



西安交通大学  
XI'AN JIAOTONG UNIVERSITY

数据库基础及应用课程

# 大作业

项目名称：学校信息管理系统

完成人： 汪洋

完成日期： 2023 年 5 月 3 日

指导教师： 赵英良

# 目录

<b>1 软件功能</b>	<b>2</b>
1.1 学生信息管理	2
1.2 课程信息管理	2
1.3 选课信息管理	2
1.4 成绩管理	2
<b>2 软件的运行环境</b>	<b>3</b>
<b>3 环境搭建说明</b>	<b>3</b>
<b>4 软件设计说明</b>	<b>4</b>
4.1 数据库设计说明	4
4.2 函数设计说明	5
<b>5 软件使用说明</b>	<b>34</b>
5.1 学生信息管理	35
5.2 课程信息管理	36
5.3 选课信息管理	38
5.4 成绩管理	39
<b>6 后记</b>	<b>43</b>
<b>7 致谢</b>	<b>43</b>

# 1 软件功能

## 1.1 学生信息管理

1. **添加学生信息**：允许教务处录入学生学籍信息。适用于入学季新生入籍、交换生学籍交接、转学申请等。
2. **查询学生信息**：允许用户根据学生的某一项信息或某几项信息，查询相应的学生信息。适用于活动信息统计、中学反馈等。
3. **更新学生信息**：允许用户根据学生的某一项信息或某几项信息，更新相应的学生信息。适用于学籍修改、转专业、转班级等。
4. **删除学生信息**：允许教务处根据学生的某一项信息或某几项信息，删除相应的学生信息。适用于学生转学、退学等。

## 1.2 课程信息管理

1. **添加课程信息**：允许教务处录入课程信息，将这些信息存储到数据库中。适用于学院开课等。
2. **查询课程信息**：允许用户根据课程的某一项信息或某几项信息，查询相应的课程信息。适用于学生选课参考、教务处安排课表等。
3. **更新课程信息**：允许用户根据课程的某一项信息或某几项信息，更新相应的课程信息。适用于教师调换、学分修改、先修课程修改等。
4. **删除课程信息**：允许用户根据课程的某一项信息或某几项信息，删除相应的课程信息。适用于教师退休、课程停开等。

## 1.3 选课信息管理

1. **添加选课信息**：允许学生选课。适用于学生选课等。
2. **查询选课信息**：允许学生查看他们已经选择的课程，教师查看选择他们课程的学生。适用于学生上课统计、教师考勤管理等。
3. **更新选课信息**：允许学生修改他们已经选择的课程。适用于学生改选课等。
4. **删除选课信息**：允许学生取消他们已经选择的课程。适用于学生退选课等。

## 1.4 成绩管理

1. **添加成绩信息**：允许教师录入学生成绩。适用于教师登记学生成绩。
2. **查询成绩信息**：允许用户根据选课的某一项信息或某几项信息，查询学生成绩、查询课程成绩。适用于学生、教师查成绩。
3. **更新成绩信息**：允许用户根据选课的某一项信息或某几项信息，更新相应的成绩信息。适用于教师修改成绩。
4. **删除成绩信息**：允许用户根据选课的某一项信息或某几项信息，删除相应的成绩信息。适用于因特殊原因取消学生考试成绩。
5. **成绩信息分析**：
  - (1) 学生成绩分析：允许用户根据学号、学年、学期，计算学生最高分、最低分、平均分（考虑课程学分权重）、绩点（最高 4.3）、优秀率（90 分以上）、及格率（60 分以上）、不及格率（60 分以下）、优秀门数（90 分以上）、及格门数（60 分以上）、不及格门数（60 分以下）、总门数。
  - (2) 课程成绩分析：允许根据课程号、学年、学期，计算课程最高分、最低分、平均分、优秀率（90 分以上）、及格率（60 分以上）、不及格率（60 分以下）、优秀人数（90 分以上）、及格人数（60 分以上）、不及格人数（60 分以下）、总人数。

注：绩点对应：95-100 分为 4.3，90-95 分为 4.0，85-90 分为 3.7，81-85 分为 3.3，78-81 分为 3.0，

75-78 分为 2.7, 72-75 分为 2.3, 68-72 分为 2.0, 64-68 分为 1.7, 60-64 分为 1.3, 小于 60 分为 0, GPA 采取平均学分绩计算方法。

## 2 软件的运行环境

1. Python3
2. 华为 ECS, openEuler OS
3. openGauss(Sqlite): DB
4. putty: 远程连接工具
5. Python 第三方库:
  - (1) tkinter: GUI
  - (2) messagebox: 消息窗口库
  - (3) tkfont: 字体库
  - (4) ttk: 下拉菜单库
  - (5) scrolledtext: 滚动文本库
  - (6) datetime: 日期函数库
  - (7) sqlite3: 连接 sqlite3 的库
  - (8) pycopg2: 连接 opengauss 的库

## 3 环境搭建说明

1. 安装 Python3: <https://www.python.org/>。
2. 安装 Putty: <https://www.putty.be/>。
3. 申请华为云 ECS, 安装 OpenGauss 数据库: <https://www.huaweicloud.com/>。
4. 完成 Putty 远程连接 OpenGauss 数据库的相关配置 (详见文档)。
5. 完成 Python 连接 OpenGauss 数据库的相关设置。
  - (1) 设置防火墙安全规则: 控制台 → 弹性云服务器 → 安全组 → 入方向规则 → 添加规则:  
优先级: 10, 策略: 允许, 协议端口: TCP、26000, 类型: IPv4, 源地址: 0.0.0.0/0。
  - (2) 设置 DB 加密方式和监听 IP  
修改密码加密方式

```
1 [omm@ecs -c9bf ~]$ gs_guc set -I all -c "password_encryption_type=1"
```

修改 DB 监听地址

```
1 [omm@ecs -c9bf ~]$ vi /gaussdb/data/db1/postgresql.conf
2 listen_addresses='10.0.0.15'
3 local_bind_address = '10.0.0.15'
4 #修改为
5 listen_addresses='*'
6 local_bind_address='*'
```

修改登录主机信任方式 (允许远程访问, 在 IPV4 下面)

```
1 vi /gaussdb/data/db1/pg_hba.conf
2 #增加
3 host all all 192.168.56.1/24 md5 #虚拟机使用
4 host all all 0.0.0.0/0 md5 #ECS使用
```

6. 在 OpenGauss 中设计数据库。
7. 在 Python 中设计程序。

## 4 软件设计说明

### 4.1 数据库设计说明

数据库名称: dbapp。

数据库用户名: xjtudba, 密码: xjtu@123。(可以使用 sqlite3, 不需密码)

数据表名称: student、course、sc。

数据表结构:

列名	说明	数据类型	约束
sid	学号	字符串, 20	主键
sname	姓名	字符串, 40	非空
gender	性别	字符串, 10	男或女
age	年龄	int	
sclass	班级	字符串, 20	
school	学院	字符串, 40	
birthday	生日	date	
province	籍贯	字符串, 40	

表 1: student

列名	说明	数据类型	约束
cid	课程号	字符串, 20	主键
cname	课程名	字符串, 40	
teacher	教师	字符串, 20	
credit	学分	float	
cpid	先修课程	字符串, 20	

表 2: course

列名	说明	数据类型	约束
sid	学号	字符串, 20	联合主键, 外键, 引用 student 表 sid
cid	课程号	字符串, 20	联合主键, 外键, 引用 course 表 cid
year	学年	字符串, 10	
term	学期	字符串, 20	第一学期、第二学期或小学期
grade	成绩	int	

表 3: sc

数据库设计代码:

```
1 [omm@ecs-5138 ~]$ gs_om -t start
2 [omm@ecs-5138 ~]$ gsql -p 26000 -d postgres -U omm -r
3 postgres=# create user xjtudba with password 'xjtu@123';
4 postgres=# create database dbapp owner xjtudba;
5 postgres=# \c dbapp
6 dbapp=# create schema xjtudba authorization xjtudba;
7 dbapp=# set search_path to xjtudba;
8
```

```

9 dbapp=# create table student (
10 sid varchar(20) primary key,
11 sname varchar(40) not null,
12 gender varchar(10), check (gender = '男' or gender = '女'),
13 age int,
14 sclass varchar(20),
15 school varchar(40),
16 birthday date,
17 province varchar(40));
18 dbapp=# create table course (
19 cid varchar(20) primary key,
20 cname varchar(40),
21 teacher varchar(20),
22 credit float,
23 cpid varchar(20));
24 dbapp=# create table sc (
25 sid varchar(20),
26 cid varchar(20),
27 year varchar(10),
28 term varchar(20), check (term = '第一学期' or term = '第二学期' or term = '小学期'),
29 grade int,
30 primary key (sid, cid),
31 foreign key (sid) references student(sid),
32 foreign key (cid) references course(cid);

```

## 4.2 函数设计说明

```

1 import tkinter as tk
2 from tkinter import messagebox #消息窗口库
3 import tkinter.font as tkfont #字体库
4 from tkinter import ttk #下拉菜单库
5 from tkinter import scrolledtext #滚动文本库
6 import datetime #日期函数库
7 #import sqlite3 #连接sqlite3的库
8 import psycopg2 #连接opengauss的库
9
10 #连接数据库(OpenGauss)
11 def connect_database():
12     con = psycopg2.connect(
13         host='123.249.97.41',
14         port=26000,
15         user='xjtudba',
16         password='xjtu@123',
17         database='dbapp')
18     cur = con.cursor();
19     return con, cur
20 #连接数据库(Sqlite3)
21 #def connect_database():
22     con = sqlite3.connect("addressbook.db")
23     cur=con.cursor()
24     return con,cur
25
26 #显示查询结果
27 def show_result(result):
28     result_window = tk.Toplevel(root)
29     result_window.geometry('800x400+250+150')
30     text = scrolledtext.ScrolledText(result_window, width=800, height=400)
31     for row in result:
32         text.insert(tk.END, ' '.join(map(str, row)) + '\n') #将每一行元素转换为字符串，用空格连接，最后加上换行符
33     text.pack()

```

```

34     text.config(state=tk.DISABLED) #使用config方法将text部件设置为只读
35     return
36
37 #添加学生信息
38 def add_student_window():
39     add_student_window = tk.Toplevel(root)
40     add_student_window.geometry('300x500+550+150')
41     add_student_window.title('添加学生信息')
42     def add_student():
43         #获取用户输入的学生信息
44         sid = sid_entry.get()
45         sname = sname_entry.get()
46         gender = gender_cmb.get()
47         age = age_entry.get()
48         sclass = sclass_entry.get()
49         school = school_entry.get()
50         birthday = birthday_entry.get()
51         province = province_entry.get()
52         #检查学生信息有效性
53         if not sid:
54             tk.messagebox.showerror('错误', '学号不能为空! ')
55             return
56         if not sname:
57             tk.messagebox.showerror('错误', '姓名不能为空! ')
58             return
59         try:
60             age = int(age)
61         except ValueError:
62             tk.messagebox.showerror('错误', '年龄必须是整数! ')
63             return
64         try:
65             birthday = datetime.datetime.strptime(birthday, '%Y-%m-%d').date()
66         except ValueError:
67             tk.messagebox.showerror('错误', '生日必须是"YYYY-MM-DD"格式的日期! ')
68             return
69         #将学生信息保存到数据库
70         con, cur = connect_database()
71         cur.execute('insert into student values(%s, %s, %s, %s, %s, %s, %s, %s)', (sid, sname,
72             gender, age, sclass, school, birthday, province))
73         con.commit()
74         cur.close()
75         con.close()
76         tk.messagebox.showinfo('提示', '数据插入成功! ')
77         # 清空表单
78         sid_entry.delete(0, 'end')
79         sname_entry.delete(0, 'end')
80         gender_cmb.delete(0, 'end')
81         age_entry.delete(0, 'end')
82         sclass_entry.delete(0, 'end')
83         school_entry.delete(0, 'end')
84         birthday_entry.delete(0, 'end')
85         province_entry.delete(0, 'end')
86
87         #学号输入框
88         sid_label = tk.Label(add_student_window, text='学号:', font=font3)
89         sid_label.pack()
90         sid_entry = tk.Entry(add_student_window)
91         sid_entry.pack()
92
93         #姓名输入框
94         sname_label = tk.Label(add_student_window, text='姓名:', font=font3)
95         sname_label.pack()
96         sname_entry = tk.Entry(add_student_window)
97         sname_entry.pack()

