VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

KISHEN DEEPAK (1BM23CS152)

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 to 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 (23CS3PCOOJ)" carried out by **KISHEN DEEPAK (1BM23CS152),** who is bonafide 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. The Lab report has been approved as it satisfies the academic requirements in respect of an Object Oriented Java Programming (23CS3PCOOJ) work prescribed for the said degree.

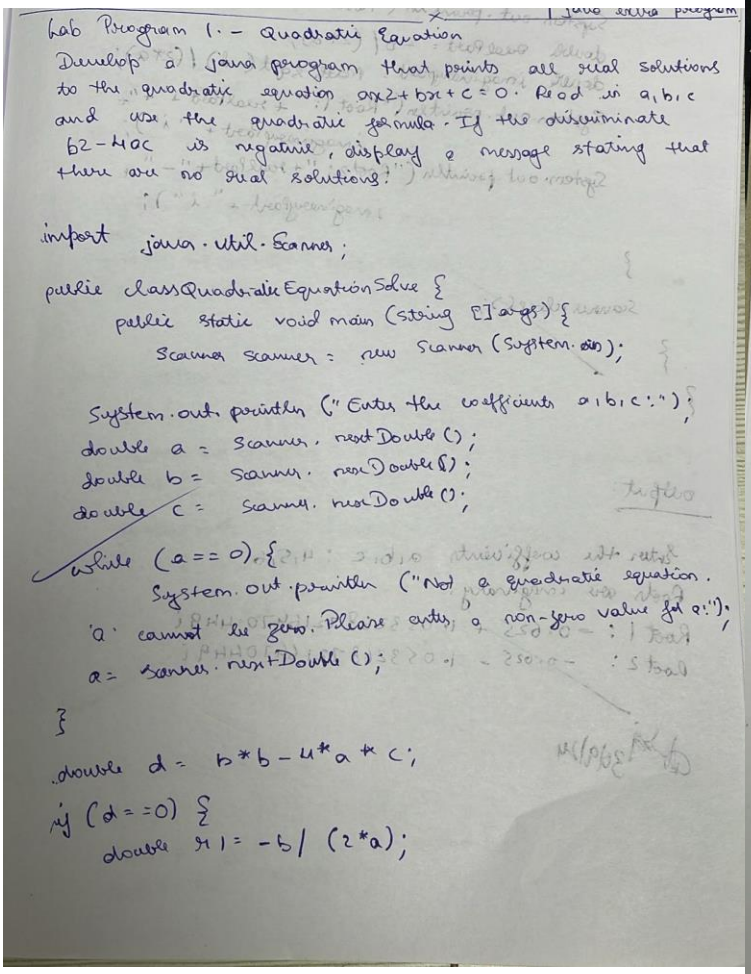
Lab faculty Incharge Name Assistant Professor Department of CSE, BMSCE	Dr. Seema Patil Professor Department of CSE, BMSCE

Index

Github Link:
https://github.com/Kishendeepak8/java-lab-programs

Program 1

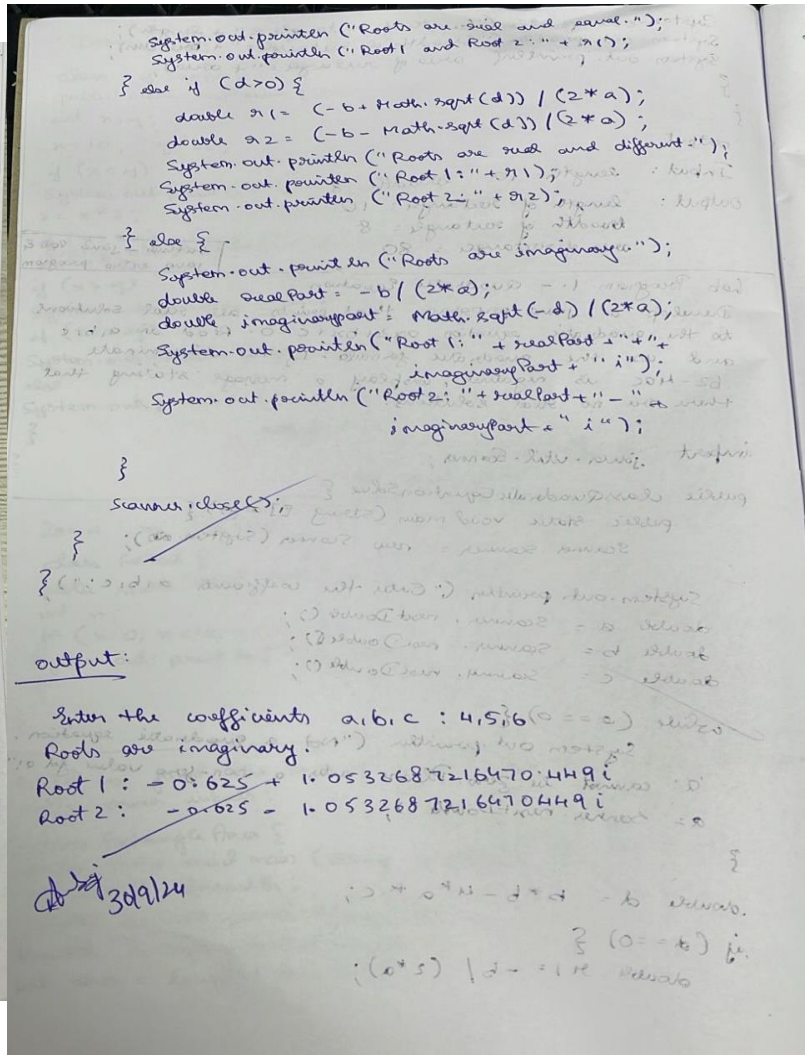
Implement Quadratic Equation



Lab Program 1 - Quadratic Equation

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.

```
import java.util.Scanner;

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

        System.out.println("Enter the coefficients a,b,c:");
        double a = scanner.nextDouble();
        double b = scanner.nextDouble();
        double c = scanner.nextDouble();

        while (a == 0) {
            System.out.println("Not a quadratic equation.
            'a' cannot be zero. Please enter a non-zero value for a:");
            a = scanner.nextDouble();
        }

        double d = b*b - 4*a*c;
        if (d == 0) {
            double r1 = -b / (2*a);
```



```
            System.out.println("Roots are real and equal.");
            System.out.println("Root 1 and Root 2 : " + r1);
        } else if (d > 0) {
            double r1 = (-b + Math.sqrt(d)) / (2*a);
            double r2 = (-b - Math.sqrt(d)) / (2*a);
            System.out.println("Roots are real and different.");
            System.out.println("Root 1 : " + r1);
            System.out.println("Root 2 : " + r2);
        } else {
            System.out.println("Roots are imaginary.");
            double realPart = -b / (2*a);
            double imaginaryPart = Math.sqrt(-d) / (2*a);
            System.out.println("Root 1 : " + realPart + "+" +
                    imaginaryPart + "i");
            System.out.println("Root 2 : " + realPart + "-" +
                    imaginaryPart + "i");
        }
        scanner.close();
    }
}
```

output:

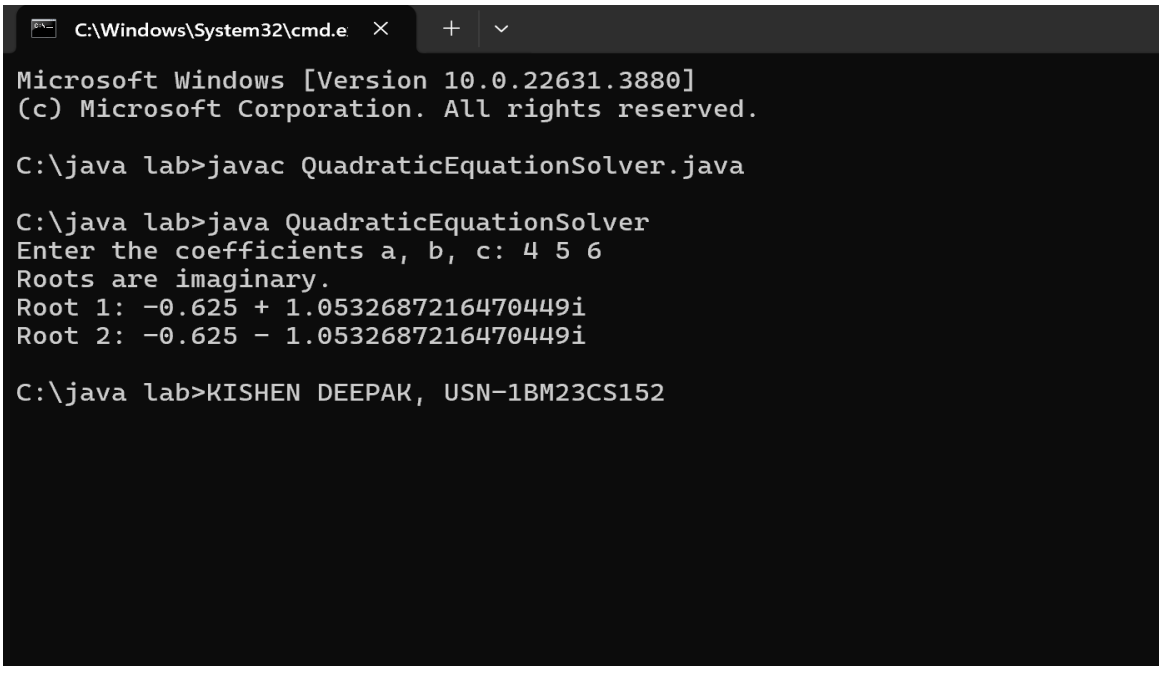
```
Enter the coefficients a,b,c : 4,5,6
Roots are imaginary.
Root 1 : -0.625 + 1.0532687216470449i
Root 2 : -0.625 - 1.0532687216470449i
```

30/9/24

CODE

```
import java.util.Scanner;

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

        System.out.print("Enter the coefficients a, b, c: ");
        double a = scanner.nextDouble();
        double b = scanner.nextDouble();
        double c = scanner.nextDouble();

        while (a == 0) {
            System.out.println("Not a quadratic equation. 'a' cannot be
zero. Please enter a non-zero value for a:");
            a = scanner.nextDouble();
        }

        double d = b * b - 4 * a * c;

        if (d == 0) {
            double r1 = -b / (2 * a);
            System.out.println("Roots are real and equal.");
            System.out.println("Root 1 and Root 2: " + r1);
        } else if (d > 0) {
            double r1 = (-b + Math.sqrt(d)) / (2 * a);
            double r2 = (-b - Math.sqrt(d)) / (2 * a);
```

```
            System.out.println("Roots are real and different.");
            System.out.println("Root 1: " + r1);
            System.out.println("Root 2: " + r2);
        } else {
            System.out.println("Roots are imaginary.");
            double realPart = -b / (2 * a);
            double imaginaryPart = Math.sqrt(-d) / (2 * a);
            System.out.println("Root 1: " + realPart + " + " +
imaginaryPart + "i");
            System.out.println("Root 2: " + realPart + " - " +
imaginaryPart + "i");
        }

        scanner.close();
    }
}
```



Program 2

SGPA CALCULATOR

Lab Program 2. 7/10/2024

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.

```
import java.util.Scanner;
public class Student {
    String name, usn;
    double SGPA;

    int [] marks = new int [4];
    int [] credits = new int [4];
    double [] gradepoints = new double [4];
    double total = 0, credittotal = 0;

    Scanner sc = new Scanner (System.in);
    void getStudentDetails () {
     System.out.println ("Enter name: ");
    name = sc.nextline ();
    System.out.println ("Enter USN:");
    USN = sc.nextline ();
    }

    void getMarks () {
        for (int j = 0; j<4; j++) {
        System.out.println ("Enter" + (i+1) + "subject marks:");
        marks[j] = sc.nextInt ();

        gradepoints [j] = (marks [j] /10) + 1;
        if (gradepoints[j] >10) {
            gradepoints[j] = 10;
    }
    }
}

void computeSGPA () {
    for (int j = 0; j < 4; j++) {
    total += gradepoints[j] * credits [j];
    credittotal += credits[j];
    }
    SGPA = total / credittotal;
}

void display () {
    System.out.println ("Name:" + name);
    System.out.println ("USN:" + USN);
    System.out.println ("SGPA: " + SGPA);
    }
    public static void main (String[] args) {
        Student s1 = new Student ();
    Student s2 = new Student ();
        s1.getStudentDetails ();
        s1.getMarks ();
        s1.computeSGPA ();
        s1.display ();
    }
}
```

output:

```
enter the details of student 1:
Enter USN : 1BM23CS152
Enter name: Arvind
Enter marks for 3 subjects:
marks for subject 1 : 20
marks for subject 2 : 30
marks for subject 3 : 35
Enter the details of student 2:
Enter USN : 1BM23CS165
Enter name: Kishan
Enter marks for 3 subjects :
marks for subject 1 : 89
marks for subject 2 : 90
marks for subject 3 : 95

Student details :
USN: 1BM23CS152
Name: Arvind
Subjects :
Subject marks : 20
credits : 4
```

grade : 0
Subject marks : 35
credits : 4
grade : 0
SGPA : 0.00

USN : 1BM23CS165
Name : Kishen
Subjects
Subject Marks : 89
credits : 4
grade : 9
Subject marks : 90
credits : 4
grade : 10
Subject marks : 95
credits : 4
grade : 10
SGPA : 9.62

KISHEN DEEPAK
1BM23CS152.

