

**数据结构课程设计**

**题目：** 学生成绩处理系统

**专业班级：** 计算机科学与技术 6班

**学 号：** 20204331

**姓 名：** 王博文

**指导教师：** 隋毅

**报告日期：** 2022年4月10日

计算机科学技术学院

**2022年4月10日**

一、课程设计内容

**设计要求：**本设计要求采用顺序存储结构，实现关于学生成绩处理的相关问题，包括：学生信息的录入、查询、修改、排序、确定名次以及信息的输出。其中学生信息包括：学号、姓名、四门课、总分以及排名；排序方法包括：双向冒泡排序、希尔排序、快速排序、堆排序等方法。

菜单内容：

1. **学生信息录入**
2. **学生信息查询**
3. **学生信息修改**
4. **学生成绩排序**
5. **学生成绩输出**
6. **结束程序**

二、算法设计

本次的代码使用的是结构体数组的形式，基本都是基础应用，跟第一题会有点类似，主要的问题其实就是排序代码的编写。

双向冒泡排序在根本上跟冒泡其实是一样的只不过是一轮走两次顺序依次，逆序一次。

希尔排序则是要选择d每次比较i与i+d的大小，随着d的减小，最后的数组会趋于有序。

快速排序则是要寻找基点，将大于他和小于他的数据分列两侧，然后两侧在进行快速排序。

堆排序我用的我自己实验三写的代码，在这个题上稍稍加以修正。

三、核心代码实现

#include<iostream>

#include<stdlib.h>

using namespace std;

int flag=1;

struct Node {

int num;

string name;

int mathg;

int englishg;

int peg;

int musicg;

int totalg;

};

void insert(Node \*head) {

cout<<"依次输入学号、姓名、数学成绩、英语成绩、体育成绩、音乐成绩："<<endl;

cout<<"输入0时停止"<<endl;

int a;

string b;

int c,d,e,f;

while(1) {

cin>>a;

cin>>b>>c>>d>>e>>f;

if(a==0)

break;

head[flag].num=a;

head[flag].name=b;

head[flag].mathg=c;

head[flag].englishg=d;

head[flag].peg=e;

head[flag].musicg=f;

head[flag].totalg=c+d+e+f;

flag++;

}

}

void search(Node \*head) {

cout<<"请输入想要查找的学号"<<endl;

int a;

cin>>a;

int i;

for(i=0; i<flag; i++) {

if(head[i].num==a)

break;

}

if(i!=flag) {

cout<<head[i].num<<" "<<head[i].name<<" "<<head[i].mathg<<" "<<head[i].englishg<<" ";

cout<<head[i].peg<<" "<<head[i].musicg<<" "<<head[i].totalg<<endl;

} else cout<<"查找失败"<<endl;

}

void correct(Node \*head) {

cout<<"请输入想要修改的学号"<<endl;

int a;

cin>>a;

int i;

for(i=0; i<flag; i++) {

if(head[i].num==a)

break;

}

if(i!=flag) {

cout<<"请依次输入更改后的数据"<<endl;

string b;

int c,d,e,f;

cin>>b>>c>>d>>e>>f;

head[i].name=b;

head[i].mathg=c;

head[i].englishg=d;

head[i].peg=e;

head[i].musicg=f;

head[i].totalg=c+d+e+f;

cout<<"更改成功"<<endl;

} else cout<<"查无此人"<<endl;

}

void bidbubblesort(Node \*head) {

int left, right, shift, i;

left=1;

right=flag ;

shift=1;

while(left<right) {

for(i=left;i< right;i++) {

if(head[i].totalg>head[i+1].totalg) {

Node t;

t=head[i];

head[i]=head[i+1];

head[i+1]=t;

shift=i;

}

}

right=shift;

for(i=right-1;i>=left;i--) {

if(head[i].totalg>head[i+1].totalg) {

Node t;

t=head[i];

head[i]=head[i+1];

head[i+1]=t;

shift=i+1;

}

}

left=shift;

