B. M. S. COLLEGE OF ENGINEERING

(AUTONOMOUS COLLEGE UNDER VTU, BELGAUM)
BANGALORE – 560019

2023-24

LAB RECORD
OBJECT ORIENTED JAVA PROGRAMMING (23CS3PCOOJ)

Submitted by :

NAME : Santosh B

USN : 1BM22CS243

SECTION : 3E

Submitted to

Dr. Seema Patil
Assistant Professor
Dept. of CSE, BMSCE

INDEX

Name : Santosh. B Class : E

Section : ..... Roll No. : ..... Subject : OOP's in Java

Sl. No.	Date	Title	Page No.	Teacher's Sign. / Remarks
1	5/12/23	Sample programs		12-12-23
2	12/12/23	Program1 of Lab		
3	19/12/23	Program 2 of Lab.		
4	26/12/23	Program 3 of Lab.		26/12/23
5	08/01/24	program 4 of Lab		2-1-2024
6	16/01/24	program 5 of Lab		16-1
7	16/01/24	program 5 of Lab		16-1
8	23/01/24	program 6 of Lab		23-1-24
9	30/01/24	program 7 of Lab		30-1-24
10	06/02/24	program 8 of Lab		6-2-24
11	13/02/24	program 10 of Lab		13-2-24
12	20/02/24	program 9 of Lab		20/2/2024

Lab programs.

① → Develop a java program that prints all real soln to the quadratic $ax^2+bx+c=0$, read a, b & c & use the quadratic formula & discriminate b^2-4ac is negative, display a message stating that are no real soln's

```
import java.util.Scanner;
class Quadratic
{
    int a, b, c;
    double r1, r2, d;
    void getd()
    {
        Scanner s = new Scanner(System.in);
        System.out.println("enter coefficients a,b,c");
        a = s.nextInt();
        b = s.nextInt();
        c = s.nextInt();
    }
    void compute()
    {
        while(a==0)
        {
            System.out.println("NOT a Quadratic eq");
            System.out.println("Enter a non zero");
            Scanner s = new Scanner(System.in);
            a = s.nextInt();
        }
        d = b*b - 4*a*c;
        if(d==0)
        {
            r1 = (b)/(2*a);
```

```
System.out.println("Roots are real and equal");
System.out.println("Root1 = Root2 =" + r1);
}
else if(d>0)
{
    r1 = (-b) + (Math.sqrt(d))/(double)(2*a);
    r2 = ((-b)) - (Math.sqrt(d))/(double)(2*a);
    System.out.println("Roots are real and distinct");
    System.out.println("Root1 = " + r1 + "root2 = " + r2);
}
else if(d<0)
{
    System.out.println("Roots are imaginary");
    r1 = (-b)/(2*a);
    r2 = (-Math.sqrt(-d)/(2*a);
    System.out.println("Root1 = " + r1 + " +i" + r2);
    System.out.println("Root1 = " + r1 + " -i" + r2);
}
}
}
class QuadraticMain
{
    public static void main(String args[])
    {
        Quadratic q = new Quadratic();
        q.getdl();
        q.computel();
        System.out.println("Done by: Santosh B,
                            USN: 1BM22CS243");
    }
}
```

output:-

enter the coefficients of a, b, c:

3 4 5

Roots are imaginary

Root1 = 0.0 + i1.1055415967851332

Root2 = 1.1055415967851332 - i1.1055415967851332

Done by: Suntosh B, USN: 1BM22CS243

Enter the coefficients of a, b, c:

1 2 1

Roots are real and equal

Root1 = Root2 = -1.0

Done by: Suntosh B, USN: 1BM22CS243

Enter the coefficients of a, b, c:

1 4 1

Roots are real and distinct

Root1 = -0.2679491924311228

Root2 = -3.732050807568877

Done by: Suntosh B, USN: 1BM22CS243

12/12/2023

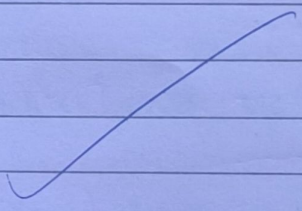
Additional programs:-

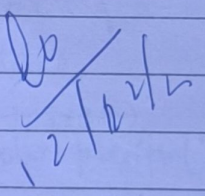
① Demonstration of 1-D array

output Enter the size of the array
5
1 2 3 4 5
Array elements are:
1 2 3 4 5.

② Demonstration on passing
Enter the length:
5
Enter the breath:
5
Area of the rectangle: 25

③ Demonstration of sum of five digits
Enter 5 digit number
12345
sum of five digits : 15





Lab program (2)

(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.

```
import java.util.Scanner;

class Subject {
    int subjectMarks;
    int credits;
    int grade;
}
class student {
    Subject subject[];
    String name;
    String USN;
    double SGPA;
    Scanner sc;

    public student() {
        subject = new Subject[8];
        for (int i=0; i<8; i++) {
            subject[i] = new Subject();
        }
    }
```

→

```
void getStudentDetails() {
    sc = new Scanner(System.in);
    System.out.println("Enter name : ");
    this.name = sc.next();

    System.out.println("Enter your USN : ");
    this.USN = sc.next();
}

void getMarks() {
    sc = new Scanner(System.in);
    for(int i=0; i<8; i++) {
        System.out.print("Enter " + (i+1) + " subject
                marks : ");
        subject[i].subjectMarks = sc.nextInt();

        System.out.print("Enter number of credits : ");
        subject[i].credits = sc.nextInt();

        subject[i].grade = (subject[i].subjectMarks/10)+1

        if(subject[i].grade == 11) subject[i].grade = 10;

        if(subject[i].grade <= 4) subject[i].grade = 0;
    }
}
```

```
void computeSGPA() {
    int totalCredits = 0;
    for(int i=0; i<8; i++) {
        totalCredits += subject[i].credits;
    }

    int totalGradeAndCredit = 0;
    for (int i=0; i<8; i++ ) {
        totalGradeAndCredit += (subject[i].credits *
                                subject[i].grade);
    }

    this.SGPA = (float)totalGradeAndCredit/totalCredits;

    System.out.println("SGPA of the student is :" + this.SGPA);
  }
}

class studentMain {

    public static void main (String[] args) {
        student s1 = new student();
        s1.getStudentDetails();
        s1.getMarks();
        s1.computeSGPA();
    }
}
```

output:-

Enter name : Santosh
Enter your USN : 1BM22CS243
Enter 1 subject marks : 100
Enter number of credits : 4
Enter 2 subject marks : 95
enter number of credits : 4
Enter 3 subject marks : 90
Enter number of credits : 3
Enter 4 subject marks : 88
Enter number of credits : 1
Enter 5 subject marks : 93
enter number of credits : 1
Enter 6 subject marks : 98
Enter number of credit : 3
Enter 7 subject marks : 93
Enter number of credits : 3
Enter 8 subject marks : 99
Enter number of credits : 1

SGPA of the student is : 9.949999802651377

name : Santosh B
USN : 1BM22CS243

19-12-23

26/12/23 Lab program ③

create a class book which contains four members: name, author, price, num-pages. Include a constructor to set the values of 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.*;

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 void set(String name, String author, int price,
                    int numpages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.numpages = numpages;
    }
```

```
public sting toSteing() {
    Sting name = "Book name: " + this.name + "\n";
    Sting author = "Author name:" + this.author + "\n";
    Sting price = "price: " + this.price + "\n";
    Sting numpages = "Number of pages: " + this.numpages + "\n";

    return name + author + price + numpages;
}

public Steing getname() {
    return this.name;
}
public sting getAuthor() {
    return this.author;
}
public int getprice() {
    return this.price;
}
public int getNumberOfpages() {
    return this.numpage;
}
}
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("enter number of books: ");
        int n = sc.nextInt();

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

```
for(int i=0; i<n; i++) {
    System.out.print("Enter" + (i+1) + " Book name,
        author name, price and number of pages: ");
    String name = sc.next();
    String author = sc.next();
    int price = sc.nextInt();
    int numOfpages = sc.nextInt();

    b[i] = new BOOK(name, author, price, numpages);
}

for(int i=0; i<n; i++){
    System.out.println(b[i].toString());
}
}
}
```

Output:-

Enter number of books : 2
Enter 1 book name, author name, price and number of pages : book1 siri 1400 68
Enter 2 book name, author name, price and number of pages : book 2 santosh 1600 70.

Book name : book 1
Author name : siri
Price : 1400
Number of pages : 68

Book name : book 2
Author name : santosh
Price : 1600
Number of pages : 70

Done by santosh B, 1BM22CS243

02/01/2024

(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 class shape. Each one of the classes contain only the method printArea() that prints the area of the given shape

```
import java.util.*;

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

abstract class Shape extends InputScanner {
    double a,b;
    abstract void getInput();
    abstract void displayArea();
}

class Rectangle extends Shape {

    @Override
    void getInput() {
        System.out.print("Enter the length and breath
                          of a Rectangle :");
        this.a = sc.nextDouble();
        this.b = sc.nextDouble();
    }
```

```
@Override
void displayArea() {
    System.out.println("The area of the Rectangle is
                        : ", (this.a * this.b));
  }
}

class Triangle extends Shape {

    @Override
    void getInput() {
        System.out.print("Enter Height and Base of
                          Triangle : ");
        this.a = sc.nextDouble();
        this.b = sc.nextDouble();
    }

    @Override
    void display() {
        System.out.println("Enter the radius of the circle:");
        this.a = sc.nextDouble();
    }

    @Override
    void displayArea() {
        System.out.println("The area of the circle
                            is : " + (Math.PI * this.a * this.a));
    }
}
```

```
class AbstractDemo {

    public static void main (String[] args) {
        Shape s;
        s = new circle();
        s.getInput();
        s.get s.displayAreal();

        s = new Rectangle();
        s.getInput();
        s.displayAreal();

        s = new Triangle();
        s.getInput();
        s.displayArea();
    }
}
```

output:-

```
enter the radius of circle : 3
The area of circle is : 28.274333882308138
enter the length and breath of Rectangle : 2 4
The area of Rectangle is 8.0
enter the height and Base of a Triangle : 4 4
The area of triangle is : 8.0
```

