

# 《数据结构》

## 课程设计报告

题        目：\_\_\_\_\_选修课信息管理系统\_\_\_\_\_

专 业 班 级：\_\_\_\_\_

学 生 姓 名：\_\_\_\_\_

学        号：\_\_\_\_\_

指 导 教 师：\_\_\_\_\_

学        期：\_\_\_\_\_2021-2022 学年第二学期\_\_\_\_\_

## 软件工程 专业课程报告任务书

学生姓名		专业班级		学号	
题 目	选修课管理信息系统				
课题性质	工程设计		课题来源	自拟课题	
指导教师			同组姓名	无	
主要内容	<p>设计完成一个选修课管理信息系统。本系统拟实现以下功能：</p> <p>(1) 学生选课：学生查看选修课程，选课退课等操作</p> <p>(2) 选课管理：管理员对选课进行查询。添加、删除、修改。</p> <p>(3) 系统管理：管理员对学生账号的管理，管理员密码更改。</p> <p>(4) 系统维护：备份恢复数据，存储提取数据</p> <p>(5) 主页面帮助信息</p> <p>采用 Visual Studio 2019 工具进行开发实现。</p>				
任务要求	<p>(1). 能够综合运用结构化程序设计知识，针对课题进行需求分析、设计功能合理的自定义函数；</p> <p>(2). 能够用菜单组织、调用各功能函数；每个功能自定义一个函数；</p> <p>(3). 能够选择合适的开发技术和开发环境进行代码实现，实现的系统功能正确、性能适用，并提供完整的程序代码；</p> <p>(4). 能够用文件保存系统数据；</p> <p>(5). 能够对系统进行异常处理，检查用户输入数据的有效性。在用户输入数据错误（如数据类型错误、数值错误）或者无效时，不会中断程序执行；系统具有一定的健壮性；</p> <p>(6). 撰写、提交课程设计报告，能够对完成的系统清晰描述分析、设计、调试、运行的结果，并能回答老师提出的问题。</p>				
参考文献	<p>[1] 甘勇等. C 语言程序设计(第二版)[M]. 北京：中国铁道出版社有限公司，2020.</p> <p>[2] 钦铭、颜辉.C 语言程序设计（第4版）[M].北京：高等教育出版社，2020.</p> <p>[3] 齐治昌等. 软件工程（第4版）[M]. 北京：高等教育出版社，2019.</p> <p>[4] 严蔚敏等.数据结构（C 语言版）第2版[M]. 北京：人民邮电出版社，2019.</p> <p>[5] 王新等.C 语言课程设计[M]. 北京：清华大学出版社，2009.</p>				
审查意见	<p style="text-align: center; font-size: 2em;">同意</p> <p>教研室主任签字：_____ 2022 年 2 月 21 日</p>				

## 1 需求分析

通过和指导老师交流,了解到本系统中的数据来源于标准输入设备(如键盘)或者来自某文件,可以实现对选修课信息和学生信息的添加,根据需要也可查询用户所需要的信息、删除无价值的信息、修改密码信息等,总之本系统,要具备如下功能:添加信息、查询信息、删除信息、修改信息、退出和保存信息。具体如下。

### 1.1 选择课程

能根据现有学生的学分与总学分进行对比。若已修学分小于总学分,实现选课功能。

根据输入的课程名称或课程编号两种方式实现最终的选课功能。通过判断该课程人数是否已满和课程是否选过以及是否存在该函数,最终用链表的形式实现选课功能。

### 1.2 查询信息

学生:能够实现根据课程开设学期和课程的名称,编号查找相关课程信息,根据方式不同实现课程信息的显示。

管理员:可将查找对象分为学生和课程。对于查找学生信息而言,根据输入学生的学号或者进查找;对于查找课程信息,根据课程开设学期和课程的名称,编号查找相关课程信息。根据方式不同,实现课程或者学生信息的显示。

### 1.3 删除信息

学生:根据输入的课程编号而删除某一门课程,实现删除课程中的学生信息和学生中的课程信息

管理员:根据输入的信息实现对学生和课程的删除。对于删除学生信息而言,输入学生的学号或姓名,判断是能够删除,若能,便删去相关信息;对于删除课程信息而言,输入课程的编号或名称,判断是否能够删除,若能,便删去相关信

息。

### 1.4 增添信息

学生：根据所列的课程信息可以增加自己的选课信息以实现选课功能，或者删除自己的选课信息以实现退课功能。

管理员：可以增添学生信息，课程信息，以供选课。

### 1.5 修改信息

学生：可以更改自己的账号的密码，更改自己的选课信息。

管理员：可以修改学生及课程信息，例如学生基本信息以及课程基本信息等，更改管理员密码。

### 1.6 退出和保存信息

可以在管理员页面实行备份恢复功能，可以在主页面和二级界面选择退出程序，在增删查改后也都会默认施行自动备份。

## 2 概要设计

### 2.1 数据类型定义

(1) 定义学生的元素类型

```
student
-成员名;
char name[MAXN];
char num[MAXN];
char majorclass[20];
double score_all;
double score_sel;
char password[9];
int course_sum;
char course[50][MAXN];
struct students* next;
```

图 2.1 student 结构体图

```
typedef struct students
{
    char name[MAXN];           //姓名
```

```

char num[MAXN];           //学号
char majorclass[20];      //专业班级
double score_all;         //应选总学分
double score_sel;         //已选总学分
char password[9];         //密码,8位
int course_sum;           //选课总数
char course[50][MAXN];    //记录选课课程的编号
struct students* next;     //链表指针
}student;

```

## (2) 定义课程的元素类型

### course

-成员名:

```

char num[MAXN];
char name[MAXN];
char nature[MAXN];
char term[MAXN];
double time_all;
double time_teach;
double time_exp;
double score;
int stu_max;
int stu_sum;
char stu[100][MAXN];
struct courses* next;

```

图 2.2 course 结构体图

```

typedef struct courses
{
    char num[MAXN];
    char name[MAXN];
    char nature[MAXN];
    char term[MAXN];
    double time_all;
    double time_teach;
    double time_exp;
    double score;
    int stu_max;
    int stu_sum;
    char stu[100][MAXN];
    struct courses* next;
}course;

```

## 2.2 功能模块结构图

根据需求分析,为了满足功能需求,按照软件开发方法学中的模块划分原则,我将本系统的主要部分分为学生界面和管理员界面两大板块,学生界面和管理员界面又可以细分为若干小模块,具体结构图如图 2.3 和图 2.4 所示。

