

Question:	1	2	3	4	5	Total
Points:	20	20	20	20	20	100
Score:						

**KAIST CS101 Introduction to Programming
2017 Fall Midterm Exam**

Section: _____

Student ID: _____

Name: _____

- Please check if you received all 20 pages of the test material.
시작하기 전에 반드시 페이지가 총 20쪽인지 확인 하십시오.
- Fill in your section, student identification number and name. Otherwise you will lose 1 point for each missing piece of information.
분반, 학번과 이름을 정확히 기입하지 않을 경우, 각 실수 당 1점이 감점됩니다.
- **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 이상이 있을 경우, 왜 문제가 이상이 있다고 생각하는지에 대해 기술하시면 됩니다. 문제가 애매하다고 생각되는 경우 문제를 풀 때 본인이 생각하는 가정을 함께 작성하면 됩니다. 채점 시 가정 및 설명을 고려하겠습니다.
- **Write your answer in Python 3.** We will grade your answers based only on Python 3.
Python 3를 사용해 문제를 해결하세요. 채점은 Python 3 기준으로만 진행됩니다.
- You should write your answer in the 'Answer box'. We will grade your answers in the 'Answer box' only.
문제에 대한 답은 'Answer box'에 적으세요. 채점은 'Answer box'에 적힌 답만으로도만 진행됩니다.
- You should read and sign the Honor Code Agreement in the next page. The contents of the agreement is same as in lab 2.
다음 장에 있는 Honor Code Agreement를 읽고 서명해주세요. 내용은 lab 2의 것과 동일합니다.

Honor Code Agreement

Students taking KAIST CS101 are expected to respect personal honor and the rights of others, and they must possess personal integrity and honesty both as students and scientists/engineering professionals. The students will neither give nor receive any unauthorized aid in class work that is to be graded by the instructor. The following acts are regarded as violations of academic integrity and honesty.

- Referring from other students/publisher's solutions, assignments, and reports.
- Allowing another student to refer from one's own work
- Submitting another student's work as his or her own
- Unpermitted collaboration or aid on take-home examinations and class assignments
- Plagiarism: the use of another person's original work without giving reasonable and appropriate credit to or acknowledging the author or source

The Honor Code works to the benefit of students, professors, and administrators in the department, which is based on the mutual trust that all those bound by it will uphold its principles. However, this makes it the duty and responsibility of students and instructors to report any suspected violations of the Honor Code. The KAIST CS101 Honor Code requires that students take the following steps if a violation of the Honor Code is observed:

- Obtain the names of the people involved
- Inform the professor or head TA of the incident

If the instructor becomes aware of a violation of the Honor Code, it is his/her responsibility to contact the department chair to report this accusation. The department chair can form a committee consisting of the student's committee chair and the academic affairs committee to investigate the suspected violation. The department faculty committee will determine whether any violation has occurred and the appropriate penalty for the violation but the departmental regulation shall be based on the following.

Cheating and Plagiarism in Exams/Quizzes/Assignments

Any form of violations or academic dishonesty in exams/quizzes/assignments will be punished according to the following procedures.

- The offense of academic misconduct will result in F in the subject.
- KAIST student examination committee will officially punish those who deny their cheating activities

"I have read and agree to abide by all of the above rules and policies, and pledge that I will neither give nor receive any unauthorized aid on examinations or other class assignments that are used by the instructor as the basis for grading."

You agree that you understand all the details written above and you will not violate academic integrity and honesty by writing your name and signature here.

Name: _____ Signature: _____

1. (20 points) (No partial points) Answer each question according to the instruction.

1-1. (6 points) Fill in the blanks.

1-1-1. (2 points) The variables that are **accessible only within a function** are _____ variables.

[Answer box]

1-1-2. (2 points) A **predicate function** is a function that returns a _____ object.

[Answer box]

1-1-3. (2 points) In Python, String and Tuple objects are mutable because their _____ can be changed during the execution of a program.

[Answer box]

1-2. (2 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.

[Answer box]

1-3. (2 points) There are three *kinds of errors* that can be generated from a computer program. What **kinds of errors** you may receive when you do not put a proper type conversion in your program? Please write **two possible types of errors** that may be resulted in by running the program. (One point for each correct answer)

[Answer box]

- 1-4. (5 points) What is the **result** of the following expression in Python? (2 points) Please also specify **the order of calculation** in the square boxes ☐ below. (3 points)

1 + 2.0 * 3.0 / 4 % (6 // 5 ** (8 - 7)) * 9 + 10

☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐

[Answer box]

- 1-5. (5 points) Please write the complete **result** of running the following program. Please write “Error” if there is any error message that will be generated as the result of running the program.

[Program]

```
1 a = 2
2 def myFunc(a):
3     x = a * 5
4     return a, x
5 print(myFunc(3))
6 print(a)
7 print(x)
```

[Answer box]

2. (20 points) This question is related to your homework assignments. However, do **NOT** make assumptions of specific functions you used in homework assignments unless stated in the question. Read the instructions carefully and answer each question according to the instruction.

2-1. (8 points) Hubo needs to find where to build the road. The whole world of cs101 is said to have exactly two beepers in two distinct location. The goal is to find these two points and place an additional beeper at each location. When this task is successfully performed, there should be two points on the map that have exactly two beepers as illustrated in Figures 1 and 2 below. Your friend Jinyeong has already programmed a solution, but some parts have been erased.

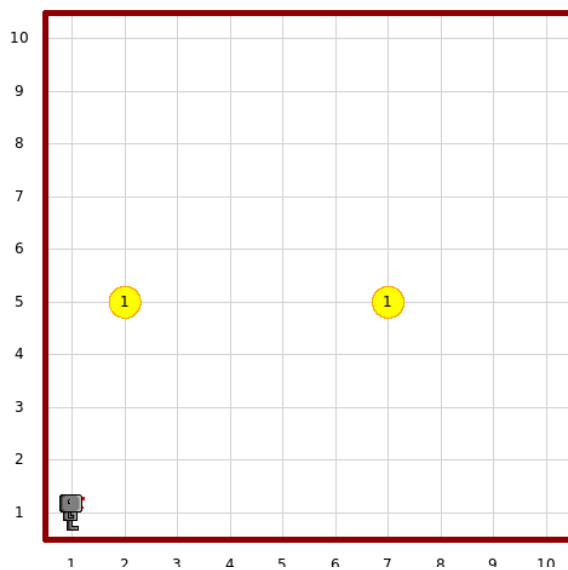


Figure 1: before

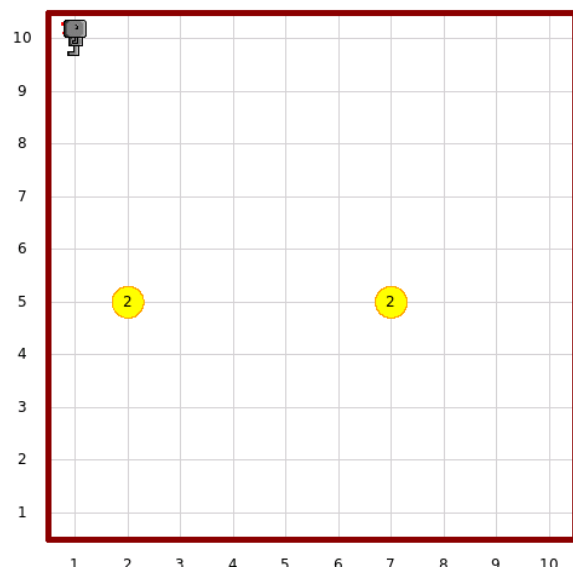


Figure 2: after

[Jinyeong's Solution]

```

1 from cs1robots import *
2 load_world("worlds/simple_world.wld")
3 hubo = Robot(beepers = 1000)
4
5 def turn_right():
6     (2-1-1)
7     hubo.turn_left()
8
9 def move_and_check():
10    hubo.move()
11    (2-1-2)
12    hubo.drop_beeper()
13
14 def go_straight():
15     for i in range(9):
16         move_and_check()
17
18 if hubo.on_beeper():
19     hubo.drop_beeper()
20
21 (2-1-3)

