四川轻化工大学毕业设计（论文）

基于 QT 的中国象棋游戏的设计与实现

学　　生：刘宇

学　　号：18104011407

专　　业：计算机科学与技术

班　　级：2018 级 14 班

指导教师：刘小芳

四川轻化工大学计算机科学与工程学院

二 O 二二年六月

独 创 性 声 明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。据我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得四川轻化工大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。

作者签名：_____ 日期： 年 月 日

关于论文使用授权的说明

本学位论文作者完全了解四川轻化工大学有关保留、使用学位论文的规定，有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许论文被查阅和借阅。本人授权四川轻化工大学可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文。

（保密的学位论文在解密后应遵守此规定）

作者签名：_____ 导师签名：_____

日期： 年 月 日

基于 QT 的中国象棋游戏的设计与实现

摘 要

本设计是一个以 QT Creator 作为开发工具、基于 QT 框架的一个中国象棋游戏系统。系统由客户端、服务器端和数据库三部分构成。客户端使用 QT 框架的 GUI 模块搭建，通过 Network 模块进行套接字通信。服务器端使用 QT 框架的 Network 模块与客户端进行套接字通信。数据库使用 MySQL 数据库来设计，并使用 QT 框架的 SQL 模块与 MySQL 数据库进行数据交换。该系统包括客户端与服务器端两大模块。客户端包括用户登录、用户注册、网络对战、对局聊天四个子模块。服务器端包括网络通信、对局匹配、登录与注册验证三个模块。

关键词：中国象棋，QT 框架，网络通讯，数据库

Design and Implementation of Chinese Chess Game Based on QT

ABSTRACT

This design is a Chinese chess game system based on QT framework with QT Creator as a development tool. The system consists of three parts: client, server and database. The client is built using the GUI module of the QT framework, and socket communication is performed through the Network module. The server side uses the Network module of the QT framework to communicate with the client side by socket. The database is designed using the MySQL database and uses the SQL module of the QT framework to exchange data with the MySQL database. The system includes two modules: client and server. The client includes four sub-modules: user login, user registration, network battle, and game chat. The server side includes three modules: network communication, game matching, login and registration verification.

Keywords: Chinese Chess, QT Framework, Network Communication, Database

目　录

第一章 绪论

千百年来，人们常把琴棋书画相提并论，象棋艺术也被认为是中华民族的瑰宝之一，它融体育、艺术、科学、文化等多领域于一身，是中国传统文化的一朵奇葩[1]。中国象棋发源历史悠久，根据调查每十五个中国人中就有一个人会下中国象棋。中国象棋不仅娱乐身心，同时也锻炼了脑力，但传统中国象棋需要棋手面对面下棋，在时间、地域方面有诸多限制。随着网络技术的不断发展，中国象棋也随之进入了互联网时代。本文设计并实现了中国象棋游戏的基本功能，主要分为中国象棋规则的设计实现、用户登录系统的设计实现，解决了游戏在各平台客户端不统一的问题，满足各操作系统用户的使用需求。

1.1 研究背景

人们熟知的 PC 端操作系统主要有三个平台，其中主流的桌面操作系统一直是 Microsoft 公司研究开发的 Windows 系列操作系统，其次是 Apple 公司研发的 Mac OS 操作系统，最后是开源的 Linux 操作系统。绝大多数用户会使用其中一个或多个操作系统，用于包括但不限于工作生活等。每个操作系统都有自己的特色，从而形成了软件生态。对于软件开发人员来说，需要在不同平台上开发统一功能、外观的软件，这无疑是困难且繁琐的。使用 QT 开发游戏应用的优势在于可以实现一次编码，多端应用，从而减轻了程序员的负担。

1.2 课题研究意义

郭建欣在“基于机器视觉的象棋对弈系统研究”一文中提到，从计算机延生开始，计算机和人在象棋上的对战就没有停止过。象棋博弈所要进行的搜索复杂度比较大，很合适考验计算机的计算能力，象棋搜索算法也是伴随着计算机的计算能力的发展而不断发展[2]。

刘淑琴，刘淑英在“基于博弈树搜索算法的中国象棋游戏的设计与实现”一文中提到，计算机游戏不仅能够在虚幻中满足玩家的欲望给予完美的体验，亦能在游戏中获取知识。比如中国象棋游戏，能够锻炼玩家拥有缜密的判别能力，博弈中提高自身修养[3]。

1.3 国内外研究现状

1. 国外研究现状

国际象棋作为一项古老的文化遗产，在国外会被更多的人选择。至今，对于它的问世、演变、传播的历史，还没有一个为大家所公认的、结论性的意见，其中争论最多的还是中国象棋与国际象棋出现时机的讨论。但是国际象棋走法规则与中国象棋有很多相似之处，中国象棋棋子走法一直没有改变过，只有胜负判定规则有过变动，而国际象棋棋子走法有过较大变动。

就程序设计而言，国际象棋设计难度略低于中国象棋，体现在国际象棋棋盘相对较小，胜负判断更简单，而中国象棋灵活多变，所以设计难度高于国际象棋。但是国外的国际象棋游戏设计已经非常成熟，具有较高的参考价值。QT 是软件开发领域中非常著名的 C++可视化开发平台，能够应用程序开发者提供建立艺术级图形用户界面所需的所有功能。它是完全面相对象的，很容易扩展，并且可用于组件编程。相较于 Visual C++，QT 更易于学习和开发[4]。因此在国外通常使用 QT 作为游戏的开发框架，例如车机系统、医疗系统、机械设备、手机系统的图形界面开发等。

2. 国内研究现状

相较于国际象棋，国内用户更青睐于具有本国特色的中国象棋游戏，因此利用 QT 框架对游戏进行开发具有较大的市场前景。从游戏开发的角度出发，早期的游戏以单机游戏为主，国内比较有名的象棋类游戏有象棋巫师等。随着网络覆盖的逐渐加深，单机游戏远远不能满足用户的需求，人们把游戏的注意力转移到真人对战，比较受欢迎的象棋游戏如象棋旋风和象棋助手等[5]。

第二章 需求分析

需求分析是开发软件项目的开端，象棋游戏设计也是如此。象棋游戏具有开发智力、培训独立思考的能力以及锻炼思维逻辑等优点。但是传统的线下象棋会受到时间地点的限制，不能够很好的满足玩家的需求。因此本系统通过完善线上象棋游戏的各个功能，从而提高玩家的满意度。

2.1 可行性分析

1. 技术可行性

在软件方面，本系统使用 QT Creator 作为开发工具，客户端开发过程中，界面设计使用 QT 内置 UI 模块进行设计，网络通信使用 Network 模块进行开发。服务器端采用 Network 模块、Database 模块进行开发，用户数据使用 MySQL 数据库进行保存。在硬件方面，客户端与服务器端均可以在 Windows、Linux、MacOS 系统上运行。通过上述分析可得出结论，在软件、硬件方面均可轻松实现本系统，因此，在技术实现方面本系统具有较高可行性。

2. 经济可行性

本系统对系统开发者来说并不需要太高的成本支出，只是对系统的管理者付出管理报酬即可，而且开发周期不需要太长，节省了人力、物力、财力资源，所以本系统在经济上是可行的。

对开发本系统的开发者来说，QT 框架有跨平台特性，使用 QT 框架开发减少了开发复杂度。QT 框架功能完善，文档手册齐全，遇到问题能很好解决，缩短了开发周期。开发本系统使用的软件、框架，运行平台均有免费开源方案可选择，具有较高的经济可行性。

3. 社会可行性

象棋在国内有着庞大的玩家群体，各个年龄段都有受众。目前网络使用广泛，象棋游戏是传统象棋与网络的结合，玩家可方便的参与到游戏中来，因此本系统具有社会可行性。

2.2 功能需求分析

本系统实现象棋游戏的主要功能，包括主要用户登录、用户注册、网络对战、对局聊天等功能，方便玩家在各种网络环境下也能进行游戏。同时在网络对战时

能互相发送消息，满足玩家交流的需求。

2.2.1 客户端功能

1. 用户登录

用户运行客户端时，客户端会默认自动与服务器建立连接，如果成功与服务器连接，界面的登录与注册按钮会变成可点击状态，窗口下方的状态栏会变为绿色，反之如果未连接服务器，登录与注册按钮为不可点击状态，状态栏红色提示。当用户网络恢复时，可通过窗口上方状态栏连接服务器。在连接服务器的状态下，输入用户 ID 与密码即可登录。

2. 用户注册

在连接服务器的状态下，单击注册按钮可变换到注册界面，注册界面需要输入想要注册的用户 ID 与两次密码，用户 ID 要求为 6 位纯数字，密码要求 6-16 位数字、字母组合，注册成功自动登录。

3. 网络对战

当用户成功登录以后，点击菜单栏中的开始匹配按钮进入匹配队列，等待匹配成功开始对战。如果中途想要停止匹配，可以点击停止匹配按钮退出匹配队列。

4. 对局聊天

在网络对战与局域网对战模式下，玩家可在左侧聊天窗口中进行实时聊天。限制单条聊天信息长度不超过 256 个字符长度，聊天信息条数不作限制。如对方与服务器断开连接，会在聊天窗口中以红色字体颜色提示。

2.2.2 服务器端功能

1. 网络通信

在服务器端开启网络监听，等待客户端连接，如果有客户端进行连接，则需要响应客户端连接，并准备好与客户端之间的数据传输工作。当接收到客户端发来的数据时，需要分析出数据所对应的指令，并发送相应的内部信号，即响应客户端的请求。

2. 对局匹配

当用户开始匹配对局时，创建对局线程，把匹配成功的用户送到对局线程中，同时响应对局中的用户操作。

3. 登录、注册验证

数据库的操作包括两部分，第一部分是在服务器刚开始运行时，就要与数据

库建立连接，如果连接失败，则发出错误信息，提醒管理员数据库连接失败，如果连接成功才进行第二部分操作。第二部分操作主要是与数据库进行数据交换，以满足业务需求。

2.3 相关技术分析

1. QT 框架

QT 是跨平台的程序开发框架，预处理器 MOC 用于扩展 C++语言，支持 Windows、Linux、iOS 等多个平台[6]。QT 最早是在 1991 年由奇趣科技开发的，1996 年进入商业领域，成为全世界范围内数千种成功的应用程序的基础[7]。虽然 QT 有商业版本，但并不影响它是一个免费开源的框架，按照不同的发行版，分为商业版与开源版，开源版在遵守 GNU 协议下是免费的，所以受到了广大程序员的喜爱。QT 开发的应用程序有很多，其中较为出名的有 Autodesk Maya、Bitcoin、Battle.net 等。

QT 框架中包含很多子模块，其中核心模块 GUI 是每个应用程序必备的，常用模块有 Network、Database 等模块，除核心模块以外，使用其他子模块需要在配置文件中配置才能使用。

QT 框架的核心思想体现在信号和槽机制，信号和槽机制用于响应界面操作，调用对应的槽函数，从而实现交互。当某个信号对其客户或所有者发生的内部状态发生改变，信号被一个对象发射。只有定义过这个信号的类及其派生类能够发射这个信号。当一个信号被发射时，与其相关联的槽将被立刻执行，就象一个正常的函数调用一样[8]。想要使用信号与槽机制的前提是所有类必须从 QObject 类派生，并在构造函数中声明 Q_OBJECT 宏，最后绑定信号与槽函数即可。

