B.M.S. COLLEGE OF ENGINEERING
Basavanagudi, Bengaluru- 560019
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

LAB REPORT

On

Object Oriented Java Programming
(23CS3PCOOJ)

Submitted By:
ADITYA RAM S H
1BM22CS019

In partial fulfilment of
BACHELOR OF ENGINEERING
In
COMPUTER SCIENCE AND ENGINEERING
2023-24

Faculty-In-Charge
Swathi Sridharan

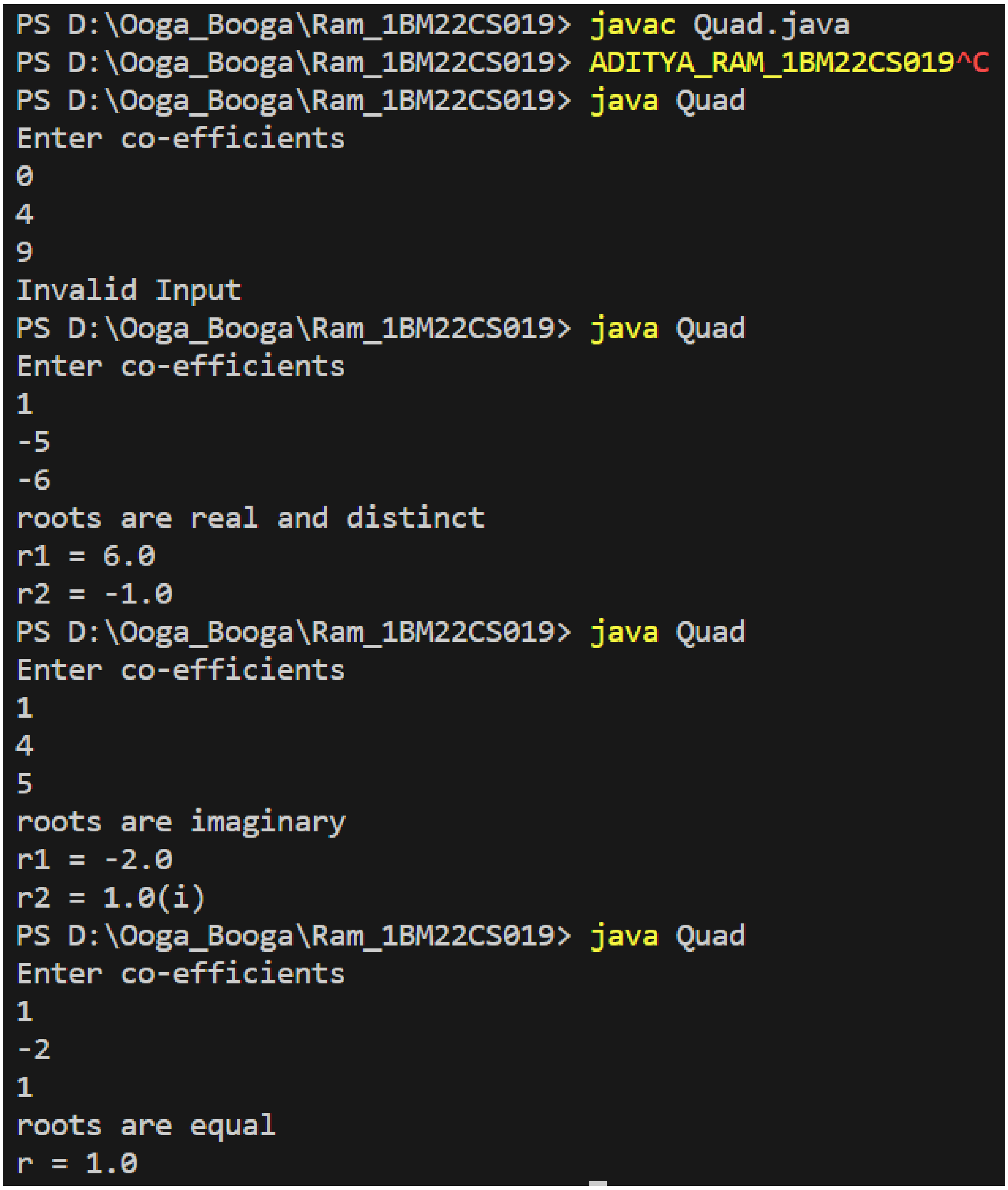
Assistant Professor
Department of Computer Science and Engineering

LAB-1: QUADRATIC EQUATION

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.

```
import java.util.Scanner;

public class Quad {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.println("Enter co-efficients");
        int a = in.nextInt();
        int b = in.nextInt();
        int c = in.nextInt();
        in.close();

        if (a == 0) {
            System.out.println("Invalid Input");
            return;
        }
        double r1, r2;
        double d = b * b - 4 * a * c;
        if (d > 0) {
            System.out.println("roots are real and distinct");
            r1 = (-b + Math.sqrt(d)) / (2 * a);
            r2 = (-b - Math.sqrt(d)) / (2 * a);
            System.out.println("r1 = " + r1 + "\n" + "r2 = " + r2);
        } else if (d < 0) {
            System.out.println("roots are imaginary");
            r1 = (-b / (2 * a));
            r2 = (Math.sqrt(Math.abs(d)) / (2 * a));
             System.out.println("r1 = " + r1 + "\n" + "r2 = " + r2 +
            "(i)");
        } else {
            System.out.println("roots are equal");
            r1 = (-b / (2 * a));
            System.out.println("r = " + r1);
        }

    }
}
```

OUTPUT :



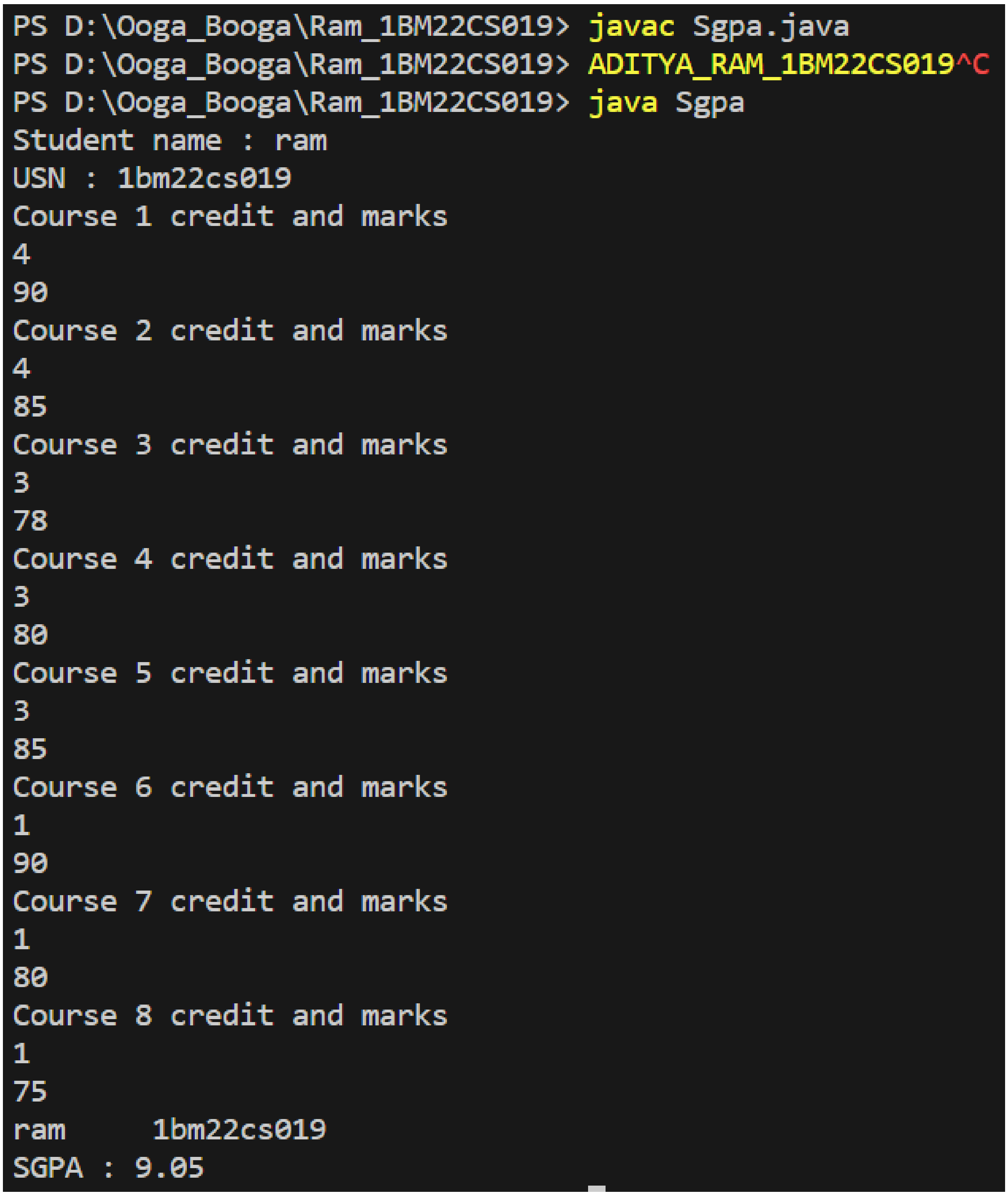
```
PS D:\Ooga_Booga\Ram_1BM22CS019> javac Quad.java
PS D:\Ooga_Booga\Ram_1BM22CS019> ADITYA_RAM_1BM22CS019^C
PS D:\Ooga_Booga\Ram_1BM22CS019> java Quad
Enter co-efficients
0
4
9
Invalid Input
PS D:\Ooga_Booga\Ram_1BM22CS019> java Quad
Enter co-efficients
1
-5
-6
roots are real and distinct
r1 = 6.0
r2 = -1.0
PS D:\Ooga_Booga\Ram_1BM22CS019> java Quad
Enter co-efficients
1
4
5
roots are imaginary
r1 = -2.0
r2 = 1.0(i)
PS D:\Ooga_Booga\Ram_1BM22CS019> java Quad
Enter co-efficients
1
-2
1
roots are equal
r = 1.0
```

