In [1]:
```
#Problem Statement: Implement N-Queens Problem as Constraints Satisfaction
#Problem.
#Roll No-3310
```

In [3]:
```
def print_board(board):
    for row in board:
        print(" ".join(row))

def is_safe(board, row, col):
    for i in range(col):
        if board[row][i] == "Q":
            return False

    for i, j in zip (range(row, -1, -1), range(col, -1, -1)):
        if board[i][j] == "Q":
            return False

    for i, j in zip(range(row, len(board), 1), range(col, -1, -1)):
        if board[i][j] == "Q":
            return False

    return True

def solve(board, col):
    if col >= len(board):
        return True
    for i in range(len(board)):
        if is_safe(board, i, col):
            board[i][col] = "Q"
            if solve(board, col+1):
                return True
            board[i][col] = "."
    return False

n = int(input("Enter the number of Queens : "))
print()
board = [[ "." for i in range (n)] for j in range(n)]

if solve(board, 0):
    print_board(board)

else:
    print("Solution not found")
```

```
Enter the number of Queens : 4

. . Q .
Q . . .
. . . Q
. Q . .
```