华中科技大学计算机科学与技术学院

C语言课程设计报告

题目：C语言课程设计报告

专　　业：计算机科学与技术

班　　级：1601

学　　号：U201614526

姓　　名：田志伟

成　　绩：

指导教师：甘早斌

完成日期：2017年 9月 5日

目 录

一、系统需求分析

科研项目信息管理系统用于学校管理各个院系团队的科研项目相关信息，主要包括院系基本信息，科研团队基本信息和科研项目基本信息。

科研项目信息管理系统主要需实现以下功能。

1. 基本信息的录入，修改和删除

科研项目信息管理系统的基本信息主要包括以下三类。

（1） 院系基本信息：院系名称，负责人，联系电话。

（2） 科研团队基本信息：团队名称，负责人，教师人数，研究生人数，所属院系。

（3） 科研项目基本信息：项目编号，项目类别，起始时间，项目经费，项目负责人，所属团队。

系统应实现对以上三种信息的录入，修改，删除功能。在实现功能的同时，应尽量快捷方便的自动保存数据，以便操作。

2. 基本信息的查询功能

系统应实现对以上三种数据的查询功能，并提供按多种条件进行查询的方式

（1） 院系信息查询功能

以院系负责人为条件查找满足条件的院系基本信息。

以院系名称的关键字为条件查找满足条件的院系基本信息。

（2） 科研团队信息查询功能

以团队名称的关键字为条件查找满足条件的团队基本信息。

以教师人数为条件查找并显示满足条件的科研团队基本信息。

（3） 科研项目基本信息查询功能

以项目编号为条件查找满足条件的科研项目基本信息。

以所属团队为条件查找满足条件的科研项目基本信息。

3. 数据统计功能

在以上数据信息的基础上，对各方面数据按多种条件进行统计并排序。

（1） 统计各院系教师总数，研究生总数，及研究生与教师人数比，按人数比降序输出。

（2） 统计各院系科研项目总数，973 项目数，863 项目数，以及科研总经费，

按科研项目降序输出。

（3） 统计自然科学基金项目总数最多的 10 个科研团队，按项目数降序输出。

（4） 统计科研项目总数与教师总数比最高的 5 个科研团队，按比值降序输出。

4. 数据存取功能

以上三种信息在程序运行时以链表结构形式存在，并且输的存储采用动态存储分配方式，同时在外存上以二进制文件的形式对数据进行存储，应保证数据在内存中与外存中内容的一致性。

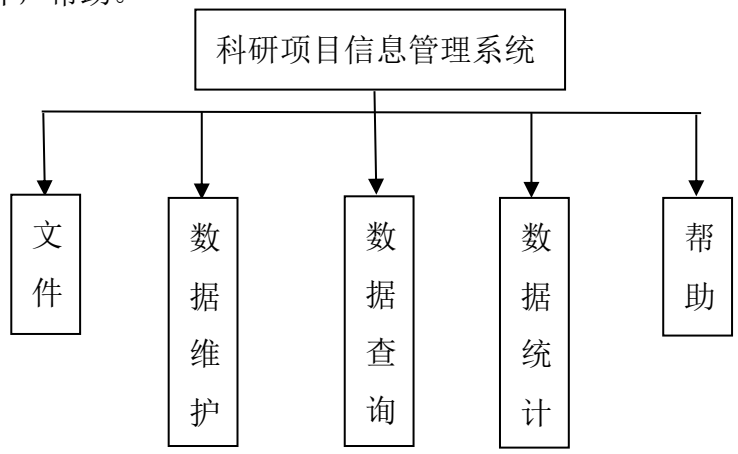
上述四种功能是系统的基本功能，此外需根据情况增加辅助功能，使系统具有良好的人机交互界面，易于操作，并体现良好的健壮性和容错性。

二、总体设计

总体设计是在功能需求分析的基础上，设计系统的功能结构和数据结构，即按照功能要求构造系统的功能框架，并确定数据的描述规范和数据之间的关系。通过总体设计建立目标系统的逻辑关系，系统功能框架通常用模块层次结构图来描述，能够直接地体现功能模块之间的调用关系。

2.1 功能结构设计

功能结构设计即按功能进行模块划分、建立模块的层次结构及调用关系、确定模块间的接口及人机界面。根据功能需求，按相关性对系统功能进行分解，组合，补充，形成如图所示的 5 个系统功能模块：文件，数据维护，数据查询，数据统计，帮助。



下面分别描述这 5 个模块及其子模块功能。

1. 文件

文件模块的功能包括一系列与系统启动运行和系统结束运行相关的环境维护和数据保障操作。进一步分为界面初始化，数据加载，数据保存，退出系统四个模块。

（1）界面初始化子模块：用于设置控制台窗口显示模式，将屏幕窗口设置为 80*25 的文本字符界面，设置窗口标题栏，清屏并显示系统菜单和系统状态栏。

（2）数据加载子模块：用于将内存中的多个数据文件中的代码数据和基础数

据读入内存，并构造数据链表，同时输出数据加载提示信息。

（3）数据保存子模块：用于将内存中的代码表数据和链表数据按缺省路径分别保存到各个数据文件。

（4）退出系统子模块：释放系统运行过程中申请的动态存储区，关闭控制台标准输入和标准输出设备句柄，将标题栏设为“运行结束”，清除屏幕窗口信息，结束系统运行。

2. 数据维护

数据维护模块完成对系统代码信息的查看和三种基础信息的录入，修改，删除功能，保证数据的准确性，完整性，时效性。该模块包含项目类别代码，院系基本信息，团队基本信息，科研项目基本信息四个子模块。

（1）项目类别代码：用于查看项目类别代码表

（2）三类信息维护：分别用于录入、修改、删除院系基本信息、团队基本信息、科研项目基本信息，并对三种信息进行自动保存，以保证这三种基础数据的正确性，在内外存的一致性。

3. 数据查询

数据查询模块提供对系统代码信息和三种基础数据信息按多种条件进行查询的功能。分为三个子模块：院系基本信息查询，科研团队基本信息查询，科研项目基本信息查询。

（1） 院系基本信息查询：提供按院系负责人查找院系和院系名称关键字查找院系功能，

（2） 科研团队基本信息查询：提供按团队名称关键字查找团队和按教师人数条件查找科研团队功能。

（3） 科研项目基本信息查询：提供按项目编号查找科研项目和按所属团队查找科研项目功能。

4. 数据统计

数据统计模块提供对三种数据进行多方面统计的功能。按统计条件，该模块划分为教师与研究生人数及人数比、各院系科研项目总体信息、优秀科研团队、教师人均科研项目情况四个子模块。

（1） 教师与研究生人数及人数比:用于统计各院系教师总数,研究生总数，及研究生与教师人数比，并按人数比降序输出。

（2） 各院系科研项目总体信息：用于统计各院系科研项目总数，973 项目数，863 项目数，以及科研总经费，并按科研项目降序输出。

（3） 优秀科研团队：用于统计自然科学基金项目总数最多的 10 个科研团队，按项目数降序输出。

（4） 教师人均科研项目：用于统计科研项目总数与教师总数比最高的 5 个科研团队，按比值降序输出。

5. 版权信息

用于显示系统版本信息和系统的版权信息。

2.2 数据结构设计

数据结构设计即确定数据的逻辑结构和物理结构，数据的逻辑结构是对数据之间关系的描述，数据的物理结构是数据逻辑结构在计算机中的表示。

按任务要求，系统需要处理的信息有三种：院系信息，科研团队信息，科研项目信息。这三种信息存在这样的关联：院系信息中的院系名称与科研团队信息中的所属院系相同，科研团队信息中的团队名称与科研项目中的所属项目相同，这些成为系统的基础数据。

系统进行统计时需要使用以上基础数据，统计结果生成的新数据称为生成数据，本系统的统计模块产生四种新数据，分别为院系研究生人数与教师人数比结构、院系各类科研项目总数、科研团队自然科学基金项目总数、科研团队教师人均科研项目。

另外，出于数据规范化、便于处理和节省存储空间等目的，我们还可以对信息中某些项进行编码，用固定长度的代码来表示长度不一的信息。

下面分别为本系统所涉及的代码数据、基础数据和生成数据的数据结构，以及数据在内存外存中的存储结构。

1. 科研项目类别代码表

数据结构名称：科研项目代码　　数据结构标识：TypeCode					
数据项名称	数据项标识	数据类型	数据长度	取值范围	示例
代码	Code	Char	1	‘0’-‘9’	1
类别	Type	String	10		“973 科研项目”

2. 院系信息表

数据结构名称：院系基本信息表　数据结构标识：CollegeInfo					
数据项名称	数据项标识	数据类型	数据长度	取值范围	示例
院系名称	name	string	20		“计算机”
负责人	dutyman	string	12		“张三”
联系电话	tel	string	15		“10086”

在内存中的存储结构：存放在十字交叉链表的主结点上，每个主链结点除了保存下一个结点地址外，还保存该院系对应科研团队基本信息链的头结点地址。

在文件中的存储结构：每条信息作为一条记录存放到二进制文件。

3. 科研团队信息表

数据结构名称：科研团队基本信息表　数据结构标识：TeamInfo					
数据项名称	数据项标识	数据类型	数据长度	取值范围	示例
团队名称	name	string	30		“127 团队”
负责人	dutyman	string	12		“张三”
所属院系	college	string	20		“计算机”
教师人数	num_tc	int	4	0-	10
研究生人数	num_stu	int	4	0-	50

在内存中的存储结构：存放在十字交叉链表中相应院系信息结点的团队基本信息链结构上，每个结点除了保存下一个团队基本信息结点的地址外，还保存该团队对应的科研项目信息链表的头结点地址。

在数据文件中的存储结构：每条信息作为一条记录存放在二进制文件中。

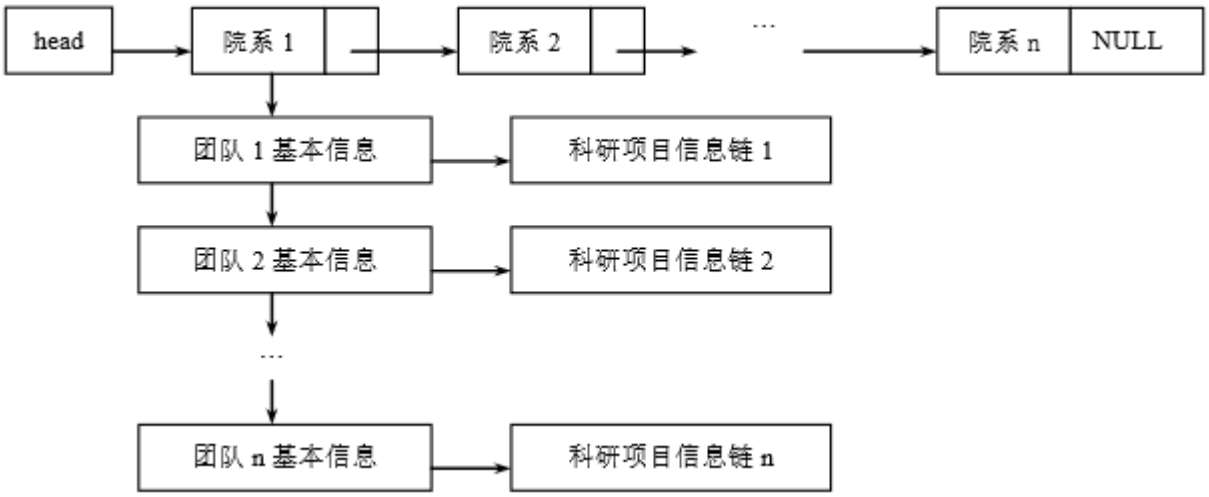
4. 科研项目信息表

数据结构名称：科研项目基本信息　数据结构标识：ProjectInfo					
数据项名称	数据项标识	数据类型	数据长度	取值范围	示例
项目编号	num	string	15		“00010”
项目类型	type	char	1		‘1’
起始时间	start	string	8		“2017/9/1”
项目经费	money	float	4	0.00-	500.00
负责人	dutyman	string	12		“张三”
所属团队	team	string	30		“127 团队”

在内存中的存储结构：存放在十字交叉链表中相应团队基本信息结点的科研项目基本信息结点上，每个结点保存下一个结点的地址。

在数据文件中的存储结构：每条信息作为一条记录保存在二进制文件中。

三类基础信息构成的十字交叉链表数据结构如下图所示：



5. 人数比信息表

数据结构名称：人数比信息表　　数据结构标识：PeoplesInfo					
数据项名称	数据项标识	数据类型	数据长度	取值范围	示例
院系名称	col_name	string	20		“计算机”
研究生总数	student	int	4	0-	50
教师总数	teacher	int	4	0-	10
人数比	scale	float	4	>0	5.00

在内存中的存储结构：存放在人数比信息单向链表的结点中

在数据文件中的存储结构：不存入外存

6. 院系科研项目总数信息表

数据结构名称：科研项目总数表　　数据结构标识：ProjectNumInfo					
数据项名称	数据项标识	数据类型	数据长度	取值范围	示例
院系名称	col_name	string	20		“计算机”
科研项目总数	total	int	4	0-	50
973 项目总数	p973	int	4	0-	10
863 项目总数	p863	int	4	0-	15

项目总资金	money	float	4	0-	1111.00

在内存中的存储结构：存放在科研项目总数信息单向链表的结点中

在数据文件中的存储结构：不存入外存

7. 科研团队自然科学基金项目总数表

数据结构名称：自然科学基金项目总数表　数据结构标识：NasiInfo					
数据项名称	数据项标识	数据类型	数据长度	取值范围	示例
团队名称	team_name	string	30		“127 团队
自然科学基金项目总数	num_nasi	int	4	0-	15
项目总资金	money	float	4	0.00	1111.00

在内存中的存储结构：存放在科研团队自然科学基金项目单向链表的结点中

在数据文件中的存储结构：不存入外存

8. 科研团队教师人均科研项目表

数据结构名称：教师人均科研项目表　数据结构标识：ProTeaInfo					
数据项名称	数据项标识	数据类型	数据长度	取值范围	示例
科研团队名称	team_name	string	30		“127 团队”
教师人数	num_tc	int	4	0-	5
科研项目总数	num_pro	int	4	0-	15
人均科研项目	scale	float	4	0.00-	3.00

在内存中的存储结构：存放在科研团队教师人均科研项目单向链表的结点中

在数据文件中的存储结构：不存入外存

以上介绍了代码数据，基础数据和生成数据的逻辑结构和物理结构设计。在系统编码实现时，还会用到一些控制信息和辅助控制信息，其相应逻辑的逻辑结构和物理结构在编码实现时再做介绍。

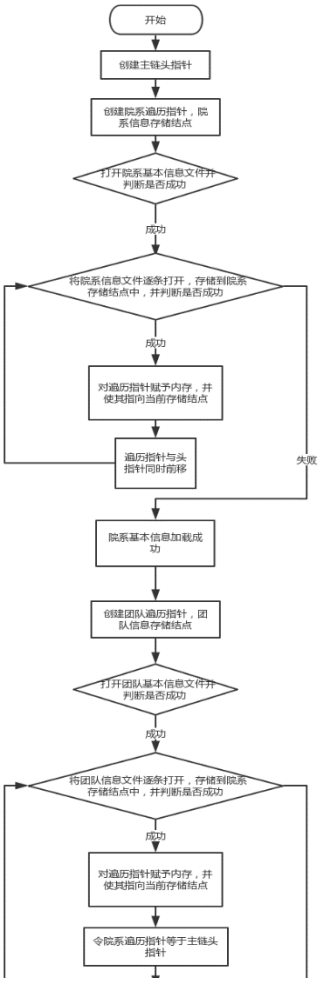
三、详细设计

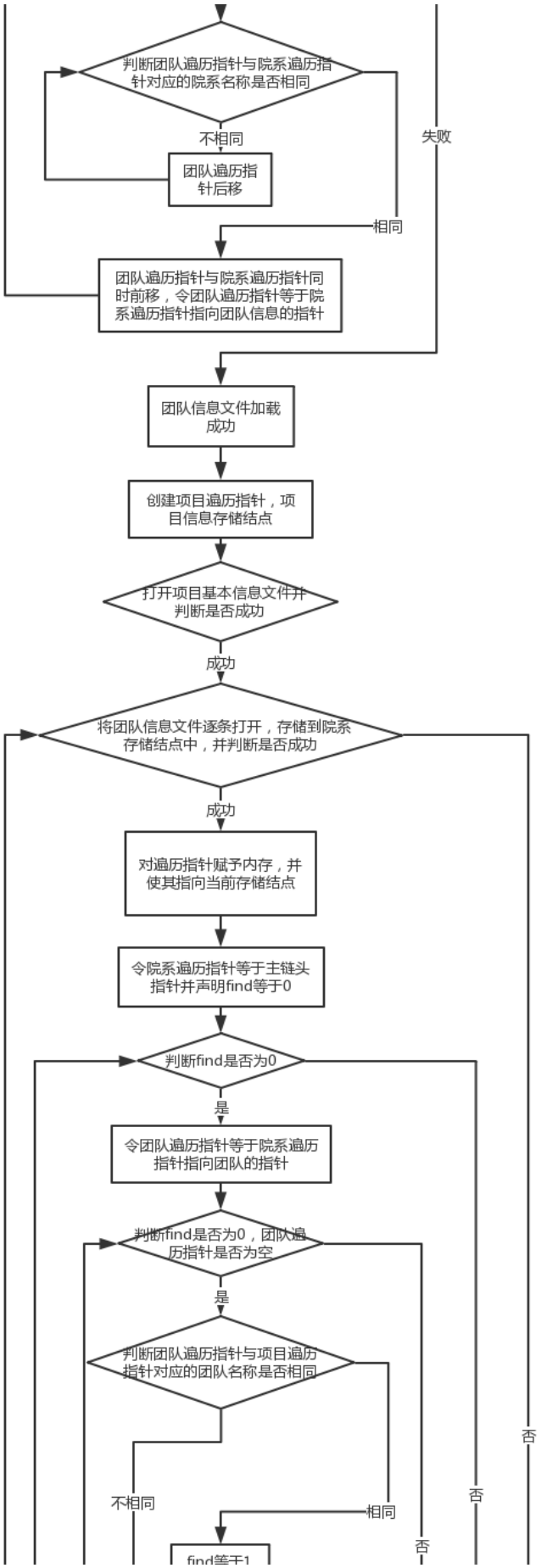
总体设计主要对系统的整体功能以及功能之间的逻辑关系进行了定义和描述，使我们看到了一个抽象的系统框架和这个框架上分层设置的模块。每个模块的功能已经确定，但如何实现这些模块的功能，需要进一步详细设计。

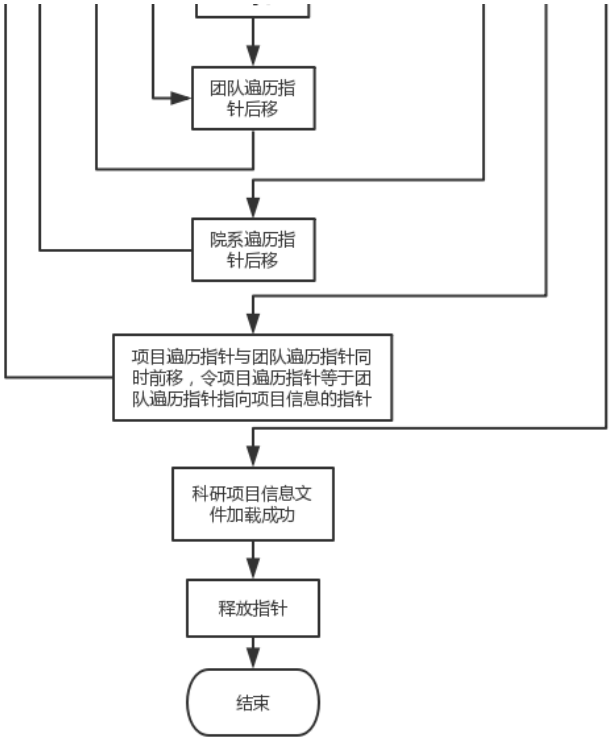
详细设计主要工作是进行算法设计，而编码阶段的主要工作则是用选定的程序设计语言实现算法。

1. 链表的创建

链表创建整体方法如下：以只读方式打开院系信息文件，将文件中的每条信息读入存储结点，遍历存入主链中，再打开团队信息文件，每条信息读入团队信息存储结点，二重循环遍历存入主链结点对应的支链中，最后打开科研项目信息文件，将每条信息读入科研项目信息存储结点，三重循环遍历存入科研团队结点对应的科研项目支链中。具体算法流程图如下：

