



SCHOOL OF
COMPUTING

LAB RECORD

23CSE111- Object Oriented Programming

Submitted by

CH.SC.U4CSE24156 -Chanatati Sai Purnisha Mahi

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND
ENGINEERING

AMRITA VISHWA VIDYAPEETHAM
AMRITA SCHOOL OF COMPUTING

CHENNAI

March - 2025



**SCHOOL OF
COMPUTING**

**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.U4CSE24156 – CHANTATI SAI PURNISHA MAHI** 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

S.NO	TITLE	PAGE.NO
	UML DIAGRAM	
1.	RESTAURENT MANAGEMENT	
	1.a)Use Case Diagram	6
	1.b)Class Diagram	7
	1.c) Sequence Diagram	8
	1.d) Object Diagram	8
	1.e) State-activity Diagram	9
2.	BANKING MANAGEMENT	
	2.a) Use Case Diagram	10
	2.b) Class Diagram	11
	2.c) Sequence Diagram	12
	2.d) Object Daigram	13
	2.e) State-Activity Diagram	13
3.	BASIC JAVA PROGRAMS	
	3.a) CountOfDigits	14
	3.b) EvenOddCheck	15
	3.c) Factorial	15
	3.d) Largestnumber	16
	3.e) LCM	17
	3.f) LeapYearecheck	18
	3.g) Palindrome Check	19
	3.h) Mulitiplication table	20
	3.i) Poweroffnumber	20
	3.j) SumofDigits	21
	INHERITANCE	
4.	SINGLE INHERITANCE PROGRAMS	
	4.a) Teacher details	23

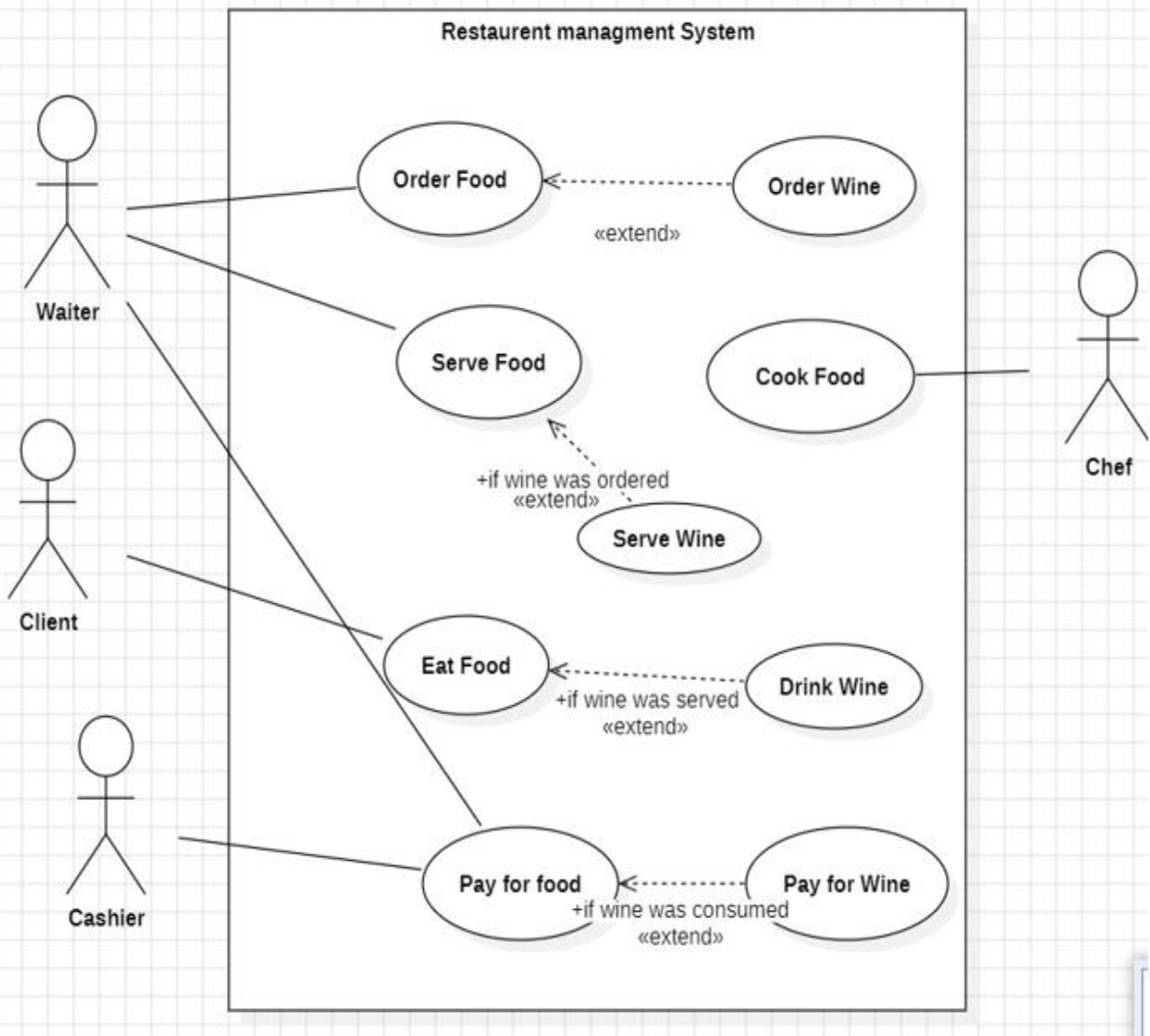
	4.b) Bank Account	24
5.	MULTILEVEL INHERITANCE PROGRAMS	
	5.a) Vehicle	25
	5.b) Electronic devices	26
6.	HIERARCHICAL INHERITANCE PROGRAMS	
	6.a) Shapes	28
	6.b) Vehicles	29
7.	HYBRID INHERITANCE PROGRAMS	
	7.a) Employee Details	30
	7.b) Bank account	32
	POLYMORPHISM	
8.	CONSTRUCTOR PROGRAMS	
	8.a) Book	34
9.	CONSTRUCTOR OVERLOADING PROGRAMS	
	9.a) Student Details	35
10.	METHOD OVERLOADING PROGRAMS	
	10.a) Calculator	37
	10.b) Employee details	38
11.	METHOD OVERRIDING PROGRAMS	
	11.a) Bank Deposit	39
	11.b) Payment	40
	ABSTRACTION	
12.	ABSTRACT CLASS PROGRAMS	
	12.a) Vehicle	42
	12.b) Printer	43
	12.c) Display	45
	12.d) Shape	47
13.	INTERFACE PROGRAMS	
	13.a) Electronic applications	50
	13.b) Bank Account	51
	13.c) Office Details	52
	13.d) Electronic Devices Prices	53
	ENCAPSULATION	
14.	ENCAPSULATION PROGRAMS	
	14.a) Laptop Details	54
	14.b) Movie Booking details	56
	14.c) Flight Booking details	57
	14.d) Temperature	58
15.	PACKAGES PROGRAMS	

	15.a)User Defined Packages	60
	15.b)User Defined Packages	62
	15.c)Built – in Package(3 Packages)	64
	15.d)Built – in Package(3 Packages)	66
16.	EXCEPTION HANDLING PROGRAMS	
	16.a) Number Format	67
	16.b)Custom Exception	68
	16.c) Input mismatch	69
	16.d) Using Finally word	70
17.	FILE HANDLING PROGRAMS	
	17.a) Create a file	71
	17.b) Write matter to that files	72
	17.c) Read File	73
	17.d) Append To File	74

