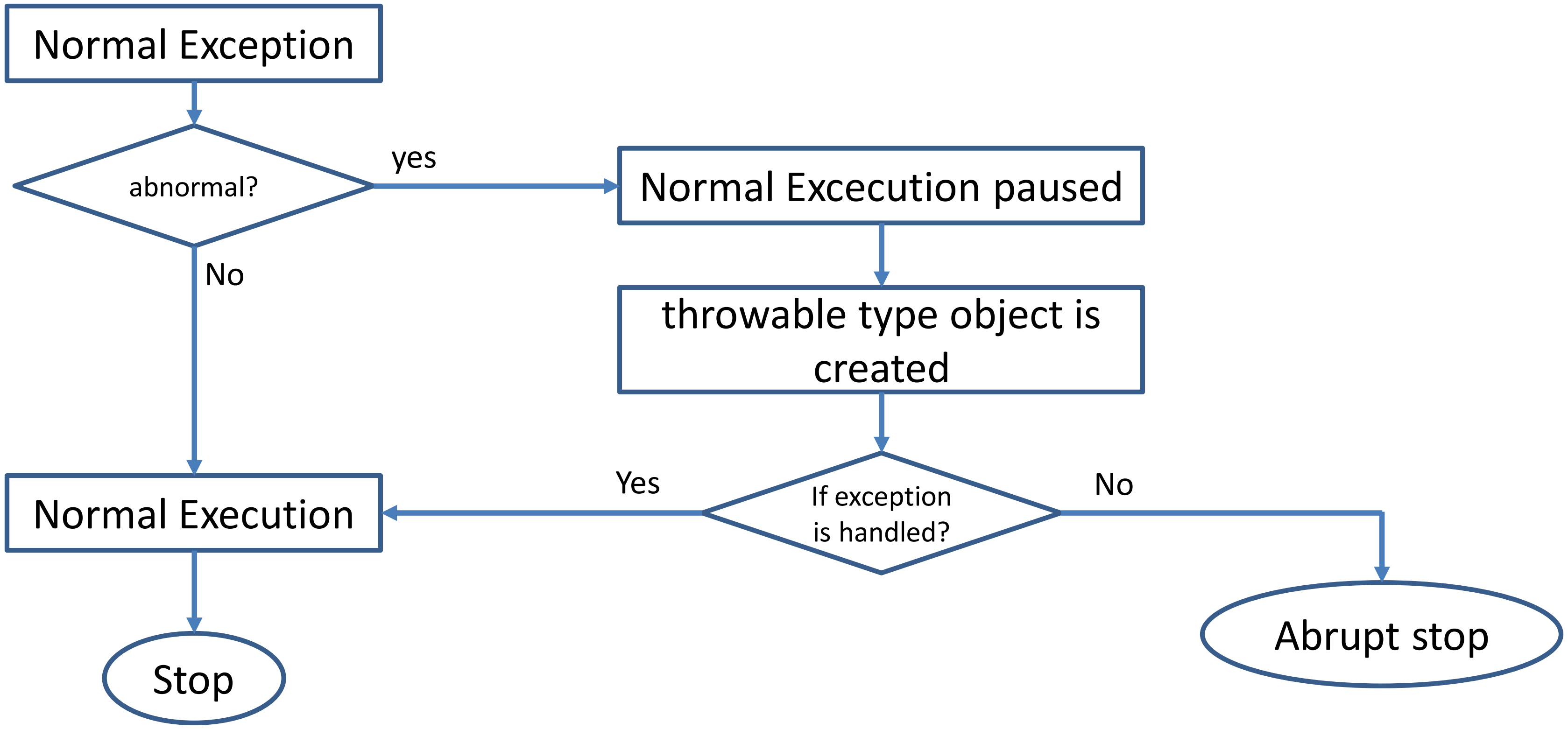
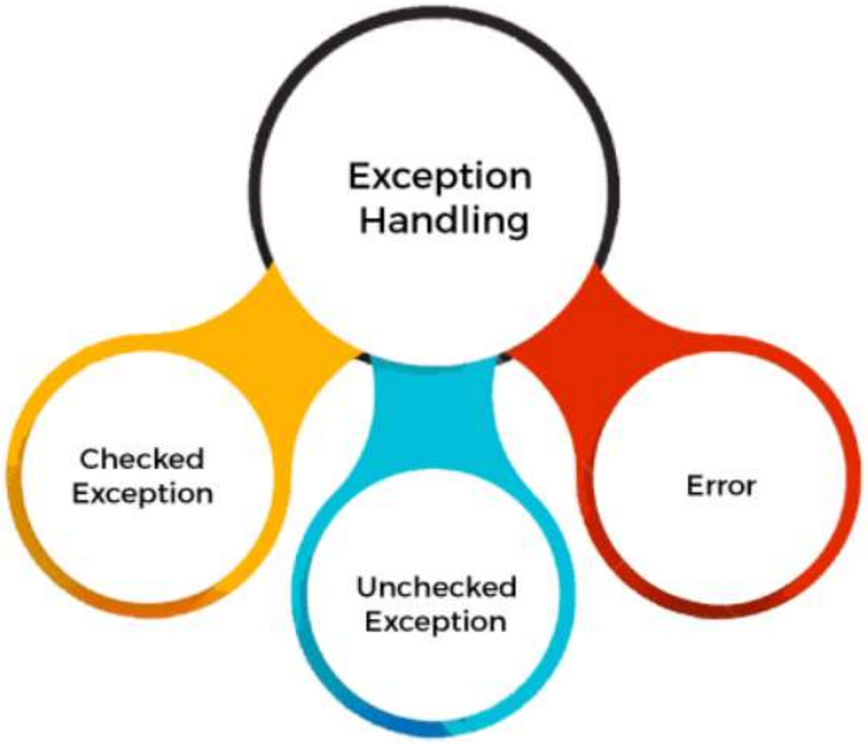
Exception

