



# GALGOTIAS UNIVERSITY

## LAB MANUAL

### JAVA PROGRAMMING

---

**Name of School:** SCHOOL OF COMPUTING SCIENCE &  
ENGINEERING

**Department:** COMPUTER SCIENCE & ENGINEERING

**Year:** 2022-2023



(Under the Uttar Pradesh Private Universities Act No. 12 of 2019)

<b>SUBJECT</b>	<b>Java Programming Lab</b>	<b>PROGRAMME</b>	<b>B. Tech.</b>
<b>SUBJECT CODE</b>	<b>BCSE2333</b>	<b>SEMESTER</b>	<b>3</b>
<b>CREDITS</b>	<b>2</b>	<b>DURATION OF SEMESTER</b>	<b>13 Weeks</b>
<b>PREREQUISITE SUBJECTS</b>	<b>Java</b>	<b>SESSION DURATION</b>	<b>2 Hrs per Week</b>

### **Vision**

"To be recognized globally as a premier School of Computing Science and Engineering for imparting quality and value based education within a multi-disciplinary and collaborative research based environment."

### **Mission**

**The mission of the school is to:**

**M1:** Develop a strong foundation in fundamentals of computing science and engineering with responsiveness towards emerging technologies.

**M2:** Establish state-of-the-art facilities and adopt education 4.0 practices to analyze, develop, test and deploy sustainable ethical IT solutions by involving multiple stakeholders.

**M3:** Foster multidisciplinary collaborative research in association with academia and industry through focused research groups, Centre of Excellence, and Industry Oriented R&D Labs.

## **PROGRAM EDUCATIONAL OBJECTIVES**

**The Graduates of Computer Science and Engineering shall:**

**PEO1:** be engaged with leading Global Software Services and Product development companies handling projects in cutting edge technologies.

**PEO2:** serve in technical or managerial roles at Government firms, Corporates and contributing to the society as successful entrepreneurs through startup.

**PEO3:** undertake higher education, research or academia at institutions of transnational reputation.

## **PROGRAMME SPECIFIC OUTCOME (PSO):**

**The students of Computer Science and Engineering shall:**

**PSO1:** Have the ability to work with emerging technologies in computing requisite to Industry 4.0.

**PSO2:** Demonstrate Engineering Practice learned through industry internship and research project to solve live problems in various domains.

## **PROGRAMME OUTCOME (PO):**

**PO1 Computing Science knowledge:** Apply the knowledge of mathematics, statistics, computing science and information science fundamentals to the solution of complex computer application problems.

**PO2 Problem analysis:** Identify, formulate, review research literature, and analyze complex computing science problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and computer sciences.

**PO3 Design/development of solutions:** Design solutions for complex computing problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

**PO4 Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

**PO5 Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern computing science and IT tools including prediction and modeling to complex computing activities with an understanding of the limitations.

**PO6 IT specialist and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional computing science and information science practice.

**PO7 Environment and sustainability:** Understand the impact of the professional computing science solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

**PO8 Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the computing science practice.

**PO9 Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

**PO10 Communication:** Communicate effectively on complex engineering activities with the IT analyst community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

**PO11 Project management and finance:** Demonstrate knowledge and understanding of the computing science and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

