King Fahd University of Petroleum & Minerals
Information & Computer Science Department
ICS-202 Data Structures and Algorithms

Lab Project

Date Assigned	Due Date	Submission	Semester
29/10/2023	TBA	Through BB/Grade scope	231

Design and implement a dictionary data structure. Your dictionary holds a list of words (strings) to be used in a spell checker.

Your dictionary should support the following operations:

1. Initialization You should initialize a dictionary using-
   - a single string, [public Dictionary(String s)]
   - an empty dictionary [public Dictionary( )]
   - a text file having strings, each on a new line. [public Dictionary(File f)]

   For example, public Dictionary(String s) creates a dictionary with only 1 string *s*.

2. Add new word This method adds a new word (string) to the existing dictionary. The suggested signature is
   - public void addWord(String s) throws WordAlreadyExistsException

3. Search for word This method searches for a word (string) in the existing dictionary. The suggested signature is
   - public boolean findWord(String s)

4. Remove word This method removes a word (string) from the existing dictionary. The suggested signature is
   - public void deleteWord(String s) throws WordNotFoundException

5. Search for similar words This method searches for words that are similar to a given word s. By similar, we mean that the string s differs in exactly 1 letter only with the words in the dictionary. The suggested signature is
   - public String[ ] findSimilar (String s)

   For example, words similar to “puinter” are “printer”, “painter”, “pointer”, and “punter” etc. Note that two similar words may differ in their length by 1 letter.

Notes:

1. You may use any data structure covered in the course. However, you have to only use the classes that we have been covered in the course (e.g. MyContainer, MyLinkedList, etc.)
2. *No built-in classes may be used for the Dictionary data structure. In particular do not use ArrayList<String> or any other built-in data structure from java.util.*.*
3. An additional text file containing words is being provided as an initialization file.
4. Include a main method in your class to test all methods in the dictionary. The following operations should be supported in the testing.

   - Load the dictionary from a file:
     Enter filename> mydictionary.txt
     dictionary loaded successfully.

   - find a word in the dictionary. For example:
     check word> puinter
     word not found.

   - add words to the dictionary. For example:
     add new word> punter
     word added successfully.

   - Remove words to the dictionary. For example:
     remove word> puinter
     Exception: Word not found.

   - search for similar words to a word in the dictionary. For example:
     search for similar words> puinter
     painter, pointer, printer, punter.

   - save the updated dictionary as a text file. For example:
     Save Updated Dictionary (Y/N)> Y
     Enter filename> mydictionary2.txt
     Dictionary saved successfully.

5. Include a text file “efficiency.txt” with your program giving the time complexity of each of your methods in the dictionary as a function of *n* where *n* is the number of words in the dictionary.

6. **Submission details**: Submit the ***java file(s), the text file, project report in pdf*** through BB before the due date as a .rar file. You should Write your filename as **200######0.rar** where 200######0 is your unique 10-digit id.