INDEX

List of Practical:		Date	Signature
1.	Write the following programs for Blockchain in Python:		
a.	A simple client class that generates the private and public keys byusing the built-in Python RSA algorithm and test it.	16/2/23	
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.		
2.	Install and configure Go Ethereum and the Mist browser. Develop andtest a sample application.(MetaMask & Remix)	22/2/23	
3.	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.	3/3/23	
b.	Functions, Function Modifiers, View functions, Pure Functions, Fallback Function, Function Overloading, Mathematical functions, Cryptographic functions.		
4.	Implement and demonstrate the use of the following in Solidity:		
a.	Withdrawal Pattern, Restricted Access.	7/3/23	
b.	Contracts, Inheritance, Constructors, Abstract Contracts, Interfaces.		
c.	Libraries, Assembly, Events, Error handling.		
5.	Write a program to demonstrate mining of Ether.	17/3/23	
6.	Demonstrate the running of the blockchain node.	03/04/23	
7.	Create your own blockchain and demonstrate its use.	10/04/23	

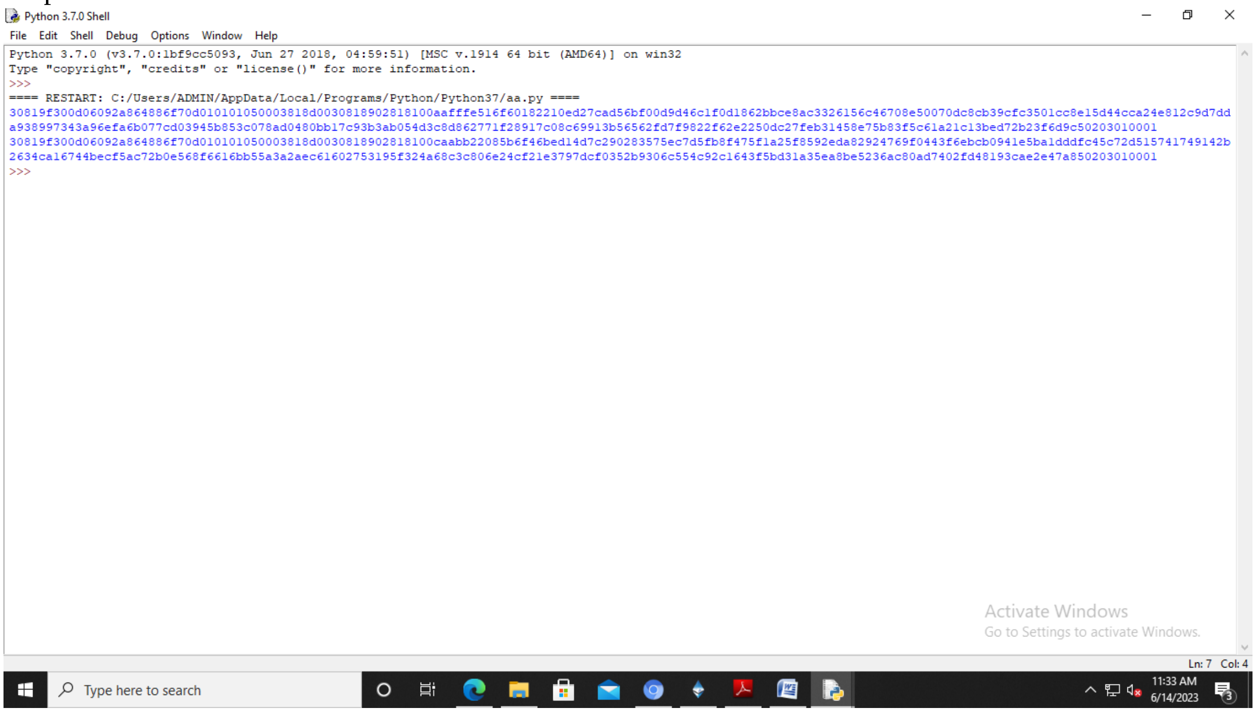
Practical 1

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.

```
import hashlib
import random
import string
import json
import binascii
import numpy as np
import pandas as pd
import pylab as pl
import logging
import datetime
import collections
import Crypto
import Crypto.Random
from Crypto.Hash import SHA
from Crypto.PublicKey import RSA
from Crypto.Signature import PKCS1_v1_5
import binascii
class Client:
 def __init__(self):
  random = Crypto.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)
  self.identity=binascii.hexlify(self._public_key.exportKey(format='DER')).decode('ascii')
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):
  if self.sender == "Genesis":
   identity = "Genesis";
  else:
   identity = 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')
Dinesh = Client()
Ramesh = Client()
print (Dinesh.identity)
print (Ramesh.identity)
```

Output:

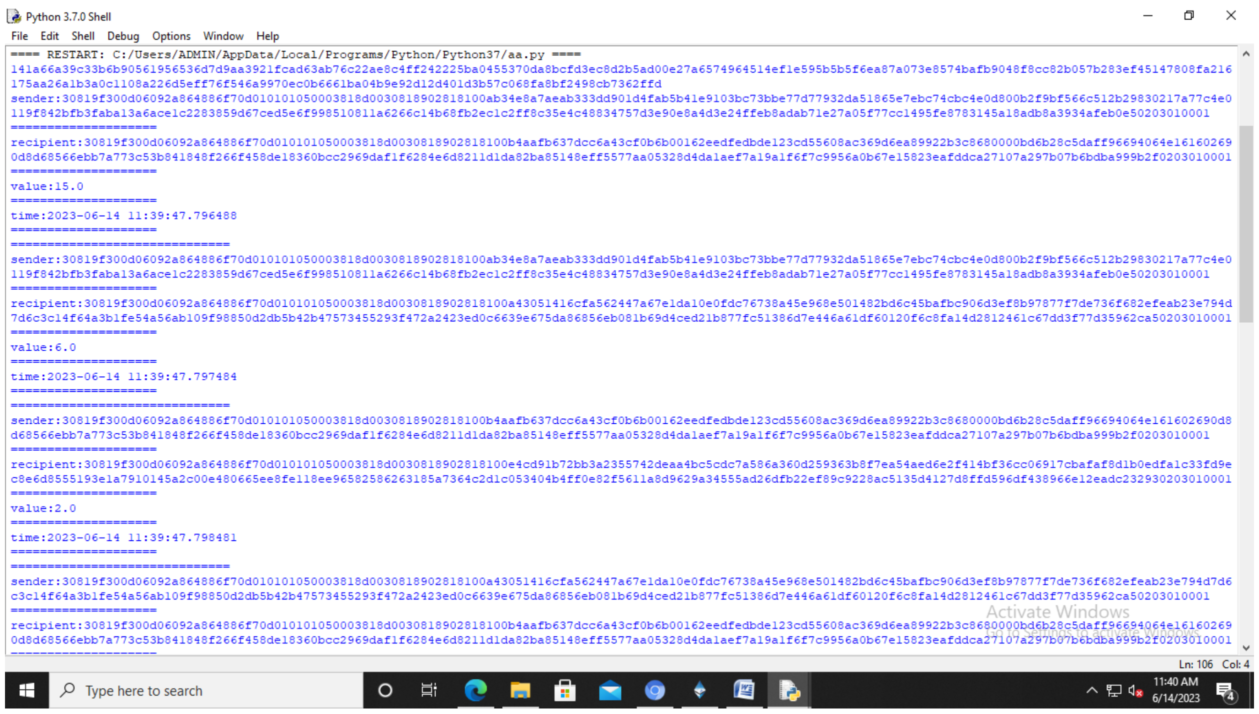


```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 
==== RESTART: C:/Users/ADMIN/AppData/Local/Programs/Python/Python37/aa.py ====
30819f300d06092a864886f70d010101050003818d0030818902818100aafffe516f60182210ed27cad56bf00d9d46c1f0d1862bbce8ac3326156c46708e50070dc8cb39cfc3501cc8e15d44cca24e812c9d7dd
a938997343a96efa6b077cd03945b853c078ad0480bb17c93b3ab054d3c8d862771f28917c08c69913b56562fd7f9822f62e2250dc27feb31458e75b83f5c61a21c13bed72b23f6d9c50203010001
30819f300d06092a864886f70d010101050003818d0030818902818100caabb22085b6f46bed14d7c290283575ec7d5fb8f475f1a25f8592eda82924769f0443f6ebcb0941e5ba1dddfc45c72d515741749142b
2634ca16744becf5ac72b0e568f6616bb55a3a2aec61602753195f324a68c3c806e24cf21e3797dcf0352b9306c554c92c1643f5bd31a35ea8be5236ac80ad7402fd48193cae2e47a850203010001
>>> 
```

b) A transaction class to send and receive money and test it.

c) Create multiple transactions and display them.

```
import hashlib
import random
import string
import json
import binascii
import numpy as np
import pandas as pd
import pylab as pl
import logging
import datetime
import collections
import Crypto
import Crypto.Random
from Crypto.Hash import SHA
from Crypto.PublicKey import RSA
from Crypto.Signature import PKCS1_v1_5
import binascii
class Client:
 def __init__(self):
  random = Crypto.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)
  self.identity=binascii.hexlify(self._public_key.exportKey(format='DER')).decode('ascii')
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):
    if self.sender == "Genesis":
      identity = "Genesis"
    else:
      identity = 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')
Dinesh = Client()
Ramesh = Client()
t = Transaction (sender=Dinesh,recipient=Ramesh.identity,value=5.0)
signature = t.sign_transaction()
print (signature)
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 ('=====================')
transactions = []
Dinesh = Client()
Ramesh = Client()
Seema = Client()
Vijay = Client()
t1 = Transaction(
 Dinesh,
 Ramesh.identity,
 15.0
)
t1.sign_transaction()
transactions.append(t1)
t2 = Transaction(
 Dinesh,
 Seema.identity,
 6.0
)
t2.sign_transaction()
transactions.append(t2)
t3 = Transaction(

 Ramesh,
 Vijay.identity,
 2.0
)
t3.sign_transaction()
transactions.append(t3)
t4 = Transaction(
 Seema,
 Ramesh.identity,
 4.0
)
t4.sign_transaction()
transactions.append(t4)
t5 = Transaction(
 Vijay,
 Seema.identity,
 7.0
)
t5.sign_transaction()
transactions.append(t5)
t6 = Transaction(
 Ramesh,
 Seema.identity,
 3.0
)
t6.sign_transaction()
transactions.append(t6)
t7 = Transaction(
 Seema,
 Dinesh.identity,
 8.0
)
t7.sign_transaction()
transactions.append(t7)
t8 = Transaction(
 Seema,
 Ramesh.identity,
 1.0
)
t8.sign_transaction()
transactions.append(t8)
t9 = Transaction(
 Vijay,
 Dinesh.identity,
 5.0
)
t9.sign_transaction()
transactions.append(t9)

t10 = Transaction(
 Vijay,
 Ramesh.identity,
 3.0
)
t10.sign_transaction()
transactions.append(t10)
for transaction in transactions:
 display_transaction (transaction)
 print ('==============================')
```

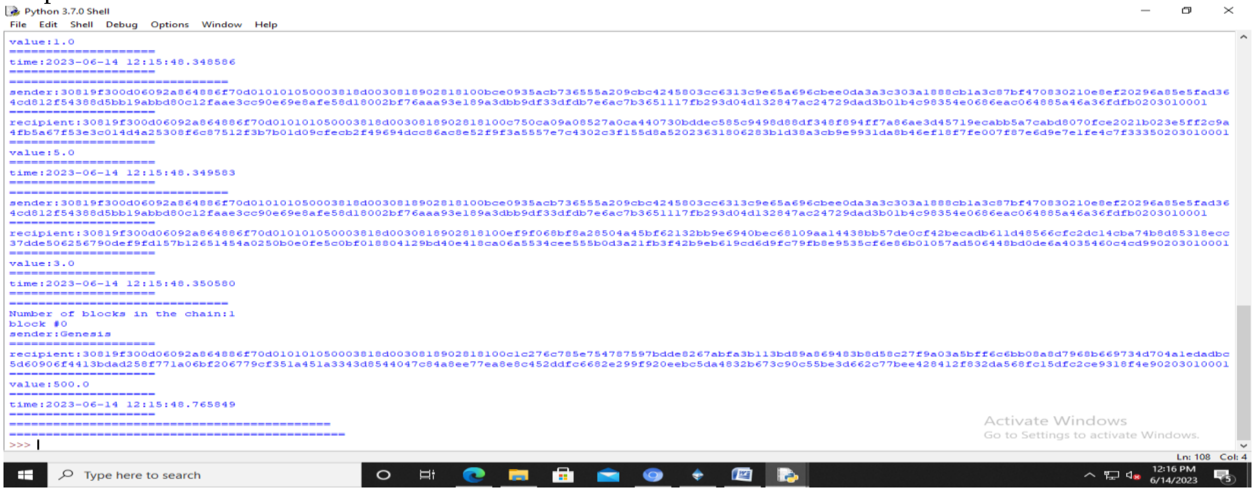


```
==== RESTART: C:/Users/ADMIN/AppData/Local/Programs/Python/Python37/aa.py ====
141a66a39c33b6b90561956536d7d9aa3921fcad63ab76c22ae8c4ff242225ba0455370da8bcfd3ec8d2b5ad00e27a6574964514ef1e595b5b5f6ea87a073e8574bafb9048f8cc82b057b283ef45147808fa216
175aa26a1b3a0c1108a226d5eff76f546a9970ec0b6661ba04b9e92d12d401d3b57c068fa8bf2498cb7362ffd
sender:30819f300d06092a864886f70d010101050003818d0030818902818100ab34e8a7aeab333dd901d4fab5b41e9103bc73bbe77d77932da51865e7ebc74cbc4e0d800b2f9bf566c512b29830217a77c4e0
119f842bfb3faba13a6ace1c2283859d67ced5e6f998510811a6266c14b68fb2ec1c2ff8c35e4c48834757d3e90e8a4d3e24ffeb8adab71e27a05f77cc1495fe8783145a18adb8a3934afeb0e50203010001
====================
recipient:30819f300d06092a864886f70d010101050003818d0030818902818100b4aafb637dcc6a43cf0b6b00162eedfedbde123cd55608ac369d6ea89922b3c8680000bd6b28c5daff96694064e16160269
0d8d68566ebb7a773c53b841848f266f458de18360bcc2969daf1f6284e6d8211d1da82ba85148eff5577aa05328d4da1aef7a19a1f6f7c9956a0b67e15823eafddca27107a297b07b6bdba999b2f0203010001
====================
value:15.0
====================
time:2023-06-14 11:39:47.796488
====================
==============================
sender:30819f300d06092a864886f70d010101050003818d0030818902818100ab34e8a7aeab333dd901d4fab5b41e9103bc73bbe77d77932da51865e7ebc74cbc4e0d800b2f9bf566c512b29830217a77c4e0
119f842bfb3faba13a6ace1c2283859d67ced5e6f998510811a6266c14b68fb2ec1c2ff8c35e4c48834757d3e90e8a4d3e24ffeb8adab71e27a05f77cc1495fe8783145a18adb8a3934afeb0e50203010001
====================
recipient:30819f300d06092a864886f70d010101050003818d0030818902818100a43051416cfa562447a67e1da10e0fdc76738a45e968e501482bd6c45bafbc906d3ef8b97877f7de736f682efeab23e794d
7d6c3c14f64a3b1fe54a56ab109f98850d2db5b42b47573455293f472a2423ed0c6639e675da86856eb081b69d4ced21b877fc51386d7e446a61df60120f6c8fa14d2812461c67dd3f77d35962ca50203010001
====================
value:6.0
====================
time:2023-06-14 11:39:47.797484
====================
==============================
sender:30819f300d06092a864886f70d010101050003818d0030818902818100b4aafb637dcc6a43cf0b6b00162eedfedbde123cd55608ac369d6ea89922b3c8680000bd6b28c5daff96694064e161602690d8
d68566ebb7a773c53b841848f266f458de18360bcc2969daf1f6284e6d8211d1da82ba85148eff5577aa05328d4da1aef7a19a1f6f7c9956a0b67e15823eafddca27107a297b07b6bdba999b2f0203010001
====================
recipient:30819f300d06092a864886f70d010101050003818d0030818902818100e4cd91b72bb3a2355742deaa4bc5cdc7a586a360d259363b8f7ea54aed6e2f414bf36cc06917cbafaf8d1b0edfa1c33fd9e
c8e6d855193e1a7910145a2c00e480665ee8fe118ee96582586263185a7364c2d1c053404b4ff0e82f5611a8d9629a34555ad26dfb22ef89c9228ac5135d4127d8ffd596df438966e12eadc232930203010001
====================
value:2.0
====================
time:2023-06-14 11:39:47.798481
====================
==============================
sender:30819f300d06092a864886f70d010101050003818d0030818902818100a43051416cfa562447a67e1da10e0fdc76738a45e968e501482bd6c45bafbc906d3ef8b97877f7de736f682efeab23e794d7d6
c3c14f64a3b1fe54a56ab109f98850d2db5b42b47573455293f472a2423ed0c6639e675da86856eb081b69d4ced21b877fc51386d7e446a61df60120f6c8fa14d2812461c67dd3f77d35962ca50203010001
====================
recipient:30819f300d06092a864886f70d010101050003818d0030818902818100b4aafb637dcc6a43cf0b6b00162eedfedbde123cd55608ac369d6ea89922b3c8680000bd6b28c5daff96694064e16160269
0d8d68566ebb7a773c53b841848f266f458de18360bcc2969daf1f6284e6d8211d1da82ba85148eff5577aa05328d4da1aef7a19a1f6f7c9956a0b67e15823eafddca27107a297b07b6bdba999b2f0203010001
```

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.

```
import hashlib
import random
import string
import json
import binascii
import numpy as np
import pandas as pd
import pylab as pl
import logging
import datetime
import collections
import Crypto
import Crypto.Random
from Crypto.Hash import SHA
from Crypto.PublicKey import RSA
from Crypto.Signature import PKCS1_v1_5
import binascii
class Client:
 def __init__(self):
  random = Crypto.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)
  self.identity=binascii.hexlify(self._public_key.exportKey(format='DER')).decode('ascii')
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):
    if self.sender == "Genesis":
      identity = "Genesis"
    else:
      identity = 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')
Dinesh = Client()
Ramesh = Client()
t = Transaction (sender=Dinesh,recipient=Ramesh.identity,value=5.0)
signature = t.sign_transaction()
print (signature)
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 ('=====================')
transactions = []
Dinesh = Client()
Ramesh = Client()
Seema = Client()
Vijay = Client()
t1 = Transaction(
 Dinesh,
 Ramesh.identity,
 15.0
)
t1.sign_transaction()
transactions.append(t1)
t2 = Transaction(
 Dinesh,
 Seema.identity,
 6.0
)
t2.sign_transaction()
transactions.append(t2)
t3 = Transaction(

 Ramesh,
 Vijay.identity,
 2.0
)
t3.sign_transaction()
transactions.append(t3)
t4 = Transaction(
 Seema,
 Ramesh.identity,
 4.0
)
t4.sign_transaction()
transactions.append(t4)
t5 = Transaction(
 Vijay,
 Seema.identity,
 7.0
)
t5.sign_transaction()
transactions.append(t5)
t6 = Transaction(
 Ramesh,
 Seema.identity,
 3.0
)
t6.sign_transaction()
transactions.append(t6)
t7 = Transaction(
 Seema,
 Dinesh.identity,
 8.0
)
t7.sign_transaction()
transactions.append(t7)
t8 = Transaction(
 Seema,
 Ramesh.identity,
 1.0
)
t8.sign_transaction()
transactions.append(t8)
t9 = Transaction(
 Vijay,
 Dinesh.identity,
 5.0
)
t9.sign_transaction()
transactions.append(t9)

t10 = Transaction(
 Vijay,
 Ramesh.identity,
 3.0
)
t10.sign_transaction()
transactions.append(t10)
for transaction in transactions:
 display_transaction (transaction)
 print ('==============================')

class Block:
 def __init__(self):
  self.verified_transactions = []
  self.previous_block_hash = ""
  self.Nonce = ""
last_block_hash = ""
Dinesh = Client()
t0 = Transaction (
 "Genesis",
 Dinesh.identity,
 500.0
)
block0 = Block()
block0.previous_block_hash = None
Nonce = None
block0.verified_transactions.append (t0)
digest = hash (block0)
last_block_hash = digest
TPCoins = []
def dump_blockchain (self):
 print ("Number of blocks in the chain:" + str(len (self)))
 for x in range (len(TPCoins)):
  block_temp = TPCoins[x]
  print ("block #" + str(x))
  for transaction in block_temp.verified_transactions:
    display_transaction (transaction)
    print ('============================================')
  print ('============================================')
TPCoins.append (block0)
dump_blockchain(TPCoins)
```

Output:

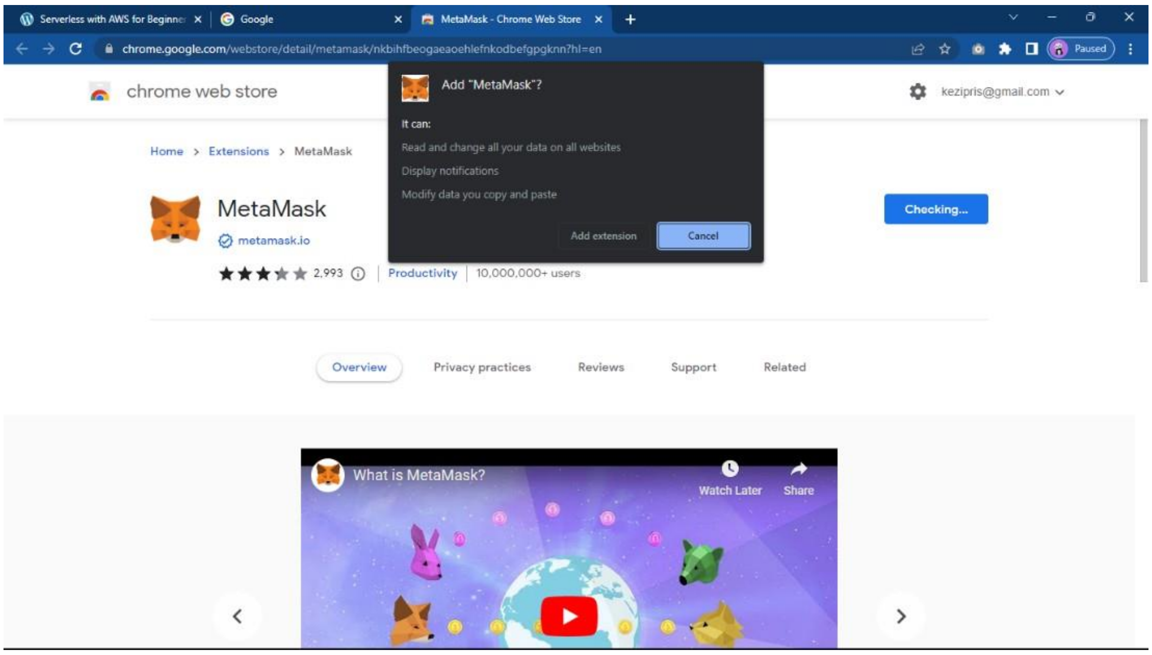


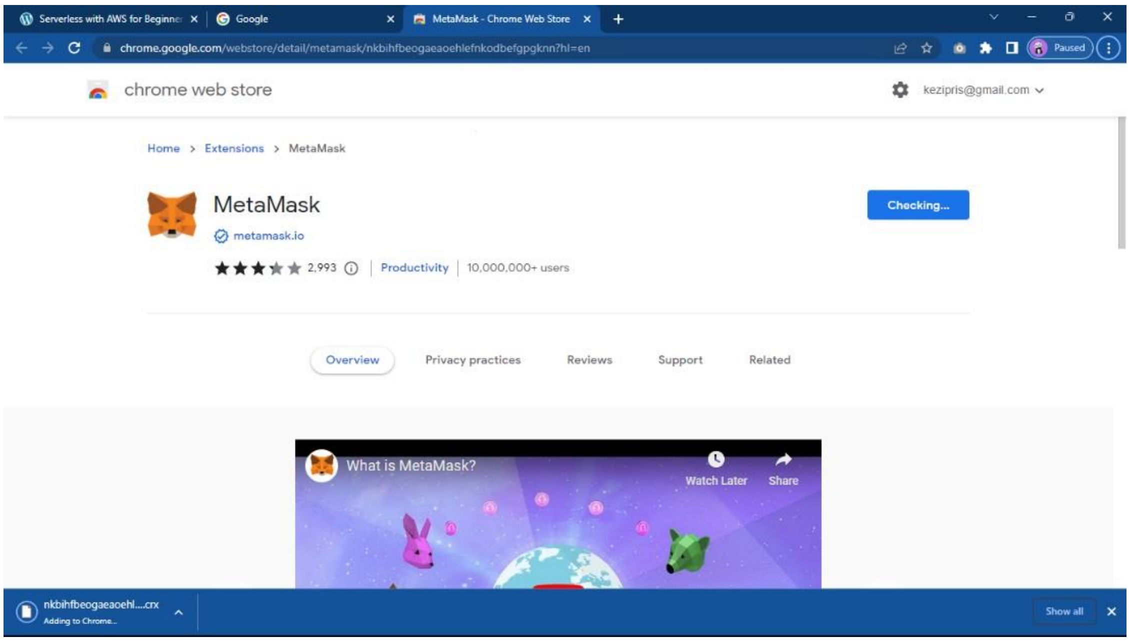
```
value:1.0
====================
time:2023-06-14 12:15:48.348586
====================
==============================
sender:30819f300d06092a864886f70d010101050003818d0030818902818100bce0935acb736555a209cbc4245803cc6313c9e65a696cbee0da3a3c303a1888cb1a3c87bf470830210e8ef20296a85e5fad36
4cd812f54388d5bb19abbd80c12faae3cc90e69e8afe58d18002bf76aaa93e189a3dbb9df33dfdb7e6ac7b3651117fb293d04d132847ac24729dad3b01b4c98354e0686eac064885a46a36fdfb0203010001
====================
recipient:30819f300d06092a864886f70d010101050003818d0030818902818100c750ca09a08527a0ca440730bddec585c9498d88df348f894ff7a86ae3d45719ecabb5a7cabd8070fce2021b023e5ff2c9a
4fb5a67f53e3c014d4a25308f6c87512f3b7b01d09cfecb2f49694dcc86ac8e52f9f3a5557e7c4302c3f155d8a52023631806283b1d38a3cb9e9931da8b46ef18f7fe007f87e6d9e7e1fe4c7f3350203010001
====================
value:5.0
====================
time:2023-06-14 12:15:48.349583
====================
==============================
sender:30819f300d06092a864886f70d010101050003818d0030818902818100bce0935acb736555a209cbc4245803cc6313c9e65a696cbee0da3a3c303a1888cb1a3c87bf470830210e8ef20296a85e5fad36
4cd812f54388d5bb19abbd80c12faae3cc90e69e8afe58d18002bf76aaa93e189a3dbb9df33dfdb7e6ac7b3651117fb293d04d132847ac24729dad3b01b4c98354e0686eac064885a46a36fdfb0203010001
====================
recipient:30819f300d06092a864886f70d010101050003818d0030818902818100ef9f068bf8a28504a45bf62132bb9e6940bec68109aa14438bb57de0cf42becadb611d48566cfc2dc14cba74b8d85318ecc
37dde506256790def9fd157b12651454a0250b0e0fe5c0bf018804129bd40e418ca06a5534cee555b0d3a21fb3f42b9eb619cd6d9fc79fb8e9535cf6e86b01057ad506448bd0de6a4035460c4cd990203010001
====================
value:3.0
====================
time:2023-06-14 12:15:48.350580
====================
==============================
Number of blocks in the chain:1
block #0
sender:Genesis
====================
recipient:30819f300d06092a864886f70d010101050003818d0030818902818100c1c276c785e754787597bdde8267abfa3b113bd89a869483b8d58c27f9a03a5bff6c6bb08a8d7968b669734d704a1edadbc
5d60906f4413bdad258f771a06bf206779cf351a451a3343d8544047c84a8ee77ea8e8c452ddfc6682e299f920eebc5da4832b673c90c55be3d662c77bee428412f832da568fc15dfc2ce9318f4e90203010001
====================
value:500.0
====================
time:2023-06-14 12:15:48.765849
====================
============================================
==============================================
>>> 
```

PRACTICAL-2

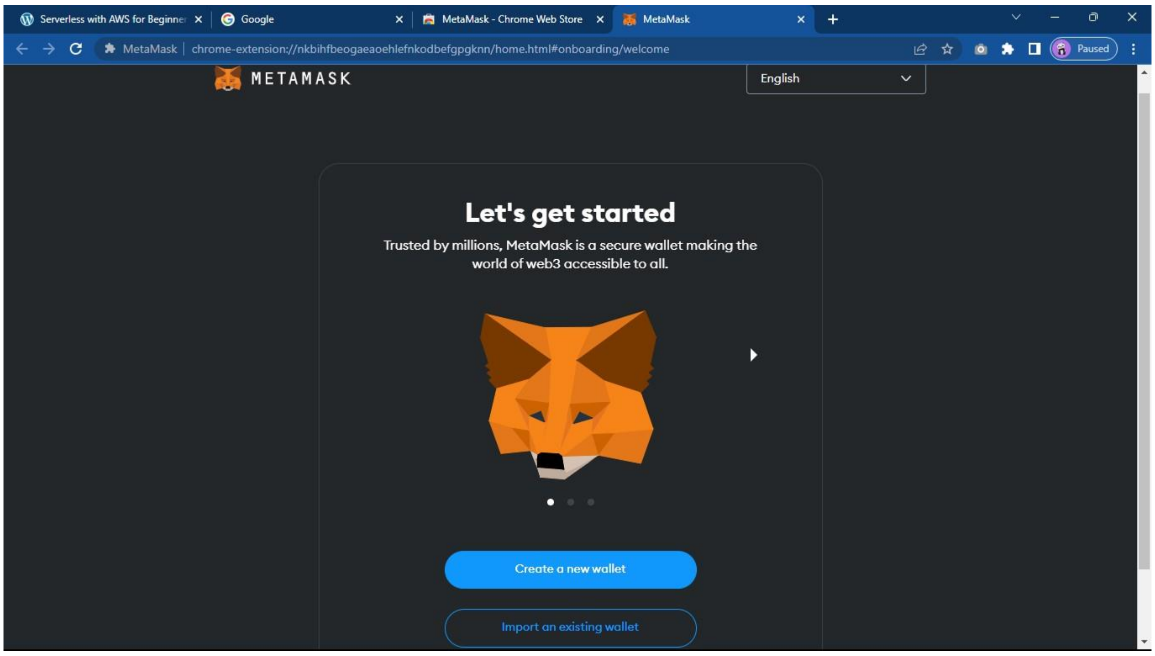
Aim: INSTALL AND CONFIGURE GO ETHEREUM AND THEMIST BROWSER. DEVELOP AND TEST A SAMPLE APPLICATION(METAMASK & REMIX)

