**程序报告**

学号：2013750 姓名：管昀玫

1. **问题重述**

（简单描述对问题的理解，从问题中抓住主干，必填）

====================================================================

如何能在 8\*8 的国际象棋棋盘上放置八个皇后，使得任何一个皇后都无法直接吃掉其他的皇后？为了到达此目的，任两个皇后都不能处于同一条横行、纵行或斜线上。

1. **设计思想**

（所采用的方法，有无对方法加以改进，该方法有哪些优化方向（参数调整，框架调整，或者指出方法的局限性和常见问题），伪代码，理论结果验证等… **思考题，非必填**）

====================================================================

**使用递归算法，穷举所有的可能（不在同一行、列、斜线上）；棋盘只有0-7行/列，如果来到了第八行，则说明可行，那么添加到解中。**

**使用copy函数，进行深拷贝。**

**一些优化想法：如果不调用board函数，可以如下快速判断是否在对角线上：i+j-1和i+j-N。**

1. **代码内容**

（能体现解题思路的主要代码，有多个文件或模块可用多个"===="隔开，必填）

====================================================================

def search(self, step, n):

#递归搜索

#终止条件

if self.chessBoard.isWin():

self.solves.append(step.copy())

return

for j in [0,1,2,3,4,5,6,7]:

#悔棋

self.chessBoard.boardInit(False)

for i,item in enumerate(step[:n]):

self.chessBoard.setQueen(i,item,False)

#尝试下棋，符合条件继续，否则停止

if self.chessBoard.setQueen(n,j,False):

step[n]=j

self.search(step,n+1)

1. **实验结果**

（实验结果，必填）

====================================================================

[0, 4, 7, 5, 2, 6, 1, 3]

[0, 5, 7, 2, 6, 3, 1, 4]

[0, 6, 3, 5, 7, 1, 4, 2]

[0, 6, 4, 7, 1, 3, 5, 2]

[1, 3, 5, 7, 2, 0, 6, 4]

[1, 4, 6, 0, 2, 7, 5, 3]

[1, 4, 6, 3, 0, 7, 5, 2]

[1, 5, 0, 6, 3, 7, 2, 4]

[1, 5, 7, 2, 0, 3, 6, 4]

[1, 6, 2, 5, 7, 4, 0, 3]

[1, 6, 4, 7, 0, 3, 5, 2]

[1, 7, 5, 0, 2, 4, 6, 3]

[2, 0, 6, 4, 7, 1, 3, 5]

[2, 4, 1, 7, 0, 6, 3, 5]

[2, 4, 1, 7, 5, 3, 6, 0]

[2, 4, 6, 0, 3, 1, 7, 5]

[2, 4, 7, 3, 0, 6, 1, 5]

[2, 5, 1, 4, 7, 0, 6, 3]

[2, 5, 1, 6, 0, 3, 7, 4]

[2, 5, 1, 6, 4, 0, 7, 3]

[2, 5, 3, 0, 7, 4, 6, 1]

[2, 5, 3, 1, 7, 4, 6, 0]

[2, 5, 7, 0, 3, 6, 4, 1]

[2, 5, 7, 0, 4, 6, 1, 3]

[2, 5, 7, 1, 3, 0, 6, 4]

[2, 6, 1, 7, 4, 0, 3, 5]

[2, 6, 1, 7, 5, 3, 0, 4]

[2, 7, 3, 6, 0, 5, 1, 4]

[3, 0, 4, 7, 1, 6, 2, 5]

[3, 0, 4, 7, 5, 2, 6, 1]

[3, 1, 4, 7, 5, 0, 2, 6]

[3, 1, 6, 2, 5, 7, 0, 4]

[3, 1, 6, 2, 5, 7, 4, 0]

[3, 1, 6, 4, 0, 7, 5, 2]

[3, 1, 7, 4, 6, 0, 2, 5]

[3, 1, 7, 5, 0, 2, 4, 6]

[3, 5, 0, 4, 1, 7, 2, 6]

[3, 5, 7, 1, 6, 0, 2, 4]

[3, 5, 7, 2, 0, 6, 4, 1]

[3, 6, 0, 7, 4, 1, 5, 2]

[3, 6, 2, 7, 1, 4, 0, 5]

[3, 6, 4, 1, 5, 0, 2, 7]

[3, 6, 4, 2, 0, 5, 7, 1]

[3, 7, 0, 2, 5, 1, 6, 4]

[3, 7, 0, 4, 6, 1, 5, 2]

[3, 7, 4, 2, 0, 6, 1, 5]

[4, 0, 3, 5, 7, 1, 6, 2]

[4, 0, 7, 3, 1, 6, 2, 5]

[4, 0, 7, 5, 2, 6, 1, 3]

[4, 1, 3, 5, 7, 2, 0, 6]

[4, 1, 3, 6, 2, 7, 5, 0]

[4, 1, 5, 0, 6, 3, 7, 2]

[4, 1, 7, 0, 3, 6, 2, 5]

[4, 2, 0, 5, 7, 1, 3, 6]

[4, 2, 0, 6, 1, 7, 5, 3]

[4, 2, 7, 3, 6, 0, 5, 1]

[4, 6, 0, 2, 7, 5, 3, 1]

[4, 6, 0, 3, 1, 7, 5, 2]

[4, 6, 1, 3, 7, 0, 2, 5]

[4, 6, 1, 5, 2, 0, 3, 7]

[4, 6, 1, 5, 2, 0, 7, 3]

[4, 6, 3, 0, 2, 7, 5, 1]

[4, 7, 3, 0, 2, 5, 1, 6]

[4, 7, 3, 0, 6, 1, 5, 2]

[5, 0, 4, 1, 7, 2, 6, 3]

[5, 1, 6, 0, 2, 4, 7, 3]

[5, 1, 6, 0, 3, 7, 4, 2]

[5, 2, 0, 6, 4, 7, 1, 3]

[5, 2, 0, 7, 3, 1, 6, 4]

[5, 2, 0, 7, 4, 1, 3, 6]

[5, 2, 4, 6, 0, 3, 1, 7]

[5, 2, 4, 7, 0, 3, 1, 6]

[5, 2, 6, 1, 3, 7, 0, 4]

[5, 2, 6, 1, 7, 4, 0, 3]

[5, 2, 6, 3, 0, 7, 1, 4]

[5, 3, 0, 4, 7, 1, 6, 2]

[5, 3, 1, 7, 4, 6, 0, 2]

[5, 3, 6, 0, 2, 4, 1, 7]

[5, 3, 6, 0, 7, 1, 4, 2]

[5, 7, 1, 3, 0, 6, 4, 2]

[6, 0, 2, 7, 5, 3, 1, 4]

[6, 1, 3, 0, 7, 4, 2, 5]

[6, 1, 5, 2, 0, 3, 7, 4]

[6, 2, 0, 5, 7, 4, 1, 3]

[6, 2, 7, 1, 4, 0, 5, 3]

[6, 3, 1, 4, 7, 0, 2, 5]

[6, 3, 1, 7, 5, 0, 2, 4]

[6, 4, 2, 0, 5, 7, 1, 3]

[7, 1, 3, 0, 6, 4, 2, 5]

[7, 1, 4, 2, 0, 6, 3, 5]

[7, 2, 0, 5, 1, 4, 6, 3]

[7, 3, 0, 2, 5, 1, 6, 4]

There are 92 results.

0 1 2 3 4 5 6 7

0 o - - - - - - -

1 - - - - o - - -

2 - - - - - - - o

3 - - - - - o - -

4 - - o - - - - -

5 - - - - - - o -

6 - o - - - - - -

7 - - - o - - - -

1. **总结**

（自评分析（是否达到目标预期，可能改进的方向，实现过程中遇到的困难，从哪些方面可以提升性能，模型的超参数和框架搜索是否合理等），**思考题，非必填**）

====================================================================

一定要使用copy函数，否则是浅拷贝。

我在实验中一直遇到输出“落子失败”，但是没有输出正确的list，检查了三天才发现board.py内置了输出“落子失败”，这只是递归算法中是尝试而已，实际上解中已经保存了正确答案，只是我没有写输出函数。

Python的list写法还是不太熟悉，要多练习。