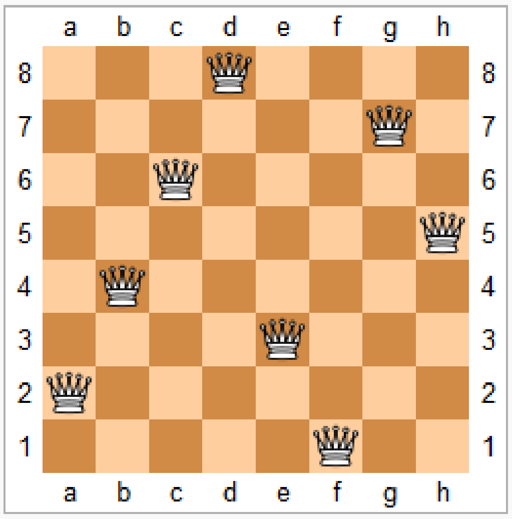
N-Queens II

Follow up for N-Queens problem.

Now, instead outputting board configurations, return the total number of distinct solutions.



One solution to the eight queens puzzle

Solution 1

```
/**
 * don't need to actually place the queen,
 * instead, for each row, try to place without violation on
 * col/ diagonal1/ diagnol2.
 * trick: to detect whether 2 positions sit on the same diagnol:
 * if delta(col, row) equals, same diagnol1;
 * if sum(col, row) equals, same diagnal2.
 */
private final Set<Integer> occupiedCols = new HashSet<Integer>();
private final Set<Integer> occupiedDiag1s = new HashSet<Integer>();
private final Set<Integer> occupiedDiag2s = new HashSet<Integer>();
public int totalNQueens(int n) {
    return totalNQueensHelper(0, 0, n);
}

private int totalNQueensHelper(int row, int count, int n) {
    for (int col = 0; col < n; col++) {
        if (occupiedCols.contains(col))
            continue;
        int diag1 = row - col;
        if (occupiedDiag1s.contains(diag1))
            continue;
        int diag2 = row + col;
        if (occupiedDiag2s.contains(diag2))
            continue;
        // we can now place a queen here
        if (row == n-1)
            count++;
        else {
            occupiedCols.add(col);
            occupiedDiag1s.add(diag1);
            occupiedDiag2s.add(diag2);
            count = totalNQueensHelper(row+1, count, n);
            // recover
            occupiedCols.remove(col);
            occupiedDiag1s.remove(diag1);
            occupiedDiag2s.remove(diag2);
        }
    }

    return count;
}
```

written by AlexTheGreat original link here

Solution 2

```
int totalNQueens(int n) {
    vector<bool> col(n, true);
    vector<bool> anti(2*n-1, true);
    vector<bool> main(2*n-1, true);
    vector<int> row(n, 0);
    int count = 0;
    dfs(0, row, col, main, anti, count);
    return count;
}
void dfs(int i, vector<int> &row, vector<bool> &col, vector<bool>& main, vector<b
ool> &anti, int &count) {
        if (i == row.size()) {
            count++;
            return;
        }
       for (int j = 0; j < col.size(); j++) {
         if (col[j] && main[i+j] && anti[i+col.size()-1-j]) {
             row[i] = j;
             col[j] = main[i+j] = anti[i+col.size()-1-j] = false;
             dfs(i+1, row, col, main, anti, count);
             col[j] = main[i+j] = anti[i+col.size()-1-j] = true;
      }
    }
}
```

written by lchen77 original link here

Solution 3

Following codes got AC. But you should never write some codes like this. This is post is just for joking.

```
/**
 * @param {number} n
 * @return {number}
 */
var totalNQueens = function(n) {
    var p = '', s = 0, l;
    for (var i = 0; i < n; i++) {
        l = '\nfor (var s# = 0; s# < ' + n + '; s#++)';
        for (var j = 0; j < i; j++)
            l += 'if (s# !== s@ && Math.abs(s# - s@) !== (# - @)) '.replace(/@/g, j);
        p += l.replace(/#/g, i);
    }
    p += '\ns++;\ns';
    return eval(p);
};
```

written by ts original link here