FACULTY OF ENGINEERING AND INFORMATION SCIENCES

SUBJECT'S INFORMATION:			
Subject:	CSIT121 Object-Oriented Design and Programming		
Session:	Autumn 2019 (February)		
Programme / Section:	Computer Science and IT		
Lecturer:	Ms. Siti Hawa		
Coursework Type *(tick appropriate box)*	✓ Individual Assignment ❑ Group Assignment ❑ Project ❑ Lab Task ❑ Seminar / Tutorial Paper ❑ Others		
Coursework Title:	**Assignment 1**	Coursework Percentage:	10%

ASSESSMENT CRITERIA:		
Correctness	The application should produce the correct result as stated in the specification.	2 marks
Class design and implementation	Design of class that follows the specification provided. Class relationship is implemented. Use of class variables for shared attributes.	3 marks
Main application	Correct implementation of the main method using appropriate control structures.	3 marks
Readability and Documentation	Appropriate comments are included. Meaningful identifiers used. Proper indentation and line spacing used.	1 mark
Well formatted output	Output should be well formatted with appropriate messages displayed and easy to understand.	1 mark

SUBMISSION:

All completed work should be submitted online through Moodle before or on the due date provided.

SUBMIT AS EARLY AS POSSIBLE. ONLY ONE SUBMISSION IS ALLOWED. IF RE-SUBMISSION IS NECESSARY, YOU ARE REQUIRED TO REMOVE THE EARLIER SUBMISSION AND THIS MUST BE DONE BEFORE THE DUE DATE. OTHERWISE YOU WILL BE PENALIZED FOR LATE SUBMISSON.

DUE DATE:	**Latest by Wednesday, 24th April 2019 (11:55 pm)**

PENALTIES FOR LATE SUBMISSION:

Penalties apply to all late work, except if student academic consideration has been granted. Late submissions will attract a penalty of 25% of the assessment mark per day including the weekend. Work more than (3) days late will be awarded a mark of zero.

PLAGIARISM:

When you submit an assessment task, you are declaring the following

1. It is your own work and you did not collaborate with or copy from others.
2. You have read and understand your responsibilities under the University of Wollongong's policy on plagiarism.
3. You have not plagiarised from published work (including the internet). Where you have used the work from others, you have referenced it in the text and provided a reference list at the end ot the assignment.

Plagiarism will not be tolerated. Students are responsible for submitting original work for assessment, without plagiarising or cheating, abiding by the University's policies on Plagiarism as set out in the University Handbook

under University Policy Directory and in Faculty handbooks and subject guides. under University Policy Directory and in Faculty handbooks and subject guides.

COURSEWORK SPECIFICATION

OBJECTIVES

This assignment aims to provide you with some experience in writing codes using Java programming language that covers the following topics:

- Classes and Objects
- Interaction between Classes

Remember that:
1. **All programs should be able to run on the lab's computers.**
2. **You must put the following information on the header of each text and source file you will be submitting in this assignment:**
   Student's full name:
   Student's ID:
   Modification Date:
   Purpose of this file (or program):
3. **Assignments that are not able to be compiled will result in zero mark given to the assignment.**
4. **You must only use the Java features that have already been covered in the lectures**

TASKS:

This assignment will require you to design a set of classes that work together to simulate a police officer issuing a parking ticket. You are then required to write a Java application to demonstrate how the objects from these class interact. The description of each classes are given below.

The `ParkedCar` Class

This class is used to simulate a parked car at a parking lot. The class should contain the following attributes:

- The car's make (example: Honda)
- The car's model (example: Accord)
- The color of the car
- The car's registration number
- The number of minutes the car has been parked

The class should have a constructor to initialize the make, model, color, and registration number with the values passed as parameters. The number of minutes should start with zero (0). There should be a set method to set number of minutes and a get method to retrieve the number of minutes.