2-1-2024

Lab 5

18/01/24 Develop a java program to develop a class Bank with includes current and savings account.

```
import java.util.Scanner;

abstract class Account {
    String name;
    int accno;
    String type;

    abstract void deposit(double amount);
    abstract void display();
    abstract void withdraw(double amount);
    abstract void menu();
}

class current extends Account {
    double balance;
    private static int MIN_BALANCE = 500;
    private static int SER_CHARGE = 10;

    public current(String name, int accno, String t
    {
        this.name = name;
        this.accno = accno;
        this.type = type;
    }

    void deposit(double amount) {
        this.balance += amount;
        System.out.println("Amount deposited"
```

```
if(this.balance < MIN_BALANCE) {
        System.out.println("Penalty");
        this.balance -= SER_CHARGE;
    }
}

void display() {
    System.out.println("Name : " + this.name);
    System.out.println("Account number : " + this.accNo);
    System.out.println("Balance : " + this.balance);
}

void withdraw(double amount) {
    if(this.balance < amount) {
        System.out.println("Insufficient balance");
    }
    else {
        this.balance -= amount;
    }
}
void menu() {
    System.out.println("--MENU--");
    int choice;
    double amount;
    do {
        System.out.println("1.Deposit 2.Withdraw 3.details
                4. Exit");
        Scanner sc = new Scanner(System.in);
        choice = sc.nextInt();
```

```
switch(choice){
    case 1: System.out.println("enter amount:");
            amount = sc.nextDouble();
            this.deposit(amount);
            break;

    case 2: System.out.println("enter withdraw
            amount:");
            amount = sc.nextDouble();
            this.withdraw(amount);
            break;

    case 3: this.display();
            break;

    case 4: return;
  }
  while(choice != 4);
 }

}
class savings extends Account {
    int balance;
    final float interest = 5f;

    public savings(String name, int accno, String ty
    {
        this.name = name;
        this.accno = accno;
        this.type = type;
    }
```

```
void deposit (double amount) {

        float interest-amount = (float) (amount *
                              (this . interval /100));
    system.out.println ("interest of : " + interestAmount
                   + " is added");
    this.balance += amount + interestAmount;
    System.out.println ("Amount deposited");
}

void display() {
    System.out.println("Name : " + this.name);
    System.out.println("Account Number : " + this.accn
    System.out.println(" Balance : " + this.balance);
}

void withdraw(double amount) {
    if(this.balance < amount) {
        System.out.println(" Insufficient balance");
    }
    else {
        this.balance -= amount;
    }
}
void menu() {
    System.out.println("--MENU--");
    int choice;
    double amount;
    do {
     System.out.println(" 1. deposit 2. withdraw
                   3. details 4. Exit ");
```

```
Scanner sc = new Scanner(System.in);
choice = sc.nextInt();
do {
    System.out.println(" 1. Deposit 2. Withdraw 3. detail
                        4. Exit");
    Scanner sc = new Scanner(System.in);
    choice = sc.nextInt();

        switch(choice) {
            case 1: System.out.print("enter amount: ");
                    amount = sc.nextDouble();
                    this.deposit(amount);
                    break;

            case 2: System.out.println("enter withdrawal:");
                    amount = sc.nextDouble();
                    this.withdrawal(amount);
                    break;

            case 3: this.display();
                    break;

            case 4: return;
        }
    } while(choice != 4);
}
```

```
public class Bank {

    public static void main (String[] args) {
        Scanner sc = new Scanner (System.in);
        System.out.println ("enter name, accno, type:");
        String name = sc.next();
        int accno = sc.nextInt();
        String type = sc.next();

        if (type.equals ("current")) {
            Account c = new Current (name, accno,
                                     type);
            c.menu();
        }
        else {
            Account s = new Savings (name, accno,
                                     type);
            s.menu();
        }
    }
}
```

output:

```
enter name, accno, type:
santosh
12
current
```

--MENU--
1. deposit 2. Withdraw 3 details 4. Exit
1
Enter ~~the~~ deposit amount : 30000
amount deposited.
1. deposit 2 Withdraw 3 details 4. Exit
2
Enter withdrawal : 1000
1. deposit 2. Withdraw 3. details 4 Exit
2
Enter withdrawal amount :
6000
insufficient balance.
1. deposit 2. Withdraw 3. details 4. Exit
3 2
Name : Santosh
Account number : 12
Balance : 4000.

Lab 5:

1) Write a java program of stack generic class to push and pop 5 integer and double values

```
import java.util.ArrayList;

public class StackGenerics <T> {
    ArrayList<T> stack;

    StackGenerics() {
        stack = new ArrayList<>();
    }
    void push (T i) {
        stack.add(i);
    }
    T pop() {
        T val = stack.remove(stack.size() - 1);
        return val;
    }
    void display() {
        for(int i=0; i< stack.size(); i++) {
            System.out.println(stack.get(i) + " ");
        }
    }

    public static void main (String[] args) {
        StackGenerics<Integer> s = new StackGenerics();
        for(int i=1; i<=5; i++) {
            s.push(i);
        }
        s.display();
        System.out.println();
```

```
for(int i=1; i<=5; i++) {
    System.out.println("popped : " + s.pop());
}
StackGenerics<Double> d = new StackGenerics<>();
for(int i=1; i<=5; i++) {
    d.push((double) i);
}
d.display();
System.out.println();
for(int i=1; i<=5; i++) {
    System.out.println("popped : " + d.pop());
}
}
}
```

output:

```
1  2  3  4  5
Popped : 5
popped : 4
popped : 3
popped : 2
popped : 1
1.0  2.0  3.0  4.0  5.0
Popped : 5.0
popped : 4.0
popped : 3.0
popped : 2.0
popped : 1.0
```

16/11/2024

Lab 6 Strings

① BMSCE
BMSCE

② 3
3
Roll no 10 is present

③ Dimensions are 10.0 by 14.0 by 12.0
box: Dimensions are 10.0 by 14.0 by 12.0

④ bmsce

⑤ 65
66
67
Welcome to bmsce college

⑥ Bmsce equals Bmsce → true
Bmsce equals college → false
Bmsce equalsIgnoreCase BMSCE → true

⑦ substring is matched
s1 = "Bmsce college"
s2 = "welcome to Bmsce college of Engineering"

⑧ true
false

⑨ false
true

(10) Hello equals Hello → true
Hello == Hello → false

(11) The names in alphabetical order are:
apple
ball
cat
iron
watch

(12) Sorted Numbers (Ascending order) : [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

(13) Thwas was a test . Thwas was too

(14) hello world

(15) commege

(16) Hello friends

(17) student 1
name : prateek
Reg no : 123
Semester : 3
CGPA : 8.87

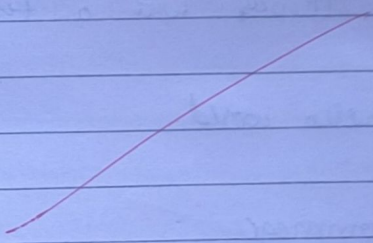
student 2
name : Houla
Reg no : 124
Semester : 4
CGPA : 9.05

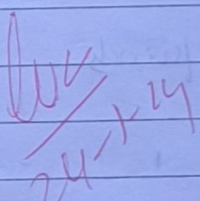
(18) charAt 3 is 'A'
reverse of SAS is SAS

(19) Eagle is flying
Eagle makes a sound

Hawk is moving
Hawk is making a sound

(20) circle - Area : 78.5398 perimeter - 31.4259
Triangle - Area : 5.0 perimeter : 12.0





Lab program 6

Create package CIE which has two classes : student and internals. The class Student has members like usn, name, sem. The class internals derived from student has an array that stores the internal marks stored in 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 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 student in all five courses

```
// Student.java
package CIE;

import java.util.Scanner;

public class Student {
    protected String usn = new String();
    protected String name = new String();
    protected int sem;

    public void inputStudentDetails() {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter usn: ");
        usn = sca sc.next();
        System.out.print("Enter name: ");
        name = sc.next();
        System.out.print("Enter semester: ");
        sem = sca sc.nextInt();
    }
```

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

// Internals.java
package CIE;

import java.util.Scanner;

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

    public Internals() { }

    public void inputCIEmarks() {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter internal
            marks for " + name);
        for(int i=0; i<5; i++) {
            System.out.print("subject " + (i+1) +
                "marks:");
            marks[i] = sc.nextInt();
        }
    }
}
```

```
//Externals.java
package SEE;

import CIE.Internals;

import java.util.Scanner;

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

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

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

        public void calculateFinalMarks() {
            for(int i=0; i<5; i++)
                finalMarks[i] = marks[i]/2 + super.marks[i];
        }
```

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

// main.java

import SEE.Externals;

public class Main {
    public static void main(String args[]) {
        int numOfStudents = 2;
        Externals finalMarks[] = new Externals[nu
                                                st
        for(int i=0; i< numOfStudents; i++) {
            finalMarks[i] = new External();
            finalMarks[i].inputStudentDetails();
            System.out.println("enter CIE marks");
            finalMarks[i].inputCIEmarks();
            System.out.println("enter SEE marks");
            finalMarks[i].inputSEEmarks();
        }

        System.out.println("Displaying data :\n");

        for(int i=0; i< numOfStudents; i++) {
            finalMarks[i].calculateFinalMarks();
            finalMarks[i].displayFinalMarks();
        }
    }
```

Output:

Enter USN: 111
Enter Name: Santosh
Enter semester: 3
Enter CIE marks
Enter internal marks for santosh
subject 1 marks: 33
subject 2 marks: 36
subject 3 marks: 28
subject 4 marks: 31
subject 5 marks: 40
Enter SEE marks for santosh
subject 1 marks: 89
Subject 2 marks: 91
Subject 3 marks: 78
subject 4 marks: 84
Subject 5 marks: 90
Enter USN: 112
Enter Name: Sini
Enter semester: 3
Enter CIE marks:
Enter internal marks for Sini
Subject 1 marks: 33
subject 2 marks: 28
Subject 3 marks: 33
Subjd 4 marks: 27
subject 5 marks: 31
Enter SEE marks
Enter SEE marks for sini
Subject 1 marks: 84
subject 2 marks: 85
Subject 3 marks: 89
Subject 4 marks: 92
Subject 5 marks: 97

Displaying data:

USN : 111

Name:- santosh

semester : 3

subject 1 : 77

subject 2 : 87

subject 3 : 67

subject 4 : 73

subject 5 : 85

USN: 112

Name: sivi

semester : 3

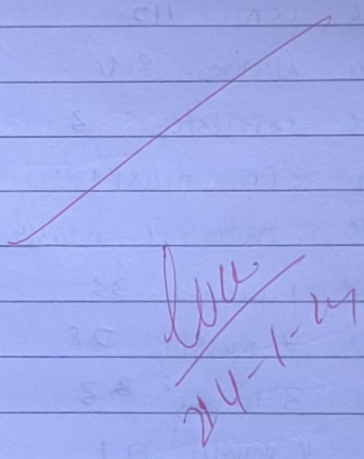
subject 1 : 75

subject 2: 76

subject 3: 72

subject 4 : 73

subject 5 : 74



Lab program 7.

→ Write a program to demonstrate 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 sons age and throws an exception if sons age >= fathers age

```
import java.util.Scanner;

class WrongAge extends Exception {

    public WrongAge(String s) {
        super(s);
    }
}
class InputScanner ~~extends~~ {
    Scanner sc;
    public InputScanner() {
        sc = new Scanner(System.in);
    }
}
class father extends InputScanner {
    private int FAge;

    public father() throws Exception {
        System.out.println("enter father's Age:");
        this.FAge = sc.nextInt();
```

```
if (this.fAge < 0) {

        throw new WrongAge("Age cannot be -ve");
}
public void display() {
        System.out.println("Father Age: " + this.fAge);
}
}
class Son extends Father
{
        private int sAge;

        public Son() throws Exception {
                System.out.println("enter son age: ");
                this.sAge = sc.nextInt();
                if (this.sAge < 0)
                        throw new WrongAge("Age cannot
                                be -ve");

                if (this.sAge >= super.fAge)
                        throw new WrongAge("Son age
                                cannot be greater than father");

        public void display() {
                super.display();
                System.out.println("Son age: " + this.sAge);
        }
}
```

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

        try {
            Son s = new Son();
            s.display();
        }
        catch (WrongAge e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

Output:

Enter Father Age: -5
Age cannot be negative -ve

Enter Father Age: 40
Enter Son Age: -5
Age cannot be -ve

Enter Father Age: 40
Enter Son Age: 45
Son age cannot be greater than Father

Enter Father Age: 40
Enter Son Age: 12
Father Age: 40
Son Age: 12

20-1-24

Lab program 8.

→ Write a program with creates two threads, one thread displaying "BMS college of Engineering" once every 10 seconds and another displaying "CSE" once every two seconds

```
class DisplayMessageThread extends Thread {
    private final String message;
    private final long interval;

    DisplayMessageThread (String message, long interval) {
        this.message = message;
        this.interval = interval;
    }
    public void run() {
        try {
            while (true) {
                System.out.println (message);
                Thread.sleep(interval);
            }
        } catch (InterruptedException e) {
            System.out.println(Thread.currentThread().
                getName() + " interrupted.");
        }
    }
}

public class TwoThreadDemo {
    public static void main (String[] args) {
        DisplayMessageThread t1 = new
            DisplayMessageThread(" BMS College of Engineering", 10000);
        DisplayMessageThread t2 = new
            DisplayMessageThread(" CSE ", 2000);
```

```
t1.setName("Thread 1");
t2.setName("Thread 2");

t1.start();
t2.start();

try {
    Thread.sleep(20000);
} catch (InterruptedException e) {
    System.out.println("Main thread interrupted.");
}

t1.interrupt();
t2.interrupt();

System.out.println("Main thread exiting.");
}
}
```

Output:-

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
CCE
CSE
BMS College of Engineering
Thread 1 interrupted.
Thread 2 interrupted.
Main thread existing.

6.2.24

13/02/24 Lab program 10

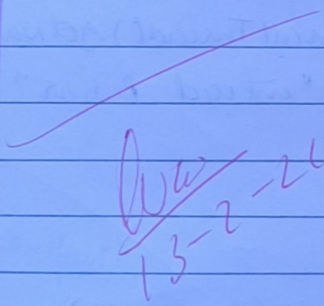
→ Implementation of a producer and consumer

```
class Q {
int n;
boolean valueSet = false;
synchronized int get() {
  while (!valueSet)
    try {
      System.out.println("\tConsumer 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 put: 1
Got: 1
Put: 2
Got: 2
Put: 3
Got: 3
put: 4
Got: 4
Put: 5
Got: 5.



→ Program to demonstrate deadlock

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

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

output:

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

Lab program 9 :-

→ Write a program that creates a user interface to perform integer division. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 & 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 NumberFormatException. If Num2 were zero, the program would throw an ArithmeticException display the exception in a message dialog box.

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

class SwingDemo {
    SwingDemo() {
        JFrame jfrm = new JFrame("Divider App");
        jfrm.setSize(275, 150);
        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();

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

ActionListener l = 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("\n A = " + a);
            blab.setText("\n B = " + b);
            anslab.setText("\n Ans = " + ans);
            ess.setText("");
        } catch (NumberFormatException e) {
            displayErrorMessage("enter only integers");
        } catch (ArithmeticException e) {
            displayErrorMessage("B should be non zero");
        }
    }
}
```

```
private void displayErrorMessage(String message) {
        alab.setText("");
        blab.setText("");
        amlab.setText("");
        err.setText(message);
    }
};
button.addActionListener(calculateListener);

jfrm.setVisible(true);
}
public static void main(String args[]) {

    SwingUtilities.invokeLater(new Runnable() {
        public void run() {
            new SwingDemo();
        }
    });
}
}
```