LAB-2: STUDENT SGPA CALCULATION

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.

```
import java.util.Scanner;

class Student {
    String name, usn;
    int[] credits = new int[8];
    int[] marks = new int[8];

    public void details() {
        Scanner reader = new Scanner(System.in);
        System.out.print("Student name : ");
        name = reader.nextLine();
        System.out.print("USN : ");
        usn = reader.nextLine();
        for (int i = 0; i < 8; i++) {
            System.out.println("Course " + (i + 1) + " credit and 
marks");
            credits[i] = reader.nextInt();
            marks[i] = reader.nextInt();
        }
        reader.close();
    }

    public double sgpa() {
        int totalcredits = 0, gradepoint = 0, sum = 0;
        for (int i = 0; i < 8; i++) {
            totalcredits += credits[i];
            gradepoint = marks[i] / 10 + 1;
            if (gradepoint == 11) {
                gradepoint = 10;
            } else if (gradepoint <= 4) {
                gradepoint = 0;
            }
            sum += gradepoint * credits[i];
        }
        return (double) sum / totalcredits;
    }
}

public class Sgpa{
    public static void main(String[] args){
        Student student = new Student();
        student.details();
        System.out.println(student.name + "\t" + student.usn);
        System.out.println("SGPA : " + student.sgpa());
    }
}
```

OUTPUT :



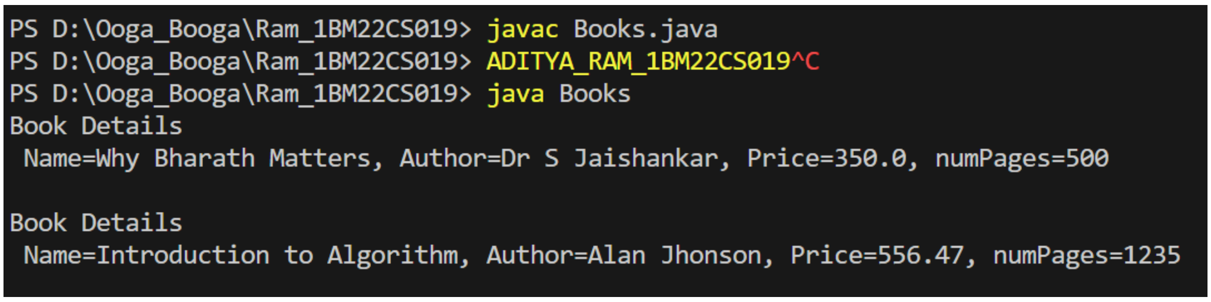
```
PS D:\Ooga_Booga\Ram_1BM22CS019> javac Sgpa.java
PS D:\Ooga_Booga\Ram_1BM22CS019> ADITYA_RAM_1BM22CS019^C
PS D:\Ooga_Booga\Ram_1BM22CS019> java Sgpa
Student name : ram
USN : 1bm22cs019
Course 1 credit and marks
4
90
Course 2 credit and marks
4
85
Course 3 credit and marks
3
78
Course 4 credit and marks
3
80
Course 5 credit and marks
3
85
Course 6 credit and marks
1
90
Course 7 credit and marks
1
80
Course 8 credit and marks
1
75
ram     1bm22cs019
SGPA : 9.05
```

LAB-3: BOOK DETAILS

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.

```
class Book {
    private String name;
    private String author;
    private double price;
    private int numPages;

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

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

    public void setAuthor(String author) {
        this.author = author;
    }

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

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

    public String getName() {
        return name;
    }

    public String getAuthor() {
        return author;
    }

    public double getPrice() {
        return price;
    }
    public int getNumPages() {
        return numPages;
    }

    @Override
    public String toString() {
        return "Book Details\n Name=" + name + ", Author=" + author +
", Price=" + price + ", numPages=" + numPages + "\n";
    }
}

public class Books{
    public static void main(String[] args) {
        int n = 2;
        Book[] books = new Book[n];
        books[0] = new Book("Why Bharath Matters", "Dr S Jaishankar",
350, 500);
        books[1] = new Book("Introduction to Algorithm", "Alan
Jhonson", 556.47, 1235);
        for (Book b : books) {
            System.out.println(b);
        }
    }
}
```

OUTPUT :



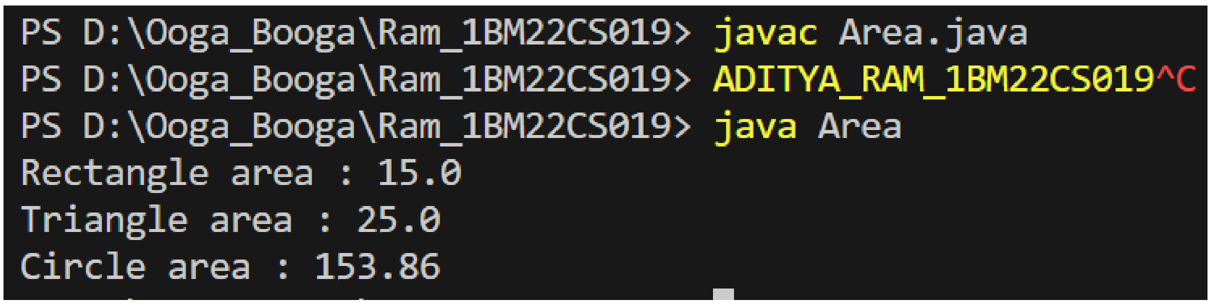
```
PS D:\Ooga_Booga\Ram_1BM22CS019> javac Books.java
PS D:\Ooga_Booga\Ram_1BM22CS019> ADITYA_RAM_1BM22CS019^C
PS D:\Ooga_Booga\Ram_1BM22CS019> java Books
Book Details
 Name=Why Bharath Matters, Author=Dr S Jaishankar, Price=350.0, numPages=500

Book Details
 Name=Introduction to Algorithm, Author=Alan Jhonson, Price=556.47, numPages=1235
```

