

**Superior University Gold Campus**

**PAI Lab Task #4**

Name: Syed Ejiz Ul Hassan Kazmi

Roll. No: SU92-BSDSM-F23-025

Section: BSDS-4A

Instructor: Sir Rasikh

### ****N-Queens problem (Dynamic)****

### ****Step 1: Understanding the Problem****

The **N-Queens problem** involves placing **N queens** on an **N×N chessboard** such that:

* No two queens attack each other.
* Queens should not be in the same row, column, or diagonal.

The goal is to find all possible solutions.

### ****Step 2: Function Definitions****

#### **1. N\_Queens(n) Function**

This function takes n (the size of the board) as input and returns all possible solutions.

#### **2. is\_safe(row, colmn, dp**) **Function**

* Checks if placing a queen at position (row, colmn) is safe.
* The dp list keeps track of queen placements.
* It ensures that no queen is in the same column or diagonal.

#### **3. solve(row, dp, solutions**) **Function**

* Uses **backtracking** to place queens row by row.
* If all queens are placed, the solution is stored.
* Otherwise, it tries all columns and checks if placing a queen is safe.
* If safe, it proceeds to the next row.
* If not, it backtracks.

### ****Step 3: Main Logic****

1. **Get User Input**
   * The user enters a board size (n).
   * If input is invalid (not an integer or less than 1), it asks again.
2. **Call N\_Queens(n) Function**
   * If a solution exists, it prints the board.
   * Otherwise, it informs the user that no solution exists.

### ****Step 4: Output Formatting****

* The board is displayed in a user-friendly format.
* Columns are labeled, and queens (Q) are placed accordingly.

### Output:

### 

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_