

# **VISVESVARAYA TECHNOLOGICAL UNIVERSITY**

“JnanaSangama”, Belgaum -590014, Karnataka.



## **LAB REPORT on**

# **Object Oriented Java Programming (23CS3PCOOJ)**

*Submitted by*

**Vikas Shashi(1WA23CS043)**

*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 **Vikas Shashi(1WA23CS043)**, 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 Syed Akram Assistant Professor Department of CSE, BMSCE	Dr. Kavitha Sooda Professor & HOD Department of CSE, BMSCE
--	--

## Index

<b>Sl. No.</b>	<b>Date</b>	<b>Experiment Title</b>	<b>Page No.</b>
1	9/10/24	Quadratic Equation	4-5
2	16/10/24	Calculate SGPA	6-7
3	23/10/24	N book objects	8-9
4	23/10/24	Area of the given Shape	10-11
5	30/10/24	Bank Account	12-17
6	30/10/24	Packages	18-20
7	13/11/24	Exceptions	21-22
8	20/11/24	Threads	23-24
9	4/12/24	Interface to perform integer division	25-26
10	4/11/24	Inter process Communication and Deadlock	27-31

### **Program 1**

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

Code:

```
import java.io.*;
import java.util.Scanner;
import java.lang.Math;
class quadEq{
    public static void main(String args[]){
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter the co-efficients:");
        int a=sc.nextInt();
        int b=sc.nextInt();
        int c=sc.nextInt();
        if(a==0){
            System.out.println("Invalid input");
        }
        else{
            double d,r1,r2;
            d=b*b-4*a*c;
            if(d>0)
            {
                r1=(-b+Math.sqrt(d))/(2*a);
                r2=(-b-Math.sqrt(d))/(2*a);
                System.out.println("The real roots are r1="+r1+"and r2="+r2);
            }
            else if(d==0)
            {
                r1=-b/(2*a);
                System.out.println("The real and equal roots are r1=r2="+r1);
            }
            else
```

```

    {
    System.out.println("The roots are imaginary");
    r1=Math.sqrt(Math.abs(d))/(2*a);
    r2=-b/(2*a);
    System.out.println("The roots r1="+r2+"+i"+r1+"and r2="+r2+"-i"+r1);
    }
    }
    }
    }
}

```

output:

```

● vikas@Hosakere-Macbook-Pro java % java quadEq
Enter the co-efficients:
1 2 1
The real and equal roots are r1=r2=-1.0
● vikas@Hosakere-Macbook-Pro java % java quadEq
Enter the co-efficients:
2 1 2
The roots are imaginary
The roots r1=0.0+i0.9682458365518543and r2=0.0-i0.9682458365518543

```

## **Program 2**

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

Code:

```
import java.util.*;
import java.io.*;
class Student
{
    String name;
    String usn;
    int credits[]=new int[10];
    int grade[]=new int[10];
    int n;
    void accept(){
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter the student name:");
        name=sc.next();
        System.out.println("Enter the student usn:");
        usn=sc.next();
        System.out.println("Enter the no. of subjects:");
        n=sc.nextInt();
        for(int i=0;i<n;i++)
        {
            System.out.println("Enter the subject grade points:");
            credits[i]=sc.nextInt();
            System.out.println("Enter the subject grade:");
            grade[i]=sc.nextInt();
        }
    }
    void display(){
        System.out.println("Student name:"+name);
        System.out.println("Student usn:"+usn);
        for(int i=0;i<n;i++)
        {
            System.out.println("subject gp:"+credits[i]);
            System.out.println("subject marks"+grade[i]);
        }
    }
}
class grade{
    public static void main(String args[]){
        double grade,sum=0,tsum=0;
        double total[]=new double[10];
        Student st=new Student();
        st.accept();
        st.display();
    }
}
```

```

for(int i=0;i<st.n;i++)
{
total[i]=st.credits[i]*st.grade[i];
sum+=total[i];
tsum+=st.credits[i]*10;
}
grade=(sum/tsum)*10;
System.out.println("The student grade is:"+grade);

}
}

