**N-Queens Problem Solver Report**

**1. Introduction**

The N-Queens problem is a classic combinatorial problem where N queens must be placed on an N×N chessboard such that no two queens attack each other. This problem has significant applications in backtracking and constraint satisfaction problems.

**2. Problem Statement**

Given an integer N, the objective is to place N queens on an N×N chessboard so that no two queens threaten each other. This means that no two queens can share the same row, column, or diagonal.

**3. Algorithm Explanation**

The N-Queens problem is typically solved using a backtracking algorithm. The approach is as follows:

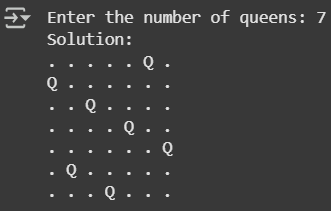
1. Start with an empty board.  
   2. Place a queen in the first available row.  
   3. Move to the next row and place another queen in a safe column.  
   4. If a conflict occurs, backtrack to the previous row and try a different column.  
   5. Repeat until all queens are placed successfully.  
   6. If a solution is found, store it; otherwise, try different placements recursively.

**4. Implementation Details**

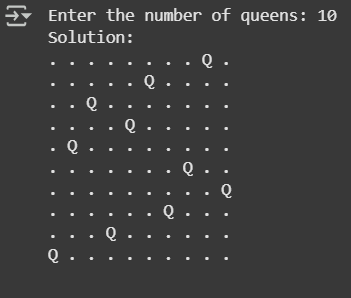
The implementation is done using Python and executed in Google Colab. The core logic is built using recursion and backtracking. The solution set is printed in a matrix form, where 'Q' represents a queen and '.' represents an empty space.

**5. Results & Outputs**

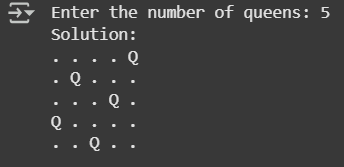
The program successfully finds all valid solutions for a given N. For example, for N=7, one of the possible solutions is:



for N= 10, one of the possible solutions is:



for N= 5, one of the possible solutions is:



**6. Conclusion**

The N-Queens problem demonstrates the power of backtracking in solving complex constraint satisfaction problems. This implementation successfully finds solutions for various N values and can be extended further for optimization or graphical visualization.