中 北 大 学

课程设计说明书

|  |  |  |  |
| --- | --- | --- | --- |
| **学 院、系：** | 软件学院 | | |
| **专 业：** | 软件工程 | | |
| **班 级：** | 14140x04 | | |
| **学 生 姓 名：** | 杨佩钰 | **学 号：** | 1414011319 |
| **设 计 题 目：** | 排序综合 | | |
|  |  | | |
| **起 迄 日 期:** | 2016年1月6日**~**2016年1月20日 | | |
| **指 导 教 师:** | 李玉荣 | | |

日期: 2016年1月20日

**1 设计目的**

1. 了解并掌握数据结构与算法的设计方法，具备初步的独立分析和设计能力；
2. 初步掌握软件开发过程的问题分析、系统设计、程序编码、测试等基本方法和技能；
3. 提高综合运用所学的理论知识和方法独立分析和解决问题的能力；
4. 训练用系统的观点和软件开发一般规范进行软件开发，培养软件工作者所应具备的科学的工作方法和作风。

**2 任务概述**

1. 至少采用三种方法实现上述问题求解（提示，可采用的方法有插入排序、希尔排序、起泡排序、快速排序、选择排序、堆排序、归并排序）。并把排序后的结果保存在不同的文件中；
2. 统计每一种排序方法的性能（以上机运行程序所花费的时间为准进行对比），找出其中两种较快的方法；
3. 如果采用4种或4种以上的方法者，可适当加分。

设计要求：

(1) 符合课题要求，实现相应功能；

(2) 要求界面友好美观，操作方便易行；

(3) 注意程序的实用性、安全性；

|  |
| --- |
| 主函数 |

**3 模块划分**

|  |
| --- |
| CreateList函数 |

|  |
| --- |
| Traverse函数 |

|  |
| --- |
| Find函数 |

|  |
| --- |
| Sort\_3函数 |

|  |
| --- |
| Sort\_2函数 |

|  |
| --- |
| Sort\_1函数 |

CreateList函数:创建序列

|  |
| --- |
| FindPos函数 |

|  |
| --- |
| Findheigh函数 |

Traverse函数：遍历序列

Sort\_1函数：选择排序 Sort\_2函数：起泡排序

Sort\_3函数：快速排序 Find函数：返回每个数据

Findheigh函数：返回序列最后数据的位置

FindPos函数：实现快速排序具体功能的函数

**4 主要函数说明及其N-S图**

定义变量L,m,n,I,flag,fp,a[200]。

输出("\n\n");

输出(" 欢迎进入排序综合系统！\n");

输出("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

输出(" 1.创建一个序列\n");

输出(" 2.查看序列\n");

输出(" 3.选择排序\n");

输出(" 4.起泡排序\n");

输出(" 5.快速排序\n");

输出(" 6.退出程序\n");

输出("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

输出("请根据您的需要输入要执行操作的序号：");

输入m的值

M

1 2 3 4 5 6

输出"请输入您 输出"您的序 调用sort\_1 调用sort\_2 调用sort\_3 输出

的序列长度：" 列为：\n 函数 函数 函数 谢谢

输入 n 调用 输出 输出 输出 使用，

调用 traverse函数 "对您的序列 "对您的序列 "对您的序列 return

CreateList函数 进行选择排 进行起泡排 进行快速排 false

输出"此操作结输出"此操作 后结果如下:” 后结果如下:” 后结果如下:”

束，请进行其 结束，请进行 调用 调用 调用

他操作！" 其他操作！” traverse函数 traverse函数 traverse函数

输出"\n\n" 输出"\n\n" 程序1 程序2 程序3

输出"此操作 输出"此操作 输出"此操作

结束，请进行结束，请进行结束，请进行

其他操作！” 其他操作！” 其他操作！”

输出"\n\n" 输出"\n\n" 输出"\n\n"

flag = true;

**图4.1 主函数N-S图**

|  |
| --- |
| (fp=fopen("G:\\选择排序.txt","w+"))==NULL  T F  输出 for (i=0; i<n; i++)  "打开文件失败!" a[i] = Find(&L, i);  fprintf(fp, "%d ", a[i]);    fclose(fp); |

