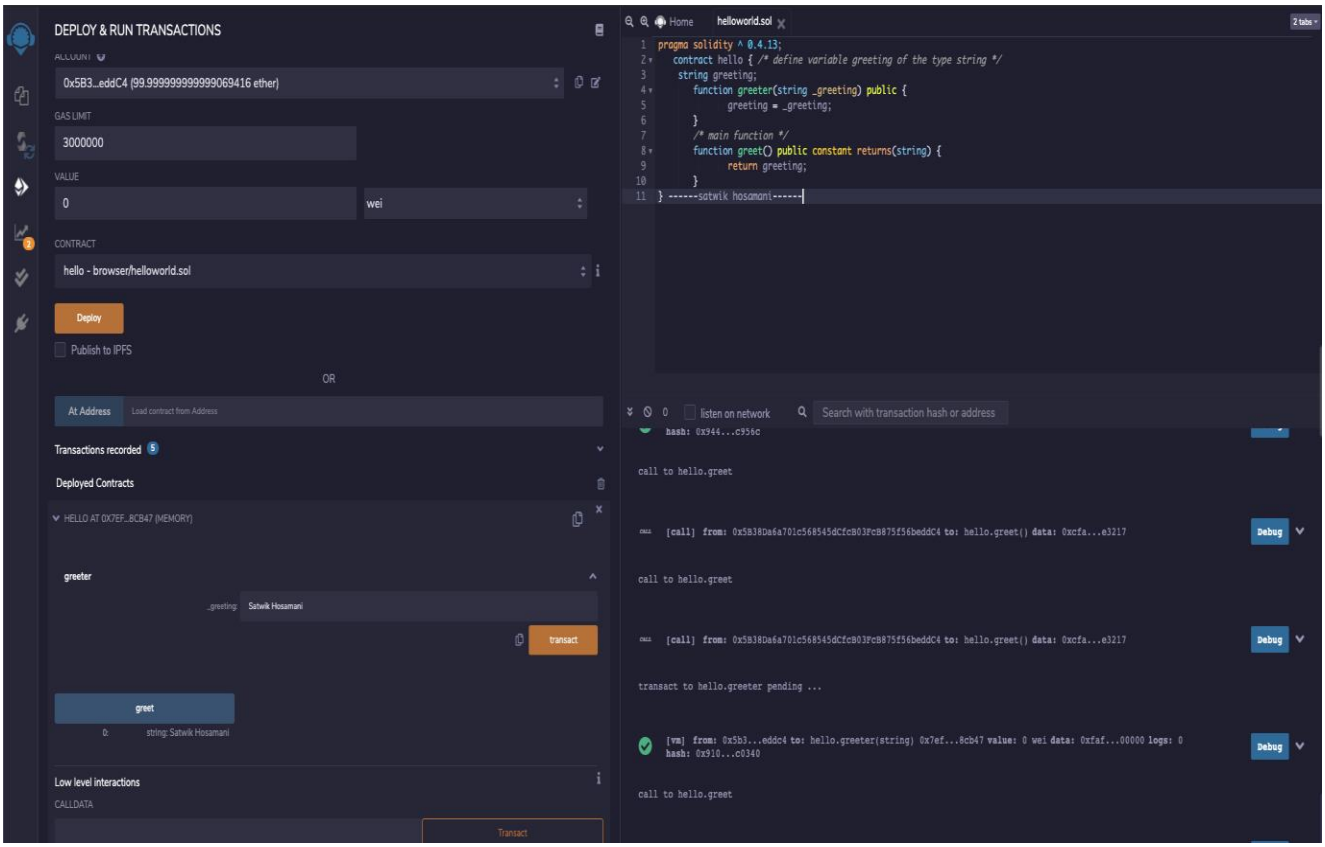
Blockchain Lab

Lab L2: Smart Contract Programming

Exercise 1a: Hello-world Contract with Remix

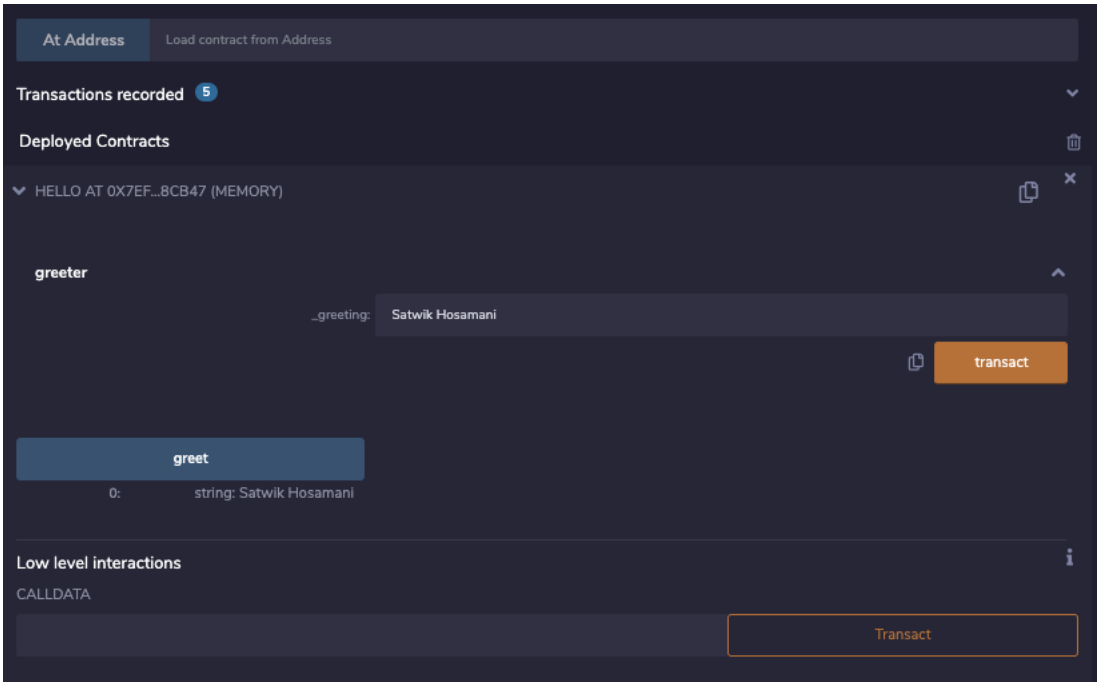
Solidity Program as provided:



Steps:

