|  |  |
| --- | --- |
| **编号:** | **1-1** |

****

信息科学与工程学院实验报告

《面向对象程序设计》

**Object-Oriented Programming**

|  |  |
| --- | --- |
| 姓名： | 姬彬荃 |
| 学号： | 201911020125 |
| 班级： | 计联培1901 |
| 时间： | 2020年11月17日 |

**《面向对象程序设计》实验报告**

**基本要求：**请围绕实验目的、实验内容、实验过程、实验结果（附图）、实验总结（重点阐述）五个部分进行撰写。若报告中若涉及源代码内容，请在附录部分提供完整源码及GitHub源码托管地址。报告撰写完毕后请提交PDF格式版本到云班课。

1. **实验目的**

* 熟练掌握复杂类的设计方法（三构一析+普函）
* 熟练掌握静态数据成员和函数的声明、定义及用法（static）
* 掌握常数据，常函数和常对象的声明、定义及用法（const）
* 掌握友元函数的声明、定义和使用方法（friend）
* 熟练掌握visual studio创建类的快捷方法（实战）
* 了解代码编辑器VIM，以及VIM的有关基本用法（扩展）

1. **实验内容**

【**简答题**】

* 什么是静态成员数据，其本质和主要作用是什么？
* 什么是静态成员函数，在类中为何要设计该函数？
* 静态数据成员与普通数据成员有何区别与联系？
* 类中可以有哪些类型的成员数据和成员函数，请详细列举并简要解释其含义。

【**程序设计题**】请采用C++面向对象程序设计思想设计类，并实现类。 要求程序代码: 清晰简洁易读，良好的开闭性， 编程风格规范统一。

设计某高校研究生 Graduate 类, 该类包含研究生姓名（string name)、性别 (bool gender)、学号 (int id)、入学成绩（double score）、研究领域 (string research) 、指导教师 (string tutor) 和学校名称（const string university）与该研究生类有关的函数如下：

* 初始化与释放研究生对象数据（三构一析）；
* 采用普通成员函数形式输出研究生基本数据信息；
* 采用常成员函数形式输出研究生的基本数据信息；
* 输出多个研究生对象的平均成绩 (要求在类内封装实现)；
* 以友元函数形式计算每个研究生对象的成绩等级（90 分以上输出 A, 80-89 分输出 B，70-79 分输出 C, 60-69 分输出 D，60 分以下输出 E)；

