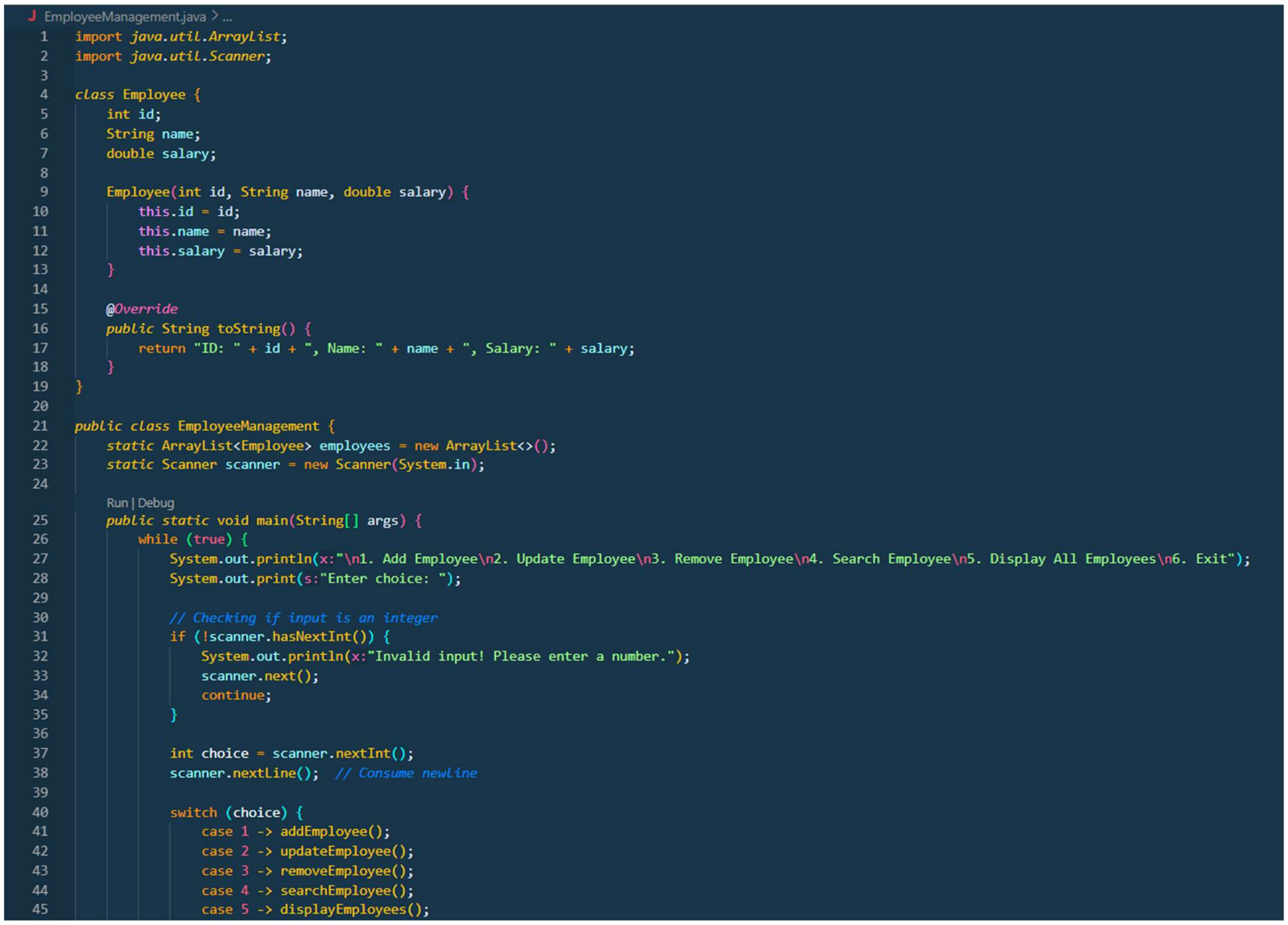
UID -22BCS14823 NAME- AYUSH S NAIR

EXP-04

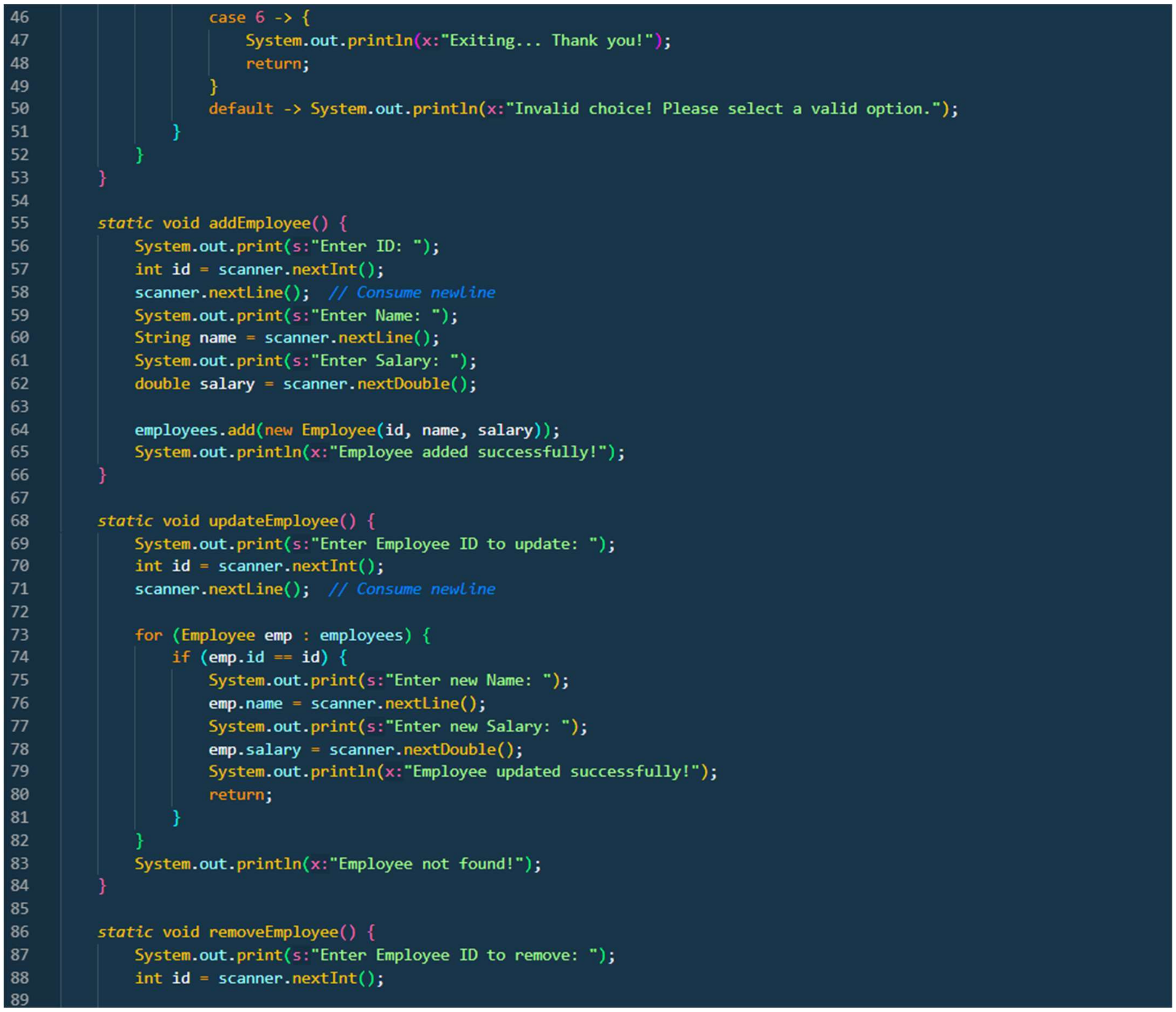
1. Write a Java program to implement an ArrayList that stores employee details (ID, Name, and Salary). Allow users to add, update, remove, and search employees.(Easy)

