VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT

on

OBJECT ORIENTED JAVA PROGRAMMING

Submitted by

SHANKAR SHIVAPPA PUJAR (1BM23CS309)

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 **SHANKAR SHIVAPPA PUJAR (1BM23CS309),** 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 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

GITHUIB LINK = https://github.com/shankar045/java-lab

LABORATORY PROGRAM – 01

Develop a Java program that prints all real solutions to the quadratic equation ax2+bx+c = 0. Read in a, b, c and use the quadratic formula. If the discriminate b2-4ac is negative, display a message stating that there are no real solutions.

```
import java.util.Scanner;

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

      Scanner input = new Scanner(System.in);

      System.out.print("Enter coefficient a: ");
      double a = input.nextDouble();
      System.out.print("Enter coefficient b: ");
      double b = input.nextDouble();
      System.out.print("Enter coefficient c: ");
      double c = input.nextDouble();

      if (a == 0) {
         if (b != 0) {

            double root = -c / b;
            System.out.println("This is a linear equation. The solution is: " + root);
         } else if (c == 0) {

            System.out.println("The equation has infinitely many solutions.");
         } else {

            System.out.println("The equation has no solutions.");
         }
      } else {

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

         if (discriminant > 0) {
            double root1 = (-b + Math.sqrt(discriminant)) / (2 * a);
            double root2 = (-b - Math.sqrt(discriminant)) / (2 * a);
            System.out.println("The equation has two real solutions: " + root1 + " and " +
```

```
root2);
            } else if (discriminant == 0) {
                double root = -b / (2 * a);
                System.out.println("The equation has one real solution: " + root);
            } else {
                System.out.println("The equation has no real solutions.");
            }
        }

        input.close();
    }
}
```

OUTPUT

```
C:\Users\Abhishek\Desktop>java QuadraticEquation.java
Enter coefficient a: 5
Enter coefficient b: 6
Enter coefficient c: 9
The equation has no real solutions.

C:\Users\Abhishek\Desktop>java  QuadraticEquation.java
Enter coefficient a: 8
Enter coefficient b: -1
Enter coefficient c: -9
The equation has two real solutions: 1.125 and -1.0

C:\Users\Abhishek\Desktop>java QuadraticEquation.java
Enter coefficient a: 4
Enter coefficient b: -8
Enter coefficient c: -2
The equation has two real solutions: 2.224744871391589 and -0.22474487139158894

C:\Users\Abhishek\Desktop>java QuadraticEquation.java
Enter coefficient a: 5
Enter coefficient b: 6
Enter coefficient c: 3
The equation has no real solutions.
```

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 equation formula. if the discriminate b^2-4ac is negative, display a message stating that there are no real solutions.

program:

```
class QDT {
    double a;
    double b;
    double c;
    double d;
    double x1;
    double x2;
Scanner s = new Scanner();
System.out.println("Enter co-efficents a, b, c: ");
    a = s.nextDouble();
    b = b.nextDouble();
    c = c.nextDouble();
    d = b*-4*a*c;
    }
    void findRoots() {
      if(d>0) {
      x1 = (-b+ Math.sqrt(d))/(2*a);
      x2 = c-b-Math.sqrt(d)/(2*a);
    System.out.println("Roots are real and distinct
                        " "+"x1+" "+x2);
      }
    elseif (d==0){
      x1=-b/(2*a);
```

```
System.out.println("Roots are real and quadratic-" +
                    x1);}
Else {
  double real part = -b/(2*a);
  double imgpart = Math.sqrt(-d)/(2*a);
  System.out.println("Roots are imaginary;
      real part ~~real~~ + " " + imgpart + "i and " + real +
      "-" + imgpart + "i");
    }
  }
}
class ROOTS {
  public static void Main(String args[]){
    QDT q = new QDT();
    q.getData();
    q.findRoots();
    }
  }
```

Output:

Enter co-efficient a: 2
enter co-efficient b: 4
enter co-efficient c: 8
The Equation ~~has no real~~ and D is negative

LABORATORY PROGRAM – 02

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;

class Student {
    String usn;
    String name;
    int[] credits;
    int[] marks;


    void acceptDetails() {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter USN: ");
        usn = sc.nextLine();

        System.out.print("Enter Name: ");
        name = sc.nextLine();

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

        credits = new int[n];
        marks = new int[n];

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

        System.out.println("Enter marks for each subject:");
        for (int i = 0; i < n; i++) {
            System.out.print("Marks for subject " + (i + 1) + ": ");
            marks[i] = sc.nextInt();
```

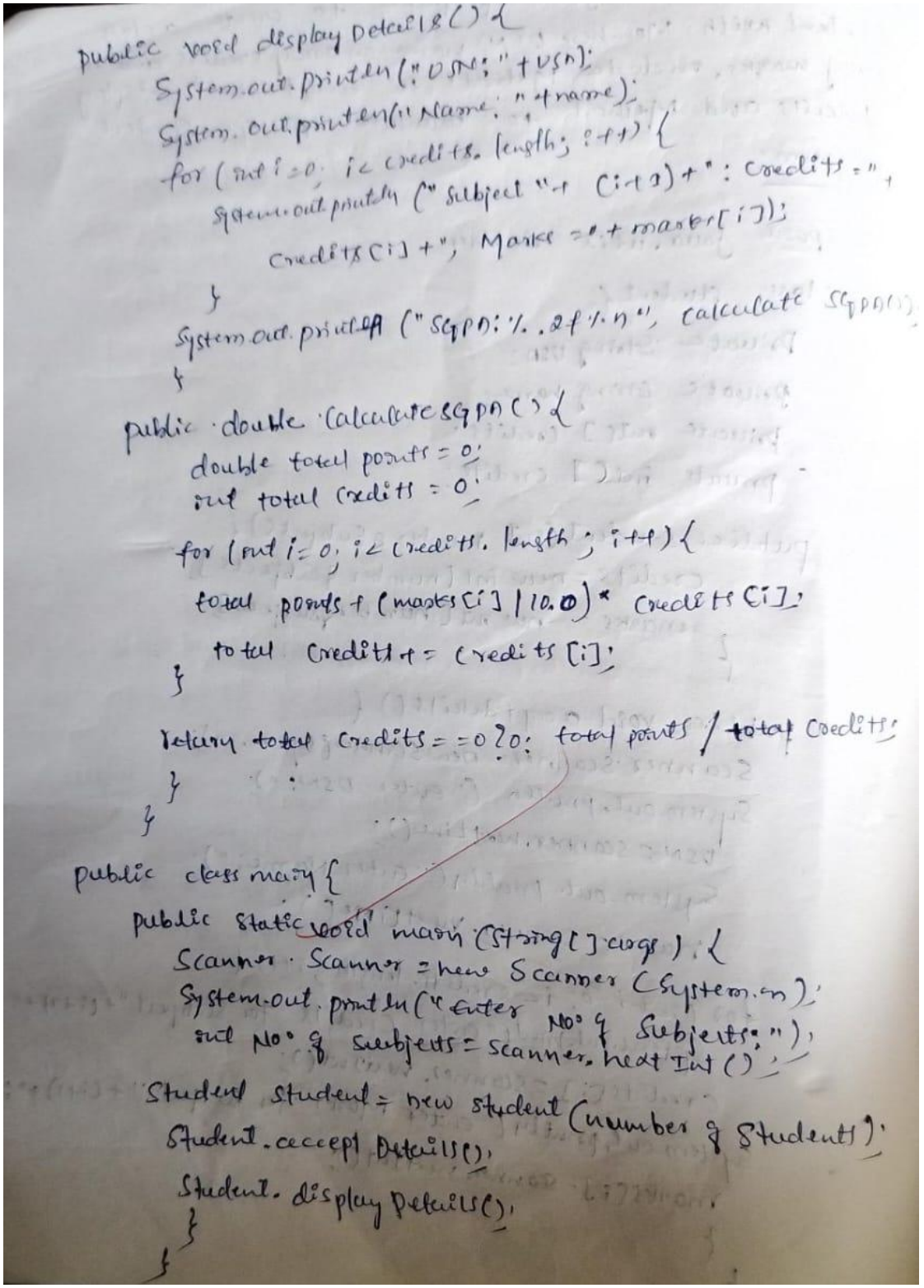
```
        }
    }


    double calculateSGPA() {
        int totalCredits = 0;
        int weightedSum = 0;

        for (int i = 0; i < credits.length; i++) {
            int gradePoint = getGradePoint(marks[i]);
            weightedSum += gradePoint * credits[i];
            totalCredits += credits[i];
        }

        return (double) weightedSum / totalCredits;
    }


    int getGradePoint(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;
    }


    void displayDetails() {
        System.out.println("\nStudent Details:");
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        System.out.println("SGPA: " + calculateSGPA());
    }

    public static void main(String[] args) {
        Student student = new Student();
        student.acceptDetails();
        student.displayDetails();
    }
}
```
OUTPUT

