

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum -590014, Karnataka.



LAB REPORT

on

OBJECT ORIENTED JAVA PROGRAMMING

Submitted by

SHASHANK SHANTHARAM NAYAK (1BM23CS313)

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING



B.M.S. COLLEGE OF ENGINEERING

(Autonomous Institution under VTU)

BENGALURU-560019 Sep

2024-Jan 2025

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



CERTIFICATE

This is to certify that the Lab work entitled "**OBJECT ORIENTED JAVA PROGRAMMING**" carried out by **SHASHANK SHANTHARAM NAYAK(1BM23CS313)**, who is bonafide student of **B. M. S. College of Engineering**. It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2024-25. The Lab report has been approved as it satisfies the academic requirements in respect of **Object-Oriented Java Programming Lab - (23CS3PCOOJ)** work prescribed for the said degree.

Dr. Nandhini Vineeth

Associate Professor,
Department of CSE,
BMSCE, Bengaluru

Dr. Kavitha Sooda

Professor and Head,
Department of CSE
BMSCE, Bengaluru

INDEX

Sl. No.	Date	Experiment Title	Page No.
1	26-09-2024	QUADRATIC EQUATION	1
2	03-10-2024	STUDENT	4
3	19-10-2024	BOOK	9
4	24-10-2024	SHAPE – TRIANGLE, RECTANGLE, CIRCLE	13
5	07-11-2024	BANK	18
6	14-11-2024	CIE/SEE PACKAGING	29
7	21-11-2024	WRONGAGE EXCEPTION	37
8	28-11-2024	THREADS	40
9	18-12-2024	DIVISION USING AWT	43
10	18-12-2024	INTERPROCESS COMMUNICATION AND DEADLOCK	47

Include all the 8 programs as instructed in the classroom.

The order to be maintain for every program is

Question

Observation writeup images (complete)

Soft copy of the program

Screenshot of the output

**Editable copies of 9th and 10th program s are attached here.
Analyze and execute and include both in the lab record pdf
giving the same order as above. Explanation for both topics
are included in textbook.**

Include page numbers from this page onwards

LABORATORY PROGRAM - 9

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

```
import  
java.awt.*;  
import  
java.awt.event.*;  
  
public class DivisionMain1 extends Frame implements ActionListener  
{  
    TextField  
    num1,num2;  
    Button dResult;  
    Label  
    outResult;  
    String  
    out="";  
    double  
    resultNum;  
    int flag=0;  
  
public DivisionMain1()  
{  
    setLayout(new FlowLayout());  
  
    dResult = new Button("RESULT");  
    Label number1 = new Label("Number  
1:",Label.RIGHT); Label number2 = new  
    Label("Number      2:",Label.RIGHT);  
    num1=new TextField(5);  
    num2=new TextField(5);
```

```

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

add(number1
 $);$ 
add(num1);
add(number2
 $);$ 
add(num2);
add(dResult)
 $;$ 
add(outResul
t);


---


num1.addActionListener(this);
num2.addActionListener(this);
dResult.addActionListener(this);
addWindowListener(new
WindowAdapter()
{
    public void windowClosing(WindowEvent we)
    {
        System.exit(0);
    }
});
}
public void actionPerformed(ActionEvent ae)
{
    int
    n1,n2;
    try
    {
        if (ae.getSource() == dResult)
        {
            n1=Integer.parseInt(num1.getText());
            n2=Integer.parseInt(num2.getText());
            /*if(n2==0)
             throw new
             ArithmeticException();*/ out=n1+
            "+n2+" ";
        }
    }
}

```

```

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

    }

}

catch(NumberFormatException e1)
{
    flag=1;
    out="Number Format Exception!
"+e1; repaint();
}

catch(ArithmaticException e2)
{
    flag=1;
    out="Divide by 0 Exception!
"+e2; repaint();
}

}

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

```

Demonstrate Interprocess communication and deadlock

```

class Q {
int n;
boolean valueSet = false;

synchronized int get() {
while(!valueSet)
try {
System.out.println("\nConsumer waiting\n");
wait();
} catch(InterruptedException e) {
System.out.println("InterruptedException caught");
}
System.out.println("Got: " + n);
valueSet = false;
System.out.println("\nIntimate Producer\n");
}

```

```

notify();
return n;
}

synchronized void put(int n) {
while(valueSet)
try {
System.out.println("\nProducer waiting\n");
wait();
} catch(InterruptedException e) {
System.out.println("InterruptedException caught");
}
this.n = n;
valueSet = true;
System.out.println("Put: " + n);
System.out.println("\nIntimate Consumer\n");
notify();
}
}

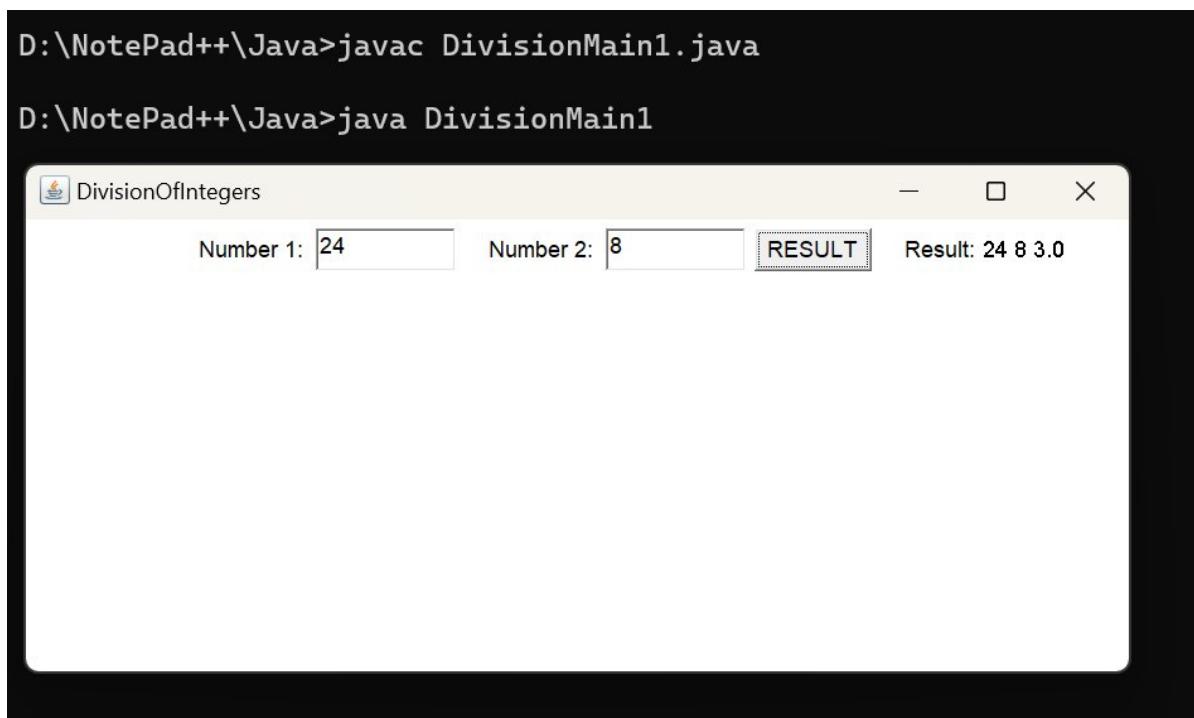
class Producer implements Runnable {
Q q;
Producer(Q q) {
this.q = q;
new Thread(this, "Producer").start();
}
public void run() {
int i = 0;
while(i<15) {
q.put(i++);
}
}
}

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



ii. Demonstration of 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();
        synchronized void last() { System.out.println("Inside A.last"); }
    }

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

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

LABORATORY 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 discriminant $b^2 - 4ac < 0$ is negative, display a message stating that there are no real solutions.

Date: 1/1
Page: 1
7
nur loc

1. Develop a Java program that prints all real solutions to the quadratic equation $ax^2 + bx + c = 0$, using quadratic formula. If determinant $b^2 - 4ac < 0$, print a message stating that there are no real solutions.

