**LL1)**

* **Pass by Value**: The method parameter values are copied to another variable and then the copied object is passed, that’s why it’s called pass by value.
* **Pass by Reference**: An alias or reference to the actual parameter is passed to the method, that’s why it’s called pass by reference.

**Ref:** [**https://www.journaldev.com/3884/java-is-pass-by-value-and-not-pass-by-reference**](https://www.journaldev.com/3884/java-is-pass-by-value-and-not-pass-by-reference)

Java is always Pass by Value and not pass by reference, we can prove it with a simple example.

**D:\All\_Automation\_Projects\SeleniumBasics\_And\_JavaPrograms\JavaPrograms\src\OopsConcepts\JavaIsPassByValue.java**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**LL2)**

Methods in Java

A method is a collection of statements that perform some specific task and return result to the caller. A method can perform some specific task without returning anything. Methods allow us to **reuse** the code without retyping the code. In Java, every method must be part of some class which is different from languages like C, C++ and Python.  
Methods are **time savers**and help us to **reuse** the code without retyping the code.

In general, method declarations has six components :

* **Modifier**-: Defines **access type** of the method i.e. from where it can be accessed in your application. In Java, there 4 type of the access specifiers.
  + public: accessible in all class in your application.
  + protected: accessible within the class in which it is defined and in its **subclass(es)**
  + private: accessible only within the class in which it is defined.
  + default (declared/defined without using any modifier) : accessible within same class and package within which its class is defined.
* **The return type** : The data type of the value returned by the the method or void if does not return a value.
* **Method Name** : the rules for field names apply to method names as well, but the convention is a little different.
* **Parameter list**: Comma separated list of the input parameters are defined, preceded with their data type, within the enclosed parenthesis. If there are no parameters, you must use empty parentheses ().
* **Exception list**: The exceptions you expect by the method can throw, you can specify these exception(s).
* **Method body**: it is enclosed between braces. The code you need to be executed to perform your intended operations.

**Method signature**: It consists of method name and parameter list (number of parameters, type of the parameters and order of the parameters). Return type and exceptions are not considered as part of it.  
Method Signature of above function:

max(int x, int y)