B. M. S. COLLEGE OF ENGINEERING

(AUTONOMOUS COLLEGE UNDER VTU, BELGAUM)
BANGALORE – 560019

2022-23

LAB RECORD

OBJECT ORIENTED JAVA PROGRAMMING (23CS3PCOOJ)

Submitted by :

NAME : Shreyas Rao M
USN : 1BM22CS272
SEMESTER: III
SECTION : E

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

INDEX

NAME: Shreyas Rao M STD.: III SEC.: E ROLL NO.: 1BM22CS272 SUB.: Object oriented Java Programming

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

1 import java.util.*;
class demo
{
    public static void main(String[] args)
    {
        System.out.println("Hello, World!");
    }
}
Output: Hello, World!

2 Addition of two numbers
import java.util.*;
class Sum
{
    public static void main(String[] args)
    {
        int a=20, b=10, sum;
        sum=a+b;
        System.out.println("Sum=" + sum);
    }
}
Output: Sum=30

3 Program to check whether number is less than other numb
import java.util.*;
class Greatnum
{
    public static void main(String[] args)
    {
        int a=10, b=15;
        if (a>b)
            System.out.println("Greatest =" +a);

```
        if (b>a)
            System.out.println("Greatest = " +b);
        if (a==b)
            System.out.println("Both are equal");
        }
}
```

Output : Greatest = 15

4 Calculator

```
import java.util.*;
class Calculator
{
    public static void main (String[] args)
    {
        float a=15, b=10;
        float sum, dif, prod, quo, rem;
        sum = a+b;
        dif = a-b;
        prod = a*b;
        quo = a/b;
        rem = a%b;
        System.out.println("Sum = "+sum+"\n Difference = "+dif
                           "\n Product = " +prod + "\n Quotient = "+quo
                           "\n Remainder = " +rem);
    }
}
```

Output: Sum = 25
Difference = 5
Product = 150
Quotient = 1.5
Remainder = 5

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

```
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 the coefficients of a,b,c");
        a=s.nextInt();
        b=s.nextInt();
        c=s.nextInt();
    }
    void compute()
    {
        while(a==0)
        {
            System.out.println("Not a quadratic 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)
        {
            r1=(-b)/(2*a);
            System.out.println("Roots are real and equal");
            System.out.println("Root1="+r1+"Root2="+r2);
        }
```

```
        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("Root2 = "+r1+"-i"+r2);
        }
    }
}
class QuadraticMain
{
    public static void main(String args[])
    {
        Quadratic q = new Quadratic();
        q.getd();
        q.compute();
        System.out.println("Shreyas Rao M - 1BM22CS222");
    }
}
```

Output

Enter the coefficients of a,b,c
1
2
1
Roots are real and equal
Root 1 = Root 2 = -1.0
Shreyas Rao M - 1BM22CS272

Enter the coefficients of a,b,c
1
3
2
Roots are real and distinct
Root 1 = -1.0 Root 2 = -2.0
Shreyas Rao M - 1BM22CS272

Enter the coefficients of a,b,c
2
1
3
Roots are imaginary
Root 1 = 0.0 + i1.1989578808281798
Root 2 = 0.0 - i1.1989578808281798
Shreyas Rao M - 1BM22CS272

10

Sun 12/12/23

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

$c \leq$ Course credits $\times$ Grade points

$\leq$ Course credits

```
import java.util.*;
class Subject {
    int subjectMarks;
    int credits;
    int grade;
}

class Student {
    Subject subject[];
    String name;
    String usn;
    double cgpa;
    Scanner sc;

    Student() {
        int i;
        subject = new Subject[8];
        sc = new Scanner(System.in);
        for (i=0; i<8; i++)
            subject[i] = new Subject();
    }

    void getstudentdetails() {
        System.out.println("Enter your name :");
        this.name = sc.next();
        System.out.println("Enter your USN:");
        this.usn = sc.next()
```

```
    void getMarks(){
        for(int i=0; i<8; i++){
            System.out.println("Enter marks for subject"+(i+1)+":");
            subject[i].subjectMarks = sc.nextInt();
            System.out.println("Enter credits for subject "+(i+1)+":");
            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 effectiveScore = 0;
        int totalCredits = 0;
        for(int i=0; i<8; i++){
            effectiveScore += (subject[i].grade * subject[i].credits);
            totalCredits += subject[i].credits;
        }
        sgpa = (double) effectiveScore/(double) totalCredits;
    }
}

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

Output:

Enter your name:
Shreyas Rao M
Enter your USN:
1BM22CS272
Enter marks for subject 1:
99
Enter credits for subject1:
4
Enter marks for subject 2:
95
Enter credits for subject2:
4
Enter marks for subject 3:
96
Enter credits for subject 3:
3
Enter marks for subject 4:
92
Enter credits for subject4:
3
Enter marks for subject 5:
89
Enter credits for subject 5:
3
Enter marks for subject 6:
99
Enter credits for subject6:
1
Enter ~~credits~~ marks for subject 7:
100
Enter credits for subject 7:
1

Enter marks for subject 8:
96
Enter credits for subject8
1
Name = Shreyas Rao M
USN = 1BM22CS272
SGPA = 9.85

(10)

19/12/23

26-12-24

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.

