

课 程 设 计 报 告

设计名称 学生成绩管理系统

班	级	<u>软件 72</u>
学	号	<u>2016010539</u>
姓	名	<u>王世杰</u>

2018 年 9 月 3 日

目 录

1.	系统需求分析	1
2.	总体设计	2
3.	详细设计	4
4.	系统调试	10
5.	测试结果与分析	11
6.	总结	20
7.	附录：源程序清单	21
1.1.	class.h	22
1.2.	classfunction.h	24
1.3.	main.cpp	29
1.4.	startpage.cpp	29
1.5.	page1.cpp	34
1.6.	page2.cpp	36
1.7.	page3.cpp	43
1.8.	page4.cpp	46
1.9.	page5.cpp	51



1. 系统需求分析

学生成绩管理系统记录了学生大一的各科成绩及每门课程的成绩，它包括：学期、姓名、班级（汉字）、学号、科目、学分、分数，试设计学生成绩管理系统，使之能提供以下功能：

1. 录入、修改学生的各科的成绩信息：从键盘输入数据（提示：为避免重复从键盘输入数据，测试时可将数据存储在文件中，利用输入重定向功能读入），输入格式为：学期 姓名 班级 学号 科目 学分 分数。每行一条记录。并在输入不合法记录时报错。若该信息已存在则覆盖原信息。系统根据分数得到该信息对应的评级、学分绩及是否挂科。
 - a) 例如：大一上学期 王世杰 无七六 2016010539 工程制图 2 87
 - b) 此时系统得到该信息评级为 B+，学分绩为 3.3，未挂科。
2. 查询某个学生某学期或整个学年各门课的成绩：按照分数降序排列，相同的课程按学分降序排列，并提供该课程的评级和学分绩。同时给出该时间段平均学分绩。
3. 统计某课程所有学生总成绩情况，按照分数（总学分绩）降序排列，相同的按学号升序排列。
4. 查询某课程所有学生成绩，按照分数（总学分绩）降序排列，相同的按学号升序排列。
5. 统计学生挂科数并按降序排列，相同的按姓名升序排列。
6. 系统以菜单方式工作。（所谓菜单指用户可以自由选择所要执行的功能。学生可以通过以上功能录入信息、修改信息、查询信息、整理统计出所要了解的信息，除了要实现上述的基本功能之外，本系统还应该在细节上下工夫，使用户使用方便）

2. 总体设计

大一学生成绩管理系统包含五个大的功能，分别是：录入、修改学生成绩，查询某学生成绩，查询某课程所有学生成绩，查询所有学生总成绩情况，查询挂科情况。每个功能对应一个界面，每个界面均有操作提示，并可返回之前的界面。学生的成绩信息主要包含学期、姓名、班级（汉字）、学号、科目、学分、分数，以及根据分数转换得到的学分绩、评级和是否挂科。信息存储基于文件操作。

打开系统首先是进入欢迎界面，打出欢迎使用的字样。在欢迎界面，系统会自动根据存储信息的文件统计文件中的信息条数，创建录入信息类对象数组来存放最新版本的信息并将其写入文件。同时制作所有学生的 `studentGrade` 类对象数组。

紧接着进入主界面，有 6 个选项，分别是进入对应 5 个功能的界面及结束界面。

在录入、修改学生成绩界面（`page1`），根据系统提示一次性输完一整条信息。若输入信息有误，如学期不正确（不为大一上学期或大一下学期），则报错，提示重新输入。系统根据信息的学期、姓名、科目信息判断是否为新信息，若为新信息则覆盖原信息。在退出该界面时，更新文件并统计文件中的信息数，创建录入信息类对象数组来存放最新版本的信息。同时制作所有学生的 `studentGrade` 类对象数组。

在查询某学生成绩界面（`page2`），用户首先输入要修改的学生学号，再选择要查询的学期。如果系统中没有该学生的相关信息，则系统会给相关提示。如果系统中有该学生的相关信息则按照分数降序排列，相同的课程按学分降序排列，并提供该课程的评级和学分绩。

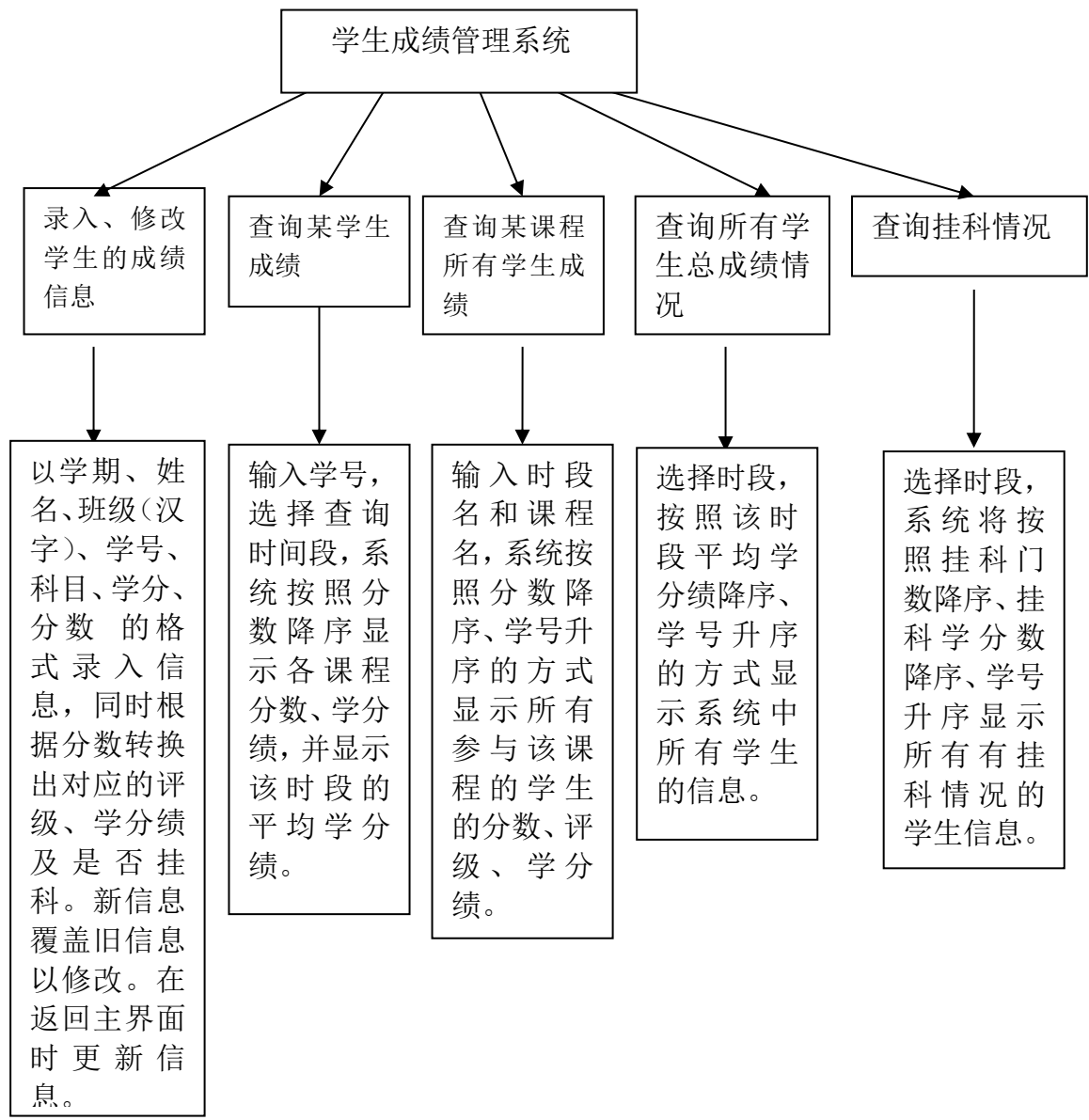
在查询某课程所有学生成绩情况界面（`page3`），用户先输入要查询的学期和课程名，系统检索判断是否存在该课程。若不存在则提示系统中无此课程。若存在显示该课程所有学生成绩，按分数降序排列，分数相同则按学号升序排列。

在查询所有学生总成绩情况界面（`page4`），用户先根据提示选择查询的学期。显示该学期所有学生成绩，按平均学分绩降序排列，相同则按学号升序排列。

在查询挂科情况界面（`page5`），用户先根据提示选择查询的学期。显示该学期所有出现挂科情况的同学的挂科学分和挂科数目，按挂科数降序排列。相同按挂科学分降序排列。再相同按学号升序排列。