}

cout<<"双向冒泡排序结果如下："<<endl;

for(i=flag; i>1; i--) {

cout<<head[i].num<<" "<<head[i].name<<" "<<head[i].totalg<<" "<<endl;

}

}

void shellsort(Node \*head) {

int d,i,j;

Node t;

for(d=flag/2; d>=1; d=d/2) {

for(i=d; i<flag; i++) {

t=head[i];

for(j=i-d; j>=0&&t.totalg<head[j].totalg; j=j-d) {

head[j+d]=head[j];

}

head[j+d]=t;

}

}

cout<<"希尔排序结果如下："<<endl;

for(i=flag; i>1; i--) {

cout<<head[i].num<<" "<<head[i].name<<" "<<head[i].totalg<<" "<<endl;

}

}

void quicksort(Node \*head,int begin,int end) {

if(begin > end)

return;

Node t=head[begin];

int i=begin;

int j=end;

while(i!=j) {

while(head[j].totalg>=t.totalg&&j>i)

j--;

while(head[i].totalg<=t.totalg&&j>i)

i++;

if(j > i) {

Node t1;

t1=head[i];

head[i]=head[j];

head[j]=t1;

}

}

head[begin]=head[i];

head[i]=t;

quicksort(head,begin,i-1);

quicksort(head,i+1,end);

}

void swap(Node \*q,Node \*p) {

Node t;

t=\*q;

\*q=\*p;

\*p=t;

}

void heapsort(Node \*head) {

int i=flag;

int max;

int m=0;

int k=flag;

while(m!=flag) {

if(i%2==0) {

max=i;

if(head[i].totalg<head[i-1].totalg)

max=i-1;

else

max=i;

if(head[i/2-1].totalg<head[max].totalg)

swap(&head[i/2-1],&head[max]);

i=i-2;

} else {

max=i;

if(head[i/2].totalg<head[max].totalg)

swap(&head[i/2],&head[max]);

i=i-1;

}

for(i; i>0; i=i-2) {

if(head[i].totalg<head[i-1].totalg)

max=i-1;

else

max=i;

if(head[i/2-1].totalg<head[max].totalg)

swap(&head[i/2-1],&head[max]);

}

swap(&head[k],&head[0]);

k--;

i=k;

m++;

}

cout<<"堆排序结果如下："<<endl;

for(i=flag; i>1; i--) {

cout<<head[i].num<<" "<<head[i].name<<" "<<head[i].totalg<<" "<<endl;

}

}

void print(Node \*head){

cout<<"输出结果如下："<<endl;

for(int i=flag; i>1; i--) {

cout<<head[i].num<<" "<<head[i].name<<" "<<head[i].mathg<<" ";

cout<<head[i].englishg<<" "<<head[i].peg<<" "<<head[i].musicg<<" ";

cout<<head[i].totalg<<" "<<flag-i+1<<endl;

}

}

int main() {

cout<<"1、学生信息录入"<<endl<<"2、学生信息查询"<<endl<<"3、学生信息修改"<<endl;

cout<<"4、学生成绩排序"<<endl<<"5、学生成绩输出"<<endl<<"6、结束程序"<<endl;

Node \*head;

head=new Node[15];

while(1) {

int z;

cin>>z;

switch(z) {

case 1:

insert(head);

break;

case 2:

search(head);

break;

case 3:

correct(head);

break;

case 4:

bidbubblesort(head);

shellsort(head);

quicksort(head,1,flag);

cout<<"快速冒泡排序结果如下："<<endl;

int i;

for(i=flag; i>1; i--) {

cout<<head[i].num<<" "<<head[i].name<<" "<<head[i].totalg<<" "<<endl;

}

heapsort(head);

break;

case 5:

print(head);

break;

case 6:

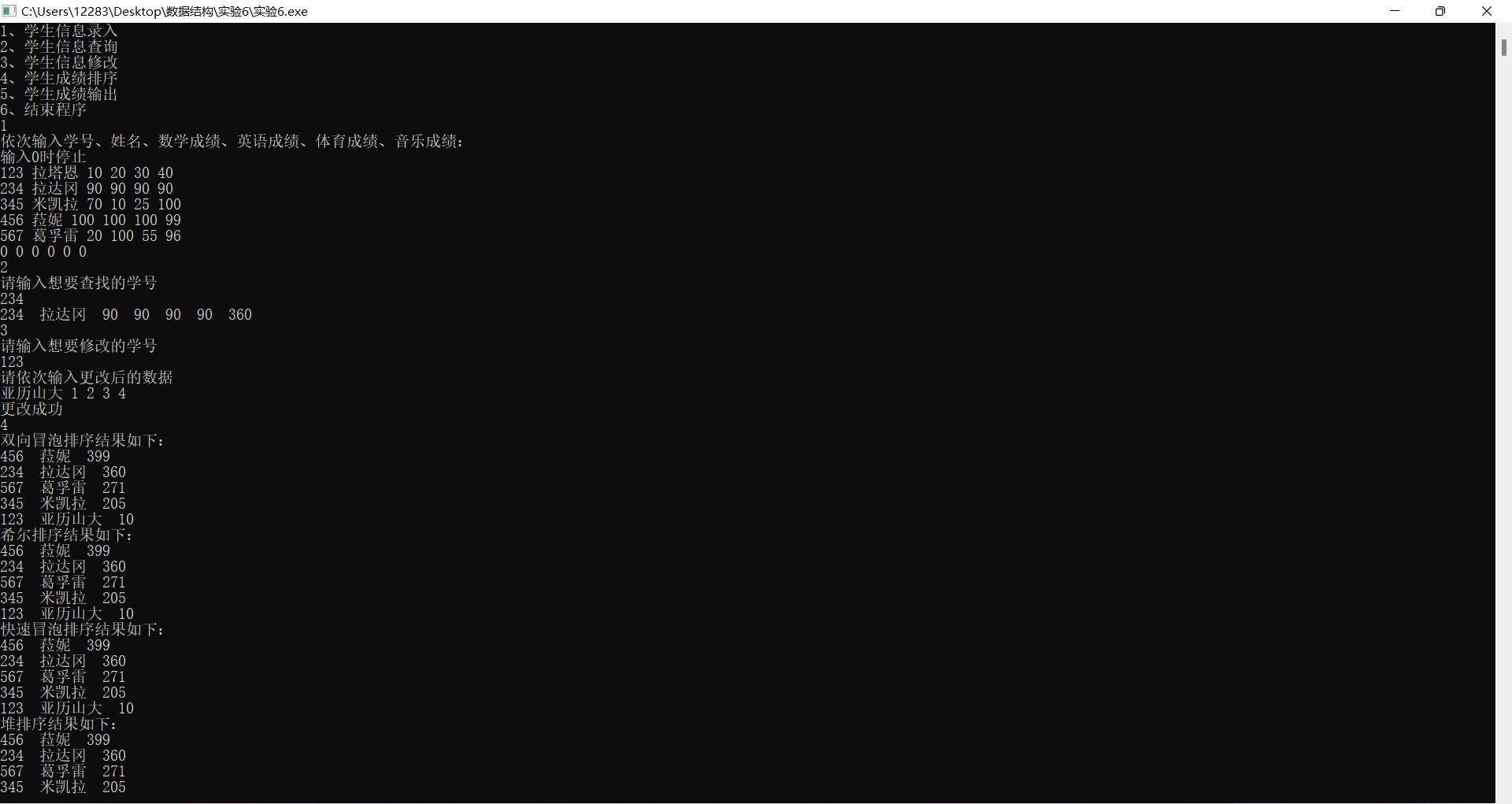
exit(0);

