[js十大排序算法：冒泡排序](https://www.cnblogs.com/ybygb-geng/p/9355425.html)

**排序算法说明:**

**（1）对于评述算法优劣术语的说明**

**稳定**：如果a原本在b前面，而a=b，排序之后a仍然在b的前面；  
**不稳定**：如果a原本在b的前面，而a=b，排序之后a可能会出现在b的后面；

**内排序**：所有排序操作都在内存中完成；  
**外排序**：由于数据太大，因此把数据放在磁盘中，而排序通过磁盘和内存的数据传输才能进行；

**时间复杂度**: 一个算法执行所耗费的时间。  
**空间复杂度**: 运行完一个程序所需内存的大小。

**(2)排序算法图片总结:**

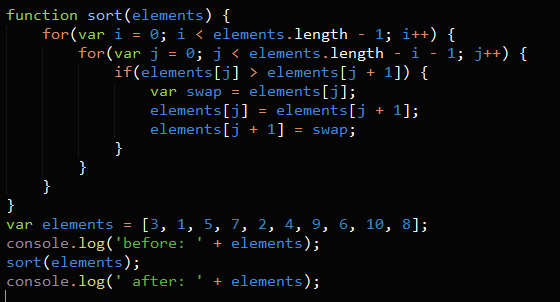
****

1.冒泡排序:

解析：1.比较相邻的两个元素，如果前一个比后一个大，则交换位置。

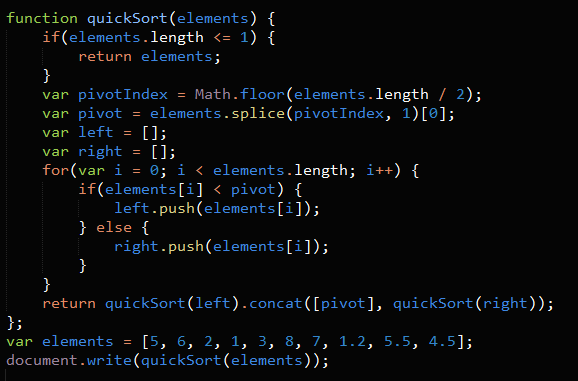
　　　2.第一轮的时候最后一个元素应该是最大的一个。

　　　3.按照步骤一的方法进行相邻两个元素的比较，这个时候由于最后一个元素已经是最大的了，所以最后一个元素不用比较。



2.快速排序:

解析：快速排序是对冒泡排序的一种改进，第一趟排序时将数据分成两部分，一部分比另一部分的所有数据都要小。然后递归调用，在两边都实行快速排序。



3.插入排序:

