LAB RECORD

23CSE111- Object Oriented Programming

Submitted by

CH.SC.U4CSE24146 -M.SRIJA

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

AMRITA VISHWA VIDYAPEETHAM

AMRITA SCHOOL OF COMPUTING

CHENNAI

March - 2025

AMRITA VISHWA VIDYAPEETHAM

AMRITA SCHOOL OF COMPUTING, CHENNAI

BONAFIDE CERTIFICATE

This is to certify that the Lab Record work for 23CSE111-Object Oriented Programming Subject submitted by ***CH.SC.U4CSE24126 – M.SRIJA*** in **"Computer Science and Engineering"** is a Bonafide record of the work carried out under my guidance and supervision at Amrita School of Computing, Chennai.

This Lab examination held on / /2025

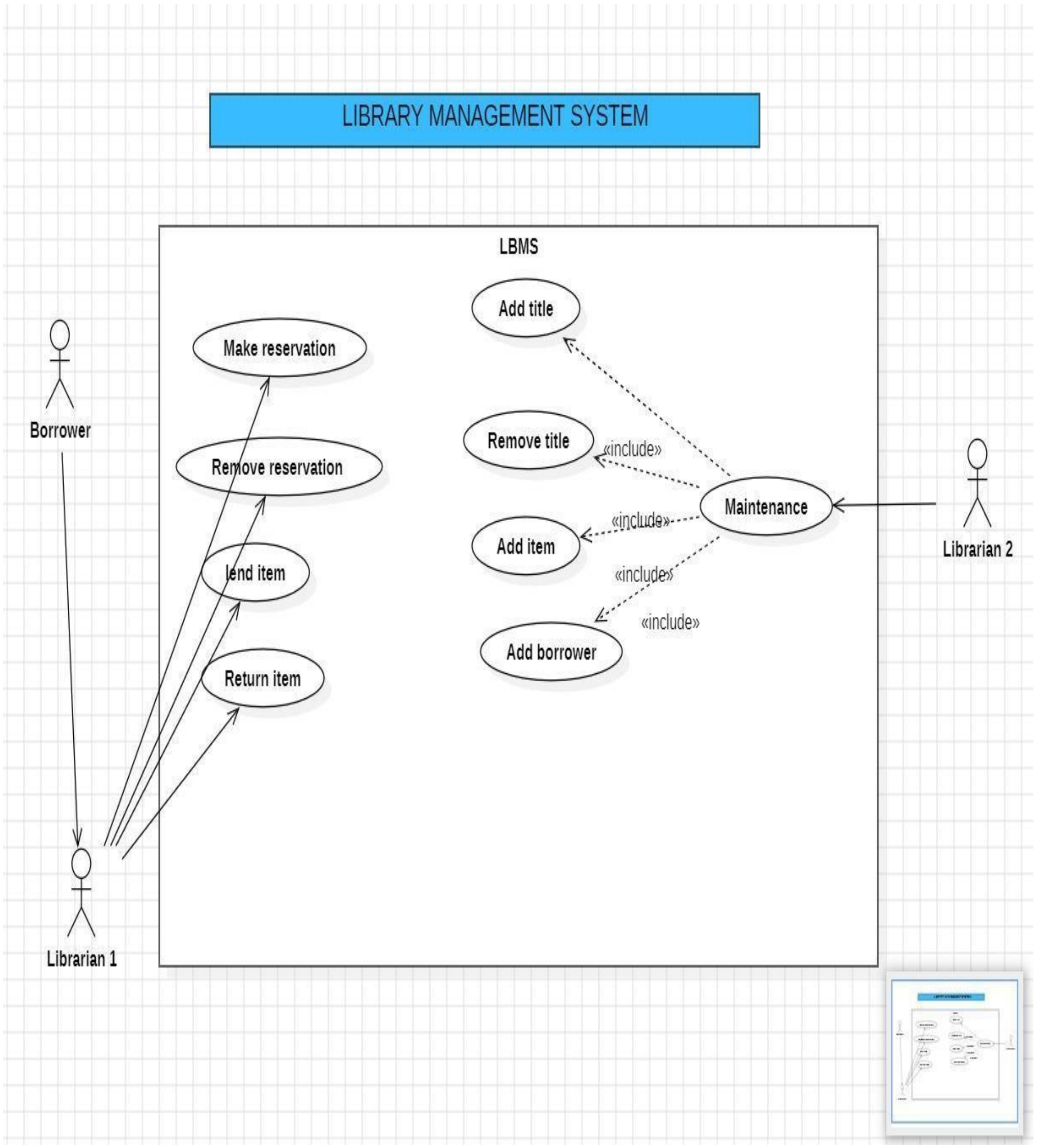
Internal Examiner 1 Internal Examiner 2

INDEX

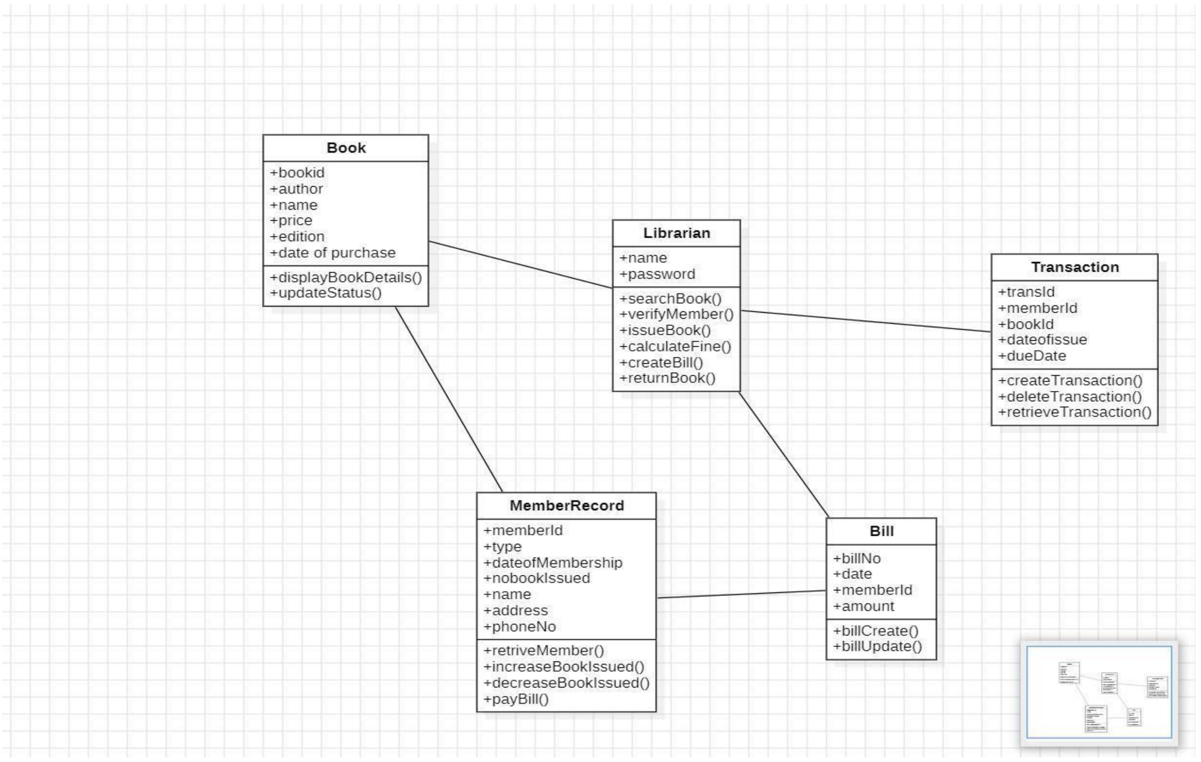
UML DIAGRAMS

1. LIBRARY MANAGEMENT SYSTEM

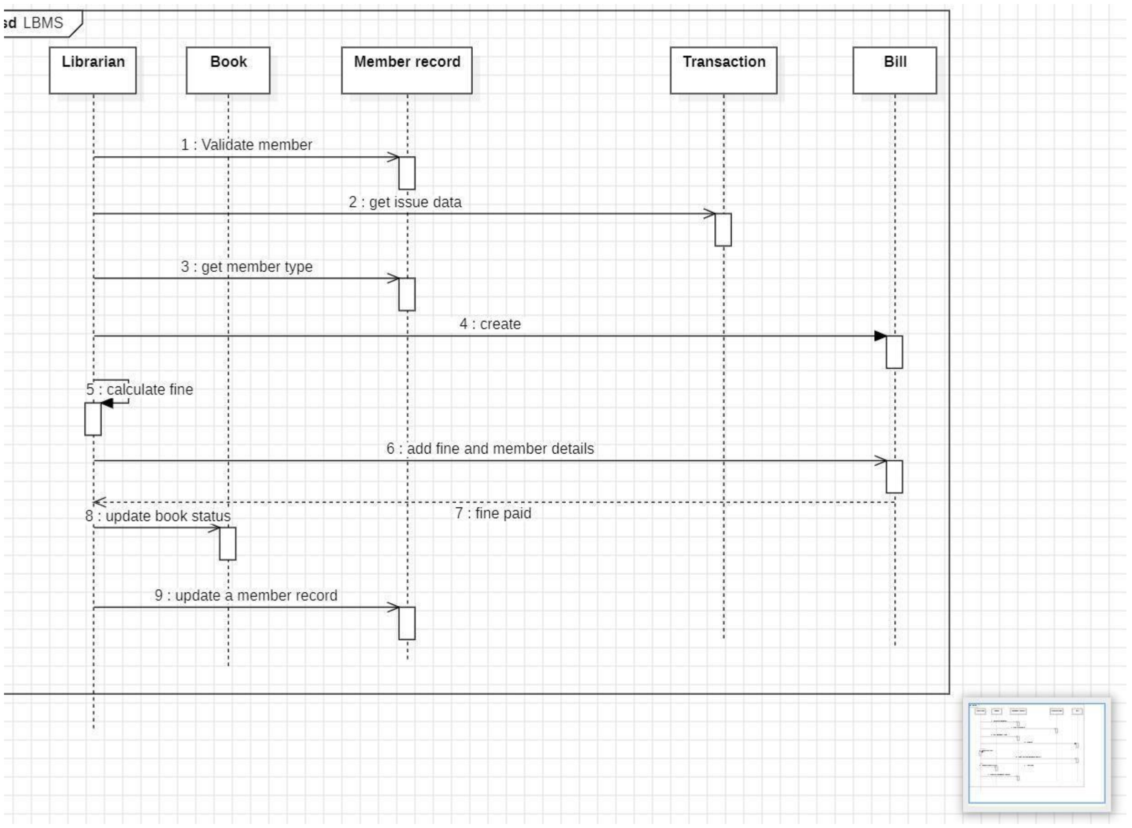
1.a) Use Case Diagram:



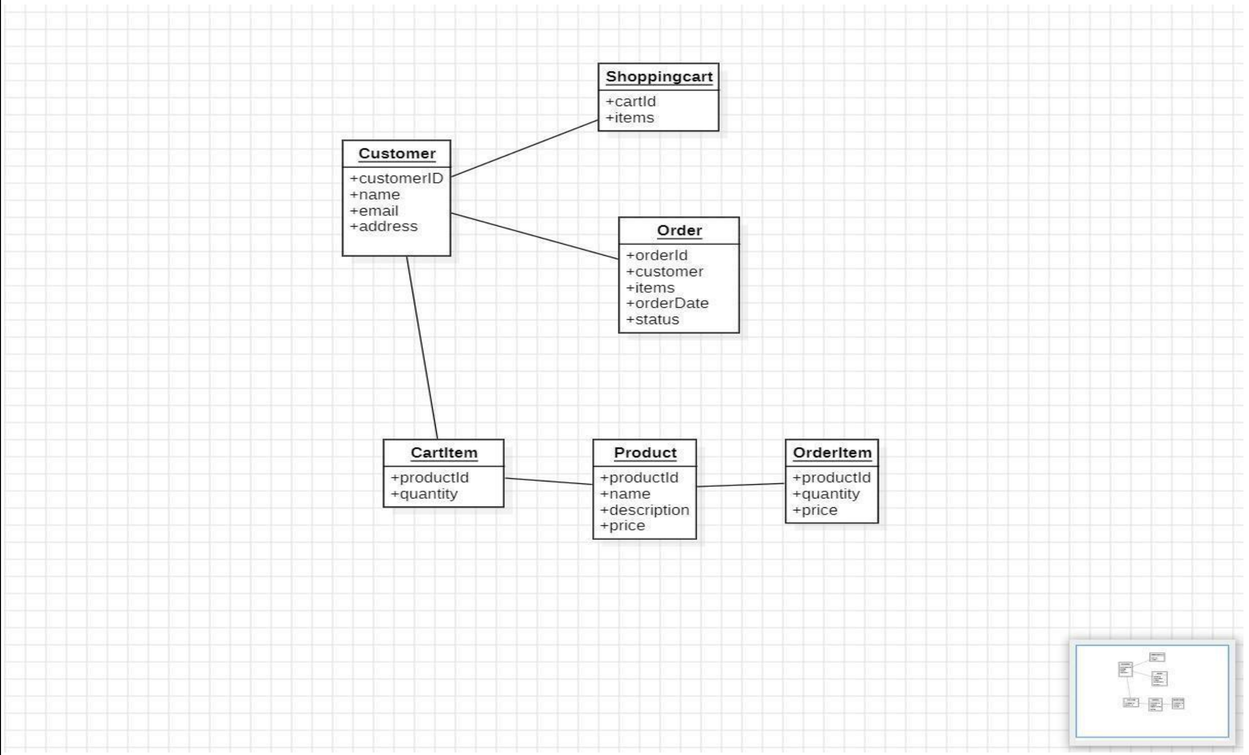
1.b) Class Diagram:



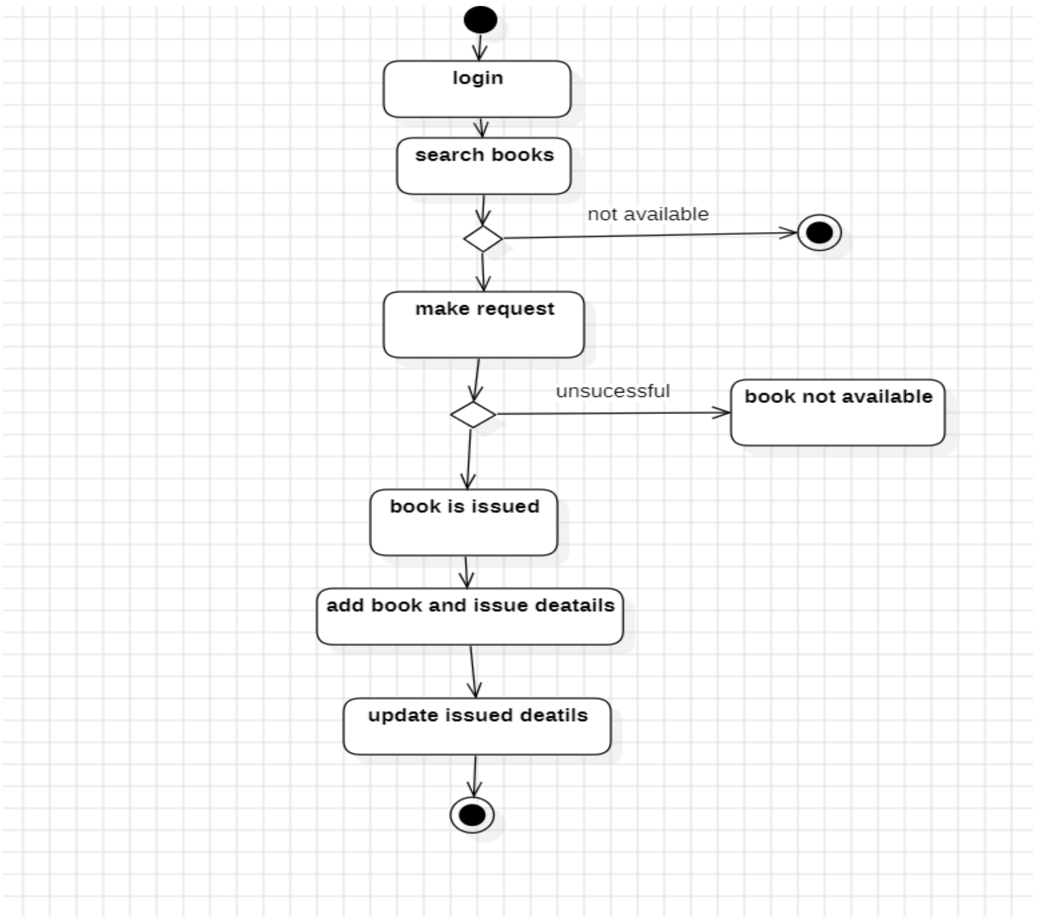
1.c) Sequence Diagram:



1.d) Object Diagram:

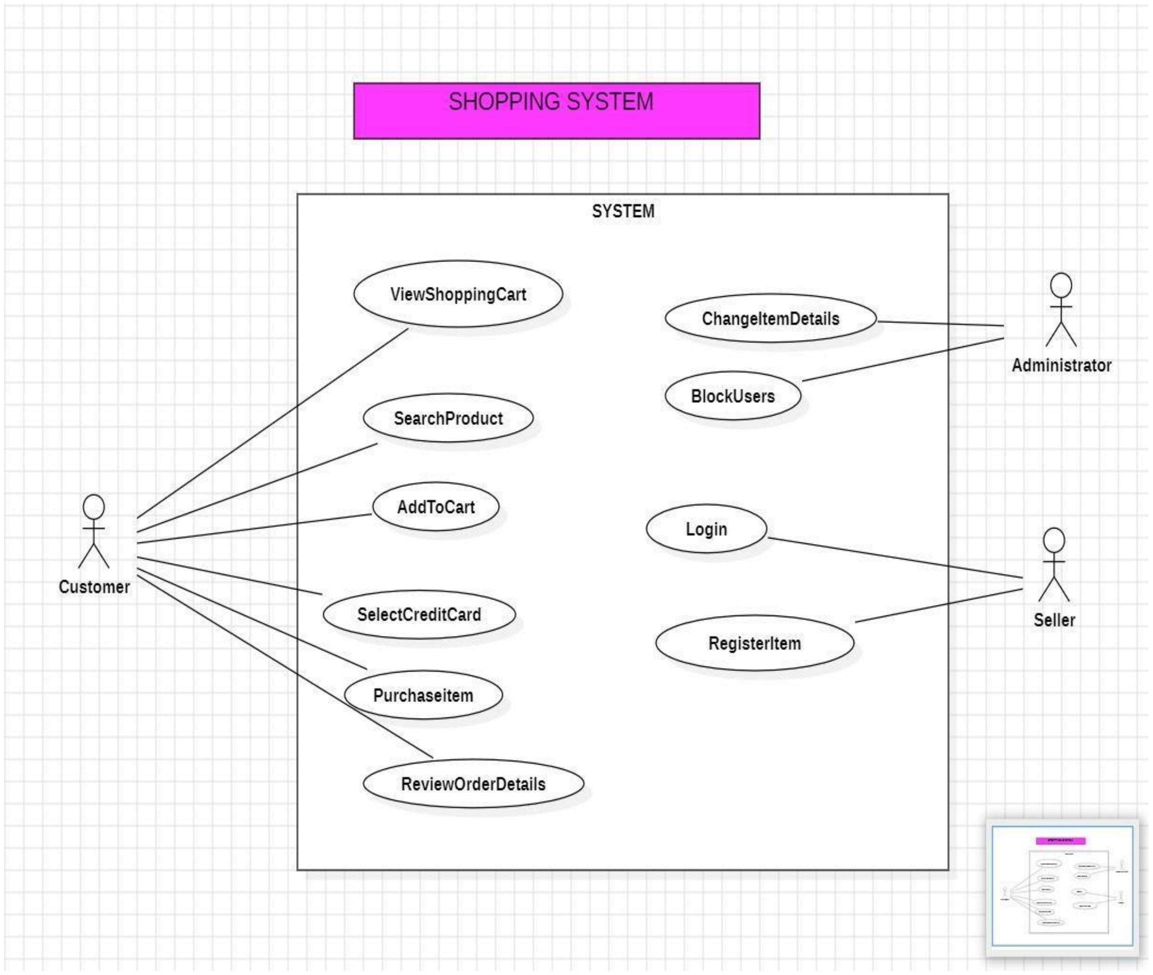


1.e) State-Activity Diagram:

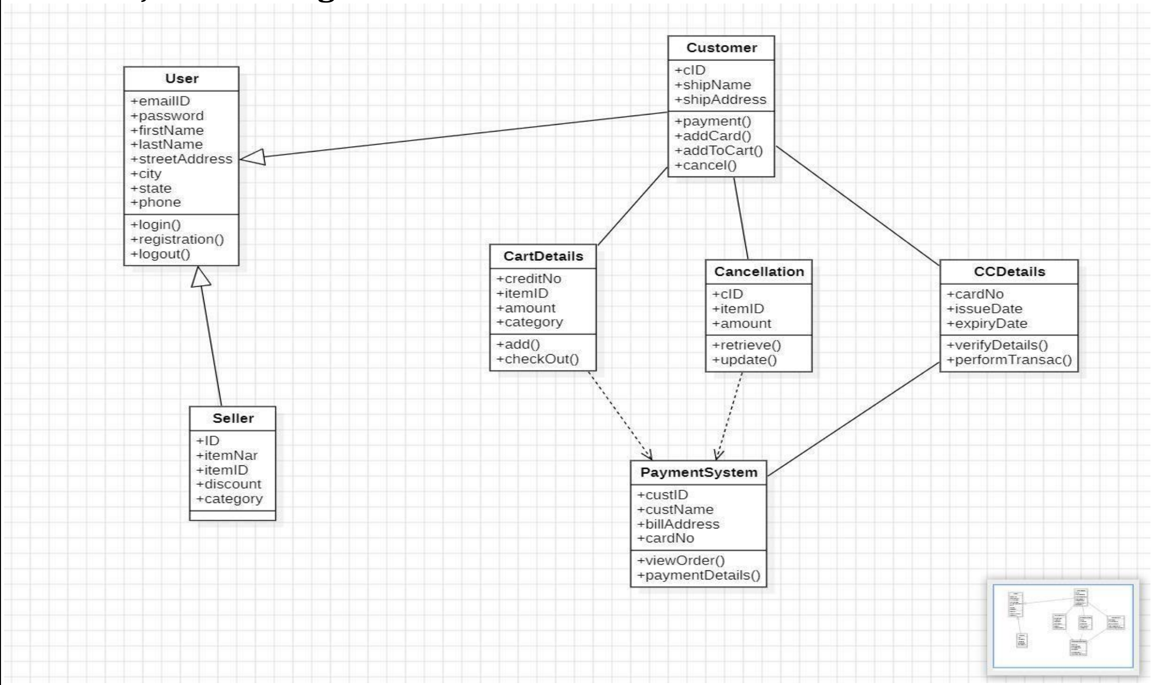


2. ONLINE SHOPPING SYSTEM

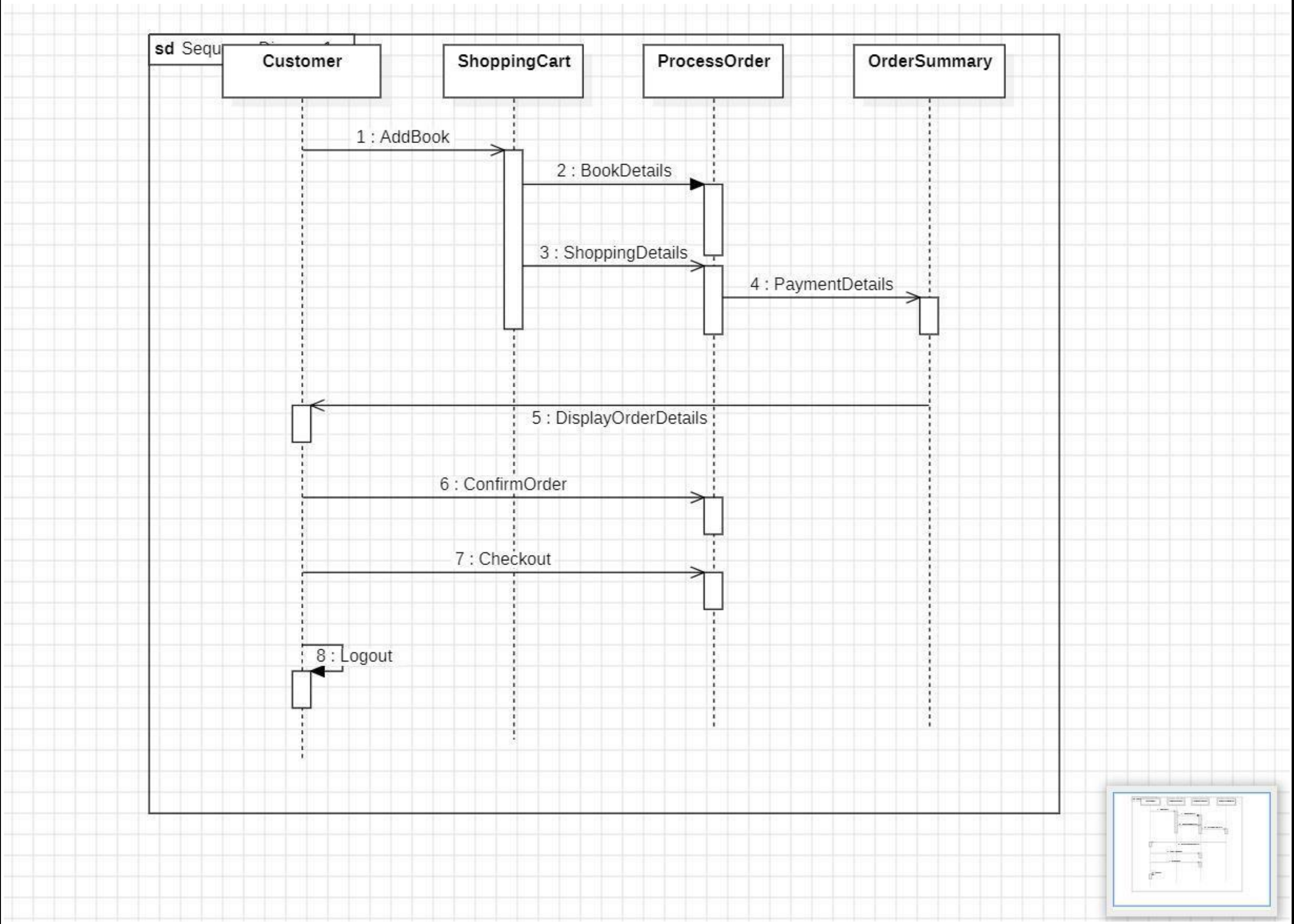
2.a) Use Case Diagram:



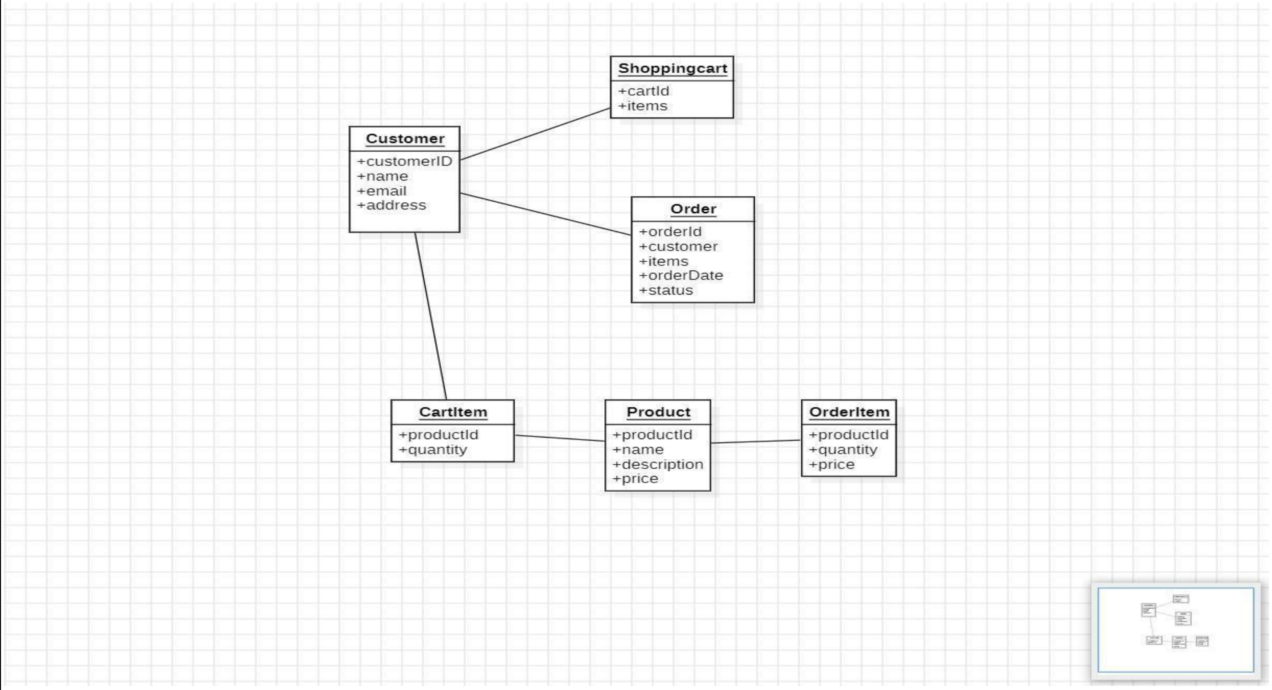
2.b) Class Diagram:



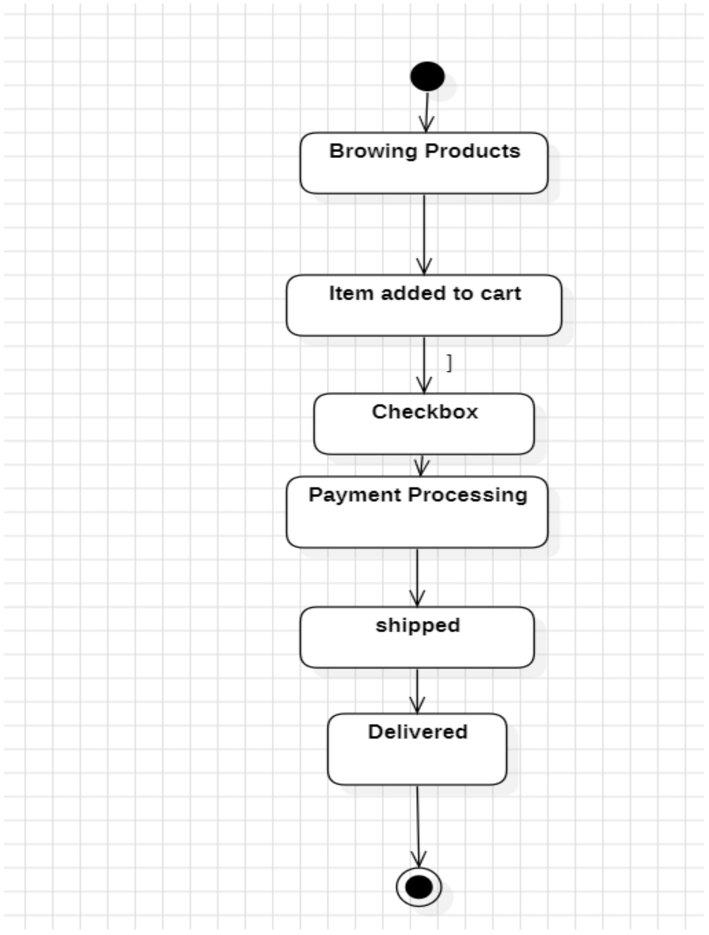
2.c) Sequence Diagram:



2.d) Object Diagram:



2.e) State-Activity Diagram:

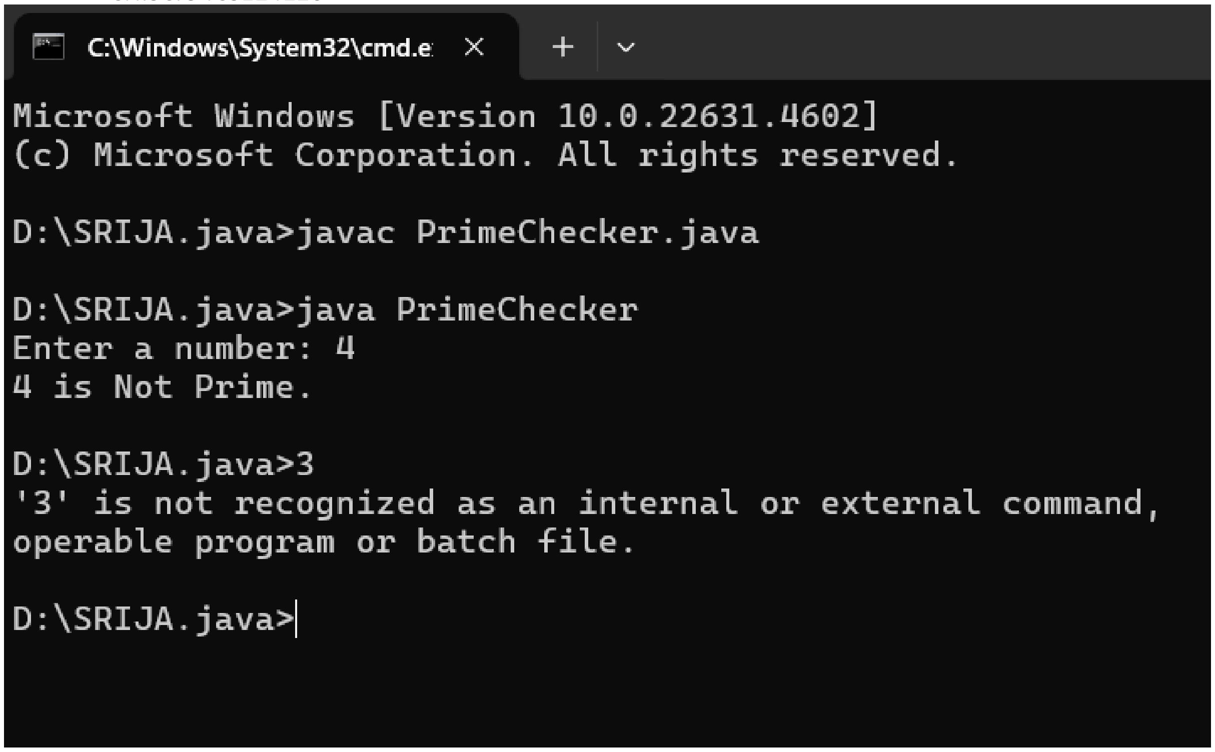


3. Basic Java Programs

1) Java Program to Check if a Number is Prime CODE:

```
import java.util.Scanner;

public class PrimeChecker {
  public static void main(String[]
    args) { Scanner scanner = new
    Scanner(System.in);

    System.out.print("Enter a
    number: "); int num = scan-
    ner.nextInt();
    boolean isPrime = true;

    if (num <= 1) {
      isPrime = false;
    } else {
      for (int i = 2; i <= num /
        2; i++) { if (num % i ==
        0) {
          isPrime =
          false; break;
        }
      }
    }

    if (isPrime)
      System.out.println(num + " is
    Prime."); else
      System.out.println(num + " is Not Prime.");

    scanner.close();
  }
}
```

OUTPUT:



```
C:\Windows\System32\cmd.e

Microsoft Windows [Version 10.0.22631.4602]
(c) Microsoft Corporation. All rights reserved.

D:\SRIJA.java>javac PrimeChecker.java

D:\SRIJA.java>java PrimeChecker
Enter a number: 4
4 is Not Prime.

D:\SRIJA.java>3
'3' is not recognized as an internal or external command,
operable program or batch file.

D:\SRIJA.java>
```

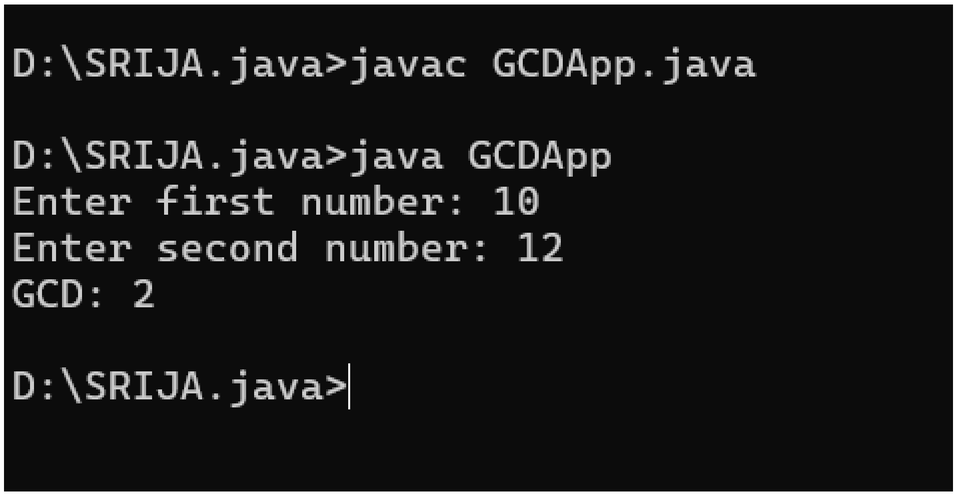
2) Find the Greatest Common Divisor (GCD) Using a Class CODE:

```
import java.util.Scanner;

class GCDCalcula-
  tor { int a, b;

  void inputNumbers() {
    Scanner scanner = new Scan-
    ner(System.in); Sys-
    tem.out.print("Enter first number:
    ");
    a = scanner.nextInt(); Sys-
    tem.out.print("Enter second num-
    ber: "); b = scanner.nextInt();
  }

  void findGCD() {
    int num1 = a, num2
    = b; while (num2 !=
    0) {
      int temp = num2;
      num2 = num1 %
      num2; num1 =
      temp;
    }
```

```
        System.out.println("GCD: " + num1);
    }
}

public class GCDApp {
    public static void main(String[]
        args) { GCDCalculator gcd = new
        GCDCalculator();
        gcd.inputNumbers();
        gcd.findGCD();
```

```
}
}
```

OUTPUT:

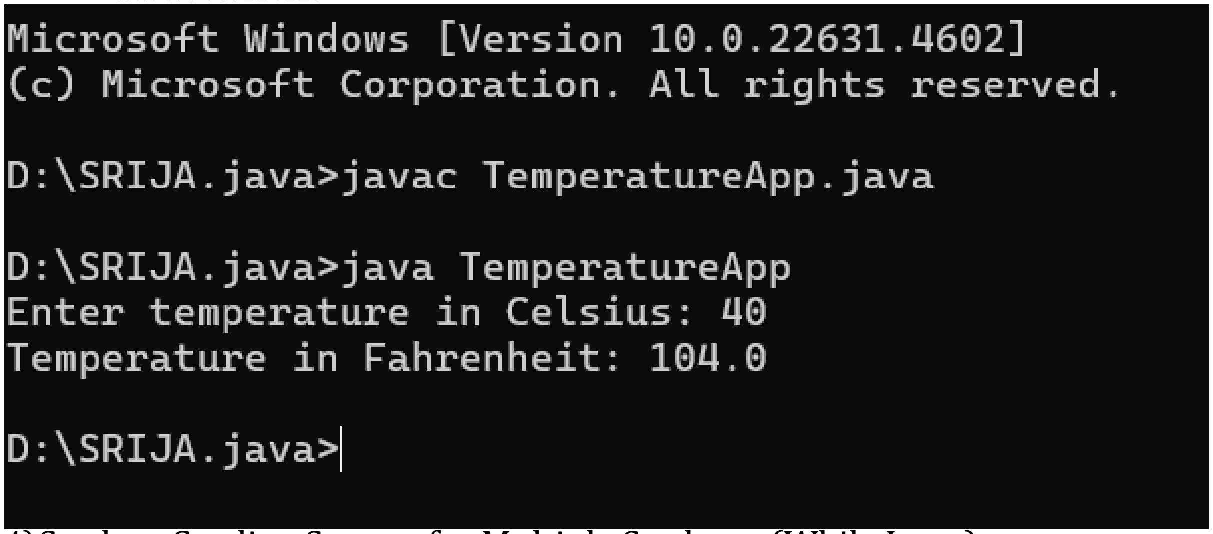


```
D:\SRIJA.java>javac GCDApp.java

D:\SRIJA.java>java GCDApp
Enter first number: 10
Enter second number: 12
GCD: 2

D:\SRIJA.java>
```

3) Convert Celsius to Fahrenheit CODE:

```
import java.util.Scanner;

class TemperatureConvert-
  er { double celsius;

  void inputTemperature() {
    Scanner scanner = new Scan-
    ner(System.in); System.out.print("Enter
    temperature in Celsius: "); celsius = scan-
    ner.nextDouble();
  }

  void convert() {
    double fahrenheit = (celsius * 9/5) + 32; Sys-
    tem.out.println("Temperature in Fahrenheit: " + fahr-
    enheit);
  }
}

public class TemperatureApp {
  public static void main(String[] args) {
    TemperatureConverter temp = new TemperatureConverter();
    temp.inputTemperature();
    temp.convert();
  }
}
```

OUTPUT:



```
Microsoft Windows [Version 10.0.22631.4602]
(c) Microsoft Corporation. All rights reserved.

D:\SRIJA.java>javac TemperatureApp.java

D:\SRIJA.java>java TemperatureApp
Enter temperature in Celsius: 40
Temperature in Fahrenheit: 104.0

D:\SRIJA.java>
```

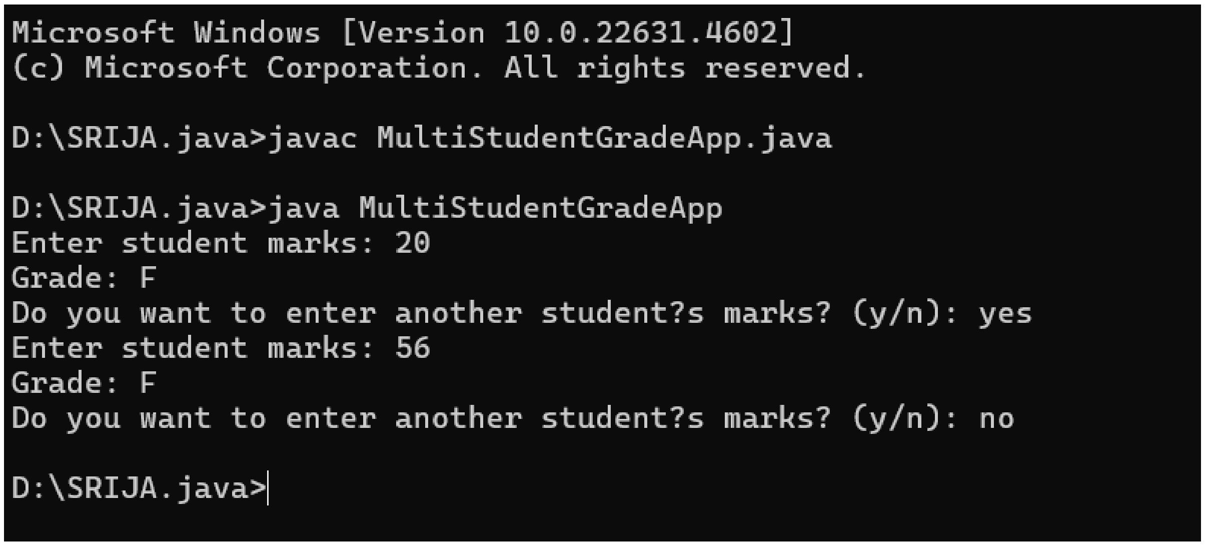
4) Student Grading System for Multiple Students (While Loop)

Problem: Allow multiple students to enter their marks using a **while loop**, and print their grades. CODE:

```
import java.util.Scanner;

class MultipleStudentsGrad-
  ing { void assignGrades() {
    Scanner scanner = new Scan-
    ner(System.in); char choice;

    while (true) {
      System.out.print("Enter student
      marks: "); int marks = scan-
      ner.nextInt();

      if (marks >= 90) Sys-
        tem.out.println("Grade:
        A");
      else if (marks >= 80) Sys-
        tem.out.println("Grade:
        B");
      else if (marks >= 70) Sys-
        tem.out.println("Grade:
        C");
      else if (marks >= 60) Sys-
        tem.out.println("Grade:
        D");
      else
        System.out.println("Grade:  F");

      System.out.print("Do you want to enter another student’s marks? (y/n): ");
```

```
         choice = scan-
         ner.next().charAt(0); if
         (choice == 'n' || choice ==
         'N')
            break;
      }
   }
}

public class MultiStudentGrade-
   App { public static void
   main(String[] args) {
      MultipleStudentsGrading msg = new MultipleStudentsGrading();
      msg.assignGrades();
   }
```

} OUT-
PUT:



```
Microsoft Windows [Version 10.0.22631.4602]
(c) Microsoft Corporation. All rights reserved.

D:\SRIJA.java>javac MultiStudentGradeApp.java

D:\SRIJA.java>java MultiStudentGradeApp
Enter student marks: 20
Grade: F
Do you want to enter another student?s marks? (y/n): yes
Enter student marks: 56
Grade: F
Do you want to enter another student?s marks? (y/n): no

D:\SRIJA.java>
```

5) Grading System for Multiple Subjects (Do-While Loop)

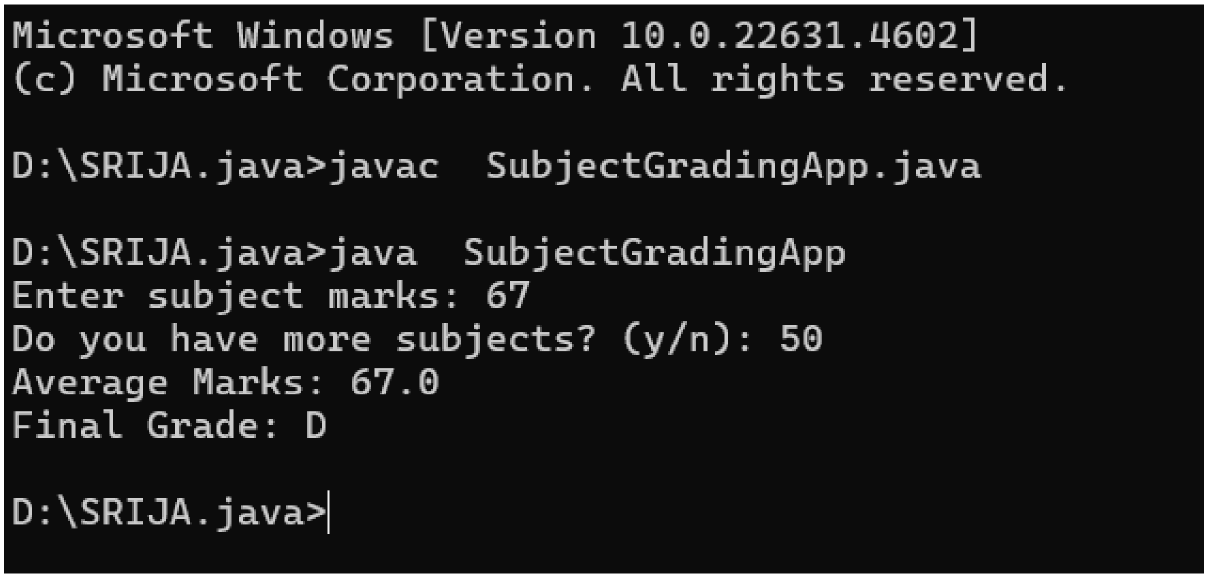
Problem: Calculate the **average marks and grade** for a student across multiple subjects using a **do-while loop**.

CODE:

```
import java.util.Scanner;

class SubjectGrading {
  void calculateAverageGrade() {
    Scanner scanner = new Scan-
    ner(System.in); int totalMarks = 0,
    subjectCount = 0, marks; char
    choice;

    do {
      System.out.print("Enter subject
      marks: "); marks = scan-
      ner.nextInt();
      totalMarks += marks; sub-
      jectCount++;

      System.out.print("Do you have more subjects?
      (y/n): "); choice = scanner.next().charAt(0);
    } while (choice == 'y' || choice == 'Y');

    double average = (double) totalMarks / subjectCount; Sys-
    tem.out.println("Average Marks: " + average);

    if (average >= 90) Sys-
      tem.out.println("Final Grade:
      A");
    else if (average >= 80) Sys-
      tem.out.println("Final Grade:
      B");
    else if (average >= 70) Sys-
      tem.out.println("Final Grade:
      C");
    else if (average >= 60) Sys-
      tem.out.println("Final Grade:
      D");
    else
      System.out.println("Final Grade: F");
```

```
}
}

public class SubjectGradingApp {
  public static void main(String[]
    args) { SubjectGrading sg = new
    SubjectGrading();
    sg.calculateAverageGrade();
  }
}
```

OUTPUT:



```
Microsoft Windows [Version 10.0.22631.4602]
(c) Microsoft Corporation. All rights reserved.

D:\SRIJA.java>javac  SubjectGradingApp.java

D:\SRIJA.java>java  SubjectGradingApp
Enter subject marks: 67
Do you have more subjects? (y/n): 50
Average Marks: 67.0
Final Grade: D

D:\SRIJA.java>
```

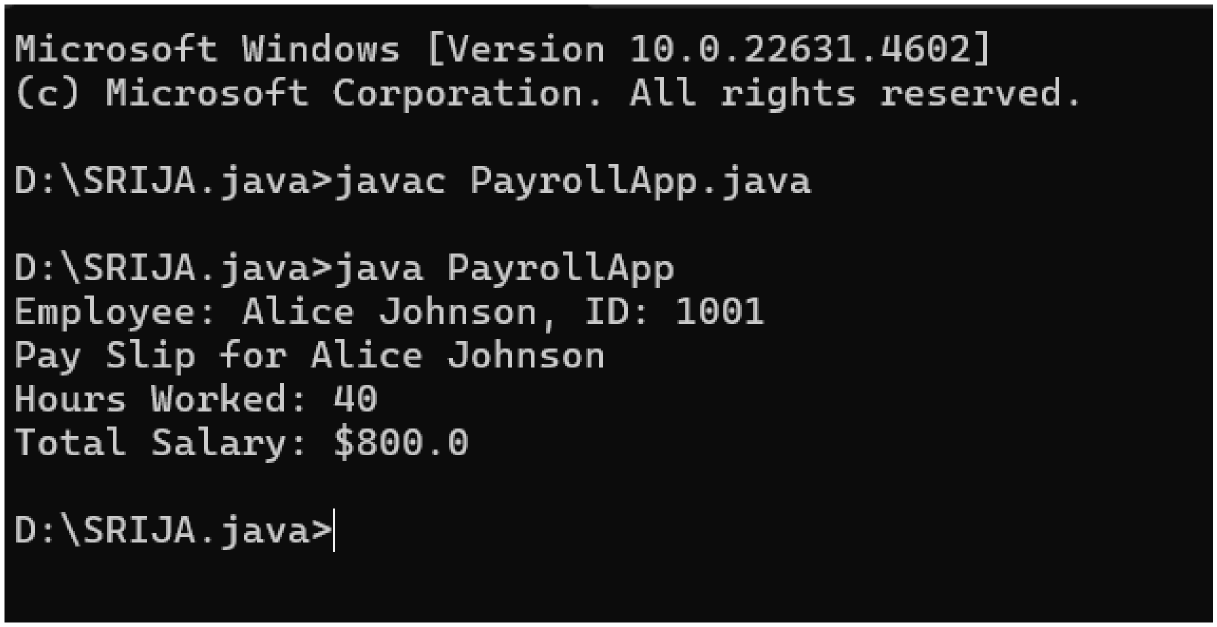
6) **Employee Payroll System**

Problem: Create classes for **Employee, Salary, and Payroll**. Implement methods to calculate salary based on working hours, generate pay slips, and apply bonuses.

CODE:

```
class Em-
  ployee {
  String
  name; int
  empId;
  double hourlyRate;

  Employee(String name, int empId, double
    hourlyRate) { this.name = name;
    this.empId = empId;
    this.hourlyRate = hourly-
    Rate;
  }
  double calculateSalary(int hours-
    Worked) { return hoursWorked *
    hourlyRate;
  }

  void displayInfo() {
    System.out.println("Employee: " + name + ", ID: " + empId);
  }
}

class Payroll {
  static void generatePaySlip(Employee employee, int hoursWorked) {
        double salary = employ-
        ee.calculateSalary(hoursWorked); Sys-
        tem.out.println("Pay Slip for " + employee.name); Sys-
        tem.out.println("Hours Worked: " + hoursWorked);
        System.out.println("Total Salary: $" + salary);
    }
}

public class PayrollApp {
    public static void main(String[] args) {
        Employee emp1 = new Employee("Alice Johnson",
        1001, 20); emp1.displayInfo();
        Payroll.generatePaySlip(emp1, 40);
    }
}
```

OUTPUT:



```
Microsoft Windows [Version 10.0.22631.4602]
(c) Microsoft Corporation. All rights reserved.

D:\SRIJA.java>javac PayrollApp.java

D:\SRIJA.java>java PayrollApp
Employee: Alice Johnson, ID: 1001
Pay Slip for Alice Johnson
Hours Worked: 40
Total Salary: $800.0

D:\SRIJA.java>
```

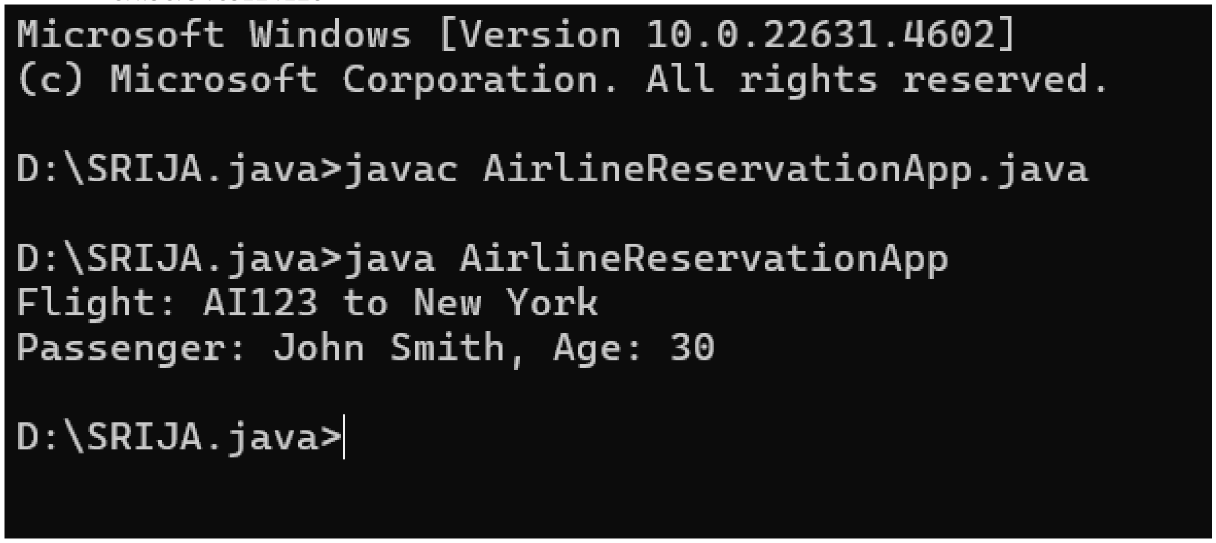
7) Airline Reservation System

Problem: Create classes for **Flight, Passenger, Ticket, and Booking**. Implement methods to book tickets, cancel reservations, and display flight details.

CODE:

```
import java.util.ArrayList;

class Flight {
    String
    flightNumber;
    String destina-
    tion;

    Flight(String flightNumber, String des-
      tination) { this.flightNumber =
      flightNumber; this.destination =
      destination;
    }

    void displayFlightDetails() {
      System.out.println("Flight: " + flightNumber + " to " + destination);
    }
}
```

```
class Passe-
  ger { String
  name; int
  age;

  Passenger(String name, int
    age) { this.name = name;
    this.age = age;
  }

  void displayPassengerInfo() { Sys-
    tem.out.println("Passenger: " + name + ", Age:
    " + age);
  }
}

class Ticket {
  Flight
  flight;
  Passenger passenger;

  Ticket(Flight flight, Passenger pas-
    senger) { this.flight = flight;
    this.passenger = passenger;
  }

  void displayTicketDetails() {
    flight.displayFlightDetails(); pas-
    senger.displayPassengerInfo();
  }
}

public class AirlineReservation-
  App { public static void
  main(String[] args) {
    Flight flight = new Flight("AI123", "New
    York"); Passenger passenger = new Passen-
    ger("John Smith", 30);

    Ticket ticket = new Ticket(flight, passen-
    ger); ticket.displayTicketDetails();
  }
}
```



```
Microsoft Windows [Version 10.0.22631.4602]
(c) Microsoft Corporation. All rights reserved.

D:\SRIJA.java>javac AirlineReservationApp.java

D:\SRIJA.java>java AirlineReservationApp
Flight: AI123 to New York
Passenger: John Smith, Age: 30

D:\SRIJA.java>
```

8) **University Management System**

Problem: Create classes for **Department, Faculty, Student, and Exam**. Implement methods to assign faculty to courses, conduct exams, and manage student records.