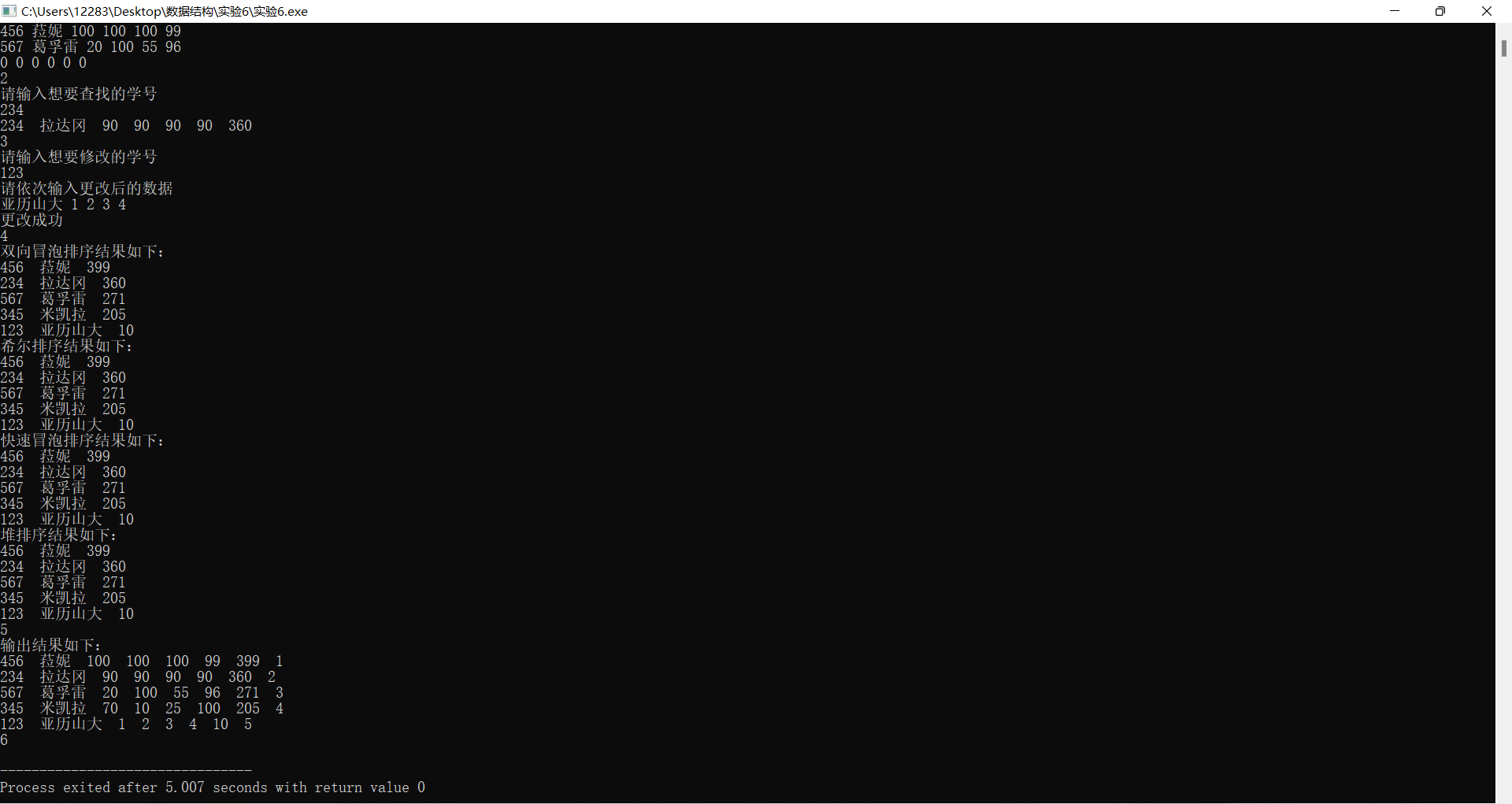
}

}

}

四、测试与分析





五、总结及体会

通过这次实验加深了我对排序算法的认识，但最开始看到题目的时候还是会愣住，因为以前学的各种排序算法虽然思路可能看一下就通，但是对于名称的记忆已经所剩无几，就好比如希尔排序，知道他的思想，但是没有把他的思想和希尔排序这几个字联系起来，也庆幸因为有了这次试验，让我对排序算法又进行了一次复习

1. 参考书目和网络资源

CSDN四种算法的思想