DDL：2017-10-27

机器学习：五子棋对弈训练

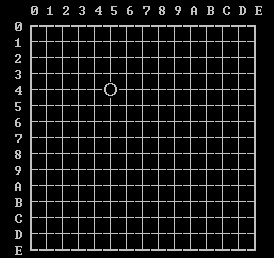
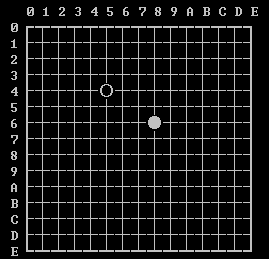
**五子棋对战平台：**

棋盘大小 15x15

玩家1（黑子）先手：●

玩家2（白子）后手：○

下图中：（4, 5）为黑子，（6, 8）为白子 【坐标格式（x, y）】

棋盘实例1 棋盘实例2

**五子棋规则：**（以下**先手**均用**黑子**代替、**后手**均用**白子**代替）

1. 决策输入不能超过棋盘大小 0 – 14，上图中A – E 对应 10 – 14，请勿输入字母。

决策输入不能覆盖原棋盘上已有棋子。

出现上述情况，虽在测试版时提示警告，但会在最终评比时直接判输。

1. 黑子胜利条件：黑子在未违规（三三禁手、四四禁手、长连禁手等）下达成五子连珠。

（系统自带判定黑子决策是否为禁手，若为禁手则判对方胜）

1. 白子胜利条件：白子没有禁手限制，任意状态达成五子连珠即可。

**工程文件说明：**

1. AI.h、AI.cpp： 自带的五子棋ai（可与之一战）
2. TABLE.h、TABLE.cpp： 棋盘画图文件
3. JUDGE.h、JUDGE.cpp：判定禁手或胜负文件
4. main.cpp： 顶层模式控制

（以上文件不建议进行大量修改，但可以适当读取所需数据）

（也不允许进行规则上的修改，最后判别规则使用原版规则）

1. decision.h、decision.cpp：决策输入。请将最终训练好的决策程序封装好，在里面引用之，获得新的x（行号）, y（列号）坐标，放入pos数组即可。

**工程输入、输出说明：**

输入：即下子操作，格式为x y（x空格y），x代表行号，y代表列号，范围0 – 14。

输出：每一次下子，Debug目录下会产生ChessLog.txt，以15x15矩阵形式表示当前棋盘状态，0代表无子，1代表白子，2代表黑子。

**期末提交内容：**

1. 五子棋决策程序接口（附上使用说明）
2. 设计报告

（报告中请简明清楚地阐述决策程序的原理、设计过程、实验结果等）

（如果能有一些关键的设计细节、思路、创新点描述，会获得更多分数）

**接口规范说明：**

请各位同学将决策程序封装好，方便以后测试、对弈调用。

程序文件命名：

**名字拼音.h 名字拼音.cpp**

接口名称命名：

**char\* 名字拼音\_Decision();**

例如调用此函数，读取目录下的ChessLog.txt，放入决策程序中运算，得到数组XX[2]，然后返回XX，再在decision.cpp中将XX[0]和XX[1]赋值给x，y作为新的棋子位置。

**void 名字拼音\_Decision(char pos[], int WorB);**

又如在decision.cpp的GetPos()函数中，把你的函数加进去，引入GetPos函数入口已给的pos数组和WorB变量，通过你定义的一系列运算修改pos的值，作为新的棋子位置。

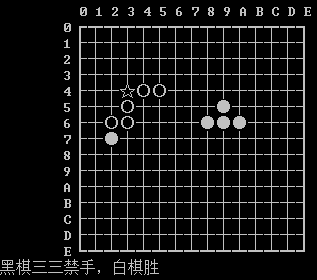
**训练目标：**

1. 最好能打败自带AI，胜率越高分数越高。
2. 最终各位同学的决策程序互相对弈，排名越高分数越高。

**参考思路：**

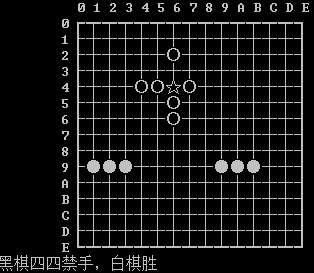
1. 对每一步的下子策略，或者棋子的连子情况，给定一个评分，根据评分优化决策。
2. 随缘乱下，根据最后的胜负，设置奖励与惩罚，优化决策。
3. 博弈树，递归搜索，强化学习

下页有附录（禁手相关说明）

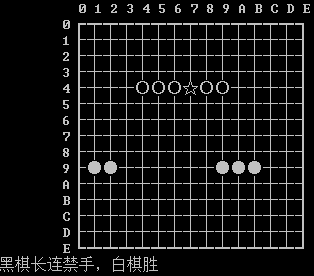
**附录：**

三三禁手：

黑方一子落下同时形成两个或两个以上的活三（或嵌四），此步为三三禁手。

四四禁手：

黑方一子落下同时形成两个或两个以上的四，活四、冲四、嵌五之四，包括在此四之内。

长连禁手：

黑方一子落下形成连续六子或六子以上相连。