VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT

on

OBJECT ORIENTED JAVA PROGRAMMING

Submitted by

SHRINANDA S DINDE (1BM23CS324)

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 **SHRINANDA S DINDE (1BM23CS324)** 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

Sl. No.	Date	Experiment Title	Page No.
1	11/10/24	Lab programme 1	1
2	18/10/24	Lab programme 2	5
3	24/10/24	Lab programme 3	13
4	24/10/24	Lab programme 4	19
5	07/11/24	Lab programme 5	26
6	21/11/24	Lab programme 6	37
7	21/11/24	Lab programme 7	44
8	28/11/24	Lab programme 8	48
9	24/12/24	Lab programme 9	51
10	24/12/24	Lab programme 10	56

LAB program 1

Q] 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.

Lab Program 1

Quadratic Equation

Q) Develop Java program that prints all real solutions to $ax^2+bx+c=0$. Read in a, b, c & use quadratic formula. If discriminant b^2-4ac is negative. Display message stating that there are no real solutions

```
import java.util.Scanner;
public class QuadraticEquation {
    public static void main (String[] args) {
        Scanner sc= new Scanner(System.in);
        System.out.println("Enter coefficient a: ");
        double a = sc.nextDouble();
        System.out.println("Enter coefficient b: ");
        double b = sc.nextDouble();
        System.out.println("Enter coefficient c: ");
        double c = sc.nextDouble();
        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);
            sout("Roots are real and distinct");
            Sout("Root 1=" + root1);
            sout(" Root2 =" + root2);
        }

        else if (discriminant == 0) {
            double root = -b / (2*a);
            sout ("Roots are real and equal");
            sout (" Root = " + root);
        }
```

```
        else{
            Sout ("There are no real solutions.");
        }
        sc.close();
    }
}
```

// output 1

```
Enter the coefficient a:
1
Enter the coefficient b:
-3
Enter the coefficient c:
2
The equation has two real and distinct roots:
Root 1: 2.0
Root 2: 1.0
```

// output 2

```
Enter the coefficient a:
1
Enter the coefficient b:
1
Enter the coefficient c:
1
There are no real solutions to the equation.
```

29/10/24

code:

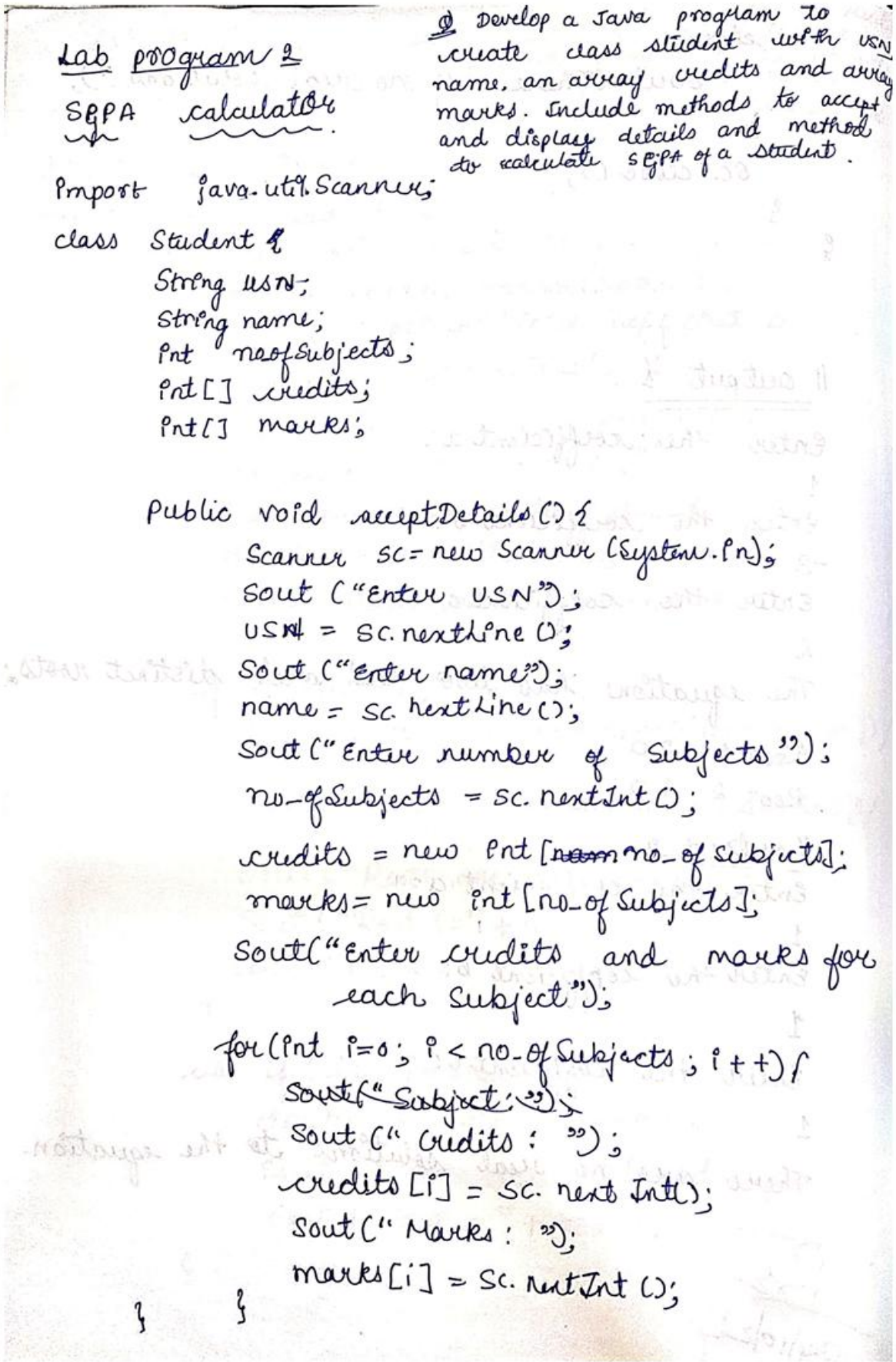
```
import java.util.Scanner;

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

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter coefficient a: ");
        double a = sc.nextDouble();

        System.out.println("Enter coefficient b: ");
        double b = sc.nextDouble();

        System.out.println("Enter coefficient c: ");
        double c = sc.nextDouble();

        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 roots are real and distinct.");
            System.out.println("Root 1 = " + root1);
            System.out.println("Root 2 = " + root2);
        } else if (discriminant == 0) {

            double root = -b / (2 * a);
            System.out.println("The roots are real and equal.");
            System.out.println("Root = " + root);
        } else {

            System.out.println("There are no real solutions.");
        }

        sc.close();
    }
}
```

//output:

```
 Enter coefficient a:
 1
Enter coefficient b:
-5
Enter coefficient c:
6
The roots are real and distinct.
Root 1 = 3.0
Root 2 = 2.0
PS E:\java lab>

Enter coefficient a:
1
Enter coefficient b:
-4
Enter coefficient c:
4
The roots are real and equal.
Root = 2.0
PS E:\java lab>

Enter coefficient a:
1
Enter coefficient b:
4
Enter coefficient c:
5
There are no real solutions.
PS E:\java lab>
```

LAB program 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



Lab program 2

SGPA calculator.

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

```
Import java.util.Scanner;
class Student {
    String USN;
    String name;
    int noofSubjects;
    int[] credits;
    int[] marks;

    Public void acceptDetails() {
        Scanner sc= new Scanner (System.in);
        Sout ("Enter USN");
        USN = sc.nextline();
        Sout ("Enter name");
        name = sc.nextLine();
        Sout (" Enter number of Subjects ");
        no_ofSubjects = sc.nextInt();
        credits = new int [no_of subjects];
        marks= new int [no_of Subjects];
        Sout("enter credits and marks for
              each Subject");
        for(int i=0; i < no_of Subjects ; i++){
            Sout(" Subject: ");
            Sout (" Credits : ");
            credits [i] = sc. next Int();
            Sout (" Marks : ");
            marks [i] = sc. nextInt ();
    }   }
```

```
Public void display(){
    Sout(" Student Details: ");
    Sout("USN: "+ USN);
    Sout("Name: " + name);
    Sout("credit and Marks: ");
    for(int i=1; i<=no_of Students; i++){
        Sout("Subject "+(i)+ ": credits="+
            credits[i] +" Marks = "+marks[i];
    }
}

private int gradepoints(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;
    }
```

```
Public double calculateSGPA(){
    int totalcredits = 0;
    int numerator = 0;
    for(int i=0; i< noofSubjects ; i++){
        int gradepoint = getGradePoint(marks[i]);
        Sum += gradepoint * credits[i];
        totalcredits += credits[i];
    }
    return (double) Sum / totalcredits;
}
}

Public class Main{
    Public Static void main(String[] args){
        Student student = new Student();
        Student.acceptDetails();
        Student.display();
        double sgpa = Student.calculateSGPA();
        Sout(sgpa);
    }
}
```

// output

Enter USN:

1BM23CS324

Enter Name:

Shrinanda

Enter number of subjects:

8

Enter credits and marks for each subject

credits : 4

marks : 95

credits : 4

marks : 97

credits : 3

marks : 91

credits : 3

marks : 87

credits : 3

marks : 86

credits : 1

marks : 91

credits : 1

marks : 95

credits : 1

marks : 99

Student Details :

USN : 1BM23CS324

Name : Shrinanda.

Credits and Marks:

Subject 1: credits = 4 , Marks = 95
Subject 2: credits = 4 , Marks = 91
Subject 3: credits = 3 Marks = 91
Subject 4: credits = 3 Marks = 87
Subject 5: credits = 3 Marks = 86
Subject 6: credits = 1 Marks = 91
Subject 7: credits = 1 Marks = 95
Subject 8: credits = 1 Marks = 99

9.7

24/10/24

CODE:

```
import java.util.Scanner;

class Student {

    String usn;
    String name;
    int numSubjects;
    int[] credits;
    int[] marks;


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

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

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

        System.out.println("Enter number of subjects: ");
        numSubjects = sc.nextInt();

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

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


    public void displayDetails() {
        System.out.println("\nStudent Details:");
        System.out.println("USN: " + usn);
        System.out.println("Name: " + name);
        System.out.println("Credits and Marks:");
```

```
        for (int i = 0; i < numSubjects; i++) {
            System.out.println("Subject " + (i + 1) + ": Credits = " + credits[i
        }
    }


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

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

        return (double) weightedSum / totalCredits;
    }


    private 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; // Fail grade
        }
    }
}

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

        Student student = new Student();
        student.acceptDetails();
            student.displayDetails();
```

```
        double sgpa = student.calculateSGPA();
        System.out.printf("SGPA: %.2f\n", sgpa);
    }
}
```

OUTPUT:

```
Enter USN:
1BM23CS324
Enter Name:
shrinanda
Enter number of subjects:
3
Enter credits and marks for each subject:
Subject 1:
Credits: 3
Marks: 70
Subject 2:
Credits: 4
Marks: 90
Subject 3:
Credits: 2
Marks: 99

Student Details:
USN: 1BM23CS324
Name: shrinanda
Credits and Marks:
Subject 1: Credits = 3, Marks = 70
Subject 2: Credits = 4, Marks = 90
Subject 3: Credits = 2, Marks = 99
SGPA: 9.33
PS E:\java lab>
```

LAB program 3

Create a class Book which contains four members: name, author, price, num_pages. Include a constructor to set the values for the members. Include methods to set and get the details of the objects. Include a toString( ) method that could display the complete details of the book. Develop a Java program to create n book objects.

Lab program 3] → Q] create a class Book which contains four memb. name, author, price, num_pages. Include constructor. Include methods to set and get the details of the objects. Include toString() method that could display the complete details of the book. Develop for n 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 (double price) {
        this.price = price;
    }

    public int getNumPages() {
        return num_pages;
    }

    public void setNumPages (int num_pages) {
        this.num_pages = num_pages;
    }

    Public String toString () {
        return "Book Name: " + name +
               "\n Author: " + author +
               "\n Price : ₹" + price +
               "\n Number of Pages: " + num_pages;
    }
}

Public class Main {
    public static void main (String[] args) {
        Scanner sc = new Scanner (System.in);
        System.out.println ("Enter the number
         of books: ");
        int n = sc.nextInt();
        sc.nextLine();

        Book[] books = new Book[n];
```

Enter number of pages: 1000
Enter details of book 2
Enter book name:
B
Enter Author name:
abc
Enter book price:
50
Enter number of pages: 500

book details:
Book name: A author: xyz price: 234
number of pages: 1000
Book name: B author: abc book price: 50
number of pages: 500.

24/10/24

```
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(double price) {
        this.price = price;
    }

    public int getNumPages() {
        return num_pages;
    }
```

```
    public void setNumPages(int num_pages) {
        this.num_pages = num_pages;
    }

    @Override
    public String toString() {
        return "Book Name: " + name + "\nAuthor: " + author + "\nPrice:
        " + price + "\nNumber of Pages: " + num_pages;
    }
}

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

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

        Book[] books = new Book[n];

        for (int i = 0; i < n; i++) {
            System.out.println("\nEnter details of book " + (i + 1) + ":");

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

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

            System.out.print("Enter price: ");
            double price = sc.nextDouble();

            System.out.print("Enter number of pages: ");
            int num_pages = sc.nextInt();
            sc.nextLine();

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

        System.out.println("\n--- Book Details ---");
        for (int i = 0; i < n; i++) {
            System.out.println("\nDetails of book " + (i + 1) + ":");
            System.out.println(books[i].toString());
        }
```

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

OUTPUT:

```
Enter the number of books:
2

Enter details of book 1:
Enter book name: Alchemist
Enter author name: paulo Coelho
Enter price: 289
Enter number of pages: 100

Enter details of book 2:
Enter book name: A Good Girl's Guide to Murder
Enter author name: Holly Jackson
Enter price: 350
Enter number of pages: 300

--- Book Details ---

Details of book 1:
Book Name: Alchemist
Author: paulo Coelho
Price: 289.0
Number of Pages: 100

Details of book 2:
Book Name: A Good Girl's Guide to Murder
Author: Holly Jackson
Price: 350.0
Number of Pages: 300
PS E:\java lab>
```

LAB program 4

Develop a Java program to create an abstract class named Shape that contains two integers and an empty method named printArea( ). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contain only the method printArea( ) that prints the area of the given.

Q) Lab program 4

Develop a Java program to create an abstract class named shape that contains two integers and an empty method named printArea(). Provide three classes named Rectangle, Triangle and circle such that each one of the classes extends the class shape. Each one of the classes contain only the method PrintArea() that prints the area of the given shape.