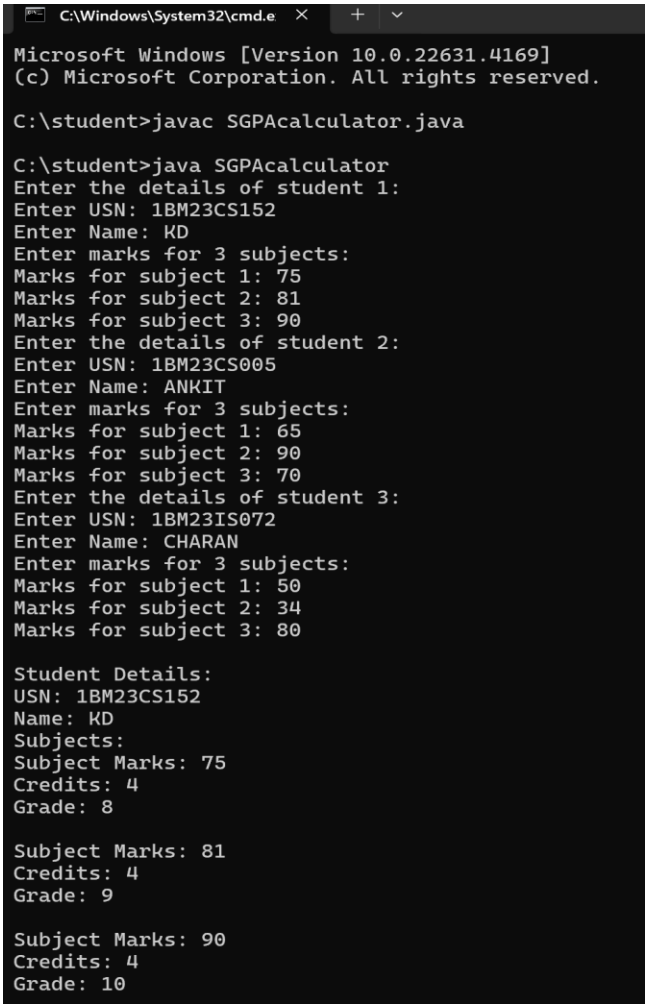
9/10/24

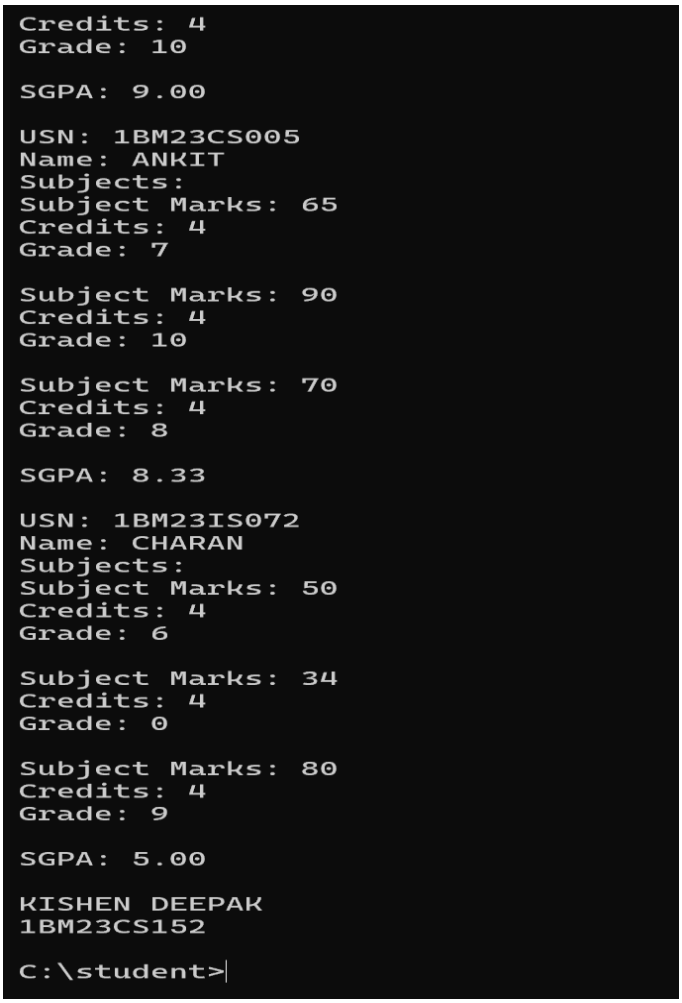
CODE: import java.util.Scanner;

```
class Subject {
    private int subjectMarks;
    private int credits;
    private int grade;

    public Subject(int subjectMarks, int credits) {
        this.subjectMarks = subjectMarks;
        this.credits = credits;
        this.grade = calculateGrade(subjectMarks);
    }

    private int calculateGrade(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 int getSubjectMarks() {
        return subjectMarks;
    }

    public int getCredits() {
        return credits;
    }

    public int getGrade() {
        return grade;
    }

    public void display() {
        System.out.println("Subject Marks: " + subjectMarks);
        System.out.println("Credits: " + credits);
        System.out.println("Grade: " + grade);
    }
}

class Student {
    private String usn;
    private String name;
    private Subject[] subjects;

    public Student() {
        subjects = new Subject[3];
    }

    void getDetails(Scanner sc) {
        System.out.print("Enter USN: ");
        usn = sc.next();
        System.out.print("Enter Name: ");
```

```
        name = sc.next();

        System.out.println("Enter marks for 3 subjects:");
        for (int i = 0; i < 3; i++) {
            System.out.print("Marks for subject " + (i + 1) + ": ");
            int marks = sc.nextInt();
            subjects[i] = new Subject(marks, 4);
        }
    }

    double calculateSGPA() {
        double totalPoints = 0;
        double totalCredits = 0;
        for (Subject subject : subjects) {
            totalPoints += subject.getGrade() * subject.getCredits();
            totalCredits += subject.getCredits();
        }
        return totalCredits > 0 ? totalPoints / totalCredits : 0;
    }

    void display() {
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        System.out.println("Subjects:");
        for (Subject subject : subjects) {
            subject.display();
            System.out.println();
        }
        System.out.printf("SGPA: %.2f%n", calculateSGPA());
    }
}

public class SGPAcalculator {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Student[] students = new Student[3];

        for (int j = 0; j < 3; j++) {
            System.out.println("Enter the details of student " + (j + 1) + ":");
            students[j] = new Student();
            students[j].getDetails(sc);
        }

        System.out.println("\nStudent Details:");
        for (Student student : students) {
            student.display();
            System.out.println();
        }
                                                System.out.println("KISHEN DEEPAK");
                                                  System.out.println("1BM23CS152");

        sc.close();
    }
}
```

OUTPUT:


C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.22631.4169]
(c) Microsoft Corporation. All rights reserved.
C:\student>javac SGPAcalculator.java
C:\student>java SGPAcalculator
Enter the details of student 1:
Enter USN: 1BM23CS152
Enter Name: KD
Enter marks for 3 subjects:
Marks for subject 1: 75
Marks for subject 2: 81
Marks for subject 3: 90
Enter the details of student 2:
Enter USN: 1BM23CS005
Enter Name: ANKIT
Enter marks for 3 subjects:
Marks for subject 1: 65
Marks for subject 2: 90
Marks for subject 3: 70
Enter the details of student 3:
Enter USN: 1BM23IS072
Enter Name: CHARAN
Enter marks for 3 subjects:
Marks for subject 1: 50
Marks for subject 2: 34
Marks for subject 3: 80
Student Details:
USN: 1BM23CS152
Name: KD
Subjects:
Subject Marks: 75
Credits: 4
Grade: 8
Subject Marks: 81
Credits: 4
Grade: 9
Subject Marks: 90
Credits: 4
Grade: 10


Credits: 4
Grade: 10
SGPA: 9.00
USN: 1BM23CS005
Name: ANKIT
Subjects:
Subject Marks: 65
Credits: 4
Grade: 7
Subject Marks: 90
Credits: 4
Grade: 10
Subject Marks: 70
Credits: 4
Grade: 8
SGPA: 8.33
USN: 1BM23IS072
Name: CHARAN
Subjects:
Subject Marks: 50
Credits: 4
Grade: 6
Subject Marks: 34
Credits: 4
Grade: 0
Subject Marks: 80
Credits: 4
Grade: 9
SGPA: 5.00
KISHEN DEEPAK
1BM23CS152
C:\student>

Program 3
BOOK DETAILS

14/10/2021

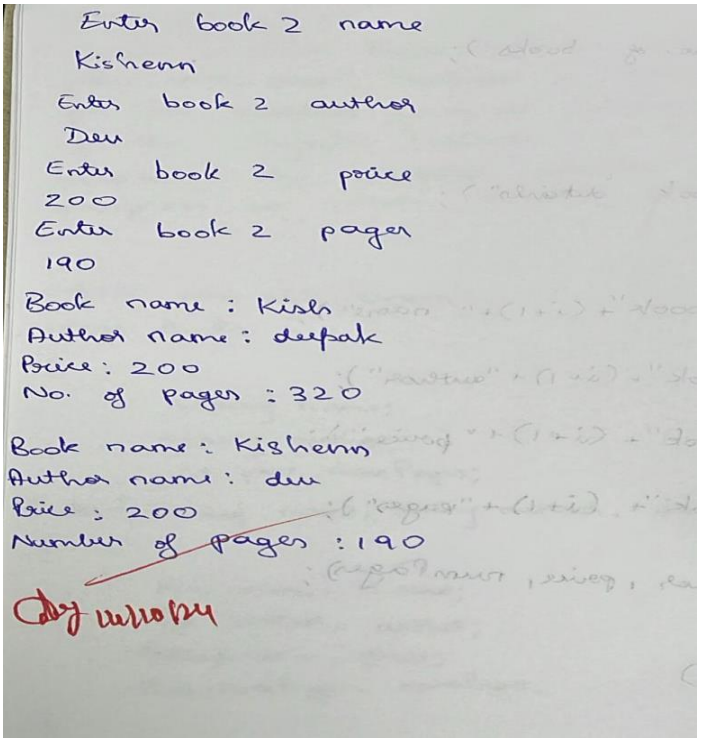
Lab Program 3

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 & 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.

```
import java.util.Scanner;
class Books
{
    String name;
    String author;
    int price, numPages;
Books(String name, String author, int price, int numPages)
{
    this.name= name;
    this.author= author;
    this.price= price;
    this.numPages= numPages;
}
public String toString()
{
String name, author, price, numPages;
name = "Book name:" + this.name + "\n";
author = "Author name:" + this.author + "\n";
price = "Price:" + this.price + "\n";
numPages = "Number of pages:" + this.numPages + "\n";
return name + author + price + numPages;
}
public static void main(String args[])
{
    Scanner sc = new Scanner(System.in);

    int n;
    String name;
    String author;
    int price;
    int numPages;
    System.out.println("Enter no. of books");
    n = sc.nextInt();
    Books b[];
    b= new Books[n];
    System.out.println("Enter book details");
    for(int i=0; i<n; i++)
{
    System.out.println("Enter book"+(i+1)+" name");
    name = sc.nextLine();
    System.out.println("Enter book"+(i+1)+"author");
    author= sc.nextLine();
    System.out.println("Enter book"+(i+1)+"price");
    price = sc.nextInt();
    System.out.println("Enter book"+(i+1)+"pages");
    numPages = sc.nextInt();
    b[i]= new Books(name, author, price, numPages);
}
    for(int i=0; i<n; i++)
{
    b[i] = b[i].toString();
        System.out.println(b[i]);
    }
}
}
```

Output:

```
Enter book details
Enter book 1 name
Kish
Enter book 1 author
deepak
Enter book 1 price
200
Enter book 1 pages
320
```

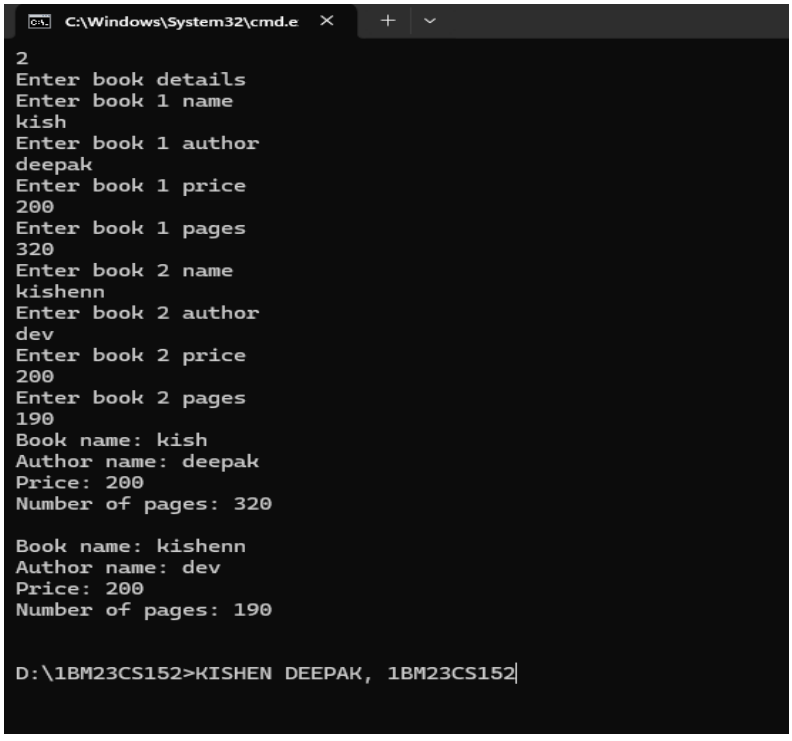


CODE: import java.util.Scanner;