LAB-4: AREA CALCULATION

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.

```
abstract class Shape {
    double x, y;

    abstract void print_area();
}

class Rectangle extends Shape {
    Rectangle(double x, double y) {
        this.x = x;
        this.y = y;
    }

    public void print_area() {
        System.out.println("Rectangle area : " + x * y);
    }
}

class Triangle extends Shape {
    Triangle(double x, double y) {
        this.x = x;
        this.y = y;
    }

    public void print_area() {
        System.out.println("Triangle area : " + x * y * 0.5);
    }
}

class Circle extends Shape {
    Circle(double x) {
        this.x = x;
    }

    public void print_area() {
        System.out.println("Circle area : " + 3.14 * x * x);
    }
}

public class Area {
    public static void main(String[] args) {
        Rectangle r = new Rectangle(5, 3);
        Triangle t = new Triangle(10, 5);
        Circle c = new Circle(7);

        r.print_area();
        t.print_area();
        c.print_area();
    }
}
```

OUTPUT :



```
PS D:\Ooga_Booga\Ram_1BM22CS019> javac Area.java
PS D:\Ooga_Booga\Ram_1BM22CS019> ADITYA_RAM_1BM22CS019^C
PS D:\Ooga_Booga\Ram_1BM22CS019> java Area
Rectangle area : 15.0
Triangle area : 25.0
Circle area : 153.86
```

LAB-5: BANK ACCOUNT DETAILS

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

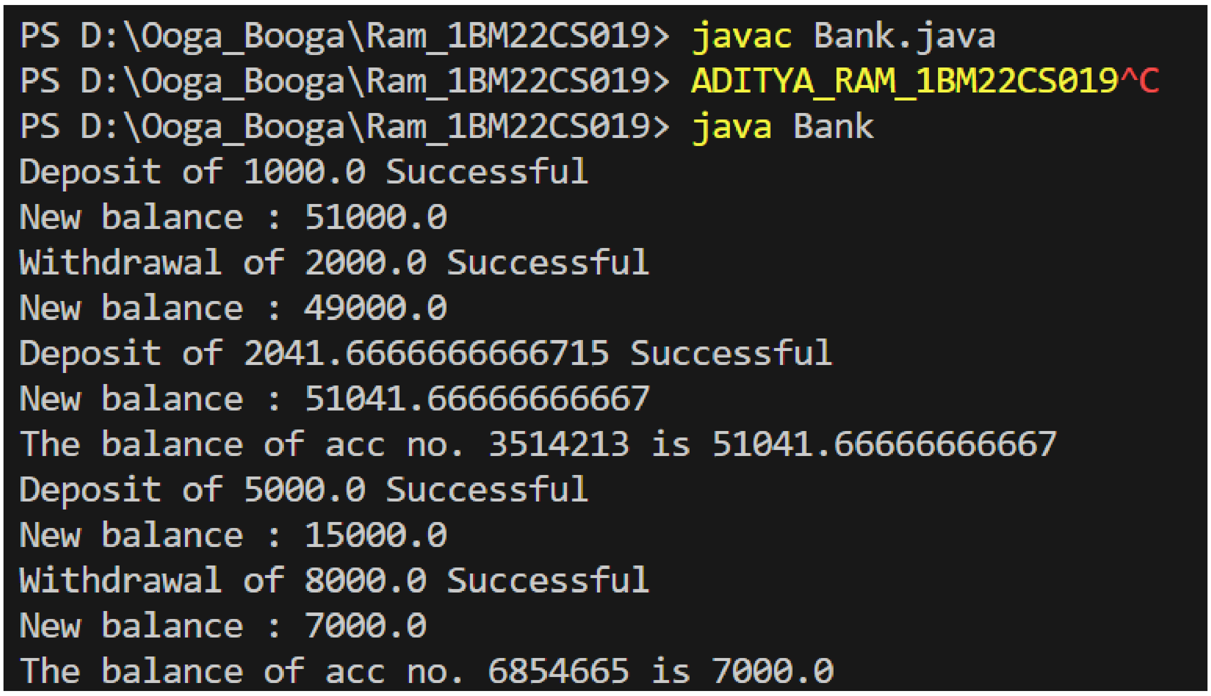
Check for the minimum balance, impose penalty if necessary and update the balance.

```
class Account {
    private String customerName;
    private int accountNumber;
    private String accountType;
    private double balance;

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

    public void setCustomerName(String customerName) {
        this.customerName = customerName;
    }

    public void setAccountNumber(int accountNumber) {
        this.accountNumber = accountNumber;
    }

    public void setAccountType(String accountType) {
        this.accountType = accountType;
    }

    public void setBalance(double balance) {
        this.balance = balance;
    }

    public String getCustomerName() {
        return customerName;
    }

    public int getAccountNumber() {
        return accountNumber;
    }

    public String getAccountType() {
        return accountType;
    }

    public double getBalance() {
        return balance;
    }

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

    public void displayBalance() {
        System.out.println("The balance of acc no. " + accountNumber +
" is " + balance);
    }

    public void compoundInterest() {
    }

    public void withdraw(double amount) {
        if (amount > 0 && amount <= balance) {
            balance -= amount;
            System.out.println("Withdrawal of " + amount + "
Successful");
            System.out.println("New balance : " + balance);
            return;
        } else if (amount > balance) {
            System.out.println("Insufficient Balance");
        } else if (amount <= 0) {
            System.out.println("Invalid amount");
        }
        System.out.println("Withdraw Failed");
    }

    public void checkBalance() {
    }
```

```
}

class sav_acct extends Account {
    private double interestRate;
    private double interest;

    public sav_acct(String customerName, int accountNumber, String
accountType, double balance, double interestRate) {
        super(customerName, accountNumber, accountType, balance);
        this.interestRate = interestRate;
    }

    public void setInterestRate(double interestRate) {
        this.interestRate = interestRate;
    }

    public void setInterest(double interest) {
        this.interest = interest;
    }

    public double getInterestRate() {
        return interestRate;
    }

    public double getInterest() {
        return interest;
    }

    public void compoundInterest() {
        interest = getBalance() * (1 + (0.5 / 12)) - getBalance();
        deposit(interest);
    }
}

class cur_acct extends Account {
    private double minBalance;
    private double serviceCharge;

    public cur_acct(String customerName, int accountNumber, String
accountType, double balance, double serviceCharge,
            double minBalance) {
        super(customerName, accountNumber, accountType, balance);
        this.minBalance = minBalance;
        this.serviceCharge = serviceCharge;
    }

    public void setMinBalance(double minBalance) {
        this.minBalance = minBalance;
    }

    public void setServiceCharge(double serviceCharge) {
        this.serviceCharge = serviceCharge;
    }

    public double getMinBalance() {
        return minBalance;
    }
```

```java
    public double getServiceCharge() {
        return serviceCharge;
    }

    public void checkBalance() {
        if (getBalance() < minBalance) {
            setBalance(getBalance() - serviceCharge);
            System.out.println("Low balance warning!");
        }
    }
}

public class Bank {
    public static void main(String[] args) {
        sav_acct sav = new sav_acct("Amba singh", 3514213, "savings", 
50000, 5);
        sav.deposit(1000);
        sav.withdraw(2000);
        sav.compoundInterest();
        sav.displayBalance();

        cur_acct cur = new cur_acct("Loli Prakash", 6854665, "current", 
10000, 500, 1000);
        cur.deposit(5000);
        cur.withdraw(8000);
        cur.checkBalance();
        cur.displayBalance();
    }
}
```

