UNIT-3(Exceptional Handling,I/O)

Exception :

an exception is an abnormal condition that disrupts the normal flow of the program. It is an object which is thrown at runtime.

For example take a scenario:

statement 1;
statement 2;
statement 3;
statement 4;
statement 5;//exception occurs
statement 6;
statement 7;
statement 8;
statement 9;
statement 10;

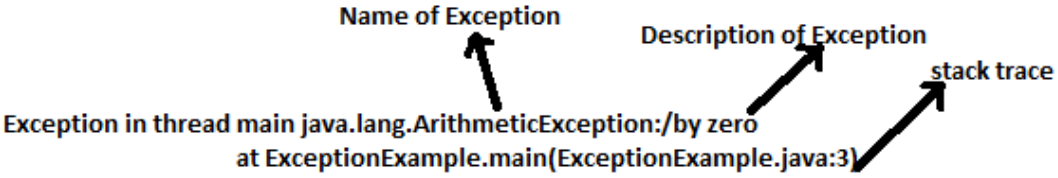
Suppose there are 10 statements in your program and there occurs an exception at statement 5, the rest of the code will not be executed i.e. statement 6 to 10 will not be executed. If we perform exception handling, the rest of the statement will be executed..

Exception Example and Default Exception Handling in Java:

```
class ExceptionExample{
        public static void main(String args[]){
                int a=100/0;//exception raise
                  //rest of code
                int b=100/2;
                System.out.println("hello after exception:"+b);
                }
                }
```

```
C:\javaprograms\innerclasses>java ExceptionExample
Exception in thread "main" java.lang.ArithmeticException: / by zero
        at ExceptionExample.main(ExceptionExample.java:3)
```

In the above example an Arithmetic exception arise at line number 3 in main method, due to which program is terminated abnormally and rest of the code after exception is not executed. Since we did not handle it in the main method so JVM terminates the main method and give control to java Default Exception handler handles this exception as shown in above output as:



Exception Handling:

=>Defining alternative way to define the exception is called Exception handling.
=>Exception Handling is a mechanism to handle runtime errors such as ArithmeticException, ClassNotFoundException, IOException, SQLException, RemoteException, etc.

=>to handle exception we use try and catch block .
=>In try block we write the code that generate Exception.
=>In catch block we write the code that handle the Exception.

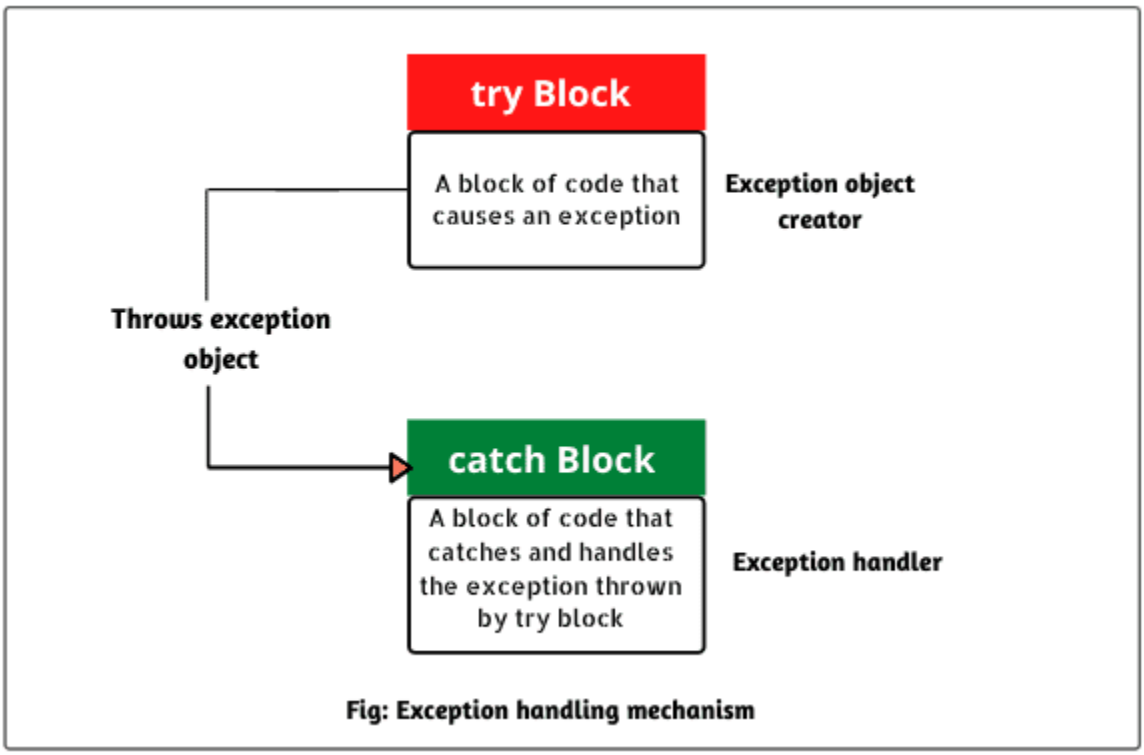


Fig: Exception handling mechanism

Example:

```
class ExceptionExample{
        public static void main(String args[]){
                try{
                int a=100/0;//exception raise
                  }
                  catch(ArithmeticException e){
                          System.out.println(e);
                  }
                int b=100/2;
```

```
//rest of code
System.out.println("hello after exception:"+b);
}
}
```

Output:

```
C:\javaprograms\innerclasses>java ExceptionExample
java.lang.ArithmeticException: / by zero
hello after exception:50
```

Now in the above program the program does not terminate abnormally since Exception is handled using try and catch block so rest of the code is also executed.

Advantage of Exception Handling:

The core 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 use exception handling.

For example take a scenario:

```
statement 1;
statement 2;
statement 3;
statement 4;
statement 5;//exception occurs
statement 6;
statement 7;
statement 8;
statement 9;
statement 10;
```

Suppose there are 10 statements in your program and there occurs an exception at statement 5, the rest of the code will not be executed i.e. statement 6 to 10 will not be executed. If we perform exception handling, the rest of the statement will be executed. That is why we use exception handling .