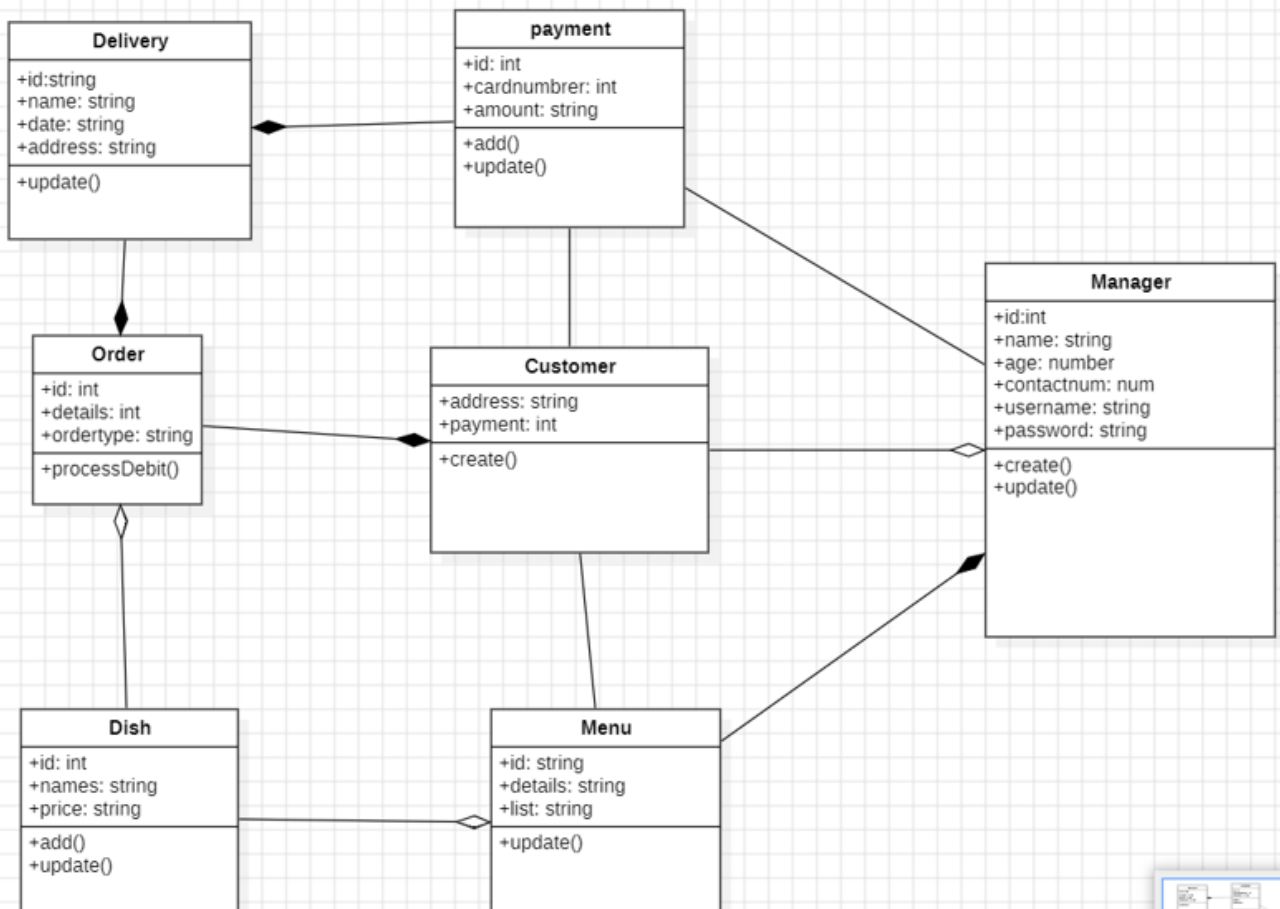
UML DIAGRAMS

1.RESTAURENT MANAGEMENT

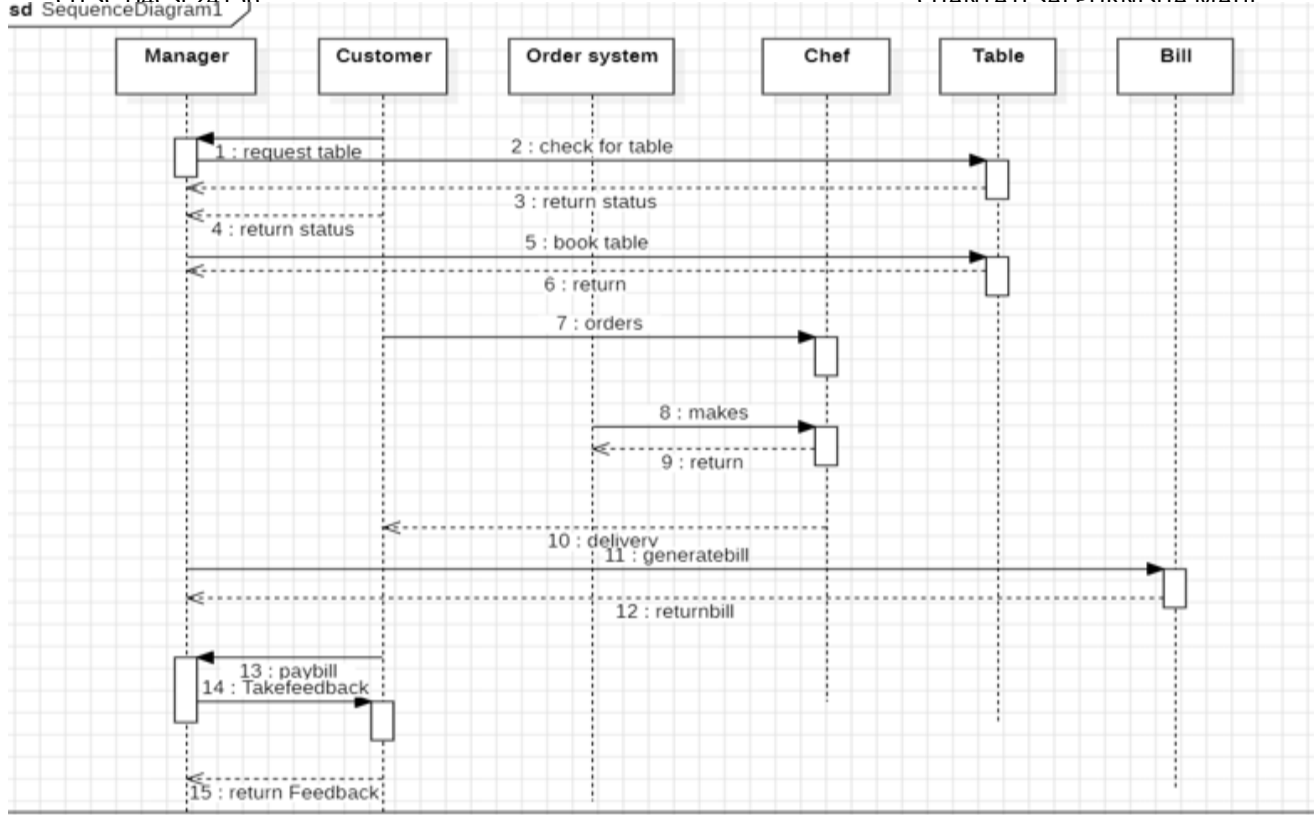
1.a) Use Case Diagram:



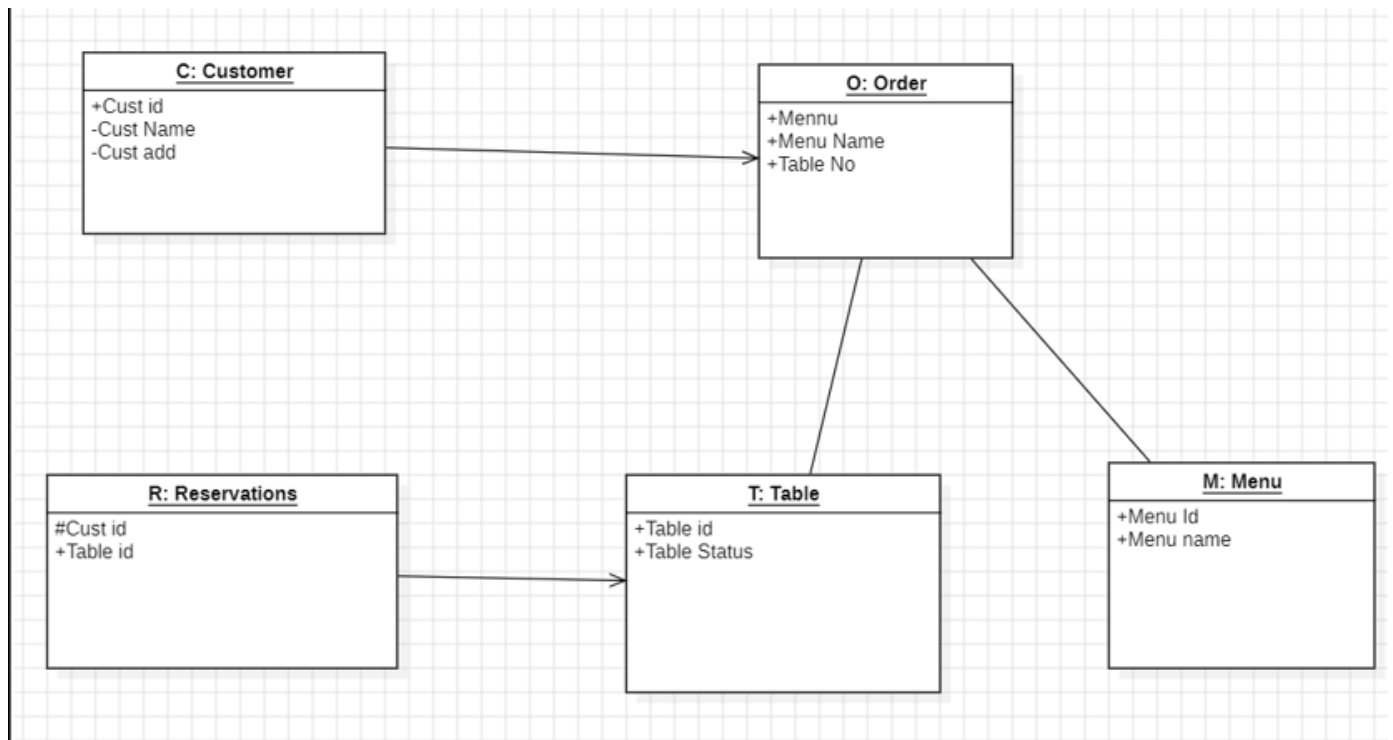
1b)Class Diagram:



