同化棋大作业报告

1. 图形界面的功能部分

首先是一个简洁的图形界面部分：



对于不熟悉规则或想要了解更多有关此程序信息的用户，在点击红色部分后：

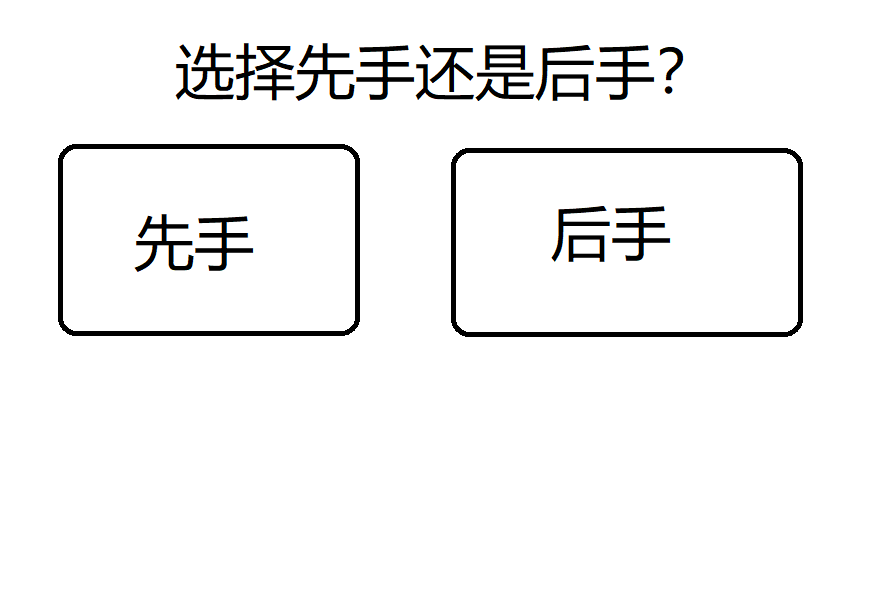
在此界面可以简单地了解规则、在线对战地址以及源代码网址。

之后便可以进入棋盘的开始界面，在这里可以选择开始新游戏（Start）加载旧存档（Load）抑或者直接退出游戏（Exit）。

