目 录

[1 课题一：七子棋](#_Toc67143263)

[1.1课题功能](#_Toc67143264)

[1.2 开发平台](#_Toc67143265)

[1.3 总体方案](#_Toc67143266)

[1.3.1 总体流程图](#_Toc67143267)

[1.3.2 相关数据结构](#_Toc67143268)

[1.4 详细设计](#_Toc67143269)

[1.4.1函数](#_Toc67143270)

[1.4.2 下棋函数](#_Toc67143271)

[1.4.3 判断输赢和禁手函数](#_Toc67143272)

[1.4.4 AI评分函数](#_Toc67143273)

1.4.5 AI下棋函…………………………………………………………………………………

1.4.6更新棋盘函数……………………………………………………………………………

[1.5 测试与运行](#_Toc67143275)

[1.6 调试记录](#_Toc67143276)

[2 心得体会](#_Toc67143278)

[3毕业要求达成度分析](#_Toc67143279)

# 1 课题一：七子棋

## 1.1课题功能

（1）Qt编程实现server程序七子棋游戏界面，20\*20格子

（2）编程实现server本地端的鼠标点击依次放置黑白棋子，实现双人手动下棋

（3）编程实现自动判别下棋的合法性,自动判别五五禁手与六六禁手及长连

（4）编程实现自动判别盘面输赢或者和棋

（5）编程实现倒计时功能，超时自动判输

（6）编程实现悔棋，重新开始，棋盘更新功能

（7）编程实现server端的agent，实现本地人机、机机下棋

（8）编程实现client程序，与server端实时通信获取当前棋面信息

（9）编程实现server端的异步交互机制，client端自主选择下棋先后手的顺序机制

（10）在client中添加agent，实现1个agent连接与server端的agent进行对抗。

## 1.2 开发平台

所使用的开发环境：Windows10操作系统、qtcreator

## 1.3 总体方案

### 1.3.1 总体流程图

程序总体思路，总的流程图或框图

**游戏主界面：七子棋**

**开始游戏**

**菜单栏模式选择**

**双人对战模式**

**人机对战模式**

**机机对战模式**

**网络对战模式**

**点击开始**

**黑棋在中心点下第一颗子，更新棋盘**

**白棋判断棋盘各点得分**

**随机选择得分最高的一点下棋**

**更新棋盘，判断是否产生禁手或输赢**

**是**

**弹出提示：游戏结果，清空棋盘**

**否**

**黑/白棋依次判断**

**鼠标点击棋盘下棋，捕获鼠标位置**

**判断下棋合法性**

**更新棋盘，更换棋子颜色**

**判断是否产生禁手或输赢**

**是**

**弹出提示：游戏结果，清空棋盘**

**否**

**下棋超时**

**玩家点击鼠标下棋，捕获鼠标位置**

**判断下棋合法性**

**更新棋盘**

**判断是否产生禁手或输赢**

**是**

**弹出提示：游戏结果，清空棋盘**

**否**

**否**

**ai**

**AI下棋**

**服务端开启监听，客户端开启连接**

**连接成功**

**客户端选择棋子颜色（黑色）**

**失败报错**

**客户端落子，向服务端发送棋盘信息**

**服务端接收信息，更新棋盘，判断客户端落子结果**

**产生禁手或输赢**

**是**

**向客户端发出输赢信号，弹出提示：游戏结束，清空棋盘**

**服务端落子，判断服务端落子结果**

**否**

**是**

**向客户端发送棋盘信息，更新棋盘**

**否**

### 1.3.2 相关数据结构

程序设计中所使用的数据结构：类、结构体，字符串数组等

## 1.4 详细设计

### 1.4.1函数

|  |  |
| --- | --- |
| **函数名称** | **函数功能** |
| **void** placepiece**(int** x**,int** y**,int** p**)** | 下棋函数，将棋盘对应位置设为相对应黑白棋的Id，并存储已下的棋子颜色及位置 |
| **int** showpiece**(int** x**,int** y**)** | 展示指定点棋子的playerId |
| **void** setundoqz**(int** x**,int** y**)** | （悔棋函数用）悔棋后将对应位置设置playerId为-1 |
| **void** undo**(int** p**)** | 悔棋函数 |
| **int** getbnum**()**  **int** getwnum**()** | 获取棋盘上黑白棋的数量 |
