C++ 大作业项目报告
——在线教学管理系统

团队成员：马铭康、李远航、郭旭涛、刘思源

2023.12.24

摘要

本报告将详细描述一个简单的在线教学管理系统的具体实现，该系统涉及学生、教师和行政人员；包括课程、通知、作业、评分、评教等多种功能。报告将阐述教学管理系统的小组分工情况、实践探索、思想方法、数据结构以及功能函数。

目录

1 小组分工情况

小组共计四名成员，分别负责完成不同的任务，具体分工如下：

- 马铭康 (项目总监)- 负责（1）完成 UI 图形化设计；（2）完成教务人员部分的代码设计（3）完成用户注册/注销/登录/退出功能；（4）进行代码的汇总与整合
- 李远航（打工人 1 号）- 负责（1）完成课程创建/解散/加入/退出功能代码设计；（2）完成项目报告中的功能测试部分（3）检查代码错误
- 郭旭涛（打工人 2 号）- 负责（1）设计作业的发布/提交部分的代码（2）设计成绩录入/查看部分的代码（3）设计评教功能部分的代码
- 刘思源（打工人 3 号）- 负责（1）初步设计课程创建/解散/加入/退出功能的代码；（2）设计完成通知部分的代码；（3）撰写项目报告中的设计文档部分。

2 程序的总框架图

为了增加项目代码的可读性，我们将通过一个框架图对其进行总体的阐释和解读；

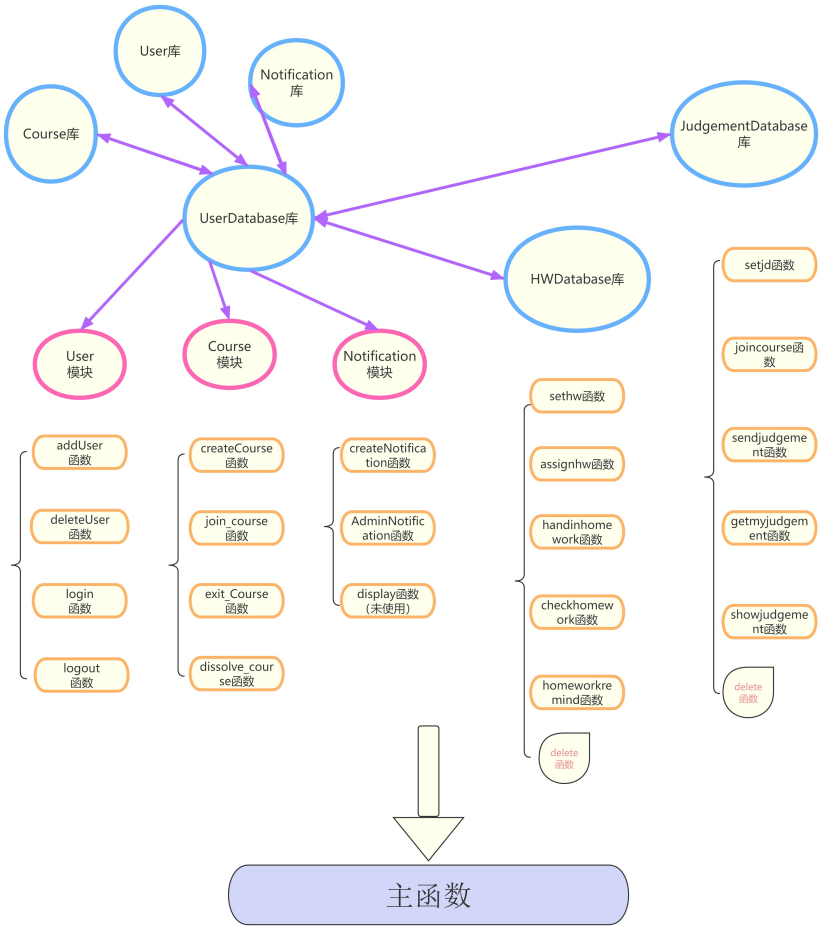

图 1: 程序总框架图

3 实践探索

在这个部分，我们将着重阐述编程过程中遇到的困难和小组成员的具体解决办法。

3.1 建立映射关系

关于这项大作业的编程任务，经过小组的讨论与商议，笔者和其他小组成员一致认为项目中最为困难的部分在于如何将老师或者学生或者助教与课程，通知和作业建立起一一对应的映射关系，显然可以使用链表的方式建立起一一映射的对应关系，但是使用这种方法效率低下，操作和实现起来也较为复杂。如何使代码实现更为高效呢？通过互联网搜索，**查阅相关资料**，我们小组决定使用 std::map 以及 vector 动态数组的方式以实现上述功能。

小组成员通过 B 站课程,CSDN 等网站自学了 std::map 的基本原理和实现方法,我们了解到:std::map 中有一棵红黑树，红黑树是一种自平衡二叉搜索树。下面笔者将对 std::map 进行简要介绍。

在 C++ 的 std::map 中，键（key）是用于唯一标识每个值（value）的对象。std::map 是一种关联容器，它按照键的值进行有序存储，并且每个键都必须是唯一的。键可以是任何可比较的类型，包括内置类型（如整数、浮点数、字符等）和自定义类型（如结构体、类等），只要能够通过比较运算符（如 <、>、== 等）进行键的比较即可。

在 std::map 中，键值对被存储在红黑树的节点中，根据键的值进行排序。这使得 std::map 能够实现高效的查找、插入和删除操作, 最差时间复杂度保持在 O(log n)。对于大量的数据集，std::map 相对于线性搜索或者其他数据结构，具有更好的性能。以下是编程任务中使用到 std::map 的一些操作：

- insert(key, value)：插入一个键值对。
- erase(key)：删除指定键的键值对。
- find(key)：查找指定键的迭代器。
- count(key)：返回指定键在 std::map 中出现的次数。
- size()：返回 std::map 中键值对的数量。

3.2 关于附加功能的选择

由小组分工可以看出，附加功能由马铭康同学完成，我们原计划选择 UI 图形设计和文件 I/O 机制，但为了降低难度（马铭康同学用血泪历史告诉我们文件 I/O 机制的困难，尤其是数据的读入和读出），减少文件 I/O 机制的困扰，我们小组重新选择了教务人员作为附加功能二。

3.3 关于 course 类的设计

由于小组成员刘思源在设计课程功能的代码时，未考虑到同一老师不能创建同课程名的课程和不同老师可已创建同课程名的课程的问题，导致整个课程部分的代码由李远航同学重新设计和完成，这虽然延误了我们的小组进度，但也让小组成员加深了对 vector 动态数组的理解。

3.4 关于 delete 函数的补充

由于协作编程的局限性，HWdatabase 和 JudgementDatabase 中的成员函数并未并入 Userdatabase 类中，故在进行 UI 调试时，发现注销用户后，原有的通知和作业仍然存在，故需要填加 delete 函数进行手动删除数据。具体代码实现如下：

```
    bool deleteteacher(string teacher){
      if (chw.find(teacher) == chw.end())return false;
      else{
        for (auto it1: chw[teacher]) {
          for (auto it2: chw[teacher][it1.first]) {
            chw[teacher][it1.first].erase(it2.first);
          }
          chw[teacher].erase(it1.first);
        }
        return true;
      }

    }
    bool deletecourse(string teacher,string course){
      if (chw.find(teacher) == chw.end())return false;
      else {

        for (auto it1: chw[teacher][course]) {
          chw[teacher][course].erase(it1.first);

        }
        return true;
      }
```

```
class User {
public:
    string username;
    string password;
    string role;
    vector<Course> courses;
    User(string username="", string password="", string role = "") :


    vector<Notification> unreadNotifications;
    vector<Notification> readNotifications;
    vector<Notification> sentNotifications;
};
```

```
class User {
public:
    string username;
    string password;
    string role;
    vector <string> courses;

    User(string username = 0, string password = 0, string role = 0) :
};
```

