课 程 设 计 报 告

**课程名称 计算机程序设计基础2**

**班 级 无14**

**学 号 2021012728**

**姓 名 毛顺宇**

**2022年7月11日**

**一、设计内容与设计要求**

**1.课程设计目的**

面向对象程序设计课程设计是集中实践性环节之一，是学习完《计算机程序设计基础2》C++面向对象程序设计课程后进行的一次全面的综合练习。**要求学生达到熟练掌握C++语言的基本知识和技能；基本掌握面向对象程序设计的思想和方法；能够利用所学的基本知识和技能，解决简单的面向对象程序设计问题，从而**提高动手编程解决实际问题的能力。**尤其重视创新思维培养。**

**2.课题题目**

1）学生成绩管理系统

**3．文档设计要求**

3.1 设计课题题目：每个同学都单独完成1道课题。后面有范题，仅供同学们参考，不列入本次课程设计的课题。

3.2 对于程设题目，按照范题的格式。**自行虚构软件需求。并按照第4点要求，编写设计文档。基本要求系统中设计的类的数目不少于4个，每个类中要有各自的属性（多于3个成员）和方法（多于3个成员函数）；需要定义一个抽象类，采用继承方式派生这些类。并设计一个多重继承的派生类。在程序设计中，引入虚函数的多态性、运算符重载等机制。**

**4. 程序设计的基本要求：**

**（1）要求利用面向对象的方法以及C++的编程思想来完成系统的设计；**

**（2）要求在设计的过程中，建立清晰的类层次；**

**（3**）根据课题完成以下主要工作：①完成系统需求分析：包括系统设计目的与意义；系统功能需求（**系统流程图）**；输入输出的要求。②完成系统总体设计：包括系统功能分析；系统功能模块划分与设计（**系统功能模块图）**。③完成系统详细设计：数据文件；类层次图；界面设计与各功能模块实现。④系统调试：调试出现的主要问题，编译语法错误及修改，重点是运行逻辑问题修改和调整。⑤使用说明书及编程体会：说明如何使用你编写的程序，详细列出每一步的操作步骤。⑥关键源程序（带注释）。

**（4）自己设计测试数据，将测试数据存在文件中，通过文件进行数据读写来获得测试结果。**

（5）按规定格式完成课程设计报告，并在网络学堂上按时提交。

（6）不得抄袭他人程序、课程设计报告，每个人应独立完成，在程序和设计报告中体现自己的个性设计。

**5、进度安排**

|  |  |  |  |
| --- | --- | --- | --- |
| 小学期  第 3 周 |  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**注： 1、一定要保留自己那个课题的完整任务书在课程设计报告里面。**

**2、“评分表”放在“附录：源程序清单”的后面。**

**学生成绩管理系统**

目 录

[1. 系统需求分析 1](#_Toc171224774)

[2. 总体设计 1](#_Toc171224775)

[3. 详细设计 1](#_Toc171224776)

[4. 系统调试 1](#_Toc171224777)

[5. 结果与使用说明书 1](#_Toc171224778)

[6. 总结 1](#_Toc171224779)

附录：源程序清单

**附录2：评分表**

# 系统需求分析

**学生成绩信息记录了以下信息：每条记录的id，学生姓名、学号、课程名称、成绩、学历、年级。试设计考勤管理系统，使之能提供以下功能：**

**1)．录入一条成绩记录: 采用csv文件的方式读取，自动赋予id，输入格式为：**

**学号 姓名 课程名称 成绩 学历 年级**

**（每行一条纪录）**

**例如：**

**2021012728 Alice 线代 88.1 本科生 大一**

**2021012728 Bob 微积分 95 本科生 大一**

**2)．修改某条记录：可以修改这条记录的任意一个部分。**

**3). 删除某条记录: 从数据文件中删除对应的记录。**

**4). 查看成绩: 可以查看所有成绩记录，也可以根据查询规则查看具体条目的记录，同时可以选择排序方式。**

**6)．系统以菜单方式工作，选择不同的功能后再具体操作。**

# 总体设计

学生成绩管理系统包含五个大的功能，分别是：**录入一条成绩记录**、**修改某条记录**、**删除某条记录、查看成绩。学生的成绩信息主要包含学号 姓名 课程名称 成绩 学历 年级。**

