**Experiment 9**

**Exception Handling**

**Date of Submission:** 16-10-2020

**Aim:** Write a Java program that shows the usage of try, catch, throws and finally.

(a)---Try-Finally example

(b)---Multiple catch example (3 catches for a single try)

(c)---Nested Try (3 levels of nesting)

(d)---Throw an exception when there are no sufficient arguments passed into command line as input for adding two numbers.

(e)---Throws example (for handling two exceptions in a method)

**Concepts used:** Exception handling

**(a)---Try-Finally example**

**Algorithm:**

**Input:** command line arguments

**Output:** Corresponding to the argument given, the required exception is handled

//args is the command line arguments

1. Start
2. a = args.length
3. try
4. b = 3/a
5. end try
6. finally
7. print(“In the finally statement”)
8. endFinally
9. Stop

**Program Code:**

/\* File name: TryFinallyExample.java

\*

\* Done By: Rohit Karunakaran

\* \*/

public class TryFinallyExample

{

public static void main(String args[]){

int a =3;

int len = args.length;

try{

int b = a/len;

}

finally{

System.out.println("In the finally Statement");

System.out.println("Have a nice Day");

System.out.println("---------------------------------------\n");

}

}

}

**Sample Input 1:**

\*no input\*

**Sample output 1:**

In the finally Statement

Have a nice Day

---------------------------------------

Exception in thread "main" java.lang.ArithmeticException: / by zero

at TryFinallyExample.main(TryFinallyExample.java:12)

**Sample input 2:**

hello

**Sample output 2:**

In the finally Statement

Have a nice Day

---------------------------------------

**(b)---Multiple catch example (3 catches for a single try)**

**Algorithm**

**Input:** command line arguments

**Output:** The Exception occured is handled properly

**Steps**

// args is the command line arguments

1. Start
2. try
3. len = args.length
4. i = Integer.valueOf(args[0])
5. b = Integer.valueOf(args[1])
6. c = i/b
7. endTry
8. catch ArrayIndexOutOfBoundsException e
9. print(“Not sufficent arguments”)
10. endCatch
11. catch NumberFormatException e
12. print(“Numbers are the expected arguments”)
13. endCatch
14. catch ArithmeticException e
15. print (“The second arguement is 0”)
16. endCatch
17. Stop

**Program Code**

/\* File Name : MultipleCatchExample.java

\*

\* Done By: Rohit Karunakaran

\*

\* \*/

public class MultipleCatchExample{

public static void main(String args[]){

//3 catches for a single try

System.out.println("Example for multiple Catch Statements");

try

{

int len = args.length;

int i = Integer.parseInt(args[0]);

int b = Integer.parseInt(args[1]);

int c = i/b;

}

catch(NumberFormatException e)

{

System.out.println("In the First Catch");

System.out.println("An Exception has occurred while parsing the command line input :"+e);

System.out.println("Expected an integer");

}

catch(ArrayIndexOutOfBoundsException e)

{

System.out.println("In the Second Catch");

System.out.println("Expected more than 1 command line argument");

System.out.println("Exception occured is : "+e);

}

catch(ArithmeticException e)

{

System.out.println("In the Third Catch");

System.out.println("An Exception has occured : "+e);

}

finally

{

//System.out.println("");

System.out.println("---------------------------------------");

}

}

}

**Sample input:**

\*no input\*

**Sample output:**

Example for multiple Catch Statements

In the Second Catch

Expected more than 1 command line argument

Exception occured is : java.lang.ArrayIndexOutOfBoundsException: Index 0 out of bounds for length 0

---------------------------------------

**Sample input:**

3 0

**Sample output:**

Example for multiple Catch Statements

In the Third Catch

An Exception has occured : java.lang.ArithmeticException: / by zero

---------------------------------------

**Sample input:**

3 hello

**Sample output:**

Example for multiple Catch Statements

In the First Catch

An Exception has occurred while parsing the command line input :java.lang.NumberFormatException: For input string: "hello"

Expected an integer

---------------------------------------

**(c)---Nested Try (3 levels of nesting)**

**Algorithm**

**Input :** No input

**Output:** Corresponding try catch block is executed according to the exception (if occured)

**Steps:**

1. a = {1,2,5,32,12}
2. try
3. b=a[3]
4. print(“In the outermost try block”)
5. try
6. print(“Inside the inner try block”)
7. c = 2\*a[1]+a[0]-a[2]
8. try
9. print(“Inside the inner most try block”)
10. e= b/c
11. endtry
12. catch ArithmeticException e
13. print(“divide by 0 error”)
14. endcatch
16. finally
17. print(“In the finally of the inner most try block”)
18. endfinally
19. end try
21. finally
22. print(“Inside the finally statement of the inner try block”)
23. end finally
24. d = a[41]
25. print(“Aslong as the size of a is less than 42 this will not be executed”)
26. endtry
27. catch ArrayIndexOutOfBoundException
28. print(“The size of array is less”)
29. end catch
30. finally
31. print(“In the final block of the outermost try”)
32. end finally

