Object Oriented Programming (CS-239)

_____

Submitted to:

Miss Fiza Abdul Razzaq

Submitted by:

Moazam Attiq (221400003)

Topic:

Assignment # 3

(Proposal & UML Diagram)

_____

BS Software Engineering (II)

GIFT University, Gujranwala

Requirements:

1. Users should be able to enroll students into courses.
2. Users should be able to assign teachers to courses.
3. Users should be able to create and manage departments.
4. Students should be able to enroll in courses.
5. Teachers should be able to allot courses.
6. Users can retrieve student, teacher, course, and department information.
7. Users can update student and teacher profiles.
8. Users can view a list of all enrolled courses.
9. Users can list all courses taught by a teacher.
10. Users can list all students in a classroom.
11. Users can search for students, teachers, or courses by specific criteria.
12. Users can remove students from courses and department.
13. Users can retain teachers from courses.
14. Users can handle course scheduling and timetable.
15. Users can manage course prerequisites and curriculum.
16. Users can add and remove courses from the curriculum.
17. Users can check attendance for each course of a particular student
18. Users can check the grade of each course individually for each student

UML for Assignment 3 (Object Oriented Programming)

<> *Person*
- -name:String
- -age:int
- -address:String

- +Person()
- +Person(n:String, a:int, add:String)
- <>
- +setters
- +*getName*():String
- +*getAge*():int
- +*getAddress*():String

<< interface >> *Enrollable*

- +enroll(c:Course):void
- +listCourses():Course[]

Attendace
- -course:Course
- -students:Student[]
- -attendanceRecord:boolean[][]

- +Attendance(c:Course, numOfStu:int)
- +recordAttendance(stu:Student, status:boolean):void
- +getAttendanceRecord(stu:Student):boolean[]

Student
- -studentId:int
- -enrolledCourses:Course[]
- -attendance:Attendance[]
- -grade:GradeBook

- +Student()
- +Student(inherited-3-args,id:int)
- +setter and getter for studentId
- +enroll(c:Course):void
- +listCourses():Course[]
- +editInfo():void
- +getInfo():void

Teacher
- -teacherId:int
- -courses:Course[]

- +Teacher()
- +Teacher(inherited-3-args,id:int)
- +setter & getter for teacherId
- +enroll(c:Course):void
- +listCourses():Course[]
- +editInfo():void
- +getInfo():void

GradeBook
- -courses:Course[]
- -students:Student[]
- -grades:char[][]

- +GradeBook()
- +GradeBook(courses:Course[], students:Student[])
- +recordGrade(stu:Student, c:Course, grade:char):void
- +getGrade(stu:Student, c:Course):char

Department
- -departmentName:String
- -students:Student[]

- +Department()
- +Department(dName:String)
- +addStudents(s:Student):void
- +removeStudent(s:Student):void
- +getStudents():Student[]

Course
- -courseCode:String
- -courseName:String
- -teacher:Teacher
- -enrolledStudents:Student[]

- +Course()
- +Course(code:String,name:String)
- +getters for course name & code
- +assignTeacher(t:Teacher):void
- +getTeacher():Teacher
- +enrollStudent(s:Student):void
- +getEnrolledStudents():Student[]