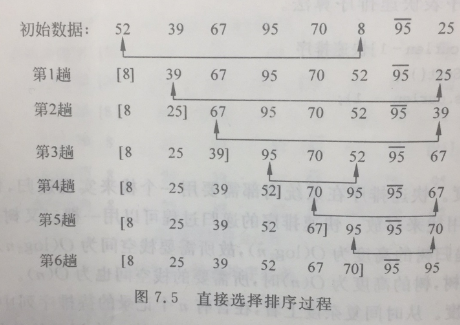
**选择排序的主要思想**

每一趟从待排序序列中选取一个关键字值最小的记录，也即第1趟从n个记录中选取关键字值最小的记录，在第2趟中，从剩下的n-1个记录中选取关键字值最小的记录，直到整个序列中的记录都选完位置。这样，由选取记录的顺序便可以得到按关键字值有序的序列。

**直接选择排序基本思想**

从第1趟中，从n个记录中找出关键字值最小的记录与第1个记录交换；在第2趟中，从第2个记录开始的n-1个记录中再选出关键字值最小的记录与第2个记录交换；以此类推，在第i趟中，从第i个记录开始的n-i+14个记录中选取关键字值最小的记录与第i个记录交换，直到整个关键字序列有序为止。



**步骤**

1.i记录要交换的位置，也就是要排序的位置，初始为0

2.当i<n-1时候，重复以下步骤

3.在无序子序列中选出一个关键字值最小的记录

4.若nums[min]不是nums[i]，则交换位置，否则不交换

5.i++

**代码**

public static void selectSort(int[] nums){//直接选择排序

int i = 0;

while(i<nums.length-1){//最后一个关键字nums[length-1]不用比较，在原位即可最大

//所以比较到length-2，i<length-1

//每一趟排序确定一个当前未排序序列中最小关键字的位置

int j = i+1; //从i+1开始比较

int minIndex = i; //minIndex记录最小关键字的坐标

while(j<nums.length){ //选取i后边最小的关键字

if(nums[j]<nums[minIndex]){ //当前关键字要是小于记录的最小关键字

minIndex = j;//记录该下标

}

j++;//大于略过

}

//因为初始的最小关键字下标是i，所以最开始关键字与最小关键字比较的关键字是nums[i]

//也就是最小关键字初始化为nums[i]

if(nums[minIndex]!=nums[i]){ //当最小关键字不等于nums[i]的时候交换

int temp = nums[i];

nums[i] = nums[minIndex];

nums[minIndex]=temp;

}

i++;//进行下一趟排序

}

}

**算法性能分析**

1.空间复杂度

仅用了一个辅助空间，空间复杂度未O(1)

2.时间复杂度

整个排序过程，关键字比较次数与初始关键字的状态无关。算法中没执行一次循环都必须进行一次关键字的比较，其外部循环共执行n-1次，内部循环共执行了n-1-i次，因为总的比较次数为

直接选择排序移动记录关键字比较少，最好情况，当前待排序序列记录序列有序时，移动记录次数为0.最坏情况，当前待排序序列逆序时候，移动记录次数为3(n-1)。

所以，直接排序算法的时间复杂度为O(n2)

**稳定性：不稳等**