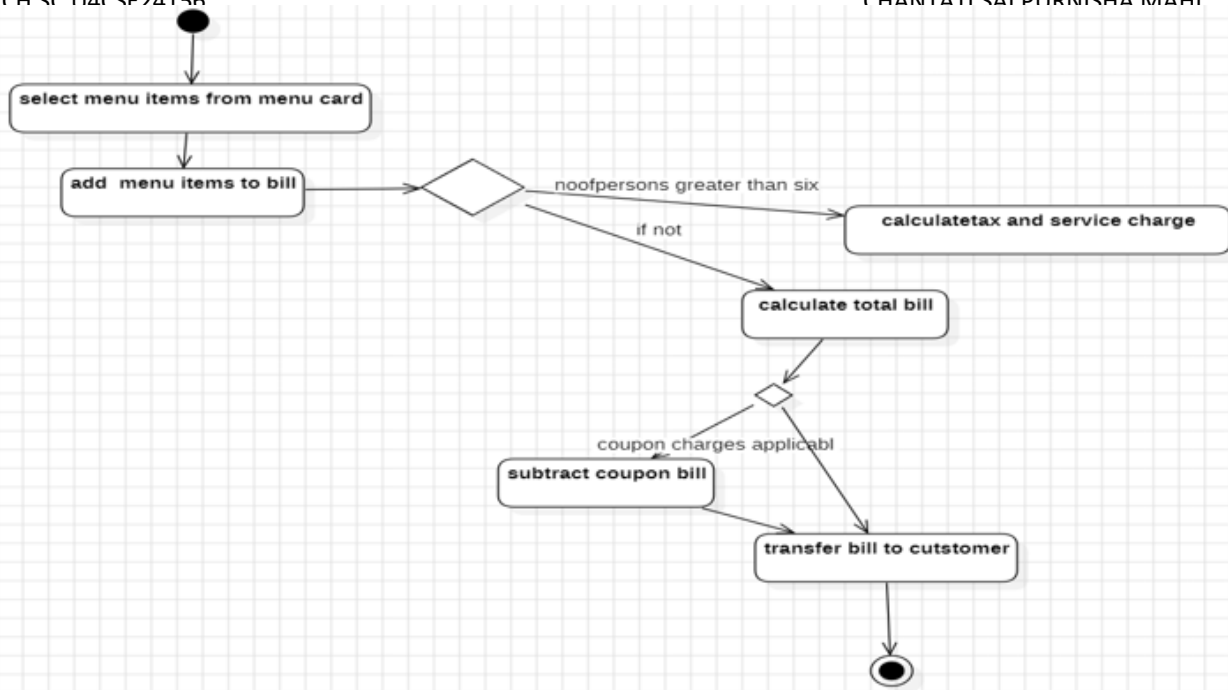
1c) Sequence Diagram:



1d)Object Diagram:

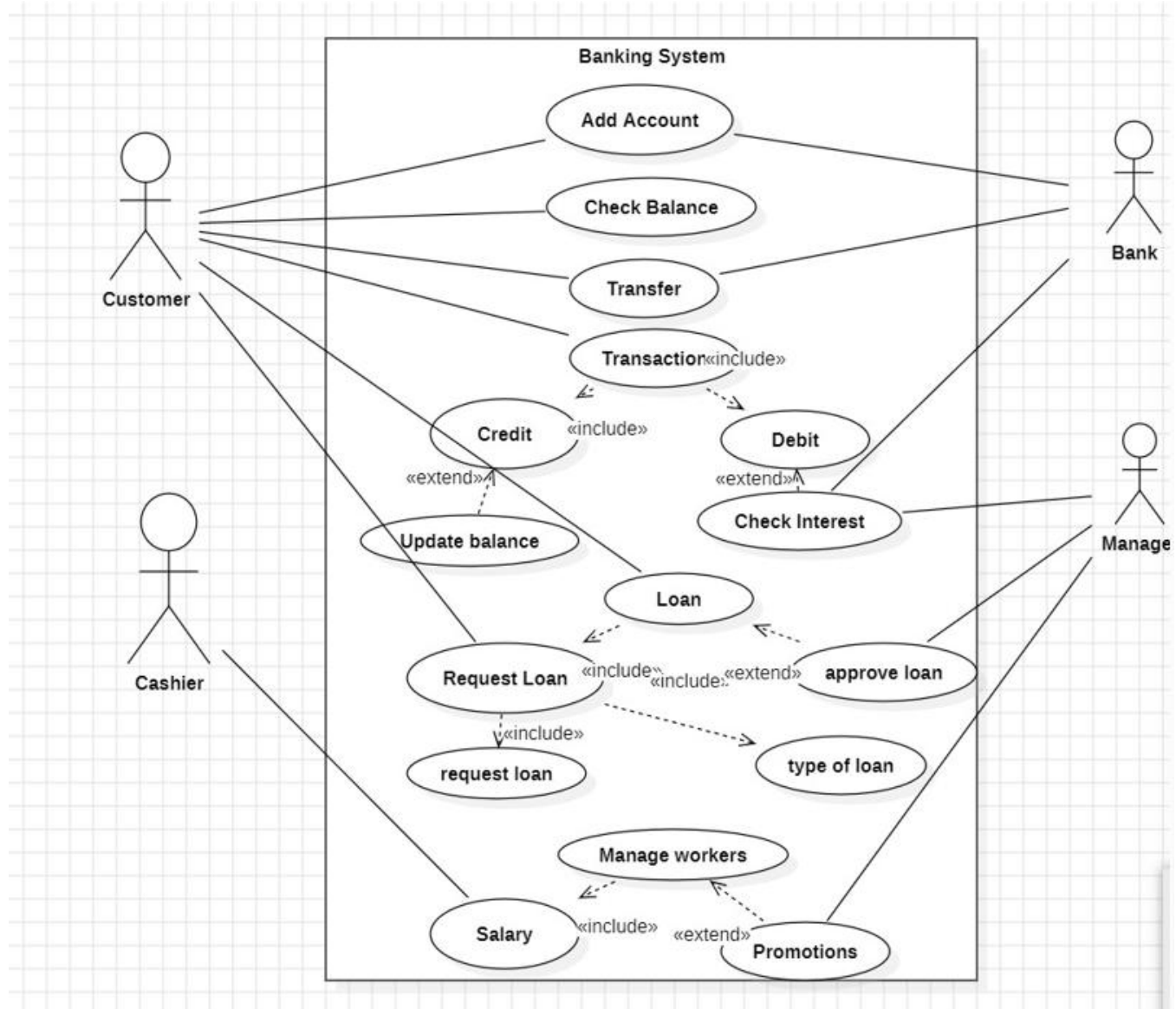


1e) State-Activity Diagram:

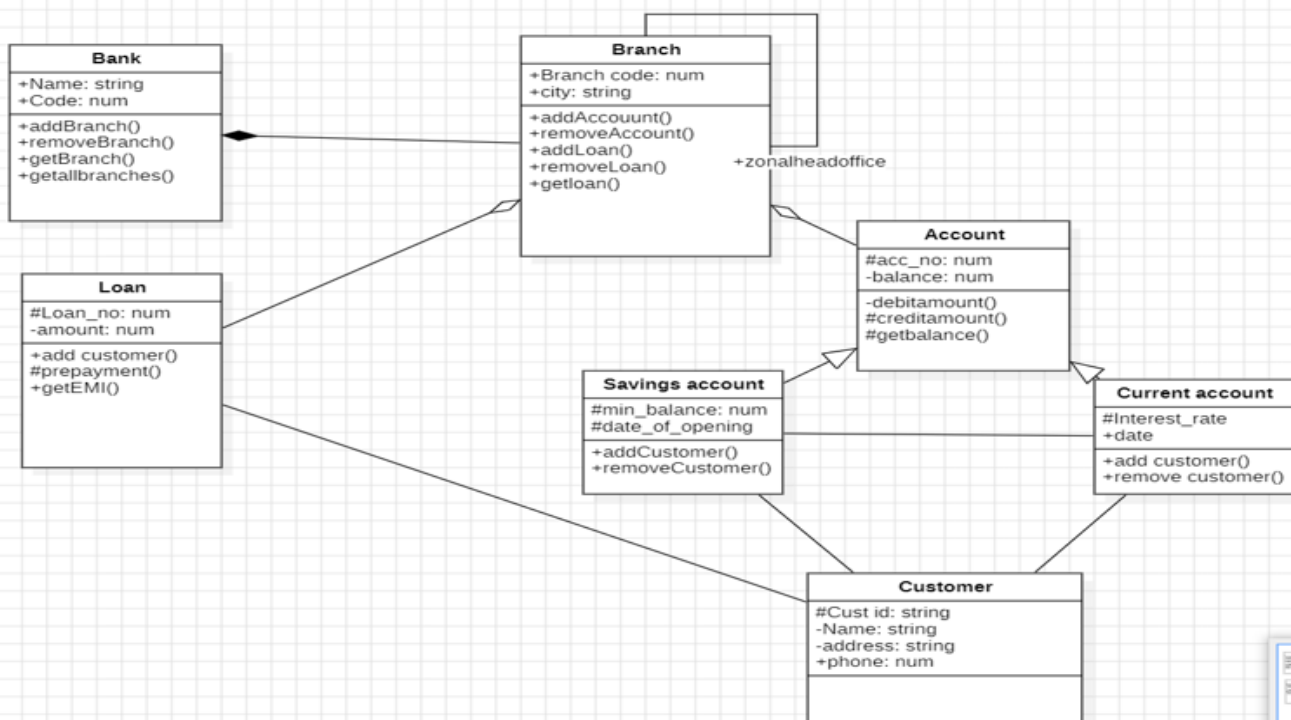


2. BANKING MANAGEMENT SYSTEM

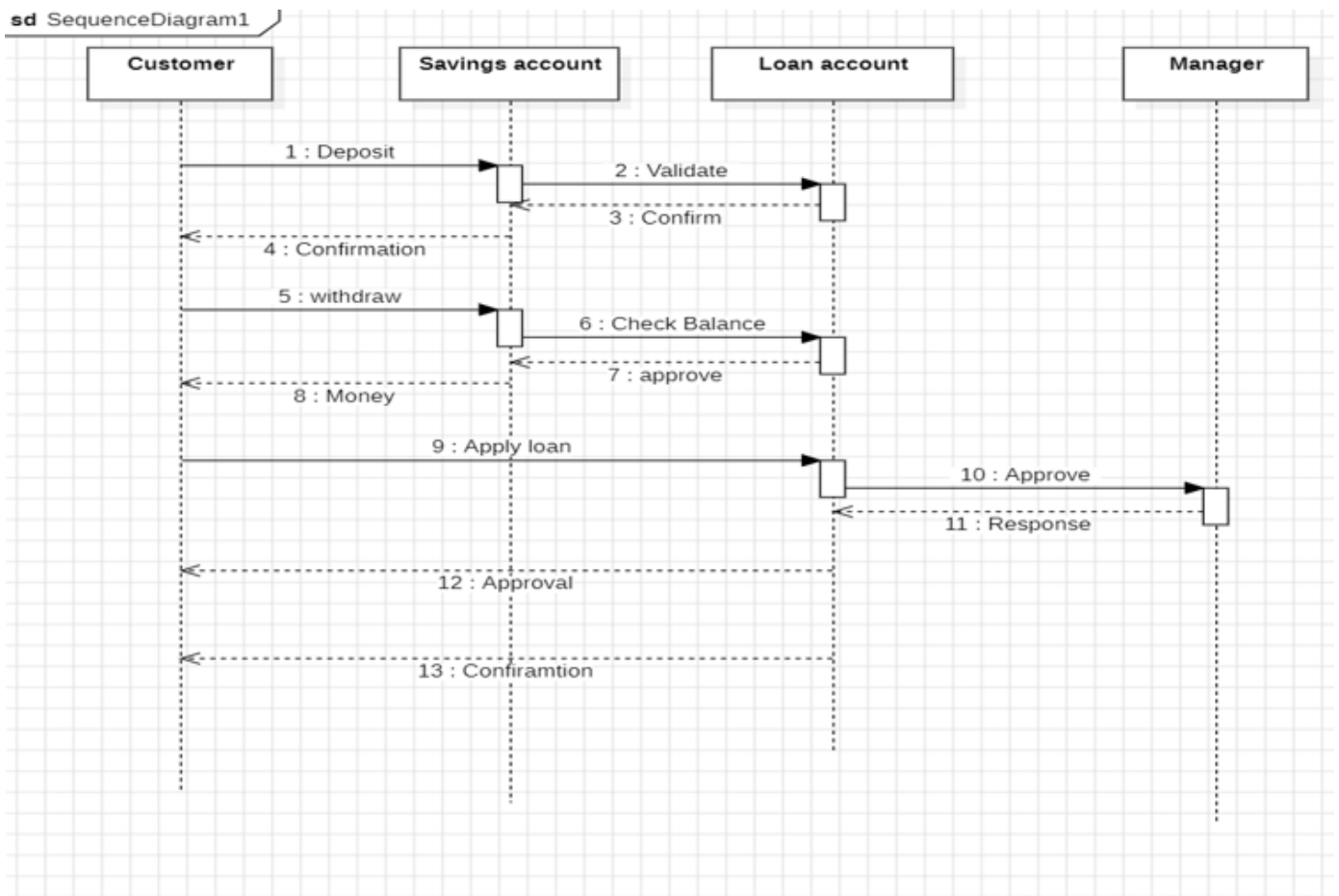
2a) Use Case Diagram:



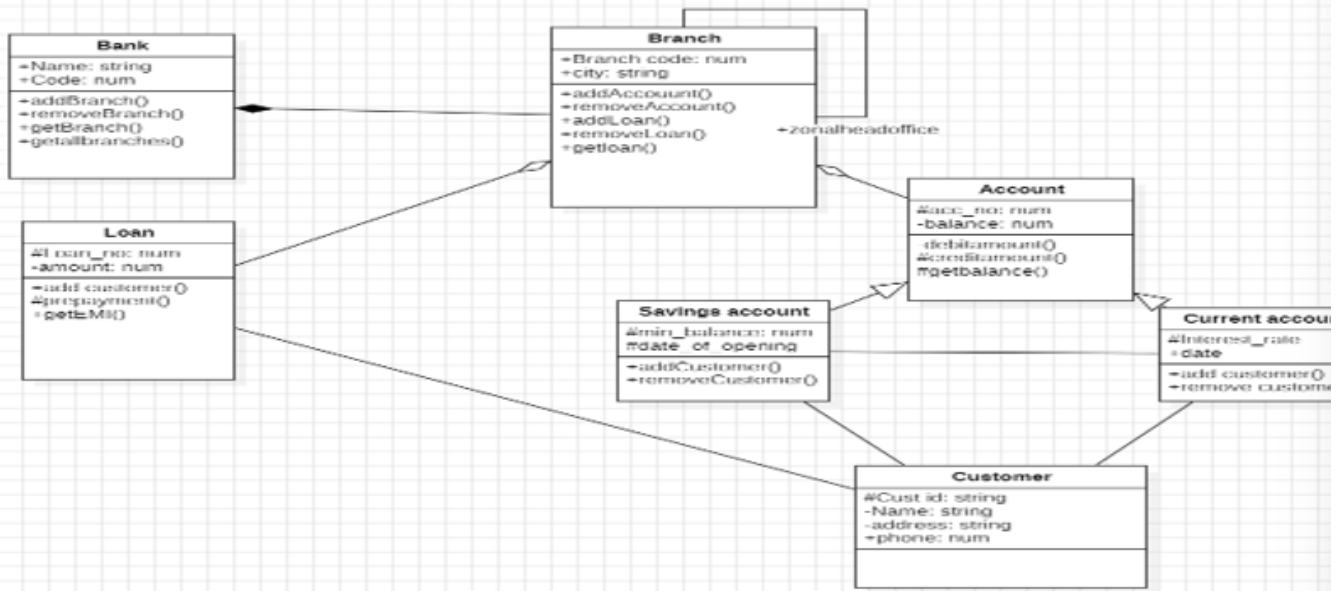
2b) Class Diagram:



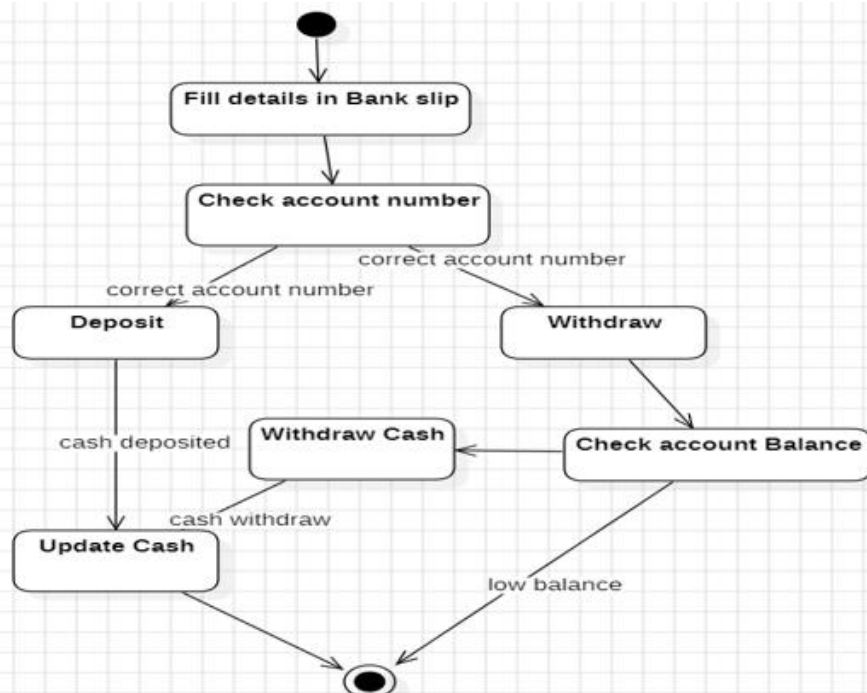
2c) Sequence Diagram:



2d) Object Diagram:



2e) State-Activity Diagram:



BASIC JAVA PROGRAMMES

3a) 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();

        int count = 0; while
        (num > 0) {
            num /= 10;
            count++;
        }

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

OUTPUT:

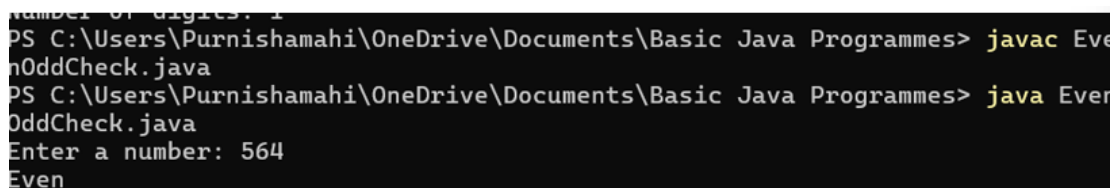
```
PS C:\Users\Purnishamahi\OneDrive\Documents\Basic Java Programmes> javac CountDigits.java
PS C:\Users\Purnishamahi\OneDrive\Documents\Basic Java Programmes> java CountDigits.java
Enter a number: 3
Number of digits: 1
```

3b)EvenOddCheck:

CODE:

```
import java.util.Scanner; public
class EvenOddCheck {
    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");
        } else {
          System.out.println("Odd");
        }
    }
}
```

OUTPUT:



```
PS C:\Users\Purnishamahi\OneDrive\Documents\Basic Java Programmes> javac EvenOddCheck.java
PS C:\Users\Purnishamahi\OneDrive\Documents\Basic Java Programmes> java EvenOddCheck.java
Enter a number: 564
Even
```

3c) 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();

      int fact = 1;
      int i = num;
```

```

        while (i > 0) {
            fact *= i;
            i--;
        }

        System.out.println("Factorial of " + num + " is: " + fact);
    }
}

```

OUTPUT:

Factorial of 6 is: 720

3D) Largest number CODE:

```

import java.util.Scanner;

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

      System.out.print("Enter first number: ");
      int num1 = sc.nextInt();

      System.out.print("Enter second number: ");
      int num2 = sc.nextInt();