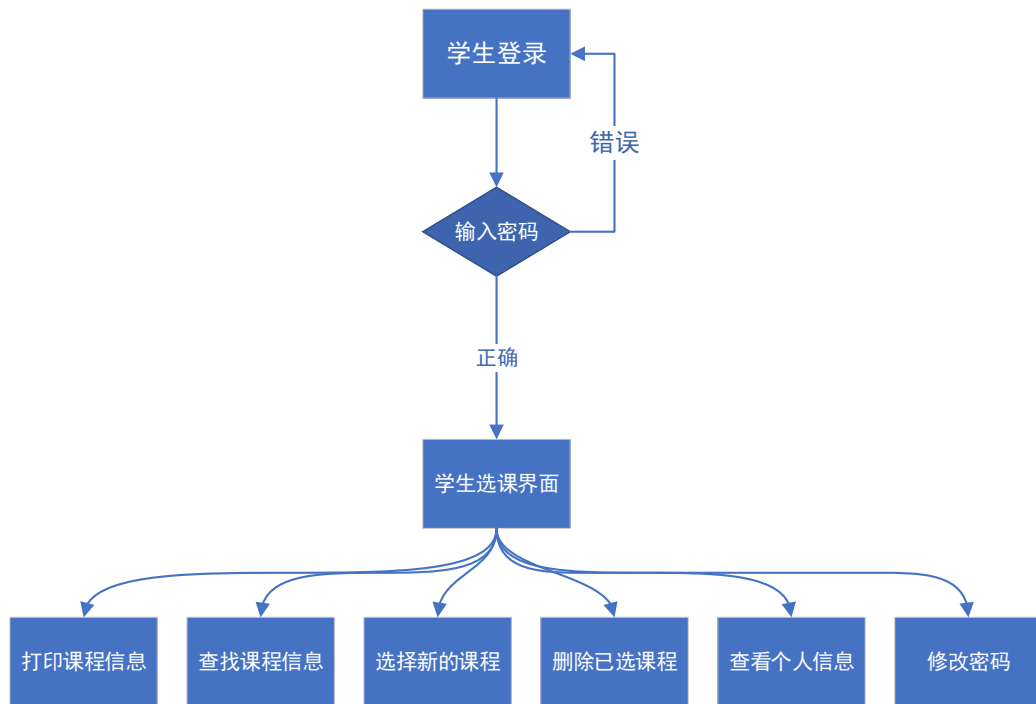


图 2.3 学生界面模块结构图

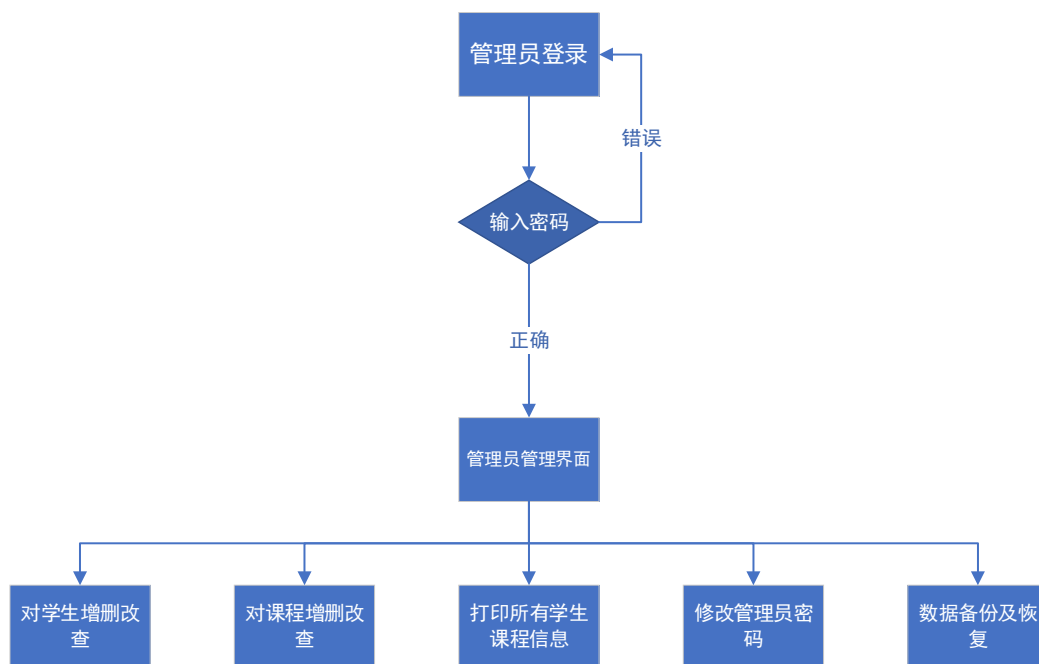


图 2.4 管理员界面模块结构图

为了实现上述功能模块，在单链表物理结构上定义了多个函数，本系统定义的函数和功能如下：

(1) 学生结构函数部分

`student* stuInfile();`

功能：将学生文件生成链表,返回链表头指针

`student* endsLocate(student* head);`

功能：传入头节点，返回尾节点

`student* stuLogin(student* head);`

功能：学生登陆函数

`student* stunameFind(student* head, char tar[]);`

功能：根据学生姓名查找学生，如果学生为头节点则返回空指针，不为头节点则返回前一个节点的指针，不存在则返回尾节点

`student* stunumFind(student* head, char tar[])`

功能：根据学生学号查找学生，如果学生为头节点则返回空指针，不为头节点则返回前一个节点的指针，不存在则返回尾节点

(2) 课程结构函数部分

`course* courInfile();`

功能：将课程文件生成链表，返回链表头指针

`course* endcLocate(course* head);`

功能：传入头节点，返回尾节点

`course* cournameFind(course* head, char tar[]);`

功能：根据课程名称查找课程节点，如果课程为头节点则返回空指针，不为头节点则返回前一个节点的指针，不存在则返回尾节点

`course* cournumFind(course* head, char tar[]);`

功能：根据课程编号查找课程，如果课程为头节点则返回空指针，不为头节点则返回前一个节点的指针，不存在则返回尾节点

剩余函数为主要功能函数，定义类型为 `void`，具体的函数定义及功能将在后面细述。

## 2.3 总结

该选修课管理信息系统主要使用的数据结构为单链表,通过定义学生与课程的结构体,并定义相应的学生课程头结点,在添加学生、课程信息后,则加入到链表当中,以达到存储信息的功能。

