MOD -02 -> OOPs

LUMINAR
TECHNOLAB

static Keyword

LUMINAR
TECHNOLAB

static keyword

static Keyword

Static is a keyword in java used to represent the class members.

It can be used with :

1. variable
2. Method
3. initializer block
4. nested class.

Types of class members

1. Static data members.
2. Static method.
3. Static initializer block.
4. Static nested class

Memory division in a java program execution

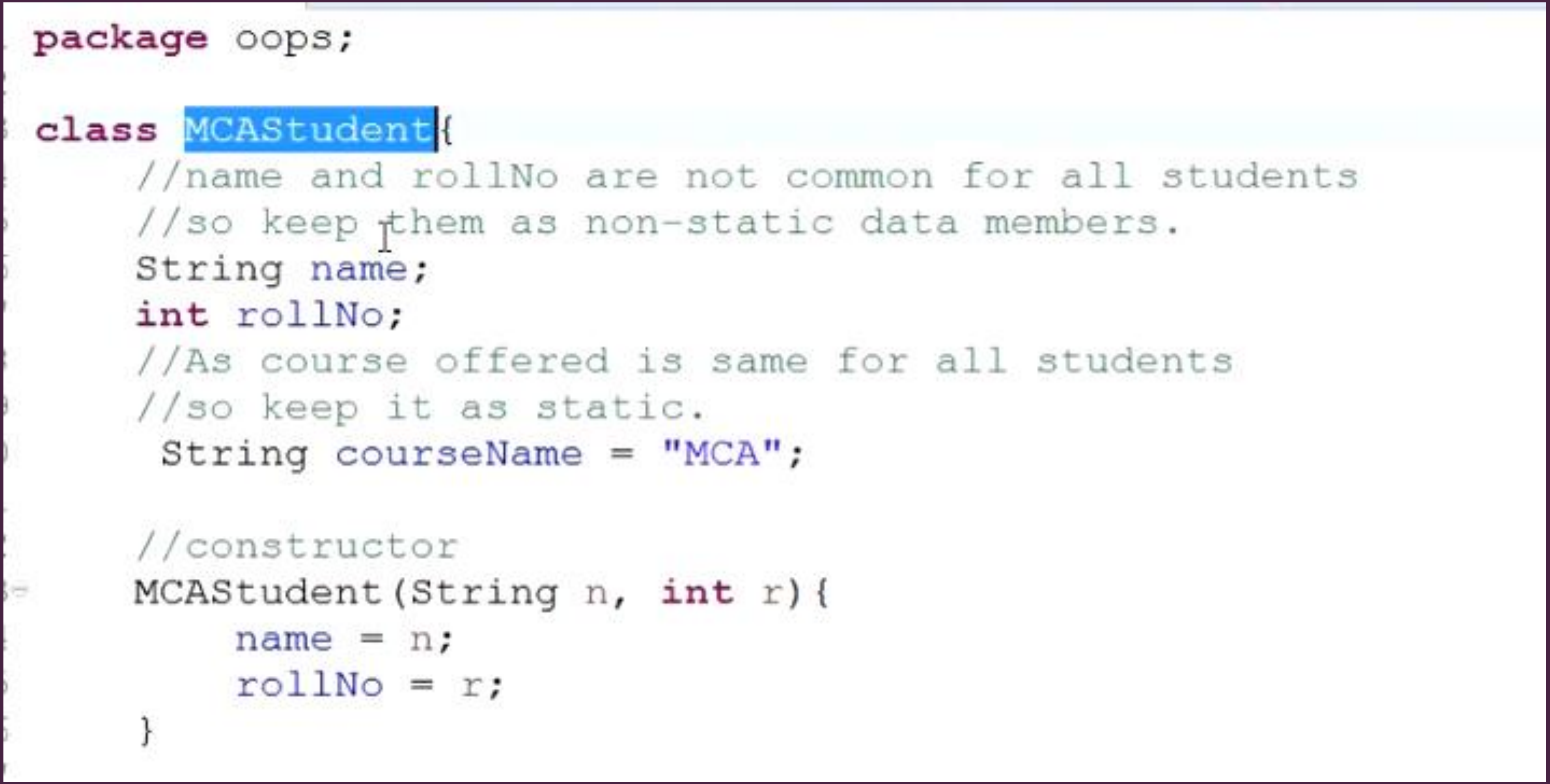
In a java program execution memory is divided into three parts:

- Stack: Stack is used to local variables of the methods.
- Heap: Heap is used to store objects.
- Class Area: Class area is used to store static data members.

