

《Python程序设计》

结课报告

项目名称：

学生姓名：

学 号：

专业班级：

日 期： 2024年1月3日

信息工程学院

目录

[一、背景及目的要求 1](#_Toc155109297)

[1.1背景介绍 1](#_Toc155109298)

[1.2目的及意义 1](#_Toc155109299)

[二、设计思路 2](#_Toc155109300)

[2.1总体设计 2](#_Toc155109301)

[2.2思维导图 2](#_Toc155109302)

[2.3详细设计 3](#_Toc155109303)

[2.3.1库详情 3](#_Toc155109304)

[2.3.2类详情 4](#_Toc155109305)

[三、设计实现及代码分析 6](#_Toc155109306)

[3.1老师端 6](#_Toc155109307)

[3.1.1添加学生 8](#_Toc155109308)

[3.1.2删除学生信息 11](#_Toc155109309)

[3.1.3查询所有学生信息 13](#_Toc155109310)

[3.1.4对学生进行排序 15](#_Toc155109311)

[3.1.5修改学生成绩 20](#_Toc155109312)

[3.1.6退出系统并保存数据 22](#_Toc155109313)

[3.2学生端 22](#_Toc155109314)

[3.2.1登录系统 23](#_Toc155109315)

[3.2.2查询个人信息 24](#_Toc155109316)

[3.2.2修改个人信息 24](#_Toc155109317)

[3.2.3退出并保存数据 25](#_Toc155109318)

[四、遇到的问题及解决方法 26](#_Toc155109319)

[4.1遇到的问题 26](#_Toc155109320)

[4.2解决方法 26](#_Toc155109321)

[五、参考文献 27](#_Toc155109322)

[六、总结与收获 28](#_Toc155109323)

# 一、背景及目的要求

## 1.1背景介绍

成绩管理系统是现代教育管理中至关重要的一环。这款基于Python开发的综合成绩管理系统是为了解决教育领域中学校教务管理的痛点而设计的。在现实生活中，学校教务管理通常面临着繁琐的学生信息录入、管理和成绩记录等问题。这个系统的出现旨在简化教师和学生的操作流程，提高教务工作的效率。

## 1.2目的及意义

教师端的功能设计充分考虑了教学工作的需求。通过这个端口，教师可以便捷地增加、删除和修改学生的基本信息，比如姓名、年龄、性别等，同时也可以随时更新学生成绩，确保信息的及时性和准确性。这一举措不仅节省了教师们大量繁杂的手工操作时间，也提高了信息管理的准确性和可靠性。而对于学生端，其设计更注重学生个人信息的查询与浏览。学生可以轻松地登录系统，查看自己的基本信息，包括个人档案和学业成绩。这样的设计不仅方便了学生随时了解自己的学习情况，也促进了学生对学业的管理和监督，激发了学习的积极性。

在日常生活中，这样的系统对学校管理和教学工作产生了积极的影响。想象一下，教师们不再需要手动整理海量的学生档案和成绩记录，而是通过简单的操作就能完成这项工作。这不仅减少了教师们的工作负担，更使得他们能够将更多的时间和精力投入到教学中去，为学生提供更优质的教育服务。同时，学生也因此获益良多。他们可以更加方便地了解自己的学习情况，及时发现问题并加以改进，提高了学习的自我管理能力。这种及时、有效的信息获取方式，也为学生提供了更好的学习体验，激发了他们更多的学习兴趣和动力。

这款基于Python的综合成绩管理系统不仅在教学管理上发挥了巨大的作用，也为教育行业的信息化发展提供了有力支持。它简化了教务管理流程，提升了工作效率，为学校教育管理带来了全新的体验和可能性。

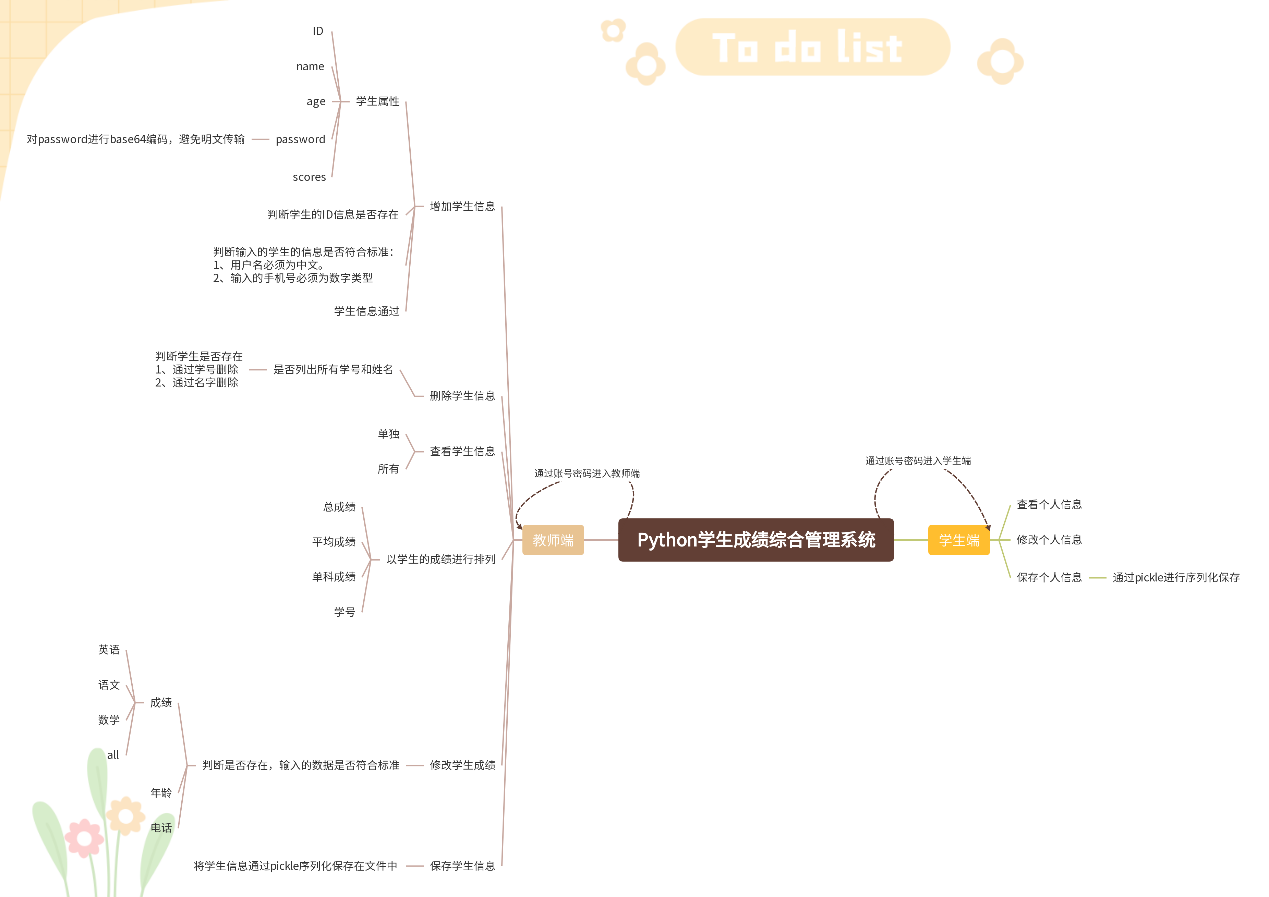
# 二、设计思路

## 2.1总体设计

本次我想设计的系统包括教师端、学生端，这样可以更大的提高该系统在实际中的应用。其中教师端除了基础的增删改查学生的信息，其中个人信息包括：学号、姓名、年龄、password、成绩，其中成绩包括英语成绩、数学成绩、语文成绩，还有对学生进行排序，方便教师对学生排名，当然在排名中也有按照不同的规则，如：平均成绩、总成绩、英语成绩、数学成绩、学号等。在学生端系统中，包括查看个人信息、修改个人信息、保存个人信息，退出系统等功能。

## 2.2思维导图

以下为该系统的脑图：



## 2.3详细设计

### 2.3.1库详情

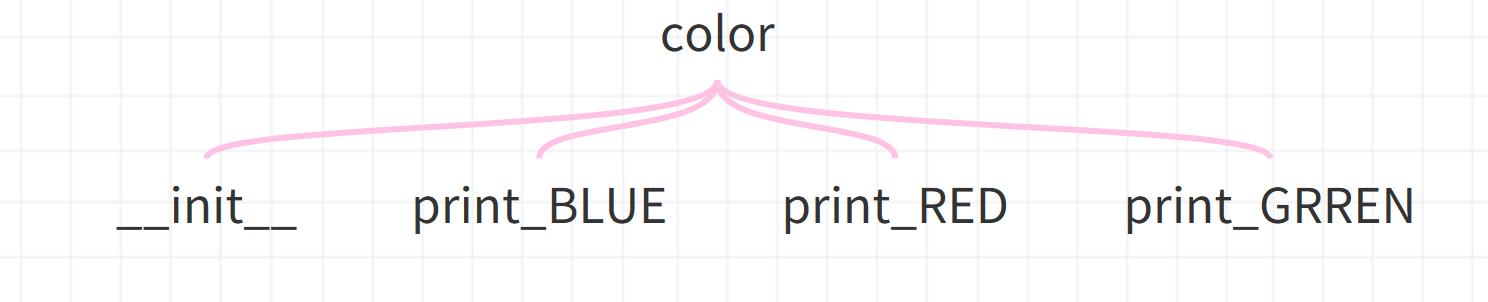
在本项目中主要使用了OS、Colorama、pickle库。其中OS在本项目中主要用于执行系统命令：os.system(‘cls’)，用来清屏，在系统中可能会出现很多提示信息等，会扰乱使用者的感受，因此，调用此函数可以减轻使用者操作压力，给使用者一个清新的视觉体验。

