University of Mumbai

PRACTICAL JOURNAL - PAPER I



PSIT4P1a Blockchain

SUBMITTED BY
Ruchita Ramesh More
SEAT NO 40419

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR QUALIFYING M.Sc. (I.T.) PART-II (SEMESTER – IV) EXAMINATION

2022-2023

Department of Information Technology

3rd FLOOR, DR. SHANKAR DAYAL SHARMA BHAVAN, IDOL
BUILDING, VIDYANAGRI,
SANTACRUZ (E), MUMBAI – 400098.

University of Mumbai



Department of Information Technology

CERTIFICATE

This is to certify that Miss. Ruchita Ramesh More Seat No. 40419 studying in Master of Science in Information Technology Part II Semester IV has satisfactorily completed the Practical of PSIT4P1a Blockchain as prescribed by University of Mumbai, during the academic year 2022-23.

Guide	External Examiner Examined by	Head of the Department Certified by
College Seal		Date:

Signature

Signature

Signature

INDEX

Practical No.	Date	Name of Practical	Page No.	Signature
1	18/04/2023	Write the following programs for Blockchain in Python: A] A simple client class that generates the private and public keys by using the builtin Python RSA algorithm and test it. B] A transaction class to send and receive money and test it. C] Create multiple transactions and display them. D] Create a blockchain, a genesis block and execute it. E] Create a mining function and test it. F] Add blocks to the miner and dump the blockchain.	4	
2	18/04/2023	Install and configure Go Ethereum and the Mist browser. Develop and test a sample application.	23	
3	18/04/2023	Implement and demonstrate the use of the following in Solidity: A] Variable, Operators, Loops, Decision Making, Strings, Arrays, Enums, Structs, Mappings, Conversions, Ether Units, Special Variables. B] Functions, Function Modifiers, View functions, Pure Functions, Fallback Function, Function Overloading, Mathematical functions, Cryptographic functions.	32	
4	18/04/2023	Implement and demonstrate the use of the following in Solidity: A] Withdrawal Pattern, Restricted Access. B] Contracts, Inheritance, Constructors, Abstract Contracts, Interfaces. C] Libraries, Assembly, Events, Error handling.	74	

Ruchita More 2 Seat No: 40419

5	23/05/2023	Install hyperledger fabric and composer. Deploy and execute the application.	105	
6	22/06/2023	Write a program to demonstrate mining of Ether.	113	
7	22/06/2023	Demonstrate the running of the blockchain node.	115	
8	22/06/2023	Demonstrate the use of Bitcoin Core API.	117	
9	23/06/2023	Create your own blockchain and demonstrate its use.	120	

Ruchita More 3 Seat No: 40419

PRACTICAL1

AIM: Write the following programs for Blockchain in Python:

[A] A simple client class that generates the private and public keys by using the builtin Python RSA algorithm and test it.

CODE:

```
import binascii
from Crypto.PublicKey import RSA
from Crypto.Signature import PKCS1 v1 5
from Crypto import Random
class Client:
   def___init__(self):
        random = Random.new().read
        self. private key = RSA.generate(1024, random)
        self. public_key = self._private_key.publickey()
        self._signer = PKCS1_v1_5.new(self._private_key)
    @property
    def identity(self):
        return
binascii.hexlify(self. public key.exportKey(format="DER")).decode("a
scii")
# Create an instance of the Client class
UDIT = Client()
# Print the public key (identity) of the client
print("\nPublic Key:", UDIT.identity)
```

Ruchita More 4 Seat No: 40419

OUTPUT:

Public Key:

30819f300d06092a864886f70d010101050003818d0030818902818100c9b694d2aa1855dbbad947
5aac327ab784b7ae7a14b81fbe416e5886f492dff2fb71a80cdb194d212a11523f42271c4a962410
ce6bd9dcf28cbd43aa9fe8a6da2a54345ebbcf3facd1213fcc800a34ded20309db83389299c8bb83
2532b28a281fef07971421d56740a95ddfd789cdaafad97aea839584e3d92aee5664ec42d7020301
0001

Ruchita More 5 Seat No: 40419

[B] A transaction class to send and receive money and test it.

CODE:

```
import binascii
import collections
import datetime
from Crypto. Hash import SHA
from Crypto.PublicKey import RSA
from Crypto.Signature import PKCS1_v1_5
from Crypto import Random
class Client:
   def init (self):
        random = Random.new().read
        self. private key = RSA.generate(1024, random)
        self. public key = self. private key.publickey()
        self. signer = PKCS1 v1 5.new(self. private key)
    @property
    def identity(self):
        return
binascii.hexlify(self. public key.exportKey(format="DER")).decode("a
scii")
class Transaction:
    def init (self, sender, recipient, value):
        self.sender = sender
        self.recipient = recipient
        self.value = value
        self.time = datetime.datetime.now()
    def to dict(self):
```

```
identity = "Genesis" if self.sender == "Genesis" else
self.sender.identity
       return collections.OrderedDict(
                "sender": identity,
               "recipient": self.recipient,
               "value": self.value,
               "time": self.time,
           }
   def sign transaction(self):
       private key = self.sender. private key
       signer = PKCS1 v1 5.new(private key)
       h = SHA.new(str(self.to dict()).encode("utf8"))
        return binascii.hexlify(signer.sign(h)).decode("ascii")
UDIT = Client()
UGC = Client()
t = Transaction(UDIT, UGC.identity, 5.0)
print("\nTransaction Recipient:\n", t.recipient)
print("\nTransaction Sender:\n", t.sender) print("\nTransaction
Value:\n", t.value)
signature = t.sign transaction()
print("\nSignature:\n", signature)
```

OUTPUT:

Transaction Recipient:

30819f300d06092a864886f70d010101050003818d0030818902818100be0978593e24a777060c9b 95c383a53f3e3213905fc67538a37f15e9c2c4bd6e0eb667999074079928c3e3d7f0636f0d7e9f8 cc0bd7a38da76342c612c103fe43a603e97c602d1f2a0dbbe794ab983aa1d1062d70485c0e0c734 3c265e90ae4094c6ede9037b0d38af530b27914df1a08c339ec4bc76de28b2dbe5cd6a0e3ebf020 3010001

Signature:

a3d6c2635b55cd891bc43dc63b137bfb10eaa68dd7ce276f8e098367cc83195f5a508fa2e4e2c74a
341e143d38432b36e4dc240f5f4b17d07d5be3ebf67a59f42583c215e9d78059091580629eee052
985b2bf355ab58eae226b41dbfaa9690bfce61745392e9f172173ce5a2f12040b753610468ca4a3
fccb1437968a11fc00

Ruchita More 8 Seat No: 40419

[C] Create multiple transactions and display them.

CODE:

```
import binascii
import collections
import datetime
from Crypto. Hash import SHA
from Crypto.PublicKey import RSA
from Crypto.Signature import PKCS1_v1_5
from Crypto import Random
class Client:
   def init (self):
        random = Random.new().read
        self. private key = RSA.generate(1024, random)
        self. public key = self. private key.publickey()
        self. signer = PKCS1 v1 5.new(self. private key)
    @property
    def identity(self):
        return
binascii.hexlify(self. public key.exportKey(format="DER")).decode("a
scii")
class Transaction:
    def init (self, sender, recipient, value):
        self.sender = sender
        self.recipient = recipient
        self.value = value
        self.time = datetime.datetime.now()
    def to dict(self):
```

```
identity = "Genesis" if self.sender == "Genesis" else
self.sender.identity
        return collections.OrderedDict(
                "sender": identity,
                "recipient": self.recipient,
                "value": self.value,
                "time": self.time,
            }
    def sign transaction(self):
        private key = self.sender. private key
        signer = PKCS1_v1_5.new(private_key)
        h = SHA.new(str(self.to dict()).encode("utf8"))
        return binascii.hexlify(signer.sign(h)).decode("ascii")
    def display transaction(transaction):
        # for transaction in transactions:
        dict = transaction.to dict()
        print("sender: " + dict['sender'])
        print('---- ')
        print("recipient: " + dict['recipient'])
        print('---- ')
        print("value: " + str(dict['value']))
        print('---- ')
        print("time: " + str(dict['time']))
        print('---- ')
UDIT = Client()
UGC = Client()
AICTE = Client()
MU = Client()
```

```
t1 = Transaction(UDIT, UGC.identity, 15.0)
t1.sign_transaction()
transactions = [t1]
t2 = Transaction(UDIT, AICTE.identity, 6.0)
t2.sign_transaction()
transactions.append(t2)
t3 = Transaction(UGC, MU.identity, 2.0)
t3.sign_transaction()
transactions.append(t3)
t4 = Transaction(AICTE, UGC.identity, 4.0)
t4.sign_transaction()
transactions.append(t4)
for transaction in transactions:
    Transaction.display_transaction(transaction)
    print(" ")
```

Ruchita More 11 Seat No: 40419

OUTPUT:

sender:

30819f300d06092a864886f70d010101050003818d00308189028181009c383a3fc7
248a85bb871be70bdad857db28da890de9de76f3df85950fa0452ab1615bf6b7967f
28c642e433a04335befdb4232c0735d271a0949a28e51e9154f327c7edd6be6e5588
73c12afbd66a48c6fc726515789d1109438f75446829b3832be14e894a40f27e1331
cc48e536a30cc47a1039b4d87364a3ac2c3f7553110203010001

recipient:

30819f300d06092a864886f70d010101050003818d0030818902818100a597e8496e f09e903eac0b9f97d8bfde76565f58599206bbd47fb020b0d0daef0568449ef7cb44 68abe58a30e7877276404a5264840f4e0ddda570cbefec408c459d58429573427694 382caa972f3878f7ae04ff27bbea057bf9360dd65e193eaf8301c5168840e6a55812 867b746de1da1695c5130d49cbee519c6cb23698710203010001

value: 15.0

time: 2023-06-25 18:38:53.536480

sender:

30819f300d06092a864886f70d010101050003818d00308189028181009c383a3fc7
248a85bb871be70bdad857db28da890de9de76f3df85950fa0452ab1615bf6b7967f
28c642e433a04335befdb4232c0735d271a0949a28e51e9154f327c7edd6be6e5588
73c12afbd66a48c6fc726515789d1109438f75446829b3832be14e894a40f27e1331
cc48e536a30cc47a1039b4d87364a3ac2c3f7553110203010001

recipient:

30819f300d06092a864886f70d010101050003818d0030818902818100cfc0b7e441 5f62ea994a001e1d30f8d23d89ec17062f162c5f2ac56c4c5883ace946a59854562d 116efa15269b50a0d180e8b46db923ec942c347826037007c3b638bd666a4949e773 bc98ba3ff5e0f4fa06e97f0d31986ee2090e02a4ba9dd60951d48e8253763a08730b 0c4e4f3aae04ac568b9c9708406a5c9564d03ace250203010001

value: 6.0

time: 2023-06-25 18:38:53.538479

sender:

30819f300d06092a864886f70d010101050003818d0030818902818100a597e8496e f09e903eac0b9f97d8bfde76565f58599206bbd47fb020b0d0daef0568449ef7cb44 68abe58a30e7877276404a5264840f4e0ddda570cbefec408c459d58429573427694 382caa972f3878f7ae04ff27bbea057bf9360dd65e193eaf8301c5168840e6a55812 867b746de1da1695c5130d49cbee519c6cb23698710203010001

recipient:

30819f300d06092a864886f70d010101050003818d0030818902818100a51bb5b3ba ef116ce6aaf25937457f3b789787be44b0512795ee48ab373b93af9d103ebd77ec1e de442c493867c4a8fad6b50dc5ee1cc7daec59e0615d855f04b45c4f35796194185d e848b59ab6ae4ca9946ceab649192b998708db6d9c62927ff4516e1d330f67fa5e4a 6c32148c6b0206686ef15e234c85a8c70366c33e910203010001

value: 2.0

time: 2023-06-25 18:38:53.539479

Ruchita More 14 Seat No: 40419

sender:

30819f300d06092a864886f70d010101050003818d0030818902818100cfc0b7e441
5f62ea994a001e1d30f8d23d89ec17062f162c5f2ac56c4c5883ace946a59854562d
116efa15269b50a0d180e8b46db923ec942c347826037007c3b638bd666a4949e773
bc98ba3ff5e0f4fa06e97f0d31986ee2090e02a4ba9dd60951d48e8253763a08730b
0c4e4f3aae04ac568b9c9708406a5c9564d03ace250203010001

recipient:

30819f300d06092a864886f70d010101050003818d0030818902818100a597e8496e f09e903eac0b9f97d8bfde76565f58599206bbd47fb020b0d0daef0568449ef7cb44 68abe58a30e7877276404a5264840f4e0ddda570cbefec408c459d58429573427694 382caa972f3878f7ae04ff27bbea057bf9360dd65e193eaf8301c5168840e6a55812 867b746de1da1695c5130d49cbee519c6cb23698710203010001

value: 4.0

time: 2023-06-25 18:38:53.540479

[D] Create a blockchain, a genesis block and execute it.

