Classes and Objects Problem Statements

1. **Bank Account Management System**

- Create a class `BankAccount` with private attributes such as `accountNumber`, `balance`, and `accountHolderName`. Implement methods to deposit, withdraw, and check the balance. Additionally, implement a method that charges a fee for each transaction and ensures that the withdrawal does not exceed the balance.

2. **Library Management System**

- Design a `Book` class with attributes like `bookTitle`, `authorName`, `publicationYear`, and `isAvailable`. Create another class `Library` to manage a collection of books. The `Library` class should have methods to add a book, remove a book, check if a book is available, and search for a book by title or author.

3. **Student Grade Management System**

- Create a `Student` class with attributes like `studentName`, `rollNo`, `marks[]` (for multiple subjects). Implement methods to input student marks, calculate the average grade, and display the result (grade A, B, C, etc., based on average marks). Use multiple classes for input handling, result generation, and grade classification.

4. **Movie Ticket Reservation System**

- Implement a `MovieTicket` class with attributes like `movieName`, `movieDuration`, `availableSeats`, `ticketPrice`, and `screenNumber`. Add methods for reserving tickets, displaying available seats, and calculating the total cost based on the number of tickets reserved.

5. **Employee Payroll System**

- Create a class `Employee` with attributes like `employeeName`, `employeeId`, `salary`, and `designation`. Implement methods to calculate the annual salary, add bonuses, deduct taxes, and display the final payroll details. Use inheritance to create subclasses like `Manager` and `Developer` with specific attributes like `bonusPercentage`.

6. **E-commerce Cart System**

- Design an `Item` class that includes attributes like `itemName`, `itemPrice`, and `itemQuantity`. Create an `ShoppingCart` class that can store a list of items, allow adding/removing items, and calculate the total price, including any discounts applied to the cart. Implement methods for calculating final prices after applying discounts or tax.

7. **Car Rental System**

- Implement a `Car` class with attributes like `carModel`, `rentalPricePerDay`, and `isAvailable`. Create a `Customer` class that can rent a car for a specified number of days. The `Customer` class should calculate the total rental cost and display the details. Additionally, handle availability status and return the car after the rental period.

8. **Traffic Light System**

- Create a class `TrafficLight` with an attribute `currentColor`. Implement methods like `changeColor()` (to switch between Red, Yellow, Green), `displayColor()` (to print the current color), and `isSafeToGo()` (to return a boolean based on the current color). Allow the light to follow a predefined sequence: Red -> Green -> Yellow -> Red.