```
import java.util.Scanner;

class Books {
    String name, author;
    int price, numPages;

    Books (String name, String author, int price, int numPages)
    {
        this.name = name;
        this.author = author;
        this.price = price;
        this.numPages = numPages;
    }
    public String toString()
    {
        String name, author, price, numPages;
        name = "Book name :" + this.name + "\n";
        author = "Author name :" + this.author + "\n";
        price = "Price :" + this.price + "\n";
        return name + author + price + numPages;
    }
```

```
class LabProg3 {
    public static void main (String [] args)
    {
        Scanner sc = new Scanner (System.in);
        int n;
        String name, author;
        int price, numPage;
        System.out.println ("Enter the number of books :");
        n = sc.nextInt();
        Books b[] = new Books[n];
        System.out.println("Enter name, author, price and number
                                of pages :");
        for (int i=0; i<n; i++) {
            name=sc.next();
            author =sc.next();
            price = sc.nextInt();
            numPages = sc.nextInt();
            b[i] = new Books (name, author, price, numPages);
        }
        System.out.println ("Book details : ");
        for (int i=0; i<n; i++)
        {
            System.out.println (b[i].toString());
        }
    }
}
```

Output:
Enter the numbes of books:
2
Enter name, author, price and number of pages:
IntroTo C
Reema
500
250
IntroToJava
Kevin
750
250
Book details:
Book name : IntroTo C
Author name : Reema
Price : 500
Number of pages : 250

Book name : IntroToJava
Author name : Kevin
Price : 750
Number of pages : 250

Shreyas Rao M
1BM22CS272
28/12/23

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 extend class Shape. Each of the shape contain only the method printArea that prints the area of the given shape

```
import java.util.*;

class InputScanner
{
        Scanner sc;
        InputScanner()
        {
            sc = new Scanner(System.in);
        }
}
abstract class Shape extends InputScanner
{
        double a;
        double b;
        abstract void getInput();
        abstract void displayArea();
}
class Rectangle extends Shape
{
    void getInput()
    {
        System.out.println("Enter the length and breadth:");
        a = sc.nextDouble();
        b = sc.nextDouble();
    }
```

```
    void displayArea()
    {
        System.out.println("Area of rectangle is:"+(a*b));
    }
}

class Triangle extends Shape
{
    void getInput()
    {
        System.out.println("Enter length and height:");
        a = sc.nextDouble();
        b = sc.nextDouble();
    }
    void displayArea()
    {
        System.out.println("Area of triangle is:"+(a*b/2));
    }
}
class Circle extends Shape
{
    void getInput()
    {
        System.out.println("Enter the radius:");
        a=sc.nextDouble();
    }
    void displayArea()
    {
        System.out.println("Area of circle:"+(a*a*3.14));
    }
}
```

```
class ShapeMain
{
    public static void main (String[] args)
    {
        Rectangle r = new Rectangle();
        Triangle t = new Triangle();
        Circle c = new Circle();
        r.getInput();
        r.displayArea();
        t.getInput();
        t.displayArea();
        c.getInput();
        c.displayArea();
    }
}
```

Output:

```
Enter the length and breadth:
10
5
Area of rectangle is : 50.0
Enter the length and height:
10
4
Area of triangle is : 20.0
Enter the radius:
5
Area of circle : 78.5
```

Shreyas Rao M
1BM22CS272

02/10/24

Lab 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. Create a class Account that stores customer name, 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 requirement.

