Name: Vaishnavi Pravin Kolse
Class: BE - A
Roll No.37

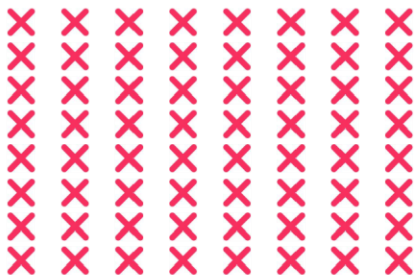
Practical No.5

Aim: Design n-Queens matrix having first Queen placed. Use backtracking to place Queens to generate the final n-queen ‘s matrix

In [1]:
```
class Queen:
    def __init__(self, N):
        self.N = N
        self.board = [[0] * N for _ in range(N)]  # Initialize N x N chessboar

    def disp_board(self):
        # Display the chessboard with queens and empty spaces
        for row in self.board:
            print()
            for col in row:
                if col == 1:
                    print(u"\U0001F451", end=' ')  # Queen emoji
                else:
                    print(u"\u274C", end=' ')  # Cross mark emoji
        print(end='\n')

    def is_attack(self, i, j):
        # Check if a queen can be placed at (i, j)
        # Check row and column
        for k in range(self.N):
            if self.board[i][k] == 1 or self.board[k][j] == 1:
                return True

        # Check diagonals
        for k in range(self.N):
            for l in range(self.N):
                if (k + l == i + j) or (k - l == i - j):
                    if self.board[k][l] == 1:
                        return True

        return False

    def N_queen(self, n):
        # Try to place N queens on the board
        if n == 0:
            return True  # Base case: all queens have been placed

        for i in range(self.N):
            for j in range(self.N):
                if not self.is_attack(i, j) and self.board[i][j] != 1:
                    # Place queen at (i, j)
                    self.board[i][j] = 1
                    if self.N_queen(n - 1):
                        return True  # Recursively place remaining queens
                    # Backtrack if placing queen at (i, j) doesn't lead to a s
                    self.board[i][j] = 0

        return False


# Input number of queens
N = int(input("Enter the number of queens: "))
Q = Queen(N)

print('Initial State:')
Q.disp_board()
```

```python
# Place N queens using backtracking
if Q.N_queen(N):
    print('\nFinal State:')
    Q.disp_board()
else:
    print("No solution found.")
```

```
Enter the number of queens: 8
Initial State:
```



```
Final State:
```

In [ ]: