高级语言程序设计

实验报告

南开大学 网络空间安全学院
姓名：辛浩然
学号：2112514
班级：信息安全、法学双学位班
2022 年 12 月 1 日

目录

高级语言程序设计大作业实验报告

一. 作业题目

基于 Qt 开发的中国象棋

二. 开发软件

Qt 5.15.2

三. 课题要求

1) 面向对象
2) 单元测试
3) 模型部分
4) 验证

四. 主要流程

1．整体流程

实现思路：

i. 搭建开始页面

定义类 Start，派生自 QDialog 类。

通过 Start 类构造函数，设置窗体大小、logo、标题，添加背景图片，添加音效。添加两个按钮，并通过 connect 函数将单击信号连接到按键槽函数 void slotClicked().slotClicked 函数中针对不同的按钮对类成员 chosen 赋不同值。

ii. 设置模式选择

定义类 Switch，派生自 QWidget 类，进行模式的选择。

Switch 类获得 Start 类得到的 chosen 值，通过指针调用不同类，打开不同窗口，实现模式的切换。不同模式添加不同音效。

iii. 棋盘棋子的绘制

a.绘制棋盘：定义类 Chess，在类中通过 paintEvent 函数实现棋盘及细节的绘制。

b.棋子的初始化：

定义类 Stone，类成员包括行坐标、列坐标、存活状态、颜色和棋子类型。定义结构体数组，遍历行列坐标设定不同的棋子类型。通过 Initialize 函数将结构体数组中的对应数据赋值给类成员。通过 stoneName，将棋子的类型返回为 QString.

c.棋子的绘制：

在 Stone 类中，通过 drawStone 函数绘制棋子。具体包括绘制圆形轮廓和文字。细节中考

虑到红黑棋子的不同颜色和 棋子被选中变色问题。

iv. 走棋的实现

通过 mouseReleaseEvent 函数获取鼠标点击位置。获取位置后调用虚函数 click 函数，每次走棋须点击两次。第一次点击确定要走的棋子，本次点击需保证选择的棋子是该走的棋子。第二次点击是确定移动到的位置。需要考虑三种情况：移动到空位置；点击到对方棋子，进行吃棋；点击到己方棋子更换要走的棋子然后再次点击。

v. 棋局规则

1.走棋规则

canMove 函数判断走棋是否合法。选到同色棋子，不可移动，需要移动的棋子变成了后选中的棋子。根据不同棋子类型调用不同的棋子移动判断函数，根据两次点击的位置结合象棋规则判断是否合法。

2.其他规则

红方先行、红黑交替、胜负判断。

vi. 人机对弈模式

定义类 Single，派生自父类 Chess.

1.轮流走棋

重写父类虚函数 click 函数。在人机对弈模式下通过指针调用子类的 click 函数，实现人机对弈模式的设置。click 函数中，红棋由人走棋，调用 Chess 类的 click 函数；红棋走完轮到黑棋，由机器给出走法。

2.获取可走棋方法

通过 getAllarea 函数遍历所有存活棋子，遍历棋盘所有位置，判断是否可以移动，如果可以移动，将走法保存。

3.保存可移动走法

定义类 Step，类成员包括两次点击的棋子及行列位置。

定义 QVector<Step *>容器，在 getAllarea 函数中，如果可走棋，就通过 saveSteps 函数保存至容器。

4.获取最佳走法

为不同的棋子设定不同的子力值，计算己方所有存活棋子分数，减去对方分数，得到局面分。

遍历所有保存的走法，比较局面分获取最佳走法。通过函数 Step *getBestScore();int getMinScore(int, int);int getMaxScore(int, int)实现。具体算法为 MinMax 算法和 Alpha-Beta 剪枝。

定义 aheadStep，设定机器考虑的步数。程序中将其设定为 3。机器需要考虑的包括向后第三步己方走棋使自己得分最高的局面，向后第二步对方走棋使自己得分最低的局面，向后第一步己方走棋使自己得分最高的局面。通过 Step *getBestScore();int getMinScore(int, int);int getMaxScore(int, int).

2．算法

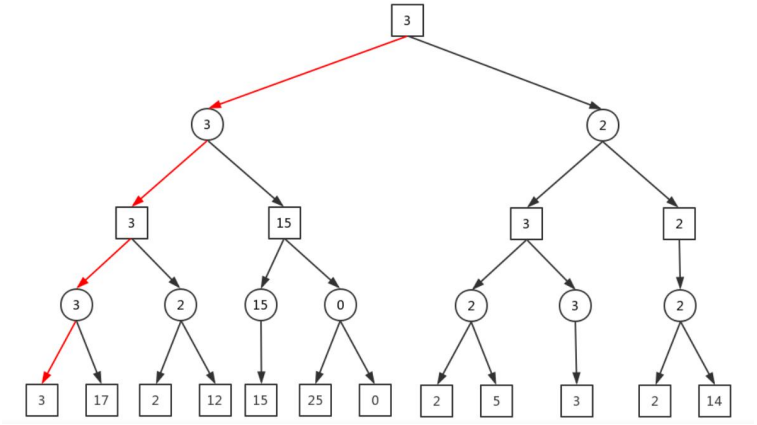
MinMax 算法和 Alpha-Beta 剪枝

（1）MinMax 算法

假设对手具有完美决策能力，我方的策略是选择对方在完美决策下对我造成的损失最小的局面。

举例说明，假设向后看四步。首先，先手计算后手在第四步的时候最佳走棋使自己胜算最小的局面；然后先手计算自己在第三步时胜算最大的局面；然后先手计算后手在第三步时最佳

走棋使胜算最小的局面。最后先手综合选择胜算最大的局面，决定下一步的走棋。



图源：https://blog.csdn.net/weixin_42165981/article/details/103263211

（2）Alpha-Beta 剪枝

α：到目前为止路径上发现的 MAX 的最佳选择（即极大值）；β：到目前为止路径上发现的 MIN 的最佳选择（即极小值）；αβ剪枝策略在搜索中不断更新α和β的值，并且当某个结点的值分别比目前的 MAX 的α或者 MIN 的β值更差的时候，终止递归搜索。

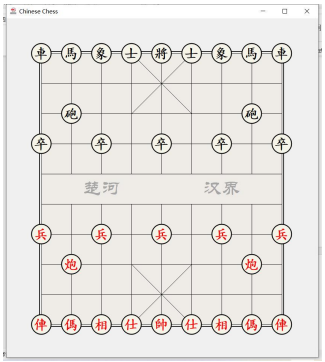
3．单元测试

进行走棋测试，判断是否符合规则。

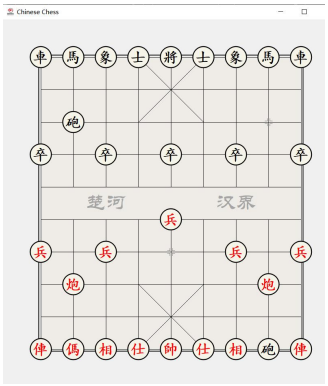
五.　单元测试

开始页面测试：

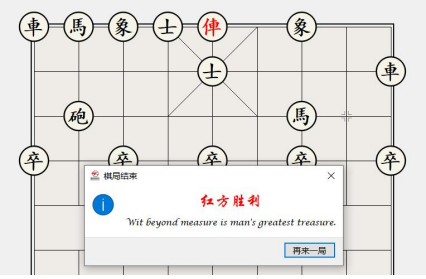
棋盘棋子绘制测试：



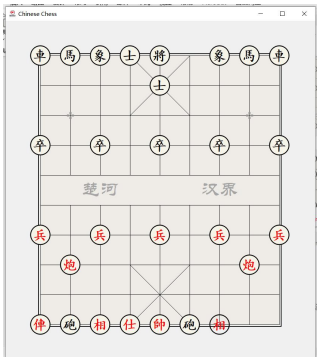
双人对弈走棋规则测试：



胜利判断测试：



人机对弈测试：电脑开局炮进行吃棋。当红方要杀黑将时，黑方进士应将。



测试结果

中间测试过程中出现一些问题。某次人机对弈测试时，电脑走棋后红方棋子全部消失，通过断点调试发现在遍历时，fakemove 中对棋子的状态修改有误。

在测试中还出现了向后看多步时电脑反应速度慢。由于黑棋紧跟红棋走棋，导致黑棋计算完成后红棋的移动才显示出来。解决方案：添加计时器，在红棋走后 0.1s 黑棋开始计算走法并给出走棋。修改算法，进行剪枝，减少搜索量。

解决了一些问题后，最终测试结果正常。

六.　收获

1．Qt 的基本操作、QT 常用库、类、函数等

2．算法：MinMax 算法与 Alpha-Beta 剪枝

3．类、类的继承与派生、虚函数