**图4.2 程序1 N-S图**

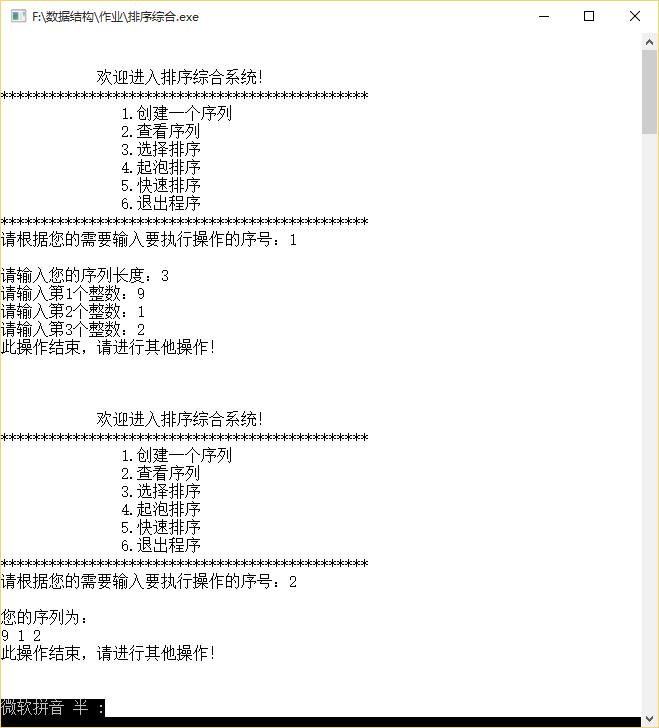
|  |
| --- |
| (fp=fopen("G:\\选择排序.txt","w+"))==NULL  T F  输出 for (i=0; i<n; i++)  "打开文件失败!" a[i] = Find(&L, i);  fprintf(fp, "%d ", a[i]);    fclose(fp); |

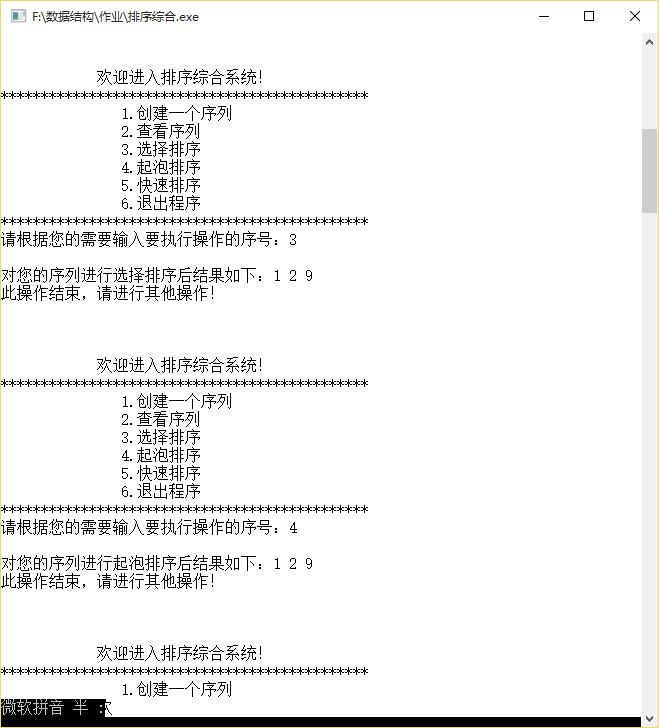
**图4.3程序2 N-S图**

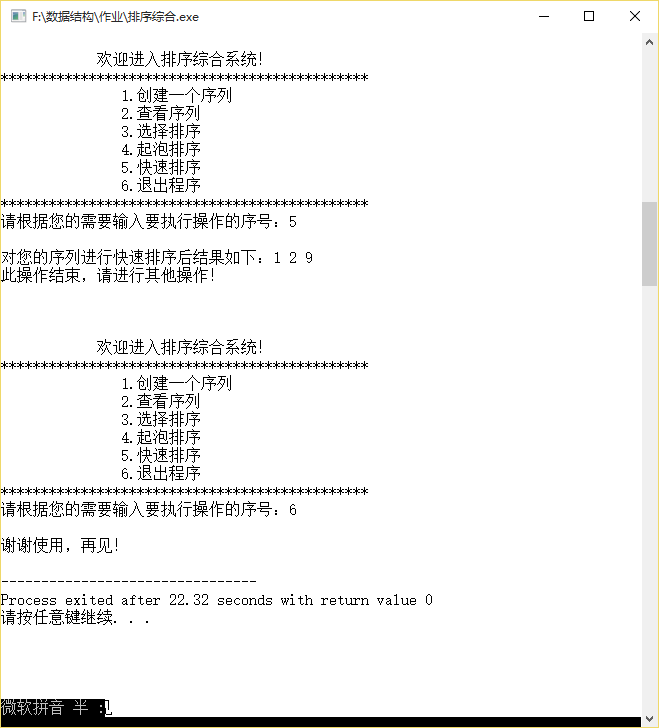
|  |
| --- |
| (fp=fopen("G:\\选择排序.txt","w+"))==NULL  T F  输出 for (i=0; i<n; i++)  "打开文件失败!" a[i] = Find(&L, i);  fprintf(fp, "%d ", a[i]);    fclose(fp); |

