

# 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		
②	12/12/23	Program 1 of Lab		7 (W) 10-12-23
③	19/12/23	Program 2 of Lab		8 (W) 10-12-23
④	26/12/23	Program 3 of Lab		8 (W) 10-12-23
⑤	02/01/24	Program 4 of Lab		W 1-1-24
⑥	09/01/24	Program 5 of Lab		W 1-1-24
⑦	16/01/24	Program 6 of Lab		W 1-1-24
⑧	23/01/24	Program 7 of Lab		W 1-1-24
⑨	30/01/24	Program 8 of Lab		W 1-1-24
⑩	06/02/24	Program 9 of Lab		W 1-1-24
⑪	13/02/24	Program 10 of Lab		W 1-1-24
⑫	20/02/24	Program 11 of Lab		W 1-1-24

## Lab programs.

- ① → develop a java program that prints all real sol<sup>n</sup> to the quadratic  $ax^2+bx+c=0$ , read a,b & c & use the quadratic formula. If discriminant  $b^2-4ac$  is negative, display a message stating that we have no real sol<sup>n</sup>.

```
import java.util.Scanner;
```

```
class Quadratic
```

```
{
```

```
    int a, b, c;
```

```
    double x1, x2, 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();
```

```
y
```

```
    void compute()
```

```
{
```

```
    while(a == 0)
```

```
{
```

```
        System.out.println("Not a quadratic eqn");
```

```
        System.out.println("Enter a non zero");
```

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

```
        a = s.nextInt();
```

```
y
```

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

```
        if (d == 0)
```

```
{
```

```
            x1 = -b / (2 * a);
```

System.out.println ("Roots are real and equal");  
System.out.println ("root1 = root2 = " + x1);

}

else if (d > 0)

{

x1 = ((-b) + Math.sqrt(d)) / (a \* a);

x2 = ((-b) - Math.sqrt(d)) / (a \* a);

System.out.println ("Roots are real and distinct");

System.out.println ("root1 = " + x1 + "root2 = " +

y

else if (d < 0)

{

System.out.println ("Roots are imaginary");

x1 = (-b) / (a \* a);

x2 = (-Math.sqrt(-d)) / (a \* a);

System.out.println ("root1 = " + x1 + " + i" + x2);

System.out.println ("root2 = " + x1 + " - i" + x2);

y

y

class QuadraticMain

{

public static void main (String args[])

{

Quadratic q = new Quadratic();

q.getd();

q.computel();

System.out.println ("Done by " + suntehs B);

USN: 1BM22CS2031

y

Output:-

Enter the coefficients of a, b, c:

3 4 5

Roots are imaginary

$$\text{Root1} = 0.0 + i \cdot 1.1055415967851332$$

$$\text{Root2} = 1.1055415967851332 - i \cdot 1.1055415967851332$$

Done by: Sunitash B, USN: 1BM22CS243

Enter the coefficients of a, b, c:

1 2 1

Roots are real and equal

$$\text{Root1} = \text{Root2} = -1.0$$

Done by: Sunitash B, USN: 1BM22CS243

Enter the coefficients of a, b, c:

1 4 1

Roots are real and distinct

$$\text{Root1} = -0.2679491924211228$$

$$\text{Root2} = -3.13205080154877$$

Done by: Sunitash B, USN: 1BM22CS243

~~Final  
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 breadth:

5

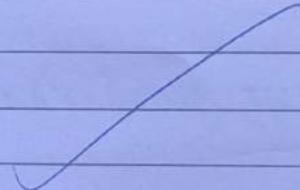
Area of the rectangle: 25

(3) Demonstration of sum of five digits

Enter 5 digit number

12345

sum of five digits : 15



Up  
Down

## 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();  
    }  
}
```

y

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

```
    system.out.print("Enter your USN : ");  
    this.USN = sc.nextLine();
```

```
}
```

```
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) +
```

```
    if (subject[0].grade == -1) subject[0].grade = 10;
```

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

```
y
```

```
y
```

```
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);  
    }  
}
```

$$\text{this.SGPA} = (\text{float})\text{totalGradeAndCredit} / \text{totalCredits}$$

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

```
}
```

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

```
}
```

output:-

Enter name : Santosh  
Enter your USN : IBM22CS243  
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 credits : 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 : 9.94999180265137

Name: Santosh B  
USN: IBM22CS243

100  
19-11-2023

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 object. 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 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;
    this.numPages + "\n";
}
```

```
return name + author + price + numPages;
```

y

```
public String getName() {
    return this.name;
```

y

```
public String getAuthor() {
    return this.author;
```

y

```
public int getPrice() {
    return this.price;
```

y

```
public int getNumberOfPages() {
    return this.numPages;
```

y

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.nextLine();
    String author = sc.nextLine();
    int price = sc.nextInt();
    int numPages = sc.nextInt();
}

```

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

}

```

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 : book2 santosh 1600 70.

