Aayush Shah
19BCE245
22 October 2022

BlockChain Technology

Practical 10

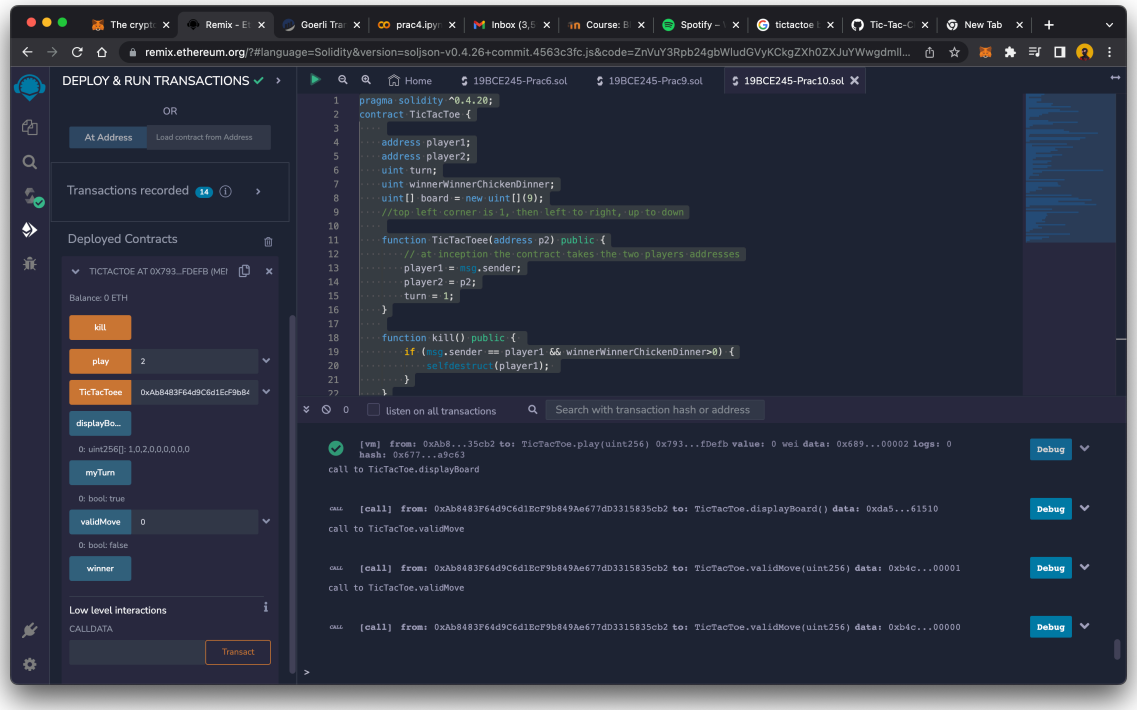
Implementing TicTacToe

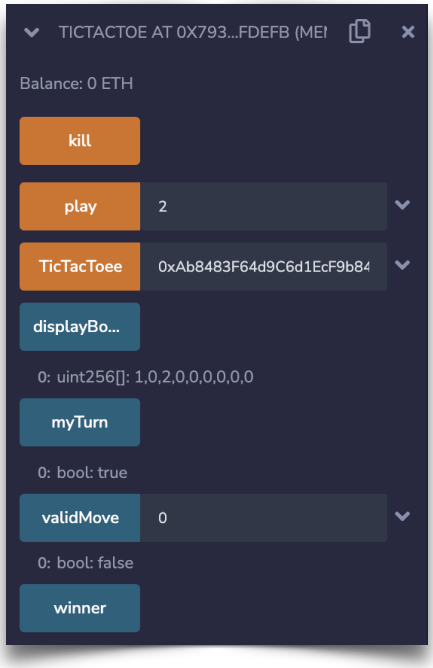
- **Code :**

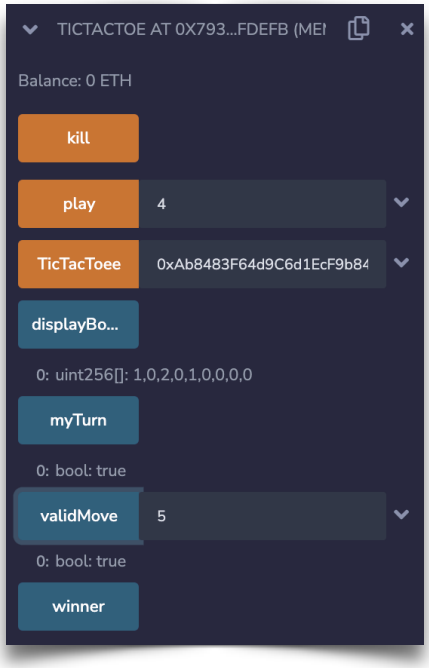
```
pragma solidity ^0.4.20;
contract TicTacToe {

    address player1;
    address player2;
    uint turn;
    uint winnerWinnerChickenDinner;
    uint[] board = new uint[](9);
    //top left corner is 1, then left to right, up to down

    function TicTacToee(address p2) public {
        // at inception the contract takes the two players
addresses
        player1 = msg.sender;
        player2 = p2;
        turn = 1;
    }

    function kill() public {
        if (msg.sender == player1 &&
winnerWinnerChickenDinner>0) {
            selfdestruct(player1);
        }
    }

    function myTurn() public view returns(bool) {
        //returns true if it's the sender's turn to play
        return (msg.sender == player1 && turn == 1) ||
(msg.sender == player2 && turn == 2) ;
    }
```

