

**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**  
“JnanaSangama”, Belgaum -590014, Karnataka.



**LAB REPORT**  
**on**  
**Object Oriented Java Programming**  
**(23CS3PCOOJ)**

*Submitted by*

Annas Sharieff (**1BM23CS041**)

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



## B.M.S. COLLEGE OF ENGINEERING

(Autonomous Institution under VTU)

**BENGALURU-560019**

**Sep-2024 to Jan-2025**

### **B.M.S. College of Engineering,**

**Bull Temple Road, Bangalore 560019**

(Affiliated To Visvesvaraya Technological University, Belgaum)

### **Department of Computer Science and Engineering**



### **CERTIFICATE**

This is to certify that the Lab work entitled “Object Oriented Java Programming (23CS3PCOOJ)” carried out by **Annas Sharieff (1BM23CS041)**, who is bonafide student of **B.M.S. College of Engineering**. It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum. The Lab report has been approved as it satisfies the academic requirements in respect of an Object Oriented Java Programming (23CS3PCOOJ) work prescribed for the said degree.

Dr. Sheeta VA Assistant Professor Department of CSE, BMSCE	Dr. Jyothi S Nayak Professor & HOD Department of CSE, BMSCE
--	---

## **Index**

<b>Sl. No.</b>	<b>Date</b>	<b>Experiment Title</b>	<b>Page No.</b>
1	9/10	QUADRATIC EQUATION	4-8
2	16/10	SGPA CALCULATOR	9-16
3	23/10	BOOK PROGRAM	17-22
4	23/10	ABSTRACT CLASS SHAPE PROGRAM	23-28
5	13/11	BANK PROGRAM	29-37
6	13/11	PACKAGES	38-45
7	20/11	EXCEPTION HANDLING	46-52
8	27/11	MULTI THREADING	53-55
9	27/11	INTEGER DIVISION WITH USER INTERFACE	56-61
10	27/11	INTER PROCESS COMMUNICATION AND DEADLOCK	62-80

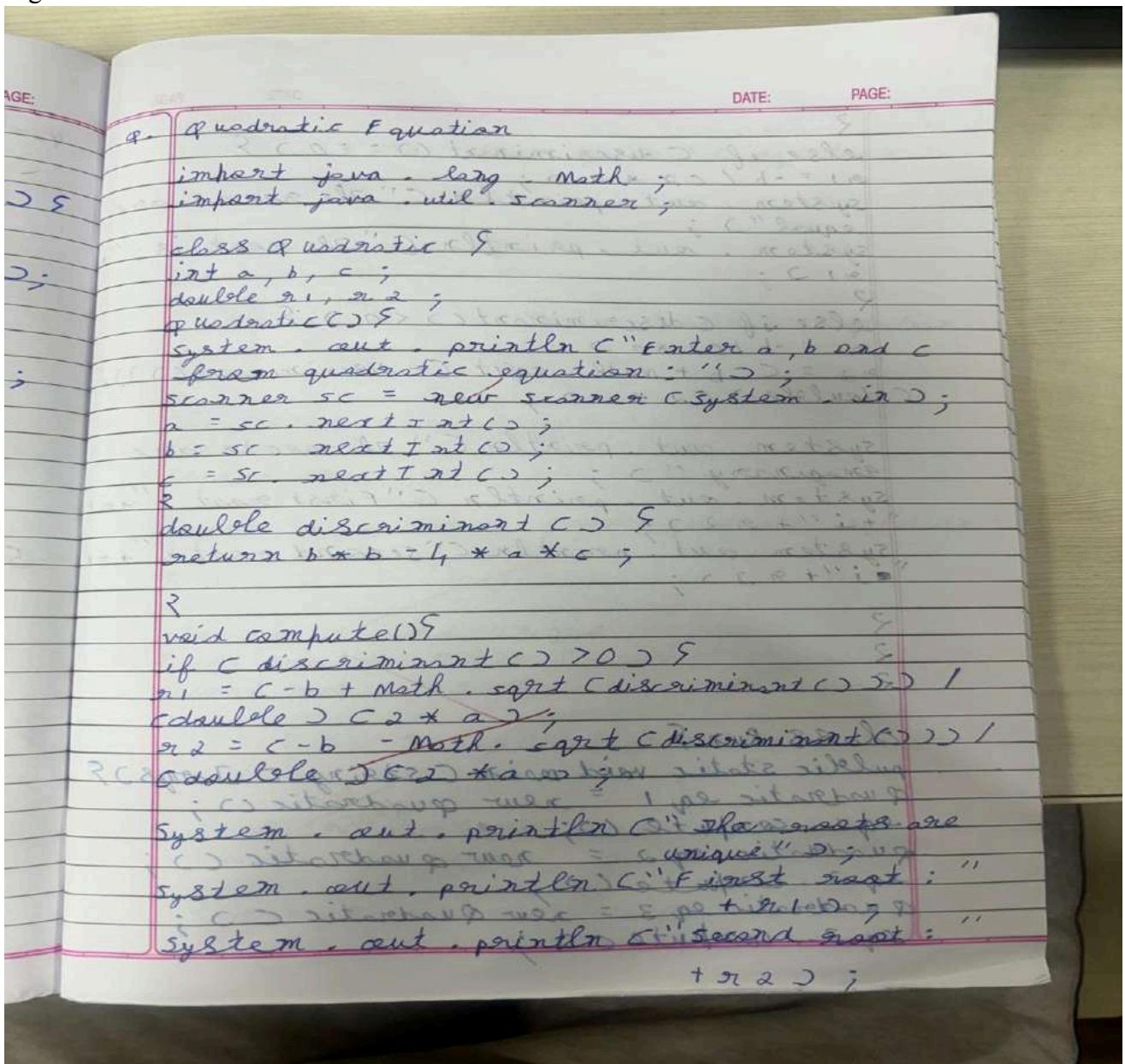
Github Link:

<https://github.com/annas05shariff/java-lab>

## Program 1

Implement Quadratic Equation

Algorithm:



```

    } else if (discriminant() == 0) {
        r1 = -b / (c2 * a);
        system.out.println("The roots are equal");
        system.out.println("The root is:");
        r1;
    } else if (discriminant() < 0) {
        r1 = -b / (c2 * a);
        r2 = (-b + Math.sqrt(-discriminant())) /
            (double)(c2 * a);
        system.out.println("The roots are imaginary");
        system.out.println("First root: " + r1);
        system.out.println("Second root: " + r1);
        - i + r2);
    }
}

```

```

class Run {
    public static void main(String[] args) {
        quadratic eq1 = new quadratic();
        eq1.compute();
        quadratic eq2 = new quadratic();
        eq2.compute();
        quadratic eq3 = new quadratic();
        eq3.compute();
    }
}

```

Output :

Enter a, b and c from quadratic equation :

1

2

3

4

5

6

7

8

9

0

.