1.) Write the Code
2.) Compile it
3.) Deploy it
4.) Execute the Greet Function

Program Output

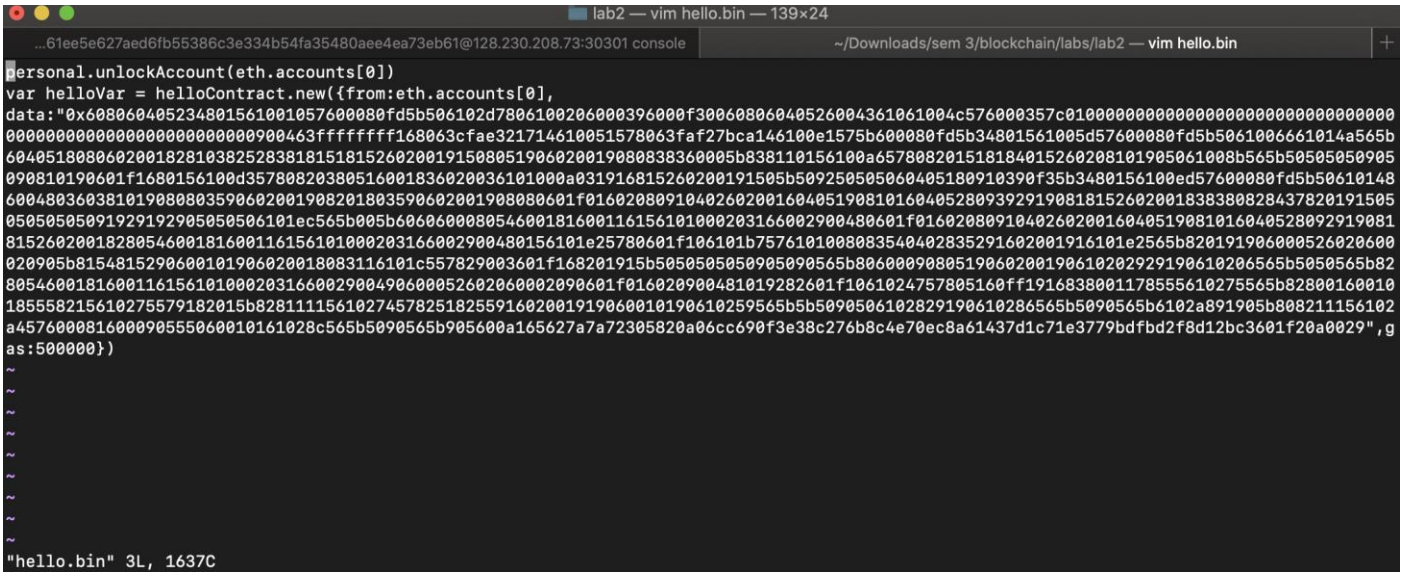


Exercise 1b (with 20% bonus): Hello-world Contract with Solc and On-Campus Ethereum

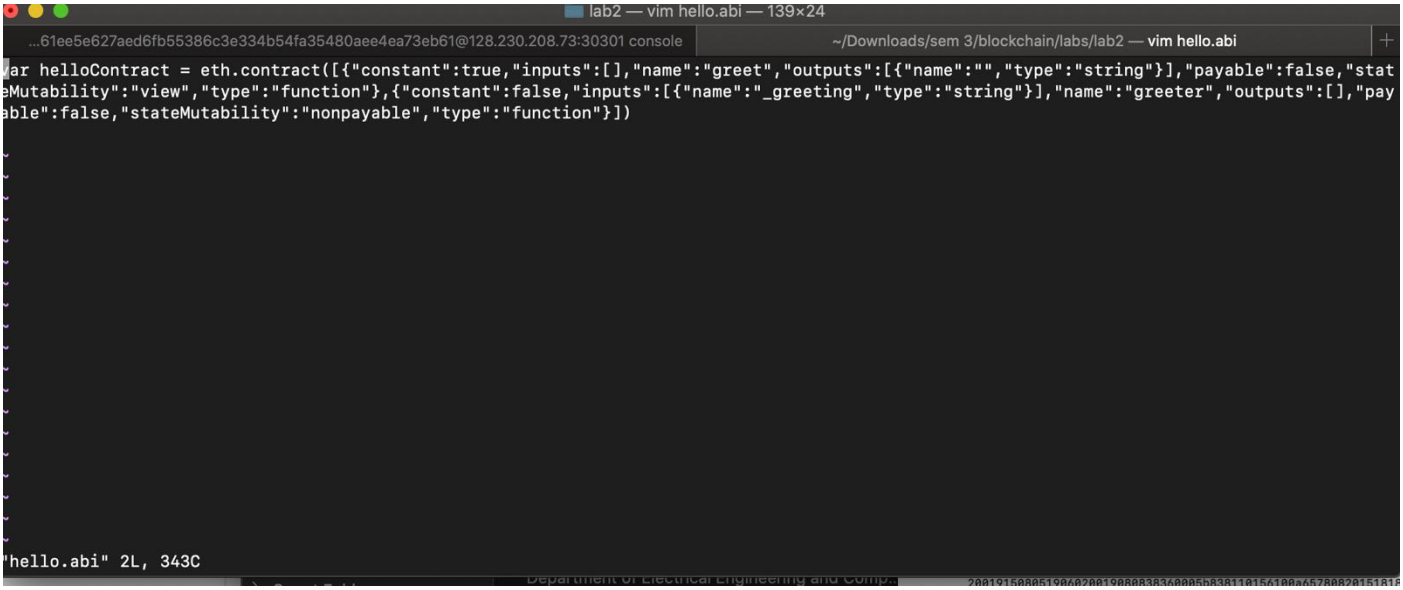
Steps:
1). Write Sol file code
2). Compile using solc

```
satwikhosamani@Satwiks-MBP lab2 % vim findmaximum.abi
satwikhosamani@Satwiks-MBP lab2 % vim findmaximum.bin
satwikhosamani@Satwiks-MBP lab2 % solc -o . --bin --abi hello.sol
satwikhosamani@Satwiks-MBP lab2 % ls
Blockchain Assignment 1 (1).pdf findmaximum.abi                genesis.json
Lab 1 create_ap(1).sql          findmaximum.adi                hello.abi
README.md                       findmaximum.bin                hello.bin
SHOSAMAN_BLOCKCHAIN_LAB1.pdf    findmaximum.sol                hello.sol
address_updates.md              finmaximum.abi                 task-2.js
bkc_data                        genesis 2.json
console.log                     genesis copy.json
satwikhosamani@Satwiks-MBP lab2 %
```

