INDIAN INSTITUTE OF TECHNOLOGY ROORKEE

Fundamentals of Object Oriented Programming

CSN- 103

Dr. R. Balasubramanian
Associate Professor
Department of Computer Science and Engineering
Indian Institute of Technology Roorkee
Roorkee 247 667
balarfcs@iitr.ac.in
https://sites.google.com/site/balaiitr/

Classes in JAVA

- In object-oriented programming technique, we design a program using objects and classes.
  - Object is the physical as well as logical entity whereas class is the logical entity only.
- Objects

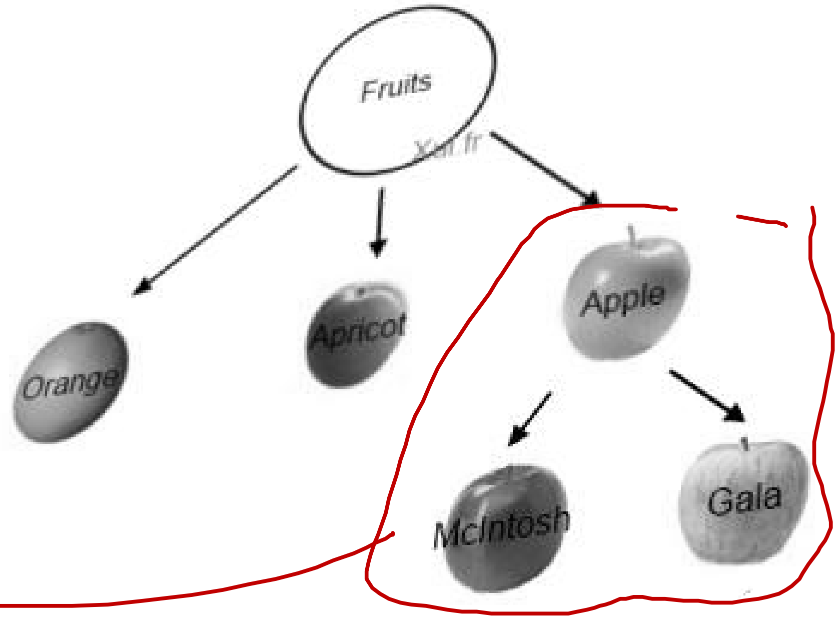
Objects in Java

- An entity that has state and behavior is known as an object e.g. chair, bike, marker, pen, table, car etc. It can be physical or logical (tangible and intangible).
  - The example of intangible object is banking system.
- An object has three characteristics:

- **state:** represents data (value) of an object.
- **behavior:** represents the behavior (functionality) of an object such as deposit, withdraw etc.
- **identity:** Object identity is typically implemented via a unique ID. The value of the ID is not visible to the external user. But it is used internally by the JVM to identify each object uniquely.

- For Example: Pen is an object. Its name is Parker, color is Golden etc. known as its state. It is used to write, so writing is its behavior.
- **Object is an instance of a class.** Class is a template or blueprint from which objects are created. So object is the instance(result) of a class.

Inheritance



Class in JAVA

- A class is a group of objects that has common properties.
- It is a template or blueprint from which objects are created.

- A class in java can contain:
  - **data member**
  - **method**
  - **constructor**
  - **block**
  - **class and interface**

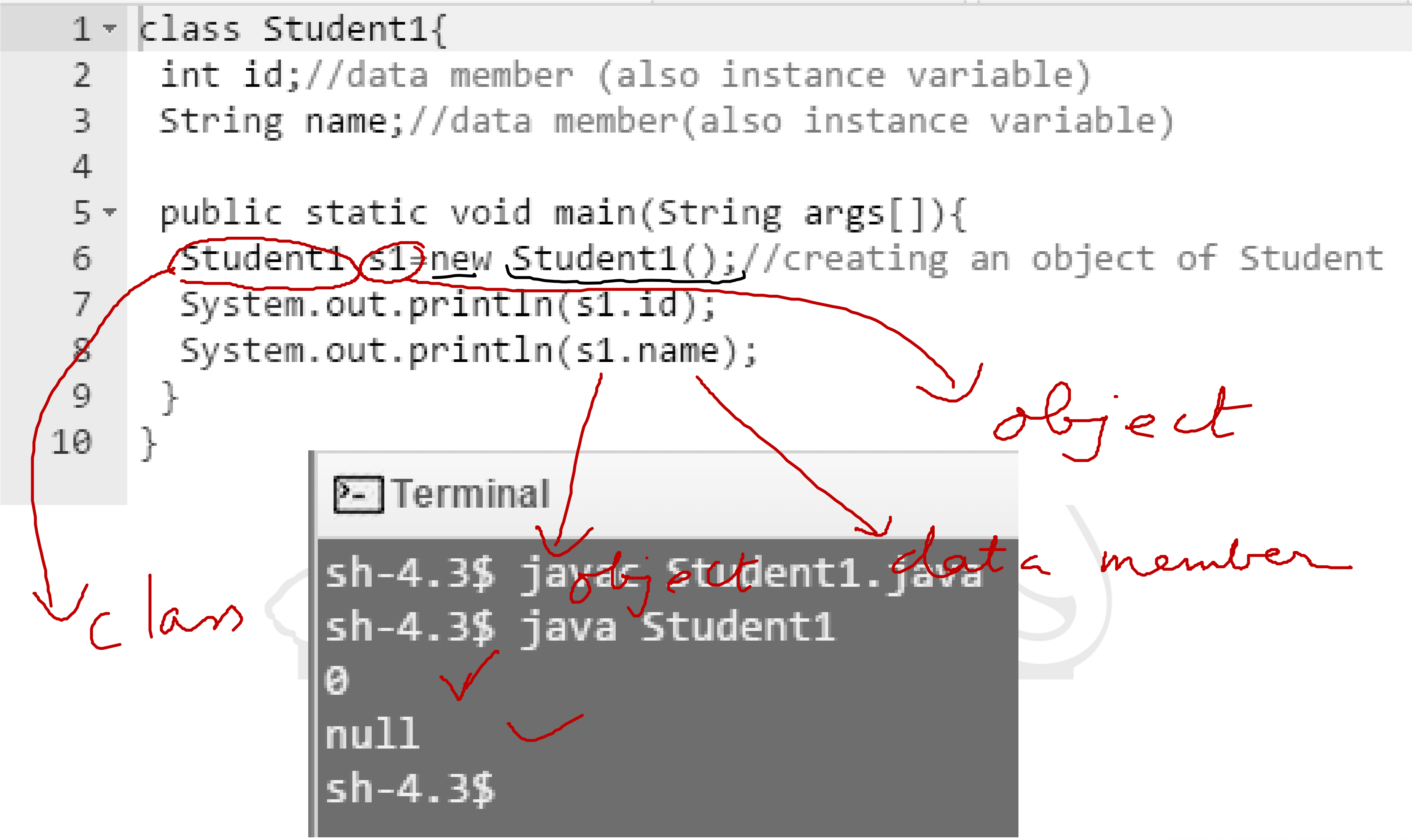
Syntax to declare a class:

```
class <class_name>{
    data member; //field
    method;
}
```

```
class Time {
     int hour;
     int minutes;
     int seconds;
  ~~}~~ int Add_time ( );
    }○ X
```

```
// C++ Programmers may note that there is no
//semicolon after the closed braces!
```

int [] a1;
a1 = new int[10];



```java
class Student1{
 int id;//data member (also instance variable)
 String name;//data member(also instance variable)

 public static void main(String args[]){
  Student1 s1=new Student1();//creating an object of Student
  System.out.println(s1.id);
  System.out.println(s1.name);
 }
}
```

```
sh-4.3$ javac Student1.java
sh-4.3$ java Student1
0
null
sh-4.3$
```

object

data member

object

class

Methods Declaration

- A Java method is a collection of statements that are grouped together to perform an operation.
  - When you call the System.out.**println()** method, for example, the system actually executes several statements in order to display a message on the console.

- Now you will learn how to create your own methods with or without return values, invoke a method with or without parameters, and apply method abstraction in the program design.

