VISVESVARAYA TECHNOLOGICAL UNIVERSITY Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ) *Submitted by*

Rohit Ramchandra Gandhi (2023BMS02599)

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
Dec-2023 to Mar-2024

12/12/2023 WEEK-1

Program 1:- Quadratic

Develop a Java program on quadratic

```
import java.util.Scanner;
class Quadratic
{
    int a,b,c;
    double r1,r2,d;
    void get()
    {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter the coefficient of
        a,b,c");
        a = s.nextInt();
        b = s.nextInt();
        c = s.nextInt();
    }
void compute()
{
    while(a==0)
    {
        System.out.println("Not a d equation");
        System.out.println("Enter a non zero value for
                                    a:");
```

```
Scanner s = new Scanner (System.in);
a = s.nextInt();
{
d = b * b - 4 * a * c;
if (d == 0)
{
s1 = (-b)/(2*a);
System.out.println("roots are real and equal");
System.out.println("root 1 = root 2 = " + s1);
}
s1 = ((-b) + (Math.sqrt(d)))/(double)(2*a)
s2 = ((-b) - (Math.sqrt(d)))/(double)(2*a)
System.out.println("roots are real & distinct")
System.out.println("roots = " s1 + "root 2 = " + s2);
}
else if (d<0)
{
System.out.println("roots are imaginary");
s1 = (-b)/2(*a)
s2 = Math.sqrt(-d)/(2*a)
System.out.println("root 1 = " + s1 + "+i" + s2);
System.out.print("root 1 = " + s1 "-i" + s2);
}
}
}
```

```
class Quadratic Run
{
    public static void main(String args[])
    {
        quadratic q = new Quadratic();
        q.get();
        q.compute();
    }
}
```

output :- My name is Rohit Gandhi
My USN is 2023BMS02599
Enter the coefficient of a, b, c
1
5
3
Roots are real and distinct
Root1 = -0.6972243622680054 and Root 2 = -4.302775637731995

output 2 :- My name is Rohit Gandhi
My USN is 2023BMS02599
Enter the coefficient of a, b, c :
1, 0, -1
Roots are real and distinct
Roots I = 1.0 Root 2 = -1.0

Program 6:-

Factorial :-

```
class factorial {
public static void main (String args[])
{
  int fac = 1;
System.out.println("Enter a no");
Scanner sc = new Scanner(System.in);
int n = sc.nextInt();
for(int i=1 ;i<=n; i++) {
    fact = fac * i;
}
System.out.println("The factorial :" + fac);
```

12.12.22

2/12/23

Lab program 2 [Week 2]

Student :- 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;
import Java.io.*;

class Student {
    private String USN;
    private String name;
    private int[] credits;
    private int[] marks;

    // Constructor
    public Student(String USN, String name,
                   int numSubject) {
        this.USN = USN;
        this.name = name;
        this.credits = new int[numSubjects];
```

```
// Method to accept details
public void acceptDetails() {
    Scanner scanner = new Scanner(System.in);

    System.out.println("Enter details for student " + name + " (USN: " + USN + "):");
    for (int i = 0; i < credits.length; i++)
    {
        System.out.print("Enter credits for Subject " + (i+1) + ": ");
        credit[i] = scanner.nextInt();

        System.out.println("Enter marks for subject " + (i+1) + ": ");
        marks[i] = scanner.nextInt();
    }
}

// Method to display details
public void displayDetails() {
    System.out.print("Details for student: " + name + "(USN: " + USN + "):");
    for (int i = 0; i < credits.length; i++)
    {
        System.out.println("Subject " + (i+1) +
```

```
"- credits : " + credits[i] + ". Marks : " + marks[i];
    }
}

//Method to calculate CGPA
public double calculateCGPA() {
    double totalcredits = 0;
    double totalgrade = 0;

for (int i = 0; i < credits.length ; i++) {
    total credits += credits[i];
    total Grade += calculategrade (marks[i] *
        credits[i];
    }
    return totalgrade / total credits ;

public double calculategrade (int marks)
{
    if (marks >= 90) {
        return 10.0;
}
} else if (marks >= 80) {
    return 90.0%;
} else if (marks >= 70) {
    return 8.0;
```

```
} else if (marks >=60) {
        setum 7.0;
} else if (marks >=50) {
        setum 6.0;
} else {
        setum 0.0;
    }
  }
}

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

    System.out.print ("Enter the no of subjects:");
    int noSubjects = scanner.nextInt ();

    System.out.print ("Enter the USN:");
    String USN = scanner.next ();

    System.out.print ("Enter the name");
    String name = scanner.next ();

    Student student = new Student (USN, name
        numberSubject);
```

```
        student.acceptdetails();

        //Display details & SGPA
        student.displaydetails();
        System.out.println("SGPA"+student.
            calculateSGPA());
    }
}
```

Output :-

```
Enter the No of subjects :8
Enter the USN :2023bms02599
Enter the name: Rohit Gandhi
Enter details of student :-

