

**Don Bosco Institute of Technology, Kurla(W)**  
**Department of Electronics and Tele-Communication Engineering**  
**ECL304 - Skill Lab: C++ and Java Programming**  
**Sem III**  
**2021-22**

<b>Lab Number:</b>	<b>10</b>
<b>Student Name:</b>	<b>Suraj Kumar Das</b>
<b>Roll No :</b>	<b>42</b>

**Title:**

1. Write a program in java if a number is less than 0 and greater than 10 it generates the user-defined exception "out of range". Else it displays the square of the number.
2. Write a program in java to enter the number. If the first and second number is not entered it will generate the exception. Also, divide the first number with the second number and generate the arithmetic exception.

**Learning Objective:**

Students will be able to implement multiple inheritance using Interface concepts

**Learning Outcome:**

- Understanding the abstraction concept and hiding of the unnecessary code using interfaces.

**Course Outcome:**

<b>ECL304.4</b>	1. Implement different programming applications using packaging.
-----------------	--

**Theory:**

- **What is complete abstraction and how is it achieved in JAVA?**  
Ans: 1. Exception is defined as a problem that arises during the execution of a program. When an Exception occurs the normal flow of the program is disrupted and the program/Application terminates abnormally, which is not recommended, therefore these exceptions are to be handled.  
2. There is a difference between the terms error and exception. An Error indicates serious problem that a reasonable application should not try to catch.  
Exception indicates conditions that a reasonable application might try to catch.

**Faculty: Ms. Deepali Kayande**

**Don Bosco Institute of Technology, Kurla(W)**  
**Department of Electronics and Tele-Communication Engineering**  
**ECL304 - Skill Lab: C++ and Java Programming**  
**Sem III**  
**2021-22**

3. Java exception handling is managed with the help of the following keywords:

try, catch, throw, throws, and finally. Program statements that we as programmers think can raise exceptions are contained within a try block.

If an

exception occurs within the try block, it is thrown. The code can catch this

exception (using catch block) and handle it.

4. Every try block should be immediately followed either by a catch block or

finally block. A catch statement involves declaring the type of exception you

are trying to catch. If an exception occurs in protected code, the catch block (or

blocks) that follows the try is checked. If the type of exception that occurred is

listed in a catch block, the exception is passed to the catch block much as an

argument is passed into a method parameter.

5. The finally block follows a try block or a catch block. A finally block of code

always executes, irrespective of occurrence of an Exception. Using a finally

block allows you to run any clean-up-type statements that you want to execute,

no matter what happens in the protected code.

6. The main advantage of exception handling is to maintain the normal flow of

the application. An exception normally disrupts the normal flow of the application; that is why we need to handle exceptions.

7. There are mainly two types of exceptions: checked and unchecked. An error

is considered as the unchecked exception. However, according to Oracle, there

are three types of exceptions namely: Checked Exception, Unchecked Exception and Error.

8. The java.lang.Throwable class is the root class of Java Exception hierarchy

**Don Bosco Institute of Technology, Kurla(W)**  
**Department of Electronics and Tele-Communication Engineering**  
**ECL304 - Skill Lab: C++ and Java Programming**  
**Sem III**  
**2021-22**

inherited by two subclasses: Exception and Error. The hierarchy of Java Exception classes is given below:

- **Explain multiple abstraction and how is it performed in Java?**

Ans: 1. Java user-defined exception is a custom exception created and throws

that exception using a keyword 'throw'. It is done by extending a class 'Exception'. An exception is a problem that arises during the execution of the

program. In Object-Oriented Programming language, Java provides a powerful

mechanism to handle such exceptions. Java allows to create own exception

class, which provides own exception class implementation. Such exceptions are

called user-defined exceptions or custom exceptions.

2. There are advantages in using these user-defined exceptions; it allows users

to throw an exception which wants user wants to mean. Users can also reuse

any existing application; any piece of code that catches exception deals with a

possibility of actual exception was not thrown by code but by some other thirdparty code.

3. User Defined Exception or custom exception is creating your own exception

class and throws that exception using 'throw' keyword. ... There is no need to

override any of the above methods available in the Exception class, in your

derived class.

<b>Algorithm :</b>	<p>1. We have created a class named outofRangeException which is inheriting the</p> <p>class Exception. Inside this class we have declared a string variable named</p>
--------------------	--

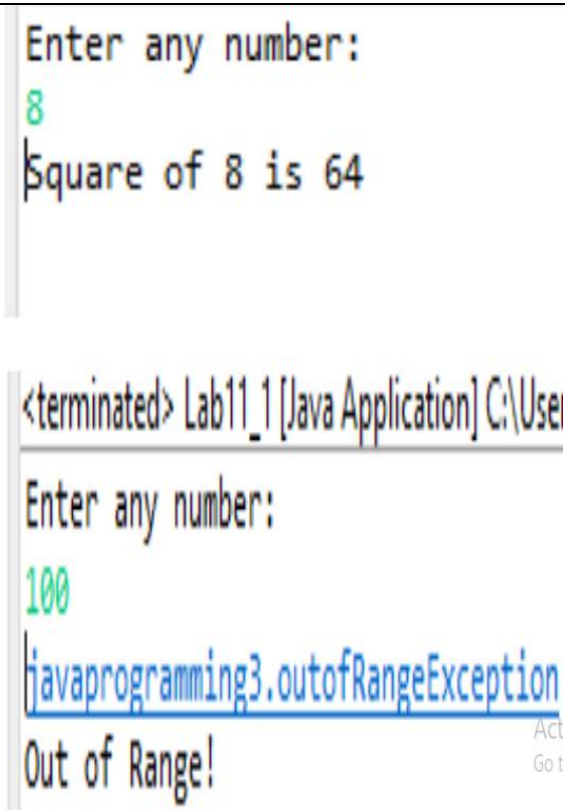
**Don Bosco Institute of Technology, Kurla(W)**  
**Department of Electronics and Tele-Communication Engineering**  
**ECL304 - Skill Lab: C++ and Java Programming**  
**Sem III**  
**2021-22**