在结束界面（page6），系统会自动清空所有动态内存、关闭文件，同时打出感谢使用本系统的字样，希望给用户最好的体验。

大一学生成绩管理系统中功能模块图：



3. 详细设计

大一学生成绩管理系统中五个类的类层次图为：

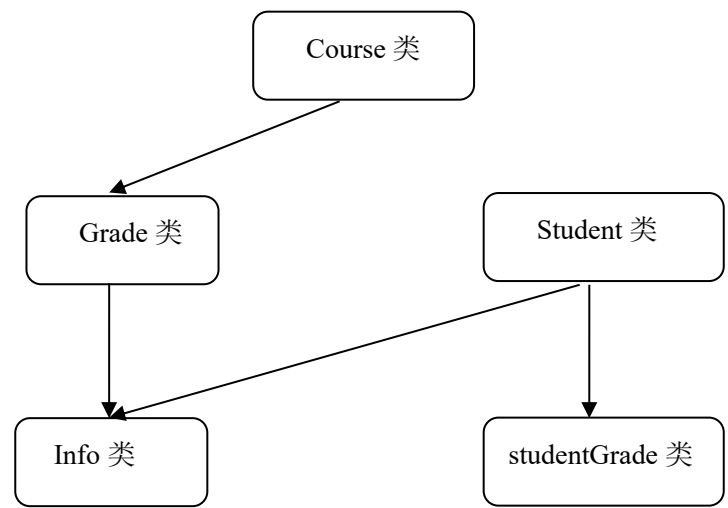


图 2 大一学生成绩管理系统中五个类的类层次图

大一学生成绩管理系统中各功能模块的实现：

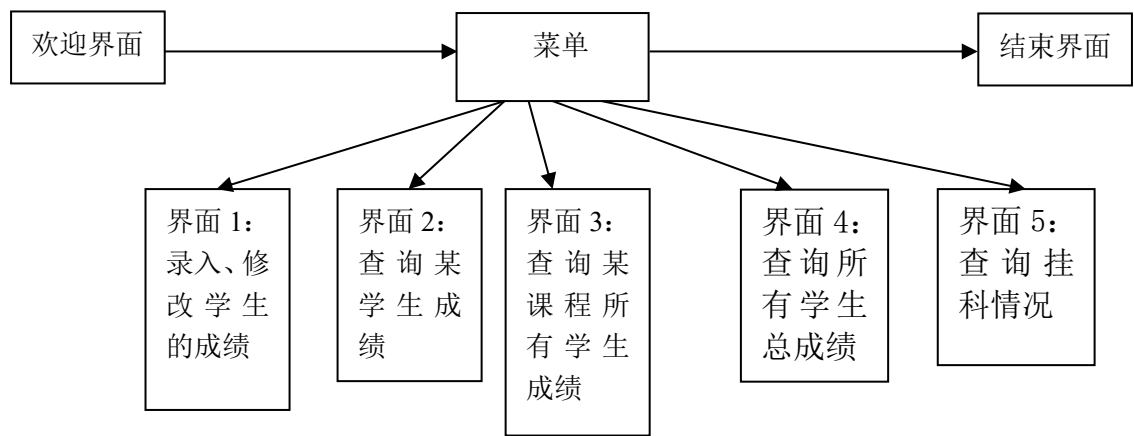


图 3 学生成绩管理系统中菜单函数的功能图

1、界面 1：录入、修改学生的成绩

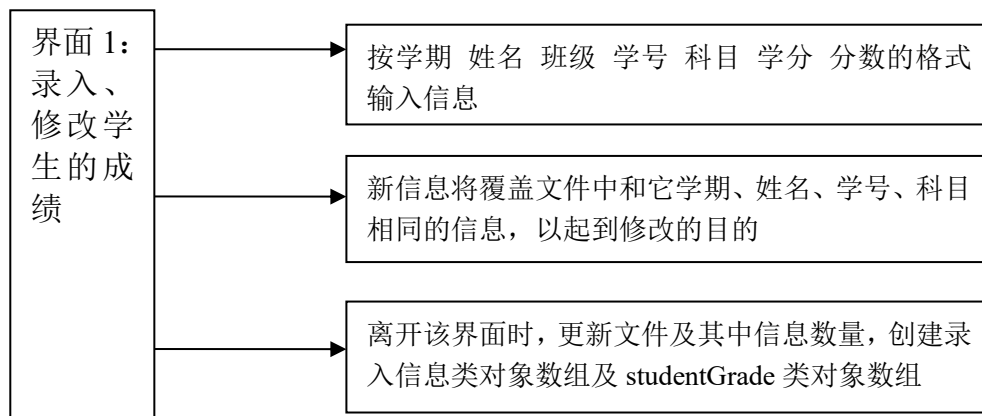


图 4 录入、修改学生的成绩界面功能图

2、界面 2：查询某学生成绩

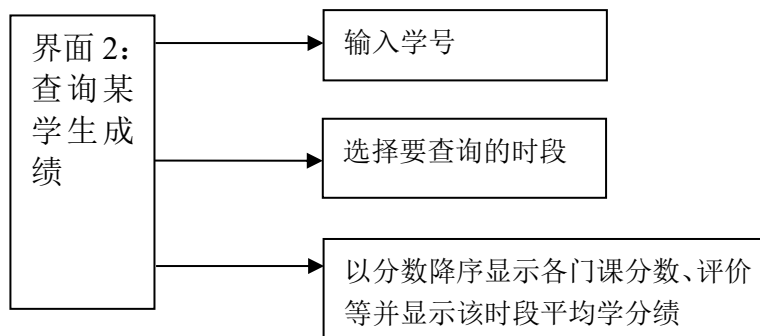


图 5 查询某学生成绩界面功能图

3、界面 3：查询某课程所有学生成绩

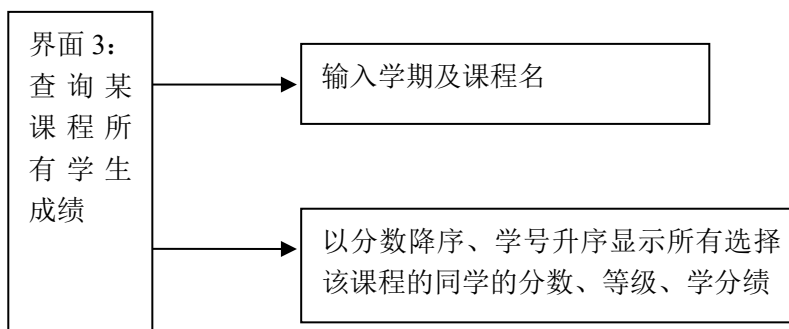


图 6 查询某课程所有学生成绩界面功能图

4、界面 4：查询所有学生总成绩

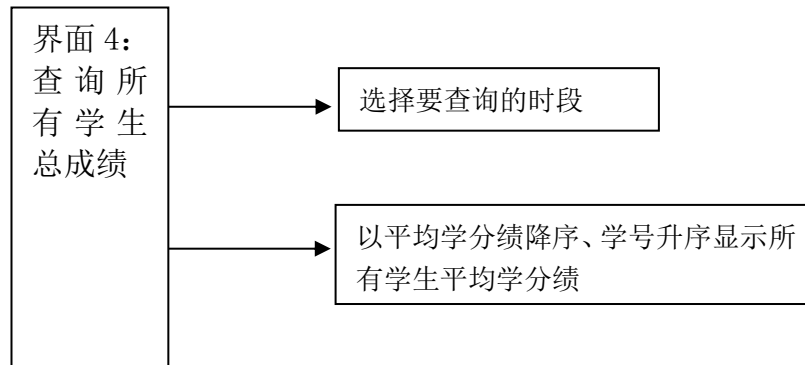


图 7 查询所有学生总成绩功能图

5、界面 5：查询挂科情况

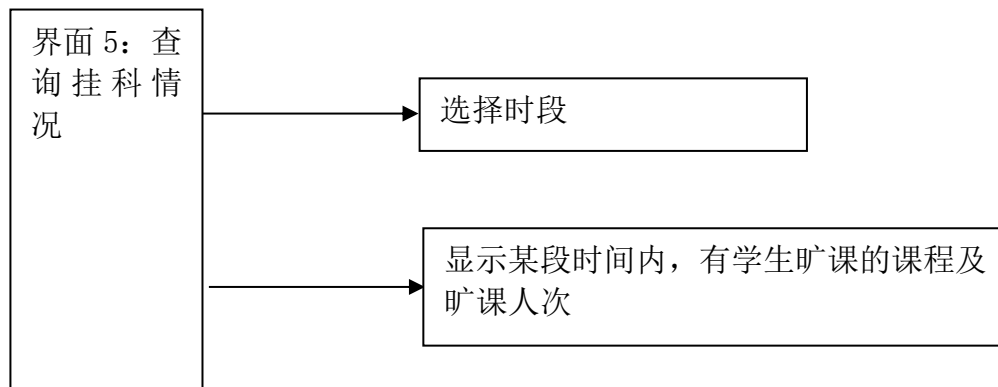
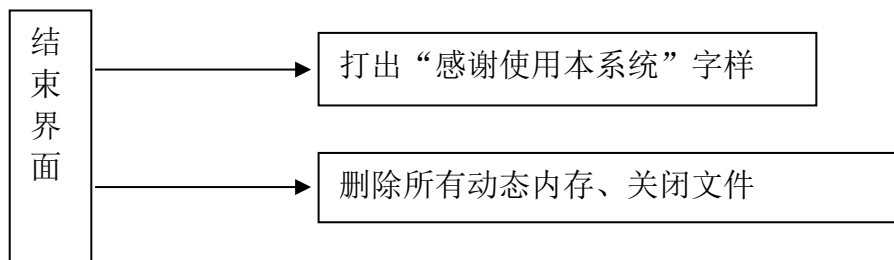


图 8 查询挂科情况功能图

6、欢迎界面



7、结束界面



大一学生成绩管理系统中五个类的 UML 图为：

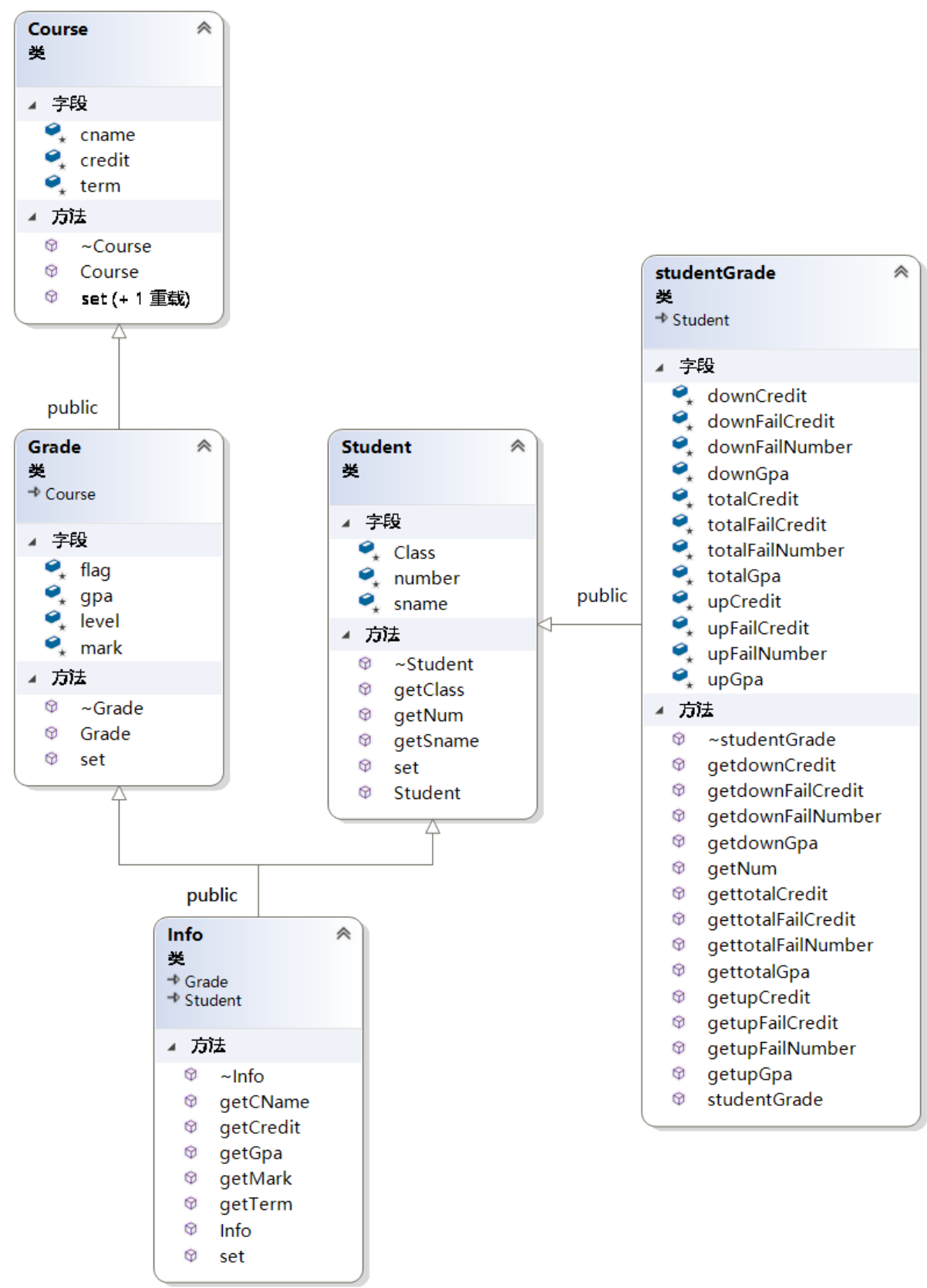


图 9 大一学生成绩管理系统中五个类的 UML 图（类名/保护成员/公有成员）

4. 系统调试

程序编写完成后，我进行了调试。调试过程中，出现了以下三个主要问题：

1. 以姓名为关键词不能对重名现象进行很好的结局。编一开始我才去以姓名为关键字进行搜索，在室友的提醒下，我将程序改为用学号为关键词进行搜索，解决了这一问题。

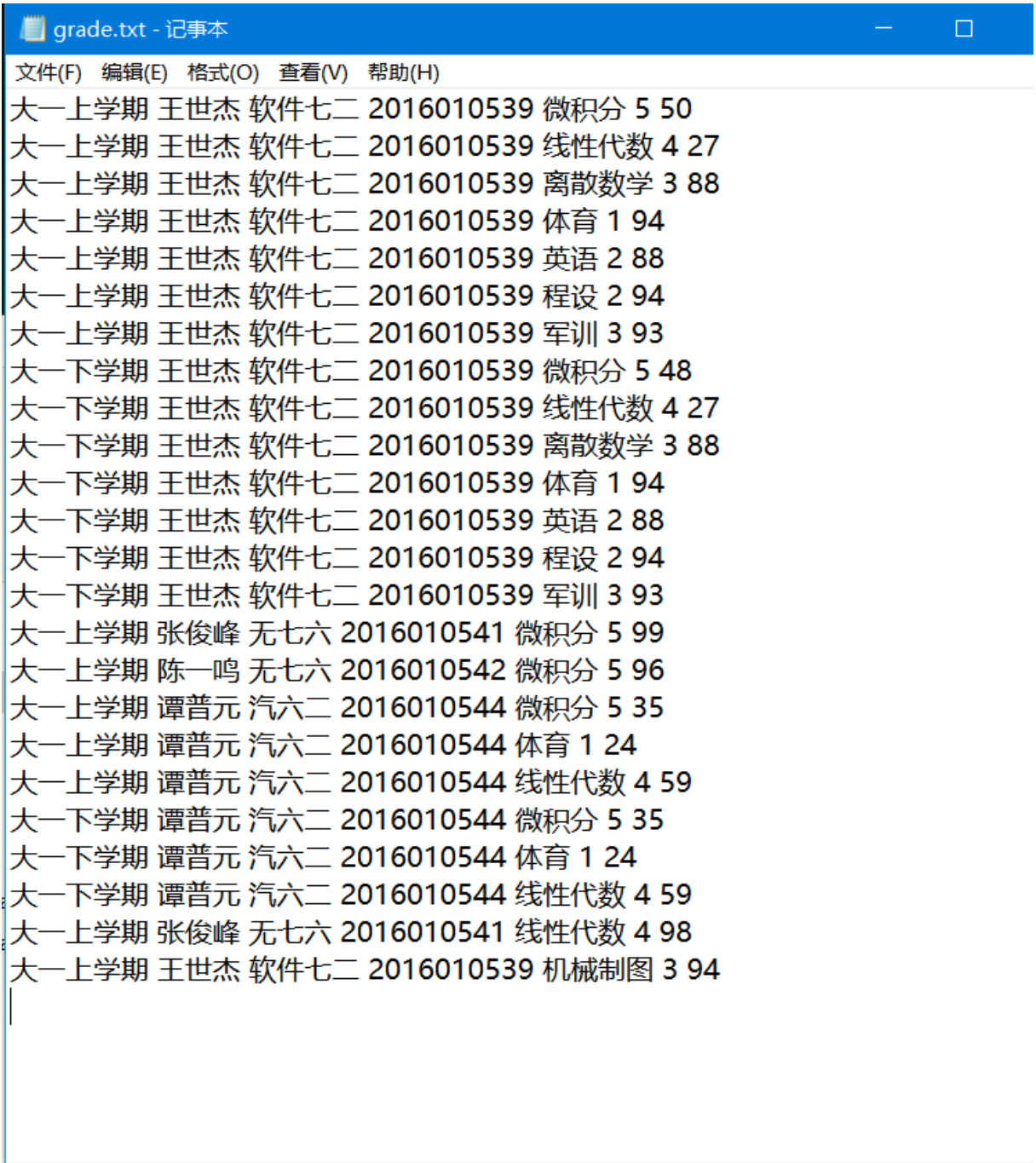
2. 系统应对错误输入的能力严重不足。最初版本的程序对输入没有任何检测，经常会出现原本应输入汉字或汉字的地方我输入一堆字母，系统没有任何提示，并将错误输入写入了文件中，导致在检索、排序的时候以及显示所有学生成绩的时候，出现重大问题。于是我在所有需要输入的地方加入了检测功能，一旦输入不合要求就出现提示并重新回到输入界面。

3. 系统应对大量误操作时会崩溃。作为开发者我对程序较为了解，输入等操作比较合法，也比较舒缓。当我把系统交给我室友检测时，他在短时间内随便按下键盘，这就导致在主界面短时间内出现大量误操作。当时我将 `update` 部分放在了进入主界面时执行，而主界面出现误操作后会重新进入主界面，这就意味着如果主界面短时间出现大量误操作，就需要短时间内执行多次 `update`，而 `update` 部分需要执行扫描整个文件、排序、重新写入文件、建立 `studentGrade` 对象动态数组等多个步骤，执行需要时间相对较长。这导致系统无法在短时间内处理多次 `update`，所以系统会崩溃。由于 `update` 函数无法更改，我就尝试从其他角度解决这个问题。我想到 `update` 函数负责更新，而只有在信息发生变化时才需要进行更新。在本系统中只有界面 1 会对信息进行更改，于是我将 `update` 函数放在了界面 1 返回主界面的时候执行，有考虑到可能使用者不会对信息进行修改，又在欢迎界面开始处执行一次 `update` 以建立 `studentGrade` 对象动态数组。这样就解决了主界面无法承受大量误操作的问题。

这种发现问题并解决问题的过程对我的帮助很大，通过对程序的设计和测试，我意识到开发一个成熟的系统需要非常的耐心以及不停的完善，后期测试也必不可少。这次程序设计真的让我在程序调试方面有了很大的进步。

