

**课 程 设 计 报 告**

**题目： 排课评估与分析助手**

**课程名称： 数据结构课程设计**

**专业班级： 计算机（校交）1601**

**学 号： U201610489**

**姓 名： 陈国轩**

**指导教师： 周全**

**报告日期： 2018年3月22日**

**计算机科学与技术学院**

# 任务书

**设计目的**

掌握图、查找表等的存储结构与基本算法，通过解决较复杂的基于图等模型的实际问题，提高学生对数据结构知识综合运用的技能与实践能力。并且可应用到对排课的合理性与科学性评价，提高教学管理水平，更好地服务教师与学生。

**设计内容**

设计有效的逻辑数据结构与存储结构表示课表信息，教师信息，课程信息，班级信息，学生信息，教室信息等，由时间、教室、课程名、班级、教师名、上课周期及课堂ID等构成课堂信息。根据课堂及其复杂相关性，依据合理的逻辑与准则评价排课课表的教师安排合理性、课程安排合理性、班级与学生课程学习合理性、教室安排合理性与利用率、能效情况等。本设计只对现有排课进行评估，不去求解最优排课方案。

**主要数据对象**

教师；教室；课程；班级；学生；课堂等。

**主要数据关系**

（1）教师之间作为普通同事可以是一般查找表集合中元素间的关系；也可以根据上下级、部门、课程组等因素构成层次关系。教室、班级、学生可以视为普通查找表。课程之间由于先修与后继课之间具有较复杂的关系。

（2）课堂之间由于教师、教室、班级、上课时间、课程等因素的作用具有十分复杂的关系。

**主要运算与功能要求**

（1）交互式操作界面(并非一定指图形式界面)；

（2）教师、教室、班级、课程等的增加、删除、修改、查找与检索等；

（3）课堂的增加、删除、修改、查找与检索，包括按教师检索其任课课堂，班级检索课堂，按教室、课程、时间等检索课堂；

（4）空闲教室查找与检索、教室利用率分析、能效分析；

（5）教师承担课堂合理性分析；

（6）班级参与课堂的合理性分析；

（7）主要数据对象的数据文件组织与存储。

**参考文献**

[1] 严蔚敏, 吴伟民. 数据结构（C语言版）. 北京: 清华大学出版社,1997

[2] 严蔚敏, 吴伟民, 米宁. 数据结构题集（C语言版）. 北京: 清华大学出版社,1999

[3] Mark Allen Weiss.Data Structures and Algorithm Analysis in C, 机械工业出版社，2010, 177-192

**目录**

[**任务书** I](#_Toc509494631)

[**1引言** - 1 -](#_Toc509494632)

[1.1课题背景与意义 - 1 -](#_Toc509494633)

[1.2课程设计的主要研究工作 - 2 -](#_Toc509494634)

[**2系统需求分析与总体设计** - 3 -](#_Toc509494635)

[2.1系统需求分析 - 3 -](#_Toc509494636)

[2.2系统总体设计 - 3 -](#_Toc509494637)

[**3系统详细设计** - 5 -](#_Toc509494638)

[3.1有关数据结构的定义 - 5 -](#_Toc509494639)

[3.2主要算法设计(给出部分复杂流程图) - 12 -](#_Toc509494640)

[3.2.1 int LessonInsert(int placei,int timei,LESSON\* temp) - 12 -](#_Toc509494641)

[3.2.2 int ToDeleteLesson(void) - 14 -](#_Toc509494642)

[3.2.3 int ToModifyLesson(void) - 16 -](#_Toc509494643)

[3.2.4 int ToInsertTeacher(void) - 18 -](#_Toc509494644)

[3.2.5 int ToDeleteTeacher(void) - 20 -](#_Toc509494645)

[3.2.6 int ToModifyTeacher(void) - 22 -](#_Toc509494646)

[3.2.7 int ToQueryCourseByTeacher(void) - 24 -](#_Toc509494647)

[3.2.8 int ToQueryLessonByClassno(void) - 25 -](#_Toc509494648)

[3.2.9 int ToQueryLessonByTimeAndPlace(void) - 27 -](#_Toc509494649)

[3.2.10 int ToQueryLessonByCourse(void) - 29 -](#_Toc509494650)

[3.2.11 int ToQuerySpareClassroom(void) - 31 -](#_Toc509494651)

[3.2.12 int ToStatClassroomRate(void) - 32 -](#_Toc509494652)

[3.2.13 int ToStatTeacherRate(void) - 34 -](#_Toc509494653)

[3.2.14 int ToStatClassnoRate(void) - 36 -](#_Toc509494654)

[**4系统实现与测试** - 38 -](#_Toc509494655)

[4.1系统实现 - 38 -](#_Toc509494656)

[4.1.1系统实现环境 - 38 -](#_Toc509494657)

[4.1.2系统实现函数 - 39 -](#_Toc509494658)

[4.2系统测试 - 45 -](#_Toc509494659)

[4.2.1 初始化界面 - 45 -](#_Toc509494660)

[4.2.2 模块测试 - 46 -](#_Toc509494661)

[**5总结与展望** - 100 -](#_Toc509494662)

[**6体会** - 102 -](#_Toc509494663)

[**参考文献** - 103 -](#_Toc509494664)

[**附录** - 104 -](#_Toc509494665)

# 1引言

## 1.1课题背景与意义

数据结构是计算机学院学生的必修课，也是最最重要的课程之一。数据结构不像编程语言那样，每种语言有着自己的语法规则和代码风格，数据结构通过比较抽象通用的方式对数据的储存和处理进行描述，总结出一套比较通用的方案，也就是说，无论你学的是什么编程语言，都能使用数据结构总结出的思想和方法。

而图也是数据结构中最常用也最最复杂的结构之一。图是一种N对N的数据结构，表示着N个数据元素对其他数据元素的关系。实现图的方法有很多，常用的有邻接矩阵，邻接表。邻接矩阵算是一种穷举法，它把每个元素对其他所有元素的关系都记录在了矩阵之中，好处是便于查找，坏处也是很显然——就是占用空间比较大。相反邻接表比较节约空间，但是查找起来没有邻接矩阵那么方便。于是为了解决这个问题，人们便引出了压缩矩阵这种数据结构。压缩矩阵实际上就是一种十字链表，用它来处理稀疏矩阵要比二维数组要节约很多很多的空间，同时又不失二维数组在查询方面的方便特性。因此本实验采用这种数据结构来储存课程信息。

本课程设计是排课评估与分析助手，主要是对学校的课程信息进行增加、删除、修改，以及综合统计评估。其中课程信息包含课程名，任课教师，上课时间，上课地点，上课班级，学分等内容。数据的增加要考虑上课冲突，也就是同一时间同一个教室不能有2门课同时上，同一个老师也不能同一时间在两个教室上课，同一个班级也是如此。因此可以用压缩矩阵的数据结构来储存这样的课程。时间一个轴，空间一个轴，同一个班级上的课堂用课堂链表穿起来，同一个课程的课堂用一个课程链表穿起来……这样就构成了一个复杂的图结构，里面有着课堂的时间空间关系，还有课程关系，上课班级的关系……这样的设计，既帮助我们掌握图、查找表等的存储结构与基本算法，又通过解决较复杂的基于图等模型的实际问题，提高我们对数据结构知识综合运用的技能与实践能力。并且可应用到对排课的合理性与科学性评价，提高教学管理水平，更好地服务教师与学生。

## 1.2课程设计的主要研究工作

首先要写出针对课程信息的插入函数int LessonInsert(int placei,int timei,LESSON\* temp)，用来在对应的时间和空间上插入课堂。然后是相对应的查找链的插入函数，比如按班级查找的插入函数int ClassnoInsert(LESSON\* temp)，还有按课程查找的插入函数int CourseInsert(LESSON\* temp)。这样就基本实现了信息的软插入功能。但是仅有这几个函数还不行，因为还没有读入数据，因此需要有读入数据的函数，比如说int ReadLesson(FILE \*fp)，从fp所指的文件中读入数据，当需要从控制台读入数据的时候，就直接用stdin作这个函数的实参。然后就是框架函数，就是给这个应用程序写一个方便的前端。需要有总的菜单函数，键盘互动函数，子菜单弹出弹回函数等。之后就可以试运行本程序了，试运行之前还要有初始化函数int Init(void)来给内存中的压缩矩阵的表头进行预处理等等一系列预处理。这样系统就可以读入数据了。但是读入数据之后我们还不能查看这些数据也不能更改这些数据，于是我们就需要有删除函数int ToDeleteLesson(void)以及修改函数int ToModifyLesson(void)等操作。能够修改之后，我们需要读取类的查询函数来查看我们读入的数据是否正确，比如按时间和空间查找课程的函数int ToFindGenInfoByCourse(char s[])。能够修改课程信息，能够查询课程信息之后，我们便要使应用程序能够对信息进行一些统计分析，就比如说int ToStatClassroomRate(void)教室利用率分析，int ToStatClassnoRate(void)班级上课分析等等统计分析。以上思路基本上涵盖了本课设的主要研究工作，但是还不够全面，因为还需要数据文件的保存，恢复功能，这样才使得本程序的功能得以完整，符合一般的功能需求。

# 2系统需求分析与总体设计

## 2.1系统需求分析

本课程设计的需求是“排课评估与分析助手”，也就是整个程序面向的对象就是课程以及相关信息。而对于课程来说最终要的信息就是，什么时间? 在哪？

上什么课？同样地，在系统检查课程文件的排课正确性的时候需要判断最多的也是“当前正在读入到内存的课堂是否与内存中已经存在的课程有了时间和空间上的冲突”，简单的说，是不是同一时间、地点已经有课堂了，如果有了，那么数据文件中的排课方案就有排课冲突，系统需要报错，终止运行，然后让用户修改排课方案。

然后基于正确的已有的排课数据，才能对数据进行进一步的处理。即使要插入、修改、删除数据元素，也要有已经正确排好的课程信息作为基础。比如要插入一节课，插入的时候，系统就要自动检测当前要插入的课是否与内存中已经存在的课程有冲突，如果有就需要告知用户，同时提示用户插入失败。由此可见，对课程的时间、空间管理极其重要，因为它直接影响着排课的正确性，和数据修改的可执行性。

## 2.2系统总体设计

本排课分析系统的主要数据是用压缩矩阵来存储的，压缩矩阵的横轴代表着时间链，纵轴代表空间链。其中时间链有不少于840个时间链结点，代表着各个上课时间（一个学期最多有20周，一周有7天，一天最多有6大节课）。空间链有260个结点，代表着260个不同的教室。这样的压缩矩阵中的每个结点代表着在对应时间和空间上课的课堂信息，同时矩阵中的每个结点不仅用时间和空间链穿起来，还被他们共同的信息（比如上课班级，课程等）穿起来形成链表，形成复杂的网络结构。

在查询的时候，如按时间查询，就在对应的时间链下方索引需要的信息，如果按照空间查询，就在需要的空间链下方索引需要的信息，如果按照班级、课程等查询，也相应的沿着对应的链来查询对应的信息。

对于该包含多条查找链的压缩矩阵的插入、删除、修改操作，参考对应的链表的操作，区别在于一般的链表是在链表的头部或尾部插入，这里插入和删除的时候，要首先找到要插入删除的数据元素的正确的位置，然后再执行操作。具体数据结构见下图：

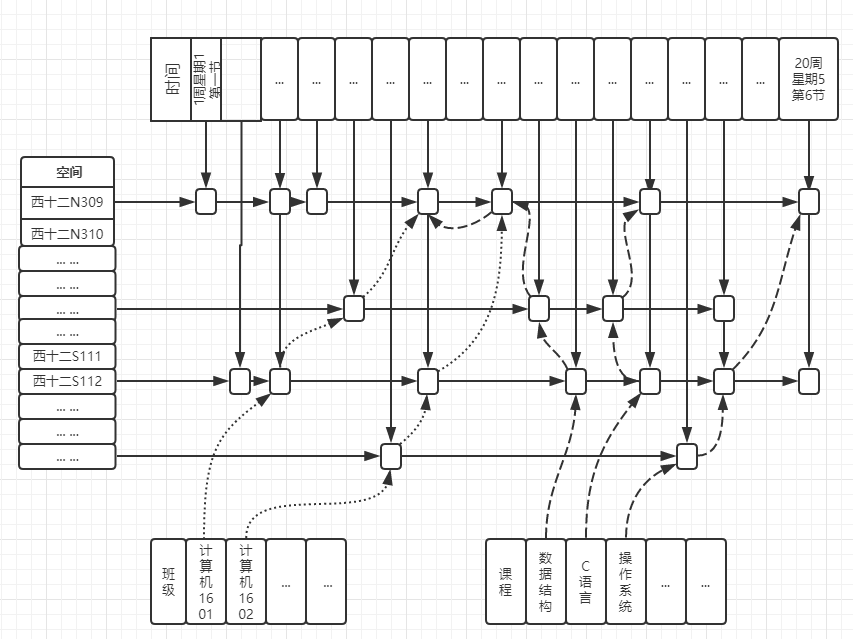


图2-1 本程序的数据结构示意图

由图可见，本程序采用的课堂信息的储存结构实际上可以看做是一个“多维压缩矩阵”，可以从不同的维度来确定一系列相应的课堂，既便于查找，同时又做到了压缩储存，节约了空间，同时便于计算排课冲突。

# 3系统详细设计

## 3.1有关数据结构的定义

主要的数据结构类型定义：

整个系统的课堂信息用一个“多维压缩矩阵”来存储， 而与课程相关的信息(包括课程的学分、学时等与课表排课无关的信息)用一个动态数组vector来存储。具体数据元素的结构，以及全局变量的定义如下：

/\*\*

时间表头节点

\*/

typedef struct time {

int week; /\*\*< 周数\*/

int day; /\*\*<星期\*/

int period; /\*\*< 节数，一天一共6大节，因此取值为1~6\*/

struct lesson \*time\_next; /\*\*< 指向同一时间不同教室的所有的课程\*/

} TIME;

/\*\*

空间表头节点

\*/

typedef struct place {

char building[15]; /\*\*< 教学楼\*/

char classroom[10]; /\*\*< 教室\*/

struct lesson \*room\_next; /\*\*< 指向同一间教室不同时间的所有的课堂\*/

} PLACE;

/\*\*

课堂结构

\*/

typedef struct lesson {

int from,to; /\*\*<从第几周到第几周\*/

int week; /\*\*< 周数\*/

int day; /\*\*<星期\*/

int period; /\*\*< 节数，一天一共6大节，因此取值为1~6\*/

char building[15];

char classroom[6]; /\*\*< 教室\*/

char classno[150]; /\*\*< 专业班级\*/

char course[80]; /\*\*< 课程名\*/

int No; /\*\*< 在内存中的编号\*/

struct lesson \*time\_next; /\*\*< 指向同一间教室不同时间的下一个课堂\*/

struct lesson \*room\_next; /\*\*<指向同一时间不同教室的下一个课堂\*/

struct lesson \*time\_pri; /\*\*< 指向同一间教室不同时间的上一个课堂\*/

struct lesson \*room\_pri; /\*\*<指向同一时间不同教室的上一个课堂\*/

struct lesson \*classno\_next; /\*\*< 指向同一班级的下一个课程\*/

struct lesson \*classno\_pri; /\*\*< 指向同一班级的上一个课程\*/

struct lesson \*course\_next; /\*\*< 指向同一课程的下一个课堂\*/

struct lesson \*course\_pri; /\*\*< 指向同一课程的上一个课堂\*/

struct time \*lesson\_time ; /\*\*<指向时间表头\*/

struct place \*lesson\_place ;/\*\*<指向空间表头\*/

} LESSON;

/\*\*

与课程相关的综合信息结构

\*/

typedef struct info {

char course[100]; /\*\*<课程名\*/

char time[10]; /\*\*<学时\*/

char credit[5]; /\*\*<学分\*/

char title[10]; /\*\*<教师职称\*/

char teacher[100];/\*\*<教师姓名\*/

char classno[150]; /\*\*<专业班级\*/

char extra[81]; /\*\*<备注\*/

} INFO;

/\*\*

班级查找链表头数据结构

\*/

typedef struct classno {

char classno[150]; /\*\*< 专业班级\*/

LESSON\* lesson; /\*\*<指向课堂的指针\*/

} CLASSNO;

/\*\*

课程查找链的结构

\*/

typedef struct course {

char course[80]; /\*\*< 课程名\*/

LESSON\* lesson; /\*\*<指向课堂的指针\*/

} COURSE;

TIME les\_time[845]; //时间表头

PLACE les\_place[260];//空间表头

bool readed[845\*260];//在查询函数中标记此课堂是否已读

std::vector<INFO> gen\_info;//储存课程综合信息

std::vector<CLASSNO> les\_classno;//专业班级查找链

std::vector<COURSE> les\_course;//课程查找链

std::vector<int>find;//删除函数中用来储存遍历过程中要删除的数据元素的下标的vector

int lesson\_sum=0;//当前课程总数

CONSOLE\_CURSOR\_INFO lpCur; //存储光标信息

**以下是图形界面框架中需要用到的结构体以及全局变量定义**：

/\*\*

\*屏幕窗口信息链结点结点结构

\*/

typedef struct layer\_node {

char LayerNo; /\*\*< 弹出窗口层数\*/

SMALL\_RECT rcArea; /\*\*< 弹出窗口区域坐标\*/

CHAR\_INFO \*pContent; /\*\*< 弹出窗口区域字符单元原信息存储缓冲区\*/

char \*pScrAtt; /\*\*< 弹出窗口区域字符单元原属性值存储缓冲区\*/

struct layer\_node \*next; /\*\*< 指向下一结点的指针\*/

} LAYER\_NODE;

/\*\*

\*标签束结构

\*/

typedef struct labe1\_bundle {

char \*\*ppLabel; /\*\*< 标签字符串数组首地址\*/

COORD \*pLoc; /\*\*< 标签定位数组首地址\*/

int num; /\*\*< 标签个数\*/

} LABEL\_BUNDLE;

/\*\*

\*热区结构

\*/

typedef struct hot\_area {

SMALL\_RECT \*pArea; /\*\*< 热区定位数组首地址\*/

char \*pSort; /\*\*< 热区类别(按键、文本框、选项框)数组首地址\*/

char \*pTag; /\*\*< 热区序号数组首地址\*/

int num; /\*\*< 热区个数\*/

} HOT\_AREA;

LAYER\_NODE \*gp\_top\_layer = NULL; /\*弹出窗口信息链链头\*/

char \*gp\_sys\_name = "排课评估与分析助手"; /\*系统名称\*/

char \*ga\_main\_menu[] = {"文件(F)", /\*系统主菜单名\*/

"数据维护(M)",

"数据查询(Q)",

"数据统计(S)",

"帮助(H)"

};

char \*ga\_sub\_menu[] = {"[S] 数据保存", /\*系统子菜单名\*/

"[B] 数据备份",

"[R] 数据恢复",

"[X] 退出 Alt+X",

"[S] 课堂信息",

"[T] 课程信息",

"", /\*空串用来在弹出菜单中分隔子菜单项，下同\*/

"[D] 教师信息",

"[P] 教室信息",

"[C] 班级信息",

"[S] 按教师检索课程",

"[T] 按班级检索课堂",

"",

"[D] 按教室、时间查询课堂",

"[P] 按课程检索课堂",

"[C] 空闲教室查询",

"[I] 教室利用与能效分析",

"[T] 教师任课分析",

"[C] 班级上课分析",

"[U] 遍历与检索",

"[T] 帮助主题",

"",

"[A] 关于..."

};

int ga\_sub\_menu\_count[] = {4, 6, 6, 4, 3}; /\*各主菜单项下子菜单的个数\*/

int gi\_sel\_menu = 1; /\*被选中的主菜单项号,初始为1\*/

int gi\_sel\_sub\_menu = 0; /\*被选中的子菜单项号,初始为0,表示未选中\*/

CHAR\_INFO \*gp\_buff\_menubar\_info = NULL; /\*存放菜单条屏幕区字符信息的缓冲区\*/

CHAR\_INFO \*gp\_buff\_stateBar\_info = NULL; /\*存放状态条屏幕区字符信息的缓冲区\*/

char \*gp\_scr\_att = NULL; /\*存放屏幕上字符单元属性值的缓冲区\*/

char gc\_sys\_state = '\0'; /\*用来保存系统状态的字符\*/

HANDLE gh\_std\_out; /\*标准输出设备句柄\*/

HANDLE gh\_std\_in; /\*标准输入设备句柄\*/

## 3.2主要算法设计(给出部分复杂流程图)

### 3.2.1 int LessonInsert(int placei,int timei,LESSON\* temp)

/\*\*

\* 函数名称: LessonInsert

\* 函数功能: 在对应的时间空间插入课堂

\* 输入参数: 空间hash值，时间hash值，要插入的课堂结构体指针

\* 返 回 值: 成功返回1，出错返回-1

\*

\*/

设计思路：

（1）根据时间hash值，在对应的教室的上课时间链上查找对应时间的课堂结点。

（2）如果找到了同一教室时间为timei的结点，那么说明同一教室同一时间排了两门课，上课冲突，报错。否者执行（3）。

（3）在对应的时间链上插入要插入的结点temp。

（4）根据空间hash值，在对应的上课时间下面查找要对应上课教室的课堂结点。

（5）如果找到了同一时间上课地点为placei的结点，那么说明同一时间同一教室排了两门课，上课冲突，报错。否者执行（6）。

（6）在对应的空间链上插入要插入的结点temp。

（7）结束

时间复杂度：O(n+m)。n表示同一时间的不同教室的课堂数，m为同一地点不同时间的课堂数。

空间复杂度：O(1)。不需要辅助空间

流程图：

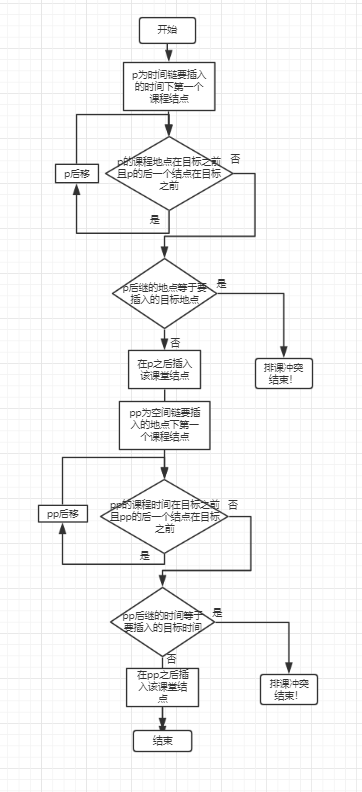


图3-1 插入课堂的算法流程

### 3.2.2 int ToDeleteLesson(void)

/\*\*

\* 函数名称: ToDeleteLesson

\* 函数功能: 删除对应时间空间的课程

\* 输入参数: 无

\* 返 回 值: 成功返回1，出错返回-1

\*

\*/

设计思路：

1. 给用户提示信息，提示用户要输入要删除的课堂的时间和空间的正确输入格式（周次，星期，节数，教学楼，教室等），然后读入用户输入的信息。
2. 如果用户输入信息的格式正确则执行（3），否者提示报错，用户可以选择重试。
3. 计算出用户输入的要删除的课堂的时间和空间的hash值
4. 然后在对应的教室下面寻找对应时间的课堂，如果没找到，提示用户该课堂不存在，报错，结束，否者执行（5）
5. 将该课堂的时间后继接到时间前驱后面。
6. 将该课堂的空间后继接到空间前驱后面。
7. 将该课堂的课程查找链后继接到课程查找链前驱后面。
8. 将该课堂的班级查找链后继接到班级查找链前驱后面。
9. 释放该课堂的空间。
10. 结束。

时间复杂度：O(n)。n为同一地点不同时间的课堂数。

空间复杂度：O(1)。不需要辅助空间

流程图：



图3-2 删除课堂的算法流程

### 3.2.3 int ToModifyLesson(void)

/\*\*

\* 函数名称: ToModifyLesson

\* 函数功能: 修改课堂信息

\* 输入参数: 无

\* 返 回 值: 成功返回1，出错返回-1

\*/

设计思路：

1. 在屏幕上输出数据文件的储存格式信息，提供用户修改的参考
2. 利用系统的API函数，调用资源管理器，打开数据文件，提供用户修改
3. 用户修改之后，重新载入数据，如果修改后有排课冲突，报错，否者完成。

时间复杂度：O(n)。n数据元素个数。

空间复杂度：O(n)。重载需要正比于n的空间，但是原来的空间也被释放，不会浪费空间。

流程图：

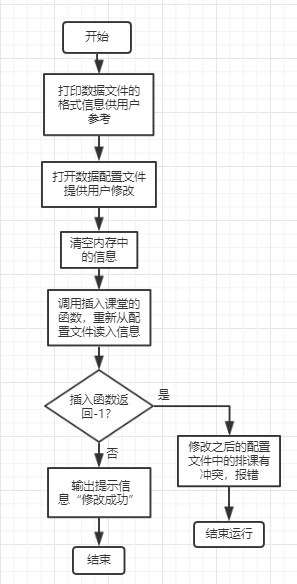


图3-3 修改课堂的算法流程

### 3.2.4 int ToInsertTeacher(void)

/\*\*

\* 函数名称: ToInsertTeacher

\* 函数功能: 通过教师名插入综合信息

\* 输入参数: 无

\* 返 回 值: 1

\*/

设计思路：

1. 输出提示信息，也就是用户所要输入的数据项，包括教师的姓名，职称，任课情况等，以及提示用户如何处理缺省项等等信息
2. 读入用户输入的信息，如果用户输入的信息的格式有问题，那么重新输入，否者执行（3）
3. 将用户输入的信息（一个数据元素）直接插入到综合课程信息的vector数组中即可完成。

时间复杂度：O(n)。n为数据元素个数。

空间复杂度：O(1)。不需要辅助空间

流程图：

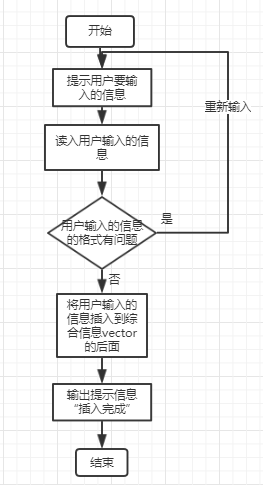


图3-4 插入教师信息的算法流程

### 3.2.5 int ToDeleteTeacher(void)

/\*\*

\* 函数名称: ToDeleteTeacher

\* 函数功能: 删除教师及其相关信息

\* 输入参数: 无

\* 返 回 值: 1

\*/

设计思路：  
 （1）读入用户输入的要删除的教师名称。

（2）遍历整个综合信息链，当找到某个结点的教师的姓名包含用户输入的姓名的时候，将该元素的下标插入到动态数组find的后面。

（3）如果动态数组find为空，那么说明没有该教师的相关信息，结束。否者执行（4）

（4）输出find数组中每一个整数作为下标的数据元素中的所有信息，用来提示用户，输出的时候，从1开始标号输出，便于查看。

（5）输出提示信息“是否删除该教师及其所有相关的信息”，如果是，执行（6），否者直接结束。

（6）i从0开始，每次删除j=find[i]-i所指的数据元素。因为find是从前往后遍历的，每删除一个，后边待删除数据元素的角标(无需删除的元素也是)集体向前移动了1,总的就是前移i。

（7）删除完之后，清空find动态数组。

（8）提示用户删除完成，结束。

时间复杂度：O(n)。n为数据元素个数。

空间复杂度：O(1)。不需要辅助空间

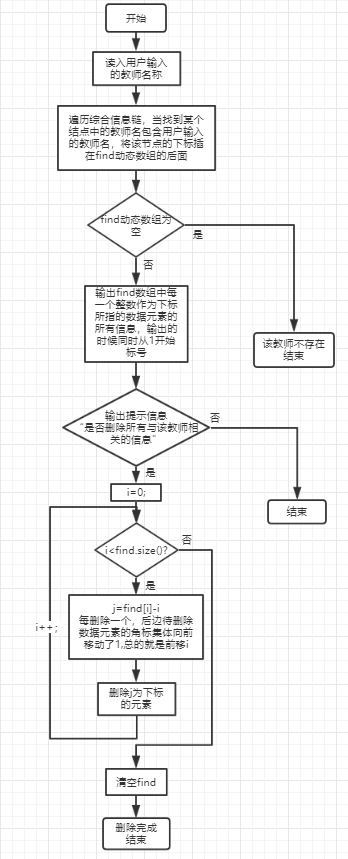


图3-5 删除教师信息的算法流程

### 3.2.6 int ToModifyTeacher(void)

/\*\*

\* 函数名称: ToModifyTeacher

\* 函数功能: 修改教师相关信息

\* 输入参数: 无

\* 返 回 值: 1

\*

\*/

设计思路：

（1）读入用户输入的要修改的教师名称。

（2）遍历整个综合信息链，当找到某个结点的教师的姓名包含用户输入的姓名的时候，将该元素的下标插入到动态数组find的后面。

（3）如果动态数组find为空，那么说明没有该教师的相关信息，结束。否者执行（4）

（4）输出find数组中每一个整数作为下标的数据元素中的所有信息，用来提示用户，输出的时候，从1开始标号输出，便于查看。

（5）提示用户输入要修改教师信息的编号no，并读入。

（6）如果用户输入的编号小于0或者大于find.size()，标号输入错误，重新输入，否者执行（7）。

（7）int j=find[no]; 提示用户输入新的信息，并录入到综合信息链下标为j的数据元素中 。

（8）清空find动态数组。

（9）结束。

时间复杂度：O(n)。n为数据元素个数。

空间复杂度：O(1)。不需要辅助空间

流程图：

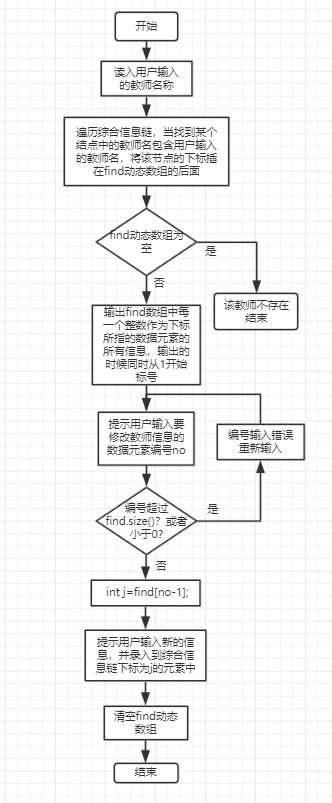


图3-6 修改教师信息的算法流程

数据维护部分（即数据的插入，删除，修改）的算法设计举例如上，关于班级、教室等的信息维护，与上面的算法完全相同，只是维护的字符串不同，在此不在赘述。

### 3.2.7 int ToQueryCourseByTeacher(void)

/\*\*

\* 函数名称: ToQueryCourseByTeacher

\* 函数功能: 查询教师综合信息

\* 输入参数: 无

\* 返 回 值: 成功返回1，出错返回-1

\*

\*/

设计思路：

1. 读入用户输入的要查询的教师姓名。
2. 遍历整个综合信息链，如果有某个结点的教师姓名字符串包含用户输入的教师姓名，则输出该结点信息。
3. 结束。

时间复杂度：O(n)。n为数据元素个数。

空间复杂度：O(1)。不需要辅助空间

流程图：

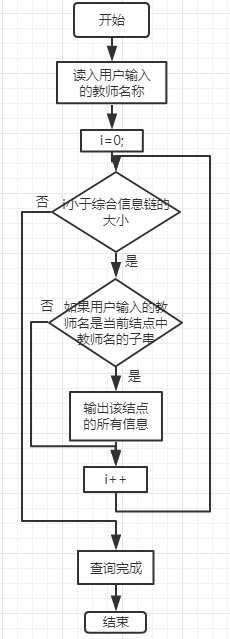


图3-7 按教师检索课程的算法流程

### 3.2.8 int ToQueryLessonByClassno(void)

/\*\*

\* 函数名称: ToQueryLessonByClassno

\* 函数功能: 通过班级检索课堂信息

\* 输入参数: 无

\* 返 回 值: 成功返回1，出错返回-1

\*

\*/

设计思路：

1. 读入用户输入的专业班号。并在班级查链上遍历查找。如果遇到某结点班级字符串包含该班级名，执行（2），否者查看下一结点。
2. 在该班级查找链结点下，顺着指针，遍历，输出所有信息。

时间复杂度：O(n)。n为数据元素个数。

空间复杂度：O(1)。不需要辅助空间

流程图：

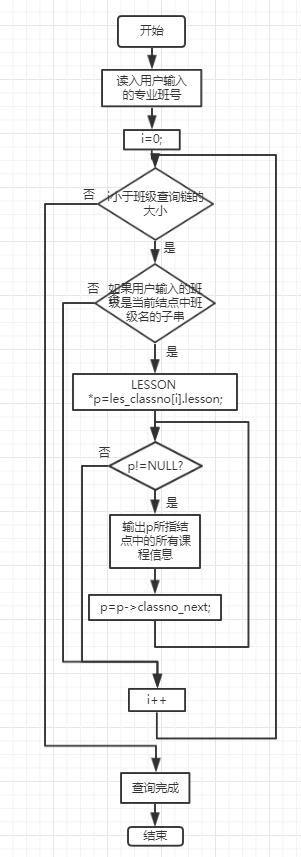


图3-8 按班级检索课堂的算法流程

### 3.2.9 int ToQueryLessonByTimeAndPlace(void)

/\*\*

\* 函数名称: ToQueryLessonByClassno

\* 函数功能: 通过时间空间查询课堂信息

\* 输入参数: 无

\* 返 回 值: 成功返回1，出错返回-1

\*

\*/

设计思路：  
 （1）提示用户输入的要查询的课堂的上课时间和地点并计算hash值。

1. 如果用户输入的格式错误，则重新输入，否则，执行（3）。
2. 在用户输入的教室的链上查找对应时间的结点。详见流程图。

时间复杂度：O(n)。n同一教室不同时间上课的课堂数。

空间复杂度：O(1)。不需要辅助空间

流程图：

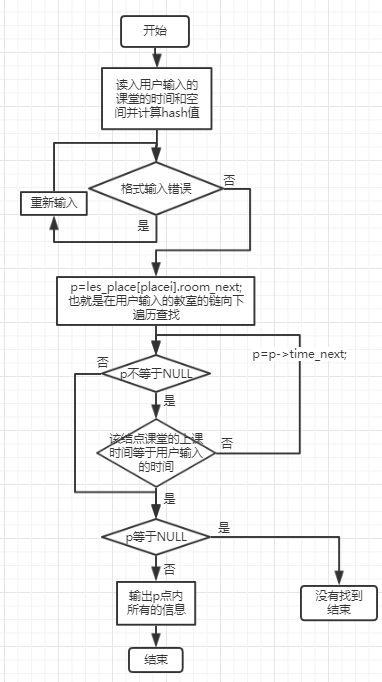


图3-9 按上课时间空间查询课堂的算法流程

### 3.2.10 int ToQueryLessonByCourse(void)

/\*\*

\* 函数名称: ToQueryLessonByCourse

\* 函数功能: 通过课程名称查询课堂信息

\* 输入参数: 无

\* 返 回 值: 成功返回1，出错返回-1

\*/

设计思路：  
 （1）读入用户输入的课程名。

（2）从课程查找链上遍历查找，如果某个表头结点的课程名包含用户输入的课程名，那么执行（3）。

（3）从该表头结点往下遍历，输出以下每个结点的信息。详见流程图。

时间复杂度：O(n)。n为同一课程的课堂数。

空间复杂度：O(1)。不需要辅助空间

流程图：

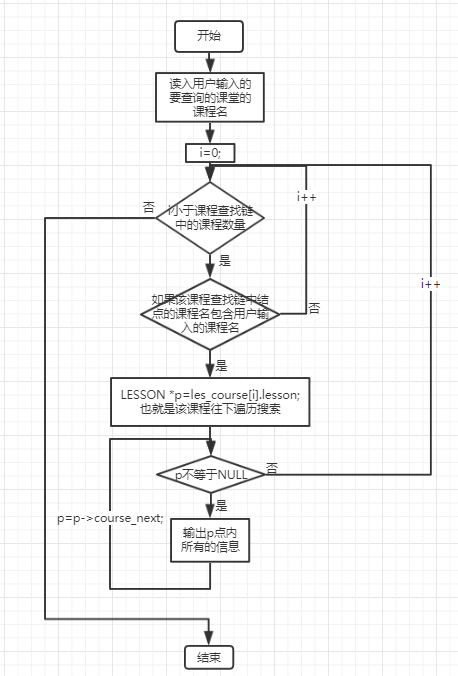


图3-10 按课程名查询课堂的算法流程

### 3.2.11 int ToQuerySpareClassroom(void)

/\*\*

\* 函数名称: ToQuerySpareClassroom

\* 函数功能: 查询空闲教室信息

\* 输入参数: 无

\* 返 回 值: 成功返回1，出错返回-1

\*/

设计思路：  
 （1）本函数的设计思路和“按时间和空间查询课堂的函数”基本相同，调用按时间和空间查询课堂的函数，如果函数返回1，说明此教室此时间非空闲，如果函数返回-1，则说明此时间此教室空闲。

时间复杂度：O(n)。n为同一教室不同时间的课堂数。

空间复杂度：O(1)。不需要辅助空间

流程图：

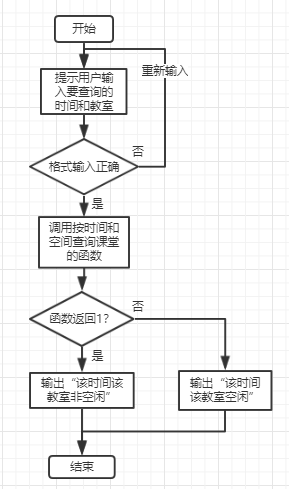


图3-11 空闲教室查询的算法流程

### 3.2.12 int ToStatClassroomRate(void)

/\*\*

\* 函数名称: ToStatClassroomRate

\* 函数功能: 统计教室利用率以及能效分析

\* 输入参数: 无

\* 返 回 值: 返回1

\*/

设计思路：  
 （1）在教室链(空间链)上遍历，如果该教室没有排课，跳过，如果有排课则执行（2）。

（2）p指向该教室的时间最早的那个课堂。执行（3）。

（3）如果p不等于NULL，执行（4），否者执行（6）。

（4）该教室的课堂数加1，将该课堂的时间加入动态数组dt中用于计算平均上课间隔。统计该教室的各个时段上课分布。如果p所指的课堂在白天上课，那么白天上课总数加一，反之晚上上课总数加一。

（5）p指向同一教室的时间上最接近的下一个课堂。执行（3）。

（6）计算该课堂的所有统计量，包括，利用率（教室课堂总数/最大排课数），能效比，平均上课间隔等。

（7）输出该教室的所有信息，包括教学楼，教室，上课时段分布，教室上课平均时间间隔，能效比，利用率等信息，以及评判分析信息，即如果当前教室的能效比低于0.33，则输出当前教室能效比过低，提示用户注意节能，如果平均的上课时间间隔太大，则提示用户注意排课紧密，增加排课等。

（8）遍历完所有的教室，输出总结信息，比如所有教室的上课时间分布汇总，然后所有教室的平均利用率，平均能效比，每个教室的平均上课时间隔。

（9）输出分析信息，比如总的能效比过低的时候，输出总结信息，当前的能效比过低，请注意节能，教室利用率过低的时候，输出提示信息增加排课，上课的平均上课间隔过大的时候，提示用户要紧密排课等等。

（10）结束。

时间复杂度：O(n)。n为教室个数。 空间复杂度：O(1)。不需要辅助空间

流程图：

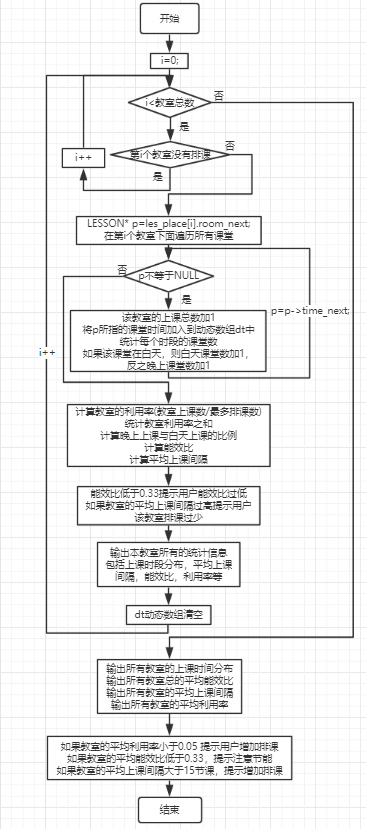


图3-12 教室利用率分析及能效分析的算法流程

### 3.2.13 int ToStatTeacherRate(void)

/\*\*

\* 函数名称: ToStatTeacherRate

\* 函数功能: 教师任课分析

\* 输入参数: 无

\* 返 回 值: 1

\*/

1.设计思路：  
 （1）遍历包含教师信息的综合信息链，i小于信息条数，如果i结点已经被读过了，就继续访问下一结点++i，否者执行（2）。如果i已经大于等于信息条数，执行（9）。

（2）标记i结点已读，然后把该结点的学分加到学分变量credit中，教师总数加一。

（3）j=i+1。

（4）在j小于信息条数时候，循环执行（5），反之执行（8）。

（5）如果j结点没有被读过则执行（6）。反之j++执行（4）。

（6）如果j结点的教师姓名是i点教师姓名的子串，则把j结点的学分信息加到credit中，并且把各种相关信息合并到i点教师的统计变量中，然后标记j结点已读。反之什么也不做。

（7）j++，执行（4）。

（8）输出i结点教师所有的统计信息，并且输出针对各项统计指标的评估与建议信息，i++,执行（1）。

（9）输出针对所教师任课情况的总的统计分析信息，然后给出总的评估以及建议。

（10）结束。

时间复杂度：O(n)。n为数据元素个数。

空间复杂度：O(1)。不需要辅助空间

2.流程图

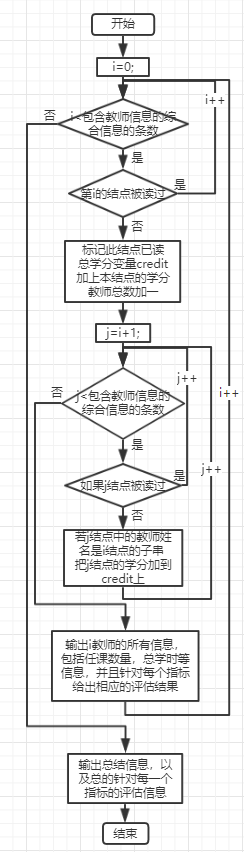


图3-13 教师任课分析的算法流程

### 3.2.14 int ToStatClassnoRate(void)

/\*\*

\* 函数名称: ToStatClassnoRate

\* 函数功能: 班级上课分析

\* 输入参数: 无

\* 返 回 值: 1

\*

\*/

设计思路：  
 （1）针对每一个班级，在班级查找链中查找该班级的上课信息。

（2）遍历整个班级查找链，如果班级查找链上的某个结点的班级包含本轮要统计的班级，则执行（3）。

（3）在该班级查找链的表头结点下往下遍历，统计所有的上课信息。

（4）如果该班级没有安排上课，回到（2）统计下一个班级，否则输出该班级的统计结果，并根据各项统计结果给出评估分析。

（5）将此班级的结果汇总。

（6）回到（2）统计下一班级。

（7）如果所有班级都统计完成，那么将汇总的结果输出，并根据各项统计结构给出总体的评估分析。

（8）结束。

\*详见流程图

时间复杂度：O(n)。n为班级的个数。

空间复杂度：O(1)。不需要辅助空间

流程图：

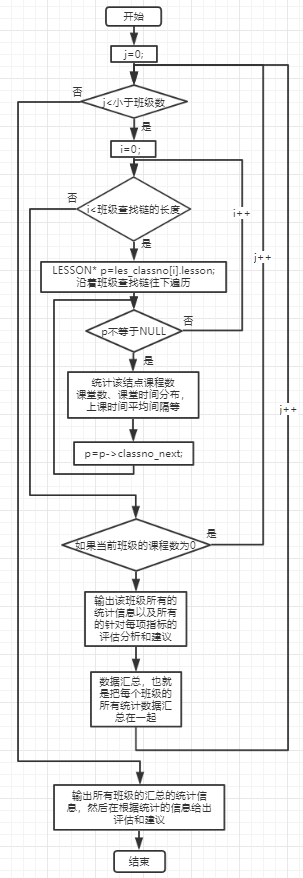


图3-14 班级上课分析的算法流程

# 4系统实现与测试

## 4.1系统实现

### 4.1.1系统实现环境

硬件以及操作系统环境：

电脑型号 戴尔 Inspiron 7559

操作系统 Microsoft Windows 10 Pro (64位)

CPU (英特尔)Intel(R) Core(TM) i5-6300HQ CPU @ 2.30GHz(2301 MHz)

主板 戴尔 0H87XC

内存 8.00 GB ( 1600 MHz)

主硬盘 130 GB ( 161596440208 已使用时间: 30小时)

显卡 NVIDIA GeForce GTX 960M

显示器 LG 9HG62 156WF6 32位真彩色 60Hz

声卡 Realtek Audio

网卡 Realtek PCIe GBE Family Controller

CPU详情：

CPU (英特尔)Intel(R) Core(TM) i5-6300HQ CPU @ 2.30GHz

CPU核心数 4

CPU默认频率 2301 MHz

CPU外频 100 MHz

CPU当前频率 800 MHz

二级缓存 1024 KB

三级缓存 6144 KB

CPU电压 0.900 V

数据宽度 64bit

指令集 MMX,SSE,SSE2,SSE3,SSSE3,SSE4.1,SSE4.2,EM64T

扩展版本 Ext.Family 0 Ext.Model 5

软件环境：本程序在64位windows10系统1709版的codeblocks下编译通过（codeblocks中自带MinGW(GNU工具在Windows上的一种移植,里面有GNU编译器套件)）

### 4.1.2系统实现函数

以下是系统实现的接口函数。接口函数通过调用3.2中介绍的功能实现函数来实现功能，当接口函数返回FALSE（0）时候系统保护性结束运行以提示用户其操作出了问题（这些问题可能来自用户对数据文件的非正常修改、强行在数据文件中冲突排课等）。main函数通过调用接口函数来实现功能。

以下只给出了与功能的实现相关的主要功能函数的调用说明，略去了部分界面框架等不重要的部分。

1. 函数名称：main

函数功能：主函数入口

输入参数：无

输出参数：无

返回值：0

调用说明：无

2. 函数名称: LoadData

函数功能: 装载数据

输入参数: 无

输出参数: 无

返 回 值: BOOL类型, 正确返回TRUE，错误返回FALSE

调用说明: 为了能够以统一的方式调用各功能函数, 将这些功能函数的原型设为 一致, 即无参数且返回值为BOOL. 返回值为FALSE时, 结束程序运行.

3. 函数名称: InitInterface

函数功能: 初始化界面.

输入参数: 无

输出参数: 无

返 回 值: 无

4. 函数名称: ClearScreen

函数功能: 清除屏幕信息.

输入参数: 无

输出参数: 无

返 回 值: 无

5. 函数名称: ExeFunction

函数功能: 执行由主菜单号和子菜单号确定的功能函数.

输入参数: m 主菜单项号

s 子菜单项号

输出参数: 无

返 回 值: BOOL类型, TRUE 或 FALSE

调用说明: 仅在执行退出系统的函数时, 才可能返回FALSE, 其他情况下总是返回TRUE

6. 函数名称: SaveData

函数功能: 向数据文件中保存数据

输入参数: 无

输出参数: 无

返 回 值: BOOL类型, TRUE(成功) 或 FALSE(失败)

7.函数名称: BackupData

函数功能: 生成数据备份文件并保存数据

输入参数: 无

输出参数: 无

返 回 值: BOOL类型, TRUE(成功) 或 FALSE(失败)

8. 函数名称: RestoreData

函数功能: 从备份文件中恢复数据

输入参数: 无

输出参数: 无

返 回 值: BOOL类型, TRUE(成功) 或 FALSE(失败)

9. 函数名称: MaintainLessonInfo

函数功能: 课堂信息维护

输入参数: 无

输出参数: 无

返 回 值: BOOL类型, TRUE(成功) 或 FALSE(失败)

10. 函数名称: MaintainCourseInfo

函数功能: 课程信息维护

输入参数: 无

输出参数: 无

返 回 值: BOOL类型, TRUE(成功) 或 FALSE(失败)

11. 函数名称: MaintainTeacherInfo

函数功能: 教师信息维护

输入参数: 无

输出参数: 无

返 回 值: BOOL类型, TRUE(成功) 或 FALSE(失败)

12. 函数名称: MaintainClassroomInfo

函数功能: 教室信息维护

输入参数: 无

输出参数: 无

返 回 值: BOOL类型, TRUE(成功) 或 FALSE(失败)

13. 函数名称: MaintainClassnoInfo

函数功能: 班级信息维护

输入参数: 无

输出参数: 无

返 回 值: BOOL类型, TRUE(成功) 或 FALSE(失败)

14. 函数名称: QueryCourseByTeacher

函数功能: 通过教师查询课程信息

输入参数: 无

输出参数: 无

返 回 值: BOOL类型, TRUE(成功)

15. 函数名称: QueryLessonByClassno

函数功能: 按班级检索课堂

输入参数: 无

输出参数: 无

返 回 值: BOOL类型, TRUE(成功)

16.函数名称: QueryLessonByTimeAndPlace

函数功能: 按教室、时间检索课堂

输入参数: 无

输出参数: 无

返 回 值: BOOL类型, TRUE(成功)

17.函数名称: QueryLessonByCourse

函数功能: 按课程检索课堂

输入参数: 无

输出参数: 无

返 回 值: BOOL类型, TRUE(成功)

18.函数名称: QuerySpareClassroom

函数功能: 空闲教室查询

输入参数: 无

输出参数: 无

返 回 值: BOOL类型, TRUE(成功)

19.函数名称: StatClassroomRate

函数功能: 教室利用与能效分析

输入参数: 无

输出参数: 无

返 回 值: BOOL类型, TRUE(成功)

20.函数名称: StatTeacherRate

函数功能: 教师任课分析

输入参数: 无

输出参数: 无

返 回 值: BOOL类型, TRUE(成功)

21.函数名称: StatClassnoRate

函数功能: 班级上课分析

输入参数: 无

输出参数: 无

返 回 值: BOOL类型, TRUE(成功)

22.函数名称: TraverseAll

函数功能: 遍历所有信息

输入参数: 无

输出参数: 无

返 回 值: BOOL类型, TRUE(成功)

23.函数名称: HelpTopic

函数功能: 输出帮助信息

输入参数: 无

输出参数: 无

返 回 值: BOOL类型, TRUE(成功)

24.函数名称: About

函数功能: 输出关于本程序的版本等等信息

输入参数: 无

输出参数: 无

返 回 值: BOOL类型, TRUE(成功)

## 4.2系统测试

### 4.2.1 初始化界面

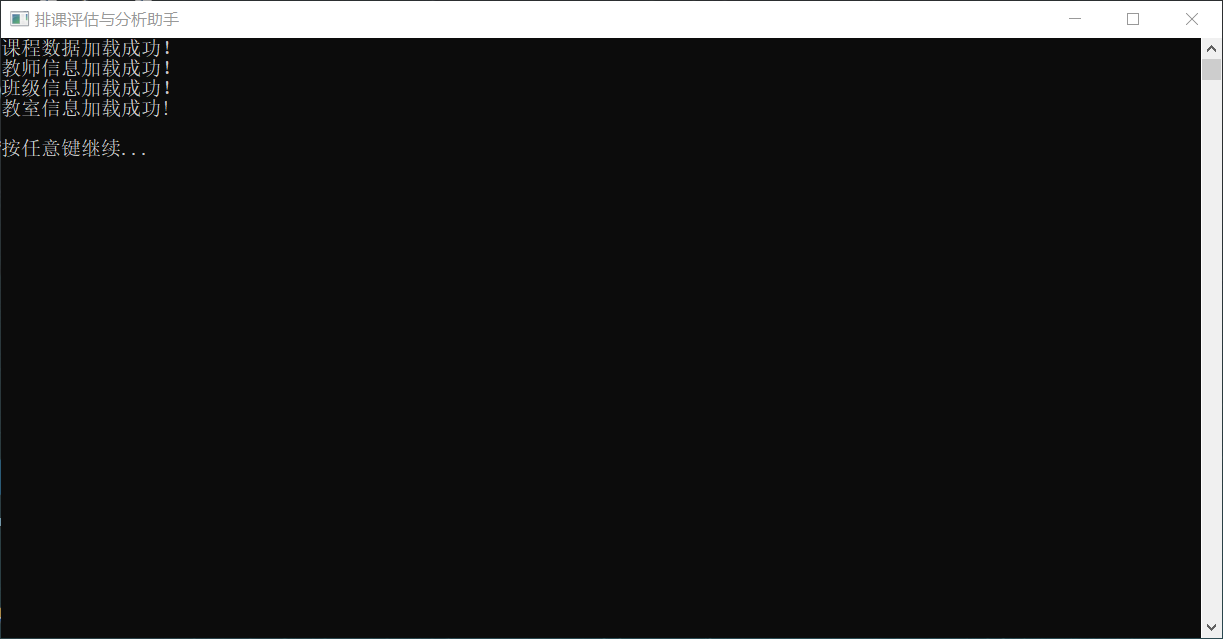
本程序是在数据文件中读入数据，属于基于文件的排课评估助手程序，当刚刚打开程序的时候，系统进行初始化，并输出初始化信息。以下是主要功能的测试。

图4-1 初始化成功

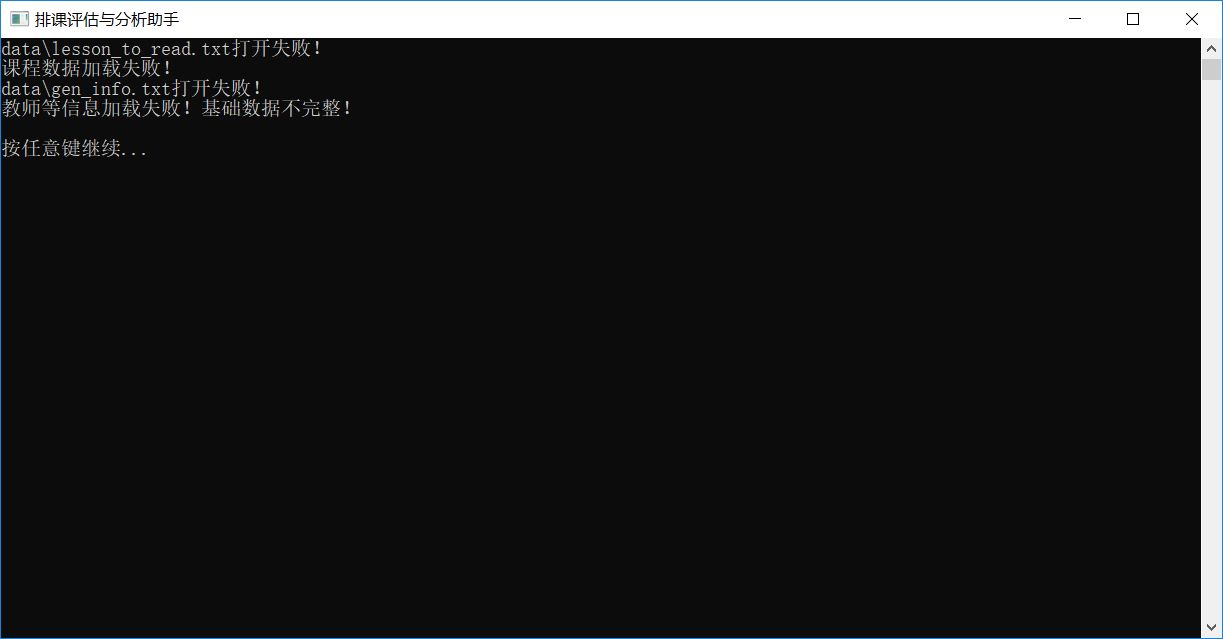
当移除数据文件后，初始化就会报错。

图4-2 数据文件损坏，初始化失败

### 4.2.2 模块测试

1. 数据维护模块

功能：实现课堂、课程、教师、教室、班级等信息数据的增加、删除、修改。

运行结果：

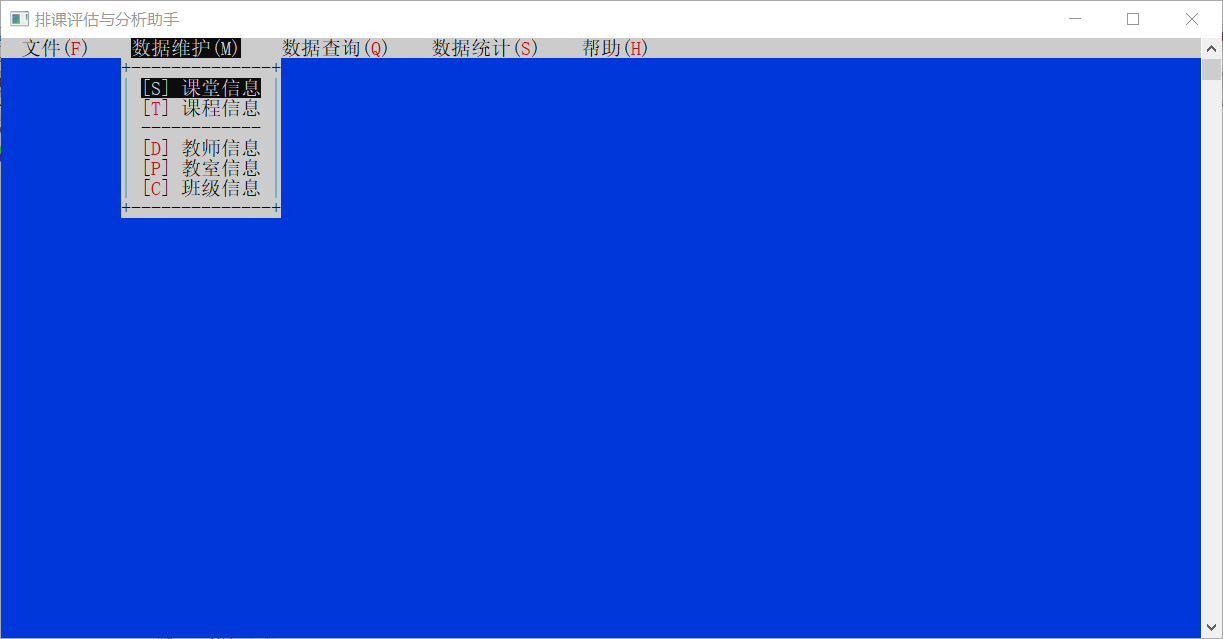
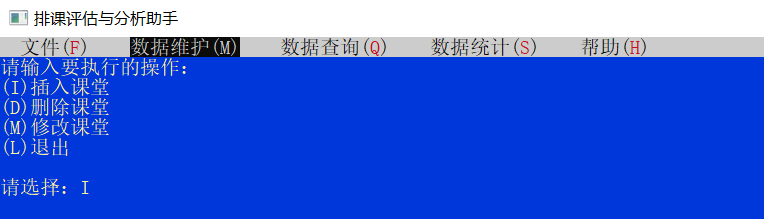
1. 运行程序，在数据文件完整正确的情况下进入系统，选择数据维护->课堂信息维护，然后首先测试插入课堂功能。