Code:

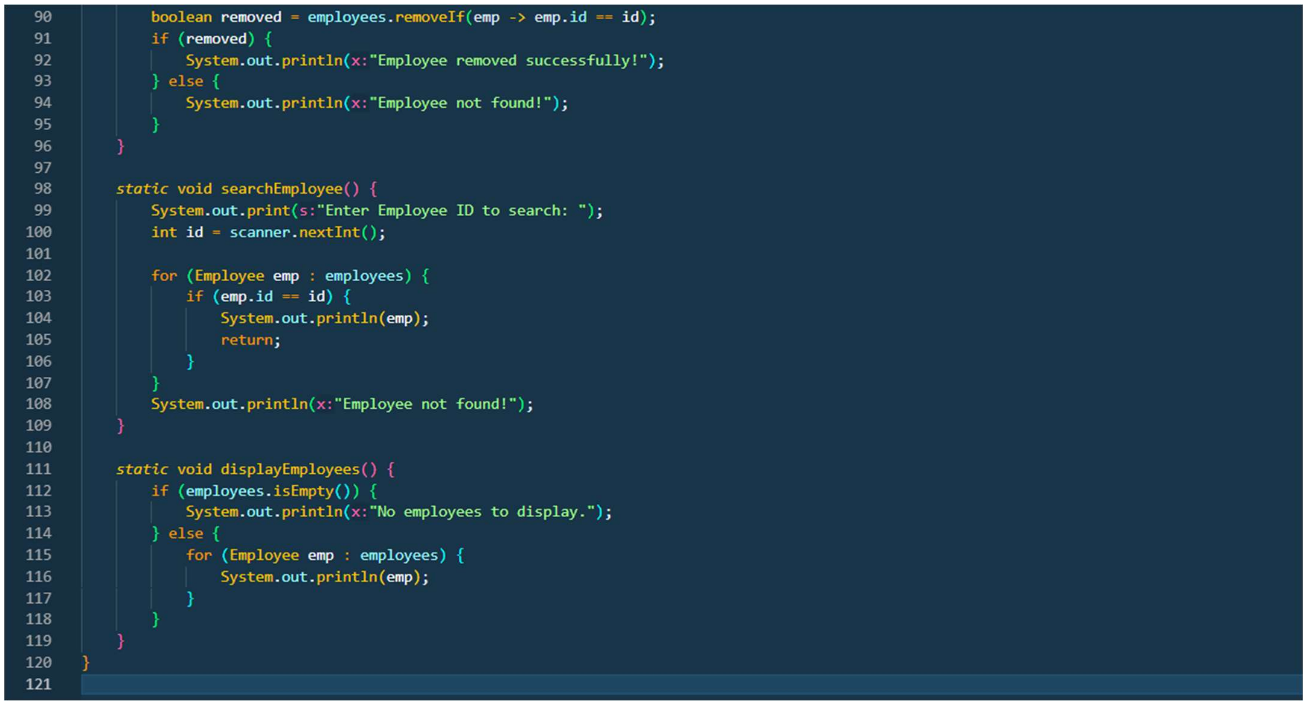


EmployeeManagement.java > ...

```java
import java.util.ArrayList;
import java.util.Scanner;

class Employee {
    int id;
    String name;
    double salary;

    Employee(int id, String name, double salary) {
        this.id = id;
        this.name = name;
        this.salary = salary;
    }

    @Override
    public String toString() {
        return "ID: " + id + ", Name: " + name + ", Salary: " + salary;
    }
}

public class EmployeeManagement {
    static ArrayList<Employee> employees = new ArrayList<>();
    static Scanner scanner = new Scanner(System.in);

    Run | Debug
    public static void main(String[] args) {
        while (true) {
            System.out.println(x:"\n1. Add Employee\n2. Update Employee\n3. Remove Employee\n4. Search Employee\n5. Display All Employees\n6. Exit");
            System.out.print(s:"Enter choice: ");

            // Checking if input is an integer
            if (!scanner.hasNextInt()) {
                System.out.println(x:"Invalid input! Please enter a number.");
                scanner.next();
                continue;
            }

            int choice = scanner.nextInt();
            scanner.nextLine();  // Consume newline

            switch (choice) {
                case 1 -> addEmployee();
                case 2 -> updateEmployee();
                case 3 -> removeEmployee();
                case 4 -> searchEmployee();
                case 5 -> displayEmployees();
```

