1. N queens

```
def solve_n_queens(n):
    def is_safe(board, row, col):
            if board[row][i] == 1:
                return False
        for i, j in zip(range(row, -1, -1), range(col, -1, -1)):
            if board[i][j] == 1:
                return False
        for i, j in zip(range(row, n, 1), range(col, -1, -1)):
            if board[i][j] == 1:
                return False

        return True

    def solve(board, col):
        if col >= n:
            solution = []
            for i in range(n):
                solution.append("".join('Q' if board[i][j] == 1 else '.' for j in range(n)))
            solutions.append(solution)
            return True

        res = False
        for i in range(n):
            if is_safe(board, i, col):
                board[i][col] = 1
                res = solve(board, col + 1) or res
                board[i][col] = 0

        return res

    solutions = []
    board = [[0 for _ in range(n)] for _ in range(n)]
    solve(board, 0)
    return solutions

solutions = solve_n_queens(8)
for solution in solutions[:8]:
    for line in solution:
        print("solution 1 ", line)
    print()
```