解析：

 （1） 从第一个元素开始，该元素可以认为已经被排序

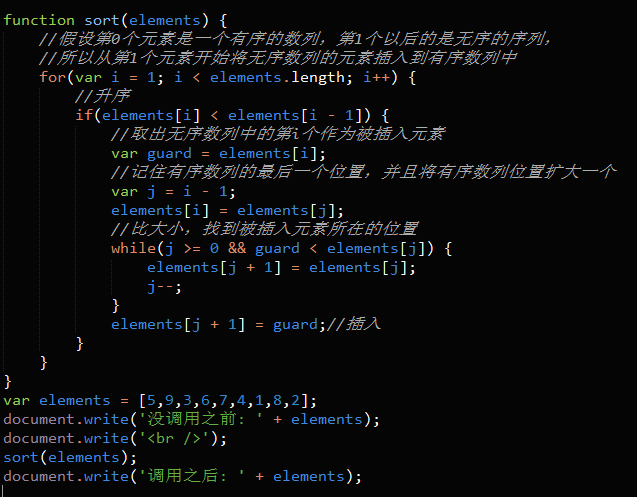
 （2） 取出下一个元素，在已经排序的元素序列中从后向前扫描

 （3） 如果该元素（已排序）大于新元素，将该元素移到下一位置

 （4） 重复步骤3，直到找到已排序的元素小于或者等于新元素的位置

 （5）将新元素插入到下一位置中

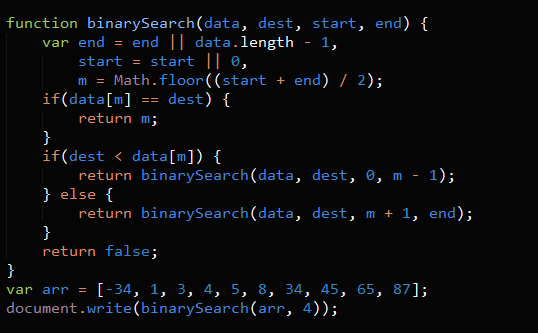
 （6） 重复步骤2



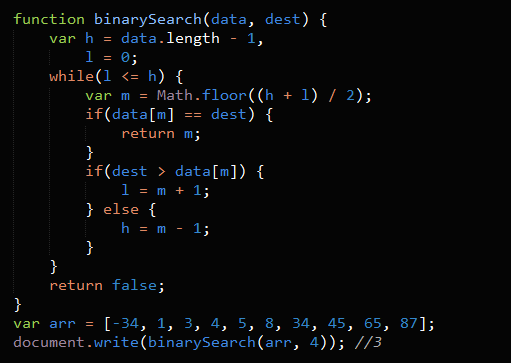
2.二分查找:

解析：二分查找，也为折半查找。首先要找到一个中间值，通过与中间值比较，大的放又，小的放在左边。再在两边中寻找中间值，持续以上操作，直到找到所在位置为止。

（1）递归方法



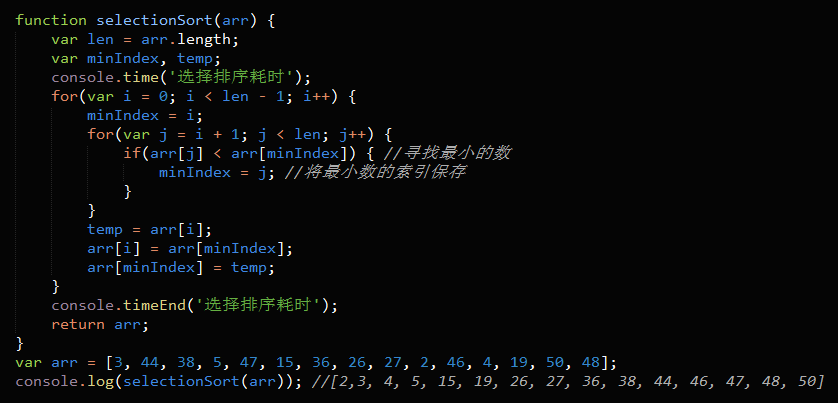
（2）非递归方法



4.选择排序:

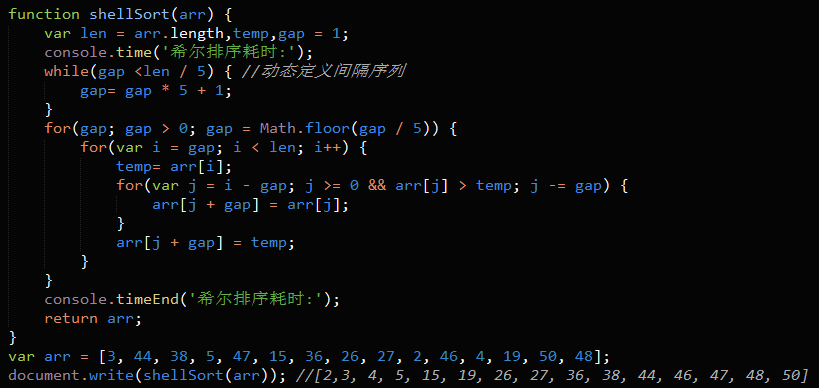
　　解析:首先在未排序序列中找到最小（大）元素，存放到排序序列的起始位置，然后，再从剩余未排序元素中继续寻找最小（大）元素，然后放到已排序序列的末尾。

