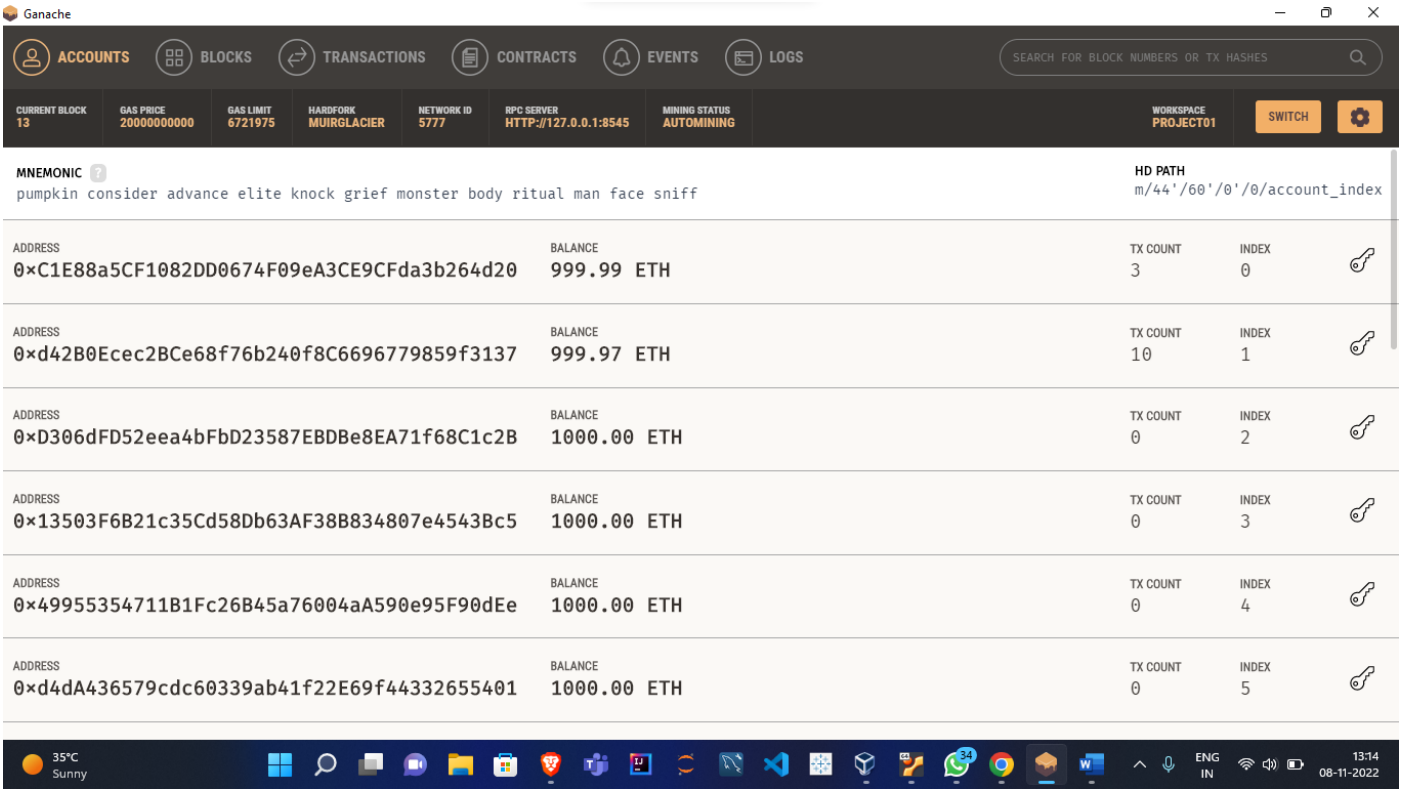
Solidity Beginner Course

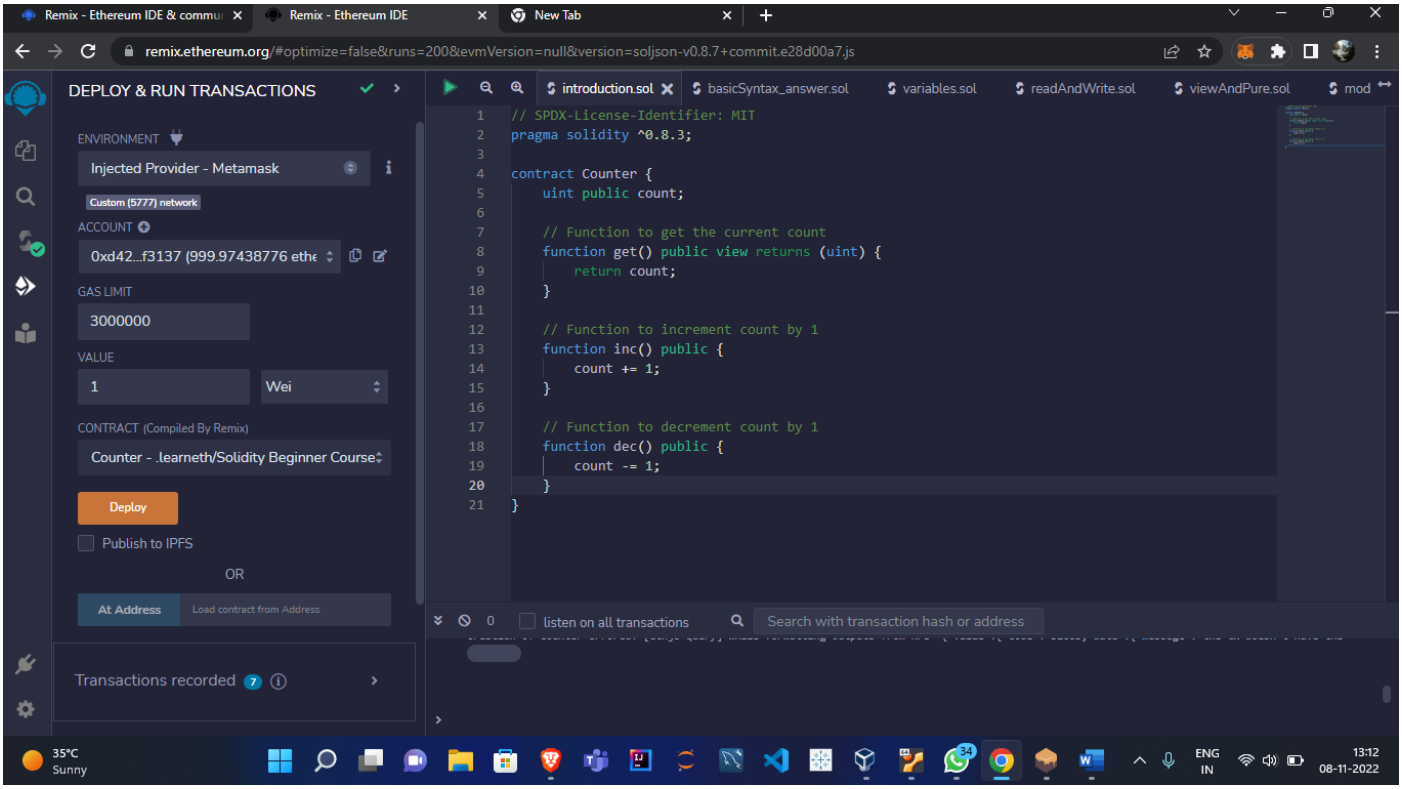
Q) COMPLETE SOLIDITY BEGINNER COURSE FROM LEARNETH FROM REMIX IDE. SOLIDITY BEGINNER COURSE SOLUTION CONTRACT SHOULD BE ADDED IN THE FOLLOWING ORDER.

- CHAPTER NAME
- Assignment
- Details of the Assignment
- SOLUTION CONTRACT [ENTIRE SMART CONTRACT CODE]
- Deployed Address on Ganache

1. Introduction:

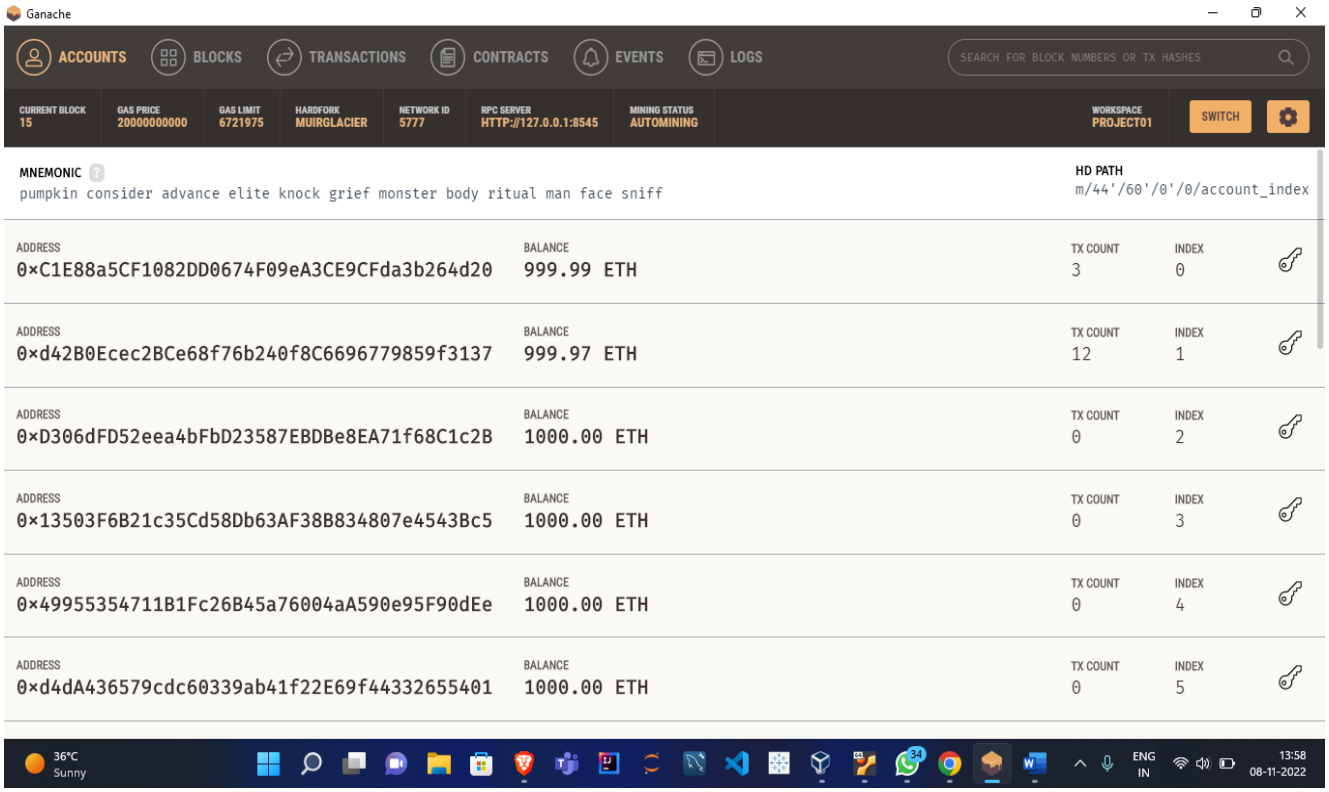
- Compile this contract.
- Deploy it to the JavaScript VM.
- Interact with your contract.

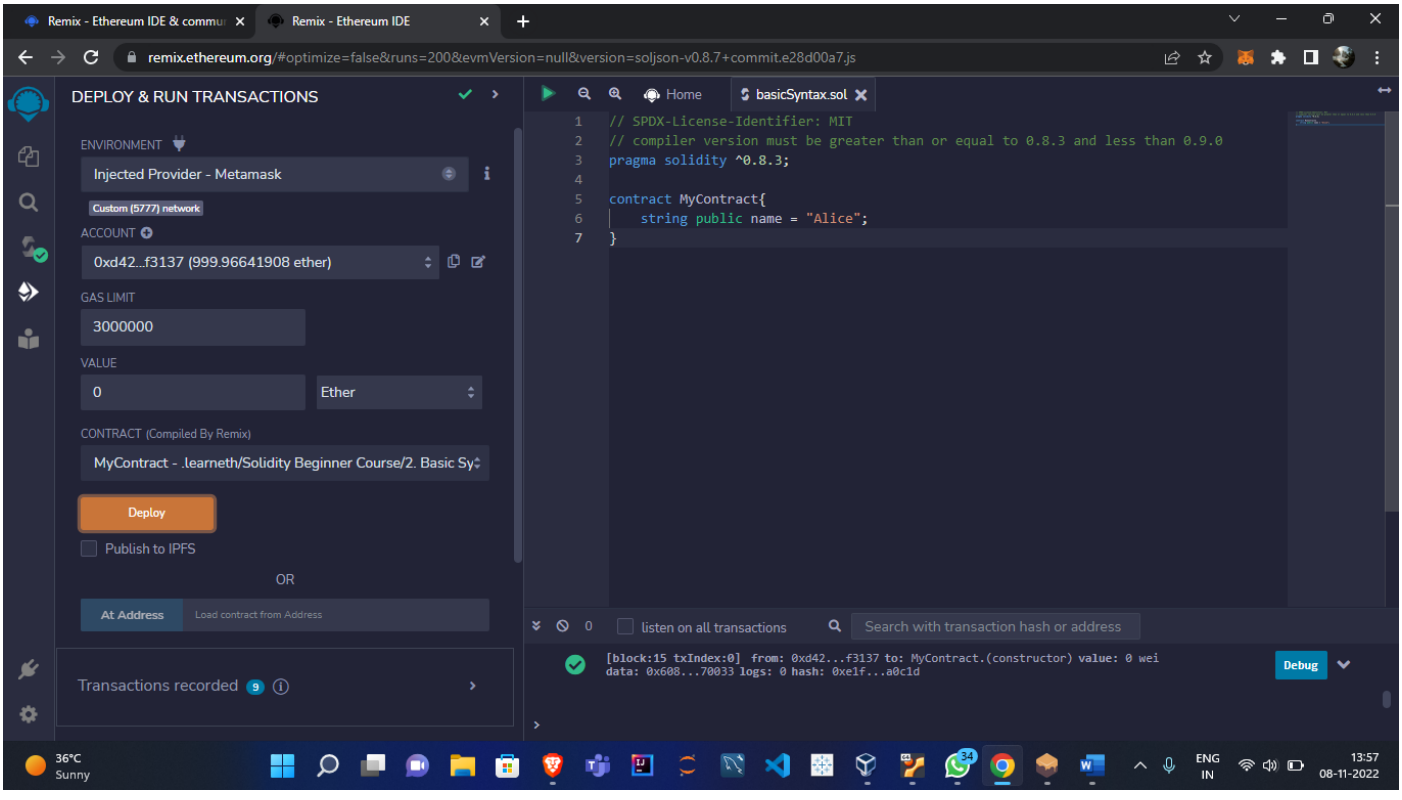



Remix - Ethereum IDE & commu
Remix - Ethereum IDE
New Tab
remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.7+commit.e28d00a7.js
DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Injected Provider - Metamask
Custom (5777) network
ACCOUNT
0xd42...f3137 (999.97438776 ethe
GAS LIMIT
3000000
VALUE
1
Wei
CONTRACT (Compiled By Remix)
Counter - .learneth/Solidity Beginner Course
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 7
introduction.sol
basicSyntax_answer.sol
variables.sol
readAndWrite.sol
viewAndPure.sol
mod
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.3;
3
4 contract Counter {
5 uint public count;
6
7 // Function to get the current count
8 function get() public view returns (uint) {
9 return count;
10 }
11
12 // Function to increment count by 1
13 function inc() public {
14 count += 1;
15 }
16
17 // Function to decrement count by 1
18 function dec() public {
19 count -= 1;
20 }
21 }
0
listen on all transactions
Search with transaction hash or address
35°C
Sunny
ENG
IN
13:12
08-11-2022

2. Basic Syntax Assignment :

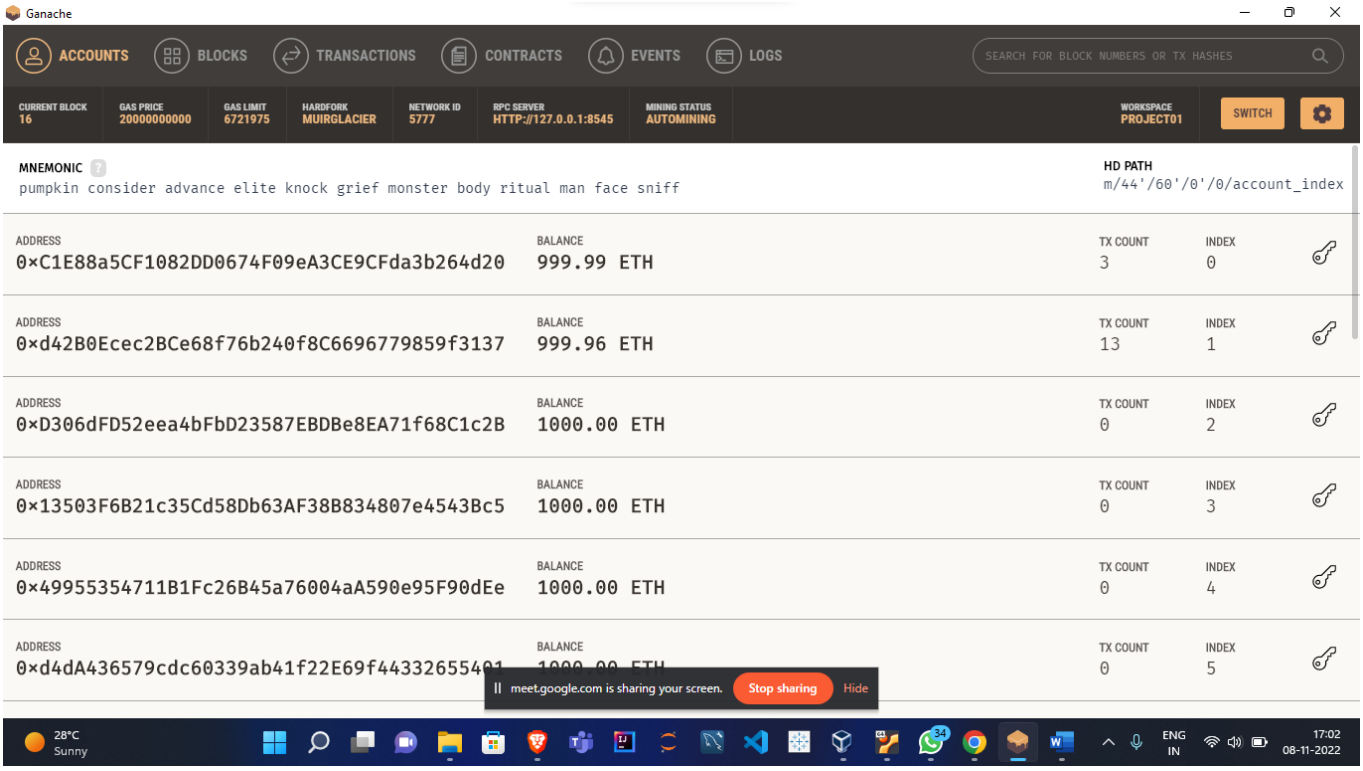
- ➢ **Delete the HelloWorld contract and its content.**
- ➢ **Create a new contract named "MyContract".**
- ➢ **The contract should have a public state variable called "name" of the type string.**
- ➢ **Assign the value "Alice" to your new variable.**

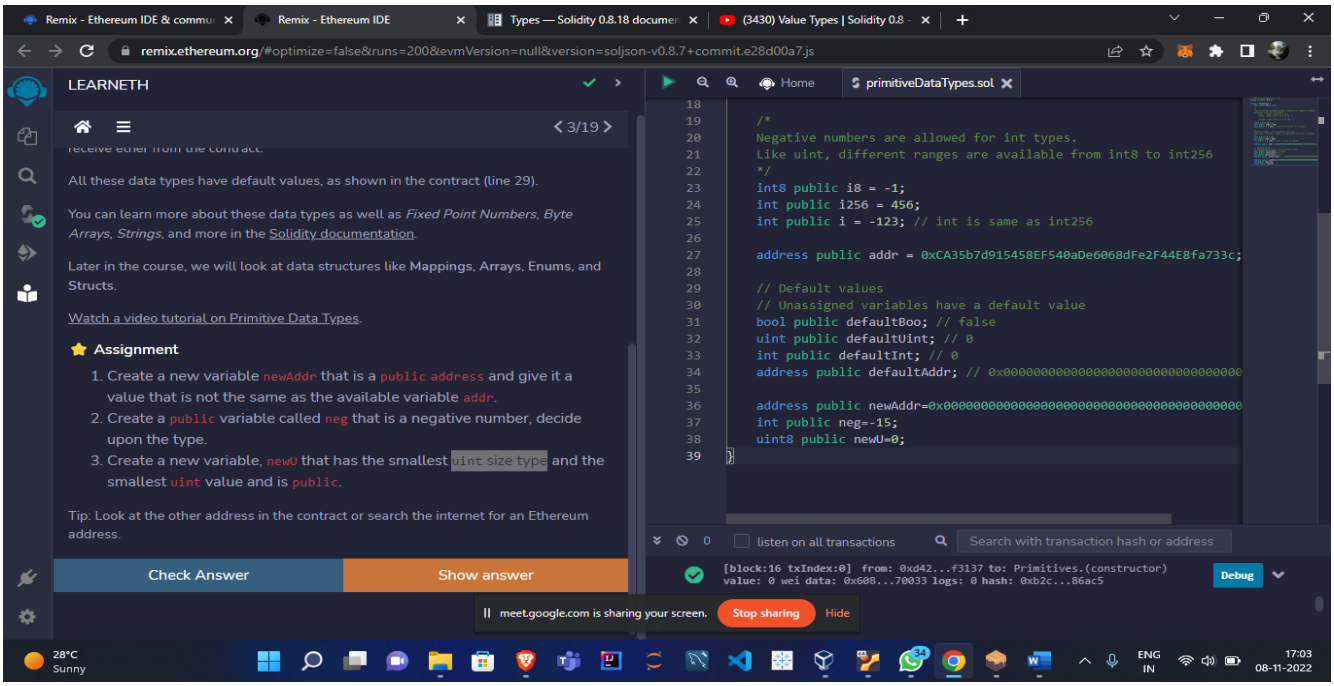




3. Primitive Data Types Assignment:

- **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.**

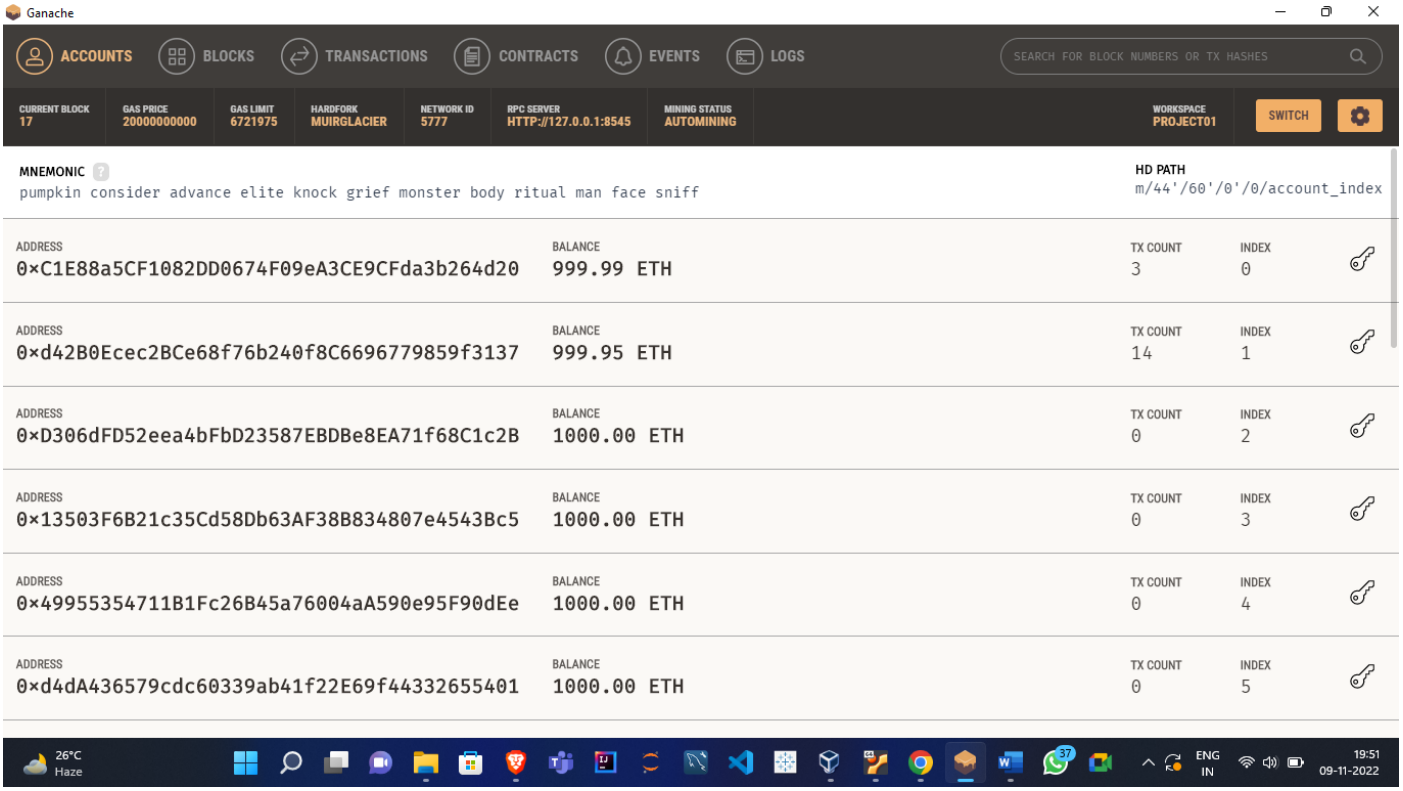


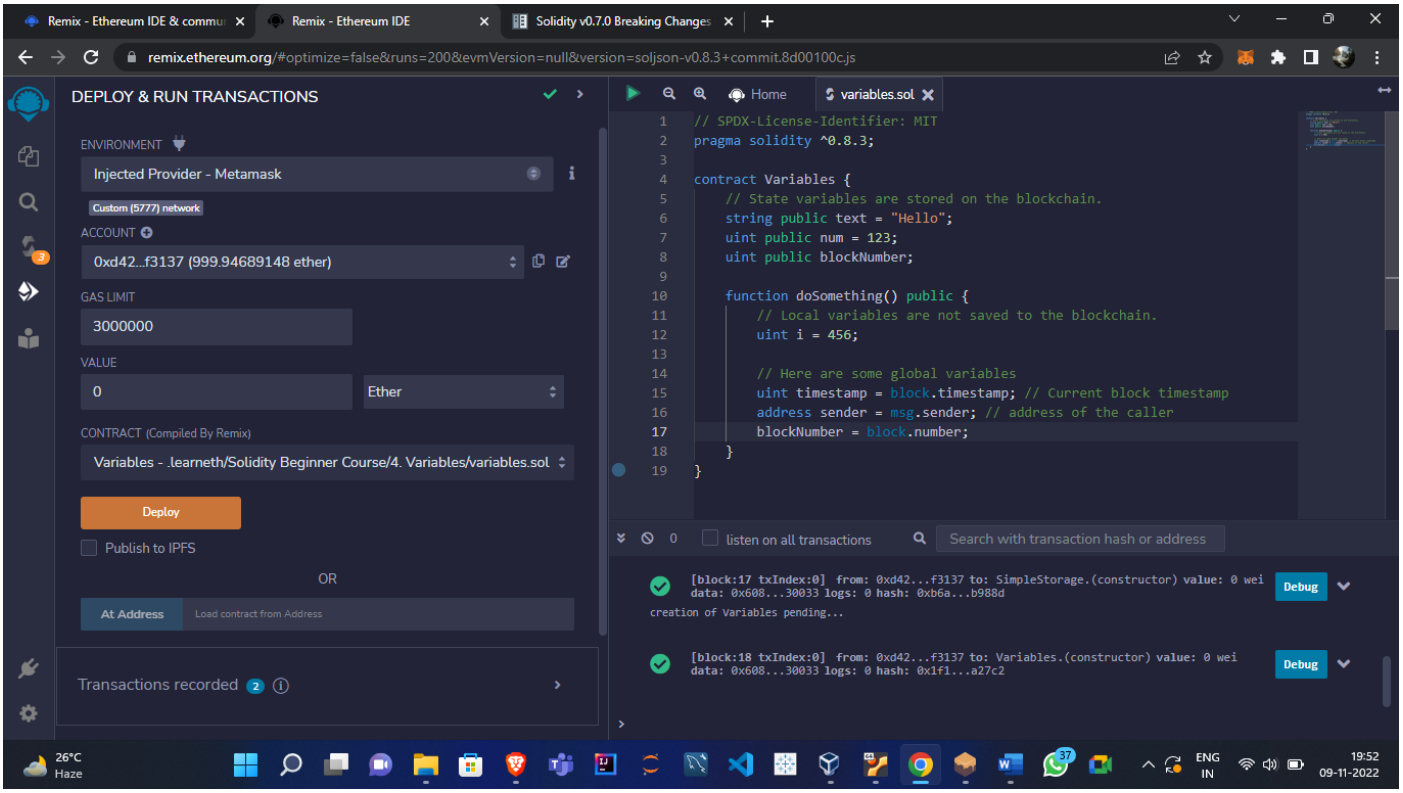


4. Variables : Assignment

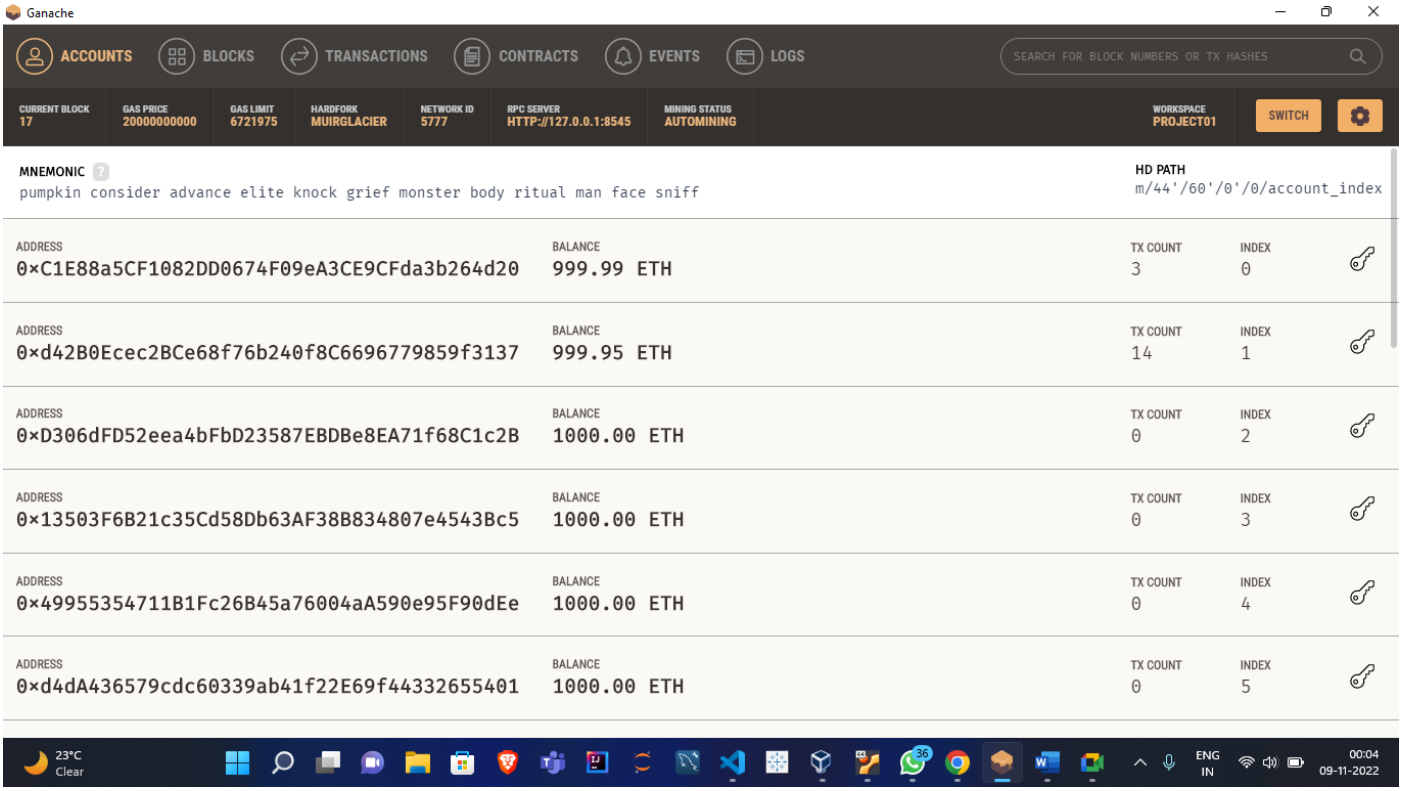
Create a new public state variable called blockNumber.

