**实验报告**

新生入学需要对入学信息进行计算机管理，包括建立、增加、删除、修改、查询入学信息等。入学信息由姓名、性别、年龄、入学成绩等组成。已知入学信息相对保持不变，个别会进行调整。利用本章所学的数据结构相关知识编写算法和程序来解决该问题，用到的测试数据如下表所示。

表1-1 测试数据

|  |  |  |  |
| --- | --- | --- | --- |
| **姓名** | **性别** | **年龄** | **入学成绩** |
| 张峰 | 男 | 20 | 95 |
| 李志刚 | 男 | 20 | 95 |
| 王芳 | 女 | 23 | 86 |
| 刘晓丽 | 女 | 21 | 78 |
| 赵明 | 男 | 20 | 63 |

**案例分析**

新生的入学信息可组成一个入学信息线性表。根据已知条件“入学信息相对保持不变，个别会进行调整”，较宜采用顺序结构存储该线性表，算法及实现比较简单。利用顺序表的插入、删除、查询等基本操作就可实现该问题的主要需求。

**解决方案**

**1.入学信息表的存储结构类型定义**

typedef struct

{

char name[10]; /\*姓名\*/

char sex[3]; /\*性别\*/

int age; /\*年龄\*/

int score; /\*入学成绩\*/

}Student; /\*入学信息\*/

**2.主要算法**

建立入学信息顺序表：可以用手工输入的方法，逐个输入建表。然而利用这样的办法，出错的机会较多，也不利于程序调试。最好的办法是先把入学信息按一定的格式存储为文本文件，建立算法中通过读取文本文件的方法建立入学信息顺序表。为简化起见，数据文件中的数据按入学成绩从高到低排序。建立操作的C语言描述的算法如下。

void CreateList(Student \*stu) /\*读取数据文件，建立顺序表\*/

{ /\*从数据文件中读取数据，该数据按入学成绩从高到低排序，建立顺序表\*/

FILE \*fp; /\*定义文件指针\*/

int i=0;

if((fp=fopen("student.dat","r"))==NULL) /\*打开数据文件\*/

{

printf("\n数据文件无法打开！");

exit(1);

}

while(!feof(fp)) /\*数据文件未读取完毕\*/

{

i++;

fscanf(fp,"%s%s%d%d",stu[i].name,stu[i].sex,&stu[i].age,&stu[i].score);

}

stu[0].age=i; /\*利用stu[0].age存放数据个数，即表长\*/

fclose(fp); /\*关闭文件\*/

}

增加入学信息：利用有序表插入算法，按入学成绩插入，插入后仍有序。

修改入学信息：根据要修改的姓名，在顺序表中查找到相应的数据，不能直接修改，否则会破坏原有入学成绩的有序性。可以先删除该数据，然后再在有序表中插入该数据，从而达到修改的目的。

**3.完整程序**

/\*案例1 入学信息管理系统\*/

#include

#include

#include

#define MAX 100

/\*入学信息类型定义\*/

typedef struct

{

char name[10]; /\*姓名\*/

char sex[3]; /\*性别\*/

int age; /\*年龄\*/

int score; /\*入学成绩\*/

}Student;

void CreateList(Student \*stu); /\*读取数据文件中的数据，建立顺序表\*/

int MenuSelect(); /\*显示菜单并选择相应功能\*/

int InsertList(Student \*stu,Student x); /\*在有序表中插入数据\*/

int DeleteList(Student \*stu,Student x); /\*按姓名查找并删除数据\*/

int UpdateList(Student \*stu,Student x); /\*按姓名查找并修改数据\*/

int FindList(Student \*stu,Student x); /\*按姓名查找\*/

void OutputList(Student \*stu); /\*输出所有数据\*/

/\*主函数\*/

void main()

{

Student stu[MAX]; /\*入学信息数组变量\*/

Student x;

int i;

CreateList(stu); /\*读取数据文件中的数据\*/

for(;;)

{

switch(MenuSelect()) /\*选择判断\*/

{

case 1: /\*增加数据\*/

printf("\n请输入入学信息（姓名、性别、年龄、入学成绩）\n\n");,/M

scanf("%s%s%d%d",x.name,x.sex ,&x.age,&x.score);

