**数据结构课程设计《N皇后问题》项目说明文档**

2253893 苗君文 软件工程

**1. 项目简介**

**1.1. 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中摆法。

本实验拓展了N皇后问题，即皇后个数由用户输入。

**1.2. 项目要求**

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航上，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。

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。

**1.3. 输入格式**

本项目首先需要用户输入皇后的个数，在程序输出结果之后，用户可以输入y/Y/n/N来选择是否继续运行程序。

**1.4. 输出格式**

本项目首先输出N皇后问题的简介，而后根据用户输入的皇后个数，若无可行的解决方案，则输出相关提示信息；若有结果，则输出各种皇后的摆法（X为皇后，0为其他），并输出解法的总数。

**2. 设计思路**

**2.1. 递归解决N皇后问题的设计思路**

solveNQueens是解决N皇后问题的核心函数，采用递归和回溯的方法来搜索所有可能的解决方案。

函数的递归终止条件为：如果row等于 N，表示已经成功放置了N个皇后，找到一种解决方案。将solutions计数加一。并输出当前棋盘的解决方法。

函数的核心在于使用一个循环遍历当前行的每一列，尝试在每个位置放置皇后，对于每个位置调用isSafe函数检查是否可以安全放置皇后。

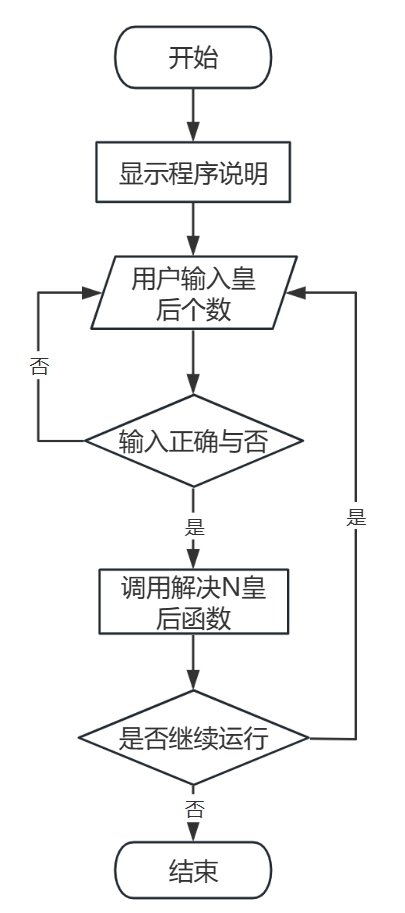
其中，isSafe函数检查同一列，左上对角线，右上对角线上是否有皇后。如果通过所有检查，说明当前位置不会受到其他皇后的威胁，则可以安全放置皇后。这样，solveNQueens函数就可以利用它确保在当前位置放置皇后。

如果在当前位置放置皇后后，无法找到解决方案，则进行回溯。将当前位置的皇后撤销，重新尝试其他可能型。

**2.2. 主函数设计**

主函数首先显示程序说明，然后通过一个无限循环接受用户输入，调用解决N皇后问题的函数，并询问用户是否继续运行程序。用户输入由dealInputError函数处理，确保输入的合法性。程序会根据用户的选择决定是继续运行还是退出。

其流程图如下：



**2.3. 输入错误处理设计**

**2.3.1. 单个参数的输入错误处理**

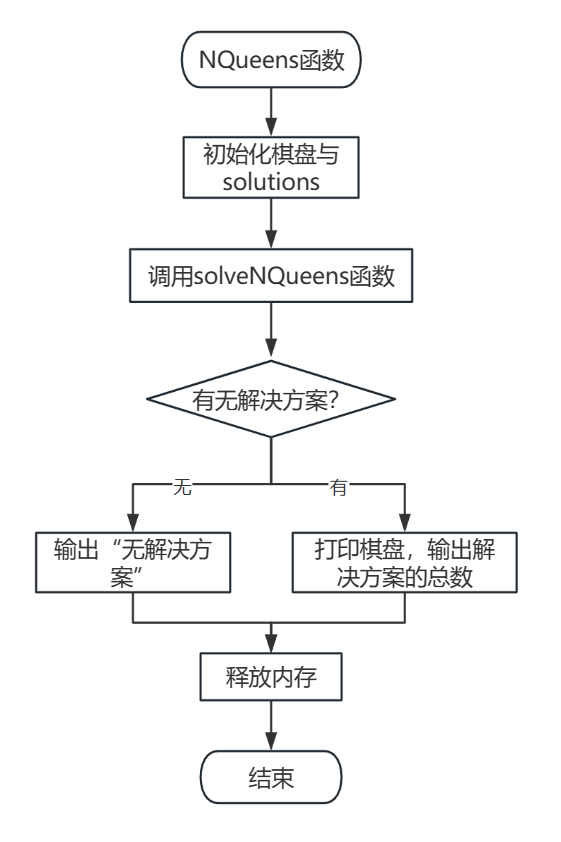
函数dealInputError使用 while 循环不断尝试获取用户输入，直到输入满足要求。使用 cin.fail() 及min、max来判断输入是否出错，检查输入是否在有效范围内。如果输入无效，则输出错误信息，清除输入缓冲区并忽略之后的字符，并重新输出输入的提示信息。通过一个字符来获取输入一个数之后的字符以检查输入字符的个数，若不正确，也输出错误信息，并重新输出输入的提示信息，执行相应清除操作。如果输入有效，则跳出循环。

此错误处理可以用在输入皇后的个数。与之类似，输入char型的单个参数可以用在用户输入选择是否继续运行程序。

**3. 功能实现**

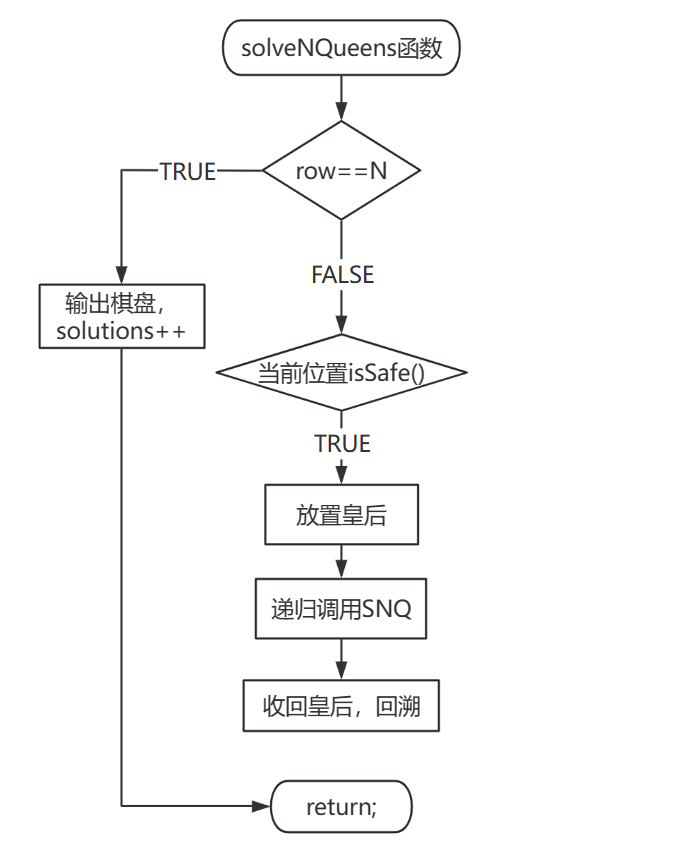
**3.1. NQueens函数**

在函数开始时，动态分配内存创建一个二维数组表示棋盘，同时初始化解决方案计数为零。传递创建的棋盘、起始行、棋盘大小（即皇后数量）以及解决方案计数作为参数，调用solveNQueens函数。在NQueens函数中输出找到的解决方案的数量。如果solveNQueens函数找到了解决方案，输出当前棋盘的布局，否则输出相关提示信息。在函数结束时，删除动态分配的内存，释放创建的二维数组。对应的流程图如下：



**3.2. solveNQueens函数**

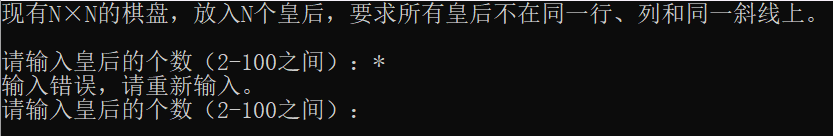
如果当前行等于 N，表示已经成功放置了 N 个皇后，找到了一个解决方案。增加解决方案计数，输出当前解决方案。这是递归的终止条件。使用循环遍历当前行的每一列，尝试在每个位置放置皇后。调用isSafe函数检查在当前位置放置皇后是否安全。如果安全，将皇后放置在当前位置，然后递归调用solveNQueens函数来处理下一行。如果在当前位置放置皇后后无法找到解决方案，则进行回溯。撤销当前位置的皇后，重新尝试其他可能性。其流程图如下：



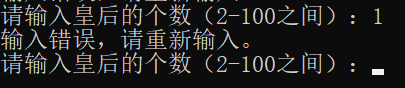
**4. 测试结果（包含边界测试）**

**4.1. 输入皇后的个数**

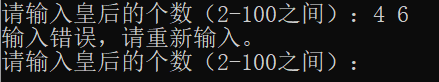
（1）输入类型错误



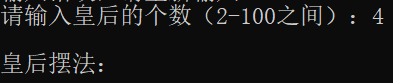
（2）输入的范围错误



（3）输入的个数超过一个

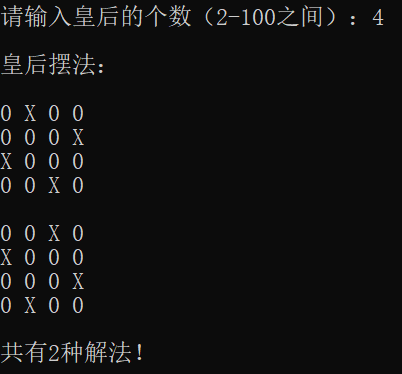


（4）输入正确即可进入下一环节

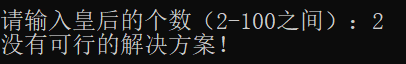


**4.2. 输出皇后的摆法即解法个数**

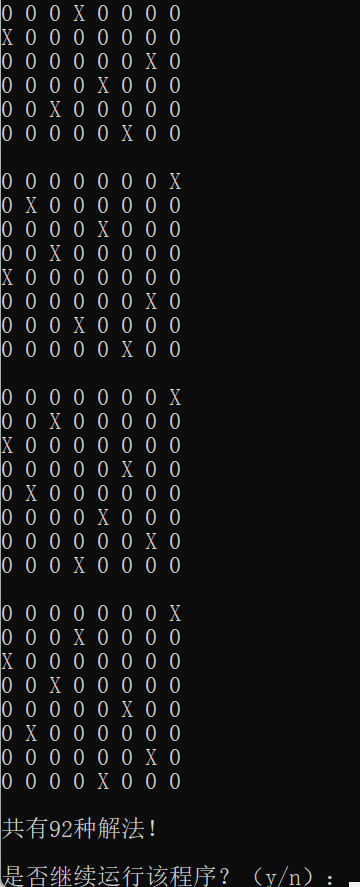
（1）N=4



（2）N=2



（3）N=8



**4.3. 输入以确定是否继续运行程序**

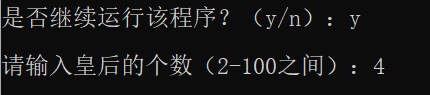
（1）输入内容非法

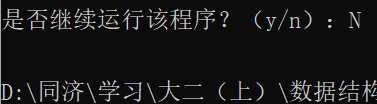


（2）输入个数非法



（3）输入正确（大小写均可）即可执行相应的操作





**4.4. 程序的测试结果总览（此处仅展示输入正确有效的情况）**

