Programming Assignment #2
CS253, Fall 2017
Due: Tuesday, Sept. 12th

"Counting Words"

Motivation

Last week you wrote your first C++ program. This week you will write a slightly more complicated program that doesn't directly build on the first, but is very close in spirit. Last week you counted numbers; this week you will count unique "words", where a word is any string of characters with no internal whitespace[1]. The biggest difference is that last week there were only 20 possible numbers. This week there is an unlimited number of possible words.

This is the first step in a larger project that is going to analyze text. Some of the analysis will be at the level of individual words, some of it will be at the level of sequences of words (e.g. syntax), some will be at the level of word frequencies, and some will be at the level of documents. To support these future analyses, I recommend that you write your program to read through the input file, producing a vector of strings containing all the "words" in the order they appear. The strings in this vector will support word-level analysis, while the vector will support sequence analysis. Your program should then sort the vector lexically, making it easy to extract unique vectors and counts of how often each unique vector appears.

Task

Your program will take a single file name as an argument. The text file will contain words separated by whitespace, where a word is any sequence of characters that does not contain whitespace. Your program will count how many times each unique word appears. It should write one line of output (to std::cout) for every unique word that appears in the file. Each line of output should contain the word followed by a space and then the number of times the word appeared. The words should appear in lexicographic order.

For example, if the input file contained "This is a test, a test it is!" (quotes not included), then the output should be:

```
This 1
a 2
is 1
is! 1
it 1
test 1
test, 1
```

[1] A whitespace character is defined as space, tab, newline, vertical tab, feed or carriage return.

Submitting Your Work

To submit your program, first create a single tar file containing all of your source files (i.e. your .cpp and .h files) and your makefile, but not your compiled files (no executable or .o files, please). Then submit the tar file as PA2 on Canvas.

Grading Your Homework

As always, to grade your assignment, the GTAs will unpack your archive in an empty directory. They will compile your code by typing 'make'. This should make your code from scratch. If your code does not compile using make, you will receive a zero for the assignment. Compiling your code should not produce any warning messages. Although we will not deduct points for warning messages, we will in future assignments so get in the habit of making sure you code compiles cleanly. Assuming your program compiles, it will be graded by how it performs on test cases, including test cases with errors in the input file. If the input file contains errors, you will receive full credit if your program returns -1 to main. (In case of an error, it should also print a meaningful error message to std::cerr.) If the input does not contain errors, your program should return 0 to main.

Hints

- Never trust a file or a user. There are fewer possible error cases this week, but there are some!
- Lexicographic order is the order used by the compare method of the string class.
- There are many useful pre-defined character functions in C++. Isspace() is one of them. They may be useful in the current assignment, they definitely will be useful in future assignments.
- Note that we are not grading on efficiency (yet). Nonetheless, your program should run in O(n log(n)) time (assuming you use an O(n log(n)) sort algorithm).
- The code you write for PA2 will be a part of PA10. You will not remember in December what you were thinking when coding in August. Document your code! Clean and documented code is self-rewarding.

Policies

All work you submit must be your own. You may not submit code written by a peer, a former student, or anyone else. You may not copy or buy code from the web. The department academic integrity policies apply.

You may not submit your program late. To receive credit, it must be submitted on Tuesday, Sept. 6th. There is no late period. The exception is an unforeseeable emergency, for example a medical crisis or a death in the immediate family. If an unforeseeable emergency arises, talk to the instructor.