Step 1-> Install MetaMask extension for chrome from Chrome Web Store



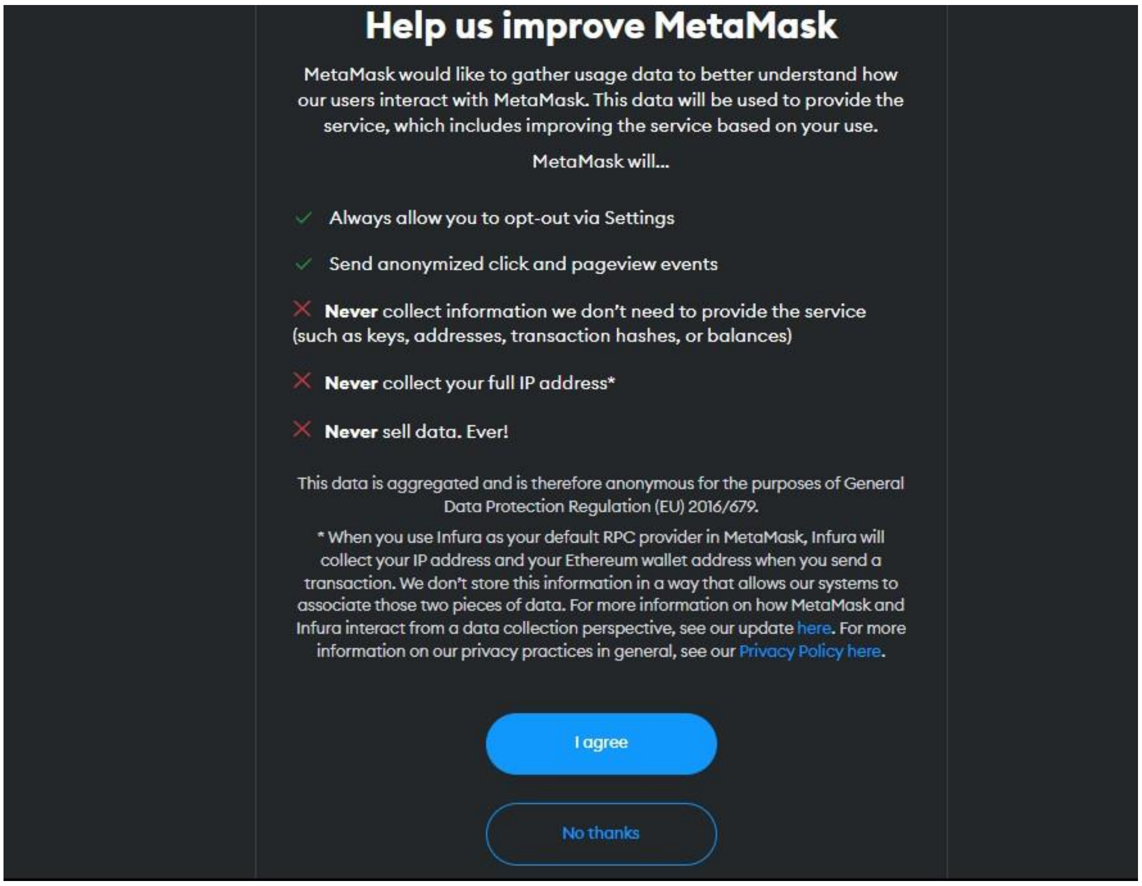


Step 2-> Click on Metamask Extension in Extensions. Below page will open in anew tab. Click on Create a New Wallet. Click on I agree.


Serverless with AWS for Beginner
Google
MetaMask - Chrome Web Store
MetaMask
MetaMask | chrome-extension://nkbihfbeogaeaoehlefnkodbefgpgknn/home.html#onboarding/welcome
Paused
METAMASK
English
Create a new wallet
Import an existing wallet

Let's get started

Trusted by millions, MetaMask is a secure wallet making the world of web3 accessible to all.



Help us improve MetaMask

MetaMask would like to gather usage data to better understand how our users interact with MetaMask. This data will be used to provide the service, which includes improving the service based on your use.

MetaMask will...

✓ Always allow you to opt-out via Settings

✓ Send anonymized click and pageview events

✗ **Never** collect information we don't need to provide the service (such as keys, addresses, transaction hashes, or balances)

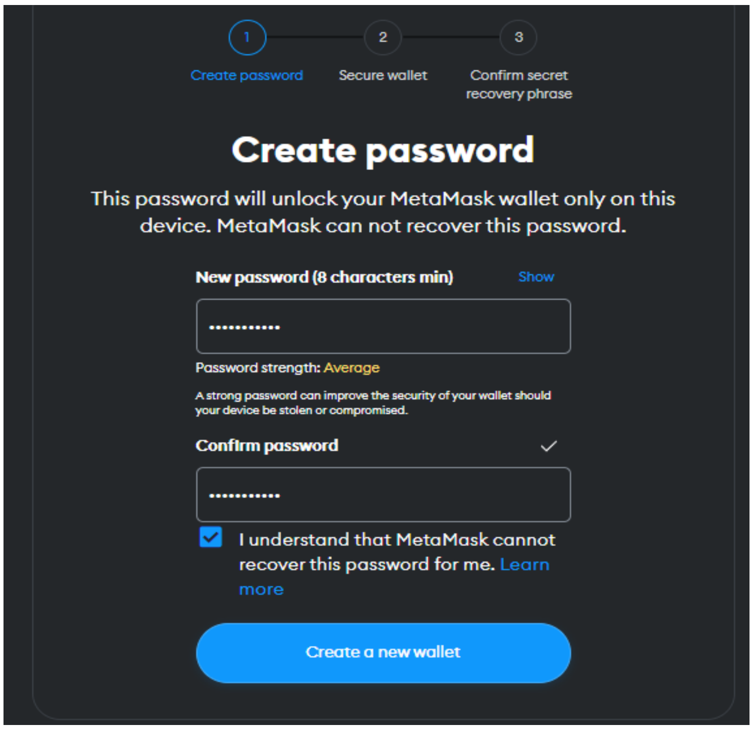
✗ **Never** collect your full IP address*

✗ **Never** sell data. Ever!

This data is aggregated and is therefore anonymous for the purposes of General Data Protection Regulation (EU) 2016/679.

* When you use Infura as your default RPC provider in MetaMask, Infura will collect your IP address and your Ethereum wallet address when you send a transaction. We don't store this information in a way that allows our systems to associate those two pieces of data. For more information on how MetaMask and Infura interact from a data collection perspective, see our update here. For more information on our privacy practices in general, see our Privacy Policy here.

Step 3-> Create a password. This password can be used only on the device it was created on. Create a Strong password and click on Create a new Wallet button



Create password

This password will unlock your MetaMask wallet only on this device. MetaMask can not recover this password.

New password (8 characters min) Show

•••••••••••

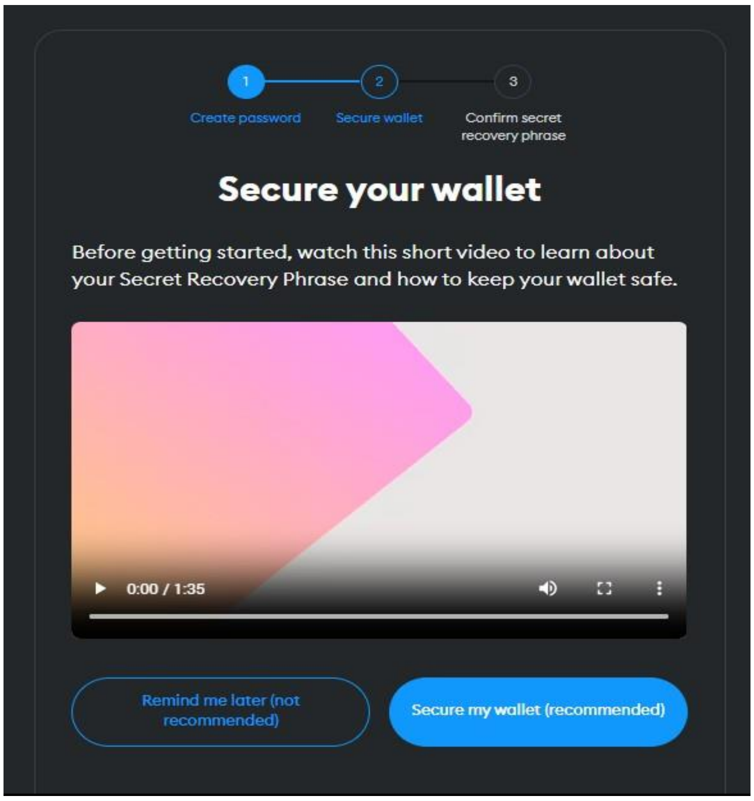
Password strength: Average

A strong password can improve the security of your wallet should your device be stolen or compromised.

