

# VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum -590014, Karnataka.



## LAB REPORT

on

## OBJECT ORIENTED JAVA PROGRAMMING

*Submitted by*

**Sharada Koundinya (1BM23CS310)**

*in partial fulfillment for the award of the degree of*

**BACHELOR OF ENGINEERING**

*in*

**COMPUTER SCIENCE AND ENGINEERING**



**B.M.S. COLLEGE OF ENGINEERING**

(Autonomous Institution under VTU)

**BENGALURU-560019**

**Sep 2024-Jan 2025**

**B. M. S. College of Engineering,**  
**Bull Temple Road, Bangalore 560019**  
(Affiliated To Visvesvaraya Technological University, Belgaum)  
**Department of Computer Science and Engineering**



**CERTIFICATE**

This is to certify that the Lab work entitled “**OBJECT ORIENTED JAVA PROGRAMMING**” carried out by **Sharada Koundinya (1BM23CS310)**, who is a bona fide student of **B. M. S. College of Engineering**. It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2024-25. The Lab report has been approved as it satisfies the academic requirements in respect of **Object-Oriented Java Programming Lab - (23CS3PCOOJ)** work prescribed for the said degree.

**Dr. Nandhini Vineeth**

Associate Professor,  
Department of CSE,  
BMSCE, Bengaluru

**Dr. Kavitha Sooda**

Professor and Head,  
Department of CSE  
BMSCE, Bengaluru

## **INDEX**

<b>Sl. No.</b>	<b>Date</b>	<b>Experiment Title</b>	<b>Page No.</b>
1	26-09-24	Quadratic equations	4-5
2	03-10-24	SGPA calculation	6-9
3	19-10-24	Book details	10-12
4	24-10-24	Shapes	13-15
5	07-11-24	Bank accounts	16-21
6	14-11-24	Packages	22-25
7	21-11-24	Exceptions	26-28
8	28-11-24	Multithreads	28-29
9	19-12-24	opened exp1	30-34
10	19-12-24	opened exp2	34-40

## Program 1

Develop a Java program that prints all real solutions to the quadratic equation  $ax^2+bx+c=0$ . Read in a, b, c and use the quadratic formula. If the discriminate  $b^2-4ac$  is negative, display a message stating that there are no real solutions.

Program 1

```
import java.util.Scanner;
class QuadEq {
    public static void main (String[] Args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter coefficient of a:");
        double a = sc.nextDouble();
        System.out.println("Enter coefficient of b:");
        double b = sc.nextDouble();
        System.out.println("Enter coefficient of c:");
        double c = sc.nextDouble();
        double disc = b*b - 4*a*c;
        if (disc > 0) {
            double root1 = (-b + Math.sqrt(disc)) / (2*a);
            double root2 = (-b - Math.sqrt(disc)) / (2*a);
            System.out.println("Equation has two real roots: " + root1 + " " + root2);
        }
        else if (disc == 0) {
            double root = -b / (2*a);
            System.out.println("Equation has one real solution: " + root);
        }
    }
}
```

else {
 System.out.println("Equation has no real roots");
}

output I

Enter coefficient of a: 2  
Enter coefficient of b: 5  
Enter coefficient of c: 3  
Equation has two real roots: -1.0 -1.5

output II

Enter coefficient of a: 1  
Enter coefficient of b: 4  
Enter coefficient of c: 4  
Equation has one real solution: -2.0

output III

Enter coefficient of a: 5  
Enter coefficient of b: 2  
Enter coefficient of c: 4  
Equation has no real roots

## CODE:

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

```
class Quad_Eq_cal {
    public static void main(String [] args){
        int y=0;
        Scanner sc=new Scanner(System.in);
        System.out.println("General form of a quadratic equation is ax^2+bx+c=0");
        do{
            System.out.print("\nEnter value of a=");
            int a=sc.nextInt();
            System.out.print("Enter value of b=");
            int b=sc.nextInt();
            System.out.print("Enter value of c=");
            int c=sc.nextInt();
            float d=(float)(Math.pow(b,2)-4*a*c);
```

```

    if(d<0){
        System.out.println("There are no real solutions");
    }
    else if(d==0){
        System.out.println("It has one repeated root(2 equal roots):");
        float r=-b/(2.0f*a);
        System.out.println("x="+r);
    }
    else{
        System.out.println("It has two distinct roots:");
        double r1=(-b+Math.sqrt(d))/(2*a);
        System.out.println("x1="+r1);
        double r2=(-b-Math.sqrt(d))/(2*a);
        System.out.println("x2="+r2);
    }
    System.out.println("\nDo you want to calculate again?(yes=0 and no=1): ");
    y=sc.nextInt();
}while(y==0);
}
}

```

## OUTPUT:

```

C:\295>java QuadraticEquation
Enter the coefficients of a, b, c:
Enter coefficient a: 8 50 20
Enter coefficient b: Enter coefficient c: Roots are real and unique.
Root 1: -0.42951766839402206
Root 2: -5.820482331605978

C:\295>javac QuadraticEquation.java

C:\295>java QuadraticEquation
Enter the coefficients of a, b, c:
Enter coefficient a: 3 4 5
Enter coefficient b: Enter coefficient c: Roots are imaginary.
Root 1: -0.6666666666666666 + 1.1055415967851332i
Root 2: -0.6666666666666666 - 1.1055415967851332i

C:\295>javac QuadraticEquation.java

C:\295>java QuadraticEquation
Enter the coefficients of a, b, c:
Enter coefficient a: 1 2 5
Enter coefficient b: Enter coefficient c: Roots are imaginary.
Root 1: -1.0 + 2.0i
Root 2: -1.0 - 2.0i

```

## Program 2

Develop a Java program to create a class Student with members usn, name, an array credits and an array marks. Include methods to accept and display details and a method to calculate SGPA of a student.

```

Program II
Q. Develop java program to create a class student w/ members usn, name, array credits and array marks. Include methods to accept and display details and a method to calculate SGPA of a student.

import java.util.Scanner;

class student
{
    String name;
    String usn;
    int numsub;
    int[] credits;
    int[] marks;

    student(int num)
    {
        numsub = num;
        credits = new int[numsub];
        marks = new int[numsub];
    }

    void Details()
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter USN:");
        usn = sc.nextLine();
        System.out.println("Enter Name:");
        name = sc.nextLine();
        System.out.println("Enter credits for each subject:");
        for(int i=0; i<numsub; i++)
    }

```

```

        System.out.println("Credits for subject " + (i+1) + ":");
        credits[i] = sc.nextInt();
    }
    System.out.println("Enter marks for each subject:");
    for(int i=0; i<numsub; i++)
    {
        System.out.println("Marks for subject " + (i+1) + ":");
        marks[i] = sc.nextInt();
    }

    void display()
    {
        System.out.println("In Student details:");
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        System.out.println("In Subject wise credits and marks");
        for(int i=0; i<numsub; i++)
        {
            System.out.println("Subject " + (i+1) + " - Credits: " + credits[i] + " Marks: " + marks[i]);
        }
        System.out.println("SGPA: " + calculateSGPA());
    }

    double calculateSGPA()
    {
        int totalCredits = 0;
        int totalGradePoints = 0;
        for(int i=0; i<numsub; i++)
        {
            totalCredits += credits[i];
            totalGradePoints += calculateGradePoint(marks[i]);
        }
        return (double) totalGradePoints / totalCredits;
    }

    int calculateGradePoint(int marks)
    {
        if(marks >= 90)
            return 10;
        else if(marks >= 80)
            return 9;
        else if(marks >= 70)
            return 8;
        else if(marks >= 60)
            return 7;
        else if(marks >= 50)
            return 6;
        else if(marks >= 40)
            return 5;
        else
            return 0;
    }

    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter number of subjects:");
        int numsub = sc.nextInt();
        student s = new student(numsub);
        s.Details();
        s.display();
    }
}

