Exam 1 – Paper and Pencil part (Winter, 2018-19)

Name: ____ Section: _______

Rules and Expectations

At the beginning of this exam, you will receive the **Expectations about Academic Integrity** for this exam -- it is the same as what you were given to read previously. Re-read that document as needed. ***Sign it and turn it in when you finish this exam (both parts).***

Two parts (this is Part 1, Paper-and-Pencil)

For this part, the ONLY external resource you may use is a single 8½ by 11-inch sheet of paper, with whatever you want on it, typed or handwritten or a combination of the two. **You may use only ONE side of the sheet.** You must have prepared the sheet *before* beginning this exam. You may also use a calculator if you like (but only for calculating).

Problem	Points Possible	Points Earned	Comments
1	10		
2	10		
3	10		
4	10		
5	2		
6	2		
7	2		
8	4		
Total (of 100 on the exam)	50		

Communication

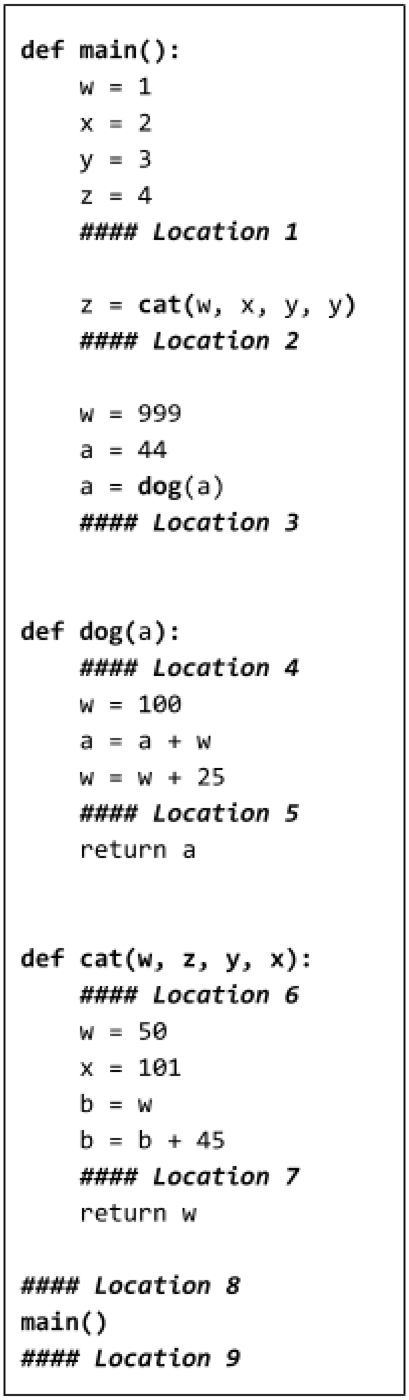
For both parts of the exam, ***you must not communicate with anyone*** except your instructors and their assistants, if any. In particular:

- You must not talk with anyone else or exchange information with them during this exam.
- After this exam, you must not talk about the exam with anyone who has not yet taken it.

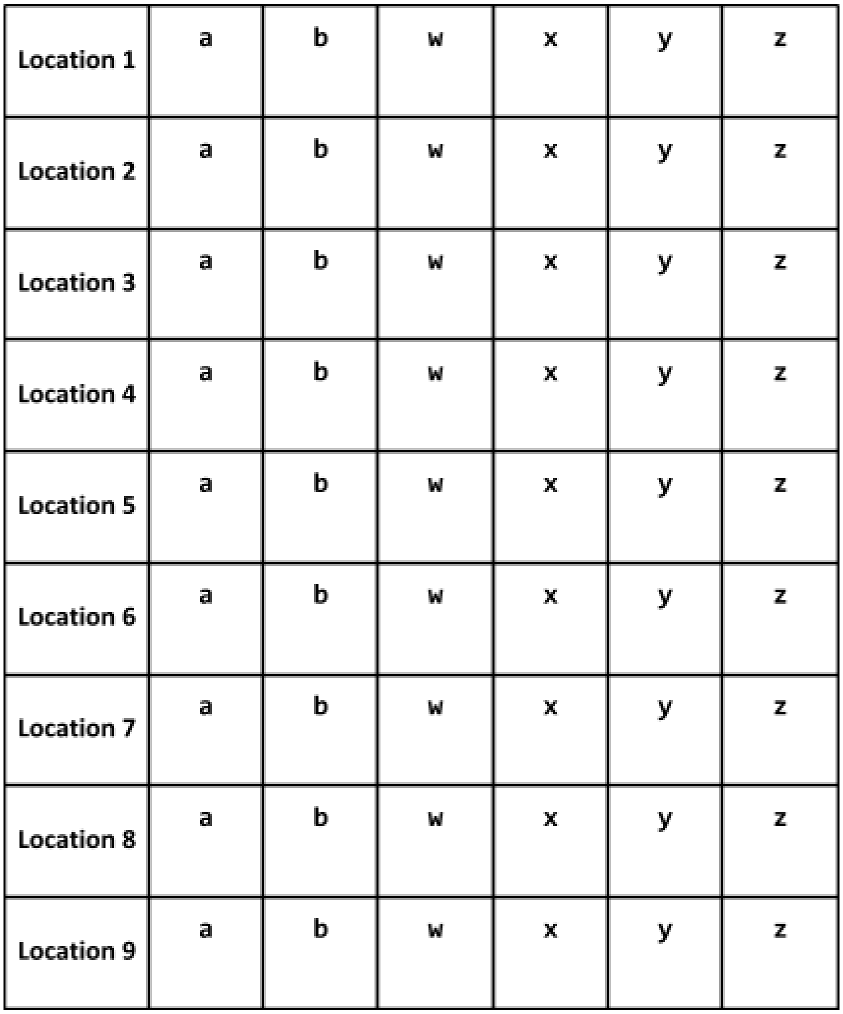
Do NOT use email, *chat* or the like during this exam. **Close any such applications now.**

1. (10 points) Consider the code below. It is a contrived example with poor style but will run without errors. In this problem, you will trace the execution of the code. As each location is encountered during the run:

    1. ***CIRCLE* each variable** that is ***defined*** at that location.
    2. ***WRITE*** t**he *VALUE*** of each variable that you ***circled*** directly ***BELOW*** the circle.



```
def main():
    w = 1
    x = 2
    y = 3
    z = 4
    #### Location 1

    z = cat(w, x, y, y)
    #### Location 2

    w = 999
    a = 44
    a = dog(a)
    #### Location 3


def dog(a):
    #### Location 4
    w = 100
    a = a + w
    w = w + 25
    #### Location 5
    return a


def cat(w, z, y, x):
    #### Location 6
    w = 50
    x = 101
    b = w
    b = b + 45
    #### Location 7
    return w

#### Location 8
main()
#### Location 9
```



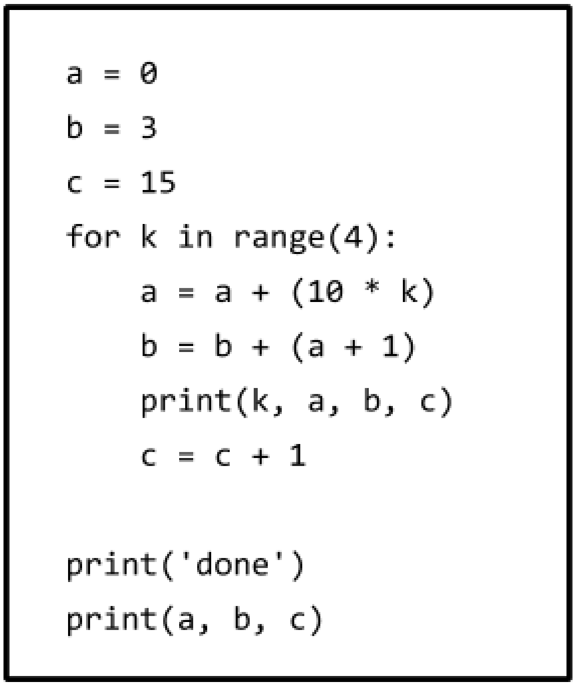
Location 1	a	b	w	x	y	z
Location 2	a	b	w	x	y	z
Location 3	a	b	w	x	y	z
Location 4	a	b	w	x	y	z
Location 5	a	b	w	x	y	z
Location 6	a	b	w	x	y	z
Location 7	a	b	w	x	y	z
Location 8	a	b	w	x	y	z
Location 9	a	b	w	x	y	z

