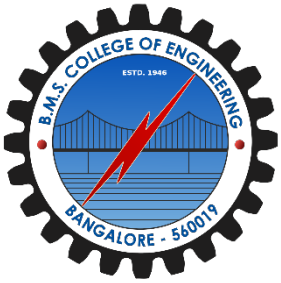


B.M.S COLLEGE OF ENGINEERING

Basavanagudi-560019

Subject:-

OBJECT ORIENTED JAVA PROGRAMMING

LAB OBSERVATION

Submitted by:

SHREYAS K

1BM22CS271

3E – BATCH 3

LAB-1

LAB-program

12/12/23

1) Quadratic Equation

```
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("Root 1 = "+r1+" Root 2 = "+r2);
}
else if(d<0)
{
System.out.println("Roots are imaginary");
r1 = (-b)/(2*a);
r2 = Math.sqrt(-d)/(2*a);
System.out.println("Root 1 = "+r1+" + i" +r2);
System.out.println("Root 2 = "+r1+" - i"+r2);
}
}
}
class Quadratic Main
{
public static void main(String args[])
{
Quadratic q = new Quadratic();
q.getd();
q.compute();
System.out.println(Shreyas.K - 1BM22CS271);
}
}
```

Output

① Enter the coefficients of a, b, c
1
2
1
Roots are real and equal
Root 1 = Root 2 = -1.0
Shreyas.K - 1BM22CS271

② Enter the coefficients of a, b, c
1
3
2
Roots are real and distinct
Root 1 = -1.0 Root 2 = -2.0
Shreyas.K - 1BM22CS271.

③ Enter the coefficient of a, b, c
2
1
3
Roots are imaginary
Root 1 = 0.1 + i1.1989578808
Root 2 = 0.0 - i1.1989578808
Shreyas.K - 1BM22CS271

16

8am
12/12/23

19/12/23

LAB Program - (2)

=> Develop a java prog to create a class student with member usn, name, an array credits and an array marks. Include method to accept and display details and a method to calculate SGPA of a student.

$$SGPA = \frac{\sum [(\text{Course Credits})(\text{Grade Points})]}{\sum [\text{Course Credits}]}$$

```
import java.util.Scanner;
class subject
{
  int subjectMarks;
  int credits;
  int grade;
}
class student
{
  subject subject[];
  String name;
  String usn;
  double sgpa=0;
  Scanner s;

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

  void getStudentDetails()
  {
    System.out.println("Enter your name");
    name = s.next();
    System.out.println("Enter your usn:");
    usn = s.next();
  }
  void getMarks()
  {
    for(int i=0; i<8; i++)
    {
      System.out.println("Enter marks for subject"+(i+1)+":");
      subject[i].subjectMarks = s.nextInt();
      System.out.println("Enter credits for subject"+(i+1)+
                         ":");
      subject[i].credits = s.nextInt();
      subject[i].grade = (subject[i].subjectMarks/10)+1;
      if(subject[i].grade>11)
        subject[i].grade=10;
      if(subject[i].grade<=4)
        subject[i].grade=0;
    }
  }

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

```
Class main
{
  public static void main(String args[])
  {
    Student s1 = new Student();
    s1.getStudentsDetails();
    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
Enter your USN
1BM22CS271
Enter marks of subject 1
91
Enter credits of subject 1
4
Enter marks of subject 2
60
Enter credits of subject 2
4
Enter marks of subject 3
85
Enter credits of subject 3
3
Enter marks of subject 4
88
Enter credits of subject 4
3.
Enter marks of subject 5
81
Enter credits of subject 5
3.
Enter marks of subject 6
85
Enter credits of subject 6
1
Enter marks of subject 7
95
Enter credits of subject 7
1
Enter marks of subject 8
98
Enter credits of subject 8
1

Name: Shreyas
USN: 1BM22CS271
SGPA: 9.4.
```

26/12/23

Shreyas.K
1BM22CS271

26/12/23

LAB-3

Q) Create a Java program to create n books object

⇒

```
import java.util.Scanner;

