CS 161B: Programming and Problem Solving II

Assignment 4: Course Rosters

Grade Level A

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.

Course numbers are the main way colleges organize their course catalog. Course Rosters will have many different types of information and one of them is the number of students enrolled in that course. You will write a program to track the number of students enrolled in certain courses.

PANTHERS

Purpose

In this assignment, you will write a program that uses arrays to represent a list of courses and students enrolled in them and manipulate those arrays.

You will:
1. Write a program to read course numbers (a string of text) and the number of students enrolled (int) in the course into 2 parallel arrays.
2. Once the data has been read, you will need to cancel all courses where the number of students is less than 10. You will go through the lists, find the courses that have enrollment below 10 and remove them from the lists.
3. The maximum number of students that can be enrolled in a course is less than or equal to 25.
4. The maximum number of courses you can have is 20. The maximum number of characters you can have for a course is 51.

After completing this assignment you will be able to:

- Shift elements in array(s) to insert or delete elements
- Use a for loop to manipulate the array(s) without going out of bounds
- Pass arrays and their size to a function and manipulate the arrays

Task - This is the Grade A version.

What grade would you like?

We understand that your time and money is valuable, and we want to ensure that this programming assignment is tailored to your needs and circumstances. Whether you are a student juggling multiple commitments, a student dealing with personal responsibilities, or a student facing time constraints, or other issues in life, this assignment aims to provide programming skills at different levels that will meet the objectives of this week. With that in mind this assignment has been broken down into 3 different grade levels - Grades A (Exceeds), B (Meets) and C (Approaching). ***This is the Grade A version.***

Tasklist for getting a grade A:

The below set of tasks, if completed correctly per the rubric will get you a grade of **Exceeds** and help master the concepts required to learn the objectives for this week's topic. You can improve your grade by resubmitting the assignment once.

- ❏ Before you get started:
  - ❏ Check out Sample Assignment A04 - Algorithmic Design document
  - ❏ Check out Sample Assignment A04 - SampleA04.cpp

    - ❑ Check out the sample for inserting into an array in the right position.
- ❑ Your job is to create a program that will read course numbers (c-string or text data type) and number of students enrolled in that course (int data type) from the user. You will then call a function to cancel or remove courses that have less than 10 students enrolled in them.
- ❑ You must have functions to do the following:
- ❑ **`void welcome()`** - This function prints a welcome message to the user.
- ❑ **`void readInput(char courseNums[][MAXCHAR], int students[], int &count)`** ***(***Note: modified for this grade level***)***
    a. This function takes an array of c-strings, and an array of ints and reads the course numbers and students enrolled from the user.
    b. Read until the user enters "Quit" or "quit" for the course number. The SampleA04.cpp should help you do this. Read the corresponding number of students enrolled.
    c. Call the **`readInt`** function to read the number of students for each course number. You must catch all invalid data such as characters, negative numbers etc.
    d. You must also do validation to make sure that the number is between 0 and 25. The maximum number of students that can be enrolled in a course is less than or equal to 25. Use a while loop to do this.
    e. Insert the course number and the number of students in the right position based on the course number. **You must not use any sorting algorithm. Insert the course numbers and its students in the right position, by finding the position and shifting the other courses. Here is an example - insertSorted.cpp. See sample run below.**
    f. **`count`** will keep track of the number of courses entered.
    g. Test your function with at least 10 courses to make sure that the insertion works correctly.
- ❑ **`void readInt(string prompt, int &num)`**
    a. This function should be used any time you read any integers from the user. Use this function to read the menu option from the user.
    b. It takes a string prompt, outputs it, reads a number from the user, validates and returns the num by reference.
    c. See Samplea01.cpp for the function. You may use the function in the sample.
    d. You must catch all invalid data such as characters, negative numbers etc.
- ❑ **`void printList(char courseNums[][MAXCHAR], int students[], int count)`**
    a. This function takes the 2 arrays and count, uses a for loop and prints the lists to the user.
    b. Format your list neatly.
- ❑ **`void cancelCourses(char courseNums[][MAXCHAR], int students[], int &count)`**

   a. This function takes the 2 arrays and the count and removes all courses with less than 10 students in the course.
   b. You must use a for loop, find the index where the number of students is less than 10, and then remove that number and the corresponding course number from the char array by shifting elements in both the arrays.
   c. Update the count appropriately. If you cancel or remove 3 courses, then count should be subtracted by 3, so the number of courses is updated.

