CS8 Midterm Exam #2 Study Guide and Sample Questions

Study Guide

For the midterm #2 exam, the topics you will be tested on include:

1) The use of functions
2) Conditional statements (if-else / if-elif-else / nested if-else)
3) Loops (for statements / while statements)
4) String formats
    a. This includes stuff we learned about string operations, indexing and slicing, string methods, etc…
5) File input/output

Of course, you should still know your stuff re: Python data types, their arithmetic operations, variable naming, lists, tuples, print vs. return, and related topics (it's hard not to bring these up since learning programming in Computer Science, like Math, is cumulative by nature).

What follows are some sample questions to try out while studying.
Some of these may seem challenging. Take your time in your analysis. If you are asked to write a program (or code, in general), think of the problem at a high-level first (that is, think of it *algorithmically*) before you even attempt at writing code.

Instructions: (Please read this)

- Read the questions carefully – make sure you understand what is being asked.
- Write out the answers by HAND FIRST. If you like, you can then check your answers by trying out the various codes on IDLE. Remember that, in the exam, you will have to present all your answers in WRITTEN form.
- Check your answers in the "Answers" section of this document.

CS8 Midterm Exam #2 Study Guide and Sample Questions

Questions:

1. What is the exact output of this Python code?

```
def Pick(L):
for x in L:
     if x[1] in ('a', 'e', 'i') and len(x) > 7:
          print(x, end="!\n*\n")


ListOSongs = [['Alive', 'Even Flow', 'Jeremy', 'Daughter',
'Better Man'], ['The Wall', 'Hey You', 'Wish You Were Here',
'Comfortably Numb'], ['Lost Cause', 'Sunday Sun', 'Paper Tiger']]
for l in ListOSongs:
     Pick(l)
```

2. What is the exact output of this Python code?

```
mer = ['Google', 73, -12, 'Apple', 'Amazon', 55, 0]
mer.append(3.1)
mer.append(mer[1])
for i in range(len(mer) - 6):
   mer.append(len(mer))
print(mer)
```

3. What is the exact output of this Python code?

```
def func1(alist):
   alist = alist + list(range(4))
   m = max(alist) - min (alist)
   n = (sum(alist)**2) // m
   return list(range(3, n))


L = [6, -4, 1, 0]
for x in func1(L):
     print(x + 2, end=".")
```

4. Explain why this Python code must fail and what you can change in the function definition (ONE change) to make it work?

```
def tree(a):
    a = a.split("/")
    print('{:3.4f}'.format(a[1]))

tree("3/44/9")
```

5. Write Python code that keeps asking, using a **while** loop, the user for an integer between 0 and 9 (inclusive) and prints the double of that number until the user enters a number outside the range of 0 thru 9. When the latter happens, the program should print out the sum AND the average (to 2 decimal points) of all the numbers that the user previously entered.

   An example run would look like this (user response is in **bold**).

```
Enter a number between 0 and 9: 7
14
Enter a number between 0 and 9: 3
6
Enter a number between 0 and 9: 9
18
Enter a number between 0 and 9: 10
The sum of all the numbers you entered is: 19
The average of all these numbers is: 6.33
```

6. Write Python code that writes a string, “Here they are:”, followed by all the numbers between 5 and 9 (inclusive) – all to a file called “outputs.txt”. Each entry is written on a separate line. You **must** use a for-loop.

7. Find the 5 mistakes in this Python code:

```
infile = open('data.txt', 'read')
x = infile.read()
x.append('x')
for y in x
     if y = 'y':
          print y
```

CS8 Midterm Exam #2 Study Guide and Sample Questions

Answers: NOTE that some of these answers are not unique, which means, particularly for the coding questions, there might be more than 1 way to solve these. As long as your answer is functionally correct AND you've only used instructions that we've covered in class, you'll get full credit on an exam.

1.

```
Daughter!
*
Better Man!
*
Wish You Were Here!
*
Paper Tiger!
*
```

(a newline at the end)

2.

```
['Google', 73, -12, 'Apple', 'Amazon', 55, 0, 3.1, 73, 9, 10, 11]
```

Explanation:
After the first 3 lines, when the for-loop is set-up, `len(mer)` is equal to 9. So the loop is set up `for i in range(3)`. Inside the loop, we append the list `mer` with the number `len(mer)`, which starts off as 9, but then becomes 10 (because the list `mer` now has an extra item, namely, the number 9 at the end of it), and then becomes 11.

3. `5.6.7.8.9.`

Explanation:
This program utilizes the `func1()` function with list `l` as its input parameter. So, first, let's figure out what the `func1()` function DOES with the list `l`:

The first line in the function makes `alist = [6,-4,1,0,0,1,2,3]` because the '+' operator on a list just appends 2 lists together. Note that `list(range(4))` is a list of [0, 1, 2, 3].

The 2nd and 3rd lines calculate m = 6 – (-4) = 10, and then n = (9**2) // 10 = 8. The function thus returns a list of range(3, 8) = [3, 4, 5, 6, 7].

This means that the for-loop (outside the function) is really:

```
for x in [3, 4, 5, 6, 7]:
```

This loop then prints out the x value plus 2. Additionally, each number printed is separated from the next one by the "." character.

4. It fails because when you **split()** a string, you get a **list** of ***strings***. So when you try to execute the instruction: `print('{:3.4f}'.format(a[1]))`, you get an error because it's expecting a **number** to format using the **:3.4f** (recall: f means floating point).

   The fix is to make sure that what we're printing is not a string (a[1]), but an integer, so we should modify the line to be:

   ```
   print('{:3.4f}'.format(int(a[1])))
   ```

5.

   ```
   sum = 0
   count = 0
   n = int(input("Enter a number between 0 and 9: "))
   while (n >= 0) and (n <= 9):
         print(2*n)
         sum += n
         count += 1
         n = int(input("Enter a number between 0 and 9: "))
   av = sum/count
   print("The sum of all the numbers you entered is:", sum)
   print("The average of all these numbers is: ", end="")
   print('{:0.2f}'.format(av))
   ```

   Note: the last 2 lines can also be combined as:

   ```
   print("The average of all these numbers is:",
   '{:0.2f}'.format(av))
   ```

6.

   ```
   outfile = open('outputs.txt', 'w')
   outfile.write('Here they are:\n')
   for j in range(5, 10):
        outfile.write( str(j) + '\n' )
   outfile.close()
   ```

   When writing the numbers from the loop into the output file, they HAVE to be written as **strings**. Hence the need for "**str(j)**" in the answer. Note also that you have to have the newline characters in there expressly. Plus, don't forget the "**outfile.close()**" statement at the end!!

7.

   1. Line 1: 'read' should be 'r'
   2. Line 3: you can't use x.append() since x is not a list, but a string.
   3. Line 4: colon is missing at the end of the for statement
   4. Line 5: you can't use =, you must use == instead
   5. Line 6: it should be print(y)