COSC 101, Exam #3 Retake
April 2024

Name: _____

Please write your name above. Do not start the exam until instructed to do so.

You have 75 minutes to complete this exam.

There are 3 questions and a total of 54 points available for this exam. Don't spend too much time on any one question.

Since indentation is important in Python, please be sure that your use of indentation is obvious for any code you write.

If you want partial credit, show as much of your work and thought process as possible.

If you run out of space for answering a question, you can continue your answer on one of the blank pages at the end of the exam. If you do so, be sure to indicate this in two places: (1) below the question, indicate which blank page contains your answer, and (2) on the blank page, indicate which question you are answering.

Question	Points	Score
1	16	
2	18	
3	20	
Total:	54	

1. (a) (5 points) Consider this code snippet:

```
def quiz(n: int, m: int) -> None:
    for i in range(m):
        j = i
        for j in range(n):
            if i == j:
                print('y')
            print('x', end='')
        print(i)

quiz(4,2)
```

What is the output produced by this code snippet? As you trace, draw the loop tracing table.

Solution:

```
y
xxxx0
xy
xxx1
```

(b) (6 points) Consider this program:

```
def mystery(n: int) -> None:
    while n > 0:
        for i in range(2):
            n = n + 1
        print(n)
        for i in range(3):
            n = n - 1
        print(n)

mystery(4)
```

What is the output produced by this program? As you trace, draw the loop tracing table.

Solution:

```
6
3
5
2
4
1
3
0
```

(c) (5 points) Consider this program:

```
def mysteryNL(management: dict) -> None:
    i = 0
    for s in management:
        print(s[i])
        avar = ''
        for value in management.values():
            avar += value[i]
        i += 1
        print(avar)

mysteryNL({
'Alice': ['Manish', 'Chen'],
'Ayumu': ['Greny', 'Amelie', 'Jaeyun']
})
```

What is the output produced by this program? As you trace, draw the loop tracing table.

Solution:

```
A
ManishGreny
y
ChenAmelie
```

2. (a) (7 points) Consider this program:

```
def main() -> None:
    lst = [1,2]
    alst = [1,2]
    lst[0] = 0
    print(alst)
    print(lst)
    lst = alst
    print(alst)
    print(lst)
    lst[0] = 2
    alst[1] = 3
    print(alst)
    print(lst)

main()
```

What is the output produced by this program? As you trace, draw the function frames diagram.

Solution:

```
[1, 2]
[0, 2]
[1, 2]
[1, 2]
[2, 3]
[2, 3]
```

(b) (11 points) Consider this program:

```
def mystA(x: int, a: list) -> int:
    print(x)
    print(a)
    a[x] = 2
    x = 3
    print(x)
    print(a)
    return x
def mystB(n: int, alst: list) -> int:
    alst = [1,2,3]
    n = 4
    print(n)
    print(alst)
    return alst
def main() -> None:
    x = 1
    lst = [1,1]
    x = mystA(x, lst)
    print(x)
    print(lst)
    lst = mystB(x, lst)
    print(x)
    print(lst)
main()
```

What is the output produced by this program? As you trace, draw the function frames diagram.

Solution:

```
1
[1, 1]
3
[1, 2]
3
[1, 2]
4
[1, 2, 3]
3
[1, 2, 3]
```

3. This question builds on the third question on Exam 3 by requiring you to write out the information in the dictionary to a file and implement additional functions.

   Recall that for question 3 on Exam 3 you were tasked to read from the `management.txt` file with the following format:

   ```
   Manager Managee
   Alice Chen
   Ayumu Greny
   Ayumu Amelie
   Ayumu Jaeyun
   ```

   and store the information in a dictionary as follows:

   ```
   {'Alice': ['Chen'],
   'Ayumu': ['Greny', 'Amelie', 'Jaeyun']
   }
   ```

   Next, we want to implement a function called `write_data` to output the information in the dictionary to a file as follows:

   - When a manager does not have any managees, say that they do not manage anyone.
   - When a manager has only one managee, say that they only manage that one managee.
   - Otherwise, say that the manager manages all the managess by indicating their names.

   For example, for this dictionary:

   ```
   {'Alice': ['Chen'],
   'Ayumu': ['Greny', 'Amelie', 'Jaeyun'],
   'John': []
   }
   ```

   The content of the file must be:

   ```
   Alice manages only Chen.
   Ayumu manages Greny, Amelie, and Jaeyun.
   John does not manage anyone.
   ```

(a) (10 points) Select only 10 lines of code from above, and **only one line from each group**. You may write only the line identifiers (*e.g.,* E2) below, or write out the code. Your selections should *only* go on numbered lines below (see next page).

```
A1      if len(management[manager]) == 0:
A2      if len(management) == 0:

B1     elif len(management[manager]) == 1:
B2     else:

C1  with open(filename, 'w') as output_obj:
C2  output_obj = open(filename, 'a')

D1          for index in range(len(management[manager])-1):
D2          for key management[manager]:

E1          output_obj.write(manager+' manages only '+
                    management[key] + '.\n')
E2          output_obj.write(manager+' manages only '+
                    management[manager][0] + '.\n')

F1         output_obj.write(manager+' does not manage anyone\n')
F2         output_obj.write('manager does not manage anyone')

G1          output_obj.write(manager+' manages '+managees+'.\n')
G2          output_obj.write(manager+' manages '+managees.values())

H1          output_obj.close()
H2          managees = ''

I1          managees+='and '+
                 management[manager[len(management[manager])-1]]
I2          managees+='and '+
                 management[manager][len(management[manager])-1]

J1            managees += management[manager][index] + ', '
J2            managees += management[key] + ', '
```

```
def write_data(management:dict, filename:str) -> None:
```

1

```
    for manager in management:
```

2

3

4

5

```
        else:
```

6

7

8

9

10

Solution: C1, A1, F1, B1, E2, H2, D1, J1, I2, G1

(b) (5 points) Next, write a function called `all_employees` that take the management dictionary and returns a list of all the employees.

For example, for this dictionary:

```
{'Alice': ['Chen'],
'Ayumu': ['Greny', 'Amelie', 'Jaeyun'],
'John': []
}
```

`all_employees` must return:

```
['Alice', 'Chen', 'Ayumu', 'Greny', 'Amelie', 'Jaeyun', 'John']
```

Solution:

```
def get_employee_list(management:dict) -> list:
    lst = []
    for key in management:
        lst += [key]
        lst += management[key]
    return lst
```

(c) (5 points) Write a function called `print_with_roles` that prints the name of each employee and either says that they are managers or indicates the name of the manager that their are managed by.

For example, for this dictionary:

```
{'Alice': ['Chen'],
'Ayumu': ['Greny', 'Amelie', 'Jaeyun'],
'John': []
}
```

`print_with_roles` must print:

```
Alice is a manager
Chen is managed by Alice
Ayumu is a manager
Greny is managed by Ayumu
Amelie is managed by Ayumu
Jaeyun is managed by Ayumu
John is a manager
```

Solution:

```
def print_with_roles(management:dict) -> None:
    for manager in management:
        print(manager, 'is a manager')
        for managee in management[manager]:
            print(managee, 'is managed by', manager)
```

(This page is intentionally blank. Label any work with the corresponding problem number.)