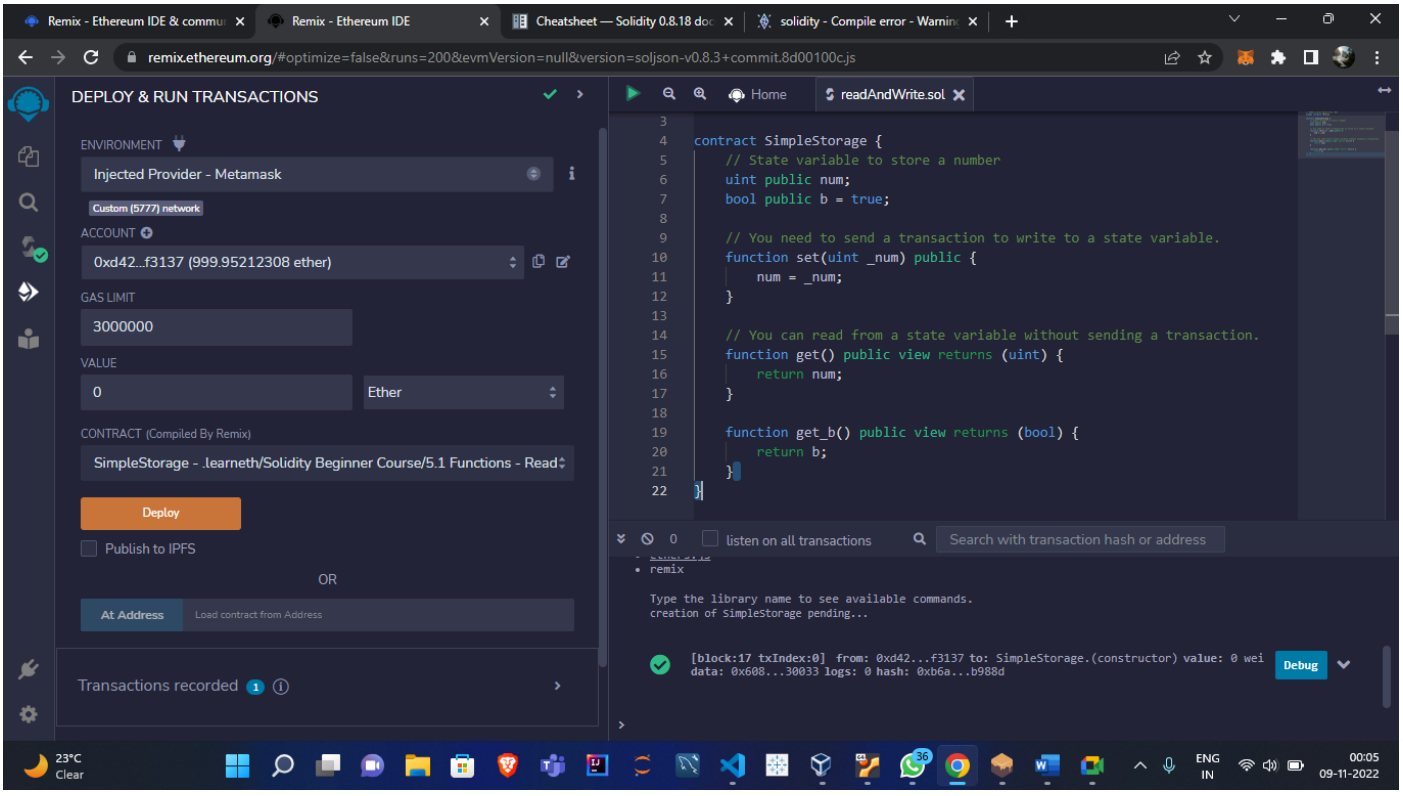
Inside the function doSomething(), assign the value of the current block number to the state variable blockNumber.





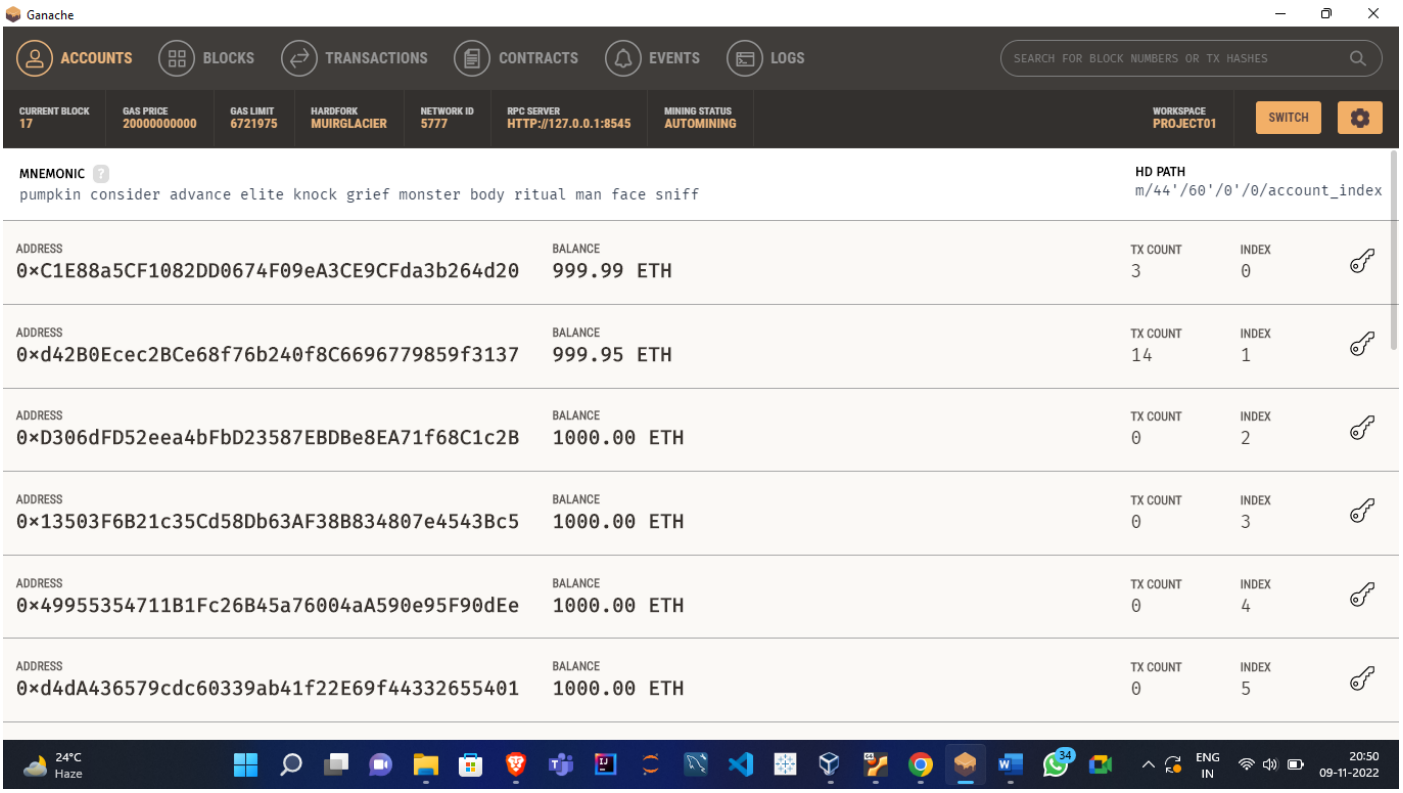
5.1)Functions - Reading and Writing to a State Variable

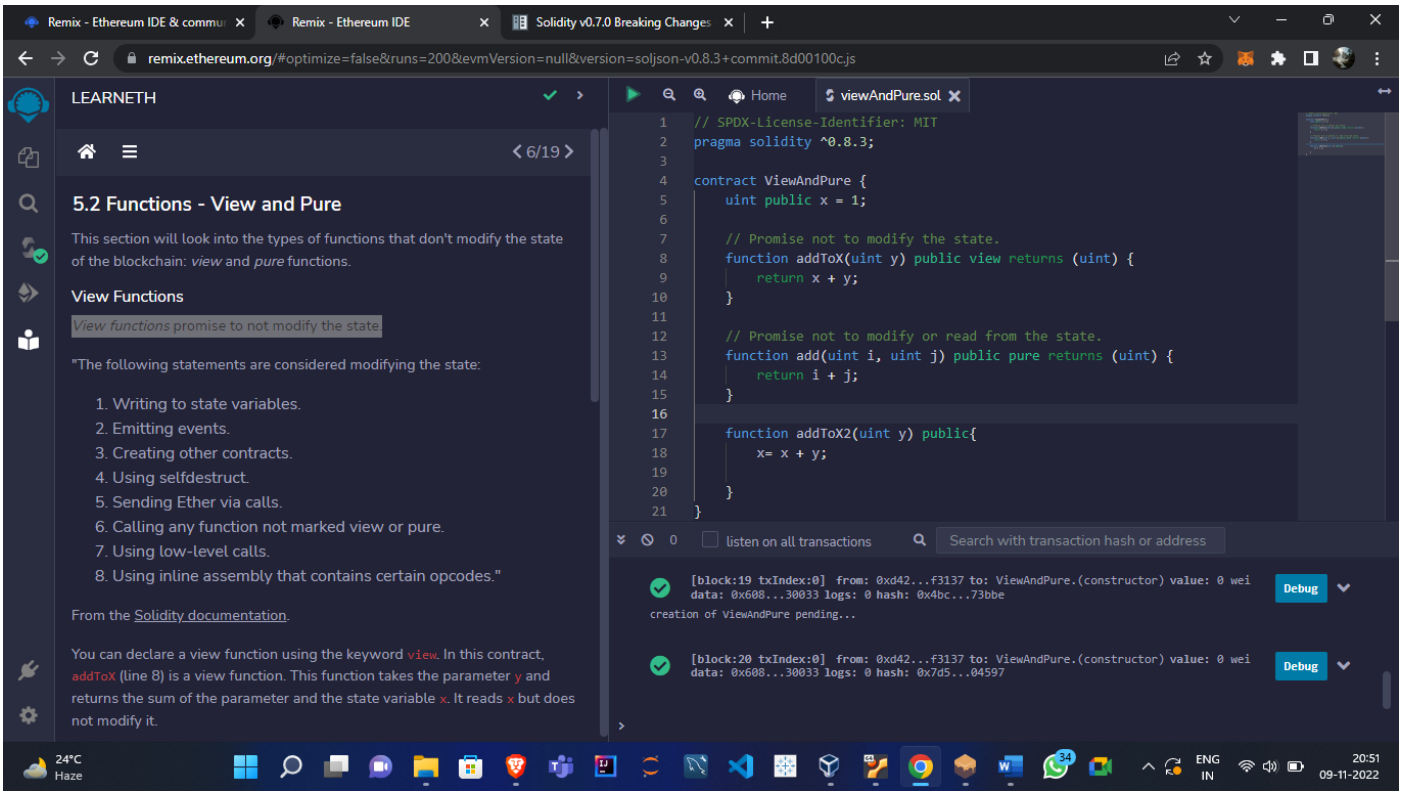




5.2)Functions - View and Pure : Assignment

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.