```

```

95     #性别下拉菜单
96     gender_label = tk.Label(add_student_window, text='性别:', font=font3)
97     gender_label.pack()
98     gender_cmb = ttk.Combobox(add_student_window, width=15, font=font3)
99     gender_cmb.pack()
100    gender_cmb['value'] = ('男', '女')
101    #年龄输入框
102    age_label = tk.Label(add_student_window, text='年龄:', font=font3)
103    age_label.pack()
104    age_entry = tk.Entry(add_student_window)
105    age_entry.pack()
106    #班级输入框
107    sclass_label = tk.Label(add_student_window, text='班级:', font=font3)
108    sclass_label.pack()
109    sclass_entry = tk.Entry(add_student_window)
110    sclass_entry.pack()
111    #学院输入框
112    school_label = tk.Label(add_student_window, text='学院:', font=font3)
113    school_label.pack()
114    school_entry = tk.Entry(add_student_window)
115    school_entry.pack()
116    #生日输入框
117    birthday_label = tk.Label(add_student_window, text='生日:', font=font3)
118    birthday_label.pack()
119    birthday_entry = tk.Entry(add_student_window)
120    birthday_entry.pack()
121    #籍贯输入框
122    province_label = tk.Label(add_student_window, text='籍贯:', font=font3)
123    province_label.pack()
124    province_entry = tk.Entry(add_student_window)
125    province_entry.pack()
126    #确定和取消按钮
127    confirm_button = tk.Button(add_student_window, text='确定', command=add_student, font=font3,
128                               fg='red', cursor='hand2')
128    confirm_button.pack()
129    cancel_button = tk.Button(add_student_window, text='取消', command=add_student_window.destroy,
130                              font=font3, fg='red', cursor='hand2')
130    cancel_button.pack()
131
132    #查询学生信息
133    def select_student_window():
134        select_student_window = tk.Toplevel(root)
135        select_student_window.geometry('300x500+550+150')
136        select_student_window.title('查询学生信息')
137        def select_student():
138            #获取用户输入的学生信息
139            sid = sid_entry.get()
140            sname = sname_entry.get()
141            gender = gender_cmb.get()
142            age = age_entry.get()
143            sclass = sclass_entry.get()
144            school = school_entry.get()
145            birthday = birthday_entry.get()
146            province = province_entry.get()
147            #检查学生信息有效性
148            if age:
149                try:
150                    age = int(age)
151                except ValueError:
152                    tk.messagebox.showerror('错误', '年龄必须是整数! ')
153                return
154            if birthday:

```



```

155         try:
156             birthday = datetime.datetime.strptime(birthday, '%Y-%m-%d').date()
157         except ValueError:
158             tk.messagebox.showerror('错误', '生日必须是"YYYY-MM-DD"格式的日期! ')
159         return
160     #根据输入的学生信息进行查询
161     con, cur = connect_database()
162     sql = 'select * from student where 1'
163     params = []
164     if sid:
165         sql += ' and sid like %s'
166         params.append('%' + sid + '%')
167     if sname:
168         sql += ' and sname like %s'
169         params.append('%' + sname + '%')
170     if gender:
171         sql += ' and gender like %s'
172         params.append(gender)
173     if age:
174         sql += ' and age like %s'
175         params.append('%' + str(age) + '%')
176     if sclass:
177         sql += ' and sclass like %s'
178         params.append('%' + sclass + '%')
179     if school:
180         sql += ' and school like %s'
181         params.append('%' + school + '%')
182     if birthday:
183         sql += ' and birthday like %s'
184         params.append('%' + str(birthday) + '%')
185     if province:
186         sql += ' and province like %s'
187         params.append('%' + province + '%')
188     cur.execute(sql, params)
189     result = cur.fetchall()
190     show_result(result)
191     con.commit()
192     cur.close()
193     con.close()
194     # 清空表单
195     sid_entry.delete(0, 'end')
196     sname_entry.delete(0, 'end')
197     gender_cmb.delete(0, 'end')
198     age_entry.delete(0, 'end')
199     sclass_entry.delete(0, 'end')
200     school_entry.delete(0, 'end')
201     birthday_entry.delete(0, 'end')
202     province_entry.delete(0, 'end')
203     #学号输入框
204     sid_label = tk.Label(select_student_window, text='学号:', font=font3)
205     sid_label.pack()
206     sid_entry = tk.Entry(select_student_window)
207     sid_entry.pack()
208     #姓名输入框
209     sname_label = tk.Label(select_student_window, text='姓名:', font=font3)
210     sname_label.pack()
211     sname_entry = tk.Entry(select_student_window)
212     sname_entry.pack()
213     #性别下拉菜单
214     gender_label = tk.Label(select_student_window, text='性别:', font=font3)
215     gender_label.pack()
216     gender_cmb = ttk.Combobox(select_student_window, width=15, font=font3)

```

```

217 gender_cmb.pack()
218 gender_cmb['value'] = ('男','女')
219 #年龄输入框
220 age_label = tk.Label(select_student_window, text='年龄:', font=font3)
221 age_label.pack()
222 age_entry = tk.Entry(select_student_window)
223 age_entry.pack()
224 #班级输入框
225 sclass_label = tk.Label(select_student_window, text='班级:', font=font3)
226 sclass_label.pack()
227 sclass_entry = tk.Entry(select_student_window)
228 sclass_entry.pack()
229 #学院输入框
230 school_label = tk.Label(select_student_window, text='学院:', font=font3)
231 school_label.pack()
232 school_entry = tk.Entry(select_student_window)
233 school_entry.pack()
234 #生日输入框
235 birthday_label = tk.Label(select_student_window, text='生日:', font=font3)
236 birthday_label.pack()
237 birthday_entry = tk.Entry(select_student_window)
238 birthday_entry.pack()
239 #籍贯输入框
240 province_label = tk.Label(select_student_window, text='籍贯:', font=font3)
241 province_label.pack()
242 province_entry = tk.Entry(select_student_window)
243 province_entry.pack()
244 #确定和取消按钮
245 confirm_button = tk.Button(select_student_window, text='确定', command=select_student, font=
    font3, fg='red', cursor='hand2')
246 confirm_button.pack()
247 cancel_button = tk.Button(select_student_window, text='取消', command=select_student_window.
    destroy, font=font3, fg='red', cursor='hand2')
248 cancel_button.pack()
249
250 #更新学生信息
251 def update_student_window():
252     update_student_window = tk.Toplevel(root)
253     update_student_window.geometry('500x300+450+150')
254     update_student_window.title('更新学生信息')
255     #标签和字段名的映射
256     field_map = {'学号':'sid', '姓名':'sname', '性别':'gender', '年龄':'age', '班级':'sclass', '学
        院':'school', '生日':'birthday', '籍贯':'province'}
257     def update_student():
258         #获取用户输入的学生信息
259         for field, old_entry, new_entry in entries:
260             old_value = old_entry.get()
261             new_value = new_entry.get()
262             field = field_map[field] #从映射中获取字段名
263             #根据输入的学生信息进行更新
264             con, cur = connect_database()
265             if old_value and new_value:
266                 cur.execute('update student set ' + f'{field}' + ' = %s where ' + f'{field}' + ' =
                    %s', (new_value, old_value))
267                 con.commit()
268             elif old_value or new_value:
269                 tk.messagebox.showerror('错误', '旧值和新值都必须填写! ')
270             return
271     cur.close()
272     con.close()
273     tk.messagebox.showinfo('成功', '更新成功! ')
274     #清空表单

```

```

275     for _, old_entry, new_entry in entries: #用_表示不关心第一个元素field
276         old_entry.delete(0, 'end')
277         new_entry.delete(0, 'end')
278
279 #标签和输入框
280 entries = []
281 for i, field_label in enumerate(field_map.keys()):
282     label = tk.Label(update_student_window, width=7)
283     label.grid(row=i, column=0)
284     old_label = tk.Label(update_student_window, text=f'原{field_label}:', font=font3)
285     old_label.grid(row=i, column=1)
286     new_label = tk.Label(update_student_window, text=f'新{field_label}:', font=font3)
287     new_label.grid(row=i, column=3)
288     old_entry = tk.Entry(update_student_window)
289     old_entry.grid(row=i, column=2)
290     new_entry = tk.Entry(update_student_window)
291     new_entry.grid(row=i, column=4)
292     entries.append((field_label, old_entry, new_entry))
293 #确定和取消按钮
294 confirm_button = tk.Button(update_student_window, text='确定', command=update_student, font=
    font3, fg='red', cursor='hand2')
295 confirm_button.grid(row=8, column=2)
296 cancel_button = tk.Button(update_student_window, text='取消', command=update_student_window.
    destroy, font=font3, fg='red', cursor='hand2')
297 cancel_button.grid(row=8, column=4)
298
299 #删除学生信息
300 def delete_student_window():
301     delete_student_window = tk.Toplevel(root)
302     delete_student_window.geometry('300x500+550+150')
303     delete_student_window.title('删除学生信息')
304     def delete_student():
305         #获取用户输入的学生信息
306         sid = sid_entry.get()
307         sname = sname_entry.get()
308         gender = gender_cmb.get()
309         age = age_entry.get()
310         sclass = sclass_entry.get()
311         school = school_entry.get()
312         birthday = birthday_entry.get()
313         province = province_entry.get()
314         #检查学生信息有效性
315         if age:
316             try:
317                 age = int(age)
318             except ValueError:
319                 tk.messagebox.showerror('错误', '年龄必须是整数!')
320             return
321         if birthday:
322             try:
323                 birthday = datetime.datetime.strptime(birthday, '%Y-%m-%d').date()
324             except ValueError:
325                 tk.messagebox.showerror('错误', '生日必须是"YYYY-MM-DD"格式的日期!')
326             return
327         #根据输入的学生信息进行查询
328         confirm = tk.messagebox.askyesno('确认', '你确定要删除此学生信息吗?')
329         if confirm:
330             con, cur = connect_database()
331             sql = ' from student where 1'
332             params = []
333             if sid:
334                 sql += ' and sid like %s'

```

```

335         params.append('%' + sid + '%')
336     if sname:
337         sql += ' and sname like %s'
338         params.append('%' + sname + '%')
339     if gender:
340         sql += ' and gender like %s'
341         params.append('%' + gender + '%')
342     if age:
343         sql += ' and age=%s'
344         params.append('%' + str(age) + '%')
345     if sclass:
346         sql += ' and sclass like %s'
347         params.append('%' + gender + '%')
348     if school:
349         sql += ' and school like %s'
350         params.append('%' + school + '%')
351     if birthday:
352         sql += ' and birthday like %s'
353         params.append('%' + str(birthday) + '%')
354     if province:
355         sql += ' and province like %s'
356         params.append('%' + province + '%')
357     #检查要删除的学生信息是否存在
358     cur.execute('select *'+sql, params)
359     result = cur.fetchall()
360     if result is None:
361         tk.messagebox.showerror('错误', '您要删除的学生信息不存在! ')
362     else:
363         cur.execute('delete'+sql, params)
364         con.commit()
365         cur.close()
366         con.close()
367         tk.messagebox.showinfo('成功', '删除成功! ')
368         # 清空表单
369         sid_entry.delete(0, 'end')
370         sname_entry.delete(0, 'end')
371         gender_cmb.delete(0, 'end')
372         age_entry.delete(0, 'end')
373         sclass_entry.delete(0, 'end')
374         school_entry.delete(0, 'end')
375         birthday_entry.delete(0, 'end')
376         province_entry.delete(0, 'end')
377     #学号输入框
378     sid_label = tk.Label(delete_student_window, text='学号:', font=font3)
379     sid_label.pack()
380     sid_entry = tk.Entry(delete_student_window)
381     sid_entry.pack()
382     #姓名输入框
383     sname_label = tk.Label(delete_student_window, text='姓名:', font=font3)
384     sname_label.pack()
385     sname_entry = tk.Entry(delete_student_window)
386     sname_entry.pack()
387     #性别下拉菜单
388     gender_label = tk.Label(delete_student_window, text='性别:', font=font3)
389     gender_label.pack()
390     gender_cmb = ttk.Combobox(delete_student_window, width=15, font=font3)
391     gender_cmb.pack()
392     gender_cmb['value'] = ('男', '女')
393     #年龄输入框
394     age_label = tk.Label(delete_student_window, text='年龄:', font=font3)
395     age_label.pack()
396     age_entry = tk.Entry(delete_student_window)

```

```

397     age_entry.pack()
398     #班级输入框
399     sclass_label = tk.Label(delete_student_window, text='班级:', font=font3)
400     sclass_label.pack()
401     sclass_entry = tk.Entry(delete_student_window)
402     sclass_entry.pack()
403     #学院输入框
404     school_label = tk.Label(delete_student_window, text='学院:', font=font3)
405     school_label.pack()
406     school_entry = tk.Entry(delete_student_window)
407     school_entry.pack()
408     #生日输入框
409     birthday_label = tk.Label(delete_student_window, text='生日:', font=font3)
410     birthday_label.pack()
411     birthday_entry = tk.Entry(delete_student_window)
412     birthday_entry.pack()
413     #籍贯输入框
414     province_label = tk.Label(delete_student_window, text='籍贯:', font=font3)
415     province_label.pack()
416     province_entry = tk.Entry(delete_student_window)
417     province_entry.pack()
418     #确定和取消按钮
419     confirm_button = tk.Button(delete_student_window, text='确定', command=delete_student, font=
        font3, fg='red', cursor='hand2')
420     confirm_button.pack()
421     cancel_button = tk.Button(delete_student_window, text='取消', command=delete_student_window.
        destroy, font=font3, fg='red', cursor='hand2')
422     cancel_button.pack()
423
424     #添加课程信息
425     def add_course_window():
426         add_course_window = tk.Toplevel(root)
427         add_course_window.geometry('300x500+550+150')
428         add_course_window.title('添加课程信息')
429         def add_course():
430             #获取用户输入的课程信息
431             cid = cid_entry.get()
432             cname = cname_entry.get()
433             teacher = teacher_entry.get()
434             credit = credit_entry.get()
435             cpid = cpid_entry.get()
436             #检查课程信息有效性
437             if not cid:
438                 tk.messagebox.showerror('错误', '课程号不能为空! ')
439                 return
440             if not cname:
441                 tk.messagebox.showerror('错误', '课程名不能为空! ')
442                 return
443             #将课程信息保存到数据库
444             con, cur = connect_database()
445             cur.execute('insert into course values(%s, %s, %s, %s, %s)', (cid, cname, teacher, credit,
                cpid))
446             con.commit()
447             cur.close()
448             con.close()
449             tk.messagebox.showinfo('提示', '数据插入成功! ')
450             # 清空表单
451             cid_entry.delete(0, 'end')
452             cname_entry.delete(0, 'end')
453             teacher_entry.delete(0, 'end')
454             credit_entry.delete(0, 'end')
455             cpid_entry.delete(0, 'end')

