



思想

n个记录的文件的[直接选择排序](https://baike.baidu.com/item/%E7%9B%B4%E6%8E%A5%E9%80%89%E6%8B%A9%E6%8E%92%E5%BA%8F" \t "_blank)可经过n-1趟直接选择排序得到有序结果：

①初始状态：无序区为R[1..n]，有序区为空。

②第1趟排序

在无序区R[1..n]中选出[关键字](https://baike.baidu.com/item/%E5%85%B3%E9%94%AE%E5%AD%97" \t "_blank)最小的记录R[k]，将它与无序区的第1个记录R[1]交换，使R[1..1]和R[2..n]分别变为记录个数增加1个的新有序区和记录个数减少1个的新无序区。

……

③第i趟排序

第i趟排序开始时，当前有序区和无序区分别为R[1..i-1]和R(i..n）。该趟排序从当前无序区中选出关键字最小的记录 R[k]，将它与无序区的第1个记录R交换，使R[1..i]和R分别变为记录个数增加1个的新有序区和记录个数减少1个的新无序区。[1] 

解释

就像桌面上了一堆正面向上的纸牌，我们想要把纸牌有序的放在手中，每次都挑选一张最大或者最小的纸牌拿起来，当纸牌取完时，手里的纸牌就是有序的。

对比数组中前一个元素跟后一个元素的大小，如果后面的元素比前面的元素小则用一个变量k来记住他的位置，接着第二次比较，前面“后一个元素”现变成了“前一个元素”，继续跟他的“后一个元素”进行比较如果后面的元素比他要小则用变量k记住它在数组中的位置(下标)，等到循环结束的时候，我们应该找到了最小的那个数的下标了，然后进行判断，如果这个元素的下标不是第一个元素的下标，就让第一个元素跟他交换一下值，这样就找到整个数组中最小的数了。然后找到数组中第二小的数，让他跟数组中第二个元素交换一下值，以此类推。

时间复杂度

选择排序的交换操作介于 0 和 (n - 1） 次之间。选择排序的比较操作为 n (n - 1） / 2 次之间。选择排序的赋值操作介于 0 和 3 (n - 1） 次之间。

比较次数O(n^2），比较次数与关键字的初始状态无关，总的比较次数N=(n-1）+(n-2）+...+1=n\*(n-1）/2。交换次数O(n），最好情况是，已经有序，交换0次；最坏情况交换n-1次，逆序交换n/2次。交换次数比[冒泡排序](https://baike.baidu.com/item/%E5%86%92%E6%B3%A1%E6%8E%92%E5%BA%8F)少多了，由于交换所需CPU时间比比较所需的CPU时间多，n值较小时，选择排序比冒泡排序快。

稳定性

选择排序是给每个位置选择当前元素最小的，比如给第一个位置选择最小的，在剩余元素里面给第二个元素选择第二小的，依次类推，直到第n-1个元素，第n个元素不用选择了，因为只剩下它一个最大的元素了。那么，在一趟选择，如果一个元素比当前元素小，而该小的元素又出现在一个和当前元素相等的元素后面，那么交换后稳定性就被破坏了。比较拗口，举个例子，序列5 8 5 2 9，我们知道第一遍选择第1个元素5会和2交换，那么原序列中两个5的相对前后顺序就被破坏了，所以选择排序是一个不稳定的排序算法