**高级语言程序设计实验报告**

**医**

**院**

**预**

**约**

**挂**

**号**

**系**

**统**

**学院：公共卫生学院**

**专业：预防医学**

**姓名：苏江慧**

**学号：2020303051052**

**目 录**

[1需求分析 2](#_Toc24723)

[1.1 目标 2](#_Toc7064)

[1.2 主体功能 2](#_Toc16840)

[1.3 开发环境 3](#_Toc32172)

[2 功能实现 4](#_Toc25114)

[2.1 类设计 4](#_Toc16948)

[2.2 结构设计 6](#_Toc14508)

[2.3 详细设计 6](#_Toc5149)

[（1） 登录或注册登录 6](#_Toc31659)

[（2） 患者界面 8](#_Toc29713)

[（3） 医生界面 15](#_Toc26523)

[（4） 管理员界面 18](#_Toc30972)

[3 遇到的问题及应对 22](#_Toc14924)

[（1）问题1 22](#_Toc10681)

[4 收获总结 24](#_Toc17996)

[参考文献 25](#_Toc24882)

# **1需求分析**

随着经济的迅速发展，人们的生活水平得到显著提高。医疗服务行业也在人们生活中占据了最重要的一环，随着医改的实施，看病难，看病贵的问题得到了解决，但是现场挂号效率较低且消耗的成本高。所以为了存储患者信息以及医生信息，为了提升医院的管理能力和运行效率，医院预约挂号管理系统在此背景下诞生，同时通过挂号管理系统对医疗信息进行整合，利用数据分析疾病流行情况，为疾病预防提供决策支撑。

医院预约挂号管理系统就是对实现挂号电子化，存储患者信息以及医生信息在PC上的编程实现。

## **1.1 目标**

设计一个患者能够查询到医生信息、预约挂号的系统，同时能够实现医院管理方对医生信息进行修改、增加、删除，对挂号的属性进行修改的功能。

## **1.2 主体功能**

医院预约挂号系统共有两大模块，分为用户模块和挂号模块。

1. 用户模块共有三类用户：患者、医生、管理员。

①患者功能：注册账号，登录，查看、修改个人信息，注销账户，查询相应科室的医生信息，预约挂号，查询个人挂号信息，取消挂号预约；

②医生功能：注册账号，登录，查看、修改个人信息，查询挂号信息；

③管理员功能：登录，查看所有人信息，添加医生账户，注销任意账户，查询所有挂号信息，添加号类型（例如某科室某医生某时间段的号就属于患者所挂的号对应的类型），添加号（某一号类型下创建多少个号）；

（2）挂号模块：患者挂号、查询挂号信息、取消挂号预约、添加号类型和号；

## **1.3 开发环境**

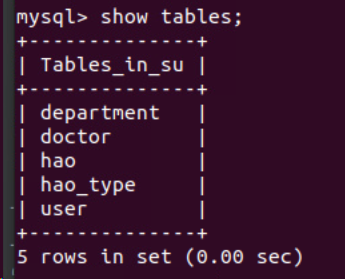
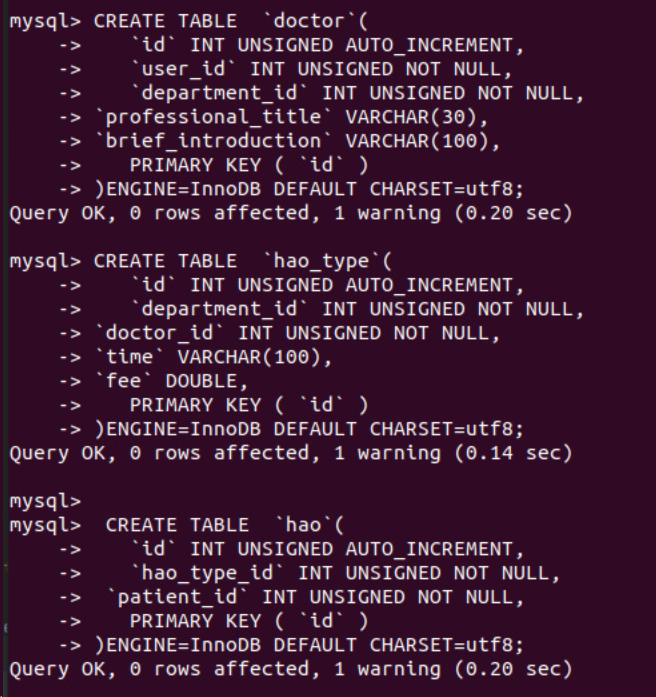
（1）开发系统：**Ubuntu** 20.04LTS 64位



（2）编译集成软件：**CLion** 版本:2022.1.1

CLion是一款适用于C和C++开发者的的跨平台IDE，不仅仅是一款编辑器，它提供强大的调试器和动态分析工具，可以轻松调查并解决问题；它支持CMake，CMake 是著名的跨平台建构系统，广泛用于 C 和 C++ 项目。通过代码生成、自动补全和自动更新来充分利用 CLion 的智能 CMake 支持，本系统中即使用到了该建构系统。

