Web Technology
(KCS-602)
Unit 1

Prepared By

Abhishek Kesharwani

Assistant Professor,UCER Naini,Allahabad

Exception Handling in Java

- The **exception handling in java** is one of the powerful mechanism to handle the runtime errors so that normal flow of the application can be maintained.
- In java, exception is an event that disrupts the normal flow of the program. It is an object which is thrown at runtime.
- Exception Handling is a mechanism to handle runtime errors such as ClassNotFound, IO, SQL, Remote etc.

Advantage of Exception Handling

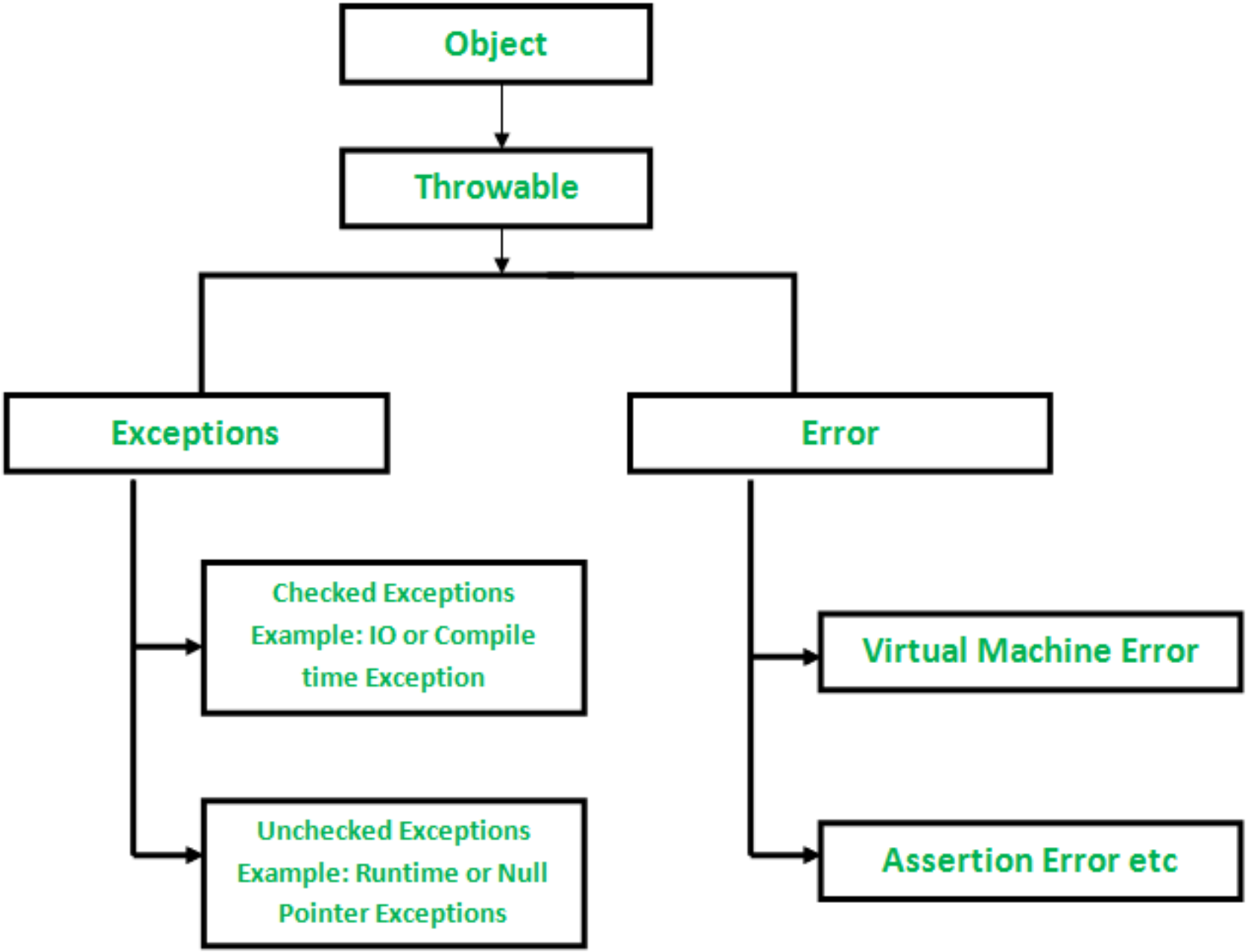
- The core advantage of exception handling is **to maintain the normal flow of the application**. Exception normally disrupts the normal flow of the application that is why we use exception handling.
- statement 1;
- statement 2;
- statement 3;
- statement 4;
- statement 5;//exception occurs
- statement 6;
- statement 7;
- statement 8;
- statement 9;
- statement 10;

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

Types of Exception

There are mainly two types of exceptions: checked and unchecked where error is considered as unchecked exception.

- Checked Exception
- Unchecked Exception
- Error


Object
Throwable
Exceptions
Error
Checked Exceptions
Example: IO or Compile time Exception
Unchecked Exceptions
Example: Runtime or Null Pointer Exceptions
Virtual Machine Error
Assertion Error etc

Difference between checked and unchecked exceptions

1) Checked Exception

The classes that extend Throwable class except RuntimeException and Error are known as checked exceptions e.g.IOException, SQLException etc. Checked exceptions are checked at compile-time.

2) Unchecked Exception

The classes that extend RuntimeException are known as unchecked exceptions e.g. ArithmeticException, NullPointerException, ArrayIndexOutOfBoundsException etc. Unchecked exceptions are not checked at compile-time rather they are checked at runtime.

3) Error

Error is irrecoverable e.g. OutOfMemoryError, VirtualMachineError, AssertionError etc.

Java Exception Handling Keywords

There are 5 keywords used in java exception handling.

- try
- catch
- finally
- throw
- throws

Java try block

- Java try block is used to enclose the code that might throw an exception. It must be used within the method.
- Java try block must be followed by either catch or finally block.

Syntax of java try-catch

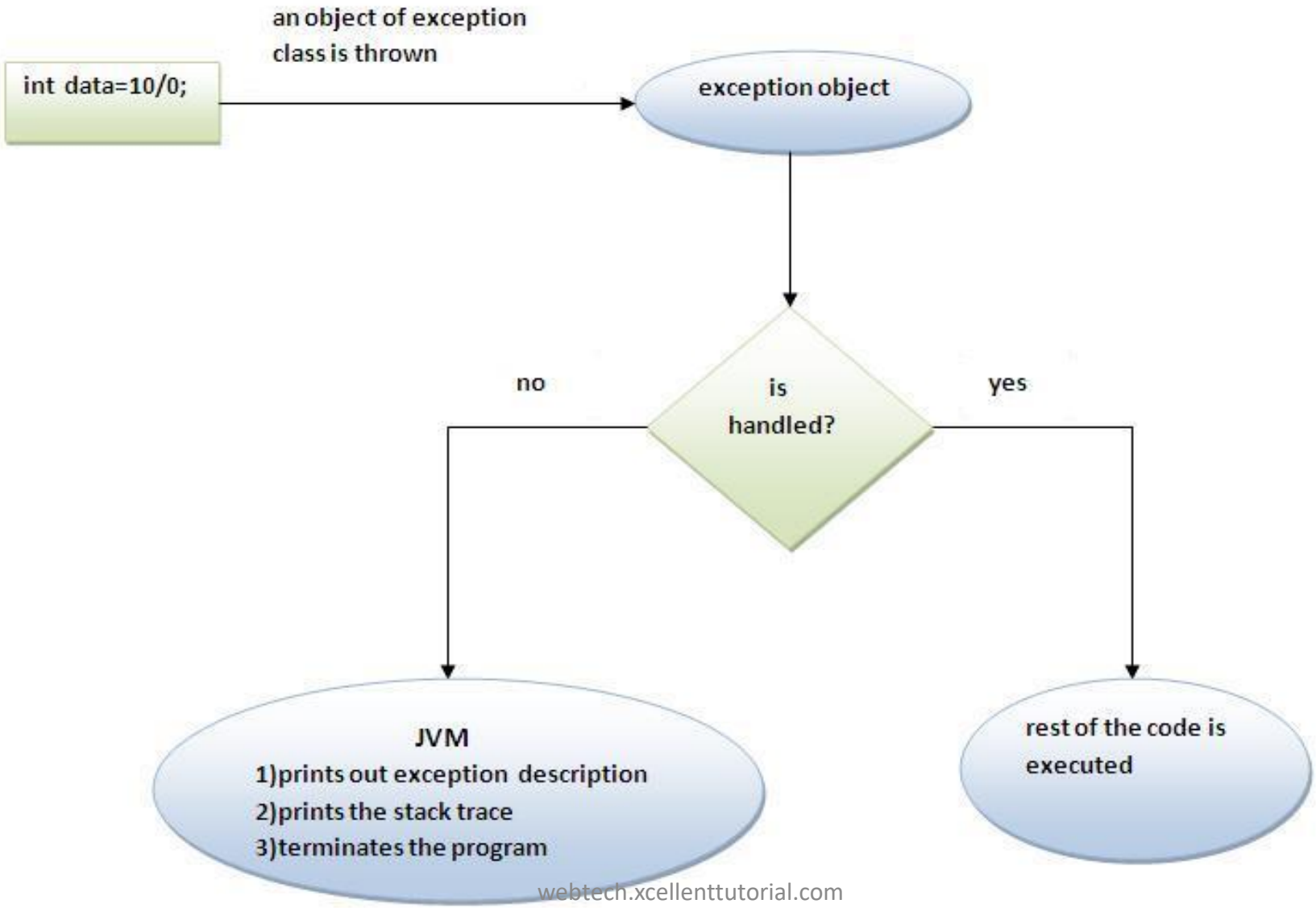
```
try{
//code that may throw exception
}catch(Exception_class_Name ref){}
```