```
PS C:\Users\Abhishek\Desktop\1bm23cs309> javac Student.java
PS C:\Users\Abhishek\Desktop\1bm23cs309> java Student.java
Enter USN: 1BM23CS309
Enter Name: SHANKAR PUJAR
Enter the number of subjects: 3
Enter credits for each subject:
Credits for subject 1: 4
Credits for subject 2: 3
Credits for subject 3: 4
Enter marks for each subject:
Marks for subject 1: 89
Marks for subject 2: 82
Marks for subject 3: 75

Student Details:
USN: 1BM23CS309
Name: SHANKAR PUJAR
SGPA: 8.636363636363637
PS C:\Users\Abhishek\Desktop\1bm23cs309>
```

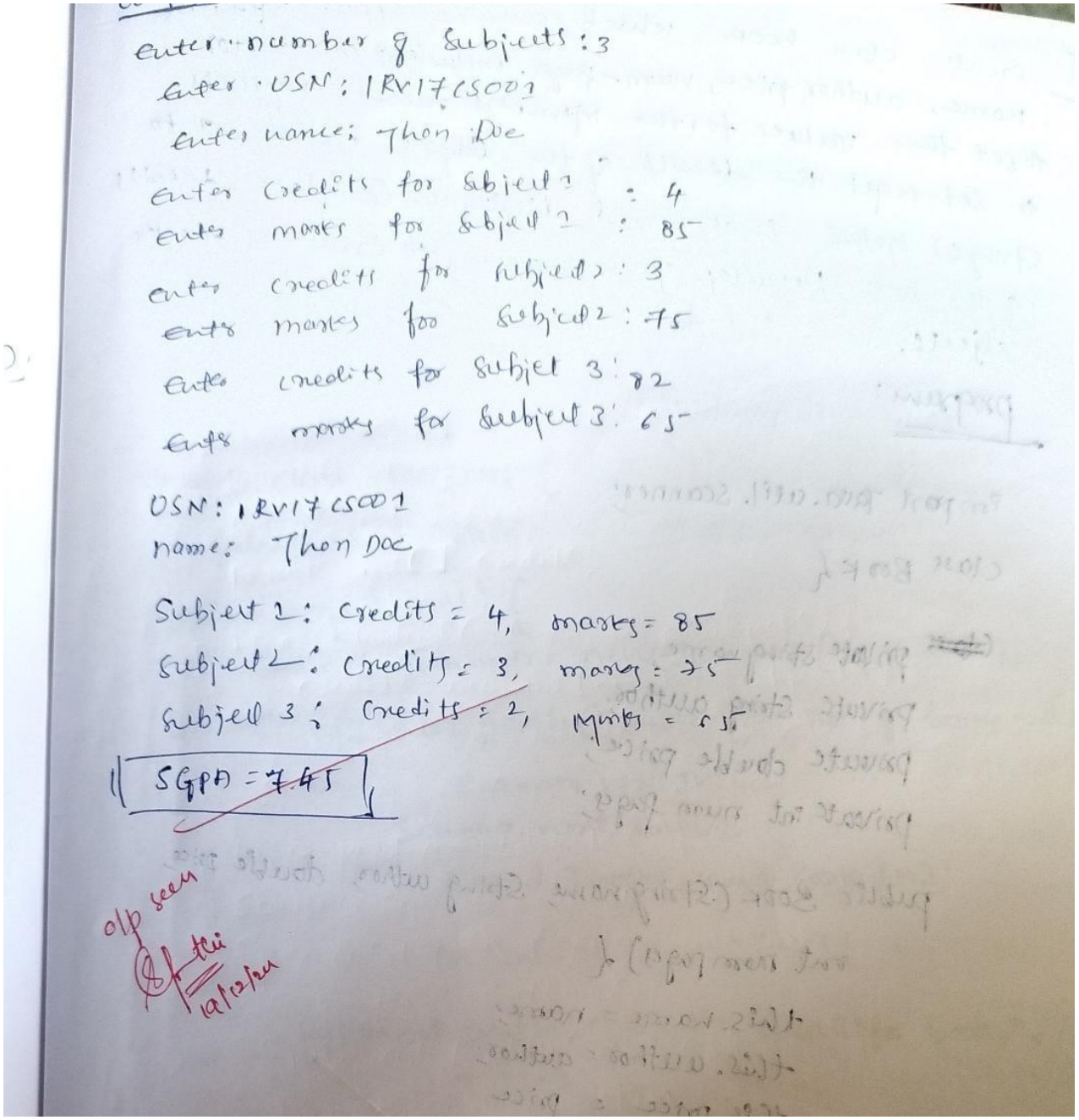
2nd) Develop a java program to create a class Student with Members USN, name, an array credits and an array marks, include Methods to accept and display details and Method to calculate SGPA of a Student

program:

```
import java.util.Scanner;

class student{
    private String usn;
    private String name;
    private int[] credits;
    private int[] credits;

public Student(int number of Subjects){
    credits = new int[number of subjects];
    marks = new int[number of subjects];
    }

public void acceptdetails(){
    Scanner scanner = new Scanner(System.in);
    System.out.println("Enter USN: ");
    USN = scanner.nextLine();
    System.out.println("Enter Name: ");
    name = scanner.nextLine();
for(int i=0; i< credits.length; i++){
    System.out.print("Enter Credits for Subject "+(i+1)+": ");
    Credits[i] = scanner.nextInt();
System.out.println("Enter Marks for Subject "+(i+1)+": ");
    marks[i] = scanner.nextInt();
    }
}
```



```
public void displayDetails() {
    System.out.println("USN: " + usn);
    System.out.println("Name: " + name);
    for (int i = 0; i < credits.length; i++) {
        System.out.println("Subject " + (i+1) + ": Credits = " +
            credits[i] + ", Marks = " + marks[i]);
    }
    System.out.printf("SGPA: %.2f%n", calculateSGPA());
}

public double calculateSGPA() {
    double totalPoints = 0;
    int totalCredits = 0;
    for (int i = 0; i < credits.length; i++) {
        totalPoints += (marks[i] / 10.0) * credits[i];
        totalCredits += credits[i];
    }
    return totalCredits == 0 ? 0 : totalPoints / totalCredits;
}
}

public class main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter No. of Subjects: ");
        int No. of Subjects = scanner.nextInt();
        Student student = new Student(number of Students);
        Student.acceptDetails();
        Student.displayDetails();
    }
}
```


Enter number of Subjects : 3
Enter USN : 1RV17CS001
Enter name : Thon Doe
Enter Credits for Subject 1 : 4
Enter marks for Subject 1 : 85
Enter Credits for Subject 2 : 3
Enter marks for Subject 2 : 75
Enter Credits for Subject 3 : 2
Enter marks for Subject 3 : 65

USN : 1RV17CS001
name : Thon Doe

Subject 1 : Credits = 4, marks = 85
Subject 2 : Credits = 3, marks = 75
Subject 3 : Credits = 2, Marks = 65

SGPA = 7.45

o/p seen
thu
19/12/24

LABORATORY PROGRAM – 03

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.

