Experiment 2: 8 QUEEN PROGRAM

Aim: Implement an Algorithm in Python for solving the 8-QUEEN Problem.

```
def is_safe(board, row, col):
    # Check if there is a queen in the same row
    for i in range(col):
        if board[row][i] == 1:
            return False

    # Check upper diagonal on the 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 the left side
    for i, j in zip(range(row, 8, 1), range(col, -1, -1)):
        if board[i][j] == 1:
            return False

    return True


def solve_queens(board, col):
    # Base case: If all queens are placed, return True
    if col >= 8:
        return True

    # Consider this column and try placing this queen in all rows
    for i in range(8):
        if is_safe(board, i, col):
            # Place this queen in board[i][col]
            board[i][col] = 1

            # Recur to place rest of the queens
            if solve_queens(board, col + 1):
                return True

            # If placing queen in board[i][col] doesn't lead to a solution, then remove queen from
board[i][col]
```

```
            board[i][col] = 0

    # If the queen cannot be placed in any row in this column, then return False
    return False


def print_solution(board):
    for i in range(8):
        for j in range(8):
            print(board[i][j], end=" ")
        print()


def solve_8_queens():
    board = [[0] * 8 for _ in range(8)]

    if not solve_queens(board, 0):
        print("No solution exists")
        return

    print("Solution for 8-Queens problem:")
    print_solution(board)


# Call the function to solve 8-Queens problem
solve_8_queens()
```

Output:

solution for 8-Queens problem:

1 0 0 0 0 0 0 0

0 0 0 0 0 0 1 0

0 0 0 0 1 0 0 0

0 0 0 0 0 0 0 1

0 1 0 0 0 0 0 0

0 0 0 1 0 0 0 0

0 0 0 0 0 1 0 0

0 0 1 0 0 0 0 0

Result:

Code has been Implemented successfully.