Course Title: Programming Language I
Course Code: CSE 110
Assignment no: 3

This Assignment is to help you develop your concept of Strings in Python.

[Please follow variable naming conventions including using meaningful variable names for all the tasks]
When you run your code, please make sure your outputs exactly match the sample outputs for each of the sample inputs given. Check if your code works for other valid inputs not given in the samples.

Write the code in Python to do the following tasks:

Task 1

Write a Python program that takes a String as an input from the user and prints that String in **reverse** order. **You cannot use the built-in reverse() function for this task.**

==========

Sample Input 1:
CSE110
Sample Output 1:
011ESC

==========

Sample Input 2:
Python
Sample Output 2:
nohtyP

==========

Sample Input 3:
1576527
Sample Output 3:
7256751

==========

Task 2

Write a python program that takes 2 inputs from the user, where the first input is a string with length greater than 1. The second input is the index of the first given string from where you have to start reversing. After reversing the first input string from that index, print the new string back to the user. See samples below for clarification.

==========

Sample Input 1:
72418 4
Sample Output 1:
81427

Explanation: Our second input, index '4' is the last index of our first input String '72418', hence the entire string is reversed giving us '81427'.

Sample Input 2:
12345 2
Sample Output 2:
32145

Explanation: The second input is '2' so we have to reverse from index 2 of our first input. The 2nd index of our first input String is '3', index 1 is '2' and index 0 is '1'. Hence, if we reverse indexes 0 to 2, we get '321'. Index 3 and 4 which is '4' and '5' respectively remains unchanged hence our final output is '32145'.

Sample Input 3:
aBcd1234defg
5
Sample Output 3:
21dcBa34defg
Explanation: From our first input String 'aBcd1234defg',
index 0 = 'a' index 1 = 'B' index 2 = 'c' index 3 = 'd' index
4 = '1' index 5 = '2' index 6 = '3'
Since our second input is 5, index 0 to index 5 is reversed and we have '21dcBa' and the rest is unchanged from indexes 6 to 11 ('34defg'). Therefore, we have '21dcBa34defg' finally.
==========

Task 3

Write a program that takes a **string** as input and prints “Binary Number” if the string contains only 0s or 1s. Otherwise, print “Not a Binary Number”.

==========

Sample Input 1:
01101101101

Sample Output 1:
Binary Number

==========

Sample Input 2:
12344ab0

Sample Output 2:
Not a Binary Number

==========

Sample Input 3:
10127490111

Sample Output 3:
Not a Binary Number

==========

Sample Input 4:
Binary Number

Sample Output 4:
Not a Binary Number

==========

Task 4

Write a Python program that will ask the user to enter a word as an input.

- If the length of the input string is less than 4, then your program should print the same string as an output. • If the input string's length is greater than 3, then your program should add "er" at the end of the input string.
- If the input string already ends with "er", then add "est" instead, regardless of the length of the input string
- If the input string already ends with "est", then your program should print the same input string as an output.

==========

Sample Input 1:
strong

Sample Output 1:
stronger

Explanation: Length of input = 6, not less than 4, greater than 3, does not end with "er" or "est", therefore "er" is added making "strong", "stronger".

==========

Sample Input 2:
stronger

Sample Output 2:
strongest

Explanation: Input string ends with "er", therefore "er" is replaced with "est" instead so we have "strongest" from "stronger".

==========

Sample Input 3:
strongest

Sample Output 3:
strongest

Explanation: Our input here already ends with "est" so the same input i.e. "strongest" is printed.

==========

: :

Sample Input 4 abc

Sample Output 4

abc

Explanation: Since length of input string is less than 4, the given input is printed as output.

Sample Input 5: ber

Sample Output 5:
best

Explanation: Although the length of the input string is 3 which is less than 4, but it ends with er so we ignore the length and replace "er" with "est" regardless.

Task 5

Write a Python program that will ask the user to input a string (containing exactly one word). Then your program should print subsequent substrings of the given string as shown in the examples below.

==========

Sample Input 1:
BANGLA

Sample Output 1:
B
BA
BAN
BANG
BANGL
BANGLA

Explanation: The length of the string is 6 so there are 6 lines where in each line a substring of the input string, of length equal to the line number is printed i.e. substring with only the letter "B" printed in the first line, substring "BA" of length 2 printed in the 2nd line, "BAN" length of which is 3 being printed in the 3rd line and so on.

: :

Sample Input 2
DREAM

Sample Output 2
D
DR
DRE
DREA
DREAM

Explanation: Simply, no of lines = length of the input string and no of letters in each line = line number.

