```
 1: //This is  the Solution of the 8 Queens Problem
 2: // from Dr. Silvia Rothen
 3:
 4: this is the solution in groovy /grails
 5: there is always a java solution
 6:
 7:
 8: class Max8DamenProblem3
 9: Integer MIN = 0;
10: Integer MAX = 7;
11: Integer maxDiag = 2*MAX
12: String QUEEN = "Q";
13: String FORBIDDEN = "X";
14: String ls = System.getProperty("line.separator");
15:
16: Boolean[][] board = new Boolean[MAX+1][MAX+1];
17: Boolean[] safeRow = new Boolean[MAX+1];
18: //is the diagonal / from upper right to lower left unthreatened?
19: Boolean[] safeLeftDiag = new Boolean[maxDiag+1];
20: //is the diagonal from \ upper left to lower right unthreatened?
21: Boolean[] safeRightDiag = new Boolean[maxDiag+1];
22: Integer tester = 0;
23: StringBuffer resList = new StringBuffer();
24:
25:
26: Boolean safe(Integer row, Integer col) {
27: return safeRow[row] && safeLeftDiag[row + col] && safeRightDiag[row - col + 7]
28: }
29:
30: void chessPrint(Boolean[][] board)
31: String tempstr = "";
32: String codestr = "";
33: tester = tester + 1;
34: resList.append(tester + ls);
35:
36: (MIN..MAX).each {col ->
37: (MIN..MAX).each {row ->
38: if (board[row][col]) {
39: tempstr = tempstr + "Q"
40: codestr = codestr + (row + 1)
41: } else {
42: tempstr = tempstr + "X";
43: }
44: }
45: resList.append(tempstr + ls)
46: tempstr= "";
47: }
48: resList.append(codestr + ls)
49: resList.append("-"*15 + ls)
50: println (tester + ": " + codestr);
51: }
52:
53: void printBoard(Boolean[][] board) {
54: (MIN..MAX).each {row ->
55: (MIN..MAX).each {col ->
56: if (board[row][col]) {
57: print QUEEN
58: } else {
59: print FORBIDDEN
60: }
61: }
62: println ""
63: }
64: println "-"*(MAX+1)
65: }
66:
67: Boolean[][] initBoard(Boolean[][] board) {
68: (MIN..MAX).each {rowcounter ->
69: (MIN..MAX).each {colcounter ->
70: board[rowcounter][colcounter] = false;
71: }
72: return board
73: }
74: }
75:
```

```
 76: void tryColumn(Integer column, Boolean[][]board) {
 77: Integer row = MIN;
 78: while (row <= MAX) {
 79: if (safe(row, column)) {
 80: safeRow[row] = false;
 81: safeLeftDiag[row + column] = false;
 82: safeRightDiag[row - column + 7] = false;
 83: board[row][column] = true;
 84: if (column < MAX) {
 85: tryColumn(column + 1, board)
 86: } else {
 87: chessPrint(board)
 88: }
 89: //removes queens
 90: safeRow[row] = true;
 91: safeLeftDiag[row + column] = true;
 92: safeRightDiag[row - column + 7] = true;
 93: board[row][column] = false;
 94: }
 95: row++
 96: }
 97: }
 98:
 99: public static void main(String[] args) {
100: Max8DamenProblem3 dp = new Max8DamenProblem3()
101: dp.run()
102: }
103:
104: public void run() {
105: (MIN..MAX).each {item ->
106: safeRow[item] = true
107: }
108: (0..(maxDiag)).each{item ->
109: safeLeftDiag[item] = true
110: }
111:
112: (0..(maxDiag)).each{item ->
113: safeRightDiag[item] = true
114: }
115:
116: this.initBoard(board)
117: this.tryColumn(MIN, board);
118: File textfile = new File(
119: "F:\\_loesch\\maxbox_200910\\maxbox2\\examples\\chesssolution_result_silvi.txt")
120: textfile.write ""
121: textfile.append resList
122: }
123: }
124:
125: Quelle für Java-Code:
126: http://www.cs.pitt.edu/~kirk/cs1501/codehandouts/JRQueens.java
```