

**课程 实 验 报 告**

**课程名称： Python程序设计**

**专业班级： 自实1901**

**学 号： U201916457**

**姓 名： 张皓然**

**指导教师： 卢仁智**

**报告日期： 2020年11月23日**

**人工智能与自动化学院**

# 题目

## 1题目

用Python编写一个学术成绩管理系统，功能包括但不限于以下：

1. 学生注册登录

2. 学生成绩录入

3. 个人成绩查询

4. 统计个人成绩

5. 统计班级成绩

## 2设计思路与代码实现

### 2.1制作流程

1.设计图形界面

2.设计简单的算法

3.完成代码编写

## 3程序介绍

### 开始界面

|  |
| --- |
|  |
| 图1. 开始界面 |

如图为程序的开始界面，普通学生在界面中可以按照自己的学号和密码进行登录，老师可以通过管理员入口进入管理员界面，对班级以及学生成绩进行管理。

### 学生注册登录

一名新学生如果想要注册新账号的话，在开始界面中点击“注册”即可。

|  |
| --- |
|  |
| 图2. 注册界面 |

若学号与文件内已有的学号重复，则会提示错误，不能注册该账号。

|  |
| --- |
|  |
| 图3. 注册失败图 |

若账号注册成功，则可以正常进行登录。

### 学生成绩录入

在管理员界面中，我们可以对学生的成绩进行录入以及修改操作。

|  |
| --- |
|  |
| 图4. 输入管理员密匙 |

**管理员密匙为“hust\_aia”**。进入管理员界面后，可以添加新的班级，或者管理学生成绩。

|  |
| --- |
|  |
| 图5. 管理功能图 |

如果点击添加班级，则可以新建一个班级的文件，如图：

|  |
| --- |
|  |
| 图6. 初始班级信息文件 |

点击添加班级后，就会新增一个文件！

|  |
| --- |
|  |
| 图7. 成功添加一个班级 |

|  |
| --- |
|  |
| 图8. 产生新班级的文件 |

此外，可以在管理学生成绩的界面中对单一的学生信息进行管理。

|  |
| --- |
|  |
| 图9. 学生成绩管理界面 |

如果某学生的成绩过于糟糕，我们就可以将其退学，例如：

|  |
| --- |
|  |
| 图10. 将U201916457退学 |

|  |
| --- |
|  |
| 图11. 退学前 |

|  |
| --- |
|  |
| 图12. 进行退学 |

|  |
| --- |
|  |
| 图13. 退学后 |

### 个人成绩查询

登录账号后，

|  |
| --- |
|  |
| 图14. 张皓然的欢迎界面 |

程序可以对三门成绩进行查询。

|  |
| --- |
|  |
| 图15. lesson1的成绩展示 |

### 统计个人成绩

程序也可以展示个人平均成绩。

|  |
| --- |
|  |
| 图16. 张皓然的个人平均成绩 |

### 统计班级成绩

程序还可以展示班级的总体成绩情况，自动按总分从高到底排序。

|  |
| --- |
|  |
| 图17. 1班的成绩情况（自动按总分从高到底排序） |

## 实验结果

程序功能丰富，满足题目要求。

## 感悟

为更好的完成最后Python的课程设计，我在网上自学了Python的Tkinter库，惊喜的发现Python设计界面非常简单。与C语言相比，Python确实更加简单易用，学习成本更低。一个C语言的程序，我们做了几个月，但如果用Python实现相同的功能，可能一个月就可以完成。在计算机算力逐渐增强的今天，Python解放了我们研究编程语言本身的时间，不必再花时间去思考复杂的指针，从而让我们把更多的时间花费在思考算法，梳理逻辑的上面，这确实是功德一件。

## 附件

1. # Student Score System
2. # function:
3. # 1. login and sign in
4. # 2. append new scores
5. # 3. show own scores
6. # 4. show class scores
7. **from** tkinter **import** \*
8. **import** pandas as pd
9. **import** os

12. **def** quit(root):
13. root.destroy()

16. **def** open\_Stu(class\_number):
17. **try**:
18. data = pd.read\_csv('class'+str(class\_number)+'.csv')
19. StuList = data.values.tolist()
20. **except**:
21. StuList = 0
22. **return** StuList

25. **def** show\_single(manager, inp1, lesson):
26. aim = inp1.get()
27. inp1.delete(0, END)
28. score = lesson.get(aim, '查找失败')
30. lb1 = Label(manager, text='您'+aim+'的成绩为：'+str(score))
31. lb1.place(relx=0.1, rely=0.6, relwidth=0.7, relheight=0.4)

