**PMAS Arid Agriculture University**

**University Institute of Information Technology**

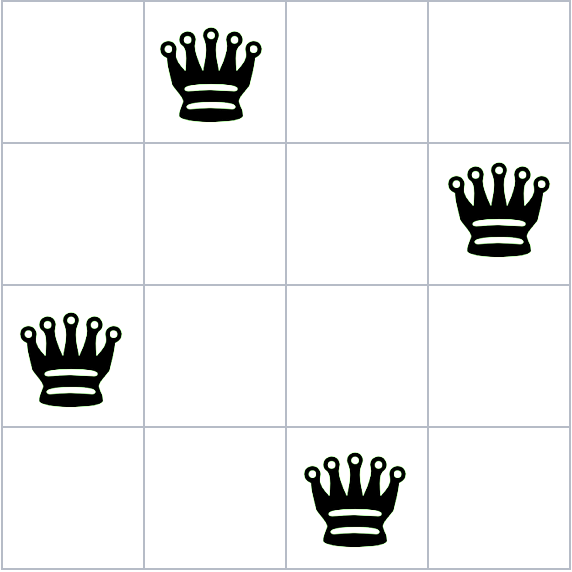
**CS-632 Artificial Intelligence**

**Handout 03**

# Eight Queens Problem

**Explanation:**

The eight queens’ problem is a problem in which we figure out a way to put 8 queens on an 8×8 chessboard in such a way that no queen should attack the other. For basic info about the queen in a chess game, you should know that a queen can move in any direction ( vertically, horizontally, and diagonally) and to any number of places. In the figure below you can see how to place 4 queens on a 4×4 chessboard.



Similarly, we have to place 8 queens on an 8×8 chessboard. We will use backtracking to solve this interesting problem(puzzle).

## **Backtracking**

Backtracking the solution of the problem depends on the previous steps taken. We take a step and then analyze it that whether it will give the correct answer or not? and if not, then we move back and change the previous step.

## **Pseudocode**

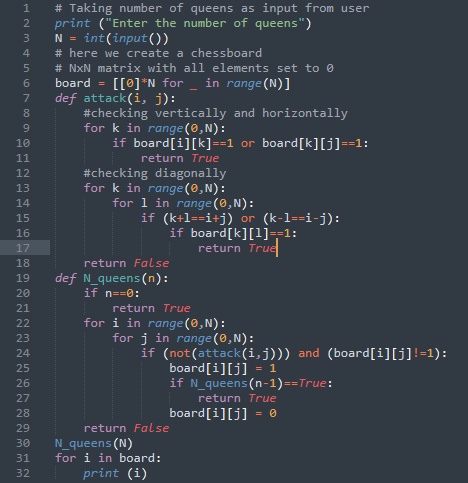
START

1. begin from the leftmost column
2. if all the queens are placed, return true/ print configuration
3. check for all rows in the current column
   1. if queen placed safely, mark row and column; and recursively check if we approach in the current configuration, do we obtain a solution or not
   2. if placing yields a solution, return true
   3. if placing does not yield a solution, unmark and try other rows
4. if all rows tried and solution not obtained, return false and backtrack

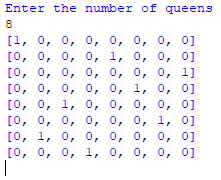
END

# **Implementation**

Implementing eight queens’ problem in python,



## **Output**



**Code:**

# Taking number of queens as input from user

print ("Enter the number of queens")

N = int(input())

# here we create a chessboard

# NxN matrix with all elements set to 0

board = [[0]\*N for \_ in range(N)]

def attack(i, j):

#checking vertically and horizontally

for k in range(0,N):

if board[i][k]==1 or board[k][j]==1:

return True

#checking diagonally

for k in range(0,N):

for l in range(0,N):

if (k+l==i+j) or (k-l==i-j):

if board[k][l]==1:

return True

return False

def N\_queens(n):

if n==0:

return True

for i in range(0,N):

for j in range(0,N):

if (not(attack(i,j))) and (board[i][j]!=1):

board[i][j] = 1

if N\_queens(n-1)==True:

return True

board[i][j] = 0

return False

N\_queens(N)

for i in board:

print (i)