**2.Difference between copy by value and copy by reference.**

|  |  |  |
| --- | --- | --- |
| **Parameters** | **Call by value** | **Call by reference** |
| Definition | While calling a function, when you pass values by copying variables, it is known as "Call By Values." | While calling a function, in programming language instead of copying the values of variables, the address of the variables is used it is known as "Call By References. |
| Arguments | In this method, a copy of the variable is passed. | In this method, a variable itself is passed. |
| Effect | Changes made in a copy of variable never modify the value of variable outside the function. | Change in the variable also affects the value of the variable outside the function. |
| Alteration of value | Does not allow you to make any changes in the actual variables. | Allows you to make changes in the values of variables by using function calls. |
| Passing of variable | Values of variables are passed using a straightforward method. | Pointer variables are required to store the address of variables |
| Value modification | Original value not modified. | The original value is modified. |
| Memory Location | Actual and formal arguments will be created in different memory location | Actual and formal arguments will be created in the same memory location |
| Safety | Actual arguments remain safe as they cannot be modified accidentally. | Actual arguments are not Safe. They can be accidentally modified, so you need to handle arguments operations carefully. |
| Default | Default in many programming languages like C++.PHP. Visual Basic NET, and C#. | It is supported by most programming languages like JAVA, but not as default. |

**3. How to copy by value a composite datatype (array+objects)**

Copy by reference : In copy by reference both actual variable and copied variable refer to the same memory address. So change in either of variables will reflect in the other.

Copy by value : Unlike copy by reference in copy by value, the object is copied in to a new memory location, so the actual variable and copied variable refer to different locations. So changes in one variable doesnt reflect in the other.

In Javascript there are three different data types.

1. Primitive(String,Number,Boolean)
2. Special(Undefined,Null)
3. Composite(Object,Array,Function)

We can copy by value a composite data type using spread operator as follows

var a = [1,2,3];  
var b= [...a];  
b[3] = 4;  
console.log(a);  
console.log(b);

We can also use Object.assign method as follows

var a = [1,2,3];   
var b= Object.assign([],a)  
b[3] = 4;  
console.log(a);  
console.log(b);