**TASK 2**

* **Write a write up on Difference between copy by value and copy by reference.**

**Copy by value:** In primitive data-type, when variable is assigned with a value, we can say that certain memory is assigned to the variable and in that memory the value is stored.

var x = 10;

var y = x;

In above situation value is assigned to the variable `x` and this value is copied into the variable `y`. Here `x` and ` y` are not connected with each other and changes to either of them doesn’t affect the another.

**Copy by reference:** In case of a non-primitive data-type the values are not directly copied. When a non-primitive data-type is assigned a value a box is created with a sticker of the name of the data-type. However, the values it is assigned is not stored directly in the box. The language itself assigns a different memory location to store the data. The address of this memory location is stored in the box created.

So when the non-primitive variables are copied, the address of memory location is copied to the variable. So two variables becomes connected.

When we make changes to one variable, values are changed at the referenced memory location, resulting in the change in another variable also.

* **How to copy by value a composite data type (array+objects).**

There are three ways to copy composite data types by value:

1. **Using spread (…)** **operator:**

Spread operator allows iterable to expand in places where 0+ arguments are expected. Using this, we can expand variable array conveniently. Spread operator will clone your object, and then later we can assign that cloned object to another variable/object, thus achieving copy by value.

var arr = [1, 2, 3];

var copied\_arr = …arr;

1. **Using Object.assign():**

The ****Object.assign()**** method copies all enumerable own properties from one or more source objects to a target object. It returns the target object.

var a = [1, 2, 3];

var b = Object.assign([], a);

console.log(b); //[1, 2, 3]

b[2] = 4545;

console.log(b); //[1, 2, 4545]

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

The JSON object, available in all modern browsers, has two useful methods to deal with JSON-formatted content: parse and stringify. 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 performs deep copy .