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

,

Code:

```
import java.lang.Math;
import java.util.Scanner;

class Quadratic{
    int a, b, c;
    double r1, r2;
    Quadratic(){
        System.out.println("Enter a, b and c from quadratic equation: ");
        Scanner sc = new Scanner(System.in);
        a = sc.nextInt();
        b = sc.nextInt();
        c = sc.nextInt();
    }
    double discriminant(){
        return b*b-4*a*c;
    }
    void compute(){
        if(discriminant() > 0){
            r1 = (-b + Math.sqrt(discriminant()))/(double)(2*a);
            r2 = (-b - Math.sqrt(discriminant()))/(double)(2*a);
            System.out.println("The roots are unique");
            System.out.println("First root: " + r1);
            System.out.println("Second root: " + r2);
        }
        else if(discriminant() == 0){
            r1 = -b/(2*a);
            System.out.println("The roots are equal");
            System.out.println("The root is: " + r1);
        }
        else if(discriminant() < 0){
            r1 = -b/(2*a);
            r2 = (-b + Math.sqrt(-discriminant()))/(double)(2*a);
            System.out.println("The roots are Imaginary");
            System.out.println("First root: " + r1 + "+i" + r2);
            System.out.println("Second root: " + r1 + "-i" + r2);
        }
    }
}

class Run{
    public static void main(String[] args){
        Quadratic eq1 = new Quadratic();
        eq1.compute();
        Quadratic eq2 = new Quadratic();
        eq2.compute();
        Quadratic eq3 = new Quadratic();
        eq3.compute();
    }
}
```

Command Prompt

Microsoft Windows [Version 10.0.22000.2538]  
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin>cd desktop

C:\Users\Admin\Desktop>javac Run.java

C:\Users\Admin\Desktop>java Run

Enter a, b and c from quadratic equation:

1

2

1

The roots are equal

The root is: -1.0

Enter a, b and c from quadratic equation:

3

4

9

The roots are Imaginary

First root: 0.0+i0.9319438411042397

Second root: 0.0-i0.9319438411042397

Enter a, b and c from quadratic equation:

1

5

6

The roots are unique

First root: -2.0

Second root: -3.0

C:\Users\Admin\Desktop>ANNAS SHARIEFF 1BM23CS041

## Program 2- SGPA Calculator :

Algorithm:

The image shows a handwritten algorithm for a SGPA Calculator in a notebook. The algorithm is written in Java-like pseudocode. It starts with importing the Scanner class, defining a Subject class with subjectMarks, credits, and grade attributes, and a calculateGrade method. The calculateGrade method uses nested if statements to determine the grade based on subjectMarks. The conditions for grades 10, 9, 8, 7, 6, and 5 are defined with their respective marks. There is a red mark with 'NGO' and '50' crossed out above the condition for grade 6.

```
q. student details, SGPA .
import java.util.Scanner ;
class Subject {
    int subjectMarks ;
    int credits ;
    int grade ;
    public void calculateGrade() {
        if (subjectMarks >= 90 && subjectMarks <= 100) {
            grade = 10 ;
        } else if (subjectMarks >= 80) {
            grade = 9 ;
        } else if (subjectMarks >= 70) {
            grade = 8 ;
        } else if (subjectMarks >= 60) {
            grade = 7 ;
        } else if (subjectMarks >= 50) {
            grade = 6 ;
        } else if (subjectMarks >= 40) {
            grade = 5 ;
        }
    }
}
```

```
else S  
grade = 0 ;
```

```
R  
R  
R
```

```
class student S  
string name;  
string usn;  
double SGPA;  
subject [ ] subject = new subject [8];  
Scanner s = new Scanner (System.in);
```

```
<=100)  
public student () S  
for (int i = 0; i < 8; i++) S  
subject [i] = new subject ();
```

```
R  
R
```

```
public void getstudentDetails () S  
System.out.println ("Enter Student Name:  
");  
name = s.nextLine();  
System.out.println ("Enter student USN:");  
usn = s.nextLine();
```

```
public void getmarks () S  
for (int i = 0; i < 8; i++) S  
System.out.println ("Enter Marks for  
subject " + (i + 1) + ": ");  
subject [i].subjectmarks = s.nextInt();
```

DATE: \_\_\_\_\_ PAGE: \_\_\_\_\_

```

if (subject[i].subjectMarks > 100) {
    subject[i].subjectMarks = 100;
}
else if (subject[i].subjectMarks < 0) {
    system.out.println("Invalid marks!");
    i--;
    continue;
}

R
system.out.println("Enter credits for"
    + subject[i+1] + ":" );
subject[i].credits = s.nextInt();
subject[i].calculateGrade();
R
R
public void computeSGPA() {
    int totalCredits = 0;
    int effectiveScore = 0;
    for (int i = 0; i < 8; i++) {
        effectiveScore += (subject[i].grade *
            subject[i].credits);
        totalCredits += subject[i].credits;
    }
    SGPA = (float) effectiveScore / totalCredits;
}

R
public void displayResult() {
    system.out.println("Student Name: " + name);
    system.out.println("Student USN: " + usn);
    system.out.println("SGPA: " + SGPA);
    system.out.println("Percentage: " +
        ((float) effectiveScore / totalCredits) * 100);
}

```

```
public class Main {  
    public static void main (String [ ] args ) {  
        student . getstudentdetails ( ) ;  
        student . getmarks ( ) ;  
        student . computeSGPA () ;  
        student . displayresult ( ) ;  
    }  
}
```

R

### Output

Enter details for student :

Enter student Name :

John Doe

Enter student USN :

123

Enter marks for subject 1 :

90

Enter credits for subject 1 :

3

Enter marks for subject 2 :

70

Enter ~~marks~~ credits for subject 2 :

3

Enter marks for subject 3 :

name ); 80

); Enter credits for subject 3 :

4

Enter Marks for subject 4 :

90

Enter credits for subject 4 :

4

DATE:

PAGE:

Enter Marks for subject 5 :

60 Enter credits

for subject 5 :

4

Enter marks for subject 6 :

75

Enter credits for subject 6 .

2

Enter marks for subject 7 :

90

Enter credits for subject 7 :

1

Enter Marks for subject 8 .

90

Enter credits for subject 8 .

1

Student Name : John Doe

Student USN : 123

SGPA : 8.812

Set  
16/02/21

Code:

```
import java.util.Scanner;

class Subject {
    int subjectMarks;
    int credits;
    int grade;

    public void calculateGrade() {
        if (subjectMarks >= 90 && subjectMarks <= 100) {
            grade = 10;
        } else if (subjectMarks >= 80) {
            grade = 9;
        } else if (subjectMarks >= 70) {
            grade = 8;
        } else if (subjectMarks >= 60) {
            grade = 7;
        } else if (subjectMarks >= 50) {
            grade = 6;
        } else if (subjectMarks >= 40) {
            grade = 5;
        } else {
            grade = 0;
        }
    }
}

class Student {
    String name;
    String usn;
    double SGPA;
    Subject[] subject = new Subject[8];
    Scanner s = new Scanner(System.in);

    public Student() {
        for (int i = 0; i < 8; i++) {
            subject[i] = new Subject();
        }
    }

    public void getStudentDetails() {
        System.out.println("Enter Student Name: ");
        name = s.nextLine();
        System.out.println("Enter Student USN: ");
        usn = s.nextLine();
    }
}
```

```

public void getMarks() {
    for (int i = 0; i < 8; i++) {
        System.out.println("Enter Marks for Subject " + (i + 1) + ": ");
        subject[i].subjectMarks = s.nextInt();

        if (subject[i].subjectMarks > 100 || subject[i].subjectMarks < 0) {
            System.out.println("Invalid marks! Please enter again.");
            i--;
            continue;
        }

        System.out.println("Enter Credits for Subject " + (i + 1) + ": ");
        subject[i].credits = s.nextInt();

        subject[i].calculateGrade();
    }
}

public void computeSGPA() {
    int totalCredits = 0;
    int effectiveScore = 0;

    for (int i = 0; i < 8; i++) {
        effectiveScore += (subject[i].grade * subject[i].credits);
        totalCredits += subject[i].credits;
    }

    SGPA = (double) effectiveScore / totalCredits;
}

public void displayResult() {
    System.out.println("\nStudent Name: " + name);
    System.out.println("Student USN: " + usn);
    System.out.println("SGPA: " + SGPA);
}

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

        Student[] students = new Student[3];

        for (int i = 0; i < 3; i++) {
            System.out.println("\nEnter details for Student " + (i + 1) + ": ");
            students[i] = new Student();
        }
    }
}

```

```
        students[i].getStudentDetails();
        students[i].getMarks();
        students[i].computeSGPA();
    }

    System.out.println("\n\nResults for all students:");
    for (int i = 0; i < 3; i++) {
        students[i].displayResult();
    }
}
```

Enter details for Student 1:  
Enter Student Name: John Doe  
Enter Student USN: 1AB20CS101  
Enter Marks for Subject 1: 85  
Enter Credits for Subject 1: 4  
...  
Enter Marks for Subject 8: 70  
Enter Credits for Subject 8: 3

Enter details for Student 2:  
Enter Student Name: Jane Smith  
Enter Student USN: 1AB20CS102  
Enter Marks for Subject 1: 90  
Enter Credits for Subject 1: 4  
...