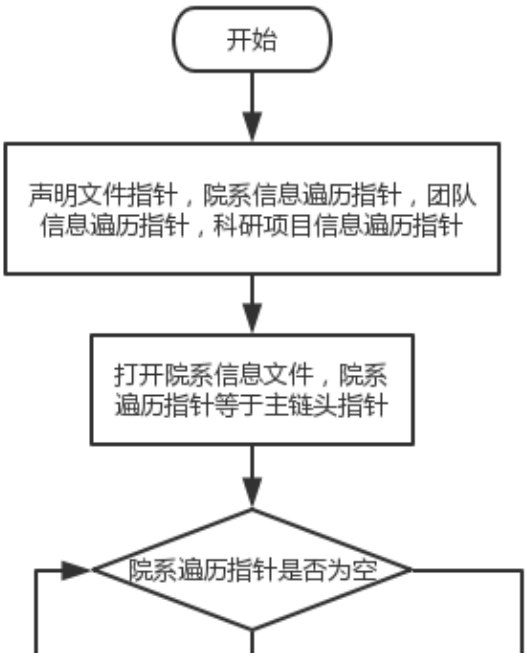


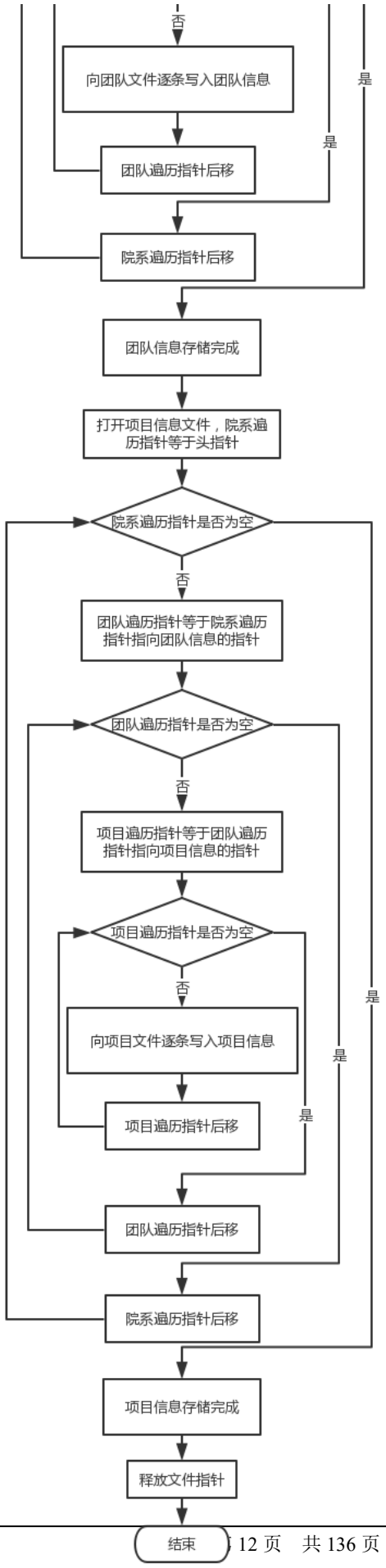

判断团队遍历指针与院系遍历指针对应的院系名称是否相同
不相同
团队遍历指针后移
失败
相同
团队遍历指针与院系遍历指针同时前移，令团队遍历指针等于院系遍历指针指向团队信息的指针
团队信息文件加载成功
创建项目遍历指针，项目信息存储结点
打开项目基本信息文件并判断是否成功
成功
将团队信息文件逐条打开，存储到院系存储结点中，并判断是否成功
成功
对遍历指针赋予内存，并使其指向当前存储结点
令院系遍历指针等于主链头指针并声明find等于0
判断find是否为0
是
令团队遍历指针等于院系遍历指针指向团队的指针
判断find是否为0，团队遍历指针是否为空
是
判断团队遍历指针与项目遍历指针对应的团队名称是否相同
否
否
否
不相同
相同
find等于1



2. 链表的保存

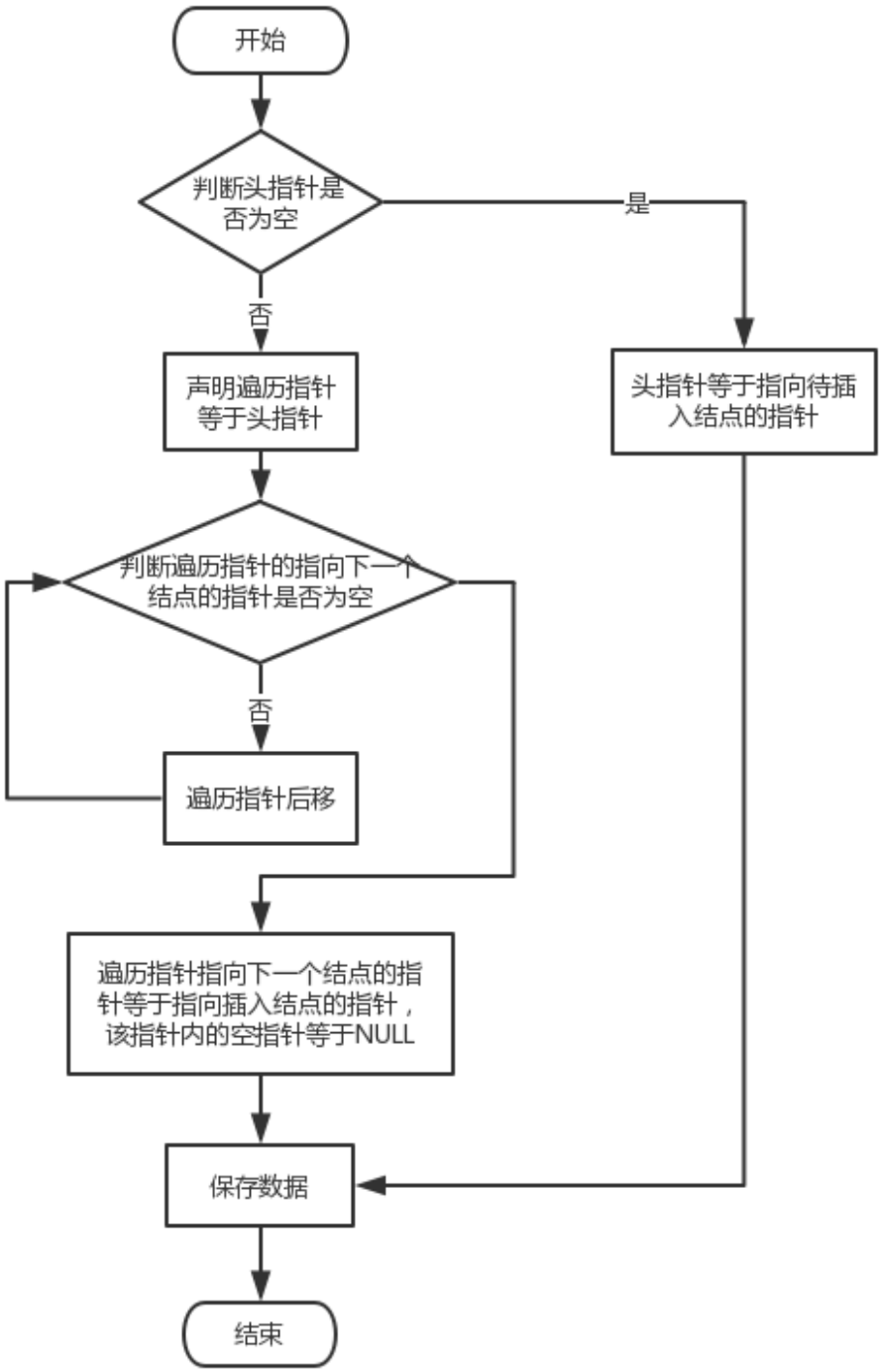
链表的保存整体方法如下：遍历主链，将链表中信息以二进制形式逐条写入存储院系信息的文件，二重循环，遍历团队信息支链，逐条写入团队信息对应的文件中，三重循环，遍历项目信息支链，逐条写入科研项目信息对应的文件中。具体算法流程图如下：




否
向团队文件逐条写入团队信息
团队遍历指针后移
是
是
院系遍历指针后移
团队信息存储完成
打开项目信息文件，院系遍历指针等于头指针
院系遍历指针是否为空
否
团队遍历指针等于院系遍历指针指向团队信息的指针
团队遍历指针是否为空
否
项目遍历指针等于团队遍历指针指向项目信息的指针
项目遍历指针是否为空
否
向项目文件逐条写入项目信息
项目遍历指针后移
是
是
是
团队遍历指针后移
院系遍历指针后移
项目信息存储完成
释放文件指针
结束

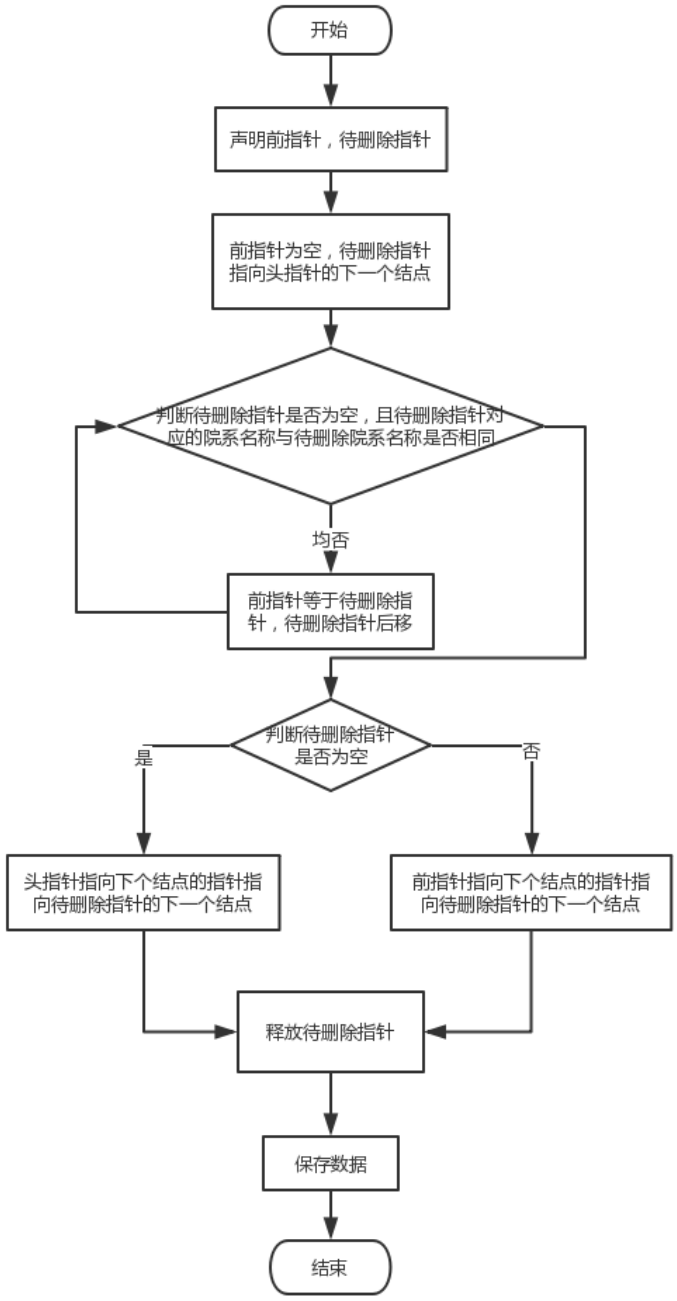
3. 链表节点的插入

链表节点插入整体方法如下：输入需要插入的信息，声明指向该信息存储位置的指针，遍历链表，直至下一个结点为空，令该结点指向下一个结点的指针等于需要插入的指针，令需要插入的指针内的空指针为 NULL。具体算法流程图如下（因三种信息插入结点方法类似，此处举插入院系结点的算法）：



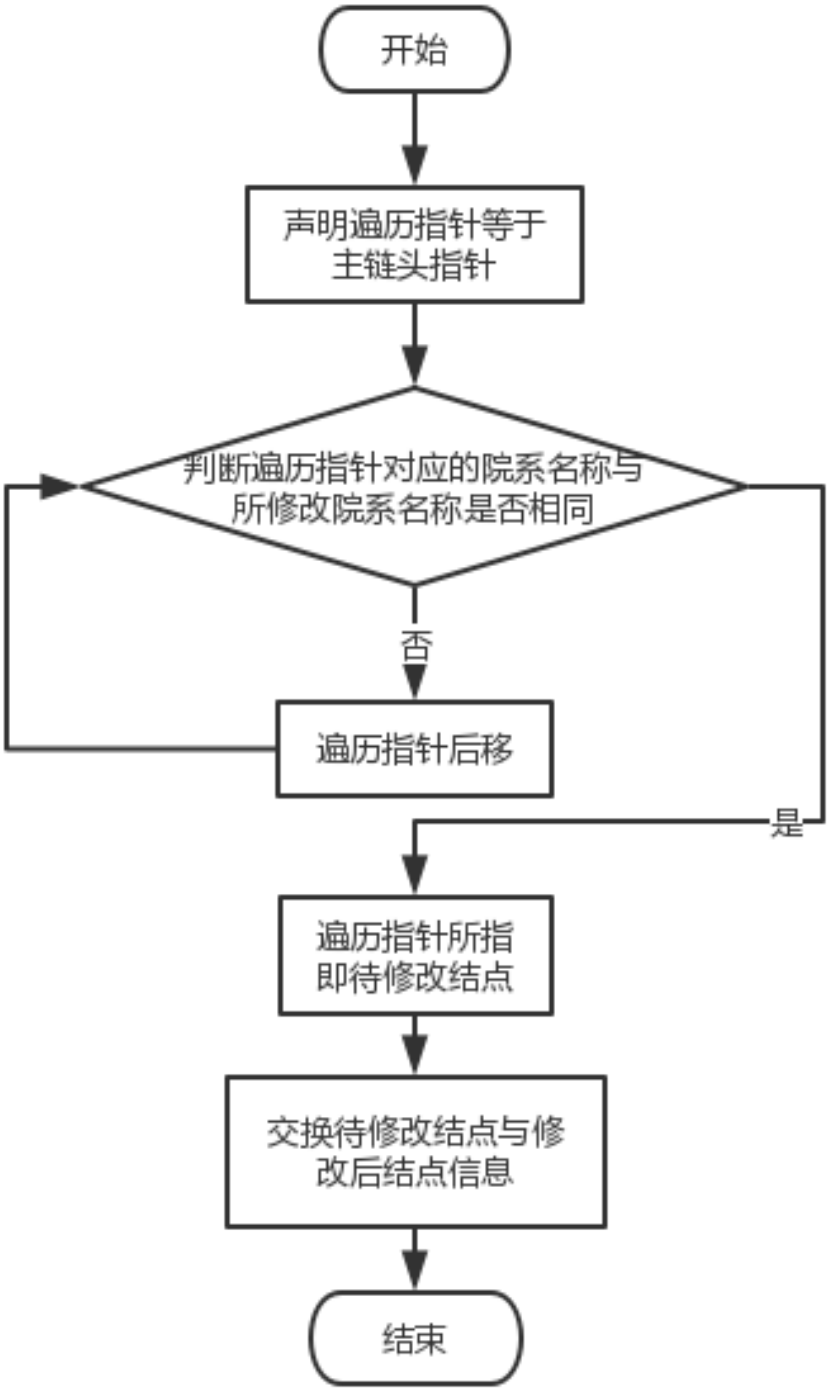
4. 链表节点的删除

链表节点删除的整体方法如下：输入所需要删除的院系名称，遍历链表寻找对应的院系信息结点，声明前指针指向该结点的前一个结点，若前指针为空，则令头指针指向下一个结点的指针指向待删除结点的下一个结点，若前指针不为空，则令前指针指向下一个结点的指针指向待删除结点的下一个结点，最后释放指向待删除结点指针。具体算法流程图如下（因三种信息删除结点方法类似，此处举删除院系结点的算法）：



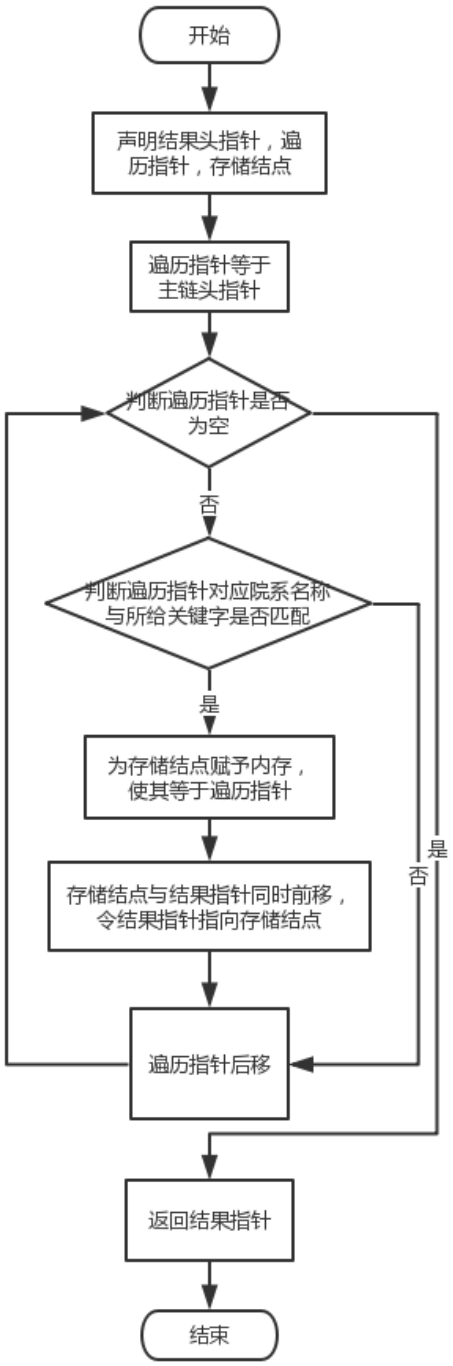
5. 链表节点的修改

链表节点修改的整体方法如下：遍历链表查找需要修改的院系信息结点，交换该节点与替换结点的所有数据。具体算法入下（因三种信息修改结点方法类似，此处举修改院系结点的算法）：



6. 链表节点的查找

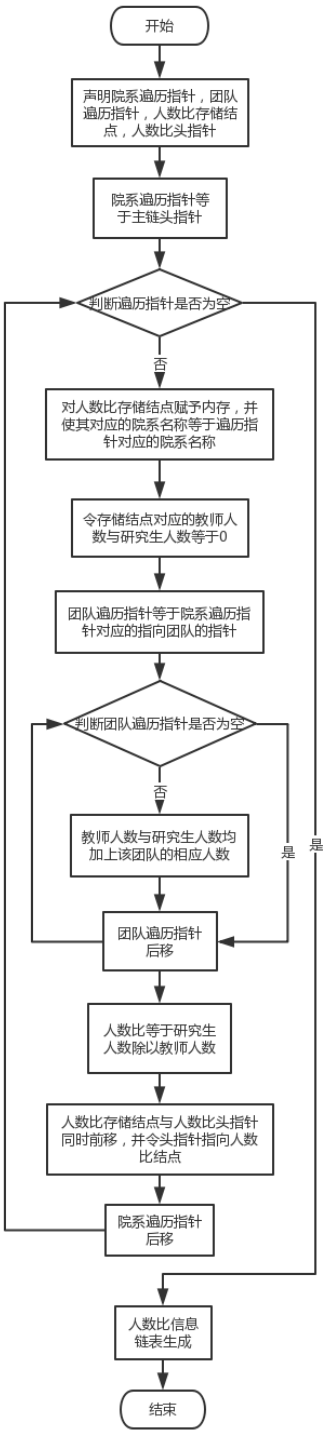
链表节点查找的整体方法如下（因各种查找方法类似，所以此处举按院系名称关键字查找院系为例）：重新声明一结果链表头指针，遍历主链表，判断院系名称与所给字符是否匹配（此处用到 strstr 函数），若匹配，则存入新建链表，继续遍历，不匹配则遍历指针后移继续遍历。直至主链结束，遍历指针为空。具体算法流程图如下：



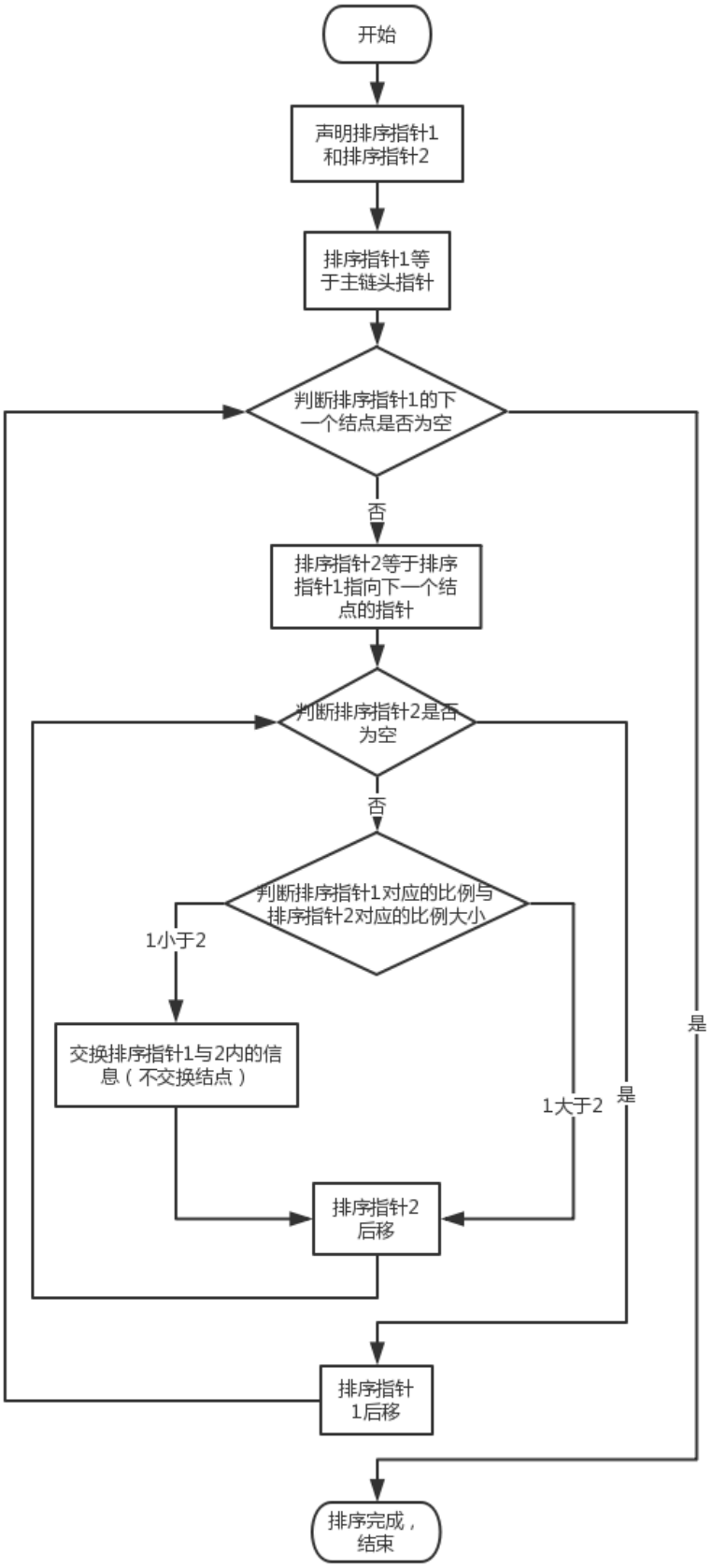
7. 总体数据的统计

数据统计整体方法如下（因几种统计方法类似，此处举统计院系人数比为例）：创建一新的结构体，存储院系名称，教师总人数，研究生总人数，研究生人数与教师人数比。遍历主链表，统计各院系人数及人数比存入新建结构体中，并建立单向链表，通过选择排序对新建链表进行排序。具体算法流程图如下：

