1 (20)	2 (20)	3 (20)	4 (20)	5 (20)	TOTAL (100)

[CS101] Introduction to Programming 2014 Fall - Midterm Examination

SECTION	STUDENT ID	NAME

※ Please check if you received all 19 pages of the test material.
※ 시작하기 전에 반드시 페이지의 수를 확인 하십시오.(전체 : 19쪽)

※ Fill in your student identification number and name. Otherwise you will lose 1 point for each missing piece of information.
※ 위의 정보(학번,이름)를 정확히 기입하지 않을 경우, 각 실수 당 1점이 감점 됩니다.

※ **TAs will not answer your questions about the exam**. If you think that there is anything ambiguous, unclear or wrong about a problem, please write the reasons and make necessary assumptions to solve the problem. We will take your explanation into consideration while grading.
※ **시험시간동안 질문을 받지 않습니다**. 만일 문제에 오류나 문제가 있을 경우, 왜 문제가 이상이 있다고 생각하는지에 대해서 기술하시면 되겠습니다. 또한 문제가 애매하다고 생각되는 경우 문제를 푸실 때 본인이 생각하는 가정을 함께 작성하셔서 문제를 푸시면 되겠습니다. 채점 시 가정 및 설명을 고려하도록 하겠습니다.

1. (20 points) Answer each question according to the following instruction.

1-1. (2 points) There are two *kinds of objects* in a Python program depending on whether their states can be changed or not. What are the kinds of objects in the Python program?

(1) Objects whose state **can be never** change: **I___________________________ objects (1 point)**

(2) Objects whose state **can be** change: **M____________________________ objects (1 point)**

1-2. (2 points) There are two *kinds of variables* in a Python program depending on where they are defined and how long they exist. What are the kinds of variables in the Python program?

(1) Variables defined outside of a function: **G____________________________ variables (1 point)**

(2) Variables that only exist during the execution of a function:

L_____________________________ variables (1 point)

1-3. (4 points) *Computational thinking* is for solving problems with a computer. What are the popular strategies used in computational thinking to develop an algorithm to solve a problem? Please explain at least one of the strategies that you learned in the course.

____

1-4. (4 points) There are three *kinds of errors* that can be generated from a computer program. What are the kinds of errors that you will receive when you do not put a proper indentation in your Python program?

____

1-5. (4 points) What is the result of the following expression in Python? You can also specify the order of calculation in the square boxes ☐ below the expression to get partial points even when you give a wrong answer.

```
10 + 9.0 * 8.0 // 7 % (6 // 5 ** (4 - 3)) * 2 + 1
   ☐    ☐    ☐   ☐    ☐    ☐    ☐   ☐   ☐
```

____

1-6. (4 points) The following is a function that returns the current weather condition of a city. Here, it is assumed that the parameters of the function are string values.

```
def getCurrentWeather(country, city):
   # implemented somehow
   return temperature, skyCondition, wind
```

Please write a statement to call this function with proper arguments.

____

2. (20 points) Please to write a program that creates a Robot, named 'hubo', and makes the robot clean a house by collecting every litter (i.e., beepers) in the house. Each room is of rectangular form and surrounded by consecutive walls and doors. A door is represented as two vertical walls in the same horizontal line. In addition, it is assumed that there is no obstacle in a room except the litter.

Answer each question according to the following instruction.

2-1. (7 points) Complete the function '*move_and_pick*' by filling in the blank **#2-1** so that hubo picks all the beepers on it, and then move one step forward.