CODE:

```
import binascii
import collections
import datetime
from Crypto. Hash import SHA
from Crypto.PublicKey import RSA
from Crypto.Signature import PKCS1_v1_5
from Crypto import Random
class Client:
   def init (self):
        random = Random.new().read
        self. private key = RSA.generate(1024, random)
        self. public key = self. private key.publickey()
        self. signer = PKCS1 v1 5.new(self. private key)
    @property
    def identity(self):
        return
binascii.hexlify(self. public key.exportKey(format="DER")).decode("a
scii")
class Transaction:
    def init (self, sender, recipient, value):
        self.sender = sender
        self.recipient = recipient
        self.value = value
        self.time = datetime.datetime.now()
    def to dict(self):
```

```
identity = "Genesis" if self.sender == "Genesis" else
self.sender.identity
        return collections.OrderedDict(
                "sender": identity,
                "recipient": self.recipient,
                "value": self.value,
                "time": self.time,
            }
    def sign transaction(self):
        private key = self.sender. private key
        signer = PKCS1_v1_5.new(private_key)
        h = SHA.new(str(self.to dict()).encode("utf8"))
        return binascii.hexlify(signer.sign(h)).decode("ascii")
    def display transaction(transaction):
        # for transaction in transactions:
        dict = transaction.to dict()
        print("sender: " + dict['sender'])
        print('---- ')
        print("recipient: " + dict['recipient'])
        print('---- ')
        print("value: " + str(dict['value']))
        print('---- ')
        print("time: " + str(dict['time']))
        print('---- ')
class Block:
   def init (self, client):
        self.verified transactions = []
        self.previous_block_hash = ""
        self.Nonce = ""
```

```
self.client = client
def dump_blockchain(blocks):
   print(f"\nNumber of blocks in the chain: {len(blocks)}")
   for i, block in enumerate(blocks):
        print(f"block # {i}")
        for transaction in block.verified transactions:
            Transaction.display transaction(transaction)
            print(" ")
   print(" ")
UDIT = Client()
t0 = Transaction("Genesis", UDIT.identity, 500.0)
block0 = Block(UDIT)
block0.previous_block_hash = ""
NONCE = None
block0.verified transactions.append(t0)
digest = hash(block0)
last block hash = digest
TPCoins = [block0]
dump blockchain(TPCoins)
```

OUTPUT:

```
Number of blocks in the chain: 1
block # 0
sender: Genesis
----
recipient:
30819f300d06092a864886f70d010101050003818d0030818902818100eca6bb5563
9066584a301764d24f95860798f3cc060ee433ea8aa72458506e8c279812b5415f5d
a25d966065b3eadd12dfa6bdd819411253bf9883a11947b7a093a52c47afe7fff165a
454859d297d02a9d0baadff621af0ceefb302183762305aa53c66c6681940f8fb9c3
05dfdd132cfb49e6d24ee01578043f591e6b28b7910203010001
----
value: 500.0
----
time: 2023-06-25 18:44:33.262135
```

[E] Create a mining function and test it.

CODE:

```
import hashlib

def sha256(message):
    return hashlib.sha256(message.encode("ascii")).hexdigest()

def mine(message, difficulty=1):
    assert difficulty >= 1
    prefix = "1" * difficulty
    for i in range(1000):
        digest = sha256(str(hash(message)) + str(i))
        if digest.startswith(prefix):
            print(f"After {str(i)} iterations found nonce: {digest}")
            return digest

print(mine("test message", 2))
```

OUTPUT:

```
After 247 iterations found nonce:
```

114ad9945d4a3b191d654352691375a6ce71606ba0e08ea737f9427b7a81d9f0 114ad9945d4a3b191d654352691375a6ce71606ba0e08ea737f9427b7a81d9f0 [F] Add blocks to the miner and dump the blockchain.

CODE:

```
import datetime
import hashlib
# Create a class with two functions
class Block:
 def init (self, data, previous hash):
    self.timestamp = datetime.datetime.now(datetime.timezone.utc)
    self.data = data
   self.previous hash = previous hash
    self.hash = self.calc hash()
 def calc hash(self):
   sha = hashlib.sha256()
   hash str = self.data.encode("utf-8")
   sha.update(hash str)
   return sha.hexdigest()
# Instantiate the class
blockchain = [Block("First block", "0")]
blockchain.append(Block("Second block", blockchain[0].hash))
blockchain.append(Block("Third block", blockchain[1].hash))
# Dumping the blockchain
for block in blockchain:
 print(
f"Timestamp: {block.timestamp}\nData: {block.data}\nPrevious Hash:
{block.previous hash} \nHash: {block.hash} \n"
  )
```

OUTPUT:

Timestamp: 2023-06-25 13:18:33.950453+00:00

Data: First block
Previous Hash: 0

Hash: 876fb923a443ba6afe5fb32dd79961e85be2b582cf74c233842b630ae16fe4d9

Timestamp: 2023-06-25 13:18:33.950453+00:00

Data: Second block

Previous Hash:

876fb923a443ba6afe5fb32dd79961e85be2b582cf74c233842b630ae16fe4d9

Hash: 8e2fb9e02898feb024dff05ee0b27fd5ea0a448e252d975e6ec5f7b0a252a6cd

Timestamp: 2023-06-25 13:18:33.950453+00:00

Data: Third block

Previous Hash:

8e2fb9e02898feb024dff05ee0b27fd5ea0a448e252d975e6ec5f7b0a252a6cd

Hash: 06e369fbfbe5362a8115a5c6f3e2d3ec7292cc4272052dcc3280898e3206208d

PRACTICAL 2

AIM: Install and configure Go Ethereum and the Mist browser. Develop and test a sample application.

Step 1: Go to Chrome Web Store Extensions Section.

Step 2: Search MetaMask.

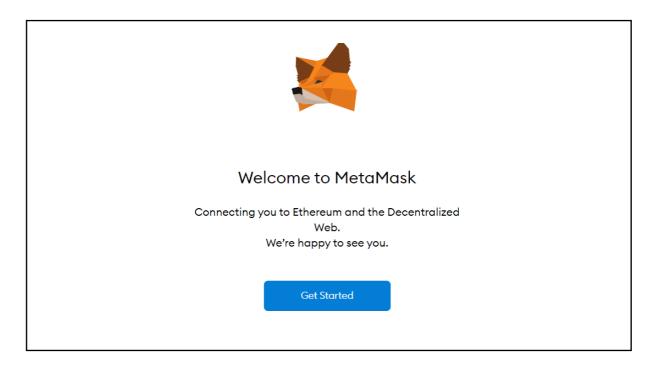
Step 3: Check the number of downloads to make sure that the legitimate MetaMask is being installed, as hackers might try to make clones of it.

Step 4: Click the Add to Chrome button.

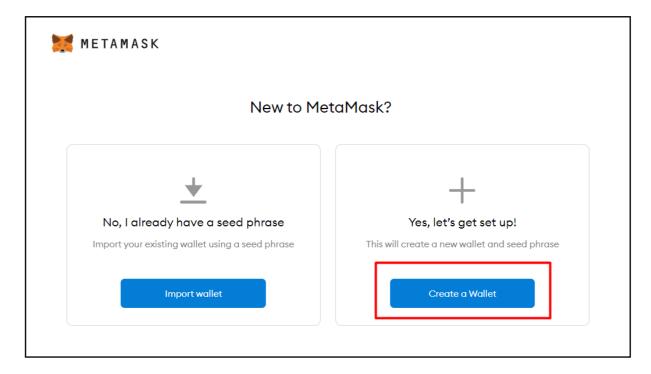


Ruchita More 23 Seat No: 40419

Step 5: Once installation is complete this page will be displayed. Click on the Get Started button.

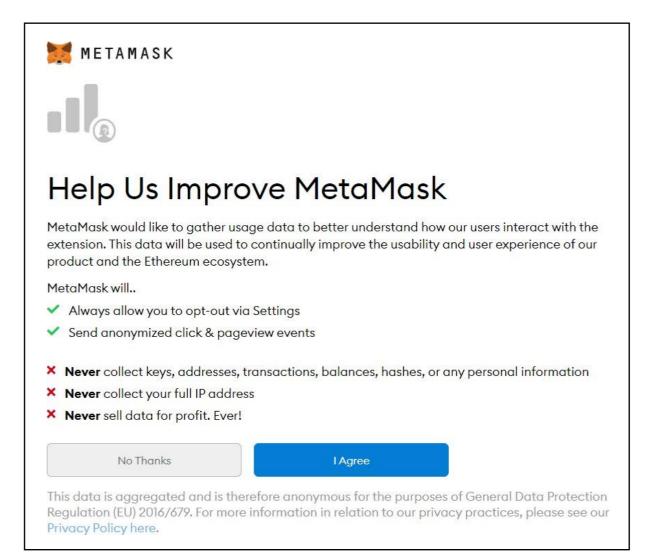


Step 6: This is the first time creating a wallet, so click the Create a Wallet button. If there is already a wallet then import the already created using the Import Wallet button.



Ruchita More 24 Seat No: 40419

Step 7: Click I Agree button to allow data to be collected to help improve MetaMask or else click the No Thanks button. The wallet can still be created even if the user will click on the No Thanks button.



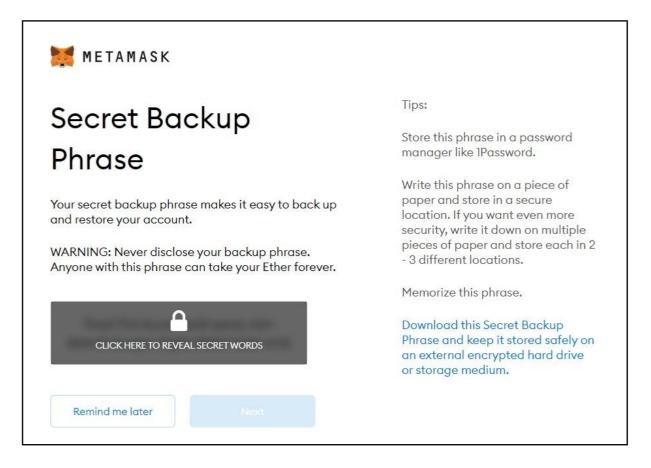
Step 8: Create a password for your wallet. This password is to be entered every time the browser is launched and wants to use MetaMask. A new password needs to be created if chrome is uninstalled or if there is a switching of browsers. In that case, go through the Import Wallet button. This is because MetaMask stores the keys in the browser. Agree to Terms of Use.

METAMASK				
< Back				
Create Password				
New password (min 8 chars)				
Confirm password				
I have read and agree to the Terms of Use				

Ruchita More 26 Seat No: 40419

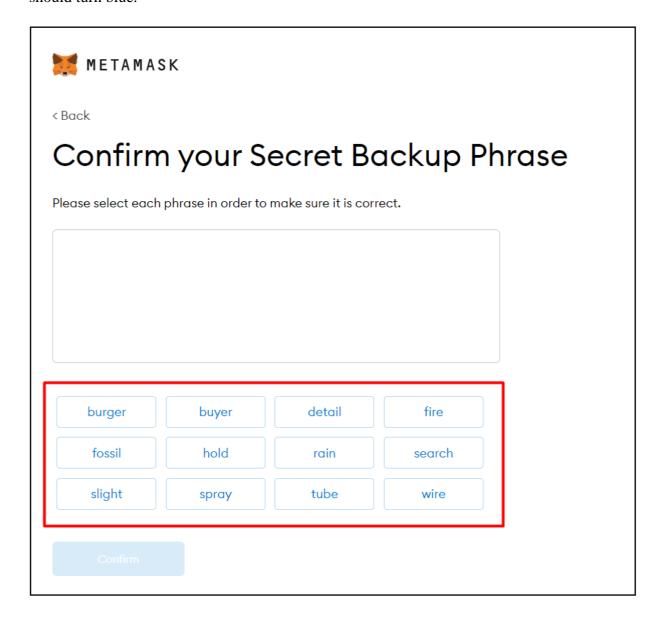
Step 9: Click on the dark area which says Click here to reveal secret words to get your secret phrase.

Step 10: This is the most important step. Back up your secret phrase properly. Do not store your secret phrase on your computer. Please read everything on this screen until you understand it completely before proceeding. The secret phrase is the only way to access your wallet if you forget your password. Once done click the Next button.



Ruchita More 27 Seat No: 40419

Step 11: Click the buttons respective to the order of the words in your seed phrase. In other words, type the seed phrase using the button on the screen. If done correctly the Confirm button should turn blue.



Step 12: Click the Confirm button. Please follow the tips mentioned.





Congratulations

You passed the test - keep your seedphrase safe, it's your responsibility!

