CSCE 2004 - Homework 6
Due Date - 11/21/2016 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 to divide 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.

Upon the completion of project five, we have a program to "manage" a course work of a student. The user may interactively use the program. The program is able to both store and retrieve a student's course work by saving to and loading from a file, respectively. For a degree audit, a student's course record is just one of two pieces needed for a complete system. The other part is a set of degree requirements for a particular degree program. Project six has all the functionalities of project five, plus the additional task of creating the degree requirement file for your personal degree program. To gain some basic experience of developing classes in C++, we shall develop a simple class, which combines the five attributes of a course (course taken) into a single object. Project six is an extension of project five. Here we have another chance to fix bugs of previous project. You should fix the bugs of project five first if any.

1.1 Understanding and representing degree requirements:

Degree requirements in general consist of University Core, Major Core, College Core, Total hour requirements, GPA requirements, and so on. To reduce the implementation effort and simplify the complexity of the project, we shall only consider the University Core requirements in our degree audit for this project and in project seven. Nonetheless, the representation should work in general.

We shall break down the University Core into sub categories such as English, Math, etc., based on the Undergraduate Catalog. Hence, each requirement may have two attributes, mainly for human understanding. They are called **requirement group** (such as University Core, BSCS Core, BSCE Core, and BACS Core) and **requirement subgroup** (such as English within the University Core or Required CSCE courses within in BSCS). In addition, each requirement must have a total credit hours needed for that requirement and a list of courses that can be used to for

earning the credit hours. We will use a text file to store the requirements. In project seven, our program will read in the requirements and store them in other data structures (classes or arrays). The text file consists of a sequence of requirements separated by a blank line. The first line of a requirement is the requirement group, the second the requirement subgroup, the third is credit hours needed for the requirement, and then each pair of lines that follow contains the course number and credit hours of a course that may be used to satisfy the requirement. The number of pairs depends on the number of courses that may be used. (Since we do not know how many, a blank line signals the end). The following is an incomplete requirement file that contains two requirements:
The following is the content of the requirement file with two requirements shown:

```
University Core
English
6
ENGL 1013
3
ENGL 1023
3

University Core
Fine Arts
3
ARCH 1003
3
ARHS 1003
3
COMM 1003
3
DANC 1003
3
ENGL 2023
3
HUMN 2114H
4
LARC 1003
3
MLIT 1003
3
MLIT 1013
3
THTR 1003
3
THTR 1013
3
```

One of your tasks for this homework is to set up a degree requirement file for your major. If your major is undecided or if you are in the graduate program or non degree seeking, then you may

use Bachelor of Science in Computer Science (BSCS) as your major. You may use the current year catalog or the handbook provided by your department for that information.

1.2 The "Course" class:

Upon the completion of project five, we have a program to "manage" a student **course work record**. A student course work is basically a list of courses taken by a student. For each course taken, we have the course name, when it was taken, course number, grade earned, and credit hours for the course. We currently use 5 parallel arrays to store the five pieces of information related to a course taken.

You now have the ability to create a user-defined type, or a **class,** to solve this problem. This will allow us to better group those pieces of information together. You will write a class named "**Course**" that has one member for each related piece of information (e.g. course name, course semester, course number, course grade, and credit hours). You will then convert your existing program so that it uses a single array of **Course**s instead of five parallel arrays containing the same information.

1. Problem Statement:

One goal of this assignment is to give students a chance to understand the requirements of their major and put the requirements into a requirement file. Remember that 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 the problem. The other goal of this programming assignment is to give students experience with **class** (a user define type), text file processing, writing and using functions, array data structures, if statements, and loops. A secondary goal is to give students a chance to modify an existing program to make it more useful. To meet these goals, students will be required to improve their previous course and course management program to make the program more modular, flexible and robust. Students will be required to implement the **Course** class EXACTLY as specified in the design section.

Enhanced features and tasks: The primary feature is to implement the **Course** class and make an array of **Courses** that holds the same information as the five parallel arrays used in project five. We will need to modify the project five source code to achieve that goal while maintaining the same functionality. The second task is to create and turn in your degree requirement file (see 0.1 above).

Students are allowed to use any "built-in" functions related to files and any combination of user defined classes, functions, array data structures, if statements, and loops that they need to complete this assignment. You are **not** required or allowed to look ahead to more advanced C++ features like pointers or vectors to perform this task. Students must develop all the classes or functions specified and may introduce additional classes or functions if they prefer (you have more freedom in your program design than before).

2. Design:

Prefix your major to “requirements.txt” as the file name for the requirement file. For example, if your major is BSCS, the file name is “BSCSrequirements.txt” See 0.1 above for a description of when your requirement file should be turned in, as well as the content and format of this text file.

The format of the requirement file has been designed for us. However, you might to want to think about how to hold the information inside your program. How many variables do we need? What are their types? Should we use array or class? The particular design decisions we make depend on how we want to use the information. We must answer these questions in project seven.

We have made the design of **Course** class for you as follows.

```
class Course
{
public:
    Course(); // default constructor

    void get(string& name, string& time, string& number,
        char& g, int& h) const;
    void set(string name, string time, string number,
        char g, int h);

    void print() const;

private:
   string courseName;
   string courseTime;
   string courseNumber;
   char grade;
   int hours;
};
```

This is a very simple class that five private members that we are all very familiar with by now. It has four public member functions (methods): a default constructor, a get, a set and a print.

The **default constructor** should initialize the five members as

```
Programming foundations I
Fall 2015
CSCE 2004
A
4
```

The **get** function allows the user to retrieve all five pieces of information of the object at the same time by using reference variables as arguments. (Note that this isn't a particularly good software engineering practice, since we do not have the ability to retrieve the value of a particular variable, but it does simplify the code a bit.)

The **set** function allows the user to assign all five attributes to the object at the same time. (Again, you would normally see a separate set function for each member, but then we would need 5 functions instead of 1.)

The **print** function outputs the values of the five members one per line to the screen.

We need to think about where we should insert some code in our main program so that our array of classes maintains the same information as the five parallel arrays. To demonstrate the correctness, you should write the information to another file called "checking.txt" at the termination of the program.

3. Implementation:

First, correct any errors from hw5. Then, develop one feature or function at a time and leave all other code the same and test it thoroughly before we move to the next feature or function.

We would suggest that you create a different file for implementing and testing the **Course** class. Since you do not have to implement all member functions at once, to avoid overwhelming compilation errors, you should implement **one member function at a time** and test each in isolation. A possible order could be to implement first the default constructor, then print, set, and finally get. Add an array of **Course**s once you are confident that the **Course** class is working and test it. After that, copy the code related the **Course** declaration and implementation to be beginning of project five code and the code becomes your project six code.

Add code to use the array of **Course**s so that it maintains the same information as that of the five parallel arrays. Leave the five parallel arrays, as well as all the code manipulating them, in your code so the program is always working and running.

Since you are starting with your previous program, you already have something that compiles and runs. Since you must add new features and maintain the current behavior of the program, it is important to make these changes **incrementally** one function at a time, writing comments, adding code, compiling, and debugging. 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. Also check for the error handling capabilities of the code. Try your program with several input values **[2-3 executions of your program]**, and save your testing output in text files for inclusion in your project report. You can include the testing at the bottom of your report. Be careful about edge cases (e.g. division by zero, a file cannot be opened, etc.).

5. Documentation:

When you have completed your C++ program, write a short report (less than one page long not counting program testing output) 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:

What is a suitable way to represent the degree requirement information inside our program?

How do we perform a degree check (or audit) provided we have suitable representations of both courses taken by a student and the degree requirement for the major for which the student is pursuing?