Enter credits for subject 1 : 3
Enter marks for subject 1 : 68

Enter credits for subject 2 : 3
Enter marks for subject 2 : 99

Enter credits for subject 3: 2
Enter marks for subject 3: 78
```

Enter credits for subject 4: 3
Enter marks for subject 4: 58

Enter credits for subject 5: 3
Enter marks for subject 5: 58

Enter credits for subject 6: 1
Enter marks for subject 6: 68

Enter credits for subject 7: 3
Enter marks for subject 7: 68

Enter credits for subject 8: 1
Enter marks for subject 8: 88

SGPA = 7.368421052631579

19.12.24

5/12/23

WEEK - 3

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. Includes 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 {
    String name;
    String author;
    int price;
    int numpages;

    public Book (String name, string author,
                 int price, int numpages)
    {
        this.name = name;
```

```
        this.author = author;
        this.price = price;
        this.numpages = numpages;
    }

    public string toString()
    {
        String bookdetails = "Book name:" + this.name
            + "author name:" + this.author + "\n"
            + "price:" + this.price + "\n"
            + Number of pages :" + this.numpages + "\n";
        return bookdetails;
    }
}
public class Bookstore {

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

        System.out.println ("Enter the no of books:");
        int n = Scanner.nextInt();

        Book[] books = new Book[n];
```

```
for (int i=0; i<n; i++) {
System.out.print("Enter name of book:");
Scanner.nextLine();
String name = Scanner.nextLine();

System.out.print("Enter author of the book:");
String author = Scanner.nextLine();

System.out.print("Enter the price of book:");
int price = scanner.nextInt();

System.out.print("Enter the number of
                  pages of book:");
int numpages = scanner.nextInt();

books[i] = new book(name, author, price,
           numpages);
}
System.out.println("\n Book details");
for (int i=0; i<n; i++) {
    System.out.println("Book" + (i+1) +
    ":\n" + book[i]);
}
}
```

Input :-

Enter no of books : 1
Enter name of book : The Great man
Enter author of book : Rohit Gandhi
Enter the price of book : 1500
Enter the no of pages of book : 180

Output :-

Book Details :
Book 1 :
Book name : The Great man
Author name : Rohit Gandhi
Price : 1500
No of pages : 180

+ Name :- Rohit. Gandhi
USN :- 2023BTS02599

Program to return Objects

```
class Test {
    int a;
    Test (int i) {
        a = i;
    }
    Test incrbyTen() {
        Test temp = new Test (a+10);
        return temp;
    }
}
class RetOb {
    public static void main (String args[]){
        Test ob1 = new Test (2);
        Test ob2;
        ob2 = ob1.incrbyTen();
        System.out.println("ob1.a: "+ob1.a);
        ob2 = ob2.incrbyTen();
        System.out.println("ob2.a after second increase: ");
        System.out.println(ob2.a);
    }
}
```

1/24

WEEK-4

+ Develop a Java program to create an abstract class named shape that contains two integers and an empty method named printbrea(). 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 printbrea() that prints the area of the given shape.

```
import java.util.Scanner;

abstract class shape {
    protected int side1;
    protected int side2;

    public shape(int side1, int side2) {
        this.side1 = side1;
        this.side2 = side2;
    }
    public abstract void printbrea();
}
```

```
class Rectangle extends Shape {
    public Rectangle(int length, int width) {
        super(length, width);
    }
    public void printArea() {
        int area = side1 * side2;
        System.out.println("Rectangle Area: " + area);
    }
}
class Triangle extends Shape {
    public Triangle(int base, int height) {
        super(base, height);
    }
    public void printArea() {
        double area = 0.5 * side1 * side2;
        System.out.println("Triangle Area: " + area);
    }
}
class Circle extends Shape {
    public Circle(int radius) {
        super(radius, 0);
    }
    public void printArea() {
        double area = Math.PI * side1 * side1;
        System.out.println("Circle Area: " + area);
```

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

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

        System.out.println("Enter base and Height for
                                Triangle :");
        int tribase = scanner.nextInt();
        int triheight = scanner.nextInt();
        Triangle Triangle = new Triangle (tribase, triHeight);
        tAngle.printArea();

        System.out.println("Enter radius for circle :");
        int circleRadius = scanner.nextInt();
        Circle circle = new Circle (circleRadius);
        circle.printArea();
```

```
        scanner.close();
    }
}
```

Output:-

Enter length and width for rectangle:
7
4
Rectangle Area = 28.

Enter base and height for Triangle:
4
8
Triangle Area : 16.0.

Enter radius for circle :
9
Circle Area : 254.469004907323.

Name :- Rohit. Gandhi
USN :- 2023BKS02599

1/1/24

10/23

Lab program 5

+ Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called savings account. The account provides compound interest and withdrawal facilities but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed.

+ Accept deposit from customer & update the balance.
+ Display the balance.
+ Compute and deposite interest
+ Permit withdrawal and update the balance
+ Check for the minimum balance, impose penalty if necessary & update the balance.

```
import java.util.Scanner;

abstract class Account {
    String customer_name;
    long account_number;
    String accountType;
    double balance;
```

```
public Account(string customersname, long accountnum
  string accounttype, double balance)
{
  this.customersname = customers_name;
  this.accountname = accountNumber;
  this.accounttype = accounttype;
  this.balance = balance;
}
public void displayBalance() {
    System.out.println("Account balance: $" + balance);
}
//public method for withdrawal
public abstract void withdraw(double amt);
{
  class CurrAcct extends Account {
    double minimumbalance;
    double servicecharge;

    public CurrAcct(string customers name, long
    account Number, double balance) {
    super(customersname, accountname, "CurrentAccount",
      balance);
    this.minimumbalance = 1000;
    this.max
                    = 50;
```

```
public void withdraw(double amount) {
    if (balance - amount >= minimumbalance) {
        balance -= amount;
    System.out.print("withdraw successfull")
    }

    else {
        System.out.println("Insufficient funds.
    }

}
}

class Savact extends Bacount {
    double interrate;

public void withdraw(double amnt) {
    if (balance - amount >= 0)
        balance -= amount;
    System.out.println("withdraw success")
    else {
        System.out.println("Insufficient balance");

public class Bank1 {
    public static void main(String[] args) {
        Scanner Scanr = new Scanner(System.in)
```

