

**VISVESVARAYA TECHNOLOGICAL  
UNIVERSITY**

“JnanaSangama”, Belgaum -590014, Karnataka.



**LAB REPORT**

**on**

**Object Oriented Java Programming**

**(23CS3PCOOJ)**

*Submitted by*

Nagaraja G (**24BECs426**)

*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 **Nagaraja G (24BECS426)**, 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.

Srushti C S 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	30/9/24	Quadratic Equation	4-8
2	7/10/24	SGPA Calculation	9-15
3	14/10/24	Books toString	16-21
4	21/10/24	Finding Area of Rectangle ,Triangle and Circle by Using abstract Class & Method.	22-26
5	28/10/24	Banking System	27-42
6	11/11/24	Package Pgm (CIE & SEE)	43-55
7	18/11/24	Exception Handling	56-62
8	18/11/24	Thread Pgm	63-67
9	25/11/24	UI Pgm to Divide two Numbers	68-74
10	2/12/24	Open Ended Pgms 1.IPC 2.Deadlock	75-90

**Github Link:** <https://github.com/Naga1400/Java-Lab-Programs>

## Program 1

Develop a Java program that prints all real solutions to the quadratic equation  $ax^2+bx+c = 0$ . Read in a, b, c and use the quadratic formula. If the discriminate  $b^2-4ac$  is negative, display a message stating that there are no real solutions.

### Algorithm:

① S&S LAB 30/09/24  
Pgm→1 2:00 PM

② Quadratic Program

```
import java.util.Scanner;
class Quadratic {
    public static void main(String args[]) {
        int a=0, b, c;
        double d1, d2;
        Scanner s=new Scanner(System.in);
        while(a==0) {
            System.out.println("Enter the value
                for a:");
            a=s.nextInt();
        }
        if(a==0) {
            System.out.println("No possible
                solution for a if a = 0");
        }
        b=s.nextInt();
        System.out.println("Enter the value for
            b:");
        c=s.nextInt();
    }
}
```

~~System.out.println("Enter the value for
 b:");
 b=s.nextInt();
 System.out.println("Enter the value for
 c:");
 c=s.nextInt();~~

```

double d = (b*b) - (4*a*c);
d**2 ①

if (d==0) {
    r1 = (-b) / (2*a);
    System.out.println ("Roots are Real &
    Equal " + r1);
}

else if (d>0) {
    r1 = ((-b) + (Math.sqrt(d))) / (2*a);
    r2 = ((-b) - (Math.sqrt(d))) / (2*a);
    System.out.println ("Roots are Real");
    System.out.println ("Root 1 is: " + r1);
    System.out.println ("Root 2 is: " + r2);
}

else if (d<0) {
    r1 = (-b) / (2*a);
    r2 = Math.sqrt(-d) / (2*a);
    System.out.println ("Roots are
    Imaginary");
    System.out.println ("Root 1 is: " + r1);
    System.out.println ("Root 2 is: " + r2);
}

```

⑬

System.out.println("Name: Nogarao (6")

System.out.println("USN: 24BECs426")

Y  $\begin{cases} \text{when this mat is} \\ \text{done w/ fiber) print no, some} \\ \text{parts of bottom sheet. From} \\ \text{bottom w/ prints "jagged & free" } \\ \text{output is w/ "A" of values of} \end{cases}$

Enter the value for A.  $\begin{cases} \text{if no of fragm. } \\ \text{is 0} \end{cases}$

No possible solutions if A is 0

Enter the value for A.  $\begin{cases} \text{if no of fragm. } \\ \text{is 1} \end{cases}$

Enter the value for B.  $\begin{cases} \text{if no of fragm. } \\ \text{is 2} \end{cases}$

Enter the value for C.  $\begin{cases} \text{if no of fragm. } \\ \text{is 3} \end{cases}$

Roots are ambiguous

Root 1 is: -1.0

Root 2 is: 1.414213561937307

Name: Nogarao (6)

USN: 24BECs426

3. Oppn

3. Oppn

3. Oppn

3. Oppn

3. Oppn

## Code:

```
import java.util.Scanner;
class Quadratic{
    public static void main(String args[]){

        int a=0,b,c;
        double r1,r2;
        Scanner s=new Scanner(System.in);
        while (a==0){
            System.out.println("Enter the value for A :");
            a=s.nextInt();
            if
(a==0){
                System.out.println("No possible Solutions if A is 0 ");
}
        }
        System.out.println("Enter the value for B :");
        b=s.nextInt();
        System.out.println("Enter the value for C :");
        c=s.nextInt();
        double d=(b*b)-(4*a*c);
        if (d==0){
            r1=(-b)/(2*a);
            System.out.println("Roots are Real and Equal "+r1);
}
        else if(d>0){
            r1 = ((-b)+(Math.sqrt(d)))/(double)(2*a);
            r2 = ((-b)-(Math.sqrt(d)))/(double)(2*a);
            System.out.println("Roots are Real");
            System.out.println("Root 1 is :" +r1);
            System.out.println("Root 2 is :" +r2);
}
        else if(d<0){
            r1 = (-b)/(2*a);
            r2 = Math.sqrt(-d)/(2*a);
            System.out.println("Roots are Imaginary");
            System.out.println("Root 1 is :" +r1);
            System.out.println("Root 2 is :" +r2);
}

        System.out.println("Name: Nagaraja G");
        System.out.println("USN: 24BECS426");
    }
}
```

## Output:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS E:\24BEC5426> javac Quadratic.java
PS E:\24BEC5426> java Quadratic
Enter the value for A :
0
No possible Solutions if A is 0
Enter the value for A :
1
Enter the value for B :
2
Enter the value for C :
3
Roots are Imaginary
Root 1 is :-1.0
Root 2 is :1.4142135623730951
Name: Nagaraja G
USN: 24BEC5426
PS E:\24BEC5426> java Quadratic
Enter the value for A :
1
Enter the value for B :
4
Enter the value for C :
2
Roots are Real
Root 1 is :-0.5857864376269049
Root 2 is :-3.414213562373095
Name: Nagaraja G
USN: 24BEC5426
PS E:\24BEC5426> java Quadratic
Enter the value for A :
1
Enter the value for B :
4
Enter the value for C :
4
Roots are Real and Equal -2.0
Name: Nagaraja G
USN: 24BEC5426
PS E:\24BEC5426>
```

## 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.

### **Algorithm:**

07/10/24 ②  
11:45 AM  
Pgm-22

⑥ Develop a Java program to create a class "Student" with members "USN", "name", an array "credits" of an array "marks". Include methods to accept & display details of a student to calculate "SGPA" of a student.

→ import java.util.Scanner;  
class Student {  
 private String USN;  
 private String name;  
 private int[] credits;  
 private int[] marks;  
 private int numCourses;  
 public Student(int numCourses) {  
 this.numCourses = numCourses;  
 credits = new int[numCourses];  
 marks = new int[numCourses];  
 }  
 public void acceptDetails() {  
 Scanner scnew = new Scanner(System.in);  
 System.out.println("Enter USN:");  
 USN = scnew.nextLine();  
 System.out.println("Enter Name:");  
 name = scnew.nextLine();  
 for (int i = 0; i < numCourses; i++) {  
 System.out.println("Enter Credits for Course " + (i + 1) + ": ");  
 credits[i] = scnew.nextInt();  
 System.out.println("Enter Marks for Course " + (i + 1) + ": ");  
 marks[i] = scnew.nextInt();  
 }  
 }  
 public void displayDetails() {  
 System.out.println("USN: " + USN);  
 System.out.println("Name: " + name);  
 System.out.println("Credits: ");  
 for (int i = 0; i < numCourses; i++) {  
 System.out.print(credits[i] + " ");  
 }  
 System.out.println();  
 System.out.println("Marks: ");  
 for (int i = 0; i < numCourses; i++) {  
 System.out.print(marks[i] + " ");  
 }  
 }  
 public float calculateSGPA() {  
 float totalCredits = 0;  
 float totalMarks = 0;  
 for (int i = 0; i < numCourses; i++) {  
 totalCredits += credits[i];  
 totalMarks += marks[i];  
 }  
 float SGPA = (totalMarks / totalCredits);  
 return SGPA;  
 }  
}

```

    credits[i] = s.nextInt();
    System.out.println("Enter marks for course " + i + ":");
    marks[i] = s.nextInt();
}

credit for course 5.00
marks for course 5.00
public void displayDetails() {
    System.out.println("USN: " + usn);
    System.out.println("Name: " + name);
    for (int i = 0; i < numCourses; i++) {
        System.out.println("Course " + (i + 1) + ":");
        credits[i] += credits[i];
        marks[i] += marks[i];
    }
    System.out.println("CGPA: " + calculateCGPA());
}

public double calculateCGPA() {
    double totalCredits = 0;
    double weightedMarks = 0;
    for (int i = 0; i < numCourses; i++) {
        totalCredits += credits[i];
        weightedMarks += (marks[i] / 10.0) * credits[i];
    }
}

```

```
+ return (totalCredit >= 20) ? 0 :  
+     (WeightedMarks / totalCredit);  
+ }  
+ }  
+ }  
+ }  
+ }  
+ }  
+ }
```