InsertList(stu,x);

break;

case 2: /\*删除数据\*/

printf("\n请输入姓名:");,/

scanf("%s",x.name);

DeleteList(stu,x);

break;

case 3: /\*修改数据\*/

printf("\n请输入新入学信息（姓名、性别、年龄、入学成绩）\n\n");

scanf("%s%s%d%d",x.name,x.sex ,&x.age,&x.score);

UpdateList(stu,x);

break;

case 4: /\*查询数据\*/

printf("\n请输入姓名:");

scanf("%s",x.name);

i=FindList(stu,x);

if(i!=0)

{

printf("\n你查找的入学信息（姓名、性别、年龄、入学成绩）为：\n");

printf("\n%-12s %-6s %-6d %-8d",stu[i].name,stu[i].sex,stu[i].age,stu[i].score);

}

else

printf("\n没有你要查找的入学信息！\n");

break;

case 5: /\*输出数据\*/

OutputList(stu);

break;

case 0: /\*退出系统\*/

exit(0);

}

printf("\n\n");

system("pause"); /\*按任何键继续\*/

}

}

/\*从数据文件中读取数据（按入学成绩从高到低排序），建立顺序表\*/

void CreateList(Student \*stu)

{

FILE \*fp; /\*定义文件指针\*/

int i=0;

if((fp=fopen("student.dat","r"))==NULL) /\*打开数据文件\*/

{

printf("\n数据文件无法打开！");

exit(1);

}

while(!feof(fp)) /\*数据文件未读取完毕\*/

{

i++;

fscanf(fp,"%s%s%d%d",stu[i].name,stu[i].sex,&stu[i].age,&stu[i].score);

}

stu[0].age=i; /\*利用stu[0].age存放数据个数，即表长\*/

fclose(fp); /\*关闭文件\*/

}

/\*显示菜单并选择相应功能，返回选择的功能编号\*/

int MenuSelect()

{

char c;

do{

system("cls"); /\*运行前清屏\*/

printf(" 入学信息管理系统\n"); /\*菜单选择\*/

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

printf(" 1. 增加数据\n");

printf(" 2. 删除数据\n");

printf(" 3. 修改数据\n");

printf(" 4. 查询数据\n");

printf(" 5. 输出数据\n");

printf(" 0. 退出系统\n");

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

printf(" 请选择(0-5):");

c=getchar(); /\*输入选择\*/

}while(c<'0'||c>'5');

return c-'0'; /\*返回选择\*/

}

/\*在有序表中插入数据x，使插入后仍按入学成绩从高到低排序\*/

int InsertList(Student \*stu,Student x)

{

int i,j;

if(stu[0].age==MAX-1)

{

printf("\n表满，无法插入！");

return 0;

}

/\*查找插入位置\*/

for(i=1; i<=stu[0].age; i++)

if(x.score>stu[i].score) break;

/\*顺序表插入操作\*/

for(j=stu[0].age; j>=i; j--)

stu[j+1]=stu[j];

stu[i]=x;

stu[0].age++;

return 1;

}

/\*按姓名查找并删除数据\*/

int DeleteList(Student \*stu,Student x)

{

int i,j;

if(stu[0].age==0)

{

printf("\n空表，无法删除！");

return 0;

}

i=FindList(stu,x); /\*按姓名查找\*/

if(i==0)

{

printf("\n没找到！"); /\*返回失败信息\*/

return 0;

}

/\*顺序表删除操作\*/

for(j=i+1;j<=stu[0].age;j++)

stu[j-1]=stu[j];

stu[0].age--;

return 1;

}

/\*按姓名查找并修改数据，修改后仍需按入学成绩降序排列\*/

int UpdateList(Student \*stu,Student x)

{

int i;

i=FindList(stu,x); /\*按姓名查找\*/

if(i==0)

{

printf("\n你输入的姓名不对，修改失败！\n"); /\*返回失败信息\*/

return 0;

}

DeleteList(stu,x); /\*先删除该数据\*/

InsertList(stu,x); /\*然后将该数据插入有序表中\*/

return 1;

}

/\*按姓名查找，查找成功返回数组下标，否则返回0 \*/

int FindList(Student \*stu,Student x)

{

int i;

for(i=1; i<=stu[0].age; i++)

if(strcmp(stu[i].name,x.name)==0)

return i;

return 0;

