**信息科学与工程学院**

**2020－2021学年第一学期**

实 验 报 告

课程名称： 数据结构与数据库技术

实验名称： 实验四

专 业 班 级 通信工程 二班

学 生 学 号 201800121050

学 生 姓 名 孟麟芝

实 验 时 间 2020年10月8日

实验报告

## 【实验目的】

1.掌握常用的排序方法，并掌握C语言实现排序算法的方法

2.了解各种排序方法过程及依据原则，并掌握各种排序方法的时间复杂度和稳定性的分析方法。

## 【实验要求】

给出n个学生的考试成绩表，每条信息由姓名和分数组成，利用排序算法完成以下任务：

（1）按照分数高低次序，打印出每个学生在考试中获得的名次，分数相同的为同一名次。

（2）按照名次列出每个学生的姓名和分数。

（3）要求学生的考试成绩需要从键盘输入数据建立，同时要设计输出格式。

## 【实验代码】

1.主程序main.c代码

#include <stdio.h>

#include <stdlib.h>

#include "SortStudents.h"

void main()

{

    int count;

    printf("请输入一共有多少个学生：\n");

    scanf("%d", &count);

    students student[200];

    getIndex(count, student);

    useShellSort(count, student);

    showIndex(count, student);

    system("PAUSE");

}

2.库文件SortStudents.c代码

#include "SortStudents.h"

struct StudentCard

{

    char Name[100];

    int Grade;

};

void useBubbleSort(int count, students student[])

{

    //下面进行冒泡排序

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

    {

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

        {

            if (student[j]->Grade < student[j + 1]->Grade)

            {

                students temp;

                temp = student[j];

                student[j] = student[j + 1];

                student[j + 1] = temp;

            }

        }

    }

}

void useShellSort(int count, students student[])

{

    //这里使用的增量是Shell建议的序列

    for (int Increment = count / 2; Increment > 0; Increment /= 2)

        for (int i = Increment; i < count; i++)

        {

            int j;

            students temp;

            temp = student[i];

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

                if (temp->Grade > student[j - Increment]->Grade)

                    student[j] = student[j - Increment];

                else

                    break;

            student[j] = temp;

        }

}

void getIndex(int count, students student[])

{

    for (int i = 0; i < count; i++)

    {

        //进行初始化

        student[i] = (students)malloc(sizeof(struct StudentCard));

        printf("请输入第%d个学生的姓名\n", i + 1);

        scanf("%s", student[i]->Name);

        printf("请输入第%d个学生的成绩\n", i + 1);

        scanf("%d", &(student[i]->Grade));

    }

}

void showIndex(int count, students student[])

{

    //用于显示输出名次，同分数的同学名次一致

    int rank = 1;

    for (int i = 0; i < count; i++)

    {

        if (!i)

        {

            printf("第%d名：%s，成绩：%d分\n", rank++, student[i]->Name, student[i]->Grade);

        }

        else

        {

            if (student[i]->Grade == student[i - 1]->Grade)

            {

                printf("       %s，成绩：%d分\n", student[i]->Name, student[i]->Grade);

            }

            else

            {

                printf("第%d名：%s，成绩：%d分\n", rank, student[i]->Name, student[i]->Grade);

            }

            rank++;

        }

    }

}

3.头文件SortStudents.h代码

typedef struct StudentCard \*students;

void useBubbleSort(int count, students studentsIndex[]);

void useShellSort(int count, students studentsIndex[]);

void getIndex(int count, students studentsIndex[]);

void showIndex(int count, students studentsIndex[]);

## 【实验记录与说明分析】

本次实验主要是围绕下面这个结构体展开的：

1. **struct** StudentCard
2. {
3. **char** Name[100];
4. **int** Grade;
5. };

它有两个成员，一个是姓名，一个是成绩，为了保证中文名或者较长的名字也可以存下，Name这个数组设置的比较长，另外要注意的是编程时不可以使用UTF-8编码，不然scanf读入中文时会有错误。

实验中有两种排序方式：冒泡排序和希尔排序，它们的代码如下：

1. **void** useBubbleSort(**int** count, students student[])
2. {
3. //下面进行冒泡排序
4. **for** (**int** i = 0; i < count - 1; i++)
5. {
6. **for** (**int** j = 0; j < count - i - 1; j++)
7. {
8. **if** (student[j]->Grade < student[j + 1]->Grade)
9. {
10. students temp;
11. temp = student[j];
12. student[j] = student[j + 1];
13. student[j + 1] = temp;
14. }
15. }
16. }
17. }
18. **void** useShellSort(**int** count, students student[])
19. {
20. //这里使用的增量是Shell建议的序列
21. **for** (**int** Increment = count / 2; Increment > 0; Increment /= 2)
22. **for** (**int** i = Increment; i < count; i++)
23. {
24. **int** j;
25. students temp;
26. temp = student[i];
27. **for** (j = i; j >= Increment; j -= Increment)
28. **if** (temp->Grade > student[j - Increment]->Grade)
29. student[j] = student[j - Increment];
30. **else**
31. **break**;
32. student[j] = temp;
33. }
34. }

下面来看一下程序的运行方式。

首先让用户输入要排序的学生总数，存入count，然后要建立一个结构体StudentCard类型的指针数组，用于实现对多个学生信息的存放检索。

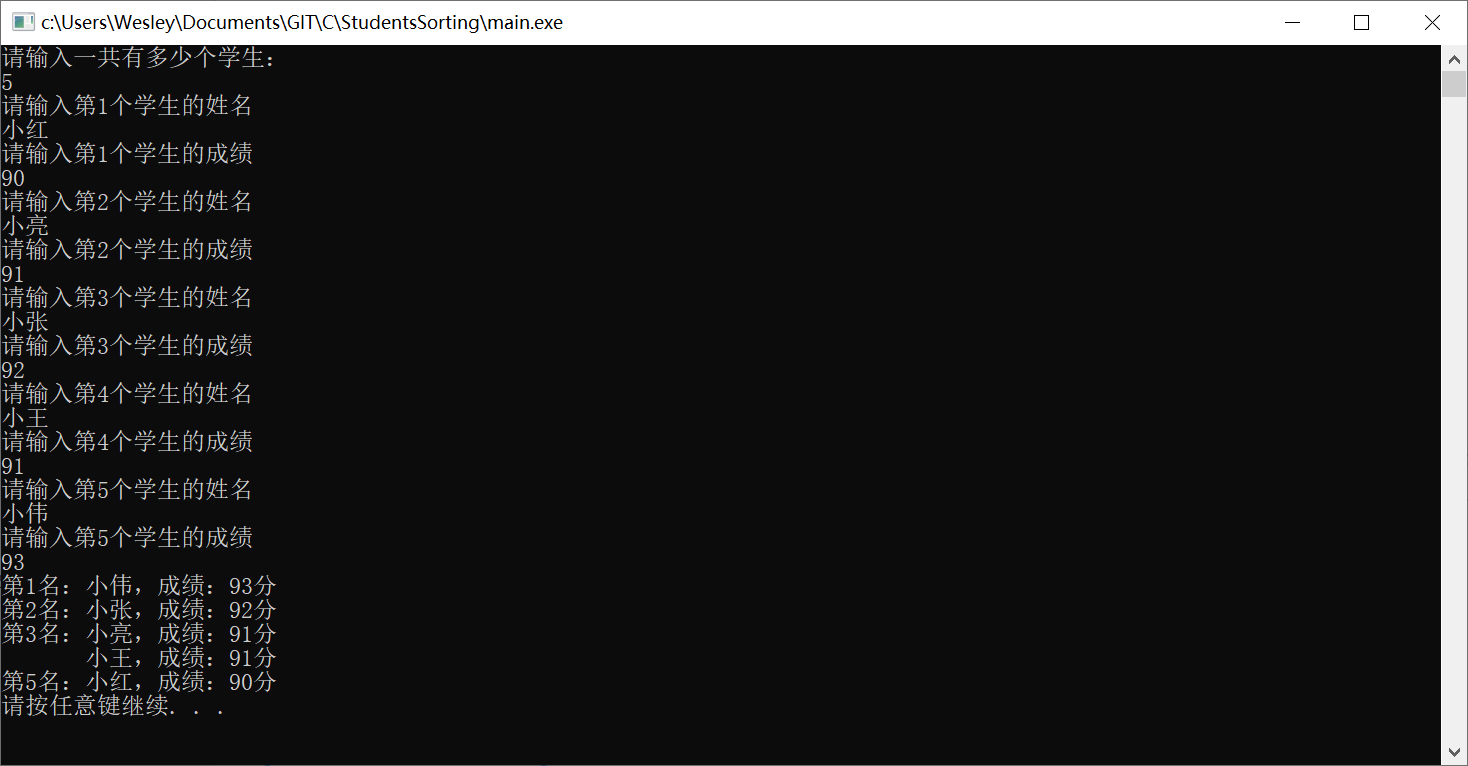
然后，调用getIndex()函数，利用malloc函数进行空间分配，并且将学生信息存入。

