Goal

The goal of this assignment is to implement a class that serves as a spellchecker for a text application (for example, a word processor, or something similar).

In this assignment you need to create a project and required class(es) yourself. However, your project and the class(es) must strictly fulfill the specification below.

The “spelling” project

The project you create must be called “spelling”. Do not call it anything else.

The SpellChecker class

You must create a class called **SpellChecker**. It must have exactly this name; this is important.

To work on this assignment you have a dictionary file that lists the words in the English language (or at least an arbitrary subset) and the source code for another class called **DictReader**. This class can read in the dictionary and provide it as an ArrayList of Strings. More about that below.

Base tasks (level 1)

Your SpellChecker can then obtain an English dictionary from a DictReader. The SpellChecker must have the following public methods (words are case sensitive, where not specified otherwise):

public int numberOfWords()

This method returns the number of words in the dictionary.

public boolean isKnownWord(String word)

This method returns true, if (and only if) the given word is found in the dictionary.

public boolean allKnown(ArrayList<String> words)

This method returns true if (and only if) all words in the given list are found in the dictionary.

Base tasks (level 2)

public ArrayList<String> wordsStartingWith(String prefix)

This method returns a list of all words from the dictionary that start with the given prefix.

public ArrayList<String> wordsContaining(String text)

This method returns a list of all words from the dictionary that include the given substring.

Make sure that your *wordsStartingWith* and *wordsContaining* methods are case-insensitive. That means, for instance, that looking up words starting with “gilb” should find “Gilbert”.

Base tasks (level 3)

public void insert(String newWord)

Insert the given word into the dictionary. The word should only be inserted if it does not already exist in the dictionary. If it does, the method does nothing. Make sure that the alphabetic order of the dictionary is maintained.

public boolean remove(String newWord)

Remove the given word from the dictionary. If the word was successfully removed, return true. If not (for example it did not exist) return false.

public void save()

Save the dictionary to disk. This is not meant to be hard – there is a method in the DictReader class that you can use. It is listed here only because it goes together with *insert* and *remove*.

Challenge tasks

public boolean isPalindrome(String word)

Return true if (and only if) the given word is a palindrome. A palindrome is a word that reads the same when read backwards. Palindromes are treated as case-insensitive. For example “racecar” is a palindrome. For a word to be a palindrome, it also must exist in the dictionary.

public ArrayList<String> anagrams(String word)

Return a list of all words that are anagrams of the given word. Anagrams are treated as case-insensitive.

public ArrayList<String> difference(ArrayList<String> dictionary)

Given another dictionary as a parameter, compare the given dictionary to this one. Return a list of words that are in one of the dictionaries, but not in the other.

How to approach the project

The base task has been divided into three levels. This is intended to give you an order of attack: We suggest that you complete the methods in each level before you move on to the methods in the next level. We would rather see submissions that implement some methods well, rather than submissions that do all methods badly.

Support material

In the assignment 1 folder you will find the following:

* A file called **words.txt**. This is a text file that lists the dictionary words (one word per line). Place a copy of this file into your project folder for the spell checker to use.
* A Java file called **DictReader.java**. This is the source code for a class that you may use in your project. It reads the dictionary file from disk and provides it as an ArrayList<String>. Add this class to your project by using the *Edit–Add Class From File* function in BlueJ.

**Submission and deadline**

* Only SpellChecker.java file (write your name and roll number in first line as comment)
* August 16, 2023. 11:59 PM

**Marking**

The submission will be marked for

* Completeness (have all required methods been implemented?)
* Correctness (does everything work as specified?)
* Documentation/commenting (is everything commented as it should?)
* Style (is the code nicely laid out and formatted, are the variable names chosen well?)
* Difficulty (are any of the challenge tasks included?)

A submission with a perfect implementation of the base tasks will receive a good mark. To get a very good mark, you need to implement challenge task as well. Do the base tasks first – they attract more marks than the challenge tasks.

**Honor code**

* **My solution to assignment will be my own work.**
* **I will not make solutions to assignment available to anyone else.**
* **I will not engage in any other activities that will dishonestly improve my results or dishonestly improve/hurt the results of others.**

**Acknowledgement**

The tasks in this assignment have been taken from University of Kent.