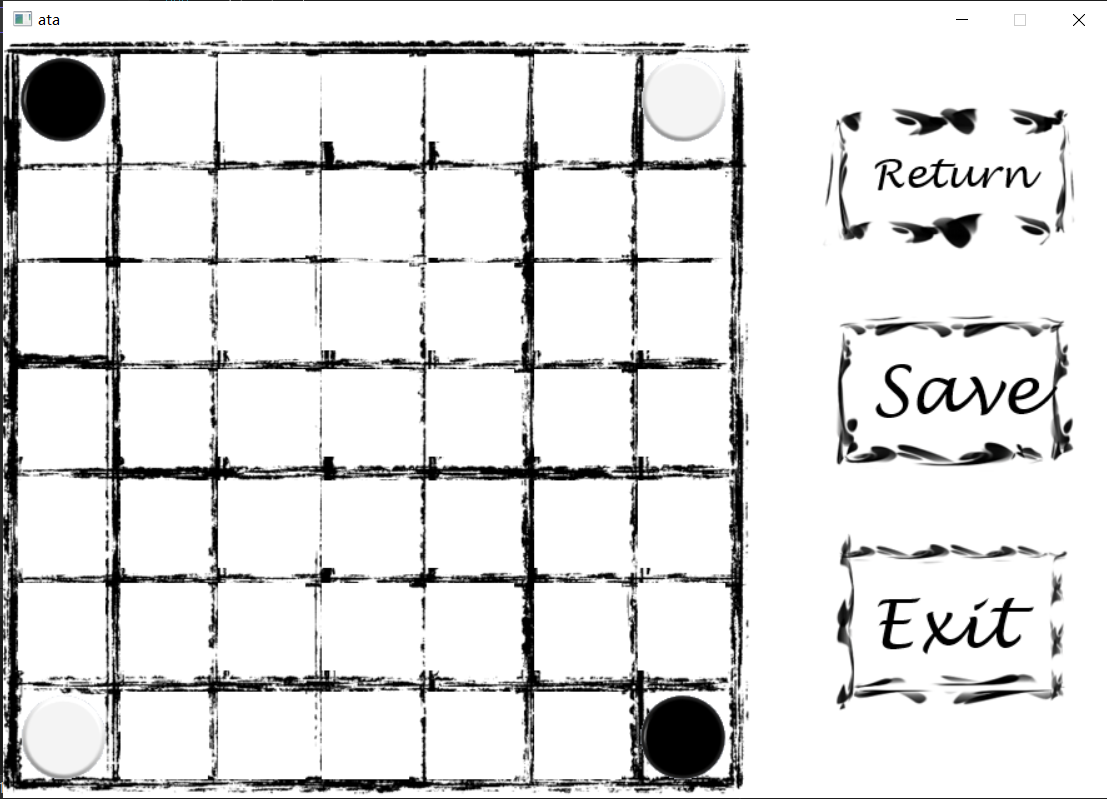
点击开始便可以进入一场新的游戏，在正式进行游戏之前需要选择单人或者双人游戏。



如果选择单人游戏，则用户需要继续选择执黑棋还是白棋。



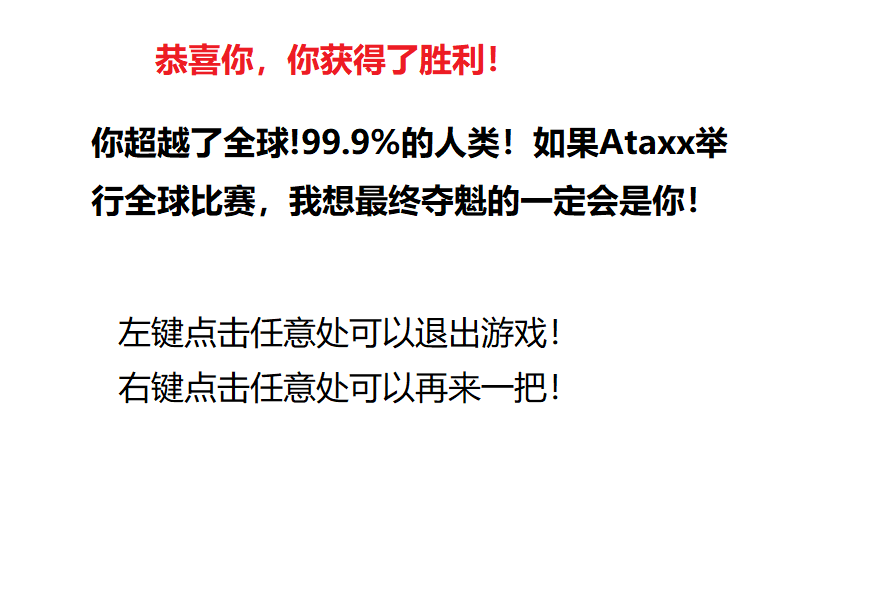
在确定双人游戏或者确定了单人游戏及执先后手之后，我们就会进入游戏界面了：



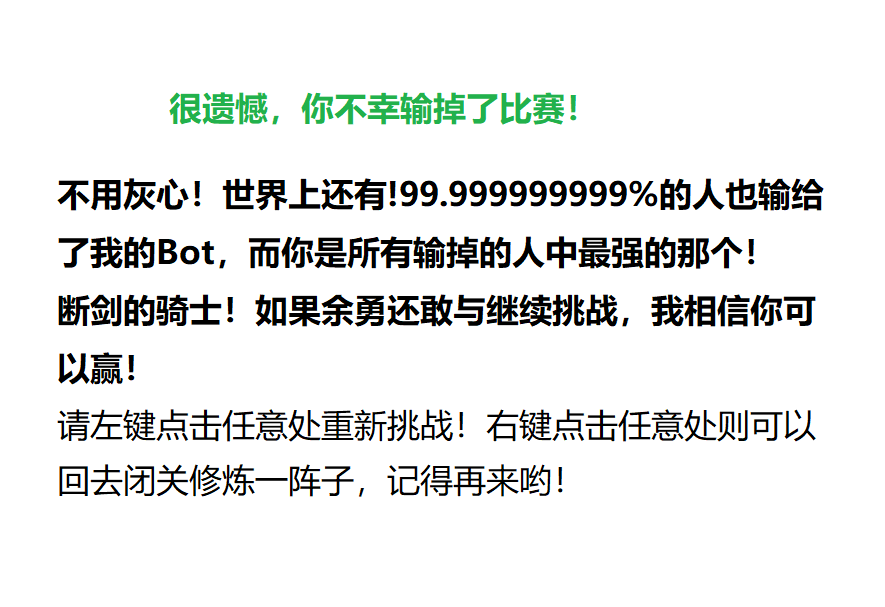
之后，如果是双人游戏，需要轮流进行鼠标移动黑白棋的操作，如果是单人游戏，程序预留了1s的时间以确保移动的路径可以被观测到。

在棋局的任何阶段用户都可以选择返回（Return），手动存档（Save），以及退出游戏（Exit）的按钮，返回会直接返回上一个界面，开启新游戏；而存档在保留当前棋局的同时也会将用户对本局游戏的设置一并保存，以便下次加载存档时直接地进入游戏环节。

