**2019秋季学期**

**数据结构实验报告**

**实验五排序实验**

**班级：\_\_\_\_计实验19\_\_\_\_\_\_\_\_**

**学号：\_\_\_\_\_19101130122\_\_\_\_\_\_\_**

**姓名：\_ 陈旻杰\_\_\_\_\_\_\_\_\_\_**

|  |
| --- |
| **评语：**  **日期：** |

**实验目的**

熟练掌握各种排序算法的实现方法，以及不同算法的特点，掌握各种排序方法的时间效率。

1、排序比较

**问题描述**：

各种内部排序算法的时间复杂度分析结果只给出了算法执行时间的阶，或大概执行时间。对一组给定的数据，采用各不同排序算法对其进行排序，给出各算法在排序中的关键字比较次数和关键字移动次数，以取得直观感受。

**基本要求：**

（1）对以下各排序算法进行比较：直接插入排序、冒泡排序、快速排序、简单选择排序、归并排序。

（2）待排序表的表长为100；对每组数据用以上各排序方法进行排序，比较的指标为有关键字参加的比较次数和关键字的移动次数。

**实现提示：**

在算法的适当地方加入计数操作，计算关键字的比较次数和移动次数。

**Input**

输入部分第一行为待排关键字的个数n，第二行为n个待排关键字，所有数据之间由空格分隔。

**Output**输出共1行，共有10个整数，表示5种排序方法排序的关键字比较次数和移动次数，即为：直接插入排序比较次数、直接插入排序移动次数、冒泡排序比较次数、冒泡排序移动次数、快速排序比较次数、快速排序移动次数、简单选择排序比较次数、简单选择排序移动次数、归并排序比较次数、归并排序移动次数。

**Sample Input**

**5**

**1 2 3 4 5**

**Sample Output**

**4 0 4 0 10 16 10 0 7 17**

**Sample Input**

**5**

**5 4 3 2 1**

**Sample Output**

**14 18 10 30 12 16 10 6 5 17**

**需求分析：**（包括对问题的理解，解决问题的策略、方法描述）

问题的理解：

分别通过插入排序，冒泡排序，快速排序，简单选择排序，归并排序计算排序过程中的比较次数和交换次数。

解决策略：

在插入排序，冒泡排序，快速排序，简单选择排序，归并排序过程中中记录比较次数和交换次数。

方法描述：

初始输入一个数组，然后再定义一个临时数组，初始数组不变，每次排序前都把初始数组的值赋给临时数组，然后将临时数组排序，并记录排序过程中元素的比较次数和交换次数。并将两个次数存放在二维数组s 中。