Tips on storing it safely

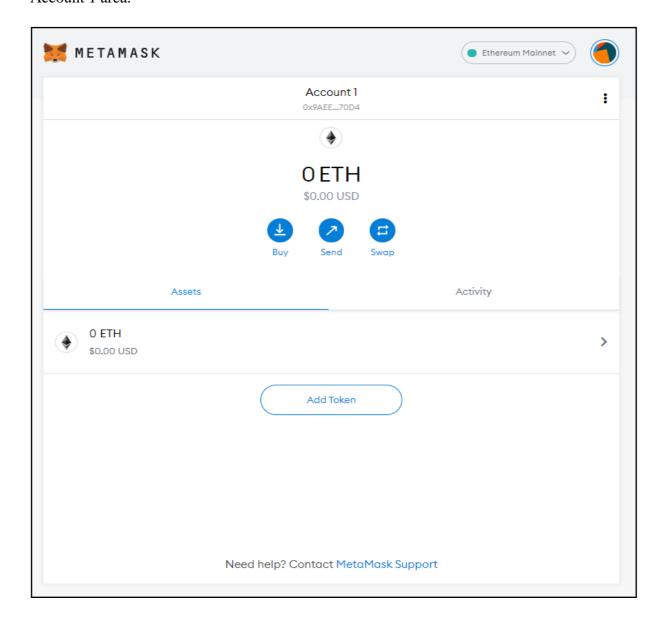
- · Save a backup in multiple places.
- · Never share the phrase with anyone.
- Be careful of phishing! MetaMask will never spontaneously ask for your seed phrase.
- If you need to back up your seed phrase again, you can find it in Settings -> Security.
- If you ever have questions or see something fishy, contact our support here.

*MetaMask cannot recover your seedphrase. Learn more.

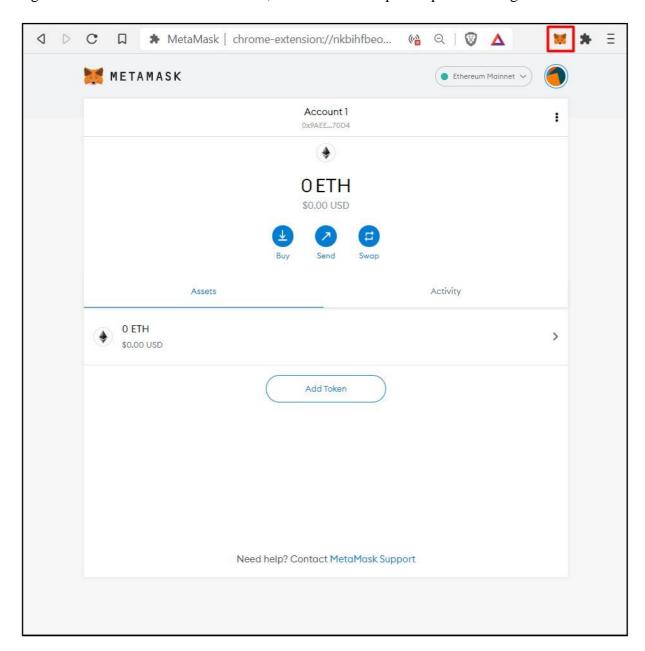
All Done

Ruchita More 29 Seat No: 40419

Step 13: One can see the balance and copy the address of the account by clicking on the Account 1 area.



Step 14: One can access MetaMask in the browser by clicking the Foxface icon on the top right. If the Foxface icon is not visible, then click on the puzzle piece icon right next to it.



Ruchita More 31 Seat No: 40419

PRACTICAL3

AIM: Implement and demonstrate the use of the following in Solidity:

- [A] Variable, Operators, Loops, Decision Making, Strings, Arrays, Enums, Structs, Mappings, Conversions, Ether Units, Special Variables.
- [I] Variable
- [i] State Variable

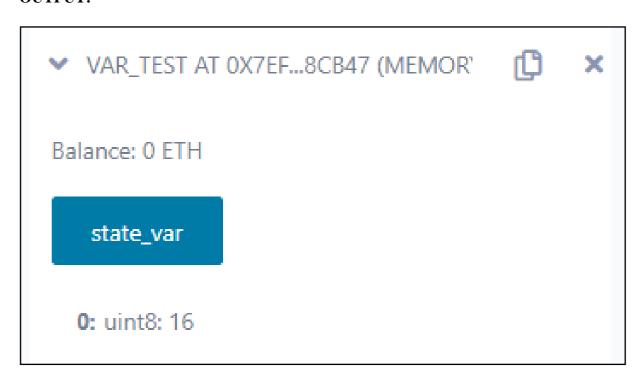
CODE:

```
//Solidity program to demonstrate state variables
pragma solidity ^0.5.0;

// Creating a contract
contract var_Test
{
    // Declaring a state variable
    uint8 public state_var;

    // Defining a constructor
    constructor() public {
        state_var = 16;
    }
}
```

OUTPUT:



[ii] Local Variable

CODE:

```
//Solidity program to demonstrate Local variables
pragma solidity ^0.5.0;
// Creating a contract
contract local_var_Test
{
    // Defining function to show the declaration and
    // scope of Local variables
    function acsess_local_variable() public pure returns(uint) {
        // Initializing Local variables
        uint a = 10;
        uint b = 40;
        uint sum = a + b;
        // Access the Local variable
        return sum;
    }
}
```

OUTPUT:

Balance: 0 ETH

acsess_local_va

0: uint256: 50

[iii] Global Variable

CODE:

```
//Solidity program to show Global variables
pragma solidity ^0.5.0;

// Creating a contract
contract globalTest
{
    // Defining a variable
    address public admin;
    // Creating a constructor to
    // use Global variable
    constructor() public
    {
        admin = msg.sender;
    }
}
```

OUTPUT:

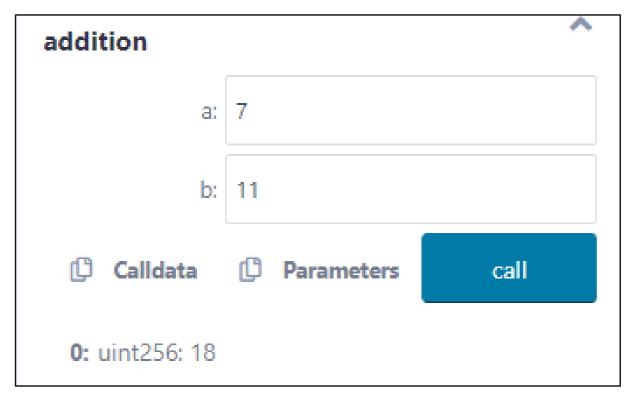
Balance: 0 ETH

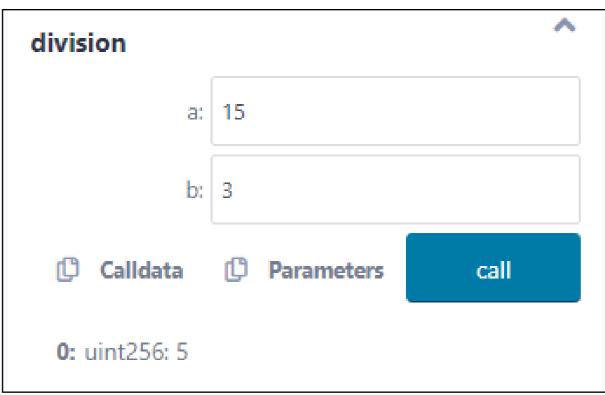
admin

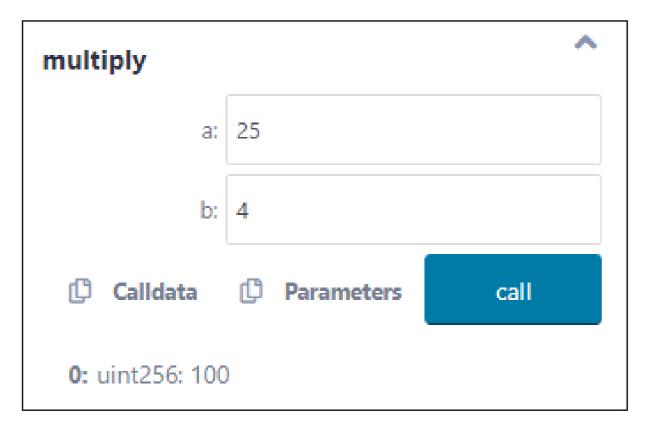
0: address: 0x5B38Da6a701c568545dCfcB03 FcB875f56beddC4

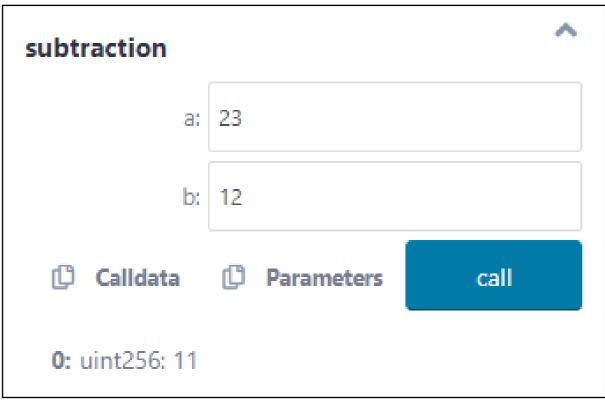
[II] Operators

```
pragma solidity ^0.4.0;
contract Operators {
   uint256 result = 0;
   function addition(uint256 a, uint256 b) public pure returns (uint256) {
       return a + b;
    }
   function subtraction (uint256 a, uint256 b) public pure returns (uint256)
{
      return a - b;
   }
   function division(uint256 a, uint256 b) public pure returns (uint256) {
      return a / b;
   }
   function multiply(uint256 a, uint256 b) public pure returns (uint256) {
       return a * b;
    }
```









[III] Loops

```
pragma solidity ^0.5.0;
contract LoopingTest {
   uint256 storedData;
   constructor() public {
       storedData = 10;
    }
    function getResult() public pure returns (string memory) {
       uint256 a = 10;
       uint256 b = 2;
       uint256 result = a + b;
       return integerToString(result);
    }
   function integerToString(uint256 _i) internal pure returns (string
memory) {
       if ( i == 0) {
          return "0";
        }
       uint256 j = 0;
       uint256 len;
        for (j = i; j != 0; j /= 10) {
           //for loop example
           len++;
       bytes memory bstr = new bytes(len);
       uint256 k = len - 1;
       while (_i != 0) {
           bstr[k--] = bytes1(uint8(48 + (i % 10)));
           i /= 10;
        }
       return string(bstr); //access local variable
    }
```







Balance: 0 ETH

getResult

0: string: 12

[IV] Decision Making

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.1;
// Creating a contract
contract Types {
   // Declaring state variables
   uint256 i = 10;
   string result;
   function decision_making() public payable returns (string memory) {
       if (i < 10) {
           result = "less than 10";
       else if (i == 10) {
           result = "equal to 10";
        }
       else {
           result = "greater than 10";
       return result;
   }
```



i: 10 uint256

result: equal to 10 string

[V] Strings

[i] Double Quotes

CODE:

```
pragma solidity ^0.5.0;

contract testString
{
   function test_string() public pure returns (string memory)
   {
     string memory a = "hello";
     return a;
   }
}
```

OUTPUT:







Balance: 0 ETH

test_string

0: string: hello

[ii] Single Quotes

```
pragma solidity ^0.5.2;
contract stringTest
   function getResult() public pure returns(string memory) {
       uint a = 25;
       uint b = 25;
       uint result = a + b;
       return integerToString(result);
    function integerToString(uint _i) internal pure returns (string memory) {
        if (i == 0)
           return "0";
        uint j = _i;
        uint len;
        while (j != 0)
           len++;
           j/= 10;
        bytes memory bstr = new bytes(len);
        uint k = len - 1;
        while ( i != 0)
           bstr[k--] = byte(uint8(48 + i % 10));
           i /= 10;
   return string(bstr);
    }
```







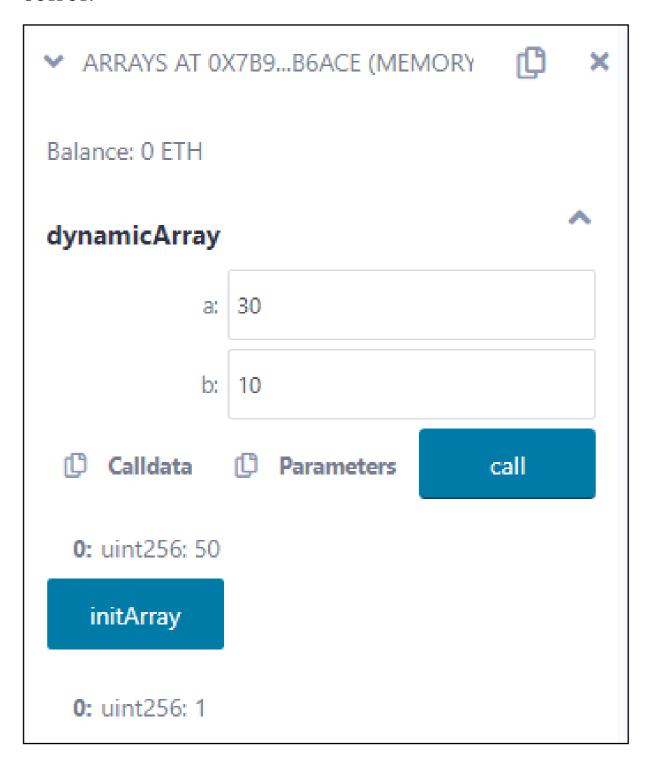
Balance: 0 ETH

getResult

0: string: 50

[VI] Arrays

