VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT

on

OBJECT ORIENTED JAVA PROGRAMMING

Submitted by

SHREYAS GOWDA C (1BM23CS319)

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-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**" carried out by **SHREYAS GOWDA C (1BM23CS319),** 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 during the year 2024-25. The Lab report has been approved as it satisfies the academic requirements in respect of **Object-Oriented Java Programming Lab - (23CS3PCOOJ)** work prescribed for the said degree.

Dr. Nandhini Vineeth

Associate Professor,
Department of CSE,
BMSCE, Bengaluru

Dr. Kavitha Sooda

Professor and Head,
Department of CSE
BMSCE, Bengaluru

INDEX

GITHUIB LINK = https://github.com/shreyasgowdac-319/1BM23CS319-JAVALAB.git

LABORATORY PROGRAM – 01

Develop a Java program that prints all real solutions to the quadratic equation ax2+bx+c = 0. Read in a, b, c and use the quadratic formula. If the discriminate b2-4ac is negative, display a message stating that there are no real solutions.

```
import java.util.Scanner;

public class QuadraticEquation {
    public static void main(String[] args) {

        Scanner input = new Scanner(System.in);

        System.out.print("Enter coefficient a: ");
        double a = input.nextDouble();
        System.out.print("Enter coefficient b: ");
        double b = input.nextDouble();
        System.out.print("Enter coefficient c: ");
        double c = input.nextDouble();

        if (a == 0) {
            if (b != 0) {

                double root = -c / b;
                System.out.println("This is a linear equation. The solution is: " + root);
            } else if (c == 0) {

                System.out.println("The equation has infinitely many solutions.");
            } else {

                System.out.println("The equation has no solutions.");
            }
        } else {

            double discriminant = b * b - 4 * a * c;

            if (discriminant > 0) {
                double root1 = (-b + Math.sqrt(discriminant)) / (2 * a);
                double root2 = (-b - Math.sqrt(discriminant)) / (2 * a);
                System.out.println("The equation has two real solutions: " + root1 + " and " +
```

```
root2);
            } else if (discriminant == 0) {
                double root = -b / (2 * a);
                System.out.println("The equation has one real solution: " + root);
            } else {
                System.out.println("The equation has no real solutions.");
            }
        }

        input.close();
    }
}
```

OUTPUT

```
C:\Users\Abhishek\Desktop>java QuadraticEquation.java
Enter coefficient a: 5
Enter coefficient b: 6
Enter coefficient c: 9
The equation has no real solutions.

C:\Users\Abhishek\Desktop>java  QuadraticEquation.java
Enter coefficient a: 8
Enter coefficient b: -1
Enter coefficient c: -9
The equation has two real solutions: 1.125 and -1.0

C:\Users\Abhishek\Desktop>java QuadraticEquation.java
Enter coefficient a: 4
Enter coefficient b: -8
Enter coefficient c: -2
The equation has two real solutions: 2.224744871391589 and -0.22474487139158894

C:\Users\Abhishek\Desktop>java QuadraticEquation.java
Enter coefficient a: 5
Enter coefficient b: 6
Enter coefficient c: 3
The equation has no real solutions.
```

```
impurt java.util.Scanner;
class quadratic {
float d;
Scanner sc = new scanner(system.in);

void check()
{
system.out.println("enter the values of a, b and c");
  int a = sc.nextInt();
  int b = sc.nextInt();
  int c = sc.nextInt();

  if (a==b) {
  System.out.println("invalid equation");
}
else {
    d = b*b-4*a*c;

system.out.println(" the solutions are");
  if (d>0) {
    System.out.println(" the solutions are");
    system.out.println("roots are unique");
    double r1 = (-b+Math.sqrt(d)) / (2*a);
    double r2 = (-b-Math.sqrt(d)) / (2*a);
    system.out.println(r1+" "+r2);
  }
  if (d==0) {
    system.out.println("roots are equal");
     double r = -b/(2*a);
    system.out.println(r2+"+i"+r1+" "+r2+"-i"+
                                    r1);
```

```
    }
  }
}
}

public class main{
  public static void main(String[] args) {
     quadratic q1 = new quadratic();
     q1.check();
}
}
```

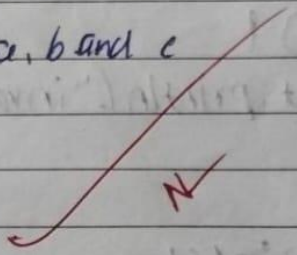
output:-

Enter the values of a, b and c

10

20

30



The solutions are

root1 and imaginary

-1.0+11.414213562373095i

LABORATORY PROGRAM – 02

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

```
import java.util.Scanner;

class Student {
    String usn;
    String name;
    int[] credits;
    int[] marks;


    void acceptDetails() {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter USN: ");
        usn = sc.nextLine();

        System.out.print("Enter Name: ");
        name = sc.nextLine();

        System.out.print("Enter the number of subjects: ");
        int n = sc.nextInt();

        credits = new int[n];
        marks = new int[n];

        System.out.println("Enter credits for each subject:");
        for (int i = 0; i < n; i++) {
            System.out.print("Credits for subject " + (i + 1) + ": ");
            credits[i] = sc.nextInt();
        }

        System.out.println("Enter marks for each subject:");
        for (int i = 0; i < n; i++) {
            System.out.print("Marks for subject " + (i + 1) + ": ");
            marks[i] = sc.nextInt();
```

```
        }
    }


    double calculateSGPA() {
         int totalCredits = 0;
        int weightedSum = 0;

        for (int i = 0; i < credits.length; i++) {
            int gradePoint = getGradePoint(marks[i]);
            weightedSum += gradePoint * credits[i];
            totalCredits += credits[i];
        }

        return (double) weightedSum / totalCredits;
    }


    int getGradePoint(int marks) {
        if (marks >= 90) return 10;
        else if (marks >= 80) return 9;
        else if (marks >= 70) return 8;
        else if (marks >= 60) return 7;
        else if (marks >= 50) return 6;
        else if (marks >= 40) return 5;
        else return 0;
    }


    void displayDetails() {
        System.out.println("\nStudent Details:");
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        System.out.println("SGPA: " + calculateSGPA());
    }

    public static void main(String[] args) {
        Student student = new Student();
        student.acceptDetails();
        student.displayDetails();
    }
}
```

