Spellchecking

Prerequisites, Goals, and Outcomes

Prerequisites: Students should have mastered the following prerequisite skills.

- Hash Tables - Understanding of the concept of a hash function
- Strings - Basic string handling skills
- Inheritance - Enhancing an existing data structure through specialization

Goals: This assignment is designed to reinforce the student's understanding of the use of hash tables as searchable containers.

Outcomes: Students successfully completing this assignment would master the following outcomes.
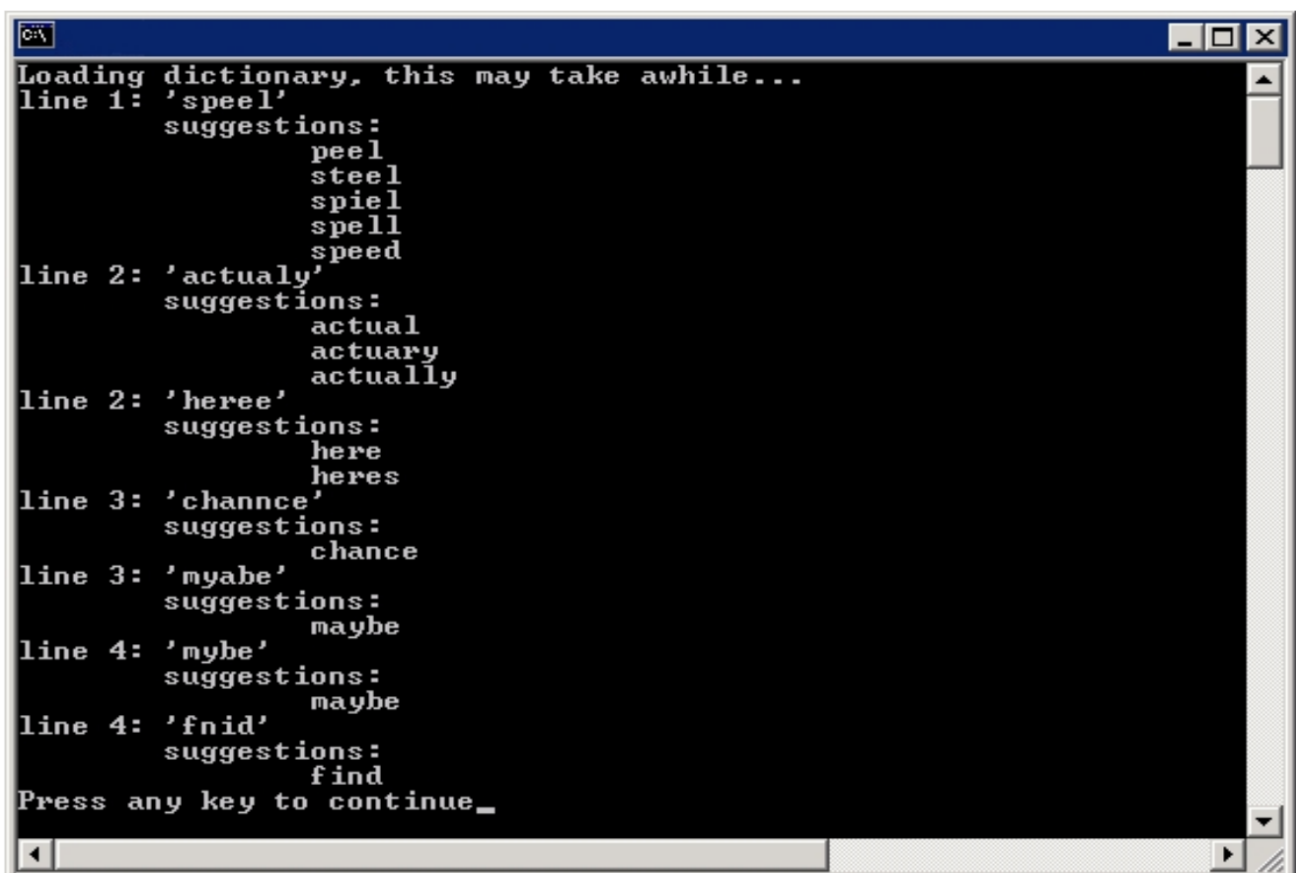
- Understand how to use hash tables, specifically hash sets

Background

Any word processing application will typically contain a spell check feature. Not only does this feature point out potentially misspelled words; it also suggests possible corrections.