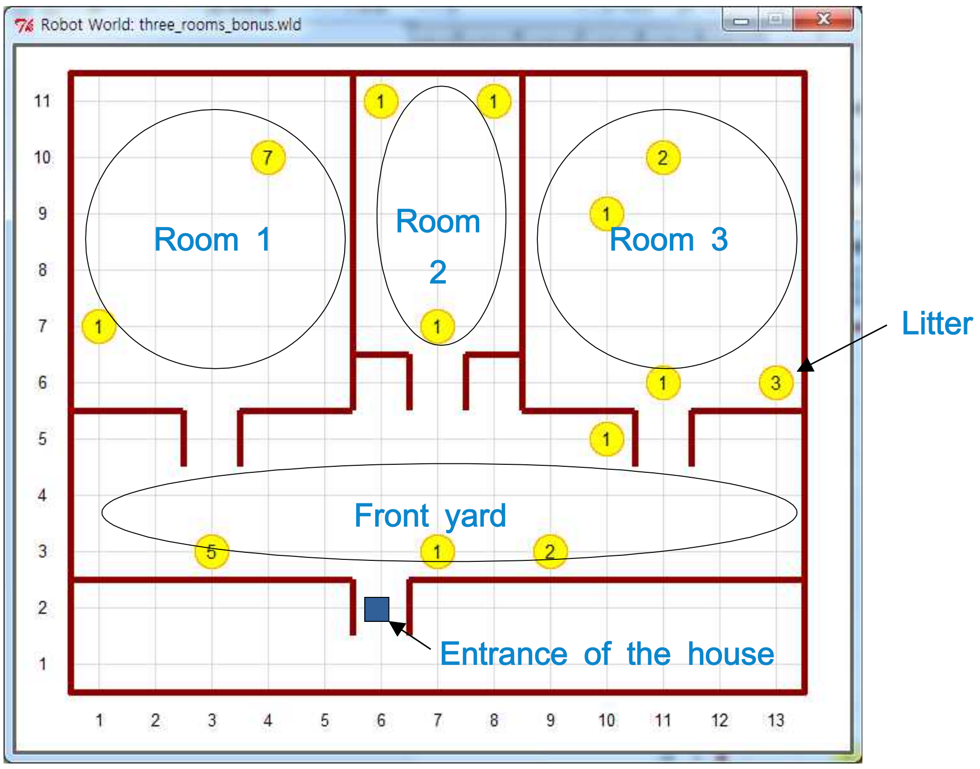
```
def move_and_pick():
```

2-2. (5 points) Fill in the blank **#2-2** to make the program correctly work. Notice that the conditional expression(s) should be evaluated when 'hubo' is in-between the door facing south or north.

```
(                    or                    )
```

2-3. (3 points) Assume that the answer of Question #2-2 is '(A or B)' with conditional expressions 'A' and 'B'. Rephrase 'not (A or B)' while the meaning of it remains the same. Note that no parentheses is allowed except the ones used in the function invocation.

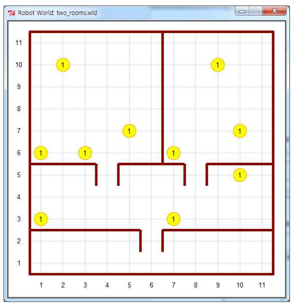
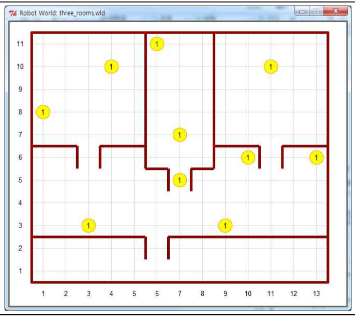
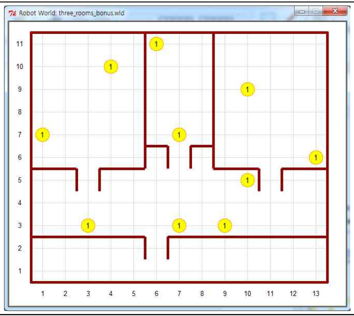
2-4. (5 points) Complete the program by filling **#2-4**.
Please write a few lines of code to make hubo pick every litter in a house using the previously defined functions.



Here are the functions of a Robot object that can be used in your answers.

