项目说明文档

数据结构课程设计

**——N**皇后问题

作 者 姓 名： 罗吉皓

学 号： 1652792

指 导 教 师： 张颖

学院、 专业： 软件学院 软件工程

同济大学

**Tongji University**

目录

1. 分析 3

1.1 项目名称 ： N皇后问题 3

1.2 项目背景 3

1.3 项目功能分析 3

II. 设计 4

2.1 数据结构设计 4

2.2 数据结构类的设计 4

2.3 系统设计 4

输入功能截屏示例： 4

III 实现 5

3.1 深度优先搜索算法的实现 5

3.1.1 深度优先搜索算法流程图 5

3.1.2 思路分析 6

3.1.3 具体实现如下： 6

3.2 判断函数的实现 7

3.2.1 判断函数流程图 7

3.2.2 思路分析 7

3.2.3 具体实现如下： 8

3.3 总体系统截屏示例 8

IV 测试 9

4.1搜索功能测试 9

V 总结 10

VI 参考文献 10

1. 分析

*1.1* 项目名称 ： N皇后问题

1.2 项目背景

​ 八皇后问题是一个古老而著名的问题，是回溯算法的经典问题。该问题是十九世纪著名的数学家高斯在1850年提出的：在8\*8的国际象棋棋盘上，安放8个皇后，要求没有一个皇后能够“吃掉”任何其它一个皇后，即任意两个皇后不能处于同一行，同一列或者同一条对角线上，求解有多少种摆法。

高斯认为有76种方案。1854年在柏林的象棋杂志上不同的作者发表了40种不同的解，后来有人用图论的方法得到结论，有92种摆法。

本实验拓展了N皇后问题，即皇后个数由用户输入。

1.3 项目功能分析

​ 作为一个皇后放置问题，基本的功能实现是

​ 1.递归所得所有的可能的皇后放置情况并且可以予以显示

​ 2.统计所有的皇后放置个数

综上所述，一个N皇后项目至少应该具有输入、输出、搜索的功能。

II. 设计

2.1 数据结构设计

​ 如上功能分析所述，该系统要求大量的搜索操作，因此考虑使用递归这种数据结构。本项目使用的算法为深度优先搜索，其具体实现将在后文中有所提及。

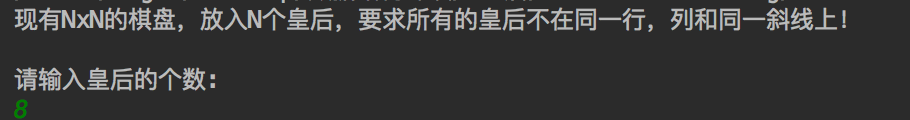
2.2 数据结构类的设计

​ 解决这个问题采用回溯法，首先将第一个皇后放置在第一行第一列，然后，依次在下一行上放置一个皇后，直到八个皇后全部放置安全。在放置每个皇后时，都依次对每一列进行检测，首先检测放在第一列是否与已放置的皇后冲突，如不冲突，则将皇后放置在该列，否则，选择改行的下一列进行检测。如整行的八列都冲突，则回到上一行，重新选择位置，依次类推。

2.3 系统设计

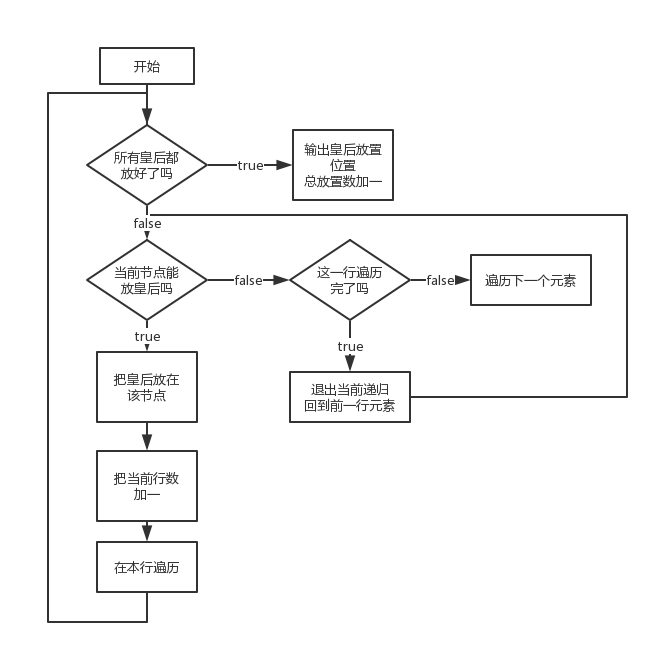
系统提示用户输入皇后的数量，皇后的数量即为棋盘的大小。

输入功能截屏示例：



III 实现

3.1 深度优先搜索算法的实现

3.1.1 深度优先搜索算法流程图

3.1.2 思路分析

深度优先搜索（缩写DFS）有点类似广度优先搜索，也是对一个连通图进行遍历的算法。它的思想是从一个顶点V0开始，沿着一条路一直走到底，如果发现不能到达目标解，那就返回到上一个节点，然后从另一条路开始走到底，这种尽量往深处走的概念即是深度优先的概念。

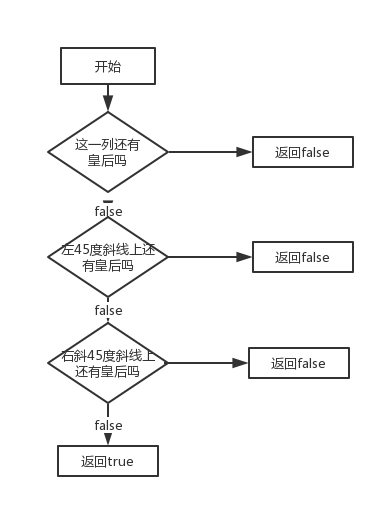
深度优先搜索采用的是模拟放子的思想，但要注意的是，在递归返回时，一定要记得把原先模拟的点复原回初始状态。

3.1.3 具体实现如下：



3.2 判断函数的实现

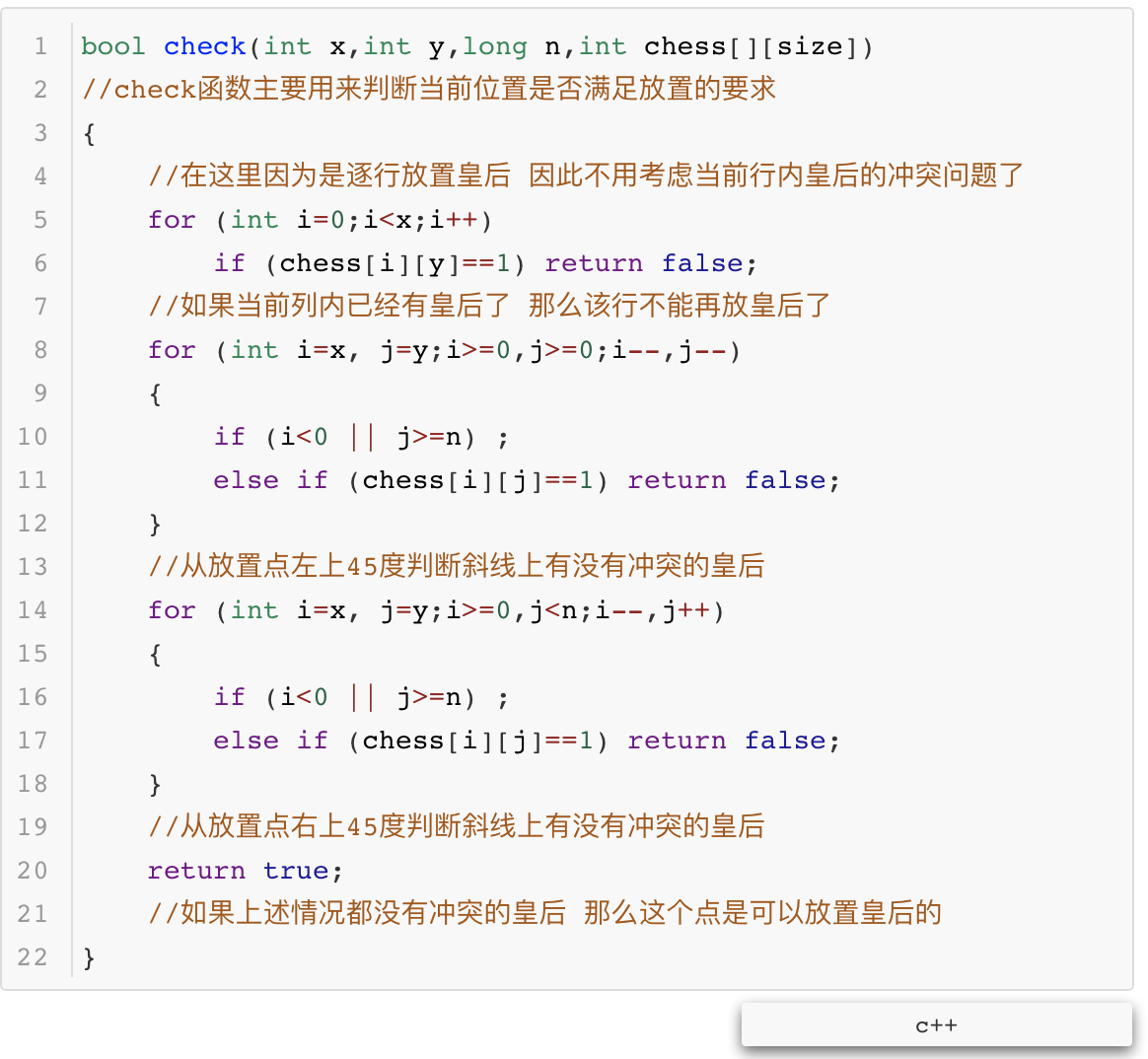
3.2.1 判断函数流程图



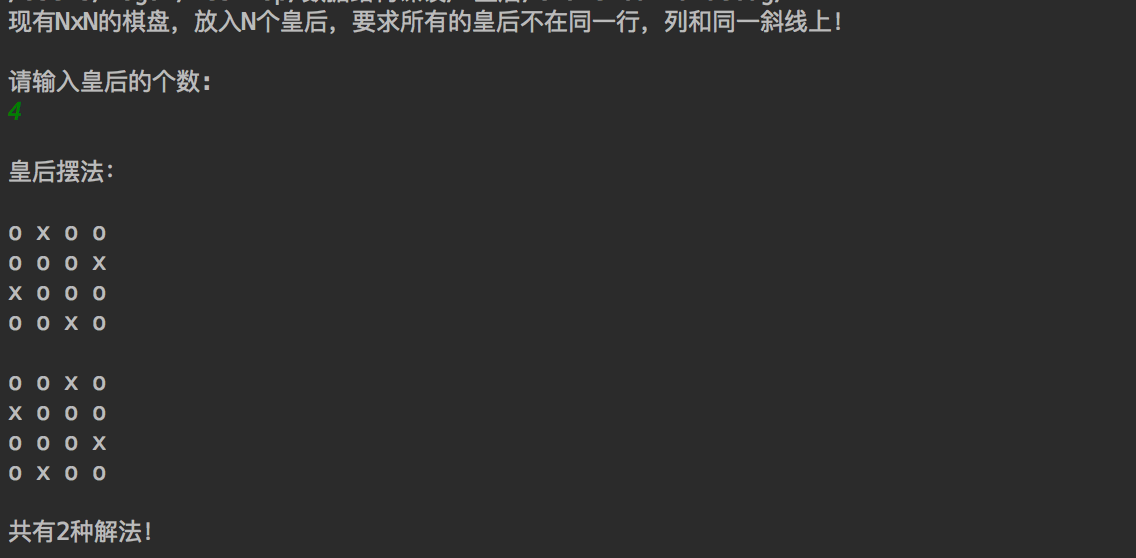
3.2.2 思路分析

八皇后在棋盘上分布的各种可能的格局数目非常大，约等于2的32次方种，但是，可以将一些明显不满足问题要求的格局排除掉。由于任意两个皇后不能同行，即每行只能放置一个皇后，因此将第i个皇后放在第i行上，这样在放置第i个皇后时，只要考虑它与前i-1个皇后处于不同列和不同对角线位置上即可。

3.2.3 具体实现如下：

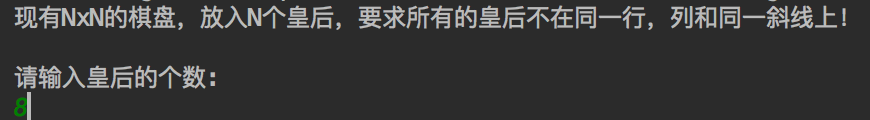