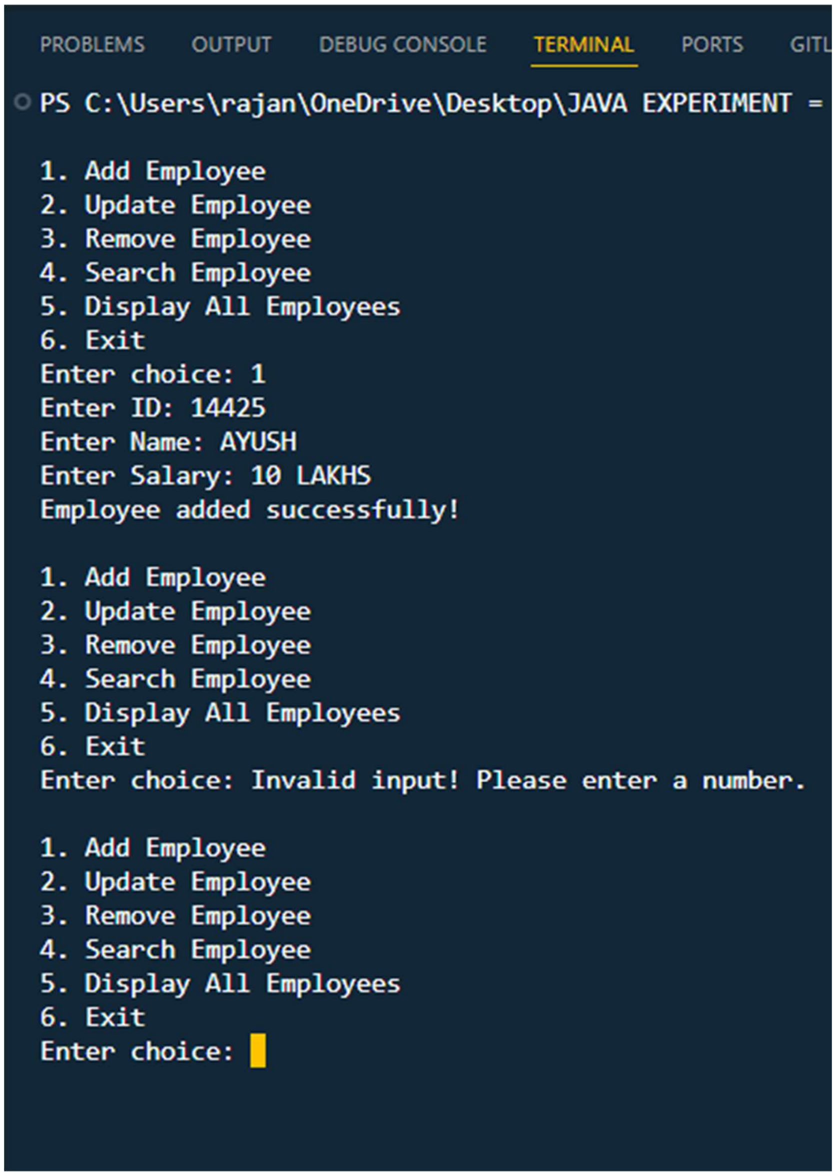


```java
                case 6 -> {
                    System.out.println(x:"Exiting... Thank you!");
                    return;
                }
                default -> System.out.println(x:"Invalid choice! Please select a valid option.");
            }
        }
    }

    static void addEmployee() {
        System.out.print(s:"Enter ID: ");
        int id = scanner.nextInt();
        scanner.nextLine();  // Consume newline
        System.out.print(s:"Enter Name: ");
        String name = scanner.nextLine();
        System.out.print(s:"Enter Salary: ");
        double salary = scanner.nextDouble();

        employees.add(new Employee(id, name, salary));
        System.out.println(x:"Employee added successfully!");
    }

    static void updateEmployee() {
        System.out.print(s:"Enter Employee ID to update: ");
        int id = scanner.nextInt();
        scanner.nextLine();  // Consume newline

        for (Employee emp : employees) {
            if (emp.id == id) {
                System.out.print(s:"Enter new Name: ");
                emp.name = scanner.nextLine();
                System.out.print(s:"Enter new Salary: ");
                emp.salary = scanner.nextDouble();
                System.out.println(x:"Employee updated successfully!");
                return;
            }
        }
        System.out.println(x:"Employee not found!");
    }

    static void removeEmployee() {
        System.out.print(s:"Enter Employee ID to remove: ");
        int id = scanner.nextInt();

```



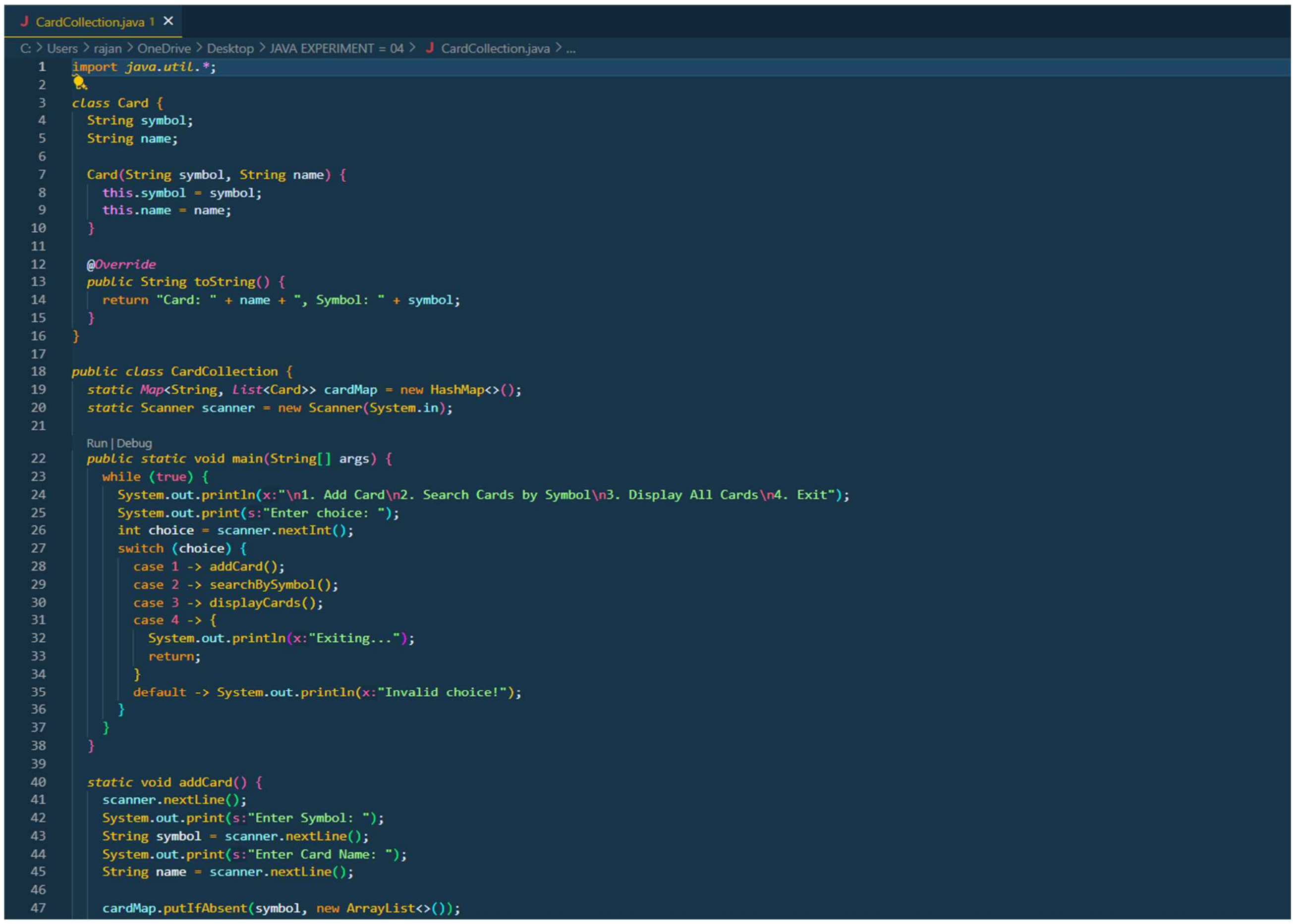
```java
        boolean removed = employees.removeIf(emp -> emp.id == id);
        if (removed) {
            System.out.println(x:"Employee removed successfully!");
        } else {
            System.out.println(x:"Employee not found!");
        }
    }

    static void searchEmployee() {
        System.out.print(s:"Enter Employee ID to search: ");
        int id = scanner.nextInt();

        for (Employee emp : employees) {
            if (emp.id == id) {
                System.out.println(emp);
                return;
            }
        }
        System.out.println(x:"Employee not found!");
    }

    static void displayEmployees() {
        if (employees.isEmpty()) {
            System.out.println(x:"No employees to display.");
        } else {
            for (Employee emp : employees) {
                System.out.println(emp);
            }
        }
    }
}
```

OUTPUT:



```
PS C:\Users\rajan\OneDrive\Desktop\JAVA EXPERIMENT =

1. Add Employee
2. Update Employee
3. Remove Employee
4. Search Employee
5. Display All Employees
6. Exit
Enter choice: 1
Enter ID: 14425
Enter Name: AYUSH
Enter Salary: 10 LAKHS
Employee added successfully!

1. Add Employee
2. Update Employee
3. Remove Employee
4. Search Employee
5. Display All Employees
6. Exit
Enter choice: Invalid input! Please enter a number.

1. Add Employee
2. Update Employee
3. Remove Employee
4. Search Employee
5. Display All Employees
6. Exit
Enter choice:
```

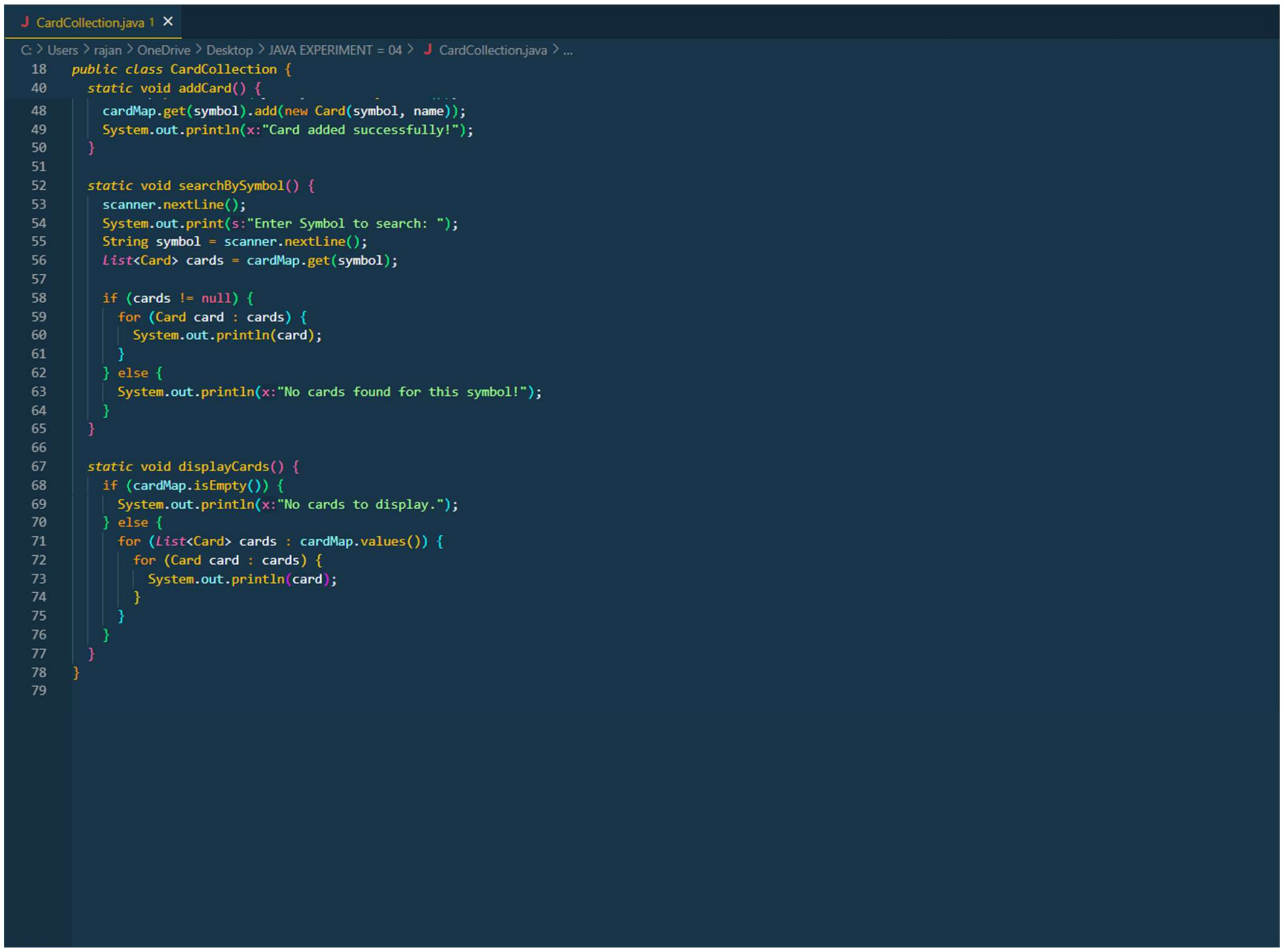
2. Create a program to collect and store all the cards to assist the users in finding all the cards in a given symbol using Collection interface.(Medium)



CardCollection.java 1

C: > Users > rajan > OneDrive > Desktop > JAVA EXPERIMENT = 04 > CardCollection.java > ...

```java
import java.util.*;

class Card {
    String symbol;
    String name;

    Card(String symbol, String name) {
        this.symbol = symbol;
        this.name = name;
    }

    @Override
    public String toString() {
        return "Card: " + name + ", Symbol: " + symbol;
    }
}

public class CardCollection {
    static Map<String, List<Card>> cardMap = new HashMap<>();
    static Scanner scanner = new Scanner(System.in);

    Run | Debug
    public static void main(String[] args) {
        while (true) {
            System.out.println(x:"\n1. Add Card\n2. Search Cards by Symbol\n3. Display All Cards\n4. Exit");
            System.out.print(s:"Enter choice: ");
            int choice = scanner.nextInt();
            switch (choice) {
                case 1 -> addCard();
                case 2 -> searchBySymbol();
                case 3 -> displayCards();
                case 4 -> {
                    System.out.println(x:"Exiting...");
                    return;
                }
                default -> System.out.println(x:"Invalid choice!");
            }
        }
    }

    static void addCard() {
        scanner.nextLine();
        System.out.print(s:"Enter Symbol: ");
        String symbol = scanner.nextLine();
        System.out.print(s:"Enter Card Name: ");
        String name = scanner.nextLine();

        cardMap.putIfAbsent(symbol, new ArrayList<>());
```

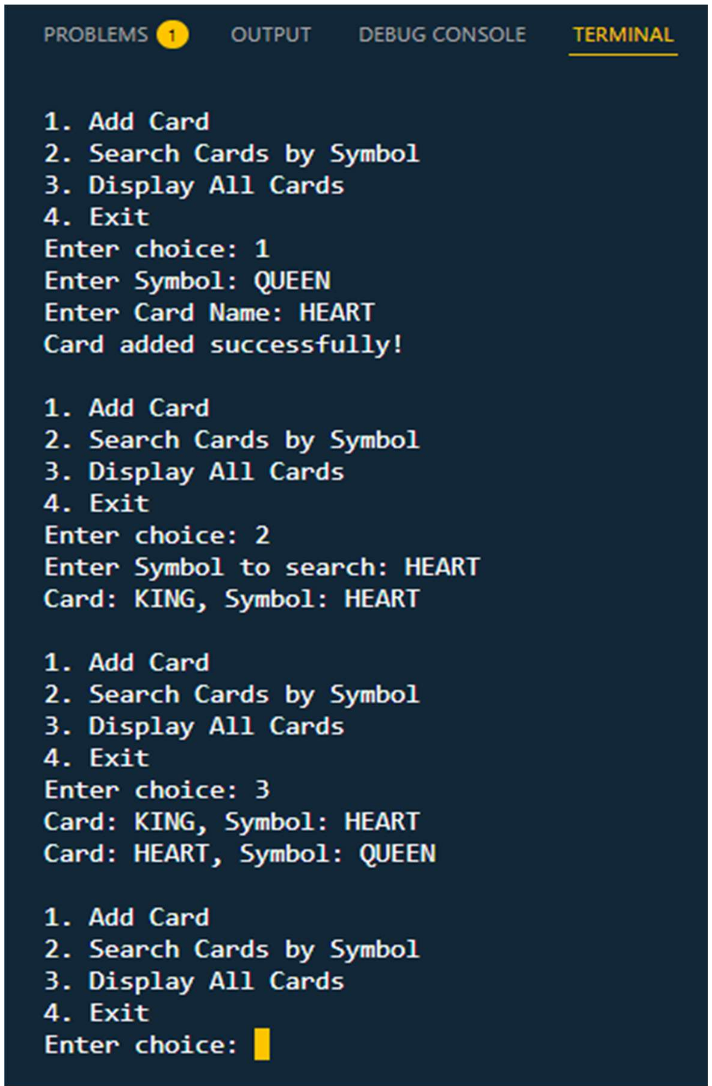


CardCollection.java 1 ×

C: > Users > rajan > OneDrive > Desktop > JAVA EXPERIMENT = 04 > CardCollection.java > ...

```java
public class CardCollection {
  static void addCard() {
    cardMap.get(symbol).add(new Card(symbol, name));
    System.out.println(x:"Card added successfully!");
  }

  static void searchBySymbol() {
    scanner.nextLine();
    System.out.print(s:"Enter Symbol to search: ");
    String symbol = scanner.nextLine();
    List<Card> cards = cardMap.get(symbol);

    if (cards != null) {
      for (Card card : cards) {
        System.out.println(card);
      }
    } else {
      System.out.println(x:"No cards found for this symbol!");
    }
  }

  static void displayCards() {
    if (cardMap.isEmpty()) {
      System.out.println(x:"No cards to display.");
    } else {
      for (List<Card> cards : cardMap.values()) {
        for (Card card : cards) {
          System.out.println(card);
        }
      }
    }
  }
}
```