**要求**：设计实现该类，并在main函数内定义研究生对象数组，然后，再单独定义普通研究生对象和常研究生对象，最后，输出各对象的详细信息输出所有研究生的平均成绩与成绩等级。

1. **实验过程**

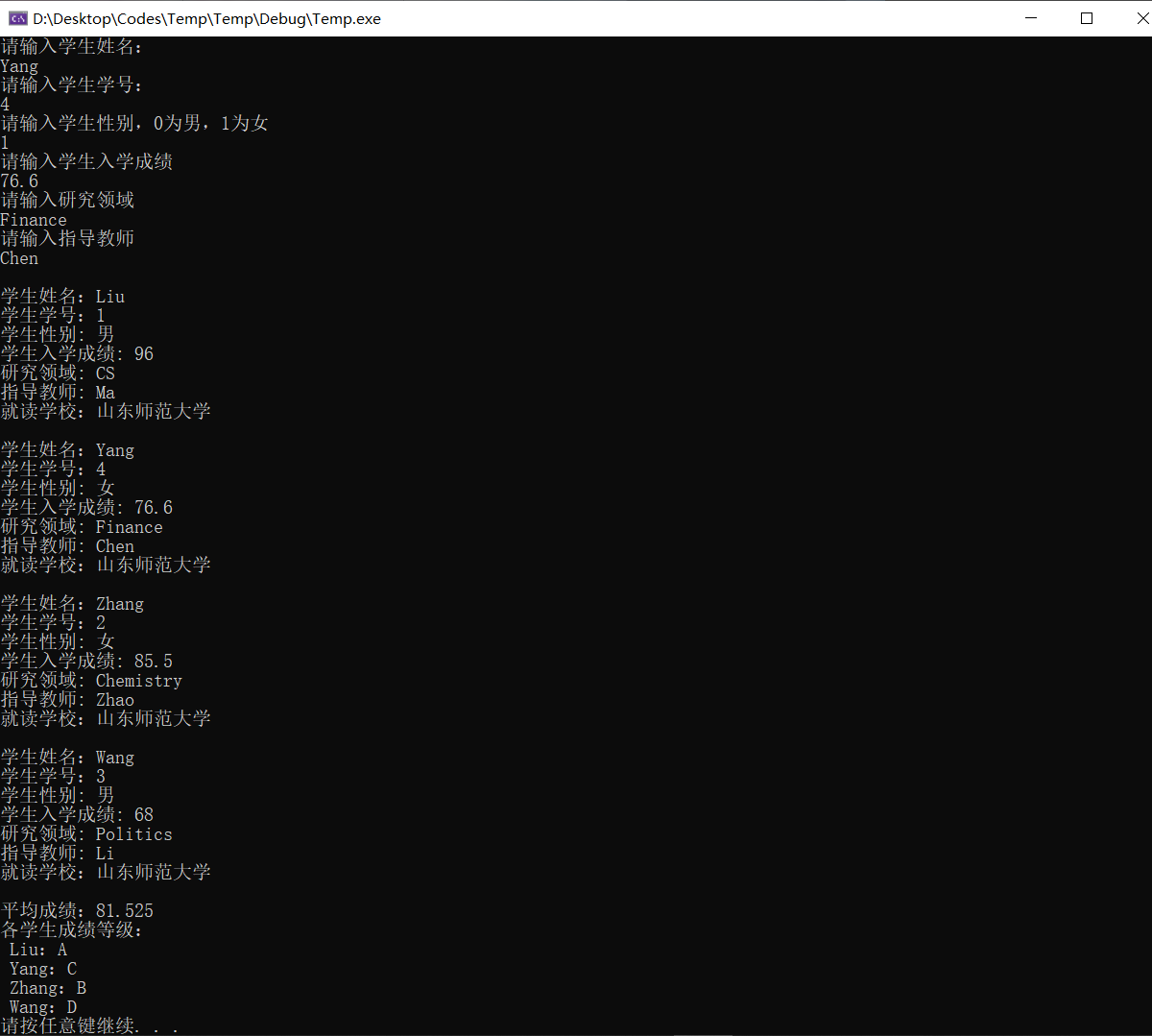
**实验班级**：19计联培 || 19计本 || 19计师本

**上机时间**： 2020年11月17日 晚 19:00 -- 20:45

**上机地点**： E315机房 (按组序入座)

**实验环境：** Visual Studio 2019

1. **实验结果**



1. **实验总结**
   1. 静态常成员函数需要在类外定义，使用语句：

const 数据类型 类名::变量名 = 值

* 1. 学会了静态成员函数、静态成员变量、常函数和常静态成员变量的使用
* **附录：程序源码（建议基于Highlight软件导入）**



1. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
2. \*\* 功能 : 第四章 类的共享和保护实验
3. \*\* 作者 : 2019Jibinquan
4. \*\* 创建 : 2020-11-17 / 19:28
5. \*\* 版权 : 431263064@qq.com
6. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/
8. #include <iostream>
9. #include <cstdio>
10. #include <cmath>
11. #include <algorithm>
12. #include <cstring>
13. #include <string>
14. #include <queue>
15. #include <stack>
16. #include <vector>
17. #include <map>
18. #include <set>
19. #define ios ios::sync\_with\_stdio(false);cin.tie(0);cout.tie(0);
20. #define debug(a) cout << #a << " " << a << endl
21. **using** **namespace** std;
22. **typedef** **long** **long** ll;
23. **const** **double** pi = acos(-1);
24. **const** **double** eps = 1e-8;
25. **const** **int** inf = 0x3f3f3f3f;
26. **const** **int** maxn = 100007;//1e5+7
27. **const** ll mod = 1000000007;//1e9+7
29. /\*----------------------------------\*
30. Class Definition
31. \*-----------------------------------\*/
33. **class** Graduate
34. {
35. **private**:
36. string name;
37. **bool** gender;
38. **int** id;
39. **double** score;
40. string research;
41. string tutor;
42. **static** **const** string university;
43. **static** **double** average;
44. **static** **int** count;
46. **public**:
47. Graduate() {
48. cout << "请输入学生姓名：" << endl;
49. cin >> name;
50. cout << "请输入学生学号：" << endl;
51. cin >> id;
52. cout << "请输入学生性别，0为男，1为女" << endl;
53. cin >> gender;
54. cout << "请输入学生入学成绩" << endl;
55. cin >> score;
56. cout << "请输入研究领域" << endl;
57. cin >> research;
58. cout << "请输入指导教师" << endl;
59. cin >> tutor;
60. **double** sum = average \* count + score;
61. count++;
62. average = sum / count;
63. }
64. Graduate(string na, **bool** sex, **int** ide, **double** sco, string resa, string tut) {
65. name = na;
66. gender = sex;
67. id = ide;
68. score = sco;
69. research = resa;
70. tutor = tut;