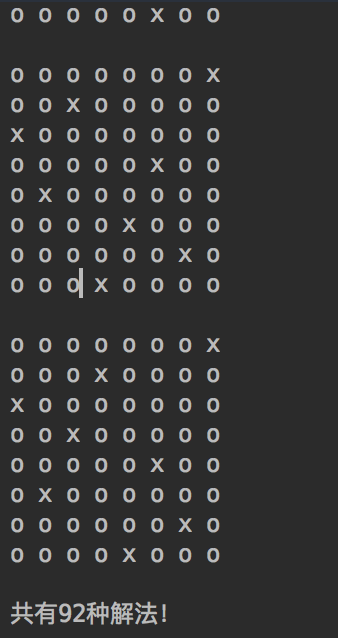
3.3 总体系统截屏示例



IV 测试

4.1搜索功能测试



由于输出结果过多，这里仅显示部分

V 总结

​ N皇后项目的基本功能至此已经全部实现了，在整个实验中,我学会利用了深度优先搜索的方法来实现N皇后问题的求解。

在本次项目中，我进一步的理解了深度优先搜索和广度优先搜索的不同：

1.在树的层次较深而且子节点数较多的情况下，广度优先搜索消耗内存十分严重，所以广度优先搜索适用于节点的子节点数量不多，并且树的层次不会太深的情况。

2.深度优先可以克服这个缺点，因为每次搜的过程，每一层只需维护一个节点。

3.深度优先搜索的缺点是难以寻找最优解，仅仅只能寻找有解。其优点就是内存消耗小，克服了刚刚说的广度优先搜索的缺点，而广度优先搜索所得到的解往往时候最优解。

VI 参考文献

1.数据结构课本