Java Coding Task – Assignment – Home Activity on Spring Tools 4 for Eclipse IDE.

Dear Students,

Task 01.a: **Download and Install Spring Tool Suite (Spring Tools 4 for Eclipse) IDE?**

Spring Tool Suite (STS) is a Java IDE tailored for developing Spring-based enterprise applications. It is easier, faster, and more convenient. And most importantly it is based on Eclipse IDE. STS is free, open-source, and powered by VMware. Spring Tools 4 is the next generation of Spring tooling for your favorite coding environment. Largely rebuilt from scratch, it provides world-class support for developing Spring-based enterprise applications, whether you prefer Eclipse, Visual Studio Code, or Theia IDE.

Follow this URL:

https://www.geeksforgeeks.org/how-to-download-and-install-spring-tool-suite-spring-tools-4-for-eclipse-ide/

Task 01.b
Your task is to design and implement a **console-based Management Information System (MIS) for a car rental company**. The system should efficiently manage different types of cars, renters, and the rental transactions between them. The program must be written in **Java** and should showcase your understanding of **Object-Oriented Programming (OOP) principles such as encapsulation, abstraction, polymorphism, inheritance, composition and aggregation.** Also follow the good coding conventions specified in **Google JAVA style.**

- This task is **Individual**.

Read below the document and cover the assigned functionality.

Assignment: Console-Based Car Rental Management System (CRMS)

Your task is to design and implement a console-based Management Information System (MIS) for a car rental company. The system should efficiently manage different types of cars, renters, and the rental transactions between them. The program must be written in Java and should showcase your understanding of Object-Oriented Programming (OOP) principles such as encapsulation, abstraction, polymorphism, inheritance, composition and aggregation.

Car Types:

Properties: Car ID, Brand, Model, Year, Rental Status, Rental Fee, Plate number

Compact Car:

- Features: Basic features, suitable for short-distance travel.

- Rent Calculation Formula: Base rent + distance traveled cost.
- Insurable: No.

SUV:

- Features: Spacious, suitable for family trips.
- Rent Calculation Formula: Base rent + distance traveled cost.
- Insurable: Yes. (Insurance cost: Fixed percentage of the base rent; Damage Cost: Percentage of the base rent)

Luxury Car:

- Features: High-end, suitable for special occasions.
- Rent Calculation Formula: Base rent + distance traveled cost + luxury tax.
- Insurable: Yes. (Insurance cost: Fixed percentage of the base rent; Damage Cost: Percentage of the total cost including luxury tax)

Renter Types:

Properties: Renter ID, Name, Email, Rented Cars (List of Cars), Total Rental Fee, phone number, address

Regular Renter:

- Features: Standard rent rates.

Frequent Renter:

- Features: Loyalty program with discounted rates.

Corporate Renter:

- Features: Special rates for corporate clients.

Functionalities of CRMS

Car Management:

Add new cars of different types.

Display available cars.

Remove a car if it is not rented.

Renter Management:

Add new renters of different types.

Display renter details.

Remove a renter after they return the car.

Rent Transactions:

Rent a car to a renter.

Display rental details.

Calculate and display the total rental cost.

Provide an option to add insurance if the rented car is insurable.

If insurance is added, calculate and include insurance cost in the total.

Calculate and display damage cost based on the car type, and insurance status upon return.

All rent transactions are stored in CRMS.

Damage Cost Calculation:

For all Cars: Damage cost is a percentage of the total cost decided by the company and damage if un-insured.

For Insured Cars: Damage cost is a percentage of the total cost minus the insurance but with a minimum set if damage is apparent.

Constraints:

Implement encapsulation to protect the internal state of classes.

Use abstraction to hide unnecessary details.

Utilize polymorphism for variations in different calculations

Implement inheritance where appropriate, such as creating a base class and/or interface where applicable

Identify static and final variables and use them appropriately.

Submission Guidelines:

Submit a well-documented Java program adhering to OOP principles.

Follow the good Coding Conventions specified by Google JAVA style

Provide sample test cases demonstrating the program's functionality. Provide junit test cases for all developed classes and the requested functionalities.

Include comments explaining the logic and design decisions.

This assignment assesses your OOP proficiency and your ability to design a robust software system. Feel free to seek clarification on the requirements. Good luck!

SDA Assignment - 01 (Rubric)

Name: _____ Total Marks: /70 Roll No: _____

Evaluation Rubric:

Sr No	Evaluation Criteria	Marks	Grade Yourself / Obtain
0	**Download and Install Spring Tool Suite (Spring Tools 4 for Eclipse) IDE**	10	
1	**Inheritance** • Identified Correct classes (Parent/child). • Variables (Static & Final) / Functions.	8	
2	**Composition & Aggregation** • Correct Association between classes.	5	
3	**Abstraction** • Identified Abstract classes. • Implementation of Data hiding functions.	7	
4	**Encapsulation**	7	

	• How, sensitive information of a class is accessible to another or itself class		
5	**Polymorphism** • Usage of function, which works for different sceniro.	7	
6	**Design of Logic** • Correctly implemented and usage of OOP principles.	6	
7	**Program Execution** • How the Main menu is organized, the console output is user friendly. • The Program should follow the smooth execution of all demo questions.	6	
8	**Correct Output** • Functionality of system must be correct. • Works on all Possible Sceniro according to the case study.	10	
9	**Testcases** • Implement checks on Input values as testcases of each functionality.	10	
10	**Java Syntax & Code Comments** • Added comments for each functionality and usage of IDE for programming.	2	

Total	**80**	

Any Comments: _____