```

```

456 #课程号输入框
457 cid_label = tk.Label(add_course_window, text='课程号:', font=font3)
458 cid_label.pack()
459 cid_entry = tk.Entry(add_course_window)
460 cid_entry.pack()
461 #课程名输入框
462 cname_label = tk.Label(add_course_window, text='课程名:', font=font3)
463 cname_label.pack()
464 cname_entry = tk.Entry(add_course_window)
465 cname_entry.pack()
466 #教师输入框
467 teacher_label = tk.Label(add_course_window, text='教师:', font=font3)
468 teacher_label.pack()
469 teacher_entry = tk.Entry(add_course_window)
470 teacher_entry.pack()
471 #学分输入框
472 credit_label = tk.Label(add_course_window, text='学分:', font=font3)
473 credit_label.pack()
474 credit_entry = tk.Entry(add_course_window)
475 credit_entry.pack()
476 #先修课程输入框
477 cpid_label = tk.Label(add_course_window, text='先修课程:', font=font3)
478 cpid_label.pack()
479 cpid_entry = tk.Entry(add_course_window)
480 cpid_entry.pack()
481 #确定和取消按钮
482 confirm_button = tk.Button(add_course_window, text='确定', command=add_course, font=font3, fg=
    'red', cursor='hand2')
483 confirm_button.pack()
484 cancel_button = tk.Button(add_course_window, text='取消', command=add_course_window.destroy,
    font=font3, fg='red', cursor='hand2')
485 cancel_button.pack()
486
487 #查询课程信息
488 def select_course_window():
489     select_course_window = tk.Toplevel(root)
490     select_course_window.geometry('300x500+550+150')
491     select_course_window.title('查询课程信息')
492     def select_course():
493         #获取用户输入的课程信息
494         cid = cid_entry.get()
495         cname = cname_entry.get()
496         teacher = teacher_entry.get()
497         credit = credit_entry.get()
498         cpid = cpid_entry.get()
499         #根据输入的学生信息进行查询
500         con, cur = connect_database()
501         sql = 'select * from course where 1'
502         params = []
503         if cid:
504             sql += ' and cid like %s'
505             params.append('%' + cid + '%')
506         if cname:
507             sql += ' and cname like %s'
508             params.append('%' + cname + '%')
509         if teacher:
510             sql += ' and teacher like %s'
511             params.append('%' + teacher + '%')
512         if credit:
513             sql += ' and credit like %s'
514             params.append('%' + credit + '%')
515         if cpid:

```

```

516         sql += ' and cpid like %s'
517         params.append('%' + cpid + '%')
518     cur.execute(sql, params)
519     result = cur.fetchall()
520     show_result(result)
521     con.commit()
522     cur.close()
523     con.close()
524     # 清空表单
525     cid_entry.delete(0, 'end')
526     cname_entry.delete(0, 'end')
527     teacher_entry.delete(0, 'end')
528     credit_entry.delete(0, 'end')
529     cpid_entry.delete(0, 'end')
530     return result
531 #课程号输入框
532 cid_label = tk.Label(select_course_window, text='课程号:', font=font3)
533 cid_label.pack()
534 cid_entry = tk.Entry(select_course_window)
535 cid_entry.pack()
536 #课程名输入框
537 cname_label = tk.Label(select_course_window, text='课程名:', font=font3)
538 cname_label.pack()
539 cname_entry = tk.Entry(select_course_window)
540 cname_entry.pack()
541 #教师输入框
542 teacher_label = tk.Label(select_course_window, text='教师:', font=font3)
543 teacher_label.pack()
544 teacher_entry = tk.Entry(select_course_window)
545 teacher_entry.pack()
546 #学分输入框
547 credit_label = tk.Label(select_course_window, text='学分:', font=font3)
548 credit_label.pack()
549 credit_entry = tk.Entry(select_course_window)
550 credit_entry.pack()
551 #先修课程输入框
552 cpid_label = tk.Label(select_course_window, text='先修课程:', font=font3)
553 cpid_label.pack()
554 cpid_entry = tk.Entry(select_course_window)
555 cpid_entry.pack()
556 #确定和取消按钮
557 confirm_button = tk.Button(select_course_window, text='确定', command=select_course, font=
558     font3, fg='red', cursor='hand2')
559 confirm_button.pack()
560 cancel_button = tk.Button(select_course_window, text='取消', command=select_course_window.
561     destroy, font=font3, fg='red', cursor='hand2')
562 cancel_button.pack()
563 #更新课程信息
564 def update_course_window():
565     update_course_window = tk.Toplevel(root)
566     update_course_window.geometry('500x300+450+150')
567     update_course_window.title('更新课程信息')
568     #创建标签和字段名的映射
569     field_map = {'课程号':'cid', '课程名':'cname', '教师':'teacher', '学分':'credit', '先修课程':'
570         cpid'}
571     def update_course():
572         #获取用户输入的课程信息
573         for field, old_entry, new_entry in entries:
574             old_value = old_entry.get()
575             new_value = new_entry.get()
576             field = field_map[field] #从映射中获取字段名

```

```

575     #根据输入的学生信息进行更新
576     con, cur = connect_database()
577     if old_value and new_value:
578         cur.execute('update course set ' + f'{field}' + ' = %s where ' + f'{field}' + ' =
                    %s', (new_value, old_value))
579         con.commit()
580     elif old_value or new_value:
581         tk.messagebox.showerror('错误', '旧值和新值都必须填写! ')
582         return
583     cur.close()
584     con.close()
585     tk.messagebox.showinfo('成功', '更新成功! ')
586     #清空表单
587     for _, old_entry, new_entry in entries: #用_表示不关心第一个元素field
588         old_entry.delete(0, 'end')
589         new_entry.delete(0, 'end')
590
591 #标签和输入框
592 entries = []
593 for i, field_label in enumerate(field_map.keys()):
594     label = tk.Label(update_course_window, width=3)
595     label.grid(row=i, column=0)
596     old_label = tk.Label(update_course_window, text=f'原{field_label}:', font=font3)
597     old_label.grid(row=i, column=1)
598     new_label = tk.Label(update_course_window, text=f'新{field_label}:', font=font3)
599     new_label.grid(row=i, column=3)
600     old_entry = tk.Entry(update_course_window)
601     old_entry.grid(row=i, column=2)
602     new_entry = tk.Entry(update_course_window)
603     new_entry.grid(row=i, column=4)
604     entries.append((field_label, old_entry, new_entry))
605 #确定和取消按钮
606 confirm_button = tk.Button(update_course_window, text='确定', command=update_course, font=
        font3, fg='red', cursor='hand2')
607 confirm_button.grid(row=5, column=2)
608 cancel_button = tk.Button(update_course_window, text='取消', command=update_course_window.
        destroy, font=font3, fg='red', cursor='hand2')
609 cancel_button.grid(row=5, column=4)
610
611 #删除课程信息
612 def delete_course_window():
613     delete_course_window = tk.Toplevel(root)
614     delete_course_window.geometry('300x500+550+150')
615     delete_course_window.title('删除课程信息')
616     def delete_course():
617         #获取用户输入的课程信息
618         cid = cid_entry.get()
619         cname = cname_entry.get()
620         teacher = teacher_entry.get()
621         credit = credit_entry.get()
622         cpid = cpid_entry.get()
623         #根据输入的课程信息进行删除
624         confirm = tk.messagebox.askyesno('确认', '您确定要删除此课程信息吗? ')
625         if confirm:
626             con, cur = connect_database()
627             sql = ' from course where 1'
628             params = []
629             if cid:
630                 sql += ' and cid like %s'
631                 params.append('%' + cid + '%')
632             if cname:
633                 sql += ' and cname like %s'

```



```

634         params.append('%' + cname + '%')
635     if teacher:
636         sql += ' and teacher like %s'
637         params.append('%' + teacher + '%')
638     if credit:
639         sql += ' and credit like %s'
640         params.append('%' + credit + '%')
641     if cpid:
642         sql += ' and cpid like %s'
643         params.append('%' + cpid + '%')
644     #检查要删除的课程信息是否存在
645     cur.execute('select *'+sql, params)
646     result = cur.fetchall()
647     if result is None:
648         tk.messagebox.showerror('错误', '您要删除的课程信息不存在! ')
649     else:
650         cur.execute('delete'+sql, params)
651         con.commit()
652         cur.close()
653         con.close()
654         tk.messagebox.showinfo('成功', '删除成功! ')
655         # 清空表单
656         cid_entry.delete(0, 'end')
657         cname_entry.delete(0, 'end')
658         teacher_entry.delete(0, 'end')
659         credit_entry.delete(0, 'end')
660         cpid_entry.delete(0, 'end')
661     #课程号输入框
662     cid_label = tk.Label(delete_course_window, text='课程号:', font=font3)
663     cid_label.pack()
664     cid_entry = tk.Entry(delete_course_window)
665     cid_entry.pack()
666     #课程名输入框
667     cname_label = tk.Label(delete_course_window, text='课程名:', font=font3)
668     cname_label.pack()
669     cname_entry = tk.Entry(delete_course_window)
670     cname_entry.pack()
671     #教师输入框
672     teacher_label = tk.Label(delete_course_window, text='教师:', font=font3)
673     teacher_label.pack()
674     teacher_entry = tk.Entry(delete_course_window)
675     teacher_entry.pack()
676     #学分输入框
677     credit_label = tk.Label(delete_course_window, text='学分:', font=font3)
678     credit_label.pack()
679     credit_entry = tk.Entry(delete_course_window)
680     credit_entry.pack()
681     #先修课程输入框
682     cpid_label = tk.Label(delete_course_window, text='先修课程:', font=font3)
683     cpid_label.pack()
684     cpid_entry = tk.Entry(delete_course_window)
685     cpid_entry.pack()
686     #确定和取消按钮
687     confirm_button = tk.Button(delete_course_window, text='确定', command=delete_course, font=
        font3, fg='red', cursor='hand2')
688     confirm_button.pack()
689     cancel_button = tk.Button(delete_course_window, text='取消', command=delete_course_window.
        destroy, font=font3, fg='red', cursor='hand2')
690     cancel_button.pack()
691
692 #添加选课信息
693 def add_sc_window():

```

```

694 add_sc_window = tk.Toplevel(root)
695 add_sc_window.geometry('300x500+550+150')
696 add_sc_window.title('添加选课信息')
697 def add_sc():
698     #获取用户输入的选课信息
699     sid = sid_entry.get()
700     cid = cid_entry.get()
701     year = year_entry.get()
702     term = term_cmb.get()
703     #检查选课信息有效性
704     if not sid:
705         tk.messagebox.showerror('错误', '学号不能为空! ')
706         return
707     if not cid:
708         tk.messagebox.showerror('错误', '课程号不能为空! ')
709         return
710     if not year:
711         tk.messagebox.showerror('错误', '学年不能为空! ')
712         return
713     if not term:
714         tk.messagebox.showerror('错误', '学期不能为空! ')
715         return
716     #将选课信息保存到数据库
717     con, cur = connect_database()
718     cur.execute('insert into sc(sid, cid, year, term) values(%s, %s, %s, %s)', (sid, cid, year
719         , term))
719     con.commit()
720     cur.close()
721     con.close()
722     tk.messagebox.showinfo('提示', '数据插入成功! ')
723     # 清空表单
724     sid_entry.delete(0, 'end')
725     cid_entry.delete(0, 'end')
726     year_entry.delete(0, 'end')
727     term_cmb.delete(0, 'end')
728     #学号输入框
729     sid_label = tk.Label(add_sc_window, text='学号:', font=font3)
730     sid_label.pack()
731     sid_entry = tk.Entry(add_sc_window)
732     sid_entry.pack()
733     #课程号输入框
734     cid_label = tk.Label(add_sc_window, text='课程号:', font=font3)
735     cid_label.pack()
736     cid_entry = tk.Entry(add_sc_window)
737     cid_entry.pack()
738     #学年输入框
739     year_label = tk.Label(add_sc_window, text='学年:', font=font3)
740     year_label.pack()
741     year_entry = tk.Entry(add_sc_window)
742     year_entry.pack()
743     #学期下拉菜单
744     term_label = tk.Label(add_sc_window, text='学期:', width=17, font=font3)
745     term_label.pack()
746     term_cmb = ttk.Combobox(add_sc_window, width=15, font=font3)
747     term_cmb.pack()
748     term_cmb['value'] = ('第一学期', '第二学期', '小学期')
749     #确定和取消按钮
750     confirm_button = tk.Button(add_sc_window, text='确定', command=add_sc, font=font3, fg='red',
751         cursor='hand2')
751     confirm_button.pack()
752     cancel_button = tk.Button(add_sc_window, text='取消', command=add_sc_window.destroy, font=
753         font3, fg='red', cursor='hand2')

```

```

753     cancel_button.pack()
754
755 #查询选课信息
756 def select_sc_window():
757     select_sc_window = tk.Toplevel(root)
758     select_sc_window.geometry('300x500+550+150')
759     select_sc_window.title('添加选课信息')
760     def select_sc():
761         #获取用户输入的选课信息
762         sid = sid_entry.get()
763         cid = cid_entry.get()
764         year = year_entry.get()
765         term = term_cmb.get()
766         #根据输入的选课信息进行查询
767         con, cur = connect_database()
768         sql = 'select sid, cid, year, term from sc where 1'
769         params = []
770         if sid:
771             sql += ' and sid like %s'
772             params.append('%' + sid + '%')
773         if cid:
774             sql += ' and cname like %s'
775             params.append('%' + cid + '%')
776         if year:
777             sql += ' and year like %s'
778             params.append('%' + year + '%')
779         if term:
780             sql += ' and term like %s'
781             params.append('%' + term + '%')
782         cur.execute(sql, params)
783         result = cur.fetchall()
784         show_result(result)
785         con.commit()
786         cur.close()
787         con.close()
788         # 清空表单
789         sid_entry.delete(0, 'end')
790         cid_entry.delete(0, 'end')
791         year_entry.delete(0, 'end')
792         term_cmb.delete(0, 'end')
793
794 #学号输入框
795 sid_label = tk.Label(select_sc_window, text='学号:', font=font3)
796 sid_label.pack()
797 sid_entry = tk.Entry(select_sc_window)
798 sid_entry.pack()
799 #课程号输入框
800 cid_label = tk.Label(select_sc_window, text='课程号:', font=font3)
801 cid_label.pack()
802 cid_entry = tk.Entry(select_sc_window)
803 cid_entry.pack()
804 #学年输入框
805 year_label = tk.Label(select_sc_window, text='学年:', font=font3)
806 year_label.pack()
807 year_entry = tk.Entry(select_sc_window)
808 year_entry.pack()
809 #学期下拉菜单
810 term_label = tk.Label(select_sc_window, text='学期:', width=17, font=font3)
811 term_label.pack()
812 term_cmb = ttk.Combobox(select_sc_window, width=15, font=font3)
813 term_cmb.pack()
814 term_cmb['value'] = ('第一学期', '第二学期', '小学期')
815 #确定和取消按钮

```

```

815     confirm_button = tk.Button(select_sc_window, text='确定', command=select_sc, font=font3, fg='
        red', cursor='hand2')
816     confirm_button.pack()
817     cancel_button = tk.Button(select_sc_window, text='取消', command=select_sc_window.destroy,
        font=font3, fg='red', cursor='hand2')
818     cancel_button.pack()
819
820 #更新选课信息
821 def update_sc_window():
822     update_sc_window = tk.Toplevel(root)
823     update_sc_window.geometry('500x300+450+150')
824     update_sc_window.title('添加选课信息')
825     #标签和字段名的映射
826     field_map = {'学号':'sid', '课程号':'cid', '学年':'year', '学期':'term'}
827     def update_sc():
828         #获取用户输入的课程信息
829         for field, old_entry, new_entry in entries:
830             old_value = old_entry.get()
831             new_value = new_entry.get()
832             field = field_map[field] #从映射中获取字段名
833             #根据输入的学生信息进行更新
834             con, cur = connect_database()
835             if old_value and new_value:
836                 cur.execute('update sc set ' + f'{field}' + ' = %s where ' + f'{field}' + ' = %s',
                        (new_value, old_value))
837                 con.commit()
838             elif old_value or new_value:
839                 tk.messagebox.showerror('错误', '旧值和新值都必须填写! ')
840             return
841         cur.close()
842         con.close()
843         tk.messagebox.showinfo('成功', '更新成功! ')
844         #清空表单
845         for _, old_entry, new_entry in entries: #用_表示不关心第一个元素field
846             old_entry.delete(0, 'end')
847             new_entry.delete(0, 'end')
848
849 #标签和输入框
850 entries = []
851 for i, field_label in enumerate(field_map.keys()):
852     label = tk.Label(update_sc_window, width=4)
853     label.grid(row=i, column=0)
854     old_label = tk.Label(update_sc_window, text=f'原{field_label}:', font=font3)
855     old_label.grid(row=i, column=1)
856     new_label = tk.Label(update_sc_window, text=f'新{field_label}:', font=font3)
857     new_label.grid(row=i, column=3)
858     old_entry = tk.Entry(update_sc_window)
859     old_entry.grid(row=i, column=2)
860     new_entry = tk.Entry(update_sc_window)
861     new_entry.grid(row=i, column=4)
862     entries.append((field_label, old_entry, new_entry))
863 #确定和取消按钮
864 confirm_button = tk.Button(update_sc_window, text='确定', command=update_sc, font=font3, fg='
        red', cursor='hand2')
865 confirm_button.grid(row=4, column=2)
866 cancel_button = tk.Button(update_sc_window, text='取消', command=update_sc_window.destroy,
        font=font3, fg='red', cursor='hand2')
867 cancel_button.grid(row=4, column=4)
868
869 #删除选课信息
870 def delete_sc_window():
871     delete_sc_window = tk.Toplevel(root)

```

```

872 delete_sc_window.geometry('300x500+550+150')
873 delete_sc_window.title('删除选课信息')
874 def delete_sc():
875     #获取用户输入的选课信息
876     sid = sid_entry.get()
877     cid = cid_entry.get()
878     year = year_entry.get()
879     term = term_cmb.get()
880     #根据输入的选课信息进行删除
881     confirm = tk.messagebox.askyesno('确认', '你确定要删除此选课信息吗? ')
882     if confirm:
883         con, cur = connect_database()
884         sql = ' from sc where 1'
885         params = []
886         if sid:
887             sql += ' and sid like %s'
888             params.append('%' + sid + '%')
889         if cid:
890             sql += ' and cid like %s'
891             params.append('%' + cid + '%')
892         if year:
893             sql += ' and year like %s'
894             params.append('%' + year + '%')
895         if term:
896             sql += ' and term like %s'
897             params.append('%' + term + '%')
898         #检查要删除的课程信息是否存在
899         cur.execute('select *'+sql, params)
900         result = cur.fetchall()
901         if result is None:
902             tk.messagebox.showerror('错误', '您要删除的课程信息不存在! ')
903         else:
904             cur.execute('delete'+sql, params)
905             con.commit()
906             cur.close()
907             con.close()
908             tk.messagebox.showinfo('成功', '删除成功! ')
909             #清空表单
910             sid_entry.delete(0, 'end')
911             cid_entry.delete(0, 'end')
912             year_entry.delete(0, 'end')
913             term_cmb.delete(0, 'end')
914     #学号输入框
915     sid_label = tk.Label(delete_sc_window, text='学号:', font=font3)
916     sid_label.pack()
917     sid_entry = tk.Entry(delete_sc_window)
918     sid_entry.pack()
919     #课程号输入框
920     cid_label = tk.Label(delete_sc_window, text='课程号:', font=font3)
921     cid_label.pack()
922     cid_entry = tk.Entry(delete_sc_window)
923     cid_entry.pack()
924     #学年输入框
925     year_label = tk.Label(delete_sc_window, text='学年:', font=font3)
926     year_label.pack()
927     year_entry = tk.Entry(delete_sc_window)
928     year_entry.pack()
929     #学期输入框
930     term_label = tk.Label(delete_sc_window, text='学期:', font=font3)
931     term_label.pack()
932     term_cmb = ttk.Combobox(delete_sc_window, width=15, font=font3)
933     term_cmb.pack()

```

```

934 term_cmb['value'] = ('第一学期', '第二学期', '小学期')
935 #确定和取消按钮
936 confirm_button = tk.Button(delete_sc_window, text='确定', command=delete_sc, font=font3, fg='
    red', cursor='hand2')
937 confirm_button.pack()
938 cancel_button = tk.Button(delete_sc_window, text='取消', command=delete_sc_window.destroy,
    font=font3, fg='red', cursor='hand2')
939 cancel_button.pack()
940
941 #添加成绩信息
942 def add_grade_window():
943     add_grade_window = tk.Toplevel(root)
944     add_grade_window.geometry('300x500+550+150')
945     add_grade_window.title('添加成绩信息')
946     def add_grade():
947         #获取用户输入的选课信息
948         sid = sid_entry.get()
949         cid = cid_entry.get()
950         year = year_entry.get()
951         term = term_cmb.get()
952         grade = grade_entry.get()
953         #检查成绩信息有效性
954         if not sid:
955             tk.messagebox.showerror('错误', '学号不能为空! ')
956             return
957         if not cid:
958             tk.messagebox.showerror('错误', '课程号不能为空! ')
959             return
960         try:
961             grade = float(grade)
962         except ValueError:
963             tk.messagebox.showerror('错误', '成绩必须是数值! ')
964             return
965         #将选课信息保存到数据库
966         con, cur = connect_database()
967         cur.execute('update sc set grade=%s where sid=%s and cid=%s and year=%s and term=%s', (
            grade, sid, cid, year, term))
968         con.commit()
969         cur.close()
970         con.close()
971         tk.messagebox.showinfo('提示', '数据插入成功! ')
972         #清空表单
973         sid_entry.delete(0, 'end')
974         cid_entry.delete(0, 'end')
975         year_entry.delete(0, 'end')
976         term_cmb.delete(0, 'end')
977     #学号输入框
978     sid_label = tk.Label(add_grade_window, text='学号:', font=font3)
979     sid_label.pack()
980     sid_entry = tk.Entry(add_grade_window)
981     sid_entry.pack()
982     #课程号输入框
983     cid_label = tk.Label(add_grade_window, text='课程号:', font=font3)
984     cid_label.pack()
985     cid_entry = tk.Entry(add_grade_window)
986     cid_entry.pack()
987     #学年输入框
988     year_label = tk.Label(add_grade_window, text='学年:', font=font3)
989     year_label.pack()
990     year_entry = tk.Entry(add_grade_window)
991     year_entry.pack()
992     #学期下拉菜单

```

```

993 term_label = tk.Label(add_grade_window, text='学期:', width=17, font=font3)
994 term_label.pack()
995 term_cmb = ttk.Combobox(add_grade_window, width=15, font=font3)
996 term_cmb.pack()
997 term_cmb['value'] = ('第一学期', '第二学期', '小学期')
998 #成绩输入框
999 grade_label = tk.Label(add_grade_window, text='成绩:', font=font3)
1000 grade_label.pack()
1001 grade_entry = tk.Entry(add_grade_window)
1002 grade_entry.pack()
1003 #确定和取消按钮
1004 confirm_button = tk.Button(add_grade_window, text='确定', command=add_grade, font=font3, fg='
    red', cursor='hand2')
1005 confirm_button.pack()
1006 cancel_button = tk.Button(add_grade_window, text='取消', command=add_grade_window.destroy,
    font=font3, fg='red', cursor='hand2')
1007 cancel_button.pack()
1008
1009
1010 #查询成绩信息
1011 def select_grade_window():
1012     select_grade_window = tk.Toplevel(root)
1013     select_grade_window.geometry('300x500+550+150')
1014     select_grade_window.title('查询成绩信息')
1015     def select_grade():
1016         #获取用户输入的选课信息
1017         sid = sid_entry.get()
1018         cid = cid_entry.get()
1019         year = year_entry.get()
1020         term = term_cmb.get()
1021         #根据输入的选课信息进行查询
1022         con, cur = connect_database()
1023         sql = 'select sid, cid, year, term, coalesce(grade,0) as grade from sc where 1' #coalesce
            实现成绩为NULL时返回结果为0
1024         params = []
1025         if sid:
1026             sql += ' and sid like %s'
1027             params.append('%' + sid + '%')
1028         if cid:
1029             sql += ' and cid like %s'
1030             params.append('%' + cid + '%')
1031         if year:
1032             sql += ' and year like %s'
1033             params.append('%' + year + '%')
1034         if term:
1035             sql += ' and term like %s'
1036             params.append('%' + term + '%')
1037         cur.execute(sql, params)
1038         result = cur.fetchall()
1039         show_result(result)
1040         con.commit()
1041         cur.close()
1042         con.close()
1043         #清空表单
1044         sid_entry.delete(0, 'end')
1045         cid_entry.delete(0, 'end')
1046         year_entry.delete(0, 'end')
1047         term_cmb.delete(0, 'end')
1048     #学号输入框
1049     sid_label = tk.Label(select_grade_window, text='学号:', font=font3)
1050     sid_label.pack()
1051     sid_entry = tk.Entry(select_grade_window)

```

```

1052     sid_entry.pack()
1053     #课程号输入框
1054     cid_label = tk.Label(select_grade_window, text='课程号:', font=font3)
1055     cid_label.pack()
1056     cid_entry = tk.Entry(select_grade_window)
1057     cid_entry.pack()
1058     #学年输入框
1059     year_label = tk.Label(select_grade_window, text='学年:', font=font3)
1060     year_label.pack()
1061     year_entry = tk.Entry(select_grade_window)
1062     year_entry.pack()
1063     #学期下拉菜单
1064     term_label = tk.Label(select_grade_window, text='学期:', width=17, font=font3)
1065     term_label.pack()
1066     term_cmb = ttk.Combobox(select_grade_window, width=15, font=font3)
1067     term_cmb.pack()
1068     term_cmb['value'] = ('第一学期', '第二学期', '小学期')
1069     #确定和取消按钮
1070     confirm_button = tk.Button(select_grade_window, text='确定', command=select_grade, font=font3,
1071                               fg='red', cursor='hand2')
1071     confirm_button.pack()
1072     cancel_button = tk.Button(select_grade_window, text='取消', command=select_grade_window.
1073                               destroy, font=font3, fg='red', cursor='hand2')
1073     cancel_button.pack()
1074
1075     #更新成绩信息
1076     def update_grade_window():
1077         update_grade_window = tk.Toplevel(root)
1078         update_grade_window.geometry('300x500+550+150')
1079         update_grade_window.title('更新成绩信息')
1080         def update_grade():
1081             #获取用户输入的选课信息
1082             sid = sid_entry.get()
1083             cid = cid_entry.get()
1084             year = year_entry.get()
1085             term = term_cmb.get()
1086             grade = grade_entry.get()
1087             #检查成绩信息有效性
1088             try:
1089                 grade = float(grade)
1090             except ValueError:
1091                 tk.messagebox.showerror('错误', '成绩必须是数值! ')
1092             return
1093             #根据输入的选课信息进行查询
1094             con, cur = connect_database()
1095             sql = 'update sc set grade=%s where 1'
1096             params = [grade]
1097             if sid:
1098                 sql += ' and sid like %s'
1099                 params.append('%' + sid + '%')
1100             if cid:
1101                 sql += ' and cid like %s'
1102                 params.append('%' + cid + '%')
1103             if year:
1104                 sql += ' and year like %s'
1105                 params.append('%' + year + '%')
1106             if term:
1107                 sql += ' and term like %s'
1108                 params.append('%' + term + '%')
1109             cur.execute(sql, params)
1110             con.commit()
1111             cur.close()

```



```

1112     con.close()
1113     tk.messagebox.showinfo('成功', '更新成功! ')
1114     #清空表单
1115     sid_entry.delete(0, 'end')
1116     cid_entry.delete(0, 'end')
1117     year_entry.delete(0, 'end')
1118     term_cmb.delete(0, 'end')
1119     grade_entry.delete(0, 'end')
1120     #学号输入框
1121     sid_label = tk.Label(update_grade_window, text='学号:', font=font3)
1122     sid_label.pack()
1123     sid_entry = tk.Entry(update_grade_window)
1124     sid_entry.pack()
1125     #课程号输入框
1126     cid_label = tk.Label(update_grade_window, text='课程号:', font=font3)
1127     cid_label.pack()
1128     cid_entry = tk.Entry(update_grade_window)
1129     cid_entry.pack()
1130     #学年输入框
1131     year_label = tk.Label(update_grade_window, text='学年:', font=font3)
1132     year_label.pack()
1133     year_entry = tk.Entry(update_grade_window)
1134     year_entry.pack()
1135     #学期下拉菜单
1136     term_label = tk.Label(update_grade_window, text='学期:', width=17, font=font3)
1137     term_label.pack()
1138     term_cmb = ttk.Combobox(update_grade_window, width=15, font=font3)
1139     term_cmb.pack()
1140     term_cmb['value'] = ('第一学期', '第二学期', '小学期')
1141     #成绩输入框
1142     grade_label = tk.Label(update_grade_window, text='成绩:', font=font3)
1143     grade_label.pack()
1144     grade_entry = tk.Entry(update_grade_window)
1145     grade_entry.pack()
1146     #确定和取消按钮
1147     confirm_button = tk.Button(update_grade_window, text='确定', command=update_grade, font=font3,
1148                               fg='red', cursor='hand2')
1149     confirm_button.pack()
1150     cancel_button = tk.Button(update_grade_window, text='取消', command=update_grade_window.
1151                               destroy, font=font3, fg='red', cursor='hand2')
1152     cancel_button.pack()
1153     #删除成绩信息
1154     def delete_grade_window():
1155         delete_grade_window = tk.Toplevel(root)
1156         delete_grade_window.geometry('300x500+550+150')
1157         delete_grade_window.title('删除成绩信息')
1158         def delete_grade():
1159             #获取用户输入的选课信息
1160             sid = sid_entry.get()
1161             cid = cid_entry.get()
1162             year = year_entry.get()
1163             term = term_cmb.get()
1164             grade = grade_entry.get()
1165             #根据输入的选课信息进行删除
1166             confirm = messagebox.askyesno('确认', f'您确定要删除此成绩信息吗? ')
1167             if confirm:
1168                 con, cur = connect_database()
1169                 sql = ' where 1'
1170                 params = []
1171                 if sid:

```

```

1172         params.append('%' + sid + '%')
1173     if cid:
1174         sql += ' and cid like %s'
1175         params.append('%' + cid + '%')
1176     if year:
1177         sql += ' and year like %s'
1178         params.append('%' + year + '%')
1179     if term:
1180         sql += ' and term like %s'
1181         params.append('%' + term + '%')
1182     if grade:
1183         sql += ' and grade like %s'
1184         params.append('%' + grade + '%')
1185     #检查要删除的课程信息是否存在
1186     cur.execute('select * from sc'+sql, params)
1187     result = cur.fetchall()
1188     if result is None:
1189         tk.messagebox.showerror('错误', '您要删除的成绩信息不存在! ')
1190     else:
1191         cur.execute('update sc set grade=null'+sql, params)
1192         con.commit()
1193         cur.close()
1194         con.close()
1195         tk.messagebox.showinfo('成功', '删除成功! ')
1196         #清空表单
1197         sid_entry.delete(0, 'end')
1198         cid_entry.delete(0, 'end')
1199         year_entry.delete(0, 'end')
1200         term_cmb.delete(0, 'end')
1201         grade_entry.delete(0, 'end')
1202     #学号输入框
1203     sid_label = tk.Label(delete_grade_window, text='学号:', font=font3)
1204     sid_label.pack()
1205     sid_entry = tk.Entry(delete_grade_window)
1206     sid_entry.pack()
1207     #课程号输入框
1208     cid_label = tk.Label(delete_grade_window, text='课程号:', font=font3)
1209     cid_label.pack()
1210     cid_entry = tk.Entry(delete_grade_window)
1211     cid_entry.pack()
1212     #学年输入框
1213     year_label = tk.Label(delete_grade_window, text='学年:', font=font3)
1214     year_label.pack()
1215     year_entry = tk.Entry(delete_grade_window)
1216     year_entry.pack()
1217     #学期下拉菜单
1218     term_label = tk.Label(delete_grade_window, text='学期:', width=17, font=font3)
1219     term_label.pack()
1220     term_cmb = ttk.Combobox(delete_grade_window, width=15, font=font3)
1221     term_cmb.pack()
1222     term_cmb['value'] = ('第一学期', '第二学期', '小学期')
1223     #成绩输入框
1224     grade_label = tk.Label(delete_grade_window, text='成绩:', font=font3)
1225     grade_label.pack()
1226     grade_entry = tk.Entry(delete_grade_window)
1227     grade_entry.pack()
1228     #确定和取消按钮
1229     confirm_button = tk.Button(delete_grade_window, text='确定', command=delete_grade, font=font3,
1230                                fg='red', cursor='hand2')
1230     confirm_button.pack()
1231     cancel_button = tk.Button(delete_grade_window, text='取消', command=delete_grade_window.
1232                                destroy, font=font3, fg='red', cursor='hand2')

```

```

1232     cancel_button.pack()
1233
1234 #成绩信息分析
1235 def analysis_grade_window():
1236     analysis_grade_window = tk.Toplevel(root)
1237     analysis_grade_window.geometry('300x500+550+150')
1238     analysis_grade_window.title('删除成绩信息')
1239     #学生的最高分
1240     def student_max_grade(sid, year, term):
1241         con, cur = connect_database()
1242         cur.execute('select max(grade) from sc where sid = %s and year = %s and term = %s', (sid,
1243             year, term))
1244         result = cur.fetchone()
1245         student_max_grade = result[0] if result else None
1246         con.commit()
1247         cur.close()
1248         con.close()
1249         return student_max_grade
1250     #学生的最低分
1251     def student_min_grade(sid, year, term):
1252         con, cur = connect_database()
1253         cur.execute('select min(grade) from sc where sid=%s and year=%s and term=%s', (sid, year,
1254             term))
1255         result = cur.fetchone()
1256         student_min_grade = result[0] if result else None
1257         con.commit()
1258         cur.close()
1259         con.close()
1260         return student_min_grade
1261     #学生的平均分
1262     def student_avg_grade(sid, year, term):
1263         con, cur = connect_database()
1264         cur.execute('select sum(c.grade*b.credit)/sum(b.credit) from course b inner join sc c on b
1265             .cid=c.cid where c.sid = %s and c.year = %s and c.term = %s', (sid, year, term))
1266         result = cur.fetchone()
1267         student_avg_grade = result[0] if result else None
1268         con.commit()
1269         cur.close()
1270         con.close()
1271         return student_avg_grade
1272     #学生的绩点
1273     def student_gpa(sid, year, term):
1274         con, cur = connect_database()
1275         cur.execute('select c.grade, b.credit from course b inner join sc c on b.cid=c.cid where c
1276             .sid=%s and c.year=%s and c.term=%s', (sid, year, term))
1277         result = cur.fetchall()
1278         con.commit()
1279         cur.close()
1280         con.close()
1281         total_credit = 0
1282         total_g = 0
1283         for grade, credit in result:
1284             if grade is not None:
1285                 if grade >= 95:
1286                     g = 4.3
1287                 elif grade >= 90:
1288                     g = 4.0
1289                 elif grade >= 85:
1290                     g = 3.7
1291                 elif grade >= 81:
1292                     g = 3.3
1293                 elif grade >= 78:

```

```

1290         g = 3.0
1291     elif grade >= 75:
1292         g = 2.7
1293     elif grade >= 72:
1294         g = 2.3
1295     elif grade >= 68:
1296         g = 2.0
1297     elif grade >= 64:
1298         g = 1.7
1299     elif grade >= 60:
1300         g = 1.3
1301     else:
1302         g = 0
1303     total_g += g * credit
1304     total_credit += credit
1305 if total_credit == 0:
1306     return None
1307 else:
1308     gpa = total_g/total_credit
1309     return gpa
1310 #学生的优秀率
1311 def student_excellent_rate(sid, year, term):
1312     con, cur = connect_database()
1313     cur.execute('select count(*) from sc where sid=%s and year=%s and term=%s and grade is not
1314                 null', (sid, year, term))
1315     total_count = cur.fetchone()[0]
1316     cur.execute('select count(*) from sc where sid=%s and year=%s and term=%s and grade>=90',
1317                 (sid, year, term))
1318     excellent_count = cur.fetchone()[0]
1319     con.commit()
1320     cur.close()
1321     con.close()
1322     if total_count > 0:
1323         student_excellent_rate = excellent_count/total_count
1324         return format(student_excellent_rate, '.2%')
1325     else:
1326         return None
1327 #学生的及格率
1328 def student_pass_rate(sid, year, term):
1329     con, cur = connect_database()
1330     cur.execute('select count(*) from sc where sid=%s and year=%s and term=%s and grade is not
1331                 null', (sid, year, term))
1332     total_count = cur.fetchone()[0]
1333     cur.execute('select count(*) from sc where sid=%s and year=%s and term=%s and grade>=60',
1334                 (sid, year, term))
1335     pass_count = cur.fetchone()[0]
1336     con.commit()
1337     cur.close()
1338     con.close()
1339     if total_count > 0:
1340         student_pass_rate = pass_count/total_count
1341         return format(student_pass_rate, '.2%')
1342     else:
1343         return None
1344 #学生的不及格率
1345 def student_fail_rate(sid, year, term):
1346     if student_pass_rate(sid, year, term) is not None:
1347         student_fail_rate = 1 - float(student_pass_rate(sid, year, term).rstrip('%'))/100
1348         return format(student_fail_rate, '.2%')
1349     else:
1350         return None
1351 #优秀门数