- **`int main()`**
   a. Declare the arrays for course numbers and students.
   b. Call the **`readInput()`** function and get the arrays filled. **The list must be ordered by the course numbers. This is important for this grade level.**
   c. Call the **`printList()`** function and print the course numbers and students neatly formatted.
   d. Call the **`cancelCourses()`** function and cancel the courses with less than10 students in them.
   e. Call the **`printList()`** function again and print the course numbers and students neatly formatted.
   f. End your program.
- Open the Algorithmic Design Document, make a copy, and follow the steps to create your algorithm.
- You must express your algorithm as **pseudocode.**
- **Check out all the coding constructs under Criteria for Success.**

Criteria for Success

- Look at the sample run below. Use the function prototypes you will need for the various options (your list may include other options):

```
Welcome to my Course Rosters program!!
Enter course number and students enrolled when prompted.
Enter Quit or quit for course number when you are done.
Enter course number: CS133G
Number of students enrolled: 7
Enter course number: CS160
Number of students enrolled: 15
Enter course number: CS161A
Number of students enrolled: 21
Enter course number: CS161B
Number of students enrolled: 23
Enter course number: CS162
Number of students enrolled: 30
Invalid number!! Please enter a number between 0 and 25
Number of students enrolled: 25
```

```
Enter course number: CS260
Number of students enrolled: 5
Enter course number: Quit

List of courses and students:
CS133G          7
CS160           15
CS161A          21
CS161B          23
CS162           25
CS260           5

List after cancellations:
CS160           15
CS162A          21
CS161B          23
CS162           25


Thank you for checking out my Course Rosters program!
```

```
Welcome to my Course Rosters program!!
Enter course number and students enrolled when prompted.
Enter Quit or quit for course number when you are done.
Enter course number: CS133G
Number of students enrolled: 7
Enter course number: CS160_1
Number of students enrolled: 15
Enter course number: CS161A_1
Number of students enrolled: 21
Enter course number: CS161B
Number of students enrolled: 23
Enter course number: CS162
Number of students enrolled: 30
Invalid number!! Please enter a number between 0 and 25
Number of students enrolled: 25
Enter course number: CS260
Number of students enrolled: 5
Enter course number: CS160_2
Number of students enrolled: 7
Enter course number: CS161A_2
```

```
Number of students enrolled: 25
Enter course number: quit

List of courses and students:
CS133G          7
CS160_1         15
CS160_2         7
CS161A_1        21
CS161A_2        25
CS161B          23
CS162           25
CS260           5

List after cancellations:
CS160_1         15
CS161A_1        21
CS161A_2        25
CS161B          23
CS162           25


Thank you for checking out my Course Rosters program!
```

- ❏ Check out Sample Assignment A04 - Algorithmic Design document
- ❏ Check out Sample Assignment A04 - sampleA04.cpp
- ❏ Complete zyBooks section **Chapter 11: Sections 11.12 to 11.14 CS 161B Arrays** activities, and **zyLabs 11.17 and 11.18.**
- ❏ 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 all sections of your Algorithmic Design Document.
- ❏ **Follow these Coding Construct Requirements:**
    - ❏ Do not use any structs, vectors or containers for this program. Use only the concepts we have learned so far.
    - ❏ You may not use the String class - you must use C-strings or char arrays.
    - ❏ For this grade level A, the lists must be ordered by Course numbers. Make sure your **`readInput()`** function is written appropriately.

    - ❑ 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.
    - ❑ All input data must be validated. For example, you should not accept any characters for numerical data types.
    - ❑ **Must not use any sorting algorithms, the course numbers should be inserted in the right order.**
    - ❑ Must use all functions mentioned under Task - there should be no code redundancy.
    - ❑ Print a welcome and goodbye message.
- ❑ Include **pseudocode** in part d of the design document.
- ❑ Please open and compare your work with the grading rubric before submitting.
- ❑ Remember to follow all style guidelines.
- ❑ Download your Algorithmic Design Document as a PDF (File -> Download -> PDF), rename it to `a04.pdf`, and upload it to the D2L assignment by **Wednesday**.
- ❑ Upload your `a04.cpp` C++ source file to the D2L assignment by **Sunday**.
- ❑ Do your own work. Consult the syllabus for more information about academic integrity.

Additional Support

- ❑ Before you get started:
    - ❑ Check out Sample Assignment A04 - Algorithmic Design document
    - ❑ Check out Sample Assignment A04 - sampleA04.cpp
- ❑ Post a question for the instructor in the Ask Questions! area of the Course Lobby.