```

output:

```

vikas@Hosakere-Macbook-Pro java % java grade
Enter the student name:
Vikas
Enter the student usn:
1WA23CS043
Enter the no. of subjects:
4
Enter the subject grade points:
4
Enter the subject grade:
90
Enter the subject grade points:
3
Enter the subject grade:
80
Enter the subject grade points:
1
Enter the subject grade:
89
Enter the subject grade points:
1
Enter the subject grade:
80
Student name:Vikas
Student usn:1WA23CS043
subject gp:4
subject marks90
subject gp:3
subject marks80
subject gp:1
subject marks89
subject gp:1
subject marks80
The student grade is:85.44444444444444

```

### **Program 3**

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

code:

```
import java.io.*;
import java.util.*;
class Book{
String name;
String author;
int num_pages;
double price;
Book()
{
}
void acceptDetails(String name,String author,int num_pages,double price)
{
this.name=name;
this.author=author;
this.num_pages=num_pages;
this.price=price;
}
void displayDetails(int i)
{
System.out.println("Book "+(i+1)+" name:"+name);
System.out.println("Book "+(i+1)+" author:"+author);
System.out.println("Book "+(i+1)+" pages:"+num_pages);
System.out.println("Book "+(i+1)+" price:"+price);
}
@Override
public String toString(){

return "Book Name:"+name+" Author name:"+author+" Number of Pages:"+num_pages+"
price:"+price;
}
}
class Books{
public static void main(String args[]){
int n,i;
Scanner sc=new Scanner(System.in);
System.out.println("Enter the number of books:");
n=sc.nextInt();
Book b[]=new Book[n];
for(i=0;i<n;i++)
{
```



```

b[i]=new Book();
System.out.println("Enter the name of the book:"+i+":");
String name=sc.next();
System.out.println("Enter the name of the author:"+i+":");
String author=sc.next();
System.out.println("Enter the number of pages of:"+i+":");
int num_pages=sc.nextInt();
System.out.println("Enter the price of book:"+i+":");
double price=sc.nextDouble();
b[i].acceptDetails(name,author,num_pages,price);
}
System.out.println("The details of books are:");
for(i=0;i<n;i++)
{
b[i].displayDetails(i);
}
}
}
}

```

Output:

```

vikas@Hosakere-Macbook-Pro java % java Books
Enter the number of books:
2
Enter the name of the book:1:
ABC
Enter the name of the author:1:
def
Enter the number of pages of:1:
400
Enter the price of book:1:
250
Enter the name of the book:2:
ghi
Enter the name of the author:2:
jkl
Enter the number of pages of:2:
309
Enter the price of book:2:
450
The details of books are:
Book 1 name:ABC
Book 1 author:def
Book 1 pages:400
Book 1 price:250.0
Book 2 name:ghi
Book 2 author:jkl
Book 2 pages:309
Book 2 price:450.0

```

## **Program 4**

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

Code:

```
import java.io.*;
import java.util.*;
abstract class Shape
{
    int a,b,c;
    Shape(int a,int b)
    {
        this.a=a;
        this.b=b;
    }
    Shape(int c)
    {
        this.c=c;
    }
    abstract void printArea();
}
class Rectangle extends Shape
{
    Rectangle(int a,int b)
    {
        super(a,b);
    }
    double i;
    void printArea()
    {
        i=a*b;
        System.out.println("Inside Area for Rectangle:"+i);
    }
}
class Triangle extends Shape
{
    Triangle(int a,int b)
    {
        super(a,b);
```

```

    }
    double j;
    void printArea()
    {
    j=(a*b)/2;
    System.out.println("Inside Area for Triangle:"+j);
    }
    }
    class Circle extends Shape
    {
    Circle(int c)
    {
    super(c);
    }
    double k;
    void printArea()
    {
    k=(3.14*c*c);
    System.out.println("Inside Area for Circle:"+k);
    }
    }
    class Main
    {
    public static void main(String args[])
    {
    Rectangle r=new Rectangle(3,4);
    Triangle t=new Triangle(4,6);
    Circle c=new Circle(7);
    r.printArea();
    t.printArea();
    c.printArea();
    }
    }

```

Output:

```

vikas@Hosakere-Macbook-Pro java % java Main
Inside Area for Rectangle:12.0
Inside Area for Triangle:12.0
Inside Area for Circle:153.86

```

## **Program 5**

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

- 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.

