**CSE 101 - Introduction to Programming**

**Lab 3**

**Name: Bhavay Aggarwal**

**Roll No.:2018384**

**Section: B**

**Group: 1**

**General Instructions:**

* For today’s lab, you need this handout, plus the various python files included in the zipped folder.
* Rename this document to Lab3\_Solutions\_2018xxx.doc, where 2018xxx is your roll number and write the answers in the document itself.
* Apart from this the questions marked as *Code Submission* require you to make python script files with the naming convention mentioned.
* ***Important:*** *Also, inside the all the script files add the following lines as comments:*
  + *Roll No.:*
  + *Name:*
  + *Section:*
  + *Group:*
* Make folder with the name Lab3\_2018xxx, put all the files ( 1 solution document and 2 script files created by you as a part of this lab), zip it and upload.

**Section 1. Printing statements to find values, errors and if, elif and else variations**

Hyphenator\_even.py is a slightly different version of the hyphenation program you saw in class. Notice the inclusion certain print statements that are designed to show you the contents of key variables:

• print ('n is',n)

• print ('even case. m=',m)

• print ('even case. first=',first)

• print ('even case. second=',second)

These are debugging print statements. (The final line print s + ' becomes ' + h is not a de- bugging print statement, but rather, one that creates the desired final result of the program.)

In the command shell, see what the program does: run the program by typing python Hyphenator even.py. When asked for input, try 'abcdef' ; the last line of the output should say that the program turns this into 'abc-def'.

Now, let’s artificially insert some errors and see how these print statements show that some variables then no longer have the “right” values in them.

Open Hyphenator even.py.

***Q1. In Line B, assign s[:m-1] to first instead of s[:m]. Then, in the command shell, run Hyphenator\_even.py. (That is, type python Hyphenator\_even.py in the command shell. Don’t forget to save the file first.) Give the input 'abcdef'.***

*Which variable has a changed value, and what should the correct value have been? Which print line tells you this? Write the print line below.*

Output- Enter an even-length string: abcdef

n is 6

even case. m is 3

even case. first is ab #value changed

even case. second is def

abcdef becomes ab-def

Print line- print ('even case. first is',first)

The value of m changed in the line (first = s[0:m-1] # Line B) which changed the slicing of the string and the m-1th character was not sliced in our program i.e ‘c’. If no changes were made to the program it would’ve worked correctly.

***Q2. Fix the mistake you made above. Now, introduce a new artificial mistake in Line C, assign s[m+1:] to second instead of s[m:]. Suppose you run Hyphenator\_even.py and give it input 'abcdef'.***

*Which variable has a changed value, and what should the correct value have been? Which print line tells you this? Write the print line below.*

Enter an even-length string: abcdef

n is 6

even case. m is 3

even case. first is abc

even case. second is ef #value changed

abcdef becomes abc-ef

Print line- print ('even case. second is',second)

The value of m changed in the line (second = s[m+1:] # Line C)which changed the slicing of the string and the mth character was not sliced in our program i.e ‘d’. If no changes were made to the program it would’ve worked correctly..

***Q3. Code Submission 1:***

*Currently the script works only for even length strings. If we were to modify the above script so that it works for odd length strings also how would you do it? Sample inputs and their expected outputs are given in the table below. Add your lines of code to the already existing script file and name it Odd\_Even\_2018xxx.py.*

|  |  |
| --- | --- |
| Sample Input | Output |
| Null String | - |
| a | a - |
| ab | a - b |
| abc | ab - c |
| abcd | ab - cd |

**Section 2. Finding errors using print statements**

Now, we give you practice in adding print statements to see if there are errors in a program. The point: learn to use print statements to figure out if your code is working or not. Read the docstring for Hyphenator\_broken.py (the comment in triple quotes) in the following program to understand what the program is supposed to do.

Run program Hyphenator\_broken.py and give it input 'abcde' to see that it currently doesn’t work correctly. Open the file Hyphenator broken.py. Insert enough print statements so that when you run the program in the command shell, you know what lines are incorrect. This corrected code is needed for the Q4.

***Q4. Code Submission 2:***

*Use the corrected program above so that it works according to the input-output table given below. Add your lines of code to the already existing script file and name it Double\_Hyphen\_2018xxx.py.*

|  |  |
| --- | --- |
| Sample Input | Output |
| Null String | Print on screen - Null String |
| a | Print on screen - Single Character String |
| ab | Print on screen - Even Character String |
| abc | a - b - c |
| abcd | Print on screen - Even Character String |

**Section 3. Boolean Expressions**

For each example in this section, write down what you think is the value of the expression. Then use Python in interactive mode to find out if your mental reasoning is OK or not. If not, try to figure out why Python gave the answer that it did.

***Q5. Numerical Examples***

Assume that x,y and z are initialized as follows

>>> x = 1

>>> y = 5

>>> z = 10

Now complete this table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Expression | I Think the Value Is | Python Says | Notes |
| 1 | x<z | True | True |  |
| 2 | 2\*y >= z | True | True |
| 3 | 2\*y < z | False | False |
| 4 | (x>1) or (z!=7) | True | True |
| 5 | y != (z/2) | False | False |
| 6 | (x>0) or ((y>0) and (z<0)) | True | True |
| 7 | ((x>0) or (y>0)) and (z<0) | False | False |

***Q6. String Examples***

Assume that x, y, and z are initialized as follows:

>>> x = 'Cornell'  
>>> y = 'Harvard'

>>> z = 'Yale'

Now complete this table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Expression | I Think the Value Is | Python Says | Notes |
| 1 | x!=z | True | True |  |
| 2 | x == 'cornell' | False | False |
| 3 | len(x) > len(y) | False | False |
| 4 | y[1:] > z[1:] | True | True |
| 5 | len(x+z) > len(y) | True | True |