## 3 运行环境

- 1.硬件环境: PC 机 内存: 16GB
- 2.软件环境: 操作系统-Windows11

## 4 开发工具和编程语言

- 开发环境: Visual Studio 2019
- 编程语言: C 语言。

## 5 详细设计

### 5.1 程序主页面及二级页面

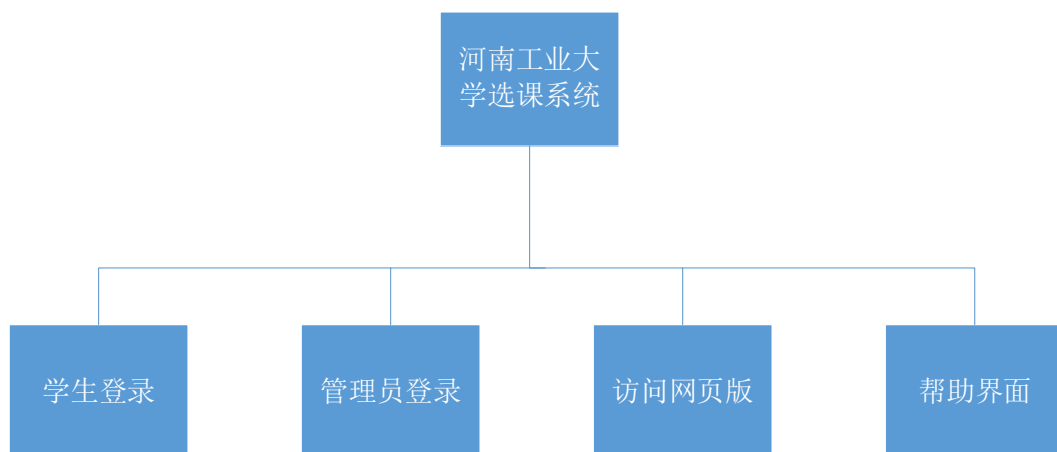


图 5.1 界面流程图



设计了一个选课管理系统，二级页面分为三种，一种为学生登录界面，一种为管理员登录界面，一种为帮助界面。学生界面需输入账号密码才可进入，管理员登录仅需密码。主界面如图 5.2 所示，学生界面如图 5.3 所示，管理员界面如图 5.4 所示，帮助界面如图 5.5 所示。登录函数代码如下所示。

```
*****欢迎使用河南工业大学选课系统*****

[1]. 学生登录
[2]. 管理员登录
[3]. 显示帮助
[4]. 访问网页版教务系统
    (更多功能请使用网页版)
[0]. 退出

请输入您的选择: _
```

图 5.2 主界面

```
-----李元昊 欢迎进入选课系统!

~~~~~学生菜单~~~~~
---[1]: 打印课程信息
---[2]: 查找课程信息
---[3]: 选择新的课程
---[4]: 删除已选课程
---[5]: 查看个人信息
---[6]: 修改登陆密码
---[0]: 注销

---请从<0-6>中选择操作类型: _
```

图 5.3 学生界面

```
~~~~~管理员菜单~~~~~
---[1]: 添加学生
---[2]: 删除学生
---[3]: 修改学生信息
---[4]: 查找学生信息

---[5]: 添加课程
---[6]: 删除课程
---[7]: 修改课程信息
---[8]: 查找课程信息

---[9]: 打印所有课程/学生信息
---[10]: 修改管理员密码
---[11]: 数据备份与恢复
---[0]: 注销

请从<0-10>中选择操作类型: _
```

图 5.4 管理员界面

请输入您的选择：3

\*\*\*\*请选择需要帮助的类型 [1]学生帮助 [2]管理员帮助 [0]返回主页

图 5.5 帮助页面

```
//学生登录
student* stuLogin(student* head)
{
    student* copy_head = head;
    char input[100];
    printf("\t\t*****请输入学号:");
    scanf("%s", input);
    while (getchar() != '\n');
    student* temp;
    temp = stunumFind(head, input);
    if (temp == NULL)

        return copy_head;
    }
    else if ((temp != NULL) && (temp->next == NULL))
    {
        return NULL;
    }
    else
        return temp->next;
}

//管理员登录
int manLogin()
{
    FILE* fp;
    if ((fp = fopen("manager.txt", "r")) == NULL)
    {
        printf("\t\t#####无法找到文件： manager.txt#####\n");
        return -1;
    }
    char key[11];
    fread(key, 10, 1, fp);
    key[10] = '\0';
    fclose(fp);
    char input[1000];
    printf("\t\t*****请输入管理员密码:");
    scanf("%s", input);
    while (getchar() != '\n');    //读入缓冲区用户输入的密码
```

```
if (strcmp(input, key) == 0)
    return 1;
else
    return 0;
}
```

5.2 添加学生信息

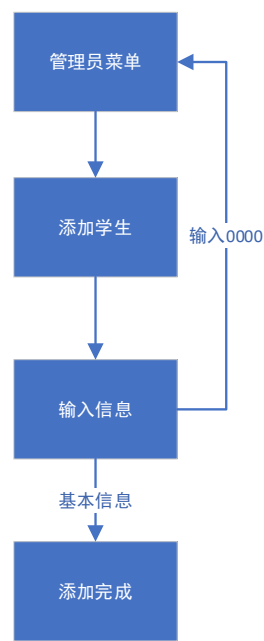


图 5.6 添加学生流程图

添加学生信息需要按照程序提示依次输入学生的基本信息，包括姓名、学号、专业班级、已得学分等信息。添加学生信息的效果如图 5.7 所示，添加后的总学生信息如图 5.8 所示。本功能主要使用单链表的插入实现，判断添加学生编号所在位置自动排序。代码如下所示：

图 5.7 添加学生

图 5.8 添加后的学生信息

```
void stuAdd(student* head)
{
    while (1)
    {
        char n[50];
        printf("\n\t\t 请输入要添加学生的学号(数字)，返回上级菜单请输入 0000: ");
        scanf("%s", n);
```

```

while (getchar() != '\n');
if (strcmp(n, "0000") == 0)
{
    system("cls");
    manmenu();
}
if (stunumFind(stu_head, n) != stu_tail){
    printf("\t\t###已存在该学生,请重新输入!####\n");
    continue;
}
//初始化学生信息
student* p = (student*)malloc(sizeof(student));
strcpy(p->num, n);
printf("\t\t 请输入学生姓名:");
scanf("%s", p->name);
while (getchar() != '\n');
printf("\t\t 请输入学生专业班级(用--隔开):");
scanf("%s", p->majorclass);
while (getchar() != '\n');
printf("\t\t 请输入当前该学生总学分:");
scanf("%lf", &p->score_all);
p->course_sum = 0;
p->score_sel = 0.0;
strcpy(p->password, p->num);
if (stu_head == NULL)
{
    p->next = NULL;
    stu_head = p;
    stu_tail = p;
}
else
{
    if (strcmp(p->num, stu_head->num) < 0)    //若该学生编号小于编号最小的学生
    {
        p->next = stu_head;
        stu_head = p;
    }
    else if (strcmp(p->num, stu_tail->num) > 0) //若该学生编号大于编号最大的学生
    {
        p->next = NULL;
        stu_tail->next = p;
        stu_tail = p;
    }
    else
        //该学生编号为中间编号

```

```

    {
        student* temp = stu_head;
        while (temp != stu_tail)
        {
            if (strcmp(p->num, temp->num) > 0)
            {
                p->next = temp->next;
                temp->next = p;
                break;
            }
        }
    }
}
printf("\n\t\t 学生信息录入成功!\n");
stuOutfile(stu_head);
}
return;
}

