Week 3 - S3 - OOP Fundamental - Lab Practice Problem

Name:Ramesh Harisabapathi Chettiar
Date of Submission:17/09/25

Program 1→

```java
/*Write a program to create a Bank Account management system
without using built-in collection classes
 * a. Create a BankAccount class with private instance variables: accountNumber
(String), accountHolderName (String), balance (double), and a static variable
totalAccounts (int)
b. Create a constructor that takes account holder name and initial deposit,
automatically generates account number using a static counter
c. Create instance methods: deposit(double amount), withdraw(double amount),
checkBalance() with proper validation for negative amounts and insufficient funds
d. Create static methods: getTotalAccounts(), generateAccountNumber() that
returns a unique account number like "ACC001", "ACC002"
e. Create a method displayAccountInfo() to show all account details in a formatted
manner
f. In the main method, create an array of BankAccount objects, demonstrate
creating multiple accounts, performing transactions, and showing the difference
between static and instance variables
 */

public class BankAccount {
    // Private instance variables
    private String accountNumber;
    private String accountHolderName;
    private double balance;

    // Static variable
    private static int totalAccounts = 0;
    private static int accountCounter = 0;

    // Constructor
    public BankAccount(String accountHolderName, double initialDeposit) {
        this.accountHolderName = accountHolderName;
        this.balance = initialDeposit;
        this.accountNumber = generateAccountNumber();
        totalAccounts++;
    }

    // Static method to generate unique account number
    private static String generateAccountNumber() {
        accountCounter++;
        return String.format("ACC%03d", accountCounter);
    }
```

```java
    // Static method to get total accounts
    public static int getTotalAccounts() {
        return totalAccounts;
    }

    // Instance method to deposit
    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            System.out.println("Deposited: " + amount + ". New balance: " + balance);
        } else {
            System.out.println("Invalid deposit amount. Amount must be positive.");
        }
    }

    // Instance method to withdraw
    public void withdraw(double amount) {
        if (amount > 0 && balance >= amount) {
            balance -= amount;
            System.out.println("Withdrew: " + amount + ". New balance: " + balance);
        } else if (amount <= 0) {
            System.out.println("Invalid withdrawal amount. Amount must be positive.");
        } else {
            System.out.println("Insufficient funds. Current balance: " + balance);
        }
    }

    // Instance method to check balance
    public double checkBalance() {
        return balance;
    }

    // Instance method to display account info
    public void displayAccountInfo() {
        System.out.println("Account Number: " + accountNumber);
        System.out.println("Account Holder Name: " + accountHolderName);
        System.out.println("Balance: " + balance);
        System.out.println("----------------------------");
    }

```

```java
    // Main method for demonstration
    Run main | Debug main
    public static void main(String[] args) {
        // Create an array of BankAccount objects
        BankAccount[] accounts = new BankAccount[3];

        // Create multiple accounts
        accounts[0] = new BankAccount("Ramesh Chettiar", 1000.0);
        accounts[1] = new BankAccount("Shah Rukh Khan", 500.0);
        accounts[2] = new BankAccount("Vijay", 2000.0);

        // Display total accounts (static variable)
        System.out.println("Total Accounts Created: " + BankAccount.getTotalAccounts());
        System.out.println();

        // Perform transactions on each account
        accounts[0].deposit(500.0);
        accounts[0].withdraw(200.0);
        accounts[0].displayAccountInfo();

        accounts[1].withdraw(600.0); // Should fail due to insufficient funds
        accounts[1].deposit(-100.0); // Should fail due to negative amount
        accounts[1].displayAccountInfo();

        accounts[2].withdraw(500.0);
        accounts[2].deposit(1000.0);
        accounts[2].displayAccountInfo();

        // Demonstrate static vs instance variables
        System.out.println("Total Accounts (static): " + BankAccount.getTotalAccounts());
        System.out.println("Balance of account 0 (instance): " + accounts[0].checkBalance());
        System.out.println("Balance of account 1 (instance): " + accounts[1].checkBalance());
        System.out.println("Balance of account 2 (instance): " + accounts[2].checkBalance());
    }
}
```

