4. Solve 8-Queens Problem with suitable assumptions

```
#Number of queens
print ("Enter the number of queens")
N = int(input())

#chessboard
#NxN matrix with all elements 0
board = [[0]*N for _ in range(N)]

def is_attack(i, j):
    #checking if there is a queen in row or column
    for k in range(0,N):
        if board[i][k]==1 or board[k][j]==1:
            return True
    #checking diagonals
    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_queen(n):
    #if n is 0, solution found
    if n==0:
        return True
    for i in range(0,N):
        for j in range(0,N):
            '''checking if we can place a queen here or not
            queen will not be placed if the place is being attacked
            or already occupied'''
```

```
            if (not(is_attack(i,j))) and (board[i][j]!=1):
                board[i][j] = 1
                #recursion
                #wether we can put the next queen with this arrangment or not
                if N_queen(n-1)==True:
                    return True
                board[i][j] = 0

    return False

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

Output:

```
Enter the number of queens
8
```

Output:

```
[1, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 1, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 1]
[0, 0, 0, 0, 0, 1, 0, 0]
[0, 0, 1, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 1, 0]
[0, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 1, 0, 0, 0, 0]
```