（3）数据库：mysql-server\_8.0.29-1

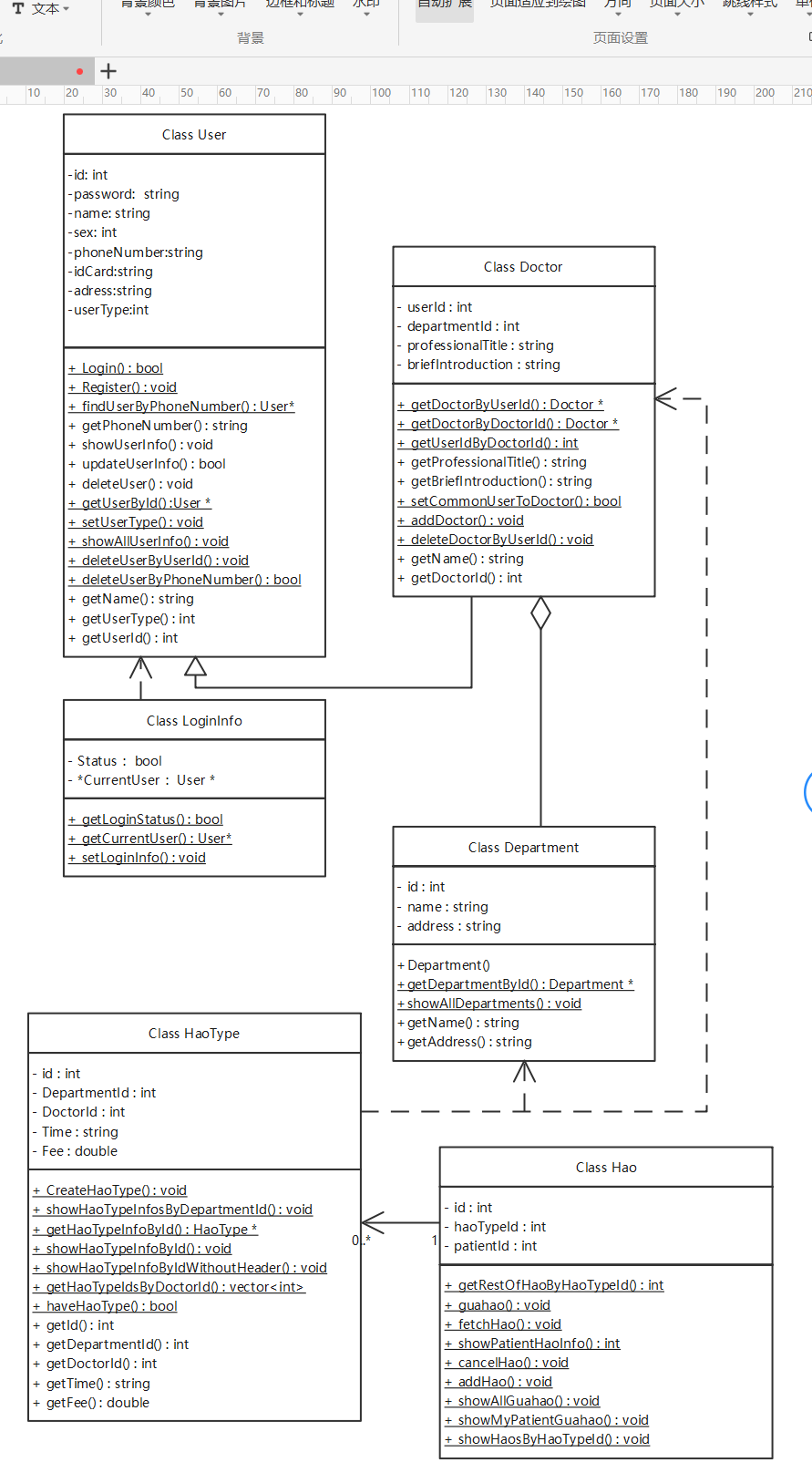


建立数据库su,并建立所需要的表。

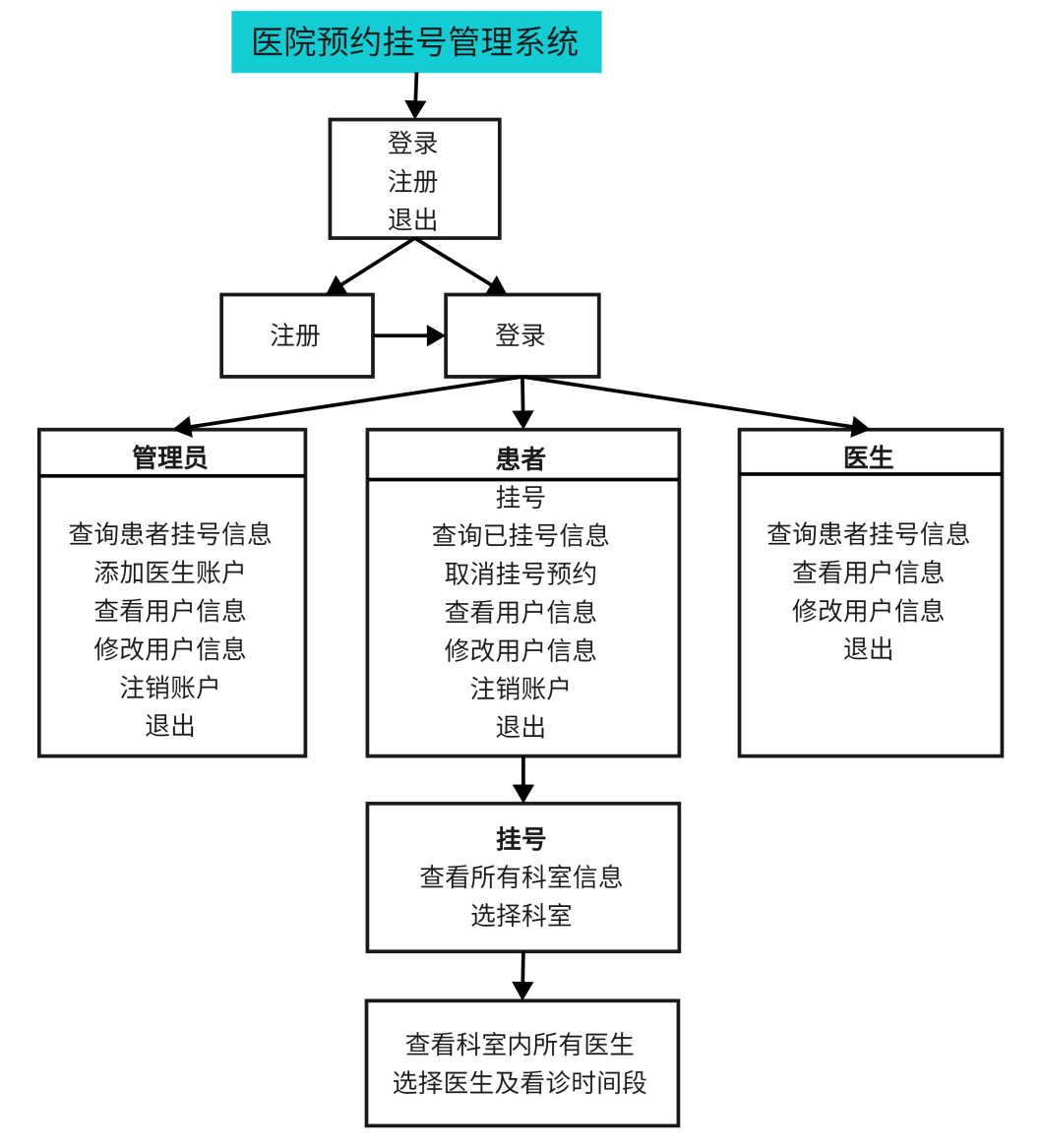
注：（数据库设置：允许用户“su”连接数据库,密码为“123”。将数据库su附加到数据库。）

# **2 功能实现**

## **2.1 类设计**



## **2.2 结构设计**



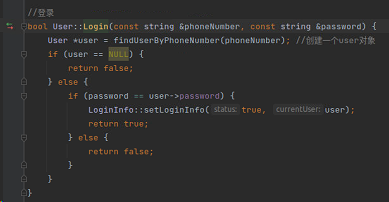
## **2.3 详细设计**

1. **登录或注册登录**



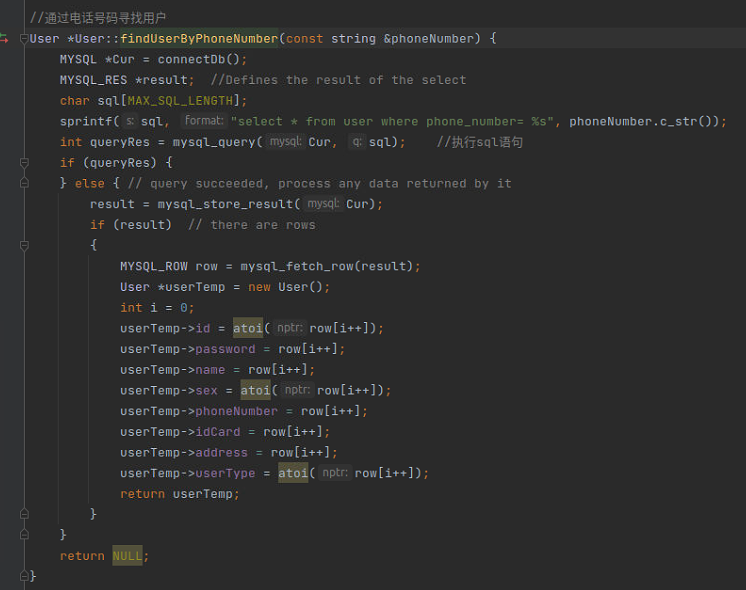
提示用户输入账户和密码后，将账号密码传入login函数进行身份验证。若账号不存在或者密码错误，提示用户账号或密码错误，进行注册或重新登录；若账号存在，将用户输入的密码与数据库中存储的账号密码进行比较，密码一致的情况下显示登录成功，并通过数据库中的用户信息判断用户类型（患者、医生、管理员），根据账户类型显示不同的界面。

具体实现如下：



这里密码是直接明文存储在数据库中的，直接通过电话号码读取用户信息。

FindUserByPhoneNumber实现如下：



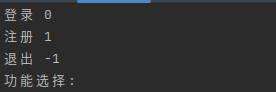
①首先连接数据库获取MYSQL\*连接句柄，连接操作和相关信息已封装到connectDb函数中，这里不作过多阐述。

②在拿到句柄后，即可开始构造sql语句对数据库进行查询。

③拿到查询结果后，利用mysql.h中的mysql\_fetch\_row逐行读取查询到的数据，存放到临时对象中，这里一个手机号只可能对应一个用户，因此读一行即可。随后对读取到的数据的格式进行处理后存放到临时user对象中，最后返回。

④失败时返回NULL，用于上级函数判断是否存在用户。

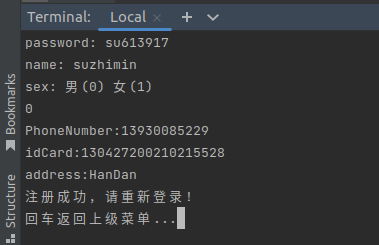
运行结果如下：



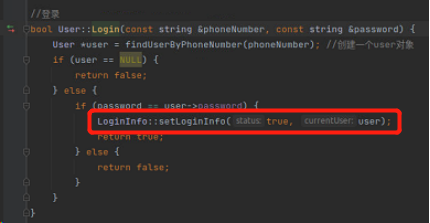
①登录界面：



②注册界面：



1. **患者界面**

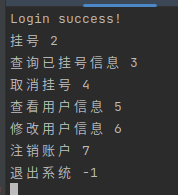


登录时用setLoginInfo函数记录当前登录用户的信息，并用LoginInfo类实现封装。



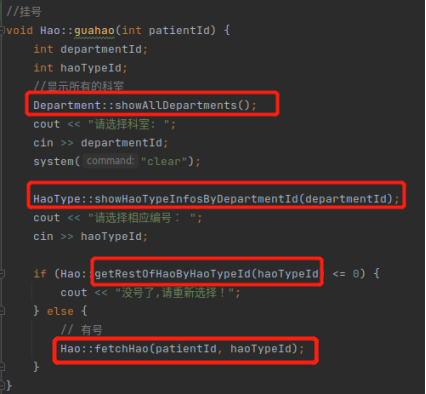
通过LoginInfo类中的getCurrentUser函数得到当前用户的信息，并将其储存在一个临时user对象中，提示用户选择需要的功能。利用switch语句实现功能选择。执行完相应功能后，用enterBack函数实现按回车键返回上级菜单。

运行结果如下：



具体功能实现如下：

**①挂号**

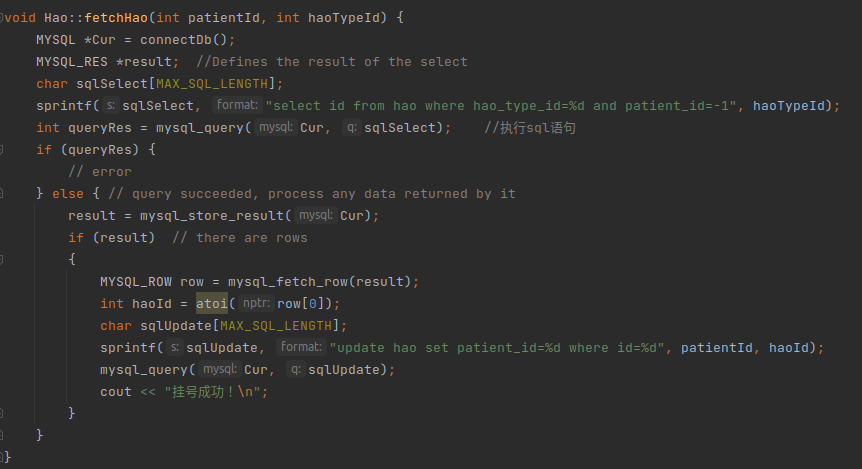


1、通过showAllDepartments函数从数据库中调用科室信息，向用户展示所有科室的名称和地址，提示用户选择科室id；

2、将科室id传入showHaoTypeInfosByDepartmentId函数中，从数据库hao\_type表中根据科室id调用号类型信息，展示所选择科室内各医生的信息及挂号的相关信息，提示用户选择号类型id；

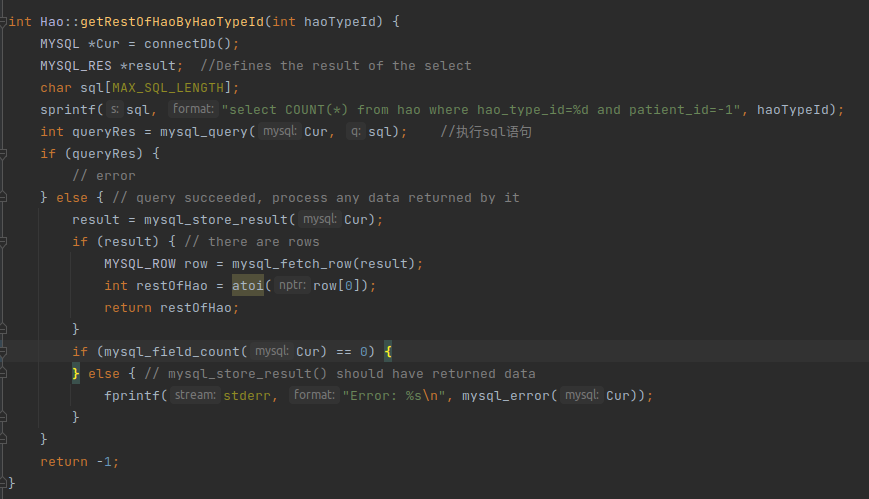
3、将号类型id传入getRestOfHaoByHaoTypeId函数中得到该号类型下的剩余号数，若剩余号数为0，则请用户重新选择；若不为0，则通过fetchHao函数执行取号操作。

fetchHao实现如下：



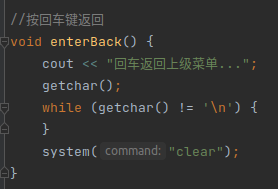
在数据库hao表中调出相应号类型id下患者id为-1的行，将第一行的患者id（添加号数时将号的患者id统一设置为-1，下文会提到）更新为该用户在user表里的id。

getRestOfHaoByHaoTypeId实现如下：



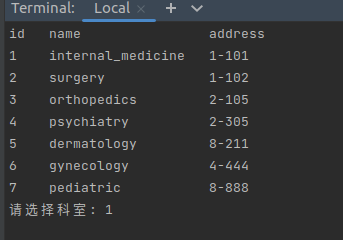
在数据库hao表中选择相应号类型id下患者id为-1的行，统计行数，即为某号类型下的剩余号数。

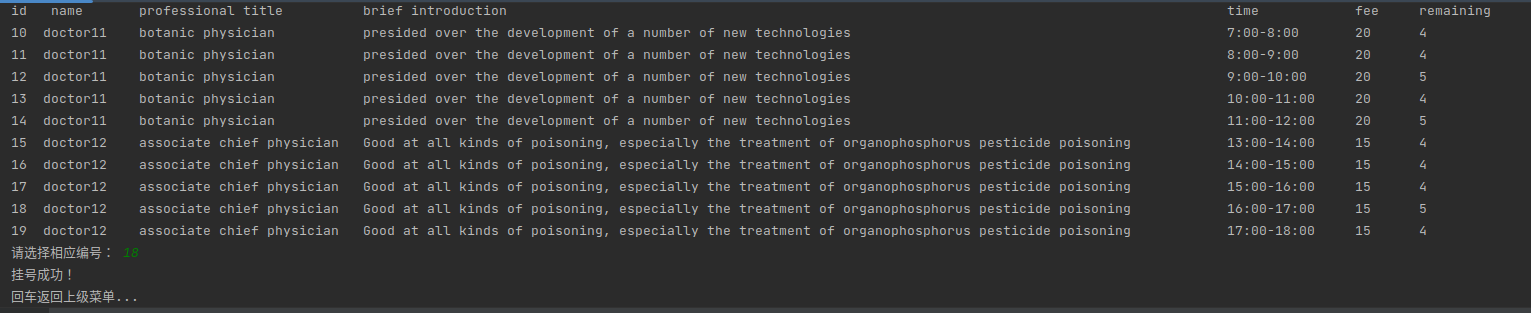
enterBack实现如下：



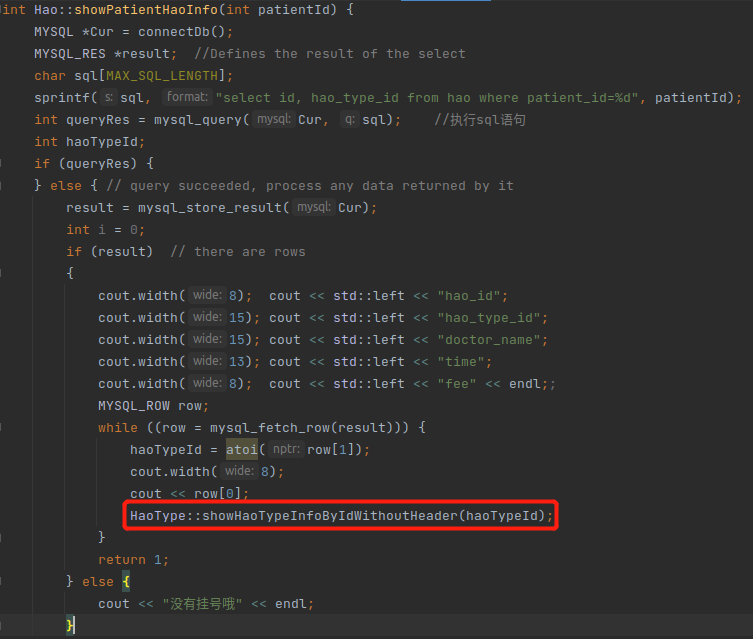
利用getchar()字符输入语句，在程序运行时接收用户从键盘输入的一个字符，当接受到回车键时，执行清屏命令，展示上级菜单。

运行结果如下：





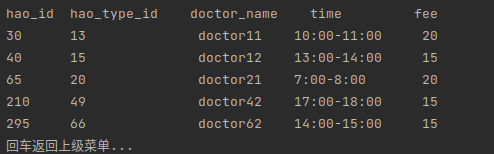
**②查询已挂号信息**



1、根据临时user对象存储的患者id，在数据库的hao表里找出患者id对应的号id和号类型id，将找到的hao类型id传给showHaoTypeInfoByIdWithoutHeader函数；

2、在showHaoTypeInfoByIdWithoutHeader函数中，根据号类型id在数据库hao\_type表里找到号类型的相关信息并输出。

运行结果如下：

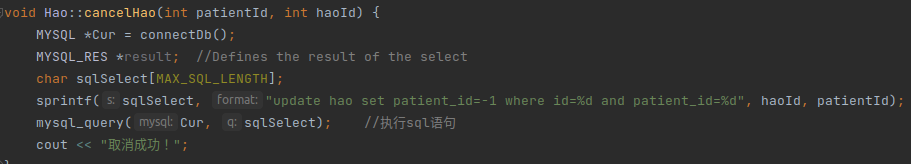


**③取消挂号**



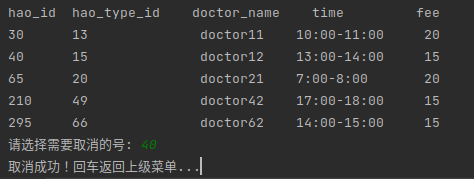
通过showPatientHaoInfo函数显示患者已挂号信息，同上，不再赘述，将用户传入的hao\_id传入cancelHao函数中，实现取消预约挂号功能。

cancelHao具体实现如下：

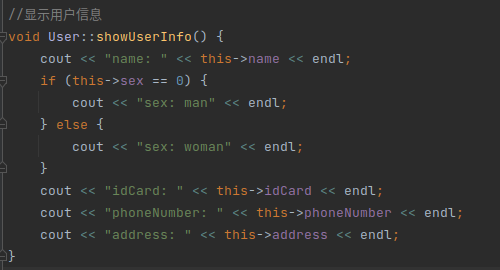


通过临时对象存储的patient id和用户传入的hao id在数据库hao表里将对应的patient id更新为-1，使该号能重新被其它用户选择。

运行结果如下：

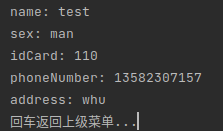


**④查看用户信息**



将临时user对象中存储的用户信息进行格式化输出

运行结果如下：



**⑤修改用户信息**



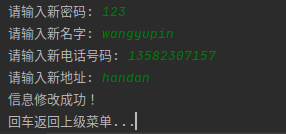
根据updateUserInfo函数的返回值进行判断，若返回“true”，则提示用户“信息修改成功”；若返回“false”，则提示用户修改失败，询问用户是否继续修改。

updateUserInfo()具体实现如下：

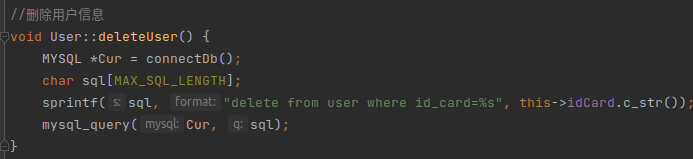


判断是否成功执行sql语句，若成功执行，返回“true”，若失败，返回“false”，返回结果用于上级函数判断是否成功修改数据库中的用户信息。

运行结果如下：



**⑥注销账户**

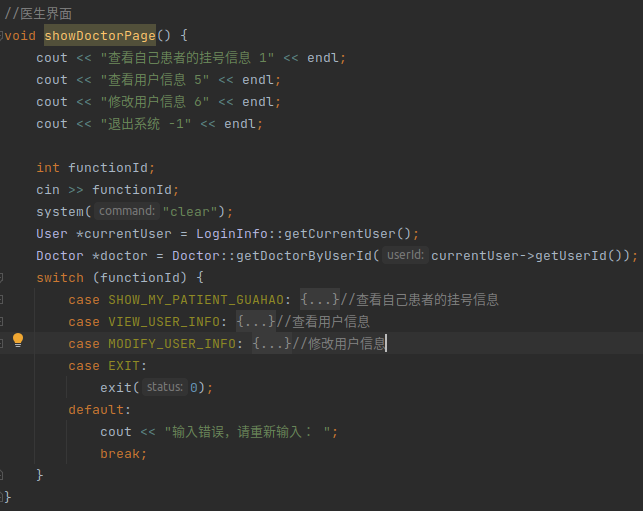


一个id card只对应一个用户，所以根据用户的id card在数据库的user表里删除用户信息

运行结果如下：



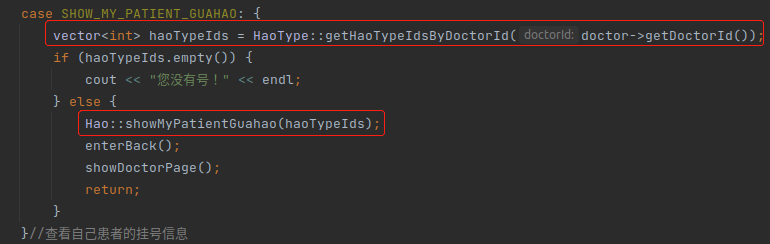
1. **医生界面**



同患者界面实现相同，创建一个临时user对象存储当前用户的信息，利用switch语句实现功能选择。

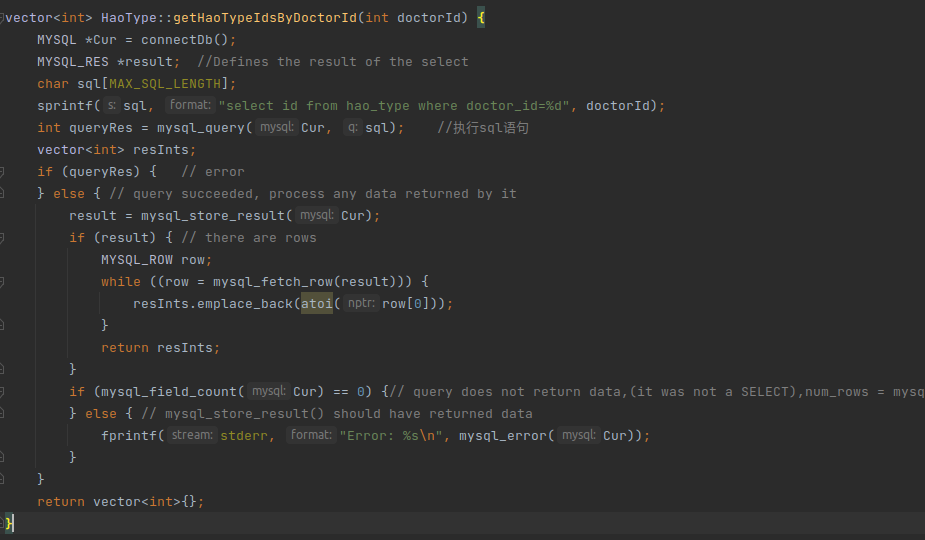
**具体功能实现如下：**

**①查看自己患者的挂号信息**



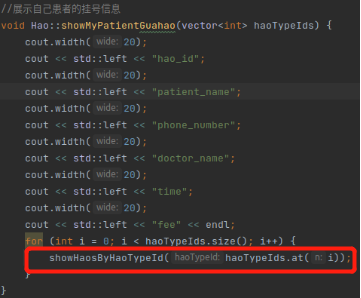
将当前医生id传入函数getHaoTypeIdsByDoctorId中，得到的号类型的数据用一个int型的容器来储存，再利用showMyPatientGuahao函数将该医生自己患者的挂号信息输出来。

getHaoTypeIdsByDoctorId实现如下：



在数据库hao\_type表里查询相应医生id对应的号类型的行，因为大部分实际情况下挂号类型的数据不止一行，所以函数返回的数据存储在int型的容器中。与push\_back相比，emplace\_back的优势在于模板参数列表可变长和完美转发，因此选用emplace\_back

showMyPatientGuahao实现如下：



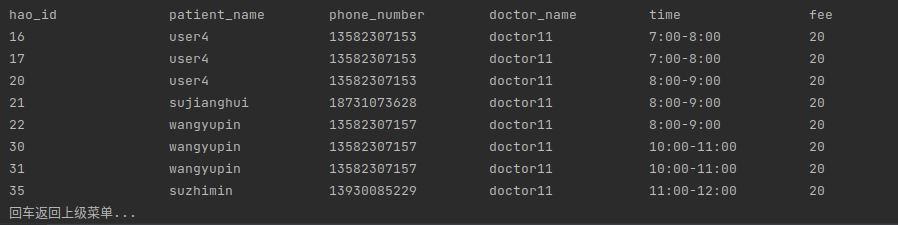
输出要显示信息的目录，通过遍历容器中的haoTypeIds,将多个haoTypeIds逐个传递给showHaosByHaoTypeId函数，实现患者挂号信息的多行输出。

showHaosByHaoTypeId函数实现如下：



1. 用临时haoType对象将号类型有关的信息封装；
2. 一个号类型对应一个医生id，所以可通过号类型得到医生id，用临时doctor对象将通过医生id得到的医生相关信息封装；
3. 通过号类型id，查询数据库hao表中患者id不为-1（代表已被挂号）的号所在行，输出首列号id对应的信息；
4. 格式化输出其它医生需要了解的信息。

运行结果如下：



**②查看用户信息**

实现同普通用户查看用户信息，不再赘述。

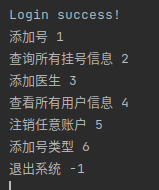
**③修改用户信息**

实现同普通用户修改用户信息，不再赘述。

1. **管理员界面**

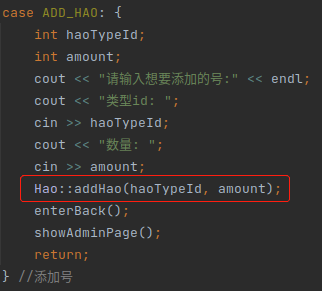
界面实现基本同上，这里不再赘述。

运行结果：



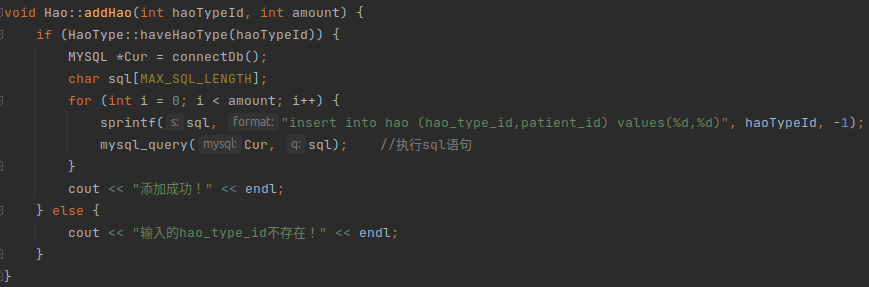
**具体功能实现如下：**

**①添加号**



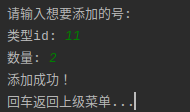
管理员输入想要添加的号类型id和数量，将输入的信息传到addHao函数中。

addHao函数具体实现如下：



将数据插入到数据库hao表中，提示用户“添加成功”。

运行结果如下：

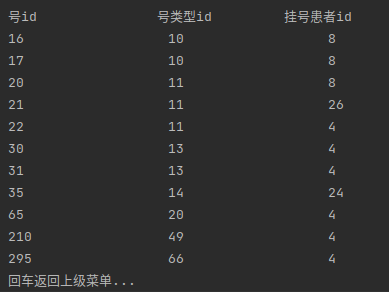


**②查询所有挂号信息**

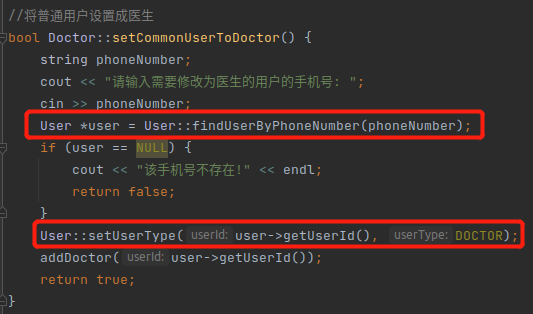


在数据库hao表里查询患者id不为-1的行，遍历各行的信息，格式化输出。

运行结果如下：

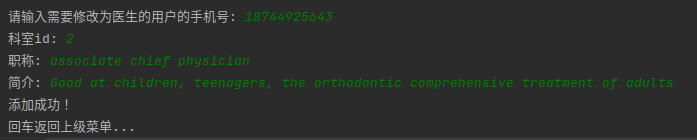


**③添加医生**

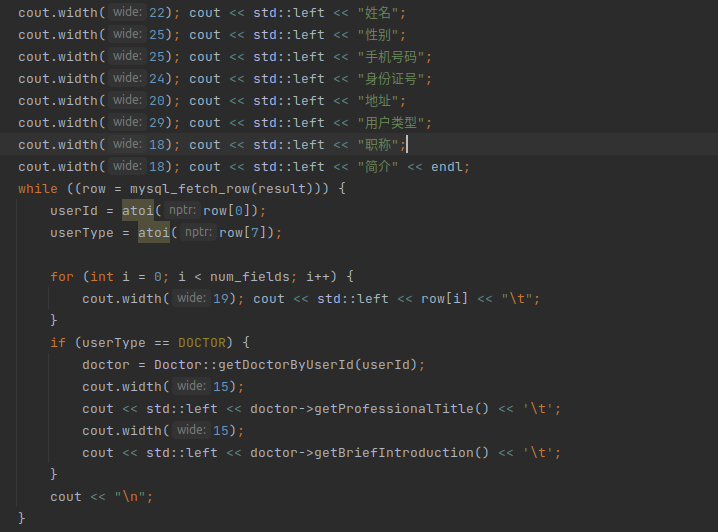


首先需要用手机号注册一个普通账户，再由管理员根据用户输入的手机号，通过findUserByPhoneNumber函数在数据库user表里找到对应的用户，利用setUserType函数在数据库中将用户类型设置为doctor。

运行结果如下：

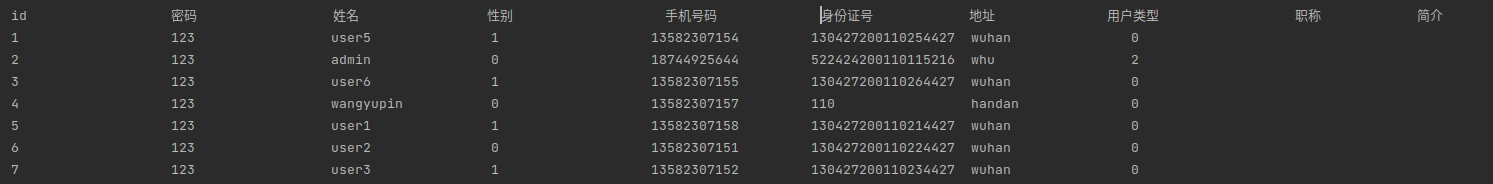


**④查看所有用户信息**

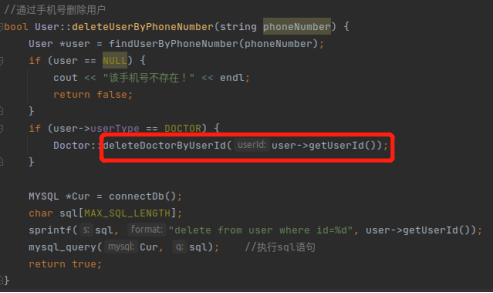


在输出数据库user表信息的基础上，用if语句判断用户类型，若为医生，则需要用函数getDoctorByUserId通过user id得到医生的信息，输出职称和简介。

