Exception Handling in JAVA :-

Error :- Error are wrong that make program goes to wrong.

* We can Categorize error into two category.

1) Compile Time Error
2) Runtime Error.

* Compile Time Error :- They are those error that occured due to bad coding. or incorrect Syntax we can Correct them by looking Compilation Error massage genrate by Compiler.

There are two types of Compiler time error.

(i) Syntax Error
(ii) Semantic Error.

Eg
```
class A
{
public static void main (String k[])
{
int a;
System.out.println(a.);
System.out.println("Hello");
system.out.println("Hello");
{
}
}
```

Semantic Error.

Syntax Error

C.T. Error

y+z=x

Some Compile time Error :- 1) double quotes missing.
2) Illegal use of keyword
3) semicolon missing
:
etc
All error generated by compiler are Compile time error

(2) Runtime Error :- They are those error that occur after Compilation or at the time of Execution.
We can Categorize runtime error into two category
(i) Logical Error
(ii) Exception

(1) Logical Error :- If your program produce unexpected output.

Ex
```
for (i=5; i<=5; i--)
    sopL(i);
```
↓
5
4
3
2
1
0
-1
:
infinite loop

(ii) Exception :- Exception are abnormal condition that arise by some code that break the rules of language like to provide a string to a variable that hold integer, divide any number by zero (0), wrong indexing of array ... etc

To free our program from exception we have to do two task.

(i) Identify which part of the program can Generate exception.

(ii) How to handle.

Eg
```
Import java.util.*;
Class T
{
public static void main (String k[])
{
Scanner ob = new Scanner (System.in);
int a,b;
SoPL ("Enter two No");
a = ob.nextInt ();
b = ob.nextInt ();
c = a/b;
SoPL ("Divide = "+c);
SoPL ("Hello");
}
}
```

Enter two No 12 -9 2 -6 Divide = 6 Hello	Enter two No 12 -9 0 -6 Divide by zero ~~Divide~~ Airthmatic Exception is occured	Enter two No RAM Input mismatch exception is occured.

```
import java.util.*;
class T
{
public static void main (String k[])
{
 Scanner ob= new Scanner (System.in);
      int a,b,c;
try
{ SopL("Enter two No");
   a= ob.nextInt();
   b= ob.nextInt();
    c= a/b;
 SopL("Divide = "+c);
}
Catch (Exception et)
 {SopL("please check values");
  }
  SopL("Hello");
 }
}
```

Enter two No 12 2 divide = 6 Hello	Enter two No 12 0 Please check values Hello	Enter two No RAM Please check values Hello

keyword for Exception Handling :- try
catch
Finally
throw
throws

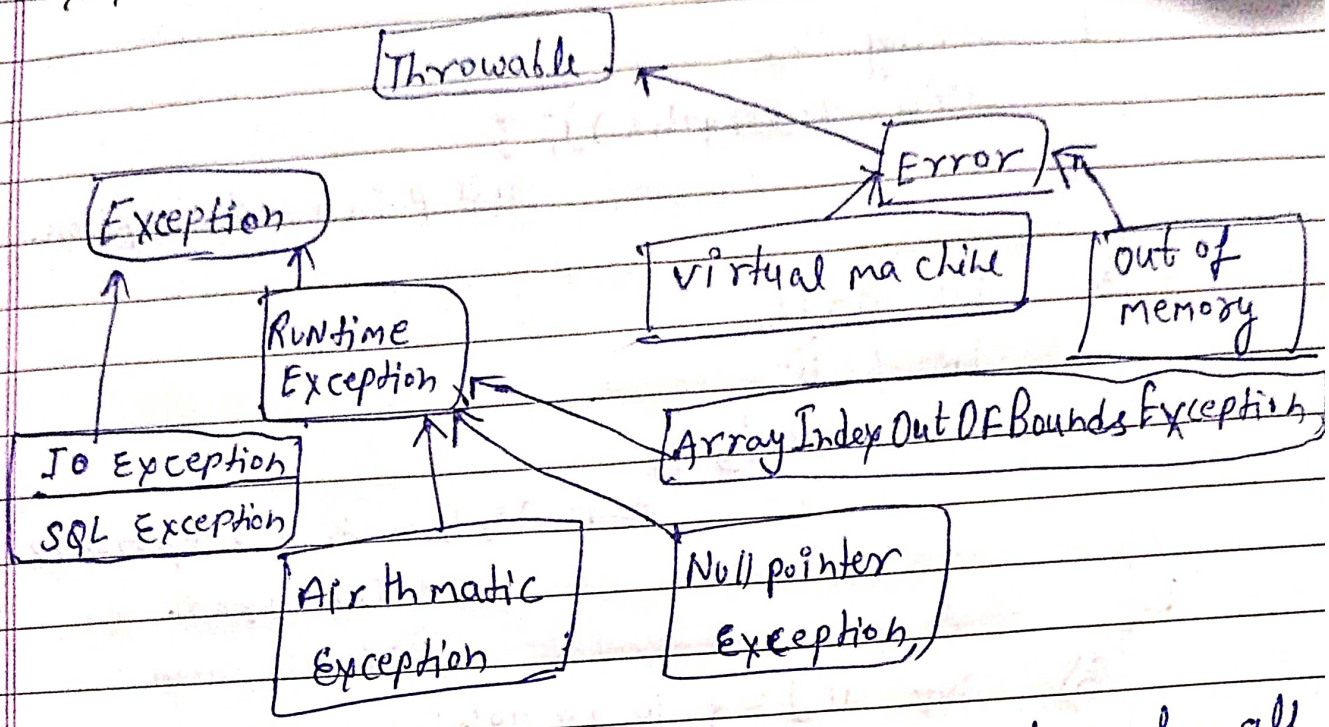
Ex

```
Scanner ob = new Scanner (System.in);
  int a,b,c;
try
{ SoPL ("Enter two No");
  a = ob.nextInt ();
  b = ob.nextInt ();
   c = a/b;
SoPL ("Divide = " +c);
}
catch (Arithmetic Exception et)
{
  SoPL ("Can not Divide any No. by zero");
  }
catch (Input Mismatch Exception obj)
  {
    SoPL ("Enter only integar Value");
  }
    SoPL (" Hello");
  }
}
```

