1st Sit Coursework 1 Question Paper **Year Long 2023 2024**

Module Code:	**CS4001NI**
Module Title:	**Programming**
Module Leader:	**Mr. Mohit Sharma** (Islington College)

Coursework Type:	**Individual**
Coursework Weight:	This coursework accounts for **30%** of your total module grades.
Submission Date:	**Friday, 26 January 2024**
When Coursework is given out:	**8th Week**
Submission Instructions:	Submit the following to Islington College's **MST Assignment** Portal before the due date: ● **A report in PDF format and a zip file which includes program file.** ● **File should be in .java format**
Warning:	London Metropolitan University and Islington College takes Plagiarism seriously. Offenders will be dealt with sternly.

© London Metropolitan University

Plagiarism Notice

You are reminded that there exist regulations concerning plagiarism.

Extracts from University Regulations on Cheating, Plagiarism and Collusion

Section 2.3: “The following broad types of offence can be identified and are provided as indicative examples …..

(i) Cheating: including copying coursework.
(ii) Falsifying data in experimental results.
(iii) Personation, where a substitute takes an examination or test on behalf of the candidate. Both candidate and substitute may be guilty of an offence under these Regulations.
(iv) Bribery or attempted bribery of a person thought to have some influence on the candidate’s assessment.
(v) Collusion to present joint work as the work solely of one individual.
(vi) Plagiarism, where the work or ideas of another are presented as the candidate’s own.
(vii) Other conduct calculated to secure an advantage on assessment.
(viii) Assisting in any of the above.

Some notes on what this means for students:

(i) Copying another student's work is an offence, whether from a copy on paper or from a computer file, and in whatever form the intellectual property being copied takes, including text, mathematical notation and computer programs.
(ii) Taking extracts from published sources without attribution is an offence. To quote ideas, sometimes using extracts, is generally to be encouraged. Quoting ideas is achieved by stating an author's argument and attributing it, perhaps by quoting, immediately in the text, his or her name and year of publication, e.g. " e = mc2 (Einstein 1905)". A reference section at the end of your work should then list all such references in alphabetical order of authors' surnames. (There are variations on this referencing system which your tutors may prefer you to use.) If you wish to quote a paragraph or so from published work then indent the quotation on both left and right margins, using an italic font where practicable, and introduce the quotation with an attribution.

Further information in relation to the existing London Metropolitan University regulations concerning plagiarism can be obtained from http://www.londonmet.ac.uk/academic-regulations

Assessment

This assignment will be marked out of 100 and carries 30% of the overall module weighting.
Your .java files and report for this part must be uploaded and submitted by RTE Deadline. The assignment must be carried out individually so you must not obtain help from anyone other than the module teaching staff. You must not copy code from any source apart from the module core text and the module materials. Collusion, plagiarism (unreferenced copying), and other forms of cheating constitute Academic Misconduct, which can lead to failure of the module and suspension.
The viva will be conducted for this assignment.

Note: ***If a student would be unable to defend his/her coursework, s/he might be penalized with 50% of total coursework marks***

Aim

The aim of this assignment is to implement a real-world problem scenario using the Object-oriented concept of Java that includes creating a class to represent a **teacher**, together with its two subclasses to represent a **Lecturer** and a **Tutor** respectively. You will also need to write a report that should contain information about your program.

Deliverables

Create a new project in **BlueJ** and create three new classes (**Teacher**, **Lecturer,** and **Tutor**) within the project. **Lecturer** and **Tutor** are **subclasses** of the class **Teacher**. When you are ready to submit your solution, upload your codes ***Teacher.java****,* ***Lecturer.java,*** *and* ***Tutor.java*** files (not any other files from the project) together with your report in pdf format.

Program (56 marks)

The program should include the following classes (with no additional attributes or methods).

1) The **Teacher** class has six attributes, which correspond to the **teacher Id, teacher name, address, working type, employment status and working hours.** The **teacher name, address, working type, employment status** are each represented as a **string of text** and **Teacher ID, and working hours** as **a number**.

   The **constructor** accepts five parameters which are**, teacher Id, teacher name, address, working type, employment status.** The attribute **teacher name** is initialized with the parameter value. Additionally, **assign teacher Id, address, working type, and employment status** with the **parameter values.**

   Each attribute has a **corresponding accessor method**.

   A method is required to **set the working hours**. The method accepts a new **working hour** as a parameter. The parameter value is then assigned to the attribute working hours.

A display method should output (suitably annotated) the **teacher Id, teacher name, address, working type, and employment status**. If the **working hours** is **not assigned, display a suitable message.**

[10 marks]

2) The **Lecturer** class is also a subclass of **Teacher** class and it has four attributes:

Department	- a String
YearsOfExperience	- an integer
gradedScore	- an integer
hasGraded	- either true or false (boolean)

The **constructor** accepts **seven** parameters which are **teacher Id, teacher name, address, working type, employment status, department and YearsOfExperience.** A call is made to the superclass constructor with five parameters and a setter method. Also, assign, **gradedScore as 0(zero)** and

YearsOfExperience with the **corresponding parameter values**.

In the constructor assign the **attribute: hasGraded** to **false**.

