

**NAME: SABIKA ZAHRA**

**ROLL NUMBER: 074**

**CLASS: BSAI**

**SECTION: ALPHA**

**SEMESTER: 4**

**SUBMITTED TO: SIR RASIKH ALI**

**DATE: 2,27,2025**

**N-Queens Problem Solution Report**

**Problem Statement**

The N-Queens problem is a classic combinatorial problem that requires placing **N queens** on an **N × N chessboard** such that no two queens threaten each other. This means that:

1. No two queens can be in the same row.
2. No two queens can be in the same column.
3. No two queens can be on the same diagonal.

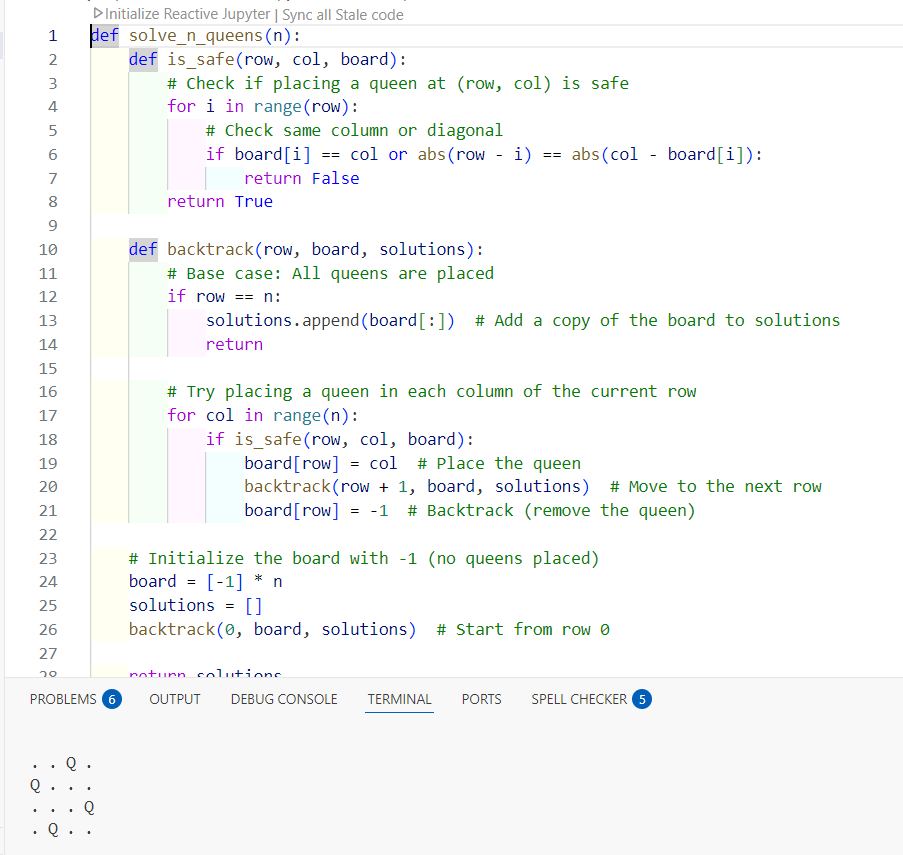
**Approach Used**

The solution is implemented using a **backtracking algorithm**. The key steps in solving the problem are:

1. **Recursive Backtracking Function:**
   * The function backtrack(row, board, solutions) attempts to place a queen in each row, ensuring it does not conflict with previously placed queens.
2. **Safety Check:**
   * The function is\_safe(row, col, board) checks whether placing a queen at (row, col) is valid based on previous queen placements.
   * This ensures no queens are placed in the same column or diagonal.
3. **Recursive Placement and Backtracking:**
   * If a queen can be safely placed, it is added to the board.
   * The function then recursively attempts to place queens in the next row.
   * If a valid solution is found (all queens placed), it is stored in the solutions list.
   * If no valid placement is possible, the function **backtracks** by removing the last placed queen and trying the next available column.

**Implementation Details**

* The board is represented using a **1D array**, where board[row] = col means a queen is placed at (row, col).
* The recursive function explores all valid placements and stores all possible solutions.
* The function print\_solutions() displays each solution as an **N × N grid** with 'Q' representing a queen and '.' representing an empty square.

**Complexity Analysis**

The time complexity of the N-Queens problem is **O(N!)**, as the algorithm explores multiple permutations of queen placements. However, pruning invalid states significantly reduces the number of possibilities.

**Conclusion**

The **backtracking approach** efficiently finds all valid solutions to the N-Queens problem. By systematically placing queens and eliminating invalid configurations early, the algorithm ensures correctness while minimizing unnecessary computations. The implementation successfully demonstrates a fundamental problem-solving strategy in combinatorial optimization and constraint satisfaction problems.