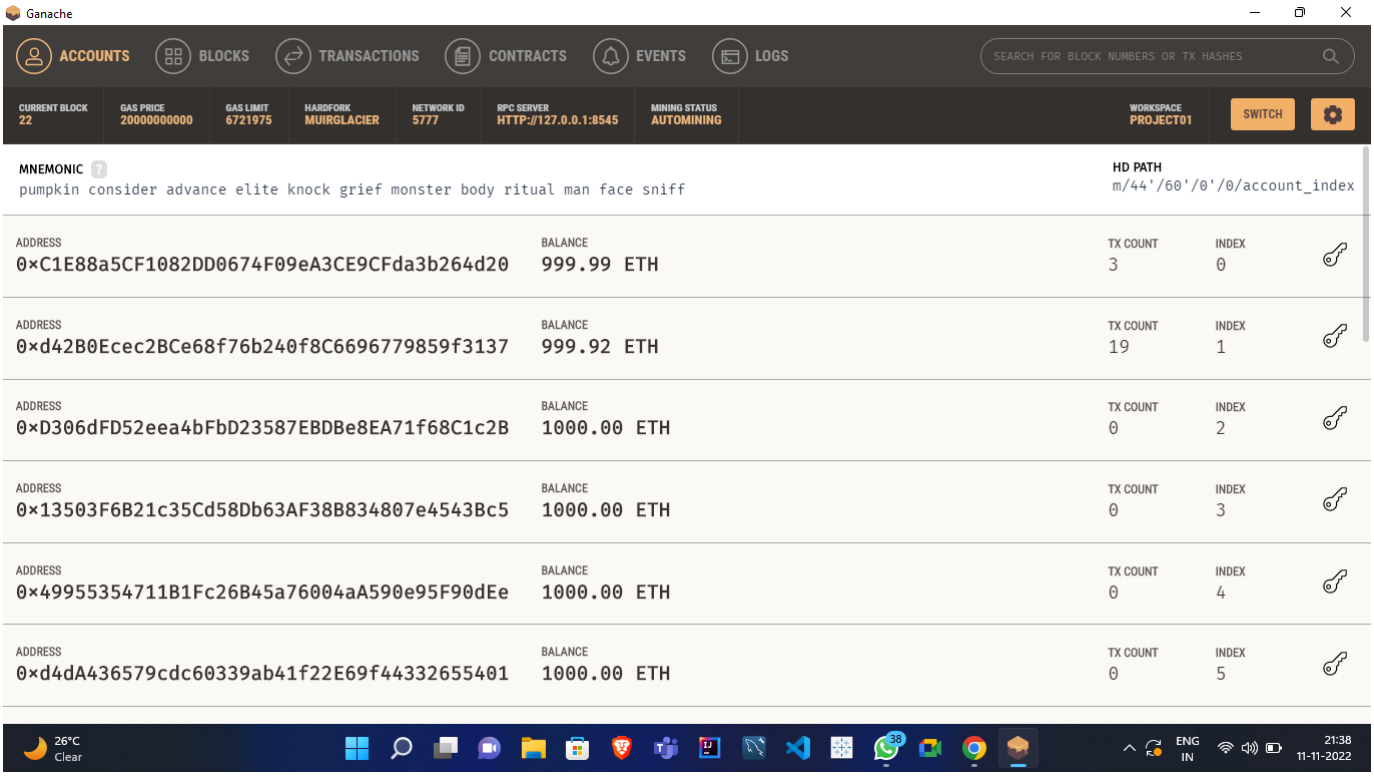


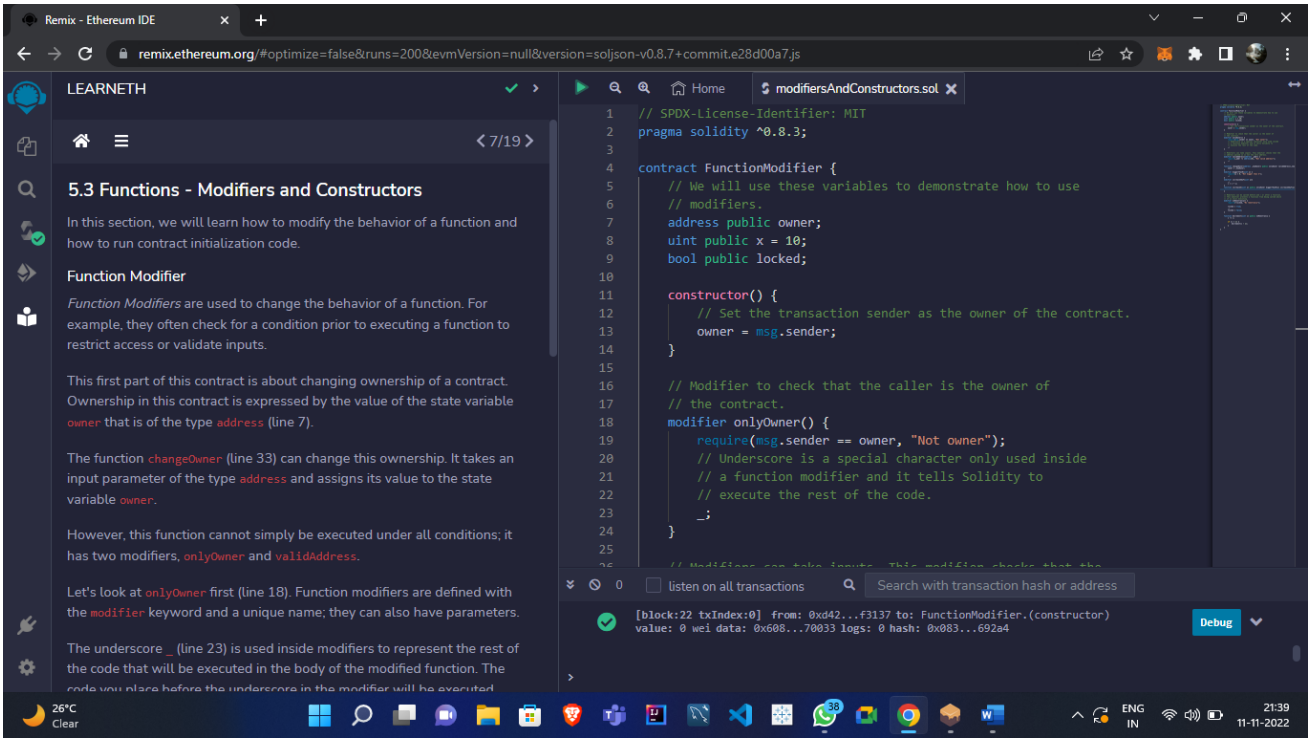
5.3 Functions - Modifiers and Constructors: Assignment

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.

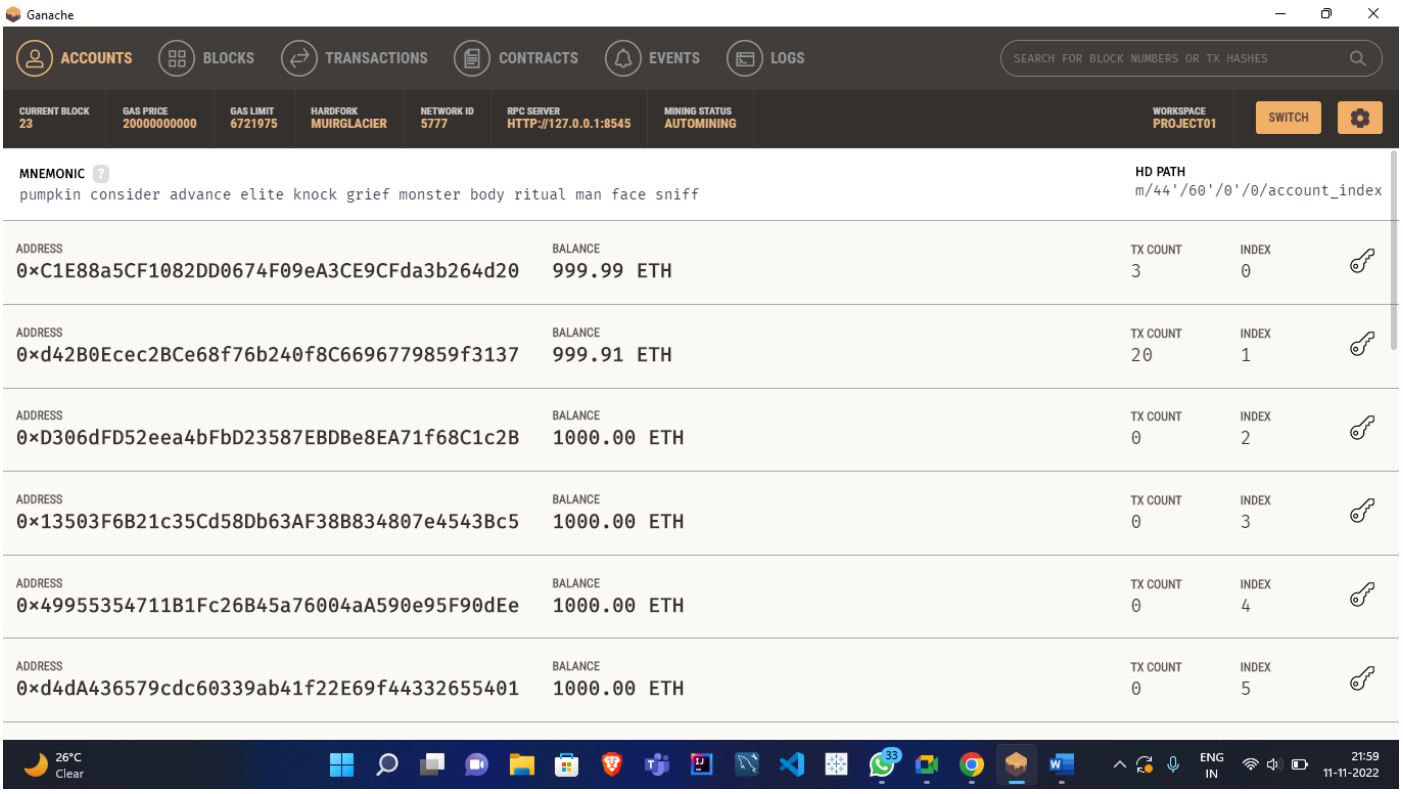
The body of the function increaseX should be empty.

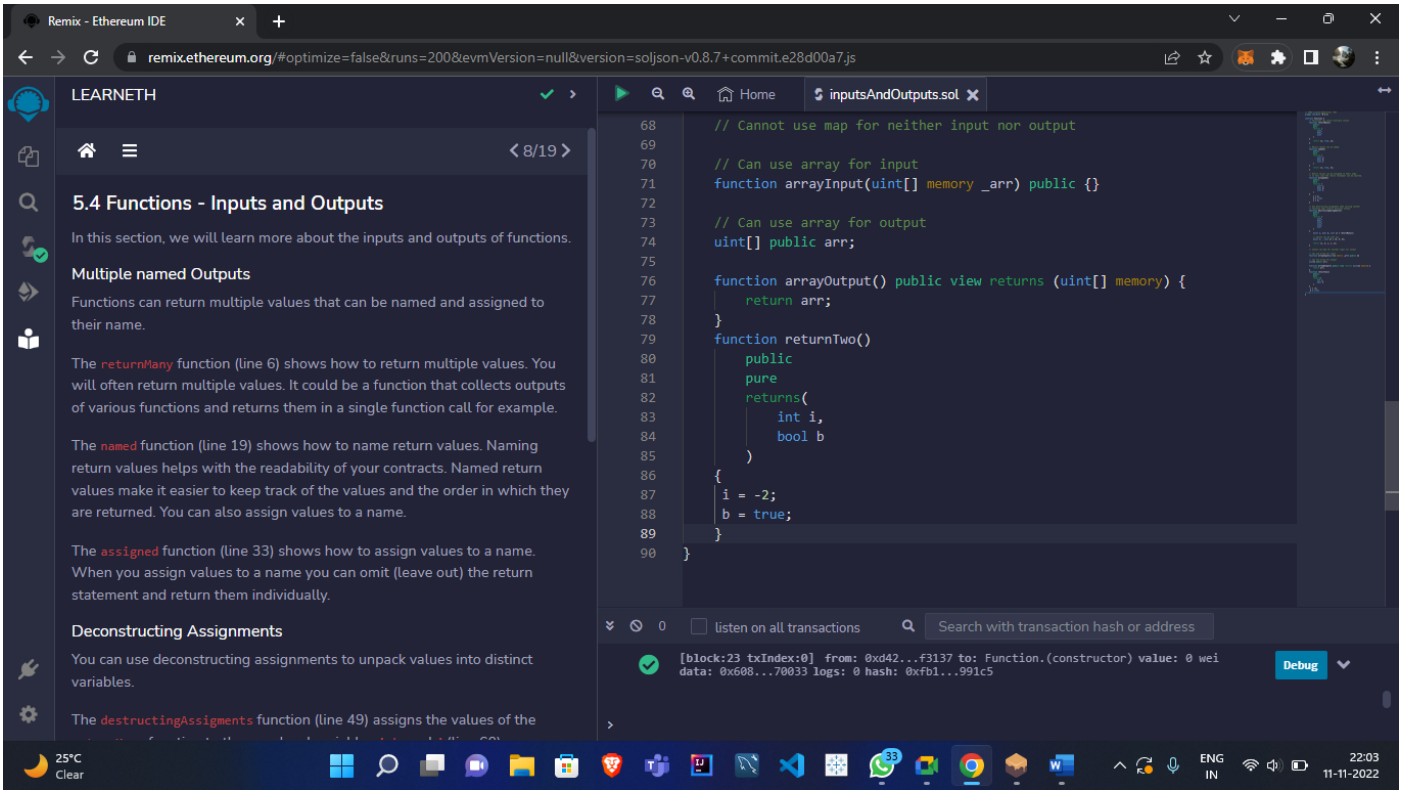




5.4 Functions - Inputs and Outputs :Assignment

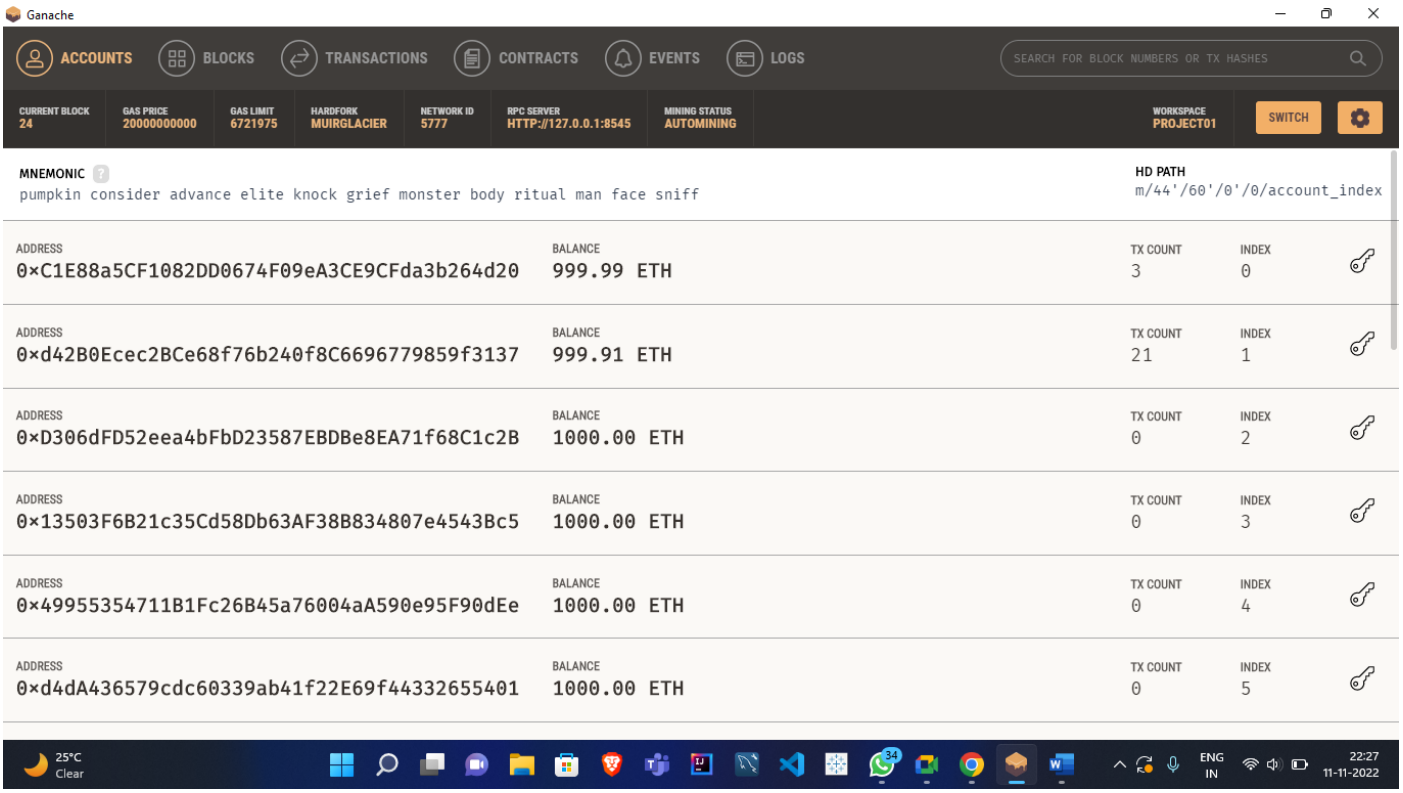
Create a new function called returnTwo that returns the values -2 and true without using a return statement.

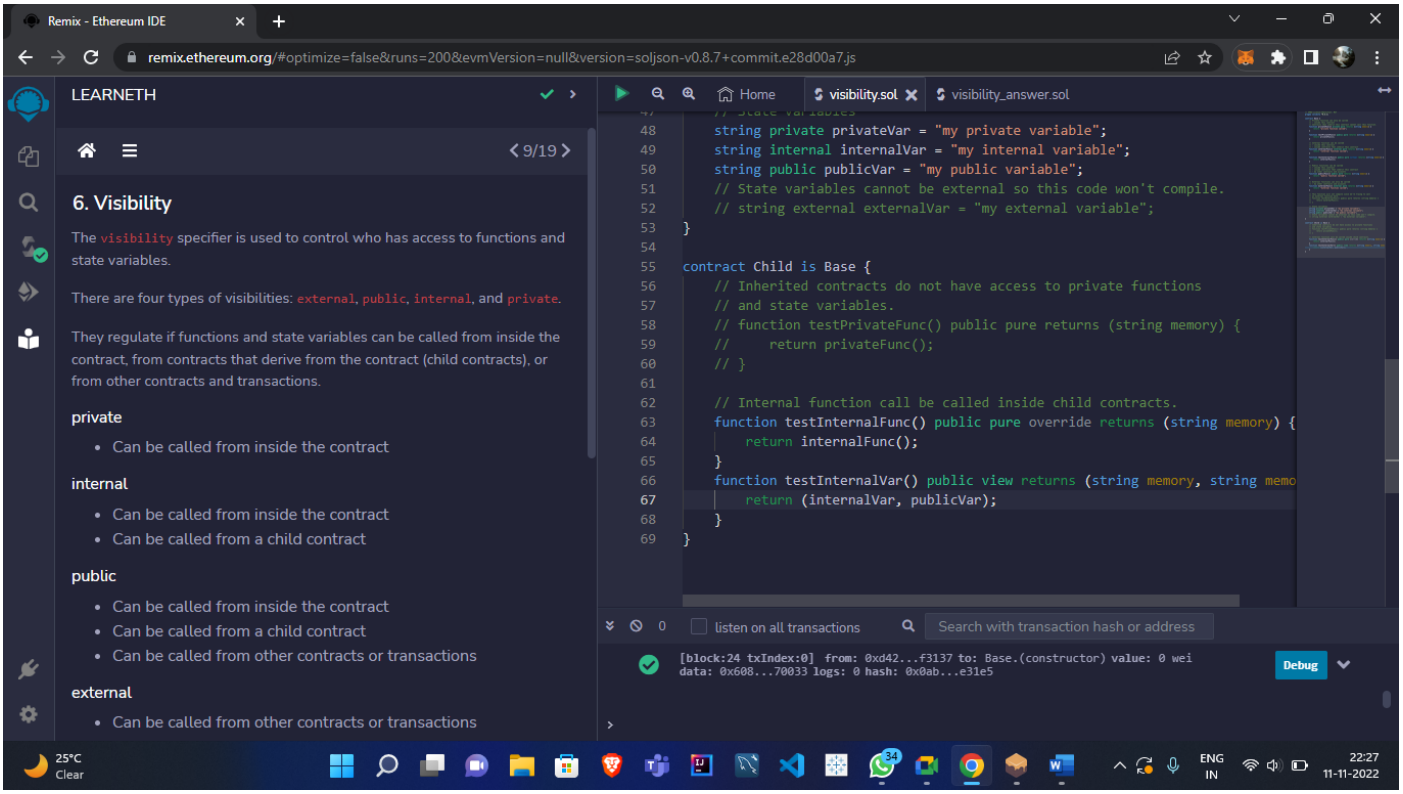




6. Visibility : Assignment

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.





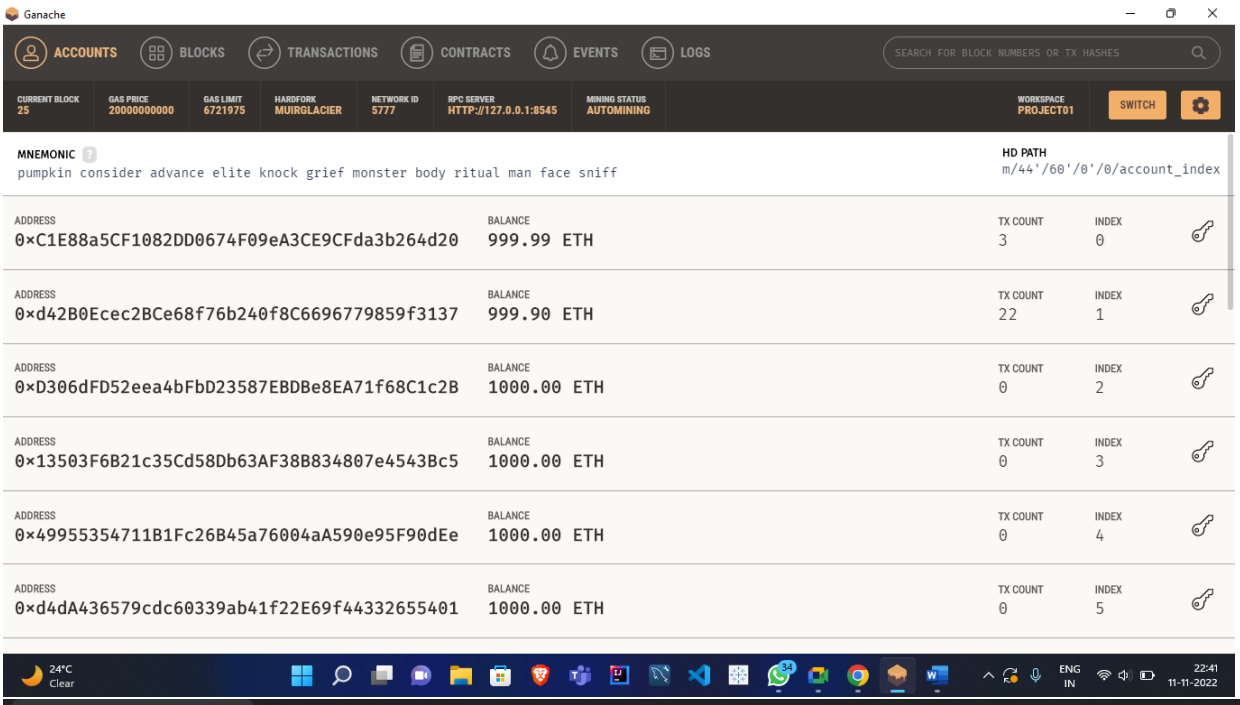
7.1 Control Flow - If/Else: Assignment

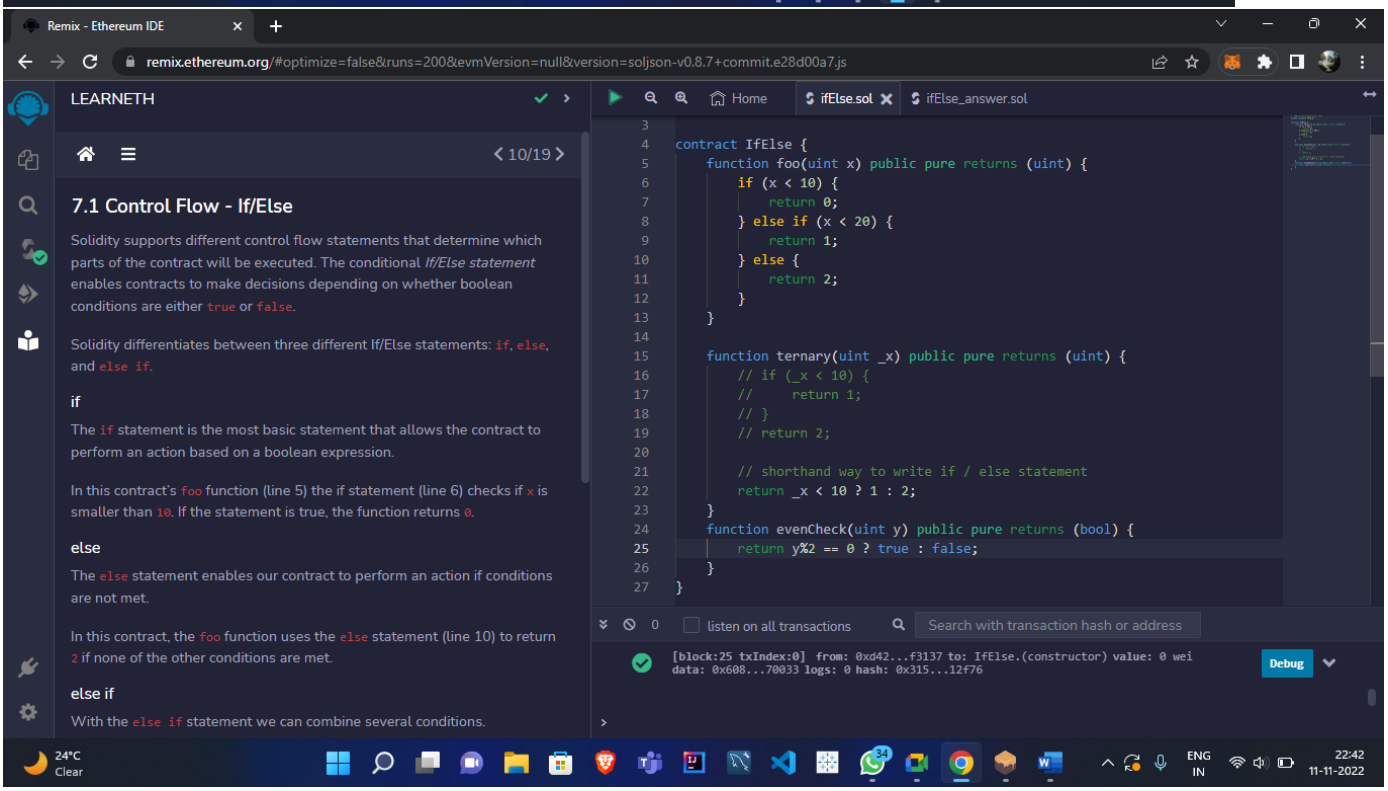
Create a new function called evenCheck in the IfElse contract:

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.





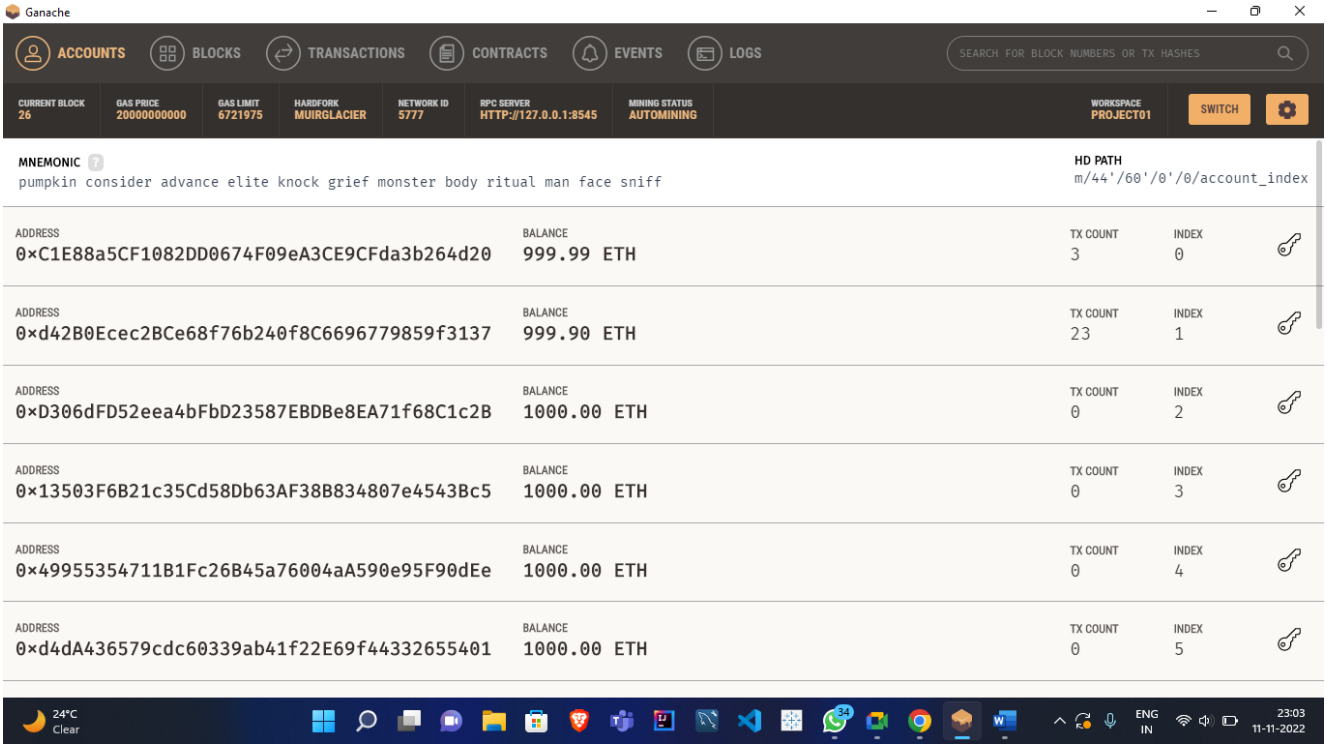
7.2 Control Flow – Loops: Assignment

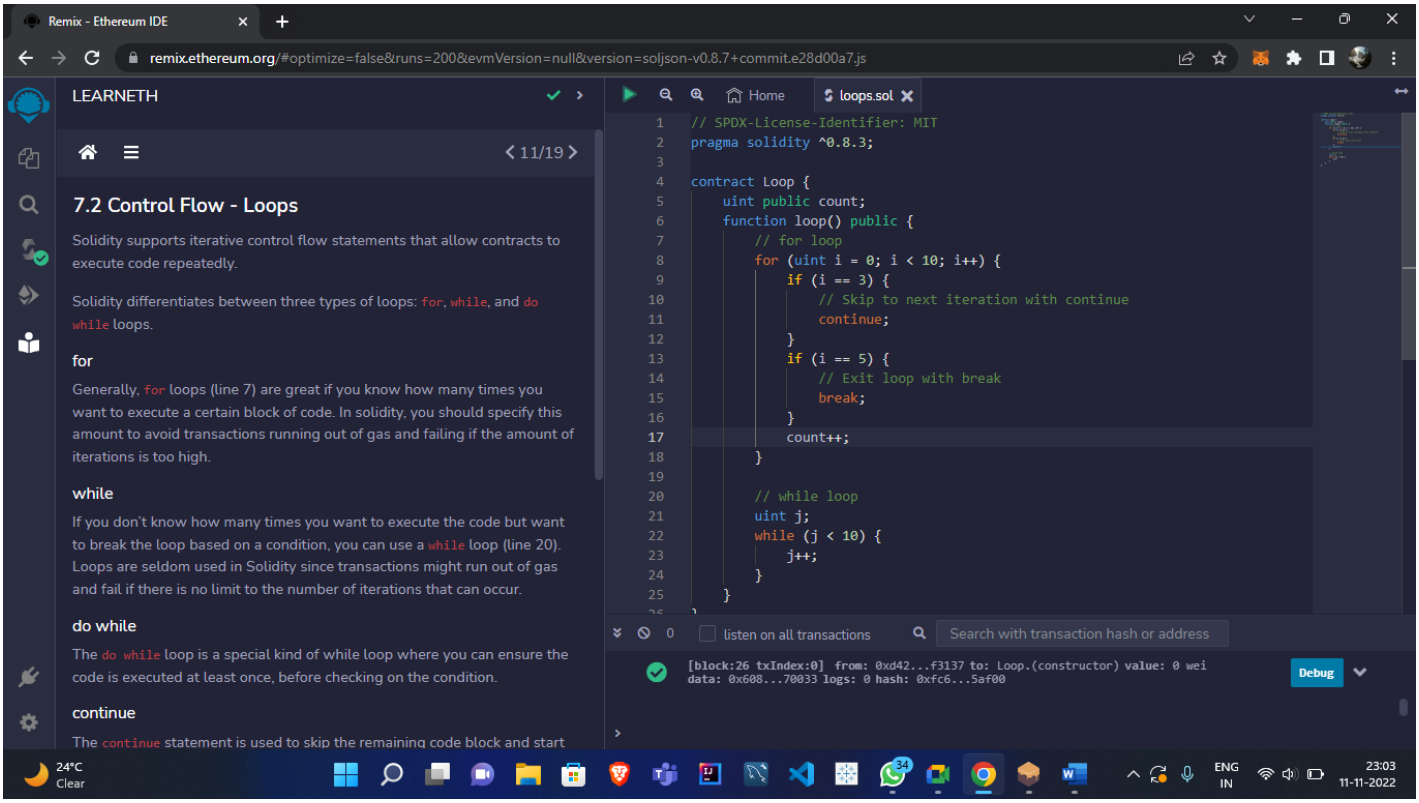
Create a new function called evenCheck in the IfElse contract:

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.

