**OOP**

**Car Rental Management System Documentation**

**Overview:**

The Car Rental Management System is designed to facilitate the management of a car rental business. It allows admin to perform various tasks such as adding cars, managing employees, and viewing bookings. It allows Users to view available cars for booking and book cars.

**Object-Oriented Programming Concepts Used**

1. **Abstraction**
   * **Classes**: Person, Employee, Customer, Car, Order
   * **Description**: Abstracts the details of different entities involved in the car rental system, providing a simplified representation for use in the system.
2. **Inheritance**
   * **Hierarchy**:
     + Person (base class)
       - Employee (inherits from Person)
         * Admin, Mechanic, SecurityGuard (inherit from Employee)
       - Customer (inherits from Person)
   * **Description**: Allows classes to inherit fields and methods from other classes, promoting code reuse and logical hierarchy.
3. **Encapsulation**
   * **Classes and Fields**: Person (name, address), Employee (salary), Car (model, registrationNumber, rentalCost, isAvailable)
   * **Description**: Bundles the data (attributes) and methods that operate on the data into a single unit or class, and restricts access to some of the object's components.
4. **Polymorphism**
   * **Method Overriding**: Overridden methods in subclasses like Admin, Mechanic, and SecurityGuard to provide specific implementations.
   * **Description**: Allows objects to be treated as instances of their parent class rather than their actual class, enabling one interface to be used for a general class of actions.

**Classes:**

1. **Person:**
   * Abstract class representing a person with attributes name and address.
   * Subclasses: Employee, Customer
2. **Employee:**
   * Inherits from Person.
   * Represents an employee with an additional attribute salary.
   * Subclasses: Admin, Mechanic, SecurityGuard
3. **Admin:**
   * Inherits from Employee.
   * Provides administrative functionalities like adding employees, viewing cars, bookings, and employees.
4. **Mechanic, SecurityGuard:**
   * Inherit from Employee.
   * Represent different types of employees in the car rental system.
5. **Customer:**
   * Inherits from Person.
   * Represents a customer of the car rental system.
6. **Car:**
   * Represents a car available for rental with attributes like model, registrationNumber, rentalCost, isAvailable.
7. **Order:**
   * Represents a booking order with attributes car, customer, startDate, endDate.
8. **CarRentalManagementSystem:**
   * Main class managing the car rental system.
   * Provides functionalities for adding cars, booking cars, managing employees, and file handling.
9. **mainWindow:**
   * Represents the main window GUI for the car rental system.
   * Provides options for the user to login as Admin or Customer or exit the system.
10. **CustomerMenuWindow:**
    * GUI for the customer menu.
    * Provides options for customers to view available cars and book cars.
11. **AdminMenuWindow:**
    * GUI for the admin menu.
    * Provides options for admins to add cars, view cars, bookings, employees, and logout.
12. **DisplayAvailableCarsWindow, DisplayCarsWindow, DisplayBookingsWindow, DisplayEmployeesWindow:**
    * GUI windows for displaying available cars, all cars, all bookings, and all employees respectively.
13. **AddCarWindow, AddEmployeeWindow:**
    * GUI windows for adding cars and employees respectively.
14. **logingWindow:**
    * GUI window for admin login authentication.
15. **ManagementSystem:**
    * Main class to run the application.

**File Handling:**

* The system utilizes serialization to store and retrieve objects (cars, employees, orders) from files.
* Custom MyObjectOutputStream class is used to avoid header issues when appending to existing files.
* Each object is written to separate .ser files (cars.ser, employees.ser, orders.ser).

**GUI:**

* GUI is implemented using Java Swing for user interaction.
* Different windows are provided for different functionalities like admin login, admin menu, customer menu, etc.

**Functionality:**

* Users (admins/customers) can interact with the system through the GUI to perform various tasks related to car rental management.
* Admins can add cars, add employees, view cars, bookings, and employees.
* Customers can view available cars and book cars.
* Password-protected admin login is implemented for security.
* Error handling is done for invalid input and file operations.

**GUI and Code Snippets:**