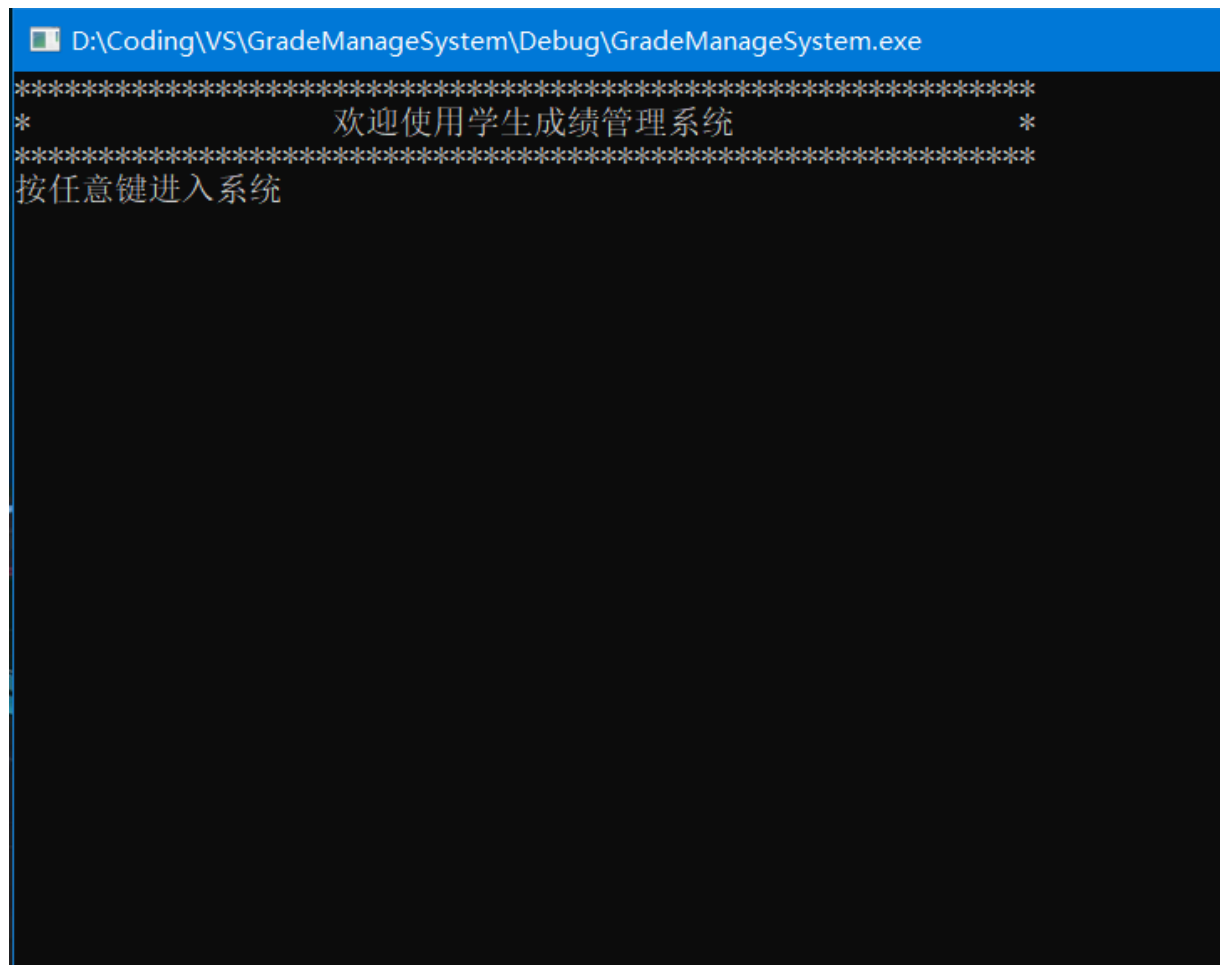
5. 测试结果与分析

本程序的测试数据文件是 grade.txt, 测试结果截图如图所示。

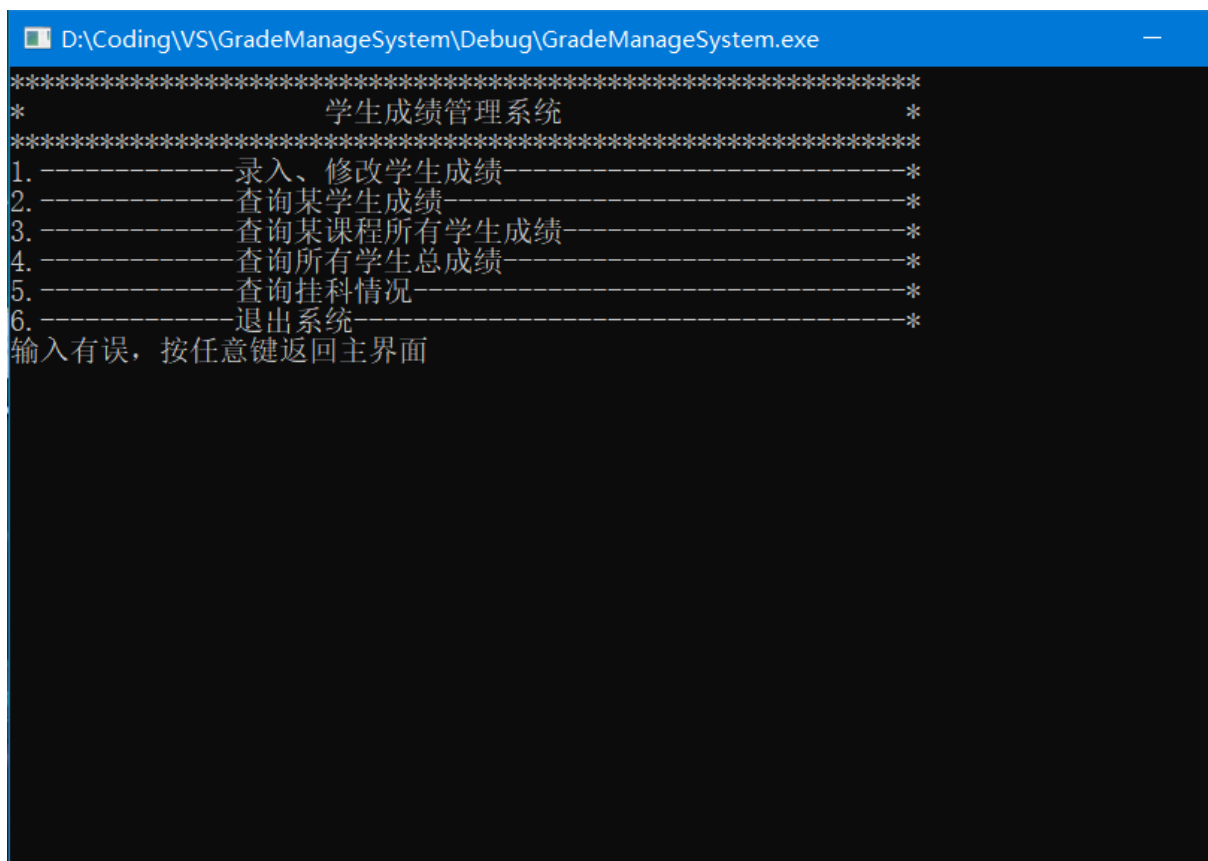


```
grade.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
大一上学期 王世杰 软件七二 2016010539 微积分 5 50
大一上学期 王世杰 软件七二 2016010539 线性代数 4 27
大一上学期 王世杰 软件七二 2016010539 离散数学 3 88
大一上学期 王世杰 软件七二 2016010539 体育 1 94
大一上学期 王世杰 软件七二 2016010539 英语 2 88
大一上学期 王世杰 软件七二 2016010539 程设 2 94
大一上学期 王世杰 软件七二 2016010539 军训 3 93
大一下学期 王世杰 软件七二 2016010539 微积分 5 48
大一下学期 王世杰 软件七二 2016010539 线性代数 4 27
大一下学期 王世杰 软件七二 2016010539 离散数学 3 88
大一下学期 王世杰 软件七二 2016010539 体育 1 94
大一下学期 王世杰 软件七二 2016010539 英语 2 88
大一下学期 王世杰 软件七二 2016010539 程设 2 94
大一下学期 王世杰 软件七二 2016010539 军训 3 93
大一上学期 张俊峰 无七六 2016010541 微积分 5 99
大一上学期 陈一鸣 无七六 2016010542 微积分 5 96
大一上学期 谭普元 汽六二 2016010544 微积分 5 35
大一上学期 谭普元 汽六二 2016010544 体育 1 24
大一上学期 谭普元 汽六二 2016010544 线性代数 4 59
大一下学期 谭普元 汽六二 2016010544 微积分 5 35
大一下学期 谭普元 汽六二 2016010544 体育 1 24
大一下学期 谭普元 汽六二 2016010544 线性代数 4 59
大一上学期 张俊峰 无七六 2016010541 线性代数 4 98
大一上学期 王世杰 软件七二 2016010539 机械制图 3 94
```