```
import java.util.*;
abstract class shape{
        Scanner sc = new Scanner (System.in);
        int side1;
        int side2;
        abstract void printArea ();
}
class rectangle extends shape{
        void printArea(){
                sout ("Enter length and breadth:");
                side1 = sc.nextInt();
                side2 = sc.nextInt();
                sout ("Area of Rectangle" + (side1 * side2));
        }
}
class triangle extends shape{
        void printArea (){
                System.out.println("Enter base and Height:");
                side1 = sc.nextInt();
                side2 = sc.nextInt();
                sout ("Area of Triangle: " + (0.5*side1*side2));
        }
```

```
    void printArea() {
        sout("Enter Radius of circle:");
        side1 = sc.nextInt();
        sout("Area of circle: " + (3.14 * side1 * side1));
    }
}

class c
public class print {
    public static void main(string [] args) {
        sout("enter shape of your choice:");
        Scanner sc = new Scanner(System.in);
        while (true) {
            sout("1. Rectangle \n 2. Triangle \n
                3. Circle \n 4. exit");
            int choice = sc.nextInt();
            switch(choice) {
                case 1:
                rectangle rt = new rectangle();
                 rt.printArea();
                 break;
                case 2:
                triangle t = new triangle();
                t.printArea();
                 break;
                case 3:
                Circle C = new Circle();
                c.printArea();
                break;
                case 4:
                sout("Exiting the program...");
                System.exit(0);
            }
        }
    }
```

output

```
Enter Shape of your choice:
1. Rectangle
2. Triangle
3. Circle
4. Exit
1
Enter length and Breadth:
2
4
Area of Rectangle : 8
1. Rectangle
2. Triangle.
3. Circle
4. Exit.
2
Enter base and height:
Area of Triangle: 5.0
1. Rectangle
2. Triangle
3. Circle
4. Exit
3
6
113.0394
1 Rectangle
2. Triangle
3. Circle
4. Exit
4
Exiting the program....
```

CODE:

```
import java.util.Scanner;

abstract class Shape {
    int dimension1;
    int dimension2;

    public Shape(int dim1, int dim2) {
        this.dimension1 = dim1;
        this.dimension2 = dim2;
    }

    abstract void printArea();
}

class Rectangle extends Shape {
    public Rectangle(int length, int width) {
        super(length, width);
    }

    @Override
    void printArea() {
        int area = dimension1 * dimension2;
        System.out.println("Rectangle Area: " + area);
    }
}

class Triangle extends Shape {
    public Triangle(int base, int height) {
        super(base, height);
    }

    @Override
```

```
    void printArea() {
        double area = 0.5 * dimension1 * dimension2;
        System.out.println("Triangle Area: " + area);
    }
}

class Circle extends Shape {
    public Circle(int radius) {
        super(radius, 0);
    }

    @Override
    void printArea() {
        double area = Math.PI * dimension1 * dimension1;
        System.out.println("Circle Area: " + area);
    }
}

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

        System.out.println("Choose a shape to calculate area:");
        System.out.println("1. Rectangle");
        System.out.println("2. Triangle");
        System.out.println("3. Circle");
        int choice = scanner.nextInt();

        switch (choice) {
            case 1:
                System.out.print("Enter length of Rectangle: ");
                int length = scanner.nextInt();
                System.out.print("Enter width of Rectangle: ");
                int width = scanner.nextInt();
                Rectangle rectangle = new Rectangle(length, wid
                rectangle.printArea();
```

```
                break;

            case 2:
                System.out.print("Enter base of Triangle: ");
                int base = scanner.nextInt();
                System.out.print("Enter height of Triangle: ");
                int height = scanner.nextInt();
                Triangle triangle = new Triangle(base, height);
                triangle.printArea();
                break;

            case 3:
                System.out.print("Enter radius of Circle: ");
                int radius = scanner.nextInt();
                Circle circle = new Circle(radius);
                circle.printArea();
                break;

            default:
                System.out.println("Invalid choice. Please choos
                break;
        }

        scanner.close();
    }
}
```

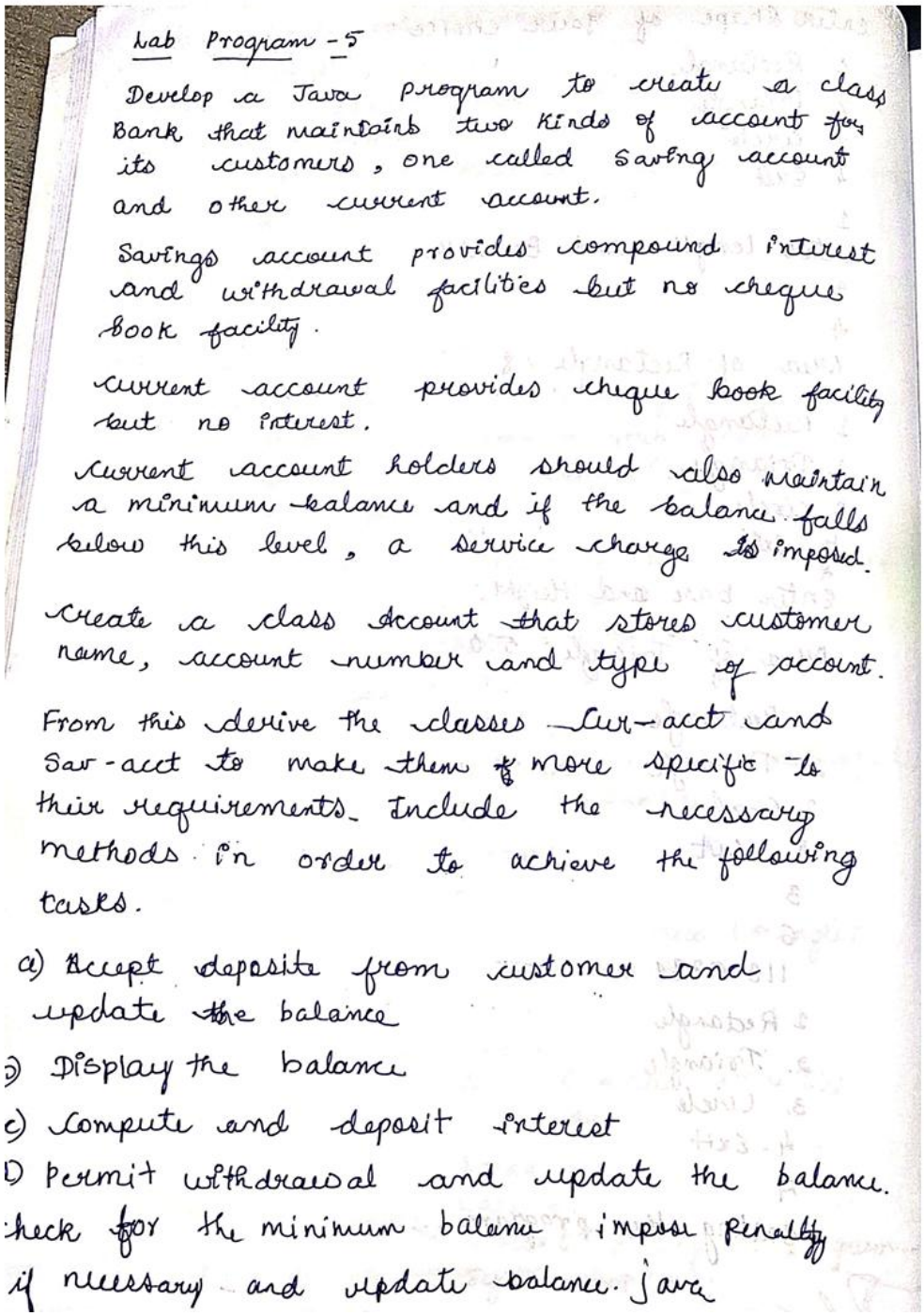
OUTPUT:

```
Choose a shape to calculate area:
1. Rectangle
2. Triangle
3. Circle
1
Enter length of Rectangle: 10
Enter width of Rectangle: 20
```

```
Rectangle Area: 200
Choose a shape to calculate area:
1. Rectangle
2. Triangle
3. Circle
2
Enter base of Triangle: 10
Enter height of Triangle: 15
Triangle Area: 75.0
Choose a shape to calculate area:
1. Rectangle
2. Triangle
3. Circle
3
Enter radius of Circle: 7
Circle Area: 153.93804002589985
Choose a shape to calculate area:
1. Rectangle
2. Triangle
3. Circle
4
Invalid choice. Please choose 1, 2, or 3.
```

LAB program 5

Q} 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.



Lab Program - 5

Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called saving account and other current account.

Savings account provides compound interest and withdrawal facilities but no cheque book facility.

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 deposite 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 balance. java

```
import java.util.Scanner;
class Account {
    String customername;
    Int accountNumber;
    String accountType;
    double balance;

    Public Account (String customername,
                    int accountNumber, String
                    accountType, double balance) {

    this.customername = customername;
    this.accountNumber = accountNumber
    this.accountType = accountType;
    this.. balance = balance;
}

  Public void deposit (double amount) {
    if (amount >0) {
        balance += amount;
        Sout (" Deposit successfull ");
    }
    else {
        sout (" Invalid deposit amount");
    }
}
```

```
    Public void displayBalance() {
        Sout("Account Balance:" + balance);
    }

class SavAcct extends Account {
    private static double interestRate 0.04;

    Public SavAcct (String CustName, acctNum, double
                    balance) {
        Super (customerName, accountNumber, "Savings"
                                    balance);
    }

  Public void depositInterest () {
      double interest = balance * InterestRate
      balance += interest;
      Sout ("Interest of" + interest + has been added)
  }

Public void withdraw (double amount) {
    if (amount > 0 && amount <= balance) {
        balance -= amount;
        System.out.println("Withdrawal
          Successful");
    }
    else {
        Sout ("Invalid withdraval");
    }
```

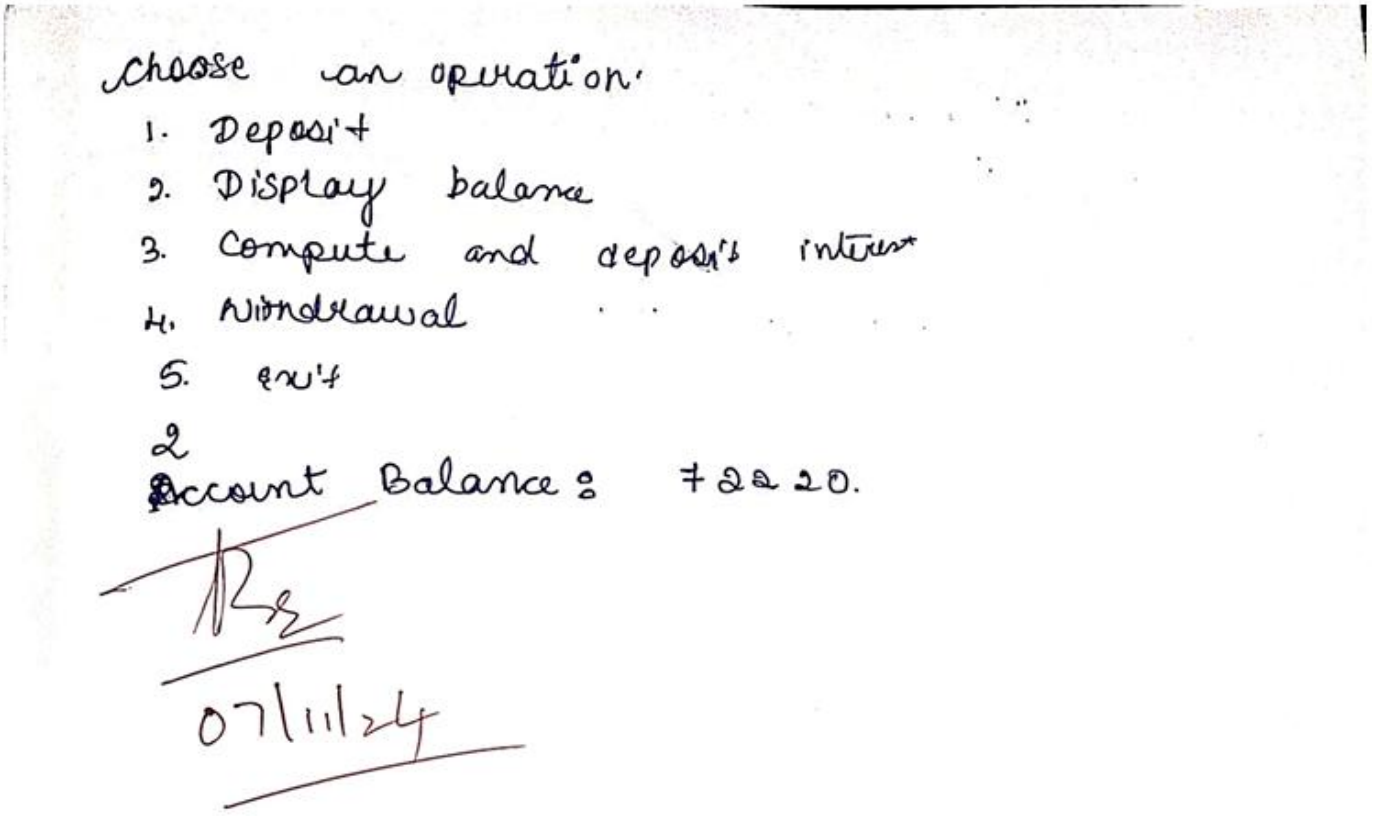
```
class CurAcct extends Account {
    private static double Min_balance = 500.0;
    private static double Service_charge = 50.0;

    public CurAcct (String customerName, int
            accountNumber, double balance) {
        super (customerName, accountNumber,
            "current", balance);
    }

    public void withdraw (double amount) {
        if (amount > 0 && amount <= balance) {
            balance -= amount;
            Sout ("Withdrawal successful");
            checkMinimumBalance();
        } else {
            Sout ("Invalid withdrawal");
        }
    }

    private void checkMinimumBalance () {
        if (balance < MIN_BALANCE) {
            balance -= SERVICE_CHARGE;
            Sout ("Balance fell below min")
        }
    }
}