**PO12 Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

## List of Programs

Sr. No.	Title of Lab Experiments
1	a) Write a JAVA program to print "Hello World." b) WAP in java to print the values of various primitive data types.
2	WAP in java to demonstrate operator precedence.
3	WAP in java to create a class Addition and a method add() to add two numbers.
4	Write a program that uses length property for displaying any number of command line arguments.
5	WAP in java to find the average of N numbers.
6	WAP in java to find out factorial of a number
7	WAP in java to find out the Fibonacci series
8	WAP in java to sort elements in 1D array in ascending order.
9	WAP in java to perform matrix addition using two 2D arrays.
10	WAP in java to find out the number of characters in a string.
11	WAP in java that implements method overloading.
12	Create a class Shape and override area() method to calculate area of rectangle, square and circle
13	WAP in java to demonstrate the properties of public private and protected variables and methods (with package).
14	WAP that illustrates method overriding
15	a) WAP in java demonstrating arithmetic exception and ArrayOutOf Bounds Exception, using try catch block b) WAP in java to demonstrate the use of nested try block and nested catch block.
16	Write a program to count the number of times a character appears in a File. [Note : The character check is case insensitive... i.e, 'a' and 'A' are considered to be the same]
17	WAP in java in two different ways that creates a thread by (i) extending thread class (ii) implementing runnable interface which displays 1 <sup>st</sup> 10 natural numbers.
18	a) WAP in java that connects to a database using JDBC & insert values into table. b) WAP to connect to a database using JDBC and delete values from table.

## Value Added List of Experiments

1. Create TreeSet, HashSet, LinkedHashSet.  
Add same integer values in all three.  
Display all three set and note down the difference.
2. WAP in java WAP in java and run applications that use "List" Collection objects
3. Create a HashMap<Rollno, Name>. Display Map, take rollno from user and display name.
4. WAP to do CRUD operation in java

**NAME & SIGNATURE OF COURSE COORDINATOR:**

## Course Outcomes

Upon successful completion of this course, students will be able to

<b>CO1</b>	Understand the java platform, structure of java class and java packages.
<b>CO2</b>	Understand object oriented concepts and implement the same using java operators, control statements, arrays.
<b>CO3</b>	Apply inheritance and exception handling concepts in solving problems.
<b>CO4</b>	Apply java IO stream concepts to solve problems efficiently.
<b>CO5</b>	Implement multithreading, collections and Java database connectivity concepts to solve problems in advance level.
<b>CO6</b>	Apply the concept of JDBC for connecting the application to the database.

**CO-PO-PSO MAPPING:**

<b>CO/PO Mapping</b>														
(S/M/W or 3/2/1 indicates strength of correlation)      S/3-Strong, M/2-Medium, L/1-Low														
CO's	Programme Outcomes (POs) / Programme Specific Outcome (PSO)													
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO1	2		1		2							2	2	2
CO2	2				2							1	1	
CO3	2	2		2	2							1	2	3
CO4	2			1	2								2	
CO5	2			1								1	1	
CO6	2	2										1	2	

**Continuous Assessment Pattern**

Practical IA	ETE	Total
100	0	100

**Rubrics for Practical IA**

S. No.	Rubrics - Parts	Marks
1	Performance	2
2	Result	3
3	File	2
4	Viva	3
Total		10

Name of the Course Coordinator

Signature

## **Experiment-1A**

**Aim:-** Write a JAVA program to print "Hello World."

### **Code:-**

```
public class Helloworld {  
    public static void main(String[] args)  
    {  
        System.out.print("Hello World");  
    }  
}
```

### **Output:-**

A screenshot of a terminal window with a dark background. The text "Hello World" is displayed in a light-colored font. Below the text, there is a horizontal line of small, colorful characters, likely a decorative separator or a result of a command.

## **Experiment-1B**

**Aim:-** WAP in java to print the values of various primitive data types

### **Code:-**

```
public class PrimitiveDataTypes {  
    static byte a;  
    static short b;  
    static int c;  
    static long d;  
    static float e;  
    static double f;  
    static char g;  
    static boolean h;  
    public static void main(String args[])  
    {  
        System.out.println("Value of different Primitive Data Types :-");  
  
        System.out.println("Value of byte : "+a);  
        System.out.println("Value of short : "+b);  
        System.out.println("Value of int : "+c);  
        System.out.println("Value of long : "+d);  
        System.out.println("Value of float : "+e);  
        System.out.println("Value of double : "+f);  
        System.out.println("Value of char : "+g);  
        System.out.println("Value of byte : "+h);  
    }  
}
```