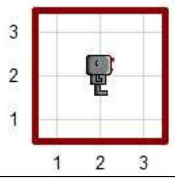
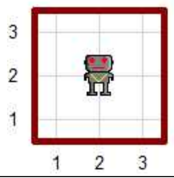
Function	Description
move	make a robot move one step forward
turn_left	make a robot turn to the left (without move)
front_is_clear	return *True* if a robot can move one step forward, *False* otherwise
left_is_clear	return *True* if there is no obstacle in the left side of a robot, *False* otherwise
right_is_clear	return *True* if there is no obstacle in the right side of a robot, *False* otherwise
carries_beepers	return *True* if a robot has at least one beeper, *False* otherwise
on_beeper	return *True* if there is at least one beeper at the position where a robot is on, *False* otherwise
pick_beeper	make a robot pick one beeper at the position where a robot is on
drop_beeper	make a robot drop one beeper at the position where a robot is on
facing_north	return *True* if a robot's facing direction is north, *False* otherwise

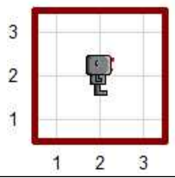
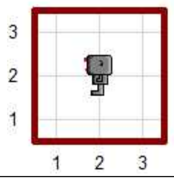
The program should work for arbitrary form of houses (worlds) satisfying aforementioned requirements. Example worlds are as follows:

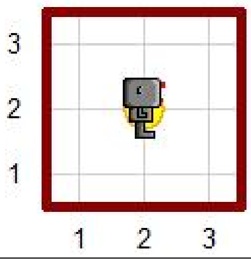
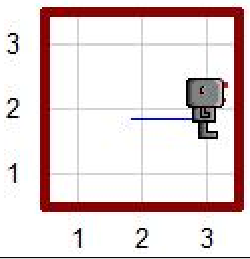
Example_worlds.wld		
		

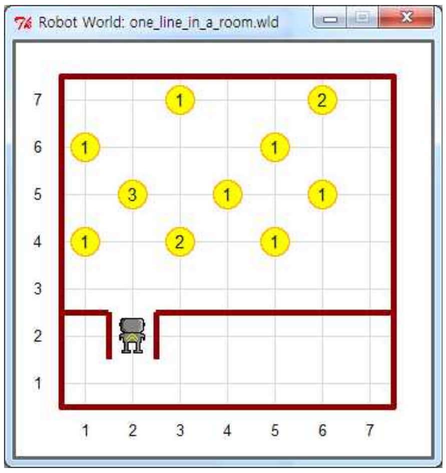
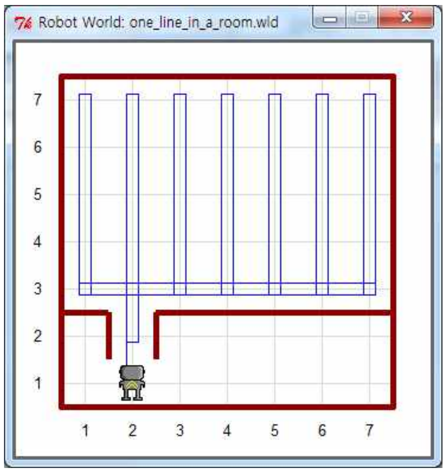
There are user-defined functions that can be used to solve the problem.

[Function definitions]

turn_right		
Description	make hubo turn right (without move)	
Example	Before	After
		

