

简单排序

2019年6月25日 20:06

原理：

简单排序

■ 冒泡排序



```
void Bubble_Sort( ElementType A[], int N )
{
    for ( P=N-1; P>=0; P-- ) {
        flag = 0;
        for( i=0; i<P; i++ ) { /* 一趟冒泡 */
            if ( A[i] > A[i+1] ) {
                Swap(A[i], A[i+1]);
                flag = 1; /* 标识发生了交换 */
            }
        }
        if ( flag==0 ) break; /* 全程无交换 */
    }
}
```

最好情况：顺序 $T = O(N)$

最坏情况：逆序 $T = O(N^2)$

稳定



Copyright © 2014, 浙江大学计算机科学与技术学院
All Rights Reserved

算法改进：添加flag，如果发现有一次的排序中已经完成了全部，则退出循环

优点：（1）可以用单向链表排序（2）严格大于的，稳定的

插入排序

简单排序

■ 插入排序



```
void Insertion_Sort( ElementType A[], int N )
{
    for ( P=1; P<N; P++ ) {
        Tmp = A[P]; /* 摸下一张牌 */
        for ( i=P; i>0 && A[i-1]>Tmp; i-- )
            A[i] = A[i-1]; /* 移出空位 */
        A[i] = Tmp; /* 新牌落位 */
    }
}
```

稳定

最好情况：顺序 $T = O(N)$

最坏情况：逆序 $T = O(N^2)$

例：给定初始序列{34, 8, 64, 51, 32, 21}，冒泡排序和插入排序分别需要多少次数组元素交换才能完成？

时间复杂度下界

- 对于下标 i, j ，则称 (i, j) 是一对逆序对(inversion)

- 问题：序列{34, 8, 64, 51, 32, 21}中有多少逆序对？ 9

(34, 8) (34, 32) (34, 21) (64, 51) (64, 32) (64, 21) (51, 32) (51, 21) (32, 21)

- 交换 2个相邻元素正好消去 1个逆序对！

- 插入排序： $T(N, I) = O(N+I)$ — 如果序列基本有序，则插入排序简单且高效

定理：任意 N 个不同元素组成的序列平均具有 $N(N-1)/4$ 个逆序对。

- 定理：任何仅以交换相邻两元素来排序的算法，其平均时间复杂度为 $O(N^2)$ 。

· 这意味着：要提高算法效率，我们必须：每次消去不止 1个逆序对！ 每次交换相隔较远的 2个元素！

代码：

```
#include<iostream>
```

```
using namespace std;
```

```
void swap(int *a, int *b)
```

```
{
```

```
    int temp;
```

```
    temp = *b;
```

```
    *b = *a;
```

```
    *a = temp;
```

```
}
```

```
void BubbleSort(int *a,int n)
```

```
{
```

```
    int count = 0;
```

```
    for (int p = n - 1; p >= 0; p--)
```

```
    {
```

```
        int flag = 0;
```

```
        for (int j = 0; j < p; j++)
```

```
        {
```

```
            if (a[j] > a[j + 1])
```

```
            {
```

```
                swap(a[j], a[j + 1]);
```

```
                count++;
```

```
            }
```

```
            flag = 1;
```

```
        }
```

```
        if (flag == 0) break;
```

```
    }
```

```
    cout << count;
```

```
}
```

```
void InsertSort(int *a,int n)
```

```
{
```

```
    int i;
```

```
    for (int j = 1; j < n; j++)
```

```
    {
```

```
        int key;//抽的卡
```

```
        key = a[j];
```

//每次排序一个数key，都要对比前面的对比该位置的值与该位置前面的值的大小，若key较小，则将与key对比的数往后移动一位。

```
        for (i = j; i > 0 && a[i - 1] > key; i--)
```

```
        {
```

```
            a[i] = a[i - 1]; //留空位
```

```

        }
        a[i] = key;//此时i是最顶头的一位了
    }

}

int main()
{
    int n;
    cin >> n;
    int* a = new int[n];
    for (int i = 0; i < n; i++)
    {
        cin >> a[i];
    }
    BubbleSort(a, n);
    //InsertSort(a, n);
    for (int i = 0; i < n; i++)
    {
        cout << a[i];
    }
    delete[] a;
}

```