```
import java.lang.*;
import java.util.*;
class NQueens {
    private int[][] board;
    private int solutions;
    private List<List<String>> allSolutions;

    public List<List<String>> totalNQueens(int n) {
        board = new int[n][n];
        solutions = 0;
        allSolutions = new ArrayList<>();
        solveNQueens(0, n);
        return allSolutions;
    }

    private boolean isSafe(int row, int col, int n) {
        // Check this row on the left side
        for (int i = 0; i < col; i++) {
            if (board[row][i] == 1) {
                return false;
            }
        }

        // Check upper diagonal on the left side
        for (int i = row, j = col; i >= 0 && j >= 0; i--, j--) {
            if (board[i][j] == 1) {
                return false;
            }
        }

        // Check lower diagonal on the left side
        for (int i = row, j = col; i < n && j >= 0; i++, j--) {
            if (board[i][j] == 1) {
                return false;
            }
        }

        return true;
    }

    private void solveNQueens(int col, int n) {
        if (col == n) {
            solutions++;
            addSolution(n);
            return;
        }

        for (int row = 0; row < n; row++) {
            if (isSafe(row, col, n)) {
                board[row][col] = 1;
                solveNQueens(col + 1, n);
                board[row][col] = 0; // Backtrack
            }
        }
    }
```

```
     private void addSolution(int n) {
        List<String> solution = new ArrayList<>();
        for (int row = 0; row < n; row++) {
            StringBuilder rowString = new StringBuilder();
            for (int col = 0; col < n; col++) {
                if (board[row][col] == 1) {
                    rowString.append("Q");
                } else {
                    rowString.append(".");
                }
            }
            solution.add(rowString.toString());
        }
        allSolutions.add(solution);
    }

    public static void main(String[] args) {
        int n = 4; // Change this value to the desired board size
        NQueens solver = new NQueens();
        List<List<String>> solutions = solver.totalNQueens(n);
        System.out.println("Number of solutions for " + n + "-Queens problem: " +
        solutions.size());
        for (List<String> solution : solutions) {
            System.out.println("Solution:");
            for (String row : solution) {
                System.out.println(row);
            }
            System.out.println();
        }
    }
}
```

OUTPUT:

```
Number of solutions for 4-Queens problem: 2

Solution:
..Q.
Q...
...Q
.Q..

Solution:
.Q..
...Q
Q...
..Q.
```