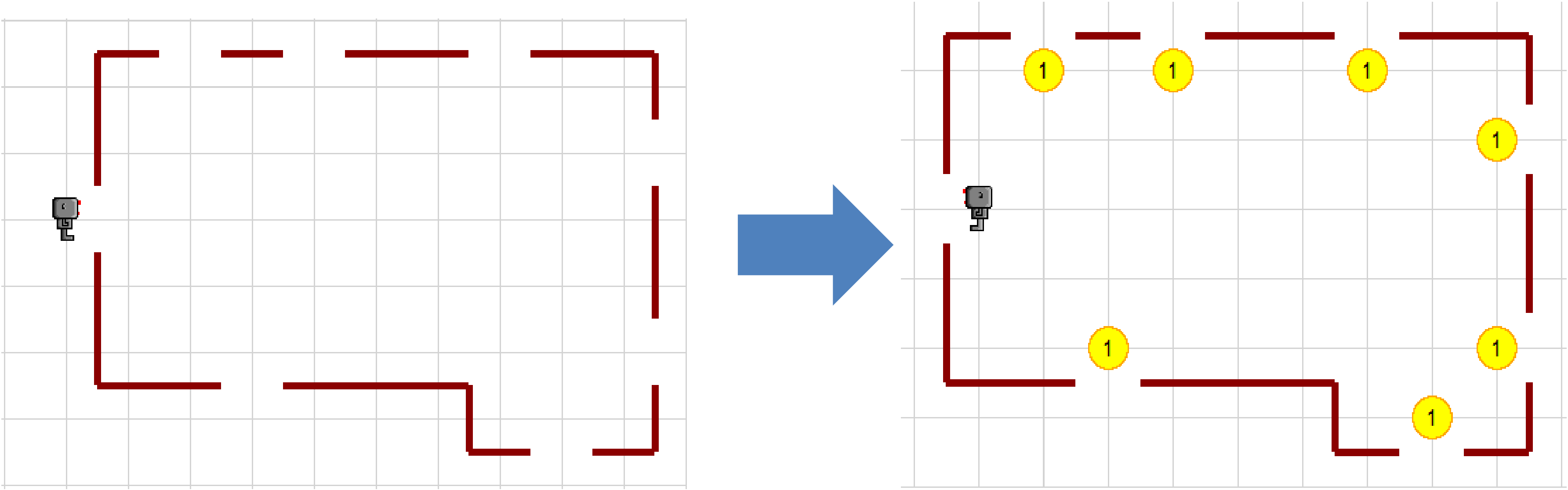


```
If right is clear,
    If there is a window,
        close a window.
```

is-window()

```
flag = False
If front is clear,
    move forward.
    If right is blocked,
        flag = True
    move backward.
```

.

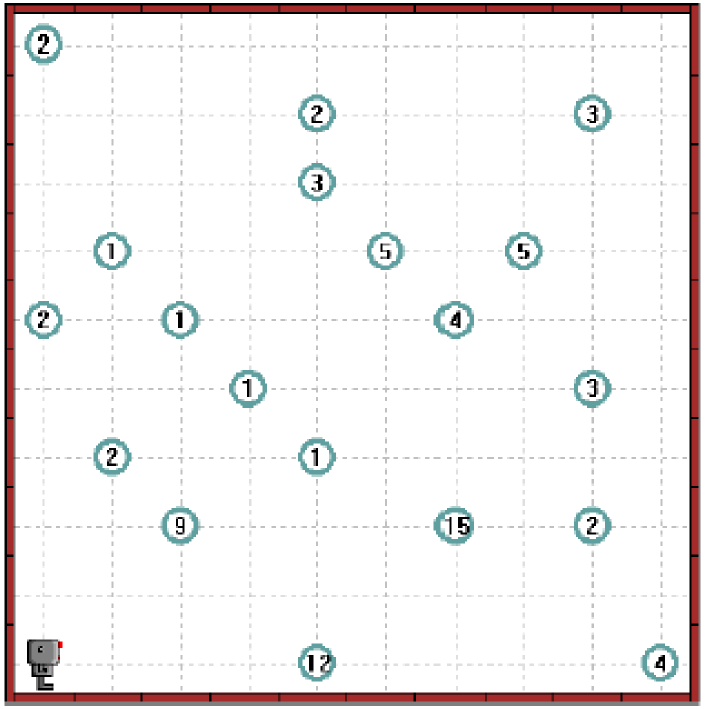


Start at position (2,6) with facing east.
Use rain2.wld as the world file.

