武汉纺织大学

面向对象课程设计

**高等院校师生信息管理系统**

**学 院： 计算机与人工智能学院**

**班 级： 计算机类12208班**

**姓 名： 刘芊蕊**

**学 号： 2204240832**

**指导老师： 张自力**

**成 绩：**

**完成日期： 2023年6月29日**

目 录

[**1 需求分析** 1](#_Toc138935604)

[**1.1主菜单** 1](#_Toc138935605)

[**1.2各项功能** 1](#_Toc138935606)

[**1.2.1 添加教师、学生信息** 1](#_Toc138935607)

[**1.3.2 修改教师、学生信息** 2](#_Toc138935608)

[**1.3.3 删除教师、学生信息** 2](#_Toc138935609)

[**1.3.4 查询教师、学生信息** 2](#_Toc138935610)

[**1.3.5 保存信息到文件** 2](#_Toc138935611)

[**1.3.6 退出系统** 2](#_Toc138935612)

[**2 系统设计** 3](#_Toc138935613)

[**2.1流程图** 3](#_Toc138935614)

[**2.2 类之间的相互关系** 3](#_Toc138935615)

[**3 系统实现（基于Linux）** 4](#_Toc138935616)

[**3.1 项目结构** 4](#_Toc138935617)

[**3.2 Makefile（基于GNU make）** 4](#_Toc138935618)

[**3.3 main.cpp** 5](#_Toc138935619)

[**3.4 class.h** 6](#_Toc138935620)

[**3.5 class.cpp** 8](#_Toc138935621)

[**3.6 func.h** 11](#_Toc138935622)

[**3.7 func.cpp** 11](#_Toc138935623)

[**3.8文本文件** 14](#_Toc138935624)

[**3.8.1 greet** 14](#_Toc138935625)

[**3.8.2 type** 14](#_Toc138935626)

[**4 系统测试** 15](#_Toc138935627)

[**4.1 Makefile** 15](#_Toc138935628)

[**4.2 主菜单** 15](#_Toc138935629)

[**4.3 添加** 16](#_Toc138935630)

[**4.4 修改** 17](#_Toc138935631)

[**4.5 删除** 17](#_Toc138935632)

[**4.6 查询** 18](#_Toc138935633)

[**4.7 保存到文件** 18](#_Toc138935634)

[**4.8 退出系统** 18](#_Toc138935635)

[**5 系统总结** 19](#_Toc138935636)

[**5.1 问题解决过程** 19](#_Toc138935637)

[**5.1.1 按id查找的实现方式** 19](#_Toc138935638)

[**5.1.2 文件写入部分的代码重用考量** 19](#_Toc138935639)

[**5.1.3 编译链接过程繁琐** 19](#_Toc138935640)

[**5.1.4 封装性不足** 19](#_Toc138935641)

[**5.1.5 基类析构函数** 19](#_Toc138935642)

[**5.1.6 内存泄露** 19](#_Toc138935643)

[**5.1.7 流程图** 19](#_Toc138935644)

[**5.2 系统存在的不足** 19](#_Toc138935645)

[**5.3 课程收获** 19](#_Toc138935646)

[**5.3.1 大型项目开发经验** 19](#_Toc138935647)

[**5.3.2 GNU make的学习** 20](#_Toc138935648)

[**5.3.3 git备份** 20](#_Toc138935649)

[**5.3.4 文件操作** 20](#_Toc138935650)

[**5.4 课程建议** 20](#_Toc138935651)

[**5.4.1 精简课设要求** 20](#_Toc138935652)

[**5.5 发展方向** 20](#_Toc138935653)

[**5.5.1 工程能力** 20](#_Toc138935654)

[**5.5.2 算法竞赛** 20](#_Toc138935655)

**1 需求分析**

设计一个高等院校师生信息管理系统，实现对教师和各类学生信息的添加、修改、删除和查找，保存到文件等功能，用户页面应尽可能友好，具体需求如下。

**1.1主菜单**

运行程序，进入系统主界面，如下：

欢迎使用大学师生管理系统！系统旨在为您提供方便快捷的教师和学生信息管理服务。

----------------------------------------

您可以通过以下操作实现对教师和学生信息的管理：

【1】添加教师和学生信息

【2】修改教师和学生信息

【3】删除教师和学生信息

【4】查找教师和学生信息

【5】保存信息到文件

【0】退出系统

请根据您的需要输入相应的编号，我们将竭诚为您服务。

----------------------------------------

请输入您的操作编号（如 3），并按下回车键:

每轮输入对应的操作（不包括退出系统）完成后，都会在用户端刷新一遍主菜单，以防用户忘记操作编号。

**1.2各项功能**

**1.2.1 添加教师、学生信息**

当选择1时，进入二级菜单（需要添加的学生或教师的类别）

请选择您要添加的学生/教师类型:

【1】UnderGraduate（本科生）

【2】Graduate（研究生）

【3】Teacher（教师）

【4】TA（教师研究生）

【0】exit（退出）

**1.3.2 修改教师、学生信息**

当选择2时，程序会提示用户输入想要修改信息的学生/教师的ID，用户可以根据ID查找到对应的学生或教师信息，若id存在，则下一行开始重新输入该教师/学生信息。否则，显示“id不存在。”