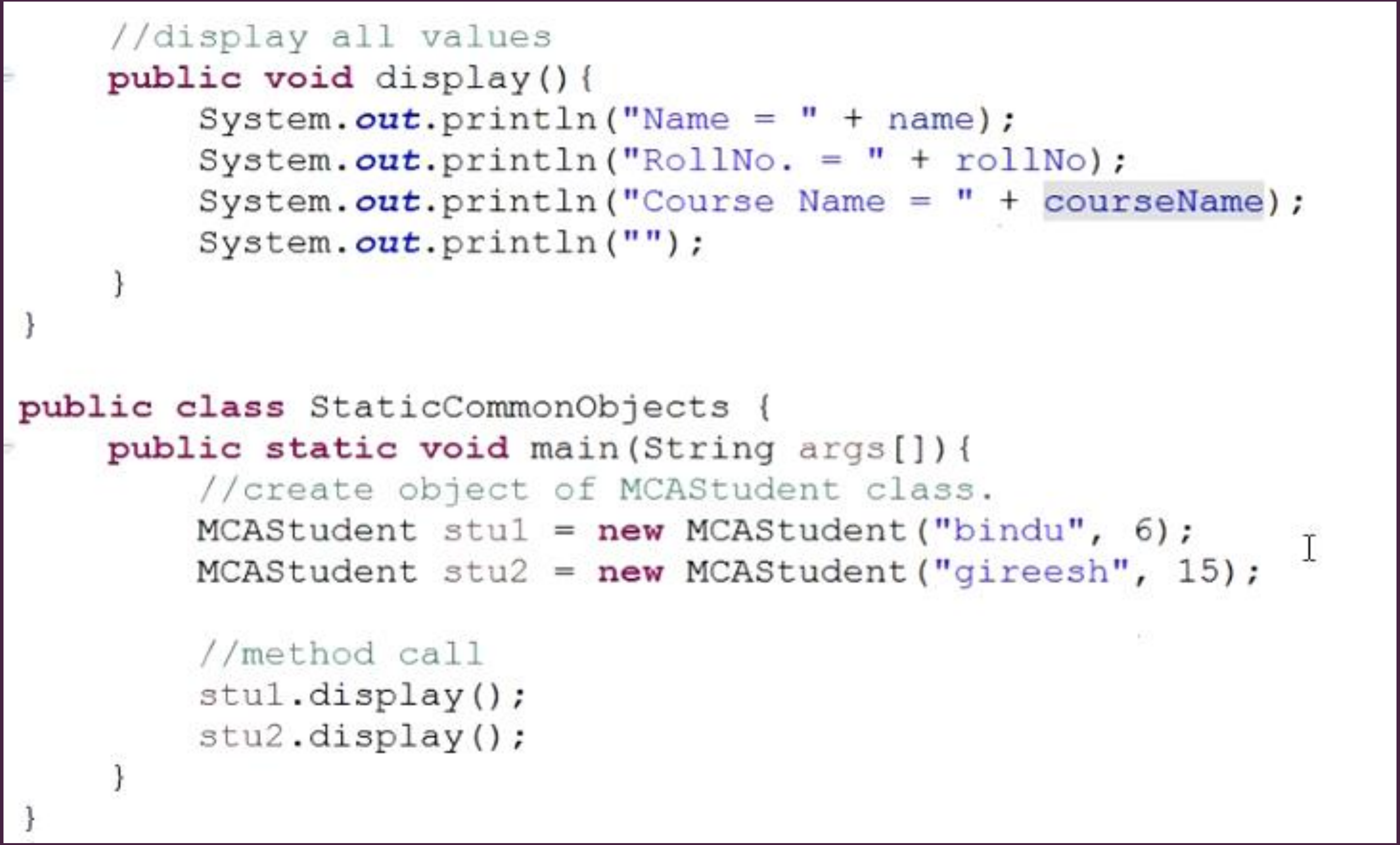
static variable

Static data members are used to represent those properties which are common to every object.

- Data members declared with static keyword are known as static data members.
- These are mainly used to represent those properties which are common to every object.
- At the time of class loading a single copy is created for static data members, which is shared by all objects.

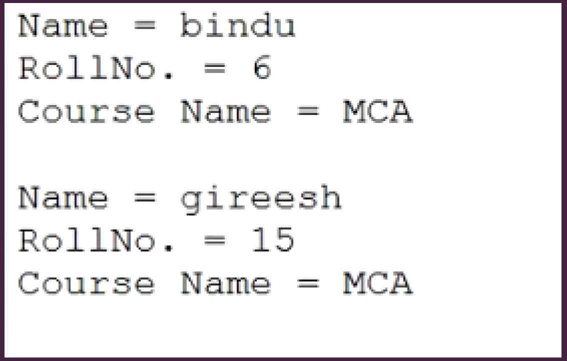


```
package oops;

class MCAStudent{
    //name and rollNo are not common for all students
    //so keep them as non-static data members.
    String name;
    int rollNo;
    //As course offered is same for all students
    //so keep it as static.
     String courseName = "MCA";

    //constructor
    MCAStudent(String n, int r){
        name = n;
        rollNo = r;
    }
```