```
public class Main {  
    public static void main(String args) {  
        Scanner reader = new Scanner(System.in);  
        System.out.println("Enter number of  
        courses:");  
        int numCourses = Integer.parseInt(reader.  
            nextLine());  
        Student student = new Student(numCourses);  
        student.acceptDetails();  
        student.displayDetails();  
    }  
}
```

Output } ~~(1) Enter student details~~

Enter number of courses: 9

Enter USN: 20CS426

Enter Name: Raja

Enter Credits for course 1: 10

Enter Marks for course 1: 80

Enter Credits for course 2: 20

Enter Marks for course 2: 60

② Enter Credits for course 1: 30 : 10.9.21  
Enter Marks for course 1: 70  
Enter Credits for course 2: 40  
Enter Marks for course 2: 70  
Enter Credits for course 3: 50  
Enter Marks for course 3: 80  
Enter Credits for course 4: 60  
Enter Marks for course 4: 70  
Enter Credits for course 5: 70  
Enter Marks for course 5: 80  
Enter Credits for course 6: 10  
Enter Marks for course 6: 70  
Enter Credits for course 7: 10  
Enter Marks for course 7: 100  
Enter Credits for course 8: 10  
Enter Marks for course 8: 56  
Enter Credits for course 9: 10  
Enter Marks for course 9: 65

UIN: 20CS426

Name: Naveen

Course 1: Credits = 10, Marks = 70

Course 2: Credits = 10, Marks = 60

Course 3: Credits = 30, Marks = 70

Course 4: Credits = 40, Marks = 70

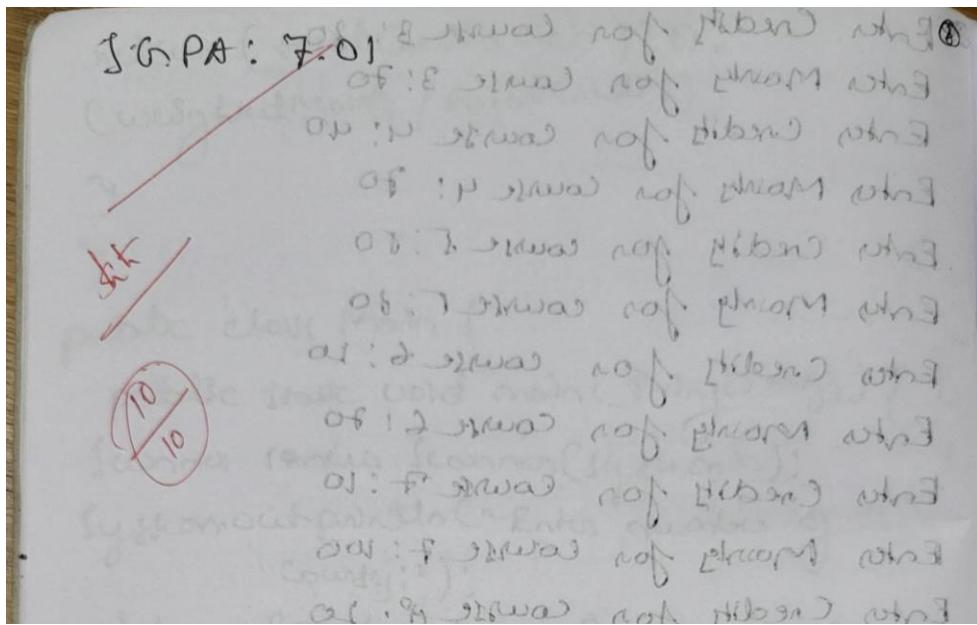
Course 5: Credits = 50, Marks = 80

Course 6: Credits = 10, Marks = 70

Course 7: Credits = 10, Marks = 100

Course 8: Credits = 10, Marks = 56

Course 9: Credits = 10, Marks = 65



## Code:

```

import java.util.Scanner;

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

  public Student(int numCourses) {
    this.numCourses = numCourses;
    credits = new int[numCourses];
    marks = new int[numCourses];
  }

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

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

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

    for (int i = 0; i < numCourses; i++) {
      System.out.print("Enter Credits for Course " + (i + 1) + ": ");
      credits[i] = scanner.nextInt();
      System.out.print("Enter Marks for Course " + (i + 1) + ": ");
      marks[i] = scanner.nextInt();
    }
  }

  public void displayDetails() {
    System.out.println("Student Details:");
    System.out.println("USN: " + usn);
    System.out.println("Name: " + name);
    System.out.println("Number of Courses: " + numCourses);
    System.out.println("Credits and Marks details:");
    for (int i = 0; i < numCourses; i++) {
      System.out.println("Course " + (i + 1) + " Credits: " + credits[i] + " Marks: " + marks[i]);
    }
  }
}
  
```

```

        System.out.print("Enter credits for course " + (i + 1) + ": ");
        credits[i] = scanner.nextInt();

        System.out.print("Enter marks for course " + (i + 1) + ": ");
        marks[i] = scanner.nextInt();
    }
}

public void displayDetails() {
    System.out.println("USN: " + usn);
    System.out.println("Name: " + name);
    for (int i = 0; i < numCourses; i++) {
        System.out.println("Course " + (i + 1) + ": Credits = " + credits[i] + ", Marks = " + marks[i]);
    }
    System.out.printf("SGPA: %.2f%n", calculateSGPA());
}

public double calculateSGPA() {
    double totalCredits = 0;
    double weightedMarks = 0;

    for (int i = 0; i < numCourses; i++) {
        totalCredits += credits[i];
        weightedMarks += (marks[i] / 10.0) * credits[i];
    }

    return (totalCredits == 0) ? 0 : (weightedMarks / totalCredits);
}

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

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

        Student student = new Student(numCourses);

        student.acceptDetails();

        student.displayDetails();
    }
}

```

## Output:

```
PS D:\24CSBE426> javac Main.java
PS D:\24CSBE426> java Main
Enter number of courses: 9
Enter USN: 20cs426
Enter Name: Naga
Enter credits for course 1: 10
Enter marks for course 1: 50
Enter credits for course 2: 20
Enter marks for course 2: 60
Enter credits for course 3: 30
Enter marks for course 3: 70
Enter credits for course 4: 40
Enter marks for course 4: 70
Enter credits for course 5: 50
Enter marks for course 5: 80
Enter credits for course 6: 20
Enter marks for course 6: 70
Enter credits for course 7: 10
Enter marks for course 7: 100
Enter credits for course 8: 20
Enter marks for course 8: 56
Enter credits for course 9: 20
Enter marks for course 9: 65
USN: 20cs426
Name: Naga
Course 1: Credits = 10, Marks = 50
Course 2: Credits = 20, Marks = 60
Course 3: Credits = 30, Marks = 70
Course 4: Credits = 40, Marks = 70
Course 5: Credits = 50, Marks = 80
Course 6: Credits = 20, Marks = 70
Course 7: Credits = 10, Marks = 100
Course 8: Credits = 20, Marks = 56
Course 9: Credits = 20, Marks = 65
SGPA: 7.01
PS D:\24CSBE426> |
```

## 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.

### **Algorithm:**

③ Pgm → 3 } (0 points) point 14/10/24  
③ Create a class Book which contains four members:  
Member: Name, Author, Price, num\_pages.  
Include a constructor to set the values for the members. Include methods to set and get the details of the objects. Include a `toString()` method that could display the complete details of the book. Develop a Java program to create n book objects.

```
→ import java.util.Scanner;  
class Book {  
    String name;  
    String author;  
    int price;  
    int numPages;  
    public Book(String name, String author, int price, int numPages) {  
        this.name = name;  
        this.author = author;  
        this.price = price;  
        this.numPages = numPages;  
    }  
    public String toString() {  
        return "Book{" + "name=" + name + ", author=" + author + ", price=" + price + ", numPages=" + numPages + '}';  
    }  
}
```

```

public String toString() {
    return ("A Book name is :" + name +
           "Author name is :" + author +
           "Book price is :" + price +
           "Number of pages is :" + numPages);
}

public static void main (String args[]) {
    Scanner s = new Scanner (System.in);
    int n;
    String name;
    String author;
    int price;
    int numPages;
    Book b[] = new Book[n];
    System.out.println ("Enter the book name");
    name = s.nextLine();
    System.out.println ("Enter the book author");
    author = s.nextLine();
    System.out.println ("Enter the book price");
    price = s.nextInt();
    b[0] = new Book(name, author, price);
    for (int i=1; i<n; i++) {
        System.out.println ("Enter the book name");
        name = s.nextLine();
        System.out.println ("Enter the book author");
        author = s.nextLine();
        System.out.println ("Enter the book price");
        price = s.nextInt();
        b[i] = new Book(name, author, price);
    }
}

```

```

① System.out.println("Enter the number of pages in a book");
    numPage = s.nextInt();
    b[i] = new Book(name, author, price, numPage);
}

for (int j = 0; j < n; j++) {
    System.out.println("Book details: " + b[j].toString());
}

```

Output:

Enter the how many books you want?

2

Enter the Book name:

Cloud

Enter the Author name:

noya

Enter the Price:

200

Enter the Number of pages in a book;

500

Enter the Book name is : Jatinga two-metres

Java book is of 1000

Enter the Author name : Moshesh

Moshesh (Author name) Java book is of 1000

Enter the price :

300

Enter the number of pages for Book : 1000

Book details : book is of 1000

Book name is : Clouds over I.P.D.

Author name is : nagesh

Book price is : 200

Number of pages is : 500

Book details

Book name is : Java

Author name is : Moshesh

Book price is : 300

Number of pages is : 1000

Review copy book is of 1000 random words with writing  
16/10/24

10/10

random book with writing

book

random words with writing

copy

copy with writing

book

book is of 1000 of random words with writing

book

## **Code:**

```
import java.util.Scanner;
class Books{
    String name;
    String author;
    int price;
    int numPages;

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

    public String toString(){
        return ("Book name is :" +name+
                "\nAuthor name is :" +author+
                "\nBook price is :" +price+
                "\nNumber of pages is :" +numPages);
    }

    public static void main(String args[]){
        Scanner s=new Scanner(System.in);
        int n;
        String name;
        String author;
        int price;
        int numPages;

        Books b[];

        System.out.println("Enter the How many books you want ?");
        n=s.nextInt();
        b=new Books[n];

        for(int i=0;i<n;i++){
            System.out.println("Enter the book name :");
            name=s.next();
        }
    }
}
```

```

System.out.println("Enter the author name :");
author=s.next();
System.out.println("Enter the Price :");
price=s.nextInt();
System.out.println("Enter the Number of Pages in Book :");
numPages=s.nextInt();
b[i] = new Books(name,author,price,numPages);
}

for(int j=0;j<n;j++){
System.out.println("Book details \n"+b[j].toString());
}
}
}

```

## Output:

```

PS D:\24CSBE426> java Books
Enter the How many books you want ?
2
Enter the book name :
cloud
Enter the author name :
naga
Enter the Price :
200
Enter the Number of Pages in Book :
1000
Enter the book name :
Java
Enter the author name :
Mahesh
Enter the Price :
300
Enter the Number of Pages in Book :
500
Book details
Book name is :cloud
Author name is :naga
Book price is :200
Number of pages is :1000
Book details
Book name is :Java
Author name is :Mahesh
Book price is :300
Number of pages is :500

```

## 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.

### **Algorithm:**

② LAB Pg m → 04  
21/10/20  
2:00 PM

Develop a Java program to create an abstract class named Shape that contains two integers and an empty method named printArea( ). Provide 3 classes named Rectangle, Triangle & 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.

(a) that prints area of the given shape.