Your program must work for both rain1.wld and rain2.wld

PROBLEM 3: TRASH2

Hubo's parents ask him to pick up all the garbage that got blown away in their backyard during the windstorm. Help him pick up all the garbage and bring it back with him to his starting position. Create your own world file such as the one illustrated in the left figure.

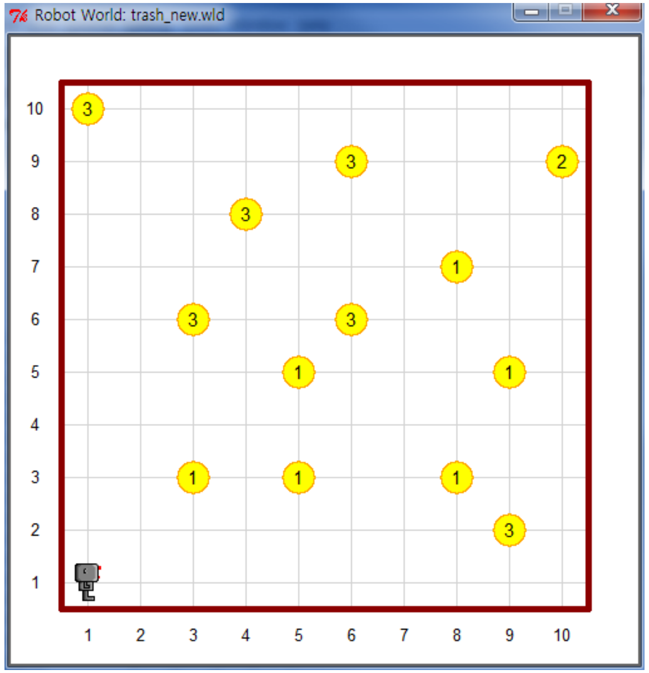


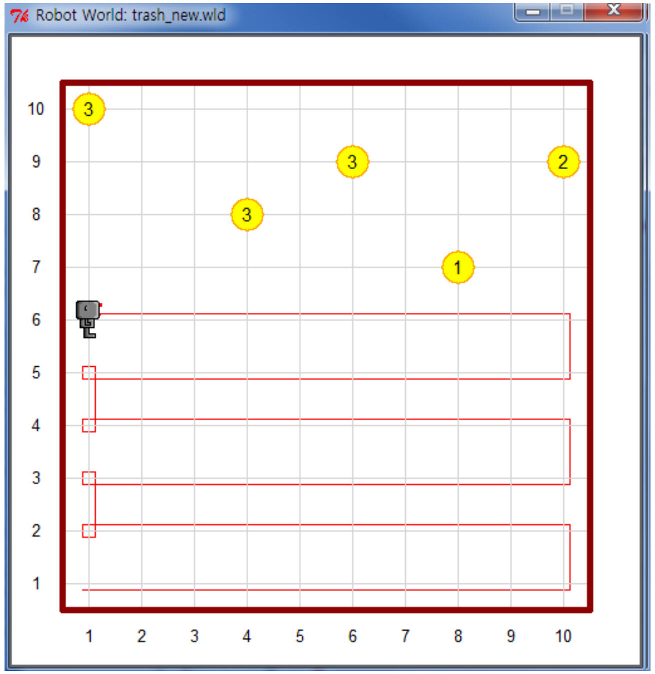
Notice that the location of any beeper is not given in advance. The size of the backyard is not given, either.