Enter details for Student 3:  
Enter Student Name: Alex Johnson  
Enter Student USN: 1AB20CS103  
Enter Marks for Subject 1: 78  
Enter Credits for Subject 1: 4  
...

Results for all students:

Student Name: John Doe  
Student USN: 1AB20CS101  
SGPA: 8.25

Student Name: Jane Smith  
Student USN: 1AB20CS102  
SGPA: 9.15

Student Name: Alex Johnson  
Student USN: 1AB20CS103  
SGPA: 7.75

### Program 3- Book Program:

Algorithm:

Book

DATE: PAGE:

```
import java.util.Scanner;  
class Book {  
    private String name;  
    private String author;  
    private double price;  
    private int numPages;  
    public Book (String name, String author,  
                double price, int numPages) {  
        this.name = name;  
        this.author = author;  
        this.price = price;  
        this.numPages = numPages;  
    }  
    public String getName () {  
        return name;  
    }  
    public String getAuthor () {  
        return author;  
    }  
    public double getPrice () {  
        return price;  
    }  
    public int getNumPages () {  
        return numPages;  
    }  
    public void setAuthor (String author) {  
        this.author = author;  
    }  
}
```

Java

DATE: PAGE:

```

public void setName (String name) {
    this.name = name;
}

public void setPrice (double price) {
    this.price = price;
}

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

public String toString () {
    return "Book Name: " + name + " by Author "
        + author + " Price: " + price + " Number "
        + " of Pages: " + numPages;
}

public class BookDemo {
    public static void main (String [] args) {
        Scanner scanner = new Scanner (System.in);
        System.out.print ("Enter number of books: ");
        int n = scanner.nextInt ();
        scanner.nextLine ();
        Book [] books = new Book [n];
        for (int i = 0; i < n; i++) {
            books[i] = new Book ();
            books[i].setName (scanner.nextLine ());
            books[i].setPrice (scanner.nextDouble ());
            books[i].setNumPages (scanner.nextInt ());
        }
    }
}

```

3048 3140  
DATE: PAGE:  
system.out.println("Enter details for  
book "+(i+1)+":");  
system.out.print("Name:");  
String name =  
scanner.nextLine();  
system.out.print("Author:");  
String author =  
scanner.nextLine();  
system.out.print("Price:");  
double price =  
scanner.nextDouble();  
system.out.print("Number of Pages:");  
int numPages =  
scanner.nextInt();  
scanner.nextLine();  
books[i] = new Book(name, author, price,  
numPages);

{  
for (int i = 0; i < n; i++) {  
system.out.println("In details of  
book "+(i+1)+"");  
};

system.out.println(books[i].  
toString());  
}

scanner.close();

2  
2

DATE:

PAGE

output

Enter number of books : 2

Enter details for book 1 :

Name : wimpy kid

Author : romanajan

Price : 200

Number of Pages : 250

Enter details for book 2 :

Name : rich dad poor dad

Author : ching cheng

Price : 500

Number of Pages : 300

Details of Book 1 :

Book Name : wimpy kid

Author : romanajan

Price : 200.0

Number of Pages : 250

Details of Book 2 :

Book Name : rich dad poor dad

Author : ching cheng

Price : 500.0

Number of Pages : 300

Code:

```
import java.util.Scanner;

class Book {

    String name;
    String author;
    int price;
    int numPages;

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

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

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

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

        Book[] books = new Book[n];

        for (int i = 0; i < n; i++) {
            System.out.print("Enter name of book " + (i + 1) + ": ");
            String name = s.next();
            System.out.print("Enter author of book " + (i + 1) + ": ");
            String author = s.next();
            System.out.print("Enter price of book " + (i + 1) + ": ");
            int price = s.nextInt();
            System.out.print("Enter number of pages in book " + (i + 1) + ": ");
        }
    }
}
```

```

        int numPages = s.nextInt();

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

    System.out.println("\nBook Details:");
    for (Book book : books) {
        System.out.println(book);
    }

    s.close();
}
}

```

```

Microsoft Windows [Version 10.0.22000.2538]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin>cd desktop

C:\Users\Admin\Desktop>javac BookDemo.java

C:\Users\Admin\Desktop>java BookDemo
Enter number of books: 2
Enter details for book 1:
Name: wimpy kid
Author: ramanujan
Price: 200
Number of Pages: 250
Enter details for book 2:
Name: rich dad poor dad
Author: ching chong
Price: 500
Number of Pages: 300

Details of Book 1:
Book Name: wimpy kid
Author: ramanujan
Price: 200.0
Number of Pages: 250

Details of Book 2:
Book Name: rich dad poor dad
Author: ching chong
Price: 500.0
Number of Pages: 300

C:\Users\Admin\Desktop>ANNAS SHARIEFF 041

```

#### Program 4- Abstract Class:

Algorithm:

Abstract Class

DATE: PAGE:

```
abstract class shape {  
    int l, b;  
    abstract void printarea();  
}  
class rectangle extends shape {  
    rectangle (int length, int breadth)  
    {  
        this.l = length;  
        this.b = breadth;  
    }  
  
    void printarea()  
    {  
        int area = l * b;  
        System.out.println ("Area of rectangle  
        = " + area);  
    }  
  
class triangle extends shape {  
    triangle (int height, int width)  
    {  
        this.l = height;  
        this.b = width;  
    }  
  
    void printarea()  
    {  
        double area = 0.5 * l * b;  
        System.out.println ("Area of the  
        triangle = " + area);  
    }  
}
```

class circle extends shape {  
 circle (int radius )  
 {  
 this. l = radius ;  
 }

void printarea ()

{  
 double area = 3.14 \* l \* l;  
 System.out.println ("Area of the  
 circle = " + area );  
 }

public class abstract class

{  
 public static void main (String []  
 args )

shape rectangle = new rectangle (5,10);

shape triangle = new triangle (6,8);

shape circle = new circle (7);

rectangle . printarea ();

triangle . printarea ();

circle . printarea ();

() displaying line

i d \* l \* 2.0 : new elephant

for area ) string. two . metryz

i ( area ) = elephant

out  
 Area  
 Are  
 Are

output

DATE:

PAGE:

Area of the rectangle = 77

Area of triangle = 49.5

Area of the circle = 28.260

~~23/10/2021~~

Code:

```
import java.util.Scanner;

abstract class Shape {
    int dim1;
    int dim2;

    public Shape() {
        this.dim1 = 0;
        this.dim2 = 0;
    }

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

    public abstract void printArea();
}

class Rectangle extends Shape {
    public Rectangle(int length, int width) {
        dim1 = length;
        dim2 = width;
    }

    public void printArea() {

        int area = dim1 * dim2;
        System.out.println("Area of Rectangle: " + area);

    }
}

class Triangle extends Shape {
    public Triangle(int base, int height) {
        dim1 = base;
        dim2 = height;
    }

    public void printArea() {

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

    }
}
```

```

class Circle extends Shape {
    public Circle(int radius) {

        dim1 = radius;
        dim2 = 0;
    }

    public void printArea() {

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

public class shapes {

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

        System.out.println("Enter length and width for Rectangle:");

        int length = in.nextInt();
        int width = in.nextInt();
        Shape rectangle = new Rectangle(length, width);
        rectangle.printArea();

        System.out.println("Enter base and height for Triangle:");

        int base = in.nextInt();
        int height = in.nextInt();
        Shape triangle = new Triangle(base, height);
        triangle.printArea();

        System.out.println("Enter radius for Circle:");

        int radius = in.nextInt();
        Shape circle = new Circle(radius);
        circle.printArea();

        in.close();
    }
}

