Exception:

Formally exception is an event, which occurs during execution of a program, that disrupts the normal flow of program’s execution. In simple words, any unwanted issues/events/errors faced during execution that stops the intended execution of the program is exception.

Handling of such errors/events/issues is called exception handling.

There are many types of exceptions in Java:

Four exceptions learned in the class are:

1. Arithmetic Exception:

This exception occurs when there is error in mathematical arithmetic errors in the code at run time. Classic example of arithmetic exception is an integer divided by 0. Since, an integer divided by zero results in infinity whose value is unknown and cannot be computed, it stops the next line of execution of program at run time.

This exception is handled with try and catch block..

Ex.

Instead of writing

int value = 200 / 0;

We put the above statement in try and catch block.

Solution with try and catch block:

try {

int value = 200 / 0;

} catch (ArithmeticException e) {

System.out.println(e);

}

1. Null pointer Exception:

This exception occurs when objects are referenced by null value. Here, name is an instance of String class whose value is null. Trying to print length of name generates nullpointer exception because it’s value is null.

String name = null;

System.out.println(name.length()); //This statement produces error.

Solution with try and catch block:

try {

String name = null;

} catch (NullPointerException e) {

System.out.println(e);

1. NumberFormatException:

This exception is generated when String or any other object is attempted to be converted to a numeric value.

String s = “Hello”;

int i = Integer.parseInt(s); //This statement produces error./

Solution with try catch block:

try {

String s = “Hello”;

int i = Integer.parseInt(s);

} catch (NumberFormatException e) {

System.out.println(e);

}

1. ArrayOutOfBoundsException:

This exception is generated when any operation is done on array index that is greater than declared array length or size. In the code below, array of 10 elements is declared but value 10 is assigned to array index of 20. Since 20 is greater than 10, it generates ArrayOutOfBoundsException.

int arr[] = new int[10];

arr[20] = 10; //This statement produces error.

Solution using try and catch block:

try {

int arr[] = new int [10];

arr[20] = 10;

} catch(ArrayOutOfBoundsException e) {

System.out.println(e);