**Experiment 1: 8 Queens Problem**

**Aim**

To write a program to solve the 8 Queens problem using backtracking.

**Procedure**

1. Place queens one by one in different columns, starting from the leftmost column.  
2. When placing a queen in a column, check for row and diagonal conflicts.  
3. If a conflict occurs, backtrack and try placing the queen in the next row.  
4. Repeat the process until all queens are placed without conflict.

**Code**

def print\_solution(board):  
 for row in board:  
 print(" ".join("Q" if col else "." for col in row))  
  
def is\_safe(board, row, col, n):  
 for i in range(col):  
 if board[row][i]:  
 return False  
 for i, j in zip(range(row, -1, -1), range(col, -1, -1)):  
 if board[i][j]:  
 return False  
 for i, j in zip(range(row, n, 1), range(col, -1, -1)):  
 if board[i][j]:  
 return False  
 return True  
  
def solve\_nqueens(board, col, n):  
 if col >= n:  
 return True  
 for i in range(n):  
 if is\_safe(board, i, col, n):  
 board[i][col] = 1  
 if solve\_nqueens(board, col + 1, n):  
 return True  
 board[i][col] = 0  
 return False  
  
def solve():  
 n = 8  
 board = [[0 for \_ in range(n)] for \_ in range(n)]  
 if solve\_nqueens(board, 0, n):  
 print\_solution(board)  
 else:  
 print("Solution does not exist")  
  
solve()

**Output**