==========

Task 6

Write a Python program that will ask the user to input a string (containing exactly one word). Then print the ASCII code for each character in the String using the ord() function.

To check if your program is working correctly or not, you can find a list of all correct values from the following website. You may look at "Dec" and "Char" columns only and ignore the other columns for this problem. link: http://www.asciitable.com/

==========

Sample Input 1:
Programming

Sample Output 1:
P : 80 r
: 114 o
: 111 g
: 103 r
: 114 a
: 97 m
: 109 m
: 109 i
: 105 n

: :

: 110 g
: 103

Explanation: Each line prints a letter sequentially from the given string and its corresponding ASCII value separated by " : ".

==========

Sample Input 2 hunger

Sample Output 2

h : 104
u : 117
n : 110
g : 103
e : 101 r
: 114

Explanation: Same as previous.

==========

Task 7

Write a Python program that takes a string as an input from the user containing all small letters and then prints the next alphabet in sequence for each alphabet in the input.

Hint: You may need to use the functions ord() and chr(). The ASCII value of 'a' is 97 and 'z' is 122.

==========

Sample Input 1:
abcd

Sample Output 1:
bcde

==========

Sample Input 2:
the cow

Sample Output 2:
uif!dpx

==========

Sample Input 3: [Must fulfil this criteria] xyzabc

: :

Sample Output 3:

yzabcd

==========

Task 8

Write a Python program that takes a String as input from the user, removes the characters at even index and prints the resulting String in uppercase without using the built-in function upper().

==========

Sample Input 1:
String

Sample Output 1:
TIG

Explanation: The characters 'S', 'r' and 'n' are at index positions 0, 2, and 4 respectively. Hence they are removed and the remaining characters 'tig' are capitalized giving us output 'TIG'.

==========

Sample Input 2:
abcd

Sample Output 2:
BD

Explanation: Only the characters at the odd indices, 1 and 3, 'b' and 'd' are captitalized, concatenated and printed.

==========

Task 9

Given a string, create a new string with all the **consecutive duplicates removed**.

Hint: You may make a new string to store the result. You can check whether the current character and the next character are the same, then add that character to the new string.

Note: Only consecutive letters are removed not all duplicate occurrences of a letter. You may try doing this alternative i.e., removing all duplicate letters from a given string, for your own practice.

Sample Input 1:
AAABBBBCDDBBECE

Sample Output 1:
ABCDBECE

Explanation: From the 3 consecutive "A"s, 2 are removed and we have 'A' only. Then from the 4 consecutives 'B's, 3 are removed and only one is added to the new string giving us "AB". Since we have only one 'C' next, it is added making the resulting string "ABC" now and so on.

Sample Input 2:
Jupyter Notebook is better. Case sensitivity check AAaaaAaaAAAa.

Sample Output 2:
Jupyter Notebok is beter. Case sensitivity check AaAaAa.

Explanation: Just the 2 consectutive 'o's and 't's are changed to one at first and the uppercase 'A' and lowercase 'a' are treated separately i.e., case sensitive when checking for consecutive duplicates.

==========

Task 10

Write a Python program that will take one input from the user made up of two strings separated by a comma and a space (see samples below). Then create a mixed string with alternative characters from each string. Any leftover chars will be appended at the end of the resulting string.
[Do not use lists for this task]

Hint: For adding the leftover characters you may use string slicing.

==========
Sample Input 1: ABCD, efgh

Sample Output 1:
AeBfCgDh

Explanation: At first, the two strings divided by ", " should be separated. Then the first character of the first string 'A' is concatenated with the first character of the second string 'e' which in turn is concatenated to the second character of the first string 'B', the second character of the second-string f and so on since the strings are of equal length.

==========
Sample Input 2: ABCDENDFGH,
ijkl

Sample Output 2:
AiBjCkDlENDFGH

Explanation: Here, since the length of the first string is greater than the length of the second string, after separation, the characters are concatenated alternatively as in sample input/output 1, till the length of the second string i.e., ijkl. Since, there are no more characters in the second string after that, the remaining characters if the first string i.e., ENDFGH in this case are concatenated at the end of the final string.

==========

Sample Input 3: ijkl,
ABCDENDFGH

Sample Output 3:
iAjBkClDENDFGH

Explanation: This time, the length of the second string is greater than the length of the first string therefore the first letters of the 2 strings 'i' and 'A', then the second letters 'j' and 'B' and so on are being concatenated until there are no more letters in the first shorter string following which the remaining letters i.e., ENDFGH again in this case too (this may be different for other different string inputs) are added at the end giving us the resulting output string.