Creating Method:

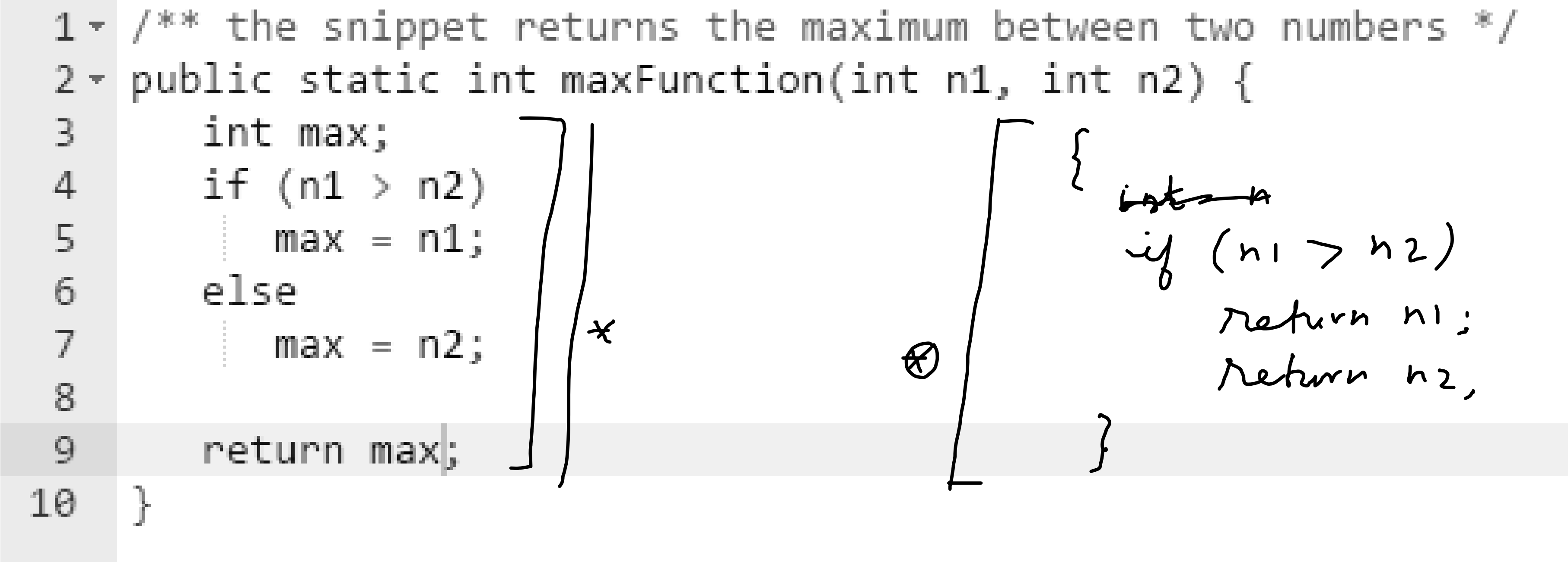
```
public static int Name_of_Method(int a, int int b)
{ // body }
```

- Here,
  - **public static** : modifier.
  - **int**: return type
  - **Name_of_Method**: name of the method
  - **a, b**: formal parameters
  - **int a, int b**: list of parameters

Syntax

-
  ```
  modifier returnType nameOfMethod (Parameter List)
  { // method body }
  ```
- The syntax includes:
- **modifier:** It defines the access type of the method and it is optional to use.
- **returnType:** Method may return a value.
- **nameOfMethod:** This is the method name. The method signature consists of the method name and the parameter list.
- **Parameter List:** The list of parameters, it is the type, order, and number of parameters of a method. These are optional, method may contain zero parameters.
- **method body:** The method body defines what the method does with statements.

Example



```
/** the snippet returns the maximum between two numbers */
public static int maxFunction(int n1, int n2) {
    int max;
    if (n1 > n2)
        max = n1;
    else
        max = n2;

    return max;
}
```

