```python
import copy
N = 8    # Size of the chessboard (8x8)
def printSolution(board):
    for row in board:
        for i in range(N):
            print("Q" if row[i] else ".", end=" ")
        print()
    print()

def isSafe(board, row, col):
    for i in range(row):
        if board[i][col]:
            return False
    for i, j in zip(range(row - 1, -1, -1), range(col - 1, -1, -1)):
        if board[i][j]:
            return False
    for i, j in zip(range(row - 1, -1, -1), range(col + 1, N)):
        if board[i][j]:
            return False
    return True

def solve(board, row, solutions):
    if row == N:
        solutions.append(copy.deepcopy(board))
        printSolution(board)
        return
    for col in range(N):
        if isSafe(board, row, col):
            board[row][col] = 1
            solve(board, row + 1, solutions)
            board[row][col] = 0

def eightQueens():
    board = [[0 for _ in range(N)] for _ in range(N)]
    solutions = []
    solve(board, 0, solutions)
```

```
. . . Q . . . .
Q . . . . . . .
. . . . . . Q .
. . . . Q . . .
. . Q . . . . .
. . . . . Q . .

. . . . . . . Q
. Q . . . . . .
. . . Q . . . .
. Q . . . . . .
Q . . . . . . .
. . . . . Q . .
. . Q . . . . .
. . . . Q . . .

. . . . . . . Q
. . Q . . . . .
Q . . . . . . .
. . . . . Q . .
. Q . . . . . .
. . . . Q . . .
. . . . . . Q .
. . . Q . . . .

. . . . . . . Q
. . Q . . . . .
Q . . . . . . .
. . . . . Q . .
. Q . . . . . .
. . . . Q . . .
. Q . . . . . .
. . . . . Q . .

Total solutions found: 92
>>>
```