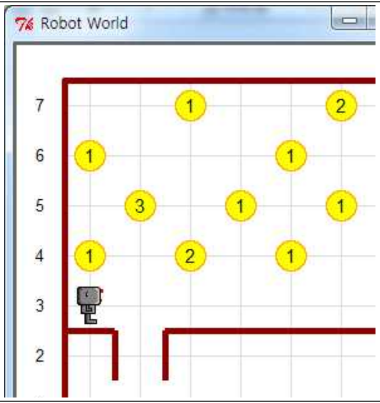
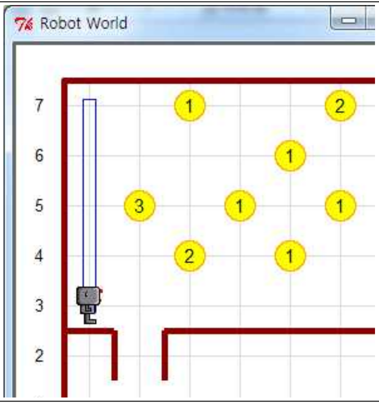
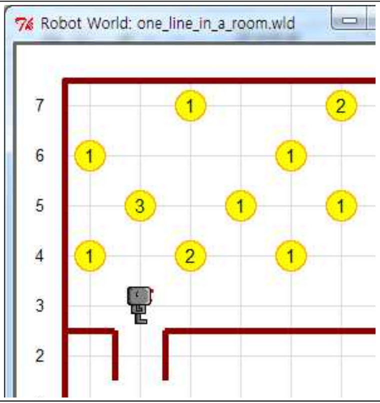
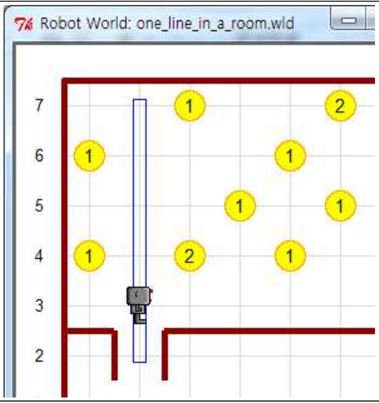
turn_around		
Description	make hubo turn around (without move)	
Example	Before	After
		

pick_and_move		
Source code	**def** pick_and_move(): **#2-1**	
Description	make hubo pick all beepers at the position where hubo is on and move one step forward	
Example	Before	After
		

<table>
<tr><th colspan="3">one_room</th></tr>
<tr><td>Source code</td><td colspan="2"><pre>
def one_room():
  hubo.move()
  hubo.turn_left()
  while hubo.front_is_clear():
    hubo.move()
  turn_around()
  one_line_in_a_room()
  while hubo.front_is_clear():
    hubo.move()
    one_line_in_a_room()
  turn_around()
  while not hubo.left_is_clear():
    hubo.move()
  hubo.turn_left()
  hubo.move()
  hubo.move()
  turn_around()
</pre></td></tr>
<tr><td>Description</td><td colspan="2">When this function is called, hubo is at in-between a door (with hubo's direction north). First, let hubo enter a room and move it to left-bottom corner of the room. Then, after making hubo face east, order hubo to gather all beepers by invoking 'one_line_in_a_room' function. Finally, move hubo to outside of the room (one step back from the door).</td></tr>
<tr><td rowspan="2">Example</td><td>Before</td><td>After</td></tr>
<tr><td>
Robot World: one_line_in_a_room.wld</td><td>
Robot World: one_line_in_a_room.wld</td></tr>
</table>

one_line_in_a_room		

Source code	**def** one_line_in_a_room():
	  hubo.turn_left()
	  while hubo.front_is_clear():
	    pick_and_move()
	  while hubo.on_beeper():
	    hubo.pick_beeper()
	  turn_around()
	
	  while hubo.front_is_clear() **and** ($\boxed{\textbf{\#2-2}}$):
	    hubo.move()
	
	  if not $\boxed{\textbf{\#2-2}}$:
	    turn_around()
	    hubo.move()
	    turn_right()
	  else:
	    hubo.turn_left()	

Description	make hubo move following a vertical line up and down while picking up all beepers in the line	
Examples	Before	Afer
		
		

