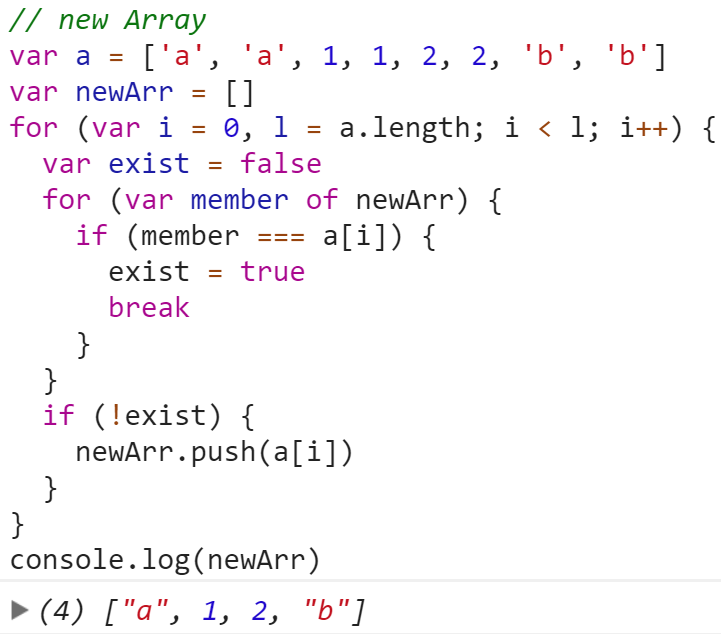
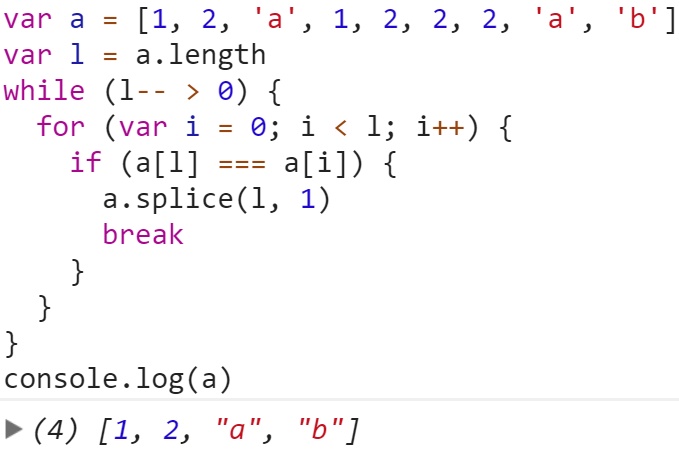
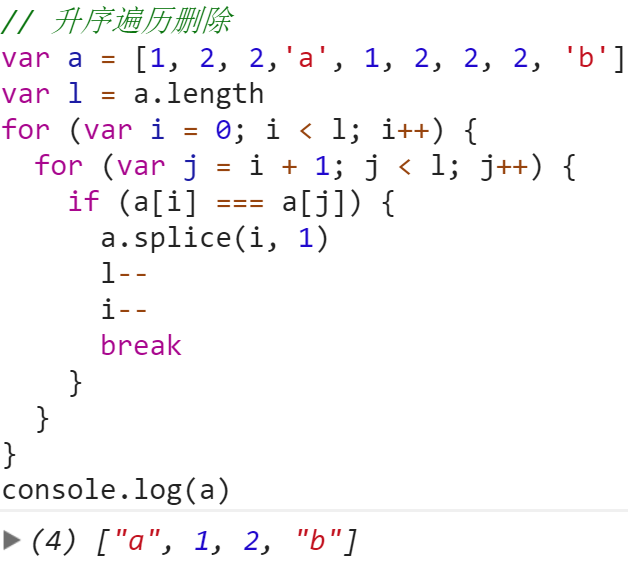
**JavaScript数组去重的几种方法**