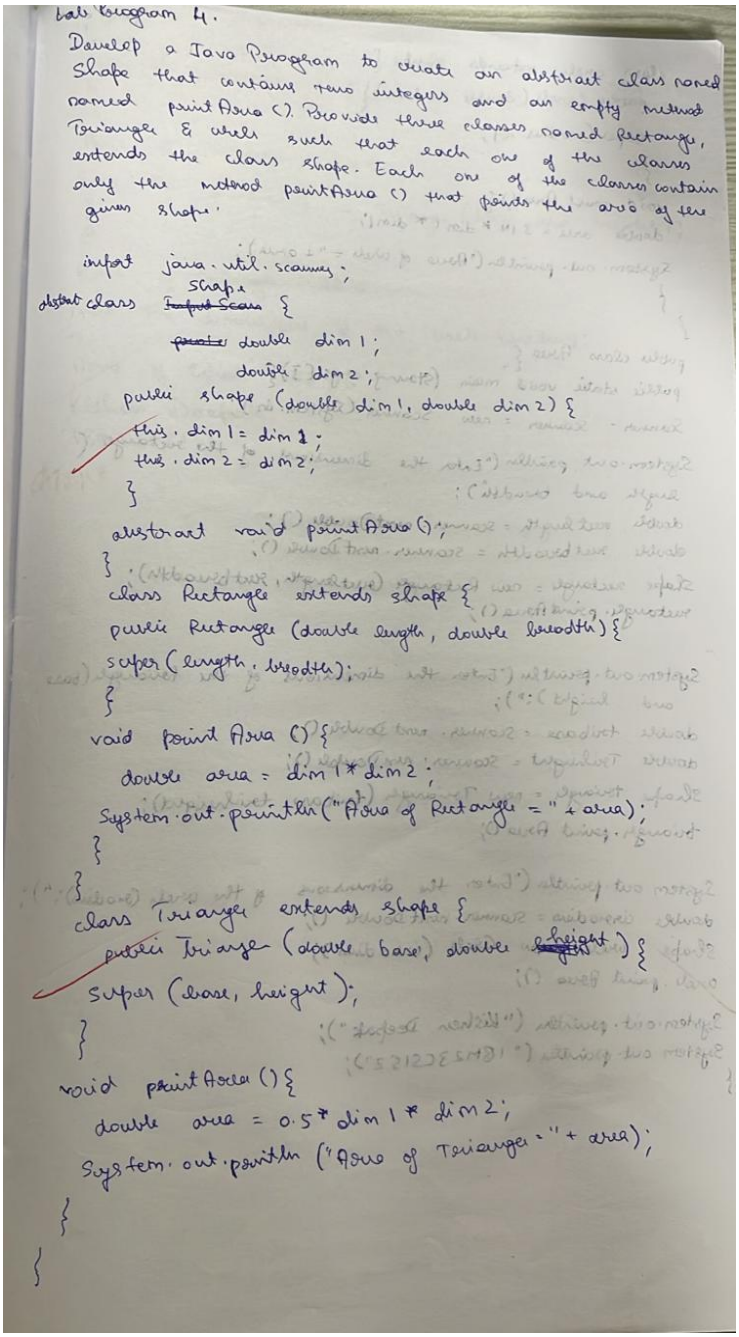
```
class Books
{
        String name;
        String author;
        int price,numPages;
Books(String name,String author,int price,int numPages)
{

        this.name = name;
        this.author = author;
        this.price = price;
        this.numPages = numPages;
}

public String toString()

{

String name, author, price, numPages;
name = "Book name: " + this.name + "\n";
author = "Author name: " + this.author + "\n";
price = "Price: " + this.price + "\n";
numPages = "Number of pages: " + this.numPages + "\n";
return name + author + price + numPages;
}



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

```
        int n;
        String name;
        String author;
        int price;
        int numPages;
        System.out.println("Enter no. of books");
        n=sc.nextInt();
        Books b[];
        b=new Books[n];
        System.out.println("Enter book details");
        for(int i=0;i<n;i++)
{
        System.out.println("Enter book "+(i+1)+" name");
        name=sc.next();
        System.out.println("Enter book "+(i+1)+" author");
        author=sc.next();
        System.out.println("Enter book "+(i+1)+" price");
        price=sc.nextInt();
        System.out.println("Enter book "+(i+1)+" pages");
        numPages=sc.nextInt();
        b[i]=new Books(name,author,price,numPages);
}
        for(int i=0;i<n;i++)
{
        b[i].toString();
        System.out.println(b[i]);

}
}
}
```

OUTPUT:



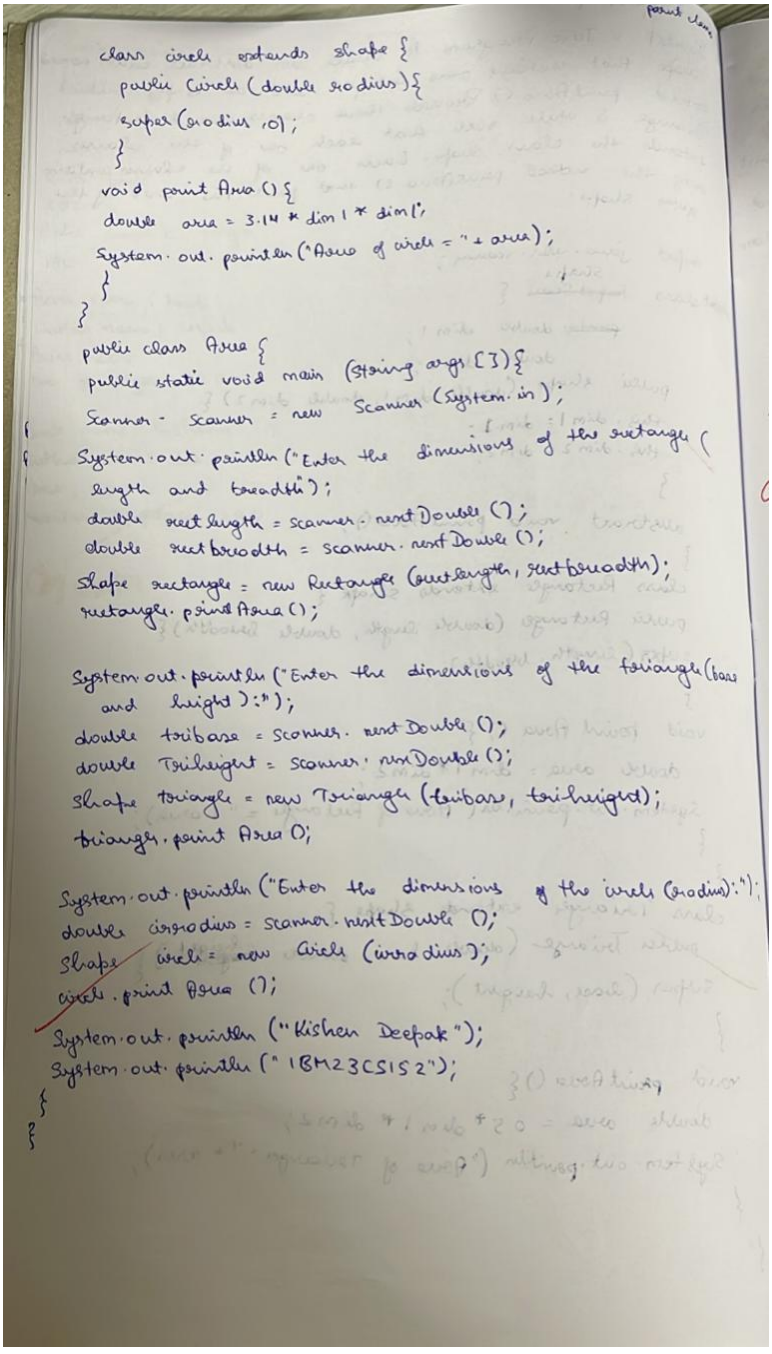
Program 4
AREA OF TRIANGLE,RECTANGLE,CIRCLE



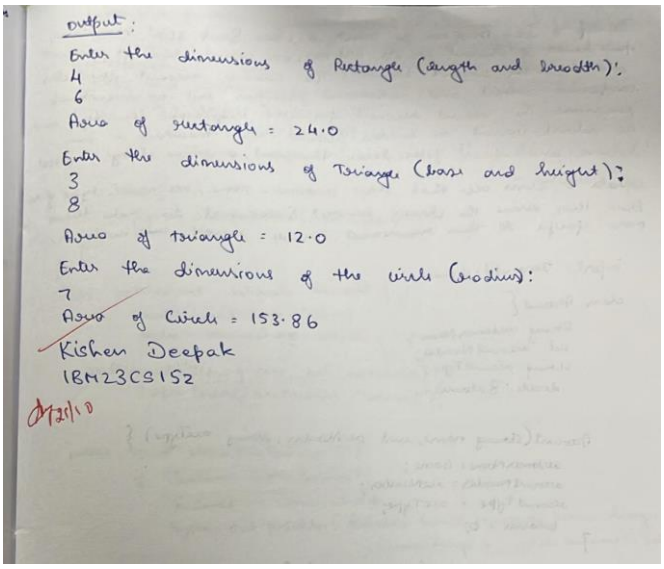
Lab Program 4.

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 & 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 {
    double dim1;
    double dim2;
    public Shape(double dim1, double dim2) {
        this.dim1 = dim1;
        this.dim2 = dim2;
    }
    abstract void printArea();
}
class Rectangle extends Shape {
    public Rectangle(double length, double breadth) {
        super(length, breadth);
    }
    void printArea() {
        double area = dim1 * dim2;
        System.out.println("Area of Rectangle = " + area);
    }
}
class Triangle extends Shape {
    public Triangle(double base, double height) {
        super(base, height);
    }
    void printArea() {
        double area = 0.5 * dim1 * dim2;
        System.out.println("Area of Triangle = " + area);
    }
}
```



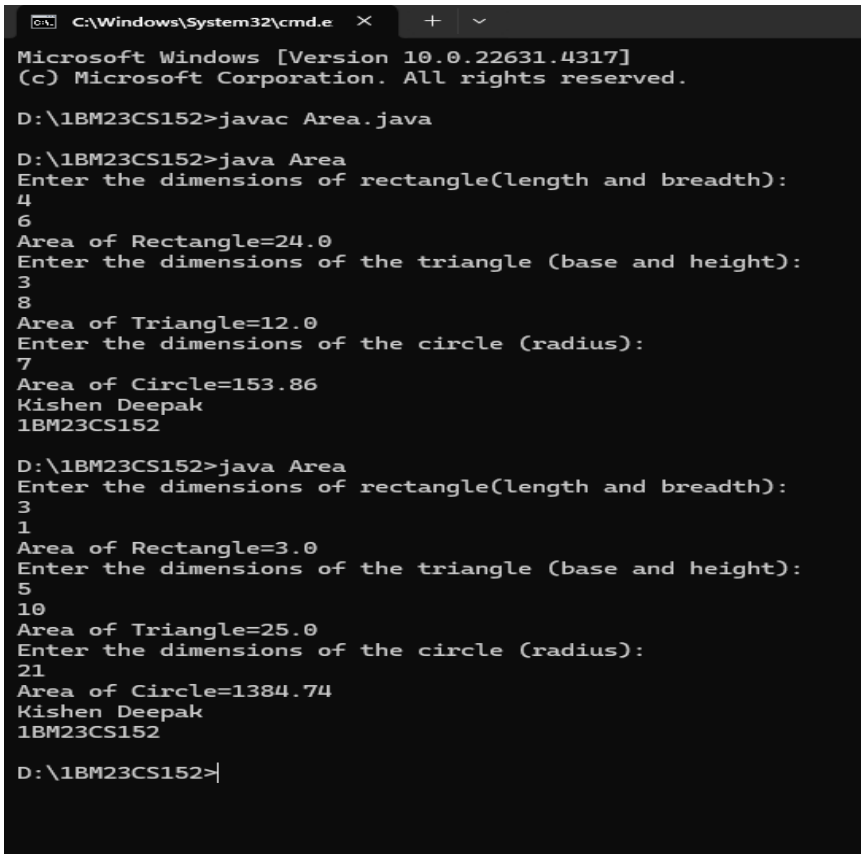
```
class Circle extends Shape {
    public Circle(double radius) {
        super(radius, 0);
    }
    void printArea() {
        double area = 3.14 * dim1 * dim1;
        System.out.println("Area of circle = " + area);
    }
}
public class Area {
    public static void main(String args[]) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter the dimensions of the rectangle (
            length and breadth)");
        double rectLength = scanner.nextDouble();
        double rectBreadth = scanner.nextDouble();
        Shape rectangle = new Rectangle(rectLength, rectBreadth);
        rectangle.printArea();

        System.out.println("Enter the dimensions of the triangle(base
            and height):");
        double triBase = scanner.nextDouble();
        double triHeight = scanner.nextDouble();
        Shape triangle = new Triangle(triBase, triHeight);
        triangle.printArea();

        System.out.println("Enter the dimensions of the circle(radius):");
        double circleRadius = scanner.nextDouble();
        Shape circle = new Circle(circleRadius);
        circle.printArea();
        System.out.println("Kishen Deepak");
        System.out.println("1BM23CS152");
    }
}
```



CODE:
```
import java.util.Scanner;
abstract class shape{
double dim1;
double dim2;
public shape(double dim1,double dim2){
this.dim1=dim1;
this.dim2=dim2;
}
abstract void printArea();
}
class Rectangle extends shape{
public Rectangle(double length, double breadth){
super(length ,breadth);
}
void printArea(){
double area=dim1*dim2;
System.out.println("Area of Rectangle="+area);
}
}
class Triangle extends shape{
public Triangle(double base,double height){
super(base,height);
}
void printArea(){
double area=0.5*dim1*dim2;
System.out.println("Area of Triangle="+area);
}
}
class Circle extends shape{
public Circle(double radius){
super(radius,0);
}
void printArea(){
double area=3.14*dim1*dim1;
```

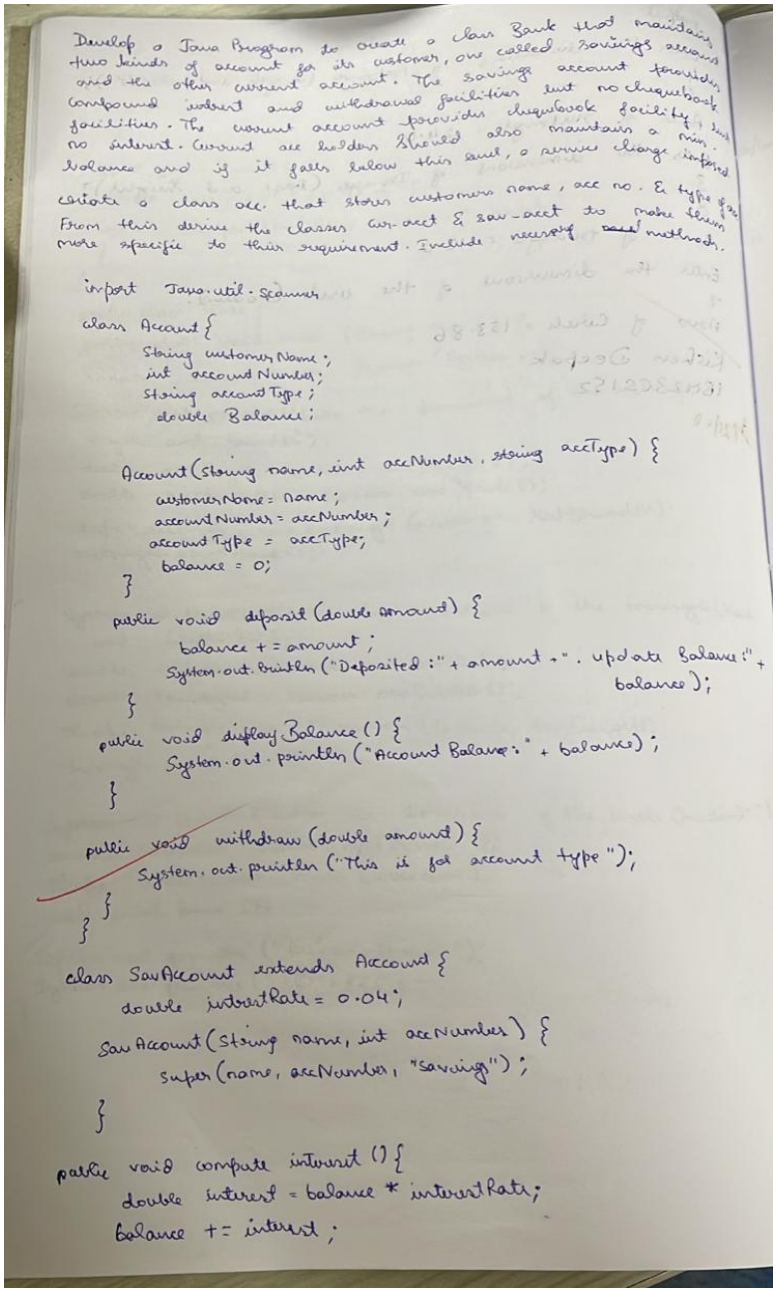
```
System.out.println("Area of Circle="+area);
}
}
public class Area{
public static void main(String args[]){
Scanner scanner=new Scanner(System.in);
System.out.println("Enter the dimensions of rectangle(length and breadth):");
double rectlength=scanner.nextDouble();
double rectbreadth=scanner.nextDouble();
shape rectangle=new Rectangle(rectlength,rectbreadth);
rectangle.printArea();

System.out.println("Enter the dimensions of the triangle (base and height):");
double tribase=scanner.nextDouble();
double triheight=scanner.nextDouble();
shape triangle=new Triangle(tribase,triheight);
triangle.printArea();

System.out.println("Enter the dimensions of the circle (radius):");
double cirradius=scanner.nextDouble();
shape circle=new Circle(cirradius);
circle.printArea();
System.out.println("Kishen Deepak");
System.out.println("1BM23CS152");
}
}
```

