EXPERIMENT 2:

1. AIM: To write the python program to solve 8-Queen problem

### Algorithm:

1. **Input Number of Queens**:
   * Prompt the user to input the number of queens (N).
2. **Create Chessboard**:
   * Create an NxN chessboard represented as a 2D matrix initialized with all elements set to 0.
3. **Define Attack Function** (**attack**):
   * Define a function to check if a queen placed at a certain position attacks any other queen.
   * Check for attacks horizontally, vertically, and diagonally.
4. **Define N Queens Function** (**N\_queens**):
   * Recursively attempt to place N queens on the board.
   * If N equals 0 (no queens remaining to place), return True.
   * Iterate through each position on the board:
     + If the position is not under attack and not already occupied by a queen:
       - Place a queen at the position.
       - Recur for N-1 queens.
       - If successful, return True.
       - If unsuccessful, backtrack by removing the queen from the current position.
   * If no solution is found, return False.
5. **Solve N Queens Problem**:
   * Call the **N\_queens** function with the input value of N.
   * If a solution is found, print the board with queens placed.
   * If no solution is found, print that no solution exists.

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)

INPUT:

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]