2. QT Creator 集成开发环境

QT Creator 是一个跨平台的 C++、JavaScript 和 QML 的集成开发环境，使用它可以简化 GUI 应用程序的开发。QT Creator 在 Linux 上使用来自 GNU Compiler Collection 进行编译。在 Windows 上，它可以使用默认安装的 MinGW 或 MSVC 进行编译，也可以在从源代码编译时使用 Microsoft 控制台调试器。QT 开发者绝大多数都会使用 QT Creator 进行应用开发，因为它是官方指定的开发工具。虽然 Visual Studio 中集成了 QT 插件，但在 QT 应用程序开发方面还是 QT Creator 会有更多优势。

3. MySQL 数据库

MySQL 数据库是关系型数据库管理系统，由瑞典公司开发，现属于 Oracle 旗下产品。MySQL 数据库属于中小型数据库，相较于 Oracle 数据库而言，MySQL 数据库在性能、安全性方面并不占优势，但 MySQL 数据库体积小，响应快，在性能要求不高的场景下有不错表现。并且，MySQL 数据库同样有免费版与商业版，对于绝大多数开发者和小公司来说，免费版无疑是最经济的选择。

第三章 系统总体设计

上一章分析了本系统的需求与可行性，本章将根据上一章的需求进行具体的功能设计。其中分为两大类，一类是客户端功能的具体实现，另一类是对服务器端功能的具体实现。

3.1 系统软件总体结构设计

本系统是一个基于 QT 框架的桌面游戏系统，采用 C/S 体系结构，根据上一章的需求分析，将本系统分为客户端功能实现与服务器端功能实现。客户端的功能包括用户登录、用户注册、网络对战、对局聊天等功能，服务器端的功能包括网络通信、对局匹配、登录与注册验证等。

3.1.1 客户端功能模块

1. 用户登录

用户运行客户端时，客户端会默认自动与服务器建立连接，如果成功与服务器连接，界面的登录与注册按钮会变成可点击状态，窗口下方的状态栏会变为绿色，反之如果未连接服务器，登录与注册按钮为不可点击状态，状态栏红色提示。当用户网络恢复时，可通过窗口上方状态栏连接服务器。在连接服务器的状态下，输入用户 ID 与密码即可登录。

2. 用户注册

在连接服务器的状态下，单击注册按钮可变换到注册界面，注册界面需要输入想要注册的用户 ID 与两次密码，用户 ID 要求为 6 位纯数字，密码要求 6-16 位数字、字母组合，注册成功自动登录。

3. 网络对战

当用户成功登录以后，点击菜单栏中的开始匹配按钮进入匹配队列，等待匹配成功开始对战。如果中途想要停止匹配，可以点击停止匹配按钮退出匹配队列。

4. 对局聊天

在网络对战与局域网对战模式下，玩家可在左侧聊天窗口中进行实时聊天。限制单条聊天信息长度不超过 256 个字符长度，聊天信息条数不作限制。如对方与服务器断开连接，会在聊天窗口中以红色字体颜色提示。

3.1.2 服务器端功能模块

1. 网络通信

在服务器端开启网络监听，等待客户端连接，如果有客户端进行连接，则需要响应客户端连接，并准备好与客户端之间的数据传输工作。当接收到客户端发来的数据时，需要分析出数据所对应的指令，并发送相应的内部信号，即响应客户端的请求。

2. 对局匹配

当用户开始匹配对局时，创建对局线程，把匹配成功的用户送到对局线程中，同时响应对局中的用户操作。

3. 登录、注册验证

数据库的操作包括两部分，第一部分是在服务器刚开始运行时，就要与数据库建立连接，如果连接失败，则发出错误信息，提醒管理员数据库连接失败，如果连接成功才进行第二部分操作。第二部分操作主要是与数据库进行数据交换，以满足业务需求。

3.2 结构总体设计图

本系统包括客户端和与服务器端两个部分。客户端的功能包括用户登录、用户注册、网络对战、对局聊天，服务器端的功能包括网络通信、对局匹配、登录与注册验证。系统总体结构图如图 3-1 所示。

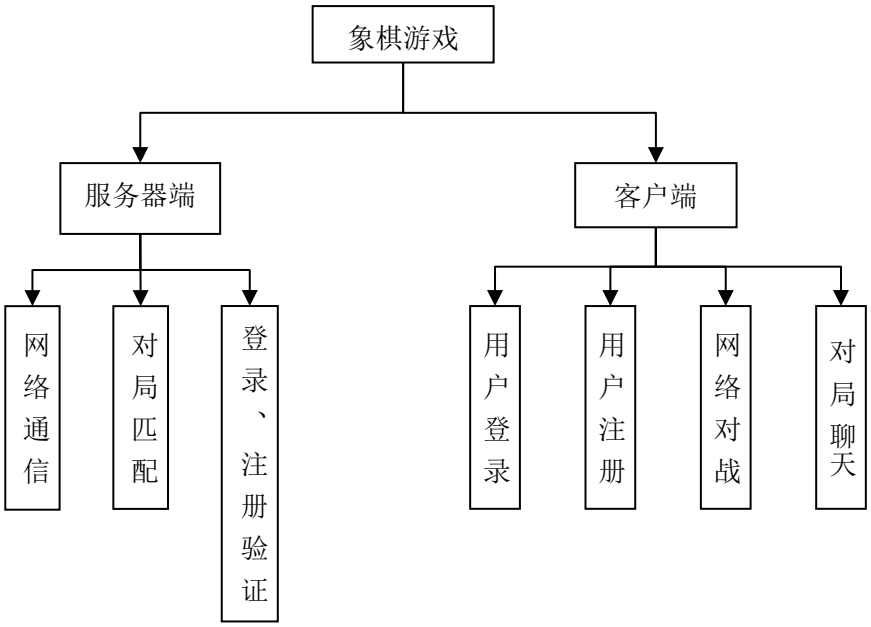

图 3-1 系统总体结构图

第四章 系统数据库设计

上一章对本系统做了详细的设计，明确了游戏的总体设计思路，本章将对其中关于数据库的模块进行详细设计与实现。

4.1 数据库实体概念

1. 数据库

数据库就是信息的集合，它可以存在很长时间，往往是很多年[9]。关系型数据库 MySQL 由表构成，表由行和列构成。每一列都包含了每一行的一个值：表中不存在裂缝或短列。行是单独的实体，列标记了那些实体的属性[10]。

2. 数据表

数据表是数据库的重要组成部分。数据表是由表名、表中的字段和表的记录三个部分组成的。设计数据表结构就是定义数据表文件名，确定数据表包含哪些字段，各字段的字段名、字段类型、及宽度，并将这些数据输入到计算机当中[11]。数据表可以附加限制，不符合规则的数据无法保存。

4.2 数据库实体设计

E-R 图也称实体-联系图（Entity Relationship Diagram），采用三个基本概念：实体集、联系集和属性。它是描述现实世界关系概念模型的有效方法。

1. 用户（**用户 ID**，用户密码）：用户信息实体如图 4-1 所示，其属性包括用户 ID、用户密码两部分，其中用户 ID 为主键。

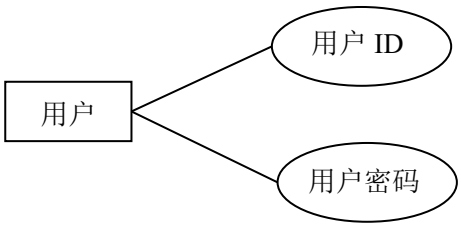

图 4-1 用户信息实体

第五章 系统功能模块实现

上一章设计并实现了数据库，在此基础上，本章节将对系统主要功能模块进行实现。本系统的功能模块包括两大部分，第一部分是客户端的功能模块，第二部分是服务器端的功能模块。

5.1 客户端功能模块的实现

5.1.1 用户登录模块

1. 功能设计

用户运行游戏直接进入登录界面，界面由三部分组成，分别为上方菜单栏、下方状态栏以及中间的登录窗口组成。登录窗口中包含账号输入文本框、密码输入文本框以及登录按钮与注册界面切换按钮。连接服务器使用基于 TCP 的套接字通信。

（1）菜单栏

菜单栏的功能丰富，默认状态下菜单栏中所有选项都默认为不可选中状态，通过用户的操作相应的开启或关闭其中的选项，菜单栏中的游戏模式菜单可以切换不同的游戏模式。

（2）状态栏

状态栏默认为未连接状态，默认显示红色状态提示。当客户端与服务器成功连接时，转换为连接状态，显示绿色状态提示，直到与客户端断开连接，会重新显示红色状态提示。

（3）登录窗口

登录窗口中包含两个文本框，一个用于输入密码，一个用于输入账号，还包含两个按钮，一个用于切换到注册界面，一个用于提交登录。只有成功连接服务器端时，登录与注册按钮才会变为用状态。

当用户运行游戏时，客户端会自动与服务器端连接，如果连接成功，下方状态栏会变成绿色，此时登录按钮与注册按钮变为可点击状态，此时才能进行游戏的登录，若无法连接服务器，则不能进行登录或注册操作。登录时账号需要满足的条件是 6 位纯数字，而密码需要满足条件是长度 6-16 位且包含字母与数字。满足登录条件以后，才会将账号密码发送到服务器端。登录流程如图 5-1 所示。

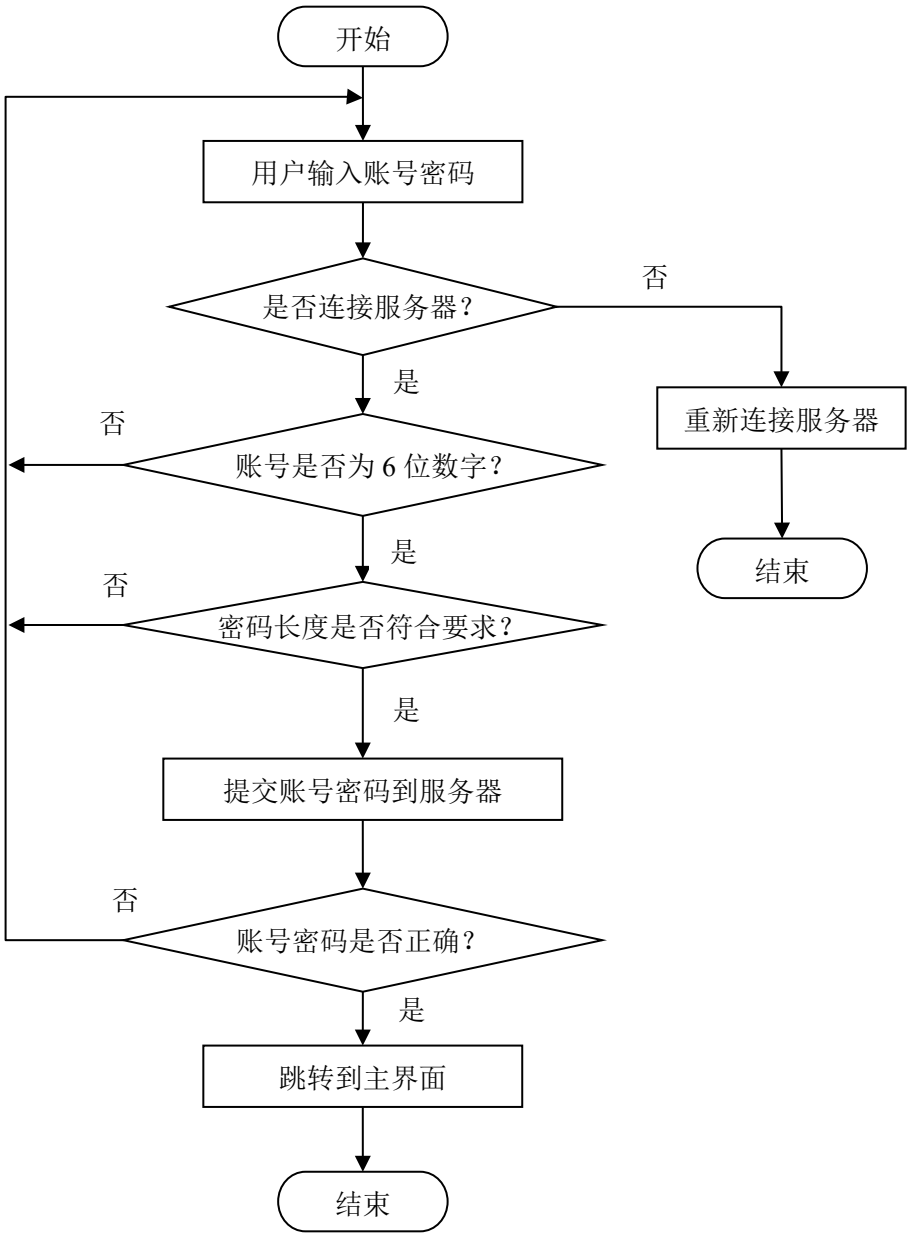

图 5-1 登录流程

2. 界面设计

（1）菜单栏，菜单栏中包含三个菜单，分别是游戏模式菜单、游戏选项菜单、游戏信息菜单。游戏模式菜单中有两个菜单项，分别是断开服务器、连接服务器，这两个按钮可以连接或与服务器断开连接，游戏模式菜单如图 5-2 所示。

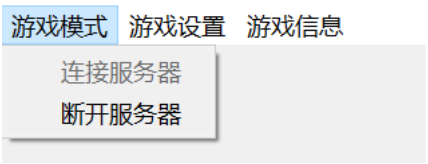

图 5-2 游戏模式菜单

游戏设置菜单一共有四个菜单项，分别是开始匹配、停止匹配、认输、悔棋。当用户登录游戏以后，菜单项才会变为可点击状态。游戏设置菜单如图 5-3 所示。

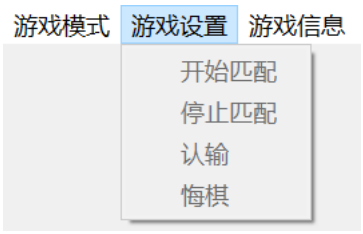

图 5-3 游戏设置菜单

游戏信息菜单中只有一个菜单项，叫做作品信息，点击作品信息选项会弹出一个消息窗口，显示作品信息，其中包括作者、游戏名称、创建日期、开源地址等。作者信息窗口如图 5-4 所示。

图 5-4 作者信息窗口

（2）状态栏，用于显示当前连接服务器的状态，服务器未连接如图 5-5 所示，服务器已连接如图 5-6 所示。

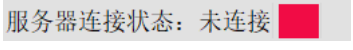

图 5-5 服务器未连接

图 5-6 服务器已连接

（3）登录窗口，登录窗口中包含两个文本框，一个用于输入密码，一个用于输入账号，还包含两个按钮，一个用于切换到注册界面，一个用于提交登录。登录窗口如图 5-7 所示。

图 5-7 登录窗口

登录界面由菜单栏、状态栏、登录窗口三部分组成，登录界面如图 5-8 所示。

图 5-8 登录界面

5.1.2 用户注册模块

1. 功能设计

用户注册功能与登录功能类似，当用户点击登录界面中的注册按钮时，关闭登录窗口，打开注册窗口。用户注册的流程与登录一致，如果注册失败会有错误弹窗提示。

2. 界面设计

从登录界面切换到注册界面时，会有弹窗提示用户 ID 与密码的格式要求，注册说明弹窗如图 5-9 所示。

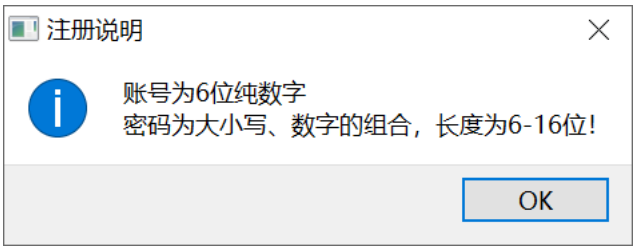

图 5-9 注册说明弹窗

注册界面中有三个文本框，用于输入 ID、输入密码、确认密码，有两个按钮，分别是返回按钮与注册按钮，返回按钮用于返回主窗口，注册按钮用于提交注册信息。注册界面如图 5-10 所示。

图 5-10 注册界面

5.1.3 网络对战模块

1. 功能设计

（1）套接字通信

QT 中自带 Network 模块用于网络通信，该模块是对 C++网络通信的二次封装，使开发者能更便捷的进行网络通信。客户端首先通过调用 Socket 函数创建 TCP 套接口，指定服务器 IP 地址和端口，然后调用 connect 函数与服务器取得连接，接着进行数据处理，读入并输出客户端的应答，最后关闭程序[12]。套接字网络通信流程如图 5-11 所示。

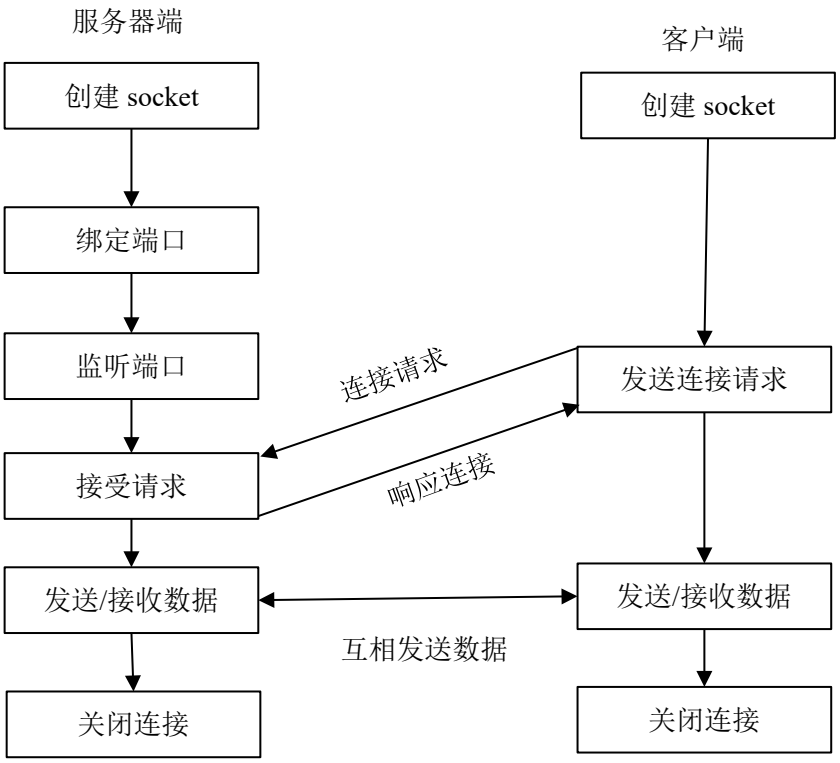

图 5-11 网络通信流程

客户端向服务器端发送的消息是以字节为单位，规定前两个字节代表所对应的操作码，后面的字节是该操作的参数，例如当登录操作时，发送的字节为“10”代表登录操作，发送字节为“11”代表注册操作。当服务器接收到操作码以后，会解析操作码后面的参数，并调用对应的操作函数。发送操作码如表 5-1 所示。

表 5-1 发送操作码

操作码	**功能**
10	登录
11	注册
12	开始匹配
13	停止匹配
14	发送聊天消息
15	主动发起求和
16	主动发起悔棋
17	对方求棋结果
18	对方悔棋结果
19	棋子移动
20	我方输棋

（2）对局匹配

当用户登录成功以后，会进入游戏主界面，主界面会默认绘制一个棋盘用于展示，但并未开始游戏，需要用户通过游戏设置菜单中选择开始匹配按钮，发送消息给服务器。等待服务器发送匹配成功的消息到客户端，此时重绘棋盘，并开始游戏。对局匹配流程如图 5-12 所示

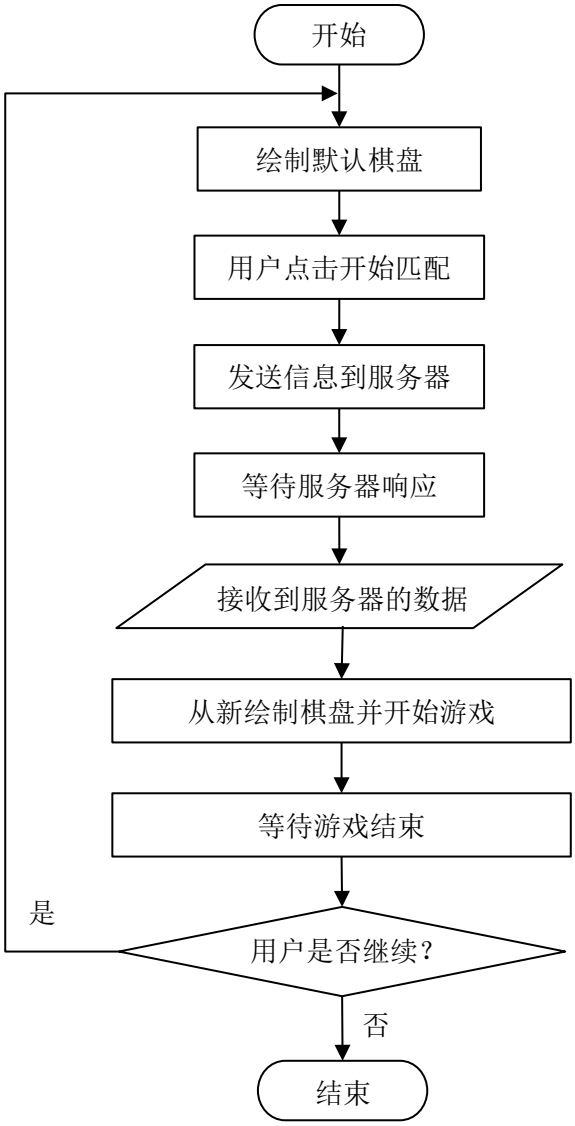

图 5-12 对局匹配流程

（3）棋子移动

棋子的移动需要用户点击棋盘窗口，并捕获用户点击的像素坐标，判断该坐标属于棋盘中的那个区域。要实现棋子移动，需要保存用户点击的最近两个像素坐标点，只有第一个坐标为我方棋子，第二个坐标为空白区域或对方棋子，才能移动棋子，否者不能移动，并将第二坐标清空。棋子类型的判定方法有如下：

① 棋子“兵”的走棋规则

首先判断兵是否过河，如果未过河则只能向前方一格移动，如果已近过河，

则可以向前、左、右移动一个。棋子“兵”的走棋规则如图 5-13 所示，其实现代码见附录 B（1.1 棋子“兵”的走棋规则）。

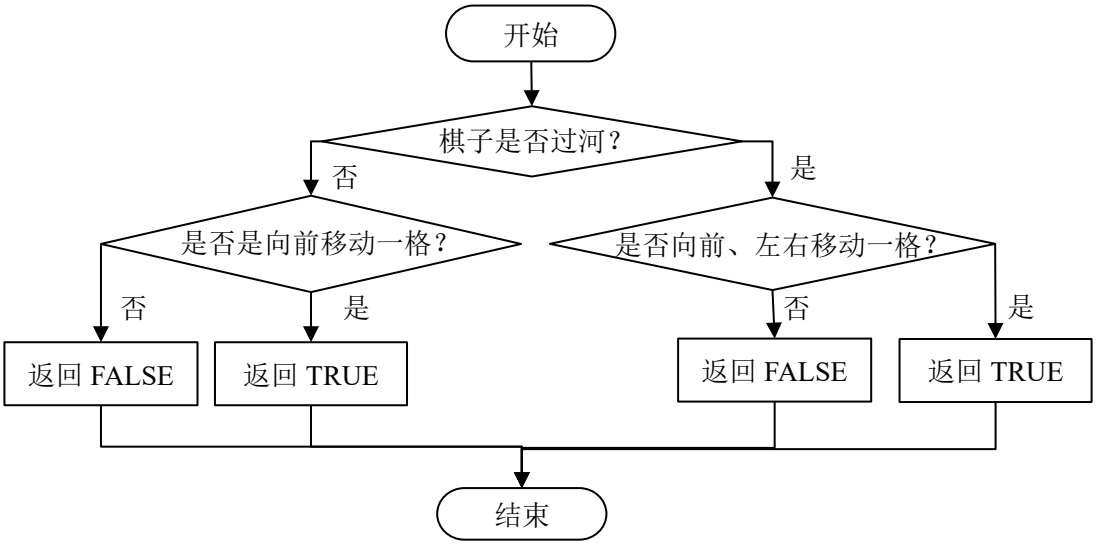

图 5-13 棋子“兵”的走棋规则

② 棋子“车”的走棋规则

首先判断落子点是否与棋子在同行或者同列，再判断落子点与棋子中间是否有其他棋子，若没有棋子则返回 TRUE。棋子“车”的走棋规则如图 5-14 所示，其实现代码见附录 B（1.2 棋子“车”的走棋规则）。

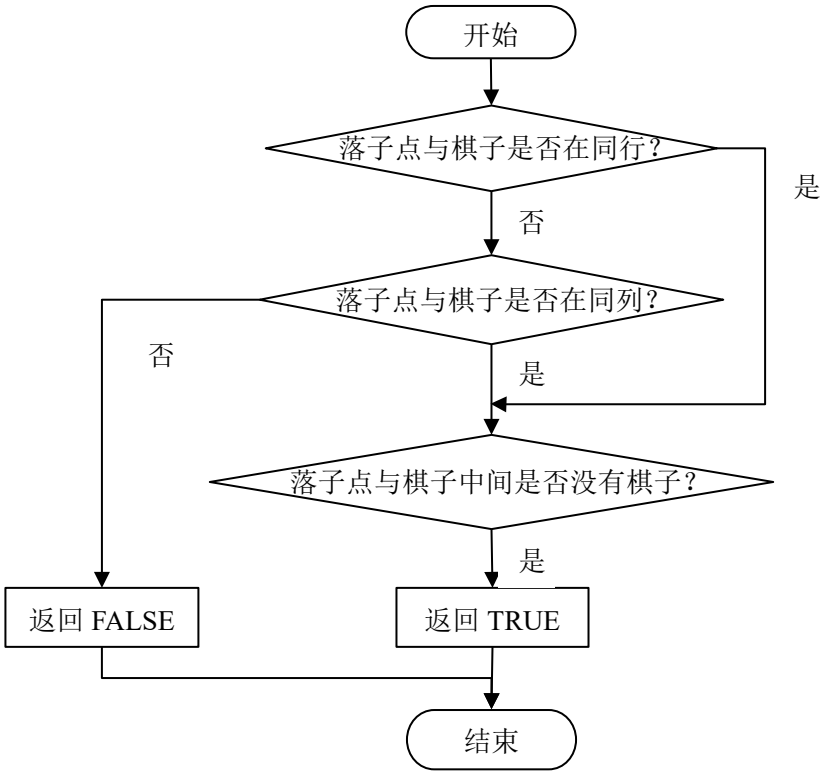

图 5-14 棋子“车”的走棋规则

③ 棋子“马”的走棋规则

算出判断落子点与棋子之间的横纵坐标的差值，会出现四种情况，分别代表与棋子相对位置为上下左右四个方位的各两个落点，判断落子点是否位于这八个点。再通过方位判断的对应方向前一格是否有棋子。棋子“马”的走棋规则如图 5-15 所示，其实现代码见附录 B（1.3 棋子“马”的走棋规则）。

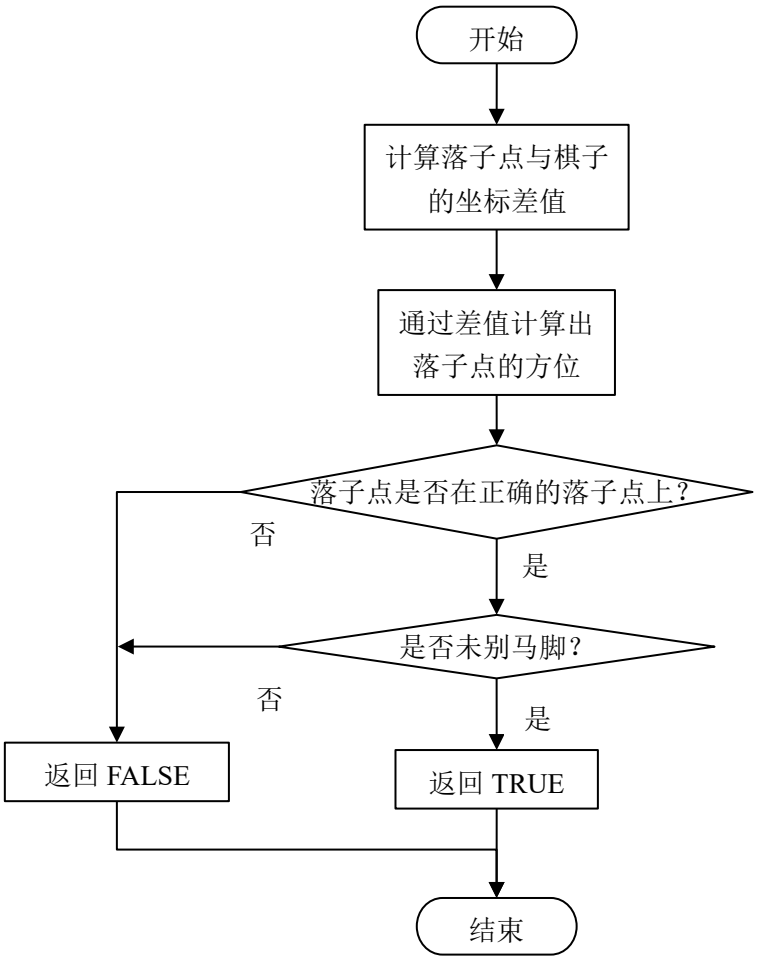

图 5-15 棋子“马”的走棋规则

④ 棋子“象”的走棋规则

先判断落子是否在河对岸，再判断落子点是否为落子点的横纵坐标与棋子横纵坐标的差的绝对值为 2 的点，这一步的目的是找到符合落子规则的点，最后判断与落子点方向相同但横纵坐标距离相差为 1 的点有无棋子，而这一步的目的是判断棋子是否有禁着点，若都满足则返回 TRUE。棋子“象”的走棋规则如图 5-16 所示，其实现代码见附录 B（1.4 棋子“象”的走棋规则）。

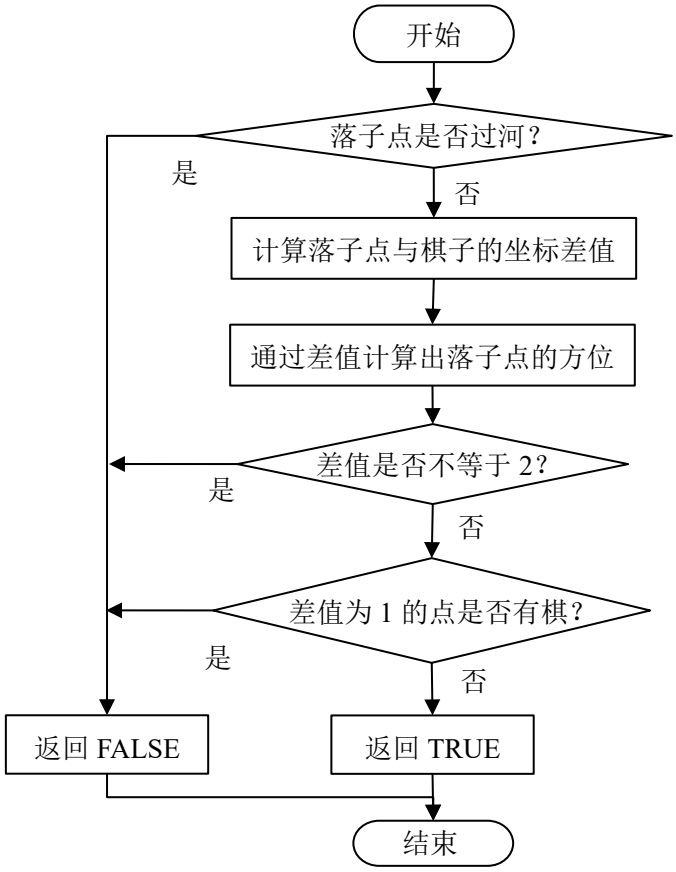

图 5-16 棋子“象”的走棋规则

⑤ 棋子“士”的走棋规则

首先判断落子点是否在皇宫内，再判断落子点与棋子的斜向距离是否为 1，若都满足返回 TRUE。棋子“士”的走棋规则如图 5-17 所示，其实现代码见附录 B（1.5 棋子“士”的走棋规则）。

