Exceptions in Java

- An exception is an abnormal condition that is caused during program execution
- When java interpreter encounters such an error, an exception object is created and is thrown.

- Exception handling mechanism performs the following tasks
- Find the error(**Hit** the exception)
- Inform that an error has occurred(**Throw** the exception)
- Receive the error of information(**Catch** the exception)
- Take corrective actions(**Handle** the exception)

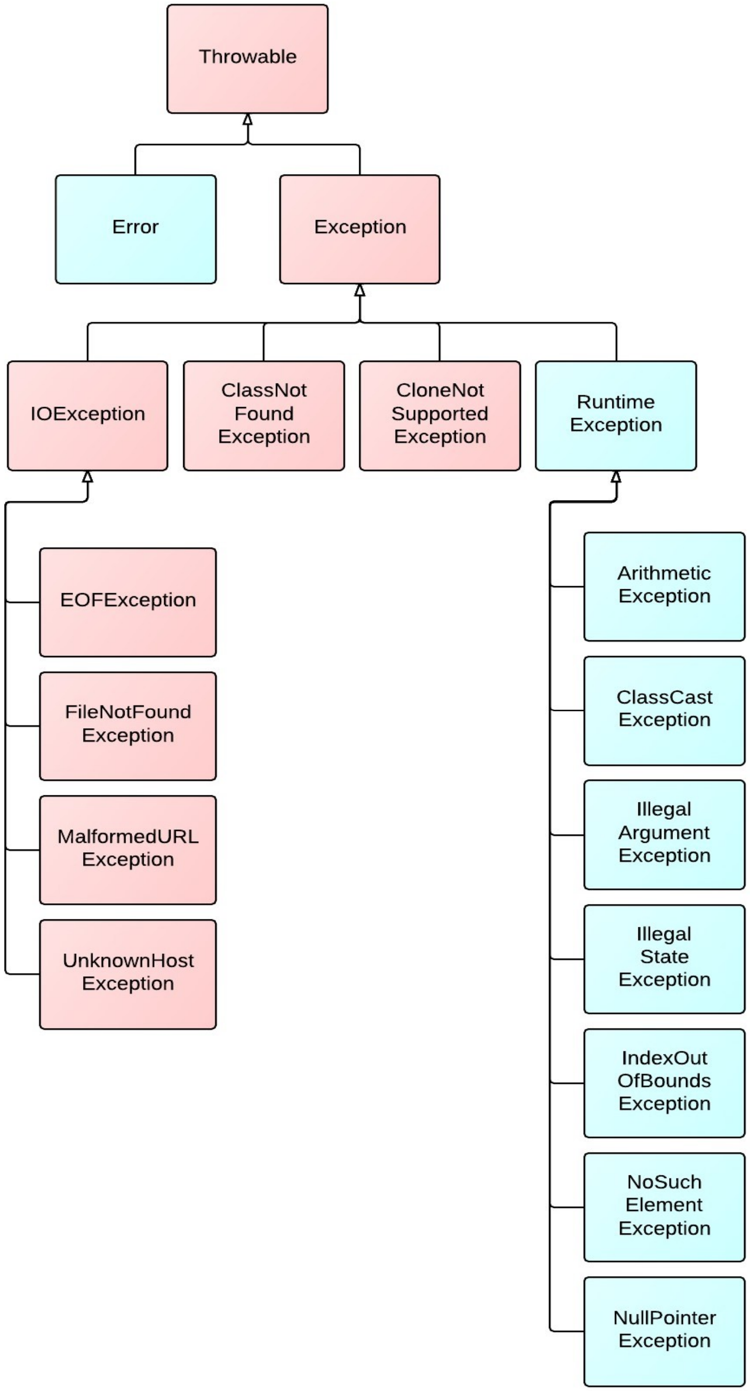
Advantage of Exception Handling

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.

