Name: ______________________________ Recitation: ____ Andrew Id: ________________

15-112 Spring 2018 Quiz 10

Up to 25 minutes. No calculators, notes, books, or computers. Show your work!

1. (20 points) **Code Tracing**: Indicate what the following program prints. Place your answers (and nothing else) in the box to the right of the code.

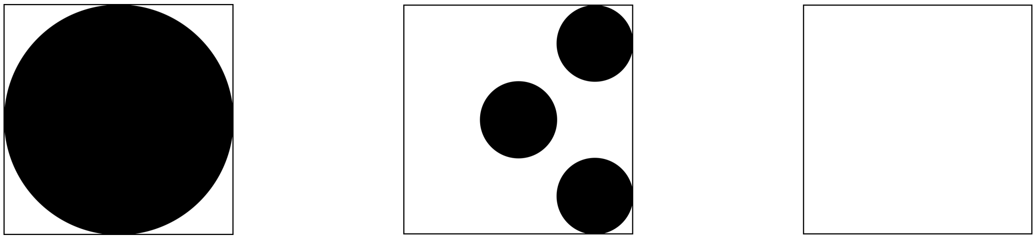
```
def ct(s, depth=0):
    print(depth, s)
    if len(s) > 1 and s[0] != "b":
        ct(s[1:], depth+1)
        ct(s[:-1], depth+1)
    print(depth, "done")

ct("abc")
```

2. (30 points) **Code Writing**: Write the function fdp(path) (short for findDuplicateFilenames) that returns a list of all file names in the given path (or any of its folders, or their folders, etc.) that appear more than once. Here we are referring to a file's immediate name, not the full path name. For example, assume we have a folder called "main". This folder contains two files ("a.txt" and "b.txt") and two folders ("sub" and "also"). "main/sub" contains two files ("a.txt" and "c.txt"), and "main/also" contains two files ("a.txt" and "c.txt"). The function should return ["a.txt", "c.txt"], since "a.txt" appears three times and "c.txt" appears twice. The order of the returned list does not matter. You may not directly use listFiles(path) in your answer.

3. (10 points) **Short Answer**: Assume that we call mergesort on the list [3,4,1,7,2,6,8,5]. In the recursive implementation, what are the two lists that are passed to merge in the call at depth 0?

4. (10 points) **Short Answer**: Say you want to solve a maze using backtracking. You can use the maze representation we went over in class or make up your own representation. Assuming you use the backtracking template, what would the possible moves for a step be, and how would you check if they were valid?

5. (10 points) **Short Answer**: The fractal shown below is an X-Fractal at depth 0 and depth 1. Draw an X-Fractal at depth 2 in the remaining box.



6. (20 points) **Reasoning Over Code**: Find an argument for roc that makes it return True. Place your answer (and nothing else) in the box under the code.

```
def rocHelper(lst):
    if len(lst) <= 1:
        return [ lst ]
    else:
        sublsts = rocHelper(lst[1:-1])
        result = []
        for i in range(len(sublsts)):
            result += [[lst[0]] + sublsts[i]]
            result += [sublsts[i] + [lst[-1]]]
        return result
def roc(lst):
    result = rocHelper(lst)
    return len(lst) == 4 and result == [ [3, 2], [2, 4], [3, 1], [1, 4] ]
```

YOU MAY CONTINUE WRITING CODE ON THIS PAGE.

USE THIS PAGE FOR SCRAP WORK. WORK ON THIS PAGE WILL NOT BE GRADED.