OUTPUT:



Program 5

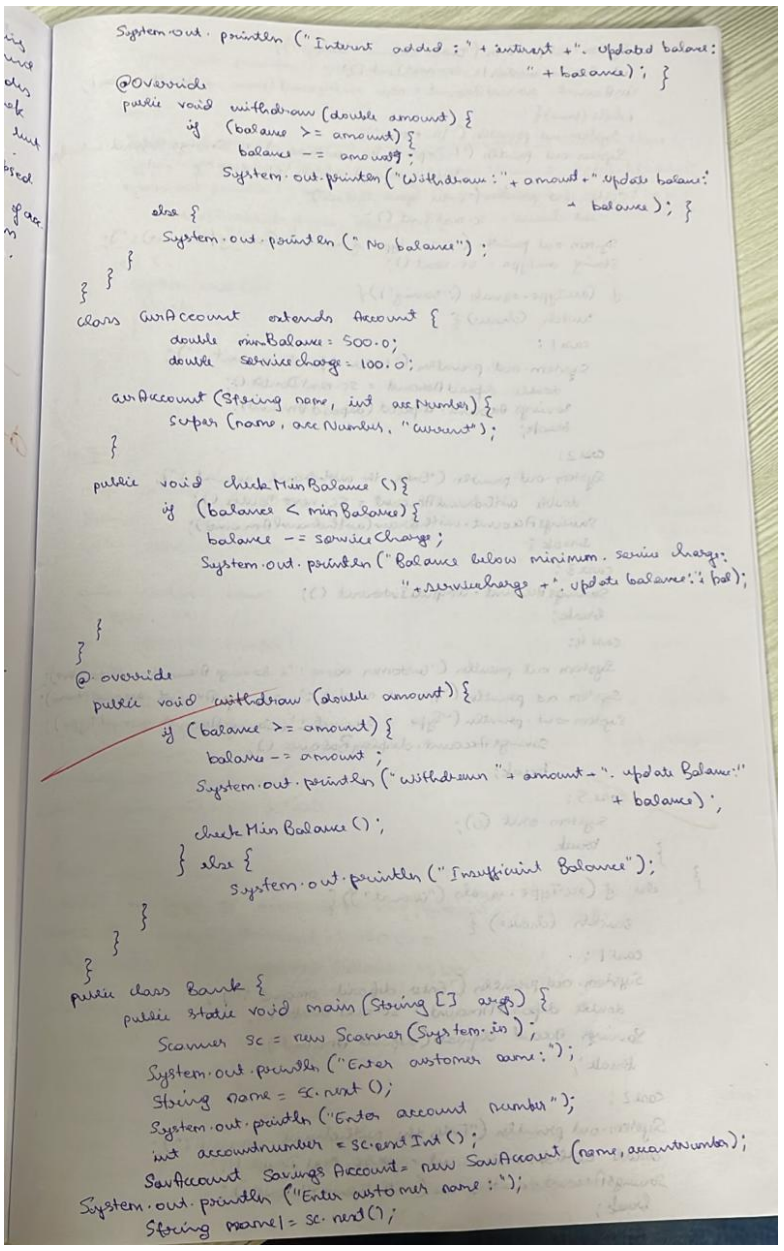
BANK ACCOUNT



Develop a Java Program to create a class Bank that maintains two kinds of account for its customer, one called savings account and the other current account. The savings account provides compound interest and withdrawal facilities but no chequebook facilities. The current account provides chequebook facility but no interest. Current account holders should also maintain a min. balance and if it falls below this level, a service charge is imposed.

Create a class acc. that stores customer name, acc no. & type of acc. From this derive the classes cur-acct & sav-acct to make them more specific to their requirement. Include necessary methods.

```
import java.util.Scanner
class Account {
    String customerName;
    int accountNumber;
    String accountType;
    double balance;

    Account(String name, int accNumber, String accType) {
        customerName = name;
        accountNumber = accNumber;
        accountType = accType;
        balance = 0;
    }
    public void deposit(double amount) {
        balance += amount;
        System.out.println("Deposited :" + amount + ". updated Balance:" +
                                                    balance);
    }
    public void displayBalance() {
        System.out.println("Account Balance:" + balance);
    }

    public void withdraw(double amount) {
        System.out.println("This is for account type");
    }
}

class SavAccount extends Account {
    double interestRate = 0.04;
    SavAccount(String name, int accNumber) {
        super(name, accNumber, "savings");
    }

public void computeInterest() {
    double interest = balance * interestRate;
    balance += interest;
```



```
    System.out.println("Interest added :" + interest + ". Updated balance:
                                                    " + balance); }
    @Override
    public void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
            System.out.println("Withdrawn :" + amount + " Updated balance:
                                                    " + balance); }
        else {
            System.out.println("No balance");
        }
    }
}
class CurAccount extends Account {
        double minBalance = 500.0;
        double serviceCharge = 100.0;
    CurAccount(String name, int accNumber) {
        super(name, accNumber, "current");
    }
    public void checkMinBalance() {
        if (balance < minBalance) {
            balance -= serviceCharge;
            System.out.println("Balance below minimum. service charge:
                                    "+serviceCharge + ". updated balance:" bal);

    }
}
@Override
public void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
            System.out.println("withdrawn " + amount + ". updated Balance:"
                                                    + balance);
            checkMinBalance();
        } else {
            System.out.println("Insufficient 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.next();
        System.out.println("Enter account number");
        int accountNumber = sc.nextInt();
        SavAccount savingsAccount = new SavAccount(name, accountNumber);
System.out.println("Enter customer name:");
        String name1 = sc.next();
```

```
System.out.println("Enter account number");
int AccountNumber1 = sc.nextInt();
CurAccount currentAccount = new CurAccount(name, accountNumber);
while(true){
  System.out.println("\n menu");
  System.out.println("1-Deposit \n2. withdraw \n3. Savings account interest \n4.
                      Display account details \n5. Exit");
  System.out.println("Enter your choice");
    int choice = sc.nextInt();
  System.out.println("Enter type of account (saving/current):");
  String accType = sc.next();
  if (accType.equals("saving")){
    switch (choice) {
      case 1:
      System.out.println("Enter deposit amount :");
        double depositAmount = sc.nextDouble();
        savingsAccount.deposit(depositAmount);
        break;
      case 2:
      System.out.println("Enter the withdrawal amount :");
        double withdrawalAmount = sc.nextDouble();
        SavingsAccount.withdraw(withdrawalAmount);
        break;
      case 3:
      SavingsAccount.computeInterest();
      break;
      case 4:
      System.out.println("customer name :"+ savingsAccount.customerName);
      System.out.println("Account number :"+ savingsAccount.accountNumber);
      System.out.println("Type of account :"+ savingsAccount.accountType);
            SavingsAccount.displayBalance();
            break;
      case 5:
        System.exit(0);
        break
    }
  } else if (accType.equals("current")){
    switch (choice) {
    case 1:
    System.out.println("Enter deposit amount :");
      double depositAmount = sc.nextDouble();
    SavingsAccount.deposit(depositAmount);
    break;
    case 2:
    System.out.println("Enter the withdrawal amount :");
    double withdrawalAmount = sc.nextDouble();
    SavingsAccount.withdraw(withdrawalAmount);
    break;
```

```
    case 3:
      System.out.println("current account no interest");
        break;
    case 4:
      System.out.println("customer name :"+SavingsAccount.customerName);
      System.out.println("Account number :"+ currentAccount.accountNumber);
      System.out.println("Type of account :"+ currentAccount.accountType);
          currentAccount.displayBalance();
          break;
    case 5:
      System.exit(0);
      break;
      }
    }
  }
}
```

output:

```
Enter customer name:
Kishen
Enter account number:
7154
Enter customer name:
Deepak
Enter account number:
1740
---Menue---
1. Deposit
2. Withdraw
3. compute Interest rate
4. Display account details
5. Exit
Enter your choice: 1
Enter the type of account (saving/current): saving
Enter the deposit amount : 1500.0
Deposited : 1500   updated balance: 1500

---Menue---
1. Deposit
2. Withdraw
3. Compute Interest Rate
4. Display account details
5. Exit
Enter your choice : 3
Enter the type of account (saving/current): saving
Interest added : 60.0   updated Balance : 1560.0
```

```
Customer name: Kishen
Account number: 7154
Type of account : Savings
Amount Balance : 1560.0

--- Menue ---
1. Deposit
2. Withdraw
3. Compute Interest for savings account
4. Display account details
5. Exit
Enter your choice : 3
Enter the type of account (saving/current): saving
Interest added : 62.4. Updated Balance : 1622.4
```

28/10/24

CODE: import java.util.Scanner;

```
class Account {
    String customerName;
    int accountNumber;
    String accountType;
    double balance;


    Account(String name, int accNumber, String accType) {
        customerName = name;
        accountNumber = accNumber;
        accountType = accType;
        balance = 0;
    }


    public void deposit(double amount) {
        balance += amount;
        System.out.println("Deposited: " + amount + ". Updated balance: " + balance);
    }

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


    public void withdraw(double amount) {
        System.out.println("This operation is specific to account type.");
    }
}


class SavAccount extends Account {
    double interestRate = 0.04;  // 4% annual interest rate

    SavAccount(String name, int accNumber) {
        super(name, accNumber, "Savings");
    }


    public void computeInterest() {
        double interest = balance * interestRate;
        balance += interest;
        System.out.println("Interest added: " + interest + ". Updated balance: " + balance);
    }


    @Override
    public void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
            System.out.println("Withdrawn: " + amount + ". Updated balance: " + balance);
        } else {
```

```
            System.out.println("Insufficient balance.");
        }
    }
}
class CurAccount extends Account {
    double minBalance = 500.0;
    double serviceCharge = 50.0;

    CurAccount(String name, int accNumber) {
        super(name, accNumber, "Current");
    }

    public void checkMinBalance() {
        if (balance < minBalance) {
            balance -= serviceCharge;
            System.out.println("Balance below minimum. Service charge imposed: " + serviceCharge + ". Updated balance: " + balance);
        }
    }


    @Override
    public void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
            System.out.println("Withdrawn: " + amount + ". Updated balance: " + balance);
            checkMinBalance();
        } else {
            System.out.println("Insufficient 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.next();
        System.out.println("Enter account number:");
        int accountnumber=sc.nextInt();
        SavAccount savingsAccount = new SavAccount(name, accountnumber);
System.out.println("Enter customer name:");
        String name1=sc.next();
        System.out.println("Enter account number:");
        int accountnumber1=sc.nextInt();
        CurAccount currentAccount = new CurAccount(name1, accountnumber1);

        while (true) {
            System.out.println("\n-----MENU-----");
            System.out.println("1. Deposit\n2. Withdraw\n3. Compute Interest for Savings Account\n4. Display Account Details\n5. Exit");
            System.out.print("Enter your choice: ");
            int choice = sc.nextInt();

            System.out.print("Enter the type of account (saving/current): ");
            String accType = sc.next();
```

```
if (accType.equals("saving")) {
    switch (choice) {
        case 1:
            System.out.print("Enter the deposit amount: ");
            double depositAmount = sc.nextDouble();
            savingsAccount.deposit(depositAmount);
            break;
        case 2:
            System.out.print("Enter the withdrawal amount: ");
            double withdrawalAmount = sc.nextDouble();
            savingsAccount.withdraw(withdrawalAmount);
            break;
        case 3:
            savingsAccount.computeInterest();
            break;
        case 4:
            System.out.println("Customer name: " + savingsAccount.customerName);
            System.out.println("Account number: " + savingsAccount.accountNumber);
            System.out.println("Type of Account: " + savingsAccount.accountType);
            savingsAccount.displayBalance();
            break;
        case 5:
            System.exit(0);
            break;
        default:
            System.out.println("Invalid choice.");
    }
} else if (accType.equals("current")) {
    switch (choice) {
        case 1:
            System.out.print("Enter the deposit amount: ");
            double depositAmount = sc.nextDouble();
            currentAccount.deposit(depositAmount);
            break;
        case 2:
            System.out.print("Enter the withdrawal amount: ");
            double withdrawalAmount = sc.nextDouble();
            currentAccount.withdraw(withdrawalAmount);
            break;
        case 3:
            System.out.println("Current accounts do not earn interest.");
            break;
        case 4:
            System.out.println("Customer name: " + currentAccount.customerName);
            System.out.println("Account number: " + currentAccount.accountNumber);
            System.out.println("Type of Account: " + currentAccount.accountType);
            currentAccount.displayBalance();
            break;
        case 5:
            System.exit(0);
            break;
        default:
            System.out.println("Invalid choice.");
    }
} else {
```

```
                System.out.println("Invalid account type.");
            }
        }
    }
}
```

