|  |  |
| --- | --- |
| **EX.NO : 01** | **8-QUEENS PROBLEM** |
| **DATE : 28.02.2024** |

**PROBLEM STATEMENT:**

You are given 8X8 chess board find a way to place 8 queens such that no queens can attack any other queens. A queen can only be attacked only if it lies on the same row or column or the same diagonal of any other queen all the possible configuration use backtracking algorithm to solve this problem.

**AIM:**

To backtracking to find all possible configurations of placing 8 queens on an 8x8 chess board without any queen attacking each other.

**ALGORITHM:**

**Step 1: Create an 8x8 board**: Initialize a 2D array to represent the chess board.

**Step 2: Start with the first row**: Begin placing queens from the first row.

**Step 3: Try each column**: For each column in the current row, try placing a queen.

**Step 4: Check for attacks**: Check if the queen can be attacked by any other queen already placed.

**Step 5: Place the queen**: If the queen is safe, place it on the board and move to the next row.

**Step 6: Recursively call the function**: Call the function again to place the next queen.

**Step 7: Backtrack if necessary**: If a queen cannot be placed without being attacked, backtrack to the previous row and try the next column.

**Step 8: Repeat steps 3-7**: Continue placing queens and backtracking until all 8 queens are placed or all possibilities are exhausted.

**Step 9: Print the solution**: If a solution is found, print the board configuration with the 8 queens placed.

**Step 10: Return if no solution**: If no solution is found, return an empty board or indicate that no solution exists.

**PROGRAM:**

N= int(input("Enter the number of queens :")) board = [[0]\*N for \_ in range(N)]

def attack(I,j):

for k in range(0,N):

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

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

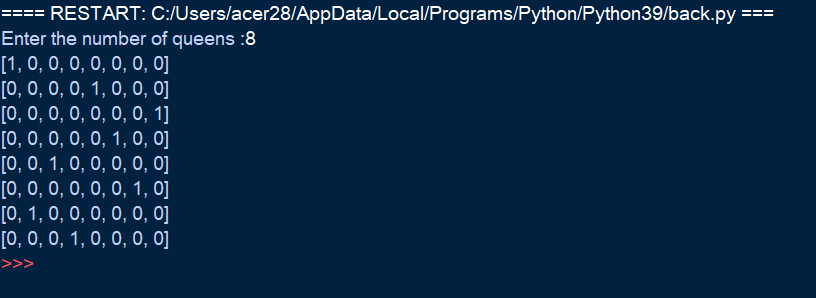
if N\_queens(n-1)==True: return True

board[i][j]=0 return False

N\_queens(N) for I in board:

print(i)

**OUTPUT:**



**RESULT:**

Thus the backtracking to find all possible configurations of placing 8 queens on an 8x8 chess board using python executed successfully.