NAME: NEELVANI VARSHA VITTAL
USN: 2023BMS02551
SECTION: 3C

JAVA LAB RECORD

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.

```
import java.util.Scanner;

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

      System.out.println("Enter the coefficients of the quadratic equation (ax^2 + bx + c =
0):");
      System.out.print("a: ");
      int a = s.nextInt();
      System.out.print("b: ");
      int b = s.nextInt();
      System.out.print("c: ");
      int c = s.nextInt();

      int dt = b * b - 4 * a * c;

      if (dt > 0) {
        System.out.println("It has real and distinct roots");
         double r1 = (-b + Math.sqrt(dt)) / (2 * a);
         double r2 = (-b - Math.sqrt(dt)) / (2 * a);
         System.out.println("The roots are " +r1+ " and " +r2);
      } else if (dt == 0) {
        System.out.println("It has real and equal roots");
         double rt = -b / (2 * a);
         System.out.println("The roots are " +rt+ " and " +rt);
      } else {
         System.out.println("It has no real roots.");
      }

      s.close();
   }
}
```

```
C:\Users\dell3\OneDrive\Desktop\3rd sem\OOJ\Record>java Quadratic
Enter the coefficients of the quadratic equation (ax^2 + bx + c = 0):
a: 1
b: 2
c: 1
It has real and equal roots
The roots are -1.0 and -1.0

C:\Users\dell3\OneDrive\Desktop\3rd sem\OOJ\Record>java Quadratic
Enter the coefficients of the quadratic equation (ax^2 + bx + c = 0):
a: 1
b: 1
c: 4
It has no real roots.

C:\Users\dell3\OneDrive\Desktop\3rd sem\OOJ\Record>java Quadratic
Enter the coefficients of the quadratic equation (ax^2 + bx + c = 0):
a: 1
b: 4
c: 1
It has real and distinct roots
The roots are -0.2679491924311228 and -3.732050807568877
```

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.

```
import java.util.Scanner;
class Student {
    String usn;
    String name;
    int credits[] = new int[5];
    float marks[] = new float[5];
    void studentdetails(String usn, String name, int credits[], float marks[]) {
        this.usn = usn;
        this.name = name;
        this.credits = credits;
        this.marks = marks;
    }
    void printdetails() {
        System.out.println("The usn of student is " + usn);
        System.out.println("The name of student is " + name);
        System.out.println("The credits of subjects are:");
```

```
    for (int i = 0; i < 5; i++) {
        System.out.println(credits[i]);
    }
    System.out.println("The marks of the student are:");
    for (int i = 0; i < 5; i++) {
        System.out.println(marks[i]);
    }
}
float studentsgpa() {
    float sum = 0;
    int creditssum = 0;
    for (int i = 0; i < 5; i++) {
        if (marks[i] >= 90) {
            sum = sum + (credits[i] * 10);
        } else if (marks[i] >= 80) {
            sum = sum + (credits[i] * 9);
        } else if (marks[i] >= 70) {
            sum = sum + (credits[i] * 8);
        } else if (marks[i] >= 60) {
            sum = sum + (credits[i] * 7);
        } else if (marks[i] >= 50) {
            sum = sum + (credits[i] * 6);
        } else if (marks[i] >= 35) {
            sum = sum + (credits[i] * 5);
        } else {
            sum = sum + (credits[i] * 0);
        }
        creditssum = creditssum + credits[i];
    }
    float sgpa = sum / (float) creditssum;
```

```
        return sgpa;
    }
}
public class StudentSGPA {
    public static void main(String xx[]) {
        String usn;
        String name;
        int credits[] = new int[5];
        float marks[] = new float[5];
        float sgpa;
        Student s = new Student();
        Scanner s1 = new Scanner(System.in);
        System.out.println("Enter the usn of student");
        usn = s1.next();
        System.out.println("Enter the name of student");
        name = s1.next();
        System.out.println("Enter the credits of subject");
        for (int i = 0; i < 5; i++) {
            credits[i] = s1.nextInt();
        }
        System.out.println("Enter the marks of student");
        for (int i = 0; i < 5; i++) {
            marks[i] = s1.nextFloat();
        }
        s.studentdetails(usn, name, credits, marks);
        s.printdetails();
        sgpa = s.studentsgpa();
        System.out.println("The sgpa of student is " + sgpa);
    }
}
```

```
Enter the usn of student
2023BMS02551
Enter the name of student
Aarohi
Enter the credits of subject
4 4 3 3 4
Enter the marks of student
65 78 85 89 95
The usn of student is 2023BMS02551
The name of student is Aarohi
The credits of subjects are:
4
4
3
3
4
The marks of the student are:
65.0
78.0
85.0
89.0
95.0
The sgpa of student is 8.555555
```

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.

```
import java.util.Scanner;
public class Book {
    String name;
    String author;
    double price;
    int numPages;
    public Book(String name, String author, double price, int numPages) {
        this.name = name;
        this.author = author;
        this.price = price;
        this.numPages = numPages;
    }
    public String getName() {
```

```
    return name;
}
public String getAuthor() {
    return author;
}
public double getPrice() {
    return price;
}
public int getNumPages() {
    return numPages;
}
public void 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 toString() {
    return "Book Details:\n" +
            "Name: " + name + "\n" +
            "Author: " + author + "\n" +
            "Price: " + price + "\n" +
            "Number of Pages: " + numPages;
}
```

```
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        System.out.print("Enter the number of books: ");
        int numBooks = s.nextInt();
        s.nextLine();
        Book[] books = new Book[numBooks];
        for (int i = 0; i < numBooks; i++) {
            System.out.println("\nEnter details for Book " + (i + 1) + ":");
            System.out.print("Name: ");
            String name = s.nextLine();
            System.out.print("Author: ");
            String author = s.nextLine();
            System.out.print("Price: ");
            double price = s.nextDouble();
            System.out.print("Number of Pages: ");
            int numPages = s.nextInt();
            s.nextLine();
            books[i] = new Book(name, author, price, numPages);
        }
        System.out.println("\nBook details:");
        for (int i = 0; i < numBooks; i++) {
            System.out.println("\nBook " + (i + 1) + ":");
            System.out.println(books[i].toString());
        }
        s.close();
    }
}
```

```
Enter the number of books: 3

Enter details for Book 1:
Name: The Thousand Splendid Suns
Author: Khaled Hoseinni
Price: 400
Number of Pages: 450

Enter details for Book 2:
Name: To Kill A Mocking Bird
Author: Harper Lee
Price: 450
Number of Pages: 400

Enter details for Book 3:
Name: Norwegian Wood
Author: Haruki Murakami
Price: 500
Number of Pages: 353

Book details:

Book 1:
Book Details:
Name: The Thousand Splendid Suns
Author: Khaled Hoseinni
Price: 400.0
Number of Pages: 450

Book 2:
Book Details:
Name: To Kill A Mocking Bird
Author: Harper Lee
Price: 450.0
Number of Pages: 400

Book 3:
Book Details:
Name: Norwegian Wood
Author: Haruki Murakami
Price: 500.0
Number of Pages: 353
```

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.

```
abstract class Shape {
    int d1;
    int d2;
    abstract void printArea();
}
class Rectangle extends Shape {
    Rectangle(int length, int width) {
        d1 = length;
        d2 = width;
    }
    @Override
    void printArea() {
        int area = dimension1 * dimension2;
```

```
        System.out.println("Area of Rectangle: " + area);
    }
}
class Triangle extends Shape {
    Triangle(int base, int height) {
        dimension1 = base;
        dimension2 = height;
    }
    @Override
    void printArea() {
        double area = 0.5 * dimension1 * dimension2;
        System.out.println("Area of Triangle: " + area);
    }
}
class Circle extends Shape {
    Circle(int radius) {
        dimension1 = radius;
    }
    @Override
    void printArea() {
        double area = Math.PI * dimension1 * dimension1;
        System.out.println("Area of Circle: " + area);
    }
}
public class ShapeMain {
    public static void main(String[] args) {
        Rectangle rectangle = new Rectangle(5, 10);
        Triangle triangle = new Triangle(3, 6);
        Circle circle = new Circle(4);
        rectangle.printArea();
```

```
        triangle.printArea();
        circle.printArea();
    }
}
```

```
Area of Rectangle: 50
Area of Triangle: 9.0
Area of Circle: 50.26548245743669
```

