ASSIGNMENT 06

1) **Are Ommer blocks included in the blockchain?**

   Ans:

   Ommer blocks are created in the Ethereum blockchain when two blocks are created and submitted to the ledger at roughly the same time.

   Only one can enter the ledger. Ommer block generated another block after that.

2) **What is CREATE2 in Solidity or EVM or Ethereum?**

   Ans: CREATE2 opcode gives us the ability predict the address where a contract will be deployed, without ever having to do so. This opens up lots of possibilities to improve user onboarding and scalability. In this guide we will precompute the address where a contract will be deployed and send Ether to it.

3) **What happens if you try to store 256 using uint8?**

   ANS: it rolls over to zero after 255.

   uint8 has a range of 0 to 2 ** 8 -1

4) **What is the difference between address & address payable?**

   Ans: The address and address payable types both store a 160-bit Ethereum address.

   The concept of payable and non-payable addresses only exists in the Solidity type system at compile-time. The difference between payable and non-payable addresses is gone in the compiled contract code.

5) **Apart from msg.sender & block.timestamp, list all the global variables of Solidity.**

Ans:

block.coinbase (address payable)	Current block miner's address
block.difficulty (uint)	Current block difficulty
msg.value (uint)	Number of wei sent with the message
block.number (uint):	Current block number
blockhash(uint blockNumber) returns (bytes32)	Gives hash of the given block and will only work for the 256 most recent block due to the reason of scalability.
block.timestamp:	Current block timestamp as seconds since unix epoch
gasleft() returns (uint256):	Remaining gas
msg.sender (address payable)	Sender of the message (current call)
msg.sig (bytes4)	First four bytes of the calldata (i.e. function identifier)
now (uint)	Current block timestamp (alias for block.timestamp)
tx.gasprice (uint)	Gas price of the transaction
block.gaslimit (uint)	Current block gaslimit
tx.origin (address payable)	Sender of the transaction (full call chain)
msg.data (bytes calldata)	Complete calldata

6) What is the difference between view & pure functions in Solidity?

Ans:

View functions are read only functions and do not modify the state of the block chain. In other words if you want to read data from the block chain one can use view.

Pure functions are more restrictive then view functions and do not modify the state *AND* do not read the state of the block chain.

	DEFAULT (NOT SPECIFIED IN FUNCTION)	VIEW	PURE
Ideal For	Transactions that alter data on the block chain	Getter functions to view data on the block chain	Defined to the scope of the function and do not alter or view data on the block chain
Data Access	read / write	read	None
Transaction Type	Send	Call	Call

7) Selfdestruct in Solidity?

Selfdestruct is a keyword that is used to terminate a contract, remove the bytecode from the Ethereum blockchain, and send any contract funds to a specified address.

8) Why memory keyword is used in case of arrays, strings & bytes in Solidity in function arguments or return statements?

Ans: Memory is a keyword used to store data for the execution of a contract. It holds functions argument data and is wiped after execution. storage can be seen as the default solidity data storage. It holds data persistently and consumes more gas.

9) **How to call external functions from within the same contract in Solidity?**

   Ans: How do you call an external function in Solidity?
   External functions are part of the contract interface, which means they can be called from other contracts and via transactions. An external function f cannot be called internally (i.e. f() does not work, but this. f() works). The two are incompatible, public allows internal calls, while external doesn't

***COMPLETE SOLIDITY BEGINNER COURSE FROM LEARNETH FROM REMIX IDE**

SOLIDITY BEGINNER COURSE SOLUTION CONTRACT SHOULD BE ADDED IN THE FOLLOWING ORDER

CHAPTER NAME

Assignment

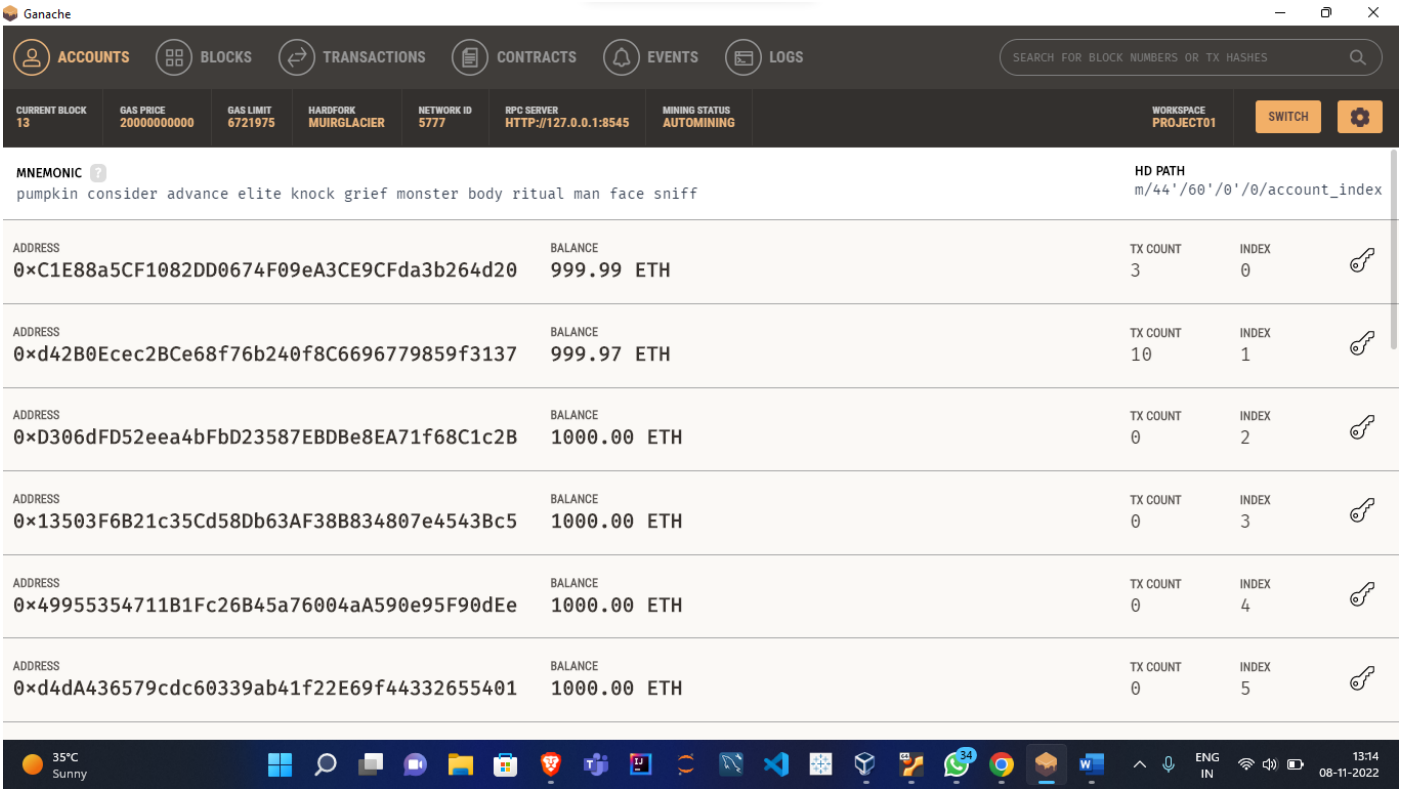
Details of the Assignment

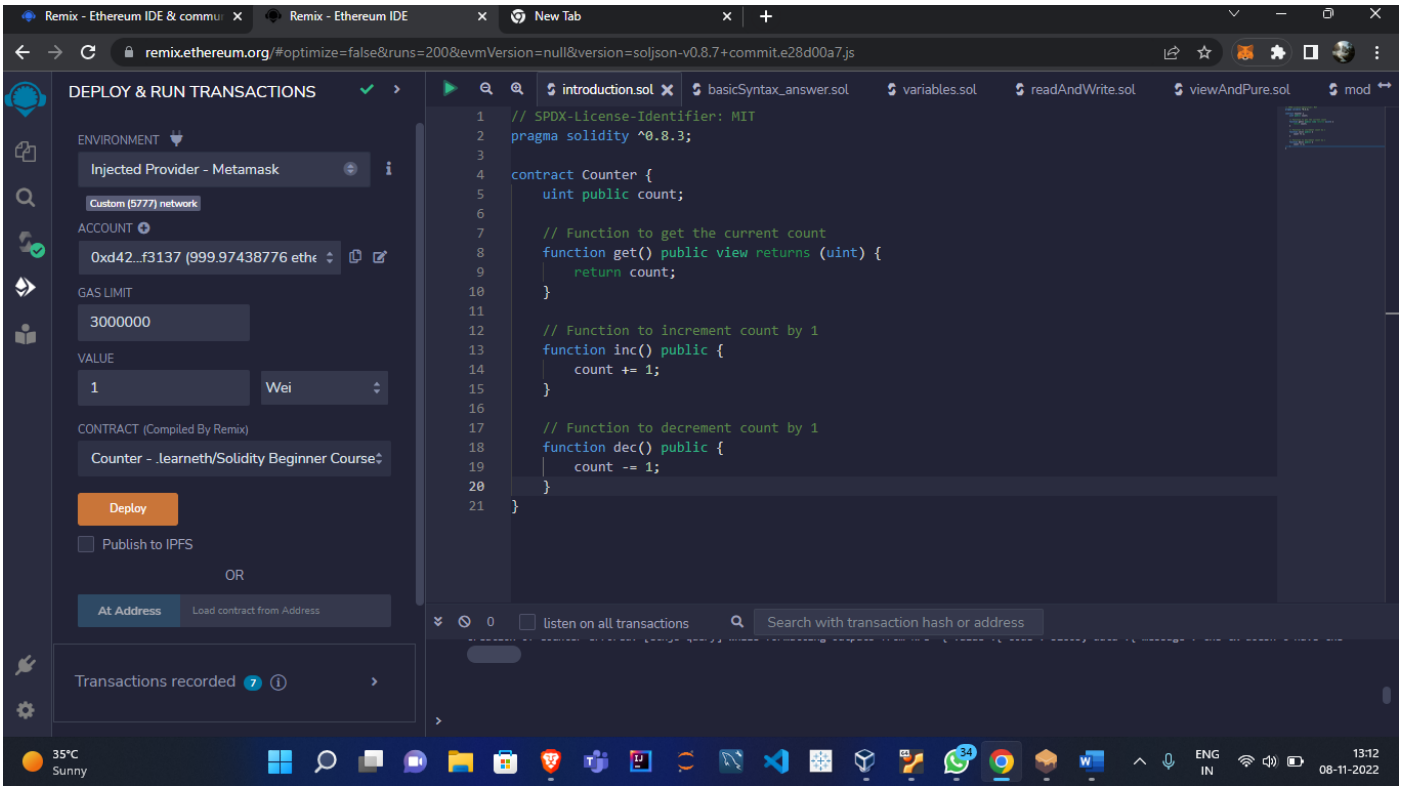
SOLUTION CONTRACT [ENTIRE SMART CONTRACT CODE]

Deployed Address on Ganache

1. Introduction:

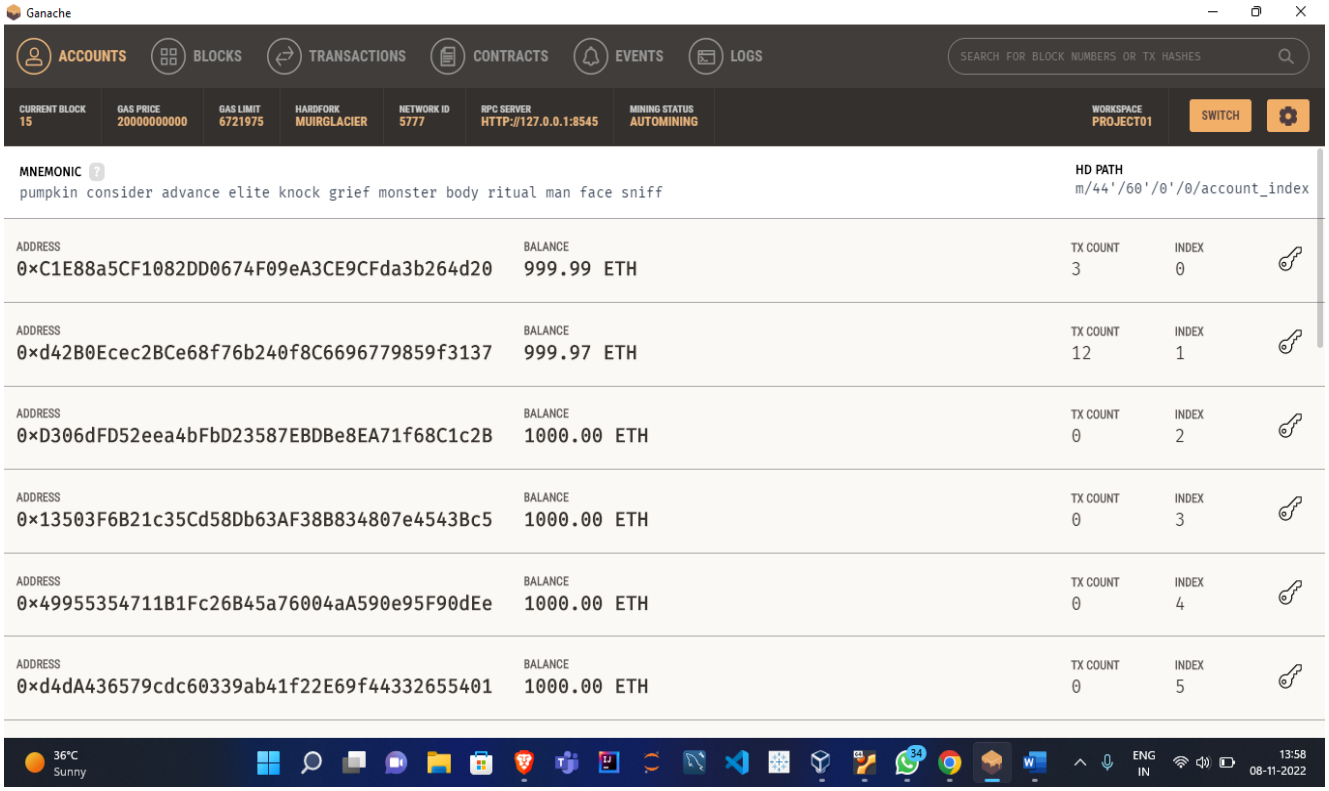
- ➢ Compile this contract.
- ➢ Deploy it to the JavaScript VM.
- ➢ Interact with your contract.