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

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

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

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

        Book[] books = new Book[n];

        for (int i=0; i<n; i++) {
            System.out.println("Enter details " + (i+1) + ":");
            System.out.println("Name: ");
            String name = s.next();
            System.out.print("Author: ");
            String author = s.next();
            System.out.print("price: ");
            int price = s.nextInt();
            System.out.print("Number of pages: ");
            int numpages = s.nextInt();

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

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

Output:-

```
Enter the number of Books:
2
Enter details for book 1:
name: Java
Author: dkdk
price: 2500
number of pages: 554
Enter details for book 2:
name: Biology
Author: NCERT
price: 250
number of pages: 150
```

Shreyas.K
1BM22CS271

26/12/23

2/1/24

LAB-4

Develop a JAVA program to create an abstract class named shape that contain two integers and an empty method named printArea(). Provide three classes named Rectangle, Triangle, Circle such that each one of the class extends the class shape. Each one of the classes contain only the method printArea() that print 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();
   Tringle t= new Triangle();
   Circle c = new circle();
    r.getInput();
    r.displayArea();
    t.getInput();
    t.displayArea();
     c.getInput();
     c.displayArea();
   }
```

Output:-

```
Enter the length and breath:
  10
  5
Area of rectangle is : 50

Enter the length and height:
  10
  10
Area of triangle is : 20
Enter the radius
  10
Area of circle:
  314.
```

Fri 05/01/24

Shreyas.k
1BM22CS271

9/1/24

LAB - 5

Develop java program to develop a class bank with current account & savings account.

```
import java.util.Scanner;

class input{
  scanner sc = new scanner (system.in);
}
class Account extends input{
  string name;
  int accNo;
  double balance;

public void getDetails(sc)
 {
  system.out.print("Enter name: ");
  name = sc.nextLine();
  System.out.print(" Enter account number: ");
  accNO = sc.nextInt();
 }

public void deposite(){
  System.out.print("Enter amount to deposit: ");
   double amt = sc.nextDouble();
  balance += amt;
  System.out.println("Amount deposited successfully.");
}

public void withdraw(){
 system.out.print("Enter amt to withdraw: ");
  double amt = sc.nextDouble();
    if (balance >= amt){
      balance -= amt;
```

```
System.out.println("Amount withdraw successfully.");
} else {
  System.out.println("Insufficient balance.");
}
}

public void display(){
  System.out.println("Name:"+name);
  System.out.println("Account No: "+accNo);
  System.out.println("Balance: "+balance);
}
}

class Savings extends Account{
  final double interestRate = 0.04;

public void computeInterest(){
  double interest = balance * interestRate;
  balance += interest;
  System.out.println("Interest credited: "+interest);
}
}

class Current extends Account{
  final double minBalance = 500;
  final double penalty = 100;

  @Override
  public void withdraw(){
    super.withdraw();
    checkMinBalance();
  }

private void checkMinBalance(){
  if (balance < minBalance){
    balance -= penalty;
    System.out.println("Penalty applied for low balance.")
  }
}
}

class Bank extends Account
{
  public static void main(String[] args)
  {
    Scanner sc = new Scanner(System.in);
    Savings ob1 = new saving();
    current ob2 = new current();

  ob1.getDetails();
  ob2.getDetails();

  int choice;
    String acc;
    System.out.println("Menu:");
    System.out.println("1. Deposite");
    System.out.println("2. withdraw");
    System.out.println("3. Display Balance");
    System.out.println("4. Compute Interest (savingsonly)");
    System.out.println("5. Exit");
    do {
      System.out.print("Enter your choice:");
      choice = sc.nextInt();
      System.out.println("Enter thay acc type:");
      acc = sc.next();
      switch(choice){
```

```
case 1:
  if (acc.equals("saving")) ob1.deposit(); else
    ob2.deposit();
  break;
case 2:
  if (acc.equals("saving")) ob1.withdraw(); else
    ob2.withdraw();
    break;
case 3:
  if (acc.equals("saving")) ob1.withdraw();
  else ob2.withdraw();
    break;
case 4:
  ob1.computeInterest();
  break;
case 5:
  System.out.println("Existing...");
  break;
  default:
  System.out.println("Invalid choice");
  }
 } while (choice != 5);
}
}
```

Output put :-

