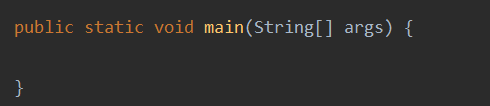
**Can we override static method ?**

In case of static method binding happens at compile time. But, for overriding a method, binding needs to be happen at runtime. That’s why static method can not be overridden.

We can overload static method as overloading happens at compile time.

**Can we overload main method ?**

Yes, we can overload main method. But, JVM will only look for pre defined method with below definition.



**Highlight important points on static method ?**

1. Static methods can not be overridden as they are class dependent.
2. A static method can be called without an object or instance.
3. Static methods will always share the same definition with all instances.
4. Mostly utility classes used to have static methods to define common functionality which can be consumed by multiple classes.
5. **Static methods will always access static data member.**
6. this and super not allowed within static method and block.

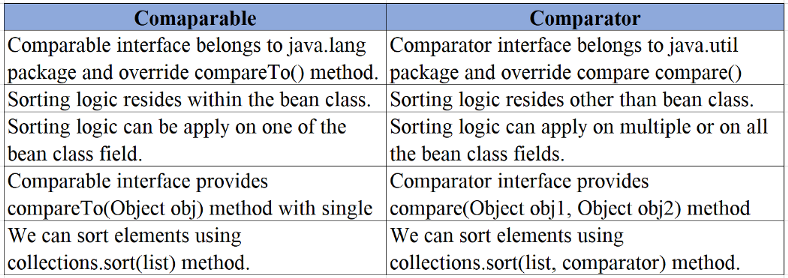
**Is it possible to have try block without catch ?**

Yes, it is possible to have try block without catch block if we use finally block in place of catch block.

**How to stop finally block execution ?**

Call **System.exit()** method at the end of the catch block to stop program execution.

**Comparable vs Comparator**



**What will be the output for below code ?**

public static void main(String[] args) {  
 System.*out*.println(10 + 10 + "Test" );  
 System.*out*.println("Test" + 10 + 10);  
 System.*out*.println("Test" + 10 \* 10 );  
}

**OUTPUT**

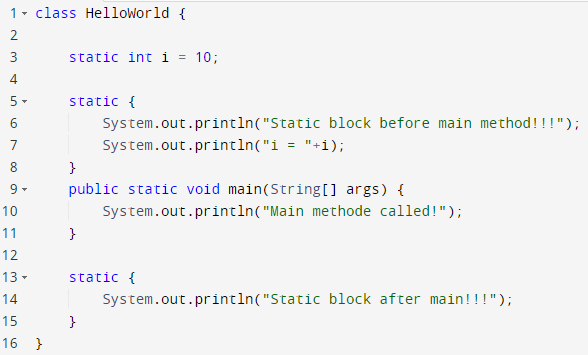
20Test  
Test1010  
Test100

**What is static block ?**

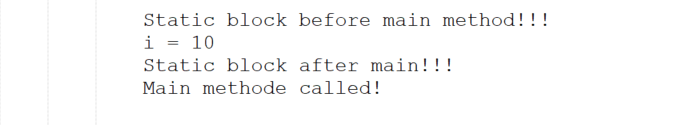
It is used to initialize the data variables and is **always called before the main method execution either it is declare before or after main method.**

In the below example, we have defined two static blocks with one static variable, one before the main method and another one after the methods. static blocks get execute in same sequence they define(from top to bottom).

In the output, we can clearly see the static blocks and variable executed before the main method execution. This static block, variable, method, and class initialize once the class gets loaded.



**OUTPUT:**



**What is abstract Method ?**

A Method contains abstract keyword in declaration known as Abstract method. We may have any number of abstract methods in a class.