**Program code:**

/\* File name: NestedTryExample.java

\*

\* Done By: Rohit Karunakaran

\*/

public class NestedTryExample

{

static void main(String args[]){

//3 levels of nesting

int a[] = {1,2,5,32,12};

System.out.println("\nExample With Nested Try Statements");

try

{

System.out.println("Inside the Outer Most Try Block");

System.out.println("=================================\n");

int b = a[3];

try

{

System.out.println("Inside the Inner Try Block");

System.out.println("=================================\n");

int c = 2\*a[1]+a[0]-a[2];

try

{

System.out.println("Inside the Innermost Try Block");

System.out.println("=================================\n");

int e = b/c;

}

catch(ArithmeticException e)

{

System.out.println("An Exception has occured in the innerMost try Block");

System.out.println("An Arithmetic expression error occured : "+e);

}

finally

{

System.out.println("In the final block of the Innermost try expression");

System.out.println("=================================\n");

}

//System.out.println("After the innermost try bolck Statement");

}

finally

{

System.out.println("In the final block of the Inner try expression");

System.out.println("=================================\n");

}

int d = a[41]; //Array out of bounds exception

System.out.println("I assure you that as long as the size of a is less than 42,");

System.out.println("This statement will not be excecuted");

}

catch(ArrayIndexOutOfBoundsException e)

{

System.out.println("Exception :"+e+" \nhas Occured in the outermost try block");

}

finally

{

System.out.println("The Final Block of the Outermost try is complete");

//System.out.println("=================================\n");

System.out.println("---------------------------------------\n");

}

}

}

**Sample Output:**

Example With Nested Try Statements

Inside the Outer Most Try Block

=================================

Inside the Inner Try Block

=================================

Inside the Innermost Try Block

=================================

An Exception has occured in the innerMost try Block

An Arithmetic expression error occured : java.lang.ArithmeticException: / by zero

In the final block of the Innermost try expression

=================================

In the final block of the Inner try expression

=================================

Exception :java.lang.ArrayIndexOutOfBoundsException: Index 41 out of bounds for length 5

has Occured in the outermost try block

The Final Block of the Outermost try is complete

---------------------------------------

**(d)---Throw an exception when there are no sufficient arguments passed into command line as input for adding two numbers.**

**Algorithm:**

**Input:** Command line arguments (args)

**Output:** sum of numbers passed as the command line argument or the corresponding exeption

**Steps:**

1. Start
2. if args.length<2 then
3. throw new ArrayIndexOutOfBoundsException(“No sufficient arguments”)
4. else
5. print Integer.valueOf(args[0]) + Integer.valueOf(args[1])
6. Endif
7. Stop

**Program Code:**

/\* File Name: NoSufficientArgument.java

\*

\* Done By: Rohit Karunakaran

\*/

public class NoSufficientArgumentForAdding

{

public static void main(String[] args) //throw Excenption

{

System.out.println("In the funcion noSufficientForAddingExample\n");

try

{

if(args.length<2){

throw new ArrayIndexOutOfBoundsException("Less than 2 command line arguments found");

}

else{

int a = Integer.parseInt(args[0]);

int b = Integer.parseInt(args[1]);

int c = a+b;

System.out.println("No Exception has occurred. The sum = "+c);

}

}

catch(ArrayIndexOutOfBoundsException e)

{

System.out.println(e);

}

catch(NumberFormatException e)

{

System.out.println("Pass an Integer as a command line argument");

System.out.println(e);

}

}

}

**Sample Input 1:**

5 4

**Sample output 1:**

No Exception has occurred. The sum = 9

**Sample Input 2:**

5

**Sample output 2:**

java.lang.ArrayIndexOutOfBoundsException: Less than 2 command line arguments found

**Sample Input 3:**

5 hello

**Sample output 3:**

Pass an Integer as a command line argument

java.lang.NumberFormatException: For input string: "hello"

**(e)---Throws example (for handling two exceptions in a method)**

**Algorithm:**

1. Start
2. import java.io
3. //main function throws IOException and Arithmetic exception
4. try
5. File f = new File(“file.txt”)
6. FileReader fr = new FileReader(f)
7. endtry
8. catch IOException e
9. Print “An IOException has occurred”
10. endcatch
11. Stop

**Program Code**

/\*File Name: ThrowsExample.java

\*

\*Done By: Rohit Karunakaran

\*/

import java.io.\*;

public class ThrowsExample

{

public static void main(String args[]) throws IOException, FileNotFoundException

{

File f = new File("File1.txt");

FileReader f1 = new FileReader(f);

System.out.println("The file is opened");

f1.close();

f.close();

}

}

**Sample output:** //Before creating the file

Exception in thread "main" java.io.FileNotFoundException: File1.txt (No such file or directory)

at java.base/java.io.FileInputStream.open0(Native Method)

at java.base/java.io.FileInputStream.open(FileInputStream.java:211)

at java.base/java.io.FileInputStream.<init>(FileInputStream.java:153)

at java.base/java.io.FileReader.<init>(FileReader.java:75)

at ThrowsExample.main(ThrowsExample.java:14)

**Sample output:** //After creating the file

The file is opened

**Result:** The programs were successfully compiled and the required output was obtained