B.M.S COLLEGE OF ENGINEERING BENGALURU
Autonomous Institute, Affiliated to VTU

Submitted in partial fulfillment of the requirements for record of

Object oriented java programming

(23CS3PCOOJ)

Submitted by:

SHREYA C Y

1BM22CS265

FACULTY INCHARGE:

DR SEEMA PATIL

Department of Computer Science and Engineering
B.M.S College of Engineering
Bull Temple Road, Basavanagudi, Bangalore 560 019
2023-2024

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();
        }

}
```

Lab Prgm - 01

12/12. Lab Program

1. Develop a java program that prints all real solutions to quadratic equation $ax^2+bx+c=0$. Read in a, b, c and use 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 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("Root1 =" + r1 + "Root2 =" + r2);

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

C:\Users\Admin\Desktop\1BM22CS265>java Quadratic

O/P: Enter the coefficients of a, b, c
5
6
8
Roots are imaginary
Root1 = 0.0 + i1.1135528125660042
Root1 = 0.0 - i1.1135528125660042

C:\Users\Admin\Desktop\1BM22CS265>java Quadratic
Enter coefficients of a, b, c
2
5
3
Roots are real and distinct
Root1 = -1.0 Root2 = -1.5

Shreya.C.Y
1BM22CS265

10
Sun
12/12/23

19/12. Lab Prgm - 02.

Lab prgm:

Q. 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.

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

$$CGPA = \frac{\Sigma[(\text{Course Credits})(\text{Grade Points})]}{\Sigma[\text{Course Credits}]}$$

```
import java.util.Scanner;
class subject
{
  int subjmarks;
  int grade;
  int credits;
}
```

```
class Student
{
    Subject subject[];
    String name;
    String usn;
    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 getStudentDetail()
{
    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 the credits for subject
        "+(i+1)+" :");
```

```
subject[i].credits = s.next();
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 < 9; 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.getStudentDetail();
    s1.getMarks();
    s1.computeSGPA();
    System.out.println("Name: " + s1.name);
    System.out.println("USN: " + s1.USN);
    System.out.println("SGPA: " + s1.SGPA);
  }
}
```

O/P:
Enter your name shreya
Enter your USN : 1BM22CS265
Enter your marks for subject 1: 90
Enter your credits for subject 1: 4
Enter your marks for subject 2: 99
Enter your credits for subject 2: 3
Enter your marks for subject 3: 89
Enter your credits for subject 3: 4
Enter your marks for subject 4: 90
Enter your credits for subject 4: 3
Enter your marks for subject 5: 99
Enter your credits for subject 5: 1
Enter your marks for subject 6: 96
Enter your credits for subject 6: 3
Enter your marks for subject 7: 87
Enter your credit for subject 7: 4
Enter your marks for subject 8: 91
Enter your credits for subject 8: 3
Name: shreya
USN: 1BM22CS265
SGPA: 9.68.

19/2/23

Shreya. C.Y
1BM22CS265

26/12

Lab Prgm - 03.

Lab Prgm 03:

```
import java.util.Scanner;
class Book
{
  String name;
  String author;
  int price;
  int numPages;
  public Book(String name, String author, int price,
              int numpages)
  {
     this.name = name;
     this.author = author;
     this.price = price;
     this.numPages = numPages;
  }
  public String toString()
  {
     String name = "Book name: " + this.name + "\n";
     String author = "Author name: " + this.author +
                     "\n";
     String price = "price : " + this.price + "\n";
     String numpages = "Number of Pages : " + this.
                       numPages + "\n";
     return name + author + price + numpages;
  }
}
class Main{
  public static void main(String[] args)
  {
     Scanner s = new Scanner(System.in);
```