OUTPUT

```
PS C:\Users\Abhishek\Desktop\1bm23cs309> javac Student.java
PS C:\Users\Abhishek\Desktop\1bm23cs309> java Student.java
Enter USN: 1BM23CS309
Enter Name: SHANKAR PUJAR
Enter the number of subjects: 3
Enter credits for each subject:
Credits for subject 1: 4
Credits for subject 2: 3
Credits for subject 3: 4
Enter marks for each subject:
Marks for subject 1: 89
Marks for subject 2: 82
Marks for subject 3: 75

Student Details:
USN: 1BM23CS309
Name: SHANKAR PUJAR
SGPA: 8.636363636363637
PS C:\Users\Abhishek\Desktop\1bm23cs309>
```

Date: 3-10-2024

Bafna Gold

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

```
import java.util.Scanner;
public class cgpa{
public static void main(String[] args){
    Scanner sc= new Scanner(System.in);

System.out.print("Enter the number of subjects:");
  int num subjects = sc.nextInt();

double [] grade points = new double [num subjects];
int [] credits = new int [num subjects];
 int total credits =0;
 double total =0;

for(int i=0; i<numsubjects; i++){
System.out.print("Enter the grade points for subject"+
      (i+1) +":");
   grade points [i] = sc.nextDouble();

System.out.print("Enter the credits for subject"+
      (i+1) +":");
   credits [i]= sc.nextInt();

total += grade Points [i] * credits [i];
 total credits += credits [i];
}
double cgpa = total/ total credits;
system.out.print("Your cgpa is:" + cgpa);
```

}
}

output:

Enter USN: 1bm23cs319
Enter Name: shreyas
Enter the number of subjects: 3
Enter credits for subject 1: 4
Enter the marks for subject 1: 3
Enter the credits for subject 2: 2
Enter the marks for subject 2: 89
Enter credits for subject 3: 4
Enter marks for subject 3: 98.

Student details:
USN: 1bm23cs319
Name: shreyas
subject and marks:
subject 1: Marks = 3, credits = 4
subject 2: marks = 89, credits = 2
subject 3: marks = 98, credits = 4

SGPA: 5.8

o/p seen
03/4/24

LABORATORY PROGRAM – 03

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{
 private String name;
 private String author;
 private double price;
 private int num_pages;

 public Book(String name,String author,double price,int num_pages){
  this.name=name;
  this.author=author;
 this.price=price;
 this.num_pages=num_pages;}
 public String getName(){
 return name;}
 public void setName(String name){
  this.name=name;}
 public String getauthor(){
  return author;}
 public void setauthor(String author){
  this.author=author;}
 public double getprice(){
  return price;}
 public void setprice(){
  this.price=price;}
 public int getnumpages(){
  return num_pages;}
 public void setnumpages(){
  this.num_pages=num_pages;}

 public String toString(){
  return("Book[name:"+name+" author:"+author+" price:"+price+"
```

```
pages:"+num_pages);}
}
public class Bookmain{
 public static void main(String[] args){
  Scanner sc=new Scanner(System.in);
  System.out.println("enter no of books");
  int n=sc.nextInt();
  sc.nextLine();
  Book[] books=new Book[n];
  for(int i=0;i<n;i++){
   System.out.println("enter the details of book"+(i+1)+":");
   System.out.print("enter name:");
   String name=sc.nextLine();
   System.out.print("enter author:");
   String author=sc.nextLine();
   System.out.print("enter price:");
   int price=sc.nextInt();
   System.out.print("enter the no of pages:");
   int num_pages=sc.nextInt();
   books[i]=new Book(name,author,price,num_pages);}
  System.out.print("\ndetails of books");
  for(int i=0;i<n;i++){
   System.out.println(books[i].toString());
  }
}
}
```

OUTPUT

```
Microsoft Windows [Version 10.0.22631.4460]
(c) Microsoft Corporation. All rights reserved.

C:\1BM23CS309>javac Bookmain.java

C:\1BM23CS309>java Bookmain.java
enter no of books
2
enter the details of book1:
enter name:the night flower
enter author:d.chandrashekar
enter price:250
enter the no of pages:400
enter the details of book2:
enter name:enter author:heli hogu karana
enter price:500
enter the no of pages:750

details of booksBook[name:the night flower author:d.chandrashekar price:250.0 pages:400
Book[name: author:heli hogu karana price:500.0 pages:750

C:\1BM23CS309>
```

Date: 19-10-2024

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 the methods to set and get the details of the object. Create a Java Program to read n Book objects.

