**CS340 P1**

Use visual C++ (2017 or above) to implement a menu-driven program that helps manage information on students, courses and grades for a university. Your program should first read input data from 3 files named “student.dat”, “course.dat” and “grade.dat”, and then display a menu including the following options:

1. Add/modify a grade

2. Remove a grade

3. Student grade report

4. Course grade report

5. List students

6. Display course listing

7. Exit

**1. Add/change a grade**

Ask the user for student id, course number and grade, and then check to see if the student already had a grade with the given course. If yes, change the grade. Otherwise, add the grade to the system. Print an appropriate message if the student number or course does not exist. However, if both student id and course number are valid, your program must make sure that a valid grade (i.e., one of A, B, C, D, F, I) is entered. The user may also quit from this option by entering a ‘Q’ grade.

**2. Remove a grade**

Ask the user for the student id number and class number, and remove the grade from the system. Print an appropriate message if such a grade does not exist.

**3. Student grade report**

Ask for a student id number from the user and the display a complete course grade report for that student. Display the grades of ALL students in ALL the classes if an \* is entered as the id. The report must be in a table format with the following fields:

Firstname Lastname ID CourseName CourseNumber Grade.

**4. Course grade report**

Ask for a course number from the user and display the grade of an entire class. If the class number is \*, then display the grades of ALL students in ALL the classes. Use the same display format as specified in option 3.

**5. List students**

Display a list of all the students in a table format with the following fields:

Firstname Lastname ID Phone

**6. Display course listing**

Display a list of all the courses in a table format with the following fields:

CourseNumber CourseName

**7. Exit**

Print the student listing, course listing and grade reports to 3 output files named “student.dat”, “course.dat”, “grade.dat” and terminate the program. Note that each output file should contain the same order of fields and records as its corresponding input files.

After each of the above option, program should return to the menu, until the user select 7, in which case the program would save and terminate.

**Sample I/O**

Sample input and output files as well as an executable have been placed in the class folder. Note that each input and output file contains a line of field headers followed by many records. For this assignment, you may assume that the university has up to 10000 students and 1000 courses. In addition, you may assume that each (string) data item such as courseName is a single string data with NO space.

# Data Structure

In addition to typical comments such as your name(s), project summary, and brief information for each (important?) function, you must also include a discussion of your data structure, pro and con, and why you chose that. If you used ChatGPT or any AI, describe what you gained and learned from it. These comments are 20% of the grade.

**Due Date**

This assignment is worth 100 points andisdue **Thursday (1/30) at 10:00 PM**. For all programming assignments, you may work in **groups of two**. Submit your source code (or the entire VS project) on Moodle. Each group only needs to submit one program. **Final deadline is 10:00 PM Saturday (2/1).** Standard late penalty applies.

**Data Structure**

Student and course are easy: 2 vectors of struct, ex

Struct studentType

{

String fnae, lname, id, phone;

}

Vector<studentType> student;

What do we use for grades?

1. 2D array (or vector of vectors)

For example, using a table (i.e., two-dimensional array) to store the grades for ALL students at SIUE. Each entry contains the grade information (ex., char grade, …)

const int maxNumOfStudents=14000;

const int maxNumberOfCourses=1500;

char grade[maxNumOfStudents][maxNumberOfCourses];

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| student\course | CS140 | CS150 | CS340 | Eng101 | Chem240 | … |
| 1348 | B |  |  |  | A |  |
| 2999 | A | B | C |  |  |  |
| 1699 | C |  |  |  |  |  |
| … |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

Good: Add/delete/change is super easy using this table

Q. How dense is the table?

Not very much

Sparsely populated table

Problem: inefficient

1. Vector of grade struct

Struct gradeType

{

Char grade;

String ID, number;

}

Pro: easy to implement, less waste

Con: they are not organized by students or courses

Ex., to list all grades for a particular class, we have to traverse the entire vector -. Inefficient

How do we make this efficient in both time and space?

1. Two Vector of vectors: one for student, one for courses, efficient but not as ideal as list, ex.,

Vector<vector<gradeType>> studentGrades;

Vector<vector<gradeType> courseGrades;

Each grade will be stored in two locations: one with studentGrade one with courseGrade

So there is potential for inconsistencies and of course some inefficiencies

Also vector deletion and resize isn’t as efficient as list

1. two vectors of list – more efficient than vector of vectors

ex.,

Vector<list<gradeType>> studentGrades;

Vector<list<gradeType> courseGrades;