图4-3 数据维护菜单栏



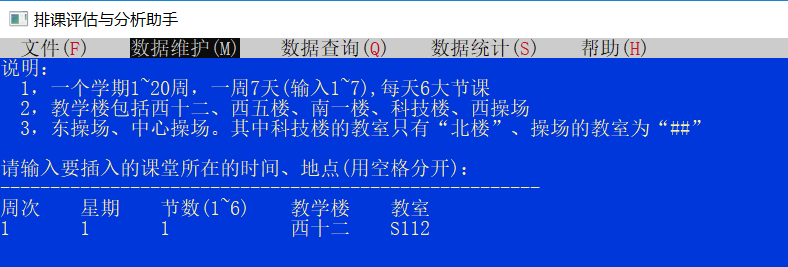
图4-4 课堂信息维护菜单栏

图4-5 输入课堂的时间空间信息，回车

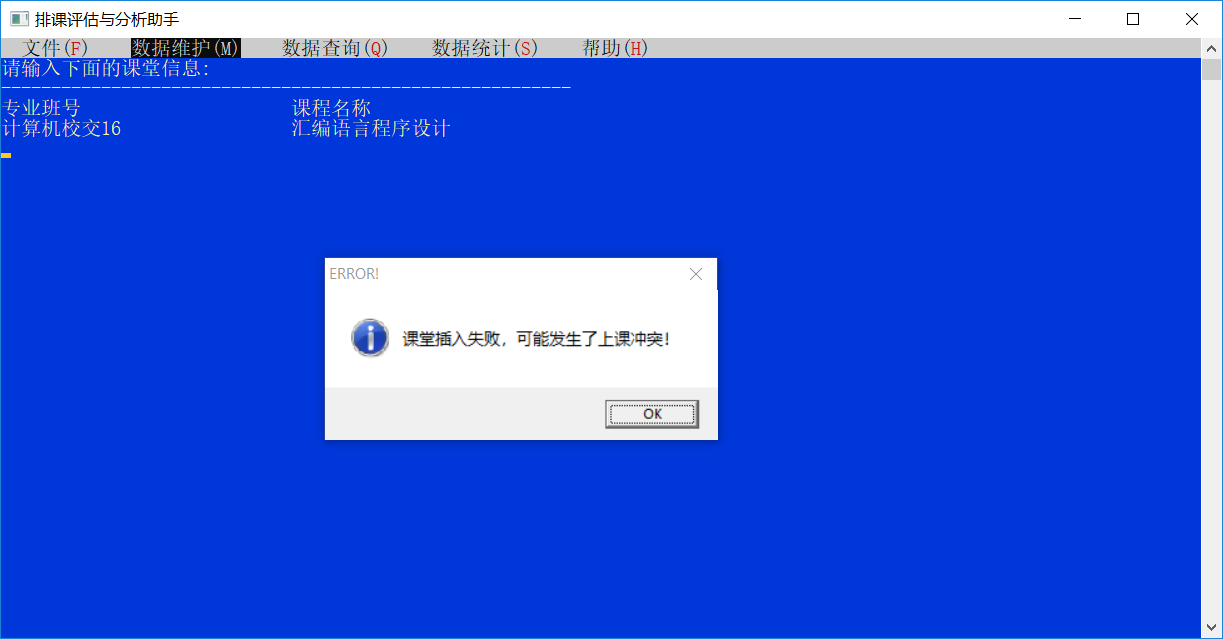
图4-6 输入课堂的详细信息，回车

图4-7 出现报错，应该是有上课冲突

如图4-6所示，插入课堂失败了，可能出现了上课冲突，我们可以在数据查询中查询一下同一时间，同一教室是不是已经被安排了课程。

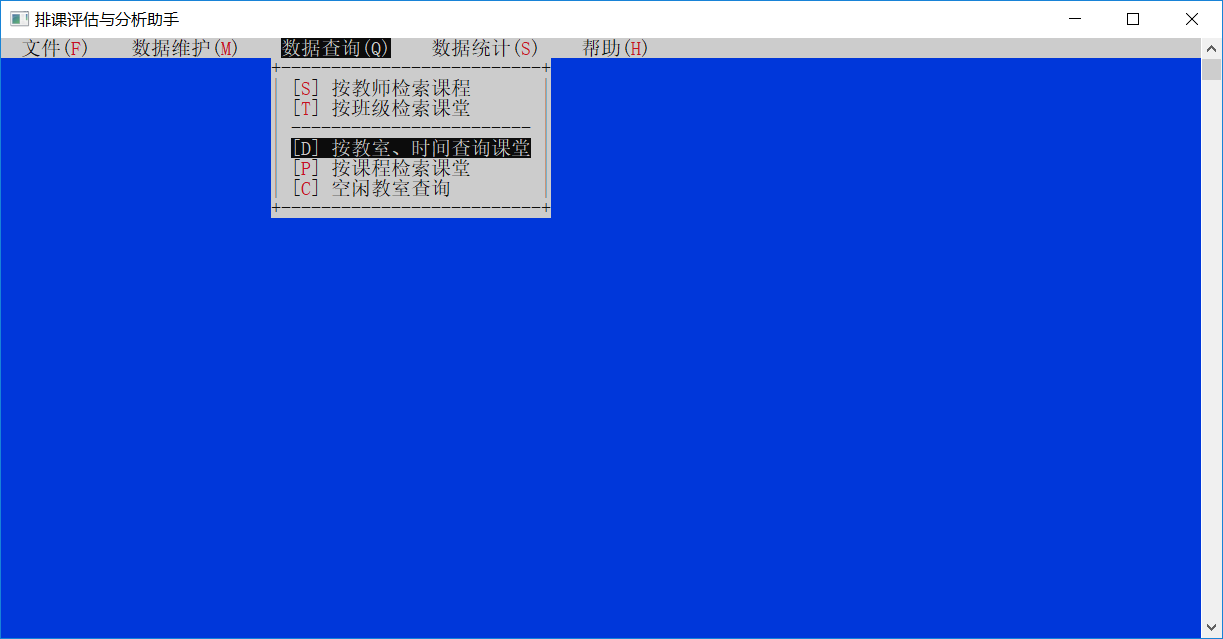


图4-8 按教室、时间查询课堂，以查看冲突原因

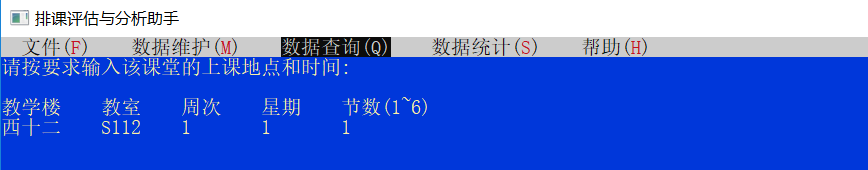


图4-9 输入刚才要插入的课堂的时间地点，回车

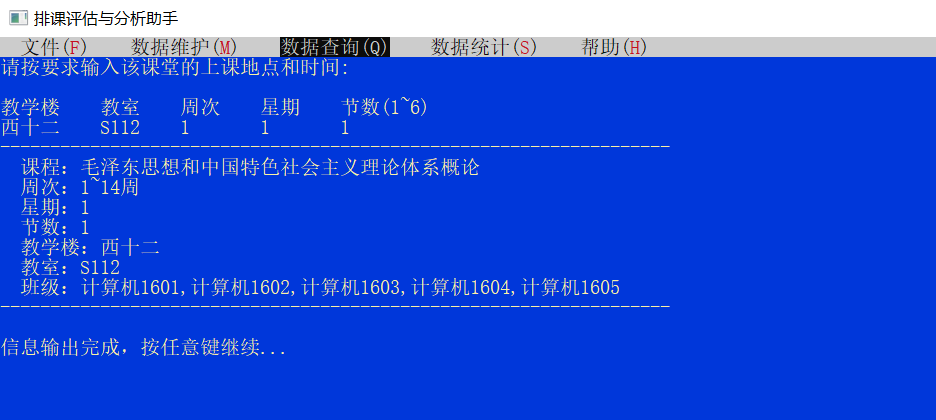


图4-10 发现此时间，该教室已经有课了，因此刚才系统报错是正确的

得知了错误的原因，我们重新的插入课堂



图4-11 更换一下课堂的时间（从第1周变为15周），再次插入

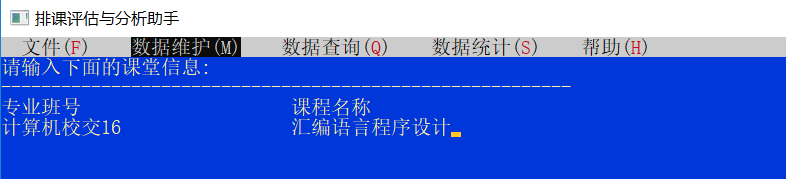
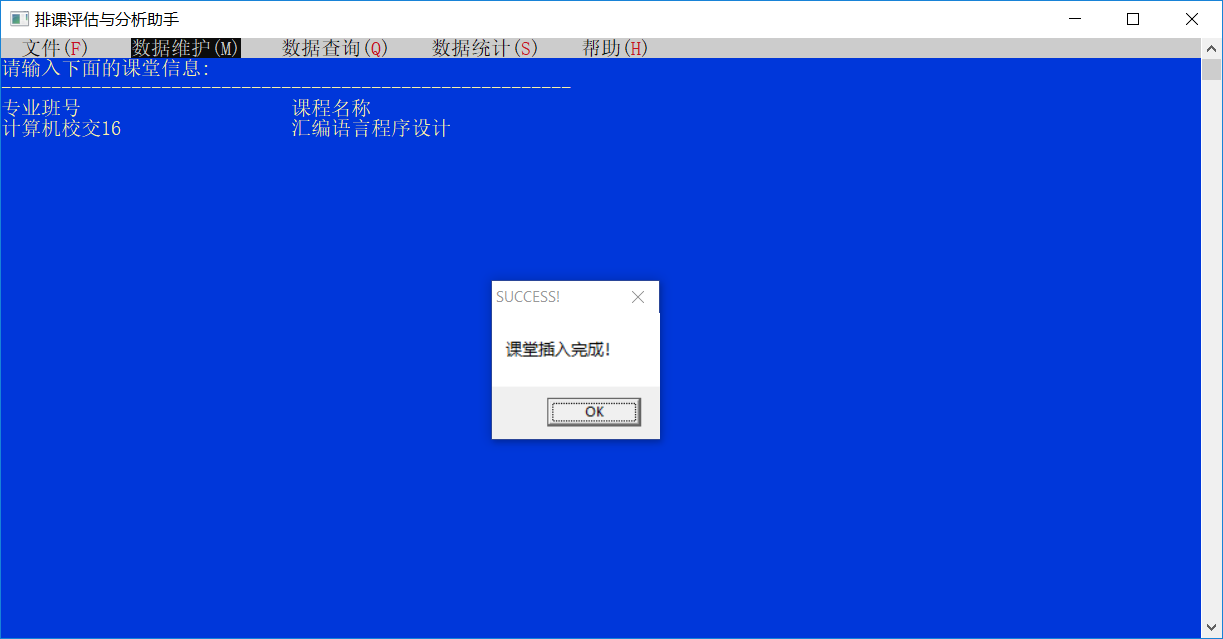


