CSCI479/112Spring2021

Moodle Simulator

Project Description:

- It is required to develop a programthatsimulates the Moodle platform for course management.
- The code should be organized in 3 files: **student.h**, **student.cpp**, **main.cpp**
- The **student.h** file will contain class definitions (data members and member function prototypes only).
- The **student.cpp** file should contain member functions definitions.
- The **main.cpp** file should contain objects creation and other needed functions.

Operations:

(1) **student.h** and **student.cpp** files details:

Operation	**Sub operation**	**Action Required**
class CStudent	private variables	char student_name[50], int student_ID, char student_email_username[10], char student_major[10], float student_grades[5] float student_score, char student_email_password[10]
	public variables	**NONE**. Create **setter** and **getter** functions to access all private variables (e.g. char* getStudentName(); int getStudentID(); void setStudentName(char *); and so on.
	Constructor	Initialize each variable with NULL for strings and zero for other numbers.
	registerStudent()	A public function that 1) prompts the user to enter student data, and 2) stores these values for the CStudent class members.
	getStudentInfo()	A public function that prints the student’s data.
	claculateGPA()	A public function that sums the items of student_grades[5] and divide the sum over 100 and assign the result to the student_score variable.
	Destructor	The destructor should delete all class data members

		(NULL for strings and zero for other numbers).
class CCourse	public variables	**NONE**. Create **setter** and **getter** functions to access all private variables.
	private variables	char course_name[20], char course_code[5], float course_cost[5]
	AddCourse()	A public function that accepts values for the Course class members and store them.
	getCourseInfo()	A public function that prints the course's details.
	Destructor	Delete all courses information (NULL for strings and zero for other numbers)..
class CPG_Student ***(BONUS)*** (A class for postgraduate students. This class inherits from CStudent class and adds a new data member: pg_student_job_title)	private variables	This class should **inherit** the CStudent class and adds another private data member: char pg_student_job_title[20].
	public variables	**NONE**. Create **setter** and **getter** functions to access all private variables for the CPG_Student class.
(2) main.cpp file:		
main()		1- First, input the number of students (n_students) to be registered in the system and the number of courses to be added to the system (n_courses) 2- Then create an array of CStudent objects and another one for CCourse objects (student[n_ n_students], courses[n_courses]. 3- Create a for loop from 1 to n_students that calls registerStudent() function to store students' data and similar loop for inputting courses' data in the same manner. 4- Create another for loop from 1 to n_students that calls getStudentInfo() function for each element in the array and display its data. Do the same for the courses. 5- (**Bonus**) Inherit the class CPG_Student from the CStudent class and perform the same operations for the inherited class as in 1,2,3,4 while taking into consideration the new data member (pg_student_job_title).

General Constraints:

- All your code should be in **one folder**.
- Output should not include any extra white spaces or anyextra text more than the results.
- Do not clear the screen after every operation.
- Just **one of the team members must submit the file**, with a comment inside the main function with student names a

Grading Rubrics:

- Specifications: The program works and meets all the requirements (55%).
- Readability: The code is well organized and easy to follow (15%).
- Documentation: The code is well documented and clearly explained (20%).
- Delivery: The program was delivered on time (10%).
- Bonus: 20%

Submission:

Each project submission (on Moodle) must include:

1- A whole code project's folder (zipped).

2- A report with the following:

a. Team member names and IDs
b. Application description
c. Flowchart of execution sequence
d. Sample input and output screens.

Due Date:

Thursday 8 July 2021, 10:00pm

Late submissions will be penalized.

Teams:

Work in groups of 4-5 students.

Plagiarism:

Plagiarism is a serious academic offence and students who share code with others or get any source from the internet will fail the course. A plagiarism detection tool will be used to check all projects submitted and check and report plagiarism cases.