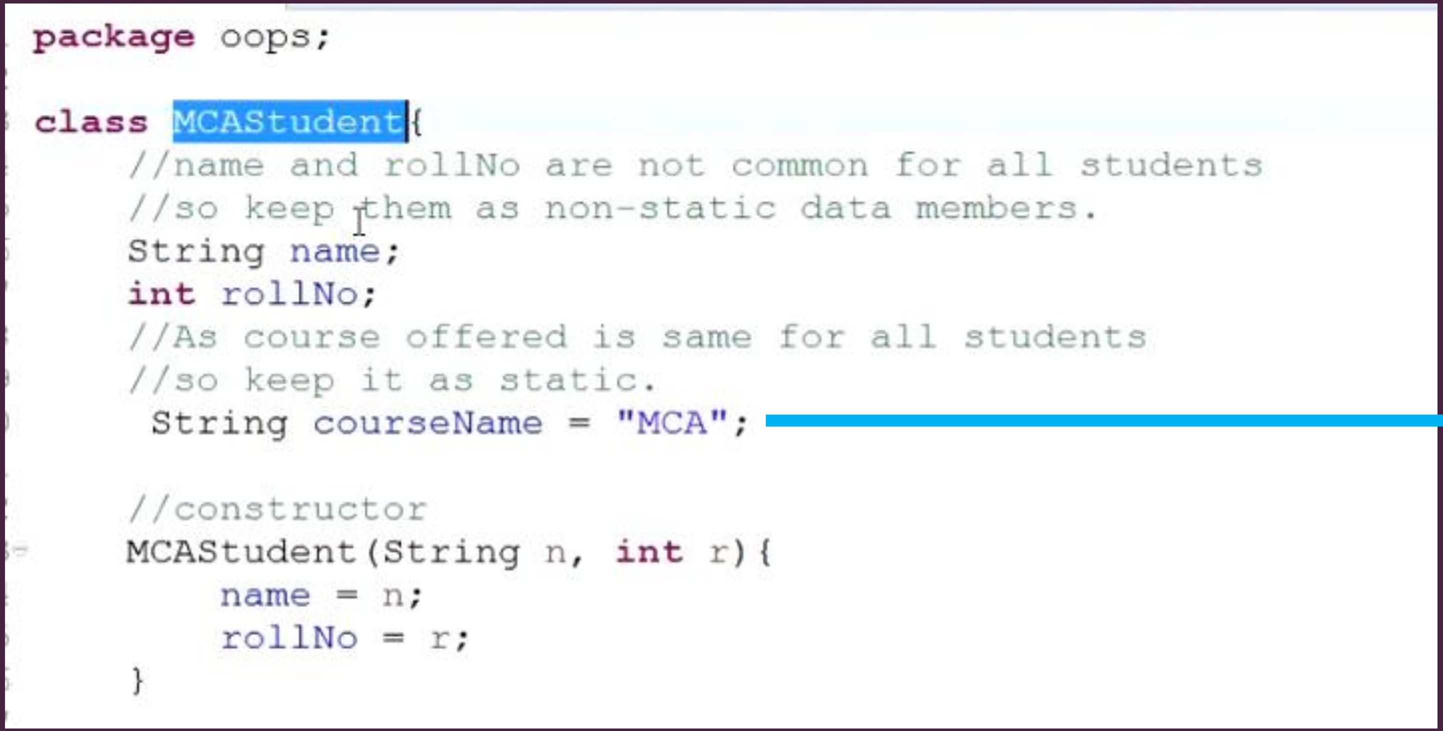


```java
    //display all values
    public void display(){
        System.out.println("Name = " + name);
        System.out.println("RollNo. = " + rollNo);
        System.out.println("Course Name = " + courseName);
        System.out.println("");
    }
}

public class StaticCommonObjects {
    public static void main(String args[]){
        //create object of MCAStudent class.
        MCAStudent stu1 = new MCAStudent("bindu", 6);
        MCAStudent stu2 = new MCAStudent("gireesh", 15);

        //method call
        stu1.display();
        stu2.display();
    }
}
```

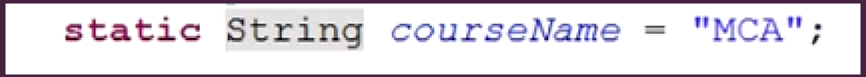
Output - console



```
Name = bindu
RollNo. = 6
Course Name = MCA

Name = gireesh
RollNo. = 15
Course Name = MCA
```



```java
package oops;

class MCAStudent{
    //name and rollNo are not common for all students
    //so keep them as non-static data members.
    String name;
    int rollNo;
    //As course offered is same for all students
    //so keep it as static.
     String courseName = "MCA";

    //constructor
    MCAStudent(String n, int r){
        name = n;
        rollNo = r;
    }
```