### **Output:-**



```
Value of different Primitive Data Types :-
```

```
Value of byte : 0
```

```
Value of short : 0
```

```
Value of int : 0
```

```
Value of long : 0
```

```
Value of float : 0.0
```

```
Value of double : 0.0
```

```
Value of char :
```

```
Value of boolean : false
```

## Experiment-2

**Aim:-** WAP in java to demonstrate operator precedence.

**Code:-**

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

        int a = 10, b = 5, c = 1, result;
        result = a-++c-++b;

        System.out.println("Result = "+result);
    }
}
```

**Output:-**

```
Result = 2
```

## Experiment-3

**Aim:-** WAP in java to create a class Addition and a method add() to add two numbers.

**Code:-**

```
class Addition{
    int a=10,b=5;
    int sum=a+b;
}

public class sum {
    public static void main(String[] args)
    {
        Addition a1=new Addition();
        System.out.println("Sum of "+a1.a+" & " +a1.b+"="+a1.sum);
    }
}
```

```
}  
}
```

### **Output:-**

```
C:\Users\Prem Prakash Ranjan\Documents\java prog  
Sum of 10 & 5=15  
C:\Users\Prem Prakash Ranjan\Documents\java prog
```

## **Experiment-4**

**Aim:-** Write a program that uses length property for displaying any number of command line arguments.

### **Code:-**

```
public class test{  
    public static void main(String args[]){  
  
        for(int i=0;i<args.length;i++)  
            System.out.println(args[i]);  
  
    }  
}
```

### **Output:-**

```
C:\Users\Prem Prakash Ranjan\Documents\SEM 3 LAB EXPERIMENTS\JAVA> javac test.java  
  
C:\Users\Prem Prakash Ranjan\Documents\SEM 3 LAB EXPERIMENTS\JAVA> java test HELLO 1 2 3 JAVA  
HELLO  
1  
2  
3  
JAVA
```

## **Experiment-5**

**Aim:-** WAP in java to find the average of N numbers.

### **Code:-**

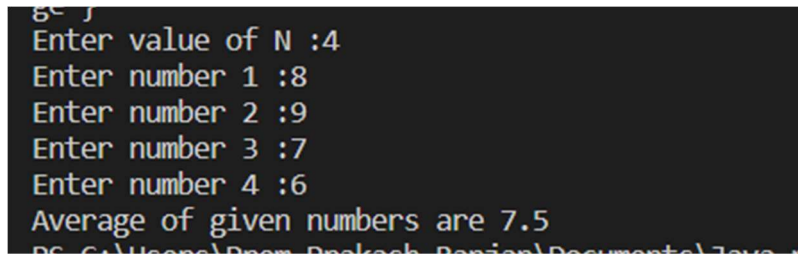
```
import java.util.Scanner;  
public class Average {  
    public static void main(String args[])  
    {  
        int i;  
        float n,j=0,k=0,avg;
```

```

        System.out.print("Enter value of N :");
        Scanner sc= new Scanner(System.in);
        n = sc.nextInt();
        for(i=1;i<=n;i++)
        {
            System.out.print("Enter number "+i+" :");
            j=sc.nextInt();
            k=k+j;
        }
        avg = k/n;
        System.out.println("Average of given numbers are "+avg);
    }
}

```

### **Output:-**



```

g> J
Enter value of N :4
Enter number 1 :8
Enter number 2 :9
Enter number 3 :7
Enter number 4 :6
Average of given numbers are 7.5
PC: C:\Users\Pran\Desktop\Project\Documents\Java

```

## **Experiment-6**

**Aim:-** WAP in java to find out factorial of a number

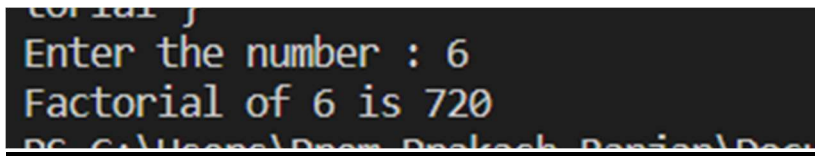
### **Code:-**

