# **Leetcode 52 – N-Queens II**

## Problem Understanding

**Leetcode 52. N-Queens II**  
You are asked to return the total number of distinct solutions to the N-Queens problem, where **N queens** must be placed on an **N×N chessboard** such that **no two queens attack each other** (no same row, column, or diagonal).

Unlike Leetcode 51 (which asks for board configurations), this problem asks only for the **count** of valid solutions.

## Optimized Java Solution (Backtracking with Sets)

public class Solution {

private int count = 0;

public int totalNQueens(int n) {

Set<Integer> cols = new HashSet<>();

Set<Integer> diagLeft = new HashSet<>(); // row - col

Set<Integer> diagRight = new HashSet<>(); // row + col

backtrack(0, n, cols, diagLeft, diagRight);

return count;

}

private void backtrack(int row, int n,

Set<Integer> cols,

Set<Integer> diagLeft,

Set<Integer> diagRight) {

if (row == n) {

count++;

return;

}

for (int col = 0; col < n; col++) {

int leftDiag = row - col;

int rightDiag = row + col;

if (cols.contains(col) || diagLeft.contains(leftDiag) || diagRight.contains(rightDiag))

continue;

// Choose

cols.add(col);

diagLeft.add(leftDiag);

diagRight.add(rightDiag);

// Explore

backtrack(row + 1, n, cols, diagLeft, diagRight);

// Un-choose

cols.remove(col);

diagLeft.remove(leftDiag);

diagRight.remove(rightDiag);

}

}

}

## Dry Run Using Table (n = 4)

We trace one valid path that reaches row == 4.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Step | Row | Col | cols | diagLeft (r-c) | diagRight (r+c) | Action |
| 1 | 0 | 1 | {1} | {-1} | {1} | place Q |
| 2 | 1 | 3 | {1,3} | {-1,-2} | {1,4} | place Q |
| 3 | 2 | 0 | {0,1,3} | {2,-1,-2} | {1,2,4} | place Q |
| 4 | 3 | 2 | {0,1,2,3} | {2,1,-1,-2} | {1,2,4,5} | place Q |
|  |  |  |  |  |  | ✅ row == 4 → count++ |
| 5 |  |  | backtrack | remove col 2 | remove r-c, r+c | backtrack |

You continue trying other paths recursively until all have been explored.

## Time / Space Complexity

|  |  |
| --- | --- |
| Complexity | Explanation |
| ⏱ Time | **O(N!)** — There are N choices for the first row, N-1 for the second, and so on. |
| 💾 Space | **O(N)** — For each of the sets: cols, diagLeft, and diagRight plus recursion stack. |

## Alternate Approaches

|  |  |
| --- | --- |
| Approach | Notes |
| ❌ Brute Force | Generate all N^N boards → extremely slow |
| ✅ Bitmasking (Advanced) | Optimize sets with integer bitmasks for O(1) ops. Ideal for n ≤ 32 |
| ✅ Array instead of Set | Use boolean[n] and boolean[2n-1] arrays for faster access |