　　　　以此类推，直到所有元素均排序完毕。



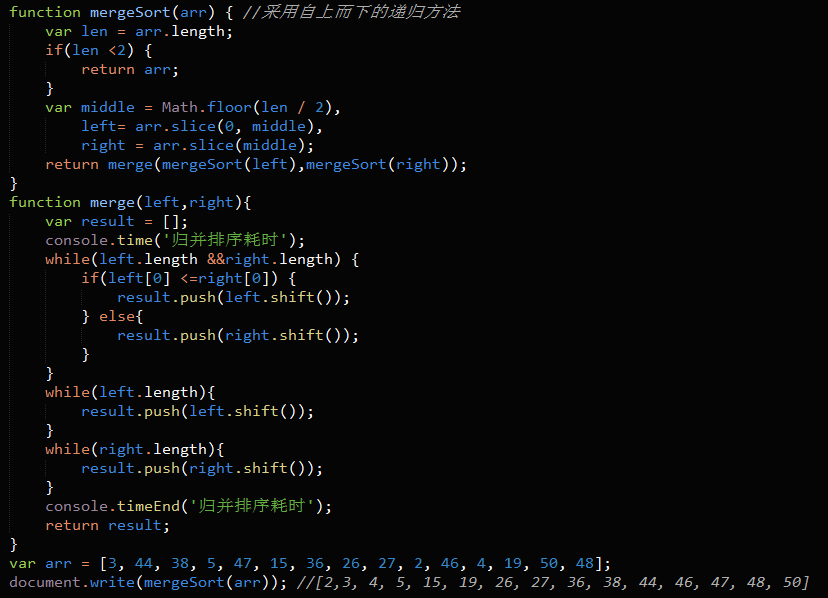
5.希尔排序:

解析:先将整个待排序的记录序列分割成为若干子序列分别进行直接插入排序



6.归并排序:

解析:归并排序是一种稳定的排序方法。将已有序的子序列合并，得到完全有序的序列；即先使每个子序列有序，再使子序列段间有序。



7.堆排序:

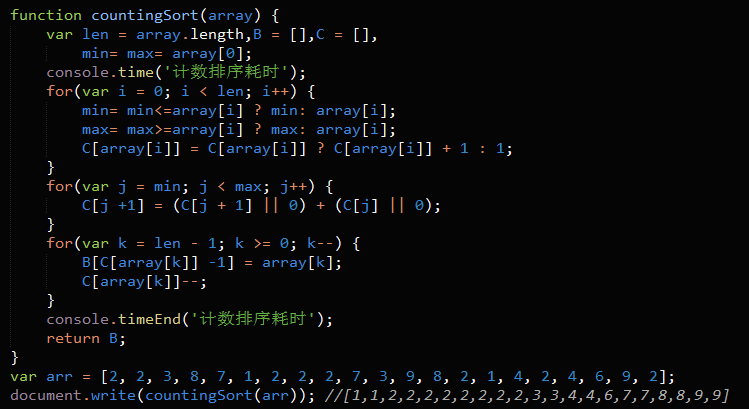
解析:堆排序（Heapsort）是指利用堆这种数据结构所设计的一种排序算法。堆积是一个近似完全二叉树的结构，并同时满足堆积的性质：即子结点的键值或索引总是