图4-12 输入详细信息

图4-13 插入成功

插入完成之后，我们查看一下内存中是否已经有了我们要插入的数据。这次我们选择按班级查找。

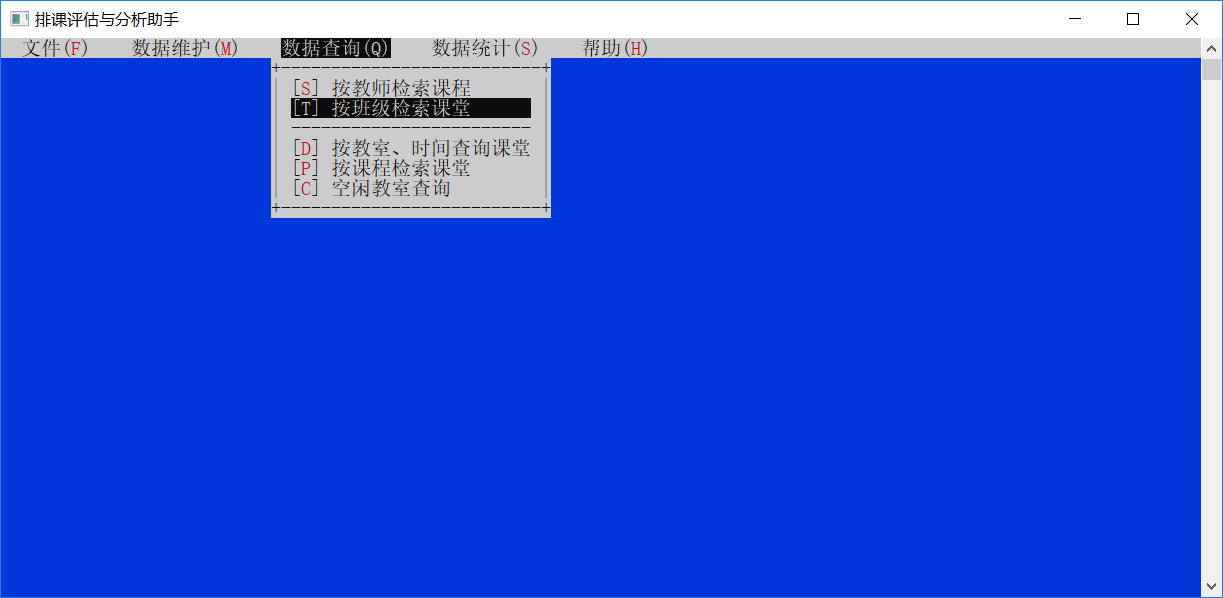


图4-14 在数据查询中选择按班级检索课堂

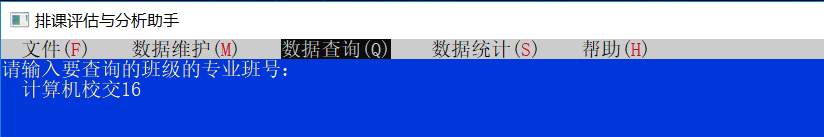


图4-15 输入刚才插入的课堂的上课班级，回车



图4-16 发现刚才插入的课堂，是图中第17个

查询到了刚才插入的课堂，说明插入功能是可以正确运行的。下面测试课堂的删除功能。先在数据维护菜单中选择课堂信息，然后输入D来执行删除课堂。

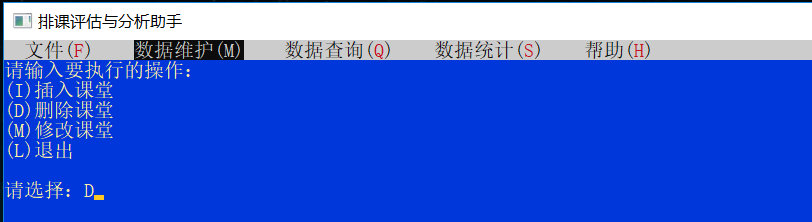


图4-17执行删除课堂的测试

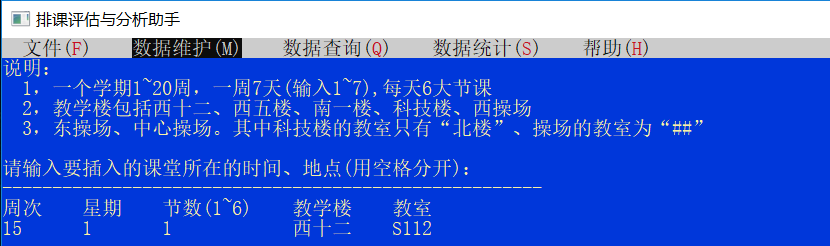


图4-18输入刚才插入的课堂的信息，回车

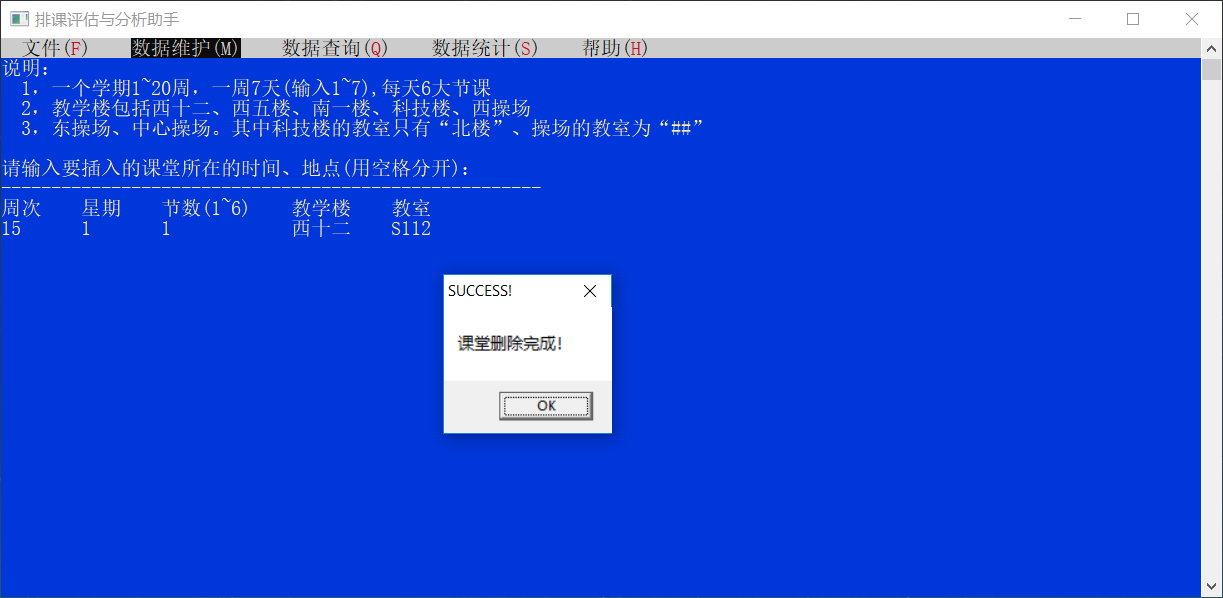


图4-19课堂删除成功

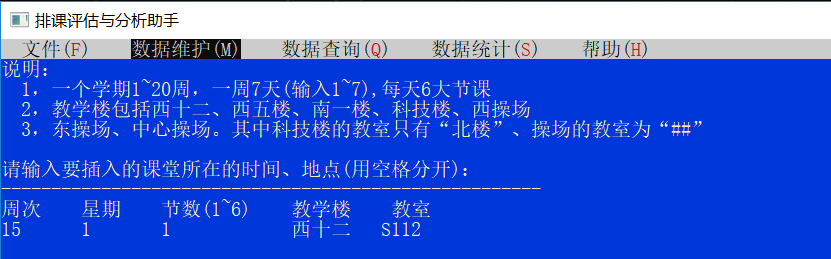


图4-20再次执行删除课堂功能，回车

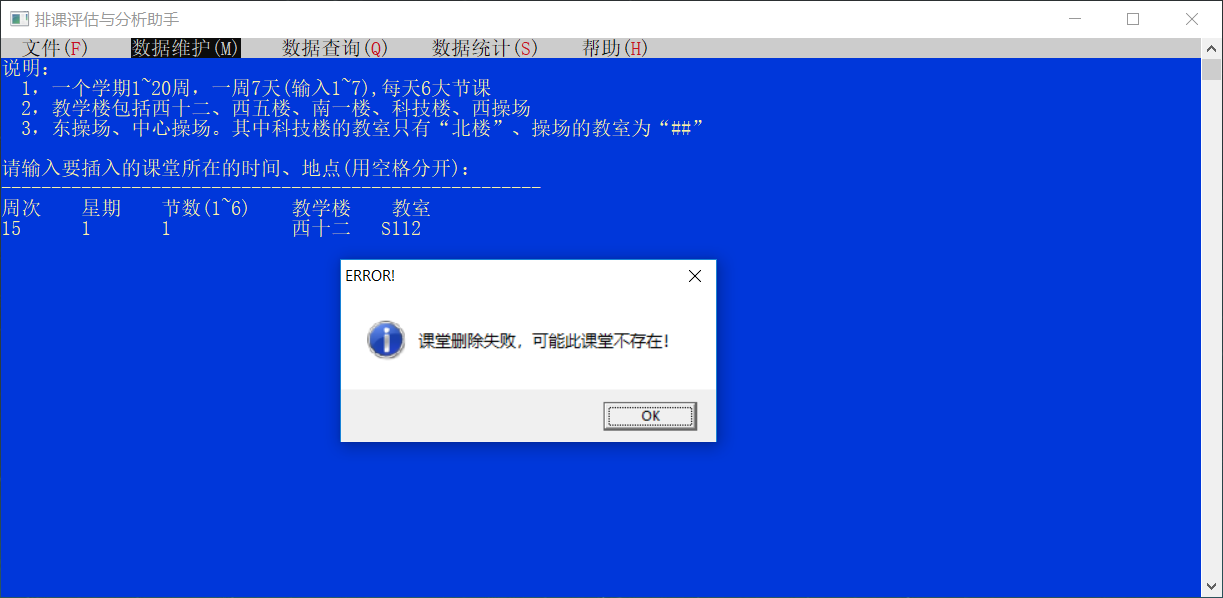


图4-21删除失败，因为这个课堂刚才已经删除，已经不存在

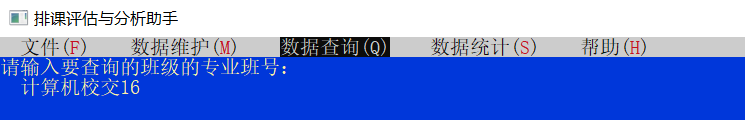


图4-22 再次按班级查询这个已经被删除的课堂

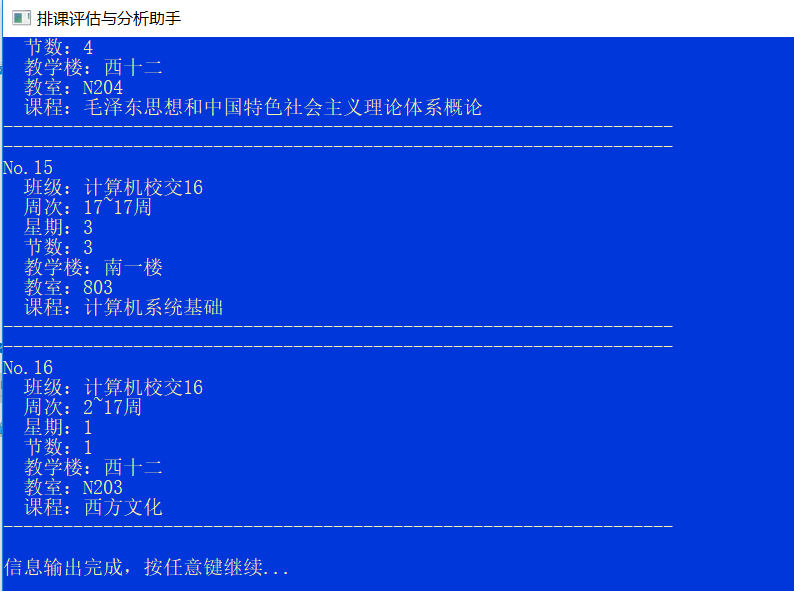


图4-23 发现该班级第17个查询结果没有了，说明删除成功

删除功能正确执行，下面测试课堂修改部分。

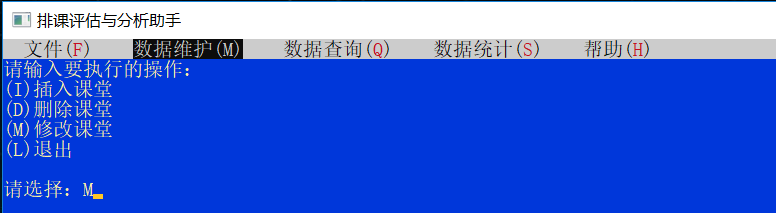


图4-24 执行课堂修改功能

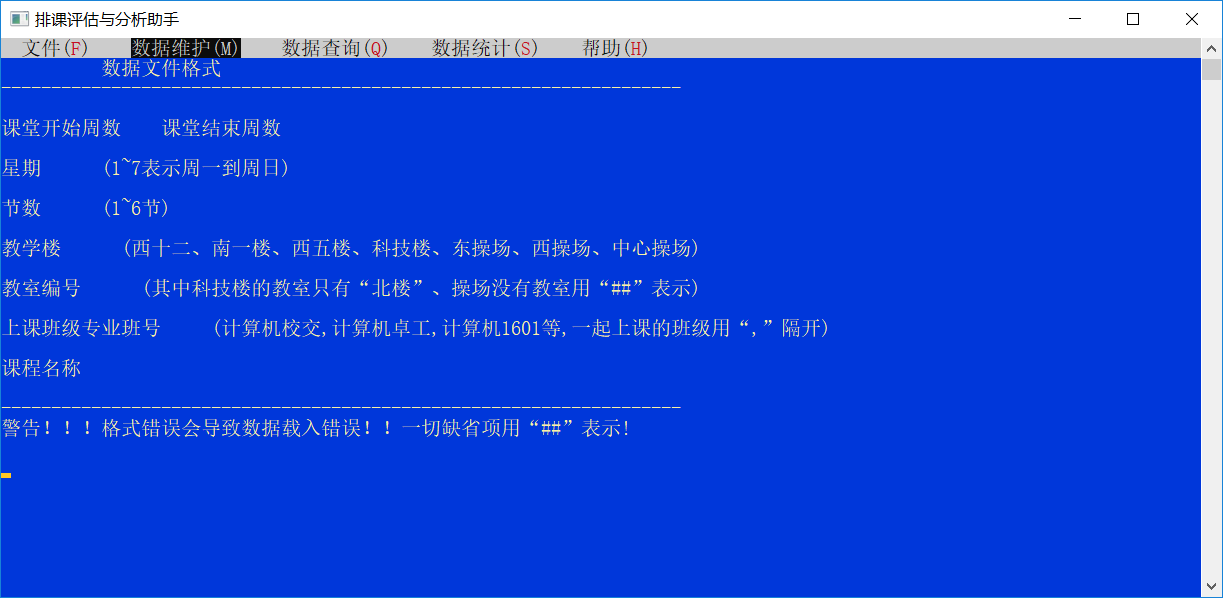


图4-25 执行课堂修改功能

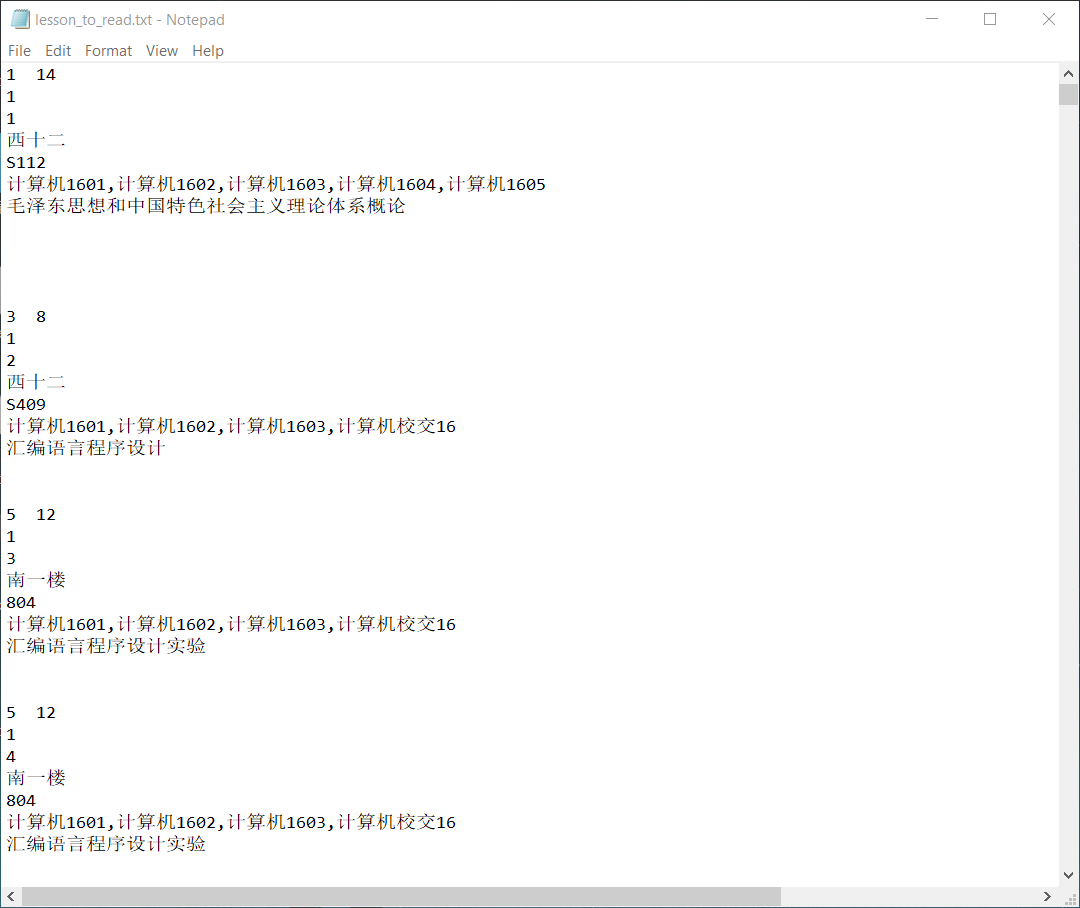


图4-26 弹出配置文件提供用户修改



图4-27 如图所示把第一个课堂改为1到15周



图4-28 修改之后保存，提示修改成功

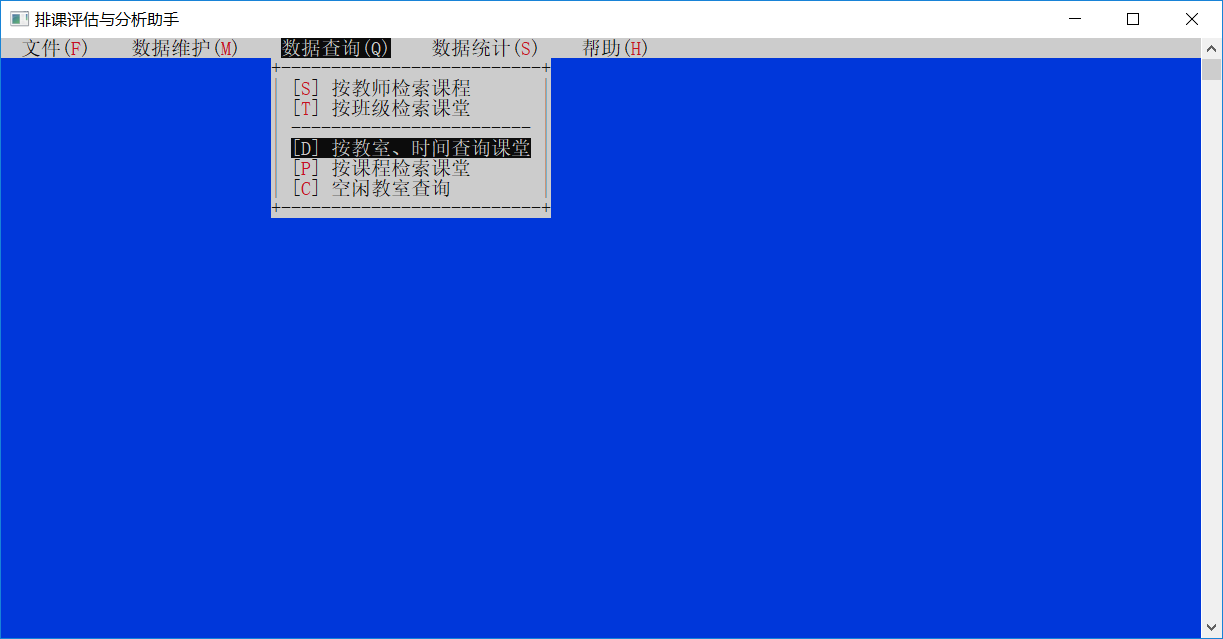


图4-29 按时间地点查询课堂

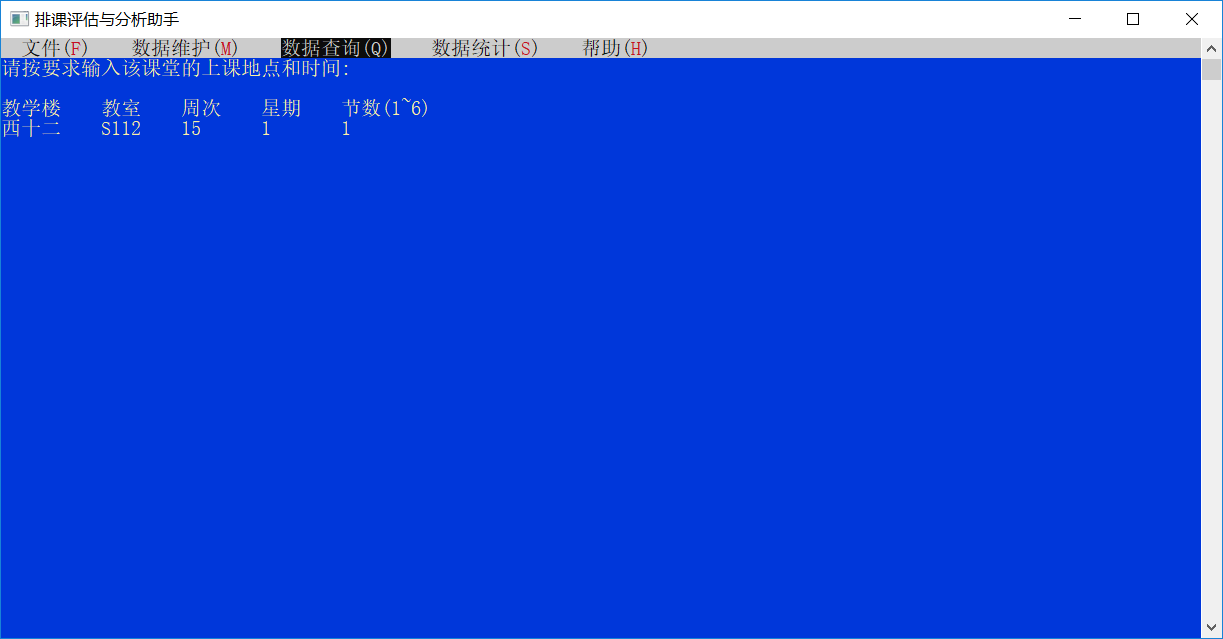


图4-30 输出刚才修改的时间



图4-31 查询到了，证明修改成功！

1. 教师信息维护演示。以下是数据维护中教师信息维护功能的执行情况。

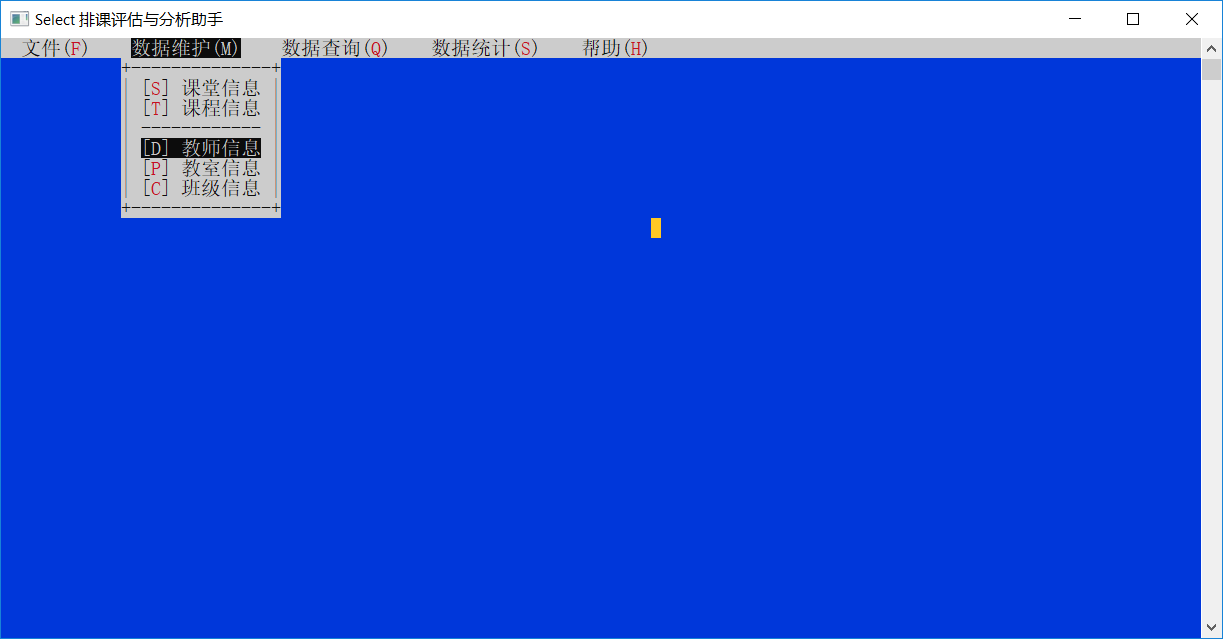


图4-32 进入教师信息维护

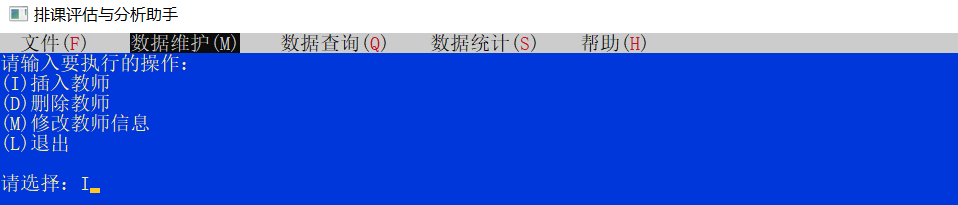


图4-33 执行插入教师

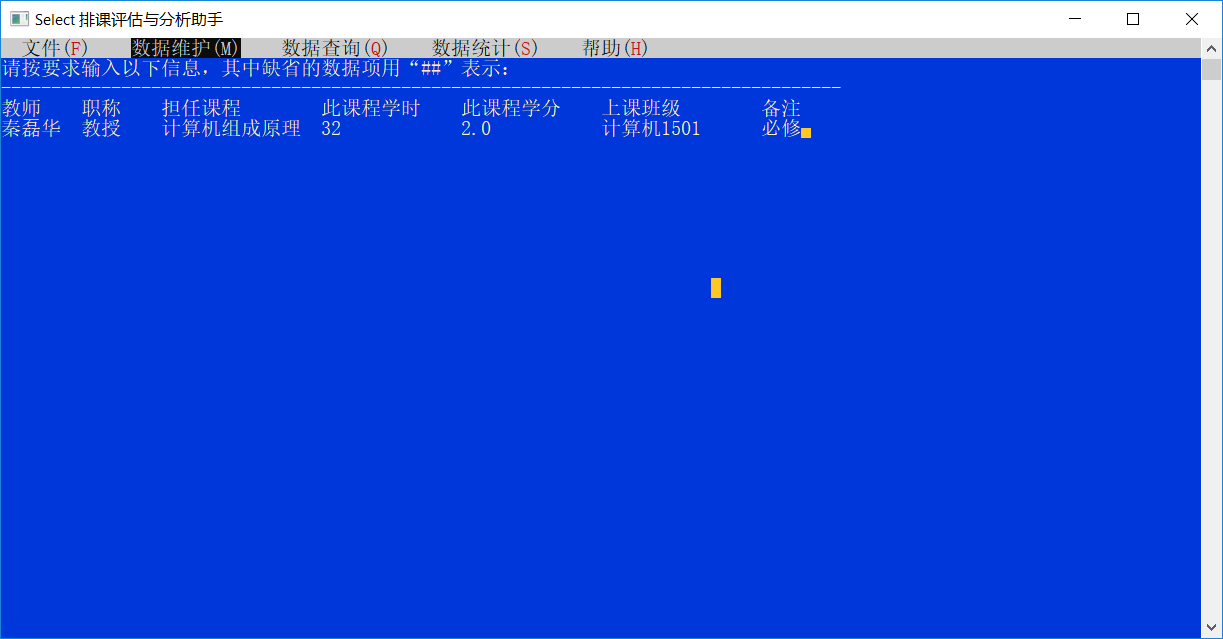


图4-34 插入一个教师的信息，回车

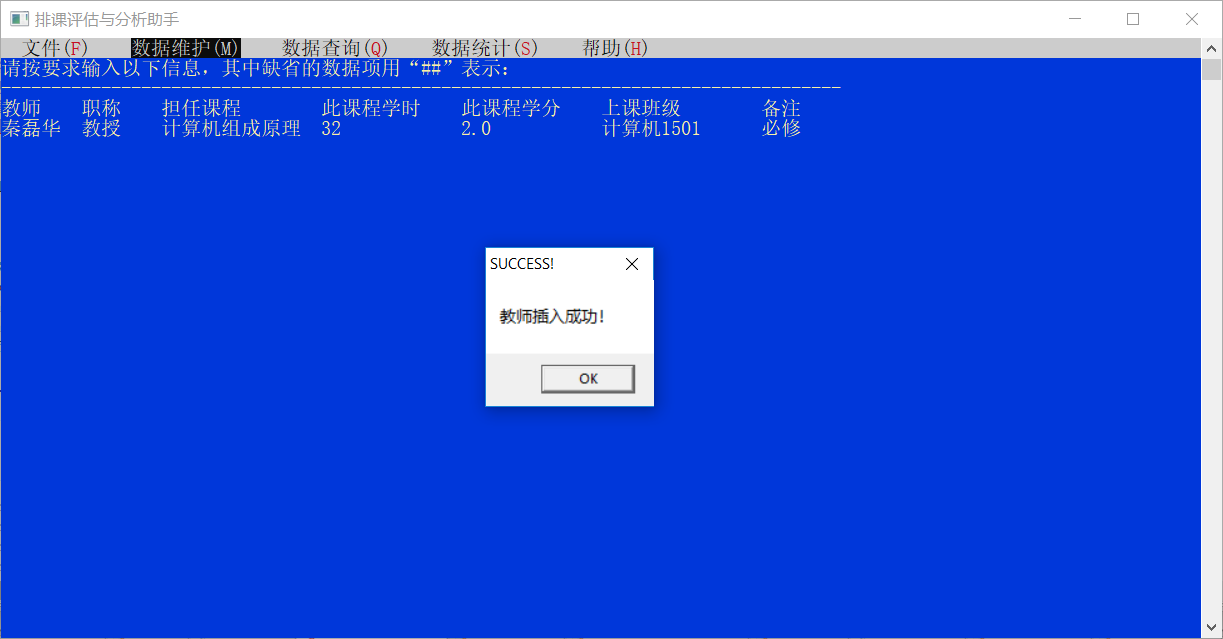


图4-35 插入成功

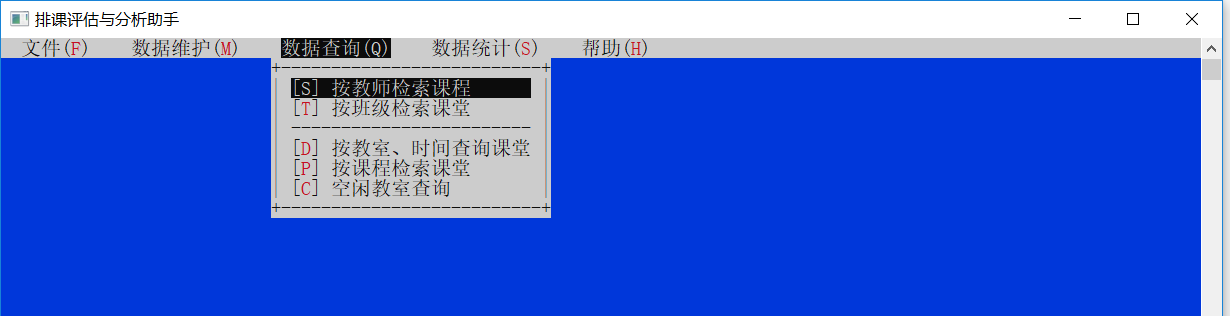


图4-36 按教师检索课堂，看是否真的插入成功

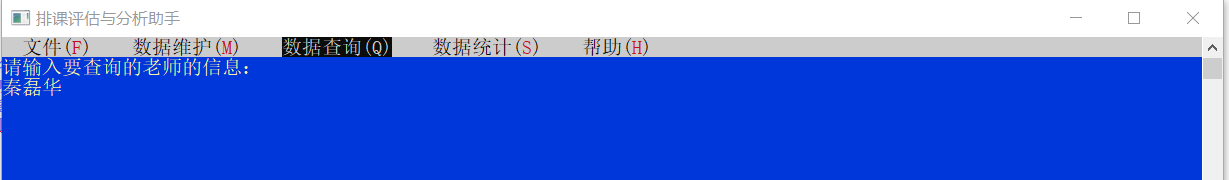


图4-37 输入刚才插入的教师姓名，回车

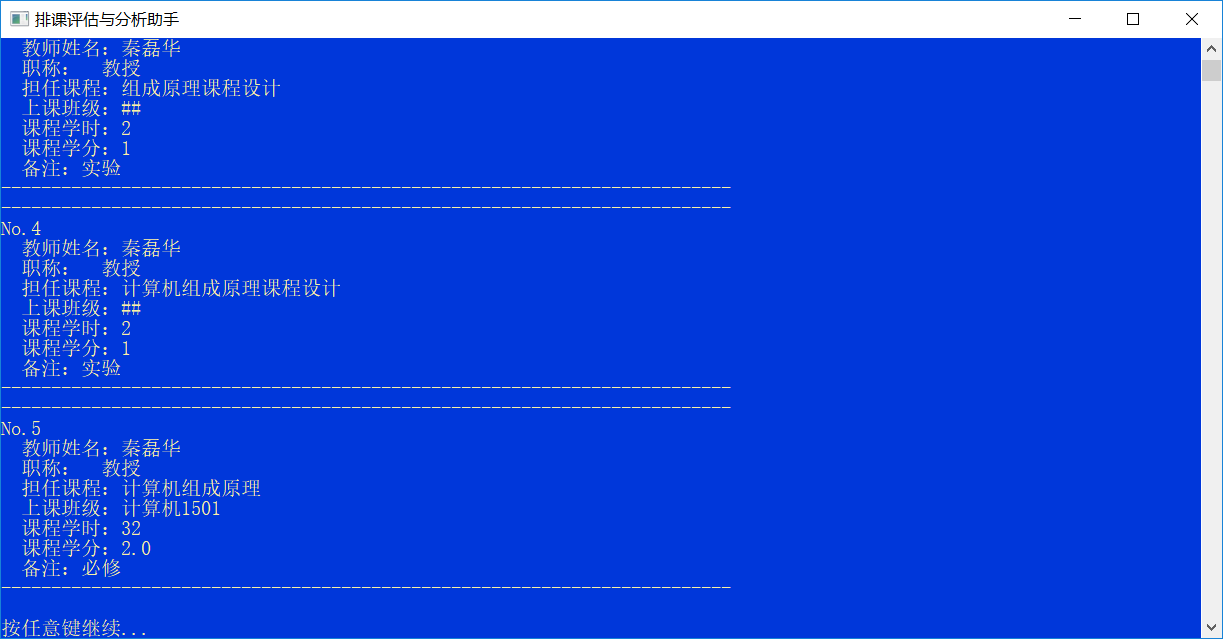


图4-38 发现图中第5个信息就是刚才插入的，说明插入成功

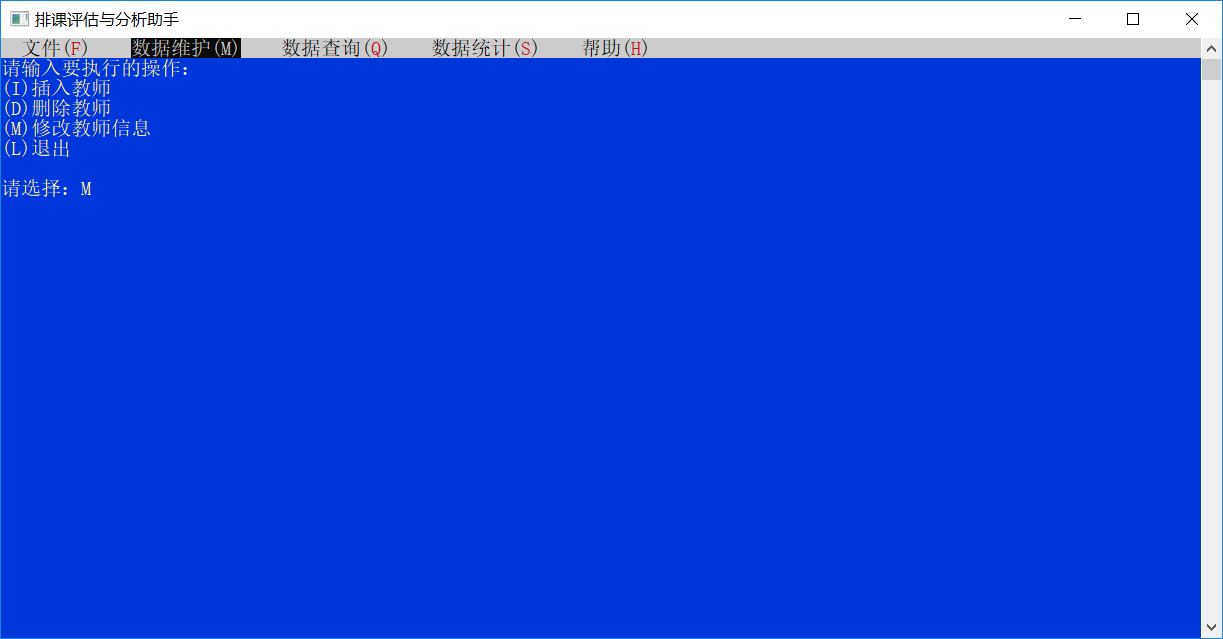


图4-39 测试教师信息的修改功能

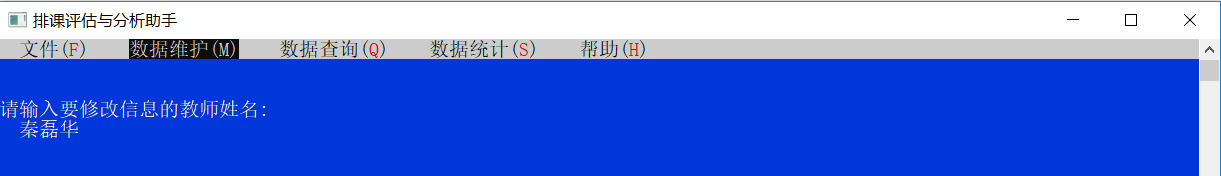


图4-40 输出要修改信息的教师姓名，回车

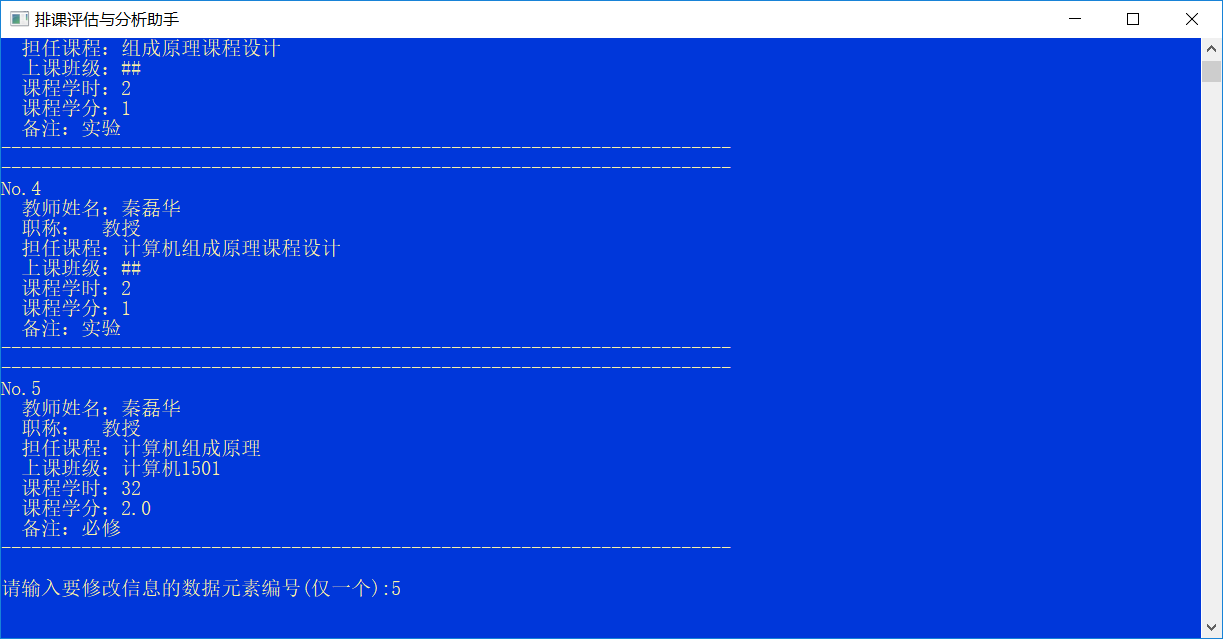


图4-41 程序输出所有与该教师相关的数据元素，输入要修改的编号，回车

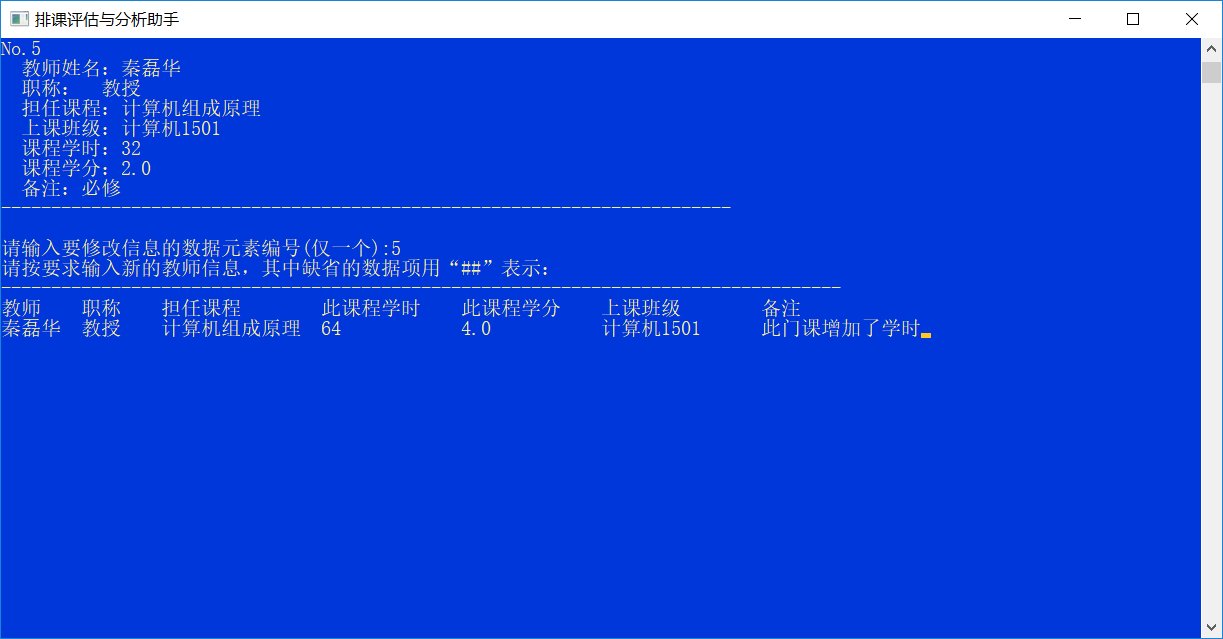


图4-42 输入新的教师信息，回车

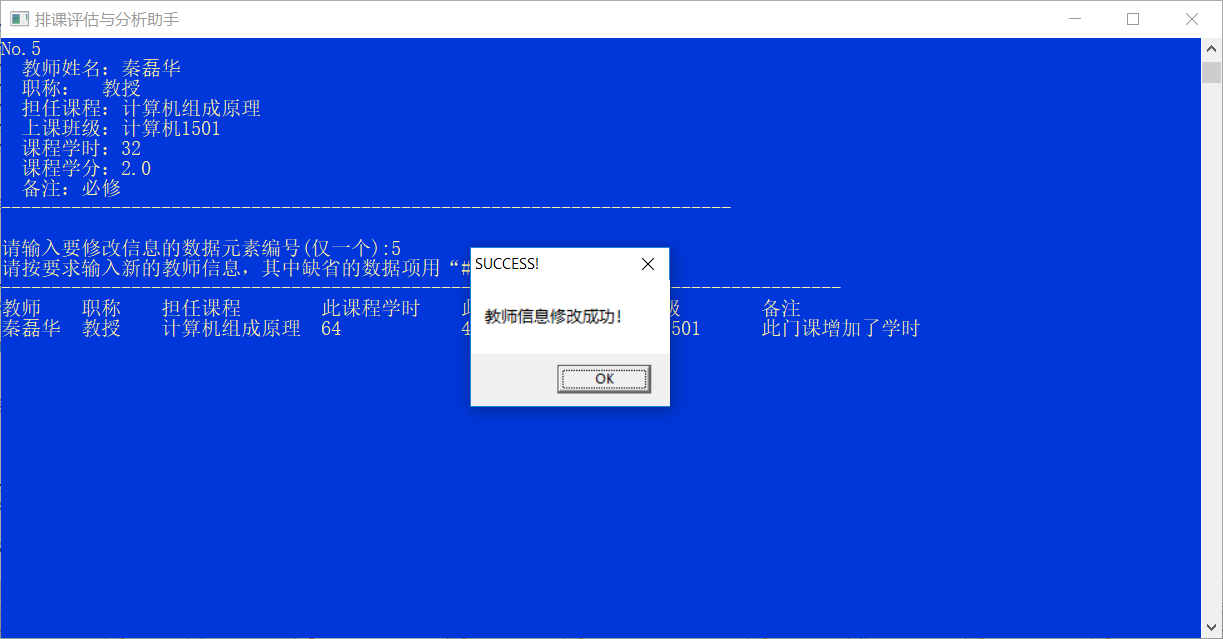


图4-43 修改成功

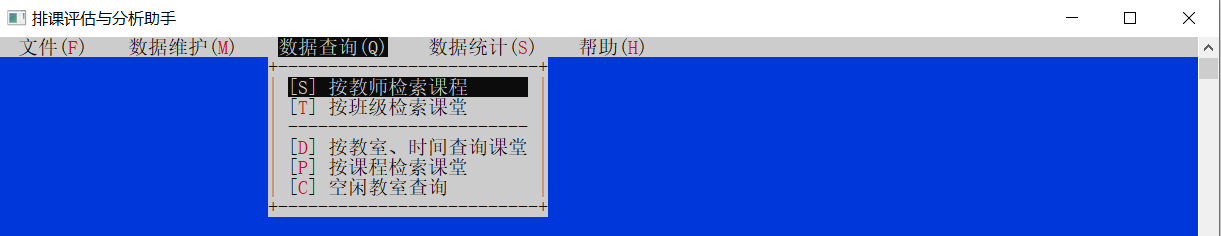


图4-44 按教师检索课程，看看是否修改成功

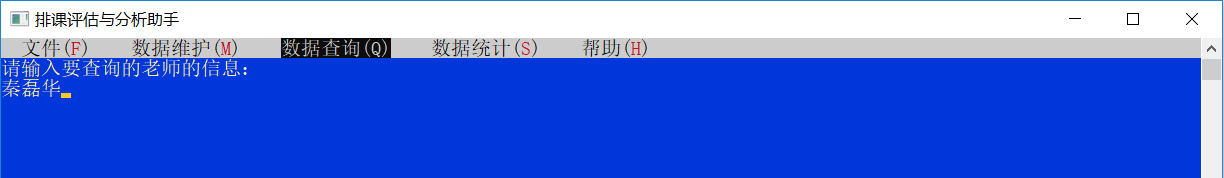


图4-45 输入刚才修改信息的教师名

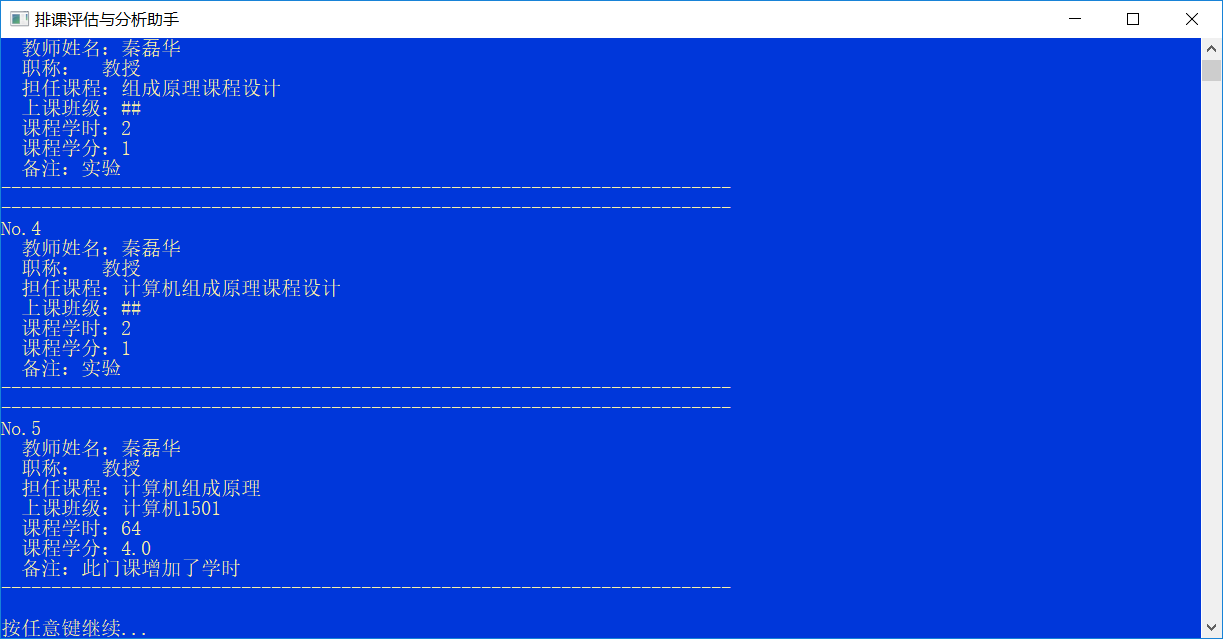


图4-46 发现第5条正是刚刚修改过得信息，说明修改成功

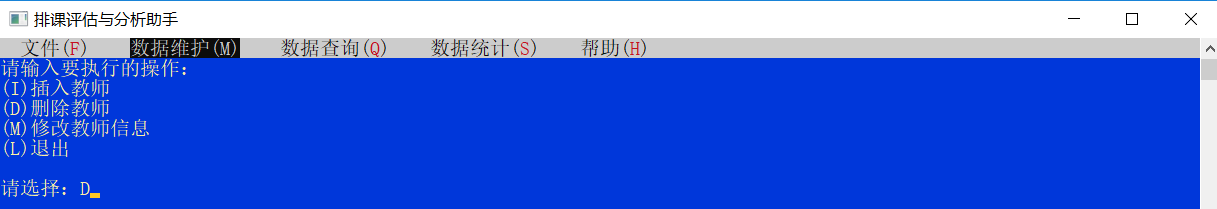


图4-47 执行删除功能

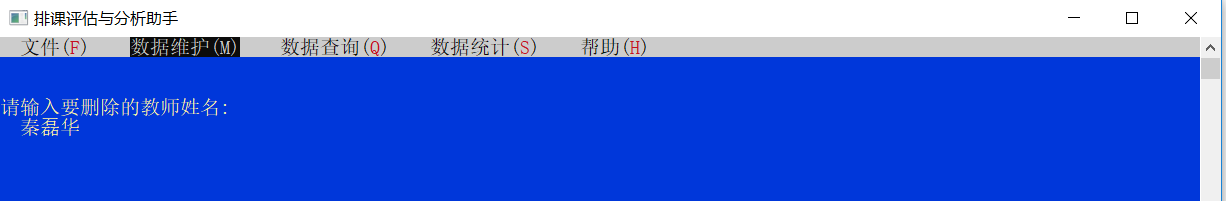


图4-48 输入要删除的教师姓名，回车



图4-49 程序输出所有与该教师相关的信息，提示用户是否删除该教师以及所有与该教师相关的信息，这里先选着否

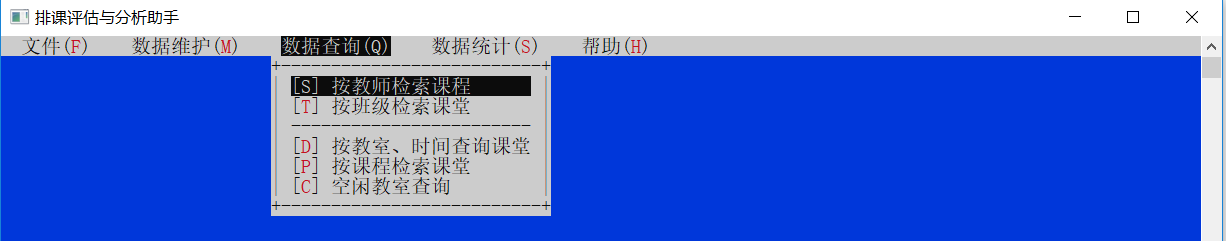


图4-49 再次查询，看看取消删除的数据是否还在

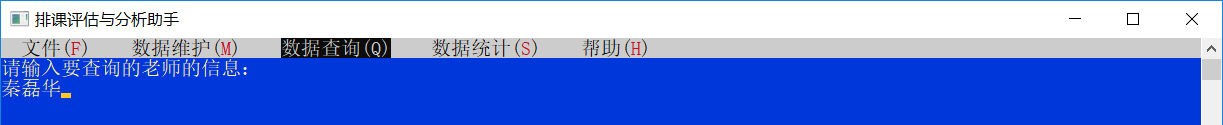


图4-50 再次输入刚才取消删除的教师姓名，回车

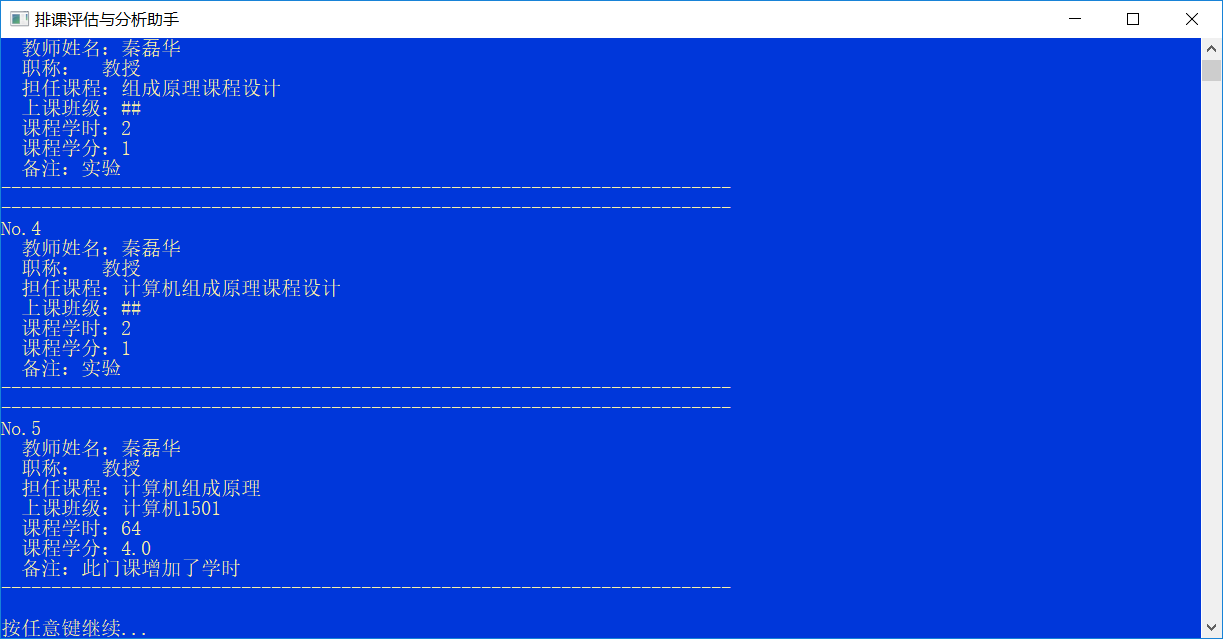


图4-51 发现数据还在，说明取消操作成功

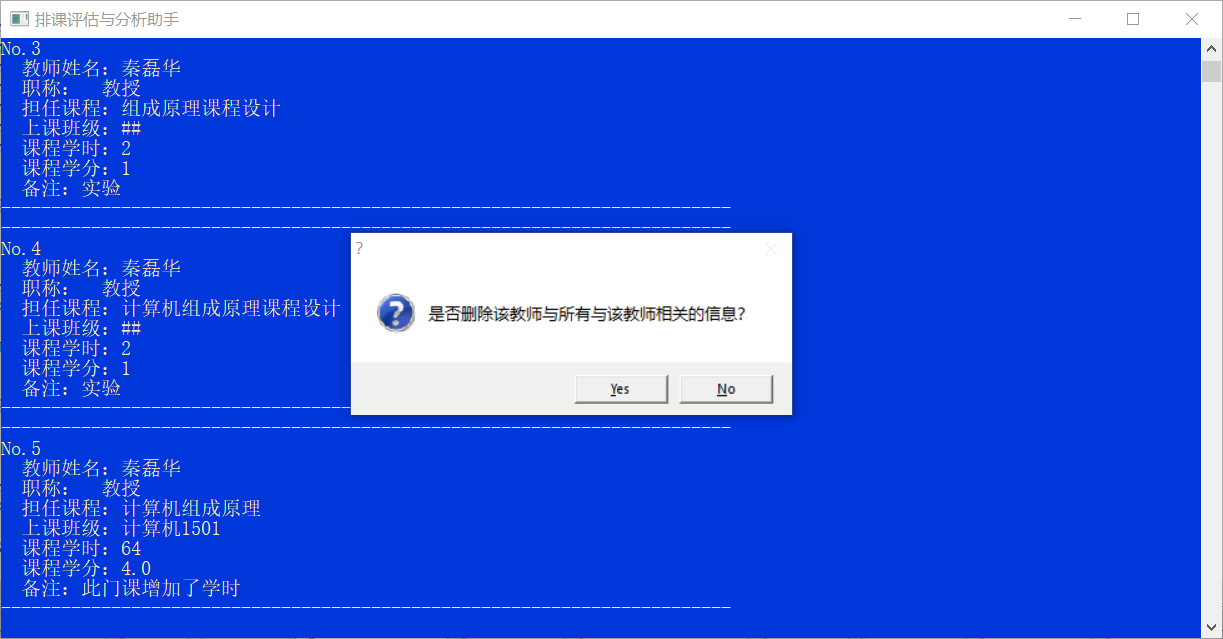


图4-52 再次执行删除操作，这次选着Yes

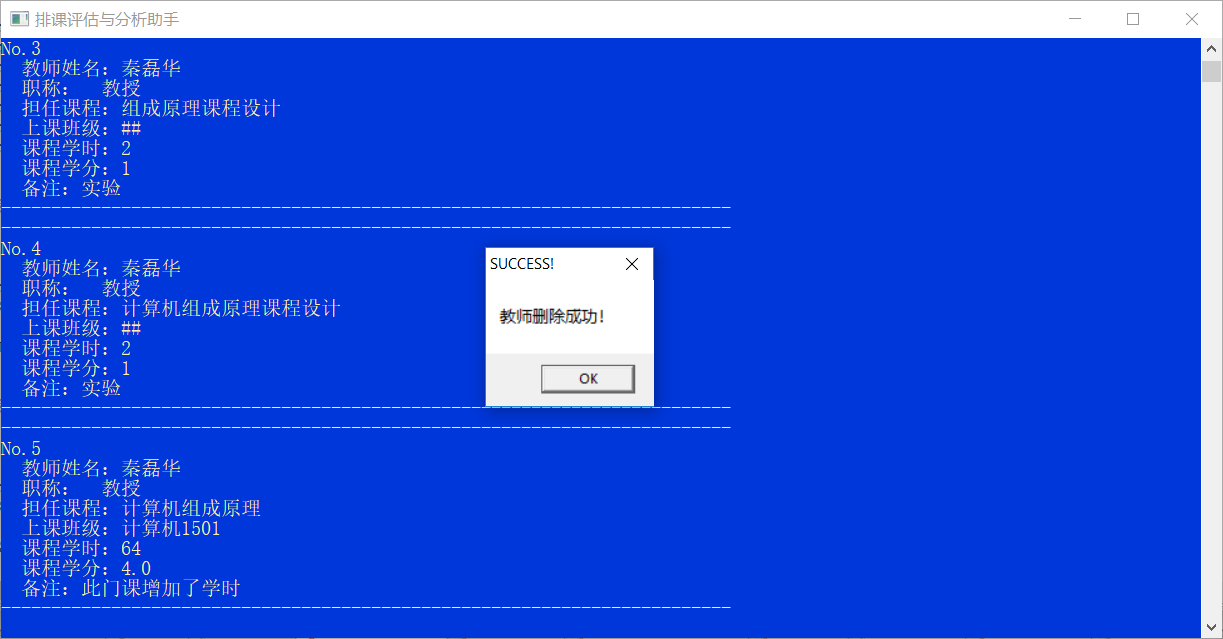


图4-53 提示删除成功

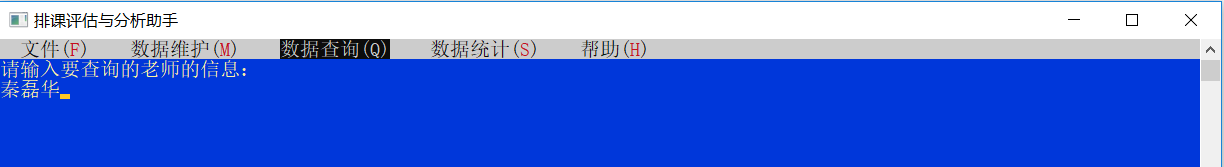


图4-54 再次查找该教师信息

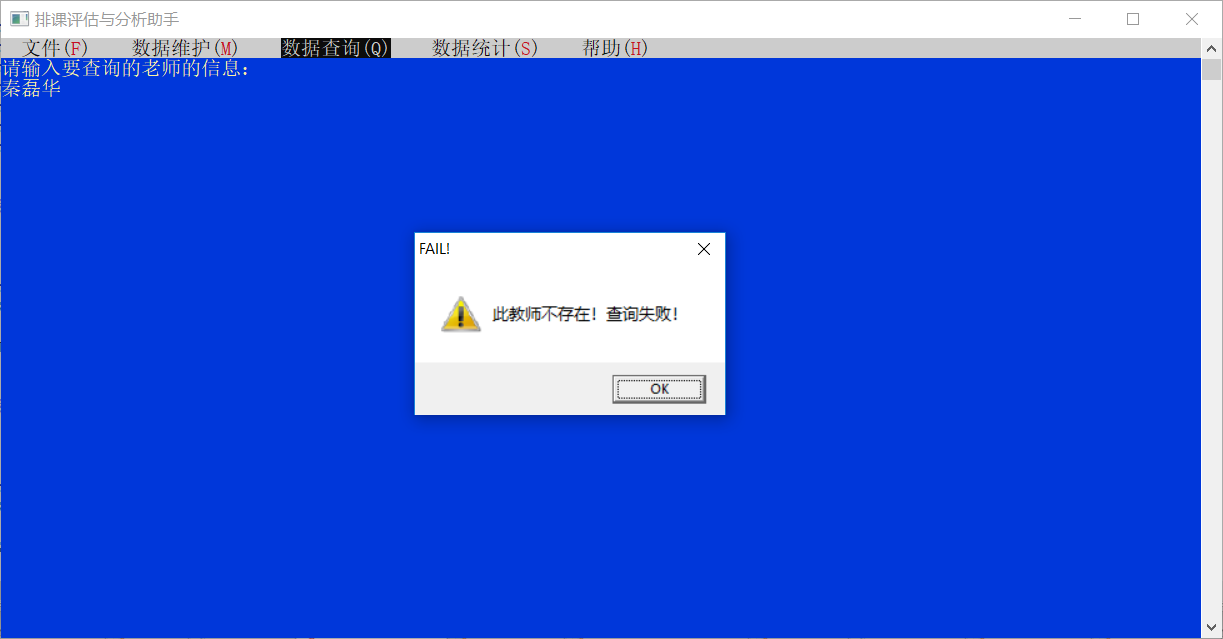
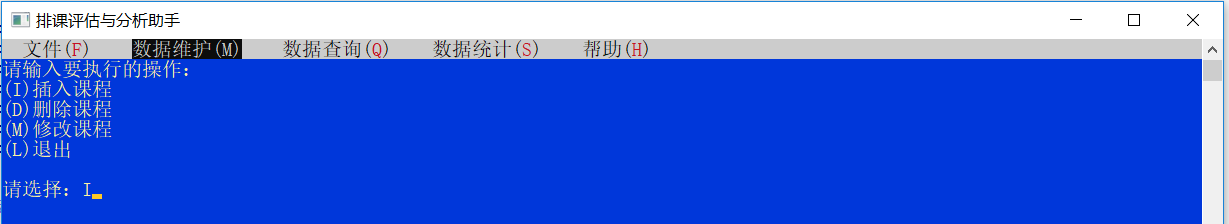


图4-55 提示该教师不存在，说明删除成功

1. 课程信息维护。下面执行本程序数据维护模块中的课程信息维护功能。



图4-56 执行课程信息维护功能



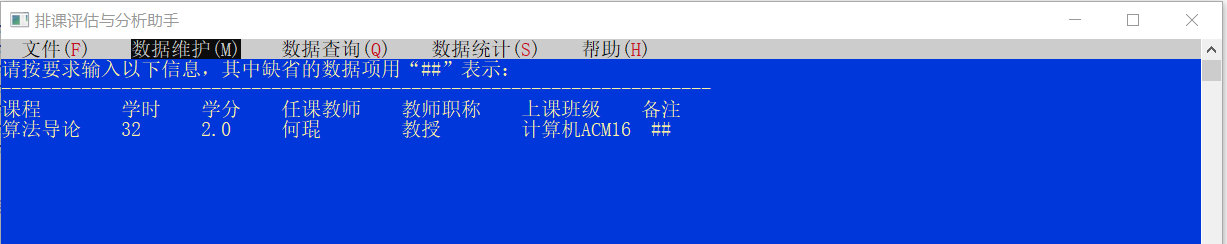
图4-57 插入课程

图4-58 输出新的要插入的课程信息

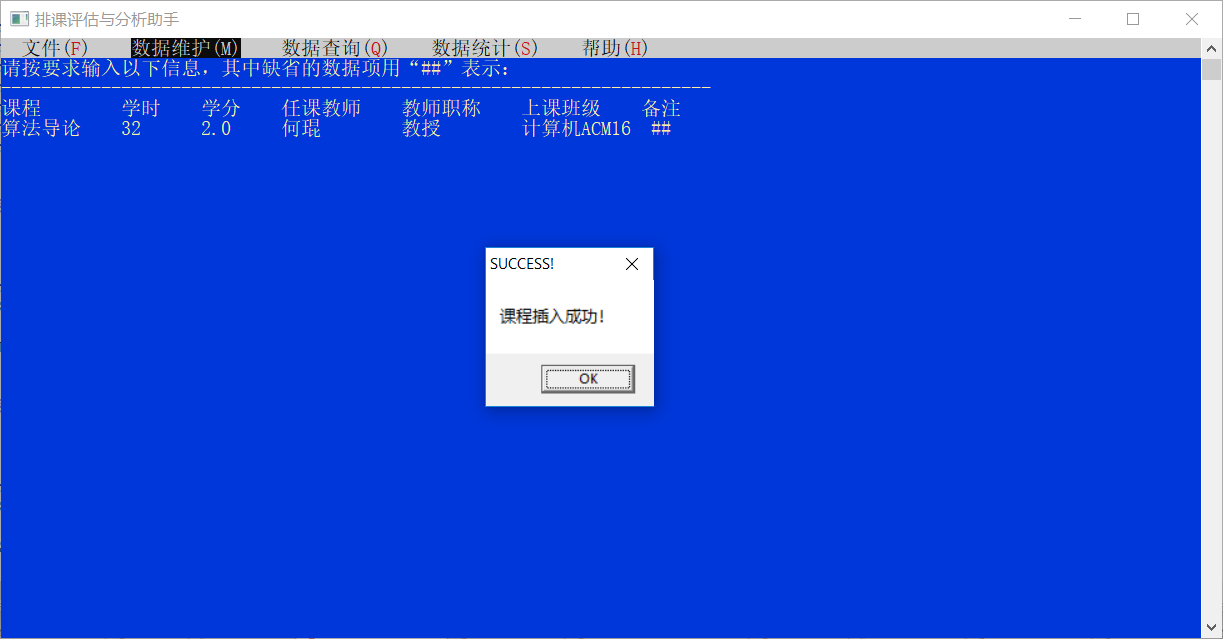


图4-59 插入成功



图4-60 按教师检索，看课程是否插入成功

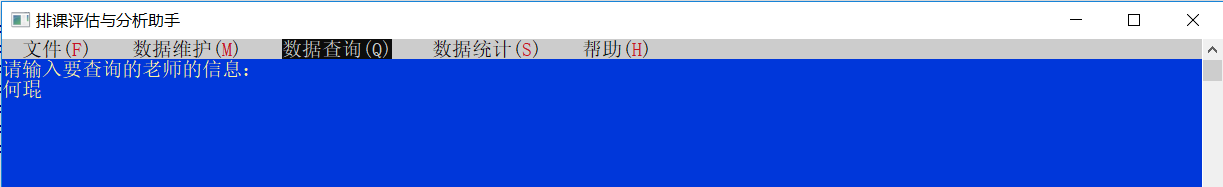


图4-61 输入刚才插入课程的任课教师，回车

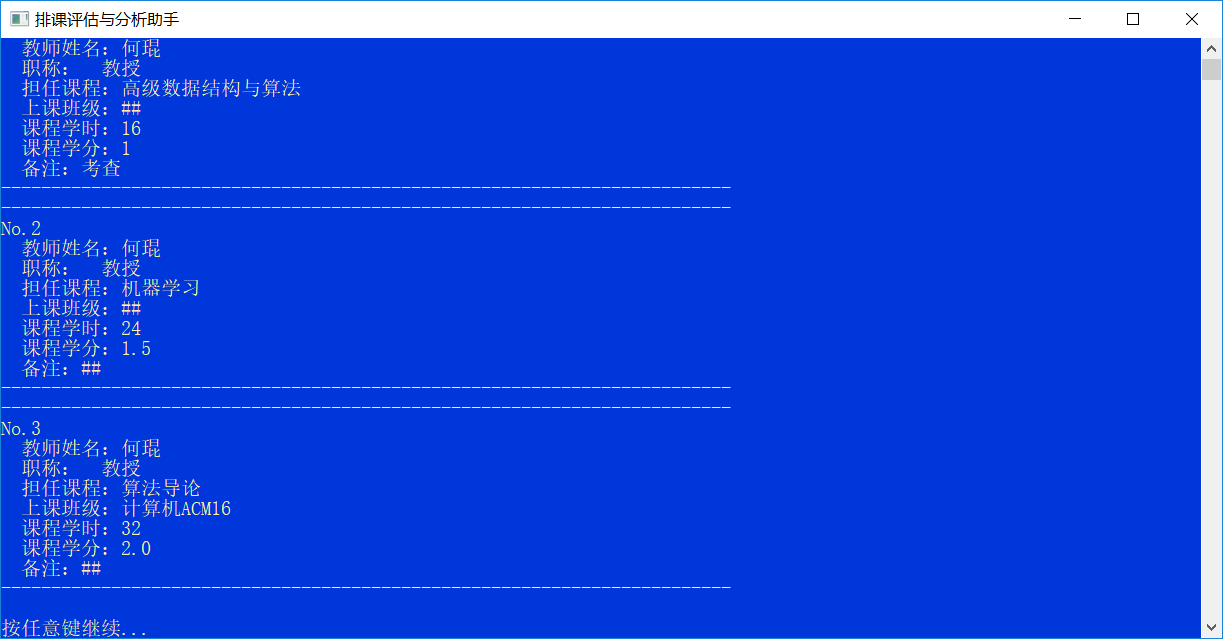


图4-62 发现第3个数据就是刚才插入的数据，插入成功

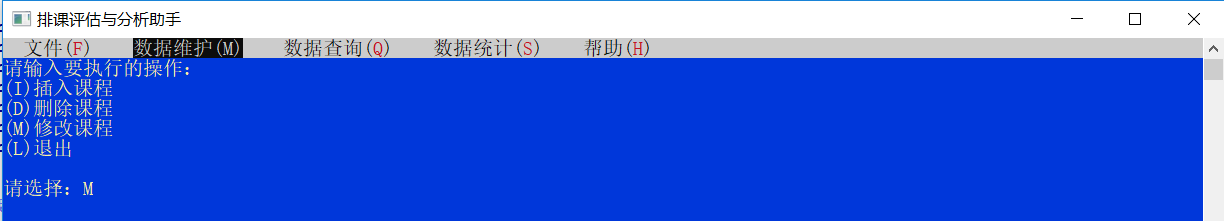


图4-63 执行修改课程信息功能



图4-64 输入要修改信息的课程名，回车

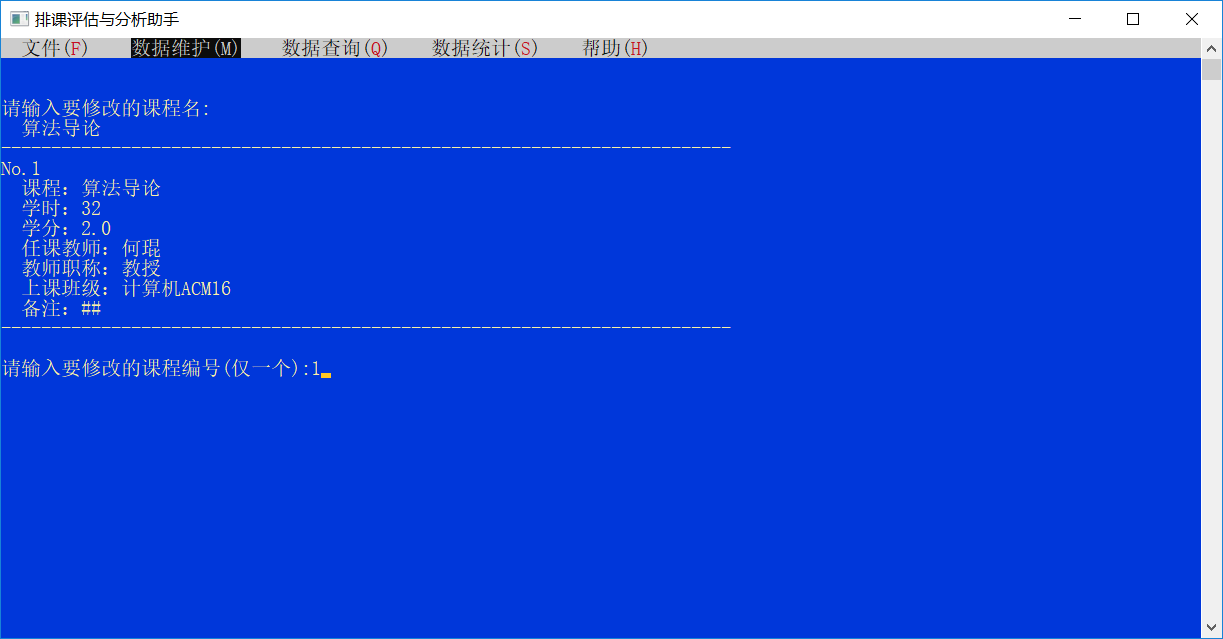


图4-65 输入要修改信息的编号，回车



图4-66 输入新的信息的编号，回车

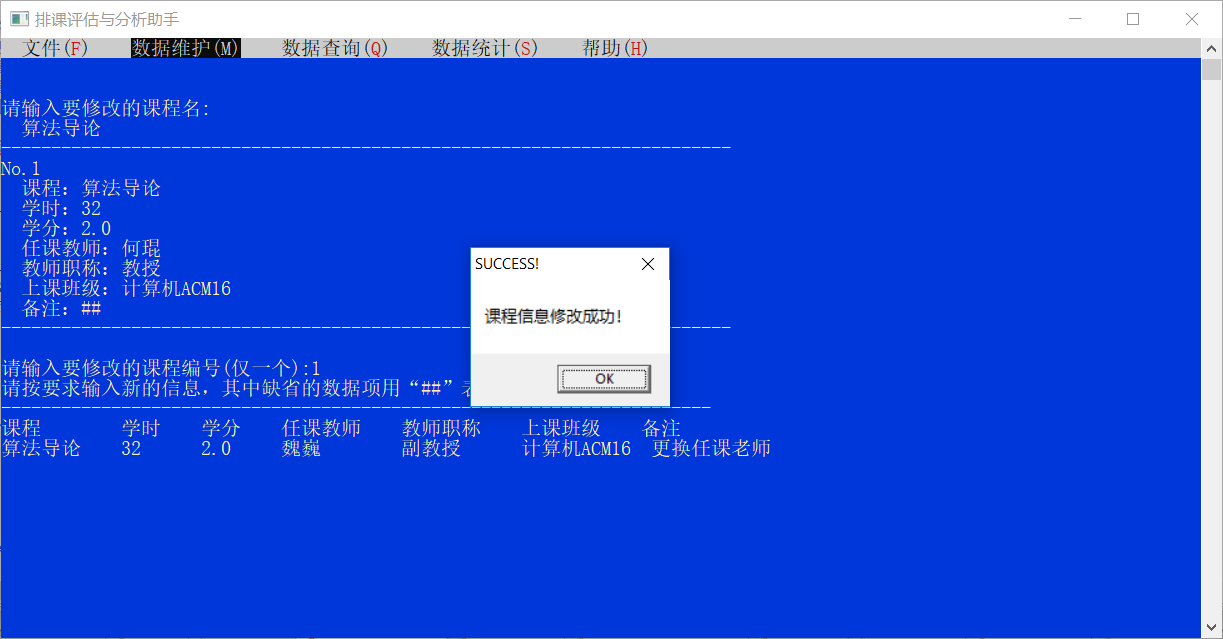


图4-67 修改信息成功



图4-68 按教师查找要刚才修改信息的课程

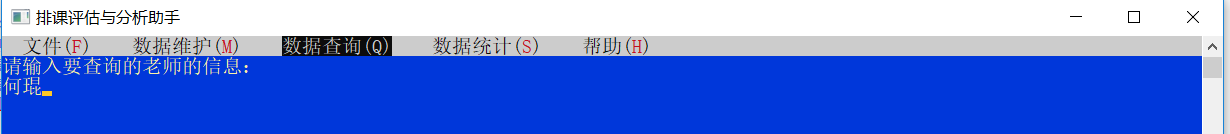


图4-69 输入原来的任课教师

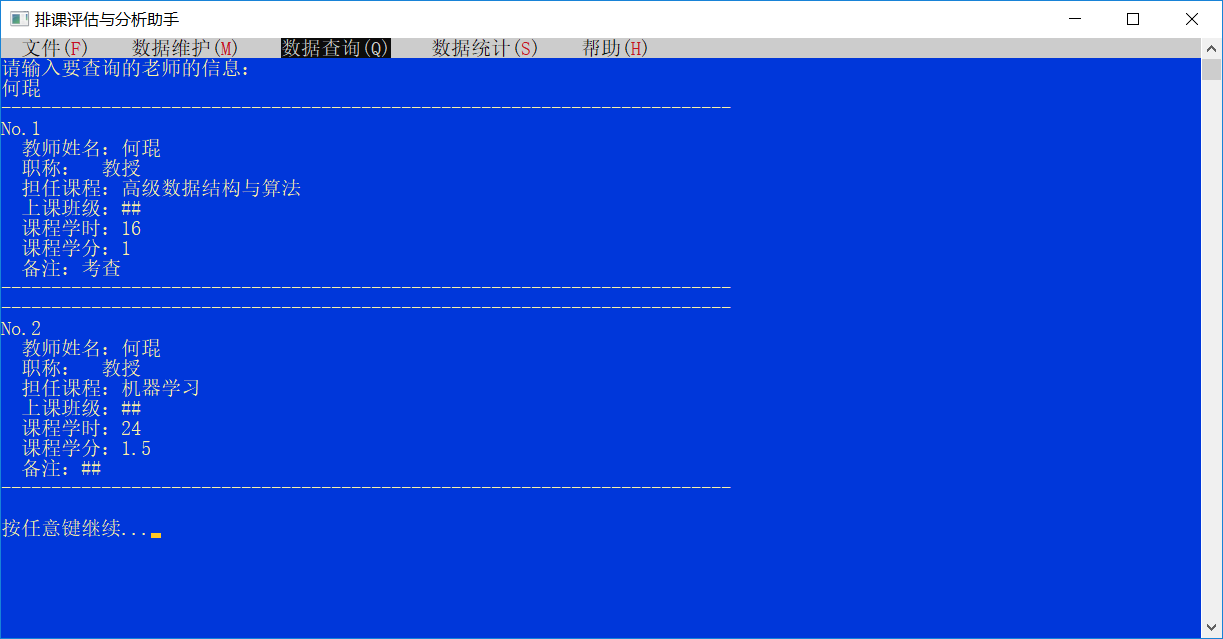


图4-70 刚才被修改的课程已经没有

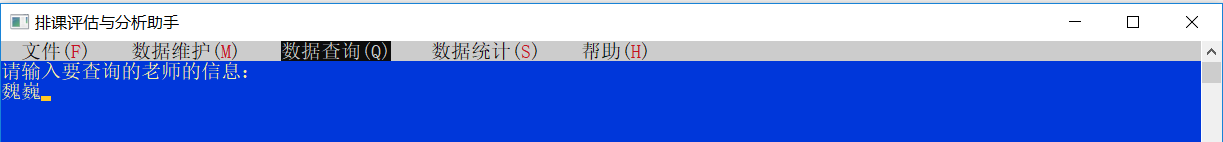


图4-71 输入新的任课教师

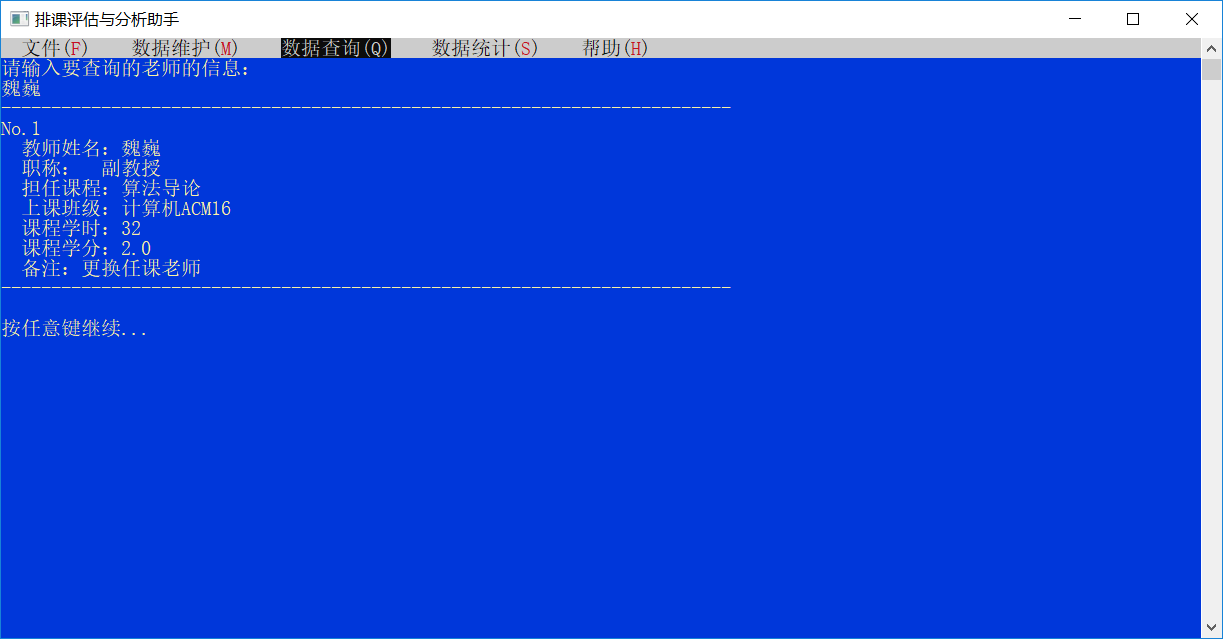


图4-72 发现刚刚修改过的课程信息，说明修改成功

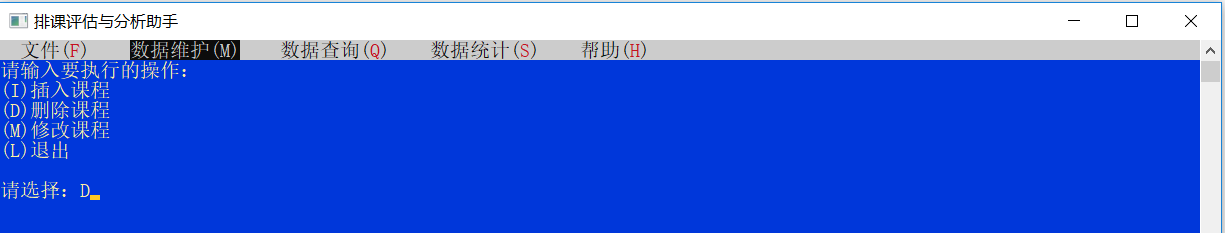


图4-73 执行课程删除功能

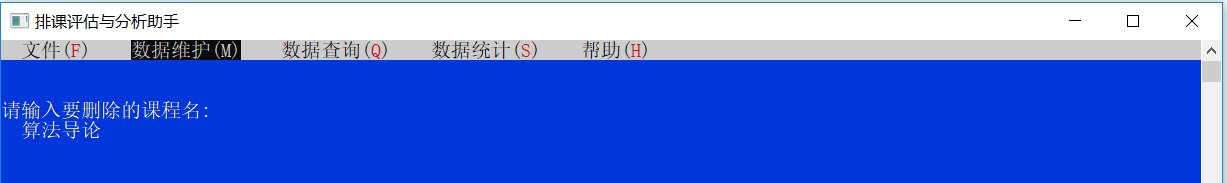


图4-74 输入课程名称

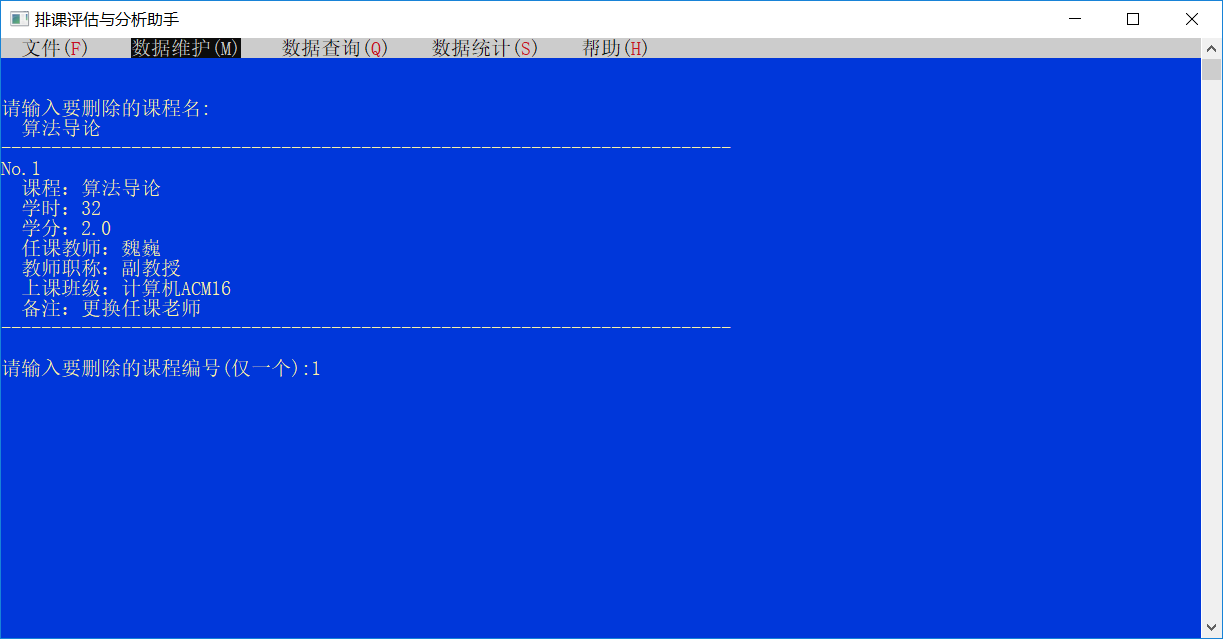


图4-75 输入要删除的课程的编号，回车

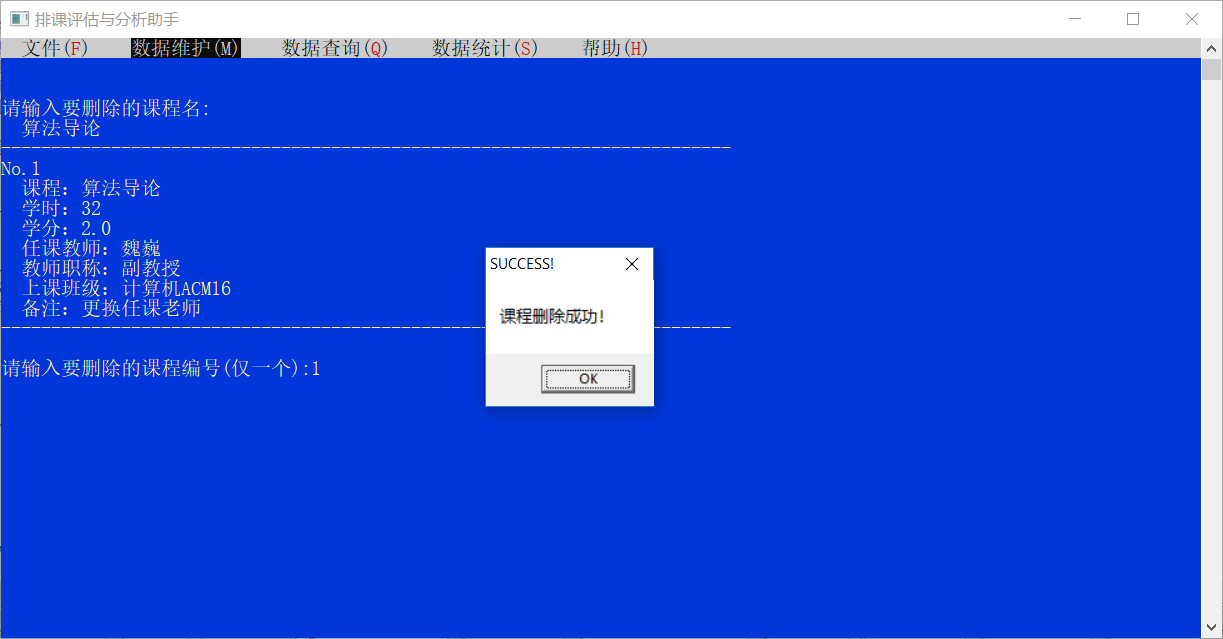


图4-76 删除成功



图4-77 再次查找



图4-78 发现无法找到，说明这门课程已经被删除

1. 执行数据维护中的教室信息维护功能。

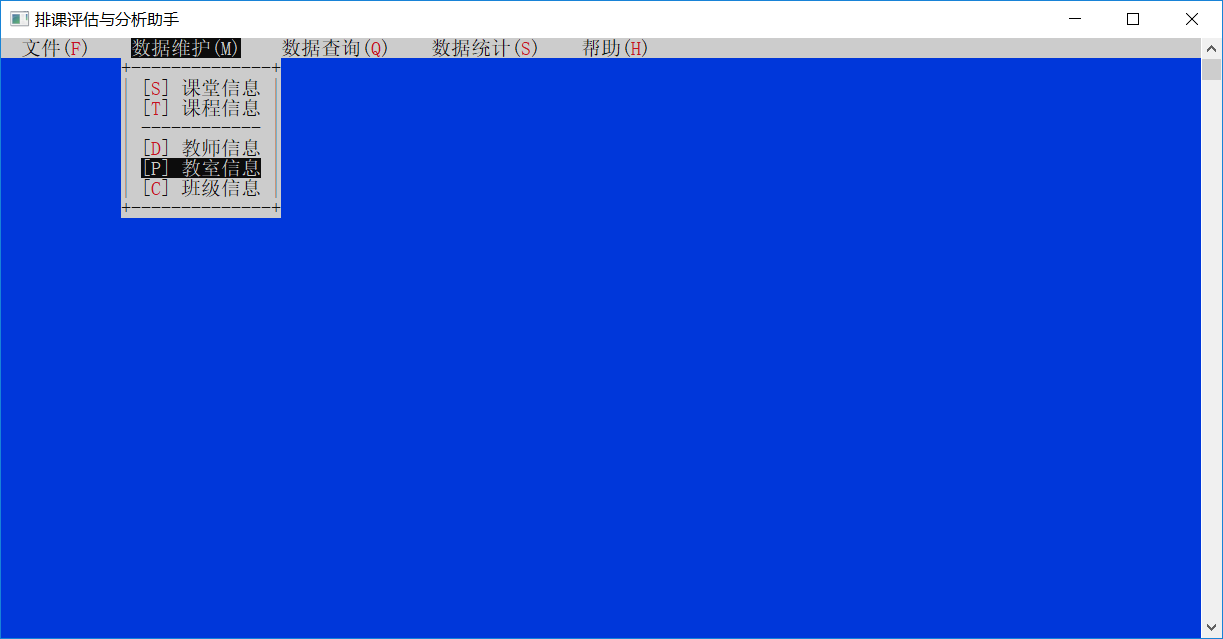


图4-79 执行教室信息修改功能

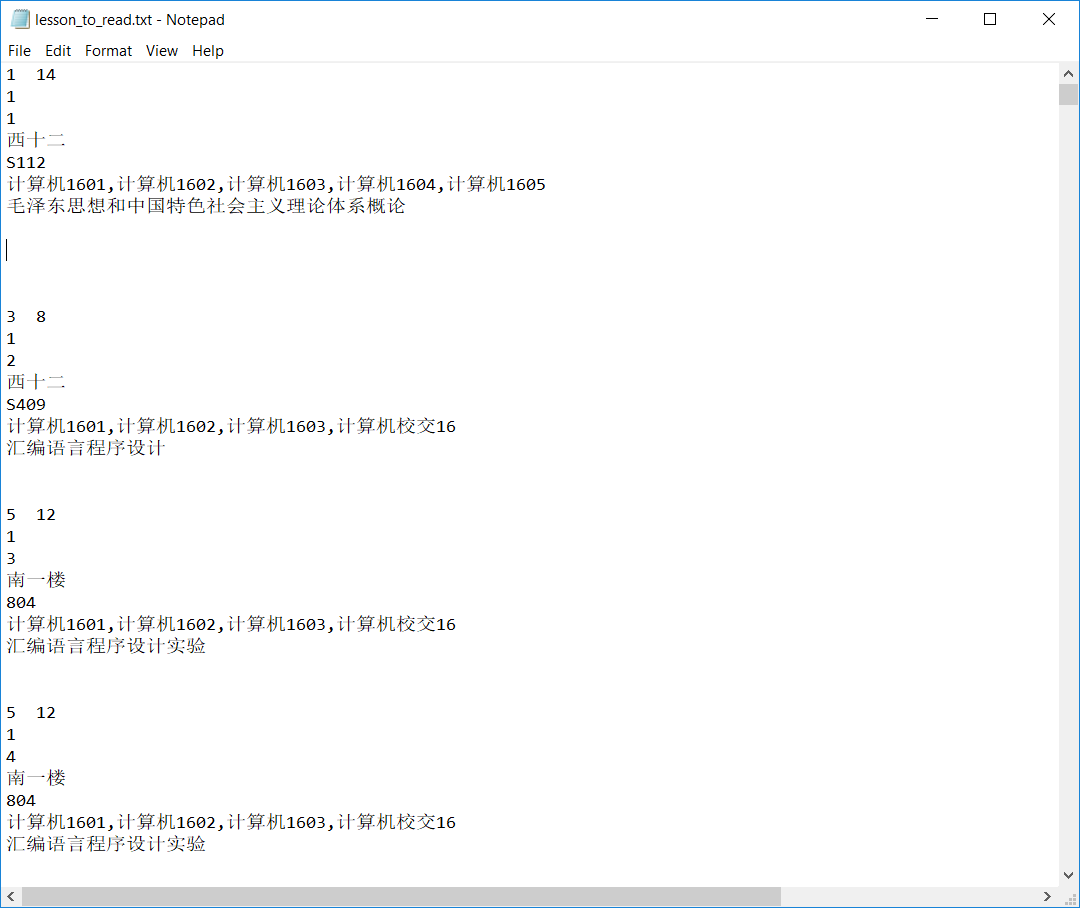


图4-80 弹出配置文件

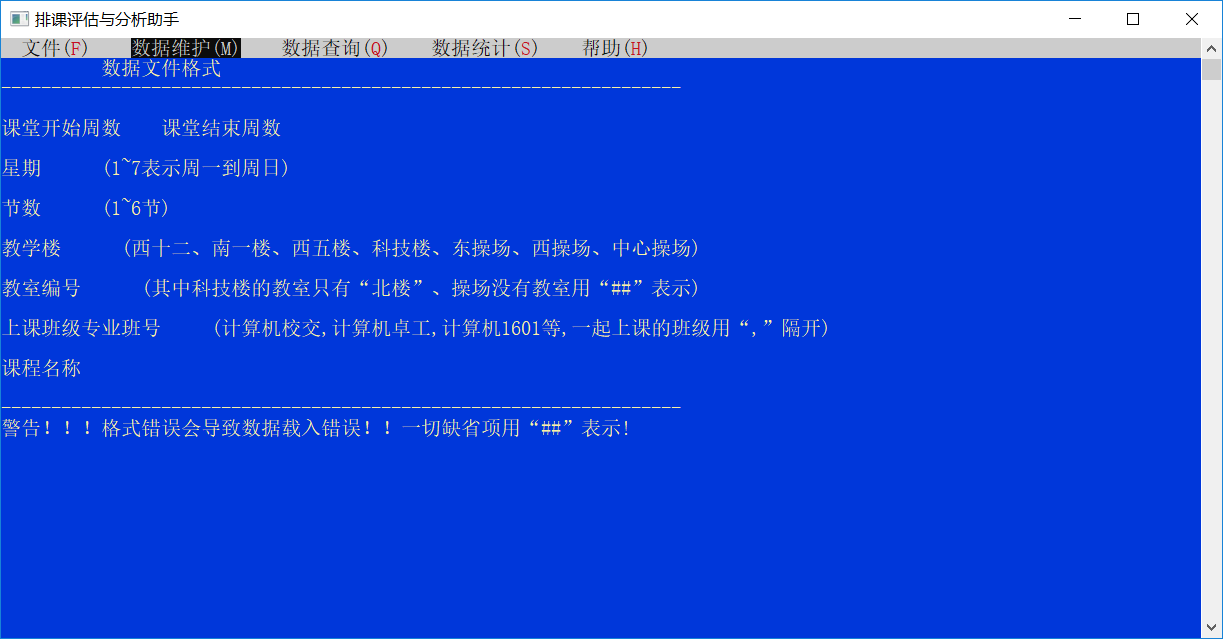


图4-81 程序弹出配置文件格式信息

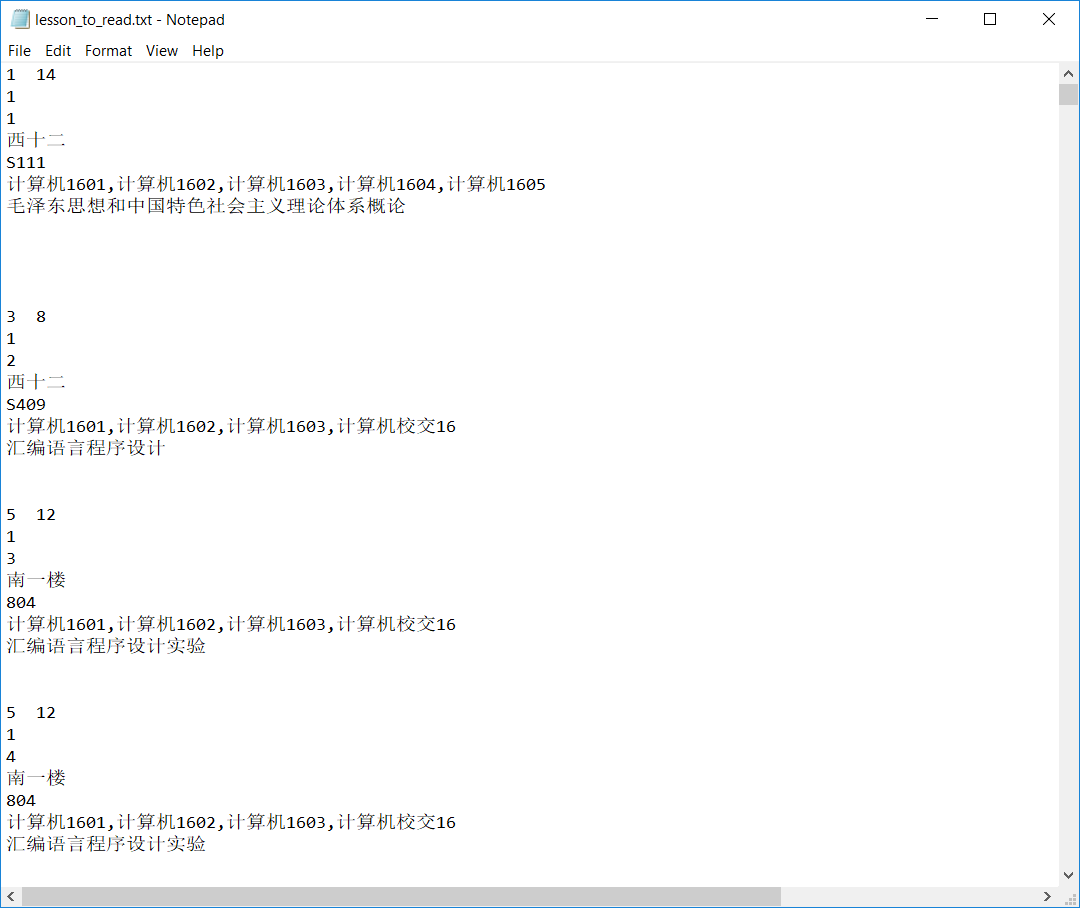


图4-82 将第一门课“毛中特”的教室改成S111

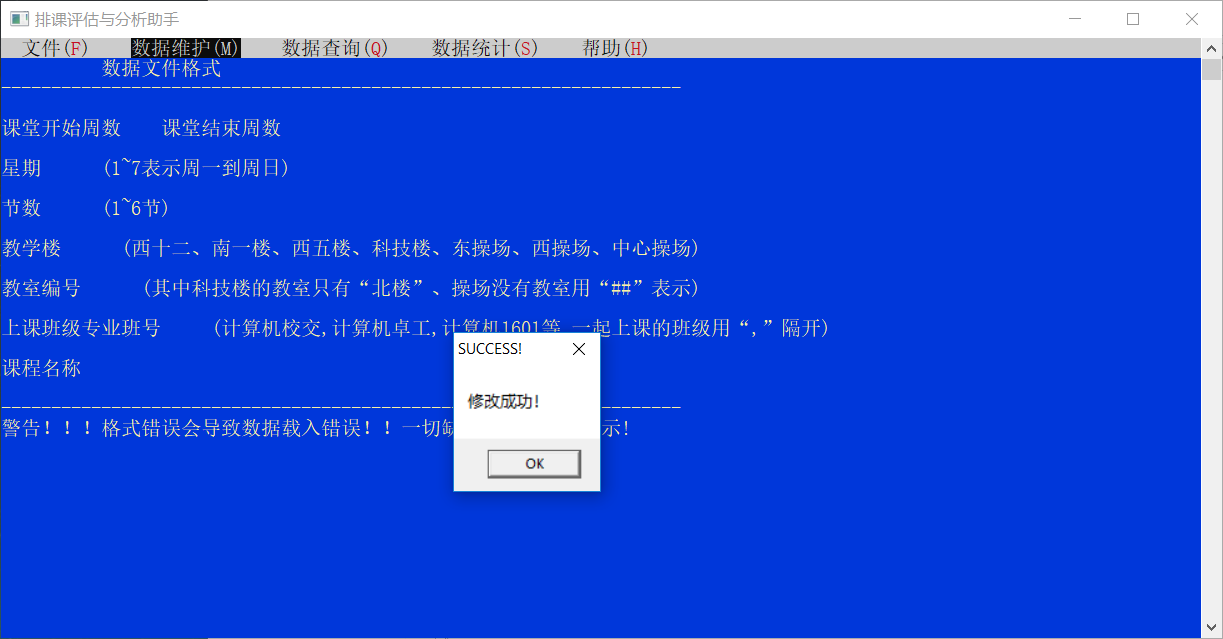


图4-83 修改成功

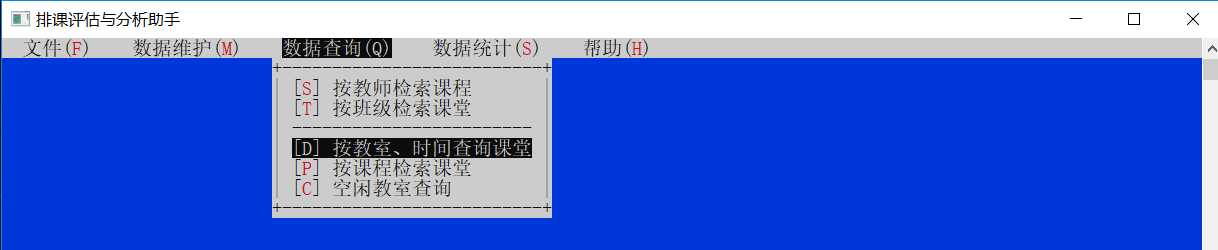


图4-84 按时间地点查找刚才修改的教室

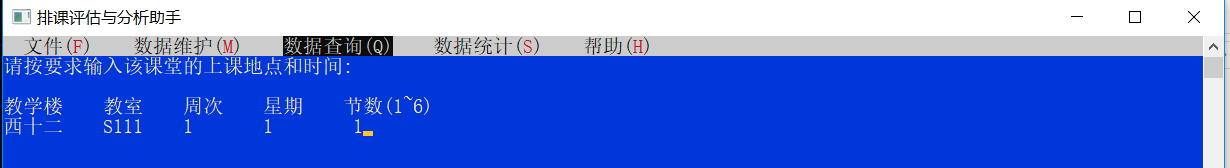


图4-85 输入刚才修改完的新的信息，回车

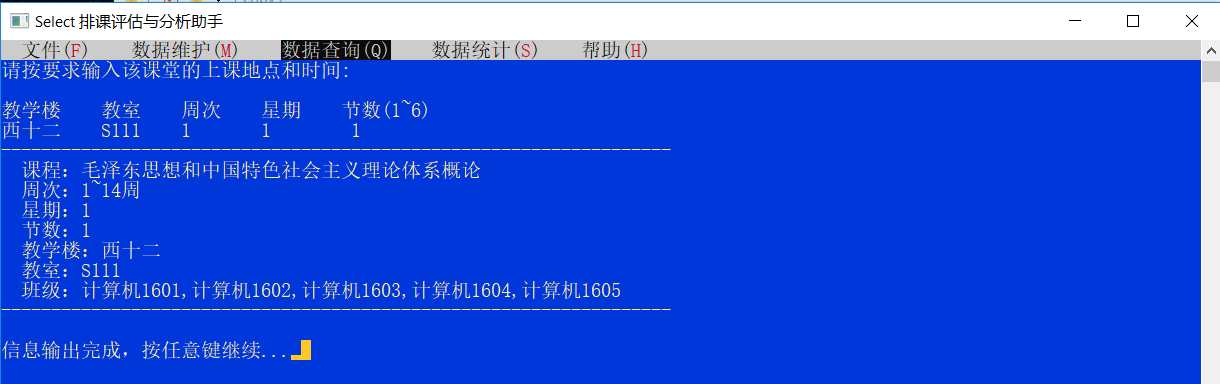


图4-86 发现刚才修改过的信息，说明操作成功

1. 执行信息维护中的班级信息。

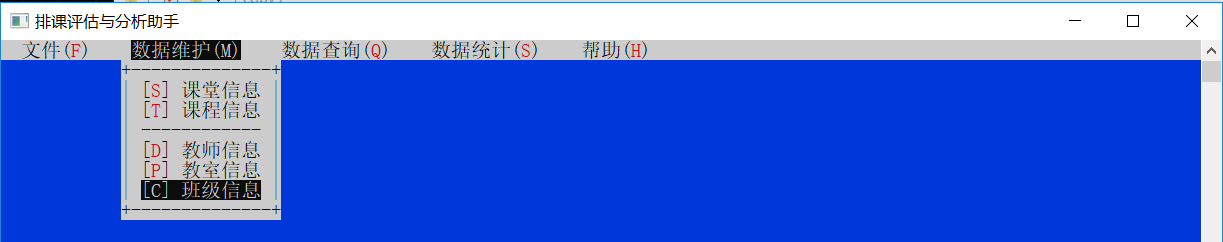


图4-87 执行班级信息维护

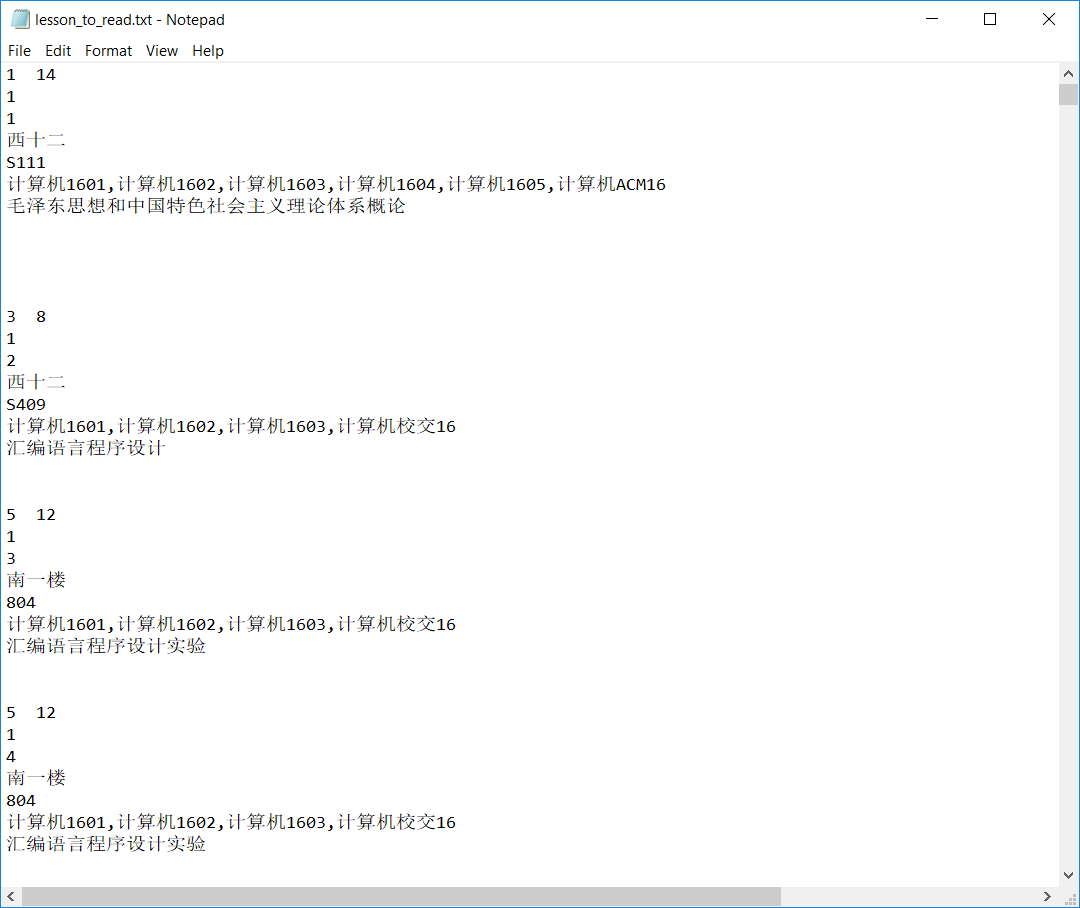


图4-88 在第一门课中加上一个班级“计算机ACM16”



图4-89 在第一门课中加上一个班级“计算机ACM16”

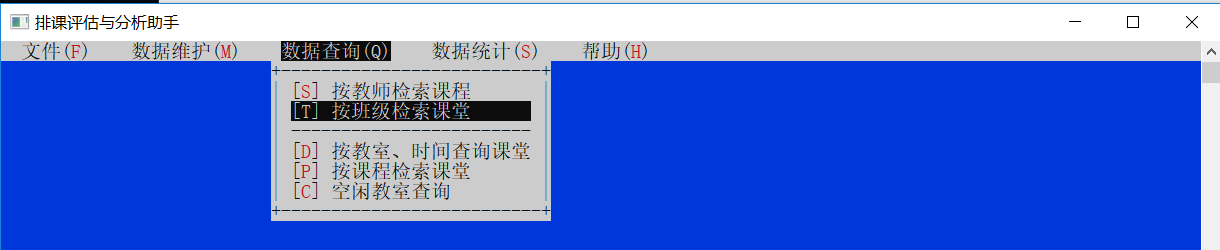


图4-90 按班级检索课堂，查看刚才的修改是否有效

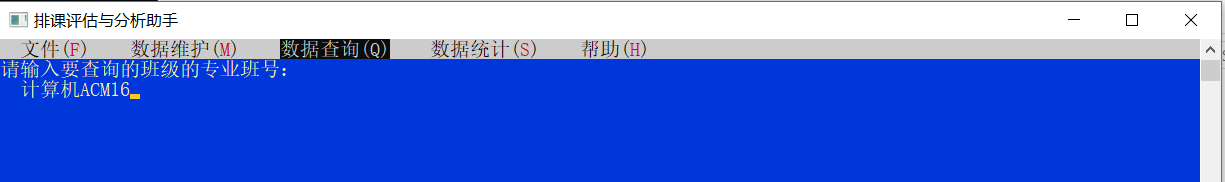


图4-91 输入班级“计算机ACM16”查询

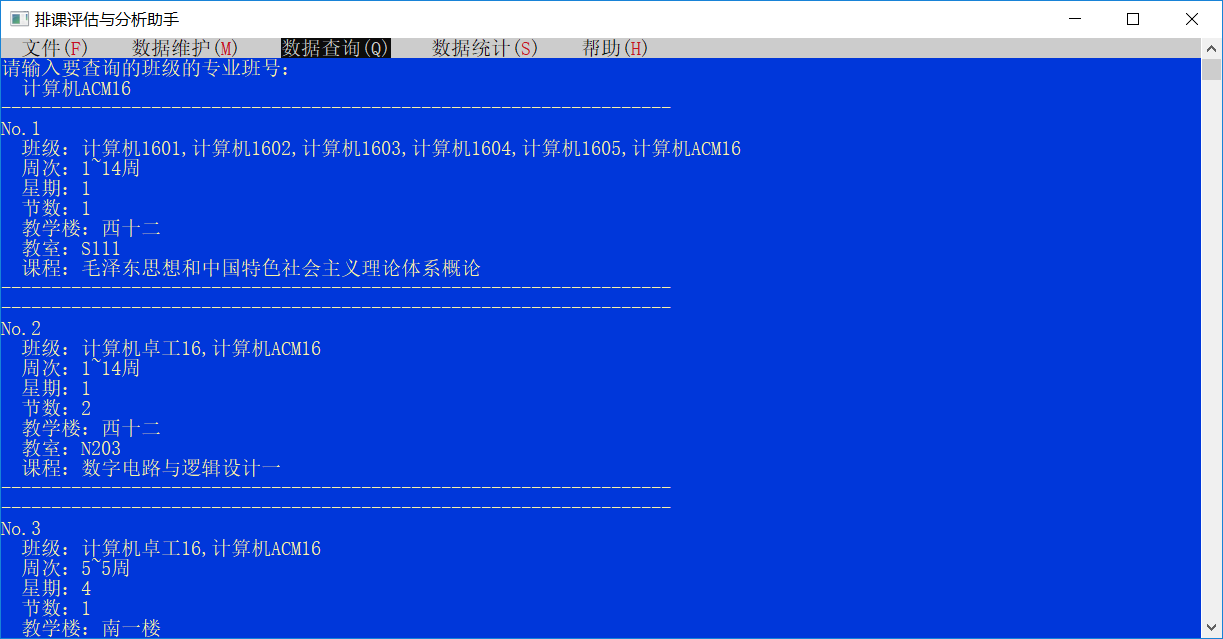


图4-92 发现第一个就是我们刚才修改的那个数据元素，说明操作成功

分析结果：整个数据维护模块的所有功能执行完毕，增加、删除、修改功能正确执行。所有增加，删除，修改操作的结果都能被查询到。一方面说明这些数据的维护功能执行正确，另一方面说明查询功能能够正确执行。