```

import java.util.Scanner;
public class Factorial {
    public static void main (String args[])
    {
        int n,i,fact=1;
        Scanner sc=new Scanner(System.in);
        System.out.print("Enter the number : ");
        n=sc.nextInt();
        sc.close();
        for(i=1;i<=n;i++)
        {
            fact=fact*i;
        }
        System.out.println("Factorial of "+n+" is "+fact);
    }
}

```

### Output:-



Enter the number : 6  
Factorial of 6 is 720

## Experiment-7

Aim:- WAP in java to find out the Fibonacci series

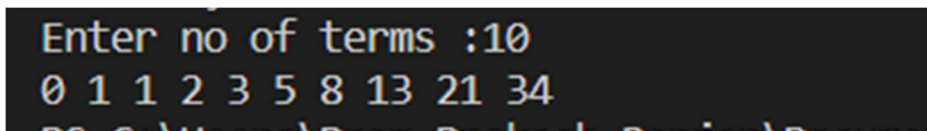
### Code:-

```
import java.util.Scanner;
public class Fibonacci {
    public static void main(String args[])
    {
        int n1=0,n2=1,n3,i,count;
        Scanner sc= new Scanner(System.in);
        System.out.print("Enter no of terms :");
        count=sc.nextInt();
        sc.close();
        System.out.print(n1+" "+n2);

        for(i=2;i<count;++i)
        {
            n3=n1+n2;
            System.out.print(" "+n3);
            n1=n2;
            n2=n3;
        }

    }
}
```

### Output:-



Enter no of terms :10  
0 1 1 2 3 5 8 13 21 34

## Experiment-8

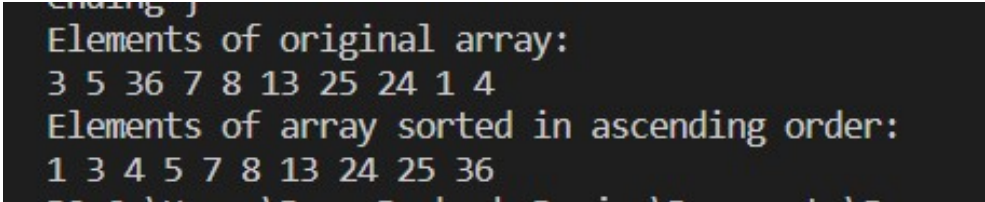
**Aim:-** WAP in java to sort elements in 1D array in ascending order.

### **Code:-**

```
public class Ascending {
    public static void main(String args[])
    {
        int arr[]={3,5,36,7,8,13,25,24,1,4},temp=0;
        System.out.println("Elements of original array: ");
        for (int i = 0; i < arr.length; i++)
        {
            System.out.print(arr[i] + " ");
        }
        for (int i = 0; i < arr.length; i++)
        {
            for (int j = i+1; j < arr.length; j++)
            {
                if(arr[i] > arr[j])
                {
                    temp = arr[i];
                    arr[i] = arr[j];
                    arr[j] = temp;
                }
            }
        }
        System.out.println();

        System.out.println("Elements of array sorted in ascending order: ");
        for (int i = 0; i < arr.length; i++)
        {
            System.out.print(arr[i] + " ");
        }
    }
}
```

### **Output:-**



```
Elements of original array:
3 5 36 7 8 13 25 24 1 4
Elements of array sorted in ascending order:
1 3 4 5 7 8 13 24 25 36
```

## Experiment-9

**Aim:-** WAP in java to perform matrix addition using two 2D arrays.

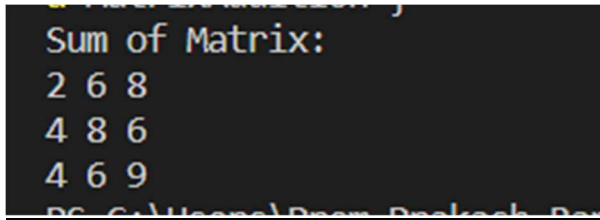
### Code:-

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

        int a[][]={{1,3,4},{2,4,3},{3,4,5}};
        int b[][]={{1,3,4},{2,4,3},{1,2,4}};
        int c[][]=new int[3][3];
        System.out.println("Sum of Matrix: ");

        for(int i=0;i<3;i++)
        {
            for(int j=0;j<3;j++)
            {
                c[i][j]=a[i][j]+b[i][j];
                System.out.print(c[i][j]+" ");
            }
            System.out.println();
        }
    }
}
```

