# **排序算法**

## **排序的基本概念**

1. **什么是排序？**

**排序是指把一组数据以某种关系（递增或递减）按顺序排列起来的一种算法。**

**例如：数列 8、3、5、6、2、9、1、0、4、7**

**递增排序后 0、1、2、3、4、5、6、7、8、9**

**递减排序后 9、8、7、6、5、4、3、2、1、0**

1. **排序的稳定性**

**如果在一组需要排序的数据序列中，数据ki和kj的值相同，即ki= =kj，且在排序前ki在序列中的位置领先于kj，那么当排序后，如果ki和kj的相对前后次序保持不变，即ki仍然领先于kj，则称此类排序算法是稳定的。如果ki和kj的相对前后次序变了，即kj领先于ki了，则称此类排序算法是不稳定的。**

**3、排序的分类**

**●内部排序：指待排序数据全部存放在内存中进行排序的过程。**

**●外部排序：指待排序数据的数量很大，内存无法全部容纳所有数据，在排序过程中需要对外存进行访问的排序过程。**

**4、排序的过程**

**排序的过程中需要进行如下两种基本操作：**

1. **比较两个数据的大小；**
2. **移动两个数据的位置。**
3. **排序算法**

**排序算法按照其实现的思想和方法的不同，可以分为许多种。**

**我们比较常用的排序算法有：冒泡排序、 插入排序、选择排序、希尔排序（缩小增量排序）、快速排序、堆排序、归并排序。**

**6、排序算法的分类**

**排序算法按照其实现的思想和方法的不同，可以分为许多种。**

**我们比较常用的排序算法有：**

**交换类排序：冒泡排序、快速排序**

**插入类排序： 直接插入排序、希尔排序（缩小增量排序）**

**选择类排序：简单选择排序、堆排序**

**归并排序**

**基数排序**

**算法的时间复杂度和空间复杂度**

## **冒泡排序**

**冒泡排序的规则：n个数据进行冒泡排序，首先将第一个数据和第二个数据进行比较，如果为逆序就交换两个数据的值，然后比较第二个和第三个数据，依此类推，直到第最后一个和倒数第二个比较完了为止。上述过程为冒泡排序的第一趟冒泡排序，其结果是最大或者最小的数据被放置在末尾的位置。然后进行第二趟排序，把第二大或者第二小的数放置在倒数第二的位置，之后每一趟排序都会使一个数据有序，直到此序列的全部数据都有序为止。**

**void fun1(int x[],int n)//冒泡排序**

**{**

**int i, j;**

**for (i = 0;i<n-1;i++)**

**for (j = 0; j <n-1-i; j++)**

**{**

**if (x[j] > x[j + 1])**

**{**

**x[j] += x[j + 1];**

**x[j + 1] = x[j] - x[j + 1];**

**x[j] = x[j] - x[j + 1];**

**}**

**}**

**for (i=0;i<n;i++)**

**printf("%d\t",x[i]);**

**}**

## **选择排序**

**对一个序列进行选择排序，首先通过一轮循环比较，从n个数据中找出最大或者最小的那个数据的位置，然后按照递增或者递减的顺序，将此数据与第一个或最后一个数据进行交换。然后再找第二大或者第二小的数据进行交换，以此类推，直到序列全部有序为止。**

**选择排序与冒泡排序的区别在于，冒泡排序每比较一次后，满足条件的数据就交换，而选择排序是每次比较后，记录满足条件数据的位置，一轮循环过后再作交换。**

**void fun2(int x[], int n)//选择排序**

**{**

**int i, j, min,k;**

**for (i = 0; i < n; i++)**

**{**

**min=i;**

**for (j = i; j < n; j++)**

**{**

**if (x[min] > x[j])**

**min = j;**

**}**

**k = x[i];**

**x[i] = x[min];**

**x[min] = k;**

**}**

**for (i=0;i<n;i++)**

**printf("%d\t",x[i]);**

**}**

## **插入排序**

**插入排序的规则是：第一轮开始时默认序列中第一个数据是有序的，之后各个数据以此为基准，判断是插入在此数据的前面还是后面，之后的数据依次向后移动，腾出位置，让数据插入，以此类推，直到整个序列有序为止。**

**每比较一次，如果满足条件（升序：前面一个数比后面需要插入的数大），就直接交换。**

**void fun3(int x[], int n)//插入排序**

**{**

**int i, j, m, k;**

**for (i = 1; i < n; i++)**

**{**

**for (j = i; j >= 0; j--)**

**if (x[j - 1]>x[j])**

**{**

**k = x[j - 1];**

**x[j - 1] = x[j];**

**x[j] = k;**

**}**

**}**

**for (i = 0; i < n;i++)**

**printf("%d\t",x[i]);**

**}**

## **快速排序**

**快速排序的基本思想是：通过一趟排序将待排序的序列划分为独立的两部分，其中一部分的数据比另一部分的数据小，然后再分别对这两部分记录继续进行排序，直到整个序列有序。**

**一趟快速排序的具体做法是：设置两个标记low和high，它们初始值分别指向序列的第一个数据和最后一个数据，并设置一个中枢变量key，初始值记录序列第一个数据的值。首先high指针向前搜索，如果找到有数据小于中枢变量key的，则将其数据与中枢变量互相交换；然后low指针向后搜索，如果找到第一个关键字大于中枢变量key的，则将其数据与中枢变量互相交换；重复上述步骤，直到low==high为止。**

**void fun4(int x[],int low,int high)**

**{**

**int i, j, mid;**

**if (low < high)**

**{**

**mid = x[low]; i = low; j = high;**

**while (i < j)**

**{**

**while (i < j&&x[j] >= mid)j--;**

**if (i < j)x[i++] = x[j];**

**while (i < j&&x[i] <= mid)i++;**

**if (i < j)x[j--] = x[i];**

**}**

**x[i] = mid;**

**fun4(x, low, i - 1);**

**fun4(x, i + 1, high);**

**}**

**}**