- **Types of Exception**
- There are mainly two types of exceptions: checked and unchecked:
- Checked Exception
- Unchecked Exception
- Error

Checked

Unchecked



Throwable

Error

Exception

IOException

ClassNot Found Exception

CloneNot Supported Exception

Runtime Exception

EOFException

FileNotFound Exception

MalformedURL Exception

UnknownHost Exception

Arithmetic Exception

ClassCast Exception

Illegal Argument Exception

Illegal State Exception

IndexOut OfBounds Exception

NoSuch Element Exception

NullPointer Exception

Checked Exception

- Checked exceptions are checked at compile-time.
- A checked exception is an exception which the Java source code must deal with, either by catching it or declaring it.
- Checked exceptions are generally caused by faults outside of the code itself - missing resources, networking errors, and problems with threads

Name	Description
IOException	While using file input/output stream related exception
SQLException.	While executing queries on database related to SQL syntax
DataAccessException	Exception related to accessing data/database
ClassNotFoundException	Thrown when the JVM can’t find a class it needs, because of a command-line error, a classpath issue, or a missing .class file
InstantiationException	Attempt to create an object of an abstract class or interface.

Unchecked Exception

- Unchecked exceptions are not checked at compile time.
- if your program is throwing an unchecked exception and even if you didn’t handle/declare that exception, the program won’t give a compilation error.
- When an unchecked exception is thrown, it is usually caused by a misuse of code

Name	Description
NullPointerException	Thrown when attempting to access an object with a reference variable whose current value is null
ArrayIndexOutOfBound	Thrown when attempting to access an array with an invalid index value (either negative or beyond the length of the array)
IllegalArgumentException.	Thrown when a method receives an argument formatted differently than the method expects.
NumberFormatException	Thrown when a method that converts a String to a number receives a String that it cannot convert.
ArithmeticException	Arithmetic error, such as divide-by-zero.

Java Exception Handling Keywords

- There are 5 keywords used in java exception handling.
- try
- catch
- finally
- throw
- throws

Java try block

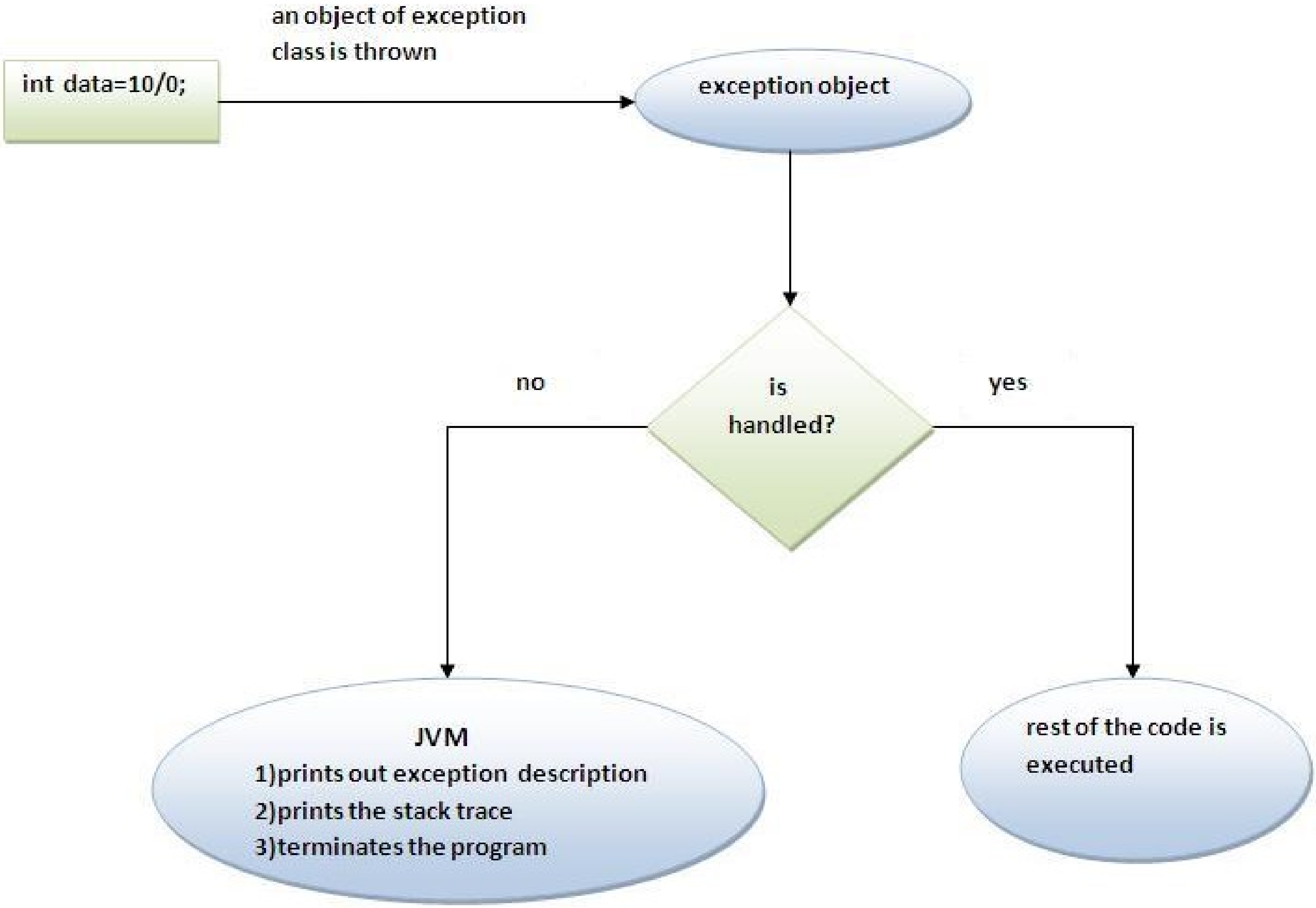
❖Java try block is used to enclose the code that might throw an exception. It must be used within the method.