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.

```
import java.util.Scanner;

class Account {
    String custName;
    long accNum;
    String accType;
    double balance;

    public Account(String custName, long accNum, String accType, double balance) {
        this.custName = custName;
        this.accNum = accNum;
        this.accType = accType;
        this.balance = balance;
    }
```

```
    public void deposit(double amount) {
        balance += amount;
        System.out.println("Deposit successful. New balance: " + balance);
    }

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

class CurAccount extends Account {
    double minBalance;
    double serviceCharge;

    public CurAccount(String custName, long accNum, String accType, double balance,
double minBalance, double serviceCharge) {
        super(custName, accNum, accType, balance);
        this.minBalance = minBalance;
        this.serviceCharge = serviceCharge;
    }

    @Override
    public void deposit(double amount) {
        super.deposit(amount);
    }

    @Override
    public void displayBalance() {
        super.displayBalance();
    }

    public void withdraw(double amount) {
        if (balance - amount >= minBalance) {
            balance -= amount;
            System.out.println("Withdrawal successful. New balance: " + balance);
        } else {
            System.out.println("Insufficient funds. Withdrawal failed.");
        }
    }

    public void imposeServiceCharge() {
        balance -= serviceCharge;
        System.out.println("Service charge imposed. New balance: " + balance);
    }
}
```

```
    class SavAccount extends Account {
        double interestRate;

        public SavAccount(String custName, long accNum, String accType, double balance,
double interestRate) {
            super(custName, accNum, accType, balance);
            this.interestRate = interestRate;
        }

        @Override
        public void deposit(double amount) {
            super.deposit(amount);
        }

        @Override
        public void displayBalance() {
            super.displayBalance();
        }

        public void computeAndDepositInterest() {
            double interest = balance * interestRate / 100;
            balance += interest;
            System.out.println("Interest computed and deposited. New balance: " + balance);
        }

        public void withdraw(double amount) {
            if (balance - amount >= 0) {
                balance -= amount;
                System.out.println("Withdrawal successful. New balance: " + balance);
            } else {
                System.out.println("Insufficient funds. Withdrawal failed.");
            }
        }
    }

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

            CurAccount currentAccount = new CurAccount("Avni Khanna", 123456789,
"Current", 1000, 500, 10);
            SavAccount savingsAccount = new SavAccount("Neil Khanna", 987654321,
"Savings", 5000, 5);

            System.out.println("1. Current Account");
            System.out.println("2. Savings Account");
```

```
    System.out.print("Select account type: ");
    int choice = s.nextInt();

    switch (choice) {
        case 1:
            System.out.println("Current Account selected.");
            operateCurrentAccount(currentAccount, s);
            break;
        case 2:
            System.out.println("Savings Account selected.");
            operateSavingsAccount(savingsAccount, s);
            break;
        default:
            System.out.println("Invalid choice.");
    }
}

public static void operateCurrentAccount(CurAccount currentAccount, Scanner s) {
    int option;

    do {
        System.out.println("\n1. Deposit");
        System.out.println("2. Withdraw");