```

```
22 go_straight ()
23 hubo.turn_left ()
24 move_and_check ()
25 hubo.turn_left ()
26 go_straight ()
27 (2-1-4)
28 turn_right ()
29 move_and_check ()
30 turn_right ()
```

2-1-1. (2 points) Fill the proper condition(s) in the blank (2-1-1). You are not allowed to use a sentence 'while True:' with a 'break' keyword.

[Answer box for (2-1-1)]

2-1-2. (2 points) Fill the proper condition(s) in the blank (2-1-2). You are not allowed to use a sentence 'while True:' with a 'break' keyword.

[Answer box for (2-1-2)]

2-1-3. (2 points) Fill the proper condition(s) in the blank (2-1-3). You are not allowed to use a sentence 'while True:' with a 'break' keyword.

[Answer box for (2-1-3)]

2-1-4. (2 points) Fill the proper condition(s) in the blank (2-1-4). You are not allowed to use a sentence 'while True:' with a 'break' keyword.

[Answer box for (2-1-4)]

- 2-2. (12 points) The following program draws graphs for polynomial functions. It receives the coefficients for the polynomial from the user. Answer the questions.

[Program]

```

1 from cslmedia import *
2 import math
3 import elice_utils
4
5 coeff_a = 0
6 coeff_b = 0
7 coeff_c = 0
8 coeff_d = 0
9
10 def draw_axes(size, ranges):
11     width, height = size
12     xmax, ymax = ranges
13
14     white = (255, 255, 255)
15
16     if width <= 0 or height <= 0:
17         return None
18
19     if :
20         return None
21
22     if :
23         return None
24
25     if ((width-1)/2)%xmax != 0 or ((height-1)/2)%ymax != 0:
26         return None
27
28     img = create_picture(width, height)
29     for x in range(width):
30         for y in range(height):
31             img.set(x, y, white)
32
33     black = (0, 0, 0)
34     half_tick = 4
35
36     xorigin = int((width-1)/2)
37     yorigin = int((height-1)/2)
38
39     xinterval = int((width-1)/2/xmax)
40     yinterval = int((height-1)/2/ymax)
41
42     for x in range(width):
43         img.set(x, yorigin, black)
44         if x % xinterval == 0:
45             for i in range(half_tick):
46                 img.set(x, yorigin+i+1, black)
47                 img.set(x, yorigin-i-1, black)
48
49
50     for y in range(height):
51         img.set(xorigin, y, black)
52         if y % yinterval == 0:
53             for i in range(half_tick):

```

```

54         img.set(xorigin+i+1, y, black)
55         img.set(xorigin-i-1, y, black)
56     return img
57
58 def polynomial(x):
59     (2-2-2)
60
61 def draw_function(img, func):
62     if img == None:
63         return None
64     width, height = img.size()
65     red = (255, 0, 0)
66     black = (0, 0, 0)
67
68     xorigin = int((width-1)/2)
69     yorigin = int((height-1)/2)
70
71     xmax = 0
72     for x in range(xorigin+1, width):
73         if black == img.get(x, yorigin+1):
74             xmax += 1
75
76     ymax = 0
77     for y in range(yorigin+1, height):
78         if black == img.get(xorigin+1, y):
79             ymax += 1
80     if xmax == 0 or ymax == 0:
81         return None
82
83     xorigin = (2-2-3-1)
84     yorigin = (2-2-3-2)
85
86     xinterval = int((width-1)/2/xmax)
87     yinterval = int((height-1)/2/ymax)
88
89     for xpixel in range(width):
90         x = (xpixel - xorigin)/xinterval
91         y = func(x)
92         ypixel = (2-2-4)
93         if ypixel < height and ypixel >= 0:
94             img.set(xpixel, ypixel, red)
95
96     return img
97
98 def main():
99
100     global coeff_a
101     global coeff_b
102     global coeff_c
103     global coeff_d
104
105     width = 0
106     height = 0
107
108     width = int(input("Insert the width of the image: "))
109     height = int(input("Insert the height of the image: "))
110