```

### 5.3 删除学生信息

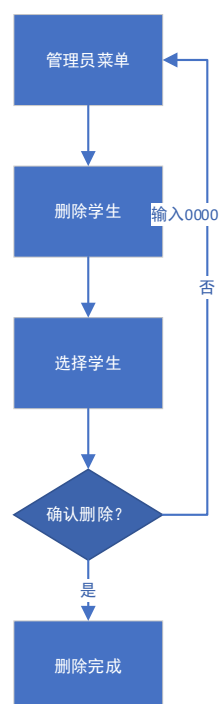


图 5.9 删除学生流程图

删除学生前的内容如图 5.8 所示，删除学生的功能图如图 5.10 所示，删除后的学生内容如图 5.11 所示。

删除学生可以根据学号或者姓名进行删除。如果找到要删除的学生，经过确认后即可删除，否则显示没有该学生。删除前查找的过程使用了程序中的查找函

数，都是由用户输入学号或者姓名，由程序来遍历链表来找到相应的学生信息，然后使用 free()函数删除该结点。代码如下所示：

图 5.10 删除学生

```
学生总数: 0  
选择操作类型: [1]打印所有学生信息 [2]打印所有课程信息 [0]返回主菜单:
```

图 5.11 删除后学生信息

```
void stuDel(student* head)
{
    while (1)
    {
        printf("\n\t\t 请输入要删除学生的学号或姓名,取消删除请输入 0000:");
        char n[50];
        scanf("%s", n);
        while (getchar() != '\n');

        if (strcmp(n, "0000") == 0)
        {
            system("cls");
            manmenu();
        }
        student* p;
        if ((n[0] >= 48) && (n[0] <= 57))           //判断 ASCII 码是否在 0~9
            p = stunumFind(stu_head, n);
        else
            p = stunameFind(stu_head, n);

        if (p == stu_tail)           //没有该学生
        {
            printf("\n\t\t####没有该学生信息!#####");
            continue;
        }

        printf("\n\t\t 确定删除%s? [1]是 [0]否:", n);
        int opt = 0;
        scanf("%d", &opt);

        if (opt == 1)
        {
            if (p == NULL)           //头结点
            {
                if (stu_head->next == NULL)
```

```

        {
            stu_tail = NULL;
        }
        student* temp = stu_head;
        stu_head = stu_head->next;
        free(temp);
        temp = NULL;
    }

    else if (p->next == stu_tail)    //尾结点
    {
        p->next = NULL;
        free(stu_tail);
        stu_tail = p;
    }

    else                                //中间结点
    {
        student* delete_point = p->next;
        p->next = delete_point->next;
        free(delete_point);
        delete_point = NULL;
    }
    printf("\n\t\t 已经删除该学生的信息! \n");
}
else
    printf("\n\t\t####未删除该学生的信息!####\n");
stuOutfile(stu_head);
}
}

```

## 5.4 修改学生信息

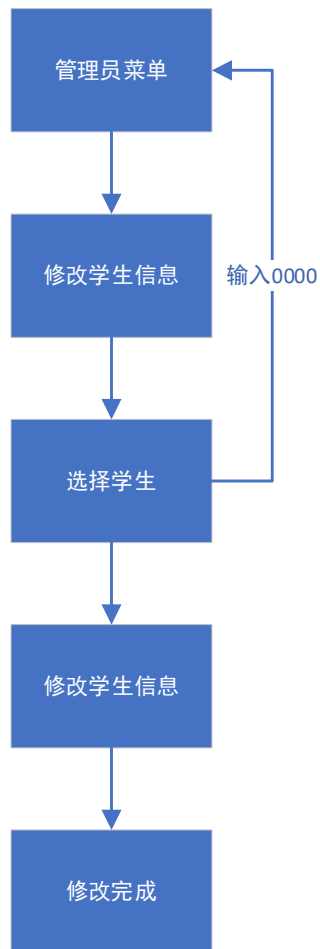


图 5.12 修改学生信息流程图

修改学生（由图 5.8）信息如图 5.13 所示，修改后的效果如图 5.14 所示，可以根据学生的学号或姓名来修改学生信息。

如果找到所要修改的学生信息，就可以修改，否则输出没有该学生。修改学生信息也主要是由查找功能定位学生的链表结点，在查找到相应的学生信息后，重新输入相应的学生信息，以实现修改学生信息的功能。具体代码如下所示：

图 5.13 修改学生

图 5.14 修改后学生信息

```
void stuModify(student* head)
{
    char n[100];
    student* find = NULL;
```



```

while (1)
{
    printf("\t\t 请输入要修改学生的学号或姓名，取消修改请输入 0000:");
    scanf("%s", n);
    while (getchar() != '\n');
    if (strcmp(n, "0000") == 0)
    {
        system("cls");
        manmenu();
    }
    if ((n[0] >= 48) && (n[0] <= 57))           //按学号查找
        find = stunumFind(head, n);
    else                                         //按姓名查找
        find = stunameFind(head, n);

    if (find == stu_tail)
    {
        printf("\n\t\t####你所查找的学生不存在！ ");
        continue;
    }
    else
    {
        char temp[50];
        if (find == NULL)
            find = head;
        else
            find = find->next;
        printf("\n\t\t 请分别输入需要修改的内容，若某一项不需要修改请输入 0");
        printf("\n\t\t 原姓名:%s 修改后的姓名: ", find->name);
        scanf("%s", &temp);
        while (getchar() != '\n');
        if (strcmp(temp, "0") != 0)
            strcpy(find->name, temp);

        printf("\t\t 原学院和班级:%s 修改后的学院和班级(中间用--隔开): ",
find->majorclass);
        scanf("%s", &temp);
        while (getchar() != '\n')
            if (strcmp(temp, "0") != 0)
                strcpy(find->majorclass, temp);

        printf("\t\t 原密码:%s 修改后的密码:", find->password);
        scanf("%s", &temp);
        while (getchar() != '\n');
    }
}

```

```

        if (strcmp(temp, "0") != 0)
            strcpy(find->password, temp);

        printf("\t\t 原总学分:%lf (增加学分请使用+,减少学分请使用-,例如-10 +10):",
find->score_all);
        scanf("%s", &temp);
        while (getchar() != '\n');
        if (strcmp(temp, "0") != 0)
        {
            int change = (int)temp[0];
            temp[0] = '0';
            double number = atof(temp);
            if (change == 43)
                find->score_all += number;
            else if (change == 45)
                find->score_all -= number;
        }
        printf("\t\t 修改完成!\n\n");
    }
    stuOutfile(stu_head);
} }

```

## 5.5 添加课程信息

添加课程信息需要按照程序提示依次输入课程的基本信息,包括编号、名字、学时、学分等信息。添加课程信息的效果如图 5.15 所示,通过单链表插入新增课程,并通过比较编号自动排序,添加后的总学生信息如图 5.16 所示。由于与添加学生代码实现大部分相同,故不再贴出代码与流程图。

```

请输入要添加课程的编号, 返回上一级菜单请输入0000: 1398794A
请输入课程名称: 数据结构
请输入课程性质:必修课
请输入开课学院:大数据学院
请输入课程总学时:54
请输入课程授课学时:44
请输入课程实验或上机学时:10
请输入课程最大容纳人数:140
请输入课程学分:3.0
请输入课程开课学期:3
课程信息录入成功!

