VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT

on

OBJECT ORIENTED JAVA PROGRAMMING

Submitted by

SRUSHTI N(1BM24CS424)

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 SRUSHTI N(1BM24CS424), 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

Program 1

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

30/09/24

Quadratic equation lab program -1

```
import java.util.Scanner;
class Quadratic{
public static void main (String args[]){
 Scanner input = new Scanner (System.in);
System.out.println ("Enter co-efficient of a :");
double a = input.nextDouble();
System.out.println ("Enter co-efficient of b:");
 double b = input.nextDouble();
System.out.println ("Enter co-efficient of c :");
 double c = input.nextDouble();
double d = b*b -4*a*c;
 if (d>0){
double r1 = (-b + Math.sqrt(d) /2*a;
double r2 = (-b - Math.Sqrt(d)) /2*a;
System.out.println ("Roots are : " +" "+r1 +
                                  " " +r2); }
else if (d==0){
double r1 = -b/2*a;
double r2 = -b/2*a;
System.out.println ("Roots are : " + " "+r1+ " "
                                  +r2); }
else if (d<0){
System.out.println ("Roots are rational"); }
 else {
System.out.println ("Invalid input");
}
 }
}
```

O/P : Enter co-efficient of a : 10
Enter co-efficient of b : 20
Enter co-efficient of c : 30
Roots are rational.

```
import java.util.*;

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

        System.out.println("Enter coefficient of a:");
        double a = input.nextDouble();
        System.out.println("Enter coefficient of b:");
        double b = input.nextDouble();
        System.out.println("Enter coefficient of c:");
        double c = input.nextDouble();
        // Calculating the discriminant
        double d = b * b - 4 * a * c;

        if (d > 0) {
            // Two real and distinct roots
            double r1 = (-b + Math.sqrt(d)) / (2 * a);
            double r2 = (-b - Math.sqrt(d)) / (2 * a);
            System.out.println("Roots are: " + r1 + " and " + r2);
        } else if (d == 0) {
            // One real root
            double r1 = -b / (2 * a);
            double r2 = -b/(2* a);
            System.out.println("Roots are: " + r1 + " and " + r2);
          }
            else if(d<0){
                System.out.println("Roots are rational" );
            } else{
                    System.out.println("Invalid input" );
```

```
        }
    }
}
```

OUTPUT:

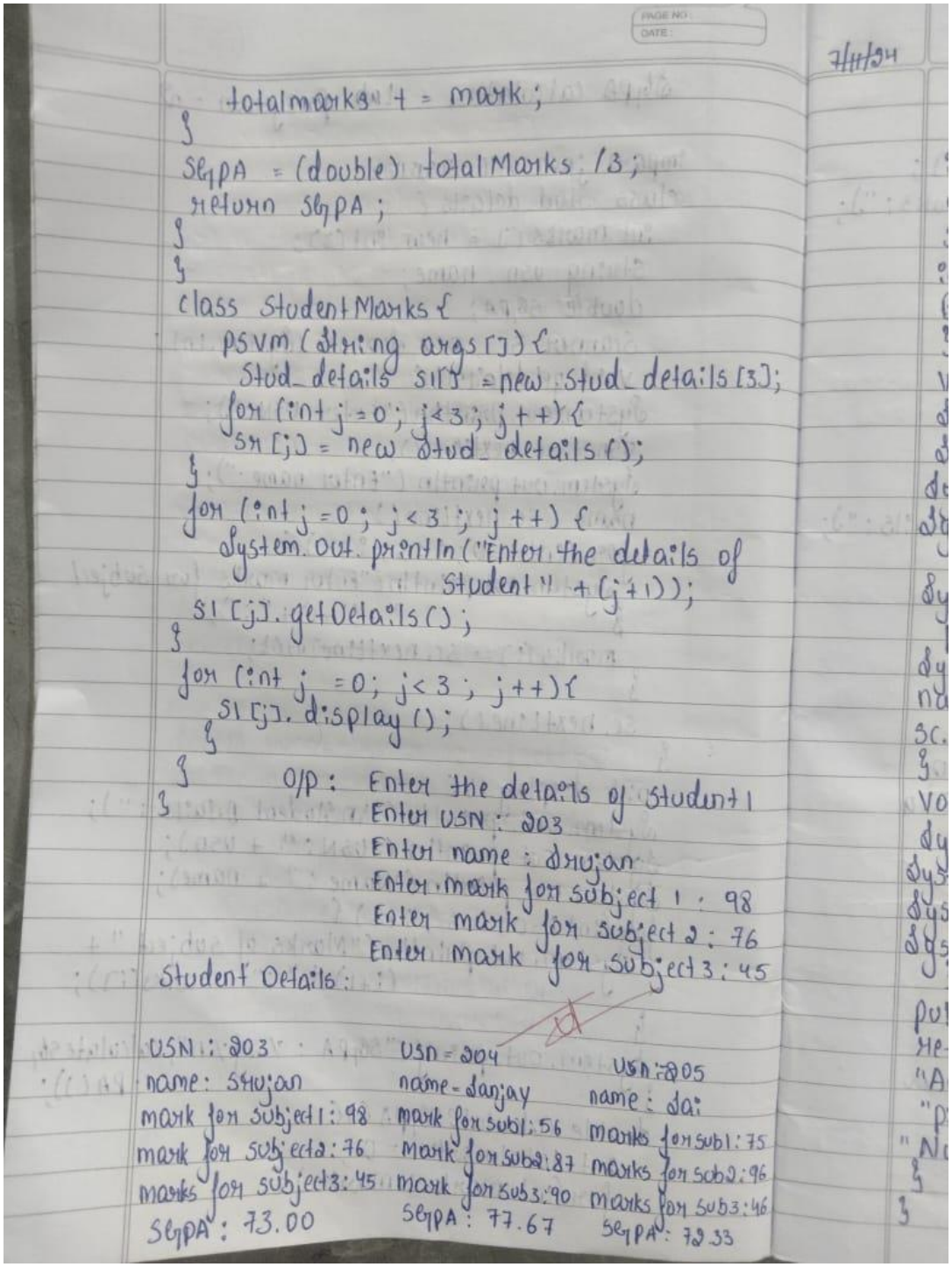
```
D:\24BECS409>javac Quadratic.java

D:\24BECS409>java Quadratic
Enter co-efficient of a:
10
Enter co-efficient of b:
20
Enter co-effiecient of c:
30
Roots are rational

D:\24BECS409>
```