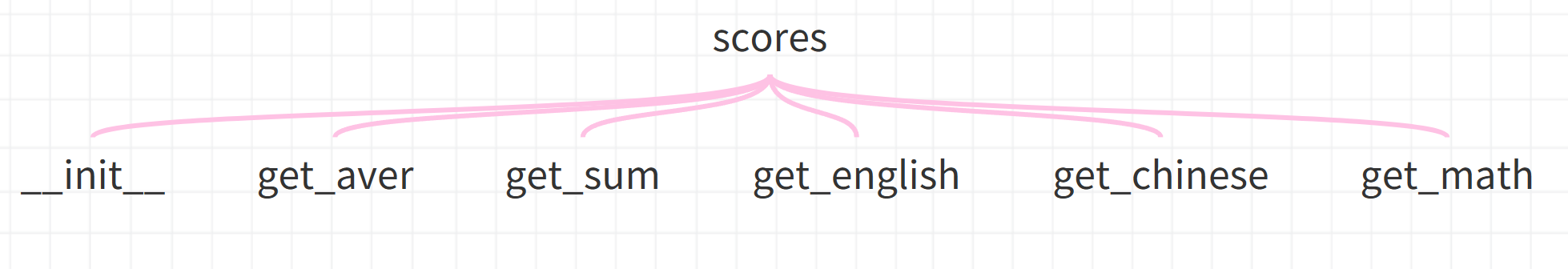
Colorama的作用是改变输出的颜色，就比如在添加用户的时候，正确的添加后，会显示出绿色的“成功添加”，在添加错误后，会显示红色的“添加失败”。在运行程序后，会显示蓝色的文字“欢迎使用系统”类似的欢迎文字。

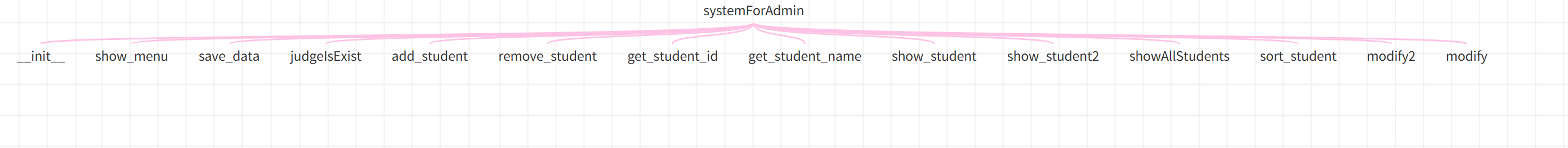
Pickle的作用是用来保存数据，保存数据可以使用文件的形式保存或者使用数据库进行保存，为了方便起见，本系统采用pickle保存。将保存的用户信息和系统配置信息使用pickle.dump保存为二进制文件，当再次打开系统的时候，会调用pickle.load加载二进制文件，将保存的数据加载到对象中。

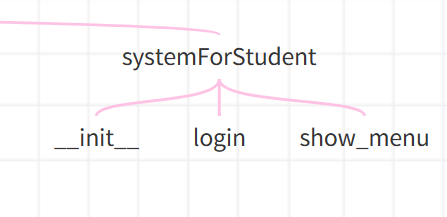
Base64的作用是用来对数据进行编码，在本系统中主要是用来将学生的密码进行base64，这样的话可以将密码不以明文的形式展现出，在一定的程度上保护用户隐私。

### 2.3.2类详情









Color类主要是用来初始化颜色，就比如print\_BLUE(message)函数，将数据以蓝色的形式打印出来，在使用Style.RESET\_ALL，将颜色重置，将下次的输出以原本的颜色打印出来。

Scores类主要是用来保存学生的成绩，在学生的score的属性中new一个对象，将score的数据其实就是一个对象，这样的话后面对学生进行操作的时候更加方便。

Students类主要是用来作为学生，属性包括：学号、姓名、数学英语语文成绩。方法包括get\_id、get\_name、get\_chinese等。

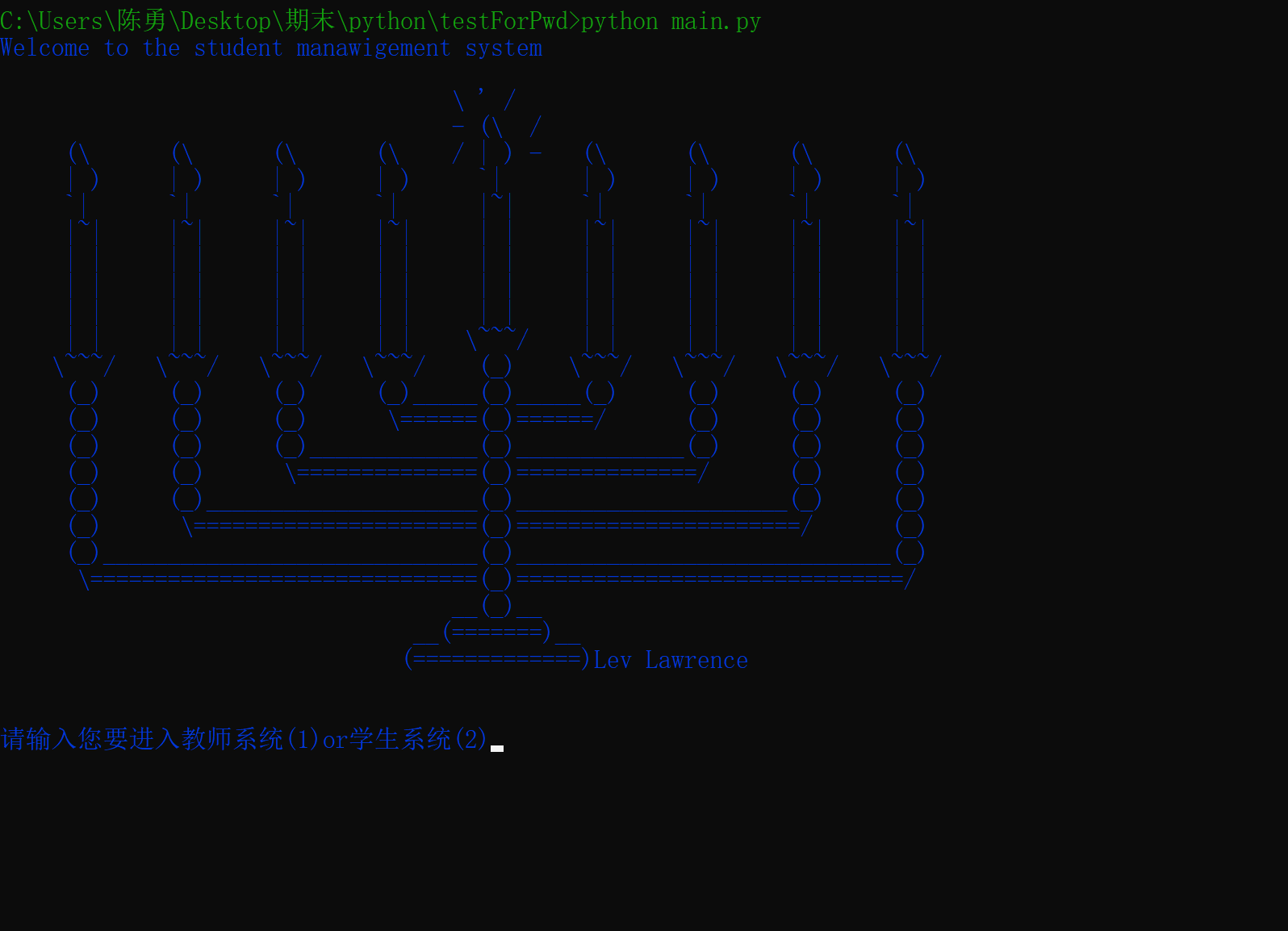
systemForAdmin类是学生管理系统的主类，这个类中包括了添加学生信息、删除学生信息、修改学生信息、更新学生信息、对学生进行排序、保存数据等功能。通过这个类可以将整个学生管理系统变得更具有模块化。在后面使用pickle进行保存数据更加方便。

systemForStudent类是主要给学生使用的，该类中有查询个人信息、修改个人信息、查看个人成绩。通过该类会是学生管理系统更加模块化。

# 三、设计实现及代码分析

## 3.1老师端

启动程序会显示一个欢迎界面，主要是通过调用了Colorama更改了颜色，并且使用ASCII打印出欢迎字符：



if \_\_name\_\_ == "\_\_main\_\_":

colorama.init()

diver = color()

# sysAdmin = systemForAdmin()

# sysAdmin.show\_menu()

# 打印欢迎字符

print(Fore.BLUE + "Welcome to the student manawigement system")

print(

Fore.BLUE

+ """

\ ' /

- (\ /

(\ (\ (\ (\ / | ) - (\ (\ (\ (\\

| ) | ) | ) | ) `| | ) | ) | ) | )

`| `| `| `| |~| `| `| `| `|

|~| |~| |~| |~| | | |~| |~| |~| |~|

| | | | | | | | | | | | | | | | | |

| | | | | | | | | | | | | | | | | |

| | | | | | | | | | | | | | | | | |

| | | | | | | | \~~~/ | | | | | | | |

\~~~/ \~~~/ \~~~/ \~~~/ (\_) \~~~/ \~~~/ \~~~/ \~~~/

(\_) (\_) (\_) (\_)\_\_\_\_\_(\_)\_\_\_\_\_(\_) (\_) (\_) (\_)

(\_) (\_) (\_) \======(\_)======/ (\_) (\_) (\_)

(\_) (\_) (\_)\_\_\_\_\_\_\_\_\_\_\_\_\_(\_)\_\_\_\_\_\_\_\_\_\_\_\_\_(\_) (\_) (\_)

(\_) (\_) \==============(\_)==============/ (\_) (\_)

(\_) (\_)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_) (\_)

(\_) \======================(\_)======================/ (\_)

(\_)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_)

\==============================(\_)==============================/

\_\_(\_)\_\_

\_\_(=======)\_\_

(=============)Lev Lawrence

"""

)