```



```

111     xmax = int(input("Insert the maximum value of the x-axis: "))
112     ymax = int(input("Insert the maximum value of the y-axis: "))
113
114     size = (width, height)
115     ranges = (xmax, ymax)
116
117     img = draw_axes(size, ranges)
118
119     print("Drawing polynomial functions")
120     print("Polynomial coefficients:")
121
122     coeff_a = float(input("Insert a: "))
123     coeff_b = float(input("Insert b: "))
124     coeff_c = float(input("Insert c: "))
125     coeff_d = float(input("Insert d: "))
126     img = draw_function(img, polynomial)
127
128     if not img == None:
129         img.save_as('result.png')
130         elice_utils.send_image('result.png')
131
132 if __name__ == '__main__':
133     main()

```

Some *math* library functions you may want use in your answers are listed and described in the table below.

<code>math.ceil(x)</code>	Return the ceiling of x, the smallest integer greater than or equal to x.
<code>math.fabs(x)</code>	Return the absolute value of x.
<code>math.factorial(x)</code>	Return x factorial.
<code>math.floor(x)</code>	Return the floor of x, the largest integer less than or equal to x.
<code>math.isfinite(x)</code>	Return True if x is neither an infinity nor a NaN, and False otherwise. (Note that 0.0 is considered finite.)
<code>math.isinf(x)</code>	Return True if x is a positive or negative infinity, and False otherwise.
<code>math.isnan(x)</code>	Return $x * (2^{**i})$. This is essentially the inverse of function <code>frexp()</code> .
<code>math.ldexp(x, i)</code>	Return the floor of x, the largest integer less than or equal to x.
<code>math.modf(x)</code>	Return the fractional and integer parts of x. Both results carry the sign of x and are floats.

- 2-2-1. (4 points) Complete the necessary sanity checks (e.g., checking whether user inputs are valid) that go into (2-2-1-1) and (2-2-1-2).

[Answer box for (2-2-1-1)]

[Answer box for (2-2-1-2)]

- 2-2-2. (4 points) Implement the polynomial function at (2-2-2). Polynomial function receives a float value x , and returns the result of the calculation $ax^3 + bx^2 + cx + d$.

[Answer box for (2-2-2)]

2-2-3. (2 points) Complete (2-2-3-1) and (2-2-3-2) to find the appropriate origins of both x axis and y axis.

[Answer box for (2-2-3-1)]

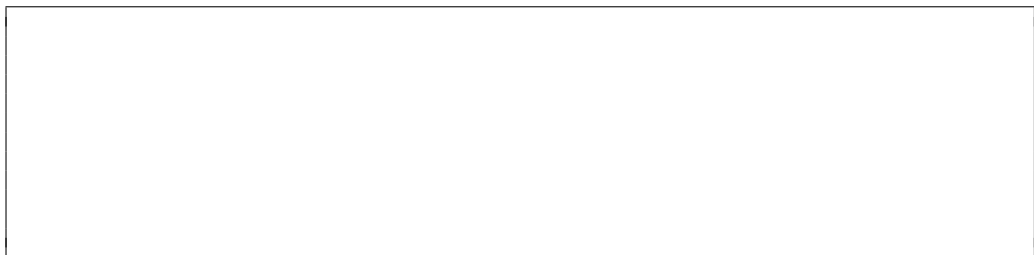
A large, empty rectangular box with a thin black border, intended for the user to provide an answer for question (2-2-3-1).

[Answer box for (2-2-3-2)]

A large, empty rectangular box with a thin black border, intended for the user to provide an answer for question (2-2-3-2).

2-2-4. (2 points) Complete (2-2-4) to calculate the correct *ypixel* value.

[Answer box for (2-2-4)]

A large, empty rectangular box with a thin black border, intended for the user to provide an answer for question (2-2-4).

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

3-1. (2 points) What is the result of the following program?

[Program]

```
1 from cslrobots import *
2
3 hubo = "Robot()"
4
5 if type(hubo) == type("True") :
6     print ("Perfect!")
7 if type(hubo) == "str" :
8     print ("Awesome!")
9 else:
10     print ("Amazing!")
11
```

[Answer box for 3-1]

--

3-2. (8 points) What is the outcome of the following program?

[Program]

```
1 for i in range(4):
2     result = str(i)
3     for j in range(i+1):
4         if j % (i+2) == 2:
5             result += "/"*j + result
6         else :
7             result += "+" + str(j)
8     print(result)
9
```

[Answer box for 3-2]

[illegible]

- 3-3. (5 points) The following program loads and modifies an image file (“test.bmp”) as shown in the [Canvas] table. Each box in the image represents the pixel, and the text in the box shows the location of the pixel. Draw the modified image on the right-side canvas (*modified image file*).

[Program]

```

1 from cslmedia import *
2
3 black = (0, 0, 0)
4 white = (255, 255, 255)
5
6 img = load_picture("test.bmp")
7 w, h = img.size()
8
9 for y in range(h):
10     for x in range(w):
11         if img.get(x, y) == black:
12             img.set(y, x, black)
13             img.set(x, y, white)
14
15 img.show()
16

```

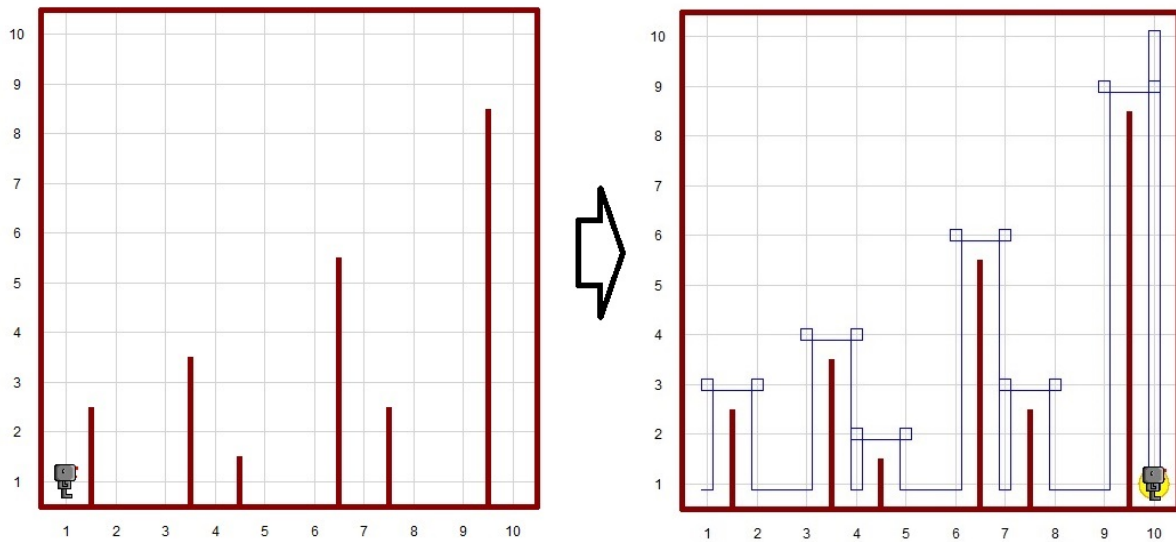
[Canvas and Answer box]