```
import java.util.Scanner;

class Book {
    int price;
    string author;
    string name;
    int pages;

public Book(int price, string author, string name, int pages)
{   this.price = price;
    this.author = author;
    this.name = name;
    this.pages = pages;
}

public void setter() {
System.out.println("enter the price, author, name and
        pages of the book");
    Scanner sc = new Scanner(System.in);
      int price = sc.nextInt();
      string author = sc.nextInt();
      string name = sc.nextInt();
      int pages = sc.nextInt();
  }

public void getter() {
    system.out.println(price);
    System.out.println(author);
```

```
System.out.println(name);
System.out.println(page);
}

public String toString(){
    return "this is object";
}
}

public class pro{
  public static void main(String[] args){
    Scanner s1 = new Scanner(System.in);
System.out.println("enter the number of objects");
    int n=s1.nextInt();

Book[] b1 = new Book[n];
for(int i=0; i<n; i++){
 b1[i] = new book(200, "virat", "the century", 111);

 b1[i].getter();
 b1[i].setter();
 b1[i].getter();

System.out.println(b1[i]);

  }

}

}
```

output

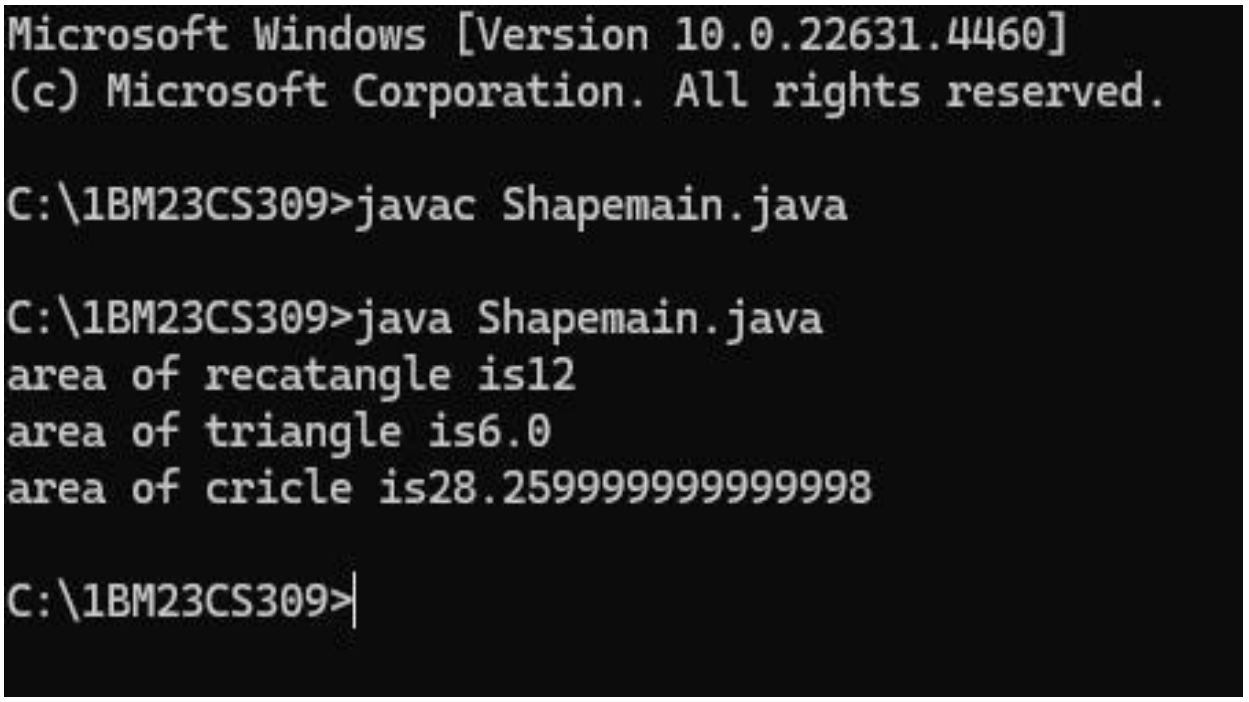
Enter the number of objects
2
Enter the price, author, name and pages of the book
999
shreyas
goat
269
999
Shreyas
goat
269
this is object
enter the price, author, name and pages of the book
899
Sanath
cold
49
899
Sanath
cold
49
this is object

LABORATORY PROGRAM – 04

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;

abstract class Shape{
 int d1,d2;
 abstract void printarea();}

class Rectangle extends Shape{
 Rectangle(int d1,int d2){
 this.d1=d1;
  this.d2=d2;}
 void printarea(){
  System.out.println("area of recatangle is"+(d1*d2));}
}

class Triangle extends Shape{
 Triangle(int d1,int d2){
 this.d1=d1;
  this.d2=d2;}
 void printarea(){
  System.out.println("area of triangle is"+(0.5*d1*d2));}
}

class Circle extends Shape{
 Circle(int d1,int d2){
 this.d1=d1;
  this.d2=1;}
 void printarea(){
  System.out.println("area of cricle is"+(3.14*d1*d1));}}

class Shapemain{
 public static void main(String[] args){
  Rectangle rectangle=new Rectangle(3,4);
  rectangle.printarea();
  Triangle triangle=new Triangle(3,4);
  triangle.printarea();
  Circle circle=new Circle(3,4); circle.printarea(); }}
```

OUTPUT


Microsoft Windows [Version 10.0.22631.4460]
(c) Microsoft Corporation. All rights reserved.
C:\1BM23CS309>javac Shapemain.java
C:\1BM23CS309>java Shapemain.java
area of recatangle is12
area of triangle is6.0
area of cricle is28.259999999999998
C:\1BM23CS309>

