VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Java Programming

(23CS3PCOOJ)

Submitted by

Ritesh Mohan Nayak(1BM23EC212)

in partial fulfillment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019

Sep-2024 to Jan-2025

B.M.S. College of Engineering,
Bull Temple Road, Bangalore 560019
(Affiliated To Visvesvaraya Technological University, Belgaum)
Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "Object Oriented Java Programming (23CS3PCOOJ)" carried out by **Ritesh Mohan Nayak(1BM23EC212),** who is bonafide student of **B.M.S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum. The Lab report has been approved as it satisfies the academic requirements in respect of an Object Oriented Java Programming (23CS3PCOOJ) work prescribed for the said degree.

Lab faculty Incharge Name Assistant Professor Department of CSE, BMSCE	Dr. Jyothi S Nayak Professor & HOD Department of CSE, BMSCE

Index

Sl. No.	Date	Experiment Title	Page No.
1	23/10/24	Week1- Quadratic Equation	4 - 8
2	23/10/24	Week2- SGPA Calculator	9 - 16
3	30/10/24	Week3- Book Class	17 - 20
4	30/11/24	Week4- Abstract Class Shape	21 - 26
5	30/10/24	Week5- Bank Class	27 - 34
6	13/11/24	Week6- Packages CIE and SEE	35 - 45
7	20/11/24	Week7- Exception handling in inheritance tree	47 - 54
8	27/11/24	Week8- Threads	55 - 58
9	27/11/24	Week9- User interface for integer divisions	59 - 65
10	27/11/24	Week10- IPS and Deadlock	65 - 78

Github Link:
https://github.com/Ritesh-Nayak-hindi/Java_lab/tree/main/Lab_programs

Program 1
Implement Quadratic Equation

```
import java.util.*;
import java.~~scan~~util.Scanner;
class quadratic {
    private double a,b,c;
    private double D;
    private double root1;
    private double root2;
    quadratic(){
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the values of a,b
and c according to the expression of the
quadratic eqn ax²+bx+c=0");
    this.a = sc.nextDouble();
    this.b = sc.nextDouble();
    this.c = sc.nextDouble();
    this.D = (b*b) - (4*a*c);
        }
}   ~~public class labpro{~~
```

```
public class solver {
    public static void main(String args[]){
        quadratic Q1 = new quadratic();
        Q1.solve();
        quadratic Q2 = new quadratic();
        Q2.solve();
        quadratic Q3 = new quadratic();
        Q3.solve();
    }
}
```

contd in class quadratic:

```
public void solve(){
    if(D < 0){
        System.out.println("The roots are
            not Real");
        return;
    }
    if(D == 0){
        root 1 = (-1*b)/(2*a);
        root 2 = (-1*b)/(2*a);
        System.out.println("The root are
identicle" + "\n" + "Root 1 : " + root 1 +
"\n Root 2 : " + root 2);
        return;
    }
```

```
if (D>0){
    root1 = (-b + Math.sqrt(D)) / (2*a) ;
    root2 = (-b - Math.sqrt(D)) / (2*a) ;
    System.out.println(" the Roots are
unique\n Root 1 : " + root1 + "\nROOT2 : " +
 root2) ;
}
```

OUTPUT:

enter the values of a, b & c according to the expression of quadratic eqn.

$ax^2+bx+c=0$

1

1

1

The roots are not real.

Code:

```
import java.util.Scanner;

class quadratic {
    private double a, b, c;
    private double D;
    private double root1;
    private double root2;

    quadratic() {
        Scanner sc = new Scanner(System.in);
        System.out.println("ENTER THE VALUES OF a, b AND c ACCORDING TO
THE EXPRESSION OF THE QUADRATIC EQUATION ax^2 + bx + c = 0 ");
        this.a = sc.nextDouble();
        this.b = sc.nextDouble();
        this.c = sc.nextDouble();
        this.D = (b * b) - (4 * a * c);
    }

    public void solve() {
        if (D < 0) {
            System.out.println("THE ROOTS ARE NOT REAL");
            return;
        }
        if (D == 0) {
            root1 = (-1 * b) / (2 * a);
            root2 = root1;  // Since both roots are the same
            System.out.println("THE ROOTS ARE IDENTICAL" + "\n" + "ROOT
1: " + root1 + "\n" + "ROOT 2: " + root2);
            return;
        }
        if (D > 0) {
            root1 = (-1 * b + (Math.sqrt(D))) / (2 * a);
            root2 = (-1 * b - (Math.sqrt(D))) / (2 * a);
            System.out.println("THE ROOTS ARE UNIQUE" + "\n" + "ROOT 1:
" + root1 + "\n" + "ROOT 2: " + root2);
            return;
        }
```

```
    }
}

public class Main {
    public static void main(String args[]) {
        quadratic Q1 = new quadratic();
        Q1.solve();
        quadratic Q2 = new quadratic();
        Q2.solve();
        quadratic Q3 = new quadratic();
        Q3.solve();
    }
}
```

OUTPUT:

```
ENTER THE VALUES OF a, b AND c ACCORDING TO THE EXPRESSION OF THE QUADRATIC EQUATION ax^2 + bx + c = 0
1
1
1
THE ROOTS ARE NOT REAL
ENTER THE VALUES OF a, b AND c ACCORDING TO THE EXPRESSION OF THE QUADRATIC EQUATION ax^2 + bx + c = 0
-1
1
23
THE ROOTS ARE UNIQUE
ROOT 1: -4.3218253804964775
ROOT 2: 5.3218253804964775
ENTER THE VALUES OF a, b AND c ACCORDING TO THE EXPRESSION OF THE QUADRATIC EQUATION ax^2 + bx + c = 0
10
10
10
THE ROOTS ARE NOT REAL
```

