\*\*Handling Exceptions\*\*   
  
- Differentiate among checked exceptions, unchecked exceptions, and Errors (\*\*Prepare a document\*\*)   
- Create a try-catch block and determine how exceptions alter normal program flow   
- Describe the advantages of Exception handling (\*\*Prepare a document\*\*)   
- Create and invoke a method that throws an exception   
- Recognize common exception classes (such as NullPointerException, ArithmeticException, ArrayIndexOutOfBoundsException, ClassCastException)

**What Is an Exception?**

The term *exception* is shorthand for the phrase "exceptional event”.

**Definition:** An *exception* is an event, which occurs during the execution of a program, that disrupts the normal flow of the program's instructions.

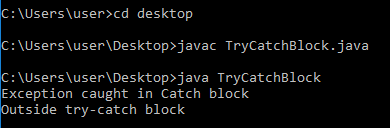
### **Types of Exception**

There are mainly two types of exceptions: checked and unchecked where error is considered as unchecked exception. The sun microsystem says there are three types of exceptions:

1. Checked Exception
2. Unchecked Exception
3. Error

|  |  |  |
| --- | --- | --- |
| **Checked Exceptions** | **Unchecked Exceptions** | **Errors** |
| * Exceptions that are checked at compile time. If some code within a method throws a checked exception, then the method must either handle the exception or it must specify the exception using throws keyword. | * The exceptions that are not checked at compiled time. * The classes that extend Runtime Exception are known as unchecked exceptions e.g. Arithmetic Exception, NullPointer Exception | * An Error indicates serious problem that a reasonable application should not try to catch. * Error is irrecoverable e.g. OutOfMemory error. |

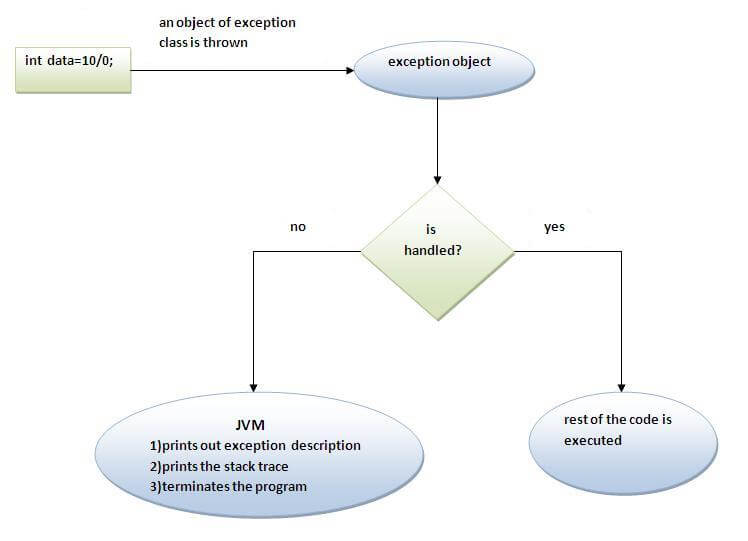
**Try Catch Block:**

****

**Source code :**

[Trycatch Block](https://drive.google.com/open?id=1SyC_SMjYp44pLtL4f17laECxTfpwfZ91)

Exceptions alter normal program flow in this way :



**Advantages of Exception handling:**

## **1: Separating Error-Handling Code from "Regular" Code :**

Exceptions provide the means to separate the details of what to do when something out of the ordinary happens from the main logic of a program. In traditional programming, error detection, reporting, and handling often lead to confusing code.

## **2: Grouping and Differentiating Error Types :**

Because all exceptions thrown within a program are objects, the grouping or categorizing of exceptions is a natural outcome of the class hierarchy. We can create groups of exceptions and handle exceptions in a general fashion, or you can use the specific exception type to differentiate exceptions and handle exceptions in an exact fashion.

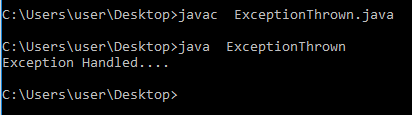
A method can catch an exception based on its group or general type by specifying any of the exception's superclasses in the catch statement. For example, to catch all I/O exceptions, regardless of their specific type, an exception handler specifies an IOException argument.

catch (IOException e) {

...

}

Method that throws an exception :



**Source code :**

[Method Throwing Exception](https://drive.google.com/open?id=1dQFcOv6F1p8W32P-GZMKEOEEMp755gqw)