**图4.4 程序3 N-S图**

**5 程序运行数据及其结果**







**6 课程设计心得**

通过本次对数据结构课程设计的实际操作，我们学会了很多东西，我的题目是“排序综合”，刚刚看到这个题目，我还比较高兴，因为这是书上讲过的内容，应该比较好写，但是当我真正实际下手去做的时候，我发现这并不容易，虽然我们平时都学习过这些相应的算法，但是没有自己真正动手去写过一个程序，所以操作起来还是有相当难度的，而其中一些看似简单的内容真正实践起来也并不是那么容易，比如说最简单的创建一个序列和遍历序列，这些都得通过指针去做形参，才能实现对数据结构成员的值的改写，而不是我们想象的那么简单。

操作的主要内容是对排序算法的操作，这些算法在平时讲课的时候，我们都基本能听懂，但是我们平时没有认真去巩固，复习，所以实践的时候也不是那么容易就懂了这些算法，更不要说能轻易写出来了，所以，我们必须在平时经常的巩固以前学过的一些知识，并且去动手实践，包括以前学的一个写入文件的函数“fprintf”,这个函数我就在网上查了半天。所以，通过这次的课程设计的试验，我们不仅学到了很多技术上的知识，而且学到了很多学习的方法，总体来说，还是有很大收获的。

**附录：**

#include <stdio.h>

#include <malloc.h>

#include <stdlib.h>// exit头文件

#include <stdbool.h>// bool头文件

#include <time.h>// 检测运行时间的clock函数的头文件

typedef int Status;

typedef struct List

{

int \* pBase;// 第一个元素地址 ,必须用指针，因为只有指针能跨函数调用

int cnt;// 有效元素的数量

}SqList;

void CreateList(SqList \* L, int n)//L也必须是指针

{

int i;

int val;

L->pBase = (int \*)malloc(sizeof(int) \* n);

if (NULL == L->pBase)

{

printf("分配内存失败，程序终止！");

exit(-1);

}

L->cnt = 0;

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

{

printf("请输入第%d个整数：", i);

scanf("%d", &val);

L->pBase[L->cnt] = val;

(L->cnt)++;

}

}

Status traverse(SqList \* L)

{

int i;

for (i=0; i<L->cnt; i++)

{

printf("%d ", L->pBase[i]);

}

printf("\n");

}

Status Find(SqList \* L, int i)// 用来返回输出时的值，保存文件时会用到

{

return L->pBase[i];

}

void sort\_1(SqList \* L)// 选择排序 4.2s

{

int i, j, t;

for (i=0; i<(L->cnt)-1; i++) // 只需要比较长度减一次

{

for (j=i+1; j<L->cnt; j++)// 每次比较都是第i个和第i+1,i+2...个进行比较

{

if (L->pBase[i] > L->pBase[j])

{

t = L->pBase[i];

L->pBase[i] = L->pBase[j];

L->pBase[j] = t;

}

}

}

}

/\*void sort\_1\_1(SqList \* L)

{

int i, j, t;

int a, b;

double c;

a = clock();

for (i=0; i<(L->cnt)-1; i++)

{

for (j=i+1; j<L->cnt; j++)

{

if (L->pBase[i] > L->pBase[j])

{

t = L->pBase[i];

L->pBase[i] = L->pBase[j];

L->pBase[j] = t;

b = clock();

c = (double)(b-a)/CLOCKS\_PER\_SEC;

printf("%f\n", c);