*

⊗

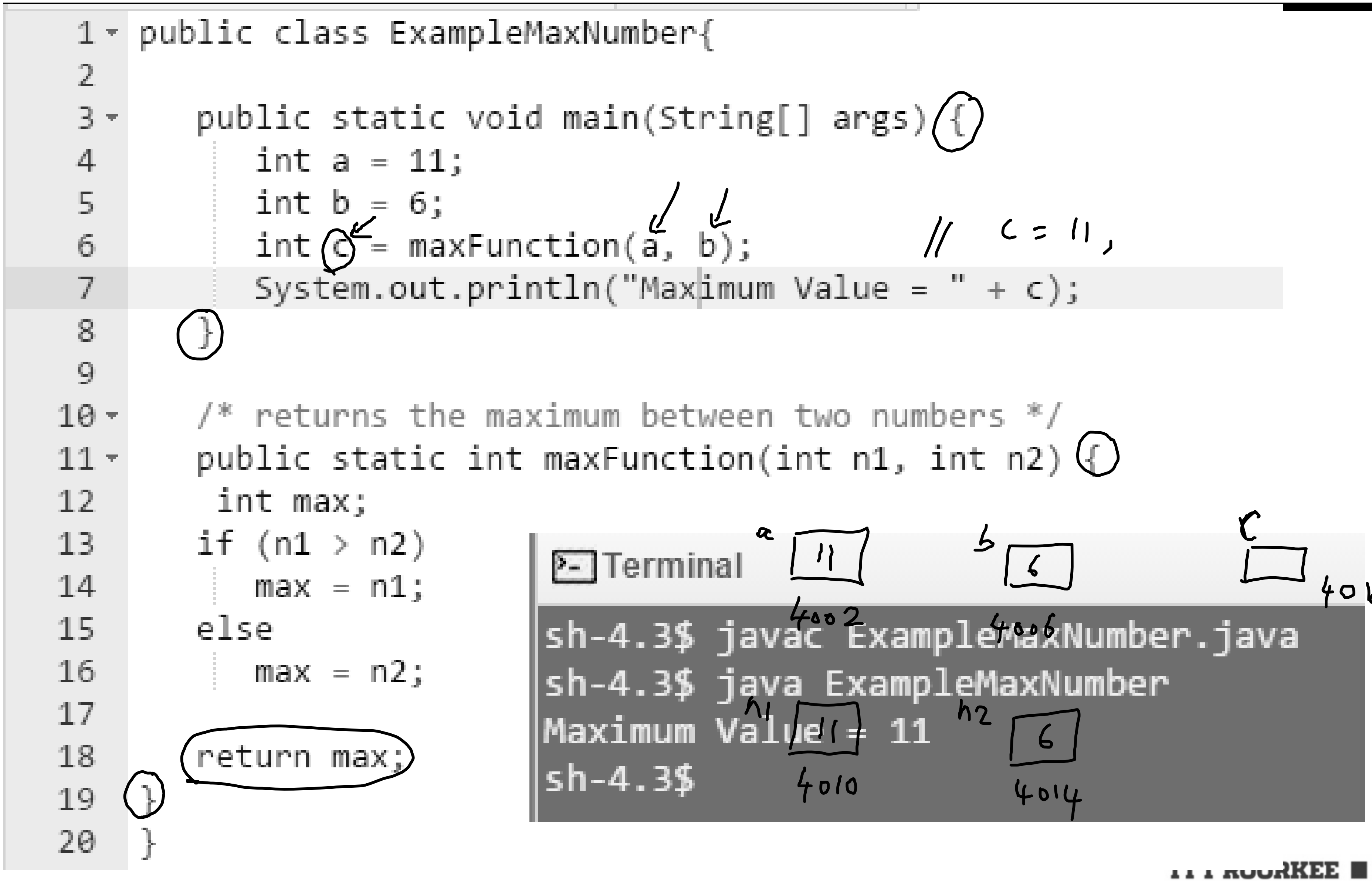
```
{
  ~~int n~~
  if (n1 > n2)
       return n1;
       Return n2,
}
```

Method Calling

- For using a method, it should be called.
- There are two ways in which a method is called,

  i.e. method returns a value or returning nothing (no return value).

- The called method then returns control to the caller in two conditions, when:
  - return statement is executed.
  - reaches the method ending closing brace.