```
System.out.println("name");
String name = scanner.nextLine();

System.out.println("no");
String no = scanner.nextLong();

System.out.println("Enter the balance");
double initialBalance = scanner.nextDouble();

Account account;

int choice;
do{
    System.out.println("1. Deposit");
    System.out.println("2 Display balance");
    System.out.println("3 Deposit interest");
    System.out.println("4 Withdraw");
    System.out.println("5 Exit");
    choice = Scanner.nextInt();

switch(choice){
case 1:
    System.out.print("Enter deposit amount:");
    double depositeamount = scanner.nextDouble();
}
else{
```

```
System.out.println("This option is applicable for
    savings only");
}
break;
case 2:
account.displayBalance();
break;
case 3:
if(account instanceof Savings) {
    ((Savings)account).depositInterest();
} else {
System.out.println("This is optional");
}
break;
case 4:
System.out.println("Enter withdrawal amount:");
double withdrawalAmount = scanner.nextDouble();
break;
case 5:
System.out.println("Exiting program. Goodbye!");
break;
}
} while (choice != 6);
scanner.close();
}
```

Output:

Enter customer name: Rohit
Enter account no: 1234768
Enter initial balance: 500

1: Deposit
2: Display Balance
3: Withdraw
4: Exit

Enter choose: 1
Enter deposit amount: 2500

Enter choise: 2
Account Balance: 3000.0

Enter choise: 3
Interest deposited. Updated balance: 3150.0

choice: 4
Exiting program. Goodbye!

Name: Rohit. Gandhi
USN: 2023BRS02579

23/01/24

Lab Program 6

+ Create a package CIE which has two classes - student and internals. class student should have USN, Name, sem. The class internals derived from student has an array that stores the internal marks scored in five courses of current sem of students. Create a package SEE and inport both packages.

+ student.java

```
package CIE;

inport java.util.Scanner;

public class Student {

protected string USN = new string();
protected string name = new string();
protected int sem;
public void input student details () {
  make scanner s object.
  read s.USN
  read s.name
  read s.sem
}
```

public void displayStudentDetails() {
display USN, name, sem
}

+ Externals.java

package CIE;

import java.util.Scanner;

public class ~~student~~ { internals extends student {
protected int marks[] = new int[5];
public void inputCIEmarks()
{
initialize scanner object
display for loop and read 5 subject marks using
5 bytes for(int i = 0; i<5; i++) {
} bytes.out.print("Enter marks "+(i+1)+": ");
} marks[i] = scanner.nextInt();

+ ~~Internals.java~~ Externals.java
package SEE;

import CIE.~~ma~~ Internals;

```
import java.util.Scanner;

public class Externals extends Externals {
    protected int marks[];
    protected int finalmarks[];

    public extends () {
        marks = new int [5]
        finalmarks = new int [5]; }

    public void input SEE marks () {
        Scanner s = new Scanner (System.in);
        for(int j=0; i<5; i++) {
            System.out.print ("Subject "+(i+1)+" marks:");
            marks[i] = s.nextInt();
        }
    }

    public void calculated Final Marks () {
        for(int i=0; i<5 ; i++)
            finalmarks[i] = marks[i]/2 + super.marks[i];
    }
    public void display Final Marks () {
        display student details();
        for(int i=0; i<5 ; i++)
            System.out.println("Subject "+(i+2)+": "+fin
```

```
Marks[i]);
}
}

+ Main.java

import SEE.externals;
class Main() {
    public static void main(String args[])
    {
    int numofstudent=2;
    Externals finalMarks[] = new
    Externals[num of students];
    for(int i=0; i<numofstudents; i++)
        {
        finalMarks[i] = new Externals();
        finalMarks[i] = input student details();
            System.out.println("Enter CSE marks");
            finalMarks[i].inputCSEmarks();
            System.out.println("Enter SEE marks");
            finalMarks[i].inputSEEmarks();
        }
System.out.println("Display the Data:\n");
        for(i=0; i<num of students; i++)
        {
```

```
        finalMarks[i].calculateFinalMarks();
        finalMarks[i].displayFinalMarks();
    }
}
}
```

Output :-

Enter No of students for Externals : 2

Enter details of student 1

USN : 2023BTS02599

Name : Rohit Gandhi

Semester : 3

Enter details of student : 2

USN : 2023BTS05811

Name : Rahul Roy

Semester : 3

Enter internal marks for 5 courses of student 1:
80, 75, 90, 87, 90

Enter internal marks for 5 courses of student 2:
85, 78, 92, 87, 90

Final marks of student 2,1:
student 1: 716
student 2: 722

23.01.24

Name: Rohit. Gandhi
USN: ~~BM~~ 2023 BMS02599

30/01/24

WEEK-7

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

```
import java.util.Scanner;

class WrongAge extends Exception {
    public WrongAge() {
        super("Age error");
    }