Syntax of try-finally block

```
try{
//code that may throw exception
}finally{}
```

- Java catch block is used to handle the Exception. It must be used after the try block only.
- You can use multiple catch block with a single try.

Internal working of java try-catch block



The JVM firstly checks whether the exception is handled or not. If exception is not handled, JVM provides a default exception handler that performs the following tasks:

- Prints out exception description.
- Prints the stack trace (Hierarchy of methods where the exception occurred).
- Causes the program to terminate.

Java Multi catch block

```
public class TestMultipleCatchBlock{
  public static void main(String args[]){
   try{
    int a[]=new int[5];
    a[5]=30/0;
   }
   catch(ArithmeticException e){System.out.println("task1 is co
     mpleted");}
   catch(ArrayIndexOutOfBoundsException e){System.out.print
     ln("task 2 completed");}
   catch(Exception e){System.out.println("common task compl
     eted");}
     System.out.println("rest of the code...");
 }}
```

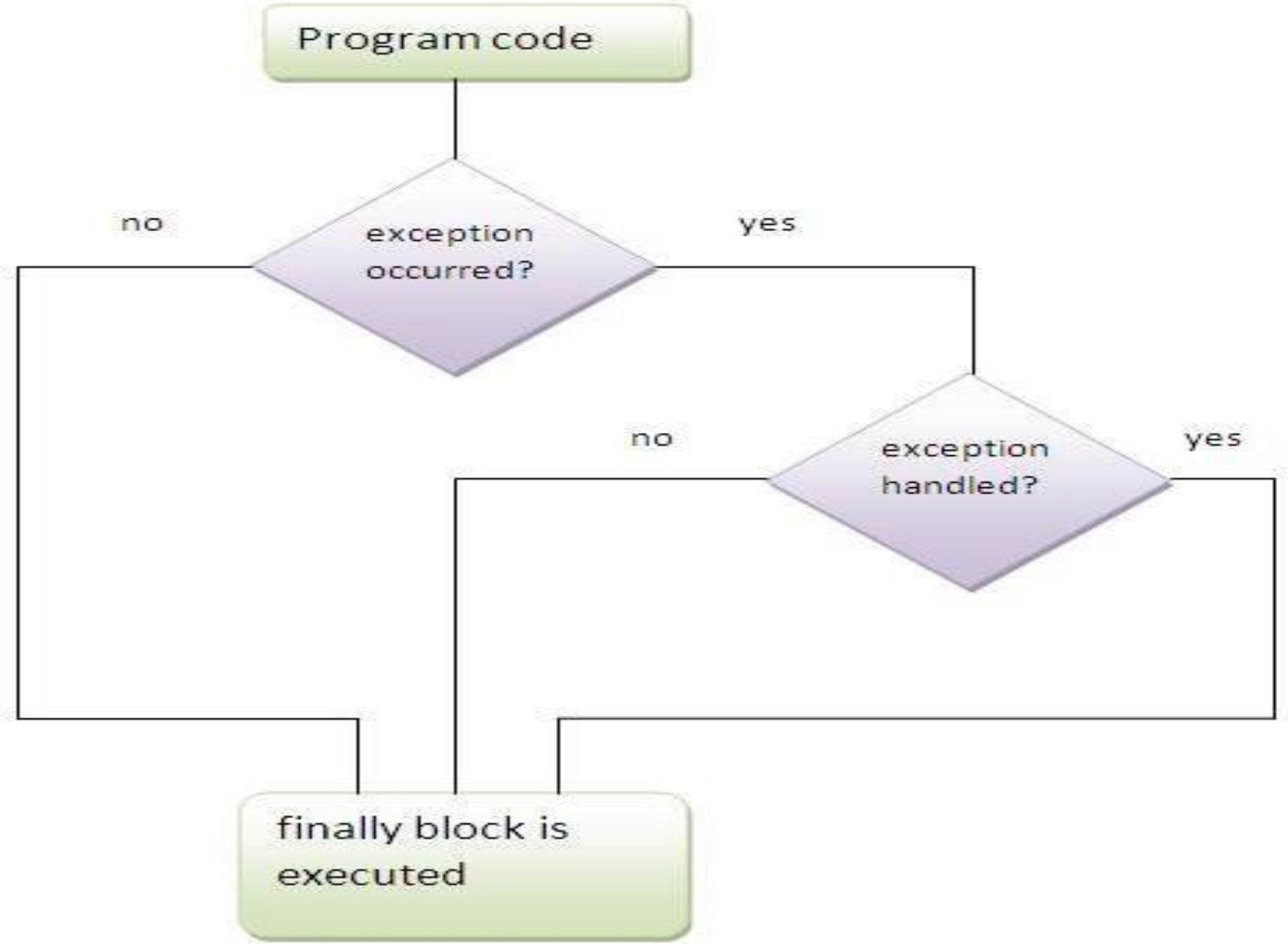
- **Rule: At a time only one Exception is occured and at a time only one catch block is executed.**
- **Rule: All catch blocks must be ordered from most specific to most general i.e. catch for ArithmeticException must come before catch for Exception**