```
→ import java.util.Scanner;
abstract class Shape {
    double val1, val2, result;
    public void printArea();
    Scanner snew = new Scanner(System.in);
}

class Rectangle extends Shape {
    Rectangle() {
        System.out.println("Enter the two values to calculate Area of Rectangle:");
        val1 = snew.nextDouble();
        val2 = snew.nextDouble();
    }
    void printArea() {
        result = val1 * val2;
        System.out.println("Area of Rectangle is: " + result);
    }
}
```

Netwrok  
 no. of sides or unequal sides is given  
 Inequal sides equal to three sides  
 class Triangle extends Shape of parent class  
~~Triangle C~~  
 System.out.println("Enter the two  
 values to calculate Area of triangle  
 triangle: ");  
 val1 = sc.nextDouble();  
 val2 = sc.nextDouble();  
 value < nextDouble;  
 }  
 (uncommented line of program)  
 void printArea(){  
 result = (val1 \* val2) / 2;  
 System.out.println("The Area of  
 triangle is: "+result);  
 }

}  
 { square's border is printed, then  
 class Circle extends Shape  
 Circle C{  
 public void printArea(){  
 System.out.println("Enter the one  
 value to calculate Area of  
 Circle: ");  
 val1 = sc.nextDouble();  
 value < nextDouble;

}  
 (Uncommented line  
 void printArea(){  
 result = 3.14 \* val1 \* val1;

② System.out.println (or Area of Circle is: " + result);

```
Y  
Y  
class ShapeDemo {  
    public static void main(String args[]) {  
        Rectangle r = new Rectangle();  
        r.printArea();  
        Triangle t = new Triangle();  
        t.printArea();  
        Circle c = new Circle();  
        c.printArea();  
    }  
}
```

Output:  
Enter the two value to calculate area  
of Rectangle:

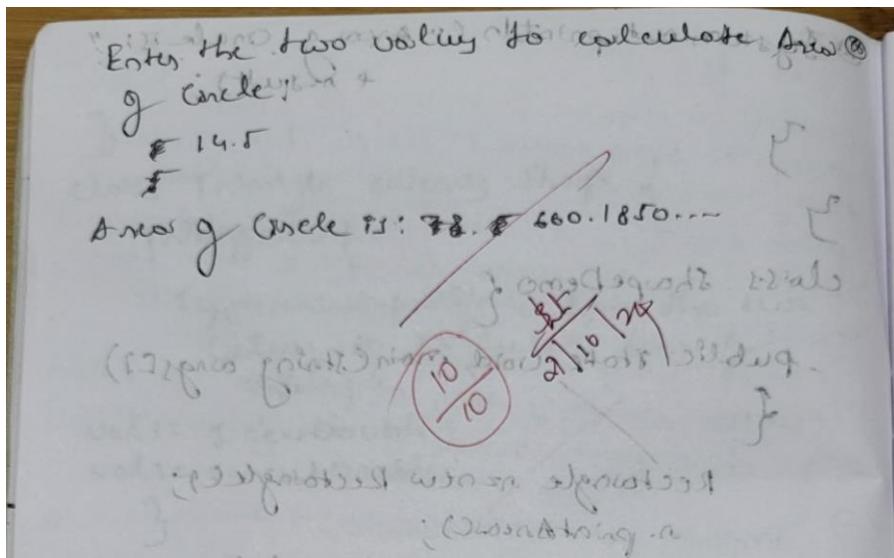
10  
20

Area of Rectangle is: 200.0

Enter the two value to calculate Area  
of Triangle:

1  
2

Area of Triangle is: 1.0



## Code:

```

import java.util.Scanner;

abstract class Shape{
    double val1,val2,result;
    abstract void printArea();
    Scanner s=new Scanner(System.in);
}

class Rectangle extends Shape{
    Rectangle(){
        System.out.println("Enter the Two values to Calculate Area of Rectangle :");
        val1=s.nextDouble();
        val2=s.nextDouble();
    }
    void printArea(){
        result=val1*val2;
        System.out.println("Area of Rectangle is :" +result);
    }
}

class Triangle extends Shape{
    Triangle(){
        System.out.println("Enter the Two values to Calculate Area of Triangle :");
        val1=s.nextDouble();
    }
}
    
```

```

val2=s.nextDouble();
}
void printArea(){
result=(val1*val2)/2;
System.out.println("Area of Triangle is :" + result);
}
}

class Circle extends Shape{
Circle(){
System.out.println("Enter the One value to Calculate Area of Circle :");
val1=s.nextDouble();
}
void printArea(){
result=3.14*val1*val1;
System.out.println("Area of Circle is :" + result);
}
}

class shapeDemo{
public static void main(String args[]){
Rectangle r=new Rectangle();
r.printArea();
Triangle t=new Triangle();
t.printArea();
Circle c=new Circle();
c.printArea();
System.out.println("Name: Nagaraja G \n USN: 24BECS426");
}
}

```

## **output:**

```

PS D:\1bm23cs156> javac shapeDemo.java
PS D:\1bm23cs156> java shapeDemo
Enter the Two values to Calculate Area of Rectangle :
10
20
Area of Rectangle is :200.0
Enter the Two values to Calculate Area of Triangle :
1
2
Area of Triangle is :1.0
Enter the One value to Calculate Area of Circle :
14.5
Area of Circle is :660.1850000000001
Name: Nagaraja G
USN: 24BECS426
PS D:\1bm23cs156> |

```

## **Program 5**

Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called savings account and the other current account. The savings account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if the balance falls below this level, a service charge is imposed.

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
- e) Check for the minimum balance, impose penalty if necessary and update the balance.

## Algorithm:

⑧ ⑨  
LAB PgM → 05      Application given: 28/10/2014  
                                2:00PM

\* Develop a Java program to create a class Bank that maintains two kinds of account for its customers, one called "Savings Account" & other "Current Account". The Savings Account provides compound interest & withdrawal facility but no cheque book facility. The Current Account provides cheque book facility but no interest. Current account holder should also maintain minimum balance & if the balance falls below this level, a service charge is imposed.

\* Create a class Account that stores customer name, account No. & type of account. From this derive the classes "Current Account" & "Savings Account" to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks:-

- a) Accept deposit from customer & update the balance.
- b) Display the balance.
- c) Compute & deposit interest.
- d) Permit withdrawal & update the balance.
- e) Check for the minimum balance, impose penalty if necessary & update the balance.

→ import java.util.Scanner;  
class Account {  
    protected String customerName;  
    protected String accountNumber;  
    protected double balance;  
}

protected string accountType;  
 public Account(string customerName, string  
 accountNumber, double initialDeposit,  
 string accountType) {  
 this.customerName = customerName;  
 this.accountNumber = accountNumber;  
 this.balance = initialDeposit;  
 this.accountType = accountType; // sets an  
 // account type to the default value of savings  
 public void deposit(double amount) {  
 if (amount > 0) {  
 balance += amount;  
 System.out.println("Deposit  
 successful. Updated Balance: " + balance);  
 } else {  
 System.out.println("Deposit amount  
 must be positive!");  
 }  
 }  
 public void displayBalance() {  
 System.out.println("Balance is " +  
 balance);  
 }  
 class SaverAccount extends Account {  
 protected double interestRate;  
 public SaverAccount(string customerName,

existing account number, double Initial Deposit, double  
Interest Rate) {  
    super (Customer Name, account number, Initial Deposit,  
    ("Savings"));

this. InterestRate = interestRate;

y account (account number) age  
    ("Customer", "Savings");

public void computeDepositedInterest() {

    double interest = balance \* (interestRate / 100);

    balance += interest;

    System.out.println("Interest Deposited:"

        + interest +  
        "Updated Balance:" + balance);

public void withdraw (double amount) {

    if (amount <= balance) {

        balance -= amount;

        System.out.println("Withdrawal

    Successful. Updated balance" +  
    balance);

} else {

    System.out.println("Insufficient

    balance for withdrawal.");

} else {  
    System.out.println("Cancelled withdrawal");

    balance += canceledBalance;

}

class Customer extends Account {

```

private static final double minBalance = 2000;
private static final double serviceCharge = 50;
public Customer(String customerName,
                 String accountNumber, double initialDeposit)
}

2
super(customerName, accountNumber,
      initialDeposit, "Current");
y ) withdraw(double amount)
public void withdraw(double amount) {
    if (amount <= balance) {
        balance -= amount;
        System.out.println("Withdrawal successful.");
        System.out.println("Updated balance: " + balance);
    } else {
        System.out.println("Insufficient balance");
        System.out.println("Balance after withdrawal: " +
                           + balance);
    }
}

3
private void checkMinBalance() {
    if (balance < minBalance) {
        balance -= serviceCharge;
        System.out.println("Balance fell below minimum. Service charge imposed.");
        System.out.println("Updated balance: " + balance);
    }
}

3 3 } main: Main class method main

```

```

public class Bank {
    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: ");
        String accountNumber = scanner.nextLine();
        System.out.print("Enter Initial Deposit: ");
        double initialDeposit = scanner.nextDouble();

        System.out.print("Enter Account Type (1 for Savings, 2 for Current): ");
        int accountType = scanner.nextInt();

        Account account;
        if (accountType == 1) {
            System.out.print("Enter Interest rate for Savings Account: ");
            double interestRate = scanner.nextDouble();
            account = new Savings(customerName,
                accountNumber, initialDeposit,
                interestRate);
        } else {
            account = new Current(
                customerName, accountNumber,
                initialDeposit);
        }
    }
}

```

```

3
while (true) {
    System.out.println("1. Deposit");
    System.out.println("2. Withdraw");
    System.out.println("3. Display Balance");
    System.out.println("4. Exit");
    System.out.print("Enter your choice: ");
    int choice = Integer.parseInt(br.readLine());
    if (choice == 1) {
        System.out.print("Enter amount to deposit: ");
        double deposit = Double.parseDouble(br.readLine());
        account.deposit(deposit);
        System.out.println("Amount deposited successfully");
    } else if (choice == 2) {
        System.out.print("Enter amount to withdraw: ");
        double withdrawl = Double.parseDouble(br.readLine());
        account.withdraw(withdrawl);
        System.out.println("Amount withdrawn successfully");
    } else if (choice == 3) {
        System.out.println("Your current balance is " + account.getBalance());
    } else if (choice == 4) {
        System.out.println("Thank you for using our services");
        break;
    }
}

```

2.

case 3:

```

if (account instanceof SavAcc) {
    ((SavAcc) account).compute
        - DepositedAmount();
} else { // which atm opn
    System.out.println("option not
available for current accounts.");
}

```

} break;

case 4: System.out.println("Enter amount to withdraw:");

double withdrawamt = r.nextDouble();

if (account instanceof SavAcc) {

((SavAcc) account).withdraw

(withdrawamt);

} else {

((Current) account).withdraw

(withdrawamt);

} break; // loop continues

case 5: System.out.println("Thank you for using the bank system.");

s.close();

return;

default: System.out.println("Invalid

option. Please try again.");

} // end of switch loop

y . You can withdraw.

break; }

2 (four) percent increase (6)

Output: `DisplayOutput()` `(void)(accm)`

\* Enter customer Name: [ ]

Nwgo - n wgo - J alting - two m t g -

(c) Enter account number if applicable

1234 Island L

Enter Initial Deposit:

Enter Account Type (1 for savings,  
(Debit Card - ATM card) & 2 for current);

f (successor function) f

Enter Interest Rate for Savings Account

10 (Intervention)

## Banking Meas.

## 1. Deposit

## 2. Display Balance Sheet

### 3. Computation of Deposit Interest

40 withdraw (attaching two notes) 17 May

8. Exit (multiple choice options)

Select an option: 2

Bolwne: 500.0

Banking Money; Banking term: deposit

## 1. Deposit

2. Display Balancer  
3. Display Balancer

3. complete f Deposits

4. with ~~draw~~

R. RUST

1820

Select an option: 1

Enter amount to deposit: 100

Deposit successful. Updated Balance: 600.00

Banking menu:

1. Deposit

2. Display Balance

3. Compute & Deposit Interest

4. Withdraw

5. Exit

Select an Option: 3

Interest Deposited: 60.00. Updated Balance  
-: 660.0. 0.003

Banking menu:

1. Deposit

2. Display Balance

3. Compute & Deposit Interest

4. Withdraw

5. Exit

Select an Option: 4

Enter amount to withdraw: 100

Withdrawal successful. Withdraw Balance

-: 500.0

Banking menu:

1. Deposit

2. Display Balance

3. Compute & Deposit Interest

4. Withdraw

5. Exit

Select an Option: 5

Thank you for using the bank system.

## Code:

```
import java.util.Scanner;

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

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

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

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

class SavAcct extends Account {
    private double interestRate;

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

    public void computeAndDepositInterest() {
```

```

        double interest = balance * (interestRate / 100);
        balance += interest;
        System.out.println("Interest Deposited: " + interest + ", Updated Balance: " + balance);
    }

    public void withdraw(double amount) {
        if (amount <= balance) {
            balance -= amount;
            System.out.println("Withdrawal Successful. Updated Balance: " + balance);
        } else {
            System.out.println("Insufficient balance for withdrawal.");
        }
    }

    class CurAcct extends Account {
        private static final double MINIMUM_BALANCE = 500;
        private static final double SERVICE_CHARGE = 50;

        public CurAcct(String customerName, String accountNumber, double initialDeposit) {
            super(customerName, accountNumber, initialDeposit, "Current");

            if (initialDeposit < MINIMUM_BALANCE) {
                System.out.println("Error: Initial deposit must be at least " + MINIMUM_BALANCE);
                this.balance = 0;
            }
        }

        public void deposit(double amount) {
            super.deposit(amount);
            if (balance >= MINIMUM_BALANCE) {
                System.out.println("Your account meets the minimum balance requirement.");
            }
        }

        public void withdraw(double amount) {
            if (balance - amount < MINIMUM_BALANCE) {
                System.out.println("Warning: Withdrawing this amount will bring your balance below the
minimum. A service charge of "
                        + SERVICE_CHARGE + " will be applied.");
            }
        }
    }
}

```

```

if (amount <= balance) {
    balance -= amount;
    checkMinimumBalance();
    System.out.println("Withdrawal Successful. Updated Balance: " + balance);
} else {
    System.out.println("Insufficient balance for withdrawal.");
}
}

private void checkMinimumBalance() {
    if (balance < MINIMUM_BALANCE) {
        balance -= SERVICE_CHARGE;
        System.out.println("Balance fell below minimum. Service charge imposed. Updated Balance:
" + balance);
    }
}
}

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

        System.out.println("Enter Customer Name:");
        String customerName = scanner.nextLine();
        System.out.println("Enter Account Number:");
        String accountNumber = scanner.nextLine();
        System.out.println("Enter Initial Deposit:");
        double initialDeposit = scanner.nextDouble();
        System.out.println("Enter Account Type (1 for Savings, 2 for Current):");
        int accountType = scanner.nextInt();

        Account account;

        if (accountType == 1) {
            System.out.println("Enter Interest Rate for Savings Account:");
            double interestRate = scanner.nextDouble();
            account = new SavAcct(customerName, accountNumber, initialDeposit, interestRate);
        } else {
            account = new CurAcct(customerName, accountNumber, initialDeposit);
        }
    }
}

```

```

while (true) {
    System.out.println("\nBanking Menu:");
    System.out.println("1. Deposit");
    System.out.println("2. Display Balance");
    if (account instanceof SavAcct) {
        System.out.println("3. Compute and Deposit Interest");
    }
    System.out.println("4. Withdraw");
    System.out.println("5. Exit");
    System.out.print("Select an option: ");
    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:
            account.displayBalance();
            break;
        case 3:
            if (account instanceof SavAcct) {
                ((SavAcct) account).computeAndDepositInterest();
            } else {
                System.out.println("Option not available for current accounts.");
            }
            break;
        case 4:
            System.out.print("Enter amount to withdraw: ");
            double withdrawAmount = scanner.nextDouble();
            if (account instanceof SavAcct) {
                ((SavAcct) account).withdraw(withdrawAmount);
            } else {
                ((CurAcct) account).withdraw(withdrawAmount);
            }
            break;
        case 5:
            System.out.println("Thank you for using the bank system.");
            scanner.close();
    }
}

```

```

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

```

## **output:**

```

D:\MyPack>java Bank
Enter Customer Name:
Naga
Enter Account Number:
1234
Enter Initial Deposit:
100
Enter Account Type (1 for Savings, 2 for Current):
1
Enter Interest Rate for Savings Account:
10

Banking Menu:
1. Deposit
2. Display Balance
3. Compute and Deposit Interest
4. Withdraw
5. Exit
Select an option: 1
Enter amount to deposit: 400
Deposit Successful. Updated Balance: 500.0

Banking Menu:
1. Deposit
2. Display Balance
3. Compute and Deposit Interest
4. Withdraw
5. Exit
Select an option: 3
Interest Deposited: 50.0, Updated Balance: 550.0

Banking Menu:
1. Deposit
2. Display Balance
3. Compute and Deposit Interest
4. Withdraw
5. Exit
Select an option: 4
Enter amount to withdraw: 200
Withdrawal Successful. Updated Balance: 350.0

```

```
Banking Menu:  
1. Deposit  
2. Display Balance  
3. Compute and Deposit Interest  
4. Withdraw  
5. Exit  
Select an option: 5  
Thank you for using the bank system.  
  
D:\MyPack>java Bank  
Enter Customer Name:  
Naga  
Enter Account Number:  
12345  
Enter Initial Deposit:  
100  
Enter Account Type (1 for Savings, 2 for Current):  
2  
Error: Initial deposit must be at least 500.0  
  
Banking Menu:  
1. Deposit  
2. Display Balance  
4. Withdraw  
5. Exit  
Select an option: 2  
Balance: 0.0  
  
Banking Menu:  
1. Deposit  
2. Display Balance  
4. Withdraw  
5. Exit  
Select an option: 1  
Enter amount to deposit: 400  
Deposit Successful. Updated Balance: 400.0
```

```
Banking Menu:  
1. Deposit  
2. Display Balance  
4. Withdraw  
5. Exit  
Select an option: 4  
Enter amount to withdraw: 200  
Warning: Withdrawing this amount will bring your balance below the minimum. A service charge of 50.0 will be applied.  
Balance fell below minimum. Service charge imposed. Updated Balance: 150.0  
Withdrawal Successful. Updated Balance: 150.0  
  
Banking Menu:  
1. Deposit  
2. Display Balance  
4. Withdraw  
5. Exit  
Select an option: 2  
Balance: 150.0  
  
Banking Menu:  
1. Deposit  
2. Display Balance  
4. Withdraw  
5. Exit  
Select an option: 5  
Thank you for using the bank system.  
  