```
System.out.print("Enter the number of books:");
int n = s.nextInt();
book b[] = new Book[n];
for(int i = 0; i<n; i++)
{
    System.out.print("Enter the name of book:");
    String name = s.next();
    System.out.print("Enter author of book:");
    String author = s.next();
    System.out.print("Enter price of the book:");
    int price = s.nextInt();
    System.out.print("Enter the number of
        pages of the book:");
    int numPages = s.nextInt();
    b[i] = new Book(name, author, price,
                    numPages);
}
System.out.println("\n Book Details:");
for(int i=0; i<n; i++)
{
    System.out.println("Book" + (i+1) + ":\n" +
                       b[i]);
}
}
}
```

O/P: Enter the number of books: 2
Enter name of the book: Alice
Enter author of the book: Pranjit
Enter price of the book: ~~Darkun~~ 500
Enter the number of pages of the book: 348
Enter name of the book: Darkun
Enter author of the book: Harry
Enter price of the book: 490
Enter number of pages of book: 265

Book Details:
Book 1: ~~Alice~~
Book name: Alice
Author name: Pranjit
Price: 500
Number of Pages: 348

Book 2:
Book name: Darkun
Author name: Harry
Price: 490
Number of Pages: 265

26/12/23

Shreya.C.Y
1BM22CS265

Lab Prgm - 04

8/1/24 Lab Prgm - 04

Q. Develop a java prgm to create an abstract class named Shape that contains two integers & 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 given shape

```
import java.util.scanner;
class Inputscanner
{
  Scanner s;
  Inputscanner()
  {
    s = 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.print("Enter length and width
```

```
        of the rectangle : ");
        a = s.nextDouble();
        b = s.nextDouble();
    }
    void displayArea()
    {
        System.out.println("Area of Rectangle :" + (a*b));
    }
}
class Triangle extends Shape
{
    void getInput()
    {
        System.out.print("Enter base and height of
        the triangle : ");
        a = s.nextDouble();
        b = s.nextDouble();
    }
    void displayArea()
    {
        System.out.println("Area of Triangle :" +
                                (0.5 * a * b));
    }
}
class Circle extends Shape
{
    void getInput()
    {
        System.out.print("Enter radius of the circle :");
        a = s.nextDouble();
    }
    void displayArea()
    {
```

```
        System.out.println("Area of circle:" +
                (Math.PI * a * a));
    }
}
public class Main
{
    public static void main(String[] args)
    {
        Rectangle rectangle = new Rectangle();
        Triangle triangle = new Triangle();
        Circle circle = new Circle();
        rectangle.getInput();
        rectangle.displayArea();
        triangle.getInput();
        triangle.displayArea();
        circle.getInput();
        circle.displayArea();
    }
}
```

op:
Enter length and width of rectangle: 9 8
Area of rectangle: 72.0
Enter base and height of the triangle: 4 6
Area of Triangle: 12.0
Enter radius of the circle: 8
Area of circle: 201.06192982974616

02/01/24

Shreya. C.Y
1BM22CS265

Lab prgm-05 (a)

3-1-24 Lab Pgm-05.

Develop a Java prgm to create a class Bank that maintains two kinds of accounts for its customers, one called savings account and 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 the minimum balance, impose penalty if necessary and update the balance.

```
import java.util.Scanner;
class Account{
    String customerName;
    String accountNumber;
    String accountType;
    double balance;
    Account(String customerName, String accountNumber,
    String accountType, double balance)
    {
        this.customerName = customerName;
        this.accountNumber = accountNumber;
        this.accountType = accountType;
        this.balance = balance;
    }
    void deposit(double amount)
    {
        balance += amount;
        System.out.println("Deposite successful,
            Updated balance:" + balance);
    }
    void displayBalance()
    {
        System.Out.println("Account Type:" + accountType);
        System.out.println("Customer Name:" + customerName);
        System.out.println("Account Number:" + accountNumber);
        System.out.println("Balance" + balance);
    }
}
void withdrawal()
{
    System.out.println("Enter withdrawal amount");
    int with.d = sc.nextInt();
```

```
if (with > balance)
{
    System.out.println("Insufficient balance");
}
else
{
    balance.withd;
}
}
void applyInterest()
{
    package java.lab;
    public class cur-act extends Accoint
    {
        cur-acct(string CustomerName, int accountNum,
        string account type)
        {
            super(Customer Name, Account Number,
                accounttype);
        }
void withdrawal()
{
    System.out.println("Enter withdrawal amt");
    int with = sc.nextInt();
    if (balance <= 2000)
    {
        double pen = balance/(0.06);
        System.out.println("Insufficient balance
        penalty to be paid
        balance + pen;
    else
    { balance.withd; } } }
```

```
package java.lab;
public class Sav_account extends Account;
{
    Sav_acct (String CustomerName, accountNum,
        accountType).
    {
        super( customer Name - AccountNumber,
                account type);
    }
void applyInterest()
{
    System.out.println("Enter interest rate").
    int rate = sc.nextInt();
    double interest = balance * (rate/100);
    balance + = interest;
    SOP("Balance after interest :" + balance);
}
}
package java.lab;
import java.util.Scanner;
public class Book
{
    public static void main(String args[])
    {
        Scanner sc = new Scanner(System.in).
        SOP("Enter customer name").
        String customerName = sc.nextLine();
        SOP("Enter account number");
        int acctNum = sc.nextLine();
        account ca = new
        Cur_acct (CustomerName, AccountNum,
                current amount);
```

```
Account ac = new
sav_acct(customer Name, account number, "savingacct");
int choice;
while (true)
{
  SOP(1. Deposit In 2. Withdrawal In 3. Compute Interest
      In 4. Display);
  choice = sc.nextInt();
  switch (choice)
  {
    case1: SOP("Enter the type of account In
      1. saving In 2. current In");
    int acc=sc.nextInt();
    if (acc==1)
    {
      sa.deposit();
    }
    else
    {
      ca.deposit();
    }
  break;
case2:
    SOP("Enter type of account");
    int acc=sc.nextInt();
    if (acc==1)
    {
      sa.withdrawal();
    }
    else
    { ca.withdrawal();
    }
  break;
```

```
case 3:
    sa.applyInterest();
    break;
case 4:
    sop("1.saving in 2.current");
    int acc2 = sc.nextInt();
    if(acc2 == 1)
    {
        sa.display();
    }
    else
    {
        ca.display();
    }
    break;
  }
 }
}
```

Op:

```
Enter customer Name: Shreya
Enter Acc no: 25487
- Menu -
1. Deposit
2. Withdrawal
3. Compute Interest
4. Display
1
Enter 1.saving 2.current
1
Enter amount 8000
Enter your choice 2
```

```
Enter 1. savings 2. Current
1
Enter amount: 2000
Balance amot: 5000
- Menu -
1. Deposit
2. Withdrawal
3. Compute interest
4. Display
4
1. saving 2 Current
1
Customer Name: Shreya
Acc No: 25487
Balance: 5000
```

Lab prgm - 06

3/1/24

LAB-06

Create a package CIE which has 2 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 semester of the student. Create another package SEE which has class external which is a derived class of Student. This class has an array that stores the SEE marks stored in five courses of 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 display()
{
    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 mark[] = new int[5];
    public void inputCIEmarks(){
        Scanner sc = new Scanner(System.in);
        for(int i=0; i<5; i++)
        {
            System.out.println("Enter marks for subject"
                    +(i+1)+ ":");
            marks[i] = sc.nextLine();
        }
    }
}
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(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(i=0; i<5; i++)
    {
        System.out.println("Subject" + (i+1) + "finalmarks:"
                + finalMarks[i]);
    }
}
}
import SEE.Externals;
class PackageMain{
  public static void main(String args[])
  {
    int numOfStudents = 2;
    ExternalFinalMarks[] = 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<numOfStudent; i++)
    {
        finalMarks[i].calculateFinalMarks();
        finalMarks[i].displayFinalMarks();
    }
}
```

O/P: Enter number of students : 3.
Student usn : 1BM2867
Student name : Saakshi
Student Semester : 3.
Enter the subject marks 1 : 45
Enter the subject marks 2 : 50
Enter the subject marks 3 : 48
Enter the subject marks 4 : 36.
Enter the subject marks 5 : 39
Enter the subject marks 1 : 89
Enter the subject marks 2 : 85
Enter the subject marks 3 : 91
Enter the subject marks 4 : 94
Enter the subject marks 5 : 91
Enter Student usn : 1BM2789
Student name : Nidhi
Student Semester : 3.
Enter subject marks 1 : 43
Enter subject marks 2 : 34

Enter subject mark 3: 45
Enter subject marks 4: 42
Enter subject marks 5: 37
Enter subject mark 1: 78
Enter subject mark 2: 76
Enter subjet marks 3: 87
Enter subject mark 4: 98
Enter subject marks 5: 91
Student usn: 1BM5621
Student name: Akshatha
Student semester: 7.
Enter subject mark 1: 40
Enter subject mark 2: 42
Enter subject mark 3: 49
Enter subject mark 4: 38
Enter subject marks 5: 35
Enter subject mark 1: 89
Enter subject mark 2: 94
Enter subject mark 3: 92
Enter subject mark 4: 87
Enter subject mark 5: 79

Total marks of student 1 (out of 100) is:
Subject 1 : 89
Subject 2 : 92
Subject 3 : 96
Subject 4 : 86
Subject 5 : 81

Total marks of student 2 (out of 100) is:
Subject 1: 82
Subject 2: 72

Subject 3: 88
Subject 4: 91
Subject 5: 82

Total marks of student 3 (out of 100) is:
Subject 1: 84
Subject 2: 89
Subject 3: 95
Subject 4: 81
Subject 5: 74

Shreya. C.4
1BM22CS265

30-1-24 Lab Prgm - 07.

Lab Prgm - 8.

Write a prgm that demonstrate handling of exceptions in heritance tree. Create a base called "Father" and derived class called "son" which extends base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge() when input age less than 0. In son class implement a constructor that cases both father and son's age and throws an exception if sons age is >= father's age.

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

```
    void display()
    {
        System.out.println("Father age is : " +
                                        fatherAge);
    }
}
class Son extends Father
{
    int sonAge;
    Son() throws WrongAge
    {
        super();
        Scanner s = new Scanner(System.in);
        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 == fatherAge)
        {
            throw new WrongAge("Son's age
            cannot be equal to Father's age");
        }
    else if(sonAge < 0)
    {
        throw new WrongAge("Age cannot be
                                "negative");
    }
}
```

```
void display()
{
    super.display()
    System.out.println(" Son's age is : " + sonAge);
}
}
public class Main
{
    public static void main (String[] args)
    {
        try {
            Sons = new Son();
            S.display();
        }
        catch (WrongAge e)
        {
            System.out.println(e.getMessage());
        }
    }
}
```

O/P: Enter Father's age : 40
Enter Son's age : 18
Father's age is : 40
Son's age is : 18

Enter Father's age : 30
Enter Son's age : 30
Son's age cannot be equal to Father's age.

Enter Father's age : -20
Age cannot be negative

Shreya. C.4
1BM22CS265

30.01.24

6/2/24 Lab Prgm - 08.

Lab Prgm - 08

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

```
class BMS implements Runnable{
    public void run(){
        while(true){
            try{
                System.out.println("BMS College of
                                    Engg");
                Thread.sleep(10000);
            }catch(InterruptedException e){
                e.printStackTrace();
            }
        }
    }
}
class CSE implements Runnable{
    public void run(){
        while(true){
            try{
                System.out.println("CSE");
                Thread.sleep(2000);
            }catch(InterruptedException e){
                e.printStackTrace();
            }
        }
    }
}
```

```
public class Main{
  public static void main(String[] args){
    Thread t1 = new Thread(new BMS College of Engg());
    Thread t2 = new Thread(new CSE());
     t1.start();
     t2.start();
  }
}
```

O/P: BMS College of Engg
CSE
CSE
CSE
CSE
CSE
BMS College of Engg
CSE
CSE
CSE
CSE
CSE
BMS College of Engg
CSE
CSE
CSE
CSE
CSE
BMS College of Engg

Shruya.C.4
1BM22CS265

20/2/24

Lab Prgm-09

09 Write a program that creates a user interface to perform integer divisions. The user enters two numbers in 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 ArithmeticException. Display the exception in a message dialog box.