运行结果如下：

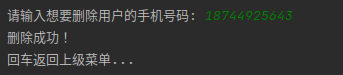


**⑤注销任意账户**



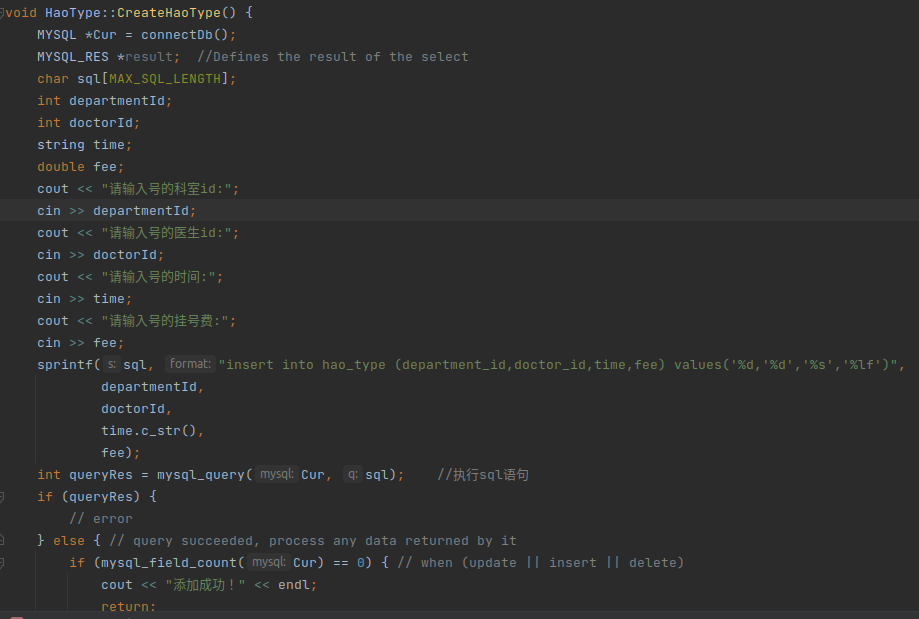
用户手机号唯一，根据手机号删除user表里对应的信息，若用户为医生，需要利用deleteDoctorByUserId函数删除doctor表里的相关账户信息，若手机号在数据库里找不到，提示用户“该手机号不存在！”

运行结果如下：



通过查询数据库进行验证，未找到已删除的用户，确定删除成功。

**⑥添加号类型**



将管理员输入的信息格式化后插入到数据库hao\_type表里。

# **3 遇到的问题及应对**

**（1）问题1**

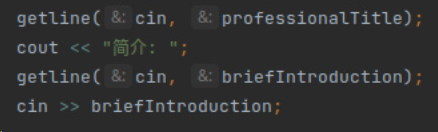
引用另一个头文件的类报错，发现问题在于引用其它头文件时，还需另外声明需要用到的类。

**（2）问题2**

在添加医生简介的时候，发现含有空格的简介，在空格处会被截断。

分析：原因是使用的cin遇到空格或者回车键，输入流即停止。

解决办法：cin改用getline即可。



随后又发现getline好像和cout有点冲突，导致中间夹着getline的两个cout会连续输出。

解决办法：在cout后穿插getchar()



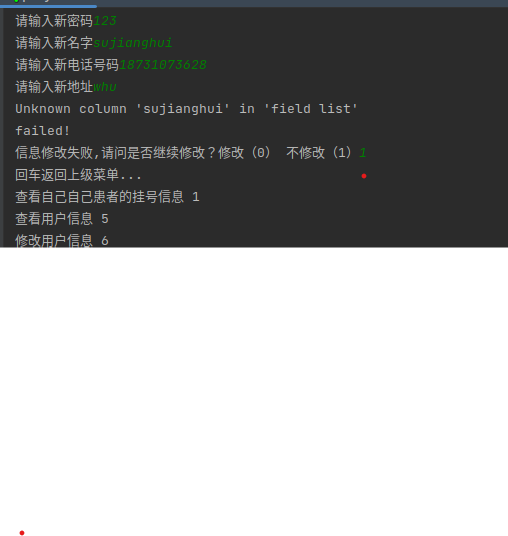
**（3）问题3**

system(“clear”)清屏不管用。

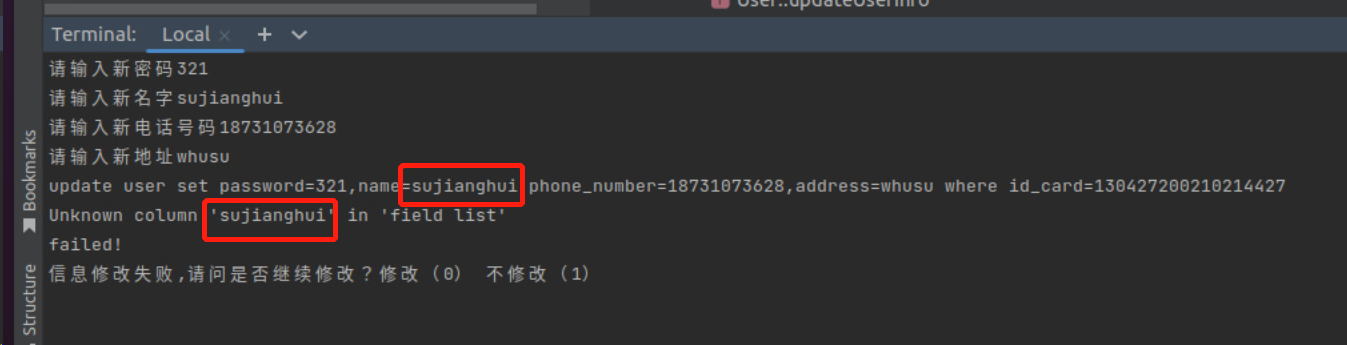
分析：应该是因为clion直接运行的地方，和linux的命令行不太一样，clear指令未能被识别到。

