|  | **Static method** | **Non-static method** |
| --- | --- | --- |
|  | A **static method** is a method that belongs to a class, but it does not belong to an instance of that class and this method can be called without the instance or object of that class. | Every method in java defaults to a non-static method without a **static** keyword preceding it. **non-static** methods can access any **static** method and **static** variable also, without using the object of the class. |
|  | In the **static** method, the method can only access only static data members and static methods of another class or same class but cannot access non-static methods and variables. | In the **non-static** method, the method can access static data members and static methods as well as non-static members and methods of another class or same class. |
|  | The static method uses compile-time or early binding. | The non-static method uses runtime or dynamic binding. |
|  | The static method cannot be overridden because of early binding. | The non-static method can be overridden because of runtime binding. |
|  | In the **static** method, less memory is used for execution because memory allocation happens only once because the static keyword fixed a particular memory for that method in ram. | In the **non-static** method, much memory is used for execution because here memory allocation happens when the method is invoked and the memory is allocated every time when the method is called. |

class A

{

     int nonStaticVariable;

     static int staticVariable;

     static void staticMethod()

     {

          System.out.println(staticVariable);

     //   System.out.println(nonStaticVariable);

     }

     void nonStaticMethod()

     {

          System.out.println(staticVariable);

          System.out.println(nonStaticVariable);

     }

}

class MainClass

{

     public static void main(String[] args)

     {

          A.staticVariable = 10;

     //   A.nonStaticVariable = 10;

          A.staticMethod();

    //    A.nonStaticMethod();

          A a1 = new A();

          A a2 = new A();

          System.out.println(a1.nonStaticVariable);

          System.out.println(a1.staticVariable);

          a1.nonStaticMethod();

          a1.staticMethod();

          System.out.println(a2.staticVariable);

          a1.staticVariable = 20;

          System.out.println(a2.staticVariable);

     }

}