| **bool** judgedraw**(checkerboard** board**)** | 判断和棋 |
| **bool** judgewin**(checkerboard** board**,int** x**,int** y**)** | 判断输赢、禁手 |
| **void** jugdevalue**(checkerboard \***board**)** | AI评分函函数，对棋盘上的每个点进行评分 |
| **void** showTimelimit1**();**  **void** showTimelimit2**();**  **void** showNettimelimit**()**; | 倒计时槽函数，调用启动倒计时函数后，开始倒计时，并判断是否超时； |
| **void** triggeredNormal**();**  **void** triggeredAI**();**  **void** triggeredai**();**  **void** triggeredNet**()**; | 模式选择槽函数，显示或隐藏对应的按钮 |
| **void** timer1**()**  **void** timer2**()**  **void** nettimer**()** | 倒计时函数，启动倒计时 |
| **void** *paintEvent***(QPaintEvent \***event**)** | 绘图函数，绘制游戏主界面，棋盘，实时绘制棋盘上的棋子 |
| **void** *mouseReleaseEvent***(QMouseEvent \***event**)** | 鼠标点击释放函数，获取鼠标事件发生时鼠标相对于窗口的位置，若双人，人机，网络模式，  根据情况调用aiplacepiece()和placepiece()函数，并更新棋盘，调用 judgedraw() 和 judgewin()函数判断结局发送结果信息 |
| **voidMainWindow::**on\_pushButton\_ai\_clicked**()** | 机机对战模式开始按钮槽函数，在中心点下第一颗黑子 |
| **void MainWindow::**on\_stop\_clicked**()** | 机机对战模式结束按钮槽函数，清空棋盘 |
| **voidMainWindow::***contextMenuEvent***(QContextMenuEvent \***event**)** | 让菜单栏显示在鼠标所在位置，并在游戏开始后不再显示模式选择菜单栏 |
| **void MainWindow::**aiplacepiece**()** | AI下棋槽函数，调用judgevalue（）函数评分，随机选择评分最高的一点下棋并更新棋盘，调用 judgedraw() 和 judgewin()函数判断结局发送结果信息 |
| **void MainWindow::**renew**()** | 更新棋盘，全部初始化 |
| **void MainWindow::**slogRecv**()** | 网络对战接收棋盘信息  服务端接收客户端信息，并判断客户端落子后是否产生禁手或赢棋等，更新界面；  客户端接收服务端信息，并判断是否有输赢等信息，更新棋盘； |
| **void MainWindow::**on\_pushButton\_restart\_clicked**()** | 客户端重新开始按钮槽函数，清空棋盘 |

### 1.4.2 下棋函数

void **placepiece**(int **x**,int **y**,int **p**)

是

根据鼠标位置，判断是否在棋盘内以及棋盘上相应位置否为空

qz[x][y].playerId(p)；

value[x][y]=0;

将棋子放入棋子数组中，并将该点评分设为0

将棋子放入对应的路径数组

### 1.4.3 判断输赢和禁手函数（朴素遍历法）

bool **judgewin**(checkerboard **board**,int **x**,int **y**)

设置八个方向的单位向量

上下，左右，左上右下，右上左下

int directionx[8]={-1,1,0,0,-1,1,-1,1};

int directiony[8]={0,0,1,-1,-1,1,1,-1};

以棋盘上最后落下的一颗子为中心，改变x,y，遇到同色棋子num[i]++，遇到异色棋子，结束此方向搜索,改变方向

八个方向全部遍历完后，标记可能产生的五五禁手和六六禁手的方向

**否**

**是**

判断禁手，根据标记的方向，判断活五和活六的个数count

return true

判断是否存在num[i]==7

If (count>=2),FLAG=1 return false

If (count<2) return false

### 1.4.4 AI评分函数

void **judgevalue**(checkerboard \***board**)