```

```

1348     def student_excellent_count(sid, year, term):
1349         con, cur = connect_database()
1350         cur.execute('select count(*) from sc where sid=%s and year=%s and term=%s and grade>=90',
1351                     (sid, year, term))
1352         student_excellent_count = cur.fetchone()[0]
1353         con.commit()
1354         cur.close()
1355         con.close()
1356         return student_excellent_count
1357     #及格门数
1358     def student_pass_count(sid, year, term):
1359         con, cur = connect_database()
1360         cur.execute('select count(*) from sc where sid=%s and year=%s and term=%s and grade>=60',
1361                     (sid, year, term))
1362         student_pass_count = cur.fetchone()[0]
1363         con.commit()
1364         cur.close()
1365         con.close()
1366         return student_pass_count
1367     #不及格门数
1368     def student_fail_count(sid, year, term):
1369         student_fail_count = student_total_count(sid, year, term) - student_pass_count(sid, year,
1370         term)
1371         return student_fail_count
1372     #总门数
1373     def student_total_count(sid, year, term):
1374         con, cur = connect_database()
1375         cur.execute('select count(*) from sc where sid=%s and year=%s and term=%s and grade is not
1376                     null', (sid, year, term))
1377         student_total_count = cur.fetchone()[0]
1378         con.commit()
1379         cur.close()
1380         con.close()
1381         return student_total_count
1382     #课程的最高分
1383     def course_max_grade(cid, year, term):
1384         con, cur = connect_database()
1385         cur.execute('select max(grade) from sc where cid = %s and year = %s and term = %s', (cid,
1386         year, term))
1387         result = cur.fetchone()
1388         course_max_grade = result[0] if result else None
1389         con.commit()
1390         cur.close()
1391         con.close()
1392         return course_max_grade
1393     #课程的最低分
1394     def course_min_grade(cid, year, term):
1395         con, cur = connect_database()
1396         cur.execute('select min(grade) from sc where cid=%s and year=%s and term=%s', (cid, year,
1397         term))
1398         result = cur.fetchone()
1399         course_min_grade = result[0] if result else None
1400         con.commit()
1401         cur.close()
1402         con.close()
1403         return course_min_grade
1404     #课程的平均分
1405     def course_avg_grade(cid, year, term):
1406         con, cur = connect_database()
1407         cur.execute('select avg(grade) from sc where cid = %s and year = %s and term = %s', (cid,
1408         year, term))
1409         result = cur.fetchone()

```

```

1403     course_avg_grade = result[0] if result else None
1404     con.commit()
1405     cur.close()
1406     con.close()
1407     return course_avg_grade
1408 #课程的优秀率
1409 def course_excellent_rate(cid, year, term):
1410     con, cur = connect_database()
1411     cur.execute('select count(*) from sc where cid=%s and year=%s and term=%s and grade is not
1412                 null', (cid, year, term))
1413     total_count = cur.fetchone()[0]
1414     cur.execute('select count(*) from sc where cid=%s and year=%s and term=%s and grade>=90',
1415                 (cid, year, term))
1416     excellent_count = cur.fetchone()[0]
1417     con.commit()
1418     cur.close()
1419     con.close()
1420     if total_count > 0:
1421         course_excellent_rate = excellent_count/total_count
1422         return format(course_excellent_rate, '.2%')
1423     else:
1424         return None
1425 #课程的及格率
1426 def course_pass_rate(cid, year, term):
1427     con, cur = connect_database()
1428     cur.execute('select count(*) from sc where cid=%s and year=%s and term=%s and grade is not
1429                 null', (cid, year, term))
1430     total_count = cur.fetchone()[0]
1431     cur.execute('select count(*) from sc where cid=%s and year=%s and term=%s and grade>=60',
1432                 (cid, year, term))
1433     pass_count = cur.fetchone()[0]
1434     con.commit()
1435     cur.close()
1436     con.close()
1437     if total_count > 0:
1438         course_pass_rate = pass_count/total_count
1439         return format(course_pass_rate, '.2%')
1440     else:
1441         return None
1442 #课程的不及格率
1443 def course_fail_rate(cid, year, term):
1444     if course_pass_rate(cid, year, term) is not None:
1445         course_fail_rate = 1 - float(course_pass_rate(cid, year, term).rstrip('%'))/100
1446         return format(course_fail_rate, '.2%')
1447     else:
1448         return None
1449 #优秀人数
1450 def course_excellent_count(cid, year, term):
1451     con, cur = connect_database()
1452     cur.execute('select count(*) from sc where cid=%s and year=%s and term=%s and grade>=90',
1453                 (cid, year, term))
1454     course_excellent_count = cur.fetchone()[0]
1455     con.commit()
1456     cur.close()
1457     con.close()
1458     return course_excellent_count
1459 #及格人数
1460 def course_pass_count(cid, year, term):
1461     con, cur = connect_database()
1462     cur.execute('select count(*) from sc where cid=%s and year=%s and term=%s and grade>=60',
1463                 (cid, year, term))
1464     course_pass_count = cur.fetchone()[0]

```

```

1459     con.commit()
1460     cur.close()
1461     con.close()
1462     return course_pass_count
1463 #不及格人数
1464 def course_fail_count(cid, year, term):
1465     course_fail_count = course_total_count(cid, year, term) - course_pass_count(cid, year,
1466                                         term)
1467     return course_fail_count
1468 #总人数
1469 def course_total_count(cid, year, term):
1470     con, cur = connect_database()
1471     cur.execute('select count(*) from sc where cid=%s and year=%s and term=%s and grade is not
1472                 null', (cid, year, term))
1473     course_total_count = cur.fetchone()[0]
1474     con.commit()
1475     cur.close()
1476     con.close()
1477     return course_total_count
1478 #显示学生成绩分析
1479 def student_grade_window():
1480     student_grade_window = tk.Toplevel(analysis_grade_window)
1481     student_grade_window.geometry('600x600+550+150')
1482     student_grade_window.title('学生成绩信息分析结果')
1483 #获取用户输入的选课信息
1484 sid = sid_entry.get()
1485 year = year_entry.get()
1486 term = term_cmb.get()
1487 #检查选课信息有效性
1488 if not sid:
1489     messagebox.showerror('错误', '学号不能为空! ')
1490     return
1491 if not year:
1492     messagebox.showerror('错误', '学年不能为空! ')
1493     return
1494 if not term:
1495     messagebox.showerror('错误', '学期不能为空! ')
1496     return
1497 #获取学生成绩分析计算值
1498 max_grade = student_max_grade(sid, year, term)
1499 min_grade = student_min_grade(sid, year, term)
1500 avg_grade = student_avg_grade(sid, year, term)
1501 gpa = student_gpa(sid, year, term)
1502 excellent_rate = student_excellent_rate(sid, year, term)
1503 pass_rate = student_pass_rate(sid, year, term)
1504 fail_rate = student_fail_rate(sid, year, term)
1505 excellent_count = student_excellent_count(sid, year, term)
1506 pass_count = student_pass_count(sid, year, term)
1507 fail_count = student_fail_count(sid, year, term)
1508 total_count = student_total_count(sid, year, term)
1509 result = f"""
1510 最高分: {max_grade}
1511 最低分: {min_grade}
1512 平均分: {avg_grade}
1513 绩点: {gpa}
1514 优秀率: {excellent_rate}
1515 及格率: {pass_rate}
1516 不及格率: {fail_rate}
1517 优秀门数: {excellent_count}
1518 及格门数: {pass_count}
1519 不及格门数: {fail_count}
1520 总门数: {total_count}

```

```

1519     """
1520     text = scrolledtext.ScrolledText(student_grade_window, width=800, height=400)
1521     text.insert(tk.END, result)
1522     text.pack()
1523     text.config(state=tk.DISABLED) #使用config方法将text部件设置为只读
1524 #显示课程成绩分析
1525 def course_grade_window():
1526     course_grade_window = tk.Toplevel(analysis_grade_window)
1527     course_grade_window.geometry('600x600+550+150')
1528     course_grade_window.title('课程成绩信息分析结果')
1529     #获取用户输入的选课信息
1530     cid = cid_entry.get()
1531     year = year_entry.get()
1532     term = term_cmb.get()
1533     #检查选课信息有效性
1534     if not cid:
1535         messagebox.showerror('错误', '学号不能为空! ')
1536         return
1537     if not year:
1538         messagebox.showerror('错误', '学年不能为空! ')
1539         return
1540     if not term:
1541         messagebox.showerror('错误', '学期不能为空! ')
1542         return
1543     #获取课程成绩分析计算值
1544     max_grade = course_max_grade(cid, year, term)
1545     min_grade = course_min_grade(cid, year, term)
1546     avg_grade = course_avg_grade(cid, year, term)
1547     excellent_rate = course_excellent_rate(cid, year, term)
1548     pass_rate = course_pass_rate(cid, year, term)
1549     fail_rate = course_fail_rate(cid, year, term)
1550     excellent_count = course_excellent_count(cid, year, term)
1551     pass_count = course_pass_count(cid, year, term)
1552     fail_count = course_fail_count(cid, year, term)
1553     total_count = course_total_count(cid, year, term)
1554     result = f"""
1555     最高分: {max_grade}
1556     最低分: {min_grade}
1557     平均分: {avg_grade}
1558     优秀率: {excellent_rate}
1559     及格率: {pass_rate}
1560     不及格率: {fail_rate}
1561     优秀人数: {excellent_count}
1562     及格人数: {pass_count}
1563     不及格人数: {fail_count}
1564     总人数: {total_count}
1565     """
1566     text = scrolledtext.ScrolledText(course_grade_window, width=800, height=400)
1567     text.insert(tk.END, result)
1568     text.pack()
1569     text.config(state=tk.DISABLED) #使用config方法将text部件设置为只读
1570 #学号输入框
1571 sid_label = tk.Label(analysis_grade_window, text='学号:', font=font3)
1572 sid_label.pack()
1573 sid_entry = tk.Entry(analysis_grade_window)
1574 sid_entry.pack()
1575 #课程号输入框
1576 cid_label = tk.Label(analysis_grade_window, text='课程号:', font=font3)
1577 cid_label.pack()
1578 cid_entry = tk.Entry(analysis_grade_window)
1579 cid_entry.pack()
1580 #学年输入框

```



```

1581     year_label = tk.Label(analysis_grade_window, text='学年:', font=font3)
1582     year_label.pack()
1583     year_entry = tk.Entry(analysis_grade_window)
1584     year_entry.pack()
1585     #学期下拉菜单
1586     term_label = tk.Label(analysis_grade_window, text='学期:', width=17, font=font3)
1587     term_label.pack()
1588     term_cmb = ttk.Combobox(analysis_grade_window, width=15, font=font3)
1589     term_cmb.pack()
1590     term_cmb['value'] = ('第一学期', '第二学期', '小学期')
1591     #确定和取消按钮
1592     confirm_student_button = tk.Button(analysis_grade_window, text='学生成绩分析', command=
        student_grade_window, font=font3, fg='red', cursor='hand2')
1593     confirm_student_button.pack()
1594     confirm_course_button = tk.Button(analysis_grade_window, text='课程成绩分析', command=
        course_grade_window, font=font3, fg='red', cursor='hand2')
1595     confirm_course_button.pack()
1596     cancel_button = tk.Button(analysis_grade_window, text='取消', command=analysis_grade_window.
        destroy, font=font3, fg='red', cursor='hand2')
1597     cancel_button.pack()
1598
1599     root = tk.Tk()
1600     root.geometry('900x500+250+150')
1601     root.title('学校信息管理系统')
1602     #字体
1603     font1=tkfont.Font(family='华文中宋', size=20, weight=tkfont.BOLD)
1604     font2=tkfont.Font(family='微软雅黑', size=15)
1605     font3=tkfont.Font(family='微软雅黑', size=10)
1606     #标题
1607     title = tk.Label(root, text='学校信息管理系统', font=font1)
1608     title.pack()
1609
1610     #6个frame
1611     frame1 = tk.Frame(root, width=900, height=90, bd=25)
1612     frame1.pack()
1613     frame2 = tk.Frame(root, width=900, height=90, bd=10)
1614     frame2.pack()
1615     frame3 = tk.Frame(root, width=900, height=90, bd=10)
1616     frame3.pack()
1617     frame4 = tk.Frame(root, width=900, height=90, bd=10)
1618     frame4.pack()
1619     frame5 = tk.Frame(root, width=900, height=90, bd=10)
1620     frame5.pack()
1621     frame6 = tk.Frame(root, width=900, height=90, bd=10)
1622     frame6.pack()
1623
1624     #frame1: 信息管理标签
1625     label_student = tk.Label(frame1, text='学生信息管理', font=font2)
1626     label_student.grid(row=0, column=0)
1627     label1_1 = tk.Label(frame1, width=11)
1628     label1_1.grid(row=0, column=1)
1629     label_course = tk.Label(frame1, text='课程信息管理', font=font2)
1630     label_course.grid(row=0, column=2)
1631     label1_2 = tk.Label(frame1, width=11)
1632     label1_2.grid(row=0, column=3)
1633     label_sc = tk.Label(frame1, text='选课信息管理', font=font2)
1634     label_sc.grid(row=0, column=4)
1635     label1_3 = tk.Label(frame1, width=11)
1636     label1_3.grid(row=0, column=5)
1637     label_grade = tk.Label(frame1, text='成绩信息管理', font=font2)
1638     label_grade.grid(row=0, column=6)
1639