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

SGPA calculator Lab program - 2

```
import java.util.Scanner;
class Stud_details {
  int marks[] = new int[3];
  String usn, name;
  double SGPA;
  Scanner sc = new Scanner(System.in);
  void getdetails(){
  System.out.println("Enter USN:");
  usn = sc.nextLine();
  System.out.println("Enter name:");
  name = sc.nextLine();
  for (int i=0; i<3; i++) {
    System.out.println("Enter marks for subject
    " +(i+1)+ ":");
    marks[i] = sc.nextLineInt();
  }
  sc.nextLine();
}
void display(){
  System.out.println("\nStudent Details :");
  System.out.println("USN:" + usn);
  System.out.println("Name:" + name);
  for (int i=0; i<3; i++) {
    System.out.println("Marks of subject " +
                (i+1) + ":" + marks[i]);
  }
  System.out.printf("SGPA : %.2f %n", calculateSG
}                                  PA());
double calculateSGPA(){
  int totalmarks = 0;
  for(int mark : marks){
```



```
totalmarks += mark;
}
SgPA = (double) totalMarks /3;
return SgPA;
}
}
class StudentMarks {
    psvm (String args[]) {
        Stud_details s1[] = new Stud_details[3];
        for (int j = 0; j<3; j++) {
            s1[j] = new Stud_details();
        }
        for (int j = 0; j < 3; j++) {
            System.out.println("Enter the details of
                                Student " + (j+1));
            s1[j].getDetails();
        }
        for (int j = 0; j< 3; j++) {
            s1[j].display();
        }
    }
}
```

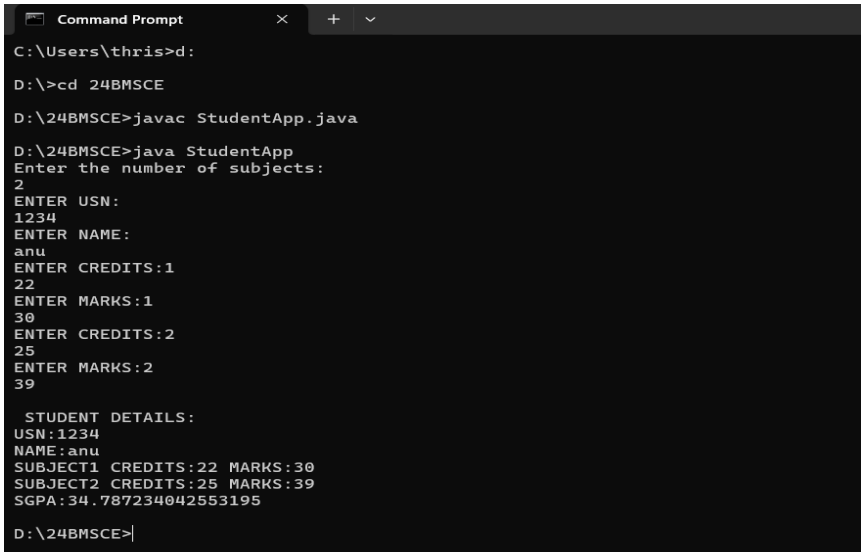
```
O/P : Enter the details of student 1
      Enter USN : 203
      Enter name : Srujan
      Enter mark for subject 1 : 98
      Enter mark for subject 2 : 76
      Enter mark for subject 3 : 45
Student Details :

USN : 203                 USN = 204                 USN : 205
name : Srujan             name = Sanjay             name : Sai
mark for subject1 : 98    mark for sub1 : 56        marks for sub1 : 75
mark for subject2 : 76    Mark for sub2 : 87        marks for sub2 : 96
marks for subject3 : 45   mark for sub3 : 90        marks for sub3 : 46
SgPA : 73.00              SgPA : 77.67              SgPA : 72.33
```

```
import java.util.Scanner;
class Stud_details{

    int marks[] = new int[3];
    String usn, name;
```

```
double SGPA;
Scanner sc = new Scanner(System.in);
void getdetails() {
    System.out.println("Enter USN:");
    usn = sc.nextLine();
    System.out.println("Enter Name:");
    name = sc.nextLine();
    for (int i = 0; i < 3; i++) {
        System.out.println("Enter marks for subject" + (i + 1) + ":");
        marks[i] = sc.nextInt();
    }
    sc.nextLine();
}
void display() {
    System.out.println("\nStudent Details:");
    System.out.println("USN: " + usn);
    System.out.println("Name: " + name);
    for (int i = 0; i < 3; i++) {
        System.out.println("Marks of subject" + (i + 1) + ":" + marks[i]);
    }
    System.out.printf("SGPA: %2f%n", calculateSGPA());
}
double calculateSGPA() {
    int totalmarks = 0;
    for (int marks : marks) {
        totalmarks += marks;
    }
    SGPA = (double) totalmarks / 3;
    return SGPA;
```

```
    }
}

class StudentMarks {
    public static void main(String args[]) {
        Stud_details s[] = new Stud_details[3];
        for (int j = 0; j < 3; j++) {
            s[j] = new Stud_details();
            System.out.println("Enter the details of student" + (j + 1));
            s[j].getdetails();
        }
        for (int j = 0; j < 3; j++) {
            s[j].display();
        }
    }
}
```

OUTPUT:



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.

7/11/24

Book details lab program - 3

```
import java.util.Scanner;
class Book{
String name, author;
int price, num_pages;
Book(){
}
void details(){
Scanner sc = new Scanner(System.in);
System.out.println("Enter name of the Book:");
System.out.println name = sc.nextLine();
System.out.println("Enter author name:");
author = sc.nextLine();
System.out.println("Enter price:");
price = sc.nextInt();
System.out.println("Enter number of pages:");
num_pages = sc.nextInt();
sc.nextLine();
}
void getDetails(){
System.out.println("Name of the book:" + name);
System.out.println("Author Name :" +author);
System.out.println("price of the book:"+price);
System.out.println("Number of pages in the book:
}                                    "+num_pages);
public String toString(){
return "Book Name:" + name + "\n"+
"Author:" +author + "\n" +
"price:" + price + "\n"+
"Number_of_pages:" + num_pages);
}
}
```

```
class MyBook {
  psvm (String args[]) {
Scanner sc = new Scanner(System.in);
System.out.println("Enter the no of books:");
int n = sc.nextInt();
sc.nextLine();
Book[] books = new book[n];
for (int i=0; i<n; i++) {
  books[i] = new Book();
books[i].details();
books[i].getDetails();
}
System.out.println("\n All Book details:");
for (Book book: books){
  System.out.println(books);
}
sc.close();
}
}
```

```
O/P : Enter the number of Books : 10
      Enter name of the book : Java
      Enter author name : tagoor
      Enter price : 200
      Enter no. of pages : 300
    Details of the book:
      Name of the book : java
      Author Name : tagoor
      price of the book : 200
      NO. of pages in the book : 300
```

```
import java.util.Scanner;
class Book {
    String name, author;
    int price, num_pages;
    Book() {
```

```
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter name of the book:");
        name = sc.nextLine();
        System.out.println("Enter author name:");
        author = sc.nextLine();
        System.out.println("Enter price:");
        price = sc.nextInt();
        System.out.println("Enter number of pages:");
        num_pages = sc.nextInt();
        sc.nextLine();
    }
    void getdetails() {
        System.out.println("Name of the book: " + name);
        System.out.println("Author name: " + author);
        System.out.println("Price of the book: " + price);
        System.out.println("Number of pages in the book: " + num_pages);
    }
    public String toString() {
        return "Book name: " + name + "\nAuthor: " + author + "\nPrice: " + price + "\nNumber
of pages: " + num_pages;
    }
}
class MyBook {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the number of Books:");
        int n = sc.nextInt();
        sc.nextLine();
        Book[] books = new Book[n];
        for (int i = 0; i < n; i++) {
            books[i] = new Book();
```

```
        }
        System.out.println("\nAll Book details:");
        for (Book book : books) {
            System.out.println(book);
        }
        sc.close();
    }
}
```

OUTPUT:

```
D:\24BMSCE>javac MyBook.java

D:\24BMSCE>java MyBook
Enter the number of books
2
enter name of the book:
motivation
enter author of the book:
shakesphere
enter price of the book:
200
enter number of pages of the book:
1999
Name of the book:motivation
Author of the book:shakesphere
Price of the book:200.0
Number of pages of the book:1999
enter name of the book:
story
enter author of the book:
william
enter price of the book:
500
enter number of pages of the book:
4678
Name of the book:story
Author of the book:william
Price of the book:500.0
Number of pages of the book:4678

 All book details:
Book Name:motivation
Author:shakesphere
price:200.0
Number of pages:1999

Book Name:story
Author:william
price:500.0
Number of pages:4678


D:\24BMSCE>
```

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.

24/10/24

Lab program-4

```
import java.util.Scanner;

abstract class Shape {
    int dimension1;
    int dimension2;

    abstract void printArea();
}
class Rectangle extends Shape {
    public Rectangle (int length, int width){
        this.dimension1 = length;
        this.dimension2 = width;
    }
    @Override
    void printArea (){
        int area = dimension1* dimension2;
        System.out.println("Rectangle Area :" +area);
    }
}
class Triangle extends Shape {
    public Triangle (int base, int height) {
        this.dimension1 = base;
        this.dimension2 = height;
    }
    @Override
    void printArea (){
        double area = 0.5 * dimension1* dimension2;
        System.out.println("Triangle Area :" + area);
    }
}
```

```
class Circle extends Shape {
    private final double pi = 3.14159;
    public Circle (int radius){
        this.dimension1 = radius;
        this.dimension2 = 0;
    }
    @Override
    void printArea(){
        double Area = pi * dimension1 * dimension1;
        System.out.println("Circle Area :"+area);
    }
}
public class Main {
    public static void main (String[] args)
        Scanner scanner = new Scanner (System.in);

    System.out.print("Enter length of rectangle:");
    int length = scanner.nextInt();
    System.out.print("Enter width of rectangle:");
    int width = scanner.nextInt();
    Rectangle rectangle = new Rectangle(length, width);
    rectangle.printArea();

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

    System.out.print("Enter radius of circle:");
    int radius = scanner.nextInt();
    Circle circle = new Circle(radius);
    circle.printArea();
    scanner.close();
    }
}
```

O/p :

```
Enter length of rectangle :3
Enter width of rectangle: 0
Rectangle Area : 6
Enter base of triangle: 4
Enter height of triangle :5
Triangle Area : 10.0
Enter radius of Circle :2
Circle Area : 12.56636
```

import java.util.Scanner;

abstract class Shape {

```
    int dimension1;
    int dimension2;
    abstract void printArea();
}
class Rectangle extends Shape {
    public Rectangle(int length, int width) {
        this.dimension1 = length;
        this.dimension2 = width;
    }
    void printArea() {
        int area = dimension1 * dimension2;
        System.out.println("Rectangle Area: " + area);
    }
}
class Triangle extends Shape {
    public Triangle(int base, int height) {
        this.dimension1 = base;
        this.dimension2 = height;
 }
     void printArea() {
        double area = 0.5 * dimension1 * dimension2;
        System.out.println("Triangle Area: " + area);
    }
}
class Circle extends Shape {
    private final double pi = 3.14159;
    public Circle(int radius) {
        this.dimension1 = radius;
        this.dimension2 = 0;
    }
```

```
    void printArea() {
        double area = pi * dimension1 * dimension1;
        System.out.println("Circle Area: " + area);
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter length of rectangle: ");
        int length = scanner.nextInt();
        System.out.print("Enter width of rectangle: ");
        int width = scanner.nextInt();
        Rectangle rectangle = new Rectangle(length, width);
        rectangle.printArea();

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

        System.out.print("Enter radius of circle: ");
        int radius = scanner.nextInt();
        Circle circle = new Circle(radius);
        circle.printArea();
        scanner.close();
    }
}
```

OUTPUT:

```
Enter length of rectangle: 3
Enter width of rectangle: 2
Rectangle Area: 6
Enter base of triangle: 4
Enter height of triangle: 5
Triangle Area: 10.0
Enter radius of circle: 2
Circle Area: 12.56636
```

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.

7/11/24

lab-5 creating a class Bank & more

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

  Account(String customerName,
    String accountNumber, double initialBalance){
      this.customerName = customerName;
      this.accountNumber = accountNumber;
      this.balance = initialBalance;
    }

  Abstract void deposit(double amount);
  abstract void displayBalance();
  abstract void withdraw(double amount);
}
class SavAcct extends Account{
  double interestRate;
    SavAcct(String customerName, String
  accountNumber, double initialBalance,
  double interestRate){
    super(customerName, accountNumber,
      initialBalance);
    this.interestRate = interestRate;
}
void deposit(double amount){
  balance += amount;
}
void displayBalance(){
  System.out.println("Savings Balance:" +
      balance);
}
void withdraw(double ammount){
  if (amount <= balance)
    balance -= amount;
  }
  void ComputeAndDepositInterest(){
    balance += balance * interestRate / 100;
  }
}
class CurAcct Extends Account{
  static final double MIN_BALANCE = 1000,
      SERVICE_CHARGE = 50;
  CurAcct(String customerName, String
    accountNumber, double initialBalance){
    super(customerName, accountNumber, initialBalance);
  }
  void deposit(double ammount){
      balance += amount;
  }
  void displayBalance(){
    System.out.println("current Balance:" +
      balance);
  }
  void withdraw(double amount){
    if (amount <= balance){
      balance -= amount;
      if (balance < MIN_BALANCE) balance -=
        SERVICE_CHARGE;
    }
  }
}
```