```
Enter name : shamanth
Enter acc no : 1
Enter name : Shreyas
Enter acc no : 2

Menu:
1) Deposit
2) withdraw
3) Display Balance
4) Compute interest (saving only)
5) Exit
Enter your choice : 1
Enter the acc type : saving
Enter amount to deposite : 1000
Amount deposite successfully
Enter your choice : 2
Enter the acc type : saving
Enter amount to withdraw : 500
Amount withdrawn successfully.
Enter your choice : 3
Enter the acc type : saving
Name : shamanth
Acc Number : 1
balance : 500.0
Enter your acc type : saving
Enter amount to withdraw : 400
Amo withdrawn successfully.
penalty applied for low balance
Enter your choice : 3
Enter the acc type : current
Name : Shreyas
Amount Number : 2
Balance : 300.0
```

16/04/24

Shreyas.K
1BM22CS271

16/1/24

String & Abstract :-

1) Write a java program to create a generic class stack which holds 5 integer & 5 double values.

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

public GenericsStack()
{
  stackArray = new Object[N];
}
public void push (T value)
 {
   if (top<N-1)
   stackArray[++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 smain
{
  public static void main (String args[])
   {
   GenericStack<Integer> integerstack = new GenericStack<>();
   GenericStack<Double> doublestack = 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()){
   Sys.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

Shreyas.K
1BM22CS271

16/1/24

2) Strings - Demonstrate string length, string literal, String concat.

```
public class String1 {
    public static void main(String args[])
    {
        System.out.println("Demo string length:");
        String a = "Hello";
        System.out.println(a.length());

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

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

output:

```
Demo string length:
5
Str concat:
He is 9 years old
Demo literals
3
```

Shreyas.K
1BM22CS271

16/1/24

3) Abstract class.
Create subclasses Circle & Triangle that extends the shape class

```
import java.lang.Math;

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

    abstract void calculateArea();
    abstract void calculatePeri();
}
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 calculatePeri()
    {
        System.out.println("Peri = " + (a+b+c));
    }
}
```

```
class Circle extends Shape{
    circle (double r)
    {
      a=r;
    }
    void calculateArea()
     {
      Sy.out.println("Area="+Math.PI*a-a));
    }
    void calculatePeri()
     {
      sout.("Peri" +(2* math.PI*a));
    }
}

class shapeM{
   public static void main(String [], args)
    {
     Triangle t= new triangle(2.0, 3.0, 3.0);
     circle c= new (circle(5.0);
     t.calculateArea();
     t.calculatePeri();
     c.calculateArea();
     c.calculatePeri()
    }
}
```

Op:- Area = 4.145
Peri = 11.0
Area = 78.53981
Peri = 31.41592.

Shreyas.K
1BM22CS271

23/1/24

LAB-6

Create package CIE which has 2 classes student & internals. The class students has members USN, name, sem. The class derived from student has an array that stores internal marks scored in 5 subject.

student.Java

```
Package CIE
Import Java.util.*;
public class student {
   protected string usn = new string();
   protected string name = new string();
    protected int sem;

public void inputstudentDetails(){
     scanner S = new Scanner (System.in);
     this.usn = S.nextline();
     this.name = S.next line();
     this.sem = S.nextInt();
  }
public void displaystudentDetails(){
      systemout.println(this.usn + " " +this.name
  }
}
```

Internals.Java.

```
package CIE;
import java.util.*;
public class Internal extends Student {
  protected int marks[] = new int[5];
    public void inputCIEmarks() {
      Scanner S = new Scanner(System.in);
        for (int i=0; i<5; i++)
          marks[i] = S.nextInt();
  }
}
```

External.java.

```
package SEE;
import CIE.Internals;
import java.util.Scanner;
public class External extends Internal {
  protected int marks[];
  protected int finalMarks[];
    public External() {
      marks = new int[5];
      finalMarks = new int[5];
  }
  public void inputSEEmarks() {
    for(int i=0; i<5; i++)
      marks[i] = S.nextInt();
  }
  public void calcFinalmarks() {
      for (int i=0; i<5; i++)
        finalMarks[i] = marks[i]/2 +
                        super.marks[i];
  }
```