```
import java.util.Scanner;
class Book{
 private String name;
 private String author;
 private double price;
 private int num_pages;

 public Book(String name,String author,double price,int num_pages){
  this.name=name;
  this.author=author;
  this.price=price;
  this.num_pages=num_pages;}
 public String getName(){
  return name;}
 public void setName(String name){
  this.name=name;}
 public String getauthor(){
  return author;}
 public void setauthor(String author){
  this.author=author;}
 public double getprice(){
  return price;}
 public void setprice(){
  this.price=price;}
 public int getnumpages(){
  return num_pages;}
 public void setnumpages(){
  this.num_pages=num_pages;}

 public String toString(){
  return("Book[name:"+name+" author:"+author+" price:"+price+"
```

```
pages:"+num_pages);}
}
public class Bookmain{
 public static void main(String[] args){
  Scanner sc=new Scanner(System.in);
  System.out.println("enter no of books");
  int n=sc.nextInt();
  sc.nextLine();
  Book[] books=new Book[n];
  for(int i=0;i<n;i++){
   System.out.println("enter the details of book"+(i+1)+":");
   System.out.print("enter name:");
   String name=sc.nextLine();
   System.out.print("enter author:");
   String author=sc.nextLine();
   System.out.print("enter price:");
   int price=sc.nextInt();
   System.out.print("enter the no of pages:");
   int num_pages=sc.nextInt();
   books[i]=new Book(name,author,price,num_pages);}
  System.out.print("\ndetails of books");
  for(int i=0;i<n;i++){
   System.out.println(books[i].toString());
  }
}
}
```

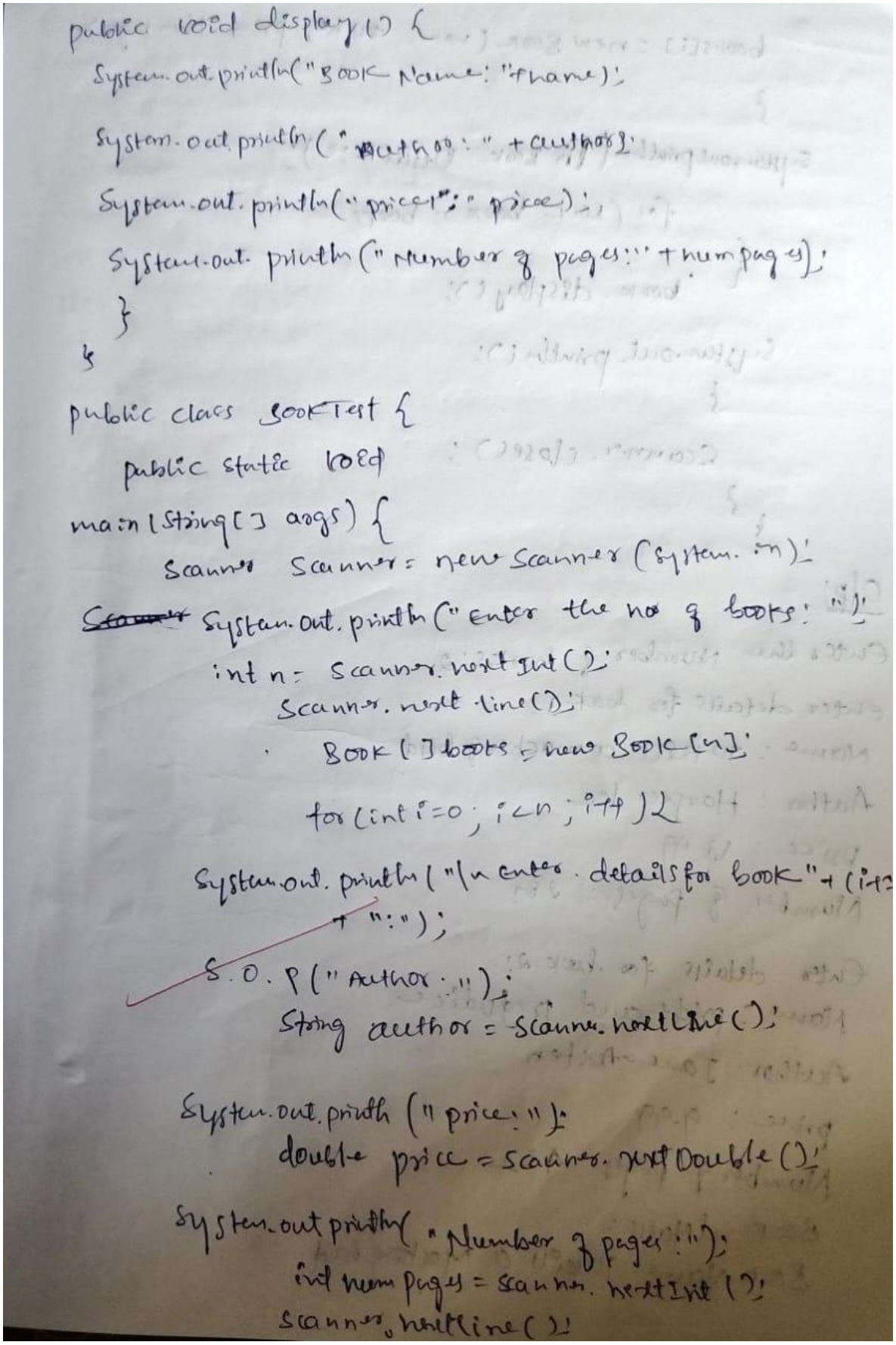
OUTPUT

```
Microsoft Windows [Version 10.0.22631.4460]
(c) Microsoft Corporation. All rights reserved.

C:\1BM23CS309>javac Bookmain.java

C:\1BM23CS309>java Bookmain.java
enter no of books
2
enter the details of book1:
enter name:the night flower
enter author:d.chandrashekar
enter price:250
enter the no of pages:400
enter the details of book2:
enter name:enter author:heli hogu karana
enter price:500
enter the no of pages:750

details of booksBook[name:the night flower author:d.chandrashekar price:250.0 pages:400
Book[name: author:heli hogu karana price:500.0 pages:750

C:\1BM23CS309>
```

03: Create class book which contains [illegible]
name, author price, num-pages. include a constructor
to set the values for the members include a methods
to set and get the details of the objects include a to
String() method that could display the complete details
of the book Develop java program to create n -books
objects.

program:

```
import java.util.Scanner;
class Book{
    private string name;
    private String author;
    private double price;
    private int num pages;
public Book (String name, String author, double price
    int num pages) {
        this.name = name;
        this.author = author;
        this.price = price
        this.num pages = num pages; }
public String getName() {
    return name;
}
public void setName (String name) {
        this.name = name;
    }
```

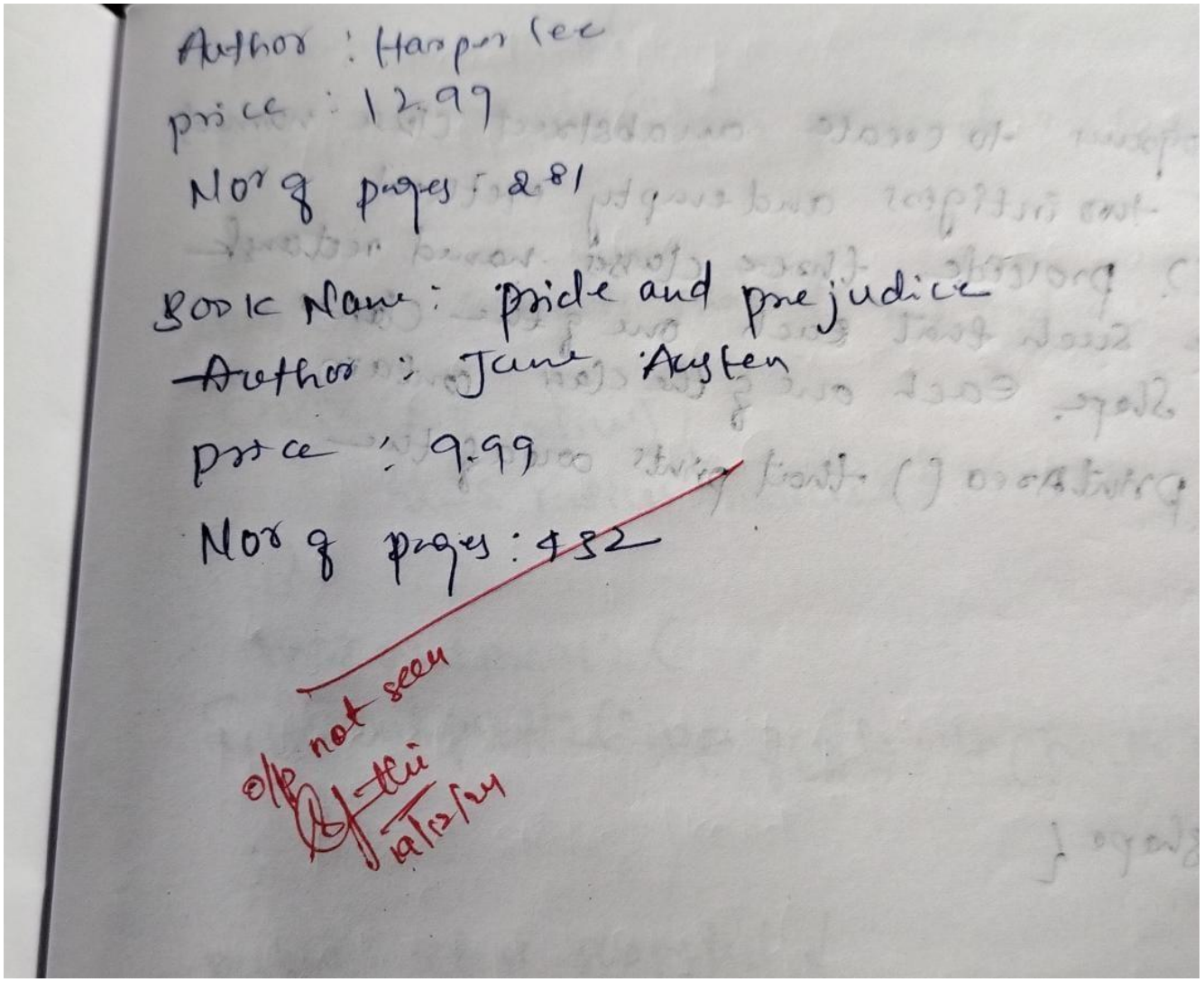


```
public void display() {
    System.out.println("Book Name: "+name);
    System.out.println("Author: "+author);
    System.out.println("price:"+price);
    System.out.println("Number of pages:"+numpages);
    }
}
public class BookTest {
    public static void
main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    System.out.println("Enter the no of books: ");
    int n = scanner.nextInt();
    scanner.nextLine();
    Book[] books = new Book[n];
    for(int i=0; i<n; i++){
    System.out.println("\n enter details for book"+(i+1
    +":");
    S.O.P("Author:");
    String author = scanner.nextLine();
    System.out.println("price:");
    double price = scanner.nextDouble();
    System.out.println("Number of pages:");
    int numpages = scanner.nextInt();
    scanner.nextLine();
```

```
        books[i] = new Book(name, author, price, numPages);
    }
    System.out.println("\n Book Details:");
    for (Book book : books) {
        book.display();
        System.out.println();
    }
    scanner.close();
}
}
```

O/P:

```
Enter the Number of books: 2
Enter details for book 1:
Name: To Kill a mocking bird
Author: Harpeer lee
price: 12.99
Number of pages: 281
Enter details for book 2:
Name: pride and prejudice
Author: Jane Austen
price: 9.99
Number of pages: 432
Book details:
Book Name: To kill a Mocking bird
```

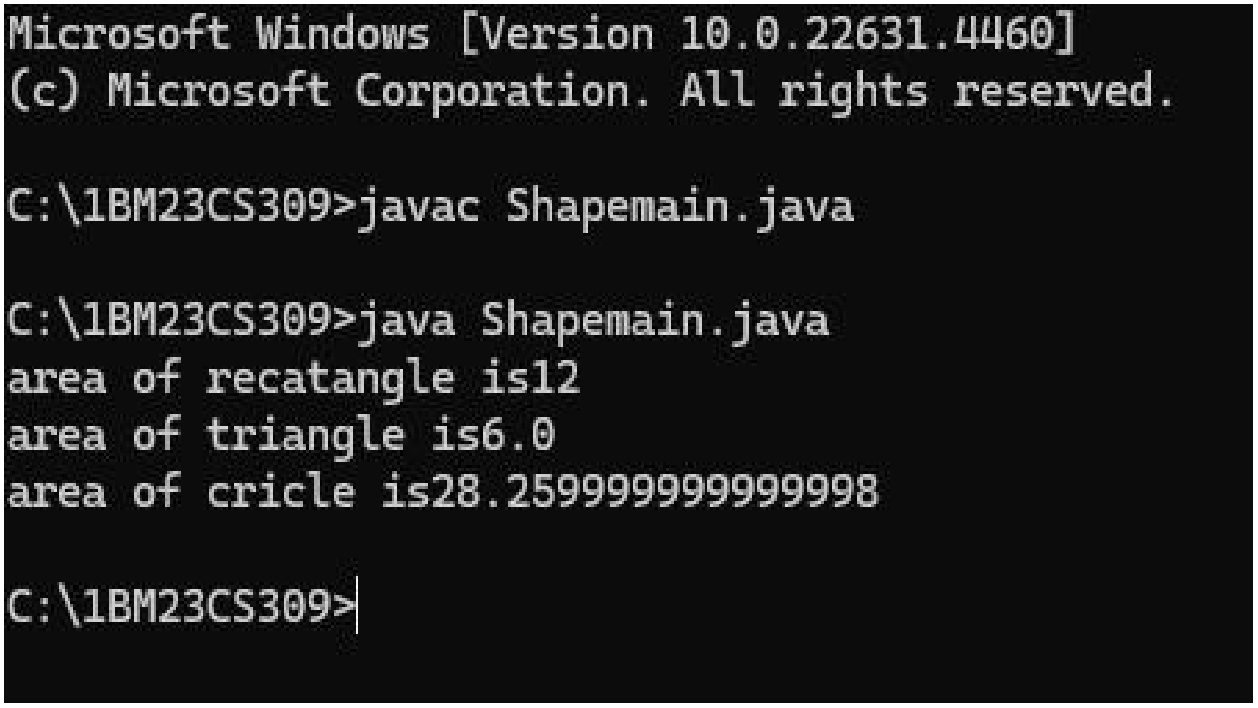

Author : Harper lee
price : 12.99
Nor of pages : 281
Book Name : pride and prejudice
Author : Jane Austen
price : 9.99
Nor of pages : 432
o/p not seen
14/12/24

LABORATORY PROGRAM – 04

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 d1,d2;
 abstract void printarea();}

class Rectangle extends Shape{
 Rectangle(int d1,int d2){
  this.d1=d1;
  this.d2=d2;}
 void printarea(){
  System.out.println("area of recatangle is"+(d1*d2));}
}

class Triangle extends Shape{
 Triangle(int d1,int d2){
  this.d1=d1;
  this.d2=d2;}
 void printarea(){
  System.out.println("area of triangle is"+(0.5*d1*d2));}
}

class Circle extends Shape{
 Circle(int d1,int d2){
  this.d1=d1;
  this.d2=1;}
 void printarea(){
  System.out.println("area of cricle is"+(3.14*d1*d1));}}

class Shapemain{
 public static void main(String[] args){
  Rectangle rectangle=new Rectangle(3,4);
  rectangle.printarea();
  Triangle triangle=new Triangle(3,4);
  triangle.printarea();
  Circle circle=new Circle(3,4);  circle.printarea(); }}
```

OUTPUT


Microsoft Windows [Version 10.0.22631.4460]
(c) Microsoft Corporation. All rights reserved.
C:\1BM23CS309>javac Shapemain.java
C:\1BM23CS309>java Shapemain.java
area of recatangle is12
area of triangle is6.0
area of cricle is28.259999999999998
C:\1BM23CS309>