```
class Bank{
 public static void main (String [] args){
  scanner scanner = new Scanner (System.in);
  System.out.println ("Enter account type
      savings/current): ");
  String type = scanner.nextLine();
  System.out.println("Enter customer name:");
  String name = scanner.nextLine();
  System.out.println
  Account account;
  if (type.equals ("savings")) {
   System.out.println("Intial balance &
      Interest rate:");
   account = new SavAcct (name, number,
      Scanner.nextDouble(), Scanner.nextDouble());
  }
  else{
   System.out.println ("Intial balance :");
   account = new CurAcct(name, number,
      Scanner.nextDouble());
  }
  while (true){
   System.out.println ("\n1. Deposit
   2. Display Balance 3. Withdraw
      4. Interest 5. Exit");
   int choice = scanner.nextInt();
   switch(choice){
   case 1: account.displayBalance(); depositor(scanner.nextDouble());
      break;
   case 2: account.withdraw(Scanner.nextDouble());
      break;
   case 1: account.depositor(scanner.nextDouble());
      break;
   case 2: account.displayBalance();
      break;
   case 3: account.withdraw(Scanner.
      nextDouble());
   break;
   case 4: if (account instance of SavAcct)
      ((SaveAcct) account).computeAndDeposit
      Interest();
   break;
   case 5:
      return;
   }
  }
 }
}
```

output:

```
javac Bank.java
 ~~jav f~~ Enter account type (savings/current):
 savings
 Enter customer name : Srushti
 Enter account number : 2837
 intial Balance : 372

 1. Deposit 2. Display balance 3. withdraw
 4. Interest 5. Exit
 1
 300

1. Deposit 2. Display balance 3. withdraw 4. Interest 5. Exit
2
current Balance : 672.0
```

```
import java.util.Scanner;

abstract class Account {
    String customerName, accountNumber;
    double balance;

    Account(String customerName, String accountNumber, double initialBalance) {
        this.customerName = customerName;
        this.accountNumber = accountNumber;
        this.balance = initialBalance;
    }

    abstract void deposit(double amount);
    abstract void displayBalance();
    abstract void withdraw(double amount);
}

class SavAcct extends Account {
    double interestRate;

    SavAcct(String customerName, String accountNumber, double initialBalance, double
interestRate) {
        super(customerName, accountNumber, initialBalance);
        this.interestRate = interestRate;
    }

    void deposit(double amount) {
        balance += amount;
    }
```

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

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

    void computeAndDepositInterest() {
        balance += balance * interestRate / 100;
    }
}

class CurAcct extends Account {
    static final double MIN_BALANCE = 1000, SERVICE_CHARGE = 50;

    CurAcct(String customerName, String accountNumber, double initialBalance) {
        super(customerName, accountNumber, initialBalance);
    }

    void deposit(double amount) {
        balance += amount;
    }

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

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

```
            balance -= amount;
            if (balance < MIN_BALANCE) balance -= SERVICE_CHARGE;
        }
    }
}

class Bank {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter account type (savings/current): ");
        String type = scanner.nextLine();

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

        System.out.println("Enter account number: ");
        String number = scanner.nextLine();

        Account account;
        if (type.equals("savings")) {
            System.out.println("Initial balance and interest rate: ");
            account = new SavAcct(name, number, scanner.nextDouble(), scanner.nextDouble());
        } else {
            System.out.println("Initial balance: ");
            account = new CurAcct(name, number, scanner.nextDouble());
        }

        while (true) {
            System.out.println("\n1. Deposit 2. Display Balance 3. Withdraw 4. Interest 5. Exit");
            int choice = scanner.nextInt();
```

```
        switch (choice) {
            case 1: account.deposit(scanner.nextDouble());
            break;
            case 2: account.displayBalance();
            break;
            case 3: account.withdraw(scanner.nextDouble());
            break;
            case 4: if (account instanceof SavAcct) ((SavAcct)
account).computeAndDepositInterest();
            break;
            case 5:
            return;
        }
    }
  }
}
```

OUTPUT:

```
D:\24BMSCE>javac Bank.java

D:\24BMSCE>java Bank
Enter account type (savings/current):
savings
Enter customer name:
anu rai
Enter account number:
123786645087301
Initial balance and interest rate:
5000
50

1. Deposit 2. Display Balance 3. Withdraw 4. Interest 5. Exit
1
200

1. Deposit 2. Display Balance 3. Withdraw 4. Interest 5. Exit
2
Savings Balance: 5200.0

1. Deposit 2. Display Balance 3. Withdraw 4. Interest 5. Exit
3
100

1. Deposit 2. Display Balance 3. Withdraw 4. Interest 5. Exit
4

1. Deposit 2. Display Balance 3. Withdraw 4. Interest 5. Exit
2
Savings Balance: 7650.0

1. Deposit 2. Display Balance 3. Withdraw 4. Interest 5. Exit
5
```

Program 6

Create a package CIE which has two classes - Personal 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 Personal. 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.

14/11/24

Week - 6

Create a package CIE which has two classes - Student & Internals. The class personal has members like usn, name, sem. The class internals has an array that stories 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 stories 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.

```
//CIE / Student.java
package CIE;

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

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

public class internals{
    public int[] internalMarks = new int[5];
```

```
    public Internals (int [] marks){
      for (int i = 0; i<5; i++){
        internalMarks[i] = marks[i];
      }
    }
}

// SEE / External.java
package SEE;
import CIE.Student;

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

  public External (String usn, String name, int sem,
      int[] marks){
    super(usn, name, sem);
    for (int i=0; i<5; i++){
      SeeMarks[i] = marks[i];
    }
  }
}

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

public class Main {
public static void main (String [] args){
Scanner sc = new Scanner (System.in);
System.out.println("Enter the number of
    students: ");
int n = sc.nextInt();
for (int i =0; i<n; i++){
```

```
sc.nextline();
System.out.println("Enter USN :");
String usn = sc.nextline();
System.out.println("Enter semester :");
int sem = sc.nextInt();
int[] internalMarks = new int[5];
System.out.println("Enter 5 Internal marks:");
for (int j = 0; j<5; j++){
internalMarks[j] = sc.nextInt();
}
sc.nextline();
int[] SeeMarks = new int[5];
System.out.println("Enter 5 SEE Marks :");
for (int j=0; j<5; j++){
SeeMarks[j] = sc.nextInt();
}
sc.nextline();
Internals internal = new Internals (internalMarks);
External external = new External (usn, name, sem, seemarks);
System.out.println("\n Final Marks for Student: "
    + name + " (USN : " + usn + ") :");
for (int j=0; j<5; j++){
int finalMarks = internal.internalMarks[j] +
(SeeMarks[j]/2);
System.out.println("Course" + (j+1) + ":" +
    finalMark);
}
System.out.println();
}
sc.close();
}
}
```