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

class InputScanner {
    protected Scanner scanner;
```

```
    public int nextInt() {
        return scanner.nextInt();
    }
}
class father extends InputScanner {
    protected int fatherage;

    public father() throws Wrongage {
        System.out.print("Enter father age:");
        fatherage = super.nextInt();

        if (fatherage < 0) {
            throw new message("age cannot be negative");
        }
    }
    public void display() {
        System.out.println("father age " + fatherage);
    }
}

class son extends father {
    private int sonage;

    public Son() throws Wrongage {
        super();
```

```
System.out.print("Enter the age:");
Sonage = sc.nextInt();

if (Sonage >= fatherage) {
    throw new WrongAge("Son age cannot be
    greater than or equal to father's age");
} else if (sonage < 0) {
    throw new Wrongage("Age cannot be negative");
}
}

public class Exception {
    public static void main(String[] args) {
    try {
        Son son = new son();
        son.display();
    } catch (Wrongage e) {
        System.out.println("Error:" + e.negative());
    }
}
}
```

Output:-

Enter father's age : 20
Enter sons age : 12
Father's age : 20
Son's age : 12

Enter father's age : 10
Enter son's age : 12
Error : Son's age cannot be greater than fathers age.

Enter father's age : 20
Enter son's age : -2
Error : Age cannot be negative

Name :- Rohit Gandhi D
USN :- 2023BMS02527

29/01/25

6/2/24 WEEK-8

+ Write a program which creates 2 threads, one thread displaying "RNS College of Engineering" once every ten seconds and another displaying "CSE" once every two seconds.

```
class displayThread extends Thread {
    private String message;
    private int interval;
}

public void run() {
    while (true)
        System.out.println(message);
    try {
        Thread.sleep(interval);
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
}
}
```

```
public class threads {

    public static void main(String[] args) {
        DisplayThread thread1 = new DisplayThread("BNS college of Engineering");
        DisplayThread thread2 = new DisplayThread("CSE", 2000);

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

Output :- Name : Rohit. Gandhi USN : 2023BNS02599

```
BNS college of Engineering
CSE
CSE
CSE
CSE
CSE
BNS college of Engineering
CSE
CSE
CSE
CSE
```

06/2/24 WEEK-10

+ Demonstrate Inter process communication & deadlock

```
public class Deadlockexample {
    final SharedResource sharedresource = new
    SharedResource();

    Thread process1 = new Thread(() -> {
    try {
    sharedresource.method1();
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
});

Thread process2 = new Thread(() -> {
    try {
        sharedResources.method2();
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
});
    process1.start();
    process2.start();
} }
```

```
class SharedResources {
    private final Object lock1 = new Object();
    private final Object lock2 = new Object();

    public void method1() throws InterruptedException {
        synchronized (lock1) {
            System.out.println("Method 1 acquired lock2");
        }
    }
}

public void method2() throws InterruptedException {
    synchronized (lock2) {
        System.out.println("Method 2 acquired lock2");
        Thread.sleep(1000);

        synchronized (lock1) {
            System.out.println("Method 2 acquired lock2");
        }
    }
}
```

Output:-
Method 1 acquired lock 1
Method 2 acquired lock 2

Output:- Main thread entered A.foo
Lacey thread entered B.teas
Main thread trying to call B.last()
Inside A.last
Back in main thread
Lacey thread trying to call A.last()
Inside A.last
Back in other thread.

06.02.24

Name:- Rohit Gandhi
USN:- 2025BR502599

2-2SN-405

classmate
Date
Page

12/2/2024

Program -10

+ Inter process communication

```
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("InterruptionException
        caught");
    }
    System.out.println("Got: " + n);
    value set = false;
      System.out.println("\n Intimate Producer\n");
      notify();
      return n;
    }
```

```
Synchronized void put(int n) {
while (value set)
try {
    System.out.println("\n Producer waiting\n");
wait();
}
catch (Interrupted Exception e)
{
  System.out.println("Interrupted Exception
                          caught");
}
this.n = n;
value set = true;
System.out.println("\n Putting 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 :-

Name : Rohit. Gandhi
USN : 2023BMS02597

```
Put : 1
Get : 1
Put : 2
Get : 2
Put : 3
Got : 3
Put : 4
Got : 4
.
.
.
```

13.02.24

WEEK-9

Write a program that creates a user Interface to perform Integer division

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

class SwingDemo {
    SwingDemo()
    {
        JFrame jfrm = new JFrame("Divider App");
        jfrm.setSize(275,200);
        jfrm.setLayout(new FlowLayout());
        jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_
                                      CLOSE);

        JLabel jlab = new JLabel("Enter the divider
        and dividend:");

        JTextField ajtf = new JTextField(8);
        JTextField ajtf = new JTextField(8);

        JButton button = new JButton("Calculate");
```

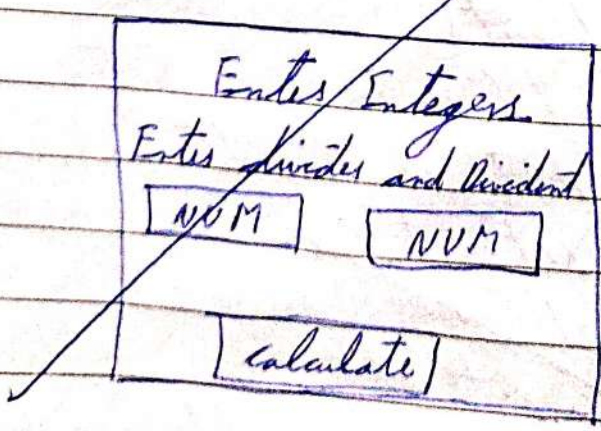
```
Ilabel ess = new Ilabel();
Ilabel alab = new Ilabel();
Ilabel blab = new Ilabel();
Ilabel anslab = new Ilabel();

jfm.add(ess);
jfm.add(ilab);
jfm.add(ajtf);
jfm.add(bjtf);
jfm.add(button);
jfm.add(alab);
jfm.add(blab);
jfm.add(anslab);

ActionListener l = new ActionListener()
{
    public void actionPerformance(ActionEvent ae)
    {
        System.out.println("Action event from a
                            text field");
    }
};
ajtf.addActionListener(l);
bjtf.addActionListener(l);
```