      System.out.print("Enter third number: ");
      int num3 = sc.nextInt();

      int largest;

      if (num1 >= num2 && num1 >= num3)
          { largest = num1;
        } else if (num2 >= num1 && num2 >= num3)
          { largest = num2;
        } else {
          largest = num3;
        }

      System.out.println("The largest number is: " + largest);
    }
}

```

OUTPIUT:

```

Factorial of 6 is: 720
PS C:\Users\Purnishamahi\OneDrive\Documents\Basic Java Programmes> javac LargestNumber.java
PS C:\Users\Purnishamahi\OneDrive\Documents\Basic Java Programmes> java LargestNumber.java
Enter first number: 5
Enter second number: 567
Enter third number: 123444
The largest number is: 123444

```


3e) LCM

CODE:

```
import java.util.Scanner;

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

        System.out.print("Enter first number: ");
        int num1 = sc.nextInt();

        System.out.print("Enter second number:
        "); int num2 = sc.nextInt();

        int max = (num1 > num2) ? num1 : num2;
        while (true) {
            if (max % num1 == 0 && max % num2 == 0) {
                System.out.println("LCM of " + num1 + " and " + num2 + "
                is:
                " + max);
                break;
            }
            max++;
        }
    }
}
```

Output

```
PS C:\Users\Purnishamahi\OneDrive\Documents\Basic Java Programmes> javac LCM
.java
PS C:\Users\Purnishamahi\OneDrive\Documents\Basic Java Programmes> java LCM.
java
Enter first number: 34
Enter second number: 2
LCM of 34 and 2 is: 34
```

3f) Leap Year Check

Code:

```
import java.util.Scanner;
public class LeapYearCheck {
    public static void main(String[] args)
    {
        Scanner sc = new
        Scanner(System.in);
        System.out.print("Enter a year: ");
        int year = sc.nextInt();

        if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0))
        {
            System.out.println(year + " is a Leap Year.");
        }
        else {
            System.out.println(year + " is NOT a Leap Year.");
        }
    }
}
```

OUTPUT:

```
PS C:\Users\Purnishamahi\OneDrive\Documents\Basic Java Programmes> javac LeapYearCheck.java
PS C:\Users\Purnishamahi\OneDrive\Documents\Basic Java Programmes> java LeapYearCheck.java
Enter a year: 2024
2024 is a Leap Year.
```

3g) Palindrome Check

Code:

```
import java.util.Scanner;
public class PalindromeCheck {
    public static void main(String[] args)
    {
        Scanner sc = new
        Scanner(System.in);

        System.out.print("Enter a number: ");
        int num = sc.nextInt();

        int originalNum = num;
        int reversedNum = 0;

        while (num > 0) {
            int digit = num % 10;
            reversedNum = reversedNum * 10 +
```

```

        digit; num /= 10;
    }

    if (originalNum == reversedNum)
    { System.out.println(originalNum + " is a
      Palindrome.");
    } else {
        System.out.println(originalNum + " is NOT a Palindrome.");
    }
}
}

```

OUTPUT:

```

PS C:\Users\Purnishamahi\OneDrive\Documents\Basic Java Programmes> javac PalindromeCheck.java
PS C:\Users\Purnishamahi\OneDrive\Documents\Basic Java Programmes> java PalindromeCheck.java
Enter a number: 2332
2332 is a Palindrome.

```

3h) Multiplication Table

Code:

```

import java.util.Scanner;

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

        System.out.print("Enter a number: ");
        int num = sc.nextInt();

        System.out.println("Multiplication Table of " + num + ":");
        for (int i = 1; i <= 10; i++) {
            System.out.println(num + " x " + i + " = " + (num * i));
        }
    }
}

```

OUTPIUT:

```
PS C:\Users\Purnishamahi\OneDrive\Documents\Basic Java Programmes> javac MultiplicationTable.java
PS C:\Users\Purnishamahi\OneDrive\Documents\Basic Java Programmes> java MultiplicationTable.java
Enter a number: 4
Multiplication Table of 4:
4 x 1 = 4
4 x 2 = 8
4 x 3 = 12
4 x 4 = 16
4 x 5 = 20
4 x 6 = 24
4 x 7 = 28
4 x 8 = 32
4 x 9 = 36
4 x 10 = 40
```

3i) Power of number

Code:

```
import java.util.Scanner;

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

        System.out.print("Enter the base number: ");
        int base = sc.nextInt();

        System.out.print("Enter the exponent: ");
        int exponent = sc.nextInt();

        int result = 1;

        for (int i = 1; i <= exponent; i++) {
            result *= base;
        }

        System.out.println(base + "^" + exponent + " = " + result);
    }
}
```

OUTPUT:

```
PS C:\Users\Purnishamahi\OneDrive\Documents\Basic Java Programmes> javac PowerOfNumber.java
PS C:\Users\Purnishamahi\OneDrive\Documents\Basic Java Programmes> java PowerOfNumber.java
Enter the base number: 4
Enter the exponent: 3
4^3 = 64
```

3j) Sum of digits**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();
        int sum = 0;

        while (num > 0) {
            sum += num % 10;
            num /= 10;
        }
        System.out.println("Sum of digits: " + sum);
    }
}
```

OUTPUT:

```
PS C:\Users\Purnishamahi\OneDrive\Documents\Basic Java Programmes> javac PowerOfNumber.java
PS C:\Users\Purnishamahi\OneDrive\Documents\Basic Java Programmes> java PowerOfNumber.java
Enter the base number: 4
Enter the exponent: 3
4^3 = 64
```

INHERITANCE

Single inheriatnce:

4A)Teacher Details

Code:

```
class Person {
    String name;
    int age;
    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }

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

class Teacher extends Person {
    String subject;
    public Teacher(String name, int age, String subject) {
        super(name, age); // Calling parent constructor
        this.subject = subject;
    }
    public void displaySubject() {
        System.out.println("Subject: " + subject);
    }
}

public class Main {
```

```
public static void main(String[] args) {  
    Teacher teacher = new Teacher("Mr. Brown", 40, "History");  
    teacher.displayInfo();  
    teacher.displaySubject();  
}  
}
```

OUTPUT:

```
C:\Users\CH.SC.U4CSE24156\Documents\Ex-4>javac Main.java  
  
C:\Users\CH.SC.U4CSE24156\Documents\Ex-4>java Main  
Name: Mr. Brown, Age: 40  
Subject: History
```

4b) Bank account

```
class BankAccount {  
    double balance;  
    public BankAccount(double balance) {  
        this.balance = balance;  
    }  
    public void showBalance() {  
        System.out.println("Balance: $" + balance);  
    }  
}  
  
class SavingsAccount extends BankAccount {  
    public SavingsAccount(double balance) {  
        super(balance);  
    }  
    public void addInterest() {  
        balance += balance * 0.05;  
        System.out.println("Interest added. New Balance: $" + balance);  
    }  
}
```

```
}  
public class Savings {  
    public static void main(String[] args) {  
        SavingsAccount savings = new SavingsAccount(1000);  
        savings.showBalance();  
        savings.addInterest();  
    }  
}
```

OUTPUT:

```
C:\Users\CH.SC.U4CSE24156\Documents\Ex-4>javac Savings.java  
  
C:\Users\CH.SC.U4CSE24156\Documents\Ex-4>java Savings  
Balance: $1000.0  
Interest added. New Balance: $1050.0
```

5.Multi-level Inheritance:

5a)Vehicle

CODE:

```
class Vehicle {  
    public void drive() {  
        System.out.println("Vehicle is driving");  
    }  
}  
  
class Car extends Vehicle {  
    public void openDoors() {  
        System.out.println("Car doors are opening");  
    }  
}  
  
class ElectricCar extends Car {
```



```
public void chargeBattery() {  
    System.out.println("Charging battery");  
}  
}
```

```
public class Main1{  
    public static void main(String[] args) {  
        ElectricCar eCar = new ElectricCar();  
        eCar.drive();  
        eCar.openDoors();  
        eCar.chargeBattery();  
    }  
}
```

OUTPUT:

```
C:\Users\CH.SC.U4CSE24156\Documents\Ex-4>java Main1  
Vehicle is driving  
Car doors are opening  
Charging battery
```

5b) ElectronicDevices

Code:

```
class Device {  
    public void powerOn() {  
        System.out.println("Device is powered on");  
    }  
}  
  
class Computer extends Device {  
    public void runSoftware() {  
        System.out.println("Computer is running software");  
    }  
}
```

```
}

class Laptop extends Computer {
    public void useBattery() {
        System.out.println("Laptop is using battery");
    }
}

public class Main3 {
    public static void main(String[] args) {
        Laptop laptop = new Laptop();
        laptop.powerOn();
        laptop.runSoftware();
        laptop.useBattery();
    }
}
```

OUTPUT:

```
C:\Users\CH.SC.U4CSE24156\Documents\Ex-4>java Main3
Device is powered on
Computer is running software
Laptop is using battery
```

6. Hierarchical Inheritance

6e) Shape

Code:

```
class Shape {
    void draw() {
        System.out.println("Drawing a shape.");
    }
}

class Circle extends Shape {
```

```
void drawCircle() {  
    System.out.println("Drawing a circle.");  
}  
}  
class Rectangle extends Shape {  
    void drawRectangle() {  
        System.out.println("Drawing a rectangle.");  
    }  
}  
public class ShapeHierarchy {  
    public static void main(String[] args) {  
        Circle circle = new Circle();  
        circle.draw();  
        circle.drawCircle();  
  
        Rectangle rectangle = new Rectangle();  
        rectangle.draw();  
        rectangle.drawRectangle();  
    }  
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-4>javac ShapeHierarchy.java  
C:\Users\Purnishamahi\Downloads\Ex-4>java ShapeHierarchy  
Drawing a shape.  
Drawing a circle.  
Drawing a shape.  
Drawing a rectangle.
```