1. 数据查询模块

功能：根据关键字查询或者检索排课相关信息

设计目标：能够正确输出原有或者已经被修改的信息

备注：本模块的功能在数据维护模块中基本已经演示过了，在此再次简要演示一遍。

运行结果：

1. 按教师检索课程。输入教师名，输出与该教师相关的所有课程信息。



图4-93 按教师检索课程

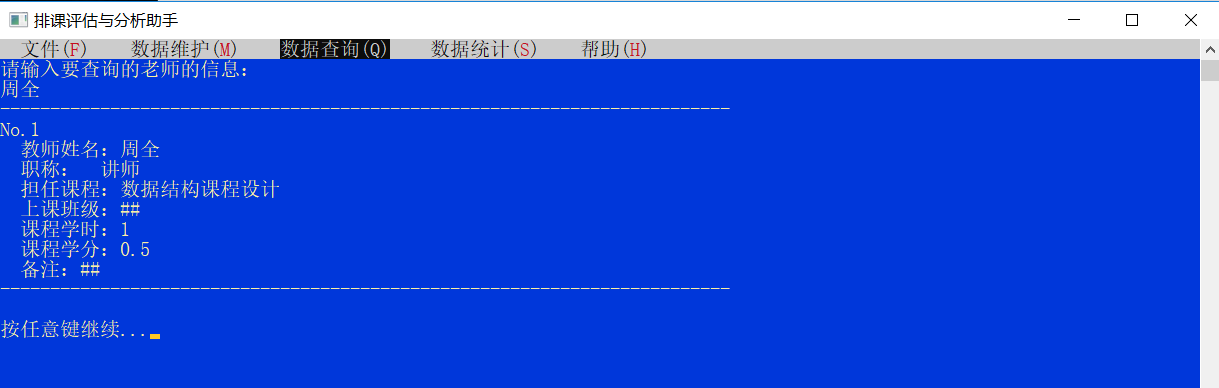


图4-94 输入教师名“周全”回车,显示结果

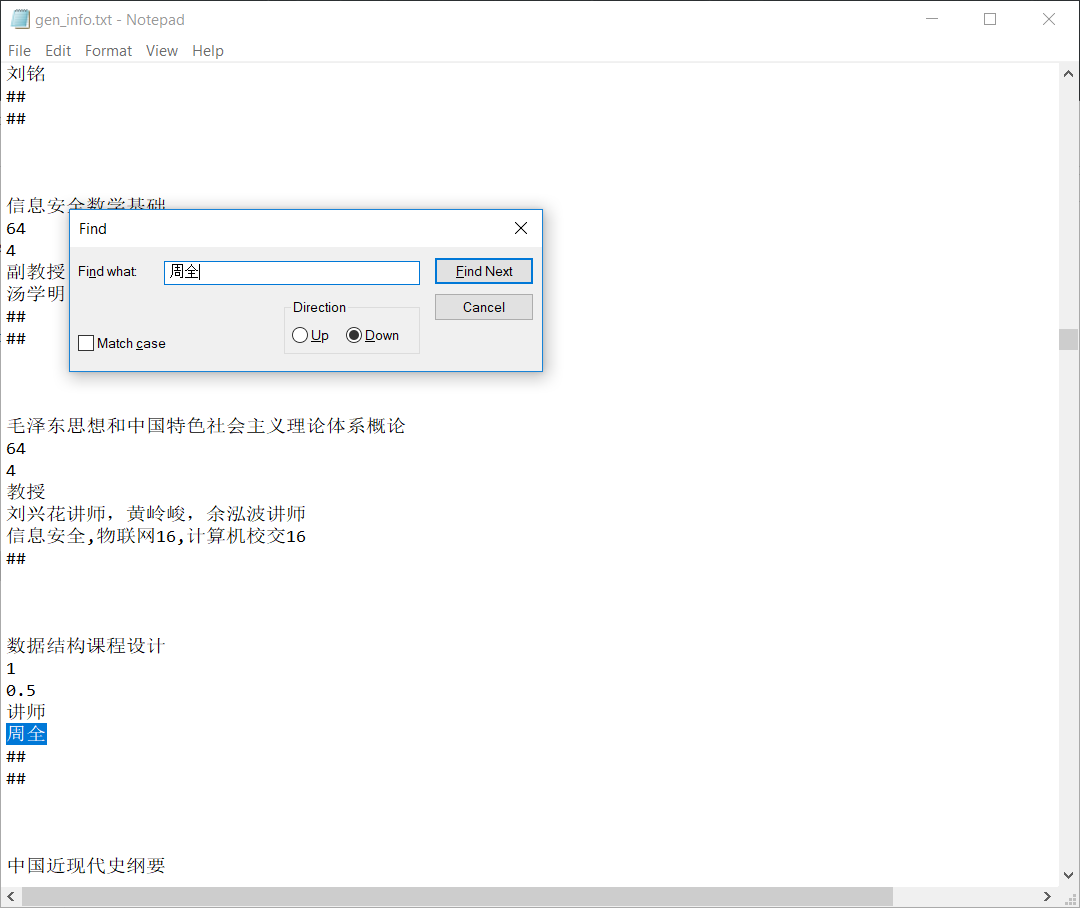


图4-95 在数据文件中查找“周全”，发现信息和程序中一样，说明文件读入功能和查询功能均正确

（2）按班级检索课堂

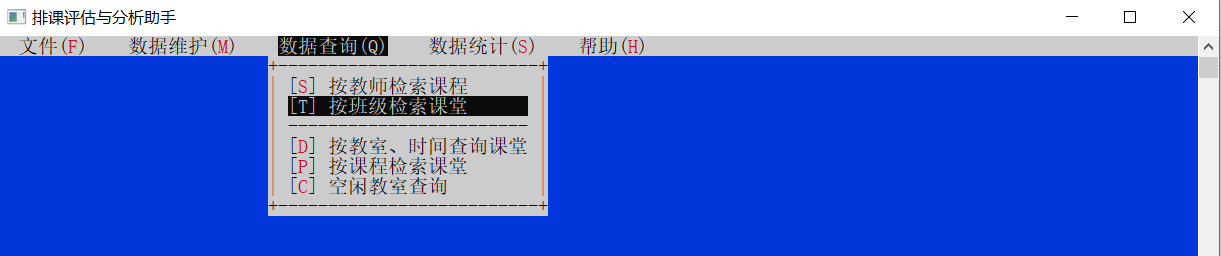


图4-96 执行按班级检索课堂



图4-97 输入“计算机1601”回车，显示与该班级相关的所有课堂

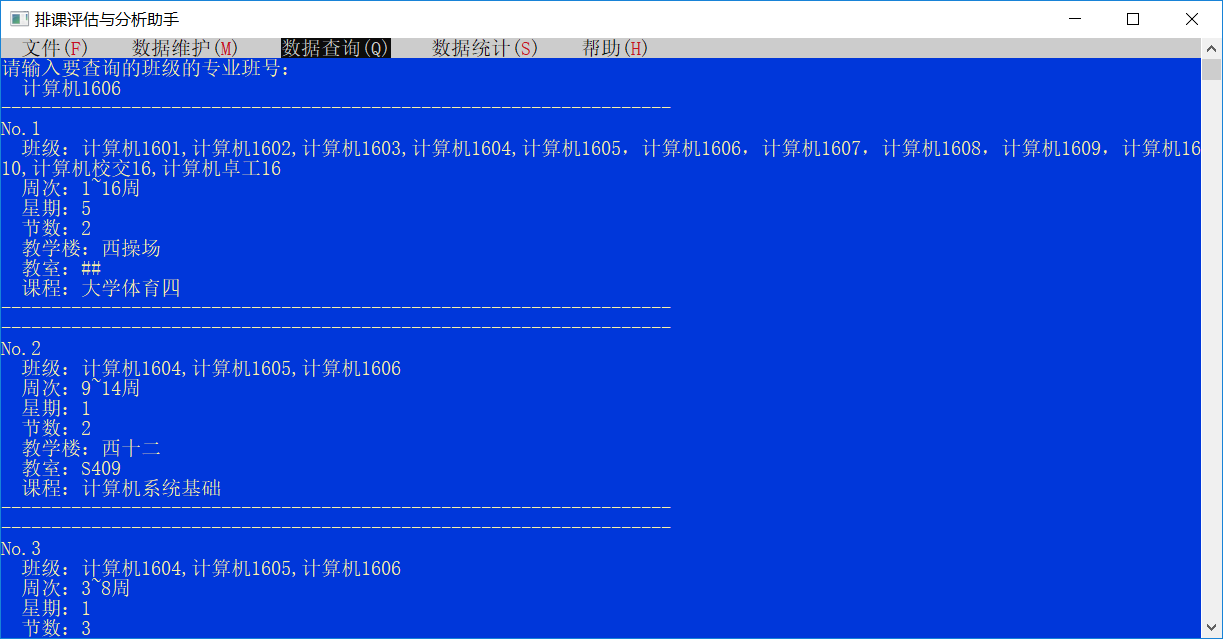


图4-98 输入“计算机1606”回车，显示与该班级相关的所有课堂

（3）按教室、时间查询课堂

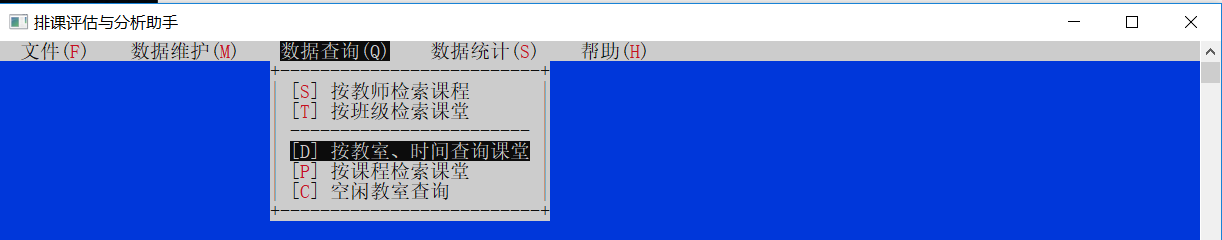


图4-99 按教师、时间查询课堂

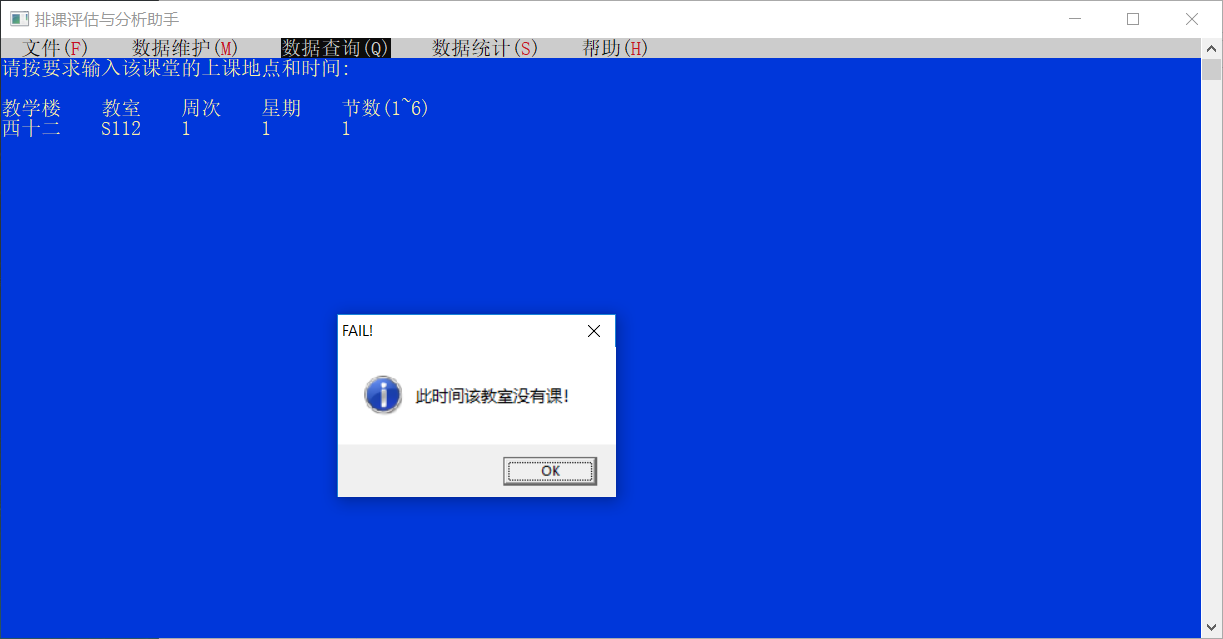


图4-100 输入第1周星期1第1节的西十二S112，程序提示没有课

（在教室信息维护中已把此时间地点的课调到了S111）

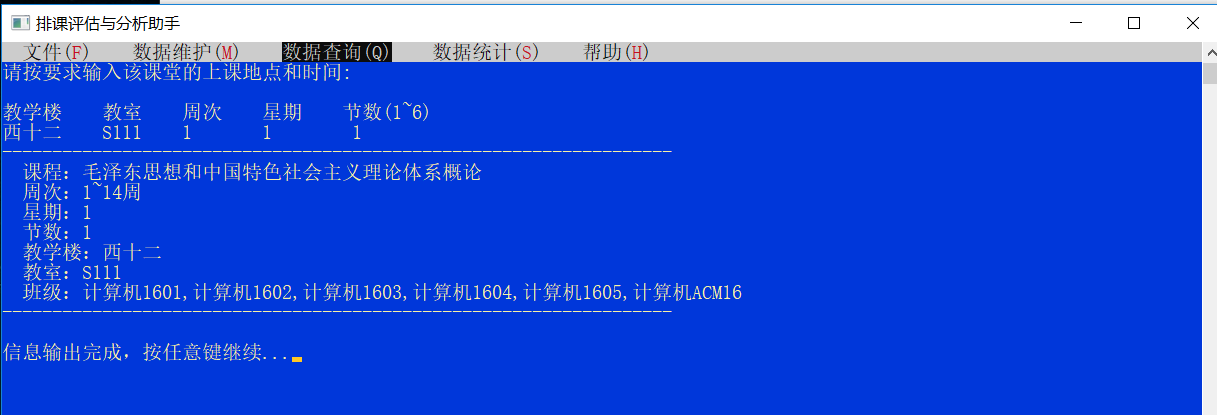


图4-101 输入第1周星期1第1节的西十二S111，显示出相关课程

（3）按课程检索课堂



图4-102 按课程检索课堂



图4-103 输入要查询的课程名，回车

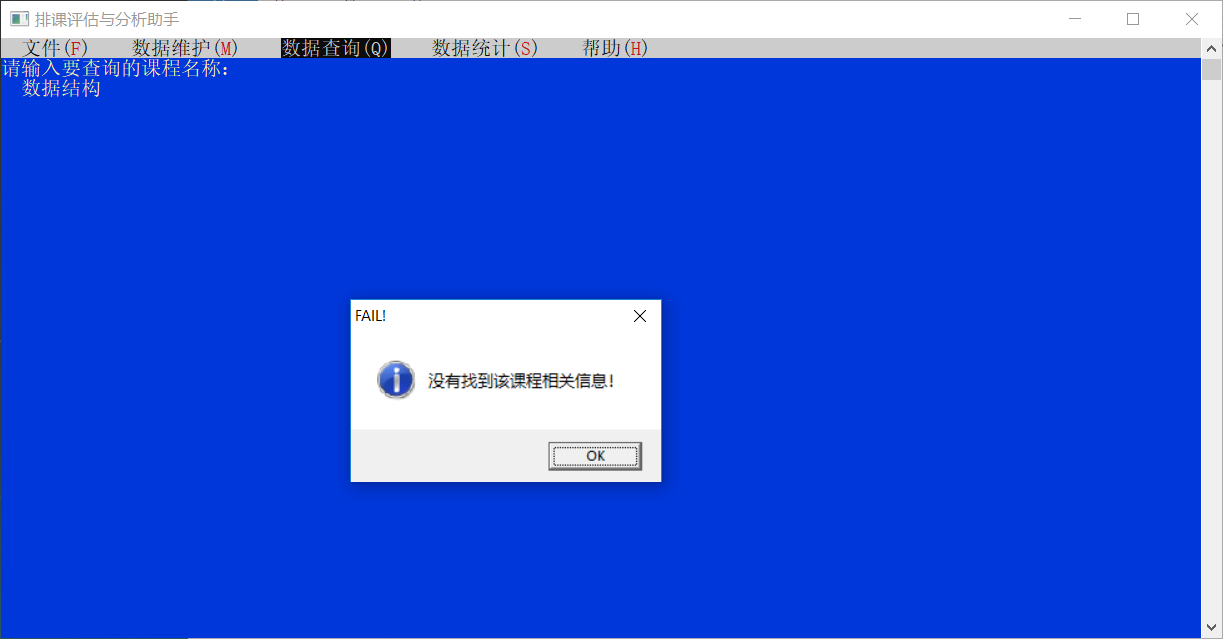


图4-104 没有查询到（下学期没有数据结构这门课）

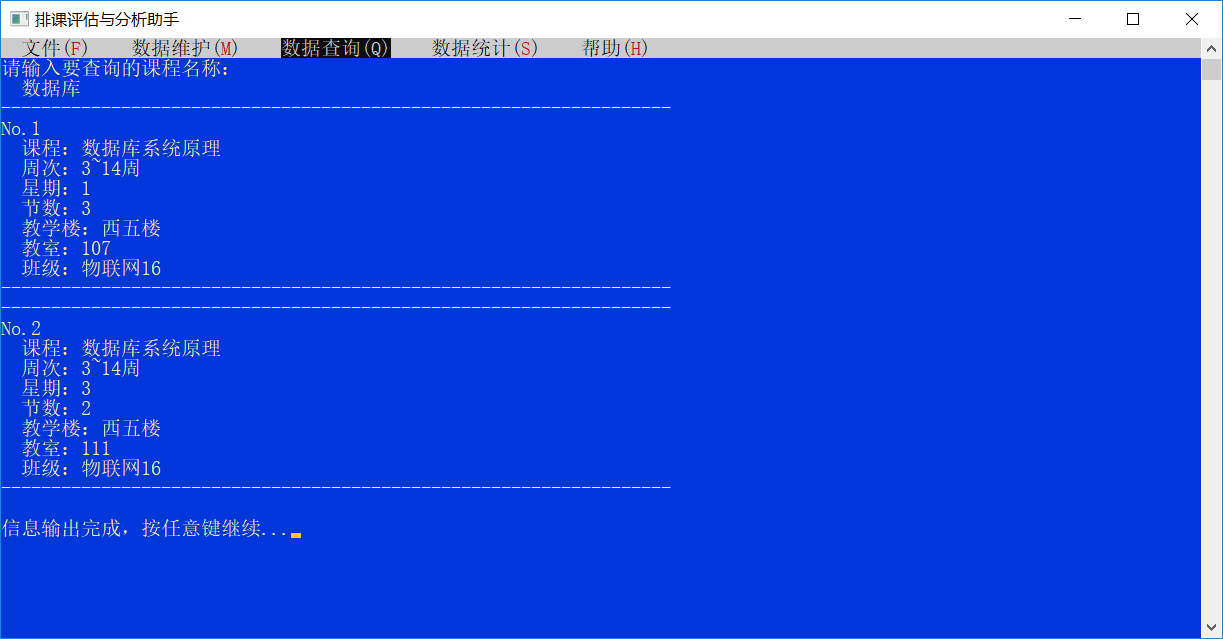


图4-105 输入关键字“数据库”，输出相关的课堂

（4）空闲教室查询

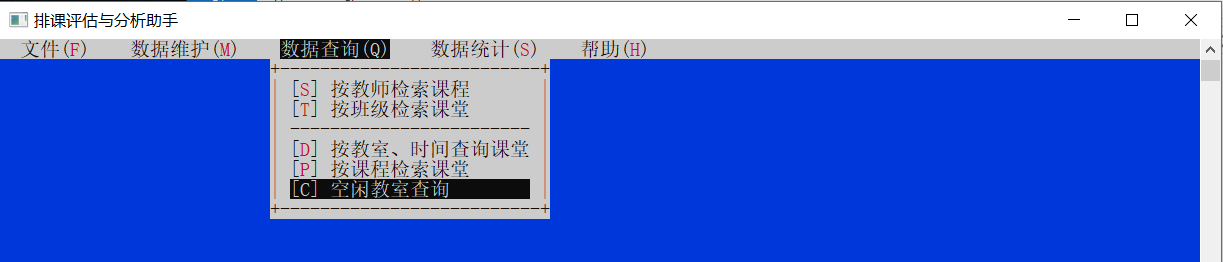


图4-106 空闲教室查询

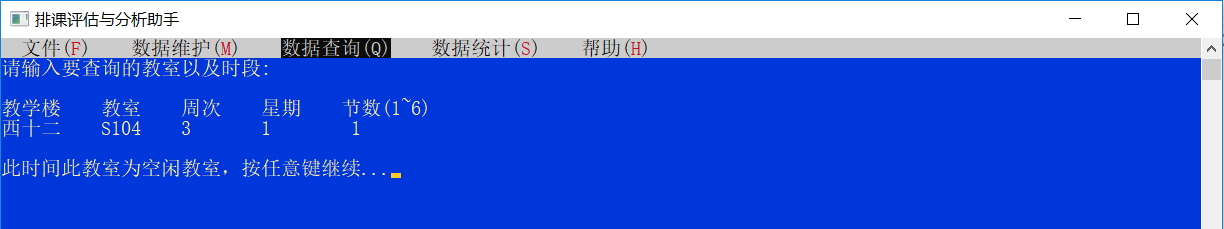


图4-107 输入要查询的教室，以及时间，输出结果，当前查询的教室在该时间为空闲教室

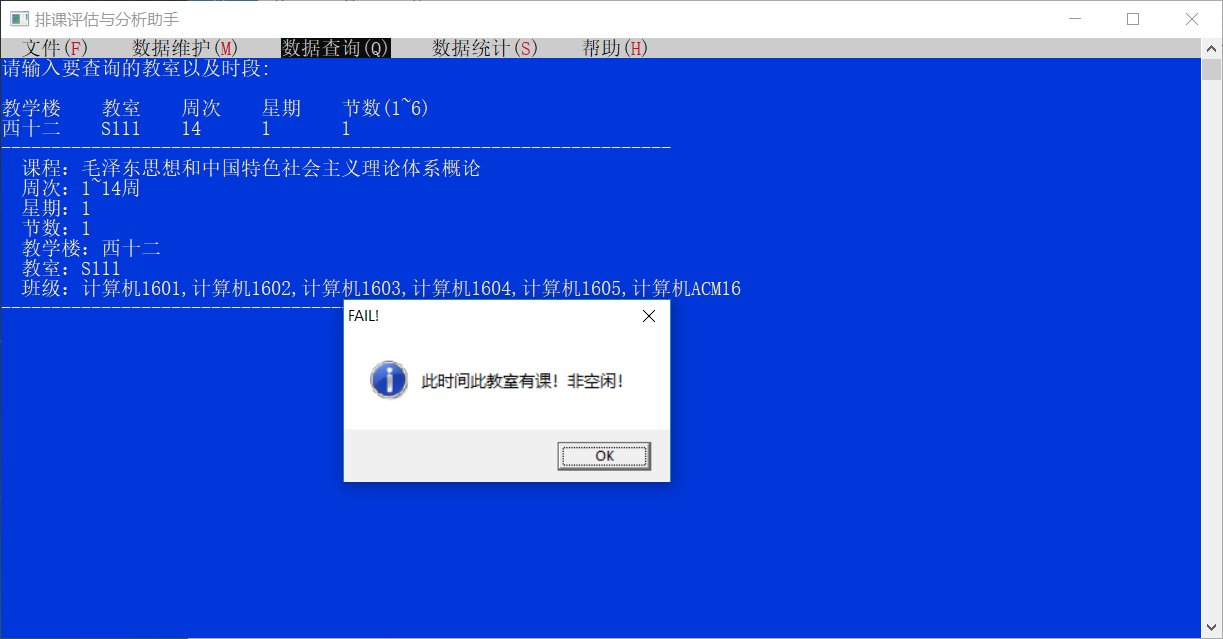


图4-108 该时间此教室有课，非空闲，输出提示信息，并输出该时间该教室的课堂

分析结果：全部查询检索功能执行正确，能够输出正确的结果，对于不存在的数据，能够给出用户提示，输出出错信息。

1. 数据统计模块

功能：对排课信息进行统计分析，包括教室利用与能效分析，教师任课分析，班级上课分析等。

设计目标：对排课信息进行统计分析，输出统计结果，并给与评估。

运行结果：

（1）教室利用与能效分析

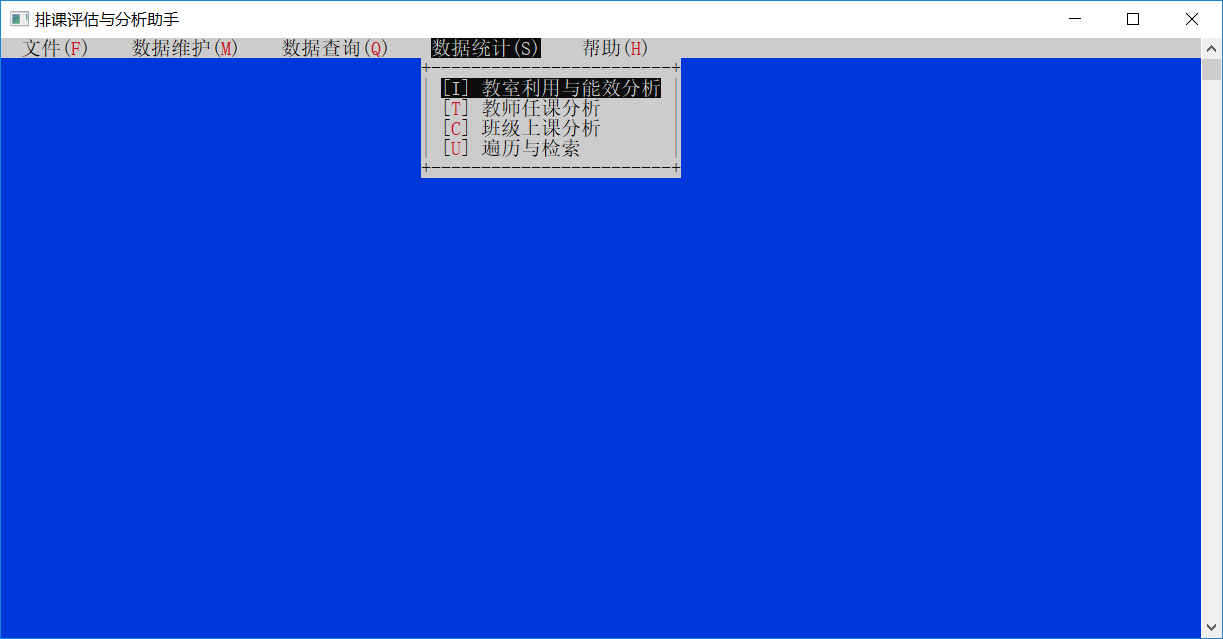


图4-109 教室利用与能效分析

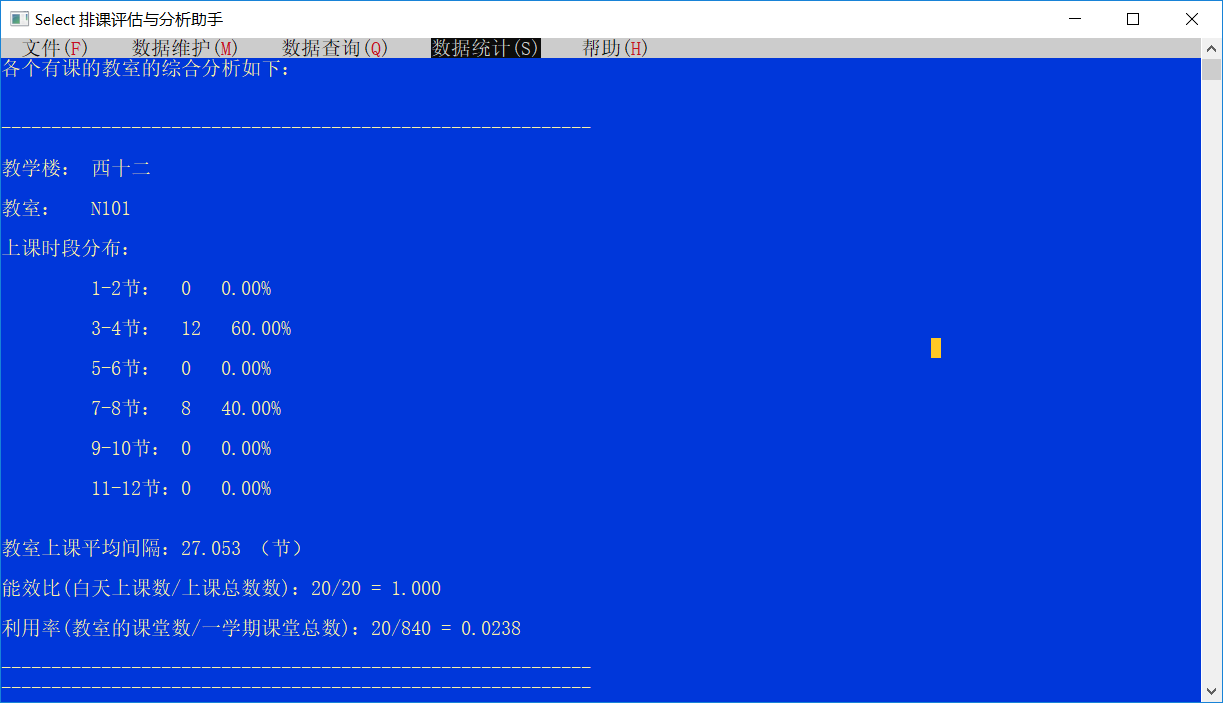


图4-110 依次输出每个教室单独的数据统计分析结果

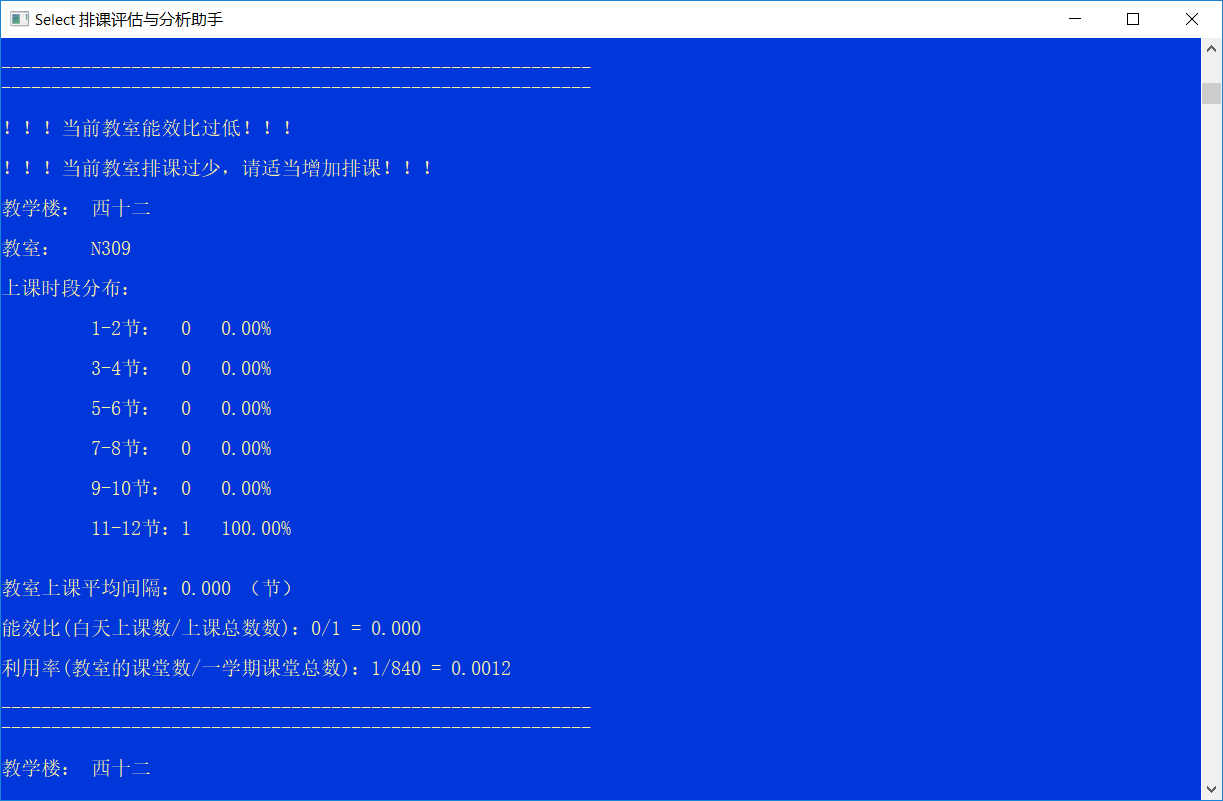


图4-111 当某个教室的某项指标异常时候，给出分析评估

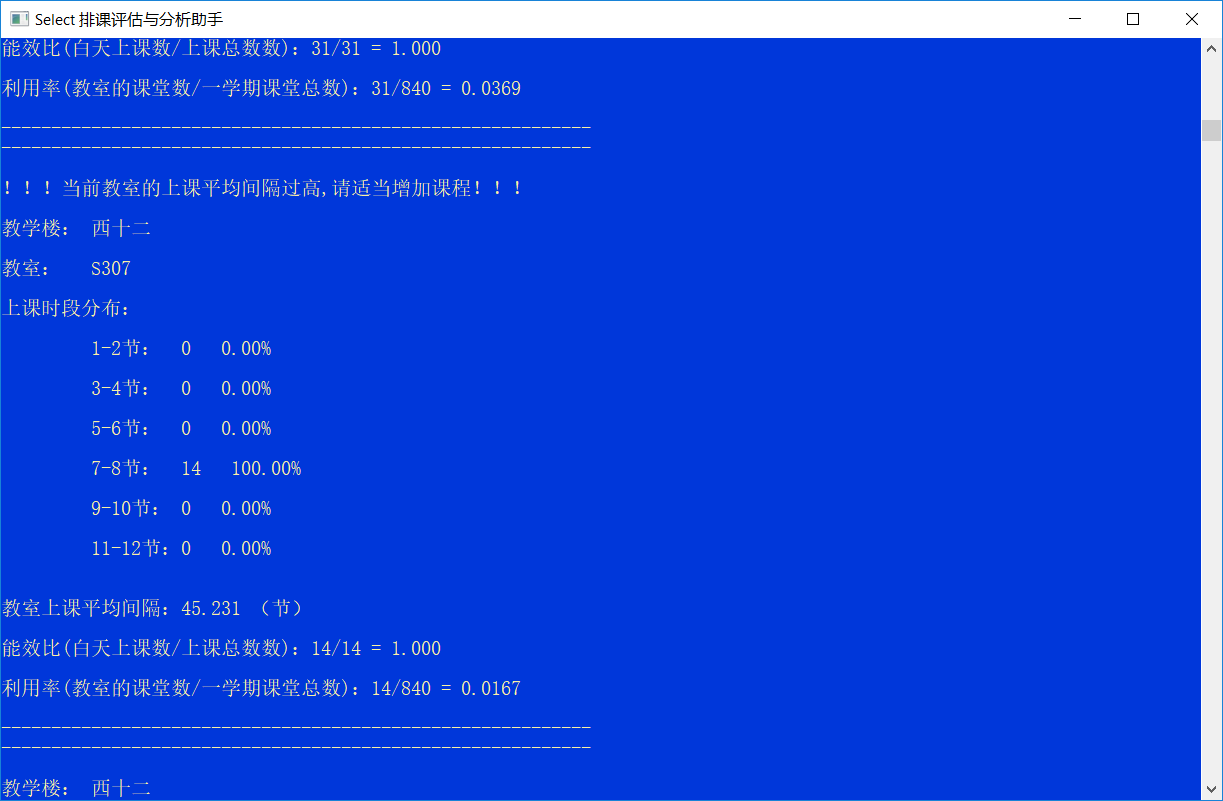


图4-112 当某个教室的某项指标异常时候，给出分析评估

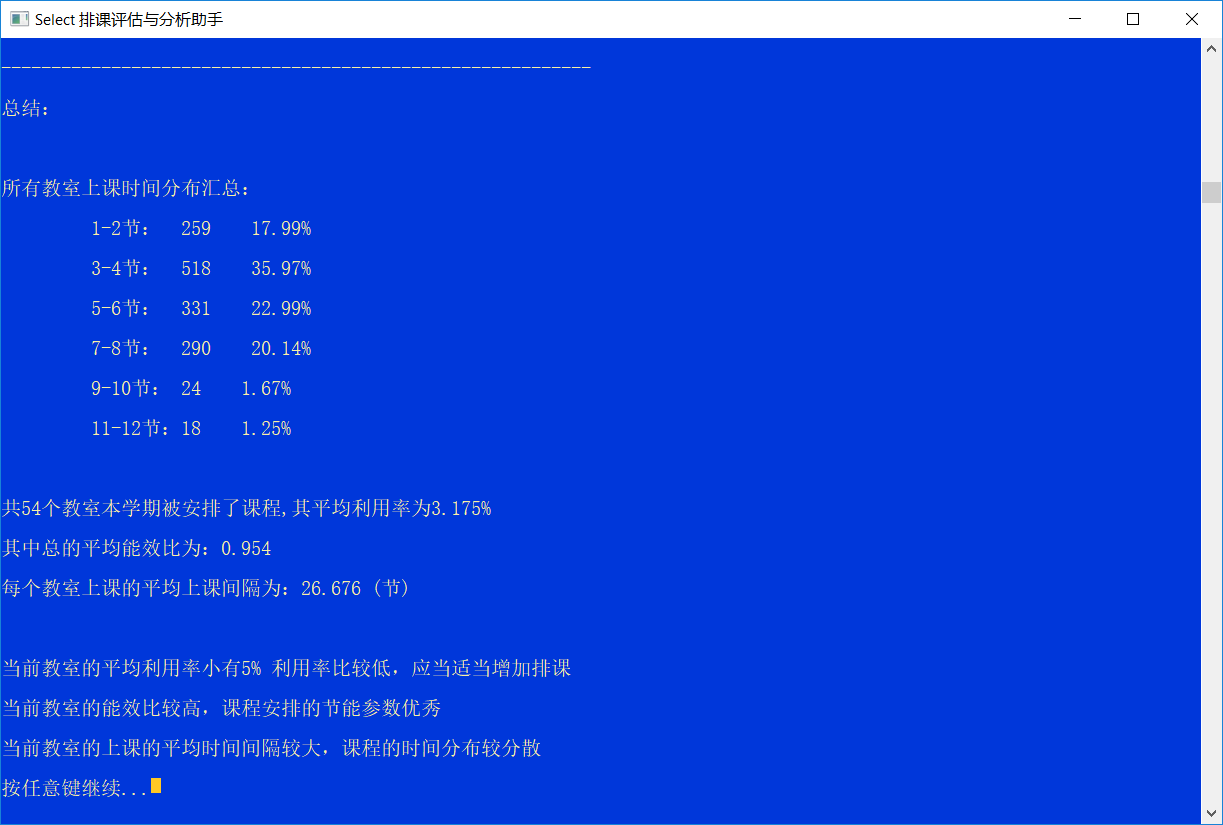


图4-113 在最后输出总结性统计信息，并给出分析、评估与建议

（2）教师任课分析

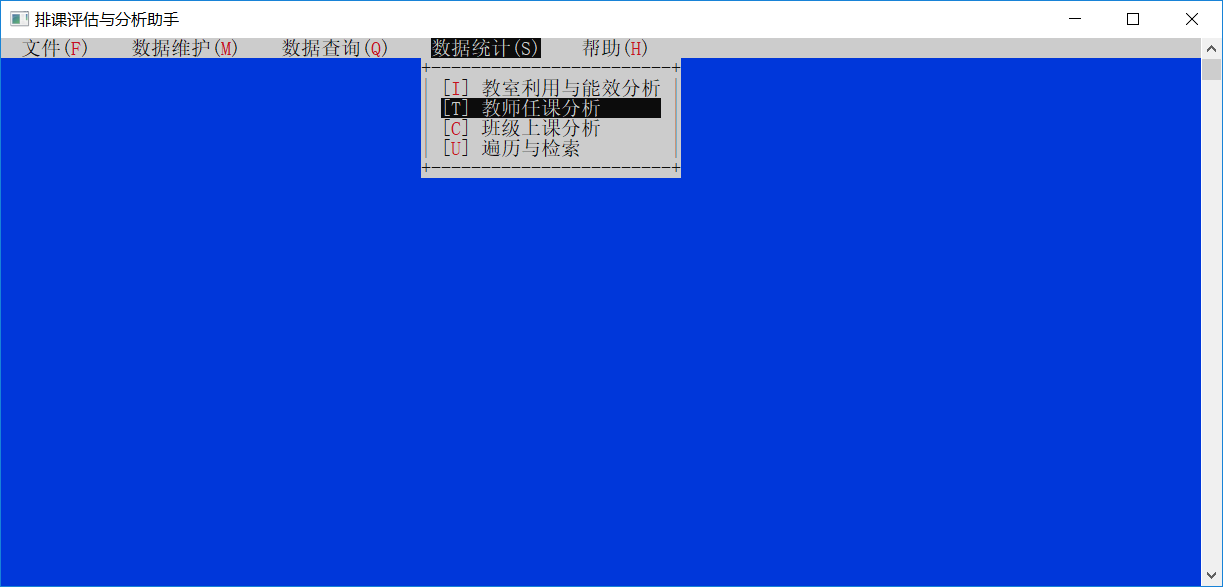


图4-114 教室任课分析

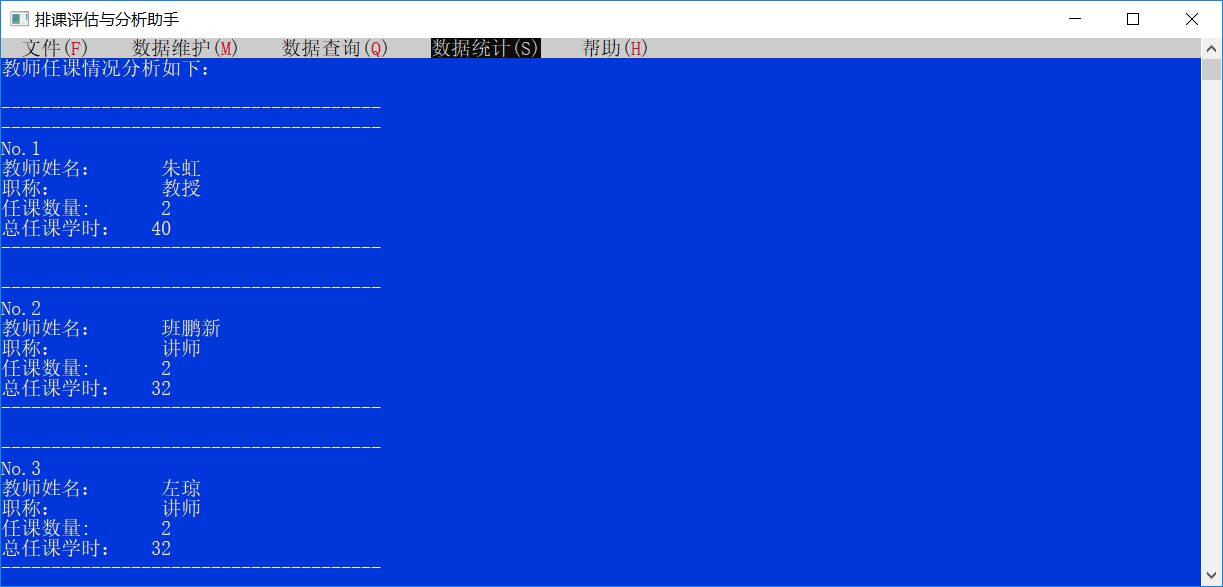


图4-115 输出每个教师的任课情况

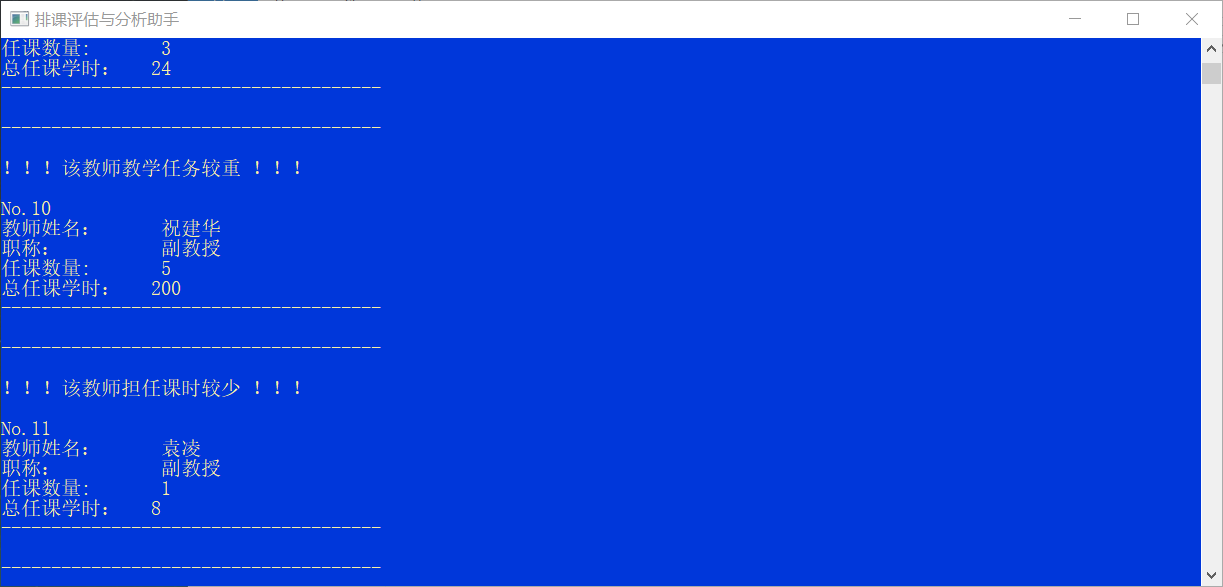


图4-116 当某个教师的某个指标超标的时候，给出分析与提示

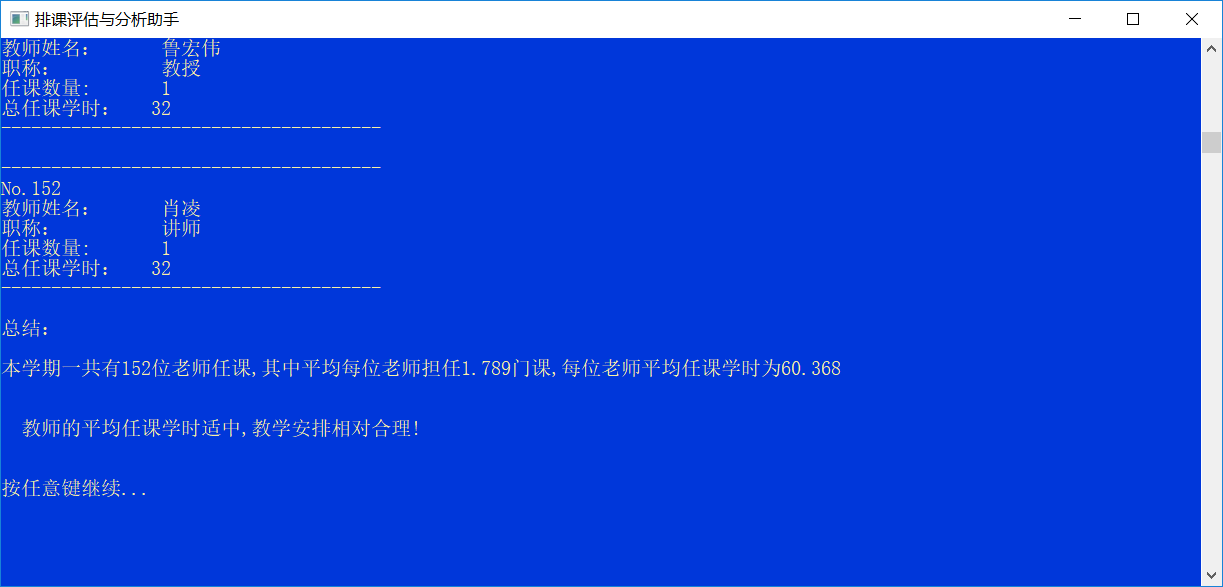


图4-117 最后输出总结性统计结果，并给出评估

（3）班级上课分析

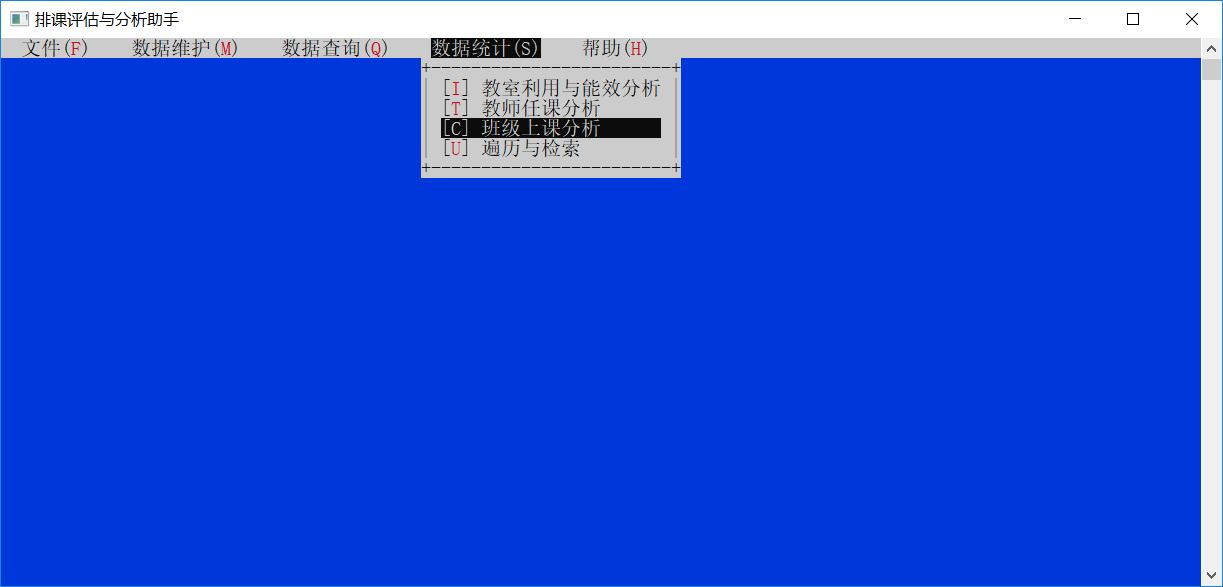


图4-118 班级上课分析

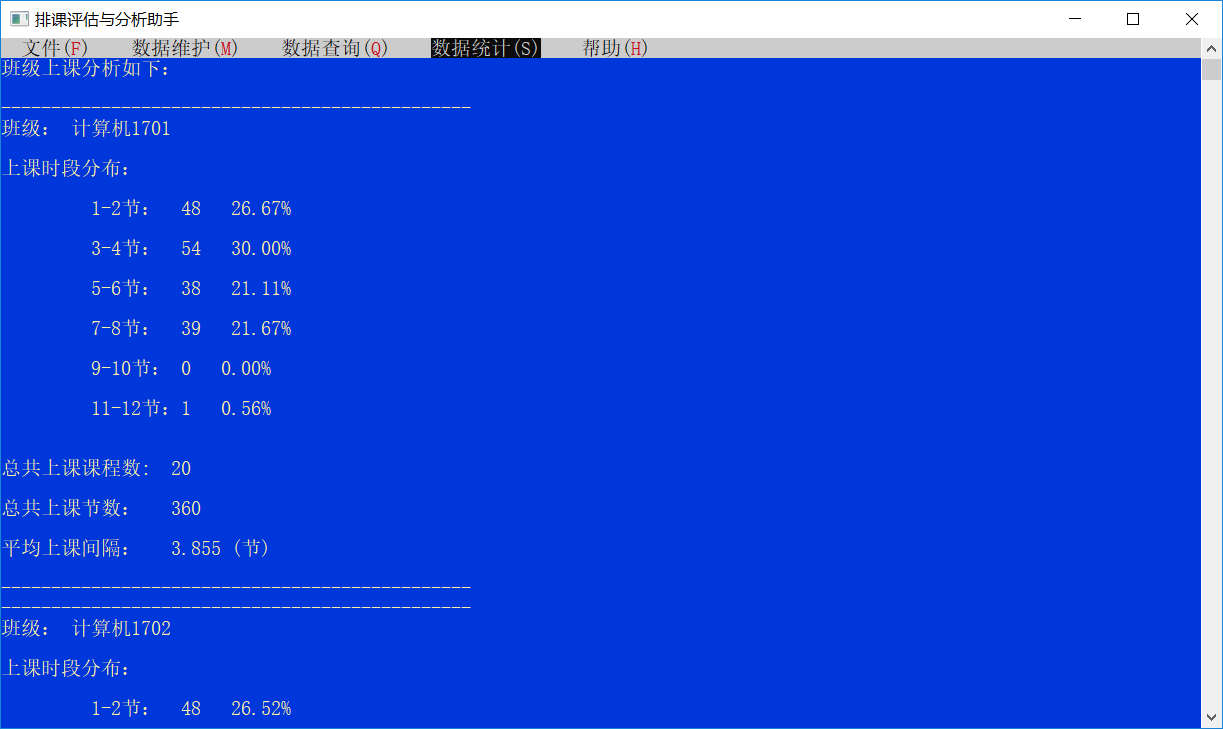


图4-119 输出每个班级的上课时间分布统计，平均上课间隔，总上课节数，课程数等统计信息

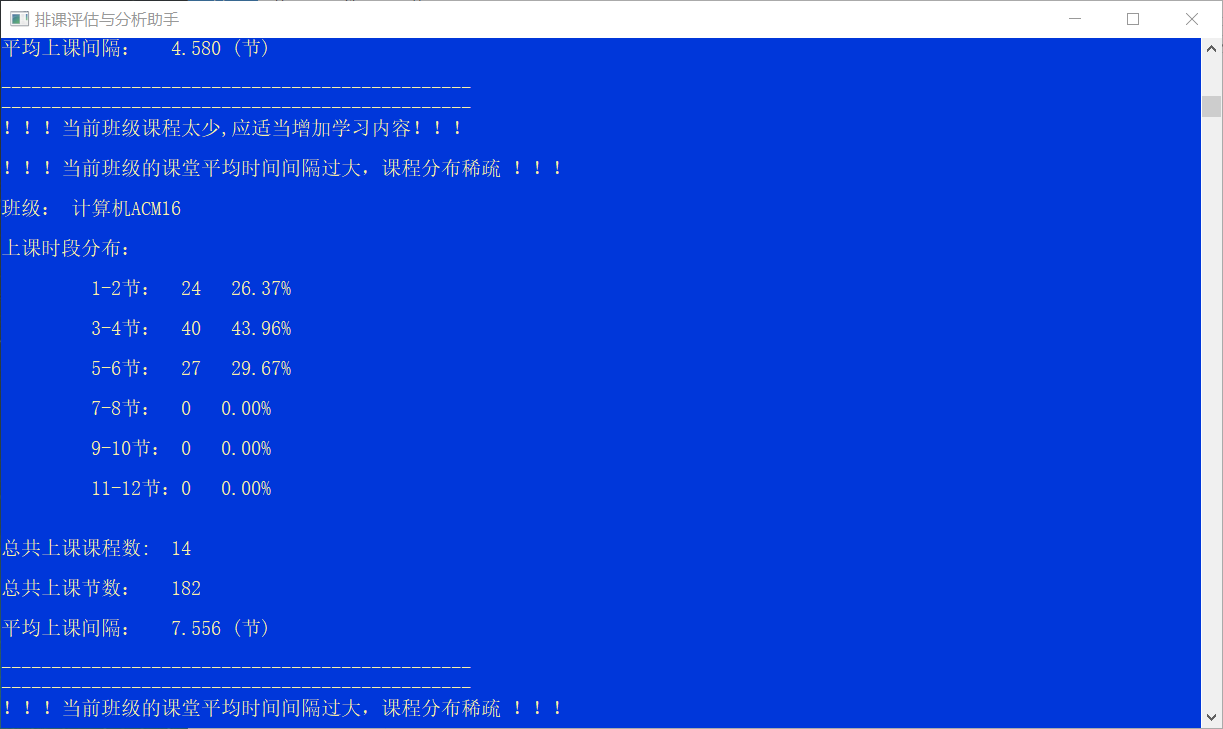


图4-120 当某个班级的排课参数失衡的时候，输出评估结果，并给出建议

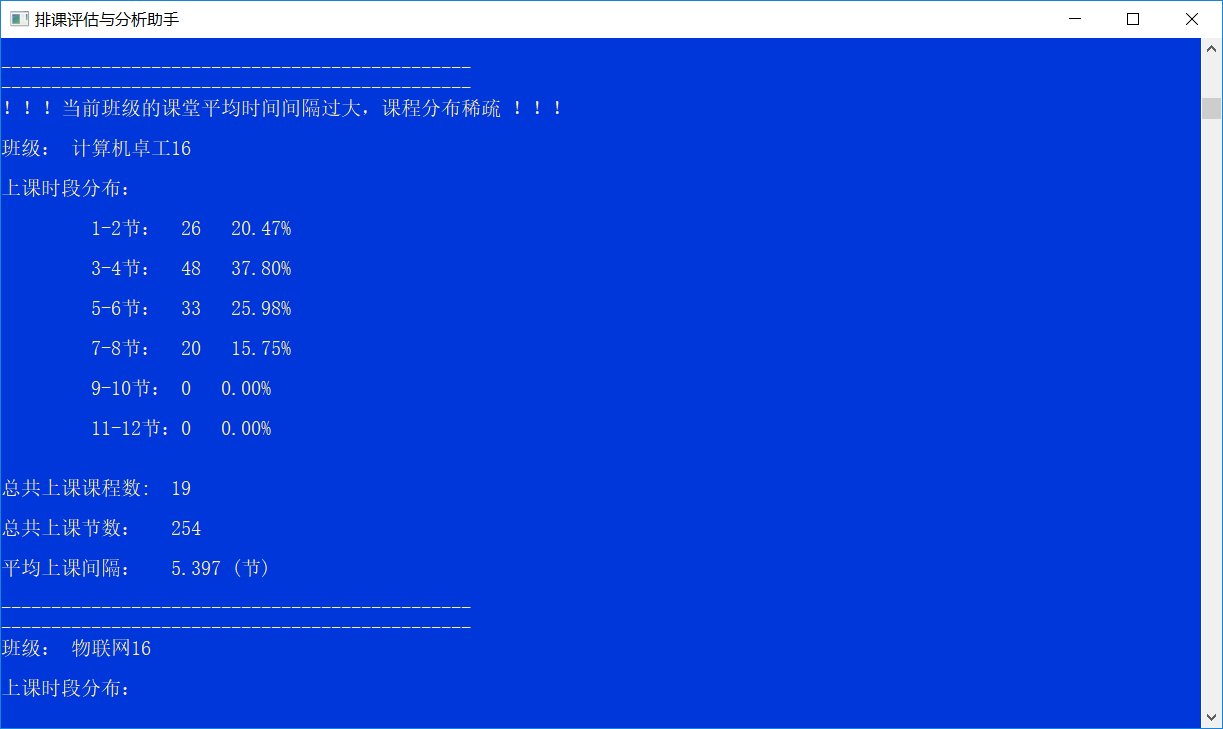


图4-121 当某个班级的排课参数失衡的时候，输出评估结果，并给出建议

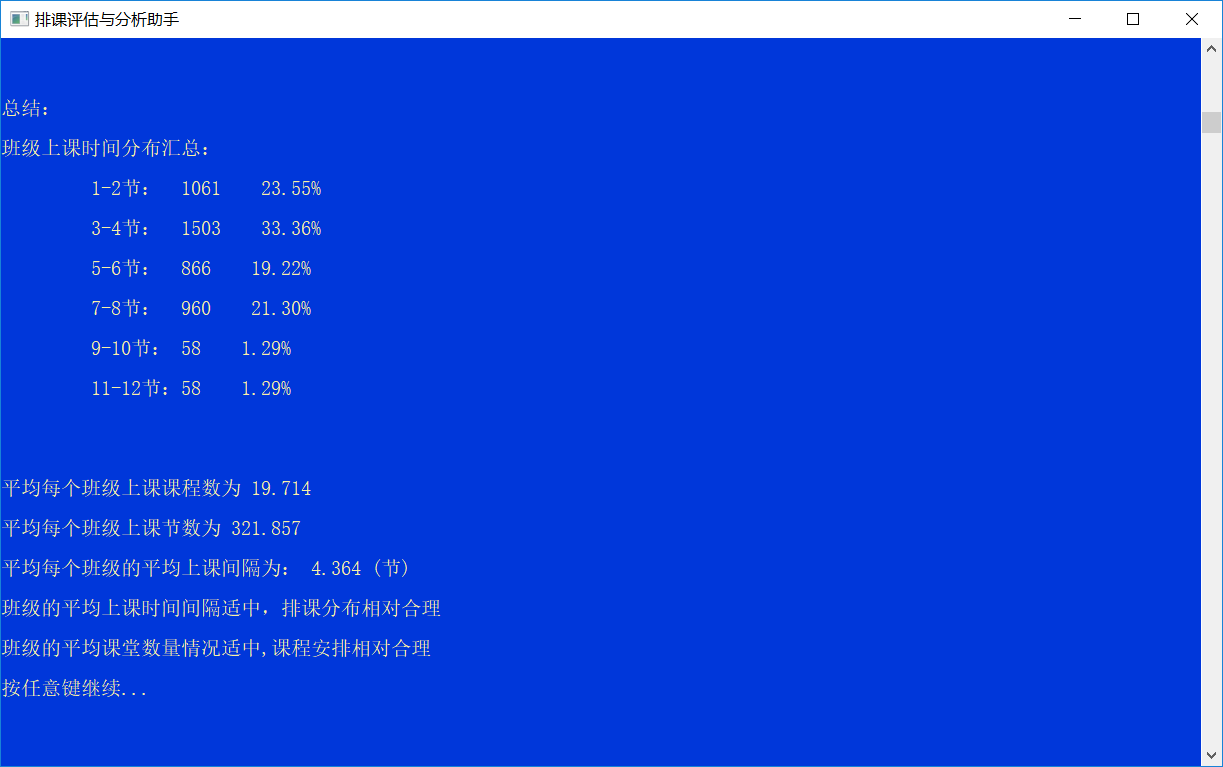


图4-122 在最后输出总结性信息，并给出评估与分析

（4）遍历与检索。本功能是遍历内存中所有的信息。

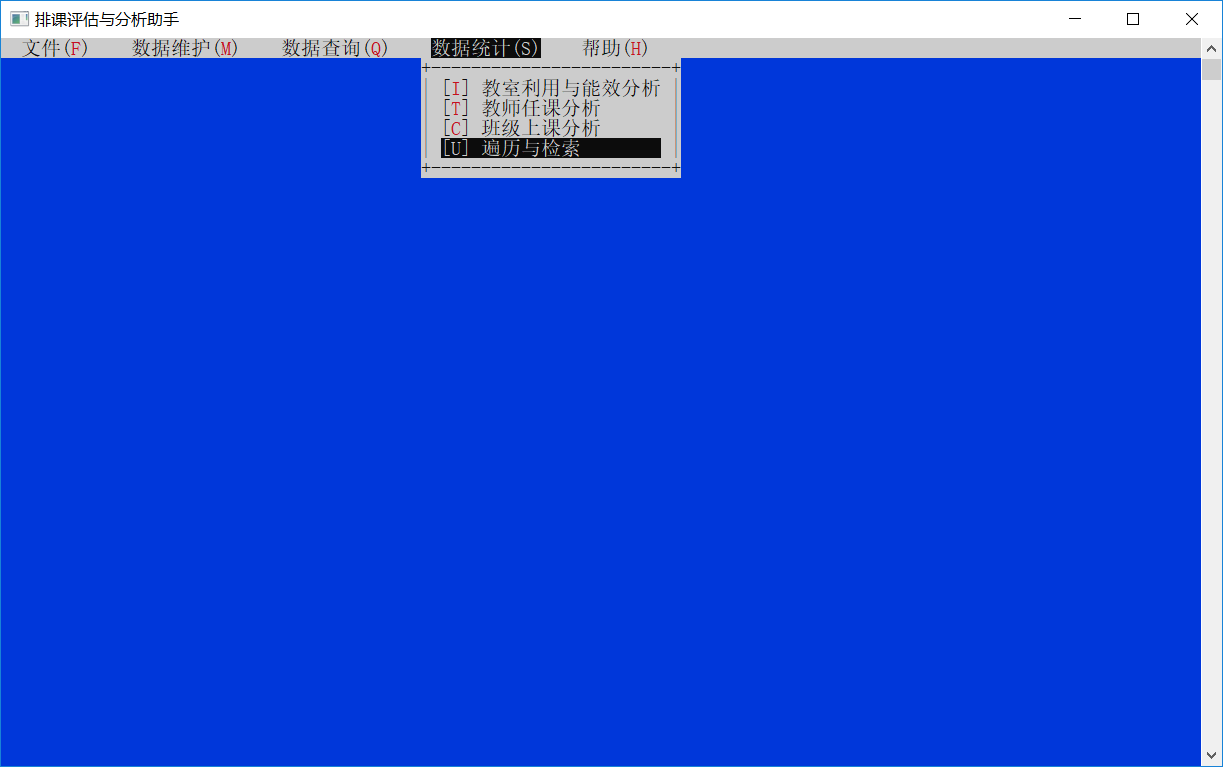


图4-123 遍历与检索

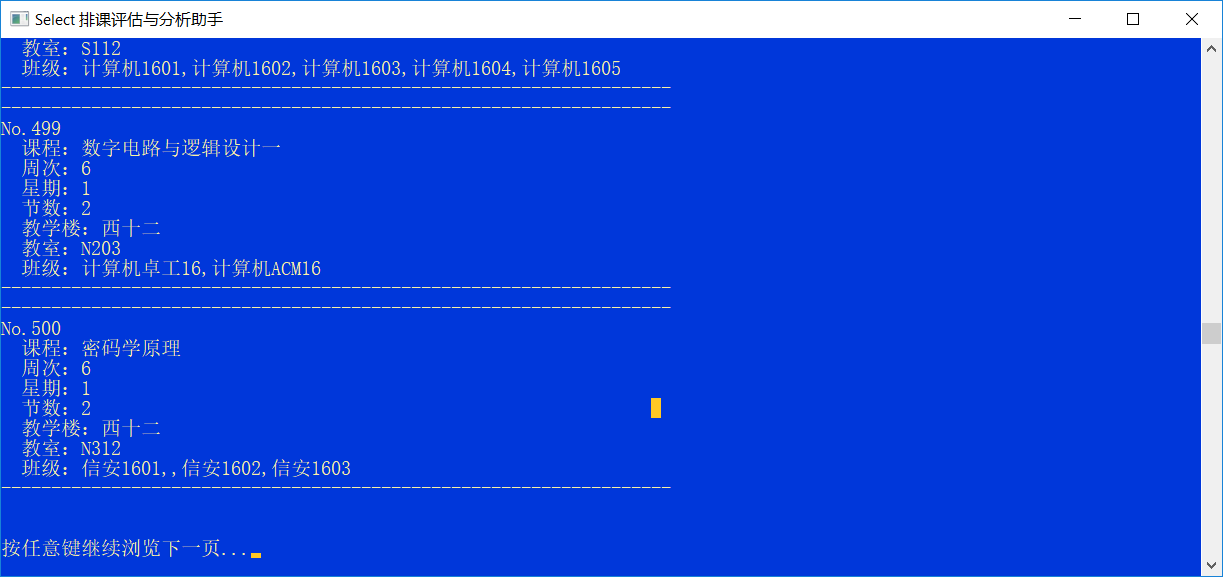


图4-124 由于数据量比较大，因此系统采用分页浏览的输出策略（此图中为前500项）



图4-125 由于数据量比较大，因此系统采用分页浏览的输出策略（此图中为前前1000项）

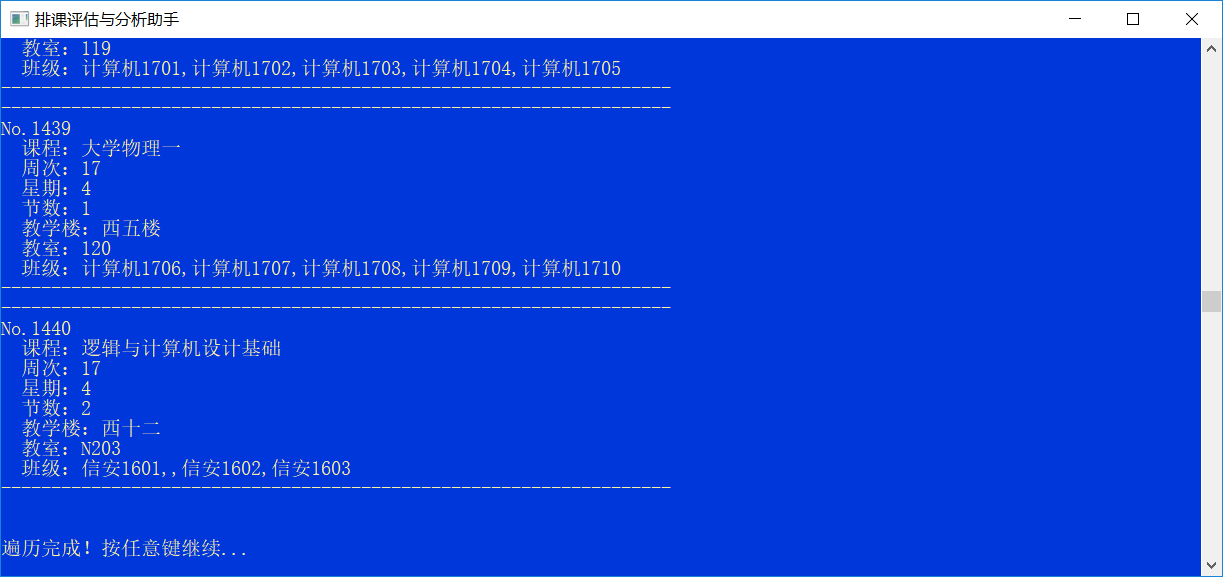


图4-126 遍历完成

分析结果：本模块为“读操作”，一方面反映了数据读入的正确性，同时也能对排课数据文件中的的信息进行统计评估，并且针对统计结果给用户一个比较合理的建议。

1. 帮助模块

功能：输出帮助文档，版本信息等。

设计目标：输出帮助信息，给用户相应的警告提示与帮助。

运行结果：

（1）查看帮助主题



图4-127 帮助主题（注意！！安装本程序的电脑需要预装Microsoft Word）

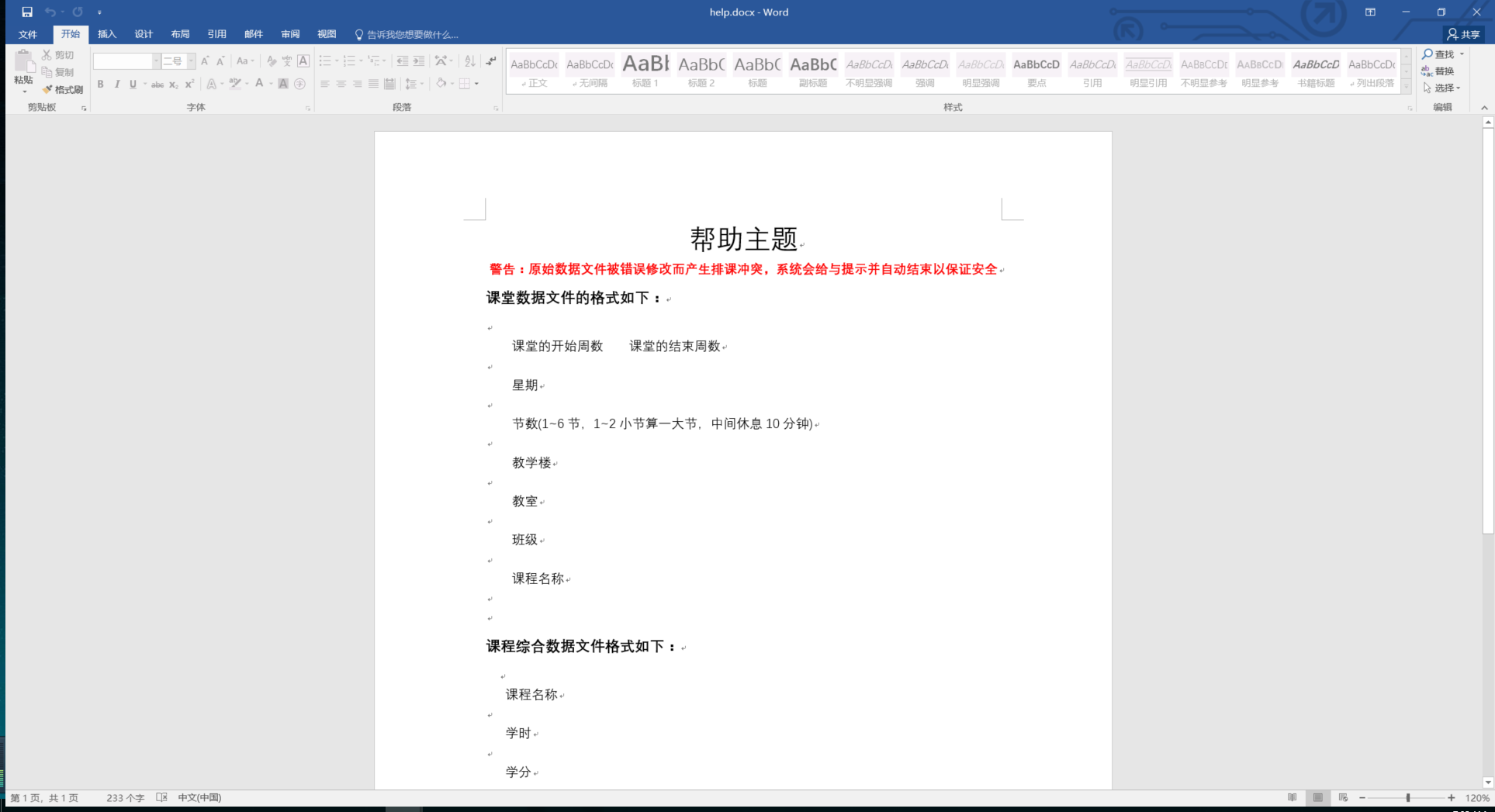


图4-128 打开帮助文档

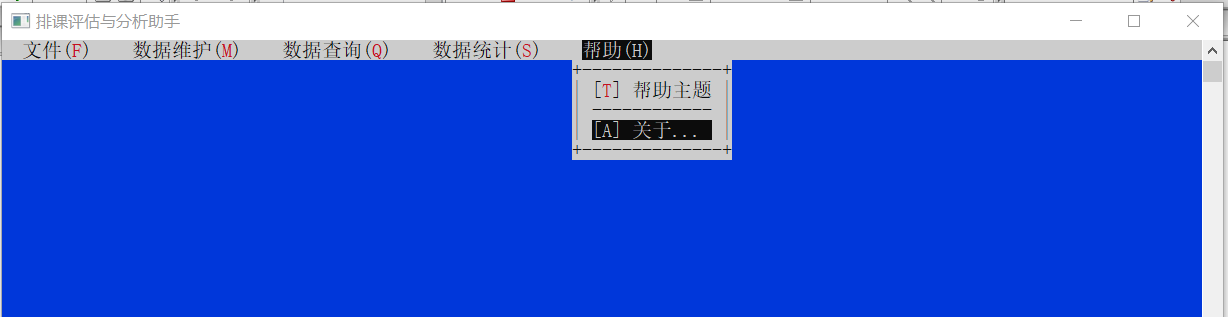
 （2）查看版本信息“关于…”

图4-129 查看版本信息

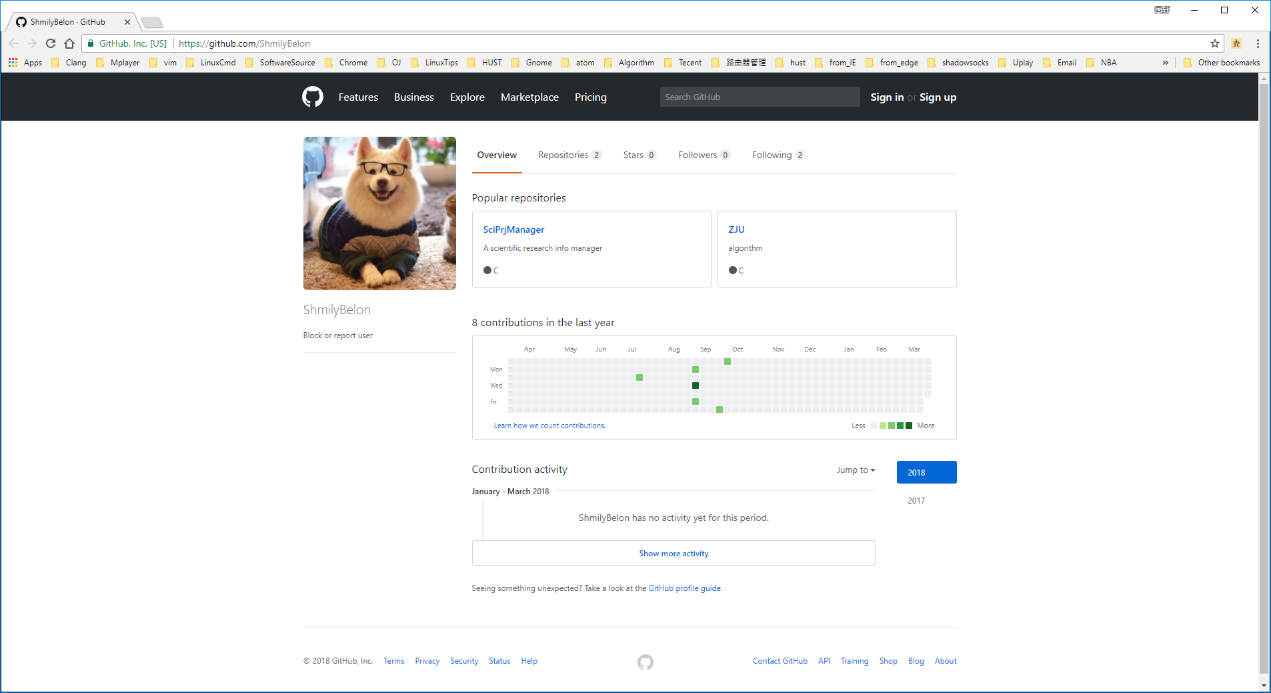
图4-130 输出版本信息

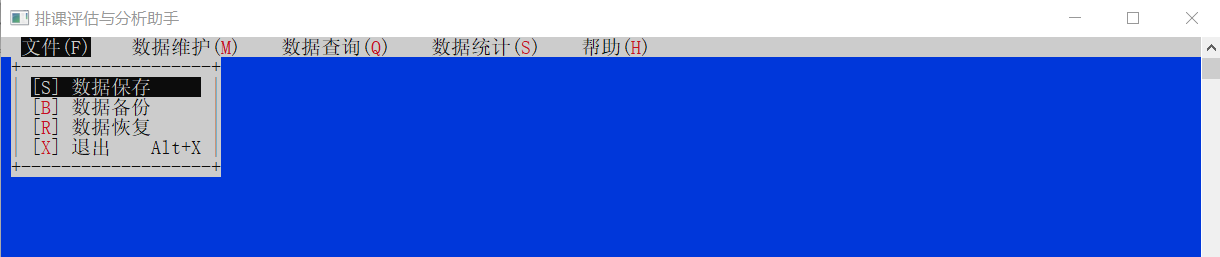
图4-131 打开本程序的源码主页（开发者个人主页代码仓库，系统需要预装浏览器）

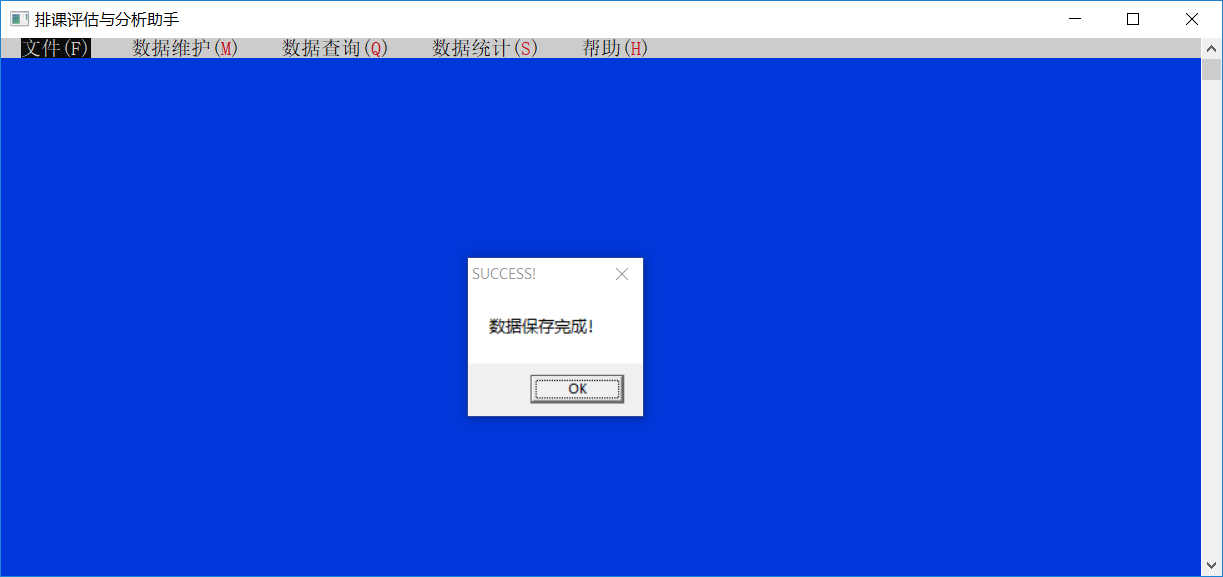
1. 文件保存恢复

功能：实现文件读取任务

设计目标：能够利用针对数据配置文件进行操作。

运行结果：

1. 数据保存

图4-132 执行数据保存功能

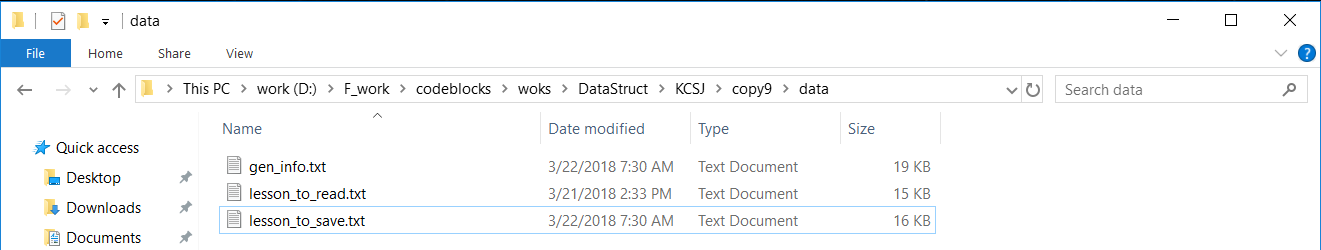
图4-133 数据保存完成

图4-134 可以在程序所在目录的data文件夹下找到data\_to\_save.txt文件

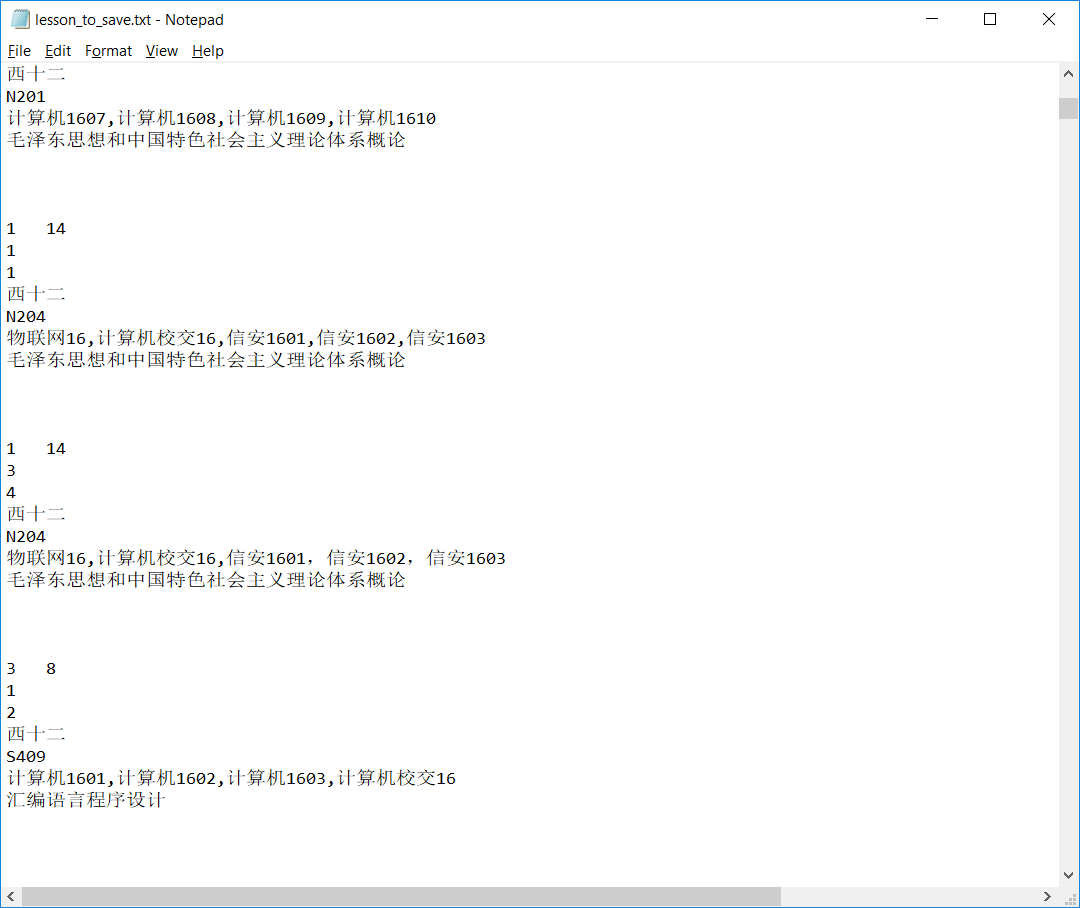


图4-135 可以打开data\_to\_save.txt查看

（2）数据备份

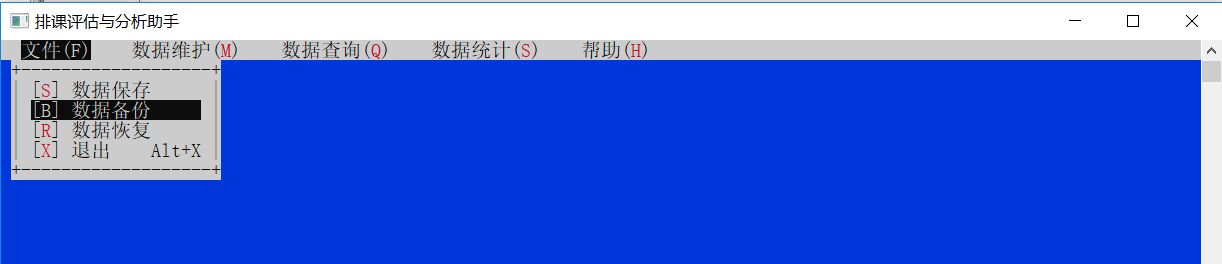


图4-136 数据备份

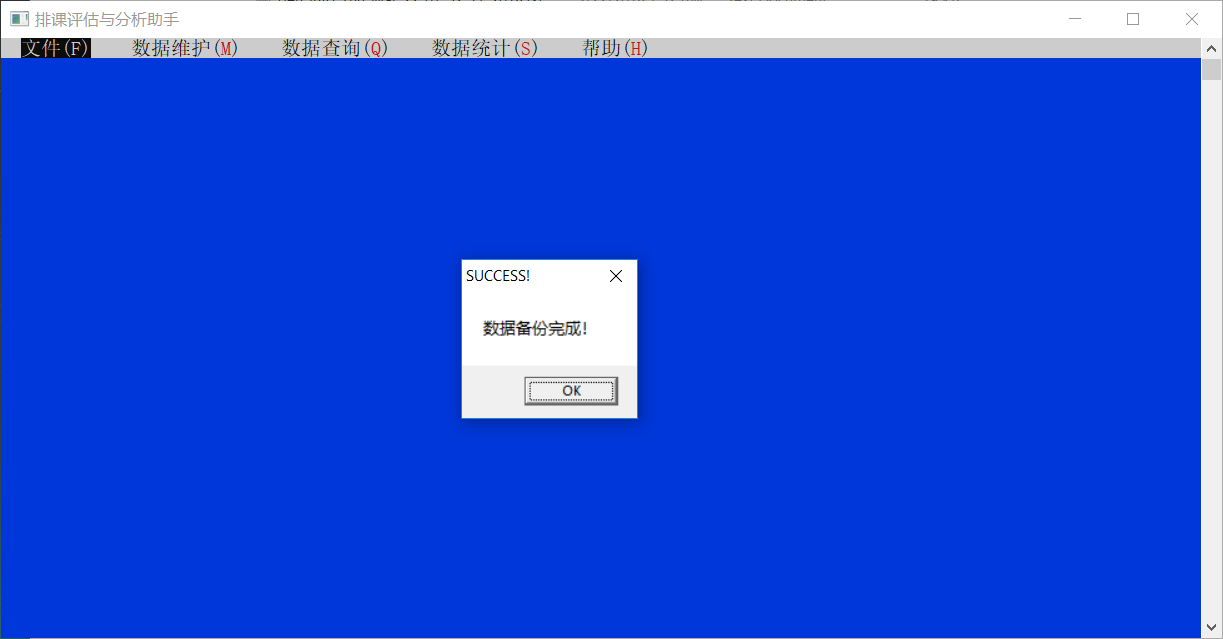


图4-137 数据备份完成

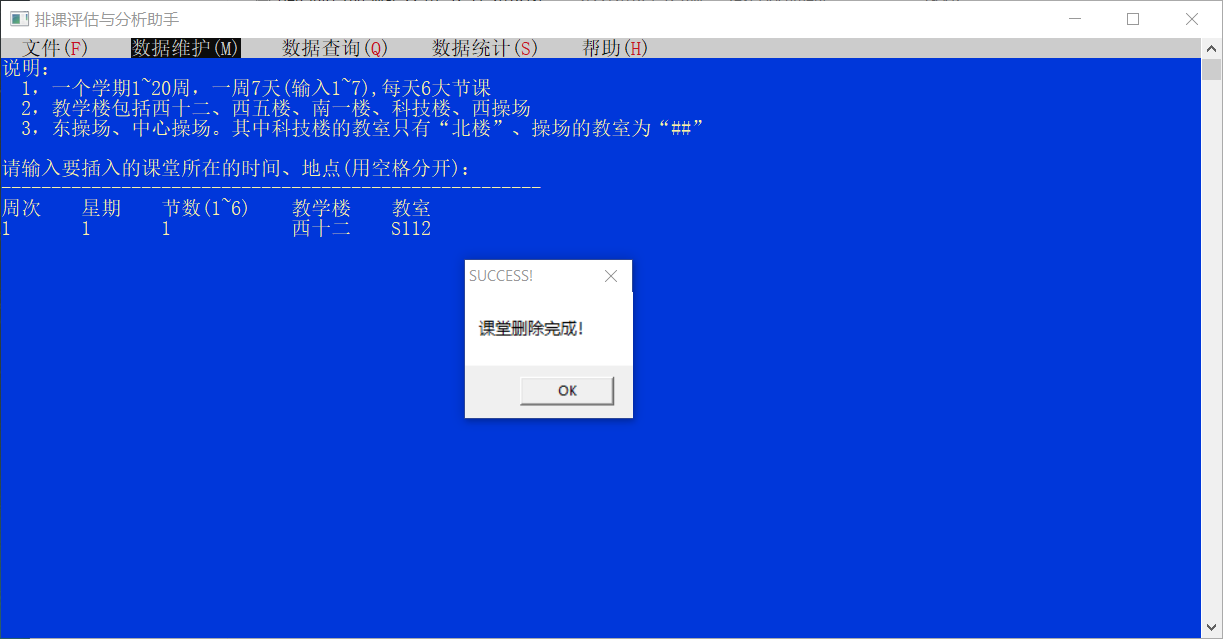


图4-138 先删除一节课（删除功能已经验证其正确性）

（3）数据恢复

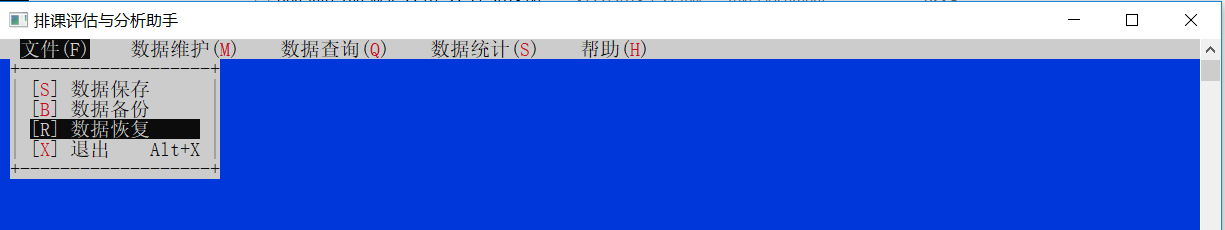


图4-139 数据恢复

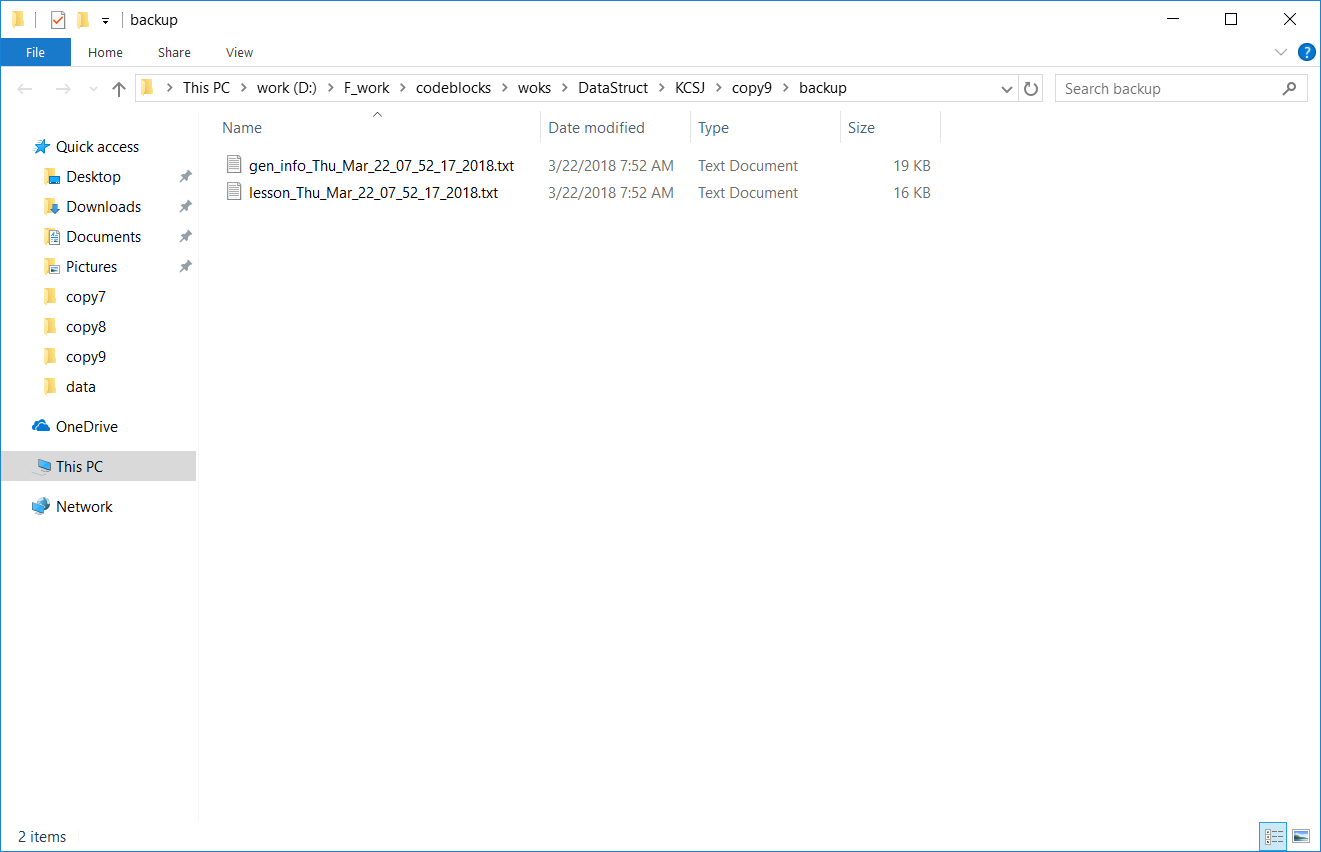


图4-140 程序自动打开程序目录下的backup备份文件夹，可以看到刚才备份的文件，其命名是备份的时间

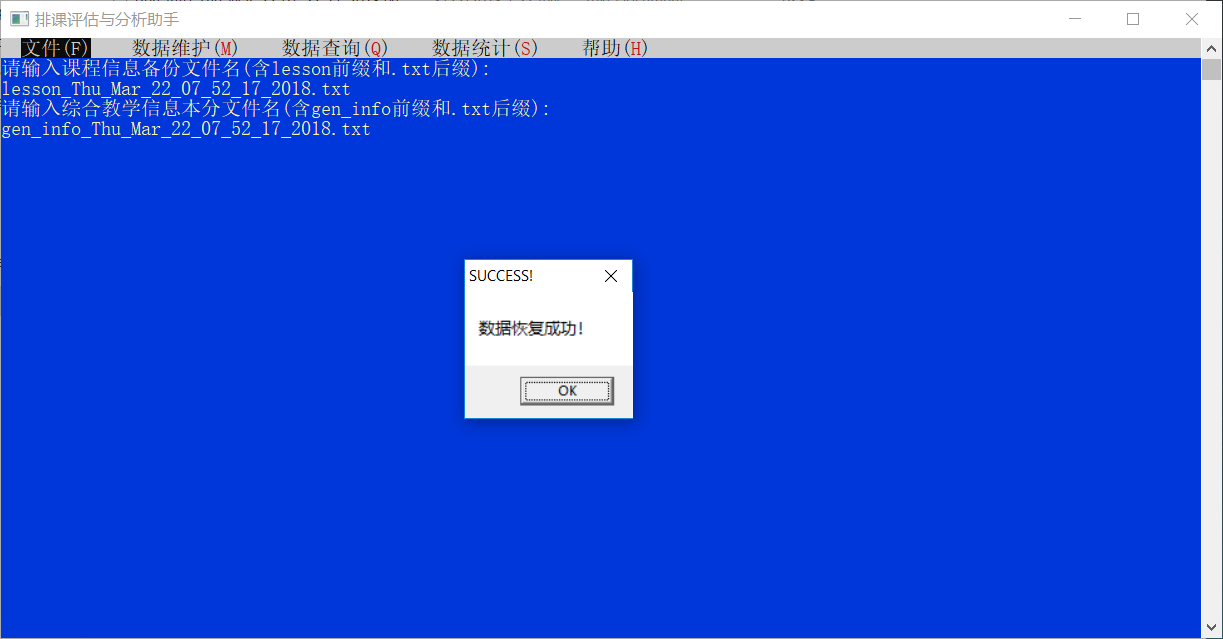


图4-141 按要求输入备份的文件名，回车，数据恢复成功

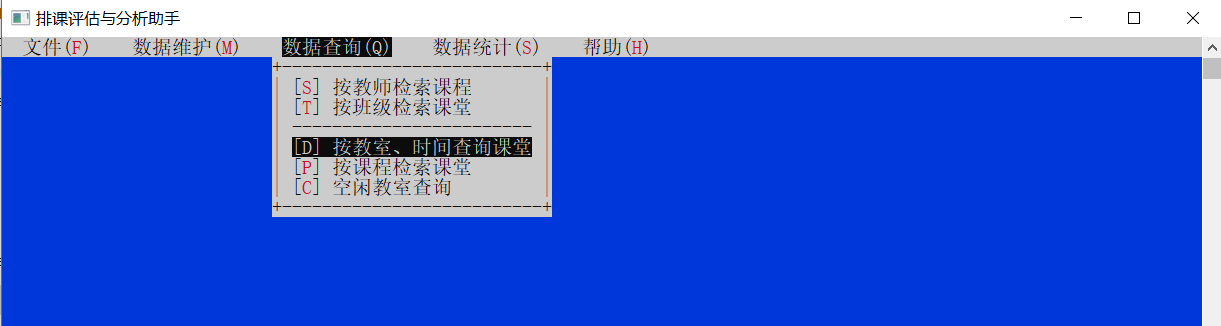


图4-142 按教室、时间查询刚才删除的课堂

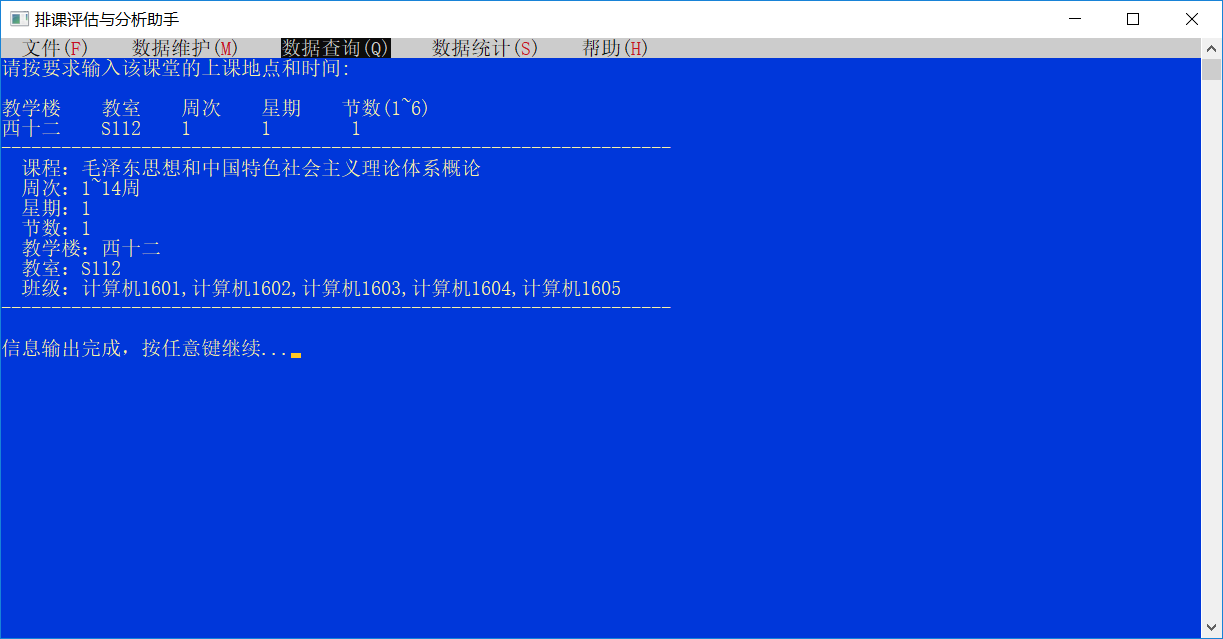


图4-143 输入刚才删除的课堂时间空间信息，发现查询到了课堂，说明数据恢复成功

（5）退出程序

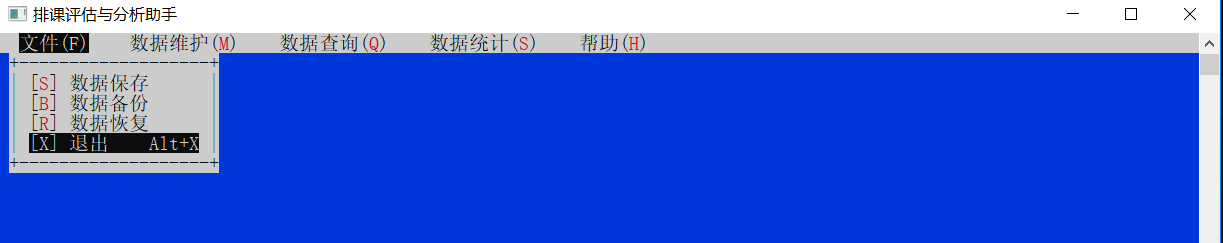


图4-144 退出程序

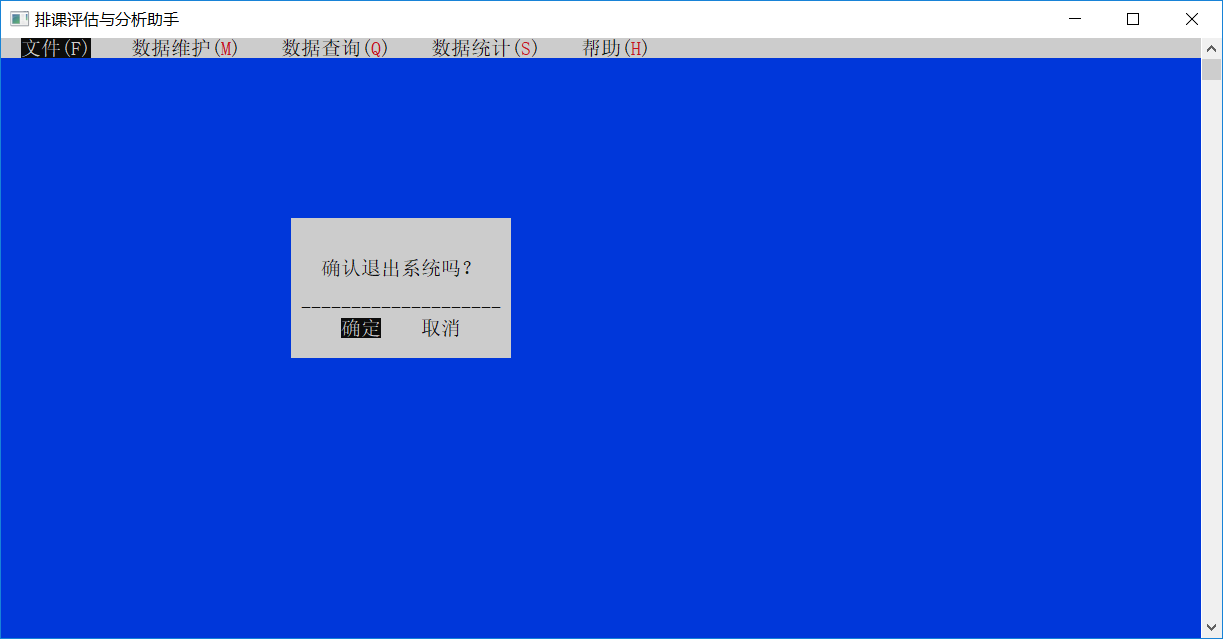


图4-145 弹出提示框，先选“取消”