LAB PROGRAM 2:

Q] SGPA CALCULATOR IMPLEMENTATION

OBSERVATION BOOK:

```
import java.util.Scanner;

class student {
    int n;
    String usn, name;
    Float marks [];
    Float credits [];
    Scanner sc = new Scanner (System.in);
    student() {
        System.out.println ("Enter the no. of
                                subjects");
        this.n = sc.nextInt();
    }
  void create() {
    System.out.println ("Enter name");
    this.name = sc.next();
    sc.nextLine();
    System.out.println ("Enter usn");
    this.usn = sc.nextLine();
    this.marks = new Float [n];
    this.credits = new Float [n];
    for (int i = 0; i < n; i++) {
      System.out.println ("Enter marks of
        subject" + (i+1));
      marks [i] = sc.nextFloat();
```

```
System.out.println("Enter the credits of Subject"+
    (i+1));
credits[i] = sc.nextFloat();
}
int grade(float a){
    if (a<=100 && a>90){
        return 10;
    }
    else if(a>80){
        return 9;
    }
    else if( a>70){
        return 8;
    }
    else if(a>60){
        return 7;
    }
    else if(a>50){
        return 6;
    }
    else if (a>40){
        return 5;
    }
    else if(a>30){
        return 4;
    }
```

```
else if (a>=0){
    return 3;
}
else if (a >=10){
    return 2;
}
else if (a >=0){
    return 1;
}
else {
 System.out.println("The marks entered
   is invalid");
 return 0;
}

void sgpa() {
  float total_credits =0;
  for(int i = 0 ; i<n ; i++){
    total_credits += this.credits[i];
  }
  float totalgradepoints = 0;
  for (int i = 0 ; i<n ; i++){
    totalgradepoints += credits[i] *
        this.grade(marks[i]);
  }
```

```
float sgpa = total grade points / total credits;
System.out.println("Name : " + name);
    "      "      "      "  ("USN : " + usn);
    "      "      "      "  ("The sgpa of student is : " + sgpa);
}
}

public class Main {
    public static void main(String args[]) {
        Student S1 = new Student();
        ~~S1.credits~~
        S1.create();
        S1.sgpa();
    }
}
```

OUTPUT:

```
Enter the number of students
2
Enter the name of Student
Tom
Enter USN
123
```

CODE:

```
import java.util.Scanner;

class Student {
    int n;
    String usn, name;
    float marks[];
    float credits[];
    Scanner sc = new Scanner(System.in);

    Student() {
        System.out.println("ENTER THE NUMBER OF SUBJECTS");
        this.n = sc.nextInt();
    }

    void create() {
        System.out.println("Enter the name of the Student");
        this.name = sc.next();
        sc.nextLine();
```

```
        System.out.println("Enter the USN of the Student");
        this.usn = sc.nextLine();

        this.marks = new float[n];
        this.credits = new float[n];

        for (int i = 0; i < n; i++) {
            System.out.println("Enter the marks of subject " + (i + 1));
            marks[i] = sc.nextFloat();

            while (marks[i] < 0 || marks[i] > 100) {
                System.out.println("Invalid marks! Please enter valid
marks (0-100): ");
                marks[i] = sc.nextFloat();
            }

            System.out.println("Enter the credits of subject " + (i +
1));
            credits[i] = sc.nextFloat();

            while (credits[i] <= 0) {
                System.out.println("Invalid credits! Please enter valid
credits (>0): ");
                credits[i] = sc.nextFloat();
            }
        }
    }

    int grade(float a) {
        if (a <= 100 && a > 90) {
            return 10;
        } else if (a > 80) {
            return 9;
        } else if (a > 70) {
            return 8;
        } else if (a > 60) {
            return 7;
        } else if (a > 50) {
            return 6;
        } else if (a > 40) {
```

```
            return 5;
        } else if (a > 30) {
            return 4;
        } else if (a > 20) {
            return 3;
        } else if (a > 10) {
            return 2;
        } else if (a > 0) {
            return 1;
        } else {
            return 0;
        }
    }

    void sgpa() {
        float totalCredits = 0;
        float totalGradePoints = 0;

        for (int i = 0; i < n; i++) {
            totalCredits += credits[i];
            totalGradePoints += credits[i] * grade(marks[i]);
        }

        float sgpa = totalGradePoints / totalCredits;

        System.out.println("NAME: " + name);
        System.out.println("USN: " + usn);
        System.out.println("THE SGPA OF THE STUDENT IS: " + sgpa);
    }
}

public class labpro {
    public static void main(String[] args) {
        Student S1 = new Student();
        S1.create();
        S1.sgpa();
    }
}
```

OUTPUT:

```
riteshmohannayak@RITESHs-MacBook-Air ddl % javac main.java
riteshmohannayak@RITESHs-MacBook-Air ddl % java main
ENTER THE NUMBER OF SUBJECTS
3
Enter the name of the Student
Ritesh
Enter the USN of the Student
1BM23EC212
Enter the marks of subject 1
90
Enter the credits of subject 1
4
Enter the marks of subject 2
91
Enter the credits of subject 2
4
Enter the marks of subject 3
88
Enter the credits of subject 3
3
NAME: Ritesh
USN: 1BM23EC212
THE SGPA OF THE STUDENT IS: 9.363636
```

Program 3:

Q] Book Class

OBSERVATION BOOK:

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

class Book {
    String name;
    String author;
    int price;
    int num_pages;
    Book (Scanner sc){
    System.out.println("Enter Name of Book");
    sc.next();
    this.name = sc.nextLine();

