****

**Superior University Lahore**

***Lab Task # 4***

**Name:** Muhammad Abdullah Kashif

**Roll No**: SU92-BSAIM-F23-070

**Instructor**: Sir Rasikh Ali

# Course: Programming for Artificial Intelligence (Lab)

N-Queens Problem

**Problem Explanation:**

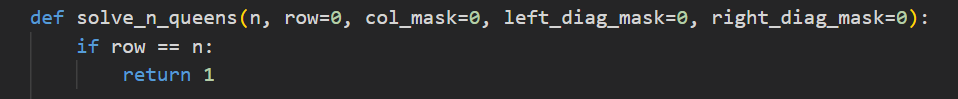
The N-Queens problem is a classic combinatorial problem in which we must place N queens on an N × N chessboard so that no two queens attack each other. This means:  
- No two queens can be in the same row.  
- No two queens can be in the same column.  
- No two queens can be on the same diagonal.  
  
The goal is to determine the total number of valid ways to place N queens on the board.

**Code Explanation:**

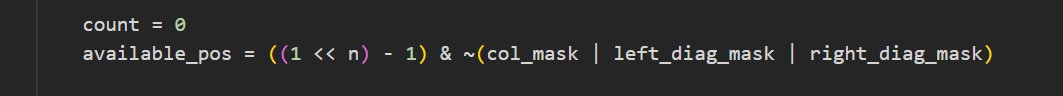
The given function `solve\_n\_queens(n, row=0, col\_mask=0, left\_diag\_mask=0, right\_diag\_mask=0)` solves the N-Queens problem using bitwise operations and backtracking.

**Step-by-Step Working:**

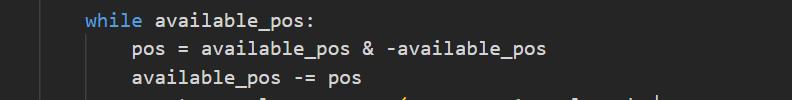
**1. Base Case (Solution Found)**

  
  
If all rows are successfully filled with queens, a valid solution is found, so we return 1.

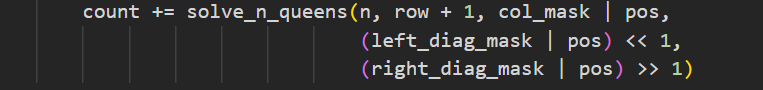
**2. Finding Available Positions**

  
  
 This line calculates the available columns in which we can place a queen in the current row. **col\_mask, left\_diag\_mask**, and **right\_diag\_mask** track occupied columns and diagonals.

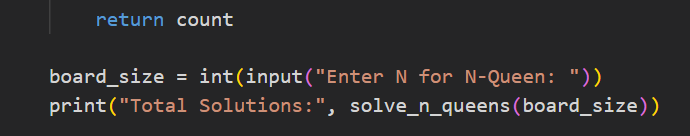
**3. Iterate Through Available Positions**

  
  
We select the rightmost available position for the queen and remove it from the available positions.

**4. Recursive Backtracking**

  
  
We place a queen in the selected position and proceed to the next row (`row + 1`).  
The column and diagonal masks are updated accordingly.

**5. Returning the Final Count & User Input and Execution**

****

The function returns the total number of solutions found. The program takes user input for N and calculates the total number of solutions.

**Example Output**

Enter N for N-Queen: 4  
Total Solutions: 2