Lab 09 - Files & Exceptions

Learning Objectives: Demonstrate understanding of reading text files and writing into a text file. Being able to implement exceptions into the code design.

So far, all of the data we used in our programs has been deleted once the program finishes running. If we want to be able to store data long term, between runs of the programs, then we need to be able to read and write to file. As an introduction to this skill, we are going to encrypt and decrypt a given text file for this lab. There are many ways to encrypt, and decrypt, data to make it more secure. We will be using a simple shift cipher, but check out this video for more about crytography.

Step 1: Create a function that determines encryption or decryption choice

Prompt the user to choose between encrypt or decrypt. The function should check that a valid response was given and loop until the user enters a valid response.

Step 2: Create a function that will read a text file

Within this function a user should be asked to enter the name of a text file that they would like to read from. This method should return a string that we will be manipulating later on. (Do not forget the extention of a text file: .txt)

Make sure you implement a try-except block into your code to prevent potential errors when opening the file.

Step 3: Write a function that will write to an output file

Within this function a user should be asked to enter the name of a text file that they would like to write to and whether to overwrite or append. This function should then perform the desired tasks.

Step 4: Set up the main method

In main, invoke the functions you wrote in steps 1 and 2 to get the choice and file name from the user. Also get input from the user for how much to shift (the key). Depending on the user choice of whether to encrypt or decrypt a given text file, your program should then perform the following operations:

1. If ecryption is chosen, then move each character of the text forward by the key value. Then, write the new encrypted text into a file named "Encrypted.txt".

2. If decryption is chosen, then move each character of the text backwards by the key value. Then, write the new decrypted text into a file named "Decrypted.txt".

Test

A sample text file called text.txt is provided for you to practice with. Choosing encryptian with a key of 3 should result in:

```
Lw#zdv#d#jrrg#lghd1#Dw#ohdvw/#wkh|#doo#wkrxjkw#lw#zdv#d#jrrg#lghd#dw#wkh#wlph1#Klqgvljkw#zrxog#uhyhdo#wkdw#lq#uhdolw|/#lw#zdv#dq#xqeholhydeo|#whuuleoh#lghd/#exw#lw#zrxog#wdnh#dqrwkhu#zhhn#iru#wkhp#wr#xqghuvwdqg#wkdw1#Uljkw#qrz/#dw#wklv#yhu|#prphqw1#wkh|#doo#djuhhg#wkdw#lw#zdv#wkh#shuihfw#frxuvh#ri#dfwlrq#iru#wkh#fxuuhqw#vlwxdwlrq1
```

Try out your functions with different text files and user choices to check that decryption/encryption is working correctly. Make sure that each created output text file is visible and performs the function according to the user's choice. Try encrypting a file, then decrypting the result. You should get the original text string back. Have fun!

Submit

There are no automated tests for this lab so make sure you have all of the required functions and that your program accomplishes the desired task. As always, stop by student hours, send an email, check in with a peer, or stop by the STEM Center if you need any assistance.

Hacker Challenge

In the above shift it is super easy to break the encryptian since we can determine the key by what the space is mapped to. Instead of shifting all characters, make a function that only shifts letters. They should loop around so that 'y' with a shift of 3 should be 'b'. All other characters (punctuation, spaces, numbers) should remain the same.

For example, with a shift of 3 the sample in text.txt would become:

```
Lw zdv d jrrg lghd. Dw ohdvw, wkhb doo wkrxjkw lw zdv d jrrg lghd dw wkh wlph. Klqgvljkw zrxog uhyhdo wkdw lq uhdolwb, lw zdv dq xqeholhydeob whuuleoh lghd, exw lw zrxog wdnh dqrwkhu zhhn iru wkhp wr xqghuvwdqg wkdw. Uljkw qrz, dw wklv yhub prphqw. wkhb doo djuhhg wkdw lw zdv wkh shuihfw frxuvh ri dfwlrq iru wkh fxuuhqw vlwxdwlrq.
```