Compiling
1). Modify .bin and .abi files



BIN FILE



ABI FILE

1). Connect to ethereum network
2). Load and Deploy contract
>loadScript(“hello.abi”)
>loadScript(“hello.bin”)
3). Execute Contract

```
> helloVar.greeter.sendTransaction("Hello satwik", {from:eth.accounts[0], gas:700000})
"0x0f3f2d7ab7b8ce7fffbd1ddaa127f60ff727c3aa88653123de9080a992688635"
> helloVar.greet.call()
"Hello satwik"
>
```

Output display

Exercise 2: Find the Maximum

Steps:

1). Write the Code

...oads/sem 3/blockchain/labs/lab2 — vim findmaximum.sol ~/Downloads/sem 3/blockchain/labs/la

```
pragma solidity ^ 0.7.2;
        contract findmax{
        function maximum(int a,int b) public pure returns(int) {
        if(a<b)
        return b;
        else
        return a;
        }
}
~
~
~
~
~
~
~
~
~
~
~
```

Program

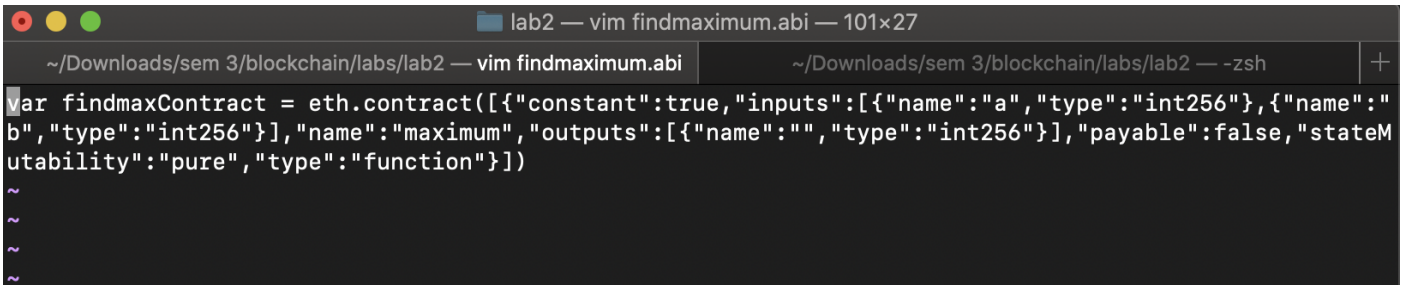
2). Compile and modify .bin and .abi files

lab2 — vim findmaximum.bin — 101×27

~/Downloads/sem 3/blockchain/labs/lab2 — vim findmaximum.bin ~/Downloads/sem 3/blockchain/labs/lab2 — -zsh

```
personal.unlockAccount(eth.accounts[0])
var findmaxVar = findmaxContract.new({from:eth.accounts[0],
data:"0x6080604052348015610010576000800fd5b506102d78061002060000396000f300608060405260043610610004c57600
0357c0100000000000000000000000000000000000000000000000000000000900463ffffffff168063cfae32171461005157
8063faf27bca146100e1575b600080fd5b34801561005d57600080fd5b5061006661014a565b604051808060200182810382
2838181518152602001915080519060200190808383600005b838110156100a657808201518184015260208101905061008b56
5b505050509050908101906001f1680156100d3578082038051600183602003610100a031916815260200191505b509250505
0604051809103900f35b3480156100ed57600080fd5b506101486004803603810190808035906020019082018035906020019
8080601f0160208091040260200160405190810160405280939291908181526020018383808284378201915050505050919
2919290505050506101ec565b005b6060600080546001816001161561010002031660029004806001f0160208091040260200160
4051908101604052809291908181526020018280546001816001161561010002031660029004801561006e25780601f106101b
7576101008083540402835291602001916101e2565b820191906000526020600020905b81548152906001019060200180831
6101c557829003601f168201915b505050505090509056 5b80600090805190602001906102029291906102065656 5b5050565b8
2805460018160011615610100020316600290049060005260206000209060 1f016020900481019282601f1061024757805160
ff1916838001178555610275565b828001600101855582156102755791820 15b828111156102745782518255916020019190 6
0010190610259565b5b509050610282919061028656 5b5090565b6102a891905b808211156102a4576000816000905506001
0161028c565b5090565b905600a165627a7a72305820a06cc690f3e38c276b8c4e70ec8a61437d1c71e3779bdfbd2f8d12bc3
601f20a0029",gas:500000})
~
~
~
~
~
~
~
~
"findmaximum.bin" 3L, 1641C
```