```
public class Equation
{
    private double root1, root2;
    private int a, b, c, determinant;
    private boolean real;

    public Equation(int a, int b, int c)
    {
        this.a = a; this.b = b; this.c = c;
        this.determinant = b * b - 4 * a * c;
        this.real = (determinant >= 0);
    }

    public void setRoots()
    {
        if (determinant > 0)
        {
            System.out.println("Unequal Roots");
            root1 = (-b + Math.sqrt(determinant)) / (2 * a);
            root2 = (-b - Math.sqrt(determinant)) / (2 * a);
        }
        else if (determinant == 0)
        {
            System.out.println("Equal Roots");
            root1 = (float) (-b) / (2 * a);
            root2 = root1;
        }
    }
}
```

```

else
    { System.out.println("No Real Roots"); }
}

public double getRoot1() { return root1; }
public double getRoot2() { return root2; }
public double
public boolean isReal() { return real; }

}

class Main {
    public static void main(String args[])
    {
        Equation e = new Equation(Integer.parseInt(args[0]),
                                Integer.parseInt(args[1]),
                                Integer.parseInt(args[2]));
        e.setRoots();
        if(e.isReal())
            System.out.println("Roots are: " + e.getRoot1()
                               + " " + e.getRoot2());
    }
}

→ Output
> java Main 1 -2 1
Equal Roots
Roots are: 1.0 1.0
Output
> java Main 1 3 1
Unequal Roots
Roots are: -0.3819660212502052 -1.618033988749895
> java Main 1 1 1
No Real Roots

```

Code:

```

public class Equation{

    private double root1;
    private double root2;
    private int a;
    private int b;
    private int c;
    private int determinant;
    private boolean real;

    public Equation (int a, int b, int c){
        this.a=a;
        this.b=b;
    }
}

```

```

this.c=c;
this.determinant=b*b-4*a*c;
real=determinant>=0;

}

public void setRoots(){
    if (determinant>0){
        System.out.println("Unequal Roots");
        root1=(-b+Math.sqrt(determinant))/(2*a);
        root2=(-b-Math.sqrt(determinant))/(2*a);
    }

    else if (determinant==0){
        System.out.println("Equal Roots");
        root1=(float)(-b)/(2*a);
        root2=root1;
    }

    else{
        System.out.println("No Real Roots");
    }
}

public double getRoot1(){
    return root1;
}

public double getRoot2(){
    return root2;
}

public boolean isReal(){

```

```

        return real;
    }

}

class Main{
    public static void main(String args[]){
        Equation e=new Equation(Integer.parseInt(args[0]),
Integer.parseInt(args[1]), Integer.parseInt(args[2]));
        e.setRoots();
        if (e.isReal())
            System.out.println("Roots are:"+e.getRoot1()+
"+e.getRoot2());
    }
}

```

OUTPUT

```

Activities Terminal ▾ bmsce@bmsce-OptiPlex-3060:~/1BM23CS313/1BM23CS313-Java/Lab-Prg-1$ java Main 1 -2 1
Equal Roots
Roots are:1.0 1.0
bmsce@bmsce-OptiPlex-3060:~/1BM23CS313/1BM23CS313-Java/Lab-Prg-1$ java Main 1 3 1
Unequal Roots
Roots are:-0.3819660112501051 -2.618033988749895
bmsce@bmsce-OptiPlex-3060:~/1BM23CS313/1BM23CS313-Java/Lab-Prg-1$ java Main 1 1 1
No Real Roots
bmsce@bmsce-OptiPlex-3060:~/1BM23CS313/1BM23CS313-Java/Lab-Prg-1$ |

```

LABORATORY 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.

```
Date _____  
Page _____  
  
grades[i] = 7;  
else if(marks[i] >= 50 && marks[i] < 60)  
    grades[i] = 6;  
else  
    grades[i] = 0;  
}  
}  
  
public void setSGPA(){  
    int creditSum=0, sum=0;  
    for (int i=0; i<marks.length; i++){  
        creditSum += credits[i];  
        sum += grades[i] * credits[i];  
    }  
    this.SGPA = (float) sum / creditSum;  
}  
  
private String makeString(int[] array){  
    String s = new StringBuffer(" ");  
    for (int ele : array)  
        s.append(ele + " ");  
    return s.toString();  
}  
  
public void display()  
{  
    System.out.println("Name:  
    String.format("Name: %s USN: %s\n",  
    Marks: %s Credits: %s SGPA: %.2f",  
    name, USN, makeString(marks),  
    makeString(credits), SGPA));  
}  
}
```

<pre> class Main { public static void main(String args[]) { Scanner sc = new Scanner(System.in); System.out.print("Enter name:"); String name = sc.nextLine(); sc.nextLine(); System.out.print("Enter USN:"); String USN = sc.nextLine().toUpperCase(); System.out.print("Enter no. of subjects:"); int n = sc.nextInt(); int[] marks = new int[n]; int[] credits = new int[n]; System.out.println("Enter the grade and marks of " + n + " subjects:"); for(int i=0; i<n; i++) { credits[i] = sc.nextInt(); marks[i] = sc.nextInt(); } Student s = new Student(name, USN, marks, credits); s.setSGPA(); s.display(); sc.close(); } } </pre>	<p style="text-align: right;">Date _____ Page _____</p> <p>→ Output</p> <p>> java Main Enter name:Shashank Nayak Enter USN: IITM93CS317 Enter no. of subjects: 8</p> <p>Enter the grade and marks of 8 subjects:</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr><td>4</td><td>95</td></tr> <tr><td>4</td><td>90</td></tr> <tr><td>3</td><td>85</td></tr> <tr><td>3</td><td>95</td></tr> <tr><td>3</td><td>90</td></tr> <tr><td>1</td><td>85</td></tr> <tr><td>1</td><td>90</td></tr> <tr><td>1</td><td>98</td></tr> </table> <p>Name:Shashank Nayak USN: IITM93CS317 Marks: 95 90 85 95 90 85 90 98 (Credits: 4 4 3 3 3 2 1 1) SGPA: 9.80</p> <p style="text-align: right;">2/10/2023</p>	4	95	4	90	3	85	3	95	3	90	1	85	1	90	1	98
4	95																
4	90																
3	85																
3	95																
3	90																
1	85																
1	90																
1	98																

Code:

```

import java.util.Scanner;

public class Student{

    private String name;

    private final String USN;

    private int[] marks, credits, grades;

    private double SGPA;

    public Student(String name, String USN, int[] marks, int[]
    credits){

        this.name=name;
        this.USN=USN;
        this.marks=marks;
        this.credits=credits;
        this.grades=new int[marks.length];
    }
}

```

```

public void setGrades(){

    for(int i=0;i<marks.length;i++){

        if(marks[i]>=90 && marks[i]<=100){

            grades[i]=10;

        }

        else if(marks[i]>=80 && marks[i]<90){

            grades[i]=9;

        }

        else if(marks[i]>=70 && marks[i]<80){

            grades[i]=8;

        }

        else if(marks[i]>=60 && marks[i]<70){

            grades[i]=7;

        }

        else if(marks[i]>=50 && marks[i]<60){

            grades[i]=6;

        }

        else

            grades[i]=0;

    }

}

public void setSGPA(){

    int creditSum=0;

    int sum=0;

    for(int i=0;i<marks.length;i++){

        creditSum+=credits[i];

        sum+=grades[i]*credits[i];

    }

```

```

        this.SGPA=(float)sum/creditSum;
    }

private String makeString(int[] array){
    StringBuffer s=new StringBuffer("");
    for(int ele: array)
        s.append(ele+"\t");
    return s.toString();
}

public void display(){
    System.out.println(
String.f
ormat("Name: %s\nUSN:
%s\nMarks: %s\nCredits: %
s\nSGPA: %.2f",name, USN, makeString(marks),
makeString(credits), SGPA));
}

class Main{
    public static void main(String args[]){
        Scanner sc=new Scanner(System.in);
        System.out.print("Enter name:");
        String name=sc.nextLine();
        System.out.print("Enter USN:");
        String USN=sc.next().toUpperCase();
        System.out.print("Enter no. of subjects:");
        int n=sc.nextInt();
        int[] marks=new int[n];
        int[] credits=new int[n];
        System.out.println("\nEnter the grade and marks of "+n+
subjects:");
}

```

```

for(int i=0;i<n;i++){
    credits[i]=sc.nextInt();
    marks[i]=sc.nextInt();
}
//Create Student
Student s=new Student(name, USN, marks, credits);
s.setGrades();
s.setSGPA();
s.display();
sc.close();
}
}

```

OUTPUT

```

Activities Terminal bmsce@bmsce-OptiPlex-3060:~/1BM23CS313/1BM23CS313-Java/Lab-Prg-2$ java Main
Enter name:Shashank Nayak
Enter USN:1bm23cs313
Enter no. of subjects:8
Enter the grade and marks of 8 subjects:
4 95
4 90
3 85
3 95
3 90
1 85
1 90
1 98
Name: Shashank Nayak
USN: 1BM23CS313
Marks: 95      90      85      95      90      85      90      98
Credits: 4       4       3       3       3       1       1       1
SGPA: 9.80
bmsce@bmsce-OptiPlex-3060:~/1BM23CS313/1BM23CS313-Java/Lab-Prg-2$ |

```

LABORATORY 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.

③ Create a class Book of members: name, author, price, num_pages. Include a constructor, setters and getters. Also include a `toString()` method to display details of the book. This must be for n book objects.