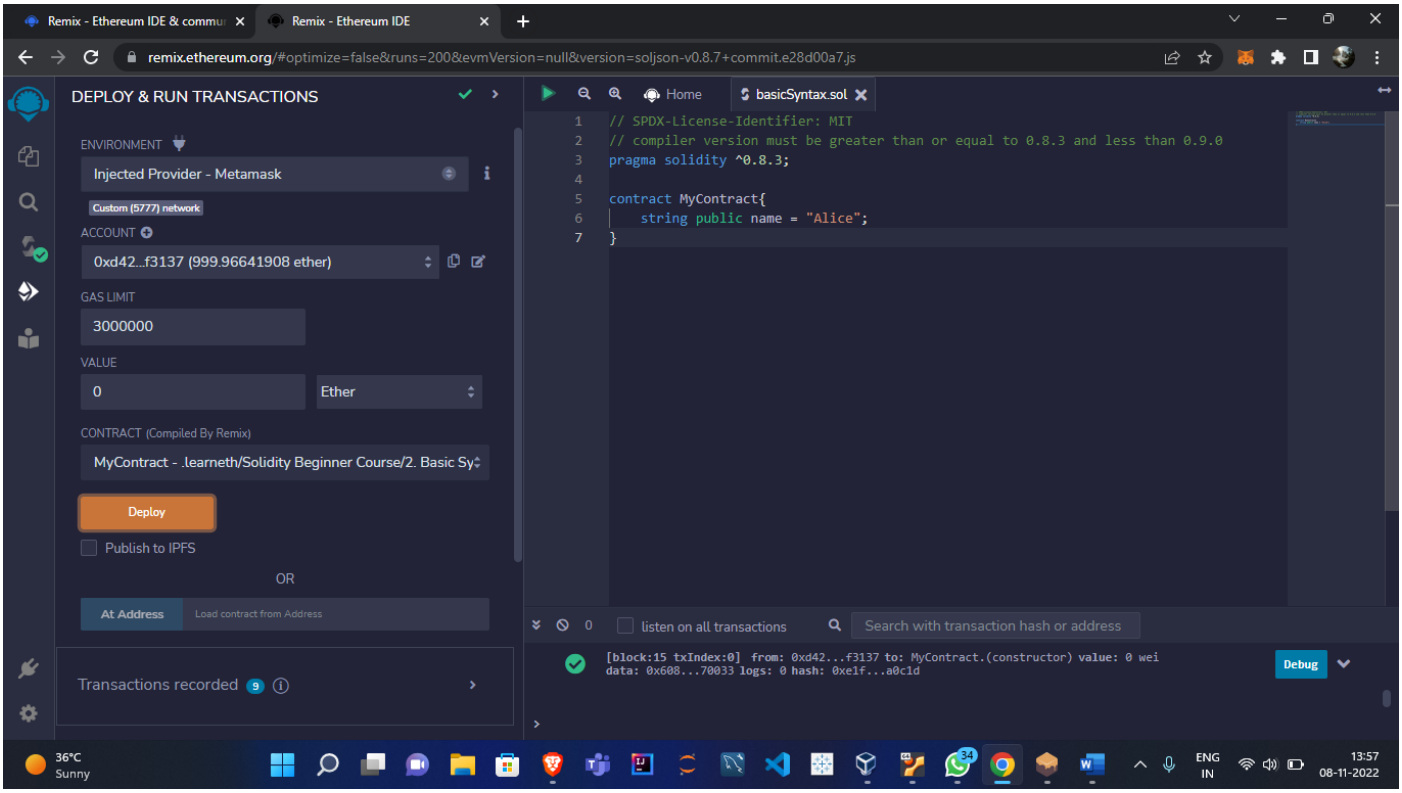




2. Basic Syntax Assignment :

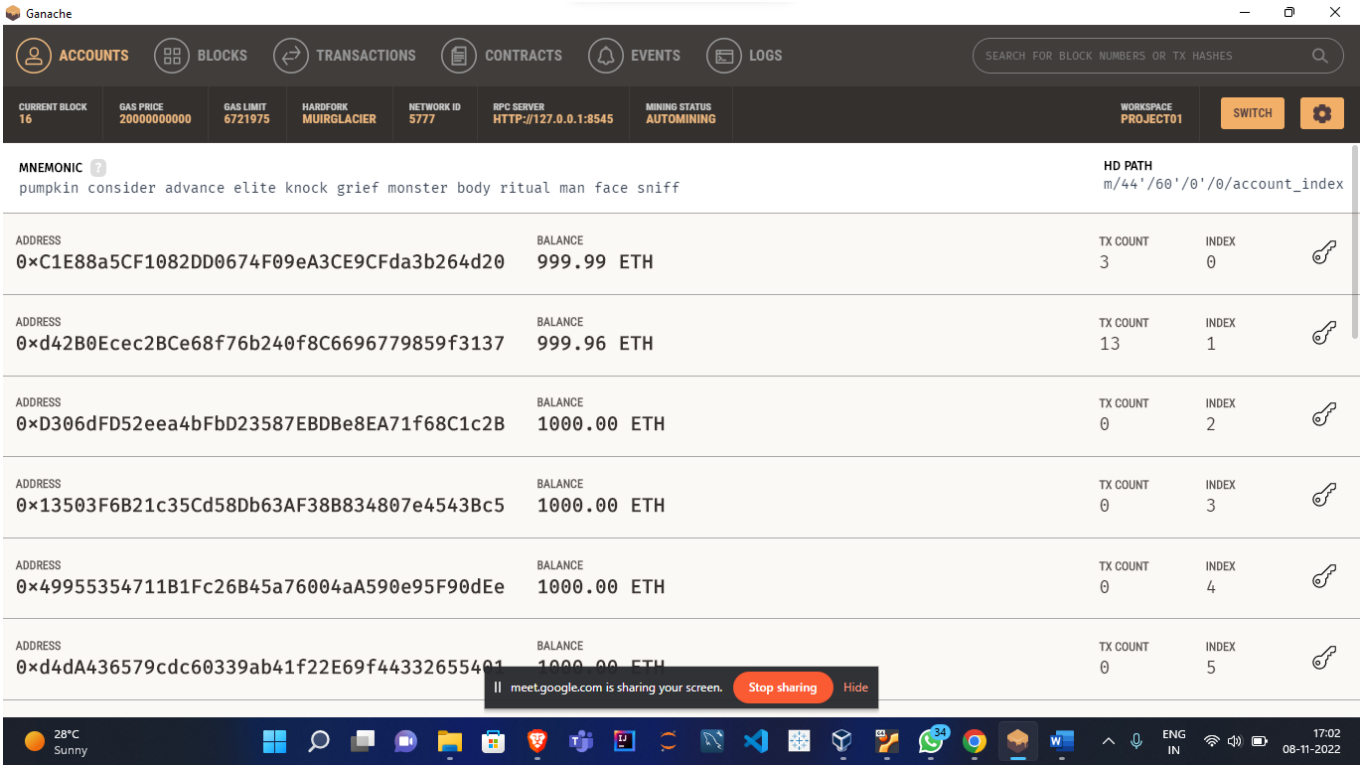
- ➢ **Delete the HelloWorld contract and its content.**
- ➢ **Create a new contract named "MyContract".**
- ➢ **The contract should have a public state variable called "name" of the type string.**
- ➢ **Assign the value "Alice" to your new variable.**

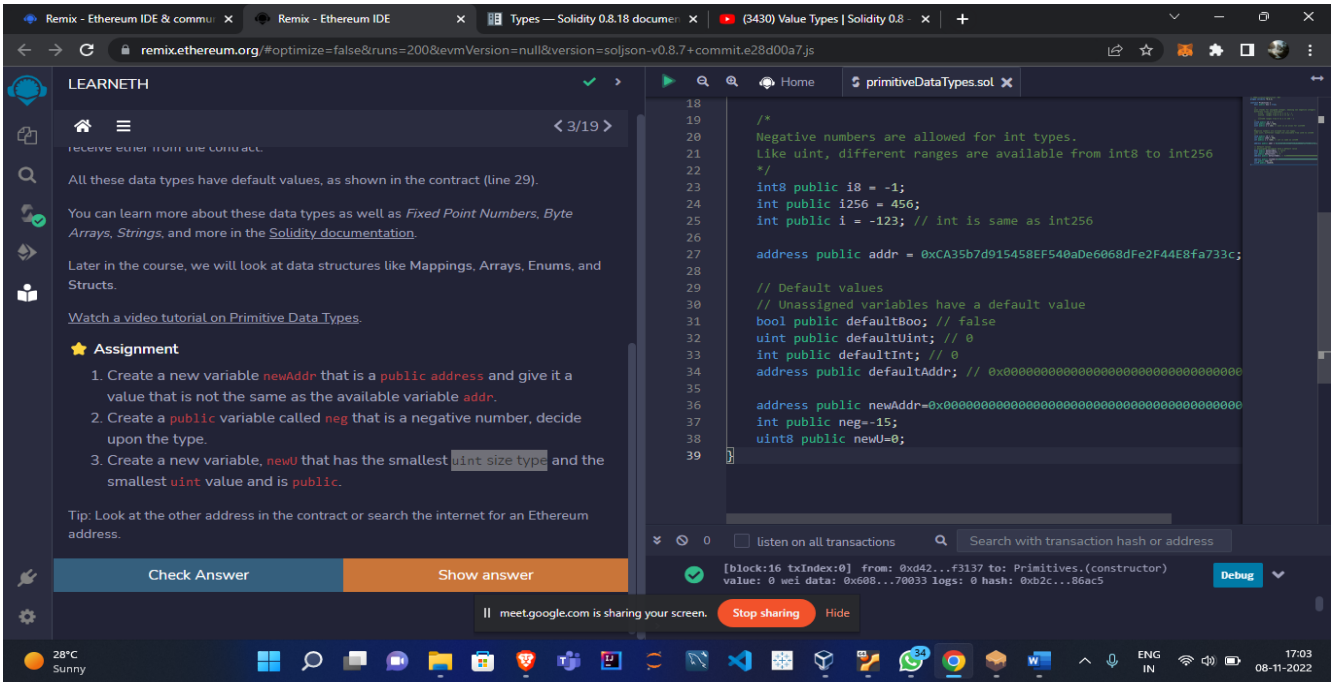




3. Primitive Data Types Assignment:

- **Create a new variable newAddr that is a public address and give it a value that is not the same as the available variable addr.**
- **Create a public variable called neg that is a negative number, decide upon the type.**
- **Create a new variable, newU that has the smallest uint size type and the smallest uint value and is public.**

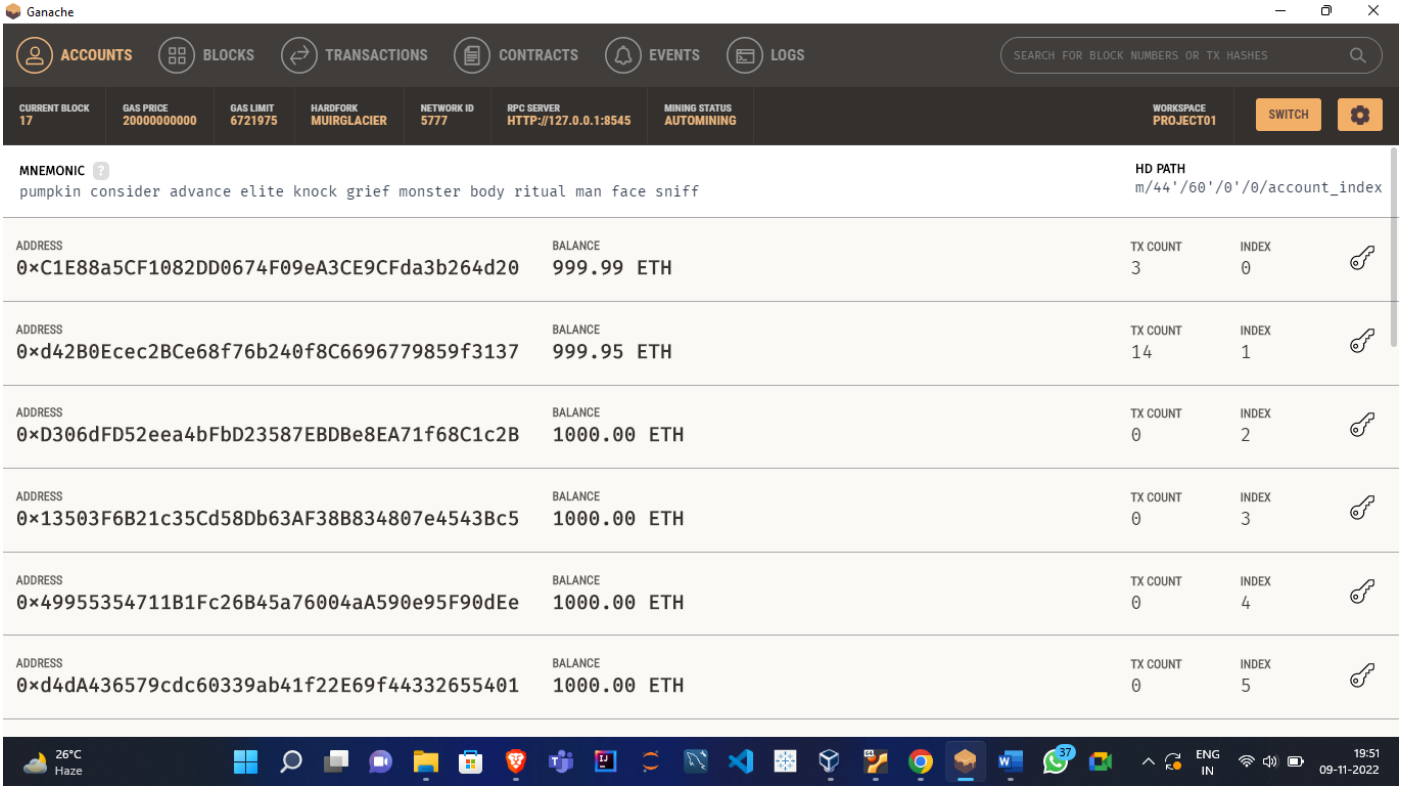


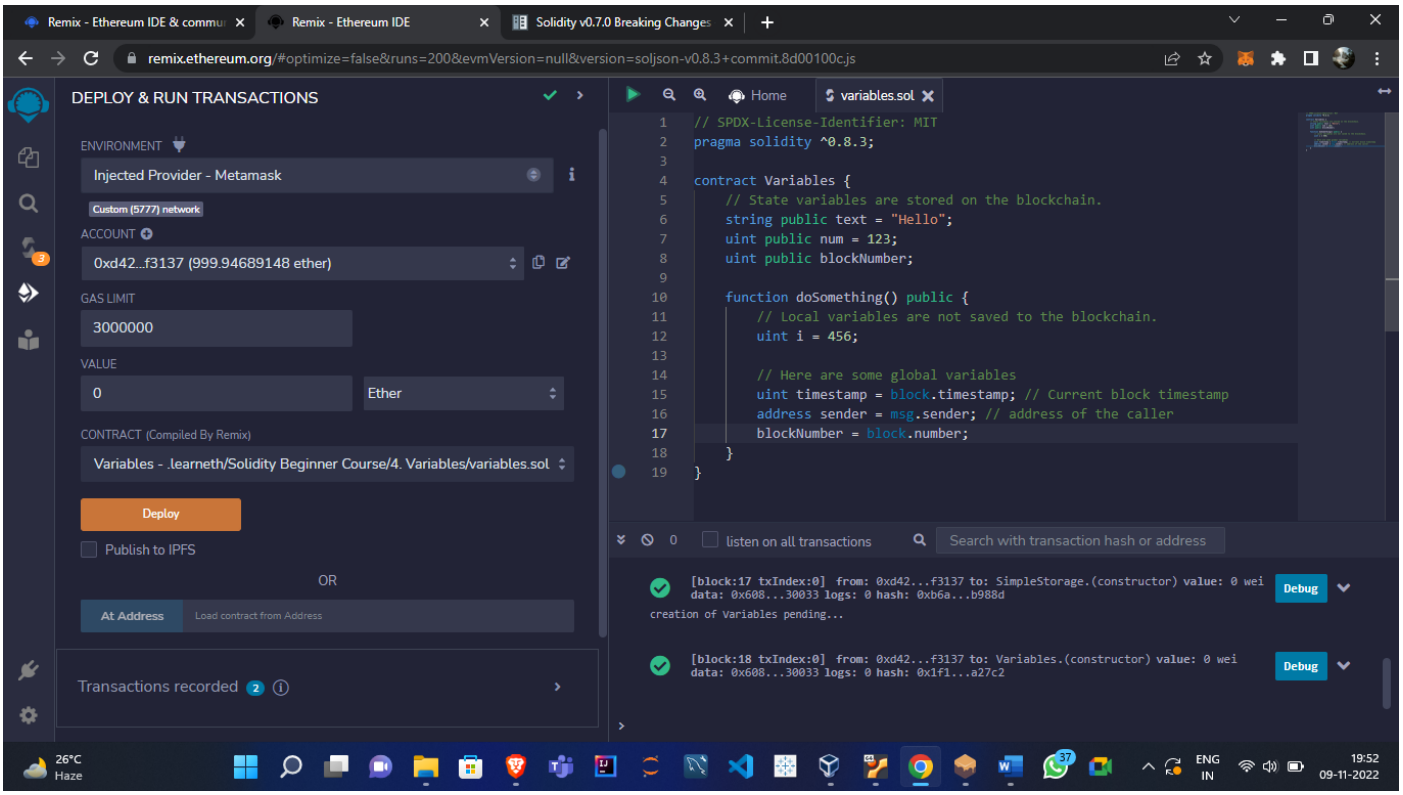


4. Variables : Assignment

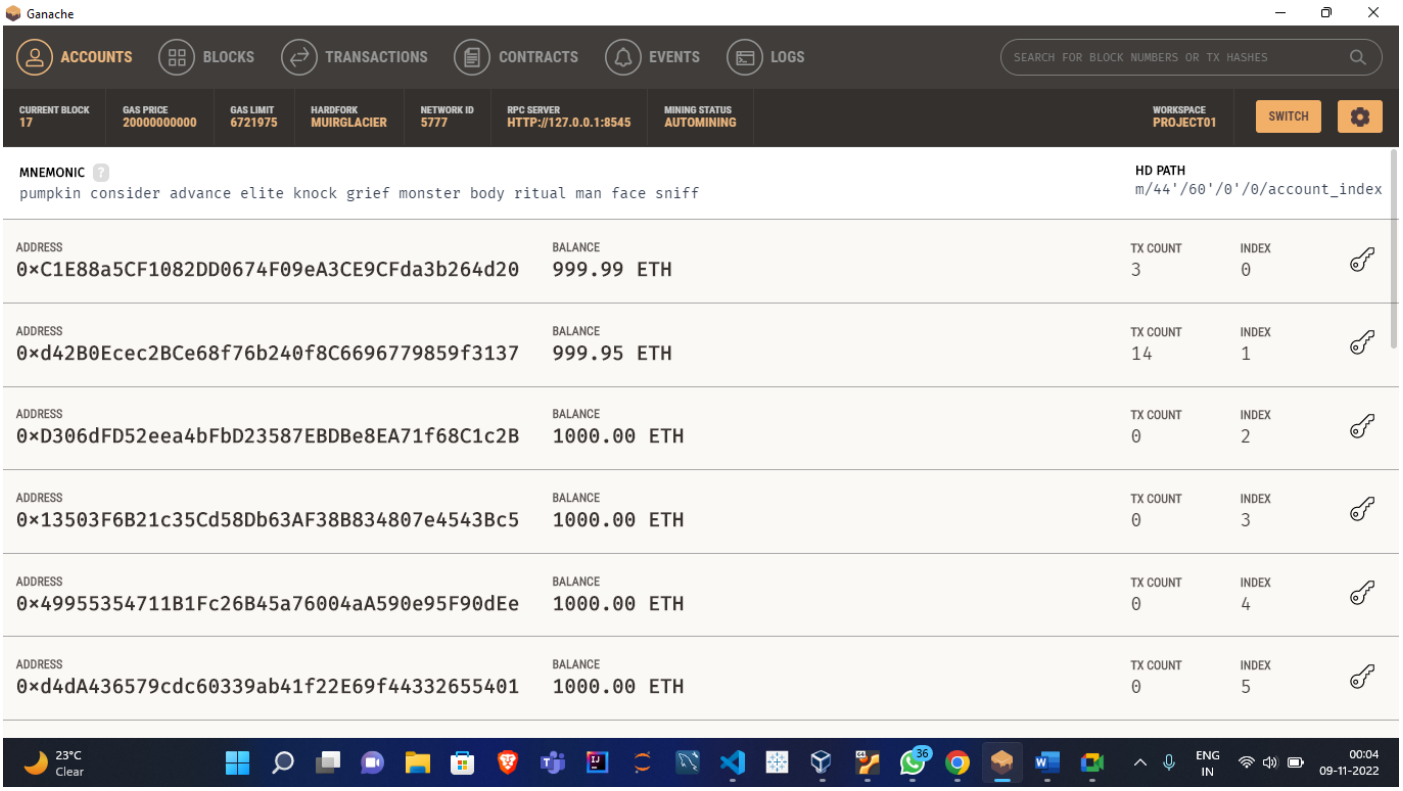
Create a new public state variable called blockNumber.

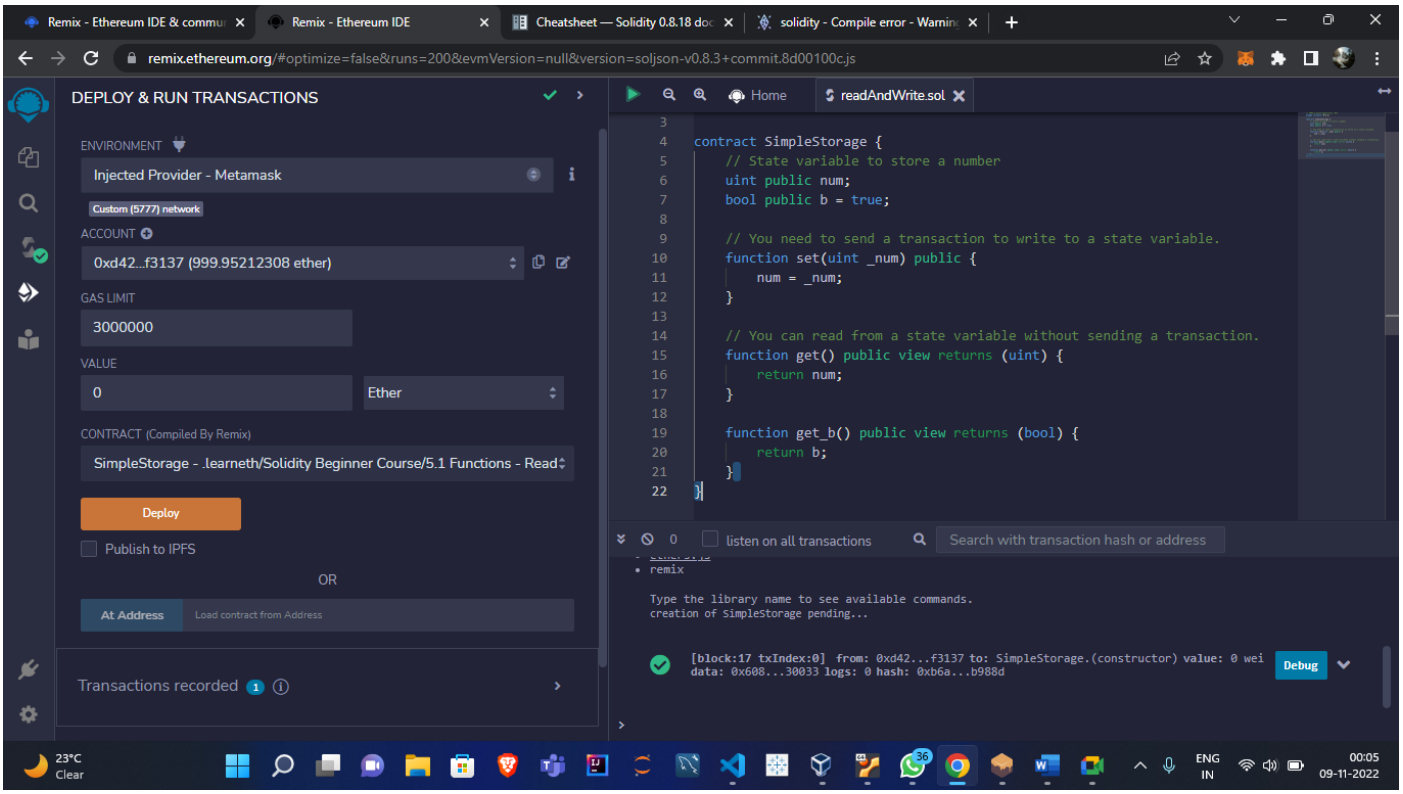
Inside the function doSomething(), assign the value of the current block number to the state variable blockNumber.





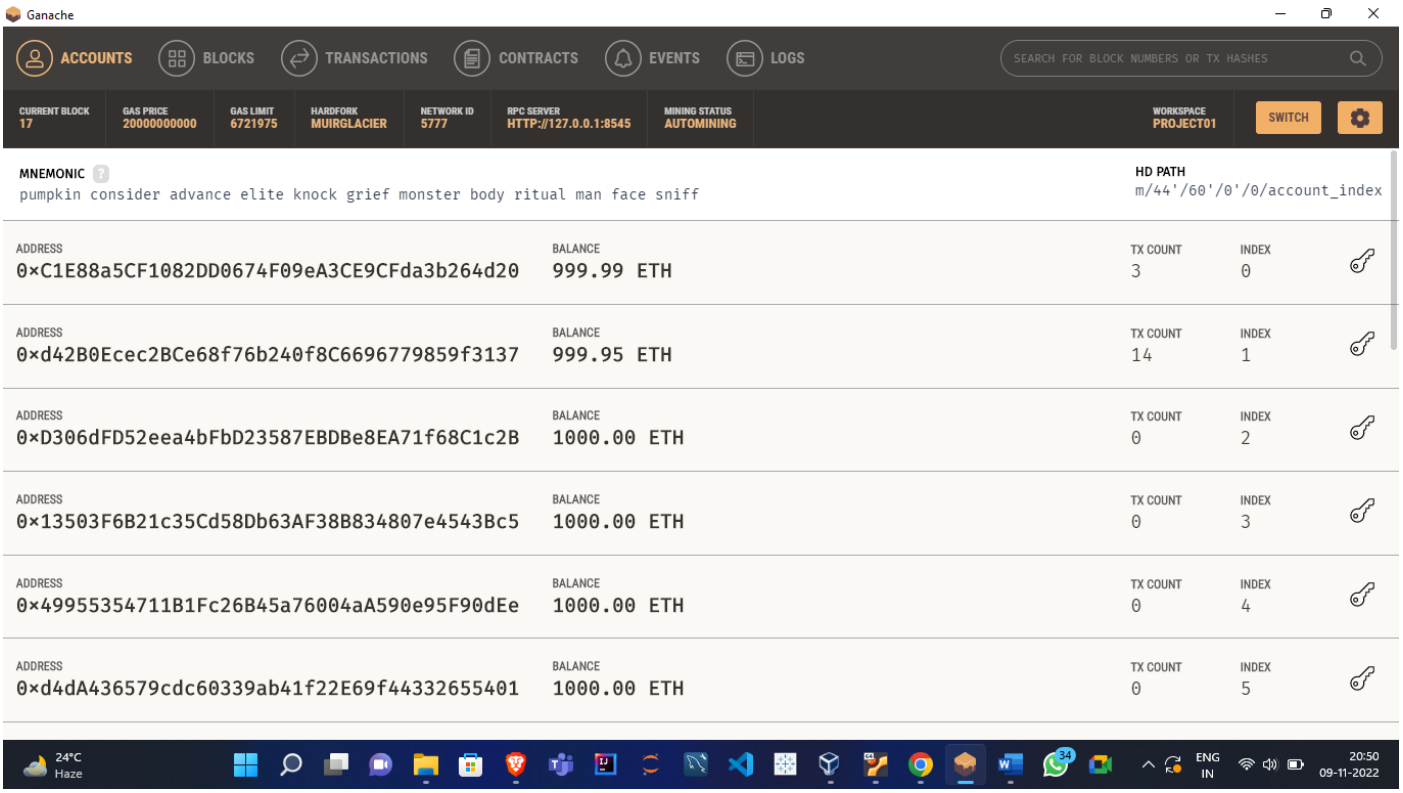
5.1)Functions - Reading and Writing to a State Variable

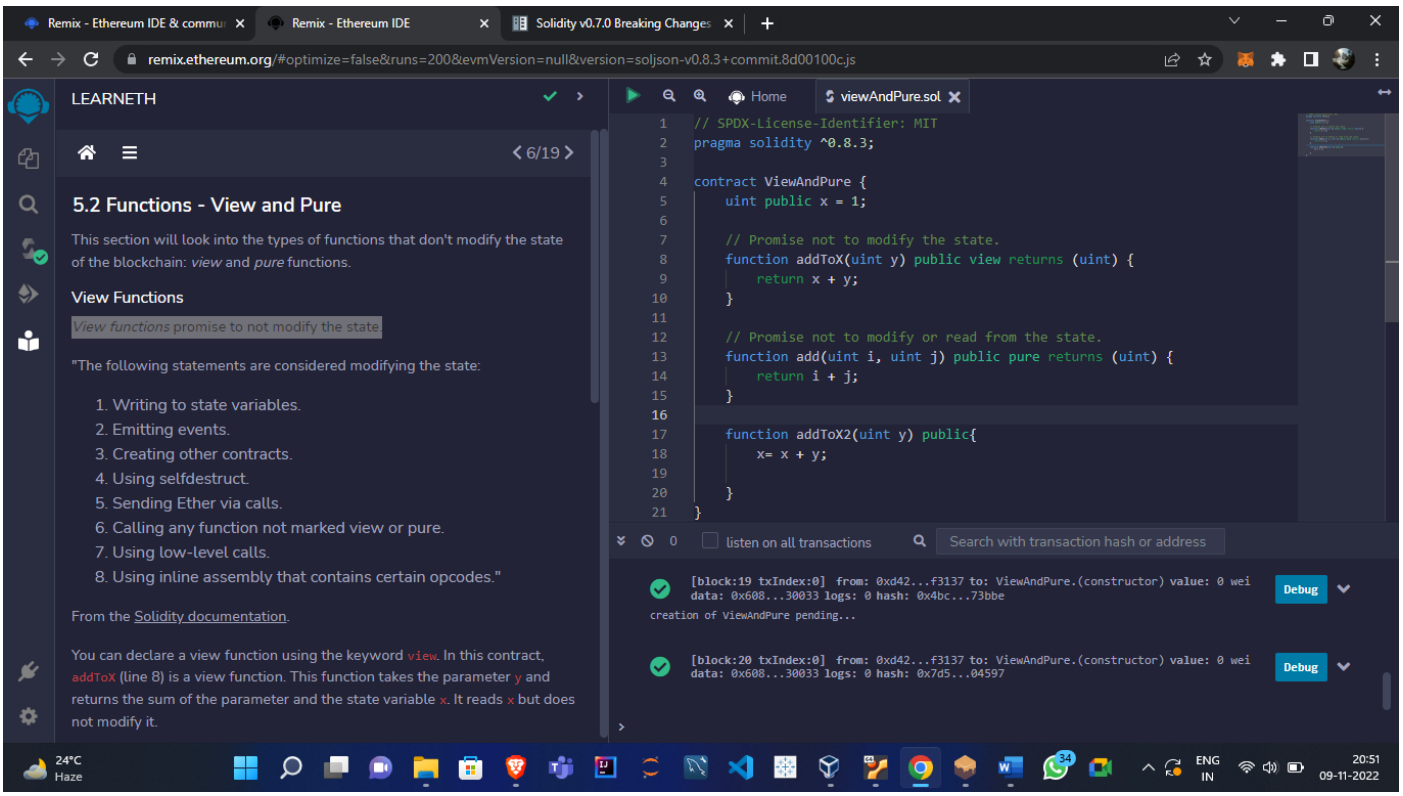

Ganache
ACCOUNTS
BLOCKS
TRANSACTIONS
CONTRACTS
EVENTS
LOGS
SEARCH FOR BLOCK NUMBERS OR TX HASHES
CURRENT BLOCK 17
GAS PRICE 20000000000
GAS LIMIT 6721975
HARDFORK MUIRGLACIER
NETWORK ID 5777
RPC SERVER HTTP://127.0.0.1:8545
MINING STATUS AUTOMINING
WORKSPACE PROJECT01
SWITCH
MNEMONIC
pumpkin consider advance elite knock grief monster body ritual man face sniff
HD PATH
m/44'/60'/0'/0/account_index
ADDRESS 0x C1E88a5CF1082DD0674F09eA3CE9CFda3b264d20 BALANCE 999.99 ETH TX COUNT 3 INDEX 0
ADDRESS 0xd42B0Ecec2BCe68f76b240f8C6696779859f3137 BALANCE 999.95 ETH TX COUNT 14 INDEX 1
ADDRESS 0xD306dFD52eea4bFbD23587EBDBe8EA71f68C1c2B BALANCE 1000.00 ETH TX COUNT 0 INDEX 2
ADDRESS 0x13503F6B21c35Cd58Db63AF38B834807e4543Bc5 BALANCE 1000.00 ETH TX COUNT 0 INDEX 3
ADDRESS 0x49955354711B1Fc26B45a76004aA590e95F90dEe BALANCE 1000.00 ETH TX COUNT 0 INDEX 4
ADDRESS 0xd4dA436579cdc60339ab41f22E69f44332655401 BALANCE 1000.00 ETH TX COUNT 0 INDEX 5
23°C Clear
ENG IN
00:04 09-11-2022


Remix - Ethereum IDE & commu
Remix - Ethereum IDE
Cheatsheet — Solidity 0.8.18 do
solidity - Compile error - Warnin
remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.3+commit.8d00100c.js
DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Injected Provider - Metamask
Custom (5777) network
ACCOUNT
0xd42...f3137 (999.95212308 ether)
GAS LIMIT
3000000
VALUE
0
Ether
CONTRACT (Compiled By Remix)
SimpleStorage - .learneth/Solidity Beginner Course/5.1 Functions - Read
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 1
Home
readAndWrite.sol
contract SimpleStorage {
// State variable to store a number
uint public num;
bool public b = true;
// You need to send a transaction to write to a state variable.
function set(uint _num) public {
num = _num;
}
// You can read from a state variable without sending a transaction.
function get() public view returns (uint) {
return num;
}
function get_b() public view returns (bool) {
return b;
}
}
listen on all transactions
Search with transaction hash or address
remix
Type the library name to see available commands.
creation of SimpleStorage pending...
[block:17 txIndex:0] from: 0xd42...f3137 to: SimpleStorage.(constructor) value: 0 wei
data: 0x608...30033 logs: 0 hash: 0xb6a...b988d
Debug
23°C Clear
ENG IN
00:05 09-11-2022

5.2)Functions - View and Pure : Assignment

Create a function called addToX2 that takes the parameter y and updates the state variable x with the sum of the parameter and the state variable x.



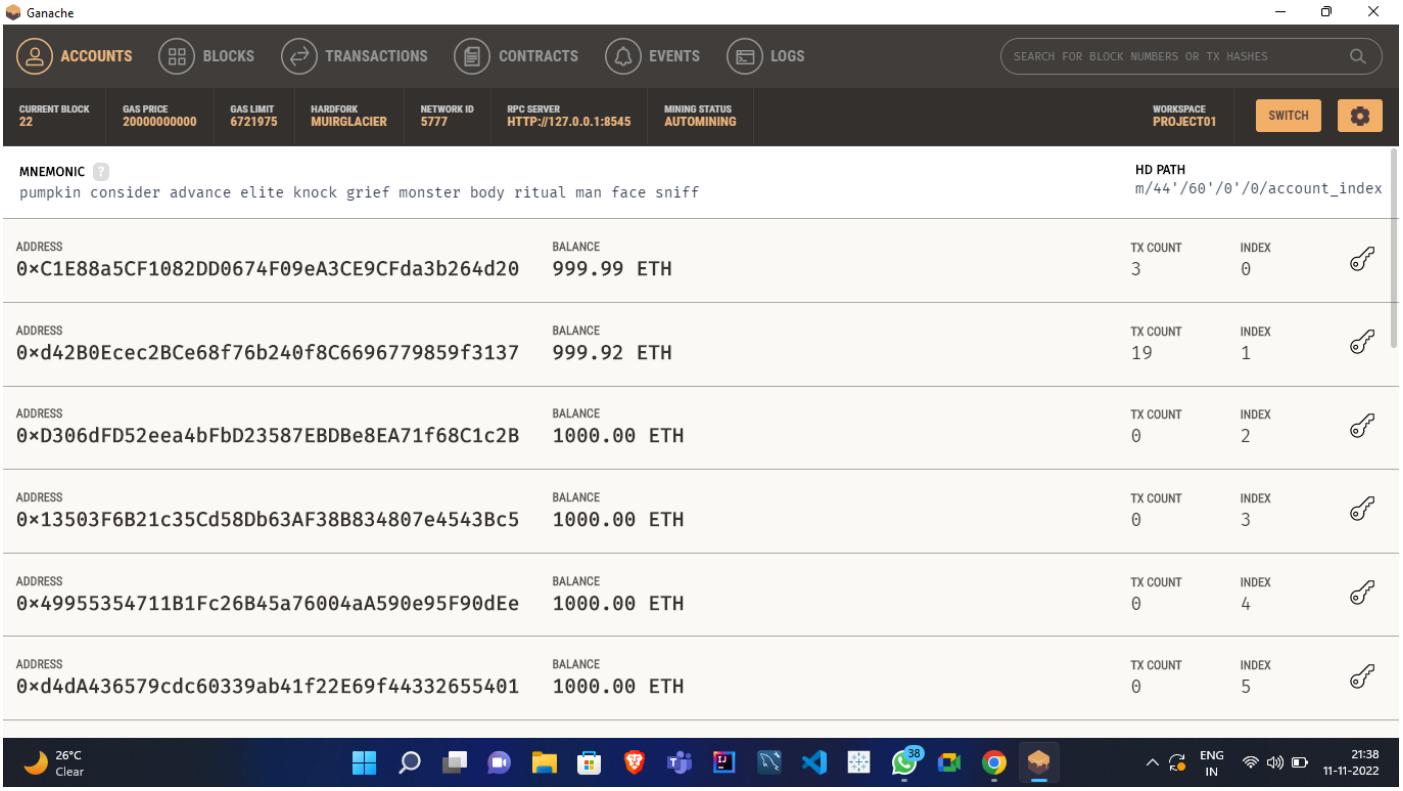


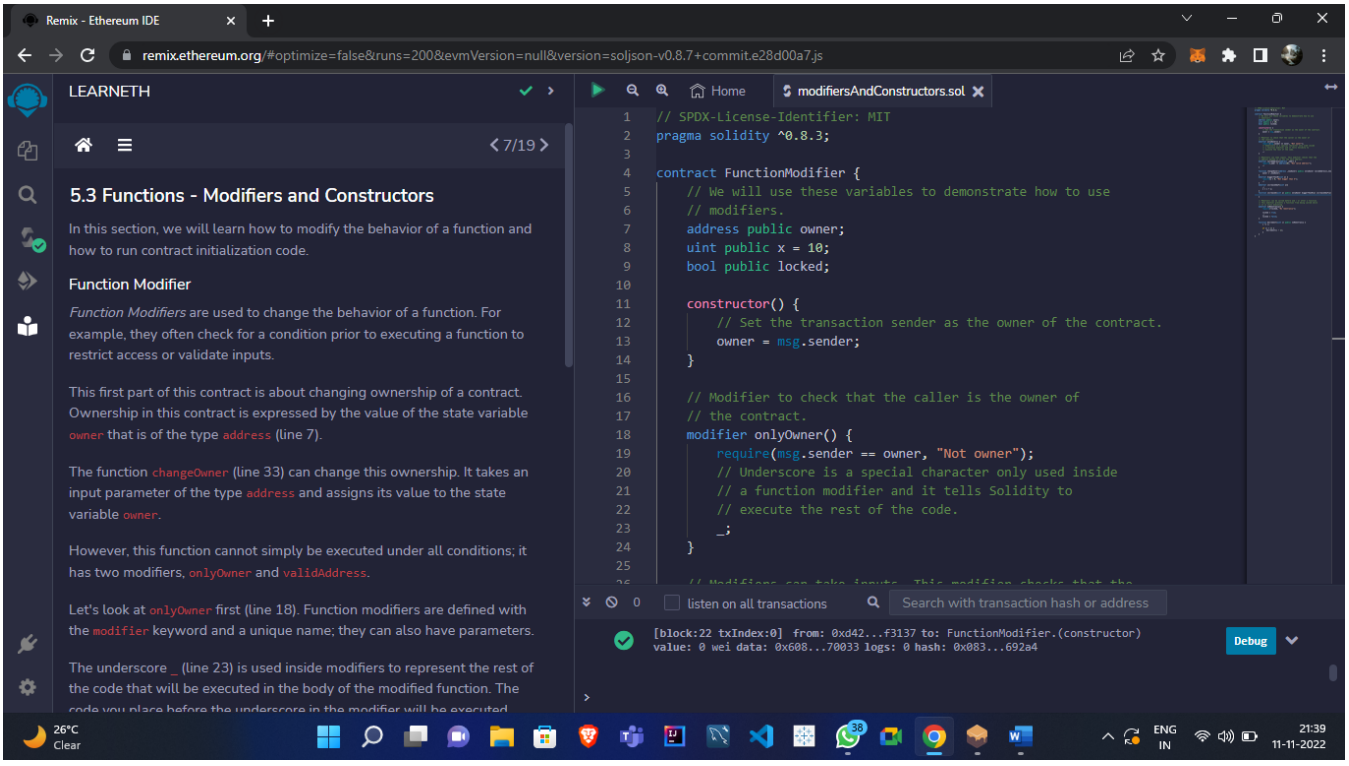
5.3 Functions - Modifiers and Constructors: Assignment

Create a new function, increaseX in the contract. The function should take an input parameter of type uint and increase the value of the variable x by the value of the input parameter.

Make sure that x can only be increased.

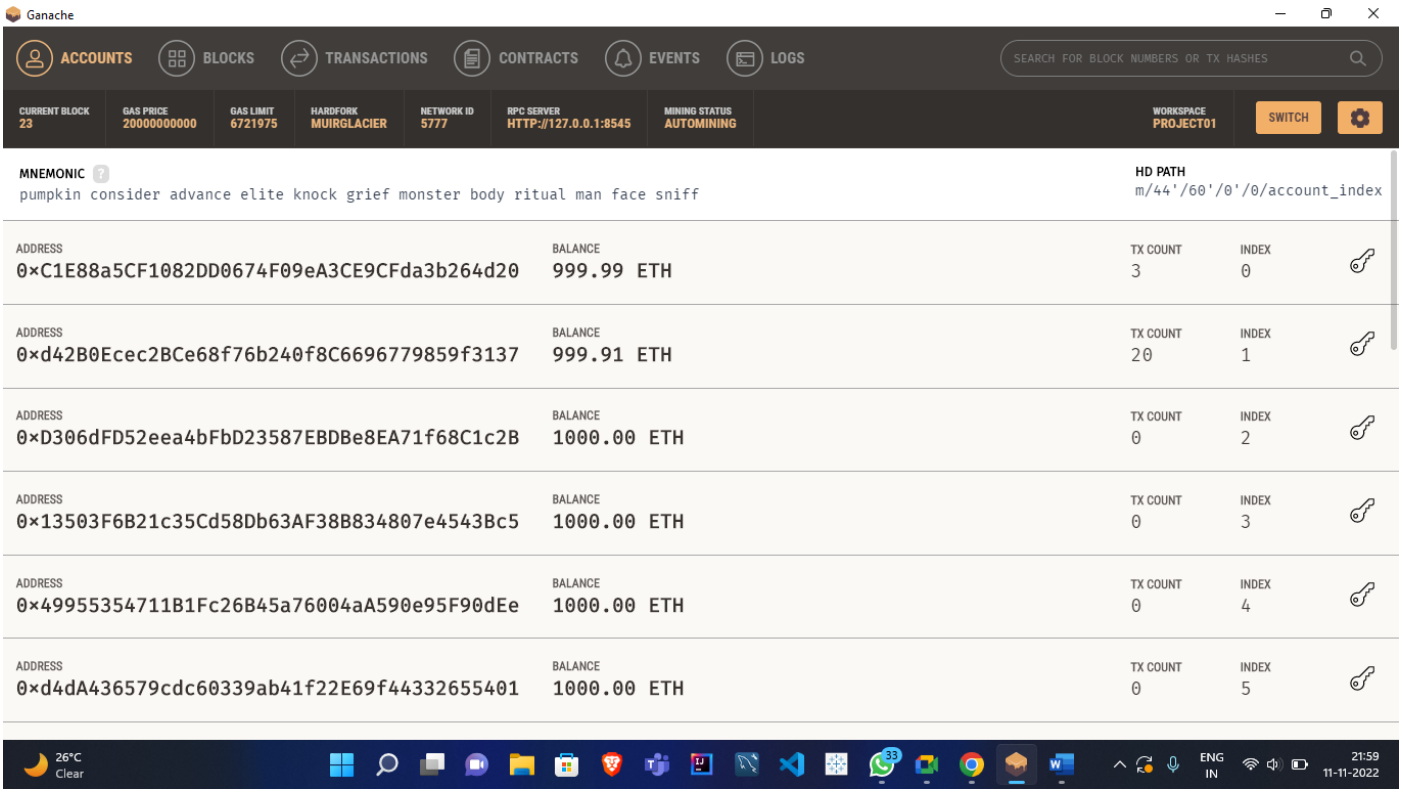
The body of the function increaseX should be empty.

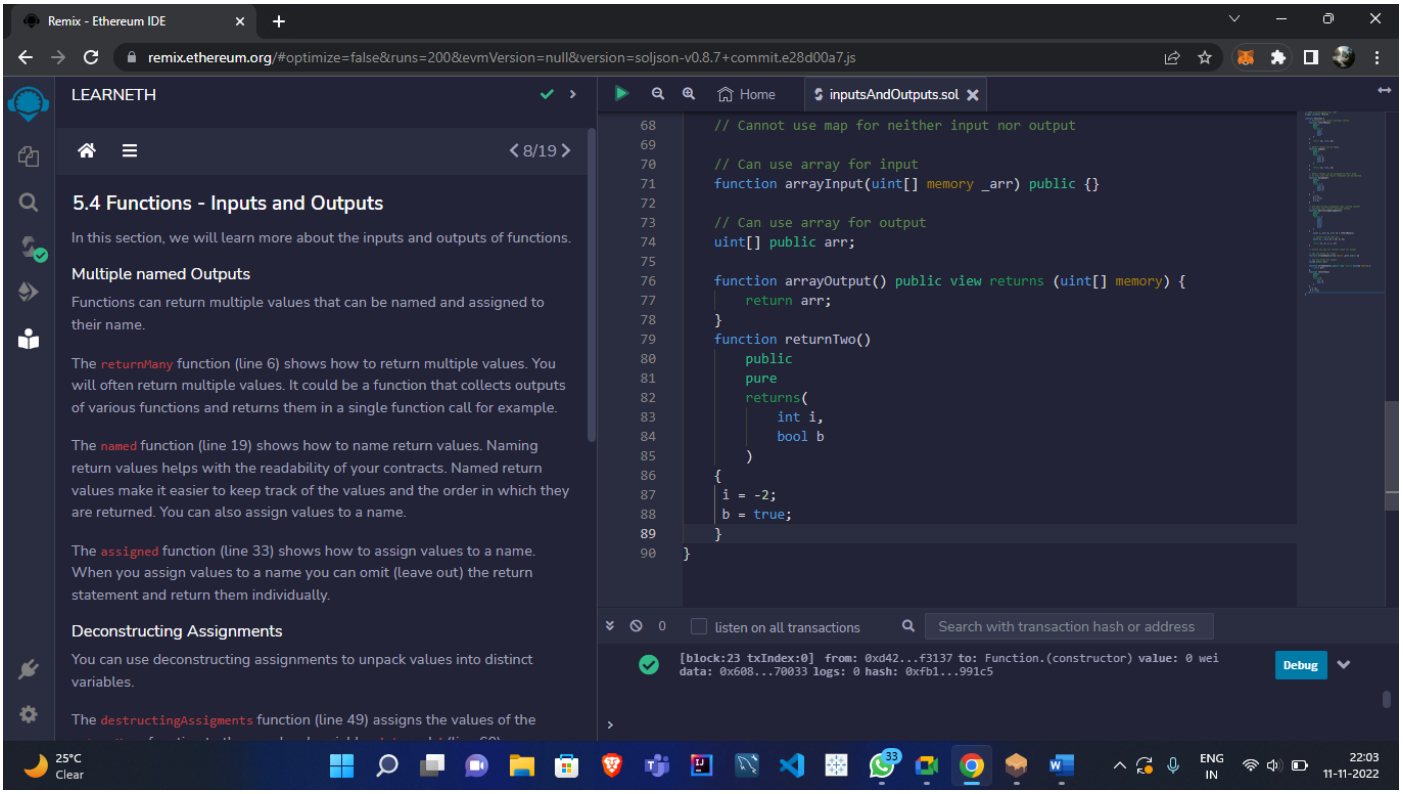




5.4 Functions - Inputs and Outputs :Assignment

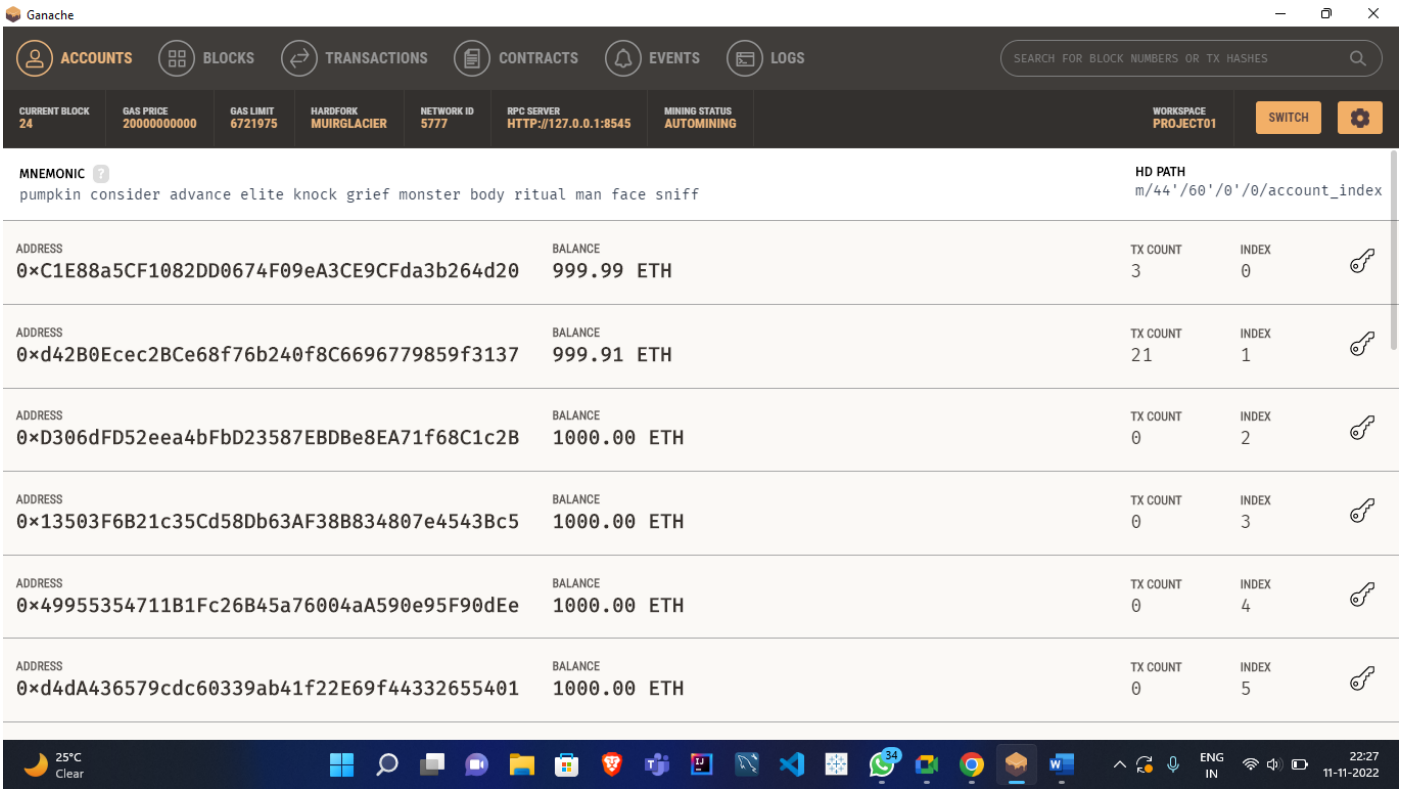
Create a new function called returnTwo that returns the values -2 and true without using a return statement.