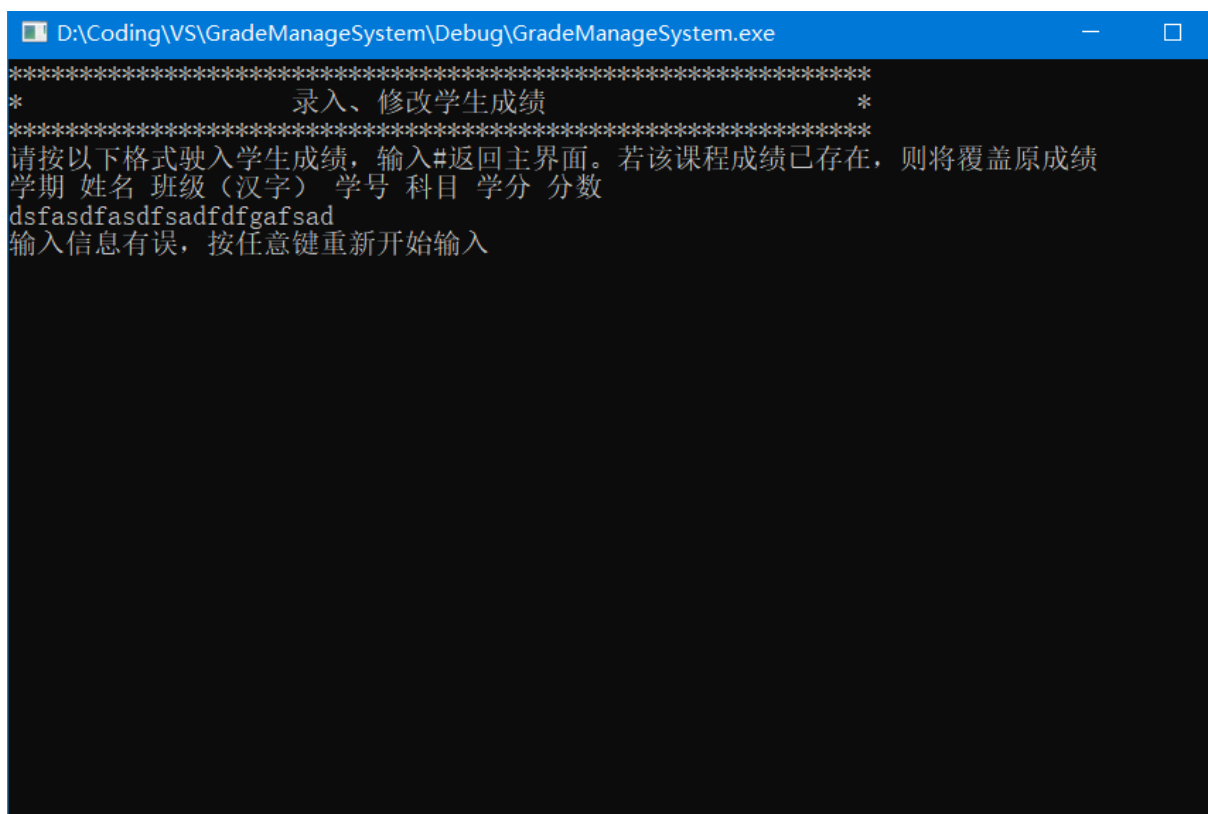
原始文件截图



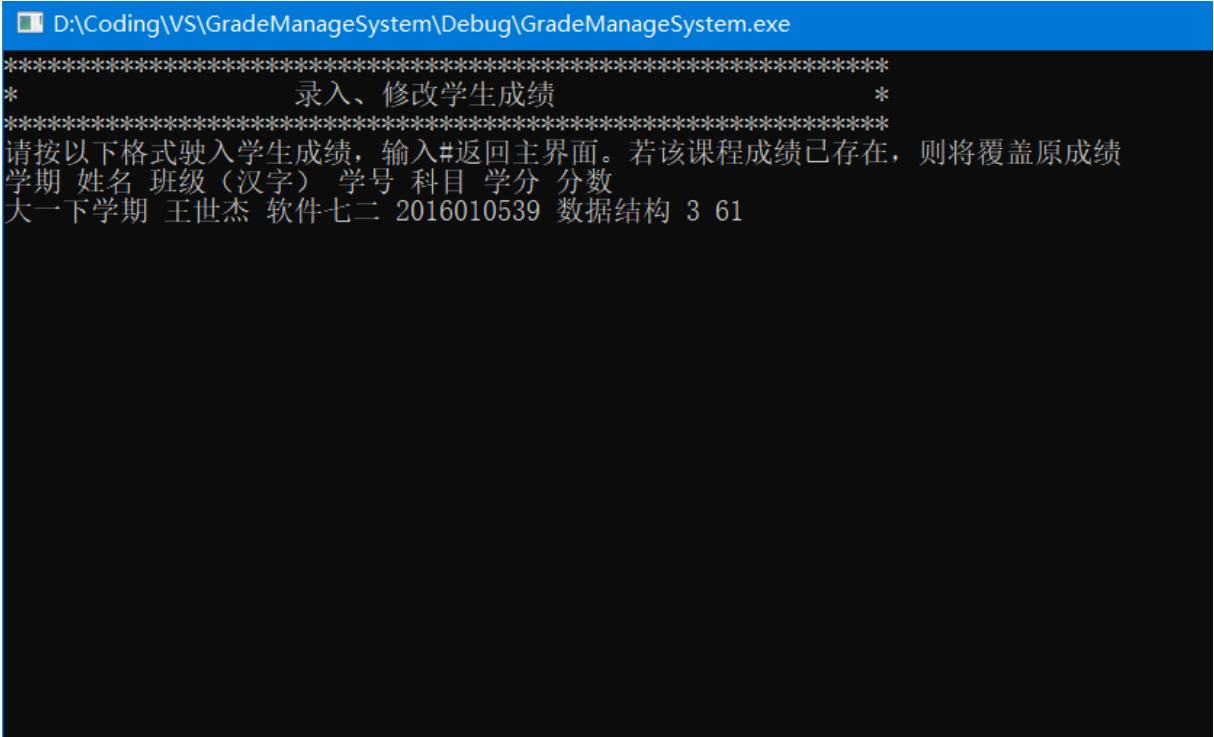
欢迎界面



主界面及其错误操作反馈演示

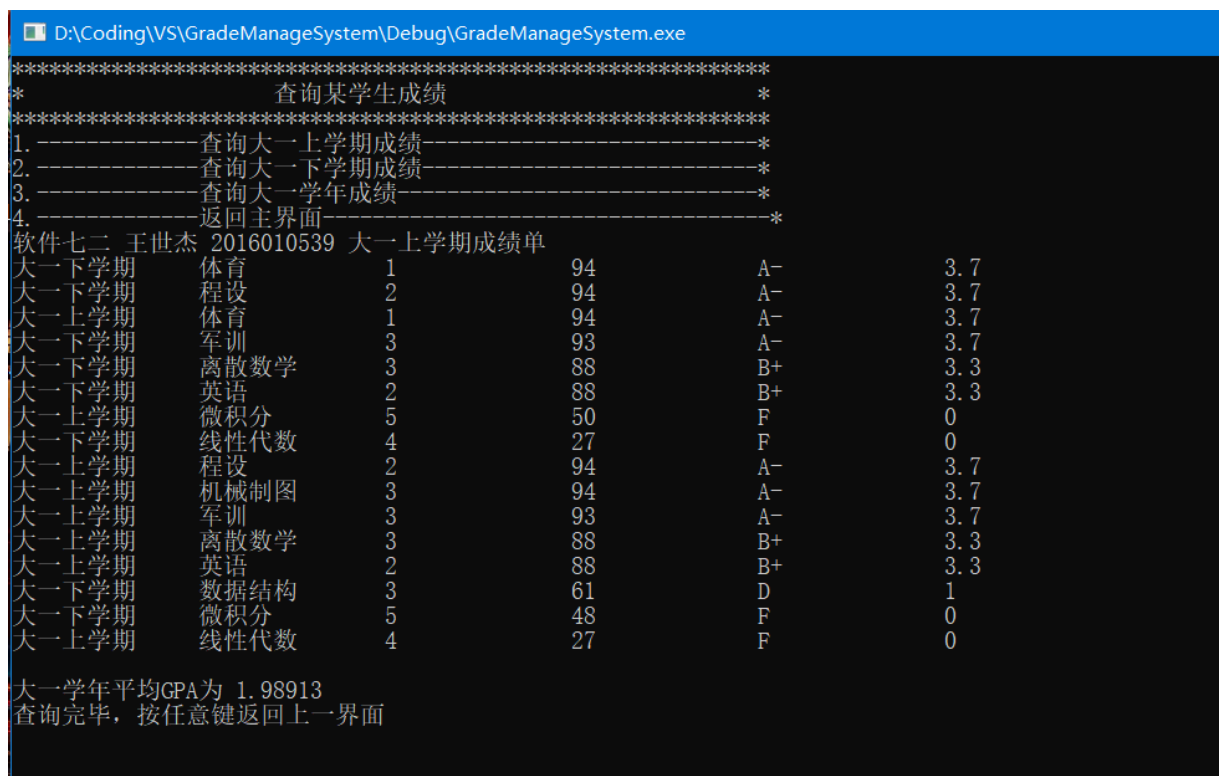
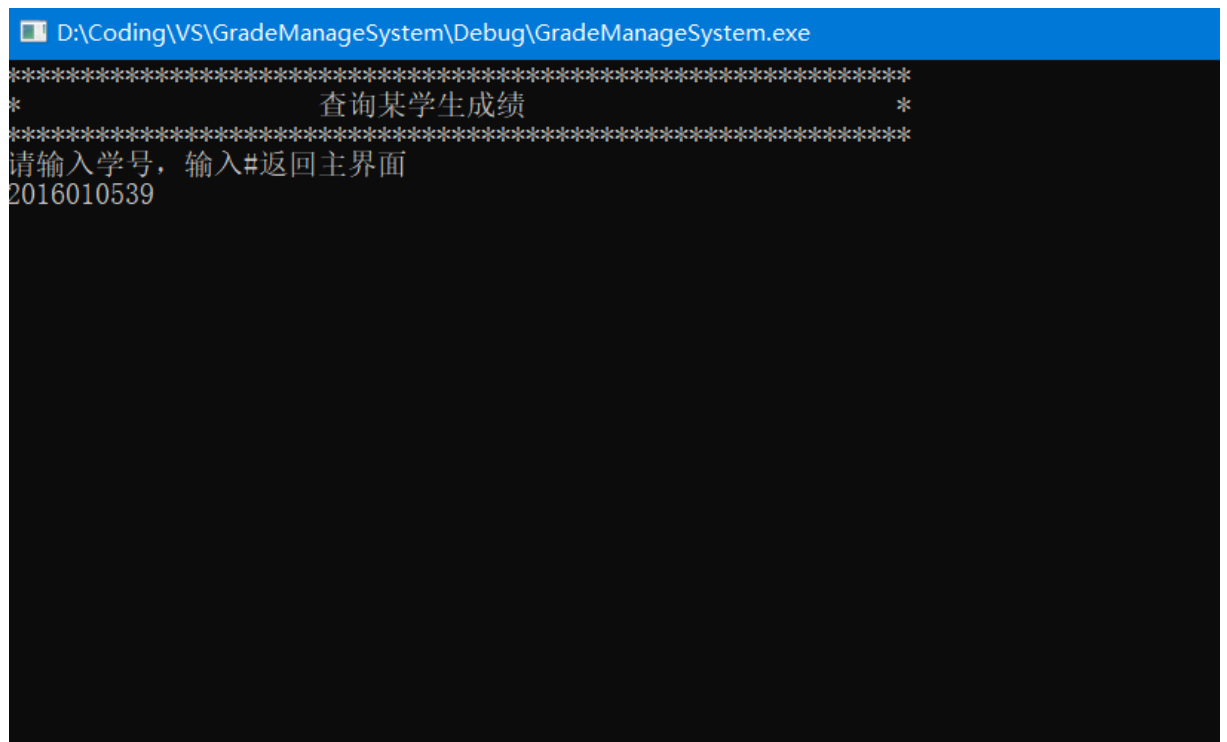


界面 1：错误操作反馈演示

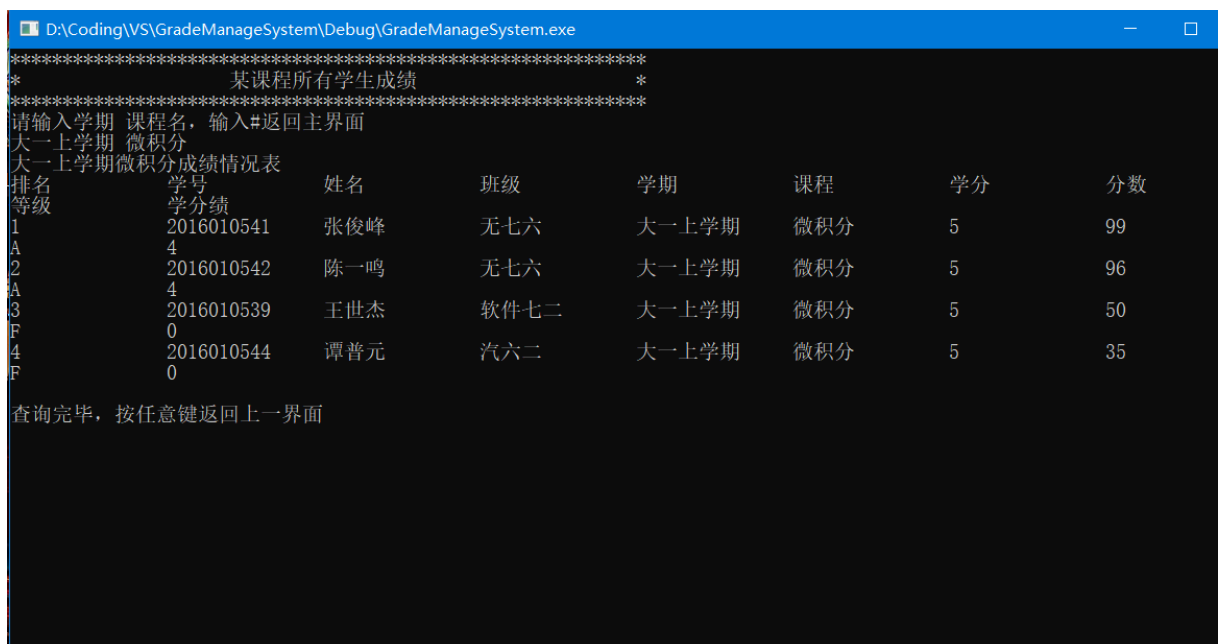
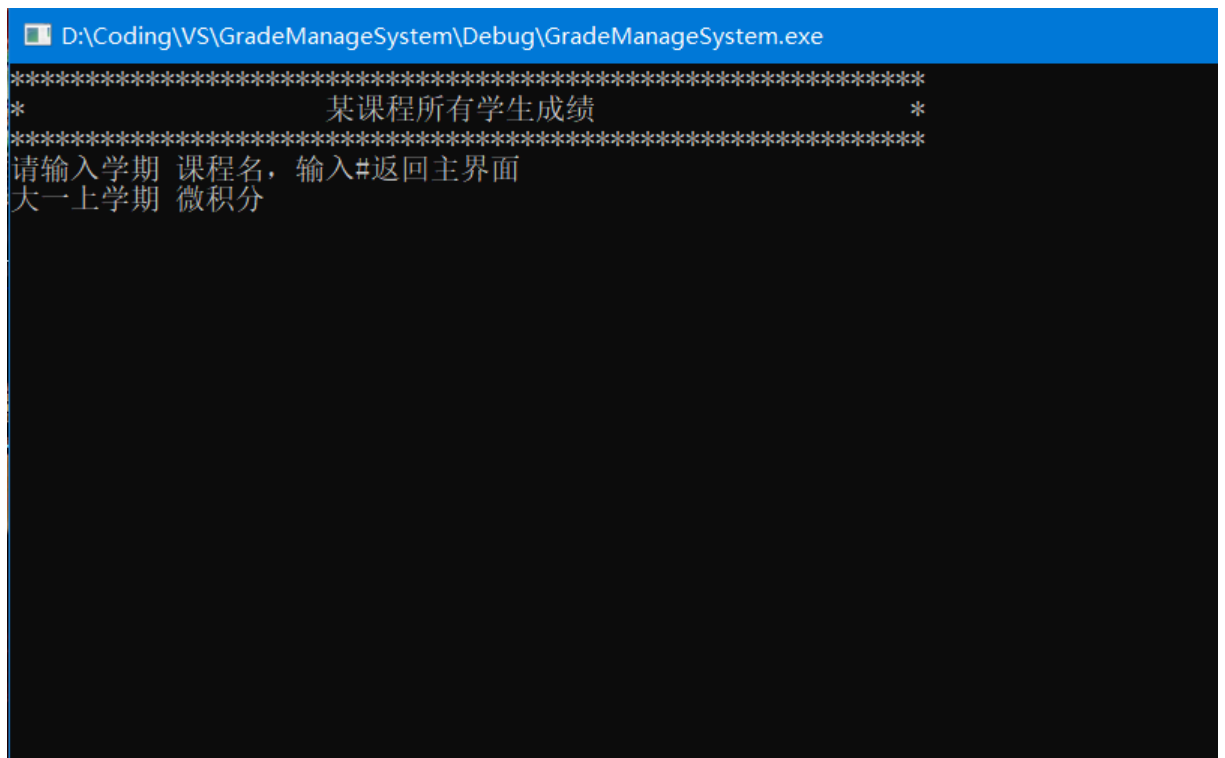


界面 1：录入修改学生成绩操作演示

大一上学期 王世杰 软件七二 2016010539 微积分 5 50
大一上学期 王世杰 软件七二 2016010539 线性代数 4 27
大一上学期 王世杰 软件七二 2016010539 离散数学 3 88
大一上学期 王世杰 软件七二 2016010539 体育 1 94
大一上学期 王世杰 软件七二 2016010539 英语 2 88
大一上学期 王世杰 软件七二 2016010539 程设 2 94
大一上学期 王世杰 软件七二 2016010539 军训 3 93
大一下学期 王世杰 软件七二 2016010539 微积分 5 48
大一下学期 王世杰 软件七二 2016010539 线性代数 4 27
大一下学期 王世杰 软件七二 2016010539 离散数学 3 88
大一下学期 王世杰 软件七二 2016010539 体育 1 94
大一下学期 王世杰 软件七二 2016010539 英语 2 88
大一下学期 王世杰 软件七二 2016010539 程设 2 94
大一下学期 王世杰 软件七二 2016010539 军训 3 93
大一上学期 张俊峰 无七六 2016010541 微积分 5 99
大一上学期 陈一鸣 无七六 2016010542 微积分 5 96
大一上学期 谭普元 汽六二 2016010544 微积分 5 35
大一上学期 谭普元 汽六二 2016010544 体育 1 24
大一上学期 谭普元 汽六二 2016010544 线性代数 4 59
大一下学期 谭普元 汽六二 2016010544 微积分 5 35
大一下学期 谭普元 汽六二 2016010544 体育 1 24
大一下学期 谭普元 汽六二 2016010544 线性代数 4 59
大一上学期 张俊峰 无七六 2016010541 线性代数 4 98
大一上学期 王世杰 软件七二 2016010539 机械制图 3 94
大一下学期 王世杰 软件七二 2016010539 数据结构 3 61