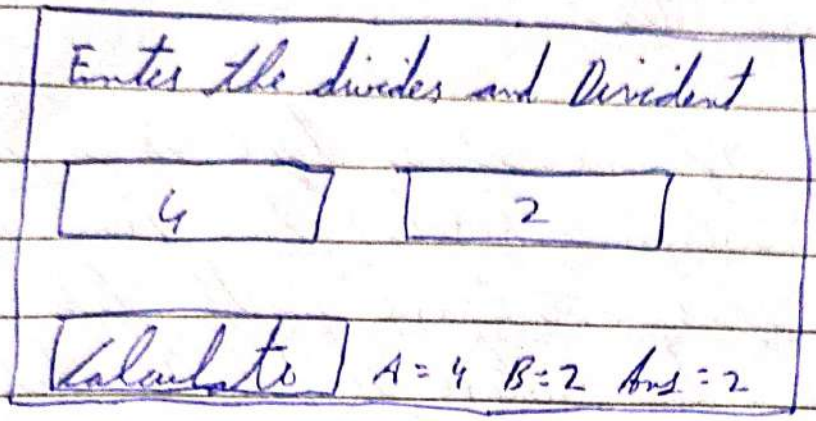
```
button.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent et)
    {
    try {
      int a = Integer.parseInt(ajtf.getText());
      int b = Integer.parseInt(bjtf.getText());
      if (b==0) {
          throw new ArithmeticException();
      }
      int ans = a/b;

      alab.setText("\nA = "+a);
      blab.setText("\nB = "+b);
      anslab.setText("\nAns = "+ans);
      ess.setText("");
    } catch (NumberFormatException e) {
      alab.setText("");
      blab.setText("");
      ess.setText("Enter only Integers!");
    } catch (ArithmeticException e) {
      alab.setText("");
      blab.setText("");
      anslab.setText("");
      ess.setText("B should be NON zero!");
    }
});
```

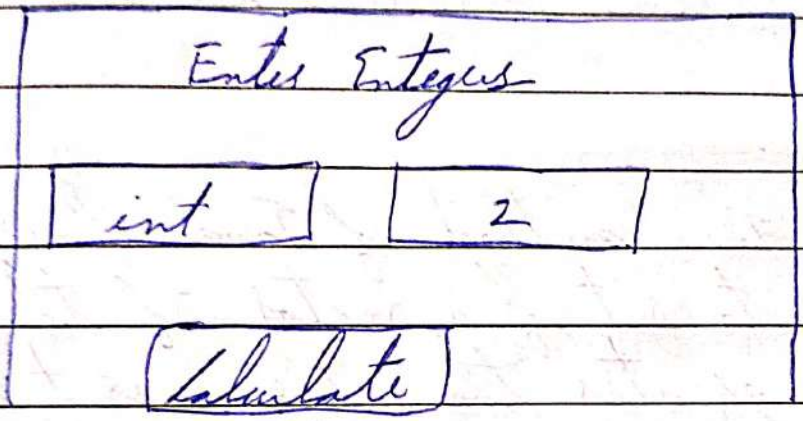
```
        jfm.setVisible(true);
    }
    public static void main(String args[]){
        SwingUtilities.invokeLater(new Runnable(){
            public void run()
            {
                new SavingZero();
            }
        });
    }
}
```

Output :-

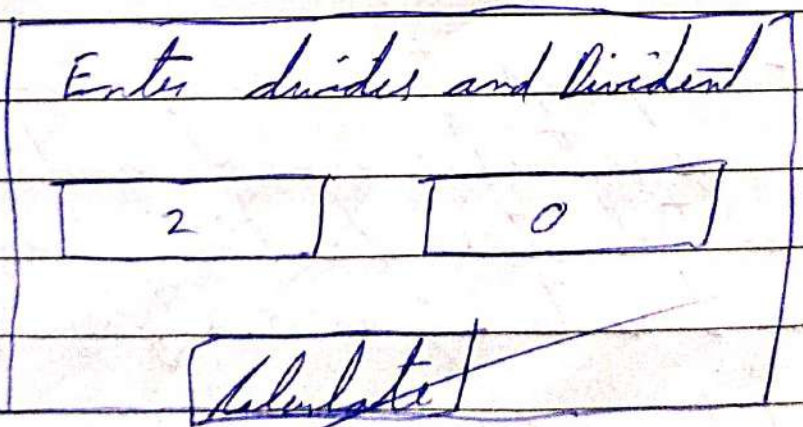
Name :- Rohit. Gandhi
USN :- 2023BMS02599







Error: Action event from a text field



Error: Text field element cannot be 0.

Date
Page

+ Functions Used:-

1) JFrame - JFrame class type is a container which inherits the java.awt.frames class. JFrame works like the main window where components like label, buttons, textfields are added to create GUI.

2) setSize :- This method of Java class is used to set of a vector. If the size is greater than the current size, null items are added to the end of the vector.

3) setLayout - It is used to arrange components in a particular manner and is used to control the positioning and size of the components in GUI forms.

4) setdefaultcloseoperation :- It plays an vital role in creating interactive applications. Its essential to handle close operations.

5) JLabel - Used to place text in a container. It is used to display a single line of read only text.

6) JTextfield - It is a text component that allows the editing of a single line text. It inherits JText Components class.

7) addframe - Used to add new frame in the GUI. it plays a vital role in GUI creation process.

8) addaction listener - It is used to notify whenever you click on the button or menue item. It is notified against action Event.

9) Set Text :- This method is used to set the new text into the ~~Breaklterator~~ used ~~it~~ as setText().

Name :- Rohit Gandhi

USN :- 2023 BNS 02599

WEEK 1

1. 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.

