Roll No: 20BCE204
Course: 2CSDE93 - Blockchain Technology
Practical No: 10
Aim: Tick-Tack-Toe in Solidity Programming Language

Code:

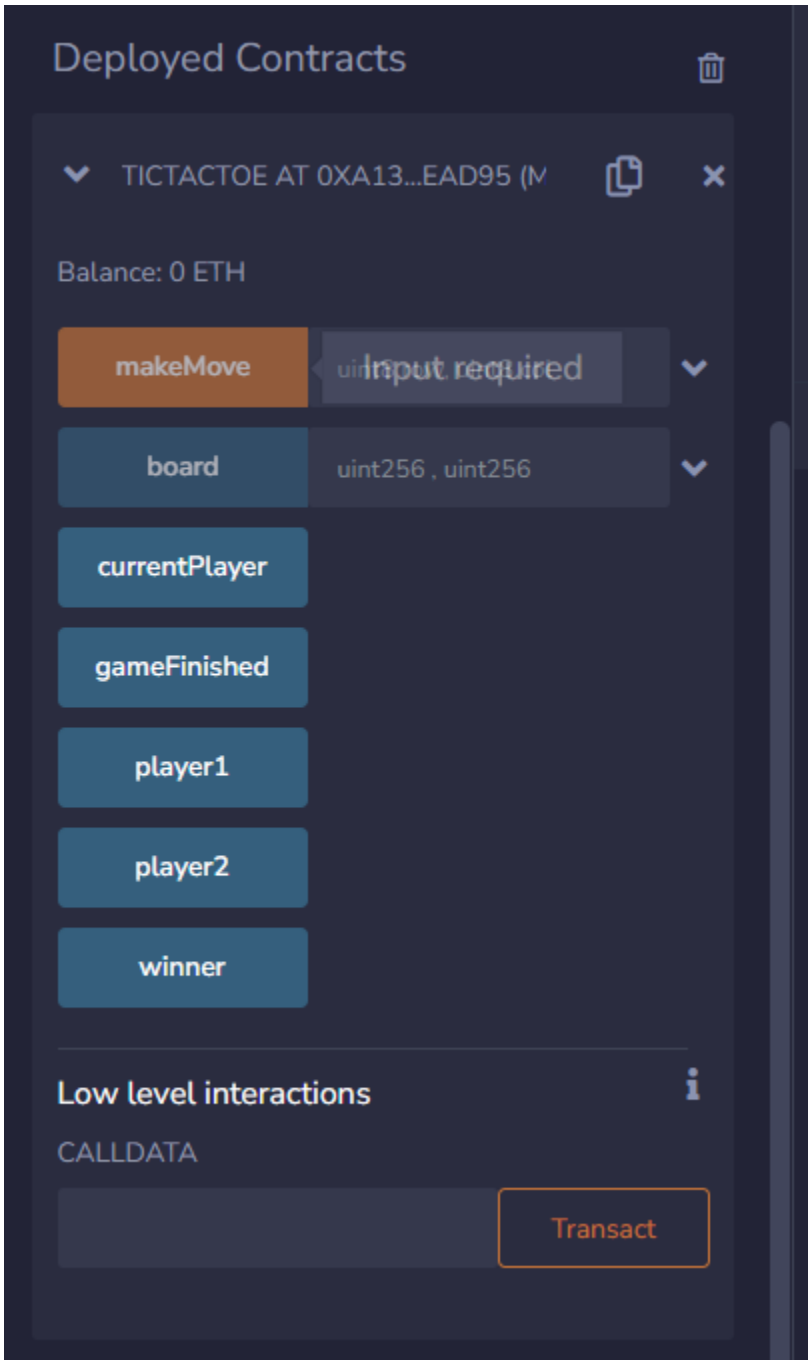
```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract TicTacToe {
    address public player1;
    address public player2;
    address public currentPlayer;
    uint8[3][3] public board;
    address public winner;
    bool public gameFinished = false;

    event MoveMade(address indexed player, uint8 row,
uint8 col);
    event GameFinished(address indexed winner);

    constructor(address _player1, address _player2) {
        player1 = _player1;
        player2 = _player2;
        currentPlayer = player1;
    }

    modifier onlyPlayers() {
        require(
```