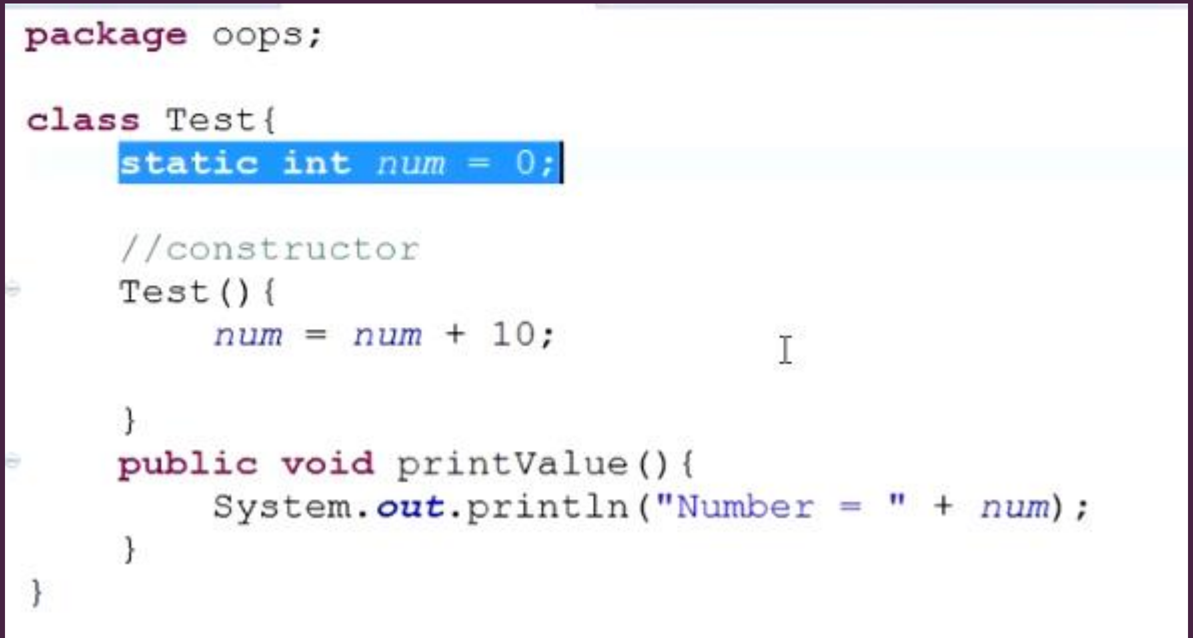


```java
static String courseName = "MCA";
```

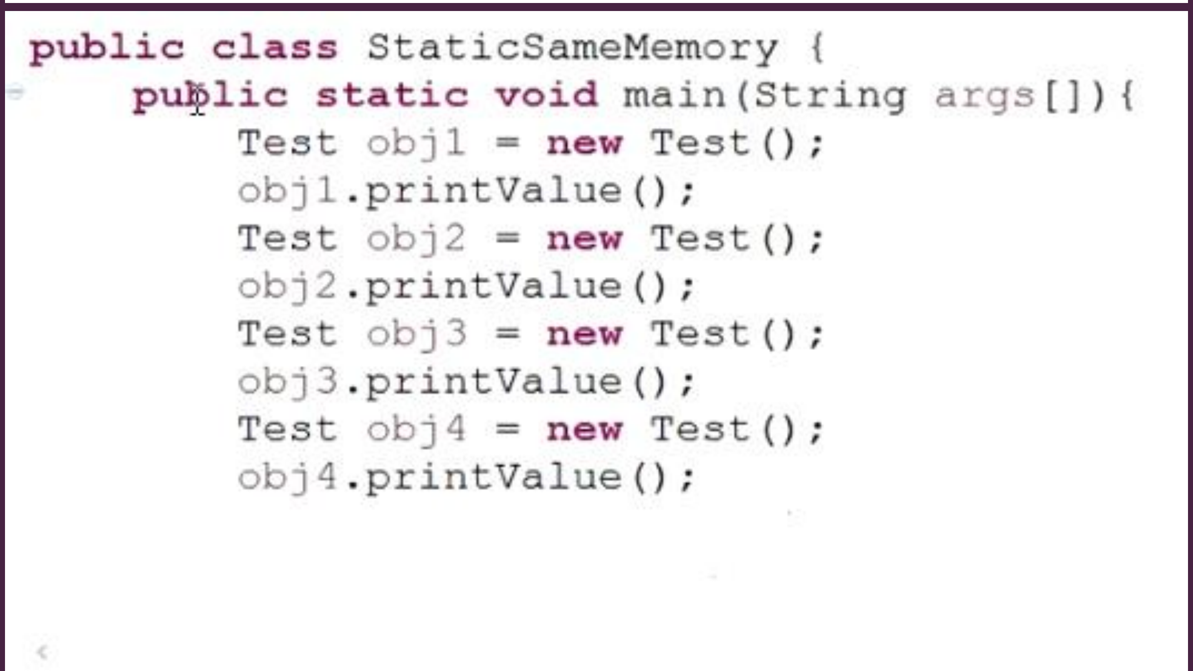
Static data members use the same memory locations for all objects.