黑子

白子

**遍历棋盘上的每一颗棋子**

以该棋子为中心遍历八个方向

以该棋子为中心遍历八个方向

对每条线上靠近该黑子的0/1/2个空格进行加分（空格数量取决于紧邻黑子的是白子还是空格）

对每条线上靠近该白子的0/1/2个空格进行加分（空格数量取决于紧邻白子的是黑子还是空格）

评分函数加分标准

|  |  |  |
| --- | --- | --- |
| **类型** | **黑子** | **白子（人机ai）** |
| **单棋** | +10 | +10 |
| **死二** | +10 | +15 |
| **活二** | +20 | +20 |
| **死三** | +30 | +30 |
| **活三** | +50 | +50 |
| **死四** | +50 | +200 |
| **活四** | +1000 | +2000 |
| **死五** | +1000 | +2000 |
| **活五** | +5000 | +10000 |
| **六连子** | +10000 | +20000 |

### 1.4.5 AI下棋函

void MainWindow::**aiplacepiece**()

调用judgevalue函数对棋盘每一点进行评分

遍历得分数组，寻找评分最大值

记录所有评分为最大值的点的坐标

随机选择评分最高的一点下棋并更新棋盘

int **n**=(rand()%(k));

board.placepiece(maxvalue[n].x,maxvalue[n].y,p);

判断胜负，弹出提示

### 1.4.6 更新棋盘函数

void MainWindow::**renew**()

黑白棋子路径数组中的棋子数是否都为零

调用悔棋函数，设置当前棋子为空，棋子数减一

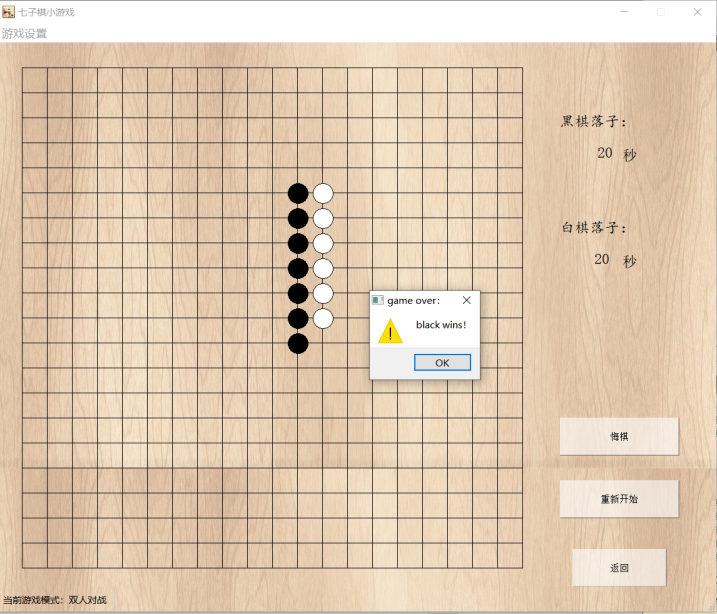
初始化评分数组，更新棋盘

**是**

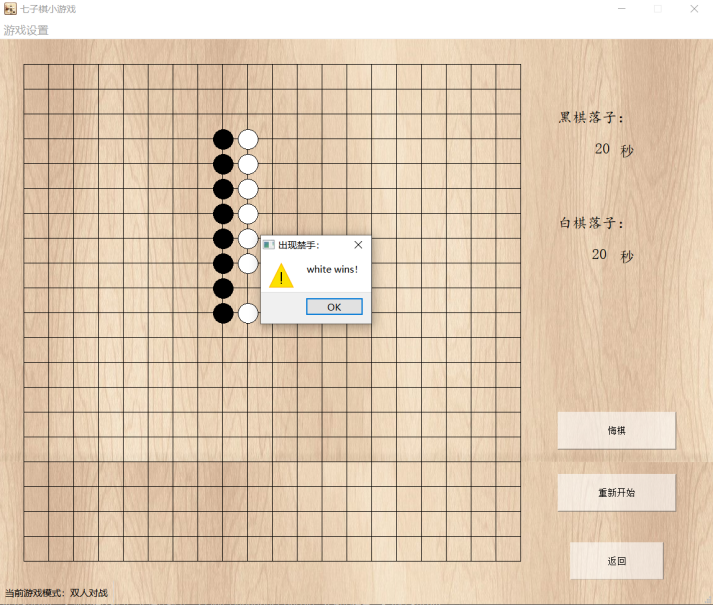
**否**

## 1.5 测试与运行

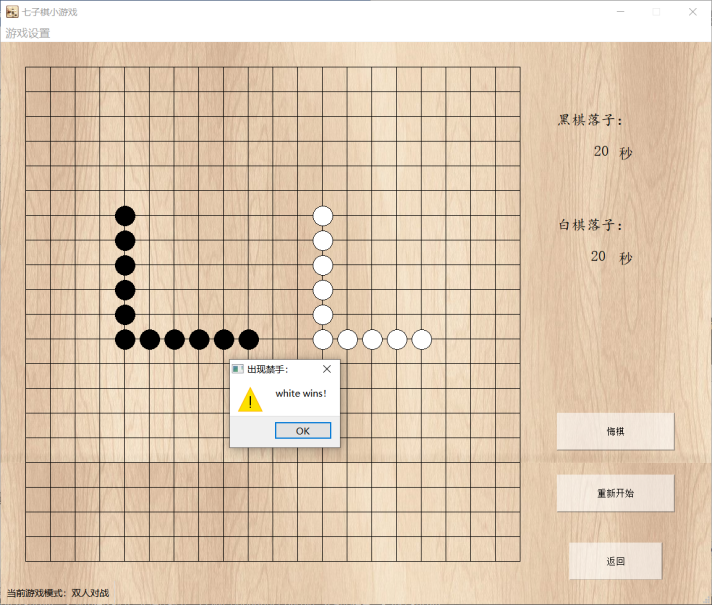
1. 判断输赢

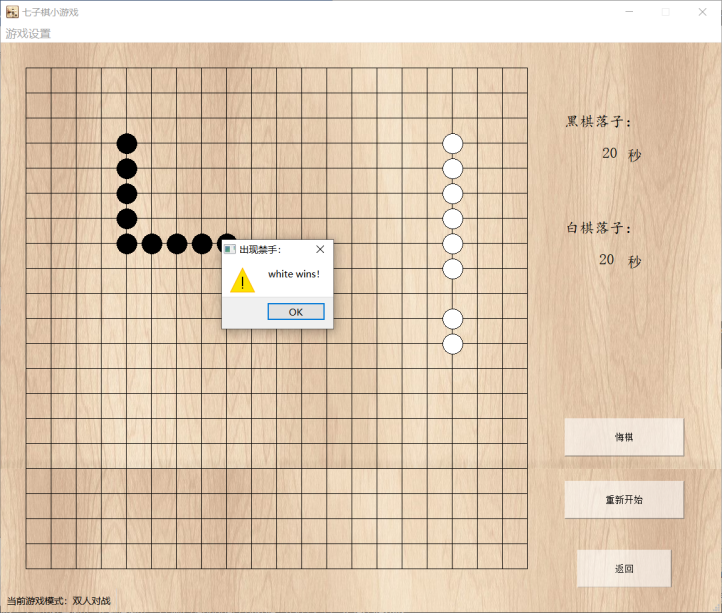


1. 长连禁手

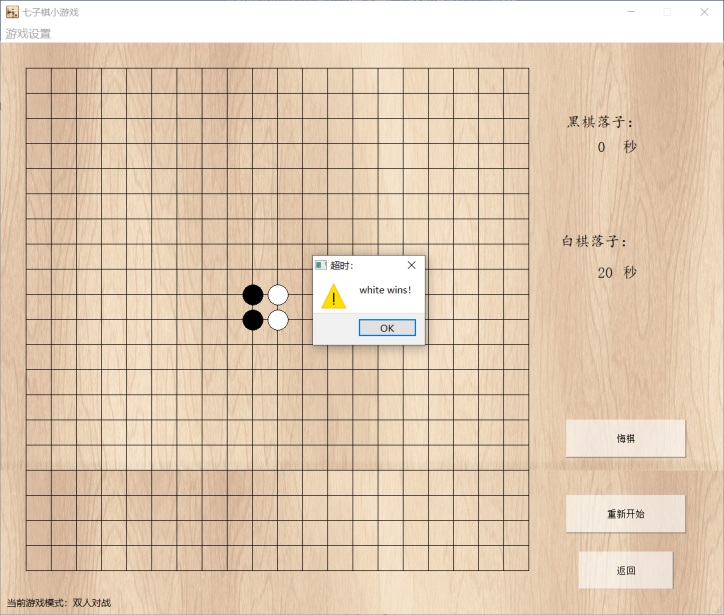


1. 五五禁手与六六禁手

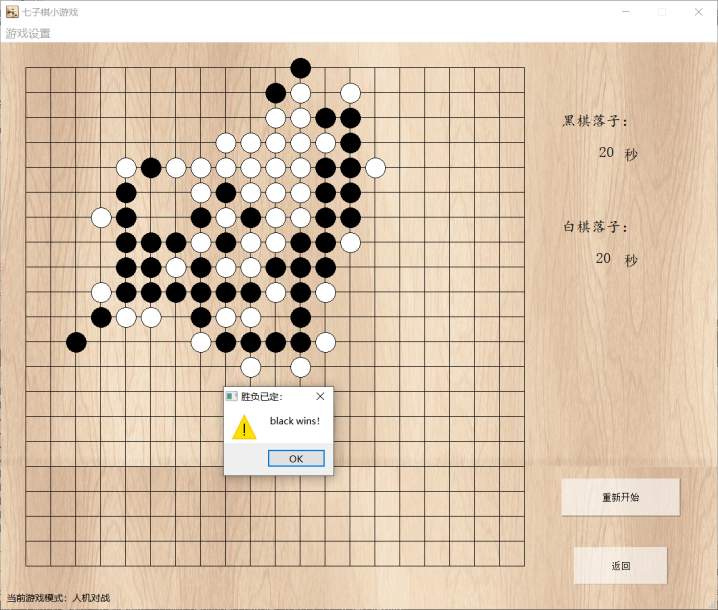




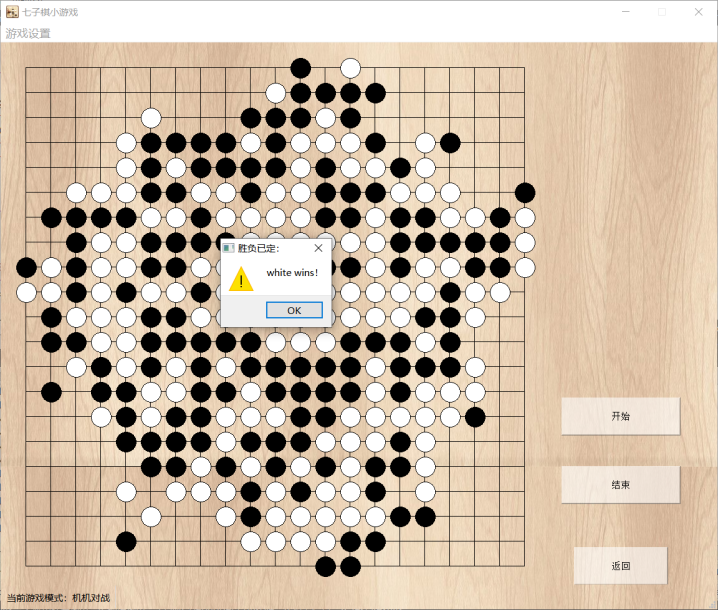