Confirm password ✓

•••••••••••

☑ I understand that MetaMask cannot recover this password for me. Learn more

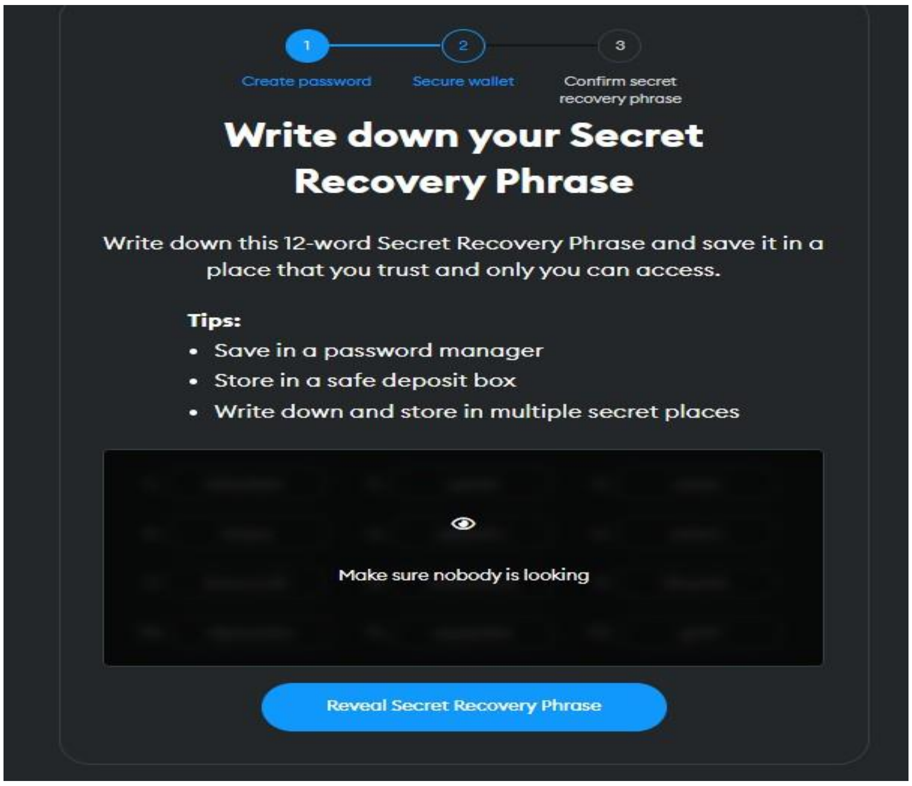

1
2
3
Create password
Secure wallet
Confirm secret recovery phrase

Secure your wallet

Before getting started, watch this short video to learn about your Secret Recovery Phrase and how to keep your wallet safe.

0:00 / 1:35
Remind me later (not recommended)
Secure my wallet (recommended)

Step 4-> Click on Secure my wallet button, following window will appear



Write down your Secret Recovery Phrase

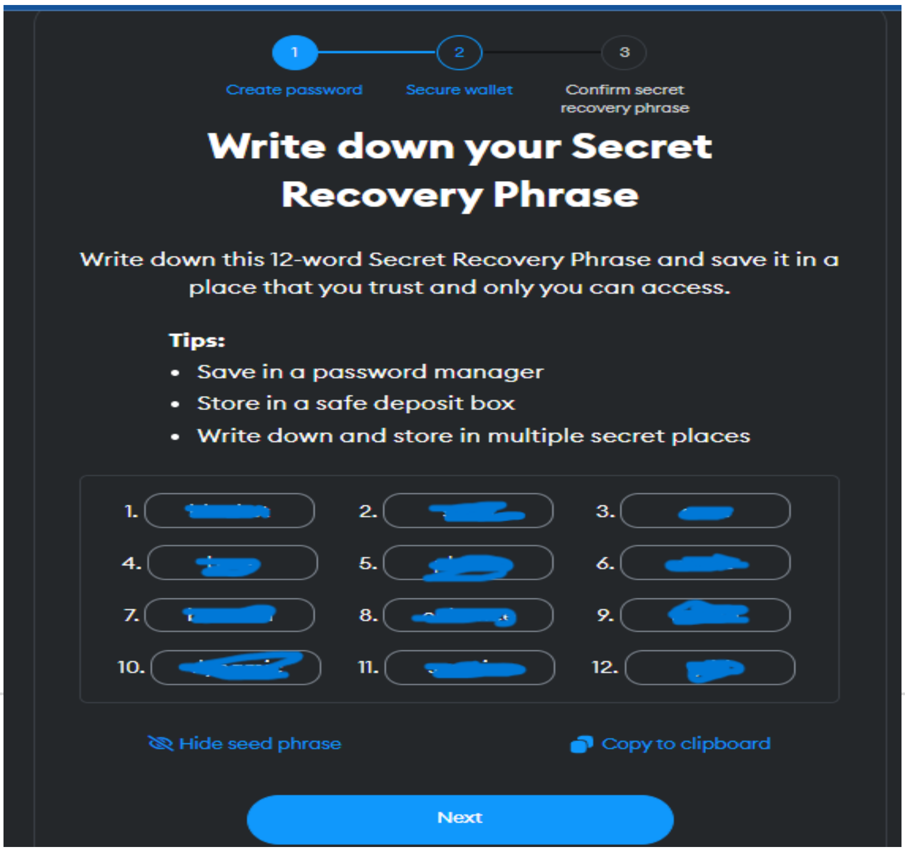
Write down this 12-word Secret Recovery Phrase and save it in a place that you trust and only you can access.

Tips:

- Save in a password manager
- Store in a safe deposit box
- Write down and store in multiple secret places

Make sure nobody is looking

Step 5-> Click on Reveal Secret Recovery Phrase button and save the words inthe same sequence



Write down your Secret Recovery Phrase

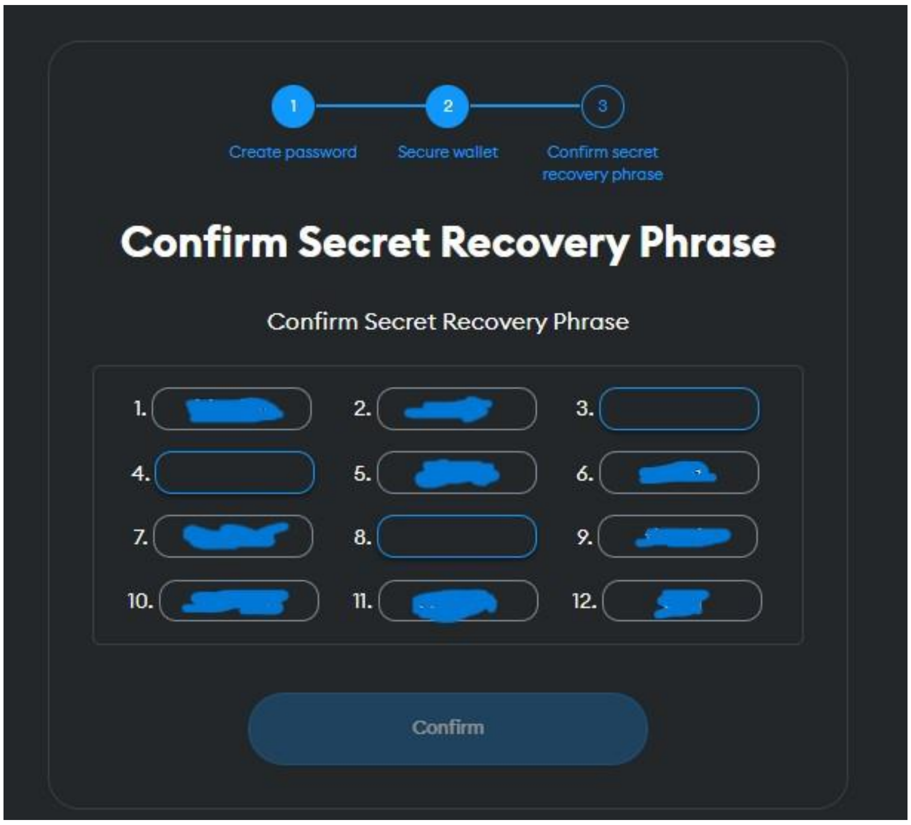
Write down this 12-word Secret Recovery Phrase and save it in a place that you trust and only you can access.

Tips:

- Save in a password manager
- Store in a safe deposit box
- Write down and store in multiple secret places

1. [redacted]	2. [redacted]	3. [redacted]
4. [redacted]	5. [redacted]	6. [redacted]
7. [redacted]	8. [redacted]	9. [redacted]
10. [redacted]	11. [redacted]	12. [redacted]

Step 6-> Enter the respective words in the empty positions and click Confirm.

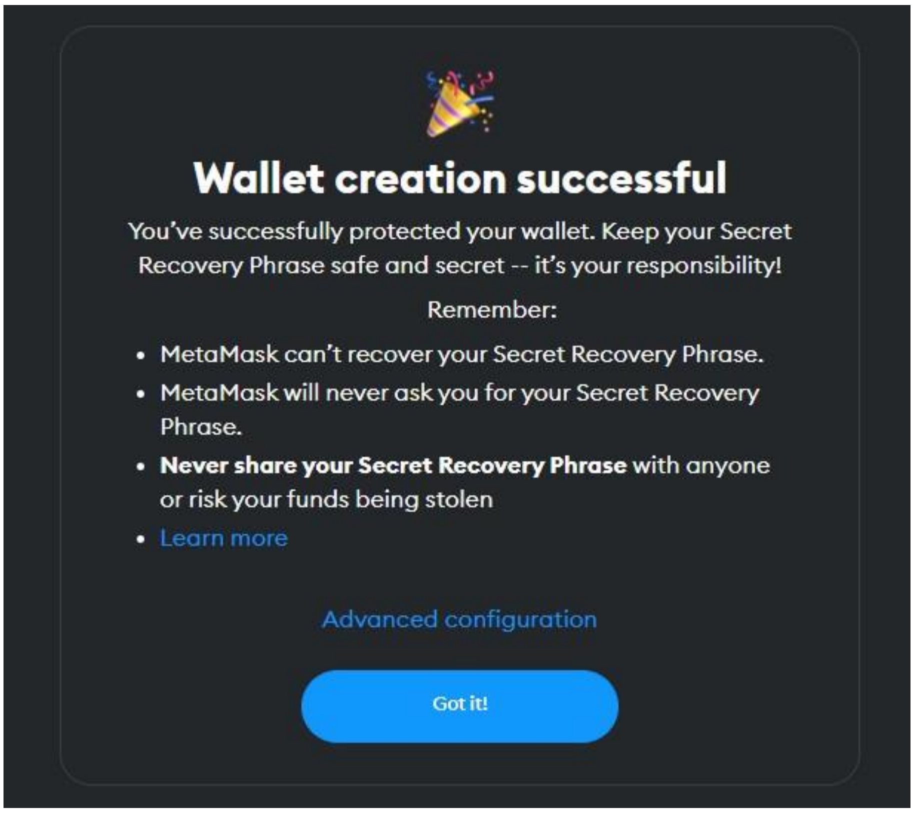


Confirm Secret Recovery Phrase

Confirm Secret Recovery Phrase

1. [redacted]	2. [redacted]	3.
4.	5. [redacted]	6. [redacted]
7. [redacted]	8.	9. [redacted]
10. [redacted]	11. [redacted]	12. [redacted]

Step 7-> Click Got it!



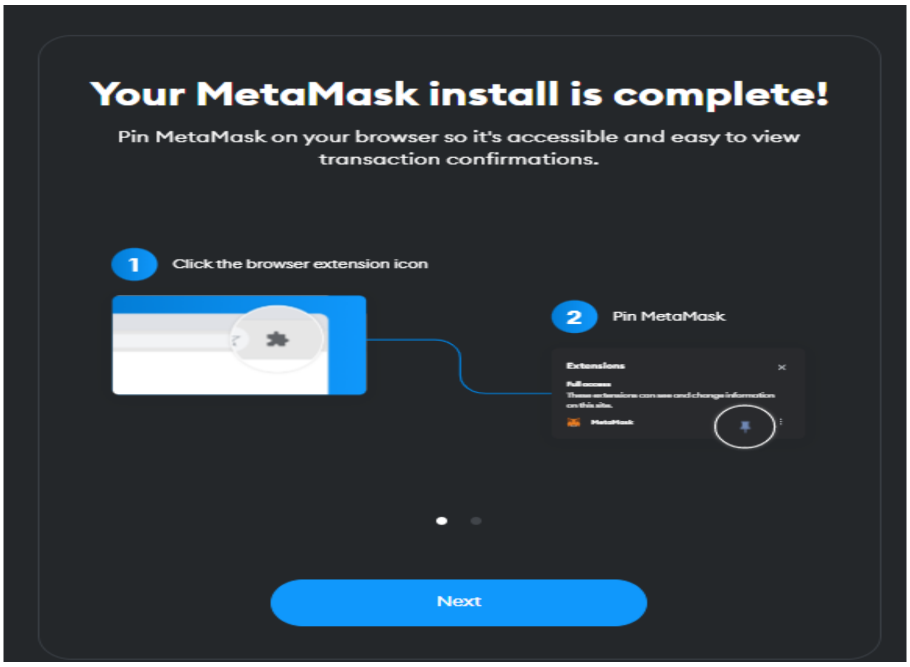
Wallet creation successful

You've successfully protected your wallet. Keep your Secret Recovery Phrase safe and secret -- it's your responsibility!

Remember:

- MetaMask can't recover your Secret Recovery Phrase.
- MetaMask will never ask you for your Secret Recovery Phrase.
- **Never share your Secret Recovery Phrase** with anyone or risk your funds being stolen
- Learn more

Step 8-> Click on Next

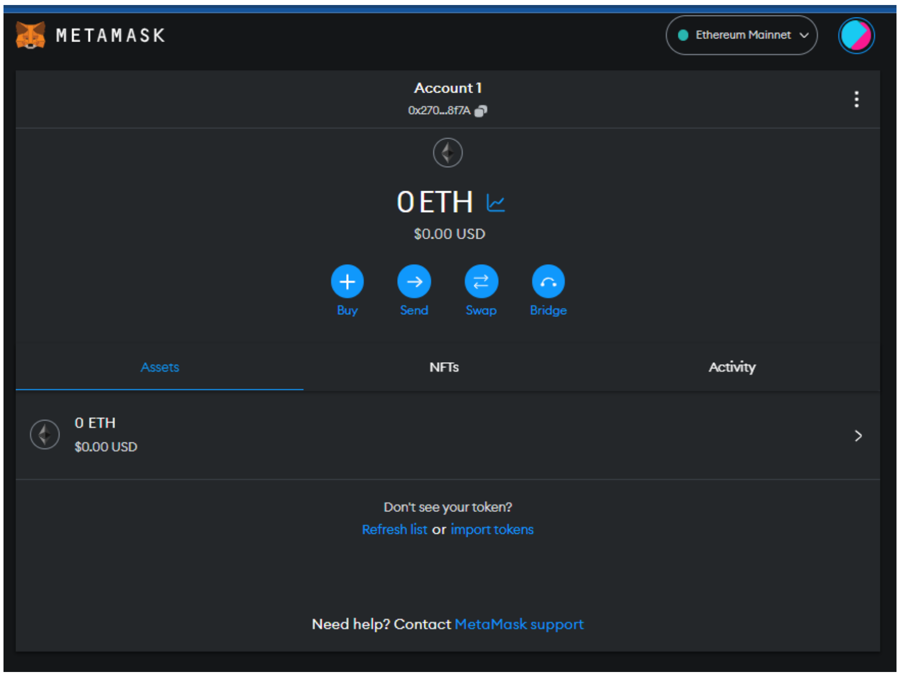


Your MetaMask install is complete!

Pin MetaMask on your browser so it's accessible and easy to view transaction confirmations.

1 Click the browser extension icon
2 Pin MetaMask
Extensions
Full access
These extensions can see and change information on this site.
MetaMask
Next

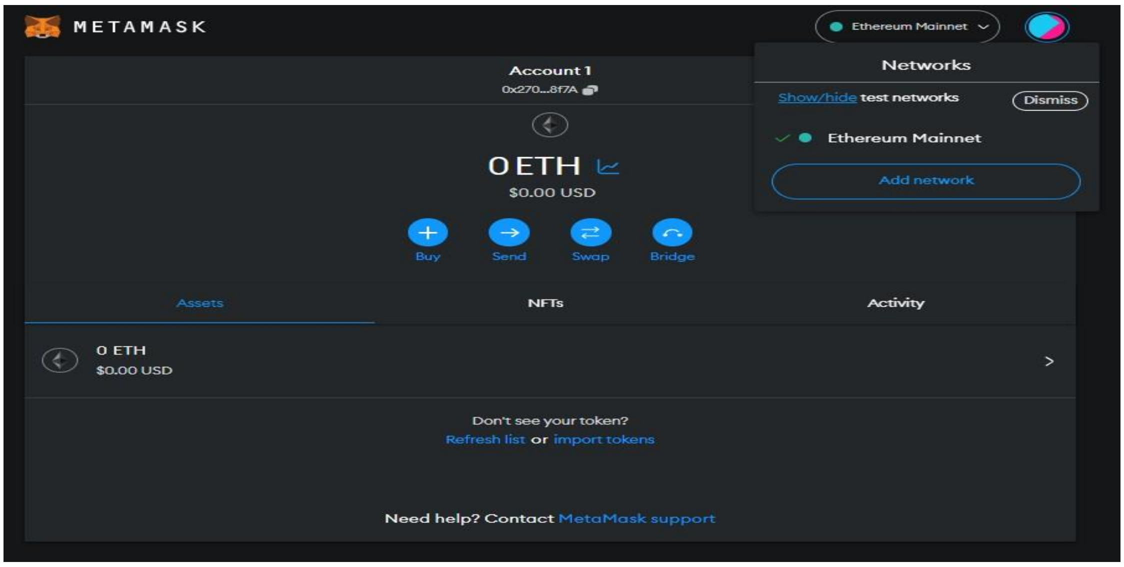
Step 9-> Following will be the Dashboard