71. **double** sum = average \* count + score;
72. count++;
73. average = sum / count;
74. }
75. Graduate(**const** Graduate& G) {
76. name = G.name;
77. gender = G.gender;
78. id = G.id;
79. score = G.score;
80. research = G.research;
81. tutor = G.tutor;
82. **double** sum = average \* count + score;
83. count++;
84. average = sum / count;
85. }
86. ~Graduate() {}

89. **void** show() **const** {
90. cout << "学生姓名：" << name << endl;
91. cout << "学生学号：" << id << endl;
92. cout << "学生性别: " << (gender ? "女" : "男") << endl;
93. cout << "学生入学成绩: " << score << endl;
94. cout << "研究领域: " << research << endl;
95. cout << "指导教师: " << tutor << endl;
96. cout << "就读学校：" << university << endl;
97. cout << endl;
98. }
99. **void** show() {
100. **const** Graduate G(\***this**);
101. G.show();
102. }
103. string getName()**const** {
104. **return** name;
105. }
106. **static** **const** **double** ShowAverage() {
107. **return** average;
108. }
109. **friend** **char** level(**const** Graduate);
110. };
112. **char** level(**const** Graduate G) {
113. **if** (G.score >= 90) {
114. **return** 'A';
115. }
116. **else** **if** (G.score >= 80) {
117. **return** 'B';
118. }
119. **else** **if** (G.score >= 70) {
120. **return** 'C';
121. }
122. **else** **if** (G.score >= 60) {
123. **return** 'D';
124. }
125. **else** {
126. **return** 'E';
127. }
128. }
130. **int** Graduate::count = 0;
131. **double** Graduate::average = 0;
132. **const** string Graduate::university = "山东师范大学";
134. /\*----------------------------------\*
135. Main Function
136. \*-----------------------------------\*/
138. Graduate Gs[2] = { {"Liu",0,1,96.00,"CS","Ma"} };
140. **int** main()
141. {
142. Graduate Zhang("Zhang", 1, 2, 85.50, "Chemistry", "Zhao");
143. **const** Graduate Wang("Wang", 0, 3, 68.00, "Politics", "Li");
144. cout << endl;
145. Gs[0].show();
146. Gs[1].show();
147. Zhang.show();
148. Wang.show();
149. cout << "平均成绩：" << Graduate::ShowAverage() << endl;
150. cout << "各学生成绩等级：" << endl;
151. cout << " " << Gs[0].getName() << "：" << level(Gs[0]) << endl;
152. cout << " " << Gs[1].getName() << "：" << level(Gs[1]) << endl;
153. cout << " " << Zhang.getName() << "：" << level(Zhang) << endl;
154. cout << " " << Wang.getName() << "：" << level(Wang) << endl;
155. system("pause");
157. **return** 0;
158. }