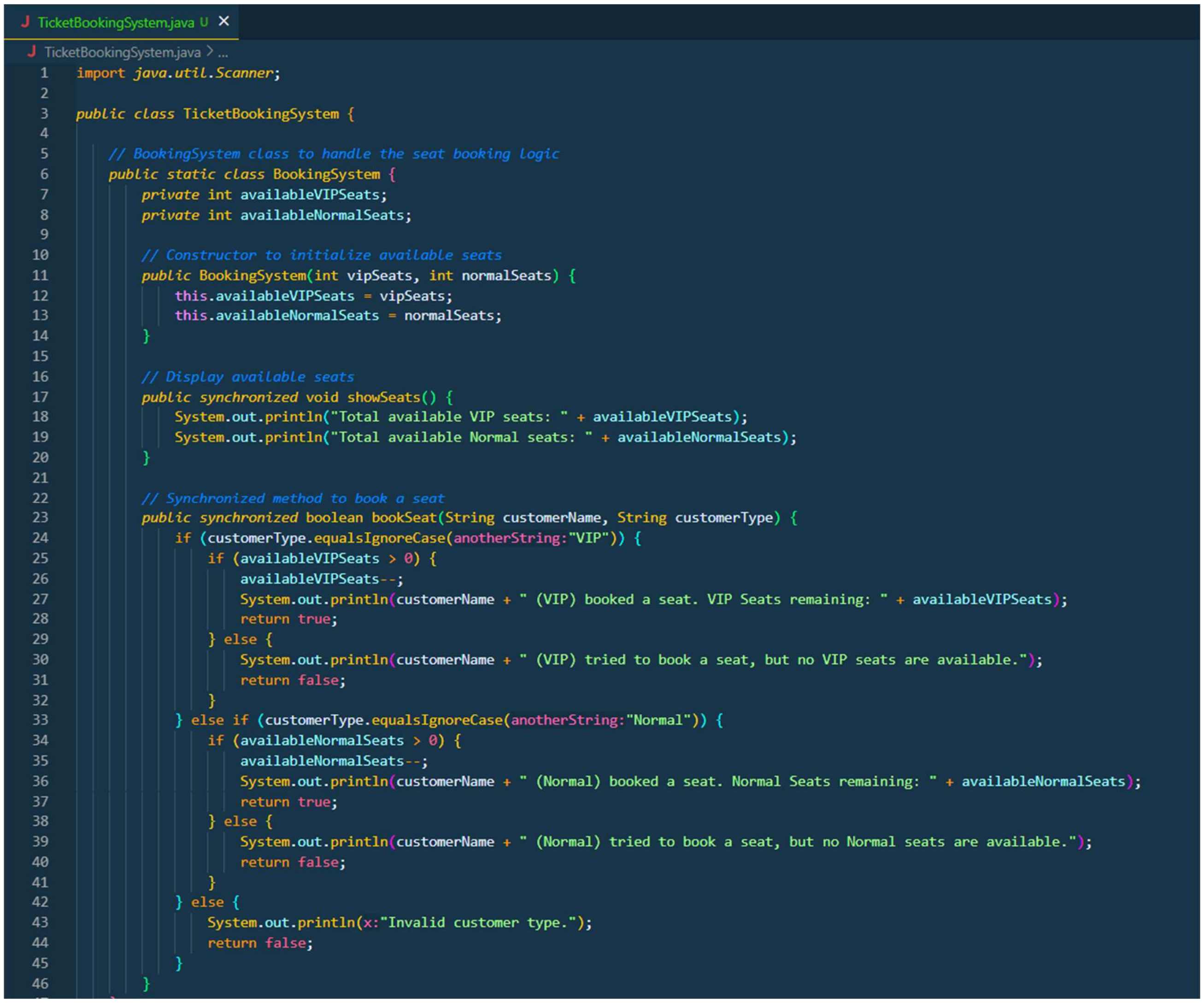
OUTPUT :



PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL

```
1. Add Card
2. Search Cards by Symbol
3. Display All Cards
4. Exit
Enter choice: 1
Enter Symbol: QUEEN
Enter Card Name: HEART
Card added successfully!

1. Add Card
2. Search Cards by Symbol
3. Display All Cards
4. Exit
Enter choice: 2
Enter Symbol to search: HEART
Card: KING, Symbol: HEART

1. Add Card
2. Search Cards by Symbol
3. Display All Cards
4. Exit
Enter choice: 3
Card: KING, Symbol: HEART
Card: HEART, Symbol: QUEEN

1. Add Card
2. Search Cards by Symbol
3. Display All Cards
4. Exit
Enter choice: 
```

3. Develop a ticket booking system with synchronized threads to ensure no double booking of seats. Use thread priorities to simulate VIP bookings being processed first.(Hard)