```

```

Output
Enter number of subjects: 5
Enter USN: 18N23CS310
Enter name: Shashik Kaurdingya
Enter credits for each subject:
Credits for subject 1: 4
subject 2: 4
subject 3: 3
subject 4: 3
subject 5: 2
Enter marks for each subject:
Marks for subject 1: 98
subject 2: 95
subject 3: 91
subject 4: 92
subject 5: 96

Student Details
USN: 18N23CS310
Name: Shashik Kaurdingya

Subjectwise Credit & Marks
Subject 1: Credits = 4, Marks = 98
Subject 2: Credits = 4, Marks = 95
Subject 3: Credits = 3, Marks = 91
Subject 4: Credits = 3, Marks = 92
Subject 5: Credits = 2, Marks = 96

SGPA = 10.0

```

## CODE:

```
import java.util.Scanner;
class Subject {
    int subM;
    int cred;
    int grade;

    void setSubDet(int marks, int cred) {
        this.subM = marks;
        this.cred = cred;
        if (subM >= 90) {
            grade = 10;
        } else if (subM >= 80) {
            grade = 9;
        } else if (subM >= 70) {
            grade = 8;
        } else if (subM >= 60) {
            grade = 7;
        } else if (subM >= 50) {
            grade = 6;
        } else if (subM >= 40) {
            grade = 5;
        } else {
            grade = 0;
        }
    }
}

class Student {
    Scanner s = new Scanner(System.in);
    Subject[] subjects = new Subject[8];
    Student() {
        for (int i = 0; i < subjects.length; i++) {
            subjects[i] = new Subject();
        }
    }

    void getMarks() {
        for (int i = 0; i < subjects.length; i++) {
            System.out.print("Enter marks for subject " + (i + 1) + ": ");
            int marks = s.nextInt();
            System.out.print("Enter credit for subject " + (i + 1) + ": ");
            int cred = s.nextInt();
            subjects[i].setSubDet(marks, cred);
        }
    }
}
```



```

    }
}

double calSGPA() {
    double Score = 0;
    int totalCred = 0;
    double SGPA = 0.0;

    for (Subject subject : subjects) {
        Score += (subject.grade * subject.cred);
        totalCred += subject.cred;
    }

    if (totalCred > 0) {
        SGPA = Score / totalCred;
    } else {
        SGPA = 0;
    }
    return SGPA;
}
}

public class StudentDetails {
    public static void main(String[] arg) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of semesters: ");
        int numSems = sc.nextInt();
        Student[] students = new Student[numSems];
        double cumulativeSGPA = 0.0;
        System.out.print("Enter USN: ");
        String usn = sc.next();
        System.out.print("Enter Name: ");
        String name = sc.next();

        for (int i = 0; i < numSems; i++) {
            System.out.println("Enter details for semester " + (i + 1));
            students[i] = new Student();
            students[i].getMarks();
            double semSGPA = students[i].calSGPA();
            cumulativeSGPA += semSGPA;
        }

        for (int i = 0; i < numSems; i++) {
            System.out.println("USN: " + usn);
            System.out.println("Name: " + name);
            System.out.println("SGPA for sem " + (i + 1) + ": " + students[i].calSGPA());
        }
    }
}

```



```
        double CGPA = cumulativeSGPA / numSems;  
        System.out.println("CGPA: " + CGPA);  
    }  
}
```

## OUTPUT:

```
C:\Users\Admin\Documents\23cs310>java StudentDetails  
Enter number of semesters: 1  
Enter USN: 1bm23cs310  
Enter Name: sharada koundinya  
Enter details for semester 1  
Enter marks for subject 1: 98  
Enter credit for subject 1: 4  
Enter marks for subject 2: 95  
Enter credit for subject 2: 4  
Enter marks for subject 3: 91  
Enter credit for subject 3: 3  
Enter marks for subject 4: 92  
Enter credit for subject 4: 3  
Enter marks for subject 5: 96  
Enter credit for subject 5: 2  
Enter marks for subject 6: 88  
Enter credit for subject 6: 2  
Enter marks for subject 7: 76  
Enter credit for subject 7: 3  
Enter marks for subject 8: 61  
Enter credit for subject 8: 3  
USN: 1bm23cs310  
Name: sharada  
SGPA for sem 1: 9.291666666666666  
CGPA: 9.291666666666666
```

### Program 3

Create a class Book which contains four members: name, author, price, num\_pages. Include a constructor to set the values for the members. Include methods to set and get the details of the objects. Include a toString() method that could display the complete details of the book. Develop a Java program to create n book objects.

Program III

Q. WAP to create a class Book which contains 4 members: name, author, price, num-pages. Include a constructor to set values for members. Include a toString() method that could display complete details of the book. Create n Book objects.

```
import java.util.Scanner;

class Book {
    String name, author;
    double price;
    int nPage;
    Book() {}
    Book(String name, String author, double price, int nPage) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.nPage = nPage;
    }

    void setDetails() {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter name of book:");
        name = sc.nextLine();
        System.out.println("Enter name of author:");
        author = sc.nextLine();
        System.out.println("Enter price of book:");
        price = sc.nextDouble();
        System.out.println("Enter no. of Pages:");
        nPage = sc.nextInt();
    }
}
```

```
void getDetails() {
    S.o.p("Name of book " + name);
    S.o.p("Author " + author);
    S.o.p("Price " + price);
    S.o.p("Number of pages " + nPage);
}

public String toString() {
    return "Book name " + name + "\n" +
        "Author " + author + "\n" + "Price " + price +
        "\n" + "Number of pages " + nPage + "\n";
}

class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        S.o.p("Enter no. of books:");
        int n = sc.nextInt();
        sc.nextLine();
        Book[] books = new Book[n];
        for (int i = 0; i < n; i++) {
            books[i] = new Book();
            S.o.p("Enter details for book " + (i+1));
            books[i].setDetails();
            books[i].getDetails();
        }
        S.o.p("All book details:");
        for (Book book : books) {
            S.o.p(book);
        }
    }
}
```

output

Enter no. of books: 2

Enter details for book 1

Enter name of book: Harry Potter

Enter author name: JK Rowling

Enter price: 500.75

Enter no. pages: 350

Enter details for book 2

Enter name of book: Magic of Lost Lamp

Enter author name: Sudha Murthy

Enter price: 650

Enter no. pages: 400

Book name - Harry Potter

Author - JK Rowling

Price - 500.75

Pages - 350

Book name - Magic of Lost Lamp

Author - Sudha Murthy

Price - 650

Pages - 400

## CODE:

```
import java.util.Scanner;
class Book {
    String name, author;
    double price;
    int noPage;
    Book() {}
    Book(String name, String author, double price, int noPage) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.noPage = noPage;
    }
    void setDetails() {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter name of book: ");
        name = sc.nextLine();
        System.out.println("Enter author name: ");
        author = sc.nextLine();
        System.out.println("Enter price of book: ");
        price = sc.nextDouble();
        System.out.println("Enter number of pages: ");
        noPage = sc.nextInt();
    }
    void getDetails() {
        System.out.println("Name of book: " + name);
        System.out.println("Author: " + author);
        System.out.println("Price: " + price);
        System.out.println("Number of pages: " + noPage);
    }
    public String toString() {
        return "Book name: " + name + "\n" + "Author: " + author + "\n" + "Price: " + price +
            "\n" + "Number of pages: " + noPage + "\n";
    }
}
class MyBook {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter number of books: ");
        int n = sc.nextInt();
        sc.nextLine();
        Book[] books = new Book[n];
        for (int i = 0; i < n; i++) {
            books[i] = new Book();
            System.out.println("Enter details for book " + (i + 1));
            books[i].setDetails();
        }
    }
}
```

```

        books[i].getDetails();
    }
    System.out.println("All book details: ");
    for (Book book : books) {
        System.out.println(book);
    }
}
}

```

## OUTPUT:

```

C:\Users\Admin\Documents\23cs310>java MyBook
Enter number of books:
2
Enter details for book 1
Enter name of book:
HarryPotter
Enter author name:
JK Rowling
Enter price of book:
500.75
Enter number of pages:
350
Name of book: HarryPotter
Author: JK Rowling
Price: 500.75
Number of pages: 350
Enter details for book 2
Enter name of book:
Magic of Lost Lamp
Enter author name:
Sudha Murthy
Enter price of book:
650
Enter number of pages:
400
Name of book: Magic of Lost Lamp
Author: Sudha Murthy
Price: 650.0
Number of pages: 400
All book details:
Book name: HarryPotter
Author: JK Rowling
Price: 500.75
Number of pages: 350

Book name: Magic of Lost Lamp
Author: Sudha Murthy
Price: 650.0
Number of pages: 400

```

## Program 4

Develop a Java program to create an abstract class named Shape that contains two integers and an empty method named printArea(). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contain only the method printArea() that prints the area of the given shape.

Program IV

```

1. Develop a Java program to create an abstract class named Shape that contains two integers and an empty method named printArea(). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contain only the method printArea() that prints the area of the given shape.

import java.util.Scanner;

abstract class Shape {
    int dimension1;
    int dimension2;

    abstract void printArea();
}

class Rectangle extends Shape {
    public Rectangle(int length, int width) {
        this.dimension1 = length;
        this.dimension2 = width;
    }

    void printArea() {
        int area = dimension1 * dimension2;
        S.O.p("Rectangle Area " + area);
    }
}

class Triangle extends Shape {
    public Triangle(int base, int height) {
        this.dimension1 = base;
        this.dimension2 = height;
    }

    void printArea() {
        double area = 0.5 * dimension1 * dimension2;
        S.O.p("Triangle Area " + area);
    }
}

class Circle extends Shape {
    private final double pi = 3.14159;

    public Circle(int radius) {
        this.dimension1 = radius;
        this.dimension2 = 0;
    }

    void printArea() {
        double area = pi * dimension1 * dimension1;
        S.O.p("Circle Area " + area);
    }
}

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

        S.O.p("Enter length of rectangle");
        int length = scanner.nextInt();
        S.O.p("Enter width of rectangle");
        int width = scanner.nextInt();
        Rectangle rectangle = new Rectangle(length, width);
        rectangle.printArea();

        S.O.p("Enter base of triangle");
        int base = scanner.nextInt();
        S.O.p("Enter height of triangle");
        int height = scanner.nextInt();
        Triangle triangle = new Triangle(base, height);
        triangle.printArea();

        S.O.p("Enter radius of circle");
        int radius = scanner.nextInt();
        Circle circle = new Circle(radius);
        circle.printArea();

        scanner.close();
    }
}

```

```

S.O.p("Enter base of triangle");
int base = scanner.nextInt();
S.O.p("Enter height of triangle");
int height = scanner.nextInt();
Triangle triangle = new Triangle(base, height);
triangle.printArea();

S.O.p("Enter radius of circle");
int radius = scanner.nextInt();
Circle circle = new Circle(radius);
circle.printArea();

scanner.close();
}

output
Enter length of rectangle 5
Enter width of rectangle 3
Rectangle Area 15

Enter base of triangle 4
Enter height of triangle 6
Triangle Area 12

Enter radius of circle 7
Circle Area 153.93804

```

## CODE:

```
import java.util.Scanner;
abstract class Shape {
    int dimension1;
    int dimension2;

    abstract void printArea();
}
class Rectangle extends Shape {
    public Rectangle(int length, int width) {
        this.dimension1 = length;
        this.dimension2 = width;
    }
    void printArea() {
        int area = dimension1 * dimension2;
        System.out.println("Rectangle Area: " + area);
    }
}

class Triangle extends Shape {
    public Triangle(int base, int height) {
        this.dimension1 = base;
        this.dimension2 = height;
    }
    void printArea() {
        double area = 0.5 * dimension1 * dimension2;
        System.out.println("Triangle Area: " + area);
    }
}

class Circle extends Shape {
    private final double pi = 3.14159;
    public Circle(int radius) {
        this.dimension1 = radius;
        this.dimension2 = 0;
    }
    void printArea() {
        double area = pi * dimension1 * dimension1;
        System.out.println("Circle Area: " + area);
    }
}

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

        System.out.print("Enter length of rectangle: ");
```

```

int length = scanner.nextInt();
System.out.print("Enter width of rectangle: ");
int width = scanner.nextInt();
Rectangle rectangle = new Rectangle(length, width);
rectangle.printArea();

System.out.print("Enter base of triangle: ");
int base = scanner.nextInt();
System.out.print("Enter height of triangle: ");
int height = scanner.nextInt();
Triangle triangle = new Triangle(base, height);
triangle.printArea();

System.out.print("Enter radius of circle: ");
int radius = scanner.nextInt();
Circle circle = new Circle(radius);
circle.printArea();
scanner.close();
}
}

```

## OUTPUT:

```

Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin\Documents\23cs310>javac Main.java

C:\Users\Admin\Documents\23cs310>java Main
Enter length of rectangle: 5
Enter width of rectangle: 3
Rectangle Area: 15
Enter base of triangle: 4
Enter height of triangle: 6
Triangle Area: 12.0
Enter radius of circle: 7
Circle Area: 153.93791

```



## Program 5

Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called savings account and the other current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed. Create a class Account that stores customer name, account number and type of account. From this derive the classes Cur-acct and Sav-acct to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks:

- Accept deposit from customer and update the balance.
- Display the balance.
- Compute and deposit interest
- Permit withdrawal and update the balance

Check for the minimum balance, impose penalty if necessary and update the balance.

Program V

Q. Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called savings account and the other current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facilities but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level a service charge is imposed. Create class Account that stores customer name, account number and type of account. From this derive the class Cur-acct and Sav-acct to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks:

- Accept deposit from customer and update the balance
- Display the balance
- Compute and deposit interest
- Permit withdrawal and update the balance

Check for the minimum balance, impose penalty if necessary and update the balance

```
import java.util.Scanner;

class Account {
    private String customerName;
    private int acc.no;
    protected double balance;
```

```
public Account(String customerName, int acc.no, double balance) {
    this.customerName = customerName;
    this.acc.no = acc.no;
    this.balance = balance;
}

public double getBalance() {
    return 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 <= getBalance()) {
        balance -= amount;
        System.out.println("Withdraw " + amount);
        System.out.println("balance is " + balance);
    }
    else {
        System.out.println("Insufficient Balance");
    }
}

public void displayBalance() {
    System.out.println("Current Balance " + balance);
}

class SavingsAccount extends Account {
    private double interestRate;

    public SavingsAccount(String customerName, int accountName, double initialBalance, double interestRate) {
        super(customerName, accountName, initialBalance);
        this.interestRate = interestRate;
    }

    public void computeAndDepositInterest() {
        double interest = getBalance() * interestRate / 12;
        deposit(interest);
    }

    class CurrentAccount extends Account {
        private double minimumBalance;
        private double serviceCharge;

        public CurrentAccount(String customerName, int accountName, double initialBalance, double minimumBalance, double serviceCharge) {
            super(customerName, accountName, initialBalance);
            this.minimumBalance = minimumBalance;
            this.serviceCharge = serviceCharge;
        }

        public void checkMinimumBalance() {
            if (getBalance() < minimumBalance) {
                System.out.println("Balance is below minimum");
                balance -= serviceCharge;
            }
        }
    }
}
```

```

S.O.p("Deducted service charge "+ serviceCharge);
S.O.p("Balance after deduction is "+ balance);
}
}

public class Bank {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        S.O.p("enter customer name");
        String name = sc.nextLine();
        S.O.p("enter acc no");
        int acc no = sc.nextInt();
        S.O.p("enter initial balance");
        double balance = sc.nextDouble();
        S.O.p("enter minimum balance");
        double minimumBalance = sc.nextDouble();
        S.O.p("enter interest rate");
        double interestRate = sc.nextDouble();
        S.O.p("enter service charge");
        double serviceCharge = sc.nextDouble();
        S.O.p("Enter choice: In 1. Current acc In 2. Savings acc");

        int ch = sc.nextInt();
        S.O.p("Customer name is " + name +
            " In Account Number " + acc no
            + " In Shoroda");

        switch(ch) {
            case(1):
                S.O.p("account is current type");
                CurrentAccount ca = new CurrentAccount(
                    name, acc no, balance, minimumBalance,
                    serviceCharge);

                do {
                    S.O.p("Enter choice: In 1. deposit In 2. withdraw In 3. display balance");

                    int c = sc.nextInt();
                    ca.checkMinimumBalance();
                    if (c == 1) {
                        S.O.p("Enter amount to be deposited");
                        double amt = sc.nextDouble();
                        ca.deposit(amt);
                    }
                    else if (c == 2) {
                        S.O.p("Enter amount to be withdrawn");
                        double amt = sc.nextDouble();
                        ca.withdraw(amt);
                    }
                    else if (c == 3) {
                        ca.displayBalance();
                    }
                } while (true);

                case(2):
                    S.O.p("account is saving type");
                    SavingsAccount sa = new SavingsAccount(
                        name, acc no, balance, interestRate);

                    do {
                        S.O.p("Enter choice: In 1. deposit In 2. withdraw In 3. display balance");

                        int c1 = sc.nextInt();
                        if (c1 == 1) {
                            S.O.p("Enter amount to be deposited");
                            double amt = sc.nextDouble();
                            sa.deposit(amt);
                        }
                    }
                } while (true);
            }
        }
    }
}