```
-> cd : C:\Users\Admin\Desktop\Shushik_N_2[illegible]
-> javac -d . CIE/Student.java CIE/Internals.java
                                SEE/External.java Main.java
O/P :                                  -> java Main
javac Student.java                  } do compile for
javac -d. Student.java              } all the files like this

Enter the number of students : 0
Enter usn : 2001
Enter name : Sudshik
Enter semester : 3
Enter 5 internal marks:
93
34
41
59
93
Enter 5 SEE Marks :
87
98
70
90
78
Final marks for student : Shushik (USN : 2001)
Course 1 : 88
Course 2 : 83
Course : 76
Course 4 : 99
Course 5 : 69

Enter usn : 200
Enter name : pallavi
Enter semester : 3
Enter 5 internal marks :
34
56
```

```
93
15
46
Enter 5 SEE marks
89
90
78
89
90
Final marks for student : pallavi (USN : 200)
Course 1 : 78
Course 2 : 101
Course 3 : 68
Course 4 : 59
Course 5 : 91
```

```
package CIE;
public class Student {
    private String usn;
    private String name;
    private int sem;
    public Student(String usn, String name, int sem) {
        this.usn = usn;
        this.name = name;
        this.sem = sem;
    }
}
package CIE;
public class Internals {
    private int[] internalMarks=new int[5];
    public Internals(int [] marks) {
         for(int i=0; i<5; i++){
        internalMarks [i] = marks[i];
      }
   }
}
package SEE;
import CIE.Student;
public class External extends Student {
    private int[] SeeMarks=new int[5];
     public External(String usn, String name, int sem,int[]marks) {
        super(usn, name, sem);
        for(int i=0; i<5; i++){
           SeeMarks[i]=marks[i];
        }
    }
```

```
}
import CIE.*;
import SEE.External;
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter number of students: ");
        int n = sc.nextInt();
        for (int i = 0; i < n; i++) {
          sc.nextLine();
          System.out.println("Enter USN for Student: ");
          String usn = sc.nextLine();
          System.out.println("Enter name for Student: ");
          String name = sc.nextLine();
          System.out.println("Enter semester for Student: ");
          int sem = sc.nextInt();
          int[] internalMarks = new int[5];
          System.out.println("Enter 5 internal marks for 5 courses: ");
          for (int j = 0; j < 5; j++) {
              internalMarks[j] = sc.nextInt();
          }
          sc.nextLine();
          int[] seeMarks = new int[5];
          System.out.println("Enter 5 SEE Marks: ");
          for (int j = 0; j < 5; j++) {
          seeMarks[j] = sc.nextInt();
}
sc.nextLine();
Internals internal = new Internals(internalMarks);
```

```
External external = new External(usn, name, sem, seeMarks);
System.out.println("\nFinal Marks for Student:");
System.out.println("USN: " + usn);
for (int j = 0; j < 5; j++) {
    int finalMarks = internal.internalMarks[j] + (seeMarks[j] / 2);
    System.out.println("Course " + (j + 1) + ": " + finalMarks);
}
System.out.println();
sc.close();
  }
}
```

OUTPUT:

```
D:\24BMSCE>java Main
Enter number of students: 1

Enter USN for student 1: 11234
Enter name for student 1: anupriyaa
Enter semester for student 1: 3
Enter internal marks for 5 courses:
21
22
23
24
25
Enter external marks for 5 courses:
89
90
91
92
93

Final Marks for Students:

Student 1: anupriyaa (11234)
Semester: 3
Final Marks:
Course 1: 110
Course 2: 112
Course 3: 114
Course 4: 116
Course 5: 118
```

Program 7

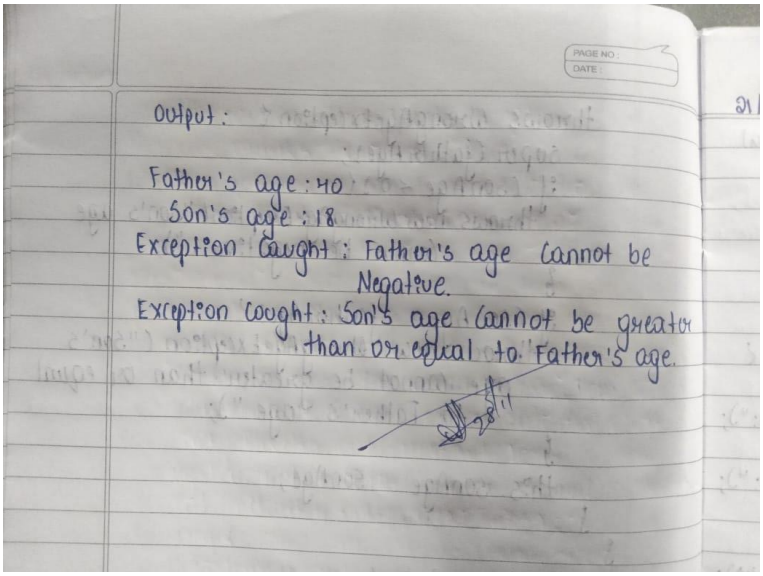
Write a program that demonstrates handling of exceptions in inheritance tree. Create a base class called “Father” and derived class called “Son” which extends the base class. In Father class, implement a constructor which takes the age and throws the exception WrongAge( ) when the input age=father’s age.

29/11/24

Week - 7

Write a program that demonstrates handling of exceptions 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 i/p age < 0. In son class, implement a constructor that uses both father and Son's age & throws an exception if Son's age >= father's age.

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

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

class Son Extends Father {
int sonAge;
public Son(int fatherAge, int SonAge)
throws WrongAgeException{
super(fatherAge);
if (SonAge < 0){
throws new WrongAgeException("Son's age cannot be negative");
}
if (SonAge >= fatherAge){
throw new WrongAgeException("Son's age cannot be greater than or equal to Father's age");
}
this.sonAge = SonAge;
}
}

public class Demo{
public static void main(String ax[]){
try{
Father father = new Father(40);
Son son = new Son(40,18);
System.out.println("Fathers age:"+father.age);
System.out.println("Son's age:"+ Son.sonAge);
Father Invalid Father = new Father(-5);
}
catch(WrongAgeException e){
System.out.println("Exception caught:"+e.getMessage());}
try{
Son invalid Son = new Son(35,36);
}
catch(WrongAgeException e){
System.out.println("Exception caught:"+
e.getMessage());
}
}
```



Output :

Father's age : 40
Son's age : 18
Exception Caught : Father's age cannot be Negative.
Exception caught : Son's age cannot be greater than or equal to Father's age.