34. **def** sort\_score(data):
35. all\_score = []
36. **for** i **in** range(len(data)):
37. all\_score.append(int(data[i][3])+int(data[i][4])+int(data[i][5]))
38. **for** i **in** range(len(data)):
39. **for** j **in** range(i, len(data)):
40. **if** all\_score[i] < all\_score[j]:
41. temp = all\_score[j]
42. all\_score[j] = all\_score[i]
43. all\_score[i] = temp
44. temp1 = data[i]
45. data[i] = data[j]
46. data[j] = temp1
47. **return** data

50. **def** show\_average(manager, lesson):
51. score\_aver = 0
52. **for** key **in** lesson.keys():
53. score\_aver += float(lesson[key])
54. score\_aver /= len(lesson)
55. lesson\_str = '\n'
56. **for** key **in** lesson.keys():
57. lesson\_str += key
58. lesson\_str +=': '
59. lesson\_str += str(lesson[key])
60. lesson\_str += '\n'
62. lb1 = Label(manager, text='您的平均成绩为'+str(score\_aver)+lesson\_str)
63. lb1.place(relx=0.1, rely=0.6, relwidth=0.7, relheight=0.4)

66. **def** show\_class(manager, orgin\_data):
67. # 展示各科平均成绩
68. # 展示自己的排名
69. data = orgin\_data.values.tolist()
70. # print(data)
71. head = orgin\_data.keys()
73. sort\_score(data)
75. txt = '\n'
76. **for** item **in** data:
77. txt = txt + item[0] + ' '  # 添加名字
78. **for** i **in** range(3, len(data[0])):
79. txt += head[i] + ': ' + str(item[i])  # 添加各科成绩
80. txt += ' '
81. txt += '\n'
82. txt += '平均成绩\n'
83. **for** i **in** range(3, len(data[0])):
84. txt += head[i] + ': '
85. aver = 0
86. **for** item **in** data:
87. aver += item[i]
88. aver /= len(data)
89. txt += str(aver)
90. txt += '\n'
91. lb1 = Label(manager, text=txt)
92. lb1.place(relx=0.1, rely=0.6, relwidth=0.7, relheight=0.4)

95. **def** manager\_sys(account, class\_number, root):
96. quit(root)  # 关闭原始界面，创建新的界面
97. **print**(class\_number)
98. orgin\_data = pd.read\_csv('class'+str(class\_number)+'.csv')
99. data = orgin\_data.values.tolist()
100. head = orgin\_data.keys()
101. lesson = {}  # 字典
102. **for** item **in** data:
103. **if** account **in** item[1]:
104. **print**(item)
105. name = item[0]
106. StuID = item[1]
107. **for** i **in** range(3, len(item)):
108. lesson[head[i]] = item[i]  # 采集所有成绩信息
109. **break**
111. manager = Tk()  # 新的界面
112. manager.title('StuScore Manager ' + str(StuID))
113. manager.geometry('400x400')
115. lb1 = Label(manager, text=str(name)+' 欢迎来到StuScore Manager!')
116. lb1.place(relx=0, rely=0.1, relwidth=0.9, relheight=0.1)
118. lb2 = Label(manager, text='查询科目成绩')
119. lb2.place(relx=0.1, rely=0.2, relwidth=0.2, relheight=0.1)
121. inp1 = Entry(manager)
122. inp1.place(relx=0.3, rely=0.225, width=150)
124. btn0 = Button(manager, text='查询该科', command=**lambda**: show\_single(manager, inp1, lesson))
125. btn0.place(relx=0.35, rely=0.3, relwidth=0.2, relheight=0.1)
127. btn1 = Button(manager, text='个人平均成绩', command=**lambda**: show\_average(manager, lesson))
128. btn1.place(relx=0.35, rely=0.4, relwidth=0.2, relheight=0.1)
130. btn2 = Button(manager, text='班级成绩情况', command=**lambda**: show\_class(manager, orgin\_data))
131. btn2.place(relx=0.35, rely=0.5, relwidth=0.2, relheight=0.1)