Date: 24-10-2024

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 contains only the method printArea() that prints the area of the given shape.

```
abstract class shape {
  int dim1;
  int dim2;

  abstract void printArea();
}
class Rectangle extends shape {
  public Rectangle(int length, int width) {
        this.dim1 = length;
        this.dim2 = width;
}

  void printArea() {
  int area = dim1 * dim2;
  System.out.println("Area of Rectangle." + area);
  }
}

class Triangle extends shape {
  public Triangle(int base, int height) {
     this.dim1 = base;
     this.dim2 = height;
```

Output:

Area of Rectangle: 50
Area of Triangle: 24.0
Area of circle: 153.93804002589985

Il seen
Sign
08/11/24

```
}

void printArea() {
    double area = 0.5 * dim1 * dim2;
System.out.println("Area of Triangle:" + area);
  }
}

class circle extends Shape {
  public circle(int radius) {
   this.dim1 = radius;
  this.dim2 = 0;
}

  void printArea() {
  double area = Math.PI * dim1 * dim1;
  System.out.println("Area of circle:" + area);
  }
}

public class Main {
  public static void main(String[] args) {
   Shape rectangle = new Rectangle(10, 5);
   Shape triangle = new Triangle(8,2);
   Shape circle = new circle(7);

   rectangle.printArea();
   triangle.printArea();
   circle.printArea();
  }
}
```

LABORATORY PROGRAM – 05

Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called savings account and the other current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed. Create a class Account that stores customer name, account number and type of account. From this derive the classes Cur-acct and Sav-acct to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks: a) Accept deposit from customer and update the balance. b) Display the balance. c) Compute and deposit interest d) Permit withdrawal and update the balance Check for the minimum balance, impose penalty if necessary and update the balance.

```
import java.util.Scanner;



class Account {
    protected String customerName;
    protected String accountNumber;
    protected String accountType;
    protected double balance;



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

```
    }

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

    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            System.out.println("Deposit Successful. Updated Balance: " + balance);
        } else {
            System.out.println("Invalid deposit amount.");
        }
    }
}

class SavAcct extends Account {
    private static final double INTEREST_RATE = 0.05;
    public SavAcct(String customerName, String accountNumber, double initialBalance)
{
        super(customerName, accountNumber, "Savings", initialBalance);
    }

    public void computeAndDepositInterest(int years) {
```

```
        double interest = balance * Math.pow(1 + INTEREST_RATE, years) - balance;
        balance += interest;
        System.out.println("Interest of " + interest + " deposited. Updated Balance: " +
balance);
    }

    public void withdraw(double amount) {
        if (amount > 0 && amount <= balance) {
            balance -= amount;
            System.out.println("Withdrawal Successful. Updated Balance: " + balance);
        } else {
            System.out.println("Invalid withdrawal amount or insufficient balance.");
        }
    }
}

class CurAcct extends Account {
    private static final double MIN_BALANCE = 1000.0;
    private static final double PENALTY = 50.0;

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

```
    public void withdraw(double amount) {
        if (amount > 0 && amount <= balance) {
            balance -= amount;
            System.out.println("Withdrawal Successful. Updated Balance: " + balance);
            if (balance < MIN_BALANCE) {
                balance -= PENALTY;
                System.out.println("Balance below minimum. Penalty of " + PENALTY + "
imposed. Updated Balance: " + balance);
            }
        } else {
            System.out.println("Invalid withdrawal amount or insufficient balance.");
        }
    }
}



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

        SavAcct savings = new SavAcct("Alice", "S123", 5000.0);



        CurAcct current = new CurAcct("Bob", "C456", 2000.0);

        System.out.println("Savings Account Operations:");
```

```
        savings.displayBalance();
        savings.deposit(2000);
        savings.computeAndDepositInterest(2);
        savings.withdraw(3000);

        System.out.println("\nCurrent Account Operations:");
        current.displayBalance();
        current.deposit(1000);
        current.withdraw(2500);
        current.withdraw(500);

        scanner.close();
    }
}
```

OUTPUT

```
PS C:\Users\Abhishek\Desktop\1bm23cs309> javac Bank.java
PS C:\Users\Abhishek\Desktop\1bm23cs309> java Bank.java
Savings Account Operations:
Account Balance: 5000.0
Deposit Successful. Updated Balance: 7000.0
Interest of 717.5 deposited. Updated Balance: 7717.5
Withdrawal Successful. Updated Balance: 4717.5

Current Account Operations:
Account Balance: 2000.0
Deposit Successful. Updated Balance: 3000.0
Withdrawal Successful. Updated Balance: 500.0
Balance below minimum. Penalty of 50.0 imposed. Updated Balance: 450.0
Invalid withdrawal amount or insufficient balance.
PS C:\Users\Abhishek\Desktop\1bm23cs309>
```

```
    }
    scanner.close();
  }
}
```