提示：系统当前仅支持通过id（身份证号）查找

请输入id：

修改成功后，输出“信息已更新。”。

**1.3.3 删除教师、学生信息**

当选择3时，删除学生/教师信息。当用户选择该功能时，程序会提示用户输入想要删除信息的学生/教师的ID，用户可以根据ID查找到对应的学生或教师信息，若id存在，则显示

确定要删除吗？(y/n)

用户输入y后，显示“信息已删除。”，否则，显示“id不存在。”。

**1.3.4 查询教师、学生信息**

当选择4时，查找学生/教师信息。当用户选择该功能时，程序会提示用户输入想要查找信息的学生/教师的ID，用户可以根据ID查找到对应的学生或教师信息，并打印出该信息。以下是一个示例：

提示：系统当前仅支持通过id（身份证号）查找

请输入id：2204240832

Name: David

ID: 2204240832

Gender: M

Birthday: 2000-10-1

Student No: 32

School Name: WTU

Class: 01

Direction: CS

Adviser Name: Zhang

**1.3.5 保存信息到文件**

当选择5时，将信息写入文件。当用户选择该功能时，程序会将信息列表中的所有对象信息写入文本文件“information”中。

**1.3.6 退出系统**

选择0时，显示“确定要退出系统吗？(y/n):”,输入y，显示“感谢您的使用，下次再见！”，退出程序。否则，返回主菜单。

**2 系统设计**

**2.1流程图**

系统单次操作的示意图如下图2-1所示：

图2-1 单次操作示意图

**2.2 类之间的相互关系**

对于时间的存储和教师/学生的信息模块，共设计如下7个类。

* Date类：用于存储日期信息，被Person类和其子类Student、Teacher、UnderGraduate、Graduate使用。
* Person类：表示人员信息，拥有姓名、身份证号、性别和生日等属性，包含了输入和展示人员信息的方法，是Student、Teacher、UnderGraduate、Graduate的父类。
* Student类：继承自Person类，表示学生信息，拥有学号、所在学校、班级等属性，包含了输入和展示学生信息的方法。
* Teacher类：继承自Person类，表示教师信息，拥有教师号、所在学校、所在部门等属性，包含了输入和展示教师信息的方法。
* UnderGraduate类：继承自Student类，表示本科生信息，拥有专业等属性，包含了输入和展示本科生信息的方法。
* Graduate类：继承自Student类，表示研究生信息，拥有研究方向和导师姓名等属性，包含了输入和展示研究生信息的方法。
* TA类：继承自Graduate和Teacher类，表示研究生助教信息，拥有教师号和所在部门等属性，包含了输入和展示研究生助教信息的方法。

各教师/类的结构及类之间的关系如图2-2所示：

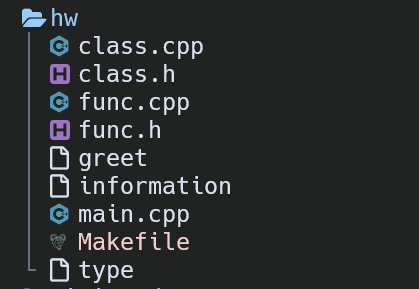
­­

图2-2 用户功能类图

**3 系统实现（基于Linux）**

该项目代码细化过程（包括主函数/其他函数的声明和实现，添加注释，Makefile文件等）全程经由git本地备份。

**3.1 项目结构**



**3.2 Makefile（基于GNU make）**

为了更好地管理项目，在项目根目录下新建Makefile文件，根据各文件间的依赖关系，将clean设为伪目标，编译器为g++，可执行文件命名为my\_hw，代码如下。（省略可自动推出的依赖关系）

1. obj = main.o class.o func.o
2. my\_hw: $(obj)
3. g++ -o my\_hw $(obj)
4. main.o: class.h func.h
5. func.o: class.h
6. .PHONY: clean
7. clean:
8. -rm my\_hw $(obj)

**3.3 main.cpp**

项目的主函数文件，主要作用是实现用户与系统之间的交互，通过调用class.h和func.h中的函数完成对学生信息的增删改查和文件的读写操作，以及释放指针申请到的空间。

1. // main.cpp
2. #include "class.h"
3. #include "func.h"
4. int main() {
5. string id;
6. int user\_num;
7. // 打印主菜单
8. print\_menu();
9. // 读取用户操作到user\_num
10. while (cin >> user\_num) {
11. // 添加
12. if (user\_num == 1) {
13. // 打印二级菜单
14. print\_add\_tip();
15. // 读入并存储教师/学生信息
16. add\_info();
17. }
18. // 修改
19. else if (user\_num == 2) {
20. // 读取id
21. read\_id\_from\_user(id);
22. // 修改id对应的信息条目
23. modify\_info(id);
24. }
25. // 删除
26. else if (user\_num == 3) {
27. // 读取id
28. read\_id\_from\_user(id);
29. // 删除id对应的信息条目
30. del\_info(id);
31. }
32. // 查找
33. else if (user\_num == 4) {
34. // 读取id
35. read\_id\_from\_user(id);
36. // 查询id对应的信息条目，并打印
37. search\_info(id);
38. }
39. // 写入文件
40. else if (user\_num == 5) {
41. write\_to\_file();
42. }
43. // 退出系统
44. else if (user\_num == 0) {
45. // 确认
46. if (exit\_comfirm())
47. break;
48. }
49. // 打印主菜单
50. print\_menu();
51. }
52. // 释放指针指向的空间。
53. for (auto [x, y] : table)
54. delete y;
55. return 0;
56. }

**3.4 class.h**

该文件的主要作用是定义了用于存储和操作学生信息的类，声明外部对象哈希表来建立id和对象指针的映射，为其他文件提供了必要的类定义和头文件包含。

1. // class.h
2. #ifndef CLASS\_H
3. #define CLASS\_H
4. #include <cstdio>
5. #include <fstream>
6. #include <iostream>
7. #include <unordered\_map>
8. using namespace std;
9. // Date类
10. class Date {
11. public:
12. Date(int y = 0, int m = 0, int d = 0) : year(y), month(m), day(d) {}
13. // 重载输出运算符
14. friend ostream &operator<<(ostream &os, Date const &birthday);
15. int year, month, day;
16. };
17. // Person类
18. class Person {
19. protected:
20. string name, id, gender;
21. Date birthday;
22. public:
23. Person(string n = "", string i = "", string g = "", Date b = Date())
24. : name(n), id(i), gender(g), birthday(b) {}
25. // 析构函数设为虚函数
26. virtual ~Person(){};
27. // 输入信息
28. virtual void inputData();
29. // 打印信息
30. virtual void displayDetails();
31. };
32. // 定义哈希表，将id映射到基类对象指针，便于查询
33. extern unordered\_map<string, Person \*> table;
34. // Student类
35. class Student : public virtual Person {
36. protected:
37. string studentNo, schoolName, classIn;
38. public:
39. Student(string n = "", string i = "", string g = "", Date b = Date(),
40. string sNo = "", string sName = "", string cIn = "")
41. : Person(n, i, g, b), studentNo(sNo), schoolName(sName), classIn(cIn) {}
42. void inputData();
43. void displayDetails();
44. };
45. // Teacher类
46. class Teacher : public virtual Person {
47. protected:
48. string teacherNo, schoolName, department;
49. public:
50. Teacher(string n = "", string i = "", string g = "", Date b = Date(),
51. string tNo = "", string sName = "", string dept = "")
52. : Person(n, i, g, b), teacherNo(tNo), schoolName(sName),
53. department(dept) {}
54. void inputData();
55. void displayDetails();
56. };
57. // UnderGraduate类
58. class UnderGraduate : public Student {
59. private:
60. string major;
61. public:
62. UnderGraduate(string n = "", string i = "", string g = "", Date b = Date(),
63. string sNo = "", string sName = "", string cIn = "",
64. string m = "")
65. : Person(n, i, g, b), Student(n, i, g, b, sNo, sName, cIn), major(m) {}
66. void inputData();
67. void displayDetails();
68. };
69. // Graduate类
70. class Graduate : public Student {
71. protected:
72. string direction, adviserName;
73. public:
74. Graduate(string n = "", string i = "", string g = "", Date b = Date(),
75. string sNo = "", string sName = "", string cIn = "", string d = "",
76. string aName = "")
77. : Person(n, i, g, b), Student(n, i, g, b, sNo, sName, cIn),
78. direction(d), adviserName(aName) {}
79. void inputData();
80. void displayDetails();
81. };
82. // TA类
83. class TA : public Graduate, public Teacher {
84. public:
85. TA(string n = "", string i = "", string g = "", Date b = Date(),
86. string sNo = "", string sName = "", string cIn = "", string d = "",
87. string aName = "", string tNo = "", string dept = "")
88. : Person(n, i, g, b), Graduate(n, i, g, b, sNo, sName, cIn, d, aName),
89. Teacher(n, i, g, b, tNo, sName, dept) {}
90. void inputData();
91. void displayDetails();
92. };
93. #endif

**3.5 class.cpp**

该文件实现了头文件" class.h"中定义的各个类的成员函数和全局变量。

1. 实现了哈希表table的定义，该哈希表将学生的身份证号映射到Person类的对象指针，便于查询和操作学生信息。
2. 实现了Date类中输出运算符<<的重载，用于将日期信息输出到控制台。
3. 实现了各个类的inputData()和displayDetails()成员函数，用于输入和展示人员信息；
4. // class.cpp
5. #include "class.h"
6. // 定义哈希表
7. unordered\_map<string, Person \*> table;
8. // 实现重载
9. ostream &operator<<(ostream &os, Date const &birthday) {
10. os << birthday.year << "-" << birthday.month << "-" << birthday.day;
11. return os;
12. }
13. // 成员函数实现
14. void Person::inputData() {
15. cout << "Name: ";
16. cin >> name;
17. cout << "ID: ";
18. cin >> id;
19. table[id] = this;
20. cout << "Gender: ";
21. cin >> gender;
22. cout << "Birthday (Year Month Day): ";
23. cin >> birthday.year >> birthday.month >> birthday.day;
24. }
25. void Person::displayDetails() {
26. cout << endl;
27. cout << "Name: " << name << endl;
28. cout << "ID: " << id << endl;
29. cout << "Gender: " << gender << endl;
30. cout << "Birthday: " << birthday << endl;
31. }
32. void Student::inputData() {
33. Person::inputData();
34. cout << "Student No: ";
35. cin >> studentNo;
36. cout << "School Name: ";
37. cin >> schoolName;
38. cout << "Class: ";
39. cin >> classIn;
40. }
41. void Student::displayDetails() {
42. Person::displayDetails();
43. cout << "Student No: " << studentNo << endl;
44. cout << "School Name: " << schoolName << endl;
45. cout << "Class: " << classIn << endl;
46. }
47. void Teacher::inputData() {
48. Person::inputData();
49. cout << "Teacher No: ";
50. cin >> teacherNo;
51. cout << "School Name: ";
52. cin >> schoolName;
53. cout << "Department: ";
54. cin >> department;
55. }
56. void Teacher::displayDetails() {
57. Person::displayDetails();
58. cout << "Teacher No: " << teacherNo << endl;
59. cout << "School Name: " << schoolName << endl;
60. cout << "Department: " << department << endl;
61. }
62. void UnderGraduate::inputData() {
63. Student::inputData();
64. cout << "Major: ";
65. cin >> major;
66. }
67. void UnderGraduate::displayDetails() {
68. Student::displayDetails();
69. cout << "Major: " << major << endl;
70. }
71. void Graduate::inputData() {
72. Student::inputData();
73. cout << "Direction: ";
74. cin >> direction;
75. cout << "Adviser Name: ";
76. cin >> adviserName;
77. }
78. void Graduate::displayDetails() {
79. Student::displayDetails();
80. cout << "Direction: " << direction << endl;
81. cout << "Adviser Name: " << adviserName << endl;
82. }
83. void TA::inputData() {
84. Graduate::inputData();
85. cout << "Teacher No: ";
86. cin >> teacherNo;
87. cout << "Department: ";
88. cin >> department;
89. }
90. void TA::displayDetails() {
91. Graduate::displayDetails();
92. cout << "Teacher No: " << teacherNo << endl;
93. cout << "Department: " << department << endl;
94. }

**3.6 func.h**

该文件声明了学生信息的增、删、改、查、写入文件以及程序的退出确认等功能的函数。

1. // func.h
2. #include "class.h"
3. #ifndef FUNC\_H
4. #define FUNC\_H
5. // 从键盘读入用户输入的id
6. void read\_id\_from\_user(string &id);
7. // 打印文件的全部内容
8. void print\_file(string const &file\_name);
9. // 打印主菜单
10. void print\_menu();
11. // 打印添加信息前的提示文字
12. void print\_add\_tip();
13. // 添加
14. void add\_info();
15. // 修改
16. void modify\_info(string const &info\_id);
17. // 删除
18. void del\_info(string const &info\_id);
19. // 查询
20. void search\_info(string const &info\_id);
21. // 写入文件
22. void write\_to\_file();
23. // 退出确认
24. bool exit\_comfirm();
25. #endif

**3.7 func.cpp**

实现了func.h中的各个函数，包括学生信息的增、删、改、查、写入文件以及程序的退出确认等。

1. #include "func.h"
2. #include "class.h"
3. void read\_id\_from\_user(string &id) {
4. cout << "\n提示：系统当前仅支持通过id（身份证号）查找\n";
5. cout << "请输入id：";
6. cin >> id;
7. }
8. void print\_file(string const &file\_name) {
9. string s;
10. ifstream file(file\_name);
11. // 逐行读入文件，并输出到标准输入流
12. while (getline(file, s)) {
13. cout << s << endl;
14. }
15. file.close();
16. }
17. void print\_menu() {
18. // 打印主菜单
19. print\_file("greet");
20. cout << "请输入您的操作编号（如 3），并按下回车键: ";
21. }
22. void print\_add\_tip() {
23. cout << "\n请选择您要添加的学生/教师类型:\n\n";
24. print\_file("type");
25. }
26. void add\_info() {
27. int x;
28. // 定义一个空的基类指针
29. Person \*ps = nullptr;
30. cin >> x;
31. // 根据读入的数字new对应类的对象
32. if (x == 1)
33. ps = (new UnderGraduate);
34. else if (x == 2)
35. ps = (new Graduate);
36. else if (x == 3)
37. ps = (new Teacher);
38. else if (x == 4)
39. ps = (new TA);
40. // 若new成功，则调用输入函数，从键盘读入教师/学生的信息
41. // 体现了运行时的多态性
42. if (ps != nullptr) {
43. ps->inputData();
44. cout << "\n信息已更新。\n\n";
45. }
46. }
47. void modify\_info(string const &info\_id) {
48. // 用哈希表判断id是否存在
49. // 其他函数的相同操作不赘述
50. if (table.count(info\_id)) {
51. table[info\_id]->inputData();
52. cout << "\n信息已更新。\n\n";
53. } else {
54. cout << "\nid不存在。\n\n";
55. }
56. }
57. void del\_info(string const &info\_id) {
58. if (table.count(info\_id)) {
59. string tmp;
60. cout << "确定要删除吗？(y/n): ";
61. cin >> tmp;
62. if (tolower(tmp[0]) == 'y') {
63. table.erase(info\_id);
64. }
65. cout << "\n信息已删除。\n\n";
66. } else {
67. cout << "\nid不存在。\n\n";
68. }
69. }
70. void search\_info(string const &info\_id) {
71. if (table.count(info\_id)) {
72. table[info\_id]->displayDetails();
73. cout << endl;
74. } else {
75. cout << "\nid不存在。\n\n";
76. }
77. }
78. void write\_to\_file() {
79. // 将输出重定向到文件information
80. freopen("information", "w", stdout);
81. // 遍历哈希表，打印每个信息条目
82. for (auto [x, y] : table) {
83. y->displayDetails();
84. }
85. // 将输出重定向到标准输出流
86. // Windows系统为"CON"
87. freopen("/dev/tty", "w", stdout);
88. cout << "\n信息已保存。\n\n";
89. }
90. bool exit\_comfirm() {
91. string tmp;
92. cout << "确定要退出系统吗？(y/n): ";
93. cin >> tmp;
94. if (tolower(tmp[0]) == 'y') {
95. cout << "\n感谢您的使用，下次再见！\n\n";
96. return true;
97. }
98. return false;
99. }