OUTPUT :



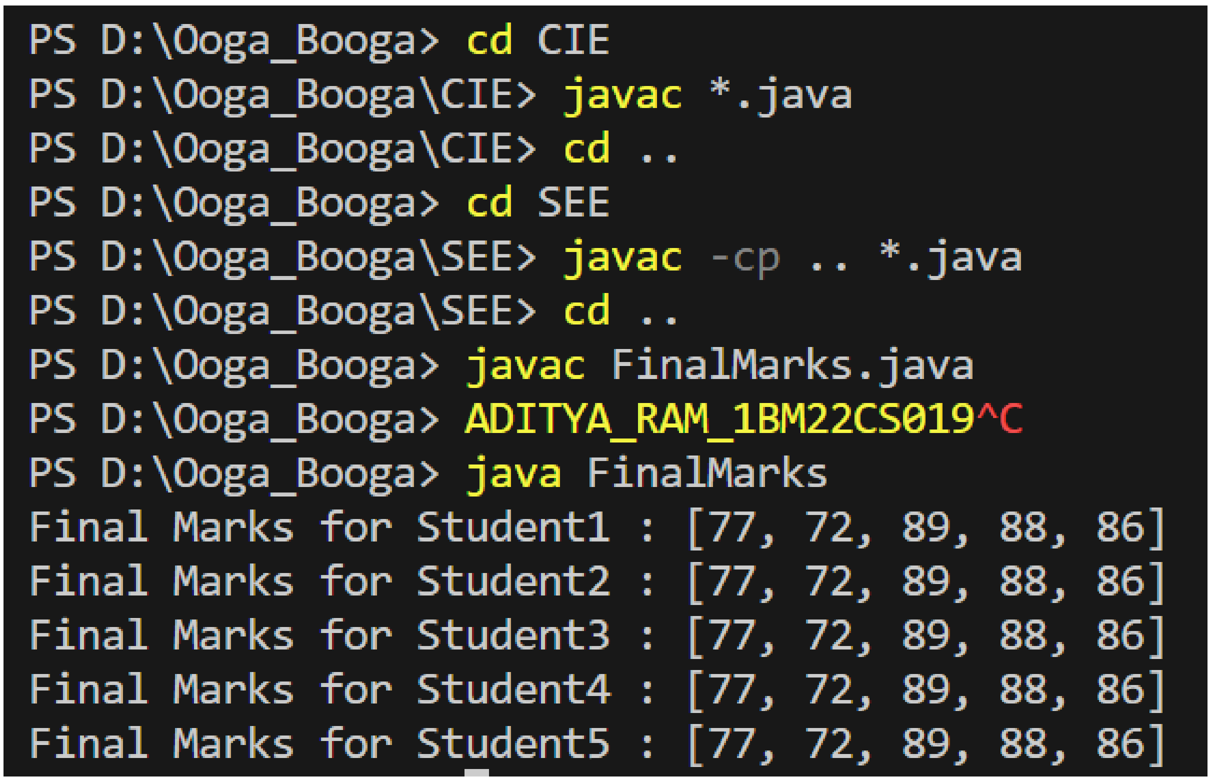
```
PS D:\Ooga_Booga\Ram_1BM22CS019> javac Bank.java
PS D:\Ooga_Booga\Ram_1BM22CS019> ADITYA_RAM_1BM22CS019^C
PS D:\Ooga_Booga\Ram_1BM22CS019> java Bank
Deposit of 1000.0 Successful
New balance : 51000.0
Withdrawal of 2000.0 Successful
New balance : 49000.0
Deposit of 2041.6666666666715 Successful
New balance : 51041.66666666667
The balance of acc no. 3514213 is 51041.66666666667
Deposit of 5000.0 Successful
New balance : 15000.0
Withdrawal of 8000.0 Successful
New balance : 7000.0
The balance of acc no. 6854665 is 7000.0
```

LAB-6: CALCULATION OF MARKS

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.

