CSCE 2004 - Homework 1
Due Date - 09/19/2015 at 11:59 PM

0. Overview:

In this assignment, you will develop a program to handle university degree auditing and course management. That is, after taking a set of courses, a student would like to know if she/he has met the graduation requirements for a particular degree (or what the student is missing to meet those requirements). The requirements include required courses, total credit hours, minimum GPA, etc. As a college student, can you see the value of such a tool in the real world?

To build any software system, one must be equipped with the following: a firm grasp of the problem to be solved, the ability of dividing the problem into smaller problems, devising related algorithms to solve the smaller problems and combining their solutions, the ability to translate the high level design into a computer program in a particular programming language (in our case C++), and the ability to debug and test the program.

All of our projects in this course are related to this core problem. So before you begin, you may want to review your own degree requirements to make sure you really understand them. If we are not able to understand a problem, then no matter how good we are as programmers or software engineers, we will not be able to solve that problem.

Achieving the minimum required GPA is one of our fundamental requirements. Therefore, our first project is to compute a student's GPA (or Grade Point Average). Please make sure that you know the concept and how to calculate it by hand.

1. Problem Statement:

The goal of this programming assignment is to give students experience performing numerical calculations and using input/output in C++. Your task is to write a program that will read a series of class grades and calculate the GPA for those classes.

To keep things simple, your program only has to read **five** (5) class grades from the user, along with the corresponding credit hours for each course. The grades are A, B, C, D and F, and hours are whole numbers between 1 and 5. For example, the user could type in the following:

```
A 3
B 4
C 2
A 1
D 4
```

To calculate the GPA for these classes, we need to add up the total number of credit hours (14) and the grade points weighted by credit hour (36) and then calculate the GPA = 36/14 = 2.57

At this early stage in the course, we have not covered any C++ tools or techniques for error checking. All you need to do is write a program that calculates the correct GPA if the user follows instructions and types in the grades and credit hours correctly. If the user types in something silly, your program will probably output a silly GPA or go crazy, and this is OK.

The goal of this assignment is for students to get experience performing numerical calculations and using input/output in C++. **You are NOT required (or allowed) to look ahead to more advanced C++ features like arrays or loops to perform this task.**

2. Design:

For this assignment, the first design task is to decide how to read input data from the user. Do you want to read in the data above in column order as "A B C A 3 4 2 1" or using row order "A 3 B 4 C 2 A 1". Do you want to read the number of credit hours before or after the class grades?

To simply the user interface design task for you and to have a uniform look and feel across students, your program will need to use the same “user interface” as the given executable file. In other words, when your program is executed, it should behave just like the given executable file, except for possible rewording of the instructions to the user. The next task is to decide on how to implement the GPA calculation. What variables and formulas are needed? The final design task is to decide how to present the results to the user. What messages should you print? How should these be formatted? Make sure you write comments in your code to explain your design decisions.

3. Implementation:

Since you are starting with a "blank piece of paper" to implement this project, it is very important to develop your code **incrementally** writing comments, adding code, compiling, debugging, a little bit at a time. This way, you always have a program that "does something" even if it is not complete.

4. Testing:

Test your program to check that it operates correctly for all of the requirements listed above. You may use the given executable file to help you. Also check for the error handling capabilities of the code. Try your program with several input values, and save your testing output in text files for inclusion in your project report.

5. Documentation:

When you have completed your C++ program, write a short report (less than one page long) describing what the objectives were, what you did, and the status of the program. Does it work properly for all test cases? Are there any known problems? Save this report in a separate text file to be submitted electronically. Please use the **Project Documentation Template** provided on Moodle as the basis for your documentation.

6. Project Submission:

In this class, we will be using electronic project submission to make sure that all students hand their programming projects and labs on time, and to perform automatic analysis of all programs that are submitted. When you have completed the tasks above go to Blackboard to "upload" your documentation (a single **.pdf**, **.doc**, or **.docx** file), and your C++ program (a single .**cpp** file). Do **NOT** upload an executable version of your program or files in other extensions or format.

The dates on your electronic submission will be used to verify that you met the due date above. **No late projects will be accepted for credit.** Please refer to class syllabus about no late assignment policy as well as suggestions given regarding to submitting a partially working program.

You will receive partial credit for all programs that compile even if they do not meet all program requirements, so please hand projects in on time.

7. Food for thought:

How should we handle user “errors” in using our program? For example, typing a grade E or, more reasonably, typing I or W as shown in our transcript.

What are the changes required to our program so that it may calculate the GPA for any number of courses, not just 5 courses?