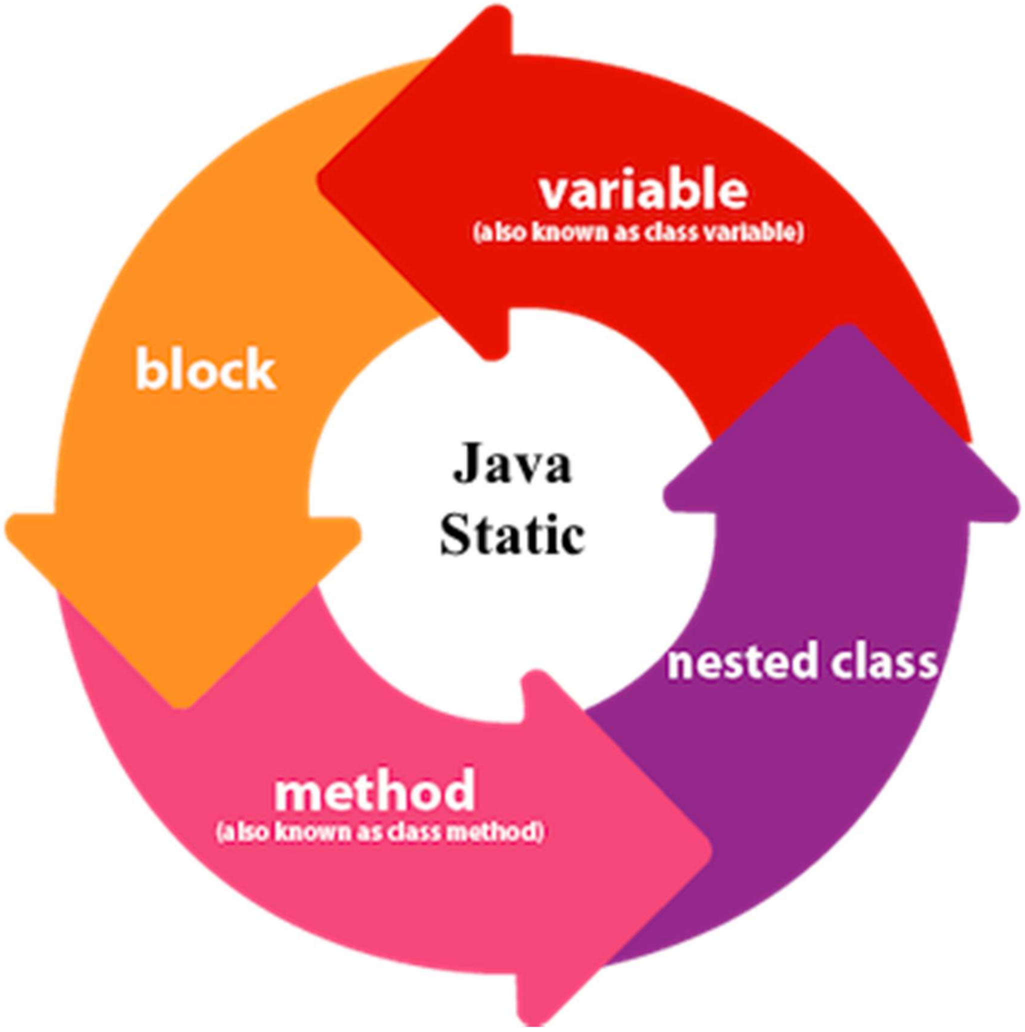
Static Class and Static Method

The static keyword in Java is used for memory management mainly. We can apply static keyword with variables, methods, blocks and nested classes. The static keyword belongs to the class than an instance of the class.

The static can be:

- Variable (also known as a class variable)
- Method (also known as a class method)
- Block
- Nested class


variable
(also known as class variable)
block
Java
Static
nested class
method
(also known as class method)

Static variable

When you declare a variable as static, then a single copy of the variable is created and divided among all objects at the class level. Static variables are, essentially, global variables. Basically, all the instances of the class share the same static variable. Static variables can be created at class-level only.

Static method

When a method is declared with the static keyword, it is known as a **static method**. The most common example of a static method is the **main( ) method**. Methods declared as static can have the following restrictions:

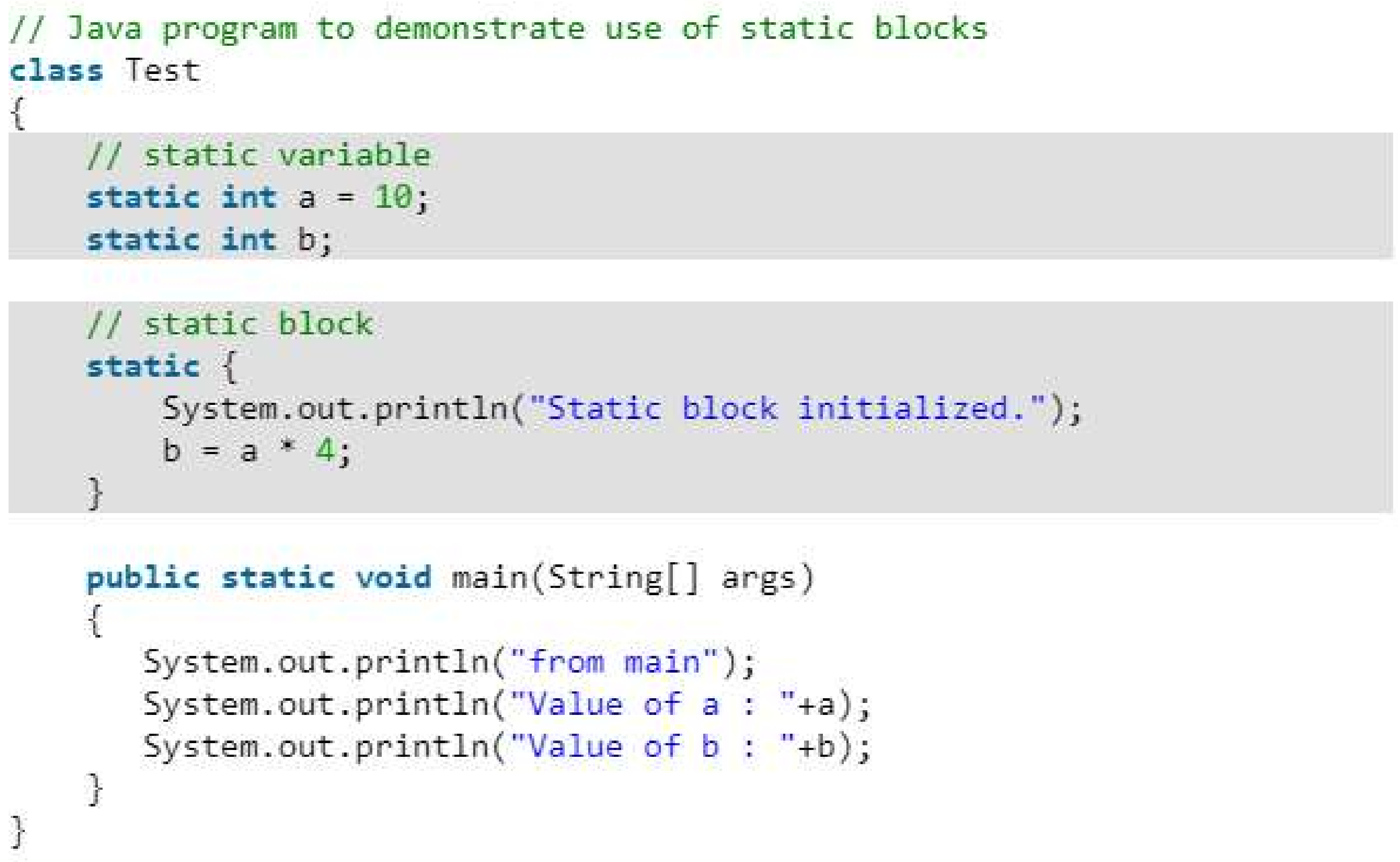
- They can directly call other static methods only.
- They can access static data directly.

There are two main restrictions for the static method. They are:

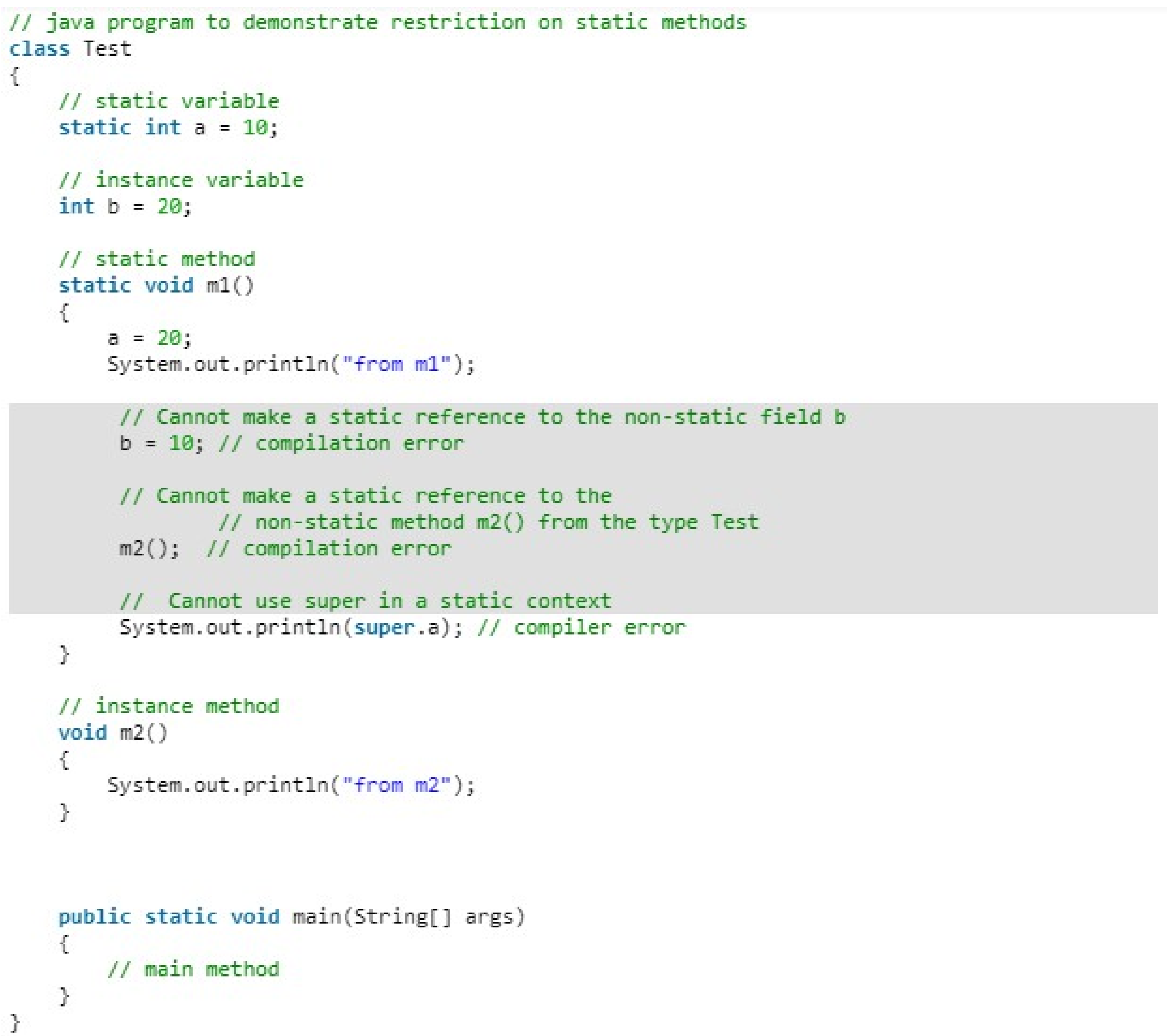
- **The static method cannot use non static data member or call non-static method directly.**
- **this and super cannot be used in static context.**

If you need to do computation in order to initialize your static variables, you can declare a static block that gets executed exactly once, when the class is first loaded.

- **Is used to initialize the static data member.**
- **It is executed before the main method at the time of classloading.**

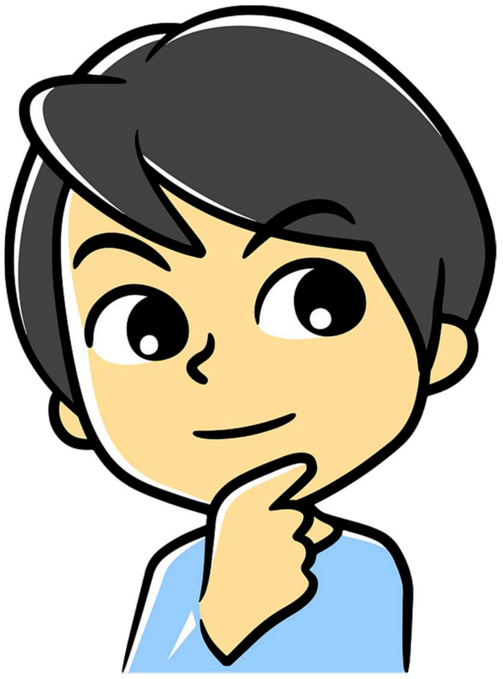


```java
// Java program to demonstrate use of static blocks
class Test
{
    // static variable
    static int a = 10;
    static int b;

    // static block
    static {
        System.out.println("Static block initialized.");
        b = a * 4;
    }

    public static void main(String[] args)
    {
       System.out.println("from main");
       System.out.println("Value of a : "+a);
       System.out.println("Value of b : "+b);
    }
}
```



```
// java program to demonstrate restriction on static methods
class Test
{
    // static variable
    static int a = 10;

    // instance variable
    int b = 20;

    // static method
    static void m1()
    {
        a = 20;
        System.out.println("from m1");

         // Cannot make a static reference to the non-static field b
         b = 10; // compilation error

         // Cannot make a static reference to the
                 // non-static method m2() from the type Test
         m2();  // compilation error

         //  Cannot use super in a static context
         System.out.println(super.a); // compiler error
    }

    // instance method
    void m2()
    {
        System.out.println("from m2");
    }



    public static void main(String[] args)
    {
        // main method
    }
}
```

Why is the Java main method static?



Why is the Java main method static?

It is because the object is not required to call a static method. If it were a non-static method, JVM creates an object first then call main() method that will lead the problem of extra memory allocation.

Can we execute a program without main() method?

Can we execute a program without main() method?

No, one of the ways was the static block, but it was possible till JDK 1.6. Since JDK 1.7, it is not possible to execute a Java class without the main method.

NEXT SESSION: String