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


package CIE;

public class Internals extends Student {
    public int[] marks = new int[5];
    public Internals(String name,String usn,int sem,int[] marks){
        super(name,usn,sem);
        this.marks = marks;
    }
}


package SEE;
import CIE.Student;

public class Externals extends Student{
    public int[] marks  = new int[5];
    public Externals(String name,String usn,int sem,int[] marks){
        super(name,usn,sem);
        this.marks = marks;
    }
}


import CIE.Internals;
import SEE.Externals;
import java.util.*;

public class FinalMarks {
    public static void main(String[] args) {
        int n = 5;
        Internals[] I = new Internals[n];
        Externals[] E = new Externals[n];

        for (int i = 0; i < n; i++) {
            I[i] = new Internals();
            E[i] = new Externals();

            I[i].usn = "usn" + (i + 1);
            I[i].name = "Student" + (i + 1);
            I[i].sem = 3;
            I[i].marks = new int[] { 80, 75, 90, 85, 88 };

            E[i].usn = "usn" + (i + 1);
            E[i].name = "Student" + (i + 1);
            E[i].sem = 3;
            E[i].marks = new int[] { 75, 70, 88, 92, 85 };
        }
        for (int i = 0; i < n; i++) {
            int[] finalMarks = new int[5];
            for (int j = 0; j < 5; j++) {
                finalMarks[j] = (I[i].marks[j] + E[i].marks[j]) / 2;
            }
            System.out.println("Final Marks for " + I[i].name + " : " +
Arrays.toString(finalMarks));
        }
    }
}
```

OUTPUT :



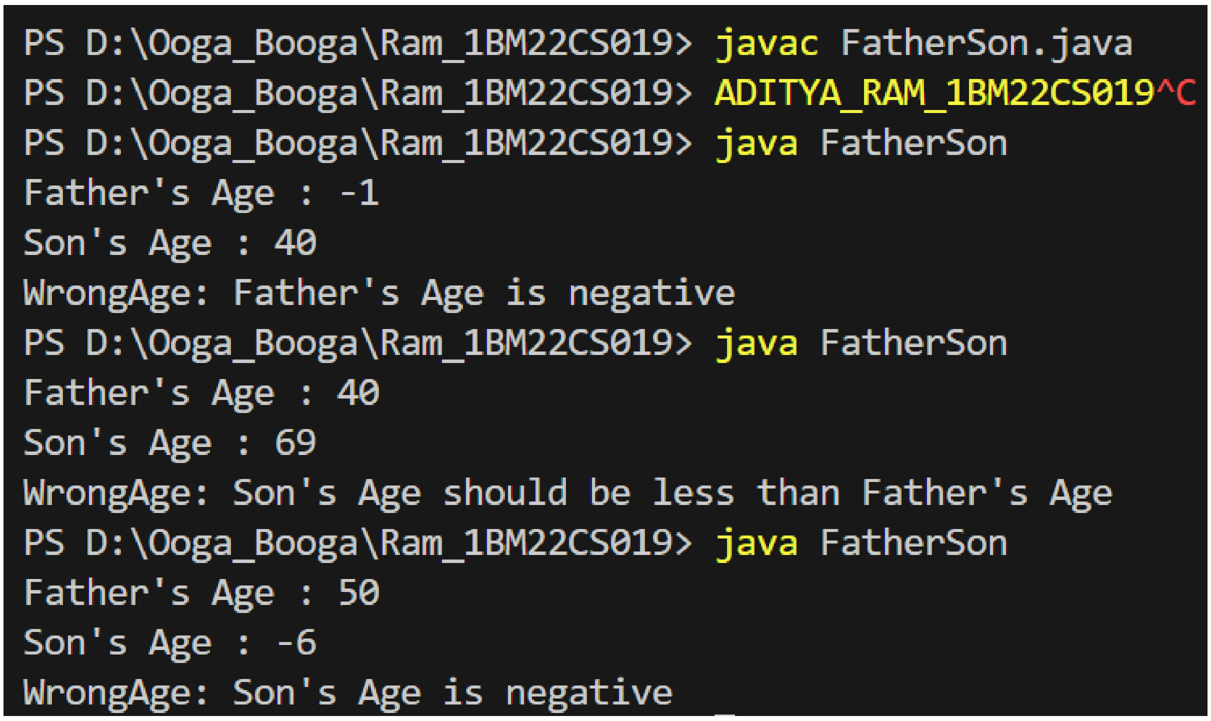
```
PS D:\Ooga_Booga> cd CIE
PS D:\Ooga_Booga\CIE> javac *.java
PS D:\Ooga_Booga\CIE> cd ..
PS D:\Ooga_Booga> cd SEE
PS D:\Ooga_Booga\SEE> javac -cp .. *.java
PS D:\Ooga_Booga\SEE> cd ..
PS D:\Ooga_Booga> javac FinalMarks.java
PS D:\Ooga_Booga> ADITYA_RAM_1BM22CS019^C
PS D:\Ooga_Booga> java FinalMarks
Final Marks for Student1 : [77, 72, 89, 88, 86]
Final Marks for Student2 : [77, 72, 89, 88, 86]
Final Marks for Student3 : [77, 72, 89, 88, 86]
Final Marks for Student4 : [77, 72, 89, 88, 86]
Final Marks for Student5 : [77, 72, 89, 88, 86]
```

LAB-7: EXCEPTION HANDLING

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