Style.RESET\_ALL

if os.path.exists("systemAdmin.txt"):

file\_systemAdmin = open("systemAdmin.txt", "rb")

sysAdmin = pickle.load(file\_systemAdmin)

else:

sysAdmin = systemForAdmin()

while True:

choice = input("请输入您要进入教师系统(1)or学生系统(2)")

if not choice.isalpha():

print("您的输入存在问题请重新输入！")

os.system("cls")

pass

if choice == "1":

sysAdmin.show\_menu()

pass

elif choice == "2":

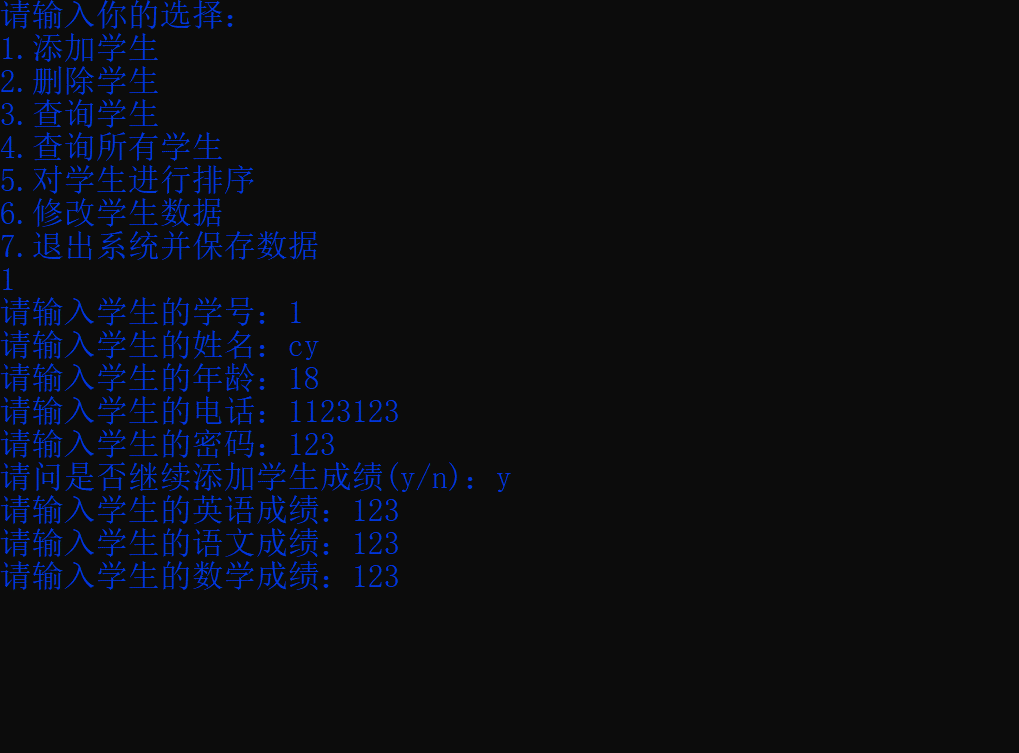
sysStudent = systemForStudent(sysAdmin)

# sysStudent.show\_menu()

pass

### 3.1.1添加学生

教师端点击添加学生后，输入学生的详细信息，并且该信息通过创建Student对象，将对象赋给学生的成绩属性，在进行输入信息的时候，系统会判断输入的信息是否符合规定，如：学号必须为数字，姓名必须为字符串而不能为数字，成绩、年龄、电话等必须为数字而不能为字符串，这样的规定可以增加代码的健壮性，因为在后面的功能中，会频繁使用这些属性和变量，如果不规定这些属性的性质，则在后面的代码中可能会出现非预期的错误。在添加学生的信息时，系统会自动判断该学生的信息是否已经存在数据库中，以防止学号和姓名同时重复。在代码中频繁使用了isdigit和isalpha，这是python自带的函数，可以判断是否为数字或者是否为字符。



def add\_student(self):

try:

id = input("请输入学生的学号：")

if not id.isdigit():

color.print\_RED("学号输入有误，添加失败，请输入数字！！！")

return

name = input("请输入学生的姓名：")

if not name.isalpha():

color.print\_RED("姓名输入有误，添加失败，请输入字符！！！")

return

age = input("请输入学生的年龄：")

if not age.isdigit():

color.print\_RED("年龄输入有误，添加失败，请输入数字！！！")

return

phone = input("请输入学生的电话：")

if not phone.isdigit():

color.print\_RED("电话输入有误，添加失败，请输入数字！！！")

return

password = input("请输入学生的密码：")

if not password.isdigit():

color.print\_RED("密码输入有误，添加失败，请输入数字！！！")

return

# 对密码进行编码

password = base64.b64encode(password.encode("utf-8"))

flag = input("请问是否继续添加学生成绩(y/n)：")

if "y" in flag or "Y" in flag:

english = eval(input("请输入学生的英语成绩："))

chinese = eval(input("请输入学生的语文成绩："))

math = eval(input("请输入学生的数学成绩："))

score = scores(english, chinese, math)

student = students(id, name, age, phone, score, password)

else:

student = students(id, name, age, phone, None, password)

if self.judgeIsExist(student):

self.students.append(student)

os.system("cls")

color.print\_GRREN(f"{name}的信息添加成功！！！")

else:

os.system("cls")

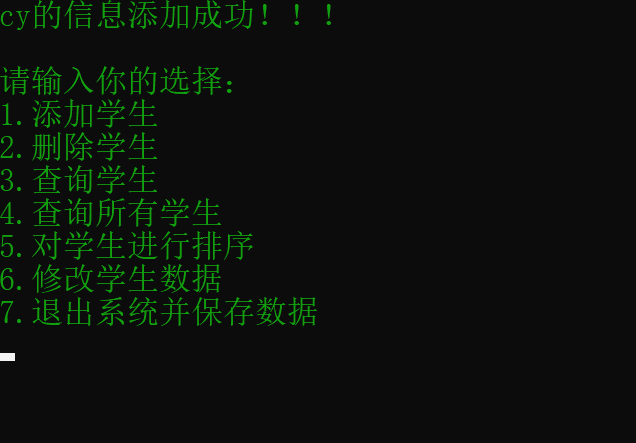
color.print\_RED(f"该学生已存在，添加失败！！！")

except Exception as e:

color.print\_RED(f"输入有误，添加失败！！！")

print(e)

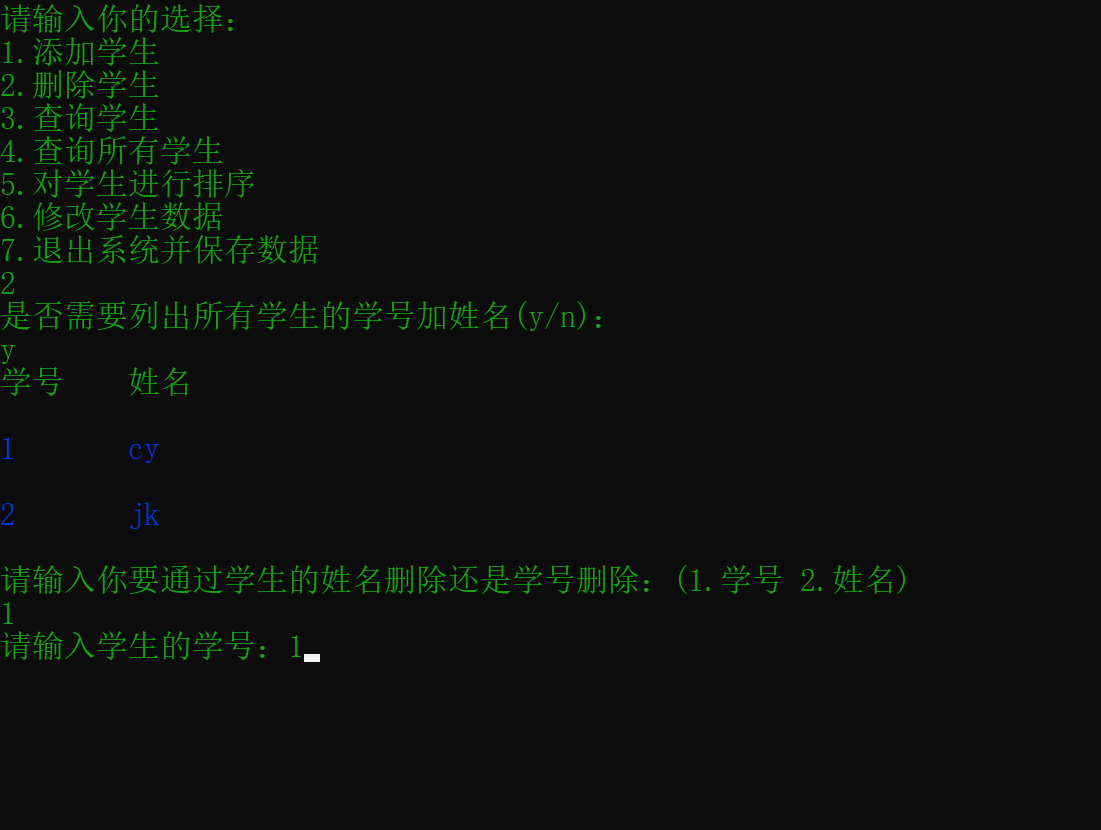
pass



添加信息成功

### 3.1.2删除学生信息

教师在需要删除学生信息的情况下，可以通过系统首先将所有学生的姓名和学号清晰地列出来。这样做的好处是，教师能够直观地看到有哪些学生需要被删除，有助于他们做出准确的判断。此外，教师还有权选择是否使用这一功能，充分尊重了教师的使用习惯和自主权。

