# 《数据结构》实验报告八

|  |  |  |
| --- | --- | --- |
| 题目： 排序算法 | | 班级： 通信1601 |
| 学号： 29 |  | 姓名： 郑祖闯 |
| 日期： |  | 程序名： |

### 上机实验的问题和要求：

实现直接插入、冒泡、直接选择、快速、归并等排序算法。具体实现要求：

1. 实现直接插入排序算法。
2. 实现冒泡排序算法。
3. 实现直接选择排序算法。
4. 实现快速排序算法。
5. 实现归并排序算法。

通过产生多组随机数据，比较多种排序方法的优劣，掌握多种排序方法的使用。掌握简单排序方法中数据元素的比较次数和移动次数。

### 二、程序设计的基本思想，原理和算法描述：

（包括程序的结构，数据结构，输入/输出设计，符号名说明等）

1、插入排序：依次将待排序的序列中的每一个记录插入到先前排序好的序列中，直到全部记录排序完毕。

2、冒泡排序：两两比较相邻记录的关键码，如果反序则交换，直到没有反序记录为止。

4、快速排序：首先选择一个基准，将记录分割为两部分，左支小于或等于基准，右支则大于基准，然后对两部分重复上述过程，直至整个序列排序完成。

5、选择排序：从待排序的记录序列中选择关键码最小（或最大）的记录并将它与序列中的第一个记录交换位置；然后从不包括第一个位置上的记录序列中选择关键码最小（或最大）的记录并将它与序列中的第二个记录交换位置；如此重复，直到序列中只剩下一个记录为止。

7、归并排序：将若干个有序序列两两归并，直至所有待排序的记录都在一个有序序列为止。

### 三、源程序及注释：

1. 快速排序程序：

/\*------------------------------快速排序-------------------------------\*/

int Partition(Sqlist L,int low,int high) /\*一趟快速排序程序\*/

{ int pivotkey;

L[0].key=L[low].key;

pivotkey=L[low].key;

while(low<high)

{ while(low<high && L[high].key>=pivotkey)

--high;

L[low].key=L[high].key;

while(low<high && L[low].key <=pivotkey)

++low;

L[high].key=L[low].key ;

}

L[low].key=L[0].key;

return low;

}

void Qsort(Sqlist L,int low,int high) /\*对顺序表L中的子序列L.r[low..high]作快速排序\*/

{ int pivotloc;

if(low<high)

{pivotloc=Partition(L,low,high);

Qsort(L,low,pivotloc-1);

Qsort(L,pivotloc+1,high);

}

}

1. 冒泡法排序：

/\*-----------------------------冒泡排序---------------------------------\*/

int maopao(Sqlist R)

{ int i,j,m;

int exchange;

input(R);

printf("请输入要输出第几趟结果：");

scanf("\n%d",&m);

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

{/\*最多做n-1趟排序\*/

exchange=FALSE;

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

{

if (R[j+1].key<R[j].key)

{

R[0]=R[j+1];

R[j+1]=R[j];

R[j]=R[0];

exchange=TRUE;

}

}

if((i==m)||(!exchange))

{

printf("第%d趟的结果是：",m);

print(R);

printf("\n");

printf("请输入还要输出第几趟结果，不想输出时请输入：");

scanf("\n%d",&m);

}

if(!exchange)

return(1);

}

printf("最终排序结果是：");

print(R);

printf("\n");

getch();

}

1. 直接插入排序:

/\*-----------------------直接插入排序----------------------------------\*/

void InsertSort(Sqlist R)

{ int i,j;

for(i=2;i<=n;i++)

{ if(LT(R[i].key,R[i-1].key))

{R[0]=R[i];

R[i]=R[i-1];

for(j=i-2;LT(R[0].key,R[j].key);j--)

R[j+1]=R[j];

R[j+1]=R[0];

}

printf("\n第%d趟的排序结果为:",i-1);

print(R);

}

}

1. 选择排序：

/\*----------------------------选择排序--------------------------\*/

int SelectMinkey(Sqlist R,int i)

{ int k,j;

KeyType temp;

temp=R[i].key;

k=i;

for(j=n;j>=i;j--)

if(R[j].key<temp)

{temp=R[j].key;

k=j;

}

return k;

}

void SelectSort(Sqlist R)

{ int i,j;

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

{ j=SelectMinkey(R,i);

if(i!=j)

{ R[0]=R[i];

R[i]=R[j];

R[j]=R[0];

}

printf("\n第%d趟的排序结果为:",i);

print(R);

}

}

1. 主函数：

/\*―――――――---――――-―――main函数――――――----――――――――\*/

void main() /\*开始主函数\*/

{ int i,dk[]={5,3,1};

for(;;) /\*开始for循环\*/

{ switch (menubar())