OUTPUT

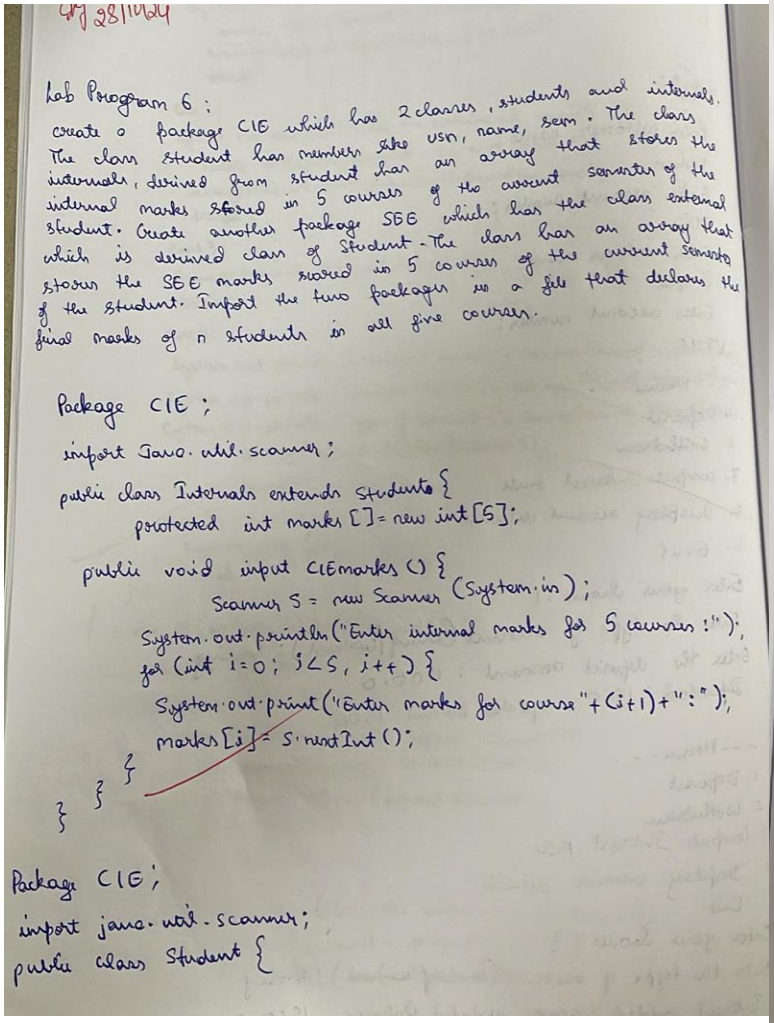
```
Microsoft Windows [Version 10.0.22631.3447]
(c) Microsoft Corporation. All rights reserved.

D:\1BM23CS152>javac Bank.java

D:\1BM23CS152>java Bank
Enter customer name:
kishen
Enter account number:
7154
Enter customer name:
deepak
Enter account number:
1234

-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 1
Enter the type of account (saving/current): saving
Enter the deposit amount: 1500
Deposited: 1500.0. Updated balance: 1500.0

-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 3
Enter the type of account (saving/current): saving
Interest added: 60.0. Updated balance: 1560.0

-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 4
Enter the type of account (saving/current): saving
Customer name: kishen
Account number: 7154
Type of Account: Savings
Account Balance: 1560.0
```

```
-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 3
Enter the type of account (saving/current): saving
Interest added: 62.4. Updated balance: 1622.4

-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice:
```

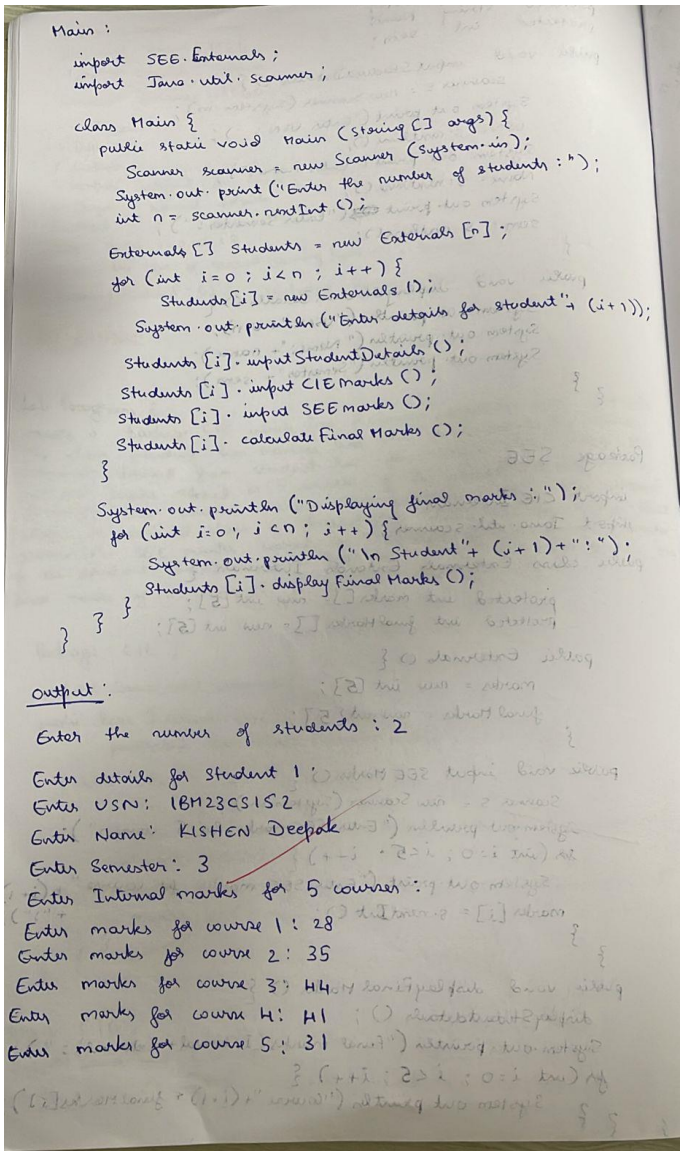
```
-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 2
Enter the type of account (saving/current): current
Enter the withdrawal amount: 10
Insufficient balance.

-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice:
```

```
-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 1
Enter the type of account (saving/current): CURRENT
Invalid account type.

-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice: 1
Enter the type of account (saving/current): current
Enter the deposit amount: 2000
Deposited: 2000.0. Updated balance: 2000.0

-----MENU-----
1. Deposit
2. Withdraw
3. Compute Interest for Savings Account
4. Display Account Details
5. Exit
Enter your choice:
```

Program 6
PACKAGE



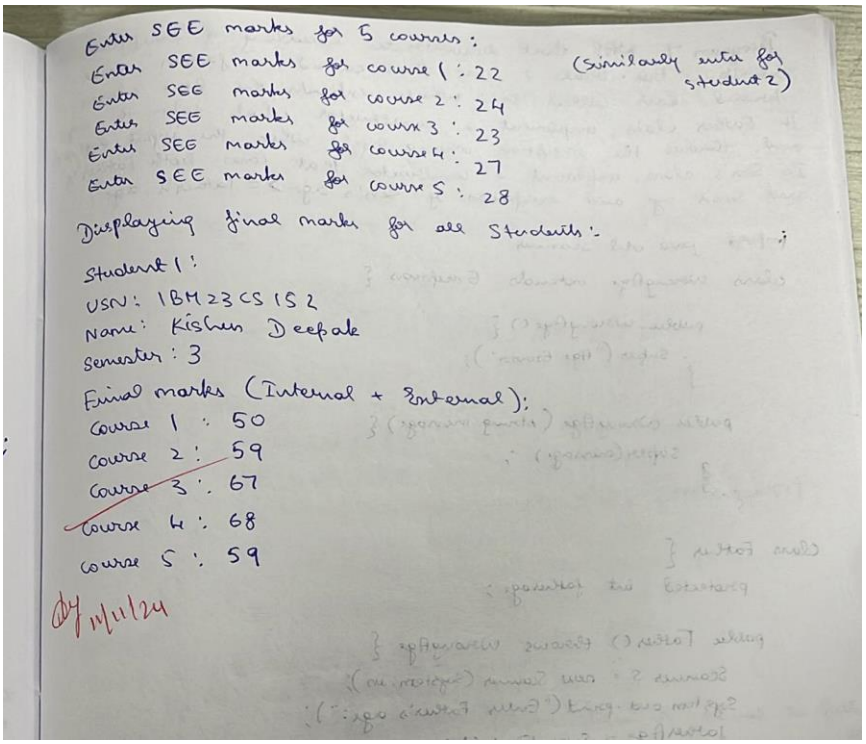
```
Lab Program 6 :
create a package CIE which has 2 classes, students and internals.
The class student has members like usn, name, sem. The class
internals, derived from student has an array that stores the
internal marks stored in 5 courses of the current semester of the
student. Create another package SEE which has the class external
which is derived class of Student. The class has an array that
stores the SEE marks scored in 5 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 Internals extends students {
    protected int marks[] = new int[5];
    public void inputCIEmarks () {
        Scanner S = 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] = S.nextInt();
        }
    }
}

Package CIE;
import java.util.scanner;
public class Student {
    protected string USN;
    protected string Name;
    protected int sem;
    public void inputStudentDetails () {
        scanner s = new Scanner (System.in);
        System.out.print("Enter USN : ");
        USN = S.nextline();
        System.out.print("Enter name:");
        Name = S.nextline();
        System.out.print("Enter Semester:");
        sem = S.nextline();
    }
    public void displayStudentDetails () {
        System.out.println("USN:"+USN);
        System.out.println("Name:"+name);
        System.out.println("Semester"+sem);
    }
}

Package SEE
import CIE.Internals;
import Java.util.scanner;
public class Externals Extends Internals {
    protected int marks[] = new int[5];
    protected int finalMarks[] = new int[5];
    public Externals () {
        marks = new int[5];
        finalMarks = new int[5];
    }
    public void inputSEEMarks () {
        Scanner S = new Scanner (System.in);
        System.out.println("Enter SEE marks for 5 courses");
        for (int i=0; i<5; i++) {
            System.out.print("Enter SEE marks for course "+(i+1)
                +":");
            marks[i] = s.nextInt();
        }
    }
    public void displayFinalMarks () {
        displayStudentdetails ();
        System.out.println("Final marks (Internal+External):");
        for (int i=0; i<5; i++) {
            System.out.println("Course "+(i+1)+finalMarks[i]);
        }
    }
}
```



```
Main :
import SEE.Externals;
import Java.util.scanner;

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].input CIE marks ();
      Students [i].input SEE marks ();
      Students [i].calculate Final Marks ();
    }

    System.out.println ("Displaying final marks :");
    for (int i=0; i<n; i++) {
      System.out.println ("\n Student"+ (i+1)+":");
      Students [i].display Final Marks ();
    }
  }
}
```

Output :

```
Enter the number of students : 2

Enter details for Student 1 :
Enter USN: 1BM23CS152
Enter Name: KISHEN Deepak
Enter Semester: 3
Enter Internal marks for 5 courses :
Enter marks for course 1 : 28
Enter marks for course 2: 35
Enter marks for course 3: 44
Enter marks for course 4: 41
Enter marks for course 5: 31
```



```
Enter SEE marks for 5 courses :
Enter SEE marks for course 1 : 22        (Similarly enter for student 2)
Enter SEE marks for course 2 : 24
Enter SEE marks for course 3 : 23
Enter SEE marker for course 4 : 27
Enter SEE marker for course 5 : 28

Displaying final marks for all Students :-

Student 1 :
USN : 1BM23CS152
Name : Kishen Deepak
Semester : 3
Final marks (Internal + Internal) :
Course 1 : 50
Course 2 : 59
Course 3 : 67
Course 4 : 68
Course 5 : 59
```

CODE:

```
package CIE;

import java.util.Scanner;

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

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

package SEE;

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

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

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

    public void inputSEEmarks() {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter SEE Marks for 5 courses:");
        for (int i = 0; i < 5; i++) {
            System.out.print("Enter SEE marks for course " + (i + 1) + ": ");
            marks[i] = s.nextInt();
        }
    }

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

    public void displayFinalMarks() {
        displayStudentDetails();
        System.out.println("Final Marks (Internal + External):");
```

```
        for (int i = 0; i < 5; i++) {
            System.out.println("Course " + (i + 1) + ": " + finalMarks[i]);
        }
    }
}

import SEE.Externals;
import java.util.Scanner;

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("\nEnter details for student " + (i + 1));


            students[i].inputStudentDetails();

            students[i].inputCIEmarks();

            students[i].inputSEEmarks();

            students[i].calculateFinalMarks();
        }

        System.out.println("\nDisplaying final marks for all students:");
        for (int i = 0; i < n; i++) {
            System.out.println("\nStudent " + (i + 1) + ":");
            students[i].displayFinalMarks();
        }
    }
}
```