ONE P:- 9

Develop a java program to create an abstract class named Shape the contains two intigers and empty method named print area(). provide three classes named rectangle Triangle and circle such that each one of the classes extends the class Shape. each one of the class contain only the method printArea() that prints area of the given Shape

program:

```
abstract class shape {
 abstract void
  printArea();
}
Class Rectangle extends shape { int length, breadth;

    Rectangle (int length, int breadth) {
        this.length = length;
        this.breadth = breadth;
      }
     void printArea() {
     System.out.println(" Area of Rectangle :" + (length * breadth)
} }
Class ~~Rectangle~~ Triangle extends shape {
      int base, height;
     Triangle (int base, int height) {
        this.base = base;
        this.height = height
      }
```

```
void printArea () {
System.out.println ("Area of Triangle:" + 10.5 * base * height));
}
}

class Circle extends Shape { int radius;
Circle (int radius) {
    this.radius = radius;
}
void printArea() {
System.out.println ("Area of Circle:" + (Math.PI * radius * radius));
}
}

public class ShapeTest {
  public static void
main (String[] args) {
    Shape rectangle = new Rectangle (10, 5);
    Shape triangle = new Triangle (10.8);
    Shape circle = new Circle (7);
    rectangle.printArea ();
    triangle.printArea ();
    circle.printArea ();
}
}
```

O/P:

Area of Rectangle : 50
Area of Triangle : 40.0
Area of circle : 157.938040

o/p not seen

LABORATORY PROGRAM – 05

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: a) Accept deposit from customer and update the balance. b) Display the balance. c) Compute and deposit interest d) 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 {
    protected String customerName;
    protected String accountNumber;
    protected String accountType;
    protected double balance;



    public Account(String customerName, String accountNumber, String accountType, double initialBalance) {
        this.customerName = customerName;
        this.accountNumber = accountNumber;
        this.accountType = accountType;
        this.balance = initialBalance;
```

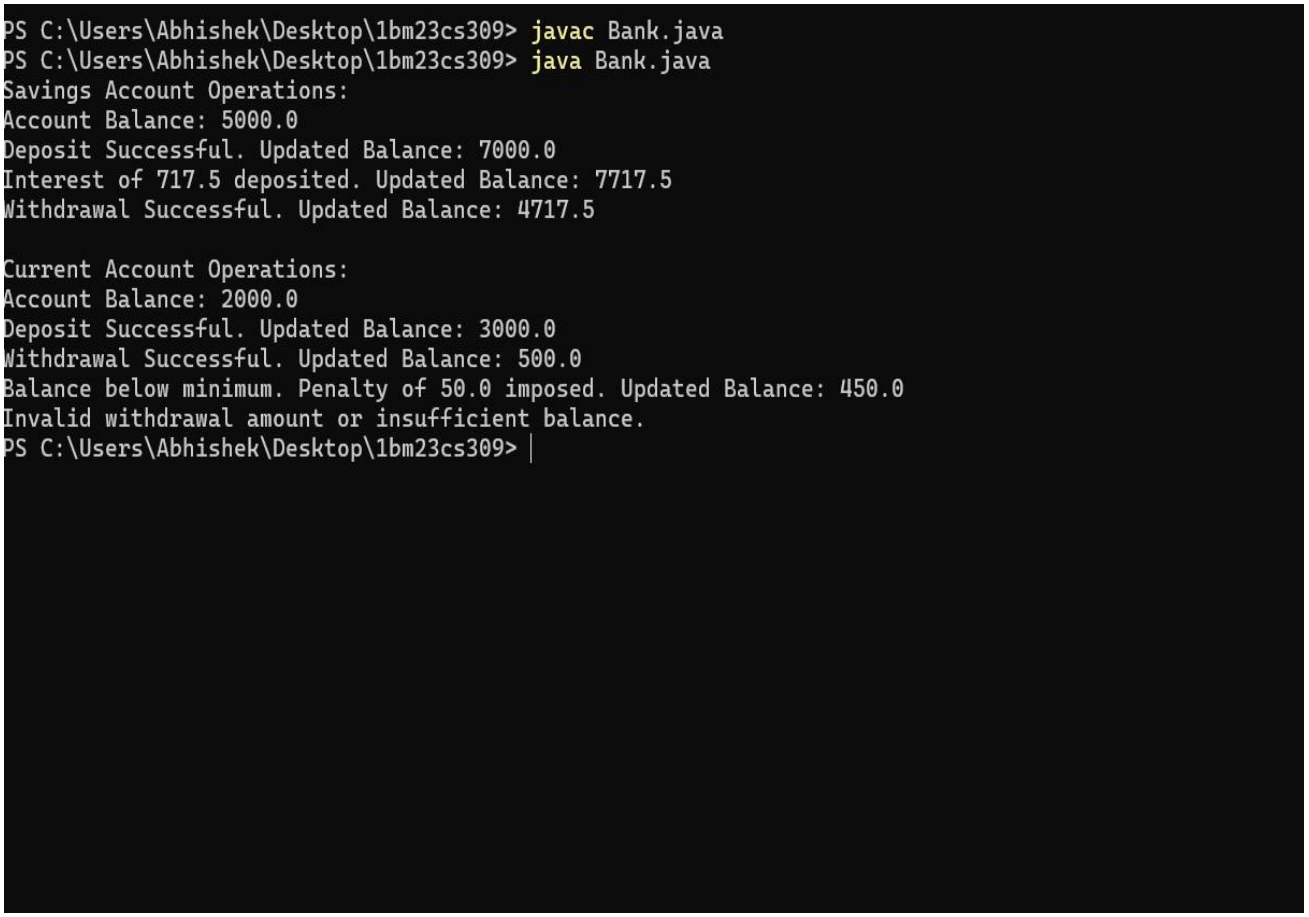
```
    }

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

    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            System.out.println("Deposit Successful. Updated Balance: " + balance);
        } else {
            System.out.println("Invalid deposit amount.");
        }
    }
}


class SavAcct extends Account {
    private static final double INTEREST_RATE = 0.05;
    public SavAcct(String customerName, String accountNumber, double initialBalance)
{
        super(customerName, accountNumber, "Savings", initialBalance);
    }


    public void computeAndDepositInterest(int years) {
```

```
        double interest = balance * Math.pow(1 + INTEREST_RATE, years) - balance;
        balance += interest;
        System.out.println("Interest of " + interest + " deposited. Updated Balance: " +
balance);
    }

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



class CurAcct extends Account {
    private static final double MIN_BALANCE = 1000.0;
    private static final double PENALTY = 50.0;

    public CurAcct(String customerName, String accountNumber, double initialBalance)
{
        super(customerName, accountNumber, "Current", initialBalance);
    }
```

```
    public void withdraw(double amount) {
        if (amount > 0 && amount <= balance) {
            balance -= amount;
            System.out.println("Withdrawal Successful. Updated Balance: " + balance);
            if (balance < MIN_BALANCE) {
                balance -= PENALTY;
                System.out.println("Balance below minimum. Penalty of " + PENALTY + "
imposed. Updated Balance: " + balance);
            }
        } else {
            System.out.println("Invalid withdrawal amount or insufficient balance.");
        }
    }
}



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

        SavAcct savings = new SavAcct("Alice", "S123", 5000.0);



        CurAcct current = new CurAcct("Bob", "C456", 2000.0);

        System.out.println("Savings Account Operations:");
```

```
        savings.displayBalance();
        savings.deposit(2000);
        savings.computeAndDepositInterest(2);
        savings.withdraw(3000);

