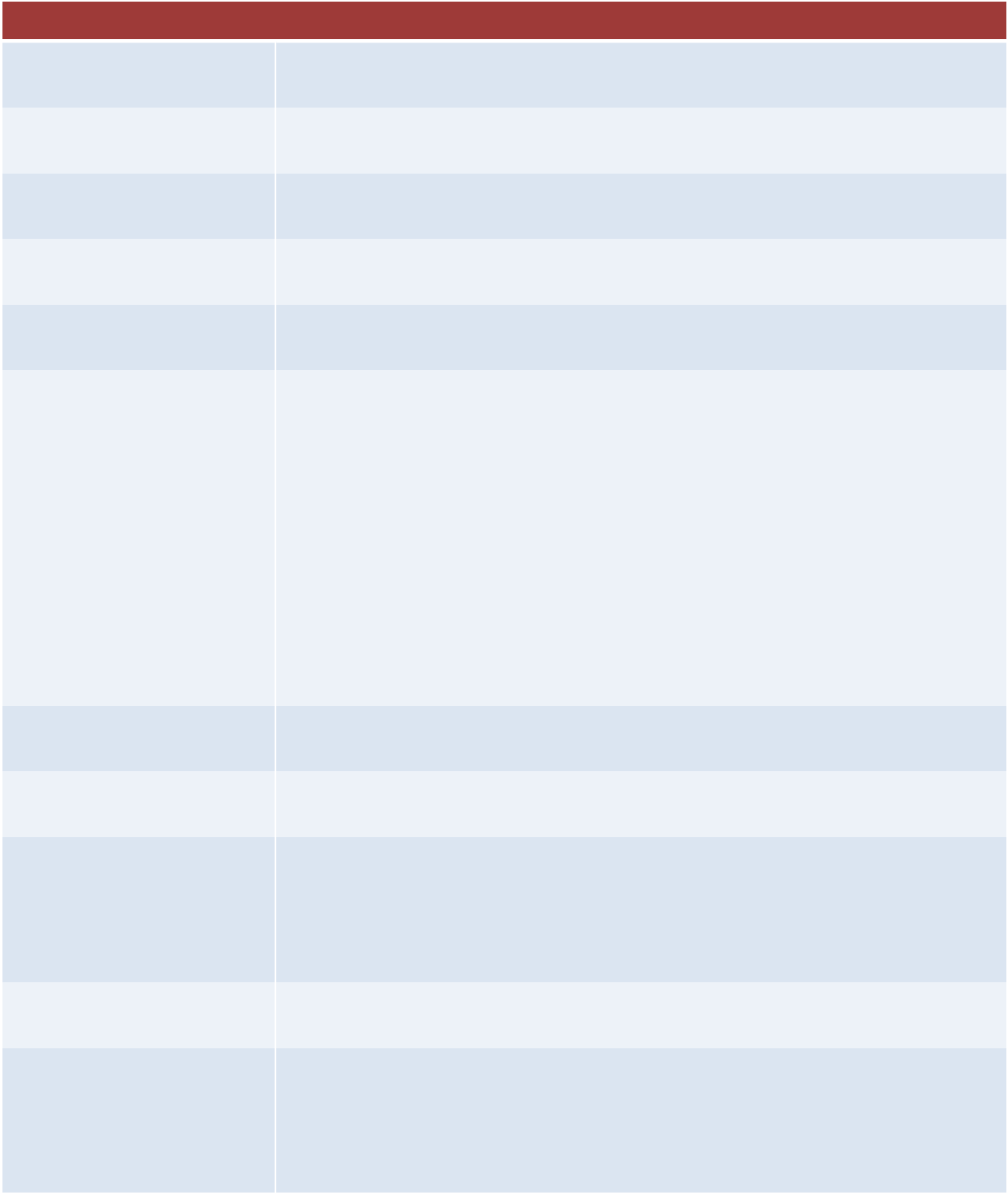
**University** **of** **Westminster**

School of Computer Science & Engineering

**5COSC001W**

Module leader

Unit

Weighting:

Qualifying mark

Description

**Object** **Oriented** **Programming** **–** **Coursework** **1** **(2019/20)**

Guhanathan P

Coursework 1

50%

30%

Object Oriented Programming and Design

Learning Outcomes Covered in this Assignment:

This assignment contributes towards the following Learning Outcomes (LOs):

- LO1 Identify and justify good practices in the development of object

oriented software;

- LO2 Apply acquired knowledge of concepts, characteristics, tools and environments to adapt to new computational environments and programming languages which are based on object oriented principles;

- LO3 Design, implement efficiently applications based on a OOP language, given a set of functional requirements.