}

}

}

// printf("快速排序的运行时间为：%f\n", c);

}\*/

void sort\_2(SqList \* L)// 起泡排序 3.558s

{

int i, j, t;

for (i=0; i<(L->cnt)-1; i++)// 比较长度减一次

{

for (j=0; j<(L->cnt)-i-1; j++) //如序列3， 2， 1， 5， 8 第一次比较3和2，3>2, 3和2互换，再比较3和1，如此类推，一次比较完后，最大的在最后

{ //第一次比较完后，比较下一次，最大的那个数就不再参与比较，所以比较的次数为 长度减一再减去第一次比较完的

if (L->pBase[j] > L->pBase[j+1])

{

t = L->pBase[j];

L->pBase[j] = L->pBase[j+1];

L->pBase[j+1] = t;

}

}

}

}

Status Findhigh(SqList \* L)// 用于返回序列的最后一个值的位置，快速排序需要

{

return L->cnt-1;

}

Status FindPos(SqList \* L, int low, int high)

{

int val = L->pBase[low];

while (low < high)

{

while (low < high && L->pBase[high]>=val)

high--;

L->pBase[low] = L->pBase[high];

while (low < high && L->pBase[low]<=val)

low++;

L->pBase[high] = L->pBase[low];

}

L->pBase[low] = val;

return low;

}

void sort\_3(SqList \* L, int low, int high)// 快速排序 3.453s

{

int pos;

if (low < high)

{

pos = FindPos(L, low, high);

sort\_3(L, low, pos-1);//pos的前一半

sort\_3(L, pos+1, high);//pos的后一半

}

}

int main(void)

{

SqList L;

int n, m;

int i;

//int start, finish;

bool flag = true;

FILE \* fp;

char a[200];

while (flag = true)

{

printf("\n\n");

printf(" 欢迎进入排序综合系统！\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf(" 1.创建一个序列\n");

printf(" 2.查看序列\n");

printf(" 3.选择排序\n");

printf(" 4.起泡排序\n");

printf(" 5.快速排序\n");

//printf(" 6.检测各排序方法运行时间\n");

printf(" 6.退出程序\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("请根据您的需要输入要执行操作的序号：");

scanf("%d", &m);

printf("\n");

switch (m)

{

case 1:printf("请输入您的序列长度：");

scanf("%d", &n);

CreateList(&L, n);

printf("此操作结束，请进行其他操作！");

printf("\n\n");

break;

case 2:printf("您的序列为：\n");

traverse(&L);

printf("此操作结束，请进行其他操作！");

printf("\n\n");

break;

case 3:sort\_1(&L);

printf("对您的序列进行选择排序后结果如下：");

traverse(&L);

if((fp=fopen("G:\\选择排序.txt","w+"))==NULL) // fopen函数为打开文件的函数 fopen("文件路径","对文件的操作")

{

printf("打开文件失败!");

getch();

}

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

{

a[i] = Find(&L, i);

fprintf(fp, "%d ", a[i]); //将a[i]保存到文件中 , 要有空格，与下一数据隔开

}

fclose(fp); // 关闭文件

printf("此操作结束，请进行其他操作！");

printf("\n\n");

break;

case 4:sort\_2(&L);

printf("对您的序列进行起泡排序后结果如下：");

traverse(&L);

if((fp=fopen("G:\\起泡排序.txt","w+"))==NULL)

{

printf("打开文件失败!");

getch();

}

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

{

a[i] = Find(&L, i);

fprintf(fp, "%d ", a[i]);

}

fclose(fp);

printf("此操作结束，请进行其他操作！");

printf("\n\n");

break;

case 5:sort\_3(&L, 0, Findhigh(&L));

printf("对您的序列进行快速排序后结果如下：");

traverse(&L);

if((fp=fopen("G:\\快速排序.txt","w+"))==NULL)

{

printf("打开文件失败!");

getch();

}

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

{

a[i] = Find(&L, i);

fprintf(fp, "%d ", a[i]);

}

fclose(fp);

printf("此操作结束，请进行其他操作！");

printf("\n\n");

break;

//case 6:sort\_1\_1(&L);

// break;

case 6:printf("谢谢使用，再见！\n");

return false;

break;

default:printf("您的输入有误，请重新输入！\n");

}

}

}