        System.out.println("\nCurrent Account Operations:");
        current.displayBalance();
        current.deposit(1000);
        current.withdraw(2500);
        current.withdraw(500);

        scanner.close();
    }
}
```

OUTPUT


PS C:\Users\Abhishek\Desktop\1bm23cs309> javac Bank.java
PS C:\Users\Abhishek\Desktop\1bm23cs309> java Bank.java
Savings Account Operations:
Account Balance: 5000.0
Deposit Successful. Updated Balance: 7000.0
Interest of 717.5 deposited. Updated Balance: 7717.5
Withdrawal Successful. Updated Balance: 4717.5
Current Account Operations:
Account Balance: 2000.0
Deposit Successful. Updated Balance: 3000.0
Withdrawal Successful. Updated Balance: 500.0
Balance below minimum. Penalty of 50.0 imposed. Updated Balance: 450.0
Invalid withdrawal amount or insufficient balance.
PS C:\Users\Abhishek\Desktop\1bm23cs309>

LAB = 5

LAB - program = 5

program:

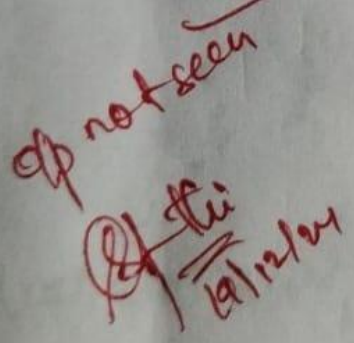
```
import java.util.Scanner;
class Account {
    String customerName;
    String accountNumber;
    double balance;
    Account(String customerName, String accountNumber,
      double balance)
    {
        this.customerName = customerName;
        this.accountNumber = accountNumber;
        this.balance = balance;
        }
    void deposit(double amount)
      {   balance += amount;
    System.out.println("Deposited: " + amount);
    }
    void displayBalance() {
  System.out.println("Current Balance:" + balance);
  }}

class SavAcct extends Account {
    final double interestRate = 0.05;
```

```
SavAcct (String customerName, String AccountNumber, double balance)
{
  super (customerName accountNumber, balance);
  }
  void addInterest() {
      double interest = balance * InterestRate;
          balance += Interest;
    System.out.println ("Interest added:" + Interest);
  }
  void withdraw (double amount) {
      if balance >= amount)
    {     balance -= amount;
    System.out.println("withdrew" + amount);
      } else {
    System.out.println(" Insufficient Balance");
      }
    }
  }
Class CurAcct extends Account{
   final double minBalance = 500.0;
    final double service charge = 50.0;
   CurAcct (String customerName, String accountNumber
   double balance)
   {
     super (customerName accountNumber, balance
   }
   void checkMinBalance() {
       if (balance < minbalance) {
           balance -= service charge;
```

```
System.out.println("Withdrew: " + amount);
        checkMyBalance();
    } else {
System.out.println("Insufficient Balance");
    }
}
}

public class BANK {
  public static void main(String[] args) {
    SAVacct SavingsAccount = new SaveAcct("Alice",
      1000);
  System.out.println("Savings Account:");
  SavingsAccount.deposit(500);
  SavingsAccount.addInterest();
  SavingsAccount.withdraw(300);
    SavingsAccount.displayBalance();
    CurrAcct CurrentAccount = new CurrAcc("Bob",
  System.out.println("\n Current Account:");
    CurrentAccount.deposit(200);
  CurrentAccount.withdraw(400);
  Current Account.withdraw(600);
  Current Account.displayBalance();
}
}
```

o/p

Saving Account:
Deposited: 500.0
Interest added: 75.0
Keithdrew: 300.0
Current Bellance: 1275.0
Current Account:
Deposited: 300.0
Keithdrew: 2000
Keithdrew: 600.0
Service charge imposed: 50.0
Current Balance: 150.0



LABORATORY PROGRAM – 06

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.

Internals.java

```
import java.util.Scanner;


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


    public void inputCIEmarks() {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter Internal Marks for 5 subjects:");
        for (int i = 0; i < 5; i++) {
            System.out.print("Subject " + (i + 1) + ": ");
            internalMarks[i] = s.nextInt();
        }
    }
}
```

Externals .java

```
import java.util.Scanner;

public class Externals extends Internals {
```

```
    private int[] seeMarks = new int[5];
    private int[] finalMarks = new int[5];

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

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

    public void displayFinalMarks() {
        displayStudentDetails();
        System.out.println("Final Marks for 5 subjects:");
        for (int i = 0; i < 5; i++) {
            System.out.println("Subject " + (i + 1) + ": " + finalMarks[i]);
        }
    }
}
```

#Student.java

```
import java.util.Scanner;

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

    public void inputCIEmarks() {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter Internal Marks for 5 subjects:");
        for (int i = 0; i < 5; i++) {
            System.out.print("Subject " + (i + 1) + ": ");
            internalMarks[i] = s.nextInt();
        }
```

```
    }
}
```

#Main.java

```
import java.util.Scanner;

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

        Externals[] students = new Externals[n];

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

        System.out.println("\nFinal Marks of Students:");
        for (int i = 0; i < n; i++) {
            System.out.println("\nStudent " + (i + 1) + ":");
            students[i].displayFinalMarks();
        }
    }
}
```

OUTPUT

```
Enter number of students: 2

Enter details for student 1:
Enter USN: 1bm23cs012
Enter Name: abhishek
Enter Semester: 3
Enter Internal Marks for 5 subjects:
Subject 1: 23
Subject 2: 65
Subject 3: 56
Subject 4: 54
Subject 5: 98
Enter SEE Marks for 5 subjects:
Subject 1: 56
Subject 2: 98
Subject 3: 63
Subject 4: 98
Subject 5: 75

Enter details for student 2:
Enter USN: aadit
Enter Name: aadity
Enter Semester: 3
Enter Internal Marks for 5 subjects:
Subject 1: 56
Subject 2: 85
Subject 3: 98
Subject 4: 75
Subject 5: 86
Enter SEE Marks for 5 subjects:
Subject 1: 98
Subject 2: 56
Subject 3: 82
Subject 4: 87
Subject 5: 88

Final Marks of Students:
```

```
Enter Semester: 3
Enter Internal Marks for 5 subjects:
Subject 1: 56
Subject 2: 85
Subject 3: 98
Subject 4: 75
Subject 5: 86
Enter SEE Marks for 5 subjects:
Subject 1: 98
Subject 2: 56
Subject 3: 82
Subject 4: 87
Subject 5: 88

Final Marks of Students:

Student 1:
USN: 1bm23cs012
Name: abhishek
Semester: 3
Final Marks for 5 subjects:
Subject 1: 79
Subject 2: 163
Subject 3: 119
Subject 4: 152
Subject 5: 173

Student 2:
USN: aadit
Name: aadity
Semester: 3
Final Marks for 5 subjects:
Subject 1: 154
Subject 2: 141
Subject 3: 180
Subject 4: 162
Subject 5: 174
```

Program: 06

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 has marks scored in five courses of the current Semester of the Student. This class has an array that stores the final marks of the student in all five courses. Student import from two packages in a file that declares

Program:

// Student.java

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.println("Enter USN:");
        USN = s.nextLine();
        System.out.println("Enter name:");
        name = s.nextLine();
        System.out.println("Enter Semester:");
        sem = s.nextInt();
    }

    public void displayStudentDetails() {
        System.out.println("USN: " + USN);

```
System.out.println("Name:" + name);
System.out.println("Semester:" + sem);
    }
}
```

// Internals.java

```
import java.util.Scanner;
public class Internals extends Student {
    protected int[] InternalMarks = new int[6];

    public void inputCIEMarks() {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter Internals Marks of " + subject);
        for (int i=0; i<5; i++) {
            System.out.println("subject" + (i+1) + ":");
            InternalMarks[i] = s.nextInt();
        }
    }
}
```

// External.java

```
import java.util.Scanner;
public class External extends Internals {
    private int[] seeMarks = new int[5];
    private int[] finalMarks = new int[5];

    public void inputSEEMarks() {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter SEE marks for 5 subjects:");
        for(int i=0; i<5; i++) {
            System.out.println("subject" + (i+1) + ":");
            seeMarks[i] = s.nextInt();
        }
    }
```

```
public void calculateFinalMarks() {
    for (int i = 0; i < 5; i++) {
        finalMarks[i] = InternalMarks[i] + SeeMarks[i];
    }
}
}
```

// Main.java