// abstract class  
abstract class TestClass { // abstract method  
 abstract void addition();  
}

**Abstract Method Rules:**

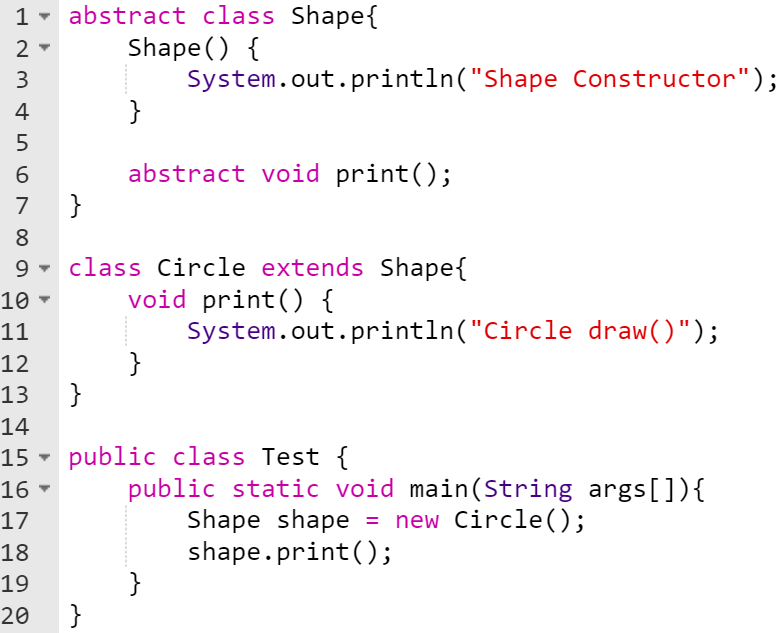
1. A class can have any number of abstract methods.
2. Class need to be declared as abstract if we have declared single abstract method in the class.
3. Abstract method doesn’t contain any method definition.

**Example:**

In the below example, there is a constructor defined with the abstract method print() in the abstract class Shape.

Sub class Circle extends the parent Shape class and override its print() method.

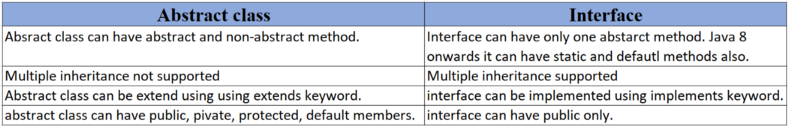
In the Test public class created a Shape object having a Circle class reference to execute the print() method.



**OUTPUT:**



**Abstract class and interface**

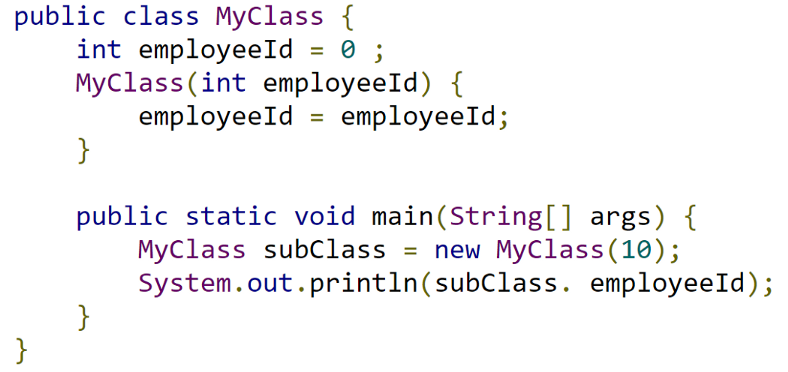


**Explain about this keyword ?**

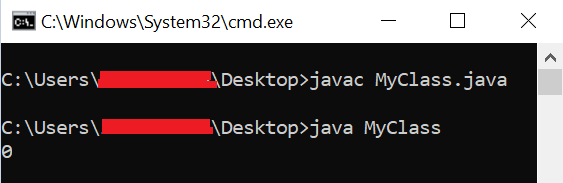
this keyword is used to access instance attributes and methods of current class and it doesn’t work inside static method and block. This keyword can be pass as a parameter in the method and constructor.

**Problem: *What will happen if we don’t use this key word inside constructor.***

In below code snippet, we are assigning constructor argument to the same variable and not to the instance variable. Due to absence of this keyword in MyClass(int employeeId) constructor, we are not assigning the employeeId to class level variable. Hence, printing the employeeId in main method is giving us the output as 0.