```

```

else if (c1 == 2) {
    S.O.p("Enter amount to be deposited");
    double amt = sc.nextDouble();
    sa.withdraw(amt);
}
else if (c1 == 3) {
    sa.computeAndDepositInterest();
    sa.displayBalance();
}
else {
    System.exit(0);
}
} while (true);
}

// Output
enter customer name : Shoroda
enter acc no : 45982
enter initial balance : 25000
enter minimum balance : 5000
enter interest rate : 5
enter service charge : 100
enter choice : 1 Current acc
                2 Savings acc
1
Customer name : Shoroda
account number : 45982
account is current type
enter choice 1 deposit
                2 withdraw
                3 display balance
1
enter amount to be deposited
10000
Deposited 10000
2
enter amount to be withdrawn
3000
withdrawn 3000
3
Current Balance is 32000

```

## CODE:

```
import java.util.Scanner;

class Account {
    private String customer_name;
    private int acc_no;
    protected double balance;

    public Account(String customer_name, int acc_no, double balance) {
        this.customer_name = customer_name;
        this.acc_no = acc_no;
        this.balance = balance;
    }

    public double getBalance() {
        return 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<=getBalance()){
            balance-=amount;
            System.out.println("withdrew:"+amount + " balance is:"+ balance);
        }
        else
            System.out.println("Insufficient funds!!");
    }

    public void displayBalance(){
        System.out.println("Current Balance: " + balance);
    }
}

class SavingsAccount extends Account {
    private double interestRate;

    public SavingsAccount(String customerName, int accountNumber, double initialBalance,
double interestRate) {
```

```

        super(customerName, accountNumber, initialBalance);
        this.interestRate = interestRate;
    }

    public void computeAndDepositInterest() {
        double interest = getBalance() * interestRate / 100;
        deposit(interest);
    }
}

class CurrentAccount extends Account {
    private double minimumBalance;
    private double serviceCharge;

    public CurrentAccount(String customerName, int accountNumber, double initialBalance,
double minimumBalance, double serviceCharge) {
        super(customerName, accountNumber, initialBalance);
        this.minimumBalance = minimumBalance;
        this.serviceCharge = serviceCharge;
    }

    public void checkMinimumBalance() {
        if (getBalance() < minimumBalance) {
            System.out.println("Balance is below minimum");
            balance-=serviceCharge;
            System.out.println("Deducted service charge:" +serviceCharge);
            System.out.println("Balance after deduction is:"+balance);
        }
    }
}

public class Bank {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("enter customer name:");
        String name=sc.nextLine();
        System.out.println("enter accno:");
        int acc_no=sc.nextInt();
        System.out.println("enter initial balance:");
        double balance=sc.nextDouble();
        System.out.println("enter minimum balance:");
        double minimum_balance=sc.nextDouble();
        System.out.println("enter interest rate:");
        double interest_rate=sc.nextDouble();
        System.out.println("enter service charge:");
        double service_charge=sc.nextDouble();
        System.out.println("Enter choice:\n 1.Current acc\n 2.Savings acc");
        int ch=sc.nextInt();
        System.out.println("Customer name is:"+ name+"\nAccount

```

```
number:"+acc_no+"\nBhoomika BG-1BM23CS067");
```

```
    switch(ch){
        case(1):
            System.out.println("account is current type");
            CurrentAccount ca = new
CurrentAccount(name,acc_no,balance,minimum_balance,service_charge);
            do{ System.out.println("enter choice:\n 1.deposit\n 2.withdraw\n 3.display balance");
                int c=sc.nextInt();
                ca.checkMinimumBalance();
                if(c==1){
                    System.out.println("enter amount to be deposited:");
                    double amt=sc.nextDouble();
                    ca.deposit(amt);}
                else if(c==2){
                    System.out.println("enter amount to withdraw:");
                    double amt=sc.nextDouble();
                    ca.withdraw(amt);}
                else if(c==3){
                    ca.displayBalance();}
                else
                    System.exit(0);
            }while(true);

        case(2):
            System.out.println("account is savings type");
            SavingsAccount sa=new SavingsAccount(name,acc_no,balance,interest_rate);
            do{ System.out.println("enter choice:\n 1.deposit\n 2.withdraw\n 3.display
balance");
                int c1=sc.nextInt();
                if(c1==1){
                    System.out.println("enter amount to be deposited:");
                    double amt=sc.nextDouble();
                    sa.deposit(amt);}
                else if(c1==2){
                    System.out.println("enter amount to withdraw:");
                    double amt=sc.nextDouble();
                    sa.withdraw(amt);}
                else if(c1==3){
                    sa.computeAndDepositInterest();
                    sa.displayBalance();}
                else{
                    System.exit(0);
                }
            } while(true);
    }
}
```



```
}
```

## OUTPUT:

```
C:\Users\Admin\Documents\23cs310>javac Bank.java
C:\Users\Admin\Documents\23cs310>java Bank
enter customer name:
Sharada
enter accno:
45982
enter initial balance:
25000
enter minimum balance:
5000
enter interest rate:
5
enter service charge:
100
Enter choice:
1.Current acc
2.Savings acc
1
Customer name is:Sharada
Account number:45982
Sharada
account is current type
enter choice:
1.deposit
2.withdraw
3.display balance
1
enter amount to be deposited:
10000
Deposited: 10000.0
enter choice:
1.deposit
2.withdraw
3.display balance
2
enter amount to withdraw:
3000
withdrew:3000.0 balance is:32000.0
enter choice:
1.deposit
2.withdraw
3.display balance
3
Current Balance: 32000.0
```

```
C:\Users\Admin\Documents\23cs310>java Bank
enter customer name:
Sharada\
enter accno:
467382
enter initial balance:
24000
enter minimum balance:
5000
enter interest rate:
5
enter service charge:
100
Enter choice:
1.Current acc
2.Savings acc
2
Customer name is:Sharada\
Account number:467382
Sharada
account is savings type
enter choice:
1.deposit
2.withdraw
3.display balance
1
enter amount to be deposited:
2000
Deposited: 2000.0
enter choice:
1.deposit
2.withdraw
3.display balance
2
enter amount to withdraw:
500
withdrew:500.0 balance is:25500.0
enter choice:
1.deposit
2.withdraw
3.display balance
3
Deposited: 1275.0
Current Balance: 26775.0
```

## Program 6

Create a package CIE which has two classes- Student and Internals. The class Personal has members like usn , name, sem. The class internals has an array that stores the internal marks scored in five courses of the current semester of the student. Create another package SEE which has the class External which is a derived class of Student. This class has an array that stores the SEE marks scored in five courses of the current semester of the student. Import the two packages in a file that declares the final marks of n students in all five courses.