6b) Vehicle Details**CODE:**

```
class Vehicle {  
    void startEngine() {
```

```
        System.out.println("Engine started.");
    }
}
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 HierarchicalInheritance {
    public static void main(String[] args) {
        Car car = new Car();
        car.startEngine();
        car.drive();

        Bike bike = new Bike();
        bike.startEngine();
        bike.ride();
    }
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-4>javac HierarchicalInheritance.java
C:\Users\Purnishamahi\Downloads\Ex-4>java HierarchicalInheritance
Engine started.
Car is driving.
Engine started.
Bike is riding.
```

7.HYBRID HERITANCE

7a) Person Details

Code:

```
class Person {
    void showPersonDetails() {
        System.out.println("This is a person.");
    }
}

class Employee extends Person {
    void showEmployeeDetails() {
        System.out.println("This person is an employee.");
    }
}

class Manager extends Employee {
    void showManagerDetails() {
        System.out.println("This employee is a manager.");
    }
}

class Student extends Person {
    void showStudentDetails() {
        System.out.println("This person is a student.");
    }
}

public class HybridInheritanceExample {
    public static void main(String[] args) {
        Manager manager = new Manager();
        manager.showPersonDetails();
        manager.showEmployeeDetails();
    }
}
```

```
manager.showManagerDetails();  
Student student = new Student();  
student.showPersonDetails();  
student.showStudentDetails();    }  
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-4>javac HybridInheritanceExample.java  
C:\Users\Purnishamahi\Downloads\Ex-4>java HybridInheritanceExample  
This is a person.  
This person is an employee.  
This employee is a manager.  
This is a person.  
This person is a student.
```

7b) Bank Account

Code:

```
class BankAccount {  
    void accountDetails() {  
        System.out.println("This is a bank account.");  
    }  
}  
  
class SavingsAccount extends BankAccount {  
    void interestRate() {  
        System.out.println("Savings Account has 4% interest.");  
    }  
}  
  
class CurrentAccount extends BankAccount {  
    void overdraftLimit() {
```

```
        System.out.println("Current Account has overdraft facility.");
    }
}
class LoanAccount extends SavingsAccount {
    void loanDetails() {
        System.out.println("Loan Account is linked to Savings Account.");
    }
}
public class HybridInheritanceBank {
    public static void main(String[] args) {
        SavingsAccount savings = new SavingsAccount();
        savings.accountDetails();
        savings.interestRate();

        CurrentAccount current = new CurrentAccount();
        current.accountDetails();
        current.overdraftLimit(); // Own method

        LoanAccount loan = new LoanAccount();
        loan.accountDetails();
        loan.interestRate();
        loan.loanDetails();
    }
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-4>java HybridInheritanceBank
This is a bank account.
Savings Account has 4% interest.
This is a bank account.
Current Account has overdraft facility.
This is a bank account.
Savings Account has 4% interest.
Loan Account is linked to Savings Account.
```

POLYMORPHISM

8.CONSTRUCTOR PROGRAMS

8a) Book

CODE:

```
class Book {
    String title;
    String author;
    double price;
    Book(String title, String author, double price) {
        this.title = title;
        this.author = author;
        this.price = price;
    }
    Book(Book b) {
        this.title = b.title;
        this.author = b.author;
        this.price = b.price;
    }
    void display() {
        System.out.println("Title: " + title + ", Author: " + author + ", Price: $"
            + price);
    }
}

public class ConstructorExample2 {
    public static void main(String[] args) {
        Book book1 = new Book("Java Programming", "James Gosling",
            29.99);
        Book book2 = new Book(book1);
    }
}
```



```
        System.out.println("Original Book:");
        book1.display();
        System.out.println("\nCopied Book:");
        book2.display();
    }
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-5>javac  ConstructorExample2.java
C:\Users\Purnishamahi\Downloads\Ex-5>java  ConstructorExample2
Original Book:
Title: Java Programming, Author: James Gosling, Price: $29.99

Copied Book:
Title: Java Programming, Author: James Gosling, Price: $29.99
```

9.ConstructorOverloading

9a)Student Details

Code:

```
class Student {
    String name;
    int age;
    double grade;
    Student(String name, int age) {
        this.name = name;
        this.age = age;
        this.grade = 0.0;
    }
    Student(int age, String name) {
        this.name = name;
        this.age = age;
        this.grade = 0.0;
    }
}
```

```
void display() {  
    System.out.println("Student: " + name + ", Age: " + age + ", Grade: " +  
        grade);  
}  
}
```

```
public class ConstructorOverloadingExample1 {  
    public static void main(String[] args) {  
        Student s1 = new Student("Alice", 20);  
        Student s2 = new Student(22, "Bob");  
        s1.display();  
        s2.display();  
    }  
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-5>javac ConstructorOverloadingExample2.java  
C:\Users\Purnishamahi\Downloads\Ex-5>java ConstructorOverloadingExample1  
Student: Alice, Age: 20, Grade: 0.0  
Student: Bob, Age: 22, Grade: 0.0
```

10. MethodOverloading Programs

10a) Calculator

Code:

```
class Calculator {  
    int add(int a) {  
        return a;  
    }  
    int add(int a, int b) {  
        return a + b;  
    }  
    int add(int a, int b, int c) {  
        return a + b + c;  
    }  
}  
  
public class MethodOverloadingExample1 {  
    public static void main(String[] args) {  
        Calculator calc = new Calculator();  
        System.out.println(calc.add(5));  
        System.out.println(calc.add(5, 10));  
        System.out.println(calc.add(5, 10, 15));    }  
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-5>javac MethodOverloadingExample1.java  
C:\Users\Purnishamahi\Downloads\Ex-5>java MethodOverloadingExample1  
5  
15  
30
```

10b) Employee Details

CODE:

```
class Bank {  
    void deposit(int amount) {  
        System.out.println("Deposited cash: $" + amount);  
    }  
    void deposit(String checkNumber, int amount) {  
        System.out.println("Deposited check #" + checkNumber + " of $" +  
            amount);  
    }  
    void deposit(int amount, String bankName) {  
        System.out.println("Deposited $" + amount + " via online transfer  
            from " + bankName);  
    }  
}  
  
public class MethodOverloadingExample2 {  
    public static void main(String[] args) {  
        Bank myBank = new Bank();  
        myBank.deposit(500);  
        myBank.deposit("CHK12345", 1000);  
        myBank.deposit(2000, "Chase Bank");  
    }  
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-5>javac MethodOverloadingExample2.java  
C:\Users\Purnishamahi\Downloads\Ex-5>java MethodOverloadingExample2  
Deposited cash: $500  
Deposited check #CHK12345 of $1000  
Deposited $2000 via online transfer from Chase Bank
```

11. MethodOverriding Programs

11a) Bank Deposit

Code:

```
class Bank {
    int getInterestRate() {
        return 0;
    }
}

class SBI extends Bank {
    int getInterestRate() {
        return 5;
    }
}

class HDFC extends Bank {
    int getInterestRate() {
        return 7;
    }
}

public class MethodOverridingExample1 {
    public static void main(String[] args) {
        Bank b1 = new SBI();
        Bank b2 = new HDFC();

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

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-5>javac MethodOverridingExample1.java
C:\Users\Purnishamahi\Downloads\Ex-5>java MethodOverridingExample1
SBI Interest Rate: 5%
HDFC Interest Rate: 7%
```

11b) Payment**Code:**

```
class Payment {
    void processPayment() {
        System.out.println("Processing generic payment...");
    }
}

class CreditCardPayment extends Payment {
    void processPayment() {
        System.out.println("Processing credit card payment...");
    }
}

class PayPalPayment extends Payment {
    void processPayment() {
        System.out.println("Processing PayPal payment...");
    }
}

public class MethodOverridingExample2 {
    public static void main(String[] args) {
        Payment payment1 = new CreditCardPayment();
        Payment payment2 = new PayPalPayment();

        payment1.processPayment();
    }
}
```

```
        payment2.processPayment();  
    }  
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-5>javac MethodOverridingExample2.java  
C:\Users\Purnishamahi\Downloads\Ex-5>java MethodOverridingExample2  
Processing credit card payment...  
Processing PayPal payment...
```

Abstarct and Interface

Abstract Programme

12a) Electronic Devices

Code:

```
abstract class Appliance {  
    abstract void turnOn();  
    abstract void turnOff();  
  
    void showUsage() {  
        System.out.println("Use with caution.");  
    }  
}
```

```
class Fan extends Appliance {  
    void turnOn() {  
        System.out.println("Fan is turning on...");  
    }  
  
    void turnOff() {  
        System.out.println("Fan is turning off...");  
    }  
}
```

```
public class Abstarct1 {  
    public static void main(String[] args) {  
        Fan myFan = new Fan();  
        myFan.turnOn();  
        myFan.showUsage();  
        myFan.turnOff();  
    }  
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-6>javac Abstarct1.java  
  
C:\Users\Purnishamahi\Downloads\Ex-6>java Abstarct1  
Fan is turning on...  
Use with caution.  
Fan is turning off...
```