1. 超时



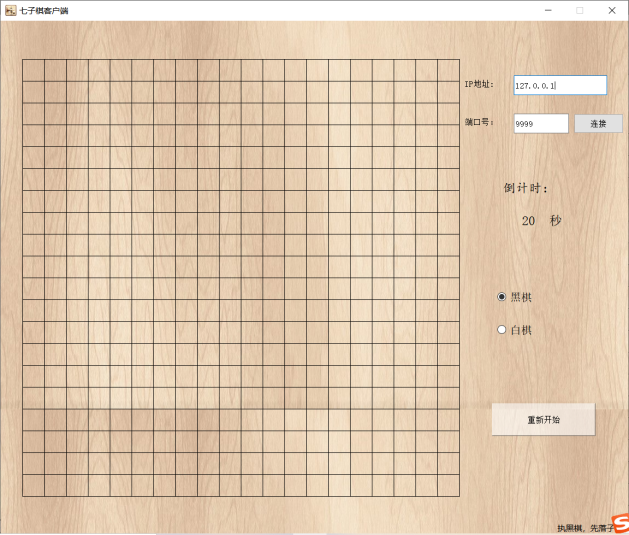
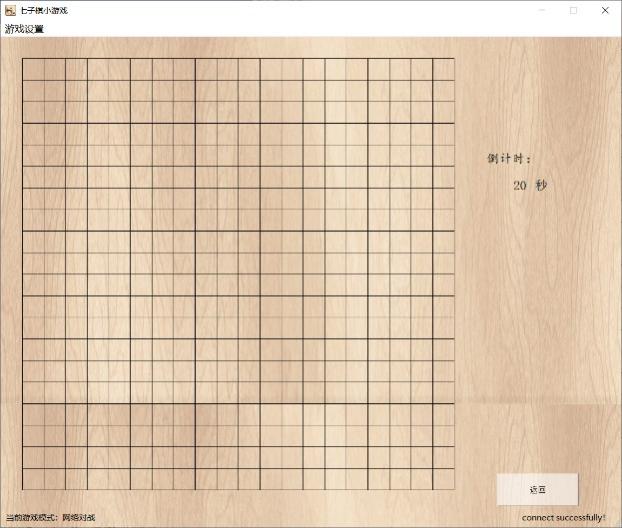
5、人机对战

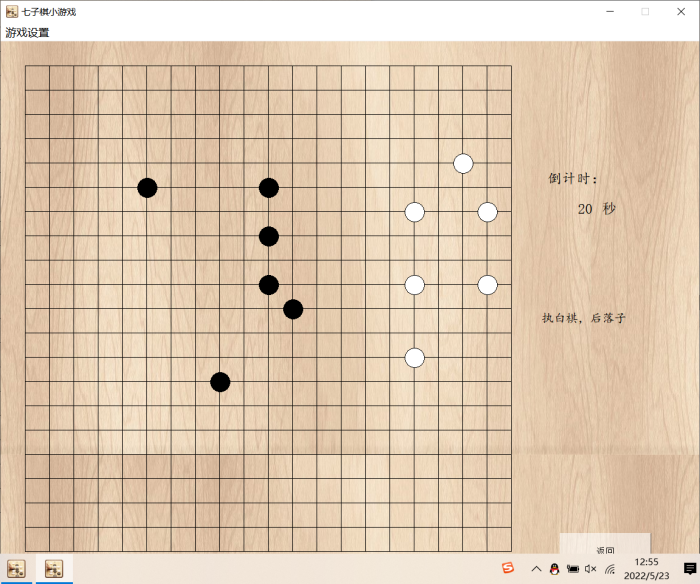
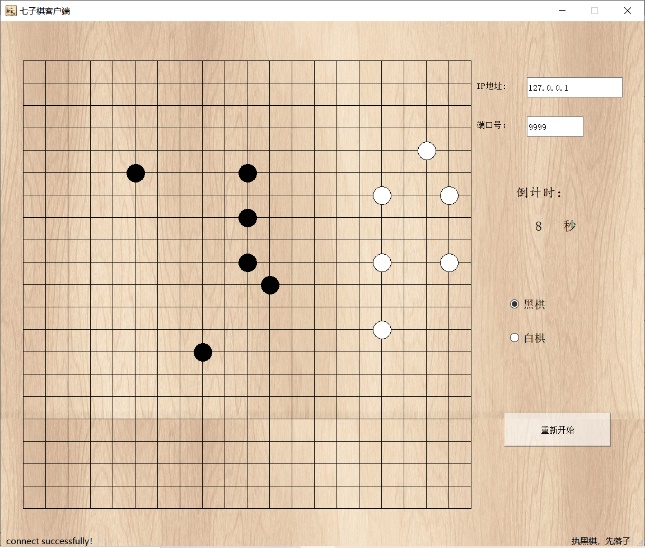