BIN FILE



ABI FILE

1). Connect to ethereum network and deploy contract

>loadScript("findmaximum.bin")

>loadScript("findmaximum.abi")

2). Execute contract

Contract Execution

Contract Call

SATWIK HOSAMANI 378000248

Exercise 3: Rock-paper-scissors game

Steps:

Step 1: Write Code:

Code Part 1

```
pragma solidity ^0.5.11;

contract rps
{
    mapping (string=>mapping(string=>int)) result;
    address payable public player1;
    address payable public player2;
    string public p1choice;
    string public p2choice;

    modifier unregistered() // modifier to check if player address is registered or not
    {
        if (msg.sender == player1 || msg.sender == player2)
            revert();
        else
            _;
    }

    modifier bidding(uint amount) // modifier to check if player paid 5 ethers
    {
        if (msg.value == amount)
            revert();
        else
            _;
    }

    constructor() public payable // creates rps winning matrix
    {
        result["rock"]["rock"] = 0;
        result["rock"]["paper"] = 2;
        result["rock"]["scissors"] = 1;
        result["paper"]["rock"] = 1;
        result["paper"]["paper"] = 0;
        result["paper"]["scissors"] = 2;
        result["scissors"]["rock"] = 2;
        result["scissors"]["paper"] = 1;
        result["scissors"]["scissors"] = 0;
    }

    function getWinner() view external returns (int X) //returns winner player number
    {
        return result[p1choice][p2choice];
    }
```

Code Part 2

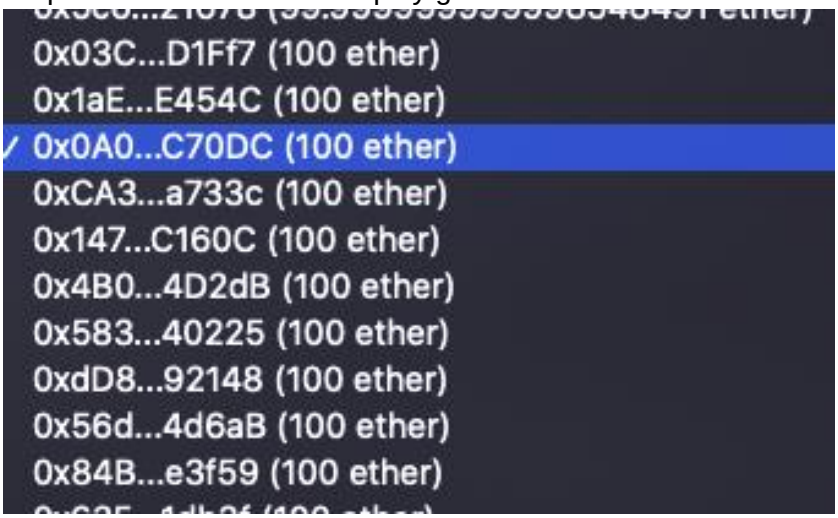
```
    function play(string memory choice) public payable returns (int w) //play function transfers et
    {
        if (msg.sender == player1)
            p1choice = choice;
        else if (msg.sender == player2)
            p2choice = choice;
            if (bytes(p1choice).length != 0 && bytes(p2choice).length != 0)
            {
                int winner = result[p1choice][p2choice];
                if (winner == 1)
                    player1.transfer(address(this).balance); //p1wins
                else if (winner == 2)
                    player2.transfer(address(this).balance); //p1wins
                else
                {
                    player1.transfer(address(this).balance/2);
                    player2.transfer(address(this).balance);
                }

                return winner;
            }
            else
                return -1;
    }

    function getBalance() public returns (uint x) //function to return current balance of palyer
    {
        return msg.sender.balance;

    }
```