```
import java.util.Scanner;
class Main {
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter No° of Students:");
        int n = s.nextInt();
        External[] Student = new External[n];
        for (int i = 0; i < n; i++) {
            System.out.println("\n Enter details for Student " + (i+1));
            Student[i] = new External();
            Student[i].inputStudentDetails();
            Student[i].inputCIEMarks();
            Student[i].inputSEEMarks();
            Student[i].calculateFinalMarks();
        }
        System.out.println("\n Final marks of Students");
        for (int i = 0; i < n; i++) {
            System.out.println("\n Student" + (i+1) + ":");
        }
    }
}
```

Check again

OUTPUT

Enter details for student 1:
Enter USN: 1BM23CS309
Enter Name: Shankar.
Enter Semester: 3
Enter Internal marks of 5 subject
Subject 1: 89
Subject 2: 98
Subject 3: 85
Subject 4: 58
Subject 5: 78
Enter SEE marks for 5 subjects.
Subject 1: 78
Subject 2: 87
Subject 3: 98
Subject 4: 69
Subject 5: 96

Enter details for student 2:
Enter USN: 1BM23CS308
Enter Name: Suresh
Enter Semester: 1
Enter Internals marks for 5 subjects
Subject 1: 56
Subject 2: 65
Subject 3: 58
Subject 4: 95
Subject 5: 78

~~Final~~

Enter SEE Marks of Student:

Subject 1 : 58

Subject 2 : 68

Subject 3 : 78

Subject 4 : 59

Subject 5 : 48

Final marks of Students:

Student 1:

USN: 1bm23cs309

Name: Shankar

Semester: 2

Final marks for 5 Subjects

Subject 1 : 167

Subject 2 : 185

Subject 3 : 183

Subject 4 : 127

Subject 5 : 174

Student 2:

USN: 1bm23cs308

Name: Suresh

Semester: 2

Final marks for Subjects:

Subject 1 : 194

Subject 2 : 137

subject 3 : 136

Subject 4 : 154

Subject 5 : 126

LABORATORY PROGRAM – 07

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=father's age.

```
import java.util.Scanner;
class WrongAgeException extends Exception {
   public WrongAgeException(String message) {
      super(message);
   }
}

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

class Father {
   private int age;
   public Father(int age) throws WrongAgeException {
      if (age < 0) {
         throw new WrongAgeException("Wrong age");
      }
      this.age = age;
   }
   public int getAge() {
      return age;
   }
}

class Son extends Father {
   private int sonAge;
   public Son(int fatherAge, int sonAge) throws WrongAgeException,
SonAgeException {
      super(fatherAge);
      if (sonAge >= fatherAge) {
         throw new SonAgeException("Son's age cannot be greater than or equal to
father's age");
      }
      this.sonAge = sonAge;
   }
   public int getSonAge() {
```

```
        return sonAge;
    }
}
public class FatherSon{
    public static void main(String[] args) {
        while(true){
            Scanner sc = new Scanner(System.in);
            System.out.print("Enter Father's Age: ");
            int fatherAge = sc.nextInt();
            System.out.print("Enter Son's Age: ");
            int sonAge = sc.nextInt();
            try {
                Son son = new Son(fatherAge, sonAge);
                System.out.println("Accepted Succesfully");
            }
            catch (WrongAgeException e) {
                System.out.println(e.getMessage());
            }
            catch (SonAgeException e) {
                System.out.println(e.getMessage());
            }
            System.out.println("Would you like to re-enter details (Y/n)");
            String input = sc.next();
            if (input.equalsIgnoreCase("n")) {
                break;
            }
        }
    }
}
```

```
PS C:\Users\Abhishek\Desktop\1bm23cs309> javac FatherSon.java
PS C:\Users\Abhishek\Desktop\1bm23cs309> java FatherSon.java
Enter Father's Age: 45
Enter Son's Age: 21
Accepted Succesfully
Would you like to re-enter details (Y/n)
y
Enter Father's Age: 56
Enter Son's Age: 25
Accepted Succesfully
Would you like to re-enter details (Y/n)
n
PS C:\Users\Abhishek\Desktop\1bm23cs309>
```

LAB program of

Father son: lab program

?

```
import java.util.Scanner;
class wrongAge exception extends Exceptn {
    public wrong Exception (String message) {
        super (message);
    }
}
class father {
    private int age;
    public father (int age) throw WrongAge Exception {
        if (age < 0) {
            throw new wrong Age Exception ("wrong age
        }
        this.age = age;
    }
    public int get age () {
        return age;
    }
}
```

```
Class Son Extends Father {

  private int SonAge
  public Son (int FatherAge, int SonAge)
    throws WrongAge Exception SonAge Exception {

    super (FatherAge);
    if (SonAge >= FatherAge){
      throw new SonAge Exception (" Son age
        Cannot be greater than or Equals to
        Father age");
      this . SonAge = SonAge;
    }
    public int getsSonAge() {
      return SonAge;
    }
}

Public class FatherSon {
  public static void main (String @] args) {
      while (true) {

    Scanner sc = new Scanner (System.in);
    System.out.println ("enter Father Age");
      int FatherAge = sc. nextInt();
    System.out.println ("Enter Son Age");
      Int SonAge = sc. nextInt();
      try {
        Son son = new Son (FatherAge
```

```
System.out.println("Accepted Successfully");
}
catch (WrongAgeException e) {
    System.out.println(e.getMessage());
}
catch (SonAgeException e) {
    System.out.println(e.getMessage());
}
System.out.println("Would you like to be re
        enter (yn)");
String input = sc.next();
if (input.equalsIgnoreCase("yn")) {
    break;
}}}
```

Output:

```
Enter your Father's Age: 45
Enter Son's age: 12
Sons age cannot be less than zero.
Would like re enter details (yes/no)
yes
Enter Father's Age: 12
Enter Son's Age: 23
Wrong age
Would you like re-enter details (yes/no)
yes
Enter Father's Age: 23
Enter Sons Age: 45
Son's age cannot be greater than or Equal to
Father age.
```

LABORATORY PROGRAM – 08

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.

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

class SecondThread extends Thread {
   public void run() {
      while (true) {
         System.out.println("CSE");
         try {
            Thread.sleep(2000);
         } catch (InterruptedException e) {
            System.out.println(e);
         }
      }
   }
}

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

      FirstThread thread1 = new FirstThread();
      SecondThread thread2 = new SecondThread();


      thread1.start();
      thread2.start();
   }
```

}

```
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
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
CSE
BMS College of Engineering
```

8th program: LAB-8

```
import java.util.Scanner;

Class Thread2 {
public static void main (String args[]) {
Thread thread BMS = new Thread (new Display BMS ());
Thread thread CSE = new Thread (new DisplayCSE());
 thread BMS. Start();
 thread CSE. start();
 }
Class DisplayBMS implements Runnable {
 public voidrun (){
  try{
   kehile(true){
    System.out. println ("BMS College of Engineering");
   Thread. sleep (10000); }
   }
  catch (IntouptedException e) {
    System.out. println ("Intupted" + e. getMessage());
   }
  }
 }
Class display CSE Implements Runnable {
  public void run(){
  try {
   kehile(true) {
```

```
        System.out.println ("CSE");
        Thread.sleep(20000); }
        }
        catch (InterruptedException e) {
        System.out.println ("Interupted"+e.getMessage());
        }
    }
}
```

Output:

```
BMS college & Engineering
CSE
CSE
CSE
CSE
CSE
CSE
BMS college & Engineering
CSE
CSE
CSE
CSE
BMS college & Engineering
CSE
CSE
CSE
CSE
BMS college & Engineering
CSE
CSE
CSE
CSE
```

o/p not seen

BMS college of Engineering

CSE

CSE

CSE

CSE

CSE

BMS college of Engineering.

CSE

CSE

CSE

CSE

CSE

CSE

CSE

BMS college of Engineering.