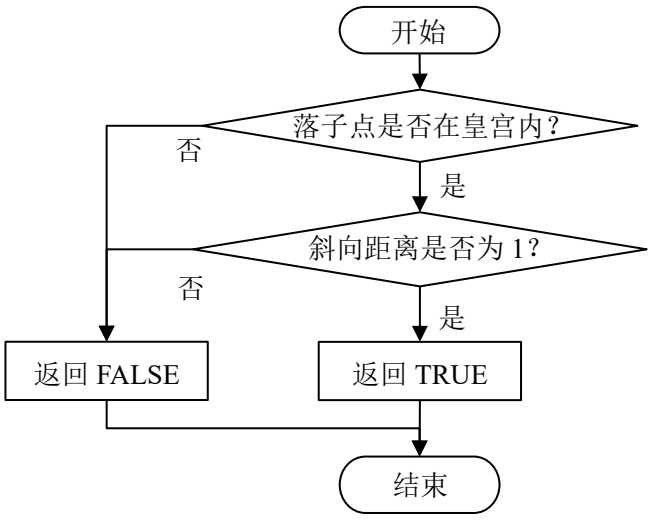

图 5-17 棋子“士”的走棋规则

⑥ 棋子“将”的走棋规则

首先判断落子点是否在皇宫内，再判断落子点距离是否为 1。棋子“将”的走棋规则如图 5-18 所示，其实现代码见附录 B（1.6 棋子“将”的走棋规则）。

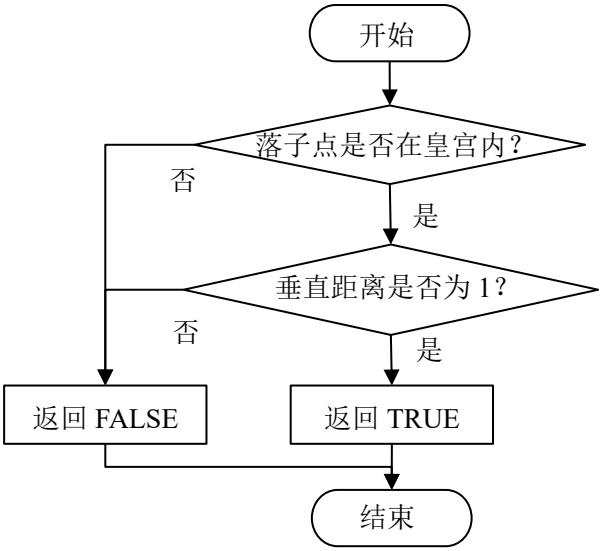

图 5-18 棋子“将”的走棋规则

⑦ 棋子“炮”的走棋规则

首先判断落子点与棋子是否在同行或同列，在判断落子点与棋子中间是否有一个个棋子，若符合则返回 TRUE。棋子“炮”的走棋规则如图 5-19 所示，其实现代码见附录 B（1.7 棋子“炮”的走棋规则）。

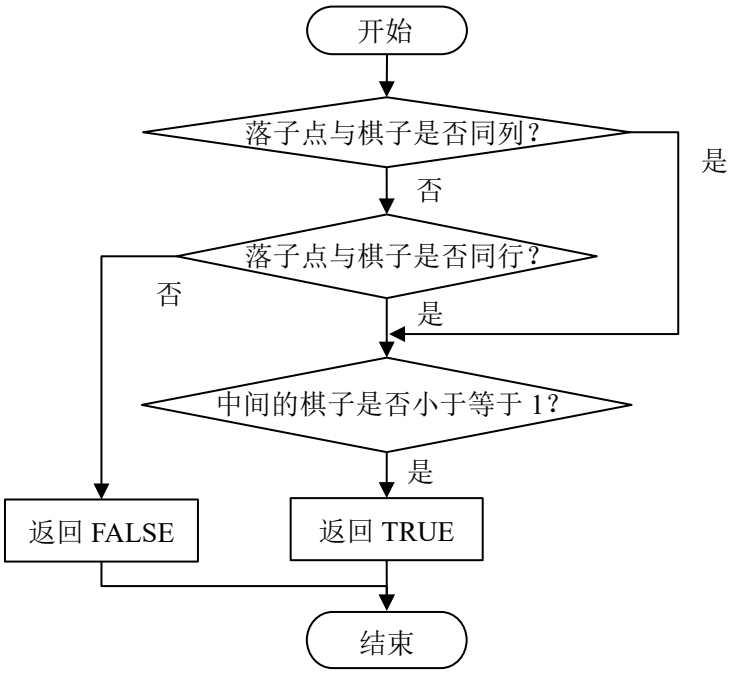

图 5-19 棋子“炮”的走棋规则

2. 界面设计

当用户成功登录以后，会进入游戏主界面，主界面包括左侧的棋盘，右方的网络聊天窗口。左侧棋盘用于象棋对弈，而右方聊天窗口包含两部分，上一部分是现实聊天的窗口，下一部分用于用户输入聊天的内容，通过点击发送按钮即可将对应内容发送出去。

（1）棋盘设计，棋盘有十条横线和九条竖线，规定左上角坐标为（1,1），则右下角的坐标为（10，9），首先绘制棋盘线，再根据棋子的坐标绘制棋子。棋盘如图 5-20 表示。

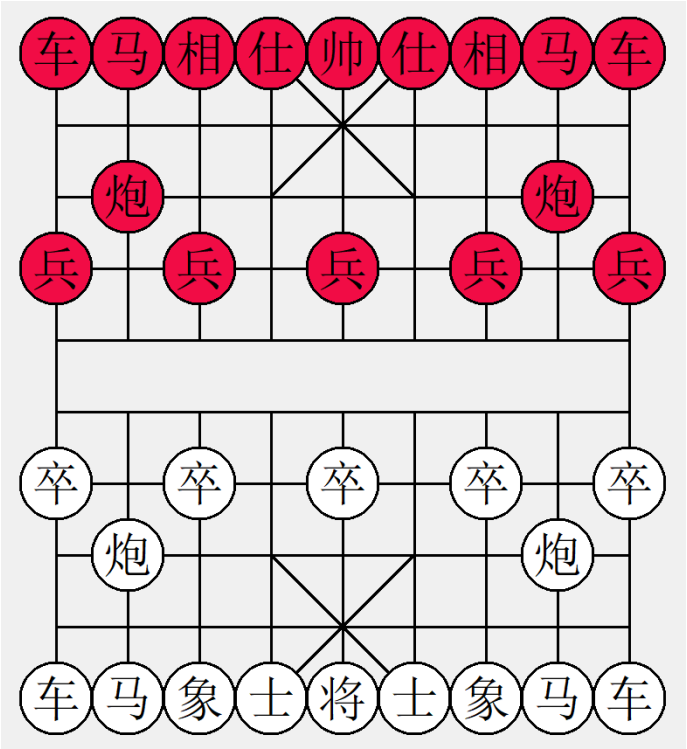

图 5-20 棋盘

5.1.4 对局聊天模块

1. 功能设计

在对局开始后，可以在聊天窗口进行聊天，同时聊天窗口也会显示当前游戏的状态，例如在对局开始时，聊天窗口会提示对局开始，而用户掉线时，会提示对方或自己掉线，并结束游戏。当用户点击发送消息按钮时，检测输入文本框内容是否为空，若不为空，则通过 socket 发送到服务器端。如果游戏结束，清除聊天窗口的消息。消息处理流程如图 5-21 所示。

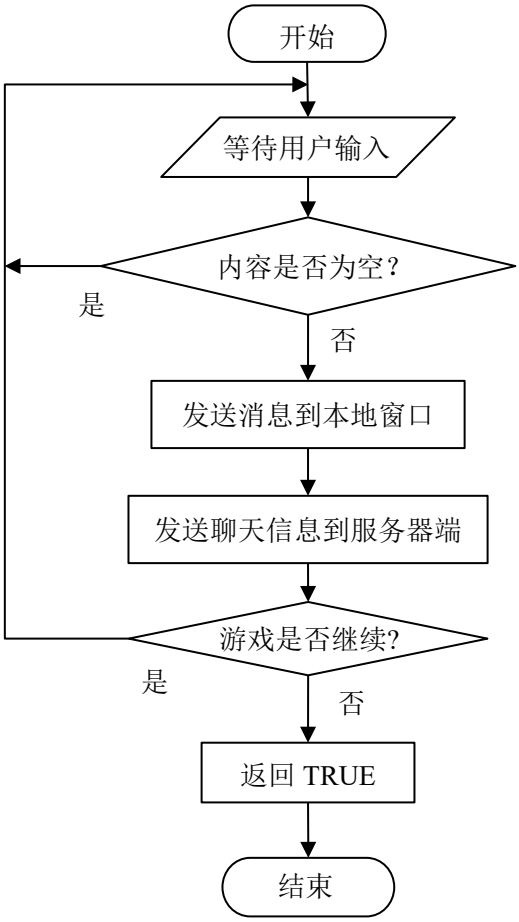

图 5-21 消息处理流程

2. 界面设计

聊天界面包括两部分，一是聊天信息的窗口，二是发送消息的窗口。聊天窗口如图 5-22 所示。

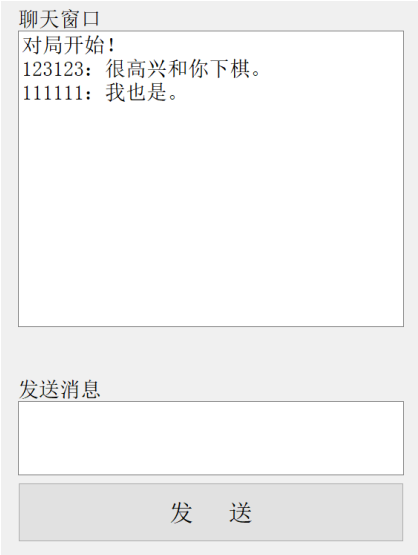

图 5-22 聊天窗口

5.2 服务器端功能模块的实现

5.2.1 网络通信模块

与客户端通信时，每次通信前的 2 个字节用于发送操作码，操作码的作用是方便服务器处理不同的指令，能通过不同的操作码判断用户的不同操作。操作码后面字符为操作的参数。参数的作用是为相对应的槽函数提供必要参数，若缺少参数，操作也会失效。例如登录匹配成功的操作码是“92”，则需要在操作码后面加对局时玩家的先后手规则与对手的 ID 号，再发送给服务器接收操作码。接收操作码如表 5-2 所示。

表 5-2 接收操作码表

操作码	功能
90	登录结果
91	注册结果
92	匹配成功
93	停止匹配成功
94	聊天消息
95	求和结果
96	悔棋结果
97	对方求棋结果
98	对方悔棋结果
99	棋子移动
89	对局结果
88	对方网络断开
87	我方输棋

5.2.2 对局匹配模块

在玩家登录以后，将玩家送到登录队列，在玩家发送匹配指令时，将玩家对应的 Socket 移动到匹配队列。匹配成功的规则是，当匹配队列出现两个玩家时，创建一个新线程，传递两位玩家的 Socket 到游戏队列。传递等待游戏结束以后释放线程，并将两位玩家的 Socket 送到登录队列。当对局结束时，把两个玩家相对应的信息传回登录队列，以便等待用户再次进入匹配队列。对局匹配流程如图 5-23 所示。