Original image file (<i>test.bmp</i>)					Modified image file				
(0, 0)	(1, 0)	(2, 0)	(3, 0)	(4, 0)	(0, 0)	(1, 0)	(2, 0)	(3, 0)	(4, 0)
(0, 1)	(1, 1)	(2, 1)	(3, 1)	(4, 1)	(0, 1)	(1, 1)	(2, 1)	(3, 1)	(4, 1)
(0, 2)	(1, 2)	(2, 2)	(3, 2)	(4, 2)	(0, 2)	(1, 2)	(2, 2)	(3, 2)	(4, 2)
(0, 3)	(1, 3)	(2, 3)	(3, 3)	(4, 3)	(0, 3)	(1, 3)	(2, 3)	(3, 3)	(4, 3)
(0, 4)	(1, 4)	(2, 4)	(3, 4)	(4, 4)	(0, 4)	(1, 4)	(2, 4)	(3, 4)	(4, 4)

- 3-4. (5 points) We want to implement a program that makes the robot object (*hubo*) jump hurdles of any sizes. However, there is no beeper in the given world file (*given_hurdle.wld*), so you have to terminate the program when *hubo* arrives at the rightmost wall of the world. Suppose that *hubo* checks if he/she is on the beeper to determine to terminate the program. So, *hubo* has to drop a beeper for the termination when he/she reaches the bottom right corner in the world. Fill in the blanks to make the following program work with the given world. Note that you are not allowed to modify other parts of the program, and your program should also work correctly if you load other worlds having hurdles with no beepers.

[Program]

```
1 from cslrobots import *
2
3 load_world("given_hurdle.wld")
4 hubo = Robot(beepers = 1)
5 hubo.set_trace("blue")
6
7 def turn_right():
8     for i in range(3):
9         hubo.turn_left()
10
11 def jump_one_hurdle():
12     hubo.turn_left()
13     while :
14         hubo.move()
15
16     if :
17         turn_right()
18         hubo.move()
19         turn_right()
20         while hubo.front_is_clear():
21             hubo.move()
22         hubo.turn_left()
23     else:
24         
25         hubo.drop_beeper()
26
27 while not hubo.on_beeper():
28     if hubo.front_is_clear():
29         hubo.move()
30     else:
31         jump_one_hurdle()
```



[Answer box for 3-4-1]

[Answer box for 3-4-2]

[Answer box for 3-4-3]

4. (20 points) Answer each question according to the instruction.

4-1. (14 points) Write the output of following program.

[Program]

```

1 a, b = 1, 2
2 def func1(a, b):
3     a, b = 2, b + 1
4     return b
5 def func2(c, a):
6     global b
7     a, b, c = a + 1, b + 2, c - 1
8     return b + c - a
9 def func3(c):
10    global a
11    a = a - 1
12    return func2(func1(b, a), c)
13 def func4(a):
14     if a > 5:
15         return a
16     return func4(a+1) + 1
17 print(a,b)
18 print(func1(b,a))
19 print(a,b)
20 print(func2(a,b))
21 print(a,b)
22 print(func3(b))
23 print(a,b)
24 print(func4(a))

```

[Answer box]

Print statements	Output
(Example) print(a,b)	1 2
print(func1(b,a))	
print(a,b)	
print(func2(a,b))	
print(a,b)	
print(func3(b))	
print(a,b)	
print(func4(a))	

- 4-2. (6 points) You are asked to implement a program which prints all prime numbers less than parameter **n**. The program also prints the number of prime numbers less than **n**. You must use **is_prime** function in **print_prime** function to determine a number is prime or not, while **is_prime** should return a boolean value that tells whether **x** is prime. **print** function **should not** be used in **is_prime** function.

[Program]

```
1 def is_prime(x):  
2     (4-2-1)  
3 def print_prime(n):  
4     (4-2-2)
```

[Demo]

```
1 >> print_prime(20)  
2 2  
3 3  
4 5  
5 7  
6 11  
7 13  
8 17  
9 19  
10 The number of prime numbers less than 20 is 8
```

[Answer box for 4-2-1]**[Answer box for 4-2-2]**

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

5-1. (4 points) The code below is faulty. Explain what the problem is and explain in detail the reason (s) of the problem (specifying the problem line number).