        System.out.println("3. Display Balance");
        System.out.println("4. Exit");
        System.out.print("Select option: ");
        option = s.nextInt();

        switch (option) {
            case 1:
                System.out.print("Enter deposit amount: ");
                double depositAmount = s.nextDouble();
                currentAccount.deposit(depositAmount);
                break;
            case 2:
                System.out.print("Enter withdrawal amount: ");
                double withdrawAmount = s.nextDouble();
              currentAccount.withdraw(withdrawAmount);
                if (currentAccount.balance < currentAccount.minBalance) {
                    currentAccount.imposeServiceCharge();
                }
                break;
            case 3:
                currentAccount.displayBalance();
                break;
            case 4:
```

```
                System.out.println("Exiting current account operations.");
                break;
            default:
                System.out.println("Invalid option.");
        }
    } while (option != 4);
}

public static void operateSavingsAccount(SavAccount savingsAccount, Scanner s) {
    int option;

    do {
        System.out.println("\n1. Deposit");
        System.out.println("2. Withdraw");
        System.out.println("3. Display Balance");
        System.out.println("4. Compute and Deposit Interest");
        System.out.println("5. Exit");
        System.out.print("Select option: ");
        option = s.nextInt();

        switch (option) {
            case 1:
                System.out.print("Enter deposit amount: ");
                double depositAmt = s.nextDouble();
                savingsAccount.deposit(depositAmt);
                break;
            case 2:
                System.out.print("Enter withdrawal amount: ");
                double withdrawAmt = s.nextDouble();
                savingsAccount.withdraw(withdrawAmt);
                break;
            case 3:
                savingsAccount.displayBalance();
                break;
            case 4:
                savingsAccount.computeAndDepositInterest();
                break;
            case 5:
                System.out.println("Exiting savings account operations.");
                break;
            default:
                System.out.println("Invalid option.");
        }
    } while (option != 5);
}
}
```

```
1. Current Account
2. Savings Account
Select account type: 1
Current Account selected.

1. Deposit
2. Withdraw
3. Display Balance
4. Exit
Select option: 1
Enter deposit amount: 1500
Deposit successful. New balance: 2500.0

1. Deposit
2. Withdraw
3. Display Balance
4. Exit
Select option: 3
Balance: 2500.0

1. Deposit
2. Withdraw
3. Display Balance
4. Exit
Select option: 2
Enter withdrawal amount: 1250
Withdrawal successful. New balance: 1250.0

1. Deposit
2. Withdraw
3. Display Balance
4. Exit
Select option: 3
Balance: 1250.0

1. Deposit
2. Withdraw
3. Display Balance
4. Exit
Select option: 4
Exiting current account operations.
```

```
1. Current Account
2. Savings Account
Select account type: 2
Savings Account selected.

1. Deposit
2. Withdraw
3. Display Balance
4. Compute and Deposit Interest
5. Exit
Select option: 4
Interest computed and deposited. New balance: 5250.0

1. Deposit
2. Withdraw
3. Display Balance
4. Compute and Deposit Interest
5. Exit
Select option: 2
Enter withdrawal amount: 500
Withdrawal successful. New balance: 4750.0

1. Deposit
2. Withdraw
3. Display Balance
4. Compute and Deposit Interest
5. Exit
Select option: 3
Balance: 4750.0

1. Deposit
2. Withdraw
3. Display Balance
4. Compute and Deposit Interest
5. Exit
Select option: 5
Exiting savings account operations.
```