To create a backyard with litter, use edit_world in Section 7.

```
from cs1robots import *
import random

x = random.randint(5, 15)
y = random.randint(5, 15)

create_world(avenues = x, streets = y)
edit_world()
save_world("myworld.wld")
```

HOW TO CLEAN THE BACKYARD



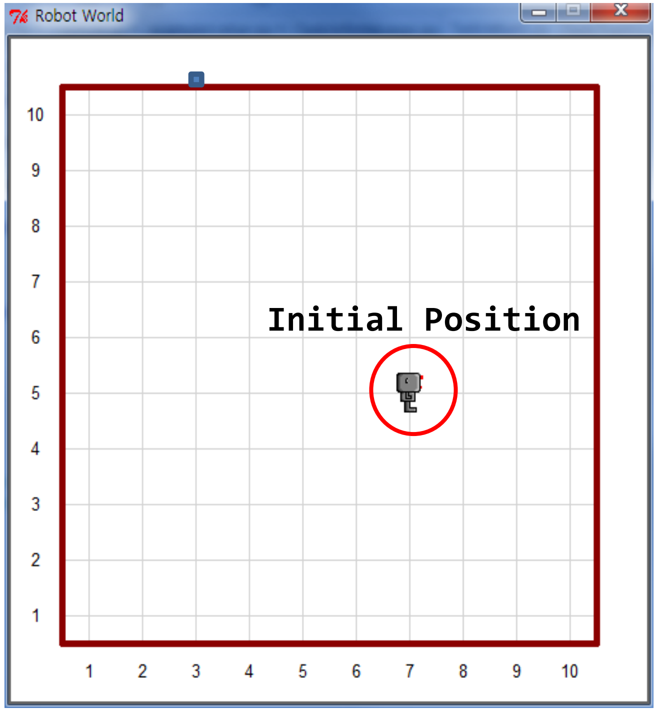


PROBLEM 4: RETURN

Write a program that will allow Hubo to return to his **usual starting position and orientation** (avenue 1, street 1, facing east), starting from **any position and orientation** in an empty world. You can create a robot with a given position and orientation like this:

```
hubo = Robot(orientation =“E”, avenue =7, street =5)
```

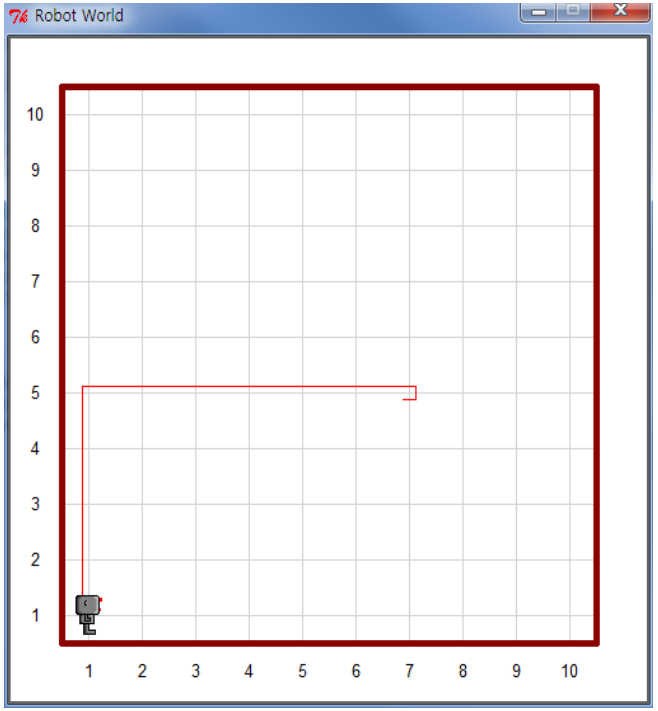
“E”, “W”, “S”, “N”



Initial Position

(orientation, avenue, street)

"E" 7 5



(orientation, avenue, street)

"E" 1 1

ORIENTATION AND POSITION

```
f = random.randint(1,4)
if f == 1:
    f = "E"
elif f == 2:
    f = "W"
elif f == 3:
    f = "S"
else:
    f  = "N"
```

```
from cs1robots import *
import random

x = random.randint(1,10)
y = random.randint(1,10)
```

orientation

position

```
hubo = Robot(orientation = f, avenue = x, street =y)
```