Each attribute has a corresponding **accessor method.**

Create a **mutator method** for attribute: **gradedScore**.

There is a method named **gradeAssignment**. The method is used to grade assignments of students who have submitted their assignments on time. The method accepts **gradedScore**, **department** and **YearsOfExperience**. If the **yearsOfExperience** is **higher than or equal to five years**, and department is also relevant to the department with **same** area of interest, then the lecturer will grade the assignments of students according to:

A ----> 70 and above

B ----> 60 and above

C----> 50 and above

D-----> 40 and above

E-----> Less than 40

Now, the attribute **hasGraded** is set to **true**. If the lecturer has not graded yet, then a suitable message should be displayed.

A method to **display** the details of the **Lecture** is required. It must have the same signature as the display method in the **Teacher** class. It will call the method in the **Teacher** class to display the **teacher Id, teacher name, address, working type, working hours, and employment status**. It should also display a department, YearsOfExperience and gradedScore. If the **score has not** been graded yet, **display** suitable message. Each output must be suitably annotated.

[16 marks]

3) The **Tutor** class is a **subclass** of **Teacher** class and has five attributes:

salary	- a double
specialization	- a String
academic qualifications	- a String
performanceIndex	- an Integer
isCertified	- a boolean

The constructor accepts ten parameters which are **teacher Id, teacher name, address, working type, employment status, working hours, salary, specialization, academic qualifications** and **performanceIndex.** A call is made to the **superclass constructor** with five parameters and a setter method. Additionally, in the constructor, assign salary, specialization, academic qualifications, performanceIndex with the **parameter values**. The attribute: isCertified is set to **false**.

Each attribute has a corresponding **accessor method.**

A method is required to **set** the salary as each tutor can have different salaries. The method accepts a **new salary**, and **new performanceIndex** as a parameter and, if the performanceIndex is **more than five**(5) and the **working hour** of that tutor is greater than **twenty**(20), then calculate the salary as:

performanceIndex --------->	appraisal
5-7	5%
8-9	10%
10	20%

[**Note**: new salary can be calculated as: **salary + appraisal% of salary**]
Likewise, the status of isCertified is then set to **true** after appraisal. If the tutor has not been certified yet, then a suitable message is output to the user indicating that the salary cannot be approved.

There is a method named **removeTutor.** This method will remove the tutor (only if the tutor has not been certified yet). The attributes **salary, specialization, academic qualifications and performance index is set to zero.** The attribute **isCertified** is then set to **false**.

A method to **display** the details of the **Tutor** class is required. It must have the same signature as the display method in the **Teacher** class. If **isCertified** is set to **false**, It will call the method in the **Teacher** class to display the details**.** However, if **isCertified** is set to true, **salary, specialization, academic qualifications** and **performanceIndex** should be displayed along with details of parent class**.** Each output must be suitably annotated.

[18 marks]

Additional marks will be awarded for good programming styles, particularly naming, layout and comments.
See http://www.bluej.org/objects-first/styleguide.html for details.

[12 marks]

Report (44 marks)

Your report should describe the process of development of your classes with:

a. A class diagram **[5 marks]**

b. Pseudocode for each class **[10 marks]**

c. A short description of what each method does **[5 marks]**

d. You should give evidence (through inspection tables and appropriate screenshots) of the following testing that you carried out on your program:
Test 1: Inspect the Lecturer class, grade the assignment, and re-inspect the Lecturer Class **[3 marks]**
Test 2: Inspect Tutor class, set salary and reinspect the Tutor class **[4 marks]**
Test 3: Inspect Tutor class again after removing the tutor. **[2 marks]**
Test 4: Display the details of Lecturer and Tutor classes. **[4 marks]**

e. The report should contain a section on error detection and error correction where you give examples and evidence of three errors encountered in your implementation. The errors (syntax, semantic or logical errors) should be distinctive and not of the same type. **[3 marks]**

f. The report should contain a conclusion, where you need to include the following things:
   - Evaluation of your work,
   - Reflection on what you learned from the assignment,
   - What difficulties do you encounter and
   - How you overcame the difficulties.

[**4 marks]**

The report should include a title page (including your name and ID number), a table

of contents (with page numbers), an introduction part that contains a brief about your work, and a listing of the code (in an appendix). Marks will also be awarded for the quality of writing and the presentation of the report.

[4 marks]

Viva

Note: ***If a student would be unable to defend through VIVA his/her coursework, s/he might be penalized with 50% of total coursework marks.***

Marking Scheme

Marking criteria		Marks
A.	**Coding Part**	**56 Marks**
	1. Creating Teacher Class 2. Creating Lecturer Class 3. Creating Tutor Class 4. Program Style	10 Marks 16 Marks 18 Marks 12 Marks
B.	**Report Structure and Format**	**44 Marks**
	1. Class Diagram 2. Pseudocode 3. Method Description 4. Test-1 5. Test-2 6. Test-3 7. Test-4 8. Error Detection and Correction 9. Conclusion 10. Overall Report Presentation/Formatting	5 Marks 10 Marks 5 Marks 3 Marks 4 Marks 2 Marks 4 Marks 3 Marks 4 Marks 4 Marks
	Total	**100 Marks**