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
class SwingDemo{
    SwingDemo(){
        JFrame jfrm = new JFrame("Divider App");
        jfrm.setSize(275, 150);
        jfrm.setLayout(new FlowLayout());
        jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JLabel jlab = new JLabel("Enter the divider and
                                  dividend");
        JTextField ajtf = new JTextField(8);
        JTextField bjtf = new JTextField(8);
        JButton button = new JButton("Calculate");
        JLabels err = new JLabel();
        JLabel alab = new JLabel();
        JLabel blab = new JLabel();
        JLabel anslab = new JLabel();
```

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

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

button.AddActionListerner(new ActionListerner(){
    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("\n Ans =" +ans);
        }
        catch(NumberFormatException e){
            alab.setText("");
            blab.setText("");
```

```
            anslab.setText("");
            err.setText("Enter Only Integers");
        }
        catch(ArithmaticException e){
            alab.setText("");
            blab.setText("");
            anslab.setText("");
            err.setText("B should be Non zero");
        }
    }
});

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

O/P :- 1. Enter the divider and divident

[10] [20]

[Calculate] A=10 B=20 Ans=0.

2. Enter the divider and divident

[10] [0.5]

[Calculate]

Enter only integer!

20.02.24

Functions:

1. JFrame.setDefaultCloseOperation(int Operation): The method sets operation that will happen by dfault when user initiates a "close" on this frame. In this case the frame will exit close.

2. JFrame.add(Component comp): This method adds the specific component to his container. It is used to add the JLabel, JTextfields and JButton to JFrame.

3. JTextfield.addActionListener(ActionListener l): This method adds an ActionListener to JTextfield. The actionlisterner is notified when an action occurs, which when user press enter.

4. JButton.addActionListener(ActionListener l): This method adds an ActionListerner to JButton. The Action Listener is notified when the button is pressed.

5. Integer.parseInt(String s): This method parses the specific string as an integer. It throws a NumberFormatException if the string cannot be parsed as an integer

6. JLabel.setText(String text): This method sets the text of the JLabel to especified text

7. ActionListener.actionPerformed(ActionEvent e): This is part of ActionListener interface. It is implemented by the anonymous ActionListener class in the code. The actionPerformed method is called when an action event occurs.

8. JFrame.setSize(int width, int height): This method sets the size of JFrame to the specified width and height.

9. JFrame.setLayout(LayoutManager layout): This set the layout manager the JFrame. In this case, a FlowLayout is used, which arranges components in left-to-right flow, similar to words in paragraph

10. SwingUtilities.invokeLater(): This method schedule the specific Runnable for execution on the event dispatching thread at a later time.

20.02.24

Shriya.C.4
1BM22CS265

6/2/24

Lab Pgm - 10.

Demonstrate inter process communication and deadlock.

```
class Q {
int n;
boolean valueSet = false;
synchronized int get() {
while (! valueSet)
 try {
    System.out.println("In Consumer waiting\n");
    wait();
  } catch (InterruptedException e) {
    System.out.println("InterruptedException caught");
  }
System.out.println("Got: " + n);
valueSet = false;
System.out.println("In Intimate Producer\n");
notify();
return n;
}
synchronized void put(int n){
while(valueSet)
 try {
  System.out.println("In Producer waiting\n");
  wait();
 } catch(InterruptedException e){System.out.println("caught");
   { System.out.println("InterruptedException caught");
   }
 this.n = n;
 valueSet = true;
```

```
        System.out.println("Put: "+n);
        System.out.println("In Intimate 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");
    }
}
```

O/P:

```
Put: 1
Got: 1
Put: 2
Got: 2
Put: 3
Got: 3
Put: 4
Got: 4
Put: 5
Got: 5
```

Shreya. C. 4
1BM22CS265

Deadlock.

```
class A {
  synchronized void for(B b){
    String name = thread.currentThread().getName();
    SOP (name + " enteredA.for");
    try {
      Thread.sleep(1000);
      }
    catch(exception e)
      {
        SOP ("AInterrupted");
        SOP(Name + "trying To call B.last()");
        b.last();
      }
    void last (){
      SOP ("Inside A.last");
    }
  }
class B {
  Synchronized vair bar (Aa){
    String name = Thread.CurrentThread().getname();;
    SOP(naue + " Entered B.bar");
    try{
      Thread.sleep(1000);
    }
    catch(exceptione)
      {
        SOP("B.Interrupted");
      }
    SOP(name + "trying To call A.last()");
        a.last(); }
```

```
void last(){
    SOP("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);
     SOP("Back in mainthread.");
    }
  public void run(){
     b.bar(a);
     SOP("Back in Other thread");
  }
  public static void main(String arg()){
     new Deadlock();
  }
}
```

O/P: Mainthread entered A.foo
Racing thread entered B.bar
Mainthread trying to call B.last()
Inside A.last
Back in Mainthread
Racing thread trying to call A.last()
Inside A.last
Back in Other thread.

13.02.24