LAB RECORD

23CSE111- Object Oriented Programming

Submitted by

CH.SC.U4CSE24160 -Mahalakshmi K

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.U4CSE24160 – Mahalakshmi K*** 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

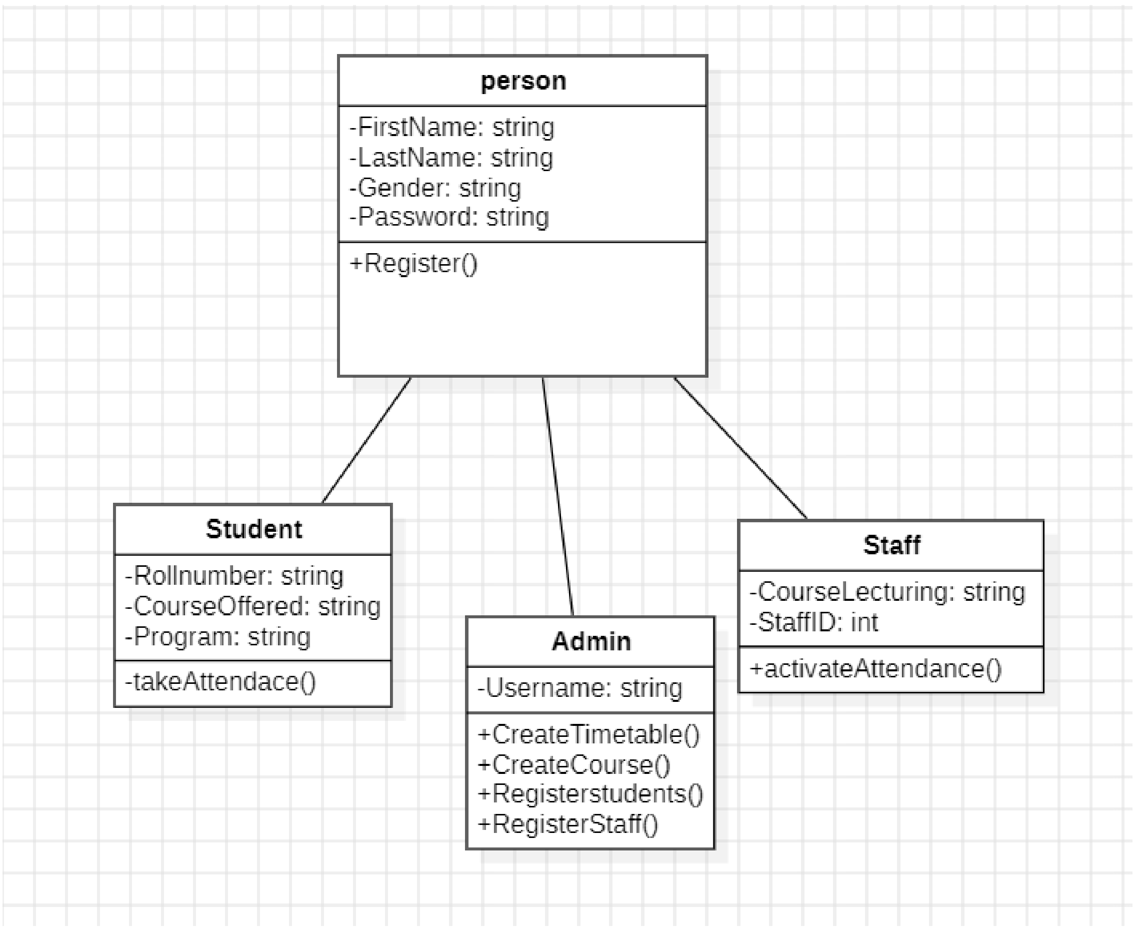
INDEX

EXPERIMENT-1

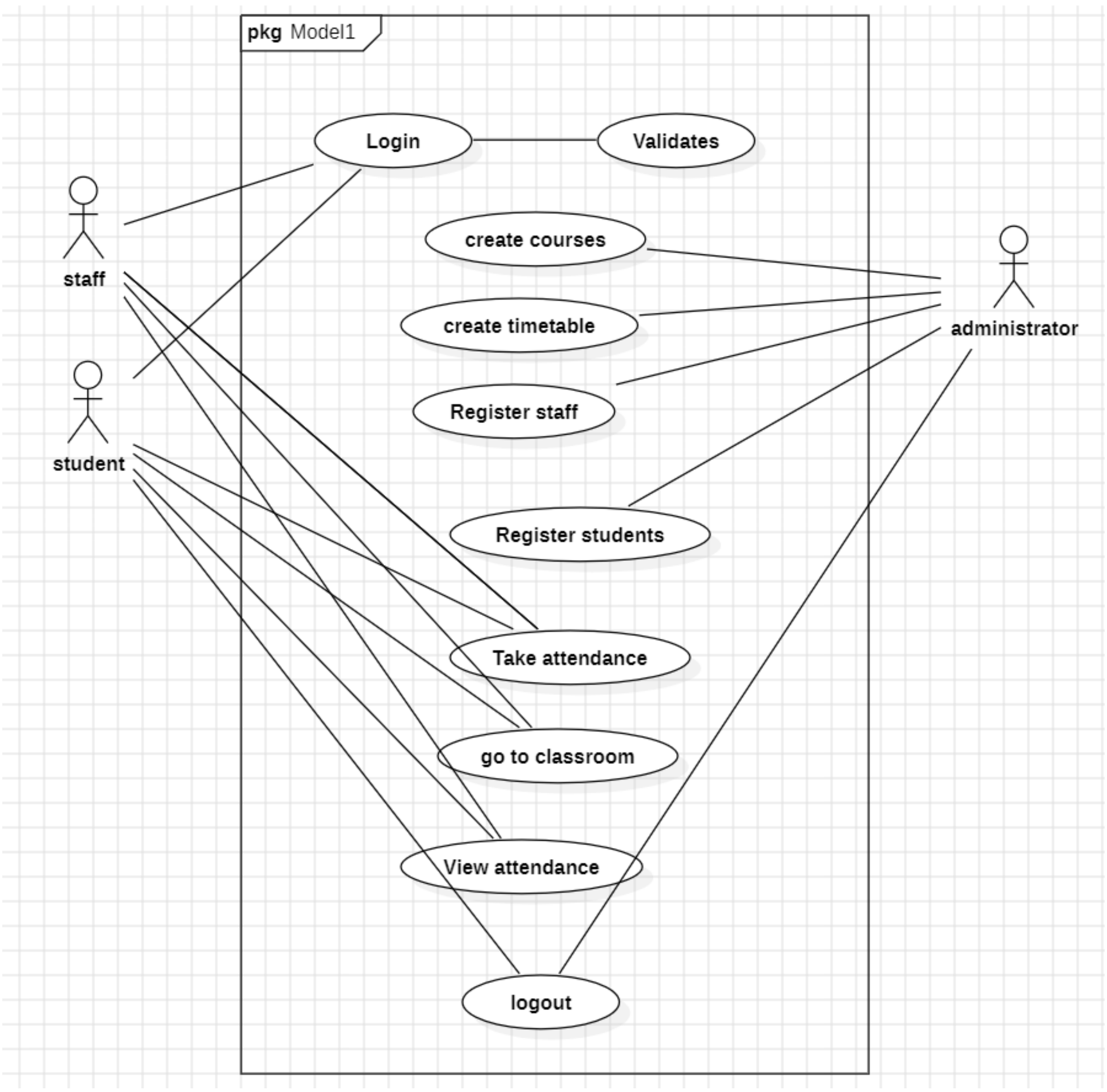
UML DIAGRAMS

1. Online attendance system:

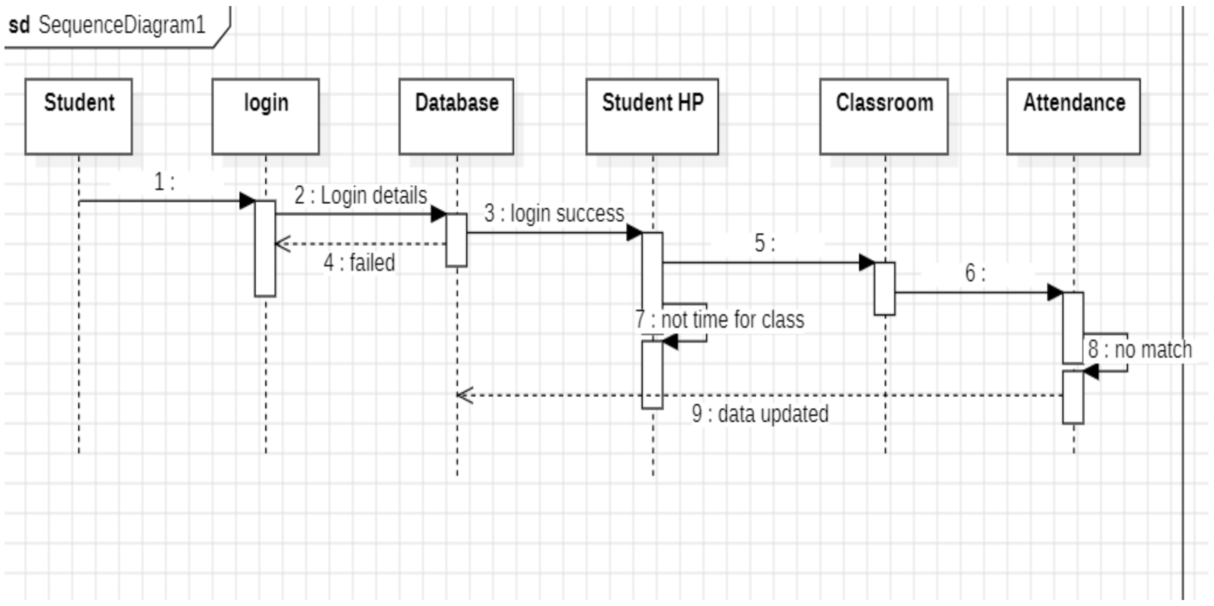
a)Class diagram



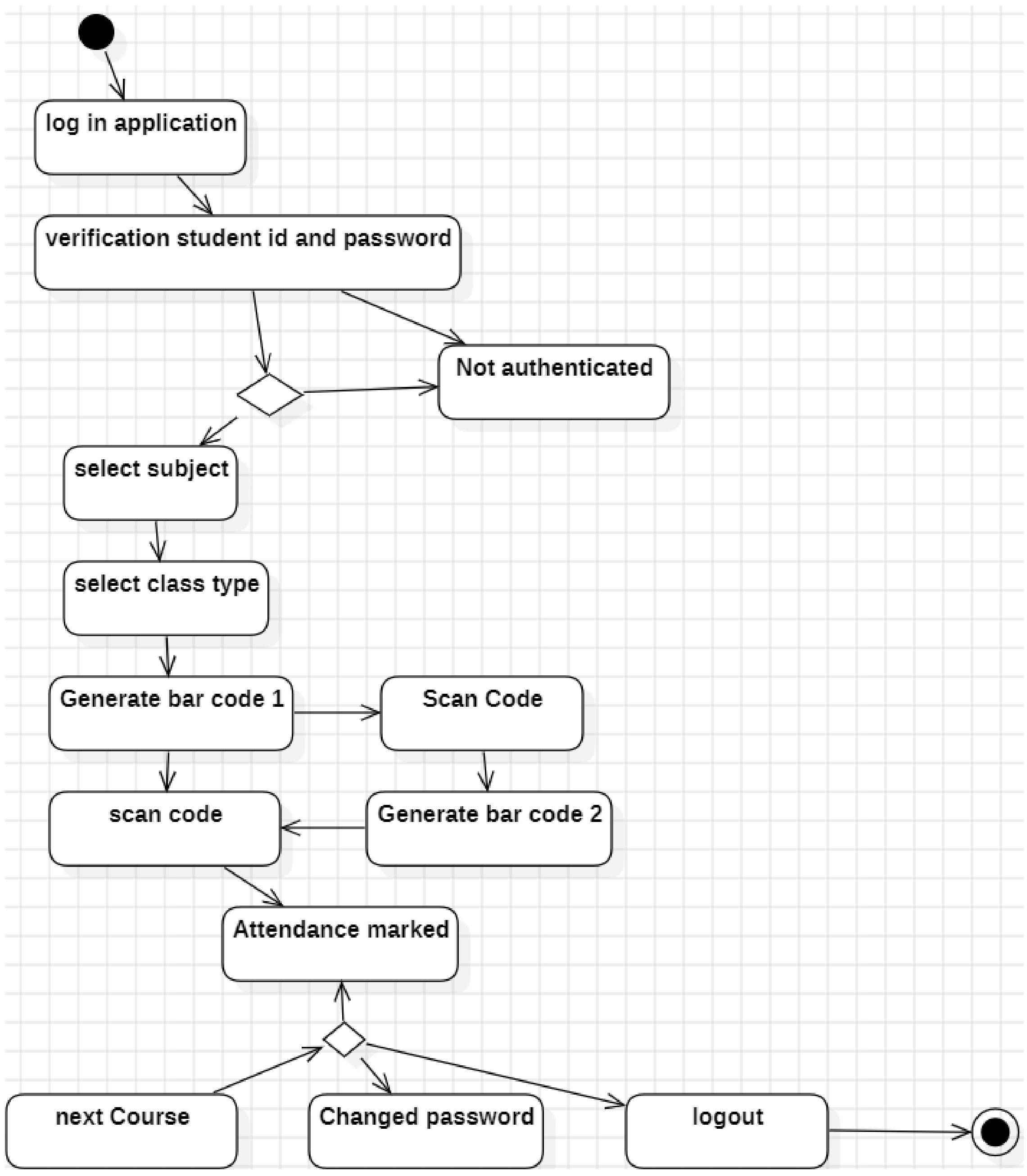
b)Use-Case Diagram



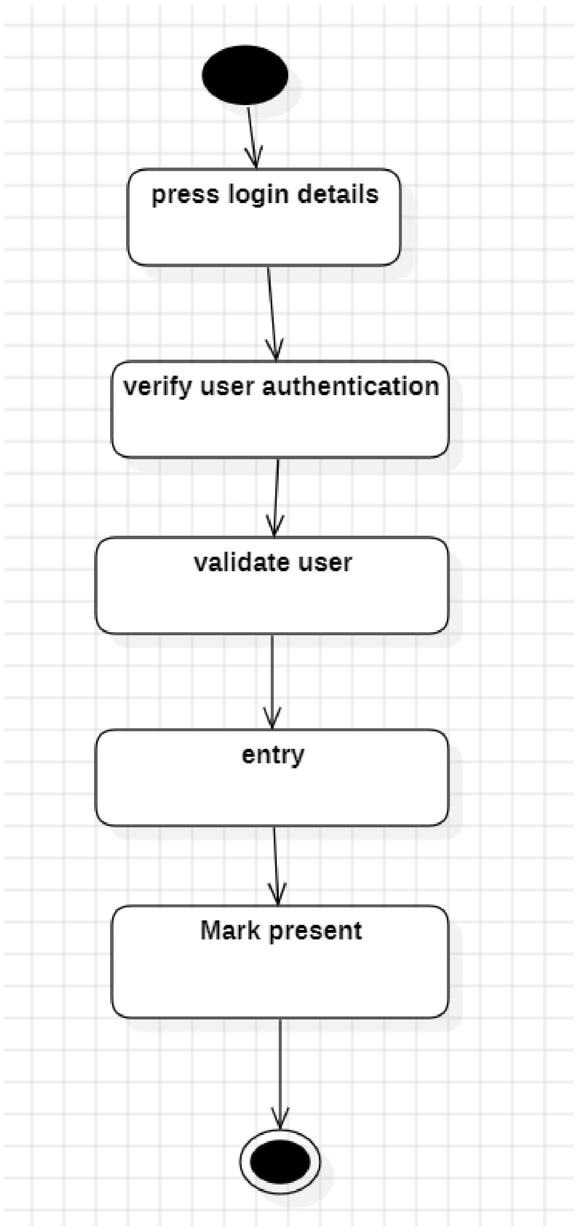
c) Sequence Diagram



d)Activity Diagram



e) State Diagram

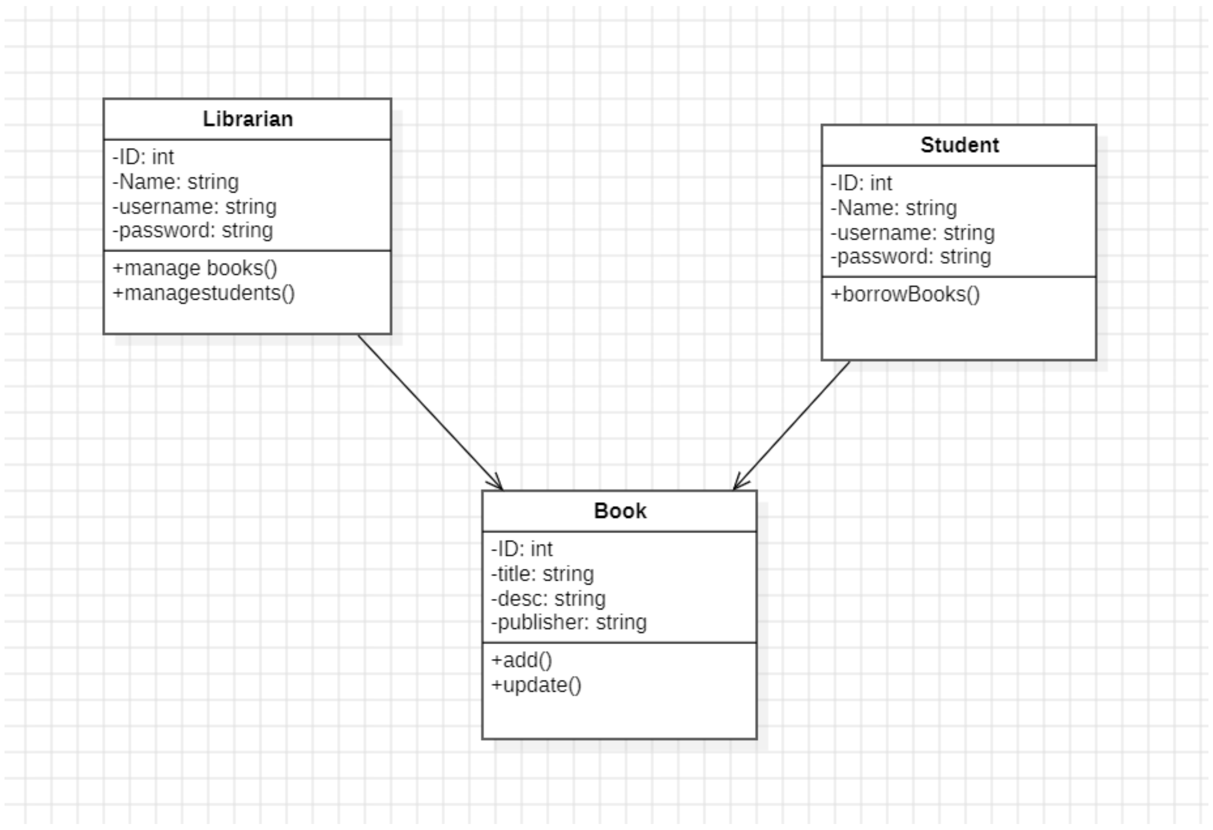

press login details
verify user authentication
validate user
entry
Mark present

EXPERIMENT-2

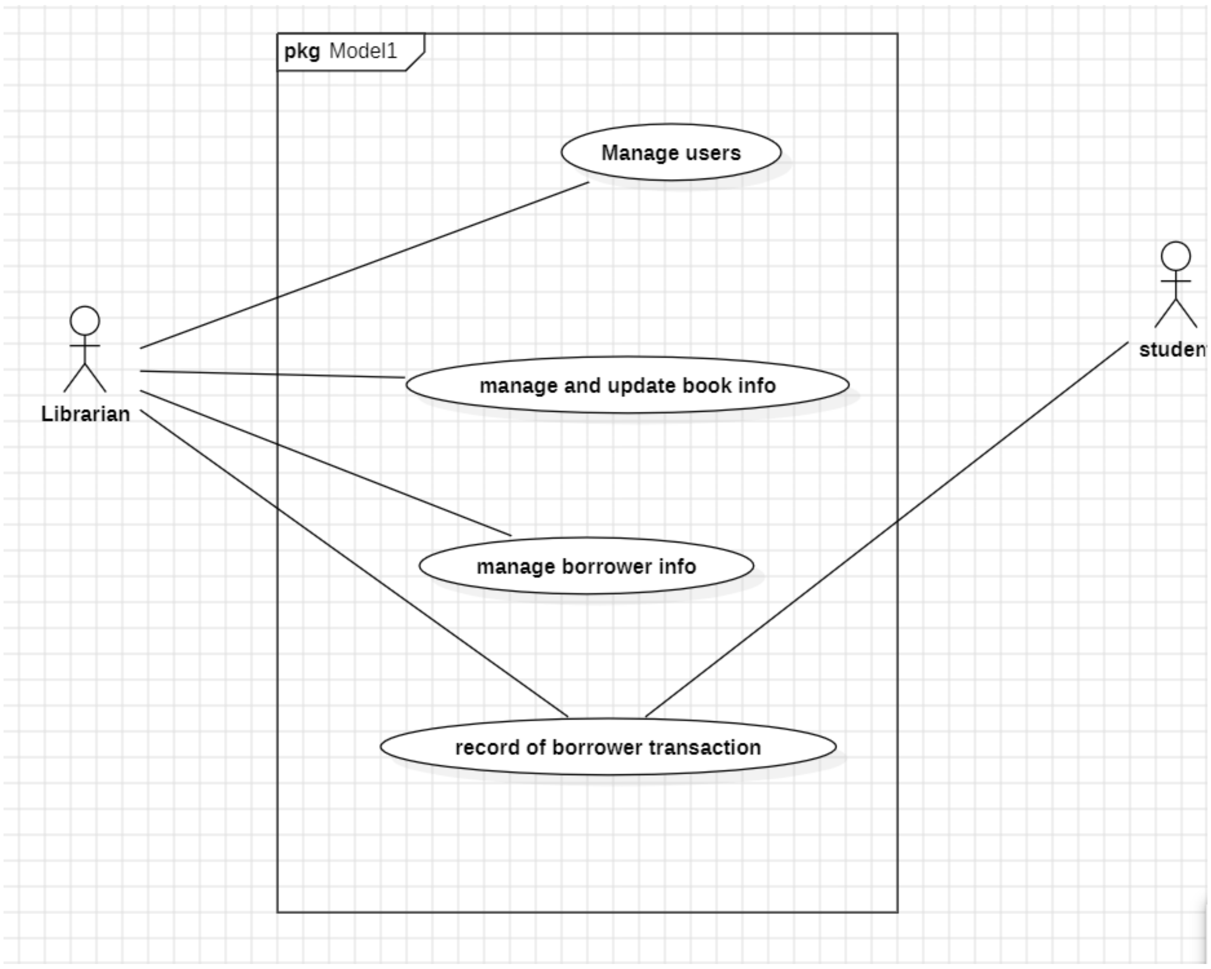
UML DIAGRAMS

2. Library Management

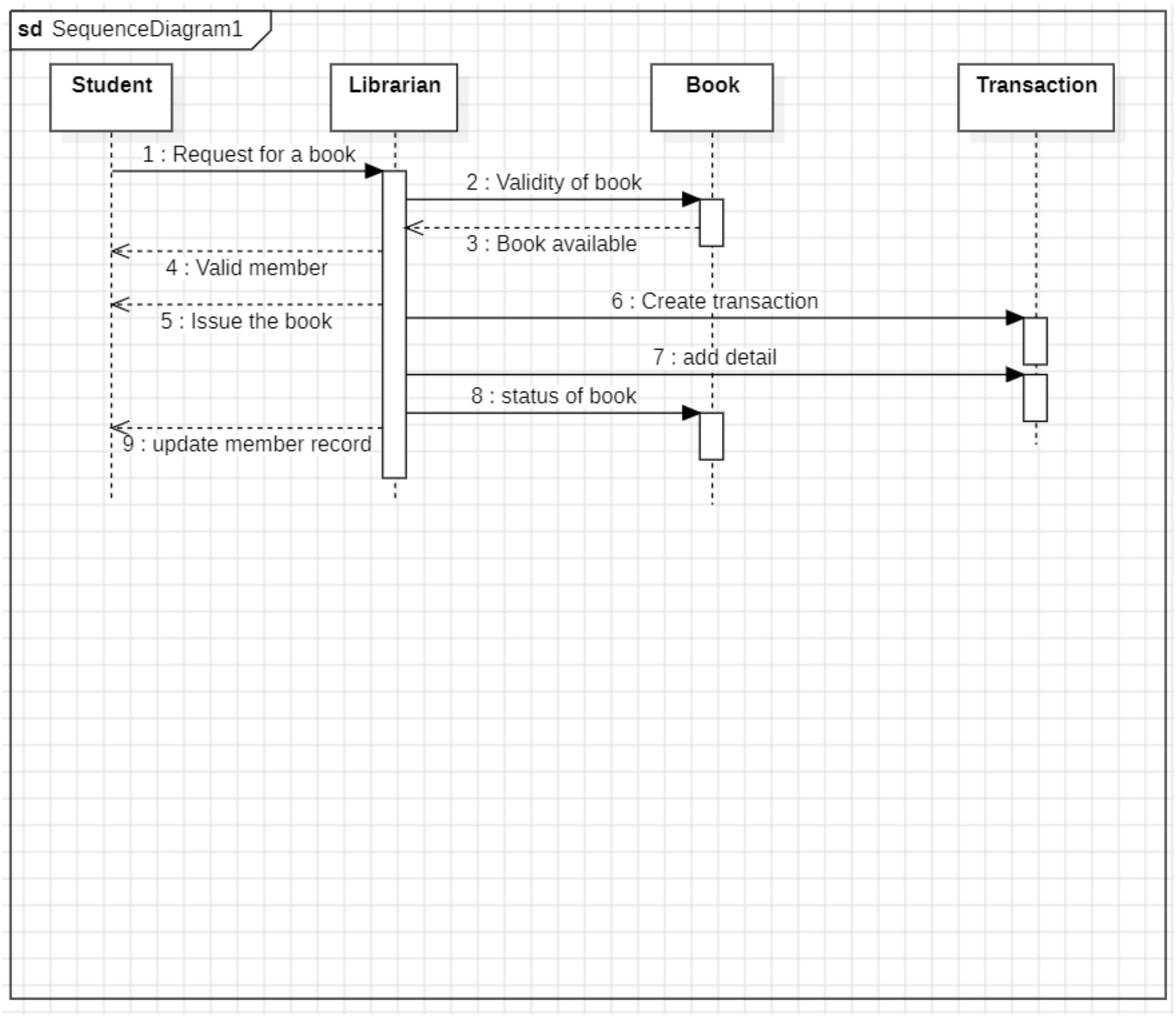
a)Class Diagram



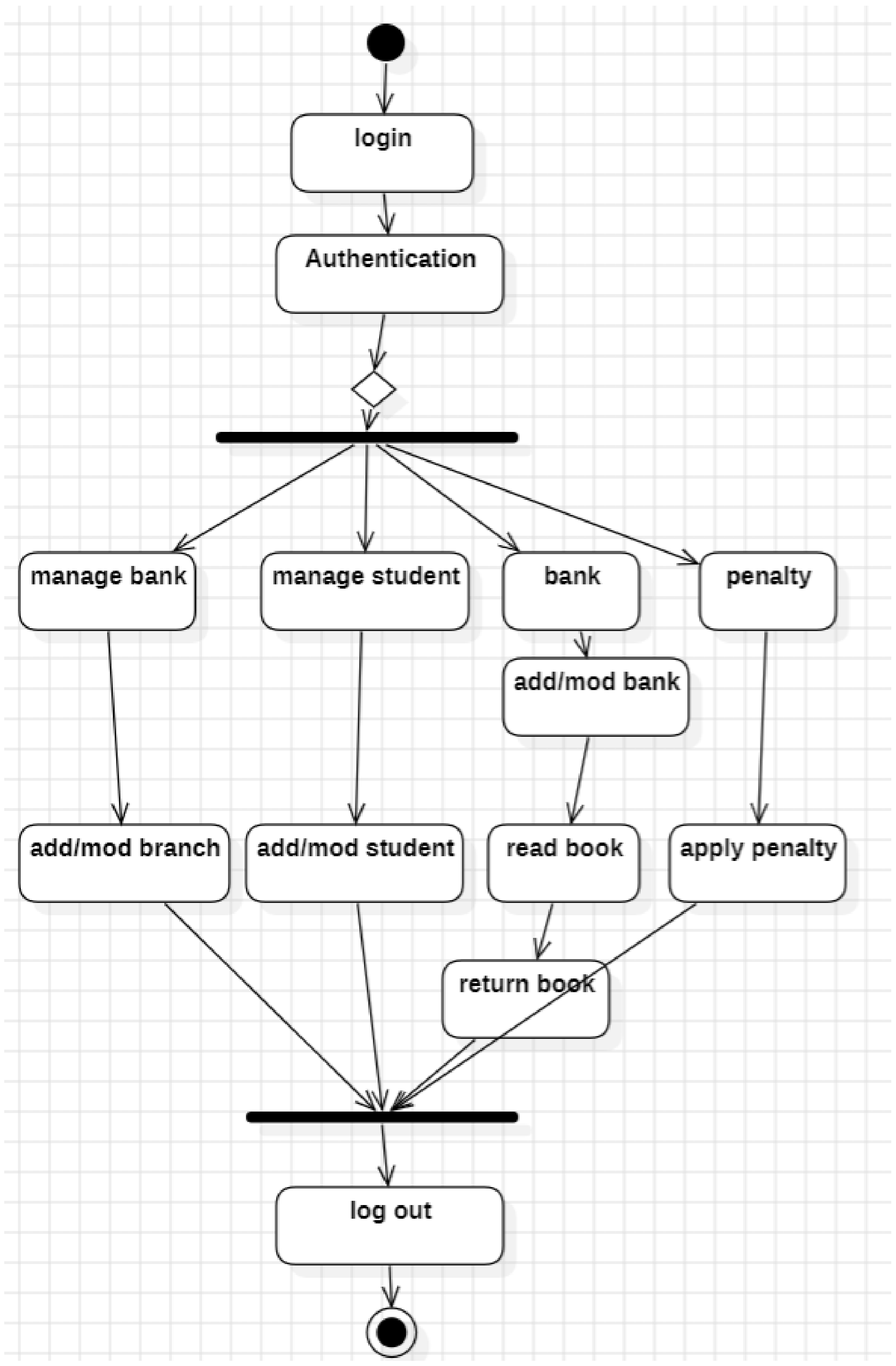
b)Use-Case Diagram



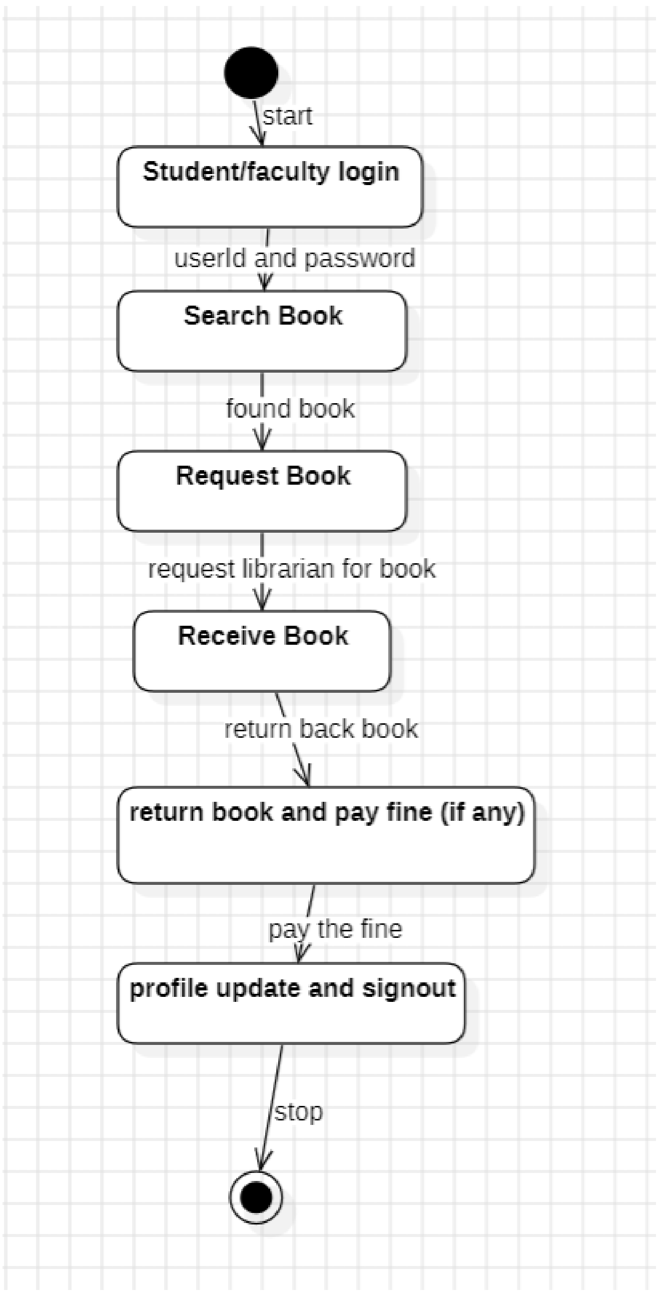
c) Sequence Diagram



d)Activity Diagram



e)State Diagram


start
Student/faculty login
userId and password
Search Book
found book
Request Book
request librarian for book
Receive Book
return back book
return book and pay fine (if any)
pay the fine
profile update and signout
stop

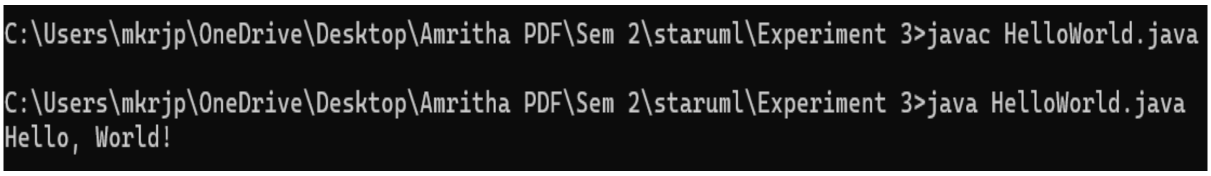
EXPERIMENT-3

1. Hello World (Basic Program)

Java Code :

```
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

OUTPUT :



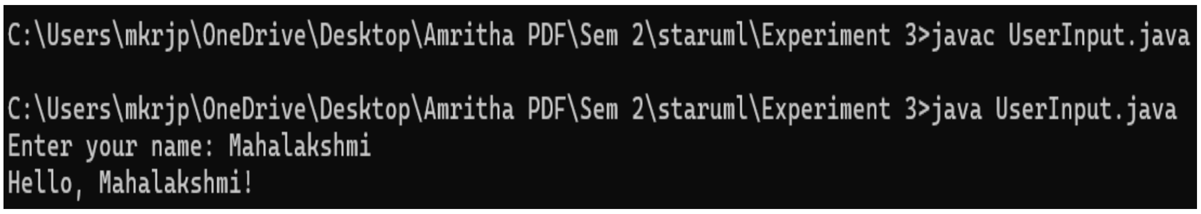
```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 3>javac HelloWorld.java

C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 3>java HelloWorld.java
Hello, World!
```

2. Taking User Input

Java Code :

```
import java.util.Scanner;

public class UserInput {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter your name: ");
        String name = scanner.nextLine();
        System.out.println("Hello, " + name + "!");
        scanner.close();
    }
}
```

OUTPUT :



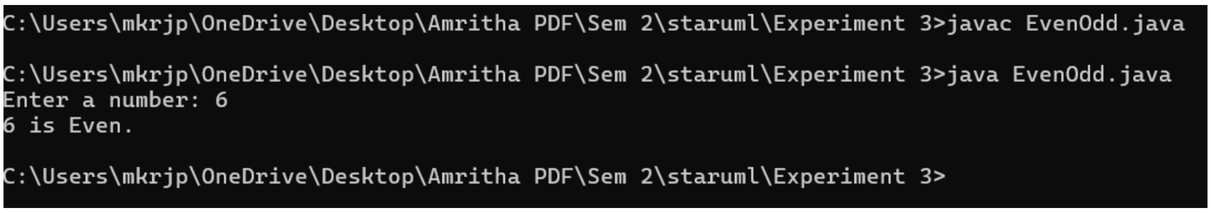
```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 3>javac UserInput.java

C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 3>java UserInput.java
Enter your name: Mahalakshmi
Hello, Mahalakshmi!
```

3. Check Even or Odd (Conditional Statement)

Java 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(num + " is Even.");
        } else {
            System.out.println(num + " is Odd.");
        }


        scanner.close();
    }
}
```

OUTPUT :



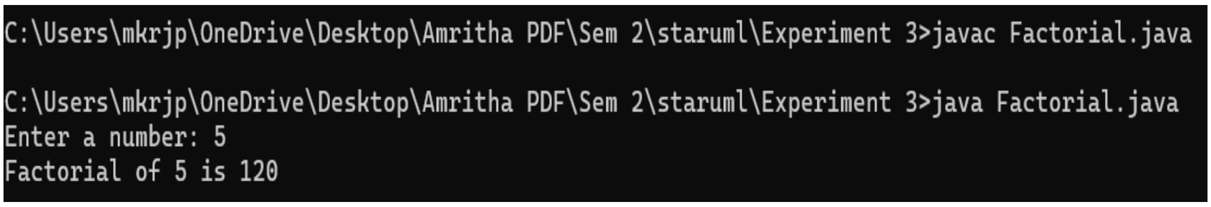
```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 3>javac EvenOdd.java

C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 3>java EvenOdd.java
Enter a number: 6
6 is Even.

C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 3>
```

4. Factorial Using Loop

Java 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();
        int fact = 1;


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


        System.out.println("Factorial of " + num + " is " + fact);
        scanner.close();
    }
}
```

Output :



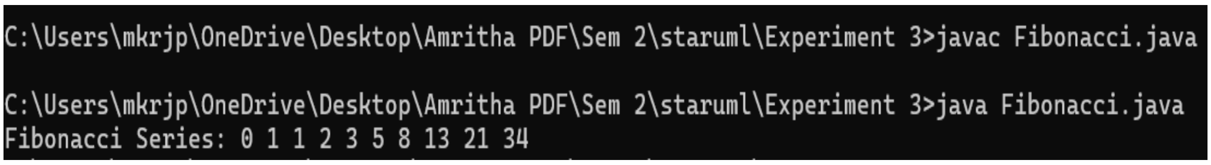
```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 3>javac Factorial.java

C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 3>java Factorial.java
Enter a number: 5
Factorial of 5 is 120
```

5. Fibonacci Series

<u>Java Code</u> :

```
public class Fibonacci {
   public static void main(String[] args) {
      int n = 10, first = 0, second = 1;

      System.out.print("Fibonacci Series: " + first + " " + second);

      for (int i = 2; i < n; i++) {
         int next = first + second;
         System.out.print(" " + next);
         first = second;
         second = next;
      }
   }
}
```

<u>Output</u> :



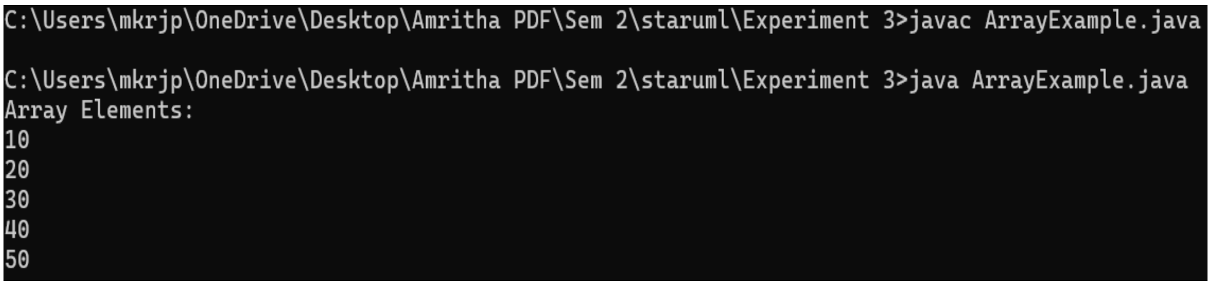
```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 3>javac Fibonacci.java

C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 3>java Fibonacci.java
Fibonacci Series: 0 1 1 2 3 5 8 13 21 34
```

6. Array Example (Printing Elements)

<u>Java Code</u> :

```
public class ArrayExample {
    public static void main(String[] args) {
        int[] numbers = {10, 20, 30, 40, 50};

        System.out.println("Array Elements:");
        for (int num : numbers) {
            System.out.println(num);
        }
    }
}
```

Output :



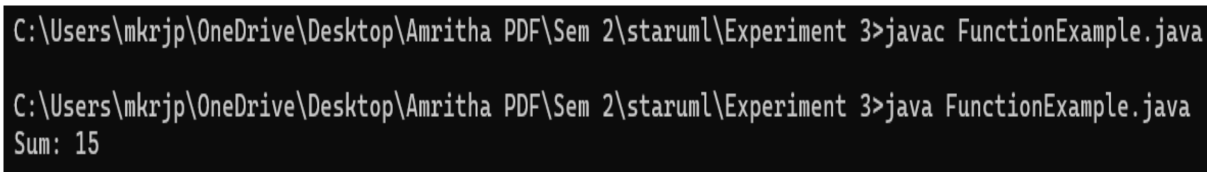
```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 3>javac ArrayExample.java

C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 3>java ArrayExample.java
Array Elements:
10
20
30
40
50
```

7. Function to Find Sum

   Java Code :

```
public class FunctionExample {
    public static void main(String[] args) {
        int result = add(5, 10);
        System.out.println("Sum: " + result);
    }

    public static int add(int a, int b) {
        return a + b;
    }
}
```

   Output :



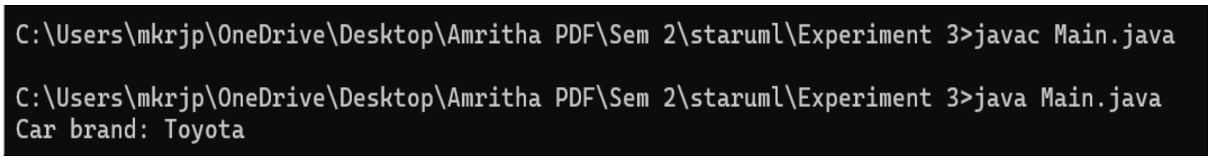
```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 3>javac FunctionExample.java

C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 3>java FunctionExample.java
Sum: 15
```

8. Class and Object Example

   Java Code :

```
class Car {
   String brand;

   public Car(String brand) {
      this.brand = brand;
   }

   public void showBrand() {
      System.out.println("Car brand: " + brand);
   }
}

public class Main {
   public static void main(String[] args) {
      Car myCar = new Car("Toyota");
      myCar.showBrand();
   }
}
```

   Output :



```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 3>javac Main.java

C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 3>java Main.java
Car brand: Toyota
```

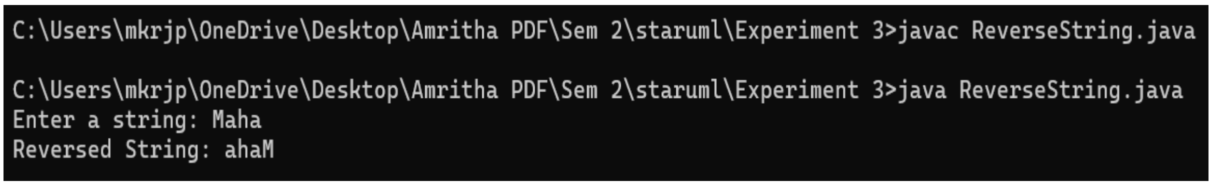
9. Reverse a String

   Java 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 = "";


        for (int i = str.length() - 1; i >= 0; i--) {
            reversed += str.charAt(i);
        }


        System.out.println("Reversed String: " + reversed);
        scanner.close();
    }
}
```

Output :



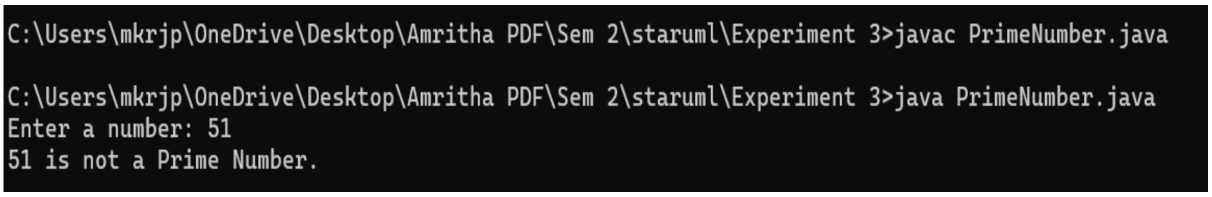
```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 3>javac ReverseString.java

C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 3>java ReverseString.java
Enter a string: Maha
Reversed String: ahaM
```

10. Check if a Number is Prime

Java 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 = true;

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

        if (isPrime) {
            System.out.println(num + " is a Prime Number.");
        } else {
            System.out.println(num + " is not a Prime Number.");
        }

        scanner.close();
    }
}
```

Output :



```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 3>javac PrimeNumber.java

C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 3>java PrimeNumber.java
Enter a number: 51
51 is not a Prime Number.
```

EXPERIMENT-4

INHERITANCE

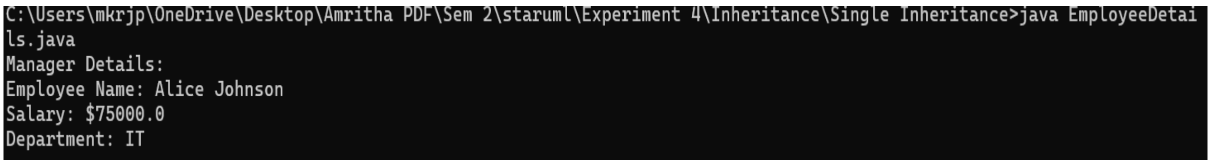
1.Single Inheritance

a)Employee Details

Code:

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

    void setDetails(String name, double salary) {
        this.name = name;
        this.salary = salary;
    }

    void showDetails() {
        System.out.println("Employee Name: " + name);
        System.out.println("Salary: $" + salary);
    }
}



class Manager extends Employee {
    String department;

    void setDepartment(String department) {
        this.department = department;
    }
    void showManagerDetails() {
        showDetails(); // Call parent class method
        System.out.println("Department: " + department);
    }
}



public class EmployeeDetails {
    public static void main(String[] args) {
        Manager m = new Manager();
        m.setDetails("Alice Johnson", 75000);
        m.setDepartment("IT");

        System.out.println("Manager Details:");
        m.showManagerDetails();
    }
}
```

Output:



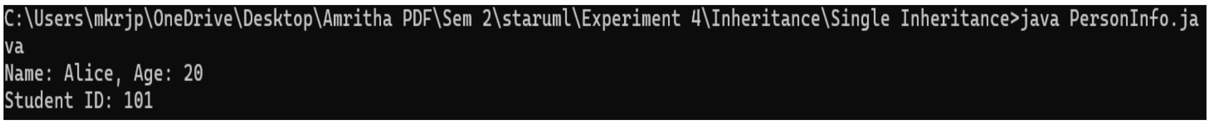
```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 4\Inheritance\Single Inheritance>java EmployeeDetai
ls.java
Manager Details:
Employee Name: Alice Johnson
Salary: $75000.0
Department: IT
```

b) Person Information

Code:

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

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

class Student extends Person {
    int studentId;

    void showStudentInfo() {
        System.out.println("Student ID: " + studentId);
    }
}

public class PersonInfo {
    public static void main(String[] args) {
        Student s = new Student();
        s.name = "Alice";
        s.age = 20;
        s.studentId = 101;
        s.display();
        s.showStudentInfo();
    }
}
```

Output:



```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 4\Inheritance\Single Inheritance>java PersonInfo.ja
va
Name: Alice, Age: 20
Student ID: 101
```

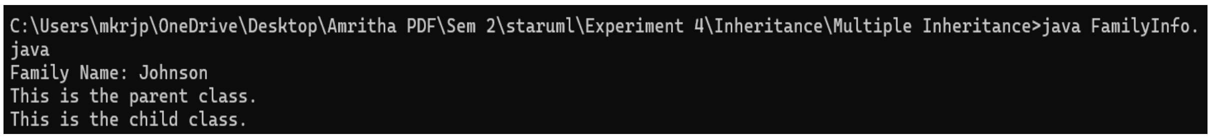
2.Multiple Inheritance

a) Family Information

Code:

```
class GrandParent {
    void familyName() {
        System.out.println("Family Name: Johnson");
    }
}


class Parent extends GrandParent {
    void displayParentInfo() {
        System.out.println("This is the parent class.");
    }
}


class Child extends Parent {
    void showChildInfo() {
        System.out.println("This is the child class.");
    }
}


public class FamilyInfo {
    public static void main(String[] args) {
        Child obj = new Child();
        obj.familyName();       // From GrandParent
        obj.displayParentInfo(); // From Parent
        obj.showChildInfo();    // From Child
    }
}
```

Output:



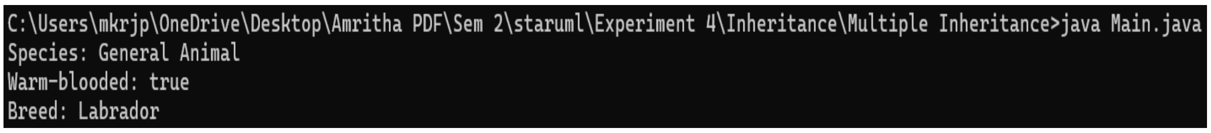
```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 4\Inheritance\Multiple Inheritance>java FamilyInfo.
java
Family Name: Johnson
This is the parent class.
This is the child class.
```

b)Animal type

Code:

```
class Animal {
    String species = "General Animal";
}

class Mammal extends Animal {
    boolean isWarmBlooded = true;
}

class Dog extends Mammal {
    String breed = "Labrador";

    void showDetails() {
        System.out.println("Species: " + species);
        System.out.println("Warm-blooded: " + isWarmBlooded);
        System.out.println("Breed: " + breed);
    }
}

public class Main {
    public static void main(String[] args) {
        Dog d = new Dog();
        d.showDetails();
    }
}
```

Output:



```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 4\Inheritance\Multiple Inheritance>java Main.java
Species: General Animal
Warm-blooded: true
Breed: Labrador
```

3. Hierarchical Inheritance

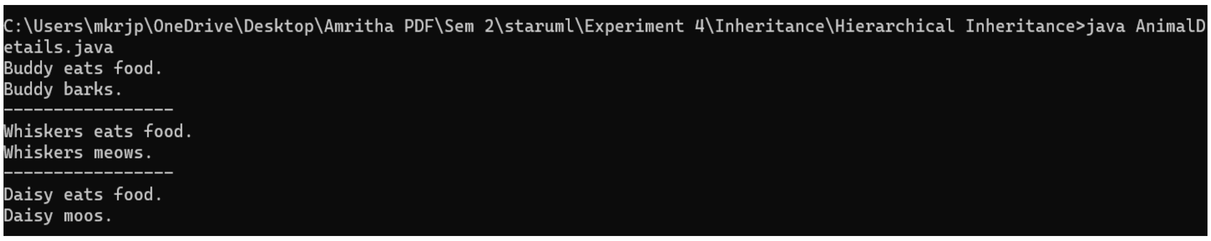
a) Animal Details

Code:

```
class Animal {
    String name;

    // Constructor
    Animal(String name) {
        this.name = name;
    }

    void eat() {
        System.out.println(name + " eats food.");
    }
}



class Dog extends Animal {
    Dog(String name) {
        super(name); // Calling parent constructor
    }
    void bark() {
        System.out.println(name + " barks.");
    }
}


class Cat extends Animal {
    Cat(String name) {
        super(name);
    }

    void meow() {
        System.out.println(name + " meows.");
    }
}


class Cow extends Animal {
    Cow(String name) {
        super(name);
    }

    void moo() {
        System.out.println(name + " moos.");
    }
}
```

```
public class AnimalDetails {
    public static void main(String[] args) {
        Dog myDog = new Dog("Buddy");
        myDog.eat();  // Inherited from Animal
        myDog.bark(); // Specific to Dog

        System.out.println("-----------------");

        Cat myCat = new Cat("Whiskers");
        myCat.eat();  // Inherited from Animal
        myCat.meow(); // Specific to Cat

        System.out.println("-----------------");

        Cow myCow = new Cow("Daisy");
        myCow.eat();  // Inherited from Animal
        myCow.moo();  // Specific to Cow
    }
}
```

Output:



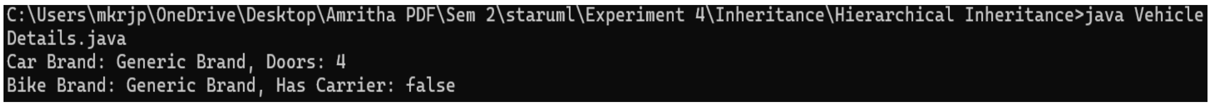
```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 4\Inheritance\Hierarchical Inheritance>java AnimalD
etails.java
Buddy eats food.
Buddy barks.
-----------------
Whiskers eats food.
Whiskers meows.
-----------------
Daisy eats food.
Daisy moos.
```

b)Vehicle Details

Code:

```
class Vehicle {
    String brand = "Generic Brand";
}

class Car extends Vehicle {
    int doors = 4;

    void showCarDetails() {
        System.out.println("Car Brand: " + brand + ", Doors: " + doors);
    }
}

class Bike extends Vehicle {
    boolean hasCarrier = false;

    void showBikeDetails() {
        System.out.println("Bike Brand: " + brand + ", Has Carrier: " + hasCarrier);
    }
}

public class VehicleDetails {
    public static void main(String[] args) {
        Car c = new Car();
        c.showCarDetails();

        Bike b = new Bike();
        b.showBikeDetails();
    }
}
```

Output:



```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 4\Inheritance\Hierarchical Inheritance>java Vehicle
Details.java
Car Brand: Generic Brand, Doors: 4
Bike Brand: Generic Brand, Has Carrier: false
```

4.Hybrid Inheritance

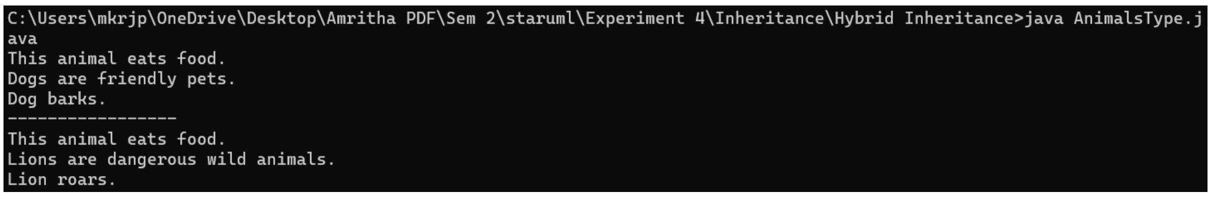
a)Animals Type

Code:

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



interface Pet {
    void friendly();
}

interface Wild {
    void dangerous();
}



class Dog extends Animal implements Pet {
    public void friendly() {
        System.out.println("Dogs are friendly pets.");
    }

    void bark() {
        System.out.println("Dog barks.");
```

```
  }
}


class Lion extends Animal implements Wild {
  public void dangerous() {
    System.out.println("Lions are dangerous wild animals.");
  }

  void roar() {
    System.out.println("Lion roars.");
  }
}


public class AnimalsType {
  public static void main(String[] args) {
    Dog myDog = new Dog();
    myDog.eat();      // Inherited from Animal
    myDog.friendly(); // Implemented from Pet
    myDog.bark();     // Specific to Dog

    System.out.println("-----------------");

    Lion myLion = new Lion();
    myLion.eat();     // Inherited from Animal
    myLion.dangerous();// Implemented from Wild
    myLion.roar();    // Specific to Lion
    }
}
```

Output:



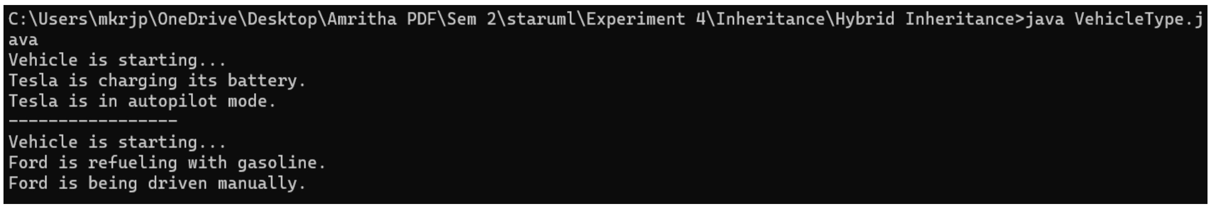
```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 4\Inheritance\Hybrid Inheritance>java AnimalsType.j
ava
This animal eats food.
Dogs are friendly pets.
Dog barks.
-----------------
This animal eats food.
Lions are dangerous wild animals.
Lion roars.
```

b) Vehicles Type

Code:

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



interface ElectricVehicle {
    void chargeBattery();
}



interface FuelVehicle {
    void refuel();
}



class Tesla extends Vehicle implements ElectricVehicle {
    public void chargeBattery() {
        System.out.println("Tesla is charging its battery.");
    }

    void autopilot() {
        System.out.println("Tesla is in autopilot mode.");
    }
}


class Ford extends Vehicle implements FuelVehicle {
    public void refuel() {
        System.out.println("Ford is refueling with gasoline.");
    }

    void manualDrive() {
        System.out.println("Ford is being driven manually.");
    }
}

public class VehicleType {
    public static void main(String[] args) {
        Tesla myTesla = new Tesla();
        myTesla.start();         // Inherited from Vehicle
        myTesla.chargeBattery(); // Implemented from ElectricVehicle
        myTesla.autopilot();     // Specific to Tesla

        System.out.println("-----------------");

        Ford myFord = new Ford();
        myFord.start();   // Inherited from Vehicle
        myFord.refuel();  // Implemented from FuelVehicle
        myFord.manualDrive();  // Specific to Ford
    }
}
```

Output:



```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 4\Inheritance\Hybrid Inheritance>java VehicleType.j
ava
Vehicle is starting...
Tesla is charging its battery.
Tesla is in autopilot mode.
-----------------
Vehicle is starting...
Ford is refueling with gasoline.
Ford is being driven manually.
```

EXPERIMENT-4

POLYMORPHISM

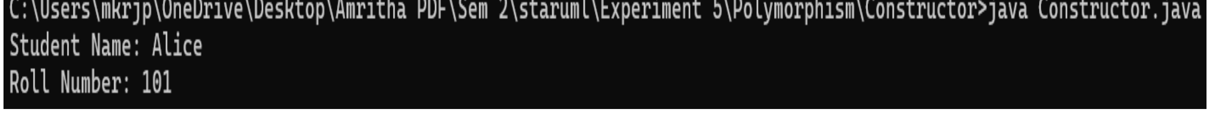
1. Constructor

a) Student enrolment

Code:

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


   Student(String studentName, int studentRollNo) {
      name = studentName;
      rollNo = studentRollNo;
   }

   void display() {
      System.out.println("Student Name: " + name);
      System.out.println("Roll Number: " + rollNo);
   }

   public static void main(String[] args) {
      Student s1 = new Student("Alice", 101);
      s1.display();
   }
}
```

Output:



```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 5\Polymorphism\Constructor>java Constructor.java
Student Name: Alice
Roll Number: 101
```

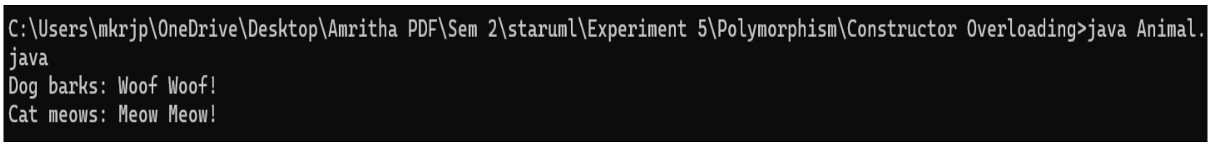
2.Constructor Overloading

a) Animal info

Code:

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

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

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

public class Animal {
  public static void main(String[] args) {
    Example d = new Dog();
    Example c = new Cat();

    d.makeSound();
    c.makeSound();
  }
}
```

Output:



```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 5\Polymorphism\Constructor Overloading>java Animal.
java
Dog barks: Woof Woof!
Cat meows: Meow Meow!
```

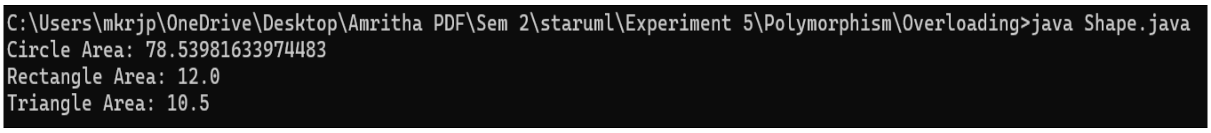
3.Overloading

a)Shape

Code:

```
class Shape {
  double area(double radius) {  // Circle
    return Math.PI * radius * radius;
  }

  double area(double length, double width) {  // Rectangle
    return length * width;
  }

  double area(int base, int height) {  // Triangle
    return 0.5 * base * height;
  }

  public static void main(String[] args) {
    Shape shape = new Shape();
    System.out.println("Circle Area: " + shape.area(5));
    System.out.println("Rectangle Area: " + shape.area(4, 6));
    System.out.println("Triangle Area: " + shape.area(3, 7));
  }
}
```

Output:



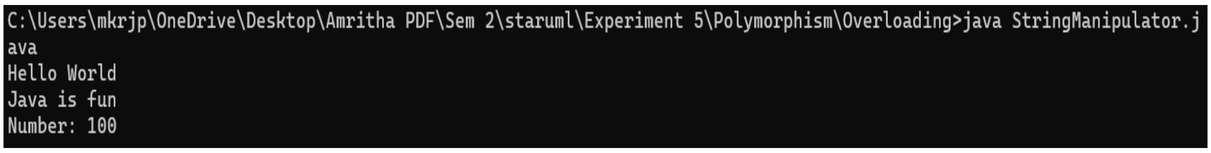
```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 5\Polymorphism\Overloading>java Shape.java
Circle Area: 78.53981633974483
Rectangle Area: 12.0
Triangle Area: 10.5
```

b) String Manipulator

Code:

```
class StringManipulator {
    String concat(String a, String b) {
        return a + b;
    }

    String concat(String a, String b, String c) {
        return a + b + c;
    }

    String concat(String a, int num) {
        return a + num;
    }

    public static void main(String[] args) {
        StringManipulator sm = new StringManipulator();
        System.out.println(sm.concat("Hello", " World"));
        System.out.println(sm.concat("Java", " is", " fun"));
        System.out.println(sm.concat("Number: ", 100));
    }
}
```

Output:



```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 5\Polymorphism\Overloading>java StringManipulator.j
ava
Hello World
Java is fun
Number: 100
```

4.Overriding

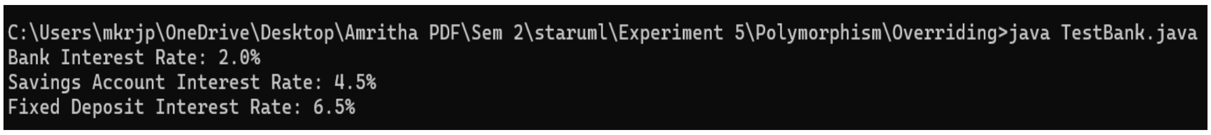
a)Bank Test

Code:

```
class Bank {
    double getInterestRate() {
        return 2.0;  // Default interest rate
    }
}

class SavingsAccount extends Bank {
    double getInterestRate() {
        return 4.5;  // Savings account rate
    }
}

class FixedDeposit extends Bank {
    double getInterestRate() {
        return 6.5;  // Fixed deposit rate
    }
}

public class TestBank {
    public static void main(String[] args) {
        Bank bank = new Bank();
        Bank savings = new SavingsAccount();
        Bank fixed = new FixedDeposit();

        System.out.println("Bank Interest Rate: " + bank.getInterestRate() +
"%");
```

```
        System.out.println("Savings Account Interest Rate: " +
savings.getInterestRate() + "%");
        System.out.println("Fixed Deposit Interest Rate: " +
fixed.getInterestRate() + "%");
    }
}
```

Output:



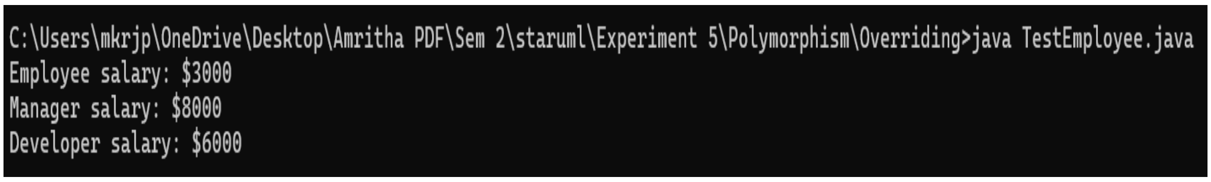
```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 5\Polymorphism\Overriding>java TestBank.java
Bank Interest Rate: 2.0%
Savings Account Interest Rate: 4.5%
Fixed Deposit Interest Rate: 6.5%
```

b)Employee Test

Code:

```
class Employee {
    void calculateSalary() {
        System.out.println("Employee salary: $3000");
    }
}

class Manager extends Employee {
    void calculateSalary() {
        System.out.println("Manager salary: $8000");
    }
}

class Developer extends Employee {
    void calculateSalary() {
        System.out.println("Developer salary: $6000");
    }
}

public class TestEmployee {
    public static void main(String[] args) {
        Employee emp = new Employee();
        Employee mgr = new Manager();
        Employee dev = new Developer();

        emp.calculateSalary();
        mgr.calculateSalary();
        dev.calculateSalary();
    }
}
```

Output:



```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 5\Polymorphism\Overriding>java TestEmployee.java
Employee salary: $3000
Manager salary: $8000
Developer salary: $6000
```

EXPERIMENT-6

ABSRTACTION

1.Absract class

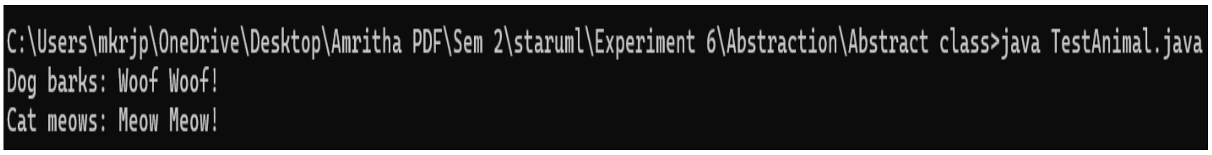
a) Animal Test

Code:

```
abstract class Animal {
   abstract void makeSound();
}

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



class Cat extends Animal {
   void makeSound() {
      System.out.println("Cat meows: Meow Meow!");
   }
}
```

```
public class TestAnimal {
    public static void main(String[] args) {
        Animal myDog = new Dog();
        Animal myCat = new Cat();

        myDog.makeSound();
        myCat.makeSound();
    }
}
```

Output:



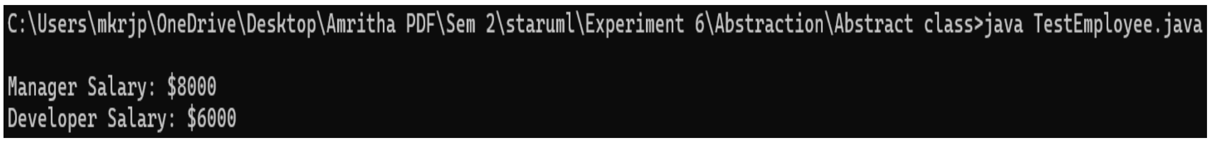
```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 6\Abstraction\Abstract class>java TestAnimal.java
Dog barks: Woof Woof!
Cat meows: Meow Meow!
```

b)Employee Test

Code:

```
abstract class Employee {
    abstract void calculateSalary();
}



class Manager extends Employee {
    void calculateSalary() {
        System.out.println("Manager Salary: $8000");
    }
}


class Developer extends Employee {
    void calculateSalary() {
        System.out.println("Developer Salary: $6000");
    }
}


public class TestEmployee {
    public static void main(String[] args) {
        Employee mgr = new Manager();
        Employee dev = new Developer();

        mgr.calculateSalary();
        dev.calculateSalary();
    }
}
```

Output:


```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 6\Abstraction\Abstract class>java TestEmployee.java

Manager Salary: $8000
Developer Salary: $6000
```

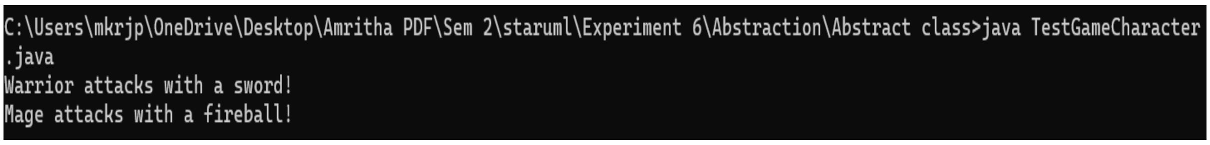
c)Game Character Test

Code:

```
abstract class GameCharacter {
   abstract void attack();
}

class Warrior extends GameCharacter {
   void attack() {
      System.out.println("Warrior attacks with a sword!");
   }
}

class Mage extends GameCharacter {
   void attack() {
      System.out.println("Mage attacks with a fireball!");
   }
}

public class TestGameCharacter {
   public static void main(String[] args) {
      GameCharacter warrior = new Warrior();
```

```
        GameCharacter mage = new Mage();

        warrior.attack();
        mage.attack();
    }
}
```

Output:



```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 6\Abstraction\Abstract class>java TestGameCharacter
.java
Warrior attacks with a sword!
Mage attacks with a fireball!
```

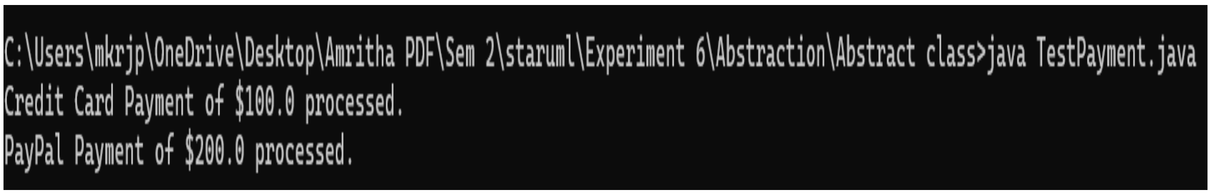
d)Payment Test

Code:

```
abstract class OnlinePayment {
    abstract void processPayment(double amount);
}

class CreditCardPayment extends OnlinePayment {
    void processPayment(double amount) {
        System.out.println("Credit Card Payment of $" + amount
+ " processed.");
    }
}
```

```
class PayPalPayment extends OnlinePayment {
   void processPayment(double amount) {
      System.out.println("PayPal Payment of $" + amount + "
processed.");
   }
}

public class TestPayment {
   public static void main(String[] args) {
      OnlinePayment creditCard = new CreditCardPayment();
      OnlinePayment paypal = new PayPalPayment();

      creditCard.processPayment(100);
      paypal.processPayment(200);
   }
}
```

Output:



```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 6\Abstraction\Abstract class>java TestPayment.java
Credit Card Payment of $100.0 processed.
PayPal Payment of $200.0 processed.
```

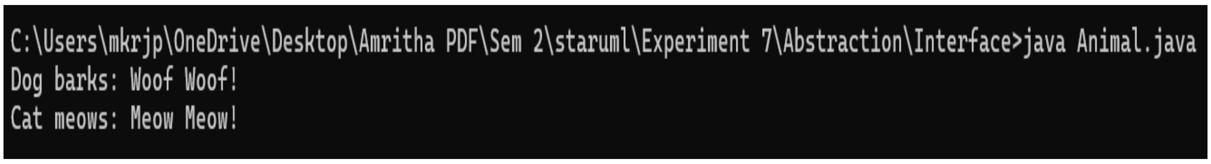
2.Interface class

a) Animal

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

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

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

public class Animal {
    public static void main(String[] args) {
        Example d = new Dog();
        Example c = new Cat();

        d.makeSound();
        c.makeSound();
    }
```

}

Output:



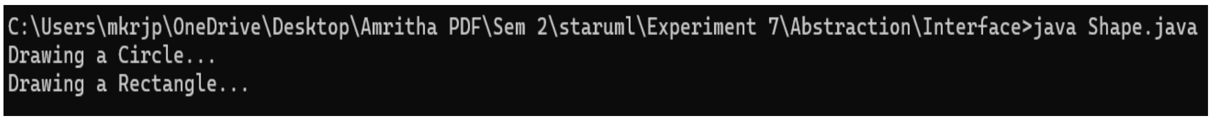
```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 7\Abstraction\Interface>java Animal.java
Dog barks: Woof Woof!
Cat meows: Meow Meow!
```

b)Shape

Code:

```
abstract class  Example {
   abstract void draw(); // Abstract method
}

class Circle extends Example {
   void draw() {
      System.out.println("Drawing a Circle...");
   }
}

class Rectangle extends Example{
   void draw() {
      System.out.println("Drawing a Rectangle...");
   }
}

public class Shape {
    public static void main(String[] args) {
        Example s1 = new Circle();
        Example s2 = new Rectangle();

        s1.draw();
        s2.draw();
    }
}
```

Output:



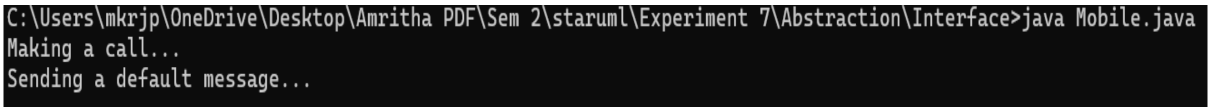
```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 7\Abstraction\Interface>java Shape.java
Drawing a Circle...
Drawing a Rectangle...
```

c)Mobile

Code:

```
interface Example {
    void call();

    default void message() {
        System.out.println("Sending a default message...");
    }
}

class Smartphone implements Example {
    public void call() {
        System.out.println("Making a call...");
    }
}

public class Mobile {
    public static void main(String[] args) {
        Smartphone phone = new Smartphone();
        phone.call();
        phone.message();
    }
}
```

Output:



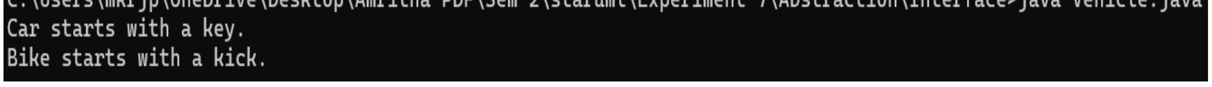
```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 7\Abstraction\Interface>java Mobile.java
Making a call...
Sending a default message...
```

d)Vehicle

Code:

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

class Car implements Example {
    public void start() {
        System.out.println("Car starts with a key.");
    }
}

class Bike implements Example {
    public void start() {
        System.out.println("Bike starts with a kick.");
    }
}

public class Vehicle {
    public static void main(String[] args) {
        Example v1 = new Car();
        Example v2 = new Bike();

        v1.start();
        v2.start();
    }
}
```

Output:



```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 7\Abstraction\Interface>java Vehicle.java
Car starts with a key.
Bike starts with a kick.
```

EXPERIMENT-8

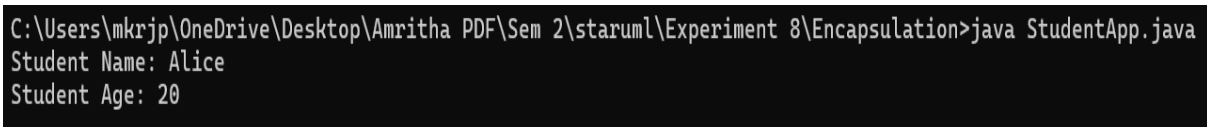
ENCAPSULATION

a)Student App

Code:

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

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

    public void setAge(int age) {
        if (age > 0) {
            this.age = age;
        }
    }

    // Getter methods
    public String getName() {
        return name;
    }
    public int getAge() {
        return age;
    }
}

public class StudentApp {
    public static void main(String[] args) {
        Student s = new Student();
        s.setName("Alice");
        s.setAge(20);
        System.out.println("Student Name: " + s.getName());
        System.out.println("Student Age: " + s.getAge());
    }
}
```

Output:



```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 8\Encapsulation>java StudentApp.java
Student Name: Alice
Student Age: 20
```

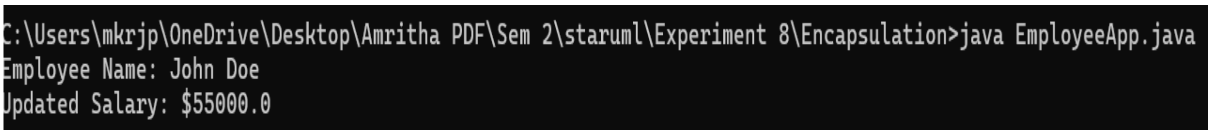
b)Employee App

Code:

```
class Employee {
    private String empName;
    private double salary;
    // Constructor
    public Employee(String empName, double salary) {
        this.empName = empName;
        this.salary = salary;
    }

    // Getter methods
    public String getEmpName() {
        return empName;
    }

    public double getSalary() {
        return salary;
    }

    // Setter method
    public void setSalary(double salary) {
        if (salary > 0) {
            this.salary = salary;
        }
    }
}
```

```
public class EmployeeApp {
    public static void main(String[] args) {
        Employee emp = new Employee("John Doe", 50000);
        emp.setSalary(55000);
        System.out.println("Employee Name: " +
emp.getEmpName());
        System.out.println("Updated Salary: $" +
emp.getSalary());
    }
}
```

Output:



```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 8\Encapsulation>java EmployeeApp.java
Employee Name: John Doe
Updated Salary: $55000.0
```

EXPERIMENT-9

PACKAGES

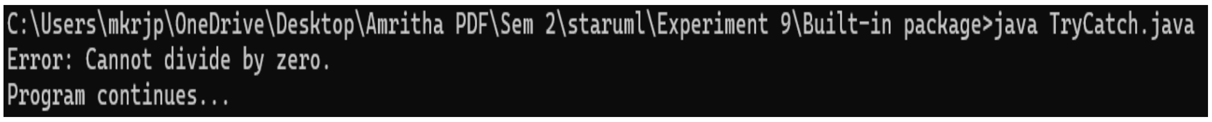
1.Built-in package

a)Try and Catch

Code:

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

Output:



```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 9\Built-in package>java TryCatch.java
Error: Cannot divide by zero.
Program continues...
```

b)Random

Code:

```
import java.util.Random; // Importing built-in package
```

```
}



public class Random {
    public static void main(String[] args) {
        Random rand = new Random();
        System.out.println("Random Number: " +
rand.nextInt(100));
    }
}
```

Output:

2.User-defined packages

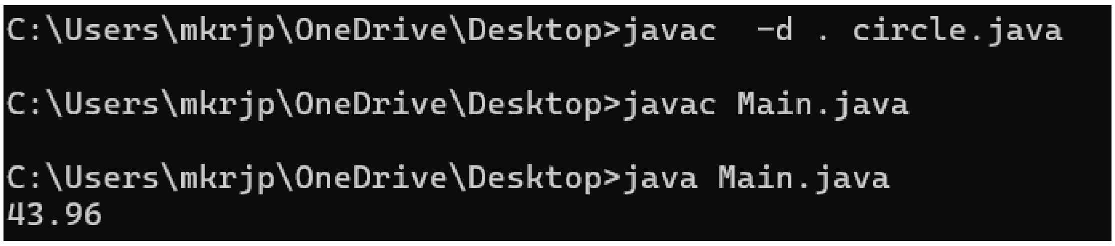
a)Shape

Code:

```
package shapes;

public class circle{
public int r;

 public void area(){
System.out.println(2*3.14*r);

}
}
```

```
import shapes.circle;

public class Main{
 public static void main(String [] args){
  circle c = new circle();
 c.r = 7;
c.area();

}
}
```

Output:



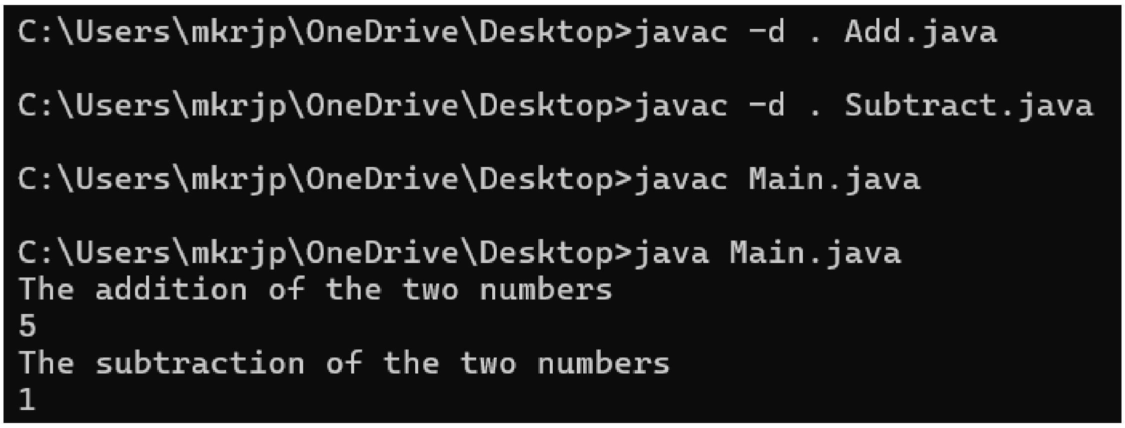
```
C:\Users\mkrjp\OneDrive\Desktop>javac  -d . circle.java

C:\Users\mkrjp\OneDrive\Desktop>javac Main.java

C:\Users\mkrjp\OneDrive\Desktop>java Main.java
43.96
```

b)Addition and subtraction

Code:

```
package add;

public class Add{
 public int a;
 public int b;
 public int addition(){
 System.out.println("The addition of the two numbers");
 return a+b;
}
}


package subtract;

public class Subtract{
 public int a;
 public int b;
 public int subtraction(){
 System.out.println("The subtraction of the two numbers");
 return a-b;
}
}


import subtract.Subtract;
import add.Add;

public class Main{
 public static void main(String[] args){
  Add d = new Add();
  Subtract s = new Subtract();

d.a = 2;
d.b = 3;

System.out.println(d.addition());
s.a = 5;
s.b = 4;

System.out.println(s.subtraction());

}
}
```

Output:



```
C:\Users\mkrjp\OneDrive\Desktop>javac -d . Add.java

C:\Users\mkrjp\OneDrive\Desktop>javac -d . Subtract.java

C:\Users\mkrjp\OneDrive\Desktop>javac Main.java

C:\Users\mkrjp\OneDrive\Desktop>java Main.java
The addition of the two numbers
5
The subtraction of the two numbers
1
```

EXPERIMENT-10

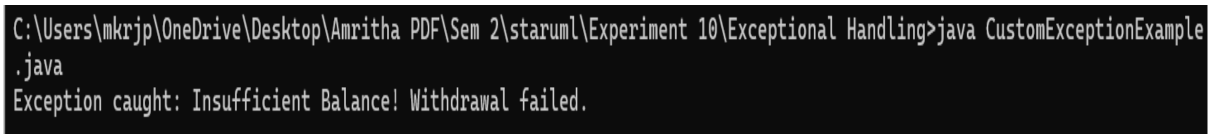
EXCEPTION HANDLING

1.Custom Exception

Code:

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

// Bank Account Class
class BankAccount {
    private double balance = 1000;

    public void withdraw(double amount) throws
InsufficientBalanceException {
        if (amount > balance) {
            throw new InsufficientBalanceException("Insufficient
Balance! Withdrawal failed.");
        } else {
            balance -= amount;
            System.out.println("Withdrawal successful. Remaining
balance: " + balance);
        }
    }
}

// Main Class
public class CustomExceptionExample {
    public static void main(String[] args) {
        BankAccount account = new BankAccount();
        try {
            account.withdraw(1500); // This will throw an
exception
        } catch (InsufficientBalanceException e) {
            System.out.println("Exception caught: " +
e.getMessage());
        }
    }
}
```

Output:



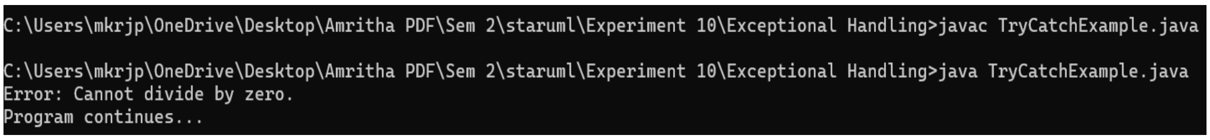
```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 10\Exceptional Handling>java CustomExceptionExample
.java
Exception caught: Insufficient Balance! Withdrawal failed.
```

2. Try Catch

   Code:

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

Output:

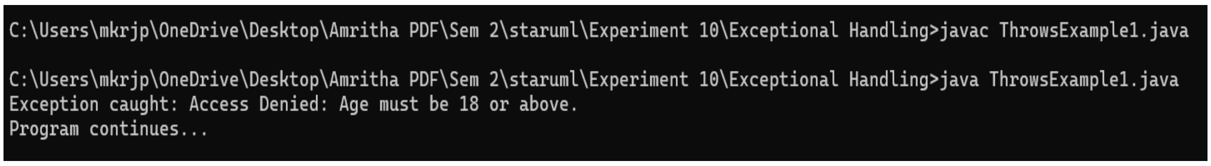


```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 10\Exceptional Handling>javac TryCatchExample.java

C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 10\Exceptional Handling>java TryCatchExample.java
Error: Cannot divide by zero.
Program continues...
```

3.Multiple Catch

```
class ExceptionExample {
    static void checkAge(int age) throws
IllegalArgumentException {
        if (age < 18) {
            throw new IllegalArgumentException("Access
Denied: Age must be 18 or above.");
        } else {
            System.out.println("Access Granted.");
        }
    }
}

public class ThrowsExample1 {
    public static void main(String[] args) {
        try {
            ExceptionExample.checkAge(15); // This will
throw an exception
        } catch (IllegalArgumentException e) {
            System.out.println("Exception caught: " +
e.getMessage());
        }
        System.out.println("Program continues...");
    }
}
```

Output:



```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 10\Exceptional Handling>javac ThrowsExample1.java

C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 10\Exceptional Handling>java ThrowsExample1.java
Exception caught: Access Denied: Age must be 18 or above.
Program continues...
```

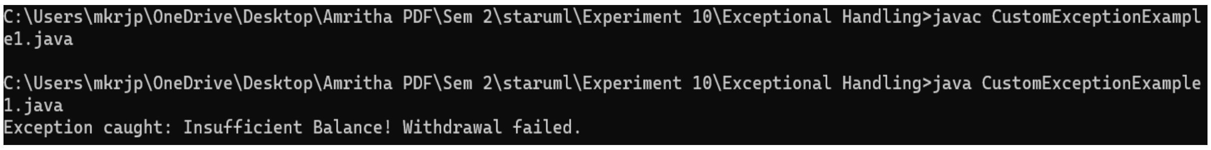
4.Throws

Code:

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


// Bank Account Class
class BankAccount {
    private double balance = 1000;
    public void withdraw(double amount) throws
InsufficientBalanceException {
        if (amount > balance) {
            throw new InsufficientBalanceException("Insufficient
Balance! Withdrawal failed.");
        } else {
            balance -= amount;
            System.out.println("Withdrawal successful. Remaining
balance: " + balance);
        }
    }
}

// Main Class
public class CustomExceptionExample1 {
    public static void main(String[] args) {
        BankAccount account = new BankAccount();
        try {
            account.withdraw(1500); // This will throw an
exception
        } catch (InsufficientBalanceException e) {
            System.out.println("Exception caught: " +
e.getMessage());
```

```
        }
    }
}
```

Output:



```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 10\Exceptional Handling>javac CustomExceptionExampl
e1.java

C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 10\Exceptional Handling>java CustomExceptionExample
1.java
Exception caught: Insufficient Balance! Withdrawal failed.
```

EXPERIMENT-11

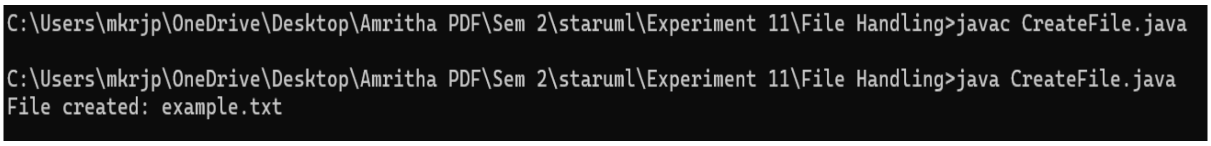
FILE HANDLING

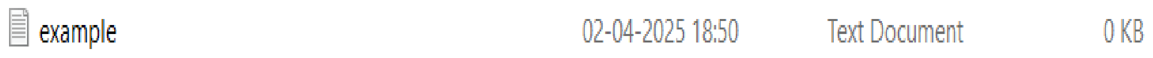
1.Create File

Code:

```
import java.io.File;
import java.io.IOException;

public class CreateFile {
    public static void main(String[] args) {
        try {
            File myFile = new File("example.txt");
            if (myFile.createNewFile()) {
                System.out.println("File created: " +
myFile.getName());
            } else {
                System.out.println("File already exists.");
            }
        } catch (IOException e) {
            System.out.println("An error occurred.");
            e.printStackTrace();
        }
    }
}
```

Output:


```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 11\File Handling>javac CreateFile.java

C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 11\File Handling>java CreateFile.java
File created: example.txt
```

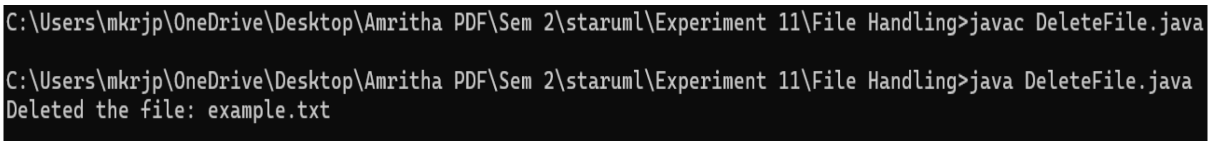

```
example                02-04-2025 18:50        Text Document        0 KB
```

2.Delete File

Code:

```
import java.io.File;


public class DeleteFile {
    public static void main(String[] args) {
        File myFile = new File("example.txt");
        if (myFile.delete()) {
            System.out.println("Deleted the file: " +
myFile.getName());
        } else {
            System.out.println("Failed to delete the file.");
        }
    }
}
```

Output:

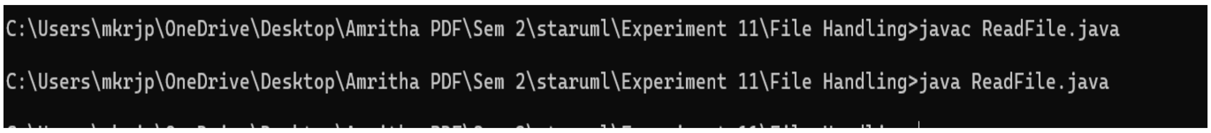

```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 11\File Handling>javac DeleteFile.java

C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 11\File Handling>java DeleteFile.java
Deleted the file: example.txt
```

3.Read File

Code:

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

public class ReadFile {
    public static void main(String[] args) {
        try {
            File myFile = new File("example.txt");
            Scanner reader = new Scanner(myFile);
            while (reader.hasNextLine()) {
                String data = reader.nextLine();
                System.out.println(data);
            }
            reader.close();
        } catch (FileNotFoundException e) {
            System.out.println("File not found.");
            e.printStackTrace();
        }
    }
}
```

Output:



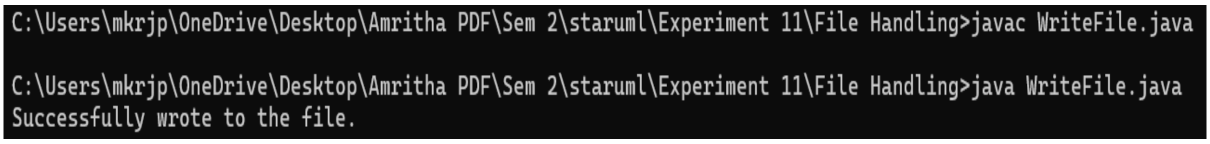
```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 11\File Handling>javac ReadFile.java

C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 11\File Handling>java ReadFile.java
```

4.Write File

Code:

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

public class WriteFile {
    public static void main(String[] args) {
        try {
            FileWriter writer = new FileWriter("example.txt");
            writer.write("Hello, this is a file handling example in Java!");
            writer.close();
            System.out.println("Successfully wrote to the file.");
        } catch (IOException e) {
            System.out.println("An error occurred.");
            e.printStackTrace();
        }
    }
}
```

Output:



```
C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 11\File Handling>javac WriteFile.java

C:\Users\mkrjp\OneDrive\Desktop\Amritha PDF\Sem 2\staruml\Experiment 11\File Handling>java WriteFile.java
Successfully wrote to the file.
```