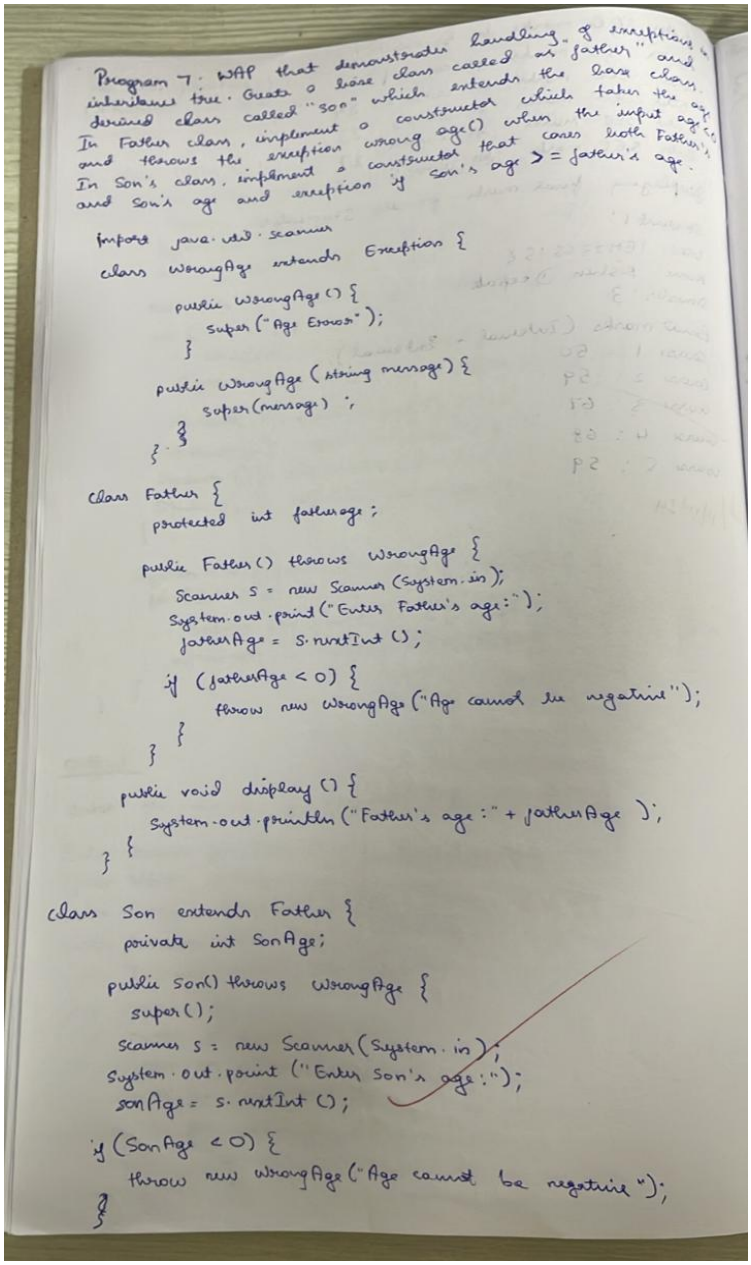
OUTPUT:

OUTPUT:

```
D:\1BM23CS152>java Main
Enter the number of students: 2

Enter details for student 1
Enter USN: 1BM23CS152
Enter Name: KISHEN DEEPAK
Enter Semester: 3
Enter Internal Marks for 5 courses:
Enter marks for course 1: 28
Enter marks for course 2: 35
Enter marks for course 3: 44
Enter marks for course 4: 41
Enter marks for course 5: 31
Enter SEE Marks for 5 courses:
Enter SEE marks for course 1: 36
Enter SEE marks for course 2: 28
Enter SEE marks for course 3: 41
Enter SEE marks for course 4: 37
Enter SEE marks for course 5: 33

Enter details for student 2
Enter USN: 1bm23cs152
Enter Name: Kishen Deepak
Enter Semester: 3
Enter Internal Marks for 5 courses:
Enter marks for course 1: 22
Enter marks for course 2: 24
Enter marks for course 3: 23
Enter marks for course 4: 27
Enter marks for course 5: 28
Enter SEE Marks for 5 courses:
Enter SEE marks for course 1: 27
Enter SEE marks for course 2: 32
Enter SEE marks for course 3: 34
Enter SEE marks for course 4: 35
Enter SEE marks for course 5: 37

Displaying final marks for all students:

Student 1:
USN: 1BM23CS152
Name: KISHEN DEEPAK
Semester: 3
Final Marks (Internal + External):
Course 1: 72
Course 2: 56
Course 3: 82
Course 4: 74
Course 5: 66
```

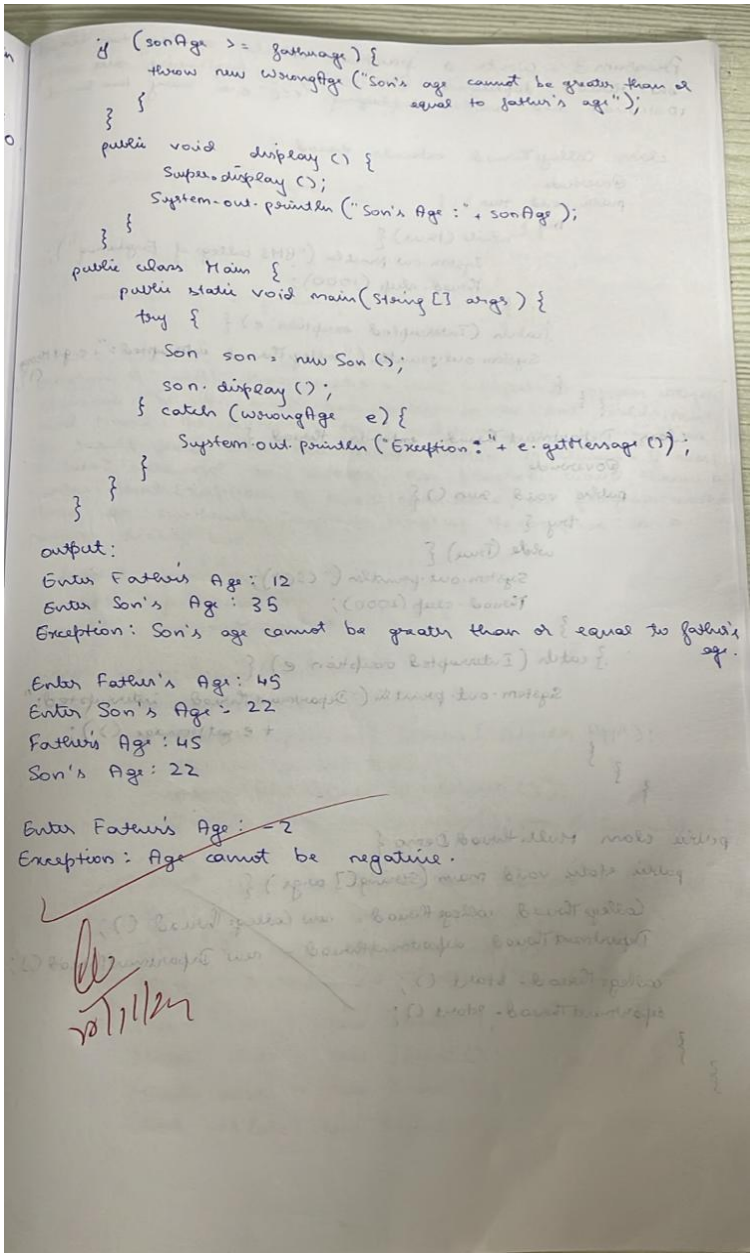
```
Student 2:
USN: 1bm23cs152
Name: Kishen Deepak
Semester: 3
Final Marks (Internal + External):
Course 1: 54
Course 2: 64
Course 3: 68
Course 4: 70
Course 5: 74

D:\1BM23CS152>
```

Program 7
FATHER-SON AGE



Program 7: WAP that demonstrates handling of exceptions in inheritance tree. Create a base class called as 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 is < 0. In Son's class, implement a constructor that takes both Father's and Son's age and exception if son's age >= father's age.

```
import java.util.Scanner
class WrongAge extends Exception {
    public WrongAge() {
        super("Age Error");
    }
    public WrongAge(String message) {
        super(message);
    }
}
class Father {
    protected int fatherAge;
    public Father() throws WrongAge {
        Scanner s = new Scanner(System.in);
        System.out.print("Enter Father's age:");
        fatherAge = s.nextInt();
        if (fatherAge < 0) {
            throw new WrongAge("Age cannot be negative");
        }
    }
    public void display() {
        System.out.println("Father's age:" + fatherAge);
    }
}
class Son extends Father {
    private int SonAge;
    public Son() throws WrongAge {
        super();
        Scanner s = new Scanner(System.in);
        System.out.print("Enter Son's age:");
        sonAge = s.nextInt();
        if (SonAge < 0) {
            throw new WrongAge("Age cannot be negative");
        }
```



```
        if (sonAge >= fatherAge) {
            throw new WrongAge("Son's age cannot be greater than or
                                equal to father's age");
        }
    }
    public void display() {
        super.display();
        System.out.println("Son's Age:" + sonAge);
    }
}
public class Main {
    public static void main(String[] args) {
        try {
            Son son = new Son();
            son.display();
        } catch (WrongAge e) {
            System.out.println("Exception:" + e.getMessage());
        }
    }
}
```

output:

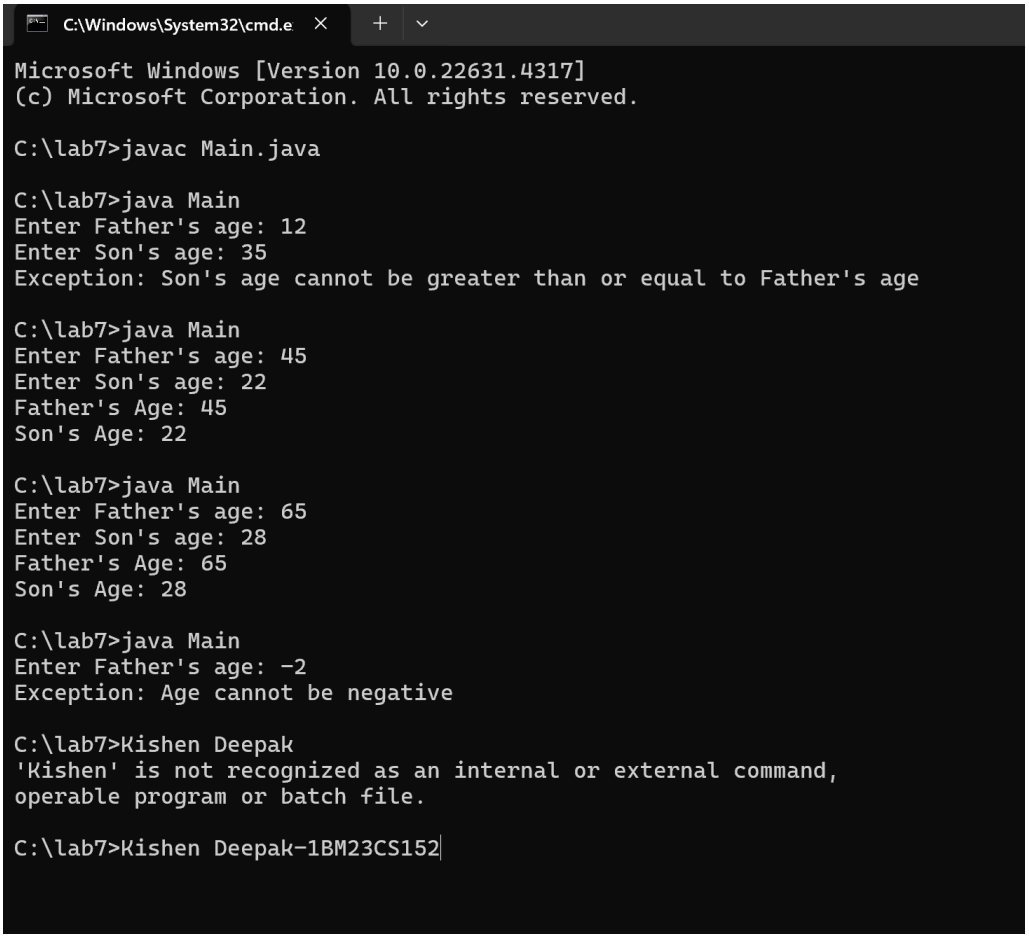
```
Enter Father's Age: 12
Enter Son's Age: 35
Exception: Son's age cannot be greater than or equal to father's age.

Enter Father's Age: 45
Enter Son's Age: 22
Father's Age: 45
Son's Age: 22

Enter Father's Age: -2
Exception: Age cannot be negative.
```

CODE:

```
import java.util.Scanner;

class WrongAge extends Exception {

    public WrongAge() {
        super("Age Error");
    }


    public WrongAge(String message) {
        super(message);
    }
}


class Father {
    protected int fatherAge;


    public Father() throws WrongAge {
        Scanner s = new Scanner(System.in);
        System.out.print("Enter Father's age: ");
        fatherAge = s.nextInt();

        if (fatherAge < 0) {
            throw new WrongAge("Age cannot be negative");
        }
    }

    public void display() {
        System.out.println("Father's Age: " + fatherAge);
    }
}

class Son extends Father {
    private int sonAge;

    public Son() throws WrongAge {
        super();
        Scanner s = new Scanner(System.in);
        System.out.print("Enter Son's age: ");
        sonAge = s.nextInt();

        if (sonAge < 0) {
            throw new WrongAge("Age cannot be negative");
        }
        if (sonAge >= fatherAge) {
            throw new WrongAge("Son's age cannot be greater than or equal to Father's age");
        }
    }

    public void display() {
        super.display();
        System.out.println("Son's Age: " + sonAge);
```

```
    }
}

public class Main {
    public static void main(String[] args) {
        try {

            Son son = new Son();

            son.display();
        } catch (WrongAge e) {

            System.out.println("Exception: " + e.getMessage());
        }
    }
}
```

OUTPUT:



Program 8
MULTITHREADING

Program 8 : Write a program which creates two threads, one thread displaying "BMS college of Engineering" once every 10 seconds and another displaying "CSE" once every two seconds.

```
class CollegeThread extends Thread {
    @Override
    public void run() {
        try {
            while (true) {
                System.out.println("BMS college of Engineering");
                Thread.sleep(1000);
            }
        } catch (InterruptedException e) {
            System.out.println("CollegeThread interrupted: " + e.getMessage());
        }
    }
}

class DepartmentThread extends Thread {
    @Override
    public void run() {
        try {
            while (true) {
                System.out.println("CSE");
                Thread.sleep(2000);
            }
        } catch (InterruptedException e) {
            System.out.println("DepartmentThread interrupted: "
                    + e.getMessage());
        }
    }
}

public class MultiThreadDemo {
    public static void main(String[] args) {
        CollegeThread collegeThread = new CollegeThread();
        DepartmentThread departmentThread = new DepartmentThread();
        collegeThread.start();
        departmentThread.start();
    }
}
```

output:

```
BMS college of Engineering
CSE
CSE
CSE
CSE
CSE
BMS college of Engineering
CSE
CSE
CSE
CSE
CSE
BMS college of Engineering
:
:
```

CODE:

```
class CollegeThread extends Thread {
    @Override
    public void run() {
        try {
            while (true) {
                System.out.println("BMS College of Engineering");
                Thread.sleep(10000);
            }
        } catch (InterruptedException e) {
            System.out.println("CollegeThread interrupted: " + e.getMessage());
        }
    }
}

class DepartmentThread extends Thread {
    @Override
    public void run() {
        try {
            while (true) {
                System.out.println("CSE");
                Thread.sleep(2000);
            }
        } catch (InterruptedException e) {
            System.out.println("DepartmentThread interrupted: " + e.getMessage());
        }
    }
}


public class MultiThreadDemo {
    public static void main(String[] args) {
        CollegeThread collegeThread = new CollegeThread();
        DepartmentThread departmentThread = new DepartmentThread();

        collegeThread.start();
        departmentThread.start();
    }
}
```