    System.out.println("Enter author");
    this.author = sc.nextLine();
    System.out.println("price of book");
    this.price = sc.nextInt();
}
void get_details(){
    System.out.println("Obj details via
                        get fnx");
    " " " " ("Book_Name: " +
                        this.name);
    " " " " ("Author_name: " +
                        this.author);
```

```
" " " " ("Book_price:" + this.
                price);
}

@Override
public String toString(){
  return("Obj details via toString
  method\n" + "Book_Name: " + this.name
  + "\nBook_Price: " + this.price +
  "\nAuthor_Name: " + this.author);

  }
}

public class Main(){
  public static void main(String args[])
  {
  Scanner sc = new Scanner(System.in);
  System.out.println("no. of obj ");
  int n = sc.nextInt();
  Book B[] = new Book[n]
  for (int i = 0; i < n; i++){
      B[i] = new Book (sc);
  }
```

```
for (int i = 0 ; i<n ; i++){
  System.out.println("Details of obj "+
(i+1) + " is : ");
System.out.println(B[i].toString());
System.out.println();
}
Sc.close();
}

}
```

OUTPUT :

Enter number of objects required
1
Enter name of book
harry_potter
Author
J.K. Rowling
Price
120

The details of obj 1 is :
Obj details via toString method
Book_name : ~~potter~~ harry_potter
Author_name : jk rowling
Book_price = 120

OUTPUT:

```
[riteshmohannayak@RITESHs-MacBook-Air ddl % javac main.java
[riteshmohannayak@RITESHs-MacBook-Air ddl % java main
ENTER THE NUMBER OF OBJECTS REQUIRED
4
Enter details for Book 1
ENTER THE NAME OF THE BOOK
harrypotter
ENTER THE NAME OF THE AUTHOR
jk_rowling
ENTER THE PRICE OF THE BOOK
356
ENTER THE NUMBER OF PAGES IN THE BOOK
567
Enter details for Book 2
ENTER THE NAME OF THE BOOK
lord of rings
ENTER THE NAME OF THE AUTHOR
tolkien
ENTER THE PRICE OF THE BOOK
900
ENTER THE NUMBER OF PAGES IN THE BOOK
1000
Enter details for Book 3
ENTER THE NAME OF THE BOOK
kshitij
ENTER THE NAME OF THE AUTHOR
ncert
ENTER THE PRICE OF THE BOOK
50
ENTER THE NUMBER OF PAGES IN THE BOOK
75
Enter details for Book 4
ENTER THE NAME OF THE BOOK
Rich Dad Poor Dad
ENTER THE NAME OF THE AUTHOR
Robert Kioski
ENTER THE PRICE OF THE BOOK
350
ENTER THE NUMBER OF PAGES IN THE BOOK
789
THE DETAILS OF OBJECT 1 ARE:
OBJECT DETAILS VIA toString METHOD
BOOK_NAME: harrypotter
AUTHOR_NAME: jk_rowling
BOOK_PRICE: 356
NO_OF_PAGES_IN_BOOK: 567


THE DETAILS OF OBJECT 2 ARE:
OBJECT DETAILS VIA toString METHOD
BOOK_NAME: lord of rings
AUTHOR_NAME: tolkien
BOOK_PRICE: 900
NO_OF_PAGES_IN_BOOK: 1000


THE DETAILS OF OBJECT 3 ARE:
OBJECT DETAILS VIA toString METHOD
BOOK_NAME: kshitij
AUTHOR_NAME: ncert
BOOK_PRICE: 50
NO_OF_PAGES_IN_BOOK: 75


THE DETAILS OF OBJECT 4 ARE:
OBJECT DETAILS VIA toString METHOD
BOOK_NAME: Rich Dad Poor Dad
AUTHOR_NAME: Robert Kioski
BOOK_PRICE: 350
NO_OF_PAGES_IN_BOOK: 789
```

PROGRAM 4:

Q]Abstract class shape

OBSERVATION BOOK:

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

abstract class Shape{
    int a, b;
    shape(){
        this.a = 30;
        this.b = 99;
    }
    abstract void printArea();
}
```

```
class Rectangle extends Shape{
    int l, b;
    Rectangle (int l, int b){
        System.out.println("Inside Rect");
        this.l = l;
        this.b = b;
    }
  void printArea(){
    System.out.println("Area:" + (l*b));
  }
}
class Triangle extends Shape{
    float h, b;
    Triangle (float h, float b){
        System.out.println("Inside Triangle
        this.h = h;
        this.b = b;
    }
  void printArea(){
    System.out.println("Area" + (float)
                                (h*b*0.
  }
}
```

```
class circle extends Shape {
    float r;
    float pi = 3.147;
    circle (float r) {
        System.out.println("Inside
                circle");

        this.r = r;
    }
    void printArea() {
        System.out.println("Area" + (float)
                (pi * r*r));
    }
}

public class Main {
    public static void main(String args[])
    {
        Rectangle R1 = new Rectangle(10,10);
        Triangle T1 = new Triangle(10,1);
        Circle C1 = new circle(10);
        R1.printArea();
        T1.printArea();
        C1.printArea();
    }
}
```

OUTPUT :

Inside Rectangle
Inside Triangle
Inside circle
Area : 100
Area : 5.0
Area : 314.0