134. **def** check\_acc\_code(inp1, inp2, root):
135. account = inp1.get()
136. code = inp2.get()
137. acc\_mark = 0
138. code\_mark = 0
139. i = 1
140. data = open\_Stu(i)
141. **while**(data):
142. **for** item **in** data:  # item[1] is ID
143. **if** account == item[1]:
144. acc\_mark = 1
145. **if** code == item[2]:
146. code\_mark = 1
147. **if** acc\_mark == 1 **and** code\_mark == 1:  # 账户密码匹配成功
148. **break**
149. **else**:
150. acc\_mark = 0
151. code\_mark = 0
152. **if** acc\_mark == 1 **and** code\_mark == 1:  # 账户密码匹配成功
153. **break**
154. i += 1
155. data = open\_Stu(i)  # 读取下一个班级的文件
157. class\_number = i  # 该学生所在班号
158. **if** acc\_mark == 1 **and** code\_mark == 1:  # 账户密码匹配成功
159. manager\_sys(account, class\_number, root)
160. **else**:  # 失败
161. lb1 = Label(root, text='账户或密码有误')
162. lb1.place(relx=0.35, rely=0.9, width=100)

165. **def** write\_stu(ID, password, login, name, classNum):
166. data\_orgin = pd.read\_csv('class' + str(classNum) + '.csv')
167. data = data\_orgin.values.tolist()
168. head = data\_orgin.keys()
169. new\_stu = []
170. new\_stu.append(name)
171. new\_stu.append(ID)
172. new\_stu.append(password)
173. **print**(data)
174. **print**(len(data))
175. **for** i **in** range(3, len(data[0])):
176. new\_stu.append('60')  # 初始化分数
177. # print(new\_stu)
178. name\_list = []
179. ID\_list = []
180. password\_list = []
181. lesson = [item[3:len(data[0])] **for** item **in** data]
182. lesson = [[str(row[i]) **for** row **in** lesson] **for** i **in** range(len(lesson[0]))]
184. **for** item **in** data:
185. name\_list.append(item[0])
186. ID\_list.append(item[1])
187. password\_list.append(item[2])
188. name\_list.append(new\_stu[0])
189. ID\_list.append(new\_stu[1])
190. password\_list.append(new\_stu[2])
191. **for** i **in** range(len(lesson)):
192. lesson[i].append(new\_stu[i+3])
193. new\_data = {head[0]: name\_list, head[1]: ID\_list, head[2]: password\_list}
194. **for** i **in** range(3, len(data[0])):
195. new\_data[head[i]] = lesson[i-3]
196. df = pd.DataFrame(new\_data)
197. df.to\_csv('class'+str(classNum)+'.csv', index=False, sep=',')
198. go\_main(login)

201. **def** login\_check(inp1, inp2, inp3, inp4, login):
202. # 检查学号是否重复
203. ID = str(inp1.get())
204. password = str(inp2.get())
205. classNum = inp3.get()
206. name = inp4.get()
207. data = pd.read\_csv('class' + str(classNum) + '.csv')
208. StuList = data.values.tolist()
209. sign = 1
210. **for** item **in** StuList:
211. **if** ID == item[1]:
212. sign = 0
213. **break**
214. **if** sign == 1:  # 代表注册成功
215. write\_stu(ID, password, login, name, classNum)
216. **else**:
217. lb1 = Label(login, text='注册失败')
218. lb1.place(relx=0.4, rely=0.8, relwidth=0.15, relheight=0.1)

221. **def** go\_main(root):
222. quit(root)
223. main\_sys()

226. **def** login(root):
227. login = Tk()
228. login.title('注册新账号')
229. login.geometry('400x400')
230. quit(root)
231. lb1 = Label(login, text='学号')
232. lb1.place(relx=0.1, rely=0.1, relwidth=0.1, relheight=0.1)
234. lb2 = Label(login, text='密码')
235. lb2.place(relx=0.1, rely=0.2, relwidth=0.1, relheight=0.1)
237. lb3 = Label(login, text='班级(如1,2,3)')
238. lb3.place(relx=0.05, rely=0.3, relwidth=0.25, relheight=0.1)
240. lb4 = Label(login, text='姓名')
241. lb4.place(relx=0.05, rely=0.4, relwidth=0.1, relheight=0.1)
243. inp1 = Entry(login)  # 登录
244. inp1.place(relx=0.3, rely=0.125, width=200)
246. inp2 = Entry(login)  # 密码
247. inp2.place(relx=0.3, rely=0.225, width=200)
249. inp3 = Entry(login)  # 班级
250. inp3.place(relx=0.3, rely=0.325, width=200)
252. inp4 = Entry(login)  # 姓名
253. inp4.place(relx=0.3, rely=0.425, width=200)
255. btn1 = Button(login, text='注册', command=**lambda**: login\_check(inp1, inp2, inp3, inp4, login))
256. btn1.place(relx=0.4, rely=0.5, relwidth=0.15, relheight=0.1)
258. btn2 = Button(login, text='返回', command=**lambda**: go\_main(login))  # 进行注册
259. btn2.place(relx=0.4, rely=0.6, relwidth=0.15, relheight=0.1)