**录入成绩时，应根据一定的格式输入，输入结束后提示录入成功，同时写入到数据文件中。**

**修改记录时，系统会要求输入这条记录的id，如果查询到这条记录，则用户重新输入修改后的信息，系统重新写入到数据文件中；如果没有查询到，则提示未找到，同时返回上一级。**

**删除记录时，系统会要求输入这条记录的id，如果查询到这条记录，则系统在数据文件中删除；如果没有查询到，则提示未找到，同时返回上一级。**

**查看成绩时，用户先选择查看的具体内容，同时可以以参数的方式选择**数据的主顺序、次顺序、是否反序，具体内容包括全部数据、某个学号的数据、某名学生的数据等等。

学生考勤管理系统中功能模块图：

# 详细设计

存储数据格式(csv):

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| id | 学号 | 姓名 | 课程名 | 成绩 | 学历 | 年级 |
| 1 | 2021012728 | Alice | 线代 | 88.1 | 本科生 | 大一 |
| 2 | 2021012728 | Alice | 微积分 | 95 | 本科生 | 大一 |
| 3 | 2020011524 | bob | 大物 | 85 | 本科生 | 大二 |

学生成绩管理系统的类图为：



学生成绩管理系统中各功能模块的实现：

1. 查询成绩功能模块:
2. 修改成绩功能模块

3、删除成绩功能

1. **录入成绩功能模块**

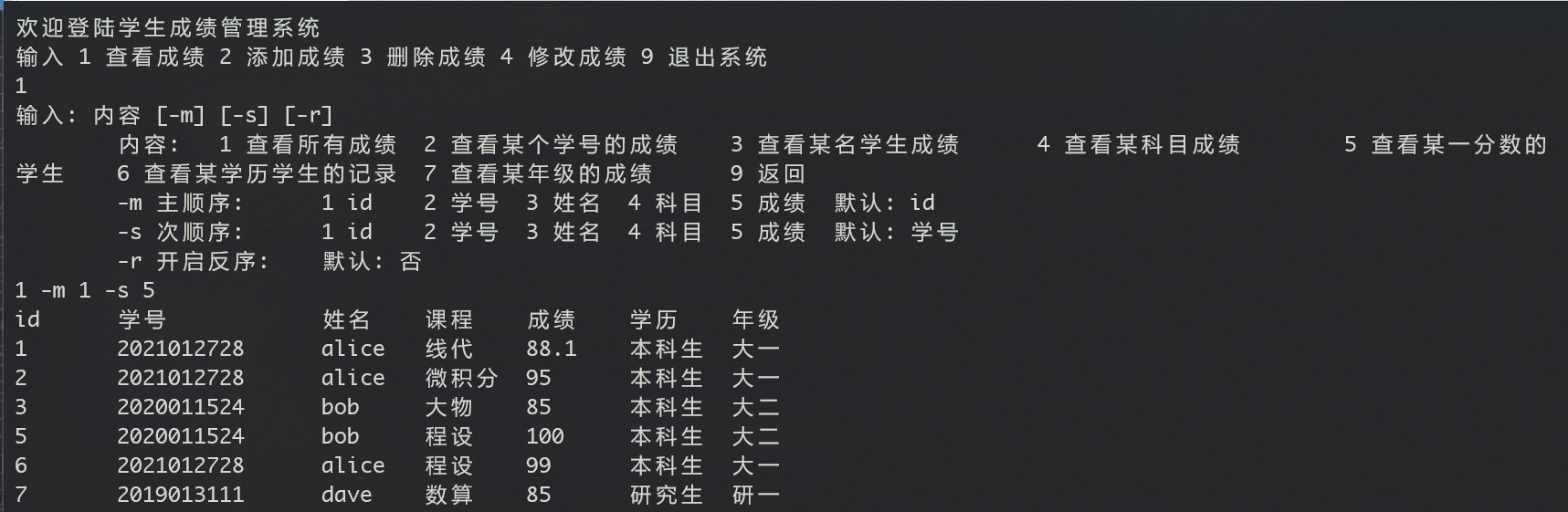
# 系统调试

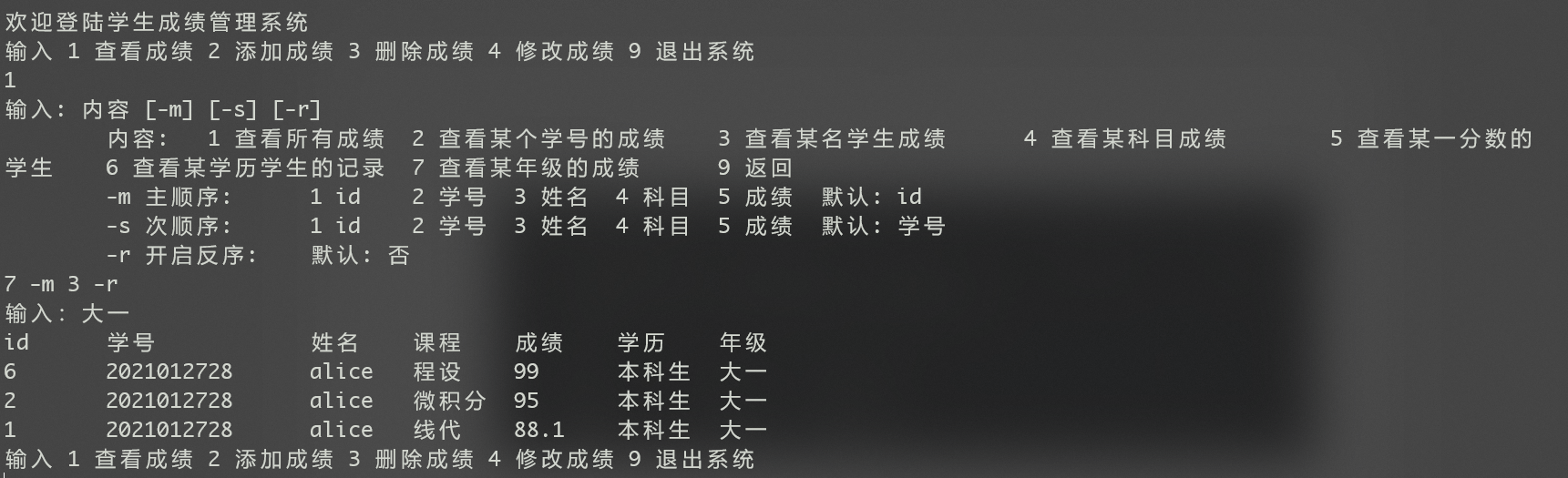
程序编写完成后，我进行了程序调试。调试过程中，出现了以下问题： 首先是为了实现排序功能，从原来的Log类中抽象出dat类，又重载了[]运算符得以获得类的某一个成员变量，原先在查看成绩时如果想要查看某一条目的成绩，代码中针对每一项都进行编写，过于复杂，现在只需根据数据类型区分。其次，在读写文件时经常出现缺少换行、多余换行的问题，仔细研究后解决，在修改、删除记录时，我选择将修改后的内容重新写入文件，但是这样需要较多IO操作，暂时没有找到好办法。另外，在查看成绩的逻辑中，原先需要依次输入内容、顺序、反序等信息，较为不必，后来修改成参数的格式，也减轻了代码量。删除记录时最初由于二重指针的存在，有一些混乱，后来解决。总之，在调试的过程中没有出现太多的bug，在代码方面进行了较大的重构。