❖Java try block must be followed by either catch or finally block.

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

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

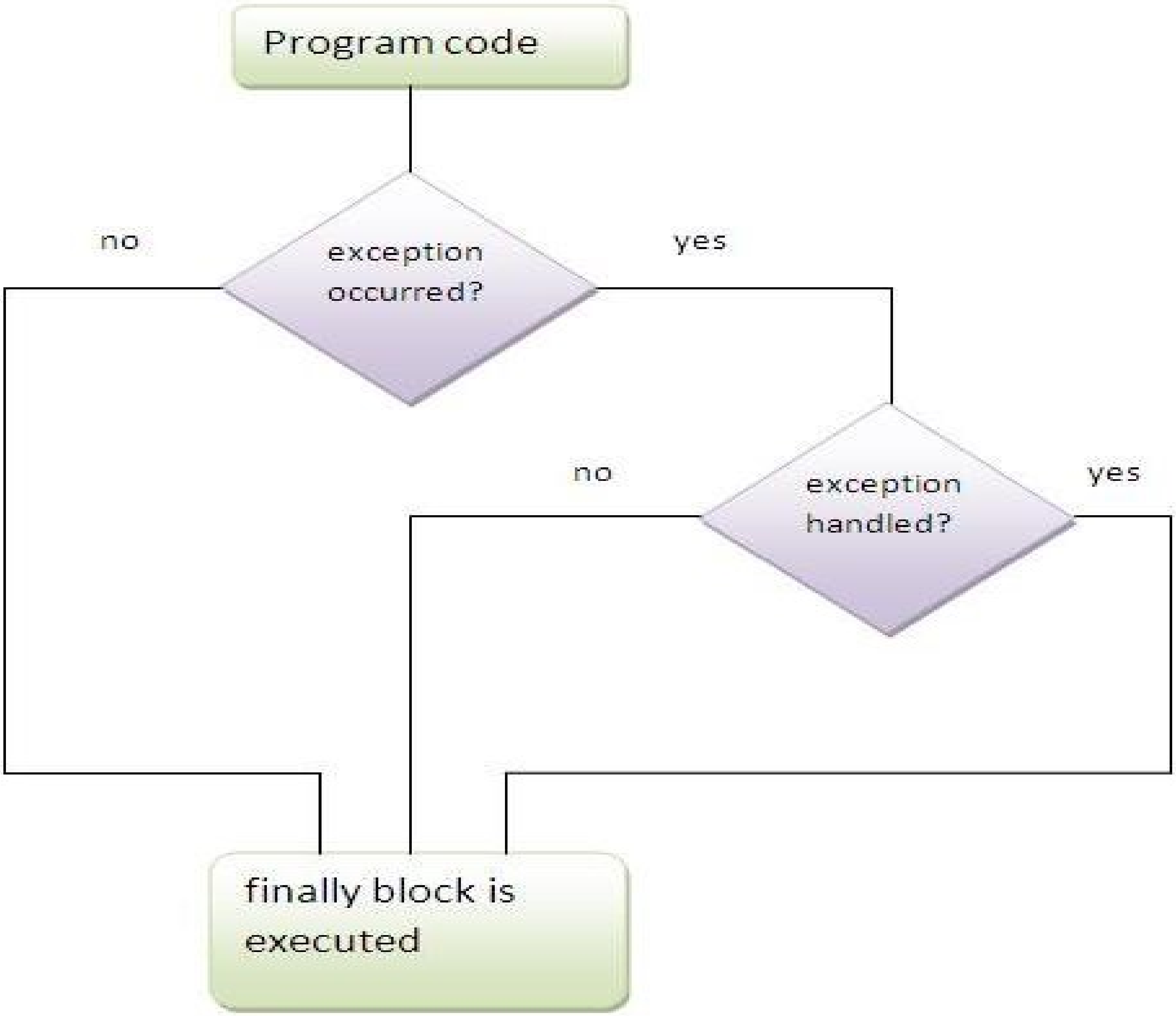
- **Java catch block**
- Java catch block is used to handle the Exception.
- It must be used after the try block only
- You can use multiple catch block with a single try