### Output:-



The screenshot shows the output of the Java program. It displays the text "Sum of Matrix:" followed by a 3x3 grid of numbers representing the sum of the two input matrices. The numbers are arranged in three rows and three columns: 2 6 8, 4 8 6, and 4 6 9.

```
Sum of Matrix:
2 6 8
4 8 6
4 6 9
```

## Experiment-10

**Aim:-** WAP in java to find out the number of characters in a string.

### Code:-

```
public class CountCharacter
{
    public static void main(String[] args)
    {
        String string = "Simplicity is the best thing.";
        int count = 0;

        for(int i = 0; i < string.length(); i++)
        {
            if(string.charAt(i) != ' ')
                count++;
        }
    }
}
```

```

        System.out.println("Total number of characters in a string: " + count);
    }
}

```

### **Output:-**

```

Total number of characters in a string: 25

```

## **Experiment-11**

**Aim:-** WAP in java that implements method overloading.

### **Code:-**

```

public class Sum2 {
    public int sum(int x, int y) { return (x + y); }

    public int sum(int x, int y, int z)
    {
        return (x + y + z);
    }

    public double sum(double x, double y)
    {
        return (x + y);
    }

    public static void main(String args[])
    {
        Sum3 s = new Sum3();
        System.out.println("Sum of 10 & 20 is "+s.sum(10, 20));
        System.out.println("Sum of 10, 20 & 30 is "+s.sum(10, 20, 30));
        System.out.println("Sum of 10.5 & 20.5 is "+s.sum(10.5, 20.5));
    }
}

```

### **Output:-**

```

Sum of 10 & 20 is 30
Sum of 10, 20 & 30 is 60
Sum of 10.5 & 20.5 is 31.0

```

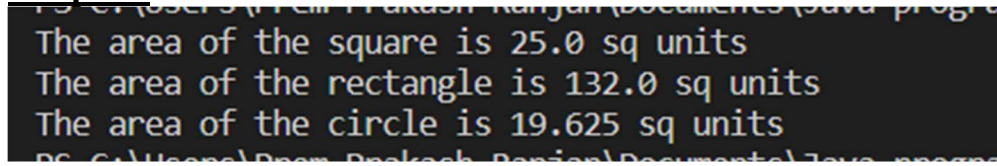
## Experiment-12

**Aim:-** Create a class Shape and override area() method to calculate area of rectangle, square and circle

**Code:-**

```
class Shape
{
    void area(float x)
    {
        System.out.println("The area of the square is "+Math.pow(x, 2)+" sq
units");
    }
    void area(float x, float y)
    {
        System.out.println("The area of the rectangle is "+x*y+" sq units");
    }
    void area(double x)
    {
        double z = 3.14 * x * x;
        System.out.println("The area of the circle is "+z+" sq units");
    }
}
public class Area {
    public static void main(String args[])
    {
        Shape ob = new Shape();
        ob.area(5);
        ob.area(11,12);
        ob.area(2.5);
    }
}
```

**Output:-**



```
The area of the square is 25.0 sq units
The area of the rectangle is 132.0 sq units
The area of the circle is 19.625 sq units
```

## Experiment-13

**Aim:-** WAP in java to demonstrate the properties of public private and protected variables and methods (with package).

