Name: Rishbha Godara

Date: Feb 24, 2023

Assignment Title: Individual Programming 3 – Part 2

Summary of how the program should run:

The program reads a vocabulary list file, which contains terms and their definitions. It prompts the user to add new terms and their definitions to the list, updates the dictionary and saves it to a new file. The program starts by asking the user to enter a filename. If the file does not exist, the user is prompted to enter the filename again. Then, the program reads the file and prints the number of lines, which is the number of terms in the vocabulary list. The user is prompted to add new terms to the list. If the user enters 'Y', the program prompts the user to enter the number of terms to be added. It then creates a dictionary from the existing vocabulary list file. The program then prompts the user to enter new terms and their definitions, validates the input and updates the dictionary. The user is prompted again if they want to add more terms. If the user enters 'N', the program prints the updated vocabulary list and saves it to a new file. If the user enters 'N' initially, the program prints the original vocabulary list and saves it to a new file.

List of changes made to the original assignment:

1. Added exception handling in the numLines method for cases when the file is empty or corrupted and can’t be read.

2. Added docstrings to the class and methods. The docstring improves readability of the program by describing what the class and each method do, what arguments they take, and what they return.

3. Used the **with** statement when opening files in numLines method to ensure that they are closed properly, even if an exception occurs.

4. In the createDict method, added default headers “terms” & “definitions” to dictionary when the dictionary is empty or when the first key and value are not present. This allows users to add new terms and their definitions to an empty file improving and expanding the program’s usability.

5. createDict method also handles invalid input data by printing a message to the console, skipping the current line, and continuing to process the rest of the file.

6. Added type hints to function arguments and return values to improve readability of the program.

7. Use f-strings to format strings instead of concatenation.

Examples of the program (please also upload any attachments necessary):

*##Example 1 – exception handing of inputs*

Text

Description automatically generated

##Example 2 – adding term-definition pairs in multiple steps

Text

Description automatically generated

Errors that will crash the program or cause illogical problems:

Due to case sensitivity, the updateDict method's key search may result in duplicate entries in the updated dictionary and the corresponding text file if new keys are added with different cases. The following example demonstrates this phenomenon.Text

Description automatically generated

Additional examples or other comments/notes:

##Example 3 – saving keys without their definition

Text

Description automatically generated

##Example 4 – skipping lines where either the key or the definition is missing

Text

Description automatically generated

##Example 5 – key search

Text

Description automatically generated