界面 2：查询某学生成绩操作演示



界面 3：查询某课程所有学生成绩操作演示

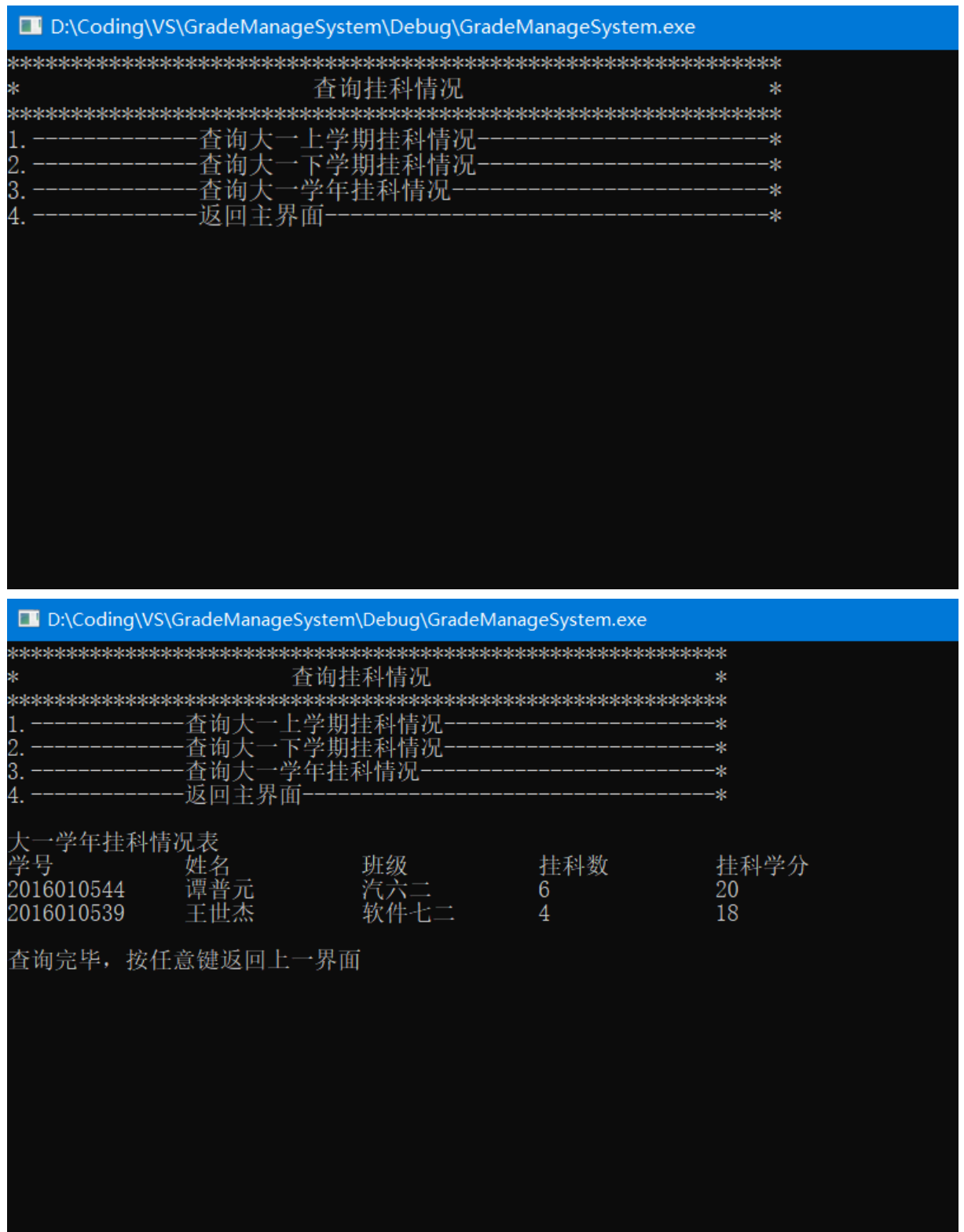
```
D:\Coding\VS\GradeManageSystem\Debug\GradeManageSystem.exe
*****
*                               *
*          查询所有学生总成绩          *
*****
1. -----查询大一上学期成绩-----*
2. -----查询大一下学期成绩-----*
3. -----查询大一学年成绩-----*
4. -----返回主界面-----*
```

```
D:\Coding\VS\GradeManageSystem\Debug\GradeManageSystem.exe
*****
*                               *
*          查询所有学生总成绩          *
*****
1. -----查询大一上学期成绩-----*
2. -----查询大一下学期成绩-----*
3. -----查询大一学年成绩-----*
4. -----返回主界面-----*

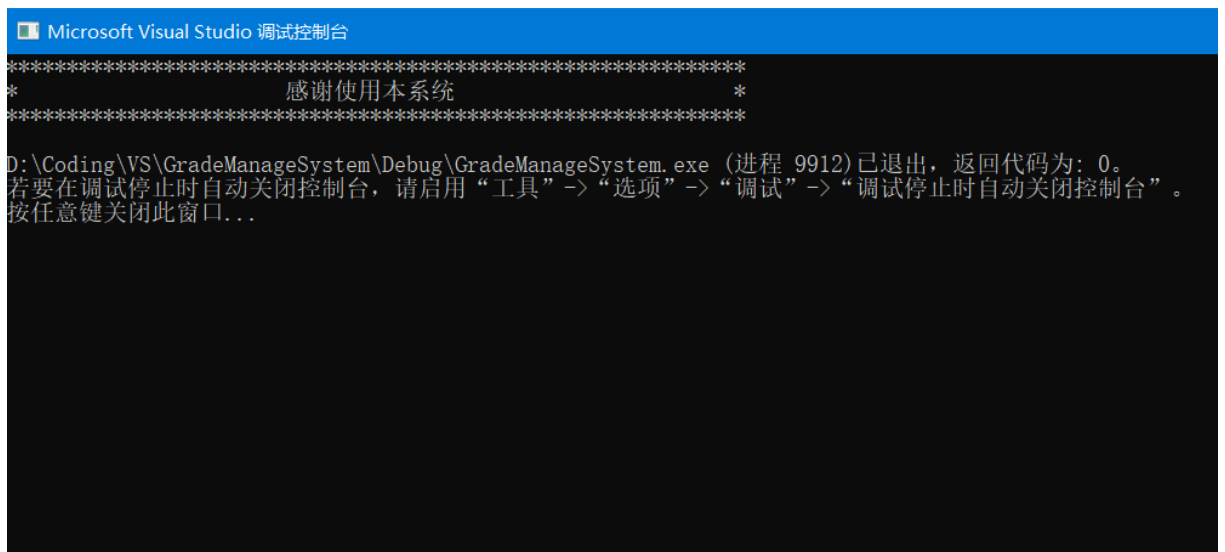
大一学年总成绩情况表
排名      学号      姓名      班级      平均学分绩
1          2016010541    张俊峰    无七六      4
2          2016010542    陈一鸣    无七六      4
3          2016010539    王世杰    软件七二    2.05814
4          2016010544    谭普元    汽六二      0

查询完毕，按任意键返回上一界面
```

界面 4：查询所有学生总成绩操作演示



界面 5：查询挂科情况操作演示



```
Microsoft Visual Studio 调试控制台
*****
*                               *
*           感谢使用本系统           *
*                               *
*****
D:\Coding\VS\GradeManageSystem\Debug\GradeManageSystem.exe (进程 9912) 已退出，返回代码为: 0。
若要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调试停止时自动关闭控制台”。
按任意键关闭此窗口...
```

界面 6：退出界面

这次大作业总得来说完成的还算顺利，主要原因是老师要求在正式做大作业前先做一份选题报告。这份选题报告帮了我大忙。它让我在一开始就想好了程序的架构，如：需要实现的功能及如何实现、如何实现题目要求的四个类等，同时了解了工作量方便安排计划。这让我有了一个非常顺利的开始。但在开始编写之后仍遇到了不小的麻烦。

第一个问题是如何存放代码。我一开始的想法是头文件放类和类的函数，之后一个文件存放界面，一个文件存放界面的操作。但后来发现这样做存放操作的文件代码量太大，而且各个界面分别的不够鲜明，不利于编译和工作的进行。于是我就改成了每个界面对应一个文件，通过主界面统一调用，解决了上面的问题，有力地推进了工作。

第二个问题是规划不到位，在动手编程前没有确定算法，经常是编写到一大半想到更好的算法、更简单的代码，但是已经无法更改了，导致代码不够简洁，有很多效率很低的算法以及很多重复代码。这让我明白在开始写代码前，一定要将每一个细节都想明白，甚至要动笔去写下来，而不是脚踩西瓜皮写到哪是哪。

第三个问题是全局变量、全局函数的使用。我对“全局”的理解非常不到位。开始编程时我连如何使用都不知道。了解如何使用之后我却未对全局变量做好规划，导致了变量乱取名、多声明了一些全局变量等问题。在编程前一定要将全局变量规划好，不能编到一半发现需要了再去声明。

第四个问题是不注意细节。经常出现 i、j，1、l，<、>，=、==，&、&&之类的字母、数字、符号打错，导致程序崩溃，浪费了大量时间在调试上。这个问题必须要引起重视！

希望能吸取这次大作业的经验教训，为以后的编程打好基础。

6. 总结

这次大作业让我的编程能力有了很大的提升。这是我第一次为实际满足需求编写一个较大的程序，很贴近实际情况。刚拿到题目时我不知所措、心乱如麻不知如何下手。冷静下来思考之后发现，只需根据功能将程序分成一个个界面，之后各个击破即可。这种将一个大问题拆成很多小问题各个击破的方式很有作用。同时编写这种代码数较多的程序对我的调试能力也有很大提升。

这次大作业同时提升了我对课上学习的基础知识的理解。尤其是全局变量、全局函数以及类的多继承、虚函数这几部分。由于笔试不考所以学完了就忘掉了，这次大作业让我明白这些知识有多么重要，使用起来多么方便。

这次大作业对我最大的提升是解决问题的能力。遇到不会的地方翻书找、上网查、问同学；编译出错了根据提示上网搜索出错原因；和同学们一起讨论算法、讨论解决问题的最优方式……这种发现问题、解决问题的过程真的很迷人。

总之，这次大作业提升了我的编程能力，增加了我对编程的兴趣，提高了我的自信心，让我明白了细节的重要性，让我受益匪浅。

7. 附录：源程序清单

1.1. class.h

//本文件中存放个各类的定义

```
#include <string>
```

```
using namespace std;
```

//课程信息类（包含开课时间、名称、所占学分三个变量）

```
class Course
```

```
{
```

```
public:
```

```
    Course(string a = "\0", string b = "\0", int c = 0);
```

```
    virtual void set(double m, string a, string b, int c) = 0;
```

```
    virtual void set(double m, string a, string b, int c, string d, string e, string f) = 0;
```

```
    ~Course() {}
```

```
protected:
```

```
    string term;
```

```
    string cname; //课程名
```

```
    int credit;
```

```
};
```

//成绩类（继承课程信息类，包含分数、等级、学分绩、是否挂科四个变量）

```
class Grade : public Course
```

```
{
```

```
public:
```

```
    Grade(double m = 0, string a = "\0", string b = "\0", int c = 0); //level,gpa,flag等变量通过a的值确定
```

```
    virtual void set(double m, string a, string b, int c);
```

```
    ~Grade() {}
```

```
protected:
```

```
    double mark;
```

```
    string level; //成绩对应的等级，如A+等
```

```
    double gpa;
```

```
    bool flag; //是否挂科变量，true表示挂科，false表示未挂科
```

```
};
```

//学生类（包含姓名、班级、学号）

```
class Student
```

```
{
```

```
public:
```

```
    Student(string a = "\0", string b = "\0", string c = "\0");
```

```

    string getSname();
    string getClass();
    string getNum();
    void set(string a, string b, string c);
    ~Student() {}
protected:
    string sname; //学生姓名
    string Class;
    string number;
};

//录入信息类（继承成绩类、学生类）
class Info : public Grade, public Student
{
public:
    Info(double m = 0, string a = "\0", string b = "\0", int c = 0, string d = "\0", string e =
"\0", string f = "\0");
    virtual void set(double m, string a, string b, int c, string d, string e, string f);
    string getTerm(); //返回该条信息中课程所在学期
    string getCName(); //返回该条信息中课程名
    double getGpa(); //返回该条信息中课程学分绩
    int getCredit(); //返回该条信息中课程学分
    double getMark(); //返回该条信息中课程分数
    friend ostream& operator <<(ostream&, Info&); //重载流插入运算符便于输出考生所见成绩信息
    ~Info() {}
};

//学生成绩类（继承学生类，包含上、下学期及整个学年所修学分、平均gpa、挂科数、挂科总学分）
class studentGrade : public Student
{
public:
    studentGrade(const char* name = "\0", const char* Class = "\0", const char* num = "\0",
const int upCredit = 0, const int downCredit = 0, double upGpa = 0, double downGpa = 0, int
upFailNumber = 0, int downFailNumber = 0, int upFailCredit = 0, int downFailCredit = 0);
    string getNum() { return number; }

    //传回变量地址以修改
    int& getupCredit();
    int& getdownCredit();
    int& gettotalCredit();
    double& getupGpa();
    double& getdownGpa();
};

```



```

    double& gettotalGpa();
    int& getupFailNumber();
    int& getdownFailNumber();
    int& gettotalFailNumber();
    int& getupFailCredit();
    int& getdownFailCredit();
    int& gettotalFailCredit();
    ~studentGrade() {}
protected:
    int upCredit, downCredit, totalCredit;
    double upGpa, downGpa, totalGpa;
    int upFailNumber, downFailNumber, totalFailNumber;
    int upFailCredit, downFailCredit, totalFailCredit;
};

```

1.2. classfunction.h

//本文件中存放各个类的成员函数

```
#include <iostream>
```

```
#include <iomanip>
```

//课程信息类

```
Course::Course(string a, string b, int c)
```

```
{
    term=a;
    cname=b;
    credit=c;
}
```