```

Command Prompt

Microsoft Windows [Version 10.0.22000.2538]  
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin>cd desktop

C:\Users\Admin\Desktop>javac abstract\_class.java

C:\Users\Admin\Desktop>java abstract\_class  
Area of the rectangle =77  
Area of the triangle =49.5  
Area of the circle =28.259999999999998

C:\Users\Admin\Desktop>ANNAS SHARIEFF 041

## Program 5- Bank Program:

Algorithm:

The image shows handwritten Java code for a bank account class. The code is written in black ink on lined paper. It includes imports, class definition, constructor, and methods for deposit and balance retrieval. The code is annotated with several question marks (?) and a checkmark (✓) at the top.

```
✓  
import java.util.Scanner;  
  
class Account {  
    private String custname;  
    private String accno;  
    private double balance;  
  
    public Account (String custname, String accno,  
        double balance) {  
        this.custname = custname;  
        this.accno = accno;  
        this.balance = balance;  
    }  
  
    public double getBalance() {  
        return this.balance;  
    }  
  
    public void deposit (double amount) {  
        if (amount > 0) {  
            this.balance += amount;  
            System.out.println ("The current  
balance is " + this.balance);  
        } else {  
            System.out.println ("Amount should  
not be negative");  
        }  
    }  
}  
(*) terminating trans file  
; trans writes
```

public void withdraw (double amount) {  
if (amount > 0 && (balance - amount) >= 0)  
this.balance -= amount;  
System.out.println ("withdraw successful");  
currentBalance = this.balance; }

```
R else {
    System.out.println ("Withdraw is not
possible");
```

1.  $\text{mild} = \text{mild}$  8th

```
class SavingsAccount {  
    private double interestRate;  
    private Account account;
```

~~public SavingsAccount [ storing customer, storing acc no, double balance, double interest ] }~~

this. interestrate = interestrate ;

this.account = new Account(custName, accNo, balance);

public void addInterest() {  
double interest = account.getBalance() \*  
this.interestRate; ~~the . setBalance~~  
account.setDeposit(~~interest~~; ~~id~~ take  
1}

```
public Account getAccount () {  
    return account ;
```

DATE: PAGE:

class CurrentAccount {  
private double minBalance;  
private Account account;  
public CurrentAccount (String custName, String  
accNo, double balance, double minBalance) {  
this.minBalance = minBalance;  
this.account = new Account (custName, accNo,  
balance);  
}  
public void withdraw (double amt) {  
if (amt > 0 && account.getBalance () - amt)   
>= minBalance) {  
account.withdraw (amt);  
}  
else {  
System.out.println ("Withdraw is not  
possible");  
}  
public Account getAccount () {  
return account;  
}  
public class Banking {  
public static void main (String args) {  
Customer c = new Customer ("System", "in");  
System.out.println ("Enter the name");  
}

string name = sc. nextline();  
 system.out.println("Enter the account  
number : ");  
 string accnt = sc.nextline();  
 while (true) {  
 system.out.println("Enter your choice : ");  
 system.out.println("1. Savings Account");  
 system.out.println("2. Current Account");  
 system.out.println("3. Exit");  
 int choice = sc.nextInt();  
 switch (choice) {  
 case 1:  
 system.out.println("Enter initial balance : ");  
 double savingsbalance = sc.nextDouble();  
 system.out.println("Enter interest rate : ");  
 double interestrate = sc.nextDouble();  
 savingsaccount savingsaccount = new savingsaccount  
 (name, accnt, savingsbalance, interestrate);  
 savingsaccount.addinterest();  
 break;  
 case 2:  
 system.out.println("Enter Balance");  
 double currentbalance = sc.nextDouble();  
 system.out.println("Enter minimum  
balance : ");  
 double minbalance = sc.nextDouble();  
 currentaccount currentaccount = new currentaccount  
 (name, accnt, currentbalance, minbalance);

System.out.println("Enter the amount to withdraw");  
double q = sc.nextInt();  
currentAccount.withdraw(q);  
System.out.println("Account inserted - current balance : " + currentAccount.getBalance());  
break;  
  
case 3 :  
System.out.println("Exiting ...");  
sc.close();  
return;  
  
default :  
System.out.println("invalid choice");  
  
int r ;  
r ;  
r ;  
r ;  
r ;  
  
output :  
Enter the name :  
ansh  
Enter the account number :  
12345  
Enter your choice :  
1. savings account

2. current account
3. Exit

1  
Enter initial balance :

54,321

Enter interest rate :

9

The current balance is 5323530.0

: some : transfer  
: some alt. rates 3

: relevant transfers alt. rates 3

: entry relay rates 3

: trans. exch. 1

Code:

```
import java.util.Scanner;

class Account {
    private String custName;
    private String accNo;
    private double balance;

    public Account(String custName, String accNo, double balance) {
        this.custName = custName;
        this.accNo = accNo;
        this.balance = balance;
    }

    public double getBalance() {
        return this.balance;
    }

    public void deposit(double amount) {
        if (amount > 0) {
            this.balance += amount;
            System.out.println("The current balance is " + this.balance);
        } else {
            System.out.println("Amount should not be negative");
        }
    }

    public void withdraw(double amount) {
        if (amount > 0 && (balance - amount) >= 0) {
            this.balance -= amount;
            System.out.println("Withdraw successful. Current balance: " + this.balance);
        } else {
            System.out.println("Withdraw is not possible");
        }
    }
}

class SavingsAccount {
    private double interestRate;
    private Account account;

    public SavingsAccount(String custName, String accNo, double balance, double interestRate) {
        this.interestRate = interestRate;
        this.account = new Account(custName, accNo, balance);
    }

    public void addInterest() {
        double interest = account.getBalance() * this.interestRate;
        account.deposit(interest);
    }

    public Account getAccount() {
        return account;
    }
}

class CurrentAccount {
    private double minBalance;
    private Account account;

    public CurrentAccount(String custName, String accNo, double balance, double minBalance) {
        this.minBalance = minBalance;
        this.account = new Account(custName, accNo, balance);
    }

    public void withdraw(double amt) {
        if (amt > 0 && (account.getBalance() - amt) >= minBalance) {
            account.withdraw(amt);
        } else {
            System.out.println("Withdraw is not possible");
        }
    }

    public Account getAccount() {
        return account;
    }
}
```

```

public class Bank {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the name:");
        String name = sc.nextLine();
        System.out.println("Enter the account number:");
        String accnt = sc.nextLine();

        while (true) {
            System.out.println("Enter your choice:");
            System.out.println("1. Savings Account");
            System.out.println("2. Current Account");
            System.out.println("3. Exit");
            int choice = sc.nextInt();

            switch (choice) {
                case 1:
                    System.out.println("Enter initial balance:");
                    double savingsBalance = sc.nextDouble();
                    System.out.println("Enter the interest rate:");
                    double interestRate = sc.nextDouble();
                    SavingsAccount savingsAccount = new SavingsAccount(name, accnt, savingsBalance, interestRate);
                    savingsAccount.addInterest();
                    break;

                case 2:
                    System.out.println("Enter initial balance:");
                    double currentBalance = sc.nextDouble();
                    System.out.println("Enter minimum balance:");
                    double minBalance = sc.nextDouble();
                    CurrentAccount currentAccount = new CurrentAccount(name, accnt, currentBalance, minBalance);

                    System.out.println("Enter the amount to be withdraw");
                    double q = sc.nextInt();
                    currentAccount.withdraw(q);
                    System.out.println("Account created. Current balance: " + currentAccount.getAccount().getBalance());
                    break;

                case 3:
                    System.out.println("1. Savings Account");
                    System.out.println("2. Current Account");
                    System.out.println("3. Exit");
                    int choice = sc.nextInt();

                    switch (choice) {
                        case 1:
                            System.out.println("Enter initial balance:");
                            double savingsBalance = sc.nextDouble();
                            System.out.println("Enter the interest rate:");
                            double interestRate = sc.nextDouble();
                            SavingsAccount savingsAccount = new SavingsAccount(name, accnt, savingsBalance, interestRate);
                            savingsAccount.addInterest();
                            break;

                        case 2:
                            System.out.println("Enter initial balance:");
                            double currentBalance = sc.nextDouble();
                            System.out.println("Enter minimum balance:");
                            double minBalance = sc.nextDouble();
                            CurrentAccount currentAccount = new CurrentAccount(name, accnt, currentBalance, minBalance);

                            System.out.println("Enter the amount to be withdraw");
                            double q = sc.nextInt();
                            currentAccount.withdraw(q);
                            System.out.println("Account created. Current balance: " + currentAccount.getAccount().getBalance());
                            break;

                        case 3:
                            System.out.println("Exiting...");
                            sc.close();
                            return;

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