NOTE

Constructor is not used to initialize the static data members because constructor initializes many times but static data members only once. So instead of constructor static initialize block is used to initialize static data members.

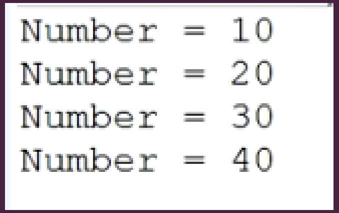


```
package oops;

class Test{
    static int num = 0;

    //constructor
    Test(){
        num = num + 10;

    }
    public void printValue(){
        System.out.println("Number = " + num);
    }
}
```



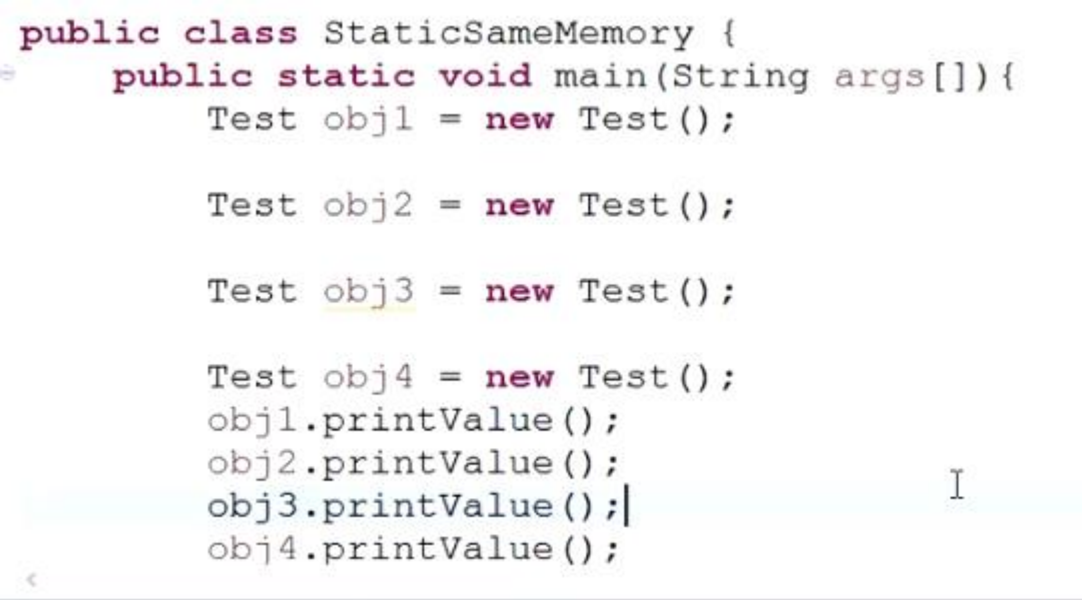
```
public class StaticSameMemory {
    public static void main(String args[]){
        Test obj1 = new Test();
        obj1.printValue();
        Test obj2 = new Test();
        obj2.printValue();
        Test obj3 = new Test();
        obj3.printValue();
        Test obj4 = new Test();
        obj4.printValue();
```

Output - console



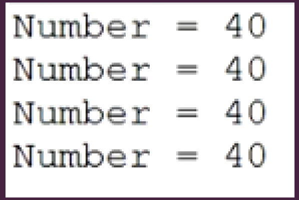
```
Number = 10
Number = 20
Number = 30
Number = 40
```

Rearrange the code



```
public class StaticSameMemory {
    public static void main(String args[]){
        Test obj1 = new Test();

        Test obj2 = new Test();

        Test obj3 = new Test();

        Test obj4 = new Test();
        obj1.printValue();
        obj2.printValue();
        obj3.printValue();
        obj4.printValue();
```

Output - console



```
Number = 40
Number = 40
Number = 40
Number = 40
```

static methods

Static methods represent the behavior of whole class.

An instance of a class is not required to execute static methods. They can be called using class name.

