CSCI 2380 **Fall 2024** **Mr. Gustavo Dietrich**

Homework 10: Binary Search **Due Date:** 11/22/24

You are a software engineer at Pear, a mobile phone company. You've been asked to implement the "autocomplete" feature of PearOS. Autocomplete suggests how a partially typed word might be completed into a word from a list, e.g. a dictionary (see Figure 1).
Figure 1: Autocomplete suggests "afternoon" and "after" as completions of "aft".
Because the data might depend upon the user and what words they use most commonly, adding words to the list must be permitted. To keep up with your users' typing, your code must give suggestions extremely fast (Θ(log n) time). However, changing the list via adding words is allowed to be slow (Θ(n) time). A dynamic array of sorted words that uses binary search to find suggestions would work.

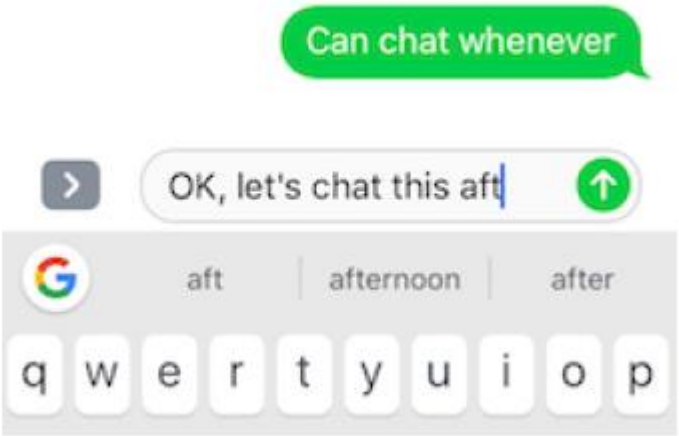

Figure 1: Autocomplete suggests "afternoon" and "after" as completions of "aft".

The following files have been given to you:

1. A C++ header file (autocompleter.h) declaring the **Autocompleter** class.
2. A C++ source file (main.cpp) containing a main() function with tests.
3. A text file (words.txt) containing 10000 common words[1].

Create new C++ source file named **autocompleter.cpp** that implements the function declared in autocompleter.h so that autocompleter.cpp and the provided files compile into a program that runs with no failed tests.

Hint:
The Autocompleter data structure uses a sorted dynamic array A of strings. Implementing completion_count() and completions() efficiently requires quickly locating where strings starting with x occur in A.

[1] Source: http://norvig.com/ngrams/count_1w.txt

The leftmost string starting with x occurs at the location where x would be if it were contained in A (i.e., the location returned by index_of(x, A, len)). Also, all of the strings starting with x occur consecutively, starting with the leftmost string at index_of(x, A, len).

Submit just the source code of **autocompleter.cpp**. You don't need to submit the main.cpp nor the other files because I will use my own autocompleter.h and main.cpp files to evaluate your autocompleter.cpp file.

Review the examples discussed in class and the textbook to get an idea of what you need to do. Carefully analyze the tests because that will help you understand how the functions that you need to create work. Use hw10_interactive.exe to understand how the insertion of a word works (it uses words10.txt). Use the hw10.exe to get an approximation of how fast your solution should work (it uses words.txt).

IMPORTANT:

Make sure your program compiles and executes in full (it should pass all the tests included in main()).
Your solution must be a generic one, do not hard code it based on the tests just to pass them. You will receive no credit for hardcoded solutions.
You must submit ONLY ONE solution per team.
Your program must be well commented, use meaningful identifiers, and use indentation to improve its readability.

Your program must have the following comments at the top:

```
//*********************************************************************
// Team #          CSCI 2380          Fall 2024          Homework # 10
// First and Last Name
// First and Last Name
//
//*********************************************************************
```

When done, submit your solution through Blackboard using the "Assignments" tool. Do Not email it.
Paste the link to your final solution along with your source code in the textbox opened when you click on Create Submission before you click on Submit.

Sample run of the interactive program

```
C:\WINDOWS\system32\cmd.exe

Please enter i to insert or c to check a word: c

Please enter the word to be checked: a
    a
    aa
    aaa
    aaron
    ab

Please enter i to insert or c to check a word: i

Enter the word to be inserted: aasum

Please enter i to insert or c to check a word: c

Please enter the word to be checked: a
    a
    aa
    aaa
    aaron
    aasum

Please enter i to insert or c to check a word: i

Enter the word to be inserted: aab

Please enter i to insert or c to check a word: c

Please enter the word to be checked: a
    a
    aa
    aaa
    aab
    aaron

Please enter i to insert or c to check a word: c

Please enter the word to be checked: aa
    aa
    aaa
    aab
    aaron
    aasum

Please enter i to insert or c to check a word:
```

The following is the basic criteria to be used to grade your submission:

You start with 100 points and then lose points as you don't do something that is required.

-20 : Incorrect implementation of index_of()
-20 : Incorrect implementation of Autocompleter()
-20 : Incorrect implementation of insert()
-20 : Incorrect implementation of size()
-20 : Incorrect implementation of completion_count()

-20 : Incorrect implementation of completions()
-20 : Program crashes when executed
-5 : Unnecessary statements in your code
-40 : Program does not compile
-10 : Missing/too few comments
-20 : Incorrect/missing source code
-20 : Incorrect/missing link to your solution
-100 : No team contribution
-100: The code submitted is not your creation (you got it from a web site or another person)
-10 : Late
-100 : No submission.

Important: more points may be lost for other reasons not specified here.