Program:

```
import java.util.Scanner;

class Quadratic {
   int a, b, c;
   double r1, r2, d;

   void getCoefficients() {
      Scanner scanner = new Scanner(System.in);
      System.out.println("Enter the coefficients of a, b, and c:");
      a = scanner.nextInt();
      b = scanner.nextInt();
      c = scanner.nextInt();
   }

   void computeRoots() {
      while (a == 0) {
         System.out.println("Not a quadratic equation");
         System.out.println("Enter a non-zero value for a:");
         Scanner scanner = new Scanner(System.in);
         a = scanner.nextInt();
      }

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

      if (d == 0) {
         r1 = -b / (2.0 * a);
         System.out.println("Roots are real and equal");
         System.out.println("Root1 = Root2 = " + r1);
      } else if (d > 0) {
         r1 = (-b + Math.sqrt(d)) / (2.0 * a);
```

```
            r2 = (-b - Math.sqrt(d)) / (2.0 * a);
            System.out.println("Roots are real and distinct");
            System.out.println("Root1 = " + r1 + " Root2 = " + r2);
        } else {
            System.out.println("Roots are imaginary");
            r1 = -b / (2.0 * a);
            r2 = Math.sqrt(-d) / (2.0 * a);
            System.out.println("Root1 = " + r1 + " + i" + r2);
            System.out.println("Root2 = " + r1 + " - i" + r2);
        }
    }
}

class QuadraticMain {
    public static void main(String[] args) {
        System.out.println("My Name is Rohit Gandhi");
        System.out.println("My USN is 2023BMS02599");
        Quadratic quadratic = new Quadratic();
        quadratic.getCoefficients();
        quadratic.computeRoots();
    }
}
```

WEEK2

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

Program:

```
import java.util.Scanner;
import java.io.*;

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

    public Student(String usn, String name, int numSubjects) {
        this.usn = usn;
        this.name = name;
        this.credits = new int[numSubjects];
        this.marks = new int[numSubjects];
    }

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

        System.out.println("Enter details for student " + name + " (USN: " + usn +
")");
        for (int i = 0; i < credits.length; i++) {
            System.out.print("Enter credits for subject " + (i + 1) + ": ");
            credits[i] = scanner.nextInt();

            System.out.print("Enter marks for subject " + (i + 1) + ": ");
            marks[i] = scanner.nextInt();
        }
    }

    public void displayDetails() {
        System.out.println("Details for student " + name + " (USN: " + usn + ")");
```

```
        for (int i = 0; i < credits.length; i++) {
            System.out.println("Subject " + (i + 1) + " - Credits: " + credits[i] + ",
Marks: " + marks[i]);
        }
    }

    public double calculateSGPA() {
        double totalCredits = 0;
        double totalGradePoints = 0;

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

        return totalGradePoints / totalCredits;
    }

      private double calculateGradePoints(int marks) {
        if (marks >= 90) {
            return 10.0;
        } else if (marks >= 80) {
            return 9.0;
        } else if (marks >= 70) {
            return 8.0;
        } else if (marks >= 60) {
            return 7.0;
        } else if (marks >= 50) {
            return 6.0;
        } else {
            return 0.0;
        }
    }
}

public class StudentSGPA {
    public static void main(String[] args) {
        System.out.print("my name is Rohit Gandhi");
        System.out.print("my USN is 2023BMS02599");
```

```
        Scanner scanner = new Scanner(System.in);

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

        System.out.print("Enter the USN: ");
        String usn = scanner.next();

        System.out.print("Enter the name: ");
        String name = scanner.next();

        Student student = new Student(usn, name, numSubjects);
        student.acceptDetails();

        student.displayDetails();
        System.out.println("SGPA: " + student.calculateSGPA());
    }
}
```

WEEK3

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

Program:

```
import java.util.Scanner;

class Book {
    String name;
    String author;
    int price;
    int numPages;

    public Book(String name, String author, int price, int numPages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.numPages = numPages;
    }

    public String toString() {
        String bookDetails = "Book name: " + this.name + "\n"
                + "Author name: " + this.author + "\n"
                + "Price: " + this.price + "\n"
                + "Number of pages: " + this.numPages + "\n";
        return bookDetails;
    }
}

public class BookStore{

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

        System.out.print("Enter the number of books: ");
```

```
        int n = scanner.nextInt();

        Book[] books = new Book[n];

        for (int i = 0; i < n; i++) {
            System.out.print("Enter name of the book: ");
            scanner.nextLine(); // consume the newline character
            String name = scanner.nextLine();

            System.out.print("Enter author of the book: ");
            String author = scanner.nextLine();

            System.out.print("Enter the price of the book: ");
            int price = scanner.nextInt();

            System.out.print("Enter the number of pages of the book: ");
            int numPages = scanner.nextInt();

            books[i] = new Book(name, author, price, numPages);
        }

        System.out.println("\nBook Details:");
        for (int i = 0; i < n; i++) {
            System.out.println("Book " + (i + 1) + ":\n" + books[i]);
        }
    }
}
```

WEEK4

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

Program:

```
import java.util.Scanner;

abstract class Shape {
    protected int side1;
    protected int side2;

    public Shape(int side1, int side2) {
        this.side1 = side1;
        this.side2 = side2;
    }

    public abstract void printArea();
}

class Rectangle extends Shape {
    public Rectangle(int length, int width) {
        super(length, width);
    }

    @Override
    public void printArea() {
        int area = side1 * side2;
        System.out.println("Rectangle Area: " + area);
    }
}

class Triangle extends Shape {
    public Triangle(int base, int height) {
        super(base, height);
    }
```

```
    @Override
    public void printArea() {
        double area = 0.5 * side1 * side2;
        System.out.println("Triangle Area: " + area);
    }
}
class Circle extends Shape {
    public Circle(int radius) {
        super(radius, 0);
    }
    @Override
    public void printArea() {
        double area = Math.PI * side1 * side1;
        System.out.println("Circle Area: " + area);
    }
}
public class Area {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter length and width for Rectangle:");
        int rectLength = scanner.nextInt();
        int rectWidth = scanner.nextInt();
        Rectangle rectangle = new Rectangle(rectLength, rectWidth);
        rectangle.printArea();

        System.out.println("Enter base and height for Triangle:");
        int triBase = scanner.nextInt();
        int triHeight = scanner.nextInt();
        Triangle triangle = new Triangle(triBase, triHeight);
        triangle.printArea();

        System.out.println("Enter radius for Circle:");
        int circleRadius = scanner.nextInt();
        Circle circle = new Circle(circleRadius);
        circle.printArea();
        scanner.close();
    }
}
```

WEEK5

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.

Program:

```
import java.util.Scanner;

abstract class Account {
    String customerName;
    long accountNumber;
    String accountType;
    double balance;

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

    public void displayBalance() {
        System.out.println("Account Balance: $" + balance);
    }
    public abstract void withdraw(double amount);
```

```
}

class CurrAcct extends Account {
    double minimumBalance;
    double serviceCharge;

    public CurrAcct(String customerName, long accountNumber, double balance) {
        super(customerName, accountNumber, "Current Account", balance);
        this.minimumBalance = 1000;
        this.serviceCharge = 50;
    }

    @Override
    public void withdraw(double amount) {
        if (balance - amount >= minimumBalance) {
            balance -= amount;
            System.out.println("Withdrawal successful. Remaining balance: $" +
balance);
        } else {
            System.out.println("Insufficient funds. Service charge of $" +
serviceCharge + " applied.");
            balance -= serviceCharge;
            System.out.println("Remaining balance after service charge: $" + balance);
        }
    }
}

class SavAcct extends Account {
    double interestRate;

    public SavAcct(String customerName, long accountNumber, double balance) {
        super(customerName, accountNumber, "Savings Account", balance);
        this.interestRate = 0.05; // Set interest rate (5%)
    }

    @Override
    public void withdraw(double amount) {
        if (balance - amount >= 0) {
            balance -= amount;
            System.out.println("Withdrawal successful. Remaining balance: $" +
```

```
balance);
        } else {
            System.out.println("Insufficient funds. Cannot complete withdrawal.");
        }
    }

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

public class Bank1 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter customer name: ");
        String customerName = scanner.nextLine();

        System.out.print("Enter account number: ");
        long accountNumber = scanner.nextLong();

        System.out.print("Enter initial balance: ");
        double initialBalance = scanner.nextDouble();

        System.out.print("Enter account type (Current/Savings): ");
        String accountType = scanner.next();

        Account account;
        if (accountType.equalsIgnoreCase("Current")) {
            account = new CurrAcct(customerName, accountNumber, initialBalance);
        } else if (accountType.equalsIgnoreCase("Savings")) {
            account = new SavAcct(customerName, accountNumber, initialBalance);
        } else {
            System.out.println("Invalid account type. Exiting program.");
            return;
        }

        int choice;
```

```
        do {
            System.out.println("\n1. Deposit");
            System.out.println("2. Display Balance");
            System.out.println("3. Deposit Interest for Savings Account");
            System.out.println("4. Withdraw");
            System.out.println("5. Exit");
            System.out.print("Enter your choice: ");
            choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    System.out.print("Enter deposit amount: ");
                    double depositAmount = scanner.nextDouble();
                    account.balance += depositAmount;
                    System.out.println("Deposit successful. Updated balance: $" +
account.balance);
                    break;

                case 2:
                    account.displayBalance();
                    break;

                case 3:
                    if (account instanceof SavAcct) {
                        ((SavAcct) account).depositInterest();
                    } else {
                        System.out.println("This option is applicable for Savings Account
only.");
                    }
                    break;

                case 4:
                    System.out.print("Enter withdrawal amount: ");
                    double withdrawalAmount = scanner.nextDouble();
                    account.withdraw(withdrawalAmount);
                    break;

                case 5:
                    System.out.println("Exiting program. Goodbye!");
                    break;
```

```
                default:
                    System.out.println("Invalid choice. Please enter a valid option.");
            }

        } while (choice != 5);

        scanner.close();
    }
}
```