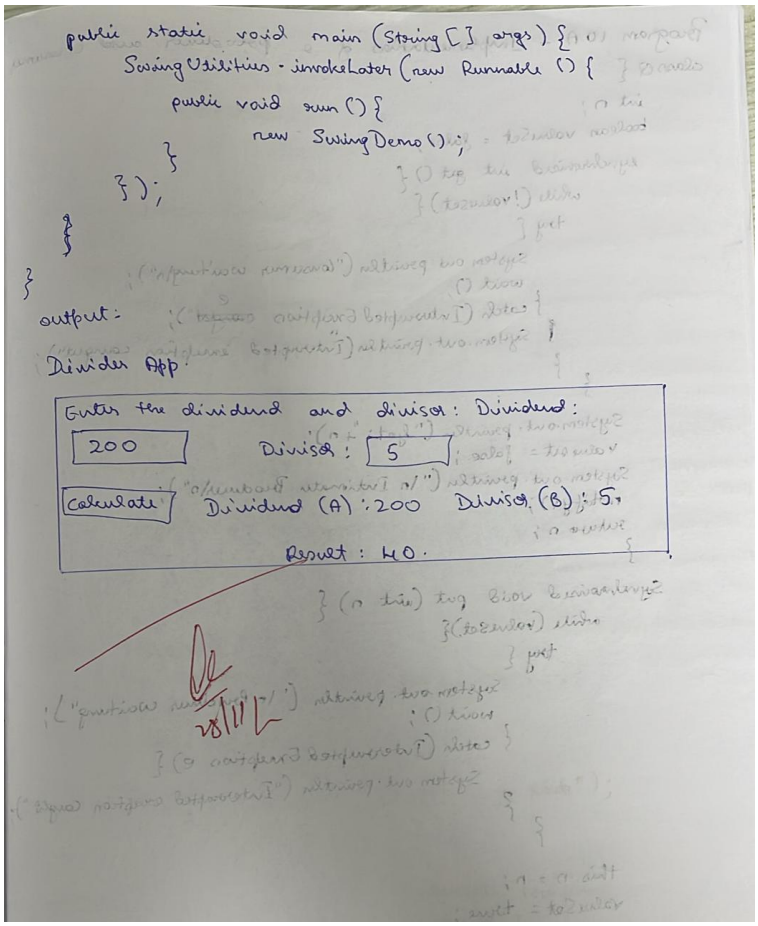
OUTPUT:

```
C:\lab8>java MultiThreadDemo
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
```

Program 9
DIVISION

Program 9 : WAP 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 ArithmeticException Display the exception in a message dialog box.

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

class SwingDemo {
    SwingDemo () {
        JFrame jfrm = new JFrame("Divider App");
        jfrm.setSize(300,200);
        jfrm.setLayout(new FlowLayout());
        jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JLabel jlab = new JLabel("Enter dividend and divisor:");
        JTextField ajtf = new JTextField(8);
        JTextField bjtf = new JTextField(8);
        JButton button = new JButton("Calculate");
        JLabel err = new JLabel();
        JLabel alab = new JLabel();
        JLabel blab = new JLabel();
        JLabel anslab = new JLabel();

        jfrm.add(jlab);
        jfrm.add(new JLabel("Dividend:"));
        jfrm.add(ajtf);
        jfrm.add(new JLabel("Divisor:"));
        jfrm.add(bjtf);
        jfrm.add(button);
        jfrm.add(alab);
        jfrm.add(blab);
        jfrm.add(anslab);
        jfrm.add(err);

        button.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent ent) {
                try {
                    int a = Integer.parseInt(ajtf.getText());
                    int b = Integer.parseInt(bjtf.getText());
                    int ans = a/b;
                    alab.setText("Dividend (A):"+a);
                    blab.SetText("Divisor (B):"+b);
                    anslab.SetText("Result:"+ans);
                    err.setText("");
                } catch (NumberFormatException e) {
                    alab.setText("");
                    blab.setText("");
                    anslab.setText("");
                    err.setText("Error: Enter valid integers!");
                } catch (ArithmeticException e) {
                    alab.setText("");
                    blab.setText("");
                    anslab.SetText("");
                    err.setText("Error: Divisor cannot be zero");
                }
            }
        });

        jfrm.setVisible(true);
    }
```

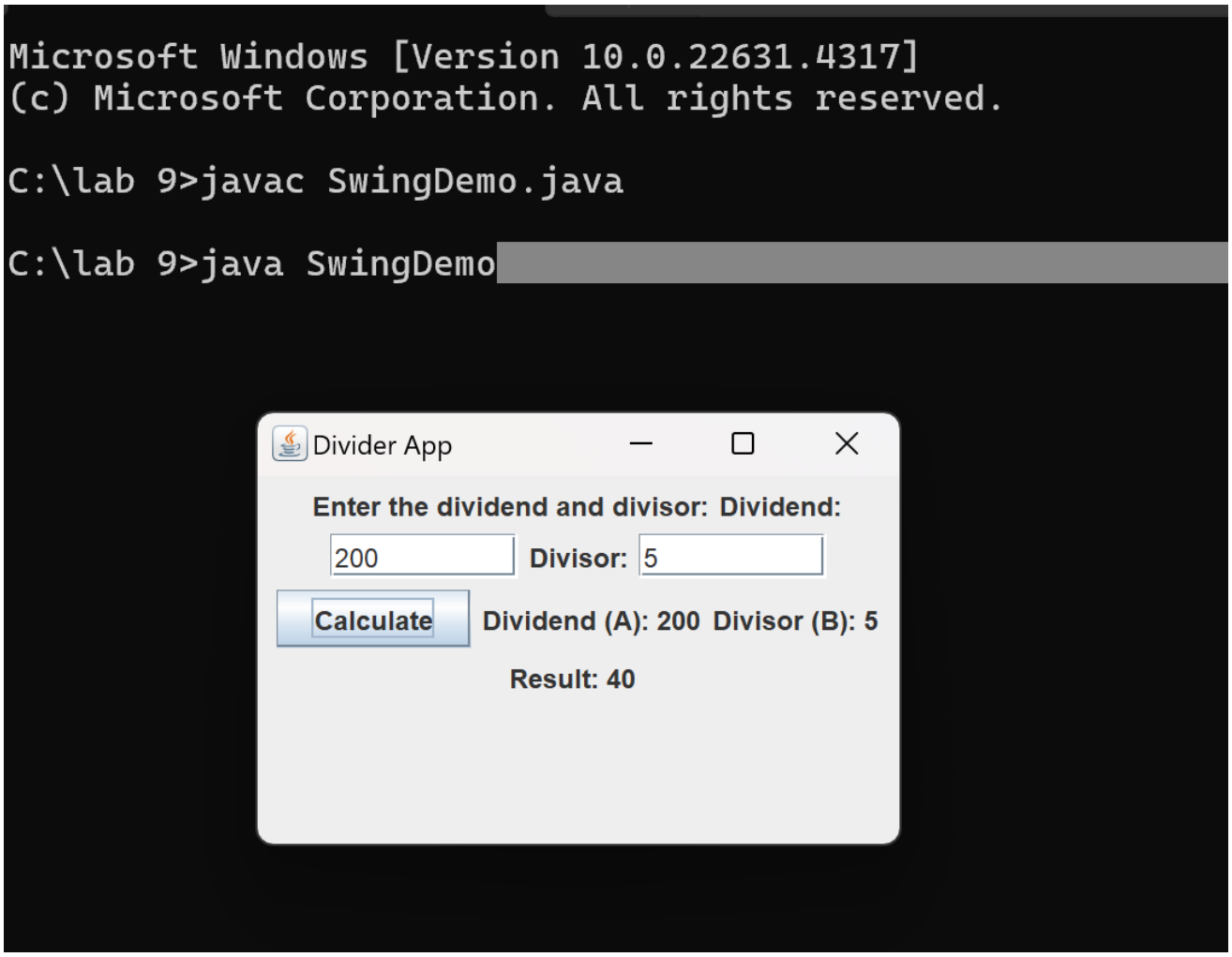


CODE:

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

class SwingDemo {
   SwingDemo() {
      JFrame jfrm = new JFrame("Divider App");
      jfrm.setSize(300, 200);
      jfrm.setLayout(new FlowLayout());
      jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      JLabel jlab = new JLabel("Enter the dividend and divisor:");
      JTextField ajtf = new JTextField(8);
      JTextField bjtf = new JTextField(8);
      JButton button = new JButton("Calculate");
      JLabel err = new JLabel();
      JLabel alab = new JLabel();
      JLabel blab = new JLabel();
      JLabel anslab = new JLabel();

      jfrm.add(jlab);
      jfrm.add(new JLabel("Dividend:"));
```

```
        jfrm.add(ajtf);
        jfrm.add(new JLabel("Divisor:"));
        jfrm.add(bjtf);
        jfrm.add(button);
        jfrm.add(alab);
        jfrm.add(blab);
        jfrm.add(anslab);
        jfrm.add(err);

        button.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent evt) {
                try {
                    int a = Integer.parseInt(ajtf.getText());
                    int b = Integer.parseInt(bjtf.getText());
                    int ans = a / b;
                    alab.setText("Dividend (A): " + a);
                    blab.setText("Divisor (B): " + b);
                    anslab.setText("Result: " + ans);
                    err.setText("");
                } catch (NumberFormatException e) {
                    alab.setText("");
                    blab.setText("");
                    anslab.setText("");
                    err.setText("Error: Enter valid integers!");
                } catch (ArithmeticException e) {
                    alab.setText("");
                    blab.setText("");
                    anslab.setText("");
                    err.setText("Error: Divisor cannot be zero!");
                }
            }
        });

        jfrm.setVisible(true);
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                new SwingDemo();
            }
        });
    }
}
```

OUTPUT:


Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.
C:\lab 9>javac SwingDemo.java
C:\lab 9>java SwingDemo
Divider App
Enter the dividend and divisor: Dividend:
200
Divisor:
5
Calculate
Dividend (A): 200 Divisor (B): 5
Result: 40

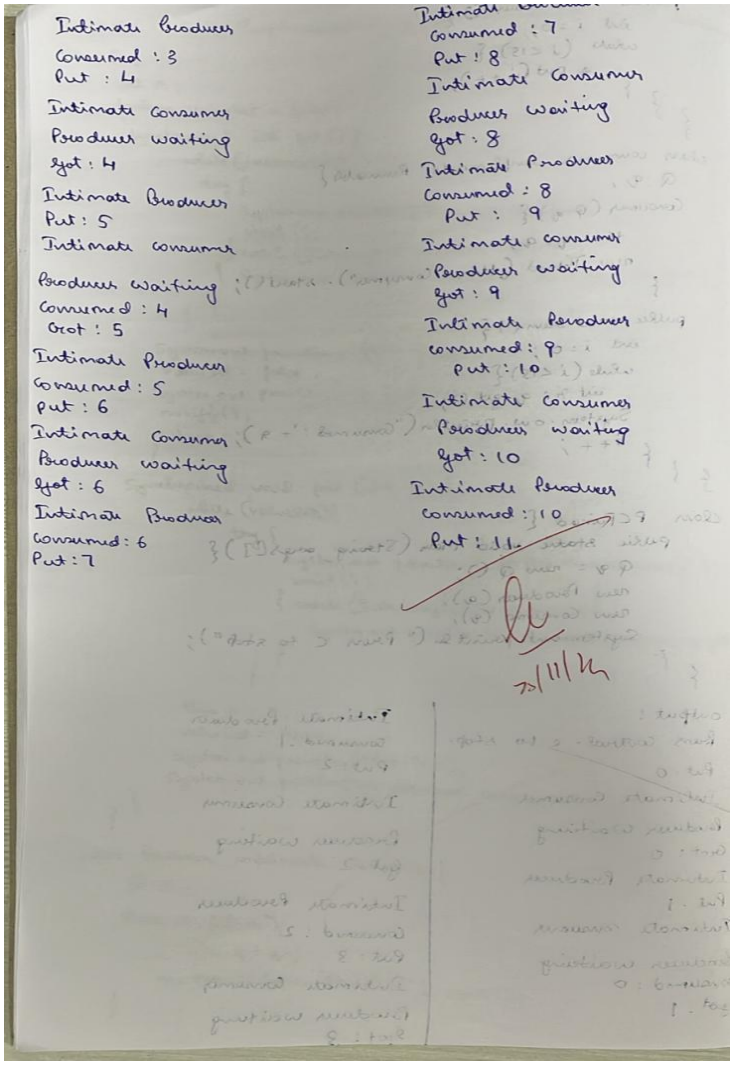
Program 10
10A-IPC

Program 10A - Implementation of a producer and consumer.