Book name : book1

Author name : siri

Price : 1400

Number of pages : 68

Book name : book2

Author name : santosh

Price : 1600

Number of pages : 70

Q4) 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 width  
of a Rectangle : ")~~

this.a = sc.nextDouble();

this.b = sc.nextDouble();

}

@Override

```
void displayArea() {
```

```
    System.out.print("The area of the rectangle is : ");
```

```
(this.a * this.b));
```

```
}
```

```
}
```

```
class Triangle extends shape {
```

@Override

```
void getIput() {
```

```
    System.out.print("Enter height and base of  
Triangle : ");
```

```
this.a = sc.nextDouble();
```

```
this.b = sc.nextDouble();
```

```
}
```

@Override

```
void display() {
```

```
System.out.print("Enter the radius of the circle : ");
```

```
this.r = sc.nextDouble();
```

```
y
```

@Override

```
void displayArea() {
```

```
System.out.println("The area of the circle  
is : " + (Math.PI * this.r * this.r));
```

```
y
```

```
y
```

```
class AbstractDemo {
```

```
    public static void main (string[] args) {
```

```
        Shape s;
```

```
        s = new Circle();
```

```
        s.getInput();
```

```
        s.get s.displayArea();
```

```
        s = new Rectangle();
```

```
        s.getInput();
```

```
        s.displayArea();
```

```
        s = new Triangle();
```

```
        s.getInput();
```

```
        s.displayArea();
```

```
}
```

```
}
```

Output:-

enter the radius of circle : 3

The area of circle is : 28.27433882308138

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

Ques  
21-WM

## Lab 5

16/01/24

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 type) {
```

```
        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.acno);  
    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. Withdrawal  
        3. 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 withdrawal  
amount: ");  
amount = sc.nextDouble();  
this.withdraw(amount);  
break;

case 3: this.display();  
break;

case 4: return;

y

} while (choice != 4);

y

y

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

public Savings (String name, int accno, String type){

this.name = name;

this.accno = accno;

this.type = type;

y

```
void deposit (double amount) {
```

```
    float interestAmount = (float) (amount *  
        (this . interest / 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 . 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();
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.print("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.nextLine();  
        int accno = sc.nextInt();  
        String type = sc.nextLine();
```

```
        if (type.equals ("current")) {
```

```
            Account c = new Current (name, accno,  
                type);
```

```
c.menu();
```

4

```
else {
```

```
            Account s = new Savings (name, accno,  
                type);
```

```
s.menu();
```

5

6

Output:

Enter name, accno, type:

santosh

12

current

-- MENU --

1. deposit 2. withdraw 3. details 4. exit  
1

Enter the deposit amount : 50000  
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. 0

Name : Santosh

Account number : 12

Balance : 6000.

## Lab 5:

- g) 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());  
}
```

```
Stack<Double> d = new Stack<Double>();  
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

16/1/2024

Ques 6

String

① BMSCE

BMSCE

② 3

3

Roll no 10 is present

③ Dimensions are 10.0 by 14.0 by 12.0

bosch: Dimensions are 10.0 by 14.0 by 12.0

④ bmsce

⑤ 65

66

67

Welcome to bmsce college

⑥ Bmsce equals Bmsc → true

Bmsc equals college → false

Bmsc equals Ignorance BMSCE → true

⑦ substring is matched

s1 = "Bmsce college"

ca = "welcome to Bmsce college of Engineering"

⑧ true

false

⑨ false

true

⑩ `Hello == Hello` → true  
`Hello == Hello` → false

⑪ The names in alphabetical order are:

apple

ball

cat

lion

watch

⑫ sorted Number (Ascending Order) : [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