D:\MyPack>
```

## **Program 6**

Create a package CIE which has two classes- Student and Internals. The class Student has members like usn, name, sem. The class Internals derived from Student has an array that stores the internal marks scored in five courses of the current semester of the student. Create another package SEE which has the class External which is a derived class of Student. This class has an array that stores the SEE marks scored in five courses of the current semester of the student. Import the two packages in a file that declares the final marks of n students in all five courses.

### **Algorithm:**

Week - 06

11/11/20  
2:00PM

Pgm → 06 ↗

Create a package "CIE" which has  
two classes - Student & Internals.  
The class Student has ~~an~~ marks  
like USN, Name, etc. The class Internals  
derived from the Student has an  
array that stores the Internal Marks  
scored in five courses of the current  
semester of the student. Create another  
package "SEE" which has the class  
Internals which is a derived class of student.  
This class has an array that stores the  
SEE Marks scored in five courses of the  
student current semester of the student.  
Import the two packages in a file  
that declares the final marks of "n"  
students in all five courses.

→ ① Student.java ↗

package CIE;  
import java.util.Scanner;

public class Student {

protected String USN = new String();

protected String Name = new String();

protected int fee = 0;

```
public void inputStudentDetails() {
```

{}

Scanner screen Scanner (System Integration)

System out.println("Enter student ID: ");  
Scanner sc = new Scanner(System.in);  
String id = sc.nextLine();

share  $UN = S \cdot \text{next}(C)$

System.out.println("Enter Student Name:");

Nomex S. roets

System-out.println("Enter Student Scores:");

~~Scanz. f. ~~not~~ next to C.~~ m. small  
dark brown. mottled with ~~white~~

~~Wally~~ with Head Lining 5332

public word display student Details CS 2019

{ Zebra - dark seems no way than a  
zebra - 10° off G. 100% think

System.out.println("OSN of shell 2nd  
root -> the deaf");

student?" + USN) 336  
else is not a sufficient justification (because of the

o - ~~different points (name of the  
chart, name of student : " + Name)~~

Systematic prints (see scan of the ~~work~~)

~~student & senior);~~

~~7~~ ~~CD - Readiness~~

2 by the ironies of life, though.  
Hartley

## ③ Internat'l. power

~~Caron's Sop~~ = 120 foot? belt along  
base CSE: 100 ft.

package etc; see front. before going  
soon saw while scanner.

public class External extends Student {

```
public int Mounts [ ] = new int [ ] ;
```

```
public void inputCEMony(CEMony)
```

1

Scanners send scanned (System, info);

Systematic points (Enter the lateral

Mark for the following subjects

out of 80%); no effect

for (int i=0; i<5; i++)

System.out.println("Course " + C(i) +  
" from " + M[i] + " : ");

Mounty [i]  $\geq$  s. neat & nt [j];

$y = \{x + i\sin(\alpha + i\beta) \mid \alpha, \beta \in \mathbb{R}\}$

### ③ Externals of govt

porchage SEE;

import CIE External, tel. (con

import issues until. (comes)

Import: *gossypium*, *cotton*  
Export: *textiles*, *exterior*

public class ExternalMounts {  
 B. ExternalMounts();

protected in eastern Mongolia.

protected vaginal mounts [ ]

```

    } public void calculateFinalMark() {
        finalMark = new int[5];
        finalMark[0] = internalMark[0];
        finalMark[1] = internalMark[1];
        finalMark[2] = internalMark[2];
        finalMark[3] = internalMark[3];
        finalMark[4] = internalMark[4];
    }

    } public void printInput() {
        System.out.println("Scanner System");
        System.out.println("External Mount for course 100%:");
        for (int i=0; i<5; i++) {
            System.out.println("Enter marks: " + i);
            course[i] = scanner.nextInt();
            externalMark[i] = scanner.nextInt();
        }
    }

    } public void calculateFinalMark() {
        for (int i=0; i<5; i++) {
            finalMark[i] = marks[i] + external
            finalMark[i] = marks[i]/2;
        }
    }

    } public void displayFinalMark() {
        System.out.println("Final Marks");
        System.out.println("Final Marks : ");
    }
}

```

for (int i=0; i<20; i++) { (i) student[i]

System.out.println(" Course " + student[i] +  
" Marks : " + finalMarks[i]);  
} (i) System.out.println(" Total Marks : " + totalMarks);

totalMarks = 0; for (int i=0; i<20; i++) {  
totalMarks = totalMarks + marks[i];  
} for (int i=0; i<20; i++) {  
marks[i] = marks[i] / 100.0; }  
}

② Main.java :

import java.util.Scanner;  
import java.util.\*;

public class Main { (i) student[i]

public static void main (String args[]) {  
Scanner scanner = new Scanner (System.in);  
System.out.println ("Enter the number of students : ");

int n = scanner.nextInt();  
External[] students = new External[n];  
External[] student = new External[1];

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

student[i] = new External();

student[i].inputStudentDetails();

student[i].inputSEEMarks();

student[i].inputSEEMarks();

students[i]. calculateFinalMark(); } of  
 + (int) + 2 marks . attsing. two methods  
 for Cint 1=0; i < 5; i++) {  
 students[i]. displayFinalMark();  
 } } usn → javaic - d . CIE / st.javaic CIE / Ext.javaic  
 SEE / Ext.javaic Main.javaic  
 } } usn → java Main  
 ↓  
Output: → java Main  
 Enter Number of Students: 3  
 1  
 2  
 Enter Student USN: 1001 1002 1003  
 1001 1002 1003  
 Enter Student Name:  
 Neeraj (markspl) Anna (markspl) Anuradha (markspl)  
 Enter Student Score: attsing. two methods  
 3 (student part)  
 Enter the Internal Marks for the  
 following subjects out of 10:  
 Course 1: 23  
 Course 2: 25 (CIE (markspl) of  
 25) attsing. two methods  
 Course 3: 25 (markspl) of 25  
 (25) attsing. two methods  
 Course 4: 25 (markspl) of 25

Course 1: 70. Enter marks out of 100

20

Enter External Marks for 8 courses for 100:

Enter Marks for Course 1: 88

Enter Marks for Course 2: 89

Enter Marks for Course 3: 90

Enter Marks for Course 4: 89

Enter Marks for Course 5: 88

Enter Student USN:

420

Enter student Name:

Moshuk

Enter student Sem:

3

Enter the External Marks for the following subjects out of 100. for each course 1:

35

Course 1:

40

Course 2:

50

Course 3:

45

Course 4:

33

Enter External Marks for 8 courses for 100:

Enter Marks for Course 1: 80

Enter Marks for course 2: 89

Enter Marks for Course 3: 78

Enter Marks for Course 4: 67

Enter Marks for Course 5: 79

USN of the Student: 426

Name of the Student: Nagas

Sec of the Student: 3

Final Marks:

Course 1: 56

Course 2: 60

Course 3: 47

Course 4: 46

Course 5: 52

USN of the Student: 420

Name of the Student: Meesha

Sec of the Student: 3

Final Marks:

Course 1: 75

Course 2: 84

Course 3: 89

Course 4: 78

Course 5: 72

10  
10  
10  
10  
10

10  
10  
10  
10  
10

10  
10  
10  
10  
10

10  
10  
10  
10  
10

Draw 2 soft from honest what

:00 not

08 if draw not present at 7

## **Code:**

### Student.java

```
package CIE;
import java.util.Scanner;

public class Student{
protected String usn = new String();
protected String name = new String();
protected int sem;

public void inputStudentDetails()
{
Scanner s=new Scanner(System.in);
System.out.println("Enter Student USN :");
usn=s.next();
System.out.println("Enter Student Name :");
name=s.next();
System.out.println("Enter Student Semester :");
sem=s.nextInt();
}

public void displayStudentDetails()
{
System.out.println("USN of the Student : "+ usn);
System.out.println("Name of the Student : "+ name);
System.out.println("Semester of the Student : "+ sem);
}:
}:
```

### Internals.java

```
package CIE;
import java.util.Scanner;

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

public void inputCIEmarks()
{
Scanner s=new Scanner(System.in);
System.out.println("Enter the Internal Marks for the following Subjects out of 50 :");
```

```

for (int i=0;i<5;i++)
{
System.out.println("Course "+(i+1)+":");
marks[i]=s.nextInt();
}

}

```

### Externals.java

```

package SEE;
import CIE.Internals;
import java.util.Scanner;

public class Externals extends Internals {
protected int externalmarks[];
protected int finalMarks[];

public Externals(){
externalmarks=new int[5];
finalMarks = new int[5];
}

public void inputSEEmarks() {
Scanner s = new Scanner(System.in);
System.out.println("Enter External marks for 5 courses for 100: ");
for (int i = 0; i < 5; i++) {
System.out.print("Enter marks for course " + (i + 1) + ": ");
externalmarks[i] = s.nextInt();
}
}

public void calculateFinalMarks(){
for (int i = 0; i < 5; i++) {
finalMarks[i] = marks[i] + externalmarks[i]/2;
}
}

public void displayFinalMarks(){
displayStudentDetails();
System.out.println("Final Marks: ");
}

```

```
        for (int i = 0; i < 5; i++) {
            System.out.println("Course "+(i+1)+":"+finalMarks[i]);
        }
    }
}
```

### main.java