12b) Bank Account

Code:

```
abstract class BankAccount {  
    double balance;
```



```
BankAccount(double balance) {  
    this.balance = balance;  
}  
  
abstract void deposit(double amount);  
abstract void withdraw(double amount);  
  
void showBalance() {  
    System.out.println("Current Balance: $" + balance);  
}  
}  
  
class SavingsAccount extends BankAccount {  
    SavingsAccount(double balance) {  
        super(balance);  
    }  
  
    void deposit(double amount) {  
        balance += amount;  
        System.out.println("Deposited: $" + amount);  
    }  
  
    void withdraw(double amount) {  
        if (balance >= amount) {  
            balance -= amount;  
            System.out.println("Withdrew: $" + amount);  
        } else {  
            System.out.println("Insufficient balance!");  
        }  
    }  
}
```

```
    }  
  }  
}
```

```
public class Abstarct2 {  
    public static void main(String[] args) {  
        SavingsAccount myAccount = new SavingsAccount(500);  
        myAccount.showBalance();  
        myAccount.deposit(200);  
        myAccount.withdraw(100);  
        myAccount.showBalance();  
    }  
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-6>javac Abstarct2.java  
  
C:\Users\Purnishamahi\Downloads\Ex-6>java Abstarct2  
Current Balance: $500.0  
Deposited: $200.0  
Withdrew: $100.0  
Current Balance: $600.0
```

12c)Employee

Code:

```
abstract class Employee {  
    String name;  
    int id;  
  
    Employee(String name, int id) {  
        this.name = name;  
        this.id = id;  
    }  
}
```

```
abstract void work();  
void displayDetails() {  
    System.out.println("Employee ID: " + id + ", Name: " + name);  
}  
}
```

```
class Developer extends Employee {  
    Developer(String name, int id) {  
        super(name, id);  
    }  
  
    void work() {  
        System.out.println(name + " is coding...");  
    }  
}
```

```
public class Abstarct3 {  
    public static void main(String[] args) {  
        Developer dev = new Developer("Alice", 101);  
        dev.displayDetails();  
        dev.work();  
    }  
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-6>javac Abstarct3.java  
C:\Users\Purnishamahi\Downloads\Ex-6>java Abstarct3  
Employee ID: 101, Name: Alice  
Alice is coding...
```

12d) Electronic Device

Code:

```
abstract class ElectronicDevice {  
    String brand;  
    double price;  
  
    ElectronicDevice(String brand, double price) {  
        this.brand = brand;  
        this.price = price;  
    }  
  
    abstract void turnOn();  
    abstract void turnOff();  
  
    void showDetails() {  
        System.out.println("Brand: " + brand + ", Price: $" + price);  
    }  
}  
  
class Laptop extends ElectronicDevice {  
    Laptop(String brand, double price) {  
        super(brand, price);  
    }  
  
    void turnOn() {  
        System.out.println(brand + " Laptop is turning on...");  
    }  
  
    void turnOff() {
```

```
        System.out.println(brand + " Laptop is shutting down...");  
    }  
}
```

```
public class Abstarct4 {  
    public static void main(String[] args) {  
        Laptop myLaptop = new Laptop("Dell", 1200);  
        myLaptop.showDetails();  
        myLaptop.turnOn();  
        myLaptop.turnOff();  
    }  
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-6>javac Abstarct4.java  
  
C:\Users\Purnishamahi\Downloads\Ex-6>java Abstarct4  
Brand: Dell, Price: $1200.0  
Dell Laptop is turning on...  
Dell Laptop is shutting down...
```

Interface programmes

13a) Vehicle

CODE:

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

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

```
public class Interface1 {  
    public static void main(String[] args) {  
        Car myCar = new Car();  
        myCar.start();  
        myCar.stop();  
    }  
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-6>javac Interface1.java
C:\Users\Purnishamahi\Downloads\Ex-6>java Interface1
Car is starting...
Car is stopping...
```

13b) Printer**Code:**

```
interface Printer {
    void print();
}

interface Scanner {
    void scan();
}

class MultiFunctionPrinter implements Printer, Scanner {
    public void print() {
        System.out.println("Printing document...");
    }

    public void scan() {
        System.out.println("Scanning document...");
    }
}

public class Interface2 {
    public static void main(String[] args) {
        MultiFunctionPrinter mfp = new MultiFunctionPrinter();
        mfp.print();
    }
}
```

```
        mfp.scan();  
    }  
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-6>javac Interface2.java  
C:\Users\Purnishamahi\Downloads\Ex-6>java Interface2  
Printing document...  
Scanning document...
```

13c) Display

CODE:

```
interface Display {  
    void showMessage(String message);  
  
    default void showDefault() {  
        System.out.println("This is a default method in the interface.");  
    }  
  
    static void showStatic() {  
        System.out.println("This is a static method in the interface.");  
    }  
}  
  
class Screen implements Display {  
    public void showMessage(String message) {  
        System.out.println("Message: " + message);  
    }  
}  
  
public class Interface3 {  
    public static void main(String[] args) {
```



```
Screen screen = new Screen();
screen.showMessageDialog("Hello, World!");
screen.showDefault();

Display.showStatic();
}
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-6>javac Interface3.java
C:\Users\Purnishamahi\Downloads\Ex-6>java Interface3
Message: Hello, World!
This is a default method in the interface.
This is a static method in the interface.
```

13d) Shape

CODE:

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

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

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

```
}
```

```
public class Interface4{  
    public static void main(String[] args) {  
        Shape myCircle = new Circle();  
        myCircle.draw();  
  
        Shape mySquare = new Square();  
        mySquare.draw();  
    }  
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-6>javac Interface4.java
```

```
C:\Users\Purnishamahi\Downloads\Ex-6>java Interface4
```

```
Drawing a Circle...
```

```
Drawing a Square...
```

ENCAPSULATION

14a) Laptop

CODE:

```
class Laptop {  
    private String brand;  
    private int ramSize;  
  
    public Laptop(String brand, int ramSize) {  
        this.setBrand(brand);  
        this.setRamSize(ramSize);  
    }  
  
    public String getBrand() {  
        return brand;  
    }  
  
    public void setBrand(String brand) {  
        if (brand != null && !brand.isEmpty()) {  
            this.brand = brand;  
        } else {  
            throw new IllegalArgumentException("Brand cannot be null or  
empty");  
        }  
    }  
  
    public int getRamSize() {  
        return ramSize;  
    }  
}
```

```
public void setRamSize(int ramSize) {
    if (ramSize > 0) {
        this.ramSize = ramSize;
    } else {
        throw new IllegalArgumentException("RAM size must be greater
        than 0");
    }
}

@Override
public String toString() {
    return "Laptop{brand='" + brand + "', ramSize=" + ramSize + "GB"}";
}

public static void main(String[] args) {
    Laptop myLaptop = new Laptop("Dell", 16);
    System.out.println("Laptop Details: " + myLaptop);
}
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-7>javac Laptop.java
C:\Users\Purnishamahi\Downloads\Ex-7>java Laptop
Laptop Details: Laptop{brand='Dell', ramSize=16GB}
```

14b) Movie

CODE:

```
class Movie {
```

```
private String title;
private double rating;

public Movie(String title, double rating) {
    this.setTitle(title);
    this.setRating(rating);
}

public String getTitle() {
    return title;
}

public void setTitle(String title) {
    if (title != null && !title.isEmpty()) {
        this.title = title;
    } else {
        throw new IllegalArgumentException("Title cannot be null or
empty");
    }
}

public double getRating() {
    return rating;
}

public void setRating(double rating) {
    if (rating >= 0 && rating <= 10) {
        this.rating = rating;
    } else {
```

```
        throw new IllegalArgumentException("Rating must be between 0
and 10");
    }
}

@Override
public String toString() {
    return "Movie{title='" + title + "', rating=" + rating + "}";
}

public static void main(String[] args) {
    Movie myMovie = new Movie("Inception", 8.8);
    System.out.println("Movie Details: " + myMovie);
}
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-7>javac Movie.java
C:\Users\Purnishamahi\Downloads\Ex-7>java Movie
Movie Details: Movie{title='Inception', rating=8.8}
```

14c) Flight**Code:**

```
class Flight {
    private String destination;
    private int seatCount;
```

```
public Flight(String destination, int seatCount) {
    this.setDestination(destination);
    this.setSeatCount(seatCount);
}

public String getDestination() {
    return destination;
}

public void setDestination(String destination) {
    if (destination != null && !destination.isEmpty()) {
        this.destination = destination;
    } else {
        throw new IllegalArgumentException("Destination cannot be null
or empty");
    }
}

public int getSeatCount() {
    return seatCount;
}

public void setSeatCount(int seatCount) {
    if (seatCount > 0) {
        this.seatCount = seatCount;
    } else {
        throw new IllegalArgumentException("Seat count must be greater
than 0");
    }
}
```

```
}
```

```
@Override
```

```
public String toString() {
```

```
    return "Flight{destination=" + destination + ", seatCount=" +  
        seatCount + "}";
```

```
}
```

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

```
    Flight myFlight = new Flight("New York", 150);
```

```
    System.out.println("Flight Details: " + myFlight);
```

```
}
```

```
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-7>javac Flight.java
```

```
C:\Users\Purnishamahi\Downloads\Ex-7>java Flight  
Flight Details: Flight{destination='New York', seatCount=150}
```

14d) Temperature

CODE:

```
class Temperature {
```

```
    private double celsius;
```

```
    public Temperature(double celsius) {
```

```
        this.setCelsius(celsius);
```

```
}
```

```
    public void setCelsius(double celsius) {
```



```
        this.celsius = celsius;
    }

    public double getCelsius() {
        return celsius;
    }

    public double toFahrenheit() {
        return (celsius * 9 / 5) + 32;
    }

    @Override
    public String toString() {
        return "Temperature{celsius=" + celsius + ", fahrenheit=" +
            toFahrenheit() + "}";
    }

    public static void main(String[] args) {
        Temperature temp = new Temperature(25);
        System.out.println("Temperature Details: " + temp);
    }
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-7>javac Temperature.java
C:\Users\Purnishamahi\Downloads\Ex-7>java Temperature
Temperature Details: Temperature{celsius=25.0, fahrenheit=77.0}
```

