String method doesn't replace the original string

```
// Returns string length and delete string value
STRING.length

// Part of the string {end value - Negative allowed}
STRING.slice(start,end)

// Part of the string {end value - Negative allowed but treated as ZERO}
STRING.substring(start,end)

// Part of the string
STRING.substr(start,length)

// Replace only first occurance string, if you need to replace multiple go with replaceAll or reg Exp
STRING.replace(searchString,replaceString)

// Replace all the occurance of the string {reg Exp also works}
STRING.replaceAll(searchString,replaceString)

// Given string into Uppercase
STRING.toUpperCase()

// Given string into Lowercase
STRING.toLowerCase()

// Merge one or more string
STRING.concat(STRING02,.....,STRING0N)

// Remove whitespace on both left and right of the string
STRING.trim()

// Remove whitespace on left of the string
STRING.trimStart()

// Remove whitespace on right of the string
STRING.trimEnd()

// Dummy/Extra string add in the beginning/End of the orginal string
STRING.padStart()
STRING.padEnd()

// Charter at particular index
STRING.charAt(index)

// Charter at particular index return ASCII code
STRING.charCodeAt(index)

// Convert String into Array
STRING.split()
```

ARRAY

reverse()
fill()
sort()
splice()

push()
pop()
shift()
unshift()

Above method change original array

// Return Array {Possible to delete array element by using length}
ARRAY.length

// All array element into single string
ARRAY.toString()

// Remove last element and return removed element
ARRAY.pop()

// Add new element and return new array length
ARRAY.push(addItem01,...,...,addItem0N)

// Shift removes the first element from an array and return the removed element
ARRAY.shift()

// Unshift add one or more element in the begining of the array and return new length
ARRAY.unshift(addItem01,...,...,addItem0N)

// Joins the array element with separator and return single string
ARRAY.join("")

//Delete an element {Better don't use it}
delete array[element]

//Concatinate one or more array
ARRAY.concat(array_01,.....,....,array_0N)

//The flat() method creates a new array with all sub-array elements concatenated into it recursively up to the specified depth.
ARRAY.flat(null/count-number)

// Change in the original array and return deleted element in another array
ARRAY.splice(start_index, remove_count, add_item1, ..., add_itemX)

//Does not Change the Original Array and return deleted element in another array

ARRAY.slice(start_index,end_index)

// Check atleast{some} one match in the given array
some(element,index,array)

// Check atmost{every} all match in the given array
every(element,index,array)

// Change/Alter within same array {Call for each element in the array}
forEach(element,index,array)

// Create a new array {Map for each element in the array}
map(element,index,array)

// Same as map {Eg:Array String to Character} Similar to flat method
flatMap(element,index,array)

//Filter an Array { Create a new array }
filter(element,index,array)

//Iterate array until get a single value { Take from left to right } total is initial value
reduce(intial,element,index,array)

//Iterate array until get a single value { Take from right to left } total is initial value
reduceRight(intial,element,index,array)

//indexOf {-1 element not found} From begining
indexOf(searchElement,start)

//lastIndexOf {-1 element not found} From Tail
lastIndexOf(searchElement,start)

//First element pass the test
find(element,index,array)

//First element pass the test
findIndex(element,index,array)

// Return True if element present in the array
includes(searchText)

//Keys PENDING
keys()

```
//entries PENDING
entries()

//Array.from() PENDING
Array.from()

//Rest and Spread Opartors



//It will not change the original array, return new array please carch it
ARRAY.with(index,Replacement);

//
ARRAY.from(OBJECT-WITH-LENGTH-PROPERTY);
```