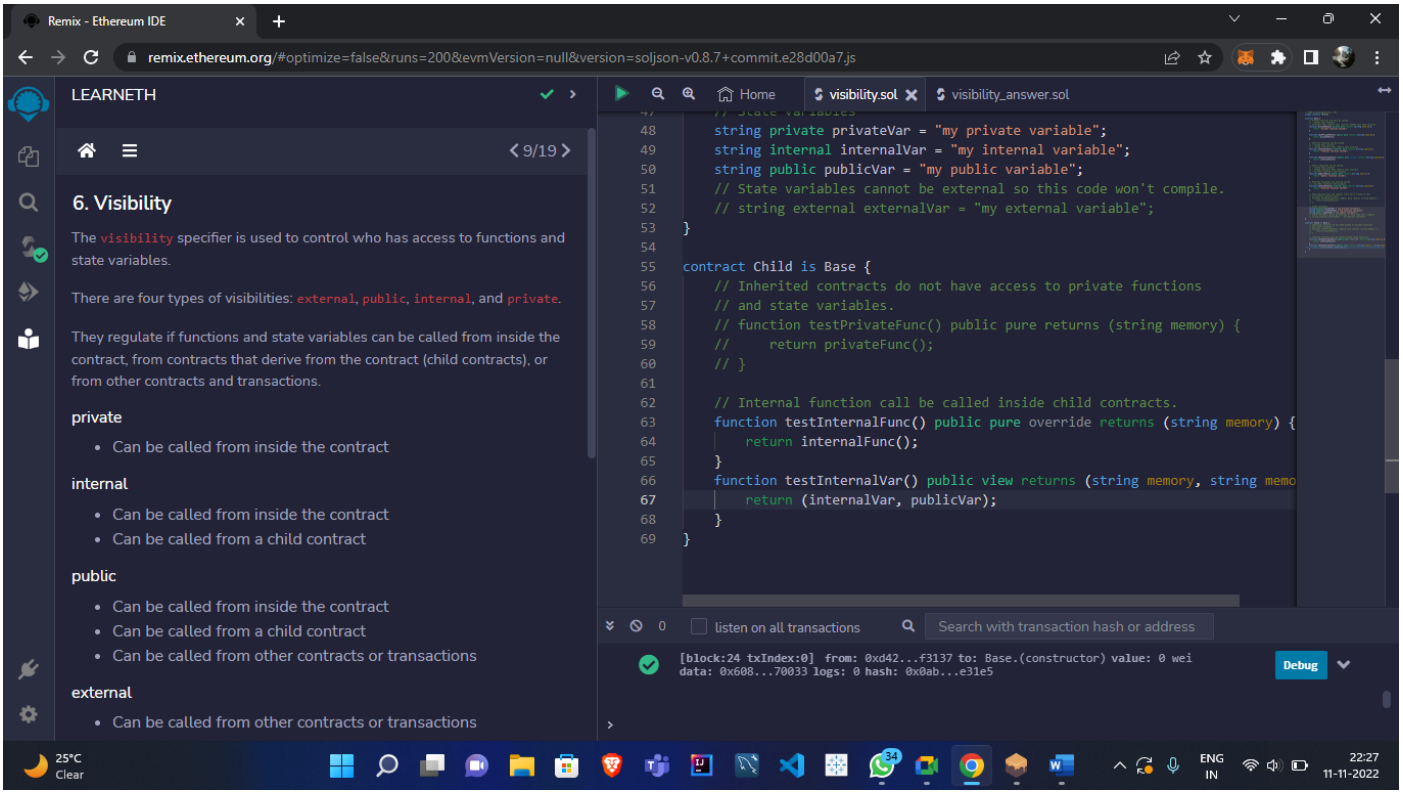




6. Visibility : Assignment

Create a new function in the Child contract called testInternalVar that returns the values of all state variables from the Base contract that are possible to return.





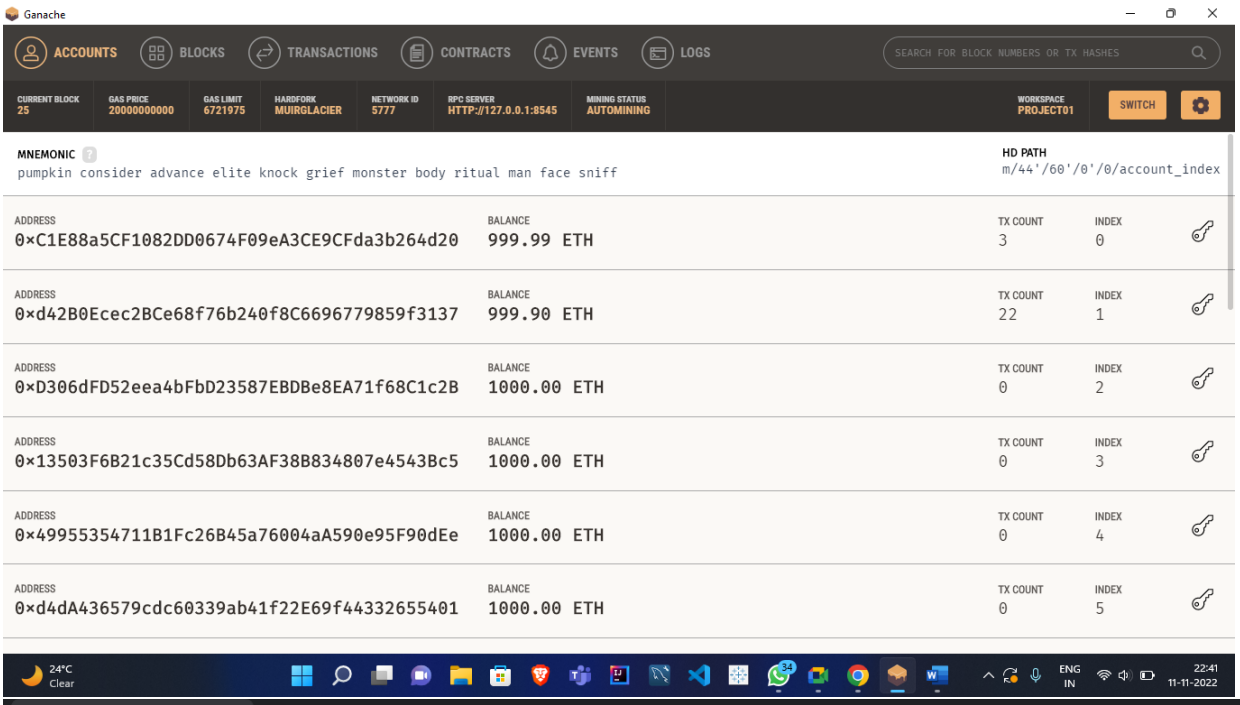
7.1 Control Flow - If/Else: Assignment

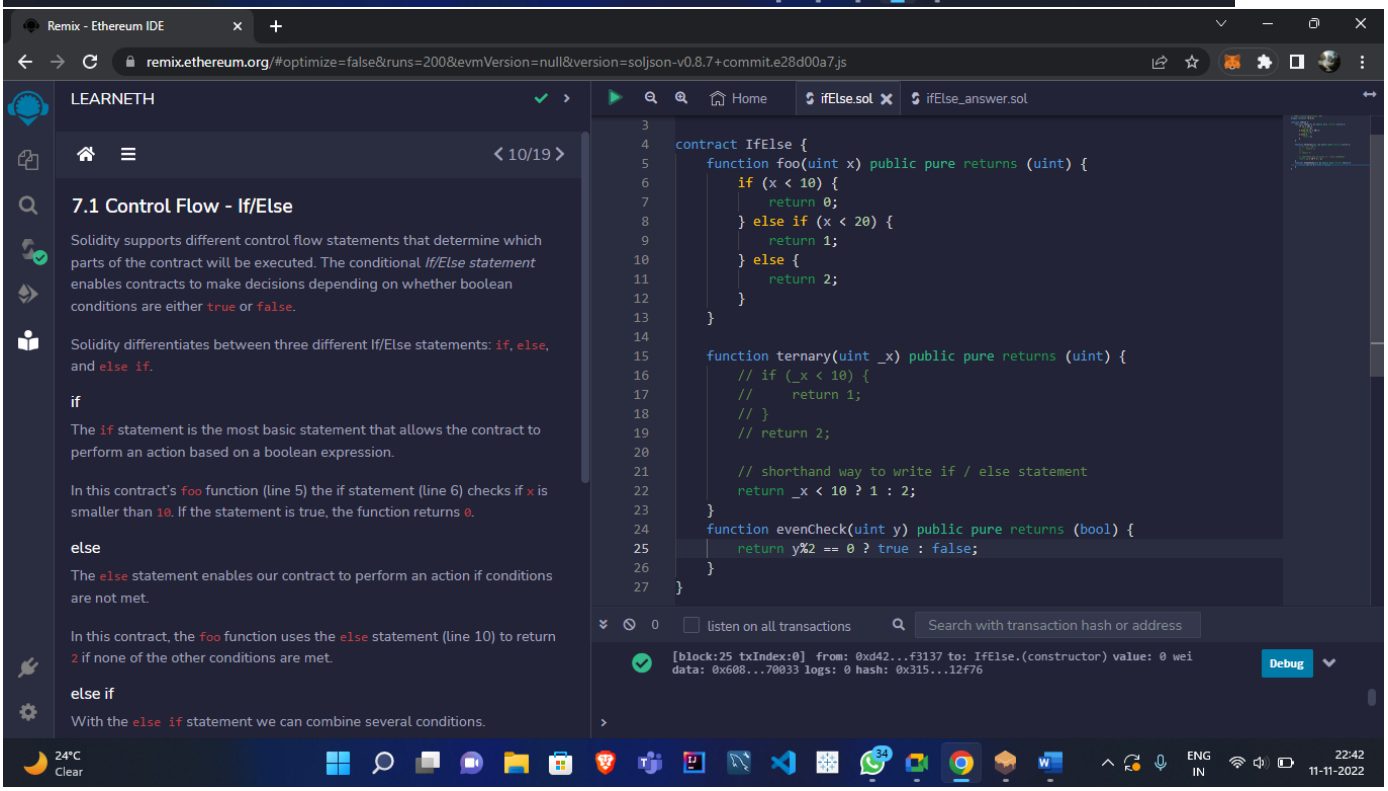
Create a new function called evenCheck in the IfElse contract:

That takes in a uint as an argument.

The function returns true if the argument is even, and false if the argument is odd.

Use a ternery operator to return the result of the evenCheck function.





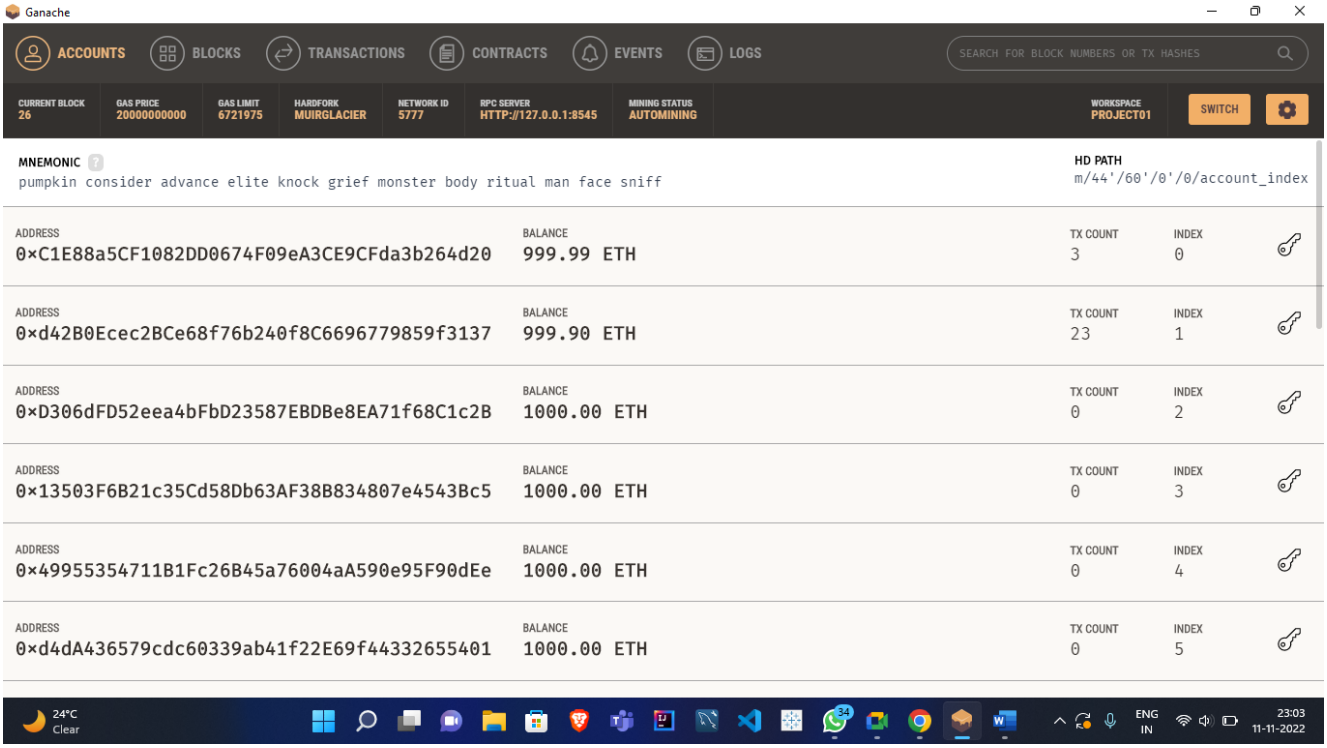
7.2 Control Flow – Loops: Assignment

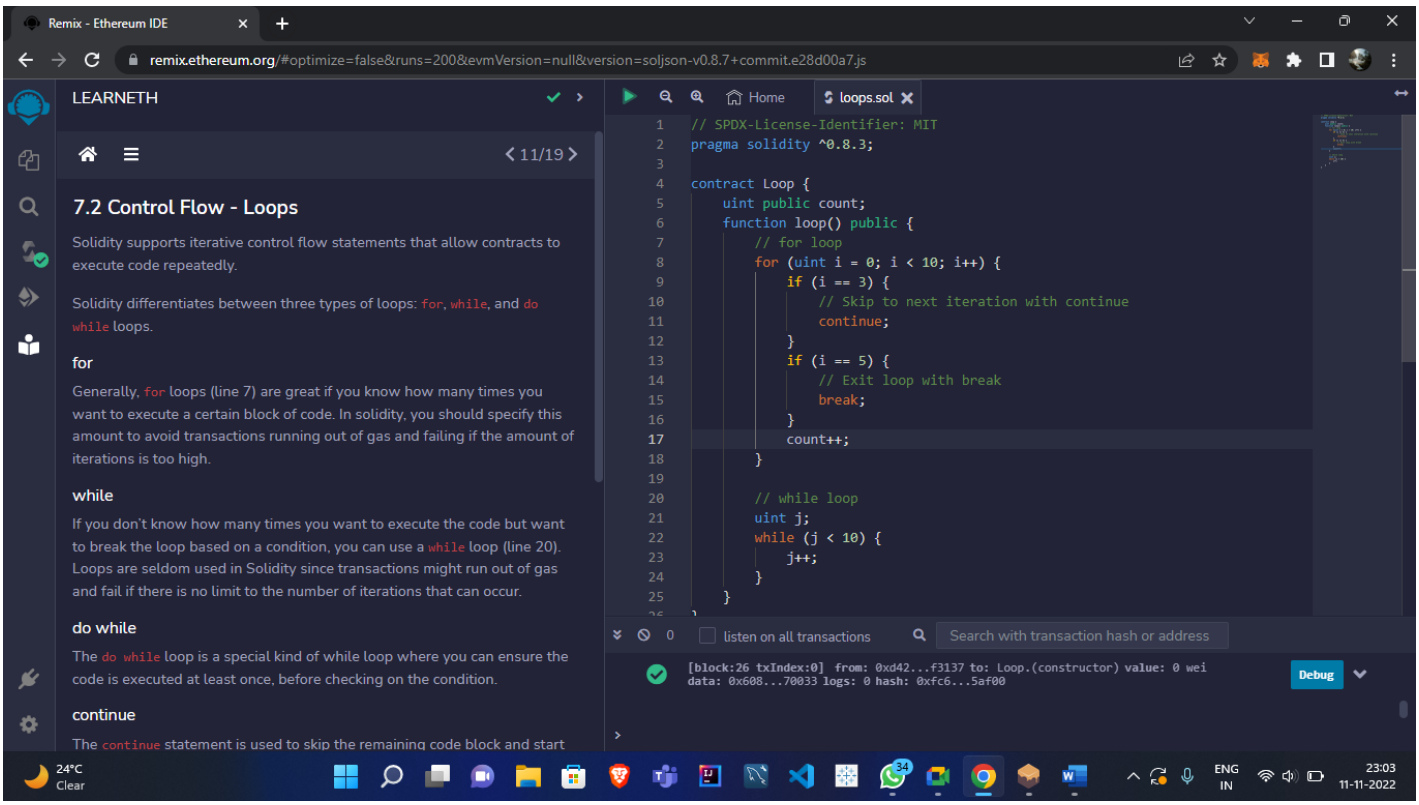
Create a new function called evenCheck in the IfElse contract:

That takes in a uint as an argument.

