UNIVERSITYOF HERTFORDSHIRE
School of Engineering and Computer Science

COURSEWORK ASSIGNMENT

Module Title: Programming for Software Engineers	**Module Code: 7COM1025**
Assignment Title: Coursework	**Individual Assignment**
Tutor: Dr Hui Cheng	**Internal Moderator**: Dr Raimund Kirner

Student ID Number **ONLY**:	Year Code:

Marks Awarded %:	Marks Awarded after Lateness Penalty applied %:

Penalties for Late Submissions

- Late submission of any item of coursework for each day or part thereof (or for hard copy submission only, working day or part thereof) for up to five days after the published deadline, coursework relating to modules at Levels 0, 4, 5, 6 submitted late (including deferred coursework, but with the exception of referred coursework), will have the numeric grade reduced by 10 grade points until or unless the numeric grade reaches or is 40. Where the numeric grade awarded for the assessment is less than 40, no lateness penalty will be applied.
- Late submission of referred coursework will automatically be awarded a grade of zero (0).
- Coursework (including deferred coursework) submitted later than five days (five working days in the case of hard copy submission) after the published deadline will be awarded a grade of zero (0).
- Where genuine serious adverse circumstances apply, you may apply for an extension to the hand-in date, provided the extension is requested a reasonable period in advance of the deadline.

Please refer to your student handbook for details about the grading schemes used by the School when assessing your work. Guidance on assessment will also be given in the Module Guide.

Guidance on avoiding academic assessment offences such as plagiarism and collusion is given at this URL: http://www.studynet.herts.ac.uk/ptl/common/LIS.nsf/lis/citing_menu

UNIVERSITYOF HERTFORDSHIRE
School of Engineering and Computer Science

ASSIGNMENT BRIEF

Students, you should delete this section before submitting your work.

This Assignment assesses the following module Learning Outcomes (Take these from the module DMD):

- LO4 - Design and implement a substantial software artefact.
- LO5 - Evaluate, choose and employ appropriate data structures and algorithms.
- LO6 - Refactor part of a software item to achieve a specified goal.
- LO7 - Be able to design a test suite and perform a set of tests.

Assignment Brief:

Implement a system according to the following description

The Extra Tuition Centre (ETC) needs software for managing all the bookings and recording book order requests. There are several tutors who work in ETC. Each tutor has one or more subject areas, e.g., English comprehension, English writing, Math, Numerical Reasoning, Verbal Reasoning, Non-verbal Reasoning. Each student is registered with one main tutor who monitors his/her overall progress. Each student has a record stored on the system holding their name, gender, DoB, address, emergency contact phone number, and a list of all lessons past and booked. Some book orders may also be requested (Mental Math 1, English Comprehension 3, for example) for a student, and will be listed in the student record too.

Students use the software system to book lessons. All the tuition lessons are one to one and last for one hour. When requesting a new lesson, the student will describe the needed subject (English comprehension, English writing, Math, Numerical Reasoning, Verbal Reasoning, Non-verbal Reasoning, etc). The system will then arrange a lesson with date/time convenient to the tutor in that subject and inform the student. A student may change the date/time of the booked lesson subject to the availability of the booked tutor or other tutors in the same subject. A student may cancel a lesson, fail to attend a lesson, or attend the lesson.

On attending a lesson, the student will first be checked in, then will be taught by the booked tutor, before finally the lesson is finished. At the end of the lesson, the tutor may add some notes to the student's record about progress, and/or amend the current list of books needed for this student.

As the book supplier, the shopkeeper at CGP Books will use the software to retrieve the current list of books from each student's record.

After each month, the Extra Tuition Centre must print a report of how many lessons each student has booked, how many lessons were attended, missed or cancelled, and how many books were ordered.

In your final deliverable, have the program set up some tutors, students, and lessons in the past and in the future. You should provide data covering at least 3 tutors, 10 students and 15 lessons, which can then be displayed in the output reports.

Note: you do not need any kind of external database for this program. The final program should be self-contained. The reports can be printed to System.out, or output to a suitable GUI interface.

You also do not need any kind of security protocol. All the students and tutors may be assumed to be pre-registered.

Your tasks:

1. Create a class diagram, highlighting the main classes in the system and their associations. Include the attributes and key methods.
2. Implement the system, using Junit tests to confirm the functionality.
3. Write a short report (up to 6 pages should be sufficient) explaining your program.

Discuss:

a) Any assumptions made about the system, in addition to the above description.
b) The overall structure and design of your program.
c) Any design patterns/design principles used.
d) The JUnit tests.
e) Any refactoring used during the development of the system.

Deliverables:

A. Develop your code using bitbucket. Send your username to Hui Cheng by emailing to: h.cheng2@herts.ac.uk, and add “huicheng” to your project.

B. Submit to StudyNet:

1. Your report with the UML class diagrams in a single PDF document.
2. An executable jar file, demonstrating your final system. The system should have some data preloaded (as described above).
3. A zip-file containing the source code of your project.

Submission Requirements:

Students should submit on StudyNet the following three items:

1. a single pdf document containing the practical report and UML class diagrams,
2. a zipped folder containing the final source code, and
3. an executable jar file for running the final system.

This assignment is worth 50**%** of the overall assessment for this module.

Marks awarded for:

The marks will be based mainly on the report, in the pdf document. Source code will be checked to confirm both that the work has been implemented by yourself, and its design.

. UML Class diagram. (10 marks)

. Working jar file. (40 marks)

. Use of version control, with suitable commit messages. (5 marks)

. Appropriate design and implementation. (30 marks)

. Use of JUnit testing. (10 marks)

. Report format and writing quality. (5 marks)

Total: 100 marks

A note to the Students:

1. For undergraduate modules, a score above 40% represent a pass performance at honours level.
2. For postgraduate modules, a score of 50% or above represents a pass mark.
3. Modules may have several components of assessment and may require a pass in all elements. For further details, please consult the relevant Module Guide or ask the Module Leader.

Typical (hours) required by the student(s) to complete the assignment: 40 hours

Date Work handed out:	**Date Work to be handed in:**	**Target Date for the return of the marked assignment:**
16/10/2019	11/12/2019 at 9am	15/1/2020

Type of Feedback to be given for this assignment:

Comments will be provided along with marks.