WEEK6

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

Program:

```
// MainProgram.java
import CIE.Internals;
import SEE.External;
import java.util.Scanner;

public class MainProgram {

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

        System.out.println("Enter the number of students:");
        int numStudents = scanner.nextInt();

        Internals[] internalsArray = new Internals[numStudents];
        External[] externalsArray = new External[numStudents];

        for (int i = 0; i < numStudents; i++) {
            System.out.println("Enter details for Student " + (i + 1));
            System.out.print("USN: ");
            String usn = scanner.next();

            System.out.print("Name: ");
            String name = scanner.next();

            System.out.print("Semester: ");
            int sem = scanner.nextInt();
```

```
            System.out.println("Enter Internal Marks for 5 courses:");
            int[] internalMarks = new int[5];
            for (int j = 0; j < 5; j++) {
                System.out.print("Course " + (j + 1) + ": ");
                internalMarks[j] = scanner.nextInt();
            }

            internalsArray[i] = new Internals(usn, name, sem, internalMarks);

            System.out.println("Enter SEE Marks for 5 courses:");
            int[] seeMarks = new int[5];
            for (int j = 0; j < 5; j++) {
                System.out.print("Course " + (j + 1) + ": ");
                seeMarks[j] = scanner.nextInt();
            }

            externalsArray[i] = new External(usn, name, sem, seeMarks);
        }

        System.out.println("\nFinal Marks of Students:");
        for (int i = 0; i < numStudents; i++) {
            int[] internalMarks = internalsArray[i].getInternalMarks();
            int[] seeMarks = externalsArray[i].getSeeMarks();

            int[] finalMarks = new int[5];
            int totalMarks = 0;

            System.out.println("Student " + (i + 1) + " - " +
internalsArray[i].getName());

            for (int j = 0; j < 5; j++) {
                finalMarks[j] = internalMarks[j] + seeMarks[j];
                totalMarks += finalMarks[j];
                System.out.println("Course " + (j + 1) + ": " + finalMarks[j]);
            }

            System.out.println("Total Marks: " + totalMarks + "\n");
        }
    }
}
```

```
// CIE/Internals.java
package CIE;

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

    public Internals(String usn, String name, int sem, int[] internalMarks) {
        super(usn, name, sem);
        this.internalMarks = internalMarks;
    }

    public int[] getInternalMarks() {
        return internalMarks;
    }
}

// CIE/Student.java
package CIE;

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

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

    public Student() {
        this("", "", 0);
    }

    public String getName() {
        return name;
    }
}
```

```
// SEE/External.java
package SEE;

import CIE.Student;

public class External extends Student {
    protected int[] seeMarks;

    public External(String usn, String name, int sem, int[] seeMarks) {
        super(usn, name, sem);
        this.seeMarks = seeMarks;
    }

    public int[] getSeeMarks() {
        return seeMarks;
    }
}
```

WEEK7

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

Program:

```
import java.util.Scanner;

class WrongAge extends Exception {
   public WrongAge() {
      super("Age Error");
   }

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

class InputScanner {
   protected Scanner scanner;

   public InputScanner() {
      scanner = new Scanner(System.in);
   }

   public int nextInt() {
      return scanner.nextInt();
   }
}

class Father extends InputScanner {
   protected int fatherAge;

   public Father() throws WrongAge {
      System.out.println("Enter father's age:");
```

```
        fatherAge = super.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();
        System.out.println("Enter son's age:");
        sonAge = super.nextInt();

        if (sonAge >= fatherAge) {
            throw new WrongAge("Son's age cannot be greater than or equal to father's
age");
        } else if (sonAge < 0) {
            throw new WrongAge("Age cannot be negative");
        }
    }
    public void display() {
        super.display();
        System.out.println("Son's age: " + sonAge);
    }
}
public class ExceptionHandlingDemo {
    public static void main(String[] args) {
        try {
            Son son = new Son();
            son.display();
        } catch (WrongAge e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

WEEK8

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.

Program:

```
class DisplayThread extends Thread {
    private String message;
    private int interval;

    public DisplayThread(String message, int interval) {
        this.message = message;
        this.interval = interval;
    }

    public void run() {
        try {
            while (true) {
                System.out.println(message);
                Thread.sleep(interval);
            }
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
    }
}

public class week8 {
    public static void main(String[] args) {
        DisplayThread thread1 = new DisplayThread("BMS College of Engineering",
10000);
        DisplayThread thread2 = new DisplayThread("CSE", 2000); // 2 seconds

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

WEEK9

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.

Program:

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

class SwingDemo{
   SwingDemo(){
      JFrame jfrm = new JFrame("Divider App");
      jfrm.setSize(275, 200);
      jfrm.setLayout(new FlowLayout());
      jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      JLabel jlab = new JLabel("Enter the divider and dividend:");

      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(err);  // to display error message
      jfrm.add(jlab);
      jfrm.add(ajtf);
      jfrm.add(bjtf);
      jfrm.add(button);
      jfrm.add(alab);
```

```
jfrm.add(blab);
jfrm.add(anslab);

ActionListener l = new ActionListener() {
   public void actionPerformed(ActionEvent evt) {

   }
};
ajtf.addActionListener(l);
bjtf.addActionListener(l);

button.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent evt) {
      try{
         int a = Integer.parseInt(ajtf.getText());
         int b = Integer.parseInt(bjtf.getText());
         if (b == 0) {
            throw new ArithmeticException();
         }
         int ans = a/b;

         alab.setText("A = " + a);
         blab.setText("B = " + b);
         anslab.setText("Ans = "+ ans);
         err.setText(""); // Clear any previous error message
      }
      catch(NumberFormatException e){
         clearLabels();
         err.setText("Enter Only Integers!");
      }
      catch(ArithmeticException e){
         clearLabels();
         err.setText("B should be NON zero!");
      }
   }

   private void clearLabels() {
      alab.setText("");
      blab.setText("");
      anslab.setText("");
```

```
            }
        });

        jfrm.setVisible(true);
    }

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

WEEK10

10.Demonstrate Inter process Communication and deadlock

Program:

A.Deadlock

```
class A {
   synchronized void foo(B b) {
      String name = Thread.currentThread().getName();
      System.out.println(name + " entered A.foo");
      try {
         Thread.sleep(1000);
      } catch (Exception e) {
         System.out.println("A Interrupted");
      }
      System.out.println(name + " trying to call B.last()");
      b.last();
   }

   synchronized void last() {
      System.out.println("Inside A.last");
   }
}

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

   synchronized void last() {
```

```
        System.out.println("Inside 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();
    }
}
```

B.InterprocessCommunication:

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