one_line

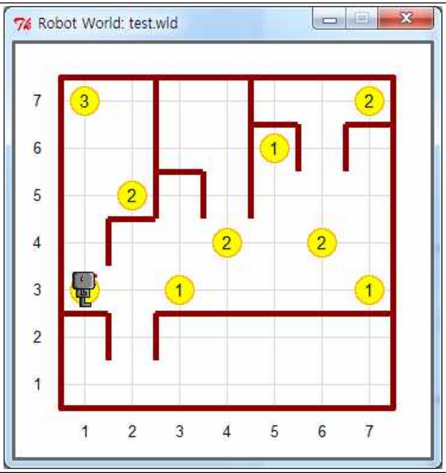
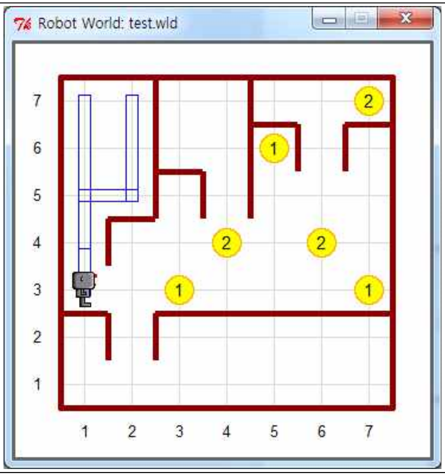
Source code

```
def one_line():
  hubo.turn_left()

  while hubo.front_is_clear() and (          #2-2          ):
    pick_and_move()
  if hubo.on_beeper():
    hubo.pick_beeper()

  if hubo.front_is_clear() and not (          #2-2          ):
    one_room()
  turn_around()
  if hubo.front_is_clear():
    hubo.move()

  while hubo.front_is_clear() and (          #2-2          ):
    hubo.move()

  if not (          #2-2          ):
    turn_around()
    hubo.move()
    turn_right()
  else:
    hubo.turn_left()
```

Description

This function is called when hubo is in the front-yard.
The purpose of this function is similar to 'one_line_in_a_room'. While picking up every litter in each vertical line, if there is a door then enter the room to clean up that room by invoking the 'one_room' function.

Example

Before	After
	

The complete program is as follows:

Program

```
from cs1robots import *

load_world('worlds/example_worlds.wld')

hubo = Robot(orientation='N',avenue=6, street=2) # hubo in-between a door
hubo.set_trace("blue")

hubo.move()
hubo.turn_left()
while hubo.front_is_clear():
  hubo.move()
turn_around() # Now, left-bottom coner. See the east
```

#2-4 (picking up litter in a house)

```
turn_around()
while not hubo.left_is_clear():
  hubo.move()
hubo.turn_left()
hubo.move()
while hubo.carries_beepers():
  hubo.drop_beeper()
```

3. (20 points) Answer each question according to the instruction.

3-1. (3 points) What is the result of the following program?

```
result = ""
for i in range(3) :
    for j in range(i+1) :
        result = str(j) + result
print result
```

3-2. (4 points) The following program prints out multiplication tables (from two times table to nine times table). You are asked to rewrite the following program using **while**-loop instead of **for**-loop and **if** statements. Your program must print out the same results as the existing program. **(If you use the keyword, 'if' or 'for' in your program, you will get 0 point.)**

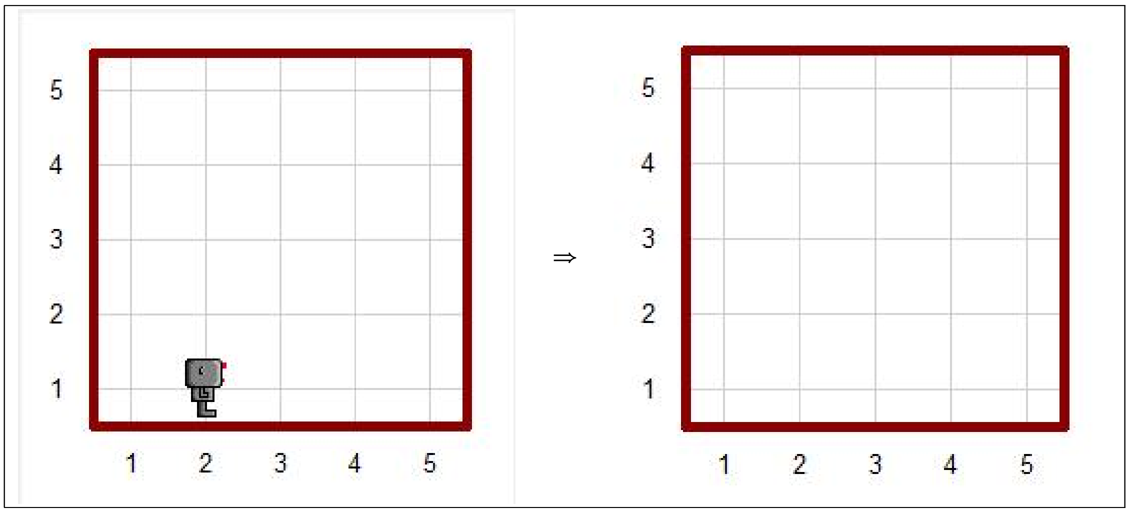
```
for i in range(10) :
    if i != 0 and i != 1:
        for j in range(10) :
            if j != 0 :
                print i, "*", j, "=", i*j
```