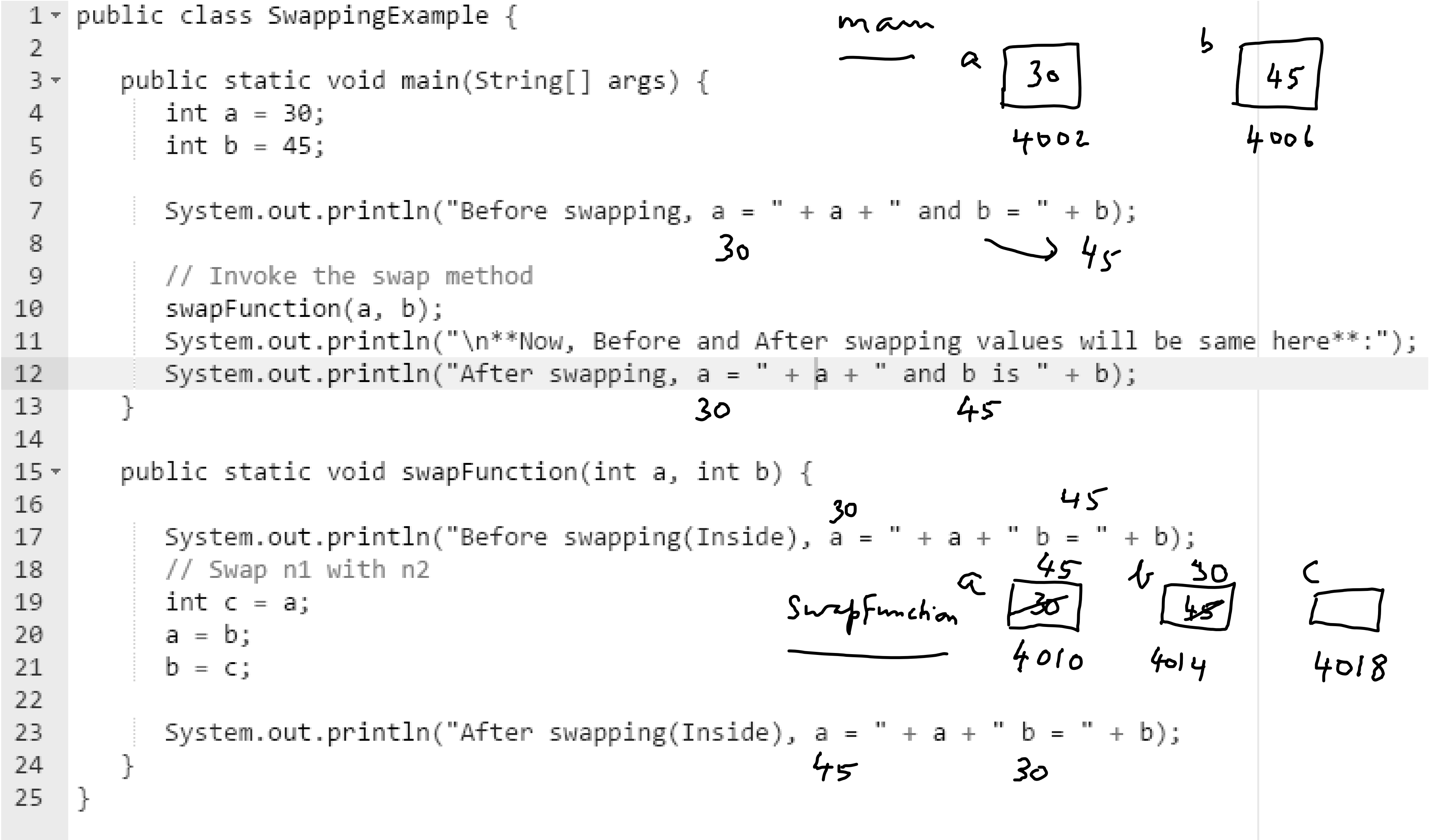
```java
public class ExampleMaxNumber{

    public static void main(String[] args){
        int a = 11;
        int b = 6;
        int c = maxFunction(a, b);
        System.out.println("Maximum Value = " + c);
    }

    /* returns the maximum between two numbers */
    public static int maxFunction(int n1, int n2) {
     int max;
    if (n1 > n2)
        max = n1;
    else
        max = n2;

    return max;
}
}
```

