**项目说明文档**

**数据结构课程设计**

**——N皇后问题**

作 者 姓 名： 戴仁杰

学 号： 1951650

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 -](#_Toc91021754)

[1.1 背景分析 - 3 -](#_Toc91021755)

[1.2 功能分析 - 3 -](#_Toc91021756)

[2 设计 - 3 -](#_Toc91021757)

[2.1 数据结构设计 - 3 -](#_Toc91021758)

[2.2 类结构设计 - 3 -](#_Toc91021759)

[2.3 成员与操作设计 - 4 -](#_Toc91021760)

[2.3.1 NQueens类 - 4 -](#_Toc91021761)

[2.4 系统设计 - 4 -](#_Toc91021762)

[3 实现 - 5 -](#_Toc91021763)

[3.1 安排皇后位置功能的实现 - 5 -](#_Toc91021764)

[3.1.1 安排皇后位置功能流程图 - 5 -](#_Toc91021765)

[3.1.2 寻找路径功能核心代码 - 5 -](#_Toc91021766)

[3.2 总体功能的实现 - 6 -](#_Toc91021767)

[3.2.1 总体功能流程图 - 6 -](#_Toc91021768)

[3.2.2 总体功能核心代码 - 7 -](#_Toc91021769)

[3.2.3 总体功能截屏示例 - 7 -](#_Toc91021770)

[4 测试 - 8 -](#_Toc91021771)

[4.1 正常功能测试 - 8 -](#_Toc91021772)

[4.2边界测试 - 9 -](#_Toc91021773)

[4.2.1 没有解法的情况 - 9 -](#_Toc91021774)

[4.2.2 仅有1个皇后的情况 - 9 -](#_Toc91021775)

[4.3 出错测试 - 9 -](#_Toc91021776)

[4.3.1 输入的皇后个数为0或负数 - 9 -](#_Toc91021777)

[4.3.2 输入的皇后个数为非数字的字符 - 10 -](#_Toc91021778)

1 分析

1.1 背景分析

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中摆法。

1.2 功能分析

本实验拓展了N皇后问题，即皇后个数由用户输入，增强了程序的适用范围。

N皇后问题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，暴力求解效率很低。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。

对于计算机而言，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，选择该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。

综上所述，一个N皇后问题解决方案需要有根据输入初始化数组，回溯放置皇后，输出皇后位置的功能。

2 设计

2.1 数据结构设计

如上功能分析所述，皇后的个数N不确定，由用户输入给定，且数据一般不会发生插入删除等操作，所以使用动态内存申请一个二维数组存储。在放置皇后时按照先行后列的方式对每一行操作，采用回溯法解决所有可能的情形。

2.2 类结构设计

为了方便管理，将解决N皇后问题的程序封装成了一个NQueens类，类内包含了与N皇后问题有关的函数方法，在main函数中进行调用即可实现N皇后问题的求解。

2.3 成员与操作设计

### 2.3.1 NQueens类

NQueens类为N皇后类。N皇后类封装了根据输入初始化类的方法、判断当前位置是否能安放皇后的方法、回溯解决N皇后问题的方法和输出结果的方法。数据存储方面，NQueens类有一个二维指针board\*\*作为地图，并有4个一维bool指针用于分别记录当前行、列以及两条斜线是否有皇后。此外还有两个int类型变量用于记录皇后个数(queenNum)和摆放方法的数目(solutionNum)，便于后继输出结果。