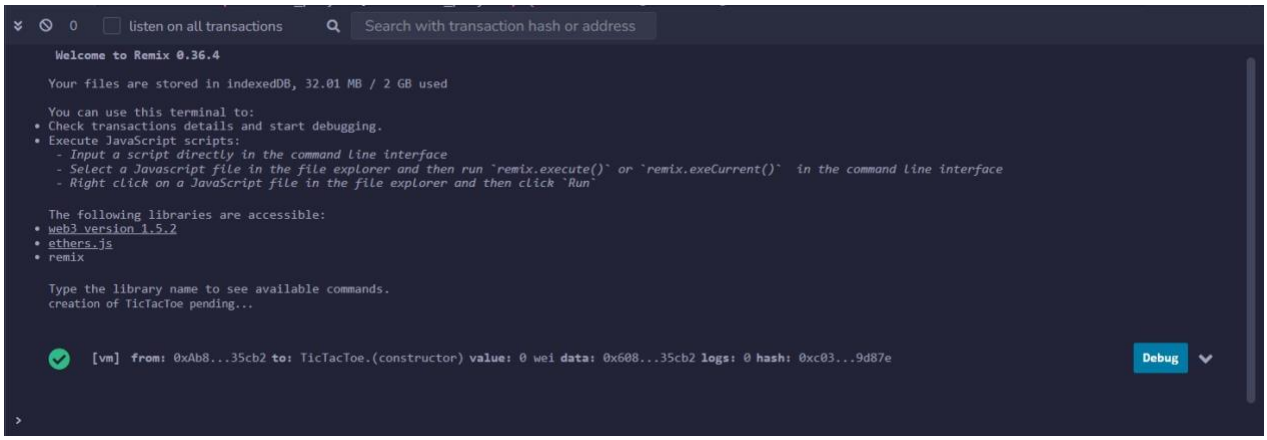
```
            msg.sender == player1 || msg.sender ==
player2,
            "Only players can make a move"
        );
        _;
    }

    modifier isGameFinished() {
        require(!gameFinished, "The game has already
finished");
        _;
    }

    function makeMove(uint8 row, uint8 col) public
onlyPlayers {
        require(
            row >= 1 && row <= 3,
            "Invalid row. Row must be between 1 and 3"
        );
        require(
            col >= 1 && col <= 3,
            "Invalid column. Column must be between 1
and 3"
        );
        require(
            board[row - 1][col - 1] == 0,
            "Invalid move. Cell already taken"
        );
```