```

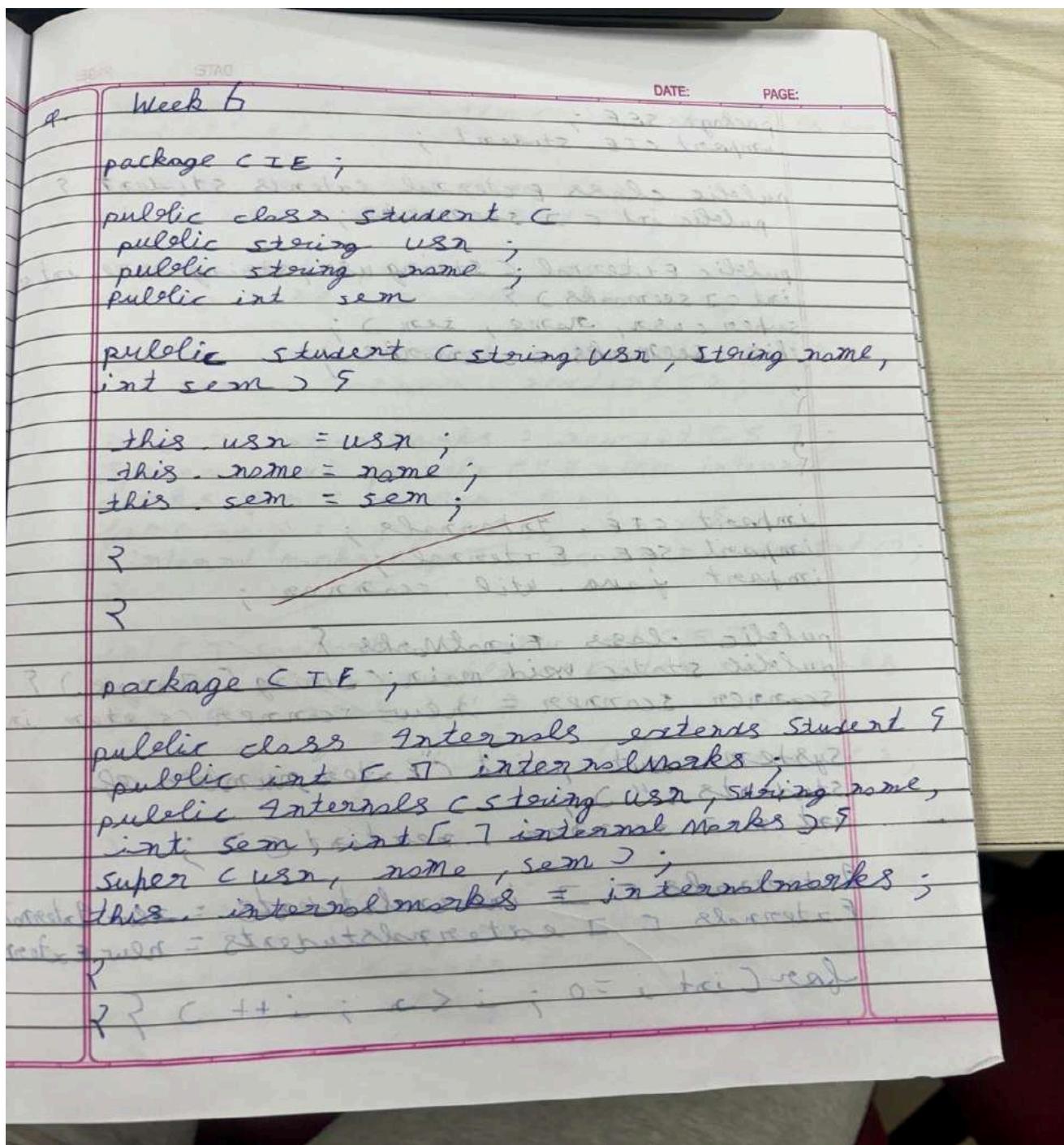
```
C:\Users\Admin\Desktop>javac Bank.java

C:\Users\Admin\Desktop>java Bank
Enter the name:
annas
Enter the account number:
12345
Enter your choice:
1. Savings Account
2. Current Account
3. Exit
1
Enter initial balance:
54321
Enter the interest rate:
7
The current balance is 434568.0
Enter your choice:
1. Savings Account
2. Current Account
3. Exit
annas sharieff 1bm23cs041
```



Program 6- Packages:

Algorithm:



DATE: PAGE:

```

package SEE;
import CIF.Student;

public class External extends Student {
    public int cmarks;
    public External (String user, String name, int marks) {
        super (user, name, marks);
        this .seemarks = seemarks;
    }
}

import CIF.Internals;
import SEE.External;
import java.util.Scanner;

public class FinalMarks {
    public static void main (String [ ] args) {
        Scanner scanner = new Scanner (System.in);
        System.out.print ("Enter number of students : ");
        int n = scanner.nextInt ();
        Internals [ ] internStudents = new Internals [n];
        External [ ] externalStudents = new External [n];
        for (int i = 0; i < n; i++) {
    }
}

```

DATE: PAGE:  
system.out.println("Enter details for  
student " + (i+1) + " ");

System.out.print("USN: ");  
String USN = scanner.next();

System.out.print("Name: ");  
String name = scanner.next();

System.out.print("Semester: ");  
int sem = scanner.nextInt();

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

int[] semMarks = new int[5];  
System.out.println("Enter SEE marks  
for 5 courses: ");  
for (int j = 0; j < 5; j++) {  
 semMarks[j] = scanner.nextInt();

internalStudents[i] = new InternalStudent(  
 USN, name, sem, internalMarks);  
externalStudents[i] = new ExternalStudent(  
 USN, name, sem, semMarks);

system.out.println ("1. n Final Marks of student")

```
for (int i=0; i < n; i++) {
```

```
    system.out.println ("student" + (i+1) + "-USN: " +
```

```
    internalStudents[i].USN);
```

```
for (int j=0; j < 5; j++) {
```

```
    int finalMark = (internalStudents[i].internalMarks
```

```
    + externalStudents[i].externalMarks[j]) / 12;
```

```
    system.out.println ("Course" + (j+1) + "
```

```
    final mark;" + finalMark);
```

```
}
```

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

```
}
```

```
}
```

~~output :~~

~~External details for student 1 :~~

~~USN : 16m23CS041~~

~~Name : Anusha~~

~~Semester : 3~~

~~External works : [ i ] student Internship~~

~~External internal works for 5 courses~~

~~50 mark = [ i ] student Internship~~

~~40 mark = [ i ] student Internship, MCA, MCA, MCA~~

~~32 mark = [ i ] student Internship, MCA, MCA, MCA~~

" );	36
" );	40
External	71
72	73
74	75
Final	
Student	
Course	
Course	
Course	
Courses	
Course	

"2 : 36  
40

Enter SEE marks for 5 courses :

71

72

73

74

75

Final marks of Students :

