CST8234- C Language

Assignment 1

Due Date - October 4, 2019

Problem statment:

In this assignment, we will look at a way to develop a students' registration system using C for some college. The goal of this program is to allow the administrators to register a student in offered courses, and also allows them to drop a course for a student.

This assignment is intended to get you to practise using arrays, pointers , functions, header files, user inputs, and formatting output among other things.

Background Information:

In this assignment we will have two arrays, one to store the students IDs, and the second is to store the course codes.

In addition, there will be a registration table, represented by two dimensional array, that will store the courses each student is registered in. This will be presented as a simple yes/no value stored in the registration table for each student using their index in the students array, and the course index in the courses array.

For example, if the system have three students with IDs {12345, 34567, 56789}, and have two courses with codes {"CST8234", "CST8288" }. Then the administrator can register a student with ID "34567" in a course with code "CST8234" by recording "Yes" in the registration table for index 1 (representing the student index) and 0 (representing the course index). The following tables illustrate the memory representation for of each of the arrays

Students		
Address	Index	Value
0x303300	0	12345
0x303304	1	34567
0x303308	2	56789

Courses		
Address	Index	Value
0x308800	0	CST8234
0x308809	1	CST8288

Registration Table			
Index	Student Index	Course Index	Value 0 = no / 1 = yes
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	1
4	2	0	1
5	2	1	0

Therefore, from the above table we can see that the student with index 0 (ID: 12345) is registered in course with index 1 (Code: CST8288) only.

Requirements:

Write a program that achieves the following requirements:

1. The program should ask the user to enter
    1. The number of students they wish to register.
    2. The student id for each student. Student IDs are 5-digits integer, i.e 45234.
2. Students should be stored in their own array
3. The program then should ask the user to enter:
    1. The number of courses offered.
    2. The course code of each of these courses. Course codes are 7-digit alphanumeric string, i.e CST8234.
4. Courses should be stored in their own array.
5. The program then should ask the user to choose one of three action:
    1. Registering a student in a course.
    2. Dropping a student from a course
    3. See Registration table.
6. If the user chooses to register a student or drop a student, then the program should ask for the student id first then the course code second and perform the correct action as follow:
    1. Registering a student will update the registration table by adding “1” to the element with the [student index][course index] element.
    2. Dropping a course will update the registration table by adding “0” to the element with the [student index][course index] element.
7. If the user chooses to see the registration table, the program should print all entries in the table
8. Bonus 2 points:
    1. Add another action to the menu that will allow the user to quit the program by choosing the quit action .
    2. The program would loop until the user quits the program. The program will loop only starting from point 5 above, so the user doesn’t need to enter the students or the courses information anymore.

Design Requirements:

1. Make sure you use functions, design your program to separate functionality into its own functions. Using only **`main`** function will make you loose 2 points.
2. Give your functions and variables a descriptive names. For example, `students[]`, not `x[].`

Supporting files:

Along with this document, you will find two file included on Brightspace. The files are named **`helper.h`** and **`helper.c.`** These files contains generic functions that can be used to help on finishing this assignment. Please read the comments above each on of these function to understand the intent and the usage example. If you are going to use these function, you will need to include **`helper.h`** in you program.

Submission instructions:

1. No late submissions are accepted.
2. You can work in a group of maximum 2 students to complete this assignment. Individual work is also accepted.
3. You must submit the source code for the program you wrote.
4. Add all your files under a folder call “lastName-firstName-Assign1”, then Zip the file and submit the zip file only. Make sure to submit all files required to compile and run the program on the instructor machine without any errors.
5. DON’T submit any extra file please. For example, the binary file (AKA object or output) files like .exe or .o.
6. If working in a group, add a Readme.txt file that contain each student name and student number.
7. Brightspace is configured to keep the last submission only. Please make sure your last submission in the one you want to get marked.
8. All submission must be done on the main Brightspace shell, 19F_CST8234_010_ALL or 19F_CST8234_020_ALL, not the lab section one.