output:

```
choose account type:
1. Savings Account
2. current Account
1
Enter customer name:
Shreyas
Enter account number:
PIDS90
Enter interest rate for savings account:
10
Enter amount to deposit
20000

Amount deposited: 20000.0
updated balance: 20000.0
Interest added: 2000.0
Updated balance: 22000.0
Enter amount to withdraw:
2000
Amount withdrawn: 2000.0
Updated balance: 20000.0
```

```
saving a account :");
double interestRate = scanner.nextDouble();
SavAcct savAccount = new SavAcct(name, accNum, interestRate);

system.out.println("Enter amount to deposit:");
double deposit = scanner.nextDouble();
savAccount.deposit(deposit);

savAccount.computeAndDepositInterest();
System.out.println("Enter amount to withdraw:");
double withdrawAmount = scanner.nextDouble();
savAccount.withdraw(withdrawAmount);
}

else if (choice==2){
system.out.println("Enter minimum balance for current a/c:");
double minBalance = scanner.nextDouble();
system.out.println("Enter service charge for falling below
minimum balance:");
double serviceCharge = scanner.nextDouble();
curAcct curAccount = new curAcct(name, accNum, minBalance,
serviceCharge);

system.out.println("Enter amount to deposit:");
double deposit = scanner.nextDouble();
curAccount.deposit(deposit);

System.out.println("Enter amount to withdraw:");
double withdrawAmount = scanner.nextDouble();
curAccount.withdraw(withdrawAmount);
else {
    system.out.println("Invalid account type selected.");
```

```
system.out.println("Amount withdrawn:" +amount);
if (balance < minimumbalance){
    imposePenalty();
}
system.out.println("Updated balance:"+balance);
} else {
 system.out.println("Insufficient Balance");
}
}

print void imposePenalty() {
 balance -= serviceCharge;
System.out.println("Balance fell below minimum,
Service charge imposed:" + serviceCharge);
}
}

public class Bank {
 public static void main(String[] args) {
Scanner scanner = new Scanner(System.in);
System.out.println("choose account type: In1. Savings
    Account In2. Current Account");
int choice = scanner.nextInt();
scanner.nextLine();

System.out.println("Enter Name to new Account:");
String name = scanner.nextLine();
System.out.println("Enter account number:");
int accNum = scanner.nextInt();

if (choice == 1) {
System.out.println("Enter interest rate for
```

```
public void compoundAndDeposit(int n) {
double interest = balance * (interestRate / 100);
balance += interest;
System.out.println("Interest added: " + interest);
System.out.println("Updated balance: " + balance);
}

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

class CurAcct extends Account {
double minimumBalance;
double serviceCharge;

public CurAcct(String customerName, int accountNumber,
double minimumBalance, double serviceCharge) {
super(customerName, accountNumber, "Current");
this.minimumBalance = minimumBalance;
this.serviceCharge = serviceCharge;
}

public void withdraw(double amount) {
    if (amount <= balance) {
    balance -= amount;
```

```
this.accountNumber = accountNumber;
this.accountType = accountType;
this.balance = 0.0;
}

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

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

class SavAcct extends Account {
private double interestRate;

public SavAcct(String customerName, int accountNumber,
    double interestRate) {
    super(customerName, accountNumber, "Savings");
    this.interestRate = interestRate;
}
```

Date 7-11-2024.

Bafna Gold

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 facilities. The current account provide cheque book facilities but no interest, current account holder 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, acc num and type of acc. From this derive the classes current and sav-acct to make them more specific to their requirements. Include the necessary methods in order to achieve the following.

a) Let's deposit from customer and update the balance.
b) display the balance.
c) compute and deposit interest
d) permit withdrawal and update the balance.
Check min. impose penalty if necessary and update the bal.

```
import java.util.Scanner;
class Account{
String customerName;
int accountNumber;
String accountType;
double balance;

public Account(String customerName, int accountNumber,
String accountType){
    this.customerName = customerName;
```

LABORATORY PROGRAM – 06

Create a package CIE which has two classes- Student and Internals. The class Personal has members like usn, name, sem. The class internals 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.

Internals.java

```
import java.util.Scanner;


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


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

Externals .java

```
import java.util.Scanner;

public class Externals extends Internals {
```

```
    private int[] seeMarks = new int[5];
    private int[] finalMarks = new int[5];

    public void inputSEEmarks() {
        Scanner s = new Scanner(System.in);
        System.out.println("Enter SEE Marks for 5 subjects:");
        for (int i = 0; i < 5; i++) {
            System.out.print("Subject " + (i + 1) + ": ");
            seeMarks[i] = s.nextInt();
        }
    }

    public void calculateFinalMarks() {
        for (int i = 0; i < 5; i++) {
            finalMarks[i] = internalMarks[i] + seeMarks[i];
        }
    }

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

#Student.java

```
import java.util.Scanner;

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

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

```
    }
}
```

#Main.java

```
import java.util.Scanner;

class Main {
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        System.out.print("Enter number of students: ");
        int n = s.nextInt();

        Externals[] students = new Externals[n];

        for (int i = 0; i < n; i++) {
            System.out.println("\nEnter details for student " + (i + 1) + ":");
            students[i] = new Externals();
            students[i].inputStudentDetails();
            students[i].inputCIEmarks();
            students[i].inputSEEmarks();
            students[i].calculateFinalMarks();
        }

        System.out.println("\nFinal Marks of Students:");
        for (int i = 0; i < n; i++) {
            System.out.println("\nStudent " + (i + 1) + ":");
            students[i].displayFinalMarks();
        }
    }
}
```

OUTPUT

```
Enter number of students: 2

Enter details for student 1:
Enter USN: 1bm23cs012
Enter Name: abhishek
Enter Semester: 3
Enter Internal Marks for 5 subjects:
Subject 1: 23
Subject 2: 65
Subject 3: 56
Subject 4: 54
Subject 5: 98
Enter SEE Marks for 5 subjects:
Subject 1: 56
Subject 2: 98
Subject 3: 63
Subject 4: 98
Subject 5: 75

Enter details for student 2:
Enter USN: aadit
Enter Name: aadity
Enter Semester: 3
Enter Internal Marks for 5 subjects:
Subject 1: 56
Subject 2: 85
Subject 3: 98
Subject 4: 75
Subject 5: 86
Enter SEE Marks for 5 subjects:
Subject 1: 98
Subject 2: 56
Subject 3: 82
Subject 4: 87
Subject 5: 88

Final Marks of Students:
```

```
Enter Semester: 3
Enter Internal Marks for 5 subjects:
Subject 1: 56
Subject 2: 85
Subject 3: 98
Subject 4: 75
Subject 5: 86
Enter SEE Marks for 5 subjects:
Subject 1: 98
Subject 2: 56
Subject 3: 82
Subject 4: 87
Subject 5: 88

Final Marks of Students:

Student 1:
USN: 1bm23cs012
Name: abhishek
Semester: 3
Final Marks for 5 subjects:
Subject 1: 79
Subject 2: 163
Subject 3: 119
Subject 4: 152
Subject 5: 173

Student 2:
USN: aadit
Name: aadity
Semester: 3
Final Marks for 5 subjects:
Subject 1: 154
Subject 2: 141
Subject 3: 180
Subject 4: 162
Subject 5: 174
```

Create a package CIE which has two classes Student and Internals. The class personal has members like usn, name, sem. The class internals has an array that stores the internal marks scored in 5 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 :

```
public class Student
    String name;
    int usn;
    int sem;

public Student (String name, int usn, int sem) {
    this.name = name;
    this.usn = usn;
    this.sem = sem;
}
}
```

```
package CIE;

public class internals extends Student {

  public int marks[] = new int[5];

  public internals(String name, int usn, int sem, int[]
          marks) { super(name, usn, sem);
      this.marks = marks;
  }
}
```

```
package SEE;
 import CIE.Student;

 public class externals extends Student {
   public int smarks[] = new int[5];

   public externals(String name, int usn, int sem,
         int[] marks) { super(name, usn, sem);
          this.smarks = marks;
   }

}
```

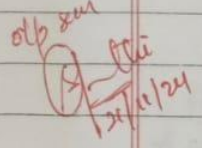
```
import CIE, internals;
import SEE, externals;
import java.util.Scanner;

public class cse {
    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        int cmarks[] = new int[5];
        int smarks[] = new int[5];
System.out.println("enter the num of students");
        int n = sc.nextInt();
for(int k=0; k<n; k++) {

System.out.println("enter the name of the student");
        String name = sc.next();
        System.out.println("enter the usn of the student");
        int usn = sc.nextInt();
        System.out.println("enter the sem of the student");
        int sem = sc.nextInt();

System.out.println("enter the cie marks of 5 subjects");
    for(int i=0; i<5; i++) {
        cmarks[i] = sc.nextInt();
    }
System.out.println("enter the see marks of 5 subjects");
for(int i=0; i<5; i++) {
    smarks[i] = sc.nextInt();
    }
internals o1 = new internals(name, usn, cmarks);
external o2 = new externals(name, usn, sem, smarks);
```

```
for (int i=1; i<=5; i++){
System.out.println("the total marks of student
    in sub" + i);
System.out.println(ol.marks[i-1] + o2.marks[i-1]);
}
}
}
```

output

enter the num of students
2
enter the name of the student.
sanath
enter the usn of the student.
123456.
enter the sem of the student.
1
enter the 5 marks of 5 subjects
49
48
49
50
50
enter the sec marks of 5 subjects
50
50
50
50
50.
the total marks of student in sub.1
99

the total marks of student in sub 2
98
the total marks of student in sub 3.
96
the total marks of student in sub 4
100
the total marks of student in sub 5.
100



LABORATORY PROGRAM – 07

Write a programthat 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=father’s age.

```
import java.util.Scanner;
class WrongAgeException extends Exception {
    public WrongAgeException(String message) {
        super(message);
    }
}

class SonAgeException extends Exception {
    public SonAgeException(String message) {
        super(message);
    }
}

class Father {
    private int age;
    public Father(int age) throws WrongAgeException {
        if (age < 0) {
            throw new WrongAgeException("Wrong age");
        }
        this.age = age;
    }
    public int getAge() {
        return age;
    }
}

class Son extends Father {
    private int sonAge;
    public Son(int fatherAge, int sonAge) throws WrongAgeException,
SonAgeException {
        super(fatherAge);
        if (sonAge >= fatherAge) {
            throw new SonAgeException("Son's age cannot be greater than or equal to
father's age");
        }
        this.sonAge = sonAge;
    }
    public int getSonAge() {
```

```
        return sonAge;
    }
}
public class FatherSon{
    public static void main(String[] args) {
        while(true){
            Scanner sc = new Scanner(System.in);
            System.out.print("Enter Father's Age: ");
            int fatherAge = sc.nextInt();
            System.out.print("Enter Son's Age: ");
            int sonAge = sc.nextInt();
            try {
                Son son = new Son(fatherAge, sonAge);
                System.out.println("Accepted Succesfully");
            }
            catch (WrongAgeException e) {
                System.out.println(e.getMessage());
            }
            catch (SonAgeException e) {
                System.out.println(e.getMessage());
            }
            System.out.println("Would you like to re-enter details (Y/n)");
            String input = sc.next();
            if (input.equalsIgnoreCase("n")) {
                break;
            }
        }
    }
}
```