Student 1 - USN : 1bm23c5041

course 1 final mark : 66

course 2 final mark : 68

course 3 final mark : 69

course 4 final mark : 71

course 5 final mark : 72

Code:

```
package CIE;

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

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

public class Internals extends Student {
    public int[] internalMarks;

    public Internals(String usn, String name, int sem, int[] internalMarks) {
        super(usn, name, sem);
        this.internalMarks = internalMarks;
    }
}
package SEE;
import CIE.Student;

public class External extends Student {
    public int[] seeMarks;

    public External(String usn, String name, int sem, int[] seeMarks) {
        super(usn, name, sem);
        this.seeMarks = seeMarks;
    }
}
import CIE.Internals;
import SEE.External;
import java.util.Scanner;
```

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

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

        Internals[] internalStudents = new Internals[n];
        External[] externalStudents = new External[n];

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

            System.out.print("USN: ");
            String usn = scanner.next();

            System.out.print("Name: ");
            String name = scanner.next();

            System.out.print("Semester: ");
            int sem = scanner.nextInt();

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

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

            internalStudents[i] = new Internals(usn, name, sem, internalMarks);
            externalStudents[i] = new External(usn, name, sem, seeMarks);
        }
    }
}
```

```

        System.out.print("USN: ");
        String usn = scanner.next();

        System.out.print("Name: ");
        String name = scanner.next();

        System.out.print("Semester: ");
        int sem = scanner.nextInt();

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

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

        internalStudents[i] = new Internals(usn, name, sem, internalMarks);
        externalStudents[i] = new External(usn, name, sem, seeMarks);
    }

    System.out.println("\nFinal Marks of Students:");
    for (int i = 0; i < n; i++) {
        System.out.println("\nStudent " + (i + 1) + " - USN: " + internalStudents[i].usn);
        for (int j = 0; j < 5; j++) {
            int finalMark = (internalStudents[i].internalMarks[j] + (externalStudents[i].seeMarks[j]) / 2);
            System.out.println("Course " + (j + 1) + " Final Mark: " + finalMark);
        }
    }

    scanner.close();
}

```

```

Enter details for student 1:
USN: 1bm23cs041
Name: annas
Semester: 3
Enter internal marks for 5 courses:
31
32
33
34
35
Enter SEE marks for 5 courses:
71
72
73
74
75

Final Marks of Students:

Student 1 - USN: 1bm23cs041
Course 1 Final Mark: 66
Course 2 Final Mark: 68
Course 3 Final Mark: 69
Course 4 Final Mark: 71
Course 5 Final Mark: 72

```



## Program 7- Exception Handling:

Algorithm:

7

DATE: PAGE:

```
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 WrongException, SonAgeException  
super (fatherAge);  
if C sonAge >= fatherAge > {  
throw new SonAgeException ("son's age  
cannot be greater than or equal to father's  
age");}

R  
this.sonAge = sonAge;

R  
public int getSonAge () {  
return sonAge;}

R  
P  
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.println ("Enter Son's age : ");  
int sonAge = sc.nextInt ();

try {

81: if (A > B) print "A is greater than B"  
82: else if (B > A) print "B is greater than A"  
83: else print "A and B are equal"

DATE: PAGE:  
 San San = new San ("fatherage", "sonage")  
 System.out.println ("Accepted successfully");  
 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;  
 }

R: (2) Father's age = accepted tri  
 output: Enter Father's age : 18  
 Enter San's Age : 3  
 Accepted successfully

Would you like to reenter details (y/n)

y

Enter Father's Age : 3

Enter Son's Age : 15

Son's age cannot be greater or equal to  
father's Age.

Code:

```
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;
}
}
}
```

```
Command Prompt
Microsoft Windows [Version 10.0.22000.2538]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin>cd desktop

C:\Users\Admin\Desktop>javac FatherSon.java

C:\Users\Admin\Desktop>java FatherSon
Enter Father's Age: 18
Enter Son's Age: 3
Accepted Successfully
Would you like to re-enter details (Y/n)
y
Enter Father's Age: 3
Enter Son's Age: 15
Son's age cannot be greater than or equal to father's age
Would you like to re-enter details (Y/n)
n

C:\Users\Admin\Desktop>ANNAS SHARIEFF 1BM23CS041
```

## Program 8- Multithreading:

Algorithm:

The image shows handwritten code in a notebook, divided into two pages by a vertical red margin line. The left page contains code for the BMS class, and the right page contains code for the CSE class. Both classes extend the Thread class and implement the run() method. The BMS class uses a sleep of 1000 ms, while the CSE class uses a sleep of 2000 ms. Both classes catch InterruptedException. The main() method creates instances of BMS and CSE and starts them. The output section shows the alternating prints of "BMS College of Engineering" and "CSE".

DATE PAGE

class BMS extends Thread {  
 public void run() {  
 try {  
 while (true) {  
 System.out.println ("BMS College of  
Engineering");  
 Thread.sleep (1000);  
 }  
 } catch (InterruptedException e) {}  
 }  
}

public class Multithreading {  
 public static void main (String [] args) {  
 BMS bms = new BMS ();  
 CSE cse = new CSE ();  
 bms.start ();  
 cse.start ();  
 }  
}

Output

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

DATE PAGE

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

## Code:

```
Multithreading.java
File Edit View

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

public class Multithreading{
    public static void main(String[] args) {
        BMS bms = new BMS();
        CSE cse = new CSE();
        bms.start();
        cse.start();
    }
}

Annas Sharieff 1bm23cs041|
```