Output:-

Enter the divider and divident

6 3

Calculate A = 6 B = 3 Ans = 2

Functions used:-

1) JFrame :- It is a top level container in Java swing that represents a window with a title bar, border and optional menubar.

2) setSize :- It is used to set size of the frame

3) setLayout :- This line sets the layout manager for the frame to FlowLayout, which arranges components from left to right in a flow like manner

4) add :- This line adds the user label to the frame

5) setText :- This lines sets the text of the 'n' label to display the value of 'n'

6) setVisible :- This line makes the frame visible

7) ~~invokeLater :- This line schedules a job for the event-dispatching~~

8) invokeLater :- To perform task asynchronously in the ~~GUI~~ event dispatcher thread.

20/2/2024

1) Quadratic

```
import java.util.*;
class Quadratic {
   int a,b,c;
   double r1,r2,d;

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

   void compute() {
      while(a == 0) {
         System.out.println("Not a quadratic equation :(");;
         System.out.println("Enter a non zero : ");
         Scanner sc = new Scanner(System.in);
         a = sc.nextInt();
      }

      d = (b*b) - (4*a*c);
      if(d == 0) {
```

```
            r1 = (-b) / (2*a);
            System.out.println("Roots are real and equal");
            System.out.println("Root1 = Root2 = " + r1);
        }
        else if(d > 0) {
            r1 = ((-b) + Math.sqrt(d)) / (double)(2*a);
            r2 = ((-b) - Math.sqrt(d)) / (double)(2*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*a);
            r2 = (Math.sqrt(-d)) / (2*a);
            System.out.println("Root1 = "+r1+"+i"+r2);
            System.out.println("Root2 = "+r2+"-i"+r2);
        }
    }
}
class QuadraticMain {
    public static void main(String[] args) {
        Quadratic q = new Quadratic();
        q.getd();
        q.compute();
    }
}
```

2) Student CGPA calculator

```
import java.util.Scanner;


class Subject {
   int subjectMarks;
   int credits;
   int grade;
}
class Student {
   Subject subject[];
   String name;
   String USN;
   double SGPA;
   Scanner sc;


   public Student() {
      subject = new Subject[8];
      for(int i = 0; i < 8; i++) {
         subject[i] = new Subject();
      }
   }


   void getStudentDetails() {
      sc = new Scanner(System.in);
      System.out.print("Enter name : ");
```

```
    this.name = sc.next();

    System.out.print("Enter your USN : ");
    this.USN = sc.next();
}

void getMarks() {
    sc = new Scanner(System.in);
    for(int i = 0; i < 8; i++) {
        System.out.print("Enter " + (i+1) + " subject marks : ");
        subject[i].subjectMarks = sc.nextInt();

        System.out.print("Enter number of credits : ");
        subject[i].credits = sc.nextInt();

        subject[i].grade = (subject[i].subjectMarks/10) + 1;

        if(subject[i].grade == 11) subject[i].grade = 10;

        if(subject[i].grade <= 4) subject[i].grade = 0;
    }
}

void computeSGPA() {
    int totalCredits = 0;
    for(int i = 0; i < 8; i++) {
```

```
            totalCredits += subject[i].credits;
        }

        int totalGradeAndCredit = 0;
        for(int i = 0; i < 8; i++) {
            totalGradeAndCredit += (subject[i].credits * subject[i].grade);
        }

        this.SGPA = ((float)totalGradeAndCredit/totalCredits);

        System.out.println("SGPA of the student is : " + this.SGPA);
    }
}


class studentMain {
    public static void main(String[] args) {
        Student s1 = new Student();
        s1.getStudentDetails();
        s1.getMarks();
        s1.computeSGPA();
    }
}
```

3) Book

```
import java.util.*;
```

```
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 void set(String name, String author, int price, int numPages) {
      this.name = name;
      this.author = author;
      this.price = price;
      this.numPages = numPages;
   }

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

```
    }

    public String getName() {
        return this.name;
    }
    public String getAuthor() {
        return this.author;
    }
    public int getPrice(){
        return this.price;
    }
    public int getNumberOfPages() {
        return this.numPages;
    }
}

public class Main3 {
   public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of books : ");
        int n = sc.nextInt();