⇓

3-3. (4 points) 'Hubo' is facing east at the 1st street and the 2nd avenue in a 5×5 empty world. What is the trace of 'Hubo' after the following program is run? In other words, draw the lines or shape the following program generates.
(You don't need to indicate the direction and position of 'Hubo'.)
(You don't need to draw the trace caused by multiple turns in the same position.)

```
def turn_right() :
    for i in range(3) :
        Hubo.turn_left()
    return False

while Hubo.front_is_clear() :
    if not (Hubo.right_is_clear() and Hubo.left_is_clear()) or turn_right() :
        Hubo.move()
        Hubo.turn_left()
        Hubo.move()
    elif True or turn_right() and Hubo.move() :
        for i in range(2) :
            Hubo.move()
            Hubo.turn_left()
            Hubo.move()
```

<initial world> <your answer here>

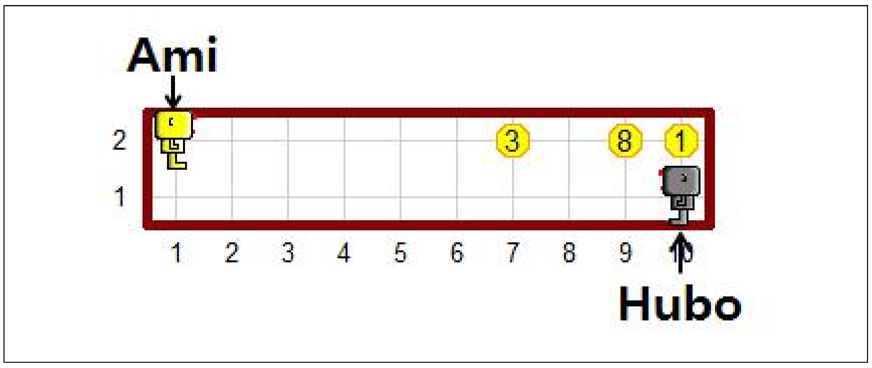


3-4. (4 points) In a 2×10 world, there are two robots and beepers. 'Ami' is facing east at the 2nd street and the 1st avenue. 'Hubo' is facing west at the 1st street and the 10th avenue, and carries an infinite number of beepers. Draw the final result of beepers after the following program is run. You should indicate the beepers as circles with a number (①, ②, ③, ...) at an exact location.

(You don't need to draw the robots.)

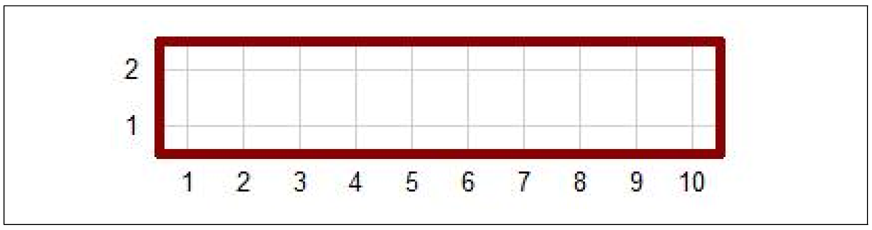
```
var1 = ""
var2 = 1
while Ami.front_is_clear() :
    Ami.move()
    temp = 0
    if Ami.on_beeper() :
        var2 += 1
        while Ami.on_beeper() :
            Ami.pick_beeper()
            temp += 1
        var1 += str(temp)
var1 = int(var1)
while not (var1 == 0) :
    for i in range((var1 % var2)) :
        Hubo.drop_beeper()
    var1 /= var2
    Hubo.move()
```