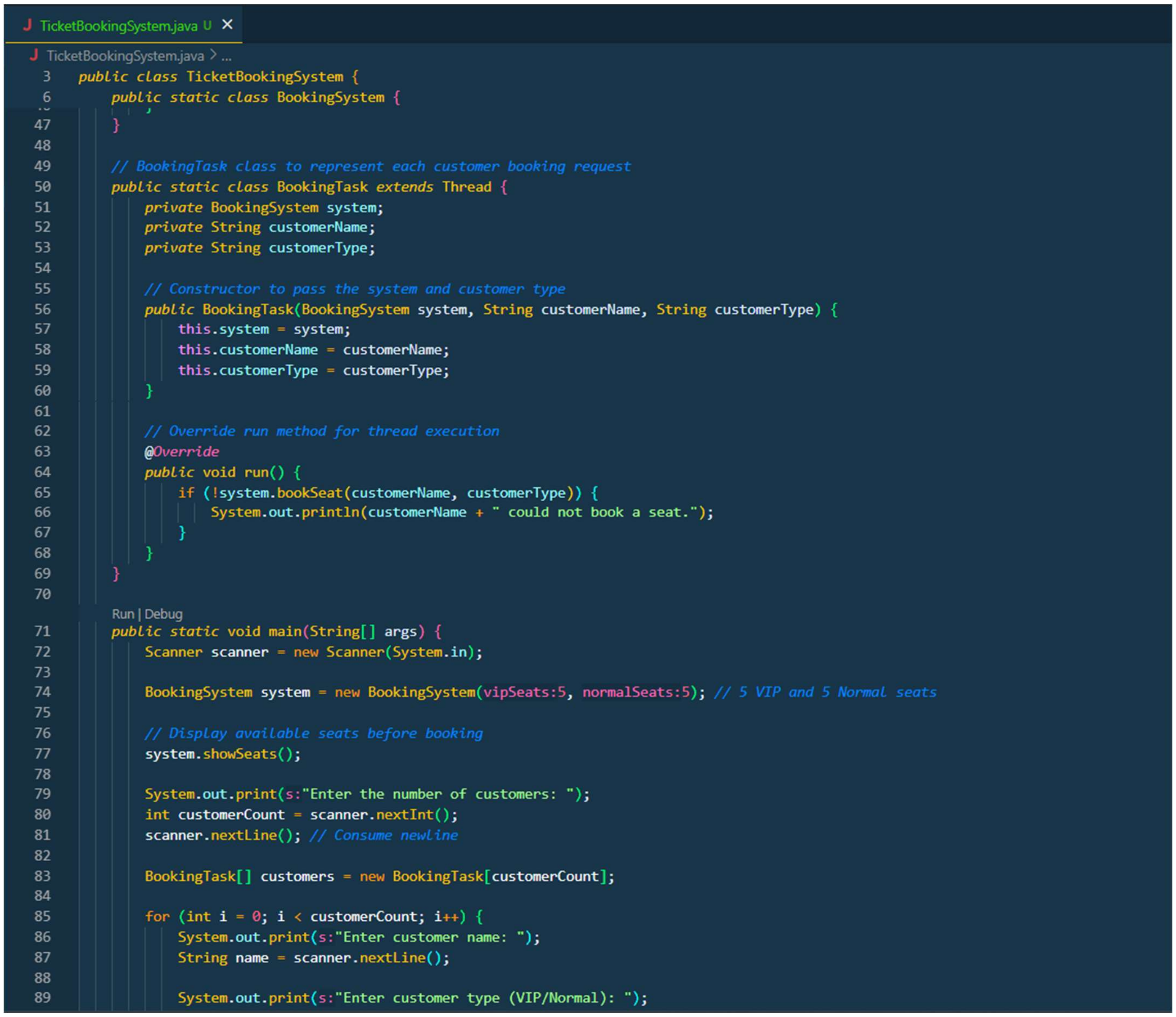


TicketBookingSystem.java U ×

TicketBookingSystem.java > ...

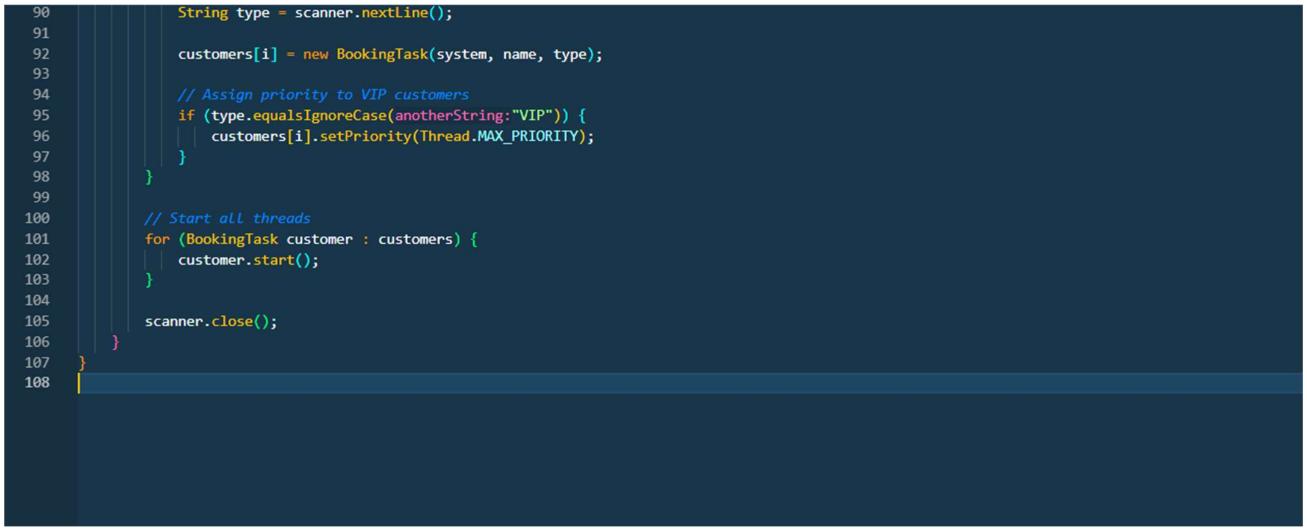
```java
import java.util.Scanner;

public class TicketBookingSystem {

    // BookingSystem class to handle the seat booking logic
    public static class BookingSystem {
        private int availableVIPSeats;
        private int availableNormalSeats;

        // Constructor to initialize available seats
        public BookingSystem(int vipSeats, int normalSeats) {
            this.availableVIPSeats = vipSeats;
            this.availableNormalSeats = normalSeats;
        }

        // Display available seats
        public synchronized void showSeats() {
            System.out.println("Total available VIP seats: " + availableVIPSeats);
            System.out.println("Total available Normal seats: " + availableNormalSeats);
        }

        // Synchronized method to book a seat
        public synchronized boolean bookSeat(String customerName, String customerType) {
            if (customerType.equalsIgnoreCase(anotherString:"VIP")) {
                if (availableVIPSeats > 0) {
                    availableVIPSeats--;
                    System.out.println(customerName + " (VIP) booked a seat. VIP Seats remaining: " + availableVIPSeats);
                    return true;
                } else {
                    System.out.println(customerName + " (VIP) tried to book a seat, but no VIP seats are available.");
                    return false;
                }
            } else if (customerType.equalsIgnoreCase(anotherString:"Normal")) {
                if (availableNormalSeats > 0) {
                    availableNormalSeats--;
                    System.out.println(customerName + " (Normal) booked a seat. Normal Seats remaining: " + availableNormalSeats);
                    return true;
                } else {
                    System.out.println(customerName + " (Normal) tried to book a seat, but no Normal seats are available.");
                    return false;
                }
            } else {
                System.out.println(x:"Invalid customer type.");
                return false;
            }
        }
```