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

```
        }
        this.sonAge = sonAge;
    }
}
public class Demo {
    public static void main(String args[]) {
        try {
            Father father = new Father(40);
            Son son = new Son(10, 40);
            System.out.println("Father's age: " + father.age);
            System.out.println("Son's age: " + son.sonAge);
        } catch (WrongAgeException e) {
            System.out.println("Exception caught: " + e.getMessage());
        }
    }
}
```

OUTPUT:

```
D:\24BECS409>javac Demo.java

D:\24BECS409>java Demo
Father's age: 40
Son's age: 18
Exception caught: Father's age cannot be negative!
Exception caught: Son's age cannot be greater than or equal to Father's age!

D:\24BECS409>
```

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.

28/11/24

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.

```
=> class BMSCollegeThread extends Thread{
    public void run(){
        try{
            while(true){
                System.out.println("BMS College of
                    Engineering");
                Thread.sleep(10000);
            }
        }
        catch (InterruptedException e){
            System.out.println("Thread Interrup
                -ted:" + e.getMessage());
        }
    }
    class CSEThread extends Thread{
        public void run(){
            try{
                while(true){
                    System.out.println("CSE");
                    Thread.sleep(2000);
                }
            }
            catch ("InterruptedException e){
                System.out.println("Thread Interru
                -pted:" + e.getMessage);
            }
        }
    }
```

```
public class MultiThreadExample{
    public static void main(String[] args){
        BMSCollegeThread bmsThread = new
            BMSCollegeThread();
        CSEThread cseThread = new CSEThread();
        bmsThread.start();
        cseThread.start();
    }
```

O/P:

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

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

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

        BMSCollegeThread bmsThread = new BMSCollegeThread();
        CSEThread cseThread = new CSEThread();
```

```
        bmsThread.start();
        cseThread.start();
    }
}
```

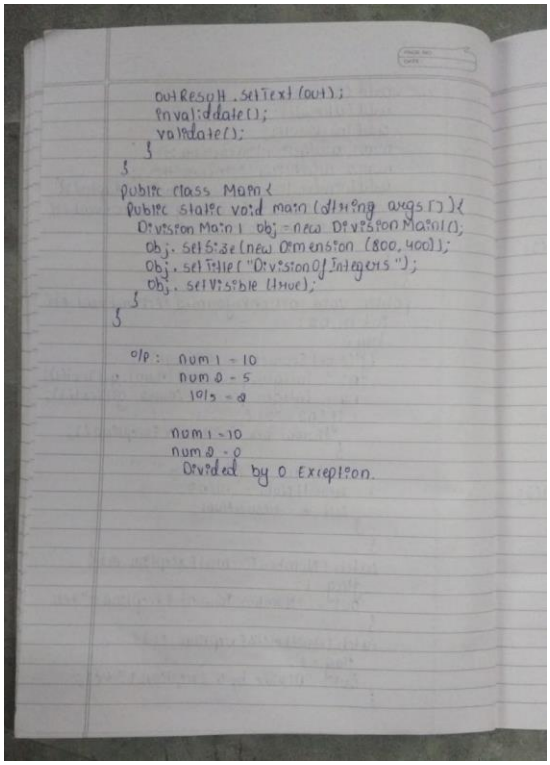
OUTPUT:

```
Command Prompt

D:\24BMSCE>javac MultiThreadExample.java

D:\24BMSCE>java MultiThreadExample
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE
CSE
^C
D:\24BMSCE>
```

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.

```
lab-9

WAP that creates a user interface to perform integer
-ca divisions. The user enters two numbers in
the text fields Num1 & Num2. The division of
Num1 & Num2 is displayed in the Result field
when the divide button is clicked. If Num1 or
Num2 were not an integer, the program would
throw a NumberFormatException if num2 were
zero, the program would throw an Arithmetic
Exception. Display the exception in a message
dialog box.

import java.awt.*;
import java.awt.event.*;
class DivisionMain extends Frame implements
    ActionListener {
    TextField num1,num2;
    Button dResult;
    Label outResult;
    String out = "";
    double resultNum;
    int flag=0;
    public DivisionMain(){
        setLayout(new FlowLayout());
        dResult = new Button("Result:");
        label number1 = new Label("Number1:",
                                  Label.RIGHT);
        Label(Number 2:", Label.RIGHT);
        num1 = new TextField(5);
        num 2 = 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);
        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 e){
            flag=1;
            out = "Number Format Exception!"+e;
        }catch(ArithmeticException e){
            flag=1;
            out = "Divide by 0 Exception!"+e;
        }
```



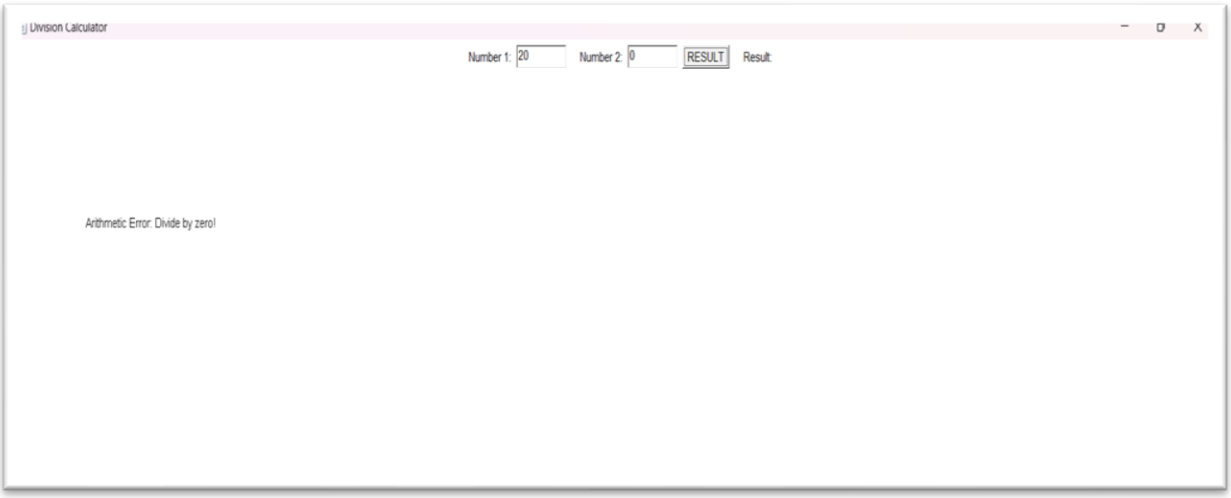
```
outResult.setText(out);
invalidate();
validate();
}
}
public class Main {
public static void main(String args[]){
DivisionMain1 obj=new DivisionMain1();
obj.setSize(new Dimension(800,400));
obj.setTitle("Division Of Integers");
obj.setVisible(true);
}
}

