|  |  |
| --- | --- |
| Constructor | Method |
| 1. Constructor is used to create and initialize an Object. | Method is used to execute certain statements. |
| 1. A constructor is invoked implicitly by the System. | A method is to be invoked during program code. |
| 1. A constructor is invoked when new keyword is used to create an object. | A method is invoked when it is called. |
| 1. A constructor cannot have any return type. | A method can have a return type. |
| 1. A constructor initializes an object which is not existent. | A method can be invoked only on existing object. |
| 1. A constructor must have same name as that of the class. | A method name cannot be same as class name. |
| 1. A constructor cannot be inherited by a subclass. | A method is inherited by a subclass. |

* What is Use of This Keyword?

1. [this can be used to refer current class instance variable.](https://www.javatpoint.com/this1)

this keyword can be used to refer current class instance variable. If there is ambiguity between the instance variables and parameters, this keyword resolves the problem of ambiguity.

1. [this can be used to invoke current class method (implicitly)](https://www.javatpoint.com/this2)

You may invoke the method of the current class by using this keyword. If you don't use this keyword, compiler automatically adds this keyword while invoking the method. Let's see the example



1. [this can be used to invoke current class constructor.](https://www.javatpoint.com/this3)

this constructor call can be used to invoke the current class constructor. It is used to reuse the constructor. In other words, it is used for constructor chaining.

1. [this can be passed as an argument in the method call.](https://www.javatpoint.com/this4)

In event handling (or) in a situation where we have to provide reference of a class to another one. It is used to reuse one object in many methods.

1. [this can be passed as argument in the constructor call.](https://www.javatpoint.com/this5)

We can pass this keyword in the constructor also. It is useful if we have to use one object in multiple classes.

1. [this can be used to return the current class instance from the method.](https://www.javatpoint.com/this6)

We can return this keyword as statement from the method. In such case, return type of the method must be the class type (non-primitive).

|  |  |
| --- | --- |
| Method Overloading | Method Overriding |
| 1.Method overloading is a compile-time polymorphism. | Method overriding is a run-time polymorphism. |
| 2.It helps to increase the readability of the program. | It is used to grant the specific implementation of the method which is already provided by its parent class or superclass. |
| 3.It occurs within the class. | It is performed in two classes with inheritance relationships. |
| 4.Method overloading may or may not require inheritance  . | Method overriding always needs inheritance. |
| 5.In method overloading, methods must have the same name and different signatures. | In method overriding, methods must have the same name and same signature. |
| 6.In method overloading, the return type can or cannot be the same, but we just have to change the parameter. | In method overriding, the return type must be the same or co-variant. |
| 7.Static binding is being used for overloaded methods. | Dynamic binding is being used for overriding methods. |
| 8.Poor Performance due to compile time polymorphism. | It gives better performance. The reason behind this is that the binding of overridden methods is being done at runtime. |
| 9.Private and final methods can be overloaded. | Private and final methods can’t be overridden. |
| 10.Argument list should be different while doing method overloading. | Argument list should be same in method overriding. |

**Super Keyword:**

The **super** keyword in Java is a reference variable which is used to refer immediate parent class object.

Whenever you create the instance of subclass, an instance of parent class is created implicitly which is referred by super reference variable.

**Usage of Java super Keyword**

1. super can be used to refer immediate parent class instance variable.
2. super can be used to invoke immediate parent class method.
3. super () can be used to invoke immediate parent class constructor.

* If both parent & child classes have the same method, then the child class would override the method available in its parent class. By using the super Keyword, we can take advantage of both classes (child and parent) to achieve this.
* We create an object of child class as it can inherit the parent class methods.
* In the child class method, we call the method available in its parent class by using super ().

|  |
| --- |
| **Method Overriding** |