	<p>display.</p> <p>2. Then in the main function we take a number from user and store it.</p> <p>3. We then check whether the number is less than zero or greater than 10 and if it is we display the message telling the user to enter a number within the confined range.</p> <p>4. If the user enters a number from the range 0 to 10 including 0 and 10, we will display the square of the entered number.</p>
<b>Program:</b>	<pre>package javaprogramming3; import java.util.Scanner; @SuppressWarnings({ "serial", "unused" }) class outofRangeException extends Exception{ String display; public outofRangeException( String display) { this.display=display; } } public class Lab11_1 { public static void main(String[] args) { Scanner in= new Scanner(System.in);</pre>

**Don Bosco Institute of Technology, Kurla(W)**  
**Department of Electronics and Tele-Communication Engineering**  
**ECL304 - Skill Lab: C++ and Java Programming**  
**Sem III**  
**2021-22**

	<pre> { try { int num1 = 0; System.out.println("Enter any number:"); num1=in.nextInt();  if(num1&lt;0   num1&gt;10) throw new outofRangeException ("Enter a number between the valid range which is from 0 to 10"); { int num2=num1*num1; System.out.println("Square of " +num1+ " is " +num2); } } catch(Exception xyz) { System.out.println(xyz); System.out.println("Out of Range!"); } } } } </pre>
<b>Input given:</b>	<p>1) For checking the range from 0 to 10 we enter num1=8</p> <p>2) For checking the range out of 0 to</p>

**Don Bosco Institute of Technology, Kurla(W)**  
**Department of Electronics and Tele-Communication Engineering**  
**ECL304 - Skill Lab: C++ and Java Programming**  
**Sem III**  
**2021-22**

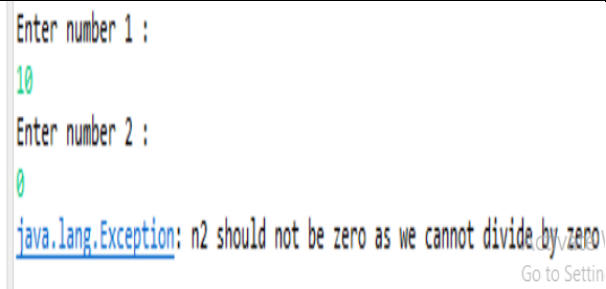
	10 we enter num1=100
<b>Output Screenshot:</b>	 <p>The first screenshot shows a Java application window titled 'Lab11_1 [Java Application]'. It prompts 'Enter any number:' and the user has entered '8'. The output is 'Square of 8 is 64'. The second screenshot shows the same application with the user entering '100'. It displays a blue error message: 'javaprogramming3.outofRangeException' followed by 'Out of Range!'.</p>

<b>Algorithm :</b>	<ol style="list-style-type: none"> <li>1. In the main function we will take two numbers form user and store it.</li> <li>2. Then in try block we are using the if condition to check when both n1 and n2 are zero we display that the numbers should be non zero.</li> <li>3. If user enters the second number as zero we will display that the n2 cant be zero as we can't divide by zero.</li> <li>4. If the user enters both the values as</li> </ol>
--------------------	--

**Don Bosco Institute of Technology, Kurla(W)**  
**Department of Electronics and Tele-Communication Engineering**  
**ECL304 - Skill Lab: C++ and Java Programming**  
**Sem III**  
**2021-22**

	non zero numbers, we will display the division of the two numbers.
<b>Program:</b>	<pre>package javaprogramming3; import java.util.Scanner; public class Lab11_2 { public static void main(String[] args) {     @SuppressWarnings("resource")     Scanner in= new Scanner(System.in);     int n1,n2;     System.out.println("Enter number 1     :");     n1=in.nextInt();     System.out.println("Enter number 2     :");     n2=in.nextInt();     try     {     if(n1==0 &amp;&amp; n2==0) {     throw(new Exception("Please enter     non zero numbers"));     }     if(n2==0) {     throw(new Exception("n2 should not     be zero as we cannot divide by     zero"));     }     int n3=n1/n2;</pre>

**Don Bosco Institute of Technology, Kurla(W)**  
**Department of Electronics and Tele-Communication Engineering**  
**ECL304 - Skill Lab: C++ and Java Programming**  
**Sem III**  
**2021-22**

	<pre> System.out.println("After Division we get "+n3); } catch(Exception abc) { System.out.println(abc); } } } } Don Bosco Ins </pre>
<b>Input given:</b>	<p>1) For 1st case we will take n1=0 and n2 =0 for generating the exception when both the numbers entered by the user are zero.</p> <p>2) For 2nd case we will take n1=10 and n2=0 for generating the exception when user enters second number as 0 and we cannot divide by zero.</p>
<b>Output Screenshot:</b>	 <p>The screenshot shows a terminal window with the following text:</p> <pre> Enter number 1 : 10 Enter number 2 : 0 java.lang.Exception: n2 should not be zero as we cannot divide by zero Go to Settings </pre>



**Don Bosco Institute of Technology, Kurla(W)**  
**Department of Electronics and Tele-Communication Engineering**  
**ECL304 - Skill Lab: C++ and Java Programming**  
**Sem III**  
**2021-22**

	<pre>1) Enter number 1 : 0 Enter number 2 : 0 <u>java.lang.Exception</u>: Please enter non zero numbers</pre>
--	---