CSC 317 / INF 307 **Assignment 02** **Date**

Due date: Sunday, 10th April 2022 at 11:55pm GMT
Late submission: Penalties apply as stated in the course outline
Submission instructions:

1. This is a group assignment. Groups should contain either 3 or 4 members
2. Submissions will be made via the e-learning platform. Only 1 submission per group
3. All documents and code should be placed together in a ***single zipped file***
4. The submission should contain all names and IDs of group members.
   Excuses about missing group member details would not be tolerated

---

1. Problem Description

This is an extension of the design problem used during the lab sessions.

The DCSIT needs a new system to track all of its students, instructors and courses. It wants to keep track of what courses are offered, who teaches each course and which students are enrolled in those courses. It would also like to be able to track the grades of each of its students across all courses. For each student and instructor, the school needs to know their address, phone number, name, and age.

Each course has a maximum and minimum number of students that they can enroll. If the minimum number of students is not reached, then the course will be cancelled. Each course is taught by at least one instructor but sometimes may be taught by many.

There are three types of instructors: part-time lecturers, full-time lecturers, and professors. Part-time lecturers can teach a maximum of 2 courses a semester. Full-time lecturers can teach a minimum of 4 courses a semester. Professors can teach as little or as many courses they wish. All instructors are salaried employees at the DCSIT and therefore we need to keep track of how much they make each year. An instructor earns a base pay of GHS2500 for each course taught in a semester. If a full-time lecturer teaches more than 4 courses in a semester, they are granted an extra onetime bonus of GHS5,000. Professors that teach more than 2 courses earn a bonus of GHS1500 for each extra course.

Students can be either full or part time. A student is considered a part time student if they are enrolled in 1 or 2 courses during any given semester. The maximum number of courses a student may be enrolled in at one time is 6. Students receive grades from each course, these grades are numeric within the range of 0-100. Any students that have an average grade across all enrolled courses lower than 60% is said to be on academic probation.

NOTE: This system will be reset and updated at the end of each semester

2. Design Requirements

Prepare a short design document that contains the following:

2.1 Use Case Diagram of the above system
2.2 Class Diagram of the above system
2.3 A discussion of any design decisions the group took.

3. Functional Requirements

Each team's implementation should be able to show the following:

3.1 A list of all courses, with the number of enrolled students, existing in the department

3.2 A list of all courses, with their assigned instructor(s)
3.3 A list of all instructors, with their types, in the department
3.4 A list of all students, with their full/part-time status
3.5 A list of all cancelled courses
3.6 A list of instructors with their salaries (broken down into base, bonus and total)
3.7 The courses and grades of an individual student
3.8 A list of students on academic probation

4. Testing Requirements

Your team must develop appropriate test cases in test files to demonstrate that each requirement is working as expected. Make sure your tests cover both successful and unsuccessful scenarios.

5. Marking Scheme

Binary Components				
Code Compiles /0	Code does not compile (will receive 0 on all technical components)		Code compiles	
Design Components				
	Unacceptable	Poor	Good	Excellent
Documentation /4	No design document and no commented code	Elements of the UML were missing. Few comments in code	UML diagrams capture most design decisions. Comments placed when necessary	Complete UML diagrams. Code is thoroughly commented
Technical Components (will receive 0 if code does not compile)				
	Unacceptable	Poor	Good	Excellent
Functional requirement implementation /8	Most functional requirements are missing	Most functional requirements implemented, but error prone or missing functionality	All functional requirements are implemented, but some have errors	All functional requirements are implemented without errors
Completeness of test suite /4	The test suite is missing	Few elements are tested within the suite, or the majority of the test suite has errors	All elements of the requirements are tested within the suite, although with some errors	All elements of the requirements are tested within the test suite without any errors
Simplicity and readability /4	Code is not readable, naming of classes, variables, and methods are meaningless or irrelevant	Code is difficult to read, solutions overly complex. Naming of classes, variables, and methods is Somewhat meaningful, and/or inconsistent	Code is readable, solutions straightforward. Naming of variables, classes, and methods is consistent and meaningful	Code is easy to read, care was put in to simplify the solution for modularity & readability. Naming of variables, classes, and methods is consistent and meaningful, with extra effort added to differentiate data types