Make notations in the code as desired to show your work.

Ask for help if you do not understand the instructions for this problem.

2. (10 points) Consider the code below. It is a contrived example with poor style, but it will run without errors.

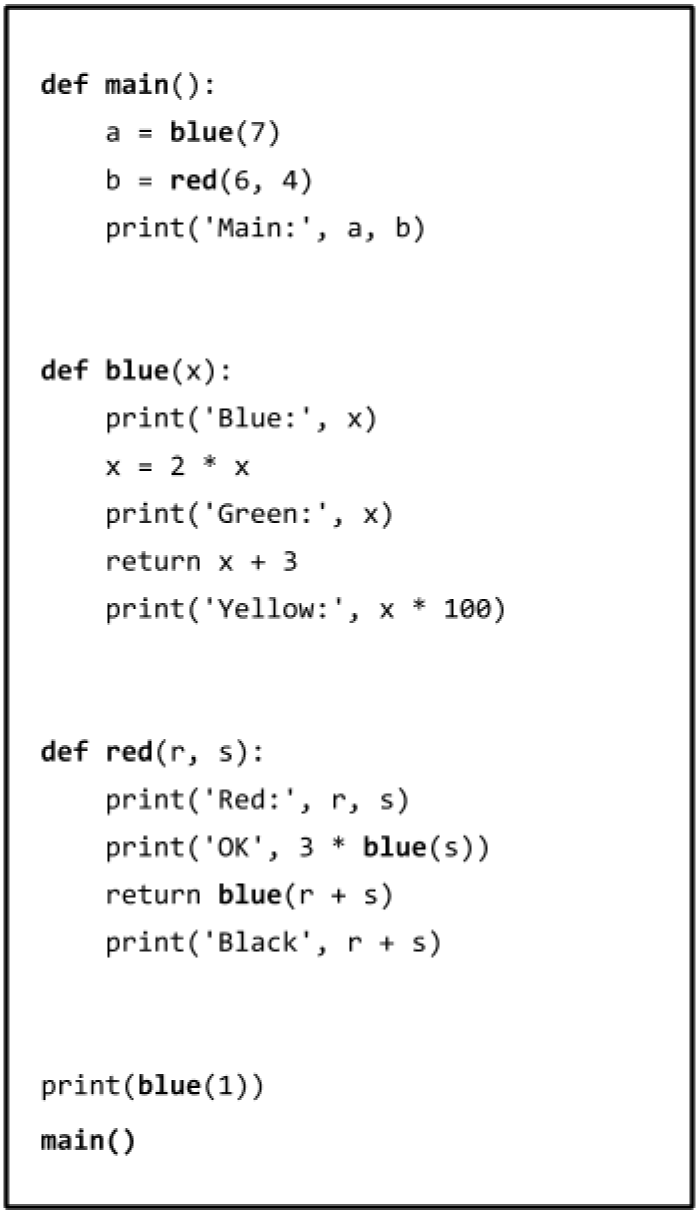
What does it print when it runs? Write your answer in the box to the right. Use the empty space to keep track of variables as you work.



```
a = 0
b = 3
c = 15
for k in range(4):
    a = a + (10 * k)
    b = b + (a + 1)
    print(k, a, b, c)
    c = c + 1

print('done')
print(a, b, c)
```

Output:

3. (10 points) Consider the code below. It is a contrived example with poor style, but it will run without errors. What does it print when it runs? Write your answer in the box to the right. Make notations in the code as desired to show your work.



```
def main():
    a = blue(7)
    b = red(6, 4)
    print('Main:', a, b)


def blue(x):
    print('Blue:', x)
    x = 2 * x
    print('Green:', x)
    return x + 3
    print('Yellow:', x * 100)


def red(r, s):
    print('Red:', r, s)
    print('OK', 3 * blue(s))
    return blue(r + s)
    print('Black', r + s)


print(blue(1))
main()
```

Output:

4. (10 points) Consider a function whose name is ***sum_powers*** that takes a positive integers ***m*** and a number ***r*** as its arguments. The function computes and returns:

$$1^r + 2^r + 3^r + \ldots + m^r$$

For example, ***sum_powers*(7, 2)** returns

$$1^2 + 2^2 + 3^2 + 4^2 + 5^2 + 6^2 + 7^2$$

(which is **1 + 4 + 9 + 16 + 25 + 36 + 49,** which is **140**)

while ***sum_powers*(3, 0.5)** returns

$$1^{0.5} + 2^{0.5} + 3^{0.5}$$

(which is approximately **1 + 1.414 + 1.732,** that is, approximately **4.146**)

Ask for a further explanation if you do not understand what the function ***sum_powers*** is to return.

Write (in the box below) a complete implementation, including the header (***def***) line, of the above ***sum_powers*** function.

5. (2 points) Assume that there is a class named **`Note`** whose constructor:

- requires a single argument that is a frequency (of the musical note to play), and
- stores that frequency in an instance variable named **`freq`**.

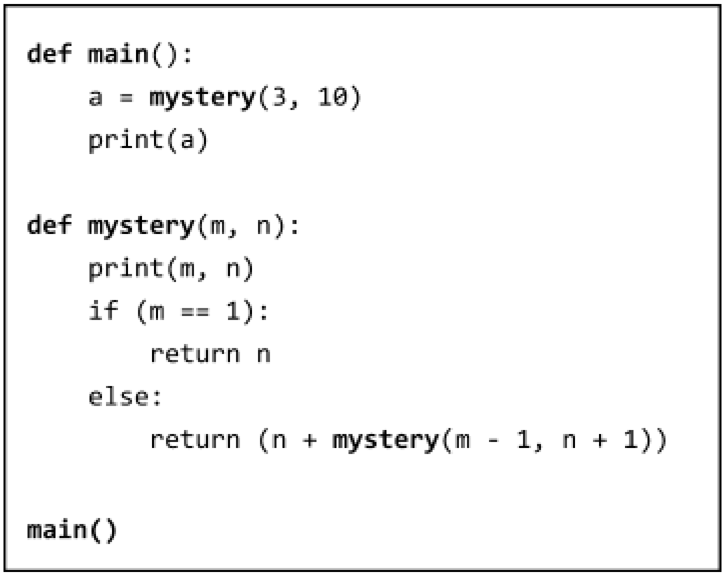
Assume further that **`Note`** objects have a **`play_note`** method that plays the note for N seconds, where N is the argument to the **`play_note`** method.

Write code that would construct a **`Note`** object whose frequency is 440, naming the constructed object ***`concert_A`***.

6. (2 points) Continuing the previous problem, write code that would make ***`concert_A`*** play for 1.88 seconds.

7. (2 points) Continuing the previous problems, suppose that you have another **`Note`** object named ***`another_note`***. Write code that would set the variable ***`f`*** to a number that is **twice** the frequency of **`another_note`**.

8. (4 points) Consider the code below. It will run without errors. What does it print when it runs? Write your answer in the box to the right of the code. Show your work by making notations in the code or by using the empty space below or on another sheet of paper, as desired.



```
def main():
    a = mystery(3, 10)
    print(a)

def mystery(m, n):
    print(m, n)
    if (m == 1):
        return n
    else:
        return (n + mystery(m - 1, n + 1))

main()
```

Output: