LAB RECORD

23CSE111- Object Oriented Programming

Submitted by

CH.SC.U4CSE24147 -VASANTHA .T

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.U4CSE24147 – Vasantha T* in **"Computer Science and Engineering"** is a Bonafide record of the work carried out under my guidance and supervision at Amrita School of Computing, Chennai.

This Lab examination held on / /2025

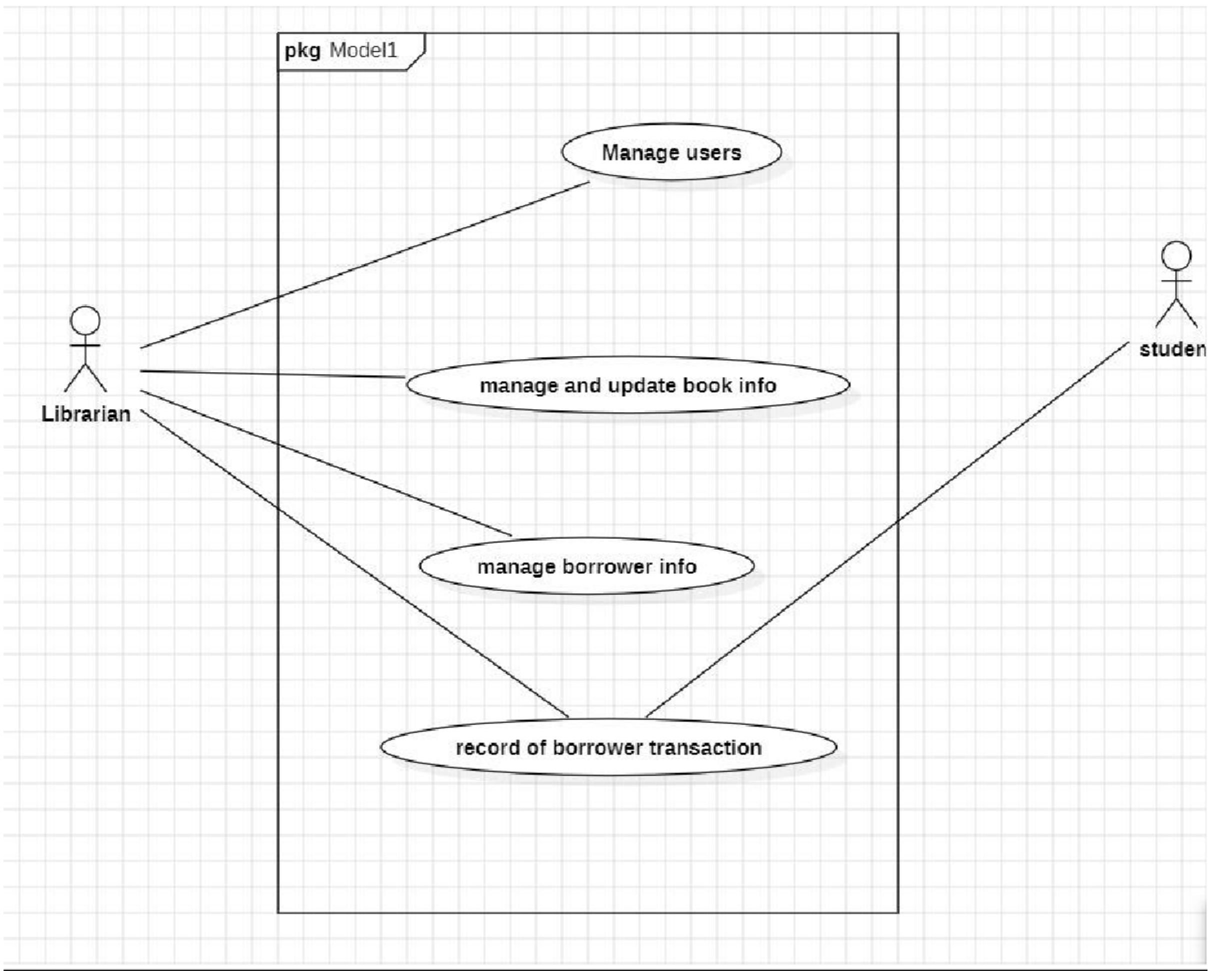
Internal Examiner 1 Internal Examiner 2

INDEX

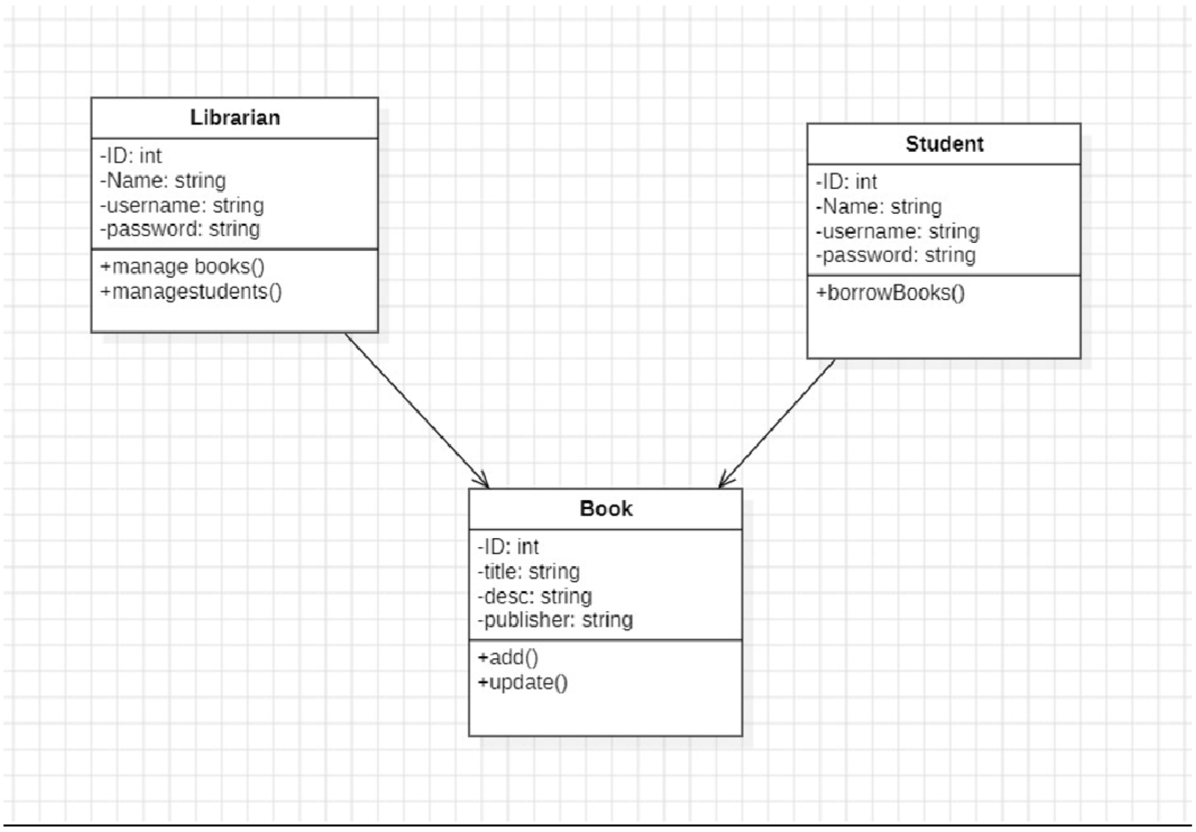
UML DIAGRAMS

1. LIBRARY MANAGEMENT SYSTEM

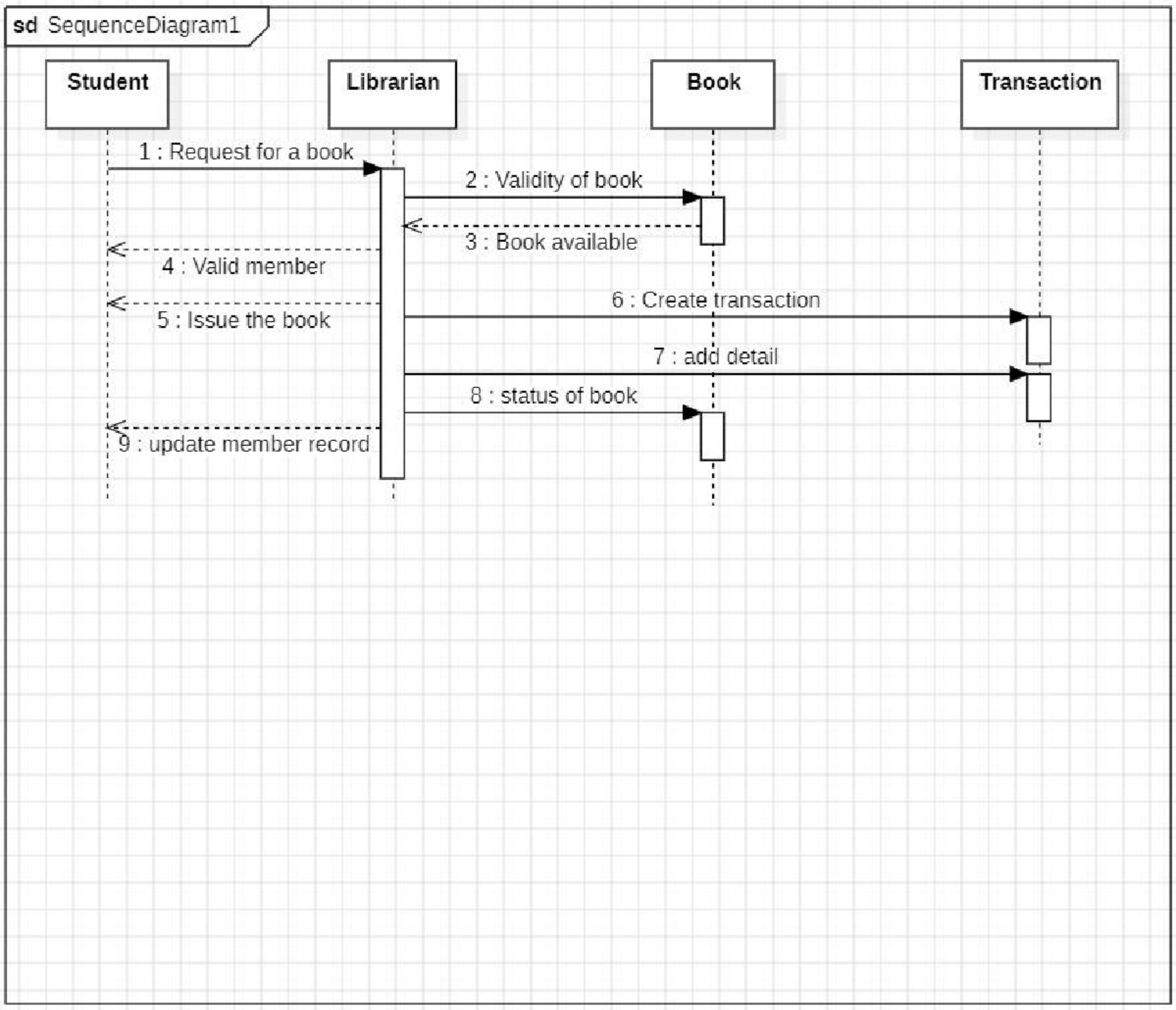
1.a) Use Case Diagram:



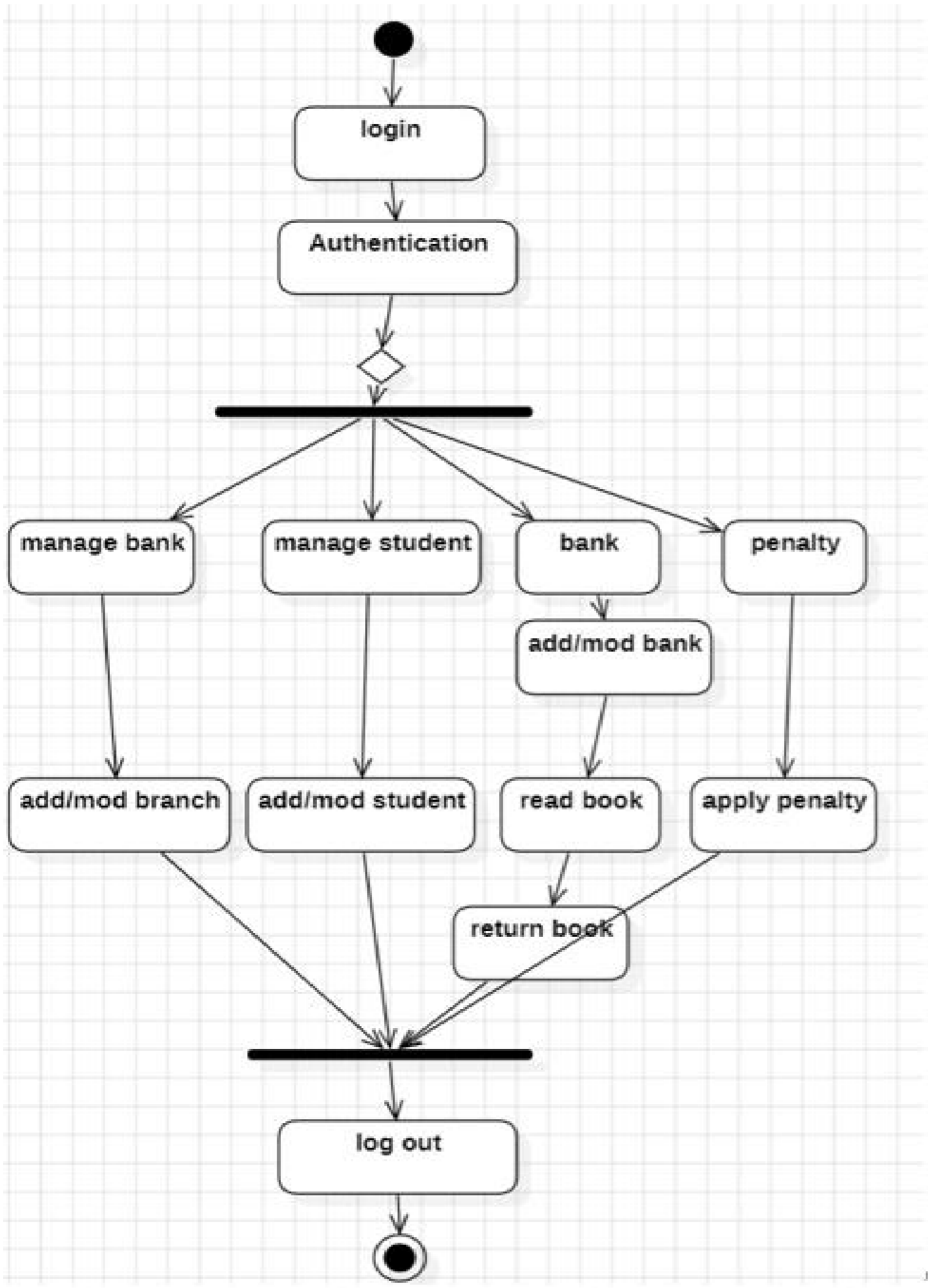
1.b) Class Diagram



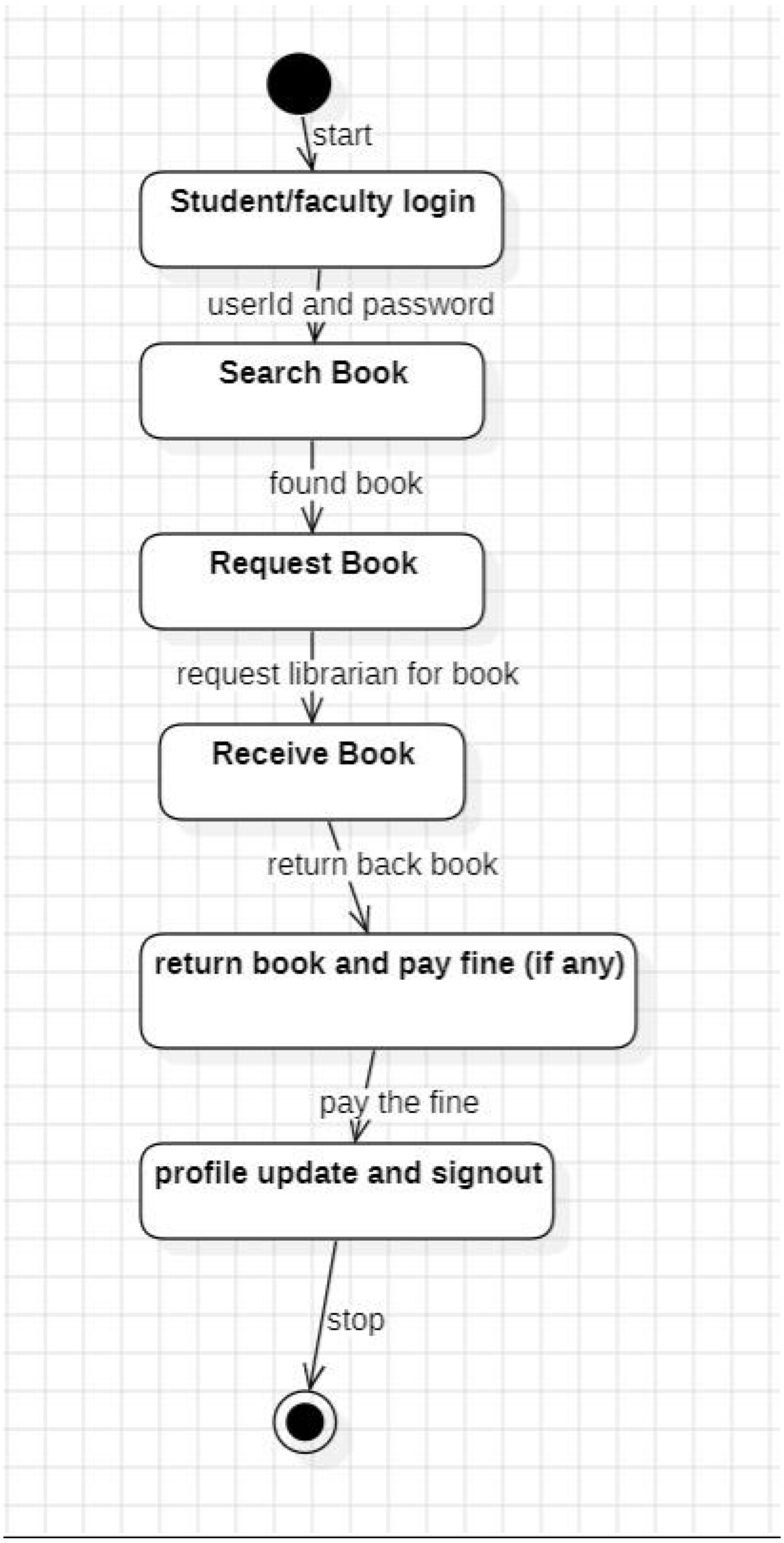
1.c) Sequence Diagram



1.d) State Diagram

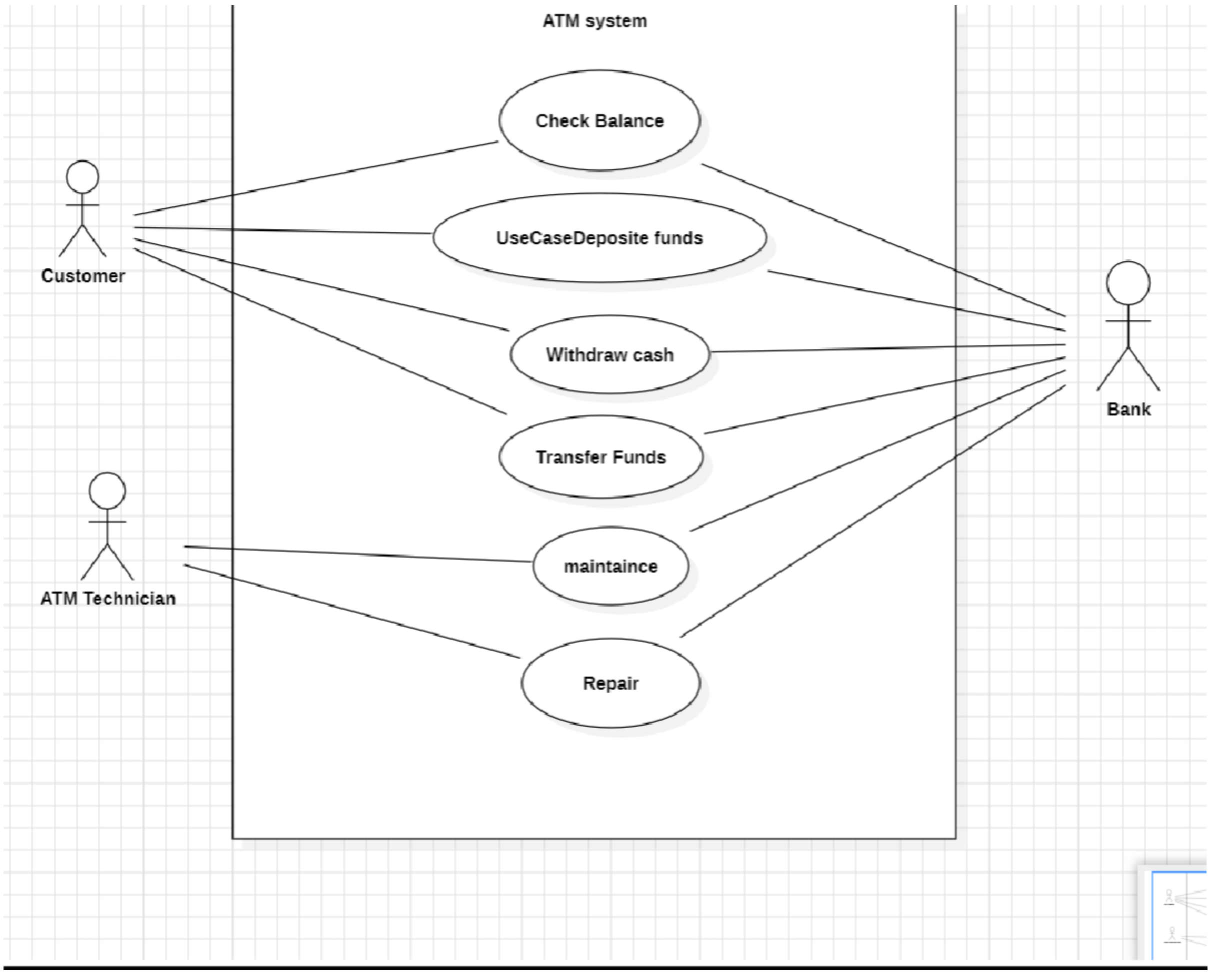


1.e) Activity Diagram

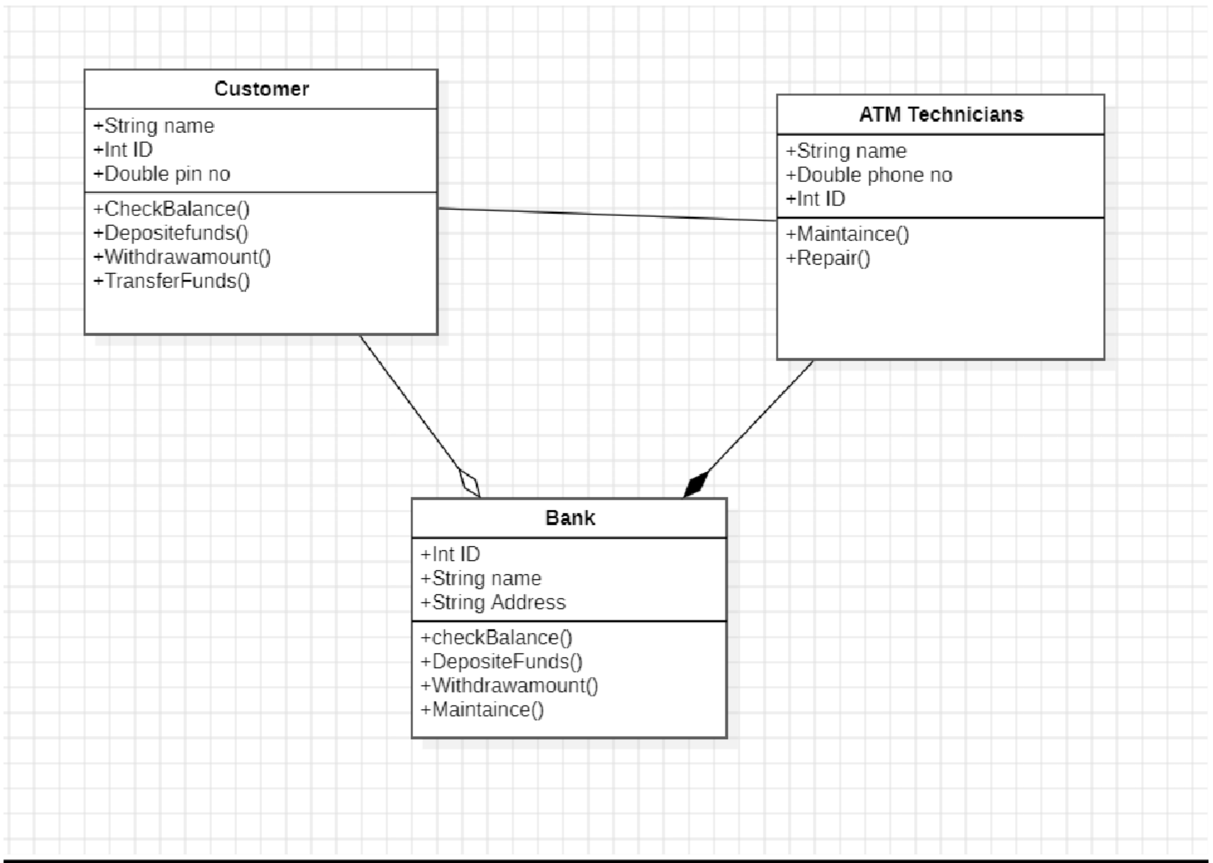

start
Student/faculty login
userId and password
Search Book
found book
Request Book
request librarian for book
Receive Book
return back book
return book and pay fine (if any)
pay the fine
profile update and signout
stop

2. ATM SYSTEM

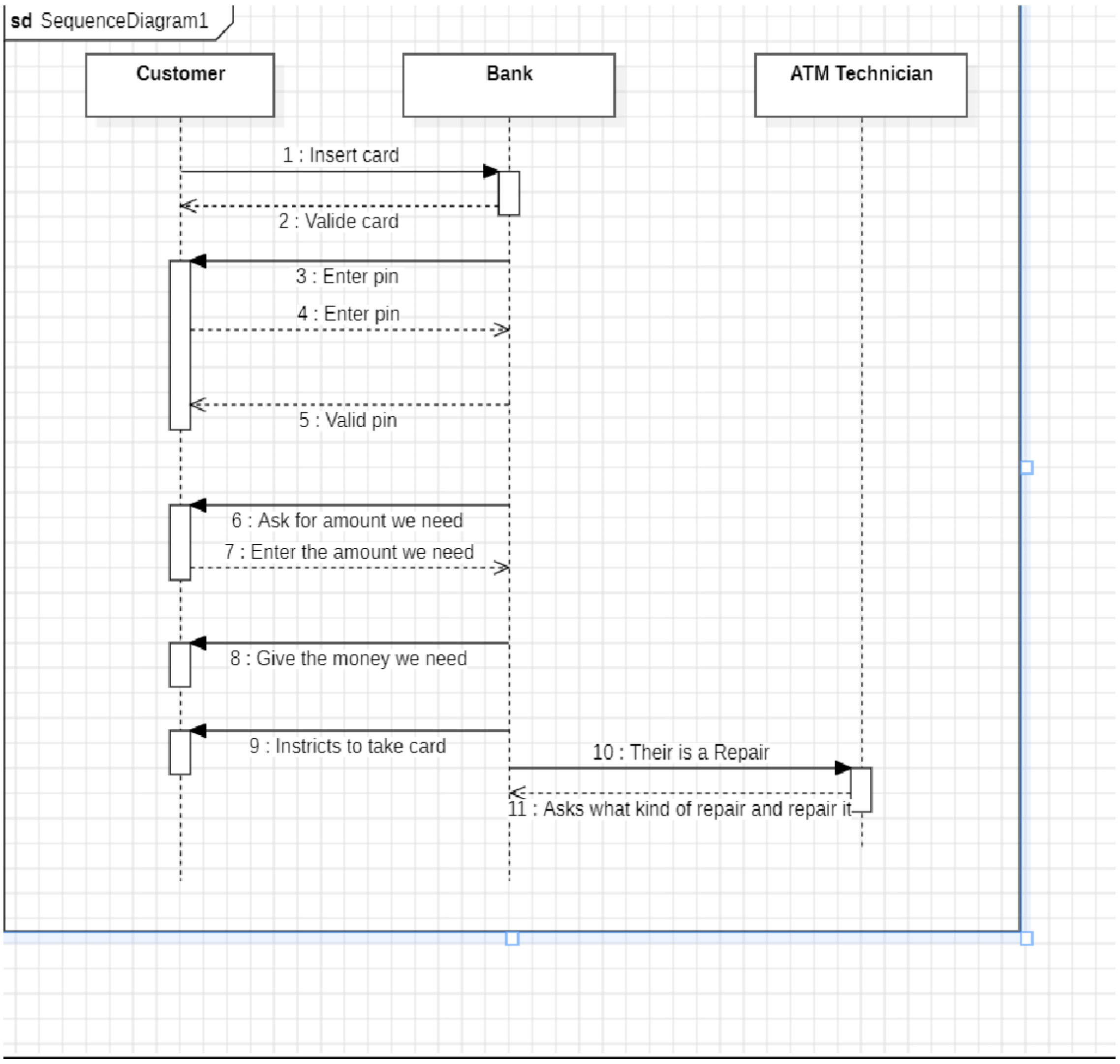
2.a) Use Case Diagram



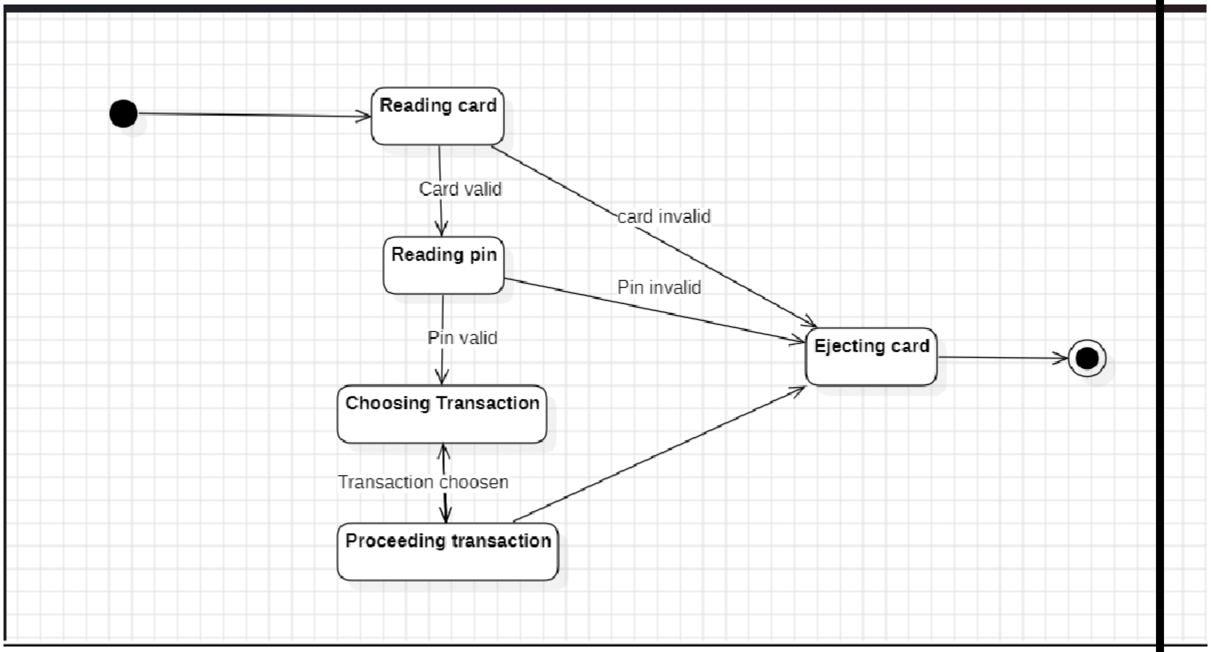
2.b) Class Diagram



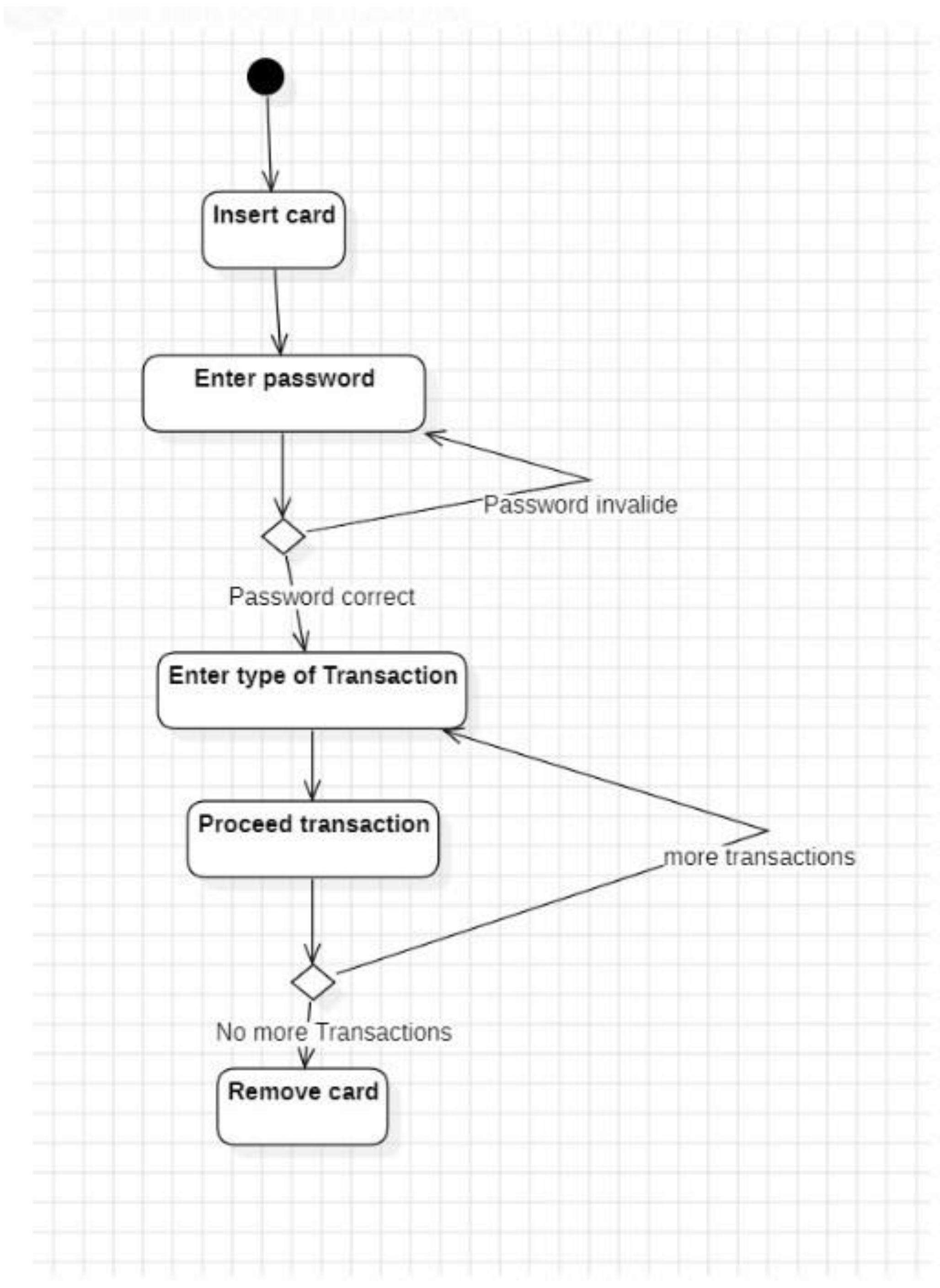
<u>2.c)</u> Sequence Diagram


sd SequenceDiagram1
Customer
Bank
ATM Technician
1 : Insert card
2 : Valide card
3 : Enter pin
4 : Enter pin
5 : Valid pin
6 : Ask for amount we need
7 : Enter the amount we need
8 : Give the money we need
9 : Instricts to take card
10 : Their is a Repair
11 : Asks what kind of repair and repair it

2.d) State Diagram



2.e) Activity Diagram


Insert card
Enter password
Password invalide
Password correct
Enter type of Transaction
Proceed transaction
more transactions
No more Transactions
Remove card

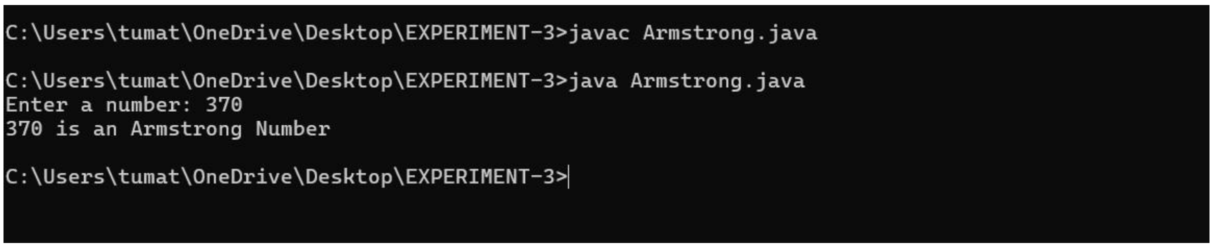
3. BASIC JAVA PROGRAMS

3.a) ARMSTRONG NUMBER

CODE:

```
import java.util.Scanner;
public class Armstrong {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter a number: ");
      int num = sc.nextInt(), sum = 0, temp = num;
      while (temp != 0) {
         int digit = temp % 10;
         sum += digit * digit * digit;
         temp /= 10;
      }
      System.out.println(num + " is " + (num == sum ? "an Armstrong Number" :
"not an Armstrong Number"));
         }
}
```

OUTPUT:



```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>javac Armstrong.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>java Armstrong.java
Enter a number: 370
370 is an Armstrong Number

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>
```

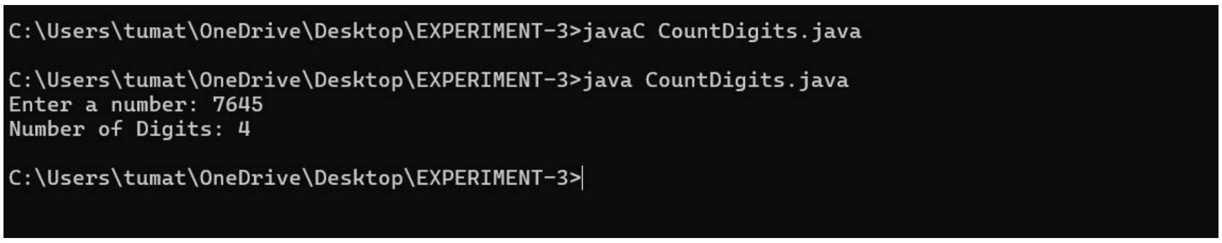
3.b) CountDigits

CODE:

```
import java.util.Scanner;
public class CountDigits {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter a number: ");
      int num = sc.nextInt(), count = 0;
      while (num != 0) {
         count++;
         num /= 10;
```

```
        }
        System.out.println("Number of Digits: " + count);

    }
}
```

OUTPUT:



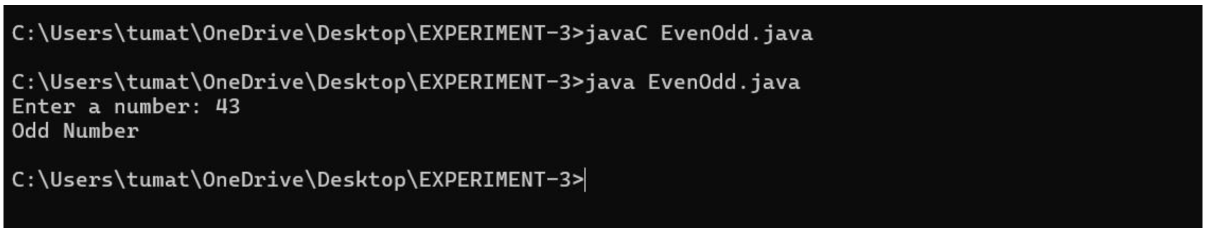
```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>javaC CountDigits.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>java CountDigits.java
Enter a number: 7645
Number of Digits: 4

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>
```

3.c) EvenOdd

CODE:

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

OUTPUT:



```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>javaC EvenOdd.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>java EvenOdd.java
Enter a number: 43
Odd Number

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>
```

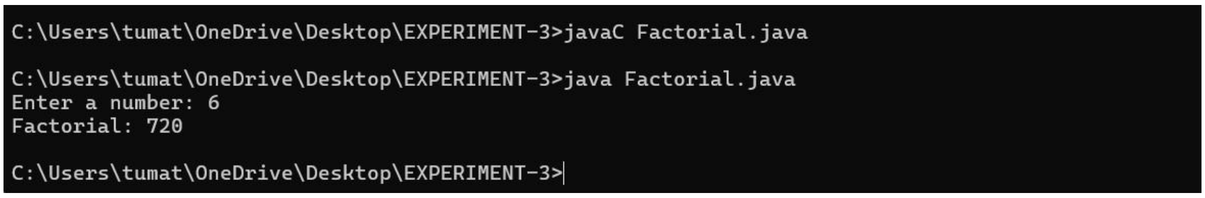
3.d) Factorial

CODE:

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

```
public class Factorial {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int num = sc.nextInt(), fact = 1;
        for (int i = 1; i <= num; i++) {
            fact *= i;
        }
        System.out.println("Factorial: " + fact);

    }
}
```

OUTPUT:



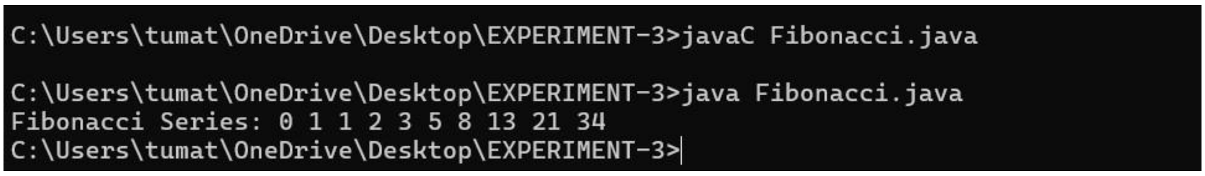
```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>javaC Factorial.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>java Factorial.java
Enter a number: 6
Factorial: 720

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>
```

3.e) Fibonacci

CODE:

```
public class Fibonacci {
        public static void main(String[] args) { int n = 10, a = 0,
                b = 1;
                System.out.print("Fibonacci Series: " + a + " " + b); for (int i = 2; i < n;
                i++) {
                        int next = a + b; System.out.print(" " + next);
                        a = b;
                        b = next;
                }
        }
}
```

OUTPUT:



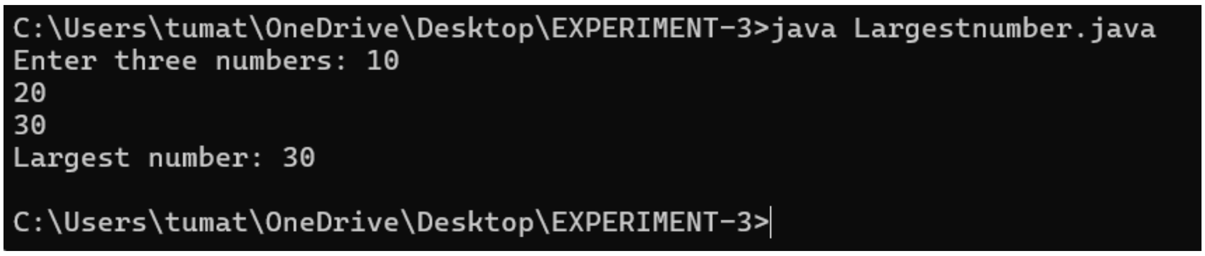
```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>javaC Fibonacci.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>java Fibonacci.java
Fibonacci Series: 0 1 1 2 3 5 8 13 21 34
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>
```

3.f) Largestnumber

CODE:

```
import java.util.Scanner; public class
Largestnumber {
        public static void main(String[] args) { Scanner sc = new
                Scanner(System.in); System.out.print("Enter three
                numbers: "); int a = sc.nextInt();
         int b = sc.nextInt(); int c =
         sc.nextInt();
                System.out.println("Largest: " + Math.max(a, Math.max(b,
c)));
                      }
}
```

OUTPUT:



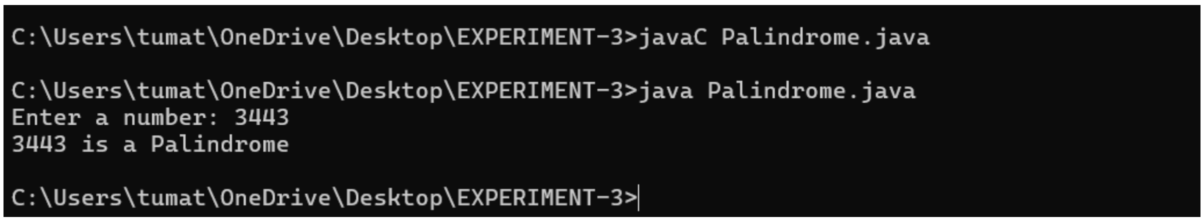
```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>java Largestnumber.java
Enter three numbers: 10
20
30
Largest number: 30

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>
```

3.g) **Palindrome**

CODE:

```
import java.util.Scanner; public  class
Palindrome {
        public static void main(String[] args) { Scanner sc = new
                Scanner(System.in); System.out.print("Enter  a
                number: ");
                int num = sc.nextInt(), original = num, rev =
0;
                while (num != 0) {
                        rev = rev * 10 + num % 10; num /=
                        10;
                }
                System.out.println(original + " is " + (original == rev ? "a
Palindrome" : "not a Palindrome"));
                      }
}
```

OUTPUT:



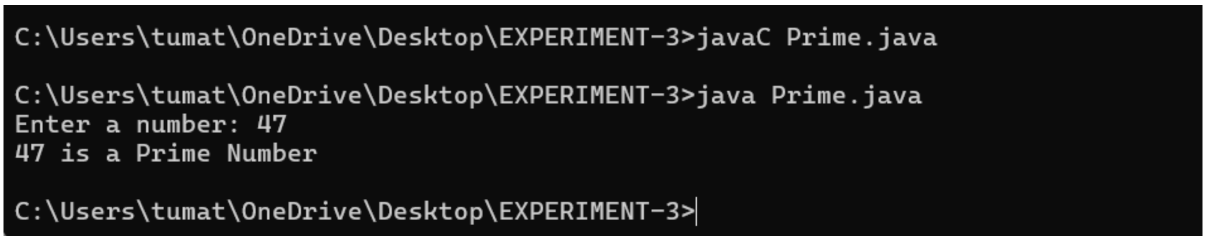
```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>javaC Palindrome.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>java Palindrome.java
Enter a number: 3443
3443 is a Palindrome

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>
```

3.h) Prime

CODE:

```
import java.util.Scanner; public class
Prime {
        public static void main(String[] args) { Scanner sc = new
                Scanner(System.in); System.out.print("Enter a
                number: "); int num = sc.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 + " is " + (isPrime ? "a Prime
Number" : "not a Prime Number"));
        }
}
```

OUTPUT:



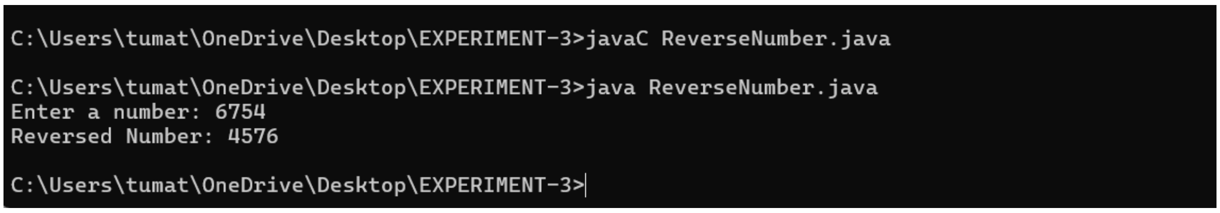
```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>javaC Prime.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>java Prime.java
Enter a number: 47
47 is a Prime Number

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>
```

3.i) ReverseNumber

CODE:

```
import java.util.Scanner; public class
ReverseNumber {
        public static void main(String[] args) { Scanner sc = new
                Scanner(System.in); System.out.print("Enter a
                number: ");
                int num = sc.nextInt(), rev = 0; while (num
                != 0) {
                        rev = rev * 10 + num % 10; num /=
                        10;
                }
                System.out.println("Reversed Number: " + rev);
                      }
}
```

OUTPUT:

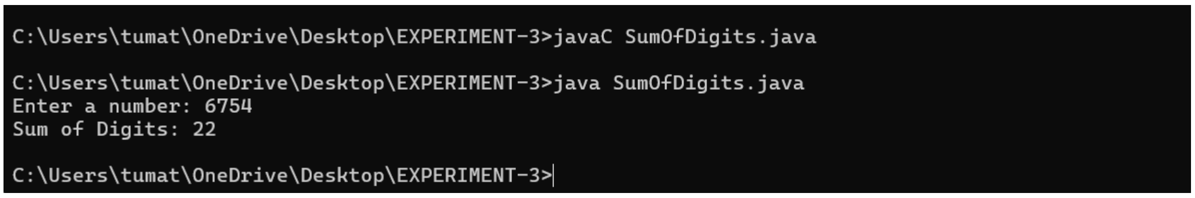


```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>javaC ReverseNumber.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>java ReverseNumber.java
Enter a number: 6754
Reversed Number: 4576

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>
```

3.j) **SumOfNumber**

CODE:

```
import java.util.Scanner; public class
SumOfDigits {
        public static void main(String[] args) { Scanner sc =
                new Scanner(System.in); System.out.print("Enter
                a number: "); int num = sc.nextInt(), sum = 0;
                while (num != 0) {
                        sum += num % 10;
                        num /= 10;



}

}
System.o
ut.printl
n("Sum
of Digits:
" +
sum);
      }
```



```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>javaC SumOfDigits.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>java SumOfDigits.java
Enter a number: 6754
Sum of Digits: 22

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-3>
```

Inheritance

4.Single Inheritance

4.a) Area_Volume

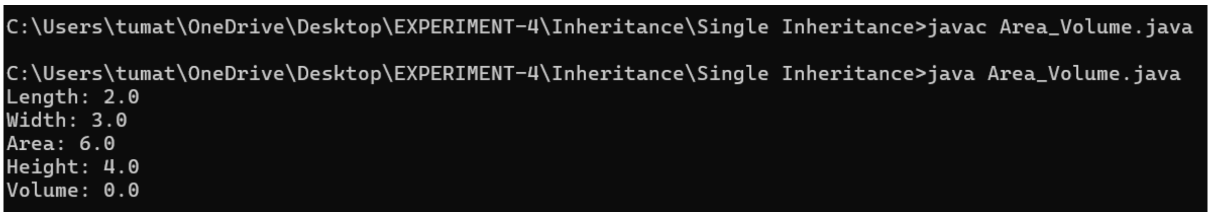
Code:

```
class Area {
       double length;
      double width;
      double area;



    public void Area(double length, double
width) {
 System.out.println("Length: " + length);
        System.out.println("Width: " + width);


            area=length*width;
```

```
                System.out.println("Area: " +
area);
    }
}

class Volume extends Area {
 double height;
  double v;



public void Volume(double height) {


                System.out.println("Height: " +
height);
            v=length*width*height;
            System.out.println("Volume: " + v);
    }
```

```
}


public class Area_Volume {
   public static void main(String[] args) {
        Volume obj = new Volume();
        obj.Area(2,3);
      obj.Volume(4);
      }
}
```

Output:



```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-4\Inheritance\Single Inheritance>javac Area_Volume.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-4\Inheritance\Single Inheritance>java Area_Volume.java
Length: 2.0
Width: 3.0
Area: 6.0
Height: 4.0
Volume: 0.0
```

4.b)StudentInformation

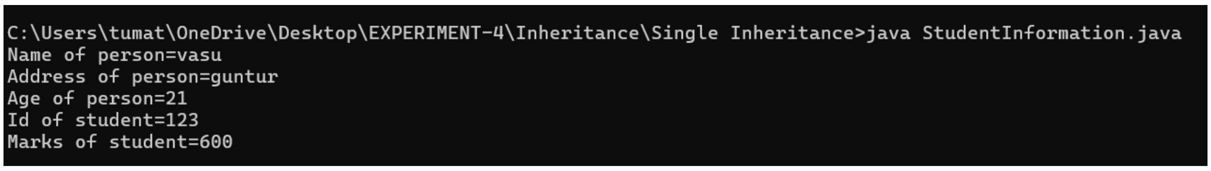
```
class person{
     String name;
```

```
	int age;
	String address;
	public void personinfo(String name,String address,int age){
		System.out.println("Name of person="+name);
		System.out.println("Address of person="+address);
		System.out.println("Age of person="+age);

	}
}
class student extends person{
	int id;
	int marks;
```

```
    public void studentinfo(int id,int
marks){
        System.out.println("Id of
student="+id);
        System.out.println("Marks of
student="+marks);

    }
}
public class StudentInformation{
    public static void main(String args[]){
        student obj=new student();
        obj.personinfo("vasu","guntur",21);
        obj.studentinfo(123,600);


    }
```

}

Output:



```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-4\Inheritance\Single Inheritance>java StudentInformation.java
Name of person=vasu
Address of person=guntur
Age of person=21
Id of student=123
Marks of student=600
```

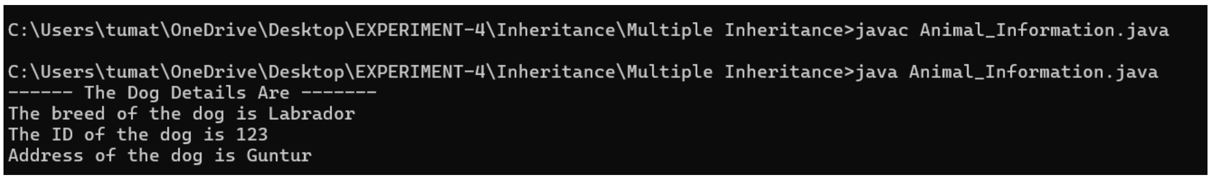
5.Multiple Inheritance

5.a)Animal_Information

```
class Animal {
   String breed;
            public void animalinfo(String
breed){
                     System.out.println("The
breed of the dog is "+breed);
            }
}


class Mammal extends Animal {
  int id;
```

```
            public void mammalinfo(int id){
                        System.out.println("The id
of the dog is"+id);
            }
}

class Dog extends Mammal {
    String address;

    public void Doginfo(String address) {

        System.out.println("Address of the
dog is " + address);
    }
}

public class Animal_Information {
    public static void main(String[] args) {
            System.out.println("------The Dog
Details Are-------");
        Dog d = new Dog();
        d.animalinfo("Lavander");
            d.mammalinfo(123);
            d.Doginfo("Guntur");
    }
```

Output:



```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-4\Inheritance\Multiple Inheritance>javac Animal_Information.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-4\Inheritance\Multiple Inheritance>java Animal_Information.java
------ The Dog Details Are -------
The breed of the dog is Labrador
The ID of the dog is 123
Address of the dog is Guntur
```

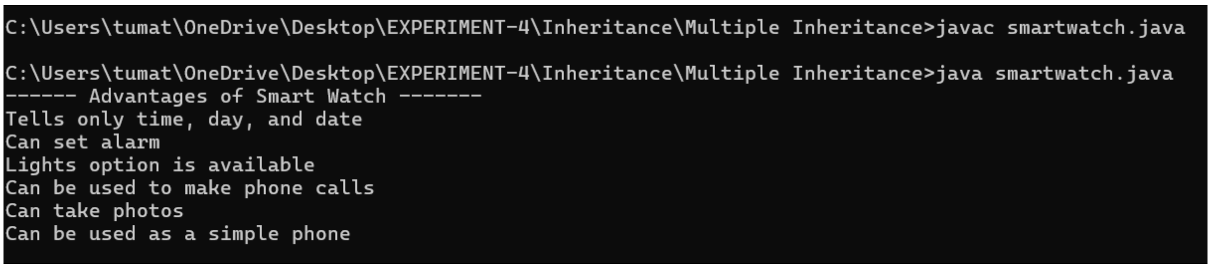
5.b)Smartwatch

```
class clock {
            public void clock(){
                        System.out.println("Tells
only time and Day and Date ");
            }
```

```
}

class Watch extends clock {
          public void watch(){
                        System.out.println("Can set
alarm");
                        System.out.println("Lights
option is there");
            }
}

class Smartwatch extends Watch {
      public void smartwatch() {
       System.out.println("Can be used to
make phone calls");
            System.out.println("Can take
photos");
```

```
            System.out.println("Can be used as
simple phone");
  }
}

public class smartwatch {
  public static void main(String[] args) {
            System.out.println("------
Advantages of smart watch-------");
        Smartwatch d = new Smartwatch();
        d.clock();
            d.watch();
            d.smartwatch();
  }
}
```

Output:



```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-4\Inheritance\Multiple Inheritance>javac smartwatch.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-4\Inheritance\Multiple Inheritance>java smartwatch.java
------ Advantages of Smart Watch -------
Tells only time, day, and date
Can set alarm
Lights option is available
Can be used to make phone calls
Can take photos
Can be used as a simple phone
```

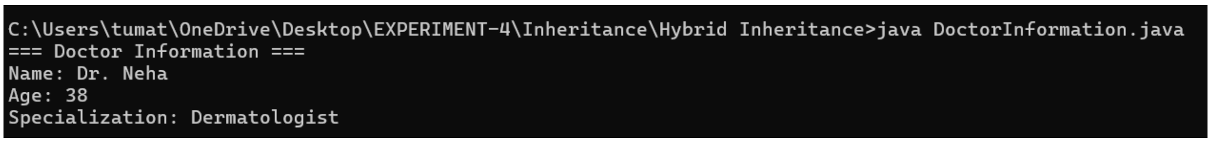
6.Hybrid Inhertance

6.a)DoctorInformation

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

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

    void showDetails() {
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
    }
}
interface Specialization {
    void setSpecialization(String spec);
    void showSpecialization();
}
class Doctor extends Person implements
Specialization {
    String specialization;

    public void setSpecialization(String spec)
{
        this.specialization = spec;
    }

    public void showSpecialization() {
        System.out.println("Specialization: " +
specialization);
    }

    void showDoctorInfo() {
        showDetails();
        showSpecialization();
    }
}
public class DoctorInformation {
    public static void main(String[] args) {
        Doctor doc = new Doctor();
        doc.setDetails("Dr. Neha", 38);

doc.setSpecialization("Dermatologist");
```

```
        System.out.println("=== Doctor
Information ===");
        doc.showDoctorInfo();
    }
}
```

Output:

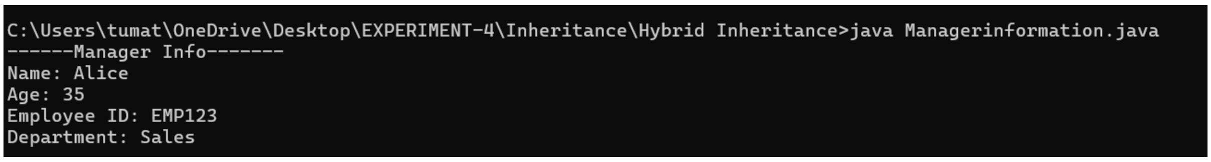


```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-4\Inheritance\Hybrid Inheritance>java DoctorInformation.java
=== Doctor Information ===
Name: Dr. Neha
Age: 38
Specialization: Dermatologist
```

6.b)ManagerInformation

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

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

    void showDetails() {
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
    }
}
class Employee extends Person {
    String employeeId;

    void setEmployeeId(String id) {
        this.employeeId = id;
    }

    void showEmployeeInfo() {
        System.out.println("Employee ID: " +
employeeId);
    }
}
interface Department {
    void setDepartment(String dept);
    void showDepartment();
}
class Manager extends Employee
implements Department {
    String department;

    public void setDepartment(String dept) {
        this.department = dept;
    }

    public void showDepartment() {
        System.out.println("Department: " +
department);
    }

    void showManagerInfo() {
        showDetails();
        showEmployeeInfo();
        showDepartment();
    }
}
public class Managerinformation{
    public static void main(String[] args) {
        Manager mgr = new Manager();
        mgr.getDetails("Alice", 35);
        mgr.setEmployeeId("EMP123");
        mgr.setDepartment("Sales");
        System.out.println("------Manager
Info-------");
        mgr.showManagerInfo();
    }
}
```

Output:



```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-4\Inheritance\Hybrid Inheritance>java Managerinformation.java
------Manager Info-------
Name: Alice
Age: 35
Employee ID: EMP123
Department: Sales
```

7.Hierarchal Inheritance

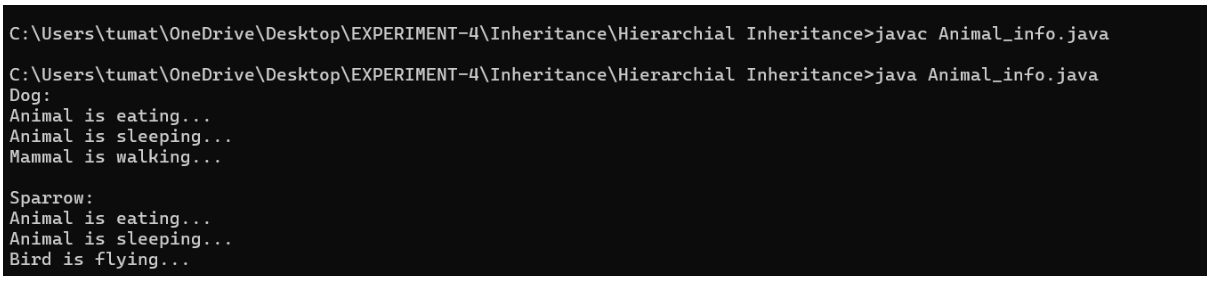
7.a)Animal_Info

```
class Animal {
    void eat() {
        System.out.println("Animal is
eating...");
    }
    void sleep() {
        System.out.println("Animal is
sleeping...");
    }
}
class Mammal extends Animal {
    void walk() {
        System.out.println("Mammal is
walking...");
    }
}
class Bird extends Animal {
    void fly() {
        System.out.println("Bird is flying...");
    }
}
```

```
public class Main {
    public static void main(String[] args) {
        // Mammal object
        Mammal dog = new Mammal();
        System.out.println(" Dog:");
        dog.eat();
       dog.sleep();
       dog.walk();
        System.out.println();

              Bird sparrow = new Bird();
        System.out.println("Sparrow:");
        sparrow.eat();
        sparrow.sleep();
        sparrow.fly();
    }
```

}

Output:



```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-4\Inheritance\Hierarchial Inheritance>javac Animal_info.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-4\Inheritance\Hierarchial Inheritance>java Animal_info.java
Dog:
Animal is eating...
Animal is sleeping...
Mammal is walking...

Sparrow:
Animal is eating...
Animal is sleeping...
Bird is flying...
```

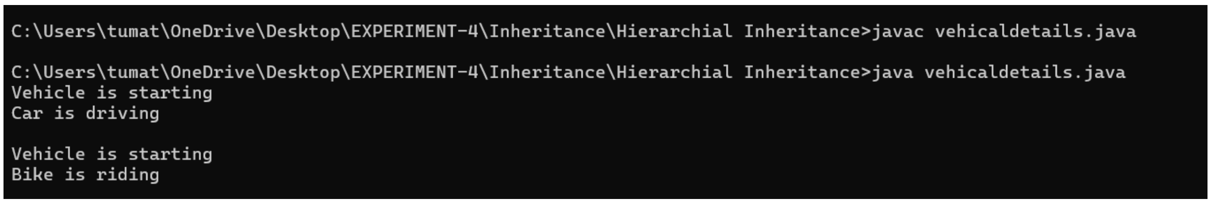
7.b)Vehicaldetails

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

```
}
class Bike extends Vehicle {
    void ride() {
        System.out.println(“Bike is riding”);
    }
}
public class Main {
    public static void main(String[] args) {
        Car myCar = new Car();
        myCar.start();
        myCar.drive();
        System.out.println();

        Bike myBike = new Bike();
        myBike.start();
     myBike.ride();     }
```

}

Output:



```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-4\Inheritance\Hierarchial Inheritance>javac vehicaldetails.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-4\Inheritance\Hierarchial Inheritance>java vehicaldetails.java
Vehicle is starting
Car is driving

Vehicle is starting
Bike is riding
```

Polymorphism

8.Constructor

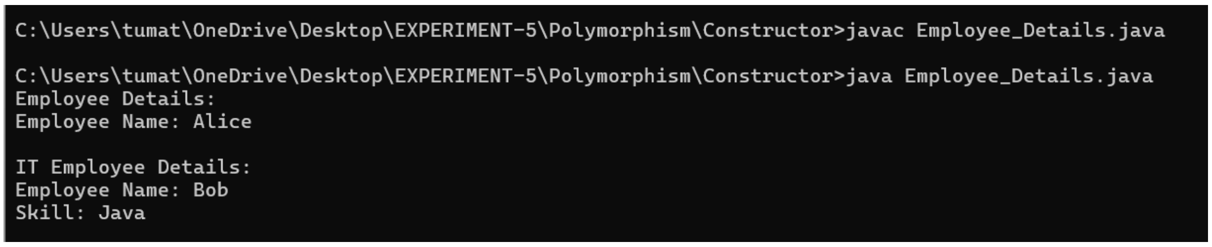
8.a)Employee_Details

```
class Employee {
    String name;


        Employee(String name) {
        this.name = name;
    }
```

```
       void display() {
        System.out.println("Employee Name:
" + name);
    }
}



class ITEmployee extends Employee {
    String skill;

      ITEmployee(String name, String skill) {
        super(name);
        this.skill = skill;
    }

       @Override
    void display() {
        super.display(); // Calls parent class
display method
        System.out.println("Skill: " + skill);
    }
}

public class Employee_Details {
    public static void main(String[] args) {
        Employee emp = new
Employee("Alice");
        ITEmployee itEmp = new
ITEmployee("Bob", "Java");

       display() method
        System.out.println("Employee
Details:");
```

```
        emp.display();

        System.out.println("\nIT Employee Details:");
        itEmp.display();
    }
}
```

Output:



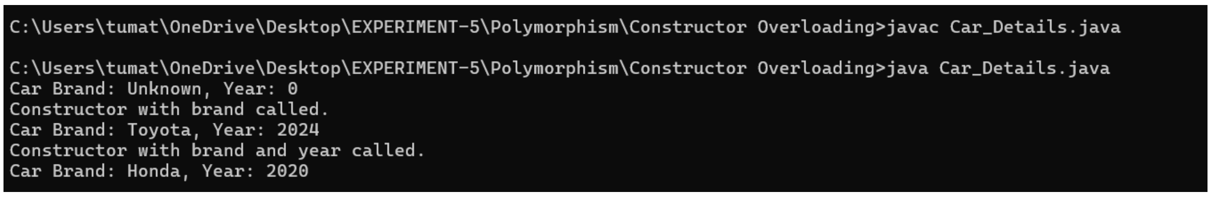
```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-5\Polymorphism\Constructor>javac Employee_Details.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-5\Polymorphism\Constructor>java Employee_Details.java
Employee Details:
Employee Name: Alice

IT Employee Details:
Employee Name: Bob
Skill: Java
```

9.Constructor Overloading

9.a)Car_Details

```
class Car {
    String brand;
    int year;
       Car() {
        this.brand = "Unknown";
        this.year = 0;

    }
        Car(String brand) {
        this.brand = brand;
        this.year = 2024; // Default year
        System.out.println("Constructor with
brand called.");
    }

    Car(String brand, int year) {
        this.brand = brand;
        this.year = year;
        System.out.println("Constructor with
brand and year called.");
    }

    void display() {
        System.out.println("Car Brand: " +
brand + ", Year: " + year);
    }
}

public class Car_Details {
    public static void main(String[] args) {
        Car car1 = new Car(); // Calls default
constructor
        car1.display();

        Car car2 = new Car("Toyota");
        car2.display();

        Car car3 = new Car("Honda", 2020);
        car3.display();
    }
}
```

Output:



```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-5\Polymorphism\Constructor Overloading>javac Car_Details.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-5\Polymorphism\Constructor Overloading>java Car_Details.java
Car Brand: Unknown, Year: 0
Constructor with brand called.
Car Brand: Toyota, Year: 2024
Constructor with brand and year called.
Car Brand: Honda, Year: 2020
```

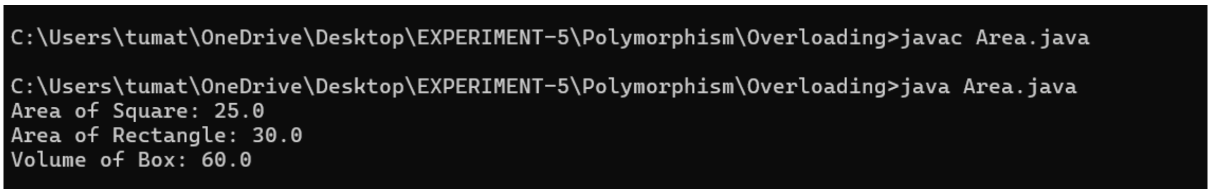
10.Overloading

10.a)Area

```
class AreaCalculator {
    double calculate(double side) {
        return side * side;
    }

      double calculate(double length,
double width) {
       return length * width;
    }

      double calculate(double length,
double width, double height) {
       return length * width * height;
    }
}

public class Area {
    public static void main(String[] args) {
       AreaCalculator obj = new
AreaCalculator();
```

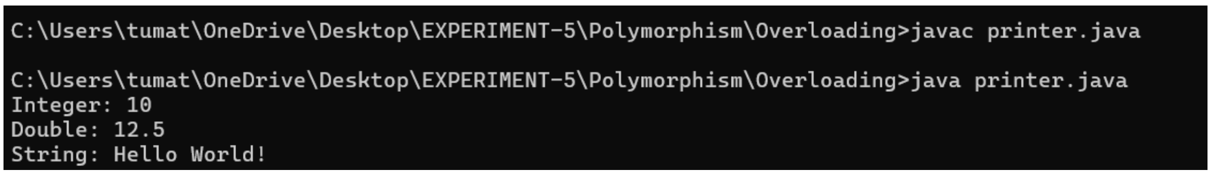
```
        System.out.println("Area of Square: "
+ obj.calculate(5));
        System.out.println("Area of
Rectangle: " + obj.calculate(5, 6));
        System.out.println("Volume of Box: "
+ obj.calculate(5, 3, 4));
    }
}
```

Output:



```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-5\Polymorphism\Overloading>javac Area.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-5\Polymorphism\Overloading>java Area.java
Area of Square: 25.0
Area of Rectangle: 30.0
Volume of Box: 60.0
```

10.b)Printer

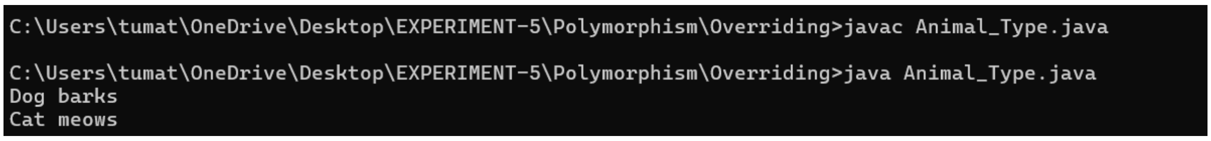


```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-5\Polymorphism\Overloading>javac printer.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-5\Polymorphism\Overloading>java printer.java
Integer: 10
Double: 12.5
String: Hello World!
```

11.Overriding

11.a)Animal_Type

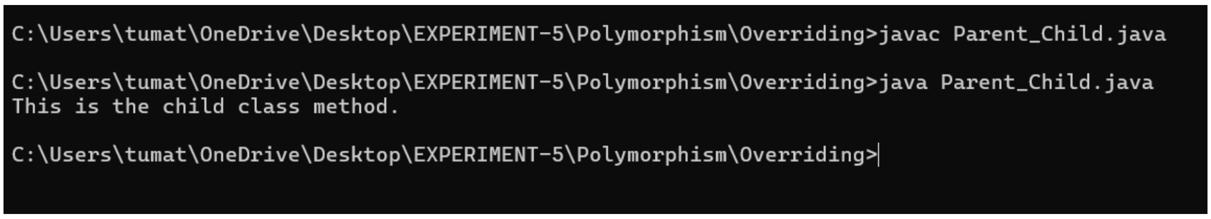
```
class Animal {
   void makeSound() {
      System.out.println("Animal makes a
sound");
   }
}

class Dog extends Animal {
   @Override
   void makeSound() {
      System.out.println("Dog barks");
   }
}

class Cat extends Animal {
    @Override
    void makeSound() {
        System.out.println("Cat meows");
    }
}

public class Animal_Type {
    public static void main(String[] args) {
        Animal a;

        a = new Dog();
        a.makeSound();
        a = new Cat();
        a.makeSound();
}
```
Output:


```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-5\Polymorphism\Overriding>javac Animal_Type.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-5\Polymorphism\Overriding>java Animal_Type.java
Dog barks
Cat meows
```

11.b)Parent_child

```
class Parent {
    void show() {
        System.out.println("This is the parent class method.");
    }
}

class Child extends Parent {
    @Override
    void show() {
        System.out.println("This is the child class method.");
    }
}

public class OverridingExample {
    public static void main(String[] args) {
        Parent obj = new Child();
     obj.show();
    }
}
```

Output:



```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-5\Polymorphism\Overriding>javac Parent_Child.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-5\Polymorphism\Overriding>java Parent_Child.java
This is the child class method.

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-5\Polymorphism\Overriding>
```

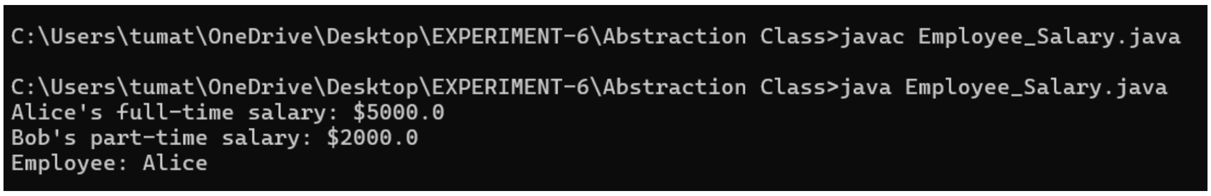
12.Abstract Class

12.a)Employee_Salary

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

    Employee(String name, double salary) {
        this.name = name;
        this.salary = salary;
    }
    abstract void calculateSalary();

    void showDetails() {
        System.out.println("Employee: " +
name);
    }
}
class FullTimeEmployee extends Employee
{
    FullTimeEmployee(String name, double
salary) {
        super(name, salary);
    }

    void calculateSalary() {
        System.out.println(name + "'s full-
time salary: $" + salary);
    }
}
class PartTimeEmployee extends
Employee {
    int hoursWorked;

    PartTimeEmployee(String name, double
salary, int hoursWorked) {
        super(name, salary);
        this.hoursWorked = hoursWorked;
    }
    void calculateSalary() {
        System.out.println(name + "'s part-
time salary: $" + (salary * hoursWorked));
    }
}

public class Employee_Salary {
    public static void main(String[] args) {
        Employee e1 = new
FullTimeEmployee("Alice", 5000);
        Employee e2 = new
PartTimeEmployee("Bob", 20, 100);

        e1.calculateSalary();
        e2.calculateSalary();
        e1.showDetails();
    }
```

}

Output:



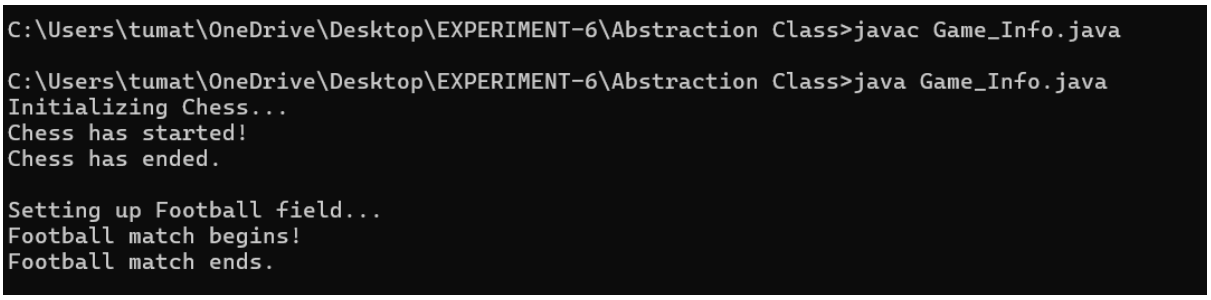
```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-6\Abstraction Class>javac Employee_Salary.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-6\Abstraction Class>java Employee_Salary.java
Alice's full-time salary: $5000.0
Bob's part-time salary: $2000.0
Employee: Alice
```

12.b)Game_Info

```
abstract class Game {
    abstract void initialize();
    abstract void start();
    abstract void end();

      public final void play() {
        initialize();
        start();
        end();
    }
```

```
}

class Chess extends Game {
    @Override
    void initialize() {
        System.out.println("Initializing
Chess...");
    }

    @Override
    void start() {
        System.out.println("Chess has
started!");
    }

    @Override
    void end() {
        System.out.println("Chess has
ended.");
    }
}

class Football extends Game {
    @Override
    void initialize() {
        System.out.println("Setting up
Football field...");
    }


    @Override
    void start() {
        System.out.println("Football match
begins!");
    }
    @Override
    void end() {
        System.out.println("Football match
ends.");
    }
}

public class Game_Info {
    public static void main(String[] args) {
        Game chess = new Chess();
        chess.play();

        System.out.println();

        Game football = new Football();
        football.play();
    }
}
```

Output:



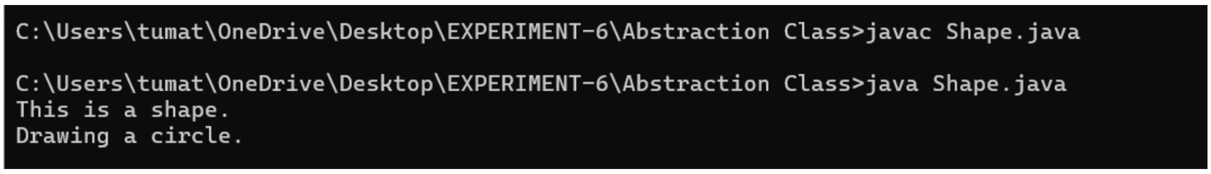
```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-6\Abstraction Class>javac Game_Info.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-6\Abstraction Class>java Game_Info.java
Initializing Chess...
Chess has started!
Chess has ended.

Setting up Football field...
Football match begins!
Football match ends.
```

12.c)Shape

```
abstract class main {
    abstract void draw();

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

```
class Circle extends main {
    @Override
    void draw() {
        System.out.println("Drawing a
circle.");
    }
}

public class Shape {
    public static void main(String[] args) {
        main myshape = new Circle();
        myShape.display();
        myShape.draw();
    }
}
```

Output:



```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-6\Abstraction Class>javac Shape.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-6\Abstraction Class>java Shape.java
This is a shape.
Drawing a circle.
```

12.d)Vehical_Type

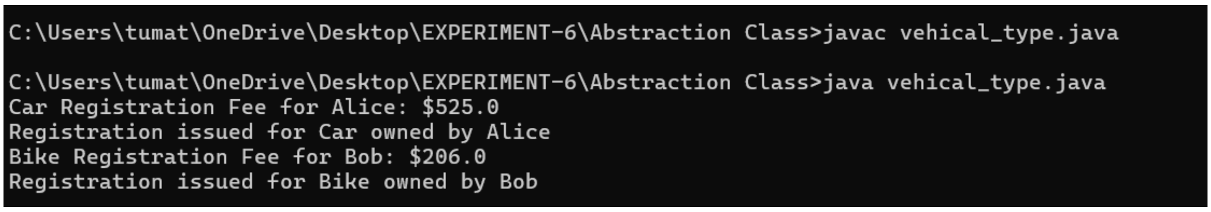
```
abstract class VehicleRegistration {
   String vehicleType;
   String owner;
   double baseFee;


   VehicleRegistration(String vehicleType,
String owner, double baseFee) {
       this.vehicleType = vehicleType;
       this.owner = owner;
       this.baseFee = baseFee;
   }
```

```
    abstract void calculateRegistrationFee();

    void issueRegistration() {
        System.out.println("Registration
issued for " + vehicleType + " owned by " +
owner);
    }
}
class CarRegistration extends
VehicleRegistration {
    CarRegistration(String owner, double
baseFee) {
        super("Car", owner, baseFee);
    }
    void calculateRegistrationFee() {
        double totalFee = baseFee + (baseFee
* 0.05);
```

```
        System.out.println("Car Registration
Fee for " + owner + ": $" + totalFee);
    }
}
class BikeRegistration extends
VehicleRegistration {
    BikeRegistration(String owner, double
baseFee) {
        super("Bike", owner, baseFee);
    }
    void calculateRegistrationFee() {
        double totalFee = baseFee + (baseFee
* 0.03);
        System.out.println("Bike Registration
Fee for " + owner + ": $" + totalFee);
    }
}
```

```
public class vehical_type {
    public static void main(String[] args) {
        VehicleRegistration v1 = new
CarRegistration("Alice", 500);
        VehicleRegistration v2 = new
BikeRegistration("Bob", 200);


        v1.calculateRegistrationFee();
        v1.issueRegistration();


        v2.calculateRegistrationFee();
        v2.issueRegistration();
    }
}
```

Output:

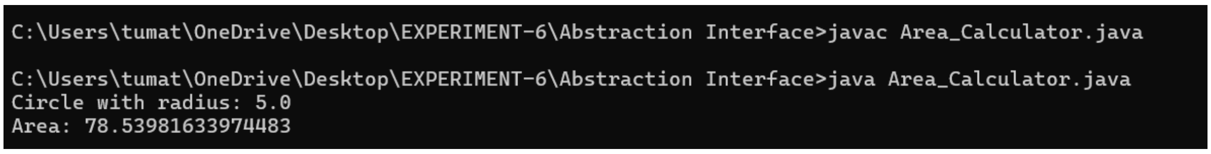


```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-6\Abstraction Class>javac vehical_type.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-6\Abstraction Class>java vehical_type.java
Car Registration Fee for Alice: $525.0
Registration issued for Car owned by Alice
Bike Registration Fee for Bob: $206.0
Registration issued for Bike owned by Bob
```

13.Abstract_Interface

13.a)Area_Caluculator

```
interface Shape {
    double area();
}

abstract class AbstractShape implements
Shape {
    double dimension;

    AbstractShape(double dimension) {
        this.dimension = dimension;
    }

    abstract void display();
}

class Circle extends AbstractShape {
    Circle(double radius) {
        super(radius);
    }

    public double area() {
        return Math.PI * dimension *
dimension;
    }

    void display() {
        System.out.println("Circle with radius: " + dimension);
    }
}

public class Area_Calculator {
    public static void main(String[] args) {
        Circle c = new Circle(5);
        c.display();
        System.out.println("Area: " + c.area());
    }
}
```



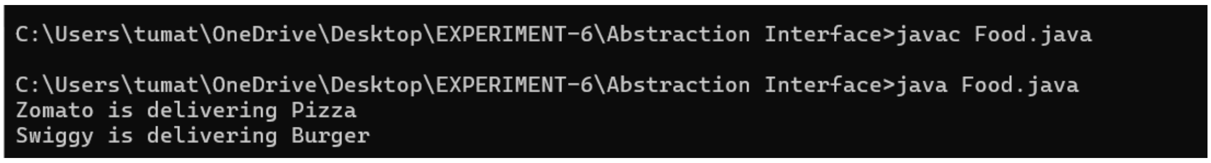
```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-6\Abstraction Interface>javac Area_Calculator.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-6\Abstraction Interface>java Area_Calculator.java
Circle with radius: 5.0
Area: 78.53981633974483
```

13.b)Food

```
interface FoodDelivery {
   void deliverOrder(String foodItem);
}
class Zomato implements FoodDelivery {
   public void deliverOrder(String
foodItem) {
      System.out.println("Zomato is
delivering " + foodItem);
   }
}
class Swiggy implements FoodDelivery {
   public void deliverOrder(String
foodItem) {
      System.out.println("Swiggy is
delivering " + foodItem);
   }
}
```

```
public class Food {
    public static void main(String[] args) {
        FoodDelivery f1 = new Zomato();
        FoodDelivery f2 = new Swiggy();

        f1.deliverOrder("Pizza");
        f2.deliverOrder("Burger");
    }
}
```

Output:

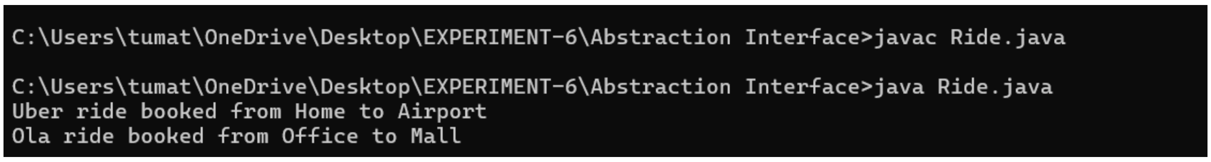


```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-6\Abstraction Interface>javac Food.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-6\Abstraction Interface>java Food.java
Zomato is delivering Pizza
Swiggy is delivering Burger
```

13.c)Ride

```
interface RideService {
    void bookRide(String pickup, String destination);
}
class Uber implements RideService {
    @Override
    public void bookRide(String pickup, String destination) {
        System.out.println("Uber ride booked from " + pickup + " to " + destination);
    }
}
class Ola implements RideService {
    @Override
    public void bookRide(String pickup, String destination) {
        System.out.println("Ola ride booked from " + pickup + " to " + destination);
    }
}
public class Ride {
    public static void main(String[] args) {
        RideService r1 = new Uber();
        RideService r2 = new Ola();

        r1.bookRide("Home", "Airport");
        r2.bookRide("Office", "Mall");
    }
}
```

Output:



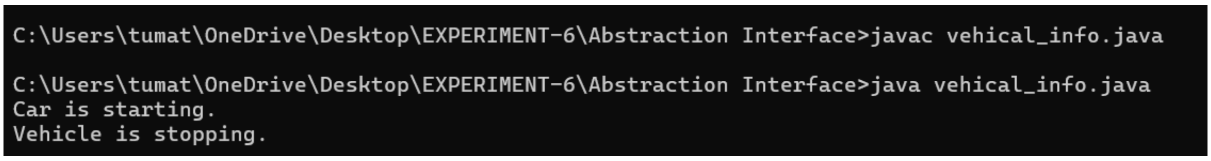
```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-6\Abstraction Interface>javac Ride.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-6\Abstraction Interface>java Ride.java
Uber ride booked from Home to Airport
Ola ride booked from Office to Mall
```

13.d)Vehical_Info

```
interface Vehicle {
    void start();

    default void stop() {
        System.out.println("Vehicle is
stopping.");
    }
}

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

```
public class vehical_info {
    public static void main(String[] args) {
        Car myCar = new Car();
        myCar.start(); // Output: Car is
starting.
        myCar.stop();  // Output: Vehicle is
stopping.
    }
}
```

Output:



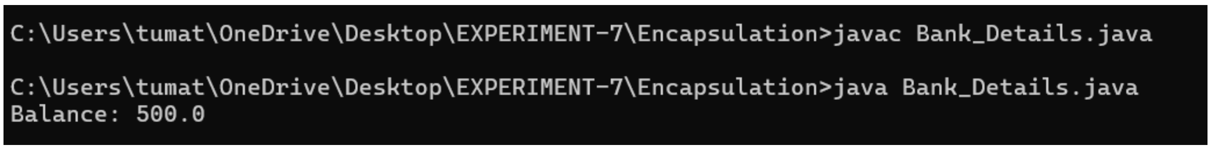
```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-6\Abstraction Interface>javac vehical_info.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-6\Abstraction Interface>java vehical_info.java
Car is starting.
Vehicle is stopping.
```

14.Encapsulation

14.a)Bank_Details

```
class BankAccount {
    private double balance;

    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
        }
    }

    public double getBalance() {
        return balance;
    }
}

public class Bank_Details {
    public static void main(String[] args) {
        BankAccount account = new
BankAccount();
        account.deposit(500);
        System.out.println("Balance: " +
account.getBalance());
    }
}
```

Output:



```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-7\Encapsulation>javac Bank_Details.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-7\Encapsulation>java Bank_Details.java
Balance: 500.0
```

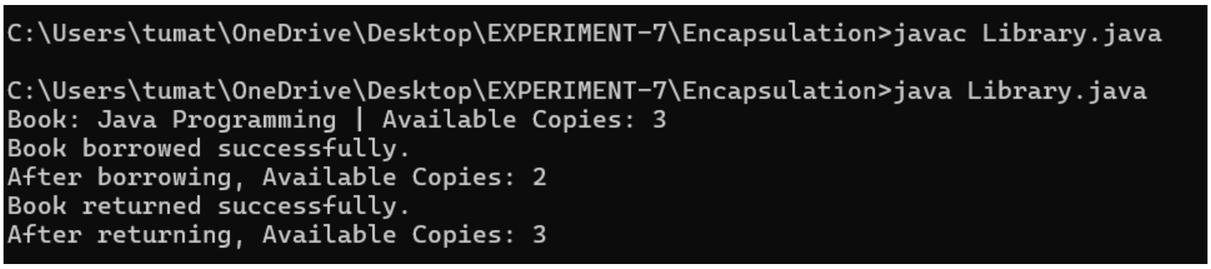
14.b)Library

```
class Book {
    private String title;
    private String author;
    private String ISBN;
    private int availableCopies;
    public Book(String title, String author,
String ISBN, int availableCopies) {
        this.title = title;
        this.author = author;
        this.ISBN = ISBN;
        this.availableCopies = availableCopies;
    }

    public String getTitle() {
        return title;
    }

    public String getAuthor() {
        return author;
    }
    public String getISBN() {
        return ISBN;
    }

    public int getAvailableCopies() {
        return availableCopies;
    }

    public void borrowBook() {
        if (availableCopies > 0) {
            availableCopies--;
            System.out.println("Book borrowed
successfully.");
        } else {
            System.out.println("Sorry, this book
is not available.");
```

```
        }
    }

    public void returnBook() {
        availableCopies++;
        System.out.println("Book returned
successfully.");
    }
}

public class Library {
    public static void main(String[] args) {
        Book book1 = new Book("Java
Programming", "James Gosling",
"123456", 3);
```

```
        System.out.println("Book: " +
book1.getTitle() + " | Available Copies: " +
book1.getAvailableCopies());

        book1.borrowBook();
        System.out.println("After borrowing,
Available Copies: " +
book1.getAvailableCopies());

        book1.returnBook();
        System.out.println("After returning,
Available Copies: " +
book1.getAvailableCopies());
    }
}
```

Output:

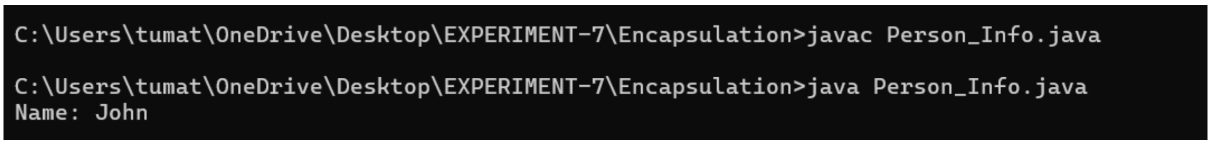


```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-7\Encapsulation>javac Library.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-7\Encapsulation>java Library.java
Book: Java Programming | Available Copies: 3
Book borrowed successfully.
After borrowing, Available Copies: 2
Book returned successfully.
After returning, Available Copies: 3
```

14.c)Person_Info

```
class Person {
    private String name;


    public String getName() {
        return name;
    }


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


public class Person_Info {
    public static void main(String[] args) {
        Person p = new Person();
        p.setName("John");
        System.out.println("Name: " +
p.getName());
    }
}
```

Output:

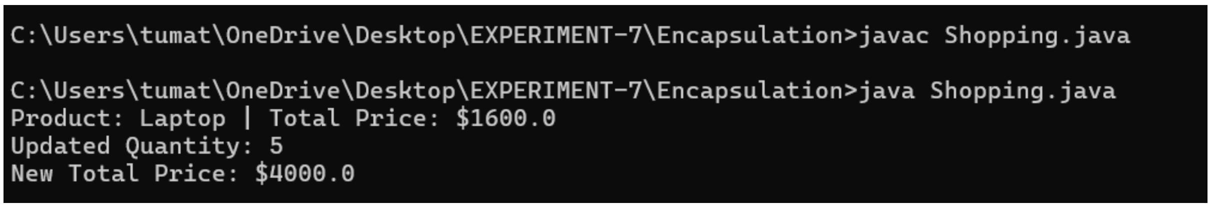


```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-7\Encapsulation>javac Person_Info.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-7\Encapsulation>java Person_Info.java
Name: John
```

14.d)Shopping

```
class Product {
    private String productName;
    private double price;
    private int quantity;
    public Product(String productName,
double price, int quantity) {
        this.productName = productName;
        this.price = price;
        this.quantity = quantity;
    }

    public String getProductName() {
        return productName;
    }

    public double getPrice() {
        return price;
    }

    public int getQuantity() {
        return quantity;
    }

    public void addStock(int amount) {
        if (amount > 0) {
            quantity += amount;
        }
    }

    public double calculateTotalPrice() {
        return price * quantity;
    }
}

public class Shopping {
    public static void main(String[] args) {
        Product product1 = new
Product("Laptop", 800, 2);
        System.out.println("Product: " +
product1.getProductName() + " | Total
Price: $" + product1.calculateTotalPrice());

        product1.addStock(3);
        System.out.println("Updated
Quantity: " + product1.getQuantity());
        System.out.println("New Total Price:
$" + product1.calculateTotalPrice());
    }
}
```

Output:



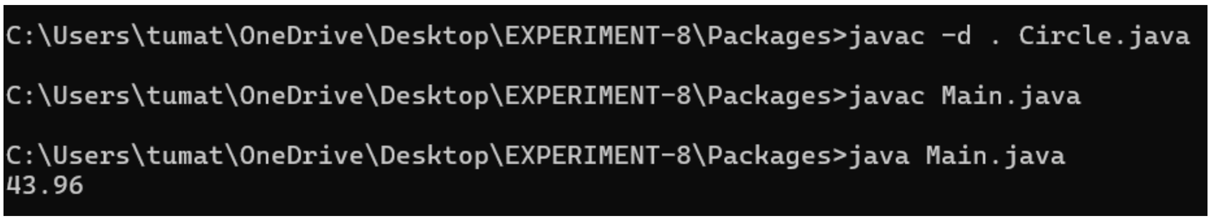
```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-7\Encapsulation>javac Shopping.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-7\Encapsulation>java Shopping.java
Product: Laptop | Total Price: $1600.0
Updated Quantity: 5
New Total Price: $4000.0
```

15.Packages

15.a)User Defined Packages