该系统不仅支持按照学号进行删除，还支持按照姓名进行删除，为用户提供了更大的便利性和灵活性。在执行删除操作之前，系统会向用户明确提示，要求用户选择按照学号删除还是按照姓名删除。这样的设计有助于避免操作中的混淆和错误，提高了系统的用户友好性和准确性。

def remove\_student(self):

print("是否需要列出所有学生的学号加姓名(y/n)：")

choice = input()

if "y" in choice or "Y" in choice:

color.print\_GRREN("学号\t姓名")

for student in self.students:

color.print\_BLUE(f"{student.get\_id()}\t{student.get\_name()}")

print("请输入你要通过学生的姓名删除还是学号删除：(1.学号 2.姓名)")

choice = input()

if choice == "1":

id = input("请输入学生的学号：")

if not id.isdigit():

os.system("cls")

color.print\_RED("学号输入有误，删除失败，请输入数字！！！")

return

student = self.get\_student\_id(id)

if student:

name = student.get\_name()

self.students.remove(student)

os.system("cls")

color.print\_GRREN(f"{name}的信息删除成功！！！")

else:

os.system("cls")

color.print\_RED(f"该学生不存在，删除失败！！！")

elif choice == "2":

name = input("请输入学生的姓名：")

if not name.isalpha():

os.system("cls")

color.print\_RED("姓名输入有误，删除失败，请输入字符！！！")

return

student = self.get\_student\_name(name)

if student:

self.students.remove(student)

os.system("cls")

color.print\_GRREN(f"{name}的信息删除成功！！！")

else:

os.system("cls")

color.print\_RED(f"该学生不存在，删除失败！！！")

else:

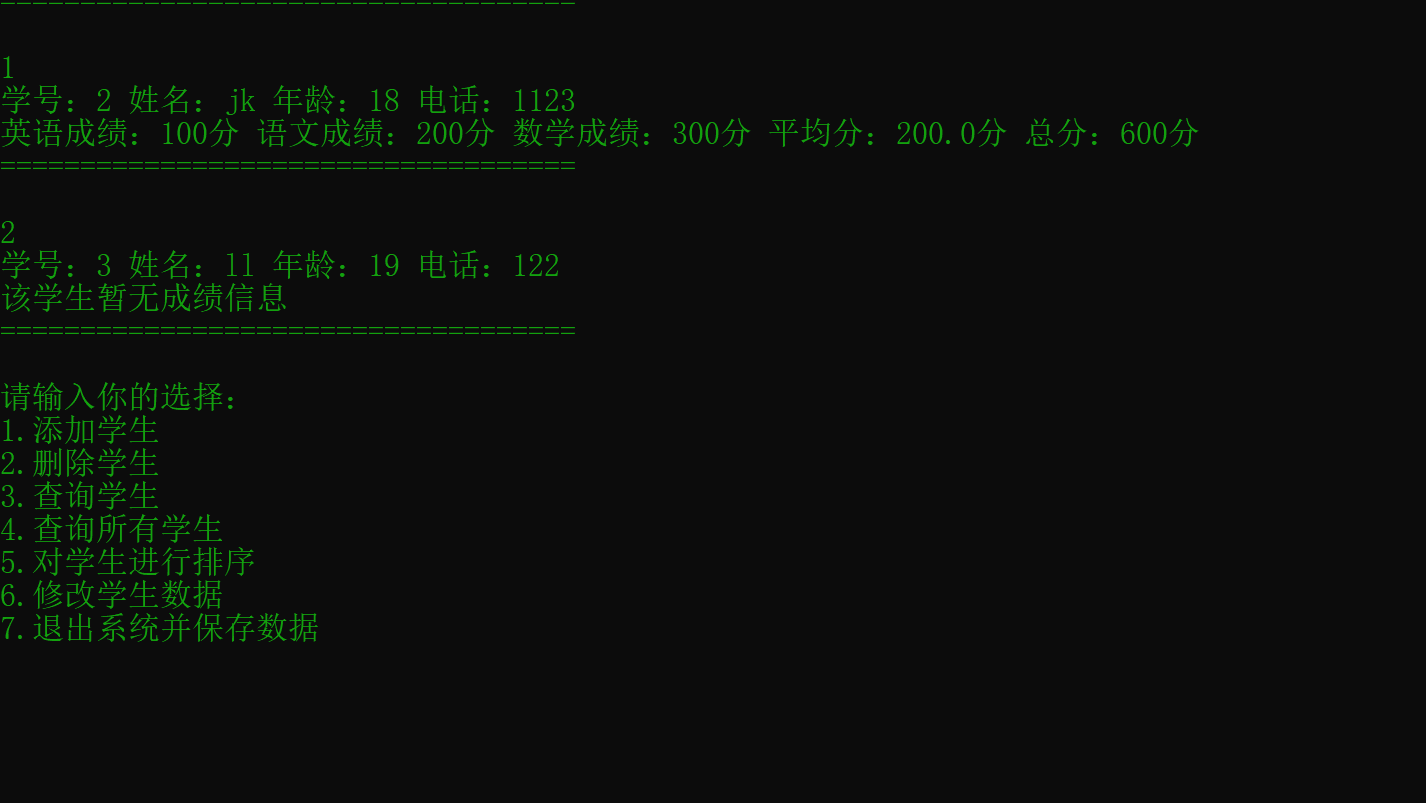
os.system("cls")

color.print\_RED("输入错误，请重新输入")

### 3.1.3查询所有学生信息

教师在使用该系统时，可以一键查询所有学生的信息。这些信息会按照预设的格式展示出来，包括学号、姓名、年龄、电话和成绩等。这样的设计使得教师能够快速、准确地获取到所有学生的信息，便于他们进行管理和教学工作。

除了查询所有学生的信息，该系统还支持单独查询功能。教师可以根据需要，输入学号或者姓名进行查询。这一功能为教师提供了更大的灵活性，他们可以根据具体需求，快速查询到特定的学生信息。

值得一提的是，该系统默认按照添加学生的顺序展示信息。这种排序方式使得教师能够更直观地了解学生的情况，有助于他们进行更有针对性的教学和管理工作。

def showAllStudents(self):

os.system("cls")

count = 1

for student in self.students:

color.print\_GRREN("====================================")

print(count)

count += 1

print(

f"学号：{student.get\_id()} 姓名：{student.get\_name()} 年龄：{student.get\_age()} 电话：{student.get\_phone()}"

)

if student.get\_scores() != None:

print(

f"英语成绩：{student.get\_scores().get\_english()}分 语文成绩：{student.get\_scores().get\_chinese()}分 数学成绩：{student.get\_scores().get\_math()}分 平均分：{student.get\_scores().get\_aver()}分 总分：{student.get\_scores().get\_sum()}分"

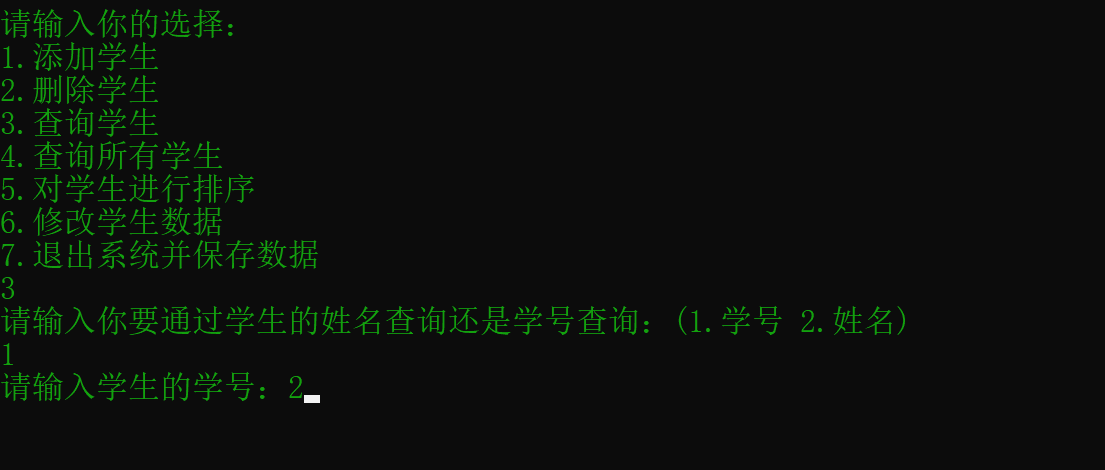
)

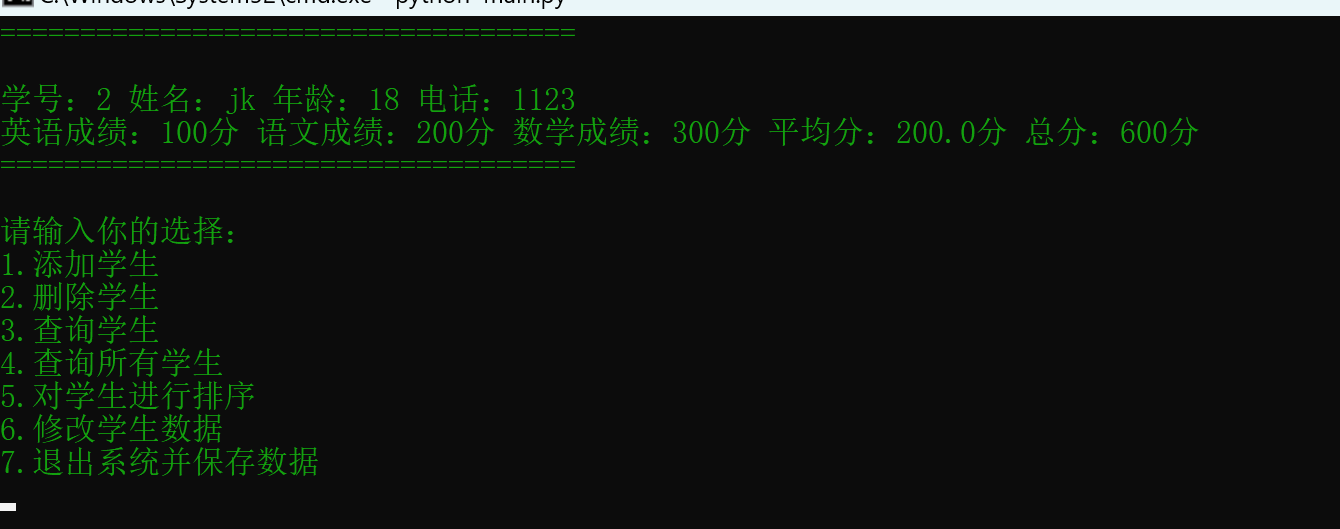
else:

print("该学生暂无成绩信息")

color.print\_GRREN("====================================")

