Assignment 5 – Word Jumbled and Encryption

Goals

- Lists
- String processing and manipulation
- Loops

Requirements

- Create a new Python file for this assignment. You may put both parts into one file or create another Python file for the second part.
- Your new file(s) must begin with comments in the following format (replace the name and email with your actual information and write text for the description):

```
# Name, USC email
# ITP 115, Spring 2020
# Assignment 5
# Description:
# Describe what this program does.
```

Part 1 – Word Jumble Game

- **Requirements**
    - Start: create a list with words that you will use in the game. Use `random.choice (…)` to pick one of the words.
    - Create a jumbled version of the word and display it to the user.
        - This part is challenging, but there is a hint below to help. You should be able to complete it with what we have covered, and may not use automated methods you may find online (this means you may not use `random.shuffle(), random.sample(),`or similar methods)
        - Hint: For all word processing, use lists. Since strings are immutable, you cannot jumble the letters directly. However, you can make a jumbled copy of the original word. To jumble a word, you could rearrange the letters in the word. You should **not** manually code a jumbled version of your word. e.g.

          ```
          list = ["ape"]
          jumbledList = ["eap"]
          ```
    - Have the user guess the word until they get it correct.
    - Count the number of guesses it takes.
    - Extra Credit
        - Have each word paired with a hint.
        - The player should be able to see the hint if he or she is stuck.

- Add a scoring system that rewards players who solve a jumble without asking for the hint.

Part 2 – Encrypt / Decrypt

- **Background**
    - The Caesar cipher (named for Julius Caesar) is a simple method for encryption a message. This involves "shifting" each letter in the message by a fixed number.
    - For example, consider a shift of 3 letters

      Original alphabet: **`abcdefghijklmnopqrstuvwxyz`**

      Cipher alphabet: **`defghijklmnopqrstuvwxyzabc`**
    - To encrypted a message, substitute plain-text letters with corresponding cipher letter. Do **not** shift spaces or punctuation.
    - Using that new encryption key (alphabet),

      Original message: **`hello world.`**

      Encrypted message: **`khoor zruog.`**
    - To decrypt, repeat the process
- **Requirements**
    - Write a program that prompts the user for the shift value (e.g. 3) and then a plain-text message to encode. Encrypt the message and print out the encrypted message. Then use your program to decrypt the message back to its original state. Compare the decrypted message with the original.
- Hints
    - To create a cipher (shifted alphabet), first start with a list of alphabet letters as strings, and then use slicing to create the cipher.
    - Do not shift punctuation or spaces—simply leave them as is (see "hello world" from above)
    - When you are replacing letters, notice that the letter in the original alphabet is at the same index as the cipher letter you are going to replace it with.
    - Design your algorithm on paper before coding

Sample Output for Part 1

```
The jumbled word is "yhtnpo"

Please enter your guess: htnpoy

Try again.
```

```
Please enter your guess: poythn

Try again.

Please enter your guess: python

You got it!

It took you 3 tries.
```

Sample Output for Part 2

```
Enter a message: lamb biryani

Enter a number to shift by (0-25): 3

Encrypting message....

     Encrypted message: odpe elubdql

Decrypting message....

     Decrypted message: lamb biryani

     Original message: lamb biryani
```

Deliverables and Submission Instructions

- You may include part 1 and 2 in the same file as long as it is very clear through comments which part is which. You may also separate parts 1 and 2 into two files, as long as the names of the files are clear.
- Create a folder on your computer called **`ITP115_A5_`*`LastName_FirstName`*** (replace *LastName* with your last/family name and FirstName with your first name).
- Inside the folder, put your python source code.
- Compress the folder (make a zip file). This cannot be done within PyCharm. Find the folder on your computer and compress it.
  a. Windows:
     1. Using File Explorer, select your lab file
     2. Right click
     3. Send to ->
     4. Compressed (zipped) folder
  b. Mac OSX:
     1. Using Finder, select your lab file
     2. Right click
     3. Compress "*FileName*"
- Upload the zip file to your Blackboard section:
  1. On Blackboard, click on the Assignments item in the course menu on the left.
  2. Click on the specific item for this assignment (starts with A and a number).
  3. Click on the Browse My Computer button and select your zip file.
  4. Click the Submit button.

Grading

Item	Points
Part 1: Word Jumble	15
Part 2: Encrypt / Decrypt	15
Total*	**30**

** Points will be deducted for poor code style, lack of error checking, improper submission.*