LAB RECORD

23CSE111- Object Oriented Programming

Submitted by

CH.SC.U4CSE24162-
B.H.SUMITRANAND

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

AMRITA VISHWA VIDYAPEETHAM

AMRITA SCHOOL OF COMPUTING

CHENNAI

March - 2025

A

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.U4CSE24162 – BH.SUMITRANAND*** 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

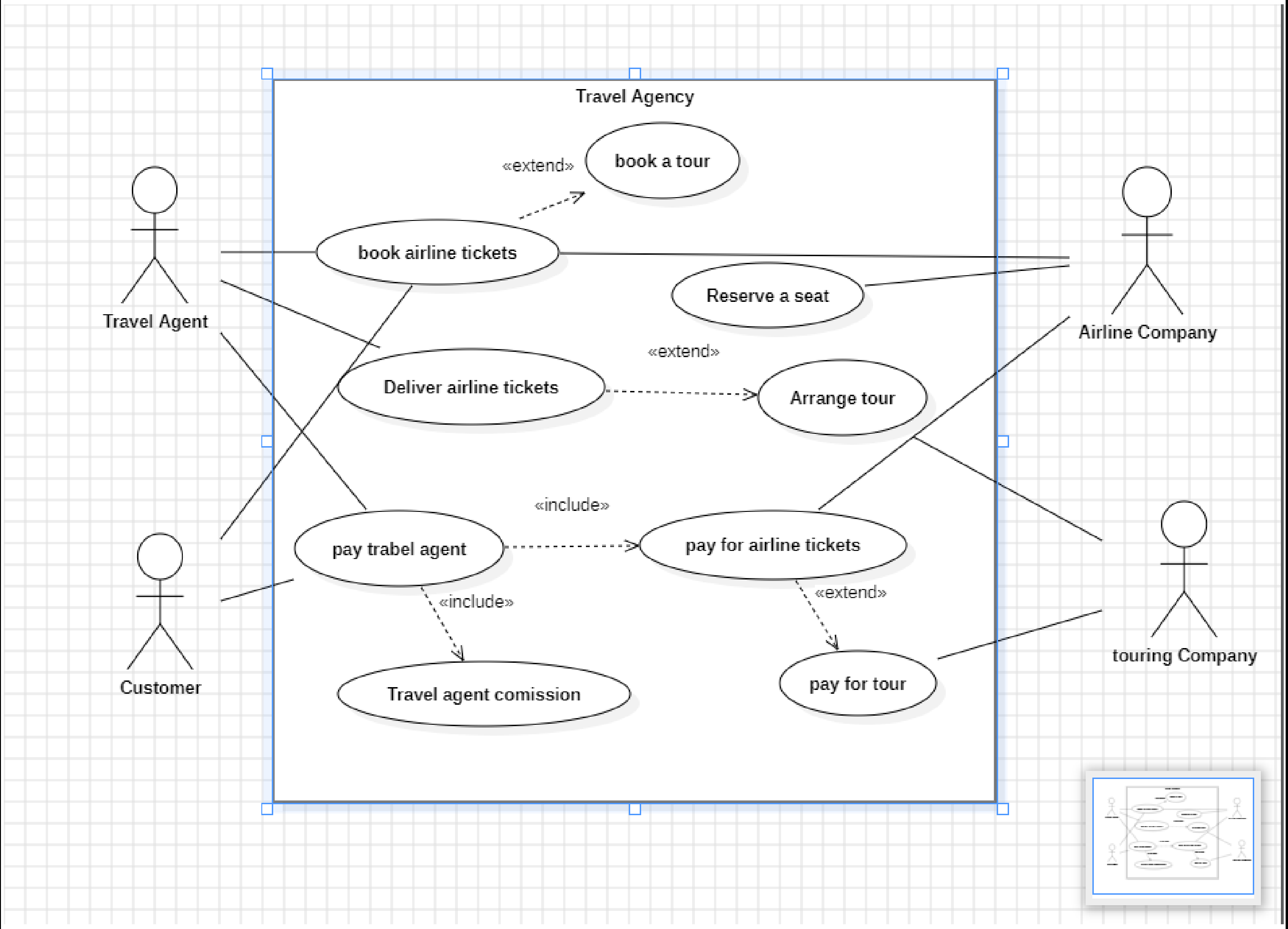
Internal Examiner 1 Internal Examiner 2

INDEX

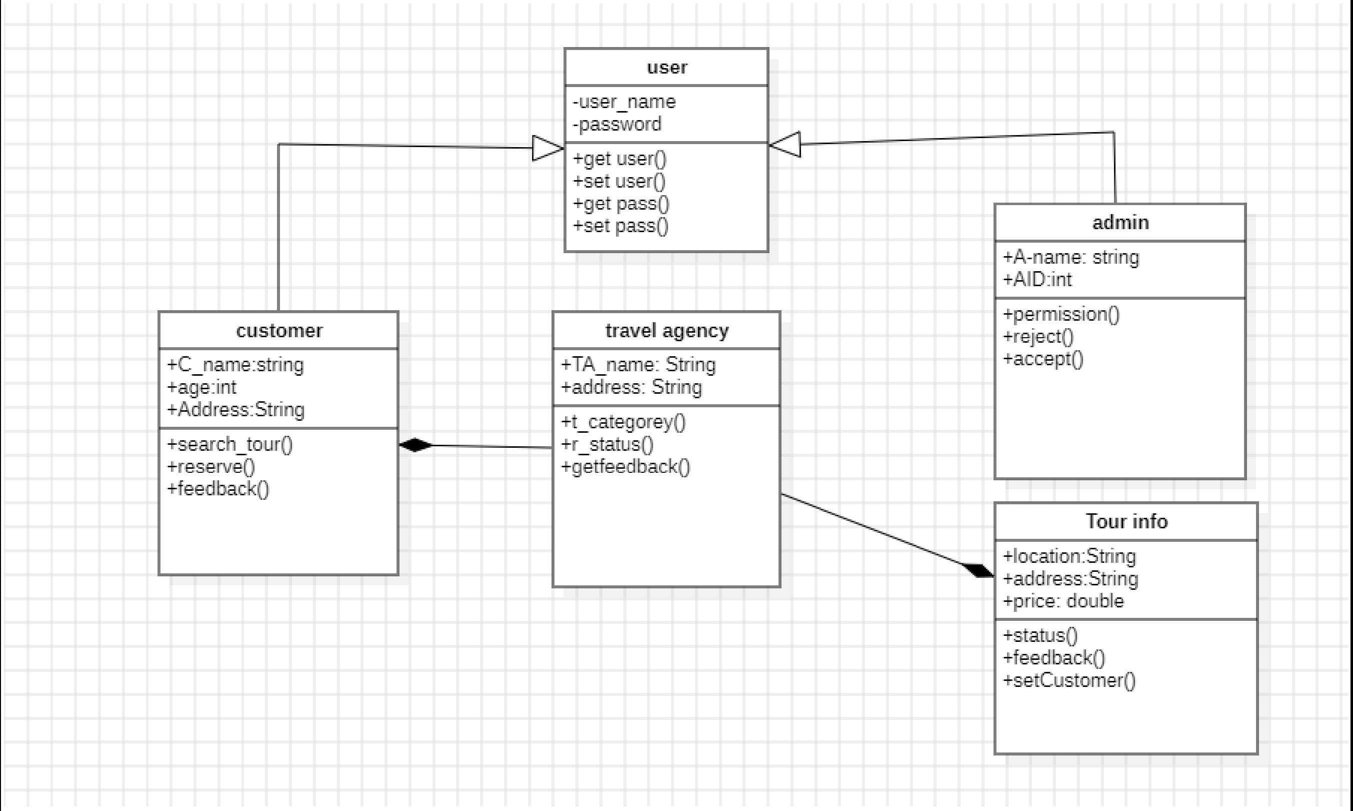
UML DIAGRAMS

1. TRAVEL AGENCY

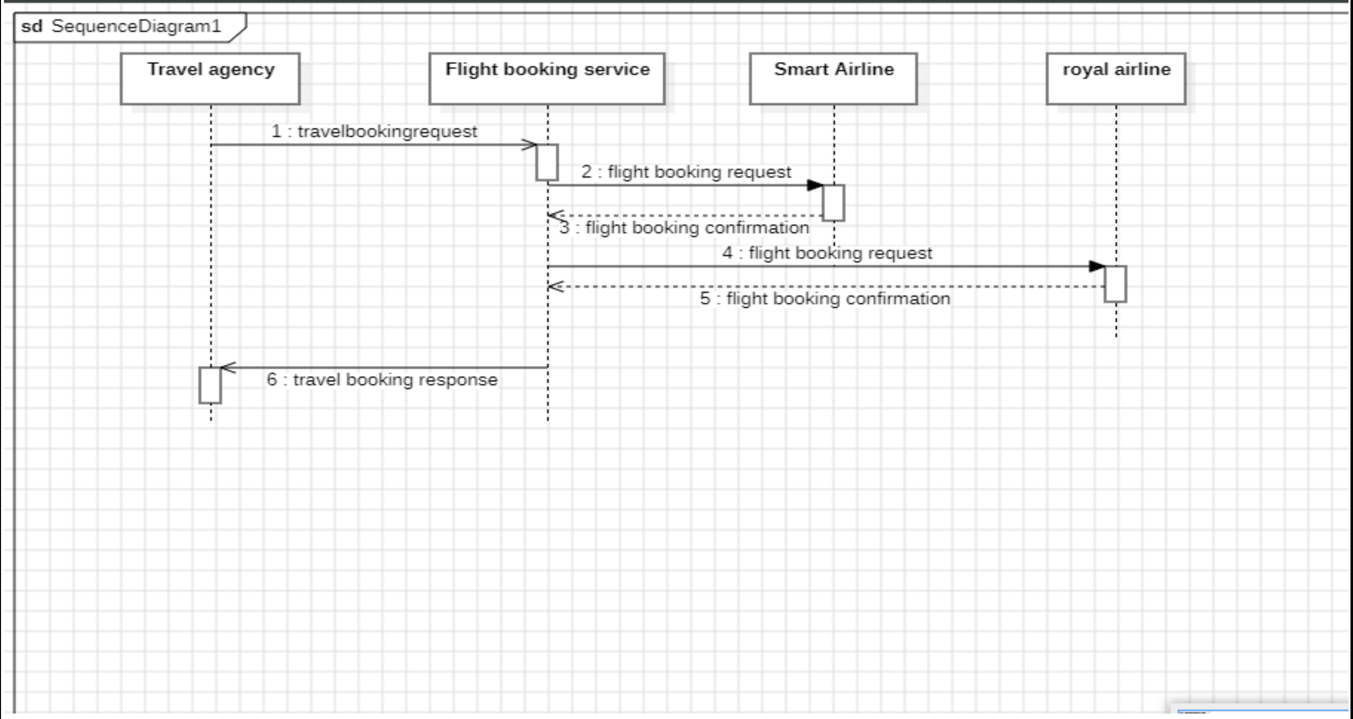
1.a) Use Case Diagram:



1.b) Class Diagram:

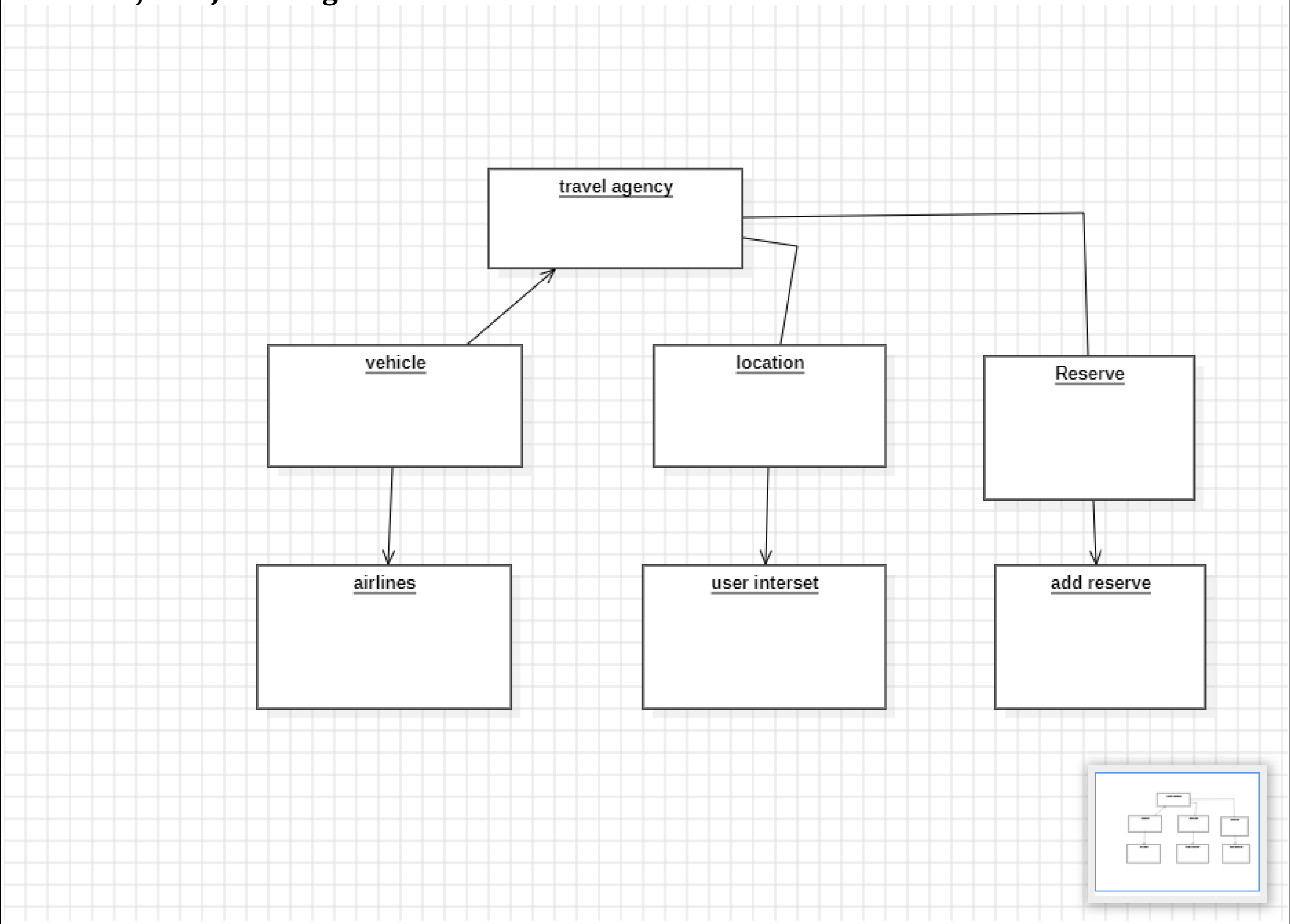


1.c) Sequence Diagram:

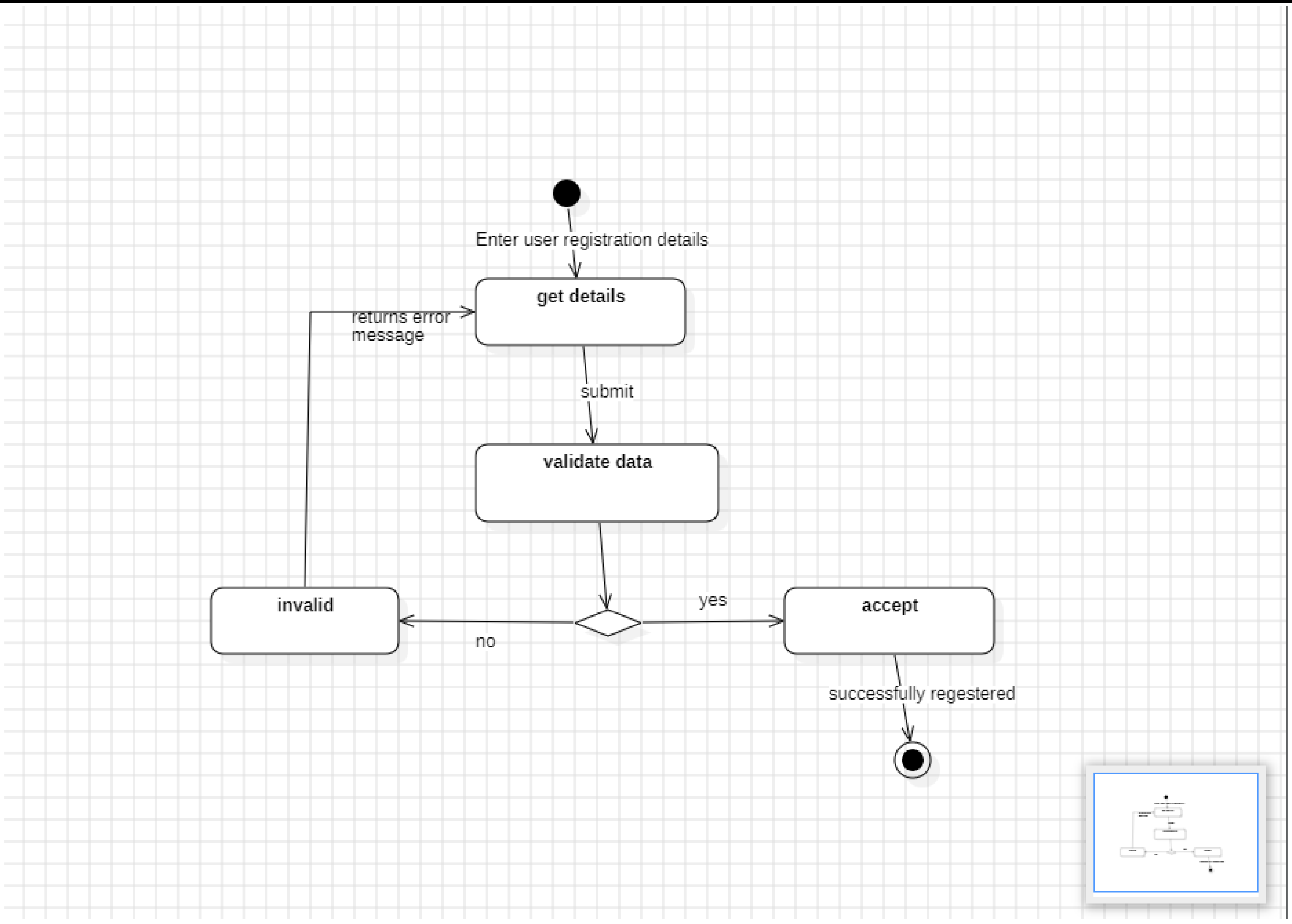


sd SequenceDiagram1
Travel agency
Flight booking service
Smart Airline
royal airline
1 : travelbookingrequest
2 : flight booking request
3 : flight booking confirmation
4 : flight booking request
5 : flight booking confirmation
6 : travel booking response

1.d) Object Diagram:

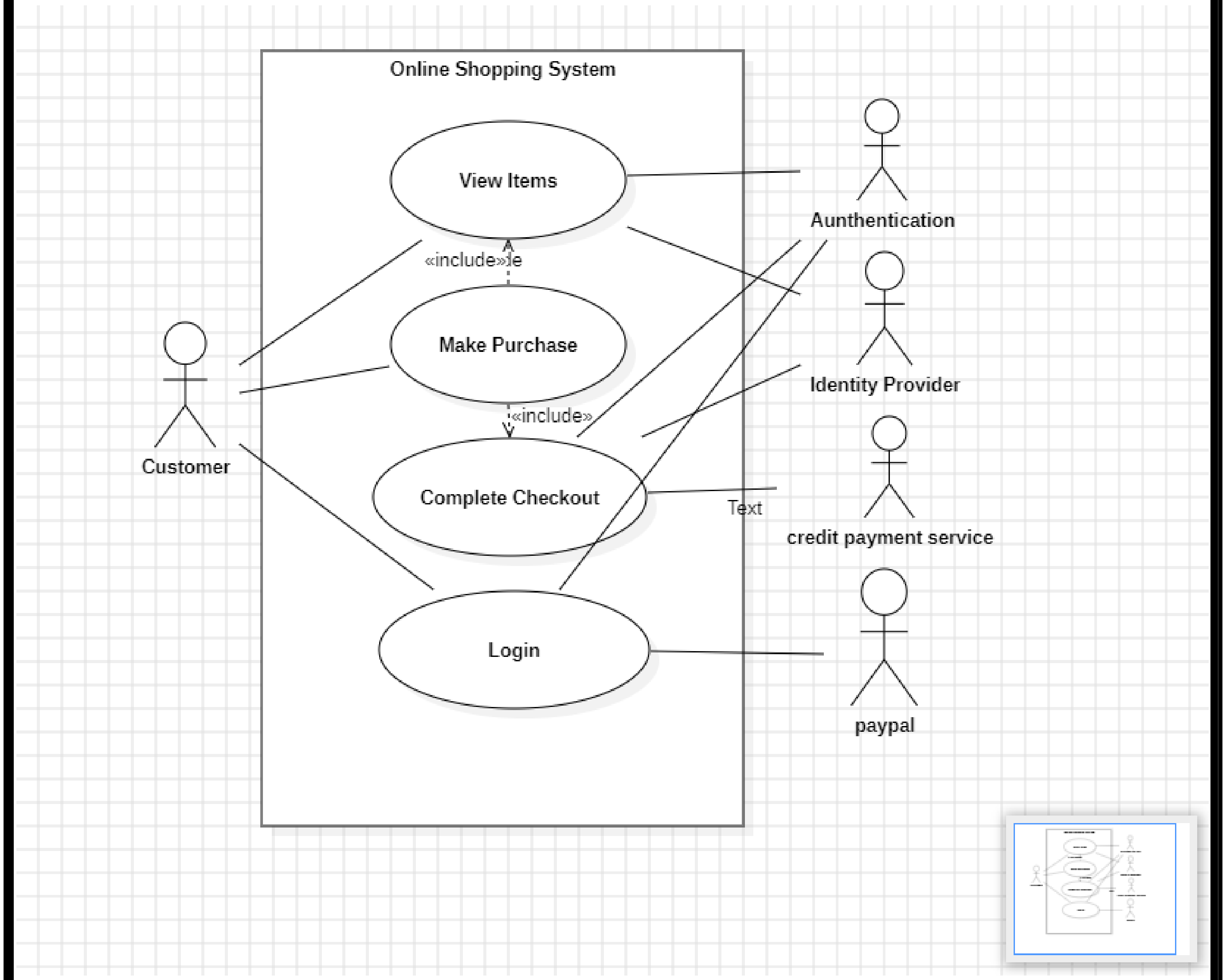


1.e) State-Activity Diagram:

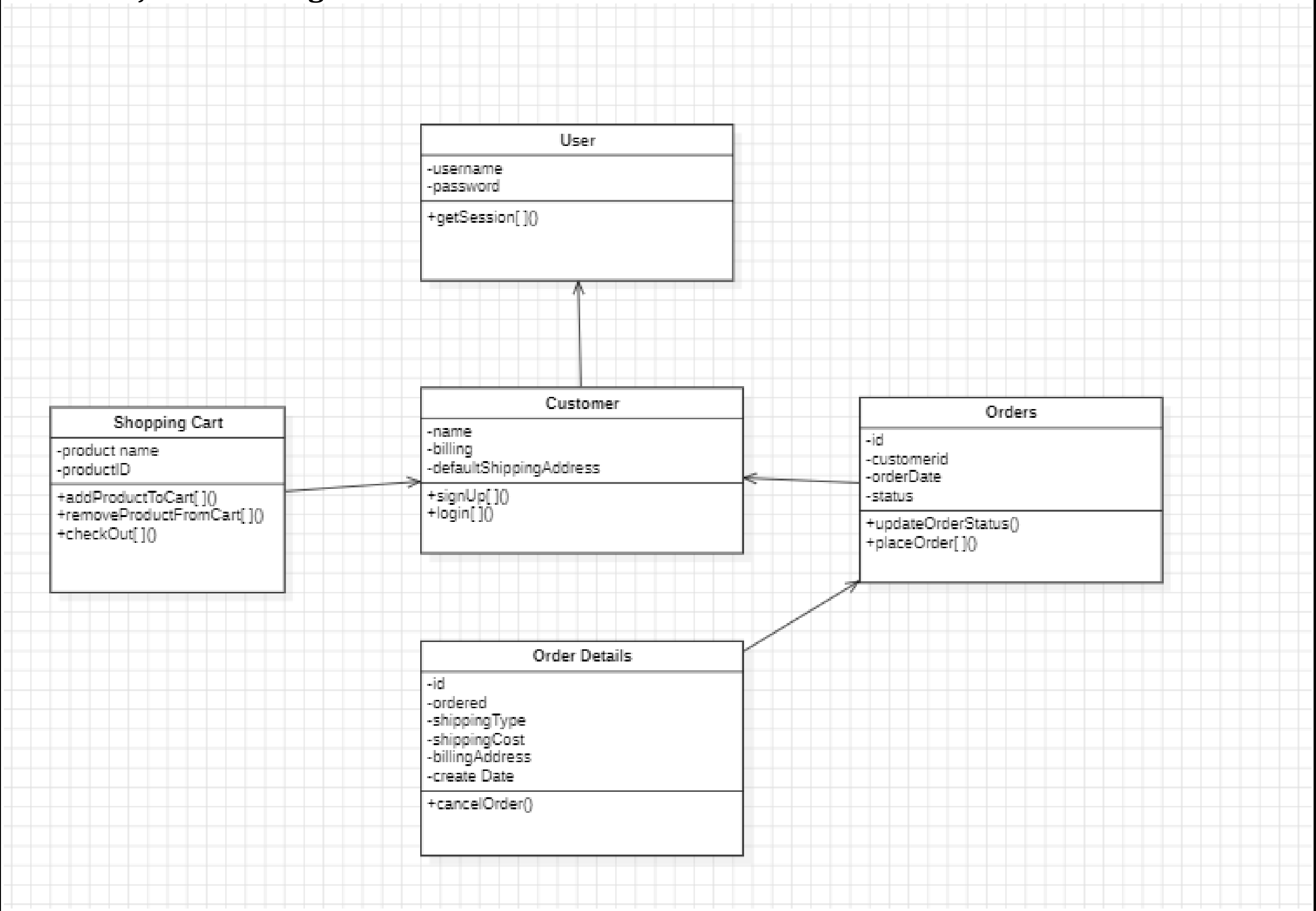

Enter user registration details
get details
returns error message
submit
validate data
invalid
no
yes
accept
successfully regestered

2. ONLINE SHOPPING SYSTEM

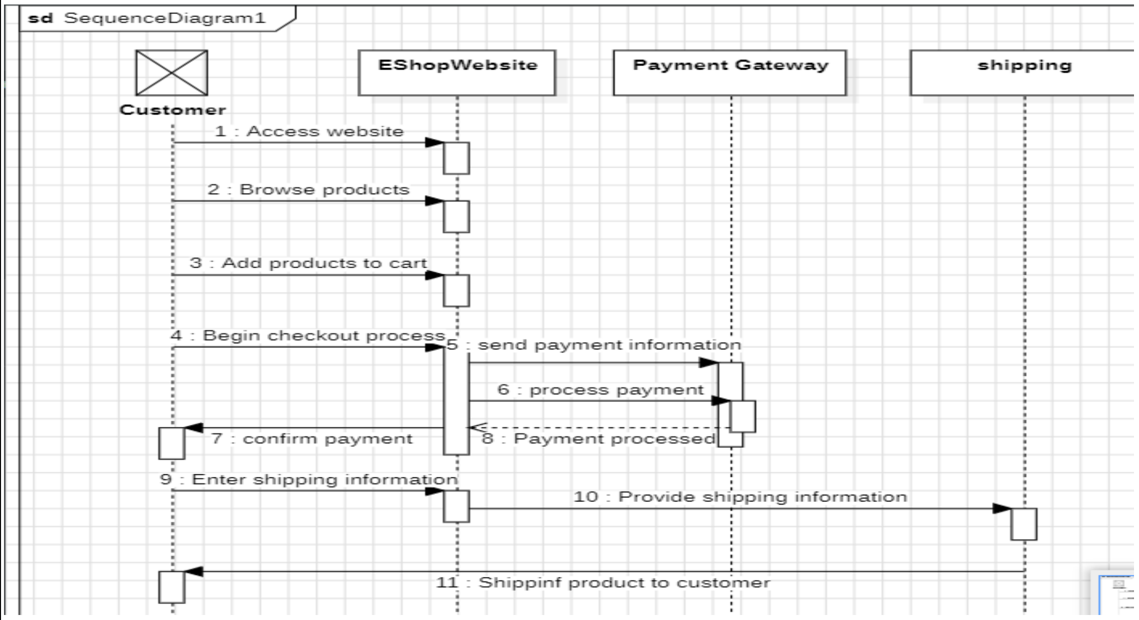
2.a) Use Case Diagram:



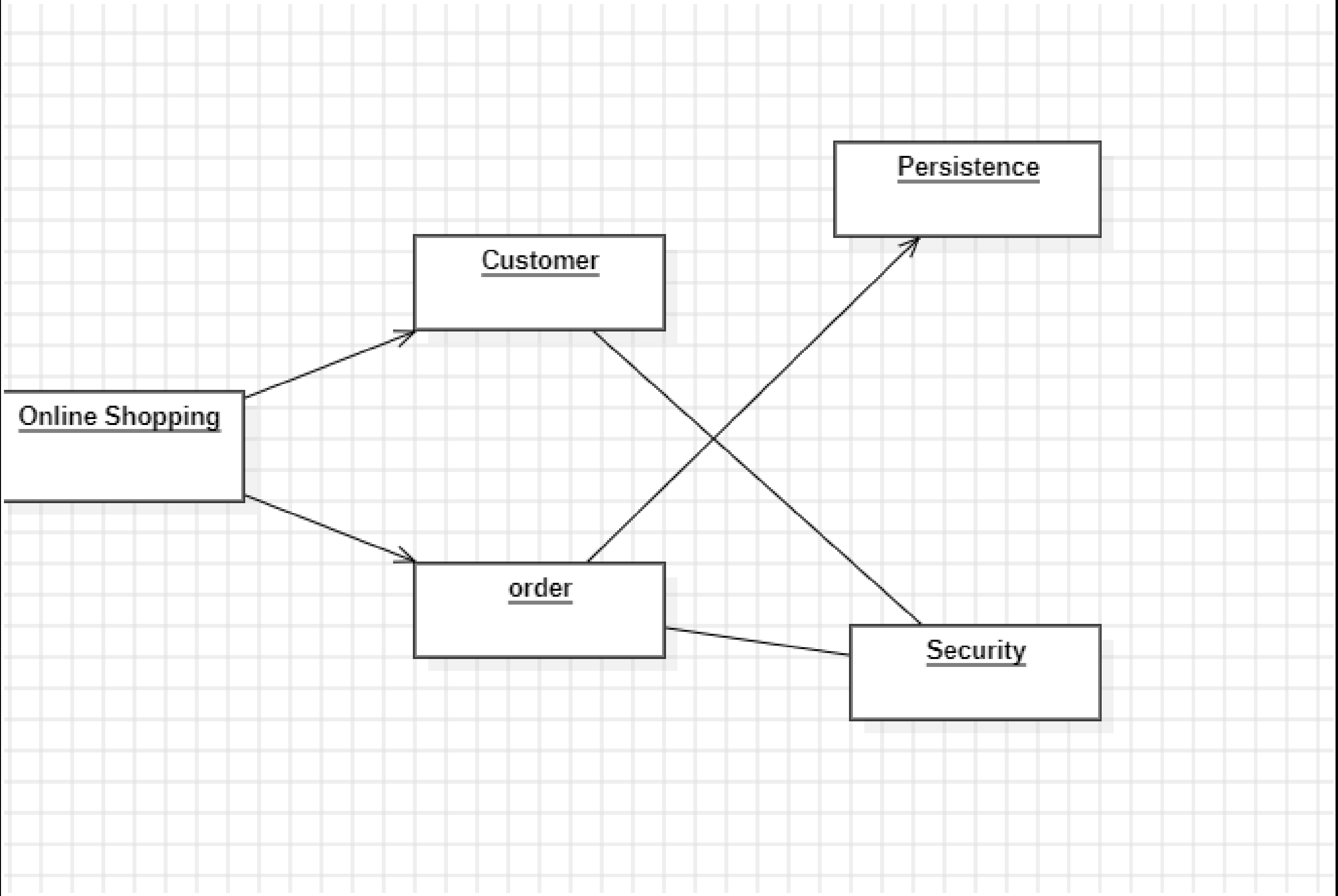
2.b) Class Diagram:



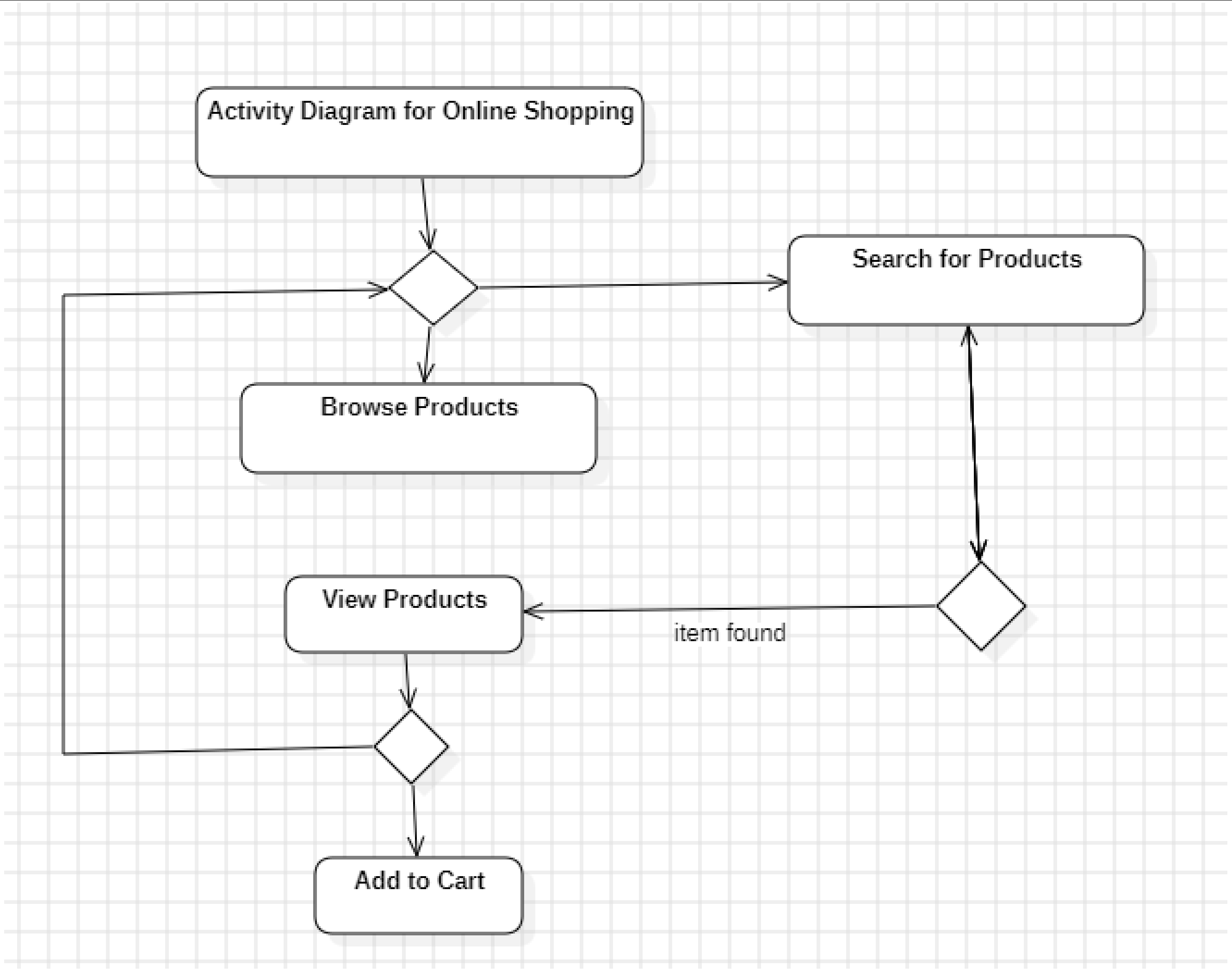
2.c) Sequence Diagram:



2.d) Object Diagram:



2.e) State-Activity Diagram:


Activity Diagram for Online Shopping
Search for Products
Browse Products
View Products
item found
Add to Cart

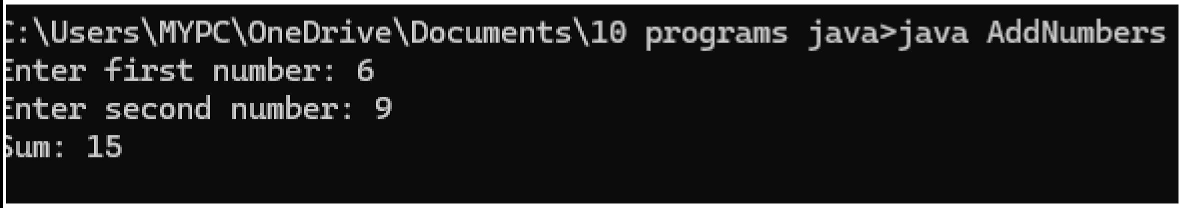
3. Basic Java Programs

3.a) Add Numbers

Code:

```
import java.util.Scanner;

public class AddNumbers {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      System.out.print("Enter first number: ");
      int num1 = scanner.nextInt();
      System.out.print("Enter second number: ");
      int num2 = scanner.nextInt();
      int sum = num1 + num2;
      System.out.println("Sum: " + sum);
      scanner.close();
   }
}
```

Output:



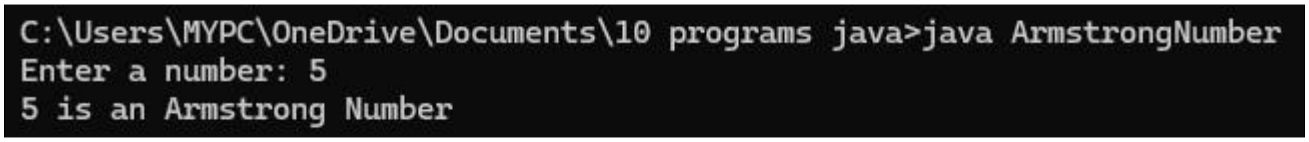
```
:\Users\MYPC\OneDrive\Documents\10 programs java>java AddNumbers
nter first number: 6
nter second number: 9
um: 15
```

3.b) Armstrong Number

Code:

```
import java.util.Scanner;

public class ArmstrongNumber {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int num = scanner.nextInt();
        int originalNum = num, sum = 0, digit;
        int digits = String.valueOf(num).length();

        while (num != 0) {
            digit = num % 10;
            sum += Math.pow(digit, digits);
            num /= 10;
        }

        System.out.println(originalNum + (sum == originalNum ? "
is an Armstrong Number" : " is not an Armstrong Number"));
        scanner.close();
    }
}
```

Output:



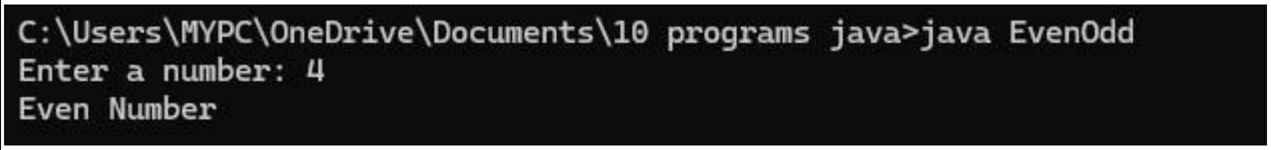
```
C:\Users\MYPC\OneDrive\Documents\10 programs java>java ArmstrongNumber
Enter a number: 5
5 is an Armstrong Number
```

3.c) Even odd

Code:

```
import java.util.Scanner;

public class EvenOdd {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int num = scanner.nextInt();
        if (num % 2 == 0) {
            System.out.println("Even Number");
        } else {
            System.out.println("Odd Number");
        }
        scanner.close();
    }
}
```

Output:



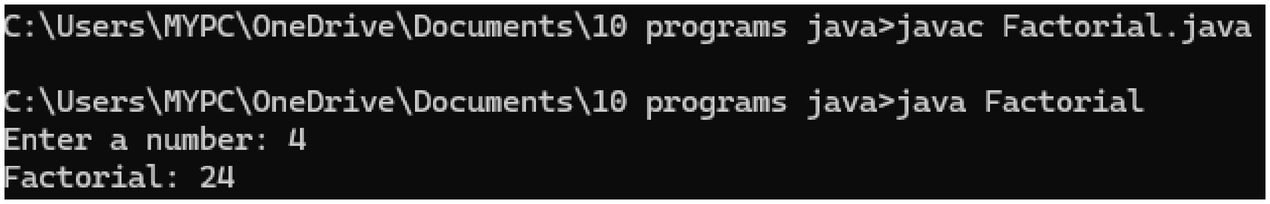
```
C:\Users\MYPC\OneDrive\Documents\10 programs java>java EvenOdd
Enter a number: 4
Even Number
```

3.d) Factorial

Code:

```
import java.util.Scanner;

public class Factorial {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a  number:  ");
        int num = scanner.nextInt();
        long fact = 1;

        for (int i = 1; i <= num; i++) {
            fact *= i;
        }

        System.out.println("Factorial: " + fact);
        scanner.close();
    }
}
```

Output;



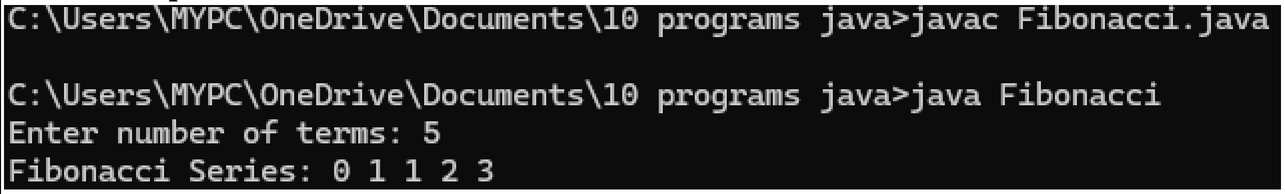
```
C:\Users\MYPC\OneDrive\Documents\10 programs java>javac Factorial.java

C:\Users\MYPC\OneDrive\Documents\10 programs java>java Factorial
Enter a number: 4
Factorial: 24
```

3.e) Fibonacci

Code:

```java
import java.util.Scanner;

public class Fibonacci {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter number of terms: ");
        int n = scanner.nextInt();
        int a = 0, b = 1, c;

        System.out.print("Fibonacci Series: " + a + " " + b);
        for (int i = 2; i < n; i++) {
            c = a + b;
            System.out.print(" " + c);
            a = b;
            b = c;
        }

        scanner.close();
    }
}
```

Output:



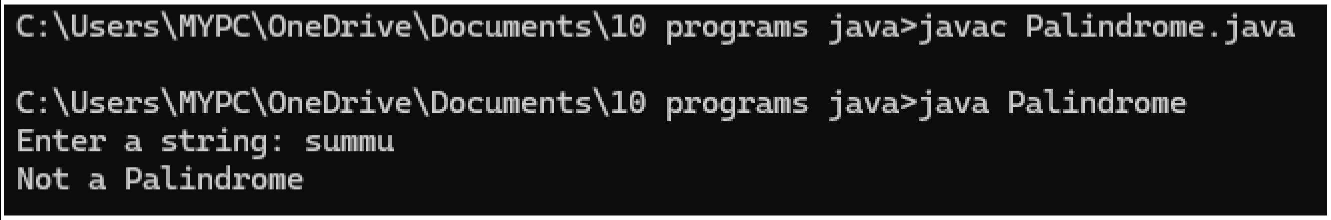
```
C:\Users\MYPC\OneDrive\Documents\10 programs java>javac Fibonacci.java

C:\Users\MYPC\OneDrive\Documents\10 programs java>java Fibonacci
Enter number of terms: 5
Fibonacci Series: 0 1 1 2 3
```

3.f) Palindrome

Code:

```
import java.util.Scanner;

public class Palindrome {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String str = scanner.nextLine();
        String reversed = new StringBuilder(str).reverse().toString();

        if (str.equalsIgnoreCase(reversed)) {
            System.out.println("Palindrome");
        } else {
            System.out.println("Not a Palindrome");
        }

        scanner.close();
    }
}
```

Output:



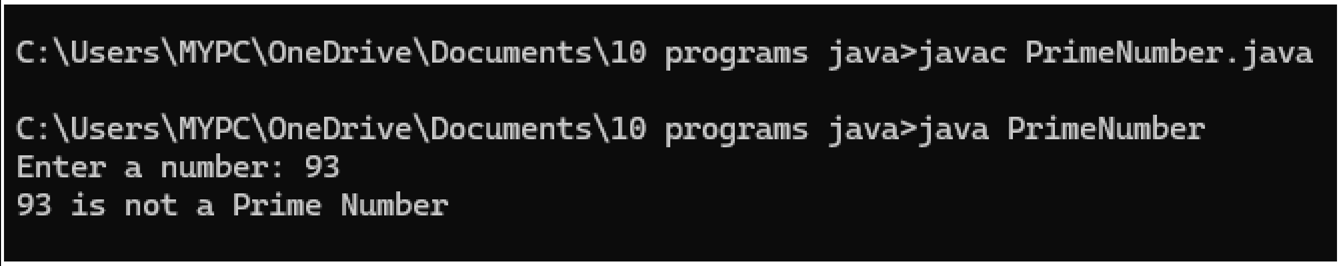
```
C:\Users\MYPC\OneDrive\Documents\10 programs java>javac Palindrome.java

C:\Users\MYPC\OneDrive\Documents\10 programs java>java Palindrome
Enter a string: summu
Not a Palindrome
```

3.g) Prime Number

Code:

```
import java.util.Scanner;

public class PrimeNumber {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a  number:  ");
        int num = scanner.nextInt();
        boolean isPrime = num > 1;

        for (int i = 2; i <= Math.sqrt(num); i++) {
            if (num % i == 0) {
                isPrime = false;
                break;
            }
        }

        System.out.println(num + (isPrime ? " is a Prime Number" : " is not a Prime
Number"));
        scanner.close();
    }
}
```

Output:



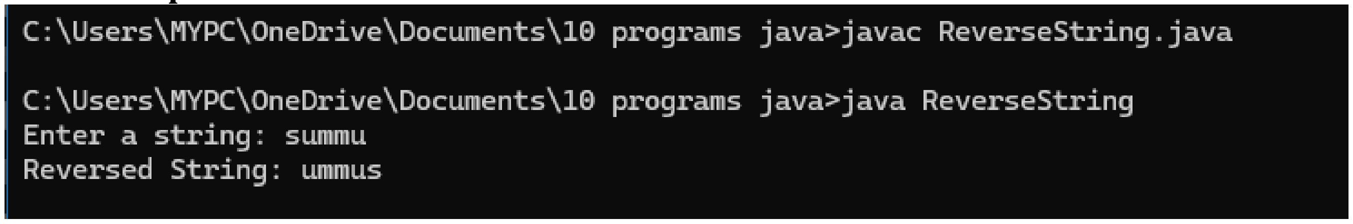
```
C:\Users\MYPC\OneDrive\Documents\10 programs java>javac PrimeNumber.java

C:\Users\MYPC\OneDrive\Documents\10 programs java>java PrimeNumber
Enter a number: 93
93 is not a Prime Number
```

3.h) Reverse string

Code:

```
import java.util.Scanner;

public class ReverseString {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String str = scanner.nextLine();
        String reversed = new StringBuilder(str).reverse().toString();
        System.out.println("Reversed String: " + reversed);
        scanner.close();
    }
}
```

Output:



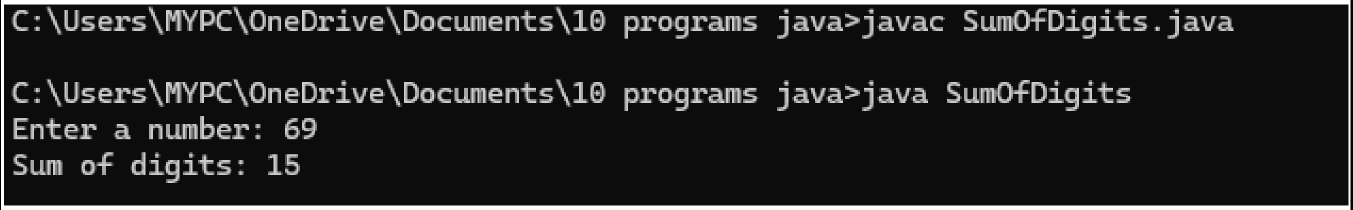
```
C:\Users\MYPC\OneDrive\Documents\10 programs java>javac ReverseString.java

C:\Users\MYPC\OneDrive\Documents\10 programs java>java ReverseString
Enter a string: summu
Reversed String: ummus
```

3.i) Sum of digits

Code:

```java
import java.util.Scanner;

public class SumOfDigits {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a  number:  ");
        int num = scanner.nextInt();
        int sum = 0;

        while (num != 0) {
            sum += num % 10;
            num /= 10;
        }

        System.out.println("Sum of digits: " + sum);
        scanner.close();
    }
}
```

Output:



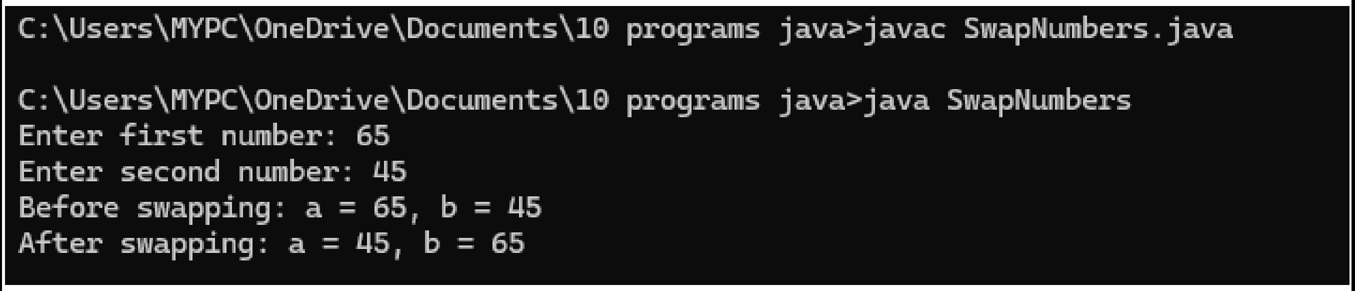
```
C:\Users\MYPC\OneDrive\Documents\10 programs java>javac SumOfDigits.java

C:\Users\MYPC\OneDrive\Documents\10 programs java>java SumOfDigits
Enter a number: 69
Sum of digits: 15
```

3.j) Swap Numbers

Code:

```
import java.util.Scanner;

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

        System.out.print("Enter first number: ");
        int a = scanner.nextInt();

        System.out.print("Enter second number: ");
        int b = scanner.nextInt();

        System.out.println("Before swapping: a = " + a + ", b = " + b);

        // Swapping without a third variable
        a = a + b;
        b = a - b;
        a = a - b;

        System.out.println("After swapping: a = " + a + ", b = " + b);

        scanner.close();
    }
}
```

Output:



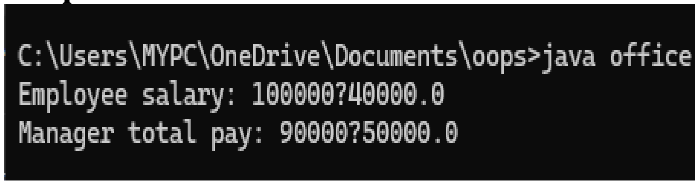
```
C:\Users\MYPC\OneDrive\Documents\10 programs java>javac SwapNumbers.java

C:\Users\MYPC\OneDrive\Documents\10 programs java>java SwapNumbers
Enter first number: 65
Enter second number: 45
Before swapping: a = 65, b = 45
After swapping: a = 45, b = 65
```

4. Single inheritence programs

4(a) employee salary

Code:
```
class Employee {
   double salary = 40000;

   void showSalary() {
      System.out.println("Employee salary: 100000₹" + salary);
   }
}

// Child class
class Manager extends Employee {
   int bonus = 10000;

   void showTotalPay() {
      System.out.println("Manager total pay: 90000₹" + (salary + bonus));
   }
}

// Main class
public class office {
   public static void main(String[] args) {
      Manager m = new Manager();
      m.showSalary(); // inherited from Employee
      m.showTotalPay(); // defined in Manager
   }
}
```

Output:

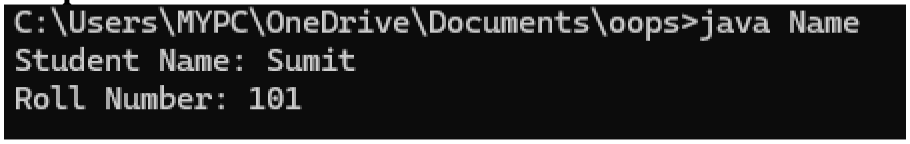


```
C:\Users\MYPC\OneDrive\Documents\oops>java office
Employee salary: 100000?40000.0
Manager total pay: 90000?50000.0
```

4(b)
Code:
```
// Parent class
class Person {
   String name = "Unknown";

   void displayInfo() {
      System.out.println("Name: " + name);
   }
}

// Child class
class Student extends Person {
   int rollNumber = 101;
```

```
    // Overriding method
    @Override
    void displayInfo() {
        name = "Sumit";
        System.out.println("Student Name: " + name);
        System.out.println("Roll Number: " + rollNumber);
    }
}

// Main class
public class Name {
    public static void main(String[] args) {
        Student s = new Student();
        s.displayInfo();  // Calls overridden method
    }
}
```

Output:



```
C:\Users\MYPC\OneDrive\Documents\oops>java Name
Student Name: Sumit
Roll Number: 101
```

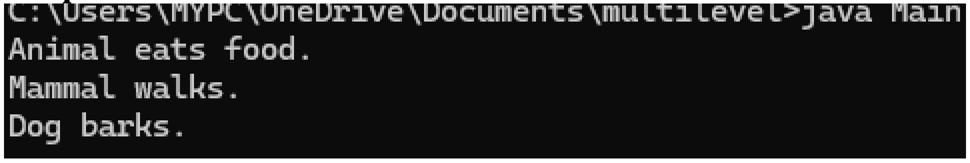
5. Multilevel inheritence

5(a) Animal

Code:

```
// Grandparent class
class Animal {
    void eat() {
        System.out.println("Animal eats food.");
    }
}

// Parent class
class Mammal extends Animal {
    void walk() {
        System.out.println("Mammal walks.");
    }
}

// Child class
class Dog extends Mammal {
    void bark() {
        System.out.println("Dog barks.");
    }
}

// Main class
public class Main {
    public static void main(String[] args) {
        Dog d = new Dog();
        d.eat();   // from Animal
        d.walk();  // from Mammal
        d.bark();  // from Dog
    }
}
```

Output:



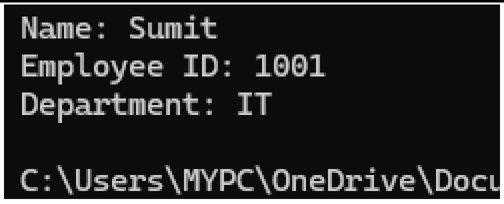
```
C:\Users\MYPC\OneDrive\Documents\multilevel>java Main
Animal eats food.
Mammal walks.
Dog barks.
```

5(b) Managers

Code:

```
// Grandparent class
class Person {
    String name = "Sumit";

    void displayPerson() {
        System.out.println("Name: " + name);
    }
}

// Parent class
class Employee extends Person {
    int empId = 1001;

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

// Child class
class Manager extends Employee {
    String dept = "IT";

    void displayManager() {
        System.out.println("Department: " + dept);
    }
}

// Main class
public class Managers {
    public static void main(String[] args) {
        Manager m = new Manager();
        m.displayPerson();     // from Person
        m.displayEmployee();   // from Employee
        m.displayManager();    // from Manager
    }
}
```

Output:


```
Name: Sumit
Employee ID: 1001
Department: IT

C:\Users\MYPC\OneDrive\Docu
```

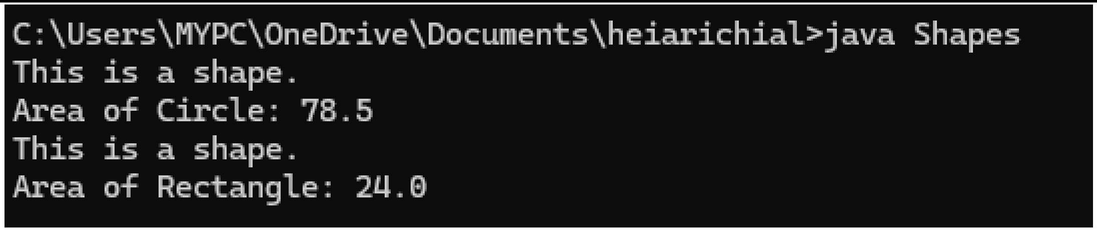
6. HIERARCHICAL INHERITANCE PROGRAMS

6(a) Shapes

Code:

```
// Parent class
class Shape {
   void display() {
      System.out.println("This is a shape.");
   }
}

// Child class 1
class Circle extends Shape {
   void areaOfCircle(double radius) {
      double area = 3.14 * radius * radius;
      System.out.println("Area of Circle: " + area);
   }
}

// Child class 2
class Rectangle extends Shape {
   void areaOfRectangle(double length, double width) {
      double area = length * width;
      System.out.println("Area of Rectangle: " + area);
   }
}

// Main class
public class Shapes {
   public static void main(String[] args) {
      Circle c = new Circle();
      Rectangle r = new Rectangle();

      c.display(); // from Shape
      c.areaOfCircle(5);

      r.display(); // from Shape
      r.areaOfRectangle(4, 6);
   }
}
```

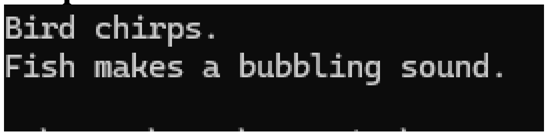
Output:


```
C:\Users\MYPC\OneDrive\Documents\heiarichial>java Shapes
This is a shape.
Area of Circle: 78.5
This is a shape.
Area of Rectangle: 24.0
```

6(b) Animal
Code:

```
class Animal {
protected String species;
protected int age;
public Animal(String species, int age) {
this.species = species;
this.age = age;
}
public void makeSound() {
System.out.println("Animal makes a sound.");
}
}
class Bird extends Animal {
private double wingSpan;
public Bird(String species, int age, double wingSpan) {
super(species, age);
this.wingSpan = wingSpan;
}
public void makeSound() {
System.out.println("Bird chirps.");
}
}
class Fish extends Animal {CH.SC.U4CSE24163
33
HARISH.R
private String waterType;
public Fish(String species, int age, String waterType) {
super(species, age);
this.waterType = waterType;
}
public void makeSound() {
System.out.println("Fish makes a bubbling sound.");
}
}
public class AnimalClassificationSystem {
public static void main(String[] args) {
Bird bird = new Bird("Parrot", 2, 1.5);
Fish fish = new Fish("Goldfish", 1, "Freshwater");
bird.makeSound();
fish.makeSound();
}
}
```

Output:

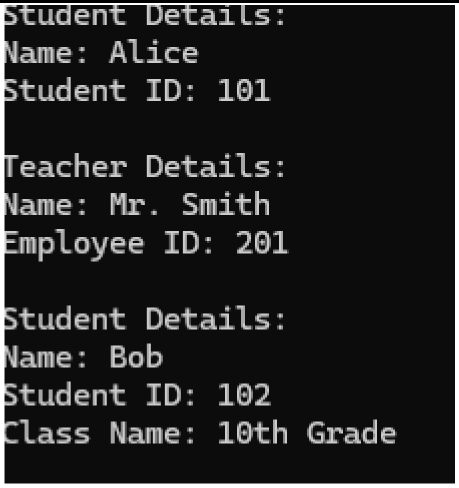


```
Bird chirps.
Fish makes a bubbling sound.
```

7. HYBRID INHERITANCE PROGRAMS

7(a)
Code:

```
interface Person {
void displayInfo();
}
class Student implements Person {
protected String name;
protected int studentId;
public Student(String name, int studentId) {
this.name = name;
this.studentId = studentId;
}
public void displayInfo() {
System.out.println("Student Details:");
System.out.println("Name: " + name);
System.out.println("Student ID: " + studentId);
}
}
class Teacher implements Person {
protected String name;
protected int employeeId;
public Teacher(String name, int employeeId) {
this.name = name;
this.employeeId = employeeId;
}
public void displayInfo() {
System.out.println("Teacher Details:");
System.out.println("Name: " + name);
System.out.println("Employee ID: " + employeeId);
}
}
class Classroom extends Student {
private String className;
public Classroom(String name, int studentId, String className) {
super(name, studentId);
this.className = className;
}
public void displayInfo() {
super.displayInfo();
System.out.println("Class Name: " + className);
}
}
public class SchoolManagementSystem {
public static void main(String[] args) {
Student student = new Student("Alice", 101);
Teacher teacher = new Teacher("Mr. Smith", 201);
Classroom classroom = new Classroom("Bob", 102, "10th Grade");
student.displayInfo();
System.out.println();
teacher.displayInfo();
System.out.println();
classroom.displayInfo();
}
}
```
Output:



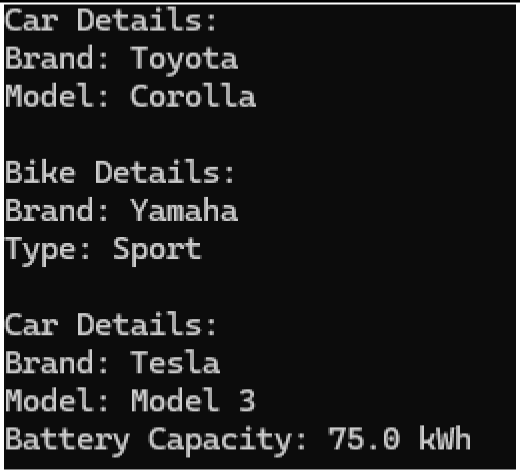
```
Student Details:
Name: Alice
Student ID: 101

Teacher Details:
Name: Mr. Smith
Employee ID: 201

Student Details:
Name: Bob
Student ID: 102
Class Name: 10th Grade
```

7(b)Vehicle management System:

Code:

```
Vehicle management System:
Code:
interface Vehicle {
void displayDetails();
}
class Car implements Vehicle {
protected String brand;
protected String model;
public Car(String brand, String model) {
this.brand = brand;
this.model = model;
}
public void displayDetails() {
System.out.println("Car Details:");
System.out.println("Brand: " + brand);
System.out.println("Model: " + model);
}
}
class Bike implements Vehicle {
protected String brand;
protected String type;
public Bike(String brand, String type) {
this.brand = brand;
this.type = type;
}
public void displayDetails() {
System.out.println("Bike Details:");
System.out.println("Brand: " + brand);
System.out.println("Type: " + type);
}
}
class ElectricCar extends Car {
private double batteryCapacity;
public ElectricCar(String brand, String model, double batteryCapacity) {
super(brand, model);
this.batteryCapacity = batteryCapacity;
}
public void displayDetails() {
super.displayDetails();
System.out.println("Battery Capacity: " + batteryCapacity + " kWh");
}
}
public class VehicleRentalSystem {
public static void main(String[] args) {
Car car = new Car("Toyota", "Corolla");
Bike bike = new Bike("Yamaha", "Sport");
ElectricCar electricCar = new ElectricCar("Tesla", "Model 3", 75.0);
car.displayDetails();
System.out.println();
bike.displayDetails();
System.out.println();
electricCar.displayDetails();
}
}
```

Output:


```
Car Details:
Brand: Toyota
Model: Corolla

Bike Details:
Brand: Yamaha
Type: Sport

Car Details:
Brand: Tesla
Model: Model 3
Battery Capacity: 75.0 kWh
```

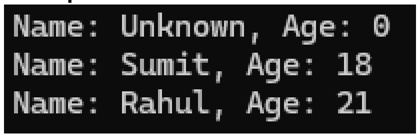
8. CONSTRUCTOR PROGRAMS

8(a) Student

Code:

```
class Student {
    String name;
    int age;

    // Constructor 1 - No arguments
    Student() {
        name = "Unknown";
        age = 0;
    }

    // Constructor 2 - One argument
    Student(String n) {
        name = n;
        age = 18; // default age
    }

    // Constructor 3 - Two arguments
    Student(String n, int a) {
        name = n;
        age = a;
    }

    void display() {
        System.out.println("Name: " + name + ", Age: " + age);
    }
}

public class Main {
    public static void main(String[] args) {
        Student s1 = new Student();              // calls
Constructor 1
        Student s2 = new Student("Sumit");       // calls
Constructor 2
        Student s3 = new Student("Rahul", 21);    // calls
Constructor 3

        s1.display();
        s2.display();
        s3.display();
    }
}
```

Output:



```
Name: Unknown, Age: 0
Name: Sumit, Age: 18
Name: Rahul, Age: 21
```

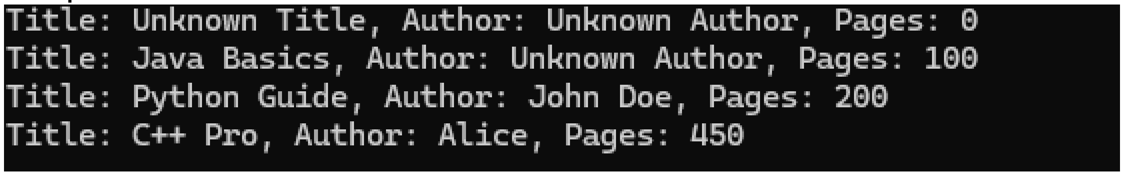
9. Constructor overloading programs

9(a) books management

Code:

```
class Book {
    String title;
    String author;
    int pages;

    // Constructor 1: No parameters
    Book() {
        title = "Unknown Title";
        author = "Unknown Author";
        pages = 0;
    }

    // Constructor 2: Title only
    Book(String t) {
        title = t;
        author = "Unknown Author";
        pages = 100;
    }

    // Constructor 3: Title and Author
    Book(String t, String a) {
        title = t;
        author = a;
        pages = 200;
    }

    // Constructor 4: All parameters
    Book(String t, String a, int p) {
        title = t;
        author = a;
        pages = p;
    }

    void display() {
        System.out.println("Title: " + title + ", Author: " + 
author + ", Pages: " + pages);
    }
}

public class BookManagement {
    public static void main(String[] args) {
        Book b1 = new Book();
        Book b2 = new Book("Java Basics");
        Book b3 = new Book("Python Guide", "John Doe");
        Book b4 = new Book("C++ Pro", "Alice", 450);

        b1.display();
        b2.display();
        b3.display();
        b4.display();
    }
}
```

Output:



```
Title: Unknown Title, Author: Unknown Author, Pages: 0
Title: Java Basics, Author: Unknown Author, Pages: 100
Title: Python Guide, Author: John Doe, Pages: 200
Title: C++ Pro, Author: Alice, Pages: 450
```

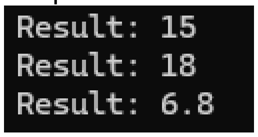
10. METHOD OVERLOADING PROGRAMS

10(a) calculator

Code:

```
class Calculator {
public int add(int a, int b) {
return a + b;
}
public int add(int a, int b, int c) {
return a + b + c;
}
public double add(double a, double b) {
return a + b;
}
public void displayResult(int result) {
System.out.println("Result: " + result);
}
public void displayResult(double result) {
System.out.println("Result: " + result);
}
}
public class Calculator12 {
public static void main(String[] args) {
Calculator calculator = new Calculator();
int sum1 = calculator.add(5, 10);
int sum2 = calculator.add(3, 6, 9);
double sum3 = calculator.add(4.5, 2.3);
calculator.displayResult(sum1);
calculator.displayResult(sum2);
calculator.displayResult(sum3);
}
}
```

Output:



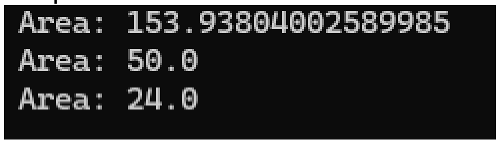
```
Result: 15
Result: 18
Result: 6.8
```

10(b)Area

Code:

```
class Shape {
public double calculateArea(double radius) {
return Math.PI * radius * radius;
}
public double calculateArea(double length, double width) {
return length * width;
}
public double calculateArea(double base, double height, boolean isTriangle)
{
if (isTriangle) {
return 0.5 * base * height;
}
return 0.0;
}
public void displayArea(double area) {
System.out.println("Area: " + area);
}
}
public class ShapeAreaCalculator {
public static void main(String[] args) {
Shape shape = new Shape();
double circleArea = shape.calculateArea(7.0);
double rectangleArea = shape.calculateArea(5.0, 10.0);
double triangleArea = shape.calculateArea(6.0, 8.0, true);
shape.displayArea(circleArea);
shape.displayArea(rectangleArea);
shape.displayArea(triangleArea);
}
}
```

Output:



```
Area: 153.93804002589985
Area: 50.0
Area: 24.0
```

11. METHOD OVERRIDING PROGRAMS

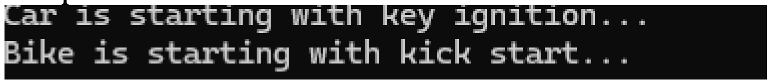
11(a)

Code:

```
class Vehicle {
public void start() {
System.out.println("Vehicle is starting...");
}
}
class Car extends Vehicle {
public void start() {
System.out.println("Car is starting with key ignition...");
}
}
class Bike extends Vehicle {
public void start() {
```

```
System.out.println("Bike is starting with kick start...");
}
}
public class Vehicle3 {
public static void main(String[] args) {
Vehicle myCar = new Car();
Vehicle myBike = new Bike();
myCar.start();
myBike.start();
}
}
```

Output:



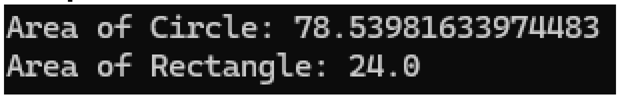
```
Car is starting with key ignition...
Bike is starting with kick start...
```

11(b)

Code:

```
class Shape {
public double area() {
return 0;
}
}
class Circle extends Shape {
private double radius;
public Circle(double radius) {
this.radius = radius;
}
public double area() {
return Math.PI * radius * radius;
}
}
class Rectangle extends Shape {
private double length;
private double width;
public Rectangle(double length, double width) {
this.length = length;
this.width = width;
}
public double area() {
return length * width;
}
}
public class Shape2 {
public static void main(String[] args) {
Shape myCircle = new Circle(5.0);
Shape myRectangle = new Rectangle(4.0, 6.0);
System.out.println("Area of Circle: " + myCircle.area());
System.out.println("Area of Rectangle: " + myRectangle.area());
}
}
```

Output:



```
Area of Circle: 78.53981633974483
Area of Rectangle: 24.0
```

12. Interface Programs

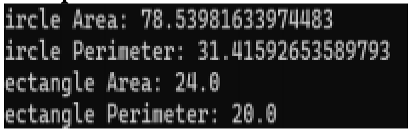
12(a)

Code:

```
import java.lang.Math;
interface Shape {
double area();
double perimeter();
}
class Circle implements Shape {
private double radius;
public Circle(double radius) {
this.radius = radius;
}
public double area() {
return Math.PI * radius * radius;
}
public double perimeter() {
return 2 * Math.PI * radius;
}
}
class Rectangle implements Shape {
private double width, height;
public Rectangle(double width, double height) {
this.width = width;
this.height = height;
}
public double area() {
return width * height;
}
public double perimeter() {
return 2 * (width + height);
}
}
public class Main {
```

```
public static void main(String[] args) {
Shape circle = new Circle(5);
Shape rectangle = new Rectangle(4, 6);
System.out.println("Circle Area: " + circle.area());
System.out.println("Circle Perimeter: " + circle.perimeter());
System.out.println("Rectangle Area: " + rectangle.area());
System.out.println("Rectangle Perimeter: " + rectangle.perimeter());
}
}
```

Output:



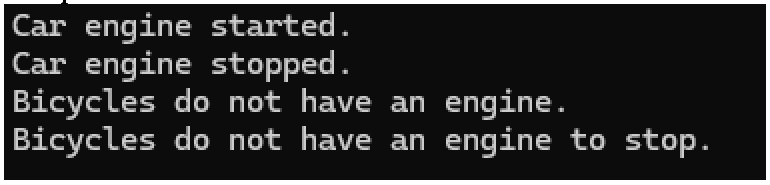
```
ircle Area: 78.53981633974483
ircle Perimeter: 31.41592653589793
ectangle Area: 24.0
ectangle Perimeter: 20.0
```

12(b)

Code:

```
interface Vehicle {
void startEngine();
void stopEngine();
}
class Car implements Vehicle {
public void startEngine() {
System.out.println("Car engine started.");
}
public void stopEngine() {
System.out.println("Car engine stopped.");
}
}
class Bicycle implements Vehicle {
public void startEngine() {
System.out.println("Bicycles do not have an engine.");
}
public void stopEngine() {
System.out.println("Bicycles do not have an engine to stop.");
}
}
public class V1 {
public static void main(String[] args) {
Vehicle car = new Car();
Vehicle bicycle = new Bicycle();
car.startEngine();
car.stopEngine();
bicycle.startEngine();
bicycle.stopEngine();
}
}
```

Output:



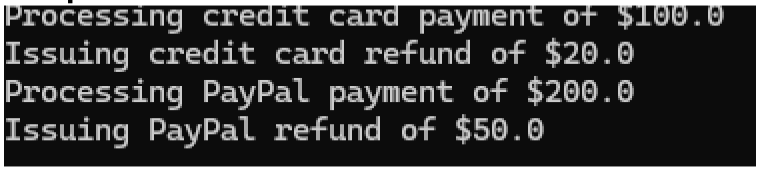
```
Car engine started.
Car engine stopped.
Bicycles do not have an engine.
Bicycles do not have an engine to stop.
```

12(c)`

Code:

```
interface Payment {
void processPayment(double amount);
void issueRefund(double amount);
}
class CreditCardPayment implements Payment {
public void processPayment(double amount) {
System.out.println("Processing credit card payment of $" + amount);
}
public void issueRefund(double amount) {
System.out.println("Issuing credit card refund of $" + amount);
}
}
class PayPalPayment implements Payment {
public void processPayment(double amount) {
System.out.println("Processing PayPal payment of $" + amount);
}
public void issueRefund(double amount) {
System.out.println("Issuing PayPal refund of $" + amount);
}
}
public class Pay {
public static void main(String[] args) {
Payment creditCard = new CreditCardPayment();
Payment payPal = new PayPalPayment();
creditCard.processPayment(100);
creditCard.issueRefund(20);
payPal.processPayment(200);
payPal.issueRefund(50);
}
}
```

Output:



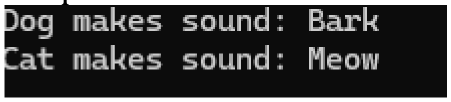
```
Processing credit card payment of $100.0
Issuing credit card refund of $20.0
Processing PayPal payment of $200.0
Issuing PayPal refund of $50.0
```

12(d)

Code:

```
interface Animal {
String makeSound();
String getType();
}
class Dog implements Animal {
public String makeSound() {
return "Bark";
}
public String getType() {
return "Dog";
}
}
class Cat implements Animal {
public String makeSound() {
return "Meow";
}
public String getType() {
return "Cat";
}
}
public class Main {
public static void main(String[] args) {
Animal dog = new Dog();
Animal cat = new Cat();
System.out.println(dog.getType() + " makes sound: " +
dog.makeSound());
System.out.println(cat.getType() + " makes sound: " + cat.makeSound());
}
}
```

Output:



```
Dog makes sound: Bark
Cat makes sound: Meow
```

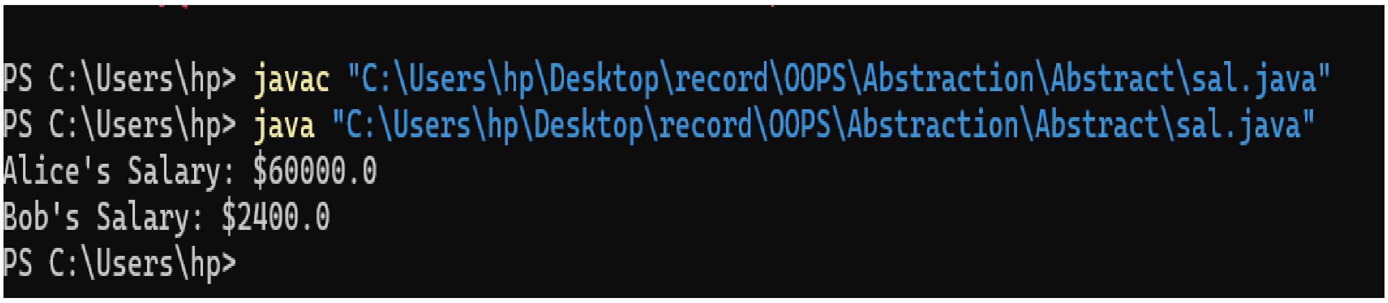
13. Abstract class programs

13a. Salary Management

Code:

```
abstract class Employee {
   String name;
   int id;

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

   abstract double calculateSalary();
}

class FullTimeEmployee extends Employee {
   double annualSalary;

   public FullTimeEmployee(String name, int id, double annualSalary) {
      super(name, id);
      this.annualSalary = annualSalary;
   }


   double calculateSalary() {
      return annualSalary;
   }
}

class PartTimeEmployee extends Employee {
   double hourlyWage;
   int hoursWorked;

   public PartTimeEmployee(String name, int id, double hourlyWage, int 
hoursWorked) {
      super(name, id);
      this.hourlyWage = hourlyWage;
      this.hoursWorked = hoursWorked;
   }


   double calculateSalary() {
        return hourlyWage * hoursWorked;
    }
}

public class sal {
    public static void main(String[] args) {
        Employee fullTime = new FullTimeEmployee("Alice", 101, 60000);
        Employee partTime = new PartTimeEmployee("Bob", 102, 20, 120);

        System.out.println(fullTime.name + "'s Salary: $" +
fullTime.calculateSalary());
        System.out.println(partTime.name + "'s Salary: $" +
partTime.calculateSalary());
    }
}
```

Output:

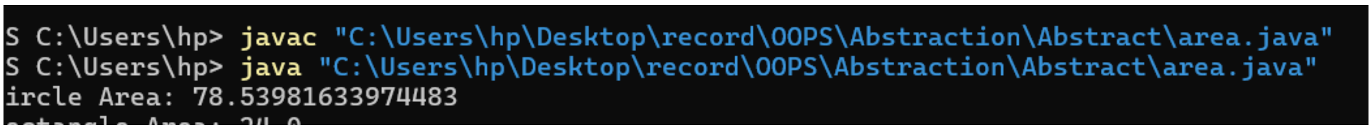


```
PS C:\Users\hp> javac "C:\Users\hp\Desktop\record\OOPS\Abstraction\Abstract\sal.java"
PS C:\Users\hp> java "C:\Users\hp\Desktop\record\OOPS\Abstraction\Abstract\sal.java"
Alice's Salary: $60000.0
Bob's Salary: $2400.0
PS C:\Users\hp>
```

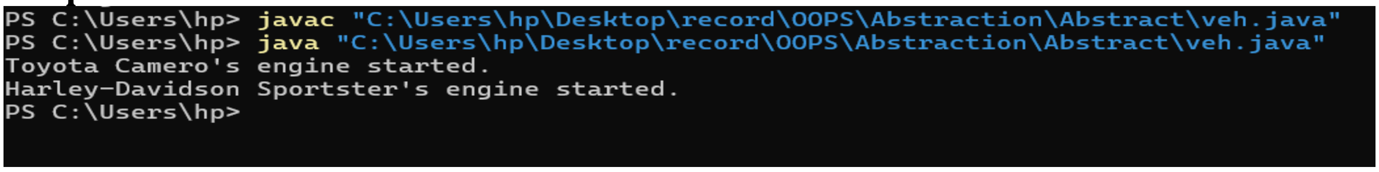
13b. Area Calculator
Code:

```
abstract class Shape {

    abstract double area();

}


class Circle extends Shape {

    double radius;


    public Circle(double radius) {

        this.radius = radius;

    }



    double area() {

        return Math.PI * radius * radius;

    }

}


class Rectangle extends Shape {

    double width, height;


    public Rectangle(double width, double height) {

        this.width = width;
        this.height = height;
    }



    double area() {
        return width * height;
    }
}

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

        System.out.println("Circle Area: " + circle.area());
        System.out.println("Rectangle Area: " + rectangle.area());
    }
}
```

Output:



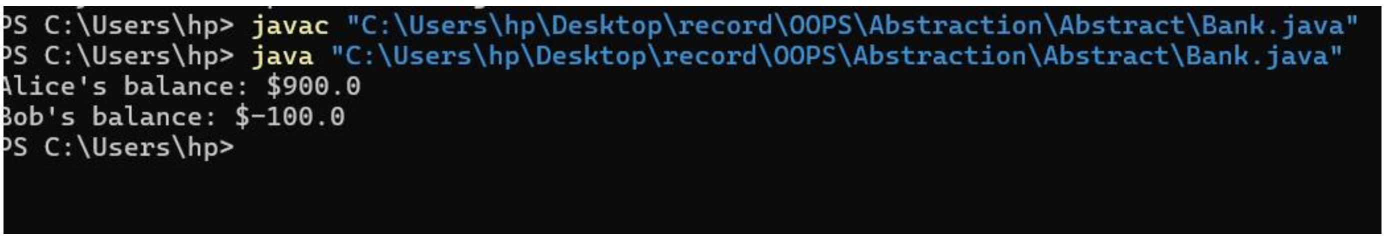
```
S C:\Users\hp> javac "C:\Users\hp\Desktop\record\OOPS\Abstraction\Abstract\area.java"
S C:\Users\hp> java "C:\Users\hp\Desktop\record\OOPS\Abstraction\Abstract\area.java"
ircle Area: 78.53981633974483
ectangle Area: 24.0
```

13c. Vehicle
Code:
```
abstract class Vehicle {
   String make, model;

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

   abstract void startEngine();
}

class Car extends Vehicle {
   public Car(String make, String model) {
      super(make, model);
   }
     void startEngine() {
      System.out.println(make + " " + model + "'s engine started.");
   }
}
class Motorcycle extends Vehicle {
   public Motorcycle(String make, String model) {
      super(make, model);
   }
      void startEngine() {
      System.out.println(make + " " + model + "'s engine started.");
   }
}
public class veh {
   public static void main(String[] args) {
      Vehicle car = new Car("Toyota", "Camero");
      Vehicle motorcycle = new Motorcycle("Harley-Davidson", "Sportster");

      car.startEngine();
      motorcycle.startEngine();
   }
}
```
Output:

```
PS C:\Users\hp> javac "C:\Users\hp\Desktop\record\OOPS\Abstraction\Abstract\veh.java"
PS C:\Users\hp> java "C:\Users\hp\Desktop\record\OOPS\Abstraction\Abstract\veh.java"
Toyota Camero's engine started.
Harley-Davidson Sportster's engine started.
PS C:\Users\hp>
```

13d. Bank Management System
Code:
```
abstract class BankAccount {
   String accountHolder;
   double balance;

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

   abstract void deposit(double amount);
   abstract void withdraw(double amount);

   void displayBalance() {
      System.out.println(accountHolder + "'s balance: $" + balance);
   }
}

class SavingsAccount extends BankAccount {
   public SavingsAccount(String accountHolder, double balance) {
      super(accountHolder, balance);
   }


   void deposit(double amount) {
      balance += amount;
   }

      void withdraw(double amount) {
      if (balance >= amount) {
         balance -= amount;
      } else {
         System.out.println("Insufficient balance in Savings Account.");
      }
   }
}

class CheckingAccount extends BankAccount {
   public CheckingAccount(String accountHolder, double balance) {
      super(accountHolder, balance);
   }
```

```
    void deposit(double amount) {
        balance += amount;
    }


    void withdraw(double amount) {
        balance -= amount;
    }
}

public class Bank {
    public static void main(String[] args) {
        BankAccount savings = new SavingsAccount("Alice", 1000);
        BankAccount checking = new CheckingAccount("Bob", 500);

        savings.deposit(200);
        savings.withdraw(300);
        savings.displayBalance();

        checking.deposit(100);
        checking.withdraw(700);
        checking.displayBalance();
    }
}
```

Output:



```
PS C:\Users\hp> javac "C:\Users\hp\Desktop\record\OOPS\Abstraction\Abstract\Bank.java"
PS C:\Users\hp> java "C:\Users\hp\Desktop\record\OOPS\Abstraction\Abstract\Bank.java"
Alice's balance: $900.0
Bob's balance: $-100.0
PS C:\Users\hp>
```

Encapsulation

14. ENCAPSULATION PROGRAMS:

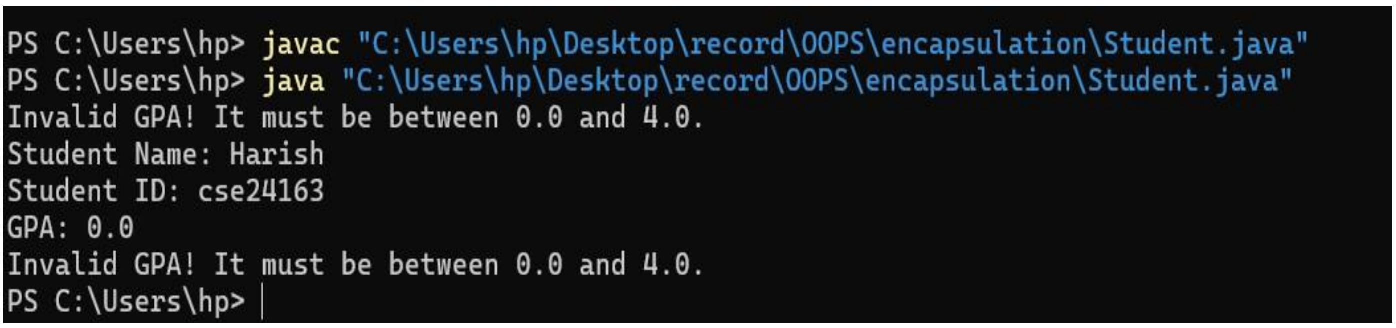
14a. Student details

Code:

```
class Student {
   private String name;
   private String id;
   private double gpa;

   public Student(String name, String id, double gpa) {
      this.name = name;
      this.id = id;
      setGpa(gpa);
   }

   public String getName() {
      return name;
   }

   public void setName(String name) {
      this.name = name;
   }

   public String getId() {
      return id;
   }

   public void setId(String id) {
      this.id = id;
   }

   public double getGpa() {
      return gpa;
   }

   public void setGpa(double gpa) {
      if (gpa >= 0.0 && gpa <= 4.0) {
         this.gpa = gpa;
      } else {
         System.out.println("Invalid GPA! It must be between 0.0 and 4.0.");
      }
   }
```

```
    public void displayStudentDetails() {
        System.out.println("Student Name: " + name);
        System.out.println("Student ID: " + id);
        System.out.println("GPA: " + gpa);
    }

    public static void main(String[] args) {
        Student student1 = new Student("Harish", "cse24163", 7.3);
        student1.displayStudentDetails();
        student1.setGpa(4.5);
    }
}
```

Output:

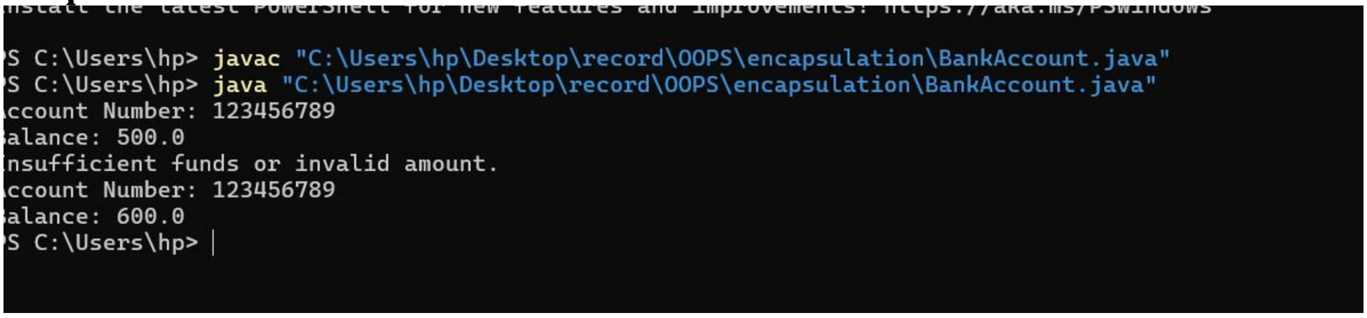


```
PS C:\Users\hp> javac "C:\Users\hp\Desktop\record\OOPS\encapsulation\Student.java"
PS C:\Users\hp> java "C:\Users\hp\Desktop\record\OOPS\encapsulation\Student.java"
Invalid GPA! It must be between 0.0 and 4.0.
Student Name: Harish
Student ID: cse24163
GPA: 0.0
Invalid GPA! It must be between 0.0 and 4.0.
PS C:\Users\hp>
```

14b. Bank Account
Code:
```
class BankAccount {
   private String accountNumber;
   private double balance;

   public BankAccount(String accountNumber, double balance) {
      this.accountNumber = accountNumber;
      this.balance = Math.max(balance, 0.0);
   }

   public String getAccountNumber() {
      return accountNumber;
   }

   public double getBalance() {
      return balance;
   }

   public void deposit(double amount) {
      if (amount > 0) {
         balance += amount;
      } else {
         System.out.println("Deposit amount must be positive.");
      }
   }

   public void withdraw(double amount) {
      if (amount > 0 && amount <= balance) {
         balance -= amount;
      } else {
         System.out.println("Insufficient funds or invalid amount.");
      }
   }

   public void displayAccountDetails() {
      System.out.println("Account Number: " + accountNumber);
      System.out.println("Balance: " + balance);
   }

   public static void main(String[] args) {
      BankAccount account = new BankAccount("123456789", 500.0);
      account.displayAccountDetails();
```

```
        account.deposit(200.0);
        account.withdraw(100.0);
        account.withdraw(700.0);
        account.displayAccountDetails();
    }
}
```

Output:



```
nstall the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

S C:\Users\hp> javac "C:\Users\hp\Desktop\record\OOPS\encapsulation\BankAccount.java"
S C:\Users\hp> java "C:\Users\hp\Desktop\record\OOPS\encapsulation\BankAccount.java"
ccount Number: 123456789
alance: 500.0
nsufficient funds or invalid amount.
ccount Number: 123456789
alance: 600.0
S C:\Users\hp> |
```

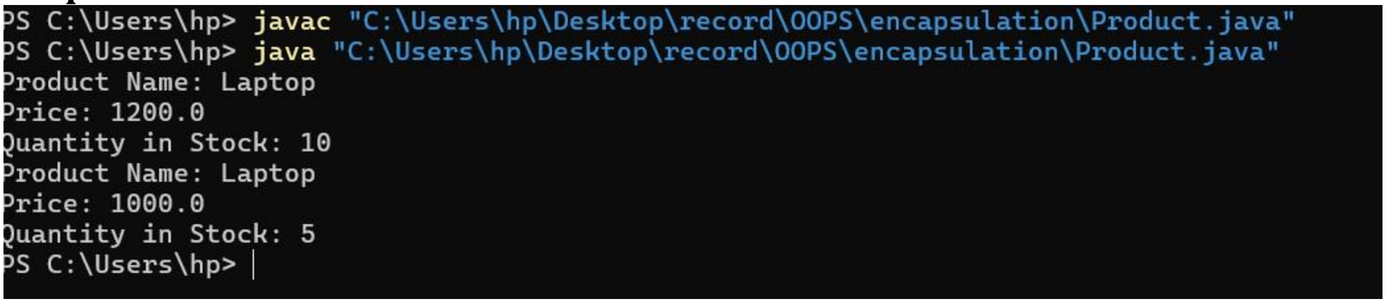
14c. Product

Code:

```
class Product {
   private String name;
   private double price;
   private int quantity;

   public Product(String name, double price, int quantity) {
      this.name = name;
      setPrice(price);
      setQuantity(quantity);
   }

   public String getName() {
      return name;
   }

   public double getPrice() {
      return price;
   }

   public void setPrice(double price) {
      if (price >= 0) {
         this.price = price;
      } else {
         System.out.println("Price cannot be negative.");
      }
   }

   public int getQuantity() {
      return quantity;
   }

   public void setQuantity(int quantity) {
      if (quantity >= 0) {
         this.quantity = quantity;
      } else {
         System.out.println("Quantity cannot be negative.");
      }
   }

   public void displayProductDetails() {
      System.out.println("Product Name: " + name);
```

```
        System.out.println("Price: " + price);
        System.out.println("Quantity in Stock: " + quantity);
    }

    public static void main(String[] args) {
        Product product = new Product("Laptop", 1200.0, 10);
        product.displayProductDetails();
        product.setPrice(1000.0);
        product.setQuantity(5);
        product.displayProductDetails();
    }
}
```

Output:

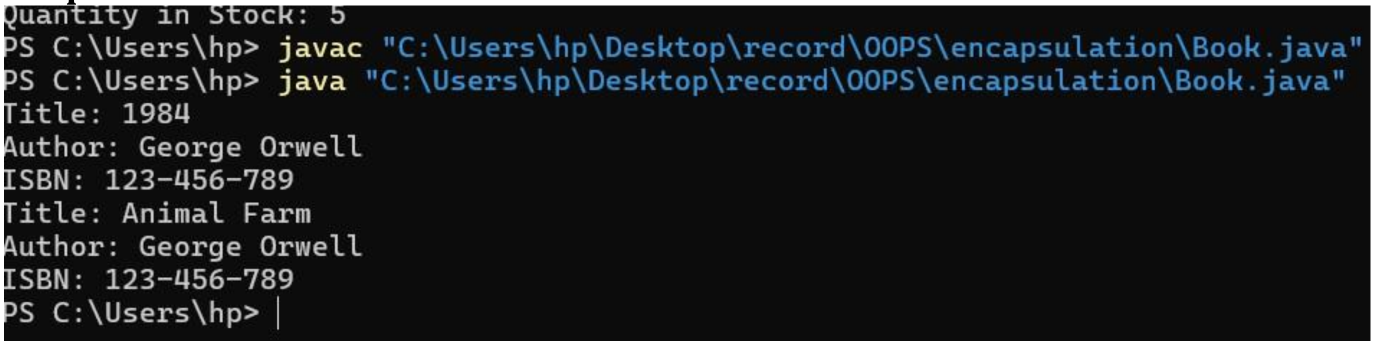


```
PS C:\Users\hp> javac "C:\Users\hp\Desktop\record\OOPS\encapsulation\Product.java"
PS C:\Users\hp> java "C:\Users\hp\Desktop\record\OOPS\encapsulation\Product.java"
Product Name: Laptop
Price: 1200.0
Quantity in Stock: 10
Product Name: Laptop
Price: 1000.0
Quantity in Stock: 5
PS C:\Users\hp>
```

17d. Library management
Code:
```
class Book {
   private String title;
   private String author;
   private String isbn;

   public Book(String title, String author, String isbn) {
      setTitle(title);
      setAuthor(author);
      this.isbn = isbn;
   }

   public String getTitle() {
      return title;
   }

   public void setTitle(String title) {
      if (!title.trim().isEmpty()) {
         this.title = title;
      } else {
         System.out.println("Title cannot be empty.");
      }
   }

   public String getAuthor() {
      return author;
   }

   public void setAuthor(String author) {
      if (!author.trim().isEmpty()) {
         this.author = author;
      } else {
         System.out.println("Author cannot be empty.");
      }
   }

   public String getIsbn() {
      return isbn;
   }

   public void displayBookDetails() {
      System.out.println("Title: " + title);
```

```
        System.out.println("Author: " + author);
        System.out.println("ISBN: " + isbn);
    }

    public static void main(String[] args) {
        Book book = new Book("1984", "George Orwell", "123-456-789");
        book.displayBookDetails();
        book.setTitle("Animal Farm");
        book.setAuthor("George Orwell");
        book.displayBookDetails();
    }
}
```

Output:



```
Quantity in Stock: 5
PS C:\Users\hp> javac "C:\Users\hp\Desktop\record\OOPS\encapsulation\Book.java"
PS C:\Users\hp> java "C:\Users\hp\Desktop\record\OOPS\encapsulation\Book.java"
Title: 1984
Author: George Orwell
ISBN: 123-456-789
Title: Animal Farm
Author: George Orwell
ISBN: 123-456-789
PS C:\Users\hp> |
```

Packages Programs

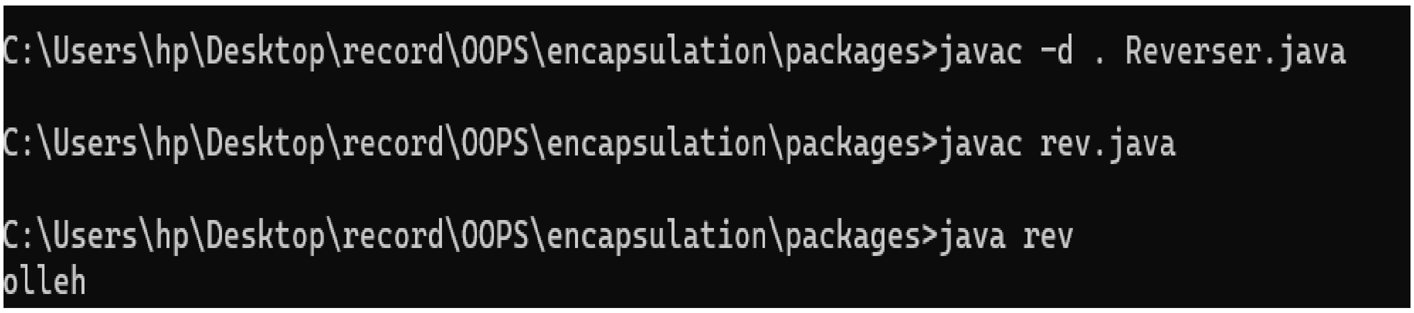
15a. Reverser
Code:
Package:

```
        package textutils;

public class Reverser {
   public String reverse(String str) {
      String result = "";
      for (int i = str.length() - 1; i >= 0; i--) {
         result += str.charAt(i);
      }
      return result;
   }
}
```

Main:

```
import textutils.Reverser;

public class rev {
   public static void main(String[] args) {
      Reverser r = new Reverser();
      System.out.println(r.reverse("hello"));
   }
}
```

Output:



```
C:\Users\hp\Desktop\record\OOPS\encapsulation\packages>javac -d . Reverser.java

C:\Users\hp\Desktop\record\OOPS\encapsulation\packages>javac rev.java

C:\Users\hp\Desktop\record\OOPS\encapsulation\packages>java rev
olleh
```

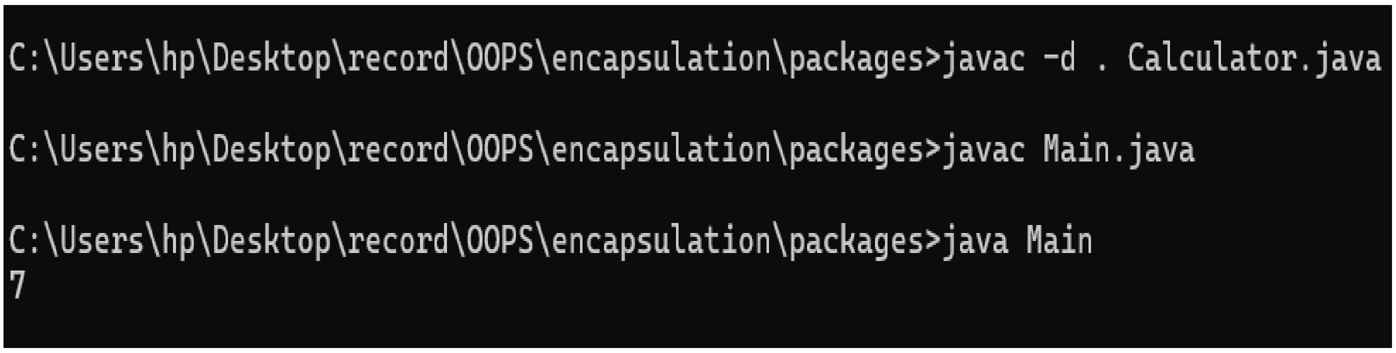
15b. Calculator
Code:
Package:

```
package mathpack;

public class Calculator
 {
   public int add(int a, int b)
{
       return a + b;

   }

}
```

Main:

```
import mathpack.Calculator;

public class Main

 {
   public static void main(String[] args)

{
       Calculator c = new Calculator();
       System.out.println(c.add(3, 4));

   }

}
```

Output:



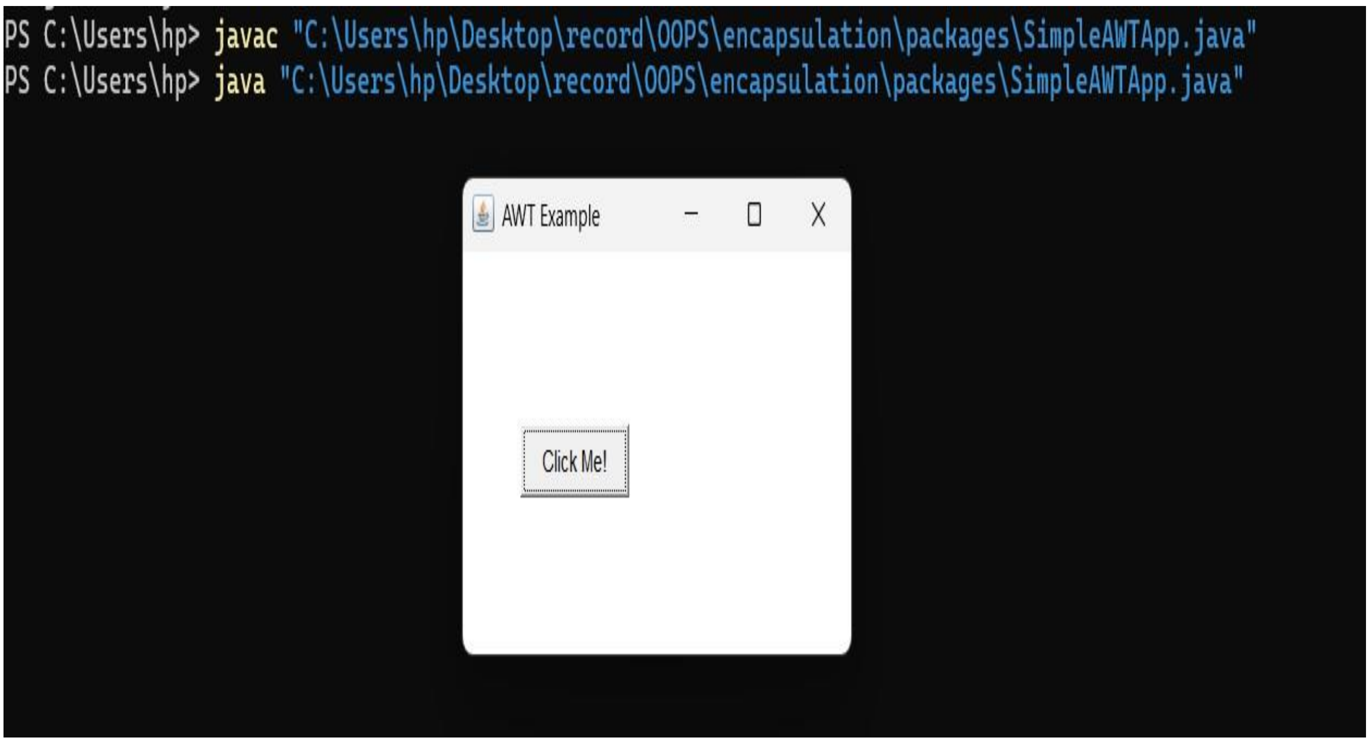
```
C:\Users\hp\Desktop\record\OOPS\encapsulation\packages>javac -d . Calculator.java

C:\Users\hp\Desktop\record\OOPS\encapsulation\packages>javac Main.java

C:\Users\hp\Desktop\record\OOPS\encapsulation\packages>java Main
7
```

15b. SimpleAWTApp

CODE:

```
import java.awt.*;
import java.awt.event.*;

 public class SimpleAWTApp {
    SimpleAWTApp() {
        Frame frame = new Frame("AWT Example");
        Button button = new Button("Click Me!");

        button.setBounds(50, 100, 80, 30);
        frame.add(button);

        frame.setSize(300, 200);
        frame.setLayout(null);
        frame.setVisible(true);

        frame.addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent e) {
                frame.dispose();
            }
        });
    }
    public static void main(String[] args) {
        new SimpleAWTApp();
    }
}
```

Output:



```
PS C:\Users\hp> javac "C:\Users\hp\Desktop\record\OOPS\encapsulation\packages\SimpleAWTApp.java"
PS C:\Users\hp> java "C:\Users\hp\Desktop\record\OOPS\encapsulation\packages\SimpleAWTApp.java"
```

15d. Shape Calculator

Code:

Package:

1. package shapes;

```
public class Circle {
   private double radius;

   public Circle(double radius) {
      this.radius = radius;
   }

   public double getArea() {
      return Math.PI * radius * radius;
   }
}
```

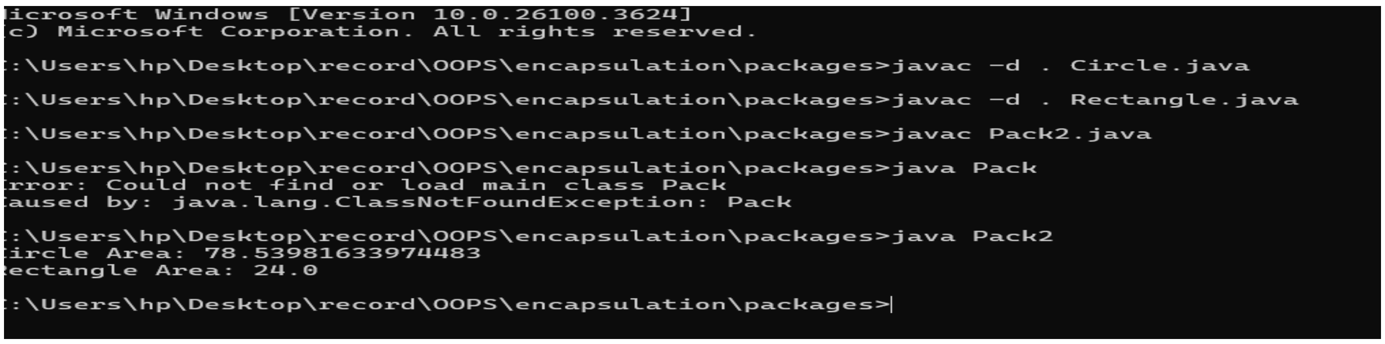
2. package shapes;

```
     public class Rectangle {
        private double length, width;

        public Rectangle(double length, double width) {
           this.length = length;
        this.width = width;
    }
    public double getArea() {
        return length * width;
    }
     }
```

Main:

```
import shapes.Circle;
import shapes.Rectangle;
public class Pack2 {
    public static void main(String[] args) {
        Circle c = new Circle(5);
        Rectangle r = new Rectangle(4, 6);
        System.out.println("Circle Area: " + c.getArea());
        System.out.println("Rectangle Area: " + r.getArea());
    }
     }
```

Output:



```
icrosoft Windows [Version 10.0.26100.3624]
c) Microsoft Corporation. All rights reserved.

:\Users\hp\Desktop\record\OOPS\encapsulation\packages>javac -d . Circle.java

:\Users\hp\Desktop\record\OOPS\encapsulation\packages>javac -d . Rectangle.java

:\Users\hp\Desktop\record\OOPS\encapsulation\packages>javac Pack2.java

:\Users\hp\Desktop\record\OOPS\encapsulation\packages>java Pack
rror: Could not find or load main class Pack
aused by: java.lang.ClassNotFoundException: Pack

:\Users\hp\Desktop\record\OOPS\encapsulation\packages>java Pack2
ircle Area: 78.53981633974483
ectangle Area: 24.0

:\Users\hp\Desktop\record\OOPS\encapsulation\packages>
```