查询学生信息





### 3.1.4对学生进行排序

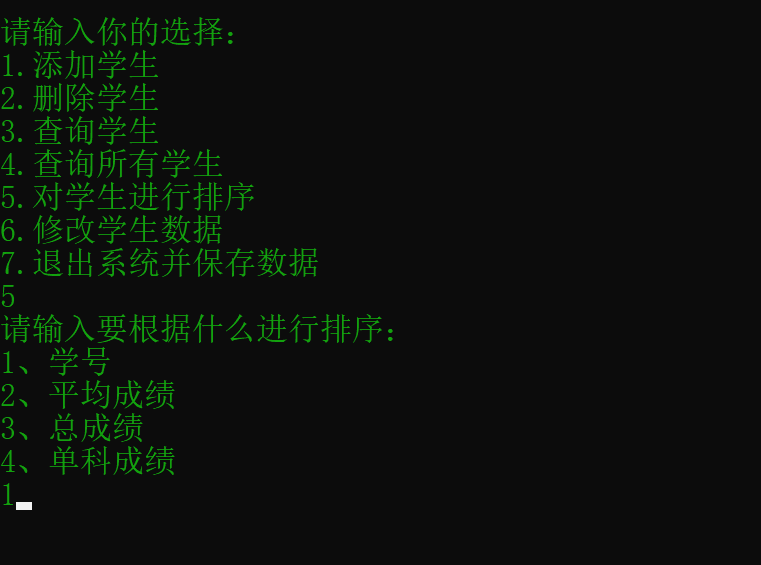
该功能是一个强大的学生信息排序工具，它允许教师按照一定的标准对学生的所有信息进行有序排列。这一功能设计的主要目的是为了方便教师查看和分析学生的各项数据。

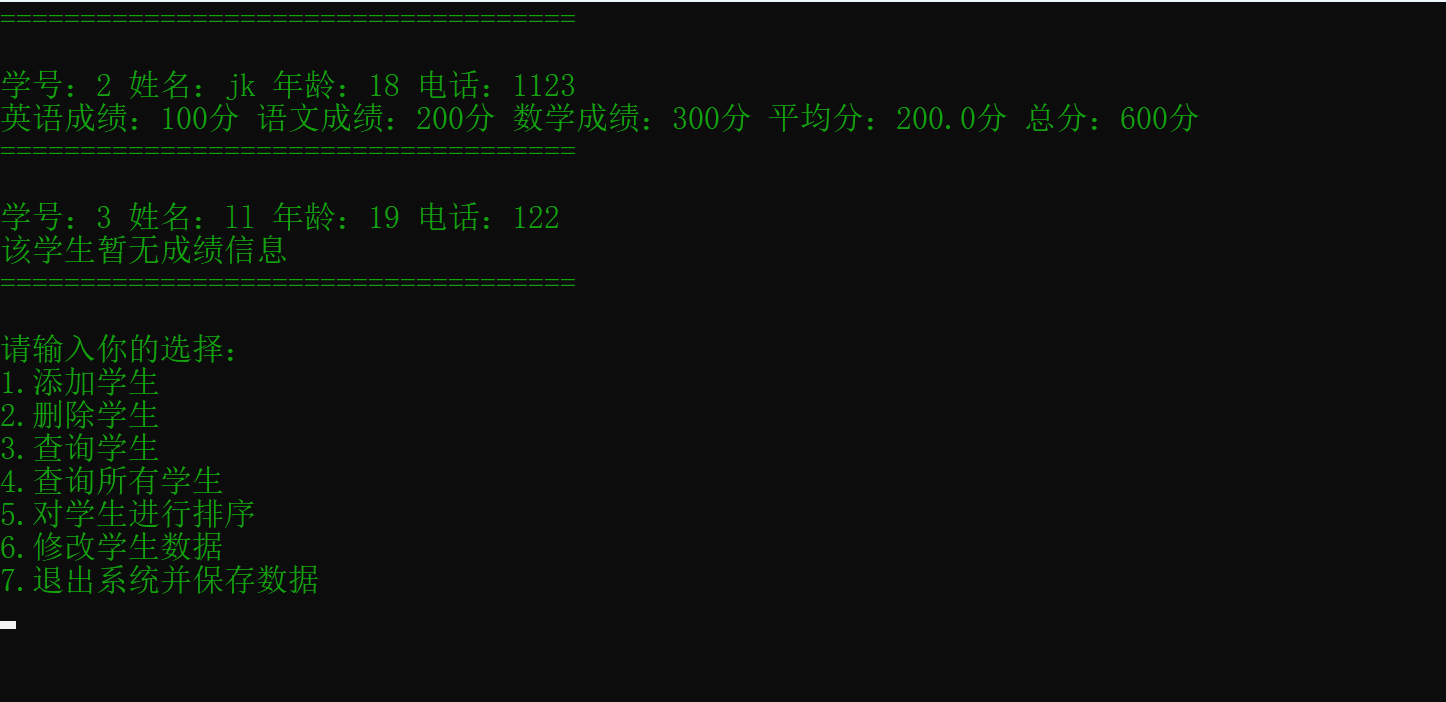
教师可以选择按照学号、平均成绩、总成绩或单科成绩进行排序。其中，单科成绩进一步细分为数学成绩、英语成绩和语文成绩。这样的分类使得教师能够更深入地了解学生在各个学科上的表现。

例如，当教师需要对学生的成绩进行分析时，该功能就显得尤为有用。通过对学生单科成绩的排序，教师可以迅速识别出哪些学生在某一学科上表现优异，哪些学生需要额外的帮助。这不仅有助于教师制定更有针对性的教学计划，还能够为学生提供更个性化的学习支持。

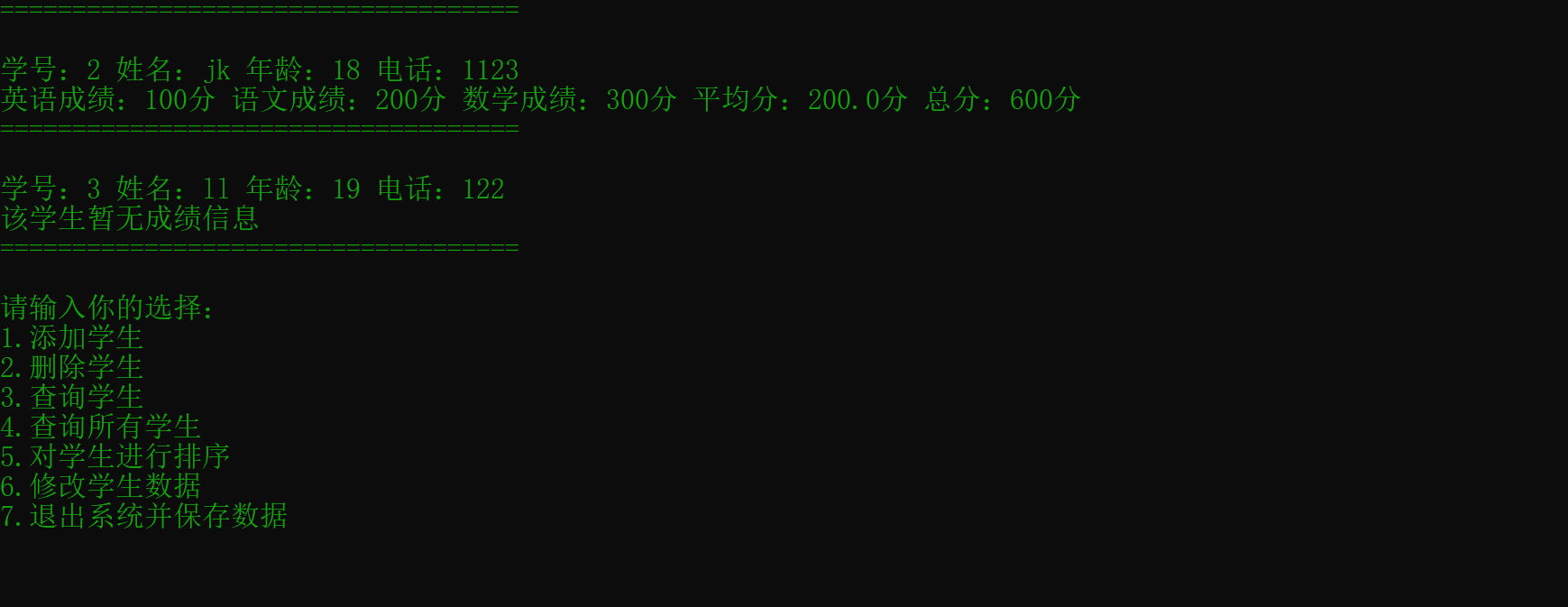
此外，该功能还具有很高的灵活性。教师可以根据需要选择不同的排序标准，从而得到不同的视图和数据解读。这种灵活性使得该功能在各种场景下都能够发挥巨大的作用，无论是进行日常教学管理还是进行深度的数据分析。