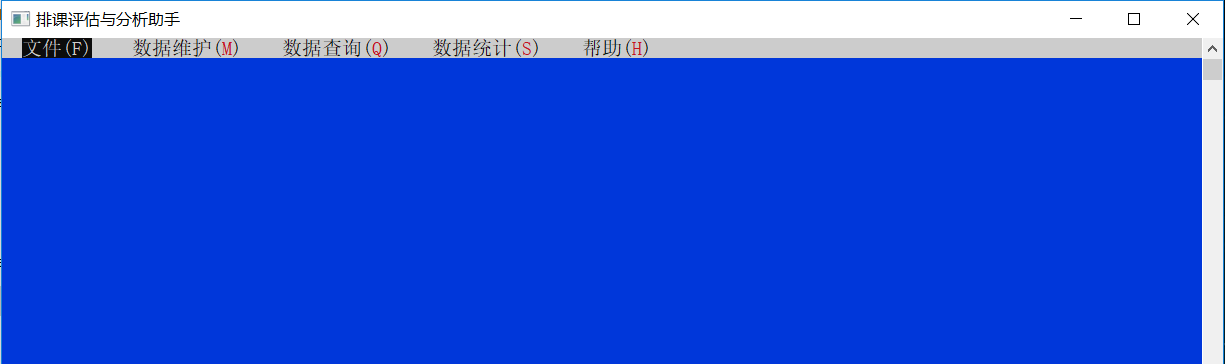


图4-146 撤销成功

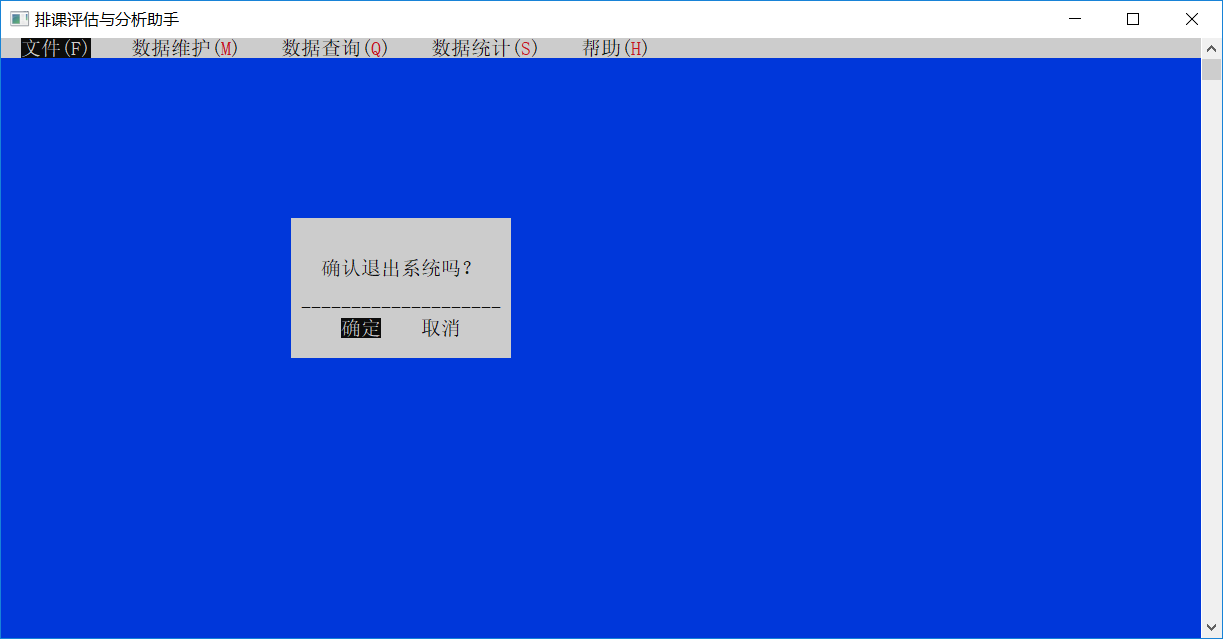


图4-147 再次执行退出系统，这次选择“确定”

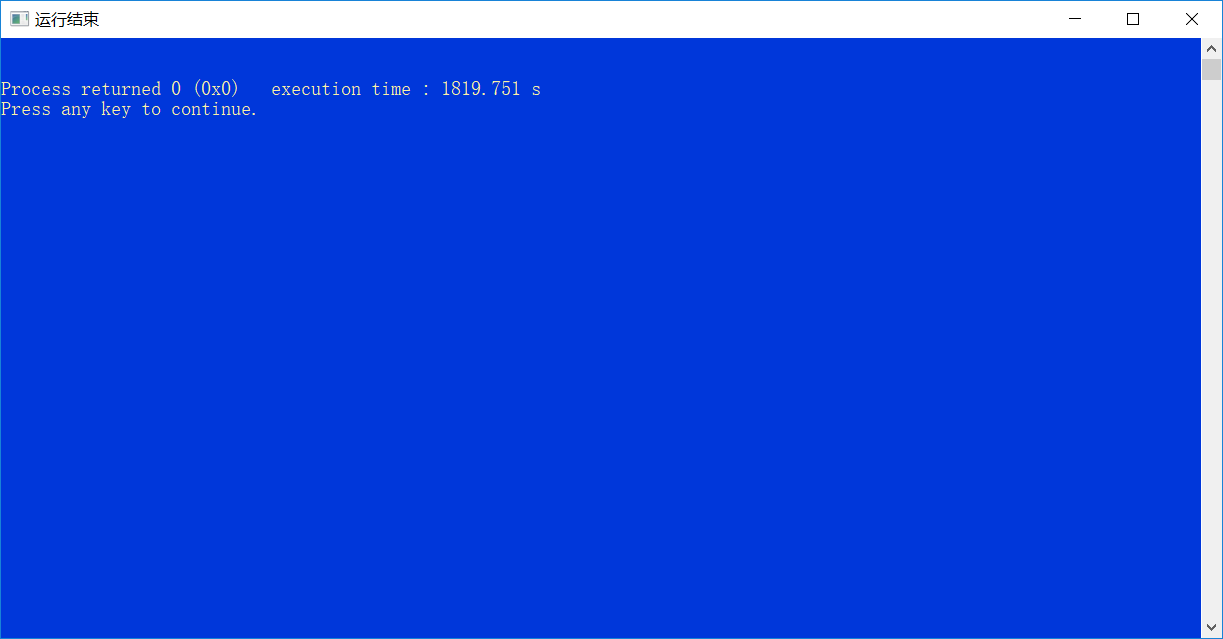


图4-148 运行结束

分析结果：本模块为文件的操作，文件保存和备份功能均执行正常，数据恢复能够成功的恢复之前的备份的数据。

# 5总结与展望

**5.1全文总结**

本报告基本描述了本程序是怎么实现的，思路是什么，以及之后的对程序的测试工作。可以说本报告基本涵盖了从接到任务到总体设计，再到实现功能，最后到测试功能的全部过程。

本程序通过综合运用图、多维压缩矩阵、hash函数等多种数据结构的相关知识，、以基本上小于等于O(n)的时间复杂度实现了所有的插入，修改、删除与查询功能，可以说本程序的实现方式是一种时间效率比较高的实现方式。同时采用压缩矩阵的方式储存课堂信息，也能有效的压缩储存空间。同时在建立矩阵储存的时候就把所有有公共信息的课堂结点用查表链表链接起来了，每个链赋予一个带关键字的表头，这样在查找的时候就非常的方便，不仅节约时间，又不占用额外的空间。

本程序面对数据比较大的情况的时候也有十分优秀的表现，而且能对这些数据以一个比较高效的方式进行统计，而且能对统计出的多种数字特征进行评估与分析，

并且根据统计出的数字特征给出合理化的建议，对于局部统计出的数字特征也能给出评估分析与提示。这样就能比较全面地对排课信息进行管理和评估，提高教学管理水平，更好的服务教师与学生。

本程序基本上采用C语言的函数编程思维来实现，只运用了少量的C++ STL容器与函数，因此基本上采用了面向过程的编程理念。将每个功能用函数模块化，然后尽量抽离多个函数中的共同点，然后把这个共同点用一个函数来实现，这样使得代码更加简洁。然后每个功能函数调用其内核的函数，再将功能函数提供给本程序main函数的接口函数调用，接口函数是与界面函数相关联的，这样用户就可以通过界面中的菜单栏来调用所需的函数，实现要实现的功能了。

由于C语言不支持模板，因此很多相似的功能内核函数还要因为参数类型不同而多写几遍，其实这是一种很不“工程”的写法，但是由于C语言本身的局限性，也不得不这样写。不过也有很多的好处，那就是这种面向过程的接近纯C语言的开发确实练习算法思维和程序设计能力，因为基本上没有什么写好的库函数可以调用，基本上都要自己写，我想这也是为什么数据结构这门课基本采用C语言为教学语言，因为它确实能锻炼开发者的能力。

**5.2工作展望**

通过本次课程设计，我确实学到了不少的东西。首先是我通过把书上学到的图论、hash、压缩矩阵等等按照按任务书的要求以自己的代码风格实现一下，显著提高我的编程能力，同时在这个过程中我强烈的感觉到，动手去做和仅仅听懂是两码事，很多自以为会的东西，到了实践阶段完全不是一回事。

本程序还有很多值得去改进的地方，首先是语言。如果用C++等高级的面向对象的程序设计语言来实现本程序，由于C++有模板，运算符重载，类等特性，可以明显的简化代码，而且使得程序的编程风格更加工程实际化，逻辑层次更加整齐严密。同时C++中的STL提供大量写好的很好用的函数，使用起来不仅方便，也不容易出错。

使用C++语言来写本程序的话还有一个很重要的好处就是可以更容易的与一些图形框架链接，这样的就使得GUI前端的编程不仅容易写，而且更加美观。

用QT5框架来编写本程序，可以省的人写很多代码。但是其实这是一种比较偷懒的做法。因为自己手写前端的话，不仅锻炼自己的编码能力，主要是能增强自己对计算机系统的理解。

通过设计本程序，我增强了对数据结构以及相关算法的理解，同时也认识到，还有很多的数据结构以及算法要去学习。比如目前本程序采用的字符串匹配方法是string.h中的strcmp函数，但是实际上还有很多更高级的处理字符串的方法值得我去学习，比如说Trie树，KMP，Aho-Corasick自动机，Manacher，最小表示法，后缀数组，线性机等。

总之，在数据结构与算法方面，确实有太多太多还要学习的东西，我现在只会一些基础的算法设计，在计数方法，递推，数论，线段树，计算几何，线性规划，FFT，动态规划，启发式搜索等众多内容我还没有涉及，所以真的需要抓紧时间，在课余时间多多学习，否者自己的编程能力很难有质的飞越。

# 6体会

本次数据结构的课程设计总体来说做还算是比较顺利，不过感觉时间上还是比较吃紧的，假期太短，又有太多事情要搞，确实应了那句话“成年人的世界中没有什么事情是简单的”，因为生活中总有这样那样的事情需要去考虑，而正常的学习工作又一点不能耽误，可是很多时候还是身心俱疲，真的不想去工作，但是在百般的纠结中起来工作，效率比较低。

其实这样的生活应该去适应，就是“Everyday Struggle”的生活，也就是每天都有大量的任务去做，自己想深入学习算法只能挤时间去学习，在疲惫中学习，在夹缝中求生存。好在我现在已经逐渐习惯了这种生活。

本次程序设计过程中，好的地方就是一开始的布局不错，也就是我一开始就想到了压缩矩阵这种数据结构，这种储存友好，查询友好，时效友好的东西确实十分好用，再加上hash，到后面代码越来越好些，也很少出bug。因此这让我感受到，一开始的架构设计确实很是重要。我也因此联想了一下，那种商业软件的架构师确实很厉害，需要掌握很多知识，还要有很多实战的经验，因为一个好的架构布局会给下面的程序员的工作带来很大的方便，可是如果一开始的架构安排不合理，然后到后面每个函数都要对这种架构的不合理性进行妥协，这样就很麻烦，如果遇到没有办法妥协的情况，那再修改架构，可谓是相当于重做，因此架构确实很重要。

从算法的角度来说，本程序还算是成功，因为架构设计的还算是可以，因此即使是暴利搜素，时间上算是可以接受。

但是要学习的确实太多了，无论是数据结构，还是基于数据结构的算法。就比如数论这个庞大的内容我就知之甚少，模方程、组合游戏、概率期望、置换与应用、矩阵和线性方程组等等，我都还只是有一些概念。而对于优先队列、优先栈、并查集、RMQ，状态压缩等只会调用成型的板子。对于字符串处理，比如Hash、KMP、Manacher、最小表示法、后缀数组、AC自动机、Trie树、最长公共前缀（LCP）、基于Hash的LCP等的代码编写很是不熟练。对于几何问题，比如圆的相关计算、球面的相关计算、点的判定、凸包、半平面交、平面区域、三维点积、三维叉积、三维凸包还需要认真的学习一下。而对于图论这种十分重要的数据结构，应该更加深入的了解，对于最短路、最小生成树、深度优先遍历、无向图的割点和桥、无向图的双连同分量、有向图的强连同分量、2-SAT问题、网络流问题等应该大量的练习，十分熟练的掌握。

总之，要学的还有太多太多，时间紧、任务重，任重而道远……

# 参考文献

[1] 严蔚敏, 吴伟民. 数据结构（C语言版）. 北京: 清华大学出版社,1997

[2] 严蔚敏, 吴伟民, 米宁. 数据结构题集（C语言版）. 北京: 清华大学出版社,1999

[3] Mark Allen Weiss.Data Structures and Algorithm Analysis in C, 机械工业出版社，2010, 177-192

[4] 《C语言实验与课程设计》科学出版社， 李开 卢萍 曹计昌编著

# 附录

源代码：

（1）pkzs.h头文件

#include <cstdio>

#include <cstdlib>

#include <windows.h>

#include <wincon.h>

#include <conio.h>

#include <cstring>

#include <io.h>

#include <fcntl.h>

#include <sys\stat.h>

#include <ctype.h>

#include<algorithm>

#include<vector>

#include <time.h>//获取系统时间

#include <unistd.h> //获得当前工作路径,改变当前目录的函数所在头文件

#ifndef TYPE\_H\_INCLUDED

#define TYPE\_H\_INCLUDED

#define SCR\_ROW 25 /\*屏幕行数\*/

#define SCR\_COL 80 /\*屏幕列数\*/

typedef struct time {

int week; /\*\*< 周数\*/

int day; /\*\*<星期\*/

int period; /\*\*< 节数，一天一共6大节，因此取值为1~6\*/

struct lesson \*time\_next; /\*\*< 指向同一时间不同教室的所有的课程\*/

} TIME;

/\*\*

空间表头节点

\*/

typedef struct place {

char building[15]; /\*\*< 教学楼\*/

char classroom[10]; /\*\*< 教室\*/

struct lesson \*room\_next; /\*\*< 指向同一间教室不同时间的所有的课堂\*/

} PLACE;

/\*\*

课堂结构

\*/

typedef struct lesson {

int from,to; /\*\*<从第几周到第几周\*/

int week; /\*\*< 周数\*/

int day; /\*\*<星期\*/

int period; /\*\*< 节数，一天一共6大节，因此取值为1~6\*/

char building[15];

char classroom[6]; /\*\*< 教室\*/

char classno[150]; /\*\*< 专业班级\*/

char course[80]; /\*\*< 课程名\*/

int No; /\*\*< 在内存中的编号\*/

struct lesson \*time\_next; /\*\*< 指向同一间教室不同时间的下一个课堂\*/

struct lesson \*room\_next; /\*\*<指向同一时间不同教室的下一个课堂\*/

struct lesson \*time\_pri; /\*\*< 指向同一间教室不同时间的上一个课堂\*/

struct lesson \*room\_pri; /\*\*<指向同一时间不同教室的上一个课堂\*/

struct lesson \*classno\_next; /\*\*< 指向同一班级的下一个课程\*/

struct lesson \*classno\_pri; /\*\*< 指向同一班级的上一个课程\*/

struct lesson \*course\_next; /\*\*< 指向同一课程的下一个课堂\*/

struct lesson \*course\_pri; /\*\*< 指向同一课程的上一个课堂\*/

struct time \*lesson\_time ; /\*\*<指向时间表头\*/

struct place \*lesson\_place ;/\*\*<指向空间表头\*/

} LESSON;

/\*\*

综合信息结构

\*/

typedef struct info {

char course[100]; /\*\*<课程名\*/

char time[10]; /\*\*<学时\*/

char credit[5]; /\*\*<学分\*/

char title[10]; /\*\*<教师职称\*/

char teacher[100];/\*\*<教师姓名\*/

char classno[150]; /\*\*<专业班级\*/

char extra[81]; /\*\*<备注\*/

} INFO;

/\*\*

班级查找链表头数据结构

\*/

typedef struct classno {

char classno[150]; /\*\*< 专业班级\*/

LESSON\* lesson; /\*\*<指向课堂的指针\*/

} CLASSNO;

/\*\*

课程查找链的结构

\*/

typedef struct course {

char course[80];

/\*\*< 课程名\*/

LESSON\* lesson; /\*\*<指向课堂的指针\*/

} COURSE;

TIME les\_time[845]; //时间表头

PLACE les\_place[260];//空间表头

bool readed[845\*260];//在查询函数中标记此课堂是否已读

std::vector<INFO> gen\_info;//储存综合信息

std::vector<CLASSNO> les\_classno;//专业班级查找链

std::vector<COURSE> les\_course;//课程查找链

std::vector<int>find;//删除函数中用来储存遍历过程中要删除的数据元素的下标的vector

int lesson\_sum=0;//当前课程总数

CONSOLE\_CURSOR\_INFO lpCur; //存储光标信息

/\*\*

\*屏幕窗口信息链结点结点结构

\*/

typedef struct layer\_node {

char LayerNo; /\*\*< 弹出窗口层数\*/

SMALL\_RECT rcArea; /\*\*< 弹出窗口区域坐标\*/

CHAR\_INFO \*pContent; /\*\*< 弹出窗口区域字符单元原信息存储缓冲区\*/

char \*pScrAtt; /\*\*< 弹出窗口区域字符单元原属性值存储缓冲区\*/

struct layer\_node \*next; /\*\*< 指向下一结点的指针\*/

} LAYER\_NODE;

/\*\*

\*标签束结构

\*/

typedef struct labe1\_bundle {

char \*\*ppLabel; /\*\*< 标签字符串数组首地址\*/

COORD \*pLoc; /\*\*< 标签定位数组首地址\*/

int num; /\*\*< 标签个数\*/

} LABEL\_BUNDLE;

/\*\*

\*热区结构

\*/

typedef struct hot\_area {

SMALL\_RECT \*pArea; /\*\*< 热区定位数组首地址\*/

char \*pSort; /\*\*< 热区类别(按键、文本框、选项框)数组首地址\*/

char \*pTag; /\*\*< 热区序号数组首地址\*/

int num; /\*\*< 热区个数\*/

} HOT\_AREA;

LAYER\_NODE \*gp\_top\_layer = NULL; /\*弹出窗口信息链链头\*/

char \*gp\_sys\_name = "排课评估与分析助手"; /\*系统名称\*/

char \*ga\_main\_menu[] = {"文件(F)", /\*系统主菜单名\*/

"数据维护(M)",

"数据查询(Q)",

"数据统计(S)",

"帮助(H)"

};

char \*ga\_sub\_menu[] = {"[S] 数据保存", /\*系统子菜单名\*/

"[B] 数据备份",

"[R] 数据恢复",

"[X] 退出 Alt+X",

"[S] 课堂信息",

"[T] 课程信息",

"", /\*空串用来在弹出菜单中分隔子菜单项，下同\*/

"[D] 教师信息",

"[P] 教室信息",

"[C] 班级信息",

"[S] 按教师检索课程",

"[T] 按班级检索课堂",

"",

"[D] 按教室、时间查询课堂",

"[P] 按课程检索课堂",

"[C] 空闲教室查询",

"[I] 教室利用与能效分析",

"[T] 教师任课分析",

"[C] 班级上课分析",

"[U] 遍历与检索",

"[T] 帮助主题",

"",

"[A] 关于..."

};

int ga\_sub\_menu\_count[] = {4, 6, 6, 4, 3}; /\*各主菜单项下子菜单的个数\*/

int gi\_sel\_menu = 1; /\*被选中的主菜单项号,初始为1\*/

int gi\_sel\_sub\_menu = 0; /\*被选中的子菜单项号,初始为0,表示未选中\*/

CHAR\_INFO \*gp\_buff\_menubar\_info = NULL; /\*存放菜单条屏幕区字符信息的缓冲区\*/

CHAR\_INFO \*gp\_buff\_stateBar\_info = NULL; /\*存放状态条屏幕区字符信息的缓冲区\*/

char \*gp\_scr\_att = NULL; /\*存放屏幕上字符单元属性值的缓冲区\*/

char gc\_sys\_state = '\0'; /\*用来保存系统状态的字符\*/

HANDLE gh\_std\_out; /\*标准输出设备句柄\*/

HANDLE gh\_std\_in; /\*标准输入设备句柄\*/

void InitInterface(void); /\*系统界面初始化\*/

void ClearScreen(void); /\*清屏\*/

void ShowMenu(void); /\*显示菜单栏\*/

void PopMenu(int num); /\*显示下拉菜单\*/

void PopPrompt(int num); /\*显示弹出窗口\*/

void PopUp(SMALL\_RECT \*, WORD, LABEL\_BUNDLE \*, HOT\_AREA \*); /\*弹出窗口屏幕信息维护\*/

void PopOff(void); /\*关闭顶层弹出窗口\*/

void DrawBox(SMALL\_RECT \*parea); /\*绘制边框\*/

void LocSubMenu(int num, SMALL\_RECT \*parea); /\*主菜单下拉菜单定位\*/

void ShowState(void); /\*显示状态栏\*/

void TagMainMenu(int num); /\*标记被选中的主菜单项\*/

void TagSubMenu(int num); /\*标记被选中的子菜单项\*/

int DealConInput(HOT\_AREA \*phot\_area, int \*pihot\_num); /\*控制台输入处理\*/

void SetHotPoint(HOT\_AREA \*phot\_area, int hot\_num); /\*设置热区\*/

void RunSys(void); /\*系统功能模块的选择和运行\*/

BOOL ExeFunction(int main\_menu\_num, int sub\_menu\_num); /\*功能模块的调用\*/

void CloseSys(void); /\*退出系统\*/

BOOL ShowModule(char \*\*pString, int n);

//接口函数

BOOL LoadData(void); /\*数据加载\*/

BOOL SaveData(void); /\*保存数据\*/

BOOL BackupData(void); /\*备份数据\*/

BOOL RestoreData(void); /\*恢复数据\*/

BOOL ExitSys(void); /\*退出系统\*/

BOOL HelpTopic(void); /\*帮助主体\*/

BOOL About(void); /\*关于系统\*/

BOOL MaintainLessonInfo(void); /\*课堂信息维护\*/

BOOL MaintainCourseInfo(void); /\*维护课程信息\*/

BOOL MaintainTeacherInfo(void); /\*维护教师楼信息\*/

BOOL MaintainClassroomInfo(void); /\*维护教室信息\*/

BOOL MaintainClassnoInfo(void); /\*维护班级信息\*/

BOOL QueryCourseByTeacher(void); /\*通过教师此查询课程\*/

BOOL QueryLessonByClassno(void); /\*按班级检索课堂\*/

BOOL QueryLessonByTimeAndPlace(void); /\*根据上课的时间地点检索课堂\*/

BOOL QueryLessonByCourse(void); /\*按课程检索课堂\*/

BOOL QuerySpareClassroom(void); /\*空间教室检索\*/

BOOL StatClassroomRate(void); /\*教师利用率分析\*/

BOOL StatTeacherRate(void); /\*教师任课分析\*/

BOOL StatClassnoRate(void); /\*班级上课分析\*/

BOOL TraverseAll(void); /\*按时间顺序遍历所有信息\*/

//自己新加的功能函数

int PlacestrToNum(char s1[] , char s2[]);//地点哈希

int TimestrToNum(int week , int day , int period);//时间哈希

int LessonInsert(int placei,int timei,LESSON\* temp);//课堂引索插入

int ClassnoInsert(LESSON\* temp);//班级引索插入

int CourseInsert(LESSON\* temp);//课程引索插入

int ReadLesson(FILE \*fp);//从课堂文件读入数据

int Init(void);//表头初始化

int SearchByCourse(char s[]);//根据课程查询

int SearchByClassno(char s[]);//根据班号 查询

int SearchByTime(int week,int day, int period );//根据上课时间查询

int SearchByPlace(char building[] , char classroom[]);//根据上课地点查询

int ReadGeneralInfo(FILE \* fp);//读常规信息（teacher相关）

int SearchTeacherInfo(char s[]);//查询教师信息

int ToSaveData(FILE\* fp1,FILE\* fp2);//用来向指定的文件中保存信息

void SpaceToUnderline(char\*s);//把空格变成下划线

void AddBackslash(char\*s) ;//单下划线变成双下划线

void GetBackupFilename(char \*dir1, char\* dir2 );//得到包含路径的备份文件名

int ToBackup(char\* filename1,char\* filename2);//向指定的文件中保存备份文件

void ClearScreenTotalInputArea(void);//清屏（除了第一行）

int DestroyLesson(void);//清除内存中所有的课程信息

int DestroyOtherVectorInfo(void);//清除所有用vector储存的信息

int DeleteLesson(int timei, int placei);//删除一个给定时间给定地点的课堂

int ToInsertLesson(void);//在控制台手动插入一节课

int ToDeleteLesson(void);//在控制台手动删除一节课

int ReloadData(void);//重新载入课堂信息

int ToModifyLesson(void);//修改课堂信息

int ToInsertCourse(void);//插入课程信息

int ToFindGenInfoByCourse(char s[]);//在综合信息中通过课程关键字检索信息，并储存在vector类型的find中

int ToDeleteCourse(void);//删除课程信息

int ToModifyCourse(void);//修改课程信息

int ToInsertTeacher(void);//插入新的教师

int ToFindGenInfoByTeacher(char s[]);//在综合信息中通过教师关键字检索信息，并储存在vector类型的find中

int ToDeleteTeacher(void);//删除教师

int ToModifyTeacher(void);//修改教师信息

void LessonFileForm(void);//向屏幕输出lesson数据文件的格式信息

int ToQueryCourseByTeacher(void);//通过教师姓名查询课程信息

int ToQueryLessonByClassno(void);//通过班级查询课程信息

int SearchByTimeAndPlace(int timei,int placei);//通过时间空间查找课堂

int ToQueryLessonByTimeAndPlace(void);//通过时间空间检索课堂并输出相关提示信息

int SearchByCourse(char s[]);//通过课程名s检索课堂

int ToQueryLessonByCourse(void);//通过课程名检索课堂并输出相关提示信息

int ToQuerySpareClassroom(void);//空闲教室查询

int ToStatClassroomRate(void);//教室利用率分析

int ToStatTeacherRate(void);//教师任课分析

int ToStatClassnoRate(void);//班级上课分析

int MakeClassno(int i,char s[]);//生成专业班号

int ToTraverseAll(void);//按时间先后顺序遍历所有的信息

double AverageDx(std::vector<int> v);//计算数组中的数的平均间隔

#endif /\*\*< TYPE\_H\_INCLUDED\*/

1. ExeFunction.h头文件

#include <cstdio>

#include <cstdlib>

#include <windows.h>

#include <wincon.h>

#include <conio.h>

#include <cstring>

#include <io.h>

#include <fcntl.h>

#include <sys\stat.h>

#include <ctype.h>

#include<algorithm>

#include<vector>

#include <time.h>//获取系统时间

#include <unistd.h> //获得当前工作路径,改变当前目录的函数所在头文件

#ifndef TYPE\_H\_INCLUDED

#define TYPE\_H\_INCLUDED

#define SCR\_ROW 25 /\*屏幕行数\*/

#define SCR\_COL 80 /\*屏幕列数\*/

//自己加的

typedef struct time {

int week; /\*\*< 周数\*/

int day; /\*\*<星期\*/

int period; /\*\*< 节数，一天一共6大节，因此取值为1~6\*/

struct lesson \*time\_next; /\*\*< 指向同一时间不同教室的所有的课程\*/

} TIME;

/\*\*

空间表头节点

\*/

typedef struct place {

char building[15]; /\*\*< 教学楼\*/

char classroom[10]; /\*\*< 教室\*/

struct lesson \*room\_next; /\*\*< 指向同一间教室不同时间的所有的课堂\*/

} PLACE;

/\*\*

课堂结构

\*/

typedef struct lesson {

int from,to; /\*\*<从第几周到第几周\*/

int week; /\*\*< 周数\*/

int day; /\*\*<星期\*/

int period; /\*\*< 节数，一天一共6大节，因此取值为1~6\*/

char building[15];

char classroom[6]; /\*\*< 教室\*/

char classno[150]; /\*\*< 专业班级\*/

char course[80]; /\*\*< 课程名\*/

int No; /\*\*< 在内存中的编号\*/

struct lesson \*time\_next; /\*\*< 指向同一间教室不同时间的下一个课堂\*/

struct lesson \*room\_next; /\*\*<指向同一时间不同教室的下一个课堂\*/

struct lesson \*time\_pri; /\*\*< 指向同一间教室不同时间的上一个课堂\*/

struct lesson \*room\_pri; /\*\*<指向同一时间不同教室的上一个课堂\*/

struct lesson \*classno\_next; /\*\*< 指向同一班级的下一个课程\*/

struct lesson \*classno\_pri; /\*\*< 指向同一班级的上一个课程\*/

struct lesson \*course\_next; /\*\*< 指向同一课程的下一个课堂\*/

struct lesson \*course\_pri; /\*\*< 指向同一课程的上一个课堂\*/

struct time \*lesson\_time ; /\*\*<指向时间表头\*/

struct place \*lesson\_place ;/\*\*<指向空间表头\*/

} LESSON;

/\*\*

综合信息结构

\*/

typedef struct info {

char course[100]; /\*\*<课程名\*/

char time[10]; /\*\*<学时\*/

char credit[5]; /\*\*<学分\*/

char title[10]; /\*\*<教师职称\*/

char teacher[100];/\*\*<教师姓名\*/

char classno[150]; /\*\*<专业班级\*/

char extra[81]; /\*\*<备注\*/

} INFO;

/\*\*

班级查找链表头数据结构

\*/

typedef struct classno {

char classno[150]; /\*\*< 专业班级\*/

LESSON\* lesson; /\*\*<指向课堂的指针\*/

} CLASSNO;

/\*\*

课程查找链的结构

\*/

typedef struct course {

char course[80]; /\*\*< 课程名\*/

LESSON\* lesson; /\*\*<指向课堂的指针\*/

} COURSE;

extern TIME les\_time[845];//时间表头

extern PLACE les\_place[260];//空间表头

extern bool readed[845\*260];//在查询函数中标记此课堂是否已读

extern std::vector<INFO> gen\_info ;//储存综合信息

extern std::vector<CLASSNO> les\_classno;//专业班级查找链

extern std::vector<COURSE> les\_course;//课程查找链

extern std::vector<int>find;//删除函数中用来储存遍历过程中要删除的数据元素的下标的vector

extern int lesson\_sum;//储存矩阵中的节点个数

extern CONSOLE\_CURSOR\_INFO lpCur; //存储光标信息

/\*\*

\*屏幕窗口信息链结点结点结构

\*/

typedef struct layer\_node {

char LayerNo; /\*\*< 弹出窗口层数\*/

SMALL\_RECT rcArea; /\*\*< 弹出窗口区域坐标\*/

CHAR\_INFO \*pContent; /\*\*< 弹出窗口区域字符单元原信息存储缓冲区\*/

char \*pScrAtt; /\*\*< 弹出窗口区域字符单元原属性值存储缓冲区\*/

struct layer\_node \*next; /\*\*< 指向下一结点的指针\*/

} LAYER\_NODE;

/\*\*

\*标签束结构

\*/

typedef struct labe1\_bundle {

char \*\*ppLabel; /\*\*< 标签字符串数组首地址\*/

COORD \*pLoc; /\*\*< 标签定位数组首地址\*/

int num; /\*\*< 标签个数\*/

} LABEL\_BUNDLE;

/\*\*

\*热区结构

\*/

typedef struct hot\_area {

SMALL\_RECT \*pArea; /\*\*< 热区定位数组首地址\*/

char \*pSort; /\*\*< 热区类别(按键、文本框、选项框)数组首地址\*/

char \*pTag; /\*\*< 热区序号数组首地址\*/

int num; /\*\*< 热区个数\*/

} HOT\_AREA;

extern LAYER\_NODE \*gp\_top\_layer ; /\*弹出窗口信息链链头\*/

extern char \*gp\_sys\_name ; /\*系统名称\*/

extern char \*ga\_main\_menu[] ; /\*系统主菜单名\*/

extern char \*ga\_sub\_menu[] ; /\*系统子菜单名\*/

extern int ga\_sub\_menu\_count[] ; /\*各主菜单项下子菜单的个数\*/

extern int gi\_sel\_menu ; /\*被选中的主菜单项号,初始为1\*/

extern int gi\_sel\_sub\_menu ; /\*被选中的子菜单项号,初始为0,表示未选中\*/

extern CHAR\_INFO \*gp\_buff\_menubar\_info ; /\*存放菜单条屏幕区字符信息的缓冲区\*/

extern CHAR\_INFO \*gp\_buff\_stateBar\_info ; /\*存放状态条屏幕区字符信息的缓冲区\*/

extern char \*gp\_scr\_att ; /\*存放屏幕上字符单元属性值的缓冲区\*/

extern char gc\_sys\_state ; /\*用来保存系统状态的字符\*/

extern HANDLE gh\_std\_out; /\*标准输出设备句柄\*/

extern HANDLE gh\_std\_in; /\*标准输入设备句柄\*/

void InitInterface(void); /\*系统界面初始化\*/

void ClearScreen(void); /\*清屏\*/

void ShowMenu(void); /\*显示菜单栏\*/

void PopMenu(int num); /\*显示下拉菜单\*/

void PopPrompt(int num); /\*显示弹出窗口\*/

void PopUp(SMALL\_RECT \*, WORD, LABEL\_BUNDLE \*, HOT\_AREA \*); /\*弹出窗口屏幕信息维护\*/

void PopOff(void); /\*关闭顶层弹出窗口\*/

void DrawBox(SMALL\_RECT \*parea); /\*绘制边框\*/

void LocSubMenu(int num, SMALL\_RECT \*parea); /\*主菜单下拉菜单定位\*/

void ShowState(void); /\*显示状态栏\*/

void TagMainMenu(int num); /\*标记被选中的主菜单项\*/

void TagSubMenu(int num); /\*标记被选中的子菜单项\*/

int DealConInput(HOT\_AREA \*phot\_area, int \*pihot\_num); /\*控制台输入处理\*/

void SetHotPoint(HOT\_AREA \*phot\_area, int hot\_num); /\*设置热区\*/

void RunSys(void); /\*系统功能模块的选择和运行\*/

BOOL ExeFunction(int main\_menu\_num, int sub\_menu\_num); /\*功能模块的调用\*/

void CloseSys(void); /\*退出系统\*/

BOOL ShowModule(char \*\*pString, int n);

BOOL LoadData(void); /\*数据加载\*/

BOOL SaveData(void); /\*保存数据\*/

BOOL BackupData(void); /\*备份数据\*/

BOOL RestoreData(void); /\*恢复数据\*/

BOOL ExitSys(void); /\*退出系统\*/

BOOL HelpTopic(void); /\*帮助主体\*/

BOOL About(void); /\*关于系统\*/

BOOL MaintainLessonInfo(void); /\*维护课堂信息\*/

BOOL MaintainCourseInfo(void); /\*维护课程信息\*/

BOOL MaintainTeacherInfo(void); /\*维护教师信息\*/

BOOL MaintainClassnoInfo(void); /\*维护班级信息\*/

BOOL MaintainChargeInfo(void); /\*维护学生缴费信息\*/

BOOL QueryCourseByTeacher(void); /\*通过教师查询课程\*/

BOOL QueryLessonByClassno(void); /\*按班级检索课堂\*/

BOOL QueryLessonByTimeAndPlace(void); /\*根据上课的时间地点检索课堂\*/

BOOL QueryLessonByCourse(void); /\*按课程检索课堂\*/

BOOL QuerySpareClassroom(void); /\*空间教室检索\*/

BOOL StatClassroomRate(void); /\*教室利用率分析\*/

BOOL StatTeacherRate(void); /\*教师任课分析\*/

BOOL StatClassnoRate(void); /\*班级上课分析\*/

BOOL TraverseAll(void); /\*按时间顺序遍历所有信息\*/

//自己新加的功能函数

int PlacestrToNum(char s1[] , char s2[]);//地点哈希

int TimestrToNum(int week , int day , int period);//时间哈希

int LessonInsert(int placei,int timei,LESSON\* temp);//课堂引索插入

int ClassnoInsert(LESSON\* temp);//班级引索插入

int CourseInsert(LESSON\* temp);//课程引索插入

int ReadLesson(FILE \*fp);//从课堂文件读入数据

int Init(void);//表头初始化

int SearchByCourse(char s[]);//根据课程查询

int SearchByClassno(char s[]);//根据班号 查询

int SearchByTime(int week,int day, int period );//根据上课时间查询

int SearchByPlace(char building[] , char classroom[]);//根据上课地点查询

int ReadGeneralInfo(FILE \* fp);//读常规信息（teacher相关）

int SearchTeacherInfo(char s[]);//查询教师信息

int ToSaveData(FILE\* fp1,FILE\* fp2);//用来向指定的文件中保存信息

void SpaceToUnderline(char\*s);//把空格变成下划线

void AddBackslash(char\*s) ;//单下划线变成双下划线

void GetBackupFilename(char \*dir1, char\* dir2 );//得到包含路径的备份文件名

int ToBackup(char\* filename1,char\* filename2);//向指定的文件中保存备份文件

void ClearScreenTotalInputArea(void);//清屏（除了第一行）

int DestroyLesson(void);//清除内存中所有的课程信息

int DestroyOtherVectorInfo(void);//清除所有用vector储存的信息

int DeleteLesson(int timei, int placei);//删除一个给定时间给定地点的课堂

int ToInsertLesson(void);//在控制台手动插入一节课

int ToDeleteLesson(void);//在控制台手动删除一节课

int ReloadData(void);//重新载入课堂信息

int ToModifyLesson(void);//修改课堂信息

int ToInsertCourse(void);//插入课程信息

int ToFindGenInfoByCourse(char s[]);//在综合信息中通过课程关键字检索信息，并储存在vector类型的find中

int ToDeleteCourse(void);//删除课程信息

int ToModifyCourse(void);//修改课程信息

int ToInsertTeacher(void);//插入新的教师

int ToDeleteTeacher(void);//删除教师

int ToModifyTeacher(void);//修改教师信息

void LessonFileForm(void);//向屏幕输出lesson数据文件的格式信息

int ToQueryCourseByTeacher(void);//通过教师姓名查询课程信息

int ToQueryLessonByClassno(void);//通过班级查询课程信息

int SearchByTimeAndPlace(int timei,int placei);//通过时间空间查找课堂

int ToQueryLessonByTimeAndPlace(void);//通过时间空间检索课堂并输出相关提示信息

int SearchByCourse(char s[]);//通过课程名s检索课堂

int ToQueryLessonByCourse(void);//通过课程名检索课堂并输出相关提示信息

int ToQuerySpareClassroom(void);//空闲教室查询

int ToStatClassroomRate(void);//教室利用率分析

int ToStatTeacherRate(void);//教师任课分析

int ToStatClassnoRate(void);//班级上课分析

int MakeClassno(int i,char s[]);//生成专业班号

int ToTraverseAll(void);//按时间先后顺序遍历所有的信息

double AverageDx(std::vector<int> v);//计算数组中的数的平均间隔

#endif /\*\*< TYPE\_H\_INCLUDED\*/

1. main.cpp文件

#include "pkzs.h"

unsigned long ul;

int main()

{

COORD size = {SCR\_COL, SCR\_ROW}; /\*窗口缓冲区大小\*/

gh\_std\_out = GetStdHandle(STD\_OUTPUT\_HANDLE); /\* 获取标准输出设备句柄\*/

gh\_std\_in = GetStdHandle(STD\_INPUT\_HANDLE); /\* 获取标准输入设备句柄\*/

SetConsoleTitle(gp\_sys\_name); /\*设置窗口标题\*/

SetConsoleScreenBufferSize(gh\_std\_out, size); /\*设置窗口缓冲区大小80\*25\*/

LoadData(); /\*数据加载\*/

InitInterface(); /\*界面初始化\*/

RunSys(); /\*系统功能模块的选择及运行\*/

CloseSys(); /\*退出系统\*/

return 0;

}

/\*\*

\* 函数名称: LoadData

\* 函数功能: 装载数据

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: BOOL类型, 正确返回TRUE，错误返回FALSE

\*

\* 调用说明: 为了能够以统一的方式调用各功能函数, 将这些功能函数的原型设为

\* 一致, 即无参数且返回值为BOOL. 返回值为FALSE时, 结束程序运行.

\*/

BOOL LoadData()

{

int bRet=1;

Init();

FILE\* fp=fopen("data\\lesson\_to\_read.txt","r+");

if(fp==NULL)

{

printf("data\\lesson\_to\_read.txt打开失败！\n");

bRet=0;

}

int res1=ReadLesson(fp);

if(res1==-1)

{

printf("课程数据加载失败！\n");

bRet=0;

}

else

printf("课程数据加载成功！\n");

fclose(fp);

fp=fopen("data\\gen\_info.txt","r+");

if(fp==NULL)

{

printf("data\\gen\_info.txt打开失败！\n");

bRet= 0;

}

int res2=ReadGeneralInfo(fp);

if(res2!=-1)

printf("教师信息加载成功！\n班级信息加载成功！\n教室信息加载成功!\n\n按任意键继续...\n");

else

{

printf("教师等信息加载失败！基础数据不完整！\n\n按任意键继续...\n");

bRet=0;

}

getchar();

fclose(fp);

return bRet;

}

/\*\*

\* 函数名称: InitInterface

\* 函数功能: 初始化界面.

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: 无

\*

\*/

void InitInterface()

{

WORD att = FOREGROUND\_RED | FOREGROUND\_GREEN | FOREGROUND\_INTENSITY

| BACKGROUND\_BLUE; /\*黄色前景和蓝色背景\*/

SetConsoleTextAttribute(gh\_std\_out, att); /\*设置控制台屏幕缓冲区字符属性\*/

ClearScreen(); /\* 清屏\*/

/\*创建弹出窗口信息堆栈，将初始化后的屏幕窗口当作第一层弹出窗口\*/

gp\_scr\_att = (char \*)calloc(SCR\_COL \* SCR\_ROW, sizeof(char));/\*屏幕字符属性\*/

gp\_top\_layer = (LAYER\_NODE \*)malloc(sizeof(LAYER\_NODE));

gp\_top\_layer->LayerNo = 0; /\*弹出窗口的层号为0\*/

gp\_top\_layer->rcArea.Left = 0; /\*弹出窗口的区域为整个屏幕窗口\*/

gp\_top\_layer->rcArea.Top = 0;

gp\_top\_layer->rcArea.Right = SCR\_COL - 1;

gp\_top\_layer->rcArea.Bottom = SCR\_ROW - 1;

gp\_top\_layer->pContent = NULL;

gp\_top\_layer->pScrAtt = gp\_scr\_att;

gp\_top\_layer->next = NULL;

ShowMenu(); /\*显示菜单栏\*/

ShowState(); /\*显示状态栏\*/

return;

}

/\*\*

\* 函数名称: ClearScreen

\* 函数功能: 清除屏幕信息.

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: 无

\*

\*/

void ClearScreen(void)

{

CONSOLE\_SCREEN\_BUFFER\_INFO bInfo;

COORD home = {0, 0};

unsigned long size;

GetConsoleScreenBufferInfo( gh\_std\_out, &bInfo );/\*取屏幕缓冲区信息\*/

size = bInfo.dwSize.X \* bInfo.dwSize.Y; /\*计算屏幕缓冲区字符单元数\*/

/\*将屏幕缓冲区所有单元的字符属性设置为当前屏幕缓冲区字符属性\*/

FillConsoleOutputAttribute(gh\_std\_out, bInfo.wAttributes, size, home, &ul);

/\*将屏幕缓冲区所有单元填充为空格字符\*/

FillConsoleOutputCharacter(gh\_std\_out, ' ', size, home, &ul);

return;

}

/\*\*

\* 函数名称: ShowMenu

\* 函数功能: 在屏幕上显示主菜单, 并设置热区, 在主菜单第一项上置选中标记.

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: 无

\*

\*/

void ShowMenu()

{

CONSOLE\_SCREEN\_BUFFER\_INFO bInfo;

CONSOLE\_CURSOR\_INFO lpCur;

COORD size;

COORD pos = {0, 0};

int i, j;

int PosA = 2, PosB;

char ch;

GetConsoleScreenBufferInfo( gh\_std\_out, &bInfo );

size.X = bInfo.dwSize.X;

size.Y = 1;

SetConsoleCursorPosition(gh\_std\_out, pos);

for (i=0; i < 5; i++) /\*在窗口第一行第一列处输出主菜单项\*/

{

printf(" %s ", ga\_main\_menu[i]);

}

GetConsoleCursorInfo(gh\_std\_out, &lpCur);

lpCur.bVisible = FALSE;

SetConsoleCursorInfo(gh\_std\_out, &lpCur); /\*隐藏光标\*/

/\*申请动态存储区作为存放菜单条屏幕区字符信息的缓冲区\*/

gp\_buff\_menubar\_info = (CHAR\_INFO \*)malloc(size.X \* size.Y \* sizeof(CHAR\_INFO));

SMALL\_RECT rcMenu ={0, 0, size.X-1, 0} ;

/\*将窗口第一行的内容读入到存放菜单条屏幕区字符信息的缓冲区中\*/

ReadConsoleOutput(gh\_std\_out, gp\_buff\_menubar\_info, size, pos, &rcMenu);

/\*将这一行中英文字母置为红色，其他字符单元置为白底黑字\*/

for (i=0; i<size.X; i++)

{

(gp\_buff\_menubar\_info+i)->Attributes = BACKGROUND\_BLUE | BACKGROUND\_GREEN

| BACKGROUND\_RED;

ch = (char)((gp\_buff\_menubar\_info+i)->Char.AsciiChar);

if ((ch >= 'A' && ch <= 'Z') || (ch >= 'a' && ch <= 'z'))

{

(gp\_buff\_menubar\_info+i)->Attributes |= FOREGROUND\_RED;

}

}

/\*修改后的菜单条字符信息回写到窗口的第一行\*/

WriteConsoleOutput(gh\_std\_out, gp\_buff\_menubar\_info, size, pos, &rcMenu);

COORD endPos = {0, 1};

SetConsoleCursorPosition(gh\_std\_out, endPos); /\*将光标位置设置在第2行第1列\*/

/\*将菜单项置为热区，热区编号为菜单项号，热区类型为0(按钮型)\*/

i = 0;

do

{

PosB = PosA + strlen(ga\_main\_menu[i]); /\*定位第i+1号菜单项的起止位置\*/

for (j=PosA; j<PosB; j++)

{

gp\_scr\_att[j] |= (i+1) << 2; /\*设置菜单项所在字符单元的属性值\*/

}

PosA = PosB + 4;

i++;

} while (i<5);

TagMainMenu(gi\_sel\_menu); /\*在选中主菜单项上做标记，gi\_sel\_menu初值为1\*/

return;

}

/\*\*

\* 函数名称: ShowState

\* 函数功能: 显示状态条.

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: 无

\*

\* 调用说明: 状态条字符属性为白底黑字, 初始状态无状态信息.

\*/

void ShowState()

{

CONSOLE\_SCREEN\_BUFFER\_INFO bInfo;

COORD size;

COORD pos = {0, 0};

int i;

GetConsoleScreenBufferInfo( gh\_std\_out, &bInfo );

size.X = bInfo.dwSize.X;

size.Y = 1;

SMALL\_RECT rcMenu ={0, bInfo.dwSize.Y-1, size.X-1, bInfo.dwSize.Y-1};

if (gp\_buff\_stateBar\_info == NULL)

{

gp\_buff\_stateBar\_info = (CHAR\_INFO \*)malloc(size.X \* size.Y \* sizeof(CHAR\_INFO));

ReadConsoleOutput(gh\_std\_out, gp\_buff\_stateBar\_info, size, pos, &rcMenu);

}

for (i=0; i<size.X; i++)

{

(gp\_buff\_stateBar\_info+i)->Attributes = BACKGROUND\_BLUE | BACKGROUND\_GREEN

| BACKGROUND\_RED;

/\*

ch = (char)((gp\_buff\_stateBar\_info+i)->Char.AsciiChar);

if ((ch >= 'A' && ch <= 'Z') || (ch >= 'a' && ch <= 'z'))

{

(gp\_buff\_stateBar\_info+i)->Attributes |= FOREGROUND\_RED;

}

\*/

}

WriteConsoleOutput(gh\_std\_out, gp\_buff\_stateBar\_info, size, pos, &rcMenu);

return;

}

/\*\*

\* 函数名称: TagMainMenu

\* 函数功能: 在指定主菜单项上置选中标志.

\* 输入参数: num 选中的主菜单项号

\* 输出参数: 无

\* 返 回 值: 无

\*

\*/

void TagMainMenu(int num)

{

CONSOLE\_SCREEN\_BUFFER\_INFO bInfo;

COORD size;

COORD pos = {0, 0};

int PosA = 2, PosB;

char ch;

int i;

if (num == 0) /\*num为0时，将会去除主菜单项选中标记\*/

{

PosA = 0;

PosB = 0;

}

else /\*否则，定位选中主菜单项的起止位置: PosA为起始位置, PosB为截止位置\*/

{

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

{

PosA += strlen(ga\_main\_menu[i-1]) + 4;

}

PosB = PosA + strlen(ga\_main\_menu[num-1]);

}

GetConsoleScreenBufferInfo( gh\_std\_out, &bInfo );

size.X = bInfo.dwSize.X;

size.Y = 1;

/\*去除选中菜单项前面的菜单项选中标记\*/

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

{

(gp\_buff\_menubar\_info+i)->Attributes = BACKGROUND\_BLUE | BACKGROUND\_GREEN

| BACKGROUND\_RED;

ch = (gp\_buff\_menubar\_info+i)->Char.AsciiChar;

if ((ch >= 'A' && ch <= 'Z') || (ch >= 'a' && ch <= 'z'))

{

(gp\_buff\_menubar\_info+i)->Attributes |= FOREGROUND\_RED;

}

}

/\*在选中菜单项上做标记，黑底白字\*/

for (i=PosA; i<PosB; i++)

{

(gp\_buff\_menubar\_info+i)->Attributes = FOREGROUND\_BLUE | FOREGROUND\_GREEN

| FOREGROUND\_RED;

}

/\*去除选中菜单项后面的菜单项选中标记\*/

for (i=PosB; i<bInfo.dwSize.X; i++)

{

(gp\_buff\_menubar\_info+i)->Attributes = BACKGROUND\_BLUE | BACKGROUND\_GREEN

| BACKGROUND\_RED;

ch = (char)((gp\_buff\_menubar\_info+i)->Char.AsciiChar);

if ((ch >= 'A' && ch <= 'Z') || (ch >= 'a' && ch <= 'z'))

{

(gp\_buff\_menubar\_info+i)->Attributes |= FOREGROUND\_RED;

}

}

/\*将做好标记的菜单条信息写到窗口第一行\*/

SMALL\_RECT rcMenu ={0, 0, size.X-1, 0};

WriteConsoleOutput(gh\_std\_out, gp\_buff\_menubar\_info, size, pos, &rcMenu);

return;

}

/\*\*

\* 函数名称: CloseSys

\* 函数功能: 关闭系统.

\* 输入参数: hd 主链头指针

\* 输出参数: 无

\* 返 回 值: 无

\*

\*/

void CloseSys(void)

{

ClearScreen(); /\*清屏\*/

/\*释放存放菜单条、状态条等信息动态存储区\*/

free(gp\_buff\_menubar\_info);

free(gp\_buff\_stateBar\_info);

/\*关闭标准输入和输出设备句柄\*/

CloseHandle(gh\_std\_out);

CloseHandle(gh\_std\_in);

/\*将窗口标题栏置为运行结束\*/

SetConsoleTitle("运行结束");

return;

}

/\*\*

\* 函数名称: RunSys

\* 函数功能: 运行系统, 在系统主界面下运行用户所选择的功能模块.

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: 无

\*

\*/

void RunSys(void)

{

INPUT\_RECORD inRec;

DWORD res;

COORD pos = {0, 0};

BOOL bRet = TRUE;

int i, loc, num;

int cNo, cAtt; /\*cNo:字符单元层号, cAtt:字符单元属性\*/

char vkc, asc; /\*vkc:虚拟键代码, asc:字符的ASCII码值\*/

while (bRet)

{

/\*从控制台输入缓冲区中读一条记录\*/

ReadConsoleInput(gh\_std\_in, &inRec, 1, &res);

if (inRec.EventType == MOUSE\_EVENT) /\*如果记录由鼠标事件产生\*/

{

pos = inRec.Event.MouseEvent.dwMousePosition; /\*获取鼠标坐标位置\*/

cNo = gp\_scr\_att[pos.Y \* SCR\_COL + pos.X] & 3; /\*取该位置的层号\*/

cAtt = gp\_scr\_att[pos.Y \* SCR\_COL + pos.X] >> 2;/\*取该字符单元属性\*/

if (cNo == 0) /\*层号为0，表明该位置未被弹出子菜单覆盖\*/

{

/\* cAtt > 0 表明该位置处于热区(主菜单项字符单元)

\* cAtt != gi\_sel\_menu 表明该位置的主菜单项未被选中

\* gp\_top\_layer->LayerNo > 0 表明当前有子菜单弹出

\*/

if (cAtt > 0 && cAtt != gi\_sel\_menu && gp\_top\_layer->LayerNo > 0)

{

PopOff(); /\*关闭弹出的子菜单\*/

gi\_sel\_sub\_menu = 0; /\*将选中子菜单项的项号置为0\*/

PopMenu(cAtt); /\*弹出鼠标所在主菜单项对应的子菜单\*/

}

}

else if (cAtt > 0) /\*鼠标所在位置为弹出子菜单的菜单项字符单元\*/

{

TagSubMenu(cAtt); /\*在该子菜单项上做选中标记\*/

}

if (inRec.Event.MouseEvent.dwButtonState

== FROM\_LEFT\_1ST\_BUTTON\_PRESSED) /\*如果按下鼠标左边第一键\*/

{

if (cNo == 0) /\*层号为0，表明该位置未被弹出子菜单覆盖\*/

{

if (cAtt > 0) /\*如果该位置处于热区(主菜单项字符单元)\*/

{

PopMenu(cAtt); /\*弹出鼠标所在主菜单项对应的子菜单\*/

}

/\*如果该位置不属于主菜单项字符单元，且有子菜单弹出\*/

else if (gp\_top\_layer->LayerNo > 0)

{

PopOff(); /\*关闭弹出的子菜单\*/

gi\_sel\_sub\_menu = 0; /\*将选中子菜单项的项号置为0\*/

}

}

else /\*层号不为0，表明该位置被弹出子菜单覆盖\*/

{

if (cAtt > 0) /\*如果该位置处于热区(子菜单项字符单元)\*/

{

PopOff(); /\*关闭弹出的子菜单\*/

gi\_sel\_sub\_menu = 0; /\*将选中子菜单项的项号置为0\*/

/\*执行对应功能函数:gi\_sel\_menu主菜单项号,cAtt子菜单项号\*/

bRet = ExeFunction(gi\_sel\_menu, cAtt);

}

}

}

else if (inRec.Event.MouseEvent.dwButtonState

== RIGHTMOST\_BUTTON\_PRESSED) /\*如果按下鼠标右键\*/

{

if (cNo == 0) /\*层号为0，表明该位置未被弹出子菜单覆盖\*/

{

PopOff(); /\*关闭弹出的子菜单\*/

gi\_sel\_sub\_menu = 0; /\*将选中子菜单项的项号置为0\*/

}

}

}

else if (inRec.EventType == KEY\_EVENT /\*如果记录由按键产生\*/

&& inRec.Event.KeyEvent.bKeyDown) /\*且键被按下\*/

{

vkc = inRec.Event.KeyEvent.wVirtualKeyCode; /\*获取按键的虚拟键码\*/

asc = inRec.Event.KeyEvent.uChar.AsciiChar; /\*获取按键的ASC码\*/

/\*系统快捷键的处理\*/

if (vkc == 112) /\*如果按下F1键\*/

{

if (gp\_top\_layer->LayerNo != 0) /\*如果当前有子菜单弹出\*/

{

PopOff(); /\*关闭弹出的子菜单\*/

gi\_sel\_sub\_menu = 0; /\*将选中子菜单项的项号置为0\*/

}

bRet = ExeFunction(5, 1); /\*运行帮助主题功能函数\*/

}

else if (inRec.Event.KeyEvent.dwControlKeyState

& (LEFT\_ALT\_PRESSED | RIGHT\_ALT\_PRESSED))

{ /\*如果按下左或右Alt键\*/

switch (vkc) /\*判断组合键Alt+字母\*/

{

case 88: /\*Alt+X 退出\*/

if (gp\_top\_layer->LayerNo != 0)

{

PopOff();

gi\_sel\_sub\_menu = 0;

}

bRet = ExeFunction(1,4);

break;

case 70: /\*Alt+F\*/

PopMenu(1);

break;

case 77: /\*Alt+M\*/

PopMenu(2);

break;

case 81: /\*Alt+Q\*/

PopMenu(3);

break;

case 83: /\*Alt+S\*/

PopMenu(4);

break;

case 72: /\*Alt+H\*/

PopMenu(5);

break;

}

}

else if (asc == 0) /\*其他控制键的处理\*/

{

if (gp\_top\_layer->LayerNo == 0) /\*如果未弹出子菜单\*/

{

switch (vkc) /\*处理方向键(左、右、下)，不响应其他控制键\*/

{

case 37:

gi\_sel\_menu--;

if (gi\_sel\_menu == 0)

{

gi\_sel\_menu = 5;

}

TagMainMenu(gi\_sel\_menu);

break;

case 39:

gi\_sel\_menu++;

if (gi\_sel\_menu == 6)

{

gi\_sel\_menu = 1;

}

TagMainMenu(gi\_sel\_menu);

break;

case 40:

PopMenu(gi\_sel\_menu);

TagSubMenu(1);

break;

}

}

else /\*已弹出子菜单时\*/

{

for (loc=0,i=1; i<gi\_sel\_menu; i++)

{

loc += ga\_sub\_menu\_count[i-1];

} /\*计算该子菜单中的第一项在子菜单字符串数组中的位置(下标)\*/

switch (vkc) /\*方向键(左、右、上、下)的处理\*/

{

case 37:

gi\_sel\_menu--;

if (gi\_sel\_menu < 1)

{

gi\_sel\_menu = 5;

}

TagMainMenu(gi\_sel\_menu);

PopOff();

PopMenu(gi\_sel\_menu);

TagSubMenu(1);

break;

case 38:

num = gi\_sel\_sub\_menu - 1;

if (num < 1)

{

num = ga\_sub\_menu\_count[gi\_sel\_menu-1];

}

if (strlen(ga\_sub\_menu[loc+num-1]) == 0)

{

num--;

}

TagSubMenu(num);

break;

case 39:

gi\_sel\_menu++;

if (gi\_sel\_menu > 5)

{

gi\_sel\_menu = 1;

}

TagMainMenu(gi\_sel\_menu);

PopOff();

PopMenu(gi\_sel\_menu);

TagSubMenu(1);

break;

case 40:

num = gi\_sel\_sub\_menu + 1;

if (num > ga\_sub\_menu\_count[gi\_sel\_menu-1])

{

num = 1;

}

if (strlen(ga\_sub\_menu[loc+num-1]) == 0)

{

num++;

}

TagSubMenu(num);

break;

}

}

}

else if ((asc-vkc == 0) || (asc-vkc == 32)){ /\*按下普通键\*/

if (gp\_top\_layer->LayerNo == 0) /\*如果未弹出子菜单\*/

{

switch (vkc)

{

case 70: /\*f或F\*/

PopMenu(1);

break;

case 77: /\*m或M\*/

PopMenu(2);

break;

case 81: /\*q或Q\*/

PopMenu(3);

break;

case 83: /\*s或S\*/

PopMenu(4);

break;

case 72: /\*h或H\*/

PopMenu(5);

break;

case 13: /\*回车\*/

PopMenu(gi\_sel\_menu);

TagSubMenu(1);

break;

}

}

else /\*已弹出子菜单时的键盘输入处理\*/

{

if (vkc == 27) /\*如果按下ESC键\*/

{

PopOff();

gi\_sel\_sub\_menu = 0;

}

else if(vkc == 13) /\*如果按下回车键\*/

{

num = gi\_sel\_sub\_menu;

PopOff();

gi\_sel\_sub\_menu = 0;

bRet = ExeFunction(gi\_sel\_menu, num);

}

else /\*其他普通键的处理\*/

{

/\*计算该子菜单中的第一项在子菜单字符串数组中的位置(下标)\*/

for (loc=0,i=1; i<gi\_sel\_menu; i++)

{

loc += ga\_sub\_menu\_count[i-1];

}

/\*依次与当前子菜单中每一项的代表字符进行比较\*/

for (i=loc; i<loc+ga\_sub\_menu\_count[gi\_sel\_menu-1]; i++)

{

if (strlen(ga\_sub\_menu[i])>0 && vkc==ga\_sub\_menu[i][1])

{ /\*如果匹配成功\*/

PopOff();

gi\_sel\_sub\_menu = 0;

bRet = ExeFunction(gi\_sel\_menu, i-loc+1);

}

}

}

}

}

}

}

}

void PopPrompt(int num)

{

}

/\*\*

\* 函数名称: PopMenu

\* 函数功能: 弹出指定主菜单项对应的子菜单.

