CS:1210 Homework 4

Due via ICON on Wednesday March 11th, 4:59 pm

What to submit: Your submission for this homework will consist of four Python files, `hw4a.py`, `hw4b.py`, `hw4c.py`, and `hw4d.py`. These files should contain Python code that solve Problems (a), (b), (c), and (d) respectively. These files should each start with with a comment block containing your name, section number, and student ID. You will get no credit for this homework if your files are named differently, have a different format, and if your files are missing your information. Your program should all be well documented, i.e., have useful comments. We will discuss what constitutes good documentation and coding style in class. Part of good documentation is choosing meaningful names for your variables. The file `hw4a.py` should contain the definition of the `parse` function and any "helper" functions it depends on. The file `hw4b.py` should contain the definition of the `computeWordFrequencies` function and any "helper" functions it depends on. Since `computeWordFrequencies` uses the function `parse`, this should also be included in the file `hw4b.py`. The file `hw4c.py` should contain the definition of `mostFrequentWords` and any "helper" functions it needs. Finally, `hw4d.py` should contain a main program and all functions needed to solve the problem. These will include all the functions you wrote for Problems (a), (b), and (c).

Computational Text Analysis. For this homework, we have made available electronic versions of 3 famous novels that I downloaded from project Gutenberg. One of them is by Leo Tolstoy and the other two are by R. L. Stevenson. Your task for this homework is to write a program that analyzes the writing of the two authors by computing the 20 most frequent words used by each. The novels that we downloaded are:

- "The Strange Case of Dr. Jekyl and Mr. Hyde" by Robert Louis Stevenson,
- "War and Peace" by Leo Tolstoy,
- "Treasure Island" by Robert Louis Stevenson.

We have posted these as text files with names `hyde.txt`, `war.txt`, and `treasure.txt` respectively.

To get started, let us agree on what a *word* means. A *word* is a contiguous sequence of letters (lower or upper case) such that the character just before (if one exists) is a non-letter and the character just after is also a non-letter. This definition might lead to strange words occasionally. For example, the contraction "`we've`" results in two words "`we`" and "`ve`." This is okay and your program does not have to do anything to fix this.

To solve the overall problem we want you to write a series of functions. Note that in order to create these functions, you might feel the need to write additional "helper" functions. This is fine to do and in fact we encourage you to do this!

(a) Write a function with function header

```
def parse(s):
```

that takes a string parameter `s` and returns a list containing all the words in `s` that are at least 4 letters long. The order in which words appear in the returned list does not matter. Also, it is okay for the list to contain multiple occurrences of the same word. Finally, it is required that all words appear in lower case implying that upper case letters need to be converted into corresponding lower case letters.

For example, if the string `s` is

```
"Lincoln's silly, flat and dishwatery utterances - Chicago Times, 1863"
```

then it is correct for the the function `parse(s)` to return

```
["lincoln", "silly", "flat", "dishwatery", "utterances", "chicago", "times"]
```

(b) Write a function with header

```
def computeWordFrequencies(filename):
```

that takes a string parameter called `filename` and reads from a file of that name and returns a list, say `L`, consisting of two lists. `L[0]` is required to be the list of all distinct, lowercase words of length at least 4 in the given file and `L[1]` is required to be the corresponding list of frequencies of these words. For example, if the function returns

```
[["hello", "this", "ball", "bombastic"], [3, 7, 1, 5]]
```

this means that in the file that was read by the function, the word "`hello`" appears 3 times, the word "`this`" appears 7 times, etc. It should be clear to you that the way to write this function is to repeatedly call the function `parse(s)` and do some additional processing of the lists of words returned by these calls.

(c) Write a function with header

```
def mostFrequentWords(wordList, frequencyList, k):
```

that takes three parameters: (i) a list of strings (words) called `wordList`, (ii) a list of positive integers called `frequencyList`, containing the frequencies of the words in `wordList`, and (iii) a positive integer `k`. You should assume that `wordList` and `frequencyList` have exactly the same length and furthermore the frequency of the word `wordList[i]` is given by `frequencyList[i]`. You may also assume that `k` is no larger than the length of `wordList`. The function should return a list consisting of the `k` most frequent words from `wordList`, with ties broken arbitrarily. The words in this list should appear in decreasing order of frequencies (again, with ties broken arbitrarily).

(d) Write a program that reads the three given novels and produces as output the 20 words most frequently used by Tolstoy and the 20 words most frequently used by R. L. Stevenson, based on the sample of their writings that we have analyzed. Note that since two of the novels are by R. L. Stevenson, your output for R. L. Stevenson should combine the analysis of the two novels. It should be clear to you that you will be solving this problem by combining the functions you have crated for the earlier problems and doing some extra work beyond that.