```

```

1640 #frame2: 添加信息按钮
1641 add_student_button = tk.Button(frame2, text='添加学生信息', command=add_student_window, font=font3
    , fg='red', padx=30, cursor='hand2')
1642 add_student_button.grid(row=0, column=0)
1643 label2_1 = tk.Label(frame2, width=8)
1644 label2_1.grid(row=0, column=1)
1645 add_course_button = tk.Button(frame2, text='添加课程信息', command=add_course_window, font=font3,
    fg='red', padx=30, cursor='hand2')
1646 add_course_button.grid(row=0, column=2)
1647 label2_2 = tk.Label(frame2, width=8)
1648 label2_2.grid(row=0, column=3)
1649 add_sc_button = tk.Button(frame2, text='添加选课信息', command=add_sc_window, font=font3, fg='red'
    , padx=30, cursor='hand2')
1650 add_sc_button.grid(row=0, column=4)
1651 label2_3 = tk.Label(frame2, width=8)
1652 label2_3.grid(row=0, column=5)
1653 add_grade_button = tk.Button(frame2, text='添加成绩信息', command=add_grade_window, font=font3, fg
    ='red', padx=30, cursor='hand2')
1654 add_grade_button.grid(row=0, column=6)
1655
1656 #frame3: 查询信息按钮
1657 select_student_button = tk.Button(frame3, text='查询学生信息', command=select_student_window, font
    =font3, fg='red', padx=30, cursor='hand2')
1658 select_student_button.grid(row=0, column=0)
1659 label3_1 = tk.Label(frame3, width=8)
1660 label3_1.grid(row=0, column=1)
1661 select_course_button = tk.Button(frame3, text='查询课程信息', command=select_course_window, font=
    font3, fg='red', padx=30, cursor='hand2')
1662 select_course_button.grid(row=0, column=2)
1663 label3_2 = tk.Label(frame3, width=8)
1664 label3_2.grid(row=0, column=3)
1665 select_sc_button = tk.Button(frame3, text='查询选课信息', command=select_sc_window, font=font3, fg
    ='red', padx=30, cursor='hand2')
1666 select_sc_button.grid(row=0, column=4)
1667 label3_3 = tk.Label(frame3, width=8)
1668 label3_3.grid(row=0, column=5)
1669 select_grade_button = tk.Button(frame3, text='查询成绩信息', command=select_grade_window, font=
    font3, fg='red', padx=30, cursor='hand2')
1670 select_grade_button.grid(row=0, column=6)
1671
1672 #frame4: 更新信息按钮
1673 update_student_button = tk.Button(frame4, text='更新学生信息', command=update_student_window, font
    =font3, fg='red', padx=30, cursor='hand2')
1674 update_student_button.grid(row=0, column=0)
1675 label4_1 = tk.Label(frame4, width=8)
1676 label4_1.grid(row=0, column=1)
1677 select_course_button = tk.Button(frame4, text='更新课程信息', command=update_course_window, font=
    font3, fg='red', padx=30, cursor='hand2')
1678 select_course_button.grid(row=0, column=2)
1679 label4_2 = tk.Label(frame4, width=8)
1680 label4_2.grid(row=0, column=3)
1681 select_sc_button = tk.Button(frame4, text='更新选课信息', command=update_sc_window, font=font3, fg
    ='red', padx=30, cursor='hand2')
1682 select_sc_button.grid(row=0, column=4)
1683 label4_3 = tk.Label(frame4, width=8)
1684 label4_3.grid(row=0, column=5)
1685 select_grade_button = tk.Button(frame4, text='更新成绩信息', command=update_grade_window, font=
    font3, fg='red', padx=30, cursor='hand2')
1686 select_grade_button.grid(row=0, column=6)
1687
1688 #frame5: 删除信息按钮

```

```

1689 delete_student_button = tk.Button(frame5, text='删除学生信息', command=delete_student_window, font
      =font3, fg='red', padx=30, cursor='hand2')
1690 delete_student_button.grid(row=0, column=0)
1691 label15_1 = tk.Label(frame5, width=8)
1692 label15_1.grid(row=0, column=1)
1693 select_course_button = tk.Button(frame5, text='删除课程信息', command=delete_course_window, font=
      font3, fg='red', padx=30, cursor='hand2')
1694 select_course_button.grid(row=0, column=2)
1695 label15_2 = tk.Label(frame5, width=8)
1696 label15_2.grid(row=0, column=3)
1697 select_sc_button = tk.Button(frame5, text='删除选课信息', command=delete_sc_window, font=font3, fg
      ='red', padx=30, cursor='hand2')
1698 select_sc_button.grid(row=0, column=4)
1699 label15_3 = tk.Label(frame5, width=8)
1700 label15_3.grid(row=0, column=5)
1701 select_grade_button = tk.Button(frame5, text='删除成绩信息', command=delete_grade_window, font=
      font3, fg='red', padx=30, cursor='hand2')
1702 select_grade_button.grid(row=0, column=6)
1703
1704 #frame6: 成绩信息分析
1705 label16 = tk.Label(frame6, width=89)
1706 label16.grid(row=0, column=0)
1707 grade_analysis_button = tk.Button(frame6, text='成绩信息分析', command=analysis_grade_window, font
      =font3, fg='red', padx=30, cursor='hand2')
1708 grade_analysis_button.grid(row=0, column=1)
1709
1710 root.mainloop() #启动主事件循环

```

## 5 软件使用说明

如果使用 OpenGauss 数据库运行，可以直接使用以上代码。如果使用 Sqlite 数据库运行，需要将第 11 行 connect\_database() 函数前加 #，并删去第 21 行前面的 #。

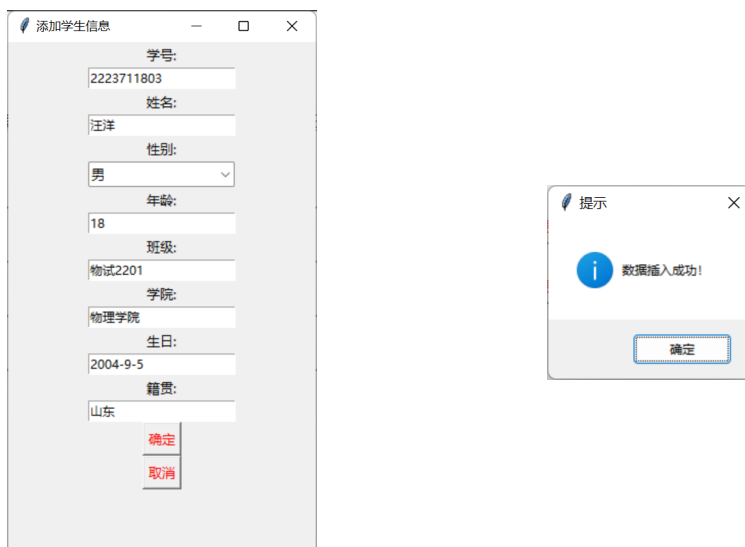
用户初始界面如下：



本学校信息管理系统目前可运行功能说明如下：

## 5.1 学生信息管理

1. **添加学生信息**：单击“添加学生信息”按钮，输入学生的学号、姓名、性别、年龄、班级、学院、生日、籍贯信息，将这些信息存储到数据库中。



The image shows two windows from a software application. The left window is titled "添加学生信息" (Add Student Information) and contains a form with the following fields: 学号 (ID Number) with value 2223711803, 姓名 (Name) with value 汪洋, 性别 (Gender) with a dropdown menu set to 男 (Male), 年龄 (Age) with value 18, 班级 (Class) with value 物试2201, 学院 (College) with value 物理学院, 生日 (Date of Birth) with value 2004-9-5, and 籍贯 (Hometown) with value 山东. There are "确定" (Confirm) and "取消" (Cancel) buttons at the bottom. The right window is a small "提示" (Message) dialog box with a blue information icon and the text "数据插入成功!" (Data insertion successful!), with a "确定" (Confirm) button.

学生信息要求：

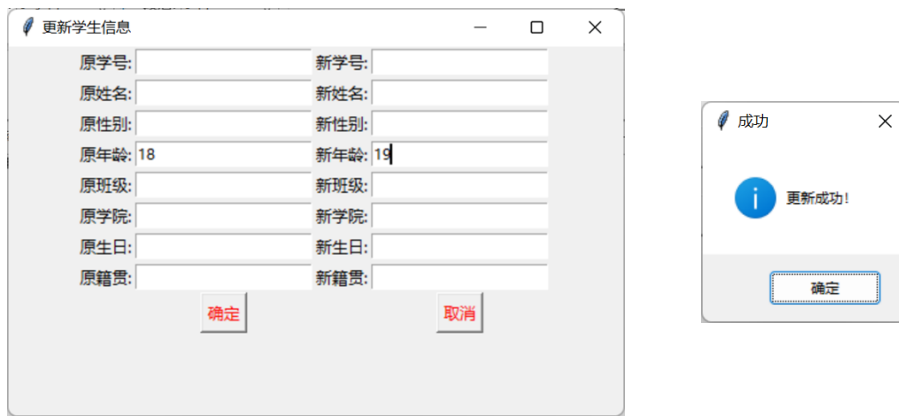
- (1) 学号不能为空。
  - (2) 姓名不能为空。
  - (3) 年龄必须是整数。
  - (4) 生日必须是"YYYY-MM-DD" 格式的日期。
2. **查询学生信息**：单击“查询学生信息”按钮，输入学生的某一项信息或某几项信息，查询相应学生信息。



The image shows two windows. The left window is titled "查询学生信息" (Query Student Information) and contains a form with the same fields as the "Add Student Information" window, but with the "确定" (Confirm) and "取消" (Cancel) buttons at the bottom. The right window is the main application window titled "学校信息管理系统" (School Information Management System). It displays a table with the following data: 2223711803, 汪洋, 男, 18, 物试2201, 物理学院, 2004-09-05 00:00:00, 山东.

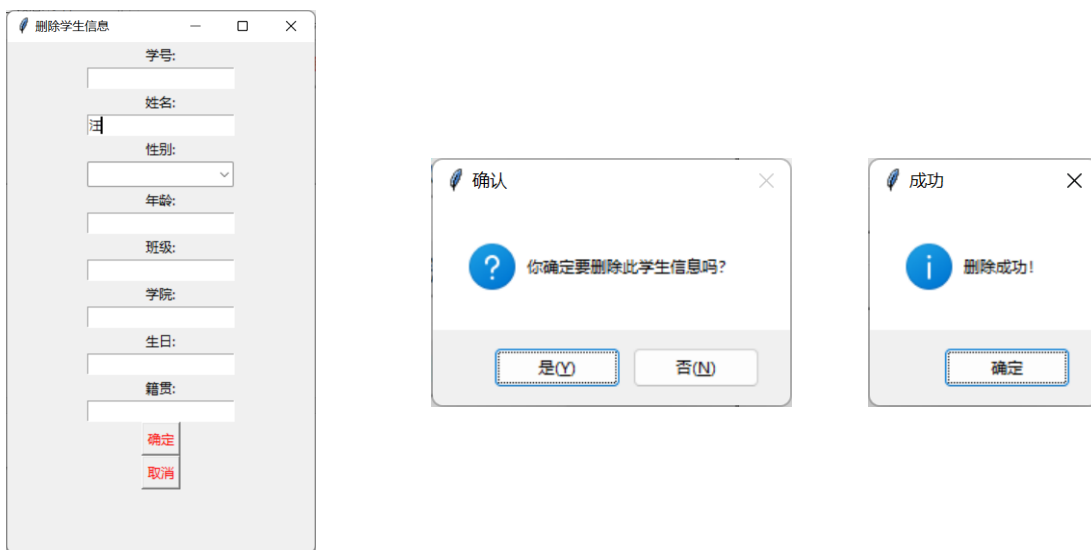
查询信息要求：

- (1) 年龄必须是整数。
  - (2) 生日必须是"YYYY-MM-DD" 格式的日期。
- 在姓名栏输入某个字，会查询名字中带这个字的人。
3. **更新学生信息**：单击“更新学生信息”按钮，输入学生的某一项信息或某几项信息，更新相应学生信息。



旧值和新值不能为空。

4. **删除学生信息**：单击“删除学生信息”按钮，输入学生的某一项信息或某几项信息，删除相应学生信息。

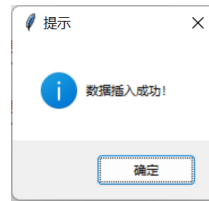


删除信息要求：

- (1) 年龄必须是整数。
- (2) 生日必须是"YYYY-MM-DD" 格式的日期。

## 5.2 课程信息管理

1. **添加课程信息**：单击“添加课程信息”按钮，输入课程名称、课程号、教师、学分、先修课程信息，将这些信息存储到数据库中。



课程信息要求：

- (1) 课程号不能为空。
- (2) 课程名不能为空。

2. **查询课程信息：**单击“查询课程信息”按钮，输入课程的某一项信息或某几项信息，查询相应课程信息。

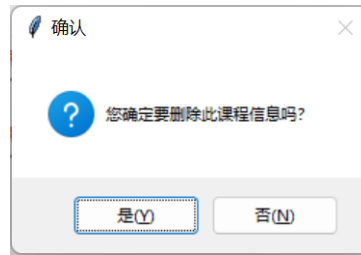


3. **更新课程信息：**单击“更新课程信息”按钮，输入课程的某一项信息或某几项信息，更新相应课程信息。



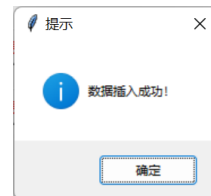
旧值和新值不能为空。

4. **删除课程信息：**单击“删除课程信息”按钮，输入课程的某一项信息或某几项信息，删除相应课程信息。



### 5.3 选课信息管理

1. **添加选课信息**：单击“添加选课信息”按钮，输入学号、课程号、学年、学期，将这些信息存储到数据库中。



选课信息要求：

- (1) 学号不能为空。
  - (2) 课程号不能为空。
  - (3) 学年不能为空。
  - (4) 学期不能为空。
2. **查询选课信息**：单击“查询选课信息”按钮，输入选课的某一项信息或某几项信息，查询相应选课信息。