```
import java.util.Scanner;

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

class Father {
    public int FatherAge;

    Father(int FatherAge) throws WrongAge {
        if (FatherAge < 0)
            throw new WrongAge("Father's Age is negative");
        this.FatherAge = FatherAge;
    }
}

class Son extends Father {
    public int SonAge;

    Son(int SonAge, int FatherAge) throws WrongAge {
        super(FatherAge);
        if (SonAge >= FatherAge)
            throw new WrongAge("Son's Age should be less than Father's
Age");
        if (SonAge < 0)
            throw new WrongAge("Son's Age is negative");
        this.SonAge = SonAge;
    }
}

public class FatherSon {
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        int f_age, s_age;
        try {
            System.out.print("Father's Age : ");
            f_age = s.nextInt();
            System.out.print("Son's Age : ");
            s_age = s.nextInt();
            try {
                Son son = new Son(s_age, f_age);
            } catch (WrongAge wa) {
                System.out.println(wa);
            }
        } catch (Exception e) {
            System.out.println(e);
        }
    }
}
```

OUTPUT :



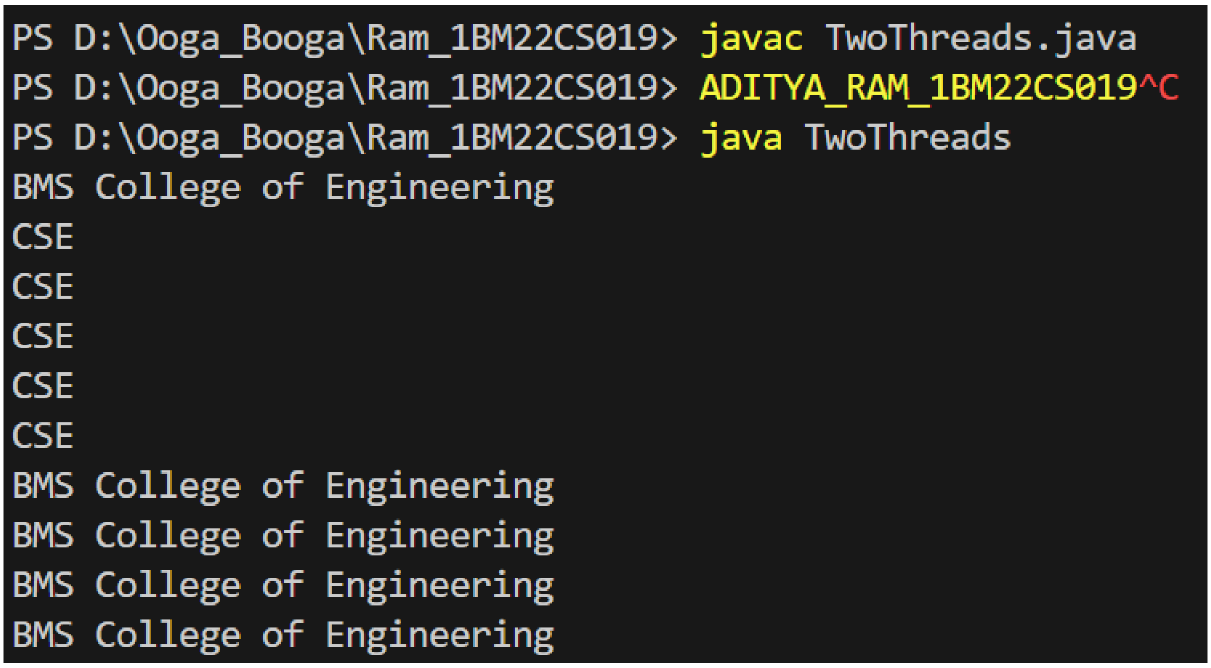
```
PS D:\Ooga_Booga\Ram_1BM22CS019> javac FatherSon.java
PS D:\Ooga_Booga\Ram_1BM22CS019> ADITYA_RAM_1BM22CS019^C
PS D:\Ooga_Booga\Ram_1BM22CS019> java FatherSon
Father's Age : -1
Son's Age : 40
WrongAge: Father's Age is negative
PS D:\Ooga_Booga\Ram_1BM22CS019> java FatherSon
Father's Age : 40
Son's Age : 69
WrongAge: Son's Age should be less than Father's Age
PS D:\Ooga_Booga\Ram_1BM22CS019> java FatherSon
Father's Age : 50
Son's Age : -6
WrongAge: Son's Age is negative
```

LAB-8: MULTI THREADING

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 A extends Thread {
    public void run() {
        try {
            for (int i = 0; i < 5; i++) {
                System.out.println("BMS College of Engineering");
                Thread.sleep(10000);
            }
        } catch (InterruptedException ie) {
        }
    }
}

class B extends Thread {
    public void run() {
        try {
            for (int i = 0; i < 5; i++) {
                System.out.println("CSE");
                Thread.sleep(2000);
            }
        } catch (InterruptedException ie) {
        }
    }
}

public class TwoThreads {
    public static void main(String[] args) {
        A a = new A();
        B b = new B();
        a.start();
        b.start();
    }
}
```

