CS 161B: Programming and Problem Solving II

Assignment 6: Students and Course (Structs)

Academic Integrity

You may NOT, under any circumstances, begin a programming assignment by looking for completed code on StackOverflow or Chegg or any such website, which you can claim as your own. Please check out the [Student Code of Conduct at PCC.]

The only way to learn to code is to do it yourself. The assignments will be built from examples during the lectures, so ask for clarification during class if something seems confusing. If you start with code from another source and just change the variable names or other content to make it look original, you will receive a zero on the assignment.

I may ask you to explain your assignment verbally. If you cannot satisfactorily explain what your code does, and answer questions about why you wrote it in a particular way, then you should also expect a zero.

For this assignment, you will be working with some starter files and modifying and adding functions to add more features to the given program. This is not a straightforward assignment - so I suggest you get started early and you have 2 weeks to complete it.

Purpose

In this assignment, you will start with the given files and add or modify functions to complete the program. Please make sure to complete zyLabs 14.9 and 14.10 which will help you do this assignment successfully.

After completing this assignment you will be able to:

- Create a user-defined data type called a struct
- Use the struct in another struct (nested structs)
- Read from a file into the array of structs
- Manipulate data in the array
- Use of multiple files

Task

- ❏ **Before you get started:**
  - ❏ **Make sure you complete zyLabs 14.9 and 14.10**
  - ❏ **Watch the video in Section 14.4 Multiple Structs and Arrays in Structs.**
- ❏ There is a Unix Lab component to this assignment that must be completed as well. Instructions to this can be found in the Linux Lab 2 document. Watch this Lab 2 video and follow along.
- ❏ **Use this A06_starter.zip file to complete your assignment.**
- ❏ Open the files, and run it and test it before you get started. The program will compile and you should see the output below:

```
Welcome to my Course Roster Program

Here is the course roster:

Thank you for using my Student Roster program!!
```

- ❏ Open the Algorithmic Design Document, (this is specific to this assignment, so use this) make a copy, and follow the steps to create your algorithm.
- ❏ You must express your algorithms as **pseudocode. Notice I have pseudocode for the functions in the starter file. Write pseudocode only for the functions you need to write.**
- ❏ Your program must complete the following functions:
  - ❏ **`void printStd(Student student);`** function in Student.cpp. This function should print one student in this format: No fancy formatting needed.

```
Gayathri;Iyer;3.5
```

- **Write the following functions in Course.cpp. The prototypes are already given in Course.h**
  - **`void readStudent(ifstream &inFile, Course &course);`**
    - This function should take a file stream variable and Course struct by reference.
    - Use a while(!eof) loop to read from the file into a single **`Student`** (declare local variables for this).
    - Call **`addStudent(Student student, Course &course);`** and send the **`Student`** and the **`course`** to get the student added to the course.
    - The **`addStudent`** function has already been written. Right now it adds the student to the end of the list and increments the number of students. Check it out and make sure you understand how this function works. In the next few steps, you will modify this function, so you will insert the student into the list in the right position, so the list will be sorted (no sorting algorithms needed)
    - Repeat this process in the while loop until all students have been added.
  - **`void printRoster(Course course);`**
    - This function takes the course by value (the struct variable course is just used to print the students on the roster, the course is not going to be changed - hence no need to pass by reference), and prints the list of students.
    - Use a **`for loop`** (remember **`numStudents`** keeps track of the number of students in the Course struct), and call the **`printStd`** function to print each student.

STOP!!! Now is a good time to check your functions!

- Test the functions you have written so far to make sure you can read from the file and print the list of students.
- Part of **`main()`** is written for you. Make sure the functions **`readStudent()`** and **`printRoster()`** are being called in **`main()`**.
- Your output should look like this.

```
Welcome to my Course Roster Program

Here is the course roster:
Henry;Nguyen;3.5
Brenda;Stern;2
Lynda;Robison;3.2
```

```
Sonya;King;3.9
Gayathri;Iyer;3.5
Glen;Sasek;3.7
Priya;Goel;3.8

Thank you for using my Student Roster program!!
```

