

## Title: Implementation of toy problems

EX. NO : 01

Name: Parth Langalia

DATE : 23-01-2023

Reg No. : RA2011033010033

### AIM :

To implement the N-queens problem by backtracking.

### PSEUDO CODE :

```
Place (k, i)
{
    For j  $\leftarrow$  1 to k - 1    do if (x
[j] = i)      or (Abs x [j]) - i) =
(Abs (j - k))
    then return false;
return true;
}

N - Queens (k, n)
{
    For i  $\leftarrow$  1 to n
do if Place (k, i) then
    {      x [k]  $\leftarrow$  i;
if
(k == n) then      write
(x [1....n]);
else
    N - Queens (k + 1, n);
    }
}
```

## **PROGRAM :**

```
: # 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

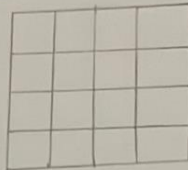
def N_queens(n):
    if n==0:
        return True
    for i in range(0,N):
        for j in range(0,N):
            if (not(attack(i,j))) and (board[i][j]!=1):
                board[i][j] = 1
                if N_queens(n-1)==True:
                    return True
                board[i][j] = 0

    return False

N_queens(N)
for i in board:
    print (i)
```

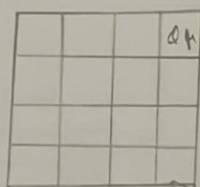
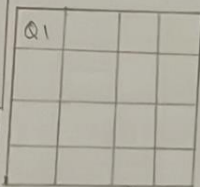
## Manual Calculations :

### Manual calculation

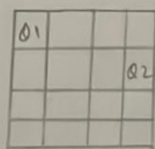
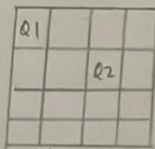


In the first row, you can place the first queen in any one of the 4 spots.

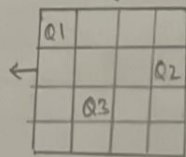
Let's place them in spot 1 and 4.



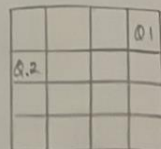
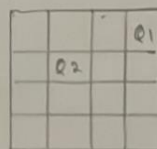
only 2 possibilities to place Q2



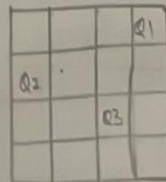
no further possibilities as it breaks the rules of the problem.



only 2 possibilities to place Q2



no further possibilities.



Positions 1 and 4 do not produce a successful result so we backtrack and try position 2 and 3.

Manual Calculations:

	Q1		

		Q1	

only 1 possibility  
of placing Q2

	Q1		
			Q2

		Q1	
Q2			

only 1 possibility  
of placing Q3

	Q1		
			Q2
Q3			

	Q1	Q2	
Q2			
			Q3

1 solution possible

1 solution possible

	Q1		
			Q2
Q3			
		Q4	

		Q1	
Q2			
			Q3
	Q4		

Total solutions : 2

## **OUTPUT :**

```
        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)

Enter the number of queens
4
[0, 1, 0, 0]
[0, 0, 0, 1]
[1, 0, 0, 0]
[0, 0, 1, 0]
```

## **RESULT :**

The N-queens problem has been successfully executed.