**系统设计：**（包括数据结构定义、抽象出基本操作描述、主程序模块处理过程描述）

***数据结构定义：***

const int maxn **=** 1010**;**

int a**[**maxn**];**

int b**[**maxn**];**

int s**[**5**][**2**]** **=** **{**0**};**

int n**;**

***抽象出基本操作描述：***

1. 临时数组赋值：void f**()**；

2. 插入排序：void insert\_sort**()**；

3. 冒泡排序：void bubble\_sort**()**；

4. 快速排序**;** void quick\_sort**();**

void quicksort**(**int l**,** int r**);**

5. 简单选择排序：void select\_sort**()**；

6. 归并排序：void merge\_sort**()**；

void mergesort**(**int l**,**int r**)**；

void merge**(**int l**,**int m**,**int r**)**；

***主程序模块处理过程描述:***

int main**()**

**{**

cin **>>** n**;**

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

**{**

cin **>>** a**[**i**];**

**}**

//插入

back**();**

insert\_sort**();**

//冒泡

back**();**

bubble\_sort**();**

//快排

back**();**

quick\_sort**();**

//选择

back**();**

select\_sort**();**

//归并

back**();**

merge\_sort**();**

**if(**s**[**0**][**0**]** **==** 155**)**

s**[**4**][**0**]--;**

**for(**int i **=** 0**;**i**<**5**;**i**++)**

**{**

**if(**i**)** cout **<<** " "**;**

cout **<<** s**[**i**][**0**]** **<<** " " **<<** s**[**i**][**1**];**

**}**

cout **<<** endl**;**

**return** 0**;**

**}**

**基本操作的实现：**（对各基本操作实现的描述）

1.插入排序

void insert\_sort**(){**

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

int v **=** b**[**i**];**

int j **=** i**-**1**;**

s**[**0**][**0**]++;**

**while(**j **>=** 0**&&**b**[**j**]** **>** v**){**

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

s**[**0**][**1**]++;**

j**--;**

s**[**0**][**0**]++;**

**}**

b**[**j**+**1**]** **=** v**;**

**if(**i **!=** j**+**1**)** s**[**0**][**1**]** **+=** 2**;**

**}**

**}**

2.冒泡排序：

void bubble\_sort**()**

**{**

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

**{**

int ans **=** 0**;**

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

**{**

s**[**1**][**0**]++;**

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

**{**

ans**++;**

int t **=** b**[**j**+**1**];**

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

b**[**j**]** **=** t**;**

s**[**1**][**1**]** **+=** 3**;;**

**}**

**}**

**if(!**ans**)** **break;**

**}**

**}**

3.快速排序：

void quicksort**(**int l**,** int r**)**

**{**

**if(**l **>=** r**)** **return** **;**

int temp **=** b**[**l**];**

s**[**2**][**1**]++;**

int i **=** l**;**

int j **=** r**;**

**while(**i **<** j**)**

**{**

s**[**2**][**0**]++;**

**while(**i**<**j**&&**b**[**j**]** **>=** temp**)**

**{**

j**--;**

**if(**i**<**j**)** s**[**2**][**0**]++;**

**}**

b**[**i**]** **=** b**[**j**];**

s**[**2**][**1**]++;**

**if(**i **<** j**)** s**[**2**][**0**]++;**

**while(**i**<**j**&&**b**[**i**]** **<=** temp**)**

**{**

i**++;**

**if(**i**<**j**)** s**[**2**][**0**]++;**

**}**

b**[**j**]** **=** b**[**i**];**

s**[**2**][**1**]++;**

**}**

b**[**i**]** **=** temp**;**

s**[**2**][**1**]++;**

quicksort**(**l**,**i**-**1**);**

quicksort**(**i**+**1**,**r**);**

**return** **;**

**}**

void quick\_sort**()**

**{**

quicksort**(**0**,**n**-**1**);**

**}**

4.简单选择排序

void select\_sort**()**

**{**

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

**{**

int minl **=** b**[**i**];**

int r **=** i**;**

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

**{**

s**[**3**][**0**]++;**

**if(**b**[**j**]** **<** minl**)**

**{**

minl **=** b**[**j**];**

r **=** j**;**

**}**

**}**

**if(**r **!=** i**)**

**{**

int t **=** b**[**i**];**//一次交换

b**[**i**]** **=** b**[**r**];**

b**[**r**]** **=** t**;**

s**[**3**][**1**]** **+=** 3**;**

**}**

**}**

**}**

5.归并排序：

void merge**(**int l**,**int m**,**int r**){**

int nl **=** m**-**l**+**1**;**

int nr **=** r**-**m**;**

int **\***al **=** **new** int**[**nl**+**1**];**

int **\***ar **=** **new** int**[**nr**+**1**];**

**for(**int i **=** 0**;**i**<**nl**;**i**++){**

al**[**i**]** **=** b**[**l**+**i**];**

**}**

**for(**int i **=** 0**;**i**<**nr**;**i**++){**

ar**[**i**]** **=** b**[**m**+**i**+**1**];**

**}**

al**[**nl**]** **=** 1000000**;**

ar**[**nr**]** **=** 1000000**;**

**for(**int i **=** 0**,**j **=** 0**,**k **=** l**;**k **<=** r**;**k**++){**

**if(**i**<**nl**&&**j**<**nr**)** **{**

s**[**4**][**0**]++;**

**}**

**if(**al**[**i**]>**ar**[**j**]){**

b**[**k**]** **=** ar**[**j**++];**

**}**

**else** **{**

b**[**k**]** **=** al**[**i**++];**

**}**

**}**

**delete** **[]**al**;**

**delete** **[]**ar**;**

**}**

void mergesort**(**int l**,**int r**){**

//cout << l << " " << r << endl;

s**[**4**][**1**]** **+=** r**-**l**+**1**;**

**if(**l**<**r**){**

int m **=** **(**l**+**r**)/**2**;**

mergesort**(**l**,**m**);**

mergesort**(**m**+**1**,**r**);**

merge**(**l**,**m**,**r**);**

**}**

**else** **return** **;**

**}**

void merge\_sort**(){**

mergesort**(**0**,**n**-**1**);**

**}**

6.复位函数

void back**()**

**{**

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

**{**

b**[**i**]** **=** a**[**i**];**

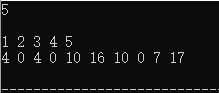
**}**

**}**

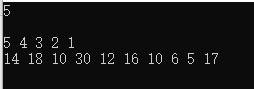
描述：每次排序之后将排序好的数组重新变为原来的初始数组

**测试结果：**（输入的测试数据及运行结果、正确性、在线测试情况）

***样例一：***



***样例二：***



**调试分析：**（包括调试过程中对原设计的修改，以及遇到的问题和解决的方法）

在处理交换次数的时候，一次赋值为一次交换，一次t代换为三次交换。

t代换;

t = x; x = y; y = t;

**小结：**

通过本次实验，我进一步学习并掌握了插入排序，冒泡排序，快速排序，简单选择排序，归并排序五种排序方式，理解了不同排序方式的复杂度差异。