（1）信息统计：



（2）链表排序：


开始
声明排序指针1和排序指针2
排序指针1等于主链头指针
判断排序指针1的下一个结点是否为空
否
排序指针2等于排序指针1指向下一个结点的指针
判断排序指针2是否为空
否
判断排序指针1对应的比例与排序指针2对应的比例大小
1小于2
交换排序指针1与2内的信息（不交换结点）
1大于2
是
排序指针2后移
是
排序指针1后移
排序完成，结束

四、系统实现

编辑器：notepad++

编译器：gcc

源程序全部代码如下：

*头文件（sicience.h）：

```
#include <stdio.h>
#include <stdlib.h>
#include <windows.h>
#include <wincon.h>
#include <conio.h>
#include <string.h>
#include <io.h>
#include <fcntl.h>
#include <sys\stat.h>
#include <ctype.h>
#include <time.h>
#include <stdbool.h>
#include <stdarg.h>

#ifndef SCIENCE_H_INCLUDED
#define SCIENCE_H_INCLUDED

#define SCR_ROW 25                  /*屏幕行数*/
#define SCR_COL 80                  /*屏幕列数*/

#define TRUE 1
#define FALSE 0
```

```
/**
 *院系基本信息链接点结构
 */
typedef struct college_node {
    char name[20];                  /*院系名称*/
    char dutyman[12];               /*负责人*/
    char tel[15];                   /*联系电话*/
    struct college_node *next;      /*指向下一结构点的指针*/
    struct team_node *tnext;        /*指向科研团队链的指针*/
} COLLEGE_NODE;

/**
 *科研团队基本信息链接点结构
 */
typedef struct team_node {
    char name[30];                  /*团队名称*/
    char dutyman[12];               /*负责人*/
    char college[20];               /*所属院系*/
    int num_tc;                     /*教师人数*/
    int num_stu;                    /*研究生人数*/
    struct team_node *next;         /*指向下一结构点的指针*/
    struct project_node *pnext;     /*指向科研项目链的指针*/
} TEAM_NODE;

/**
 *科研项目基本信息链接点结构
 */
typedef struct project_node {
    char num[15];                   /*项目编号*/
    char type;                      /*项目类型*/
    char start[8];                  /*起始时间*/
    float money;                    /*项目经费*/
```

```
    char dutyman[12];                    /*负责人*/
    char team[30];                       /*所属团队*/
    struct project_node *next;          /*指向下一结构点的指针*/
} PROJECT_NODE;

/*
  研究生总数，教师总数及人数比链结点结构
*/
typedef struct peoples_node{
    char col_name[20]; /*院系名称*/
    int student;     /*研究生总数*/
    int teacher;     /*教师总数*/
    float scale;       /*人数比*/
    struct peoples_node *next;
} PEOPLES_NODE;

/*
  各类科研项目数链结点结构
*/
typedef struct project_num_node{
    char col_name[20]; /*院系名称*/
    int total;        /*科研项目总数*/
    int p973;          /*973 项目总数*/
    int p863;          /*863 项目总数*/
    float money;      /*项目总资金*/
    struct project_num_node *next;
} PROJECT_NUM_NODE;

/*
  科研团队自然科学基金项目总数链结点结构
*/
typedef struct team_nasi_node{
```

```
    char team_name[30];   /*科研团队名称*/
    int num_nasi;     /*自然科学基金项目总数*/
    float money; /*项目总资金*/
    struct team_nasi_node *next;
} TEAM_NASI_NODE;

/*
   科研团队科研项目与教师人数比
*/
typedef struct team_protea_node{
    char team_name[30];     /*科研团队名称*/
    int num_tc;     /*教师人数*/
    int num_pro;     /*科研项目总数*/
    float scale;     /*人均科研项目*/
    struct team_protea_node *next;
} TEAM_PROTEA_NODE;

/**
  *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*/
COLLEGE_NODE *gp_head = NULL;                       /*主链头指针*/

char *gp_sys_name = "科研项目信息管理系统";      /*系统名称*/
char *gp_college_info_filename = "college.dat";          /*院系基本信息数据文件*/
char *gp_team_info_filename = "team.dat";          /*团队基本信息数据文件*/
char *gp_project_info_filename = "project.dat";         /*科研项目基本信息数据文件*/
char *gp_type_code_filename = "type.dat";           /*项目类别数据文件*/
char *gp_backup_filename = "backup.dat";     /*备份数据文件*/

char *ga_main_menu[] = {                            /*系统主菜单名*/
                        "文件(F)",
                        "数据维护(M)",
                        "数据查询(Q)",
                        "数据统计(S)",
                        "版权信息(H)"
                       };
```

```
char *ga_sub_menu[] = {
                        "[S]  数据保存",
                        "[X]  退出  ALT+X" ,
                        "[S]  项目类别代码",
                        "",
                        "[D]  院系基本信息",              /*系统子菜单名*/
                        "[P]  科研团队基本信息",
                        "[C]  科研项目基本信息",
                        "[D]  院系信息",
                        "[P]  科研团队基本信息",
                        "[C]  科研项目基本信息",
                        "[D]  教师与研究生的人数及人数比",
                        "[P]  各院系科研项目总体信息",
                        "[C]  优秀科研团队",
                        "[I]  教师人均科研项目情况",
                        "[A]  查看版本及版权信息"
                       };

int ga_sub_menu_count[] = { 2, 5, 3, 4, 1};   /*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 *gp_type_code = NULL;    /*存放项目类别代码表的数据缓冲区*/
char gc_sys_state = '\0';     /*用来保存系统状态的字符*/

unsigned long gul_type_code_len = 0;    /*项目类型代码表长度*/
```

```
HANDLE gh_std_out;              /*标准输出设备句柄*/
HANDLE gh_std_in;               /*标准输入设备句柄*/

int LoadCode(char *filename, char **pbuffer);   /*代码表加载*/
int CreatList(COLLEGE_NODE **phead);                    /*数据链表初始化*/
void ClearScreen(void);                                 /*清屏*/
void ShowMenu(void);                                    /*显示菜单栏*/
void InitInterface(void);                               /*初始化界面*/
void PopMenu(int num);                                  /*显示下拉菜单*/
void PopUp(SMALL_RECT *pRc, WORD att, LABEL_BUNDLE *pLabel, HOT_AREA
*pHotArea);   /*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 *pRc);                          /*绘制边框*/
void LocSubMenu(int num, SMALL_RECT *rc);       /*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 DealInput(HOT_AREA *pHotArea, int *piHot);      /*控制台输入处理*/
void SetHotPoint(HOT_AREA *phot_area, int hot_num);     /*设置热区*/
void RunSys(COLLEGE_NODE **phead);                      /*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(COLLEGE_NODE *hd);                        /*退出系统*/
BOOL ShowModule(char **pString, int n);

BOOL LoadDate(void);            /*数据加载*/
BOOL SaveData(void);            /*保存数据*/
BOOL ExitSys(void);             /*退出系统*/
BOOL HelpTopic(void);           /*帮助主体*/
BOOL AboutDorm(void);           /*关于系统*/
```

```
BOOL TypeCode(void);                /*项目类别代码*/
BOOL MaintainCollegeInfo(void);     /*维护院系基本信息*/
BOOL MaintainTeamInfo(void);        /*维护科研团队基本信息*/
BOOL MaintainProjectInfo(void); /*维护科研项目信息*/

BOOL QueryCollegeInfo(void);          /*查询院系基本信息*/
BOOL QueryTeamInfo(void);           /*查询科研团队基本信息*/
BOOL QueryProjectInfo(void);       /*查询科研项目基本信息*/

BOOL StatManRate(void);           /*统计人数及人数比*/
BOOL StatProjectType(void);            /*科研项目总体统计*/
BOOL StatTeam(void);              /*优秀科研团队统计*/
BOOL StatProjectPer(void);          /*人均科研项目统计*/

BOOL InsertCollegeNode(COLLEGE_NODE **hd, COLLEGE_NODE *pcol_node); /*插入院系基本信息节点*/
BOOL DelCollegeNode(COLLEGE_NODE *hd, char *col_name);      /*删除院系基本信息节点*/
BOOL ModifCollegeNode(COLLEGE_NODE *hd, char *col_name, COLLEGE_NODE *pcol_node); /*修改院系基本信息节点*/

BOOL InsertTeamNode(COLLEGE_NODE **hd,TEAM_NODE *pteamnode);    /*插入团队信息节点*/
BOOL DelTeamNode(COLLEGE_NODE *hd, char *col_name, char *team_name);   /*删除团队信息节点*/
BOOL ModifTeamNode(COLLEGE_NODE *hd, char *team_name, TEAM_NODE *pteamnode);
/*修改科研团队基本信息节点*/

BOOL InsertProjectNode(COLLEGE_NODE **hd,PROJECT_NODE *ppro_node);       /*插入科研项目信息节点*/
BOOL DelProjectNode(COLLEGE_NODE *hd, char *team_name, char *pro_num);    /*删除科
```

研项目信息结点*/

BOOL ModifProjectNode(COLLEGE_NODE *hd, char *team_name, char *num, PROJECT_NODE *ppro_node);/*修改科研项目信息结点*/

BOOL MarchString(char *string_item, char *cond); /*匹配字符串*/

COLLEGE_NODE *SeekCollegeNode(COLLEGE_NODE *hd, char *col_name);/*查找院系基本信息节点*/

TEAM_NODE *SeekTeamNode(COLLEGE_NODE *hd, char* team_name); /*查找团队基本信息结点*/

PROJECT_NODE *SeekProNode(COLLEGE_NODE *hd, char *team_name, char *num);/*查找缴费信息结点*/

COLLEGE_NODE *SeekColNodeMan(COLLEGE_NODE *hd, char* dutyman); /*按负责人查找院系信息节点*/

COLLEGE_NODE *SeekCollegeNodeNameM(COLLEGE_NODE *hd, char *namem); /*按院系名称关键字查找院系信息节点*/

TEAM_NODE *SeekTeamNodeNameM(COLLEGE_NODE *hd, char *namem); /*按团队名称关键字查找团队信息节点*/

TEAM_NODE *SeekTeamNodeTeacher(COLLEGE_NODE *hd, int juge, int num_tc);/*按教师人数查找团队信息节点*/

PROJECT_NODE *SeekProjectNodeNum(COLLEGE_NODE *hd, char *p_num); /*按项目编号查找科研项目信息节点*/

PROJECT_NODE *SeekProjectNodeT(COLLEGE_NODE *hd,char *tname); /*按所属团队查找科研项目信息节点*/

PEOPLES_NODE *StatPeopleRate(COLLEGE_NODE *hd); /*统计研究生，教师总数及人数比*/

void SortPeopleInfo(PEOPLES_NODE *ppeo_node_hd); /*按研究生教师人数比对链表排序*/

```
PROJECT_NUM_NODE *StatCollegeProjectNum(COLLEGE_NODE *hd); /*统计各院系各类项目总数*/
void SortColProInfo(PROJECT_NUM_NODE *ppro_num_node_hd); /*按科研项目降序对链表排序*/
TEAM_NASI_NODE *StatTeamNasiTen(COLLEGE_NODE *hd);  /*统计自然科学基金项目排名前十的科研团队*/
void SortTeamNasiInfo(TEAM_NASI_NODE *pteam_nasi_node_hd);
TEAM_PROTEA_NODE *StatProTeaScale(COLLEGE_NODE *hd);  /*统计科研项目教师人数比前五的团队*/
void SortProTeaInfo(TEAM_PROTEA_NODE *protea_node_hd);

BOOL SaveSysData(COLLEGE_NODE *hd);                    /*保存系统数据*/

#endif /**< TYPE_H_INCLUDED*/
```