CODE:

```
import java.util.*;
import java.util.Scanner;
abstract class Shape{
    int a,b;
    Shape(){
        this.a=90;
        this.b=99;
    }
    abstract void printArea();//ABSTRACT METHODS SHOULD NEVER HAVE A
BODY OF ITS OWN;
}
class Rectangle extends Shape{
    int l,b;
    Rectangle(int l,int b){
        System.out.println("INSIDE RECTANGLE CLASS");
        this.l=l;
        this.b=b;
    }
    void printArea(){
        System.out.println("AREA OF THE RECTANGLE IS: "+(l*b));
```

```
    }
}
class Triangle extends Shape{
    float h,b;
    Triangle(float h,float b){
        System.out.println("INSIDE Triangle CLASS");
        this.h=h;
        this.b=b;
    }
    void printArea(){
        System.out.println("AREA OF THE Triangle IS:
"+(float)(h*b*0.5));
    }
}
class Circle extends Shape{
    float r;
    float pi=3.14f;
    Circle(float r){
        System.out.println("INSIDE Circle CLASS");
        this.r=r;
    }
    void printArea(){
        System.out.println("AREA OF THE Circle IS: "+(float)(pi*r*r));
    }
}
public class labpro{
    public static void main(String args[]){
        Rectangle R1=new Rectangle(10,10);
        Triangle T1= new Triangle(10,1);
        Circle C1=new Circle(10);
        R1.printArea();
        T1.printArea();
        C1.printArea();
    }
}
```

OUTPUT:

```
riteshmohannayak@RITESHs-MacBook-Air ddl % javac main.java
riteshmohannayak@RITESHs-MacBook-Air ddl % java main
INSIDE RECTANGLE CLASS
INSIDE Triangle CLASS
INSIDE Circle CLASS
AREA OF THE RECTANGLE IS: 100
AREA OF THE Triangle IS: 5.0
AREA OF THE Circle IS: 314.0
riteshmohannayak@RITESHs-MacBook-Air ddl %
```

PROGRAM 5:

Q] Bank Class

OBSERVATION BOOK:

```
class Account{
    string cust_name;
    int acc_num;
    string acc_type;
    double balance;

    Account (string cust_name,
    int acc_num, string acc_type,
    double initial_balance){

    this.cust_name = cust_name;
    this.acc_num = acc_num;
    this.acc_type = acc_type;
    this.balance = initial_balance;
    }

    void deposit (double amount){
        if (amount > 0){
            balance += amount;
            System.out.println("Amount
            deposited: " + amount);
        }
    else{
        System.out.println("Invalid dep
        amount.");
    }
```

```
void displayBalance (){
    System.out.println("Current Balance:
    + balance);
  }
}


class SavAcct extends Account {
    final double ir = 0.05;
    SavAcct (String cName, int accNo,
    double initialbalance){
      super (cName, accNo, "Savings",
        initial balance);
    }
    void getInterest (){
      double Interest = balance * ir;
      balance + = Interest;
      System.out.println ("Interest added
      + interest);
    }
    void withdraw (double Amount){
    if (Amount > balance){
        System.out.println ("Not enough
          balance");
    }
    ~~et~~
```

```
else {
        balance+=amount;
        System.out.println("withdrawn
          +amount);
        }
    }
}

class CurrAcc extends Account{
    final double minbalance = 500.0;
    final double servicecharge = 50.0;
    CurrAcct (String cname , int accN,
  double initial balance) {
        super (cName, accNo, "current",
        initial balance);
    }
  void checkminbalance () {
        if (balance < minbalance){
            balance -= service charge
            System.out.println ("Service
            charge imposed: " + service
            charge());
        }
    }
```

```
void withdraw (double amount) {
    if (amount > balance) {
        System.out.println ("Insufficient
            balance");
    }
    else {
        balance -= amount;
        checkMinimumBalance();
        System.out.println("Amount
        withdrawn: " + amount);
    }
}
}

public class Bank {
    public static void main
    (String args[]) {
        SavAcc sa = new SavAcc
                ("Ritesh", 101, 20000);

        CurrAcc ca = new CurrAcc
                ("Vilgex", 102, 1000);
        System.out.println("Savings account
        sa.deposit (500);
        sa.getInterest (2);
```

```
sa.withdraw(200);
sa.displaybalance();
System.out.println("Current Account
ca.deposit(200);
ca.withdraw(800);
ca.displayBalance();
}
}
```

OUTPUT:

Savings account
Deposited = 500.00
Interest added = 125.00
withdrawn = 200.00
current Balance = 2425.00

Current account
Deposited = 200.00
Service charge imposed = 50.00
Amount withdrawn = 800.0
current Balance = 350.0