OUTPUT→

```
PS C:\Users\Ramesh\Personal Folders\MISCELLANEOUS\ENTRANCE EXAMS\SRM\SEMESTERS\SEMESTER-3\JAVA-STEP\Weeks\Week 3\Lab Practise> cd "c:\Users\Ramesh\Personal Folders\MISCELLANE
ENTRANCE EXAMS\SRM\SEMESTERS\SEMESTER-3\JAVA-STEP\Weeks\Week 3\Lab Practise\Problem 1\" ; if ($?) { javac BankAccount.java } ; if ($?) { java BankAccount }
Total Accounts Created: 3

Deposited: 500.0. New balance: 1500.0
Withdrew: 200.0. New balance: 1300.0
Account Number: ACC001
Account Holder Name: Ramesh Chettiar
Balance: 1300.0
----------------------------
Insufficient funds. Current balance: 500.0
Invalid deposit amount. Amount must be positive.
Account Number: ACC002
Account Holder Name: Shah Rukh Khan
Balance: 500.0
----------------------------
Withdrew: 500.0. New balance: 1500.0
Deposited: 1000.0. New balance: 2500.0
Account Number: ACC003
Account Holder Name: Vijay
Balance: 2500.0
----------------------------
Total Accounts (static): 3
Balance of account 0 (instance): 1300.0
Balance of account 1 (instance): 500.0
Balance of account 2 (instance): 2500.0
PS C:\Users\Ramesh\Personal Folders\MISCELLANEOUS\ENTRANCE EXAMS\SRM\SEMESTERS\SEMESTER-3\JAVA-STEP\Weeks\Week 3\Lab Practise\Problem 1>
```

Program 2→

```java
import java.util.UUID;

public class LibrarySystem {

    // --------- Book Class ---------
    static class Book {
        private String bookId;
        private String title;
        private String author;
        private boolean isAvailable;

        private static int totalBooks = 0;
        private static int availableBooks = 0;

        // Constructor
        public Book(String title, String author) {
            this.bookId = generateBookId();
            this.title = title;
            this.author = author;
            this.isAvailable = true;
            totalBooks++;
            availableBooks++;
        }

        // Issue the book
        public boolean issueBook() {
            if (isAvailable) {
                isAvailable = false;
                availableBooks--;
                return true;
            } else {
                System.out.println(title + " is currently not available.");
                return false;
            }
        }

        // Return the book
        public void returnBook() {
            if (!isAvailable) {
                isAvailable = true;
                availableBooks++;
```

```java
        // Display book info
        public void displayBookInfo() {
            System.out.println("Book ID: " + bookId + " | Title: " + title + " | Author: " + author + " | Available: " + isAvailable);
        }

        // Generate unique book ID
        public static String generateBookId() {
            return "B-" + UUID.randomUUID().toString().substring(0, 8);
        }

        // Getters
        public String getBookId() {
            return bookId;
        }

        public boolean isAvailable() {
            return isAvailable;
        }

        // Static stats
        public static void displayStats() {
            System.out.println("Total Books: " + totalBooks + " | Available Books: " + availableBooks);
        }
    }

    // --------- Member Class ---------
    static class Member {
        private String memberId;
        private String memberName;
        private String[] booksIssued;
        private int bookCount;

        // Constructor
        public Member(String memberName, int maxBooks) {
            this.memberId = generateMemberId();
            this.memberName = memberName;
            this.booksIssued = new String[maxBooks];
            this.bookCount = 0;
        }
```

```java
        public void borrowBook(Book book) {
            if (bookCount >= booksIssued.length) {
                System.out.println(memberName + " cannot borrow more books.");
                return;
            }
            if (book.issueBook()) {
                booksIssued[bookCount] = book.getBookId();
                bookCount++;
                System.out.println(memberName + " borrowed " + book.getBookId());
            }
        }

        // Return a book by bookId
        public void returnBook(String bookId, Book[] books) {
            for (int i = 0; i < books.length; i++) {
                if (books[i].getBookId().equals(bookId)) {
                    for (int j = 0; j < bookCount; j++) {
                        if (booksIssued[j].equals(bookId)) {
                            books[i].returnBook();
                            booksIssued[j] = booksIssued[bookCount - 1]; // Shift last book to current
                            booksIssued[bookCount - 1] = null;
                            bookCount--;
                            System.out.println(memberName + " returned " + bookId);
                            return;
                        }
                    }
                }
            }
            System.out.println(memberName + " did not borrow book " + bookId);
        }

        // Display member info
        public void displayMemberInfo() {
            System.out.print("Member ID: " + memberId + " | Name: " + memberName + " | Books Issued: ");
            for (int i = 0; i < bookCount; i++) {
                System.out.print(booksIssued[i] + " ");
            }
            System.out.println();
```

```
126         // Generate unique member ID
127         public static String generateMemberId() {
128             return "M-" + UUID.randomUUID().toString().substring(0, 8);
129         }
130     }
131
132     // --------- Main Method ---------
        Run main | Debug main
133     public static void main(String[] args) {
134         // Create books
135         Book[] books = new Book[3];
136         books[0] = new Book("1984", "George Orwell");
137         books[1] = new Book("The Hobbit", "J.R.R. Tolkien");
138         books[2] = new Book("To Kill a Mockingbird", "Harper Lee");
139
140         // Create members
141         Member[] members = new Member[2];
142         members[0] = new Member("Alice", 2);
143         members[1] = new Member("Bob", 3);
144
145         // Borrow books
146         members[0].borrowBook(books[0]);
147         members[0].borrowBook(books[1]);
148         members[0].borrowBook(books[2]); // Should fail (max 2 books)
149
150         members[1].borrowBook(books[2]); // Should succeed
151
152         // Display info
153         System.out.println("\n--- Books Info ---");
154         for (Book b : books) b.displayBookInfo();
155         Book.displayStats();
156
157         System.out.println("\n--- Members Info ---");
158         for (Member m : members) m.displayMemberInfo();
159
160         // Return books
161         members[0].returnBook(books[0].getBookId(), books);
162         members[1].returnBook(books[1].getBookId(), books); // Not borrowed
163
164         System.out.println("\n--- After Returning Books ---");
165         for (Book b : books) b.displayBookInfo();
166         Book.displayStats();
167
168         for (Member m : members) m.displayMemberInfo();
169     }
170 }
171
```