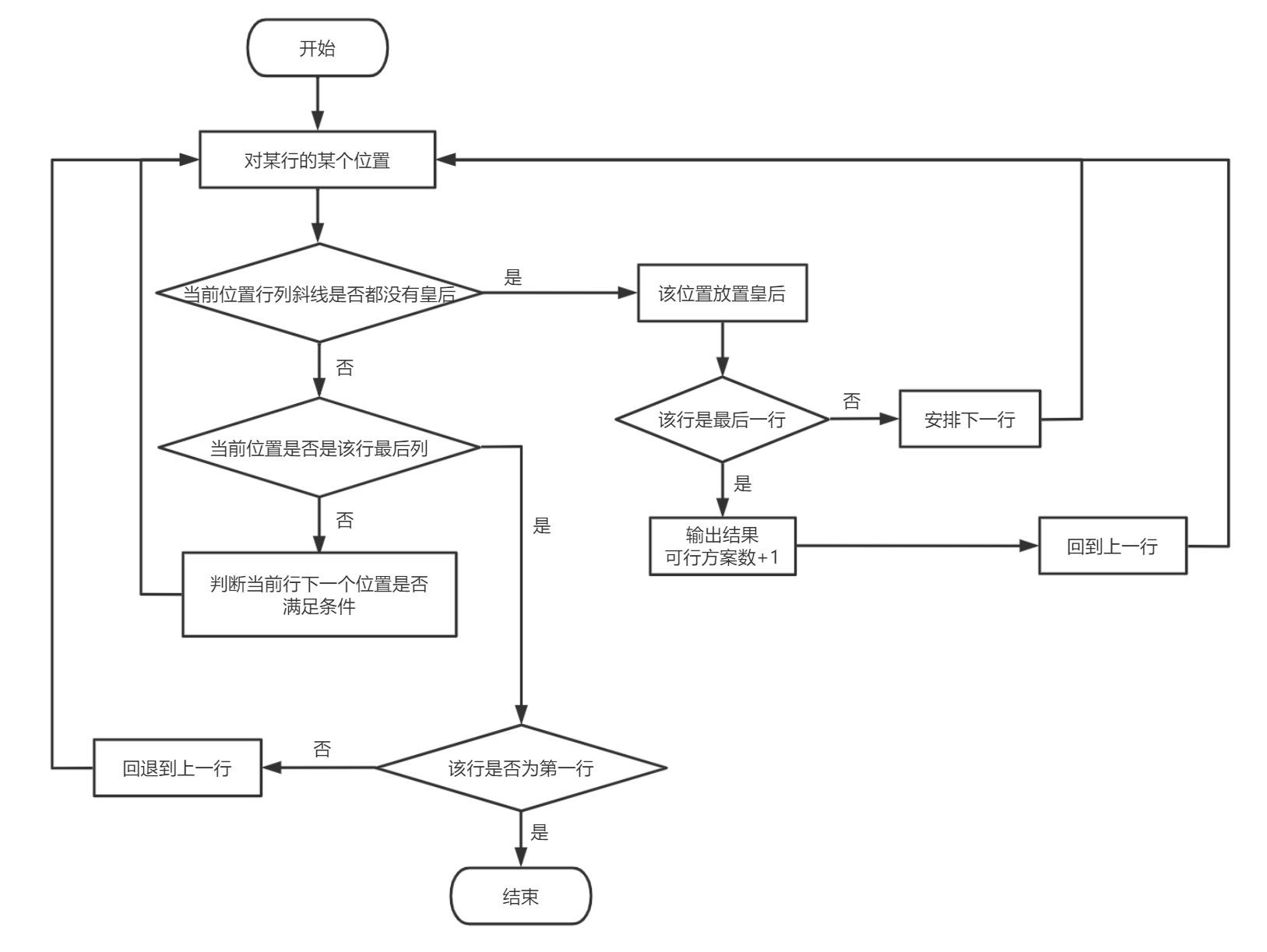
2.4 系统设计

程序开始后，首先初始化地图全部为0，并将4个bool类型数组全部赋值false。然后按照行的顺序，从第0行开始直到第n-1行用回溯法安排皇后的位置，棋盘上安排上皇后位置的地方置为X。每找到一种可行的方案就输出一次结果，并将解法数目+1，最后输出解法总数。