# 结果与使用说明书

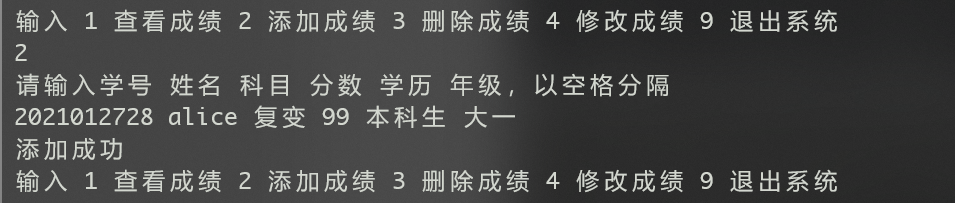
测试结果截图如图所示。

查看成绩模块：

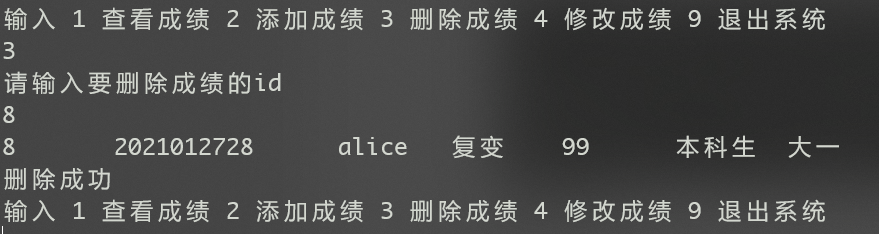




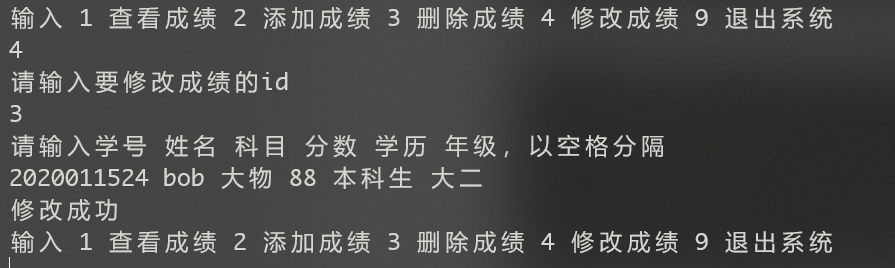
添加成绩模块:



删除成绩模块:



修改成绩模块:



# 总结

这次的大作业中，我最初对于类的划分不太明晰，后来才形成这四个类，在文件读写方面遇到了一点小困难。最复杂同时也是重构最多的是查询成绩模块，最初的版本代码比较冗杂，后来采用std::variant使得代码更加简洁，再后来改用参数的形式更加方便使用，总而言之，这次的大作业在编写过程中没有遇到太多问题，相比于上学期我认为有不少进步。

