**Task – 4**

N-Queens Problem using Backtracking

Objective: Place N queens on an N×N chessboard such that no two queens attack each other.

Algorithm: Backtracking to explore all valid board configurations.

Functions: - print\_board(board, n): Display board state. - is\_safe(board, row, col): Check if a queen can be placed at (row, col). - solve\_n\_queens\_util(board, row, n, solutions): Recursive utility for backtracking. - solve\_n\_queens(n): Initialize board, store all solutions, print results.

Input: Integer N representing the number of queens.

Validation: Ensure user inputs a valid number.

State Representation: board[i] = column index of queen in row i.

Backtracking Logic: - Place queen row by row. - Check safety in same column and diagonals. - If safe, place queen and recurse to next row. - Backtrack if no valid placement possible.

Output: - Print total number of solutions. - Display each solution board.

Loop: - Repeatedly ask user for N. - Option to exit with ‘exit’. - Validate input before solving.

Example Interaction: - User inputs 4 → Displays 2 solutions for 4-Queens. - User inputs ‘exit’ → Program terminates with message.