o/p: num1 = 10
num2 = 5
10/5 = 2

num1 = 10
num2 = 0
Divided by 0 Exception.
```

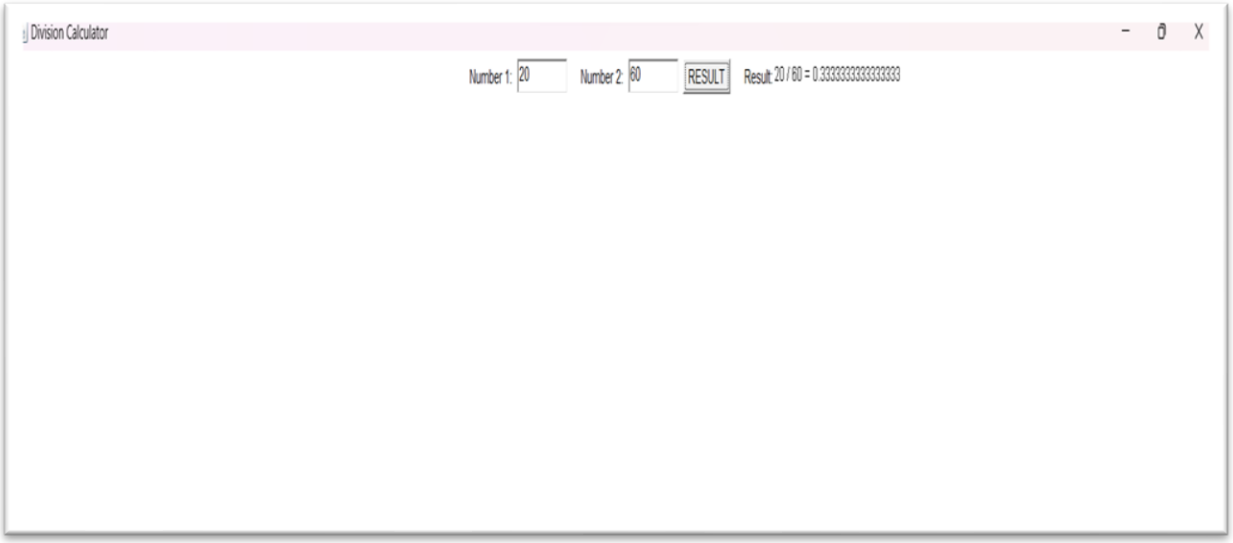
```
import java.awt.*;
import java.awt.event.*;
class DivisionMain1 extends Frame implements ActionListener{
        TextField num1,num2;
        Button dResult;
        Label outResult;
        String out="";
        double resultNum;
        int flag=0;
        public DivisionMain1(){
                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 class Main{
        public static void main(String args[]){
            DivisionMain1 obj=new DivisionMain1();
            obj.setSize(new Dimension(800,400));
            obj.setTitle("DivisionOfIntegers");
            obj.setVisible(true);
        }
}
```

OUTPUT:


Division Calculator
Number 1: 20
Number 2: 0
RESULT
Result:
Arithmetic Error: Divide by zero!


Division Calculator
Number 1: 20
Number 2: 60
RESULT
Result: 20 / 60 = 0.3333333333333333

Program 10

Demonstrate Inter process Communication and deadlock

LAB - 10

Demonstrate Interprocess Communication & deadlock.

```
class Q {
 int n;
 boolean valueSet = false;
 synchronized int get() {
   while (! valueset)
    try {
     System.out.println ("\n. Consumer waiting\n");
     wait();
    }
    catch (InterruptedException e) {
     System.out. println ("Interrupted Exception
      caught");
    }
    System.out.println ("Got: " + n);
    ValueSet = false;
    System.out. println (" \n Intimate producer\n");
    notify();
    return n;
 }
 synchronized void put(int n) {
   while (ValueSet) {
    try {
     S.o.pln ("\n. producer waiting \n");
     wait();
    }
    catch (InterruptedException e) {
     S.o.pln (" Interrupted Exception caught");
    }
    this.n = n;
    ValueSet = true;
    S.o.pln ("put: " + n);
```

```
class PcFixed {
public static void main(String args[]) {
    Q q = new Q();
    new Producer(q);
    new Consumer(q);
    s.o.pln("Press Control-C to stop");
}
}
```

```
o/p:
Press Control-C to stop
Put : 0
Intimate Consumer
Producer waiting
Put : 0
Intimate ~~Con~~ Producer
Producer waiting
Consumed : 0
Put : 1
Intimated Producer
Consumed : 1
Put : 2
Intimate Consumer
Producer waiting
Put : 2
```

```
s.o.pln("\n Intimate 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 n = q.get();
            s.o.pln("Consumed: "+n);
            i++;
        }
    }
}
```

```
class Q {
    int n;
    boolean valueSet = false;
    synchronized int get() {
        while (!valueSet) {
```

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

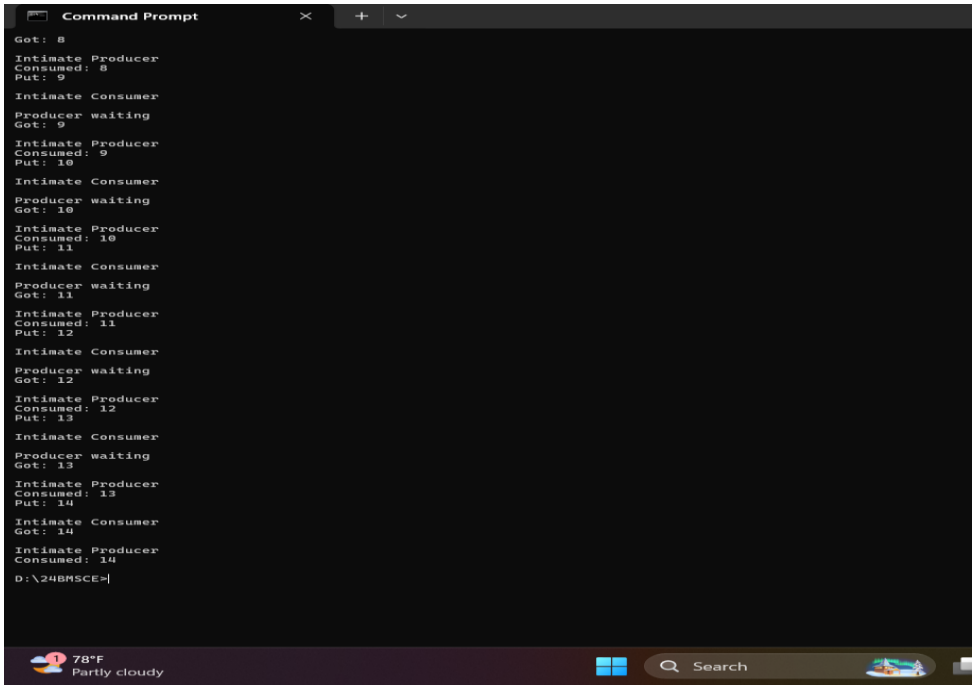
```
class Producer implements Runnable {
    Q q;
    Producer(Q q) {
        this.q = q;
        new Thread(this, "Producer").start();
    }
    public void run() {
        int i = 0;
        while (i < 15) {
            q.put(i++);
        }
    }
}
class Consumer implements Runnable {
    Q q;
  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