```
// SPDX-License-Identifier: MIT
pragma solidity 0.7.0;
contract Arrays {
    function initArray() public pure returns (uint256) {
       uint128[3] memory array = [1, 2, uint128(3)];
       return array[0];
    }
    function dynamicArray(uint256 a, uint256 b) public pure returns (uint256)
{
       uint128[] memory array = new uint128[](a);
        uint128 val = 5;
        for (uint128 j = 0; j < a; j++) {
            array[j] = j * val;
        }
       return array[b];
    }
```



[VII] Enums

```
pragma solidity ^0.5.0;
contract enumTest {
   enum FreshJuiceSize {
       SMALL,
       MEDIUM,
       LARGE
   FreshJuiceSize choice;
   FreshJuiceSize constant defaultChoice = FreshJuiceSize.MEDIUM;
   function setLarge() public {
      choice = FreshJuiceSize.LARGE;
    }
   function getChoice() public view returns (FreshJuiceSize) {
       return choice;
    }
   function getDefaultChoice() public pure returns (uint256) {
       return uint256 (defaultChoice);
    }
```



getChoice

setLarge

0: uint8: 2

getDefaultCho

0: uint256: 1

[VIII] Structs

STEPS:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.1;
contract test {
   struct Book {
       string title;
       string author;
       uint book_id;
    }
   Book book;
   function setBook() public {
       book = Book('Learn Java', 'TP', 1);
    }
    function getBookId() public view returns (uint) {
       return book.book id;
    }
}
```



[IX] Mappings

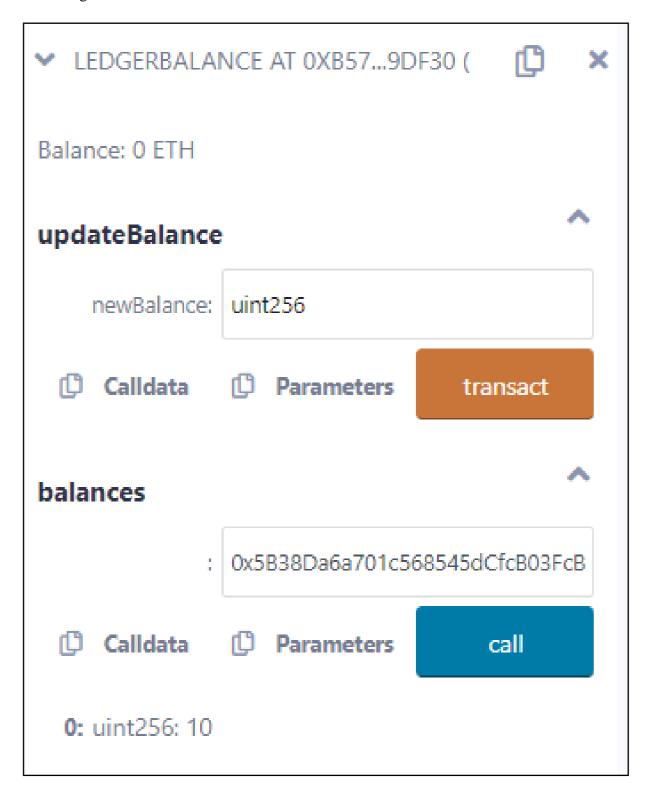
```
pragma solidity ^0.5.0;

contract LedgerBalance
{
    mapping(address => uint) public balances;

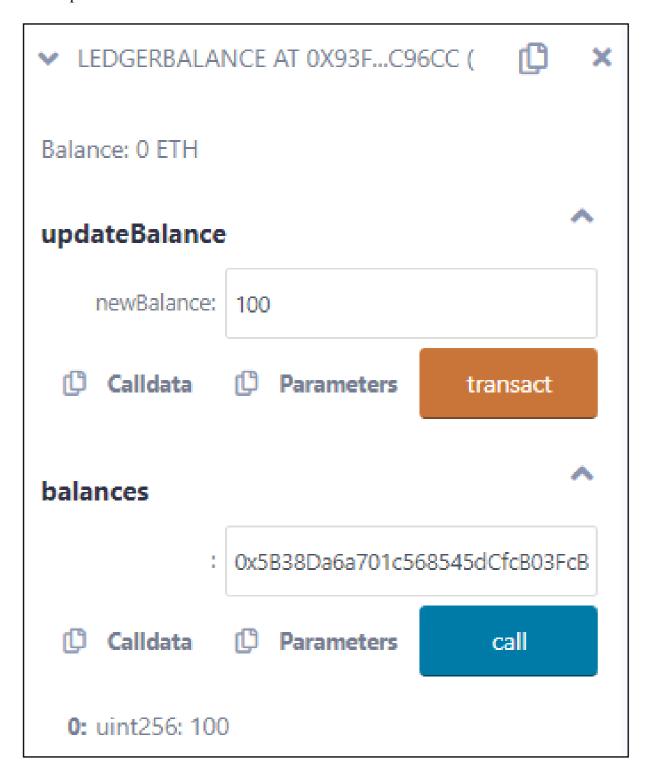
    function updateBalance(uint newBalance) public
    {
        balances[msg.sender] = newBalance;
    }
}

contract Updater
{
    function updateBalance() public returns (uint)
    {
        LedgerBalance ledgerBalance = new LedgerBalance();
        ledgerBalance.updateBalance(10);
        return ledgerBalance.balances(address(this));
    }
}
```

With Original Balance: 10



With Updated Balance: 100

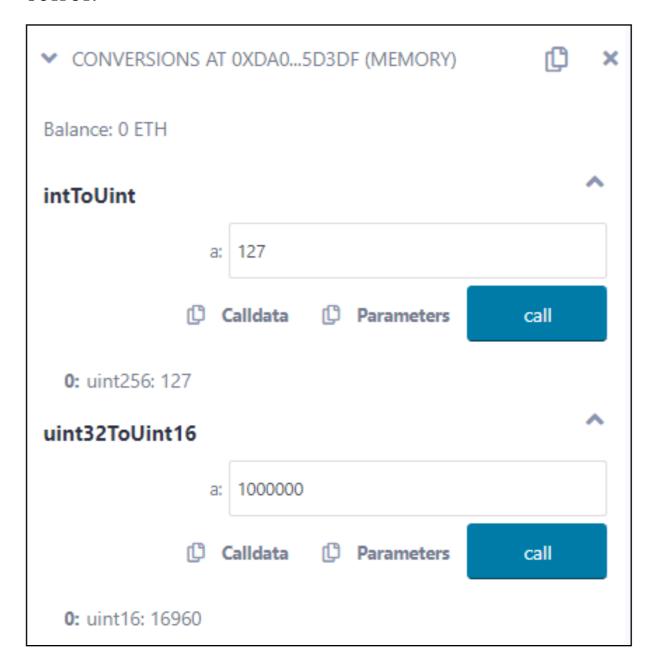


[X] Conversions

```
pragma solidity ^0.4.0;

contract Conversions {
    function intToUint(int8 a) public pure returns (uint256) {
        uint256 b = uint256(a);
        return b;
    }

    function uint32ToUint16(uint32 a) public pure returns (uint16) {
        uint16 b = uint16(a);
        return b;
    }
}
```



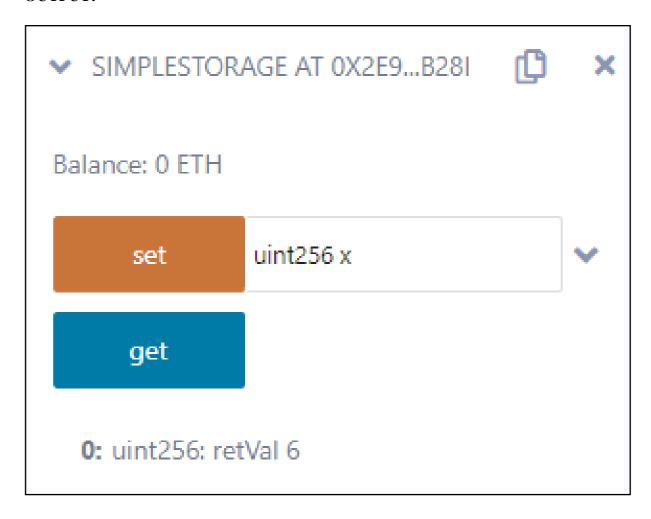
[XI] Ether Units

```
pragma solidity ^0.4.6;
contract SimpleStorage {
   uint256 storedData = 2;
   function set(uint256 x) public {
       storedData = x;
       /*
       Ether Units
       Wei
       Finney
       Szabo
       Ether
       * /
       storedData = 2;
       }
       else {
          storedData = 3;
       /*
       Time Units
       seconds
       minutes
       hours
       days
       weeks
       month
       years
       if (120 seconds == 2 minutes) {
          storedData = 6;
       }
       else {
          storedData = 9;
```

```
}

function get() constant public returns (uint256 retVal)

{
    return storedData;
}
```



[XII] Special Variables

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.1;

contract LedgerBalance {
    mapping(address => uint256) public balances;

    function updateBalance(uint256 newBalance) public {
        balances[msg.sender] = newBalance;
    }
}

contract Updater {
    function updateBalance() public returns (uint256) {
        LedgerBalance ledgerBalance = new LedgerBalance();
        ledgerBalance.updateBalance(10);
        return ledgerBalance.balances(address(this));
    }
}
```



- ▼ Function Stack 🗓

 - 0: updateBalance(newBalance) -2313 gas
- ▼ Solidity Locals
 - newBalance: 1000 uint256

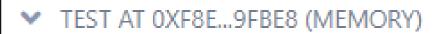
- ▼ Solidity State

 - balances:
 - mapping(address => uint256)
- ▼ Step details

vm trace step: 114 execution step: 114 [B] Functions, Function Modifiers, View Functions, Pure Functions, Fallback Function, Function Overloading, Mathematical functions, Cryptographic functions.

[I] Functions

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;
contract Test {
 function return example()
   public
   pure
   returns (
     uint256,
     uint256,
     uint256,
      string memory
   )
  {
   uint256 num1 = 10;
   uint256 num2 = 16;
   uint256 sum = num1 + num2;
   uint256 prod = num1 * num2;
   uint256 diff = num2 - num1;
   string memory message = "Multiple return values";
   return (sum, prod, diff, message);
  }
```







Balance: 0 ETH

return_example

0: uint256: 26

1: uint256: 160

2: uint256: 6

3: string: Multiple return values

[II] Function Modifiers

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

contract ExampleContract {
   address public owner = 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4;
   uint256 public counter;

   modifier onlyowner() {
     require(msg.sender == owner, "Only the contract owner can call");
     _;
   }

   function incrementcounter() public onlyowner {
     counter++;
   }
}
```

incrementcour

counter

0: uint256: 1

owner

0: address: 0x5B38Da6a701c568545dCfcB0 3FcB875f56beddC4

[III] View Functions

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;
contract view_demo {
 uint256 num1 = 2;
 uint256 num2 = 4;
 function getResult() public view returns (uint256 product, uint256 sum) {
   product = num1 * num2;
   sum = num1 + num2;
 }
```

OUTPUT:







Balance: 0 ETH

getResult

0: uint256: product 8

1: uint256: sum 6

[IV] Pure Functions

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;
contract pure demo {
 function getResult() public pure returns (uint256 product, uint256 sum) {
   uint256 num1 = 2;
   uint256 num2 = 4;
   product = num1 * num2;
   sum = num1 + num2;
 }
```

OUTPUT:







Balance: 0 ETH

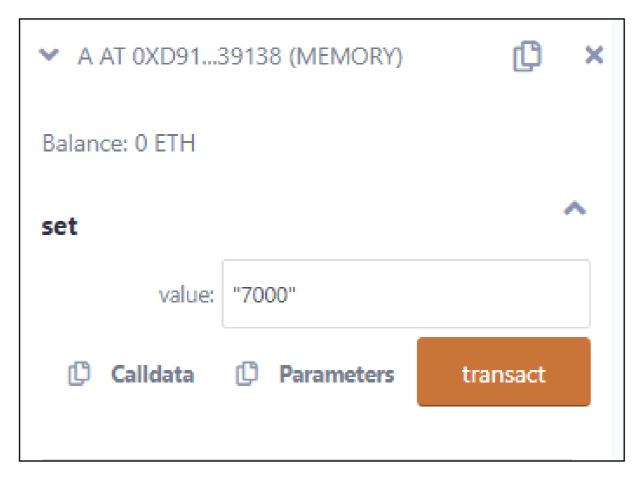
getResult

0: uint256: product 8

1: uint256: sum 6

[V] Fallback Function

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;
contract A {
 uint256 n;
 function set(uint256 value) external {
   n = value;
 }
 function() external payable {
   n = 0;
 }
}
contract example {
 function callA(A a) public returns (bool) {
    (bool success, ) = address(a).call(abi.encodeWithSignature("setter()"));
   require(success);
   address payable payableA = address(uint160(address(a)));
   return (payableA.send(2 ether));
 }
```



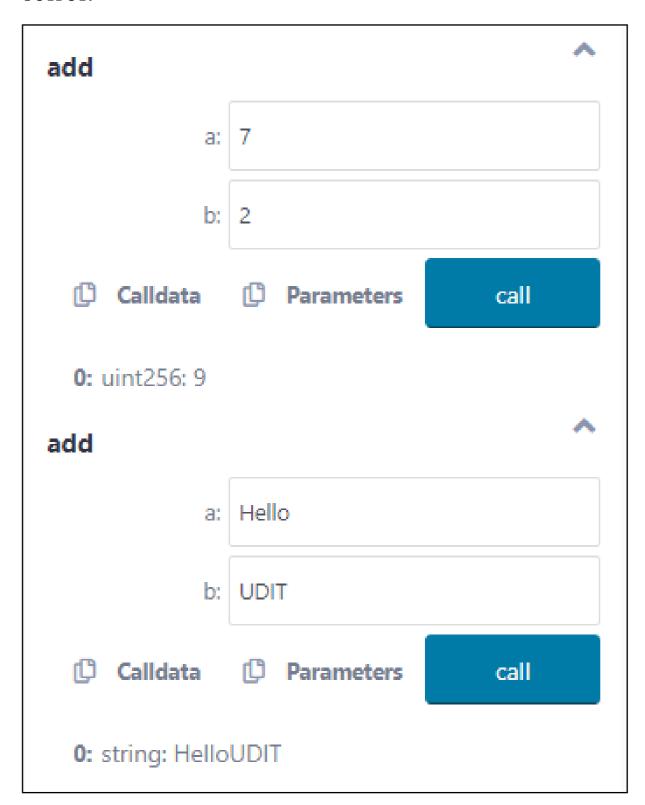


[VI] Function Overloading

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

contract OverloadingExample {
  function add(uint256 a, uint256 b) public pure returns (uint256) {
    return a + b;
  }

  function add(string memory a, string memory b)
    public
    pure
    returns (string memory)
    {
      return string(abi.encodePacked(a, b));
    }
}
```



[VII] Mathematical Functions

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

contract Test{ function CallAddMod() public pure returns(uint){
    return addmod(7,3,3);
}

function CallMulMod() public pure returns(uint){
    return mulmod(7,3,3);
    }
}
```

OUTPUT:

CallAddMod

0: uint256: 1

CallMulMod

0: uint256: 0

[VIII] Cryptographic Functions

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;
contract Test{ function callKeccak256() public pure returns(bytes32 result){
   return keccak256("BLOCKCHAIN");
}
function callsha256() public pure returns(bytes32 result){
   return sha256("BLOCKCHAIN");
}
function callripemd() public pure returns (bytes20 result){
   return ripemd160("BLOCKCHAIN");
   }
}
```



TEST AT 0X0FC...9A836 (MEMORY)





Balance: 0 ETH

callKeccak256

0: bytes32: result 0xedb146af3dfbb7f995ec6 5b249dd88d2d54ca0707a84f08c25e4cc0 8f5168aea

callripemd

0: bytes20: result 0x638cf4481022e8be8fab4 3fa5f76ccffc62f2a09

callsha256

0: bytes32: result 0xdffdca1f7dd5c94afea29 36253a2463a26aad06fa9b5f36b5affc8851 e8c8d42

PRACTICAL4

AIM: Implement and demonstrate the use of the following in Solidity:

[A] Withdrawal Pattern, Restricted Access.

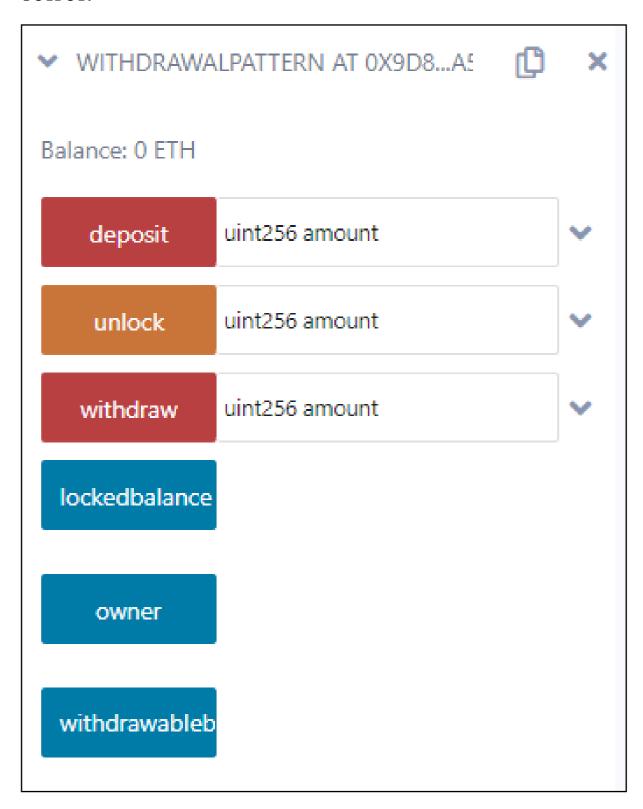
[I] Withdrawal Pattern

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;
contract WithdrawalPattern {
 address public owner;
 uint256 public lockedbalance;
 uint256 public withdrawablebalance;
 constructor() {
   owner = msg.sender;
  }
 modifier onlyowner() {
   require(msg.sender == owner, "Only the owner can call this
function");
 function deposit(uint256 amount) public payable {
   require(amount > 0, "Amount must be greater than zero");
   lockedbalance += amount;
  }
  function withdraw(uint256 amount) public payable onlyowner {
```

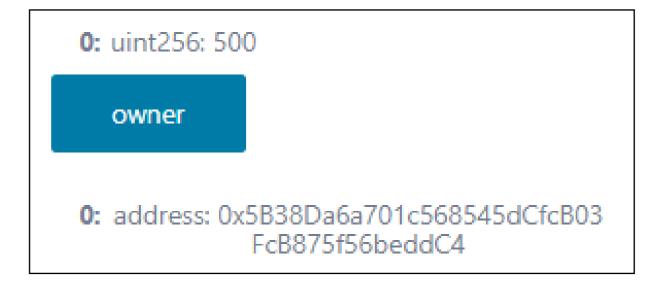
```
require(
    amount <= withdrawablebalance,
    "Insufficient withdrawable balance"
);
    withdrawablebalance -= amount;
    payable(msg.sender).transfer(amount);
}

function unlock(uint256 amount) public onlyowner {
    require(amount <= lockedbalance, "Insufficient locked balance");
    lockedbalance -= amount;
    withdrawablebalance += amount;
}</pre>
```

Ruchita More 75 Seat No: 40419



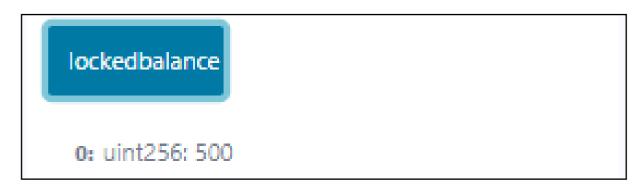
Step 1: Click on owner to create the owner object.



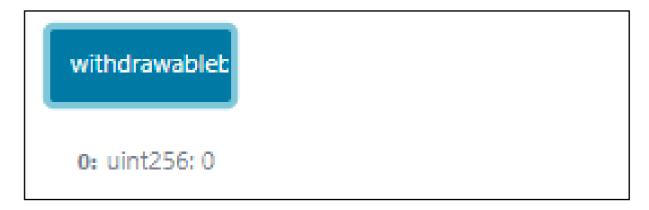
Step 2: Enter an amount and click on deposit.



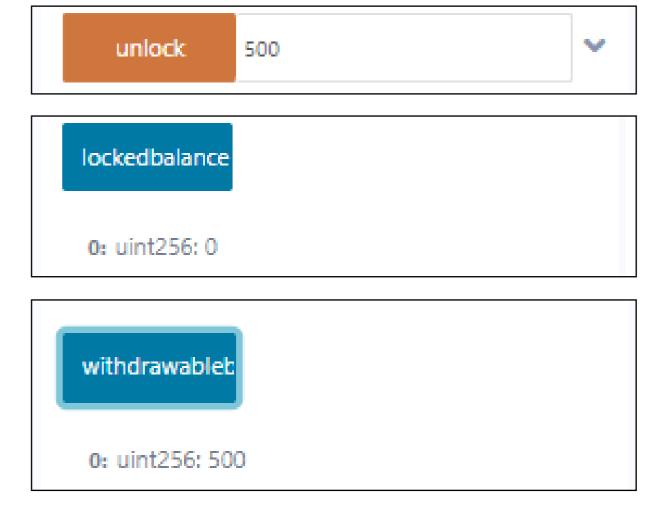
Step 3: Click on locked balance button to display the locked amount in the account.



Step 4: Click on withdrawable balance button.

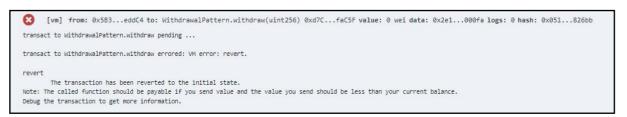


Step 5: Click on unlock button and enter any amount to transfer amount to withdrawable balance. Check locked balance and withdrawable balance.



Step 6: Enter any amount you want to withdraw and click the withdraw button. You should get an error and the transaction should be reverted.





[II] Restricted Access

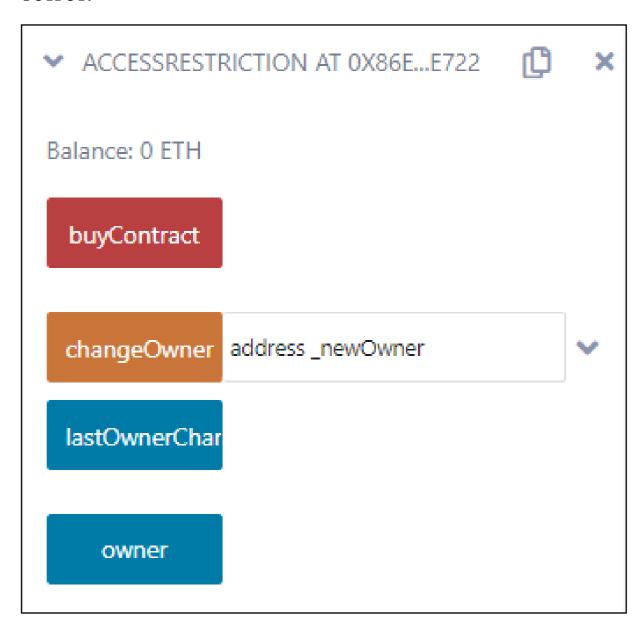
```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;
contract AccessRestriction {
    address public owner = msg.sender;
    uint public lastOwnerChange = now;
   modifier onlyBy(address account) {
        require(msg.sender == account);
        _;
    modifier onlyAfter(uint _time) {
       require(now >= _time);
        _;
    }
    modifier costs(uint amount) {
        require(msg.value >= amount);
        _;
        if (msg.value > amount) {
           msg.sender.transfer(msg.value - amount);
        }
    }
    function changeOwner(address _newOwner) public onlyBy(owner) {
        owner = newOwner;
    }
    function buyContract() public payable onlyAfter(lastOwnerChange +
4 weeks) costs(1 ether) {
```

UDIT, M.Sc. IT SEM IV

BLOCKCHAIN JOURNAL (2022-23)

```
owner = msg.sender;
lastOwnerChange = now;
}
```

Ruchita More 81 Seat No: 40419



Step 1: Click on owner to create the owner object.



