DCS 229: Lab 1 **Python Review # 1** **Sun 14 Jan 2024**

Overview: In this lab, you will review previous Python experience, using conditionals, loops, functions, file I/O, and basic Python data structures such as lists and dictionaries.

Assignment: For this lab, enter your code in VSC (or your editor/IDE of choice), and then use the terminal to execute your programs. All of your functions can go in the same source file, which you should name `dcs229_lab1.py`. **In your program's block comment at the top, provide a list of all resources you used in completing this assignment, including a list of URLs as appropriate.**

1. Write a fruitful function named `readFile` having a single parameter corresponding to a filename. The function must open the file (assumed to be an ASCII text file with any number of words per line) for reading, and return a tuple containing two lists: the first list must contain the unique words in the file; the second must contain the corresponding count of each word in the file. You are not permitted to use a dictionary for this solution, i.e., your solution here must be entirely list-based.

   Use good style, including good naming conventions, type hints for all parameters and function return type, complete Google-style docstrings (including description of the function purpose, each parameter, and the return type), and use of exceptions when appropriate (e.g., when opening the file).

   For this case, ignore capitalization (e.g., "Snow" and "snow" should be considered the same word), and omit punctuation. If a word is possessive (e.g., "cat's"), drop the possessive part (e.g., "cat"); otherwise, words that are contractions (e.g., "don't") should be considered as a word.

2. Provided is a simple file `wilco.txt` containing the lyrics to a song by my favorite band, Wilco. Use this file (and at least two others of your own creating) to test your function carefully.

   Make sure to include your tests inside a separate single `main` function at the bottom of your program (**NOT** in the global scope), and then call your `main` function protecting it with an appropriate condition:

   ```
   def main():
       # your tests go here

   # only call main if executing this script directly (i.e., not when importing)
   if __name__ == "__main__":
       main()
   ```

   Remember that a good *regression* test inside `main` will clearly print what is being tested, what the expected result should be, and what your actual result is.

3. Modify the function to have a second parameter named `use_dict` with the default value of `False`. Make the necessary modifications to your function so that, whenever that second parameter is `True`, rather than using and returning two lists, your function will instead return a Python dictionary representing the same information. Remember to update all of your type hints and docstring appropriately.

   Again test with the provided file and at least two of your own files.

4. Write a function name `writeTopK` having three parameters: the first corresponding to a dictionary like that produced from the previous step above, the second corresponding to a filename, and the third corresponding to an integer k. Your function should write to the given file the top k words, in decreasing order of frequency (any ties should be output in alphabetical order), from the original input file. Each line of the new file produced should contain the word (left-justified to width 20), followed by a space, followed by the integer count (right-justified to width 4) of the number of times that word appeared in the original input file. If there are any ties for ith place, all words with that kth-place count should be included (again, in alphabetical order).

   Again test with the provided file and at least two of your own files.

5. **TO BE PROVIDED:** When you are ready to test your solution against the autograder (**after** you have thoroughly tested with your own tests), visit the appropriate Ed link posted on Lyceum.

6. In the online Lyceum text, provide answers to each of the following:

   (a) What grade (numeric and letter) would you objectively give to your work on this assignment, and justify your choice?
   (b) What was challenging on this assignment, if anything?
   (c) What do you still need work on, in terms of concepts covered in this assignment?
   (d) List all resources you used in completing this assignment (which you should have also included as comments in your program's top block comment).