OUTPUT→

```
PS C:\Users\Ramesh\Personal Folders\MISCELLANEOUS\ENTRANCE EXAMS\SRM\SEMESTERS\SEMESTER-3\JAVA-STEP\Weeks\Week 3\Lab Practise\Problem 2> cd "c:\Users\Ramesh\Personal Folders\
ELLANEOUS\ENTRANCE EXAMS\SRM\SEMESTERS\SEMESTER-3\JAVA-STEP\Weeks\Week 3\Lab Practise\Problem 2\" ; if ($?) { javac LibrarySystem.java } ; if ($?) { java LibrarySystem }
Alice borrowed B-437498fc
Alice borrowed B-d187b521
Alice cannot borrow more books.
Bob borrowed B-688e94b4

--- Books Info ---
Book ID: B-437498fc | Title: 1984 | Author: George Orwell | Available: false
Book ID: B-d187b521 | Title: The Hobbit | Author: J.R.R. Tolkien | Available: false
Book ID: B-688e94b4 | Title: To Kill a Mockingbird | Author: Harper Lee | Available: false
Total Books: 3 | Available Books: 0

--- Members Info ---
Member ID: M-0c7577ea | Name: Alice | Books Issued: B-437498fc B-d187b521
Member ID: M-927db171 | Name: Bob | Books Issued: B-688e94b4
Alice returned B-437498fc
Bob did not borrow book B-d187b521

--- After Returning Books ---
Book ID: B-437498fc | Title: 1984 | Author: George Orwell | Available: true
Book ID: B-d187b521 | Title: The Hobbit | Author: J.R.R. Tolkien | Available: false
Book ID: B-688e94b4 | Title: To Kill a Mockingbird | Author: Harper Lee | Available: false
Total Books: 3 | Available Books: 1
Member ID: M-0c7577ea | Name: Alice | Books Issued: B-d187b521
Member ID: M-927db171 | Name: Bob | Books Issued: B-688e94b4
```