```
import java.util.Scanner;

class account{
    String name;
    int accno;
    String type;
    double balance;

    account (String name, int accno, String type, double balance)
    {
        this.name = name;
        this.accno = accno;
        this.type = type;
        this.balance = balance;
    }
    void deposit (double amount)
    {
        balance += amount;
    }
    void withdraw(double amount)
    {
        if ((balance - amount) >= 0)
        {
            balance -= amount;
        }
```

```
        else
        {
            System.out.println("Insufficient balance, can't wit
        }
    }
    void display()
    {
        System.out.println("Name:"+name+"\nAccno:"+accno
                +"\nType:"+type+"\nBalance:"+ba
    }
}

class savAcct extends account{
    private static double rate=5;
    savAcct(String name, int accno, double balance)
    {
        super(name, accno, "Savings", balance);
    }
    void interest()
    {
        balance+=balance*(rate)/100;
        System.out.println("Balance:"+balance);
    }
}

class currAcct extends account{
    private double minBal=500;
    private double serviceCharges=50;

    currAcct(String name, int accno, double balance)
    {
        super(name, accno, "Current", balance);
    }
```

```
void checkmin ()
{
    if (balance < minBal)
    {
        System.out.println("Balance is less than min balance
                            service charges imposed :" +serviceCharges);
        balance -= serviceCharges;
        System.out.println("Balance is :" +balance);
    }
}
}
class Bank
{
    public static void main(String args[])
    {
        Scanner s=new Scanner(System.in);
        System.out.println("Enter the name, type(current/savings),
                            account number, initial balance:");
        String name=s.next();
        String type=s.next();
        int accno=s.nextInt();
        double balance=s.nextDouble();
        int ch;
        double amount1,amount2;
        account acc=new account(name,accno,type,balance);
        saveAcct sa=new saveAcct(name,accno,balance);
        curAcct ca=new curAcct(name,accno,balance);

        while(true){
            if(acc.type.equals("savings")){
                System.out.println("Menu\n1.deposit 2.withdraw
                                    3.compute interest 4.display");
                System.out.println("Enter the choice:");
                ch=s.nextInt();
```

```
switch(ch){
    case 1:
        System.out.println("Enter the amount:");
        amount1 = sc.nextInt();
        ca.deposit(amount1);
        break;
    case 2:
        System.out.println("Enter the amount:");
        amount2 = s.nextInt();
        sa.withdraw(amount2);
        break;
    case 3:
        sa.interest();
        break;
    case 4:
        sa.display();
        break;
    case 5:
        System.exit(0);
    default:
        System.out.println("Invalid input");
        break;
    }
}
else {
    System.out.println("Menu 1. Deposit 2. Withdraw
    System.out.println("Enter the choice:");
    ch = c.nextInt();
    switch(ch)
    {
        case 1:
            System.out.println("Enter the amount:");
            amount1 = s.nextInt();
            ca.deposit(amount1);
```

```
        case 2:
            System.out.println("Enter the amount:");
            amount = s.nextInt();
            ca.withdraw(amount2);
            ca.checkmin();
            break;
        case 3:
            ca.display();
            break;
        case 4:
            System.exit(0);
        default:
            System.out.println("Invalid input");
            break;
        }
    }
}
```

Output:

```
Enter the name, type (current/saving), account number, initial balance
Shreyas
saving
123
50000

Menu
1. Deposit 2. Withdraw 3. Compute interest 4. Display
Enter the choice:
1
Enter the amount:
5000
```

Menu:
1. Deposit 2. Withdraw 3. Compute Interest 4. Display
2
Enter the amount:
500

Menu:
1. Deposit 2. Withdraw 3. Compute Interest 4. Display
3
Balance: 57225.0

Menu:
1. Deposit 2. Withdraw 3. Compute Interest 4. Display
Enter the choice:
4
Name: Shreyas
Accno: 123
Type: Savings
Balance: 57225.0

Enter the ~~choice~~ name, type Current/saving), account
initial balance:
Shreyas S
current
101
500000

Menu
1. deposit 2. withdraw 3. display
Enter the choice
1
Enter the amount:
5000

Menu
1. deposit 2-withdraw 3-display
Enter the choice:
2
Enter the amount:
500

Menu
1-deposit 2-withdraw 3-display
Enter the choice:
Name: Shreyas
Acno: 101
Type: Current
Balance: 504500.0

USN=1BM22CS272

6 Write a java program to create a generic class stack which holds 5 integer and 5 double values

```
class GenericStack<T>{
    private Object[] stackArray;
    private int top=-1;
    private static final int N=5;

    public GenericStack()
    {
        stackArray = new object[N];
    }

    public void push(T value)
    {
        if (top<N-1)
            stackArray[T+top] = value
        else
            System.out.println("Stack is full");
    }
    public T pop()
    {
        if (top>=0)
            return (T) stackArray[top--];
        else
        {
            System.out.println("Stack is empty");
            return null;
        }
    }
    public boolean isEmpty()
    { return top==-1; }
    public boolean isFull()
    { return top==N-1; }
}
```