Public class Bank {
    PSVM () {
    Scanner sc = new Scanner (System.in);
    Account account = null;
    Sout ("Select Bank Type");
    Sout ("1. SB");
    Sout ("2. current Account");
```

```
int accountchoice = scnextInt();
Sout ("Enter your name: ");
String customername = sc.nextLine();
sout ("enter your account number: ");
int accountnumber = sc.nextInt();
sout (" Enter initial deposit ");
double balance = sc.nextDouble();

switch (accountchoice) {
    case 1:
        account = new SavAcct(customername, accountNu
                balance);
        Sout (" Saving account created ");
        break;

    case 2:
        account = new CurAcct (CustOmername, accountNu
                balance);
        Sout (" current Account created ");
        break;
    default:
        Sout (" Invalid choice ");
        return ;
}

While (true) {
    Sout (" Choose an operation: ");
    Sout ("1. Deposit ");
    sout ("2. Display Balance");
    Sout (" 3. Compute and Deposit Interest
                (Savings Account only) ");
    Sout ("4. Withdraw ");
    Sout ("5. Exit ");
```

```
            case 5:
                System.out.println("Thank You");
                sc.close();
                return;
            default:
                System.out.println("Invalid choice");
        }
    }
}
```

output

```
Select Type of account:
1. Savings account
2. current account
1
Enter your name: Shrinanda
Enter your acct number: 123
Enter initial deposit: 2220
Savings Account created:
Choose an operation
1. Deposit
2. Display balance
3. Compute and Deposit Interest
4. Withdrawal
5. Exit
1
Enter deposit amount: 70006
Deposit Successful.
```



CODE:

```
import java.util.Scanner;

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

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

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

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

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

class SavAcct extends Account {
    private static final double INTEREST_RATE = 0.07;

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

    public void computeAndDepositInterest() {
        double interest = balance * INTEREST_RATE;
        balance += interest;
        System.out.println("Interest of " + interest +
        " has been added. New balance: " + balance);
    }

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

class CurAcct extends Account {
    private static final double MIN_BALANCE = 500.0;
    private static final double SERVICE_CHARGE = 50.0;
```

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

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

    private void checkMinimumBalance() {
        if (balance < MIN_BALANCE) {
            balance -= SERVICE_CHARGE;
            System.out.println("Balance fell below minimum. Service charge of "
            + SERVICE_CHARGE + " applied. New balance: " + balance);
        }
    }
}

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

        System.out.print("Enter customer name: ");
        String customerName = scanner.nextLine();

        System.out.print("Enter account number: ");
        int accountNumber = scanner.nextInt();

        System.out.print("Enter initial balance: ");
        double initialBalance = scanner.nextDouble();

        System.out.println("Choose account type:");
        System.out.println("1. Savings Account");
        System.out.println("2. Current Account");
        int accountChoice = scanner.nextInt();

        Account account;
        if (accountChoice == 1) {
            account = new SavAcct(customerName, accountNumber, initialBalance);
```

```
        } else if (accountChoice == 2) {
            account = new CurAcct(customerName, accountNumber, initialBalance);
        } else {
            System.out.println("Invalid account type selection.");
            scanner.close();
            return;
        }

        boolean exit = false;
        while (!exit) {
            System.out.println("\nChoose an operation:");
            System.out.println("1. Deposit");
            System.out.println("2. Withdraw");
            System.out.println("3. Display Balance");
            if (account instanceof SavAcct) {
                System.out.println("4. Compute and Deposit Interest ");
            }
            System.out.println("5. Exit");
            int choice = scanner.nextInt();

            switch (choice) {
                case 1:
                    System.out.print("Enter amount to deposit: ");
                    double depositAmount = scanner.nextDouble();
                    account.deposit(depositAmount);
                    break;

                case 2:
                    System.out.print("Enter amount to withdraw: ");
                    double withdrawAmount = scanner.nextDouble();
                    if (account instanceof SavAcct) {
                        ((SavAcct) account).withdraw(withdrawAmount);
                    } else if (account instanceof CurAcct) {
                        ((CurAcct) account).withdraw(withdrawAmount);
                    }
                    break;

                case 3:
                    account.displayBalance();
                    break;

                case 4:
                    if (account instanceof SavAcct) {
                        ((SavAcct) account).computeAndDepositInterest();
                    } else {
```

```
                        System.out.println("Invalid choice for Current Account.'
                    }
                    break;

                case 5:
                    exit = true;
                    System.out.println("Exiting...");
                    break;

                default:
                    System.out.println("Invalid choice.");
            }
        }

        scanner.close();
    }
}
```

OUTPUT:

```
Enter customer name:xyz
Enter account number: 1001
Enter initial balance: 1000
Choose account type:
1. Savings Account
2. Current Account
1

Choose an operation:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute and Deposit Interest
5. Exit
1
Enter amount to deposit: 200
Deposit successful. New balance: 1200.0

Choose an operation:
1. Deposit
2. Withdraw
3. Display Balance
4. Compute and Deposit Interest
5. Exit
4
```

Program 6

Q] 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

Lab program VI

Create package CIE which has two classes - Student and Internals. The class personal has members USN, name, Sem - Class internals has 5 courses SEE class External derived class of Student. This class has an array that stores the SEE marks scored in five courses of current Semesters of studen

Import the two packages in a file that declares the final marks of n students in all five courses

```
package CIE;
import java.util.Scanner;
public class student {
        String name;
        String USN;
        int sem;
        Public void getd() {
            Scanner sc = new Scanner(System.in);
            Sout("Entie USN");
            USN = sc.nextline();
            Sout("Enater student name");
            name = sc.nextline();
            Sout("Enter Sem")
            Sem = sc.nextInt();
        }
```

```
    Public void display () {
        sout ( );
        sout ("student USN:" + USN + "name" + name
            + "Sem" + sem);
        sout ();
    }
```

```
Package CIE;
import java.util.Scanner;
public class Internals {
    Public int marksCIE [] = new int[5];
    public void getmarks () {
        for (int i=0; i<5; i++) {
            Scanner sc = new Scanner (system.in);
            Sout ("Enter CIE marks in sub" + (i+1));
            marksCIE[i] = sc.nextINt ();
        }
    }
    public int returnmarksCIE (int i) {
        return marksCIE[i];
    }
}
```

```
Package SEE;
Import CIE.Student;
import CIE.Internals;
Import java.util.Scanner;
Public class Externals extends student {
    int marksSEE [] = new int[5];
```

```
    Public void getmarks() {
        for(int i=0; i<6; i++) {
            Scanner sc = new Scanner(System.in);
            sout("Enter SEE marks sub "+(i+1));
            MarksSEE[i] = sc.nextInt();
        }
    }
    Public void calcTotalMarks(Internals I) {
        for(int i=0; i<5; i++) {
            sout("Sub"+(i+1)+" : "+
            (i1.returnMarks(IF[i] + (MarksSEE[i]/2)));
        }
        sout();
    }
}
import CIE.Student;
import CIE.Internals;
import SEE.Externals;
import java.util.Scanner;
Public class main{
    Public srm() {
        Scanner
        sout("Enter number of Students");
        int n = sc.nextInt();
        Internals[] i1 = new Internals[n];
        Externals[] e1 = new Externals[n];
        for(int i=0; i<n; i++) {
            sout("Student" + (i+1) + "details:");
            e1[i] = new Externals();
            I1[i] = new Internals();
```

```
        c1[i].getd();
        i1[i].getMarks();
        c1[i].getmarks();
    }

    for(int i=0; i<n; i++){
        c1[i].display();
        c1[i].calcTotalMarks(i1[i]);
    }
```