PROGRAM 4→

```java
public class Vehicle {
    // Private instance variables
    private String vehicleId;
    private String brand;
    private String model;
    private double rentPerDay;
    private boolean isAvailable;

    // Static variables
    private static int totalVehicles = 0;
    private static double totalRevenue = 0.0;
    private static String companyName = "Default Rental Company";
    private static int rentalDays = 0;
    private static int vehicleCounter = 0;

    // Constructor
    public Vehicle(String brand, String model, double rentPerDay) {
        this.brand = brand;
        this.model = model;
        this.rentPerDay = rentPerDay;
        this.isAvailable = true;
        this.vehicleId = generateVehicleId();
        totalVehicles++;
    }

    // Private static method to generate vehicle ID
    private static String generateVehicleId() {
        vehicleCounter++;
        return String.format("V%03d", vehicleCounter);
    }

    // Static method to set company name
    public static void setCompanyName(String name) {
        companyName = name;
    }

    // Static method to get total revenue
```

```java
public class Vehicle {
    // Static method to get total revenue
    public static double getTotalRevenue() {
        return totalRevenue;
    }

    // Static method to get average rent per day
    public static double getAverageRentPerDay() {
        if (rentalDays > 0) {
            return totalRevenue / rentalDays;
        } else {
            return 0.0;
        }
    }

    // Static method to display company stats
    public static void displayCompanyStats() {
        System.out.println("Company Name: " + companyName);
        System.out.println("Total Vehicles: " + totalVehicles);
        System.out.println("Total Revenue: $" + totalRevenue);
        System.out.println("Average Rent Per Day: $" + getAverageRentPerDay());
        System.out.println("----------------------------");
    }

    // Instance method to calculate rent
    public double calculateRent(int days) {
        return days * rentPerDay;
    }

    // Instance method to rent vehicle
    public void rentVehicle(int days) {
        if (isAvailable && days > 0) {
            double rent = calculateRent(days);
            totalRevenue += rent;
            rentalDays += days;
            isAvailable = false;
            System.out.println("Vehicle " + vehicleId + " rented for " + days + " days. Rent: $" + rent);
```

```
        } else if (!isAvailable) {
            System.out.println("Vehicle " + vehicleId + " is not available.");
        } else {
            System.out.println("Invalid number of days.");
        }
    }

    // Instance method to return vehicle
    public void returnVehicle() {
        if (!isAvailable) {
            isAvailable = true;
            System.out.println("Vehicle " + vehicleId + " returned and is now available.");
        } else {
            System.out.println("Vehicle " + vehicleId + " is already available.");
        }
    }

    // Instance method to display vehicle info
    public void displayVehicleInfo() {
        System.out.println("Vehicle ID: " + vehicleId);
        System.out.println("Brand: " + brand);
        System.out.println("Model: " + model);
        System.out.println("Rent Per Day: $" + rentPerDay);
        System.out.println("Available: " + (isAvailable ? "Yes" : "No"));
        System.out.println("-----------------------------");
    }
```

```
    public static void main(String[] args) {
        // Set company name
        Vehicle.setCompanyName("Super Rentals");

        // Create an array of Vehicle objects
        Vehicle[] vehicles = new Vehicle[3];

        // Create multiple vehicles
        vehicles[0] = new Vehicle("Toyota", "Highlander", 4000.0);
        vehicles[1] = new Vehicle("Honda", "BR-V", 4500.0);
        vehicles[2] = new Vehicle("Ford", "Endeavour", 8000.0);

        // Display company stats initially
        System.out.println("Initial Company Stats:");
        Vehicle.displayCompanyStats();

        // Perform rentals
        vehicles[0].rentVehicle(3); // Rent for 3 days
        vehicles[1].rentVehicle(5); // Rent for 5 days
        vehicles[2].rentVehicle(2); // Rent for 2 days

        // Try to rent an unavailable vehicle
        vehicles[0].rentVehicle(1); // Should fail

        // Display vehicle info
        System.out.println("\nVehicle Information:");
        for (Vehicle v : vehicles) {
            v.displayVehicleInfo();
        }

        // Return vehicles
        vehicles[0].returnVehicle();
        vehicles[1].returnVehicle();

        // Display updated company stats
        System.out.println("Updated Company Stats:");
```

```
        // Display updated company stats
        System.out.println("Updated Company Stats:");
        Vehicle.displayCompanyStats();

        // Demonstrate static vs instance
        System.out.println("\nDemonstrating static vs instance:");
        System.out.println("Total Vehicles (static): " + Vehicle.totalVehicles);
        System.out.println("Total Revenue (static): INR." + Vehicle.getTotalRevenue());
        System.out.println("Vehicle 0 Available (instance): " + vehicles[0].isAvailable);
        System.out.println("Vehicle 1 Available (instance): " + vehicles[1].isAvailable);
        System.out.println("Vehicle 2 Available (instance): " + vehicles[2].isAvailable);
    }
}
```