　　小于（或者大于）它的父节点。



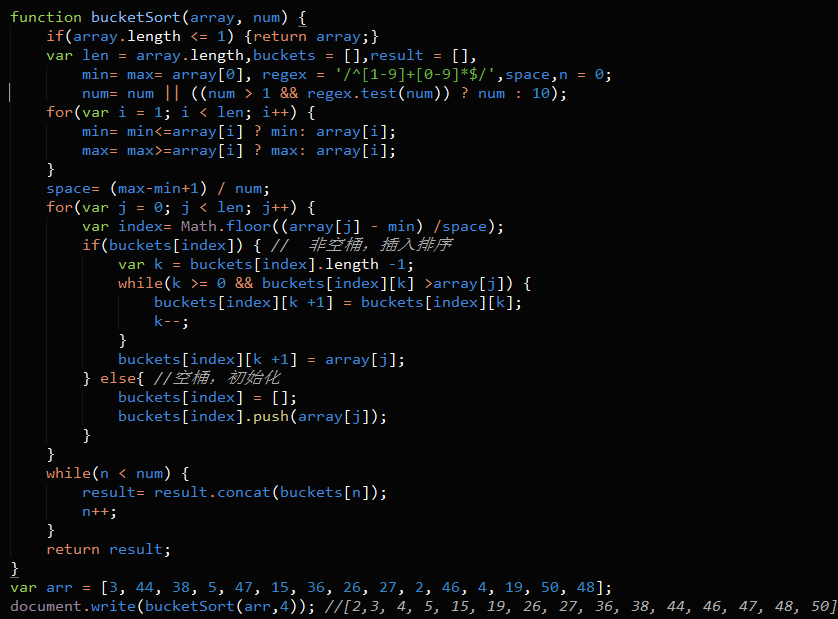
8.计数排序:

 解析:计数排序使用一个额外的数组C，其中第i个元素是待排序数组A中值等于i的元素的个数。然后根据数组C来将A中的元素排到正确的位置。它只能对整数进行排序。



9.桶排序:

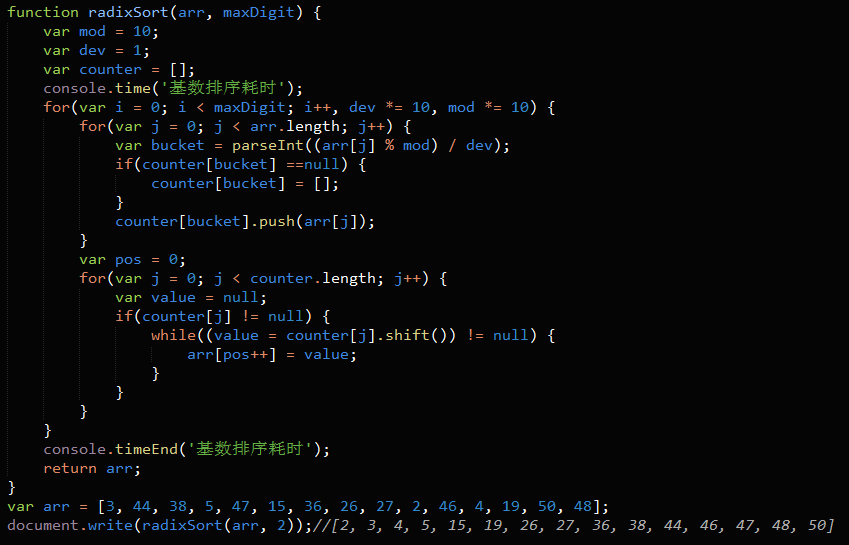
解析:假设输入数据服从均匀分布，将数据分到有限数量的桶里，每个桶再分别排序（有可能再使用别的排序算法或是以递归方式继续使用桶排序进行排



10.基数排序:

解析:基数排序是按照低位先排序，然后收集；再按照高位排序，然后再收集；依次类推，直到最高位。有时候有些属性是有优先级顺序的，先按低优先级排序，再按高优

　　先级排序。最后的次序就是高优先级高的在前，高优先级相同的低优先级高的在前。基数排序基于分别排序，分别收集，所以是稳定的。



**基数排序 vs 计数排序 vs 桶排序**

这三种排序算法都利用了桶的概念，但对桶的使用方法上有明显差异：

基数排序：根据键值的每位数字来分配桶 计数排序：每个桶只存储单一键值