```
import SEE.Externals;
import java.util.Scanner;

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

        for (int i = 0; i < n; i++) {
            students[i] = new Externals();
            students[i].inputStudentDetails();
            students[i].inputCIEmarks();
            students[i].inputSEEmarks();
            students[i].calculateFinalMarks();
        }

        for (int i = 0; i < n; i++) {
            students[i].displayFinalMarks();
        }
    }
}
```

## Output:

```
D:\MyPack>java Main.java
Enter number of students:

2
Enter Student USN :
426
Enter Student Name :
naga
Enter Student Semester :
3
Enter the Internal Marks for the following Subjects out of 50 :
Course 1:
23
Course 2:
25
Course 3:
2
Course 4:
19
Course 5:
20
Enter External marks for 5 courses for 100:
Enter marks for course 1: 67
Enter marks for course 2: 89
Enter marks for course 3: 90
Enter marks for course 4: 54
Enter marks for course 5: 65
```

```
Enter Student USN :
420
Enter Student Name :
mahesh
Enter Student Semester :
3
Enter the Internal Marks for the following Subjects out of 50 :
Course 1:
35
Course 2:
40
Course 3:
50
Course 4:
45
Course 5:
33
Enter External marks for 5 courses for 100:
Enter marks for course 1: 80
Enter marks for course 2: 89
Enter marks for course 3: 78
Enter marks for course 4: 67
Enter marks for course 5: 79
USN of the Student : 426
Name of the Student : naga
Semester of the Student : 3
Final Marks:
Course 1:56
Course 2:69
Course 3:47
Course 4:46
Course 5:52
USN of the Student : 420
Name of the Student : mahesh
Semester of the Student : 3
Final Marks:
Course 1:75
Course 2:84
Course 3:89
Course 4:78
Course 5:72
```

## **Program 7**

Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called “Father” and derived class called “Son” which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge( ) when the input age<0. In Son class, implement a constructor that cases both father and son’s age and throws an exception if son’s age is  $\geq$ father’s age

### **Algorithm:**

Write a program that demonstrates handling of exception in inheritance tree.  
 Create a base class called "Father" & derived class called "son" which extends the base class. In Father class, implement a constructor which takes the age & throws the exception WrongAge when the input age < 0. In son class implement a constructor that copy both father & son age & throws an exception if son's age is  $\geq$  Father's age.

$\rightarrow$  Import java.util.Scanner;

\* class WrongAge extends Exception {  
 public WrongAge() {  
 super("age Erroneous");  
 }  
} } // general words (not added)

public WrongAge(String message) {  
 super(message);  
}

} // (Hibben. i expect)

\* class Father {  
 protected int Age; } // (as of now)  
} } // (Reference was added)

protected int Age; } // (as of now)

public Father() throws WrongAge {  
 Scanner scnew = new Scanner(System.in);  
 System.out.println("Enter Father's  
 Age: ");  
 Age = scnew.nextInt();  
 if (Age < 0) {  
 throw new WrongAge("Age Erroneous");  
 }
}

Page 2 & next date);  
 if ( $Page < 0$ ) {  
 shows new wrong age ("Age cannot be  
 negative!");  
 }  
 if ( $Page > 2$ ) {  
 shows new wrong age ("Age cannot be  
 greater than or equal  
 to Father's Age!");

180-4336

```

y     00011
y     00011 class Father implements Comparable {
y     public void display() { System.out.println("Father's Age: " + age);
y           super.display();
y           System.out.println("Son's Age: " + sonAge);
y           System.out.println("Total Age: " + (age + sonAge));
y       }
y   }
y   class Son implements Comparable {
y     public void display() { System.out.println("Father's Age: " + fatherAge);
y           System.out.println("Son's Age: " + sonAge);
y           System.out.println("Total Age: " + (fatherAge + sonAge));
y       }
y   }
y   public class ExceptionDemo {
y     public static void main(String args[]) {
y         try {
y             Father father = new Father();
y             Son son = new Son();
y             father.display();
y             son.display();
y         } catch (WrongAge e) {
y             System.out.println("Exception: " + e.getMessage());
y         }
y     }
y   }
y   class WrongAge extends Exception {
y     public WrongAge() {
y         super("Age cannot be negative");
y     }
y   }
y }
```

Yellow = catching two methods  
Blue = returning blank

### Output

- 1) Enter Father's Age: 30
  - Father's Son's Age: 35
  - Exception: Son's age cannot be greater than or equal to Father's Age.
- 2) Enter Father's Age: 50
  - Father's Son's Age: 20
  - Father's Age: 50
  - Son's Age: 20
  - Total Age: 70

## Code:

```
import java.util.Scanner;

class WrongAge extends Exception {
    public WrongAge() {
        super("Age Error!");
    }
    public WrongAge(String message) {
        super(message);
    }
}

class Father {
    protected int fatherAge;

    public Father() throws WrongAge {
        Scanner s = new Scanner(System.in);
        System.out.print("Enter Father's Age: ");
        fatherAge = s.nextInt();

        if (fatherAge < 0) {
            throw new WrongAge("Age cannot be negative!");
        }
    }

    public void display() {
        System.out.println("Father's Age: " + fatherAge);
    }
}

class Son extends Father {
    private int sonAge;
    public Son() throws WrongAge {
        super();
        Scanner s = new Scanner(System.in);
        System.out.print("Enter Son's Age: ");
        sonAge = s.nextInt();
    }
}
```

```

if (sonAge < 0) {
    throw new WrongAge("Age cannot be negative!");
}
if (sonAge >= fatherAge) {
    throw new WrongAge("Son's age cannot be Greater than or Equal to father's
age!");
}

public void display() {
    super.display();
    System.out.println("Son's Age: " + sonAge);
}
}

public class ExcAgeDemo{
    public static void main(String[] args) {
        try {
            Son son = new Son();
            son.display();
        } catch (WrongAge e) {
            System.out.println("Exception: " + e.getMessage());
        }
    }
}

```

## **Output:**

```
D:\>cd JavaPgm

D:\JavaPgm>javac ExcAgeDemo.java

D:\JavaPgm>java ExcAgeDemo
Enter Father's Age: 30
Enter Son's Age: 35
Exception: Son's age cannot be greater than or equal to father's age!

D:\JavaPgm>java ExcAgeDemo
Enter Father's Age: 50
Enter Son's Age: 20
Father's Age: 50
Son's Age: 20

D:\JavaPgm>_
```

## **Program 8**

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

**Algorithm:**

Week-08

18/11/24

2:00 PM

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

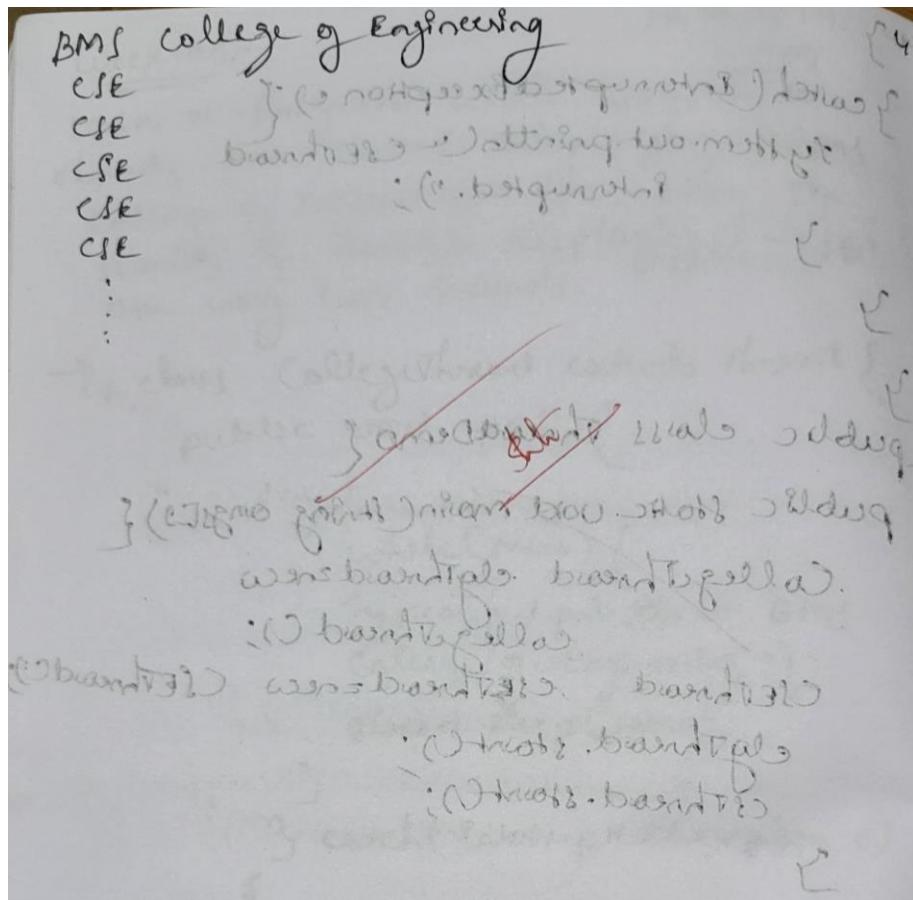
\* class CollegeThread extends Thread {  
 public void run() {  
 try {  
 while (true) {  
 System.out.println("BMS  
College of Engineering");  
 Thread.sleep(10000);  
 }  
 } catch (InterruptedException e) {  
 System.out.println("College  
Thread interrupted.");  
 }  
 }  
}

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

```
    } catch (InterruptedException e) {
        System.out.println("Exception occurred  
Interrupted.");
    }
}

public class ThreadDemo {
    public static void main(String args[]) {
        CollegeThread c1 = new CollegeThread();
        c1.start();
        CSEThread c2 = new CSEThread();
        c2.start();
    }
}

output:
CSE
BMS College of Engineering
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
```



## Code:

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

```
class CSEThread extends Thread {  
    public void run() {  
        try {  
            while (true) {  
                System.out.println("CSE");  
            }  
        }  
    }  
}
```

```
        Thread.sleep(2000);
    }
} catch (InterruptedException e) {
    System.out.println("CSEThread interrupted.");
}
}

public class ThreadDemo{
    public static void main(String[] args) {
        CollegeThread collegeThread = new CollegeThread();
        CSEThread cseThread = new CSEThread();
        collegeThread.start();
        cseThread.start();
    }
}
```

### Output:

```
D:\JavaPgm>javac ThreadDemo.java

D:\JavaPgm>java ThreadDemo
CSE
BMSCE
CSE
CSE
CSE
CSE
BMSCE
CSE
CSE
CSE
CSE
CSE
BMSCE
CSE
CSE
CSE
CSE
CSE
```

## **Program 9**

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

### **Algorithm:**

Week - 09:-

27/04/2024

2:00PM

Write a program that creates a User Interface (UI) to perform integer division. The user enters two numbers in the text fields, num1 & num2. The division of num1 by num2 is displayed in the result field when the divide button is clicked. If num1 & num2 were not an Integer, the program would throw a NumberFormatException. If num2 were zero, the program would throw an ArithmeticException. Display the exception in a message dialog box.

