Programming Assignment #1
CS253, Fall 2017
Due: Tuesday, Sept. 5th

"*Welcome to C++: Counting*"

Motivation

The primary goal of this assignment is to get you started writing C++. In particular, as Java programmers there are some initial hurdles to writing in C++: breaking code into .h and .cpp files, inserting the right include statements, invoking the compiler, interpreting the arguments to main, differences in I/O, etc. This assignment gives you a task whose logic is pretty simple, in order to give you a chance to get over those hurdles while still writing pieces of code that might help you in future assignments.

A secondary goal is to give you a gentle introduction to the more demanding expectations in CS253. For starters, you should notice that the assignment specifies *what* to do, not *how* to do it. We will not tell you what classes to define or what methods they should have (although we may give hints). You are expected to design your own programs from scratch. Program design is an important part of the assignment. Also, there may be times when part of the assignment specification is ambiguous. In such cases, it is incumbent on *you* to ask the instructor for clarification. The instructor is playing the role of client in these exercises, and what the client intends is correct. If you are not sure of exactly what the client wants, you need to find out. "But I thought it was supposed to …" is not an excuse.

Along the same lines, testing your program is your responsibility. *We will not hand out test files*. Nor is it sufficient to handle legal input files; when given an illegal file, you must print an error statement and return -1. To succeed, you need to come up your own test files and test your code. Indeed, designing test cases is an important part of the assignment. Testing for errors is your responsibility. The assignment below is to count numbers. But here is a hint: never trust a user or a file. What if the file has letters in it? Or punctuation? Or nothing at all? It is part of the assignment to test for error cases and handle them cleanly.

Task

Your program will take a single file name as an argument. The text file should contain 1 or more values, where every value is an integer between 0 and 19. The integers should be separated by white space (e.g. spaces, tabs, or newlines). Your program should write 20 lines of output to the terminal (more specifically, to std::cout). The first line should have the number zero, a space, and then the number of zeros that appear in the input file. The second line should have the number one, a space, and then the number of ones that appear in the input file. This continues, so that the last (twentieth) line contains the number nineteen, a space, and the number of 19's in the input file.

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 PA1 on Canvas.

Grading Your Homework

To grade your assignment, the GTAs will unpack your archive in an empty directory using the tar command. 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 may 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. When the input file does not contain errors, you will receive full credit for a test if (1) all 20 counts written are correct, (2) the output is in the correct format and written to std::cout, and (3) your program returns the value 0 from main. 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.)

Hints

- Never trust a file or a user. You can be sure that we will include test cases with illegally formatted input, illegal arguments, etc.
- Test your program on the department machines in CSB120. That is where we will evaluate it. Your grade depends on how your program performs on those machines. (Warning: C++ is not as portable across platforms as Java.)
- Test your program by unpacking it in a clean directory and running it. After all, that is what we will do.
- Be careful when you make your tar file. Too many students have lost points because they either overwrote code when trying to make their tar file or forgot to include source and/or make files.

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. 5th. 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.