//output

Enter no. of students :

2

Student 1 details :

Enter student USN

324

Enter student name

Shrinanda

Enter Semester

3

~~Enter no of subjects:~~

2

Enter CIE marks in Subject 1

34

Enter CIE Marks in Subject2

37 ——— Subject5

~~Enter number of subjects~~

2

Enter SEE Marks in subject 1

78

Enter SEE marks in subject 2

89

Enter SEE marks in subject5

91 ——— subject5

Student 2 details

21/11/24

CODE

```
package CIE;

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

package CIE;

public class Internals {
    public int[] internalMarks = new int[5];
}

package SEE;

import CIE.Student;

public class External extends Student {
    public int[] seeMarks = new int[5];
}

import CIE.*;
import SEE.*;
import java.util.Scanner;

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

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

        Student[] students = new Student[n];
        Internals[] internals = new Internals[n];
        External[] externals = new External[n];

        for (int i = 0; i < n; i++) {
            students[i] = new Student();
            internals[i] = new Internals();
            externals[i] = new External();

            System.out.println("Enter details for student " + (i + 1) + ":");
```

```
            System.out.print("USN: ");
            students[i].usn = sc.next();
            System.out.print("Name: ");
            students[i].name = sc.next();
            System.out.print("Semester: ");
            students[i].sem = sc.nextInt();

            System.out.println("Enter internal marks for 5 courses: ");
            for (int j = 0; j < 5; j++) {
                internals[i].internalMarks[j] = sc.nextInt();
            }

            System.out.println("Enter SEE marks for 5 courses: ");
            for (int j = 0; j < 5; j++) {
                externals[i].seeMarks[j] = sc.nextInt();
            }
        }

        System.out.println("Final Marks:");
        for (int i = 0; i < n; i++) {
            System.out.println("Student " + (i + 1) + " ("
            + students[i].usn + ", " + students[i].name + "):");
            for (int j = 0; j < 5; j++) {
                int finalMark = (internals[i].internalMarks[j] +
                 externals[i].seeMarks[j]) / 2;
                System.out.println("Course " + (j + 1) + ": " + finalMark);
            }
        }
    }
}
```

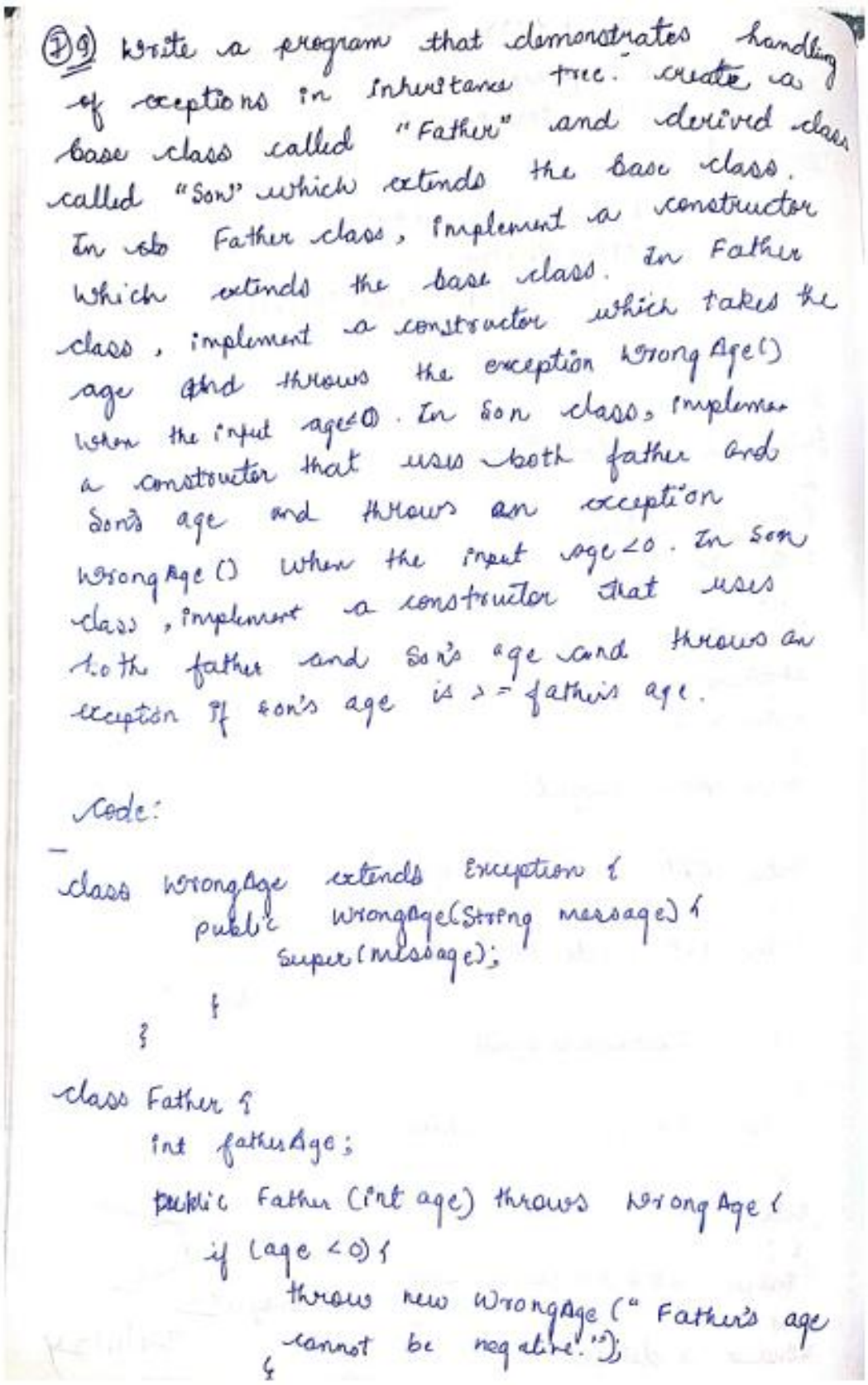
OUTPUT:

```
Enter the number of students: 2
Enter details for student 1:
USN: 1BM21CS001
Name: Alice
Semester: 5
Enter internal marks for 5 courses:
20 18 22 25 24
Enter SEE marks for 5 courses:
60 55 70 80 75
Enter details for student 2:
USN: 1BM21CS002
Name: Bob
```

```
Semester: 5
Enter internal marks for 5 courses:
19 21 20 23 22
Enter SEE marks for 5 courses:
65 60 75 85 80
Final Marks:
Student 1 (1BM21CS001, Alice):
Course 1: 40
Course 2: 36
Course 3: 46
Course 4: 52
Course 5: 49
Student 2 (1BM21CS002, Bob):
Course 1: 42
Course 2: 40
Course 3: 47
Course 4: 54
Course 5: 51
```

Program 7

Q] 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=father's age.



