Santa Monica College
CS 20A, Fall 2014

Programming Project 5 - Due: **Wednesday**, December 17 at **6:45 PM**

Your boss has grown an interest in word puzzles and would like you to write a program that, given a word, finds other valid English words that have exactly the same letters. She hands you a file containing an English dictionary.

You find this request a little strange, but the problem seems interesting and you agree to work on it.

Part 1:

On first thought, you plan to make a set of all the words in the dictionary (all converted to lowercase). Then, given a word, you plan to make "words" from all the combinations of the letters and check whether each is in the set. If yes, it is a valid English word and you add it to the result. For example, for the word "cat", the combinations are "act", "atc", "cat", "cta", "tac", "tca". "act" is in the dictionary and thus valid.

Analyze the runtime of this algorithm and give the big O. Do you think this algorithm will be fast enough and should you try to implement it? Write your brief and clear answer (**5-10 lines max**) as a comment in the .cpp file under the header "Part 1 Analysis:".

Part 2:

After some thinking you realize that if you could somehow group all the dictionary words having the same letters into separate groups, then you could find all the matching words for a given word by only searching through that word's group. This approach reminds you very much of the way a hash table implemented using separate chaining is implemented.

The only library data structures you can use are arrays and linked lists.

Hints:

- Make sure you understand the problem and your approach really well before you write even one line of code. Start by looking at the given methods and the given test code. The algorithm and code you will write is not complicated, but can get complicated really fast if you are not clear about what you are doing. Make sure you understand the concept of hashing and how separate chaining was implemented.
- A common problem when performing a mod operation (e.g. h % size) is when h is negative. `abs(h) % size` handles the problem correctly.
- For Part 2, we will be using an array of linked lists. We will use STL linked list, so you can focus on the problem as opposed to fixing the bugs from linking nodes through pointers (we already did that in project 2). Have a look at the documentation and the examples below. You will only need to use a handful of methods like adding to the list and iterating over it.
  http://www.cplusplus.com/reference/list/list/push_back/
  http://www.cplusplus.com/reference/list/list/begin/
- Test code has been given to you. Start first with some simple tests and, as you gain confidence that your code works, run all the functional tests and then the stress tests to ensure your solution is efficient.

Deliverables:

You should submit a zip file named project5_first_last.zip (where first and last are your first and last name) containing **ONLY the file below.**

CS20AProj5.cpp

How you get points:

Part 1	15 points
Part 2	85 points

How you lose points:

- You change the given public methods' signatures and your code does not compile when I run it. **You should only fill in the empty public methods and add any private methods you will need.**
- If your solution is inefficient. Your solution should be **efficient to a level seen in class for similar problems**.
- You submit your whole workspace or executable files. **Submit only the files the project asks for.**
- If any of your code prints anything at all on the console except for the Part 2 output. **Remove all your print outs, debug statements, etc. Clean up your code and do not leave clutter behind.**
- Your code has no comments where needed. **Comment your code appropriately.**