- LO4 Implement GUI interfaces using an OOP language;

Handed Out:

Due Date

Expected deliverables

Method of Submission:

Type of Feedback and Due Date:

15th October 2019

Monday 2nd December 2019 Submissions by 13:00

Submit on Blackboard a zip file containing:

**A** **folder** **with** **all** **the** **UML** **documents** **and** **test** **case** **plan** **attached**

**A** **folder** **with** **the** **developed** **project** **(IntelliJ** **Solution** **with** **your** **Java** **code)** **A** **report** **containing** **all** **diagram** **and** **code** **you** **wrote** **for** **each** **of** **the**

**requirements** **given** **below** **to** **turnitin** **(should** **cover** **everything** **in** **the** **marking** **scheme)**

Electronic submission on BB via a provided link close to the submission time.

Written feedback within 15 working days and generic feedback during the demonstration.

**Assessment** **regulations**

Refer to section 4 of the “How you study” guide for undergraduate students for a clarification of how you are assessed, penalties and late submissions, what constitutes plagiarism etc.

**Penalty** **for** **Late** **Submission**

If you submit your coursework late but within 24 hours or one working day of the specified deadline, 10 marks will be deducted from the final mark, as a penalty for late submission, except for work which obtains a mark in the range 40 – 49%, in which case the mark will be capped at the pass mark (40%). If you submit your coursework more than 24 hours or more than one working day after the specified deadline you will be given a mark of zero for the work in question unless a claim of Mitigating Circumstances has been submitted and accepted as valid.

It is recognised that on occasion, illness or a personal crisis can mean that you fail to submit a piece of work on time. In such cases you must inform the Campus Office in writing on a mitigating circumstances form, giving the reason for your late or non-submission. You must provide relevant documentary evidence with the form. This information will be reported to the relevant Assessment Board that will decide whether the mark of zero shall stand. For more detailed information regarding University Assessment Regulations, please refer to the following website:**http://www.westminster.ac.uk/study/current-students/resources/academic-regulations**

**Coursework** **Description**

**Objectives**:

The aim of this coursework is to assess the knowledge and skills that you have acquired about object- oriented programming during the module. You are asked to implement a program in which objects interact in order to fulfil with a set of functional requirements.

**Analyse** **the** **problem** **statement**:

An important skill that you will start to develop in this module is analysing a problem statement in order to identify the details needed to develop a solution.

In this assignment, the first task you should perform is a careful analysis of the problem statement in order to make sure you have all the information to elaborate a solution.

**Design** **a** **solution**

The design of your system should be consistent with the Object Oriented principles and easy to understand by an independent programmer*.*

You are required to design your program using UML diagrams. In particular you have to draw:  A use cases for the system *(6* *marks)*.

 A class diagram *(6* *marks)*

**Problem** **description** **and** **requirement** **statement**

You are required to develop a program that implements a basic vehicle rental system.

You should implement a **console** **system** from where the manager can add new vehicles, delete if needed, edit the vehicle information, etc. (as described in detailed below).

You should implement a **Graphical** **User** **Interface** (GUI) from where a customer can see the list of vehicles and book available vehicles for specific dates.

In this assignment, you will be required to implement the following functionality*:*

1. According to the *Inheritance* principle you have to design and implement a super class **Vehicle** *(6* *marks)* and the subclasses **Car** and **Motorbike**. The classes should include appropriate methods in order to comply with the *encapsulation* *principle* and hold information about the *Plate* *number*, and the *Make* (You can add any other information that you consider appropriate for a rental vehicle system and you can implement additional classes with justification to make the code more robust or user friendly).

In particular:

The **Car** class should hold specific information and methods. You should add at least 2 instance variables (attributes that you believe are important to be held in this class) and the relative get/set methods *(5* *marks)*.

The **Motorbike** class should hold specific information and methods. You should add at least 2 instance variables (attributes) and the relative get/set methods *(5* *marks)*.

You should implement a class **Schedule** to represent the time slot when a vehicle has been booked. The class should have a pick-up date and a drop-off date (to represent the date you can use either the class provided during tutorials or you can use any java API). *(5* *marks)*.

2. Design and implement a class called **WestminsterRentalVehicleManager**, which implements the interface **RentalVehicleManager** *(2* *marks).* WestminsterRentalVehicleManager maintains the list of the vehicle to rent and provides all the methods for the system manager.

The class should display **in** **the** **console** **a** **menu** containing the following management actions from which the manager can select one.

 **Add** **a** **new** **vehicle** in the rental system. It should be possible to add a new vehicle, either a car or a motorbike with all the relevant information. You should consider that in the rental car park there are max 50 parking lots *(5* *marks).*

 **Delete** **a** **vehicle** from the system, selecting the plate number. Display a message with the information of the vehicle that has been deleted (if it is a car or motorbike) and the number of available parking lots in the garage *(5* *marks)*.

 **Print** **the** **list** **of** **the** **vehicles** in the system. For each vehicle, print the plate number, the type of vehicle (if is a car, a van or a motorbike). The list should be ordered alphabetically according to the vehicle make *(5* *marks)*.

 **Write/Save** the vehicle stock list in a file after any changes (5 marks).

**Create** **a** **Graphical** **User** **Interface** (GUI) that can be opened selecting an option from the menu console *(?* *marks).*

**You** **should** implement the GUI according the following specification:

 The user can visualise the list of vehicles in WestminsterRentalVehicle system *(8* *marks)*.  The user can filter the vehicles according at least one parameter (e.g. the type of the

vehicle, or the engine size, or the make, etc.) *(4* *marks).*

 The user can check the availability of the vehicle in specific dates and can book make a reservation if it is available *(8* *marks)*.

Note: You can choose how the GUI should look like and how to meet at the best these specifications.

3. Testing and system validation:

 Write a test plan designed to ensure that the coded solution works as expected. The test plan will include specific instructions about the data and conditions the program will be tested with *(5marks).*

 Implement an automated testing (you can use JUnit or feel free to use any other tool or scripts for unit testing) that runs scenarios of each of the use cases you implemented in the console menu *(10* *marks)*.

 The following will be evaluated:

o The robustness of the code through the use of error handling and input validation *(5marks).* o The quality of the code and the adherence to coding standards and conventions *(5* *marks).*

**Demonstration**

You will be expected to be familiar with your code and required to make any ad-hoc changes during the demonstration

**Please** **note** **that** **no** **marks** **will** **be** **given** **to** **a** **task** **if** **there** **is** **no** **understanding** **of** **the** **code**.

**Note** **that** **the** **maximum** **mark** **for** **work** **marked** **without** **a** **viva/demonstration** **shall** **range** **from** **1-30%** **only.**

**Coursework Guidance**

The implementation goals are split into 4 phases:

**Phase** **I** This involves an initial design of your system through UML diagrams and the implementation of the classes Vehicle, Car, Motorbike and Schedule (plus any accessory class you might need), and the Interface RentalVehicleManager. The marking scheme allows for students to score up to 35%. I would suggest concluding this Phase 1 by Week 05.

**Phase** **II** This is for implementing the system functionalities through a console menu. You should provide the implementation (in the class WestminsterRentalVehicleManager) of methods to add, delete, print, sort vehicles and generate report. The marking scheme allows for students to score up to 55%. I would suggest concluding this phase 2 by Week 07.

**Phase** **III** This phase involves the GUI implementation and the possibility for a user to visualise and book a vehicle. The marking scheme allows for students to score up to 75%. I would suggest concluding this phase 3 by Week 09.

**Phase** **IV** This phase is dedicated to the testing and evaluation of your system and the marking scheme allows for students to score up 100% for completing all the phases. This Phase 4 needs to be concluded by week 11 (week of final submission).