<initial world>

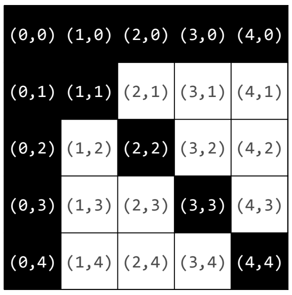


⇓

<your answer here>



3-5. (5 points) An image object named 'img' is shown below which consists of black and white pixels.



After the execution of the following programs, please show the resulting image objects ('img2' and 'img3') by shading the appropriate pixels black.

```
black = (0,0,0)
white = (255,255,255)
width, height = img.size()
img2 = create_picture(width,height)
for y in range(height):
    for x in range(width):
        if img.get(x,y) == black :
            img2.set(x,y,white)
        else :
            img2.set(x,y,black)
img2.show()
```

<answer for img2>

(0,0)	(1,0)	(2,0)	(3,0)	(4,0)
(0,1)	(1,1)	(2,1)	(3,1)	(4,1)
(0,2)	(1,2)	(2,2)	(3,2)	(4,2)
(0,3)	(1,3)	(2,3)	(3,3)	(4,3)
(0,4)	(1,4)	(2,4)	(3,4)	(4,4)

```
width, height = img.size()
img3 = create_picture(width,height)
for y in range(height):
    for x in range(width):
        img3.set(x, y,img.get(x,y))
for y in range(height/2):
    for x in range(width):
        temp = img3.get(x,height-y-1)
        img3.set(x,height-y-1,img3.get(x,y))
        img3.set(x,y,temp)
img3.show()
```

<answer for img3>

(0,0)	(1,0)	(2,0)	(3,0)	(4,0)
(0,1)	(1,1)	(2,1)	(3,1)	(4,1)
(0,2)	(1,2)	(2,2)	(3,2)	(4,2)
(0,3)	(1,3)	(2,3)	(3,3)	(4,3)
(0,4)	(1,4)	(2,4)	(3,4)	(4,4)

4. (20 points) Implement each function according to the instruction

4-1 (8 points) Please implement the following functions in Python.

$$f_1(x,y,z) = 3xf_2(y,z) + 2yf_2(x,z) + 5zf_2(x,y)$$
$$f_2(x,y) = xf_3(2y) + yf_3(3x)$$
$$f_3(x) = 3x + 2$$

※ 'function1' must call 'function2', and 'function2' must call 'function3'

```
def function1 (x,y,z) :




    return


def function2 (x,y) :




    return


def function3 (x) :




    return
```

4-2 (12 points)

Please implement a program which prints the properties of a number whether the number is positive or negative, or 0 or even or odd. You need to implement four functions to complete this program. The *'is_zero', 'is_positive'* and *'is_even'* functions return *'True'* when a number is 0 or positive or even respectively. The *'print_properties'* function prints out the properties of a parameter *x* by calling the *'is_zero', 'is_positive'* and *'is_even'* functions.

※ All codes of printing messages must be implemented in the *'print_properties'* function.

※ *'print_properties'* takes only an integer input. Hence, you must check the type of *x* in the 'print_properties' function.

※ In this program, 0 is not regarded as an even nor odd number. In addition, 0 is neither a positive nor negative number. Hence, if a number is 0 then you must not call the *'is_positive'* and *'is_even'* functions.

※ Please consider the output example below to implement the functions.

```
def is_zero(x):




def is_even(x):




def is_positive(x):

```

```
def print_properties(x) :
```

Output example:

```
print_properties('abc1')
>>> 'abc1' is not an integer number.
print_properties(1.1)
>>> 1.1 is not an integer number.
print_properties(0)
>>> 0 is 0, that's it.
print_properties(2)
>>> 2 is a positive and even number.
print_properties(3)
>>> 3 is a positive and odd number.
print_properties(-3)
>>> -3 is a negative and odd number.
```

5. (20 points) Answer each question according to the instruction.

