CMP_SC 3050 Homework 1
Due: 11:59:59 pm, 9/8/2016

The first homework of the semester is a programming assignment. The homework is out of 30 points and will contribute to 3% of your total grade. The first part of the homework specifies the exact problems that your submission should solve and the second part describes the constraints that you **must** follow (no exceptions). Absolutely no late work will be accepted.

Specification:

For this programming assignment, one compressed file containing the directory of all of the assignment files should be submitted to Blackboard prior to the due date. Your assignment can be built with a standard Makefile, or as a Netbeans Project. Your program should compile on a UNIX machine.

The assignment should:

- Take one filename as input in the command line. The files is going to be the input to the program.

- Each input file represents a list of non-negative integers. Each line of an input file consists of a non-negative integer followed by a newline character. Each integer is a string of digits. One sample input file can be found on Blackboard.

- Your program should output on the standard output the number that occurs with the second highest frequency in the input followed by a new line character. If there are more than one such number, your program should output the highest one. If all numbers in the input file have the same frequency then output 0.

  For example, if the input file contains the integers 1, 2, 10, 1, 1, 2, 3, 2 then frequency of 1 in the input file is 3, frequency of 2 is also 3 and frequencies of 10 and 3 is 1 each. Your program should output 10\n in this case.

- It is important that the standard output should itself consist of exactly one number. It is absolutely essential that you do not put any whitespaces or any other character in that line (except, of course, the end of line).

Constraints:

The following is a list of constraints for this assignment; failure to adhere to any of these constrains will result in a loss of points or even a zero.

- The assignment can be completed in C or C++.

- Built-in data structures and external libraries other than standard input/output functions may not be used for this assignment. If the program requires a stack, linked list, or any other structure, it is up to you to provide it. If you have any doubt on whether or not any technique you wish to use is acceptable, do not hesitate to ask.

- A moderate amount of error checking and resource management is required. Even if you do some error recovery, you must report errors in the input files. Your application should ensure that each line from the input file is properly formatted, that the file is successfully opened, the file is successfully closed upon reading of the file and that all allocated space is de-allocated at the exit.

- For reporting input format error, please use the enum type provided in input_error.h. Please do not alter the file input_error.h. You need to include input_error.h in your main program. The file input_error.h contains documentation of how to use input_error.h. Please do not not report anything else on the standard output.

- A moderate amount of formatting and documentation is required. Comments should be descriptive and used to illustrate the purpose and inner workings of an algorithm or function; they should not be used to annotate each line or self-evident logic.

- The assignment must be submitted via Blackboard. Please compress the folder before submitting using zip.

- The input size is unknown, and the input range is 0 to UINT_MAX. This means that your program should not make any assumptions about the range or size of the input.

- You have to write your own test-cases. We will check your program against our own test-cases . To receive any credit, your program must pass at least one of the tests. For full credit, you would have to pass all of those test-cases.

Timing Constraints

A good algorithm for this homework should run in O(n log n) where n is the number of integers in the input file. While we shall not be grading the exact time complexity of your solution, we shall be timing your solution and grading efficiency. As a guideline, on an input files each consisting of million integers, your program should run under a minute of system time to receive full credit for efficiency.

Grading:

There are30 points possible for this assignment. The grade breakdown is as follows:

- 4 points for error checking and resource management.
- 4 points for general programming style and adherence to the constraints.
- 18 points for correctly outputs.
- 4 points for efficiency.

If the program fails to compile or crashes due to a runtime exception, a grade of zero will be assigned.