**Code:-**

**Output:-**



## Experiment-14

**Aim:-** WAP that illustrates method overriding.

### **Code:-**

```
class subtraction{
    public double sub(double x, double y)
    {
        return (x - y);
    }
}

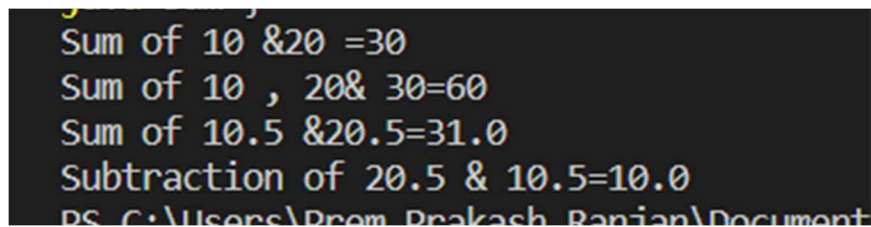
public class Sum extends subtraction{
    public int sum(int x, int y) { return (x + y); }

    public int sum(int x, int y, int z)
    {
        return (x + y + z);
    }

    public double sum(double x, double y)
    {
        return (x + y);
    }

    public static void main(String args[])
    {
        Sum s = new Sum();
        System.out.println("Sum of 10 &20 =" +s.sum(10, 20));
        System.out.println("Sum of 10 , 20& 30=" +s.sum(10, 20, 30));
        System.out.println("Sum of 10.5 &20.5=" +s.sum(10.5, 20.5));
        System.out.println("Subtraction of 20.5 & 10.5=" +s.sub(20.5, 10.5));
    }
}
```

### **Output:-**



```
Sum of 10 &20 =30
Sum of 10 , 20& 30=60
Sum of 10.5 &20.5=31.0
Subtraction of 20.5 & 10.5=10.0
PS C:\Users\Deep Prakash Baniya\Documents
```

## Experiment-15A

**Aim:-** WAP in java demonstrating arithmetic exception and ArrayOutOf Bounds Exception using try catch block

### **Code:-**

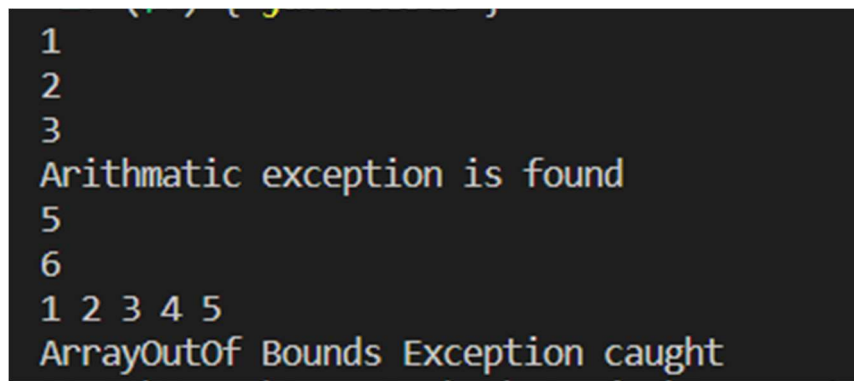
```

public class test3 {
public static void main(String args[])
{
System.out.println("1");
System.out.println("2");
System.out.println("3");
try{
System.out.println(100/0);
}
catch(ArithmeticException e)
{System.out.println("Arithmetic exception is found");}
System.out.println("5");
System.out.println("6");
int ar[] = { 1, 2, 3, 4, 5 };
try {
    for (int i = 0; i <= ar.length; i++)
        System.out.print(ar[i]+" ");
}
catch (ArrayIndexOutOfBoundsException e) {
    System.out.println("\nArrayOutOf Bounds Exception caught");
}

}
}

```

### **Output:-**



```

1
2
3
Arithmetic exception is found
5
6
1 2 3 4 5
ArrayOutOf Bounds Exception caught

```

## **Experiment-15B**

**Aim:-** WAP in java to demonstrate the use of nested try block and nested catch block

### **Code:-**

```

public class test{
    public static void main(String args[]){
        //outer try block
        try{
            //inner try block 1
            try{

