项目说明文档

数据结构课程设计

——N皇后问题

作 者 姓 名： 杨宇琨

学 号： 2252843

指 导 教 师： 张颖

学院、 专业： 软件学院 软件工程

同济大学

Tongji University

1 分析

## 1.1 背景分析

八皇后问题是一个经典的数学问题，涉及到组合数学、排列组合、递归算法等领域。该问题在计算机科学中被广泛应用，作为回溯算法的经典问题之一。解决八皇后问题需要找到一种在8×8的棋盘上放置8个皇后，使得它们互相不能攻击的摆放方式。这个问题的解决对于理解回溯算法、深度优先搜索以及递归算法有着重要的意义。因此，解决八皇后问题不仅是一种数学问题，也是一种对算法设计和实现能力的考验。

## 1.2 功能分析

1. 棋盘初始化：创建一个6×6的棋盘，用于摆放皇后。

2. 回溯算法实现：设计回溯算法，通过递归实现对棋盘上皇后位置的搜索和摆放。

3. 攻击判断：在每一次摆放皇后时，需要判断当前位置是否受到其他皇后的攻击，即是否处于同一行、同一列或者同一对角线上。

4. 摆放皇后：找到合适的位置，将皇后摆放在棋盘上，并标记已经占据的位置。

5. 回溯与剪枝：在搜索过程中，及时发现冲突并回溯，或者进行剪枝操作，以提高搜索效率。

6. 统计结果：记录所有满足条件的摆放方式，并统计总数。

以上功能的实现将帮助解决八皇后问题，展示回溯算法在解决组合优化问题中的应用，同时也有助于加深对算法设计与实现的理解。

# 2 设计

# 2.1.1 \*\*主函数main()\*\*：

# - 主函数负责整个程序的流程控制，包括输入皇后个数、调用solveNQueens函数解决问题、显示解法数量以及控制程序的终止或继续运行。

# 2.1.2 \*\*solveNQueens函数\*\*：

# - solveNQueens函数用于解决八皇后问题，它调用了solveNQueensUtil函数来递归地尝试摆放皇后，寻找解法。

# - 函数内部使用了动态内存分配来创建二维数组表示棋盘，通过调用solveNQueensUtil函数来解决问题，并在解决完成后释放内存。

# 2.1.3. \*\*solveNQueensUtil函数\*\*：

# - solveNQueensUtil函数是实际用于递归摆放皇后的工具函数，它通过递归的方式尝试每一行中的每一个位置，并在找到解法时打印输出，并统计总共的解法数量。

# 2.1.4. \*\*printSolution函数\*\*：

# - printSolution函数用于打印棋盘的解法，以可视化的方式展示出每一种解法的棋盘状态。

# 2.1.5. \*\*isSafe函数\*\*：

# - isSafe函数用于判断在当前位置放置皇后是否安全，即不会被其他皇后攻击到，它用于solveNQueensUtil函数中的位置合法性判断。

# 2.1.6. \*\*input函数：

# - input函数用于输入数据，对用户的输入进行验证并返回合法的输入值。

2.2 系统设计

高层次系统设计

1.输入处理：程序首先接收用户输入的皇后个数，对输入进行验证和处理，确保输入合法。

2.问题解决：通过调用solveNQueens函数，解决八皇后问题，该函数内部调用solveNQueensUtil函数实现具体的问题求解。

3.结果输出：在求解完成后，输出满足条件的解法数量，并提供继续运行或者退出程序的选择。

低层次系统设计

1.数据表示：通过二维字符数组表示棋盘及皇后位置，使用字符'X'表示皇后位置，'0'表示空位置。

2. 递归求解：solveNQueensUtil函数使用递归的方式尝试每一个位置的摆放，直到找到所有解法。

