TASK :8

```
# Python3 program to solve N Queen Problem using backtracking


N = 4  # Size of the chessboard


def printSolution(board):
   for i in range(N):
      for j in range(N):
         if board[i][j] == 1:
            print("Q", end=" ")
         else:
            print(".", end=" ")
      print()


def isSafe(board, row, col):
   # Check this row on left side
   for i in range(col):
      if board[row][i] == 1:
         return False


   # Check upper diagonal on left side
   for i, j in zip(range(row, -1, -1), range(col, -1, -1)):
      if board[i][j] == 1:
         return False


   # Check lower diagonal on left side
   for i, j in zip(range(row, N, 1), range(col, -1, -1)):
      if board[i][j] == 1:
         return False


   return True
```

```
def solveNQUtil(board, col):
    # Base case: If all queens are placed
    if col >= N:
        return True

    # Consider this column and try placing
    # this queen in all rows one by one
    for i in range(N):
        if isSafe(board, i, col):
            board[i][col] = 1  # Place this queen

            if solveNQUtil(board, col + 1):
                return True

            board[i][col] = 0  # Backtrack

    return False

def solveNQ():
    board = [[0 for _ in range(N)] for _ in range(N)]

    if not solveNQUtil(board, 0):
        print("Solution does not exist")
        return False

    printSolution(board)
    return True

# Driver Code
if _name_ == '_main_':
```

solveNQ()

OUTPUT: