排序 - 选择排序(Selection sort)

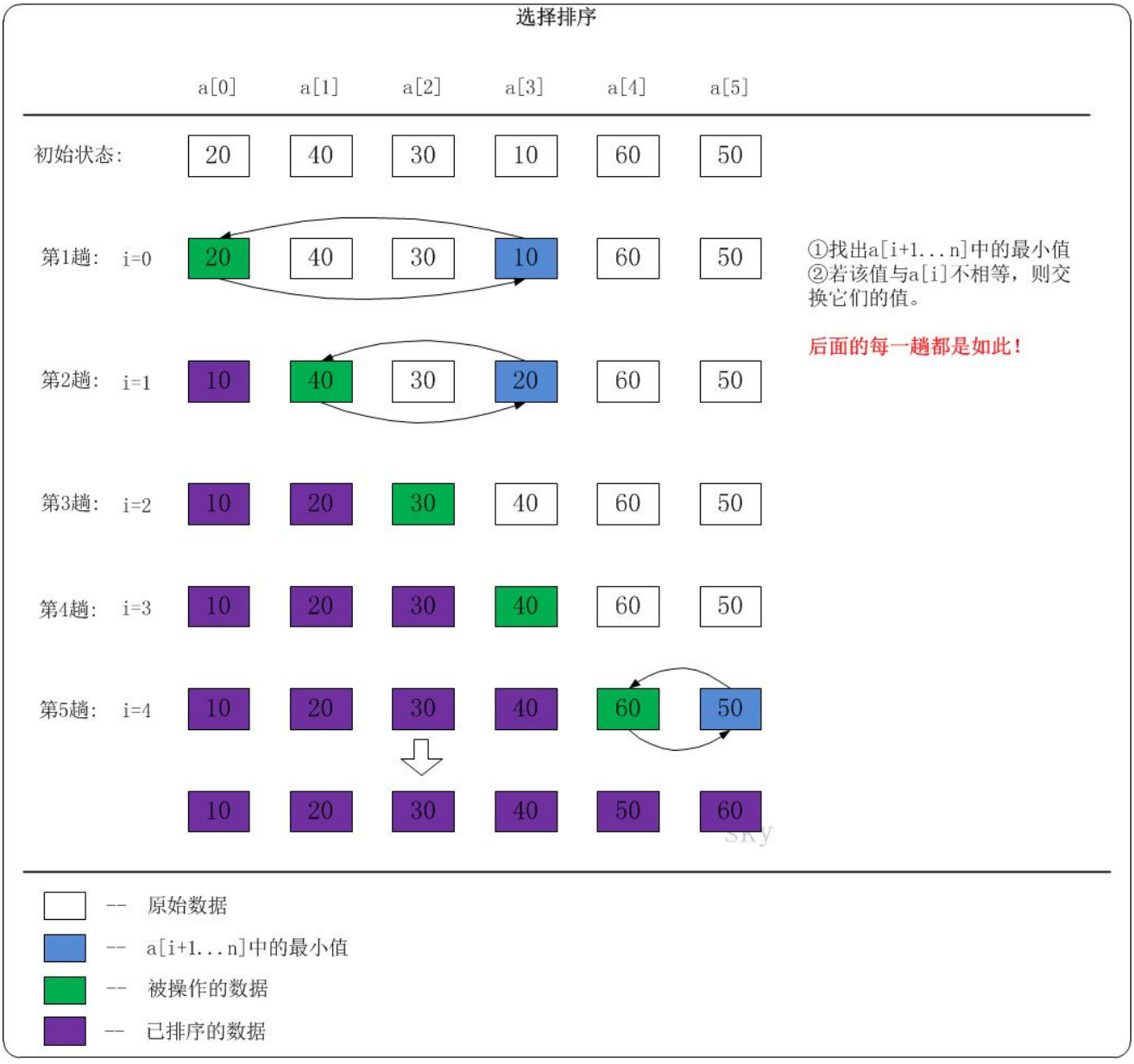
选择排序(Selection sort)是一种简单直观的排序算法。

选择排序介绍

它的基本思想是: 首先在未排序的数列中找到最小(or最大)元素，然后将其存放到数列的起始位置；接着，再从剩余未排序的元素中继续寻找最小(or最大)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。

选择排序实现

下面以数列{20,40,30,10,60,50}为例，演示它的选择排序过程(如下图)。



排序流程

- 第1趟: i=0。找出a[1...5]中的最小值a[3]=10，然后将a[0]和a[3]互换。 数列变化: 20,40,30,10,60,50 -- > 10,40,30,20,60,50

- 第2趟: i=1。找出a[2...5]中的最小值a[3]=20，然后将a[1]和a[3]互换。 数列变化: 10,40,30,20,60,50 -- > 10,20,30,40,60,50

- 第3趟: i=2。找出a[3...5]中的最小值，由于该最小值大于a[2]，该趟不做任何处理。

- 第4趟: i=3。找出a[4...5]中的最小值，由于该最小值大于a[3]，该趟不做任何处理。

- 第5趟: i=4。交换a[4]和a[5]的数据。 数列变化: 10,20,30,40,60,50 -- > 10,20,30,40,50,60

选择排序的时间复杂度和稳定性

选择排序时间复杂度

选择排序的时间复杂度是O(N2)。

假设被排序的数列中有N个数。遍历一趟的时间复杂度是O(N)，需要遍历多少次呢? N-1！因此，选择排序的时间复杂度是O(N2)。

选择排序稳定性

选择排序是稳定的算法，它满足稳定算法的定义。

算法稳定性 -- 假设在数列中存在a[i]=a[j]，若在排序之前，a[i]在a[j]前面；并且排序之后，a[i]仍然在a[j]前面。则这个排序算法是稳定的!

代码实现

```
public class SelectSort {

    /*
     * 选择排序
     *
     * 参数说明:
     *      a -- 待排序的数组
```

```
     *      n -- 数组的长度
     */
    public static void selectSort(int[] a, int n) {
        int i;          // 有序区的末尾位置
        int j;          // 无序区的起始位置
        int min;     // 无序区中最小元素位置

        for(i=0; i<n; i++) {
            min=i;

            // 找出"a[i+1] ... a[n]"之间的最小元素，并赋值给min。
            for(j=i+1; j<n; j++) {
                if(a[j] < a[min])
                    min=j;
            }

            // 若min!=i，则交换 a[i] 和 a[min]。
            // 交换之后，保证了a[0] ... a[i] 之间的元素是有序的。
            if(min != i) {
                int tmp = a[i];
                a[i] = a[min];
                a[min] = tmp;
            }
        }
    }

    public static void main(String[] args) {
        int i;
        int[] a = {20,40,30,10,60,50};

        System.out.printf("before sort:");
        for (i=0; i<a.length; i++)
            System.out.printf("%d ", a[i]);
        System.out.printf("\n");

        selectSort(a, a.length);

        System.out.printf("after  sort:");
        for (i=0; i<a.length; i++)
            System.out.printf("%d ", a[i]);
        System.out.printf("\n");
    }
}
```