15.Packages

15a) Built packages

```
import java.util.regex.*;
import java.lang.String;
import java.time.*;

public class StringRegexDateExample {
    public static void main(String[] args) {
        String text = "Java is fun!";
        String upperCaseText = text.toUpperCase();
        System.out.println("Uppercase Text: " + upperCaseText);
        Pattern pattern = Pattern.compile("\\bJ\\w*");
        Matcher matcher = pattern.matcher("Java is fun! JavaScript is also fun.");
        System.out.println("Words starting with 'J':");
        while (matcher.find()) {
            System.out.println(matcher.group());
        }
        LocalDate currentDate = LocalDate.now();
        LocalTime currentTime = LocalTime.now();
        LocalDateTime currentDateTime = LocalDateTime.now();

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

OUTPUT:

```
C:\Users\CH.SC.U4CSE24156\Documents>javac MultiPackageExample.java

C:\Users\CH.SC.U4CSE24156\Documents>java MultiPackageExample
Numbers List: [10, 20, 30, 40]
Square root of 16: 4.0
2 raised to the power of 3: 8.0
Task 2 is running, calculating sum of 10 and 20: 30
Task 1 is running, calculating square of 5: 25.0
```

15b) Built in Package**Code:**

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

public class MultiPackageExample {
    public static void main(String[] args) throws InterruptedException {
        List<Integer> numbers = new ArrayList<>();
        numbers.add(10);
        numbers.add(20);
        numbers.add(30);
        numbers.add(40);
        System.out.println("Numbers List: " + numbers);
        double sqrtResult = Math.sqrt(16);
        double powResult = Math.pow(2, 3);
        System.out.println("Square root of 16: " + sqrtResult);
        System.out.println("2 raised to the power of 3: " + powResult);
        ExecutorService executor = Executors.newFixedThreadPool(2);
        Runnable task1 = () -> {
            System.out.println("Task 1 is running, calculating square of 5: " +
```

```
Math.pow(5, 2));  
};  
Runnable task2 = () -> {  
    System.out.println("Task 2 is running, calculating sum of 10 and  
20: " + (10 + 20));  
};  
  
executor.submit(task1);  
executor.submit(task2);  
  
executor.shutdown();  
executor.awaitTermination(1, TimeUnit.SECONDS); }  
}
```

OUTPUT:

```
C:\Users\CH.SC.U4CSE24156\Documents>javac MultiPackageExample.java  
  
C:\Users\CH.SC.U4CSE24156\Documents>java MultiPackageExample  
Numbers List: [10, 20, 30, 40]  
Square root of 16: 4.0  
2 raised to the power of 3: 8.0  
Task 2 is running, calculating sum of 10 and 20: 30  
Task 1 is running, calculating square of 5: 25.0
```

15c) Bank account

CODE:

```
package mypackage;  
  
public class BankAccount {  
    private String accountHolder;  
    private double balance;  
    public BankAccount(String accountHolder, double balance) {  
        this.accountHolder = accountHolder;  
        this.balance = balance;  
    }  
}
```

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

public void withdraw(double amount) {
    if (amount > 0 && amount <= balance) {
        balance -= amount;
        System.out.println("Withdrew: $" + amount);
    } else {
        System.out.println("Invalid withdrawal amount or insufficient
        funds.");
    }
}

public double getBalance() {
    return balance;
}

public String getAccountHolder() {
    return accountHolder;
}

public static void main(String[] args) {
    BankAccount account = new BankAccount("John Doe", 1000);
    System.out.println("Account Holder: " +
        account.getAccountHolder());
    System.out.println("Initial Balance: $" + account.getBalance());
    account.deposit(200);
}
```

```
        account.withdraw(150);
        account.withdraw(1200);
        account.deposit(-50);
        System.out.println("Final Balance: $" + account.getBalance());
    }
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-8>javac -d . mypackage/BankAccount.java
C:\Users\Purnishamahi\Downloads\Ex-8>java mypackage.BankAccount
Account Holder: John Doe
Initial Balance: $1000.0
Deposited: $200.0
Withdrew: $150.0
Invalid withdrawal amount or insufficient funds.
Deposit amount must be positive.
Final Balance: $1050.0
```

15d) Calculator**CODE:**

```
package mypackage;

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

    public int subtract(int a, int b) {
        return a - b;
    }

    public int multiply(int a, int b) {
        return a * b;
    }

    public double divide(int a, int b) {
        if (b != 0) {
            return (double) a / b;
        }
    }
}
```

```
    } else {  
        System.out.println("Cannot divide by zero.");  
        return 0;  
    }  
}  
  
public static void main(String[] args) {  
    Calculator calc = new Calculator();  
    int sum = calc.add(10, 5);  
    int difference = calc.subtract(10, 5);  
    int product = calc.multiply(10, 5);  
    double quotient = calc.divide(10, 5);  
    double invalidQuotient = calc.divide(10, 0);  
    System.out.println("Addition: 10 + 5 = " + sum);  
    System.out.println("Subtraction: 10 - 5 = " + difference);  
    System.out.println("Multiplication: 10 * 5 = " + product);  
    System.out.println("Division: 10 / 5 = " + quotient);  
    System.out.println("Division by zero: 10 / 0 = " + invalidQuotient);  
}  
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-8>javac -d . mypackage/Calculator.java  
C:\Users\Purnishamahi\Downloads\Ex-8>java mypackage.Calculator  
Cannot divide by zero.  
Addition: 10 + 5 = 15  
Subtraction: 10 - 5 = 5  
Multiplication: 10 * 5 = 50  
Division: 10 / 5 = 2.0  
Division by zero: 10 / 0 = 0.0
```

16. EXCEPTION HANDLIN PROGRAMS

16a) Number Format

CODE:

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

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-9>javac NumberFormat.java  
C:\Users\Purnishamahi\Downloads\Ex-9>java NumberFormat  
Error: Invalid number format.
```

16b) custom Exception

Code:

```
class AgeException extends Exception {  
    public AgeException(String message) {
```



```
        super(message);
    }
}

public class CustomException {
    public static void checkAge(int age) throws AgeException {
        if (age < 18) {
            throw new AgeException("Age must be 18 or above.");
        }
    }

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

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-9>javac CustomException.java
C:\Users\Purnishamahi\Downloads\Ex-9>java CustomException
Custom Exception: Age must be 18 or above.
```

16c) Input Mismatch

Code:

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

public class InputMismatch {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        try {
            System.out.print("Enter a number: ");
            int number = scanner.nextInt();
            System.out.println("You entered: " + number);
        } catch (InputMismatchException e) {
            System.out.println("Error: Please enter a valid integer.");
        } finally {
            scanner.close();
        }
    }
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-9>javac InputMismatch.java
C:\Users\Purnishamahi\Downloads\Ex-9>java InputMismatch
Enter a number: hghggh
Error: Please enter a valid integer.
```

16d) Using Finally Word

Code:

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

public class Finally {
    public static void main(String[] args) {
        FileWriter writer = null;
        try {
            writer = new FileWriter("output.txt");
            writer.write("Hello, World!");
        } catch (IOException e) {
            System.out.println("Error: Unable to write to file.");
        } finally {
            try {
                if (writer != null) {
                    writer.close();
                    System.out.println("File closed successfully.");
                }
            } catch (IOException e) {
                System.out.println("Error closing the file.");
            }
        }
    }
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-9>javac Finally.java  
C:\Users\Purnishamahi\Downloads\Ex-9>java Finally  
File closed successfully.
```

17.FILE HANDLING PROGRAMS

17a) Create a file

CODE:

```
import java.io.File;  
import java.io.IOException;  
  
public class CreateFile {  
    public static void main(String[] args) {  
        try {  
            File myFile = new File("purnisha.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\Purnishamahi\Downloads\Ex-9>javac CreateFile.java  
C:\Users\Purnishamahi\Downloads\Ex-9>java CreateFile  
File created: purnisha.txt
```

17b) Write matter to that files

CODE:

```
import java.io.FileWriter;  
import java.io.IOException;  
  
public class WriteToFile {  
    public static void main(String[] args) {  
        try {  
            FileWriter writer = new FileWriter("purnisha.txt");  
            writer.write("Hello, this is a test file.\nWelcome to Java File  
Handling!");  
            writer.close();  
            System.out.println("Successfully wrote to the file.");  
        } catch (IOException e) {  
            System.out.println("An error occurred.");  
            e.printStackTrace();  
        }  
    }  
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-9>javac WriteToFile.java
C:\Users\Purnishamahi\Downloads\Ex-9>java WriteToFile
Successfully wrote to the file.
```

```
Hello, this is a test file.
Welcome to Java File Handling!
```

17c) 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("purnisha.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\Purnishamahi\Downloads\Ex-9>javac ReadFile.java  
C:\Users\Purnishamahi\Downloads\Ex-9>java ReadFile  
Hello, this is a test file.  
Welcome to Java File Handling!
```

17d) Append To file

Code:

```
import java.io.FileWriter;  
import java.io.IOException;  
public class AppendToFile {  
    public static void main(String[] args) {  
        try {  
            FileWriter writer = new FileWriter("purnisha.txt", true);  
            writer.write("\nThis is an appended text.");  
            writer.close();  
            System.out.println("Successfully appended to the file.");  
        } catch (IOException e) {  
            System.out.println("An error occurred.");  
            e.printStackTrace();  
        }  
    }  
}
```

OUTPUT:

```
C:\Users\Purnishamahi\Downloads\Ex-9>javac AppendToFile.java  
C:\Users\Purnishamahi\Downloads\Ex-9>java AppendToFile  
Successfully appended to the file.
```

```
Hello, this is a test file.  
Welcome to Java File Handling!  
This is an appended text.
```