再调用useSellSort()或者useBubbleSort()函数，实现对学生的排序，由于形参是指针，所以在函数内部进行操作时也对实参指向的对象产生了影响。

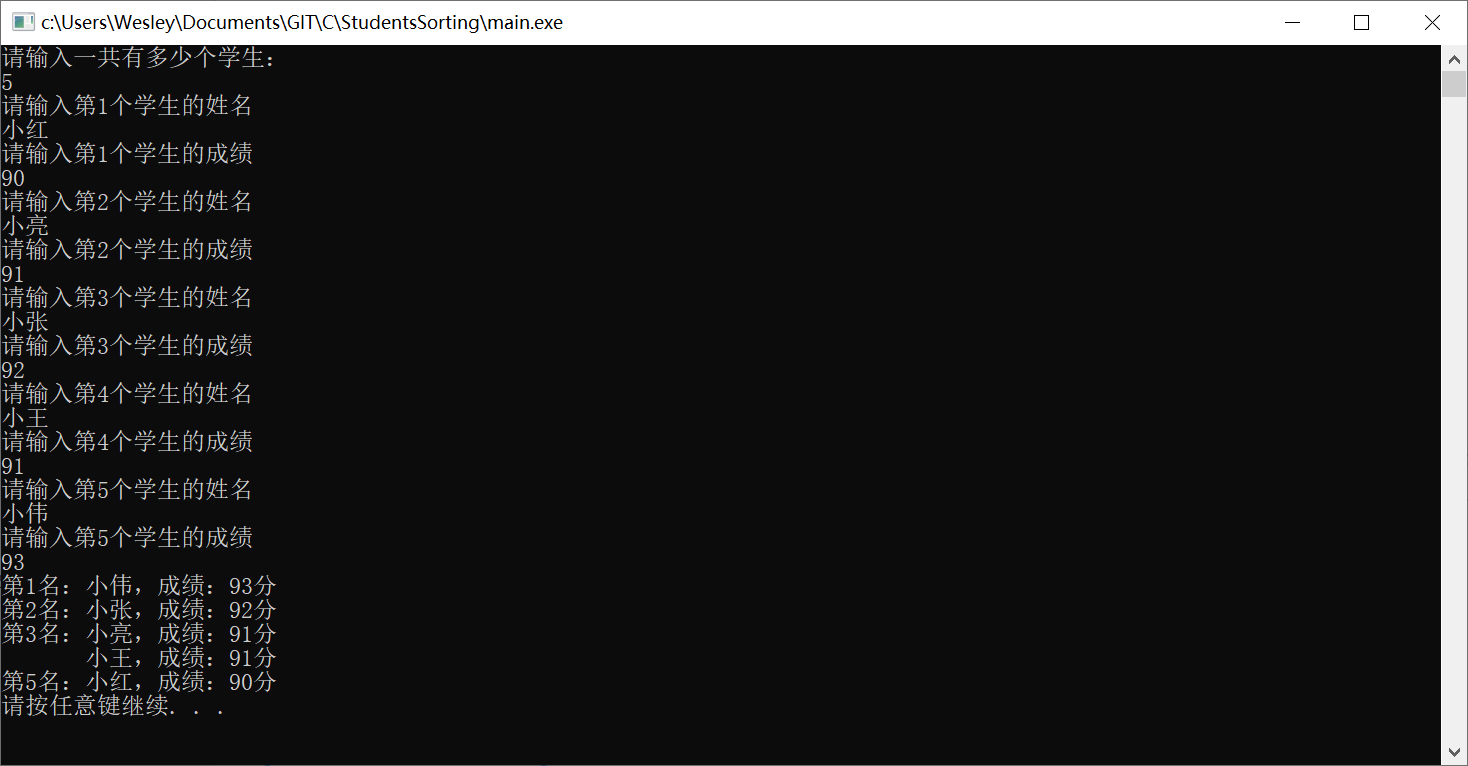
最后，调用showIndex()函数进行显示，显示时考虑了同分的情况。

1. **void** main()
2. {
3. **int** count;
4. printf("请输入一共有多少个学生：\n");
5. scanf("%d", &count);
6. students student[200];
7. getIndex(count, student);
8. useShellSort(count, student);
9. showIndex(count, student);
10. system("PAUSE");
11. }

使用希尔排序时，其运行结果如下：



使用冒泡排序时，其运行结果如下：



可见两种排序方法的结果是一样的，只是在时间上有区别，希尔排序的最坏情况与冒泡法的时间复杂度是大体一致的，都是，在其他情况下则要更快一些。