(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 WrongAge() when the input age<0. In Son class, implement a constructor that uses both the father and Son's age and throws an exception if son's age is >= father's age.

Code:

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

class Father {
    int fathersAge;
    public Father(int age) throws WrongAge {
        if (age < 0) {
            throws new WrongAge("Father's age
            cannot be negative.");
        }
```

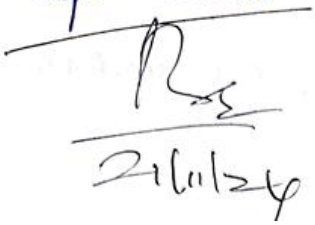
```
        this.fatherAge = age;
    }
}
class Son extends Father {
    int sonAge;
    public Son(int fatherAge, int sonAge)
    throws WrongAge {
        super(fatherAge);
        if (sonAge >= fatherAge) {
            throw new WrongAge("Son's age can't be
greater or equals father Age");
        }
        this.sonAge = sonAge;
    }
}
public class Main {
    PSVM() {
    try {
        Son son3 = new Son(-1, 10);
        Son son1 = new Son(50, 25);
        Sout(" Father's Age : " + son1.fatherAge
        + " Son's Age : " + son1.sonAge);
    }
    catch (WrongAge e) {
        Sout(" Exception caught: " + e.getMessage());
    }
}
```

~~Output~~

// Output

Son Age can't be negative

Son Age can't be greater than father's age



CODE:

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

class Father {
    int age;

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

```
class Son extends Father {
    int sonAge;

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

public class ExceptionInheritanceDemo {
    public static void main(String[] args) {
        try {
            Father father = new Father(45);
            Son son = new Son(45, 20);
            System.out.println("Father's age: " + father.age);
            System.out.println("Son's age: " + son.sonAge);
        } catch (WrongAgeException e) {
            System.out.println("Exception: " + e.getMessage());
        }

        try {
            Son invalidSon = new Son(30, 35);
        } catch (WrongAgeException e) {
            System.out.println("Exception: " + e.getMessage());
        }

        try {
            Father invalidFather = new Father(-5);
        } catch (WrongAgeException e) {
            System.out.println("Exception: " + e.getMessage());
        }
    }
}
```

OUTPUT:

```
Father's age: 45
Son's age: 20
Exception: Son's age cannot be greater than or equal to Father's age
Exception: Father's age cannot be negative
```

Lab Program 8

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

```
//code

class DisplayBMS extends Thread {
    Public void run (){
        try {
            While (true) {
                Sout ("BMS College of engineering");
                Thread. sleep (10000);
            }
        }
        catch (InterruptedException e) {
            Sout (" Thread interrupted:" + e.getMessage());
        }
    }
}
class DisplayCSE extends Thread {
    public void run () {
        try {
            while (true) {
                Sout ("CSE");
                Thread. sleep (2000);
            }
        }
```

```
        catch (InterruptedException e) {
            sout ("Thread Interrupted: " + e.getMess
        }
    }
}

Public class MultiThreadExample {
    PSVM() {
        DisplayBMS thread1 = new DisplayBMS();
        DisplayCSE thread2 = new DisplayCSE();
        thread1.start();
        thread2.start();
    }
}

//output
BMS college of engineering
CSE
CSE
CSE
CSE
CSE
BMS college of engineering
CSE
CSE
CSE
CSE
BMS college of engineering
)
{
}
```

11/12/24

```
class CollegeThread extends Thread {
    private String message;
    private int interval;

    public CollegeThread(String message, int interval) {
        this.message = message;
        this.interval = interval;
    }

    public void run() {
        try {
            while (true) {
                System.out.println(message);
                Thread.sleep(interval * 1000);
            }
        } catch (InterruptedException e) {
            Thread.currentThread().interrupt();
        }
    }
}

public class Main {
    public static void main(String[] args) {
        Thread thread1 = new CollegeThread("BMS College of Engineering", 10);
        Thread thread2 = new CollegeThread("CSE", 2);
        thread1.start();
        thread2.start();
    }
}
```

//output

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

Lab Program 9

Q] 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.

8) Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 & Num2. The division of Num1 & 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 message dialog box.

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

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 (" ", Label.Right);
add (number1);
add (num1);
add (number2);
add (num2);
add (dResult);
add (outResult);
num1.addActionListener (this);
num2.addActionListener (this);
dResult.addActionListener (this);
addWindowListener (new WindowAdapter () {
    Public void windowClosing (WindowEvent e) {
        System.exit(0);
    }
});
}

Public void actionPerformed (ActionEvent e) {
    Int n1, n2;
    try {
        if (e.getSource() == dResult) {
        n1 = Integer.parseInt (num1.getText());
        n2 = Integer.parseInt (num2.getText());
        if (n2 == 0) {
            throw new ArithmeticException();
        }
        out = n1 + "/" + n2 + " = ";
        resultNum = (double) n1 / n2;
        out += resultNum;
    }
}
```

```
        catch (NumberFormatException e1) {
            flag = 1;
            out = "Number Format exception! " +
                  e1.getMessage();
        }
        catch (ArithmeticException e1) {
            flag = 1;
            out = "Divide by 0 exception! " + e1.getMessage();
        }
        outResult.setText(out);
        invalidate();
        validate();
    }
}
public class Main {
    p.s.v.m(C ) {
        DivisionMain1 obj = new DivisionMain1();
        obj.setSize(new Dimension(800, 400));
        obj.setTitle("Division of Integers")
        obj.setVisible(true);
    }
}
```

Sample outputs:

Number 1 : 20 Number 2 : 4 (Button Pressed) Result : 20/4 5.0

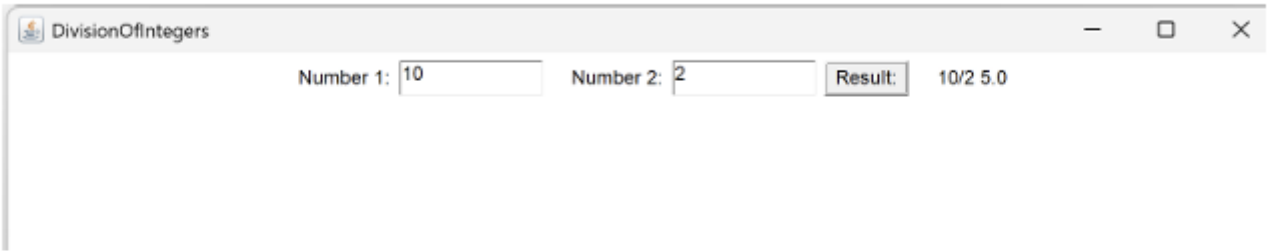
Number 1 : 20 Number 2 : 0 Result : Divide by 0 exception! / by zero.