图 2: 代码改进

4 设计的结构体及其数据成员、成员函数

在这一部分，我们将描述文件中每个类/结构体及其数据成员、成员函数，作为一项面向对象的编程任务，我们主要设计了 6 个类，分别是：User 类，Course 类，Notification 类，Userdatabase 类，HWdatabase 类，Judgementdatabase 类，具体的成员函数和数据成员如下：

4.1 User 类

该类定义了一个名为 User 的类，该类包含以下成员变量：

- username：表示用户名的字符串。
- password：表示密码的字符串。
- role：表示用户角色的字符串。
- courses：表示用户所参与的课程的向量。

该类还定义了一个构造函数，用于初始化类的对象。构造函数参数包括 username、password 和 role，它们都有默认值为空字符串。此外，该类还包括以下成员变量：

- unreadNotifications：表示未读通知的向量。
- readNotifications：表示已读通知的向量。
- sentNotifications：表示已发送通知的向量。

这个类的目的是表示用户的信息和与通知相关的操作。

4.2 Course 类

该类定义了一个名为 Course 的类，该类包含以下成员变量：

- name：表示课程名称的字符串。
- teacher：表示教师名称的字符串。
- students：表示学生名称的字符串向量。该类还定义了一个构造函数，用于初始化类的对象。构造函数参数包括 name 和 teacher，它们都有默认值为空字符串。这个类的目的是表示课程的信息，其中包括课程的名称、教师和学生。

4.3 Notification 类

该类定义了一个名为 Notification 的类，该类包含以下成员变量：

- content：表示通知内容的字符串。
- sender：表示发送者的字符串。该类还定义了一个构造函数，用于初始化类的对象。构造函数参数包括 content 和 sender，它们都有默认值为空字符串。这个类的目的是表示通知的信息，其中包括通知的内容和发送者。

4.4 Userdatabase 类

这一部分使整个项目的结晶所在，其中定义了众多函数，包括 addUser 等用户基本操作模块的函数；createCourse 等课程模块的函数；createNotification 等通知模块的函数。关于具体的函数解释我们将在第五部分函数的输入输出及功能部分进行展示。

4.5 HWdatabase 类

该类定义了一个名为 HWdataBase 的类，这个类包含了以下成员变量和方法，用于管理学生作业的数据库：

成员变量：map<string, map<string, map<string, map<string, homework»» chw; 这是一个多层嵌套的 map，用于存储作业信息。结构如下：最外层的 map 的键是教师的用户名，值是另一个 map。第二层的 map 的键是课程名称，值是第三层的 map。第三层的 map 的键是作业指令，值是第四层的 map。第四层的 map 的键是学生的用户名，值是 homework 类的实例。这个结构允许程序为每个教师的每门课程跟踪不同的作业指令，并且对每个学生单独存储作业信息。

4.6 Judgementdatabase 类

该类定义了一个名为 jdDB 的类，这个类负责处理与评价（judgements）相关的数据。下面是对成员变量的介绍：

成员函数：map<string, map<string, map<string, judgement> cjd; 这是一个多层嵌套的 map 结构，用于存储评价信息。其结构如下所示：最外层的 map 的键（string 类型）是教师的用户名，值是另一个 map。第二层的 map 的键是课程名称，值是第三层的 map。第三层的 map 的键是学生的用户名，值是 judgement 类的实例。此结构允许程序为每个教师的每门课程跟踪每个学生的评价信息。

5 函数的输入、输出、功能及其蕴含的思想方法举例

在这一部分，我们将主要描述主要函数的功能、输入参数、输出结果和某些复杂函数的思想方法。由于项目工作量巨大，我们将在报告中以每个模块中的某一函数为例进行说明（每一模块函数的思想方法大致相近）。另外，输出函数由于UI设计不同于控制台输出，不适用简单的cin,cout，具体实现基于qt自带的库，与本课程知识关联不大，不予解释。

5.1 addUser 函数

```
1    bool addUser(string username, string password, string role);
```