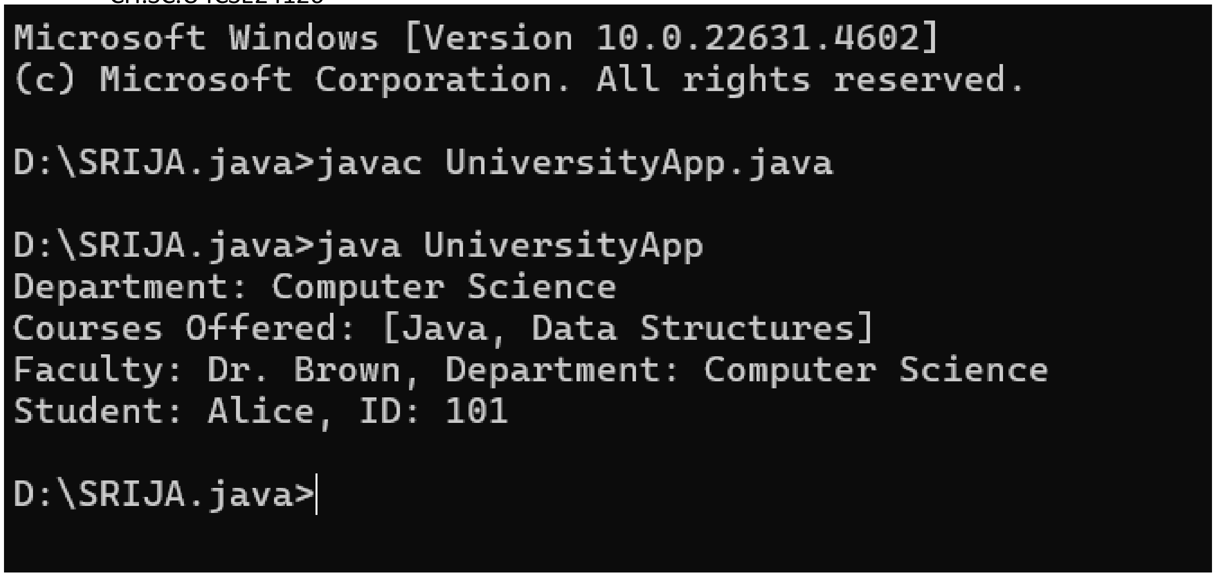
CODE:

```
import java.util.ArrayList;

class Department
  { String name;
  ArrayList<String> courses = new ArrayList<>();

  Department(String name) {
    this.name = name;
  }

  void addCourse(String course) {
    courses.add(course);
  }

  void displayDetails() { Sys-
    tem.out.println("Department: " + name);
    System.out.println("Courses Offered: " +
    courses);
  }
}

class Facul-
  ty { String
  name;
  String department;
```

```
Faculty(String name, String de-
  partment) { this.name = name;
  this.department = department;
}

void displayInfo() {
  System.out.println("Faculty: " + name + ", Department: " + department);
}
```

```
}

class    Stu-
  dent       {
  String
  name;  int
  id;

  Student(String name, int id)
    { this.name = name;
    this.id = id;
  }

  void displayInfo() {
    System.out.println("Student: " + name + ", ID: " + id);
  }
}

public class UniversityApp {
  public static void main(String[] args) {
    Department csDept = new Department("Computer Science");
    csDept.addCourse("Java");
    csDept.addCourse("Data Structures");

    Faculty prof = new Faculty("Dr. Brown", "Computer
    Science"); Student student = new Student("Alice",
    101);

    csDept.displayDetails();
    prof.displayInfo(); stu-
    dent.displayInfo();
  }
}
```

OUTPUT:



```
Microsoft Windows [Version 10.0.22631.4602]
(c) Microsoft Corporation. All rights reserved.

D:\SRIJA.java>javac UniversityApp.java

D:\SRIJA.java>java UniversityApp
Department: Computer Science
Courses Offered: [Java, Data Structures]
Faculty: Dr. Brown, Department: Computer Science
Student: Alice, ID: 101

D:\SRIJA.java>
```

9) **Hotel Booking System**

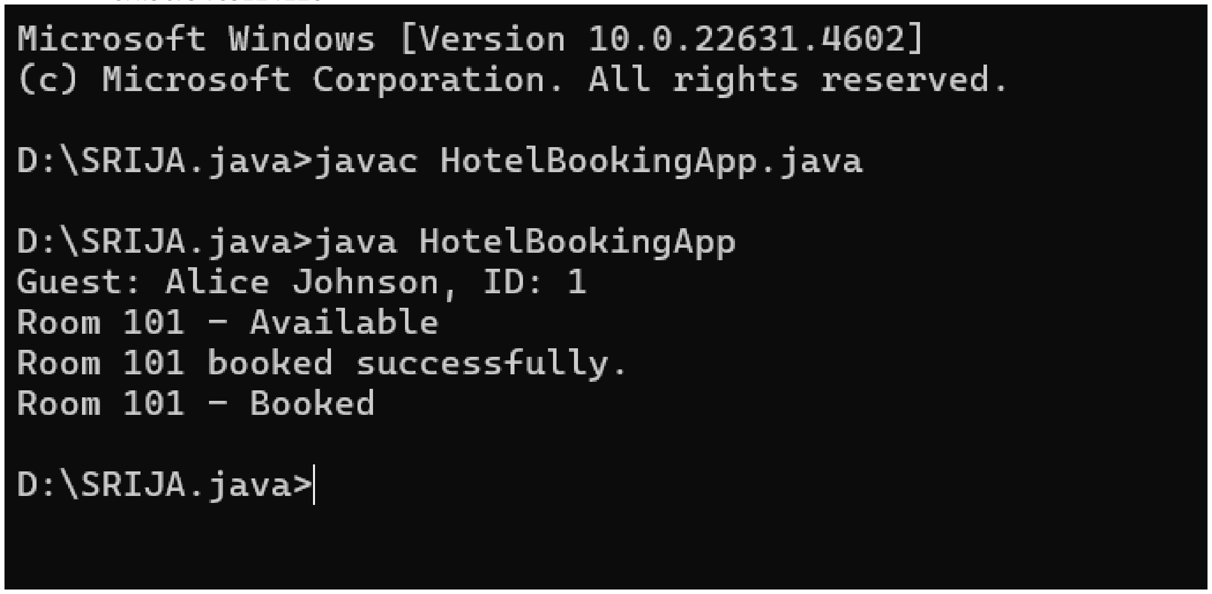
Problem: Create classes for **Hotel, Room, Guest, and Reservation**. Implement methods to check room availability, book rooms, and manage guest details.

CODE:

```
import java.util.ArrayList;

class Room {
  int roomNumber;
  boolean isBooked;

  Room(int roomNumber) {
    this.roomNumber = room-
    Number; this.isBooked =
    false;
  }

  void
    bookRoom()
    { if (!is-
    Booked) {
      isBooked = true;
      System.out.println("Room " + roomNumber + " booked successfully.");
    } else {
      System.out.println("Room " + roomNumber + " is already booked.");
    }
  }

  void displayRoomInfo() {
    System.out.println("Room " + roomNumber + " - " + (isBooked ? "Booked" :
    "Available"));
  }
}

class Guest
  { String
  name; int
  guestId;

  Guest(String name, int guestId)
    { this.name = name;
    this.guestId = guestId;
  }

  void displayGuestInfo() {
    System.out.println("Guest: " + name + ", ID: " + guestId);
  }
}
```

```
public class HotelBookingApp {
  public static void main(String[]
    args) { Room room1 = new
    Room(101); Room room2 =
    new Room(102);

    Guest guest1 = new Guest("Alice Johnson", 1);

    guest1.displayGuestInfo();
    room1.displayRoomInfo();
    room1.bookRoom();
    room1.displayRoomInfo();
  }
}
```

OUTPUT:



```
Microsoft Windows [Version 10.0.22631.4602]
(c) Microsoft Corporation. All rights reserved.

D:\SRIJA.java>javac HotelBookingApp.java

D:\SRIJA.java>java HotelBookingApp
Guest: Alice Johnson, ID: 1
Room 101 - Available
Room 101 booked successfully.
Room 101 - Booked

D:\SRIJA.java>
```

10)**Online Shopping System**

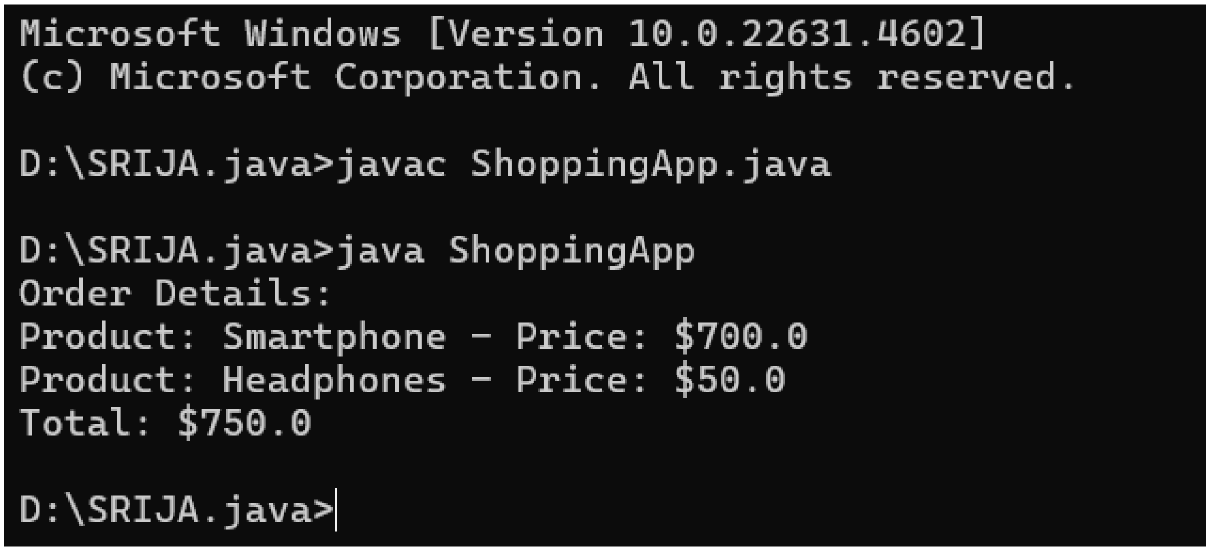
Problem: Define classes for **User, Product, Order, and Payment**. Implement methods to browse products, place orders, process payments, and track shipments.

CODE:

```
import java.util.ArrayList;

class  Prod-
  uct        {
  String
  name;
  double
  price;

  Product(String name, double price)
    { this.name = name;
    this.price = price;
  }

  void displayProduct() {
    System.out.println("Product: " + name + " - Price: $" + price);
  }
}

class Order {
  ArrayList<Product> products = new ArrayList<>();

  void addProduct(Product product) {
    products.add(product);
  }
```

```
void displayOrderDetails() {
  System.out.println("Order
  Details:"); double total = 0;
  for (Product p : products) {
    p.displayProduct();
    total += p.price;
  }
  System.out.println("Total: $" + total);
  }
}

public class ShoppingApp {
  public static void main(String[] args) {
    Product p1 = new Product("Smartphone",
    700); Product p2 = new Prod-
    uct("Headphones", 50);

    Order order = new Order(); or-
    der.addProduct(p1); or-
    der.addProduct(p2); or-
    der.displayOrderDetails();
  }
}
```

OUTPUT:



```
Microsoft Windows [Version 10.0.22631.4602]
(c) Microsoft Corporation. All rights reserved.

D:\SRIJA.java>javac ShoppingApp.java

D:\SRIJA.java>java ShoppingApp
Order Details:
Product: Smartphone - Price: $700.0
Product: Headphones - Price: $50.0
Total: $750.0

D:\SRIJA.java>
```

}

4. INHERITANCE

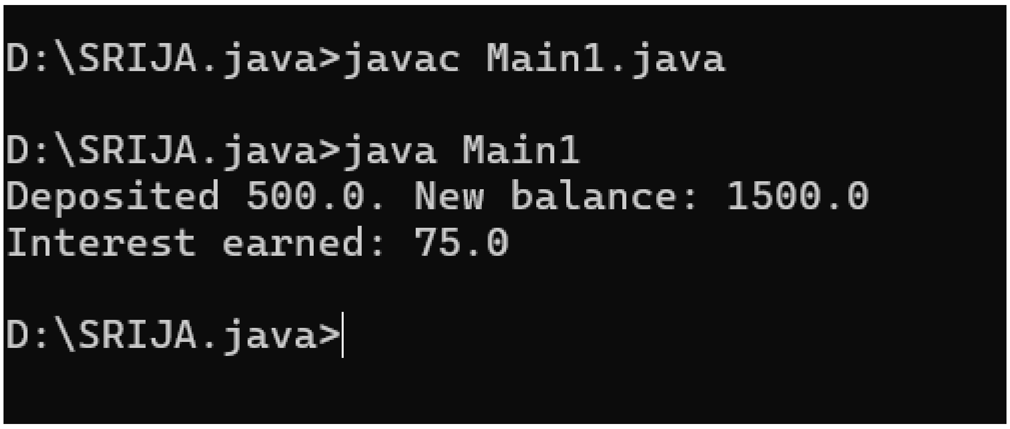
SINGLE INHERITANCE

4.a)Bank Account System

CODE:

```
class BankAccount {
    protected String accountHolder;
    protected double balance;

    public BankAccount(String accountHolder, double balance) {
        this.accountHolder = accountHolder;
        this.balance = balance;
    }

    public void deposit(double amount) {
        balance += amount;
        System.out.println("Deposited " + amount + ". New balance: " + balance);
    }

    public void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
            System.out.println("Withdrew " + amount + ". New balance: " + bal-
ance);
        } else {
            System.out.println("Insufficient balance.");
        }
    }
}

class SavingsAccount extends BankAccount {
    private double interestRate;

    public SavingsAccount(String accountHolder, double balance, double in-
terestRate) {
        super(accountHolder, balance);
        this.interestRate = interestRate;
    }

    public void calculateInterest() {
        double interest = balance * interestRate / 100;
        System.out.println("Interest earned: " + interest);
    }
}

public class Main1 {
    public static void main(String[] args) {
        SavingsAccount savings = new SavingsAccount("Alice", 1000, 5);
        savings.deposit(500);
        savings.calculateInterest();
    }
}
```

OUTPUT:



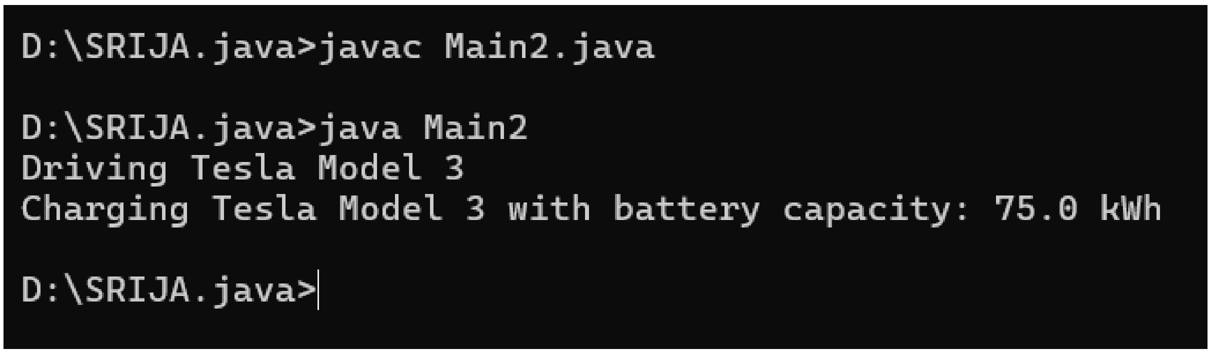
```
D:\SRIJA.java>javac Main1.java

D:\SRIJA.java>java Main1
Deposited 500.0. New balance: 1500.0
Interest earned: 75.0

D:\SRIJA.java>
```

4.b)Vehicle and ElectricCar

CODE

```
class Vehicle {
   protected String brand;
   protected String model;

   public Vehicle(String brand, String model) {
      this.brand = brand;
      this.model = model;
   }

   public void drive() {
      System.out.println("Driving " + brand + " " + model);
   }
}

class ElectricCar extends Vehicle {
   private double batteryCapacity;

   public ElectricCar(String brand, String model, double batteryCapacity) {
        super(brand, model);
        this.batteryCapacity = batteryCapacity;
    }

    public void charge() {
        System.out.println("Charging " + brand + " " + model + " with battery ca-
pacity: " + batteryCapacity + " kWh");
    }
}

public class Main2 {
    public static void main(String[] args) {
        ElectricCar electricCar = new ElectricCar("Tesla", "Model 3", 75);
        electricCar.drive();
        electricCar.charge();
    }
}
```

OUTPUT:



```
D:\SRIJA.java>javac Main2.java

D:\SRIJA.java>java Main2
Driving Tesla Model 3
Charging Tesla Model 3 with battery capacity: 75.0 kWh

D:\SRIJA.java>
```

5.Multilevel Inheritance

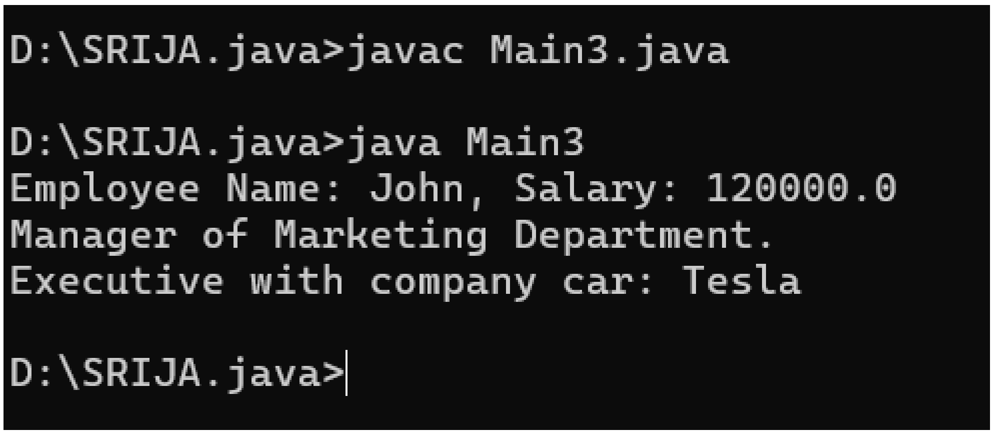
5.a)Employee, Manager, and Executive

CODE:

```
class Employee {
    protected String name;
    protected double salary;

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

    public void displayDetails() {
        System.out.println("Employee Name: " + name + ", Salary: " + salary);
    }
}

class Manager extends Employee {
    private String department;

    public Manager(String name, double salary, String department) {
        super(name, salary);
        this.department = department;
    }

    public void displayManagerDetails() {
        System.out.println("Manager of " + department + " Department.");
    }
}
```

```
class Executive extends Manager {
   private String companyCar;


   public Executive(String name, double salary, String department, String com-
panyCar) {
      super(name, salary, department);
      this.companyCar = companyCar;
   }


   public void displayExecutiveDetails() {
      System.out.println("Executive with company car: " + companyCar);
   }
}


public class Main3 {
   public static void main(String[] args) {
      Executive exec = new Executive("John", 120000, "Marketing", "Tesla");
      exec.displayDetails();
      exec.displayManagerDetails();
      exec.displayExecutiveDetails();
   }
}
```

OUTPUT:



```
D:\SRIJA.java>javac Main3.java

D:\SRIJA.java>java Main3
Employee Name: John, Salary: 120000.0
Manager of Marketing Department.
Executive with company car: Tesla

D:\SRIJA.java>
```

