1. **Give a write-up on the Difference between copy by value and copy by reference.**

* In a primitive data-type when a variable (a) is assigned a value. And this variable is used to another variable (let b=a) contains the same value they are not connected to each other. It is so because the values are directly copied into the new variables.
* In case of a non-primitive data-type the values are not directly copied it references the main variable.
* This happens because both ‘main’ and ‘copied’ variable are stored at the same memory location. And when one changes the values in the other variable changes automatically.

|  |  |
| --- | --- |
| **Copy by value** | **Copy by reference** |
| 1. The original variables are not modified on changes in other variables. | 1. The original variables get modified on changes in other variables. |
| 1. Actual and copied variables will be created in different memory locations. | 1. Actual and copied variables are created in the same memory location. |
| 1. On passing variables in a function, any changes made in the passed variable will not affect the original one. | 1. On passing variables in a function, any changes made in the passed parameter will update the original variable’s reference too. |
| 1. All Primitive data types are copied by values. | 1. All composite data types are copied by reference. |

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

* **Using JSON.parse() and JSON.stringify()**

The JSON.parse() takes a JSON string and transforms it into a JavaScript object. JSON.stringify() takes a JavaScript object and transforms it into a JSON string.Using JSON.parse() and JSON.stringify() for copy and this performs deep copy

.

* **Using Object.assign()**

The **Object.assign()** method copies all enumerable own properties from one or more source object to a target object. It returns the target object. Note this will be a shallow copy. the empty [] as the first argument, this will ensure you don't mutate the original object.

* **Using Spread operator**

It allows an iterable to expand in places where 0+ arguments are expected. It is mostly used in the variable array where there is more than 1 values are expected. It allows us the privilege to obtain a list of parameters from an array.Using spread will clone your object. Note this will be a shallow copy.