```
class main
{
    public static void main (String args[])
    {
        GenericStack<Integer> integerstack = new GenericStack<>();
        GenericStack<Double> doublestack = new GenericStack<>();

        for (int i=1; i<=5; i++){
            integerStack.push(i);
        }
        for (double i=1.0; i<=5.0; i++){
            doubleStack.push(i);
        }

        System.out.println("Popped integers from the stack:");
        while (!integerStack.isEmpty()){
            System.out.println(integerStack.pop());
        }

        System.out.println("Popped double from the stack:");
        while (!doublestack.isEmpty()){
            System.out.println(doublestack.pop());
        }
    }
}
```

Output

```
Popped integers from the stack:
5
4
3
2
1
```

Popped doubles from the stack:
5.0
4.0
3.0
2.0
1.0

String

Demonstrate string length, string literal, string concat

```
public class string1{
    public static void main(String args[])
    {
        System.out.println("Demonstrating string length:")
        String a = "Hello";
        System.out.println(a.length());

        System.out.println("Str concat:");
        String age = "9";
        String msg = "He is " + age + " years old.";
        System.out.println(msg);

        System.out.println("Demonstrate literals");
        System.out.println("abc".length());
    }
}
```

Output:
Demonstrating string length:
5
Str concat:
He is 9 years old
Demonstrate literals
3

Use getchars() to extract Bmsce from "Welcome to Bmsce college"

```
public class string2 {
    public static void main (String args[]) {
        String s = "Welcome to BMSCE college";
        int start = 10;
        int end = 16;
        char buf[] = new char[end - start];
        s.getChars(start, end, buf, 0);
        System.out.println(buf);
}
}
```

Output

BMSCE

Shreyas Rao M
1BM22CS272

6. Write a Java program to create an abstract class Shape with abstract method calculateArea() and calculatePerimeter. Create subclasses Circle and Triangle that extend the Shape class.

```
import java.lang.Math;

abstract class Shape{
    double a;
    double b;
    double c;

    abstract void calculateArea();
    abstract void calculatePerimeter();
}

class Triangle extends Shape{
    Triangle(double x, double y, double z)
    {
        a=x;
        b=y;
        c=z;
    }

    void calculateArea()
    {
        double s = (a+b+c)/2;
        System.out.println("Area = "+(Math.sqrt(s*(s-a)*(s-b)*(s-c))));
    }

    void calculatePerimeter()
    {
        System.out.println("Perimeter = "+(a+b+c));
    }
}
```

```
class Circle extends Shape{
    Circle (double r)
    {
        a=r;
    }
    void calculateArea ()
    {
        System.out.println ("Area =" + (Math.PI * a * a));
    }
     void calculatePerimeter()
    {
        System.out.println (" Perimeter = " + (2 * Math.PI * a));
    }
}

class ShapeM{
    public static void main(String [] args)
    {
        Triangle t = new Triangle (3.0,3.0, 5.0);
        Circle c = new Circle (5.0);
        t.calculateArea ();
        t.calculatePerimeter();
        c. calculateArea();
        c. calculatePerimeter();
    }
}
```

Output:

Area = 4.145280987944425
Perimeter = 11.0
Area = 78.53981633974483
Perimeter = 31.41592653589793

Shreyas Rao M
1BM22CS272

16/01/24

7 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. Import the two packages

```
//Class Student
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 s = new Scanner(System.in);
        System.out.println("Enter USN:");
        usn = s.next();
        System.out.println("Enter Name:");
        name = s.next();
        System.out.println("Enter Semester:");
        sem = s.nextInt();
    }
```

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

// Internals.java

```
package CIE;
import java.util.Scanner;

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

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

```
// Externals.java
package SEE;

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 s = 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] = s.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]);
    }
```

```
}
```