The function returns true if the argument is even, and false if the argument is odd.

Use a ternery operator to return the result of the evenCheck function.

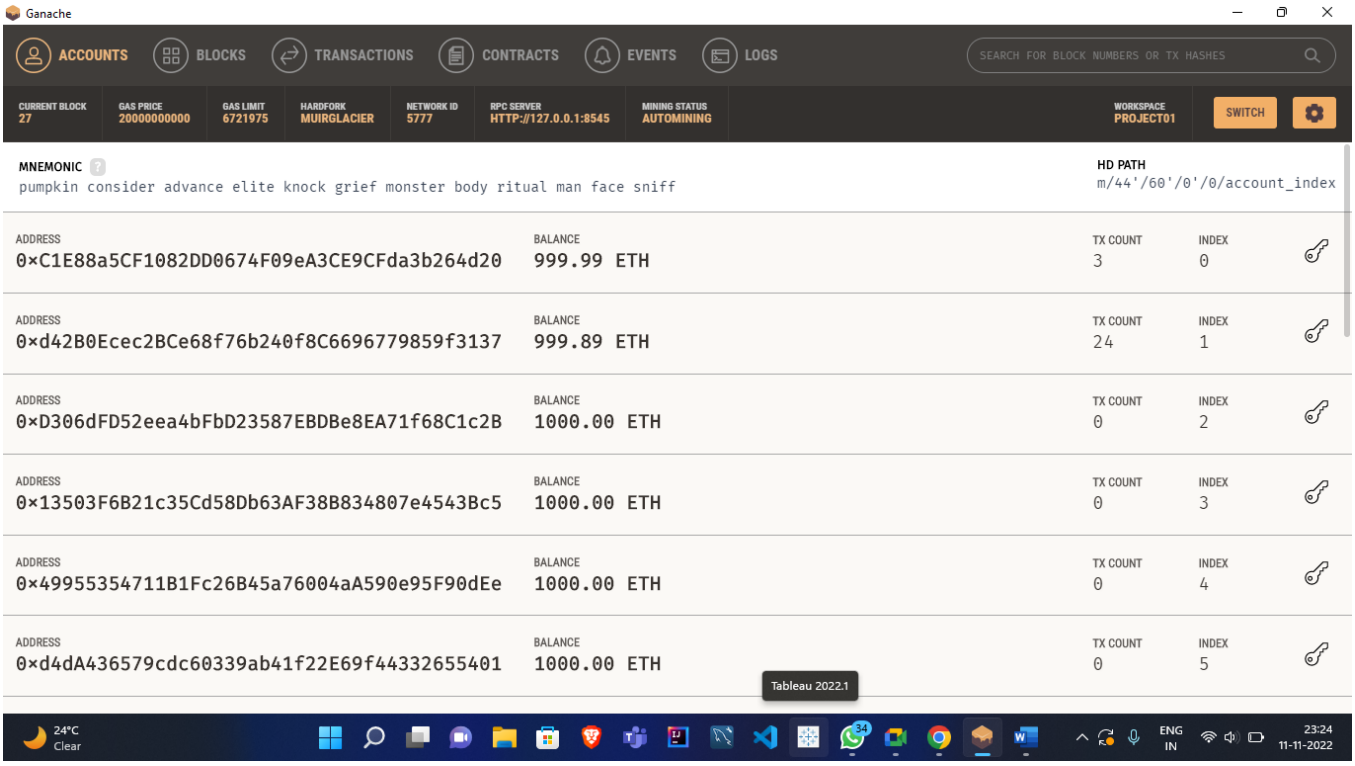


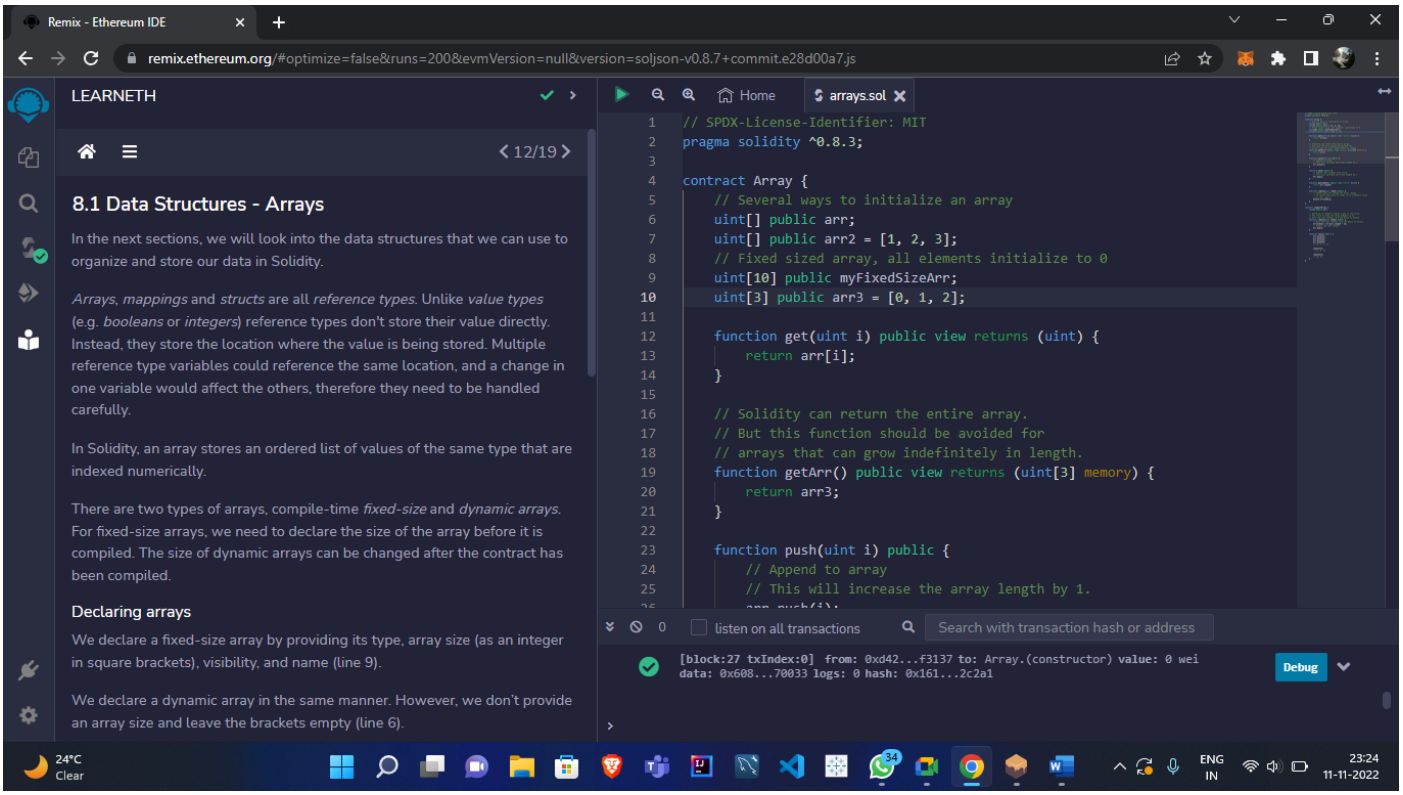


8.1 Data Structures – Arrays: Assignment

Initialize a public fixed-sized array called arr3 with the values 0, 1, 2. Make the size as small as possible.

Change the getArr() function to return the value of arr3.



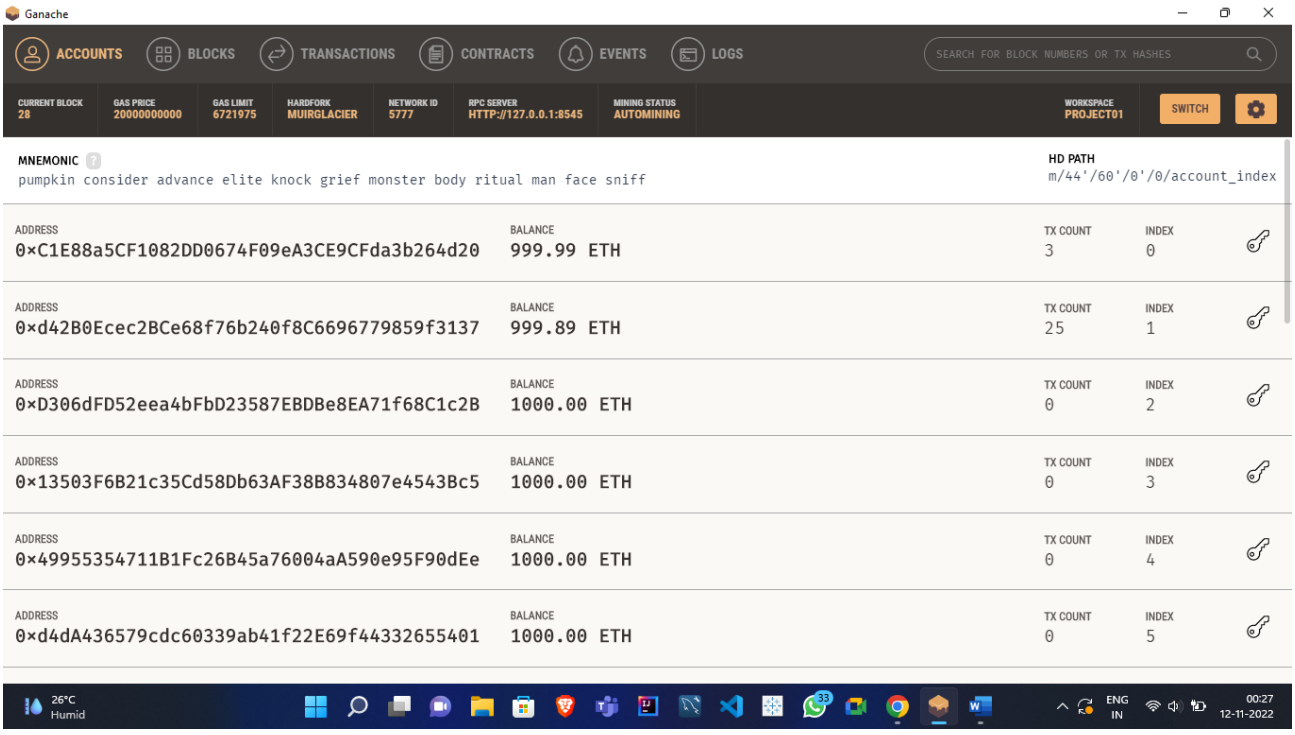


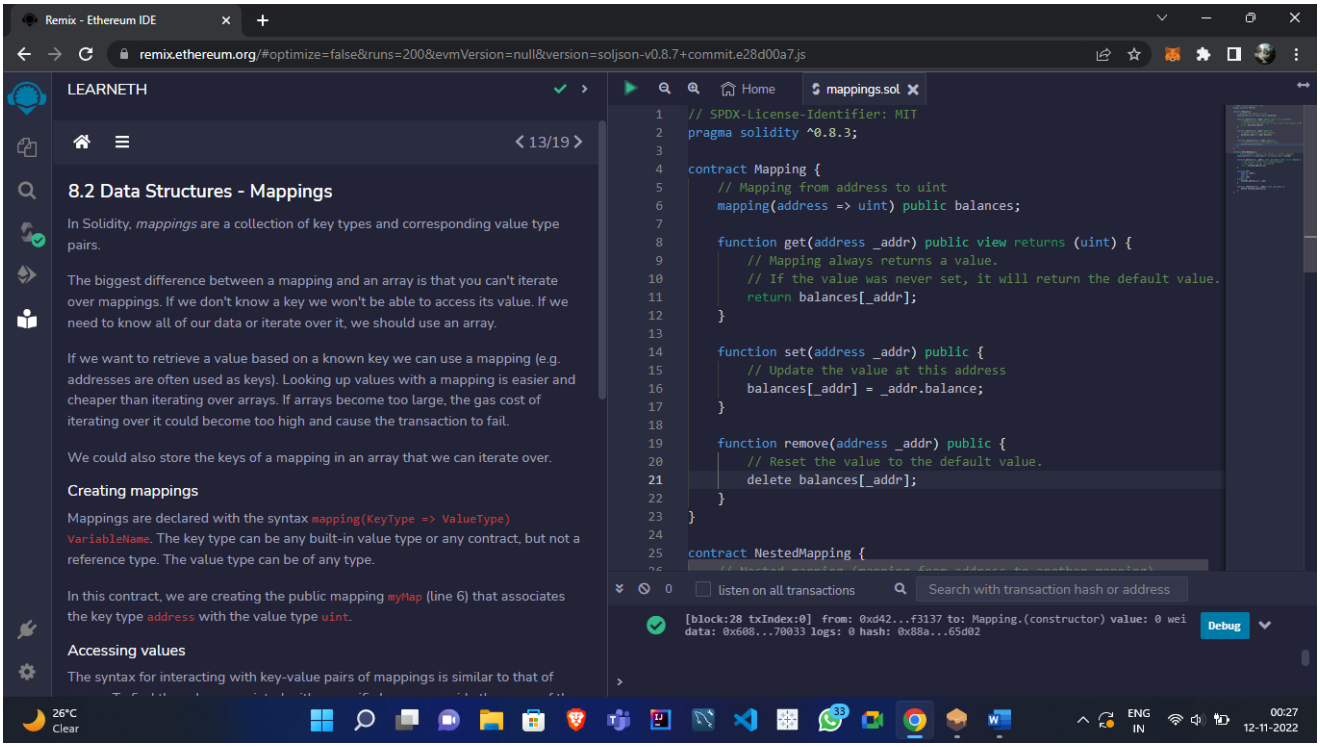
8.2 Data Structures – Mappings: Assignment

Create a public mapping balances that associates the key type address with the value type uint.

Change the functions get and remove to work with the mapping balances.