Code:

```
import java.io.*;
import java.util.*;
abstract class Account
{
    String cusName, accNo;
    double balance;
    Account(String name,String num)
    {
        this.cusName=name;
        this.accNo=num;
        this.balance=0.0;
    }
    abstract void deposit(double num);
    abstract void withdraw(double num);
    abstract void display();
}
class SavAcc extends Account
{
    double rate;
    SavAcc(String cusName,String accNo,double rate)
    {
        super(cusName,accNo);
        this.rate=rate;
    }
    @Override
```

```

void deposit(double num)
{
    balance+=num;
    System.out.println("The current balance after deposit is:"+balance);
}
void withdraw(double num)
{
    if(num>balance)
    {
        System.out.println("Not enough balance");
    }
    else
    {
        balance-=num;
        System.out.println("Withdrew amount="+num);
    }
}
void calcInterest()
{
    double interest=balance*(rate/100);
    balance+=interest;
    System.out.println("The interest deposited is:"+interest);
}
void display()
{
    System.out.println("Savings account balance:"+balance);
}
}
class CurAcc extends Account
{
    double minBal,serviceCharge;
    CurAcc(String cusName,String accNo,double minBal,double serviceCharge)
    {
        super(cusName,accNo);
        this.minBal=minBal;
        this.serviceCharge=serviceCharge;
    }
    @Override
    void deposit(double num)
    {
        balance+=num;

```

```

System.out.println("The current balance after deposit is:"+balance);
}
void withdraw(double num)
{
if(num>balance)
{
System.out.println("Not enough balance");
}
else
{
balance-=num;
System.out.println("Withdrew amount="+num);
checkMinBalance();
}
}
void checkMinBalance()
{
if (balance<minBal)
{
System.out.println("Lower than min balance.");
balance-=serviceCharge;
System.out.println("Service charge of "+ serviceCharge+" applied. New
Balance: "+balance);
}
}
void display()
{
System.out.println("Savings account balance:"+balance);
}
}
class Applicant
{
public static void main(String args[])
{
int choice,n;
String name,accno;
double rate=5.0,minbal=500.0,sercha=25.0;
double amount;
Scanner sc=new Scanner(System.in);
System.out.println("Enter name:");
name=sc.nextLine();

```

```

System.out.println("Enter account no.:");
accno=sc.nextLine();
System.out.println("Enter[1.Savings,2.Current]");
n=sc.nextInt();
if(n==1)
{
SavAcc a=new SavAcc(name,accno,rate);
System.out.println("Savings Account created");
do
{
System.out.println("Enter[1.Deposit,2.Withdraw,3.GetBalance,4.ADD Interest,5.Exit]");
choice=sc.nextInt();
switch(choice)
{
case 1:
System.out.println("Enter the amount to be inserted:");
amount=sc.nextInt();
a.deposit(amount);
break;
case 2:
System.out.println("Enter the amount to be withdrawn:");
amount=sc.nextInt();
a.withdraw(amount);
break;
case 3:
a.display();
break;
case 4:
a.calcInterest();
break;
case 5:
System.out.println("Exiting");
break;
default:
System.out.println("Enter proper choice.");
break;
}
}while(choice!=5);
}
else if(n==2)
{

```

```

CurAcc b=new CurAcc(name,accno,minbal,sercha);
System.out.println("Current account created");
do
{
System.out.println("Enter[1.Deposit,2.Withdraw,3.GetBalance,4.Exit]");
choice=sc.nextInt();
switch(choice)
{
case 1:
System.out.println("Enter the amount to be inserted:");
amount=sc.nextInt();
b.deposit(amount);
break;
case 2:
System.out.println("Enter the amount to be withdrawn:");
amount=sc.nextInt();
b.withdraw(amount);
break;
case 3:
b.display();
break;
case 4:
System.out.println("Exiting");
break;
default:
System.out.println("Enter proper choice.");
break;
}
}while(choice!=4);
}
else
{
System.out.println("Invalid account type");
return;
}
}
}
}

```

Output:



```

vikas@Hosakere-Macbook-Pro java % java Applicant
Enter name:
Vikas
Enter account no.:
1
Enter[1.Savings,2.Current]
1
Savings Account created
Enter[1.Deposit,2.Withdraw,3.GetBalance,4.]
1
Enter the amount to be inserted:
200
The current balance after deposit is:200.0
Enter[1.Deposit,2.Withdraw,3.GetBalance,4.]
2
Enter the amount to be withdrawn:
300
Not enough balance
Enter[1.Deposit,2.Withdraw,3.GetBalance,4.]
3
Savings account balance:200.0
Enter[1.Deposit,2.Withdraw,3.GetBalance,4.]
4
The interest deposited is:10.0
Enter[1.Deposit,2.Withdraw,3.GetBalance,4.]
5
Exiting