// c = 11,

Terminal

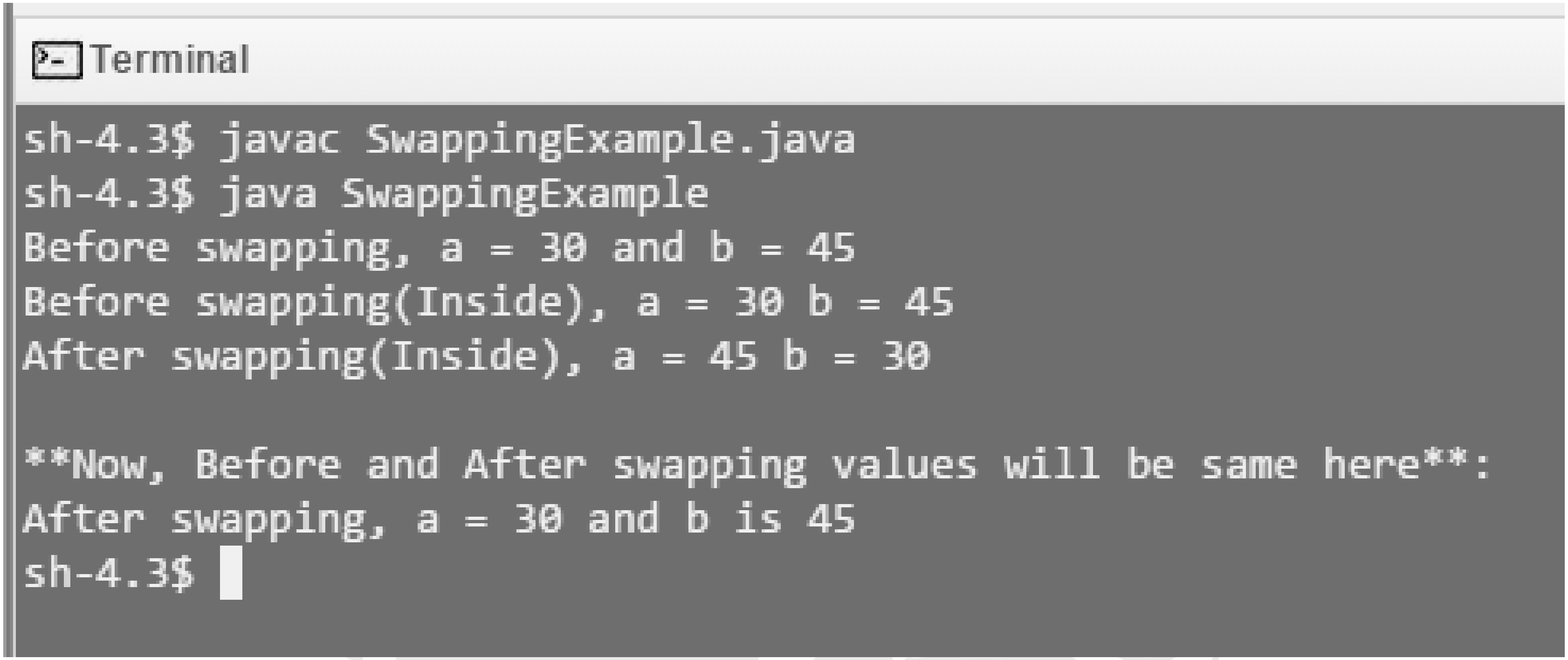
```
sh-4.3$ javac ExampleMaxNumber.java
sh-4.3$ java ExampleMaxNumber
Maximum Value = 11
sh-4.3$
```

The void Keyword and Call by Value



```
public class SwappingExample {

    public static void main(String[] args) {
        int a = 30;
        int b = 45;

        System.out.println("Before swapping, a = " + a + " and b = " + b);

        // Invoke the swap method
        swapFunction(a, b);
        System.out.println("\n**Now, Before and After swapping values will be same here**:");
        System.out.println("After swapping, a = " + a + " and b is " + b);
    }

    public static void swapFunction(int a, int b) {

        System.out.println("Before swapping(Inside), a = " + a + " b = " + b);
        // Swap n1 with n2
        int c = a;
        a = b;
        b = c;

        System.out.println("After swapping(Inside), a = " + a + " b = " + b);
    }
}
```

Output



```
sh-4.3$ javac SwappingExample.java
sh-4.3$ java SwappingExample
Before swapping, a = 30 and b = 45
Before swapping(Inside), a = 30 b = 45
After swapping(Inside), a = 45 b = 30

**Now, Before and After swapping values will be same here**:
After swapping, a = 30 and b is 45
sh-4.3$
```

Call by Reference

- There is only call by value in JAVA, not call by reference.