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

Main.java.

```
import SEE.Externals;
class main {
  public static void main(String args[]) {
    int numOfStudents = 2;
    Externals finalMarks[] = new Externals[];
    for (int i=0; i<numOfStudents; i++) {
      finalMarks[i] = new Externals();
      finalMarks[i] = inputStudentDetails();
      System.out.println("Enter CIEMarks");
      finalMarks[i].inputCIEmarks();
      System.out.println("Enter SEE Marks");
      finalMarks[i].inputSEEmarks();
    }
  for (int i=0; i<numOfStudents; i++) {
    final[i].calculateFinalMarks();
    final[i].displayfinalMarks();
  }
}
```

Output:-

```
Enter Student Details:
1BM22CS271
Shreyas
2
Enter CIE Makes
35
40
41
47
44
Enter SEE Marks
80
90
86
80
84
Enter Student Details
1BM22CS256
Shashank
2
Enter CIE marks
50
45
40
48
46
Enter SEE marks
90
96
100
90
92
```

```
Display Details:
USN : 1BM22CS271
Name : Shreyas
sem : 2
subject 1 : 80
subject 2 : 90
subject 3 : 84
subject 4 : 88
subject 5 : 86
USN 1BM22CS256
Name : Shashank
sem : 2
subject 1 : 95
subject 2 : 93
subject 3 : 90
subject 4 : 96
subject 5 : 92
```

Shreyas.K
1BM22CS271

30/12/24

LAB-7

Exception Handling

Write a program that demonstrate handling of exceptions in inheritance tree. Create a base class father & derive class son from it. In Father class implement a constructor which throws wrongAge when age<0. In son class implement a constructor which throws wrongAge() when age >= Father age.

```
import java.util.Scanner;

class WrongAge extends Exception{
    public WrongAge(String s){
        super(s);
    }
}

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

class Father extends Input{
    int fatherAge;
    Father(){
    fatherAge = sc.nextInt();
    try{
      check();
    }catch(WrongAge e){
      System.out.println(e);
    }
    void check() throws WrongAge{
        if (fatherAge<0)
        throw new WrongAge("cannot
                            be negative");

    }
}
```

```
void display (){
    System.out.println("Father Age : " + fatherAge);
  }
}
class son extends Father {
  int sonAge;
  son(){
    super();
      try {
      check();
    }
    Catch (WrongAge R){
        System.out.println(e);
    }
  }
void check() throws WrongAge {
  if (sonAge < 0)
  throw new WrongAge ("cannot be negative");
    else if (sonAge > fatherAge)
      throw new WrongAge ("son Age can't be
          greater than father's Age");
      else if (sonAge == fatherAge");
    throw new WrongAge("son Age can't be
      equal to father's age.");
    }
  void display (){
    System.out.println(e);
  }
}
```

```
class Main {
  public static void main (String, args[])
    {
      son s = new son();
        s.display();
    }
}
```

Output:

50

30

Father Age : 50

Son Age : 30

-10

Age cannot be negative.

20

40

Son's Age cannot be greater than Father's Age

Shreyas.K

1BM22CS271

6/2/24

LAB-8

write a program which creates two threads one displaying "BMS college of Engineering" every 10sec & another displaying "CSE" every 2 sec.

```
class displayMessage implements Runnable {
    String message;
    int interval;

    displayMessage (String message, int interval) {
        this.message = message;
        this.interval = interval;
    }
    public void run() {
        while (true) {
        System.out.println (message);
        try {
        Thread.sleep (interval *1000);
        } catch (InterruptedException e) {
            System.out.println(e);
        }
    }
}

public class multi {
    public static void main (String args[]) {
    Thread t1 = new Thread (new displayMessage
        ("BMS College of Eng", 10);
    Thread t2 = new Thread (new Display message
        ("CSE", 2);
        t1.start();
        t2.start();
    }
}
```

Output:-

```
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
```

Shreyas.K
1BM22CS271.