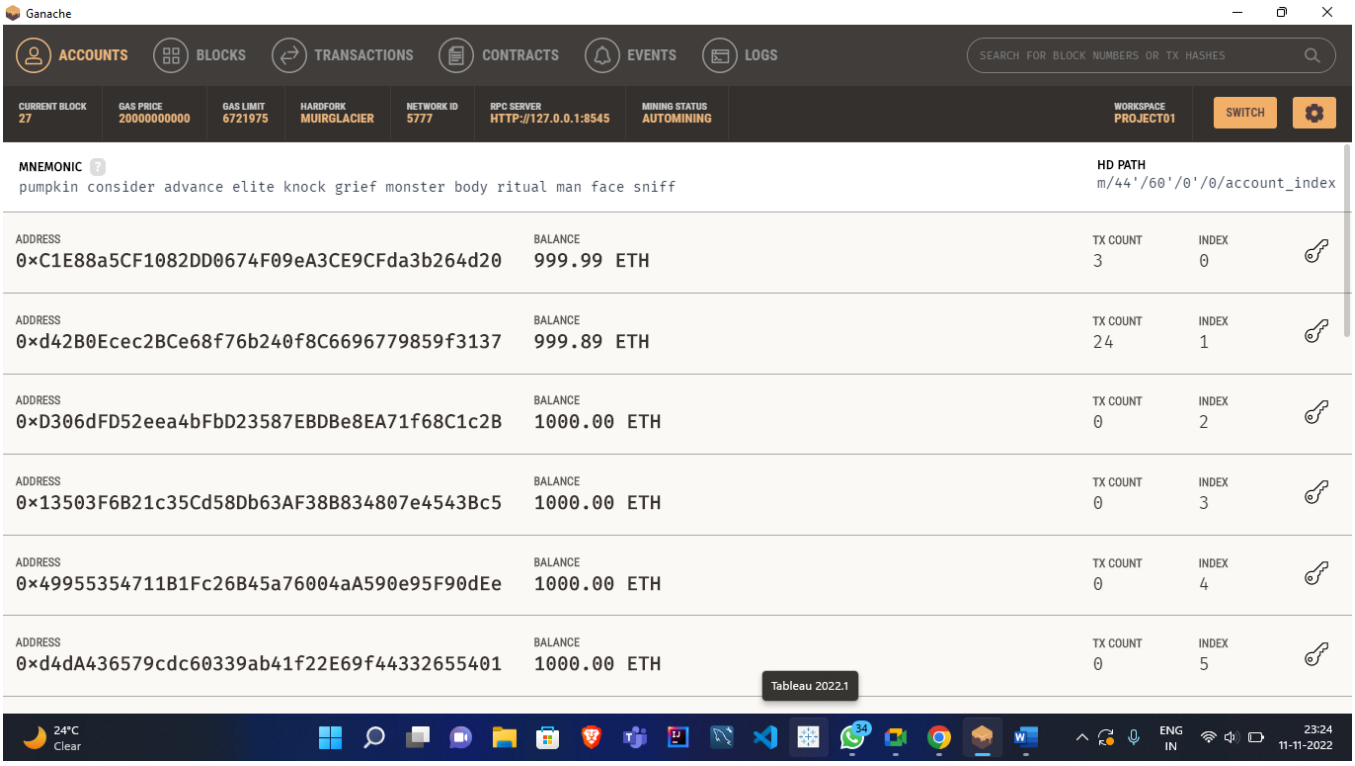


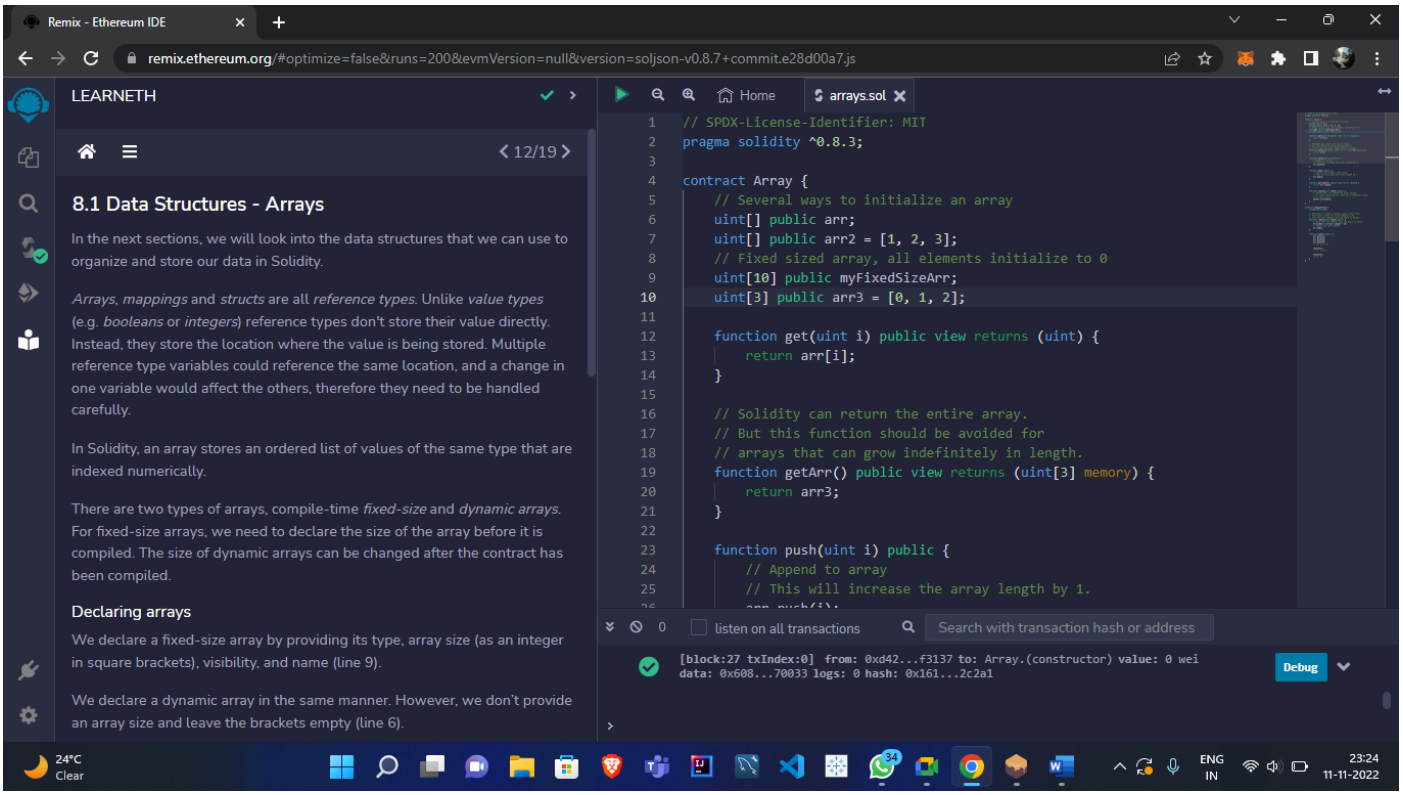


8.1 Data Structures – Arrays: Assignment

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.



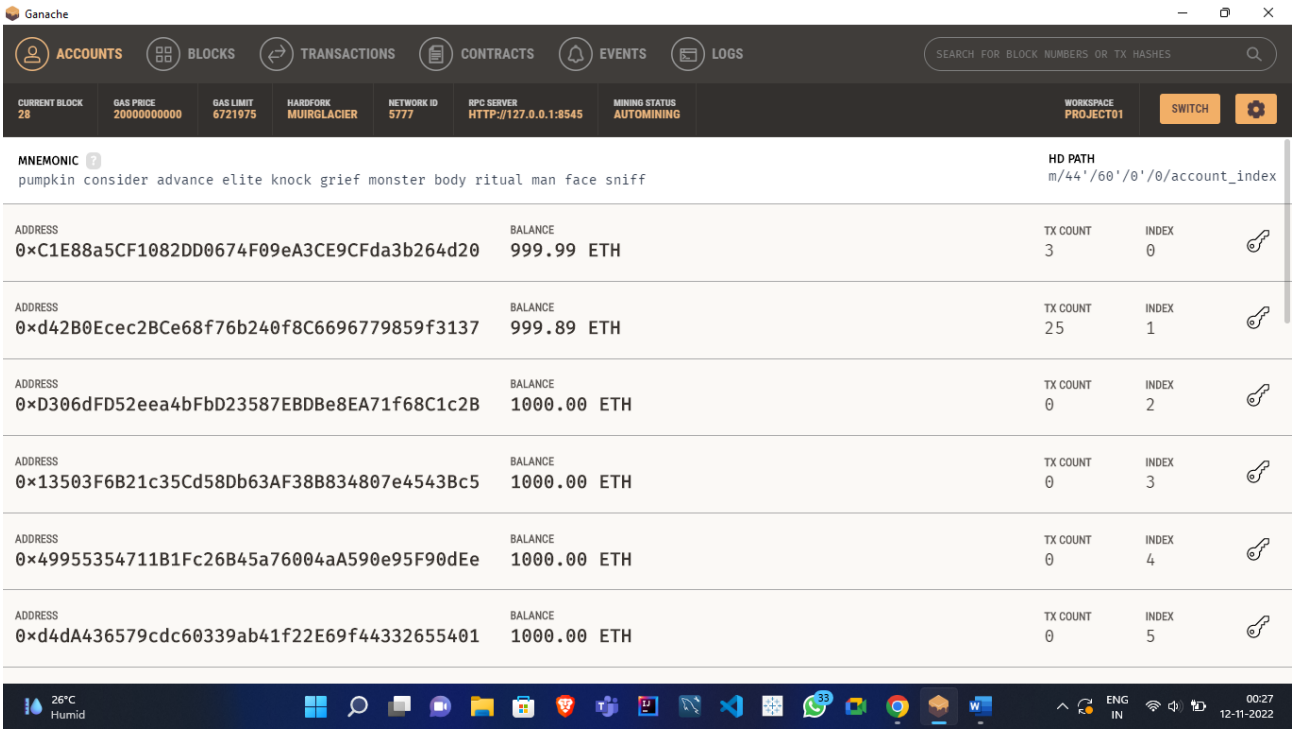


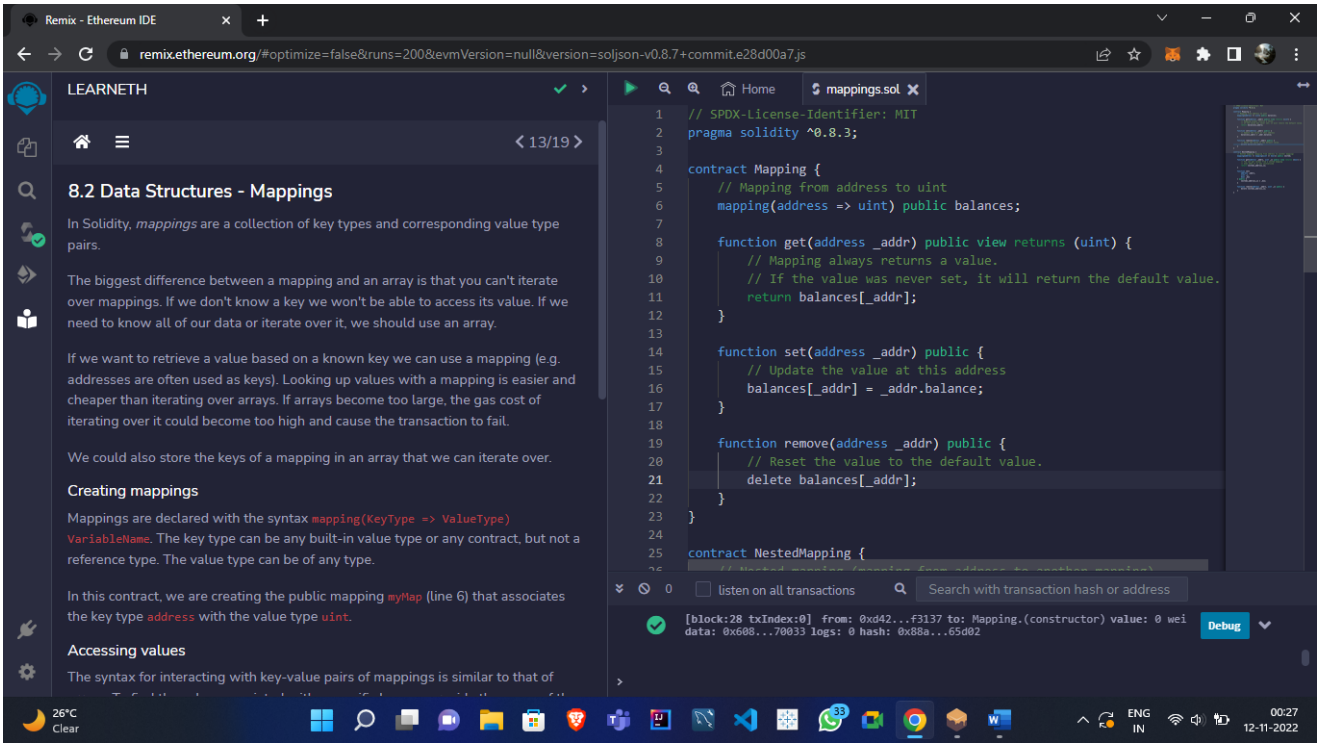
8.2 Data Structures – Mappings: Assignment

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.

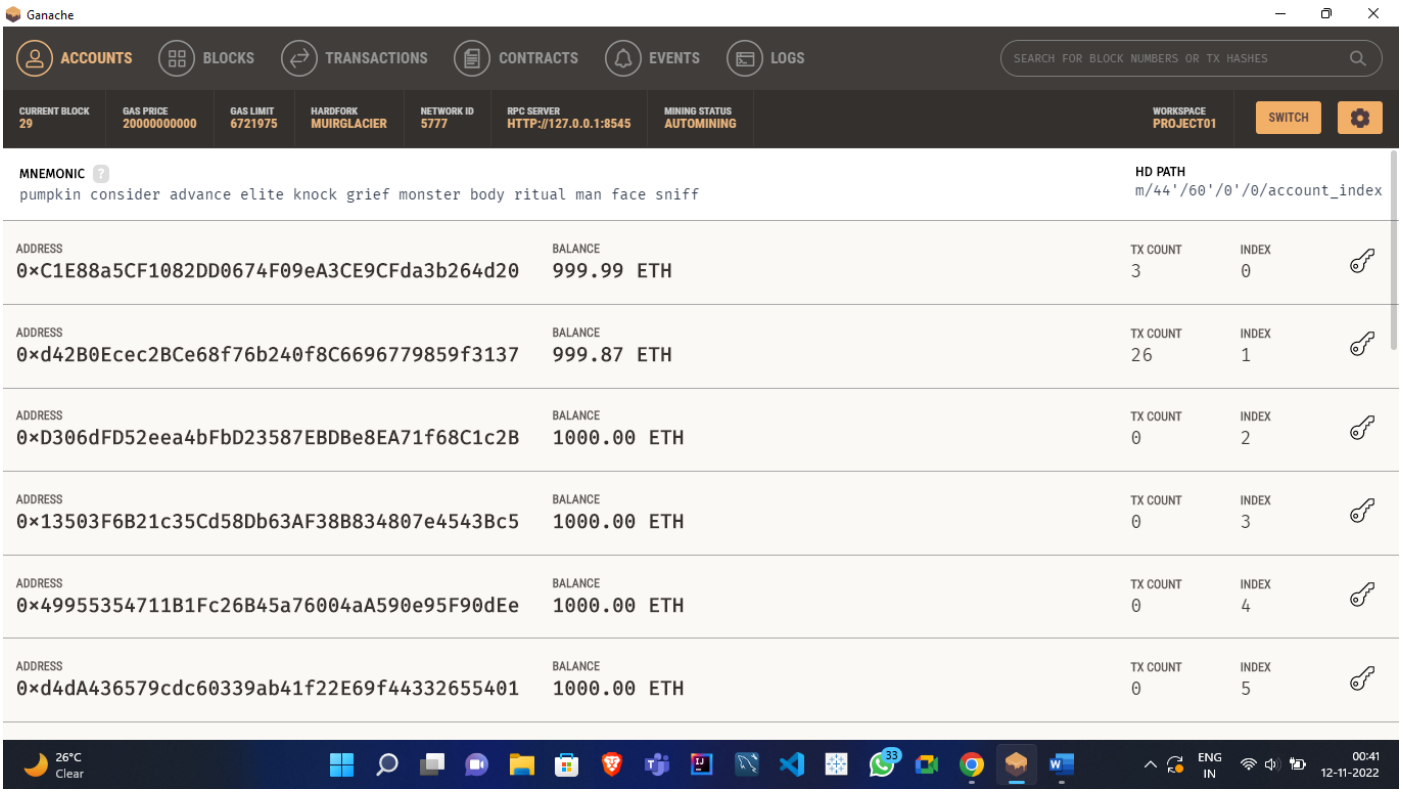
Change the function set to create a new entry to the balances mapping, where the key is the address of the parameter and the value is the balance associated with the address of the parameter.

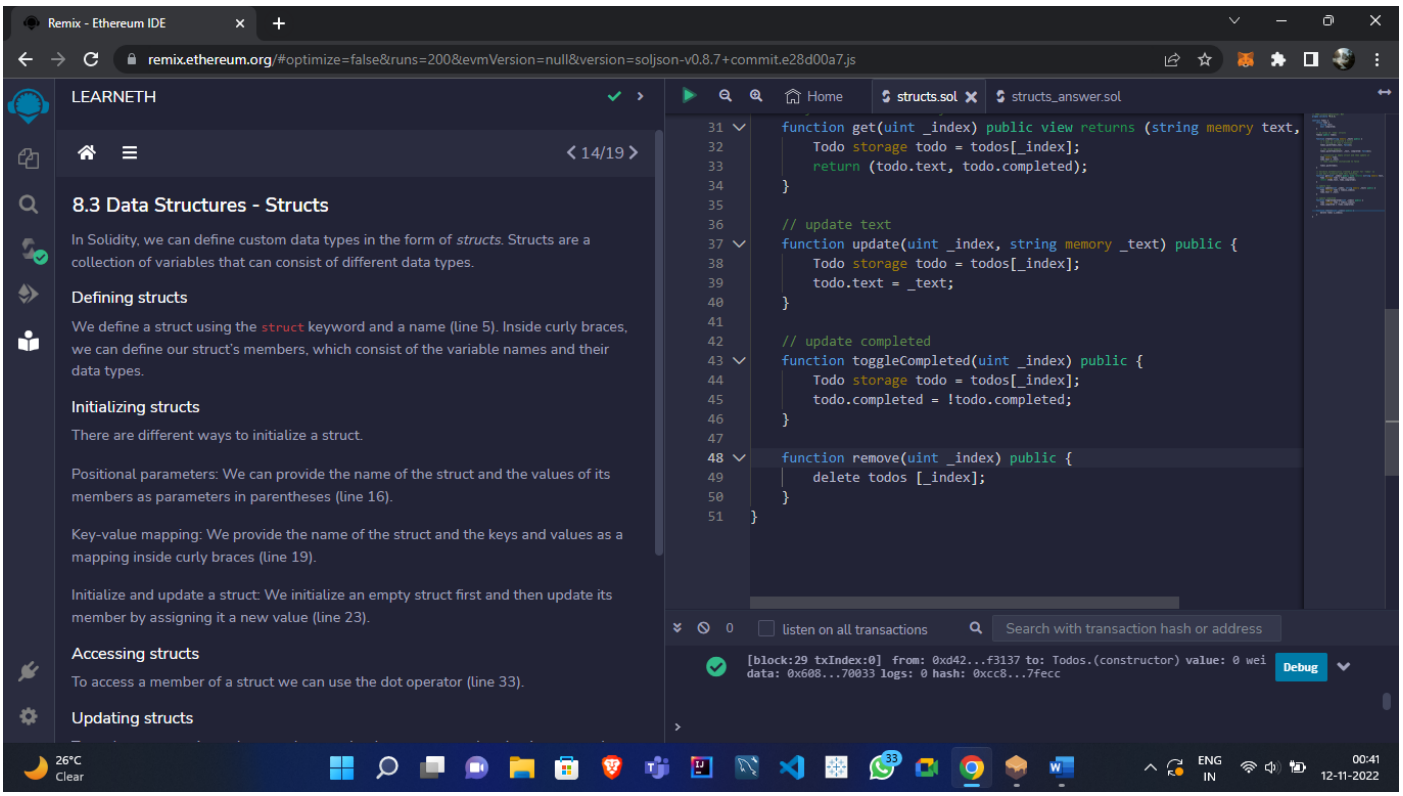




8.3 Data Structures – Structs: Assignment

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



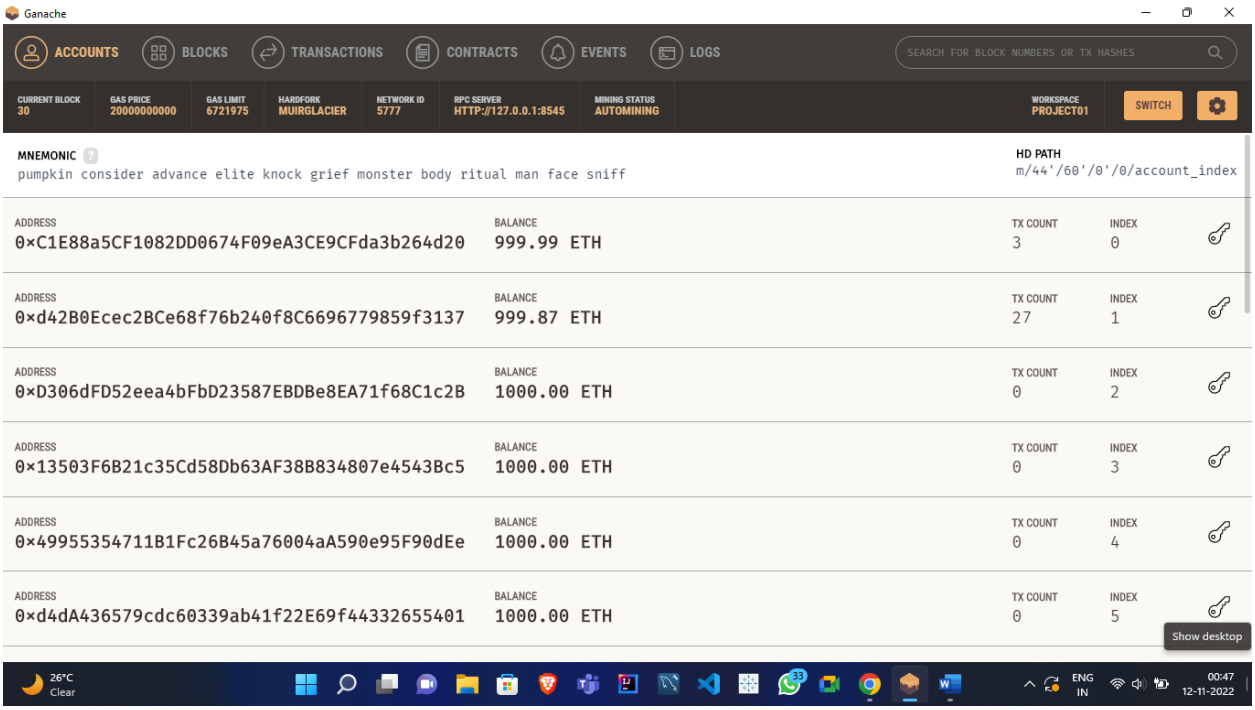


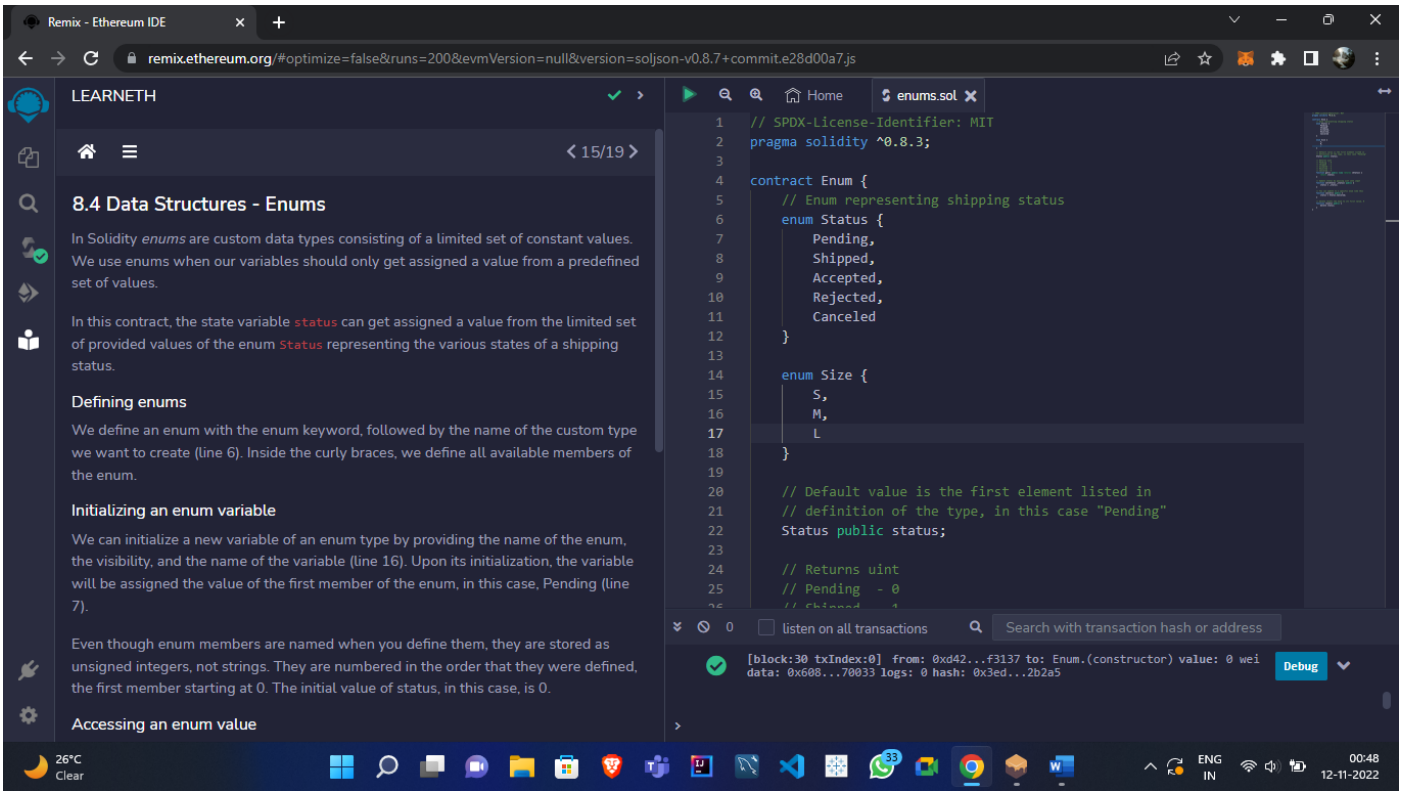
8.4 Data Structures – Enums: Assignment

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

Initialize the variable sizes of the enum type Size.

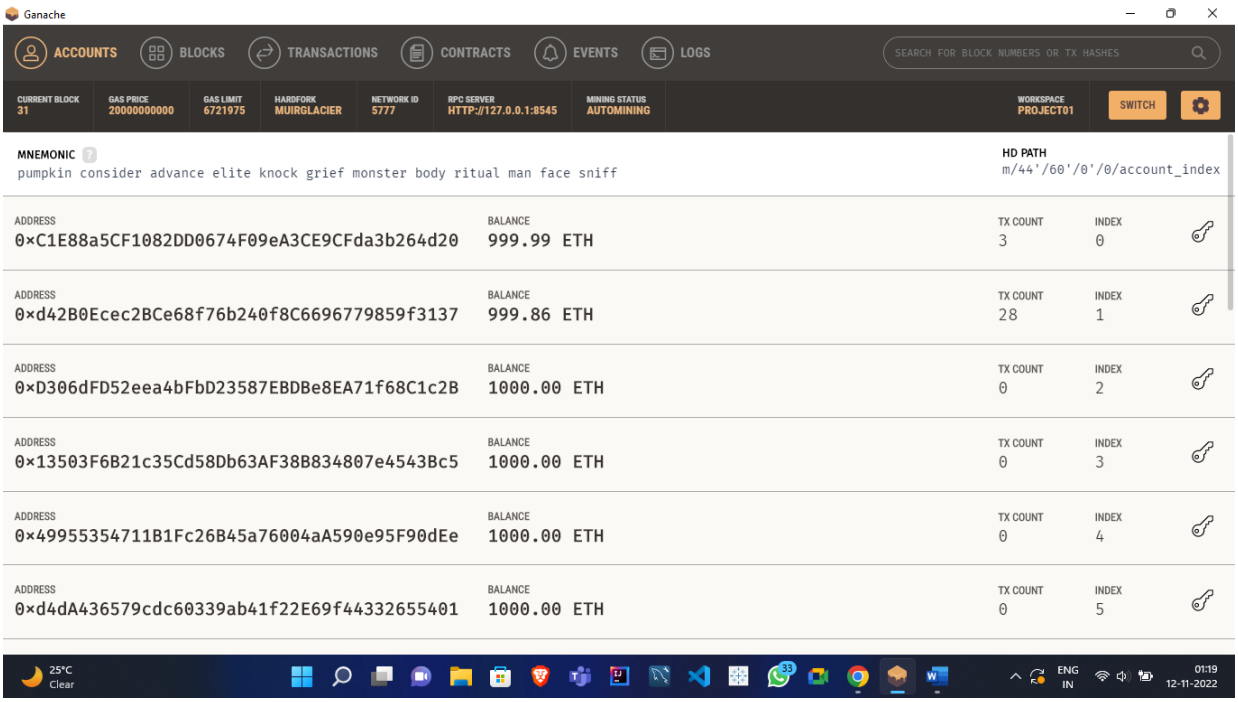
Create a getter function getSize() that returns the value of the variable sizes.

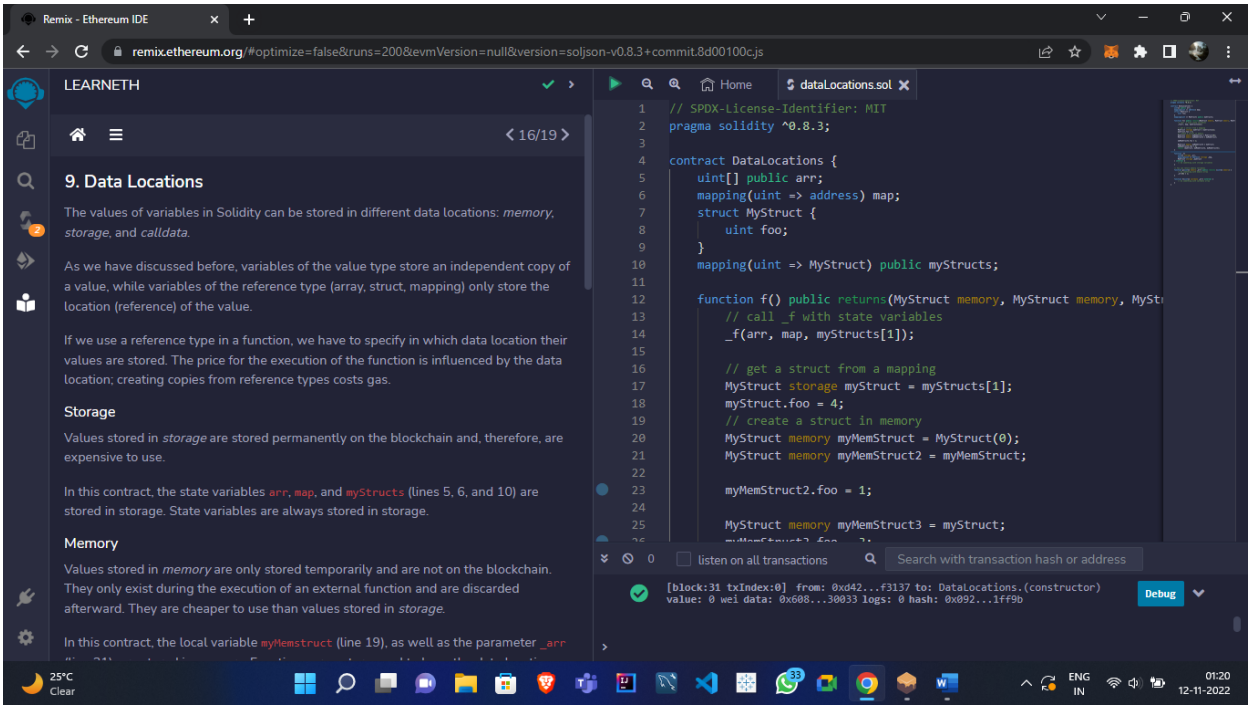




9. Data Locations: Assignment

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.

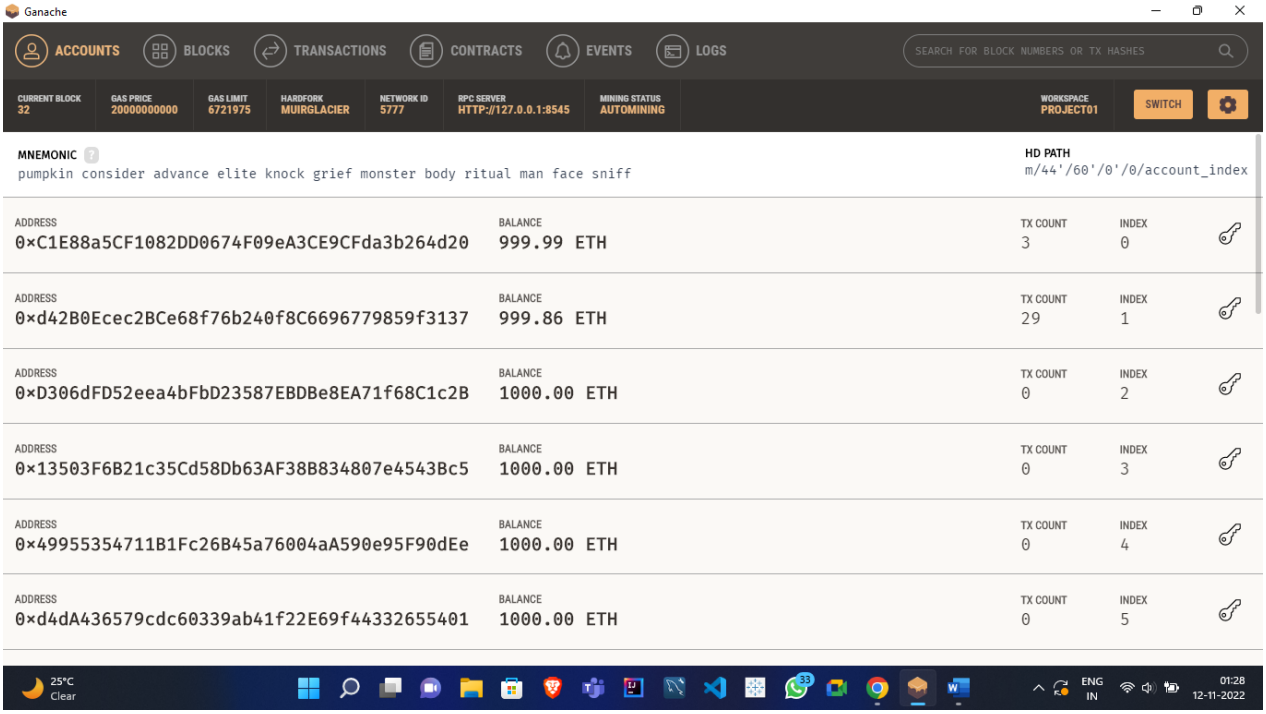


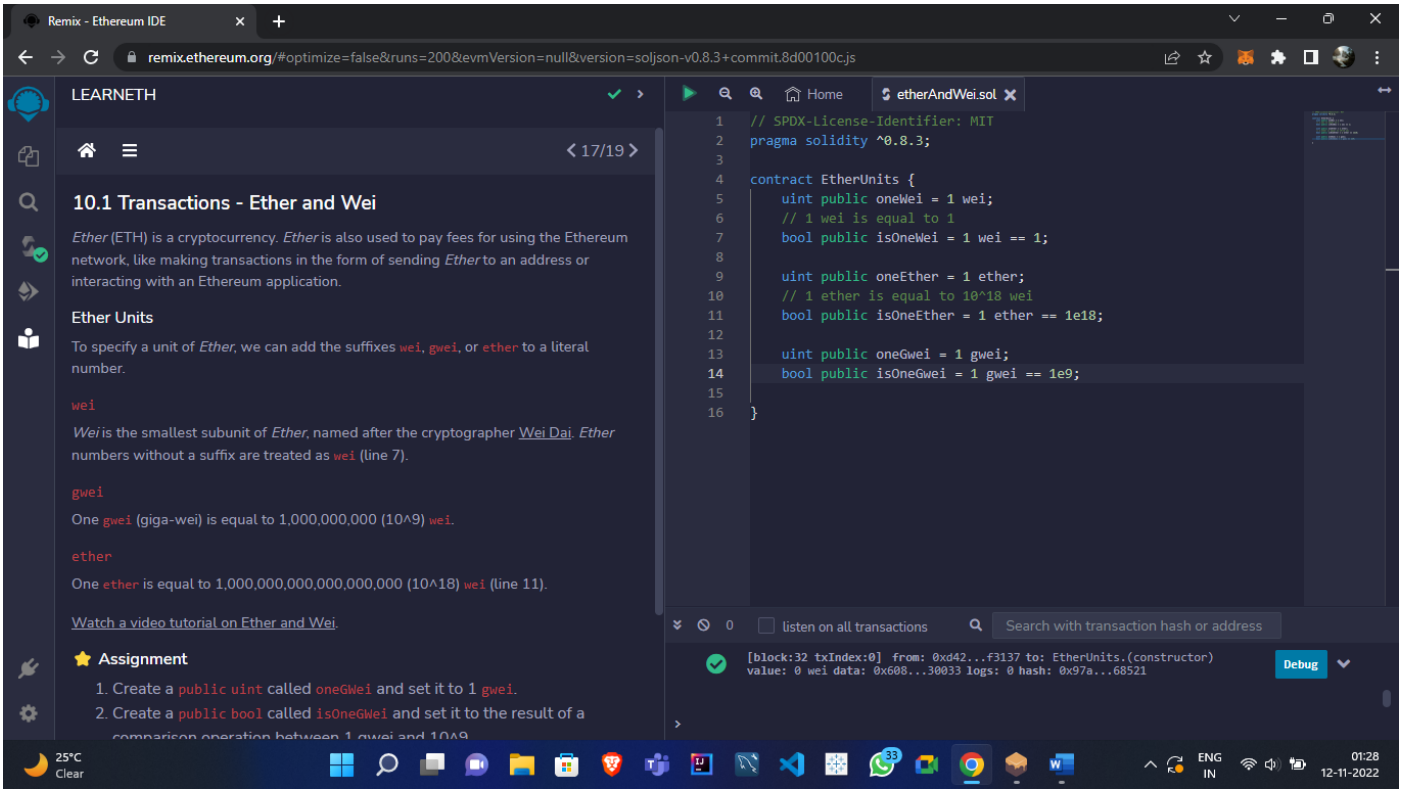


10.1 Transactions - Ether and Wei:

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

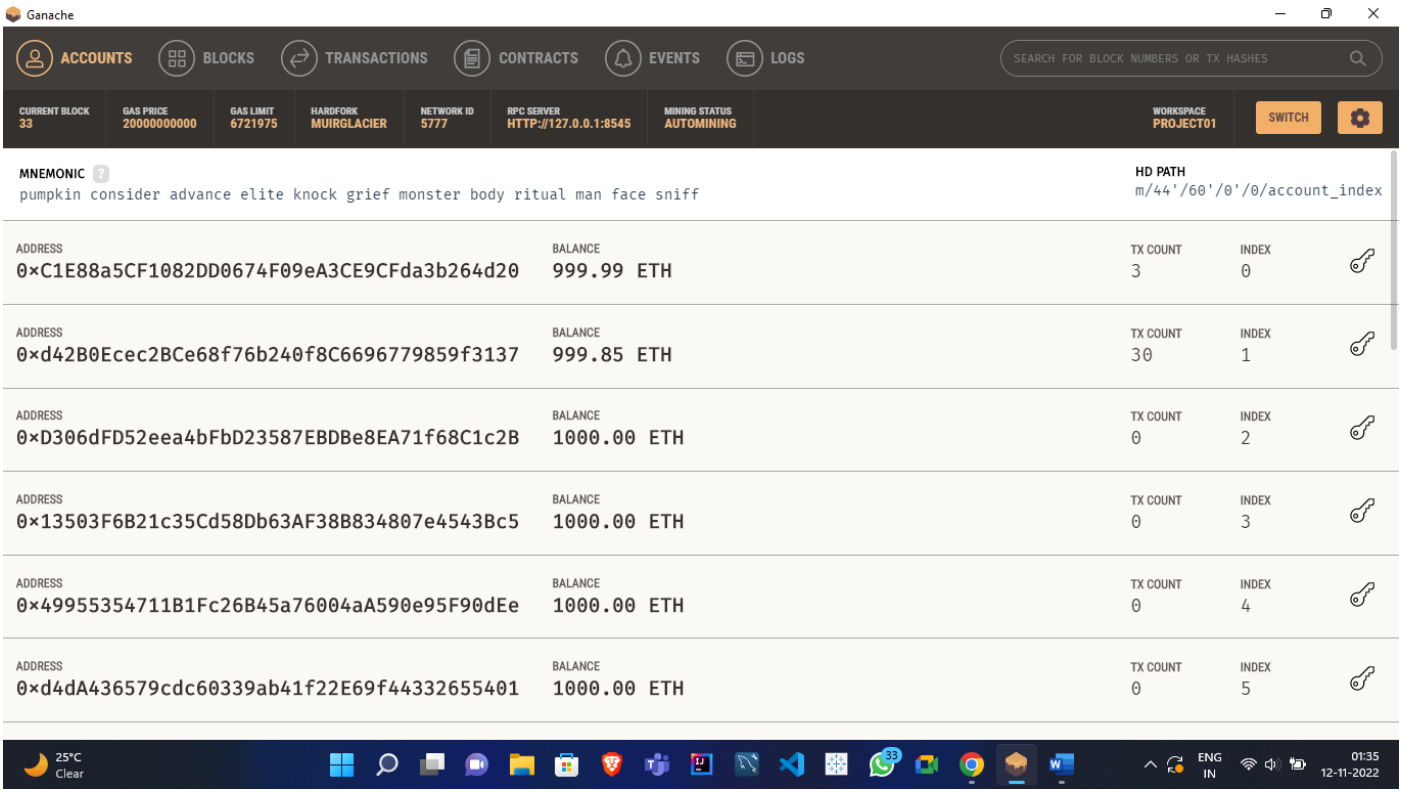
Create a public bool called isOneGWei and set it to the result of a comparison operation between 1 gwei and 10^9.

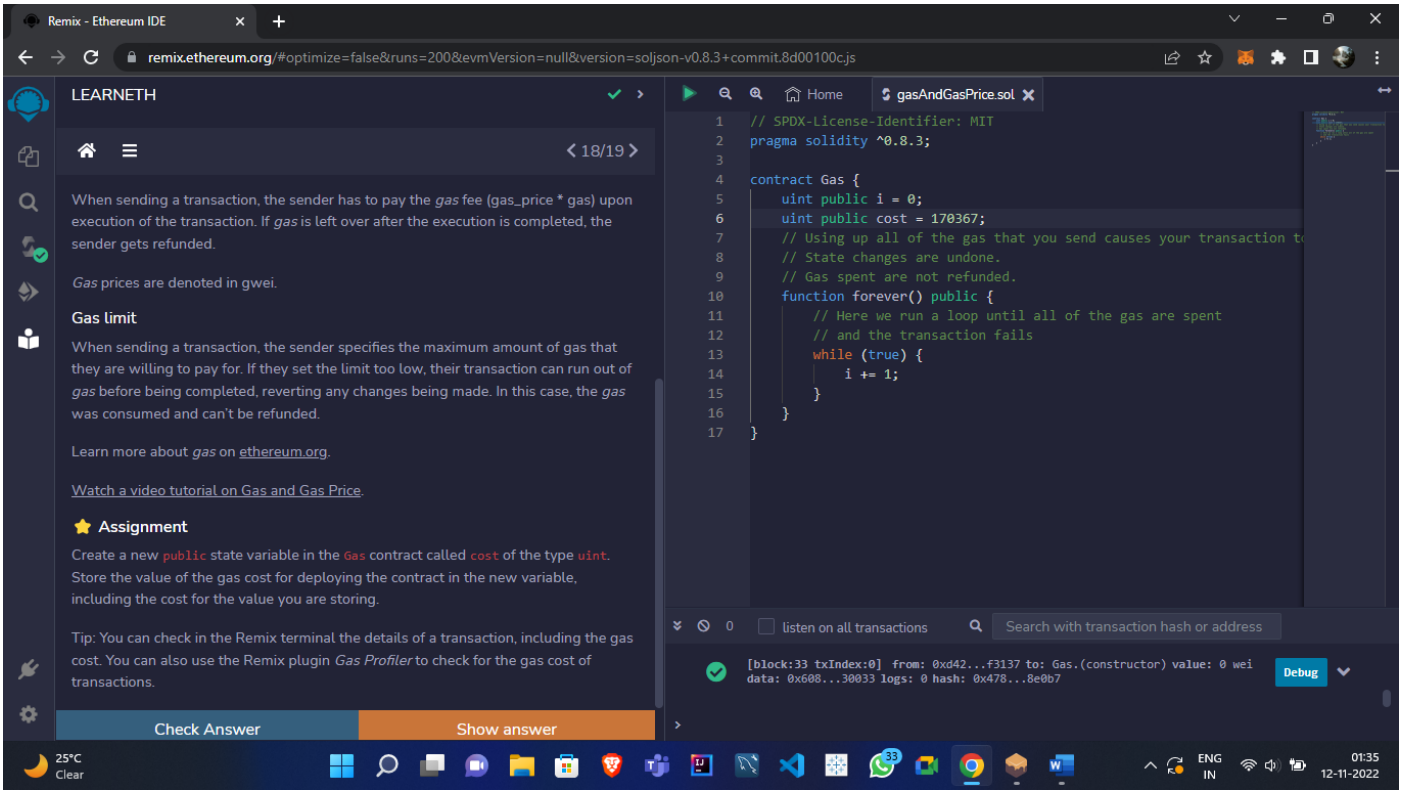




10.2 Transactions - Gas and Gas Price: Assignment

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.