Logic Building OOPs using JAVA

Contents

Question 1: Employee ID Generation

Scenario:

You need to generate unique employee IDs for a company. The ID generation process uses the employee's Department, FirstName, LastName, and a RandomNumber. The steps to generate the employee ID are as follows:

Step 1: Compare the lengths of FirstName and LastName. The shorter string will be called ShorterString, and the longer one will be called LongerString. If they are of equal length, the one that appears first alphabetically is ShorterString.

Step 2: The employee ID should be generated as follows:

- First letter of ShorterString + Entire Department + Last letter of LongerString + RandomNumber digits at positions X and X+1.

Implement the method `String generateEmployeeID(String department, String firstName, String lastName, int randomNumber, int x)` to accomplish this.

Test Cases:

1. Input: ("HR", "John", "Smith", 654321, 3)

   Output: "JHRS612"

2. Input: ("Finance", "Anna", "Taylor", 987654, 2)

   Output: "AFinanceR76"

3. Input: ("Engineering", "Michael", "Brown", 123456, 1)

   Output: "MEngineeringB23"

4. Input: ("Marketing", "Linda", "White", 567890, 4)

   Output: "LMarketingW90"

5. Input: ("Sales", "James", "Bond", 345678, 2)

   Output: "JSalesD45"

6. Input: ("IT", "Eve", "Adams", 876543, 3)

   Output: "EITs543"

7. Input: ("Support", "Chris", "Jones", 234567, 1)

   Output: "CSupportJ34"

8. Input: ("Admin", "Peter", "Parker", 123987, 2)

   Output: "PAdminR39"

9. Input: ("Logistics", "Alice", "Cooper", 567123, 4)

   Output: "ALogisticsR23"

10. Input: ("Development", "Nancy", "Hall", 678912, 2)

    Output: "NDevelopmentL89"

Question 2: Order ID Generation

Scenario:

You need to generate unique order IDs for a shopping platform. The ID generation process uses the order's Category, CustomerName, Product, and an OrderNumber. The steps to generate the order ID are as follows:

Step 1: Compare the lengths of CustomerName and Product. The shorter string will be called ShorterString, and the longer one will be called LongerString. If they are of equal length, the one that appears first alphabetically is ShorterString.

Step 2: The order ID should be generated as follows:

- Last letter of ShorterString + Entire Category + First letter of LongerString + OrderNumber digits at positions M and M+1.

Implement the method `String generateOrderID(String category, String customerName, String product, int orderNumber, int m)` to accomplish this.

Test Cases:

1. Input: ("Books", "Alice", "Novel", 345678, 2)

   Output: "eBooksN56"

2. Input: ("Electronics", "Bob", "TV", 876543, 3)

   Output: "bElectronicsT543"

3. Input: ("Furniture", "Clara", "Sofa", 123456, 1)

   Output: "aFurnitureS23"

4. Input: ("Toys", "David", "Car", 567890, 4)

   Output: "dToysC90"

5. Input: ("Clothing", "Eva", "Shirt", 987654, 2)

   Output: "aClothingS76"

6. Input: ("Appliances", "Frank", "Blender", 654321, 3)

   Output: "kAppliancesB321"

7. Input: ("Gadgets", "Gina", "Watch", 345987, 2)

   Output: "aGadgetsW59"

8. Input: ("Beauty", "Hank", "Cream", 456789, 1)

   Output: "kBeautyC56"

9. Input: ("Stationery", "Ivy", "Pen", 987123, 4)

   Output: "yStationeryP12"

10. Input: ("Gardening", "Jack", "Seeds", 678912, 2)

    Output: "kGardeningS89"

Question 3: Membership ID Generation

Scenario:

You need to generate unique membership IDs for a club. The ID generation process uses the member's ClubName, FirstName, LastName, and a MemberNumber. The steps to generate the membership ID are as follows:

Step 1: Compare the lengths of FirstName and LastName. The shorter string will be called ShorterString, and the longer one will be called LongerString. If they are of equal length, the one that appears first alphabetically is ShorterString.

Step 2: The membership ID should be generated as follows:

- First letter of ShorterString + Entire ClubName + Last letter of LongerString + MemberNumber digits at positions P and P+1.

Implement the method `String generateMembershipID(String clubName, String firstName, String lastName, int memberNumber, int p)` to accomplish this.