6/2/24

LAB-10

Demonstrate inter communication and deadlock

```
class Q {
    int n;
    boolean valueSet = false;
    synchronized int get() {
    while (!valueSet)
      try {
      System.out.println("\nConsumer waiting\n");
      wait();
    } catch (InterruptedException e) {
    System.out.println("InterruptException 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("InterruptException 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 consume(q);
    System.out.println("Press Control-C to stop");
```

Output:-

```
put:0
intimate consume
Producer waiting
got:0
Intimate Producer
consumed:0
consume waiting
Put:1
Intimate Consumer
Producer Waiting
Got:1
Intiamate producer
consumed:1
```

```
Put:2
Intimate Consume
Producer waiting
Got:2
Put:3
Intimed consumed
Producer waiting
Got:3
Intimate consumed
consumed:3
Put:4
```

Shreyas.K
1BM22CS271

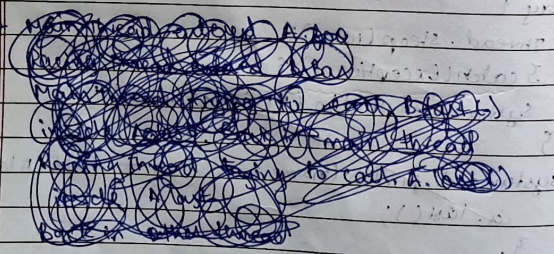
3/2/24

=> Program to Demostrate Deadlock :-

```
Class A {
  Synchronized void foo(B b) {
  String name = Tread.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()");
  }
  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, "Racing thread");
  t.start();
   a.foo(b);
   System.out.println("Back in main thread");
  }
public void run() {
b.bar(a);
System.out.println("Back in other thread");
}
public static void main(String args[]) {
new Deadlock();
}
}
```



Output -

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

13.02.24

Shreyas.K
1BM22CS271

20/2/24

LAB-9

Q) WAP that creates a user interface to perform integer division. The user enters 2 nos in the text fields, Num1 & Num2.

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

class SwingDemo
 {
    SwingDemo ()
    {
     JFrame jfrm = new JFrame ("Divider App");
     jfrm.setSize (275, 150);
     jfrm.setLayout (new FlowLayout ());
     jfrm.setDefaultCloseOperation (JFrame.EXIT_ON_
                                    CLOSE);
    JLabel jlab = new JLabel ("Enter the divider &
                               dividend:");
    JTextField ajtf = new JTextField (8);
    JTextField bjtf = new JTextField (8);

    JButton button = new JButton ("Calculate");
     JLabel err = new JLabel ();
     JLabel alab = new JLabel ();
     JLabel blab = new JLabel ();
     JLabel anslab = new JLabel ();

     jfrm.add (err);
     jfrm.add (jlab);
     jfrm.add (ajtf);
     jfrm.add (bjtf);
     jfrm.add (button);
     jfrm.add (alab);
     jfrm.add (blab);
```

```
ActionListener l = new ActionListener ();
 {
   public void actionPerformed (ActionEvent evt)
   {
     System.out.println ("Action event from a textfield")
   }
};
 ajtf.addActionListener (l);
 bjtf.addActionListener (l);

 button.ActionListener (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 ("In A = "+a);
     blab = setText ("In B = "+b);
     anslab.setText ("In = "+ans);
     } catch (NumberFormatException e)
         { alab.setText (" ");
           blab.setText (" ");
           anslab.setText (" ");

   } catch (ArithmeticException e) {
           alab.setText (" ");
           blab.setText (" ");
           anslab.setText (" ");
           err.setText ("B should be non zero");
   }
}
```

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

output:-

Enter the divider and divident:

10 2

calculate A=10 B=2 Ans=5

Sreyas.K
1BM22CS271

20/2/24

Functions

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

setsize(int width, int height) - used to resize a frame using width & height parameters.

setlayout() - method allows you to set the layout of the container. The layout managers help by out the components held by this container.

setDefaultCloseOp() - method is used to seperate one of several option for the close button Jframe.Exit_on_close - Exit the application.

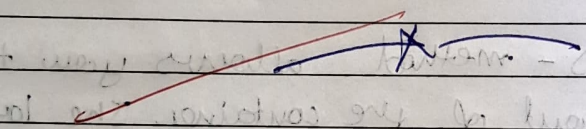
Jlabel - The object of Jlabel class is a component for placing text in a container It is used to display a single line of read only text.

JText field - The object of Jtext field class is a text component that allows the editing of a single line text. It inherits textcomp class.

add(frame) = adds new frame to the existing frame.

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 field.

setVisible() - is a method that has return type boolean



Shreyas.K
1BM22CS271

22.02.24

LAB: 1

Develop a Java program that prints all real solutions to the quadratic equation $ax^2+bx+c = 0$. Read a, b, c and use the quadratic formula. If the discriminate b^2-4ac is negative, display a 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("Roo1 = 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();
	}
}
```