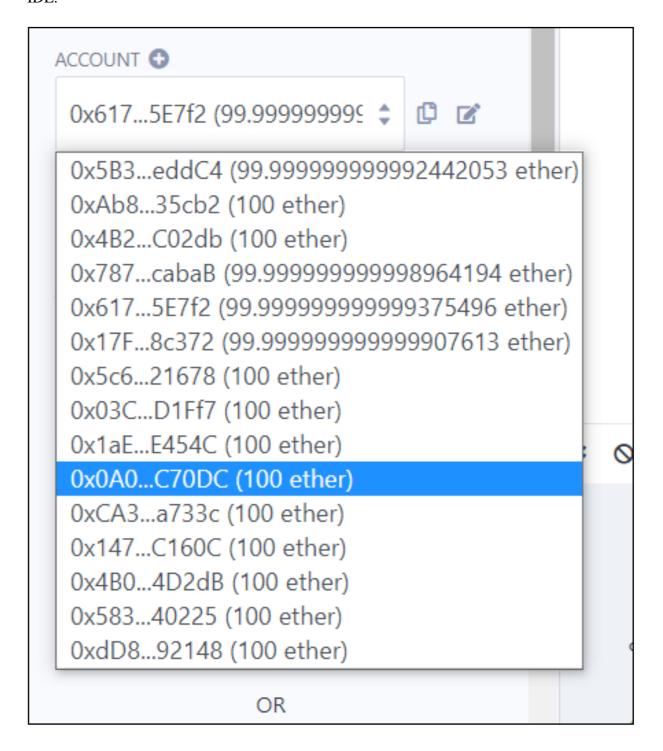
0: address: 0x617F2E2fD72FD9D550319709 2aC168c91465E7f2

Step 2: Click on lastOwnerChange button.

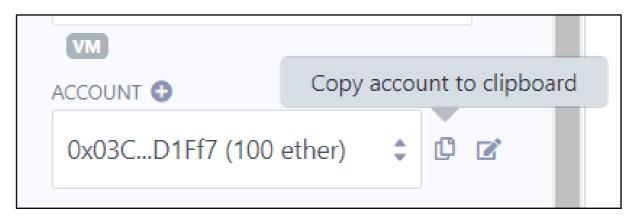
lastOwnerChar

0: uint256: 1687627628

Step 3: Change the address of the account from Account dropdown in Deploy tab of Remix IDE.



Step 4: Copy and paste the address in changeOwner input and click on changeOwner.





Step 5: You should get an error as following.



Step 6: Now, change back to the actual address of the account and click on changeOwner.



owner

0: address: 0x03C6FcED478cBbC9a4FAB34eF 9f40767739D1Ff7

[B] Contracts, Inheritance, Constructors, Abstract Contracts, Interfaces.

[I] Contracts

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

contract Contract_demo {
   string message = "Hello";

function dispMsg() public view returns (string memory) {
   return message;
   }
}
```

OUTPUT:



Balance: 0 ETH

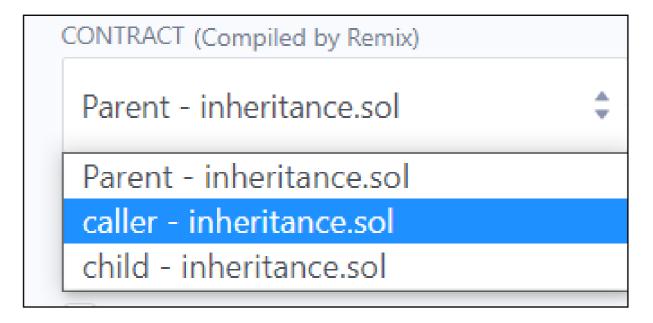
dispMsg

0: string: Hello

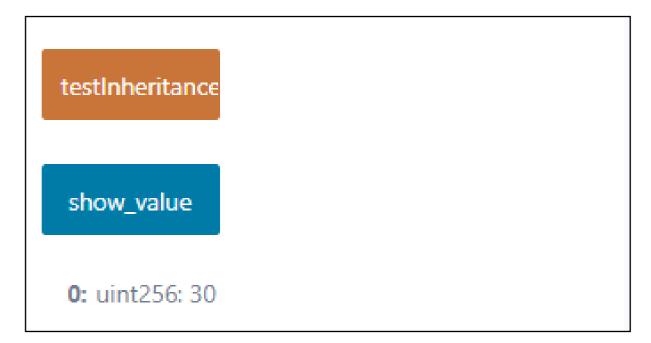
[II] Inheritance

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;
contract Parent {
 uint256 internal sum;
 function setValue() external {
   uint256 a = 10;
   uint256 b = 20;
   sum = a + b;
 }
}
contract child is Parent {
 function getValue() external view returns (uint256) {
   return sum;
 }
}
contract caller {
 child cc = new child();
 function testInheritance() public returns (uint256) {
   cc.setValue();
   return cc.getValue();
 }
 function show value() public view returns (uint256) {
   return cc.getValue();
  }
```

Step 1: Select caller contract and deploy.



Step 2: Click test Inheritance and then click on show_value to view value.



[III] Constructors

CODE:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;
// Creating a contract
contract constructorExample {
 string str;
 constructor() public {
   str = "UDIT";
  }
 function getValue() public view returns (string memory) {
   return str;
  }
```

OUTPUT:





Balance: 0 ETH

getValue

0: string: UDIT

[IV] Abstract Contracts

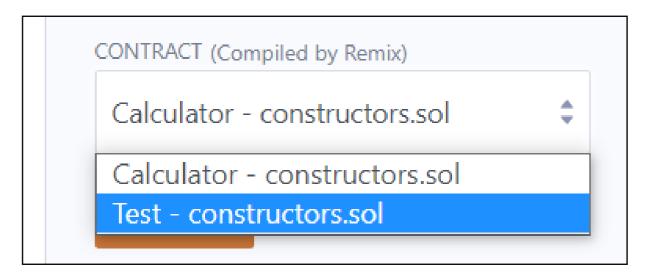
```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

contract Calculator {
  function getResult() external view returns (uint256);
}

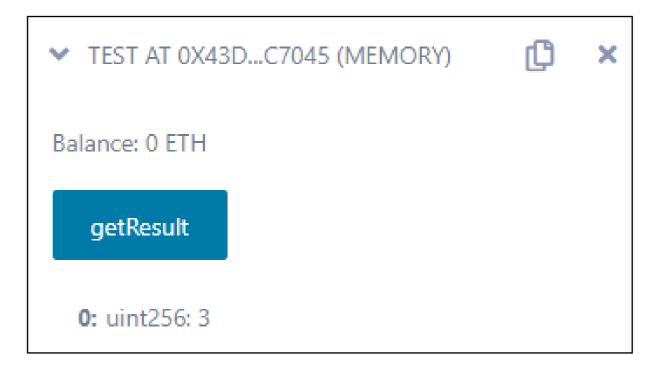
contract Test is Calculator {
  constructor() public {}

  function getResult() external view returns (uint256) {
    uint256 a = 1;
    uint256 b = 2;
    uint256 result = a + b;
    return result;
  }
}
```

Step 1: Select Test contract and deploy.



Step 2: Click on getResult to get sum of a+b.



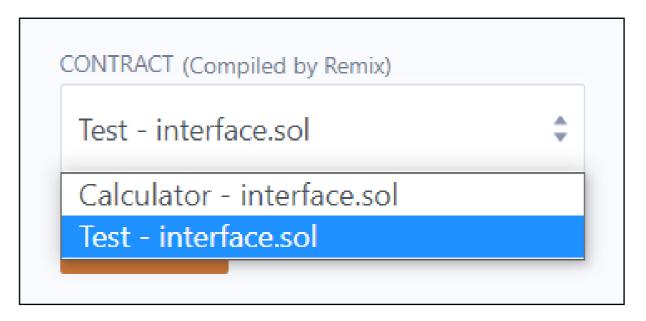
[V] Interfaces

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

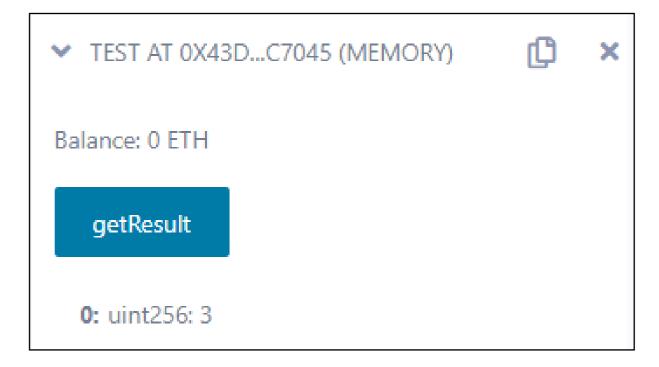
interface Calculator {
  function getResult() external view returns(uint);
}

contract Test is Calculator {
  constructor() public {}
  function getResult() external view returns(uint) {
    uint a = 1;
    uint b = 2;
    uint result = a + b;
    return result;
  }
}
```

Step 1: Select Test interface contract and deploy.



Step 2: Click on getResult to get sum of a+b.



[C] Libraries, Assembly, Events, Error handling.

[I] Libraries

CODE:

Create library myLib.sol

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

library myMathLib {
  function sum(uint256 a, uint256 b) public pure returns (uint256) {
    return a + b;
  }

  function exponent(uint256 a, uint256 b) public pure returns
(uint256) {
    return a**b;
  }
}
```

Using the library myLib.sol

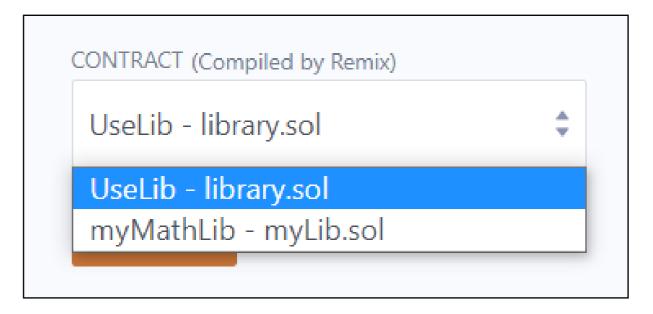
```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;
import "myLib.sol";

contract UseLib {
  function getsum(uint256 x, uint256 y) public pure returns (uint256)
  {
    return myMathLib.sum(x, y);
  }

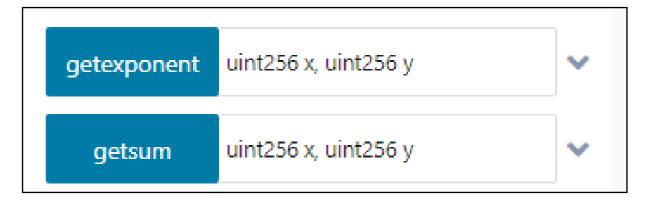
  function getexponent(uint256 x, uint256 y) public pure returns
  (uint256) {
```

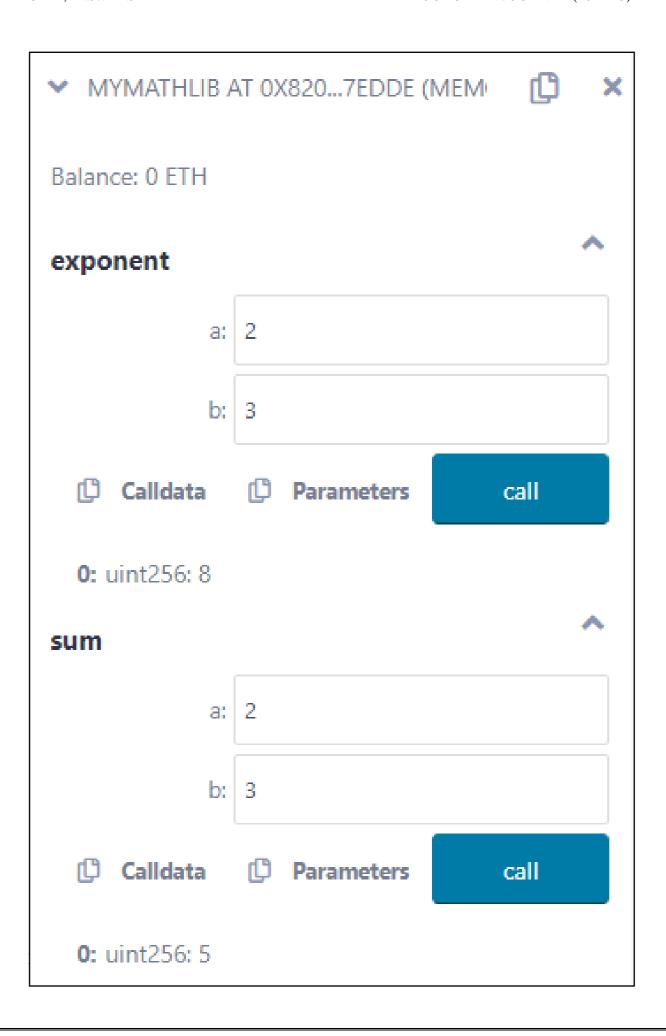
```
return myMathLib.exponent(x, y);
}
```

Step 1: Change contract to UseLib and deploy.



Step 2: Input values to both getexponent and getsum functions and get their values as below.

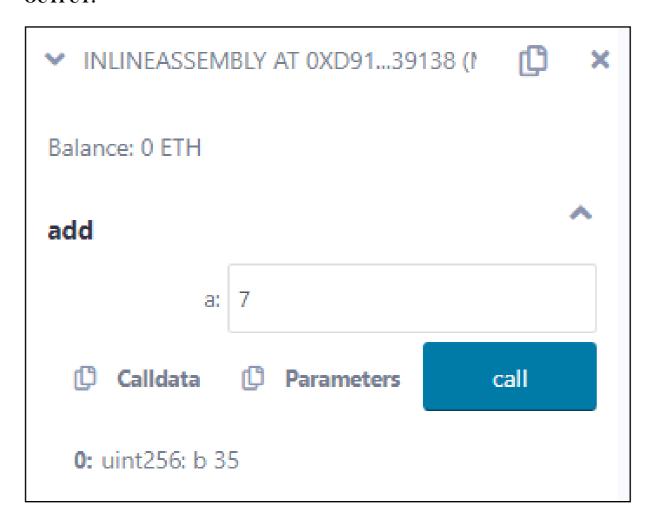




[II] Assembly

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.4.16 <0.9.0;

contract InlineAssembly {
    // Defining function
    function add(uint256 a) public view returns (uint256 b) {
        assembly {
        let c := add(a, 16)
            mstore(0x80, c)
        {
            let d := add(sload(c), 12)
            b := d
        }
        b := add(b, c)
    }
}</pre>
```