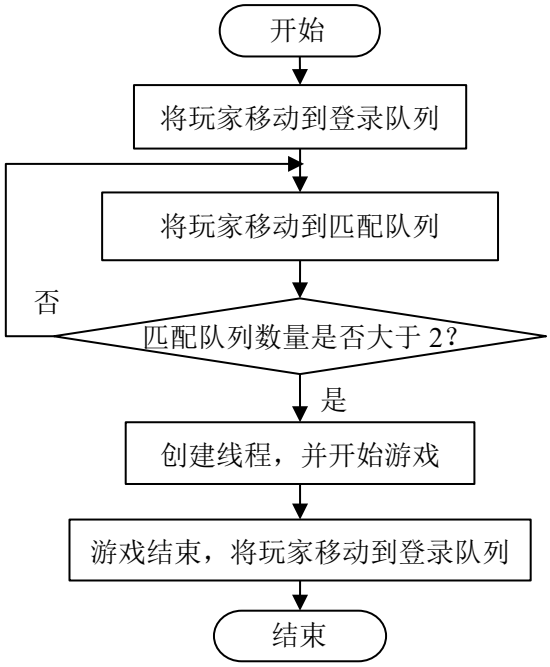

图 5-23 对局匹配流程

5.2.3 登录、注册验证模块

登录与注册的验证功能很相似，都是对数据库进行操作，获取到结果以后根据结果发送到客户端，数据库操作流程如图 5-24 所示。

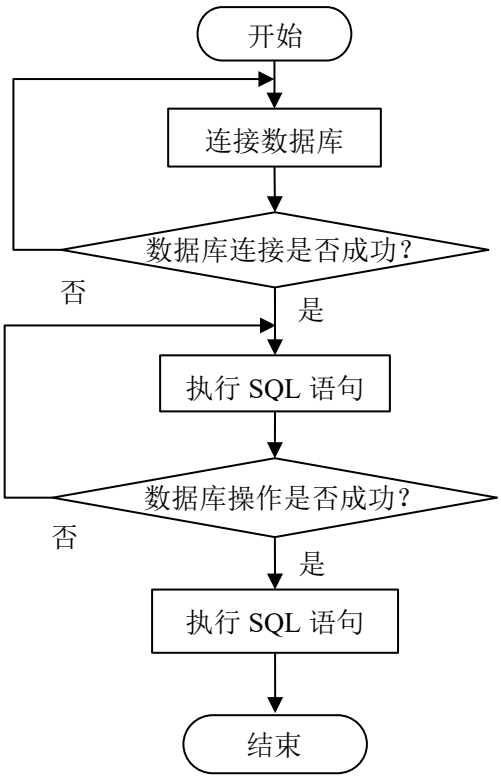

图 5-24 数据库操作流程

第六章 系统测试与维护

系统开发完成以后，还需要对系统进行测试，测试可以保证系统具有一定的稳定性。在计算机软件测试与开发技术研究中，技术人员要做好设计及程序优化等工作，用于提升产品质量及综合性能[13] 。本章将详细的对软件进行测试，保证软件能正常运行。

6.1 测试目的和方法

1. 测试目的

测试阶段的根本目标是尽可能多地发现并排除软件中潜藏的错误，最终把一个高质量的软件系统交给用户使用[14]。同时利用测试过程中得到的测试结果和测试信息，作为后续项目开发的重要输入。

2. 测试方法

程序测试主要分为两类，一类是白盒测试，另一类是黑盒测试。本系统主要采用黑盒测试方法。白盒测试是把测试对象看做一个透明的盒子，它允许测试人员利用程序内部的逻辑结构及有关信息，设计或选择测试用例，对程序所有逻辑路径进行测试。黑盒测试是把测试对象看做一个黑盒子，测试人员完全不考虑程序内部的逻辑结构和内部特性，只依据程序的需求规格说明书，检查程序的功能是否符合它的功能说明。本系统是象棋游戏设计，需要测试的项目有四项，分别是用户登录测试、用户注册测试、棋子规则测试、对局聊天测试。

6.2 测试实例

6.2.1 用户登录测试

测试内容：模拟用户登录，测试当用户登录系统时，在输入密码错误的情况下，是否会弹出警告窗口。

操作：在登录页面输入错误的用户名或密码。

结果：弹窗警告登录错误。

结论：测试成功，当用户输入错误用户名或者错误的密码尝试登陆时，登录会失败，并且会弹出警告窗口，表示账号或者密码输入错误。登录错误提示窗口如图 6-1 所示。

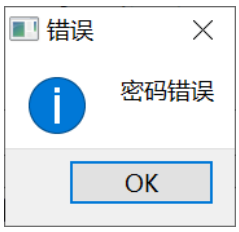

图 6-1 登录错误提示窗口

6.2.2 用户注册测试

测试内容：用户通过登录界面跳转到注册界面时会提示账号密码的格式，当用户正确输入并注册成功后，会有弹窗提示，提示登录成功，当登录成功以后，自动跳转到游戏界面。

操作：在登录界面输入账号、密码、确认密码后点击注册。

结果：成功注册账号并有弹窗提示。

结论：测试成功，当输入六位数的账号与符合规则的密码且账号未被注册的时，注册账号会注册成功，在注册成功时会提示弹窗并自动跳转到游戏界面。注册成功如图 6-2 所示。

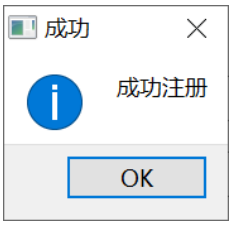

图 6-2 注册成功

6.2.3 棋子规则测试

1. 棋子“炮”的吃棋测试

测试内容：棋子“炮”的吃棋测试，在游戏对局进行的状态下，测试棋子是否按照规则进行吃棋，如果能正常移动，及也能正常吃棋，表示测试成功，间隔多个棋子仍可以移动棋子或者吃棋，则代表测试失败。

操作：游戏开局炮八进七吃马。

结果：能把对方棋子吃掉。

结论：测试成功，棋子能正确的吃棋，当中间间隔多个棋子时，棋子不能移动，棋子吃棋如图 6-3 所示。

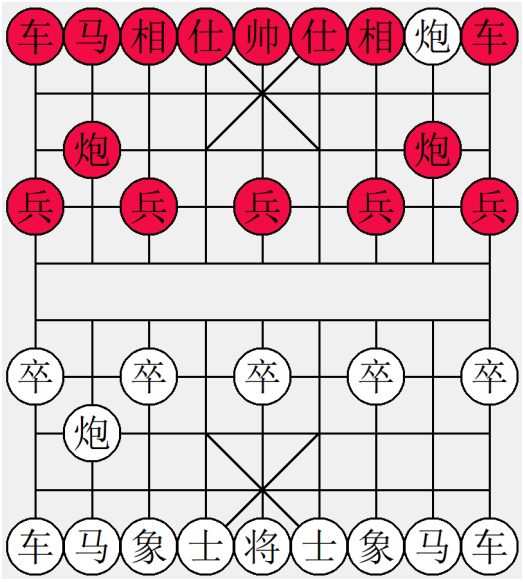

图 6-3 棋子吃棋

2. 棋子“马”的移动测试

测试内容：棋子“马”的走法测试，在游戏对局进行的状态下，测试棋子是否按照规则进行走棋。

操作：游戏开局马八进六。

结果：能把正确移动棋子。

结论：测试成功，棋子能正确的移动，棋子移动如图 6-4 所示。

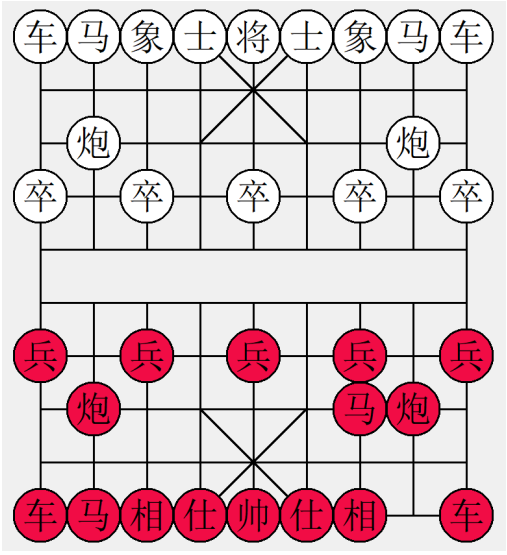

图 6-4 棋子移动

3. 棋子“象”的移动测试

测试内容：棋子“象”的走法测试，在游戏对局进行的状态下，当象移动路径上有棋子时，棋子“象”无法移动。

操作：象三进五。

结果：棋子无法移动。

结论：测试成功，无法移动，棋子未移动如图 6-5 所示。

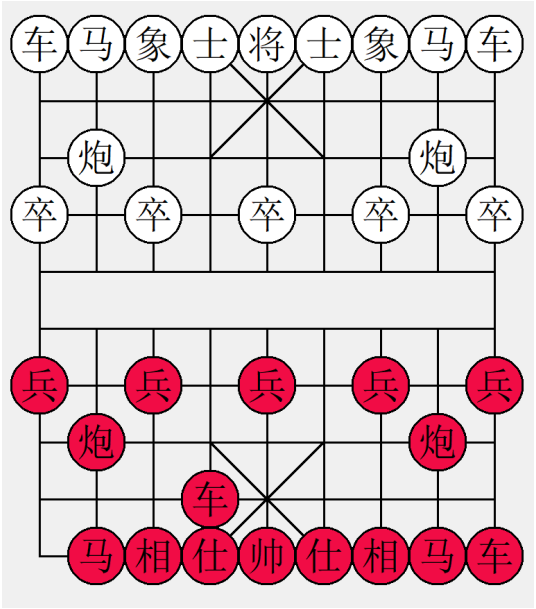

图 6-5 棋子未移动

6.2.4 对局聊天测试

测试内容：在游戏对局进行的状态下，测试聊天功能是否可用。

操作：在输入窗口输入“你好”，观察聊天窗口是否会出现。

结果：聊天窗口出现“你好”二字。

结论：测试成功，对局聊天功能正常，对局聊天如图 6-6 所示。

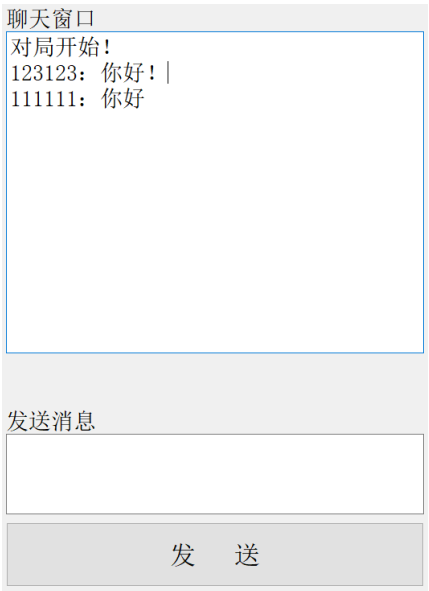

图 6-6 对局聊天

6.3 系统维护

1. 改正性维护

改正性维护是指错误不能通过在开发过程中运行的系统的维护来检查和测试发现。本系统在开发阶段出现 BUG 已及时更改，后期维护工作必须制定计划，进行修改，并且要进行复查和控制。

2. 适应性维护

为了使系统适应环境的变化而进行的维护工作，在技术不断更迭发展，硬件更新周期越来越短，操作系统的更新，运行环境或者外部设备的更新，数据结构和数据格式的更新都要随时对系统进行调整，使之适应应用对象的变化，满足用户的需求。

3. 完善性维护

在系统的基本功能能够满足用户的需求的情况下，随着用户的体验，在某些模块或者功能上做调整或者修改，使系统更能够满足用户的实际体验需求，根据用户的使用习惯做改进和完善，这需要随时跟进和系统更新。对整个系统做好部署，对代码模块化，添加好代码注释，代码的命名遵循命名规则，要有很好的辨识度，这样才能在后期方便系统的维护工作。