        Book[] b = new Book[n];
        for(int i = 0; i < n; i++) {
            System.out.print("Enter " + (i+1) + " book name, author name, price and number of
pages : ");
```

```
            String name = sc.next();
            String author = sc.next();
            int price = sc.nextInt();
            int numPages = sc.nextInt();

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

        for(int i = 0; i < n; i++) {
            System.out.println(b[i].toString());
        }
    }

}
```

4) Abstract Demo of class Shape

```
import java.util.*;

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

abstract class Shape extends InputScanner {
```

```
    double a, b;
    abstract void getInput();
    abstract void displayArea();
}

class Rectangle extends Shape {

    @Override
    void getInput() {
        System.out.print("Enter length and breath of a Rectangle : ");
        this.a = sc.nextDouble();
        this.b = sc.nextDouble();
    }

    @Override
    void displayArea() {
        System.out.println("The area of the Rectangle is : " + (this.a * this.b));
    }
}
class Triangle extends Shape {

    @Override
    void getInput() {
        System.out.print("Enter Height and Base of a Triangle : ");
        this.a = sc.nextDouble();
        this.b = sc.nextDouble();
```

```
    }

    @Override
    void displayArea() {
        System.out.println("The area of the Triangle is : " + ((this.a * this.b)/2));
    }
}
class Circle extends Shape {

    @Override
    void getInput() {
        System.out.print("Enter the radius of the Circle : ");
        this.a = sc.nextDouble();
    }

    @Override
    void displayArea() {
        System.out.println("The area of the Circle is : " + (Math.PI*this.a*this.a));
    }
}

public class AbstractDemo {
    public static void main(String[] args) {
        Shape s;
        s = new Circle();
        s.getInput();
```

```
        s.displayArea();

        s = new Rectangle();
        s.getInput();
        s.displayArea();

        s = new Triangle();
        s.getInput();
        s.displayArea();
    }
}
```

5) Bank

```
import java.util.Scanner;

abstract class Account {
    String name;
    int accno;
    String type;

    abstract void deposit(double amount);
    abstract void display();
    abstract void withdraw(double amount);
    abstract void menu();

}
```

```
class Current extends Account {
    double balance;
    private static int MIN_BALANCE = 500;
    private static int SER_CHARGE = 10;



    public Current(String name, int accno, String type) {
        this.name = name;
        this.accno = accno;
        this.type = type;
    }

    @Override
    void deposit(double amount) {

        this.balance += amount;
        System.out.println("amount deposited ");

                if(this.balance < MIN_BALANCE) {
            System.out.println("Penalty");
            this.balance -= SER_CHARGE;
        }
    }

    @Override
    void display() {
```

```
    System.out.println("Name : " + this.name);
    System.out.println("Account number : " + this.accno);
    System.out.println("Balance : " + this.balance);
}

@Override
void withdraw(double amount) {
    if(this.balance < amount) {
        System.out.println("insufficient balance");
    }
    else {
        this.balance -= amount;
    }
}

@Override
void menu() {
    System.out.println("---MENU---");
    int choice;
    double amount;
    do {
        System.out.println("1. deposit 2. Withdraw 3. details 4. Exit");
        Scanner sc = new Scanner(System.in);
        choice = sc.nextInt();
            switch(choice) {
                case 1 : System.out.print("Enter the deposit amount : ");
```

```
                    amount = sc.nextDouble();
                    this.deposit(amount);
                    break;
                case 2 : System.out.println("Enter withdrawal amount : ");;
                    amount = sc.nextDouble();
                    this.withdraw(amount);
                    break;
                case 3 : this.display();
                    break;
              case 4 : return;

        }
        }while(choice != 4);
    }

}
class Savings extends Account {
    int balance;
    final float interest = 5f;

            public Savings(String name, int accno, String type) {
            this.name = name;
            this.accno = accno;
                        this.type = type;
            }
    @Override
```

```
void deposit(double amount) {

   float interestAmount = (float) (amount*(this.interest/100));
   System.out.println("Inetrest of : " + interestAmount + " is added");
   this.balance += amount + interestAmount;
   System.out.println("amount deposited ");
}

@Override
void display() {
   System.out.println("Name : " + this.name);
   System.out.println("Account number : " + this.accno);
   System.out.println("Balance : " + this.balance);
}

@Override
void withdraw(double amount) {
   if(this.balance < amount) {
      System.out.println("insufficient balance");
   }
   else {
      this.balance -= amount;
   }
}

@Override
```

```
    void menu() {
        System.out.println("---MENU---");
        int choice;
        double amount;
        do {
            System.out.println("1. deposit 2. Withdraw 3. details 4. Exit");
            Scanner sc = new Scanner(System.in);
            choice = sc.nextInt();
                switch(choice) {
                    case 1 : System.out.print("Enter the deposit amount : ");
                        amount = sc.nextDouble();
                        this.deposit(amount);
                        break;
                    case 2 : System.out.println("Enter withdrawal amount : ");;
                        amount = sc.nextDouble();
                        this.withdraw(amount);
                        break;
                    case 3 : this.display();
                        break;
                 case 4 : return;

        }
        }while(choice != 4);
    }

}
```

```
public class Bank {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.println("Enter name, accno, type : ");
      String name = sc.next();
      int accno = sc.nextInt();
      String type = sc.next();

      if(type.equals("current")) {
         Account c = new Current(name, accno, type);
         c.menu();
      }
      else {
         Account s = new Savings(name, accno, type);
         s.menu();
      }
   }
}
```

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

the two packages in a file that declares the final marks of n students in all five courses.