[III] Events

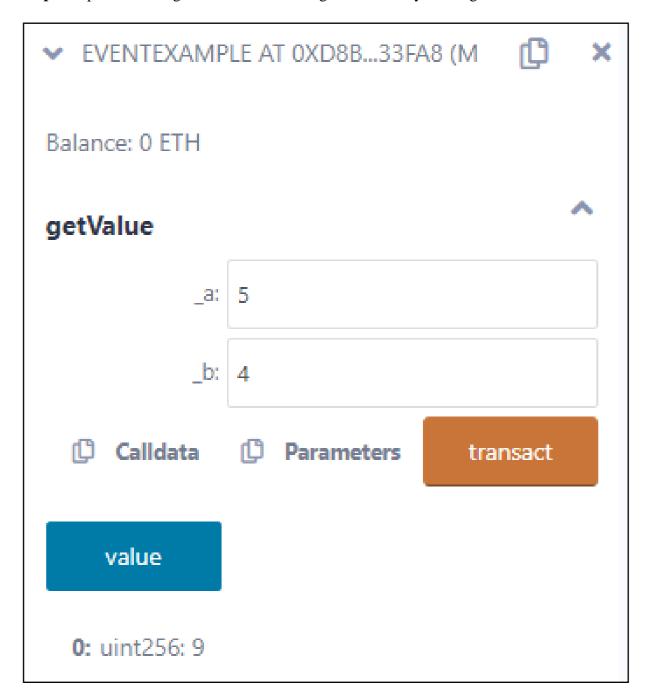
```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

// Creating a contract
contract eventExample {
    // Declaring state variables
    uint256 public value = 0;

    // Declaring an event
    event Increment(address owner);

    // Defining a function for logging event
    function getValue(uint256 _a, uint256 _b) public {
        emit Increment(msg.sender);
        value = _a + _b;
    }
}
```

Step 1: Input values to getValue function and get the result by clicking on the value button.



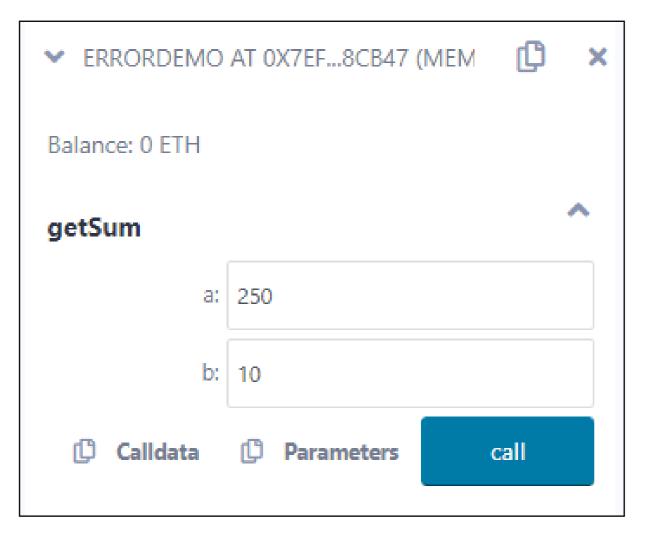
Step 2: In the terminal, check for logs.

[IV] Error handling

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

contract ErrorDemo {
  function getSum(uint256 a, uint256 b) public pure returns (uint256)
  {
    uint256 sum = a + b;
    require(sum < 255, "Invalid");
    assert(sum<255);
    return sum;
  }
}</pre>
```

Step 1: Input values to the getSum function.



Step 2: Check terminal panel.

```
[call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: ErrorDemo.getSum(uint256,uint256) data: 0x8e8...0000a

call to ErrorDemo.getSum errored: VM error: revert.

The transaction has been reverted to the initial state.

Reason provided by the contract: "Invalid".

Debug the transaction to get more information.
```

PRACTICAL5

AIM: Install hyperledger fabric and composer. Deploy and execute the application.

Step 1: Install Docker on Ubuntu

sudo apt-get install curl

```
udit@udit-ubuntu: ~
                                                           Q
udit@udit-ubuntu:~$ sudo apt-get install curl
[sudo] password for udit:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libcurl4
The following NEW packages will be installed:
 curl
The following packages will be upgraded:
 libcurl4
1 upgraded, 1 newly installed, 0 to remove and 149 not upgraded.
Need to get 194 kB/484 kB of archives.
After this operation, 459 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://security.ubuntu.com/ubuntu jammy-security/main amd64 curl amd64 7.
81.0-1ubuntu1.10 [194 kB]
Fetched 194 kB in 1s (215 kB/s)
(Reading database ... 161313 files and directories currently installed.)
Preparing to unpack .../libcurl4_7.81.0-1ubuntu1.10_amd64.deb ...
Unpacking libcurl4:amd64 (7.81.0-1ubuntu1.10) over (7.81.0-1ubuntu1.7) ...
Selecting previously unselected package curl.
Preparing to unpack .../curl_7.81.0-1ubuntu1.10_amd64.deb ...
Unpacking curl (7.81.0-1ubuntu1.10) ...
Setting up libcurl4:amd64 (7.81.0-1ubuntu1.10) ...
Setting up curl (7.81.0-1ubuntu1.10) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for libc-bin (2.35-Oubuntu3.1) ...
udit@udit-ubuntu:~$
```

curl -fSSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add –

```
udit@udit-ubuntu:~$ curl -fSSL https://download.docker.com/linux/ubuntu/gpg | s
udo apt-key add
                                                          Time Current
 % Total
           % Received % Xferd Average Speed
                                           Time
                                                  Time
                             Dload Upload
                                                          Left Speed
                                           Total
                                                  Spent
                0
                     0
                          0
                                      0 --:--:--
           0
                                0
arning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (s
ee apt-key(8)).
100 3817 100 3817
                          0 43852
                                      0 --:--:- 44383
                     0
OK
udit@udit-ubuntu:~$
```

```
sudo add-apt-repository \
  "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) \
  stable"
```

```
udit@udit-ubuntu:~$ sudo add-apt-repository \
   "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
   $(lsb_release -cs) \
   stable"
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu jammy st
able'
Description:
Archive for codename: jammy components: stable
More info: https://download.docker.com/linux/ubuntu
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.
Adding deb entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_c
om_linux_ubuntu-jammy.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive_uri-https_down
load_docker_com_linux_ubuntu-jammy.list
Get:1 https://download.docker.com/linux/ubuntu jammy InRelease [48.9 kB]
Get:2 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages [19.
2 kB]
Get:3 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:4 http://in.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:5 http://security.ubuntu.com/ubuntu jammy-security/main amd64 DEP-11 Metada
ta [41.5 kB]
Get:6 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 DEP-11 Me
tadata [21.9 kB]
```

sudo apt-get update

```
udit@udit-ubuntu:~$ sudo apt-get update
Hit:1 https://download.docker.com/linux/ubuntu jammy InRelease
Hit:2 http://in.archive.ubuntu.com/ubuntu jammy InRelease
Hit:3 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:4 http://in.archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:5 http://in.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:6 http://in.archive.ubuntu.com/ubuntu jammy-updates/main i386 Packages [430
Get:7 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [72
4 kB]
Get:8 http://in.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [19
0 kB]
Get:9 http://in.archive.ubuntu.com/ubuntu jammy-updates/main amd64 DEP-11 Metad
ata [99.4 kB]
Get:10 http://in.archive.ubuntu.com/ubuntu jammy-updates/main DEP-11 48x48 Icon
s [33.0 kB]
Get:10 http://in.archive.ubuntu.com/ubuntu jammy-updates/main DEP-11 48x48 Icon
s [33.0 kB]
Get:10 http://in.archive.ubuntu.com/ubuntu jammy-updates/main DEP-11 48x48 Icon
s [33.0 kB]
Get:10 http://in.archive.ubuntu.com/ubuntu jammy-updates/main DEP-11 48x48 Icon
s [33.0 kB]
```

apt-cache policy docker-ce

```
udit@udit-ubuntu:~$ apt-cache policy docker-ce
docker-ce:
  Installed: (none)
  Candidate: 5:24.0.2-1~ubuntu.22.04~jammy
  Version table:
     5:24.0.2-1~ubuntu.22.04~jammy 500
        500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Package
s
     5:24.0.1-1~ubuntu.22.04~jammy 500
        500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Package
s
     5:24.0.0-1~ubuntu.22.04~jammy 500
        500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Package
s
     5:23.0.6-1~ubuntu.22.04~jammy 500
        500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Package
s
     5:23.0.5-1~ubuntu.22.04~jammy 500
        500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Package
s
     5:23.0.4-1~ubuntu.22.04~jammy 500
        500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Package
s
     5:23.0.3-1~ubuntu.22.04~jammy 500
        500 https://download.docker.com/linux/ubuntu jammy/stable amd64 Package
```

sudo apt-get install -y docker-ce

```
udit@udit-ubuntu:~$ sudo apt-get install -y docker-ce
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  containerd.io docker-buildx-plugin docker-ce-cli docker-ce-rootless-extras
  docker-compose-plugin git git-man liberror-perl libslirp0 pigz slirp4netns
Suggested packages:
  aufs-tools cgroupfs-mount | cgroup-lite git-daemon-run
  | git-daemon-sysvinit git-doc git-email git-gui gitk gitweb git-cvs
  git-mediawiki git-svn
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce docker-ce-cli
  docker-ce-rootless-extras docker-compose-plugin git git-man liberror-perl
  libslirp0 pigz slirp4netns
0 upgraded, 12 newly installed, 0 to remove and 245 not upgraded.
Need to get 115 MB of archives.
After this operation, 422 MB of additional disk space will be used.
Get:1 https://download.docker.com/linux/ubuntu jammy/stable amd64 containerd.io
amd64 1.6.21-1 [28.3 MB]
Get:2 http://in.archive.ubuntu.com/ubuntu jammy/universe amd64 pigz amd64 2.6-1
[63.6 kB]
```

Step 2: Create a docker network

sudo docker network create udit-iroha-network

udit@udit-ubuntu:~\$ sudo docker network create udit-iroha-network
508883d5de27b2b369c31cd428acc669783ca7eb6536eeaebcc77f324a7e4dec

```
Step 3: Add PostgreSQL to the created docker network.
```

```
sudo docker run --name some-postgres \
-e POSTGRES_USER=postgres \
-e POSTGRES_PASSWORD=mysecretpassword \
-p 5432:5432 \
--network=udit-iroha-network \
-d postgres:9.5
```

```
udit@udit-ubuntu:~$ sudo docker run --name some-postgres \
-e POSTGRES_USER=postgres \
-e POSTGRES_PASSWORD=mysecretpassword \
-p 5432:5432 \
--network=udit-iroha-network \
-d postgres:9.5
Unable to find image 'postgres:9.5' locally
9.5: Pulling from library/postgres
fa1690ae9228: Pull complete
a73f6e07b158: Pull complete
973a0c44ddba: Pull complete
07e5342b01d4: Pull complete
578aad0862c9: Pull complete
a0b157088f7a: Pull complete
6c9046f06fc5: Pull complete
ae19407bdc48: Pull complete
e53b7c20aa96: Pull complete
a135edcc0831: Pull complete
fed07b1b1b94: Pull complete
18d9026fcfbd: Pull complete
4d2d5fae97d9: Pull complete
d419466e642d: Pull complete
Digest: sha256:75ebf479151a8fd77bf2fed46ef76ce8d518c23264734c48f2d1de42b4eb40ae
Status: Downloaded newer image for postgres:9.5
b7e9907af8e45279b3ee8ba8cf96ed73945fec76c5fb32c3d6464f8196c84b2f
```

Step 4: Create a volume of persistant storage named "blockstore" to store the blocks on blockchain.

sudo docker volume create blockstore

```
udit@udit-ubuntu:~$ sudo docker volume create blockstore
blockstore
```

Step 5: Configure Iroha on the network. Download the Iroha code from github and install git. sudo apt-get install git

```
udit@udit-ubuntu:~$ sudo apt-get install git
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
git is already the newest version (1:2.34.1-1ubuntu1.9).
git set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 245 not upgraded.
```

git clone -b develop https://github.com/hyperledger/iroha --depth=1

```
udit@udit-ubuntu:~$ git clone -b develop https://github.com/hyperledger/iroha -
-depth=1
Cloning into 'iroha'...
remote: Enumerating objects: 2037, done.
remote: Counting objects: 100% (2037/2037), done.
remote: Compressing objects: 100% (1772/1772), done.
remote: Total 2037 (delta 433), reused 784 (delta 215), pack-reused 0
Receiving objects: 100% (2037/2037), 15.84 MiB | 9.03 MiB/s, done.
Resolving deltas: 100% (433/433), done.
```

Step 6: Run the Iroha docker container.