5-1. (8 points) What is the result of the following program?

```
var1, var2, var3, var4 = 1, 2, 3, 5

def func1(var2, var4):
   var1 = 1
   var2 += 0
   var3 = 1
   var4 += 1
def func2(var1, var2):
   global var3, var4
   var1 += 5
   var2 += 5
   var3 /= 5
   var4 / 5
   return var1, var2
def func3(var):
   var = 15
   return var
def func4(a, b):
   return a + b

func1(var1, var3)
print var1, var2, var3, var4
var1, var2 = func2(var2, var4)
print var1, var2, var3, var4
var2 = func3(var1)
var3 = func4(func3(var3), 25)
var4 = func3(var4)
print var1, var2, var3, var4
```

Here are the descriptions of the objects and their methods used in 5-2 and 5-3.

Function	Description
Canvas(width, height, backgroundColor, title)	Create a new drawing canvas.
Square(length of side, centerPoint)	Construct a new Square instance.
Circle(radius, centerPoint)	Construct a new Circle instance.
setDepth(depth)	Set the depth of the object. * Objects with smaller depth appear in foreground.
setFillColor(color)	Set the interior color of the shape to the color.
getFillColor()	Return the color of the shape's interior.
rotate(angle of degrees)	Rotate the object around its center point.
sqrt(number)	Return the square root of the number.
scale(scaling factor)	Make an object smaller or larger with scaling factor
clone()	Return a duplicate of the drawable object.
move(dx, dy)	Move the object dx units along X-axis and dy units along Y-axis.
add(instance)	Add the Drawable object to the canvas.

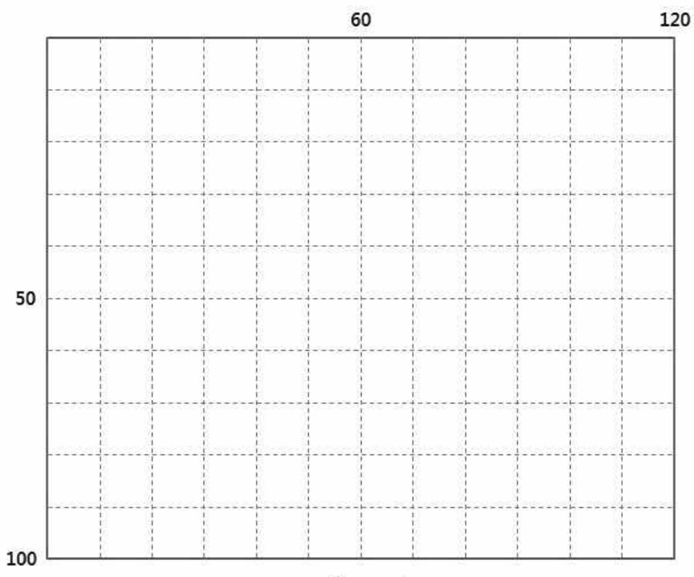
5-2. (4 points) What is the result of the following program?

```
from cs1graphics import *

book = Square(50)
book.setFillColor("black")
desk = book
desk.setFillColor("yellow")
photo = book
photo.setFillColor("green")

print desk.getFillColor()
```

____

5-3. (8 points) What is the result of following program? Draw it on the canvas.

```
from cs1graphics import *
import math

paper = Canvas( 120, 100, 'white', 'Canvas' )

sq1 = Square( 50, Point( 35,50 ) )
sq1.setDepth( 50 )
sq1.setFillColor( 'transparent' )

sq2 = sq1.clone()
sq2.rotate( 45 )
sq2.scale( math.sqrt(2) / 2 )

sq3 = sq2.clone()
sq3.setFillColor( 'white' )
sq3.move( 50, 0 )

circle = Circle( 25, Point( 85, 50 ) )
circle.setFillColor( 'black' )
circle.setDepth( 60 )

paper.add( sq1 )
paper.add( sq2 )
paper.add( sq3 )
paper.add( circle )
```



<Canvas>