LABORATORY PROGRAM - 9

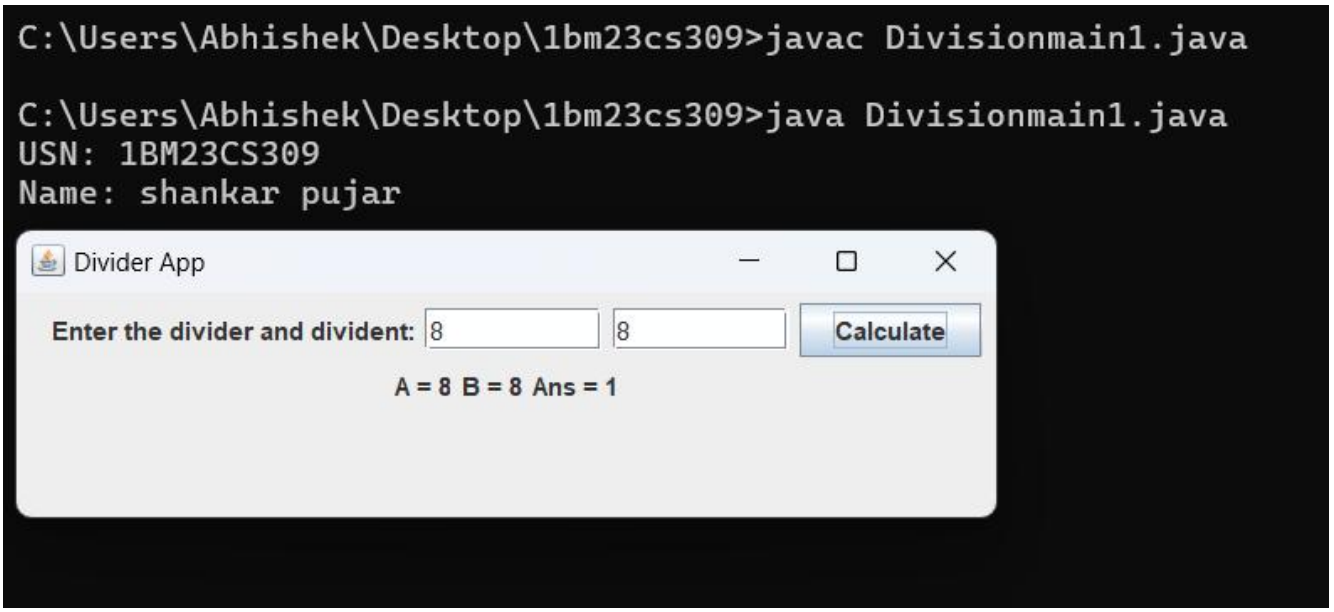
Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException.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

```
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(number 1);
add(num1); add(number 2);
add(num2); add(dResult)
```

```
;
add(outResul t);

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(result Num); 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(),outR
esult.getY()+outResult. getHeight()-8);
else g.drawString(out,1 00,200); flag=0;
}
```

OUTPUT



LAB=09

Write a program that creats a user interface to perform intiger division The user entres 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 intiger the program would throw an Airthmatic Exception Dispkay the Exception in a message dialog box,

```
import java.awt.*;
 import
java.awt.event.*;
public class Division main1 extends Fram-
implements Action listner
{
Textfield num1, num2; Button dresult; label out
result;
 String out: " "; double resultNum; int flag=0;
  public Division main1()
  {
   setLayout(new FlowLayout());
  dresult=new Button("Result");
  label number1=new Label1("Number1:",
     label1, RIGHT); Label1number2
```

```
=newlable:("number2:" 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);
num2.addActionListner(this); num2.addAction
    Listner(this); dresult.addActionListner
    (this); addwindow Listner(new windowadapter()

  public void windowClosing(windowevent we)
    {
    System.exist(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 ArithmaticException(); */ out=n1+" "+n2+" ";
resultNum=n1/n2; out+=String.valueOf(resultNum); repaint();
}
catch(NumberFormatException e1)
{
flag=1;
out="NumberFormatException!"+e1; repaint();
}
catch(ArithmaticException 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.getHight()-8);
else g.drawString(out, 100, 200); flag=0;
}
```

LABORATORY PROGRAM – 10

Demonstrate Inter process Communication and deadlock

```
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<05) {
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<05) {
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

```
C:\Users\Abhishek\Desktop\1bm23cs309>javac PCFixed.java

C:\Users\Abhishek\Desktop\1bm23cs309>java PCFixed.java
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

Intimate Producer

consumed:3
Put: 4

Intimate Consumer

Got: 4

Intimate Producer

consumed:4

C:\Users\Abhishek\Desktop\1bm23cs309>
```

LAB-10

Demonstrate Inter process communication and Deadlock,

```
Class Q {
int n;
boolean valueSet = false;
Synchronized int get() {
while (!valueSet)
try {
System.out.println("\n consumer waiting\n");
wait();
}
catch (InterruptedException e) {
System.out.println("InterruptedException 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\n");
wait();
```

```
}
catch (InterruptedException e) {
    System.out.println("Interrupted caught");
}
this.n = n;
valueSet = true;
System.out.println("put:"+n);
System.out.println("\n 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<65){
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<05){
int r= q.get();
System.out.println("i consumed:" +r);
i++;
}}
class PCFixed{
public static void main(String args[]){
 q.q = new q();
new producer(q);
 new producer(q);
 new consumer(q);
 System.out.println("press control-c to stop.");
}}
```

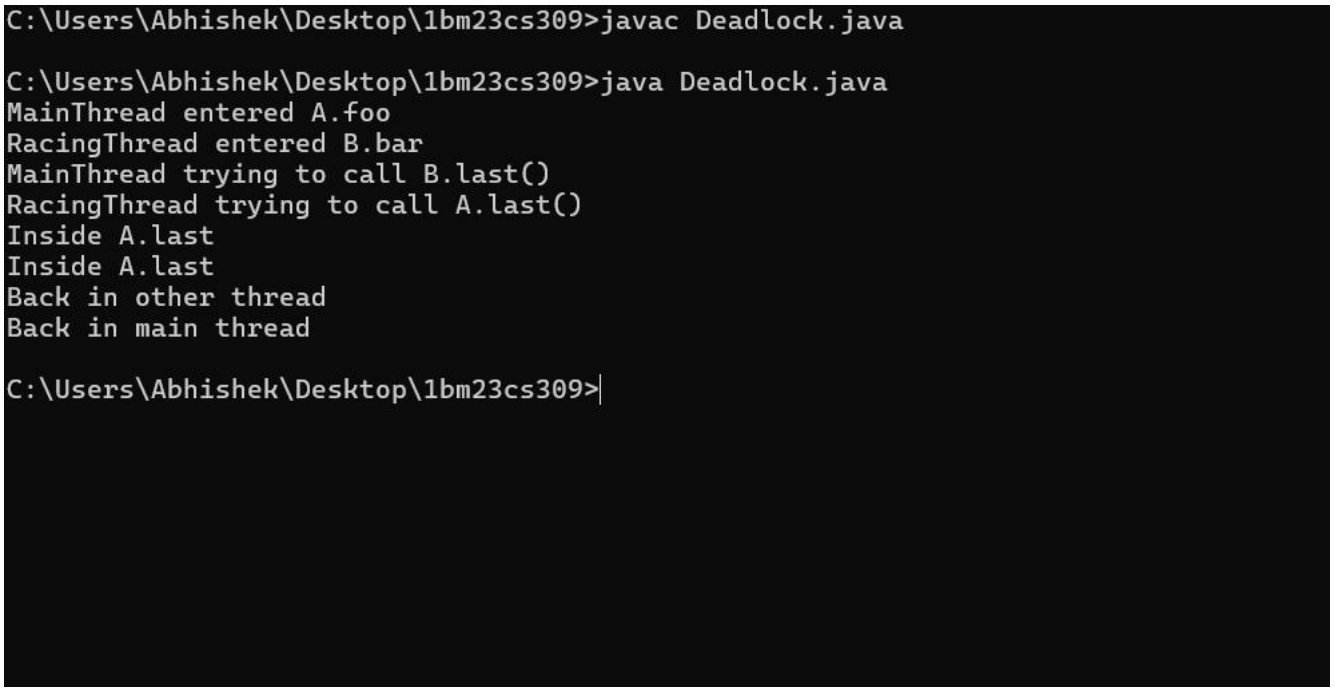
DEADLOCK PROGRAM :

```
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 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();

}

}
```

OUTPUT



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(name + " trying to call A.last()");
    a.last();
  }
System.out.println("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, "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();
   }
}
```

-: COMPLETE:-