Pkgmain.java

```
import SEE.Externals;

public class Pkgmain{
    public static void main (String args[]){
        int numOfStudents =2;
        Externals finalMarks[]=new Externals[numOfStudents];

        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]. input SEE marks();
        }
        System.out.println("Displaying data:\n");

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

Output:

Enter USN: 101
Enter Name: Shreyas
Enter Semester: 3
Enter CIE marks
Enter Internal Marks for Shreyas
Subject 1 marks: 45
Subject 2 marks: 47
Subject 3 marks: 48
Subject 4 marks: 49
Subject 5 marks: 50
Enter SEE mark for Shreyas
Subject 1 marks: 99
Subject 2 marks: 99
Subject 3 marks: 97
Subject 4 marks: 98
Subject 5 marks: 96
Enter USN: 102
Enter Name: kevin
Enter Semester: 3
Enter CIE marks:
Enter Internal marks for kevin
Subject 1 marks: 9
Subject 2 marks: 6
Subject 3 marks 4
Subject 4 marks 5
Subject 5 marks 7
Enter SEE marks
Enter SEE marks for kevin
Subject 1 marks: 12
Subject 2 marks: 14
Subject 3 marks: 15
Subject 4 marks: 16
Subject 5 marks: 17

Displaying data:

USN : 101
Name: Shreyas
Semester : 3
Subject 1 : 94
Subject 2 : 96
Subject 3 : 96
Subject 4 : 98
Subject 5 : 98

USN : 102
Name : kevin
Semester : 3
Subject 1 : 15
Subject 2 : 13
Subject 3 : 11
Subject 4 : 13
Subject #5 : 15

Shreyas Room
1BM22CS272

23/01/24

30/1/24

5 Write a program that demonstrates handling exceptions in inheritance tree. Create a base class called "Father" and derived class "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.*;
class WrongAge extends Exception{
        public WrongAge(String s){
                super(s);
        }
}
class InputScanner{
        protected Scanner s;
        public InputScanner(){
                s=new Scanner(System.in);
        }
}
class Father extends InputScanner{
        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{
        int sonAge;
        public Son() throws WrongAge{
            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(){
            System.out.println("Son's Age " + sonAge);
        }
}
public class ExceptionDemo{
    public static void main (String[] args)
    {
        try {
            Son sn = new Son();
            sn.display();
        }
        catch(WrongAge e) {
            System.out.println("Negative error" + e);
            System.out.println("e.getMessage());
        }
}
```

Output:

Enter father's age:
41
Enter son's age:
18
Son's Age 18
Enter father's age:
41
Enter son's age:
45
Negative Error Wrong Age: Son's age cannot be greater than father's age
Son's age cannot be greater than father's age
Enter father's age
41
Enter son's age
-21
Negative Error Wrong Age: Age cannot be negativ
Age cannot be negative

Shreyas Rao M
1BM22CS272

31.01.24

Write a program which creates two threads, one thread displaying "BMS College of Engineering" once every ten seconds and another displays "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 thread1 = new
        DisplayMessageThread ("BMS College of Engineering", 10000);
        DisplayMessageThread thread2 = new
        DisplayMessageThread ("CSE", 2000);
```

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

        try {
            Thread.sleep(30000);
        } catch (InterruptedException e) {
            System.out.println("Main thread interrupted.");
        }

        thread1.interrupt();
        thread2.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
CSE
CSE
CSE
```

Main thread exiting.
Thread 1 interrupted.
Thread 2 interrupted.
Shreyas Rao M
1BM22CS272

Demonstrate Inter process Exercise

```
class Q
{
    int n;
    boolean valueSet = false;

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

    synchronized void put(int n)
    {
        while (valueSet)
            try {
                System.out.println("\n Producer waiting\n");
                wait();
            } catch(InterruptedException e)
            {
                System.out.println("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:

Press Control-C to stop.
Put: 0

Intimate Consumer

Producer waiting

Got: 0
Intimate ~~Consumer~~ Producer

Put: 1

Intimate Consumer

Consumed: 0

Producer waiting

Got: 1

Intimate Producer
Consumed: 1
Put: 2

Intimate Consumer

Producer waiting

Got : 2

Intimate Producer

Consumed : 2
Put : 3

Intimate Consumer

Producer waiting
Got : 3
Intimate Producer
Consumed : 3
Put : 4
Intimate Consumer
Producer waiting
Got : 4
Intimate Producer
Consumed : 4
Put : 5
Intimate Consumer
Producer waiting
Got : 5
Intimate Producer
Consumed. 5
Put : 6

:

Shreyas Rao M
1BM22CS272

10b

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

```
class Deadlock implements Runnable{
    A a = new A();
    B b = new B();

    Deadlock()
    {
        Thread.currentThread().setName("MainThread");
        Thread t = new Thread(this, "RacingThread");
        t.start();
        a.foo(b);
        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:

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

Sreeya Rao
1BM22CS272
13.08.24

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 & Num2 is displayed in the Result field when the divide is clicked if Num1 or Num2 were not an integer, 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 SwingDemo
{
    SwingDemo()
    {
        JFrame jfrm = new JFrame("Divides App");
        jfrm.setSize(275,150);
        jfrm.setLayout(new FlowLayout());
        jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_
                                      CLOSE);
        JLabel jlab = new JLabel("Enter the dividor
                                  and divident:");
        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);
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)
    {
        System.out.println("Action event from a textfield");
    }
};
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());
            int ans= a/b;

            alab.setText("\nA = "+a);
            blab.setText("\nB = "+b);
            anslab.setText("\nAns = "+ans);
        } Catch (NumberFormatException e)
        {   alab.setText("");
            blab.setText("");
            anslab.setText("");
            err.setText("Enter only Integers!");
```

```
        } catch (ArithmeticException e) {
            alab.setText("");
            blab.setText("");
            anslab.setText("");
            err.setText("B should be NON zero!");
        }
    }
});
jfrm.setVisible(true);
}

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

Output

Enter the divider and divident:

10 2

Calculate A=10 B=2 Ans=5

Shreyas Rao M
1BM22CS277

20.02.24

Functions

JFrame :- The javax.swing.JFrame class is a type of container which inherits the java.awt.Frame class. JFrame works like the main window where components like labels, textfields are added to create a GUI

setSize(int width, int height) - used to resize a frame using width and height parameters

setLayout( ) - method allows you to set the layout of the container. The layout manager helps to lay out the components held by this container

setDefaultCloseOperation() - methods is used to specify one of several options for the close button
JFrame.EXIT_ON_CLOSE - Exit the application

JLabel - The object of JLabel class is a component for placing text in a container. It is used to display a single line of read only text

JTextfield - The object of a JTextfield class is a text component that allows the editing of a single line text. It inherits JTextComponent class

add(frame) - adds new frame to the existing frame

ActionListener - The Java ActionListener is notified whenever you click on the button or menu item. It is notified against ActionEvent

setText() - This method substitutes new text for all or part of the text in the text field. This works only with the first line of multi-line text fields

setVisible() is a method that has return type boolean

30.08.24

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

```
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 the coefficients of a,b,c");
      a = s.nextInt();
      b = s.nextInt();
      c = s.nextInt();
   }


   void compute() {
      while (a == 0) {
         System.out.println("Not a quadratic 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) {


         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("Roo1 = " + 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.getd();

        q.compute();
        System.out.println("Shreyas Rao M-1BM22CS272");
    }
}
```

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.*;


class Subject {
   int subjectMarks;
   int credits;
   int grade;
}


class Student {
   Subject subject[];
   String name;
   String usn;
   double sgpa;
   Scanner sc;


   Student() {
      int i;
      subject = new Subject[8];
      sc = new Scanner(System.in);
      for (i = 0; i < 8; i++)
         subject[i] = new Subject();
   }


   void getstudentdetails() {
      System.out.println("Enter your name:");
      this.name = sc.next();
      System.out.println("Enter your USN:");
      this.usn = sc.next();
   }
```

```
    void getMarks() {
        for (int i = 0; i < 8; i++) {
            System.out.println("Enter marks for subject " + (i + 1) + ":");
            subject[i].subjectMarks = sc.nextInt();
            System.out.println("Enter credits for subject" + (i + 1) + ":");
            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 effectiveScore = 0;
        int totalCredits = 0;
        for (int i = 0; i < 8; i++) {
            effectiveScore += (subject[i].grade * subject[i].credits);
            totalCredits += subject[i].credits;
        }
        sgpa = (double) effectiveScore / (double) totalCredits;
    }

}

class Main {
    public static void main(String[] args) {
        Student s1 = new Student();
        s1.getstudentdetails();
```

```
        s1.getMarks();
        s1.computeSGPA();
        System.out.println("Name=" + s1.name);
        System.out.println("USN:" + s1.usn);
        System.out.println("SGPA=" + s1.sgpa);
    }
}
```

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.

```
import java.util.Scanner;


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

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

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

```
public class Mainbook
{
	public static void main(String args[])
	{
		Scanner s=new Scanner(System.in);
		int n;
		int i;
		String name;
		String author;
		int price;
		int numPages;

		System.out.println("Enter the number of books:");
		n=s.nextInt();

		Books b[];
		b=new Books[n];

		for(i=0;i<n;i++)
		{
			System.out.println("Enter the details of book" + (i+1) + ":");
			System.out.println("Enter the name of the book:");
			name=s.next();
			System.out.println("Enter the author name:");
			author=s.next();
			System.out.println("Enter the price:");
			price=s.nextInt();
			System.out.println("Enter the number of pages:");
			numPages=s.nextInt();
```

