Python HW #5

Name: Chris Welch

1. Split a String Based on Vowels and Consonants

Write a function Splitter that takes a string as an input, breaks it up and returns a new string with consonants first and vowels second. Vowels are “a, e, i, o, and u”. For any character that's not a vowel (like special characters or spaces), treat them like consonants. Assume the input string can be of any length.

For example:

"abcde" ➞ "cdeab"
“Hello!” ➞ “Hll!eo”
"What's the time?" ➞ "Wht's th tm?aeie"

```
#-----------------Problem 1-----------------
def Splitter():
    strInput = input("Enter a Word or Sentence for the Splitter:")
    vowels = ['a','e','i','o','u']
    vowelBuffer = []
    consonantBuffer = []
    for x in strInput:
        if x in vowels:
           vowelBuffer.append(x)
        else: consonantBuffer.append(x)
    print("\""+ ''.join(consonantBuffer) + ''.join(vowelBuffer) + "\"")

Splitter()
```

2. Create a dictionary from 2 lists

Write a function NewDict that takes two lists as an input (keyList and valueList) and returns a dictionary from combining the two input lists into key-value pairs for that dictionary. Assume that the two input lists can be of any length, but both lists have to be the same length. Also assume the first item in keyList matches with first item in valueList, the second item in keyList matches with second item in valueList, and so on.

For example:
keyList = [“shirts”, “pants”, “socks”, “shoes”]
valueList = [2, 4, 6, 8]
clothes = {“shirts”:2, “pants”:4, “socks”:6, “shoes”:8}

```
#-----------------Problem 2-----------------
def newDict():
    numOfInputs = int(input("How many Inputs do you want for your dictionary?\t"))
    keyList = []
    valueList = []
    while numOfInputs > 0:
        val = input("Please Enter a Key:\t")
        keyList.append(val)
        val = input("Please Enter a Value:\t")
        valueList.append(val)
        numOfInputs -= 1
    valSpot = 0
    newList = []
    zipObj = zip(keyList, valueList)
    newdiction = dict(zipObj)
    print("YOUR DICTIONARY IS:\n" , newdiction)
newDict()
```

3. Encrypt a string

Write a function Encrypt that takes string as an input and returns it's encrypted equivalent. The hashing algorithm for doing the encryption is a simple character substitution:

For alphabetic characters:

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

For numeric characters: leave as is, i.e. no substitution is done, except add a leading zero to the number, i.e. "0" ➞ "00", "1" ➞ "01", etc.

Assume the input string contains only alphanumeric characters (a thru z and 0 thru 9), e.g. no spaces or special characters, each character in the input string will translate to two digits in the encrypted string (see example), and the input string can be of any length. Also assume no difference between uppercase and lowercase alpha characters.

For example: input string = "happy9day7" ➞ "08011616250904012507"

```
# ------------Problem 3-------------------
def simpleEncryption():
    encryptString = input(
        "Enter alphanumeric characters (a thru z and 0 thru 9), e.g. no spaces or special characters\n").lower()
    print(encryptString)
    encryption = []
    for x in encryptString:
        if x.isalpha():
            if 10 > ord(x) - 96:
                encryption.append(str("0" + str(ord(x) - 96)))
            else:
                encryption.append(str(ord(x) - 96))
        if x.isdigit():
            encryption.append(str("0" + x))
    encryptedString = ""
    encryptedString = encryption[0:len(encryption)]
    print(str(encryptedString).replace('\'', "").replace(',', "").replace(" ", "").replace("[", "").replace("]", ""))

simpleEncryption()
```

---

4. Running total of product and sum of integers input by a user

Write a function ProdSum that asks a user to input a number from one to 99. If the input number is less than or equal to 25, then calculate the running product of each number inputted by the user that is less that or equal to 25. If the number is greater than 25, then calculate the running sum of each number inputted by the user that is greater that 25. If the user enters zero as input, then print out the total running product calculated and the total running sum calculated and end the function.

Assume the user can input as many numbers as desired until inputting a zero. Make sure you test the number inputted by the user to ensure it is in the range from zero to 99 and inform the user if the number is not in the range and to try again. Also print the appropriate labels for the two calculations in your print statements, as well as the number inputted by the user..

For Example:
Enter a number from 1 to 99 or 0 to Exit: 5
Enter a number from 1 to 99 or 0 to Exit: 83
Enter a number from 1 to 99 or 0 to Exit: 21
Enter a number from 1 to 99 or 0 to Exit: 55
Enter a number from 1 to 99 or 0 to Exit: 13
Enter a number from 1 to 99 or 0 to Exit: 64
Enter a number from 1 to 99 or 0 to Exit: 0

The product of the numbers [5, 21, 14] is 1470
The sum of the numbers [83, 55, 63] is 201

```
#-------------Problem 4-------------------
def ProdSum():
    userInput = 1
    productTotal = 1
    sumTotal = 0
    inputHistorySum = []
    inputHistoryProduct = []
    upperBounds = 99
    lowerBounds = 0
    breakNumber = 25
    while userInput:
        userInput = int(input("Input a number from 1-99\tInput 0 to End\t\t"))
        if userInput > upperBounds or userInput < lowerBounds:
            while userInput > upperBounds or userInput < lowerBounds:
                userInput = int(input("INVALID INPUT:\nInput a number from 1-99\tInput 0 to End\t\t"))
        if userInput <= breakNumber and userInput != lowerBounds:
            productTotal *= userInput
            inputHistoryProduct.append(userInput)
        if userInput > breakNumber and userInput <= upperBounds:
            sumTotal += userInput
            inputHistorySum.append(userInput)
        if userInput == lowerBounds:
            print("\n\n\n\t\tThe Product of the numbers " + str(inputHistoryProduct) + " is: %d\n\t\tThe
Sum of the numbers " % productTotal + str(inputHistorySum) + " is: " + str(sumTotal) + "\n\n\n")
ProdSum()
```