\* 输入参数: num 指定的主菜单项号

\* 输出参数: 无

\* 返 回 值: 无

\*

\*/

void PopMenu(int num)

{

LABEL\_BUNDLE labels;

HOT\_AREA areas;

SMALL\_RECT rcPop;

COORD pos;

WORD att;

char \*pCh;

int i, j, loc = 0;

if (num != gi\_sel\_menu) /\*如果指定主菜单不是已选中菜单\*/

{

if (gp\_top\_layer->LayerNo != 0) /\*如果此前已有子菜单弹出\*/

{

PopOff();

gi\_sel\_sub\_menu = 0;

}

}

else if (gp\_top\_layer->LayerNo != 0) /\*若已弹出该子菜单，则返回\*/

{

return;

}

gi\_sel\_menu = num; /\*将选中主菜单项置为指定的主菜单项\*/

TagMainMenu(gi\_sel\_menu); /\*在选中的主菜单项上做标记\*/

LocSubMenu(gi\_sel\_menu, &rcPop); /\*计算弹出子菜单的区域位置, 存放在rcPop中\*/

/\*计算该子菜单中的第一项在子菜单字符串数组中的位置(下标)\*/

for (i=1; i<gi\_sel\_menu; i++)

{

loc += ga\_sub\_menu\_count[i-1];

}

/\*将该组子菜单项项名存入标签束结构变量\*/

labels.ppLabel = ga\_sub\_menu + loc; /\*标签束第一个标签字符串的地址\*/

labels.num = ga\_sub\_menu\_count[gi\_sel\_menu-1]; /\*标签束中标签字符串的个数\*/

COORD aLoc[labels.num];/\*定义一个坐标数组，存放每个标签字符串输出位置的坐标\*/

for (i=0; i<labels.num; i++) /\*确定标签字符串的输出位置，存放在坐标数组中\*/

{

aLoc[i].X = rcPop.Left + 2;

aLoc[i].Y = rcPop.Top + i + 1;

}

labels.pLoc = aLoc; /\*使标签束结构变量labels的成员pLoc指向坐标数组的首元素\*/

/\*设置热区信息\*/

areas.num = labels.num; /\*热区的个数，等于标签的个数，即子菜单的项数\*/

SMALL\_RECT aArea[areas.num]; /\*定义数组存放所有热区位置\*/

char aSort[areas.num]; /\*定义数组存放所有热区对应类别\*/

char aTag[areas.num]; /\*定义数组存放每个热区的编号\*/

for (i=0; i<areas.num; i++)

{

aArea[i].Left = rcPop.Left + 2; /\*热区定位\*/

aArea[i].Top = rcPop.Top + i + 1;

aArea[i].Right = rcPop.Right - 2;

aArea[i].Bottom = aArea[i].Top;

aSort[i] = 0; /\*热区类别都为0(按钮型)\*/

aTag[i] = i + 1; /\*热区按顺序编号\*/

}

areas.pArea = aArea;/\*使热区结构变量areas的成员pArea指向热区位置数组首元素\*/

areas.pSort = aSort;/\*使热区结构变量areas的成员pSort指向热区类别数组首元素\*/

areas.pTag = aTag; /\*使热区结构变量areas的成员pTag指向热区编号数组首元素\*/

att = BACKGROUND\_BLUE | BACKGROUND\_GREEN | BACKGROUND\_RED; /\*白底黑字\*/

PopUp(&rcPop, att, &labels, &areas);

DrawBox(&rcPop); /\*给弹出窗口画边框\*/

pos.X = rcPop.Left + 2;

for (pos.Y=rcPop.Top+1; pos.Y<rcPop.Bottom; pos.Y++)

{ /\*此循环用来在空串子菜项位置画线形成分隔，并取消此菜单项的热区属性\*/

pCh = ga\_sub\_menu[loc+pos.Y-rcPop.Top-1];

if (strlen(pCh)==0) /\*串长为0，表明为空串\*/

{ /\*首先画横线\*/

FillConsoleOutputCharacter(gh\_std\_out, '-', rcPop.Right-rcPop.Left-3, pos, &ul);

for (j=rcPop.Left+2; j<rcPop.Right-1; j++)

{ /\*取消该区域字符单元的热区属性\*/

gp\_scr\_att[pos.Y\*SCR\_COL+j] &= 3; /\*按位与的结果保留了低两位\*/

}

}

}

/\*将子菜单项的功能键设为白底红字\*/

pos.X = rcPop.Left + 3;

att = FOREGROUND\_RED | BACKGROUND\_BLUE | BACKGROUND\_GREEN | BACKGROUND\_RED;

for (pos.Y=rcPop.Top+1; pos.Y<rcPop.Bottom; pos.Y++)

{

if (strlen(ga\_sub\_menu[loc+pos.Y-rcPop.Top-1])==0)

{

continue; /\*跳过空串\*/

}

FillConsoleOutputAttribute(gh\_std\_out, att, 1, pos, &ul);

}

return;

}

/\*\*

\* 函数名称: PopUp

\* 函数功能: 在指定区域输出弹出窗口信息, 同时设置热区, 将弹出窗口位置信息入栈.

\* 输入参数: pRc 弹出窗口位置数据存放的地址

\* att 弹出窗口区域字符属性

\* pLabel 弹出窗口中标签束信息存放的地址

pHotArea 弹出窗口中热区信息存放的地址

\* 输出参数: 无

\* 返 回 值: 无

\*

\* 调用说明:

\*/

void PopUp(SMALL\_RECT \*pRc, WORD att, LABEL\_BUNDLE \*pLabel, HOT\_AREA \*pHotArea)

{

LAYER\_NODE \*nextLayer;

COORD size;

COORD pos = {0, 0};

char \*pCh;

int i, j, row;

/\*弹出窗口所在位置字符单元信息入栈\*/

size.X = pRc->Right - pRc->Left + 1; /\*弹出窗口的宽度\*/

size.Y = pRc->Bottom - pRc->Top + 1; /\*弹出窗口的高度\*/

/\*申请存放弹出窗口相关信息的动态存储区\*/

nextLayer = (LAYER\_NODE \*)malloc(sizeof(LAYER\_NODE));

nextLayer->next = gp\_top\_layer;

nextLayer->LayerNo = gp\_top\_layer->LayerNo + 1;

nextLayer->rcArea = \*pRc;

nextLayer->pContent = (CHAR\_INFO \*)malloc(size.X\*size.Y\*sizeof(CHAR\_INFO));

nextLayer->pScrAtt = (char \*)malloc(size.X\*size.Y\*sizeof(char));

pCh = nextLayer->pScrAtt;

/\*将弹出窗口覆盖区域的字符信息保存，用于在关闭弹出窗口时恢复原样\*/

ReadConsoleOutput(gh\_std\_out, nextLayer->pContent, size, pos, pRc);

for (i=pRc->Top; i<=pRc->Bottom; i++)

{ /\*此二重循环将所覆盖字符单元的原先属性值存入动态存储区，便于以后恢复\*/

for (j=pRc->Left; j<=pRc->Right; j++)

{

\*pCh = gp\_scr\_att[i\*SCR\_COL+j];

pCh++;

}

}

gp\_top\_layer = nextLayer; /\*完成弹出窗口相关信息入栈操作\*/

/\*设置弹出窗口区域字符的新属性\*/

pos.X = pRc->Left;

pos.Y = pRc->Top;

for (i=pRc->Top; i<=pRc->Bottom; i++)

{

FillConsoleOutputAttribute(gh\_std\_out, att, size.X, pos, &ul);

pos.Y++;

}

/\*将标签束中的标签字符串在设定的位置输出\*/

for (i=0; i<pLabel->num; i++)

{

pCh = pLabel->ppLabel[i];

if (strlen(pCh) != 0)

{

WriteConsoleOutputCharacter(gh\_std\_out, pCh, strlen(pCh),

pLabel->pLoc[i], &ul);

}

}

/\*设置弹出窗口区域字符单元的新属性\*/

for (i=pRc->Top; i<=pRc->Bottom; i++)

{ /\*此二重循环设置字符单元的层号\*/

for (j=pRc->Left; j<=pRc->Right; j++)

{

gp\_scr\_att[i\*SCR\_COL+j] = gp\_top\_layer->LayerNo;

}

}

for (i=0; i<pHotArea->num; i++)

{ /\*此二重循环设置所有热区中字符单元的热区类型和热区编号\*/

row = pHotArea->pArea[i].Top;

for (j=pHotArea->pArea[i].Left; j<=pHotArea->pArea[i].Right; j++)

{

gp\_scr\_att[row\*SCR\_COL+j] |= (pHotArea->pSort[i] << 6)

| (pHotArea->pTag[i] << 2);

}

}

return;

}

/\*\*

\* 函数名称: PopOff

\* 函数功能: 关闭顶层弹出窗口, 恢复覆盖区域原外观和字符单元原属性.

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: 无

\*

\*/

void PopOff(void)

{

LAYER\_NODE \*nextLayer;

COORD size;

COORD pos = {0, 0};

char \*pCh;

int i, j;

if ((gp\_top\_layer->next==NULL) || (gp\_top\_layer->pContent==NULL))

{ /\*栈底存放的主界面屏幕信息，不用关闭\*/

return;

}

nextLayer = gp\_top\_layer->next;

/\*恢复弹出窗口区域原外观\*/

size.X = gp\_top\_layer->rcArea.Right - gp\_top\_layer->rcArea.Left + 1;

size.Y = gp\_top\_layer->rcArea.Bottom - gp\_top\_layer->rcArea.Top + 1;

WriteConsoleOutput(gh\_std\_out, gp\_top\_layer->pContent, size, pos, &(gp\_top\_layer->rcArea));

/\*恢复字符单元原属性\*/

pCh = gp\_top\_layer->pScrAtt;

for (i=gp\_top\_layer->rcArea.Top; i<=gp\_top\_layer->rcArea.Bottom; i++)

{

for (j=gp\_top\_layer->rcArea.Left; j<=gp\_top\_layer->rcArea.Right; j++)

{

gp\_scr\_att[i\*SCR\_COL+j] = \*pCh;

pCh++;

}

}

free(gp\_top\_layer->pContent); /\*释放动态存储区\*/

free(gp\_top\_layer->pScrAtt);

free(gp\_top\_layer);

gp\_top\_layer = nextLayer;

gi\_sel\_sub\_menu = 0;

return;

}

/\*\*

\* 函数名称: DrawBox

\* 函数功能: 在指定区域画边框.

\* 输入参数: pRc 存放区域位置信息的地址

\* 输出参数: 无

\* 返 回 值: 无

\*

\*/

void DrawBox(SMALL\_RECT \*pRc)

{

char chBox[] = {'+','-','|'}; /\*画框用的字符\*/

COORD pos = {pRc->Left, pRc->Top}; /\*定位在区域的左上角\*/

WriteConsoleOutputCharacter(gh\_std\_out, &chBox[0], 1, pos, &ul);/\*画边框左上角\*/

for (pos.X = pRc->Left + 1; pos.X < pRc->Right; pos.X++)

{ /\*此循环画上边框横线\*/

WriteConsoleOutputCharacter(gh\_std\_out, &chBox[1], 1, pos, &ul);

}

pos.X = pRc->Right;

WriteConsoleOutputCharacter(gh\_std\_out, &chBox[0], 1, pos, &ul);/\*画边框右上角\*/

for (pos.Y = pRc->Top+1; pos.Y < pRc->Bottom; pos.Y++)

{ /\*此循环画边框左边线和右边线\*/

pos.X = pRc->Left;

WriteConsoleOutputCharacter(gh\_std\_out, &chBox[2], 1, pos, &ul);

pos.X = pRc->Right;

WriteConsoleOutputCharacter(gh\_std\_out, &chBox[2], 1, pos, &ul);

}

pos.X = pRc->Left;

pos.Y = pRc->Bottom;

WriteConsoleOutputCharacter(gh\_std\_out, &chBox[0], 1, pos, &ul);/\*画边框左下角\*/

for (pos.X = pRc->Left + 1; pos.X < pRc->Right; pos.X++)

{ /\*画下边框横线\*/

WriteConsoleOutputCharacter(gh\_std\_out, &chBox[1], 1, pos, &ul);

}

pos.X = pRc->Right;

WriteConsoleOutputCharacter(gh\_std\_out, &chBox[0], 1, pos, &ul);/\*画边框右下角\*/

return;

}

/\*\*

\* 函数名称: TagSubMenu

\* 函数功能: 在指定子菜单项上做选中标记.

\* 输入参数: num 选中的子菜单项号

\* 输出参数: 无

\* 返 回 值: 无

\*

\*/

void TagSubMenu(int num)

{

SMALL\_RECT rcPop;

COORD pos;

WORD att;

int width;

LocSubMenu(gi\_sel\_menu, &rcPop); /\*计算弹出子菜单的区域位置, 存放在rcPop中\*/

if ((num<1) || (num == gi\_sel\_sub\_menu) || (num>rcPop.Bottom-rcPop.Top-1))

{ /\*如果子菜单项号越界，或该项子菜单已被选中，则返回\*/

return;

}

pos.X = rcPop.Left + 2;

width = rcPop.Right - rcPop.Left - 3;

if (gi\_sel\_sub\_menu != 0) /\*首先取消原选中子菜单项上的标记\*/

{

pos.Y = rcPop.Top + gi\_sel\_sub\_menu;

att = BACKGROUND\_BLUE | BACKGROUND\_GREEN | BACKGROUND\_RED; /\*白底黑字\*/

FillConsoleOutputAttribute(gh\_std\_out, att, width, pos, &ul);

pos.X += 1;

att |= FOREGROUND\_RED;/\*白底红字\*/

FillConsoleOutputAttribute(gh\_std\_out, att, 1, pos, &ul);

}

/\*在制定子菜单项上做选中标记\*/

pos.X = rcPop.Left + 2;

pos.Y = rcPop.Top + num;

att = FOREGROUND\_BLUE | FOREGROUND\_GREEN | FOREGROUND\_RED; /\*黑底白字\*/

FillConsoleOutputAttribute(gh\_std\_out, att, width, pos, &ul);

gi\_sel\_sub\_menu = num; /\*修改选中子菜单项号\*/

return;

}

/\*\*

\* 函数名称: LocSubMenu

\* 函数功能: 计算弹出子菜单区域左上角和右下角的位置.

\* 输入参数: num 选中的主菜单项号

\* 输出参数: rc 存放区域位置信息的地址

\* 返 回 值: 无

\*

\*/

void LocSubMenu(int num, SMALL\_RECT \*rc)

{

int i, len, loc = 0;

rc->Top = 1; /\*区域的上边定在第2行，行号为1\*/

rc->Left = 1;

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

{ /\*计算区域左边界位置, 同时计算第一个子菜单项在子菜单字符串数组中的位置\*/

rc->Left += strlen(ga\_main\_menu[i-1]) + 4;

loc += ga\_sub\_menu\_count[i-1];

}

rc->Right = strlen(ga\_sub\_menu[loc]);/\*暂时存放第一个子菜单项字符串长度\*/

for (i=1; i<ga\_sub\_menu\_count[num-1]; i++)

{ /\*查找最长子菜单字符串，将其长度存放在rc->Right\*/

len = strlen(ga\_sub\_menu[loc+i]);

if (rc->Right < len)

{

rc->Right = len;

}

}

rc->Right += rc->Left + 3; /\*计算区域的右边界\*/

rc->Bottom = rc->Top + ga\_sub\_menu\_count[num-1] + 1;/\*计算区域下边的行号\*/

if (rc->Right >= SCR\_COL) /\*右边界越界的处理\*/

{

len = rc->Right - SCR\_COL + 1;

rc->Left -= len;

rc->Right = SCR\_COL - 1;

}

return;

}

/\*\*

\* 函数名称: DealInput

\* 函数功能: 在弹出窗口区域设置热区, 等待并相应用户输入.

\* 输入参数: pHotArea

\* piHot 焦点热区编号的存放地址, 即指向焦点热区编号的指针

\* 输出参数: piHot 用鼠标单击、按回车或空格时返回当前热区编号

\* 返 回 值:

\*

\*/

int DealInput(HOT\_AREA \*pHotArea, int \*piHot)

{

INPUT\_RECORD inRec;

DWORD res;

COORD pos = {0, 0};

int num, arrow, iRet = 0;

int cNo, cTag, cSort;/\*cNo:层号, cTag:热区编号, cSort: 热区类型\*/

char vkc, asc; /\*vkc:虚拟键代码, asc:字符的ASCII码值\*/

SetHotPoint(pHotArea, \*piHot);

while (TRUE)

{ /\*循环\*/

ReadConsoleInput(gh\_std\_in, &inRec, 1, &res);

if ((inRec.EventType == MOUSE\_EVENT) &&

(inRec.Event.MouseEvent.dwButtonState

== FROM\_LEFT\_1ST\_BUTTON\_PRESSED))

{

pos = inRec.Event.MouseEvent.dwMousePosition;

cNo = gp\_scr\_att[pos.Y \* SCR\_COL + pos.X] & 3;

cTag = (gp\_scr\_att[pos.Y \* SCR\_COL + pos.X] >> 2) & 15;

cSort = (gp\_scr\_att[pos.Y \* SCR\_COL + pos.X] >> 6) & 3;

if ((cNo == gp\_top\_layer->LayerNo) && cTag > 0)

{

\*piHot = cTag;

SetHotPoint(pHotArea, \*piHot);

if (cSort == 0)

{

iRet = 13;

break;

}

}

}

else if (inRec.EventType == KEY\_EVENT && inRec.Event.KeyEvent.bKeyDown)

{

vkc = inRec.Event.KeyEvent.wVirtualKeyCode;

asc = inRec.Event.KeyEvent.uChar.AsciiChar;;

if (asc == 0)

{

arrow = 0;

switch (vkc)

{ /\*方向键(左、上、右、下)的处理\*/

case 37: arrow = 1; break;

case 38: arrow = 2; break;

case 39: arrow = 3; break;

case 40: arrow = 4; break;

}

if (arrow > 0)

{

num = \*piHot;

while (TRUE)

{

if (arrow < 3)

{

num--;

}

else

{

num++;

}

if ((num < 1) || (num > pHotArea->num) ||

((arrow % 2) && (pHotArea->pArea[num-1].Top

== pHotArea->pArea[\*piHot-1].Top)) || ((!(arrow % 2))

&& (pHotArea->pArea[num-1].Top

!= pHotArea->pArea[\*piHot-1].Top)))

{

break;

}

}

if (num > 0 && num <= pHotArea->num)

{

\*piHot = num;

SetHotPoint(pHotArea, \*piHot);

}

}

}

else if (vkc == 27)

{ /\*ESC键\*/

iRet = 27;

break;

}

else if (vkc == 13 || vkc == 32)

{ /\*回车键或空格表示按下当前按钮\*/

iRet = 13;

break;

}

}

}

return iRet;

}

void SetHotPoint(HOT\_AREA \*pHotArea, int iHot)

{

CONSOLE\_CURSOR\_INFO lpCur;

COORD pos = {0, 0};

WORD att1, att2;

int i, width;

att1 = FOREGROUND\_BLUE | FOREGROUND\_GREEN | FOREGROUND\_RED; /\*黑底白字\*/

att2 = BACKGROUND\_BLUE | BACKGROUND\_GREEN | BACKGROUND\_RED; /\*白底黑字\*/

for (i=0; i<pHotArea->num; i++)

{ /\*将按钮类热区置为白底黑字\*/

pos.X = pHotArea->pArea[i].Left;

pos.Y = pHotArea->pArea[i].Top;

width = pHotArea->pArea[i].Right - pHotArea->pArea[i].Left + 1;

if (pHotArea->pSort[i] == 0)

{ /\*热区是按钮类\*/

FillConsoleOutputAttribute(gh\_std\_out, att2, width, pos, &ul);

}

}

pos.X = pHotArea->pArea[iHot-1].Left;

pos.Y = pHotArea->pArea[iHot-1].Top;

width = pHotArea->pArea[iHot-1].Right - pHotArea->pArea[iHot-1].Left + 1;

if (pHotArea->pSort[iHot-1] == 0)

{ /\*被激活热区是按钮类\*/

FillConsoleOutputAttribute(gh\_std\_out, att1, width, pos, &ul);

}

else if (pHotArea->pSort[iHot-1] == 1)

{ /\*被激活热区是文本框类\*/

SetConsoleCursorPosition(gh\_std\_out, pos);

GetConsoleCursorInfo(gh\_std\_out, &lpCur);

lpCur.bVisible = TRUE;

SetConsoleCursorInfo(gh\_std\_out, &lpCur);

}

}

/\*\*

\* 函数名称: ExeFunction

\* 函数功能: 执行由主菜单号和子菜单号确定的功能函数.

\* 输入参数: m 主菜单项号

\* s 子菜单项号

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE 或 FALSE

\*

\* 调用说明: 仅在执行函数ExitSys时, 才可能返回FALSE, 其他情况下总是返回TRUE

\*/

BOOL ExeFunction(int m, int s)

{

BOOL bRet = TRUE;

/\*函数指针数组，用来存放所有功能函数的入口地址\*/

BOOL (\*pFunction[ga\_sub\_menu\_count[0]+ga\_sub\_menu\_count[1]+ga\_sub\_menu\_count[2]+ga\_sub\_menu\_count[3]+ga\_sub\_menu\_count[4]])(void);

int i, loc;

/\*将功能函数入口地址存入与功能函数所在主菜单号和子菜单号对应下标的数组元素\*/

pFunction[0] = SaveData;

pFunction[1] = BackupData;

pFunction[2] = RestoreData;

pFunction[3] = ExitSys;

pFunction[4] = MaintainLessonInfo;

pFunction[5] = MaintainCourseInfo;

pFunction[6] = NULL;

pFunction[7] = MaintainTeacherInfo;

pFunction[8] = MaintainClassroomInfo;

pFunction[9] = MaintainClassnoInfo;

pFunction[10] = QueryCourseByTeacher;

pFunction[11] = QueryLessonByClassno;

pFunction[12] = NULL;

pFunction[13] = QueryLessonByTimeAndPlace;

pFunction[14] = QueryLessonByCourse;

pFunction[15] = QuerySpareClassroom;

pFunction[16] = StatClassroomRate;

pFunction[17] = StatTeacherRate;

pFunction[18] = StatClassnoRate;

pFunction[19] = TraverseAll;

pFunction[20] = HelpTopic;

pFunction[21] = NULL;

pFunction[22] = About;

for (i=1,loc=0; i<m; i++) /\*根据主菜单号和子菜单号计算对应下标\*/

{

loc += ga\_sub\_menu\_count[i-1];

}

loc += s - 1;

if (pFunction[loc] != NULL)

{

bRet = (\*pFunction[loc])(); /\*用函数指针调用所指向的功能函数\*/

}

return bRet;

}

/\*\*

\* 函数名称: SaveData

\* 函数功能: 向数据文件中保存数据

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE(成功) 或 FALSE(失败)

\*

\*/

BOOL SaveData(void)

{

BOOL bRet = TRUE;

char \*plabel\_name[] = {"主菜单项：文件",

"子菜单项：数据保存",

"确认"

};

ShowModule(plabel\_name, 3);

FILE\* fp1=fopen("data\\lesson\_to\_save.txt","w+");

FILE\* fp2=fopen("data\\gen\_info.txt","w+");

/\* FILE\* fp3=fopen("data\\class.txt","w+");

FILE\* fp4=fopen("data\\.txt","w+");\*/

if(fp1==NULL || fp2==NULL)

{

MessageBox(NULL,"数据保存失败，可能是数据文件损坏！","ERROR!",MB\_OK);

return 0;

}

ToSaveData(fp1,fp2);

MessageBox(NULL," 数据保存完成！","SUCCESS!",MB\_OK);

return bRet;

}

/\*\*

\* 函数名称: BackupData

\* 函数功能: 生成数据备份文件并保存数据

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE(成功) 或 FALSE(失败)

\*

\*/

BOOL BackupData(void)

{

BOOL bRet = TRUE;

char \*plabel\_name[] = {"主菜单项：文件",

"子菜单项：数据备份",

"确认"

};

ShowModule(plabel\_name, 3);

char filename1[100];

char filename2[100];

GetBackupFilename(filename1, filename2 );//获得包含备份路径的备份文件名

int res = ToBackup(filename1,filename2);//保存备份信息

if(res==-1)

{

MessageBox(NULL," 数据备份失败 ！","ERROR!",MB\_OK);

bRet=0;

}

else

MessageBox(NULL," 数据备份完成！","SUCCESS!",MB\_OK);

return bRet;

}

/\*\*

\* 函数名称: RestoreData

\* 函数功能: 从备份文件中恢复数据

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE(成功) 或 FALSE(失败)

\*

\*/

BOOL RestoreData(void)

{

BOOL bRet = TRUE;

char \*plabel\_name[] = {"主菜单项：文件",

"子菜单项：数据恢复",

"确认"

};

ShowModule(plabel\_name, 3);

/\*显示光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=TRUE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

/\*打开备份文件所在目录\*/

char dir\_togo[100];//存放要去的那个目录

char DataDir[]="\\backup";//存放数据文件的那一层目录

char cmd[110]="explorer ";

getcwd(dir\_togo,sizeof(dir\_togo));//把当前目录地址字符串复制到dir\_togo中

strcat(dir\_togo,DataDir);//在当前工作地址之后加上\data构造数据文件夹所在目录

/\*\*注意不用AddBackslash(dir\_togo)，system函数与fopen等函数不同，表达\不需要用\\转意

而且注意，比如char DataDir[7]="\\data"内存单元中只有一个\,转意的意思是用\\来表示\而已 \*/

strcat(cmd,dir\_togo);//构造打开目录的cmd命令

system(cmd);

/\*读入文件名称信息\*/

char filename1[100]="backup\\";

char filename2[100]="backup\\";

char filename[100];

Flag1:

printf("请输入课程信息备份文件名(含lesson前缀和.txt后缀): \n");

scanf("%s",filename);

if( (!strstr(filename,"lesson")) || (!strstr(filename,".txt")) )//判断文件名的正确性

{

// MessageBox(NULL,"文件名错误！文件名含有lesson前缀和txt后缀\n","ERROR!",MB\_OK);

int result = MessageBox(NULL,"文件名错误！文件名含有lesson前缀和.txt后缀\n是否重新输入？","ERROR！",MB\_ICONINFORMATION|MB\_YESNO);

if(result==IDYES)

{

ClearScreenTotalInputArea();

goto Flag1;

}

else

{

ClearScreenTotalInputArea();//清屏

/\*隐藏光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=FALSE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

return 1;

}

}

strcat(filename1,filename);

printf("请输入综合教学信息本分文件名(含gen\_info前缀和.txt后缀):\n");

scanf("%s",filename);

if( (!strstr(filename,"gen\_info")) || (!strstr(filename,".txt")) )//判断文件名的正确性

{

//MessageBox(NULL,"文件名错误！文件名含有lesson前缀和txt后缀\n","ERROR!",MB\_OK);

int result = MessageBox(NULL,"文件名错误！文件名含有gen\_info前缀和.txt后缀\n是否重新输入？","ERROR！",MB\_ICONINFORMATION|MB\_YESNO);

if(result==IDYES)

{

ClearScreenTotalInputArea();

goto Flag1;

}

else

{

ClearScreenTotalInputArea();//清屏

/\*隐藏光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=FALSE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

return 1;

}

}

strcat(filename2,filename);

FILE\* fp1=fopen(filename1,"r+");

FILE\* fp2=fopen(filename2,"r+");

if(fp1==NULL || fp2==NULL)

{

MessageBox(NULL,"数据恢复失败！可能是备份文件信息损坏!\n","ERROR!",MB\_OK);

return bRet=0;

}

//先清除内存中的信息再重新载入

DestroyLesson();

DestroyOtherVectorInfo();

int res1=ReadLesson(fp1);

int res2=ReadGeneralInfo(fp2);

// printf("res1: %d res2: %d\n",res1,res2);

if(res1==-1 || res2==-1)

{

MessageBox(NULL,"数据恢复失败!排课信息有误！\n原因可能是教师上课冲突，教室利用冲突等错误！","ERROR!",MB\_OK);

return bRet=0;

}

else

{

MessageBox(NULL,"数据恢复成功！","SUCCESS!",MB\_OK);

ClearScreenTotalInputArea();//清屏

/\*隐藏光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=FALSE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

bRet=1;

}

return bRet;

}

/\*\*

\* 函数名称: ExitSys

\* 函数功能: 退出系统

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE(成功) 或 FALSE(失败)

\*

\*/

BOOL ExitSys(void)

{

LABEL\_BUNDLE labels;

HOT\_AREA areas;

BOOL bRet = TRUE;

SMALL\_RECT rcPop;

COORD pos;

WORD att;

char \*pCh[] = {"确认退出系统吗？", "确定 取消"};

int iHot = 1;

pos.X = strlen(pCh[0]) + 6;

pos.Y = 7;

rcPop.Left = (SCR\_COL - pos.X) / 2;

rcPop.Right = rcPop.Left + pos.X - 1;

rcPop.Top = (SCR\_ROW - pos.Y) / 2;

rcPop.Bottom = rcPop.Top + pos.Y - 1;

att = BACKGROUND\_BLUE | BACKGROUND\_GREEN | BACKGROUND\_RED; /\*白底黑字\*/

labels.num = 2;

labels.ppLabel = pCh;

COORD aLoc[] = {{rcPop.Left+3, rcPop.Top+2},

{rcPop.Left+5, rcPop.Top+5}};

labels.pLoc = aLoc;

areas.num = 2;

SMALL\_RECT aArea[] = {{rcPop.Left + 5, rcPop.Top + 5,

rcPop.Left + 8, rcPop.Top + 5},

{rcPop.Left + 13, rcPop.Top + 5,

rcPop.Left + 16, rcPop.Top + 5}};

char aSort[] = {0, 0};

char aTag[] = {1, 2};

areas.pArea = aArea;

areas.pSort = aSort;

areas.pTag = aTag;

PopUp(&rcPop, att, &labels, &areas);

pos.X = rcPop.Left + 1;

pos.Y = rcPop.Top + 4;

FillConsoleOutputCharacter(gh\_std\_out, '-', rcPop.Right-rcPop.Left-1, pos, &ul);

if (DealInput(&areas, &iHot) == 13 && iHot == 1)

{

bRet = FALSE;

}

else

{

bRet = TRUE;

}

PopOff();

return bRet;

}

/\*\*

\* 函数名称: MaintainLessonInfo

\* 函数功能: 课堂信息维护

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE(成功) 或 FALSE(失败)

\*

\*/

BOOL MaintainLessonInfo(void)

{

BOOL bRet = TRUE;

char \*plabel\_name[] = {"主菜单项：数据维护",

"子菜单项：课堂信息",

"确认"

};

ShowModule(plabel\_name, 3);

/\*显示光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=TRUE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

Flag1:

printf("请输入要执行的操作：\n");

printf("(I)插入课堂\n");

printf("(D)删除课堂\n");

printf("(M)修改课堂\n");

printf("(L)退出\n");

printf("\n请选择：");

char ch;

scanf("%1s",&ch);

getchar();

ClearScreenTotalInputArea();//清屏

switch(ch)

{

case 'i': case 'I': ToInsertLesson();break;

case 'd': case 'D': ToDeleteLesson();break;

case 'm': case 'M': {ToModifyLesson()==-1 ? bRet=0 : bRet=1; break;}

case 'l': case 'L': break;

default : {printf("\n!!命令输入错误！！请重新输入...\n\n");fflush(stdin);goto Flag1;}

}

/\*隐藏光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=FALSE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

ClearScreenTotalInputArea();//清屏

return bRet;

}

/\*\*

\* 函数名称: MaintainCourseInfo

\* 函数功能: 课程信息维护

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE(成功) 或 FALSE(失败)

\*

\*/

BOOL MaintainCourseInfo(void)

{

BOOL bRet = TRUE;

char \*plabel\_name[] = {"主菜单项：数据维护",

"子菜单项：课程信息",

"确认"

};

ShowModule(plabel\_name, 3);

/\*显示光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=TRUE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

Flag1:

printf("请输入要执行的操作：\n");

printf("(I)插入课程\n");

printf("(D)删除课程\n");

printf("(M)修改课程\n");

printf("(L)退出\n");

printf("\n请选择：");

char ch;

scanf("%1s",&ch);

getchar();

ClearScreenTotalInputArea();//清屏

switch(ch)

{

case 'i': case 'I': ToInsertCourse();break;

case 'd': case 'D': ToDeleteCourse();break;

case 'm': case 'M': ToModifyCourse(); break;

case 'l': case 'L': break;

default : {printf("\n!!命令输入错误！！请重新输入...\n\n");fflush(stdin);goto Flag1;}

}

/\*隐藏光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=FALSE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

ClearScreenTotalInputArea();//清屏

return bRet;

}

/\*\*

\* 函数名称: MaintainTeacherInfo

\* 函数功能: 教师信息维护

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE(成功) 或 FALSE(失败)

\*

\*/

BOOL MaintainTeacherInfo(void)

{

BOOL bRet = TRUE;

char \*plabel\_name[] = {"主菜单项：数据维护",

"子菜单项：教师信息",

"确认"

};

ShowModule(plabel\_name, 3);

/\*显示光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=TRUE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

Flag1:

printf("请输入要执行的操作：\n");

printf("(I)插入教师\n");

printf("(D)删除教师\n");

printf("(M)修改教师信息\n");

printf("(L)退出\n");

printf("\n请选择：");

char ch;

scanf("%1s",&ch);

getchar();

ClearScreenTotalInputArea();//清屏

switch(ch)

{

case 'i': case 'I': ToInsertTeacher();break;

case 'd': case 'D': ToDeleteTeacher();break;

case 'm': case 'M': ToModifyTeacher(); break;

case 'l': case 'L': break;

default : {printf("\n!!命令输入错误！！请重新输入...\n\n");fflush(stdin);goto Flag1;}

}

/\*隐藏光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=FALSE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

ClearScreenTotalInputArea();//清屏

return bRet;

}

/\*\*

\* 函数名称: MaintainClassroomInfo

\* 函数功能: 教室信息维护

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE(成功) 或 FALSE(失败)

\*

\*/

BOOL MaintainClassroomInfo(void)

{

BOOL bRet = TRUE;

char \*plabel\_name[] = {"主菜单项：数据维护",

"子菜单项：教室信息",

"确认"

};

ShowModule(plabel\_name, 3);

LessonFileForm();

system(".\\data\\lesson\_to\_read.txt");

//system(".\\data\\lesson\_to\_save.txt");

int res=ReloadData();

if(res==-1)

{

MessageBox(NULL,"修改错误！可能发生了上课冲突！","ERROR!",MB\_ICONINFORMATION|MB\_OK);

bRet=0;

}

else

{

MessageBox(NULL,"修改成功！","SUCCESS!",MB\_OK);

bRet=1;

}

ClearScreenTotalInputArea();//清屏

return bRet;

}

/\*\*

\* 函数名称: MaintainClassnoInfo

\* 函数功能: 班级信息维护

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE(成功) 或 FALSE(失败)

\*

\*/

BOOL MaintainClassnoInfo(void)

{

BOOL bRet = TRUE;

char \*plabel\_name[] = {"主菜单项：数据维护",

"子菜单项：班级信息",

"确认"

};

ShowModule(plabel\_name, 3);

LessonFileForm();

system(".\\data\\lesson\_to\_read.txt");

//system(".\\data\\lesson\_to\_save.txt");

int res=ReloadData();

if(res==-1)

{

MessageBox(NULL,"修改错误！可能发生了上课冲突！","ERROR!",MB\_ICONINFORMATION|MB\_OK);

bRet=0;

}

else

{

MessageBox(NULL,"修改成功！","SUCCESS!",MB\_OK);

bRet=1;

}

ClearScreenTotalInputArea();//清屏

return bRet;

}

/\*\*

\* 函数名称: QueryCourseByTeacher

\* 函数功能: 通过教师查询课程信息

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE(成功)

\*

\*/

BOOL QueryCourseByTeacher(void)

{

BOOL bRet = TRUE;

char \*plabel\_name[] = {"主菜单项：数据查询",

"子菜单项：通过教师查询课程",

"确认"

};

ShowModule(plabel\_name, 3);

/\*显示光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=TRUE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

ToQueryCourseByTeacher();

/\*隐藏光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=FALSE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

ClearScreenTotalInputArea();//清屏

return bRet;

}

/\*\*

\* 函数名称: QueryLessonByClassno

\* 函数功能: 按班级检索课堂

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE(成功)

\*

\*/

BOOL QueryLessonByClassno(void)

{

BOOL bRet = TRUE;

char \*plabel\_name[] = {"主菜单项：数据查询",

"子菜单项：按班级检索课堂",

"确认"

};

ShowModule(plabel\_name, 3);

/\*显示光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=TRUE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

memset(readed,0,sizeof(readed));

ToQueryLessonByClassno();

memset(readed,0,sizeof(readed));

/\*隐藏光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=FALSE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

ClearScreenTotalInputArea();//清屏

return bRet;

}

/\*\*

\* 函数名称: QueryLessonByTimeAndPlace

\* 函数功能: 按教室、时间检索课堂

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE(成功)

\*

\*/

BOOL QueryLessonByTimeAndPlace(void)

{

BOOL bRet = TRUE;

char \*plabel\_name[] = {"主菜单项：数据查询",

"子菜单项：按教室、时间检索课堂",

"确认"

};

ShowModule(plabel\_name, 3);

/\*显示光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=TRUE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

//查询结果只有0或1个，而且不涉及readed

ToQueryLessonByTimeAndPlace();

/\*隐藏光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=FALSE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

ClearScreenTotalInputArea();//清屏

return bRet;

}

/\*\*

\* 函数名称: QueryLessonByCourse

\* 函数功能: 按课程检索课堂

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE(成功)

\*

\*/

BOOL QueryLessonByCourse(void)

{

BOOL bRet = TRUE;

char \*plabel\_name[] = {"主菜单项：数据查询",

"子菜单项：按课程检索课堂",

"确认"

};

ShowModule(plabel\_name, 3);

/\*显示光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=TRUE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

memset(readed,0,sizeof(readed));

ToQueryLessonByCourse();

memset(readed,0,sizeof(readed));

/\*隐藏光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=FALSE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

ClearScreenTotalInputArea();//清屏

return bRet;

}

/\*\*

\* 函数名称: QuerySpareClassroom

\* 函数功能: 空闲教室查询

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE(成功)

\*

\*/

BOOL QuerySpareClassroom(void)

{

BOOL bRet = TRUE;

char \*plabel\_name[] = {"主菜单项：数据查询",

"子菜单项：空闲教室查询",

"确认"

};

ShowModule(plabel\_name, 3);

/\*显示光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=TRUE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

memset(readed,0,sizeof(readed));

ToQuerySpareClassroom();

memset(readed,0,sizeof(readed));

/\*隐藏光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=FALSE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

ClearScreenTotalInputArea();//清屏

return bRet;

}

/\*\*

\* 函数名称: StatClassroomRate

\* 函数功能: 教室利用与能效分析

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE(成功)

\*

\*/

BOOL StatClassroomRate(void)

{

BOOL bRet = TRUE;

char \*plabel\_name[] = {"主菜单项：数据统计",

"子菜单项：教室利用与能效分析",

"确认"

};

ShowModule(plabel\_name, 3);

/\*显示光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=TRUE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

ToStatClassroomRate();

/\*隐藏光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=FALSE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

ClearScreenTotalInputArea();//清屏

return bRet;

}

/\*\*

\* 函数名称: StatTeacherRate

\* 函数功能: 教师任课分析

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE(成功)

\*

\*/

BOOL StatTeacherRate(void)

{

BOOL bRet = TRUE;

char \*plabel\_name[] = {"主菜单项：数据统计",

"子菜单项：教师任课分析",

"确认"

};

ShowModule(plabel\_name, 3);

/\*显示光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=TRUE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

ToStatTeacherRate();

/\*隐藏光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=FALSE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

ClearScreenTotalInputArea();//清屏

return bRet;

}

/\*\*

\* 函数名称: StatClassnoRate

\* 函数功能: 班级上课分析

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE(成功)

\*

\*/

BOOL StatClassnoRate(void)

{

BOOL bRet = TRUE;

char \*plabel\_name[] = {"主菜单项：数据统计",

"子菜单项：班级上课分析",

"确认"

};

ShowModule(plabel\_name, 3);

/\*显示光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=TRUE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

ToStatClassnoRate();

/\*隐藏光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=FALSE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

ClearScreenTotalInputArea();//清屏

return bRet;

}

/\*\*

\* 函数名称: TraverseAll

\* 函数功能: 遍历所有信息

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE(成功)

\*

\*/

BOOL TraverseAll(void)

{

BOOL bRet = TRUE;

char \*plabel\_name[] = {"主菜单项：数据统计",

"子菜单项：遍历所有信息",

"确认"

};

ShowModule(plabel\_name, 3);

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=TRUE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

ToTraverseAll();

/\*隐藏光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=FALSE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

ClearScreenTotalInputArea();//清屏

return bRet;

}

/\*\*

\* 函数名称: HelpTopic

\* 函数功能: 输出帮助信息

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE(成功)

\*

\*/

BOOL HelpTopic(void)

{

BOOL bRet = TRUE;

char \*plabel\_name[] = {"主菜单项：帮助",

"子菜单项：帮助主题",

"确认"

};

ShowModule(plabel\_name, 3);

system(".\\help\\help.docx");

return bRet;

}

/\*\*

\* 函数名称: About

\* 函数功能: 输出关于本程序的版本等等信息

\* 输入参数: 无

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE(成功)

\*

\*/

BOOL About(void)

{

BOOL bRet = TRUE;

char \*plabel\_name[] = {"主菜单项：帮助",

"子菜单项：关于...",

"确认"

};

ShowModule(plabel\_name, 3);

ShellExecute(NULL,"open","https://github.com/ShmilyBelon",NULL,NULL,SW\_SHOWNORMAL);

/\*显示光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=TRUE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

printf("\n开发者：陈国轩\n");

printf("\n单位：华中科技大学计算机科学与技术学院，计算机校交1601班\n");

printf("\n版本：win1709\n");

printf("\n参考文献：严蔚敏, 吴伟民. 数据结构（C语言版）. 北京: 清华大学出版社,1997\n");

printf(" 《C语言实验与课程设计》科学出版社， 李开 卢萍 曹计昌编著\n");

printf("\n\n鸣谢：魏巍老师\n\n");

printf("备注：此系统为windows版本，不支持在linux，Mac OS X等其他系统上运行！\n\n");

printf("按任意键继续...");

getch();

ClearScreenTotalInputArea();//清屏

/\*隐藏光标\*/

GetConsoleCursorInfo(gh\_std\_out,&lpCur);

lpCur.bVisible=FALSE;

SetConsoleCursorInfo(gh\_std\_out,&lpCur);

return bRet;

}

/\*\*

\* 函数名称: ShowModule

\* 函数功能: 按用户要求输出提示框

\* 输入参数: 提示框要输出的提示信息字符串地址，提示框输出行数

\* 输出参数: 无

\* 返 回 值: BOOL类型, TRUE(成功)

\*

\*/

BOOL ShowModule(char \*\*pString, int n)

{

LABEL\_BUNDLE labels;

HOT\_AREA areas;

BOOL bRet = TRUE;

SMALL\_RECT rcPop;

COORD pos;

WORD att;

int iHot = 1;

int i, maxlen, str\_len;

for (i=0,maxlen=0; i<n; i++) {

str\_len = strlen(pString[i]);

if (maxlen < str\_len) {

maxlen = str\_len;

}

}

pos.X = maxlen + 6;

pos.Y = n + 5;

rcPop.Left = (SCR\_COL - pos.X) / 2;

rcPop.Right = rcPop.Left + pos.X - 1;

rcPop.Top = (SCR\_ROW - pos.Y) / 2;

rcPop.Bottom = rcPop.Top + pos.Y - 1;

att = BACKGROUND\_BLUE | BACKGROUND\_GREEN | BACKGROUND\_RED; /\*白底黑字\*/

labels.num = n;

labels.ppLabel = pString;

COORD aLoc[n];

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

aLoc[i].X = rcPop.Left + 3;

aLoc[i].Y = rcPop.Top + 2 + i;

}

str\_len = strlen(pString[n-1]);

aLoc[n-1].X = rcPop.Left + 3 + (maxlen-str\_len)/2;

aLoc[n-1].Y = aLoc[n-1].Y + 2;

labels.pLoc = aLoc;

areas.num = 1;

SMALL\_RECT aArea[] = {{aLoc[n-1].X, aLoc[n-1].Y,

aLoc[n-1].X + 3, aLoc[n-1].Y}};

char aSort[] = {0};

char aTag[] = {1};

areas.pArea = aArea;

areas.pSort = aSort;

areas.pTag = aTag;

PopUp(&rcPop, att, &labels, &areas);

pos.X = rcPop.Left + 1;

pos.Y = rcPop.Top + 2 + n;

FillConsoleOutputCharacter(gh\_std\_out, '-', rcPop.Right-rcPop.Left-1, pos, &ul);

DealInput(&areas, &iHot);

PopOff();

return bRet;

}

1. func.cpp文件

#include"ExeFunction.h"

//char s[100000];

/\*\*

\* 函数名称: PlacestrToNum

\* 函数功能: 把教学楼教室字符串转化为数组下标

\* 输入参数: 教学楼、教室

\* 返 回 值: les\_place数组下标

\*

\*/

int PlacestrToNum(char s1[] , char s2[])//s1表示教学楼s2表示教室

{

if(!strcmp(s1,"西十二"))//1~120

{

int n=0;

if( (s2[0]!='N')&&(s2[0]!='S')&&(s2[0]!='n')&&(s2[0]!='s') )//判断南楼和北楼

return -1;

if((s2[1]>'5')||(s2[1]<'1')||(s2[2]>'1')||(s2[2]<'0') || (s2[3]<'0') || (s2[3]>'9'))

return -1 ;

if( (s2[0]=='S')||(s2[0]=='s'))

n += 60;

n += 12\*(s2[1]-'0'-1);//增加南楼偏置

if(s2[2]!='0')

{

if( (s2[3]!='0')&&(s2[3]!='1')&& (s2[3]!='2') )//一层只有12个教室

return -1;

n += (10+(s2[3]-'0'));

}

else if(s2[3]!='0')

n += (s2[3]-'0');

else

return -1;

return n;

}

else if(!strcmp(s1,"西五楼"))//131~230

{

if((s2[0]>'5')||(s2[0]<'1')||(s2[1]>'2')||(s2[1]<'0')||(s2[2]>'9')||(s2[2]<'0'))//如果出现非法的教室编号

return -1;

else

{

if((s2[1]=='0')&&(s2[2]=='0'))

return -1;

if((s2[1]=='2')&&(s2[2]!='0'))

return -1;

int n=0;

n += (s2[0]-'1')\*20 + (s2[1]-'0')\*10 + (s2[2]-'0');

n+=130;

return n;

}

}

else if(!strcmp(s1,"南一楼"))//121~130

{

if(!strcmp(s2,"116"))

return 121;

else if(!strcmp(s2,"802"))

return 122;

else if(!strcmp(s2,"803"))

return 123;

else if(!strcmp(s2,"804"))

return 124;

else if(!strcmp(s2,"805"))

return 125;

else if(!strcmp(s2,"806"))

return 126;

else if(!strcmp(s2,"807"))

return 127;

else if(!strcmp(s2,"808"))

return 128;

else if(!strcmp(s2,"809"))

return 129;

else if(!strcmp(s2,"810"))

return 130;

else

return -1;

}

else if(!strcmp(s1,"科技楼"))//240

{

if(!strcmp(s2,"北楼"))

return 240;

else

return -1;

}

else if(!strcmp(s1,"西操场"))

return 241;

else if(!strcmp(s1,"中心操场"))

return 242;

else if(!strcmp(s1,"东操场"))

return 243;

else

return -1;

}

/\*\*

\* 函数名称: TimestrToNum

\* 函数功能: 把周数、星期、节数转化为les\_time数组下标

\* 输入参数: 周数、星期、节数

\* 返 回 值: les\_time数组下标

\*

\*/

int TimestrToNum(int week , int day , int period)//计算时间hash值

{

if( (week<1)||(week>22) ||(day<1) || (day>7)||(period<1)||(period>6) )

return -1;//如果超出范围

int res =7\*6\*(week-1) + 6\*(day-1) +period;//计算时间hash值

if( res>840 || res <1)//hash值超出范围

return -1;

return res;

}

/\*\*

\* 函数名称: LessonInsert

\* 函数功能: 在对应的时间空间插入课堂

\* 输入参数: 空间hash值，时间hash值，要插入的课堂结构体指针

\* 返 回 值: 成功返回1，出错返回-1

\*

\*/

int LessonInsert(int placei,int timei,LESSON\* temp)

{

LESSON \*p=NULL;

//时间链、空间链对应的节点都为NULL

if((les\_place[placei].room\_next==NULL)&&(les\_time[timei].time\_next==NULL))

{

temp->lesson\_time=&(les\_time[timei]);temp->lesson\_place=&(les\_place[placei]);

// les\_time[timei].time\_next=(LESSON\*)malloc(sizeof(LESSON));

les\_time[timei].time\_next=temp;

//les\_place[placei].room\_next=(LESSON\*)malloc(sizeof(LESSON));

les\_place[placei].room\_next=temp;

return 1;

}

//空间链不空，时间链为NULL，也就是同一教室别的时间有课

else if((les\_place[placei].room\_next!=NULL)&&(les\_time[timei].time\_next==NULL))

{

//les\_time[timei].time\_next=(LESSON\*)malloc(sizeof(LESSON));

temp->lesson\_time=&(les\_time[timei]);

les\_time[timei].time\_next=temp;

//在该教室的对应时间位置上面插入

p=les\_place[placei].room\_next;

while(p!=NULL)

{

int t1=TimestrToNum(p->week , p->day , p->period);

if(t1<timei)

{

if(p->time\_next!=NULL)

{

int t2=TimestrToNum(p->time\_next->week , p->time\_next->day , p->time\_next->period);

if(t2<timei) {p=p->time\_next;continue;}//寻找合适的插入点

else if(t2>timei)

{

p->time\_next->time\_pri=temp; temp->time\_next=p->time\_next;

p->time\_next=temp; temp->time\_pri=p;

temp->lesson\_place=&(les\_place[placei]);

return 1;

}

else

{ //printf("11111\n" );

return -1;

}

}

else//p点后没有课堂了，此课堂在p点之后上课，那么插在p后面

{

temp->lesson\_place=&(les\_place[placei]);

p->time\_next=temp;temp->time\_pri=p;

return 1;

}

}

else if(t1>timei)//说明此课堂在p之前上课

{

temp->lesson\_place=&(les\_place[placei]);

les\_place[placei].room\_next=temp;

temp->time\_next=p; p->time\_pri=temp;

return 1;

}

else//同一地点的同一时间，上课冲突

{

//printf("22222\n" );

return -1;

}

}

}

//空间链为NULL，时间链不空，也就是同一时间有多个教室上课，插在对应的位置

else if((les\_place[placei].room\_next==NULL)&&(les\_time[timei].time\_next!=NULL))

{

temp->lesson\_place=&(les\_place[placei]);

les\_place[placei].room\_next=temp;

//在同一时间合适的教室插入该课堂

p=les\_time[timei].time\_next;

while(p!=NULL)

{

int t1=PlacestrToNum(p->building,p->classroom);

if(t1<placei)//该课堂空间hash值较大

{

if(p->room\_next!=NULL)

{

int t2 = PlacestrToNum(p->room\_next->building, p->room\_next->classroom);

if(t2<placei) {p=p->room\_next;continue;}//寻找合适的插入点

else if(t2>placei)

{

p->room\_next->room\_pri=temp; temp->room\_next=p->room\_next;

p->room\_next=temp; temp->room\_pri=p;

temp->lesson\_time=&(les\_time[timei]);

return 1;

}

else

{

//printf("33333\n" );

return -1;

}

}

else//否者插到p的后面

{

temp->lesson\_time=&(les\_time[timei]);

p->room\_next=temp;temp->room\_pri=p;

return 1;

}

}

else if(t1>placei)//插到p的前面

{

temp->lesson\_time=&(les\_time[timei]);

les\_time[timei].time\_next=temp;

temp->room\_next=p; p->room\_pri=temp;

return 1;

}

else//上课冲突

{

// printf("4444\n");

return -1;

}

}

}

else

{

//时间链空间链都不空，其实前两个if不用写，只写下面的理论上也可以

//寻找合适的时间

p=les\_place[placei].room\_next;

while(p!=NULL)

{

int t1=TimestrToNum(p->week , p->day , p->period);

if(t1<timei)

{

if(p->time\_next!=NULL)

{

int t2=TimestrToNum(p->time\_next->week , p->time\_next->day , p->time\_next->period);

if(t2<timei) {p=p->time\_next;continue;}

else if(t2>timei)

{

p->time\_next->time\_pri=temp; temp->time\_next=p->time\_next;

p->time\_next=temp; temp->time\_pri=p;

temp->lesson\_place=&(les\_place[placei]);

break;

}

else

{

// printf("4.5\n");

//printf("timei: %d placei :%d\n",timei,placei);

//printf("t2: %d\n",t2);

return -1;

}

}

else

{

temp->lesson\_place=&(les\_place[placei]);

p->time\_next=temp;temp->time\_pri=p;

break;

}

}

else if(t1>timei)

{

temp->lesson\_place=&(les\_place[placei]);

les\_place[placei].room\_next=temp;

temp->time\_next=p; p->time\_pri=temp;

break;

}

else

{

// printf("5555\n");

// printf("timei:%d placei : %d\n",timei,placei);

return -1;

}

}

//寻找合适的空间

p=les\_time[timei].time\_next;

while(p!=NULL)

{

int t1=PlacestrToNum(p->building,p->classroom);

if(t1<placei)

{

if(p->room\_next!=NULL)

{

int t2 = PlacestrToNum(p->room\_next->building, p->room\_next->classroom);

if(t2<placei) {p=p->room\_next;continue;}

else if(t2>placei)

{

p->room\_next->room\_pri=temp; temp->room\_next=p->room\_next;

p->room\_next=temp; temp->room\_pri=p;

temp->lesson\_time=&(les\_time[timei]);

return 1;

}

else

{

//printf("6666\n");

return -1;

}

}

else

{

temp->lesson\_time=&(les\_time[timei]);

p->room\_next=temp;temp->room\_pri=p;

return 1;

}

}

else if(t1>placei)

{

temp->lesson\_time=&(les\_time[timei]);

les\_time[timei].time\_next=temp;

temp->room\_next=p; p->room\_pri=temp;

return 1;

}

else

{

// printf("77777\n");

return -1;

}

}

}

return 0;

}

/\*\*

\* 函数名称: ClassnoInsert

\* 函数功能: 在专业班级的维度上建立查找链

\* 输入参数: 要插入的课堂结构体指针

\* 返 回 值: 返回1

\*

\*/

int ClassnoInsert(LESSON\* temp)

{

int done=0;

for(int i=0 ; i<les\_classno.size() ; i++)//寻找符合条件的表头

{

if(!strcmp(les\_classno[i].classno , temp->classno))//如果班级名称相等，那么在后面插入

{

done=1;

if(les\_classno[i].lesson==NULL)//如果之后没有结点，那么直接插入

{

les\_classno[i].lesson=temp;

return 1;

}

else//否者寻找合适的插入位置

{

LESSON \*p=les\_classno[i].lesson;

LESSON\* pp=p;

while(p!=NULL)

{

pp=p;

p=p->classno\_next;

}

pp->classno\_next=temp;

temp->classno\_pri=pp;

return 1;//插入完成返回1

}

}

}

if(!done)//如果没有在上面执行插入操作，在班级中增加一个班级，然后再插入

{

CLASSNO t; t.lesson=NULL;

strcpy(t.classno,temp->classno);

t.lesson=temp;

les\_classno.push\_back(t);

}

return 1;

}

/\*\*

\* 函数名称: CourseInsert

\* 函数功能: 在课程的维度上建立查找链

\* 输入参数: 要插入的课堂结构体指针

\* 返 回 值: 返回1

\*

\*/

int CourseInsert(LESSON\* temp)

{

int done=0;

for(int i=0 ; i<les\_course.size() ; i++)//遍历课程查找链，寻找同名课程

{

//printf("i1: %d \n",i);

//printf("2 temp->course: %s\n",temp->course);

if(!strcmp(les\_course[i].course , temp->course))//如果课程查找链中有同名课程

{

//printf("i2: %d \n",i);

//printf("2 temp->course: %s\n",temp->course);

done=1;

if(les\_course[i].lesson==NULL)//如果该课程名后面没有课堂

{

les\_course[i].lesson=temp;

// printf("i3: %d \n",i);

return 1;

}

else//如果后面有课堂

{

// printf("i4: %d \n",i);

LESSON \*p=les\_course[i].lesson;

LESSON\* pp=p;

while(p!=NULL)//寻找合适的插入位置

{

pp=p;

p=p->course\_next;

}

pp->course\_next=temp;

temp->course\_pri=pp;

return 1;

}

}

}

if(!done)//如果前面没有找到同名的课程，那么在课程查找链中新添加一个课程，然后再插入

{

COURSE t; t.lesson=NULL;

strcpy(t.course,temp->course);

t.lesson=temp;

les\_course.push\_back(t);

}

return 1;

}

/\*\*

\* 函数名称: ReadLesson

\* 函数功能: 从数据文件中读取课堂信息

\* 输入参数: 数据文件的文件指针

\* 返 回 值: 成功返回1，遇到上课冲突则出错返回-1

\*

\*/

int ReadLesson(FILE \*fp)

{

int res=1,sum=1;

int from,to;

LESSON \*temp;

temp=(LESSON\*)malloc(sizeof(LESSON));

//int i=0;

// char s[10];

if(fp==NULL)

return -1;

while( fscanf(fp,"%d%d",&from ,&to)!=EOF)//读入周次

{

// printf("T: %d\n", T);

// printf("i : %d\n",++i);

/\*char building[15];

char classroom[6];

char teacher[20];

char classno[80];

char course[45];\*/

//对应的赋值

temp->from=from; temp->to=to;

fscanf(fp,"%d%d",&(temp->day),&(temp->period));

fscanf(fp,"%s",temp->building);

fscanf(fp,"%s",temp->classroom);

// fscanf(fp,"%s",temp->teacher);

fscanf(fp,"%s",temp->classno);

fscanf(fp,"%s",temp->course);

temp->time\_next=temp->room\_next=temp->time\_pri=temp->room\_pri=NULL;

/\*temp->teacher\_next=temp->teacher\_pri=NULL\*/

temp->classno\_next=temp->classno\_pri=temp->course\_next=temp->course\_pri=NULL;

temp->lesson\_time=NULL;

temp->lesson\_place=NULL;