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

            System.out.println("Book Details:");
            for(i=0;i<n;i++)
            {
                System.out.println(b[i]);
            }
        }
}
```

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.

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

class InputScanner {

   Scanner sc;

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

}

abstract class Shape extends InputScanner {

   double a;

   double b;

   abstract void getInput();

   abstract void displayArea();

}

class Rectangle extends Shape {
   void getInput() {
```

```
        System.out.println("Enter the length and breadth:");
        a = sc.nextDouble();
        b = sc.nextDouble();
    }

    void displayArea() {
        System.out.println("Area of rectangle is :" + (a * b));
    }
}

class Triangle extends Shape {
    void getInput() {
        System.out.println("Enter the length and height:");
        a = sc.nextDouble();
        b = sc.nextDouble();
    }

    void displayArea() {
        System.out.println("Area of triangle is :" + (a * b * 0.5));
    }
}

class Circle extends Shape {
    void getInput() {
        System.out.println("Enter the radius:");
        a = sc.nextDouble();
    }

    void displayArea() {
        System.out.println("Area of circle is :" + (a * a * Math.PI));
    }
```

```
}


class ShapeMain {
    public static void main(String[] args) {
        Rectangle r = new Rectangle();
        Triangle t = new Triangle();
        Circle c = new Circle();
        r.getInput();
        r.displayArea();
        t.getInput();
        t.displayArea();
        c.getInput();
        c.displayArea();
    }
}
```

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.**

```
import java.util.Scanner;

class account {
   String name;
   int accno;
   String type;
   double balance;

   account(String name, int accno, String type, double balance) {
      this.name = name;
      this.accno = accno;
      this.type = type;
      this.balance = balance;
   }

   void deposit(double amount) {
```

```
        balance += amount;
    }

    void withdraw(double amount) {
        if ((balance - amount) >= 0) {
            balance -= amount;
        } else {
            System.out.println("Insufficient balance,cant withdraw");
        }
    }

    void display() {
        System.out.println("Name:" + name + "\nAccno:" + accno + "\nType:" + type + "\nBalance:" +
balance);
    }
}

class savAcct extends account {

    private static double rate = 5;

    savAcct(String name, int accno, double balance) {
        super(name, accno, "Savings", balance);

    }

    void interest() {
        balance += balance * (rate) / 100;
        System.out.println("Balance:" + balance);
    }
```

```
}

class curAcct extends account {

   private double minBal = 500;
   private double serviceCharges = 50;

   curAcct(String name, int accno, double balance) {
      super(name, accno, "Current", balance);

   }

   void checkmin() {

      if (balance < minBal) {
         System.out.println("Balance is less than min balance,service charges imposed:" +
serviceCharges);
         balance -= serviceCharges;
         System.out.println("Balance is:" + balance);
      }

   }

}

class Bank {
   public static void main(String a[]) {
      Scanner s = new Scanner(System.in);
      System.out.println("Enter the name,type(current/savings),account number,initial balance:");
      String name = s.next();
      String type = s.next();
```

```
int accno = s.nextInt();
double balance = s.nextDouble();
int ch;
double amount1, amount2;
account acc = new account(name, accno, type, balance);
savAcct sa = new savAcct(name, accno, balance);
curAcct ca = new curAcct(name, accno, balance);
while (true) {
   if (acc.type.equals("savings")) {
      System.out.println("\nMenu\n1.deposit 2.withdraw 3.compute interest 4.display");
      System.out.println("Enter the choice:");
      ch = s.nextInt();
      switch (ch) {
         case 1:
            System.out.println("Enter the amount:");
            amount1 = s.nextInt();
            sa.deposit(amount1);
            break;
         case 2:
            System.out.println("Enter the amount:");
            amount2 = s.nextInt();
            sa.withdraw(amount2);
            break;
         case 3:
            sa.interest();
            break;
         case 4:
            sa.display();
            break;
         case 5:
            System.exit(0);
```

```
        default:
            System.out.println("invalid input");
            break;
    }
} else {
    System.out.println("\nMenu\n1.deposit 2.withdraw  3.display");
    System.out.println("Enter the choice:");
    ch = s.nextInt();
    switch (ch) {
        case 1:
            System.out.println("Enter the amount:");
            amount1 = s.nextInt();
            ca.deposit(amount1);
            break;
        case 2:
            System.out.println("Enter the amount:");
            amount2 = s.nextInt();
            ca.withdraw(amount2);
            ca.checkmin();
            break;

        case 3:
            ca.display();
            break;
        case 4:
            System.exit(0);
        default:
            System.out.println("Invalid input");
            break;
    }
}
```

```
    }


  }
}
```