```
import java.util.Scanner;
public class Book {
    private String name, author;
    private double price;
    private int num_pages;
    public Book(String name, String author, double price, int num_pages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.num_pages = num_pages;
    }
    public void setName(String newVar) {
        name = newVar;
    }
    public String getName() {
        return name;
    }
    public void setAuthor(String newVar) {
        author = newVar;
    }
    public String getAuthor() {
        return author;
    }
    public void setPrice(double newVar) {
        price = newVar;
    }
    public double getPrice() {
        return price;
    }
    public void setNum_pages(int newVar) {
        num_pages = newVar;
    }
    public int getNum_pages() {
        return num_pages;
    }
    @Override
    public String toString() {
        return String.format("%-25s %-.2f %d", this.name, this.author, this.price, this.num_pages);
    }
}
```

```

class Main {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter no. of books:");
        int n = sc.nextInt();
        sc.nextLine();
        Book[] books = new Book[n];
        for(int i=0; i< books.length; i++) {
            System.out.println("Enter book " + (i+1) + " details:");
            String name = sc.nextLine();
            String author = sc.nextLine();
            double price = sc.nextDouble();
            int num_pages = sc.nextInt();
            sc.nextLine();
            books[i] = new Book(name, author, price,
                num_pages);
        }
        System.out.println("ALL Books Listed Below:");
        for(Book book: books) {
            System.out.println(book);
        }
        sc.close();
    }
}

```

Date _____
Page _____

→ Output
java Main
Enter no. of books:
2
Enter book 1 details:
Sherlock Holmes
Sir Arthur Conan Doyle
829.99
521
Enter book 2. details:
Orient Express
Agatha Christie
1029.99
681
ALL Books Listed Below:
Sherlock Holmes Sir Arthur Conan Doyle 829.99 521
Orient Express Agatha Christie 1029.99 681

26/10/24

Code:

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

```
public class Book {
```

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

```
    public Book (String name, String author, double price, int
num_pages) {
```

```
        this.name=name;
        this.author=author;
        this.price=price;
        this.num_pages=num_pages;
    }
```

```
    public void setName (String newVar) {
        name = newVar;
    }
```

```
    public String getName () {
        return name;
```

```

}

public void setAuthor (String newVar) {
author = newVar;
}

public String getAuthor () {
return author;
}

public void setPrice (double newVar) {
price = newVar;
}

public double getPrice () {
return price;
}

public void setNum_pages (int newVar) {
num_pages = newVar;
}

public int getNum_pages () {
return num_pages;
}

public String toString()
{
    return(String.format("%-25s\t%-25s\t%-8.2f\t%-.5d",this.name,this.author,this.price,this.num_pages));
}
}

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

        System.out.print("Enter no. of books:");
        int n=sc.nextInt();
        sc.nextLine();
        Book[] books=new Book[n];

        for(int i=0;i<books.length;i++){
            System.out.println("\nEnter book "+(i+1)+" details:");
            String name=sc.nextLine();

```

```

        String author=sc.nextLine();
        double price=sc.nextDouble();
        int num_pages=sc.nextInt();
        sc.nextLine();

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

    }

    System.out.println("\nALL BOOKS Listed Below:\n");

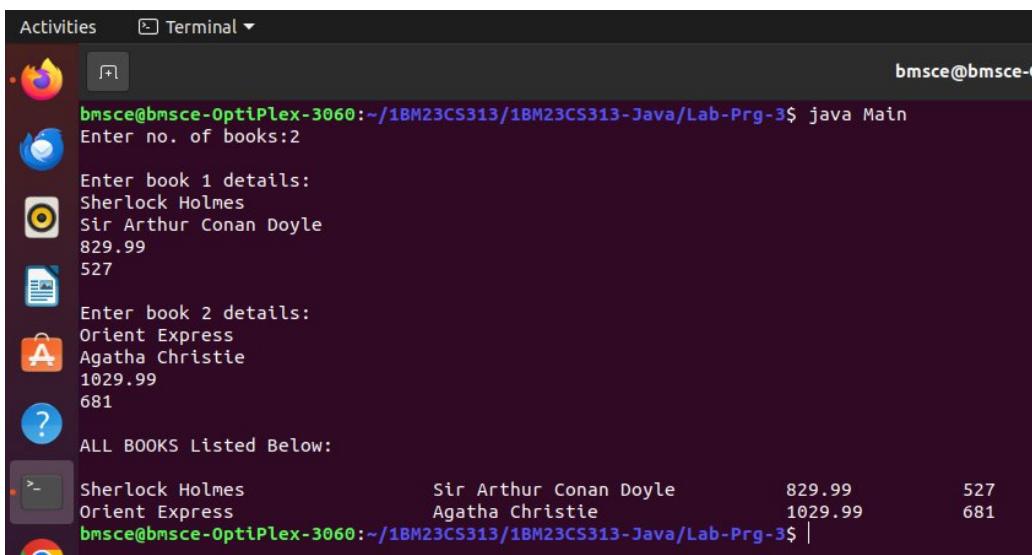
    for(Book book:books){
        System.out.println(book);
    }

    sc.close();

}

```

OUTPUT



```

Activities Terminal bmsce@bmsce-OptiPlex-3060:~/1BM23CS313/1BM23CS313-Java/Lab-Prg-3$ java Main
Enter no. of books:2
Enter book 1 details:
Sherlock Holmes
Sir Arthur Conan Doyle
829.99
527
Enter book 2 details:
Orient Express
Agatha Christie
1029.99
681
ALL BOOKS Listed Below:

Sherlock Holmes          Sir Arthur Conan Doyle      829.99      527
Orient Express           Agatha Christie         1029.99      681
bmsce@bmsce-OptiPlex-3060:~/1BM23CS313/1BM23CS313-Java/Lab-Prg-3$ |

```

LABORATORY 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.

<p>(4) Develop a Java program to create an abstract class named Shape that contains two integers and an empty method printArea(). Provide 3 classes Rectangle, Triangle and Circle such that they only contain printArea().</p> <pre> import java.util.Scanner; abstract class Shape { protected int xDimension; protected int yDimension; Shape() {} Shape(int x, int y) { xDimension = x; yDimension = y; } abstract public void printArea(); } class Rectangle extends Shape { Rectangle(int x, int y) { super(x, y); } public void printArea() { System.out.println("Area is: " + (xDimension * yDimension)); } } </pre>	<p>Date _____ Page _____</p> <pre> class Circle extends Shape { Circle() {} Circle(int x) { super(x, x); } public void printArea() { System.out.println("Area is: " + (Math.PI * xDimension * yDimension)); } } class Triangle extends Shape { Triangle(int x, int y) { super(x, y); } public void printArea() { System.out.println("Area is: " + (0.5 * xDimension * yDimension)); } } class Main { public static void main(String args[]) { Scanner sc = new Scanner(System.in); System.out.println("1. Rectangle\n2. Triangle\n3. Circle\n0. Exit"); Shape s; int choice; choice = sc.nextInt(); if (choice == 1) { s = new Rectangle(); } else if (choice == 2) { s = new Triangle(); } else if (choice == 3) { s = new Circle(); } else { s = null; } s.printArea(); } } </pre>
---	--

The image shows two pages of handwritten notes. The left page contains Java code for a program that calculates the area of rectangles, triangles, or circles based on user input. The right page shows the execution of this code, displaying the menu, user choices, and calculated areas.

```

loop:
    while(true)
    {
        System.out.print("Enter choice:");
        int choice = sc.nextInt();

        switch(choice)
        {
            case 1: System.out.print("Enter dimensions of rectangle:");
                int x=sc.nextInt();
                int y=sc.nextInt();
                s = new Rectangle(x,y);
                s.printArea();
                break;
            case 2: System.out.print("Enter dimensions of triangle:");
                int a = sc.nextInt();
                int b = sc.nextInt();
                s = new Triangle(a,b);
                s.printArea();
                break;
            case 3: System.out.print("Enter dimensions of circle:");
                int ch = sc.nextInt();
                s = new Circle(ch);
                s.printArea();
                break;
            case 0: break loop;
            default: continue loop;
        }
        sc.nextLine();
    }
    System.out.println("Exiting...");
}

```

Output

> Java Main
1. Rectangle
2. Triangle
3. Circle
0. Exit
Enter choice:1
Enter dimensions of rectangle: & 3
Area is: 6
Enter choice:2
Enter dimensions of triangle: 2 3
Area is: 3.0
Enter choice:3
Enter dimensions of circle:5
Area is: 78.53981633974473
Enter choice:0
Exiting...

Code:

```

import java.util.Scanner;

abstract class Shape
{
    protected int xDimension;
    protected int yDimension;
    Shape(){}
    Shape(int x, int y)
    {
        xDimension=x;
        yDimension=y;
    }
    abstract public void printArea();
}

```

```

class Rectangle extends Shape
{
    Rectangle(int x, int y){super(x,y);}
    public void printArea(){System.out.println("Area
is:"+ (xDimension*yDimension));}
}

class Circle extends Shape
{
    Circle(){}
    Circle(int x){
        super(x,x);
    }
    public void printArea(){System.out.println("Area
is:"+(Math.PI*xDimension*yDimension));}
}

class Triangle extends Shape
{
    Triangle(int x, int y){super(x,y);}
    public void printArea(){System.out.println("Area
is:"+(0.5*xDimension*yDimension));}
}

class Main
{
    public static void main(String args[])
    {
        Scanner sc=new Scanner(System.in);
        System.out.println("1. Rectangle\n2. Triangle\n3. Circle\n0.
Exit");
        Shape s;
        loop:

```

```

while(true)
{
    System.out.print("Enter choice:");
    int choice=sc.nextInt();
    switch(choice)
    {
        case 1:      System.out.print("Enter dimensions
of rectangle:");
        int x=sc.nextInt();
        int y=sc.nextInt();
        s=new Rectangle(x,y);
        s.printArea();
        break;

        case 2: System.out.print("Enter dimensions of
triangle:");
        int a=sc.nextInt();
        int b=sc.nextInt();
        s=new Triangle(a,b);
        s.printArea();
        break;

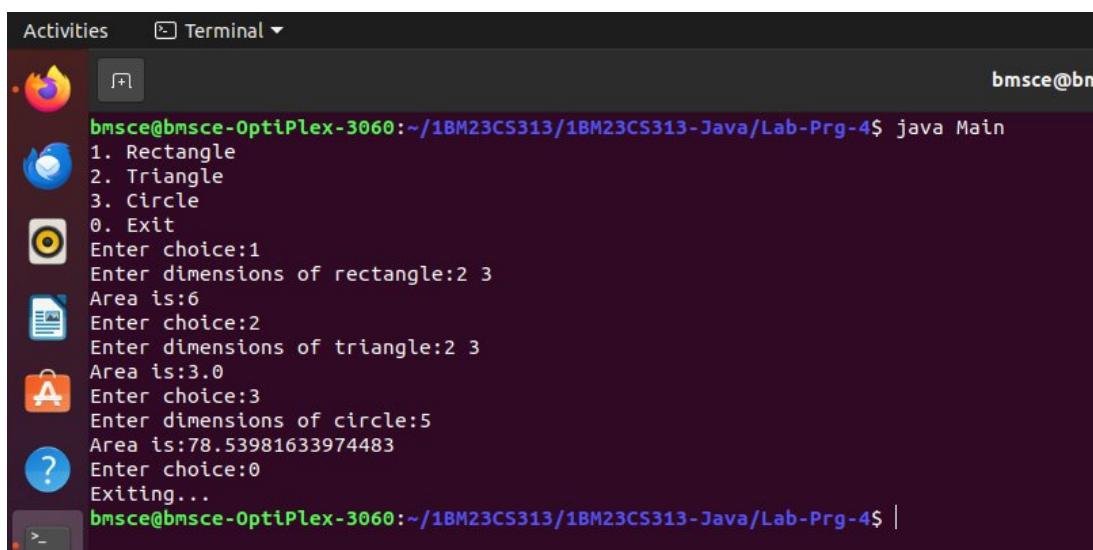
        case 3: System.out.print("Enter dimensions of
circle:");
        int ch=sc.nextInt();
        s=new Circle(ch);
        s.printArea();
        break;

        case 0: break loop;
        default: continue loop;
    }
}

```

```
        sc.nextLine();  
    }  
  
    System.out.println("Exiting...");  
  
    sc.close();  
}  
}
```

OUTPUT



```
Activities Terminal ▾ bmsce@bmsce-OptiPlex-3060:~/1BM23CS313/1BM23CS313-Java/Lab-Prg-4$ java Main  
1. Rectangle  
2. Triangle  
3. Circle  
0. Exit  
Enter choice:1  
Enter dimensions of rectangle:2 3  
Area is:6  
Enter choice:2  
Enter dimensions of triangle:2 3  
Area is:3.0  
Enter choice:3  
Enter dimensions of circle:5  
Area is:78.53981633974483  
Enter choice:0  
Exiting...  
bmsce@bmsce-OptiPlex-3060:~/1BM23CS313/1BM23CS313-Java/Lab-Prg-4$ |
```

LABORATORY 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.

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

<p>(B) WAP to create a class Bank that maintains two kinds of Account for its customers - Savings Account and Current Account.</p> <p>Account -</p> <table border="1" style="margin-left: 20px; border-collapse: collapse;"> <tr> <td>CustomerName</td> <td>Savings Account - Interest - compounds</td> <td>CurrentAccount - Interest - null</td> </tr> <tr> <td>accountNumber</td> <td></td> <td>minimumBalance</td> </tr> <tr> <td>accountType</td> <td></td> <td>serviceCharge</td> </tr> <tr> <td>interest</td> <td></td> <td>chequeBook</td> </tr> <tr> <td>balance</td> <td></td> <td></td> </tr> </table> <p>Tasks to be achieved</p> <ul style="list-style-type: none"> (a) Accept deposit from Customer and update balance (b) Compute and deposit interest (c) Permit withdrawal (d) Display balance (e) Check for min Balance, impose penalty <pre style="font-family: monospace; margin-top: 10px;"> import java.util.Scanner; public class Account { private static long accountCounter=0; protected String customerName; protected String accountNumber; protected String accountType; protected double balance; } public Account() { public void getData(Scanner sc, String accountType, long accountCount) { System.out.print("Enter customer name:"); customerName = sc.nextLine(); } } </pre>	CustomerName	Savings Account - Interest - compounds	CurrentAccount - Interest - null	accountNumber		minimumBalance	accountType		serviceCharge	interest		chequeBook	balance			<p>accountNumber = (long)accountCounter + "</p> <p>accountType" + accountCount</p> <pre style="font-family: monospace; margin-top: 10px;"> System.out.print("Enter initial Balance:"); balance = sc.nextDouble(); display(); } public void display() { System.out.println("Name: " + customerName + "Acc No: " + accountNumber + "Balance: " + balance); } public void withdrawal(double amount) { if(balance > amount) { balance -= amount; System.out.println("Withdrawn " + amount + " Balance: " + balance); } else System.out.println("Transaction failed"); } public void deposit(double amount) { balance += amount; System.out.println("Deposited " + amount + " Balance: " + balance); } public void addInterest() { } </pre>
CustomerName	Savings Account - Interest - compounds	CurrentAccount - Interest - null														
accountNumber		minimumBalance														
accountType		serviceCharge														
interest		chequeBook														
balance																

```

public String getAccountNumber() {
    return accountNumber;
}

class SavingsAccount extends Account {
    static public static double interestRate = 0.05;
    static long count = 0;

    SavingsAccount() {}

    public void getData() {
        Scanner sc = new Scanner(System.in);
        accountType = "S";
        super.getData(sc, accountType, ++count);
    }

    public void addInterest() {
        balance += balance * interestRate;
    }

    class CurrentAccount extends Account {
        static long minimumBalance = 5000;
        static long count = 0;
        String chequeBook = "M";

        CurrentAccount() {}

        public void getData() {
            Scanner sc = new Scanner(System.in);
            accountType = "C";
        }
    }
}

Super.getData(sc, accountType, ++count);

public void withdrawal(double amount) {
    if (balance - amount < minimumBalance)
        double serviceCharge = amount * 0.1;
    System.out.println("Transaction to go below minimum balance (" + minimumBalance +
        "+"). Service charge of " + serviceCharge +
        " levied for the transaction");
    super.withdrawal(amount + serviceCharge);
    return;
}

super.withdrawal(amount);

public void addCheque(String i) {
    chequeBook += i + "\n";
}

```

```

File: Bank.java
import java.util.Scanner;

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

        SavingsAccount[] sa = new SavingsAccount[10];
        CurrentAccount[] ca = new CurrentAccount[10];
        int sIndex=0, cIndex=0;

        System.out.println("1. Savings \n2. Current \n3. Deposit \n4. Withdraw \n5. Interest \n6. Display \n7. Cheque \n0. Exit");

        int choice;
        do {
            System.out.print("\nEnter choice:");
            choice = sc.nextInt();
            switch(choice) {
                case 1: sa[sIndex] = new SavingsAccount();
                    sa[sIndex+1].getData();
                    break;
                case 2: ca[cIndex] = new CurrentAccount();
                    ca[cIndex+1].getData();
                    break;
            }
        } while(choice != 0);

        case 3: System.out.print("Enter Account Number: ");
        String accNo = sc.nextLine().strip().toUpperCase();
        if(accNo.charAt(0) == 'S') {
            for(int i=0; i<sIndex; i++) {
                if(sa[i].getAccountNumber().equals(
                    accNo)) {
                    System.out.print("Enter amount:");
                    double amount = sc.nextDouble();
                    sa[i].deposit(amount);
                }
            }
        } else if(accNo.charAt(0) == 'C') {
            for(int i=0; i<cIndex; i++) {
                if(ca[i].getAccountNumber().equals(
                    accNo)) {
                    System.out.print("Enter amount:");
                    double amount = sc.nextDouble();
                    ca[i].deposit(amount);
                }
            }
        } else {
            System.out.println("Invalid Account no.");
            break;
        }
    }
}

```

```

Date / / Page / /
case 4: System.out.print("Enter account number: ");
String accnumber = sc.next().strip().toUpperCase();
if(accnumber.charAt(0) == 'S')
{
    for(int i=0; i<=5; index++; i++)
    {
        if(sal[i].getAccountNumber().equals(accnumber))
        {
            System.out.print("Enter amount:");
            double amount = sc.nextDouble();
            sal[i].withdrawal(amount);
        }
    }
}
else if(accnumber.charAt(0) == 'C')
{
    for(int i=0; i<=5; index++; i++)
    {
        if(ca[i].getAccountNumber().equals(accnumber))
        {
            System.out.print("Enter amount:");
            double amount = sc.nextDouble();
            ca[i].withdrawal(amount);
        }
    }
}
else
{
    System.out.println("Invalid account no.");
}
break;
}

case 5: for(Savings Account s:s)
{
    if(s != null)
    {
        s.addInterest();
    }
}
System.out.println("Added Interest");
break;

case 6:
for(Savings Account a:s)
{
    if(a != null)
    {
        a.display();
    }
}
for(Current Account b:c)
{
    if(b != null)
    {
        b.display();
    }
}
break;

case 7: System.out.print("Enter Account No:");
String accNo = sc.next().strip().toUpperCase();
if(accNo.charAt(0) == 'C')
{
    System.out.print("Enter Cheque No:");
    System.out.print("Enter Cheque No:");
    String checkNumber = sc.next();
    checkNumber.strip().toUpperCase();
    for(Current Account c:c)
    {
        if(c != null && c.getAccountNumber()
        .equals(accNo))
        {
            c.addCheque(checkNumber);
        }
    }
}
break;
}

```

Outputs
1. S
2. C
3. D
4. W
5. I
6. Di
7. Cr
0. Ex
Enter
Enter
Enter
Name
Acc N
Balance
Enter
Enter
Enter

```

Date / / Page / /
Name: John
Acc No: 2C1
Balance: 9870.15

Enter choice: 3
Enter Account number: 151
Enter amount: 500
Deposited 500.0, Balance: 5300.0

Enter choice: 4
Enter Account number: 2C1
Enter amount: 500
Transaction to go below minimum balance(5000). Service
charge of 50.0 levied for the transaction
Withdrawn 500.0, Balance: 4870.15

Enter choice: 5
Added Interest

Enter choice: 6
Name: Sean
Acc No: 151
Balance: 5775.0

Name: John
Acc No: 2C1
Balance: 9870.15
On your
    Bal
    will be

Enter choice: 7
Enter Account No: 2C1
Enter Cheque No: 3912
Added Cheque

Enter choice: 0
Exiting...
E

```

```

Date _____ / _____ / _____
Page _____ / _____ / _____


```

class SavingsAccount {
 double balance;
 void addInterest() {
 balance = balance * 0.05;
 }
 void withdraw(double amount) {
 if (amount < 0) {
 System.out.println("Amount cannot be negative");
 } else if (balance < amount) {
 System.out.println("Insufficient funds");
 } else {
 balance -= amount;
 System.out.println("Withdrawal successful");
 }
 }
 void deposit(double amount) {
 if (amount < 0) {
 System.out.println("Amount cannot be negative");
 } else {
 balance += amount;
 System.out.println("Deposit successful");
 }
 }
 void display() {
 System.out.println("Customer Name: " + customerName);
 System.out.println("Account Number: " + accountNumber);
 System.out.println("Balance: " + balance);
 }
}

class CurrentAccount extends SavingsAccount {
 void addCheque(double amount) {
 if (amount < 0) {
 System.out.println("Amount cannot be negative");
 } else {
 balance += amount;
 System.out.println("Cheque added successfully");
 }
 }
}

class Account {
 static int accountCounter = 1;
 String customerName;
 String accountNumber;
 String accountType;
 double balance;

 public Account() {
 }

 public void getData(Scanner sc, String accountType, long accountCount) {
 System.out.print("Enter customer name:");
 customerName = sc.nextLine();
 accountNumber =
 (++accountCounter) + "-" + accountType + "-" + accountCount;
 System.out.print("Enter initial Balance:");
 balance = sc.nextDouble();
 }

 void withdraw(double amount) {
 if (amount < 0) {
 System.out.println("Amount cannot be negative");
 } else if (balance < amount) {
 System.out.println("Insufficient funds");
 } else {
 balance -= amount;
 System.out.println("Withdrawal successful");
 }
 }

 void deposit(double amount) {
 if (amount < 0) {
 System.out.println("Amount cannot be negative");
 } else {
 balance += amount;
 System.out.println("Deposit successful");
 }
 }

 void display() {
 System.out.println("Customer Name: " + customerName);
 System.out.println("Account Number: " + accountNumber);
 System.out.println("Balance: " + balance);
 }

 void addInterest() {
 balance = balance * 0.05;
 }
}

```


```

```

Date _____ / _____ / _____
Page _____ / _____ / _____


```


```

        break;
    } else {
        System.out.println("Invalid Account");
    }
    default:
        System.out.println("Invalid Input");
    }
    sc.nextLine();
} while (choice != 0);
System.out.println("Exiting..."); sc.close();
}

Output:
1. Savings
2. Current
3. Deposit
4. Withdraw
5. Interest
6. Display
7. Cheque
0. Exit

Enter choice: 1
Enter customer name: Sean
Enter initial Balance: 5000
Name: Sean
Acc No: 151
Balance: 5000.0

Enter choice: 2
Enter customer name: John
Enter initial Balance: 5420.15

```


```


```

Code:

```
// Account.java
import java.util.Scanner;
```

```
public class Account
```

```
{
```

```
    private static long accountCounter=0;
    protected String customerName;
    protected String accountNumber;
    protected String accountType;
    protected double balance;
```

```
    public Account(){
    }
```

```
    public void getData(Scanner sc, String accountType, long accountCount)
```

```
{
```

```
        System.out.print("Enter customer name:");
        customerName = sc.nextLine();
```

```
        accountNumber =
                (++accountCounter) + "-" + accountType + "-" + accountCount;
        System.out.print("Enter initial Balance:");
        balance = sc.nextDouble();
```

```

        display();
    }

public void display()
{
    System.out.println(
        "\nName:"+customerName+
        "\nAcc No:"+accountNumber+
        "\nBalance:"+balance);
}

public void withdrawal(double amount)
{
    if(balance-amount>0)
    {
        balance -= amount;
        System.out.println("Withdrawn "+amount+
            ". Balance:"+balance);
    }
    else
    {
        System.out.println("Transaction failed.");
    }
}

public void deposit(double amount)
{
    balance += amount;
    System.out.println("Deposited "+amount+
        ". Balance:"+balance);
}

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

public String getAccountNumber(){return accountNumber;}

public void addInterest(){
}

class SavingsAccount extends Account
{
    static public double interestRate=0.05;
    static long count=0;
}

```

```

SavingsAccount(){
}

public void getData(){
    Scanner sc = new Scanner(System.in);
    accountType = "S";
    super.getData(sc, accountType, ++count);
}

public void addInterest(){
    balance += balance*interestRate;
}
}

class CurrentAccount extends Account
{
    static long minimumBalance = 5000;
    static long count=0;
    String chequeBook = "";
    CurrentAccount(){
    }

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

        accountType = "C";
        super.getData(sc,accountType,++count);
    }

    public void withdrawal(double amount){
        if (balance-amount < minimumBalance)
        {
            double serviceCharge = amount * 0.1;
            System.out.println("Transaction to go below minimum
balance"+
                minimumBalance+""). Service charge of "+
                serviceCharge+" levied for the transaction");

            super.withdrawal(amount+serviceCharge);
            return;
        }
        super.withdrawal(amount);
    }

    public void addCheque(String s)
{
}

```

```

        chequeBook = chequeBook+s+"\n";
    }
}

// Bank.java
import java.util.Scanner;

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

        SavingsAccount[] sa = new SavingsAccount[10];
        CurrentAccount[] ca = new CurrentAccount[10];

        int sIndex=0, cIndex=0;

        System.out.println("1. Savings\n2. Current\n3. Deposit\n4.
Withdraw\n5. Interest\n6. Display\n7. Cheque\n0. Exit");
        int choice;

        do
        {
            System.out.print("\nEnter choice:");
            choice = sc.nextInt();

            switch(choice)
            {
                case 1 :sa[sIndex]=new SavingsAccount();
                           sa[sIndex++].getData();
                           break;
                case 2 :ca[cIndex]=new CurrentAccount();
                           ca[cIndex++].getData();
                           break;
                case 3 :System.out.print("Enter Account number:");
                           String accno = sc.next().strip().toUpperCase();

                           if(accno.charAt(1)=='S')
                           {
                               for(int i=0;i<sIndex;i++)
                               {
                                   if(sa[i].getAccountNumber().equals(accno))
                                   {

```

```

        System.out.print("Enter amount:");
        double amount = sc.nextDouble();
        sa[i].deposit(amount);
    }
}
else if(accno.charAt(1)=='C')
{
    for(int i=0;i<cIndex;i++)
    {

if(ca[i].getAccountNumber().equals(accno))
{
    System.out.print("Enter amount:");
    double amount = sc.nextDouble();
    ca[i].deposit(amount);
}
}
else
{
    System.out.println("Invalid account no.");
}
break;
case 4 :System.out.print("Enter Account number:");
String accnumber = sc.next().strip().toUpperCase();

if(accnumber.charAt(1)=='S')
{
    for(int i=0;i<sIndex;i++)
    {

if(sa[i].getAccountNumber().equals(accnumber))
{
    System.out.print("Enter amount:");
    double amount = sc.nextDouble();
    sa[i].withdrawal(amount);
}
}
else if(accnumber.charAt(1)=='C')
{
    for(int i=0;i<cIndex;i++)

```