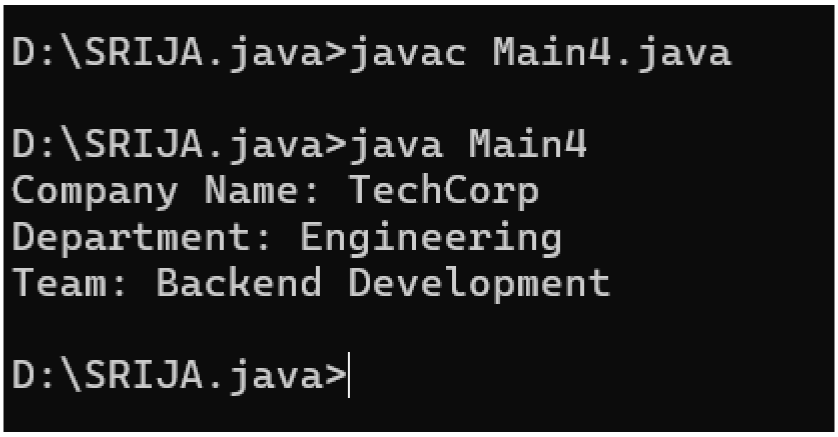
5.b)Company, Department, and Team

CODE:

```
class Company {
   protected String name;

   public Company(String name) {
      this.name = name;
   }

   public void displayCompany() {
      System.out.println("Company Name: " + name);
   }
}

class Department extends Company {
   protected String departmentName;

   public Department(String name, String departmentName) {
      super(name);
```

```
        this.departmentName = departmentName;
    }

    public void displayDepartment() {
        System.out.println("Department: " + departmentName);
    }
}

class Team extends Department {
    private String teamName;

    public Team(String name, String departmentName, String teamName) {
        super(name, departmentName);
        this.teamName = teamName;
    }

    public void displayTeam() {
        System.out.println("Team: " + teamName);
    }
}

public class Main4 {
    public static void main(String[] args) {
        Team team = new Team("TechCorp", "Engineering", "Backend Develop-
ment");
        team.displayCompany();
        team.displayDepartment();
        team.displayTeam();
    }
}
```

OUTPUT:



```
D:\SRIJA.java>javac Main4.java

D:\SRIJA.java>java Main4
Company Name: TechCorp
Department: Engineering
Team: Backend Development

D:\SRIJA.java>
```

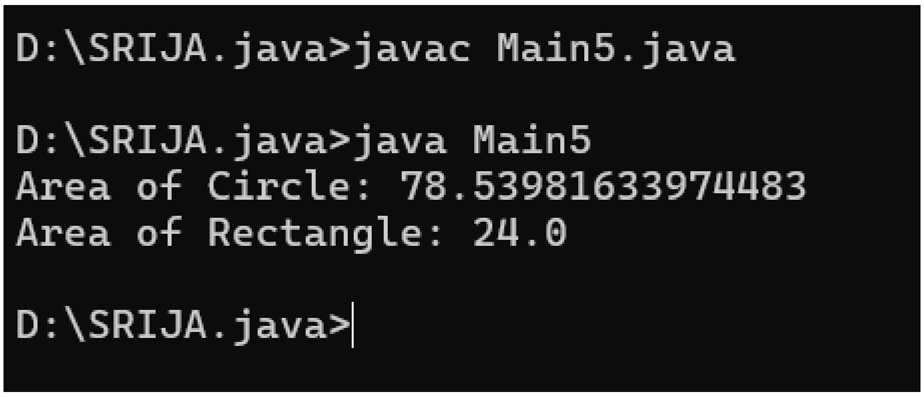
6Hierarchical Inheritance

6.a)Shape, Circle, and Rectangle

CODE

```
class Shape {
    public void area() {
        System.out.println("Calculating area...");
    }
}


class Circle extends Shape {
    private double radius;


    public Circle(double radius) {
        this.radius = radius;
    }
    public void area() {
        System.out.println("Area of Circle: " + Math.PI * radius * radius);
    }
}

class Rectangle extends Shape {
    private double width;
    private double height;

    public Rectangle(double width, double height) {
        this.width = width;
        this.height = height;
    }



    public void area() {
        System.out.println("Area of Rectangle: " + width * height);
    }
}

public class Main5 {
    public static void main(String[] args) {
        Circle circle = new Circle(5);
        Rectangle rectangle = new Rectangle(4, 6);
        circle.area();

        rectangle.area();

    }

}
```

OUTPUT:



```
D:\SRIJA.java>javac Main5.java

D:\SRIJA.java>java Main5
Area of Circle: 78.53981633974483
Area of Rectangle: 24.0

D:\SRIJA.java>
```

6.b)Employee, Developer, and Designer

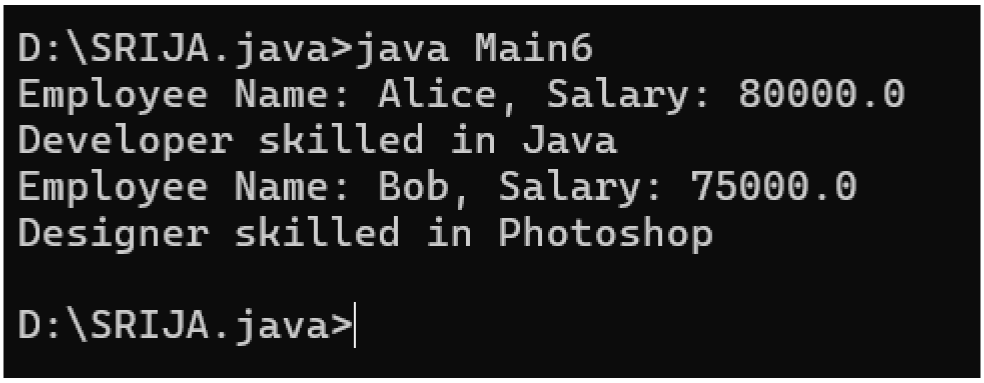
CODE

```
class Employee {

    protected String name;

    protected double salary;


    public Employee(String name, double salary) {

        this.name = name;

        this.salary = salary;

    }


    public void displayDetails() {

        System.out.println("Employee Name: " + name + ", Salary: " + salary);

    }

}
```

```
class Developer extends Employee {
    private String programmingLanguage;

    public Developer(String name, double salary, String programmingLanguage) {
        super(name, salary);
        this.programmingLanguage = programmingLanguage;
    }

    public void displayDeveloperDetails() {
        System.out.println("Developer skilled in " + programmingLanguage);
    }
}

class Designer extends Employee {
    private String designTool;

    public Designer(String name, double salary, String designTool) {
        super(name, salary);
        this.designTool = designTool;
    }

    public void displayDesignerDetails() {
        System.out.println("Designer skilled in " + designTool);
    }
}
```

```
public class Main6 {
    public static void main(String[] args) {
        Developer dev = new Developer("Alice", 80000, "Java");
        Designer des = new Designer("Bob", 75000, "Photoshop");

        dev.displayDetails();
        dev.displayDeveloperDetails();

        des.displayDetails();
        des.displayDesignerDetails();
    }
}
```

OUTPUT



```
D:\SRIJA.java>java Main6
Employee Name: Alice, Salary: 80000.0
Developer skilled in Java
Employee Name: Bob, Salary: 75000.0
Designer skilled in Photoshop

D:\SRIJA.java>
```

7.Hybrid Inheritance

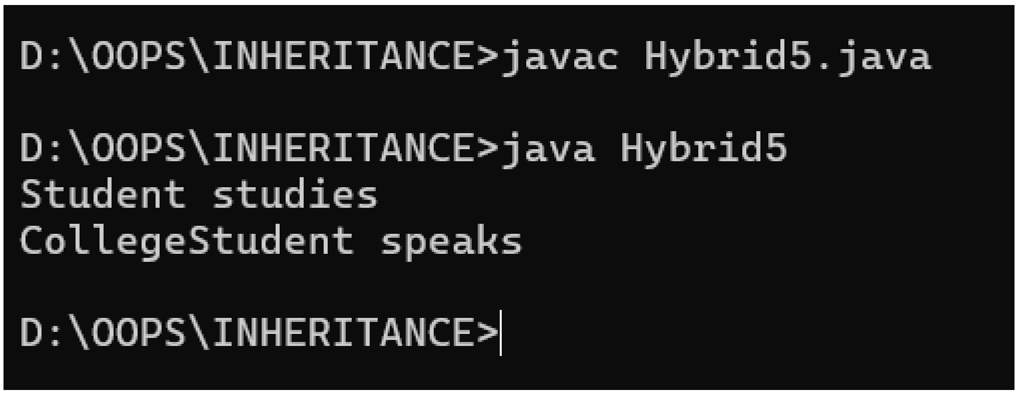
7.a)Student details

CODE

```
interface Person {
    void speak();
}
```

```
class Student {
    void study() {
        System.out.println("Student studies");
    }
}

class CollegeStudent extends Student implements Person {
    public void speak() {
        System.out.println("CollegeStudent speaks");
    }
}

public class Hybrid5 {
    public static void main(String[] args) {
        CollegeStudent cs = new CollegeStudent();
        cs.study();
        cs.speak();
    }
}
```

OUTPUT



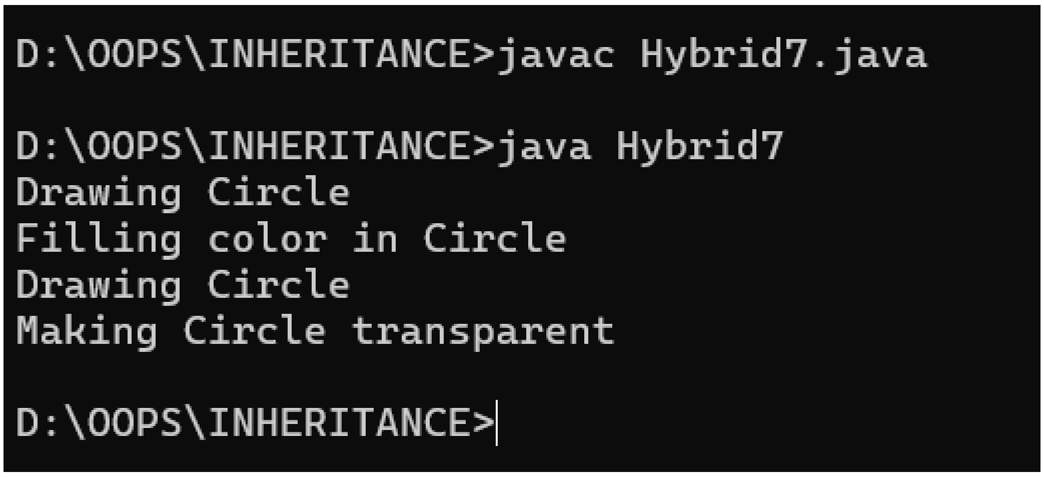
```
D:\OOPS\INHERITANCE>javac Hybrid5.java

D:\OOPS\INHERITANCE>java Hybrid5
Student studies
CollegeStudent speaks

D:\OOPS\INHERITANCE>
```

7.b)Shapes

CODE:

```
interface Shape {
   void draw();
}

class Circle implements Shape {
   public void draw() {
      System.out.println("Drawing Circle");
   }
}

class ColoredCircle extends Circle {
   void fillColor() {
      System.out.println("Filling color in Circle");
   }
}

class TransparentCircle extends Circle {
   void transparency() {
      System.out.println("Making Circle transparent");
   }
}

public class Hybrid7 {
   public static void main(String[] args) {
      ColoredCircle c = new ColoredCircle();
      c.draw();
      c.fillColor();

      TransparentCircle t = new TransparentCircle();
      t.draw();
      t.transparency();
   }
}
```

OUTPUT:



```
D:\OOPS\INHERITANCE>javac Hybrid7.java

D:\OOPS\INHERITANCE>java Hybrid7
Drawing Circle
Filling color in Circle
Drawing Circle
Making Circle transparent

D:\OOPS\INHERITANCE>
```

POLYMORPHISM

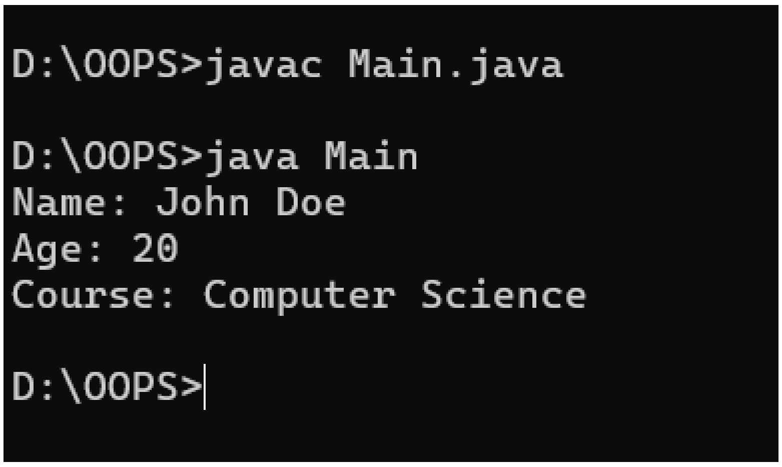
8**CONSTRUCTOR PROGRAMS**

8.a)Java Program to Demonstrate a Constructor in the Student Class

CODE:

```
 class Student {
String name;
int age;
String course;
Student() {
        name = "John Doe";

        age = 20;

        course = "Computer Science";

    }


    void displayInfo() {

        System.out.println("Name: " + name);

        System.out.println("Age: " + age);

        System.out.println("Course: " + course);

    }

}


public class Main {

    public static void main(String[] args) {

        Student s1 = new Student();

        s1.displayInfo();

    }

}
```

OUTPUT:



```
D:\OOPS>javac Main.java

D:\OOPS>java Main
Name: John Doe
Age: 20
Course: Computer Science

D:\OOPS>
```

9 CONSTRUCTOR OVERLOADING PROGRAMS

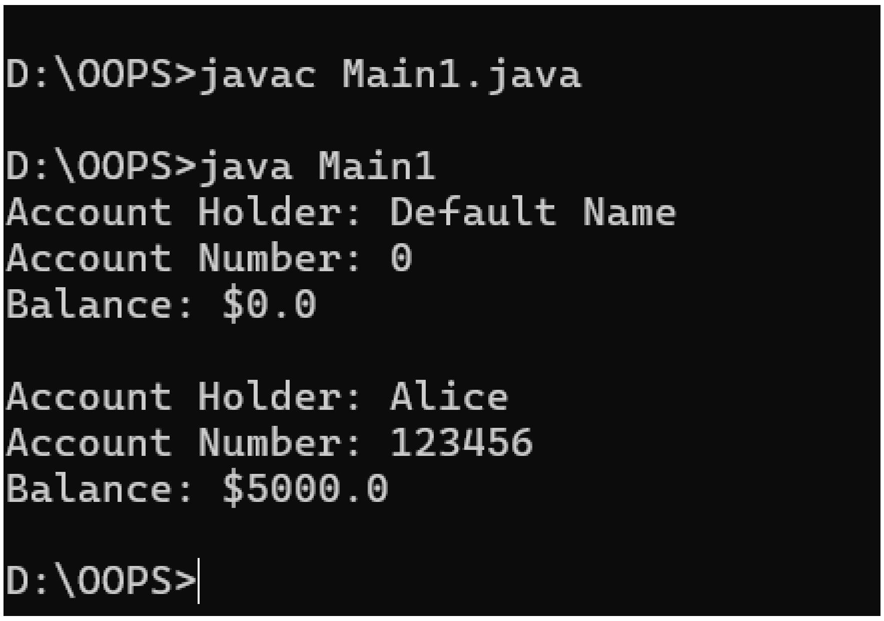
9.a) Constructor Overloading in a **BankAccount** Class

CODE:

```
class BankAccount {
    String accountHolder;
    int accountNumber;
    double balance;
    BankAccount() {
        accountHolder = "Default Name";
        accountNumber = 000000;
        balance = 0.0;
     }
    BankAccount(String name, int accNum, double bal) {
        accountHolder = name;
        accountNumber = accNum;
        balance = bal;
    }

    void displayAccount() {
        System.out.println("Account Holder: " + accountHolder);
        System.out.println("Account Number: " + accountNumber);
        System.out.println("Balance: $" + balance);
    }
}
```

```
public class Main {
    public static void main(String[] args) {
        BankAccount acc1 = new BankAccount(); // Default Constructor
        BankAccount acc2 = new BankAccount("Alice", 123456, 5000.0);

        acc1.displayAccount();
        System.out.println();
        acc2.displayAccount();
    }
}
```

OUTPUT:



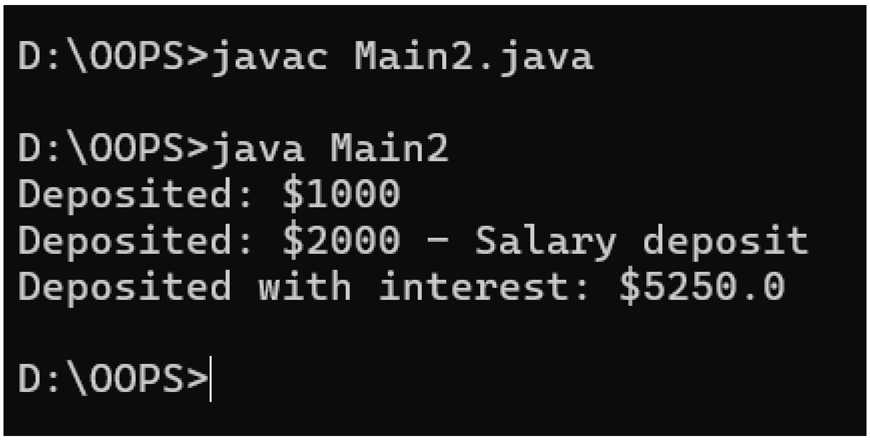
```
D:\OOPS>javac Main1.java

D:\OOPS>java Main1
Account Holder: Default Name
Account Number: 0
Balance: $0.0

Account Holder: Alice
Account Number: 123456
Balance: $5000.0

D:\OOPS>
```

10.**METHOD OVERLOADING PROGRAMS**

10.a) Method Overloading in a Bank System java

CODE:

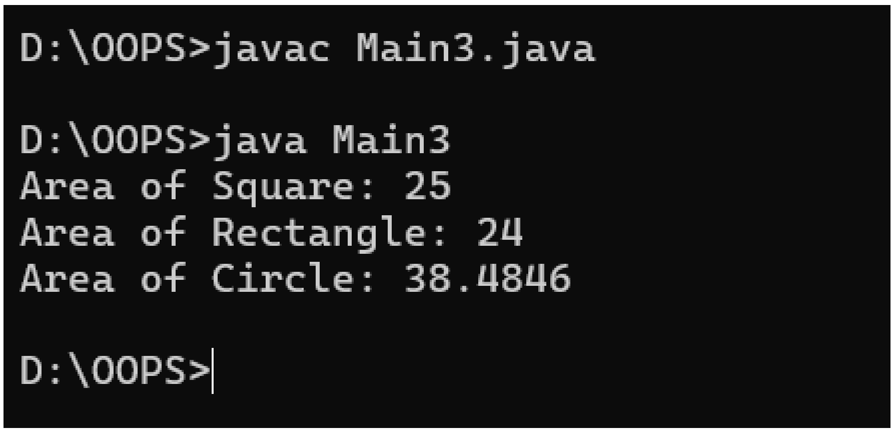
```
class Bank {
    void deposit(int amount) {
        System.out.println("Deposited: $" + amount);
    }
    void deposit(int amount, String message) {
        System.out.println("Deposited: $" + amount + " - " + message);
    }
    void deposit(double amount, double interestRate) {
        double total = amount + (amount * interestRate / 100);
        System.out.println("Deposited with interest: $" + total);
    }
}

public class Main2 {
    public static void main(String[] args) {
        Bank bank = new Bank();

        bank.deposit(1000);
        bank.deposit(2000, "Salary deposit");
        bank.deposit(5000, 5.0);
    }
}
```
OUTPUT:



```
D:\OOPS>javac Main2.java

D:\OOPS>java Main2
Deposited: $1000
Deposited: $2000 - Salary deposit
Deposited with interest: $5250.0

D:\OOPS>
```
10.b) Method Overloading in a Shape Class