- The exception is a problem that occurs during execution of a program(Runtime).When an exception occurs, the execution of a program stops abruptly(Unexcepted stop).
- **What happens when exception Occurs?**



Types of Java Exceptions

- There are mainly two types of exceptions: checked and unchecked. An error is considered as the unchecked exception. However, according to Oracle, there are three types of exceptions namely:
- Checked Exception
- Unchecked Exception
- Error



<u>Checked Exception:</u>

- The complier aware exception is known as checked Exception i.e the complier knows the statement responsible for abnormal situations.Therefore,the complier forces the programmer to either handle it or declare the exception. If it is not done we will get an unexpected compile time error.

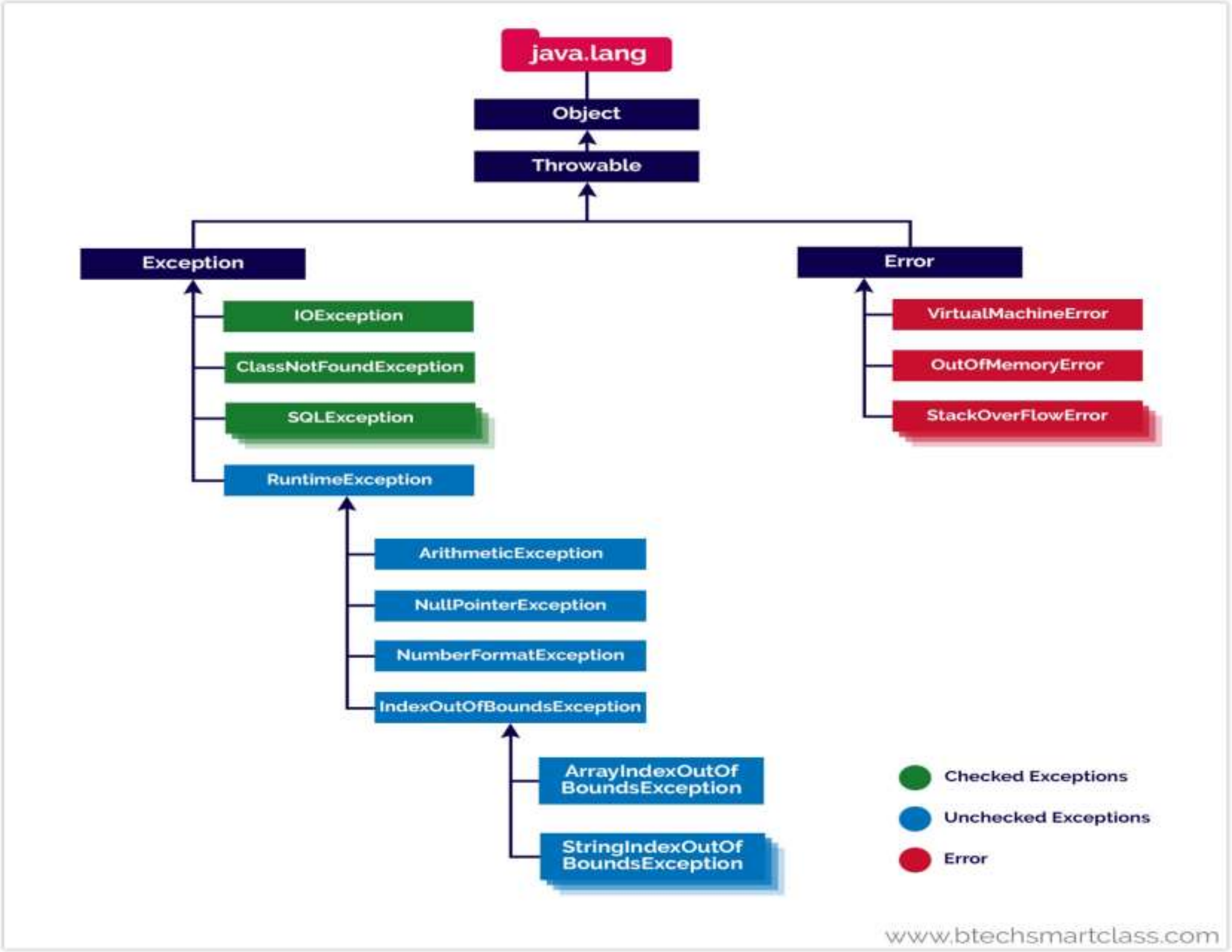
Example:FileNotFoundException

<u>Unchecked Exception:</u>

The complier unaware exception is known as the unchecked exception i.e the complier does not know the statements responsible for abnormal situations(Exception).Hence, the complier will not force the programmer either to handle or declare the exception.

Example:Arithemetic Exception

Exception hierarchy



Exception Handling

- The **Exception Handling in Java** is one of the powerful *mechanism to handle the runtime errors* so that the normal flow of the application can be maintained.
- 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 need to handle exceptions.

How to handle Exception??

In java, we can handle the exception with the help of try catch block.

Syntax to use try catch block:

```
try
{
//statements
}
catch(declare a variable of throwable type)
{
//statements
}
```

- Statements responsible for exception should be written in try block.
- When exception occurs,