LAB: 2

Develop a Java program to create a class Student with members usn, name, an array credits and an array mark. Include methods to accept and display details and a method to calculate SGPA of a student.
//Develop a java program to create a class Student with members usn,name, an array creadits 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 Student{
   String name;
   String usn;
   double SGPA;
   Subject subject[];
   Scanner s;

   Student(){
      int i;
      subject=new Subject[9];
      for(i=0;i<9;i++){
         subject[i]=new Subject();
         s=new Scanner(System.in);
      }

   }

   void getStudentDetails(){
      System.out.println("Enter your Name:");
      name=s.next();
      System.out.println("Enter your USN:");
      usn=s.next();
   }

   void getMarks(){
      for(int i=0;i<9;i++){
         System.out.println("Enter marks for subject "+(i+1)+":");
         subject[i].subjectMarks=s.nextInt();
         System.out.println("Enter credits for subject "+(i+1)+":");
         subject[i].credits=s.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 totalCreadits=0;
```

```
      for(int i=0;i<9;i++){
         effectiveScore += (subject[i].grade*subject[i].credits);
         totalCreadits += subject[i].credits;
      }

      SGPA=(double)effectiveScore/(double)totalCreadits;
   }
}
class Subject{
         int subjectMarks;
         int credits;
         int grade;

}
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);
         }
}
```

LAB: 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 Book{
   String name, author;
   int price,page_num;

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

   String toStrings(){
      String name,author,price,page_num;
      name="Book name: "+this.name+"\n";
      author="Author name: "+this.author+"\n";
      price="Price: "+this.price+"\n";
      page_num="Number of pages: "+this.page_num+"\n";

      return (name+author+price+page_num);
   }

   public static void main(String args[]){
      int n;
      String name,author;
      int price,page_num;

      Scanner sc=new Scanner(System.in);

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

      Book b[]=new Book[n];

      for (int i=0;i<n;i++){
         System.out.println("Enter name of book:");
         sc.nextLine();
         name=sc.nextLine();

         System.out.println("Enter author of a book:");
         author=sc.nextLine();

         System.out.println("Enter the price of book:");
         price=sc.nextInt();

         System.out.println("Enter the no. of pages of book:");
         page_num=sc.nextInt();
```

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

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

LAB: 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.Scanner;
public class InputScanner {
	Scanner s;
	InputScanner(){
		s=new Scanner(System.in);
	}
}
abstract public class Shape extends InputScanner{
	double a,b;
	abstract void getInput();
	abstract void displayArea();
}
public class Rectangle extends Shape {
	void getInput() {
		System.out.println("Enter the dimension of rectangle:");
		a=s.nextDouble();
		b=s.nextDouble();
	}

	void displayArea() {
		System.out.println("Area of reactangle:"+(a*b));
	}
}
public class Circle extends Shape{
	void getInput() {
		System.out.println("Enter the dimension of circle:");
		a=s.nextDouble();
	}

	void displayArea() {
		System.out.println("Area of circle:"+(3.14*a*a));
	}
}
public class Triangle extends Shape {
	void getInput() {
		System.out.println("Enter the dimension of triangle:");
		a=s.nextDouble();
		b=s.nextDouble();
	}

	void displayArea() {
		System.out.println("Area of triangle:"+((a*b)/2));
	}
}
public class MainMethod {

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

```
                Rectangle R=new Rectangle();
                R.getInput();
                R.displayArea();

                Triangle T=new Triangle();
                T.getInput();
                T.displayArea();

                Circle C=new Circle();
                C.getInput();
                C.displayArea();
        }

}
```