6] Create a package CIE which has two classes- Student and Internals. The class Student 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.

```
//CIE-Student.java;
package CIE;
public class Student {
    String usn;
    String name;
```

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

//CIE-Internals.java
package CIE;
public class Internals {
    int[] internalMarks;
    public Internals(int[] internalMarks) {
        this.internalMarks = internalMarks;
    }
}

//SEE-External.java
package SEE;
import CIE.Student;
public class External extends Student {
    int[] externalMarks;
    public External(String usn, String name, int sem, int[] externalMarks) {
        super(usn, name, sem);
        this.externalMarks = externalMarks;
    }
}

//MainProgram.java
import java.util.Scanner;
public class MainProgram {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of students: ");
        int n = scanner.nextInt();
        External[] students = new External[n];
        for (int i = 0; i < n; i++) {
            System.out.println("\nEnter details for Student " + (i + 1) + ":");
            System.out.print("Enter USN: ");
            String usn = scanner.next();
            System.out.print("Enter Name: ");
            String name = scanner.next();
            System.out.print("Enter Semester: ");
            int sem = scanner.nextInt();
            System.out.println("Enter Internal Marks for 5 courses:");
            int[] internalMarks = new int[5];
```

```
            for (int j = 0; j < 5; j++) {
                System.out.print("Course " + (j + 1) + ": ");
                internalMarks[j] = scanner.nextInt();
            }
            System.out.println("Enter External Marks for 5 courses:");
            int[] externalMarks = new int[5];
            for (int j = 0; j < 5; j++) {
                System.out.print("Course " + (j + 1) + ": ");
                externalMarks[j] = scanner.nextInt();
            }
            students[i] = new External(usn, name, sem, internalMarks, externalMarks);
        }
        scanner.close();
        displayFinalMarks(students);
    }
    private static void displayFinalMarks(External[] students) {
        System.out.println("\nFinal Marks for Students:");
        for (int i = 0; i < students.length; i++) {
            System.out.println("\nDetails for Student " + (i + 1) + ":");
            System.out.println("USN: " + students[i].usn);
            System.out.println("Name: " + students[i].name);
            System.out.println("Semester: " + students[i].sem);
            System.out.println("Internal Marks:");
            for (int j = 0; j < 5; j++) {
                System.out.println("Course " + (j + 1) + ": " + students[i].internalMarks[j]);
            }
            System.out.println("External Marks:");
            for (int j = 0; j < 5; j++) {
                System.out.println("Course " + (j + 1) + ": " + students[i].externalMarks[j]);
            }
        }
    }
}
```

```
Enter the number of students: 2

Enter details for Student 1:
Enter USN: USN001
Enter Name: Aarohi
Enter Semester: 3
Enter Internal Marks for 5 courses:
Course 1: 75
Course 2: 80
Course 3: 75
Course 4: 80
Course 5: 90
Enter External Marks for 5 courses:
Course 1: 80
Course 2: 85
Course 3: 80
Course 4: 85
Course 5: 90

Enter details for Student 2:
Enter USN: USN002
Enter Name: Neil
Enter Semester: 3
Enter Internal Marks for 5 courses:
Course 1: 80
Course 2: 85
Course 3: 80
Course 4: 85
Course 5: 95
Enter External Marks for 5 courses:
Course 1: 85
Course 2: 90
Course 3: 85
Course 4: 90
Course 5: 95
```

```
Final Marks, Percentage, and CGPA for Students:

Details for Student 1:
USN: USN001
Name: Aarohi
Semester: 3
Internal Marks:
Course 1: 75
Course 2: 80
Course 3: 75
Course 4: 80
Course 5: 90
External Marks:
Course 1: 80
Course 2: 85
Course 3: 80
Course 4: 85
Course 5: 90
Total Marks: 820
Percentage: 82%
CGPA: 8.2

Details for Student 2:
USN: USN002
Name: Neil
Semester: 3
Internal Marks:
Course 1: 80
Course 2: 85
Course 3: 80
Course 4: 85
Course 5: 95
External Marks:
Course 1: 85
Course 2: 90
Course 3: 85
Course 4: 90
Course 5: 95
Total Marks: 870
Percentage: 87%
CGPA: 8.7
```

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.

```
import java.util.Scanner;
class WrongAge extends Exception {
    public WrongAge(String message) {
        super(message);
    }
}
class Father {
    int age;
    public Father(int age) throws WrongAge {
        if (age < 0) {
            throw new WrongAge("Age cannot be negative.");
        }
        this.age = age;
    }
    public int getAge() {
        return age;
    }
}
class Son extends Father {
    int sonAge;
    public Son(int fatherAge, int sonAge) throws WrongAge {
        super(fatherAge);
        if (sonAge >= fatherAge) {
            throw new WrongAge("Son's age cannot be greater than or equal to the father's
age.");
        }
        this.sonAge = sonAge;
    }
    public int getSonAge() {
        return sonAge;
    }
}
public class ExceptInheriDemo {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        try {
            System.out.print("Enter Father's Age: ");
            int fatherAge = scanner.nextInt();
```

```
            Father father = new Father(fatherAge);
            System.out.print("Enter Son's Age: ");
            int sonAge = scanner.nextInt();
            Son son = new Son(father.getAge(), sonAge);
            System.out.println("Father's Age: " + father.getAge());
            System.out.println("Son's Age: " + son.getSonAge());
            System.out.print("Enter another Son's Age: ");
            int invalidSonAge = scanner.nextInt();
            Son invalidSon = new Son(father.getAge(), invalidSonAge);
        } catch (WrongAge e) {
            System.out.println("Exception caught: " + e.getMessage());
        } finally {
            scanner.close();
        }
    }
}
```

```
Enter Father's Age: 40
Enter Son's Age: 20
Father's Age: 40
Son's Age: 20
Enter another Son's Age: 45
Exception caught: Son's age cannot be greater than or equal to the father's age.
```

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.