**3.8文本文件**

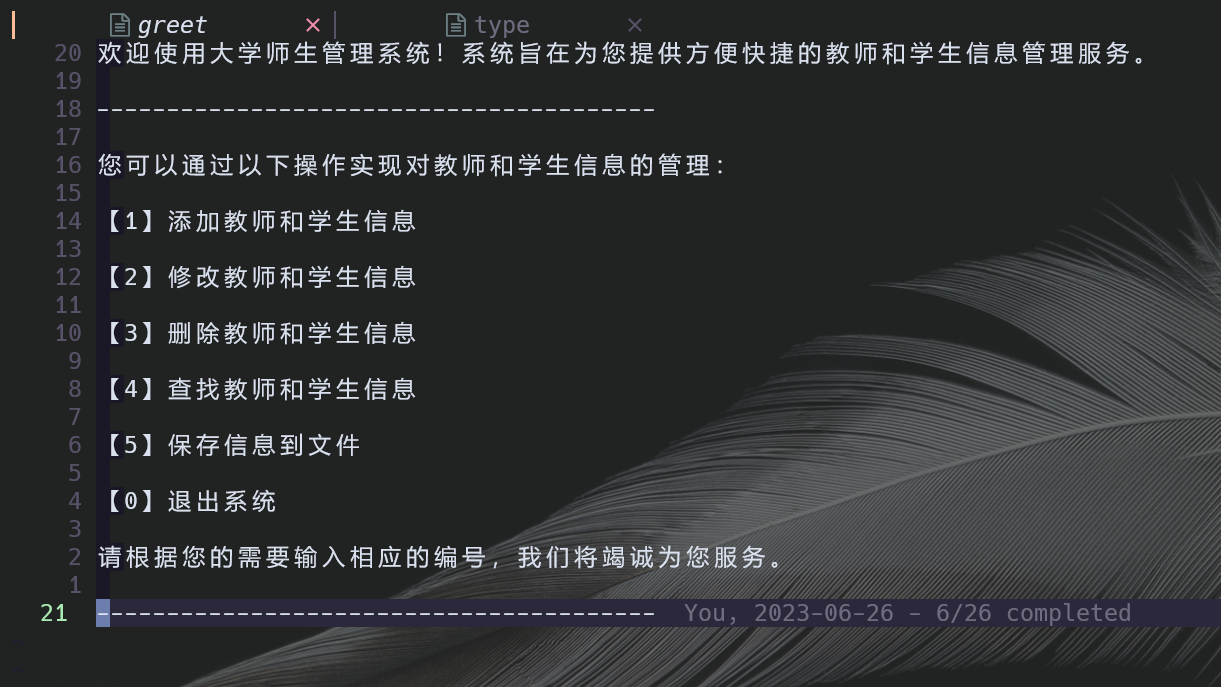
（information为写入文件，不作介绍）

项目用文本文件存储提示语，以节省代码长度。

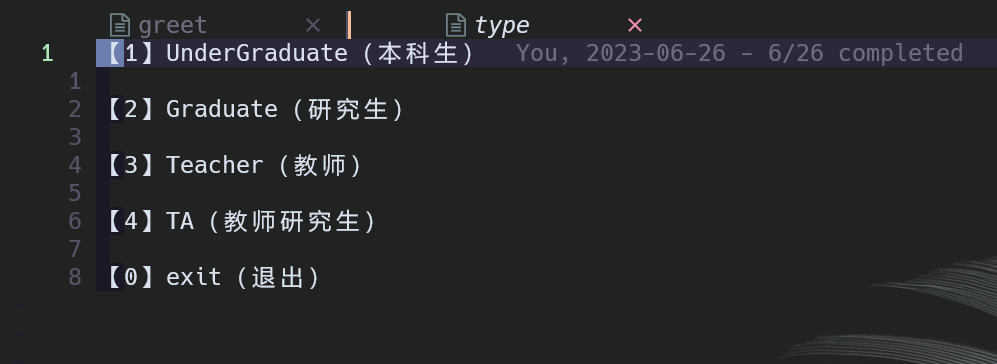
**3.8.1 greet**

（光标所在行的暗淡文字为编辑器内集成git显示的所属分支，非文件内容）

内容为问候语和主菜单。

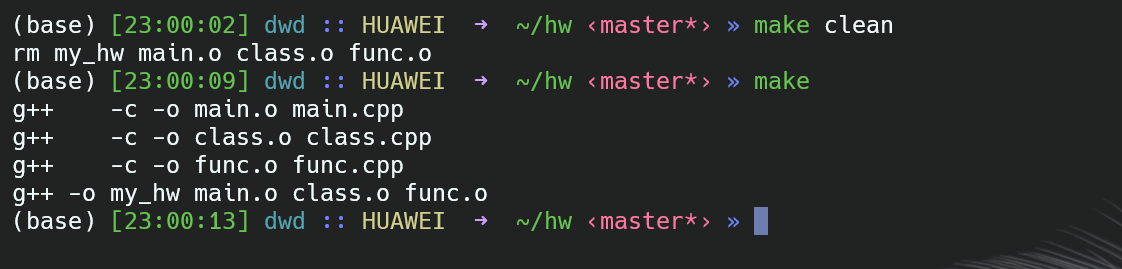


**3.8.2 type**

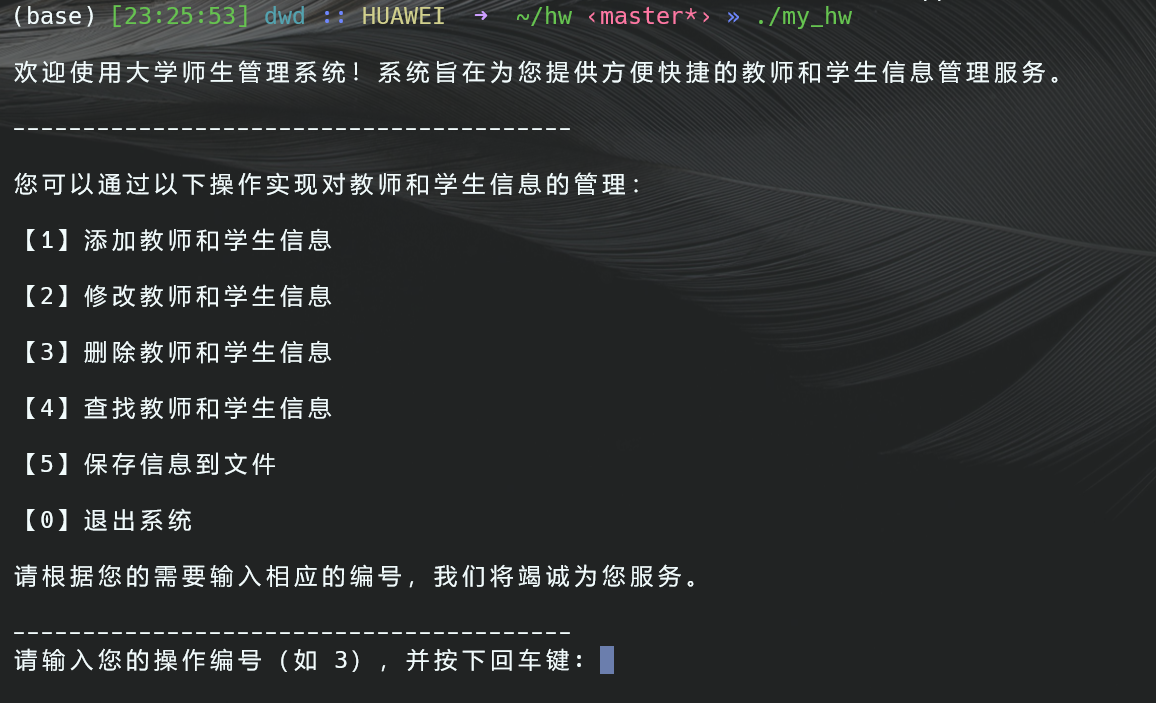


**4 系统测试**

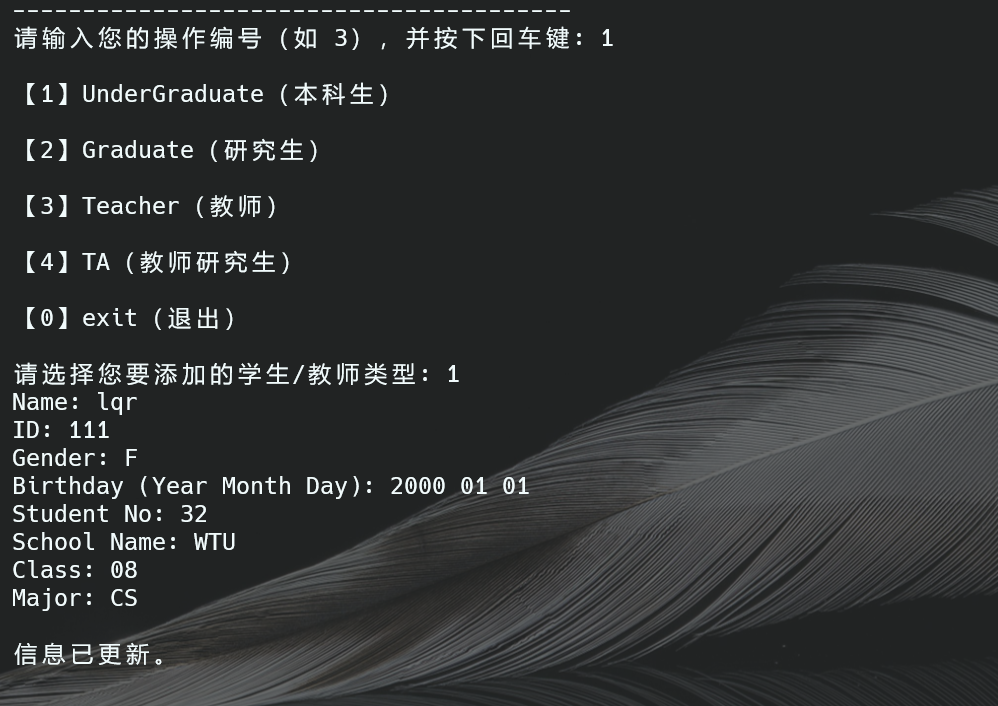
**4.1 Makefile**

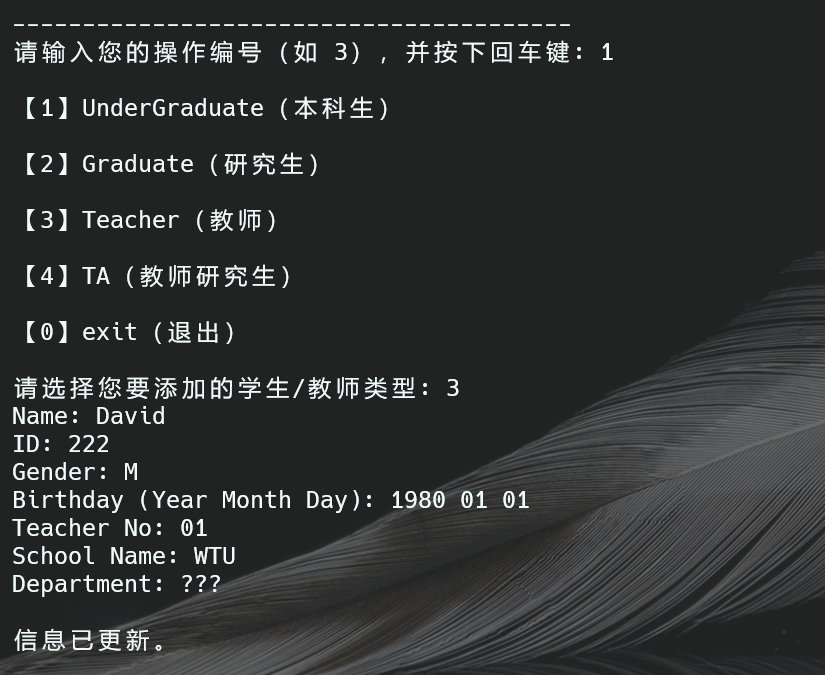


**4.2 主菜单**



**4.3 添加**

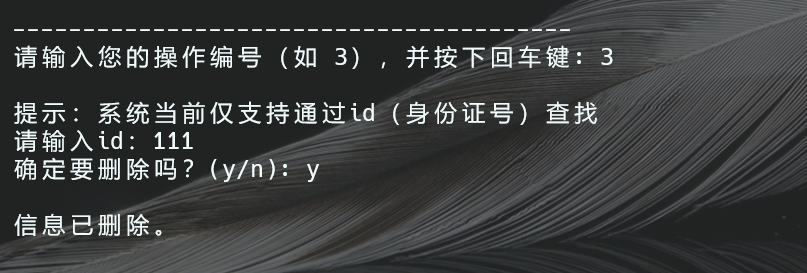




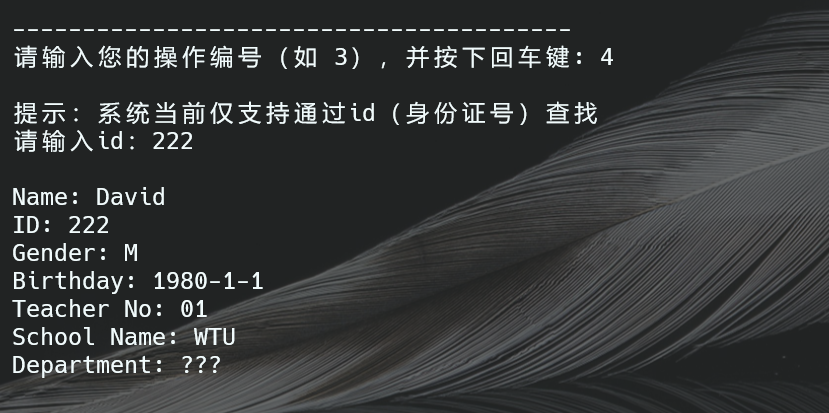
**4.4 修改**



**4.5 删除**

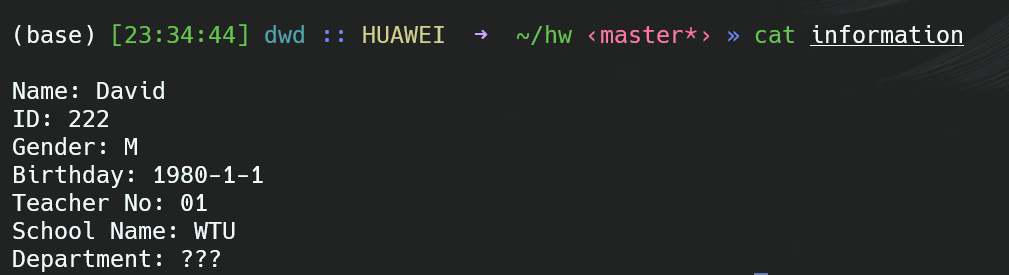