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. 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.

```
import java.util.Scanner;
public class Account {
	String customer_name;
	int account_no;
	String type_acc;
	double balance=0;
	Scanner sc;

	Account(String customer_name,int account_no,String type_acc){
		this.customer_name=customer_name;
		this.account_no=account_no;
		this.type_acc=type_acc;
		sc=new Scanner(System.in);
	}

	void diposite() {
		System.out.println("Enter the deposit amount:");
		int dipo=sc.nextInt();
		balance+=dipo;
	}

	void withdrawal() {
		System.out.println("Enter the withdrawl amount:");
		int with=sc.nextInt();
		if(with>balance) {
			System.out.println("Insufficient Balance");
		}
		else{
			balance-=with;
		}
	}

	void display() {
		System.out.println("Customer name:"+customer_name);
		System.out.println("Account number:"+account_no);
		System.out.println("Type of Account:"+type_acc);
		System.out.println("Balance:"+balance);
	}
	void applyinterest() {}

}

public class Cur_acct extends Account {

	Cur_acct(String customer_name,int account_no,String type_acc){
		super(customer_name,account_no,type_acc);}
```

```
	void withdrawal() {
		System.out.println("Enter the withdrawl amount:");
		int with=sc.nextInt();
		if (balance<=2000) {
			double pen=balance/(0.06);
			System.out.println("Insuffiecient balance penalty to be paid:"+pen);
			balance+=pen;
		}
		else{
			balance-=with;
		}
	}
}

public class Sav_acct extends Account{
Sav_acct(String customer_name,int account_no,String type_acc){
super(customer_name,account_no,type_acc);
}

void applyinterest() {
System.out.println("Enter the interest rate:");
int rate=sc.nextInt();
double interest=balance*rate;
balance+=interest;
System.out.println("Balance after interest:"+balance);
}
}

import java.util.Scanner;
public class Bank {
	public static void main(String[] args) {
		Scanner sc=new Scanner(System.in);
		System.out.println("Enter customer name:");
		String cus_name=sc.nextLine();
		System.out.println("Enter account number:");
		int acc_no=sc.nextInt();

		Account ca=new Cur_acct(cus_name,acc_no,"Current Account");
		Account sa=new Sav_acct(cus_name,acc_no,"Saving Account");

		int choice;
		while(true){
			System.out.println("----MENU----");
			System.out.println("1.Deposite\n2.Withdrawl\n3.Compute intereset for Saving
			account\n4.Display account details\n5.Exit");
			choice=sc.nextInt();

			switch(choice) {
			case 1:
			System.out.println("Enter the type of account:\n1.Saving Account \n2.Current Account");
			int acc=sc.nextInt();
			if(acc==1) {
```

```
                    sa.diposite();
                    }
                    else {
                    ca.diposite();
                    }
                    break;
                    case 2:
                    System.out.println("Enter the type of account:\n1.Saving Account \n2.Current Account");
                    int acc1=sc.nextInt();
                    if(acc1==1) {
                    sa.withdrawal();
                    }
                    else {
                    ca.withdrawal();
                    }
                    break;
                    case 3:
                    sa.applyinterest();
                    break;
                    case 4:
                    System.out.println("Enter the type of account:\n1.Saving Account \n2.Current Account");
                    int acc2=sc.nextInt();
                    if(acc2==1) {
                    sa.display();
                    }
                    else {
                    ca.display();
                    }
                    break;
                    case 5:
                    break;
                    default:
                    System.out.println("Invalid Choice");
                    break;
                    }

                    if(choice==5) {
                    break;
                    }
                }
        }
}
```

LAB: 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 Student{
   public String usn,name;
   public int sem;
   public void inputStudentDetails(){
      Scanner sc=new Scanner(System.in);
      System.out.println("Enter the student usn:");
      usn=sc.nextLine();
      System.out.println("Enter the student name:");
      name=sc.nextLine();
      System.out.println("Enter student semester:");
      sem=sc.nextInt();
   }

   public void disply(){
      System.out.println("Student USN:"+usn);
      System.out.println("Student Name:"+name);
      System.out.println("Student Sem:"+sem);
   }
}

package CIE;
import java.util.Scanner;
public class Internals extends Student{
   public int marks[]=new int[5];
   public void inputCIEmarks(){
      Scanner sc=new Scanner(System.in);
      for(int i=0;i<5;i++){
         System.out.println("Enter the marks for subject "+(i+1)+":");
         marks[i]=sc.nextInt();
      }
   }
}
package SEE;
import CIE.Internals;
import java.util.Scanner;
public class Externals extends CIE.Internals{
   public int marks[];
   public int finalMarks[];