在单人游戏中，如果获得胜利，则会显示以下画面：

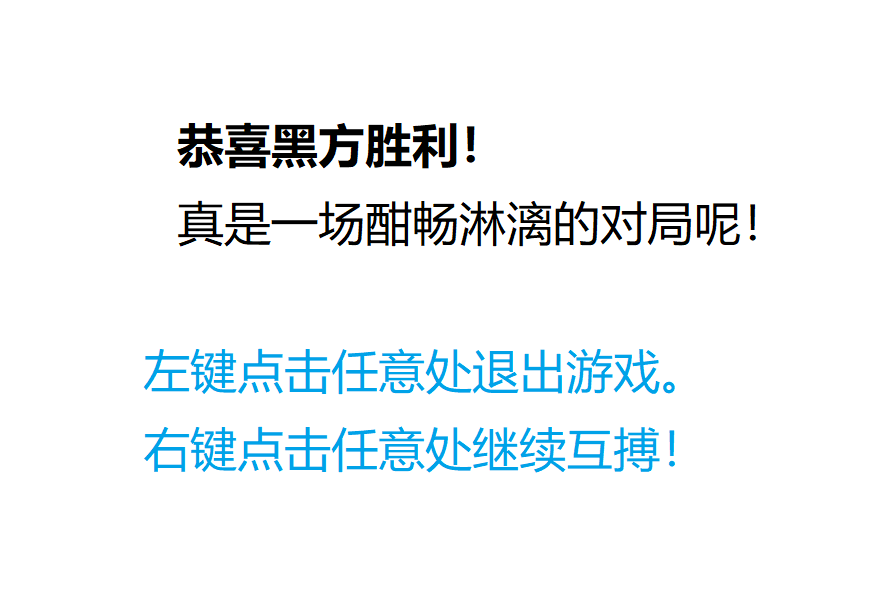


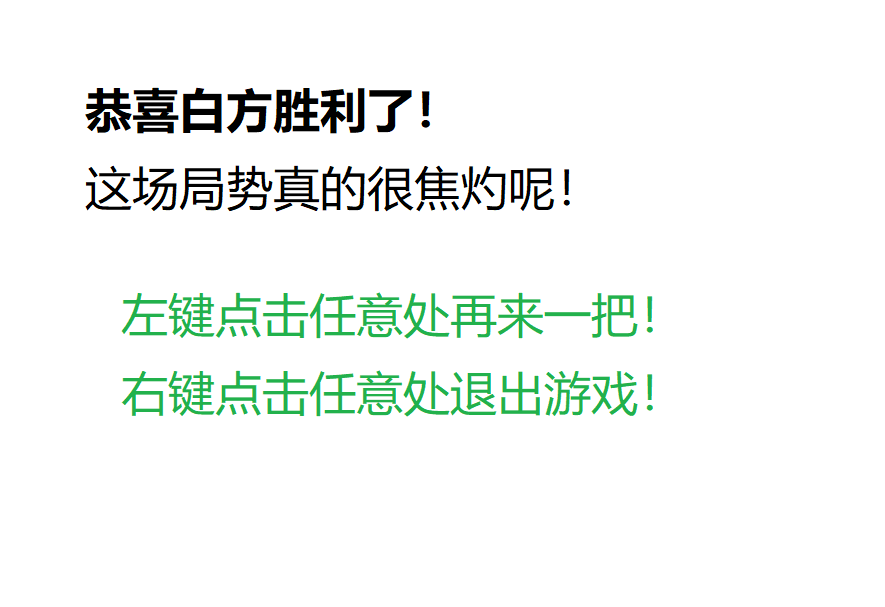
右键点击任意处后可以继续进行新游戏或读取上次手动存档时的内容。输掉比赛之后，会显示：



此时左键点击可以进行新游戏、读取存档。

在双人游戏中，当比赛决出胜负时，会根据可能的结果展现如下两种界面：



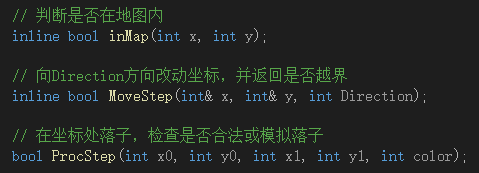


同样的，用户可以在不退出程序的情况下继续游玩，实现了长时间游戏的功能。

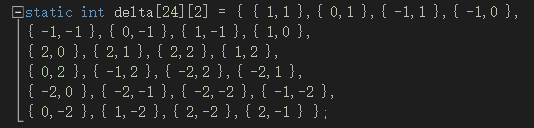
1. 设计思路部分

后端部分：

主要包括走棋函数以及决策函数，棋盘的状态使用7\*7的二维数组进行存储，走棋函数主要包括这几个函数：

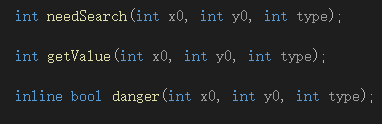


在ProcStep函数中需要调用前两个函数以进行同化棋规则的实现，需要补充说明，我使用int型变量color表示棋盘状态，1表示有黑色棋子，0表示无棋子，-1表示有白色棋子。而int型变量Direction则代表delta1、delta2两数组的下标。（delta1与delta2下标一致时才有意义），具体如下：