```
package CIE;

import java.util.Scanner;

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

   public Internals() {
      // Constructor for Internals
   }

   public void inputCIEmarks() {
      Scanner scanner = new Scanner(System.in);
      System.out.println("Enter Internal Marks for " + super.name);
      for (int i = 0; i < 5; i++) {
         System.out.print("Subject " + (i + 1) + " marks: ");
         this.marks[i] = scanner.nextInt();
      }
   }
}
package CIE;
import java.util.*;

public class Student {
```

```
    protected String usn = new String();
    protected String name = new String();
    protected int sem;

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

    public void displayStudentDetails() {
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        System.out.println("Semester: " + sem);
    }
}
package SEE;
import CIE.Internals;


import java.util.Scanner;
```

```
public class External extends Internals {
    protected int marks[];
    protected int finalMarks[];

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

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

    public void calculateFinalMarks() {
        for (int i = 0; i < 5; i++)
            finalMarks[i] = marks[i] / 2 + super.marks[i];
    }

    public void displayFinalMarks() {
        displayStudentDetails();
        for (int i = 0; i < 5; i++)
```

```
            System.out.println("Subject " + (i + 1) + ": " + finalMarks[i]);
        }
    }
    package SEE;
    import CIE.Internals;



    import java.util.Scanner;

    public class External extends Internals {
        protected int marks[];
        protected int finalMarks[];

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

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

```
    public void calculateFinalMarks() {
        for (int i = 0; i < 5; i++)
            finalMarks[i] = marks[i] / 2 + super.marks[i];
    }

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

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.

```
import java.util.Scanner;

class WrongAge extends Exception
{
    public WrongAge(String message)
    {
        super(message);
```

```
    }
}

class InputScanner
{
    protected Scanner s;

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

class Father extends InputScanner
{
    protected int fatherAge;

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

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

```
    }

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

class Son extends Father
{
    private int sonAge;

    public Son() throws WrongAge
    {
        super();

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

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

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.

```
class DisplayMessageThread extends Thread {
   private final String message;
   private final long interval; // in milliseconds

   DisplayMessageThread(String message, long interval) {
      this.message = message;
      this.interval = interval;
   }

   public void run() {
      try {
         while (true) {
            System.out.println(message);
            Thread.sleep(interval);
         }
      } catch (InterruptedException e) {
         System.out.println(Thread.currentThread().getName() + " interrupted.");
      }
   }
}

public class TwoThreadDemo {
   public static void main(String[] args) {
      DisplayMessageThread thread1 = new DisplayMessageThread("BMS College of Engineering", 10000); // 10 seconds
```

```
        DisplayMessageThread thread2 = new DisplayMessageThread("CSE", 2000); // 2
seconds


        thread1.setName("Thread 1");
        thread2.setName("Thread 2");


        thread1.start();
        thread2.start();


        try {
            // Let the threads run for a while
            Thread.sleep(30000); // Let the program run for 30 seconds
        } catch (InterruptedException e) {
            System.out.println("Main thread interrupted.");
        }


        // Interrupt both threads to stop them
        thread1.interrupt();
        thread2.interrupt();


        System.out.println("Main thread exiting.");
    }
}
```

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

class UserInterface {
   UserInterface() {
      // create JFrame container
      JFrame jfrm = new JFrame("Divider App");
      jfrm.setSize(275, 150);
      jfrm.setLayout(new FlowLayout());
      // to terminate on close
      jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

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

      // add text field for both numbers
      JTextField ajtf = new JTextField(8);
      JTextField bjtf = new JTextField(8);

      // calc button
      JButton button = new JButton("Calculate");

      // labels
      JLabel err = new JLabel();
```

```
JLabel alab = new JLabel();

JLabel blab = new JLabel();

JLabel anslab = new JLabel();


// add in order :)

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 calculateListener = 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("\nA = " + a);

         blab.setText("\nB = " + b);
```

```
            anslab.setText("\nAns = " + ans);
            err.setText(""); // Clear any previous error message
         } catch (NumberFormatException e) {
            displayErrorMessage("Enter Only Integers!");
         } catch (ArithmeticException e) {
            displayErrorMessage("B should be non-zero!");
         }
      }

      private void displayErrorMessage(String message) {
         alab.setText("");
         blab.setText("");
         anslab.setText("");
         err.setText(message);
      }
   };

   button.addActionListener(calculateListener);

   // display frame
   jfrm.setVisible(true);
}

public static void main(String args[]) {
   // create frame on event dispatching thread
   SwingUtilities.invokeLater(new Runnable() {
```

```
        public void run() {
            new UserInterface();
        }
      });
   }
}
```

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<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.");

}

}
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("M
ainThread");

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();

}

}
```