```
→ import javax.swing.*;
import javax.awt.event.*;
import javax.swing.JTextField;
import javax.swing.JButton;
public class DivCalculator {
    public static void main(String args[]) {
        JFrame frame = new JFrame("Integer
Division Calculator");
        frame.setSize(400, 300);
        frame.setDefaultCloseOperation(JFrame.
EXIT_ON_CLOSE);
        frame.setLayout(new FlowLayout());
        JLabel label1 = new JLabel("Enter
num1:");
        JTextField num1Field = new JTextField();
        JLabel label2 = new JLabel("Enter
num2:");
        JTextField num2Field = new JTextField();
        frame.add(label1);
        frame.add(num1Field);
        frame.add(label2);
        frame.add(num2Field);
    }
}
```



```

try {
    catch (NumberFormatException ex) {
        JOptionPane.showMessageDialog(frame,
            "Please enter valid integer.", "Input Error",
            JOptionPane.ERROR_MESSAGE);
    }
    catch (ArithmaticException ex) {
        JOptionPane.showMessageDialog(frame,
            "Cannot divide by zero.", "Arithmatic Error",
            JOptionPane.ERROR_MESSAGE);
    }
}

; (at this point
frame.setVisible(true);

Output (at this point)
    case 1: Enter Num1: 
    Enter Num2: 
    Divide
    ; (at "for") after two inputs
    Result: 
    case 2: Enter Num1: 
    Enter Num2: 
    Divide
    ; (at "for")
    Result: 
    10% 50% report: 
    50% Cannot divide
    by zero
    10% Please enter
    valid category
}

```

## Code:

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

public class DivCalculator {

    public static void main(String[] args) {
        JFrame frame = new JFrame("Integer Division Calculator");
        frame.setSize(400, 300);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLayout(new FlowLayout());

        JLabel label1 = new JLabel("Enter Num1:");
        JTextField num1Field = new JTextField(15);
        JLabel label2 = new JLabel("Enter Num2:");
        JTextField num2Field = new JTextField(15);
        JLabel resultLabel = new JLabel("Result:");
        JTextField resultField = new JTextField(15);
        resultField.setEditable(false); // Result field is non-editable
        JButton divideButton = new JButton("Divide");

        frame.add(label1);
        frame.add(num1Field);
        frame.add(label2);
        frame.add(num2Field);
        frame.add(divideButton);
        frame.add(resultLabel);
        frame.add(resultField);

        divideButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                try {
                    String num1Text = num1Field.getText();
                    String num2Text = num2Field.getText();

                    int num1 = Integer.parseInt(num1Text);
                    int num2 = Integer.parseInt(num2Text);

                    int result = num1 / num2;
                }
            }
        });
    }
}
```

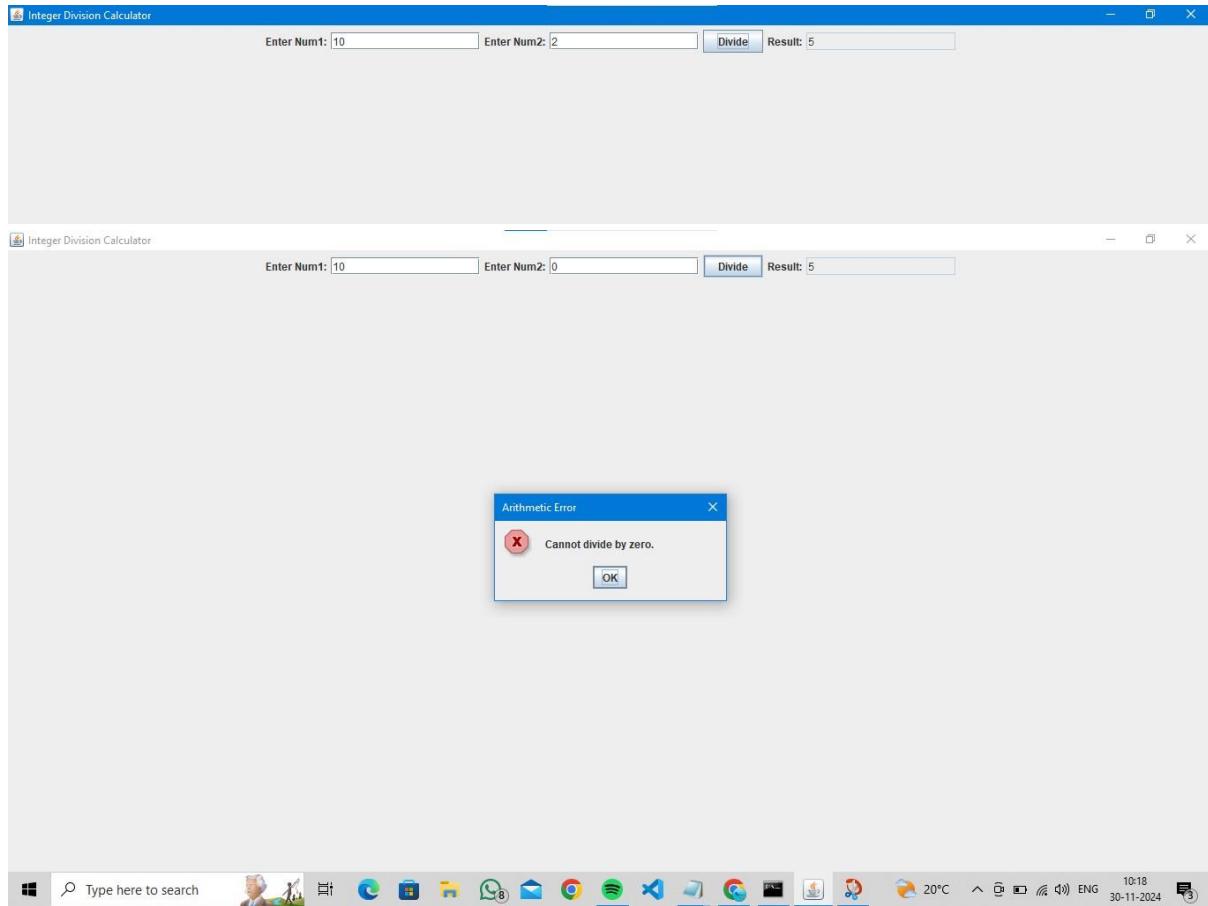
```

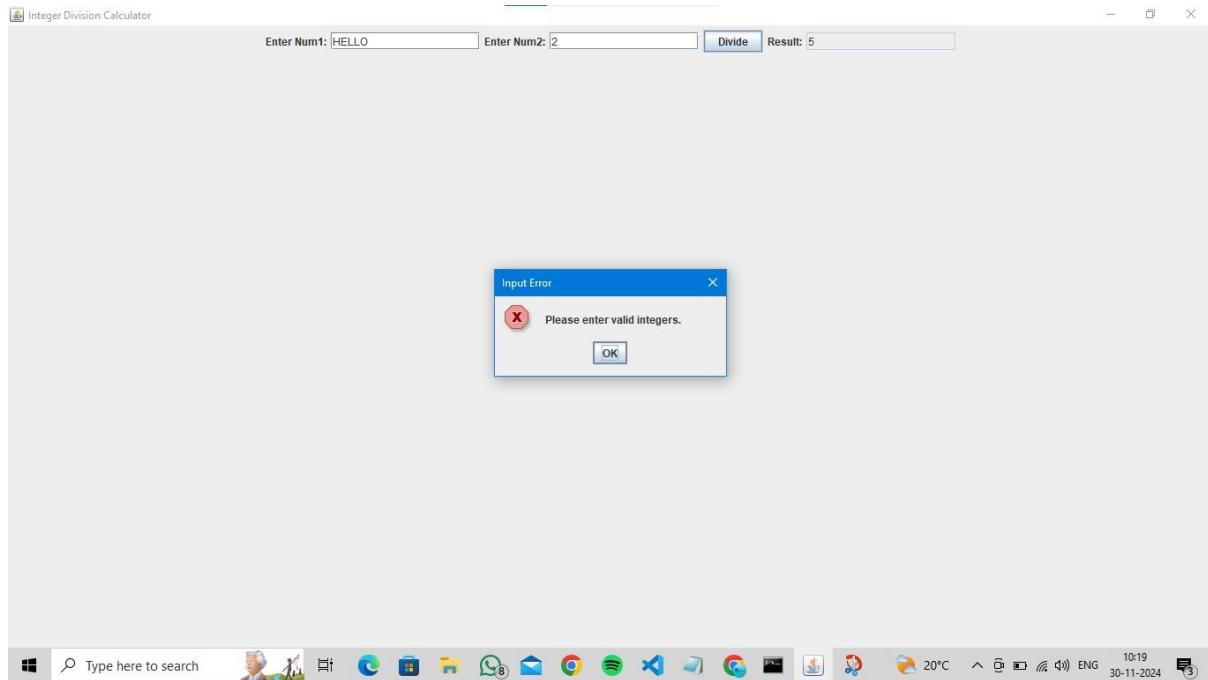
        resultField.setText(String.valueOf(result));
    } catch (NumberFormatException ex) {
        JOptionPane.showMessageDialog(frame, "Please enter valid integers.",
"Input Error", JOptionPane.ERROR_MESSAGE);
    } catch (ArithmaticException ex) {
        JOptionPane.showMessageDialog(frame, "Cannot divide by zero.",
"Arithmatic Error", JOptionPane.ERROR_MESSAGE);
    }
}
}

frame.setVisible(true);
}
}

```

## Output:





## **Program 10**

Open Ended Exercise

**Demonstrate Inter process Communication and deadlock**

**Algorithm:**

21/2/2014  
2:00PM

Week - 10: Open Ended Exercise.