```
class BMSCEthread implements Runnable{
    Thread t;
    BMSCEthread(){
        t = new Thread(this, "BMSThread");
        System.out.println("CT: " + t);
        t.start();
    }

    public void run(){
        try{
            for (int i=5; i>0; i--){
                System.out.println("BMS College of Engineering");
                Thread.sleep(10);
            }
        } catch(InterruptedException ie){
            System.out.println("BMSThread Interrupted");
        }
        System.out.println("BMSThread is quitting");
    }
}
```

```
class CSEThread{
   public static void main(String[] args) {
      new BMSCEthread();
      try{
         for (int i=5; i>0; i--){
            System.out.println("CSE");
            Thread.sleep(2);
         }
      } catch(InterruptedException ie){
         System.out.println("CSE Thread Interrupted");
      }
      System.out.println("CSE Thread is quitting");
   }
}
```

```
CT: Thread[#20,BMSThread,5,main]
CSE
BMS College of Engineering
CSE
CSE
CSE
CSE
BMS College of Engineering
CSE Thread is quitting
BMS College of Engineering
BMS College of Engineering
BMS College of Engineering
BMSThread is quitting
```

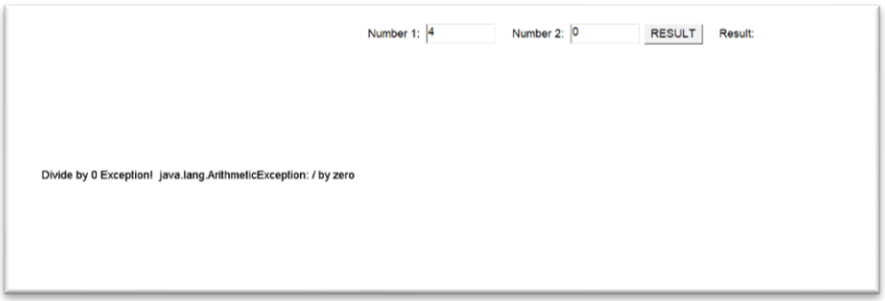
9] write a program thar creates a user interface to perform integer divisions. the user enters two numbers in the text fields, Num1 and Num2. the division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a NumberformatException. If Num2 were zero, the program would throw an Arithmetic Exception display the exception in a message dialog box.

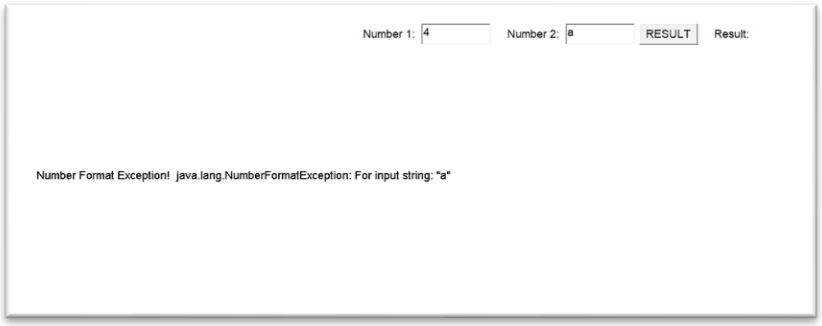
```
import java.awt.*;
import java.awt.event.*;
public class DivisionMain1 extends Frame implements ActionListener
{
      TextField num1,num2;
      Button dResult;
      Label outResult;
      String out="";
```

```
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(outResult);
	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(resultNum);
			repaint();
		}
	}
```

```
                catch(NumberFormatException e1)
                {
                        flag=1;
                        out="Number Format Exception!  "+e1;
                        repaint();
                }
                catch(ArithmeticException 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(),outResult.getY()+outResult.
getHeight()-8);
                else
                g.drawString(out,100,200);
                flag=0;
        }
        public static void main(String[] args)
        {
                DivisionMain1 dm=new DivisionMain1();
                dm.setSize(new Dimension(800,400));
                dm.setTitle("DivionOfIntegers");
                dm.setVisible(true);
        }
   }
```




Number 1: 4
Number 2: a
RESULT
Result:
Number Format Exception! java.lang.NumberFormatException: For input string: "a"