```

```

vikas@Hosakere-Macbook-Pro java % java Applicant
Enter name:
Vikas
Enter account no.:
2
Enter[1.Savings,2.Current]
2
Current account created
Enter[1.Deposit,2.Withdraw,3.GetBalance,4.Exit]
1
Enter the amount to be inserted:
400
The current balance after deposit is:400.0
Enter[1.Deposit,2.Withdraw,3.GetBalance,4.Exit]
2
Enter the amount to be withdrawn:
3000
Not enough balance
Enter[1.Deposit,2.Withdraw,3.GetBalance,4.Exit]
3
Savings account balance:400.0
Enter[1.Deposit,2.Withdraw,3.GetBalance,4.Exit]
4
Exiting

```

## **Program 6**

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

Code:

```
//Internals.java
```

```
package cie;
```

```
public class Internals {  
    public int[] internalMarks = new int[5];  
    public Internals(int[] marks) {  
        for (int i = 0; i < 5; i++) {  
            internalMarks[i] = marks[i];  
        }  
    }  
}
```

```
//Student.java
```

```
package cie;
```

```
public class Student {  
    public String usn;  
    public String name;  
    public int sem;  
    public Student(String usn, String name, int sem) {  
        this.usn = usn;  
        this.name = name;  
        this.sem = sem;  
    }  
}
```

```
//Externals.java
```

```
package see;
```

```
import cie.Student;
```

```
public class External extends Student  
{  
    public int[] seeMarks = new int[5];  
    public External(String usn, String name, int sem, int seeMarks[])  
    {  
        super(usn, name, sem);  
        for (int i = 0; i < 5; i++)
```

```

    {
        this.seeMarks[i] = seeMarks[i];
    }
}
}
//Main.java
import cie.Internals;
import see.External;
import java.util.Scanner;
public class Main
{
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);

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

        for (int i=0;i<n;i++)
        {
            System.out.println("\nEnter details for the "+(i + 1)+" Student " + ":");

            sc.nextLine();
            System.out.print("USN: ");
            String usn = sc.nextLine();
            System.out.print("Name: ");
            String name = sc.nextLine();
            System.out.print("Semester: ");
            int sem = sc.nextInt();

            System.out.println("Enter internal marks for 5 courses (out of 50): ");
            int internalMarks[] = new int[5];
            for (int j=0;j<5;j++)
            {
                System.out.print("Course " + (j + 1) + " internal marks: ");
                internalMarks[j] = sc.nextInt();
            }
            Internals internal = new Internals(internalMarks);

            System.out.println("Enter SEE marks for 5 courses (out of 100): ");
            int seeMarks[] = new int[5];

```

```

    for (int j=0;j<5;j++)
    {
        System.out.print("Course "+(j + 1)+" SEE marks: ");
        seeMarks[j] = sc.nextInt();
    }
    External external = new External(usn, name, sem, seeMarks);

    System.out.println("\nFinal Marks for " + external.name + " : " + external.usn + ":for:");
    System.out.println("Semester: " + external.sem);
    System.out.println("Total Marks:");
    for (int j=0;j<5;j++)
    {
        System.out.println("Course " + (j + 1) + ": " + (double)(internal.internalMarks[j] +
        (external.seeMarks[j]/2)));
    }
    }
}
}
}

```

Output:

```

vikas@Hosakere-Macbook-Pro 056 % javac cie/*.java
vikas@Hosakere-Macbook-Pro 056 % javac see/*.java
vikas@Hosakere-Macbook-Pro 056 % javac Main.java
vikas@Hosakere-Macbook-Pro 056 % java Main
Enter the number of students: 1

Enter details for the 1 Student :
USN: 1WA23CS043
Name: Vikas
Semester: 3
Enter internal marks for 5 courses (out of 50):
Course 1 internal marks: 35
Course 2 internal marks: 37
Course 3 internal marks: 38
Course 4 internal marks: 36
Course 5 internal marks: 39
Enter SEE marks for 5 courses (out of 100):
Course 1 SEE marks: 89
Course 2 SEE marks: 90
Course 3 SEE marks: 91
Course 4 SEE marks: 92
Course 5 SEE marks: 93

Final Marks for Vikas :1WA23CS043:for:
Semester: 3
Total Marks:
Course 1: 79.0
Course 2: 82.0
Course 3: 83.0
Course 4: 82.0
Course 5: 85.0

```

## **Program 7**

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

Code:

```
import java.io.*;
class WrongAge extends Exception {
    public WrongAge(String txt) {
        super(txt);
    }
}
class Father {
    private int age;
    public Father(int age) throws WrongAge {
        if (age < 0) {
            throw new WrongAge("Age cannot be negative.");
        }
        this.age = age;
    }
    public int getAge() {
        return age;
    }
}
class Son extends Father {
    private int sonAge;
    public Son(int fatherAge, int sonAge) throws WrongAge {
        super(fatherAge);
        if (sonAge < 0) {
            throw new WrongAge("Son's age cannot be negative.");
        }
        if (sonAge >= fatherAge) {
            throw new WrongAge("Son's age must be less than father's age.");
        }
        this.sonAge = sonAge;
    }
    public int getSonAge() {
        return sonAge;
    }
}
```

```

}
class Main {
    public static void main(String[] args) {
        try {
            Father father = new Father(40);
            Son son = new Son(40, 20);
            System.out.println("Father's Age: " + father.getAge());
            System.out.println("Son's Age: " + son.getSonAge());
        } catch (WrongAge e) {
            System.out.println(e);
        }
        try {
            Son son2 = new Son(-5, 10);
        } catch (WrongAge e) {
            System.out.println(e);
        }
        try {
            Son son3 = new Son(30, 30);
        } catch (WrongAge e) {
            System.out.println(e);
        }
        try {
            Son son4 = new Son(30, -5);
        } catch (WrongAge e) {
            System.out.println(e);
        }
    }
}

```

Output:

```

vikas@Hosakere-Macbook-Pro java % java Main
Father's Age: 40
Son's Age: 20
WrongAge: Age cannot be negative.
WrongAge: Son's age must be less than father's age.
WrongAge: Son's age cannot be negative.

```

## **Program 8**

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

Code:

```
class CollegeThread extends Thread {
    public void run() {
        try {
            for (int i = 0; i < 5; i++) {
                System.out.println("BMS College of Engineering");
                Thread.sleep(10000);
            }
        } catch (InterruptedException e) {
            System.out.println("CollegeThread interrupted.");
        }
    }
}

class CSEThread extends Thread {
    public void run() {
        try {
            for (int i = 0; i < 25; i++) {
                System.out.println("CSE");
                Thread.sleep(2000);
            }
        } catch (InterruptedException e) {
            System.out.println("CSEThread interrupted.");
        }
    }
}

public class ThreadExample {
    public static void main(String[] args) {
        CollegeThread collegeThread = new CollegeThread();
        CSEThread cseThread = new CSEThread();

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

Output:

```
vikas@Hosakere-Macbook-Pro java % java ThreadExample
BMS College of Engineering
CSE
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
```



## **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 Number Format Exception. If Num2 were Zero, the program would throw an Arithmetic Exception Display the exception in a message dialog box.

Code:

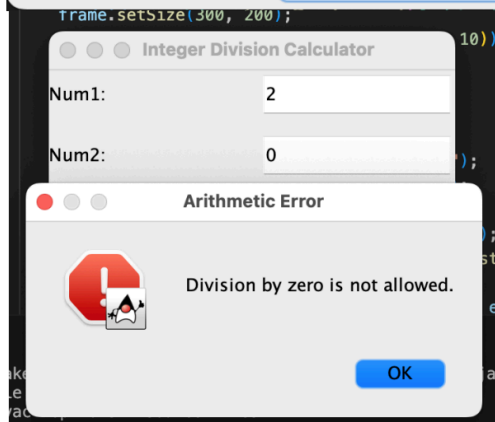
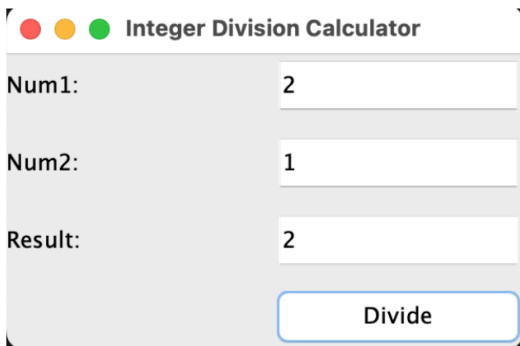
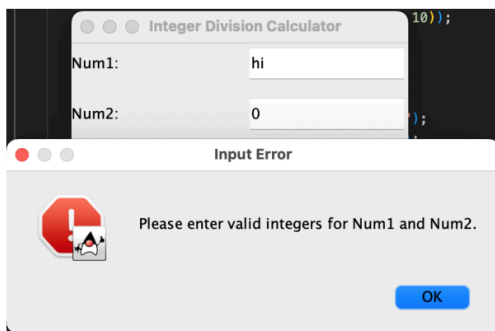
```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
class DivisionCalculator {
public static void main(String[] args) {
JFrame frame = new JFrame("Integer Division Calculator");
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.setSize(300, 200);
frame.setLayout(new GridLayout(4, 2, 10, 10));
JLabel num1Label = new JLabel("Num1:");
JTextField num1Field = new JTextField();
JLabel num2Label = new JLabel("Num2:");
JTextField num2Field = new JTextField();
JLabel resultLabel = new JLabel("Result:");
JTextField resultField = new JTextField();
resultField.setEditable(false);
JButton divideButton = new JButton("Divide");
divideButton.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
try {
int num1 = Integer.parseInt(num1Field.getText());
int num2 = Integer.parseInt(num2Field.getText());
int result = num1 / num2;
resultField.setText(String.valueOf(result));
} catch (NumberFormatException ex) {
JOptionPane.showMessageDialog(frame, "Please enter valid integers
for Num1 and Num2.", "Input Error", JOptionPane.ERROR_MESSAGE);
} catch (ArithmeticException ex) {
JOptionPane.showMessageDialog(frame, "Division by zero is not
allowed.", "Arithmetic Error", JOptionPane.ERROR_MESSAGE);
}
}
});
frame.add(num1Label);
frame.add(num1Field);
```

```

frame.add(num2Label);
frame.add(num2Field);
frame.add(resultLabel);
frame.add(resultField);
frame.add(new JLabel());
frame.add(divideButton);
frame.setVisible(true);
}
}

```

Output:



### **Program 10**

Demonstrate Inter process Communication and deadlock

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:



```

vikas@Hosakere-Macbook-Pro java % java PCFixed
Press Control-C to stop.
Put: 0
Got: 0
Put: 1
Got: 1
Put: 2
Got: 2
Put: 3
Got: 3
Put: 4
Got: 4
Put: 5
Got: 5
Put: 6
Got: 6
Put: 7
Got: 7
Put: 8
Got: 8
Put: 9
Got: 9
Put: 10
Got: 10

```

//Deadlock

```

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

```

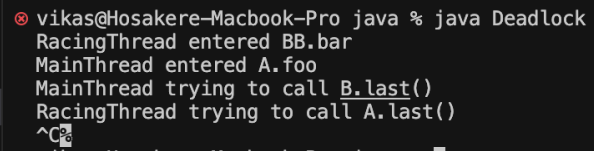
```

        Thread.sleep(1000);
    } catch (Exception e) {
        System.out.println("A Interrupted");
    }
    System.out.println(name + " trying to call B.last()");
    b.last();
}
synchronized void last() {
    System.out.println("Inside A.last");
}
}
class BB {
    synchronized void bar(AA a) {
        String name = Thread.currentThread().getName();
        System.out.println(name + " entered BB.bar");
        try {
            Thread.sleep(1000);
        } catch (Exception e) {
            System.out.println("B Interrupted");
        }
        System.out.println(name + " trying to call A.last()");
        a.last();
    }
    synchronized void last() {
        System.out.println("Inside A.last");
    }
}
class Deadlock implements Runnable {
    AA a = new AA();
    BB b = new BB();
    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:

A terminal window with a dark background and light-colored text. It shows the execution of a Java program named 'Deadlock'. The output indicates that two threads, 'RacingThread' and 'MainThread', have entered different methods ('BB.bar' and 'A.foo' respectively). Then, 'MainThread' attempts to call 'B.last()' while 'RacingThread' is in 'A.foo', and 'RacingThread' attempts to call 'A.last()' while 'MainThread' is in 'B.last()'. This results in a deadlock. The prompt '^C' is visible at the bottom, indicating the user pressed Ctrl-C to stop the program.

```
vikas@Hosakere-Macbook-Pro java % java Deadlock  
RacingThread entered BB.bar  
MainThread entered A.foo  
MainThread trying to call B.last()  
RacingThread trying to call A.last()  
^C
```