Program VI

Q. Create a package CIE which has two classes- Student and Internals. The class Internals has an array that stores the internal marks scored in five courses of the current semester of the student. Create another package SEE which has the class External which is a derived class of Student. This class has an array that stores the SEE marks scored in five courses of the current semester of the student. Import the two packages in a file that declares the final marks of n students in all five courses.

```
package CIE;

import java.util.Scanner;

public class Student {
    protected String usn;
    protected String name;
    protected int sem;

    public void inputStudentDetails() {
        Scanner scanner = new Scanner(System.in);
        S.O.p("Enter USN: ");
        usn = scanner.nextLine();
        S.O.p("Enter Name: ");
        name = scanner.nextLine();
        S.O.p("Enter Semester: ");
        sem = scanner.nextInt();
    }

    public void displayStudentDetails() {
        S.O.p("USN: " + usn);
        S.O.p("Name: " + name);
        S.O.p("Semester: " + sem);
    }
}

package CIE;

import java.util.Scanner;

public class Internals extends Student {
    protected int[] marks = new int[5];

    public void inputCIEMarks() {
        Scanner scanner = new Scanner(System.in);
        S.O.p("Enter Internal marks for 5 courses");
        for (int i = 0; i < 5; i++) {
            S.O.p("Enter marks for Course " + (i+1) + ": ");
            marks[i] = scanner.nextInt();
        }
    }

    public void displayCIEMarks() {
        S.O.p("Internal Marks for 5 courses");
        for (int i = 0; i < 5; i++) {
            S.O.p("Course " + (i+1) + ": " + marks[i]);
        }
    }
}
```

```
package SEE;

import CIE.Internals;
import java.util.Scanner;

public class External extends Internals {
    protected int[] externalMarks = new int[5];
    protected int[] finalMarks = new int[5];

    public External() {
        marks = new int[5];
        externalMarks = new int[5];
        finalMarks = new int[5];
    }

    public void inputSEEMarks() {
        Scanner scanner = new Scanner(System.in);
        S.O.p("Enter External marks for 5 courses");
        for (int i = 0; i < 5; i++) {
            S.O.p("Enter marks for Course " + (i+1) + ": ");
            externalMarks[i] = scanner.nextInt();
        }
    }

    public void calculateFinalMarks() {
        for (int i = 0; i < 5; i++) {
            finalMarks[i] = marks[i] + externalMarks[i];
        }
    }
}

package SEE;

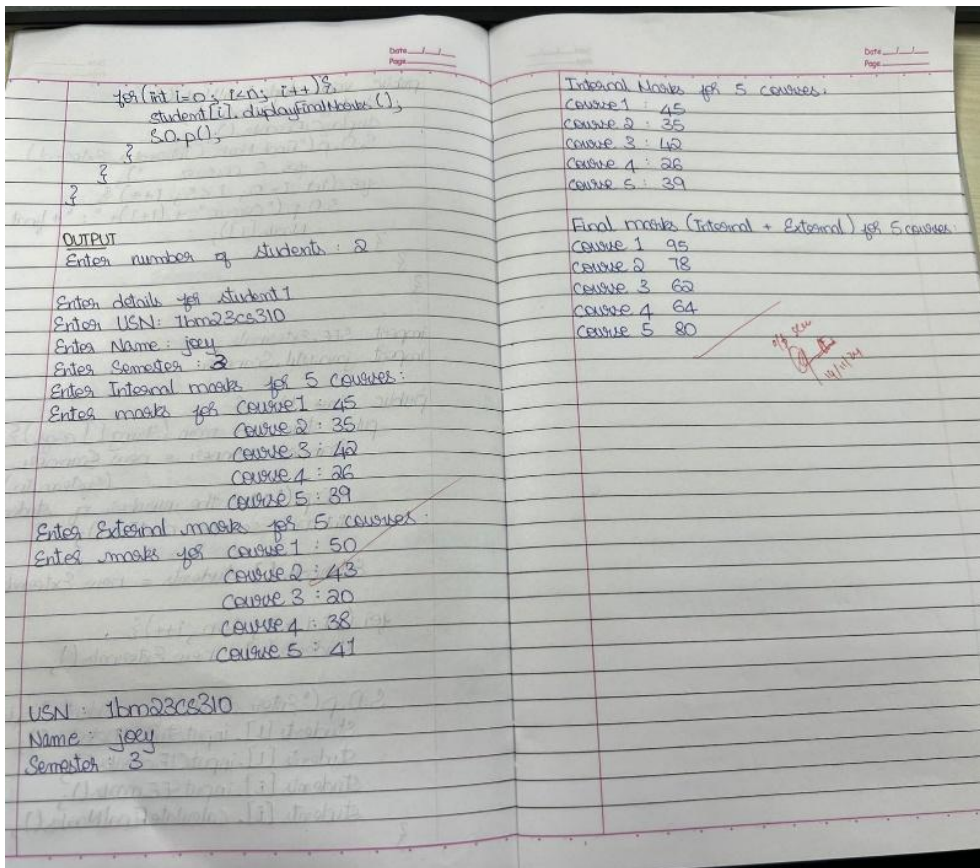
import CIE.Internals;
import CIE.Student;
import SEE.External;
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        S.O.p("Enter the number of students");
        int n = scanner.nextInt();

        External[] students = new External[n];

        for (int i = 0; i < n; i++) {
            students[i] = new External();
            S.O.p("Enter details for student " + (i+1));
            students[i].inputStudentDetails();
            students[i].inputCIEMarks();
            students[i].inputSEEMarks();
            students[i].calculateFinalMarks();
        }
    }
}
```





## CODE:

```

package CIE;
import java.util.Scanner;

public class Student {
    protected String usn;
    protected String name;
    protected int sem;

    public void inputStudentDetails() {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter USN: ");
        usn = scanner.nextLine();
        System.out.print("Enter Name: ");
        name = scanner.nextLine();
        System.out.print("Enter Semester: ");
        sem = scanner.nextInt();
    }

    public void displayStudentDetails() {
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        System.out.println("Semester: " + sem);
    }
}
  
```

```

}
package CIE;
import java.util.Scanner;

public class Internals extends Student {
    protected int[] marks = new int[5];
    public void inputCIEmarks() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter Internal marks for 5 courses:");
        for (int i = 0; i < 5; i++) {
            System.out.print("Enter marks for Course " + (i + 1) + ": ");
            marks[i] = scanner.nextInt();
        }
    }
    public void displayCIEmarks() {
        System.out.println("Internal Marks for 5 courses:");
        for (int i = 0; i < 5; i++) {
            System.out.println("Course " + (i + 1) + ": " + marks[i]);
        }
    }
}

package SEE;
import CIE.Internals;
import java.util.Scanner;

public class Externals extends Internals {
    protected int[] externalMarks = new int[5];
    protected int[] finalMarks = new int[5];

    public Externals() {
        marks = new int[5];
        externalMarks = new int[5];
        finalMarks = new int[5];
    }
    public void inputSEEmarks() {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter External marks for 5 courses:");
        for (int i = 0; i < 5; i++) {
            System.out.print("Enter marks for Course " + (i + 1) + ": ");
            externalMarks[i] = scanner.nextInt();
        }
    }
    public void calculateFinalMarks() {
        for (int i = 0; i < 5; i++) {
            finalMarks[i] = marks[i] + externalMarks[i];
        }
    }
}