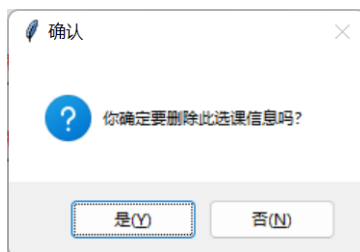


3. **更新选课信息**：单击“更新选课信息”按钮，输入选课的一项信息或某几项信息，更新相应选课信息。



旧值和新值不能为空。

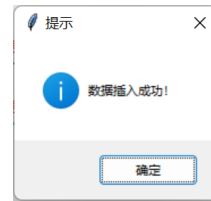
4. **删除选课信息**：单击“删除选课信息”按钮，输入选课的一项信息或某几项信息，删除相应选课信息。



## 5.4 成绩管理

1. **添加成绩信息**：单击“添加成绩信息”按钮，输入学生号、课程号、学年、学期、成绩，将成绩信息存储到数据库中。





成绩信息要求：

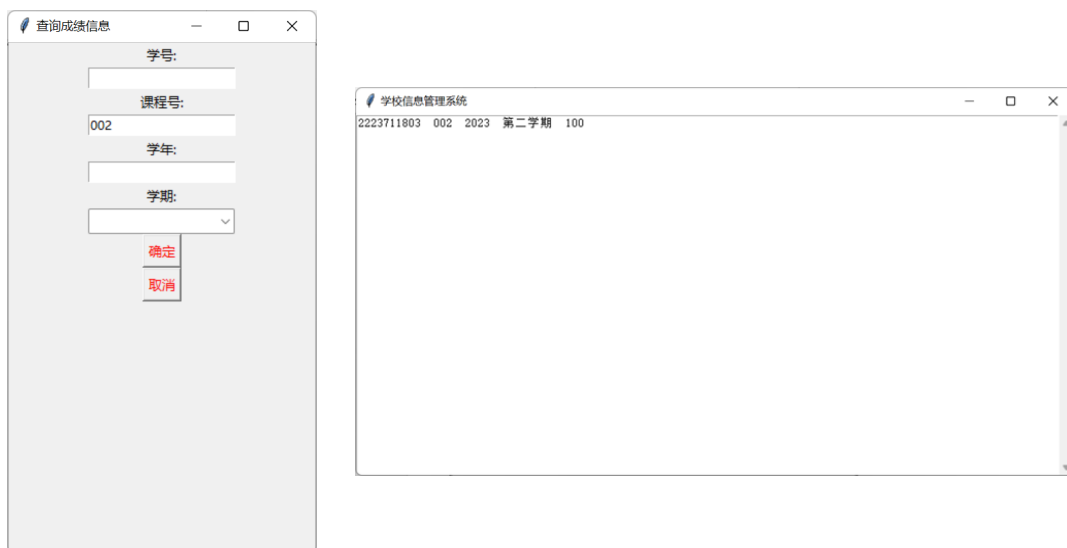
- (1) 学号不能为空。
- (2) 课程号不能为空。
- (3) 成绩必须是数值。

2. **查询成绩信息：**单击“添加成绩信息”按钮，输入选课的某一项信息或某几项信息，查询相应成绩信息。

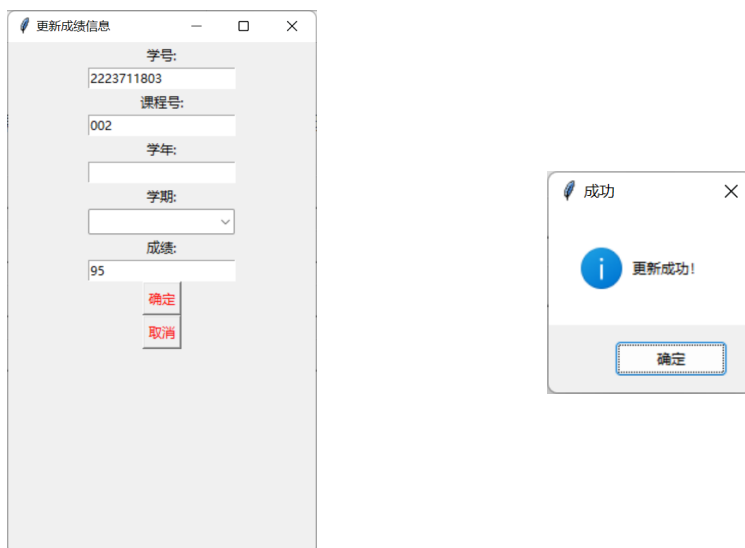
- (1) 查询学生成绩信息。



- (2) 查询课程成绩信息。

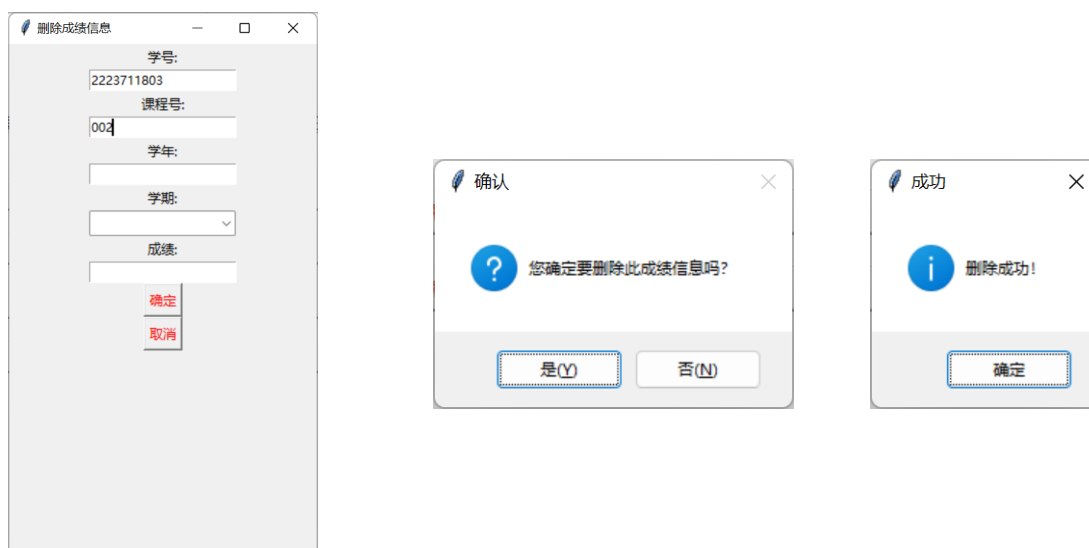


3. **更新成绩信息**：单击“添加成绩信息”按钮，输入选课的一项信息或某几项信息，更新相应成绩信息。



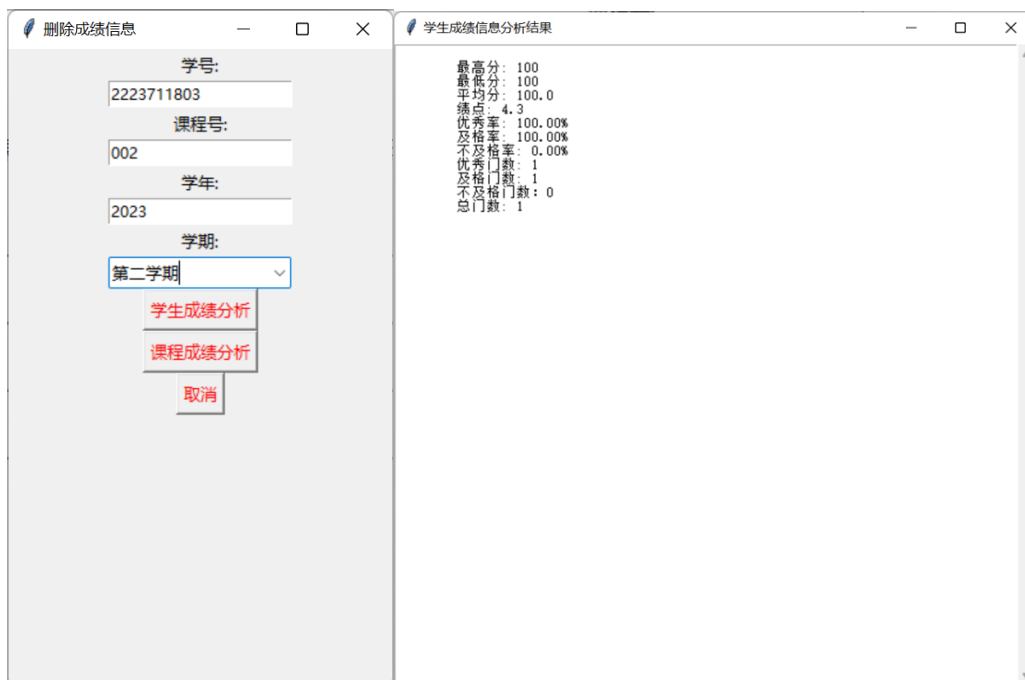
旧值和新值不能为空。

4. **删除成绩信息**：单击“添加成绩信息”按钮，输入选课的一项信息或某几项信息，删除相应成绩信息。

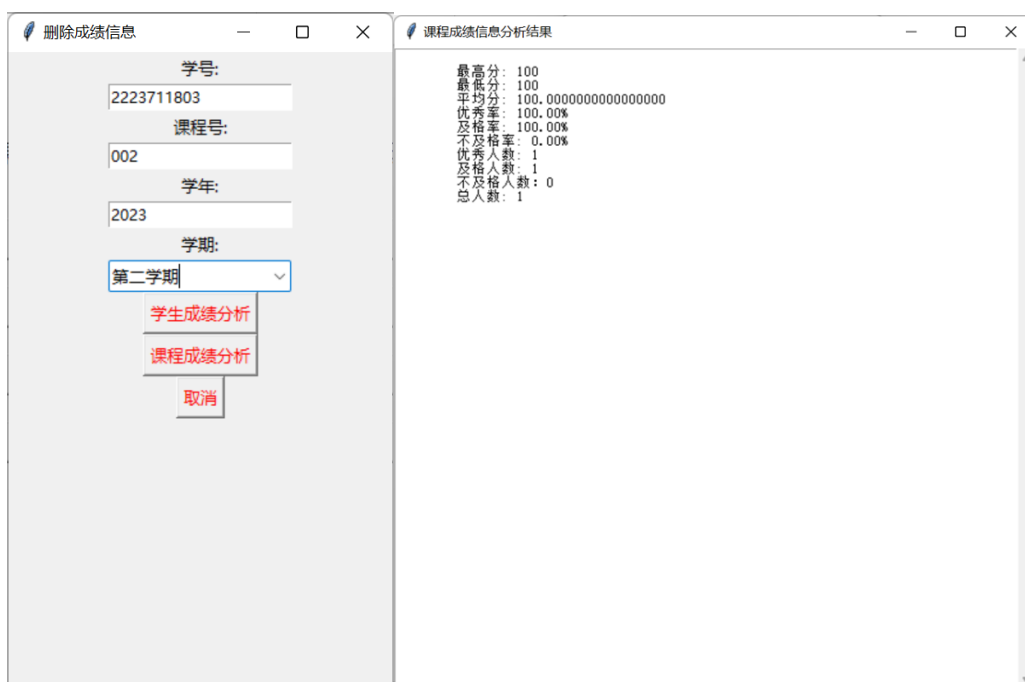


5. **成绩信息分析**：单击“成绩信息分析”按钮。

- (1) 学生成绩分析：输入学号、学年、学期，计算学生最高分、最低分、平均分（考虑课程学分权重）、绩点（最高 4.3）、优秀率（90 分以上）、及格率（60 分以上）、不及格率（60 分以下）、优秀门数（90 分以上）、及格门数（60 分以上）、不及格门数（60 分以下）、总门数。



- (2) 课程成绩分析：输入课程号、学年、学期，计算课程最高分、最低分、平均分、优秀率（90 分以上）、及格率（60 分以上）、不及格率（60 分以下）、优秀人数（90 分以上）、及格人数（60 分以上）、不及格人数（60 分以下）、总人数。



注：绩点对应：95-100 分为 4.3，90-95 分为 4.0，85-90 分为 3.7，81-85 分为 3.3，78-81 分为 3.0，75-78 分为 2.7，72-75 分为 2.3，68-72 分为 2.0，64-68 分为 1.7，60-64 分为 1.3，小于 60 分为 0，GPA 采取平均学分绩计算方法。

## 6 后记

本程序是基于 Python3 的 tkinter 库设计的可视化数据库操作窗口。作为学校信息管理系统 1.0，不是最优程序，仍有很大优化空间。

可能的优化方向：

1. 输入框的循环优化。
2. 用户登录窗口的实现。
3. 成绩信息实现平时成绩、期中成绩、期末成绩的区分。
4. 信息更新与删除界面实现教师端与学生端的分离。
5. 用户界面美化。

经初步调试，本程序可正常运行。虑及本人水平有限，潜在的代码错误与冗余在所难免，用户如在使用时发现 bug，或者有关于本程序的修改意见，欢迎与我联系 wangyangdh2015@126.com。

## 7 致谢

感谢赵英良老师在教学过程中给予的悉心指导。