//成绩类

```
Grade::Grade(double m, string a, string b, int c):Course(a, b, c)
```

```
{
    mark=m;
    if(mark<60) {flag=true;gpa=0;level="F";}
    else
    {
        flag=false;
        if(mark>=60&&mark<=62) {gpa=1;level="D";}
        else if(mark>=63&&mark<=66) {gpa=1.3;level="D+";}
        else if(mark>=67&&mark<=69) {gpa=1.7;level="C-";}
        else if(mark>=70&&mark<=72) {gpa=2.0;level="C";}
    }
}
```

```

        else if (mark>=73&&mark<=76)    {gpa=2.3;level="C+";}
        else if (mark>=77&&mark<=79)    {gpa=2.7;level="B-";}
        else if (mark>=80&&mark<=84)    {gpa=3.0;level="B";}
        else if (mark>=85&&mark<=89)    {gpa=3.3;level="B+";}
        else if (mark>=90&&mark<=94)    {gpa=3.7;level="A-";}
        else if (mark>=95) {gpa=4.0;level="A";}
    }
}

```

```

void Grade::set(double m,string a,string b,int c)
{
    mark=m;
    if (mark<60) {flag=true;gpa=0;level="F";}
    else
    {
        flag=false;
        if (mark>=60&&mark<=62) {gpa=1;level="D";}
        else if (mark>=63&&mark<=66) {gpa=1.3;level="D+";}
        else if (mark>=67&&mark<=69)    {gpa=1.7;level="C-";}
        else if (mark>=70&&mark<=72)    {gpa=2.0;level="C";}
        else if (mark>=73&&mark<=76)    {gpa=2.3;level="C+";}
        else if (mark>=77&&mark<=79)    {gpa=2.7;level="B-";}
        else if (mark>=80&&mark<=84)    {gpa=3.0;level="B";}
        else if (mark>=85&&mark<=89)    {gpa=3.3;level="B+";}
        else if (mark>=90&&mark<=94)    {gpa=3.7;level="A-";}
        else if (mark>=95) {gpa=4.0;level="A";}
    }
    term=a;
    cname=b;
    credit=c;
}

```

//学生类

```

Student::Student(string a,string b,string c)
{
    sname=a;
    Class=b;
    number=c;
}

```

```

void Student::set(string a,string b,string c)
{
    sname=a;

```

```

        Class=b;
        number=c;
    }

    string Student:: getSname()
    {
        return sname;
    }

    string Student:: getClass()
    {
        return Class;
    }

    string Student:: getNum()
    {
        return number;
    }

//录入信息类
Info::Info(double m,string a,string b,int c,string d,string e,string f):
Grade(m,a,b,c),Student(d,e,f)
{
}

void Info::set(double m,string a,string b,int c,string d,string e,string f)
{
    Grade::set(m,a,b,c);
    Student::set(d,e,f);
}

string Info::getTerm()
{
    return term;
}

string Info::getCName()
{
    return cname;
}

int Info::getCredit()
{

```

```

        return credit;
    }

double Info::getMark()
{
    return mark;
}

double Info::getGpa()
{
    return gpa;
}

ostream& operator <<(ostream& output, Info& info)
{
    cout.width(15);
    output<<setiosflags(_IOSleft)<<info.term;
    cout.width(15);
    output<<setiosflags(_IOSleft)<<info.cname;
    cout.width(15);
    output<<setiosflags(_IOSleft)<<info.credit;
    cout.width(15);
    output<<setiosflags(_IOSleft)<<info.mark;
    cout.width(15);
    output<<setiosflags(_IOSleft)<<info.level;
    cout.width(15);
    output<<setiosflags(_IOSleft)<<info.gpa<<endl;
    return output;
}

```

//学生成绩类

```

studentGrade::studentGrade(const char* name, const char* Class, const char* num, int
upCredit, int downCredit, double upGpa, double downGpa, int upFailNumber, int downFailNumber,
int upFailCredit, int downFailCredit): Student(name, Class, num)
{
    this->upCredit=upCredit; this->downCredit=downCredit;
    totalCredit=upCredit+downCredit;

    this->upGpa=upGpa; this->downGpa=downGpa;
    totalGpa=(upGpa*upCredit+downGpa*downCredit)/(upCredit+downCredit);

    this->upFailNumber=upFailNumber; this->downFailNumber=downFailNumber;
    totalFailNumber=upFailNumber+downFailNumber;
}

```

```
        this->upFailCredit=upFailCredit;this->downFailCredit=upFailCredit;  
        totalFailCredit=upFailCredit+downFailCredit;  
    }  
}
```

```
int& studentGrade:: getupCredit()  
{  
    return upCredit;  
}
```

```
int& studentGrade:: getdownCredit()  
{  
    return downCredit;  
}
```

```
int& studentGrade:: gettotalCredit()  
{  
    return totalCredit;  
}
```

```
double& studentGrade:: getupGpa()  
{  
    return upGpa;  
}
```

```
double& studentGrade:: getdownGpa()  
{  
    return downGpa;  
}
```

```
double& studentGrade:: gettotalGpa()  
{  
    return totalGpa;  
}
```

```
int& studentGrade:: getupFailNumber()  
{  
    return upFailNumber;  
}
```

```
int& studentGrade:: getdownFailNumber()  
{  
    return downFailNumber;  
}
```

```
int& studentGrade:: gettotalFailNumber()
{
    return totalFailNumber;
}
```

```
int& studentGrade:: getupFailCredit()
{
    return upFailCredit;
}
```

```
int& studentGrade:: getdownFailCredit()
{
    return downFailCredit;
}
```

```
int& studentGrade:: gettotalFailCredit()
{
    return totalFailCredit;
}
```

1.3. main.cpp

```
#include "stdafx.h"
extern void startPage();           //初始化系统

using namespace std;

int main()
{
    startPage();
    return 0;
}
```

1.4. startpage.cpp

```
//存放各种界面界面
#include "stdafx.h"
#include "class.h"
#include "classfunction.h"
#include <iostream>
#include <cmath>
```

```

#include <cstdlib>
#include <string>
#include <conio.h>
#include <fstream>
using namespace std;

int counta=0;           //全局变量, 文件内有效信息数
studentGrade *stugrade=NULL; //全局变量, 文件内的每位学生成绩的对象数组的首地址
int num=0;              //全局变量, 文件内学生数

void startPage();       //开始界面
void page0();           //界面0: 主菜单界面
extern void page1();     //界面1: 录入、修改学生成绩
extern void page2();     //界面2: 查询某学生成绩
extern void page3();     //界面3: 查询某课程所有学生成绩
extern void page4();     //界面4: 查询所有学生总成绩情况
extern void page5();     //界面5: 查询挂科情况
void page6();           //界面6: 系统的结束界面
void update1();          //更新文件, 删除修改前的学生成绩
void update2();          //制作所有学生的studentGrade对象

//界面0: 主菜单界面
void page0()
{
    system("cls");
    cout<<"*****"<<endl;
    cout<<"*                  学生成绩管理系统                  *"<<endl;
    cout<<"*****"<<endl;
    cout<<"1.-----录入、修改学生成绩-----*"<<endl;
    cout<<"2.-----查询某学生成绩-----*"<<endl;
    cout<<"3.-----查询某课程所有学生成绩-----*"<<endl;
    cout<<"4.-----查询所有学生总成绩-----*"<<endl;
    cout<<"5.-----查询挂科情况-----*"<<endl;
    cout<<"6.-----退出系统-----*"<<endl;

    switch(_getwch())
    {
        case '1': page1();break;
        case '2': page2();break;
        case '3': page3();break;
        case '4': page4();break;
        case '5': page5();break;
        case '6': page6();break;
    }
}

```

```

        default:
        {
            cout<<"输入有误，按任意键返回主界面\n";
            _getwch();
            page0();
        }
    }
}

//界面6：结束界面
void page6()
{
    system("cls");
    cout<<"*****"<<endl;
    cout<<"*                  感谢使用本系统                  *"<<endl;
    cout<<"*****"<<endl;
    exit (0);
    delete []stugrade;
}

//更新文件，删除修改前的学生成绩
void update1()
{
    counta=0;
    ifstream infile("grade.txt",ios::in|ios::_Nocreate);
    if(!infile)
    {
        cout<<"文件grade.txt无法打开，按任意键退出系统\n";
        _getwch();
        page6();
    }
}

//统计文件中的信息数，便于创建录入信息类对象数组
int n=0;
while(1)
{
    if(infile.get()=='\n')n++;
    if(infile.fail()) break;
}
infile.clear();infile.sync();
infile.seekg(0);
//创建录入信息类对象数组，存放最新版本的信息
Info *p=new Info[n];
string term,sname,Class,number,cname;

```



```

double mark;
int credit;
int i=0;
int k=0, flag=0;
for(i=0; i<n; i++)
{
    flag=0;
    infile>>term>>sname>>Class>>number>>cname>>credit>>mark;
    for(k=0; k<i; k++)
    {
        if(term==p[k].getTerm() && number==p[k].getNum() && cname==p[k].getCName())
{flag=1; break;}
    }
    if(flag==0) p[counta].set(mark, term, cname, credit, sname, Class, number); counta++;
    else p[k].set(mark, term, cname, credit, sname, Class, number);
}
infile.close();

//将最新版本的信息写入文件
ofstream outfile("grade.txt", ios::out|ios::trunc);
if(!outfile)
{
    cout<<"文件grade.txt无法打开，按任意键退出系统\n";
    _getwch();
    page6();
}
for(i=0; i<counta; i++)
{
    outfile<<p[i].getTerm()<<" "<<p[i].getSname()<<" "<<p[i].getClass()<<"
"<<p[i].getNum()<<" "<<p[i].getCName()<<" "<<p[i].getCredit()<<" "<<p[i].getMark()<<endl;
}

outfile.close();
delete []p;
}

```

//制作所有学生的studentGrade类对象数组

```

void update2()
{
    stugrade=new studentGrade[counta];
    ifstream infile("grade.txt", ios::in|ios::_Nocreate);
    if(!infile)
    {
        cout<<"文件grade.txt无法打开，按任意键退出系统\n";
    }
}

```

```

        _getwch();
        page6();
    }

```

```

Info test;

```

```

int flag=0, i=0, k=0;
string term, sname, Class, number, cname;
int credit;
double mark;
for(i=0; i<counta; i++)
{
    flag=0;
    infile>>term>>sname>>Class>>number>>cname>>credit>>mark;

    //用信息对象得到完整版信息
    test.set(mark, term, cname, credit, sname, Class, number);

    for(k=0; k<=i; k++)
    {
        if(number==stugrade[k].getNum()) {flag=1; break;}
        if(stugrade[k].getNum()=="") break;
    }

    if(flag==0) num++;    //确定系统中学生人数

    //统计信息
    stugrade[k].set(sname, Class, number);
    if(term=="大一上学期")
    {
        stugrade[k].getupCredit()+=credit;
        stugrade[k].getupGpa()+=test.getCredit()*test.getGpa();
        if(mark<60)
        {
            stugrade[k].getupFailCredit()+=credit;
            stugrade[k].getupFailNumber()+=1;
        }
    }
    if(term=="大一下学期")
    {
        stugrade[k].getdownCredit()+=credit;
        stugrade[k].getdownGpa()+=test.getCredit()*test.getGpa();
        if(mark<60)
        {

```

```

        stugrade[k].getdownFailCredit()+=credit;
        stugrade[k].getdownFailNumber()+=1;
    }
}
infile.close();

//在获得完整版信息后完善信息
for(i=0;i<num;i++)
{
    stugrade[i].gettotalCredit()=stugrade[i].getupCredit()+stugrade[i].getdownCredit();

    stugrade[i].gettotalFailNumber()=stugrade[i].getupFailNumber()+stugrade[i].getdownFailNumber();

    stugrade[i].gettotalFailCredit()=stugrade[i].getupFailCredit()+stugrade[i].getdownFailCredit();

    if(stugrade[i].getupCredit()!=0) stugrade[i].getupGpa()/=stugrade[i].getupCredit();
    if(stugrade[i].getdownCredit()!=0)
stugrade[i].getdownGpa()/=stugrade[i].getdownCredit();

    stugrade[i].gettotalGpa()=(stugrade[i].getupGpa()*stugrade[i].getupCredit()+stugrade[i].getdownGpa()*stugrade[i].getdownCredit())/stugrade[i].gettotalCredit();
}
}

//开始界面
void startPage()
{
    update1();
    update2();
    cout<<"*****"<<endl;
    cout<<"*                欢迎使用学生成绩管理系统                *"<<endl;
    cout<<"*****"<<endl;
    cout<<"按任意键进入系统"<<endl;
    _getwch();
    page0();
}

```

1.5. page1.cpp

```

#include"stdafx.h"
#include "class.h"

```

```

#include <iostream>
#include <cmath>
#include <cstdlib>
#include <string>
#include <conio.h>
#include <fstream>
using namespace std;

void operation1();
extern void page0();
extern void page6();
extern void update1();
extern void update2();
//界面1: 录入、修改学生成绩
void page1()
{
    system("cls");
    cout << "*****" << endl;
    cout << "          录入、修改学生成绩          *" << endl;
    cout << "*****" << endl;
    cout << "请按以下格式输入学生成绩，输入#返回主界面。若该课程成绩已存在，则将覆盖原成绩\n";
    cout << "学期 姓名 班级（汉字） 学号 科目 学分 分数\n";
    operation1();
}

void operation1()
{
    string term, sname, Class, number, cname;
    double mark;
    int credit;
    ofstream outfile("grade.txt", ios::out | ios::_Nocreate | ios::app);
    if (!outfile)
    {
        cout << "文件grade.txt无法打开，按任意键退出系统\n";
        _getwch();
        page6();
    }
    while (1)    //一直执行直到读到#
    {
        cin >> term;
        if (term == "#")
        {
            outfile.close();
            update1();
        }
    }
}

```

```

        update2;
        page0();
    }
    if ((term != "大一上学期"&&term != "大一下学期") || cin.fail())    //防止错误输入
    {
        cout << "输入信息有误，按任意键重新开始输入\n";
        cin.clear();
        cin.sync();
        _getwch();
        page1();
    }
    cin >> sname >> Class >> number >> cname >> credit >> mark;
    cin.clear(); cin.sync();

    if (cin.fail())
    {
        cout << "输入信息有误，请重新输入\n";
        cin.clear();
        cin.sync();
    }
    else
    {
        outfile << term << " " << sname << " " << Class << " " << number << " " << cname
        << " " << credit << " " << mark << endl;
    }
}
}