Number 1 : abc Number 2 : 5 Result : "abc" Number Format exception! For input string: "abc"

```
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("", Label.RIGHT);
        add(number1);
        add(num1);
        add(number2);
        add(num2);
        add(dResult);
        add(outResult);
        num1.addActionListener(this);
        num2.addActionListener(this);
        dResult.addActionListener(this);
        addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent e) {
                System.exit(0);
            }
        });
    }

    public void actionPerformed(ActionEvent e) {
        int n1, n2;
        try {
            if (e.getSource() == dResult) {
                n1 = Integer.parseInt(num1.getText());
                n2 = Integer.parseInt(num2.getText());
                if (n2 == 0) {
                    throw new ArithmeticException();
                }
                out = n1 + "/" + n2 + " ";
                resultNum = (double) n1 / n2;
                out += resultNum;
```

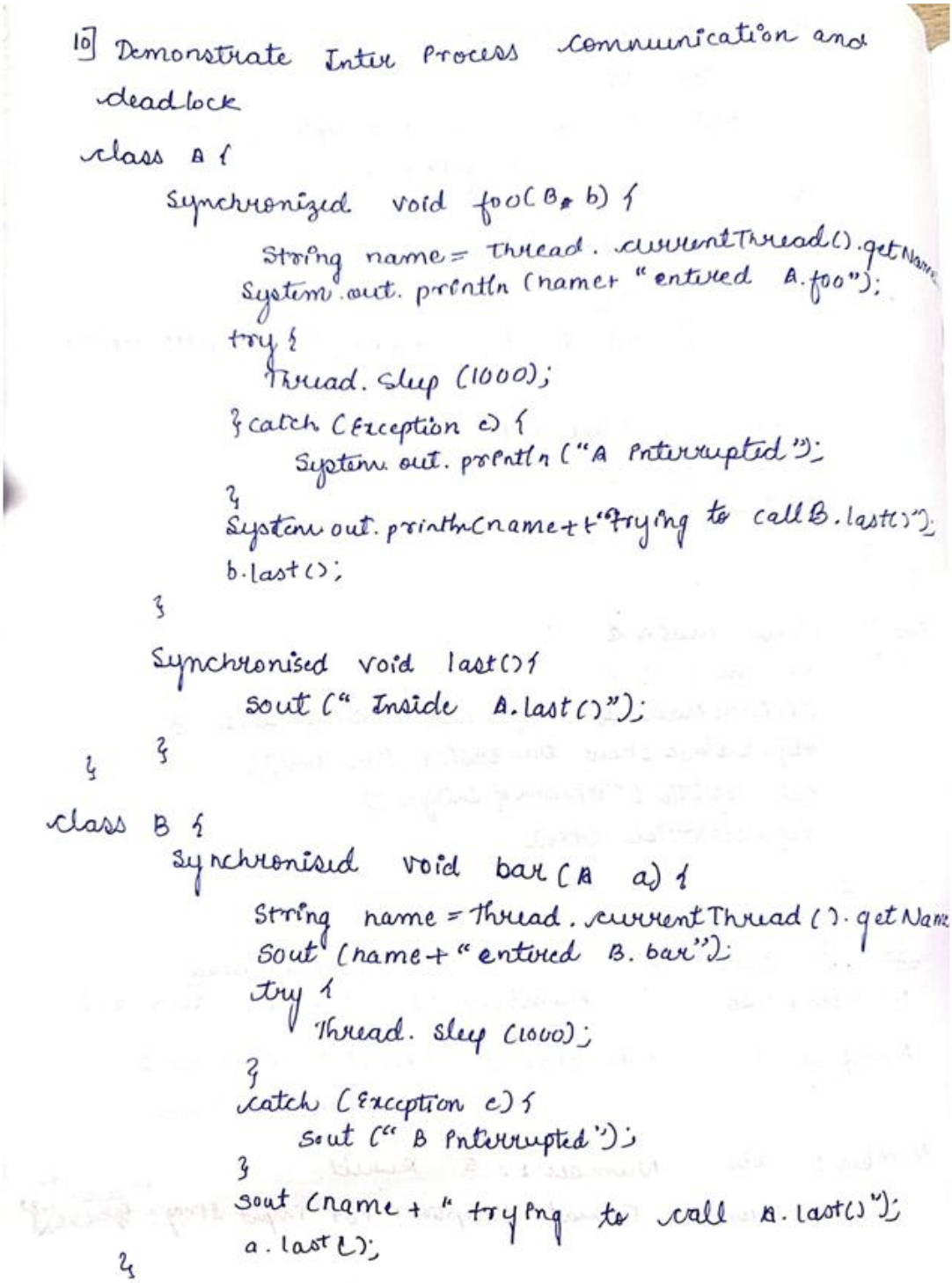
```
            }
        } catch (NumberFormatException e1) {
            flag = 1;
            out = "Number Format Exception! " + e1.getMessage();
        } catch (ArithmeticException e1) {
            flag = 1;
            out = "Divide by 0 Exception! " + e1.getMessage();
        }
        outResult.setText(out);
        invalidate();
        validate();
    }
}

public class Main {
    public static void main(String args[]) {
        DivisionMain1 obj = new DivisionMain1();
        obj.setSize(new Dimension(800, 400));
        obj.setTitle("DivisionOfIntegers");
        obj.setVisible(true);
    }
}
```

//output



Lab Program 10

Q] Demonstrate Inter process Communication and deadlock



```
10] Demonstrate Inter Process communication and
deadlock
class A {
    Synchronized void foo(B b) {
        String name = Thread.currentThread().getName
        System.out.println(name+ "entered A.foo");
        try {
            Thread.sleep(1000);
        } catch (Exception e) {
            System.out.println("A Interrupted");
        }
        System.out.println(name+"trying to call B.last()");
        b.last();
    }
    Synchronised void last() {
        sout(" Inside A.last()");
    }
}
class B {
    Synchronised void bar(A a) {
        String name = Thread.currentThread().getName
        sout(name+" entered B.bar");
        try {
            Thread.sleep(1000);
        }
        catch (Exception e) {
            sout(" B Interrupted");
        }
        sout(name + " trying to call A.last()");
        a.last();
    }
```

```
    Synchronised  void  last() {
        Sout("Inside  B.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);
        Sout("Back in main");
    }

    public void run() {
        b.bar(a);
        Sout("Back in other thread");
    }

    P S VM (String[]  args) {
        new Deadlock();
    }
}
```

output

```
MainThread  entered  A.foo
MainThread  trying  to call  B.last()
RacingThread  entered  B.bar
RacingThread  trying  to call  B.last()
```

```
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();
    }

    synchronized 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();
    }

    synchronized void last() {
        System.out.println("Inside B.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); // Main thread locks A and tries to call B.last()
        System.out.println("Back in main");
    }

    public void run() {
```

```
        b.bar(a); // Racing thread locks B and tries to call A.last()
        System.out.println("Back in other thread");
    }

    public static void main(String[] args) {
        new Deadlock();
    }
}
```

//OUTPUT

```
MainThread entered A.foo
MainThread trying to call B.last()
RacingThread entered B.bar
RacingThread trying to call A.last()
```