Introduction to Programming (COP 3223) Assignment #7
File Input: Programming Contest problems.

Due date: *Please consult WebCourses*

Objectives

1. Learn how to read input from files and write output to files.
2. Review the use of if statements and loops **in C.**

Programming Contests: Background

In the course textbook, a brief history of programming contests is covered. UCF hosts a programming contest for high school students. In order to test students' programs on many possible input cases, we ask students to write programs that read in their data from a file (without prompting for the name of the file or for any of the information). Students must then read in the input from the file, calculate a solution to each case of input in the file and output the results in the exact format requested to the screen.

Your Assignment

For this assignment you will solve two of the easier problems we have posed in our high school contests. Since I want you getting practice writing to an output file as well, this assignment will consist of two programs. For the first program, Chalice, you will just write your output to the screen. **For your second program, Ice, please write your output to a file called out.txt. For both programs, please read your input from the files chalice.in and ice.in, respectively.**

Problem Descriptions

The descriptions of both problems are separate .pdf files: Chalice.pdf and Ice.pdf, exactly as they appeared in their original contest. Here is a summary of each section of the description:

The Problem – This states what problem you will be solving.
The Input – This states the exact format of the input file.
The Output – This states the exact format you must provide the output.
Sample Input – A sample input which is NOT comprehensive, but shows an example of a valid input file that adheres to the specification in "The Input"
Sample Output – The output your program should produce corresponding to the Sample Input.

Deliverables

Two source files:

1) *chalice.c*, for your solution to Problem A (Chalice.pdf),
   which reads from chalice.in and which writes to the screen.
2) *ice.c*, for your solution to Problem B (Ice.pdf),
   which reads from ince.in and which writes to the file out.txt.

All files are to be submitted over WebCourses.

Restrictions

Although you may use other compilers, your program must compile and run using Code::Blocks with gcc. Each of your three programs should include a header comment with the following information: your name, course number, section number, assignment title, and date. Also, make sure you include comments throughout your code describing the major steps in solving the problem.

Grading Details

Your programs will be graded upon the following criteria:

1) Your correctness
2) Your programming style and use of white space. Even if you have a plan and your program works perfectly, if your programming style is poor or your use of white space is poor, you could get 10% or 15% deducted from your grade.
3) Compatibility to Code::Blocks (in Windows). If your program does not compile in this environment, **the maximum credit you will receive is 50%.**

Final Note

I will be nice this time and provide our real testing data for the problem Ice, the harder of the two problems. My hope is that by providing you this data, you will expedite the debugging process by more quickly determining cases you hadn't thought of.