```

```

    public void displayFinalMarks() {
        displayStudentDetails();
        displayCIEmarks();
        System.out.println("Final Marks (Internal + External) for 5 courses:");
        for (int i = 0; i < 5; i++) {
            System.out.println("Course " + (i + 1) + ": " + finalMarks[i]);
        }
    }
}
import SEE.Externals;
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of students: ");
        int n = scanner.nextInt();

        Externals[] students = new Externals[n];

        for (int i = 0; i < n; i++) {
            students[i] = new Externals();
            System.out.println("Enter details for student " + (i + 1));
            students[i].inputStudentDetails();
            students[i].inputCIEmarks();
            students[i].inputSEEmarks();
            students[i].calculateFinalMarks();
        }
        for(int i=0; i<n; i++){
            students[i].displayFinalMarks();
            System.out.println();
        }
    }
}

```

## OUTPUT:

```

C:\Users\Admin\Documents\23cs310\Lab6 packages>java Main.java
Enter the number of students: 1
Enter details for student 1
Enter USN: 1bm23cs310
Enter Name: joey
Enter Semester: 3
Enter Internal marks for 5 courses:
Enter marks for Course 1: 45
Enter marks for Course 2: 35
Enter marks for Course 3: 42
Enter marks for Course 4: 26
Enter marks for Course 5: 39
Enter External marks for 5 courses:
Enter marks for Course 1: 50
Enter marks for Course 2: 43
Enter marks for Course 3: 20
Enter marks for Course 4: 38
Enter marks for Course 5: 41
USN: 1bm23cs310
Name: joey
Semester: 3
Internal Marks for 5 courses:
Course 1: 45
Course 2: 35
Course 3: 42
Course 4: 26
Course 5: 39
Final Marks (Internal + External) for 5 courses:
Course 1: 95
Course 2: 78
Course 3: 62
Course 4: 64
Course 5: 80

```

## Program 7

Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called "Father" and derived class called "Son" which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge() when the input age < 0. In Son class, implement a constructor that uses both father and son's age and throws an exception if son's age is >= father's age.

Lab Program VII

Q. Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called "Father" and derived class "Son" which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge() when the input age < 0. In Son class, implement a constructor that uses both father and son's age and throws an exception if son's age is >= father's age.

import java.util.Scanner;

class WrongAge extends Exception {  
String message;  
WrongAge(String message) {  
this.message = message;  
}  
public String toString() {  
return "WrongAge Exception: " + message;  
}  
}

class Father {  
int Age;  
Father(int age) throws WrongAge {  
if (age < 0) {  
throw new WrongAge("Father's age cannot be negative!");  
}  
Age = age;  
}  
}

class Son extends Father {  
int Age;  
Son(int fAge, int sAge) throws WrongAge {  
super(fAge);  
if (sAge < 0) {  
throw new WrongAge("Son's age cannot be negative!");  
}  
if (sAge >= fAge) {  
throw new WrongAge("Son's age cannot be greater than or equal to Father's age!");  
}  
this.Age = sAge;  
}  
}

public class LP7 {  
public static void main(String[] args) {  
try {  
Father father = new Father(40);  
Son son = new Son(father.Age, 20);  
S.O.P(father.Age + " + " + son.Age);  
Father father2 = new Father(-5);  
catch (WrongAge e) {  
S.O.P(e);  
}

try {  
Son son2 = new Son(25, 40);  
catch (WrongAge e) {  
S.O.P(e);  
}

try {  
Son son3 = new Son(50, -10);  
catch (WrongAge e) {  
S.O.P(e);  
}

output  
Father's age : 40  
Son's age : 20  
WrongAge Exception : Father's Age cannot be negative!  
WrongAge Exception : Son's Age cannot be greater than or equal to Father's Age  
WrongAge Exception : Son's Age cannot be negative

offsecm  
21/11/24

## CODE:

```
class WrongAge extends Exception {
    String message;
    WrongAge(String message) {
        this.message = message;
    }

    public String toString() {
        return "WrongAge Exception: " + message;
    }
}

class Father {
    int fAge;
    Father(int age) throws WrongAge {
        if (age < 0) {
            throw new WrongAge("Father's age cannot be negative!");
        }
        fAge = age;
    }
}

class Son extends Father {
    int sAge;

    Son(int fAge, int sAge) throws WrongAge {
        super(fAge);
        if (sAge < 0) {
            throw new WrongAge("Son's age cannot be negative!");
        }
        if (sAge >= fAge) {
            throw new WrongAge("Son's age cannot be greater than or equal to Father's age!");
        }
        this.sAge = sAge;
    }
}

public class FSAge {
    public static void main(String[] args) {
        try {
            Father father1 = new Father(40);
            Son son1 = new Son(40, 20);
            System.out.println("Father's age: " + father1.fAge + ", Son's age: " + son1.sAge);
            Father father2 = new Father(-5);
        }
        catch (WrongAge e) {
            System.out.println(e);
        }
    }
}
```

```
try {
    Son son2 = new Son(35, 40);
}
catch (WrongAge e) {
    System.out.println(e);
}

try {
    Son son3 = new Son(50, -10);
}
catch (WrongAge e) {
    System.out.println(e);
}
```

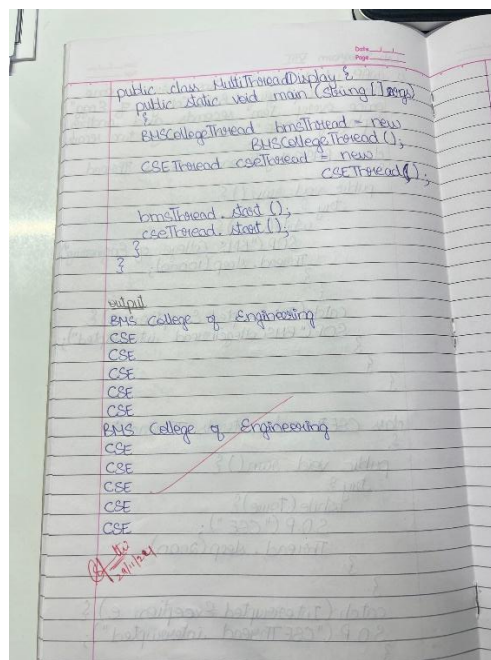
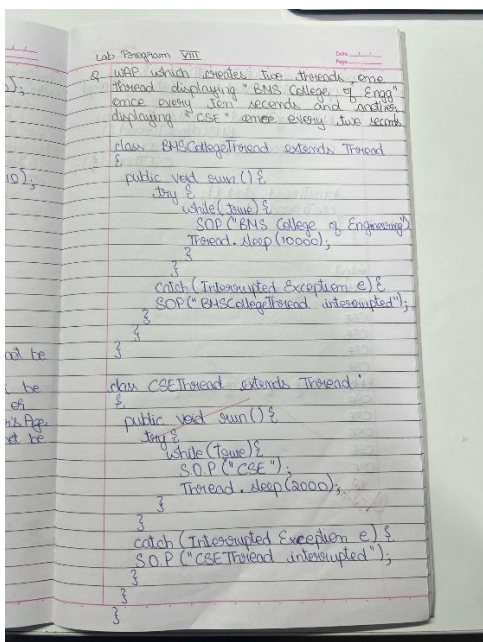
**OUTPUT:**

```
C:\Users\Admin\Documents\23cs310>javac FSAGE.java

C:\Users\Admin\Documents\23cs310>java FSAGE.java
Father's age: 40, Son's age: 20
WrongAge Exception: Father's age cannot be negative!
WrongAge Exception: Son's age cannot be greater than or equal to Father's age!
WrongAge Exception: Son's age cannot be negative!
```

## Program 8

Write a program which creates two threads, one thread displaying “BMS College of Engineering” once every ten seconds and another displaying “CSE” once every two seconds.