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

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

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

    public void displayFinalMarks(){
        inputStudentDetails();
        for(int i=0;i<5;i++){
            System.out.println("Subject "+(i+1)+" final marks:"+finalMarks[i]);
        }
    }
}
import SEE.Externals;
class PackageMain{
    public static void main(String args[]){
        int numOfStudent=2;
        Externals finalMarks[]=new Externals[numOfStudent];
        for(int i=0;i<numOfStudent;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<numOfStudent;i++){
            finalMarks[i].calculateFinalMarks();
            finalMarks[i].disply();
            finalMarks[i].displayFinalMarks();
        }
    }
}
```

LAB: 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.

```
public class WrongAge extends Exception{
   WrongAge(String msg){
      super(msg);
   }
}
import java.util.Scanner;
class InputScanner{
   Scanner sc;
   InputScanner(){
      sc=new Scanner(System.in);
   }
}
class Father extends InputScanner{
   int fatherAge;
   Father() throws WrongAge{
      System.out.println("Enter father's age:");
      fatherAge=sc.nextInt();
         if(fatherAge<0){
         throw new WrongAge("Age cannot be negative");
      }

   }
   void display(){
      System.out.println("Father's age:"+fatherAge);
   }
}
class Son extends Father{
   int sonAge;
   Son() throws WrongAge{
      System.out.println("Enter Son's age:");
      sonAge=sc.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");
      }

   }

   void display(){
      super.display();
      System.out.println("Son's age:"+sonAge);
   }

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

```
        try{
            Son son=new Son();
            son.display();
        }
        catch(WrongAge e){
            System.out.println(e);

        }
    }
}
```

LAB: 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 BMS_College_of_Engineering implements Runnable {
   public void run() {
      while (true) {
         try {
            System.out.println("BMS College of Engineering");
            Thread.sleep(10000); // Sleep for 10000 milliseconds (10 seconds)
         } catch (InterruptedException e) {
            e.printStackTrace();
         }
      }
   }
}

class CSE implements Runnable {
   public void run() {
      while (true) {
         try {
            System.out.println("CSE");
            Thread.sleep(2000); // Sleep for 2000 milliseconds (2 seconds)
         } catch (InterruptedException e) {
            e.printStackTrace();
         }
      }
   }
}

public class ThreadMain {
   public static void main(String[] args) {
      Thread t1 = new Thread(new BMS_College_of_Engineering());
      Thread t2 = new Thread(new CSE());
      t1.start();
      t2.start();
   }
}
```

LAB: 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();
            }
        });
    }
}
```

LAB: 10

Demonstrate Inter process Communication and deadlock.

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

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

	synchronized void put(int n) {
		while(valueSet)
			try {
				System.out.println("\nProducer waiting\n");
				wait();
			} catch(InterruptedException e) {
				System.out.println("InterruptedException caught");
			}
		this.n = n;
		valueSet = true;
		System.out.println("Put: " + n);
		System.out.println("\nIntimate Consumer\n");
		notify();
	}

}

class Producer implements Runnable {
	Q q;
	Producer(Q q) {
		this.q = q;
		new Thread(this, "Producer").start();
	}
	public void run() {
		int i = 0;
		while(i<5) {
			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<5) {
			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.");
	}
}
```

Deadlock:

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

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

class B {
	synchronized void bar(A a) {
		String name =Thread.currentThread().getName();
		System.out.println(name + " entered B.bar");
		try {
```

```
                Thread.sleep(1000);
            } catch(Exception e) {
                System.out.println("B Interrupted");
            }
            System.out.println(name + " trying to call A.last()");
            a.last();
        }
        void last() {
            System.out.println("Inside A.last");
        }
}

class Deadlock implements Runnable
{
        A a = new A();
        B b = new B();
        Deadlock() {
            Thread.currentThread().setName("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();
        }

}
```