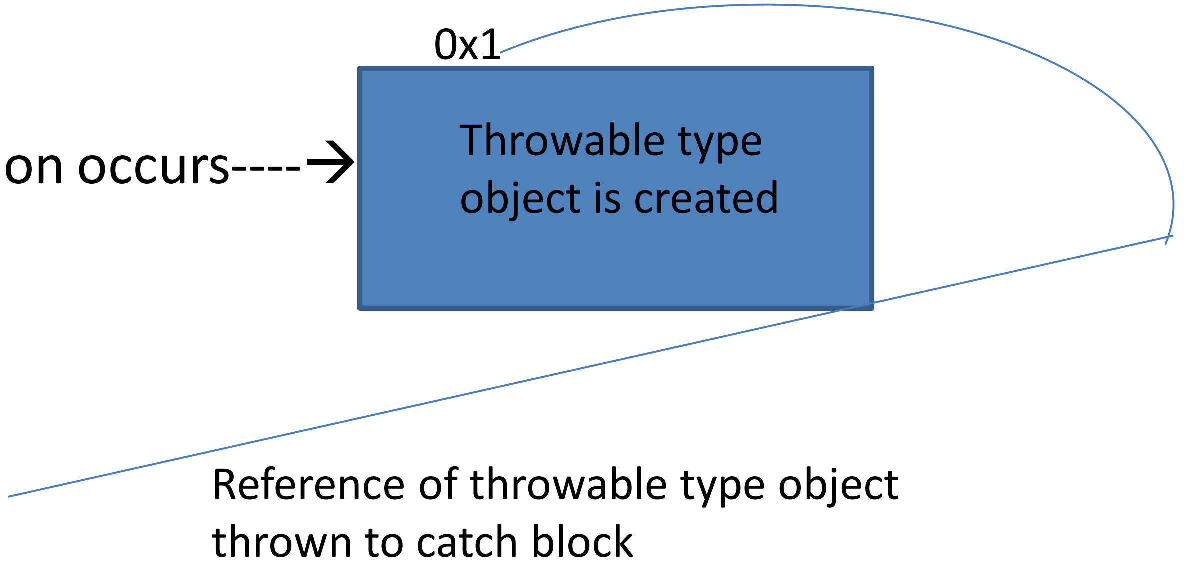
1.Execution of try block is stopped

2.Throwable type object is created

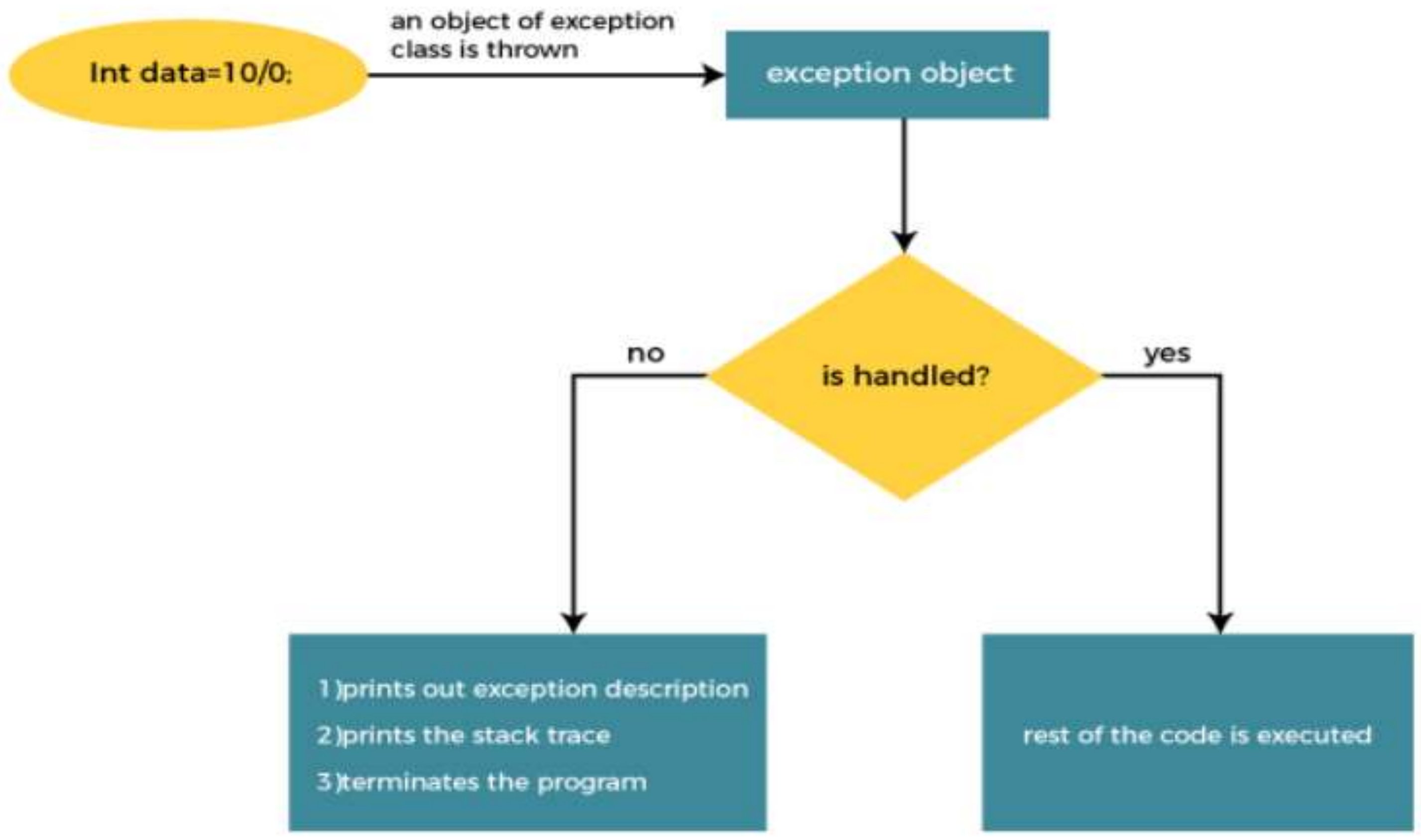
3.The reference of throwable type Object is created is passed to the catch block.