**4.6 查询**

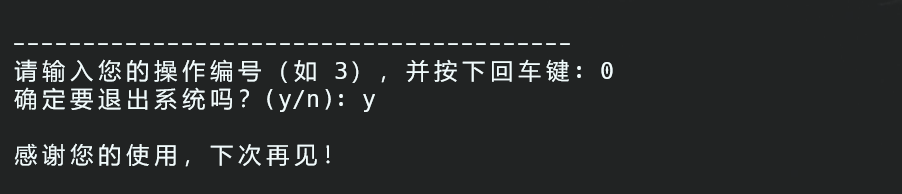


**4.7 保存到文件**





**4.8 退出系统**



**5 系统总结**

**5.1 问题解决过程**

**5.1.1 按id查找的实现方式**

id查找在多处均有应用，因此考虑到信息条目数量级较大时，相较于存储基类的指针数组并遍历查找（查询时间复杂度O(n)），最终选择了哈希表映射（查询操作复杂度O(1)）。

**5.1.2 文件写入部分的代码重用考量**

用来输出的displayDetails函数是直接输出到标准输出流，为了写入文件时可以重用这部分的代码，放弃定义文件流对象，选择用freopen函数重定向stdout到文件中，写入操作完成后再重定向回stdout的方法。

**5.1.3 编译链接过程繁琐**

由于使用编辑器而非IDE开发，不能自动编译链接，于是编写基于GNU make的Makefile文件，大大简化了编译过程的操作。

**5.1.4 封装性不足**

main.cpp很长，易读性差，经过重新封装后，多出了func.cpp和func.h模块，将main函数的篇幅缩短到四十行以内（不包含注释）。

**5.1.5 基类析构函数**

定义为虚函数，才能保证子类的数据成员析构时不会遗漏。

**5.1.6 内存泄露**

主函数结束前，遍历哈希表，释放指针空间。

