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

Lab Number:	**11**
Student Name:	**Sarika Galphade**
Roll No :	**36**

Title:

1. Write a program injava 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 user-defined exceptions

Learning Outcome:

Understanding the exception handling concept and making the programming interface error-free.

Course Outcome:

ECL304.3	Articulate exception handling methods.

Theory:

- What is exception handling and how is it achieved in JAVA?

The Exception Handling in Java is one of the powerful mechanism to handle the runtime errors so that the normal flow of the application can be maintained.

In this tutorial, we will learn about Java exceptions, it's types, and the difference between checked and unchecked exceptions.

All exception and errors types are sub classes of class **Throwable**, which is base class of hierarchy. One branch is headed by **Exception**. This class is used for exceptional conditions that user programs should catch. NullPointerException is an example of such an exception. Another branch, **Error** are used by the Java run-time system(JVM) to indicate errors having to do with the run-time environment itself(JRE). StackOverflowError is an example of such an error.

Default Exception Handling :

- Whenever inside a method, if an exception has occurred, the method creates an Object known as Exception Object and hands it off to the run-time system(JVM). The exception object contains name and description of the exception, and current state of the program where exception has occurred. Creating the Exception Object and handling it to the run-time system is called throwing an Exception.

- There might be the list of the methods that had been called to get to the method where exception was occurred. This ordered list of the methods is called Call Stack.Now the following procedure will happen.
- The run-time system searches the call stack to find the method that contains block of code that can handle the occurred exception. The block of the code is called Exception handler.
- The run-time system starts searching from the method in which exception occurred, proceeds through call stack in the reverse order in which methods were called.
- If it finds appropriate handler then it passes the occurred exception to it.

- Explain user defined exceptions in java?

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. Let us dig deeper and look at how user-defined exceptions are created in Java, its syntax, if any, and how it is implemented by solving some examples.

Syntax:

We do not have any particular syntax for Java user-defined exception; we will see how to create a User-defined exception.

Below is the code which will help to Create a User-defined exception class,

```
class SampleException{
public static void main(String args[]){
try{
throw new UserException(<value>); // used to create new exception and throw
}
catch(Exception e){
System.out.println(e);
}
}
}
class UserException extends Exception{
// code for exception class
}
```

Algorithm :	Step1:Create class Step2:Create main class Step3:Now create try class and perform task . Step4:Trow exception where ever required Step5:Create catch class and followed by main class.
Program:	/*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".

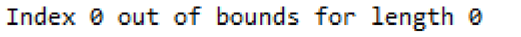
<table>
<tr><td></td><td>

```
Else it displays the square of the number.*/


public class Demo4 {
       public static void main(String arr[])
       {
       try
       {
       int a=Integer.parseInt(arr[0]);
       if(a<0|| a>10)
       throw new  ArrayIndexOutOfBoundsException("valid range is 0 
to 10");
       {
       int s=a*a;
       System.out.println("Square is:"+s);
       }
       }
       catch(Exception ex)
       {
       System.out.println(ex);
       }
       }

}
```

</td></tr>
<tr><td><b>Input given:</b></td><td>-</td></tr>
<tr><td><b>Output Screenshot:</b></td><td>



```
Index 0 out of bounds for length 0
```

</td></tr>
</table>

<table>
<tr><td><b>Algorithm :</b></td><td>

Step1:Create class

Step2:Create main class

Step3:Now create try class and perform task .

Step4:Trow exception where ever required

Step5:Create catch class and followed by main class.

</td></tr>
<tr><td><b>Program:</b></td><td>

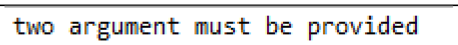
```
/*2.   Write a program in java to enter the number.
 * If the first and second number is not entered it will generate 
the exception.
 * Also, divide the first number with the second number and generate
 * the arithmetic exception./*
 */
public class Divide2 {
```

</td></tr>
</table>

<table>
<tr>
<td></td>
<td>

```
        public static void main(String[] arr)
        {
                try
                {
                if(arr.length<2)
                throw(new Exception("two argument must be provided"));
                int a= Integer.parseInt(arr[0]);
                int b=Integer.parseInt(arr[1]);
                if(b==0)
                throw(new Exception("second argument should be non
zero"));
                int c=a/b;
                System.out.println("result:"+c);
                }
                catch(Exception e)
                {
                System.out.println(e);
                }
```

</td>
</tr>
<tr>
<td><b>Input given:</b></td>
<td>-</td>
</tr>
<tr>
<td><b>Output Screenshot:</b></td>
<td>



```
two argument must be provided
```

</td>
</tr>
</table>