Syntax:

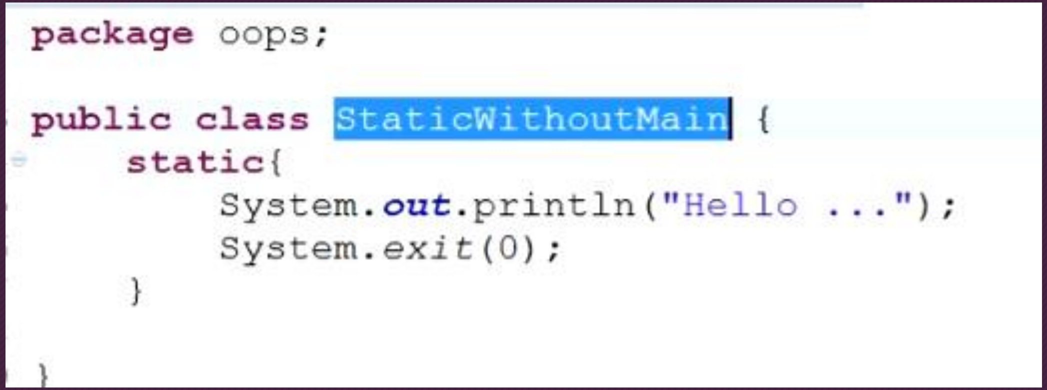
```
ClassName.methodName
```

```java
package staticpack;


class Test{
    public static void display() {
        System.out.println("inside display method");
    }
}


public class StaticMethod {


    public static void main(String[] args) {
        Test.display();
    }


}
```

static block

We can execute a program without main method.

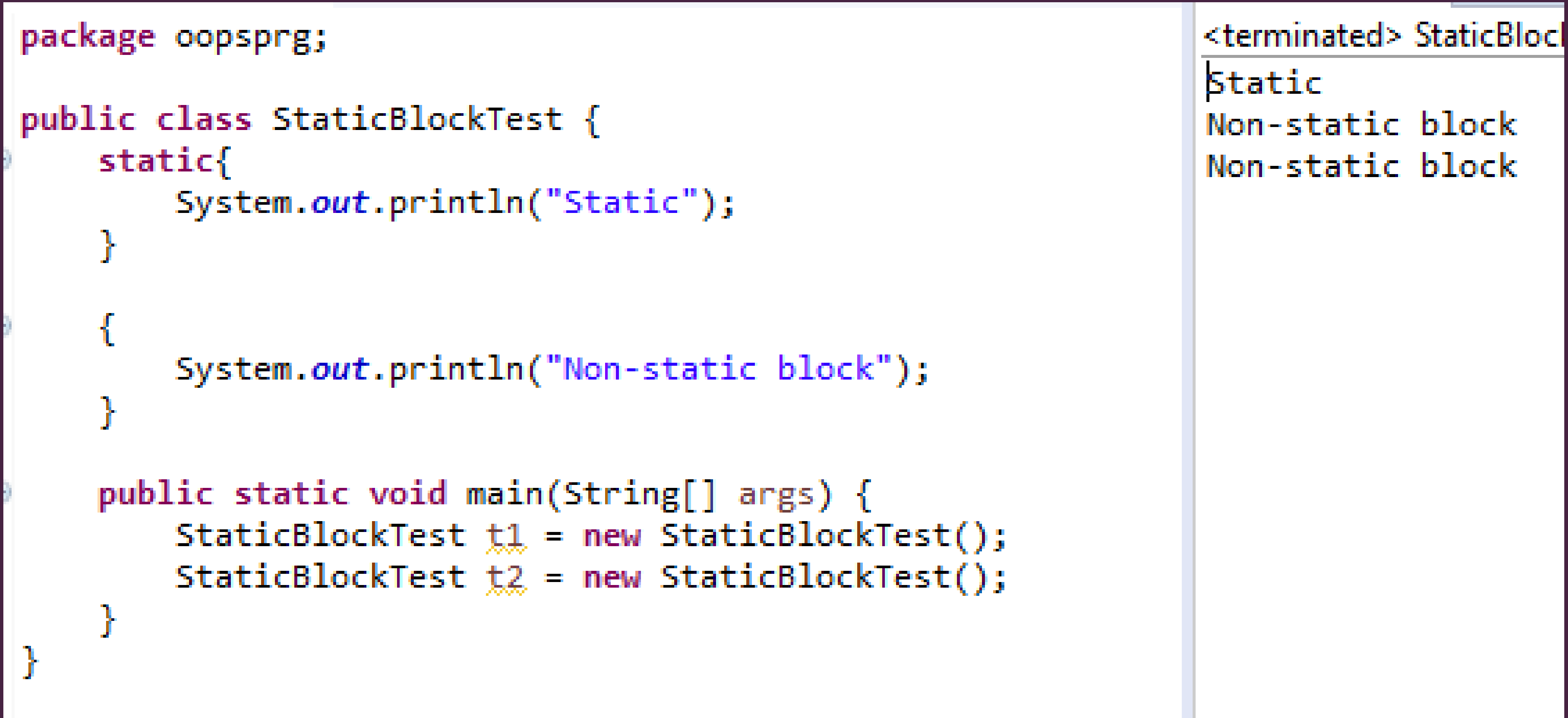
Calling of static block -> Nothing to do , as it is called automatically as class is loaded in memory



```
package oops;

public class StaticWithoutMain {
    static{
        System.out.println("Hello ...");
        System.exit(0);
    }

}
```

Note: This approach doesn't work in Java 7 and later because the Java Virtual Machine (JVM) strictly requires the main() method.