Java Nested try block

```
class Excep6{
 public static void main(String args[]){
  try{
    try{
     System.out.println("going to divide");
     int b =39/0;
    }catch(ArithmeticException e){System.out.println(e);}
       try{
    int a[]=new int[5];
    a[5]=4;
    }catch(ArrayIndexOutOfBoundsException e){System.out.println
     (e);}
        System.out.println("other statement);
  }catch(Exception e){System.out.println("handeled");}
    System.out.println("normal flow..");
} }
```

Java finally block

- **Java finally block** is a block that is used to execute important code such as closing connection, stream etc.
- Java finally block is always executed whether exception is handled or not.
- Java finally block must be followed by try or catch block.


Program code
exception occurred?
no
yes
exception handled?
no
yes
finally block is executed

Why use java finally

Finally block in java can be used to put "cleanup" code such as closing a file, closing connection etc.

Rule: For each try block there can be zero or more catch blocks, but only one finally block.

Note: The finally block will not be executed if program exits(either by calling System.exit() or by causing a fatal error that causes the process to abort).

Java throw keyword

The Java throw keyword is used to explicitly throw an exception.

We can throw either checked or uncheked exception in java by throw keyword. The throw keyword is mainly used to throw custom exception.

throw exception;

throw new IOException("sorry device error);

```
public class TestThrow1{
   static void validate(int age){
     if(age<18)
      throw new ArithmeticException("not valid");
     else
      System.out.println("welcome to vote");
   }
   public static void main(String args[]){
      validate(13);
      System.out.println("rest of the code...");
  }
}
```

Output:

```
Exception in thread main
    java.lang.ArithmeticException:not valid
```

Java throws keyword

- The **Java throws keyword** is used to declare an exception. It gives an information to the programmer that there may occur an exception so it is better for the programmer to provide the exception handling code so that normal flow can be maintained.
- Exception Handling is mainly used to handle the checked exceptions. If there occurs any unchecked exception such as NullPointerException, it is programmers fault that he is not performing check up before the code being used.

Syntax of java throws

return_type method_name() **throws** exception_class_name

{

//method code

}

Rule: If you are calling a method that declares an exception, you must either caught or declare the exception.

There are two cases:

- **Case1:**You caught the exception i.e. handle the exception using try/catch.
- **Case2:**You declare the exception i.e. specifying throws with the method.

No.	Throw	throws
1)	Java throw keyword is used to explicitly throw an exception.	Java throws keyword is used to declare an exception.
2)	Checked exception cannot be propagated using throw only.	Checked exception can be propagated with throws.
3)	Throw is followed by an instance.	Throws is followed by class.
4)	Throw is used within the method.	Throws is used with the method signature.
5)	You cannot throw multiple exceptions.	You can declare multiple exceptions e.g. public void method()throws IOException,SQLException.