⑬ ~~Th~~ was was a test . That was too

⑭ hello world

⑮ commeg

⑯ hello friendy

⑰ student 1

name : prateek

Reg no : 123

Semester : 3

CGPA : 8.87

~~student 2~~

name : howla

Reg no : 124

Semester : 4

CGPA : 9.05

(8) chord at 3 is 'n'

reverse of SAS is SAS

(9) Eagle is flying

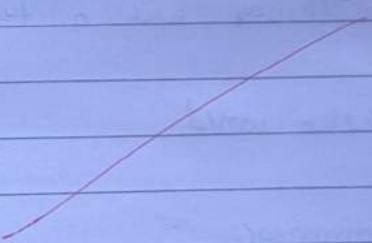
Eagle makes a sound

Hawk is moving

Hawk is making a sound

(10) Circle - Area =  $18.5398$  Perimeter =  $31.4259$

Triangle - Area =  $5.0$  Perimeter =  $12.0$



$$\frac{W}{2H-XW}$$

## Lab program 6

Create package CIT which has two classes : student and internal. The class Student has members like usn, name, sem. The class internal derived from student has an array that stores the internal marks stored on 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 stored on five courses of the current semester of the student. Import the two packages in a file that calculates the final marks of n students in all for courses.

Student.java

package CIT;

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 = sc.nextLine();

System.out.println("Enter name:");

name = sc.nextLine();

System.out.println("Enter semester:");

sem = sc.nextInt();

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

y  
y

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

```
import java.util.Scanner;
```

```
public class Internals extends Student {  
    protected int marks[] = new int[5];  
    public Internals() {  
        super();  
    }
```

```
    public void inputMarks() {  
        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();  
        }  
    }
```

y

//externals.java.

package SEE;

import SEE.Internal;

import java.util.Scanner;

public class externals extends Internal {

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];

y

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

// Main.java

import SEE.External;

```
public class Main {
    public static void main (String args[]) {
        int numofStudents = 2;
        External finalMarks[] = new External [numofStudents];
        for(int i=0; i< numofStudents; i++) {
            finalMarks[i] = new External();
            finalMarks[i].inputStudentDetails();
            System.out.println("Enter CGS marks");
            finalMarks[i].inputCGSMarks();
            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: 18

subject 4 marks: 84

subject 5 marks: 90

Enter USN: 112

Enter Name: Sivu

Enter semester: 3

Enter CIE marks

Enter internal marks for sivu

subject 1 marks: 33

subject 2 marks: 28

subject 3 marks: 33

subject 4 marks: 21

subject 5 marks: 31

Enter SEE marks

Enter SEE marks for sivu

subject 1 marks: 84

subject 2 marks: 85

subject 3 marks: 89

subject 4 marks: 92

subject 5 marks: 93

displaying marks:

USN : 111

Name :- sumanth

semester : 3

subject 1 : 77

subject 2 : 81

subject 3 : 67

subject 4 : 73

subject 5 : 85

USN : 112

Name : siva

semester : 3

subject 1 : 75

subject 2 : 76

subject 3 : 72

subject 4 : 73

subject 5 : 74

date  
24-1-24

## Lab program 7.

Date \_\_\_\_\_  
Page \_\_\_\_\_

- 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 takes both father and sons age and throws an exception if sons age is != father's 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() { throw new 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);

}

y

class Main {

public static void main (String[] args) {

try {

Son s = new Son();

s.display();

y

catch (WrongAge e) {

System.out.println("Error: " + e.getMessage());

y

y

### 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

W  
20-1-2  
y

## 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 2000 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);
```

```
y
```

```
} catch (InterruptedException e) {
```

```
    System.out.println(Thread.currentThread().  
        getName() + " interrupted.");
```

```
y
```

```
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.

BMS college of engineering

CSE

CSE

CSE

CCE

CSE

BMS college of Engineering

Thread 1 interrupted.

Thread 2 interrupted.

Main thread exiting.

✓  
Date / 2024  
6/2

13/02/24 Lab program 10

→ Implementation of a producer and consumer

class A {

int n;

boolean valueset = false;

synchronized int get() {

while (!valueset)

try {

System.out.println("Consumer waiting (" + n + ")");

wait();

} catch (InterruptedException e) {

System.out.println("InterruptedException caught");

}

System.out.println("got: " + n);

valueset = true;

System.out.println("Intimate producer");

notify();

return n;

}

synchronized void put(int n) {

while (valueset)

try {

System.out.println("Producer waiting");

wait();

} catch (InterruptedException e) {

System.out.println("InterruptedException caught");

}

this.n = n;

valueset = true;

System.out.println("Put: " + n);

System.out.println("Intimate consumer");

notify();

}

class Producer implements Runnable {

a q;

Producer(a q) {

this.q = q;

new Thread(this, "producer").start();  
}

public void run() {

int i=0;

while (i<15) {

q.put(i++);

}

y

}

class Consumer implements Runnable {

a q;

Consumer(a q) {

this.q = q;

new Thread(this, "consumer").start();

y

public void run() {

int i=0;

while (i<15) {

int s = q.get();

System.out.println("consumed: " + s);

i++;

y

y

}

class Producer {

```
public static void main(String[] args) {  
    Q q = new Q();  
    new producer(q),  
    new consumer(q),  
    System.out.println("Press Control-C to stop");  
}
```

y  
y

output put: 1

Got: 1

put: 2

Got: 2

put: 3

Got: 3

put: 4

Got: 4

put: 5

Got: 5

Waz  
13-2-24

→ 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.bar");

b.bar();

}

void last() {

System.out.println("Inside A.last");

b

}

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")

instead t = new Thread(this, "Running Thread"),

t.start();

a.foo(b);

System.out.println("Back in main thread");

y

public void run() {

b.bar(a);

System.out.println("Back in other thread");

y

public static void main(String args[]) {

new Deadlock();

y

~~output~~

Main Thread entered A.foo

Running Thread entered B.bar

Main Thread trying to call B.last()

Inside A.last

Back in main thread.

Running Thread trying to call A.last()

Inside A.last

Back in other thread

Q1  
15-2-24

## Lab programming :-

- 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 & num2 were not an integer the program would throw NumberFormatException. If num2 were zero, the program would throw an ArithmeticException display the exception in message dialog box.

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

```
class DivDemo {
```

```
    DivDemo() {
```

```
        JFrame jfm = new JFrame("Divider App");
```

```
        jfm.setSize(250, 150);
```

```
        jfm.setLayout(new GridLayout());
```

```
        jfm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

```
JLabel glab = new JLabel("Enter the divisor  
and dividend:");
```

```
JTextField afield = new JTextField(8);
```

```
JTextField bfield = new JTextField(8);
```

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

```
JLabel ar = new JLabel();
```

```
JLabel acab = new JLabel();
```

```
JLabel blab = new JLabel();
```

JLabel ansLab = new JLabel();

```
jpm.add(ui);
jpm.add(jLab);
jpm.add(ajtt);
jpm.add(bjtt);
jpm.add(button);
jfm.add(alab);
jfm.add(blab);
jfm.add(ansLab);
```

ActionListener l = new ActionListener() {

private void

public void actionPerformed(ActionEvent evt) {

try {

int a = Integer.parseInt(ajtt.getText());
int b = Integer.parseInt(bjtt.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);

err.setText("");

} catch (NumberFormatException e) {

displayErrorMessage("Enter only integers!");

} catch (ArithmaticException e) {

displayErrorMessage("B should be non zero");

}

y

private void displayMessage (String message) {

    aLab.setText (" " );

    bLab.setText (" " );

    cLab.setText (" " );

    msg.setText (message);

}

y;

button.addActionListener (calculateListener);

if (m.isVisible () == true) {

}

public static void main (String args []) {

    SwingUtilities.invokeLater (new Runnable () {

        public void run () {

            new Application ();

}

});

y

b

Output:-

Enter the divisor and dividend

6        3

Calculate    A = 6    B = 3    Ans = 2

## functions used:-

- 1) `Frame` :- It is a top-level container in Java Swing that represents a window with a title bar, border and optional menu bar.
- 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 row like manner.
- 4) `add` : This line adds the user label to the frame
- 5) `setText` : This line sets the text of the 'A' label to display the value of 'A'
- 6) `setVisible` : This line makes the frame visible
- 7) `invokeLater` :- This line schedules a job for the event-dispatcher
- 8) ~~invokeLater~~ :- To perform task asynchronously in the ~~not~~ event dispatcher thread.

SJ  
29/2/2022