```
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();
}
```

}

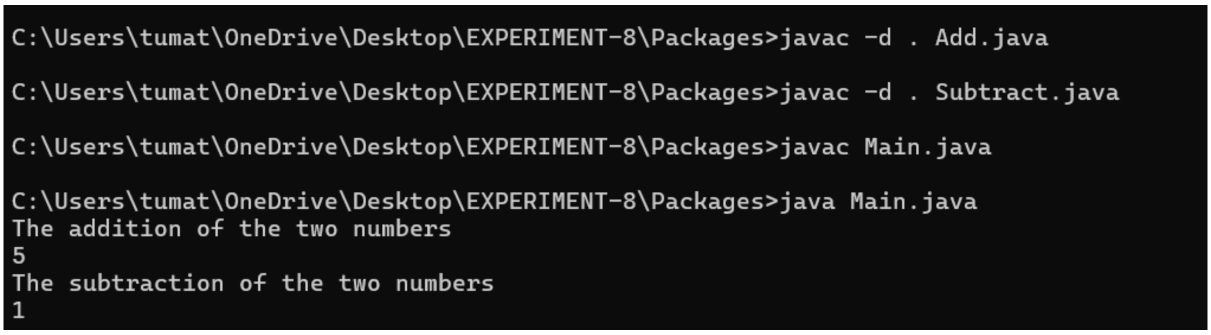


```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-8\Packages>javac -d . Circle.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-8\Packages>javac Main.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-8\Packages>java Main.java
43.96
```

15.b)user Defined Packages

```
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());
}
}
```



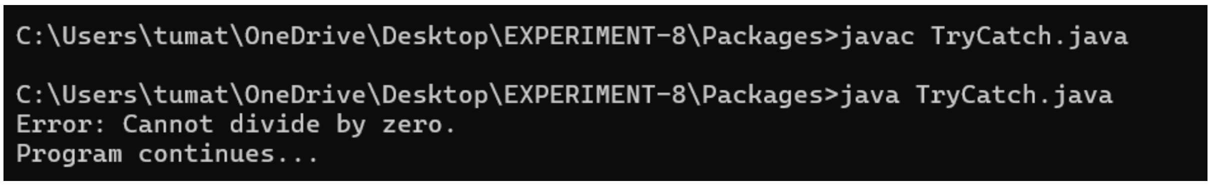
```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-8\Packages>javac -d . Add.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-8\Packages>javac -d . Subtract.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-8\Packages>javac Main.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-8\Packages>java Main.java
The addition of the two numbers
5
The subtraction of the two numbers
1
```

15.c)Built-in Packages

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

15.d)Built-in Packages

```
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...");
    }
}
```



```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-8\Packages>javac TryCatch.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-8\Packages>java TryCatch.java
Error: Cannot divide by zero.
Program continues...
```

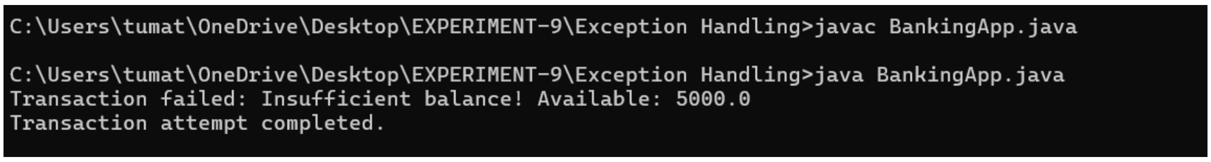
16.Exception Handling

16.a)Banking App

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


class BankAccount {
    private double balance = 5000;


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

```
public class BankingApp {
    public static void main(String[] args) {
        BankAccount account = new
BankAccount();
        try {
            account.withdraw(6000); // Will
throw exception
        } catch (InsufficientFundsException e)
{
            System.out.println("Transaction
failed: " + e.getMessage());
        } finally {
            System.out.println("Transaction
attempt completed.");
        }
    }
}
```

Output:

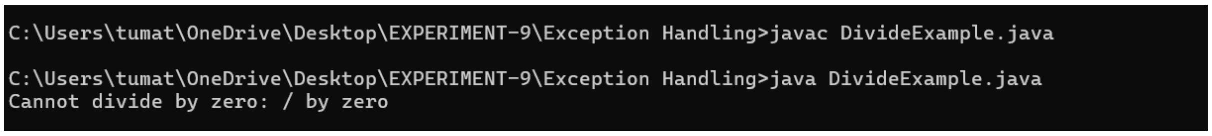


```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-9\Exception Handling>javac BankingApp.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-9\Exception Handling>java BankingApp.java
Transaction failed: Insufficient balance! Available: 5000.0
Transaction attempt completed.
```

16.b)DivideExample

```
import java.io.*;


public class FileExample {
    public static void main(String[] args) {
        try {
            FileReader reader = new
FileReader("nonexistentfile.txt");
            int data = reader.read();
            while (data != -1) {
                System.out.print((char) data);
                data = reader.read();
            }
            reader.close();
        } catch (FileNotFoundException e) {
            System.out.println("File not
found!");
        } catch (IOException e) {
            System.out.println("An error
occurred while reading the file.");
        }
    }
}
```

Output:

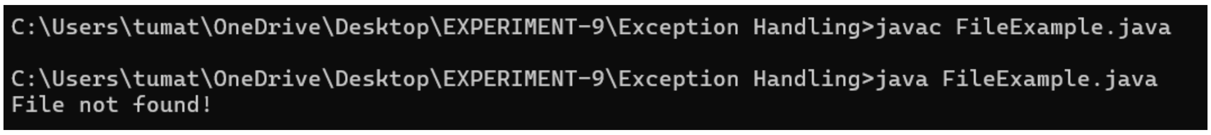


```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-9\Exception Handling>javac DivideExample.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-9\Exception Handling>java DivideExample.java
Cannot divide by zero: / by zero
```

16.c)FileExample

```
import java.io.*;


public class FileExample {
    public static void main(String[] args) {
        try {
            FileReader reader = new
FileReader("nonexistentfile.txt");
            int data = reader.read();
            while (data != -1) {
                System.out.print((char) data);
                data = reader.read();
            }
            reader.close();
        } catch (FileNotFoundException e) {
            System.out.println("File not
found!");
        } catch (IOException e) {
            System.out.println("An error
occurred while reading the file.");
        }
    }
}
```

Output:

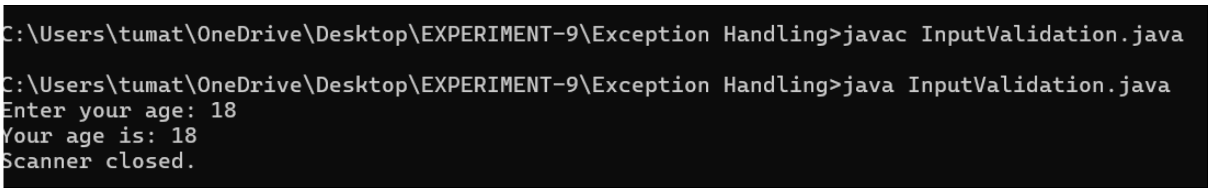


```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-9\Exception Handling>javac FileExample.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-9\Exception Handling>java FileExample.java
File not found!
```

16.d)InputValidation

```
import java.util.InputMismatchException;
import java.util.Scanner;


public class InputValidation {
    public static void main(String[] args) {
        Scanner scanner = new
Scanner(System.in);
        int age = 0;


        try {
            System.out.print("Enter your age: ");
            age = scanner.nextInt();

            if (age < 0) {
                throw new IllegalArgumentException("Age cannot be negative.");
            }

            System.out.println("Your age is: " + age);

        } catch (InputMismatchException e) {
            System.out.println("Invalid input! Please enter a number.");
        } catch (IllegalArgumentException e) {
            System.out.println("Validation
error: " + e.getMessage());
        } finally {
            scanner.close();
            System.out.println("Scanner
closed.");
        }
    }
}
```

Output:



```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-9\Exception Handling>javac InputValidation.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-9\Exception Handling>java InputValidation.java
Enter your age: 18
Your age is: 18
Scanner closed.
```

17.File Handling

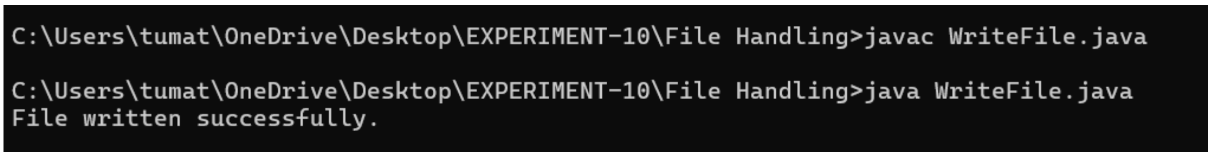
17.a)WriteFile

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

```
public class WriteFile {
    public static void main(String[] args) {
        try {
            FileWriter writer = new
FileWriter("output.txt");
            writer.write("Hello, this is a file
write example in Java!\n");
            writer.write("This is the second
line.");
            writer.close();
            System.out.println("File written
successfully.");
        } catch (IOException e) {
            System.out.println("An error
occurred during writing: " +
e.getMessage());
```

```
        }
    }
}
```

Output:



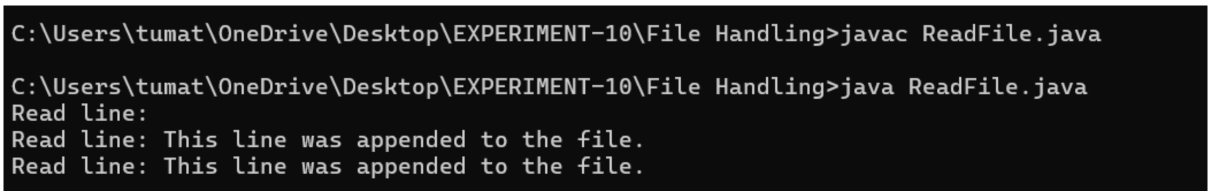
```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-10\File Handling>javac WriteFile.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-10\File Handling>java WriteFile.java
File written successfully.
```

17.b)ReadFile

```
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;


public class ReadFile {
    public static void main(String[] args) {
        try {
            BufferedReader reader = new
BufferedReader(new
FileReader("output.txt"));
```

```
            String line;
            while ((line = reader.readLine()) !=
null) {
                System.out.println("Read line: " +
line);
            }
            reader.close();
        } catch (IOException e) {
            System.out.println("Error reading
the file: " + e.getMessage());
        }
    }
}
```

Output:



```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-10\File Handling>javac ReadFile.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-10\File Handling>java ReadFile.java
Read line:
Read line: This line was appended to the file.
Read line: This line was appended to the file.
```

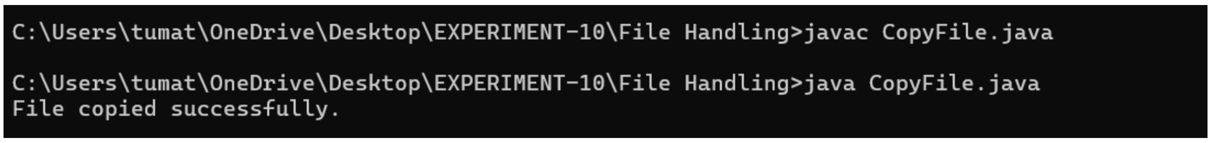
17.c)CopyFile

```
import java.io.*;

public class CopyFile {
    public static void main(String[] args) {
        try {
            File inputFile = new
File("output.txt");
            File outputFile = new
File("copy_output.txt");

            BufferedReader reader = new
BufferedReader(new
FileReader(inputFile));
            BufferedWriter writer = new
BufferedWriter(new
FileWriter(outputFile));
```

```
            String line;
            while ((line = reader.readLine()) !=
null) {
                writer.write(line);
                writer.newLine();
            }

            reader.close();
            writer.close();

            System.out.println("File copied
successfully.");
        } catch (IOException e) {
            System.out.println("Error during file
copy: " + e.getMessage());
        }
    }
```

}

Output:

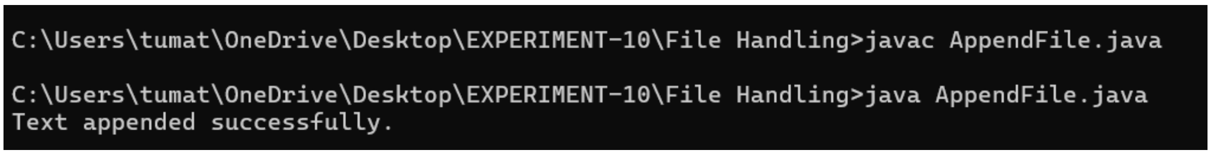


```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-10\File Handling>javac CopyFile.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-10\File Handling>java CopyFile.java
File copied successfully.
```

17.d)AppendFile

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


public class AppendFile{
    public static void main(String[] args) {
        try {
            FileWriter writer = new FileWriter("output.txt", true); // `true` = append mode
            writer.write("\nThis line was appended to the file.");
            writer.close();
            System.out.println("Text appended successfully.");
        } catch (IOException e) {
            System.out.println("Error appending to the file: " + e.getMessage());
        }
    }
}
```

Output:



```
C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-10\File Handling>javac AppendFile.java

C:\Users\tumat\OneDrive\Desktop\EXPERIMENT-10\File Handling>java AppendFile.java
Text appended successfully.
```