OUTPUT:



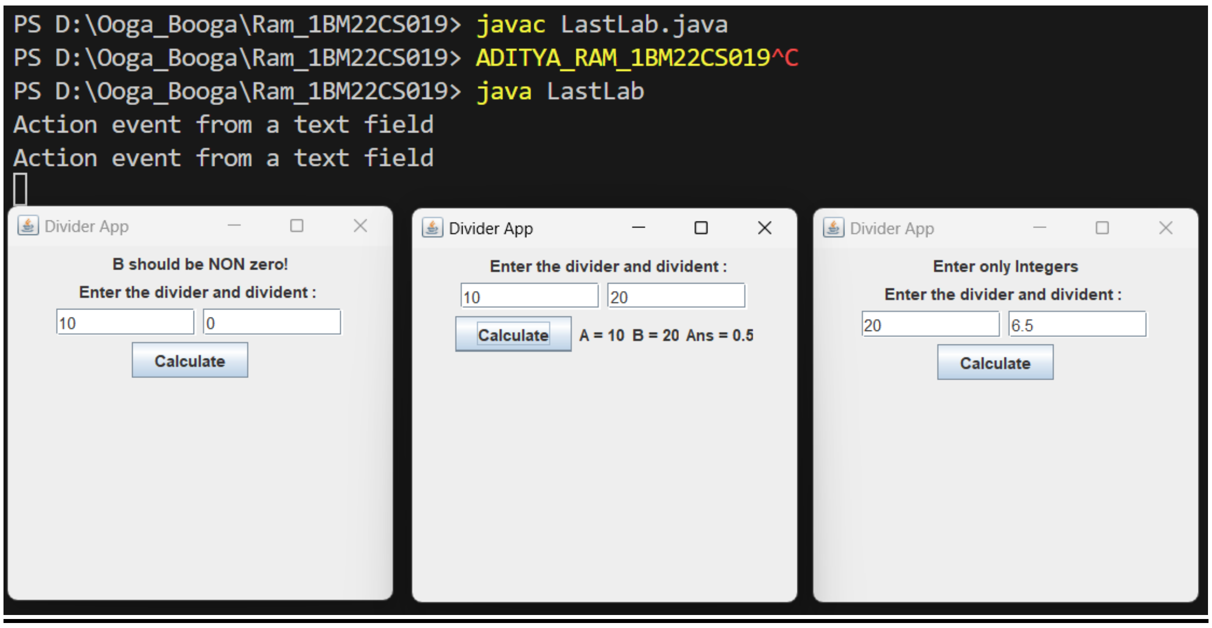
```
PS D:\Ooga_Booga\Ram_1BM22CS019> javac TwoThreads.java
PS D:\Ooga_Booga\Ram_1BM22CS019> ADITYA_RAM_1BM22CS019^C
PS D:\Ooga_Booga\Ram_1BM22CS019> java TwoThreads
BMS College of Engineering
CSE
CSE
CSE
CSE
CSE
BMS College of Engineering
BMS College of Engineering
BMS College of Engineering
BMS College of Engineering
```

LAB-9: AWT

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 Num2 were Zero, the program would throw an Arithmetic Exception. Display the exception in a message dialog box.

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

public class LastLab {
    public LastLab() {
        JFrame jfrm = new JFrame("Divider App");
        jfrm.setSize(300, 300);
        jfrm.setLayout(new FlowLayout());
        jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JLabel jlab = new JLabel("Enter the divider and divident : ");

        JTextField ajtf = new JTextField(10);
        JTextField bjtf = new JTextField(10);

        JButton button = new JButton("Calculate");

        JLabel err = new JLabel();
        JLabel alab = new JLabel();
        JLabel blab = new JLabel();
        JLabel anslab = new JLabel();

        jfrm.add(err);
        jfrm.add(jlab);
        jfrm.add(ajtf);
        jfrm.add(bjtf);
        jfrm.add(button);
        jfrm.add(alab);
        jfrm.add(blab);
        jfrm.add(anslab);

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

        button.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent evt) {
                try {
                    int a = Integer.parseInt(ajtf.getText());
                    int b = Integer.parseInt(bjtf.getText());
                    double ans = a / (double) b;
                    if(b==0) throw new ArithmeticException();
                    alab.setText("\nA = " + a);
                    blab.setText("\nB = " + b);
                    anslab.setText("\nAns = " + ans);
                } catch (NumberFormatException e) {
                    alab.setText("");
                    blab.setText("");
                    anslab.setText("");
                    err.setText("Enter only Integers\n");
                } catch (ArithmeticException e) {
                    alab.setText("");
                    blab.setText("");
                    anslab.setText("");
                    err.setText("B should be NON zero!\n");
                }
            }
        });
        jfrm.setVisible(true);
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                new LastLab();
            }
        });
    }
}
```

OUTPUT:



```
PS D:\Ooga_Booga\Ram_1BM22CS019> javac LastLab.java
PS D:\Ooga_Booga\Ram_1BM22CS019> ADITYA_RAM_1BM22CS019^C
PS D:\Ooga_Booga\Ram_1BM22CS019> java LastLab
Action event from a text field
Action event from a text field
```

Divider App
B should be NON zero!
Enter the divider and divident :
10 | 0
Calculate

Divider App
Enter the divider and divident :
10 | 20
Calculate A = 10 B = 20 Ans = 0.5

Divider App
Enter only Integers
Enter the divider and divident :
20 | 6.5
Calculate