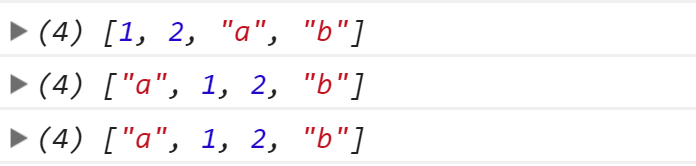
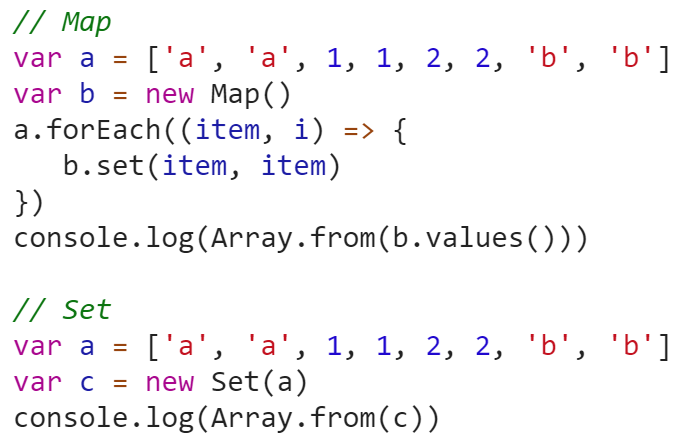
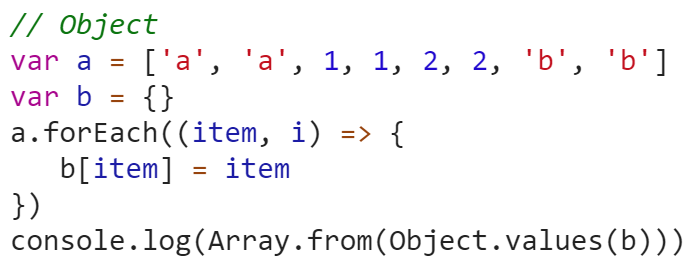
1. 数组去重的几种方法
2. **新建数组法**
3. **同一个数组删除法**
4. **利用object/map/set去重法**
5. **先排序再移除法**
6. **Filter与indexOf法**
7. 数组去重代码实现
8. 新建数组法，即新建立一个数组，用来存储结果，将原数组项逐个与新数组比较，如果不存在就添加到新数组。时间复杂度：O(n^2)



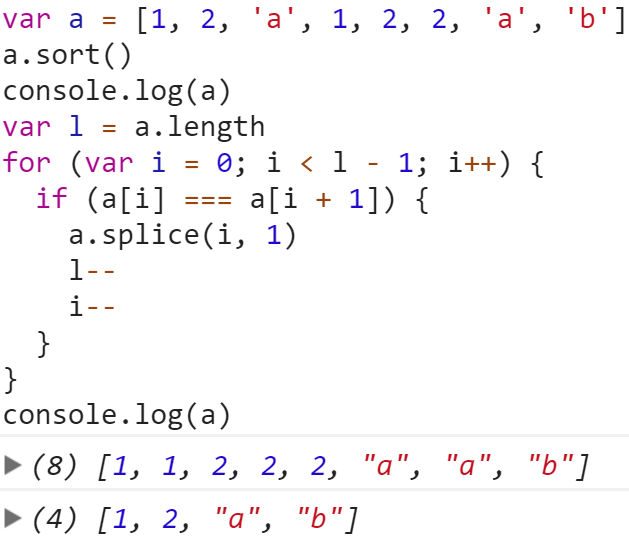
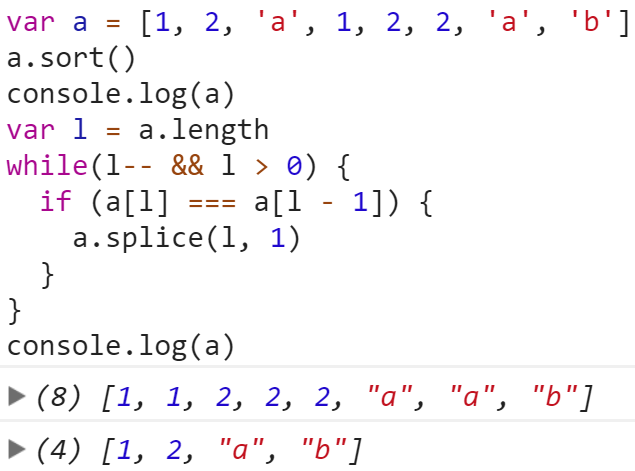
1. 同一个数组删除法，两个循环，将数组中的每个元素与其他未比较的元素进行比较，遇到有重复时，将自己删除。时间复杂度：O(logN)