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

    synchronized void put(int n) {
        while (valueSet) {
            try {
                System.out.println("\n Producer waiting");
                wait();
            } catch (InterruptedException e) {
                System.out.println("Interrupted exception caught");
            }
        }
        this.n = n;
        valueSet = true;
        System.out.println("Put: " + n);
        System.out.println("\n Intimate consumer\n");
    }
}

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 C to stop");
    }
}
```

output:

```
Press Control - c to stop.
Put: 0
Intimate Consumer
Producer Waiting
Got: 0
Intimate Producer
Put: 1
Intimate consumer
Producer waiting
consumed: 0
Got: 1
Intimate Producer
consumed: 1
Put: 2
Intimate Consumer
Producer waiting
Got: 2
Intimate Producer
Consumed: 2
Put: 3
Intimate Consumer
Producer waiting
Got: 3
```



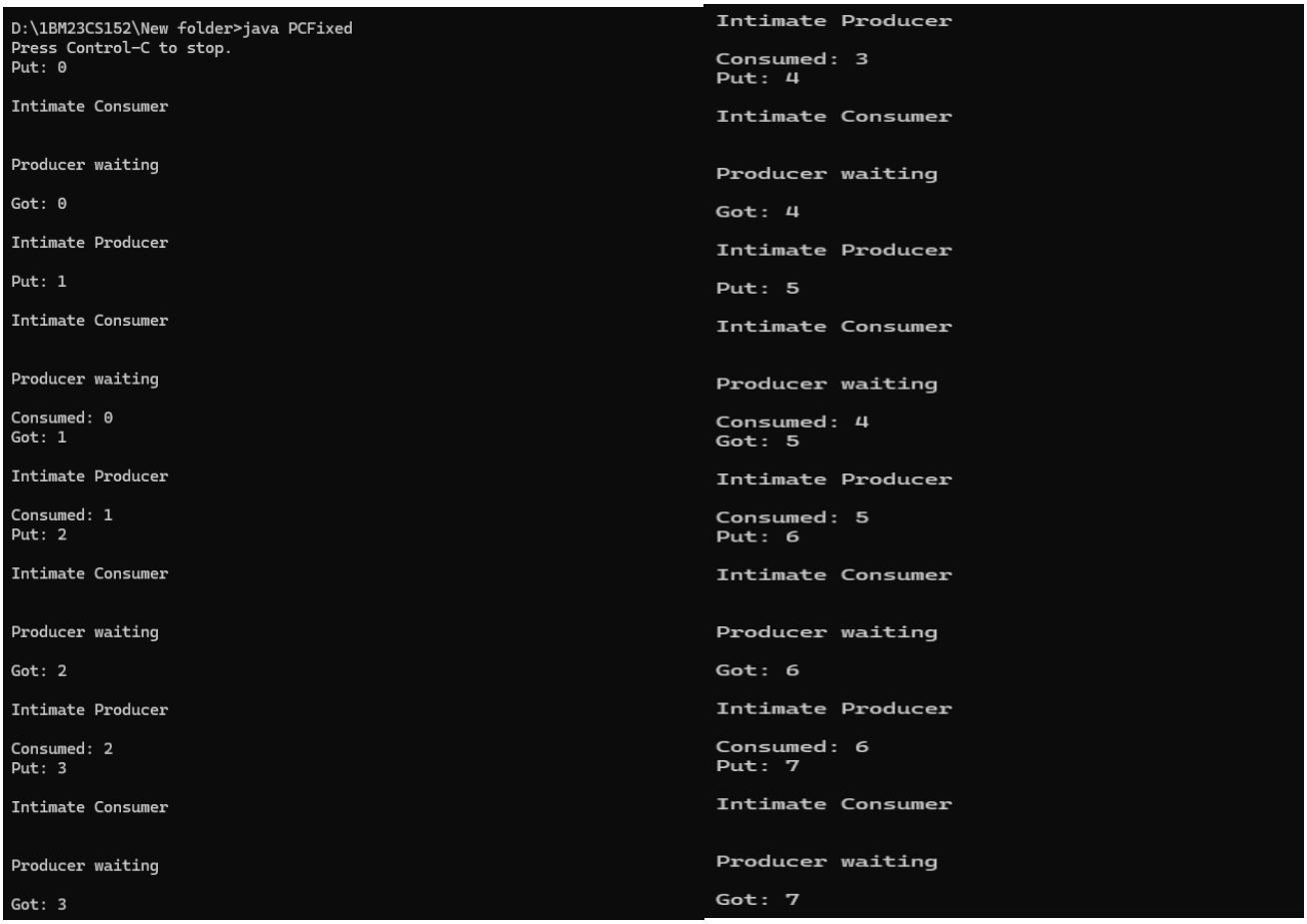
CODE:

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

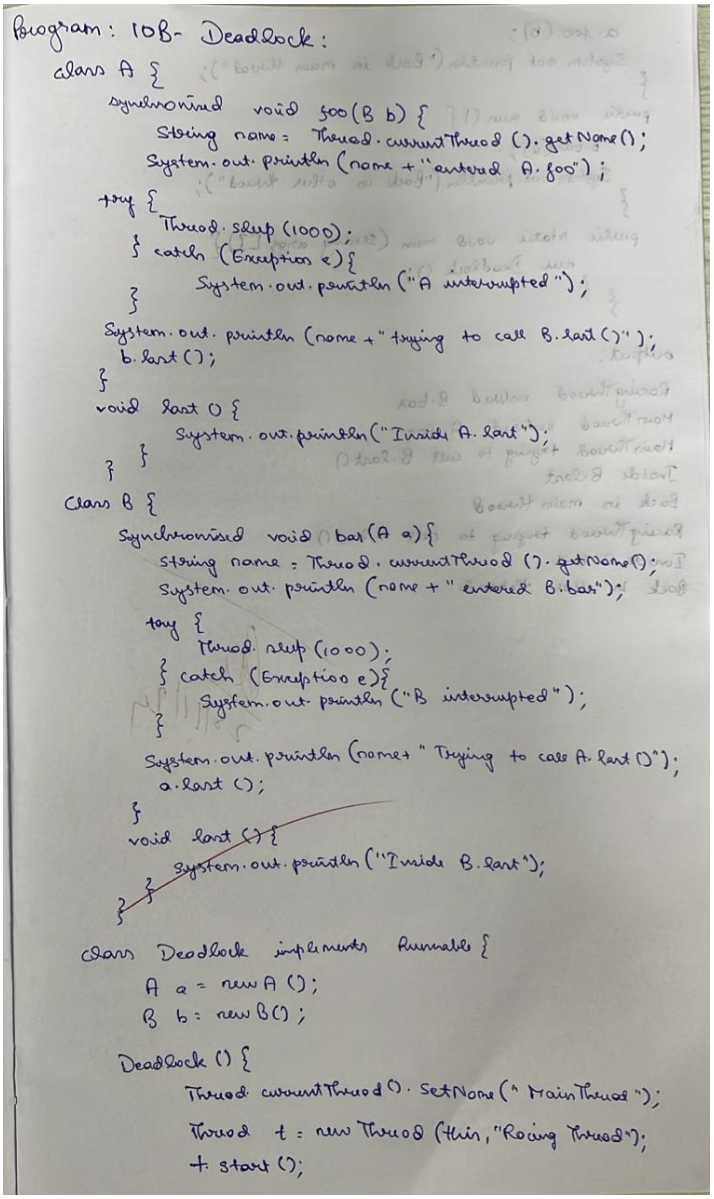
OUTPUT:



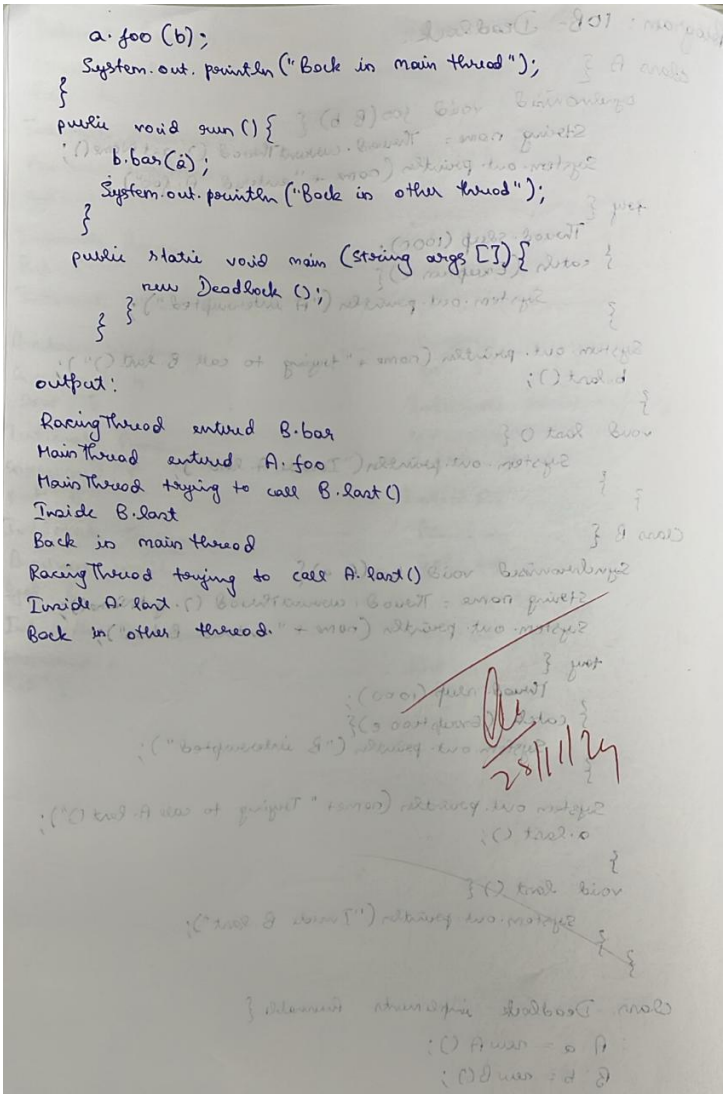
```
Intimate Producer

Consumed: 7
Put: 8

Intimate Consumer


Producer waiting

Got: 8

Intimate Producer

Consumed: 8
Put: 9

Intimate Consumer


Producer waiting

Got: 9

Intimate Producer

Consumed: 9
Put: 10

Intimate Consumer


Producer waiting

Got: 10

Intimate Producer

Consumed: 10
Put: 11

Intimate Consumer


Producer waiting

Got: 11
```

```
Intimate Producer

Consumed: 11
Put: 12

Intimate Consumer


Producer waiting

Got: 12

Intimate Producer

Consumed: 12
Put: 13

Intimate Consumer


Producer waiting

Got: 13

Intimate Producer

Consumed: 13
Put: 14

Intimate Consumer

Got: 14

Intimate Producer

Consumed: 14

D:\1BM23CS152\New folder>KISHEN DEEPAK-1BM23CS152
```

Program 10B
DEADLOCK



```
Program: 10B- 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();
    }
    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();
    }
    void last() {
        System.out.println("Inside B.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);
        System.out.println("Back in main thread");
    }
    public void run() {
        b.bar(a);
        System.out.println("Back in other thread");
    }
    public static void main(String args[]) {
        new Deadlock();
    }
}
```

output:

```
RacingThread entered B.bar
MainThread entered A.foo
MainThread trying to call B.last()
Inside B.last
Back in main thread
RacingThread trying to call A.last()
Inside A.last
Back in other thread.
```

25/11/24

CODE:

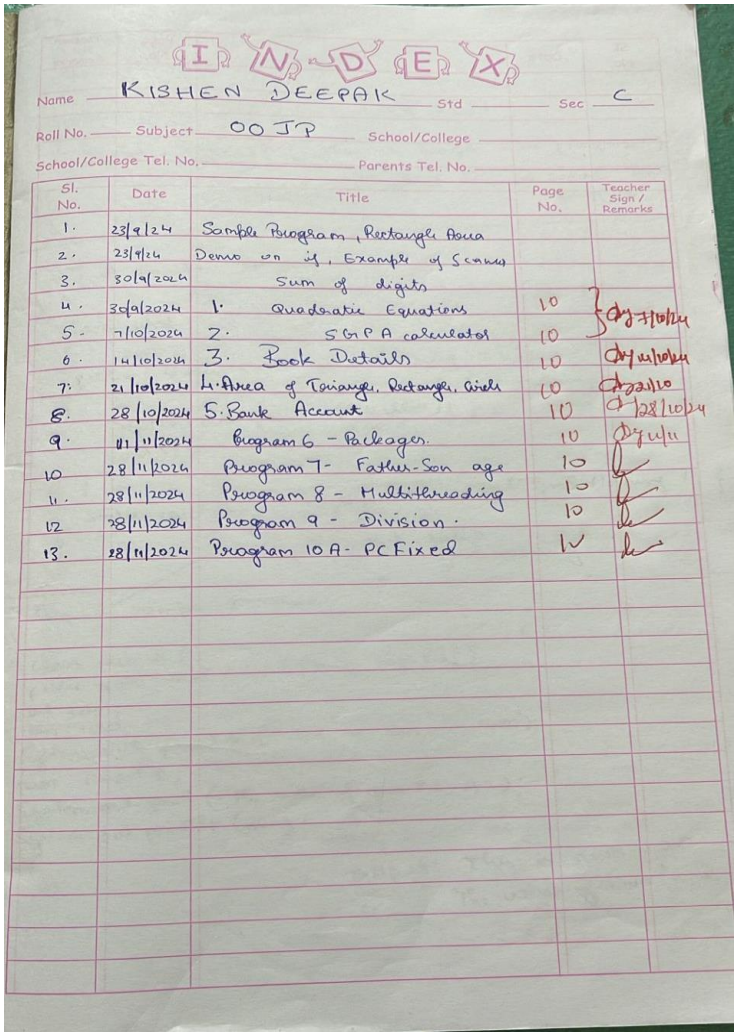
```
class A {
   synchronized void foo(B b) {
      String name = Thread.currentThread().getName();
      System.out.println(name + " entered A.foo");

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

      System.out.println(name + " trying to call B.last()");
```

```
        b.last();
    }

    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();
    }

    void last() {
        System.out.println("Inside B.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);
        System.out.println("Back in main thread");
    }

    public void run() {
        b.bar(a);
        System.out.println("Back in other thread");
    }

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

OUTPUT:

```
D:\1BM23CS152\PROGRAM 10B-DEADLOCK>javac Deadlock.java

D:\1BM23CS152\PROGRAM 10B-DEADLOCK>java Deadlock
RacingThread entered B.bar
MainThread entered A.foo
MainThread trying to call B.last()
Inside B.last
Back in main thread
RacingThread trying to call A.last()
Inside A.last
Back in other thread

D:\1BM23CS152\PROGRAM 10B-DEADLOCK>Kishen Deepak-1BM23CS152
```

FINAL SIGNATURE AND SUBMITION:



INDEX

Name KISHEN DEEPAK Std Sec C

Roll No. Subject OOJP School/College

School/College Tel. No. Parents Tel. No.

Sl. No.	Date	Title	Page No.	Teacher Sign / Remarks
1.	23/9/24	Sample Program, Rectangle Area		
2.	23/9/24	Demo on if, Example of Scanner		
3.	30/9/2024	Sum of digits		
4.	30/9/2024	1. Quadratic Equations	10	7/10/24
5.	7/10/2024	2. SGPA calculator	10	
6.	14/10/2024	3. Book Details	10	14/10/24
7.	21/10/2024	4. Area of Triangle, Rectangle, Circle	10	21/10
8.	28/10/2024	5. Bank Account	10	28/10/24
9.	11/11/2024	Program 6 - Packages.	10	11/11
10	28/11/2024	Program 7 - Father-Son age	10	
11.	28/11/2024	Program 8 - Multithreading	10	
12	28/11/2024	Program 9 - Division.	10	
13.	28/11/2024	Program 10A - PCFixed	10	