```
PS C:\Users\Abhishek\Desktop\1bm23cs309> javac FatherSon.java
PS C:\Users\Abhishek\Desktop\1bm23cs309> java FatherSon.java
Enter Father's Age: 45
Enter Son's Age: 21
Accepted Succesfully
Would you like to re-enter details (Y/n)
y
Enter Father's Age: 56
Enter Son's Age: 25
Accepted Succesfully
Would you like to re-enter details (Y/n)
n
PS C:\Users\Abhishek\Desktop\1bm23cs309>
```

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 uses both father and son's age and throws an exception if son's age >= father age.

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

class Father {
    int fatherAge;

    public Father(int fatherAge) throws WrongAgeException {
        if (fatherAge < 0) {
            throw new WrongAgeException("Father's age cannot be negative.");
        }
        this.fatherAge = fatherAge;
    }
}

class Son extends Father {
    int sonAge;
```

```
public Son(int fatherAge, int sonAge) throws WrongAge
 Exception{ super(fatherAge);
 if (sonAge >= fatherAge){
  throw new WrongAgeException("Son's age cannot
be greater than or equal to Fathers Age.");
}
 if (sonAge < 0){
  throw new WrongAgeException("Son's age cannot
be negative.");
 }
 this.sonAge = sonAge;
}
}

public class ExceptionHandlingDemo{
  public static void main(String[] args){
   try{
 System.out.println("Creating a Father object...");
  Father father = new Father(40);
 System.out.println("Father create with age:"+ father.fatherAge);

 System.out.println("Creating a Son object...");
  Son son = new Son(40, 20);
 System.out.println("Son created with age:" + son.sonAge
} catch(WrongAgeException e){
   System.out.println("\n Testing invalid scenerios...");
 try{
  Father invalidFather = new Father(-10);
} catch(WrongAgeException e){
 System.out.println("Exception caught:"+e.getMessage());
}
```

```
try {
    Son invalidSon = new Son(30,40);
} catch (WrongAgeException e) {
    System.out.println("Exception caught: " +
    e.getMessage());
    }
}
```

output:

Enter father's age 50
Enter son's age 60
Son's age cannot be greater than or equal to father's age
SonAgeException.

write other two cases

Ok done
21/11/24

LABORATORY PROGRAM – 08

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

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

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

class thread{
    public static void main(String[] args) {

        FirstThread thread1 = new FirstThread();
        SecondThread thread2 = new SecondThread();


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

}


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

28-11-2021

Write a program which creates two threads, displaying "BMS college of engineering" once every ten seconds and another displaying "CSE" every two seconds.

```
class Thread {
public static void main (String args[]) {
Threads threadBMS = new thread (new DisplayBMS());
Thread thread CSE = new thread (new Display CSE());
thread BMS.start();
thread CSE.start();
}
}

class DisplayBMS implements Runnable {
  public void run() {
  try {
   while (true) {
   System.out.println("BMS college of Engineering");
   thread.sleep(10000);
   }
  }
  catch (InterruptedException e) {
   System.out.println("Interrupted" + e.getMessage());
  }
  }
}

class Display CSE implements Runnable {
 public void run() {
  try {
   while (true) {
      system.out.println("CSE");
       thread.sleep(2000);
```

```
    }
}
catch (InterruptedException e) {
System.out.println("Interrupted" + e.getMessage());
    }
  }
}
```

output:

```
BMS College of engineering
CSE
CSE
CSE
CSE
BMS College of engineering
CSE
CSE
CSE
CSE
CSE
```

o/p not seen

thi
20/12/24

LABORATORY PROGRAM - 9

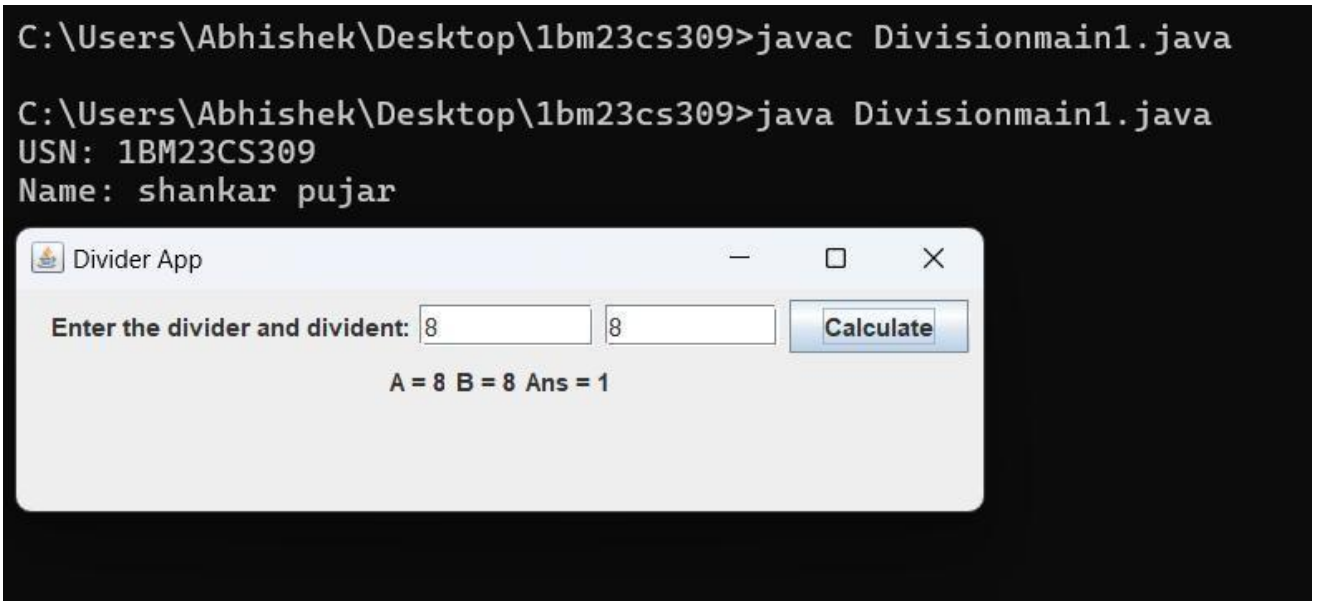
Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberFormatException. If Num2 were Zero, the program would throw an Arithmetic Exception Display the exception in a message dialog box.