**5.1.7 流程图**

选择合适的模板多尝试几次。

**5.2 系统存在的不足**

* + 修改操作粗糙，只能重新输入全部信息，无法选择部分修改。
  + 读入的信息不能含有空格（因为是std::cin>>std::string，而std::getline容易读到空行）
  + 使用了一些C++新标准才含有的操作（如auto [x, y]:table遍历哈希表），使得项目无法直接在老旧的编译器上运行。
  + 基于本地Linux环境开发，部分操作不兼容Windows系统，如freopen函数从文件重定向回标准输出流操作。
  + 没有身份验证环节（登录系统）。

**5.3 课程收获**

**5.3.1 大型项目开发经验**

平时写的代码基本都少于100行，也不会分多个.cpp文件，甚至不会写头文件。真正写大项目的时候就显得杂乱无章。这次作业我动手实现了一个400行左右的项目，积累了大项目的开发经验。让我深刻感到实现过程的麻烦，和克服各种问题的乐趣。

**5.3.2 GNU make的学习**

出于更好地管理这个项目的目的，我自学了一些Makefile文件的编写方法和理论知识，动手实现了一个Makefile文件，简化了编译流程。

**5.3.3 git备份**

开发大项目的过程，更让我意识到经常用git备份的重要性。同时，git也是一个越用越熟的工具，下次开发类似的大型项目时，我想尝试将代码托管在github等平台上（这次仅使用了本地提交）