CODE:
```
class Shape {
     int area(int side) {
        return side * side;
    }
    int area(int length, int breadth) {
        return length * breadth;
    }
    double area(double radius) {
        return 3.1416 * radius * radius;
    }
}

public class Main3 {
    public static void main(String[] args) {
        Shape shape = new Shape();

        System.out.println("Area of Square: " + shape.area(5));
        System.out.println("Area of Rectangle: " + shape.area(4, 6));
        System.out.println("Area of Circle: " + shape.area(3.5));
    }
}
```

OUTPUT:



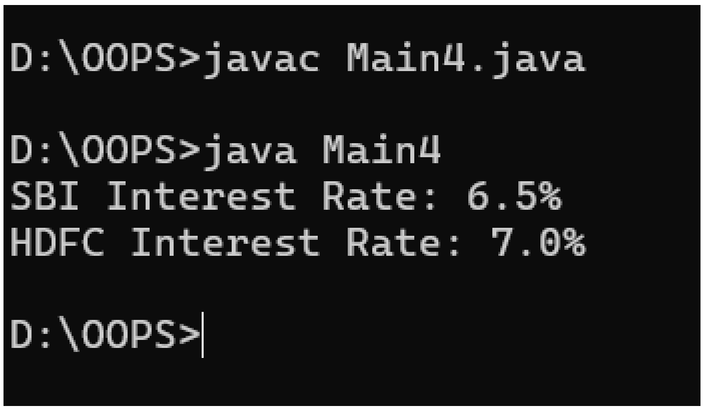
```
D:\OOPS>javac Main3.java

D:\OOPS>java Main3
Area of Square: 25
Area of Rectangle: 24
Area of Circle: 38.4846

D:\OOPS>
```

11.METHOD OVERRIDING PROGRAMS

11.a)Bank Interest Calculation

CODE:

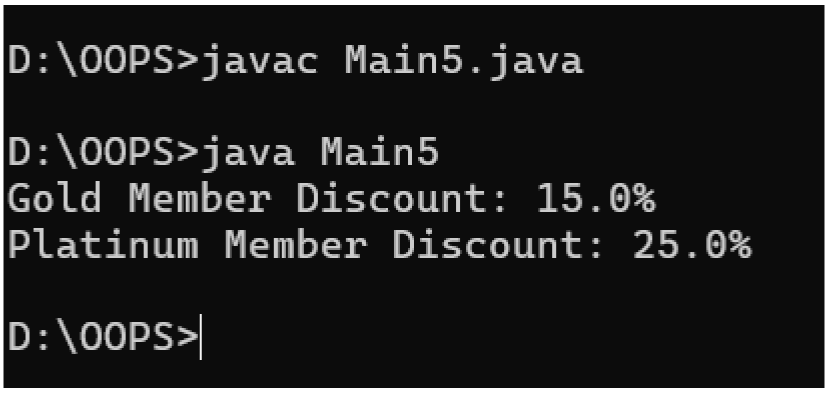
```
class Bank {
    double getInterestRate() {
        return 5.0; // Default interest rate
    }
}

class SBI extends Bank {
    double getInterestRate() {
        return 6.5; // SBI offers 6.5% interest
    }
}

class HDFC extends Bank {
    double getInterestRate() {
        return 7.0; // HDFC offers 7.0% interest
    }
}

public class Main4 {
    public static void main(String[] args) {
        Bank sbi = new SBI();
        Bank hdfc = new HDFC();

        System.out.println("SBI Interest Rate: " + sbi.getInterestRate() + "%");
        System.out.println("HDFC Interest Rate: " + hdfc.getInterestRate() +
"%");
    }
}
```
OUTPUT:



```
D:\OOPS>javac Main4.java

D:\OOPS>java Main4
SBI Interest Rate: 6.5%
HDFC Interest Rate: 7.0%

D:\OOPS>
```
11.b)Online Shopping Discount System

CODE:
```
class Discount {
    double getDiscount() {
        return 5; // Default discount
    }
}

class GoldMember extends Discount {
        double getDiscount() {
        return 15; // Gold members get higher discounts
    }
}

class PlatinumMember extends Discount {
       double getDiscount() {
        return 25; // Platinum members get even more discount
    }
}

public class Main5 {
    public static void main(String[] args) {
        Discount customer1 = new GoldMember();
        Discount customer2 = new PlatinumMember();

        System.out.println("Gold Member Discount: " +
customer1.getDiscount() + "%");
        System.out.println("Platinum Member Discount: " +
customer2.getDiscount() + "%");
    }
}
```

OUTPUT:



```
D:\OOPS>javac Main5.java

D:\OOPS>java Main5
Gold Member Discount: 15.0%
Platinum Member Discount: 25.0%

D:\OOPS>
```

ABSTRACTION

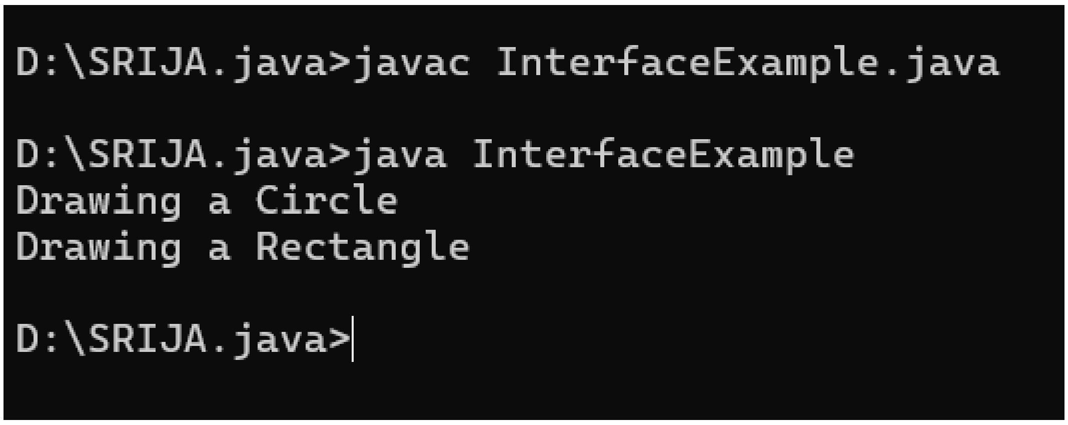
12.Abstraction using Interface Classes:

12.a)Shape Interface

CODE

```
interface Shape {

    void draw();

}


class Circle implements Shape {

    public void draw() {
        System.out.println("Drawing a Circle");
    }
}

class Rectangle implements Shape {
    public void draw() {
        System.out.println("Drawing a Rectangle");
    }
}

public class InterfaceExample{
    public static void main(String[] args) {
        Shape s1 = new Circle();
        Shape s2 = new Rectangle();
        s1.draw();
        s2.draw();
    }
}
```

OUTPUT:



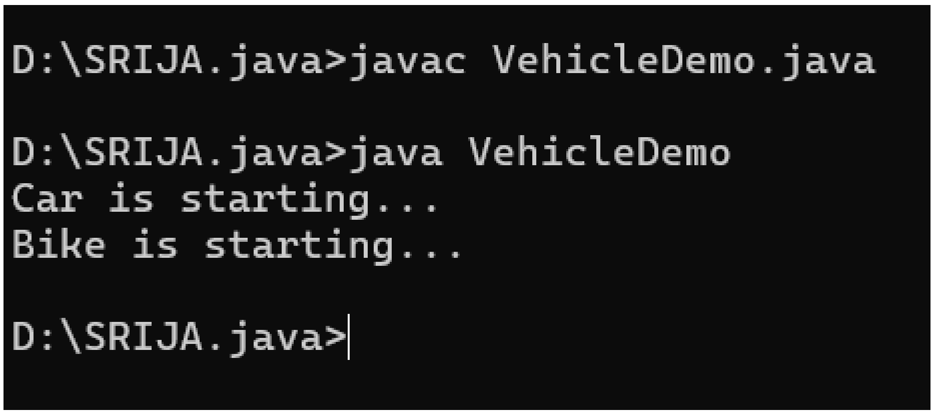
```
D:\SRIJA.java>javac InterfaceExample.java

D:\SRIJA.java>java InterfaceExample
Drawing a Circle
Drawing a Rectangle

D:\SRIJA.java>
```

12.b) Vehicle Interface

CODE:

```
interface Vehicle {
    void start();
}

class Car implements Vehicle {
    public void start() {
        System.out.println("Car is starting...");
    }
}

class Bike implements Vehicle {
    public void start() {
        System.out.println("Bike is starting...");
    }
}

public class VehicleDemo{
    public static void main(String[] args) {
        Vehicle car = new Car();
        Vehicle bike = new Bike();
        car.start();
        bike.start();
    }
}
```

OUTPUT:



```
D:\SRIJA.java>javac VehicleDemo.java

D:\SRIJA.java>java VehicleDemo
Car is starting...
Bike is starting...

D:\SRIJA.java>
```

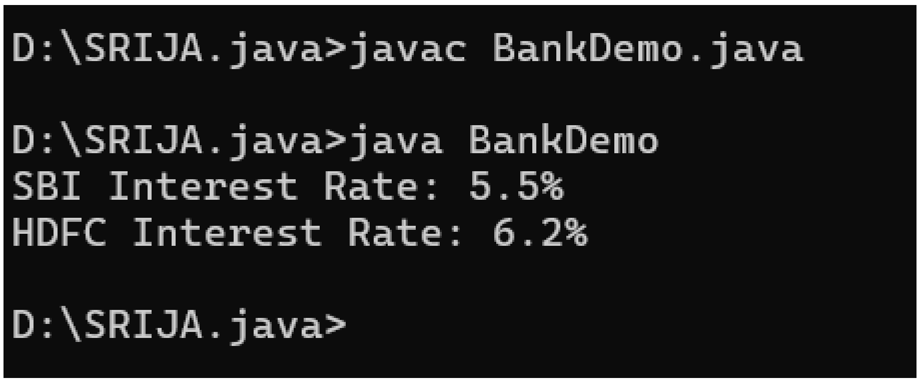
12.c) Bank Interface

CODE

```
interface Bank {
   double getInterestRate();
}


class SBI implements Bank {
   public double getInterestRate() {
      return 5.5;
   }
}


class HDFC implements Bank {
   public double getInterestRate() {
      return 6.2;
   }
}
public class BankDemo {
   public static void main(String[] args) {
      Bank b1 = new SBI();
```

```
        Bank b2 = new HDFC();
        System.out.println("SBI Interest Rate: " + b1.getInterestRate() + "%");
        System.out.println("HDFC Interest Rate: " + b2.getInterestRate() + "%");
    }
}
```

OUTPUT:



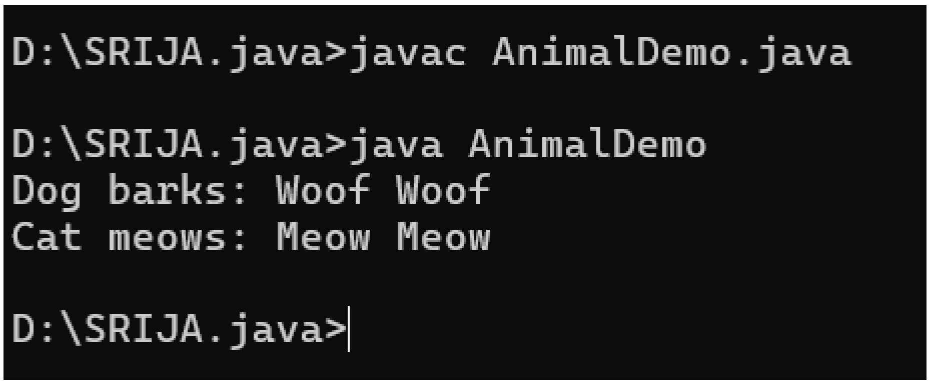
```
D:\SRIJA.java>javac BankDemo.java

D:\SRIJA.java>java BankDemo
SBI Interest Rate: 5.5%
HDFC Interest Rate: 6.2%

D:\SRIJA.java>
```

12.d) Animal Interface

CODE

```
interface Animal {
    void makeSound();
}

class Dog implements Animal {
    public void makeSound() {
        System.out.println("Dog barks: Woof Woof");
    }
}

class Cat implements Animal {
    public void makeSound() {
        System.out.println("Cat meows: Meow Meow");
    }
}

public class AnimalDemo {
    public static void main(String[] args) {
        Animal dog = new Dog();
        Animal cat = new Cat();
        dog.makeSound();
        cat.makeSound();
    }
}
```

OUTPUT:



```
D:\SRIJA.java>javac AnimalDemo.java

D:\SRIJA.java>java AnimalDemo
Dog barks: Woof Woof
Cat meows: Meow Meow

D:\SRIJA.java>
```

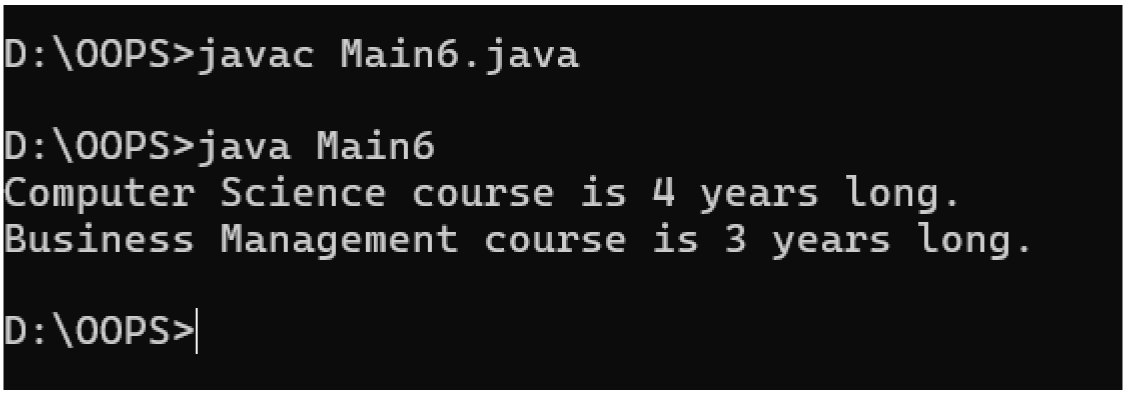
13.Abstraction using Abstract Classes

13.a) University Courses

CODE:

```
abstract class Course {
    abstract void courseDuration();
}

class ComputerScience extends Course {
       void courseDuration() {
        System.out.println("Computer Science course is 4 years long.");
    }
}

class BusinessManagement extends Course {
       void courseDuration() {
        System.out.println("Business Management course is 3 years long.");
    }
}

public class Main6 {
    public static void main(String[] args) {
        Course c1 = new ComputerScience();
        Course c2 = new BusinessManagement();
        c1.courseDuration();
        c2.courseDuration();
    }
}
```

OUTPUT:



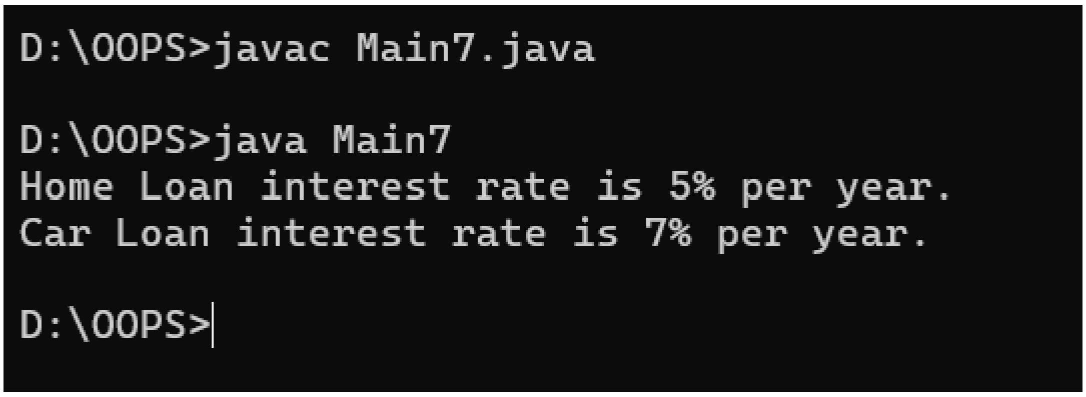
```
D:\OOPS>javac Main6.java

D:\OOPS>java Main6
Computer Science course is 4 years long.
Business Management course is 3 years long.

D:\OOPS>
```

13.b)Loan Interest Calculation

CODE:

```
abstract class Loan {
   abstract void calculateInterest();
}


class HomeLoan extends Loan {
     void calculateInterest() {
       System.out.println("Home Loan interest rate is 5% per year.");
   }
}


class CarLoan extends Loan {
       void calculateInterest() {
       System.out.println("Car Loan interest rate is 7% per year.");
   }
}


public class Main7 {
   public static void main(String[] args) {
       Loan l1 = new HomeLoan();
       Loan l2 = new CarLoan();
       l1.calculateInterest();
       l2.calculateInterest();
   }
}
```

OUTPUT:



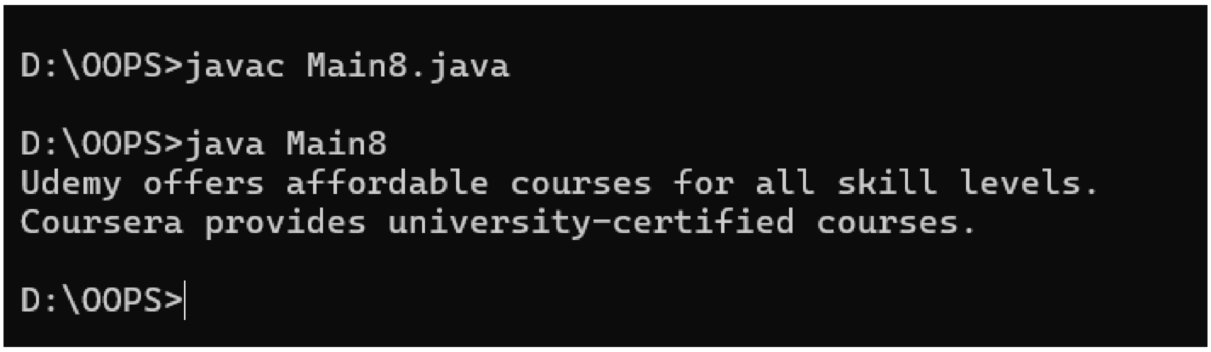
```
D:\OOPS>javac Main7.java

D:\OOPS>java Main7
Home Loan interest rate is 5% per year.
Car Loan interest rate is 7% per year.

D:\OOPS>
```