3 实现

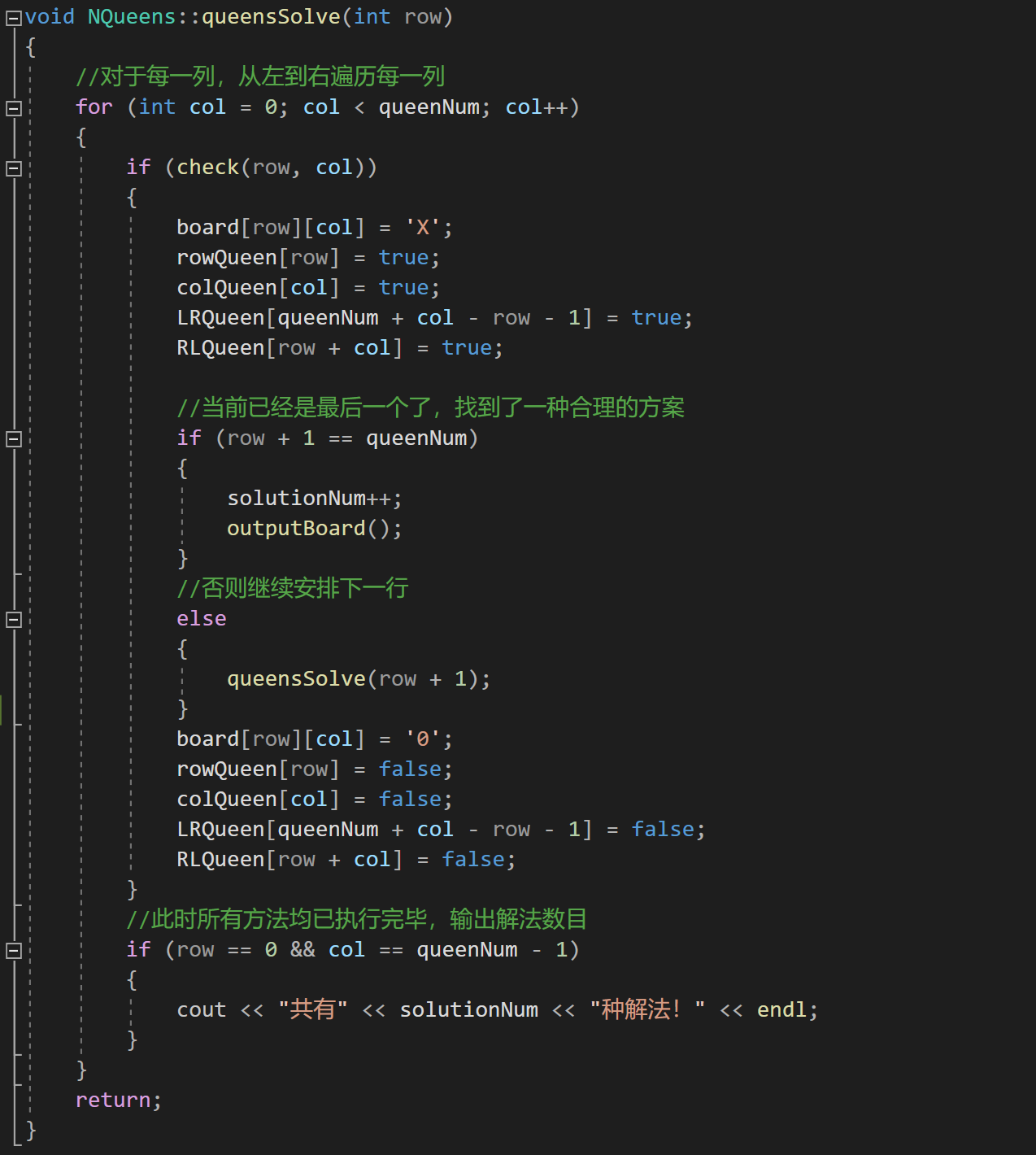
3.1 安排皇后位置功能的实现

3.1.1 安排皇后位置功能流程图



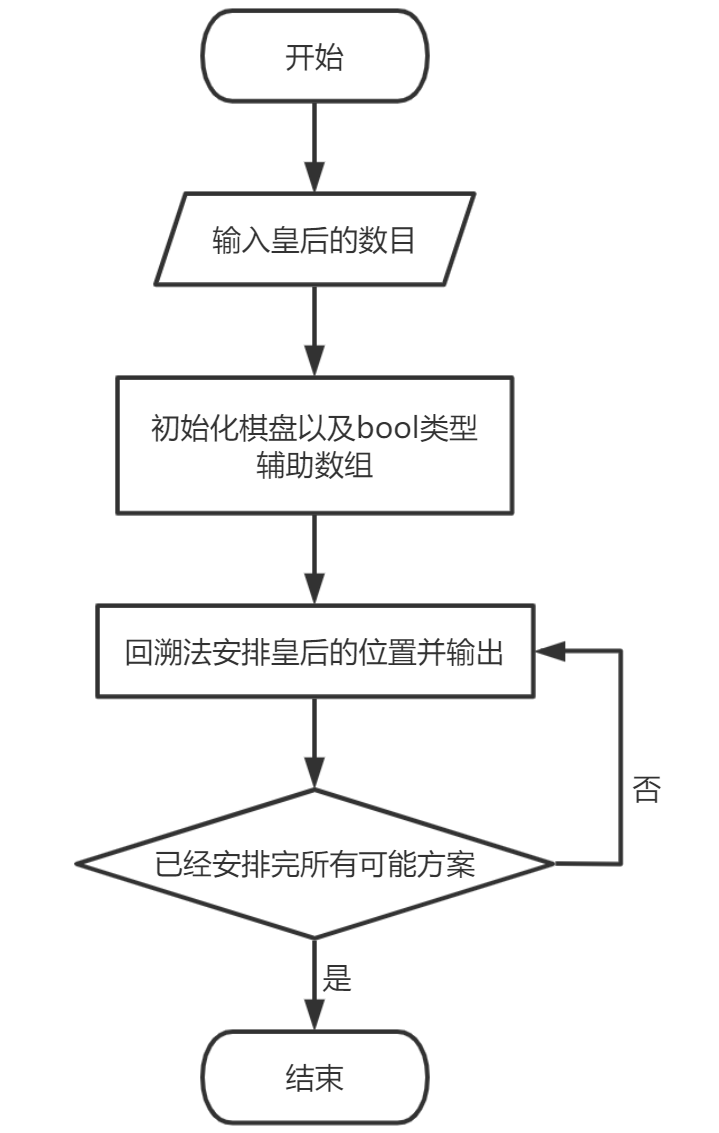
3.1.2 寻找路径功能核心代码

1.NQueens类的queenSolve(int row)函数



3.2 总体功能的实现

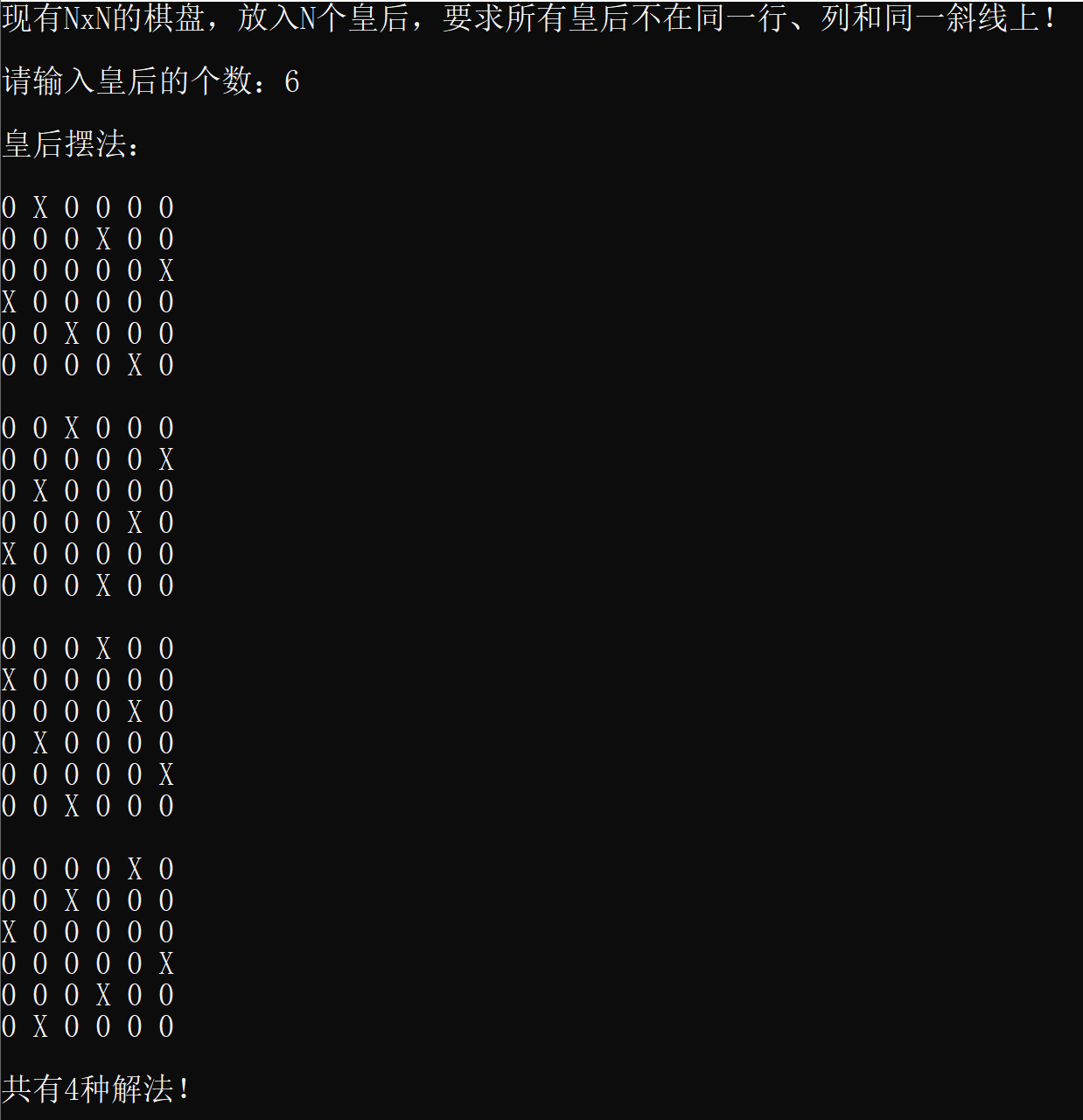
3.2.1 总体功能流程图



3.2.2 总体功能核心代码



3.2.3 总体功能截屏示例



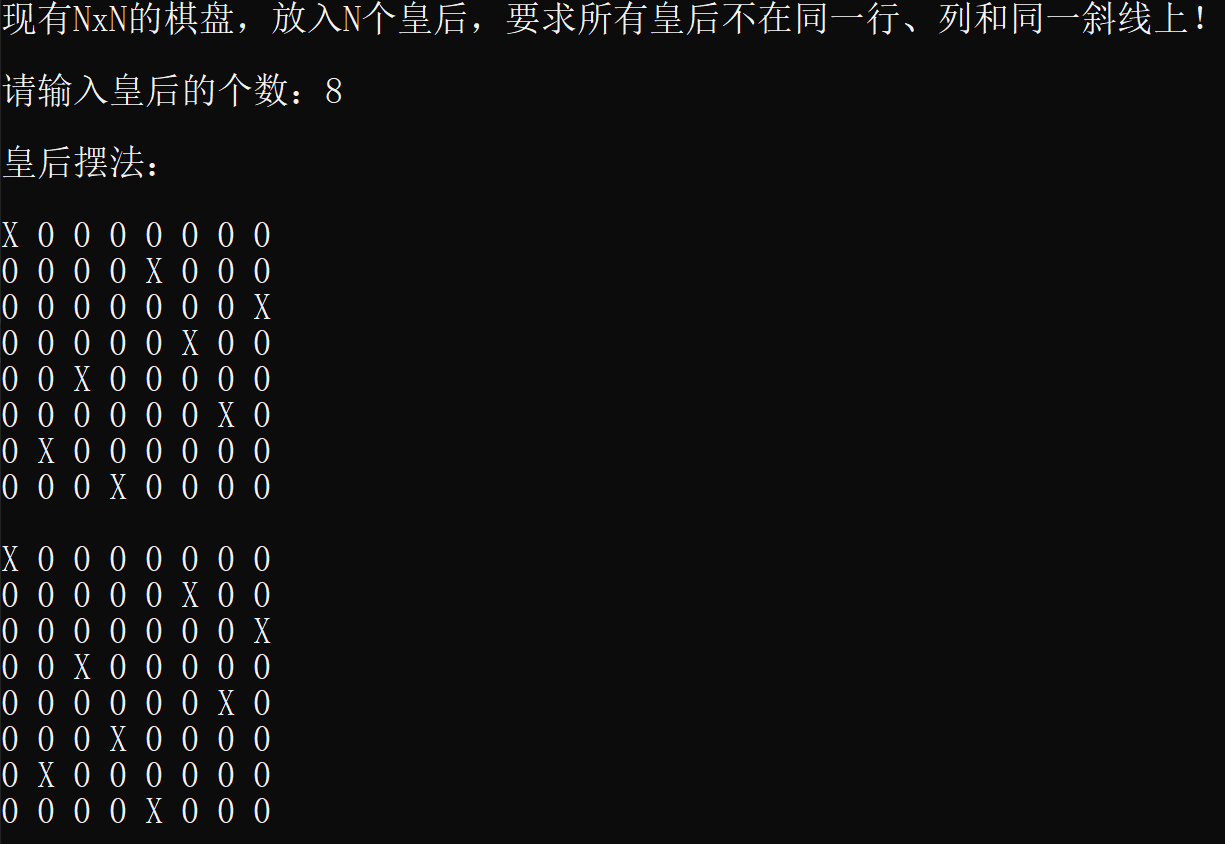
4 测试

4.1 正常功能测试

