Blog Write-Up Task:

1. Difference between copy by value and copy by reference.

In JavaScript, we have primitive data type and composite data type.
Primitive data types include:

- ✓ Number – It can be integer or float values
- ✓ String – It can be a single character or a collection of characters
- ✓ Boolean – It has true or false value.
- ✓ Null – A value that exists but is empty (Also called trivial data type).
- ✓ Undefined – A value that does not exist. (Also called trivial data type)

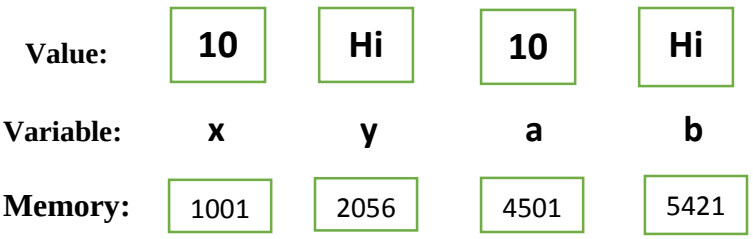
Composite data types include:

- ✓ Arrays
- ✓ Objects
- ✓ Function

Important Note:

- In JavaScript all composite data is internally stored as an Object.
- Primitive data types are copy by value and composite data type is copy by reference.

Copy by value:

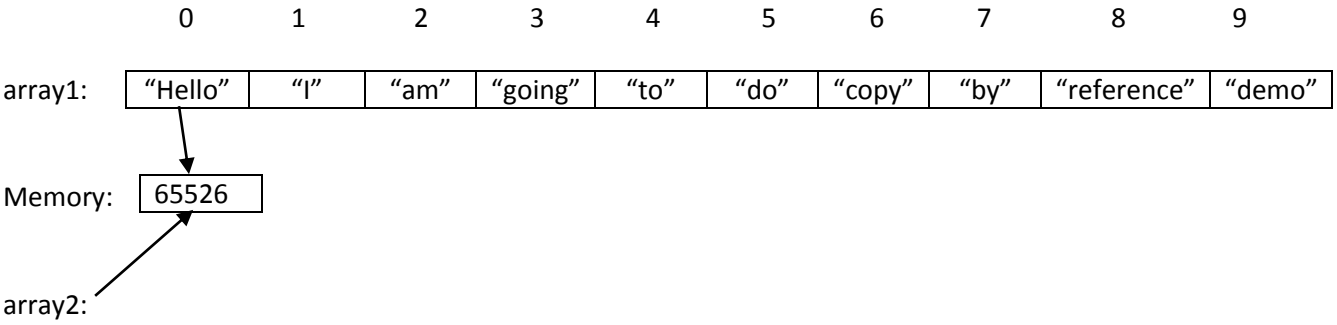


Let us have a variable x that contain the value 10. In the memory a separate space for the variable x will be created and will hold the value 10. When we assign a new variable a to x, only the value that x holds is copied to a and a has a different location allocated to it. This is called as copy by value. Changing the value of x or a has no effect on each other.

Code:

```
x = 10;
y = ”Hi”;
a = x; // only value of x copied to a
b = y; // only value of y copied to b
```

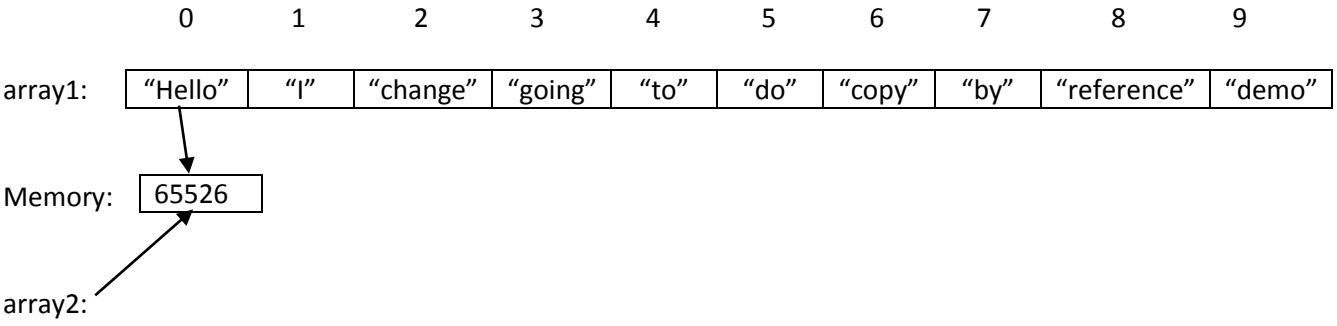
Copy by reference:



Let us have a string array, array1 with the values given as above. Let's take a new array, array2 and have array1 assigned to it. This will not copy all the values from array1, instead it will just hold/point to the reference/memory of array1. Any changes made to the array1 will affect array2 and vice versa.

Code:

```
array1 = ["Hello","I","am","going","to","do","copy","by","reference","demo"];
array2 = array1; //points to array1
array1.splice(2,1,"change"); // replacing am with change in array1
```



This is how the array changes and when we try to print the values in array2 it will print.

"Hello","I","change","going","to","do","copy","by","reference","demo"

So changing/altering the data on 1 array will affect the other array as well.

Copy by Value	Copy by Reference
1. The values that are copied to and from the variables has no relation to each other. They have different memory location	1. The values are not actually copied instead the copied variable holds the memory location of the other variable.
2. Modifying the values have no effect on both the variables.	2. Modifying the value in one variable changes the value in the other variable as well.

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

For Object:

To copy an object by value we can use stringify() of JSON. It converts the object to strings. So can take each key: value pair to strings and then assign them to the new object.

Code:

```
let person = {
    firstName: 'Lee',
    MiddleName: 'Min',
    lastName: 'Ho',
    address: {
        city: 'Seoul',
        country: 'South Korea'
    }
};

let copiedPerson = JSON.parse(JSON.stringify(person));
copiedPerson.firstName = 'Kim';
console.log(copiedPerson); //firstName is changed to Kim
console.log(person); //firstName remains the same “Lee”
```

For Arrays:

Copying Arrays by values can be done by different methods,

- ✓ Using spread operator (…)
- ✓ Using splice()
- ✓ Iterating through each value and pushing it to the new array.

Code:

```
arr1=[1,3,4];
arr2 = arr1.slice();
arr3 = [...arr1];
arr2[0]=100; //changing first value in arr2
arr3[0]=500; //changing first value in arr3
arr4=[];
for(var i=0;i<arr1.length;i++)
arr4.push(arr1[i]);
arr4[0]=200; //changing first value in arr4
console.log(arr1); //1,3,4
console.log(arr2); //100,3,4
console.log(arr3); //500,3,4
console.log(arr4); //200,3,4
```