附录：源程序清单

1. /\*
2. \* @Descripttion: 程设大作业
3. \* @version: 1.0
4. \* @Author: Mao Shunyu
5. \* @Date: 2022-06-27 16:21:08
6. \* @LastEditors: Do not edit
7. \* @LastEditTime: 2022-07-13 12:23:03
8. \*/
9. #include <algorithm>
10. #include <fstream>
11. #include <iostream>
12. #include <map>
13. #include <sstream>
14. #include <variant>
15. #include <vector>
16. **using** **namespace** std;
18. /\*-------------------------------------------变量-------------------------------------------\*/
19. **class** Log;
20. vector<Log\*> logs;
21. /\*-------------------------------------------声明-------------------------------------------\*/
22. **void** view();
23. **void** add();
24. **void** del();
25. **void** change();
26. **void** display(**int** choose, **int** main\_index, **int** sub\_index, **bool** is\_reverse);
27. /\*-------------------------------------------类-------------------------------------------\*/
28. **class** dat {
29. **public**:
30. dat(string& student\_id, string& name, string& course, **double** score)
31. : student\_id(student\_id),
32. name(name),
33. course(course),
34. score(score){};  //储存基本信息
35. **int** id;
36. string student\_id;
37. string name;
38. string course;
39. **double** score;
40. **virtual** variant<string, **double**> operator[](**int** i) {
41. **switch** (i) {
42. **case** 1: **return** id;
43. **case** 2: **return** student\_id;
44. **case** 3: **return** name;
45. **case** 4: **return** course;
46. **case** 5: **return** score;
47. **default**: **return** -1;
48. }
49. }
50. };
52. **class** Log : **public** dat {
53. **private**:
54. **static** **int** max\_id;
56. **public**:
57. Log(string& student\_id, string& name, string& course, **double** score,
58. **int** id = 0);
59. **virtual** ~Log();
60. **virtual** **void** print(ostream& out) = 0;
61. **virtual** **void** write(ofstream& out) = 0;
62. **virtual** **void** set(string& student\_id, string& name, string& course,
63. **double** score) {
64. **this**->student\_id = student\_id;
65. **this**->name = name;
66. **this**->course = course;
67. **this**->score = score;
68. };
69. **virtual** **void** set\_grade(string& grade) = 0;
70. **static** **void** get\_max\_id(**int** except = -1);
71. };
73. **class** Undergraduate\_Log : **virtual** **public** Log {
74. **public**:
75. string grade;
76. Undergraduate\_Log(string& student\_id, string& name, string& course,
77. **double** score, string& grade, **int** id = 0)
78. : Log(student\_id, name, course, score, id), grade(grade){};
79. /\*\*
80. \* @msg: 打印到屏幕
81. \* @param {ostream&} out 输出
82. \* @return {\*}
83. \*/
84. **virtual** **void** print(ostream& out) {
85. out << id << "\t" << student\_id << "\t" << name << "\t" << course << "\t"
86. << score << "\t"
87. << "本科生"
88. << "\t" << grade;
89. }
91. /\*\*
92. \* @msg: 输出到文件
93. \* @param {ofstream&} out 输出
94. \* @return {\*}
95. \*/
96. **virtual** **void** write(ofstream& out) {
97. out << id << "," << student\_id << "," << name << "," << course << ","
98. << score << ","
99. << "本科生"
100. << "," << grade << endl;
101. }
102. **void** set\_grade(string& grade) { **this**->grade = grade; }
103. variant<string, **double**> operator[](**int** i) {
104. **if** (i == 6) **return** “本科生”;
105. else if(i==7) return grade;
106. **else** **return** Log::operator[](i);
107. }
108. };
109. **class** Graduate\_Log : **virtual** **public** Log {
110. **public**:
111. string grade;
112. Graduate\_Log(string& student\_id, string& name, string& course, **double** score,
113. string& grade, **int** id = 0)
114. : Log(student\_id, name, course, score, id), grade(grade){};
116. **virtual** **void** print(ostream& out) {
117. out << id << "\t" << student\_id << "\t" << name << "\t" << course << "\t"
118. << score << "\t"
119. << "研究生"
120. << "\t" << grade;
121. }
122. **virtual** **void** write(ofstream& out) {
123. out << id << "," << student\_id << "," << name << "," << course << ","