CODE:

```
class Account {
    String customerName;
    int accountNumber;
    String accountType;
    double balance;

    Account(String customerName, int accountNumber, String accountType,
double initialBalance) {
        this.customerName = customerName;
        this.accountNumber = accountNumber;
        this.accountType = accountType;
        this.balance = initialBalance;
    }

    void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            System.out.println("Amount deposited: " + amount);
        } else {
            System.out.println("Invalid deposit amount.");
        }
    }

    void displayBalance() {
        System.out.println("Current balance: " + balance);
    }
}

class SavAcct extends Account {
    final double interestRate = 0.04;

    SavAcct(String customerName, int accountNumber, double
initialBalance) {
        super(customerName, accountNumber, "Savings", initialBalance);
    }

    void computeInterest() {
        double interest = balance * interestRate;
        balance += interest;
```

```
        System.out.println("Interest added: " + interest);
    }

    void withdraw(double amount) {
        if (amount > balance) {
            System.out.println("Insufficient balance.");
        } else {
            balance -= amount;
            System.out.println("Amount withdrawn: " + amount);
        }
    }
}

class CurAcct extends Account {
    final double minimumBalance = 500.0;
    final double serviceCharge = 50.0;

    CurAcct(String customerName, int accountNumber, double
initialBalance) {
        super(customerName, accountNumber, "Current", initialBalance);
    }

    void checkMinimumBalance() {
        if (balance < minimumBalance) {
            balance -= serviceCharge;
            System.out.println("Service charge imposed due to low
balance: " + serviceCharge);
        }
    }

    void withdraw(double amount) {
        if (amount > balance) {
            System.out.println("Insufficient balance.");
        } else {
            balance -= amount;
            checkMinimumBalance();
            System.out.println("Amount withdrawn: " + amount);
        }
    }
}
```

```java
public class Bank {
    public static void main(String[] args) {
        SavAcct savingsAccount = new SavAcct("Alice", 101, 1000.0);
        CurAcct currentAccount = new CurAcct("Bob", 102, 600.0);

        System.out.println("Savings Account:");
        savingsAccount.deposit(500);
        savingsAccount.computeInterest();
        savingsAccount.withdraw(200);
        savingsAccount.displayBalance();

        System.out.println("\nCurrent Account:");
        currentAccount.deposit(300);
        currentAccount.withdraw(700);
        currentAccount.displayBalance();
    }
}
```

OUTPUT:

```
riteshmohannayak@RITESHs-MacBook-Air ddl % javac main.java
riteshmohannayak@RITESHs-MacBook-Air ddl % java main
Savings Account:
Amount deposited: 500.0
Interest added: 60.0
Amount withdrawn: 200.0
Current balance: 1360.0

Current Account:
Amount deposited: 300.0
Service charge imposed due to low balance: 50.0
Amount withdrawn: 700.0
Current balance: 150.0
```

LAB PROGRAM 6:

Q] Packages CIE and SEE.

OBSERVATION:

```
CIE//Internals.java
package CIE;
import java.util.*;
import java.util.Scanner;
public class Internals extends Student{
    protected int marks[];
    protected int n;
    public Internal(){
        super();
        this.n=5;
        this.marks=new int[n];
    }
    public void set_date(Scanner sc){
        for(int i=0;i<this.n;i++){
            System.out.println("Enter the
            marks");
            this.marks[i]=sc.nextInt();
        }
    }
```

```
public void get_date() {
        display();
        system.out.println("the marks
          of the student in 5 courses");
        for (int i = 0; i<n ; i++) {
                System.out.println("Marks
                in subject" + (i+1) + "is :"
                + this.marks[i] );
        }
}

}
```

CIE/Student.java

```
package cIE;
import java.util.*;
import java.util.scanner;
public class Student {
        protected String USN;
        protected String Name;
        protected int sem;

    public Student() {
        Scanner sc = new Scanner(System.in)
        System.out.print("Enter USN")
```

```
this.USN = sc.next();
System.out.println("Enter Name");
this.name = sc.next();
System.out.println("Enter Semester");
this.Sem = sc.nextInt();
}
public void display(){
    System.out.println("Name : " + this.Name
+"\nUSN:" + this.USN +"\n Semester: "
+ this.Sem);
}

}
```

SEE/External.java

```
package SEE;
import java.util.*;
import java.util.Scanner;
import CIE.Student;

public class External extends Student{
    protected int marks[];
    protected int n;
    public External(){
```

```
        super();
        this.n = 5;
        this.marks = new int[5];
    }

public void set_data(Scanner sc){
        for(int i = 0; i<this.n; i++){
            System.out.println("Enter marks")
        this.marks[i] = sc.nextInt();
        }
}

public void get_data(){
        display();
        ~~System.out.println("Enter~~

    for(int i = 0; i<n; i++){
        System.out.print( +this.marks[i]
    }

}.

}
```

Main.java

```
import java.util.*;
import java.util.Scanner;
import CIE.Student;
import CIE.Internals;
import SEE.Externals;

public Class Main{
    public static void main(String args[])
    {
    Scanner sc= new Scanner(System.in);
    Internals I1 = new Internals();
    Externals E1 = new Externals();

    I1.set_date(sc);
    E1.set_date(sc);

    I1.get_date(sc);
    E1.get_date(sc);

    }
}
```

DATE / /
PAGE

OUTPUT:

Enter USN of Student 123
Enter Name of " Ritesh
Enter Semester 3
Enter marks of Student 10
" " " " 10
" " " " 10
" " " " 10
" " " " 10
" " " " 10
" " " " 10
" " " " 10
" " " " 10
" " " " 10
" " " " 10

Name : Ritesh
USN : 123
Semester : 3
Marks in Subject 1 is 10
" " " 2 is 10
" " " 3 is 10
" " " 4 is 10
" " " 5 is 10
" " " 1 is 10
" " " 2 is 10

" " " 3 is 10
" " " 4 is 10
" " " 5 is 10.