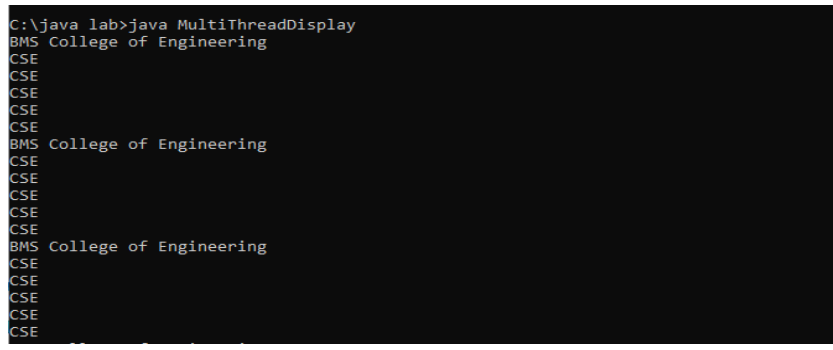
## CODE:

```
class BMSCollegeThread extends Thread {
    public void run() {
        try {
            while (true) {
                System.out.println("BMS College of Engineering");
                Thread.sleep(10000); // 10 seconds
            }
        } catch (InterruptedException e) {
            System.out.println("BMSCollegeThread interrupted.");
        }
    }
}
```

```
class CSEThread extends Thread {
    public void run() {
        try {
            while (true) {
                System.out.println("CSE");
                Thread.sleep(2000); // 2 seconds
            }
        } catch (InterruptedException e) {
            System.out.println("CSEThread interrupted.");
        }
    }
}
```

```
public class MultiThreadDisplay {
    public static void main(String[] args) {
        BMSCollegeThread bmsThread = new BMSCollegeThread();
        CSEThread cseThread = new CSEThread();
        bmsThread.start();
        cseThread.start();
    }
}
```

## OUTPUT:



```
C:\java lab>java MultiThreadDisplay
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
```



## Program 9

Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException. If Num2 were Zero, the program would throw an Arithmetic Exception Display the exception in a message dialog box.

9/12 Program IX

Q. Write a program that creates a user interface to perform integer division. The user enters two numbers in the text fields Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException. If Num2 were zero, the program would throw an Arithmetic Exception. Display the exception in a message dialog box.

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

public class DivisionMain extends JFrame
implements ActionListener {
    Text field
    num1, num2;
    Button dResult;
    Label outResult;
    String out = "";
    double resultNum;
    int flag = 0;

    public DivisionMain() {
        setLayout(new FlowLayout());
        dResult = new Button("Result");
        Label num1Label = new Label
        ("Number 1:", Label.RIGHT);
```

```
Label num2Label = new Label
("Number 2:", Label.RIGHT);
num1 = new TextField(5);
num2 = new TextField(5);

outResult = new Label("Result:", Label.RIGHT);

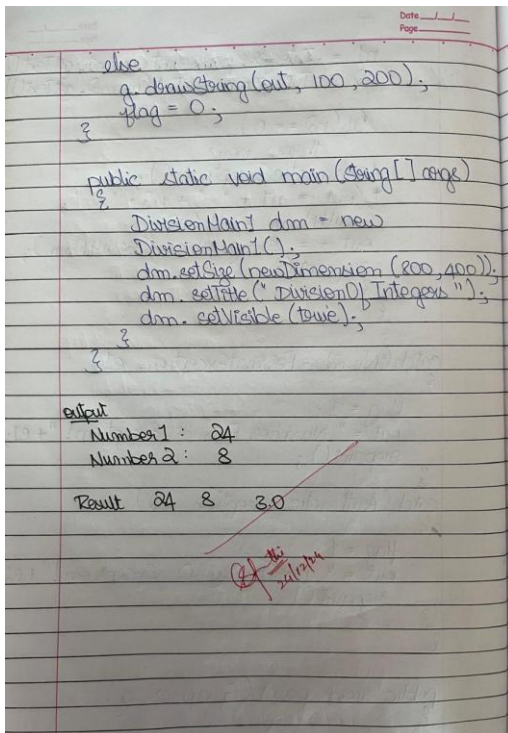
add(num1Label);
add(num1);
add(num2Label);
add(num2);
add(dResult);
add(outResult);

num1.addActionListener(this);
num2.addActionListener(this);
dResult.addActionListener(this);
addWindowListener(new WindowAdapter() {
    public void windowClosing
(WindowEvent we) {
        System.exit(0);
    }
});

public void actionPerformed(ActionEvent ae) {
    int n1, n2;
    try {
        if (ae.getSource() == dResult) {
            n1 = Integer.parseInt(num1.getText());
            n2 = Integer.parseInt(num2.getText());

            if (n2 == 0)
                throw new
                ArithmeticException();
            * / out = n1 * " " + n2 * " ";
            resultNum = n1 / n2;
            out += String.valueOf(resultNum);
            repaint();
        }
    } catch (NumberFormatException e1) {
        flag = 1;
        out = "Number Format Exception! " + e1;
        repaint();
    } catch (ArithmeticException e2) {
        flag = 1;
        out = "Divide by 0 Exception! " + e2;
        repaint();
    }

    public void paint(Graphics g) {
        if (flag == 0)
            g.drawString(out, outResult.getX() +
            outResult.getWidth(), outResult.getY() +
            outResult.getHeight());
    }
}
```



## CODE:

```

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

public class DivisionMain1 extends Frame implements ActionListener
{
    TextField num1, num2;
    Button dResult;

    Label outResult;
    String out="";
    double resultNum;
    int flag=0;

    public DivisionMain1()
    {
        setLayout(new FlowLayout());

        dResult = new Button("RESULT");

        Label number1 = new Label("Number 1:", Label.RIGHT);
        Label number2 = new Label("Number 2:", Label.RIGHT);
        num1 = new TextField(5);

```

```
num2=new TextField(5);  
outResult = new Label("Result:",Label.RIGHT);
```

```
add(number1);  
add(num1);  
add(number2);  
add(num2);  
add(dResult);  
add(outResult);
```

```
num1.addActionListener(this);  
num2.addActionListener(this);  
dResult.addActionListener(this);  
addWindowListener(new WindowAdapter()  
{  
public void windowClosing(WindowEvent we)  
{  
System.exit(0);  
}  
});  
}  
public void actionPerformed(ActionEvent ae)  
{  
int n1,n2;  
try  
{  
if (ae.getSource() == dResult)  
{  
n1=Integer.parseInt(num1.getText());  
n2=Integer.parseInt(num2.getText());  
  
/*if(n2==0)  
throw new ArithmeticException();  
*/ out=n1+" "+n2+" ";
```

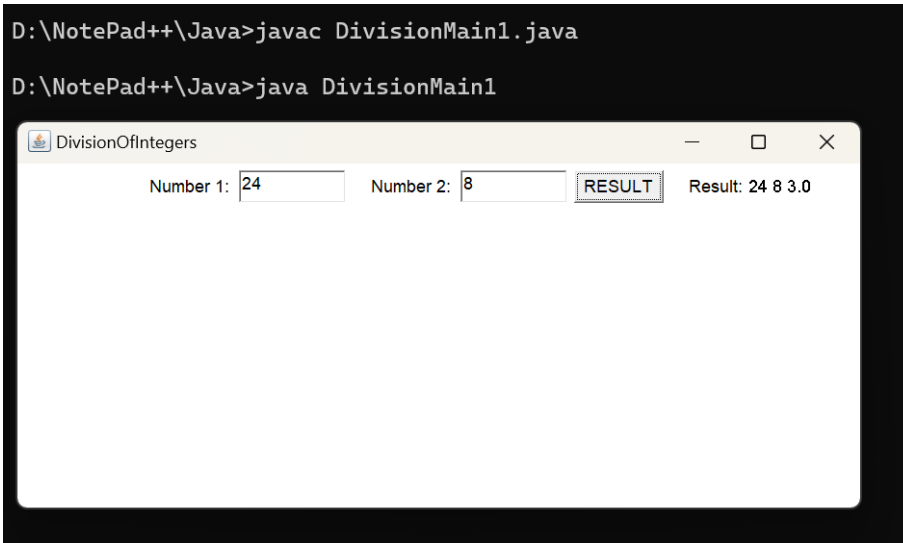
```

resultNum=n1/n2;
out+=String.valueOf(resultNum);
repaint();
}
}
catch(NumberFormatException e1)
{
flag=1;
out="Number Format Exception! "+e1;
repaint();
}
catch(ArithmeticException e2)
{
flag=1;
out="Divide by 0 Exception! "+e2;
repaint();
}
}
public void paint(Graphics g)
{
if(flag==0)
g.drawString(out,outResult.getX()+outResult.getWidth(),outResult.getY()+outResult.
getHeight()-8);
else g.drawString(out,100,200);
flag=0;
}
public static void main(String[] args)
{
DivisionMain1 dm=new DivisionMain1();
dm.setSize(new Dimension(800,400));
dm.setTitle("DivisionOfIntegers");
dm.setVisible(true);
}

```

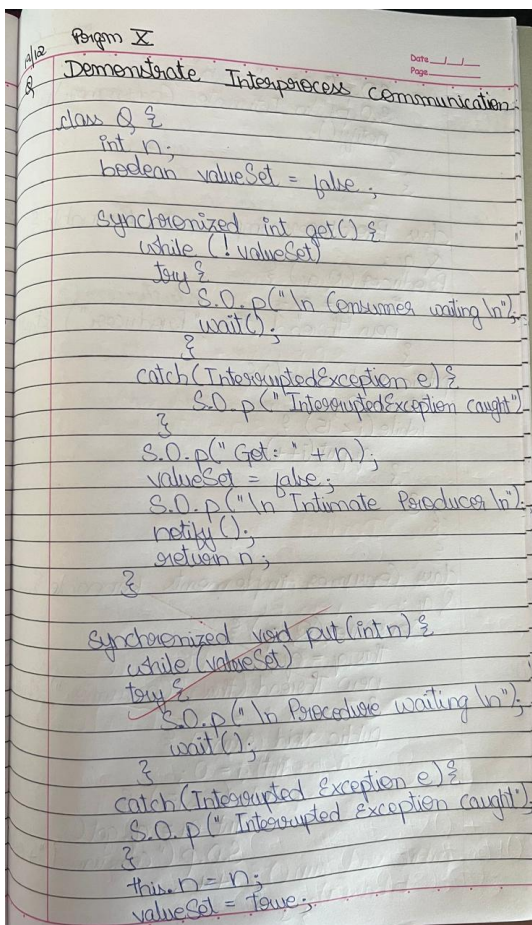
```
}
```

## OUTPUT:



## Program 10

Demonstrate Inter process Communication and deadlock





```

S.O.p("Pit: " + n);
S.O.p("In Intimate Consumer\n");
notify();
}
}

class Producer implements Runnable {
    Q q;
    Producer(Q q) {
        this.q = q;
        new Thread(this, "Producer").start();
    }
    public void run() {
        int i = 0;
        while (i < 15) {
            q.put(i++);
        }
    }
}

class Consumer implements Runnable {
    Q q;
    Consumer(Q q) {
        this.q = q;
        new Thread(this, "Consumer").start();
    }
    public void run() {
        int i = 0;
        while (i < 15) {
            int o1 = q.get();
            S.O.p("Consumed " + o1);
            i++;
        }
    }
}

```

class PCFixed {

```

public static void main(String args[]) {
    Q q = new Q();
    new Producer(q);
    new Consumer(q);
    S.O.p("Press Control-C to stop");
}
}

```

Demonstration of deadlock

```

class A {
    synchronized void go(B b) {
        String name = Thread.currentThread().getName();
        S.O.p(name + " entered A.go");
        try {
            Thread.sleep(1000);
        } catch (Exception e) {
            S.O.p("A Interrupted");
        }
        S.O.p(name + " Trying to call B.last");
        b.last();
    }
    synchronized void last() {
        S.O.p("Inside A.last");
    }
}

```

P.T.O

```

class B {
    synchronized void bar(A a) {
        String name = Thread.currentThread().getName();
        S.O.p(name + " entered B.bar");
        try {
            Thread.sleep(1000);
        } catch (Exception e) {
            S.O.p("B Interrupted");
        }
        S.O.p(name + " Trying to call A.last");
        a.last();
    }
    synchronized void last() {
        S.O.p("Inside A.last");
    }
}

class Deadlock implements Runnable {
    A a = new A();
    B b = new B();
    Deadlock() {
        Thread.currentThread().setName("Main Thread");
        Thread t = new Thread(this);
        t.start();
        a.go(b);
        S.O.p("Back in main thread");
    }
    public void run() {
        b.bar(a);
    }
}

```

```

S.O.p("Back in other thread");
}

public static void main(String args[]) {
    new Deadlock();
}
}

```

23/12/14

**CODE:**

```
class Q {  
    int n;  
    boolean valueSet = false;  
  
    synchronized int get() {  
        while(!valueSet)  
        try {  
            System.out.println("\nConsumer waiting\n");  
            wait();  
        }  
        catch(InterruptedException e) {  
            System.out.println("InterruptedException caught");  
        }  
        System.out.println("Got: " + n);  
        valueSet = false;  
        System.out.println("\nIntimate Producer\n");  
        notify();  
        return n;  
    }  
  
    synchronized void put(int n) {  
        while(valueSet)  
        try {  
            System.out.println("\nProducer waiting\n");  
            wait();  
        }  
        catch(InterruptedException e) {  
            System.out.println("InterruptedException caught");  
        }  
    }  
}
```



```

this.n = n;
valueSet = true;
System.out.println("Put: " + n);
System.out.println("\nIntimate Consumer\n");
notify();
}
}

```

```

class Producer implements Runnable {
    Q q;
    Producer(Q q) {
        this.q = q;
        new Thread(this, "Producer").start();
    }
    public void run() {
        int i = 0;
        while(i<15) {
            q.put(i++);
        }
    }
}

```

```

class Consumer implements Runnable {
    Q q;
    Consumer(Q q) {
        this.q = q;
        new Thread(this, "Consumer").start();
    }
    public void run() {

```

```

        int i=0;
while(i<15) {
int r=q.get();
System.out.println("consumed:"+r);
i++;
}
}
}

class PCFixed {
public static void main(String args[]) {
Q q = new Q();
new Producer(q);
new Consumer(q);
System.out.println("Press Control-C to stop.");
}
}

```

## ii. Demonstration of deadlock

```

class A
{
    synchronized void foo(B b)
    {
String name = Thread.currentThread().getName();
System.out.println(name + " entered A.foo");

try {
Thread.sleep(1000);
}
catch(Exception e) {
System.out.println("A Interrupted");
}
}
}

```

```

        System.out.println(name + " trying to call B.last()");
b.last();
{
    synchronized void last() {
System.out.println("Inside A.last");
    }
}

class B {
    synchronized void bar(A a) {
        String name = Thread.currentThread().getName();
        System.out.println(name + " entered B.bar");

        try {
            Thread.sleep(1000);
        }

        catch(Exception e) {
            System.out.println("B Interrupted");
        }

        System.out.println(name + " trying to call A.last()");
        a.last();
        {
            synchronized void last() {
                System.out.println("Inside A.last");
            }
        }
    }
}

```

```

class Deadlock implements Runnable
{
    A a = new A();
    B b = new B();

    Deadlock( ) {
        Thread.currentThread().setName("MainThread");

        Thread t = new Thread(this, "RacingThread");
        t.start(); a.foo(b); // get lock on a in this thread.

        System.out.println("Back in main thread");
    }
}

```

```
    }  
    public void run() { b.bar(a); // get lock on b in other thread.  
        System.out.println("Back in other thread");  
    }  
    public static void main(String args[]) { new Deadlock(); }
```

---

---