1. 利用object/map/set去重法，利用数据结构key是唯一的特性来去重。时间复杂度：O(n^2)

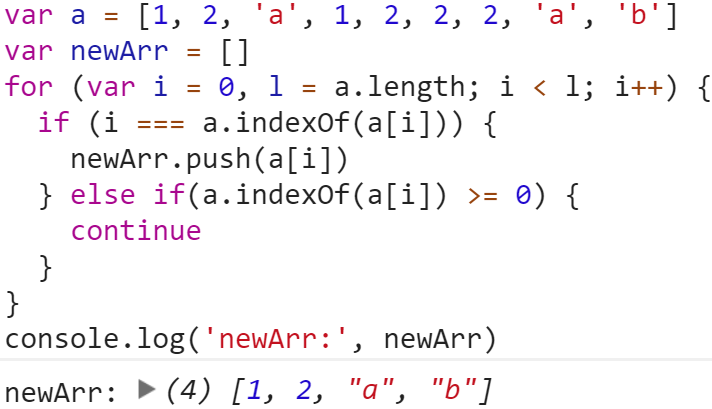


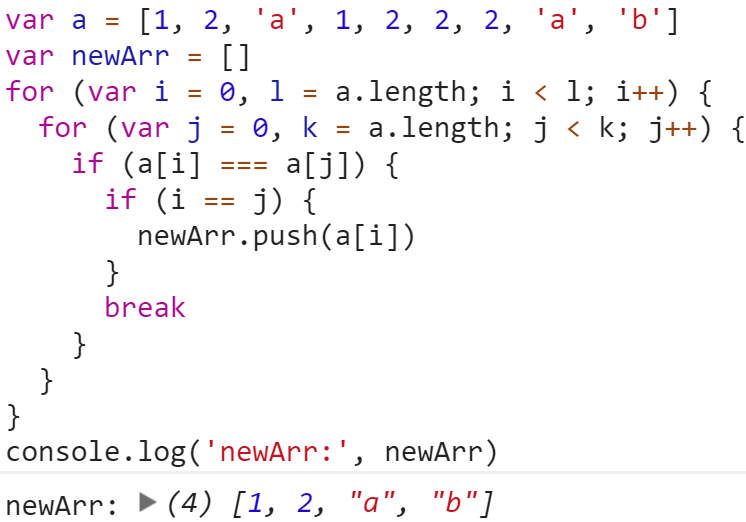
1. 先排序再移除法，先将数组排好序，然后从后至前或从前往后逐个与下一个进行比较，如果相同就删除当前项。时间复杂度：O(n^2)



1. Filter结合IndexOf法，利用indexOf返回数组中第一次出现目标项下标的特点，内容相同且下标的就是之前不存在的项目，把这种项追加到新数组中。时间复杂度：O(n^2)

最简写：**a.filter((item, i) => i === a.indexOf(item))**





附：

// 直接双循环+新数组

var a = ['a', 'a', 1, 1, 2, 2, 'b', 'b']

var newArr = []

for (var i = 0, l = a.length; i < l; i++) {

for (var j = 0, k = a.length; j < k; j++) {

if (a[i] === a[j]) {

if (i == j) {

newArr.push(a[i])

console.log(i, j, a[i], a[j])

}

break

}

}

console.log('newArr:', newArr)

}

// forEach + indexOf

var a = ['a', 'a', 1, 1, 2, 2, 'b', 'b']

var newArr = []

for (var i = 0, l = a.length; i < l; i++) {

if (i === a.indexOf(a[i])) {

console.log(newArr.push(a[i]))

} else if(a.indexOf(a[i]) >= 0) {

// 优化

continue

}

console.log('newArr:', newArr)

}