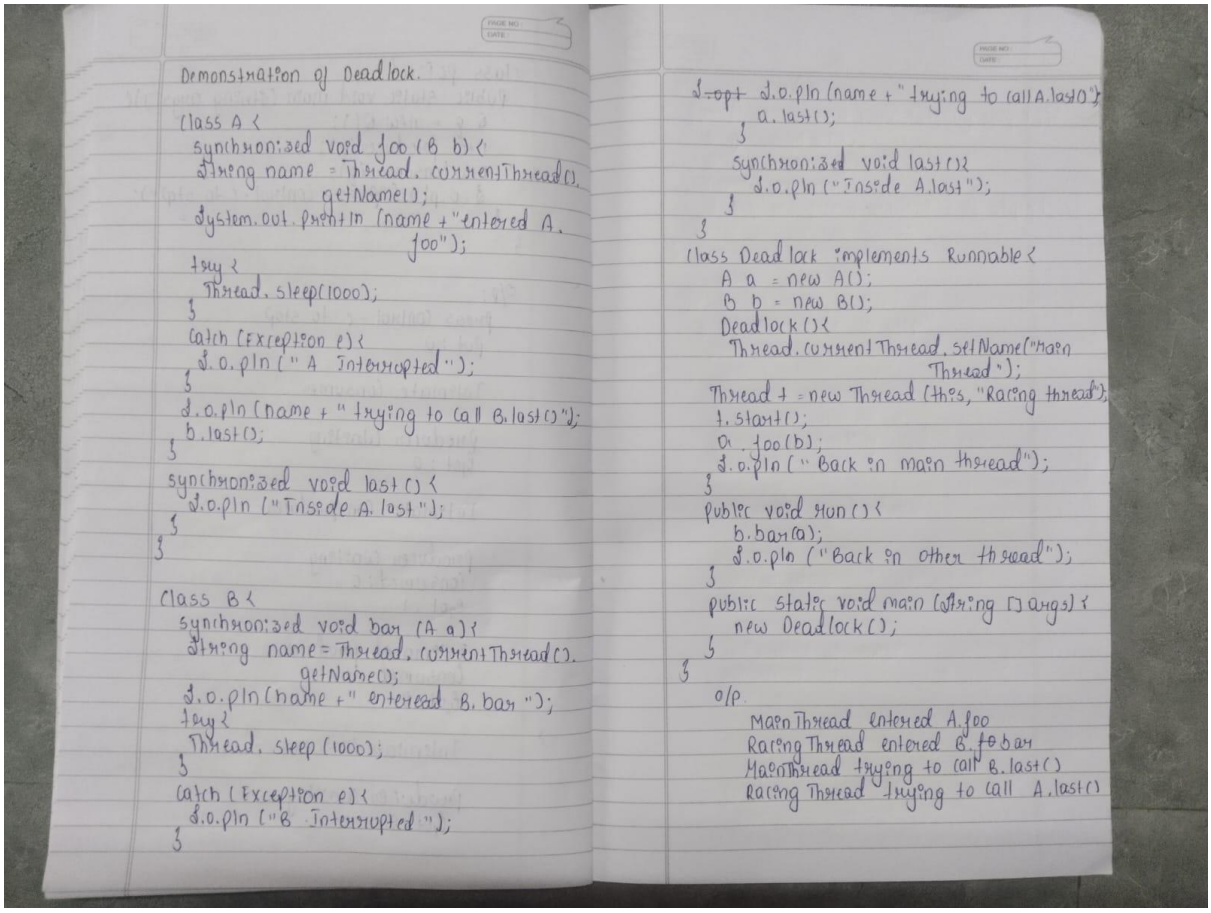
OUTPUT:

```
D:\24BMSCE>javac PCFixed.java

D:\24BMSCE>java PCFixed
Press Control-C to stop.
Put: 0

Intimate Consumer

Producer waiting
Got: 0

Intimate Producer
Put: 1

Intimate Consumer

Producer waiting
Consumed: 0
Got: 1

Intimate Producer
Consumed: 1
Put: 2

Intimate Consumer

Producer waiting
Got: 2

Intimate Producer
Consumed: 2
Put: 3

Intimate Consumer

Producer waiting
Got: 3

Intimate Producer
Consumed: 3
```

```
Command Prompt

Intimate Producer
Consumed: 3
Put: 4

Intimate Consumer

Producer waiting
Got: 4

Intimate Producer
Consumed: 4
Put: 5

Intimate Consumer

Producer waiting
Got: 5

Intimate Producer
Consumed: 5
Put: 6

Intimate Consumer

Producer waiting
Got: 6

Intimate Producer
Consumed: 6
Put: 7

Intimate Consumer

Producer waiting
Got: 7

Intimate Producer
Consumed: 7
Put: 8

78°F
Partly cloudy
Search
```



Demonstration of Deadlock



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) {
      S.o.pln ("A Interrupted");
    }
    S.o.pln (name + " trying to call B.last()");
    b.last();
  }
  synchronized void last () {
    S.o.pln ("Inside A.last");
  }
}

class B {
  synchronized void bar (A a) {
    String name = Thread.currentThread().
                  getName();
    S.o.pln (name + " enteread B.bar ");
    try {
      Thread.sleep (1000);
    }
    catch (Exception e) {
      S.o.pln ("B Interrupted");
    }
    S.o.pln (name + " trying to call A.last()");
    a.last();
  }
  synchronized void last () {
    S.o.pln ("Inside A.last");
  }
}
class Deadlock implements Runnable {
  A a = new A();
  B b = new B();
  Deadlock () {
    Thread.currentThread.setName("main
                  Thread");
    Thread t = new Thread (this, "Racing thread");
    t.start();
    a.foo(b);
    S.o.pln ("Back in main thread");
  }
  public void run () {
    b.bar(a);
    S.o.pln ("Back in other thread");
  }
  public static void main (String [] args) {
    new Deadlock();
  }
}
```

o/p

```
Main Thread entered A.foo
Racing Thread entered B.bar
Main Thread trying to call B.last()
Racing Thread trying to call A.last()
```

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

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

OUTPUT:

```
D:\24BMSCE>javac Deadlock.java

D:\24BMSCE>java Deadlock
MainThread entered A.foo
RacingThread entered B.bar
RacingThread trying to call A.last
MainThread trying to call B.last
```