<u>Example</u>

```
try
{
Stmt1;
Stmt2;//Exception occurs---->
Stmt 3;
Stmt 4;
}
catch(variable)
```

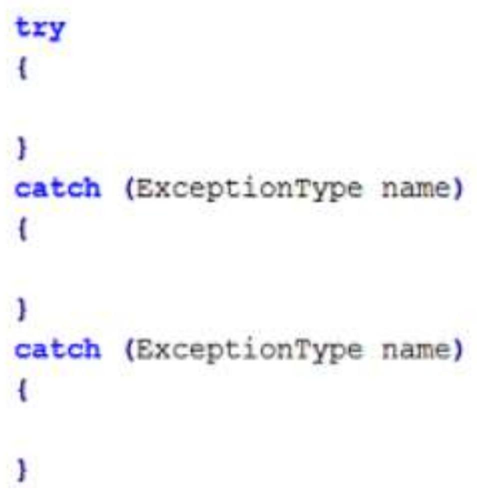

0x1
Throwable type object is created
Reference of throwable type object thrown to catch block

Internal Working of Java try-catch block


Int data=10/0;
an object of exception class is thrown
exception object
is handled?
no
yes
1)prints out exception description
2)prints the stack trace
3)terminates the program
rest of the code is executed

try with multiple catch blocks

- try block can be associated with more than one catch block.
- **Syntax:**



```
try
{

}
catch (ExceptionType name)
{

}
catch (ExceptionType name)
{

}
```

Note:

- The exception type object thrown from top to bottom order.

Rule: The order of catch block should be maintained in such that, child type should be at the top and parent type at the bottom.

Example:

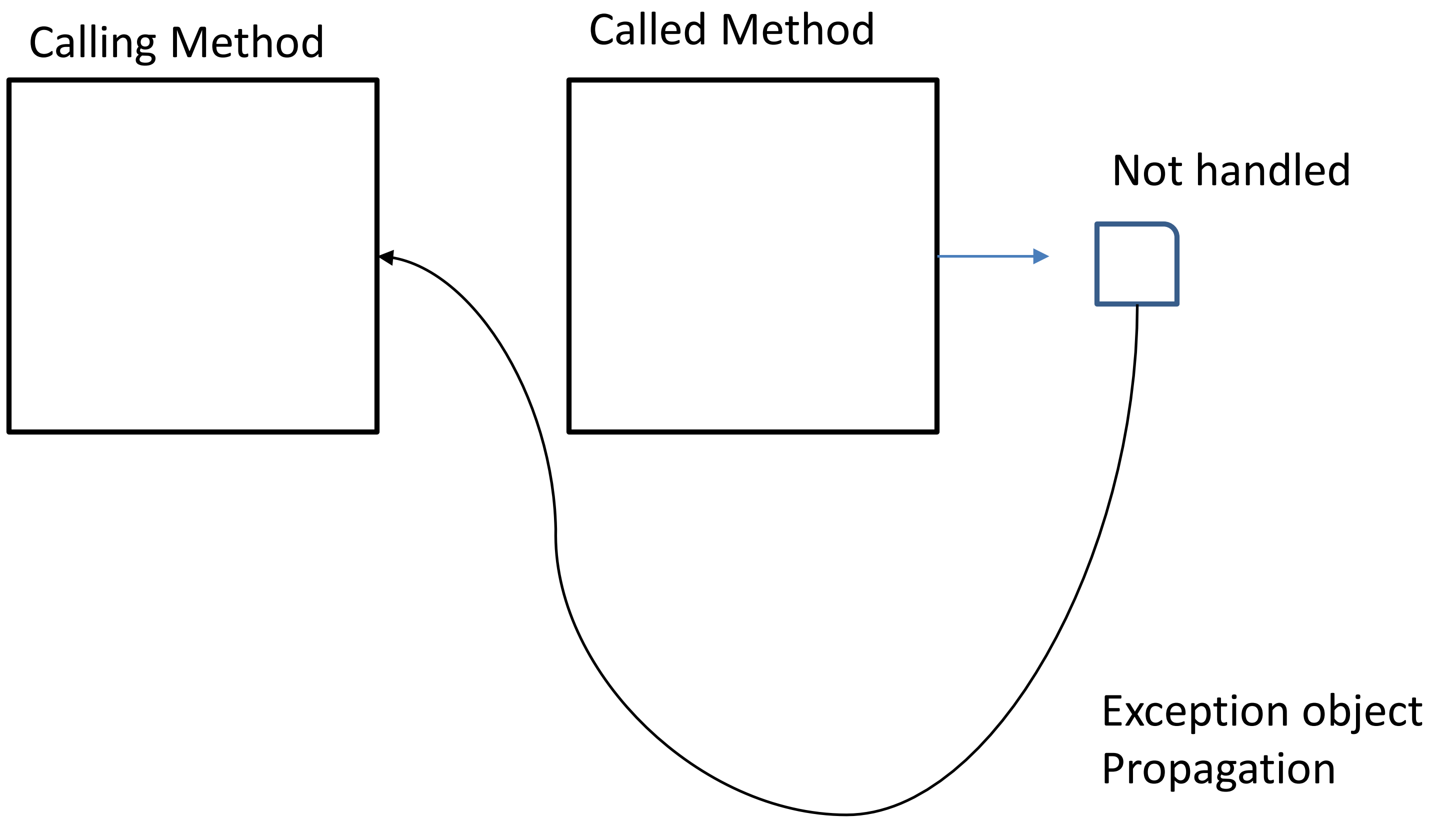
Case1:

```
try
{
        10/0;
}
catch(Exception e)
{
}
catch(ArithmeticException e)
{
}
```

CTE: Parent type is declared on top and child is at bottom.

Exception Propagation

- The movement of exception from called method to calling method when it is not handled is known as Exception object Propagation.



Exception occurred and handled by the calling method

```
class Case
{
public static void main(String[] args)
{
try
		{
		test();
		}
catch(ArthmeticException e)
{
		S.O.P("exception handled by calling method");
}
}
public static void test()
{
		int a=10/0;
}
}
```

Throwable :

- Throwable class is defined in java.lang package.