- **Write the rest of the functions in Course.cpp**
  - **`void dropStudent(char *lastname, Course &course);`**
    - This function receives the name of a student and the course and removes the student with that name from the roster.
    - If the name does not exist, do not do anything.
    - Notice the lastname is received as a pointer. See this example in zybooks for the pointer syntax.
    - Course variable is received by reference since the number of students and the array will change if you remove a student from the list.
    - Test the function by adding code to **`main()`** - there are comments where you should add code and call the function.
  - **`Student findStudentHighestGPA(Course course);`**
    - This function should go through the roster and return a **`Student`** with the highest gpa. The **`course`** struct has all the information you need - you can declare some local variables as needed.
    - Test the function by adding code to **`main()`** - there are comments where you should add code and call the function.
- **Modify the following function in Course.cpp**
  - **`void addStudent(Student student, Course &course);`**
    - Here you will modify the existing addStudent function. Right now the function adds students to the end of the list.
    - Modify the function so it adds students sorted by lastname - you should not use any sorting algorithm. Insert the students in the right position, by finding the position and shifting the other students. Here is an example from zyBooks Section 14.7.
    - Once you have modified this function, rerun the program to make sure it reads from the file and inserts students in the right position. Your output should look like this:

```
Priya;Goel;3.8
Gayathri;Iyer;3.5
Sonya;King;3.9
Henry;Nguyen;3.5
Lynda;Robison;3.2
Glen;Sasek;3.7
Brenda;Stern;2
```

Criteria for Success

- ❏ Run your program and compare it with the sample run below.

```
First sample run without modifying addStudent:

Welcome to my Course Roster Program

Here is the course roster:
Henry;Nguyen;3.5
Brenda;Stern;2
Lynda;Robison;3.2
Sonya;King;3.9
Gayathri;Iyer;3.5
Glen;Sasek;3.7
Priya;Goel;3.8

Enter the last name of the student to drop: King

Here is the course roster:
Henry;Nguyen;3.5
Brenda;Stern;2
Lynda;Robison;3.2
Gayathri;Iyer;3.5
Glen;Sasek;3.7
Priya;Goel;3.8

The student with the highest GPA:
Priya;Goel;3.8

Thank you for using my Student Roster program!!
```

```
Second sample run after modifying addStudent:

Welcome to my Course Roster Program

Here is the course roster:
Priya;Goel;3.8
Gayathri;Iyer;3.5
Sonya;King;3.9
Henry;Nguyen;3.5
Lynda;Robison;3.2
```

```
Glen;Sasek;3.7
Brenda;Stern;2

Enter the last name of the student to drop: Iyer

Here is the course roster:
Priya;Goel;3.8
Sonya;King;3.9
Henry;Nguyen;3.5
Lynda;Robison;3.2
Glen;Sasek;3.7
Brenda;Stern;2

The student with the highest GPA:
Sonya;King;3.9

Thank you for using my Student Roster program!!
```

- ❏ Complete zyBooks section **13. CS 161B: Structs Part I** and **14. CS 161B: Structs Part II** activities.
- ❏ You may use any IDE to write your code. But to be successful in future CS classes use the Linux server as much as you can.
- ❏ Please bookmark the PCC Linux and Vim Manual - this will become a frequently used reference!
- ❏ Complete the Linux Lab 2 document. Watch this Lab 2 video and follow along.
- ❏ Complete the required sections of your Algorithmic Design Document.
- ❏ **Follow these Coding Construct Requirements:**
  - ❏ Please open and compare your work with the grading rubric before submitting.
  - ❏ Remember to follow all style guidelines.
  - ❏ Do not use any vectors or containers for this program. Use only the concepts we have learned so far. Use only the 2 arrays for the 2 sets. No more extra arrays are needed.
  - ❏ Do not use any goto statements. You may not use any breaks or return statements inside any loops - you are allowed to use breaks inside a switch statement.
  - ❏ Must not use any sorting algorithms, the last names should be inserted in the right position.
  - ❏ You must use all the given function prototypes and not change any of them.
  - ❏ Download your Algorithmic Design Document as a PDF (File -> Download -> PDF), rename it to `a06.pdf`, and upload it to the D2L assignment by **Wednesday**.

- ❏ Upload all the files to D2L as a zip file or individual files to the D2L assignment by **Sunday**.
- ❏ Do your own work. Consult the syllabus for more information about academic integrity.

Additional Support

- ❏ Post a question for the instructor in the Ask Questions! area of the Course Lobby.
- ❏ **Make sure you complete zyLabs 14.9 and 14.10**
- ❏ **Watch the video in Section 14.4 Multiple Structs and Arrays in Structs.**