CSE

CSE

CSE

CSE

BMS College of Engineering

CSE

CSE

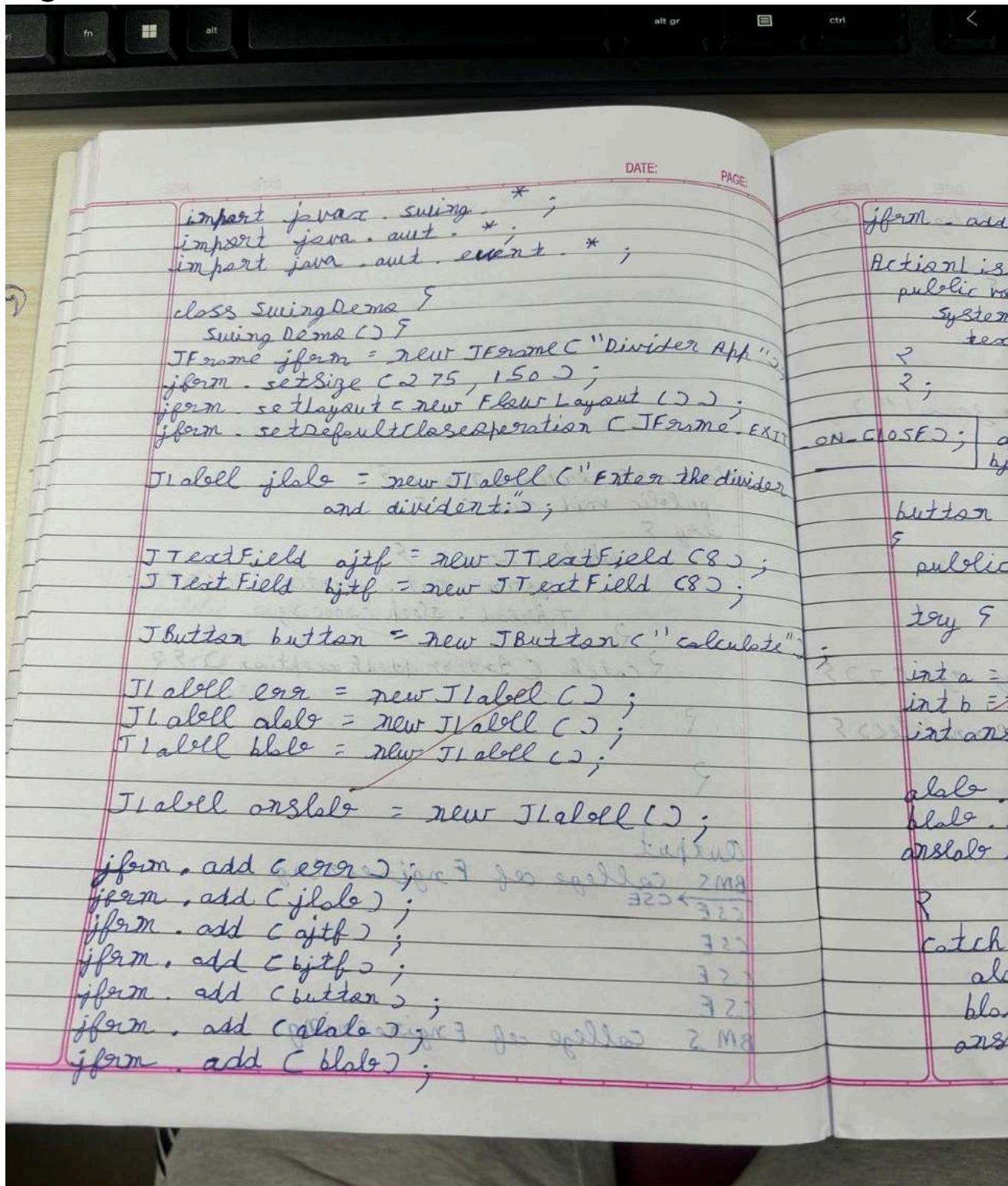
CSE

CSE

BMS College of Engineering

## Program 9- Integer Division With User Interface:

Algorithm:



jfrm.add(console);

ActionListener I = new ActionListener() {  
public void actionPerformed(ActionEvent evt) {  
System.out.println("Action event from a  
text field");  
}}

};

IT.ONCLOSED(); jtf.addActionListener(I);  
bjtf.addActionListener(I);

button.addActionListener(new ActionListener()) {

public void actionPerformed(ActionEvent evt) {

try {

int a = Integer.parseInt(jtf.getText());  
int b = Integer.parseInt(bjtf.getText());  
int ans = a \* b;

else. setText("In A = " + a);

else. setText("In B = " + b);

ans. setText("In Ans = " + ans);

R

catch (NumberFormatException e) {

else. setText("");

else. setText("");

ans. setText("");

```
err.setText("Enter only Integers!");
```

catch ArithmeticException

```
able.setText("");
```

```
blob.setText("111");
```

```
ansible.setText(" ");
```

err. setText C ("B should be Non zero !")

P 112 ~~success~~ ~~leaving~~ 113 114

*R*  
*2*

~~times~~ ~~concrete~~ ~~brackets~~ ~~reinforced~~

right) bacterial capsids may exhibit

```
ifrm.setVisible(true);
```

R

subdue static wind noise / strings -

*Swine Flu* - *the early days*

*inveklaten* (new run rule)  
public raid 9117 C-8

New Swing Demo ( )

$$d+1 = \text{axi}^3 + \text{axi}^2 + \text{axi}$$

~~850 ft = near 1 mi~~ the top of the hill

); ✓

~~(incorrect) tenses~~

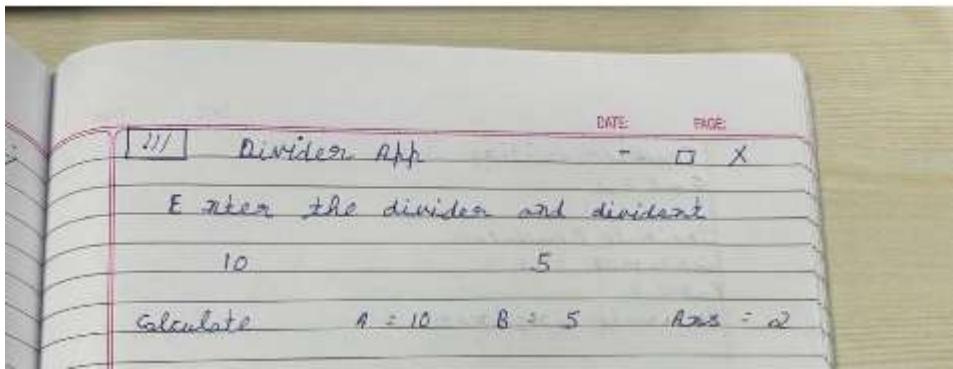
City Charter Revision

Constituyentes. Estado

*(11) top*

11-192-9108150

Figure 1. A schematic diagram of the experimental setup. The light source (labeled 1) is a pulsed Nd:YAG laser operating at 532 nm. The beam passes through a lens (labeled 2) and is focused onto a sample (labeled 3). The sample is a thin film of polyimide deposited on a substrate. The beam is reflected by the sample and passes through a lens (labeled 4) and a polarizer (labeled 5). The polarized light is detected by a photodiode (labeled 6).



Code:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
class SwingDemo{
SwingDemo(){
// create jframe container
JFrame jfrm = new JFrame("Divider App");
jfrm.setSize(275, 150);
jfrm.setLayout(new FlowLayout());
// to terminate on close
jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
// text label
JLabel jlab = new JLabel("Enter the divider and dividend:");
// add text field for both numbers
JTextField ajtf = new JTextField(8);
JTextField bjtf = new JTextField(8);
// calc button
JButton button = new JButton("Calculate");
// labels
JLabel err = new JLabel();
JLabel alab = new JLabel();
JLabel blab = new JLabel();
```

```

JLabel anslab = new JLabel();
// add in order :
jfrm.add(err); // to display error bois
jfrm.add(jlab);
jfrm.add(ajtf);
jfrm.add(bjtf);
jfrm.add(button);
jfrm.add(alab);
jfrm.add(blab);
jfrm.add(anslab);

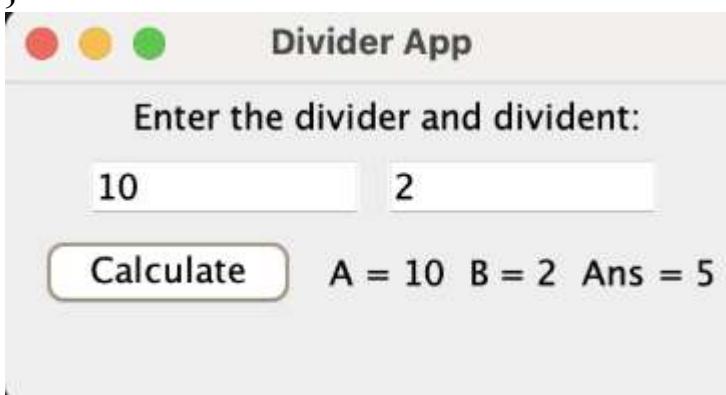
ActionListener l = new ActionListener() {
    public void actionPerformed(ActionEvent evt) {
        System.out.println("Action event from a text field");
    }
};

ajtf.addActionListener(l);
bjtf.addActionListener(l);
button.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent evt) {
        try{
            int a = Integer.parseInt(ajtf.getText());
            int b = Integer.parseInt(bjtf.getText());
            int ans = a/b;
            alab.setText("\nA = " + a);
            blab.setText("\nB = " + b);
            anslab.setText("\nAns = "+ ans);
        }
        catch(NumberFormatException e){
            alab.setText("");
            blab.setText("");
            anslab.setText("");
        }
    }
});

```

```
err.setText("Enter Only Integers!");
}
catch(ArithmetricException e){
alab.setText("");
blab.setText("");
anslab.setText("");
err.setText("B should be NON zero!");
}
}
});
// display frame
jfrm.setVisible(true);
}

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



## Program 10- IPC And Deadlock:

Algorithm:

Deadlock - code

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

class B {
    synchronized void bar(A a) {
        String name = Thread.currentThread().getName();
        System.out.println(name + " entered B - bar");
        try {
            Thread.sleep(1000);
        }
    }
}
```

