## East West University

**Department of Computer Science and Engineering**

## A/2, Jahurul Islam Avenue, Jahurul Islam City, Aftabnagar, Dhaka

**Lab Manual:** 08

**Lab Topic:** Exception Handling

**Course Code:** CSE110 (Object Oriented Programming)

**Course Instructor:** Tanni Mittra, Senior Lecturer, CSE

**Lab Objective**

1. **Learn a** mechanismto handle Exception in Java program

**Lab Activities:**

1. [**Built-in Exceptions**](https://www.geeksforgeeks.org/built-exceptions-java-examples/) **Handle**

class ArithmeticException\_Demo

{

    public static void main(String args[])

    {

        try {

            int a = 30, b = 0;

            int c = a/b;

            System.out.println ("Result = " + c);

        }

        catch(ArithmeticException e) {

System.out.println (e);

            System.out.println ("Can't divide a number by 0");

        }

    }

}

Class Testthrows1 {

Void m () throws IOException

{

throw new IOException("device error");//checked exception

}

}

**Lab problem 1:**

* Write a program that creates a *Calculator* class. The class contains two variables of integer type. Design a constructor that accepts two values as parameter and set those values.
* Design four methods named *Add ()*, *Subtract ()*, *multiply ()*, *Division ( )* for performing addition, subtraction, multiplication and division of two numbers.
* For addition and subtraction, two numbers should be positive. If any negative number is entered then throw an exception in respective methods. So design an exception handler (***ArithmeticException***) in *Add ()* and *Subtract ()* methods respectivelyto check whether any number is negative or not.
* For division and multiplication two numbers should not be zero. If zero is entered for any number then throw an exception in respective methods. So design an exception handler (***ArithmeticException***) in *multiply ()* and *Division ()* methods respectivelyto check whether any number is zero or not.
* Write a main class and declare four objects of *Calculator* class. Perform addition (obj1), subtraction (obj2), multiply (obj3) and division (obj4) operations for these objects. If any non integer values are provided as input;then you should throw an exception (***NumberFormatException***) and display a message that informs the user ofthe wrong input before exiting.

1. [**User Defined Exceptions**](https://www.geeksforgeeks.org/built-exceptions-java-examples/) **Handle**

**Lab problem 2:**

* Create an exception class named *MyException* that extend a base class named *Exception*
* Design a constructor in your class that accepts a string value set it to the super class constructor to display the exception message.
* Create a main class named *product*. Write a method inside the class called *productCheck(int weight)* that accepts weight of the product. Inside the method, if the weight is less than 100 then throw an exception “Product is invalid” otherwise print the weight of the product.
* Inside the main method declare single object of the product class and call the *productCheck()* method to display the weight of the product.

1. **Java Multi-catch block**

* At a time only one exception occurs and at a time only one catch block is executed.
* All catch blocks must be ordered from most specific to most general, i.e. catch for ArithmeticException must come before catch for Exception.

public class MultipleCatchBlock1 {

     public static void main(String[] args) {

            try{

                int a[]=new int[5];

                a[5]=30/0;

               }

            catch(ArithmeticException e)

                  {

                   System.out.println("Arithmetic Exception occurs");

                  }

            catch(ArrayIndexOutOfBoundsException e)

                  {

                   System.out.println("ArrayIndexOutOfBounds Exception occurs");

                  }

            catch(Exception e)

                  {

                   System.out.println("Parent Exception occurs");

                  }

          System.out.println("rest of the code");

    }

}

* Try for the following code:

1. try{

int a[]=new int[5];

System.out.println(a[10]);

}

2.

try{

int a[]=new int[5];

a[5]=30/0;

System.out.println(a[10]);              }

3.

try{

String s=null;

System.out.println(s.length());

}

4.

class MultipleCatchBlock5{

  public static void main(String args[]){

    try{

     int a[]=new int[5];

     a[5]=30/0;

    }

   catch(Exception e){System.out.println("common task completed");}

   catch(ArithmeticException e){System.out.println("task1 is completed");}

   catch(ArrayIndexOutOfBoundsException e){System.out.println("task 2 completed");}

   System.out.println("rest of the code...");

 }  }

* Grouping exception in catch block

catch (NoSuchPaddingException | NoSuchAlgorithmException

| InvalidKeyException | BadPaddingException

| IllegalBlockSizeException | IOException ex) {

System.err.println(ex);

}

1. **Java Nested try block**

class Excep6{

public static void main(String args[]){

try{

    try{

      System.out.println("going to divide");

      int b =39/0;

    }

catch(ArithmeticException e)

{System.out.println(e);}

    try{

     int a[]=new int[5];

     a[5]=4;

    }

catch(ArrayIndexOutOfBoundsException e)

{System.out.println(e);}

System.out.println("other statement);

}

catch(Exception e)

{System.out.println("handeled");}

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

 }

}