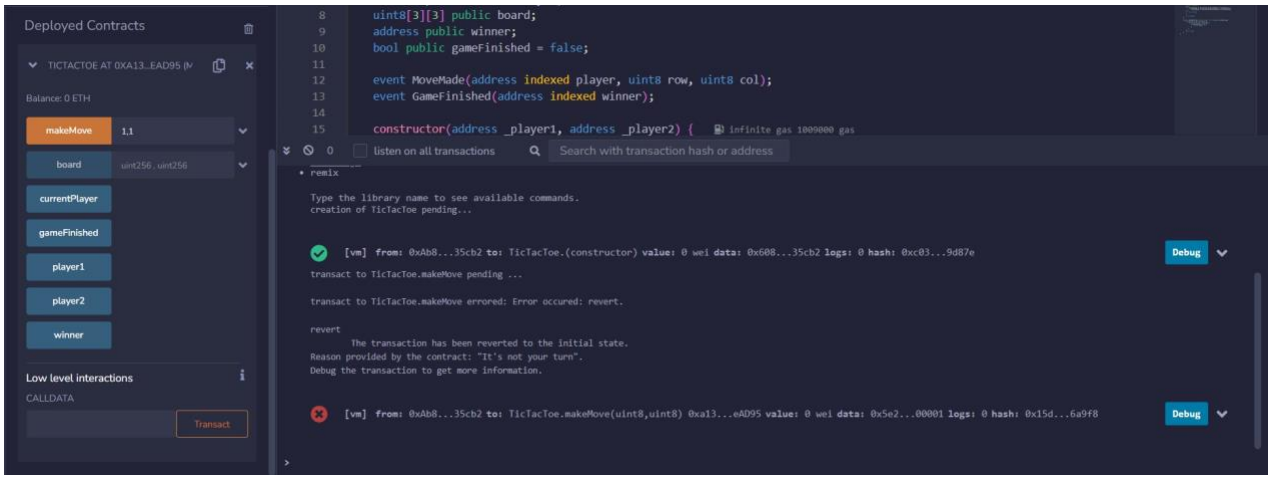
```
        require(msg.sender == currentPlayer, "It's not
your turn");

        board[row - 1][col - 1] = currentPlayer ==
player1 ? 1 : 2;
        emit MoveMade(currentPlayer, row, col);

        if (checkWinner(row, col)) {
            winner = currentPlayer;
            gameFinished = true;
            emit GameFinished(winner);
        } else if (boardIsFull()) {
            gameFinished = true;
            emit GameFinished(address(0)); // It's a
draw
        } else {
            currentPlayer = (currentPlayer == player1)
? player2 : player1;
        }
    }

    function checkWinner(uint8 row, uint8 col) internal
view returns (bool) {
        // Check the row
        if (
            board[row - 1][0] == board[row - 1][1] &&
            board[row - 1][0] == board[row - 1][2]
        ) {
            return true;
        }
```

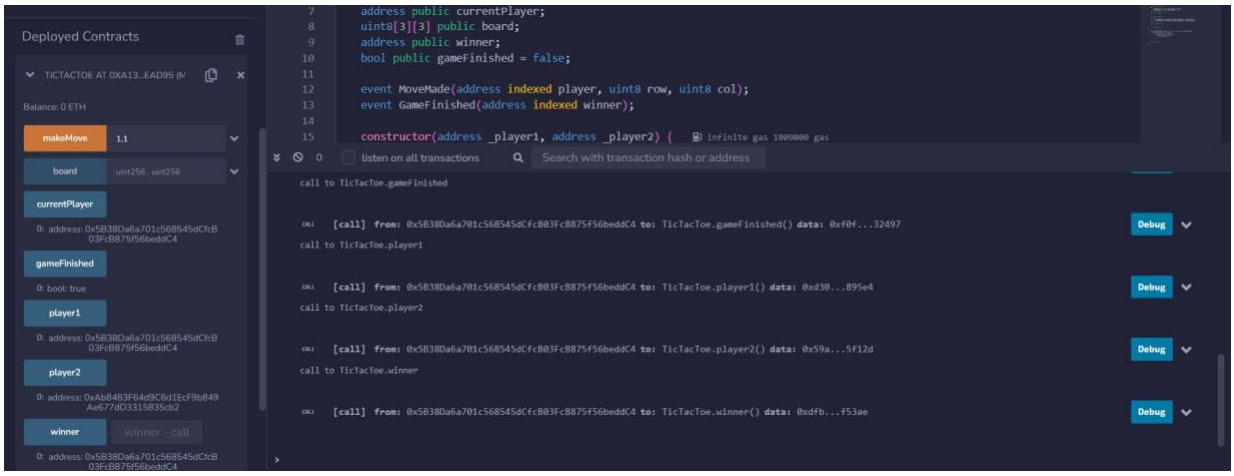
```
        // Check the column
        if (
            board[0][col - 1] == board[1][col - 1] &&
            board[0][col - 1] == board[2][col - 1]
        ) {
            return true;
        }

        // Check diagonals
        if (
            ((board[0][0] == board[1][1] && board[0][0]
== board[2][2]) ||
                (board[0][2] == board[1][1] &&
board[0][2] == board[2][0]))
        ) {
            return true;
        }

        return false;
    }

    function boardIsFull() internal view returns (bool)
{
        for (uint8 i = 0; i < 3; i++) {
            for (uint8 j = 0; j < 3; j++) {
                if (board[i][j] == 0) {
                    return false;
                }
            }
```

```
            }
            return true;
        }
}
```


