N Queens

Aim:

To solve the N-Queens problem using the backtracking algorithm in python, where the goal is to place N queens on an N*N, Queens should not threaten each other.

Code:

```
def is_safe(board, row, col, N):
    for i in range(col):
        if board[row][i] == 1:
            return false

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

    for i, j in zip(range(row, N), range(col, -1, -1)):
        if board[i][j] == 1:
            return false.

    return true.

def solve_n_queen_util(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_n_queens_util(board, col+1, N):
                return True.

            board[i][col] = 0.
    return False.
```

```
def solve_n_Queens(N):
    board = [[0 for _ in range(N)] for _ in range(N)]
if not solve_n_Queens_util(board, 0, N):
    Print ("no solution exists")
    return
for row in board:
    print(row)
```

output:

4×4

[0, 0, 1, 0]

[1, 0, 0, 0]

[0, 0, 0, 1]

[0, 1, 0, 0]

Algorithm:

* Start
* place queen row by row.
* check for conflicts
  -> No other Queen is in the same column
  -> No other Queen is on the same diagonal
* Back track if need.
* Repeat until a solution is found
* Once all queens are placed in valid positions, print or return the solution
* Stop.