Assignment #9

General Instructions. For this assignment, you will be writing 2 separate programs. They should be submitted as 2 different files using NYU Classes. The files should be named:

Lastname-Firstname-assign9-1.py
Lastname-Firstname-assign9-2.py

1. Transposition Cypher (10 points).

The transposition cypher encrypts messages using an even-odd shuffle of characters. To encode a message, it creates two strings. The first string consists of the characters in the message that have an even index. The second string consists of the characters in the message that have an odd index. (This means that if \texttt{len(message)} is odd, then the first “even” string will be longer.) Then the two strings are concatenated (odd characters first) to form the encoded message.

Write a program to implement the transposition cypher. It should contain two functions, \texttt{encode(message)} and \texttt{decode(message)}. The arguments of both functions are strings, and both functions returns strings, encoded or decoded respectively.

The main part of your program should test your function on an input message. It should print the original message, the message encoded, and then the encoded result decoded. The decoded message should be the same as the original message.

For example, your output might look like:

‘If the facts don’t fit the theory, change the facts.’
‘ftefcsdntftteter,cag h at.I h at o’ i h hoy hnetefcs’
‘If the facts don’t fit the theory, change the facts.’
2. Extracting Words from Sentences (10 points)

This problem uses string and list methods to transform a sentence in the following steps.

1. Prompt the user to enter a sentence,
2. Print the sentence that was entered.
3. Remove all of the punctuation from the sentence
4. Capitalize all of the letters.
5. Split the sentence into a list of words (Hint: use the string split method)
6. Sort the list (Hint: use the list sort method)
7. Create a new list with duplicate words removed.
8. Print out the words in this list, along with how many times each word
   appears in the list created in step 5. (Hint: Use the count method.)

For example, consider this sentence entered in step 1:

‘If the facts don’t fit the theory, change the facts.’

Result after step 3:
‘If the facts dont fit the theory change the facts’

Result after step 4:
‘IF THE FACTS DONT FIT THE THEORY CHANGE THE FACTS’

List created in step 5:
['IF', 'THE', 'FACTS', 'DONT', 'FIT', 'THE', 'THEORY', 'CHANGE', 'THE', 'FACTS']

Result after step 6:
['CHANGE', 'DONT', 'FACTS', 'FACTS', 'FIT', 'IF', 'THE', 'THE', 'THE', 'THEORY']

New list after step 7:
['CHANGE', 'DONT', 'FACTS', 'FIT', 'IF', 'THE', 'THEORY']

Output from step 8:

<table>
<thead>
<tr>
<th>Word</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHANGE</td>
<td>1</td>
</tr>
<tr>
<td>DONT</td>
<td>1</td>
</tr>
<tr>
<td>FACTS</td>
<td>2</td>
</tr>
<tr>
<td>FIT</td>
<td>1</td>
</tr>
<tr>
<td>IF</td>
<td>1</td>
</tr>
<tr>
<td>THE</td>
<td>3</td>
</tr>
<tr>
<td>THEORY</td>
<td>1</td>
</tr>
</tbody>
</table>