**测试用例**：皇后个数为8

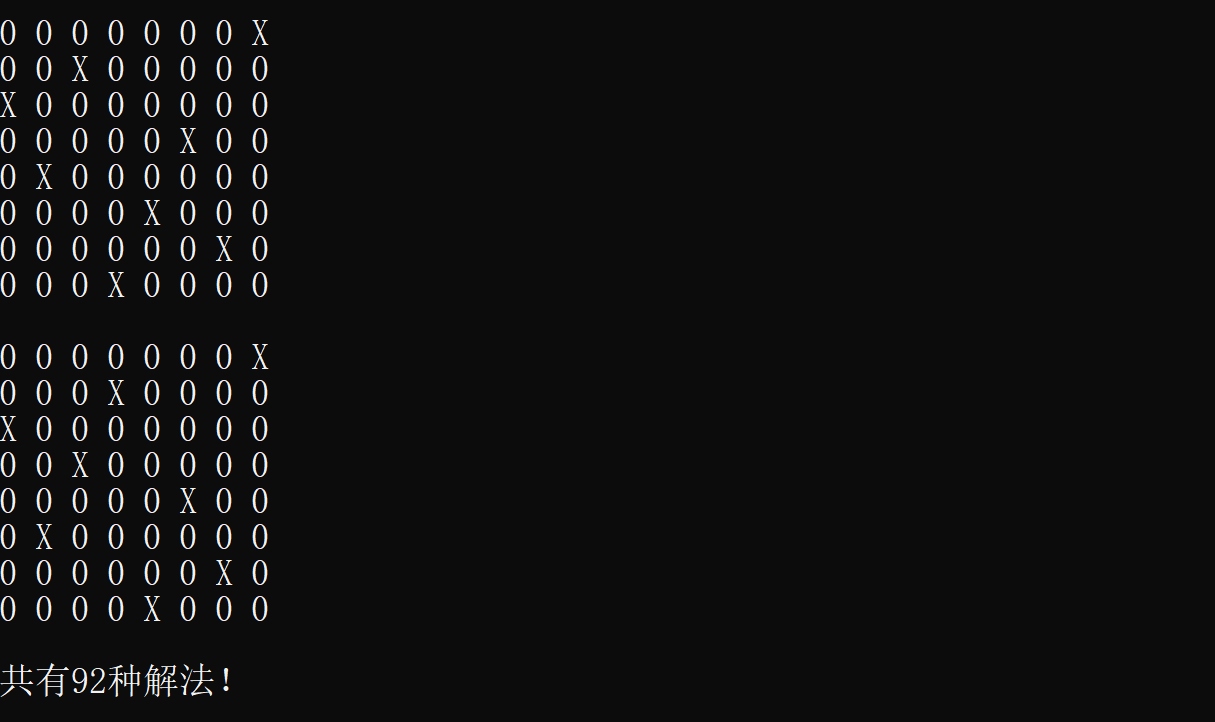
**预期结果**：输出92种排布结果

**实验结果：**符合预期（由于结果较多，截图中间省略了部分摆法）



……

……



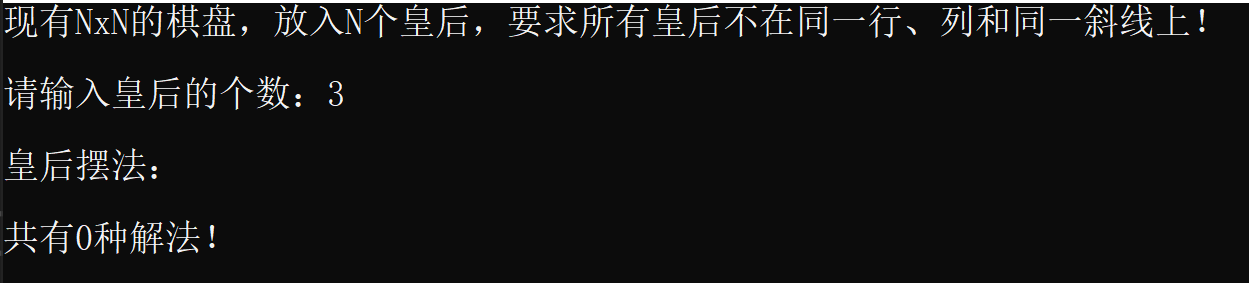