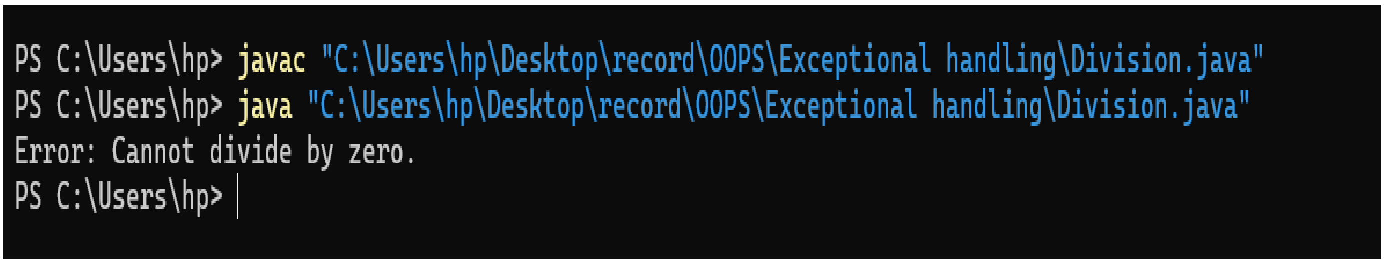
EXCEPTION HANDLING PROGRAMS

16a. Division

Code:

```
public class Division {
    public static void main(String[] args) {
        int a = 10, b = 0;
        try {
            int result = a / b;
            System.out.println("Result: " + result);
        } catch (ArithmeticException e) {
            System.out.println("Error: Cannot divide by zero.");
        }
    }
 }
```

Output:



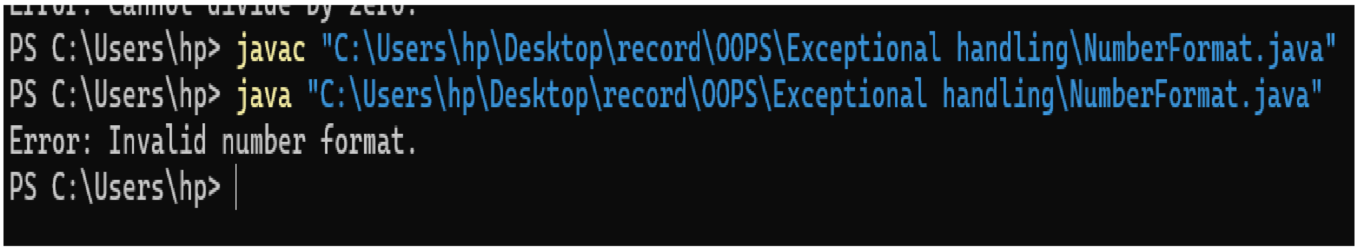
```
PS C:\Users\hp> javac "C:\Users\hp\Desktop\record\OOPS\Exceptional handling\Division.java"
PS C:\Users\hp> java "C:\Users\hp\Desktop\record\OOPS\Exceptional handling\Division.java"
Error: Cannot divide by zero.
PS C:\Users\hp>
```

16b. Number Format

Code:

```
public class NumberFormat {
    public static void main(String[] args) {
        String number = "abc";
        try {
            int num = Integer.parseInt(number);
            System.out.println("Number is: " + num);
        } catch (NumberFormatException e) {
            System.out.println("Error: Invalid number format.");
        }
    }
}
```

Output:



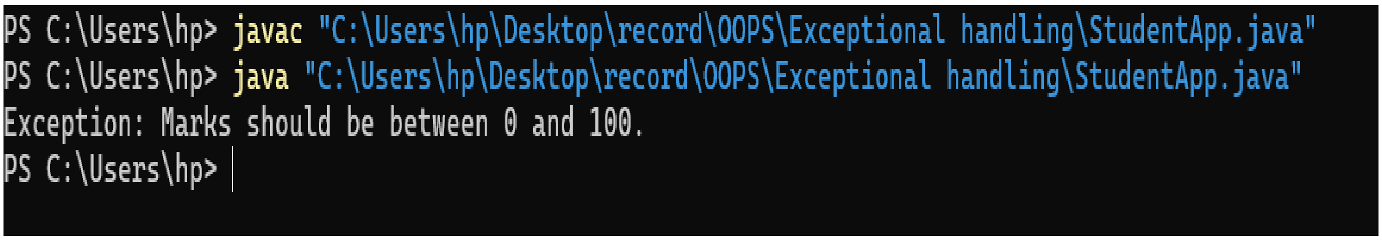
```
Error: Cannot divide by zero.
PS C:\Users\hp> javac "C:\Users\hp\Desktop\record\OOPS\Exceptional handling\NumberFormat.java"
PS C:\Users\hp> java "C:\Users\hp\Desktop\record\OOPS\Exceptional handling\NumberFormat.java"
Error: Invalid number format.
PS C:\Users\hp>
```

14c. Student

Code:

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

class Student {
    public void setMark(int mark) throws InvalidMarkException {
        if (mark < 0 || mark > 100) {
            throw new InvalidMarkException("Marks should be between 0 and 100.");
        }
        System.out.println("Valid mark: " + mark);
    }
}

public class StudentApp {
    public static void main(String[] args) {
        Student student = new Student();
        try {
            student.setMark(105);
        } catch (InvalidMarkException e) {
            System.out.println("Exception: " + e.getMessage());
        }
    }
}
```

Output:



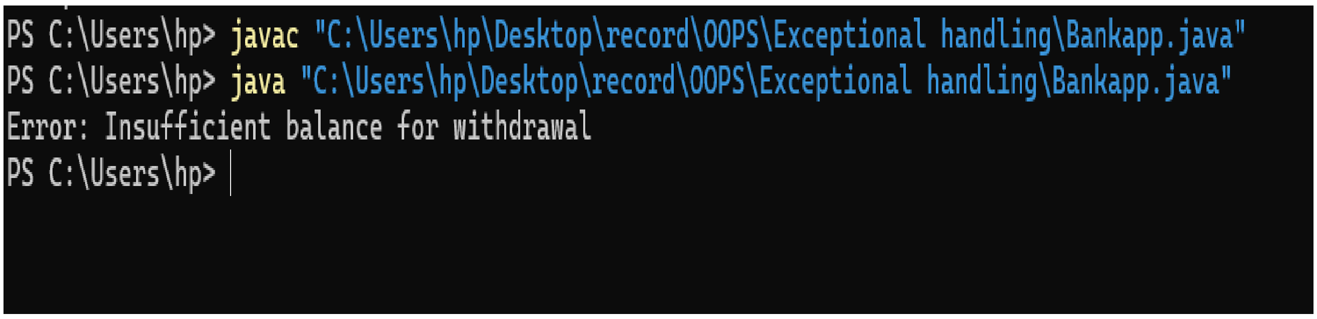
```
PS C:\Users\hp> javac "C:\Users\hp\Desktop\record\OOPS\Exceptional handling\StudentApp.java"
PS C:\Users\hp> java "C:\Users\hp\Desktop\record\OOPS\Exceptional handling\StudentApp.java"
Exception: Marks should be between 0 and 100.
PS C:\Users\hp>
```

14d. BankApp

Code:

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

class BankAccount {
    private double balance = 1000;

    public void withdraw(double amount) throws InsufficientFundsException {
        if (amount > balance) {
            throw new InsufficientFundsException("Insufficient balance for withdrawal");
        }
        balance -= amount;
        System.out.println("Withdrawal successful. Remaining balance: " + balance);
    }
}

public class Bankapp {
    public static void main(String[] args) {
        BankAccount account = new BankAccount();
        try {
            account.withdraw(1500);
        } catch (InsufficientFundsException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

Output:



```
PS C:\Users\hp> javac "C:\Users\hp\Desktop\record\OOPS\Exceptional handling\Bankapp.java"
PS C:\Users\hp> java "C:\Users\hp\Desktop\record\OOPS\Exceptional handling\Bankapp.java"
Error: Insufficient balance for withdrawal
PS C:\Users\hp>
```

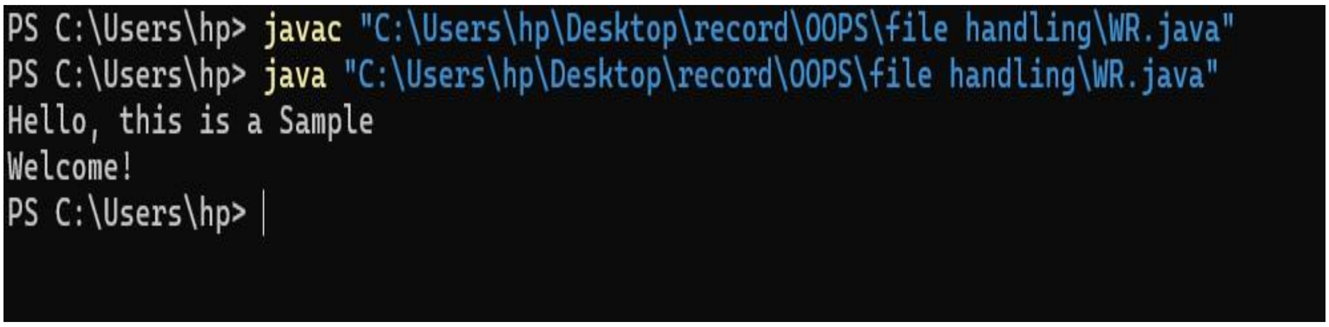
FILE HANDLING PROGRAMS

17a. Write and Read

Code:

```
import java.io.*;


public class WR {
    public static void main(String[] args) {
        String fileName = "sample.txt";


        try {
            FileWriter writer = new FileWriter(fileName);
            writer.write("Hello, this is a Sample\nWelcome!");
            writer.close();


            FileReader reader = new FileReader(fileName);
            int ch;
            while ((ch = reader.read()) != -1) {
                System.out.print((char) ch);
            }
            reader.close();


        } catch (IOException e) {
            System.out.println("File error: " + e.getMessage());
        }
    }
}
```

Output:



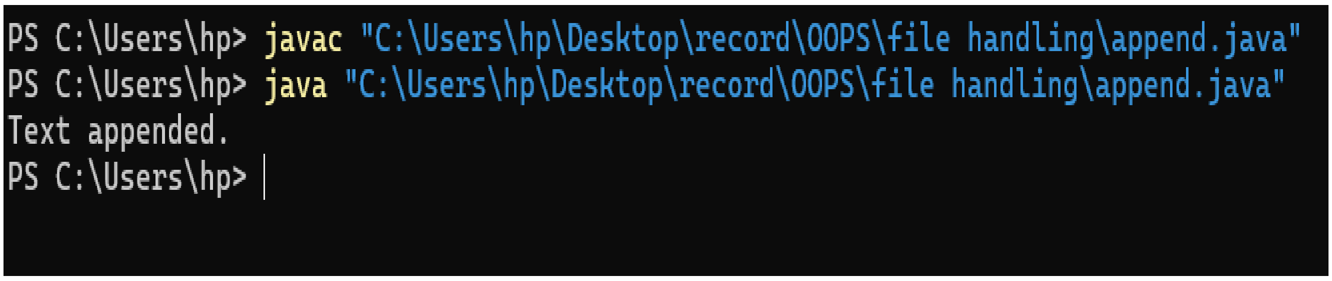
```
PS C:\Users\hp> javac "C:\Users\hp\Desktop\record\OOPS\file handling\WR.java"
PS C:\Users\hp> java "C:\Users\hp\Desktop\record\OOPS\file handling\WR.java"
Hello, this is a Sample
Welcome!
PS C:\Users\hp>
```

17b. Append a text

Code:

```java
import java.io.FileWriter;

import java.io.IOException;


public class append {

    public static void main(String[] args) {

        try {

            FileWriter writer = new FileWriter("easy.txt", true);

            writer.write("\nAppended line.");

            writer.close();

            System.out.println("Text appended.");

        } catch (IOException e) {

            System.out.println("Error: " + e.getMessage());

        }

    }

}
```

Output:



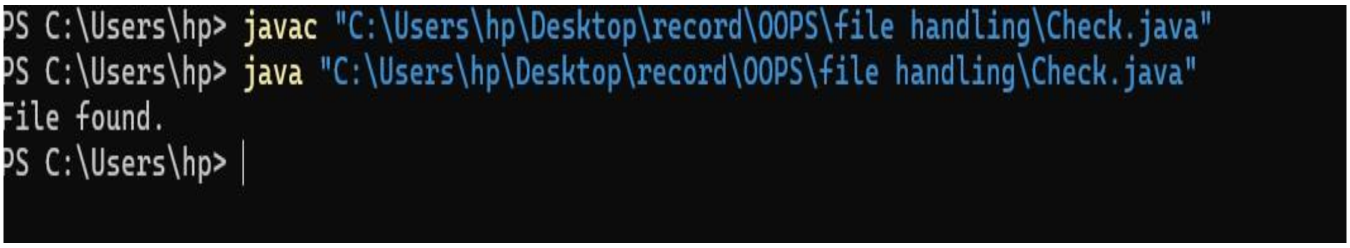
```
PS C:\Users\hp> javac "C:\Users\hp\Desktop\record\OOPS\file handling\append.java"
PS C:\Users\hp> java "C:\Users\hp\Desktop\record\OOPS\file handling\append.java"
Text appended.
PS C:\Users\hp>
```

17c. Checking the presence of a file

Code:

```
import java.io.File;


public class Check {
    public static void main(String[] args) {
        File file = new File("easy.txt");
        if (file.exists()) {
            System.out.println("File found.");
        } else {
            System.out.println("File not found.");
        }
    }
}
```

Output:



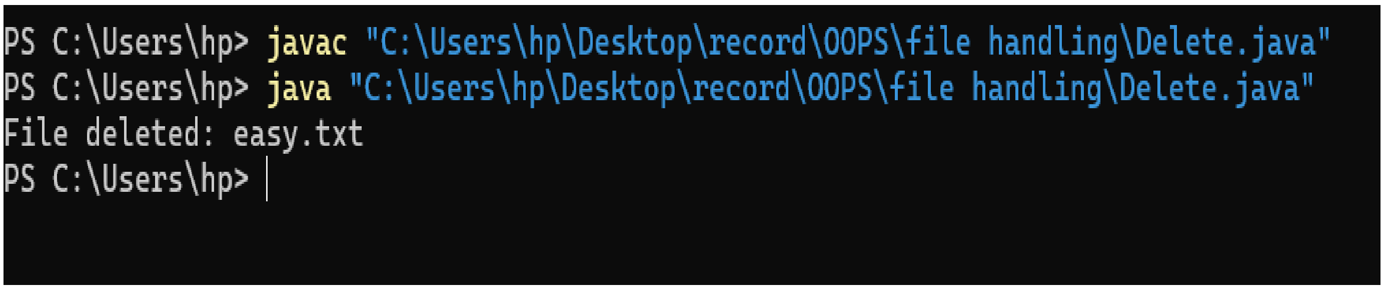
```
PS C:\Users\hp> javac "C:\Users\hp\Desktop\record\OOPS\file handling\Check.java"
PS C:\Users\hp> java "C:\Users\hp\Desktop\record\OOPS\file handling\Check.java"
File found.
PS C:\Users\hp> |
```

17d. Delete a file

Code:

```
import java.io.File;


public class Delete {
    public static void main(String[] args) {
        File file = new File("easy.txt");
        if (file.delete()) {
            System.out.println("File deleted: " + file.getName());
        } else {
            System.out.println("File not found or couldn't be deleted.");
        }
    }
}
```

Output:



```
PS C:\Users\hp> javac "C:\Users\hp\Desktop\record\OOPS\file handling\Delete.java"
PS C:\Users\hp> java "C:\Users\hp\Desktop\record\OOPS\file handling\Delete.java"
File deleted: easy.txt
PS C:\Users\hp> 
```