124. << score << ","
125. << "研究生"
126. << "," << grade << endl;
127. }
128. **void** set\_grade(string& grade) { **this**->grade = grade; }
129. variant<string, **double**> operator[](**int** i) {
130. **if** (i == 6) **return** “研究生”;
131. else if(i==7) return grade;
132. **else** **return** Log::operator[](i);
133. }
134. };
136. Log::Log(string& student\_id, string& name, string& course, **double** score, **int** id)
137. : dat(student\_id, name, course, score) {
138. get\_max\_id();
139. //设定新纪录的id
140. **if** (id == 0) {
141. **this**->id = max\_id + 1;
142. Log::max\_id = **this**->id;
143. } **else** **this**->id = id;
144. }
146. Log::~Log() { get\_max\_id(**this**->id); }
148. **int** Log::max\_id = 0;
150. /\*-------------------------------------------方法-------------------------------------------\*/
151. vector<string> stringSplit(**const** string& str, **char** delim) {
152. stringstream ss(str);
153. string item;
154. vector<std::string> elems;
155. **while** (getline(ss, item, delim)) {
156. **if** (!item.empty()) { elems.push\_back(item); }
157. }
158. **return** elems;
159. }
160. /\*\*
161. \* @msg: 更新所有记录中的最大id
162. \* @param {int} except 忽略的元素
163. \* @return {\*}
164. \*/
165. **void** Log::get\_max\_id(**int** except) {
166. **if** (logs.size() == 0) {
167. max\_id = 0;
168. } **else** {
169. **int** max = 0;
170. **for** (**int** i = 0; i < logs.size(); i++) {
171. **if** (logs[i]->id != except && logs[i]->id > max) {
172. max = logs[i]->id;
173. max = logs[i]->id;
174. }
175. }
176. max\_id = max;
177. }
178. }
179. /\*-----------------------------------------------------------------------\*/
180. **void** display(**int** choose, **int** main\_index, **int** sub\_index, **bool** is\_reverse) {
181. vector<**int**> indexes;
182. //选择要查看的条目
183. **if** (choose == 9) **return**;
184. **else** **if** (choose == 1)
185. **for** (**int** i = 0; i < logs.size(); i++) indexes.push\_back(i);
186. **else** {
187. string t;
188. variant<string, **double**> temp;
189. cout << "输入：";
190. cin >> t;
191. **if** (choose == 5) temp = atof(t.c\_str());
192. **else** temp = t;
193. **for** (**int** i = 0; i < logs.size(); i++) {
194. //查找对应的条目
195. **if** ((choose != 5 && (\*logs[i])[choose] == temp) ||
196. (choose == 5 &&
197. abs(get<**double**>(temp) - get<**double**>((\*logs[i])[choose])) <= 1.0e-4))
198. indexes.push\_back(i);
199. }
200. }
202. //排序
203. sort(indexes.begin(), indexes.end(), [=](**int** a, **int** b) {
204. **return** (((\*logs[a])[main\_index] == (\*logs[b])[main\_index])
205. ? (\*logs[a])[sub\_index] < (\*logs[b])[sub\_index]
206. : (\*logs[a])[main\_index] < (\*logs[b])[main\_index]);
207. });
208. **if** (is\_reverse) reverse(indexes.begin(), indexes.end());
210. cout << "id\t学号\t\t姓名\t课程\t成绩\t学历\t年级" << endl;
211. **for** (auto&& i : indexes) {
212. logs[i]->print(cout);
213. cout << endl;
214. }
215. }
217. **void** view() {
218. cout << "输入: 内容 [-m] [-s] [-r]" << endl;
219. cout << "\t内容: \t1 查看所有成绩 \t2 查看某个学号的成绩 \t3 "
220. "查看某名学生成绩 \t4 查看某科目成绩 \t5 查看某一分数的学生 \t6 "
221. "查看某学历学生的记录 \t7 查看某年级的成绩 \t9 返回"
222. << endl;
223. cout << "\t-m 主顺序: \t1 id \t2 学号 \t3 姓名 \t4 科目 \t5 成绩 \t默认: id"
224. << endl;
225. cout << "\t-s 次顺序: \t1 id \t2 学号 \t3 姓名 \t4 科目 \t5 成绩 \t默认: 学号"
226. << endl;
227. cout << "\t-r 开启反序: \t默认: 否" << endl;
228. // example:    1 -m 1 -s 2 -r
229. string line;
230. getchar();
231. getline(cin, line);
232. vector<string> v = stringSplit(line, ' ');
233. map<string, **int**> m;  //记录每个参数的位置
234. **for** (**int** i = 0; i < v.size(); i++) {
235. **if** (v.at(i)[0] == '-') { m[v.at(i).substr(1, v.at(i).size() - 1)] = i; }
236. }
237. //遍历map
238. **int** main\_index = 1, sub\_index = 2;
239. **bool** is\_reverse = **false**;
240. **for** (auto&& arg : m) {
241. **const** string a = arg.first;
242. **if** (a == "m") {
243. main\_index = stoi(v.at(arg.second + 1));
244. } **else** **if** (a == "s") {
245. sub\_index = stoi(v.at(arg.second + 1));