1. Non –Static block Gets called every time an instance of the class is constructed.
2. The static block only gets called once, when the class itself is initialized, no matter how many objects of that type you create.



```java
package oopsprg;

public class StaticBlockTest {
    static{
        System.out.println("Static");
    }

    {
        System.out.println("Non-static block");
    }

    public static void main(String[] args) {
        StaticBlockTest t1 = new StaticBlockTest();
        StaticBlockTest t2 = new StaticBlockTest();
    }
}
```

```
<terminated> StaticBloc
Static
Non-static block
Non-static block
```

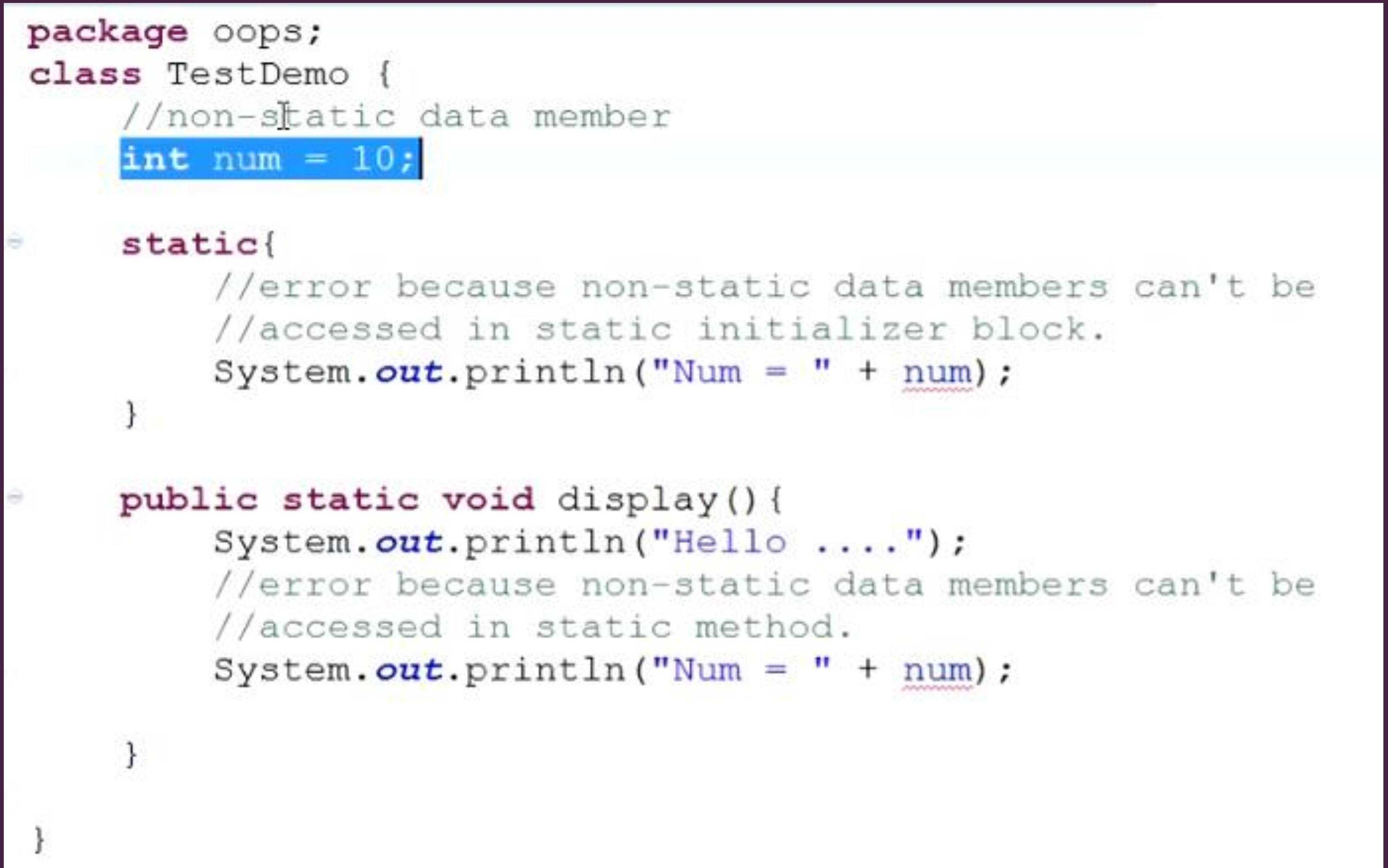
Limitations of static methods and static initialize blocks.

1. Non-static data members can't be accessed in static methods and static initialize blocks

2. Non-static methods can't be invoked in static methods and static initialize blocks.

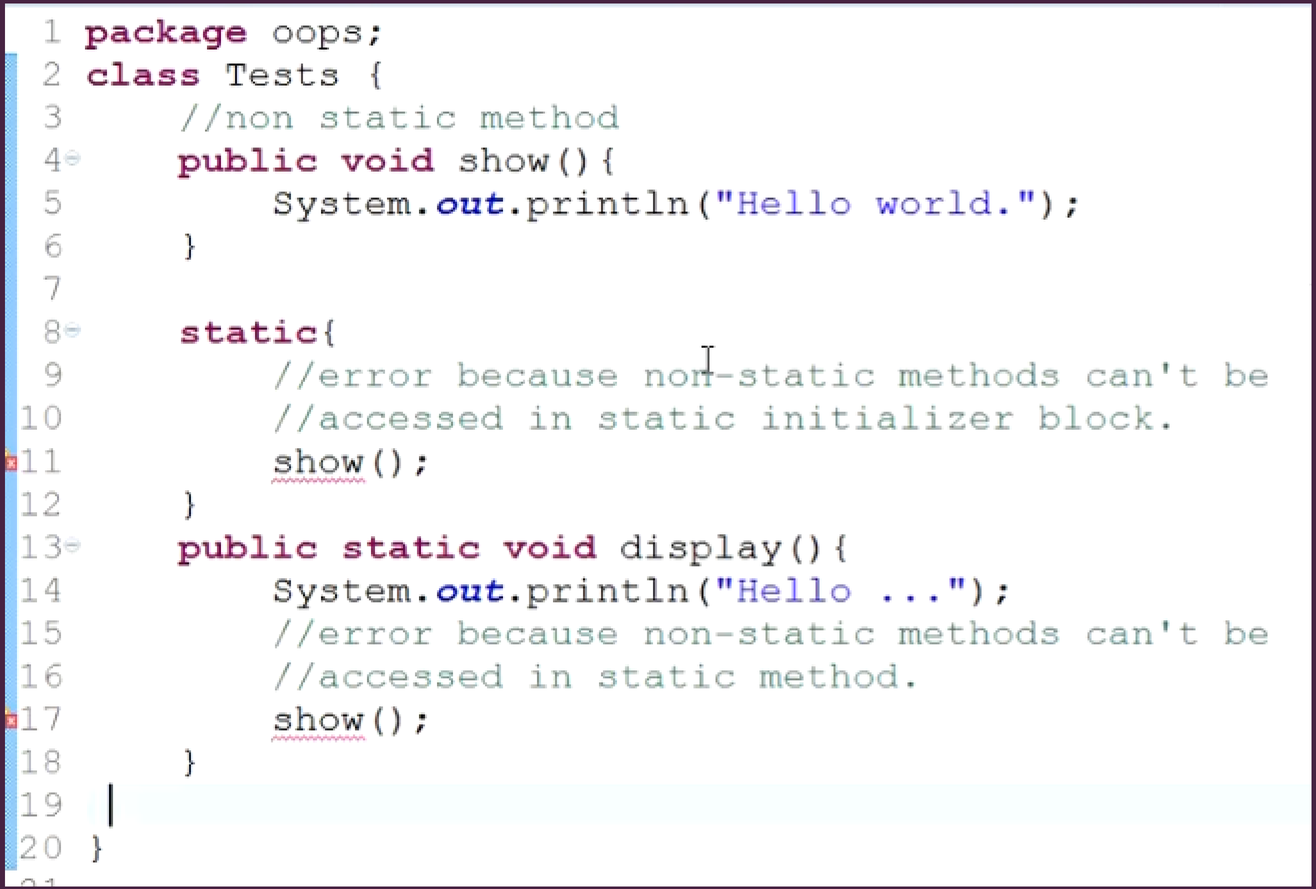
3. *This* or *super* keyword can't be refers in static methods and static initialize blocks.

Non-static data members can’t be accessed in static methods and static initialize blocks



```java
package oops;
class TestDemo {
    //non-static data member
    int num = 10;

    static{
        //error because non-static data members can't be
        //accessed in static initializer block.
        System.out.println("Num = " + num);
    }

    public static void display(){
        System.out.println("Hello ....");
        //error because non-static data members can't be
        //accessed in static method.
        System.out.println("Num = " + num);

    }

}
```

Non-static methods can’t be invoked in static methods and static initialize blocks.



```java
package oops;
class Tests {
    //non static method
    public void show(){
        System.out.println("Hello world.");
    }

    static{
        //error because non-static methods can't be
        //accessed in static initializer block.
        show();
    }
    public static void display(){
        System.out.println("Hello ...");
        //error because non-static methods can't be
        //accessed in static method.
        show();
    }

}
```

Thank you ☺ Happy coding ☺