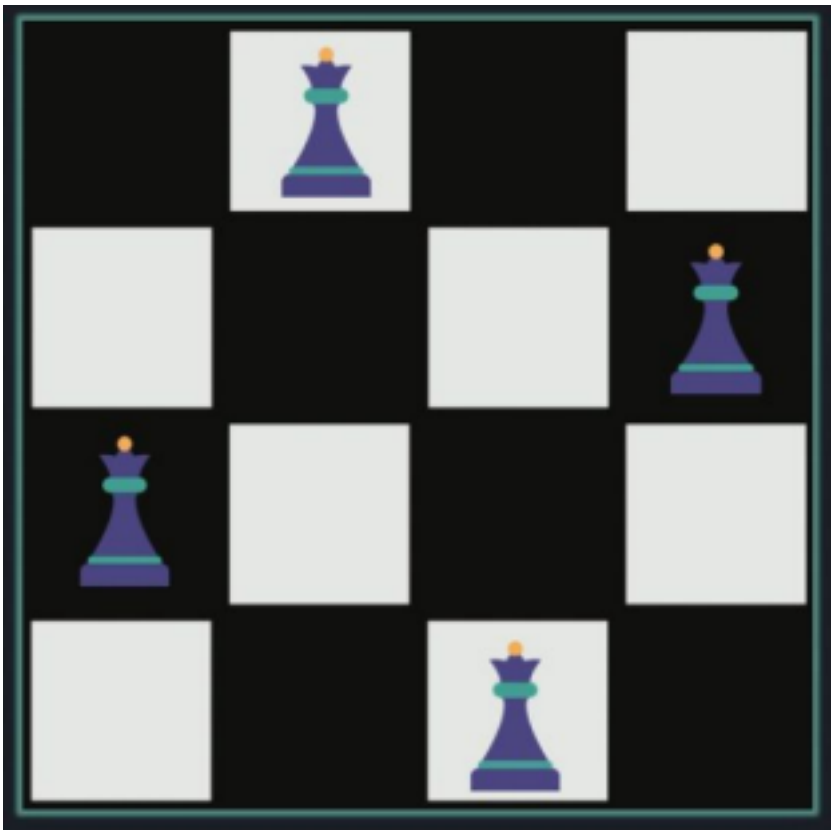
Recursion - IV (Backtracking)

N-Queen Problem

You have to place n queens on an nxn chessboard.

Idea: Try all the possible positions for the queen. isSafe functions check whether the current position is safe or not.



Input:

A single integer n

Output:

n x n matrix, where one denotes queen, and 0 for an empty cell.

Test Case 1:

Input:

4

Output:

0 1 0 0

0 0 0 1

1 0 0 0

0 0 1 0

```
bool isSafe(int** arr, int x, int y, int n) {
    for (int row = 0; row < x; row++) {
        if (arr[row][y] == 1) {
            return false;
        }
    }
    int row = x;
    int col = y;
    while (row >= 0 && col >= 0) {
        if (arr[row][col] == 1) {
            return false;
        }
        row--;
        col--;
    }
    row = x;
    col = y;
    while (row >= 0 && col < n) {
        if (arr[row][col] == 1) {
            return false;
        }
        row--;
        col++;
    }
    return true;
}
```

```
bool nQueen(int** arr, int x, int n) {
    if (x >= n) {
        return true;
    }
    for (int col = 0; col < n; col++) {
        if (isSafe(arr, x, col, n)) {
            arr[x][col] = 1;

            if (nQueen(arr, x + 1, n)) {
                return true;
            }

            arr[x][col] = 0; //backtracking
        }
    }
    return false;

}
```

Time Complexity: $O(2^n)$

Space Complexity: $O(2^n)$