246. } **else** **if** (a == "r") {
247. is\_reverse = **true**;
248. } **else** {
249. cout << "输入错误！" << endl;
250. **return**;
251. }
252. }
253. display(stoi(v.at(0)), main\_index, sub\_index, is\_reverse);
254. }
255. /\*-----------------------------------------------------------------------\*/
256. **void** add() {
257. cout << "请输入学号 姓名 科目 分数 学历 年级，以空格分隔" << endl;
258. string a, b, c;
259. **double** d;
260. string e, f;
261. cin >> a >> b >> c >> d >> e >> f;
262. ofstream file("data.csv", ios::app);
263. Log\* log;
264. **if** (e == "本科生") log = **new** Undergraduate\_Log(a, b, c, d, f);
265. **else** **if** (e == "研究生") log = **new** Graduate\_Log(a, b, c, d, f);
266. logs.push\_back(log);
267. log->write(file);
268. cout << "添加成功" << endl;
269. file.close();
270. }
271. /\*-----------------------------------------------------------------------\*/
272. **void** del() {
273. cout << "请输入要删除成绩的id" << endl;
274. **int** id;
275. cin >> id;
276. **bool** have\_found = **false**;
277. **for** (auto it = logs.begin(); it != logs.end(); it++) {
278. **if** ((\*it)->id == id) {
279. have\_found = **true**;
280. (\*it)->print(cout);
281. cout << endl;
282. **delete** \*it;
283. \*it = nullptr;
284. logs.erase(it);  //删除这条记录
285. **break**;
286. }
287. }
288. **if** (have\_found) {
289. ofstream file("data.csv", ios::out);
290. **for** (**int** i = 0; i < logs.size(); i++) logs[i]->write(file);  //重新写入
291. cout << "删除成功" << endl;
292. file.close();
293. } **else** cout << "未找到该成绩记录!" << endl;
294. }
295. /\*-----------------------------------------------------------------------\*/
296. **void** change() {
297. cout << "请输入要修改成绩的id" << endl;
298. **int** id;
299. cin >> id;
300. **bool** have\_found = **false**;
301. **for** (auto it = logs.begin(); it != logs.end(); it++) {
302. **if** ((\*it)->id == id) {
303. have\_found = **true**;
304. cout << "请输入学号 姓名 科目 分数 学历 年级，以空格分隔" << endl;
305. string a, b, c;
306. **double** d;
307. string e, f;
308. cin >> a >> b >> c >> d >> e >> f;
309. (\*it)->set(a, b, c, d);
310. (\*it)->set\_grade(f);
311. **break**;
312. }
313. }
314. **if** (have\_found) {
315. ofstream file("data.csv", ios::out);
316. **for** (**int** i = 0; i < logs.size(); i++) logs[i]->write(file);
317. cout << "修改成功" << endl;
318. file.close();
319. } **else** cout << "未找到该成绩记录!" << endl;
320. }
321. /\*-----------------------------------------------------------------------\*/
322. **int** main(**int** argc, **char** **const**\* argv[]) {
323. ifstream file("data.csv");
324. **if** (!file) {
325. cout << "文件打开失败" << endl;
326. **return** 0;
327. }
328. string line;
329. //读取文件内容
330. **while** (getline(file, line)) {
331. vector<string> strs = stringSplit(line, ',');
332. **if** (!strs.empty()) {
333. Log\* log;
334. **if** (strs[5] == "本科生")
335. log = **new** Undergraduate\_Log(strs[1], strs[2], strs[3], stof(strs[4]),
336. strs[6], stoi(strs[0]));
337. **else** **if** (strs[5] == "研究生")
338. log = **new** Graduate\_Log(strs[1], strs[2], strs[3], stof(strs[4]),
339. strs[6], stoi(strs[0]));
340. logs.push\_back(log);
341. } **else** **break**;
342. }
343. file.close();
344. **int** choose;
345. cout << "欢迎登陆学生成绩管理系统" << endl;
346. cout << "输入 1 查看成绩 2 添加成绩 3 删除成绩 4 修改成绩 9 退出系统" << endl;
347. cin >> choose;
348. **while** (choose != 9) {
349. **switch** (choose) {
350. **case** 1: view(); **break**;
351. **case** 2: add(); **break**;
352. **case** 3: del(); **break**;
353. **case** 4: change(); **break**;
354. **default**: cout << "输入错误" << endl; **break**;
355. }
356. cin.sync();
357. cout << "输入 1 查看成绩 2 添加成绩 3 删除成绩 4 修改成绩 9 退出系统"
358. << endl;
359. cin >> choose;
360. }
362. **return** 0;
363. }

**附录2**：

|  |  |  |
| --- | --- | --- |
| **项 目** | **评 价** | |
| 设计方案的合理性与创新性 | **3** |  |
| 设计与调试结果 | **4** |  |
| 设计说明书的质量 | **1** |  |
| 程序基本要求涵盖情况 | **4** |  |
| 程序代码编写素养情况 | **2** |  |
| 课程设计周表现情况 | **1** |  |
| 综合成绩 | **15** |  |