UID -22BCS14823 NAME- AYUSH S NAIR


TicketBookingSystem.java U ×

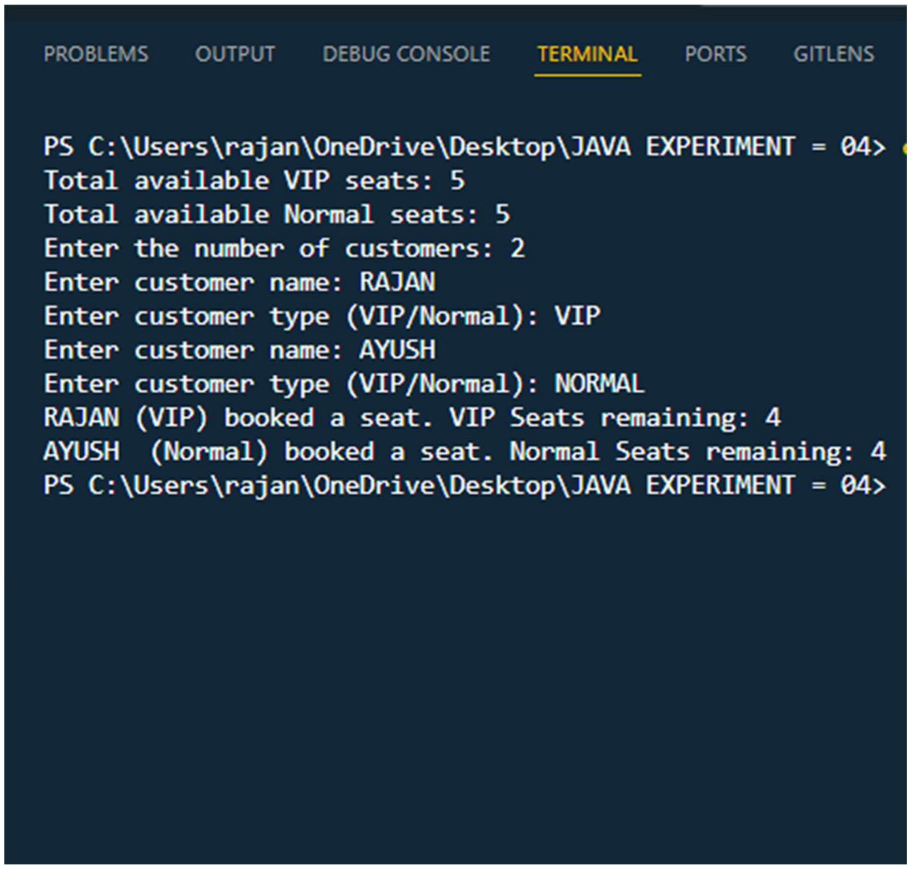
TicketBookingSystem.java > ...

```java
public class TicketBookingSystem {
    public static class BookingSystem {
    }

    // BookingTask class to represent each customer booking request
    public static class BookingTask extends Thread {
        private BookingSystem system;
        private String customerName;
        private String customerType;

        // Constructor to pass the system and customer type
        public BookingTask(BookingSystem system, String customerName, String customerType) {
            this.system = system;
            this.customerName = customerName;
            this.customerType = customerType;
        }

        // Override run method for thread execution
        @Override
        public void run() {
            if (!system.bookSeat(customerName, customerType)) {
                System.out.println(customerName + " could not book a seat.");
            }
        }
    }

    Run | Debug
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        BookingSystem system = new BookingSystem(vipSeats:5, normalSeats:5); // 5 VIP and 5 Normal seats

        // Display available seats before booking
        system.showSeats();

        System.out.print(s:"Enter the number of customers: ");
        int customerCount = scanner.nextInt();
        scanner.nextLine(); // Consume newline

        BookingTask[] customers = new BookingTask[customerCount];

        for (int i = 0; i < customerCount; i++) {
            System.out.print(s:"Enter customer name: ");
            String name = scanner.nextLine();

            System.out.print(s:"Enter customer type (VIP/Normal): ");
```



```java
            String type = scanner.nextLine();

            customers[i] = new BookingTask(system, name, type);

            // Assign priority to VIP customers
            if (type.equalsIgnoreCase(anotherString:"VIP")) {
                customers[i].setPriority(Thread.MAX_PRIORITY);
            }
        }

        // Start all threads
        for (BookingTask customer : customers) {
            customer.start();
        }

        scanner.close();
    }
}
```

Output:



```
PS C:\Users\rajan\OneDrive\Desktop\JAVA EXPERIMENT = 04>
Total available VIP seats: 5
Total available Normal seats: 5
Enter the number of customers: 2
Enter customer name: RAJAN
Enter customer type (VIP/Normal): VIP
Enter customer name: AYUSH
Enter customer type (VIP/Normal): NORMAL
RAJAN (VIP) booked a seat. VIP Seats remaining: 4
AYUSH  (Normal) booked a seat. Normal Seats remaining: 4
PS C:\Users\rajan\OneDrive\Desktop\JAVA EXPERIMENT = 04>
```