```

        {
    if(ca[i].getAccountNumber().equals(accnumber))
    {
        System.out.print("Enter amount:");
        double amount = sc.nextDouble();
        ca[i].withdrawal(amount);
    }
}
else
{
    System.out.println("Invalid account no.");
}
break;
case 5 :for(SavingsAccount s: sa)
{
    if(s!=null)
    {
        s.addInterest();
    }
}
System.out.println("Added Interest");
break;
case 6:
    // Display Savings Accounts
    for (SavingsAccount a : sa) {
        if (a != null) { // Check if the account is initialized
            a.display();
        }
    }
    // Display Current Accounts
    for (CurrentAccount b : ca) {
        if (b != null) { // Check if the account is initialized
            b.display();
        }
    }
}
break;
case 7:   System.out.print("Enter Account No:");
String accNo = sc.next().strip().toUpperCase();
if(accNo.charAt(1)=='C')
{
    System.out.print("Enter Cheque No:");
String checkNumber = sc.next().strip().toUpperCase();

```

```

        for(CurrentAccount c: ca)
        {
            if(c!=null && c.getAccountNumber().equals(accNo))
                c.addCheque(checkNumber);
            }
            System.out.println("Added Cheque");
        }
        else
        {
            System.out.println("Invalid Account");
        }
        break;
    case 0: break;
    default:System.out.println("Invalid Input");

}
sc.nextLine();
}while(choice!=0);

System.out.println("Exiting...");
sc.close();
}
}

```

OUTPUT

```

> javac *.java
> java Bank
1. Savings
2. Current
3. Deposit
4. Withdraw
5. Interest
6. Display
7. Cheque
0. Exit

Enter choice:1
Enter customer name:Sean
Enter initial Balance:5000

Name:Sean
Acc No:151
Balance:5000.0

Enter choice:2
Enter customer name:John
Enter initial Balance:5420.15

Name:John
Acc No:2C1
Balance:5420.15

Enter choice:3
Enter Account number:1s1
Enter amount:500
Deposited 500.0. Balance:5500.0

Enter choice:4
Enter Account number:2C1
Enter amount:500
Transaction to go below minimum balance(5000). Service charge of 50.0 levied for the transaction

```

```
Enter choice:4
Enter Account number:2C1
Enter amount:500
Transaction to go below minimum balance(5000). Service charge of 50.0 levied for the transaction
Withdrawn 500.0. Balance:4870.15

Enter choice:4
Enter Account number:2C1
Enter amount:5000
Transaction to go below minimum balance(5000). Service charge of 500.0 levied for the transaction
Transaction failed.

Enter choice:5
Added Interest

Enter choice:6

Name:Sean
Acc No:131
Balance:5775.0

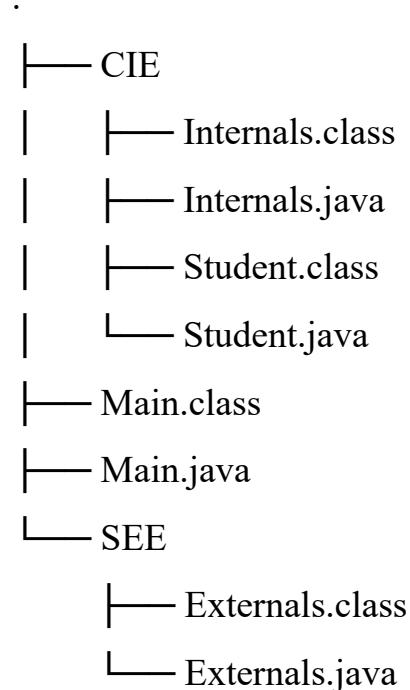
Name:John
Acc No:2C1
Balance:4870.15

Enter choice:7
Enter Account No:2C1
Enter Cheque No:SQ12
Added Cheque

Enter choice:0
Exiting...
```

LABORATORY PROGRAM - 6

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



Create a package CIE of two classes Student & Internals. Student stores user name, semester. Internals contains an array of 5 integers for 5 course marks. In package SEE create a class Externals that implements Student. This is for SEE marks. Import both packages in a file and find the final marks of n students.

```

    package CIE;
    import java.util.Scanner;
    public class Student {
        private String name, usn;
        private int semester;
        public Student() {}
        public Student(String usn, String name, int semester) {
            this.usn = usn; this.name = name; this.semester = semester;
        }
        public String getName() { return name; }
        public String getUSN() { return usn; }
        public int getSemester() { return semester; }
        public Student(Student s) {
            this.name = s.name; this.usn = s.usn; this.semester = s.semester;
        }
    }
  
```

```

    package CIE;
    import java.util.Scanner;
    public class Internals extends Student {
        private int[] marks;
        private Internals(String usn, String name, int[] marks) {
            super(usn, name, marks);
            this.marks = marks;
        }
        public static Internals getInstance() {
            Scanner sc = new Scanner(System.in);
            System.out.println("Enter student details");
            System.out.println("and array of CIE marks");
            String usn = sc.nextLine().toUpperCase();
            String name = sc.nextLine();
            int semester = sc.nextInt();
            int[] marks = new int[5];
            for(int i=0; i<marks.length; i++) {
                marks[i] = sc.nextInt();
            }
            sc.nextLine();
            return new Internals(usn, name, marks);
        }
        public int getMarks(int i) { return marks[i]; }
    }
  
```

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

```

    public class Externals extends Student {
        private int[] marks;
        private Externals(String usn, String name, int semester, int[] marks) {
            super(usn, name, semester);
            this.marks = marks;
        }
        private Externals(Student s, int[] marks) {
            super(s); this.marks = marks;
        }
        public static Externals getInstance(Student s) {
            Scanner sc = new Scanner(System.in);
            System.out.print("Enter SEE details:");
            int[] marks = new int[5];
            for(int i=0; i<marks.length; i++) {
                marks[i] = sc.nextInt();
            }
            sc.nextLine();
            return new Externals(s, marks);
        }
        public int getMarks(int i) { return marks[i]; }
    }
  
```

```

    package SEE;
    import CIE.*;
    import java.util.Scanner;
    class Main {
        public static void main(String args[]) {
            Scanner sc = new Scanner(System.in);
            System.out.print("Enter the number of Students:");
            int n = sc.nextInt();
            Internals[] ai = new Internals[n];
            Externals[] ae = new Externals[n];
            int[] result = new int[5];
            for(int i=0; i<n; i++) {
                ai[i] = Internals.getInstance();
                ae[i] = Externals.getInstance(ai[i]);
            }
            System.out.println("Details");
            for(int i=0; i<n; i++) {
                System.out.println("USN: " + ai[i].getUSN());
                System.out.println("Name: " + ai[i].getName());
                System.out.println("Semester: " + ai[i].getSemester());
            }
            System.out.println("Internals:");
            for(int j=0; j<5; j++) {
                System.out.print(ai[0].getMarks(j) + " ");
                result[j] = ai[0].getMarks(j);
            }
        }
    }
  
```

```

Date _____
Page _____
System.out.println();
System.out.print("Internals:");
for (int j = 0; j < 5; j++) {
    System.out.print(ae[i].getMarks(j) + "\t");
    result[j] += ae[i].getMarks(j) / 2.0;
}
System.out.println();
System.out.print("Results: ");
for (int j = 0; j < 5; j++) {
    System.out.print(result[j] + "\t");
}
System.out.println();
sc.close();
}

> java Main
Enter the number of students: 2
Enter student details and array of CIE marks
1BM83CS313 Shashank 3
50 49 48 47 46
Enter SEE marks details: 92 94 96 98 100
Details:
USN:1BM83CS313 Name:Shashank Semester:3
Internals:50 49 48 47 46
Externals: 92 94 96 98 100
Results: 96 96 96 96 96

```

Code:

```

// CIE/Student.java
package CIE;

import java.util.Scanner;

public class Student
{
    private String name;
    private String usn;
    private int semester;

    public Student(){}
    public Student(String usn, String name, int semester)
    {
        this.usn=usn;
        this.name=name;
        this.semester=semester;
    }
    public Student(Student s)

```

```

{
    this.usn = s.usn;
    this.name = s.name;
    this.semester = s.semester;
}
/**/
public void getData()
{
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter USN:");
    this.usn = sc.next().strip().toUpperCase();
    System.out.print("Enter name:");
    this.name = sc.nextLine();
    System.out.print("Enter semester:");
    this.semester=sc.nextInt();
}
*/
}

public String getName(){return name;}
public String getUSN(){return usn;}
public int getSemester(){return semester;}
}

// CIE/Internals.java
package CIE;

import java.util.Scanner;

public class Internals extends Student
{
    private int[] marks;

    private Internals(String usn, String name, int semester, int marks[]){
        super(usn,name,semester);
        this.marks=marks;
    }

    private Internals(Student s, int[] marks)
    {
        super(s);
        this.marks=marks;
    }

    /**
     * @param marks
     */
    public Internals(int[] marks)

```

```

{
    this.marks=marks;
}
*/
public static Internals getInstanceOf()
{
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter student details and array of CIE
marks");
    String usn = sc.nextLine().toUpperCase();
    String name = sc.nextLine();
    int semester = sc.nextInt();
    int[] marks = new int[5];
    for(int i=0;i<marks.length;i++){
        marks[i]=sc.nextInt();
    }
    sc.nextLine();
    return new Internals(usn,name,semester,marks);
}

public static Internals getInstanceOf(Student s)
{
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter CIE marks details:");
    int marks[] = new int[5];
    for(int i=0;i<marks.length;i++){
        marks[i]=sc.nextInt();
    }
    sc.nextLine();
    return new Internals(s,marks);
}

public int getMarks(int i){return marks[i];}

}

// SEE/Externals.java
package SEE;

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

public class Externals extends Student
{

```

```

private int[] marks;

private Externals(String usn, String name, int semester, int[] marks){
    super(usn,name,semester);
    this.marks=marks;
}

private Externals(Student s, int[] marks)
{
    super(s);
    this.marks=marks;
}
/**
public Externals(int[] marks)
{
    this.marks=marks;
}
**/
public static Externals getNewInstanceOf() {
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter student details and array of SEE
marks");
    String usn = sc.next().strip().toUpperCase();
    String name = sc.next();
    int semester = sc.nextInt();
    int[] marks = new int[5];
    for (int i = 0; i < marks.length; i++) {
        marks[i] = sc.nextInt();
    }
    sc.nextLine();
    return new Externals(usn, name, semester, marks);
}

public static Externals getNewInstanceOf(Student s) {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter SEE marks details:");
    int[] marks = new int[5];
    for (int i = 0; i < marks.length; i++) {
        marks[i] = sc.nextInt();
    }
    sc.nextLine();
    return new Externals(s, marks);
}
public int getMarks(int i){return marks[i];}

```

```

}

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

class Main
{
    public static void main(String args[])
    {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number of Students:");
        int n = sc.nextInt();
        Internals[] ai = new Internals[n];
        Externals[] ae = new Externals[n];
        int[] result=new int[5];

        for (int i = 0; i < n; i++) {
            ai[i] = Internals.newInstance();
            ae[i] = Externals.newInstance(ai[i]);
        }

        System.out.println("Details:");

        for(int i=0;i<n;i++)
        {
            System.out.println("USN:"+ai[i].getUSN()
                +"\tName:"+ai[i].getName()
                +"\tSemester:"+ai[i].getSemester());
            System.out.print("Internals:");
            for(int j=0;j<5;j++){
                System.out.print(ai[i].getMarks(j)+"\t");
                result[j]=ai[i].getMarks(j);
            }
            System.out.println();
            System.out.print("Externals:");
            for(int j=0;j<5;j++){
                System.out.print(ae[i].getMarks(j)+"\t");
                result[j]+=(ae[i].getMarks(j)/2.0);
            }
            System.out.println();
            System.out.print("Results: ");
            for(int j=0;j<5;j++)
                System.out.print(result[j]+"\t");
        }
    }
}

```

```
        System.out.println();
    }
    sc.close();
}
}
```

OUTPUT

```
> java Main
Enter the number of Students:1
Enter student details and array of CIE marks
1BM Kevin 3
40 41 42 43 44
Enter SEE marks details:100 99 98 97 96
Details:
USN:1BM Name:Kevin      Semester:3
Internals:40    41    42    43    44
Externals:100   99    98    97    96
Results:   90    90    91    91    92
```

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

<p>7 Create classes as follows:</p> <pre> class Father { String name; int age; Father(String name, int age) { this.name = name; this.age = age; } public String toString() { return "Father's name: " + name + ", age: " + age; } } class Son extends Father { String name; int age; Son(String name, int age) { super(name); if (age < 0) throw new WrongAge("Age cannot be negative"); else this.age = age; } public String toString() { return "Son's name: " + name + ", age: " + age; } } class WrongAge extends Exception { String message; WrongAge(String message) { this.message = message; } public String toString() { return message; } } </pre>	<pre> class Son extends Father { String name; int age; Son(String name, int age) { super(name); if (age < 0) throw new WrongAge("Age cannot be negative"); else this.age = age; } public String toString() { return "Son's name: " + name + ", age: " + age; } } class Father { String name; int age; Father(String name, int age) { this.name = name; this.age = age; } public String toString() { return "Father's name: " + name + ", age: " + age; } } class WrongAge extends Exception { String message; WrongAge(String message) { this.message = message; } public String toString() { return message; } } public class Main { public static void main(String args[]) { Scanner sc = new Scanner(System.in); System.out.print("Enter father's age: "); int fage = sc.nextInt(); System.out.print("Enter son's age: "); int sage = sc.nextInt(); Son son = new Son(name, sage); System.out.println(son); catch (WrongAge e) { System.out.println("Error: " + e.getMessage()); } catch (Exception e) { System.out.println(e.getMessage()); } } } </pre>
---	--

Date _____
Page _____

```

> java Main
Enter father's age: -1
Enter son's age: 1
Error! Age cannot be negative

> java Main
Enter father's age: 20
Enter son's age: 21
Error! Son's age >= Father's age

> java Main
Enter father's age: 25
Enter son's age: 22
Father's age: 25
Son's age: 22
    off scale
    21 22

```

Code:

```

import java.util.Scanner;
class WrongAge extends Exception
{
    public WrongAge(String message)
    {super(message);}
}
class Father
{
    int fage;
    public Father(int fage) throws WrongAge{
        if(fage<0)
            {throw new WrongAge("Age cannot be negative");}
        this.fage=fage;
    }
    public String toString(){
        return "Father's age:"+fage;
    }
}
class Son extends Father
{
    int sage;

```

```

public Son(int fage, int sage) throws WrongAge
{
    super(fage);

    if(sage>=fage)
        {throw new WrongAge("Son\'s age >= Father\'s age");}

    this.sage=sage;
}
public String toString()
{ return super.toString() +"\n" + "Son\'s age:" +sage; }
}

public class Main
{
    public static void main(String args[])
    {
        try{
            Scanner sc = new Scanner(System.in);
            System.out.print("Enter father\'s age:");
            int fage=sc.nextInt();
            System.out.print("Enter son\'s age:");
            int sage=sc.nextInt();
            Son son = new Son(fage, sage);
            System.out.println(son);
        }
        catch(WrongAge e)
        {System.err.println("Error!:"+e.getMessage());}
    }
}

```

OUTPUT

```

> java Main
Enter father's age:21
Enter son's age:18
Father's age:21
Son's age:18
> java Main
Enter father's age:20
Enter son's age:21
Error!:Son's age >= Father's age
> java Main
Enter father's age:17
Enter son's age:16
Father's age:17
Son's age:16

```

LABORATORY 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.

The image shows handwritten notes on a page with horizontal lines. On the left, there is a question and the Java code for a thread class. On the right, there is a try-catch block and the output of the Java application showing the交替输出 of "BMS College of Engineering" and "CSE".

Handwritten Notes:

- Q: WAP to create two threads, one thread displaying "BMS College of Engineering" once every ten seconds, and "CSE" once every 5 seconds.
- Java Code (Left):

```

class Newthread extends Thread {
    String name;
    int time;
    Newthread(String name, int time) {
        this.name = name;
        this.time = time;
    }
    @Override
    public void run() {
        try {
            while(true) {
                System.out.println(name);
                Thread.sleep(time);
            }
        } catch(InterruptedException e) {
            System.out.println("Thread Interrupted");
        }
    }
}
class Main {
    public static void main() {
        Newthread t1 = new Newthread("BMS College of Engineering", 10000);
        Newthread t2 = new Newthread("CSE", 5000);
    }
}

```
- Try-Catch Block (Right):

```

try {
    t1.start();
    t2.start();
} catch(InterruptedException e) {
    System.out.println("Exception" + e);
}

```
- Output (Right):

```

java Main
BMS College of Engineering
CSE
BMS College of Engineering
CSE
BMS College of Engineering
CSE
^C

```

Output of the second run:

```

java Main
BMS College of Engineering
CSE
CSE
CSE
CSE
^C

```

Annotations on the right side of the output area include "OP been off the monitor" and "OP been off the monitor".

Code:

```

class Newthread extends Thread {
    String name;
    int time;
    Newthread(String name, int time) {
        this.name = name;
        this.time = time;
    }
}

```

```

@Override
public void run()
{
    try
    {
        while(true)
        {
            System.out.println(name);
            Thread.sleep(time);
        }
    }
    catch(InterruptedException e)
    {
        System.out.println("Thread interrupted:\n"+e);
    }
}

public class Main
{
    public static void main(String[] args)
    {
        Newthread t1 = new Newthread("BMS College of
Engineering",10000);

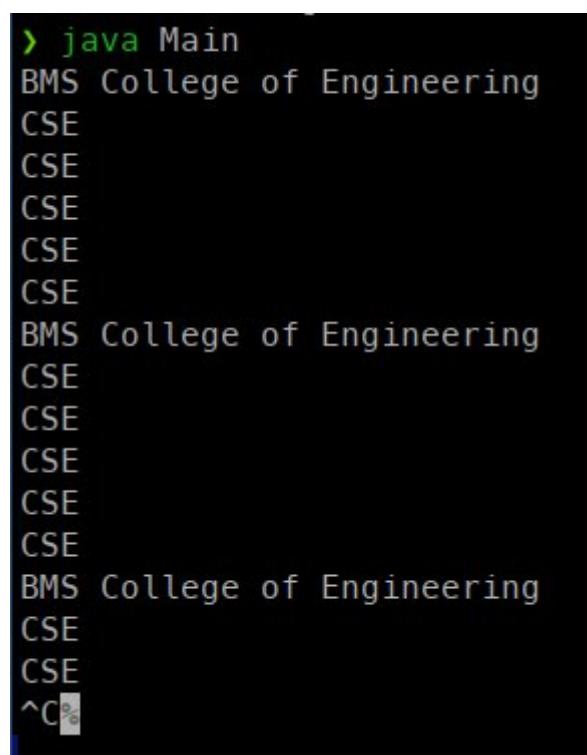
        Newthread t2 = new Newthread("CSE",2000);

        try
        {
            t1.start();
            t2.start();
        }
    }
}

```

```
        catch(Exception e)
        {System.out.println("Exception\n"+e);}
    }
}
```

OUTPUT

A terminal window showing the execution of a Java program named Main. The output consists of three identical lines of text: "BMS College of Engineering" followed by five "CSE" entries. The command "java Main" is visible at the top left.