}

/\*按格式输出所有数据\*/

void OutputList(Student \*stu)

{

int i;

printf("\n------------------------------------"); /\*格式头\*/

printf("\n姓名 性别 年龄 入学成绩");

printf("\n------------------------------------");

for(i=1;i<=stu[0].age;i++) /\*循环输出\*/

printf("\n%-10s %-8s %-8d %-8d",stu[i].name,stu[i].sex,stu[i].age,stu[i].score);

}

**运行结果**

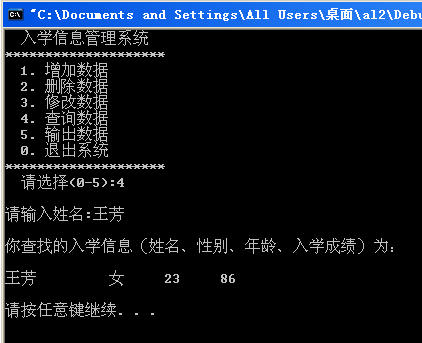
1. 选择输出数据功能时

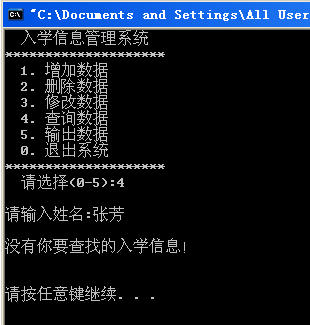
程序运行结果如下：



2. 选择查询数据功能时

分别输入王芳和张芳，运行结果如下：

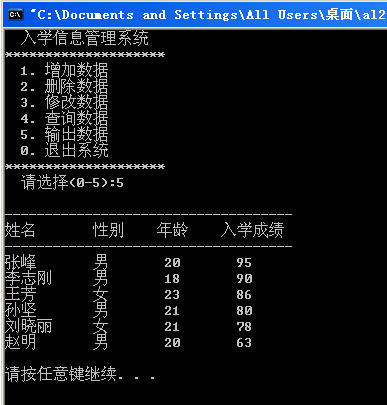




3. 选择增加数据功能时

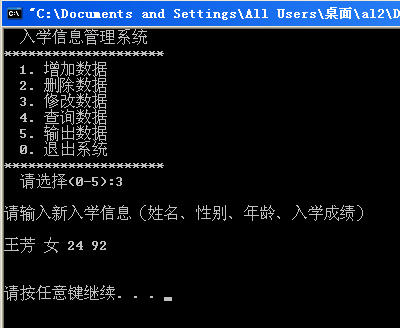
输入入学信息：孙坚 男 21 80，再选择输出数据功能时运行结果如下：

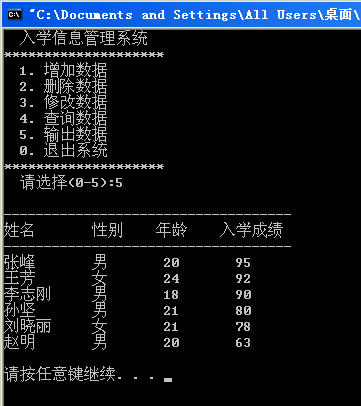




4. 选择修改数据功能时

输入新入学信息：王芳 女 24 92，再选择输出数据功能时运行结果如下：

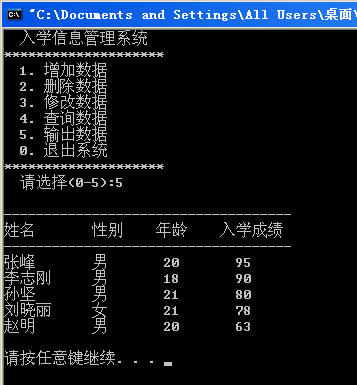




5. 选择删除数据功能时

输入王芳，再选择输出数据功能时运行结果如下：





**结论分析与讨论**

本案例使用的算法大多是基本算法，如插入、删除、查询等算法和程序与知识点中介绍的完全一样，而如修改等操作，只是运用已有的基本算法实现而已，程序实现只是针对具体的数据类型和要求做了很少的修改。其他功能，也都可以利用基本操作很方便地实现。通过对入学信息管理系统案例的学习，更深入地理解顺序表的知识和应用。