262. **def** append\_class(control):
263. file\_name = os.listdir()  # 读取当前路径的文件
264. **for** item **in** file\_name:
265. **if** '.csv' **not** **in** item:
266. aim = file\_name.index(item)
267. **del** file\_name[aim]
268. num = len(file\_name) + 1
269. txt = 'class' + str(num) + '.csv'
270. new\_file = open(txt, 'w')
271. txt2 = 'name,stuID,code,lesson1,lesson2,lesson3\nanyone,U000,1234,60,60,60\n'
272. new\_file.write(txt2)
274. lb1 = Label(control, text='添加成功')
275. lb1.place(relx=0.35, rely=0.5, relwidth=0.2, relheight=0.1)

278. **def** del\_stu(inp1, inp2, manage):
279. classNum = inp1.get()
280. data\_orgin = pd.read\_csv('class' + str(classNum) + '.csv')
281. data = data\_orgin.values.tolist()
282. head = data\_orgin.keys()
283. ID = inp2.get()
284. **for** i **in** range(len(data)):
285. **if** ID == data[i][1]:
286. **del** data[i]
287. **break**
288. name\_list = []
289. ID\_list = []
290. password\_list = []
291. lesson = [item[3:len(data[0])] **for** item **in** data]
292. lesson = [[str(row[i]) **for** row **in** lesson] **for** i **in** range(len(lesson[0]))]
294. **for** item **in** data:
295. name\_list.append(item[0])
296. ID\_list.append(item[1])
297. password\_list.append(item[2])
298. new\_data = {head[0]: name\_list, head[1]: ID\_list, head[2]: password\_list}
299. **for** i **in** range(3, len(data[0])):
300. new\_data[head[i]] = lesson[i - 3]
301. df = pd.DataFrame(new\_data)
302. df.to\_csv('class' + str(classNum) + '.csv', index=False, sep=',')

305. lb1 = Label(manage, text='退学成功')
306. lb1.place(relx=0.3, rely=0.7, relwidth=0.2, relheight=0.1)

309. **def** change\_score(inp1, inp2, inp3, inp4, manage):
310. classNum = inp1.get()
311. ID = inp2.get()
312. new\_score = inp3.get()
313. lesson\_ = inp4.get()
315. data\_orgin = pd.read\_csv('class' + str(classNum) + '.csv')
316. data = data\_orgin.values.tolist()
317. head = data\_orgin.keys()
319. new\_stu = []
321. **for** i **in** range(len(data)):
322. **if** ID == data[i][1]:
323. new\_stu.append(data[i][0])
324. new\_stu.append(data[i][1])
325. new\_stu.append(data[i][2])
326. **for** j **in** range(3, len(data[0])):
327. new\_stu.append(data[i][j])
328. **del** data[i]
329. **break**
330. **for** i **in** range(len(data)):
331. **if** lesson\_ == head[i]:
332. new\_stu[i] = new\_score
333. name\_list = []
334. ID\_list = []
335. password\_list = []
336. lesson = [item[3:len(data[0])] **for** item **in** data]
337. lesson = [[str(row[i]) **for** row **in** lesson] **for** i **in** range(len(lesson[0]))]
339. **for** item **in** data:
340. name\_list.append(item[0])
341. ID\_list.append(item[1])
342. password\_list.append(item[2])
343. name\_list.append(new\_stu[0])
344. ID\_list.append(new\_stu[1])
345. password\_list.append(new\_stu[2])
346. **for** i **in** range(len(lesson)):
347. lesson[i].append(new\_stu[i + 3])
348. new\_data = {head[0]: name\_list, head[1]: ID\_list, head[2]: password\_list}
349. **for** i **in** range(3, len(data[0])):
350. new\_data[head[i]] = lesson[i - 3]
351. df = pd.DataFrame(new\_data)
352. df.to\_csv('class' + str(classNum) + '.csv', index=False, sep=',')
353. go\_main(manage)

