**Week 7 In-Class Exercises (Review - Strings and Lists)**

**Q1: Code Tracing [ \*\* ]**

Given the following pieces of code, what are the outputs? Use memory state diagram to help you.

**(a)**

def do\_trick(a\_list):

a\_list = a\_list + a\_list[1:3] + [a\_list[3]]  
 print(a\_list)

my\_list = ['a', 'b', 'c', 'd']  
do\_trick(my\_list)  
print(my\_list)

['a', 'b', 'c', 'd', 'b', 'c', 'd']

[‘a’, ‘b’, ‘c’, ‘d’]

**(b)**

def do\_trick(a\_list):

a\_list.append(a\_list[1:3] + [a\_list[3]])

print(a\_list)

my\_list = ['a', 'b', 'c', 'd']

do\_trick(my\_list)

print(my\_list)

['a', 'b', 'c', 'd', ['b', 'c', 'd']]

['a', 'b', 'c', 'd', ['b', 'c', 'd']]

**(c)**

my\_str = '123'

for index in range(0, len(my\_str)–1):  
 n = int(my\_str[index] + my\_str[index+1])  
 print(n)  
 m = int(my\_str[index:index+2])  
 print(m)

12

12

23

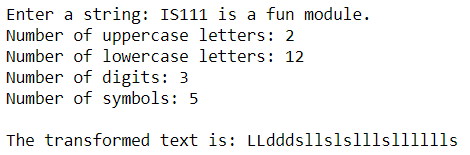
23

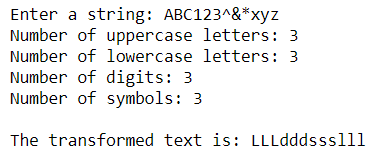
**Q2: String Transformation [ \*\* ]**

Inside a file called **q2.py**, write a function called transform\_string(). The function takes in a string as its parameter. The function transforms the string as follows:

* Any uppercase letter is replaced by the character 'L' to indicate that it’s a letter in uppercase.
* Any lowercase letter is replaced by the character 'l' to indicate that it’s a letter in lowercase.
* Any digit is replaced by the character 'd' to indicate that it’s a digit.
* Any other character (e.g., a space, a semi-colon, etc.) is replaced by the character 's' to indicate that it’s symbol.

The function ***returns*** the transformed string. The function ***also prints out*** the numbers of each group of characters.

We have a test file called **q2\_test.py**. DO NOT MODIFY THIS FILE! When **q2\_test.py** is run, we should see the following output:



**Q3: Triangle and Frame [ \*\* ]**

1. Inside a file called **q3.py**, write a function called print\_triangle(ch, num\_rows) that takes in a character and a number as its parameters. The number indicates the number of rows. The function ***prints out*** a triangle using the character.

For example, if ch is '\*', num\_rows is 3, the following output is to be generated:

\*  
 \*\*\*  
\*\*\*\*\*

If ch is '#', num\_rows is 5, the following output is to be generated:

#  
 ###  
 #####  
 #######  
#########

1. Inside **q3.py**, write another function called print\_frame(ch, num\_rows, num\_cols) that takes in the same three parameters as above. The function ***prints out*** a frame as shown below. For example, if ch is '\*', num\_rows is 3 and num\_cols is 4, the following output is to be generated:

\*\*\*\*  
\* \*  
\*\*\*\*

If ch is '#', num\_rows is 5 and num\_cols is 6, the following output is to be generated:

######  
# #  
# #  
# #  
######

You can use **q3\_test.py** to test your code.

**Q4: Pangram [ \*\*\* ]**

Inside a file **q4.py**, write a function called is\_pangram to determine if a sentence is a pangram. A pangram is a sentence using every letter of the English alphabet at least once (either in uppercase or in lowercase). The best-known pangram is:

"The quick brown fox jumps over the lazy dog."

The function ***returns*** True if the given sentence is a pangram or False if it is not.

You can use **q4\_test.py** to test your code.