GENERICS

```
class GenericStack<T> {
    private Object[] stackArray;
    private int top = -1;
    private static final int MAX_SIZE = 5;

    public GenericStack() {
        stackArray = new Object[MAX_SIZE];
    }

    public void push(T value) {
        if (top < MAX_SIZE - 1) stackArray[++top] = value;
        else System.out.println("Stack is full. Cannot push more elements.");
    }

    @SuppressWarnings("unchecked")
    public T pop() {
        if (top >= 0)
            return (T) stackArray[top--];
        else {
            System.out.println("Stack is empty. Cannot pop more elements.");
            return null;
        }
    }

    public boolean isEmpty() {
        return top == -1;
    }

    public boolean isFull() {
        return top == MAX_SIZE - 1;
```

```
    }
}
class Main{
public static void main(String[] args) {
        GenericStack<Integer> integerStack = new GenericStack<>();
        GenericStack<Double> doubleStack = new GenericStack<>();

        // Push integers to the integer stack
        for (int i = 1; i <= 5; i++) {
            integerStack.push(i);
        }

        // Push doubles to the double stack
        for (double i = 1.0; i <= 5.0; i++) {
            doubleStack.push(i);
        }

        // Pop and print integers from the integer stack
        System.out.println("Popped integers from the stack:");
        while (!integerStack.isEmpty()) {
            System.out.println(integerStack.pop());
        }

        // Pop and print doubles from the double stack
        System.out.println("Popped doubles from the stack:");
        while (!doubleStack.isEmpty()) {
            System.out.println(doubleStack.pop());
        }
    }
}
```

Abstract class prog

```
import java.lang.Math;

abstract class Shape {
   double a;
   double b;
   double c;

   abstract void calculateArea();

   abstract void calculatePerimeter();
}

class Triangle extends Shape {
   Triangle(double x, double y, double z) {
      a = x;
      b = y;
      c = z;
   }

   void calculateArea() {
      double s = (a + b + c) / 2;
      System.out.println("Area=" + (Math.sqrt(s * (s - a) * (s - b) * (s - c))));
   }

   void calculatePerimeter() {
      System.out.println("Perimeter=" + (a + b + c));
   }
}

class Circle extends Shape {
```

```
    Circle(double r) {
        a = r;
    }

    void calculateArea() {
        System.out.println("Area=" + (Math.PI * a * a));
    }

    void calculatePerimeter() {
        System.out.println("Perimeter=" + (2 * Math.PI * a));
    }
}

class ShapeM {
    public static void main(String[] args) {
        Triangle t = new Triangle(2.0, 3.0, 5.0);
        Circle c = new Circle(5.0);
        t.calculateArea();
        t.calculatePerimeter();
        c.calculateArea();
        c.calculatePerimeter();
    }
}
```

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

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

public class Pkgmain {
    public static void main(String args[]) {
        int numOfStudents = 2;
        Externals finalMarks[] = new Externals[numOfStudents];

        for (int i = 0; i < numOfStudents; i++) {
```

```
            finalMarks[i] = new Externals();
            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();
        }
    }
}
```

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 negetive:");
 }
}

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 FatherSonAge
{
 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;

   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);
      DisplayMessageThread thread2 = new DisplayMessageThread("CSE", 2000);

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

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

        try {
            Thread.sleep(30000);
        } catch (InterruptedException e) {
            System.out.println("Main thread interrupted.");
        }

        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 SwingDemo {
  SwingDemo() {
    // 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 divident:");

    // 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 bois
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) {
      System.out.println("Action event from a text field");
   }
};
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());
         int ans = a / b;

         alab.setText("\nA = " + a);
         blab.setText("\nB = " + b);
         anslab.setText("\nAns = " + ans);
```

```
            } catch (NumberFormatException e) {
                alab.setText("");
                blab.setText("");
                anslab.setText("");
                err.setText("Enter Only Integers!");
            } catch (ArithmeticException e) {
                alab.setText("");
                blab.setText("");
                anslab.setText("");
                err.setText("B should be NON zero!");
            }
          }
        });

        // 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 SwingDemo();
            }
        });
    }
}
```

10a) Interprocess communication using consumer and producer

```
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 < 6) {
            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 < 6) {
            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.");
    }
}
```

10b) Deadlock

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

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

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

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

class Deadlock implements Runnable {
    A a = new A();
    B b = new B();

    Deadlock() {
        Thread.currentThread().setName("MainThread");
        Thread t = new Thread(this, "RacingThread");
        t.start();
        a.foo(b); // 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();
    }
}
```