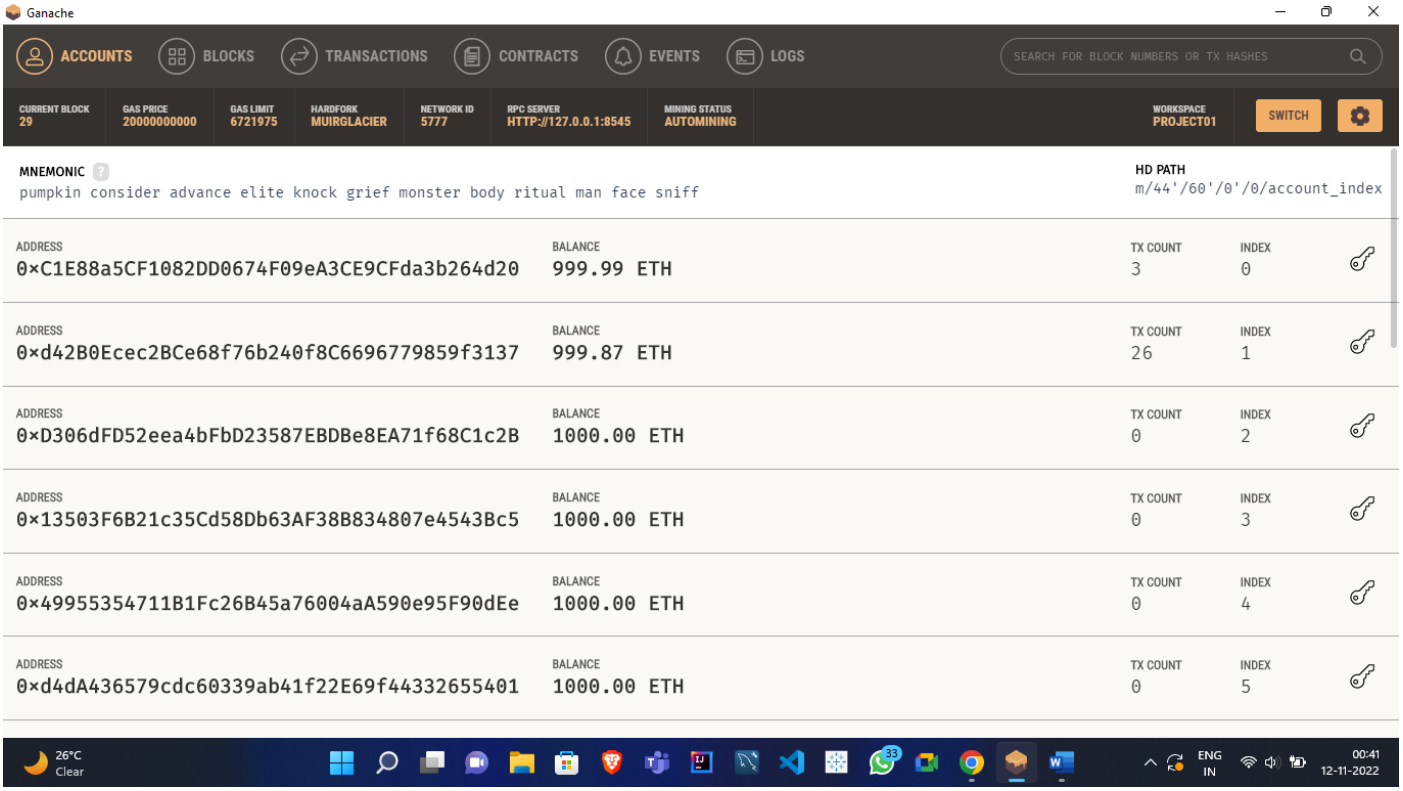
Change the function set to create a new entry to the balances mapping, where the key is the address of the parameter and the value is the balance associated with the address of the parameter.

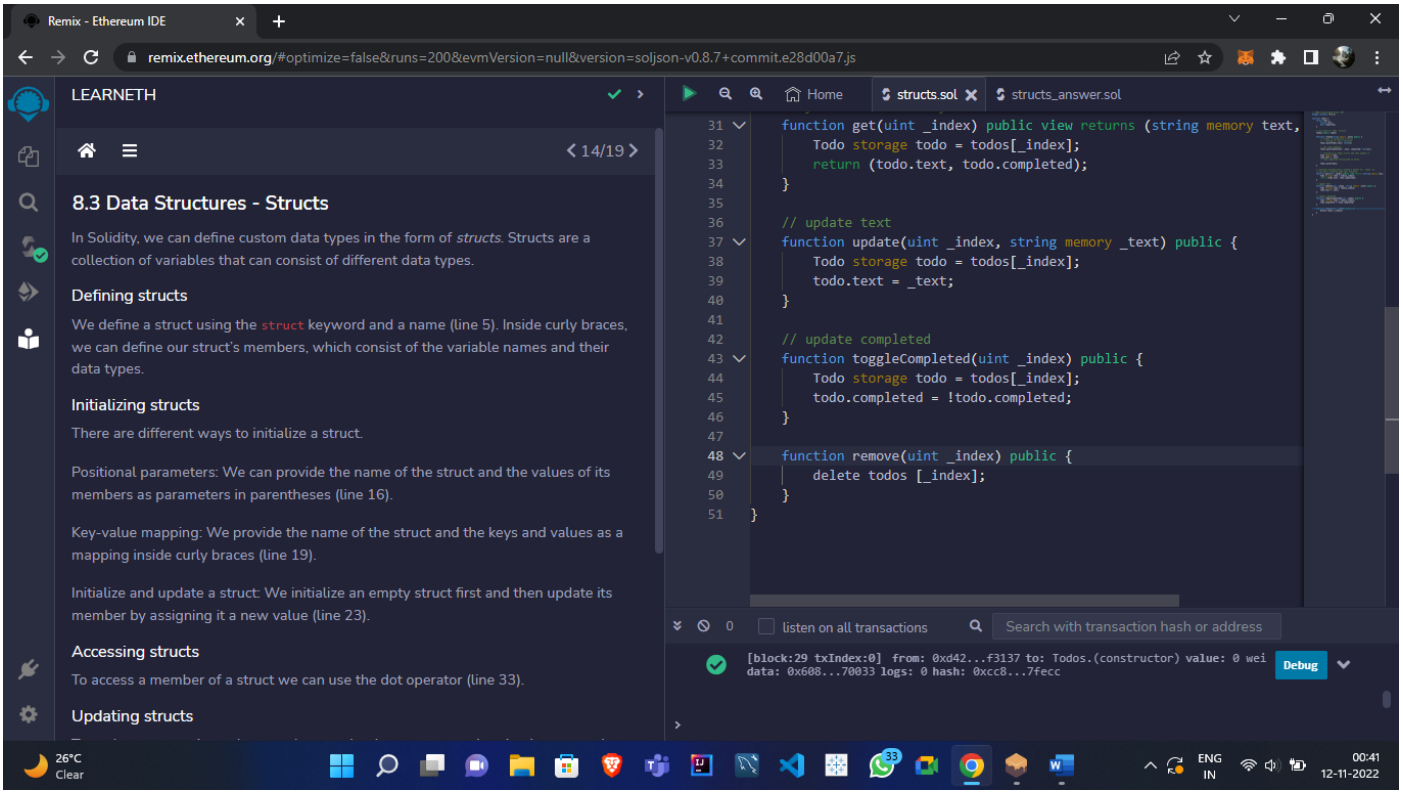




8.3 Data Structures – Structs: Assignment

Create a function remove that takes a uint as a parameter and deletes a struct member with the given index in the todos mapping.



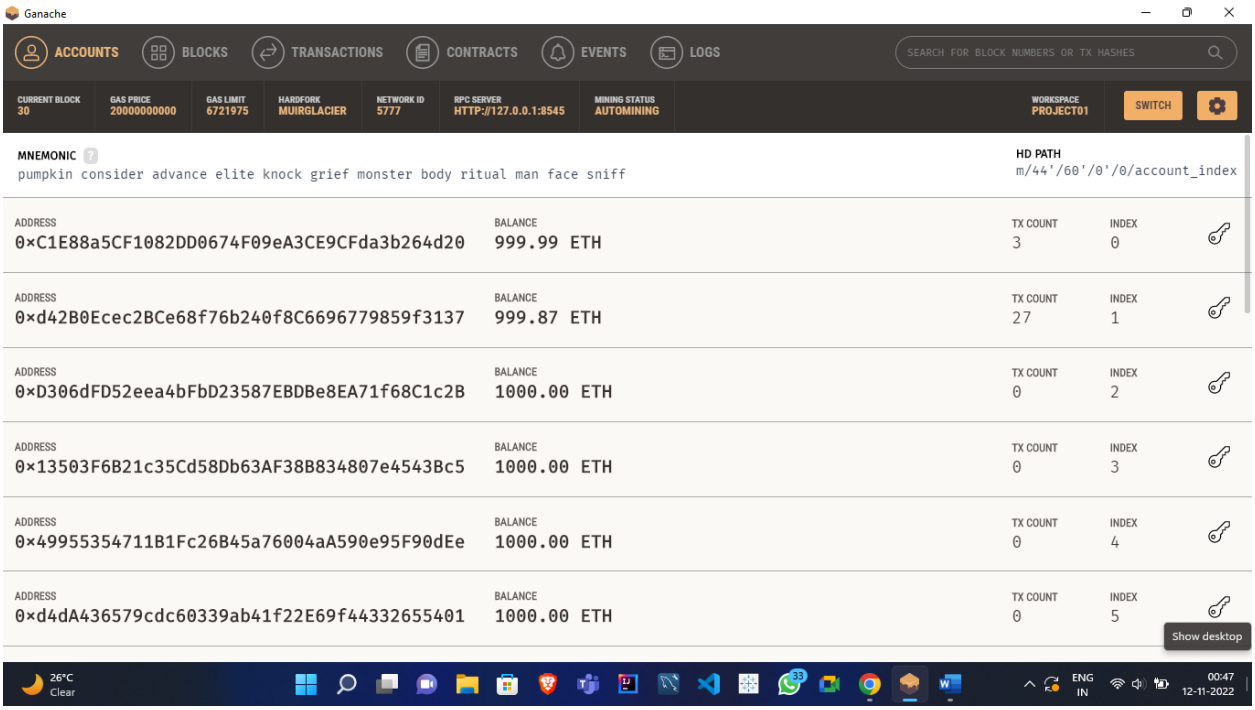


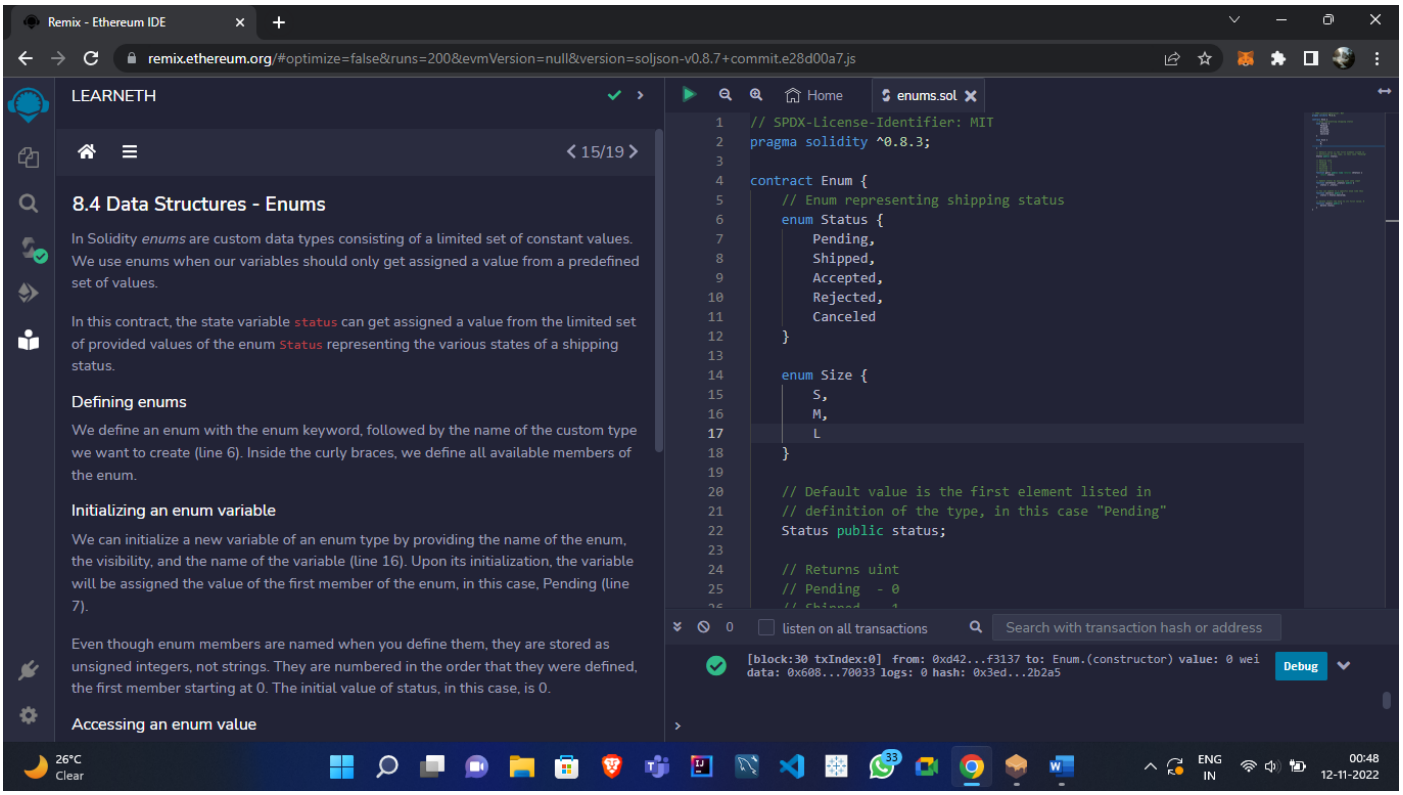
8.4 Data Structures – Enums: Assignment

Define an enum type called Size with the members S, M, and L.

Initialize the variable sizes of the enum type Size.

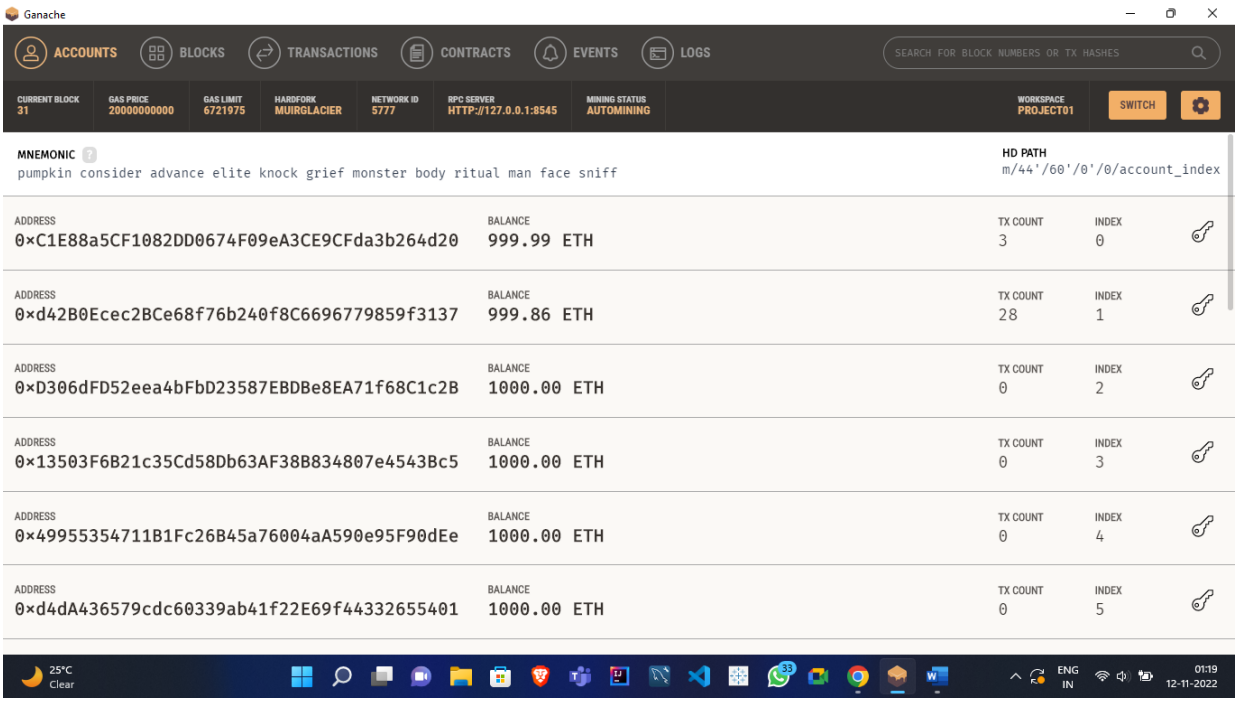
Create a getter function getSize() that returns the value of the variable sizes.

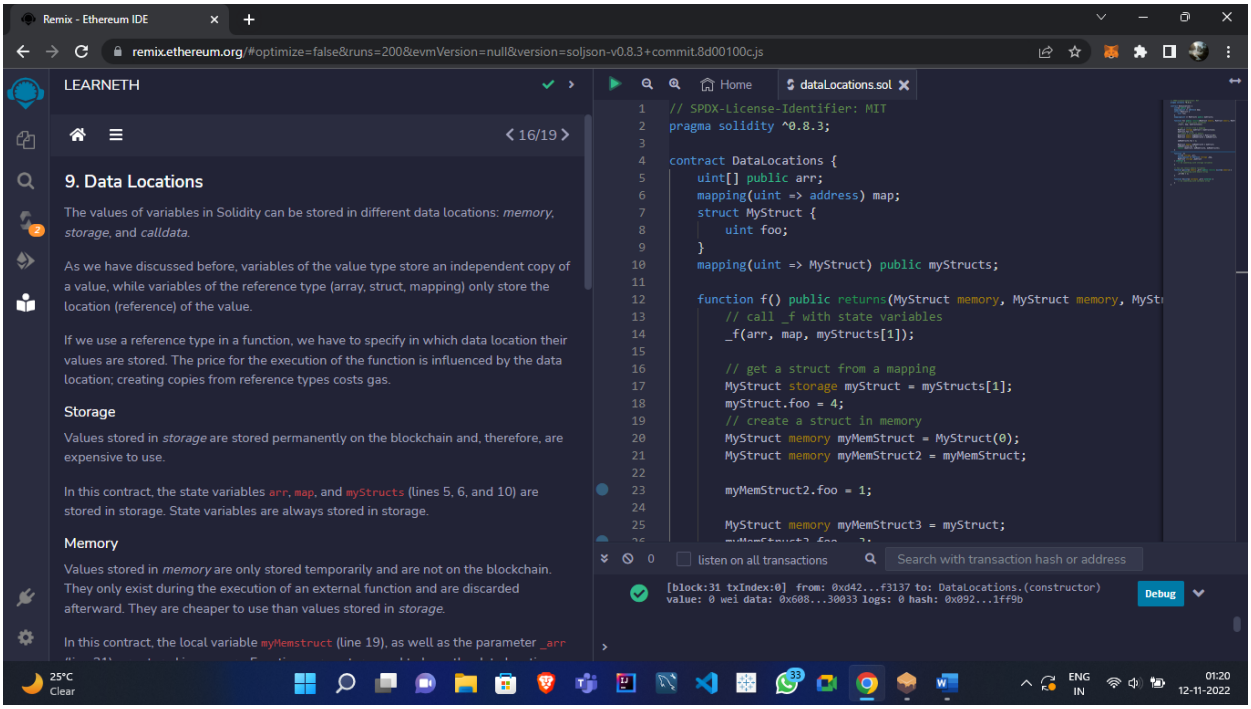




9. Data Locations: Assignment

1)Change the value of the myStruct member foo, inside the function f, to 4. 2)Create a new struct myMemStruct2 with the data location memory inside the function f and assign it the value of myMemStruct. 3)Change the value of the myMemStruct2 member foo to 1. Create a new struct myMemStruct3 with the data location memory inside the function f and assign it the value of myStruct. Change the value of the myMemStruct3 member foo to 3. 4) Let the function f return myStruct, myMemStruct2, and myMemStruct3.

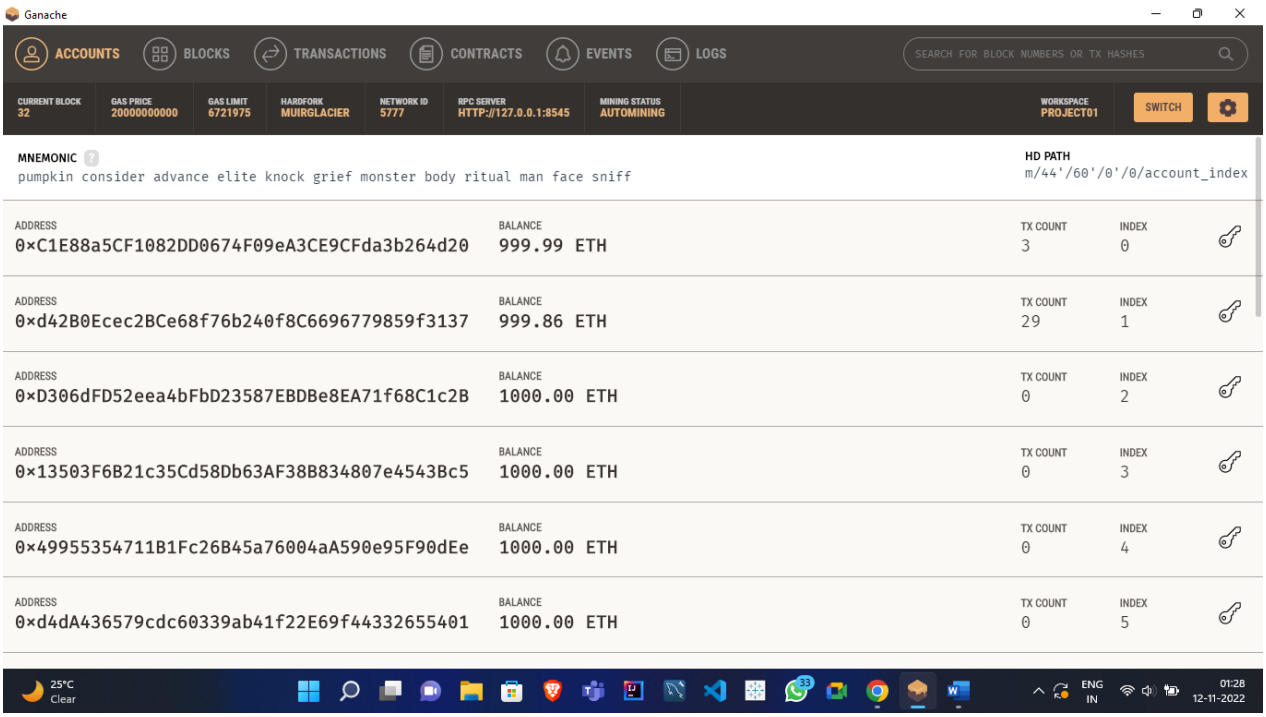


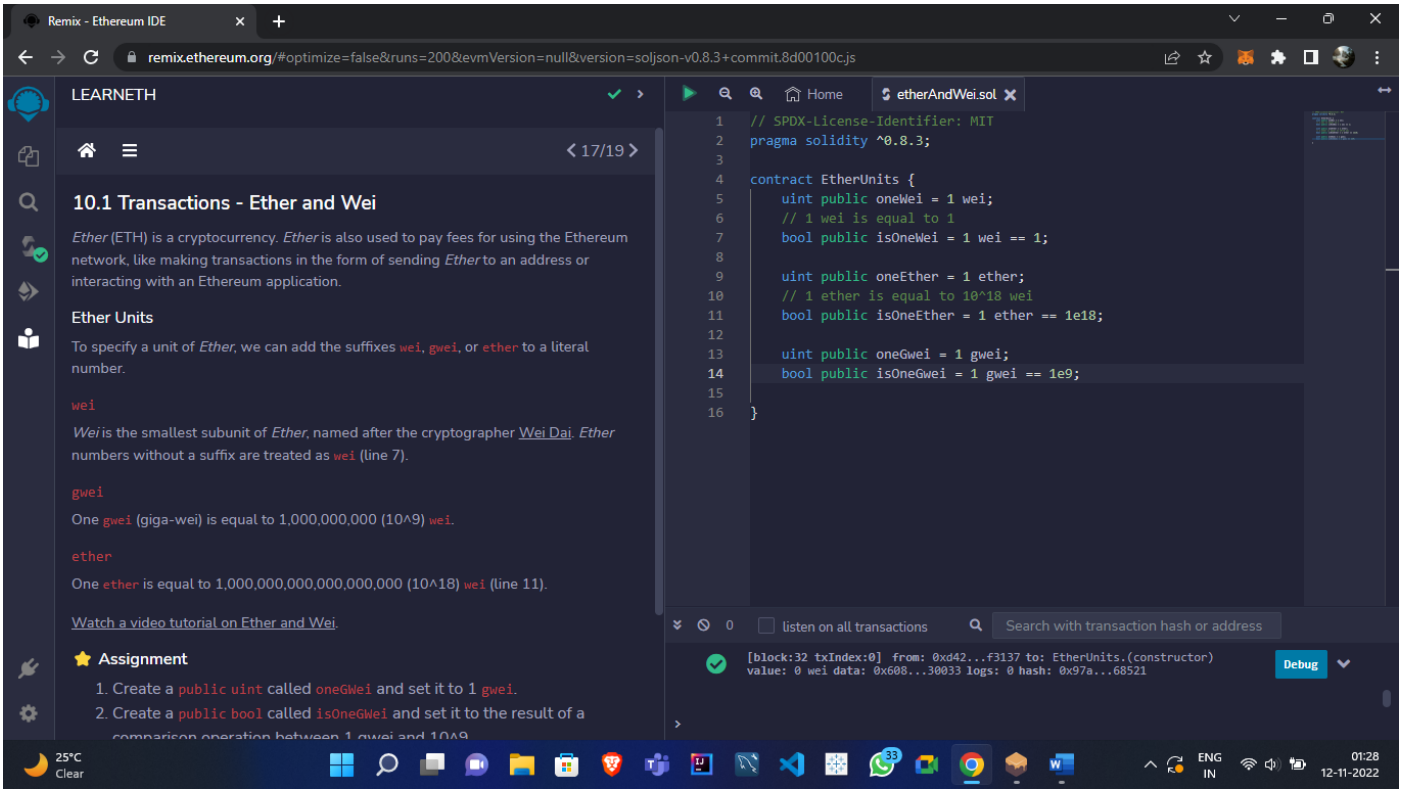


10.1 Transactions - Ether and Wei:

Create a public uint called oneGWei and set it to 1 gwei.

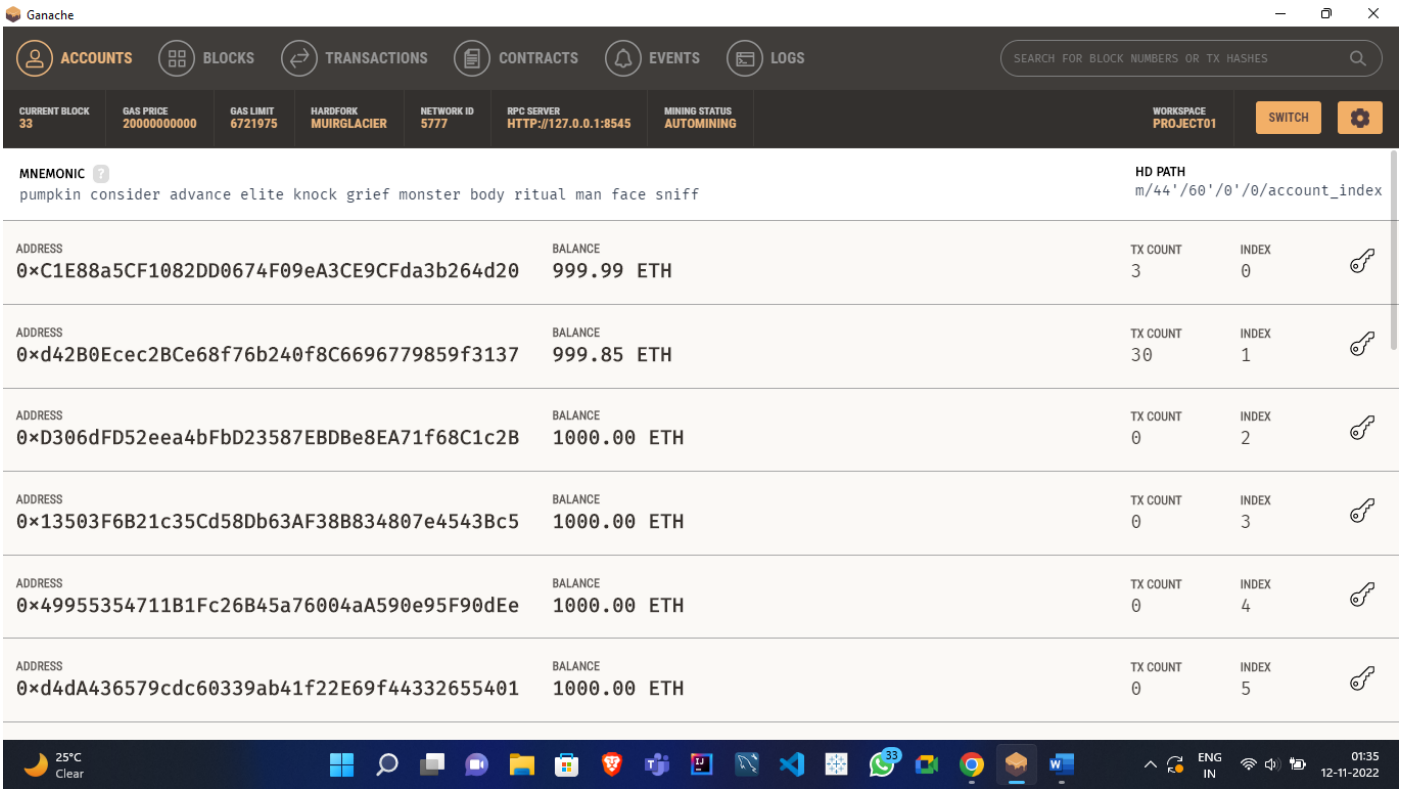
Create a public bool called isOneGWei and set it to the result of a comparison operation between 1 gwei and 10^9.

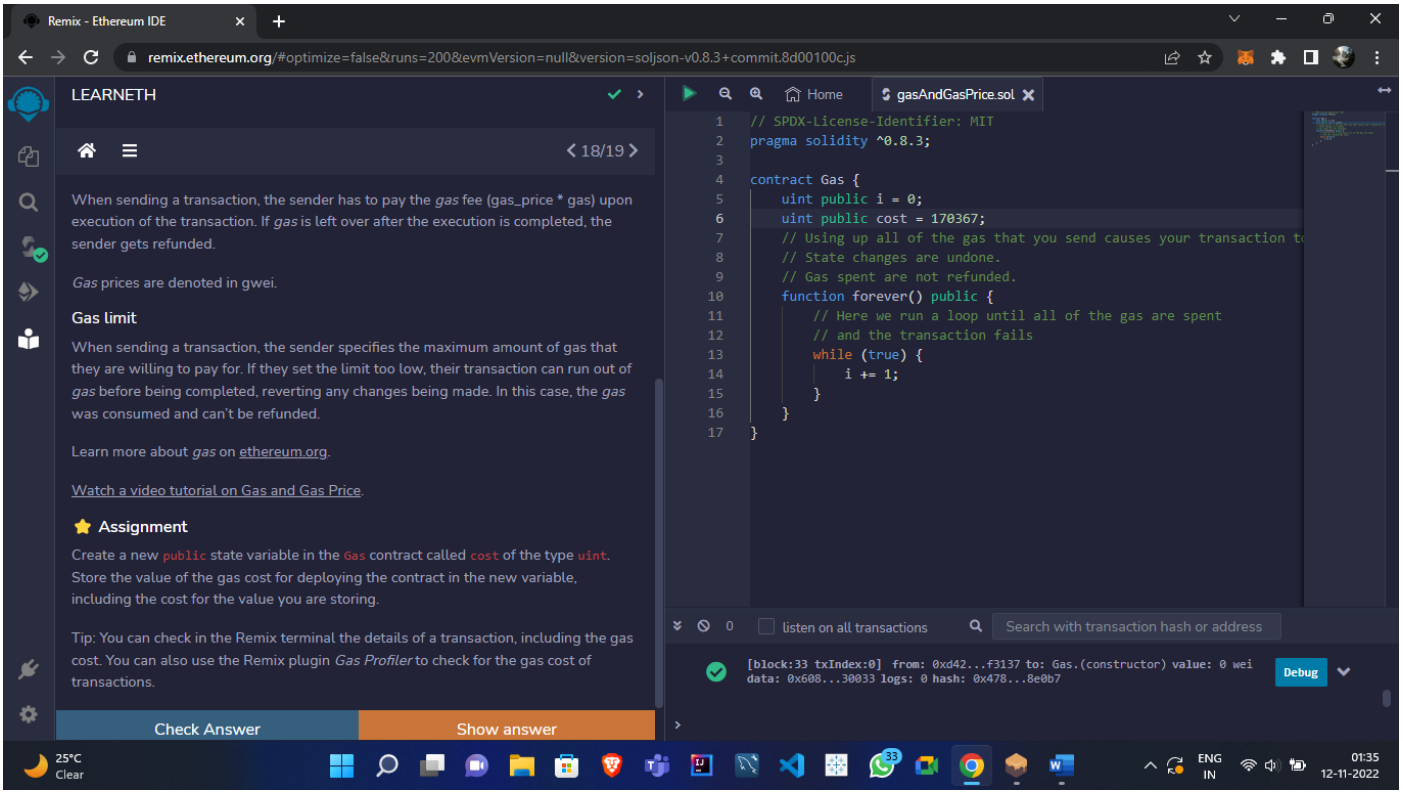




10.2 Transactions - Gas and Gas Price: Assignment

Create a new public state variable in the Gas contract called cost of the type uint. Store the value of the gas cost for deploying the contract in the new variable, including the cost for the value you are storing.





10.3 Transactions - Sending Ether: Assignment

Build a charity contract that receives Ether that can be withdrawn by a beneficiary. Create a contract called Charity.

Add a public state variable called owner of the type address.

Create a donate function that is public and payable without any parameters or function code.

Create a withdraw function that is public and sends the total balance of the contract to the owner address.