.vscode\practical5.java

```java
public class NQueens {
    private int[][] board;
    private int n;

    // Constructor to initialize the board and n
    public NQueens(int n) {
        this.n = n;
        this.board = new int[n][n];
    }

    // Function to print the board
    private void printBoard() {
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                System.out.print(board[i][j] + " ");
            }
            System.out.println();
        }
        System.out.println();
    }

    // Function to check if it's safe to place a queen at board[row][col]
    private boolean isSafe(int row, int col) {
        // Check the column
        for (int i = 0; i < row; i++) {
            if (board[i][col] == 1) {
                return false;
            }
        }

        // Check the upper left diagonal
        for (int i = row, j = col; i >= 0 && j >= 0; i--, j--) {
            if (board[i][j] == 1) {
                return false;
            }
        }

        // Check the upper right diagonal
        for (int i = row, j = col; i >= 0 && j < n; i--, j++) {
            if (board[i][j] == 1) {
                return false;
            }
        }

        return true; // Safe to place the queen
    }

    // Backtracking function to solve the N-Queens problem
```

```java
    private boolean solveNQueens(int row) {
        if (row == n) { // All queens are placed
            return true;
        }

        for (int col = 0; col < n; col++) {
            if (isSafe(row, col)) {
                board[row][col] = 1; // Place the queen

                if (solveNQueens(row + 1)) {
                    return true; // Continue to place the rest
                }

                // If placing queen in the current position doesn't lead to a solution
                board[row][col] = 0; // Backtrack
            }
        }
        return false; // No solution found
    }

    // Function to solve the N-Queens problem with the first queen placed
    public void placeFirstQueen(int row, int col) {
        board[row][col] = 1; // Place the first queen
        if (solveNQueens(0)) {
            printBoard(); // Print the board
        } else {
            System.out.println("No solution found.");
        }
    }

    public static void main(String[] args) {
        int n = 8; // Size of the chessboard
        NQueens nQueens = new NQueens(n);

        // Place the first queen at (0, 0) (the first row and first column)
        nQueens.placeFirstQueen(0, 0);
    }
}


/*  Output-
1 0 0 0 0 0 0 0
0 0 0 1 0 0 0 0
0 0 0 0 0 0 1 0
0 0 0 0 0 0 0 1
0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0
0 0 0 0 0 1 0 0
*/
```