Description

The program to be completed for this assessment is a spell checker. Below is a screen shot of the program in execution. The program begins by opening a word list text file, specified by a command line parameter. The program outputs an error message and terminates if it cannot open the specified word list text file. A sample word list text file (wordlist.txt) is given in the supplied spellchecker.zip archive. After successfully opening the specified word list text file, the program then stores each word into a hash table.

The program then opens a file to spell check. This user specifies this file through the command line. After opening this file, the program then compares each word in the file against the words stored in the hash table. The program considers a word to be misspelled if the word does not exist in the hash table. When this occurs, the program displays the line number the word appeared in, the word, and a list of possible corrections.

The list of possible corrections for a misspelled word is generated using a simple algorithm. Any variation of a misspelled word that is itself a word (i.e. it is found in the word list file) is a possible correction. Your solution to this assessment should consider the following variations of a misspelled word.

- *Transposing of adjacent letters*
  For the misspelled word "acr", transposing adjacent letters yields the possible corrections of "car" and "arc".
- *Removal of each letter*
  For example, removing each letter from the misspelled word "boaot" yields only the possible correction of "boat". Removing letters other than the second "o" does not generate a correctly spelled word.
- *Replacement of each letter*
  For each character in a misspelled word, the program should check if the replacement with any letter generates a correctly spelled word. For the misspelled word "acr", replacing the "c" with an "i" yields "air", replacing the "r" with an "e" yields "ace", and so on.
- *Inserting any letter at any position in a word*
  The program should consider if inserting any letter at any position in a misspelled word generates a correctly spelled word. For the misspelled word "acr", inserting an "e" after the "r" yields "acre".

Files

Following is a list of files needed to complete this assessment.

spellchecker.zip contains all of the following necessary files:
- main.cpp - This file contains the main routine.
- hashset.h - This declares a hash set class.
- dictionary.h - This file contains the partial definition of class Dictionary .

Class Dictionary inherits from class HashSet .
- wordlist.txt - This file is a word list text file.
- test.txt - This is a sample text file that contains spelling errors.

Tasks

To complete this assessment, you need to complete the implementation of class Dictionary and complete the spell checking program contained in main.cpp .

To begin, verify the files needed for this assessment.

      1. **Extract** the archive to retrieve the files needed to complete this assessment.

Following is an ordered list of steps that serves as a guide to completing this assessment. Work and test incrementally. Save often.

1. **Begin** by completing the definition of class Dictionary. Class Dictionary must provide a constructor that accepts a single string as a parameter. This parameter is the file name of the word list text file. If file unavailable throw an *invalid_argument* exception. This constructor must place all the words contained in the text file into the dictionary. Remember, class Dictionary is a type of HashSet , so use the inherited methods accordingly.

2. **Next**, complete the hash function encapsulated in class HashFunction in dictionary.h . Don't use *std::hash*.

3. **Then**, finish the implementation of methods HashSet::insert() and HashSet::remove().

4. **Finally**, finish the implementation of function CheckSpelling. This function already contains code that reads a file line by line. It also extracts each word from a line using an instance of class stringstream . Your task is to check the spelling of each word. Use the inherited search function of class Dictionary to determine if a word exists in the dictionary. If the word exists in the dictionary, assume that it is spelled correctly. It if does not exist, assume it is misspelled. For each misspelled word, generate and display a list of possible corrections.

Submission

Submit **only** the following.

1. dictionary.h - your completed class Dictionary definition
2. dictionary.cpp - if created
3. hashset.h – is changed
4. main.cpp - your completed spell checker program