```
catch (Exception e) {  
    System.out.println("B Interrupted");
```

```
    System.out.println(name + " trying  
    to call A.last()");  
    a.last();
```

R

```
void last () {
```

```
    System.out.println("Inside A.last");
```

R

{

```
class Deadlock implements Runnable {
```

```
    A a = new A();
```

```
    B b = new B();
```

```
Deadlock() {
```

```
    Thread currentThread = setName("main thread");
```

```
    Thread t = new Thread(this, "Racing Thread");
```

```
    t.start();
```

```
    a.foo(b);
```

```
    System.out.println("Back in main  
    thread");
```

{

```
public void run () {
```

```
    b.bar(a);
```

```
    System.out.println("Back in other  
    thread");
```

{

public static void main (String args [ ]) {

new deadlock () ;

?

?

## TPC Code

class OS

```
int n;
boolean valueset = false;
synchronized int get () {
    while (C : value set)
        try {
            System.out.println ("In consumer waiting for " + n);
            wait ();
        } catch (InterruptedException e) {
            System.out.println ("Interrupted Exception caught");
        }
    System.out.println ("Gof = " + n);
    valueset = false;
    System.out.println ("Informed Producer " + n);
    notify ();
    return n;
}
```

```
synchronized void put (int n) {
    while (valueset)
        try {
            System.out.println ("In Producer waiting");
            wait ();
        }
```

DATE: PAGE:  
catch (InterruptedException e) {  
 System.out.println ("Interrupted Exception  
caught");

R

this.n = n;  
valueset = true;  
System.out.println ("Put = " + n);  
System.out.println ("Estimate consumer ");

notify();

P P

class producer implements runnable {

Q q;

producer (Q q) {

this.q = q;

new thread (this, "producer").start();

R

public void run () {

int i = 0;

while (i < 15) {

q.put (i++);

R

Q Q

class consumer implements runnable {

Q q;

Consumer (q q) &  
this q = q;

new Thread (this, "consumer.start());

public void run () &

int i = 0;  
while (i < 15) &

int x = q.get();  
System.out.println ("Consumed: " + x);  
i++;

P

P

P

class PCFixed &

public static void main (String args [ ] ) &

q = new Q ();

new Producer (q);

new Consumer (q);

System.out.println ("Press control-c to stop");

P

P

~~8/6/2014  
8/1/2014~~

## 10. Inter process communication (IPC)

process central - C to start

Put = 0

Intimate consumer

producer waiting

Put = 0

Intimate Producer

Put = 1

Intimate consumer

producer waiting

consumed = 0

Get = 1

Intimate Producer

consumed = 1

Put = 2

Intimate consumer

X Producer waiting

Get = 2

Intimate producer

Consumed = 2

Put = 3

Intimate consumer

Producer waiting

Get = 3

Intimate producer

Consumed = 3

Put = 4

Intimate consumer

Get = 4

Intimate producer

Consumed = 4

Deadlock

Racing Thread entered B. list

Main Thread entered A. list

Racing Thread trying to call A. list

Main Thread trying to call B. list

class

Synch

stri

sys

fro

R c

sys

R

sy

ca

b

re

sy

ch

cl

sy

ch

cl

Code:

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

```
System.out.println("\nIntimate Consumer\n");
notify();
}
}
class Producer implements Runnable {
Q q;
Producer(Q q) {
this.q = q;
new Thread(this, "Producer").start();
}
public void run() {
int i = 0;
while(i<15) {
q.put(i++);
}
}
}
class Consumer implements Runnable {
Q q;
Consumer(Q q) {
this.q = q;
new Thread(this, "Consumer").start();
}
public void run() {
int i=0;
while(i<15) {
int r=q.get();
System.out.println("consumed:"+r);
i++;
}
}
```

```
}
```

```
class PCFixed {
```

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

```
Q q = new Q();
```

```
new Producer(q);
```

```
new Consumer(q);
```

```
System.out.println("Press Control-C to stop.");
```

```
}
```

```
}
```

## DEADLOCK

```
class A {
```

```
synchronized void foo(B b) {
```

```
String name = Thread.currentThread().getName();
```

```
System.out.println(name + " entered A.foo");
```

```
try {
```

```
Thread.sleep(1000);
```

```
} catch(Exception e) {
```

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

```
}
```

```
System.out.println(name + " trying to call B.last()");
```

```
b.last();
```

```
}
```

```
void last() {
```

```
System.out.println("Inside A.last");
```

```
}
```

```
}
```

```
class B {
```

```
synchronized void bar(A a) {
```

```
String name = Thread.currentThread().getName();
```

```
System.out.println(name + " entered B.bar");
```

```

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

class Deadlock implements Runnable
{
A a = new A();
B b = new B();
Deadlock() {
    Thread.currentThread().setName("MainThread");
    Thread t = new Thread(this,"RacingThread");
    t.start();
    a.foo(b); // get lock on a in this thread.
    System.out.println("Back in main thread");
}
public void run() {
    b.bar(a); // get lock on b in other thread.
    System.out.println("Back in other thread");
}
public static void main(String args[]) {
    new Deadlock();
}
}

```

```
Microsoft Windows [Version 10.0.22621.4169]
(c) Microsoft Corporation. All rights reserved.

C:\Users\smart>cd desktop

C:\Users\smart\Desktop>javac PCFixed.java

C:\Users\smart\Desktop>java PCFixed
Press Control-C to stop.
Put: 0

Intimate Consumer

Producer waiting

Got: 0

Intimate Producer

Put: 1

Intimate Consumer

Producer waiting

consumed:0
Got: 1
```

```
Intimate Producer
```

```
consumed:1
```

```
Put: 2
```

```
Intimate Consumer
```

```
Producer waiting
```

```
Got: 2
```

```
Intimate Producer
```

```
consumed:2
```

```
Put: 3
```

```
Intimate Consumer
```

```
Producer waiting
```

```
Got: 3
```

```
Intimate Producer
```

```
consumed:3
```

```
Put: 4
```

```
Intimate Consumer
```

```
Producer waiting
```

```
Got: 4
```

```
Intimate Producer
```

```
consumed:4
```

```
Put: 5
```

```
Intimate Consumer
```

```
Producer waiting
```

```
Got: 5
```

```
Intimate Producer
```

```
consumed:5
```

```
Put: 6
```

```
Intimate Consumer
```

```
Producer waiting
```

```
Producer waiting
```

```
Got: 6
```

```
Intimate Producer
```

```
consumed:6
```

```
Put: 7
```

```
Intimate Consumer
```

```
Producer waiting
```

```
Got: 7
```

```
Intimate Producer
```

```
consumed:7
```

```
Put: 8
```

```
Intimate Consumer
```

```
Producer waiting
```

```
Got: 8
```

```
Got: 8

Intimate Producer

consumed:8
Put: 9

Intimate Consumer

Producer waiting

Got: 9

Intimate Producer

consumed:9
Put: 10

Intimate Consumer

Producer waiting

Got: 10

Intimate Producer

consumed:10
Put: 11
```

```
Intimate Consumer
```

```
Producer waiting
```

```
Got: 11
```

```
Intimate Producer
```

```
consumed:11
```

```
Put: 12
```

```
Intimate Consumer
```

```
Producer waiting
```

```
Got: 12
```

```
Intimate Producer
```

```
consumed:12
```

```
Put: 13
```

```
Intimate Consumer
```

```
Producer waiting
```

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

Producer waiting
Got: 13
Intimate Producer
consumed:13
Put: 14
Intimate Consumer
Got: 14
Intimate Producer
consumed:14
C:\Users\smart\Desktop>annas sharieff 1bm23cs041|
```

```
MainThread entered A.foo
RacingThread entered B.bar
MainThread trying to call B.last()
Inside A.last
Back in main thread
RacingThread trying to call A.last()
Inside A.last
Back in other thread
```