{ case 1:printf("\n欢迎使用冒泡排序\n");

maopao(R);

system("CLS");

break;

case 2:printf("\n欢迎使用快速排序\n");

input(R);

Qsort(R,1,n);

printf("\n 最终的排序结果为:");

print(R);

getch();

system("CLS");

break;

case 3:printf("\n欢迎使用希尔排序\n");

input(R);

for(i=0;i<3;i++)

{ShellInsert(R,dk[i]);

printf("\n第%d趟排序的结果为:",i+1);

print(R);}

printf("\n 最终的排序结果为:");

print(R);

getch();

system("CLS");

break;

case 4:printf("\n欢迎使用直接插入排序\n");

input(R);

InsertSort(R);

printf("\n 最终的排序结果为:");

print(R);

getch();

system("CLS");

break;

case 5:printf("\n欢迎使用选择排序\n");

input(R);

SelectSort(R);

printf("\n 最终的排序结果为:");

print(R);

getch();

system("CLS");

break;

case 6:printf("\n欢迎使用堆排序\n");

input(R);

HeapSort(R);

printf("\n 最终的排序结果为:");

print(R);

getch();

system("CLS");

break;

case 0:system("CLS");

printf("\n谢谢您的使用\n\n\n\n");

getch();

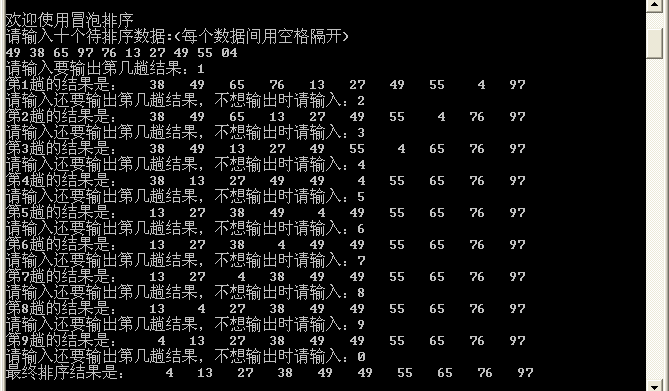
exit(0); /\*退出for循环\*/

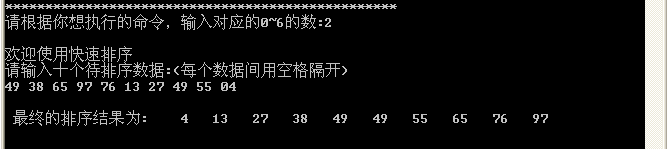
}

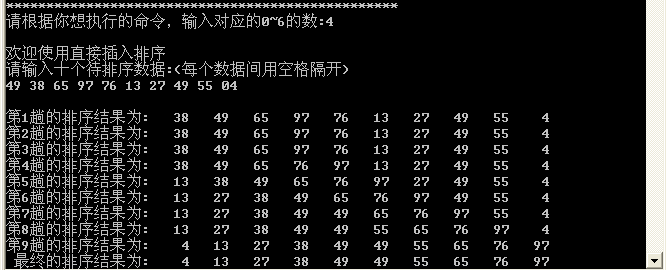
}

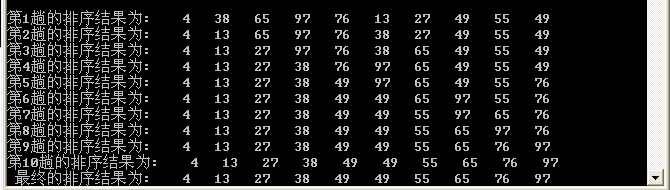
}

### 四、运行输出结果：









### 五、调试和运行程序过程中产生的问题及采取的措施：

几种排序的算法都是采用书上和老师讲的算法，所以操作起来没有太大的问题。

### 六、对算法的程序的讨论、分析，改进设想，其它经验教训：

排序是计算机程序设计中的一种重要操作，它的功能是将一个数据元素（或记录）的任意序列，重新排列成一个按关键字有序的序列。

在具体操作中对这学期所学的数据结构和以前学的Ｃ语言的理论知识得到巩固，也发现自己的不足之处，在以后的上机中应更加注意，通过实际操作，学会数据结构书中程序的编写方法和基本步骤，开发了自己的逻辑思维能力，培养了分析问题、解决问题的能力。

### 七、对实验方式、组织、设备、题目的意见和建议：

对排序算法的实验方式，可以提供一组数据让学生使用各个方法排序，输出各个步骤的过程和结果，这样不仅可以知道结果对不对，还可以知道过程对不对。