Demonstrating Java Process Communication & Deadlock.

```

    int i;
    boolean valueset = false;
    synchronized int get() {
        while (!valueset)
            try {
                System.out.println("In Consumer waiting for");
                wait();
            } catch (InterruptedException e) {
                System.out.println("InterruptedException caught");
            }
        System.out.println("Value is " + i);
        return i;
    }
    void set(int n) {
        valueset = true;
        notify();
    }
}
    
```

System.out.println("In Producer");  
 valueset = true;  
 System.out.println("Producer is " + n);  
 notify();  
 return n;

```

    synchronized void put(int n) {
        while (valueset)
            try {
                System.out.println("In Producer, waiting");
                wait();
            } catch (InterruptedException e) {
                System.out.println("Interrupted Exception caught");
            }
        this.n = n;
        valueset = true;
        System.out.println("Put " + n);
        System.out.println("In estimate consumer " + n);
        notify();
    }

    class Producer implements Runnable {
        Queue q;
        Producer(Q q) {
            this.q = q;
            new Thread(this, "Producer").start();
        }
    }
}

```

```

public void run() {
    int i = 0;
    while (i < 15) {
        q.put(i);
        i++;
    }
}

* class Consumer implements Runnable {
    Queue<Integer> q;

    Consumer(Queue<Integer> q) {
        this.q = q;
    }

    public void run() {
        int i = 0;
        while (i < 15) {
            int n = q.get();
            System.out.println("Consumed " + n);
            i++;
        }
    }
}

```



notify consumer } 10:30 29/3/2013  
 producer waiting } you shot 10 deg  
 (not 1) }  
 notify producer } 10:30 29/3/2013  
 consumer: 2 }  
 Put: 3 ... etc } 10:30 29/3/2013  
 1) Deadlock Pgm: } 10:30 29/3/2013  
 → class A {  
     synchronized void foo(B b) {  
         String name = Thread.currentThread().  
             getName();  
         System.out.println("Name & entered A: " +  
             name);  
         try {  
             Thread.sleep(1000);  
             } catch (Exception e) {  
                 System.out.println("Caught exception");  
             }  
         System.out.println("Name & trying to  
             call B.last()");  
         b.last();  
     }  
 }  
 void last() {  
     System.out.println("Inside A.last()");  
     }

\* class B {

    synchronized void boom(A a) {  
        String name = Thread.currentThread().getName();  
        // bank has no goal to print  
        // get name  
        // Boom system.out.println("Name entered  
        // B. boom");  
    }

    try {

        // open bank account  
        Thread.sleep(1000);  
    } catch (Exception e) {

        System.out.println("B interrupted");  
    }

    System.out.println("Name trying to  
    call A.lost()");  
    or.lost();

    void lost() {

        System.out.println("Inside A.lost()");

    }

    and B tries bank帐户

    and C tries bank帐户

\* class Deadlock implements Runnable {

    A or new A();

    B or new B();

    Deadlock() {

        Thread.currentThread().setname("Main  
        thread");

Thread race Thread C(this, "Racing at 52.02")

? (A) and B is thread "B"

a. start();  
b. (B) and B is thread "B".  
c. foo(b); // get lock on A in this thread.

System.out.println("Back in Main thread");

Y and B

public void race() {

B.bars(); // get lock on B in other

} (C) Thread C bars {

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

Y

public static void main(String args[]) {

new Deadlock();

Y

} catch (Exception e) {

System.out.println("Exception caught: " + e);

Output:

Main Thread entered A.foo

Racing Thread entered B.bars

Main Thread trying to call B.last()

Inside A.last

Back in Main thread

Racing Thread trying to call A.last()

Inside A.last

Back in other thread

10  
10

JK  
2/2/34

**Code 10A:**

```
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("\nNotify 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("\nNotify Consumer\n");  
        notify();  
    }  
}
```

```

        }
    }

class Producer implements Runnable {
    Q q;

    Producer(Q q) {
        this.q = q;
        new Thread(this, "Producer").start();
    }

    public void run() {
        int i = 0;
        while (i < 15) {
            q.put(i++);
        }
    }
}

class Consumer implements Runnable {
    Q q;

    Consumer(Q q) {
        this.q = q;
        new Thread(this, "Consumer").start();
    }

    public void run() {
        int i = 0;
        while (i < 15) {
            int r = q.get();
            System.out.println("Consumed: " + r);
            i++;
        }
    }
}

```

```
}

class PCFixed {
    public static void main(String args[]) {
        Q q = new Q();
        new Producer(q);
        new Consumer(q);
        System.out.println("Press Control-C to stop.");
    }
}
```

## Output:

```
cmd Select Command Prompt
D:\JavaPgm>java PCFixed
Press Control-C to stop.
Put: 0

Notify Consumer

Producer waiting

Got: 0

Notify Producer

Put: 1
Consumed: 0

Notify Consumer

Producer waiting

Got: 1

Notify Producer

Consumed: 1
Put: 2

Notify Consumer

Producer waiting

Got: 2

Notify Producer

Consumed: 2
Put: 3

Notify Consumer
```

```
Producer waiting
```

```
Got: 3
```

```
Notify Producer
```

```
Consumed: 3
```

```
Put: 4
```

```
Notify Consumer
```

```
Got: 4
```

```
Notify Producer
```

```
Consumed: 4
```

```
Put: 5
```

```
Notify Consumer
```

```
Producer waiting
```

```
Got: 5
```

```
Notify Producer
```

```
Put: 6
```

```
Notify Consumer
```

```
Producer waiting
```

```
Consumed: 5
```

```
Got: 6
```

```
Notify Producer
```

```
Put: 7
```

```
[ca] Select Command Prompt
```

```
Notify Producer
```

```
Put: 7
```

```
Notify Consumer
```

```
Producer waiting
```

```
Consumed: 6
```

```
Got: 7
```

```
Notify Producer
```

```
Consumed: 7
```

```
Put: 8
```

```
Notify Consumer
```

```
Producer waiting
```

```
Got: 8
```

```
Notify Producer
```

```
Consumed: 8
```

```
Put: 9
```

```
Notify Consumer
```

```
Producer waiting
```

```
Got: 9
```

```
Notify Producer
```

```
Consumed: 9
```

```
Put: 10
```

```
Notify Consumer
```

```
Producer waiting
```

```
Got: 10
```

```
Notify Producer
```

```
Consumed: 10
```

```
Put: 11
```

```
Notify Consumer
```

```
Producer waiting
```

```
Got: 11
```

```
Notify Producer
```

```
Consumed: 11
```

```
Put: 12
```

```
Notify Consumer
```

```
Got: 12
```

```
Notify Producer
```

```
Consumed: 12
```

```
Put: 13
```

```
Notify Consumer
```

```
Producer waiting
```

```
Got: 13
```

```
Notify Producer
```

```
Consumed: 13
```

```
Put: 14
```

```
Notify Consumer
```

```
Notify Consumer
```

```
Got: 14
```

```
Notify Producer
```

```
Consumed: 14
```

## Code 10B:

```
class A {  
    synchronized void foo(B b) {  
        String name = Thread.currentThread().getName();  
        System.out.println(name + " enteredA.foo");  
        try {  
            Thread.sleep(1000);  
        } catch(Exception e) {  
            System.out.println("A Interrupted");  
        }  
  
        System.out.println(name + " trying to call B.last()");  
        b.last();  
    }  
    void last() {  
        System.out.println("Inside A.last");  
    }  
}  
class B {  
    synchronized void bar(A a) {  
        String name = Thread.currentThread().getName();  
        System.out.println(name + " entered B.bar");  
        try {  
            Thread.sleep(1000);  
        } catch(Exception e) {  
            System.out.println("B Interrupted");  
        }  
        System.out.println(name + " trying to call A.last()");  
  
        a.last();  
    }  
  
    void last() {  
        System.out.println("Inside A.last");  
    }  
}  
  
class Deadlock implements Runnable  
{  
    A a = new A();
```

```

B b = new B();
Deadlock() {
    Thread.currentThread().setName("MainThread");
    Thread t = new Thread(this, "RacingThread");
    t.start();
    a.foo(b); // get lock on a in this thread.
    System.out.println("Back in main thread");
}

public void run() {
    b.bar(a); // get lock on b in other thread.
    System.out.println("Back in other thread");
}

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

```

### **Output:**

```

D:\JavaPgm>javac Deadlock.java

D:\JavaPgm>java Deadlock
MainThread enteredA.foo
RacingThread entered B.bar
MainThread trying to call B.last()
RacingThread trying to call A.last()
Inside A.last
Inside A.last
Back in other thread
Back in main thread

```