keyword for Exception handling

try, catch, Finally, throw, throws

Exception classes Hierachical Inheritance :-



Note :- Throwable is the super class for all Exception classes in java

~~Type of Exception~~

Type of Exception :- There are three type of exception.

1. checked
2. Unchecked
3. Error

1) checked Exception :- They checke Compiler IOException & SQLException are the example of checked exception.

2) unchecked :- They do not by Compiler like Airthmatic exception, Nullpointer exception ..... etc.

3) Error :- We Can handle exception but Can not handle Error.

Exception Class example :-

1. String s;
   SOPL(s.length());
   Null pointer Exception.

2. int a=10;
   int b=0;
   int c=a/b;
   Arithmatic Exception is occured.

3) int a[] = {10, 20, 40};
   SOPL(a[0]); ⇒ 10
   SOPL(a[4]);
   Array Index Out Bound Exception is occured.

Finaly :- This block is always executed either exception is occure or not.

1) try
{
}
Catch (Exception et)
{
}
Finally
{
}

2) try
{
}
finaly
{
}

3) try
{
}
C.T. Error

try
{
}
Catch (Exception)
{
}

4) try
{
}
Catch
{
} C.T.Error

5) try
{
}
stat;
Catch (Exception ob) → C.T.Error
{
}

Ex 1) try
{
}
Catch (Airthmetic Exception obj)
{
}
catch (Exception kp)
{
}

2) try
{
}
Catch (Exception ob)
{
}
Catch (Airthmetic Exception obj) → C.T. Error
{
}

③
```
try.
{ }
finaly
{ }
```

④
```
try
{ }
catch (Exception e6)
{ }
finaly
{ }
```

throws keyword :- It is used to UnHandle the exception.

```
Class A
{
  public static void main (String k[])
  {
    FileWriter fw = new FileWriter ("abc.txt");
    :
  }
}
```

C.T. Error

(File handling code Must written in try & catch)

2)
```
Class A
{
public static void main (String k[])
{
try
{
FileWriter fo = new FileWriter ("abc.txt");
:
}
Catch (IOException obj)
{
}
}
```

3)
```
Class A
{
public static void main (String k[])
                    Throws Excepti
{
. . . .
. . . .
}
}
```

Throw keyword :- It is used by programmer in rare cases to throw the exception.

Eg
```
import java.util.*;
class T
{
public static void main (String k[])
{
Scanner ob = new Scanner (System.in);
try
{
int i, a, int f=1;
SoPL ("Enter a No");
a= ob.nextInt();
if (a<0)
throw new Exception ("Enter only positive Number");
for (i=a; i>=1; i--)
f= f*i;
SoPL ("fact = " +f);
}
catch (Exception obj)
{
SoPl (obj);
}
}
}
```

S → a	-5 a
fact = 120	Enter only positive Number