**JavaScript冒泡排序**

1. 冒泡排序概述

冒泡排序是通过遍历待排序的数列，一次比较两个元素，根据大小调换位置，直到把最大的或最小的冒出来的排序方式。

冒泡排序平均时间复杂度是：O(n^2)

步骤是：

1. 先建立两个循环，外循环用于遍历整个数组，内循环遍历待排序的区间。
2. 内循环每次都从第一项开始，将该项与后项进行比较，再两两交换，一直到待排序结尾。
3. 重复第二项，一直到数组遍历完。
4. 冒泡排序执行过程分析

**数列： 4, 1, 3, 5, 2**

1. **1, 3, 4, 2, 5**

1,4,3,5,2 => 1,3,4,5,2 => 1,3,4,5,2 => 1,3,4,2,5

1. **1, 3, 2, 4, 5**

1,3,4,2,5 => 1,3,4,2,5 => 1,3,2,4,5

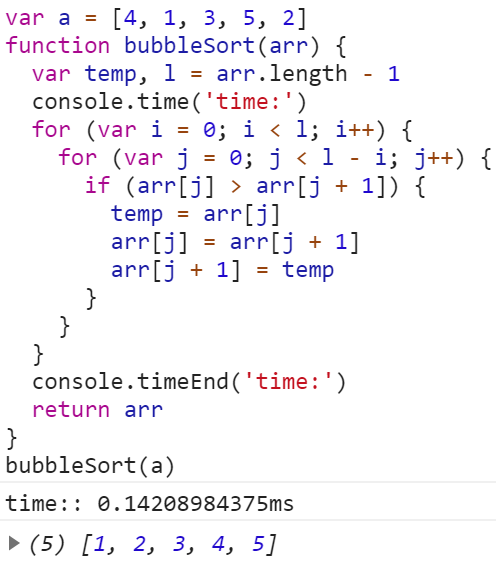
1. **1, 2, 3, 4, 5**

1,3,2,4,5 => 1,2,3,4,5

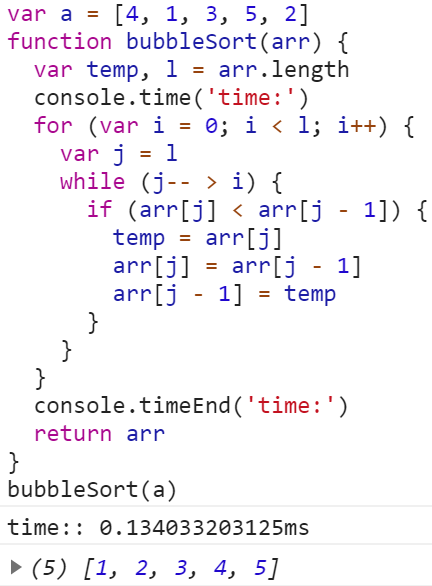
1. **1, 2, 3, 4, 5**

1,2,3,4,5

1. 冒泡排序实现
   1. 自左往右逐个将最大项冒出

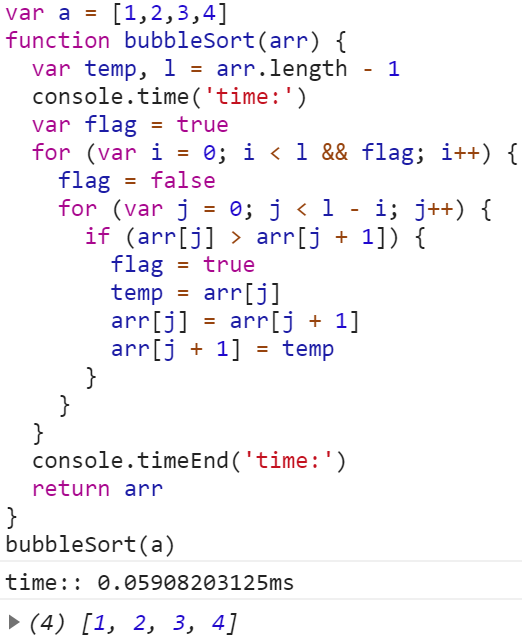


* 1. 自右往左逐个将最小项冒出



1. 冒泡排序的优化

加入一个标记，用来记录上一趟排序中是否发生过交换，如果没有进行过交换，说明当前数组已经排好序了，则不必再继续后面的遍历。



**源码：**

1、自左往右冒出最大项

var a = [4, 1, 3, 5, 2]

function bubbleSort(arr) {

var temp, l = arr.length - 1

console.time('time:')

for (var i = 0; i < l; i++) {

for (var j = 0; j < l - i; j++) {

if (arr[j] > arr[j + 1]) {

temp = arr[j]

arr[j] = arr[j + 1]

arr[j + 1] = temp

}

}

}

console.timeEnd('time:')

return arr

}

bubbleSort(a)

2、自右往左冒出最小项

var a = [4, 1, 3, 5, 2]

function bubbleSort(arr) {

var temp, l = arr.length

console.time('time:')

for (var i = 0; i < l; i++) {

var j = l

while (j-- > i) {

if (arr[j] < arr[j - 1]) {

temp = arr[j]

arr[j] = arr[j - 1]

arr[j - 1] = temp

}

}

}

console.timeEnd('time:')

return arr

}

bubbleSort(a)

3、加入一个标记，用来记录上一趟排序中是否发生过交换，如果没有进行过交换，说明当前数组已经排好序了，则不必再继续后面的遍历。这对于提前得到有序结果时可以省去后面的循环

var a = [1,2,3,4]

function bubbleSort(arr) {

var temp, l = arr.length - 1

console.time('time:')

var flag = true

for (var i = 0; i < l && flag; i++) {

flag = false

for (var j = 0; j < l - i; j++) {

if (arr[j] > arr[j + 1]) {

flag = true

temp = arr[j]

arr[j] = arr[j + 1]

arr[j + 1] = temp

}

}

}

console.timeEnd('time:')

return arr

}

bubbleSort(a)