CSE 241 Homework Assignment 4

DUE

April 25, 2019, 23:55

Description

• This is an individual assignment. Please do not collaborate.

• If you think that this document does not clearly describe the assignment, ask questions before it's too late.

You won't be given a chance to correct any mistakes.

Scope: University Management System

You are expected to develop university management system.

This module you develop in the scope of assignment 4 will be continuation of assignment 2 and assignment 3.

Main Problem: Attendance Record

Requirements:

1. Lecturers have an ability to get attendance record of courses s/he gives.
2. Attendance records are kept in the classroom. It comprises the action of students; as entering the classroom and quitting classroom.

Expectation:

1. Separate compilation: You should separate implementations and interfaces for all classes you define.
2. Define and use your namespace
3. Provide big three for all classes; copy constructor, destructor and assignment operator overloading
4. Attendance records will be managed with dynamic array.
5. **Struct** of Classroom will be converted to **Class**. Also, you might define new class or struct if you think it is needed.
6. EnterClassroom and QuitClassroom functions will determine attendance. So, attendance records will be reached from Classroom.
7. Assume that academic semester will be only one week. So, date of attendance will not be considered. It is important to count hour of courses, i.e. , how many hours students participates.

Test Scenarios

1. Read txt file for courses, classrooms and lecturers. Automatically assign lecturers, classrooms and timetable.
   (input)>-enter [student_id] [classroom_id] [hour_of_week] ex: -enter 3 312 18 : student 3 enters classroom 312 at 18th day of the week.
   (output)> 23 (student_inroom) | BLOCK: s/he is at another classroom | ERROR: no class or no student
   (input>-quit [student_id]
   (output)> 22 (student_inroom) | BLOCK: s/he is not in any classroom | ERROR: no class or no student
   (input)>-attendance [course_id]
   (output)> Ali 3 – Veli 4 – Necdet 1 [list of enrolled students with attendance count]

Remarks

• Do not use any elements which is not covered in class.

• Do not submit your code without testing it with several different scenarios.

• Write comments in your code.

• Prepare a Makefile for your programme.

Turn in:

• Source code of a complete C++ program. Name of the file should be in this format: <full_name>_<id>.cpp.

• Example: gokhan_kaya_000000.cpp. Please do not use any Turkish special characters.

• You don't need to use an IDE for this assignment. Your code will be compiled and run in a command window.

• Your code will be compiled and tested on a Linux machine(Ubuntu). GCC will be used

• Make sure you don't get compile errors when you issue this command : g++ <full_name>_<id>.cpp.

• A script will be used in order to check the correctness of your results. So, be careful not to violate the expected output format.

• Provide comments unless you are not interested in partial credit. (If I cannot easily understand your design, you may loose points.)

• You may not get full credit if your implementation contradicts with the statements in this document.

Late Submission

- (0,24] hours: -20%
- (24,48] hours: -40%
- (48,72] hours: -60%
- (72,-) hours: -100%

Grading (Tentative)

- Max Grade : 100.
- Multiple tests(at least 5) will be performed.

All of the followings are possible deductions from Max Grade.

- #define HARD_CODED_VALUES -10.
- No submission: -100. (be consistent in doing this and your overall grade will converge to N/A) (To be specific: if you miss 3 assignments you'll get N/A)
- Compile errors: -100.
- Irrelevant code: -100.
- Major parts are missing: -100.
- Unnecessarily long code: -30.
- Using language elements and libraries which are not allowed: -100.
- Not caring about the structure and efficiency: -30. (avoid using hard-coded values, avoid hard-to-follow expressions, avoid code repetition, avoid unnecessary loops).
- Significant number of compiler warnings: -10.
- Not commented enough: -5. (Comments are in English).
- Source code encoding is not UTF-8 and characters are not properly displayed: -5. (You can use 'Visual Studio Code', 'Sublime Text', 'Atom' etc. . . Check the character encoding of your text editor and set it to UTF-8).
- Missing or wrong output values: Fails the test.
- Output format is wrong: -30.
- Infinite loop: Fails the test.
- Segmentation fault: Fails the test.
- Fails 5 or more random tests: -100.
- Fails the test: deduction up to 20.
- Prints anything extra: -30.
- Unwanted chars and spaces in output.txt: -30.
- Submission includes files other than the expected: -10.
- Submission does not follow the file naming convention: -10.
- Sharing or inheriting code: -200