*主函数（main.c）：

```
#include "science.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*/

    LoadDate();
    InitInterface();              /*界面初始化*/
```

```
    RunSys(&gp_head);
    CloseSys(gp_head);

    return 0;
}


/*
    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*/
}

/*
    函数名称：ShowMenu
    函数功能：显示主菜单，并设置热区，在主菜单第一项上置选标记
    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：TagMainMenu
    函数功能：在主菜单中选中标志
    输入参数：num 选中的主菜单编号
    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：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：PopMenu
    函数功能：弹出主菜单项对应的子菜单
    输入参数：指定的主菜单号
    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：在指定区域输出窗口信息并设置热区，将弹出窗口信息入栈
    输入参数：弹出窗口位置数据存放地址，弹出窗口区域字符属性，弹出窗口中标签束信
息存放地址，弹出窗口中热区信息存放地址
```

```
    输出参数：无
    返回值：无
*/
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：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：在指定子菜单做选中标记
    输入参数：选中的子菜单号
    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：选中的主菜单项号
    输出参数：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: 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: 主链头指针的地址
    返 回 值: 无
*/
void RunSys(COLLEGE_NODE **phead)
{
    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);
                    }
                }
            }
        }
```

```
                }
            }
        }
        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;

    }

    WriteConsoleOutput(gh_std_out, gp_buff_stateBar_info, size, pos, &rcMenu);

    return;
}


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;

}

/*
    函数名称：CloseSys
    函数功能：关闭系统
    输入参数：主链头指针
    输出参数：无
    返回值： 无
*/
void CloseSys(COLLEGE_NODE *hd)
{
    COLLEGE_NODE *pcol_node1=hd, *pcol_node2;
    TEAM_NODE *pteamnode1, *pteamnode2;
    PROJECT_NODE *ppro_node1, *ppro_node2;

    while (pcol_node1 != NULL) /*释放十字交叉链表的动态存储区*/
    {
        pcol_node2 = pcol_node1->next;
```

```
        pteamnode1 = pcol_node1->tnext;
        while (pteamnode1 != NULL) /*释放学生基本信息支链的动态存储区*/
        {
            pteamnode2 = pteamnode1->next;
            ppro_node1 = pteamnode1->pnext;
            while (ppro_node1 != NULL) /*释放缴费信息支链的动态存储区*/
            {
                ppro_node2 = ppro_node1->next;
                free(ppro_node1);
                ppro_node1 = ppro_node2;
            }
            free(pteamnode1);
            pteamnode1 = pteamnode2;
        }
        free(pcol_node1);   /*释放主链结点的动态存储区*/
        pcol_node1 = pcol_node2;
    }

    ClearScreen();              /*清屏*/

    /*释放存放菜单条、状态条和项目类别代码等信息动态存储区*/
    free(gp_buff_menubar_info);
    free(gp_buff_stateBar_info);
    free(gp_type_code);

    /*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: 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_m
enu_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] = ExitSys;
    pFunction[2] = TypeCode;
    pFunction[3] = NULL;
    pFunction[4] = MaintainCollegeInfo;
    pFunction[5] = MaintainTeamInfo;
    pFunction[6] = MaintainProjectInfo;
    pFunction[7] = QueryCollegeInfo;
    pFunction[8] = QueryTeamInfo;
```

```
    pFunction[9] = QueryProjectInfo;
    pFunction[10] = StatManRate;
    pFunction[11] = StatProjectType;
    pFunction[12] = StatTeam;
    pFunction[13] = StatProjectPer;
    pFunction[14] = AboutDorm;

    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：CreatList
    函数功能：从文件读取数据，并存放到链表中
    输入参数：无
    输出参数：phead 主链头指针的地址，用来返回创建的十字链表
    返回值： int 型，表示链表创建情况
                0，空链
                4，已加载院系基本信息
                12，已加载院系，团队基本信息
                28，已加载院系，团队，科研项目基本信息
```

```
*/
int CreatList(COLLEGE_NODE **phead)
{
    COLLEGE_NODE *hd = NULL, *pcol_node, temp1;
    TEAM_NODE *pteamnode, temp2;
    PROJECT_NODE *ppro_node, temp3;
    FILE *pFile;
    int find;
    int re = 0;

    if((pFile=fopen(gp_college_info_filename, "rb"))==NULL)
    {
        printf("院系基本信息文件打开失败！\n");
        return re;
    }
    printf("院系基本信息文件打开成功！\n");
    /*从文件读取院系基本信息*/

    while(fread(&temp1,sizeof(COLLEGE_NODE),1,pFile)==1)
    {
        pcol_node=(COLLEGE_NODE*)malloc(sizeof(COLLEGE_NODE));
        *pcol_node=temp1;
        pcol_node->tnext=NULL;
        pcol_node->next = hd;
        hd = pcol_node;
    }
    fclose(pFile);
    if(hd==NULL)
    {
        printf("院系基本信息文件加载失败！\n");
        return re;
    }
```

```
printf("院系基本信息文件加载成功！\n");

*phead=hd;
re += 4;
if((pFile=fopen(gp_team_info_filename, "rb"))==NULL)
{
    printf("团队基本信息文件打开失败！\n");
    return re;
}
printf("团队基本信息打开成功！\n");
re += 8;

/*将团队基本信息存入院系基本信息主链对应节点的支链中*/
while(fread(&temp2, sizeof(TEAM_NODE),1,pFile)==1)
{
    pteamnode=(TEAM_NODE *)malloc(sizeof(TEAM_NODE));
    *pteamnode=temp2;
    pteamnode->pnext=NULL;

    pcol_node=hd;
    while(pcol_node!=NULL&&strcmp(pcol_node->name,pteamnode->college)!=0)
    {
        pcol_node=pcol_node->next;
    }
    if(pcol_node!=NULL)
    {
        pteamnode->next=pcol_node->tnext;
        pcol_node->tnext=pteamnode;
    }
    else
    {
        free(pteamnode);
```

```
        }
    }
    fclose(pFile);


    if((pFile=fopen(gp_project_info_filename, "rb"))==NULL)
    {
        printf("科研项目基本信息文件打开失败！\n");
        return re;
    }
    printf("科研项目基本信息打开成功！\n");
    re+=16;

    /*将科研项目基本信息存入科研团队基本信息支链对应的节点的项目支链中*/
    while(fread(&temp3,sizeof(PROJECT_NODE), 1, pFile)==1)
    {
        ppro_node=(PROJECT_NODE *)malloc(sizeof(PROJECT_NODE));

        *ppro_node=temp3;
        pcol_node = hd;
        find = 0;
        while(pcol_node != NULL && find==0)
        {
            pteamnode=pcol_node->tnext;
            while(pteamnode!= NULL && find==0)
            {
                if(strcmp(ppro_node->team,pteamnode->name)==0)
                    {
                        find=1;
                        break;
                    }
                pteamnode=pteamnode->next;
```

```
            }
            pcol_node=pcol_node->next;
        }
        if(find)
        {
            ppro_node->next=pteamnode->pnext;
            pteamnode->pnext=ppro_node;
        }
        else
        {
            free(ppro_node);
        }
    }
    fclose(pFile);

    return re;
}

/*
    函数名称：SeekCollegeNode
    函数功能：按院系名称查找指定院系基本信息节点
    输入参数：主链头指针，所查找的院系名称
    输出参数：无
    返回值：查中返回节点地址，没查中返回 NULL
*/
COLLEGE_NODE *SeekCollegeNode(COLLEGE_NODE *hd, char *col_name)
{
    COLLEGE_NODE *pcol_node;
    int find=0;

    for(pcol_node=hd;pcol_node!=NULL;pcol_node=pcol_node->next)
    {
```

```
        if(strcmp(pcol_node->name,col_name)==0)
        {
            find=1;
            break;
        }
    }
    if(find) return pcol_node;
    else return NULL;
}

/*
    函数名称：SeekTeamNode
    函数功能：按团队名称查找指定科研团队基本信息节点
    输入参数：主链头指针，所查找的团队名称
    输出参数：无
    返回值：查中返回节点地址，没查中返回 NULL
*/
TEAM_NODE *SeekTeamNode(COLLEGE_NODE *hd, char *team_name)
{
    COLLEGE_NODE *pcol_node;
    TEAM_NODE *pteamnode;
    int find = 0;

    for(pcol_node=hd;pcol_node!=NULL;pcol_node=pcol_node->next)
    {
        for(pteamnode=pcol_node->tnext;pteamnode != NULL;pteamnode=pteamnode->next)
        {
            if(strcmp(pteamnode->name,team_name)==0)
            {
                find=1;
                break;
            }
```

```
            }
            if(find)
            {
                break;
            }
        }
        if(find)
            return pteamnode;
        else
            return NULL;
}

/*
    函数名称：SeekProNode
    函数功能：按照团队名称，项目编号查找科研项目信息节点
    输入参数：主链头指针，团队名称，项目编号
    输出参数：无
    返回值：查中返回节点地址，没查中返回 NULL
*/
PROJECT_NODE *SeekProNode(COLLEGE_NODE *hd, char *team_name, char *num)
{
    TEAM_NODE *pteamnode;
    PROJECT_NODE *ppro_node;
    int find = 0;
    pteamnode = SeekTeamNode(hd, team_name); /*查找团队信息节点*/

    if(pteamnode!=NULL)
    {
        ppro_node=pteamnode->pnext;
        while(ppro_node != NULL)
        {
            if(strcmp(ppro_node->num, num)==0)
```

```
                {
                    find=1;
                    break;
                }
                ppro_node=ppro_node->next;
            }
        }
        if(find)
        {
            return ppro_node;
        }
        else return NULL;
}

/*
    函数名称：InsertCollegeNode
    函数功能：在链表中插入一个院系基本信息节点
    输入参数：主连头指针，所要插入的节点指针
    输出参数：无
    返回值：BOOL 类型，TRUE 表示插入成功，FALSE 表示插入失败
*/
BOOL InsertCollegeNode(COLLEGE_NODE **hd, COLLEGE_NODE *pcol_node)
{
    if(*hd==NULL)
    {
        *hd=pcol_node;
        return TRUE;
    }
    else if(*hd!=NULL)
    {
        COLLEGE_NODE *p;
        p=*hd;
```

```
        while(p->next!=NULL)
        {
            p=p->next;
        }
        p->next=pcol_node;
        SaveSysData(gp_head);
        return TRUE;
    }
    else return FALSE;
}

/*
    函数名称：InsertTeamNode
    函数功能：在链表中插入一个团队基本信息节点
    输入参数：主连头指针，所要插入的节点指针
    输出参数：无
    返回值：BOOL 类型，TRUE 表示插入成功，FALSE 表示插入失败
*/
BOOL InsertTeamNode(COLLEGE_NODE **hd,TEAM_NODE *pteamnode)
{
    COLLEGE_NODE *pcol_node;

    /*在链表中找到对应的院系*/
    pcol_node = SeekCollegeNode(*hd, pteamnode->college);
    if(pcol_node!=NULL)
    {
        pteamnode->next = pcol_node->tnext;
        pcol_node->tnext = pteamnode;
        pteamnode->pnext=NULL;
        SaveSysData(gp_head);
        return TRUE;
    }
```

```
    else
        return FALSE;
}

/*
    函数名称：InsertProjectNode
    函数功能：在链表中插入一个科研项目信息节点
    输入参数：主连头指针，所要插入的节点指针
    输出参数：无
    返回值：BOOL 类型，TRUE 表示插入成功，FALSE 表示插入失败
*/
BOOL InsertProjectNode(COLLEGE_NODE **hd,PROJECT_NODE *ppro_node)
{
    TEAM_NODE *pteamnode;

    /*在链表中找到对应的科研团队*/
    pteamnode = SeekTeamNode(*hd, ppro_node->team);
    if(pteamnode!=NULL)
    {
        ppro_node->next = pteamnode->pnext;
        pteamnode->pnext = ppro_node;
        SaveSysData(gp_head);
        return TRUE;
    }
    else
        return FALSE;
}

/*
    函数名称：DelCollegeNode
    函数功能：删除院系基本信息节点
    输入参数：主链头指针，院系名称
```

```
    输出参数：无
    返回值：BOOL 类型，TRUE 表示删除成功，FALSE 表示删除失败
*/
BOOL DelCollegeNode(COLLEGE_NODE *hd, char *col_name)
{
    COLLEGE_NODE *pcol_node_prior;
    COLLEGE_NODE *pcol_node_current;
    BOOL bRet=FALSE;

    pcol_node_prior=NULL;
    pcol_node_current=hd->next;
    while(pcol_node_current!=NULL&&strcmp(pcol_node_current->name,col_name)!=0)
        {
            pcol_node_prior=pcol_node_current;
            pcol_node_current=pcol_node_current->next;
        }
        if(pcol_node_current!=NULL)
        {
            bRet=TRUE;
            if(pcol_node_prior==NULL)
            {
                hd->next=pcol_node_current->next;
            }
            else
            {
                pcol_node_prior->next=pcol_node_current->next;
            }
            free(pcol_node_current);
        }
    SaveSysData(gp_head);
    return bRet;
}
```

```
/*
    函数名称：DelTeamNode
    函数功能：删除科研团队基本信息节点
    输入参数：主链头指针,院系名称,团队名称
    输出参数：无
    返回值：BOOL 类型，TRUE 表示删除成功，FALSE 表示删除失败
*/
BOOL DelTeamNode(COLLEGE_NODE *hd, char *col_name, char *team_name)
{
    COLLEGE_NODE *pcol_node;
    TEAM_NODE *pteamnode_current;
    TEAM_NODE *pteamnode_prior;
    BOOL bRet=FALSE;
    pcol_node=SeekCollegeNode(hd,col_name);

    if(pcol_node!=NULL)
    {
        pteamnode_prior=NULL;
        pteamnode_current=pcol_node->tnext;

    while(pteamnode_current!=NULL&&strcmp(pteamnode_current->name,team_name)!=0)
        {
            pteamnode_prior=pteamnode_current;
            pteamnode_current=pteamnode_current->next;
        }
        if(pteamnode_current!=NULL)
        {
            bRet=TRUE;
            if(pteamnode_prior==NULL)
            {
```

```
                pcol_node->tnext=pteamnode_current->next;
            }
            else
            {
                pteamnode_prior->next=pteamnode_current->next;
            }
            free(pteamnode_current);
        }
    }
    SaveSysData(gp_head);
    return bRet;
}

/*
    函数名称：DelProjectNode
    函数功能：删除科研项目信息节点
    输入参数：主链头指针，团队名称，项目编号
    输出参数：无
    返回值：BOOL 类型，TRUE 表示删除成功，FALSE 表示删除失败
*/
BOOL DelProjectNode(COLLEGE_NODE *hd, char *team_name, char *pro_num)
{
    TEAM_NODE *pteamnode;
    PROJECT_NODE *ppro_node_prior;
    PROJECT_NODE *ppro_node_current;
    BOOL bRet=FALSE;
    pteamnode = SeekTeamNode(hd, team_name);
    if(pteamnode!=NULL)
    {
        ppro_node_prior=NULL;
        ppro_node_current=pteamnode->pnext;
        while(ppro_node_current!=NULL&&strcmp(ppro_node_current->num,pro_num)!=0)
```

```
        {
            ppro_node_prior=ppro_node_current;
            ppro_node_current=ppro_node_current->next;
        }

        if(ppro_node_current != NULL)
        {
            bRet=TRUE;
            if(ppro_node_prior==NULL)
            {
                pteamnode->pnext=ppro_node_current->next;
            }
            else
            {
                ppro_node_prior->next=ppro_node_current->next;
            }
            free(ppro_node_current);
        }
    }
    SaveSysData(gp_head);
    return bRet;
}

/*
    函数名称：ModifCollegeNode
    函数功能：修改指定的院系基本信息
    输入参数： 主链头指针，院系名称，存放修改内容节点的指针
    输出参数：无
    返回值：BOOL 类型，TRUE 表示修改成功，FALSE 表示修改失败
*/
BOOL ModifCollegeNode(COLLEGE_NODE *hd, char *col_name, COLLEGE_NODE
*pcol_node)
```

```
{
    COLLEGE_NODE *pcol_node_temp;
    COLLEGE_NODE *pcol_node_next;
    TEAM_NODE *pcol_node_tnext;

    pcol_node_temp=SeekCollegeNode(hd,col_name);
    if(pcol_node_temp!=NULL)
    {
        pcol_node_tnext=pcol_node_temp->tnext;
        pcol_node_next=pcol_node_temp->next;
        *pcol_node_temp=*pcol_node;
        pcol_node_temp->next=pcol_node_next;
        pcol_node_temp->tnext=pcol_node_tnext;
        SaveSysData(gp_head);
        return TRUE;
    }
    else return FALSE;
}


/*
    函数名称：ModifTeamNode
    函数功能：修改指定的科研团队信息
    输入参数：主链头指针，科研团队名称，存放修改内容节点的指针
    输出参数：无
    返回值：BOOL 类型，TRUE 表示修改成功，FALSE 表示修改失败
*/
BOOL ModifTeamNode(COLLEGE_NODE *hd, char *team_name, TEAM_NODE *pteamnode)
{
    TEAM_NODE *pteamnode_temp;
    TEAM_NODE *pteamnode_next;
    PROJECT_NODE *pteamnode_pnext;
```

```
    pteamnode_temp=SeekTeamNode(hd, team_name);
    if(pteamnode_temp!=NULL)
    {
        pteamnode_next=pteamnode_temp->next;
        pteamnode_pnext=pteamnode_temp->pnext;
        *pteamnode_temp=*pteamnode;
        pteamnode_temp->next=pteamnode_next;
        pteamnode_temp->pnext=pteamnode_pnext;
        SaveSysData(gp_head);
        return TRUE;
    }
    else return FALSE;
}

/*
    函数名称：ModifProjectNode
    函数功能：修改指定的科研项目信息
    输入参数：主链头指针，科研项目编所属团队名称，科研项目编号，指向存放修改内容节点的指针
    输出参数：无
    返回值：BOOL 类型，TRUE 表示修改成功，FALSE 表示修改失败
*/
BOOL ModifProjectNode(COLLEGE_NODE *hd, char *team_name, char *num, PROJECT_NODE *ppro_node)
{
    PROJECT_NODE *ppro_node_temp;
    PROJECT_NODE *ppro_node_next;
    ppro_node_temp=SeekProNode(hd,team_name,num);
    if(ppro_node_temp!=NULL)
    {
        ppro_node_next=ppro_node_temp->next;
```

```
            *ppro_node_temp=*ppro_node;
            ppro_node_temp->next=ppro_node_next;
            SaveSysData(gp_head);
            return TRUE;
        }
        else    return FALSE;
}

/*
        函数名称：LoadDate
        函数功能：将代码表和三类基础数据从数据文件载入到内存区和链表中
        输入参数：无
        输出参数：无
        返回值：BOOL 型，功能函数除了 ExitSys 的返回值可以为 FALSE 外，其他必须为 TRUE
*/
BOOL LoadDate()
{
        int Re = 0;
        if(gp_type_code!=NULL)
        {
            free(gp_type_code);
        }
        gul_type_code_len=LoadCode(gp_type_code_filename, &gp_type_code);
        if(gul_type_code_len<3)
        {
            printf("项目类别代码表加载失败！\n");
            gc_sys_state &= 0xfe;
        }
        else
        {
            printf("项目类别代码表加载成功！\n");
            gc_sys_state |= 1;
```

```
    }

    Re=CreatList(&gp_head);
    gc_sys_state |= Re;
    gc_sys_state &=~ (4+8+16-Re);
    if(gc_sys_state<(1|4|8|16))
    {
        printf("\n 系统基础数据不完整！\n");
        printf("\n 按任意键继续...\n");
        getch();
    }
    else
    {
        printf("\n 系统基础数据录入完毕。\n");
        printf("按任意键继续...\n");
        getch();
    }

    return TRUE;
}

/*
    函数名称：LoadCode
    函数功能：将代码表从数据文件载入到内存缓冲区，并进行排序和去除空格
    输入参数：FileName 存放代码表的数据文件名
    输出参数：指向内存缓冲区的指针地址
    返回值：存放代码表的内存缓冲区的大小
*/
int LoadCode(char *filename, char **pbuffer)
{
    char *pTemp, *pStr1, *pStr2;
    int handle;
```

```
    int BufferLen, len, loc1, loc2, i;
    long filelen;

    if ((handle = open(filename, O_RDONLY | O_TEXT)) == -1) /*如果以只读方式打开失败
*/
    {
        handle = open(filename, O_CREAT | O_TEXT, S_IREAD); /*以创建方式打开*/
    }
    filelen = filelength(handle);          /*数据文件的长度*/
    pTemp = (char *)calloc(filelen + 1, sizeof(char)); /*申请同样大小的动态存储区*/
    BufferLen = read(handle, pTemp, filelen); /*将数据文件的内容全部读入到内存*/
    close(handle);

    *(pTemp + BufferLen) = '\0'; /*在动态存储区尾存一个空字符，作为字符串结束标志*/
    BufferLen++;

    for (i=0; i<BufferLen; i++) /*将动态存储区中的所有换行符替换成空字符*/
    {
        if (*(pTemp + i) == '\n')
        {
            *(pTemp + i) = '\0';
        }
    }

    /*再申请一块同样大小的动态存储区，用于存放排序后的代码串*/
    *pbuffer = (char *)calloc(BufferLen, sizeof(char));
    loc2 = 0;
    pStr1 = pTemp;
    len = strlen(pStr1);

    while (BufferLen > len + 1) /*选择法排序*/
    {
```

```
    loc1 = len + 1;
    while (BufferLen > loc1) /*每趟找到序列中最小代码串，首地址存入 pStr1*/
    {
        pStr2 = pTemp + loc1;
        if (strcmp(pStr1, pStr2) > 0)
        {
            pStr1 = pStr2;
        }
        loc1 += strlen(pStr2) + 1;
    }
    len = strlen(pStr1);  /*这一趟所找到的最小代码串长度*/

    /*如果不是空串，则进行复制，loc2 是下一个最小代码串存放地址的偏移量*/
    if (len > 0)
    {
        strcpy(*pbuffer + loc2, pStr1);
        loc2 += len + 1;  /*已复制的代码串所占存储空间大小*/
    }

    /*将最小代码串从序列中删除掉*/
    for(i=0; i<BufferLen-(pStr1-pTemp)-(len+1); i++)
    {
        *(pStr1 + i) = *(pStr1 + i + len + 1);
    }

    BufferLen -= len + 1; /*下一趟排序所处理序列的长度*/
    pStr1 = pTemp;  /*假定序列的第一个代码串为最小代码串*/
    len = strlen(pStr1);
} /*序列中只剩下一个代码串时，排序结束*/

/*复制最后这个代码串*/
len = strlen(pStr1);
```

```
    strcpy(*pbuffer + loc2, pStr1);

    /*修改动态存储区大小，使其正好放下排序后代码串*/
    loc2 += len + 1;
    *pbuffer = (char *)realloc(*pbuffer, loc2);
    free(pTemp);   /*释放最先申请的动态存储区*/

    return loc2;   /*返回存放代码串的内存缓冲区实际大小*/
}

/*
 * 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: 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;
}

/*
    函数名称：SetHotPoint
    函数功能：设置热区
    输入参数：焦点热区存放地址，焦点热区编号
    输出参数：无
    返回值：无
*/
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：SaveSysData
        函数功能：保存系统代码表和三类基础数据
        输入参数：主链头指针
        输出参数：无
        返回值：BOOL 类型，总是为 TRUE
    */
```

```
BOOL SaveSysData(COLLEGE_NODE *hd)
{
    COLLEGE_NODE *pcol_node;
    TEAM_NODE *pteamnode;
    PROJECT_NODE *ppro_node;
    FILE *pfout;
    int handle;

    if((handle=open(gp_type_code_filename,O_WRONLY|O_TEXT))==-1)
    {
        handle=open(gp_type_code_filename,O_CREAT|O_TEXT,S_IWRITE);
    }
    write(handle, gp_type_code, gul_type_code_len);
    close(handle);

    pfout=fopen(gp_college_info_filename, "wb");
    for(pcol_node=hd;pcol_node!=NULL;pcol_node=pcol_node->next)
    {
        fwrite(pcol_node,sizeof(COLLEGE_NODE),1,pfout);
    }
    fclose(pfout);

    pfout=fopen(gp_team_info_filename, "wb");
    for(pcol_node=hd;pcol_node!=NULL;pcol_node=pcol_node->next)
    {
        pteamnode=pcol_node->tnext;
        while(pteamnode!=NULL)
        {
            fwrite(pteamnode,sizeof(TEAM_NODE),1,pfout);
            pteamnode=pteamnode->next;
        }
    }
```

```
    fclose(pfout);

    pfout=fopen(gp_project_info_filename,"wb");
    for(pcol_node=hd;pcol_node!=NULL;pcol_node=pcol_node->next)
    {
        pteamnode=pcol_node->tnext;
        while(pteamnode!=NULL)
        {
            ppro_node=pteamnode->pnext;
            while(ppro_node!=NULL)
            {
                fwrite(ppro_node,sizeof(PROJECT_NODE),1, pfout);
                ppro_node=ppro_node->next;
            }
            pteamnode=pteamnode->next;
        }
    }
    fclose(pfout);

    return TRUE;
}

/*
    函数名称：MarchString
    函数功能：判断给定的字符串是否匹配
    输入参数：给定字符串，条件字符串
    输出参数：无
    返回值：BOOL 类型，匹配成功返回 TRUE，匹配失败返回 FALSE
*/
BOOL MarchString(char *string_item, char *cond)
{
    char *sub_string_pos;
```

```
    BOOL bRet=FALSE;
    sub_string_pos=strstr(string_item,cond);
    if(sub_string_pos!=NULL) bRet=TRUE;
    return bRet;
}

/*
    函数名称：SeekColNodeMan
    函数功能：按院系负责人查找满足条件的院系
    输入参数：主链头指针，院系负责人
    输出参数：无
    返回值：若找到返回结果链表的头指针，未找到返回 NULL
*/
COLLEGE_NODE *SeekColNodeMan(COLLEGE_NODE *hd, char* dutyman)
{
    COLLEGE_NODE *pcol_node;
    COLLEGE_NODE *pcol_node_ret=NULL;
    COLLEGE_NODE *pcol_node_temp;
    int find=0;

    for(pcol_node=hd;pcol_node!=NULL;pcol_node=pcol_node->next)
    {
        if(strcmp(pcol_node->dutyman,dutyman)==0)
        {
            pcol_node_temp=(COLLEGE_NODE *)malloc(sizeof(COLLEGE_NODE));
            *pcol_node_temp=*pcol_node;
            pcol_node_temp->next=pcol_node_ret;
            pcol_node_ret=pcol_node_temp;
            find=1;
        }
    }
```

```
    if(find) return pcol_node_ret;
    else return NULL;
}

/*
    函数名称：SeekCollegeNodeNameM
    函数功能：按照院系名称的全部或部分查找符合条件的院系信息
    输入参数：主链头指针，关键字符串
    输出参数：无
    返回值：若找到返回结果链表头指针，未找到返回 NULL
*/
COLLEGE_NODE *SeekCollegeNodeNameM(COLLEGE_NODE *hd, char *namem)
{
    COLLEGE_NODE *pcol_node;
    COLLEGE_NODE *pcol_node_ret=NULL;
    COLLEGE_NODE *pcol_node_temp;
    int find=0;

    for(pcol_node=hd;pcol_node!=NULL;pcol_node=pcol_node->next)
    {
        if(MarchString(pcol_node->name,namem)==TRUE)
        {
            pcol_node_temp=(COLLEGE_NODE *)malloc(sizeof(COLLEGE_NODE));
            *pcol_node_temp=*pcol_node;
            pcol_node_temp->next=pcol_node_ret;
            pcol_node_ret=pcol_node_temp;
            find=1;
        }
    }
    if(find) return pcol_node_ret;
    else return NULL;
}
```

```
/*
    函数名称：SeekTeamNodeNameM
    函数功能：按照团队名称的全部或部分查找符合条件的团队信息
    输入参数：主链头指针，关键字符串
    输出参数：无
    返回值：若找到返回结果链表头指针，未找到返回 NULL
*/
TEAM_NODE *SeekTeamNodeNameM(COLLEGE_NODE *hd, char *namem)
{
    COLLEGE_NODE *pcol_node;
    TEAM_NODE *pteamnode;
    TEAM_NODE *pteamnode_ret=NULL;
    TEAM_NODE *pteamnode_temp;
    int find=0;

    for(pcol_node=hd;pcol_node!=NULL;pcol_node=pcol_node->next)
    {
        for(pteamnode=pcol_node->tnext;pteamnode!=NULL;pteamnode=pteamnode->next)
        {
            if(MarchString(pteamnode->name,namem)==TRUE)
            {
                pteamnode_temp=(TEAM_NODE *)malloc(sizeof(TEAM_NODE));
                *pteamnode_temp=*pteamnode;
                pteamnode_temp->next=pteamnode_ret;
                pteamnode_ret=pteamnode_temp;
                find=1;
            }
        }
    }
    if(find) return pteamnode_ret;
    else return NULL;
```

```
}

/*
    函数名称：SeekTeamNodeTeacher
    函数功能：按教师人数查找符合条件的科研团队
    输入参数：主链头指针，教师人数
    输出参数：无
    返回值： 若找到返回结果链表头指针，未找到返回 NULL
*/
TEAM_NODE *SeekTeamNodeTeacher(COLLEGE_NODE *hd, int juge, int num_tc)
{
    COLLEGE_NODE *pcol_node;
    TEAM_NODE *pteamnode;
    TEAM_NODE *pteamnode_ret=NULL;
    TEAM_NODE *pteamnode_temp;
    int find=0;
    switch(juge)
    {
        case 2:
        {
            for(pcol_node=hd;pcol_node!=NULL;pcol_node=pcol_node->next)
            {
                pteamnode=pcol_node->tnext;
                while(pteamnode!=NULL)
                {
                    if(pteamnode->num_tc==num_tc)
                    {
                        pteamnode_temp=(TEAM_NODE *)malloc(sizeof(TEAM_NODE));
                        *pteamnode_temp=*pteamnode;
                        pteamnode_temp->next=pteamnode_ret;
                        pteamnode_ret=pteamnode_temp;
                        find=1;
```

```
                }
                pteamnode=pteamnode->next;
            }
        }
        if(find)
            return pteamnode_ret;
        else
            return NULL;
    }
    case 3:
    {
        for(pcol_node=hd;pcol_node!=NULL;pcol_node=pcol_node->next)
        {
            pteamnode=pcol_node->tnext;
            while(pteamnode!=NULL)
            {
                if(pteamnode->num_tc<num_tc)
                {
                    pteamnode_temp=(TEAM_NODE *)malloc(sizeof(TEAM_NODE));
                    *pteamnode_temp=*pteamnode;
                    pteamnode_temp->next=pteamnode_ret;
                    pteamnode_ret=pteamnode_temp;
                    find=1;
                }
                pteamnode=pteamnode->next;
            }
        }
        if(find)
            return pteamnode_ret;
        else
            return NULL;
    }
```

```
        case 1:
        {
            for(pcol_node=hd;pcol_node!=NULL;pcol_node=pcol_node->next)
            {
                pteamnode=pcol_node->tnext;
                while(pteamnode!=NULL)
                {
                    if(pteamnode->num_tc>num_tc)
                    {
                        pteamnode_temp=(TEAM_NODE *)malloc(sizeof(TEAM_NODE));
                        *pteamnode_temp=*pteamnode;
                        pteamnode_temp->next=pteamnode_ret;
                        pteamnode_ret=pteamnode_temp;
                        find=1;
                    }
                    pteamnode=pteamnode->next;
                }
            }
            if(find)
                return pteamnode_ret;
            else
                return NULL;
        }
        default:
            return NULL;
    }
}

/*
    函数名称：SeekProjectNodeNum
    函数功能：按照项目编号查找符合条件的科研项目信息节点
    输入参数：主链头指针，项目编号
```

```
    输出参数：无
    返回值：若找到返回结果链表头指针，未找到返回 NULL
*/
PROJECT_NODE *SeekProjectNodeNum(COLLEGE_NODE *hd, char *p_num)
{
    COLLEGE_NODE *pcol_node;
    TEAM_NODE *pteamnode;
    PROJECT_NODE *ppro_node;
    PROJECT_NODE *ppro_node_ret=NULL;
    PROJECT_NODE *ppro_node_temp;
    int find=0;

    for(pcol_node=hd;pcol_node!=NULL;pcol_node=pcol_node->next)
    {
