

**定义**

假定在待排序的记录序列中，存在多个具有相同的关键字的记录，若经过排序，这些记录的相对次序保持不变，即在原序列中，A1=A2，且A1在A2之前，而在排序后的序列中，A1仍在A2之前，则称这种排序算法是稳定的；否则称为不稳定的。

稳定也可以理解为一切皆在掌握中,元素的位置处在你在控制中.而不稳定算法有时就有点碰运气,随机的成分.当两元素相等时它们的位置在排序后可能仍然相同.但也可能不同.是未可知的.

**判断方法**

对于不稳定的[排序算法](https://link.zhihu.com/?target=http://baike.baidu.com/view/297739.htm" \t "https://zhuanlan.zhihu.com/p/_blank)，只要举出一个实例，即可说明它的不稳定性；而对于稳定的排序算法，必须对算法进行分析从而得到稳定的特性。需要注意的是，排序算法是否为稳定的是由具体算法决定的，不稳定的算法在某种条件下可以变为稳定的算法，而稳定的算法在某种条件下也可以变为不稳定的算法。

算法思想的本身是独立于编程语言的,所以你写代码去实现算法的时候很多细节可以做不同的处理.采用不稳定算法不管你具体实现时怎么写代码,最终相同元素位置总是不确定的(可能位置没变也可能变了).而稳定排序算法是你在具体实现时如果细节方面处理的好就会是稳定的,但有些细节没处理得到的结果仍然是不稳定的.

例如，对于如下冒泡排序算法，原本是稳定的排序算法，如果将记录交换的条件改成a[j].key>=a[j+1].key，则两个相等的记录就会交换位置。

**常见算法的稳定性（要记住）**

**不稳定排序算法**

[堆排序](https://link.zhihu.com/?target=http://baike.baidu.com/view/157305.htm" \t "https://zhuanlan.zhihu.com/p/_blank)、[快速排序](https://link.zhihu.com/?target=http://baike.baidu.com/view/115472.htm" \t "https://zhuanlan.zhihu.com/p/_blank)、[希尔排序](https://link.zhihu.com/?target=http://baike.baidu.com/view/178698.htm" \t "https://zhuanlan.zhihu.com/p/_blank)、[直接选择排序](https://link.zhihu.com/?target=http://baike.baidu.com/view/3632419.htm" \t "https://zhuanlan.zhihu.com/p/_blank)

**稳定排序算法**

[基数排序](https://link.zhihu.com/?target=http://baike.baidu.com/view/1170573.htm" \t "https://zhuanlan.zhihu.com/p/_blank)、[冒泡排序](https://link.zhihu.com/?target=http://baike.baidu.com/view/254413.htm" \t "https://zhuanlan.zhihu.com/p/_blank)、[直接插入排序](https://link.zhihu.com/?target=http://baike.baidu.com/view/1193395.htm" \t "https://zhuanlan.zhihu.com/p/_blank)、[折半插入排序](https://link.zhihu.com/?target=http://baike.baidu.com/view/9681546.htm" \t "https://zhuanlan.zhihu.com/p/_blank)、[归并排序](https://link.zhihu.com/?target=http://baike.baidu.com/view/90797.htm" \t "https://zhuanlan.zhihu.com/p/_blank)

**稳定性的意义**

1、如果只是简单的进行数字的排序，那么稳定性将毫无意义。

2、如果排序的内容仅仅是一个复杂对象的某一个数字属性，那么稳定性依旧将毫无意义（所谓的交换操作的开销已经算在算法的开销内了，如果嫌弃这种开销，不如换算法好了？）

3、如果要排序的内容是一个复杂对象的多个数字属性，但是其原本的初始顺序毫无意义，那么稳定性依旧将毫无意义。

4、除非要排序的内容是一个复杂对象的多个数字属性，且其原本的初始顺序存在意义，那么我们需要在二次排序的基础上保持原有排序的意义，才需要使用到稳定性的算法，例如要排序的内容是一组原本按照价格高低排序的对象，如今需要按照销量高低排序，使用稳定性算法，可以使得想同销量的对象依旧保持着价格高低的排序展现，只有销量不同的才会重新排序。（当然，如果需求不需要保持初始的排序意义，那么使用稳定性算法依旧将毫无意义）。

**工业界的意义**

**1. 排序数值关联到其他信息，并且其他信息也会影响到顺序**

**2. 排序算法影响到程序的健壮性**

**3. 多次排序，想要延续以前排序顺序**