356. **def** manage\_class(control):
357. quit(control)
358. manage = Tk()
359. manage.title('manager')
360. manage.geometry('400x400')
362. lb1 = Label(manage, text='班级')
363. lb1.place(relx=0.2, rely=0.1, width=30)
365. lb2 = Label(manage, text='学号')
366. lb2.place(relx=0.2, rely=0.2, width=30)
368. lb3 = Label(manage, text='成绩')
369. lb3.place(relx=0.2, rely=0.3, width=30)
371. lb4 = Label(manage, text='科目')
372. lb4.place(relx=0.2, rely=0.4, width=30)
374. inp1 = Entry(manage)  # 班级
375. inp1.place(relx=0.3, rely=0.1, width=100)
377. inp2 = Entry(manage)  # 学号
378. inp2.place(relx=0.3, rely=0.2, width=100)
380. inp3 = Entry(manage)  # 成绩
381. inp3.place(relx=0.3, rely=0.3, width=100)
383. inp4 = Entry(manage)  # 科目
384. inp4.place(relx=0.3, rely=0.4, width=100)
386. btn1 = Button(manage, text='将该生退学', command=**lambda**: del\_stu(inp1, inp2, manage))
387. btn1.place(relx=0.3, rely=0.5, width=100)
389. btn2 = Button(manage, text='修改成绩', command=**lambda**: change\_score(inp1, inp2, inp3, inp4, manage))
390. btn2.place(relx=0.3, rely=0.6, width=100)

393. **def** admin\_control(admin):
394. quit(admin)
395. control = Tk()
396. control.title('Control System')  # 进入控制系统
397. control.geometry('400x400')
399. btn1 = Button(control, text='增加班级', command=**lambda**: append\_class(control))
400. btn1.place(relx=0.4, rely=0.3, relwidth=0.15, relheight=0.1)
402. btn2 = Button(control, text='管理学生成绩', command=**lambda**: manage\_class(control))
403. btn2.place(relx=0.35, rely=0.4, relwidth=0.25, relheight=0.1)
405. btn3 = Button(control, text='返回', command=**lambda**: go\_main(control))
406. btn3.place(relx=0.35,rely=0.7, relwidth=0.25, relheight=0.1)

409. **def** check\_BEST(inp1, admin):
410. # ########################管理员密匙###############################
411. BEST\_code = 'hust\_aia'
412. # ###############################################################
413. code = str(inp1.get())
414. **if** code == BEST\_code:
415. admin\_control(admin)
416. **else**:
417. lb1 = Label(admin, text='错误')
418. lb1.place(relx=0.425, rely=0.5, relwidth=0.1, relheight=0.1)

421. **def** administrator(root):
422. admin = Tk()
423. admin.title('Administrator')
424. admin.geometry('400x400')
425. quit(root)
426. lb1 = Label(admin, text='输入管理员密匙')
427. lb1.place(relx=0.325, rely=0.2, relwidth=0.3, relheight=0.1)
429. inp1 = Entry(admin)
430. inp1.place(relx=0.35, rely=0.3, width=100)
432. btn = Button(admin, text='验证', command=**lambda**: check\_BEST(inp1, admin))
433. btn.place(relx=0.4, rely=0.4, relwidth=0.15, relheight=0.1)

436. **def** main\_sys():
437. root = Tk()
438. root.title('StuScore Manager')
439. root.geometry('400x400')
441. lb1 = Label(root, text='成绩管理系统', font=('华文新宋', 20))
442. lb1.place(relx=0.1, rely=0, relwidth=0.8, relheight=0.3)
443. lb2 = Label(root, text='账户')
444. lb2.place(relx=0.15, rely=0.25, relwidth=0.1, relheight=0.1)
445. lb3 = Label(root, text='密码')
446. lb3.place(relx=0.15, rely=0.35, relwidth=0.1, relheight=0.1)
447. lb4 = Label(root, text='designed by Zhang Haoran')
448. lb4.place(relx=0.225, rely=0.8, relwidth=0.5, relheight=0.1)
450. inp1 = Entry(root)  # 登录账户
451. inp1.place(relx=0.25, rely=0.275, width=200)
453. inp2 = Entry(root)  # 登陆密码
454. inp2.place(relx=0.25, rely=0.375, width=200)
456. btn1 = Button(root, text='登录', command=**lambda**: check\_acc\_code(inp1, inp2, root))
457. btn1.place(relx=0.4, rely=0.5, relwidth=0.15, relheight=0.1)
459. btn2 = Button(root, text='注册', command=**lambda**: login(root))  # 进行注册
460. btn2.place(relx=0.4, rely=0.6, relwidth=0.15, relheight=0.1)
462. btn3 = Button(root, text='管理员入口', command=**lambda**: administrator(root))  # 进入管理员模式
463. btn3.place(relx=0.375, rely=0.7, relwidth=0.2, relheight=0.1)
465. root.mainloop()

468. main\_sys()