按照学号进行排序：

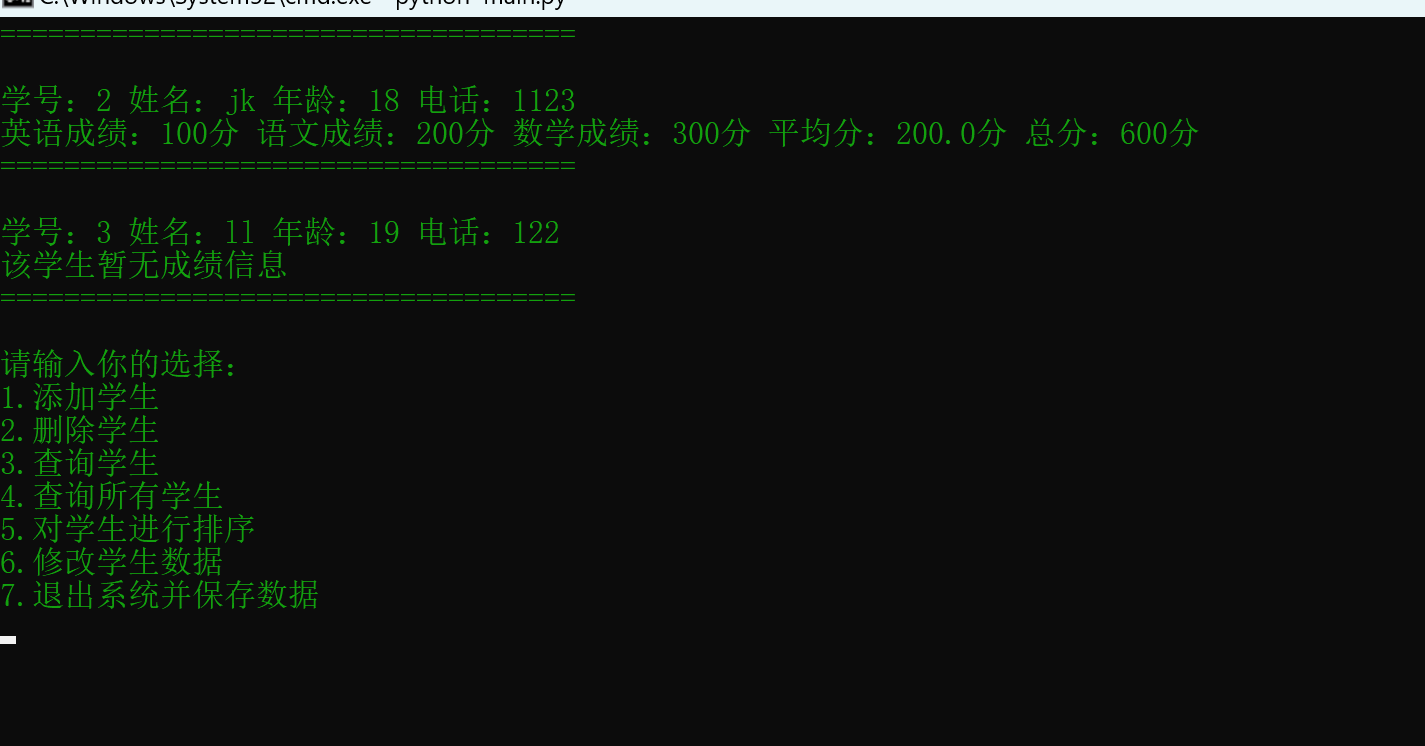




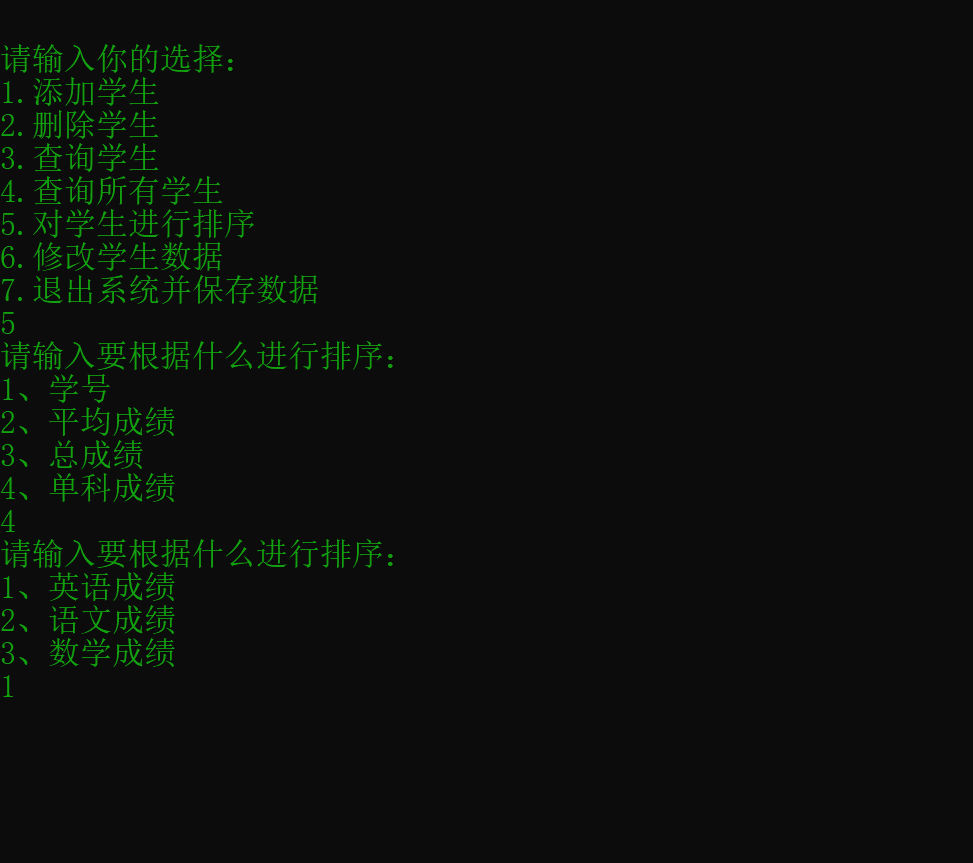
按照平均成绩进行排序：

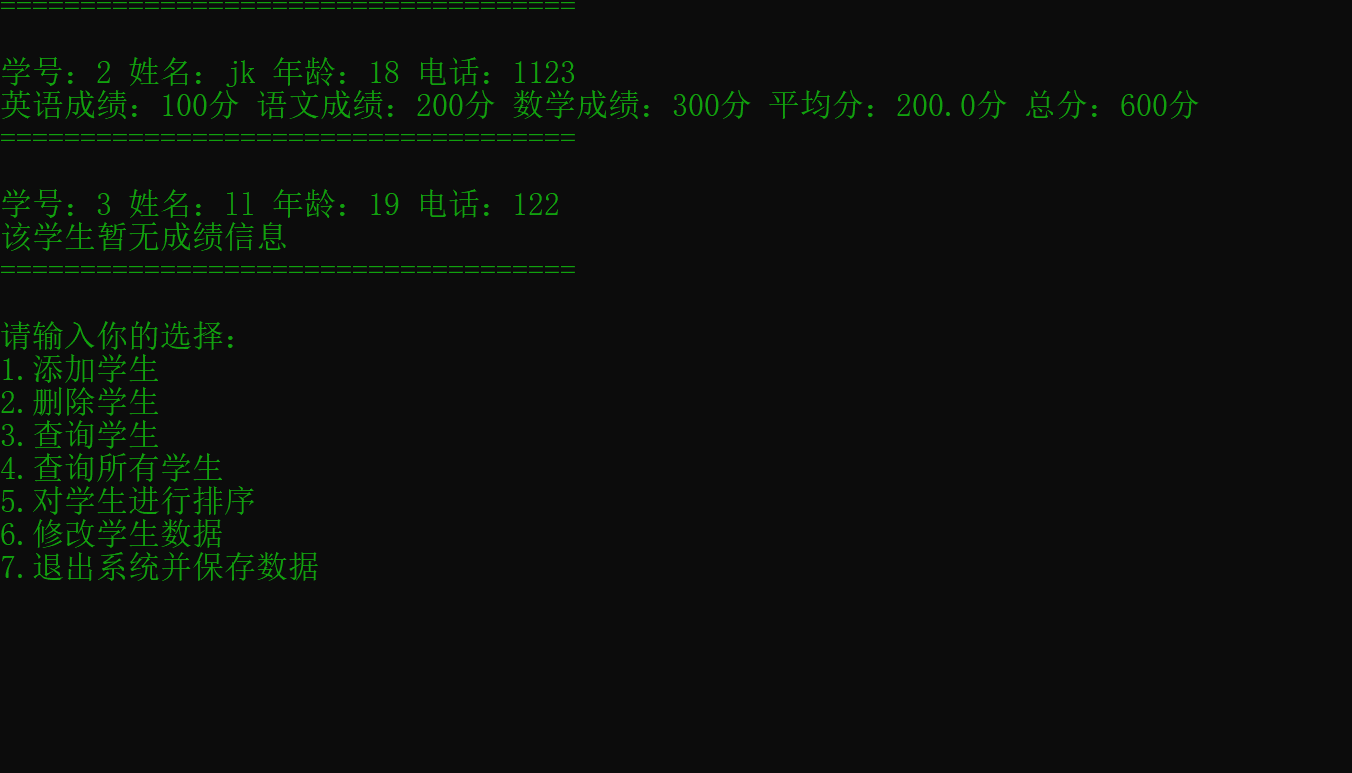


按照总成绩进行排序：



按照单科成绩进行排序：





主要代码如下：

def sort\_student(self):

print("请输入要根据什么进行排序：")

print("1、学号")

print("2、平均成绩")

print("3、总成绩")

print("4、单科成绩")

choice = input()

if choice == "1":

self.students\_tmp = sorted(self.students, key=lambda x: x.get\_id())

os.system("cls")

for student in self.students\_tmp:

color.print\_GRREN("====================================")

print(

f"学号：{student.get\_id()} 姓名：{student.get\_name()} 年龄：{student.get\_age()} 电话：{student.get\_phone()}"

)

if student.get\_scores() != None:

print(

f"英语成绩：{student.get\_scores().get\_english()}分 语文成绩：{student.get\_scores().get\_chinese()}分 数学成绩：{student.get\_scores().get\_math()}分 平均分：{student.get\_scores().get\_aver()}分 总分：{student.get\_scores().get\_sum()}分"

)