Test Cases:

1. Input: ("ChessClub", "John", "Doe", 123456, 2)

   Output: "JChessClubD34"

2. Input: ("BookClub", "Anna", "Smith", 654321, 3)

   Output: "ABookClubS321"

3. Input: ("FitnessClub", "Mike", "Jones", 789012, 1)

   Output: "MFitnessClubJ89"

4. Input: ("ArtClub", "Lisa", "Brown", 456789, 4)

   Output: "LArtClubB89"

5. Input: ("MusicClub", "Tom", "White", 987654, 2)

   Output: "TMusicClubW76"

6. Input: ("DanceClub", "Eva", "Black", 345678, 3)

   Output: "EDanceClubB567"

7. Input: ("CookingClub", "Chris", "Green", 876543, 2)

   Output: "CCookingClubG54"

8. Input: ("DramaClub", "Alice", "Red", 234567, 1)

   Output: "ADramaClubR34"

9. Input: ("ScienceClub", "Peter", "Blue", 567123, 4)

   Output: "PScienceClubE23"

10. Input: ("NatureClub", "Nancy", "Gray", 678912, 2)

    Output: "NNatureClubY89"

Question 4: Reservation Code Generation

Scenario:

You need to generate unique reservation codes for a travel agency. The code generation process uses the Destination, CustomerName, TravelDate, and a ReservationNumber. The steps to generate the reservation code are as follows:

Step 1: Compare the lengths of CustomerName and TravelDate. The shorter string will be called ShorterString, and the longer one will be called LongerString. If they are of equal length, the one that appears first alphabetically is ShorterString.

Step 2: The reservation code should be generated as follows:

- Last letter of ShorterString + Entire Destination + First letter of LongerString + ReservationNumber digits at positions R and R+1.

Implement the method `String generateReservationCode(String destination, String customerName, String travelDate, int reservationNumber, int r)` to accomplish this.

Test Cases:

1. Input: ("Paris", "Alice", "2022-09-15", 123456, 2)

   Output: "eParis22034"

2. Input: ("London", "Bob", "2022-10-21", 654321, 3)

   Output: "bLondon221321"

3. Input: ("NewYork", "Clara", "2022-11-30", 789012, 1)

   Output: "aNewYork22089"

4. Input: ("Tokyo", "David", "2022-12-25", 456789, 4)

   Output: "dTokyo22289"

5. Input: ("Berlin", "Eva", "2022-08-01", 987654, 2)

   Output: "aBerlin22076"

6. Input: ("Sydney", "Frank", "2023-01-10", 345678, 3)

   Output: "kSydney223567"

7. Input: ("Dubai", "Gina", "2023-02-14", 876543, 2)

   Output: "aDubai22354"

8. Input: ("Rome", "H

ank", "2023-03-18", 234567, 1)

   Output: "kRome22334"

9. Input: ("Barcelona", "Ivy", "2023-04-22", 567123, 4)

   Output: "yBarcelona22323"

10. Input: ("Moscow", "Jack", "2023-05-26", 678912, 2)

    Output: "kMoscow22389"

Question 5: Vehicle Registration Number Generation

Scenario:

You need to generate unique registration numbers for vehicles. The registration process uses the vehicle's Type, OwnerName, Model, and a RegistrationNumber. The steps to generate the registration number are as follows:

Step 1: Compare the lengths of OwnerName and Model. The shorter string will be called ShorterString, and the longer one will be called LongerString. If they are of equal length, the one that appears first alphabetically is ShorterString.

Step 2: The registration number should be generated as follows:

- First letter of ShorterString + Entire Type + Last letter of LongerString + RegistrationNumber digits at positions Y and Y+1.

Implement the method `String generateRegistrationNumber(String type, String ownerName, String model, int registrationNumber, int y)` to accomplish this.

Test Cases:

1. Input: ("SUV", "John", "X5", 123456, 2)

   Output: "JSUVX34"

2. Input: ("Sedan", "Anna", "Camry", 654321, 3)

   Output: "ASedanY321"

3. Input: ("Truck", "Mike", "F150", 789012, 1)

   Output: "MTruck01589"

4. Input: ("Coupe", "Lisa", "Mustang", 456789, 4)

   Output: "LCoupeG89"

5. Input: ("Hatchback", "Tom", "Golf", 987654, 2)

   Output: "THatchbackF76"

6. Input: ("Convertible", "Eva", "Z4", 345678, 3)

   Output: "EConvertibleR567"

7. Input: ("Van", "Chris", "Sprinter", 876543, 2)

   Output: "CVanR54"

8. Input: ("Wagon", "Alice", "Outback", 234567, 1)

   Output: "AWagonK34"

9. Input: ("Motorcycle", "Peter", "Ninja", 567123, 4)

   Output: "PMotorcycleA23"

10. Input: ("Bus", "Nancy", "Transit", 678912, 2)

    Output: "NBusT89"