[Program]

```
1 msg = 'Hello, cs101'
2 def test():
3     print(msg)
4     msg = 'Hi, cs101'
5     print(msg)
6 print(msg)
7 test()
8 print(msg)
```

[Answer box]

5-2. (8 points) Below is an example of drawing using the cs1graphics module. After you understand the code, answer each question.

[Program]

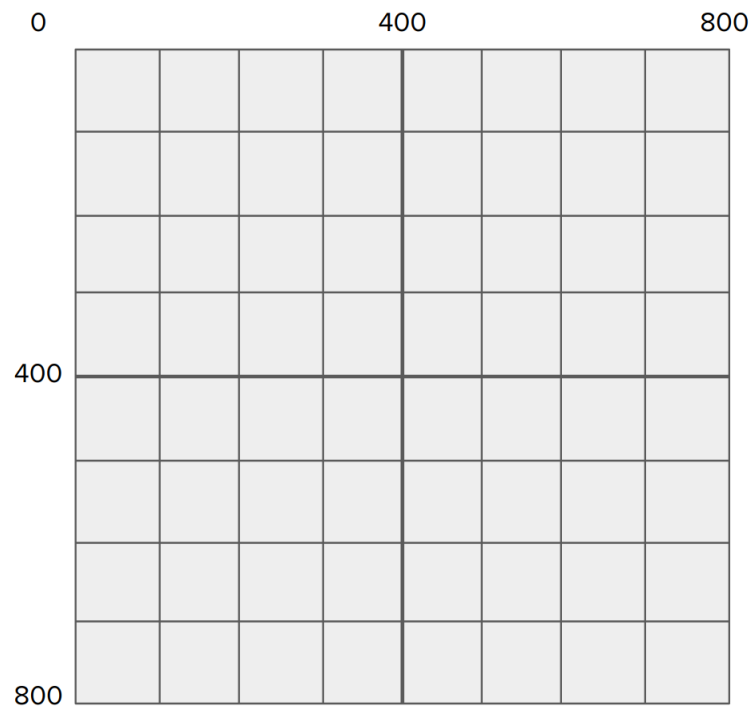
```
1 from cs1graphics import *
2
3 canvas = Canvas(800,800)
4
5 def coloring(drawing_object, fill_color, border_color):
6     drawing_object.setFillColor(fill_color)
7     drawing_object.setBorderColor(border_color)
8
9 sq = Square(200)
10 sq.setBorderColor("red")
11 print(sq.getBorderColor())
12 coloring(sq, "blue", "gray")
13 sq.setBorderWidth(50)
14 sq.moveTo(400,400)
15 canvas.add(sq)
16 print(sq.getBorderColor())
17
18 for i in range(90):
19     sq.rotate(1)
```

Write down the result of the code

[Answer box: 5-2-1]

Draw a picture that is generated as a result of executing the code below. (If you do not have a colored pencil to paint the color, describe the surface and color in writing.)

[Answer box: 5-2-2]



- 5-3. (8 points) Below is an example of animating "Drawable objects". After all the animations are done, draw the last picture. If you think your picture is not descriptive enough, write a description of the picture.

[Program]

```

1 from cs1graphics import*
2 import math
3
4 canvas = Canvas(800,400)
5
6 def create_sun(radius, color):
7     sun = Circle(radius)
8     sun.setFillColor(color)
9     sun.setBorderColor(color)
10    sun.moveTo(100, 100)
11    return sun
12
13 sun = create_sun(100, "red")
14
15 def animate_sunrise(sun):
16     w = canvas.getWidth()
17     h = canvas.getHeight()
18
19     for angle in range(181):
20         sun = create_sun(50, "red")
21         r = sun.getRadius()
22         x0 = w / 2.0
23         y0 = h + r
24         xradius = w / 2.0 - r
25         yradius = h
26         rad = (angle/180.0) * math.pi
27         x = x0 - xradius * math.cos(rad)
28         y = y0 - yradius * math.sin(rad)
29         sun.moveTo(x, y)
30         canvas.add(sun)
31 animate_sunrise(sun)

```

[Answer box]