**5.3.4 文件操作**

在写项目的过程中，对比了多种文件输入输出的方式，复习和拓展了流的理论知识和操作。

**5.4 课程建议**

**5.4.1 精简课设要求**

报告已经很完备，很复杂了，还要做ppt，学分少，负担重。

**5.5 发展方向**

**5.5.1 工程能力**

开发项目的过程很锻炼人，通过项目来学习新东西也很有乐趣。我想在大学四年里提高工程能力，提高自己的码量，多实现一些项目，拓展自己的技术栈。

**5.5.2 算法竞赛**

但近期的目标还是提高算法竞赛的能力，即使意识到自己天赋和基础都很一般，还是想在这方面做的更好一些。

我还是时常感到迷茫，我真的算走在acm的路上了吗？我真的适合这条路吗，学一些自己更感兴趣的东西会不会更好呢？毕竟当初参加集训队筛选也不过是因为暑假自学了C语言所以对编程有点自信。当我真正看到那些姑且算是“同行”的人时，我惊讶地发现自己竟然这么没有自觉，抱着一点比周围人强那么一点的自信和偶尔偷懒也无所谓的侥幸心理，就敢说自己在打acm。

在2023湖北省赛之前，我现在的队友邀请我组队。接触之后认识到，他们都是很坚定，很优秀的人，我很荣幸，同样也很害怕，因为我认为他们值得比我更好的队友。

还有很多想说的话，但是算了。

实力是不会骗人的，至少先为了帮上我的队友，继续走下去吧。