CODE:

```
package CIE;

public class Student {
    protected String usn;
    protected String name;
    protected int sem;

    // Constructor to initialize student details
    public Student(String usn, String name, int sem) {
        this.usn = usn;
        this.name = name;
        this.sem = sem;
    }

    // Display student details
    public void displayStudentInfo() {
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        System.out.println("Semester: " + sem);
    }
}
public class Internals extends Student {
    protected int[] internalMarks;
    public Internals(String usn, String name, int sem, int[]
internalMarks) {
        super(usn, name, sem);
        this.internalMarks = internalMarks;
    }
    public void displayInternalMarks() {
        System.out.println("Internal Marks for the student:");
        for (int i = 0; i < internalMarks.length; i++) {
            System.out.println("Course " + (i+1) + ": " +
internalMarks[i]);
        }
    }
}
package SEE;
import CIE.Student;
```

```java
public class External extends Student {
    protected int[] externalMarks;
    public External(String usn, String name, int sem, int[]
externalMarks) {
        super(usn, name, sem);
        this.externalMarks = externalMarks;
    }
    public void displayExternalMarks() {
        System.out.println("External Marks for the student:");
        for (int i = 0; i < externalMarks.length; i++) {
            System.out.println("Course " + (i+1) + ": " +
externalMarks[i]);
        }
    }
}
import CIE.Internals;
import SEE.External;
import java.util.Scanner;

public class StudentMarks {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter number of students: ");
        int n = scanner.nextInt();
        scanner.nextLine();
        Internals[] internalsArray = new Internals[n];
        External[] externalsArray = new External[n];
        for (int i = 0; i < n; i++) {
            System.out.println("\nEnter details for Student " + (i+1));
            System.out.print("USN: ");
            String usn = scanner.nextLine();
            System.out.print("Name: ");
            String name = scanner.nextLine();
            System.out.print("Semester: ");
            int sem = scanner.nextInt();
            scanner.nextLine();
            int[] internalMarks = new int[5];
            System.out.println("Enter internal marks for 5 courses:");
            for (int j = 0; j < 5; j++) {
                internalMarks[j] = scanner.nextInt();
```

```
            }
            scanner.nextLine();
            internalsArray[i] = new Internals(usn, name, sem,
internalMarks);
            int[] externalMarks = new int[5];
            System.out.println("Enter external marks (SEE) for 5
courses:");
            for (int j = 0; j < 5; j++) {
                externalMarks[j] = scanner.nextInt();
            }
            scanner.nextLine();
            externalsArray[i] = new External(usn, name, sem,
externalMarks);
        }
        System.out.println("\n--- Final Marks of Students ---");
        for (int i = 0; i < n; i++) {
            System.out.println("\nDetails of Student " + (i+1));
            internalsArray[i].displayStudentInfo();
            internalsArray[i].displayInternalMarks();
            externalsArray[i].displayExternalMarks();
            int totalInternalMarks = 0;
            for (int mark : internalsArray[i].internalMarks) {
                totalInternalMarks += mark;
            }
            int totalExternalMarks = 0;
            for (int mark : externalsArray[i].externalMarks) {
                totalExternalMarks += mark;
            }
            int finalMarks = totalInternalMarks + totalExternalMarks;
            System.out.println("Final Marks: " + finalMarks);
        }
        scanner.close();
    }
}
```

OUTPUT:

```
Student 1 Info:
USN: USN123
Name: Alice
Semester: 3
Internal Marks:
20 30 25 28 22
External Marks:
60 70 55 65 50
Final Marks (Internal + External):
80 100 80 93 72

Student 2 Info:
USN: USN124
Name: Bob
Semester: 3
Internal Marks:
18 25 20 23 28
External Marks:
50 65 60 58 45
Final Marks (Internal + External):
68 90 80 81 73
```

PROGRAM 7:

Q]Exception Handling in inheritance tree

OBSERVATION BOOK:

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

class WrongAgeException extends
Exception {
    public WrongAgeException (String m){
        super(m);
    }
}
```

```
class SonageException extends
Exception {
    public SonageException (String m) {
        super(m);
    }
}

class Father {
    int age;
    public Father (Scanner sc) throws
    WrongageException {
        System.out.println("Enter the age
            of the father");
        this.age = sc.nextInt();
        if (age<0) {
            throw new WrongageException
            ("The age of father can't
            be negative");
        }
    }
    public void print_age() {
        System.out.println("The age
            of father is: " + this.age);
    }
}
```

```
class Son extends Father {
int age;
Scanner sc = new Scanner (System.in);
public Son (Scanner sc) throws
Sonage Exception, Wrongage Exception
{
    super(sc);
    System.out.println("The age of
     father cannot be negative");

    System.out.println("Enter the
        son's age");
    this.age = sc.nextInt();

    if (this.age < 0) {
        throw new WrongageExcep.
        tion("The age of father
        cannot be negative");
    }

    if (this.age > super.age) {
    throw new SonageException(
        "The age of father can't
        be smaller than that
        of son");
    }
}
```

```
public void get_data(){
    System.out.println("Father Name:"
+ super.age + "\n Son Name:" +
this.age);
  }
}


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

    try{
    Father F = new Father(sc);
    F.printage();
    Son S = new Son(sc);
    S.get_data();
    }
    catch(WrongageException |
    SonageException e){
    System.out.println("Exception
    occured:" + e.getMessage());
    }
    finally{
        sc.close()
}   }
```

OUTPUT:

(1) Enter age of father
-90
Exception Occured: The age of father cannot be negative.

(2) Enter age of father
23
~~Enter~~
The age of father is: 23
Enter age of father: ~~23~~
23
Enter age of son
45
Exception occurred: The age of the father cannot be smaller than that of son.

CODE:

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

class WrongageException extends Exception{
public WrongageException(String m){
super(m);
}
}

class SonageException extends Exception{
public SonageException(String m){
super(m);
}
}
class Father{
int age;
public Father(Scanner sc) throws WrongageException{
System.out.println("ENTER THE AGE OF THE FATHER");
this.age =sc.nextInt();
if(age<0){
throw new WrongageException("THE AGE OF THE FATHER CANNOT BE NEGATIVE");
}
}
public int get_age(){

return this.age;
}
public void print_age(){
System.out.println("THE AGE OF THE FATHER IS: "+this.age);
}
}

class Son extends Father{
int age;
Scanner sc=new Scanner (System.in);
public Son(Scanner sc) throws SonageException,WrongageException {
super(sc);
System.out.println("ENTER THE AGE OF THE SON");
```

```
this.age=sc.nextInt();

if(this.age<0){
throw new WrongageException("THE AGE OF THE FATHER CANNOT BE NEGATIVE");

}

if(this.age>super.age){
throw new SonageException("THE AGE OF THE FATHER CANNOT BE SMALLER THAN
THAT OF THE SON");

}

}

public void get_data(){
System.out.println("FATHER AGE :"+super.age+"\nSON AGE: "+this.age);
}
}


public class exception{
public static void main(String args[]){
Scanner sc=new Scanner(System.in);
try{
Father F=new Father(sc);
F.print_age();
Son S=new Son(sc);
S.get_data();
}
catch(WrongageException | SonageException e){
System.out.println("Exception Occured: "+e.getMessage());
}

finally{
sc.close();
}
}
}
```