6、机机对战



7、网络对战





## 调试记录

1、问题内容：AI白棋下棋堵黑棋优先于自身连子，赢棋概率低（4.8日）。

解决方案：调整评分分值，适当调高白棋评分，使其比同等级黑棋分略高。

2、问题内容：界面美化过程中，ui界面无法设置背景图，且按钮上图案为背景图 缩小版。（5.3日）

解决方案：重写paintEvent虚函数进行背景图片绘制，并使用png格式文件

3、问题内容：服务端下棋后，客户端更新界面后崩溃（5.21日）。

解决方案：缓存区接收到的数组[441]不存在有效字符，无法进行if判断导致崩溃，在发送信息前先标记数组不为空。

# 2 心得体会

第一次完成了一个c＋＋项目，从零开始到最终的几千行代码基本实现所要求的的功能，共历时两个月，其中感悟可以说是很深，在这过程中遇到不小的艰辛，从刚开始选题目的纠结，到拿到题目后的毫无思路，再到b站上学习qt五子棋程序，做出了七子棋的基本框架，再从GitHub，CSDN上寻找某些特定功能的代码思路，自己进行改写，再到后期的游戏界面美化，一环扣一环，每一个过程都会出现或大或小的bug，需要自己进行代码分析，在网上寻找出错原因，每一个bug的修复都要耗费大量的时间，但同时也一定程度上培养了我的细心与耐心，也很庆幸自己坚持了下来，体会到程序按预期运行成功时的喜悦，我想这也正是编程的乐趣所在，在这过程中也展现出了我对于编程的知识还是了解的很少很少，我未来需要学习的路还很长，希望我在未来的学习过程中也能抱着一个积极乐观的心态面对编程，面对编程过程中奇形异状的bug，成就更好的自己。

# 3毕业要求达成度分析

|  |  |  |  |
| --- | --- | --- | --- |
| **达成指标** | **具体的针对性方案** | **自评等级（1~5）** | **综合评定等级**  **（1~5）** |
| （2-3）能够针对计算机领域中的复杂工程问题进行需求分析和描述。 | 对于七子棋项目，将其拆分为四种游戏模式，分析每一种模式下下棋步骤与每一步可能产生的结果，并对其进行分类判断与处理，增添用户可能需要的功能，提升游戏总体的体验感 | 3 |  |
| （5-1）针对复杂工程问题，了解与计算机专业相关的重要资料与信息的来源及其获取方法，能够通过图书馆、互联网及其他资源或信息检索工具，进行资料查询、文献检索，掌握运用现代信息技术和工具获取相关信息的基本方法。 | 利用《qt 5.9 c++开发指南》学习qt中一些函数的使用，利用bilibili学习qt七子棋的总体思路与框架，利用Stack Overflow询问问题解决代码bug，利用CSDN、GitHub学习类似项目某些特定功能的思路。并对其进行改写。 | 4 |  |
| （8-2）能够在计算机领域工程实践中理解并遵守工程职业道德和规范，履行责任。 | 在写代码的过程中坚持不照搬照抄，只借鉴他人编程思路，并自己进行改写。 | 4 |  |
| （9-1）具有团队意识，能够理解团队中每个角色的含义以及完成自己在团队中的充当角色的工作。 | 无 |  |  |
| （9-2）具备多学科背景知识，能够在多学科背景下的团队中与团队成员沟通，了解团队成员想法，并能够协调和组织。 | 无 |  |  |
| （10-1）能够就计算机领域复杂工程问题与业界同行及社会公众进行有效沟通和交流，包括撰写报告和设计文稿、陈述发言、清晰表达或回应指令。 | 能就项目中出现的问题及时与同学交流讨论，并撰写课程设计报告，进行课程答辩。 | 3 |  |