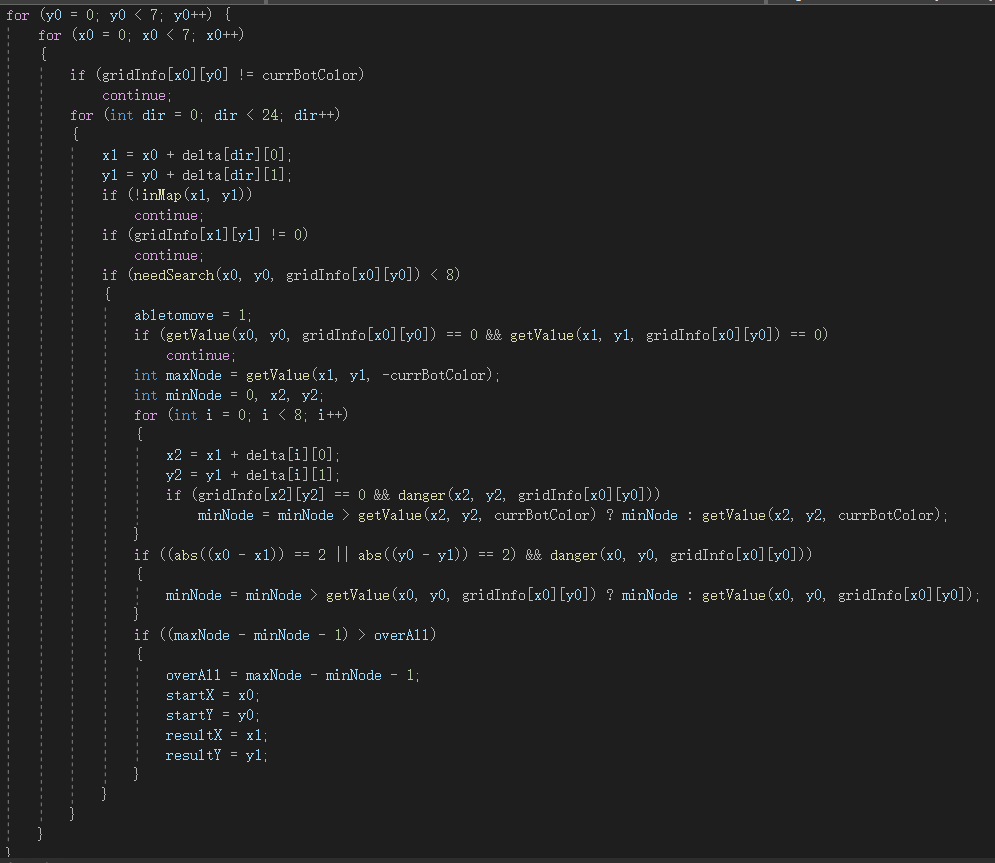


表示24种可能移动棋子的相对位置。

决策函数部分主要如下：

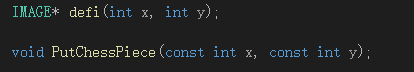


采用了minmax以及Alpha-beta剪枝，needSearch剪去了max节点中收益较小的部分，getValue则是在第一种走法（指原位置棋子不会消失的走法）的估值函数，danger则是用于计算目标移动位置可改变对方棋子的个数，主要的估值进行于主函数中：



通过对现有可能走法收益由两步之内对方min走法及自身max走法决定，较差的情况会被剪枝以节省时间。最终找出估值最高的走法。

前端部分：

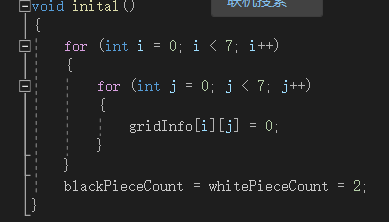


Defi函数用来确定在此棋盘格由二维数组改变之后现有情况对每个格子进行下黑棋、下白棋或者是擦去棋的操作种类。PutChessPiece则是具体地对每个像素点操作使背景颜色不会覆盖到棋盘上。



Click函数用来接收在棋盘格上对棋子进行移动的鼠标信息，其中错误的点击不会被接受，仍然可以在程序持续运行的情况下直至点击可被接受的类型。

正如游戏界面显示的那样，同样的click也能进行返回、手动存档、退出的操作。

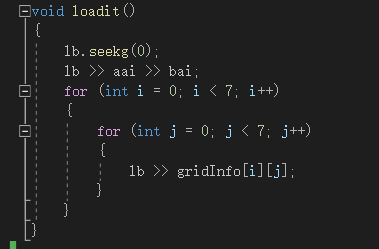


Initial是在返回进行新游戏时初始化棋盘的操作。

文件存储部分：

使用文本输入及输出。

加载存档：



手动存档：