4. 预防性维护

为了改进应用软件的可靠性和可维护性，为了适应未来的软硬件环境的变化，应主动增加预防性的新的功能，以使应用系统适应各类变化而不被淘汰。

第七章 结论

7.1 系统完成的功能

本设计是一个以 QT Creator 作为开发工具。系统由客户端、服务器端和数据库三部分构成。客户端使用 QT 框架的 GUI 模块搭建，通过 Network 模块进行套接字通信。服务器端使用 QT 框架的 Network 模块与客户端进行套接字通信。数据库使用 MySQL 数据库来设计，并使用 QT 框架的 SQL 模块与 MySQL 数据库进行数据交换。该系统包括客户端与服务器端两大模块。客户端包括用户登录、用户注册、网络对战、对局聊天四个功能。用户登录功能，用户可以登录游戏系统进行游戏对局。用户注册功能方便用户注册账号，当了账号以后直接进入游戏，不用再次登录。网络对战功能需要匹配对手，在登录状态下，用户点击开始匹配按钮时开始匹配，当匹配到对手以后才能开始游戏。对局聊天功能是在对局开始的状态下，方便与对方沟通。服务器端包括网络通信、对局匹配、登录与注册验证三个功能。网络通信面向多个客户端，使用多线程通信，能够同时服务多个客户端。对局匹配功能，服务器对局聊天功能服务器端使用多线程编程，每一个对局都会有一个线程提供服务。本系统基本实现了中国象棋游戏系统的基本功能，是一个合格的系统。

7.2 系统的不足及展望

本系统虽然完成了象棋游戏的大部分功能，但还有很多待完善的地方，客户端方面，例如在登录界面没有记住密码的功能。在注册界面注册错误时不能清除错误的账号密码。在网络对战时，当断开网络以后，不能重新连接服务器。对局聊天功能不能清除历史聊天记录。服务器端方面，网络通信模块虽然能分辨出客户端发送的命令是哪个用户发出，算法效率太低，算法的效率取决于时间和空间的复杂性[15]。对局匹配模块，匹配规则太简单，无法做出难易区分。登录、注册模块，操作数据库效率太低，查询语句执行效率太低。虽然系统有很多不足，但不影响系统正常使用，希望通过不断地版本迭代，能完善上面提到的诸多不足。

致谢

时光荏苒，四年的大学生活即将划上圆满的句号，而我也将以母校为起点，踏上新的征程。借此机会，感谢这一路走来给予我帮助的老师、朋友和家人们。

首先，我要感谢我的论文指导老师刘小芳老师。从论文的选题到中期检查再到最终论文的定稿，一路走来刘老师都给予了我许多指导和鼓励，也正是因为刘老师严格的要求，才得以让我的论文成为一篇合格的毕业论文。刘老师认真严谨的教学态度、敬业的工作精神都为我日后的工作树立了榜样。

其次，感谢所有四川轻化工大学的老师们，是你们教会了我许多专业知识，让我在专业的学习上迈出了一大步，祝愿所有老师身体健康、万事胜意。同时感谢这篇论文所引用的诸多学者的文献资料，在你们研究的基础上我才能更好的完善丰富我的论文。

还有我的各位同学和室友小伙伴们，感谢你们在这两年的学习和生活中给予我的各种帮助，希望以后我们都能够在各自的领域里面闪闪发光。最后感谢我的家人，谢谢你们给予我物质上的支持和精神上的鼓励，正是因为你们的陪伴，才让我在异地求学的道路上感受到家的温暖。

参考文献

[1] 杜帮国. 基于Java平台的中国象棋游戏的设计与实现[D]. 大连: 大连理工大学, 2013.

[2] 郭建欣. 基于机器视觉的象棋对弈系统研究[D]. 西安: 西安科技大学, 2019.

[3] 刘淑琴, 刘淑英. 基于博弈树搜索算法的中国象棋游戏的设计与实现[J]. 自动化与仪器仪表, 2017, (10): 96-98.

[4] 侯健明, 静国刚, 吴松洋, 等. 基于QT的网络设备拓扑管理平台设计与实现[J]. 工业控制计算机, 2022, 35(01): 87-88.

[5] 王丹. 基于cocos2d_x引擎的中国象棋手机游戏的设计与实现[D]. 长春: 吉林大学, 2018.

[6] 杨潇. 基于QT的富客户端软件设计与实现[D]. 成都: 电子科技大学, 2009.

[7] 陆文周. QT5开发及实例(第4版)[M]. 北京: 电子工业出版社, 2015.

[8] 金繁, 崔培雷. 嵌入式QT中信号与槽机制的研究[J]. 电子设计工程, 2014, 22(24): 168-170.

[9] Garcia-Molina等 美. Hector. 数据库系统实现[M]. 北京: 机械工业出版社, 2010.

[10] 韩雨佟. 基于B/S物联网环境监测系统MySQL数据库的设计与实现[D]. 天津: 天津大学, 2014.

[11] 孙宝林等. Access数据库应用技术[M]. 北京: 清华大学出版社, 2010.

[12] 黄翩, 张琼, 祝婷. 基于QT的一个服务器多个客户端的TCP通信[J]. 电子科技, 2015, 28(03): 76-78.

[13] 张堃. 计算机软件测试技术与开发应用策略分析[J]. 电子技术与软件工程, 2021, (23): 24-25.

[14] 张海藩, 牟永敏. 软件工程导论(第6版)[M]. 北京: 清华大学出版社, 2013.

[15] Mohanty S N, Tripathy P K. Data Structure and Algorithms Using C++: A Practical Implementation[M]. John Wiley & Sons, Inc., 2021.

附 录

附录 A：系统使用说明书

1. 系统概述

本系统现了中国象棋游戏的基本功能，基本功能包括用户登录、网络对战、局域网对战等。

1.1 功能

登录游戏后可进行游戏，可以随时退出游戏。

1.2 原理

通过后端服务，对数据进行处理与传递，实现多客户端通信。

2. 软件安装

本系统提供安装包，用户只需要安装软件即可运行游戏。服务器端需要配置 MySQL8 数据库。

2.1 系统要求

系统要求最好是 Windows 7 系统及以上的笔记本或者台式电脑或者 linux 的 centos 7 系统的服务器以及 macOS 操作系统；4GB 运行内存。

2.2 数据库支持

数据支持版本是 MySQL8.x，数据库建表服务软件是 Navicat for MySQL，连接端口号是 3306，数据库用户连接是 liuyu 连接，数据库名称是 chess。

3. 运行说明

后端服务器首先安装好数据库，再通过 Navicat for MySQL 连接到数据库，初始化数据库代码见附录 B(2)，可使用本文提供的 SQL 语句即可创建好数据库，并启动服务器端软件即可正常运行。

4. 服务与支持

本系统提供 SQL 语句，方便配置后端数据库。

附录 B：主要源代码

1. 棋子走棋规则代码

1.1 棋子“兵”的走棋规则

```
// ----------------------------------------------------------------------
bool board::bing()
{
    if(saveFirstPoint.x() < 6)
    {
        if(((saveSecondPoint.x() == saveFirstPoint.x() - 1) && (saveSecondPoint.y() == saveFirstPoint.y())) ||
            ((saveSecondPoint.x() == saveFirstPoint.x()) && (saveSecondPoint.y() == saveFirstPoint.y() - 1)) ||
            ((saveSecondPoint.x() == saveFirstPoint.x()) && (saveSecondPoint.y() == saveFirstPoint.y() + 1))
           ) return true;
        else return false;
    }
    else
    {
        if((saveSecondPoint.x() == saveFirstPoint.x() - 1) && (saveSecondPoint.y() == saveFirstPoint.y())) return true;
        else return false;
    }
}
// ----------------------------------------------------------------------
```

