Programming Assignment #1
CS253, Fall 2014
Due: Tuesday, Sept. 6th

"*Histograms*"

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 a piece of code that will 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 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 do this, you need to come up your own test files, and test your code. Indeed, designing test cases is an important part of program design. Testing for errors is your responsibility. The assignment below is to compute a histogram. 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 this and all following assignments 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 a value is an integer between 0 and 255. Your program will write 64 numbers (separated by white space) to std::cout. The first value written to std::cout is the number of input values in the range [0, 3] (i.e. 0 to 3 inclusive). The second output value is the number of input values in the range [4,7]. In general, the *n*th output value is the number of input values in the range [4×(n-1), (4×n)-1]. Equivalently, input value X is added to output count number Floor(X/4).

The format of the input file is flexible. It may contain many numbers on one line, or many lines with one (or a few) numbers each. It may contain blank lines. However, it should only contain integers between 0 and 255 (inclusive) and white space. If the file contains numbers outside of this range, or anything other than integer numbers, your program should write out an error message to std::cerr and return the value -1 to the operating system. Otherwise, your program should write 64 values to st::cout.

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) 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 and compile it using "g++ -I. –Wall *.cpp –o exec1". If your program does not compile, you will be awarded no points for the assignment. If the compiler produces any warning messages, points will be deducted (up to 10% of the total, depending on the number and severity of the messages). Assuming your program compiles, the remaining 80 points will be determined by how your program 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 all 64 values written to std::cout are correct. If the input file does contain errors, you will receive full credit if your program produces a meaningful error message and returns -1.

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.
- Notice the "-Wall" compiler flag above. It tells the compiler to generate all possible warnings. I suggest using this flag yourself, to avoid getting penalized for warnings you never saw.
- 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.)

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.