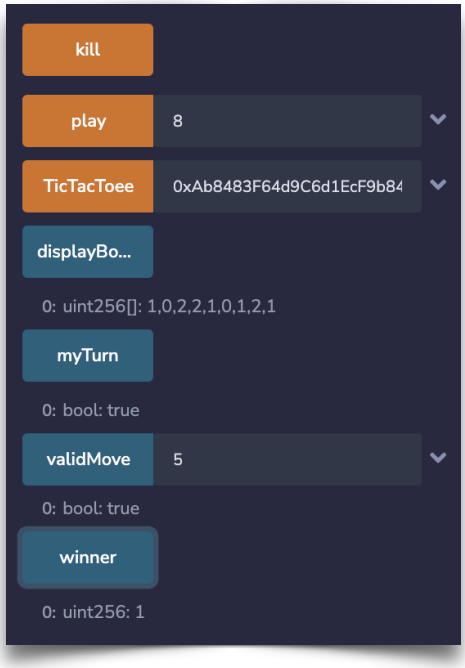
```
    function subCheckWin(uint a, uint b, uint c) private view
returns (bool){
        return board[a]>0 && board[a]==board[b] &&
board[b]==board[c];
    }

    function checkWin() private view returns (uint) {

        //function used to check if there is a winning player
        for (uint i = 0; i < 3; i++) {
            if (subCheckWin(i,3+i,6+i)) {
                return board[i];
            }
            if (subCheckWin(3*i,3*i+1,3*i+2)) {
                return board[3*i];
            }
        }
        if (subCheckWin(0,4,8)) {
            return board[0];
        }
        if (subCheckWin(2,4,6)) {
            return board[2];
        }
        if
(board[0]+board[1]+board[1]+board[1]+board[1]+board[1]+board[1
]+board[1]+board[1] == 13){
            return 3;
        }
        return 0;
    }

    function winner() public view returns (uint) {

        //getter for the winner attribute
        if (winnerWinnerChickenDinner>0) {
            return winnerWinnerChickenDinner;
        } else {
            return 0;
        }
    }

    function validMove(uint a) public view returns (bool) {

        //function to assess whether a required move is valid
or not
        //used both as an external callable function and
internally to validate moves
        //before writing them to the board
```

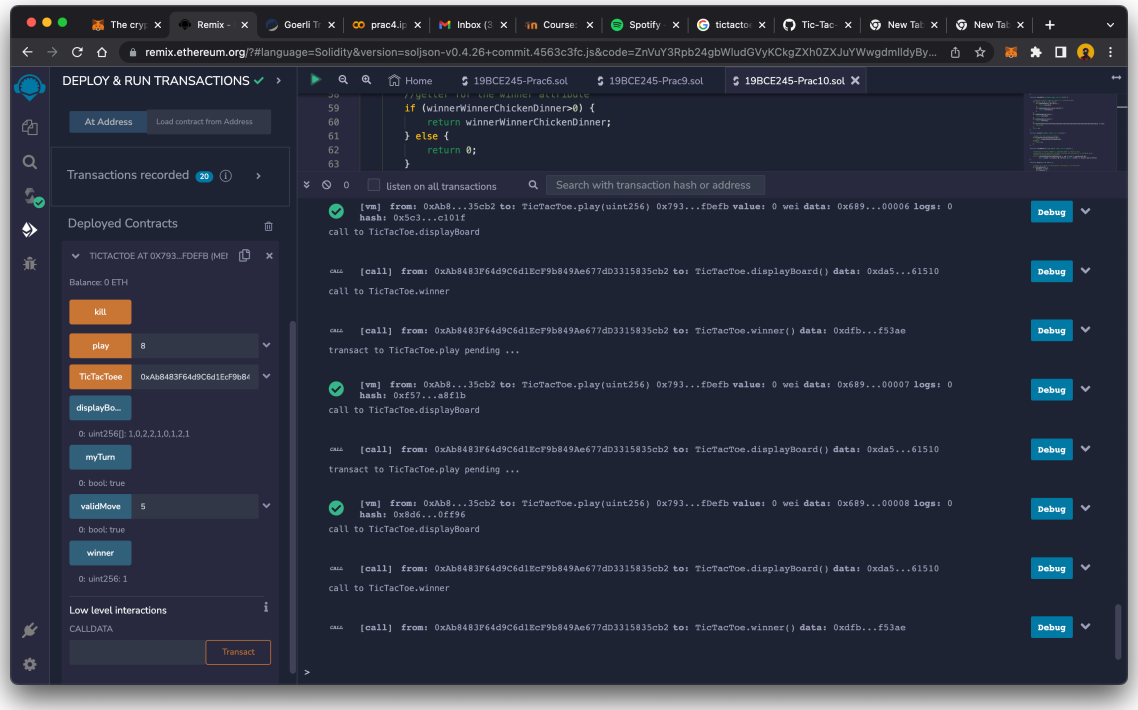
```
        return !(winnerWinnerChickenDinner>0 || a<0 || a>8 ||
board[a]>0) &&
                ((msg.sender == player1 && turn==1) ||
(msg.sender == player2 && turn==2));
    }

    function play(uint a) public {

        //checks if a move is valid before inserting it to the
board
        if (validMove(a)) {
            board[a] = turn;
            if (turn==1) {
                turn = 2;
            } else {
                turn = 1;
            }
            winnerWinnerChickenDinner = checkWin();
        } else {
            revert();
        }
    }

    function displayBoard() public view returns(uint[]) {

        //getter for the board
        return board;
    }
}
```











Conclusion

In this practical, we implemented tic tac toe game through smart contract.

.