Code Part 3

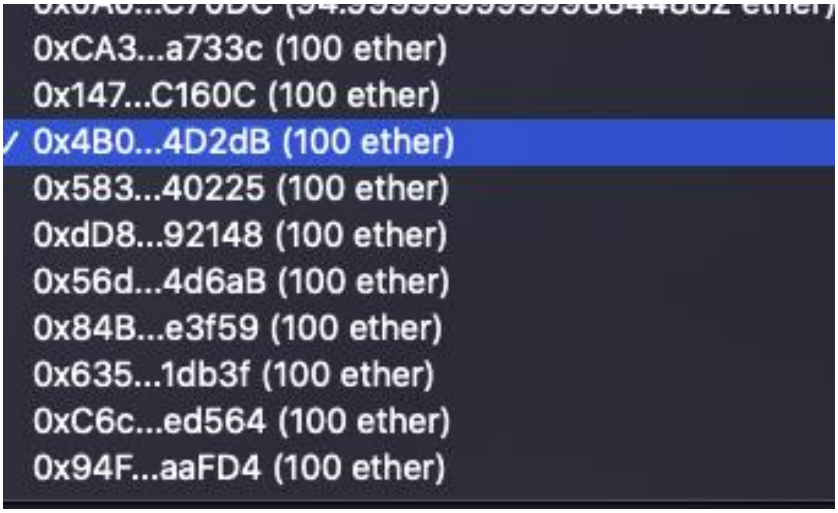
```
    function prizepool() public returns (uint x) //function to balance of prizepool - contract
    {
        return address(this).balance;

    }

    function register() public payable bidding(5) unregistered() //function to register new players
    {
        if (uint(player1) == 0)
            player1 = msg.sender;
        else if (uint(player2) == 0)
            player2 = msg.sender;
    }
}
```