3. 位置合法性判断：isSafe函数用于判断在当前位置放置皇后是否安全，即不会被其他皇后攻击到。

4. 内存管理：使用动态内存分配创建二维数组表示棋盘，解决问题后及时释放内存，避免内存泄漏。

系统交互设计

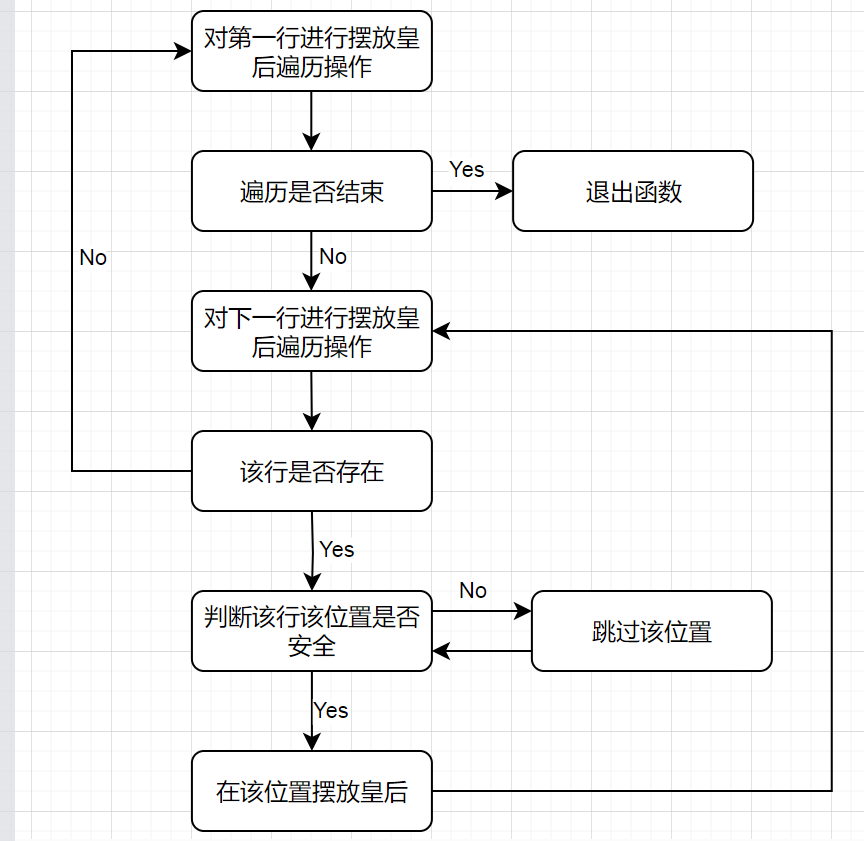
1. 输入交互：程序通过控制台接收用户输入，对输入进行验证和处理，要求用户按照规定的范围输入皇后个数。

2. 结果展示：在求解完成后，程序通过控制台输出所有满足条件的解法，并提供继续运行或者退出程序的选择。

# 3 实现

## 3.1 N皇后solveNQueensUtil的实现

### 3.1.1solveNQueensUtil功能流程图



### 3.1.2 solveNQueensUtil核心代码

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Function Name:solveNQueensUtil

Function:To place the queen safely until we reached the bottom

Input Parameters:board row size amount of result

Return Value:

Documentation：

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void solveNQueensUtil(char\*\* board, int row, int N,int &total)

{

//If we reached the bottom,we it means we have placed all the queens

if (row == N) {

printSolution(board, N);//Print it out

total++;

return ;//Termination condition

}

//Try every position in every row

for (int i = 0; i < N; i++) {

if (isSafe(board, row, i, N)) {

board[row][i] = 'X';//If queen can be placed in this position

solveNQueensUtil(board, row + 1, N, total);//Place the queen and go to the next row

board[row][i] = '0';//After we place all the queen reset the position

}

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Function Name:isSafe

Function:To judge whether the queen can be placed in this position

Input Parameters:board position(row and column) size

Return Value:true for safe,false for not safe

Documentation：

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

bool isSafe( char\*\* board, int row, int col, int N)

{

//If at the same column,not safe

for (int i = 0; i < row; i++) {

if (board[i][col] == 'X') {

return false;

}

}

//If the same diagonal,not safe(check left)

for (int i = row, j = col; i >= 0 && j >= 0; i--, j--) {

if (board[i][j] == 'X') {

return false;

}

}

//If the same diagonal,not safe(check right)

for (int i = row, j = col; i >= 0 && j < N; i--, j++) {

if (board[i][j] == 'X') {

return false;

}

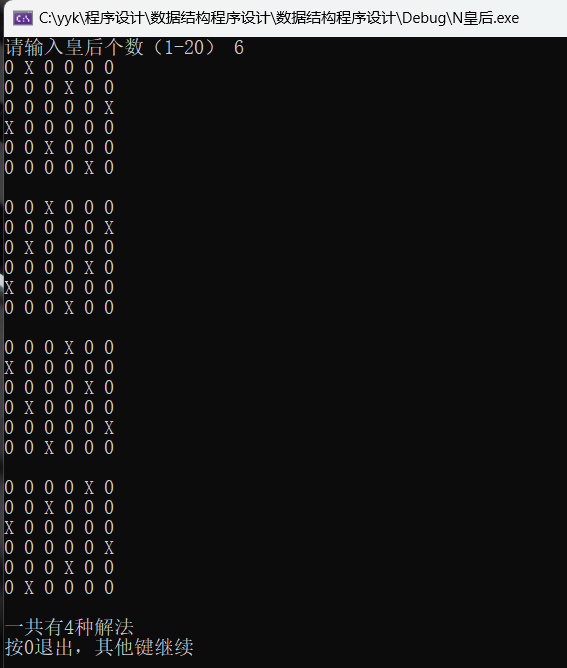
}

//Is safe

return true;

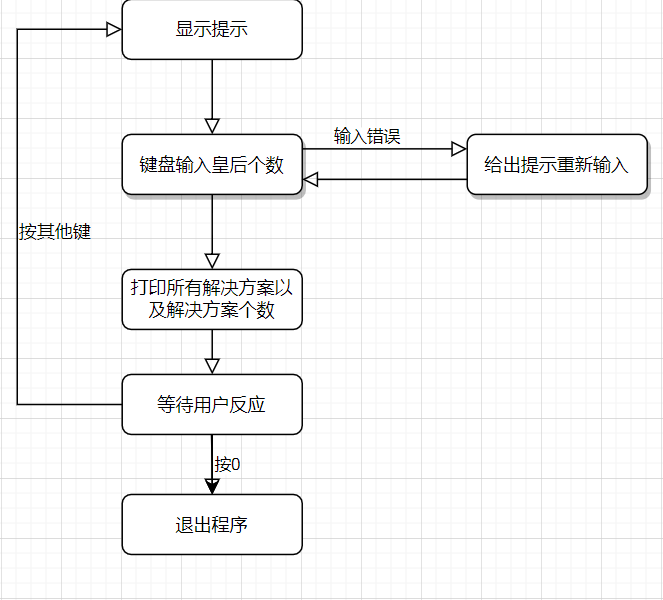
}

### 3.1.3 插入功能截屏示例



## 3.2 总体系统的实现

### 3.2.1 总体系统流程图



### 3.2.2 总体系统核心代码

int N;

while (true) {

//Clear our screen

system("cls");

std::cout << "请输入皇后个数（1-20） ";

N = input<int>(l\_limit, U\_limit);

solveNQueens(N);

//Display the prompt

std::cout << "按0退出，其他键继续" << std::endl;

char controller;//Create a variable to hold the choice we input

controller = \_getch();//\_getch can avoid displaying undesired input on the screen

if (controller == '0') {

std::cout << "程序结束！" << std::endl;

return 0;

}

}//end of while

### return 0;

### 3.2.3 总体系统截屏示例