        for(pteamnode=pcol_node->tnext;pteamnode!=NULL;pteamnode=pteamnode->next)
        {
            ppro_node=pteamnode->pnext;
            while(ppro_node!=NULL)
            {
                if(strcmp(ppro_node->num, p_num)==0)
                {

    ppro_node_temp=(PROJECT_NODE*)malloc(sizeof(PROJECT_NODE));
                    *ppro_node_temp=*ppro_node;
                    ppro_node_temp->next=ppro_node_ret;
                    ppro_node_ret=ppro_node_temp;
                    find=1;
                }
                ppro_node=ppro_node->next;
            }
        }
    }
```

```
    if(find) return ppro_node_ret;
    else return NULL;
}


/*
    函数名称：StatPeopleRate
    函数功能：统计院系研究生，教师总数及人数比并排序
    输入参数：主链头指针
    输出参数：无
    返回值：指向统计人数结果链表的指针
*/
PEOPLES_NODE *StatPeopleRate(COLLEGE_NODE *hd)
{
    COLLEGE_NODE *pcol_node;
    TEAM_NODE *pteamnode;
    PEOPLES_NODE *ppeo_node;
    PEOPLES_NODE *ppeo_node_hd=NULL;

    for(pcol_node=hd;pcol_node!=NULL;pcol_node=pcol_node->next)
    {
        ppeo_node=(PEOPLES_NODE *)malloc(sizeof(PEOPLES_NODE));
        strcpy(ppeo_node->col_name,pcol_node->name);
        ppeo_node->student=0;
        ppeo_node->teacher=0;
        for(pteamnode=pcol_node->tnext;pteamnode!=NULL;pteamnode=pteamnode->next)
        {
            ppeo_node->student+=pteamnode->num_stu;
            ppeo_node->teacher+=pteamnode->num_tc;
        }
        ppeo_node->scale=(float)ppeo_node->student/(float)ppeo_node->teacher;
```

```
            ppeo_node->next=ppeo_node_hd;
            ppeo_node_hd=ppeo_node;
        }
        SortPeopleInfo(ppeo_node_hd);

        return ppeo_node_hd;
}



/*
        函数名称：SortPeopleInfo
        函数功能：对人数比信息排序
        输入参数：人数比信息链链头
        输出参数：排序后链表链头
        返回值：无
*/
void SortPeopleInfo(PEOPLES_NODE *ppeo_node_hd)
{
        PEOPLES_NODE *ppeo_node_1;
        PEOPLES_NODE *ppeo_node_2;
        ppeo_node_1=ppeo_node_hd;
        if(ppeo_node_1==NULL)
        {
            return ;
        }

        char col_name[20];
        int num_stu, num_tc;
        float scale;
        for(;ppeo_node_1->next!=NULL;ppeo_node_1=ppeo_node_1->next)
        {
```

```
    for(ppeo_node_2=ppeo_node_1->next;ppeo_node_2!=NULL;ppeo_node_2=ppeo_node_2->
next)
        {
            if((ppeo_node_1->scale)<(ppeo_node_2->scale))
            {
                strcpy(col_name,ppeo_node_1->col_name);
                strcpy(ppeo_node_1->col_name,ppeo_node_2->col_name);
                strcpy(ppeo_node_2->col_name,col_name);
                num_stu=ppeo_node_1->student;
ppeo_node_1->student=ppeo_node_2->student;   ppeo_node_2->student=num_stu;
                num_tc=ppeo_node_1->teacher;
ppeo_node_1->teacher=ppeo_node_2->teacher;   ppeo_node_2->teacher=num_tc;
                scale=ppeo_node_1->scale;        ppeo_node_1->scale=ppeo_node_2->scale;
ppeo_node_2->scale=scale;
            }
        }
    }

    return ;
}

/*
    函数名称：StatCollegeProjectNum
    函数功能：统计各类科研项目总数并排序
    输入参数：主链头指针
    输出参数：无
    返回值：指向结果链表的头指针
*/
PROJECT_NUM_NODE *StatCollegeProjectNum(COLLEGE_NODE *hd)
{
    COLLEGE_NODE *pcol_node;
    TEAM_NODE *pteamnode;
```

```
    PROJECT_NODE *ppro_node;
    PROJECT_NUM_NODE *ppro_num_node;
    PROJECT_NUM_NODE *ppro_num_node_hd=NULL;

    for(pcol_node=hd;pcol_node!=NULL;pcol_node=pcol_node->next)
    {
        ppro_num_node=(PROJECT_NUM_NODE
*)malloc(sizeof(PROJECT_NUM_NODE));
        strcpy(ppro_num_node->col_name,pcol_node->name);
        ppro_num_node->total=0;
        ppro_num_node->p973=0;
        ppro_num_node->p863=0;
        ppro_num_node->money=0;
        for(pteamnode=pcol_node->tnext;pteamnode!=NULL;pteamnode=pteamnode->next)
        {

    for(ppro_node=pteamnode->pnext;ppro_node!=NULL;ppro_node=ppro_node->next)
            {
                ppro_num_node->total++;
                ppro_num_node->money+=ppro_node->money;
                if(ppro_node->type=='1') ppro_num_node->p973++;
                else if(ppro_node->type=='3') ppro_num_node->p863++;
            }
        }
        ppro_num_node->next=ppro_num_node_hd;
        ppro_num_node_hd=ppro_num_node;
    }

    SortColProInfo(ppro_num_node_hd);

    return ppro_num_node_hd;
}
```

```
/*
    函数名称：SortColProInfo
    函数功能：对科研项目总数排序
    输入参数：科研项目数目信息链链头
    输出参数：排序后链表链头
    返回值：无
*/
void SortColProInfo(PROJECT_NUM_NODE *ppro_num_node_hd)
{
    PROJECT_NUM_NODE *ppro_num_node_1;
    PROJECT_NUM_NODE *ppro_num_node_2;
    ppro_num_node_1=ppro_num_node_hd;
    if(ppro_num_node_1==NULL)
    {
        return ;
    }

    char col_name[20];
    int total, p973, p863;
    float money;
    for(;ppro_num_node_1->next!=NULL;ppro_num_node_1=ppro_num_node_1->next)
    {

    for(ppro_num_node_2=ppro_num_node_1->next;ppro_num_node_2!=NULL;ppro_num_no
de_2=ppro_num_node_2->next)
        {
            if((ppro_num_node_1->total)<(ppro_num_node_2->total))
            {
                strcpy(col_name,ppro_num_node_1->col_name);
                strcpy(ppro_num_node_1->col_name,ppro_num_node_2->col_name);
                strcpy(ppro_num_node_2->col_name,col_name);
```

```
                    total=ppro_num_node_1->total;
ppro_num_node_1->total=ppro_num_node_2->total;   ppro_num_node_2->total=total;
                    p973=ppro_num_node_1->p973;
ppro_num_node_1->p973=ppro_num_node_2->p973;   ppro_num_node_2->p973=p973;
                    p863=ppro_num_node_1->p863;
ppro_num_node_1->p863=ppro_num_node_2->p863;   ppro_num_node_2->p863=p863;
                    money=ppro_num_node_1->money;
ppro_num_node_1->money=ppro_num_node_2->money;   ppro_num_node_2->money=money;
                }
            }
        }

    return ;
}

/*
    函数名称：StatTeamNasiTen
    函数功能：统计团队自然科学基金项目总数排名
    输入参数：主链头指针
    输出参数：无
    返回值：指向结果链表的头指针
*/
TEAM_NASI_NODE *StatTeamNasiTen(COLLEGE_NODE *hd)
{
    COLLEGE_NODE *pcol_node;
    TEAM_NODE *pteamnode;
    PROJECT_NODE *ppro_node;
    TEAM_NASI_NODE *pteam_nasi_node;
    TEAM_NASI_NODE *pteam_nasi_node_hd=NULL;

    for(pcol_node=hd;pcol_node!=NULL;pcol_node=pcol_node->next)
    {
```

```
        pteamnode=pcol_node->tnext;
        for(;pteamnode!=NULL;pteamnode=pteamnode->next)
        {
            pteam_nasi_node=(TEAM_NASI_NODE *)malloc(sizeof(TEAM_NASI_NODE));
            strcpy(pteam_nasi_node->team_name,pteamnode->name);
            pteam_nasi_node->money=0;
            pteam_nasi_node->num_nasi=0;

    for(ppro_node=pteamnode->pnext;ppro_node!=NULL;ppro_node=ppro_node->next)
            {
                pteam_nasi_node->money+=ppro_node->money;
                if(ppro_node->type=='2')
                    pteam_nasi_node->num_nasi++;
            }
            pteam_nasi_node->next=pteam_nasi_node_hd;
            pteam_nasi_node_hd=pteam_nasi_node;
        }
    }

    SortTeamNasiInfo(pteam_nasi_node_hd);

    return pteam_nasi_node_hd;
}

/*
    函数名称：SortTeamNasiInfo
    函数功能：对自然科学基金项目信息链排序
    输入参数：自然科学基金项目信息链链头
    输出参数：排序后链表头
    返回值： 无
*/
void SortTeamNasiInfo(TEAM_NASI_NODE *pteam_nasi_node_hd)
```

```
{
    TEAM_NASI_NODE *team_nasi_node_1;
    TEAM_NASI_NODE *team_nasi_node_2;
    team_nasi_node_1=pteam_nasi_node_hd;
    if(team_nasi_node_1==NULL)
    {
        return ;
    }

    char team_name[20];
    int total;
    float money;
    for(;team_nasi_node_1->next!=NULL;team_nasi_node_1=team_nasi_node_1->next)
    {

    for(team_nasi_node_2=team_nasi_node_1->next;team_nasi_node_2!=NULL;team_nasi_node_2=team_nasi_node_2->next)
        {
            if((team_nasi_node_1->num_nasi)<(team_nasi_node_2->num_nasi))
            {
                strcpy(team_name,team_nasi_node_1->team_name);
                strcpy(team_nasi_node_1->team_name,team_nasi_node_2->team_name);
                strcpy(team_nasi_node_2->team_name,team_name);
                total=team_nasi_node_1->num_nasi;
team_nasi_node_1->num_nasi=team_nasi_node_2->num_nasi;
team_nasi_node_2->num_nasi=total;
                money=team_nasi_node_1->money;
team_nasi_node_1->money=team_nasi_node_2->money;    team_nasi_node_2->money=money;
            }
        }
    }
```

```
    return ;
}

/*
    函数名称：StatProTeaScale
    函数功能：统计团队研究项目和教师人数比并排序
    输入参数：主链头指针
    输出参数：无
    返回值：指向结果链表头指针
*/
TEAM_PROTEA_NODE *StatProTeaScale(COLLEGE_NODE *hd)
{
    COLLEGE_NODE *pcol_node;
    TEAM_NODE *pteamnode;
    PROJECT_NODE *ppro_node;
    TEAM_PROTEA_NODE *protea_node;
    TEAM_PROTEA_NODE *protea_node_hd=NULL;

    for(pcol_node=hd;pcol_node!=NULL;pcol_node=pcol_node->next)
    {
        for(pteamnode=pcol_node->tnext;pteamnode!=NULL;pteamnode=pteamnode->next)
        {
            protea_node=(TEAM_PROTEA_NODE
*)malloc(sizeof(TEAM_PROTEA_NODE));
            strcpy(protea_node->team_name, pteamnode->name);
            protea_node->num_pro=0;
            protea_node->num_tc=pteamnode->num_tc;

    for(ppro_node=pteamnode->pnext;ppro_node!=NULL;ppro_node=ppro_node->next)
            {
                protea_node->num_pro++;
            }
```

```
            protea_node->scale=(float)protea_node->num_pro/(float)protea_node->num_tc;

            protea_node->next=protea_node_hd;
            protea_node_hd=protea_node;
        }
    }
    SortProTeaInfo(protea_node_hd);

    return protea_node_hd;
}

/*
  函数名称：SortProTeaInfo
  函数功能：根据比例对团队科研项目与教师人数比信息排序
  输入参数：团队科研项目教师人数比链头指针
  输出参数：排序后的链表头指针
  返回值：无
*/
void SortProTeaInfo(TEAM_PROTEA_NODE *protea_node_hd)
{
    TEAM_PROTEA_NODE *protea_node_1;
    TEAM_PROTEA_NODE *protea_node_2;
    protea_node_1=protea_node_hd;
    if(protea_node_1==NULL)
    {
        return ;
    }

    char team_name[30];
    int num_pro, num_tc;
    float scale;
    for(;protea_node_1->next!=NULL;protea_node_1=protea_node_1->next)
```

```
    {

    for(protea_node_2=protea_node_1->next;protea_node_2!=NULL;protea_node_2=protea_node_2->next)
        {
            if((protea_node_1->scale)<(protea_node_2->scale))
            {
                strcpy(team_name,protea_node_1->team_name);
                strcpy(protea_node_1->team_name,protea_node_2->team_name);
                strcpy(protea_node_2->team_name,team_name);
                num_pro=protea_node_1->num_pro;
protea_node_1->num_pro=protea_node_2->num_pro;  protea_node_2->num_pro=num_pro;
                num_tc=protea_node_1->num_tc;
protea_node_1->num_tc=protea_node_2->num_tc;  protea_node_2->num_tc=num_tc;
                scale=protea_node_1->scale;  protea_node_1->scale=protea_node_2->scale;
protea_node_2->scale=scale;
            }
        }
    }

    return ;
}

BOOL SaveData(void)
{
    BOOL bRet=TRUE;

    SaveSysData(gp_head);
    printf("保存成功！\n");
    printf("按任意键返回。\n");
    getch();
    InitInterface();
```

```
    return bRet;
}

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;
}
BOOL TypeCode(void)
{
    BOOL bRet = TRUE;
    printf("项目类别代码：\n");
    printf("1       973 计划项目\n");
    printf("2       国家自然科学基金项目\n");
```

```
    printf("3       863 计划项目\n");
    printf("4       国际合作项目\n");
    printf("5       横向项目\n");
    printf("按任意键返回..\n");
    getch();
    InitInterface();

    return bRet;
}

/*维护院系信息*/
BOOL MaintainCollegeInfo(void)
{
    BOOL bRet = TRUE;
    printf("插入院系基本信息按 1\n");
    printf("删除院系项目基本信息按 2\n");
    printf("修改院系项目基本信息按 3\n");
    int juge;

    scanf("%d", &juge);
    InitInterface();
    if(juge==1)
    {
        COLLEGE_NODE *pcol_node;
        pcol_node=(COLLEGE_NODE *)malloc(sizeof(COLLEGE_NODE));
        printf("请输入院系名称：");
        scanf("%s", pcol_node->name);
        printf("请输入院系负责人：");
        scanf("%s", pcol_node->dutyman);
        printf("请输入联系电话：");
        scanf("%s", pcol_node->tel);
        pcol_node->next=NULL;
```

```
        pcol_node->tnext=NULL;
        BOOL juge0=InsertCollegeNode(&gp_head, pcol_node);
        if(juge0==TRUE)
            printf("插入成功！\n");
        else
            printf("插入失败！\n");
        printf("按任意键返回..\n");
        getch();
        InitInterface();
    }
    else if(juge==2)
    {
        char col_name[20];
        printf("请输入需要删除的院系名称：");
        scanf("%s", col_name);
        BOOL juge1=DelCollegeNode(gp_head, col_name);
        if(juge1==TRUE)
            printf("删除成功！\n");
        else
            printf("删除失败！\n");
        printf("按任意键返回..\n");
        getch();
        InitInterface();
    }
    else if(juge==3)
    {
        COLLEGE_NODE *col_node0;
        char col_name0[20];
        printf("请输入需要修改的院系名称：");
        scanf("%s", col_name0);
        col_node0=(COLLEGE_NODE *)malloc(sizeof(COLLEGE_NODE));
        strcpy(col_node0->name,col_name0);
```

```
            printf("请输入修改后的院系负责人：");
            scanf("%s", col_node0->dutyman);
            printf("请输入修改后的院系联系电话：");
            scanf("%s", col_node0->tel);

            BOOL juge2=ModifCollegeNode(gp_head, col_name0, col_node0);
            if(juge2==TRUE)
                printf("修改成功！\n");
            else
                printf("修改失败！\n");
            printf("按任意键返回..\n");
            getch();
            InitInterface();
        }

        return bRet;
}
/*维护团队信息*/
BOOL MaintainTeamInfo(void)
{
        BOOL bRet = TRUE;
        printf("插入团队基本信息按 1\n");
        printf("删除团队项目基本信息按 2\n");
        printf("修改团队项目基本信息按 3\n");
        int juge;

        scanf("%d", &juge);
        InitInterface();
        if(juge==1)
        {
            TEAM_NODE *pteamnode;
            pteamnode=(TEAM_NODE *)malloc(sizeof(TEAM_NODE));
```

```
        printf("请输入团队名称：");
        scanf("%s", pteamnode->name);
        printf("请输入负责人：");
        scanf("%s", pteamnode->dutyman);
        printf("请输入所属院系：");
        scanf("%s", pteamnode->college);
        printf("请输入研究生人数：");
        scanf("%d", &pteamnode->num_stu);
        printf("请输入教师人数：");
        scanf("%d", &pteamnode->num_tc);
        BOOL juge1=InsertTeamNode(&gp_head, pteamnode);
        if(juge1==TRUE)
            printf("插入成功！\n");
        else
            printf("未找到所属院系，插入失败！\n");
        printf("按任意键返回..\n");
        getch();
        InitInterface();
    }
    else if(juge==2)
    {
        char col_name[20];
        char team_name[30];
        printf("请输入需要删除的团队所属院系名称：");
        scanf("%s", col_name);
        printf("请输入需要删除团队名称：");
        scanf("%s", team_name);
        BOOL juge0=DelTeamNode(gp_head, col_name, team_name);
        if(juge0==TRUE)
            printf("删除成功！\n");
        else
            printf("删除失败！\n");
```

```
        printf("按任意键返回..\n");
        getch();
        InitInterface();
    }
    else if(juge==3)
    {
        TEAM_NODE *pteamnode0;
        char team_name0[30];
        printf("请输入需要修改的团队名称：");
        scanf("%s", team_name0);
        pteamnode0=(TEAM_NODE *)malloc(sizeof(TEAM_NODE));
        strcpy(pteamnode0->name,team_name0);
        printf("请输入修改后的团队负责人：");
        scanf("%s", pteamnode0->dutyman);
        printf("请输入修改后的团队所属院系：");
        scanf("%s", pteamnode0->college);
        printf("请输入修改后的团队研究生人数：");
        scanf("%d", &pteamnode0->num_stu);
        printf("请输入修改后的团队教师人数：");
        scanf("%d", &pteamnode0->num_tc);

        BOOL juge2=ModifTeamNode(gp_head, team_name0, pteamnode0);
        if(juge2==TRUE)
            printf("修改成功！\n");
        else
            printf("修改失败！\n");
        printf("按任意键返回..\n");
        getch();
        InitInterface();
    }
```

```
    return bRet;
}
BOOL MaintainProjectInfo(void)
{
    BOOL bRet = TRUE;
    printf("插入科研项目基本信息按 1\n");
    printf("删除科研项目基本信息按 2\n");
    printf("修改科研项目基本信息按 3\n");
    int juge;

    scanf("%d", &juge);
    InitInterface();
    if(juge==1)
    {
        PROJECT_NODE *ppro_node;
        ppro_node=(PROJECT_NODE *)malloc(sizeof(PROJECT_NODE));
        printf("请输入项目编号:");
        scanf("%s", ppro_node->num);
        char in=getchar();
        printf("请输入项目类型:");
        scanf("%c", &ppro_node->type);
        printf("请输入起始时间：");
        scanf("%s", ppro_node->start);
        printf("请输入项目经费：");
        scanf("%f", &ppro_node->money);
        printf("请输入负责人：");
        scanf("%s", ppro_node->dutyman);
        printf("请输入所属团队：");
        scanf("%s", ppro_node->team);
        BOOL juge0=InsertProjectNode(&gp_head, ppro_node);
        if(juge0==TRUE)
            printf("插入成功！\n");
```

```
        else
            printf("未找到所属团队，插入失败！\n");
        printf("按任意键返回..\n");
        getch();
        InitInterface();
    }
    if(juge==2)
    {
        char team_name[30];
        char pro_num[15];
        printf("请输入需要删除的项目所属团队名称：");
        scanf("%s", team_name);
        printf("请输入需要删除项目编号：");
        scanf("%s", pro_num);
        BOOL juge1=DelProjectNode(gp_head, team_name, pro_num);
        if(juge1==TRUE)
            printf("删除成功！\n");
        else
            printf("删除失败！\n");
        printf("按任意键返回..\n");
        getch();
        InitInterface();
    }
    if(juge==3)
    {
        PROJECT_NODE *ppro_node0;
        char pro_num0[15];
        char team_name0[30];
        printf("请输入需要修改的科研项目所属团队名称：");
        scanf("%s", team_name0);
        printf("请输入需要修改的科研项目编号：");
        scanf("%s", pro_num0);
```

```
            getchar();
            ppro_node0=(PROJECT_NODE *)malloc(sizeof(PROJECT_NODE));
            strcpy(ppro_node0->num,pro_num0);
            strcpy(ppro_node0->team,team_name0);
            printf("请输入修改后的项目类型：");
            scanf("%c", &ppro_node0->type);
            printf("请输入修改后的项目起始时间：");
            scanf("%s", ppro_node0->start);
            printf("请输入修改后的项目经费：");
            scanf("%f", &ppro_node0->money);
            printf("请输入修改后的负责人：");
            scanf("%s", ppro_node0->dutyman);

            BOOL juge2=ModifProjectNode(gp_head, team_name0, pro_num0, ppro_node0);
            if(juge2==TRUE)
                printf("修改成功！\n");
            else
                printf("修改失败！\n");
            printf("按任意键返回..\n");
            getch();
            InitInterface();
        }

        return bRet;
}
/*查询院系基本信息*/
BOOL QueryCollegeInfo(void)
{
        BOOL bRet = TRUE;
        printf("按院系负责人查找院系按 1\n");
        printf("按院系名称关键字查找院系按 2\n");
        int juge;
```

```
    scanf("%d", &juge);
    InitInterface();

    if(juge==1)
    {
        char dutyman[12];
        COLLEGE_NODE *pcol_node;
        printf("请输入院系负责人：");
        scanf("%s", dutyman);
        pcol_node=SeekColNodeMan(gp_head, dutyman);
        printf("%-20s%-20s%-20s\n", "院系名称", "负责人", "联系电话");
        for(;pcol_node!=NULL;pcol_node=pcol_node->next)
        {
            printf("%-20s%-20s%-20s\n", pcol_node->name, pcol_node->dutyman,
pcol_node->tel);
        }
        printf("查找完毕，按任意键返回。\n");
        getch();
        InitInterface();
    }
    else if(juge==2)
    {
        char name_m[20];
        COLLEGE_NODE *pcol_node;
        printf("请输入院系名称关键字：");
        scanf("%s", name_m);
        pcol_node=SeekCollegeNodeNameM(gp_head,name_m);
        printf("%-20s%-20s%-20s\n", "院系名称", "负责人", "联系电话");
        for(;pcol_node!=NULL;pcol_node=pcol_node->next)
        {
            printf("%-20s%-20s%-20s\n", pcol_node->name, pcol_node->dutyman,
pcol_node->tel);
```

```
            }
            printf("查找完毕，按任意键返回。\n");
            getch();
            InitInterface();
        }

        return bRet;
}
/*查询团队基本信息*/
BOOL QueryTeamInfo(void)
{
        BOOL bRet = TRUE;
        printf("按团队名称关键字查找科研团队按 1\n");
        printf("按教师人数条件查找科研团队按 2\n");
        int juge;
        scanf("%d", &juge);
        InitInterface();

        if(juge==1)
        {
            char name_m[20];
            TEAM_NODE *pteamnode;
            printf("请输入团队名称关键字：");
            scanf("%s", name_m);
            pteamnode=SeekTeamNodeNameM(gp_head,name_m);
            printf("%-20s%-20s%-20s%-20s%-20s\n", "团队名称", "负责人", "所属院系", "教师人
数", "研究生人数");
            for(;pteamnode!=NULL;pteamnode=pteamnode->next)
            {
                printf("%-20s%-20s%-20s%-20d%-20d\n",                pteamnode->name,
pteamnode->dutyman, pteamnode->college, pteamnode->num_tc, pteamnode->num_stu);
            }
```

```
            printf("查找完毕，按任意键返回。\n");
            getch();
            InitInterface();
        }
        else if(juge==2)
        {
            char mark[20];
            int num_tc;
            int juge0;
            TEAM_NODE *pteamnode;
            printf("请输入判断条件大于 1，等于 2，小于 3：");
            scanf("%d", &juge0);
            printf("请输入教师人数：");
            scanf("%d", &num_tc);
            pteamnode=SeekTeamNodeTeacher(gp_head, juge0, num_tc);
            for(;pteamnode!=NULL;pteamnode=pteamnode->next)
            {
                printf("%-20s%-20s%-20s%-20d%-20d\n", pteamnode->name,
pteamnode->dutyman, pteamnode->college, pteamnode->num_tc, pteamnode->num_stu);
            }
            printf("查找完毕，按任意键返回。\n");
            getch();
            InitInterface();
        }

        return bRet;
}
/*查询科研项目基本信息*/
BOOL QueryProjectInfo(void)
{
        BOOL bRet = TRUE;
        printf("按项目编号查找科研项目按 1\n");
```

```
    printf("按所属团队查找科研项目按 2\n");
    int juge;
    scanf("%d", &juge);
    InitInterface();

    if(juge==1)
    {
        PROJECT_NODE *ppro_node;
        char pro_num[15];
        printf("请输入项目编号：");
        scanf("%s", pro_num);
        printf("%-20s%-30s%-20s%-20s%-20s%-20s\n", "项目编号", "项目类型", "起始时间",
"项目经费", "负责人", "所属团队");
        ppro_node=SeekProjectNodeNum(gp_head, pro_num);
        for(;ppro_node!=NULL;ppro_node=ppro_node->next)
        {
            char pro_type[50];
            if(ppro_node->type=='1') strcpy(pro_type,"973 计划项目");
            else if(ppro_node->type=='2') strcpy(pro_type,"国家自然科学基金项目");
            else if(ppro_node->type=='3') strcpy(pro_type,"863 计划项目");
            else if(ppro_node->type=='4') strcpy(pro_type,"国际合作项目");
            else if(ppro_node->type=='5') strcpy(pro_type,"横向项目");
            printf("%-20s%-30s%-20s%-20.2f%-20s%-20s\n",  ppro_node->num,  pro_type,
ppro_node->start, ppro_node->money, ppro_node->dutyman, ppro_node->team);
        }
        printf("查找完毕，按任意键返回。\n");
        getch();
        InitInterface();
    }
    else if(juge==2)
    {
        COLLEGE_NODE *pcol_node;
```

```
        TEAM_NODE *pteamnode;
        PROJECT_NODE *ppro_node;
        char pro_team[30];
        printf("请输入项目所属团队：");
        scanf("%s", pro_team);
        printf("%-20s%-30s%-20s%-20s%-20s%-20s\n", "项目编号", "项目类型", "起始时间",
"项目经费", "负责人", "所属团队");

        for(pcol_node=gp_head;pcol_node!=NULL;pcol_node=pcol_node->next)
        {

    for(pteamnode=pcol_node->tnext;pteamnode!=NULL;pteamnode=pteamnode->next)
            {

    for(ppro_node=pteamnode->pnext;ppro_node!=NULL;ppro_node=ppro_node->next)
                {
                    if(strcmp(ppro_node->team,pro_team)==0)
                    {
                        char pro_type[50];
                        if(ppro_node->type=='1') strcpy(pro_type,"973 计划项目");
                        else if(ppro_node->type=='2') strcpy(pro_type,"国家自然科学基金
项目");
                        else if(ppro_node->type=='3') strcpy(pro_type,"863 计划项目");
                        else if(ppro_node->type=='4') strcpy(pro_type,"国际合作项目");
                        else if(ppro_node->type=='5') strcpy(pro_type,"横向项目");
                        printf("%-20s%-30s%-20s%-20.2f%-20s%-20s\n",
ppro_node->num, pro_type, ppro_node->start, ppro_node->money, ppro_node->dutyman,
ppro_node->team);
                    }
                }
            }
        }
```

```
            printf("查找完毕，按任意键返回。\n");
            getch();
            InitInterface();
        }

        return bRet;
}
/*统计人数及人数比*/
BOOL StatManRate(void)
{
        printf("各院系人员情况\n");
        printf("%-20s%-20s%-20s%-20s\n", "院系名称", "研究生总数", "教师总数", "人数比");
        PEOPLES_NODE *ret;
        ret=StatPeopleRate(gp_head);
        for(;ret!=NULL;ret=ret->next)
        {
            printf("%-20s%-20d%-20d%-20.2f\n", ret->col_name, ret->student, ret->teacher,
ret->scale);
        }
        printf("统计完毕，按任意键返回。\n");
        getch();
        InitInterface();

        return TRUE ;
}
/*统计各类科研项目总数*/
BOOL StatProjectType(void)
{
        printf("各院系各类科研项目数\n");
        printf("%-20s%-20s%-20s%-20s%-20s\n", "院系名称", "科研项目总数", "973 项目", "863
项目", "总资金");
        PROJECT_NUM_NODE *ret;
```

```
    ret=StatCollegeProjectNum(gp_head);
    for(;ret!=NULL;ret=ret->next)
    {
        printf("%-20s%-20d%-20d%-20d%-20.2f\n", ret->col_name, ret->total, ret->p973, ret->p863, ret->money);
    }
    printf("统计完毕，按任意键返回。\n");
    getch();
    InitInterface();

    return TRUE;
}
/*统计优秀科研团队*/
BOOL StatTeam(void)
{
    int i;
    printf("优秀科研团队\n");
    printf("%-20s%-30s%-20s\n", "团队名称", "自然科学基金项目总数", "总资金");
    TEAM_NASI_NODE *ret;
    ret=StatTeamNasiTen(gp_head);
    TEAM_NASI_NODE *p;
    int len=0;
    for(p=ret;p!=NULL;p=p->next)
    {len++;}
    for(i=1;i<=10&&i<=len;i++)
    {
        printf("%-20s%-30d%-20.2f\n", ret->team_name, ret->num_nasi, ret->money);
        ret=ret->next;
    }
    printf("统计完毕，按任意键返回。\n");
    getch();
    InitInterface();
```

```
    return TRUE;
}
/*统计教师人均科研项目*/
BOOL StatProjectPer(void)
{
    int i;
    printf("各科研团队教师人均科研项目\n");
    printf("%-20s%-20s%-20s%-20s\n", "团队名称", "教师人数", "项目总数", "人均科研项目
");
    TEAM_PROTEA_NODE *ret;
    TEAM_PROTEA_NODE *p;
    ret=StatProTeaScale(gp_head);
    int len=0;
    for(p=ret;p!=NULL;p=p->next)
    {len++;}
    for(i=1;i<=5&&i<=len;i++)
    {
        printf("%-20s%-20d%-20d%-20.2f\n", ret->team_name, ret->num_tc, ret->num_pro,
ret->scale);
        ret=ret->next;
    }
    printf("统计完毕，按任意键返回。\n");
    getch();
    InitInterface();

    return TRUE;
}
/*帮助*/
BOOL AboutDorm(void)
{
    BOOL bRet = TRUE;
```

```
    printf("版本号：tzt-sys1.1.1\n");
    printf("最终解释权归我所有\n");
    printf("按任意键返回...");
    getch();
    InitInterface();
    return bRet;
}
```