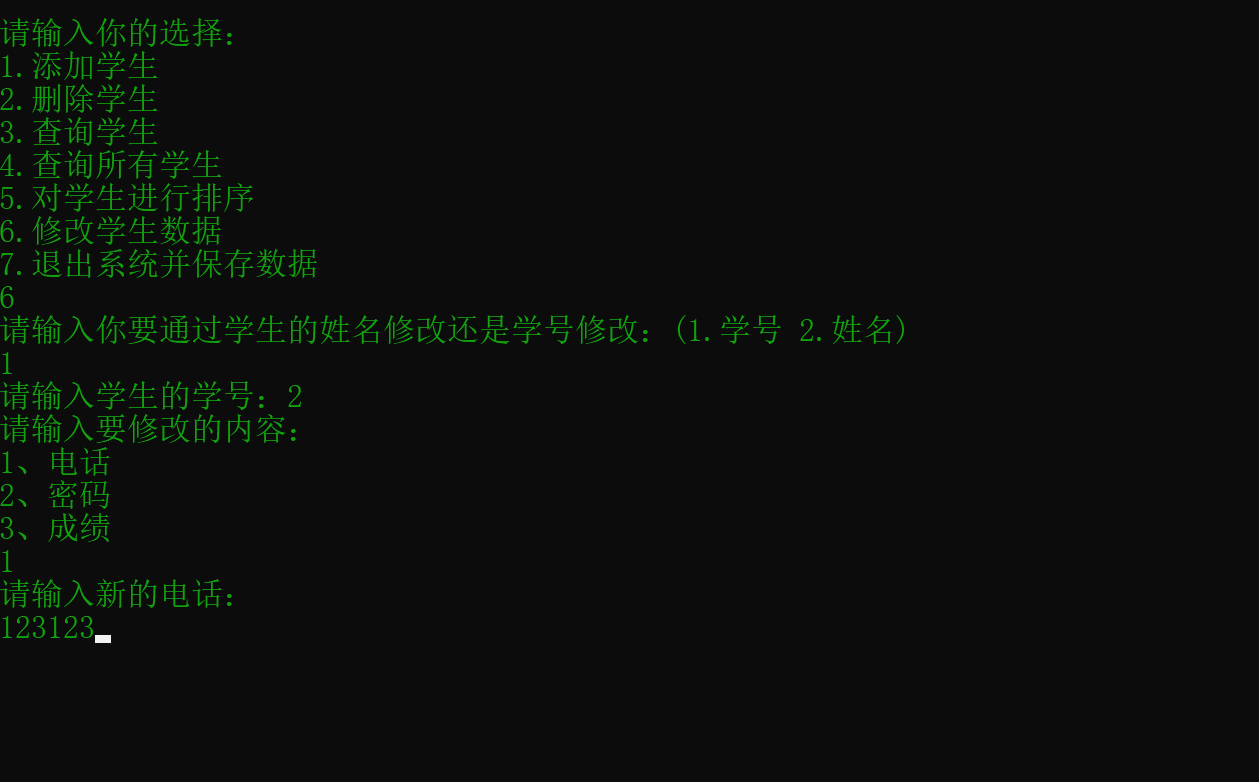
else:

print("该学生暂无成绩信息")

color.print\_GRREN("====================================")

### 3.1.5修改学生成绩

教师在需要修改学生成绩时，可以通过该功能进行操作。这一功能的设计旨在提高教师对学生信息管理的便利性和准确性。首先，教师可以选择按照学号或姓名来修改成绩。这样的选择方式考虑到了教师可能需要根据不同的情况来查找和修改学生成绩。选择学号或姓名后，系统会提示教师需要修改哪些信息，如电话、密码和成绩等。其中，成绩又进一步细分为数学成绩、英语成绩和语文成绩。这样的分类使得教师能够更加精确地修改学生的各科成绩。通过这一功能，教师不仅可以方便地管理学生的基本信息，还能够更快捷地对学生的成绩进行修改和更新。

该功能的设计充分考虑到了教师的实际需求和使用习惯，旨在提供更加人性化和高效的管理工具。通过这一功能，教师能够更加轻松地完成对学生信息的日常管理和维护工作，从而提高工作效率和准确性。

部分主要代码：

def modify(self):

print("请输入你要通过学生的姓名修改还是学号修改：(1.学号 2.姓名)")

choice = input()

if choice == "1":

id = input("请输入学生的学号：")

if not id.isdigit():

os.system("cls")

color.print\_RED("学号输入有误，修改失败，请输入数字！！！")

return

student = self.get\_student\_id(id)

if not student:

os.system("cls")

color.print\_RED(f"该学生不存在，修改失败！！！")

return

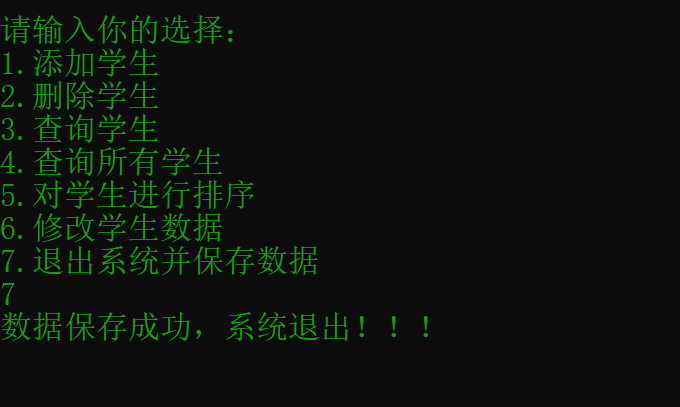
self.modify2(student)

os.system("cls")

color.print\_GRREN(f"{student.get\_name()}的信息修改成功！！！")

### 3.1.6退出系统并保存数据

点击退出系统后，系统会自动保存用户的所有操作，并序列化存放到由pickle生成的文件中，如果用户再次使用该系统时，用户可以接着上次的操作进行，而不用再次输入数据。同样这样的功能，更贴合实际。





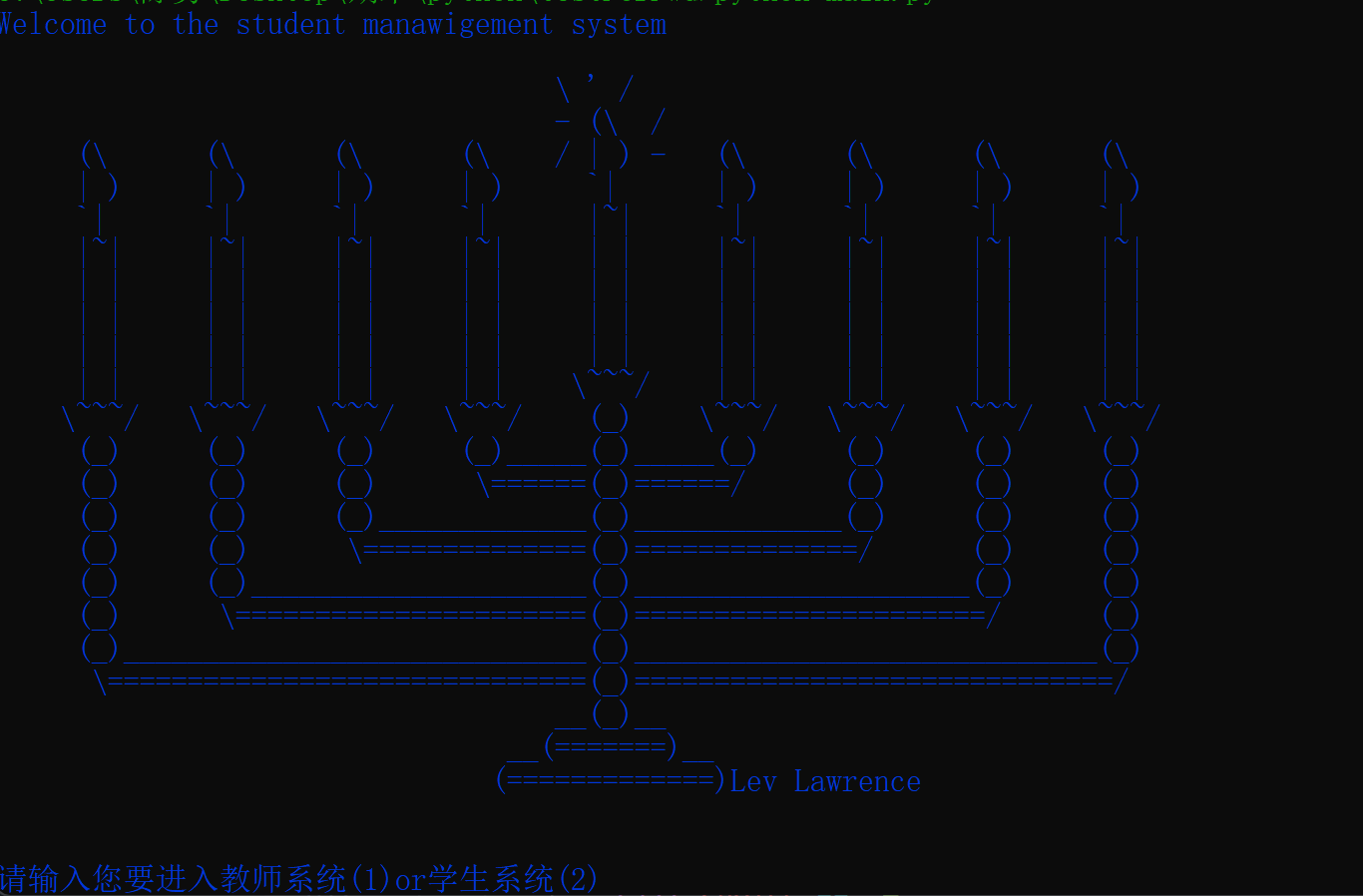
def save\_data(self):

file = open("systemAdmin.txt", "wb")

pickle.dump(self, file)

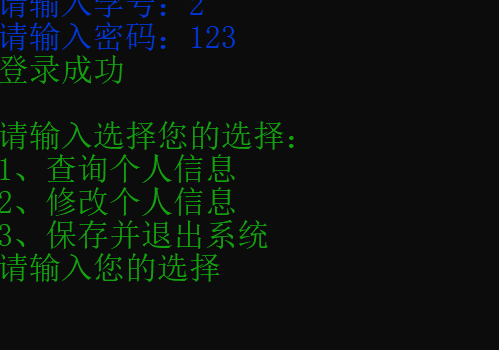
file.close()