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

public class DivisionMain1 extends Frame implements ActionListener
{
TextField num1,num2; Button dResult; Label outResult;
String out=""; double resultNum; int flag=0;

public DivisionMain1()
{
setLayout(new FlowLayout());

dResult = new Button("RESULT");
Label number1 = new Label("Number 1:",Label.RIGHT); Label number2 = new Label("Number 2:",Label.RIGHT); num1=new TextField(5);
num2=new TextField(5);

outResult = new Label("Result:",Label.RIGHT);

add(number 1);
add(num1); add(number 2);
add(num2); add(dResult)
```

```
;
add(outResul t);

num1.addActionListener(this); num2.addActionListener(this);
dResult.addActionListener(this); addWindowListener(new WindowAdapter()
{
public void windowClosing(WindowEvent we)
{

System.exit(0);
}
});
}
public void actionPerformed(ActionEvent ae)
{
int n1,n2; try
{
if (ae.getSource() == dResult)
{
n1=Integer.parseInt(num1.getText()); n2=Integer.parseInt(num2.getText());

/*if(n2==0)
throw new ArithmeticException();*/ out=n1+" "+n2+" ";

resultNum=n1/n2; out+=String.valueOf(result Num); repaint();

}
}
```

```
catch(NumberFormatException e1)
{
flag=1;
out="Number Format Exception! "+e1; repaint();
}
catch(ArithmeticException e2)
{
flag=1;
out="Divide by 0 Exception! "+e2; repaint();
}


}
public void paint(Graphics g)
{
if(flag==0)          g.drawString(out,outResult.getX()+outResult.getWidth(),outR
esult.getY()+outResult. getHeight()-8);
else g.drawString(out,1 00,200); flag=0;
}
```

OUTPUT



LABORATORY PROGRAM – 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<05) {
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<05) {
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

```
C:\Users\Abhishek\Desktop\1bm23cs309>javac PCFixed.java

C:\Users\Abhishek\Desktop\1bm23cs309>java PCFixed.java
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

Got: 4

Intimate Producer

consumed:4

C:\Users\Abhishek\Desktop\1bm23cs309>
```

DEADLOCK PROGRAM :

```
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) {
```

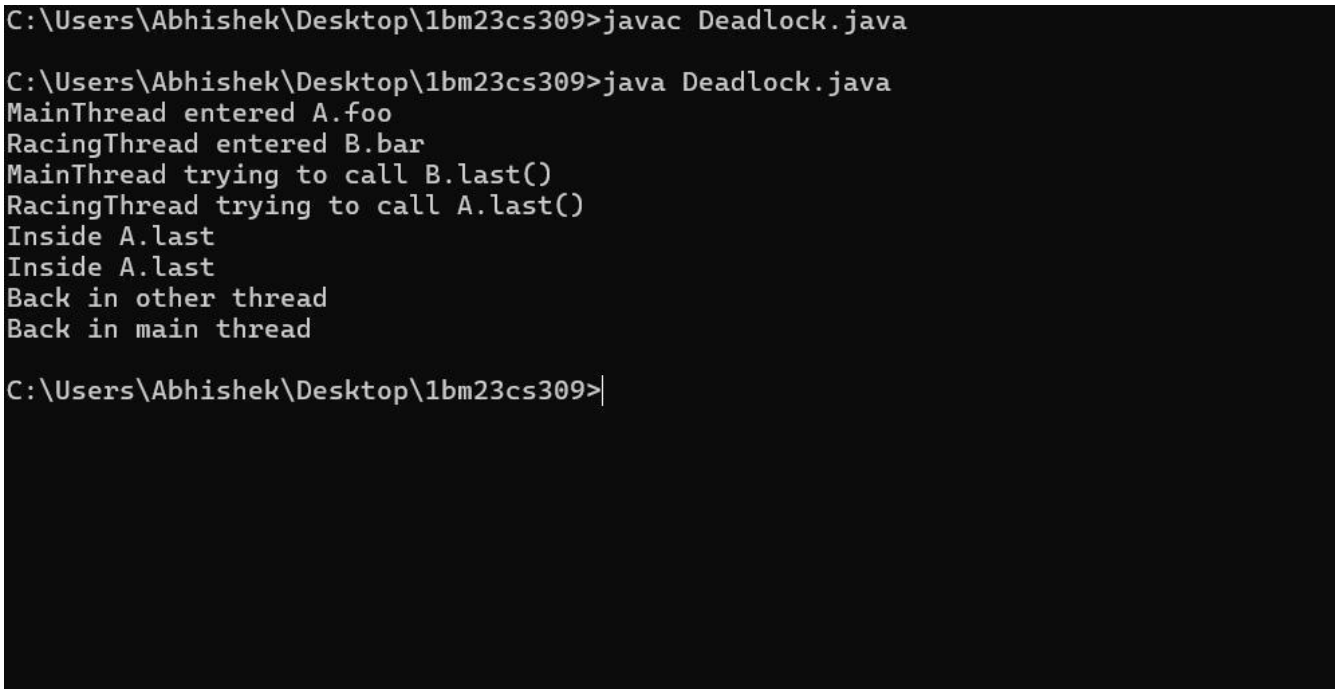
```
System.out.println("B Interrupted");

}

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();

}

}

OUTPUT



-: COMPLETE:-