约定：

变量一律小写，board，两个单词之间用下划线隔开

常量一律大写，PLAYER

类一律大写开头，Board

类方法，骆驼写法，第一个单词小写，后面单词第一个字母大写，updateBoard

围棋记录

深度学习的reward获取在时效性上分2种，一种是立即获取，比如扫雷游戏，另外一种是将来获取。将来获取又分两种，当前步在未来获取reward，其后的操作步对reward没有影响，比如弹珠游戏，挡板接住珠子后弹出去后，一直到小珠碰到得分砖块，挡板之间的运动对得分是没有影响的。还有一种是当前步在未来的reward和其后的操作步相关，比如围棋，未来的每一步都关系到棋局最后的输赢。不同的情况就要对应不同的学习策略。

类的抽象：

一局棋由四部分组成：

1. 棋盘
2. 棋子
3. 棋的规则
4. 下棋的人

因此，需要有四个类来对应这四部分的抽象概念。当一局棋开始时，把这四个类实例化，程序就可以开始跑了。程序中对应的四个类是：

1. 棋盘-goBoard
2. 棋子-goString
3. 棋的规则-goJudge
4. 下棋的人-goAgent

在这其中，最重要的是goAgent，他也是本书的主角。为了遵循由简入繁的原则，先从棋盘的类开始讲起

stone表示棋盘上的坐标

goString才是表示棋子

行棋结束的评价:

1：双方都是pass，才结束

2：有人投降

pass的条件：

1：没有一个落子位可以既行棋合法又策略合法

策略违法：

1：要注意，人为定义的策略应该越少越好，这样才能学到更多的形式。在强化学习中，人为定义的策略其实可以是0，全靠机器自己去学，但是这样会消耗更多的时间去学习，考虑到go的学习时间相当长，我们可以人为定义一些简单且明确的策略行为来减少学习时间。就增加一条关于眼位的策略约定。眼位的定义是（包含假眼）：落子位上下左右的位置均已经被己方占领。策略原则是，当且仅当落子位的上下左右的气有至少一个等于0，才可以占用

实现的蒙特卡洛是比较简单的，加入了启发式的方法。以获取己方最大影响力为目标，这个方法的缺点是不考虑减弱对方的影响力，思考方法是在对方获取影响力的情况下，最大化自己利益。这个算法有个缺点，就是没有对抗，各顾各下棋，就像对方不存在一样。一个可以考虑的优化方法是，落子在对方最大影响力的位子。