## 4.2边界测试

4.2.1 没有解法的情况

**测试用例**：皇后个数为3

**预期结果**：结果为0

**实验结果：**符合预期

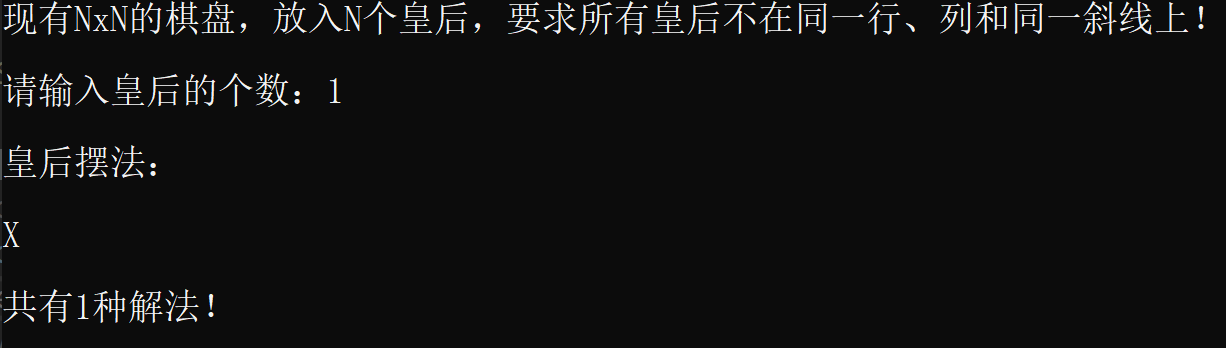


4.2.2 仅有1个皇后的情况

**测试用例**：皇后个数为1

**预期结果**：结果为1种排布结果