13.c)Online Learning Platforms

CODE:

```
abstract class OnlineCourse {
   abstract void platformDetails();
}


class Udemy extends OnlineCourse {
      void platformDetails() {
      System.out.println("Udemy offers affordable courses for all skill levels.");
   }
}


class Coursera extends OnlineCourse {
      void platformDetails() {
      System.out.println("Coursera provides university-certified courses.");
   }
}


public class Main8 {
   public static void main(String[] args) {
      OnlineCourse o1 = new Udemy();
      OnlineCourse o2 = new Coursera();
      o1.platformDetails();
      o2.platformDetails();
   }
}
```

OUTPUT:



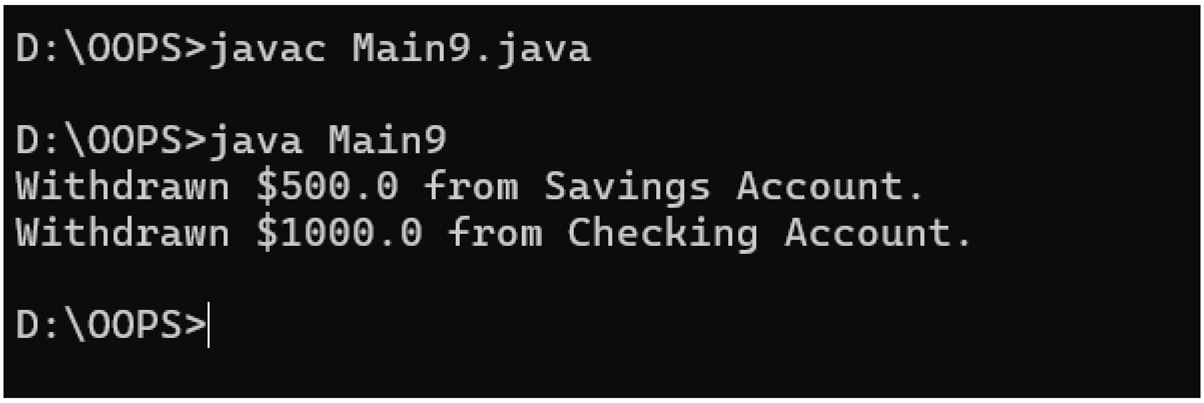
```
D:\OOPS>javac Main8.java

D:\OOPS>java Main8
Udemy offers affordable courses for all skill levels.
Coursera provides university-certified courses.

D:\OOPS>
```

13.d) Bank Account Transactions

CODE:

```
abstract class BankAccount {
    abstract void withdraw(double amount);
}


class SavingsAccount extends BankAccount {
    void withdraw(double amount) {
        System.out.println("Withdrawn $" + amount + " from Savings Account.");
    }
}


class CheckingAccount extends BankAccount {
    void withdraw(double amount) {
        System.out.println("Withdrawn $" + amount + " from Checking Account.");
    }
}


public class Main9 {
    public static void main(String[] args) {
        BankAccount acc1 = new SavingsAccount();
        BankAccount acc2 = new CheckingAccount();
        acc1.withdraw(500);
        acc2.withdraw(1000);
    }
}
```

OUTPUT:



```
D:\OOPS>javac Main9.java

D:\OOPS>java Main9
Withdrawn $500.0 from Savings Account.
Withdrawn $1000.0 from Checking Account.

D:\OOPS>
```

ENCAPSULATION

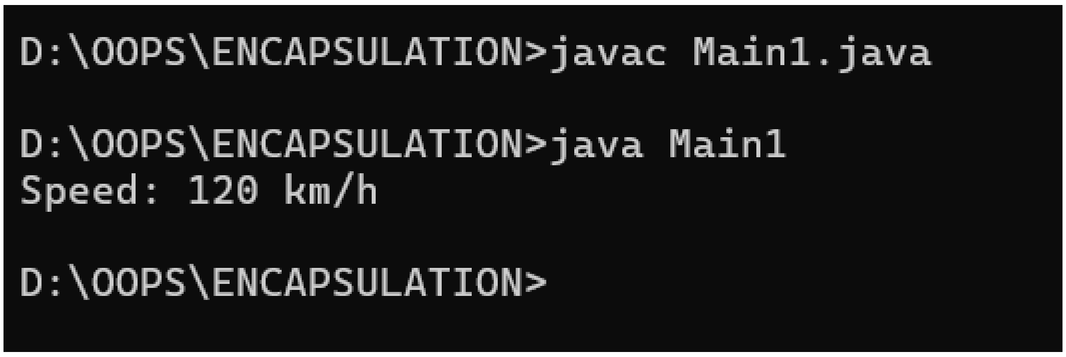
14ENCAPSULATION PROBLEMS

14.a)Car Speed Control

CODE:

```
class Car {
    private int speed;


    public void setSpeed(int speed) {
        if (speed >= 0 && speed <= 200) this.speed = speed;
        else System.out.println("Invalid speed!");
    }


    public int getSpeed() { return speed; }
}


public class Main1 {
    public static void main(String[] args) {
        Car myCar = new Car();
        myCar.setSpeed(120);
        System.out.println("Speed: " + myCar.getSpeed() + " km/h");
    }
}
```

OUTPUT

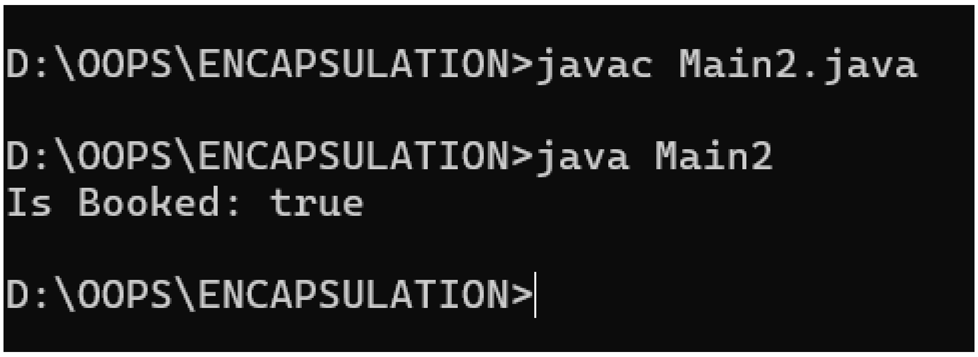


```
D:\OOPS\ENCAPSULATION>javac Main1.java

D:\OOPS\ENCAPSULATION>java Main1
Speed: 120 km/h

D:\OOPS\ENCAPSULATION>
```

14.b)Movie Ticket Booking
CODE:

```
class Ticket {
    private boolean isBooked;

    public void bookTicket() {
        if (!isBooked) isBooked = true;
        else System.out.println("Already booked!");
    }

    public boolean isBooked() { return isBooked; }
}

public class Main2 {
    public static void main(String[] args) {
        Ticket t = new Ticket();
        t.bookTicket();
        System.out.println("Is Booked: " + t.isBooked());
    }
}
```

OUTPUT:



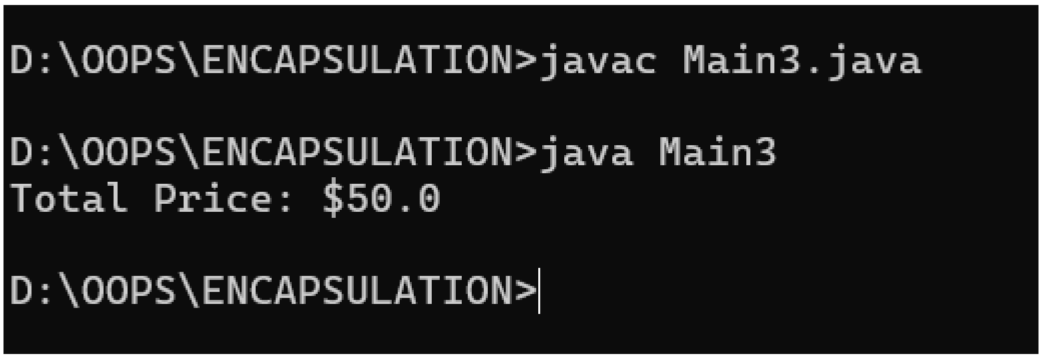
```
D:\OOPS\ENCAPSULATION>javac Main2.java

D:\OOPS\ENCAPSULATION>java Main2
Is Booked: true

D:\OOPS\ENCAPSULATION>
```

14.c)Online Shopping Cart

CODE:

```
class Cart {
    private double totalPrice;

    public void addItem(double price) {
        if (price > 0) totalPrice += price;
    }

    public double getTotalPrice() { return totalPrice; }
}

public class Main {
    public static void main(String[] args) {
        Cart c = new Cart();
        c.addItem(50);
        System.out.println("Total Price: $" + c.getTotalPrice());
    }
}
```

OUTPUT:



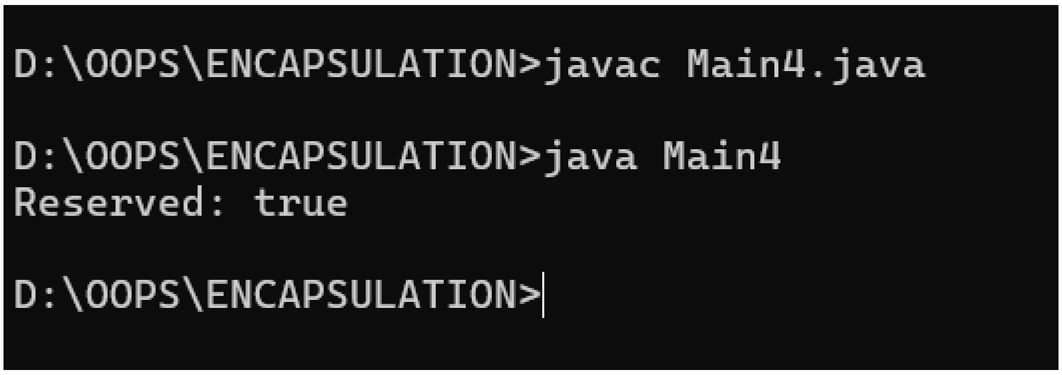
```
D:\OOPS\ENCAPSULATION>javac Main3.java

D:\OOPS\ENCAPSULATION>java Main3
Total Price: $50.0

D:\OOPS\ENCAPSULATION>
```

14.d)Train Ticket Reservation

CODE:

```
class TrainTicket {
    private boolean isReserved;

    public void reserveTicket() {
        if (!isReserved) isReserved = true;
        else System.out.println("Already reserved!");
    }

    public boolean isReserved() { return isReserved; }
}

public class Main4 {
    public static void main(String[] args) {
        TrainTicket t = new TrainTicket();
        t.reserveTicket();
        System.out.println("Reserved: " + t.isReserved());
    }
}
```

OUTPUT:



```
D:\OOPS\ENCAPSULATION>javac Main4.java

D:\OOPS\ENCAPSULATION>java Main4
Reserved: true

D:\OOPS\ENCAPSULATION>
```

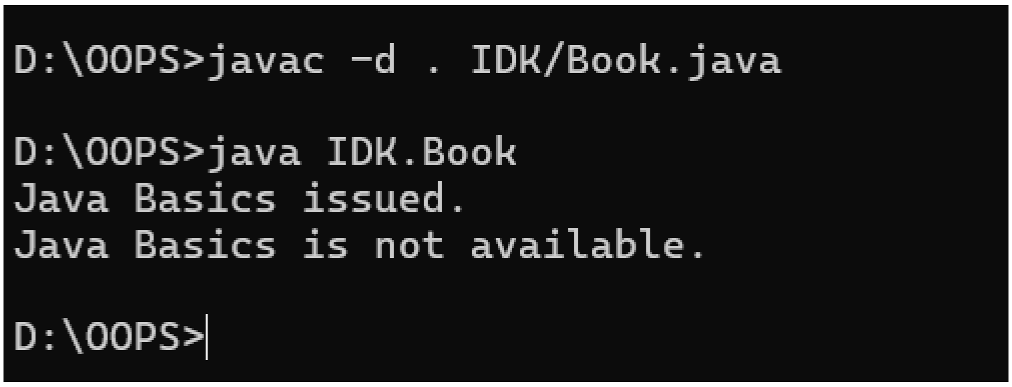
15.PACKAGES PROGRAMS

15.a) Library Book Program

CODE:

```
package IDK;

public class Book {
    private String title;
    private String author;
    private boolean isAvailable;

    public Book(String title, String author) {
        this.title = title;
        this.author = author;
        this.isAvailable = true;
    }

    public void issue() {
        if (isAvailable) {
            isAvailable = false;
            System.out.println(title + " issued.");
        } else {
            System.out.println(title + " is not available.");
        }
    }

    public static void main(String[] args) {
        Book b = new Book("Java Basics", "James");
        b.issue();
        b.issue();
    }
}
```

OUTPUT:



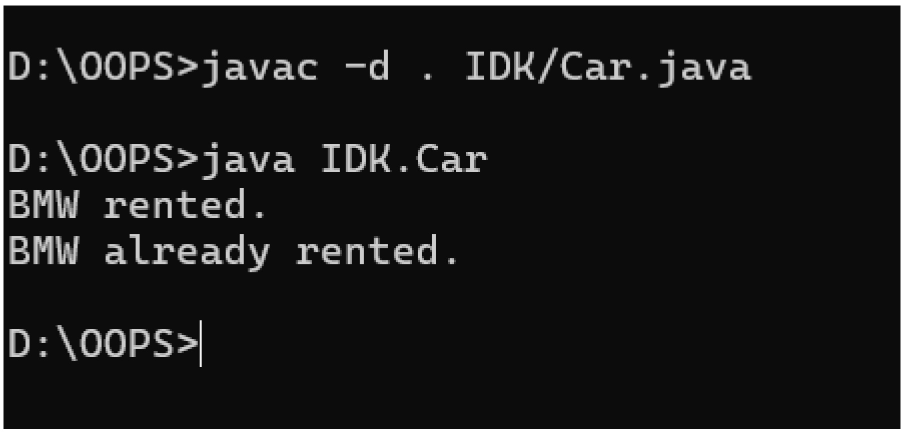
```
D:\OOPS>javac -d . IDK/Book.java

D:\OOPS>java IDK.Book
Java Basics issued.
Java Basics is not available.

D:\OOPS>
```

15.b) Car Rental Program

CODE:

```
package IDK;


public class Car {
    private String model;
    private double rate;
    private boolean rented;


    public Car(String model, double rate) {
        this.model = model;
        this.rate = rate;
        this.rented = false;
    }
    public void rent() {
        if (!rented) {
            rented = true;
            System.out.println(model + " rented.");
        } else {
            System.out.println(model + " already rented.");
        }
    }

    public static void main(String[] args) {
        Car c = new Car("BMW", 100);
        c.rent();
        c.rent();
    }
}
```

OUTPUT:



```
D:\OOPS>javac -d . IDK/Car.java

D:\OOPS>java IDK.Car
BMW rented.
BMW already rented.

D:\OOPS>
```

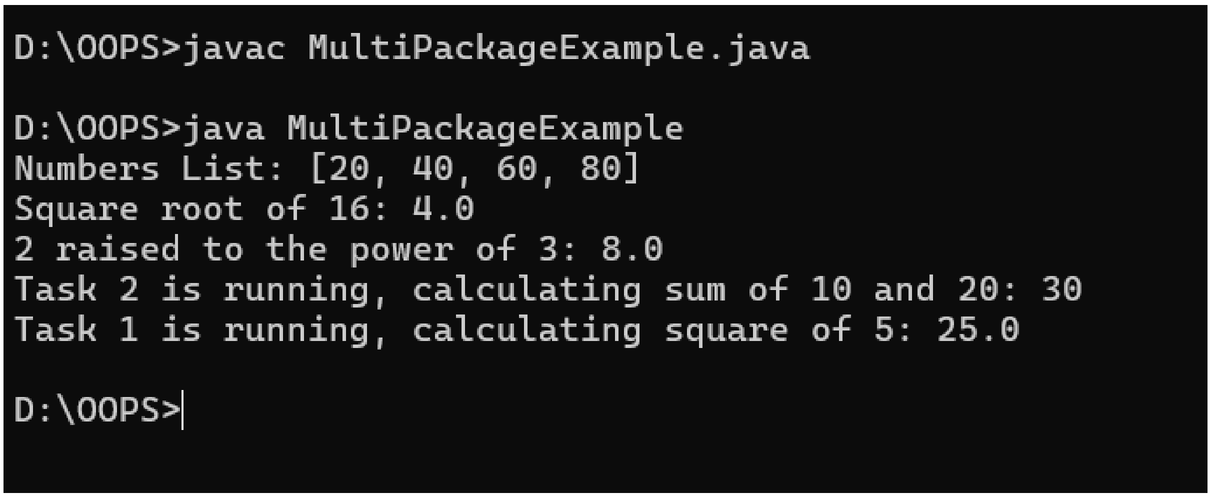
15.c) **Simple File Reader**

CODE:

```
import java.util.*;
import java.lang.Math;
import java.util.concurrent.*;

public class MultiPackageExample {
    public static void main(String[] args) throws InterruptedException {
        List<Integer> numbers = new ArrayList<>();
        numbers.add(10);
        numbers.add(20);
        numbers.add(30);
        numbers.add(40);
        System.out.println("Numbers List: " + numbers);

        double sqrtResult = Math.sqrt(16);
        double powResult = Math.pow(2, 3);
        System.out.println("Square root of 16: " + sqrtResult);
        System.out.println("2 raised to the power of 3: " + powResult);

        ExecutorService executor = Executors.newFixedThreadPool(2);

        Runnable task1 = () -> {
            System.out.println("Task 1 is running, calculating square of 5: " +
Math.pow(5, 2));
        };

        Runnable task2 = () -> {
            System.out.println("Task 2 is running, calculating sum of 10 and 20: " +
(10 + 20));
```

```
        };


        executor.submit(task1);

        executor.submit(task2);


        executor.shutdown();

        executor.awaitTermination(1, TimeUnit.SECONDS);

    }

}
```

OUTPUT:



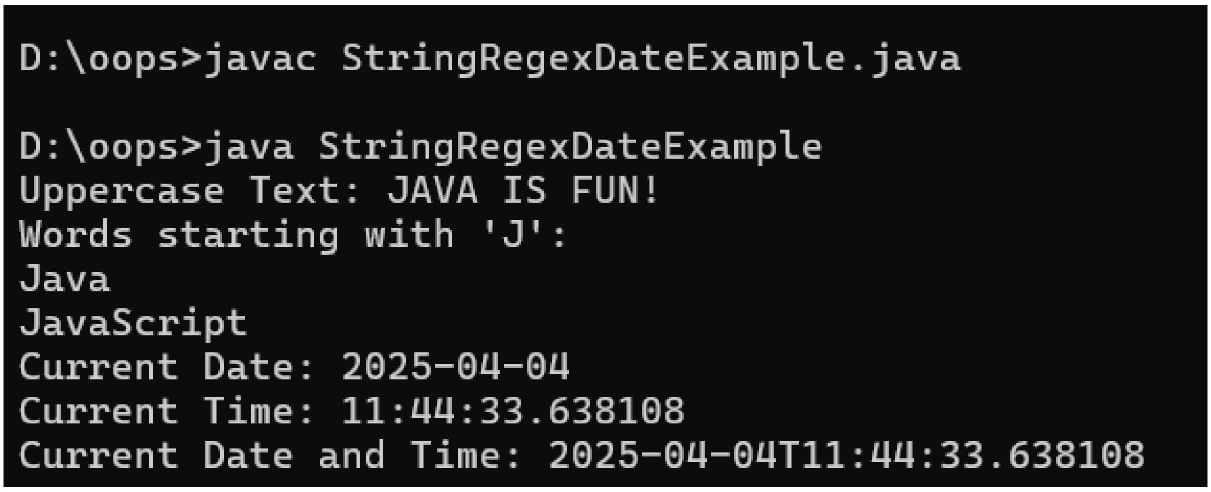
```
D:\OOPS>javac MultiPackageExample.java

D:\OOPS>java MultiPackageExample
Numbers List: [20, 40, 60, 80]
Square root of 16: 4.0
2 raised to the power of 3: 8.0
Task 2 is running, calculating sum of 10 and 20: 30
Task 1 is running, calculating square of 5: 25.0

D:\OOPS>
```

15.D)

CODE:

```
import java.util.regex.*;

import java.lang.String;

import java.time.*;


public class StringRegexDateExample {

    public static void main(String[] args) {
        String text = "Java is fun!";
        String upperCaseText = text.toUpperCase();
        System.out.println("Uppercase Text: " + upperCaseText);

        Pattern pattern = Pattern.compile("\\bJ\\w*");
        Matcher matcher = pattern.matcher("Java is fun! JavaScript is also fun.");
        System.out.println("Words starting with 'J':");
        while (matcher.find()) {
            System.out.println(matcher.group());
        }

        LocalDate currentDate = LocalDate.now();
        LocalTime currentTime = LocalTime.now();
        LocalDateTime currentDateTime = LocalDateTime.now();

        System.out.println("Current Date: " + currentDate);
        System.out.println("Current Time: " + currentTime);
        System.out.println("Current Date and Time: " + currentDateTime);
    }
}
```

OUTPUT:



```
D:\oops>javac StringRegexDateExample.java

D:\oops>java StringRegexDateExample
Uppercase Text: JAVA IS FUN!
Words starting with 'J':
Java
JavaScript
Current Date: 2025-04-04
Current Time: 11:44:33.638108
Current Date and Time: 2025-04-04T11:44:33.638108
```

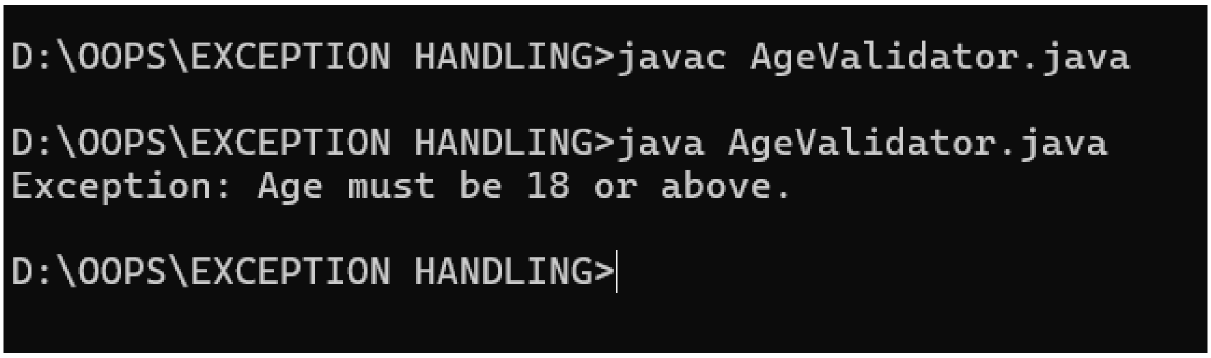
16)EXCEPTION HANDLING

16.a) Age Validation

CODE:

```
class InvalidAgeException extends Exception {
    public InvalidAgeException(String message) {
        super(message);
    }
}

public class AgeValidator {
    public static void validateAge(int age) throws InvalidAgeException {
        if (age < 18) {
            throw new InvalidAgeException("Age must be 18 or above.");
        } else {
            System.out.println("Age is valid for voting.");
        }
    }

    public static void main(String[] args) {
        try {
            validateAge(16);
        } catch (InvalidAgeException e) {
            System.out.println("Exception: " + e.getMessage());
        }
    }
}
```

OUTPUT:



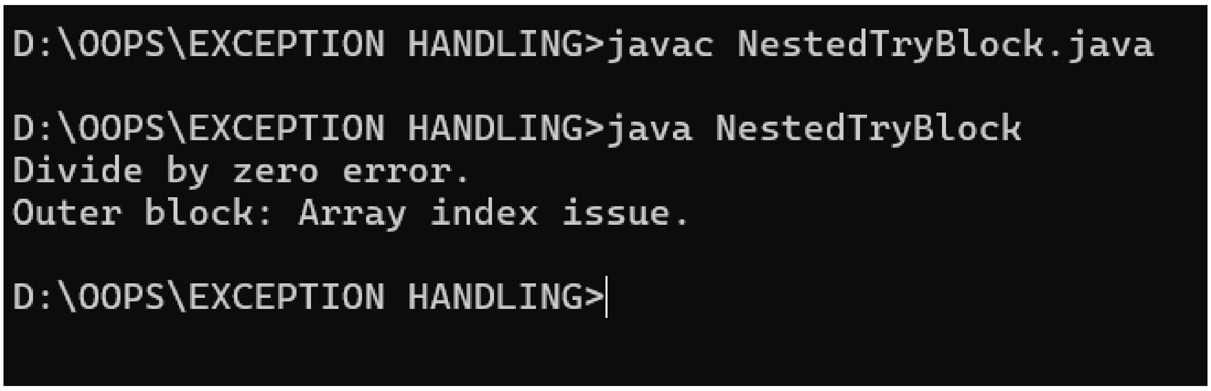
```
D:\OOPS\EXCEPTION HANDLING>javac AgeValidator.java

D:\OOPS\EXCEPTION HANDLING>java AgeValidator.java
Exception: Age must be 18 or above.

D:\OOPS\EXCEPTION HANDLING>
```

16.b) Nested Try Blocks

CODE:

```
public class NestedTryBlock {
    public static void main(String[] args) {
        try {
            int[] nums = {1, 2, 3};
            try {
                int result = nums[2] / 0;
                System.out.println("Result: " + result);
            } catch (ArithmeticException e) {
                System.out.println("Divide by zero error.");
            }
            System.out.println(nums[5]);
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Outer block: Array index issue.");
        }
    }
}
```

OUTPUT:



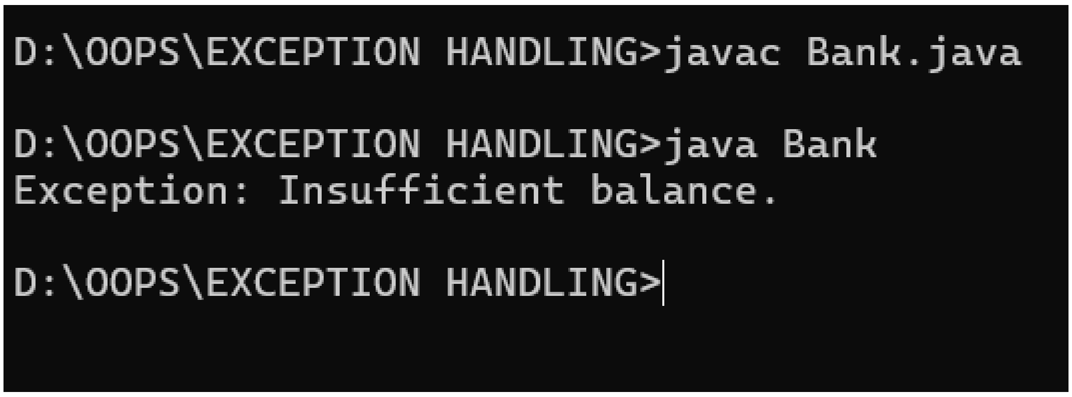
```
D:\OOPS\EXCEPTION HANDLING>javac NestedTryBlock.java

D:\OOPS\EXCEPTION HANDLING>java NestedTryBlock
Divide by zero error.
Outer block: Array index issue.

D:\OOPS\EXCEPTION HANDLING>
```

16.C) Bank Withdrawal

CODE:

```
class InsufficientFundsException extends Exception {
    public InsufficientFundsException(String message) {
        super(message);
    }
}

public class Bank {
    static void withdraw(double amount, double balance) throws Insuffi-
cientFundsException {
        if (amount > balance) {
            throw new InsufficientFundsException("Insufficient balance.");
        } else {
            System.out.println("Withdrawal successful. Remaining: " + (balance -
amount));
        }
    }

    public static void main(String[] args) {
        try {
            withdraw(1500, 1000);
        } catch (InsufficientFundsException e) {
            System.out.println("Exception: " + e.getMessage());
        }
    }
}
```

OUTPUT:



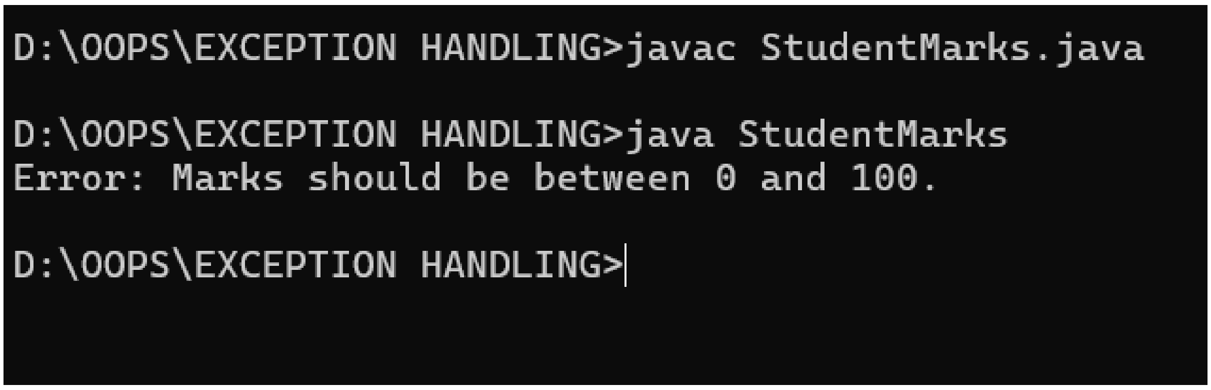
```
D:\OOPS\EXCEPTION HANDLING>javac Bank.java

D:\OOPS\EXCEPTION HANDLING>java Bank
Exception: Insufficient balance.

D:\OOPS\EXCEPTION HANDLING>
```

16.D) Student Marks Validation

CODE:

```
class InvalidMarkException extends Exception {
    public InvalidMarkException(String message) {
        super(message);
    }
}


public class StudentMarks {
    public static void checkMarks(int marks) throws InvalidMarkException {
        if (marks < 0 || marks > 100) {
            throw new InvalidMarkException("Marks should be between 0 and
100.");
        } else {
            System.out.println("Valid marks entered: " + marks);
        }
    }


    public static void main(String[] args) {
        try {
            checkMarks(110);
        } catch (InvalidMarkException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

OUTPUT:



```
D:\OOPS\EXCEPTION HANDLING>javac StudentMarks.java

D:\OOPS\EXCEPTION HANDLING>java StudentMarks
Error: Marks should be between 0 and 100.

D:\OOPS\EXCEPTION HANDLING>
```

17)FILE HANDLING PROGRAMS

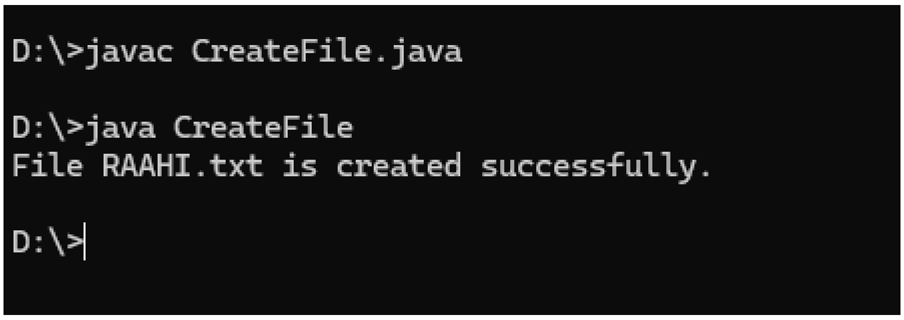
17.A)

CODE: import java.io.File;

```
import java.io.IOException;
 class CreateFile {
        public static void main(String args[]) {
        try {
            File f0 = new File("RAAHI.txt");
            if (f0.createNewFile()) {
                System.out.println("File " + f0.getName() + " is created suc-
cessfully.");
            } else {
                System.out.println("File is already exist in the directory.");
            }
          } catch (IOException exception) {
```

```
                    System.out.println("An unexpected error is occurred.");
                    exception.printStackTrace();
            }
      }
}
```

OUTPUT:



```
D:\>javac CreateFile.java

D:\>java CreateFile
File RAAHI.txt is created successfully.

D:\>
```

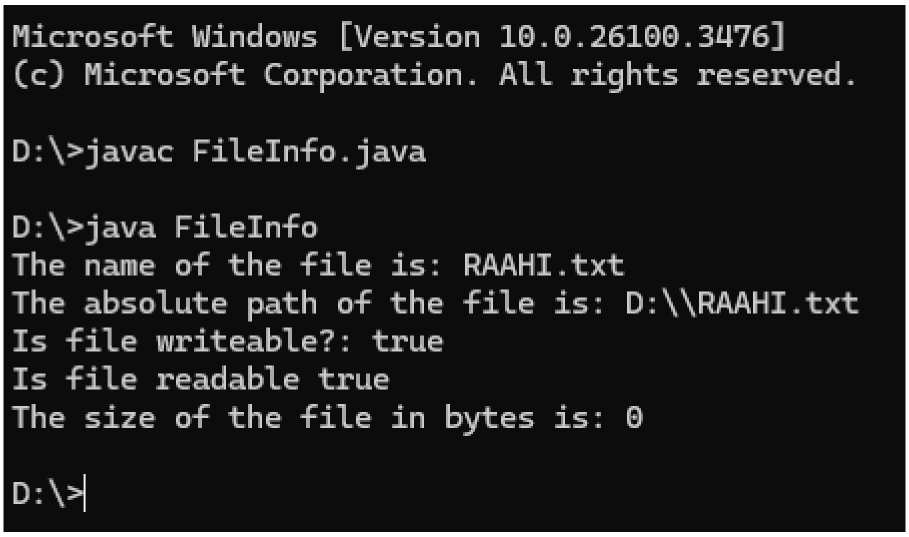
17.B)

CODE:

```
 import java.io.File;
class FileInfo {
   public static void main(String[] args) {
       File f0 = new File("RAAHI.txt");
       if (f0.exists()) {
           System.out.println("The name of the file is: " + f0.getName());

           System.out.println("The absolute path of the file is: " +
f0.getAbsolutePath());

           System.out.println("Is file writeable?: " + f0.canWrite());
```

```
            System.out.println("Is file readable " + f0.canRead());


            System.out.println("The size of the file in bytes is: " + f0.length());
        } else {
            System.out.println("The file does not exist." + " "+"please try again");
        }
    }
}
```

OUTPUT:



```
Microsoft Windows [Version 10.0.26100.3476]
(c) Microsoft Corporation. All rights reserved.

D:\>javac FileInfo.java

D:\>java FileInfo
The name of the file is: RAAHI.txt
The absolute path of the file is: D:\\RAAHI.txt
Is file writeable?: true
Is file readable true
The size of the file in bytes is: 0

D:\>
```

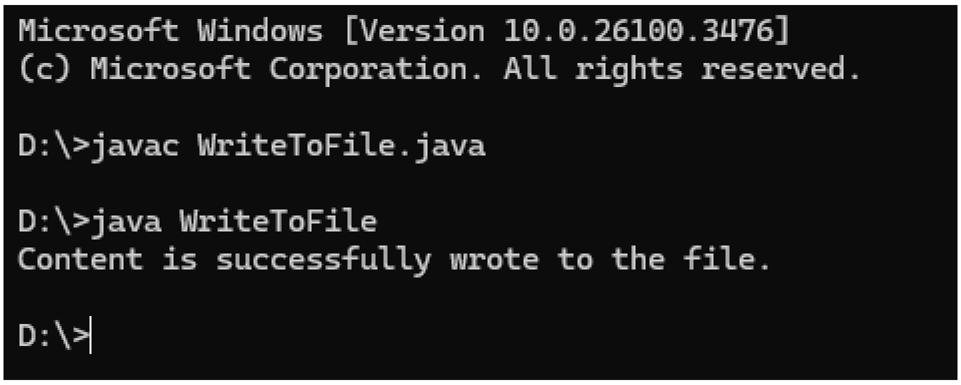
17.C)

CODE:

```
import java.io.FileWriter;
import java.io.IOException;


class WriteToFile {
    public static void main(String[] args) {
```

```
   try {
      FileWriter fwrite = new FileWriter("D:RAAHI.txt");


      fwrite.write("A named location used to store related information is re-
ferred to as a File.");
         fwrite.close();
      System.out.println("Content is successfully wrote to the file.");
   } catch (IOException e) {
      System.out.println(" error");
      e.printStackTrace();
      }
   }
}
```

OUTPUT:



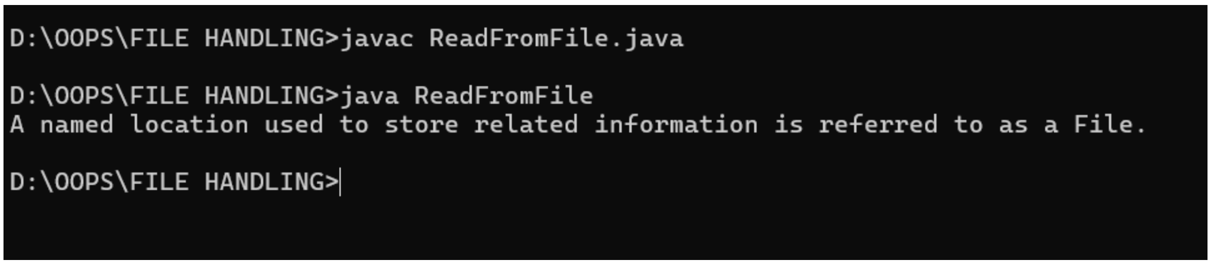
```
Microsoft Windows [Version 10.0.26100.3476]
(c) Microsoft Corporation. All rights reserved.

D:\>javac WriteToFile.java

D:\>java WriteToFile
Content is successfully wrote to the file.

D:\>
```

17.D)

CODE:

```
import java.io.File;


import java.io.FileNotFoundException;
import java.util.Scanner;


class ReadFromFile {
    public static void main(String[] args) {
        try {
            // Create f1 object of the file to read data
            File f1 = new File("RAAHI.txt");
            Scanner dataReader = new Scanner(f1);
            while (dataReader.hasNextLine()) {
                String fileData = dataReader.nextLine();
                System.out.println(fileData);
            }
            dataReader.close();
        } catch (FileNotFoundException exception) {
            System.out.println("Unexcpected error occurred!");
            exception.printStackTrace();
        }
    }
}
```

OUTPUT:



```
D:\OOPS\FILE HANDLING>javac ReadFromFile.java

D:\OOPS\FILE HANDLING>java ReadFromFile
A named location used to store related information is referred to as a File.

D:\OOPS\FILE HANDLING>
```