五、运行测试与结果分析

1.测试数据：

院系信息：

院系名称	负责人	联系电话
计算机	田二	10086
新闻	张三	10010
建筑	李四	10000

团队信息：

团队名称	负责人	所属院系	教师人数	研究生人数
计 127	田二	计算机	5	20
计 128	刘一	计算机	3	15
新 111	张三	新闻	4	18
新 112	王五	新闻	1	10
建 101	李四	建筑	4	17
建 102	赵六	建筑	5	20

项目信息：

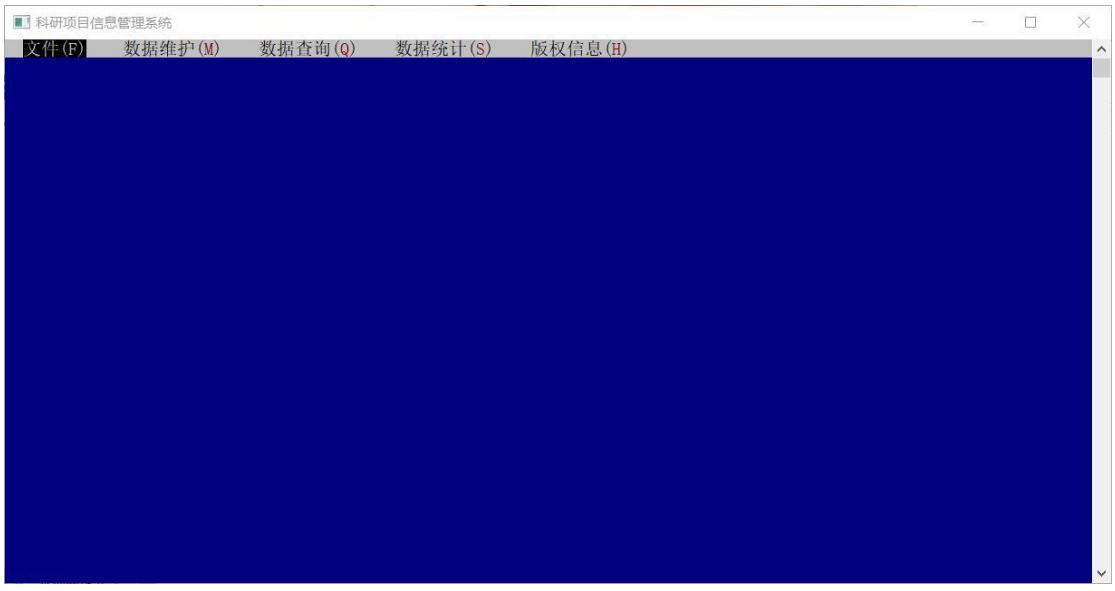
项目编号	项目类型	起始时间	项目经费	负责人	所属团队
001	1	20170901	1111	田二	计 127
002	2	20170902	1210	田二	计 127
003	3	20170903	1000	刘一	计 128
004	4	20170904	1314	刘一	计 128

005	5	20170901	1100	张三	新 111
006	1	20170902	1001	张三	新 111
007	2	20170903	1010	王五	新 112
008	3	20170904	1011	王五	新 112
009	4	20170901	1501	李四	建 101
010	5	20170902	1717	李四	建 101
011	1	20170903	1000	赵六	建 102
012	2	20170904	999	赵六	建 102

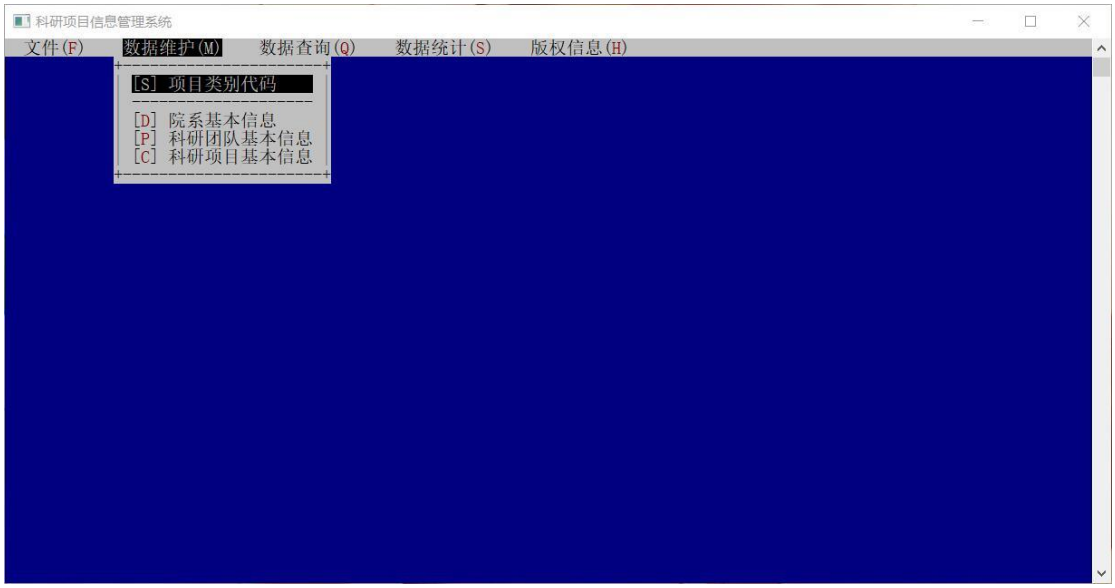
2.系统运行：

系统初始化界面：

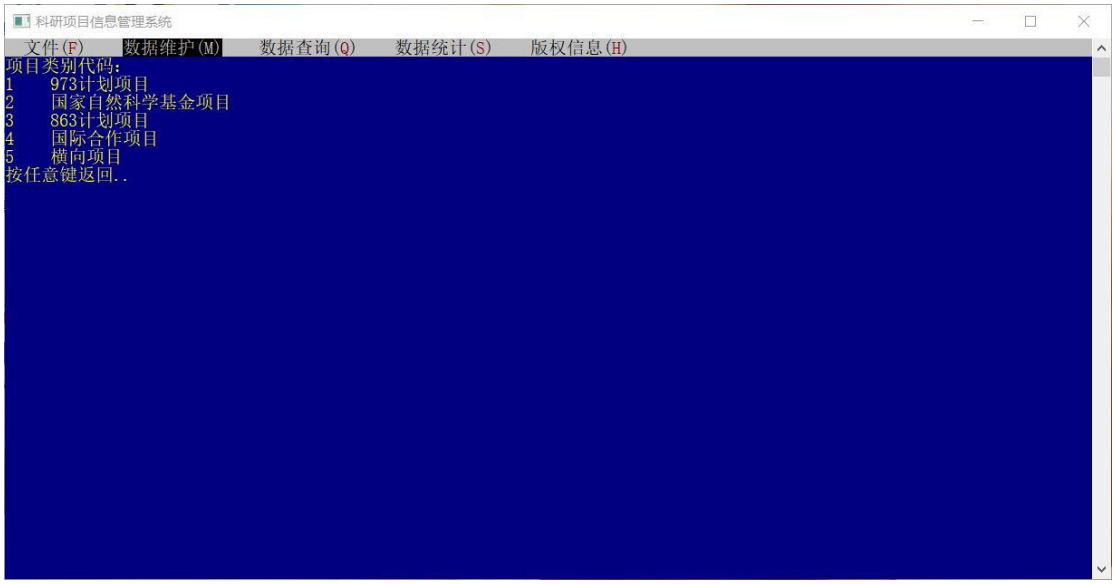
系统界面：



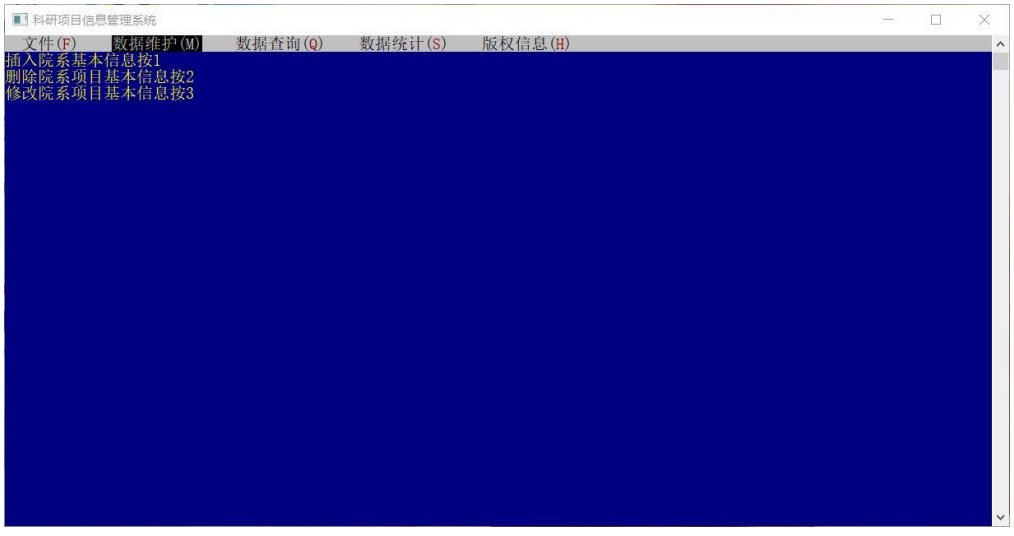
数据维护界面：



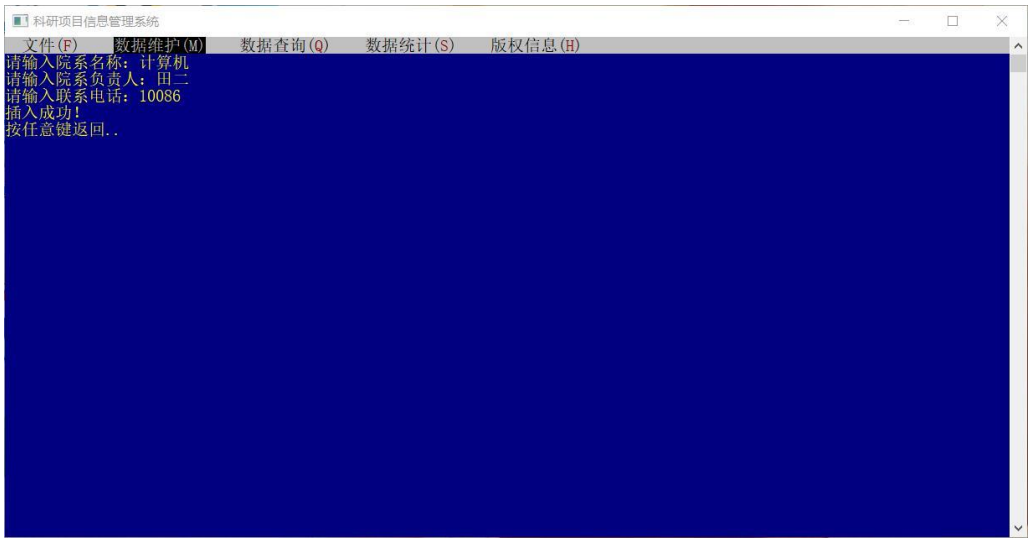
项目类别代码：



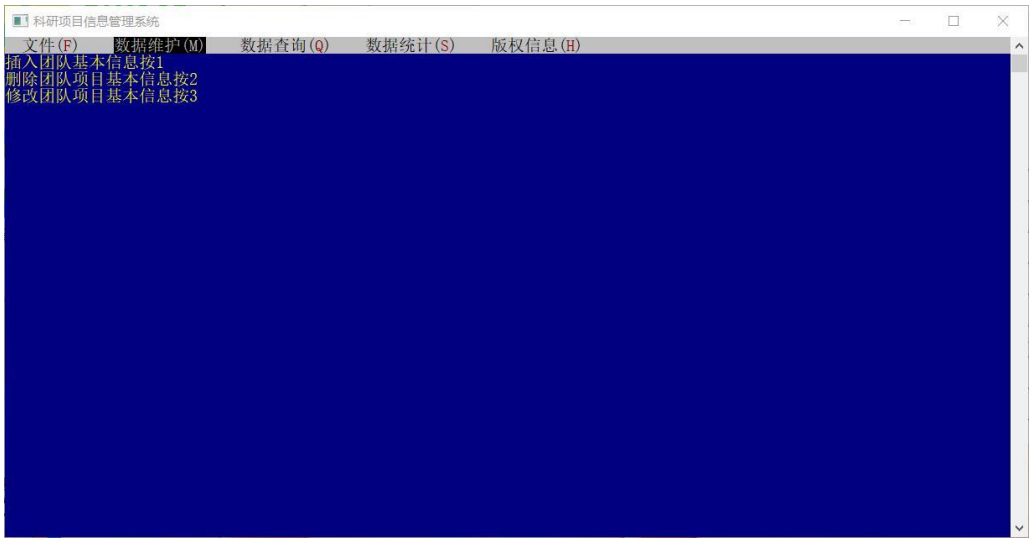
院系信息维护：



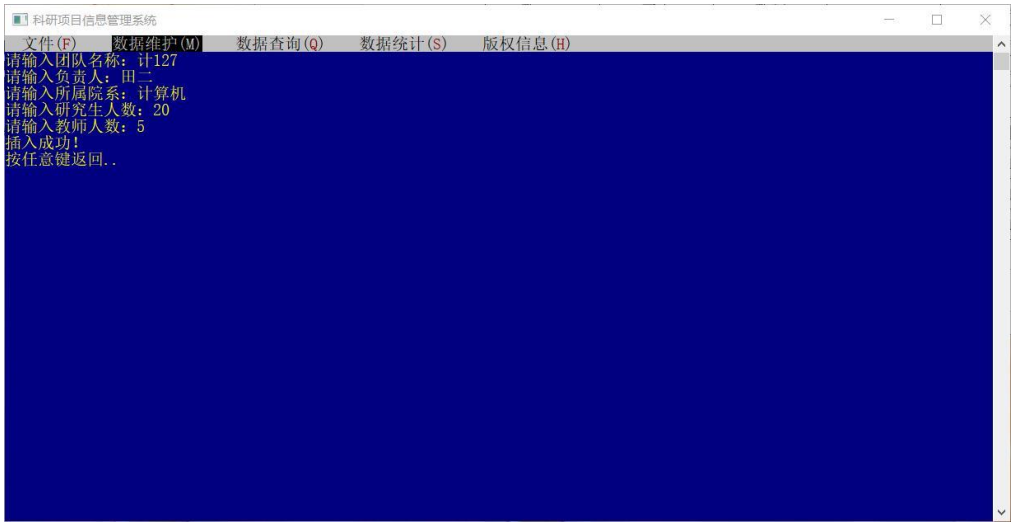
插入院系信息：



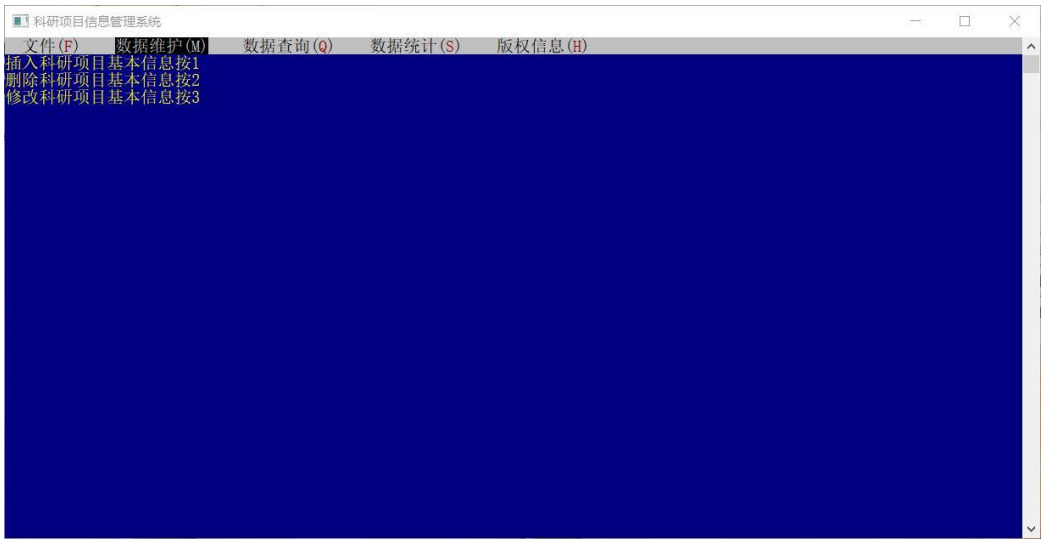
团队信息维护：



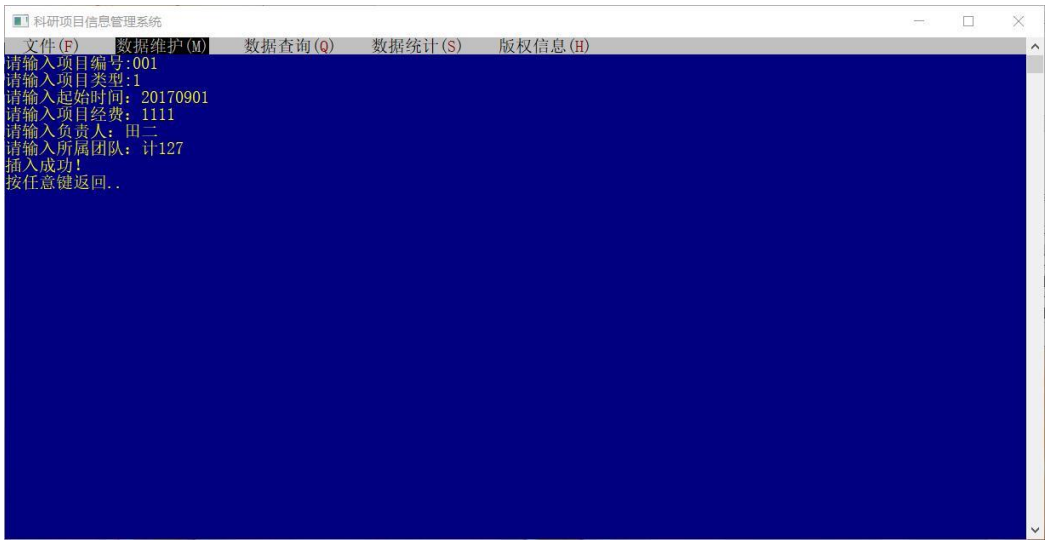
插入团队信息：



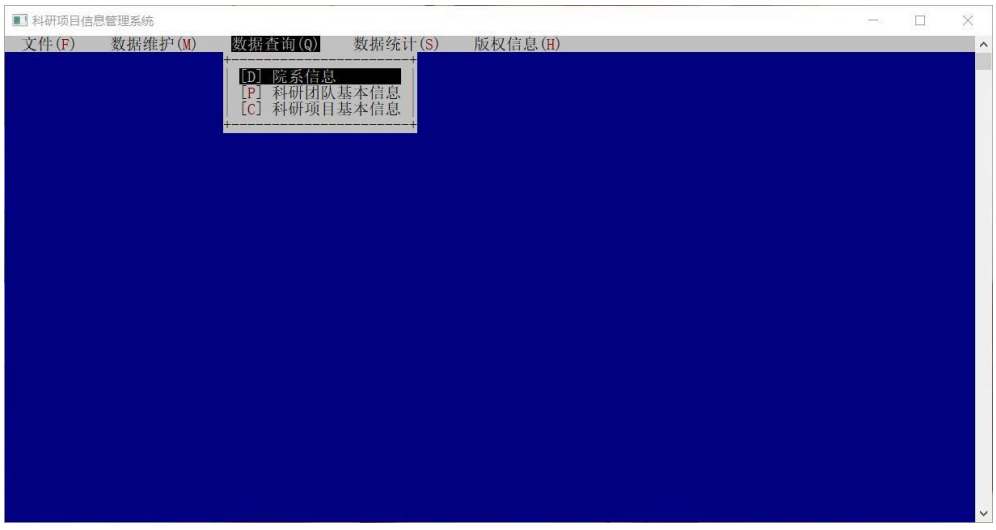
项目信息维护：



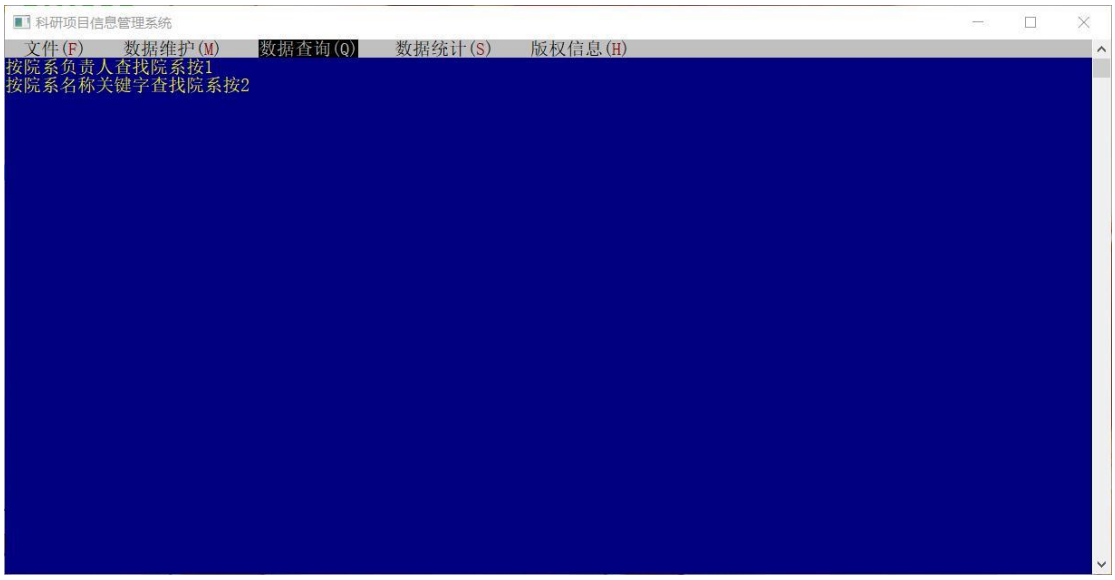
插入项目信息：



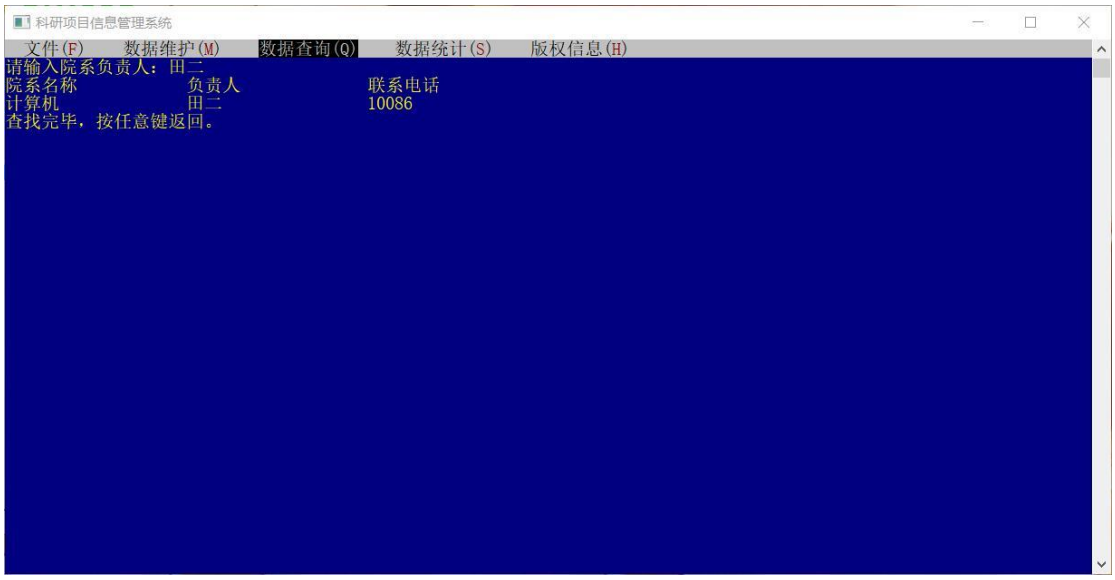
数据查询：



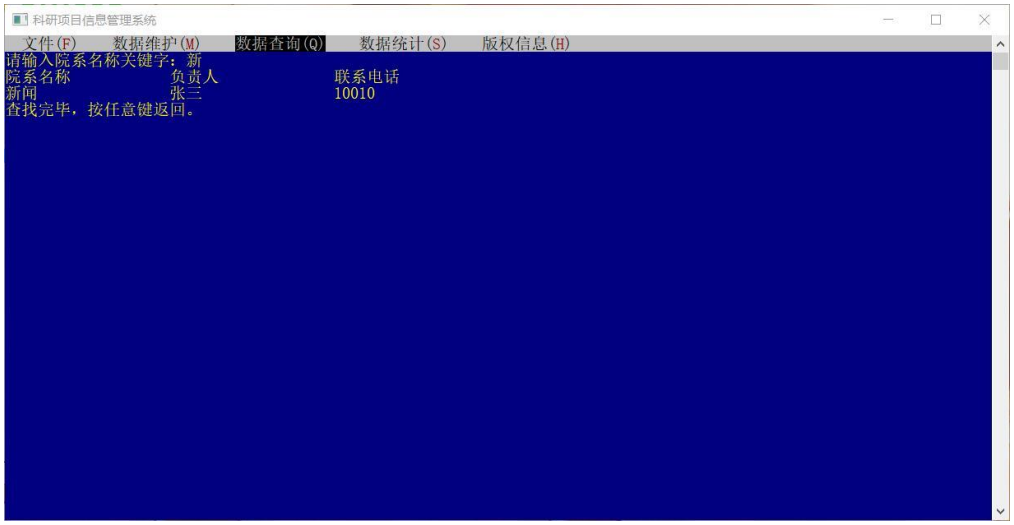
院系信息查询：



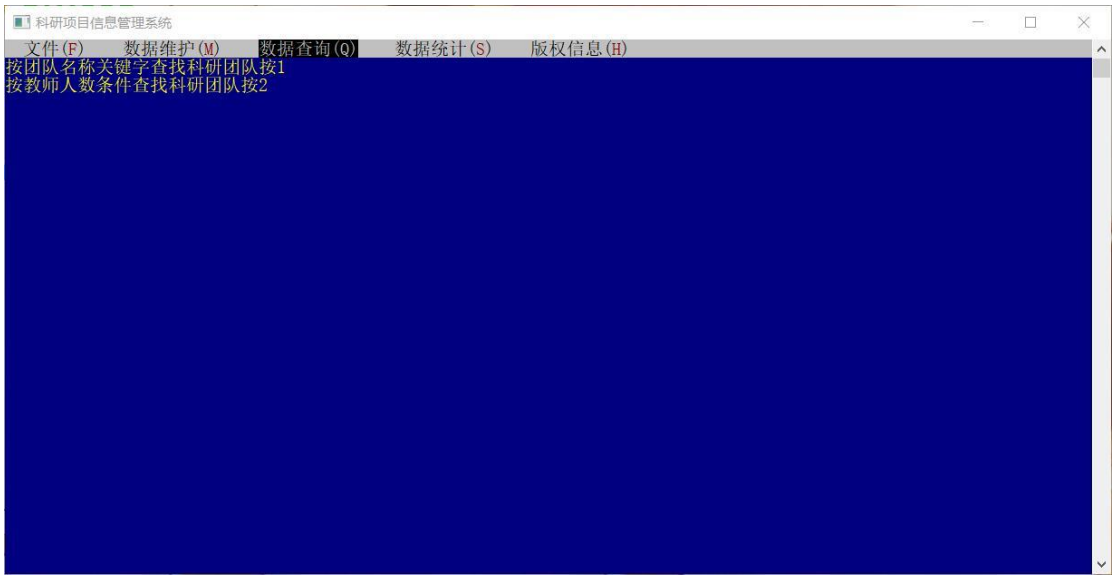
按院系负责人查询：



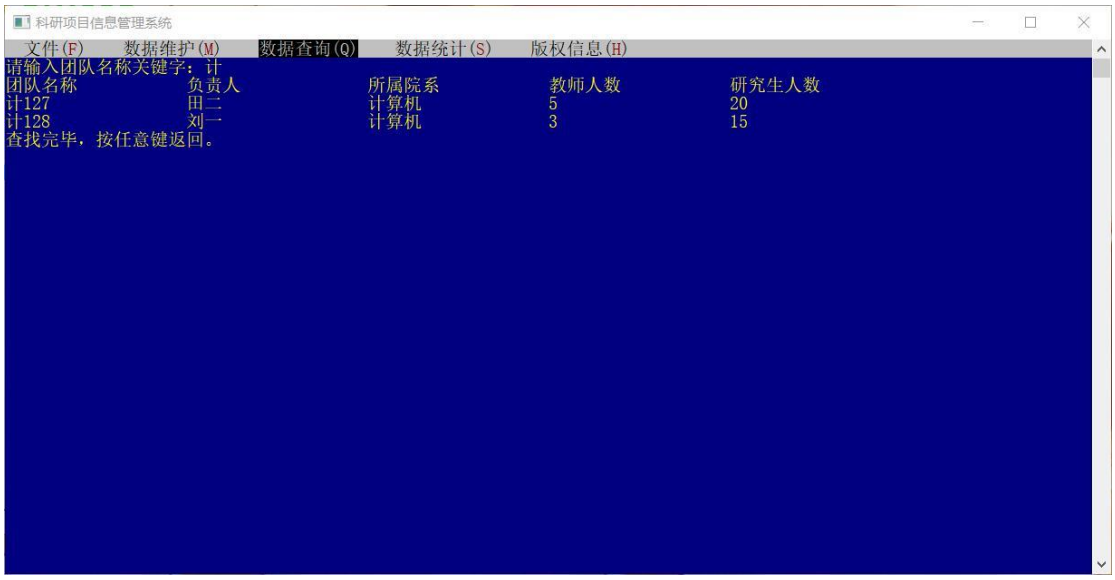
按院系名关键字查询：



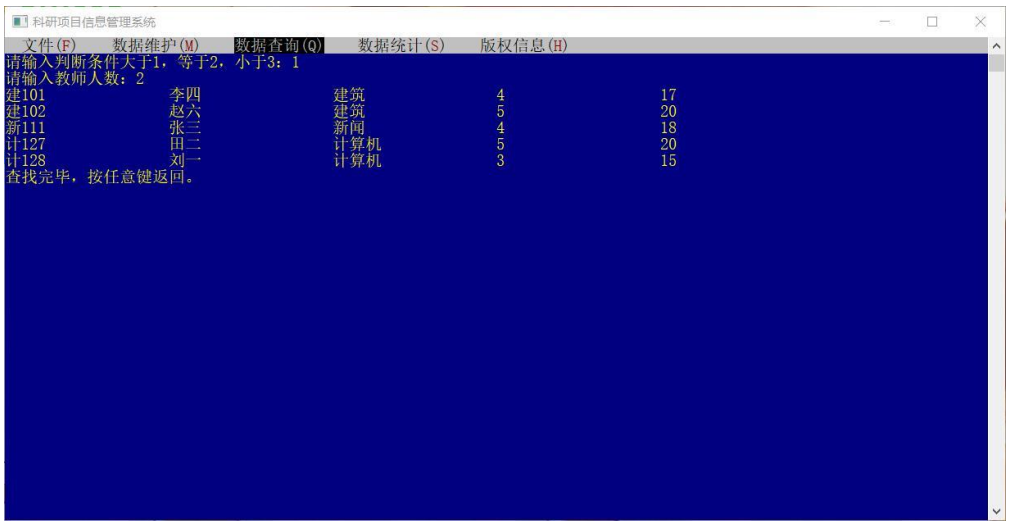
团队信息查询：



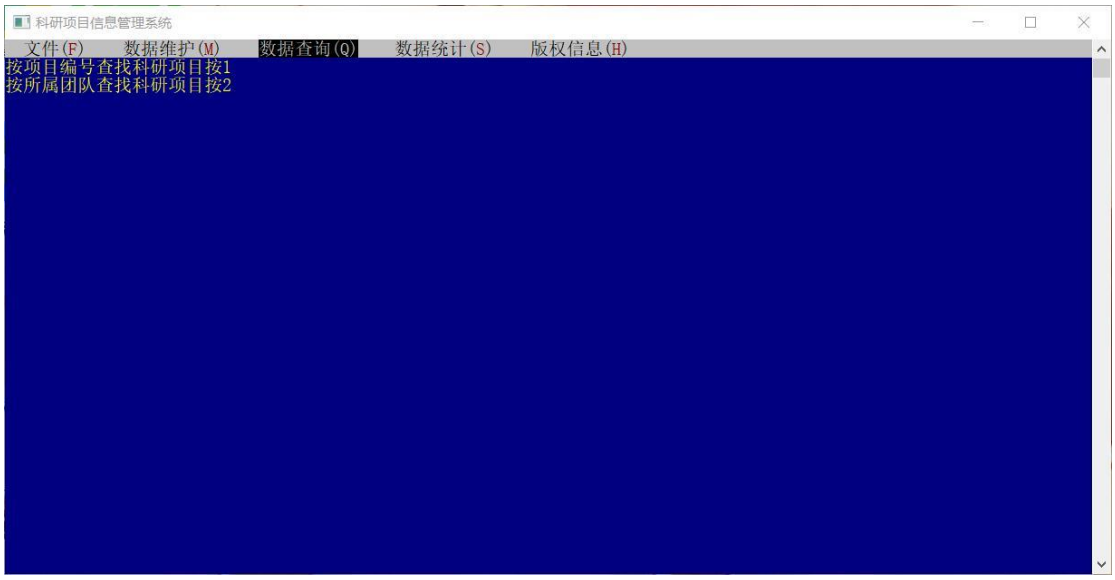
按团队名关键字查询：



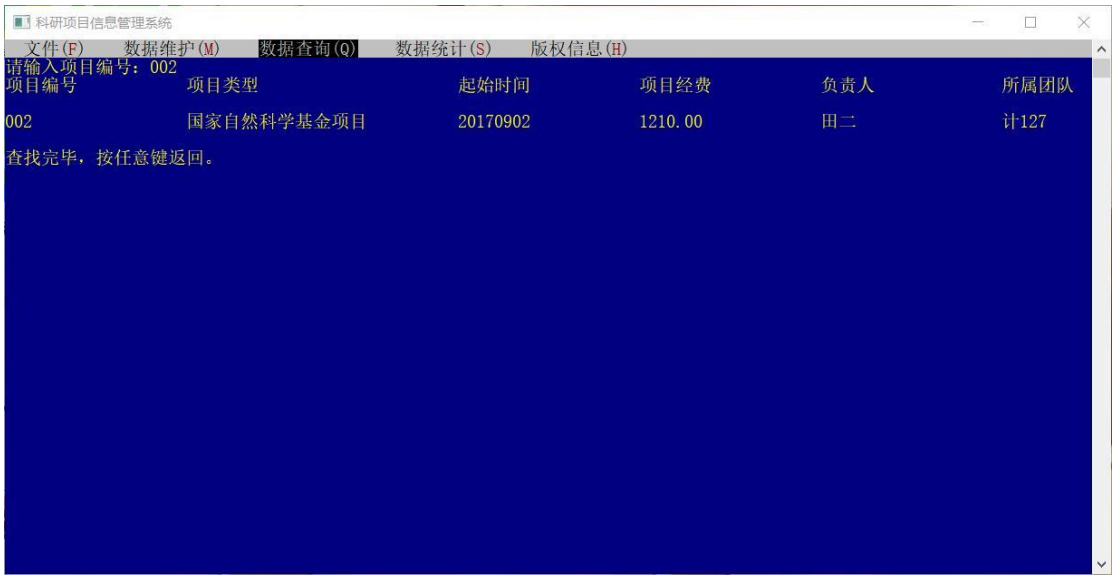
按团队教师人数查询：



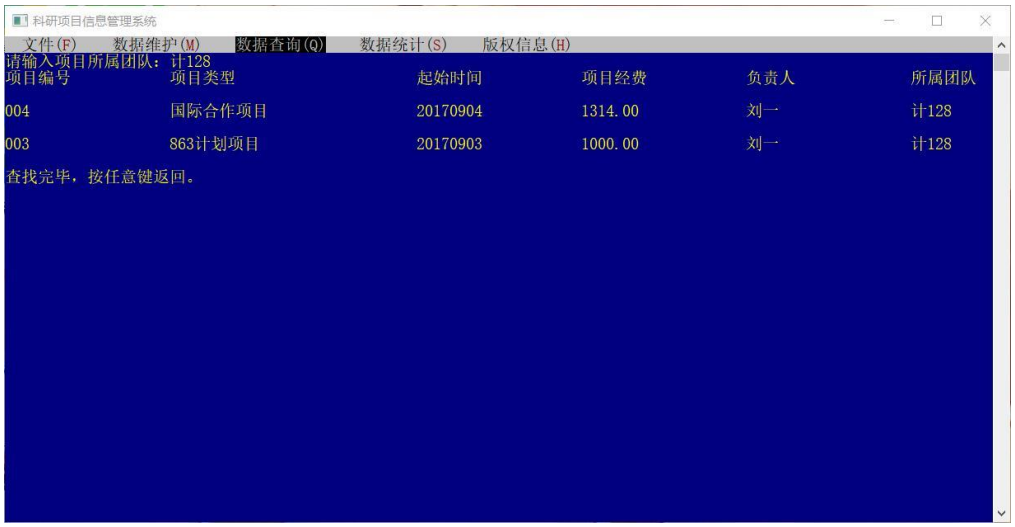
项目信息查询：



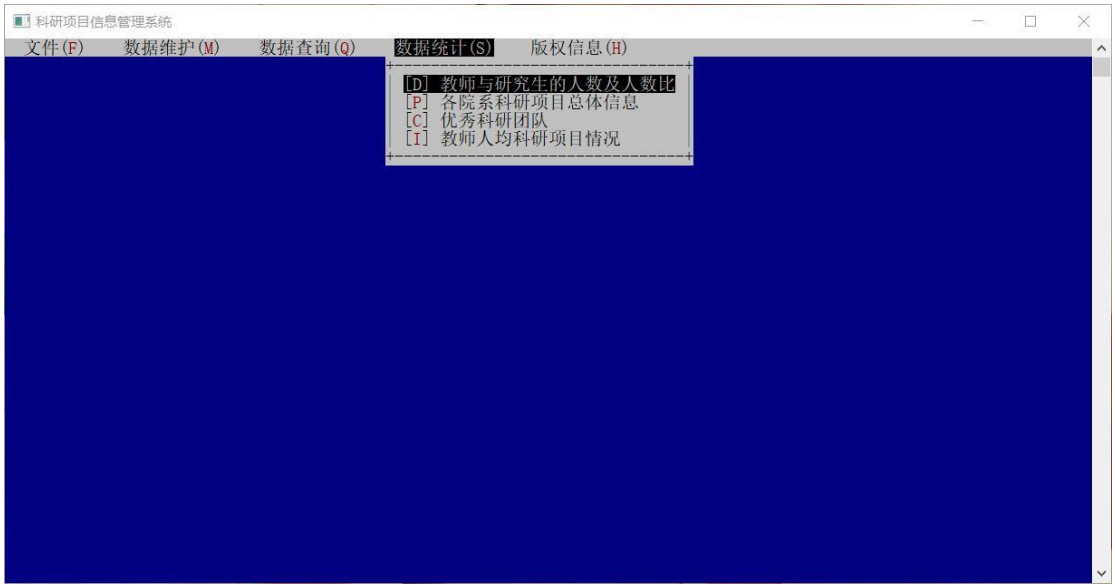
按项目编号查询：



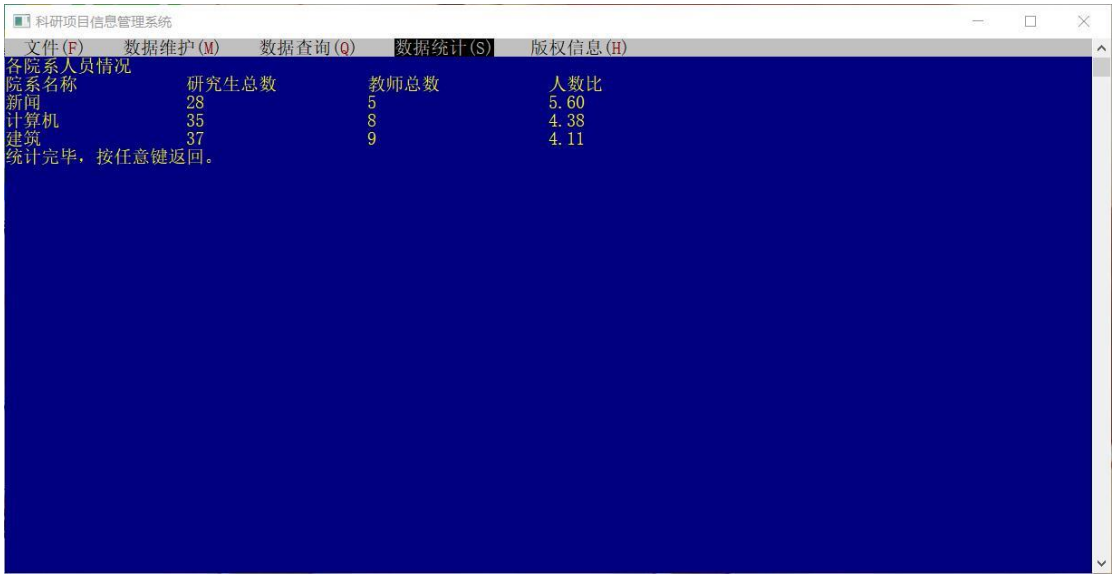
按项目所属团队查询：



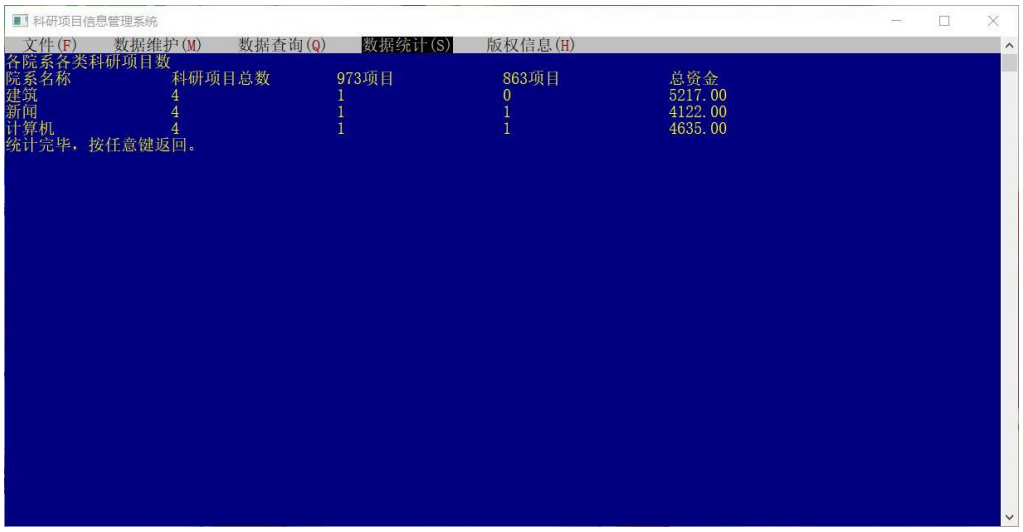
数据统计：



