Documentation: N-Queens II

Overview:

The N-Queens II problem is a classic chessboard puzzle where the goal is to place N queens on an N x N chessboard in such a way that no two queens threaten each other. In this context, "threaten" means that no two queens share the same row, column, or diagonal.

Problem Statement:

Given an integer N, representing the size of the chessboard (N x N), the task is to find the total number of distinct solutions to the N-Queens problem.

Input:

- n (int): An integer representing the size of the chessboard (N x N). It must satisfy the constraint 1 <= n <= 9.

Output:

- int: The number of distinct solutions to the N-Queens problem for the given input.

Example:

Example 1:

Input: n = 4

Output: 2

Explanation: There are two distinct solutions to the 4-queens puzzle as shown.

Example 2:

Input: n = 1

Output: 1

Approach:

The solution utilizes backtracking to explore all possible configurations of queen placements on the chessboard while ensuring that no two queens threaten each other.

- The `is_not_under_attack` function checks whether a queen placed at a specific row and column is not threatened by any other queen already placed on the board.
- The `backtrack` function recursively explores valid queen placements on each row of the chessboard and counts the total number of valid solutions.
- The main function `totalNQueens` initiates the backtracking process by calling the `backtrack` function.

Complexity Analysis:

- **Time Complexity:** The time complexity of the solution is O(N!), where N is the size of the chessboard.
- **Space Complexity:** The space complexity is O(N), where N is the size of the chessboard, due to the space used by the call stack during recursion.