```

1.6. page2.cpp

```

#include "stdafx.h"
#include <iostream>
#include <cmath>
#include <cstdlib>
#include <string>
#include <conio.h>
#include <fstream>
#include "class.h"
using namespace std;

extern int counta;

extern void page0();
extern void page6();

```

```

void FirstGrade(string tnumber);
void SecondGrade(string tnumber);
void AllGrade(string tnumber);
//界面2: 查询某学生成绩
void page2()
{
    string S_number;
    system("cls");
    cout << "*****" << endl;
    cout << "          查询某学生成绩          *" << endl;
    cout << "*****" << endl;
    cout << "请输入学号，输入#返回主界面\n";
    cin >> S_number;
    if (S_number == "#") page0();
    if (S_number.length() != 10 || cin.fail())
    {
        cout << "输入信息有误，按任意键重新开始输入\n";
        _getwch();
        cin.clear();
        cin.sync();
        page2();
    }

    system("cls");
    cout << "*****" << endl;
    cout << "          查询某学生成绩          *" << endl;
    cout << "*****" << endl;
    cout << "1.-----查询大一上学期成绩-----*" << endl;
    cout << "2.-----查询大一下学期成绩-----*" << endl;
    cout << "3.-----查询大一学年成绩-----*" << endl;
    cout << "4.-----返回主界面-----*" << endl;
    switch (_getwch())
    {
    case '1': FirstGrade(S_number); break;
    case '2': SecondGrade(S_number); break;
    case '3': AllGrade(S_number); break;
    case '4': page0(); break;
    default:
    {
        cout << "输入有误，按任意键重新开始输入\n";
        _getwch();
        page2();
    }
    }
}

```

```

        cout << "查询完毕，按任意键返回上一界面\n";
        _getwch();
        page2();
    }

//查询大一上学期成绩
void FirstGrade(string tnumber)
{
    ifstream infile("grade.txt", ios::in | ios::_Nocreate);
    if (!infile)
    {
        cout << "文件grade.txt无法打开，按任意键退出系统\n";
        _getwch();
        page6();
    }

    Info *p = new Info[counta];
    string term, sname, Class, number, cname;
    double mark;
    int credit;
    int i = 0;
    int num = 0;
    for (i = 0; i < counta; i++)
    {
        infile >> term >> sname >> Class >> number >> cname >> credit >> mark;
        if (number == tnumber && term == "大一上学期")
        {
            p[num].set(mark, term, cname, credit, sname, Class, number);
            num++;
        }
    }
    if (num == 0)
    {
        cout << "此学生不在系统中\n";
        cout << "按任意键返回上一界面\n";
        _getwch();
        page2();
    }

//排序
    Info **q = new Info*[num];
    Info *t = NULL;
    for (i = 0; i < num; i++) q[i] = &p[i];

//先对成绩降序排序

```

```

    for (i = 0; i < num; i++)
    {
        for (int k = i; k < num; k++)
            if (q[i]->getMark() <= q[k]->getMark())
            {
                t = q[i]; q[i] = q[k]; q[k] = t;
            }
    }
    //再将成绩相同的按学分降序排列
    for (i = 0; i < num; i++)
    {
        for (int k = i; k < num; k++)
            if (q[i]->getMark() == q[k]->getMark() && q[i]->getCredit() <= q[k]->getCredit())
            {
                t = q[i]; q[i] = q[k]; q[k] = t;
            }
    }

    //输出信息;
    double Gpa = 0;
    int totalcredit = 0;
    cout << q[0]->getClass() << " " << q[0]->getSname() << " " << q[0]->getNum() << " 大一上学
期成绩单" << endl;
    for (i = 0; i < num; i++)
    {
        cout << *q[i];
        Gpa += q[i]->getGpa()*q[i]->getCredit();
        totalcredit += q[i]->getCredit();
    }
    cout << "大一上学期平均GPA为 " << (Gpa / totalcredit) << endl;

    delete[]p;
    delete[]q;
    infile.close();
}

//查询大一下学期成绩
void SecondGrade(string tnumber)
{
    ifstream infile("grade.txt", ios::in | ios::_Nocreate);
    if (!infile)
    {
        cout << "文件grade.txt无法打开, 按任意键退出系统\n";
        _getwch();
    }
}

```



```

        page6();
    }

    Info *p = new Info[counta];
    string term, sname, Class, number, cname;
    double mark;
    int credit;
    int i = 0;
    int num = 0;
    for (i = 0; i < counta; i++)
    {
        infile >> term >> sname >> Class >> number >> cname >> credit >> mark;
        if (number == tnumber && term == "大一下学期")
        {
            p[num].set(mark, term, cname, credit, sname, Class, number);
            num++;
        }
    }
    if (num == 0)
    {
        cout << "此学生不在系统中\n";
        cout << "按任意键返回上一界面\n";
        _getwch();
        page2();
    }

    //排序
    Info **q = new Info*[num];
    Info *t = NULL;
    for (i = 0; i < num; i++) q[i] = &p[i];

    //先对成绩降序排序
    for (i = 0; i < num; i++)
    {
        for (int k = i; k < num; k++)
            if (q[i]->getMark() <= q[k]->getMark())
            {
                t = q[i]; q[i] = q[k]; q[k] = t;
            }
    }

    //再将成绩相同的按学分降序排列
    for (i = 0; i < num; i++)
    {
        for (int k = i; k < num; k++)
            if (q[i]->getMark() == q[k]->getMark() && q[i]->getCredit() <= q[k]->getCredit())

```

```

        {
            t = q[i]; q[i] = q[k]; q[k] = t;
        }
    }

    //输出信息;
    double Gpa = 0;
    int totalcredit = 0;
    cout << q[0]->getClass() << " " << q[0]->getSname() << " " << q[0]->getNum() << " 大一上学
期成绩单" << endl;
    for (i = 0; i < num; i++)
    {
        cout << *q[i];
        Gpa += q[i]->getGpa()*q[i]->getCredit();
        totalcredit += q[i]->getCredit();
    }
    cout << "大一下学期平均GPA为 " << (Gpa / totalcredit) << endl;

    delete[]p;
    delete[]q;
    infile.close();
}

//查询大一学年学期成绩
void AllGrade(string tnumber)
{
    ifstream infile("grade.txt", ios::in | ios::_Nocreate);
    if (!infile)
    {
        cout << "文件grade.txt无法打开, 按任意键退出系统\n";
        _getwch();
        page6();
    }

    Info *p = new Info[counta];
    string term, sname, Class, number, cname;
    double mark;
    int credit;
    int i = 0;
    int num = 0;
    for (i = 0; i < counta; i++)
    {
        infile >> term >> sname >> Class >> number >> cname >> credit >> mark;
        if (number == tnumber && (term == "大一上学期" || term == "大一下学期"))

```

```

        {
            p[num].set(mark, term, cname, credit, sname, Class, number);
            num++;
        }
    }
    if (num == 0)
    {
        cout << "此学生不在系统中\n";
        cout << "按任意键返回上一界面\n";
        _getwch();
        page2();
    }
    //排序
    Info **q = new Info*[num];
    Info *t = NULL;
    for (i = 0; i < num; i++) q[i] = &p[i];

    //先将上学期的课程放在靠前位置
    for (i = 0; i < num; i++)
    {
        for (int k = i; k < num; k++)
            if (q[k]->getTerm() == "大一上学期" && q[i]->getTerm() == "大一下学期")
            {
                t = q[i]; q[i] = q[k]; q[k] = t;
            }
    }
    //先对成绩降序排序
    for (i = 0; i < num; i++)
    {
        for (int k = i; k < num; k++)
            if (q[i]->getMark() <= q[k]->getMark() && q[i]->getTerm() == q[k]->getTerm())
            {
                t = q[i]; q[i] = q[k]; q[k] = t;
            }
    }
    //再将成绩相同的按学分降序排列
    for (i = 0; i < num; i++)
    {
        for (int k = i; k < num; k++)
            if (q[i]->getMark() == q[k]->getMark() && q[i]->getCredit() <= q[k]->getCredit())
            {
                t = q[i]; q[i] = q[k]; q[k] = t;
            }
    }
}

```

```

//输出信息;
double Gpa = 0;
int totalcredit = 0;
cout << q[0]->getClass() << " " << q[0]->getSname() << " " << q[0]->getNum() << " 大一上学
期成绩单" << endl;
for (i = 0; i < num; i++)
{
    cout << *q[i];
    Gpa += q[i]->getGpa()*q[i]->getCredit();
    totalcredit += q[i]->getCredit();
}
cout << "\n大一学年平均GPA为 " << (Gpa / totalcredit) << endl;

delete[]p;
delete[]q;
infile.close();
}

```

1.7. page3.cpp

```

#include "stdafx.h"
#include <iostream>
#include <cmath>
#include <cstdlib>
#include <string>
#include <conio.h>
#include <fstream>
#include <iomanip>
#include "class.h"
using namespace std;

extern void page0();
extern void page6();
extern int counta;

void operation3(string tterm, string tcname);
//界面3: 查询某课程所有学生成绩
void page3()
{
    string tterm, tcname;
    system("cls");
}

```

```

cout << "*****" << endl;
cout << "          某课程所有学生成绩          " << endl;
cout << "*****" << endl;
cout << "请输入学期 课程名, 输入#返回主界面\n";
cin >> tterm;
if (tterm == "#") page0();
if ((tterm != "大一上学期"&& tterm != "大一下学期") || cin.fail())
{
    cout << "输入信息有误, 按任意键重新开始输入\n";
    _getwch();
    cin.clear();
    cin.sync();
    page3();
}
cin >> tcname;

operation3(tterm, tcname);

cout << "\n查询完毕, 按任意键返回上一界面\n";
_getwch();
page3();
}

void operation3(string tterm, string tcname)
{
    ifstream infile("grade.txt", ios::in | ios::_Nocreate);
    if (!infile)
    {
        cout << "文件grade.txt无法打开, 按任意键退出系统\n";
        _getwch();
        page6();
    }
    Info *p = new Info[counta];
    string term, sname, Class, number, cname;
    double mark;
    int credit;
    int i = 0;
    int num = 0;
    for (i = 0; i < counta; i++)
    {
        infile >> term >> sname >> Class >> number >> cname >> credit >> mark;
        if (term == tterm && cname == tcname)
        {
            p[num].set(mark, term, cname, credit, sname, Class, number);

```

```

        num++;
    }
}
if (num == 0)
{
    cout << "系统中无此课程中\n";
    cout << "按任意键返回上一界面\n";
    _getwch();
    page3();
}

//排序
Info **q = new Info*[num];
Info *t = NULL;
for (i = 0; i < num; i++) q[i] = &p[i];

//先对成绩降序排序
for (i = 0; i < num; i++)
{
    for (int k = i; k < num; k++)
        if (q[i]->getMark() <= q[k]->getMark())
        {
            t = q[i]; q[i] = q[k]; q[k] = t;
        }
}

//再对学号升序排序
for (i = 0; i < num; i++)
{
    for (int k = i; k < num; k++)
        if (q[i]->getNum().compare(q[k]->getNum()) > 0 && q[i]->getMark() ==
q[k]->getMark())
        {
            t = q[i]; q[i] = q[k]; q[k] = t;
        }
}

//输出
cout << q[0]->getTerm() << q[0]->getCName() << "成绩情况表\n";
cout.width(15);
cout << setw(15, _IOSleft) << "排名";
cout.width(15);
cout << setw(15, _IOSleft) << "学号";
cout.width(15);

```

```

    cout << setiosflags(_IOSleft) << "姓名";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "班级";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "学期";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "课程";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "学分";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "分数";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "等级";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "学分绩";
    cout << endl;
    for (i = 0; i < num; i++)
    {
        cout.width(15);
        cout << setiosflags(_IOSleft) << (i + 1);
        cout.width(15);
        cout << setiosflags(_IOSleft) << q[i]->getNum();
        cout.width(15);
        cout << setiosflags(_IOSleft) << q[i]->getSname();
        cout.width(15);
        cout << setiosflags(_IOSleft) << q[i]->getClass();
        cout << *q[i];
    }

    delete[] p;
    delete[] q;
    infile.close();
}

```

1.8. page4.cpp

```

#include "stdafx.h"
#include <iostream>
#include <cmath>
#include <stdlib>
#include <string>
#include <conio.h>
#include <fstream>
#include <iomanip>
#include "class.h"

```

```

using namespace std;

extern void page0();
extern void page6();
extern studentGrade *stugrade;
extern int num;

void operation41(studentGrade *p, int num);
void operation42(studentGrade *p, int num);
void operation43(studentGrade *p, int num);

//界面4: 查询所有学生总成绩情况
void page4()
{
    system("cls");
    cout << "*****" << endl;
    cout << "                查询所有学生总成绩                *" << endl;
    cout << "*****" << endl;
    cout << "1. -----查询大一上学期成绩-----*" << endl;
    cout << "2. -----查询大一下学期成绩-----*" << endl;
    cout << "3. -----查询大一学年成绩-----*" << endl;
    cout << "4. -----返回主界面-----*" << endl;

    switch (_getwch())
    {
        case '1': operation41(stugrade, num); break;
        case '2': operation42(stugrade, num); break;
        case '3': operation43(stugrade, num); break;
        case '4': page0(); break;
        default:
        {
            cout << "输入有误，按任意键重新开始输入\n";
            _getwch();
            page4();
        }
    }

    cout << "查询完毕，按任意键返回上一界面\n";
    _getwch();
    page4();
}

//查询大一上
void operation41(studentGrade *p, int num)