Deployed Contracts
TICTACTOE AT 0XA13...EAD95 (M
Balance: 0 ETH
makeMove
Input required
board
uint256 , uint256
currentPlayer
gameFinished
player1
player2
winner
Low level interactions
CALLDATA
Transact


0
listen on all transactions
Search with transaction hash or address
Welcome to Remix 0.36.4
Your files are stored in indexedDB, 32.01 MB / 2 GB used
You can use this terminal to:
• Check transactions details and start debugging.
• Execute JavaScript scripts:
- Input a script directly in the command line interface
- Select a Javascript file in the file explorer and then run `remix.execute()` or `remix.exeCurrent()` in the command line interface
- Right click on a JavaScript file in the file explorer and then click `Run`
The following libraries are accessible:
• web3 version 1.5.2
• ethers.js
• remix
Type the library name to see available commands.
creation of TicTacToe pending...
[vm] from: 0xAb8...35cb2 to: TicTacToe.(constructor) value: 0 wei data: 0x608...35cb2 logs: 0 hash: 0xc03...9d87e
Debug


Deployed Contracts
TICTACTOE AT 0XA13...EAD95 (M
Balance: 0 ETH
makeMove
1,1
board
uint256, uint256
currentPlayer
gameFinished
player1
player2
winner
Low level interactions
CALLDATA
Transact
uint8[3][3] public board;
address public winner;
bool public gameFinished = false;
event MoveMade(address indexed player, uint8 row, uint8 col);
event GameFinished(address indexed winner);
constructor(address _player1, address _player2) {
infinite gas 1000000 gas
listen on all transactions
Search with transaction hash or address
• remix
Type the library name to see available commands.
creation of TicTacToe pending...
[vm] from: 0xAb8...35cb2 to: TicTacToe.(constructor) value: 0 wei data: 0x608...35cb2 logs: 0 hash: 0xc03...9d87e
Debug
transact to TicTacToe.makeMove pending ...
transact to TicTacToe.makeMove errored: Error occured: revert.
revert
The transaction has been reverted to the initial state.
Reason provided by the contract: "It's not your turn".
Debug the transaction to get more information.
[vm] from: 0xAb8...35cb2 to: TicTacToe.makeMove(uint8,uint8) 0xa13...eAD95 value: 0 wei data: 0x5e2...00001 logs: 0 hash: 0x15d...6a9f8
Debug


Deployed Contracts
TICTACTOE AT 0XA13...EAD95 (M
Balance: 0 ETH
makeMove
1,1
board
uint256, uint256
currentPlayer
0: address: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
gameFinished
0: bool: true
player1
0: address: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
player2
0: address: 0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2
winner
winner - call
0: address: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
address public currentPlayer;
uint8[3][3] public board;
address public winner;
bool public gameFinished = false;
event MoveMade(address indexed player, uint8 row, uint8 col);
event GameFinished(address indexed winner);
constructor(address _player1, address _player2) {
infinite gas 1000000 gas
listen on all transactions
Search with transaction hash or address
call to TicTacToe.gameFinished
call [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: TicTacToe.gameFinished() data: 0xf0f...32497
Debug
call to TicTacToe.player1
call [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: TicTacToe.player1() data: 0xd30...895e4
Debug
call to TicTacToe.player2
call [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: TicTacToe.player2() data: 0x59a...5f12d
Debug
call to TicTacToe.winner
call [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: TicTacToe.winner() data: 0xdfb...f53ae
Debug