**实验结果：**符合预期



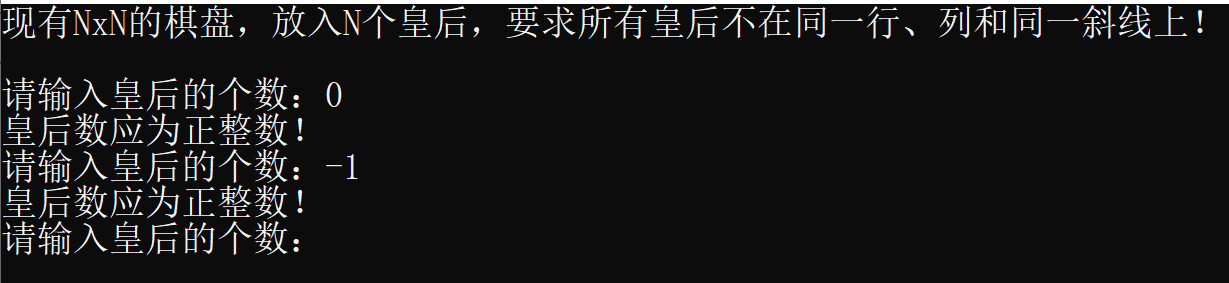
## 4.3 出错测试

4.3.1 输入的皇后个数为0或负数

**测试用例**：皇后个数为0、-1

**预期结果**：提示输入错误，并要求重新输入

**实验结果：**符合预期

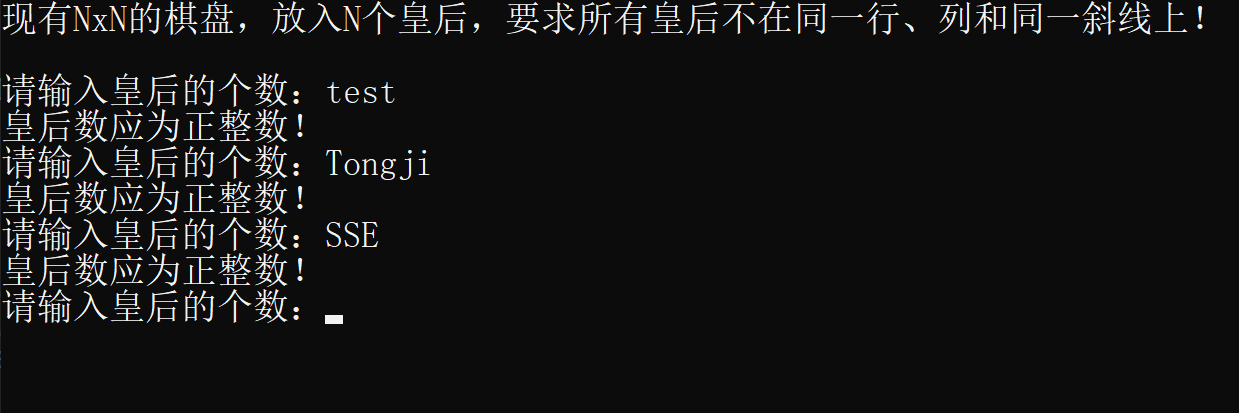


### 4.3.2 输入的皇后个数为非数字的字符

**测试用例**：皇后个数为test、Tongji、SSE

**预期结果**：提示输入错误，并要求重新输入

**实验结果：**符合预期



**测试用例：**皇后个数为 中文测试

**预期结果：**提示输入错误，并要求重新输入

**实验结果：**符合预期

