Call by Value –

1. The original variable is not modified on changes in other variables.

2. Actual and copied variables will be created in different memory locations.

3. On passing variables in a function, any changes made in the passed variable will not affect the original one.

4. In call by value, function is called by directly passing the value of the variable as an argument. So any changes made inside the function does not affect the original value.

5. In call by value, parameters passed as an arguments create its **own copy.** So any changes made inside the function is made to the copied value not to the original value .

Features of call by value:

- Function arguments are always passed by value.
- It copies the value of a variable passed in a function to a local variable.
- Both these variables occupy separate locations in memory. Thus, if changes are made in a particular variable it does not affect the other one.

Call by Reference –

1. The original variable gets modified on changes in other variables.

1. Actual and copied variables are created in the same memory location.

2. On passing variables in a function, any changes made in the passed parameter will update the original variable's reference too.

3. In call by Reference, Function is called by directly passing the reference/address of the variable as an argument. So changing the value inside the function also change the original value.

4. In JavaScript **array and Object** follows pass by reference property.

5. In call by reference, parameters passed as an arguments does not create its own copy, it refers to the original value so changes made inside function affect the original value.

Features of call by value:

- In JavaScript, all objects interact by reference.
- If an object is stored in a variable and that variable is made equal to another variable then both of them occupy the same location in memory.
- Changes in one object variable affect the other object variable.