```
> java Main
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
^C
```

LABORATORY PROGRAM - 9

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

9 WAP to create a UI to perform integer division. The user enters two numbers in the textfields num1 and num2. The division num1/num2 is displayed in the result field. If num1, num2 are not integer, throw a NumberFormatException. If num2 is zero, throw an ArithmeticException. Display the exception in a message dialog box.

```
import java.awt.*;
import java.awt.event.*;
DivisionMain
public class Div extends Frame implements ActionListener
{
    TextField num1, num2;
    Label outResult;
    Button dResult;
    String out = "";
    double resultNum;
    int flag=0;
    public DivisionMain()
    {
        setLayout(new FlowLayout());
        dResult = new Button("Result");
        Label number1 = new Label("Number 1:",Label.RIGHT);
        Label number2 = new Label("Number 2:",Label.RIGHT);
        num1 = new TextField(5);
        num2 = new TextField(5);
        outResult = new Label("Result",Label.RIGHT);
    }
    public void actionPerformed(ActionEvent e)
    {
        if(flag==0)
        {
            try
            {
                resultNum = Double.parseDouble(num1.getText());
                resultNum = Double.parseDouble(num2.getText());
                outResult.setText(resultNum+"");
            }
            catch(NumberFormatException nfe)
            {
                JOptionPane.showMessageDialog(null,"Please enter integer values");
            }
            catch(ArithmeticException ae)
            {
                JOptionPane.showMessageDialog(null,"Division by zero is not allowed");
            }
        }
        else
        {
            JOptionPane.showMessageDialog(null,"Please enter integer values");
        }
    }
}
```

```

Date _____
Page _____
add(number1);
add(num2);
add(number2);
add(num2);
add(dResult);
add(outResult);

num1.addActionListener(this);
num2.addActionListener(this);
dResult.addActionListener(this);

addWindowListener(new WindowAdapter()
{
    public void windowClosing(WindowEvent e)
    {
        System.exit(0);
    }
});

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

        if(n2 == 0)
        {
            throw new ArithmeticException();
        }
        out = n1 + " / " + n2 + " = ";
        resultNum = n1 / n2;
        out += String.valueOf(resultNum);
    }
}

```

```

    } repaint();
}

catch(NumberFormatException e)
{
    flag = 1;
    out = "Number Format Exception," + e; repaint();
}

catch(ArithmeticException e)
{
    flag = 2;
    out = "Division by 0 Exception," + e;
    repaint();
}

public void paint(Graphics g)
{
    if(flag == 0)
    {
        g.drawString(out, outResult.getX() +
                    outResult.getWidth(),
                    outResult.getY() + outResult.getHeight());
    }
    else
    {
        g.drawString(out, 10, 0, 200);
        flag = 0;
    }
}

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

```

Code:

```

import java.awt.*;
import java.awt.event.*;
class DivisionMain1 extends Frame implements ActionListener
{
    TextField num1,num2;
    Button dResult;
    Label outResult;
    String out="";
    double resultNum;
    int flag=0;
    public DivisionMain1()
    {
        setLayout(new FlowLayout());
        dResult = new Button("Result:");
        Label number1 = new Label("Number 1:",Label.RIGHT);
        Label number2 = new Label("Number 2:",Label.RIGHT);
        num1=new TextField(5);
        num2=new TextField(5);
        outResult = new Label("",Label.RIGHT);
        add(number1);
        add(num1);
        add(number2);
        add(num2);
        add(dResult);
        add(outResult);
        num1.addActionListener(this);
        num2.addActionListener(this);
        dResult.addActionListener(this);
        addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e)
            {
                System.exit(0);
            }
        });
    }
    public void actionPerformed(ActionEvent e)
    {
        int n1,n2;

```

```

try
{
    if (e.getSource() == dResult)
    {
        n1=Integer.parseInt(num1.getText());
        n2=Integer.parseInt(num2.getText());
        if(n2==0)
            {throw new ArithmeticException();}
        out=n1+"/"+n2+" ";
        resultNum=n1/n2;
        out+=resultNum;
    }
}
catch(NumberFormatException e1)
{
    flag=1;
    out="Number Format Exception!"+e1;
}
catch(ArithmeticException e1)
{
    flag=1;
    out="Divide by 0 Exception!"+e1;
}
outResult.setText(out);
invalidate();
validate();
}

//public void paint(Graphics g)
//{
//    if(flag==0)
//
//    {g.drawString(out,dResult.getX()+dResult.getWidth(),dResult.g
//etY()+outResult.getHeight()-8);}
//    else
//    {g.drawString(out,100,200); flag=0;}
//}
}

}

```

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



LABORATORY PROGRAM - 10

Demonstrate Interprocess Communication and deadlock.

```
Date _____  
Page _____
```

```
class Q {  
    int n;  
    boolean valueSet = false;  
    synchronized int get()  
    {  
        while (!valueSet)  
        {  
            try  
            {  
                System.out.println("In Consumer waiting");  
                wait();  
            }  
            catch (InterruptedException e)  
            {  
                System.out.println("InterruptedException caught");  
            }  
        }  
        if (this.n == n) valueSet = true;  
        System.out.println("Put: " + n);  
        System.out.println("In Intimate Consumer");  
        notify();  
    }  
}  
  
class Producer implements Runnable  
{  
    Q q;  
    Producer(Q q)  
    {  
        this.q = q;  
        new Thread(this, "Producer").start();  
    }  
    public void run()  
    {  
        int i = 0;  
        while (i < 15)  
        {  
            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;  
    }  
}
```

Date: / /
Page: / /

```

while(i < 15)
{
    int x = q.get();
    System.out.println("Consumed: " + x);
    if(x == 1)
    {
        break;
    }
}

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

```

Date: / /
Page: / /

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.");
    }
}

```

```

> java PCFixed
Press Control-C to stop.
Put: 0
Producer waiting
Got: 3
Intimate Consumer
Intimate Producer
consumed:3
Put: 4
Producer waiting
Intimate Consumer
Got: 0
Intimate Producer
Producer waiting
Put: 1
Intimate Consumer
Got: 4
Intimate Producer
consumed:4
Put: 5
Producer waiting
Intimate Consumer
Got: 1
Intimate Producer
Producer waiting
Put: 2
Intimate Producer
consumed:1
Put: 2
Intimate Consumer
Intimate Producer
Put: 6
Producer waiting
Intimate Consumer
Got: 2
Intimate Producer
Producer waiting
Put: 3
Intimate Producer
consumed:2
Put: 3
Intimate Consumer
Intimate Producer
consumed:5
Got: 6
Intimate Producer
consumed:6

```

consumed:6	Got: 10	
Put: 7	Intimate Producer	
Intimate Consumer	consumed:10	
Producer waiting	Put: 11	
Got: 7	Intimate Consumer	
Intimate Producer	Producer waiting	
consumed:7	Got: 11	
Put: 8	Intimate Producer	
Intimate Consumer	consumed:11	
Producer waiting	Put: 12	
Got: 8	Intimate Consumer	
Intimate Producer	Producer waiting	
consumed:8	Got: 12	
Put: 9	Intimate Producer	
Intimate Consumer	consumed:12	
Producer waiting	Put: 13	
Got: 9	Intimate Consumer	
Intimate Producer	Producer waiting	Intimate Consumer
consumed:9	Got: 13	Got: 14
Put: 10	Intimate Producer	Intimate Producer
Intimate Consumer	consumed:13	consumed:14
Producer waiting	Put: 14	

10 Demonstrate Inter Process Communication and deadlock.

```

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

    synchronized void last() { System.out.println("In
        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 A.last()");
        a.last();
    }

    synchronized void last() { System.out.println("In
        B.last()"); }
}

```

```

class Deadlock implements Runnable
{
    A a = new A();
    B b = new B();
    Deadlock()
    {
        Thread.currentThread().setName("Main");
        Thread t = new Thread(this, "Race");
        t.start();
        a.foo(b);
    }

    public void run()
    {
        b.bar(a);
    }

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

```

Output:

```

Main entered A.foo
Race entered B.bar
Race trying A.last()
Race trying B.last()

```

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 B.last()");
        b.last();
    }

    synchronized void last() { System.out.println("In 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 A.last()");
        a.last();
    }
}

```

```

        }
    synchronized void last() { System.out.println("In B.last()"); }
}

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

```

```

> javac Deadlock.java
> java Deadlock
Main entered A.foo
Race entered B.bar
Race trying A.last()
Main trying B.last()
^C%

```