输入: 用户名、密码和角色

输出: 布尔值，表示添加用户是否成功

功能: 在用户数据库中添加一个新用户。如果用户名已存在，函数返回 false。

这段代码的 map 思想方法体现在：它将字符串（用户名）映射到 User 对象。map 是一种关联容器，它存储元素的键值对，其中每个键唯一地映射到一个值。

5.2 createCourse 函数

```
bool createCourse(string username, string courseName)
```

输入: username: 表示要为其创建新课程的用户的用户名。
courseName: 表示要创建的课程的名称。
输出: 布尔值, 指示课程是否成功创建。
功能: 为特定用户（教师）创建一个新的课程。

map 遍历的思想方法体现在这里是通过键（在此案例中为 username）来直接访问 map 中存储的数据。而不是像在数组或标准 vector 遍历中那样使用迭代器或索引。这种直接通过键来访问的特性使得 map 在需要快速查找元素时非常高效。遍历 map 中的元素通常使用迭代器，但在这段代码中，并没有显示遍历整个 map，而是直接通过关键字访问特定用户的信息。

5.3 createNotification 函数

```
bool createNotification(const string& teacherName, const string &
    courseName, const string& content)
```

输入: teacherName：教师的用户名，用于识别发送通知的教师。
courseName：课程名称，用于指定通知所关联的课程。
content：通知的具体内容。
输出: 布尔值，表示通知是否被成功创建并发送
功能: 创建通知实例并将通知发送给学生，然后通过将通知添加到教师的已发送通知列表中来记录教师发送的通知。

map 的思想方法:在这个函数中,map 的作用体现在能够快速定位到特定的用户信息。通过 users.find (teacherName) 检索 map 来确认教师是否存在。一旦定位到特定的教师用户，就可以直接访问其课程列表 users[teacher Name].courses，通过键（用户名）访问 map 中的用户信息。在找到具体的课程后，再次利用 map 的直接访问能力，通过学生的用户名作为键访问其用户信息，以便向他们的未读通知列表中添加通知。

5.4 assignhw 函数

```
bool assignhw(string course,string teacher,string instruction);
```

输入: course：课程名称，用于指定要分配作业的课程。
teacher：教师的用户名，用于指定分配作业的教师。
instruction：作业指令或描述，用于具体说明这次作业的要求。
输出: 布尔值，表示作业是否成功分配。
功能: 创建作业容器，插入作业指令然后将作业分配给学生。

这段代码中的 assignhw 方法用于分配作业，涉及到了 map 的多层嵌套结构。在 HwDataBase 类中，chw 成员是一个复杂的 map，它的结构是这样的：

- 最外层的 map 以教师的名字作为键，其值是另一个 map;
- 第二层的 map 以课程名称为键，其值是第三层的 map;

- 第三层的 map 以作业指令为键，其值是第四层的 map；
- 第四层的 map 以学生名字为键，其值是最终的 homework 对象。

通过上述步骤，该方法将为特定课程下的每个学生分配一个作业条目。这个多层嵌套的 map 结构使得能够方便地访问和管理不同层级的信息，比如可以快速检索特定教师的特定课程的特定作业指令下的所有学生作业情况。

5.5 setjd 函数

```
1    bool jdDB::setjd(string teacher,string course)
```

输入: teacher：教师的用户名。
course：课程名称。
输出: 布尔值, 表示是否成功设置了判断（judgement）
功能: 检查教师是否存在，然后插入教师条目；检查课程是否存在，然后插入课程条目；为后续的评教功能做准备。

以上是对项目中一些典型函数的举例，而并非是对所有函数的解释，尽管如此，其他函数的实现方法与上述函数也大致相同，通过 vector 动态数组建立列表储存用户，课程，作业的数据，并通过 std::map 进行数据的插入，删除，遍历等操作。

6 总结

通过实现这个系统，我们小组不仅提高了编程能力，也加深了对面向对象设计的理解。但由于时间仓促和编写团队的协调沟通以及自身能力有限等原因，该大作业项目仍存在等问题，如果未来仍有机会，我们会将工作集中在提高系统的安全性、扩展功能和优化用户界面，来实现项目的迭代与完善。