```

```

        System.out.println("going to divide by 0");
        int b = 39/0;
    }
    //catch block of inner try block 1
    catch(ArithmeticException e)
    {
        System.out.println(e);
    }

    //inner try block 2
    try{
        int a[] = new int[5];

        //assigning the value out of array bounds
        a[5] = 4;
    }

    //catch block of inner try block 2
    catch(ArrayIndexOutOfBoundsException e)
    {
        System.out.println(e);
    }

    System.out.println("other statement");
}
//catch block of outer try block
catch(Exception e)
{
    System.out.println("handled the exception (outer catch)");
}

System.out.println("normal flow..");
}
}

```

### **Output:-**

```

going to divide by 0
java.lang.ArithmeticException: / by zero
java.lang.ArrayIndexOutOfBoundsException: Index 5 out of bounds for length 5
other statement
normal flow..

```

## **Experiment-16**

**Aim:-** Write a program to count the number of times a character appears in a File.

### **Code:-**

```
import java.util.Scanner;
public class CountOccuranceOfChar1
{
public static void main(String args[])
{
String str;
int i, len;
int counter[] = new int[256];
Scanner scanner = new Scanner(System.in);
System.out.print("Please enter a string: ");
//reading a string from the user
str = scanner.nextLine();
//finds the length of the string
len = str.length();
// loop through the string and count frequency of every character and store it in counter array
for (i = 0; i < len; i++)
{
counter[(int) str.charAt(i)]++;
}
//print Frequency of characters
for (i = 0; i < 256; i++)
{
if (counter[i] != 0)
{
//prints frequency of characters
System.out.println((char) i + " --> " + counter[i]);
}
}
}
}
```

### **Output:-**

```
Please enter a string: javatpoint
a --> 2
i --> 1
j --> 1
n --> 1
o --> 1
p --> 1
t --> 2
v --> 1
```

## **Experiment-17**

**Aim:-** WAP in java in two different ways that creates a thread by (i) extending thread class (ii) implementing runnable interface which displays 1<sup>st</sup> 10 natural numbers.

### **Code:-**

```
class MyNumber implements Runnable
{
public void run()
{
try
{
for(int i=1;i<=10;i++)
{
Thread.sleep(5000);
System.out.println(i);
}
}
catch(InterruptedException e)
{
System.out.println("Exception..." + e);
}
}
}
```

```
class Number
{
public static void main(String [] args)
{
MyNumber m1 = new MyNumber();
Thread thread = new Thread(m1);
thread.start();
}
}
```

### **Output:-**

```
1
2
3
4
5
6
7
8
9
10
```

## **Experiment-18**

**Aim:-** WAP in java that connects to a database using JDBC & insert values into table.

**Code:-**

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;

public class JDBC {
    static final String DB_URL = "jdbc:mysql://localhost/TUTORIALSPOINT";
    static final String USER = "guest";
    static final String PASS = "guest123";

    public static void main(String[] args) {
        // Open a connection
        try(Connection conn = DriverManager.getConnection(DB_URL, USER, PASS);
            Statement stmt = conn.createStatement();
        ) {
            // Execute a query
            System.out.println("Inserting records into the table...");
            String sql = "INSERT INTO Registration VALUES (100, 'Zara', 'Ali', 18)";
            stmt.executeUpdate(sql);
            sql = "INSERT INTO Registration VALUES (101, 'Mahnaz', 'Fatma', 25)";
            stmt.executeUpdate(sql);
            sql = "INSERT INTO Registration VALUES (102, 'Zaid', 'Khan', 30)";
            stmt.executeUpdate(sql);
            sql = "INSERT INTO Registration VALUES (103, 'Sumit', 'Mittal', 28)";
            stmt.executeUpdate(sql);
            System.out.println("Inserted records into the table...");
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

**Output:-**

Inserting records into the table...  
Inserted records into the table...