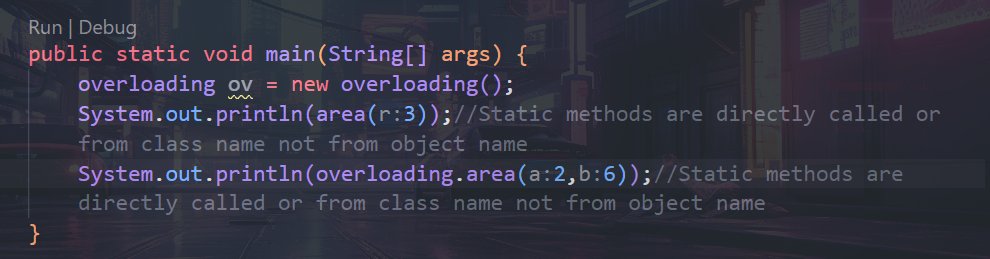
* We cant use super keyword inside static
* Inheritance
* Abstraction

Polymorphism

* Important feature of Object Oriented Programming
* Poly 🡪 Many
* Morphism 🡪 Form
* Hence Many form
* Types:-
  1. Compile time polymorphism
     + Method Overloading
       - Used again and again
       - Method name should be same
       - Method return type, No of Arguments and types may or may not be different

1. Static methods are directly called or from class name not from object name



* + - Method Overriding
  1. Run Time Polymorphism

Method Overloading:-

class overloading {

    static float area(int r){

        return 3.14f\*r\*r;

    }

    static int area(int a,int b){

        return a\*b;

    }

    public static void main(String[] args) {

        overloading ov = new overloading();

        System.out.println(area(3));//Static methods are directly called or from class name not from object name

        System.out.println(overloading.area(2,6));//Static methods are directly called or from class name not from object name

    }

}

Method Overriding

* Method name should be same.
* Method return type, No or Arguments and types of arguments should be same
* It works in Inheritance.
* Jiska object uske methods ko call.
* class overridedemo{
* void draw(int m){ //no of arguments, return type of method and type of arguments should be same
* System.out.println("Base Class draw method");
* }
* }
* class overridding extends overridedemo {//no of arguments, return type of method and type of arguments should be same
* void draw(int m){
* System.out.println("Child Class draw method");
* }


* public static void main(String[] args) {
* overridding ov = new overridding();
* ov.draw(0);
* }
* }

Final Keyword

* Final is keyword in java. When final is used within variable, method and class.
* When final is used within variable then it cannot be modified.
* When final is used within method then method cannot be overridden
* When final is used with class then class cannot be inherited.

Exception Handling in JAVA

* Exception is run time error in Program.
* Exception is an abnormal condition.
* Exception is an event that disrupts normal of program.
* Exception in java is an object that is thrown during runtime.
* Types 🡪
  + Checked Exception🡪 These are checked during compile time.
  + Unchecked Exception🡪 These are not checked during compile time.

A computer screen shot of a diagram

Description automatically generated



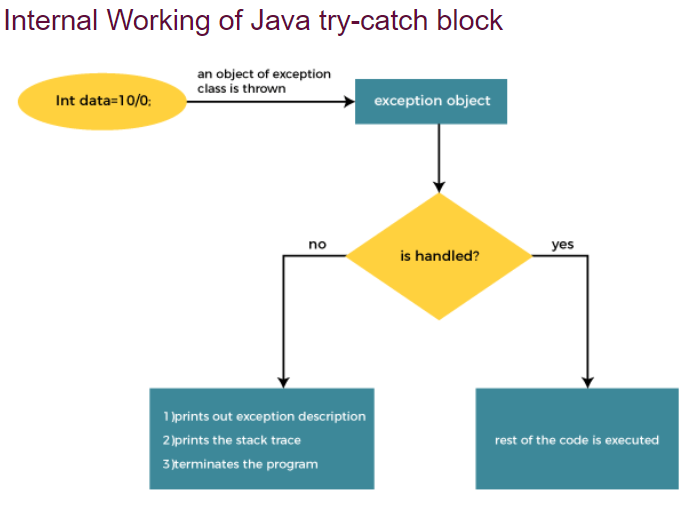
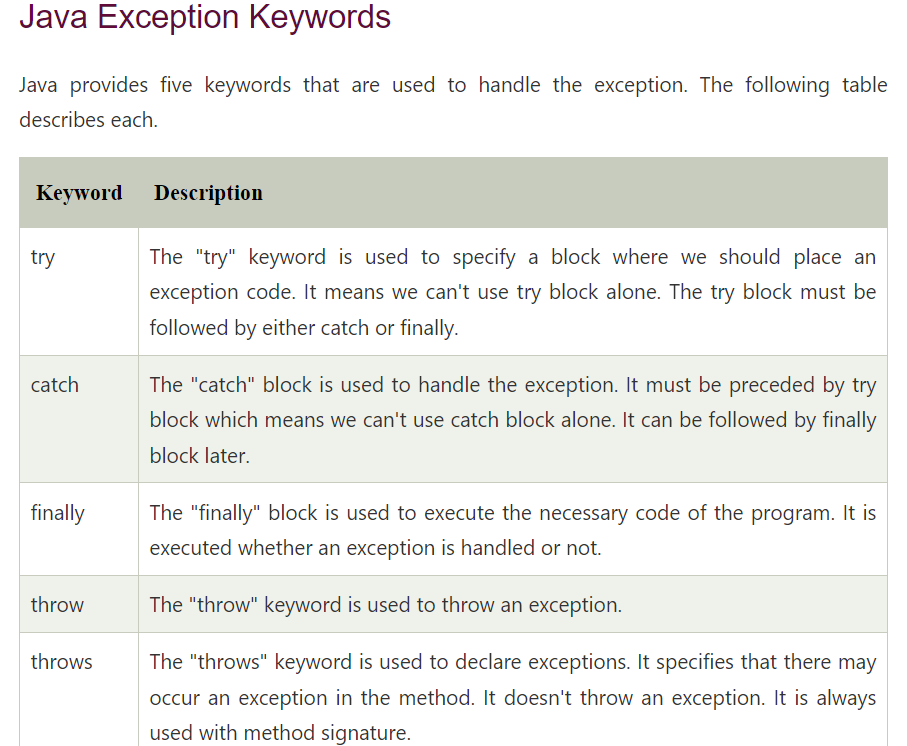
Unchecked

Exception

The java.lang.Throwable class is the root class of Java Exception hierarchy inherited by two subclasses: Exception and Error. The hierarchy of Java Exception classes is given below:

#### Note: Every subclass of Error and RuntimeException is an unchecked exception in Java. A checked exception is everything else under the Throwable class.

* Avoid throwing Checked Exception using throw keyword. Only throw unchecked exceptions by throw keyword. If you want to throw checked exceptions then use throws keyword with method signature.



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.

But if the application programmer handles the exception, the normal flow of the application is maintained, i.e., rest of the code is executed.



Exception Handling Mechanism

try…catch…finally

* In “try” Block exception code is written.
* When exception is raised from try block it is caught in ‘catch’ block. It means that the catch block is responsible to handle the exception.
* “finally,” block is used to include the code in every case when the exception handled or not by the catch block
* Any one catch block is executed or else try nested try catch block

Throw vs Throws keywords in java.

|  |  |
| --- | --- |
| throw | throws |
| 1. It is used to throw an exception explicitly. 2. It is followed by object of a class. 3. It is used inside the method. 4. Throw the exception object. | * 1. It is used with method signature.   2. Multiple Exception can be thrown in method signature by throws keyword.   3. Used for throwing multiple exception after the name of the method   4. Or using throws keyword the Exception is Declared with methods. |
|  |  |

If whole method throws exception, the it should be handled at the point from where it is called i.e. here from main method and it must be handled using try..catch.

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