## 3.2学生端



### 3.2.1登录系统

通过输入学生的学号和密码，系统会自动在系统中查询信息，如果正确，则会登录成功，否则则会失败，并返回主界面。



def login(self):

id = input("请输入学号：")

if not id.isdigit():

color.print\_RED("出现错误，请输入数字")

return

student = self.systemAdmin.get\_student\_id(id)

if not student:

color.print\_RED("未查询到学生相关信息")

return

password = input("请输入密码：")

# 对密码进行编码

password = base64.b64encode(password.encode("utf-8"))

if student.get\_password() != password:

color.print\_RED("密码错误")

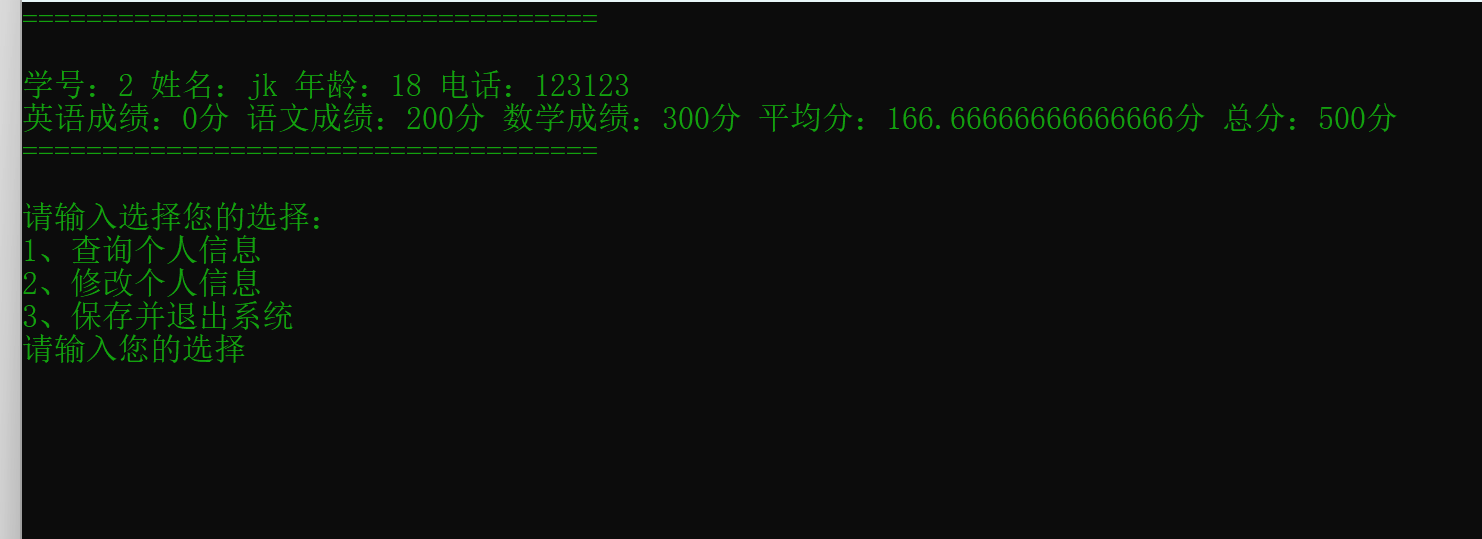
return

color.print\_GRREN("登录成功")

return student

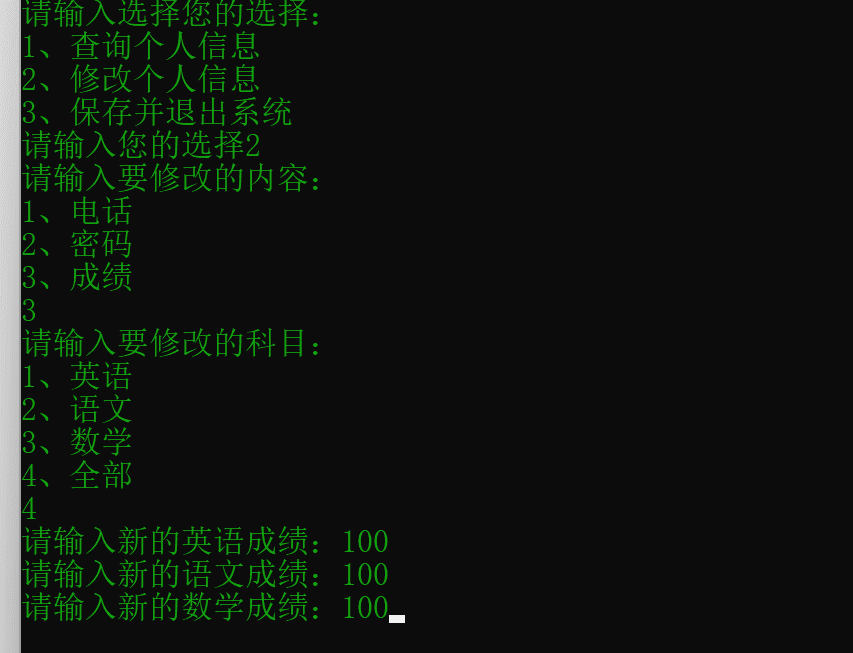
### 3.2.2查询个人信息

在查询个人信息的时候，学生可以查看到自己的所有信息，学号、姓名、年龄、电话，成绩等。以及和成绩有关的所有结果，单科成绩、平均分、总分等，更加直观的展现自己的学习情况。



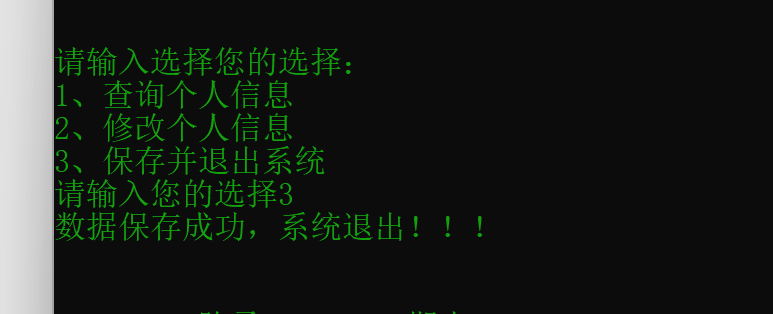
### 3.2.2修改个人信息

修改个人信息，和教师端的差不多，也都是修改电话、密码、成绩等。



### 3.2.3退出并保存数据

和教师端类似。



# 四、遇到的问题及解决方法

## 4.1遇到的问题

1.首先最先遇到的问题当然是输出的颜色问题。我想要一个功能就是系统返回的信息是错误或者正确的提示信息更加具有代表性，如红色代表失败、绿色代表成功、蓝色代表欢迎。还有就是添加不同的颜色可以使用户更加便于区分，减少审美疲劳，如果全是一个颜色的话，用户看的肯定会难受，所以添加一点颜色会使用户眼前一亮。提高用户的使用感。

2.还有就是如何实现老师端和学生端的分离，通过将老师端和学生端分离可以提高学生管理系统的可扩展性，可以更加方便添加新功能。

## 4.2解决方法

1.颜色的问题，我是通过使用Colorma库函数进行输出的。

2.老师端和学生端的分离是通过，编写两个类，分别为systemForStudent和systemForAdmin两类。

# 五、参考文献

[1] 刘秀丽 基于Python语言的职工信息管理系统设计与实现 . 现代信息科技 2023

[2] 姚松言 基于Python编程的智慧图书馆信息管理系统 . 信息与电脑 2023

[3] 常见 基于大数据的高校学生信息化管理系统设计 . 科技创新与生产力 2023

# 六、总结与收获

在这学期的Python课程中，我经历了一个令人兴奋的学习过程。从最初的基础概念到最后的期末项目——学生管理系统，我的编程技能和对Python语言的理解都有了显著的提高。这学期让我对Python语言的掌握程度有了很大的进步。我学会了如何定义函数、使用循环和条件语句，并且更熟练地处理数据结构。这些基础的知识为我在期末项目中的实践打下了坚实的基础。而最令我收获良多的部分是期末项目的实践经验。通过开发学生管理系统，我不仅学会了如何管理数据和处理用户输入，还掌握了设计用户界面的技巧。这让我更清晰地理解了Python编程在实际场景中的应用，为我未来更复杂项目的开发提供了宝贵的经验。在项目的过程中，遇到了许多问题，但也学会了如何去解决它们。有时是简单的语法错误，有时是复杂的逻辑问题，但每一次解决都让我更加自信，并提高了我的问题解决能力。如果项目是与团队合作完成的，那么团队合作与沟通也是一个重要的方面。通过与同学们的协作，我学到了如何更好地分工合作、有效沟通并最终完成项目。这种合作精神和技能对于未来的团队工作是非常宝贵的。这学期的学习经历不仅提升了我的编程能力，还激发了我对技术和编程更深层次的兴趣。我意识到自己对于这个领域的热情，并且计划在未来更深入地探索编程和技术的世界。

这学期的Python课程为我带来了巨大的收获。它不仅加强了我的编程技能，还培养了我的解决问题能力、团队合作能力和对未来的规划。我期待着在未来将所学应用到更多有趣的项目中，并继续不断地学习和成长。

**结课成绩表**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **学生姓名** |  | **班级** |  | | **任课教师** | |  |
| **题目** | 综合成绩管理系统 | | | | | | |
| **评分标准** | | | | **总分** | | **得分** | |
| 1.报告选题符合要求、层次清晰、结构合理；逻辑性强、文字流畅、图文格式规范。 | | | | 20分 | |  | |
| 2.报告中能完成一个简单系统的设计。 | | | | 30分 | |  | |
| 3.报告中利用Python完成系统的编码。 | | | | 30分 | |  | |
| 4.报告总结真诚，并有反思和收获。 | | | | 10分 | |  | |
| 5.报告完整、内容充实。 | | | | 10分 | |  | |
| **合计** | | | |  | | | |
| 任课教师签字： 年 月 日 | | | | | | | |

注：1.评分标准栏应根据课程的不同进行适当的调整，包括评分标准选项的数量、内容、分值等；

2.评分标准要与考查课考试审批中的评分标准保持一致；

3.该表由学生打印并填写抬头内容，装订在学生考查课材料的最后一页。