// printf("EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE\n");

int i;

for( i=from;i<=to;i++)//从开始周次，到结束周次插入课堂

{

LESSON\* atemp=(LESSON\*)malloc(sizeof(LESSON));

\*atemp=\*temp;

atemp->week=i;

int timei=TimestrToNum(atemp->week,atemp->day,atemp->period);//取得对应时间hash值

int placei=PlacestrToNum(atemp->building,atemp->classroom);//取得空间hash值

if(timei==-1 || placei==-1)//如果没有找到对应的hash值，返回-1

return -1;

res = LessonInsert( placei, timei,atemp);//插入对应的课堂

if(res==-1)

{

// printf("asdfasdfasdfasdf\n");

return -1;

}

else

{

// printf("222222\n");

atemp->No=sum++;

lesson\_sum=atemp->No;

//TeacherInsert(atemp);

if(i==from)

{

// printf("timei :%d placei: %d \n",timei,placei);

ClassnoInsert(atemp);

//printf("33333333\n");

CourseInsert(atemp);

//printf("4444444444\n");

}

}

}

// printf("2 T: %d\n", T);

}

free(temp);

return res;

}

/\*\*

\* 函数名称: Init

\* 函数功能: 初始化表头

\* 输入参数: 无

\* 返 回 值: 1

\*

\*/

int Init(void)

{

//时间表头赋值

for(int i=1;i<=20;i++)

{

for(int j=1;j<=7;j++)

{

for(int k=1;k<=6;k++)

{

//les\_time[7\*6\*(i-1)+6\*(j-1)+k].lesson\_num=0;

les\_time[7\*6\*(i-1)+6\*(j-1)+k].week=i;

les\_time[7\*6\*(i-1)+6\*(j-1)+k].day=j;

les\_time[7\*6\*(i-1)+6\*(j-1)+k].period=k;

les\_time[7\*6\*(i-1)+6\*(j-1)+k].time\_next=NULL;

}

}

}

//西十二楼的表头赋值

for(int i=1;i<=5;i++)

for(int j=1;j<=12;j++)

{

int n=100\*i+j;

char sroom[10];

sprintf(sroom,"%d",n);

for(int k=3;k>=0;k--)

{

sroom[k+1]=sroom[k];

}

sroom[0]='N';//北楼对应的教室编号

sroom[4]='\0';

strcpy(les\_place[12\*(i-1)+j].building, "西十二");

strcpy(les\_place[12\*(i-1)+j].classroom,sroom);

les\_place[12\*(i-1)+j].room\_next=NULL;

sroom[0]='S';//南楼对应编号

strcpy(les\_place[12\*(i-1)+j+60].classroom,sroom);

strcpy(les\_place[12\*(i-1)+j+60].building,"西十二");

les\_place[12\*(i-1)+j+60].room\_next=NULL;

}

for(int i=1;i<=10;i++)//南一楼的表头赋值

{

int n=800; char sroom[10];

n+=i;

sprintf(sroom,"%d",n);

strcpy(les\_place[120+i].building, "南一楼");

strcpy(les\_place[120+i].classroom,sroom);

les\_place[120+i].room\_next=NULL;

}

strcpy(les\_place[121].classroom,"116");//没有801，所以替换成806

//西五楼的表头初始化赋值

for(int i=1;i<=5;i++)

for(int j=1;j<=20;j++)

{

int n=100\*i+j;

char sroom[10];

sprintf(sroom,"%d",n);

int placei=PlacestrToNum("西五楼",sroom);

strcpy(les\_place[placei].building, "西五楼");

strcpy(les\_place[placei].classroom,sroom);

les\_place[placei].room\_next=NULL;

}

//其他上课表头赋值

strcpy(les\_place[240].building, "科技楼");

strcpy(les\_place[240].classroom,"北楼");

les\_place[240].room\_next=NULL;

strcpy(les\_place[241].building, "西操场");

strcpy(les\_place[241].classroom,"##");

les\_place[241].room\_next=NULL;

strcpy(les\_place[242].building, "中心操场");

strcpy(les\_place[242].classroom,"##");

les\_place[242].room\_next=NULL;

strcpy(les\_place[243].building, "东操场");

strcpy(les\_place[243].classroom,"##");

les\_place[243].room\_next=NULL;

return 1;

}

/\*\*

\* 函数名称: SearchByTimeAndPlace

\* 函数功能: 根据时间和空间查找课堂

\* 输入参数: 时间空间的hash值

\* 返 回 值: 找到返回1，没找返回-1

\*

\*/

int SearchByTimeAndPlace(int timei,int placei)

{

LESSON\* p=les\_place[placei].room\_next;

//printf("classroom : %s\n",les\_place[placei].classroom);

while(p!=NULL)//在要查找的空间下寻找对应的时间的结点

{

if((p->lesson\_time) == &(les\_time[timei]))//如果查找到了，那么直接退出循环

break;

p=p->time\_next;//没找到找下一个结点

}

if(p==NULL)//如果彻底没找到

return -1;

else//找到,输出对应结点的信息

{

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

printf(" 课程：%s\n",p->course);

printf(" 周次：%d~%d周\n",p->from,p->to);

printf(" 星期：%d\n",p->day);

printf(" 节数：%d\n",p->period);

printf(" 教学楼：%s\n",p->building);

printf(" 教室：%s\n",p->classroom);

printf(" 班级：%s\n",p->classno);

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

return 1;

}

}

/\*\*

\* 函数名称: SearchByCourse

\* 函数功能: 在课程查找链上查找课堂

\* 输入参数: 课程名称

\* 返 回 值: 找到返回1，没找返回-1

\*

\*/

int SearchByCourse(char s[])//根据课程信息查找课堂

{

int no=0;

for(int i=0 ; i<(int)les\_course.size() ; i++)//遍历课程查找链

{

if(strstr(les\_course[i].course ,s))//如果用户输入的课堂名是查找链中某个表头的子串

{

if(les\_course[i].lesson==NULL)

continue;

else

{

LESSON \*p=les\_course[i].lesson;

while(p!=NULL)//输出对应结点的信息

{

if(readed[p->No])

continue;

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

printf("No.%d\n",++no);

printf(" 课程：%s\n",p->course);

printf(" 周次：%d~%d周\n",p->from,p->to);

printf(" 星期：%d\n",p->day);

printf(" 节数：%d\n",p->period);

printf(" 教学楼：%s\n",p->building);

printf(" 教室：%s\n",p->classroom);

printf(" 班级：%s\n",p->classno);

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

readed[p->No]=true;

p=p->course\_next;

}

}

}

}

if(no==0)

return -1;

else

return 1;

}

/\*\*

\* 函数名称: SearchByClassno

\* 函数功能: 在专业班号查找链上查找课堂信息

\* 输入参数: 专业班号

\* 返 回 值: 找到返回1，没找返回-1

\*

\*/

int SearchByClassno(char s[])

{

int no=0;

for(int i=0 ; i<les\_classno.size() ; i++)//遍历班级查找链

{

if(strstr(les\_classno[i].classno ,s))//如果用户输入的班级是查找链中某个表头的子串

{

if(les\_classno[i].lesson==NULL)

continue;

else//如果找到了对应的班级

{

LESSON \*p=les\_classno[i].lesson;

while(p!=NULL)//查找对应的结点，然后输出

{

if(readed[p->No])

continue;

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

printf("No.%d\n",++no);

printf(" 班级：%s\n",p->classno);

printf(" 周次：%d~%d周\n",p->from,p->to);

printf(" 星期：%d\n",p->day);

printf(" 节数：%d\n",p->period);

printf(" 教学楼：%s\n",p->building);

printf(" 教室：%s\n",p->classroom);

printf(" 课程：%s\n",p->course);

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

readed[p->No]=true;

p=p->classno\_next;

}

}

}

}

if(no==0)

return -1;

else

return 1;

}

/\*\*

\* 函数名称: ReadGeneralInfo

\* 函数功能: 从数据文件中读取综合数据信息

\* 输入参数: 综合数据信息文件

\* 返 回 值: 成功返回1，没有读到任何信息返回-1

\*

\*/

int ReadGeneralInfo(FILE \* fp)

{

// int i=0;

INFO temp;int done=0;

while(fscanf(fp,"%s",temp.course)!=EOF)//循环读入信息

{

// printf("i: %d\n",++i);

done=1;

fscanf(fp,"%s",temp.time);

fscanf(fp,"%s",temp.credit);

fscanf(fp,"%s",temp.title);

fscanf(fp,"%s",temp.teacher);

fscanf(fp,"%s",temp.classno);

fscanf(fp,"%s",temp.extra);

gen\_info.push\_back(temp);

}

if(!done)//如果一个都没有读到

return -1;

else

return 1;

}

/\*\*

\* 函数名称: ToSaveData

\* 函数功能: 将课堂信息和综合信息保存在对应的文件中

\* 输入参数: 两个文件对应的文件指针

\* 返 回 值: 返回1

\*

\*/

int ToSaveData(FILE\* fp1,FILE\* fp2)

{

/\* for(int i=1;i<=260;i++)

{

LESSON \*p = les\_place[i].room\_next;

// printf("p%d: %p\n",i,p);

while(p!=NULL)

{

fprintf(fp1,"%d %d\n",p->week,p->week);

fprintf(fp1,"%d\n",p->day);

fprintf(fp1,"%d\n",p->period);

fprintf(fp1,"%s\n",p->building);

fprintf(fp1,"%s\n",p->classroom);

fprintf(fp1,"%s\n",p->classno);

fprintf(fp1,"%s\n",p->course);

// fprintf(fp1,"%d\n",p->No);

fprintf(fp1,"\n\n\n");

p=p->time\_next;

}

}\*/

for(int i=0;i<les\_course.size();i++)//按照课程查找链中的信息

{

LESSON \*p = les\_course[i].lesson;

while(p)//把所有课程查找链中的所有的信息输出

{

fprintf(fp1,"%d %d\n",p->from,p->to);

fprintf(fp1,"%d\n",p->day);

fprintf(fp1,"%d\n",p->period);

fprintf(fp1,"%s\n",p->building);

fprintf(fp1,"%s\n",p->classroom);

fprintf(fp1,"%s\n",p->classno);

fprintf(fp1,"%s\n",p->course);

// fprintf(fp1,"%d\n",p->No);

fprintf(fp1,"\n\n\n");

p=p->course\_next;

}

}

for(int i=0;i<gen\_info.size();i++)//把综合课程信息中的所有的信息输出

{

fprintf(fp2,"%s\n",gen\_info[i].course);

fprintf(fp2,"%s\n",gen\_info[i].time);

fprintf(fp2,"%s\n",gen\_info[i].credit);

fprintf(fp2,"%s\n",gen\_info[i].title);

fprintf(fp2,"%s\n",gen\_info[i].teacher);

fprintf(fp2,"%s\n",gen\_info[i].classno);

fprintf(fp2,"%s\n",gen\_info[i].extra);

fprintf(fp2,"\n\n\n");

}

fclose(fp1);

fclose(fp2);

return 1;

}

/\*\*

\* 函数名称: SpaceToUnderline

\* 函数功能: 把字符串中的空格变成下划线

\* 输入参数: 要转化的字符串首地址

\* 返 回 值: 无

\*

\*/

void SpaceToUnderline(char\*s) //C语言中create函数能创建的文件的文件名只能是标识符，因此把系统时间字符串中的冒号的空格变成下划线

{

int i=0;

for(i=0;(s)[i]!='\0';i++)

{

if( (s[i]==' ' ) || (s[i]==':' ) )

s[i]='\_';

else

{

continue;

}

}

return;

}

/\*\*

\* 函数名称: AddBackslash

\* 函数功能: 把字符串中的单斜线变成双斜线

\* 输入参数: 要转化的字符串首地址

\* 返 回 值: 无

\*

\*/

void AddBackslash(char\*s) //在一个不可能连续出现2个或两个以上 \ 字符的字符串中，把出现的 \ 字符变成 "\\"

{

int i=0,j,len;

for(i=0;s[i]!='\0';i++)

{

if(s[i]=='\\') // 如果s[i]是\ 。

{

len=strlen(s);

for(j=len;j>i;j--)//for循环是把s[i]后面的所有字符向后平移一下，为了在s[i]后面加\腾出空间

{

s[j+1]=s[j];

}

s[i+1]='\\';

i++; //注意这时候s[i]之后的那个字符也是\了，因此i要自增1来跳过后加的\ 在其后再搜索\。

}

else

continue;

}

return;

}

/\*\*

\* 函数名称: GetBackupFilename

\* 函数功能: 得到包含路径的备份文件名

\* 输入参数: 储存备份文件名的字符串首地址

\* 返 回 值: 无

\*

\*/

void GetBackupFilename(char \*dir1, char\* dir2 )//得到包含路径的备份文件名

{

int len;//字符串长度

time\_t t;//定义一个时间变量

t=time(NULL);

char \*time;//定义一个字符串用来保存获取到的日期和时间

time=ctime(&t);//赋值

SpaceToUnderline(time);//C语言中create函数能创建的文件的文件名只能是标识符，因此把系统时间字符串中的冒号的空格变成下划线

/\*把时间字符串后面加上后缀名构成文件名，注意不能用strcat函数，因为时间字符串time在'\0'字符之前还有个看不见的换行符，直接接上会把".dat"接到换行符后面\*/

len=strlen(time);

time[len-1]='.';

time[len]='t';

time[len+1]='x';

time[len+2]='t';

time[len+3]='\0'; //这是文件名构造完成了，这时time里面的字符串是带后缀名的文件名了，不仅仅是时间字符串

char BackupDir1[25]="backup\\lesson\_";

char BackupDir2[25]="backup\\gen\_info\_";

//getcwd(dir1,100);//获取当前工作地址,重点是！！！看下一行：

/\*char \*getcwd(char \*buf, size\_t size); 这是获取当前地址函数的原型，正常第二个参数应该是sizeof(dir),但是注意数组名是个指针常量并不完全同于指针变量，比如char s[10],\*p=s \*\*pp=&p; ,sizeof(s)等于10

而sizeof(p)和sizeof(pp)却等于4，因为一个指针变量占四个字节是固定的，因此char指针作为函数参数时候传递不了字符串所占的字节数。因此第二个参数只能是100（因为在BackupData函数中

声明的用于储存含地址的备份文件名的字符数组 为char filename[100] sizeof(filename)等于100\*）而getcwd(dir,sizeof(dir))是错误的\*/

//getcwd(dir2,100);

strcpy(dir1,BackupDir1);//在当前工作地址之后加上\backup\构造备份文件夹所在目录

strcpy(dir2,BackupDir2);

strcat(dir1,time);//在目录后面加上文件名

strcat(dir2,time);

AddBackslash(dir1);//把整个路径字符串里面所有的单个 \ 变成\\ 以便当做文件名字符串在create等函数中做实参时 \\ 转意成 “\”

AddBackslash(dir2);

// printf("%s\n",dir1 );

// printf("%s\n",dir2 );

return ;

}

/\*\*

\* 函数名称: ToBackup

\* 函数功能: 备份数据

\* 输入参数: 备份文件的名称

\* 返 回 值: 1

\*

\*/

int ToBackup(char\* filename1,char\* filename2)//向两个文件中存入备份信息

{

FILE\* fp1=fopen(filename1,"w+");

FILE\* fp2=fopen(filename2,"w+");

if(fp1==NULL || fp2==NULL)

return -1;

// printf("fp1:%p fp2:%p\n",fp1,fp2 );

ToSaveData(fp1,fp2);//向对应文件指针所指的文件中存入对应的备份信息

return 1;

}

/\*\*

\* 函数名称: ClearScreenTotalInputArea

\* 函数功能: 清屏

\* 输入参数: 无

\* 返 回 值: 无

\*

\*/

void ClearScreenTotalInputArea(void)

{

CONSOLE\_SCREEN\_BUFFER\_INFO bInfo;

COORD home = {0,1};

unsigned long size , ul;

GetConsoleScreenBufferInfo(gh\_std\_out,&bInfo); //取屏幕缓冲区信息

size = bInfo.dwSize.X \* (bInfo.dwSize.Y-1); //计算屏幕缓冲区除了第一行字符单元数

/\*将屏幕缓冲区除了第一行所有单元的字符属性设置为当前屏幕缓冲区字符属性\*/

FillConsoleOutputAttribute(gh\_std\_out, bInfo.wAttributes, size, home, &ul);

/\*将屏幕缓冲区除了第一行所有单元填充为空格符\*/

FillConsoleOutputCharacter(gh\_std\_out, ' ', size, home, &ul);

SetConsoleCursorPosition(gh\_std\_out,home);

return;

}

/\*\*

\* 函数名称: DestroyLesson

\* 函数功能: 销毁内存中的课堂数据

\* 输入参数: 无

\* 返 回 值: 1

\*

\*/

int DestroyLesson(void)

{

//清除空间同时制空空间表头指针

for(int i=0;i<260;i++)

{

LESSON\* p=les\_place[i].room\_next;

LESSON\* pp=p;

while(p!=NULL)

{

pp=p->time\_next;

free(p);

p=pp;

}

les\_place[i].room\_next=NULL;

}

//置空时间表头指针

for(int i=0;i<845;i++)

{

les\_time[i].time\_next=NULL;

}

lesson\_sum=0;

// printf("1111111111111\n");

return 1;

}

/\*\*

\* 函数名称: DestroyOtherVectorInfo

\* 函数功能: 销毁综合信息数据以及查找链中的数据

\* 输入参数: 无

\* 返 回 值: 1

\*

\*/

int DestroyOtherVectorInfo(void)

{

gen\_info.clear();

les\_classno.clear();

les\_course.clear();

//printf("2222222222\n");

return 1;

}

/\*\*

\* 函数名称: DeleteLesson

\* 函数功能: 删除对应时间空间的课程

\* 输入参数: 要删除课堂的时间和空间的hash值

\* 返 回 值: 成功返回1，出错返回-1

\*

\*/

int DeleteLesson(int timei, int placei)

{

LESSON\* p=les\_place[placei].room\_next;

while(p!=NULL)//在placei链下寻找时间正确的结点

{

if((p->lesson\_time) == &(les\_time[timei]))

break;

p=p->time\_next;

}

if(p==NULL)//如果没找到

{

// printf("111111\n");

return -1;

}

//否者找到

/\*空间轴上的删除\*/

if(p->time\_pri==NULL)

{

les\_place[placei].room\_next=p->time\_next;

if(p->time\_next!=NULL)

p->time\_next->time\_pri=NULL;

// printf("111\n");

}

else

{

p->time\_pri->time\_next=p->time\_next;

if(p->time\_next != NULL)

p->time\_next->time\_pri=p->time\_pri;

// printf("222\n");

}

/\*时间轴上的删除\*/

if(p->room\_pri==NULL)

{

les\_time[timei].time\_next=p->room\_next;

if(p->room\_next!=NULL)

p->room\_next->room\_pri=NULL;

// printf("333\n");

}

else

{

p->room\_pri->room\_next=p->room\_next;

if(p->room\_next != NULL)

p->room\_next->room\_pri=p->room\_pri;

// printf("444\n");

}

/\*班级轴上的删除\*/

int classnoi=-1;

for(int i=0;i<les\_classno.size();i++)//查找les\_classno中对应的下标

{

if(!strcmp(p->classno,les\_classno[i].classno))

{

classnoi=i; break;

}

// printf("les\_classno i: %d\n",i);

}

if(classnoi==-1)//没找到说明赋值的时候出了问题

{

//printf("2222222\n");

return -1;

}

if(p->classno\_pri==NULL)

{

les\_classno[classnoi].lesson=p->classno\_next;

if(p->classno\_next!=NULL)

p->classno\_next->classno\_pri=NULL;

}

else

{

p->classno\_pri->classno\_next=p->classno\_next;

if(p->classno\_next != NULL)

p->classno\_next->classno\_pri=p->classno\_pri;

}

/\*课程轴上的删除\*/

int coursei=-1;

for(int i=0;i<les\_course.size();i++)//查找les\_classno中对应的下标

{

if(!strcmp(p->course,les\_course[i].course))

{

coursei=i; break;

}

// printf("les\_course i: %d\n",i);

}

if(coursei==-1)//没找到说明赋值的时候出了问题

{

//printf("3333333\n");

return -1;

}

if(p->course\_pri==NULL)

{

les\_course[coursei].lesson=p->course\_next;

if(p->course\_next!=NULL)

p->course\_next->course\_pri=NULL;

}

else

{

p->course\_pri->course\_next=p->course\_next;

if(p->course\_next != NULL)

p->course\_next->course\_pri=p->course\_pri;

}

free(p);

return 1;

}

/\*\*

\* 函数名称: ToInsertLesson

\* 函数功能: 插入对应时间空间的课程

\* 输入参数: 无

\* 返 回 值: 成功返回1，出错返回-1

\*

\*/

int ToInsertLesson(void)

{

LESSON\* temp=(LESSON\*)malloc(sizeof(LESSON));

int timei,placei;

/\*临时结点的初始化\*/

temp->time\_next=temp->room\_next=temp->time\_pri=temp->room\_pri=NULL;

/\*temp->teacher\_next=temp->teacher\_pri=NULL\*/

temp->classno\_next=temp->classno\_pri=temp->course\_next=temp->course\_pri=NULL;

temp->lesson\_time=NULL;

temp->lesson\_place=NULL;

Flag1://输出提示信息

printf("说明：\n");

printf(" 1，一个学期1~20周，一周7天(输入1~7),每天6大节课\n");

printf(" 2，教学楼包括西十二、西五楼、南一楼、科技楼、西操场\n");

printf(" 3，东操场、中心操场。其中科技楼的教室只有“北楼”、操场的教室为“##”\n\n");

printf("请输入要插入的课堂所在的时间、地点(用空格分开)：\n");

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

printf("周次 星期 节数(1~6) 教学楼 教室\n");

scanf("%d%d%d",&(temp->week),&(temp->day),&(temp->period) );

scanf("%s%s",temp->building,temp->classroom );

getchar();

temp->from=temp->to=temp->week;

timei=TimestrToNum(temp->week,temp->day,temp->period);//取得时间hash值

placei=PlacestrToNum(temp->building,temp->classroom);//取得空间hash值

if(timei==-1 || placei==-1)//格式输入错误

{

int result = MessageBox(NULL,"您输入的信息格式或者范围错误！","ERROR",MB\_ICONINFORMATION|MB\_RETRYCANCEL);

if(result==IDRETRY)

{

ClearScreenTotalInputArea();

goto Flag1;

}

else

{

ClearScreenTotalInputArea();

return -1;

}

}

else//格式正确

{

ClearScreenTotalInputArea();

printf("请输入下面的课堂信息: \n");

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

printf("专业班号 课程名称 \n");

scanf("%s%s",temp->classno,temp->course);

getchar();

if(LessonInsert(placei,timei,temp)==-1)//插入失败，发生上课冲突

{

MessageBox(NULL,"课堂插入失败，可能发生了上课冲突！","ERROR!",MB\_ICONINFORMATION|MB\_OK);

ClearScreenTotalInputArea();

return -1;

}

else//在课程和班级查找链上查找对应的结点

{

ClassnoInsert(temp);

//printf("33333333\n");

CourseInsert(temp);

//printf("4444444444\n");

temp->No = ++lesson\_sum;

MessageBox(NULL,"课堂插入完成！","SUCCESS!",MB\_OK);

ClearScreenTotalInputArea();

return 1;

}

}

}

/\*\*

\* 函数名称: ToDeleteLesson

\* 函数功能: 删除对应时间空间的课程

\* 输入参数: 无

\* 返 回 值: 成功返回1，出错返回-1

\*

\*/

int ToDeleteLesson(void)

{

int timei,placei;

int week,day,period;

char s1[100],s2[100];

Flag1:

printf("说明：\n");

printf(" 1，一个学期1~20周，一周7天(输入1~7),每天6大节课\n");

printf(" 2，教学楼包括西十二、西五楼、南一楼、科技楼、西操场\n");

printf(" 3，东操场、中心操场。其中科技楼的教室只有“北楼”、操场的教室为“##”\n\n");

printf("请输入要插入的课堂所在的时间、地点(用空格分开)：\n");

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

printf("周次 星期 节数(1~6) 教学楼 教室\n");

scanf("%d%d%d",&week,&day,&period);

scanf("%s%s",s1,s2);

getchar();

timei=TimestrToNum(week,day,period);//获得时间hash值

placei=PlacestrToNum(s1,s2);//获得空间hash值

if(timei==-1 || placei==-1)//格式输入错误

{

int result = MessageBox(NULL,"您输入的信息格式或者范围错误！","ERROR",MB\_ICONINFORMATION|MB\_RETRYCANCEL);

if(result==IDRETRY)

{

ClearScreenTotalInputArea();

goto Flag1;

}

else

{

ClearScreenTotalInputArea();

return -1;

}

}

else//格式正确

{

int res = DeleteLesson(timei,placei);//删除

if(res==1)//删除执行成功

{

MessageBox(NULL,"课堂删除完成！","SUCCESS!",MB\_OK);

ClearScreenTotalInputArea();

return 1;

}

else//删除失败

{

MessageBox(NULL,"课堂删除失败，可能此课堂不存在！","ERROR!",MB\_ICONINFORMATION|MB\_OK);

ClearScreenTotalInputArea();

return -1;

}

}

}

/\*\*

\* 函数名称: ReloadData

\* 函数功能: 重载内存中的数据

\* 输入参数: 无

\* 返 回 值: 成功返回1，出错返回-1

\*

\*/

int ReloadData(void)

{

FILE\* fp1=fopen("data\\lesson\_to\_read.txt","r+");

// printf("1111\n");

FILE\* fp2=fopen("data\\gen\_info.txt","r+");

DestroyLesson();//销毁内存中的课堂数据

DestroyOtherVectorInfo();//销毁内存中的综合课程信息

// printf("22222\n");

// printf("fp: %p \n",fp);

int res1=ReadLesson(fp1);//再次从文件读入课堂信息

//printf("res : %d \n",res);

int res2=ReadGeneralInfo(fp2);//再次从文件读入综合课程信息

fclose(fp1);

fclose(fp2);

if((res1==-1)||(res2==-1))//失败

{

return -1;

}

else//成功

{

return 1;

}

}

/\*\*

\* 函数名称: ToModifyLesson

\* 函数功能: 修改课堂信息

\* 输入参数: 无

\* 返 回 值: 成功返回1，出错返回-1

\*

\*/

int ToModifyLesson(void)

{

LessonFileForm();//打印数据配置文件的格式信息

system(".\\data\\lesson\_to\_read.txt");//打开对应的数据配置文件

//system(".\\data\\lesson\_to\_save.txt");

int res=ReloadData();//用户修改完之后重载数据

if(res==-1)//数据文件中的课堂排课冲突

{

MessageBox(NULL,"课堂修改错误！可能发生了上课冲突！","ERROR!",MB\_ICONINFORMATION|MB\_OK);

return -1;

}

else//修改成功

{

MessageBox(NULL,"课堂修改成功！","SUCCESS!",MB\_OK);

return 1;

}

}

/\*\*

\* 函数名称: ToInsertCourse

\* 函数功能: 插入课程

\* 输入参数: 无

\* 返 回 值: 成功返回1

\*

\*/

int ToInsertCourse(void)

{

INFO temp;

printf("请按要求输入以下信息，其中缺省的数据项用“##”表示：\n");

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

printf("课程 学时 学分 任课教师 教师职称 上课班级 备注\n");

scanf("%s%s%s%s%s%s%s",temp.course,temp.time,temp.credit,temp.teacher,temp.title,temp.classno,temp.extra );

getchar();

gen\_info.push\_back(temp);//插入信息

MessageBox(NULL,"课程插入成功！","SUCCESS!",MB\_OK);

return 1;

}

/\*\*

\* 函数名称: ToFindGenInfoByCourse

\* 函数功能: 通过课程名查询综合信息

\* 输入参数: 课程名

\* 返 回 值: 成功返回1，出错返回-1

\*

\*/

int ToFindGenInfoByCourse(char s[])

{

for(int i=0;i<gen\_info.size();i++)//根据课程名查找想要查找的数据结点的下标，存在find中

{

if(strstr(gen\_info[i].course,s))

{

find.push\_back(i);

}

}

if(find.size()==0)

return -1;

for(int i=0;i<find.size();i++)//把找到的都打印出来

{

INFO t=gen\_info[find[i]];

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

printf("No.%d\n",i+1);

printf(" 课程：%s\n",t.course);

printf(" 学时：%s\n",t.time);

printf(" 学分：%s\n",t.credit);

printf(" 任课教师：%s\n",t.teacher);

printf(" 教师职称：%s\n",t.title);

printf(" 上课班级：%s\n",t.classno);

printf(" 备注：%s\n",t.extra);

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

}

return 1;

}

/\*\*

\* 函数名称: ToDeleteCourse

\* 函数功能: 通过课程名删除综合信息

\* 输入参数: 课程名

\* 返 回 值: 成功返回1，出错返回-1

\*

\*/

int ToDeleteCourse(void)

{

char s[100];

printf("\n\n请输入要删除的课程名:\n ");

scanf("%s",s);

getchar();

int res=ToFindGenInfoByCourse(s);

if(res!=-1)//有这门课

{

Flag1:

printf("\n请输入要删除的课程编号(仅一个):");

int no; scanf("%d",&no);getchar();//读入要删除的课堂的编号

if(no<1 || no>(find.size()))//编号输入的格式错误

{

MessageBox(NULL,"编号输入错误!请重新输入！","ERROR!",MB\_ICONINFORMATION|MB\_OK);

fflush(stdin);

goto Flag1;

}

else//格式输入正确，执行删除

{

int j=find[no-1];

std::vector<INFO>::iterator it=gen\_info.begin()+j;

gen\_info.erase(it);

MessageBox(NULL,"课程删除成功！","SUCCESS!",MB\_OK);

find.clear();

return 1;

}

}

else//根本没有这门课

{

MessageBox(NULL,"此课程不存在！删除失败","FAIL!",MB\_ICONINFORMATION|MB\_OK);

find.clear();

return -1;

}

}

/\*\*

\* 函数名称: ToModifyCourse

\* 函数功能: 通过课程名修改综合信息

\* 输入参数: 无

\* 返 回 值: 成功返回1，出错返回-1

\*

\*/

int ToModifyCourse(void)

{

char s[100];

printf("\n\n请输入要修改的课程名:\n ");

scanf("%s",s);

getchar();

int res=ToFindGenInfoByCourse(s);

if(res!=-1)//有这门课

{

Flag1:

printf("\n请输入要修改的课程编号(仅一个):");

int no; scanf("%d",&no);getchar();//读入用户输入的编号

if(no<1 || no>(find.size()))//编号格式输入错误

{

MessageBox(NULL,"编号输入错误！请重新输入！","ERROR!",MB\_ICONINFORMATION|MB\_OK);

fflush(stdin);

goto Flag1;

}

else//执行删除功能

{

int j=find[no-1];

printf("请按要求输入新的信息，其中缺省的数据项用“##”表示：\n");

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

printf("课程 学时 学分 任课教师 教师职称 上课班级 备注\n");

scanf("%s%s%s%s%s%s%s",gen\_info[j].course,gen\_info[j].time,gen\_info[j].credit,gen\_info[j].teacher,gen\_info[j].title,gen\_info[j].classno,gen\_info[j].extra );

getchar();

MessageBox(NULL,"课程信息修改成功！","SUCCESS!",MB\_OK);

find.clear();

return 1;

}

}

else//根本没有这门课

{

MessageBox(NULL,"此课程不存在！修改失败","FAIL!",MB\_ICONINFORMATION|MB\_OK);

find.clear();

return -1;

}

}

/\*\*

\* 函数名称: ToInsertTeacher

\* 函数功能: 通过教师名插入综合信息

\* 输入参数: 无

\* 返 回 值: 1

\*

\*/

int ToInsertTeacher(void)

{

INFO temp;

printf("请按要求输入以下信息，其中缺省的数据项用“##”表示：\n");

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

printf("教师 职称 担任课程 此课程学时 此课程学分 上课班级 备注\n");

scanf("%s%s%s%s%s%s%s",temp.teacher,temp.title,temp.course,temp.time,temp.credit,temp.classno,temp.extra );//读入用户输入的信息

getchar();

gen\_info.push\_back(temp);//执行插入

MessageBox(NULL,"教师插入成功！","SUCCESS!",MB\_OK);

return 1;

}

/\*\*

\* 函数名称: ToInsertTeacher

\* 函数功能: 通过教师名插入综合信息

\* 输入参数: 无

\* 返 回 值: 1

\*

\*/

int ToFindGenInfoByTeacher(char s[])

{

for(int i=0;i<gen\_info.size();i++)//寻找对应老师的课程综合信息

{

if(strstr(gen\_info[i].teacher,s))//如果符合条件，那么加入find中

{

find.push\_back(i);

}

}

if(find.size()==0)//如果一个也没找到

return -1;

for(int i=0;i<find.size();i++)//输出所有的找到的信息以及 编号

{

INFO t=gen\_info[find[i]];

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

printf("No.%d\n",i+1);

printf(" 教师姓名：%s\n",t.teacher);

printf(" 职称： %s\n",t.title);

printf(" 担任课程：%s\n",t.course);

printf(" 上课班级：%s\n",t.classno);

printf(" 课程学时：%s\n",t.time);

printf(" 课程学分：%s\n",t.credit);

printf(" 备注：%s\n",t.extra);

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

}

return 1;

}

/\*\*

\* 函数名称: ToDeleteTeacher

\* 函数功能: 通过教师名删除综合信息

\* 输入参数: 无

\* 返 回 值: 1

\*

\*/

int ToDeleteTeacher(void)

{

char s[100];

printf("\n\n请输入要删除的教师姓名:\n ");

scanf("%s",s);

getchar();

int res=ToFindGenInfoByTeacher(s);//执行查找

if(res!=-1)//有这个老师

{

int res= MessageBox(NULL,"是否删除该教师与所有与该教师相关的信息？","？",MB\_ICONQUESTION|MB\_YESNO);

if(res==IDYES)//选yes

{

for(int i=0;i<find.size();i++)

{

int j=find[i]-i;/\*\*重点！因为find是从前往后遍历的，因为每删除一个，后边待删除数据元素的角标(无需删除的元素也是)集体向前移动了1,总的就是前移i\*/

std::vector<INFO>::iterator it=gen\_info.begin()+j;

gen\_info.erase(it);

}

find.clear();

MessageBox(NULL,"教师删除成功！","SUCCESS!",MB\_OK);

return 1;

}

else//选NO

{

find.clear();

return 0;

}

}

else//根本没有这个老师

{

MessageBox(NULL,"此教师不存在！删除失败","FAIL!",MB\_ICONEXCLAMATION|MB\_OK);

find.clear();

return -1;

}

}

/\*\*

\* 函数名称: ToModifyTeacher

\* 函数功能: 通过教师名修改综合信息

\* 输入参数: 无

\* 返 回 值: 1

\*

\*/

int ToModifyTeacher(void)

{

char s[100];

printf("\n\n请输入要修改信息的教师姓名:\n ");

scanf("%s",s);

getchar();

int res=ToFindGenInfoByTeacher(s);//执行查找

if(res!=-1)//有这门课

{

Flag1:

printf("\n请输入要修改信息的数据元素编号(仅一个):");

int no; scanf("%d",&no);getchar();

if(no<1 || no>(find.size()))//编号格式错误

{

MessageBox(NULL,"编号输入错误！请重新输入！","ERROR!",MB\_ICONINFORMATION|MB\_OK);

fflush(stdin);

goto Flag1;

}

else//格式正确

{

int j=find[no-1];

printf("请按要求输入新的教师信息，其中缺省的数据项用“##”表示：\n");

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

printf("教师 职称 担任课程 此课程学时 此课程学分 上课班级 备注\n");

scanf("%s%s%s%s%s%s%s",gen\_info[j].teacher,gen\_info[j].title,gen\_info[j].course,gen\_info[j].time,gen\_info[j].credit,gen\_info[j].classno,gen\_info[j].extra );

getchar();

MessageBox(NULL,"教师信息修改成功！","SUCCESS!",MB\_OK);

find.clear();

return 1;

}

}

else//没有找到这门课

{

MessageBox(NULL,"此教师不存在！修改失败","FAIL!",MB\_ICONEXCLAMATION|MB\_OK);

find.clear();

return -1;

}

}

/\*\*

\* 函数名称: LessonFileForm

\* 函数功能: 打印文件格式信息

\* 输入参数: 无

\* 返 回 值: 无

\*

\*/

void LessonFileForm(void)

{

printf(" 数据文件格式\n");

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

printf("\n课堂开始周数 课堂结束周数\n\n");

printf("星期 (1~7表示周一到周日)\n\n");

printf("节数 (1~6节)\n\n");

printf("教学楼 (西十二、南一楼、西五楼、科技楼、东操场、西操场、中心操场)\n\n");

printf("教室编号 (其中科技楼的教室只有“北楼”、操场没有教室用“##”表示)\n\n");

printf("上课班级专业班号 (计算机校交,计算机卓工,计算机1601等,一起上课的班级用“,”隔开)\n\n");

printf("课程名称\n\n" );

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

printf("警告！！！格式错误会导致数据载入错误！！一切缺省项用“##”表示!\n\n");

}

/\*\*

\* 函数名称: ToQueryCourseByTeacher

\* 函数功能: 通过教师查询综合信息

\* 输入参数: 无

\* 返 回 值: 成功返回1，出错返回-1

\*

\*/

int ToQueryCourseByTeacher(void)

{

char s[100];

printf("请输入要查询的老师的信息：\n");

scanf("%s",s);

fflush(stdin);

int res = ToFindGenInfoByTeacher(s);//查找的对应的老师的结点的信息

if(res==-1)//没找到

{

MessageBox(NULL,"此教师不存在！查询失败！","FAIL!",MB\_ICONEXCLAMATION|MB\_OK);

find.clear();

return -1;

}

else//找到

{

find.clear();

printf("\n按任意键继续...");

getch();

return 1;

}

}

/\*\*

\* 函数名称: ToQueryLessonByClassno

\* 函数功能: 通过班级查询综合信息

\* 输入参数: 无

\* 返 回 值: 成功返回1，出错返回-1

\*

\*/

int ToQueryLessonByClassno(void)

{

char s[100];

printf("请输入要查询的班级的专业班号：\n ");

scanf("%s",s);

fflush(stdin);

int res=SearchByClassno(s);//查找

if(res==1)//找到

{

printf("\n信息输出完成，按任意键继续...");

getch();

return 1;

}

else//没找到

{

MessageBox(NULL,"没有找到该班级相关信息！","FAIL!",MB\_ICONINFORMATION|MB\_OK);

return -1;

}

}

/\*\*

\* 函数名称: ToQueryLessonByTimeAndPlace

\* 函数功能: 通过时间空间查询课堂信息

\* 输入参数: 无

\* 返 回 值: 成功返回1，出错返回-1

\*

\*/

int ToQueryLessonByTimeAndPlace(void)

{

int week,day,period;

char s1[81],s2[81];

printf("请按要求输入该课堂的上课地点和时间: \n\n");

printf("教学楼 教室 周次 星期 节数(1~6)\n");

scanf("%s%s%d%d%d",s1,s2,&week,&day,&period);

fflush(stdin);

int timei=TimestrToNum(week,day,period);//时间hash值

int placei=PlacestrToNum(s1,s2);//空间hash值

if(timei==-1 || placei==-1)//格式错误

{

MessageBox(NULL,"上课时间空间输入错误！","FAIL!",MB\_ICONINFORMATION|MB\_OK);

return -1;

}

int res=SearchByTimeAndPlace(timei ,placei);//根据时间和空间查找

if(res==1)

{

printf("\n信息输出完成，按任意键继续...");

getch();

return 1;

}

else

{

MessageBox(NULL,"此时间该教室没有课！","FAIL!",MB\_ICONINFORMATION|MB\_OK);

return -1;

}

}

/\*\*

\* 函数名称: ToQueryLessonByCourse

\* 函数功能: 通过课程名称查询信息

\* 输入参数: 无

\* 返 回 值: 成功返回1，出错返回-1

\*

\*/

int ToQueryLessonByCourse(void)

{

char s[100];

printf("请输入要查询的课程名称：\n ");

scanf("%s",s);

fflush(stdin);

int res=SearchByCourse(s);//通过课程名查询

if(res==1)

{

printf("\n信息输出完成，按任意键继续...");

getch();

return 1;

}

else

{

MessageBox(NULL,"没有找到该课程相关信息！","FAIL!",MB\_ICONINFORMATION|MB\_OK);

return -1;

}

}

/\*\*

\* 函数名称: ToQuerySpareClassroom

\* 函数功能: 查询空闲教室信息

\* 输入参数: 无

\* 返 回 值: 成功返回1，出错返回-1

\*

\*/

int ToQuerySpareClassroom(void)

{

int week,day,period;

char s1[81],s2[81];

printf("请输入要查询的教室以及时段: \n\n");

printf("教学楼 教室 周次 星期 节数(1~6)\n");

scanf("%s%s%d%d%d",s1,s2,&week,&day,&period);

fflush(stdin);

int timei=TimestrToNum(week,day,period);//获得时间hash值

int placei=PlacestrToNum(s1,s2);//获得空间hash值

if(timei==-1 || placei==-1)//时间空间格式输入错误

{

MessageBox(NULL,"教室与时段信息输入错误！","FAIL!",MB\_ICONINFORMATION|MB\_OK);

return -1;

}

int res=SearchByTimeAndPlace(timei ,placei);//查找

if(res==1)

{

MessageBox(NULL,"此时间此教室有课！非空闲！","FAIL!",MB\_ICONINFORMATION|MB\_OK);

getchar();

return -1;

}

else//没有查找到

{

printf("\n此时间此教室为空闲教室，按任意键继续...");

getch();

return 1;

}

}

/\*\*

\* 函数名称: AverageDx

\* 函数功能: 计算v数组中数的平均间隔

\* 输入参数: 要处理的数组

\* 返 回 值: 间隔平均值

\*

\*/

double AverageDx(std::vector<int> v)//计算v数组中数的平均间隔

{

//double aver=0;

std::sort(v.begin(),v.end());

double sumdx=0;

int n=v.size();

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

{

if(i<v.size()-1)//数组中至少要有2个数

{

sumdx += (v[i+1]-v[i]);

}

}

return n>1? (sumdx/(1.0\*(n-1))) : 0;//数组中只有一个数，那么间隔设为0

}

/\*\*

\* 函数名称: ToStatClassroomRate

\* 函数功能: 统计教室利用率以及能效分析

\* 输入参数: 无

\* 返 回 值: 返回1

\*

\*/

int ToStatClassroomRate(void)

{

int roomnum=0; double ratesum=0.0,energ\_rate\_sum=0.0,dt\_sum=0.0,sum\_cnt=0;

int day=0,night=0;//统计一个教室白天和晚上的上课数量以统计能效比

std::vector<int>dt ;//用来统计一个教室上课的平均间隔

int peri[6];//储存每个教室每个时间段的课堂总数

int sum\_peri[6];

memset(sum\_peri,0,sizeof sum\_peri);

printf("各个有课的教室的综合分析如下：\n\n\n");

// printf("教室 利用率(教室的课堂数/一学期课堂总数)\n");

for(int i=1;i<=259;i++)

{

memset(peri,0,sizeof peri);

if(les\_place[i].room\_next==NULL)//如果教室没有课程

continue;

else

{

int cnt=0; roomnum++;

day=night=0;//每个教室的白天和晚上的上课数量清零

LESSON\* p=les\_place[i].room\_next;

while (p!=NULL)

{

cnt++;

dt.push\_back(TimestrToNum(p->week,p->day,p->period));

peri[p->period -1]++;//统计每个时段的课堂数

sum\_peri[p->period -1]++;//统计所有教室每个时段的课堂数

if(p->period<=4) day++;//如果白天上课

else night++;//如果晚上上课

p=p->time\_next;

}

double rate=(1.0\*cnt)/(840.0);

ratesum += rate;//用于计算总的利用率

energ\_rate\_sum += (1.0\*day)/(1.0\*(day+night));//用于计算平均能效比

double aver = AverageDx(dt);//教室上课的平均的时间间隔

dt\_sum += aver;//用于计算平均能效比

sum\_cnt+=cnt;//用于统计所有教室一共有多少个课堂

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

// printf("%s %s : %d/%d = %.4f\n\n",les\_place[i].building,les\_place[i].classroom,cnt,840,rate);

if((1.0\*day)/(1.0\*(day+night)) <0.33)

printf("！！！当前教室能效比过低！！！\n\n");

if(aver>45)

printf("！！！当前教室的上课平均间隔过高,请适当增加课程！！！\n\n");

if(aver==0)

printf("！！！当前教室排课过少，请适当增加排课！！！\n\n");

printf("教学楼： %s\n\n",les\_place[i].building);

printf("教室： %s\n\n",les\_place[i].classroom);

printf("上课时段分布：\n\n");

printf(" 1-2节： %d %.2f%%\n\n",peri[0],100.0\*peri[0]/(1.0\*cnt));

printf(" 3-4节： %d %.2f%%\n\n",peri[1],100.0\*peri[1]/(1.0\*cnt));

printf(" 5-6节： %d %.2f%%\n\n",peri[2],100.0\*peri[2]/(1.0\*cnt));

printf(" 7-8节： %d %.2f%%\n\n",peri[3],100.0\*peri[3]/(1.0\*cnt));

printf(" 9-10节： %d %.2f%%\n\n",peri[4],100.0\*peri[4]/(1.0\*cnt));

printf(" 11-12节：%d %.2f%%\n\n\n",peri[5],100.0\*peri[5]/(1.0\*cnt));

printf("教室上课平均间隔：%0.3f （节）\n\n",aver);

printf("能效比(白天上课数/上课总数数)：%d/%d = %.3f\n\n",day,day+night,(1.0\*day)/(1.0\*(day+night)));

printf("利用率(教室的课堂数/一学期课堂总数)：%d/%d = %.4f\n\n",cnt,840,rate);

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

dt.clear();

}

}

printf("\n总结：\n\n");

printf("\n\n所有教室上课时间分布汇总：\n\n");

printf(" 1-2节： %d %.2f%%\n\n",sum\_peri[0],100.0\*sum\_peri[0]/(1.0\*sum\_cnt));

printf(" 3-4节： %d %.2f%%\n\n",sum\_peri[1],100.0\*sum\_peri[1]/(1.0\*sum\_cnt));

printf(" 5-6节： %d %.2f%%\n\n",sum\_peri[2],100.0\*sum\_peri[2]/(1.0\*sum\_cnt));

printf(" 7-8节： %d %.2f%%\n\n",sum\_peri[3],100.0\*sum\_peri[3]/(1.0\*sum\_cnt));

printf(" 9-10节： %d %.2f%%\n\n",sum\_peri[4],100.0\*sum\_peri[4]/(1.0\*sum\_cnt));

printf(" 11-12节：%d %.2f%%\n\n\n",sum\_peri[5],100.0\*sum\_peri[5]/(1.0\*sum\_cnt));

printf("\n共%d个教室本学期被安排了课程,其平均利用率为%.3f%% \n\n",roomnum,100.0\*ratesum/(1.0\*roomnum));

printf("其中总的平均能效比为：%.3f\n\n",energ\_rate\_sum/(1.0\*roomnum));

printf("每个教室上课的平均上课间隔为：%.3f (节)\n\n\n\n",dt\_sum/(1.0\*roomnum));

if(ratesum/(1.0\*roomnum)<0.5)

printf("当前教室的平均利用率小有5%% 利用率比较低，应当适当增加排课\n\n");

if(energ\_rate\_sum/(1.0\*roomnum)<0.3)

printf("当前教室的平均能效比较低，排课时应注意节能\n\n");

else

printf("当前教室的能效比较高，课程安排的节能参数优秀\n\n");

if(dt\_sum/(1.0\*roomnum)>15)

printf("当前教室的上课的平均时间间隔较大，课程的时间分布较分散\n\n按任意键继续...");

else

printf("当前教室的上课平均间隔合理，时间分布适中\n\n按任意键继续...");

getch();

return 1;

}

/\*\*

\* 函数名称: ToStatTeacherRate

\* 函数功能: 教师任课分析

\* 输入参数: 无

\* 返 回 值: 1

\*

\*/

int ToStatTeacherRate(void)

{

int sum=gen\_info.size();

bool done[sum+5];

int cnt=0,sum\_teacher=0;

double credit=0,sum\_credit=0;

char\* addr;

char\*\* paddr=&addr;

memset(done,0,sizeof done);

printf("教师任课情况分析如下：\n\n");

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

//printf("sum: %d\n",sum);

for(int i=0;i<sum;i++)

{

if(done[i])

continue;

done[i]=1;

credit=0; cnt=1;

credit += strtod(gen\_info[i].credit , paddr);//统计一个老师任课的总学分

sum\_teacher += 1;//相当于数一数一共有多少个老师

for(int j=i+1;j<sum;j++)

{

if(done[j])

continue;

if(!strcmp(gen\_info[i].teacher , gen\_info[j].teacher) )

{

done[j]=1;

credit += strtod(gen\_info[j].credit , paddr);//统计一个老师任课的总学分

cnt++;//统计一个老师的任课数量

}

}

sum\_credit += credit;//统计所有老师总学分

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

if(16\*credit > 88)

printf("\n！！！该教师教学任务较重 ！！！\n\n");

else if(16\*credit < 24)

printf("\n！！！该教师担任课时较少 ！！！\n\n");

printf("No.%d\n",sum\_teacher);

printf("教师姓名： %s\n",gen\_info[i].teacher);

printf("职称： %s\n",gen\_info[i].title);

printf("任课数量: %d\n",cnt);

printf("总任课学时： %.0f\n",16\*credit);

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

}

printf("总结：\n\n本学期一共有%d位老师任课,其中平均每位老师担任%.3f门课,每位老师平均任课学时为%.3f\n\n\n",sum\_teacher,(1.0\*sum)/(1.0\*sum\_teacher),(16\*sum\_credit)/(1.0\*sum\_teacher));

if((16\*sum\_credit)/(1.0\*sum\_teacher)>88 )

{

printf(" 教师的平均任课学时过多,应该多引进教师,分担教学任务!\n");

}

else if((16\*sum\_credit)/(1.0\*sum\_teacher)<24)

{

printf(" 教师的平均任课学时过少,人浮于事,应该增加教学任务!\n");

}

else

printf(" 教师的平均任课学时适中,教学安排相对合理!\n");

printf("\n\n按任意键继续...");

getch();

return 1;

}

/\*\*

\* 函数名称: MakeClassno

\* 函数功能: 生成班号字符串

\* 输入参数: 编号、 储存班号的字符串首地址

\* 返 回 值: 1

\*

\*/

int MakeClassno(int i,char s[])

{

switch (i)

{

case 1: strcpy(s,"计算机1701");break;

case 2: strcpy(s,"计算机1702");break;

case 3: strcpy(s,"计算机1703");break;

case 4: strcpy(s,"计算机1704");break;

case 5: strcpy(s,"计算机1705");break;

case 6: strcpy(s,"计算机1706");break;

case 7: strcpy(s,"计算机1707");break;

case 8: strcpy(s,"计算机1708");break;

case 9: strcpy(s,"计算机1709");break;

case 10: strcpy(s,"计算机1710");break;

case 11: strcpy(s,"计算机1711");break;

case 12: strcpy(s,"计算机1712");break;

case 13: strcpy(s,"计算机1713");break;

case 14: strcpy(s,"计算机校交17");break;

case 15: strcpy(s,"计算机卓工17");break;

case 16: strcpy(s,"计算机1601");break;

case 18:strcpy(s,"计算机1602");break;

case 19:strcpy(s,"计算机1603");break;

case 20:strcpy(s,"计算机1604");break;

case 21: strcpy(s,"计算机1605");break;

case 22: strcpy(s,"计算机1606");break;

case 23: strcpy(s,"计算机1607");break;

case 24: strcpy(s,"计算机1608");break;

case 25: strcpy(s,"计算机1609");break;

case 26: strcpy(s,"计算机1610");break;

case 27: strcpy(s,"计算机ACM16");break;

case 28: strcpy(s,"计算机卓工16");break;

case 29: strcpy(s,"物联网16");break;

case 30: strcpy(s,"信安1601");break;

case 31: strcpy(s,"信安1602");break;

case 32: strcpy(s,"信安1603");break;

case 33: strcpy(s,"计算机1501");break;

case 34:strcpy(s,"计算机1502");break;

case 35:strcpy(s,"计算机1503");break;

case 36:strcpy(s,"计算机1504");break;

case 37: strcpy(s,"计算机1505");break;

case 38: strcpy(s,"计算机1506");break;

case 39: strcpy(s,"计算机1507");break;

case 40: strcpy(s,"计算机1508");break;

case 41: strcpy(s,"计算机1509");break;

case 42: strcpy(s,"计算机1510");break;

case 43: strcpy(s,"计算机ACM15");break;

case 44: strcpy(s,"计算机卓工15");break;

case 45: strcpy(s,"物联网15");break;

case 46: strcpy(s,"信安1501");break;

case 47: strcpy(s,"信安1502");break;

case 48: strcpy(s,"信安1503");break;

case 49: strcpy(s,"计算机校交16");break;

}

return 1;

}

/\*\*

\* 函数名称: ToStatClassnoRate

\* 函数功能: 班级上课分析

\* 输入参数: 无

\* 返 回 值: 1

\*

\*/

int ToStatClassnoRate(void)

{

int sum\_lesson=0;//所有班级一共上了多少节课

int sum\_course=0;//所有班级一共上了多少门课

int cnt\_lesson=0,cnt\_course=0;

int n=les\_classno.size();

int sum\_class=0;

char classno[25];

double classno\_dt=0.0;//一个班级的平均上课的时间间隔；

double dt\_sum=0.0; //总的时间间隔

int peri[6];//储存每个班级每个时间段的课堂总数

int sum\_peri[6];

memset(sum\_peri,0,sizeof sum\_peri);

printf("班级上课分析如下：\n\n");

for(int j=1;j<=49;j++)

{

MakeClassno(j,classno);

cnt\_lesson=0,cnt\_course=0;//一个班级的课程和课堂数

memset(peri,0,sizeof peri);

std::vector<int>dt;

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

{

if(strstr(les\_classno[i].classno,classno)==NULL)

continue;

LESSON\* p=les\_classno[i].lesson;

while(p!=NULL)//按课程遍历统计

{

cnt\_course++;

cnt\_lesson += (p->to) - (p->from) +1;

peri[p->period -1] += (p->to) - (p->from) +1 ;//统计一个班级上课在各个时段的分布

for(int k=p->from;k<=p->to;k++)//计算平均间隔

dt.push\_back(TimestrToNum(k,p->day,p->period));

p=p->classno\_next;

}

}

if(cnt\_course==0)//没有录入该班级的课程，不打印信息

continue;

dt\_sum += (classno\_dt=AverageDx(dt)) ;//用于计算总的平均时间间隔

for(int i=0;i<6;i++) sum\_peri[i]+=peri[i];//用于统计总的时间分布

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

if(cnt\_lesson>=350)

printf("！！！当前班级课业压力过大！！！\n\n");

else if(cnt\_lesson<100)

printf("！！！当前班级课程太少,应适当增加学习内容！！！\n\n");

if(classno\_dt>5)

printf("！！！当前班级的课堂平均时间间隔过大，课程分布稀疏 ！！！\n\n");

printf("班级： %s\n\n",classno);

printf("上课时段分布：\n\n");

printf(" 1-2节： %d %.2f%%\n\n",peri[0],100.0\*peri[0]/(1.0\*cnt\_lesson));

printf(" 3-4节： %d %.2f%%\n\n",peri[1],100.0\*peri[1]/(1.0\*cnt\_lesson));

printf(" 5-6节： %d %.2f%%\n\n",peri[2],100.0\*peri[2]/(1.0\*cnt\_lesson));

printf(" 7-8节： %d %.2f%%\n\n",peri[3],100.0\*peri[3]/(1.0\*cnt\_lesson));

printf(" 9-10节： %d %.2f%%\n\n",peri[4],100.0\*peri[4]/(1.0\*cnt\_lesson));

printf(" 11-12节：%d %.2f%%\n\n\n",peri[5],100.0\*peri[5]/(1.0\*cnt\_lesson));

printf("总共上课课程数: %d \n\n",cnt\_course);

printf("总共上课节数： %d \n\n",2\*cnt\_lesson);

printf("平均上课间隔： %.3f (节)\n\n",classno\_dt);

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

sum\_lesson += cnt\_lesson;

sum\_course += cnt\_course;

sum\_class++;

dt.clear();

}

// printf("class\_sum:%d\n",sum\_class);

double everage\_course = 1.0\*sum\_course/(1.0\*sum\_class);

double everage\_lesson = 1.0\*sum\_lesson/(1.0\*sum\_class);

printf("\n\n\n总结：\n\n班级上课时间分布汇总：\n\n");

printf(" 1-2节： %d %.2f%%\n\n",sum\_peri[0],100.0\*sum\_peri[0]/(1.0\*sum\_lesson));

printf(" 3-4节： %d %.2f%%\n\n",sum\_peri[1],100.0\*sum\_peri[1]/(1.0\*sum\_lesson));

printf(" 5-6节： %d %.2f%%\n\n",sum\_peri[2],100.0\*sum\_peri[2]/(1.0\*sum\_lesson));

printf(" 7-8节： %d %.2f%%\n\n",sum\_peri[3],100.0\*sum\_peri[3]/(1.0\*sum\_lesson));

printf(" 9-10节： %d %.2f%%\n\n",sum\_peri[4],100.0\*sum\_peri[4]/(1.0\*sum\_lesson));

printf(" 11-12节：%d %.2f%%\n\n\n",sum\_peri[5],100.0\*sum\_peri[5]/(1.0\*sum\_lesson));

printf("\n\n平均每个班级上课课程数为 %.3f \n\n平均每个班级上课节数为 %.3f\n\n",everage\_course,2\*everage\_lesson);

printf("平均每个班级的平均上课间隔为： %.3f (节)\n\n", 1.0\*dt\_sum/(1.0\*sum\_class));

if(1.0\*dt\_sum/(1.0\*sum\_class)>5)//平均时间间隔判断

printf("班级的平均上课时间间隔过大，排课分布稀疏，应适当紧密\n\n");

else

printf("班级的平均上课时间间隔适中，排课分布相对合理\n\n");

if(everage\_lesson>350)

printf("班级的平均课堂数量过多,应当适当减小课业压力！\n\n");

else if(everage\_lesson<100)

printf("班级的平均课堂数量过少,应当适当增加学习内容！\n\n");

else

printf("班级的平均课堂数量情况适中,课程安排相对合理\n\n");

printf("按任意键继续...");

getch();

return 1;

}

/\*\*

\* 函数名称: ToTraverseAll

\* 函数功能: 遍历所有课堂信息

\* 输入参数: 无

\* 返 回 值: 1

\*

\*/

int ToTraverseAll(void)

{

int no=0;

for(int i=1;i<845;i++)

{

LESSON\* p=les\_time[i].time\_next;

while(p)

{

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

printf("No.%d\n",++no);

printf(" 课程：%s\n",p->course);

printf(" 周次：%d\n",p->week);

printf(" 星期：%d\n",p->day);

printf(" 节数：%d\n",p->period);

printf(" 教学楼：%s\n",p->building);

printf(" 教室：%s\n",p->classroom);

printf(" 班级：%s\n",p->classno);

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

if((no%500) == 0)

{

printf("\n\n按任意键继续浏览下一页...");

getch();

ClearScreenTotalInputArea();//清屏

}

p=p->room\_next;

}

}

printf("\n\n遍历完成！按任意键继续...");

getch();

return 1;

}