ECS 32A: Practice Problem Set #1 for Exam #1

Instructor: Aaron Kaloti

Summer Session #1 2020

Contents

1 Changelog

You should always refer to the latest version of the syllabus.

- v.1: Initial version.

2 Problems

2.1 Problem #1

Order the following strings according to the rules discussed during class, towards the end of the lecture #1 slides. That is, in your ordering, each string must be "less than" the string after it.

- "abc"
- "aaaa"
- "Aba"
- "acaA"
- "zzz"

2.2 Problem #2

Order the following strings according to the rules discussed during class, towards the end of the lecture #1 slides. That is, in your ordering, each string must be "less than" the string after it.

- "ma"
- "mad"
- "maD"

- “mmd”
- “mAd”

2.3 Problem #3

For each of the programs below, can short-circuit evaluation prevent the program from crashing? If so, explain how. If not, explain how the program can be modified so that short-circuit evaluation can prevent the program from crashing. (When I say “program”, I mean that that is all the code of that program; there is no unshown code.)

2.3.1 Program #1

```
x = int(input("Enter: "))
if x < 10 or 5 % x == 2:
    print("AAA")
else:
    print("BBB")
```

2.3.2 Program #2

```
firstInput = False
# Keep looping until the user enters 8.
while user_input != 8 or firstInput == False:
    user_input = int(input("Enter integer: "))
    if firstInput == False:
        firstInput = True
```

2.4 Problem #4

Write a program that asks the user for one integer and prints all integers that are between 3 and the given integer (inclusive), in order from *highest to lowest*. You may assume that the user will always enter an integer greater than 3. You must use *one and only one* loop in this program.

You may not store the integers in a data structure such as a list. You must use a `while` loop.

Here are some examples of how your program should work:

```
Enter: 5
5
4
3
```

```
Enter: 8
8
7
6
5
4
3
```

2.5 Problem #5

Remove as many lines of code as possible from the below program, without changing the output of this program. That is, whatever output that results from the user entering a specific input should be the same regardless of any changes that you make to the code shown below.

Write the letters of the lines you would cross off. You may only delete lines; you may not change or add lines. Deleting a line does not change any other line and does not change any other line’s indentation; for example, if you delete line F, then line G will still be indented.

If a given line has no associated letter, then that means I am not allowing you to cross it off.

```
n = int(input("Enter: "))         # A
i = 0                             # B
j = 1                             # C
while i < n:
    if i == 4:                    # D
        pass                      # E
    if i < 5:                     # F
        print(i)                  # G
    elif i < 3:                   # H
        print(2 * i)              # I
    else:                        # J
        print(3 * i)             # K
    i += 1                       # L
    j += 2                       # M
    continue                     # N
```

2.6 Problem #6

This is an example of a question that I would probably consider too difficult to ask for this first exam.

Write a program that keeps prompting the user for the radius of a circle until the area of said circle would be greater than 100. Alternatively, if the user enters "end", then that should stop the program. You may assume pi is 3. The radius of a circle is given by pi times the radius squared. For example, if the radius is 2, then the area is 3 times 2 squared, which is 3 times 4, which is 12. You may assume the user will only enter numbers (which could include floating-point numbers). Do not worry about rounding *at all*.

Hint: Use `float()` to convert a string (e.g. "4.2") to a floating-point number.

Here are some examples of how your program should work (ignore the comments, which are in square brackets and are menat to assist your understanding):

```
Enter: 2        [area is 12, so program will not end]
Enter: 5        [area is 75, so program will not end]
Enter: end
```

```
Enter: 10       [area is 300, so program will end]
```

```
Enter: 4.2       [area is 52.92, so program will not end]
Enter: 50
```

2.7 Problem #7

Write a program that prompts the user for two integers. If the first integer is larger, then the program should print the sum; otherwise, the program should print the difference (i.e. the first integer minus the second integer).

In problems like these and the next several ones, I would provide examples of what the input/output prompts should look like on the real exam.

2.8 Problem #8

Write a program that prompts the user to enter an integer and prints the result of adding 2 to that integer and then multiplying it by 3.

2.9 Problem #9

Write a program that prompts the user to enter two strings and then tells the user if the first string they entered is "Hello" *and* the second string they entered is "World".

2.10 Problem #10

Write a program that prompts the user for two integers and prints all integers between those two integers (exclusive) in descending order. You may assume the second integer will always be higher than the first integer.

UCDAVIS
COMPUTER SCIENCE