1.2 棋子“车”的走棋规则

```
// ----------------------------------------------------------------------
bool board::che()
{
    if(saveSecondPoint.x() != saveFirstPoint.x() && saveSecondPoint.y() == saveFirstPoint.y())
    {
        int i = 1;
        for(; i <= 32; i++)
        {
            if(s[i].dead == true || (s[i].row == saveFirstPoint.x() && s[i].col ==
```

```
saveFirstPoint.y()))
                {
                    continue;
                }
                if((saveSecondPoint.x() - saveFirstPoint.x()) < 0)
                {
                    if(s[i].col == saveFirstPoint.y() && s[i].row < saveFirstPoint.x() && s[i].row > saveSecondPoint.x())
                    {
                        qDebug() << "上下的中间有棋 1";
                        break;
                    }
                }else
                {
                    if(s[i].col == saveFirstPoint.y() && s[i].row > saveFirstPoint.x() && s[i].row < saveSecondPoint.x())
                    {
                        qDebug() << "上下的中间有棋 1";
                        break;
                    }
                }
            }
            if(i > 32) return true;
            else return false;
        }
        else if(saveSecondPoint.x() == saveFirstPoint.x() && saveSecondPoint.y() != saveFirstPoint.y())
        {
            int i = 1;
            for(; i <= 32; i++)
            {
                if(s[i].dead == true || (s[i].row == saveFirstPoint.x() && s[i].col == saveFirstPoint.y()))
                {
                    continue;
                }
                if((saveSecondPoint.y() - saveFirstPoint.y()) < 0)
                {
```

```
                    qDebug() << "saveSecondPoint.y() = " << saveSecondPoint.y() << "s[i].col = "
<< s[i].col << "saveFirstPoint.y()=" << saveFirstPoint.y();
                    if(s[i].row == saveFirstPoint.x() && s[i].col < saveFirstPoint.y() && s[i].col >
saveSecondPoint.y())
                    {
                        qDebug() << "左右中间有棋 1";
                        break;
                    }
                }else
                {
                    qDebug()<< "saveFirstPoint.y()=" << saveFirstPoint.y()<< "s[i].col = " <<
s[i].col   << "saveSecondPoint.y() = " << saveSecondPoint.y() ;
                    if(s[i].row == saveFirstPoint.x() && s[i].col > saveFirstPoint.y() && s[i].col <
saveSecondPoint.y())
                    {
                        qDebug() << "左右中间有棋 2";
                        break;
                    }
                }
            }
            if(i > 32) return true;
            else return false;
        }else
        {
            return false;
        }
}
// ----------------------------------------------------------------------
1.3 棋子“马”的走棋规则
// ----------------------------------------------------------------------
bool board::ma()
{
    int tepm1 = saveSecondPoint.x() - saveFirstPoint.x();
    int tepm2 = saveSecondPoint.y() - saveFirstPoint.y();
    if((tepm1 == -2 && tepm2 == -1) || (tepm1 == -2 && tepm2 == 1))
    {
        int i = 1;
```

```
        for(; i <= 32; i++)
        {
            if(s[i].dead == true || (s[i].row == saveFirstPoint.x() && s[i].col == saveFirstPoint.y()))
            {
                continue;
            }
            if(s[i].row == saveFirstPoint.x() - 1 && s[i].col == saveFirstPoint.y())
            {
                qDebug() << "上";
                break;
            }
        }
        if(i > 32) return true;
        else return false;
    }
    if((tepm1 == -1 && tepm2 == 2) || (tepm1 == 1 && tepm2 == 2))
    {
        int i = 1;
        for(; i <= 32; i++)
        {
            if(s[i].dead == true || (s[i].row == saveFirstPoint.x() && s[i].col == saveFirstPoint.y()))
            {
                continue;
            }
            if(s[i].row == saveFirstPoint.x() && s[i].col == saveFirstPoint.y() + 1)
            {
                qDebug() << "右";
                break;
            }
        }
        if(i > 32) return true;
        else return false;
    }
    if((tepm1 == 2 && tepm2 == -1) || (tepm1 == 2 && tepm2 == 1))
    {
        int i = 1;
```

```
        for(; i <= 32; i++)
        {
            if(s[i].dead == true || (s[i].row == saveFirstPoint.x() && s[i].col == saveFirstPoint.y()))
            {
                continue;
            }
            if(s[i].row == saveFirstPoint.x() + 1 && s[i].col == saveFirstPoint.y())
            {
                qDebug() << "下";
                break;
            }
        }
        if(i > 32) return true;
        else return false;
    }
    if((tepm1 == -1 && tepm2 == -2) || (tepm1 == 1 && tepm2 == -2))
    {
        int i = 1;
        for(; i <= 32; i++)
        {
            if(s[i].dead == true || (s[i].row == saveFirstPoint.x() && s[i].col == saveFirstPoint.y()))
            {
                continue;
            }
            if(s[i].row == saveFirstPoint.x() && s[i].col == saveFirstPoint.y() - 1)
            {
                qDebug() << "左";
                break;
            }
        }
        if(i > 32) return true;
        else return false;
    }
    return false;
}
```

```
// -------------------------------------------------------------------------
1.4 棋子“象”的走棋规则
// -------------------------------------------------------------------------
bool board::xiang()
{
    if(saveSecondPoint.x() < 6) return false;
    int tepm1 = saveSecondPoint.x() - saveFirstPoint.x();
    int tepm2 = saveSecondPoint.y() - saveFirstPoint.y();
    if((tepm1 == -2 && tepm2 == -2))
    {
        int i = 1;
        for(; i <= 32; i++)
        {
            if(s[i].dead == true || (s[i].row == saveFirstPoint.x() && s[i].col ==
saveFirstPoint.y()))
            {
                continue;
            }
            if(s[i].row == saveFirstPoint.x() - 1 && s[i].col == saveFirstPoint.y() - 1)
            {
                qDebug() << "左上";
                break;
            }
        }
        if(i > 32) return true;
        else return false;
    }
    if((tepm1 == -2 && tepm2 == 2))
    {
        int i = 1;
        for(; i <= 32; i++)
        {
            if(s[i].dead == true || (s[i].row == saveFirstPoint.x() && s[i].col ==
saveFirstPoint.y()))
            {
                continue;
            }
            if(s[i].row == saveFirstPoint.x() - 1 && s[i].col == saveFirstPoint.y() + 1)
```

```
                {
                    qDebug() << "右上";
                    break;
                }
            }
            if(i > 32) return true;
            else return false;
        }
        if((tepm1 == 2 && tepm2 == -2))
        {
            int i = 1;
            for(; i <= 32; i++)
            {
                if(s[i].dead == true || (s[i].row == saveFirstPoint.x() && s[i].col == saveFirstPoint.y()))
                {
                    continue;
                }
                if(s[i].row == saveFirstPoint.x() + 1 && s[i].col == saveFirstPoint.y() - 1)
                {
                    qDebug() << "左下";
                    break;
                }
            }
            if(i > 32) return true;
            else return false;
        }
        if((tepm1 == 2 && tepm2 == 2))
        {
            int i = 1;
            for(; i <= 32; i++)
            {
                if(s[i].dead == true || (s[i].row == saveFirstPoint.x() && s[i].col == saveFirstPoint.y()))
                {
                    continue;
                }
```

```
                if(s[i].row == saveFirstPoint.x() + 1 && s[i].col == saveFirstPoint.y() + 1)
                {
                    qDebug() << "右下";
                    break;
                }
            }
            if(i > 32) return true;
            else return false;
        }
        return false;
}
// -------------------------------------------------------------------------
1.5 棋子“士”的走棋规则
// -------------------------------------------------------------------------
bool board::shi()
{
    if((saveSecondPoint.x() < 8 || saveSecondPoint.x() > 10) || (saveSecondPoint.y() < 4 ||
saveSecondPoint.y() > 6))
    {
        return false;
    }
    int tepm1 = saveSecondPoint.x() - saveFirstPoint.x();
    int tepm2 = saveSecondPoint.y() - saveFirstPoint.y();
    if(tepm1 == -1 && tepm2 == -1)return true;
    if(tepm1 == -1 && tepm2 == 1)return true;
    if(tepm1 == 1 && tepm2 == -1)return true;
    if(tepm1 == 1 && tepm2 == 1)return true;
    return false;
}
// -------------------------------------------------------------------------
1.6 棋子“将”的走棋规则
// -------------------------------------------------------------------------
bool board::king()
{
    int num = 1;
    // 判断落子点是否在与棋子距离为 1 的地方
    if(! ((saveSecondPoint.x() == saveFirstPoint.x()+num && saveSecondPoint.y() ==
saveFirstPoint.y()) ||
```

```
        (saveSecondPoint.x() == saveFirstPoint.x()-num && saveSecondPoint.y() == saveFirstPoint.y()) ||
        (saveSecondPoint.y() == saveFirstPoint.y()+num && saveSecondPoint.x() == saveFirstPoint.x()) ||
        (saveSecondPoint.y() == saveFirstPoint.y()-num && saveSecondPoint.x() == saveFirstPoint.x())
       ))
    {
        qDebug() << "不在四周！(" << saveFirstPoint.x() << saveFirstPoint.y() << ")(" << saveSecondPoint.x() << saveSecondPoint.y() << ")";
        return false;
    }
    // 判断落子点是否在皇宫内
    if((saveSecondPoint.x() < 8 || saveSecondPoint.x() > 10) || (saveSecondPoint.y() < 4 || saveSecondPoint.y() > 6))
    {
        qDebug() << "不在圈内！";
        qDebug() << saveSecondPoint.x() << saveSecondPoint.y();
        return false;
    }
    return true;
}
// ----------------------------------------------------------------------
```

1.7 棋子“炮”的走棋规则

```
// ----------------------------------------------------------------------
bool board::pao()
{
    int num = 0;
    if(saveSecondPoint.x() != saveFirstPoint.x() && saveSecondPoint.y() == saveFirstPoint.y())
    {
        int i = 1;
        for(; i <= 32; i++)
        {
            if(s[i].dead == true || (s[i].row == saveFirstPoint.x() && s[i].col == saveFirstPoint.y()))
            {
                continue;
```

```
            }
            if((saveSecondPoint.x() - saveFirstPoint.x()) < 0)
            {
                if(s[i].col == saveFirstPoint.y() && s[i].row < saveFirstPoint.x() && s[i].row > saveSecondPoint.x())
                {
                    qDebug() << "num++";
                    num++;
                }
            }else
            {
                if(s[i].col == saveFirstPoint.y() && s[i].row > saveFirstPoint.x() && s[i].row < saveSecondPoint.x())
                {
                    qDebug() << "num++";
                    num++;
                }
            }
        }
        qDebug() << "num = " << num;
        if(num >= 2) return false;
        if(num == 1)
        {
            for(int j = 1; j <= 32; j++)
            {
                if(s[j].dead == true || (s[j].row == saveFirstPoint.x() && s[j].col == saveFirstPoint.y()))
                {
                    continue;
                }
                if(s[j].row == saveSecondPoint.x() && s[j].col == saveSecondPoint.y()) return true;
            }
        }
        if(num == 0)
        {
            int j = 1;
            for(; j <= 32; j++)
```

```
            {
                if(s[j].dead == true || (s[j].row == saveFirstPoint.x() && s[j].col == saveFirstPoint.y()))
                {
                    continue;
                }
                if(s[j].row == saveSecondPoint.x() && s[j].col == saveSecondPoint.y()) break;
            }
            if(j > 32) return true;
            else return false;
        }

    }
    else if(saveSecondPoint.x() == saveFirstPoint.x() && saveSecondPoint.y() != saveFirstPoint.y())
    {
        int i = 1;
        for(; i <= 32; i++)
        {
            if(s[i].dead == true || (s[i].row == saveFirstPoint.x() && s[i].col == saveFirstPoint.y()))
            {
                continue;
            }
            if((saveSecondPoint.y() - saveFirstPoint.y()) < 0)
            {
                if(s[i].row == saveFirstPoint.x() && s[i].col < saveFirstPoint.y() && s[i].col > saveSecondPoint.y())
                {
                    num++;
                }

            }else
            {
                //qDebug()<< "saveFirstPoint.y()=" << saveFirstPoint.y()<< "s[i].col = " << s[i].col   << "saveSecondPoint.y() = " << saveSecondPoint.y() ;
                if(s[i].row == saveFirstPoint.x() && s[i].col > saveFirstPoint.y() && s[i].col <
```

```
saveSecondPoint.y())
                    {
                        num++;
                    }
                }
            }
            qDebug() << "num = " << num;
            if(num >= 2) return false;
            if(num == 1)
            {
                for(int j = 1; j <= 32; j++)
                {
                    if(s[j].dead == true || (s[j].row == saveFirstPoint.x() && s[j].col == saveFirstPoint.y()))
                    {
                        continue;
                    }
                    if(s[j].row == saveSecondPoint.x() && s[j].col == saveSecondPoint.y()) return true;
                }
            }
            if(num == 0)
            {
                int j = 1;
                for(; j <= 32; j++)
                {
                    if(s[j].dead == true || (s[j].row == saveFirstPoint.x() && s[j].col == saveFirstPoint.y()))
                    {
                        continue;
                    }
                    if(s[j].row == saveSecondPoint.x() && s[j].col == saveSecondPoint.y()) break;
                }
                if(j > 32) return true;
                else return false;
            }
        }else
        {
```

```
            return false;
        }
}
// -----------------------------------------------------------------------
```

2. 创建数据库的 SQL 语句

```
-- Table structure for table `user`
--

DROP TABLE IF EXISTS `user`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `user` (
  `id` int unsigned NOT NULL AUTO_INCREMENT,
  `passwd` varchar(20) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=123456 DEFAULT CHARSET=utf8mb3;
/*!40101 SET character_set_client = @saved_cs_client */;
--
-- Dumping data for table `user`
--
LOCK TABLES `user` WRITE;
/*!40000 ALTER TABLE `user` DISABLE KEYS */;
INSERT INTO `user` VALUES
(100000,'111111111a'),(100001,'dUOdc7m3cqZccDoJ'),(100002,'gjZLwQAmTpd4SYHM'),(1000
03,'QGSQQk21RwhogqmM'),(100004,'z26CrnlOFeBvOj3V'),(100005,'PA781bOTM62hCxXs'),(
100006,'g8qEvtlTJ4czMNDJ'),(100007,'tHDlH7YsN6JDbUcg'),(100008,'OQSMyXsJ561HBJvx'),
(100009,'8JWqeLWyeXmI7vc6'),(100010,'4UQko8ii1AKQmEzU'),(100011,'asdfasdfa'),(100234,'
QGSQQk21RwhogqmM'),(111111,'111111'),(112312,'123123'),(123123,'123123'),(123125,'12312
5'),(123455,'123123');
/*!40000 ALTER TABLE `user` ENABLE KEYS */;
UNLOCK TABLES;
--
-- Dumping routines for database 'chess'
--
/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;
/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
```

```
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */;
/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;
```