山东大学 计算机科学与技术 学院

数据结构与算法 课程实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号：201900150221 | 姓名： 张进华 | | 班级： 智能19 |
| 实验题目：排序算法 | | | |
| 实验学时：2 | | 实验日期： 9.21 | |
| 实验目的：  掌握各种简单排序算法。 | | | |
| 软件开发工具：  visual studio code  g++ | | | |
| 1. 实验内容   用任意一种排序方式给出n个整数按升序排序后的结果，排序类需提供以下操作：名次排序、及时终止的选择排序、及时终止的冒泡排序、插入排序。   1. 数据结构与算法描述（整体思路描述，所需要的数据结构与算法）   名次排序：首先计算出数组中各个元素的名次，然后按照名次将元素移到相应位置，可通过构建新数组或者原地重排两种方式实现  及时终止的选择排序：首先找出最大的元素，把它移动到数组末尾a[n-1]，然后在余下的元素中找出最大的移动到a[n-2]，如此进行直到剩下一个元素。中间过程用sorted标记小范围数组是否已经排好序，若已排好，则退出。  及时终止的冒泡排序：通过n-1次排序，在每一次排序中，相邻元素比较，将最大元素移到最右边，中间过程用swapped标记是否已经部分排好序，若排好，则退出减少比较次数。  插入排序:首先在一个有序数组中插入元素就是遍历找到合适的位置插入该元素，而插入排序以此为基础，从第一个元素构成的单元素输入开始，不断地将新元素插入。   1. 测试结果（测试输入，测试输出）      1. 分析与探讨（结果分析，若存在问题，探讨解决问题的途径）   使用插入排序将数组进行排序，结果显示正确，完成了对数组的排序   1. 附录：实现源代码（本实验的全部源程序代码，程序风格清晰易理解，有充分的注释）   #include "bits/stdc++.h"  #define MAXSIZE 1000  using namespace std;  template <class T>  class SortFunction  {  public:      SortFunction(){};      ~SortFunction(){};      //名次计算      void rankArray(T a[], int n, int r[])      {          //给数组a中元素排序          for (int i = 0; i < n; i++)          { //初始化              r[i] = 0;          }          //比较排序          for (int i = 1; i < n; i++) //r记录名次          {              for (int j = 0; j < i; j++)              {                  if (a[j] <= a[i])                      r[i]++;                  else                      r[j]++;              }          }      }      //名次排序      void rearrange(T a[], int n, int r[])      {          //使用附加数组u将元素排序          T \*u = new T[n];          //将数组a中的元素按r中的顺序移到u中          for (int i = 0; i < n; i++)          {              u[r[i]] = a[i];          }          //将数组u中的元素移到a中          for (int i = 0; i < n; i++)              a[i] = u[i];          delete[] u;      }      //按名次排序(原地重排)      void rearrange2(T a[], int n, int r[])      {          //原地重排          for (int i = 0; i < n; i++)          {              //把正确的元素移动到a[i]              while (r[i] != i)              {                  int t = r[i];                  swap(a[i], a[t]);                  swap(r[i], r[t]);              }          }      }      //寻找最大元组位置      int indexOfMax(T a[], int n)      {          if (n < 0)          {              cout << "n must be > 0" << endl;          }          int indexOfMax = 0;          for (int i = 1; i < n; i++)          {              if (a[indexOfMax] < a[i])              {                  indexOfMax = i;              }          }          return indexOfMax;      }      //选择排序      void selectionSort2(T a[], int n)      {          //给数组a排序          for (int size = n; size > 1; size--)          {              int j = indexOfMax(a, size);              swap(a[j], a[size - 1]);          }      }      //及时终止的选择排序      void selectionSort(T a[], int n)      {          bool sorted = false;          for (int size = n; !sorted && (size > 1); size--)          {              int indexOfMax = 0;              sorted = true;              for (int i = 1; i < size; i++)              {                  //找最大元素                  if (a[indexOfMax] <= a[i])                      indexOfMax = i;                  else                      sorted = false; //无序              }              swap(a[indexOfMax], a[size - 1]);          }      }      //一次冒泡      bool bubble(T a[], int n)      {          //把数组a中最大的元素移到右边          bool swapped = false; //尚未发生交换          for (int i = 0; i < n - 1; i++)          {              if (a[i] > a[i + 1])              {                  swap(a[i], a[i + 1]);                  swapped = true;              }          }          return swapped;      }      //及时终止的冒泡排序      void bubbleSort(T a[], int n)      {          for (int i = n; i > 1 && bubble(a, i); i--)              ;      }      //向有序数组中插入元素      void insert(T a[], int n, const T &x)      {          int i;          //数组a容量大于n          for (i = n - 1; i >= 0 && x < a[i]; i--)          {              a[i + 1] = a[i];          }          a[i + 1] = x;      }      //插入排序      void insertionSort(T a[], int n)      {          for (int i = 1; i < n; i++)          {              T t = a[i];              insert(a, i, t);          }      }  };  int main(void)  {      int n;      cin >> n;      int \*number = new int[MAXSIZE];      for (int i = 0; i < n; i++)          cin >> number[i];      int r[MAXSIZE];      SortFunction<int> sort;      //sort.rearrange(number,n,r);      //sort.selectionSort2(number, n);      //sort.bubbleSort(number,n);      sort.insertionSort(number, n);      for (int i = 0; i < n; i++)          cout << number[i] << " ";      cout << endl;      return 0;  } | | | |