```
sudo docker run -it --name iroha \
-p 50051:50051 \
-v $(pwd)/iroha/example:/opt/iroha_data \
-v blockstore:/tmp/block_store \
--network=udit-iroha-network \
--entrypoint=/bin/bash \
hyperledger/iroha:latest
```

```
udit@udit-ubuntu:~$ sudo docker run -it --name iroha \
-p 50051:50051 \
-v $(pwd)/iroha/example:/opt/iroha_data \
-v blockstore:/tmp/block_store \
--network=udit-iroha-network \
--entrypoint=/bin/bash \
hyperledger/iroha:latest
Unable to find image 'hyperledger/iroha:latest' locally
latest: Pulling from hyperledger/iroha
e0b25ef51634: Pull complete
e1300e501129: Pull complete
a3124d67b376: Pull complete
657a8a434a5c: Pull complete
ffcf37f233dc: Pull complete
cd4d37888857: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:d3382063b858070786e13811e8cdcd8d7f0215b84a4444e0ea9c134ed662adc8
Status: Downloaded newer image for hyperledger/iroha:latest
root@951ab987e810:/opt/iroha_data#
```

Step 7: Run Iroha.

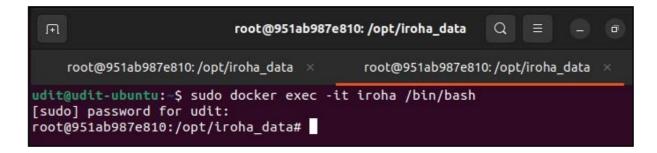
irohad --config config.docker \

- --genesis_block genesis.block \
- --keypair_name node0

```
root@951ab987e810:/opt/iroha data# irohad --config config.docker \
 --genesis block genesis.block \
> --keypair name node0
[2023-06-23 19:32:50.446019430][I][Init]: Irohad version: 1.5.0
[2023-06-23 19:32:50.446059998][I][Init]: config initialized
[2023-06-23 19:32:50.446477847][I][Irohad]: created
[2023-06-23 19:32:50.446503056][I][Irohad]: [Init] => pending transactions stor
age
WARNING:
         hash indexes are not WAL-logged and their use is discouraged
WARNING: hash indexes are not WAL-logged and their use is discouraged
[2023-06-23 19:32:50.782608193][I][Irohad]: [Init] => storage
[2023-06-23 19:32:50.783609295][I][Irohad/Storage/Storage]: drop block storage
[2023-06-23 19:32:50.784025818][I][Irohad]: Recreating schema.
WARNING: hash indexes are not WAL-logged and their use is discouraged
WARNING: hash indexes are not WAL-logged and their use is discouraged
[2023-06-23 19:32:51.218452015][I][Irohad]: [Init] => storage
[2023-06-23 19:32:51.218482842][I][Irohad/Storage/Storage]: create mutable stor
[2023-06-23 19:32:51.240896513][I][Irohad/Storage/Storage/MutableStorageImpl]:
Applying block: height 1, hash 9debdb1a70db2cede2222427b849f6bf7ab20845da7c3db1
837c0df25ec1c61a
[2023-06-23 19:32:51.251196301][I][Init]: Genesis block inserted, number of tra
nsactions: 1
[2023-06-23 19:32:51.251845525][I][Irohad/Storage/Storage/PostgresSettingQuery]
: Kept value for MaxDescriptionSize: 64
[2023-06-23 19:32:51.251855733][I][Irohad]: [Init] => settings
[2023-06-23 19:32:51.251860007][I][Irohad]: [Init] => validators configs
[2023-06-23 19:32:51.251862104][I][Irohad]: [Init] => transaction batch parser
[2023-06-23 19:32:51.251879528][I][Irohad]: [Init] => validators
```

Step 8: Open new terminal and attach the docker container to our terminal.

sudo docker exec -it iroha /bin/bash



Step 9: Launch the iroha-cli tool and login as admin@test.

iroha-cli -account_name admin@test

```
udit@udit-ubuntu:~$ sudo docker exec -it iroha /bin/bash
[sudo] password for udit:
root@951ab987e810:/opt/iroha_data# iroha-cli -account_name admin@test
Welcome to Iroha-Cli.
Choose what to do:
1. New transaction (tx)
2. New query (qry)
3. New transaction status request (st)
> :
```

AIM: Write a program to demonstrate mining of Ether.

```
from hashlib import sha256
import time
MAX NONCE = 10000000000
def hashGenerator(text):
    return sha256(text.encode("ascii")).hexdigest()
def mine(block number, transactions, previous hash, prefix zeros):
   prefix str = '0'*prefix zeros
    for nonce in range (MAX NONCE):
        text = str(block number) + transactions + previous hash +
str(nonce)
        new hash = hashGenerator(text)
        if new hash.startswith(prefix str):
           print(f"Successfully mined Ethers with nonce value :
{nonce}")
            return new hash
   raise BaseException(f"Couldn't find correct hash after trying
{MAX NONCE} times")
if_name_== '_main_':
   transactions = '''
   Prateek->Hajra->77,
   Kunal->Soham->18
    difficulty = 4
    start = time.time()
```

Ether mining started.

Successfully mined Ethers with nonce value : 35167

Ether mining finished.

Ether Mining took : 0.06797456741333008 seconds

Calculated Hash =

0000fc864cda3e3c5ff5be050d0d74068e1367c510e162bb27af608d0d1edbea

AIM: Demonstrate the running of the blockchain node.

```
from hashlib import sha256
def hashGenerator(text):
    return sha256(text.encode("ascii")).hexdigest()
class Block:
    def init (self, data, hash, prev hash):
        self.data = data
        self.hash = hash
        self.prev hash = prev hash
class Blockchain:
    def init (self):
        hashLast = hashGenerator('gen last')
        hashStart = hashGenerator('gen hash')
        genesis = Block('Genesis Block', hashStart, hashLast)
        self.chain = [genesis]
    def add_block(self, new_block):
        new block.prev hash = self.chain[-1].hash
        new block.hash = hashGenerator(str(new block.data)
new_block.prev_hash)
        self.chain.append(new block)
my coin = Blockchain()
my coin.add block(Block({"amount": 72}, "", ""))
my_coin.add_block(Block({"amount": 24}, "", ""))
```

```
for block in my_coin.chain:

print(block.__dict__)
```

```
{'data': 'Genesis Block', 'hash':
   '0a87388e67f16d830a9a3323dad0fdfa4c4044a6a6389cab1a0a37b651a5717b',
   'prev_hash':
   'bd6fecc16d509c74d23b04f00f936705e3eaa907b04b78872044607665018477'}
{'data': {'amount': 72}, 'hash':
   '0e97f2864c0431a30ae70be804f7e396136debab3836a0998c232efd0a9551e7',
   'prev_hash':
   '0a87388e67f16d830a9a3323dad0fdfa4c4044a6a6389cab1a0a37b651a5717b'}
{'data': {'amount': 24}, 'hash':
   'd12245a64c2b169c7877e35d6b7a8de1e514ab8e4df6f016235044438fa10f55',
   'prev_hash':
   '0e97f2864c0431a30ae70be804f7e396136debab3836a0998c232efd0a9551e7'}
```

AIM: Demonstrate the use of Bitcoin Core API.

```
from bitcoinlib.wallets import Wallet

w = Wallet.create("UDITWallet")

key1 = w.get_key()

print(key1.address)

w.scan()

print(w.info())
```

18 W Pe X whb Mh 4 f To sr 4 b TmHzkq 6 h 3 Rbh 3 Nk

=== WALLET ===

Witness type

ID 2

Name UDITWallet

Owner

Scheme bip32

Multisig False

Main network bitcoin

Latest update 2023-06-25 18:56:51.183095

legacy

= Wallet Master Key =

ID 17

Private True

Depth 0

- NETWORK: bitcoin -	
Keys	
22 m/44'/0'/0'/0/0	18WPeXwhbMh4fTosr4bTmHzkq6h3Rbh3Nk
address index 0	0.00000000 #
23 m/44'/0'/0'/0/1	1KmMGyiGK7QzUEv3i3yWGra8dcwuJy7vw7
address index 1	0.00000000 #
24 m/44'/0'/0'/0/2	1DPHYJiULScrqc8cfhV9D71m9R77rr1ei1
address index 2	0.00000000 #
25 m/44'/0'/0'/0/3	1CRBZCJUBKJEf27TMdAJgPPTbiiFaJf1sD
address index 3	0.00000000 #
26 m/44'/0'/0'/0/4	1J6U3rXAmHq9ys8G5i2VX2yJPJCCtDNehu
address index 4	0.00000000 #
28 m/44'/0'/0'/1/0	1FyAbZiJXkEbvpAfG6jWpQ9iFc63wd2zkK
address index 0	0.00000000 #
29 m/44'/0'/0'/1/1	187Dgp9RAUxRtNGuZurzeCxGqBPwfLkB1i
address index 1	0.00000000 #
30 m/44'/0'/0'/1/2	1A2fszNSRK9av23jPs5bMYREv2Vyf6xShq
address index 2	0.00000000 #
31 m/44'/0'/0'/1/3	1AameeN7QCpPzXc76tgwegATXCSHfhMutc
address index 3	0.00000000 #
32 m/44'/0'/0'/1/4	1DNwVEQE4heKmP5moERQdGn4gGzSkZp8cY
address index 4	0.00000000 \$

```
- - Transactions Account 0 (0)
```

= Balance Totals (includes unconfirmed) =

None

AIM: Create your own blockchain and demonstrate its use.

```
from hashlib import sha256
def hashGenerator(text):
    return sha256(text.encode("ascii")).hexdigest()
class Block:
    def init (self,data,hash,prev hash):
        self.data=data
        self.hash=hash
        self.prev hash=prev hash
class Blockchain:
    def init (self):
      hashLast=hashGenerator('gen last')
      hashStart=hashGenerator('gen hash')
      genesis=Block('gen-data', hashStart, hashLast)
      self.chain=[genesis]
    def add_block(self,data):
        prev hash=self.chain[-1].hash
        hash=hashGenerator(data+prev hash)
        block=Block(data, hash, prev_hash)
        self.chain.append(block)
bc=Blockchain()
bc.add block('1')
bc.add block('2')
bc.add block('3')
```

```
for block in bc.chain:

print(block.__dict__)
```

```
{'data': 'gen-data', 'hash':
 '0a87388e67f16d830a9a3323dad0fdfa4c4044a6a6389cab1a0a37b651a5717b',
 'prev_hash':
 'bd6fecc16d509c74d23b04f00f936705e3eaa907b04b78872044607665018477'}
{'data': '1', 'hash':
 'e3e6c97161f3deaf01599fda60ba85593b07f70328bf228473d1d408f7400241',
 'prev_hash':
 '0a87388e67f16d830a9a3323dad0fdfa4c4044a6a6389cab1a0a37b651a5717b'}
{'data': '2', 'hash':
 '47e8645e3c14bd4034a498aa88ea630bc0793375207bf90ca469792a5d9484e1',
 'prev_hash':
 'e3e6c97161f3deaf01599fda60ba85593b07f70328bf228473d1d408f7400241'}
{'data': '3', 'hash':
 '82084603decb1a14a8819dacaa86197659f1e150c4a50186e68043004b5a3c06',
 'prev_hash':
 '47e8645e3c14bd4034a498aa88ea630bc0793375207bf90ca469792a5d9484e1'}
```