冒泡排序法的改进是从后向前比较，第一轮是a[n]与a[n－1]，a[n－1]与a[n－2] …到a[2]与a[1]比较，小的数交换前移，记住这一轮最后发生两数交换的位置例如a[k]，下一轮扫描则可从a[n]开始比较到a[k]即可，因为上一轮比较中，a[k]之前的数据没有发生两数交换，说明这部分数据已经排好位置了……如果数据本身是有序的，一趟扫描即可完成排序，不必一定要扫描n－1轮，对部分有序的数据排序也可节约大量时间。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28 | //冒泡排序法的改进 — 有一类数据无法通过测试，请找出问题所在并解决  #include <bits/stdc++.h>  using namespace std;  int main()  {  int n;  cin>>n;  int a[n+1];//C++可动态定义数组大小,但不能同时赋值,如int a[n]={0};  for(int i=1; i<=n; i++)  scanf("%d",&a[i]);  int k=2,LastSwap;//k靠近数组左端的位置，LastSwap为最后交换的位置  while(k<n) //当左端a[k]的位置小于右端a[n]的位置,即还有数要比较  {  LastSwap=n; //先设定这一轮扫描最后交换位置为n  for(int j=n; j>=k; j--)//从后向前扫描到k（从前向后扫描也一样）  if(a[j]<a[j-1]) //如果后面的数比前面的数小  {  swap(a[j],a[j-1]); //交换a[j]和a[j-1]的值  LastSwap=j; //记录最后的变更位置  }  k=LastSwap; //下一轮只需比较到上一轮的LastSwap  }  for(int i=1; i<n; i++)  printf("%d ",a[i]);  printf("%d\n",a[n]);  return 0;  } |