Note:

- In the throwable class, all the built in classes of the throwable type are overridden with toString() method such that it return fully qualified name of the class and reason for the exception.
- Important methods of throwable class

✓ String getMessage()

✓ void printStackTrace()

StackTrace

- It provides the order in which the exception is occurred and flow from top of the stack to the bottom of the stack.
- It contains fully qualified name, reason for the exception method name and line number.

Throw Keyword:

- It is a keyword
- It is used to throw the exception manually
- By using throw we can throw checked execption,unchecked exception .

Syntax

throw **new ArithmeticException();**

Example:

```
public class ThrowDemo
{
        public static void main(String[] args)
        {
        int a=10;
        int b=15;
        if(a>b)
        throw new ArithmeticException("manually thrown");
        else
        System.out.println("no exception");
        }
}
```

Throws keyword

- It is a keyword
- It is used to declare an exception to be thrown to the caller.

Note:

- Throws keyword should be used in the method declaration statement.

Syntax:

[modifier]returntype method Name(formal args) **throws** exeption1,exception2….

Note:

It the method declare an exception using throws keyword, then caller of the method must handle or throw the exception. If not we will get compile time error.

Example:

```
class ThrowsDemo
{
public static void main(String[] args)
{
try
		{
		test();
		}
catch (ArithmeticException e)
   {
		S.O.P(“exception handled by calling method”);
   }
}
public static void test() throws ArithematicException
{
		int a=10/0;
}
}
```

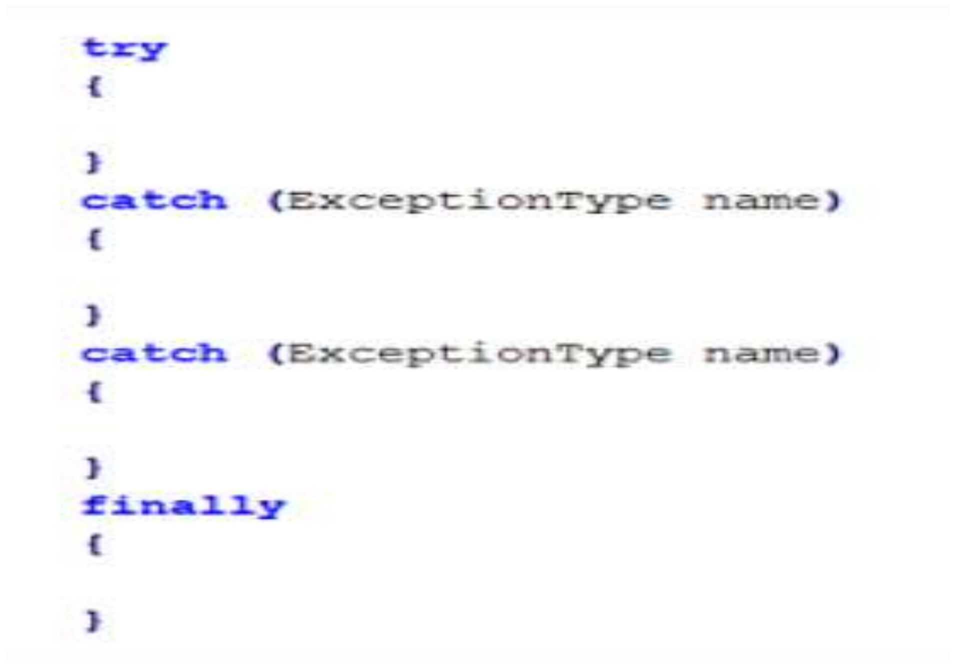
Finally block

```
finally
 {
 }
```

- It is a block which is used along with try-catch block.
- It will execute even if the exception is not handled

Note: We can use finally block with try block alone

Syntax:



```
try
{

}
catch (ExceptionType name)
{

}
catch (ExceptionType name)
{

}
finally
{

}
```