OUTPUT:

```
riteshmohannayak@RITESHs-MacBook-Air ddl % javac main.java
riteshmohannayak@RITESHs-MacBook-Air ddl % java main
ENTER THE AGE OF THE FATHER
12
THE AGE OF THE FATHER IS: 12
ENTER THE AGE OF THE FATHER
13
ENTER THE AGE OF THE SON
21
Exception Occured: THE AGE OF THE FATHER CANNOT BE SMALLER THAN THAT OF THE SON
riteshmohannayak@RITESHs-MacBook-Air ddl % javac main.java
riteshmohannayak@RITESHs-MacBook-Air ddl % java main
ENTER THE AGE OF THE FATHER
-90
Exception Occured: THE AGE OF THE FATHER CANNOT BE NEGATIVE
riteshmohannayak@RITESHs-MacBook-Air ddl %
```

PROGRAM 8:
Q] Threads
OBSERVATION BOOK:

```
import java.util.*;

class CSE extends Thread{
    public void ~~main~~ run(){
        try{
            while(true){
                System.out.println("CSE");
                Thread.sleep(2000);
            }
        }catch(InterruptedException e)
        {
            System.out.println(e.getMessage())
        }
    }
}
```

```
class BMS extends Thread{
    public void run(){
        try{
            while(true){
                System.out.println("BMS");
                Thread.sleep(10000);
            }
        }
        catch(InterruptedException e){
            System.out.println(e.getMessage
        }
    }
}

public class multi{
    public static void main
    (String args[]){
        CSE th1 = new CSE();
        BMS th2 = new BMS();
        th1.start();
        th2.start();
    }
}
```

OUTPUT

```
CSE
BMS
CSE
CSE
CSE
CSE
BMS
CSE
CSE
^C.
```

CODE:

```
import java.util.*;


class CSE extends Thread{
public void run(){
try{ while(true){
System.out.println("CSE");
Thread.sleep(2000);
}
}
catch(InterruptedException e){
System.out.println("EXCEPTION CAUGHT:"+e.getMessage());
}
```

```
}
}

class BMS extends Thread{
public void run(){
try{
while(true){
System.out.println("BMS");
Thread.sleep(10000);
}
}
catch(InterruptedException e){
System.out.println("EXCEPTION CAUGHT:"+e.getMessage());
}

}

}
public class multi{
public static void main(String args[]){
CSE th1=new CSE();
BMS th2=new BMS();
th1.start();
th2.start();
}
}
```

OUTPUT:

```
riteshmohannayak@RITESHs-MacBook-Air ddl % javac main.java
riteshmohannayak@RITESHs-MacBook-Air ddl % java main
CSE
BMS
CSE
CSE
CSE
CSE
BMS
CSE
CSE
CSE
CSE
CSE
BMS
CSE
CSE
CSE
CSE
CSE
BMS
CSE
CSE
CSE
CSE
CSE
^C%
riteshmohannayak@RITESHs-MacBook-Air ddl %
```

PROGRAM 9:

Q] User Interface for division of integers

OBSERVATION BOOK:

WEEK -9

CODE:

```
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");

    JTextFeild ejtf = new JTextField(8);
    JTextFeild bjft = 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(ers);
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.add ActionListener(l);
button.addAction Listener( new
ActionListener () {
public void actionPerformed (ActionE
vent evt){
```

```
try{
  int a = Integer.parseInt(ajtf.getText());
int b = Integer.parseInt(bjtf.getText());
int ans = a/b;
alab.setText("\nA = " + a);
blab.setText("\n = " + b);
anslab.setText("\nAns = " + ans);
}
catch (NumberFormatException e){
  alab.setText("");
  blab.setText("");
  anslab.setText("");

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

WEEK-9

Q] Write a program that creates a user interface to perform integer divisions. The user enter two numbers in the text. If Num2 were zero, the ~~excepti~~ program would throw an Arithmetic Exception Display the exception in message dialog box:

OUTPUT:

DIVIDER APP

Enter the divider & dividend:

[10] [10]

[calculate] A = 10 B = 10 Ans = 1

CODE:

```
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[]){
System.out.println("Ritesh Mohan Nayak 1BM23EC212");
// create frame on event dispatching thread
SwingUtilities.invokeLater(new Runnable(){
public void run(){
new SwingDemo();
}
});
}
}
```

OUTPUT:

```
riteshmohannayak@RITESHs-MacBook-Air ddl % javac main.java
riteshmohannayak@RITESHs-MacBook-Air ddl % java main
Ritesh Mohan Nayak 1BM23EC212
2024-12-03 20:14:12.742 java[74125:3894032] +[IMKClient subclass]: chose IMKClient_Modern
2024-12-03 20:14:12.742 java[74125:3894032] +[IMKInputSession subclass]: chose IMKInputSession_Modern
```

PROGRAM 10:

Q] INTERPROCESSING QUESTION

OBSERVATION BOOK:

a] Interprocessing code.