```

图 5.15 添加课程

课程编号	课程名称	课程性质	开课学院	开课学期	总学时	授课学时	实验或上机学时	学分	已选学生
101111314A	跨文化交际	必修课	外语学院	4	54.00	54.00	0.00	2.0	该课程暂无已选学生
1398794A	数据结构	必修课	大数据学院	3	54.00	44.00	10.00	3.0	该课程暂无已选学生
101121402A	高等数学	必修课	理学院	2	118.00	108.00	10.00	6.0	该课程暂无已选学生
101131002A	美术鉴赏	必修课	设计艺术学院	2	18.00	18.00	0.00	1.0	该课程暂无已选学生
101121408A	大学物理	必修课	理学院	2	64.00	64.00	0.00	4.0	该课程暂无已选学生

图 5.16 添加后效果图示

## 5.6 删除课程信息

删除课程前的内容如图 5.16 所示，删除学生的功能图如图 5.17 所示，删除后的学生内容如图 5.18 所示。删除学生可以根据编号或者课程名字进行删除。

如果找到要删除的课程，经过确认后即可删除，否则显示没有该课程。删除前查找的过程使用了程序中的查找函数，都是由用户输入编号或者课程名称，由程序来遍历链表来找到相应的课程信息，然后使用 `free()` 函数删除该结点。

由于与删除学生代码实现大部分相同，故不再贴出代码与流程图。

```
请输入要删除课程的编号或名称，取消删除请输入0000:数据结构
确定删除数据结构? [1]是 [0]否:1
数据结构已经删除！
```

图 5.17 删除课程

101161009B	形势与政策	必修课	马克思学院	4	8.00	8.00	0.00	0.2	该课程暂无已选学生
301041303A	离散数学	必修课	大数据学院	4	30.00	30.00	0.00	3.0	该课程暂无已选学生
301041307A	操作系统原理	必修课	大数据学院	4	54.00	54.00	0.00	3.0	该课程暂无已选学生
301041308A	软件工程概论	必修课	大数据学院	4	54.00	44.00	10.00	3.0	该课程暂无已选学生
301041309A	算法分析与设计	必修课	大数据学院	4	54.00	44.00	10.00	2.5	该课程暂无已选学生

图 5.18 删除后课程内容

## 5.7 修改课程信息

修改课程（由图 5.18）信息如图 5.19 所示，修改后的效果如图 5.20 所示，可以根据课程的编号或者课程名称来修改课程信息。

如果找到所要修改的课程信息，就可以修改，否则输出没有该课程。修改课程信息也主要是由查找功能定位课程的，在查找到相应的课程信息后，重新输入相应的课程信息，以达到修改课程信息的功能。

由于与修改学生代码实现大部分相同，故不再贴出代码与流程图。

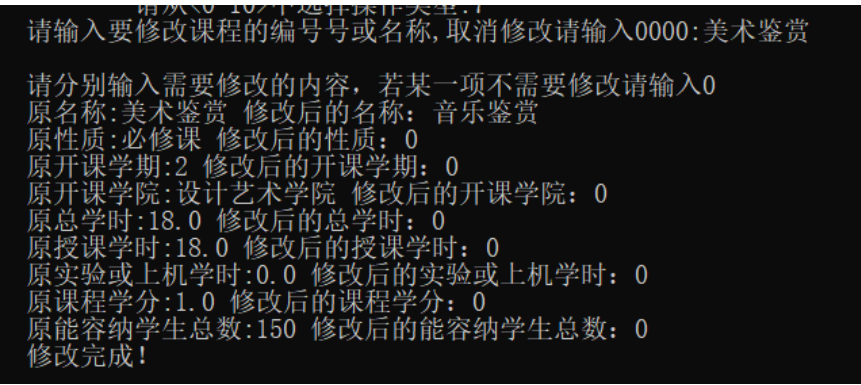


图 5.19 修改课程信息

课程编号	课程名称	课程性质	开课学院	开课学期	总学时	授课学时	实验或上机学时	学分	已选学生
101131002A	音乐鉴赏	必修课	设计艺术学院	2	18.00	18.00	0.00	1.0	

图 5.20 修改后的课程信息

5.8 查找学生或课程

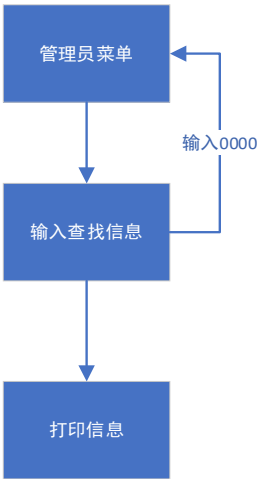


图 5.21 查找信息

管理员菜单提供了查找学生或课程的功能, 学生可以根据学号或姓名进行查找, 课程可以根据编号、课程名称、课程性质等进行查找。输入相应的信息, 如果该学生或课程存在, 就可以输出该学生或课程对应的信息, 否则显示未找到。如图 5.22 为查找学生, 5.23 为查找课程。具体代码由于两者相仿, 仅放出学生依据学号查找的代码:

图 5.22 查找学生

图 5.23 查找课程

```
student* stunumFind(student* head, char tar[])
{
    student* p, * q;
    p = head;
    q = NULL;
    while (p != NULL)
    {
        if (strcmp(tar, p->num) == 0)
            return q;
        q = p;
        p = p->next;
    }
    return q;
}

void stuFinding(student* head)
{
    while (1)
    {
        printf("\n\t\t 请输入学号或姓名进行查找，返回上级菜单请输入 0000: ");
        char n[50];
        scanf("%s", n);
        while (getchar() != '\n');
        if (strcmp(n, "0000") == 0)
        {
            system("cls");
            manmenu();
        }
        student* p_stu;
        if ((n[0] >= 48) && (n[0] <= 57))
            p_stu = stunumFind(head, n);
        else
            p_stu = stunameFind(head, n);

        if (p_stu == stu_tail){
            printf("\n\t\t#####没有该学生的信息，请重新输入! #####\n");
            continue;
        }
        if (p_stu == NULL)
            p_stu = head;
        else p_stu = p_stu->next;
        printf("\n\t\t 该学生信息如下: ");
        stuPrint(p_stu);
    }
}
```

```
}
```