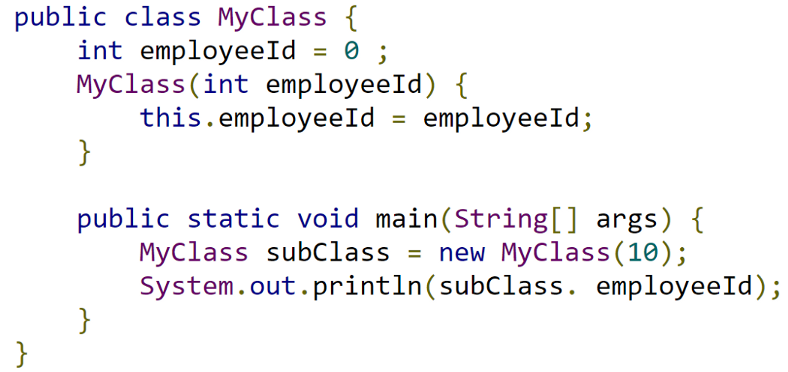


**OUTPUT:**

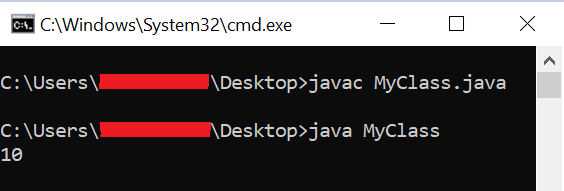


***Let’s make above program correct:***

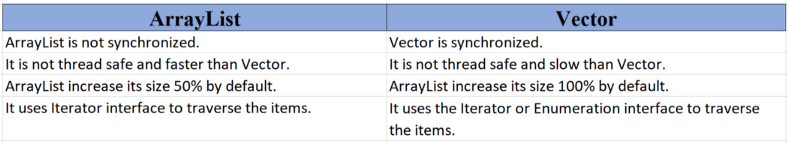
Below is the code snippet where we are assigning constructor argument value to instance variable using **this**keyword inside MyClass(int employeeId) constructor.



**OUTPUT:**



**ArrayList vs Vector**



**Why Multiple inheritance no supported in Java ?**

Let’s try to understand complete logic using below code.

Create two classes, One and Two, having the same method name, **printNumber()**. Extend both classes to Math class, which will give a compile-time error anyway. **math.printNumber()**callin inside main method will not be able to choose whether to call method **printNumber()** from class One or class Two.

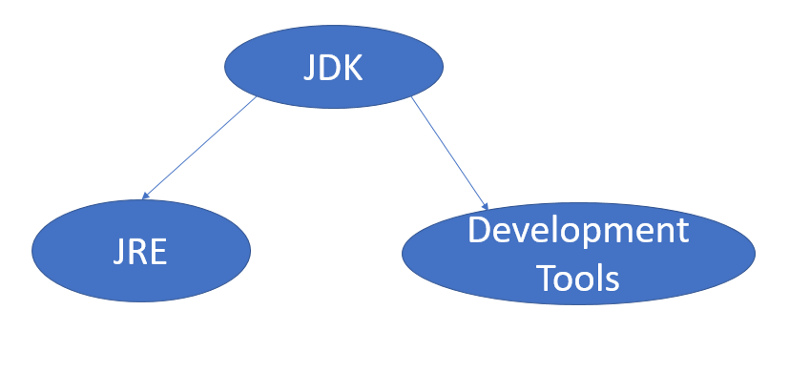
class One {  
 void printNumber() {  
 System.*out*.println("One");  
 }  
}  
  
class Two {  
 void printNumber() {  
 System.*out*.println("Two");  
 }  
}  
  
public class Math **extends One, Two** { // It will give compile Time Error  
  
 public static void main(String[] args) {  
 Math math = new Math();// Compiler will get confuse to execute printNumber() method either from Class One or Two  
 **math.printNumber();**   
 }  
}

**What is JDK ?**

Java Development Kit (JDK) is software development environment which is used to develop java applications and applets.

JDK help us to develop Java application and have all below resources to complete or create a java application:

java runtime environment (JVM), Interpreter/loader (Java), compiler (javac), archiver(jar) and java documentation (Java doc).



**What are constructors in Java ?**

Constructor is a block of code which use to initialize an object.

Constructor must have a **same name of the class with no return type**. Constructor gets automatically invokes when an object of that class is created.

There are three types of constructor in Java:

* No-arg Constructor
* Defalut Constructor
* Parameterized Constructor

I hope you found out this article interesting and informative. Please share it with your friends to spread the knowledge.