```
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(" Interupted Exception
                          caught");
    }
    System.out.println("Got: " + n);
    valueSet = false;
    System.out.println("\nProducer waiting\n");
    wait();
    }
    catch (Interupted Exception e){
    System.out.println("Interupted Exception
      caught");
    }
    this.n = n;
    valueSet = true;
```

```
System.out.println("Put : " + n);
System.out.println("\nIntimate Consumer");
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){
    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.");
  }
}
```

WEEK-10

Q] Demonstrate interprocess communication & deadlock.

OUTPUT INTERPROCESSING:

Press Control-C to stop.
Put : 0

Intimate consumer
Producer waiting
Got : 0
Intimate Producer
Put : 1
Intimate consumer
Producer waiting
consumed : 0
Got : 1
Intimate Producer
consumed : 1
Put : 2
Intimate consumer
Producer waiting
Got : 2
Intimate Producer
consumed : 2

Put : 3
Intimate consumer.

Producer waiting
Got : 3
Intimate Producer
consumed : 3
Put : 4
Intimate consumer
Producer waiting
Got : 4
Intimate consumer
consumed : 4
Put : 5
Intimate Producer
consumed : 5
Put : 6
Intimate consumer
Producer ~~G~~ waiting
Got : 6
Intimate Producer
consumed : 6
Put : 7

CODE:

```
class Q {
int n;
boolean valueSet = false;
synchronized int get() {
while(!valueSet)
try {
System.out.println("\nConsumer waiting\n");
wait();
} catch(InterruptedException e) {
System.out.println("InterruptedException caught");
}
System.out.println("Got: " + n);
valueSet = false;
System.out.println("\nIntimate Producer\n");
notify();
return n;
}
synchronized void put(int n) {
while(valueSet)
try {
System.out.println("\nProducer waiting\n");
wait();
} catch(InterruptedException e) {
System.out.println("InterruptedException caught");
}
this.n = n;
valueSet = true;
System.out.println("Put: " + n);
System.out.println("\nIntimate Consumer\n");
notify();
}
}
class Producer implements Runnable {
Q q;
Producer(Q q) {
this.q = q;
new Thread(this, "Producer").start();
}
```

```
public void run() {
int i = 0;
while(i<15) {
q.put(i++);
}
}
}
class Consumer implements Runnable {
Q q;
Consumer(Q q) {
this.q = q;
new Thread(this, "Consumer").start();
}
public void run() {
int i=0;
while(i<15) {
int r=q.get();
System.out.println("consumed:"+r);
i++;
}
}
}
class PCFixed {
public static void main(String args[]) {
Q q = new Q();
new Producer(q);
new Consumer(q);
System.out.println("Press Control-C to stop.");
}
}
```

OUTPUT:

```
riteshmohannayak@RITESHs-MacBook-Air ddl % javac main.java
riteshmohannayak@RITESHs-MacBook-Air ddl % java main
Press Control-C to stop.
Put: 0
Intimate Consumer

Producer waiting
Got: 0
Intimate Producer
Put: 1
Intimate Consumer

Producer waiting
consumed:0
Got: 1
Intimate Producer
consumed:1
Put: 2
Intimate Consumer

Producer waiting
Got: 2
Intimate Producer
consumed:2
Put: 3
Intimate Consumer

Producer waiting
Got: 3
Intimate Producer
consumed:3
Put: 4
Intimate Consumer

Producer waiting
Got: 4
Intimate Producer
consumed:4
Put: 5
Intimate Consumer

Producer waiting
Got: 5
Intimate Producer
consumed:5
Put: 6
Intimate Consumer

Producer waiting
Got: 6
Intimate Producer
consumed:6
Put: 7
Intimate Consumer

Producer waiting
Got: 7
Intimate Producer
Put: 8
Intimate Consumer

Producer waiting
consumed:7
Got: 8
Intimate Producer
consumed:8
Put: 9
Intimate Consumer

Producer waiting
Got: 9
Intimate Producer
consumed:9
Put: 10
Intimate Consumer

Producer waiting
Got: 10
Intimate Producer
consumed:10
Put: 11
Intimate Consumer

Producer waiting
Got: 11
Intimate Producer
consumed:11
Put: 12
Intimate Consumer

Producer waiting
Got: 12
Intimate Producer
consumed:12
Put: 13
Intimate Consumer

Producer waiting
Got: 13
Intimate Producer
consumed:13
Put: 14
Intimate Consumer
```

Q] DEADLOCK

OBSERVATION BOOK:

week-10:

Q] Dead lock code.

Demonstrate Inter process communication & dead lock.

```
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, "Racing Thread");
t.start();
a.foo(b);
System.out.println("Back in other Thread");
}
public static void main(String args[]){
    new Deadlock();
}
}
```

→ OUTPUT : Deadlock

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

Seen
27/11/24

CODE:

```
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 thisthread.
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();
}
}
```

OUTPUT:

```
riteshmohannayak@RITESHs-MacBook-Air ddl % javac main.java
riteshmohannayak@RITESHs-MacBook-Air ddl % java main
MainThread entered A.foo
RacingThread entered B.bar
MainThread trying to call B.last()
Inside A.last
Back in main thread
RacingThread trying to call A.last()
Inside A.last
Back in other thread
riteshmohannayak@RITESHs-MacBook-Air ddl %
```