## 5.9 查看所有课程与学生信息

管理员菜单中提供了查看所有课程与学生信息的功能，功能演示如图 5.24 和图 5.25 所示，如果存在学生和课程，则会显示所有学生和课程，否则会显示学生或课程数为 0。这里仅展示查看所有学生信息代码：

图 5.24 查看所有学生

```
选择操作类型：[1]打印所有学生信息 [2]打印所有课程信息 [0]返回主菜单:2
课程总数：1

-----
课程名称:面向对象程序设计
课程编号:1344991A
课程性质:必修课
开课学期:3
总学时:54.00
授课学时:36.00
实验或上机学时:18.00
学分:3.0
```

图 5.25 查看所有课程

## 5.10 修改管理员密码

本功能中，管理员默认密码为 10 个 0，每次修改管理员密码均需修改为 10 位密码存储在 manager.txt 文件中，更改密码的功能图如图 5.26 所示，代码如下所示：

```
请从<0-10>中选择操作类型:10
*****请输入原密码(10位):0000000000
*****请输入新密码(10位):1111111111
*****请再次确认新密码(10位):1111111111

密码修改成功!
```

图 5.26 修改管理员密码

```
void keyModify()
{
    printf("\t\t*****请输入原密码(10 位):");
    char input[20];
    scanf("%s", input);
    while (getchar() != '\n');
```

```

FILE* fp;
if ((fp = fopen("manager.txt", "r")) == NULL)
{
    printf("\t\t\t\t\t 无法打开文件:manager.txt\n");
    exit(0);
}
char keyOld[11];
fread(keyOld, 10, 1, fp);
keyOld[10] = '\0';
if (strcmp(keyOld, input) == 0)//修改
{
    fclose(fp);
    char new1[100];
    char new2[100];
    while (1)
    {
        printf("\t\t\t\t\t*****请输入新密码（10 位）:");
        scanf("%s", new1);
        while (getchar() != '\n');
        printf("\t\t\t\t\t*****请再次确认新密码（10 位）:");
        scanf("%s", new2);
        while (getchar() != '\n');
        if (!strcmp(new1, new2))
        {
            fp = fopen("manager.txt", "w");
            fwrite(new1, 10, 1, fp);
            fclose(fp);
            printf("\n\t\t\t\t\t 密码修改成功!\n");
            system("pause");
            system("cls");
            manmenu();
        }
        else
            printf("\t\t\t\t\t####前后密码不一致，请重新输入！####\n");
    }
}
else
{
    fclose(fp);
    printf("\t\t\t\t\t####原密码输入错误，请重新选择操作类型####\n");
}
}

```

### 5.11 数据备份与恢复

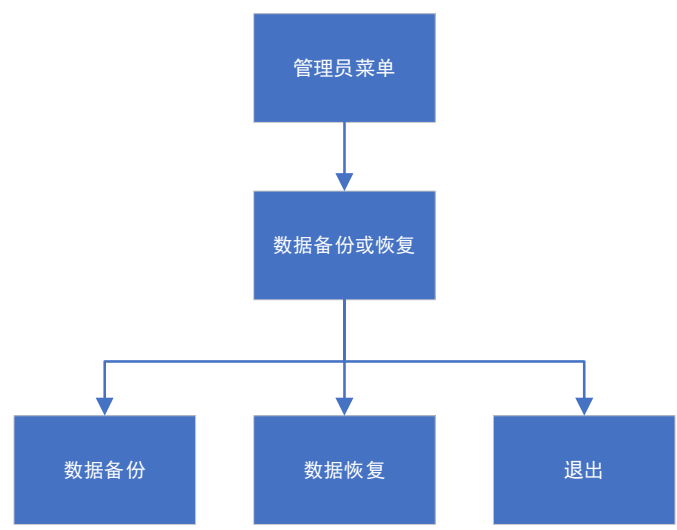


图 5.27 数据备份与恢复流程图

管理员菜单中提供了数据备份与恢复功能，处于对数据的安全私密性的保护，本程序中的所有存取文件功能全部基于二进制，数据备份将从默认存储文件拷贝一份到 backup 文件,当恢复操作时则从 backup 文件中拷贝到默认存储文件。效果图如 5.28 和 5.29 所示，代码如下所示（仅展示学生文件备份）。

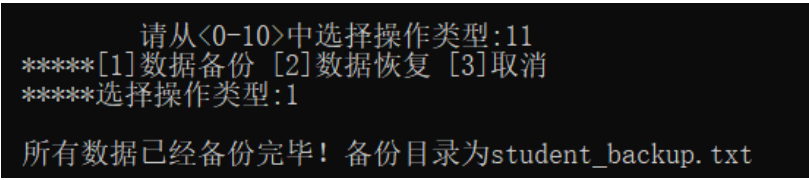


图 5.28 备份文件

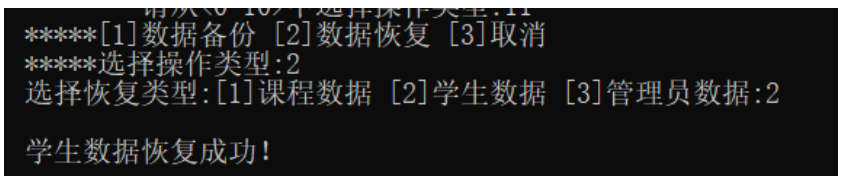


图 5.29 恢复文件

```
void backups_recover()
{
    printf("\t\t****[1]数据备份 [2]数据恢复 [3]取消\n");
    printf("\t\t****选择操作类型:");
    int n = 0;
    scanf("%d", &n);
    if (n == 1)
    {
```



```

FILE* fp_source;
FILE* fp_destination;

{//学生信息备份
    if ((fp_source = fopen("student.txt", "r")) == NULL)
    {
        printf("\t\t##学生数据备份失败：无法找到 student.txt 文件！##\n");
        exit(0);
    }
    fp_destination = fopen("student_backup.txt", "w");
    char temp;
    int count = 0;
    while (!feof(fp_source))
    {
        if (fread(&temp, 1, 1, fp_source)) count++;
    }
    rewind(fp_source);
    int i;
    for (i = 0; i < count; i++)
    {
        fread(&temp, 1, 1, fp_source);
        fwrite(&temp, 1, 1, fp_destination);
    }
    fclose(fp_destination); fclose(fp_source);
}

printf("\n\t\t所有数据已经备份完毕！备份目录为 student_backup.txt\n");
system("pause");
system("cls");
manmenu();
}
else if (n == 2)
{
    printf("\t\t选择恢复类型:[1]课程数据 [2]学生数据 [3]管理员数据:");
    int in = 0;
    scanf("%d", &in);
    while (getchar() != '\n');
    FILE* fp_source;
    FILE* fp_destination;

    else if (in == 2)
    {
        if ((fp_source = fopen("student_backup.txt", "r")) == NULL)
        {

```

```

        printf("\t\t###学生信息恢复失败！ 未能找到 student.txt\n");
        exit(0);
    }
    fp_destination = fopen("student.txt", "w");
    char temp;
    int count = 0;
    while (!feof(fp_source))
    {
        if (fread(&temp, 1, 1, fp_source))
            count++;
    }
    rewind(fp_source);
    int i;
    for (i = 0; i < count; i++)
    {
        fread(&temp, 1, 1, fp_source);
        fwrite(&temp, 1, 1, fp_destination);
    }
    fclose(fp_destination);
    fclose(fp_source);
    printf("\n\t\t 学生数据恢复成功！ \n");
    stu_head = stuInfile();
    stu_tail = endsLocate(stu_head);
}
else
    printf("\t\t###输入错误！ ###\n");
system("pause");
system("cls");
manmenu();
}
else if (n == 3)
{
    system("cls");
    manmenu();
}
else
{
    printf("\t\t###输入错误， 请重新输入!####\n");
}
}
}

```

## 5.12 学生选课功能

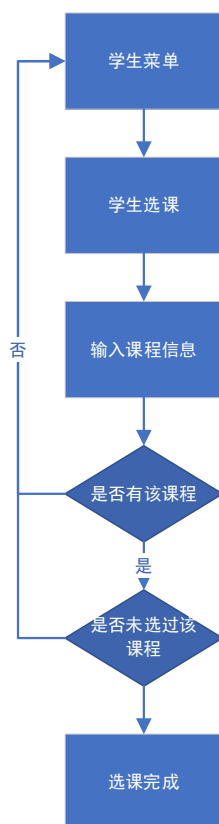


图 5.30 学生选课流程图

本程序在学生菜单界面提供了学生选课功能，若自己已经得到的学分低于应修学分，则会提示学分低于 XX 分，赶快去选课吧！输入课程库中存在的课程的编号或课程名称，则可以选择上该课，该学生的已选学分也会相应增加，学生信息中也会添加上该课程，选课功能如图 5.31 所示，选课后学生信息如图 5.32 所示，代码如下所示：

```
你的总学分少于45.6,赶快去选课吧！  
请输入课程的编号或名称进行选课，取消选课请输入0000:程序设计实践  
该课程选择成功！
```

图 5.31 学生选课

图 5.32 选课后个人信息

```
void courChoose(course* head, student* stu)
{
    if (stu->score_sel < stu->score_all)
```

```

printf("\n\t\t 你的总学分少于%f,赶快去选课吧! \n", stu->score_all);
while (1)
{
    printf("\n\t\t 请输入课程的编号或名称进行选课, 取消选课请输入 0000:");
    char input[50];
    scanf("%s", input);
    while (getchar() != '\n');
    if (strcmp(input, "0000") == 0)
    {
        system("cls");
        stumenu();
        break;
    }
    course* find = NULL;
    if ((input[0] >= 48) && (input[0] <= 57))
        find = cournumFind(head, input);
    else
        find = cournameFind(head, input);

    if (find == cour_tail)
    {
        printf("\n\t\t#####该课程不存在!请重新输入#####\n");
        continue;
    }
    else if (find == NULL)
        find = head;
    else
        find = find->next;

    if (find->stu_sum == find->stu_max)
        printf("\n\t\t#####该课程人数已满,无法选择!#####\n");
    else
    {
        int judge = 0;
        for (int i = 0; i < stu->course_sum; i++)
        {
            if (strcmp(stu->course[i], find->num) == 0)
            {
                judge = 1;
                break;
            }
        }
        if (judge == 1)
        {

```

```

        printf("\n\t\t####该课程已经选择过了!####\n");
        continue;
    }
    else
    {
        stu->course_sum++;
        strcpy(stu->course[stu->course_sum - 1], find->num);
        stu->score_sel += find->score;

        find->stu_sum++;
        strcpy(find->stu[find->stu_sum - 1], stu->num);
        printf("\n\t\t 该课程选择成功!\n");
    }
}
courOutfile(cour_head);
stuOutfile(stu_head);
}
}

```

### 5.13 学生退课功能

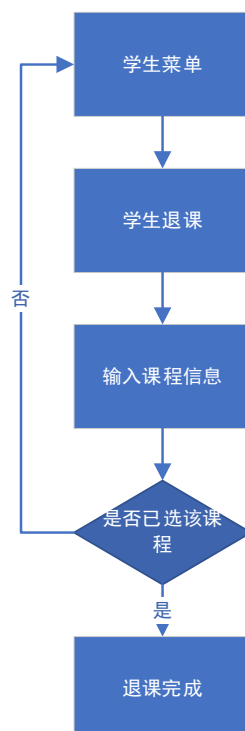


图 5.33 学生退课

与学生选课对应，学生菜单界面也提供了退课功能，学生在已选课程的基础上选择删除已选课程，输入已选课程的编号或课程名称，便可以达成退课功能，

退课功能图如图 5.34 所示，删除后学生的个人信息如图 5.35 所示，代码如下所示：

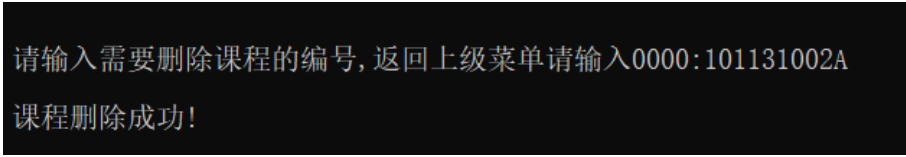


图 5.34 学生退课

图 5.35 退课后个人信息

### 5.14 学生修改密码功能

与管理员菜单相似，学生也有修改自己密码功能，学生选择修改密码功能后即可修改自己的登录密码，修改功能图如图 5.36 所示，代码与管理员修改密码相似，不再列出。

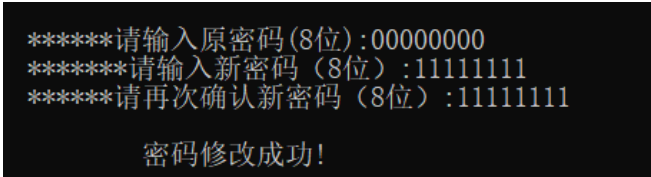


图 5.36 修改学生密码

## 6 调试分析

1. 在最开始登录管理员界面时，默认密码如何设置一直找不到合适的方法，后来在 main 函数中写了一段如下的代码

```
FILE* fp = fopen("manager.txt", "w+");
    for (int i = 0; i < 10; i++)
    {
        fputc('0', fp);
    }
    fclose(fp);
```

设置了一个默认密码，在 manager.txt 文件中利用 for 循环输入了 10 个 0，作为默认密码，但是使用之后要记得注释掉，否则会出现每次打开该程序，密码都被默认初始化为 10 个 0，导致 bug 出现。

2.在每次函数执行完功能之后，虽然返回了上一级菜单，但是没有重新刷新当前界面，导致了界面不太美观，后来想到一种较为笨拙的办法，就是调用一个函数仅显示界面内容，输入相应数字依然可以实现功能，页面较为美观一些。

3.在写入文件时，本来想要使用 `fprint()`函数写入文件，但是发现 `student` 和 `course` 结构体中有一些成员刚开始的时候无内容，写入之后依旧是乱码，之后发现无法解决该问题，索性改用 `fwrite()`函数写入二进制，同时也使文件不易读，也保证了文件内容的安全私密，以后如果还遇到这种问题，尽量使用 C++写这种程序，功能更为强大。

4. 在进行程序测试时，发现在执行完一个功能（例如排序）后，按回车键没有任何回应，陷入了循环，后来进过在网上查询，在函数的最后加入了 `return;` 语句，使之可以回到前一个界面，解决了这一问题。

## 7 测试结果

### 7.1 学生信息增删改查

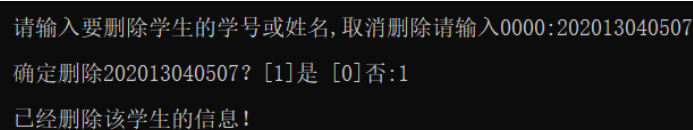
学生信息，增加（图 7.1）后进行修改（图 7.2），查询（图 7.3），最后删除（图 7.4）操作。

数据：输入学号-；学生姓名；班级--；当前学分---35.6；学院--大数据学院  
修改数据：学分--+10；

图 7.1 添加学生

图 7.2 修改学生信息

图 7.3 查询学生信息



```
请输入要删除学生的学号或姓名,取消删除请输入0000:202013040507
确定删除202013040507? [1]是 [0]否:1
已经删除该学生的信息!
```

图 7.4 删除学生

## 7.2 课程信息的增删改查

课程信息，增加（图 7.5）后进行修改（图 7.6），查询（图 7.7），最后删除（图 7.8）操作。

数据：课程编号--1398794A；课程名称--数据结构；课程性--必修课；课程总学时—54；课程授课学时--36；课程实验学时--18；最大容纳人数--120；课程学分--3.0；开课学期--3；

修改数据：原课程名称--美术鉴赏 修改--音乐鉴赏

课程编号	课程名称	课程性质	开课学院	开课学期	总学时	授课学时	实验或上机学时	学分	已选学生
101111314A	跨文化交际	必修课	外语学院	4	54.00	54.00	0.00	2.0	该课程暂无已选学生
1398794A	数据结构	必修课	大数据学院	3	54.00	44.00	10.00	3.0	该课程暂无已选学生
101121402A	高等数学	必修课	理学院	2	118.00	108.00	10.00	6.0	该课程暂无已选学生
101131002A	美术鉴赏	必修课	设计艺术学院	2	18.00	18.00	0.00	1.0	该课程暂无已选学生
101121408A	大学物理	必修课	理学院	2	64.00	64.00	0.00	4.0	该课程暂无已选学生

图 7.5 添加课程

课程编号	课程名称	课程性质	开课学院	开课学期	总学时	授课学时	实验或上机学时	学分	已选学生
101131002A	音乐鉴赏	必修课	设计艺术学院	2	18.00	18.00	0.00	1.0	

图 7.6 修改课程

课程编号	课程名称	课程性质	开课学院	开课学期	总学时	授课学时	实验或上机学时	学分	已选学生
101111314A	跨文化交际	必修课	外语学院	4	54.00	54.00	0.00	2.0	该课程暂无已选学生
1398794A	数据结构	必修课	大数据学院	3	54.00	44.00	10.00	3.0	该课程暂无已选学生
101121402A	高等数学	必修课	理学院	2	118.00	108.00	10.00	6.0	该课程暂无已选学生
101131002A	美术鉴赏	必修课	设计艺术学院	2	18.00	18.00	0.00	1.0	该课程暂无已选学生
101121408A	大学物理	必修课	理学院	2	64.00	64.00	0.00	4.0	该课程暂无已选学生

图 7.7 查看课程

101161009B	形势与政策	必修课	马克思学院	4	8.00	8.00	0.00	0.2	该课程暂无已选学生
301041303A	离散数学	必修课	大数据学院	4	30.00	30.00	0.00	3.0	该课程暂无已选学生
301041307A	操作系统原理	必修课	大数据学院	4	54.00	54.00	0.00	3.0	该课程暂无已选学生
301041308A	软件工程概论	必修课	大数据学院	4	54.00	44.00	10.00	3.0	该课程暂无已选学生
301041309A	算法分析与设计	必修课	大数据学院	4	54.00	44.00	10.00	2.5	该课程暂无已选学生

图 7.8 删除课程

## 7.3 打印所有学生与课程信息

图 7.9 查看课程信息

图 7.10 查看学生信息



## 7.4 修改密码

修改密码功能包括管理员（图 7.11）及学生（图 7.12）

数据：原密码--0000000000，现密码--1111111111；

```
      请从<0-10>中选择操作类型:10
*****请输入原密码(10位):0000000000
*****请输入新密码(10位):1111111111
*****请再次确认新密码(10位):1111111111

      密码修改成功!
```

图 7.11 管理员修改密码

```
*****请输入原密码(8位):00000000
*****请输入新密码(8位):11111111
*****请再次确认新密码(8位):11111111

      密码修改成功!
```

图 7.12 学生修改密码

## 7.5 数据备份与恢复

数据：

输入数字 1 进行数据备份；

输入数据 2 进行数据恢复；

```
      请从<0-10>中选择操作类型:11
*****[1]数据备份 [2]数据恢复 [3]取消
*****选择操作类型:1

      所有数据已经备份完毕！备份目录为student_backup.txt
```

图 7.13 备份数据

```
      请从<0-10>中选择操作类型:11
*****[1]数据备份 [2]数据恢复 [3]取消
*****选择操作类型:2
      选择恢复类型:[1]课程数据 [2]学生数据 [3]管理员数据:2

      学生数据恢复成功!
```

图 7.14 恢复数据

## 7.6 学生选课退课

学生选课退课功能演示如图 7.15 和图 7.16 所示

输入数据：输入音乐鉴赏进行选课

输入 1344991A 进行退课；

图 7.15 学生选课

图 7.16 学生退课

## 参考文献

- [1] 齐治昌等. 软件工程（第 4 版）[M]. 北京：高等教育出版社, 2019.
- [2] 何钦铭、颜辉.C 语言程序设计（第 4 版）[M].北京：高等教育出版社，2020.
- [3] 甘勇等. C 语言程序设计(第二版)[M]. 北京：中国铁道出版社有限公司，2020.
- [4] 严蔚敏等.数据结构(C 语言版)第 2 版[M]. 北京：人民邮电出版社, 2019.
- [5] 王新等. C 语言课程设计[M]. 北京：清华大学出版社, 2009.