解决办法：使用clion build好之后，直接在linux命令行运行即可。

**（4）问题4**



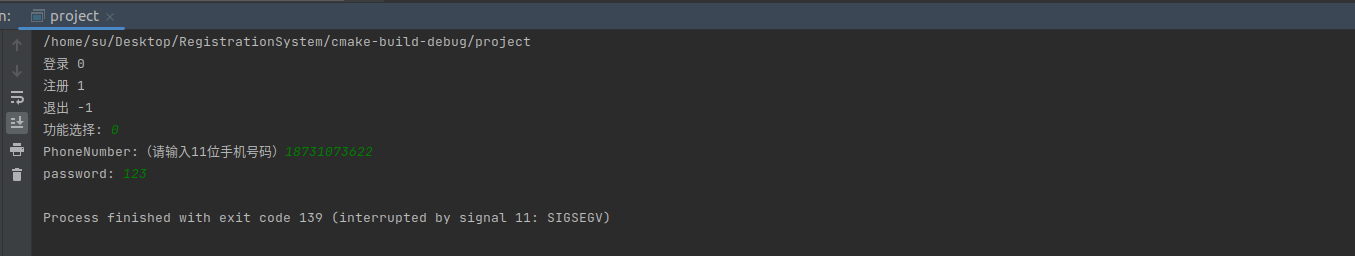
分析：发现问题在于



解决办法：Sql语句中，插入的字段内容如果是字符串，应该加上单引号或者双引号。

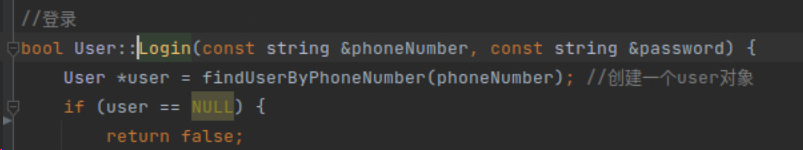
1. **问题5**

输入账号错误时不提示账号错误，直接退出



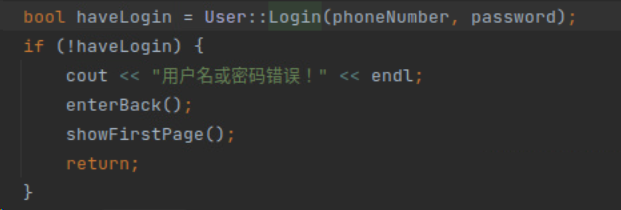
原因：未对账号不存在的情况进行处理

解决方法：



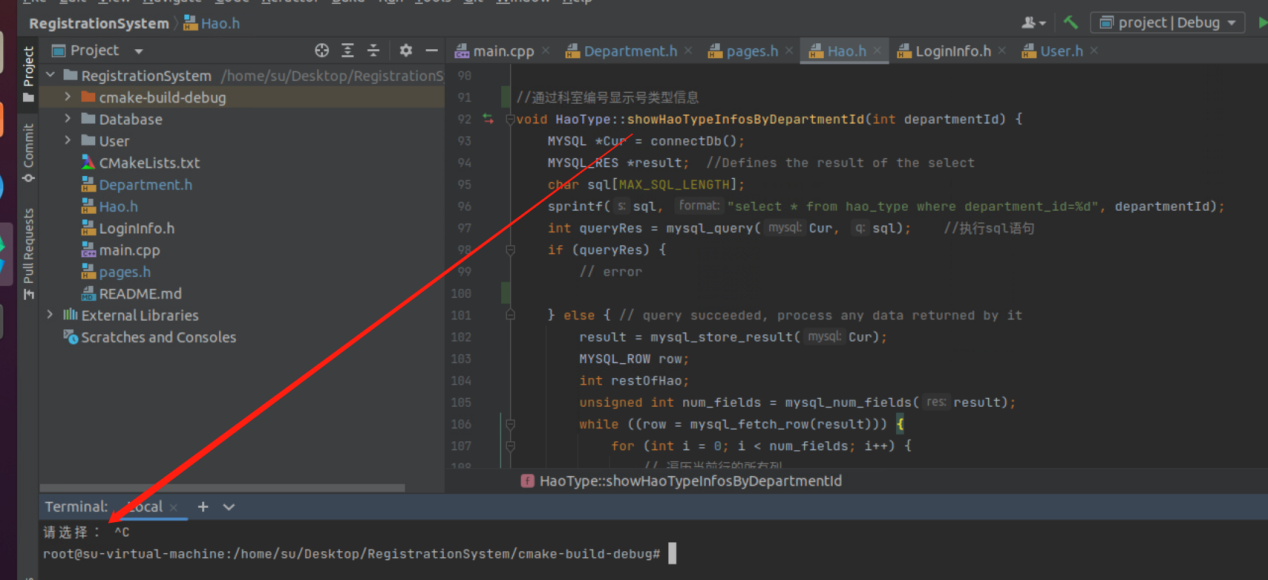
在数据库中如果找不到用户，返回用户为NULL，再判断如果没用户，就返回登录失败

最后根据登录成功与否提示相关信息



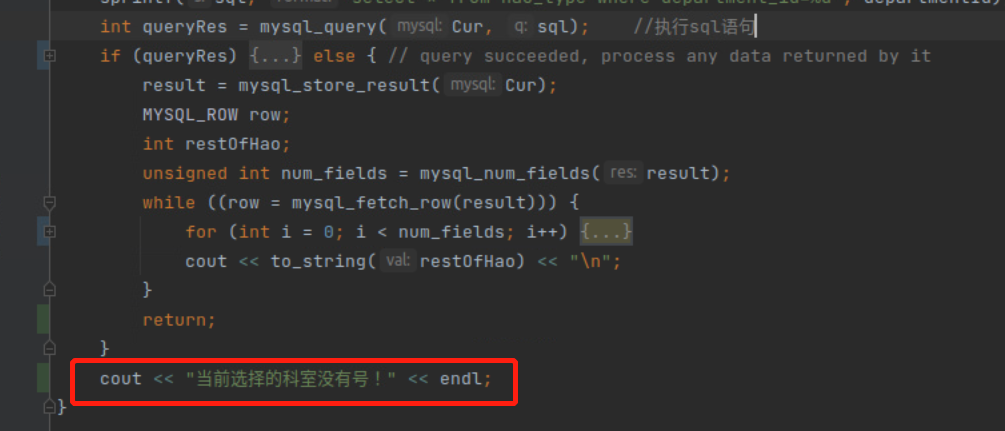
1. **问题6**

根据科室id查找号类型时，提示请选择但没有可选择的项



原因分析：此时该科室下不存在号类型

解决方案：在该科室下不存在号类型时，提醒用户该科室下没有号



# **4 收获总结**

通过完成这个项目，我学到了很多书上得不到的东西。首先，在实现这个系统的过程当中，感觉最大的收获并不是在于使用c++语言本身来进行编程，而在于怎么利用程序的逻辑去构思一个系统应该怎么设计。我在设计的过程当中确实遇到了非常多的问题，比如“在空格处被截断”、“输入账号错误时不提示账号错误”等，有些是关于语法的，也有些是关于逻辑的，要解决这些问题，需要花费很多的时间和精力，但是通过自己检索关于那些问题的信息和解决方法，通过自己实际动手去解决那些沉睡的逻辑上的问题的时候，不仅仅学会了一种全新的看待程序bug的角度，更是在解决问题的过程当中，学会了如何去使用一些系统提供的库函数，以及什么样的情况下，该使用到什么样的数据结构，编程能力在实践中得到了非常大的提升。

另外，在本次实验设计的过程中，我还发现了把患者信息，医生信息以及挂号等信息存储起来是一件非常麻烦的事情，如果说仅仅使用的我们所学过的文件操作的话，数据的读取，以及统计等操作都非常麻烦，因此，我上网搜索了类似的系统是如何处理数据的，发现几乎都使用到了数据库，其中看起来比较方便的，是一种叫做orm的框架，但是由于C++这方面生态非常不完善，相关框架版本都非常老旧，在安装的过程当中也频繁地遇到了各种各样的问题，绝大多数的问题和系统的库函数相关，由于我对系统底层了解的比较浅薄，所以我放弃了使用这种方案。选择了另一种方案，直接利用比较通用的myqsql.h提供的数据库接口，虽然使用起来不如orm框架方便，但是比起文件操作来，数据的结构化还是强了很多。

最后则是在开发的过程中发现，有些功能很相似，做了不少重复的工作，查阅了不少类似系统以及一些轻量的开源工具后才发现，功能应尽量独立，这样才能在后续的开发中避免反复造轮子。在经过了一番优化之后再添加一些功能的过程当中，我也感受到了尽可能的拆分成多个函数的优点。

# **参考文献**

[1]Stephen Prata著.张海龙,袁国忠译.C++ Primer Plus.人民邮电出版社.2012年;

[2]钱能.C++程序设计教程[M]2版.北京:清华大学出版社.2005;

[3]郑莉编.C++语言程序设计.北京：清华大学出版社.2008年;

[4]缪淮扣.顾训穰,沈俊.数据结构Q--C++实现[M].北京:科学出版社，2002;

[5]Bruce Eckel,Chuck Allison.C++编程思想.机械工业出版社.2011;

[6]Bjarne Stroustrup.C++程序设计语言.机械工业出版社.2016;

[7]http://www.cplusplus.com/reference/;

[8]https://www.mysqlzh.com/.