Step 2. Compile the smart contract & Deploy

Step 3. Provide Accounts to play game

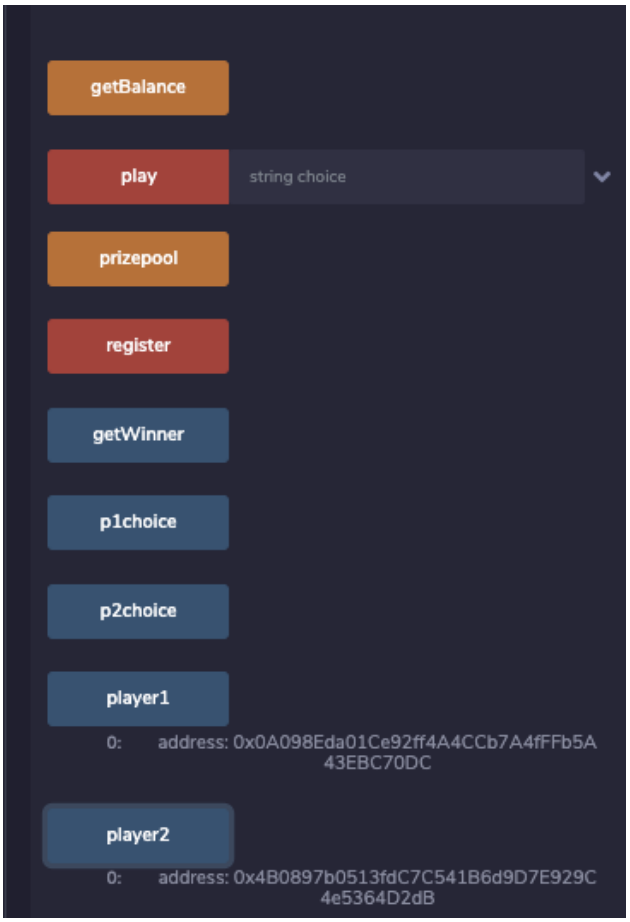


Account 1



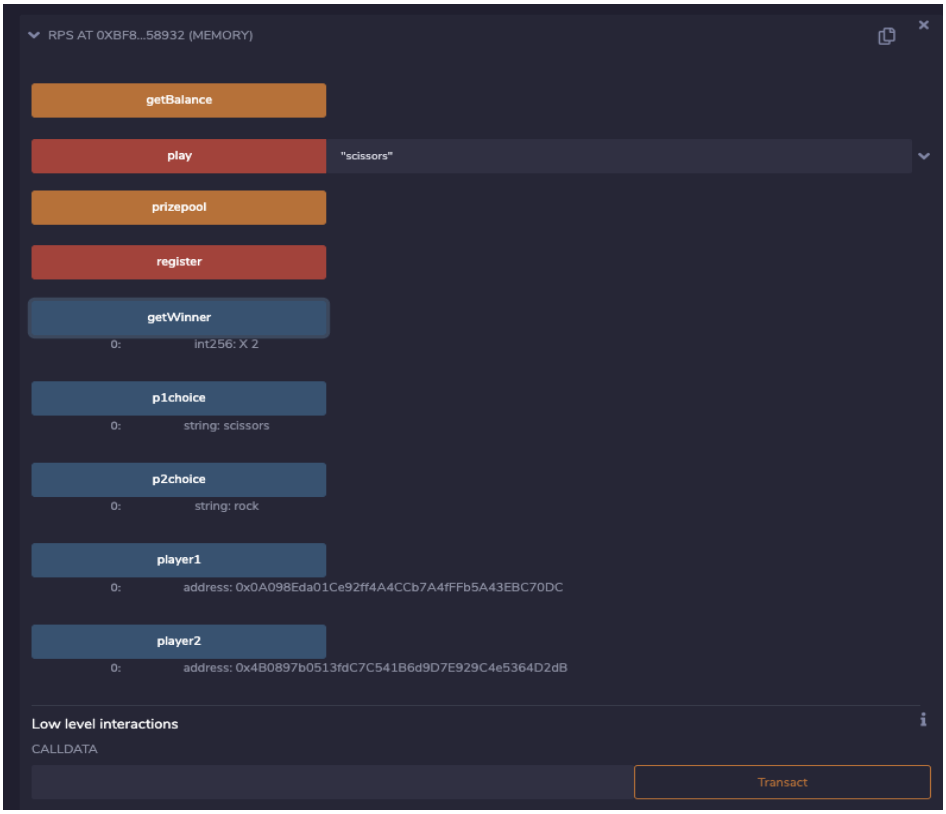
Account 2

Step 4: Register both accounts with 5 ethers



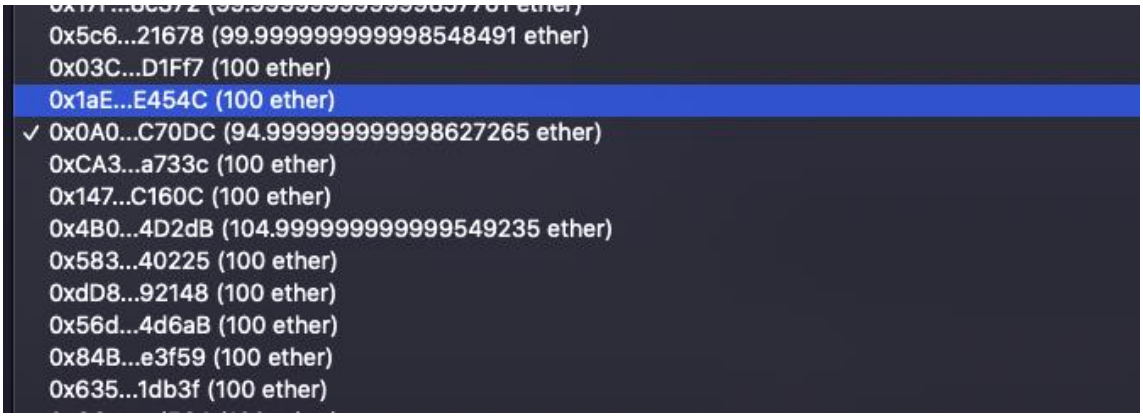
Registered Accounts

Step 5. Execute ‘Play’ with their respective choices



Player 1: Scissors Player 2: Rock

Step 6. After both accounts choose their choice, winner is declared and ethers are transferred



Ether Transfer

WINNER: Player 2