The `ParkingMeter` Class

This class is used to simulate a parking meter. The class's only responsibility is to set and know the number of minutes of parking that has been purchased by a car parked at the lot. The class will contain only one attribute:

- The number of minutes of parking time purchased

Include a default constructor to initialize the time to zero (0), a method to set the minutes purchased, and a method to retrieve the minutes purchased.

The `ParkingTicket` Class

This class is used to simulate a parking ticket issued by a police officer for a car that has parked illegally. The class should keep the details of:

- The illegally parked car. This should be declared as a `ParkedCar` object you have declared above.
- The amount of fine.
- The police officer who issue this parking ticket. This should be declared as a `PoliceOfficer` object (details below)
- A public class variable to represent the fine rate for the first hour and is initialized to RM150.00, a public class variable to represent the fine rate for additional hour and is initialized to RM50.00, and a public class variable to represent the maximum fine rate and is initialized to RM300.00.

The class should have a default constructor, a set method to assign a `ParkedCar` object, a set method to assign the `PoliceOfficer` object, a method to show the details of the `ParkingTicket` (which includes the `ParkedCar` details, the `PoliceOfficer` details and the amount of fine), and a method to calculate the amount of fine.

The method to calculate the amount of fine should take as parameter the number of minutes of parking time purchased by the car which is taken from the `ParkingMeter` object. The fine is calculated by first calculating the number of minutes the car is parked beyond the parking time purchased. The fine is RM150.00 for the first hour or part of an hour that the car is illegally parked, plus RM50.00 for every additional hour or part of an hour that the car is illegally parked. The maximum fine should be RM300.00.

The `PoliceOfficer` Class

The `PoliceOfficer` class is used to simulate a police officer inspecting parked cars. The police officer will issue a parking ticket if a parked car is found to be illegally parked at a parking lot. This class keeps the following details about a police officer.

- The police officer's name
- The police officer's badge number

The class should implement a suitable constructor, set methods, and get methods.

The main application

Write another class to be the main application that simulate the interaction between all the classes above. Your application should simulate the following activities:

- A police officer sign up for duty. This will involve creating a `PoliceOfficer` object. If a ticket is issued when this officer is on duty, the details will be stated on the parking ticket issued. You may have more than one police officer but only one should be on duty at a time in this simulation.

- Creating `ParkedCar` objects. This will involve creating several `ParkedCar` objects. You should make some of them are illegally parked and some are not. Use an array named `parkingLot` to keep all your `ParkedCar` objects. For each of the `ParkedCar` object, there should be a `ParkingMeter` object created to simulate the purchasing of the parking time. The `ParkingMeter` objects should be kept in another array named `meters`. The index of the arrays will be used to simulate parking lot and its parking meter. For example, if a `ParkedCar` object is stored at `parkingLot[3]` then its `ParkingMeter` will be stored in `meters[3]`.

- Inspecting `ParkedCar` objects and `ParkingMeter` objects. This is where the police officer will inspect all cars that are parked to check if they are illegally parked. A `ParkingTicket` will be issued (a `ParkingTicket` object is created) if a `ParkedCar` is found to be illegally parked. A `ParkedCar` is illegally parked when the parking time purchased by the car has expired. All `ParkingTicket` objects should be stored in an `ArrayList`.

- Displaying all `ParkingTicket` objects. This will display a list of all `ParkingTicket` objects that have issued during the simulation.

Remember to include suitable exceptions handling in your program, write meaningful comments, and use meaningful variables and methods names. Take an effort to make your output presentable and easy to understand.

Assumptions:

In a typical situation, a parking meter will usually show the time remaining and as time passes, the time shown on the meter will decrease. For the purpose of this simulation, you are not required to show this. You may assume that the value entered for the `ParkingMeter` class is showing the exact number of minutes of parking time purchased.

This is also the same with the number of minutes the car has been parked in the `Car` class. In a usual situation, this should be increasing. However, for the purpose of this simulation, you may assume that the value stored in the variable is showing the number of minutes the car has been parked as of that time.