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

- **Java Multi catch block**
- If you have to perform different tasks at the occurrence of different Exceptions, use java multi catch block.
- At a time only one Exception is occurred and at a time only one catch block is executed.
- All catch blocks must be ordered from most specific to most general i.e. catch for ArithmeticException must come before catch for Exception

Java finally block

- **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 follows try or catch block.


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

- When an exception occurs the execution of the program is transferred to an appropriate exception handler. The **try-catch-finally** block is used to handle the exception.
- The code in which the exception may occur is enclosed in a try block, also called as a guarded region.
- The **catch clause** matches a specific exception to a block of code which handles that exception.
- And the cleanup code which needs to be executed no matter the exception occurs or not is put inside the **finally block**

Can we have try without catch

- It is possible to have try block without catch block by using finally block
- Java supports try with finally block
- As we know finally block will always executes even there is an exception occurred in try block, except System.exit() it will execute always.
- We can place logic like connections closing or closing streams in finally.

Java throw keyword

- The Java throw keyword is used to explicitly throw an exception.
- We can throw either checked or unchecked exception in java by throw keyword. The throw keyword is mainly used to throw custom exception.

Java throws keyword

- The **Java throws keyword** is used to declare an exception.
- Any method that is capable of causing exceptions must list all the exceptions possible during its execution, so that anyone calling that method gets a prior knowledge about which exceptions are to be handled
- Syntax:

```
return_type method_name() throws exception_class_name{
//method code
}
```

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

Which exception should be declared

Ans) checked exception only, because:

unchecked Exception: under your control so correct your code.

error: beyond your control e.g. you are unable to do anything if there occurs OutOfMemoryError or StackOverflowError.

Difference between throw and throws

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

- **What is the difference between Error and Exception?**
- Ans) An error is an irrecoverable condition occurring at runtime. Such as OutOfMemory error. These JVM errors you can not repair them at runtime, the execution of application will come to a halt and is not recoverable.
- While exceptions are conditions that occur because of bad input or human error etc. e.g. FileNotFoundException will be thrown if the specified file does not exist. Or a NullPointerException will take place if you try using a null reference. In most of the cases it is possible to recover from an exception (probably by giving user a feedback for entering proper values etc.)