**why there is a difference in behaviour for copying contents in primitive and non primitive type?**

Data types are used to store different kinds of variables.

Basically there are six data types which is divided into 3 categories:

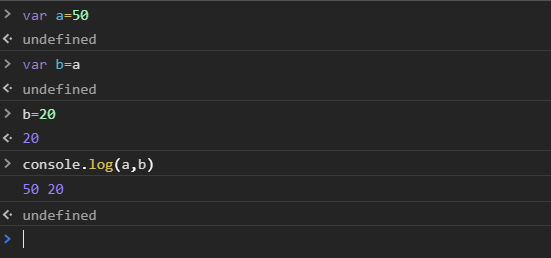
Primitive data types: →Strings, Number & Boolean.

Composite data types: →Object, Array, Function(All these 3 are objects).

Special data types: →Undefined and Null.

**Copy by value(deep copy)**

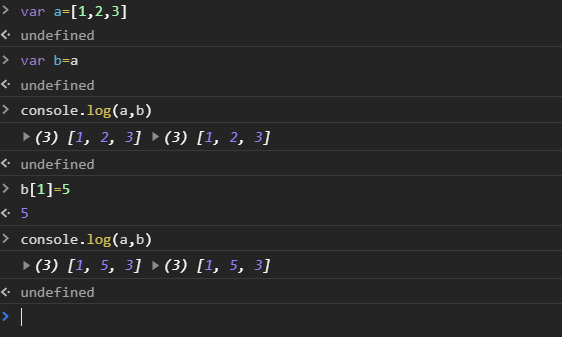
In this, the data which is stored in the variable is passed to another variable.



When we assign these variables to other variables using ‘**=**’, we **copy** the value to the new variable. Therefore they are **copied by value**.However, it should be noted that these exist as separate variables and that just the values are copied. These variables in no way are related to each other, they simply carry the same values.Both the variable points to two different memory location.

**Copy by reference(shallow copy)**

Composite data types are copied by reference.



When an object, such as an array, is copied to another variable using the assignment operator “**=**”**,** the **address** of the object is what’s copied , instead of the actual value. This is why objects are normally known to copied by reference than value.

Here change in data of variable “b” will lead to change in data of the original variable that is the variable “a” ,that is both the orginal variable and copied variable points to same memory location.

The behaviours of deep and shallow copies differ in terms of memory allotment. Deep copy allots memory to the variable that carries a primitive’s value. While, shallow copy simply points to the address of the original object when copied to another variable. No new memory is allotted in case of shallow copy so it saves up on a considerable amount of memory that would otherwise be unnecessarily used up by deep copying.