统计院系人数比：

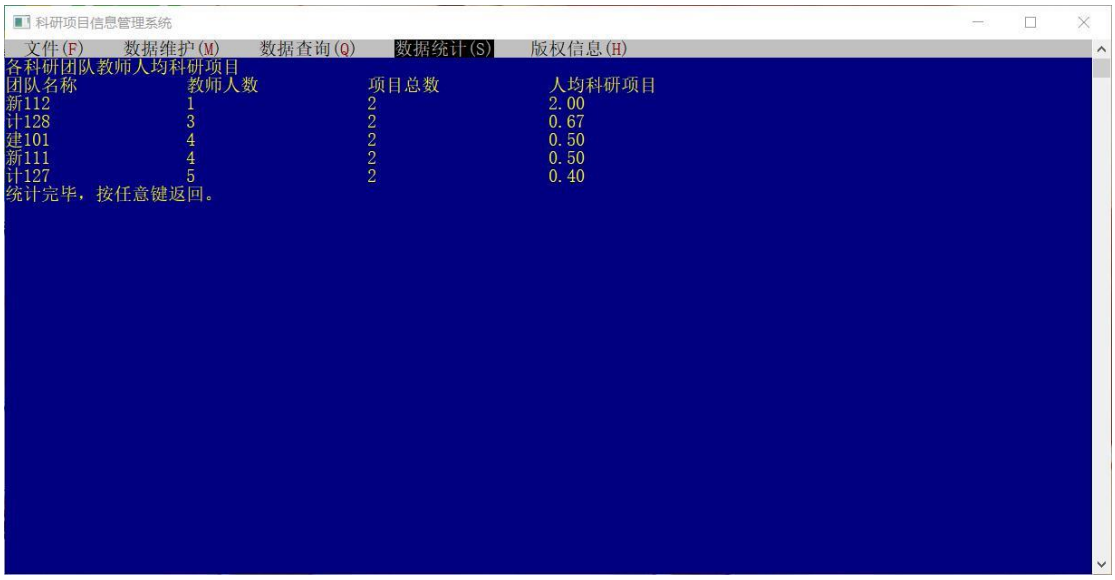


统计各类科研项目总数：

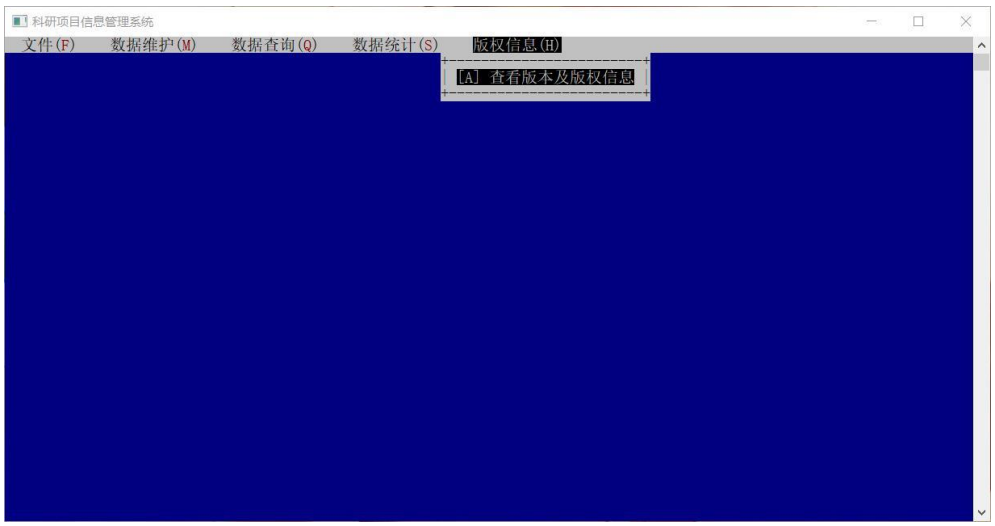


统计自然科学基金项目总数：

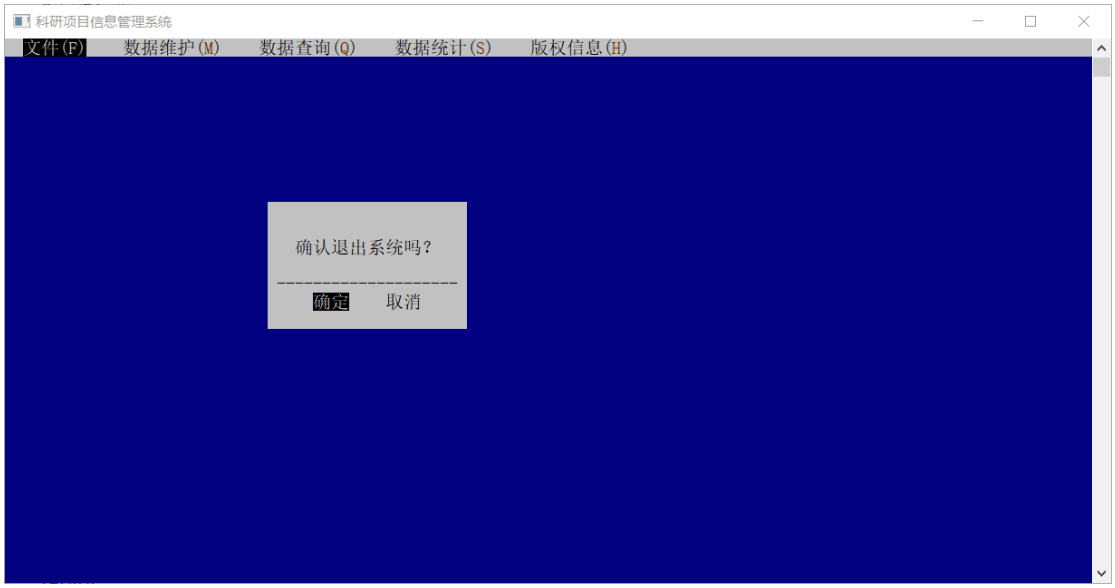
统计人均科研项目：



版权信息：



退出系统：



六、总结

课程设计师培养学生综合运用所学知识，发现、提出、分析和解决实际问题，锻炼动手能力的重要环节，是对我们的实际工作能力的具体训练和考察。

随着科学技术发展的日新月异，当今计算机应用在生活中可以说是无处不在，一次作为二十一世纪的大学生来说掌握程序开发技术是十分必要的，而 C 语言是最常见，功能强大的一种编程语言。因此做好 C 语言课设是十分必要的。回顾此次课程设计，至今我仍感慨颇多，的确，从拿到题目待完成整个题目，从理论到实践，在整整 10 天的日子里，我学到了很多很多东西，不仅巩固了之前的知识，也学到了很多书本上没有的知识，养成了良好的编程习惯。

我做的是科研项目管理系统的课程设计，虽然是一个很简单的小程序，但对我一个初学者来说却十分困难。由于是第一次写较大型的程序，开始我的进展十分缓慢，第一天下午在宿舍做了一个下午却没有丝毫进展，不知从何做起。还好我翻阅了 C 语言实验与课程设计这本书，书里对课程设计的整体设计方法给了我很大的启示。经过一波三折，我终于开始了正式编程。

编程是一件枯燥痛苦却又伴随着乐趣的事，开始时出于完成作业得到学分的压力，我强迫自己坚持写下去，按照老师所说的模块化思维，分部分的进行编写。而后在代码的编写过程中，每个模块的完成都会带给我不同的喜悦，对自己所创造的程序有着深深的满足感。

编程是一件高精度，模范化的事情，稍有疏忽就会影响全局，可能因为某一处的小错误导致程序无法正常运行。比如我在编写链表排序函数的过程中，在 for 循环的后面多加了一个分号，导致我的程序始终出现难以理解的 bug，最终查找了整整一天时间才找到。

通过这次课程设计，是我对 C 语言有了更进一步的了解，要想学好它重在实践，要不断地上机操作才能更好的学习并掌握它。并且在收获知识，提高能力的同时，我也学到了很多人生哲理，懂得了怎样将一件事模块化，分层次地处理，并且养成了做事严谨的态度。因此，在此后的生活学习中，我一定会把此次课程设计学习到的知识铭记于心，不畏艰难，勇往直前。

七、参考文献

C 语言实验与课程设计　　李开　卢萍　曹计昌

C 语言与程序设计　曹计昌 卢萍 李开