```



```

{
    int i = 0, k = 0;
    studentGrade **q = new studentGrade*[num];
    studentGrade *t;
    for (i = 0; i < num; i++) q[i] = &p[i];

    //对成绩降序排序
    for (i = 0; i < num; i++)
    {
        for (k = i; k < num; k++)
            if (q[i]->getupGpa() < q[k]->getupGpa())
            {
                t = q[i]; q[i] = q[k]; q[k] = t;
            }
    }

    //成绩相同按学号升序排序
    for (i = 0; i < num; i++)
    {
        for (int k = i; k < num; k++)
            if (q[i]->getNum().compare(q[k]->getNum()) > 0 && q[i]->getupGpa() ==
q[k]->getupGpa())
            {
                t = q[i]; q[i] = q[k]; q[k] = t;
            }
    }

    cout << "\n大一上学期总成绩情况表\n";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "排名";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "学号";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "姓名";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "班级";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "平均学分绩";
    cout << endl;
    for (i = 0; i < num; i++)
    {
        cout.width(15);
        cout << setiosflags(_IOSleft) << (i + 1);
        cout.width(15);
        cout << setiosflags(_IOSleft) << q[i]->getNum();
    }
}

```

```

        cout.width(15);
        cout << setiosflags(_IOSleft) << q[i]->getSname();
        cout.width(15);
        cout << setiosflags(_IOSleft) << q[i]->getClass();
        cout.width(15);
        cout << setiosflags(_IOSleft) << q[i]->getupGpa() << endl;
    }
    delete[]q;
    cout << "\n查询完毕，按任意键返回上一界面\n";
    _getwch();
    page4();
}

```

//查询大一下

```

void operation42(studentGrade *p, int num)
{
    int i = 0, k = 0;
    studentGrade **q = new studentGrade*[num];
    studentGrade *t;
    for (i = 0; i < num; i++) q[i] = &p[i];

    //对成绩降序排序
    for (i = 0; i < num; i++)
    {
        for (k = i; k < num; k++)
            if (q[i]->getdownGpa() < q[k]->getdownGpa())
            {
                t = q[i]; q[i] = q[k]; q[k] = t;
            }
    }

    //成绩相同按学号升序排序
    for (i = 0; i < num; i++)
    {
        for (int k = i; k < num; k++)
            if (q[i]->getNum().compare(q[k]->getNum()) > 0 && q[i]->getdownGpa() ==
q[k]->getdownGpa())
            {
                t = q[i]; q[i] = q[k]; q[k] = t;
            }
    }

    cout << "\n大一下学期总成绩情况表\n";
    cout.width(15);

```

```

cout << setiosflags(_IOSleft) << "排名";
cout.width(15);
cout << setiosflags(_IOSleft) << "学号";
cout.width(15);
cout << setiosflags(_IOSleft) << "姓名";
cout.width(15);
cout << setiosflags(_IOSleft) << "班级";
cout.width(15);
cout << setiosflags(_IOSleft) << "平均学分绩";
cout << endl;
for (i = 0; i < num; i++)
{
    cout.width(15);
    cout << setiosflags(_IOSleft) << (i + 1);
    cout.width(15);
    cout << setiosflags(_IOSleft) << q[i]->getNum();
    cout.width(15);
    cout << setiosflags(_IOSleft) << q[i]->getSname();
    cout.width(15);
    cout << setiosflags(_IOSleft) << q[i]->getClass();
    cout.width(15);
    cout << setiosflags(_IOSleft) << q[i]->getdownGpa() << endl;
}
delete[]q;
cout << "\n查询完毕，按任意键返回上一界面\n";
_getwch();
page4();
}

```

//查询大一学年

```

void operation43(studentGrade *p, int num)
{
    int i = 0, k = 0;
    studentGrade **q = new studentGrade*[num];
    studentGrade *t;
    for (i = 0; i < num; i++) q[i] = &p[i];

    //对成绩降序排序
    for (i = 0; i < num; i++)
    {
        for (k = i; k < num; k++)
            if (q[i]->gettotalGpa() < q[k]->gettotalGpa())
            {
                t = q[i]; q[i] = q[k]; q[k] = t;
            }
    }
}

```

```

        }
    }
    //成绩相同按学号升序排序
    for (i = 0; i < num; i++)
    {
        for (int k = i; k < num; k++)
            if (q[i]->getNum().compare(q[k]->getNum()) > 0 && q[i]->gettotalGpa() ==
q[k]->gettotalGpa())
            {
                t = q[i]; q[i] = q[k]; q[k] = t;
            }
    }

    cout << "\n大一学年总成绩情况表\n";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "排名";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "学号";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "姓名";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "班级";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "平均学分绩";
    cout << endl;
    for (i = 0; i < num; i++)
    {
        cout.width(15);
        cout << setiosflags(_IOSleft) << (i + 1);
        cout.width(15);
        cout << setiosflags(_IOSleft) << q[i]->getNum();
        cout.width(15);
        cout << setiosflags(_IOSleft) << q[i]->getName();
        cout.width(15);
        cout << setiosflags(_IOSleft) << q[i]->getClass();
        cout.width(15);
        cout << setiosflags(_IOSleft) << q[i]->gettotalGpa() << endl;
    }
    delete[]q;
    cout << "\n查询完毕，按任意键返回上一界面\n";
    _getwch();
    page4();
}

```

1.9. page5.cpp

```
#include "stdafx.h"
#include <iostream>
#include <cmath>
#include <cstdlib>
#include <string>
#include <conio.h>
#include <fstream>
#include <iomanip>
#include "class.h"
using namespace std;

extern void page0();
extern int num;
extern studentGrade* stugrade;
extern void page6();

void operation51(studentGrade *p, int num);
void operation52(studentGrade *p, int num);
void operation53(studentGrade *p, int num);

//界面5: 查询挂科情况
void page5()
{
    system("cls");
    cout << "*****" << endl;
    cout << "          查询挂科情况          *" << endl;
    cout << "*****" << endl;
    cout << "1.-----查询大一上学期挂科情况-----*" << endl;
    cout << "2.-----查询大一下学期挂科情况-----*" << endl;
    cout << "3.-----查询大一学年挂科情况-----*" << endl;
    cout << "4.-----返回主界面-----*" << endl;

    switch (_getwch())
    {
        case '1': operation51(stugrade, num); break;
        case '2': operation52(stugrade, num); break;
        case '3': operation53(stugrade, num); break;
        case '4': page0(); break;
        default:
        {
```

```

        cout << "输入有误，按任意键重新开始输入\n";
        _getwch();
        page5();
    }
}

cout << "查询完毕，按任意键返回上一界面\n";
_getwch();
page5();
}

void operation5l(studentGrade *p, int num)
{
    int i = 0, k = 0;
    int failnum;
    studentGrade **q = new studentGrade*[num];
    studentGrade *t;
    for (i = 0; i < num; i++)
    {
        if (p[i].getupFailCredit() != 0)
        {
            q[k] = &p[i];
            k++;
        }
    }
    failnum = k;

    //对挂科数降序排序
    for (i = 0; i < failnum; i++)
    {
        for (k = i; k < failnum; k++)
            if (q[i]->getupFailNumber() < q[k]->getupFailNumber())
            {
                t = q[i]; q[i] = q[k]; q[k] = t;
            }
    }

    //挂科数相同按挂科学分降序排序
    for (i = 0; i < failnum; i++)
    {
        for (k = i; k < failnum; k++)
            if (q[i]->getupFailNumber() == q[k]->getupFailNumber() && q[i]->getupFailCredit()
< q[k]->getupFailCredit())
            {
                t = q[i]; q[i] = q[k]; q[k] = t;
            }
    }
}

```

```

    }
    //挂科数、挂科学分相同按学号升序排序
    for (i = 0; i < failnum; i++)
    {
        for (k = i; k < failnum; k++)
            if (q[i]->getupFailNumber() == q[k]->getupFailNumber() && q[i]->getupFailCredit()
== q[k]->getupFailCredit() && q[i]->getNum().compare(q[k]->getNum()))
            {
                t = q[i]; q[i] = q[k]; q[k] = t;
            }
    }

    //输出
    cout << "\n大一上学期挂科情况表\n";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "学号";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "姓名";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "班级";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "挂科数";
    cout.width(15);
    cout << setiosflags(_IOSleft) << "挂科学分";
    cout << endl;
    for (i = 0; i < failnum; i++)
    {
        cout.width(15);
        cout << setiosflags(_IOSleft) << q[i]->getNum();
        cout.width(15);
        cout << setiosflags(_IOSleft) << q[i]->getSname();
        cout.width(15);
        cout << setiosflags(_IOSleft) << q[i]->getClass();
        cout.width(15);
        cout << setiosflags(_IOSleft) << q[i]->getupFailNumber();
        cout.width(15);
        cout << setiosflags(_IOSleft) << q[i]->getupFailCredit() << endl;
    }
    delete[]q;
    cout << "\n查询完毕，按任意键返回上一界面\n";
    _getwch();
    page5();
}

```

```

void operation52(studentGrade *p, int num)
{
    int i = 0, k = 0;
    int failnum;
    studentGrade **q = new studentGrade*[num];
    studentGrade *t;
    for (i = 0; i < num; i++)
    {
        if (p[i].getdownFailCredit() != 0)
        {
            q[k] = &p[i];
            k++;
        }
    }
    failnum = k;

    //对挂科数降序排序
    for (i = 0; i < failnum; i++)
    {
        for (k = i; k < failnum; k++)
            if (q[i]->getdownFailNumber() < q[k]->getdownFailNumber())
            {
                t = q[i]; q[i] = q[k]; q[k] = t;
            }
    }

    //挂科数相同按挂科学分降序排序
    for (i = 0; i < failnum; i++)
    {
        for (k = i; k < failnum; k++)
            if (q[i]->getdownFailNumber() == q[k]->getdownFailNumber() &&
                q[i]->getdownFailCredit() < q[k]->getdownFailCredit())
            {
                t = q[i]; q[i] = q[k]; q[k] = t;
            }
    }

    //挂科数、挂科学分相同按学号升序排序
    for (i = 0; i < failnum; i++)
    {
        for (int k = i; k < failnum; k++)
            if (q[i]->getdownFailNumber() == q[k]->getdownFailNumber() &&
                q[i]->getdownFailCredit() == q[k]->getdownFailCredit() &&
                q[i]->getNum().compare(q[k]->getNum()) > 0)
            {
                t = q[i]; q[i] = q[k]; q[k] = t;
            }
    }
}

```



```

    }
}

//输出
cout << "\n大一下学期挂科情况表\n";
cout.width(15);
cout << setiosflags(_IOSleft) << "学号";
cout.width(15);
cout << setiosflags(_IOSleft) << "姓名";
cout.width(15);
cout << setiosflags(_IOSleft) << "班级";
cout.width(15);
cout << setiosflags(_IOSleft) << "挂科数";
cout.width(15);
cout << setiosflags(_IOSleft) << "挂科学分";
cout << endl;
for (i = 0; i < failnum; i++)
{
    cout.width(15);
    cout << setiosflags(_IOSleft) << q[i]->getNum();
    cout.width(15);
    cout << setiosflags(_IOSleft) << q[i]->getSname();
    cout.width(15);
    cout << setiosflags(_IOSleft) << q[i]->getClass();
    cout.width(15);
    cout << setiosflags(_IOSleft) << q[i]->getdownFailNumber();
    cout.width(15);
    cout << setiosflags(_IOSleft) << q[i]->getdownFailCredit() << endl;
}
delete[]q;
cout << "\n查询完毕, 按任意键返回上一界面\n";
_getwch();
page5();
}

void operation53(studentGrade *p, int num)
{
    int i = 0, k = 0;
    int failnum;
    studentGrade **q = new studentGrade*[num];
    studentGrade *t;
    for (i = 0; i < num; i++)
    {
        if (p[i].gettotalFailCredit() != 0)
        {

```

```

        q[k] = &p[i];
        k++;
    }
}
failnum = k;

//对挂科数降序排序
for (i = 0; i < failnum; i++)
{
    for (k = i; k < failnum; k++)
        if (q[i]->gettotalFailNumber() < q[k]->gettotalFailNumber())
        {
            t = q[i]; q[i] = q[k]; q[k] = t;
        }
}

//挂科数相同按挂科学分降序排序
for (i = 0; i < failnum; i++)
{
    for (k = i; k < failnum; k++)
        if (q[i]->gettotalFailNumber() == q[k]->gettotalFailNumber() &&
q[i]->gettotalFailCredit() < q[k]->gettotalFailCredit())
        {
            t = q[i]; q[i] = q[k]; q[k] = t;
        }
}

//挂科数、挂科学分相同按学号升序排序
for (i = 0; i < failnum; i++)
{
    for (int k = i; k < failnum; k++)
        if (q[i]->gettotalFailNumber() == q[k]->gettotalFailNumber() &&
q[i]->gettotalFailCredit() == q[k]->gettotalFailCredit() &&
q[i]->getNum().compare(q[k]->getNum()))
        {
            t = q[i]; q[i] = q[k]; q[k] = t;
        }
}

//输出
cout << "\n大一学年挂科情况表\n";
cout.width(15);
cout << setiosflags(_IOSleft) << "学号";
cout.width(15);
cout << setiosflags(_IOSleft) << "姓名";
cout.width(15);

```

```

cout << setiosflags(_IOSleft) << "班级";
cout.width(15);
cout << setiosflags(_IOSleft) << "挂科数";
cout.width(15);
cout << setiosflags(_IOSleft) << "挂科学分";
cout << endl;
for (i = 0; i < failnum; i++)
{
    cout.width(15);
    cout << setiosflags(_IOSleft) << q[i]->getNum();
    cout.width(15);
    cout << setiosflags(_IOSleft) << q[i]->getSname();
    cout.width(15);
    cout << setiosflags(_IOSleft) << q[i]->getClass();
    cout.width(15);
    cout << setiosflags(_IOSleft) << q[i]->gettotalFailNumber();
    cout.width(15);
    cout << setiosflags(_IOSleft) << q[i]->gettotalFailCredit() << endl;
}
delete[]q;
cout << "\n查询完毕，按任意键返回上一界面\n";
_getwch();
page5();
}

```

附录：评分表

第 1 题评分标准

项 目	评 价	
设计方案的合理性与创新性	3	
设计与调试结果	4	
设计说明书的质量	1	
程序基本要求涵盖情况	4	
程序代码编写素养情况	2	
课程设计周表现情况	1	

综合成绩	15	
------	----	--

教师签名：_____

日 期：_____