Don't see your token?
Refresh list or import tokens

Need help? Contact MetaMask support

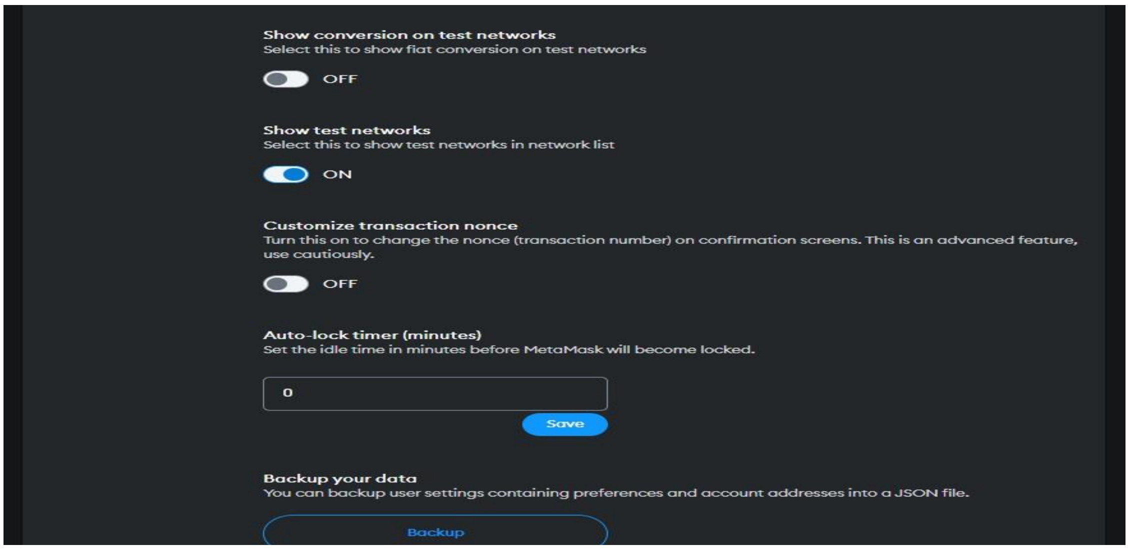
Step 10-> Click on Ethereum Mainnet button. Next click on Show/hide test networks.



0 ETH
$0.00 USD

Don't see your token?
Refresh list or import tokens

Need help? Contact MetaMask support



Show conversion on test networks
Select this to show fiat conversion on test networks
OFF

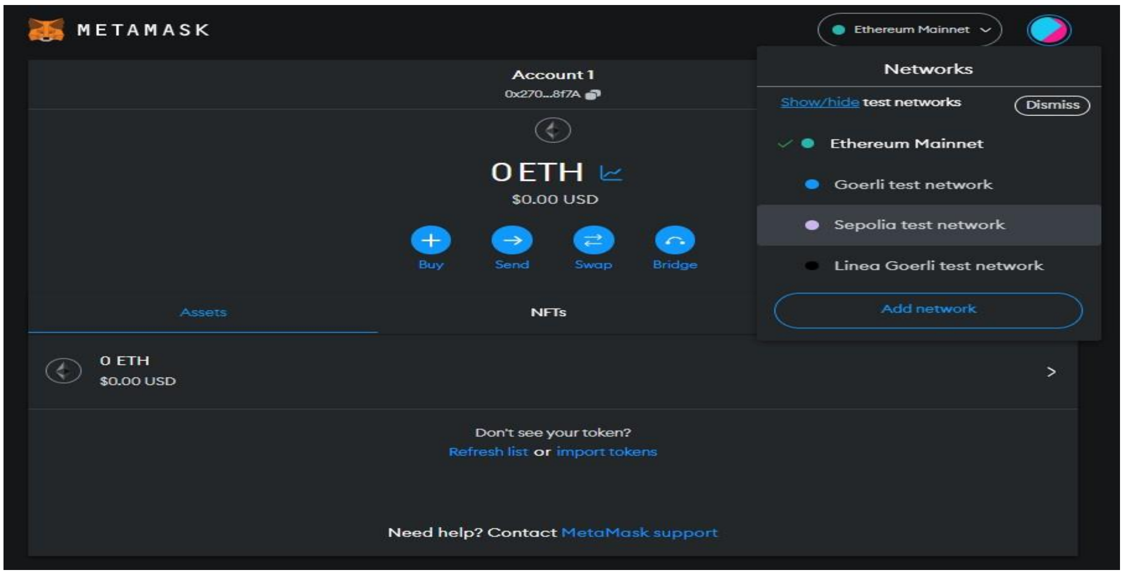
Show test networks
Select this to show test networks in network list
ON

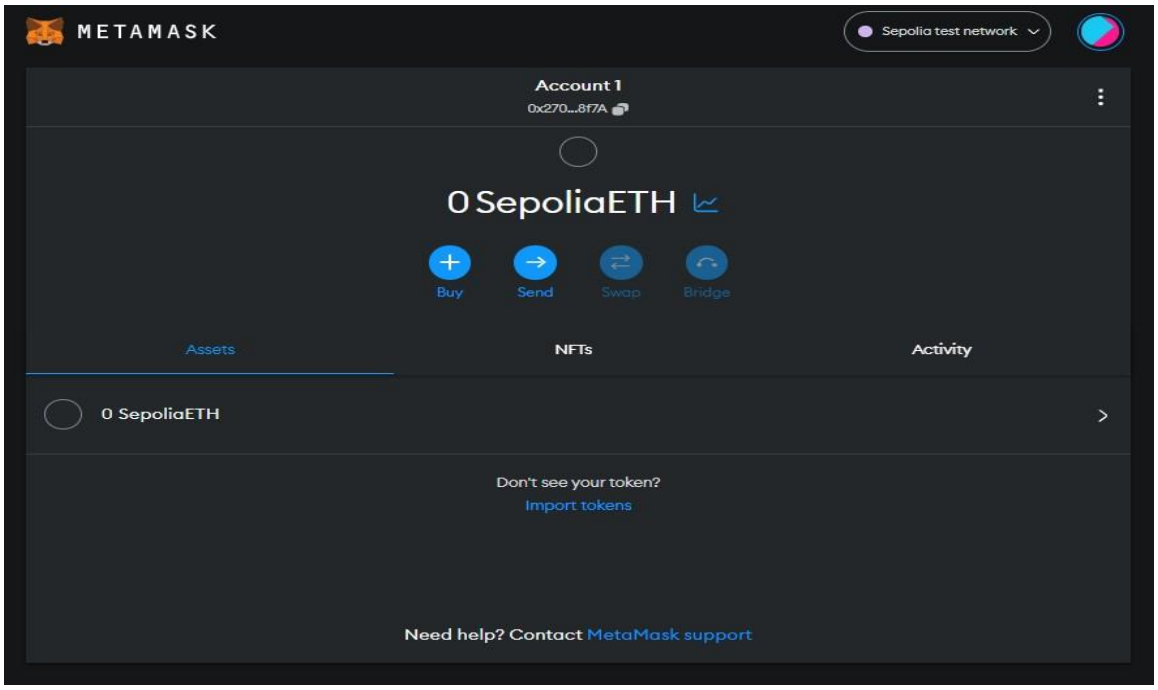
Customize transaction nonce
Turn this on to change the nonce (transaction number) on confirmation screens. This is an advanced feature, use cautiously.
OFF

Auto-lock timer (minutes)
Set the idle time in minutes before MetaMask will become locked.
0
Save

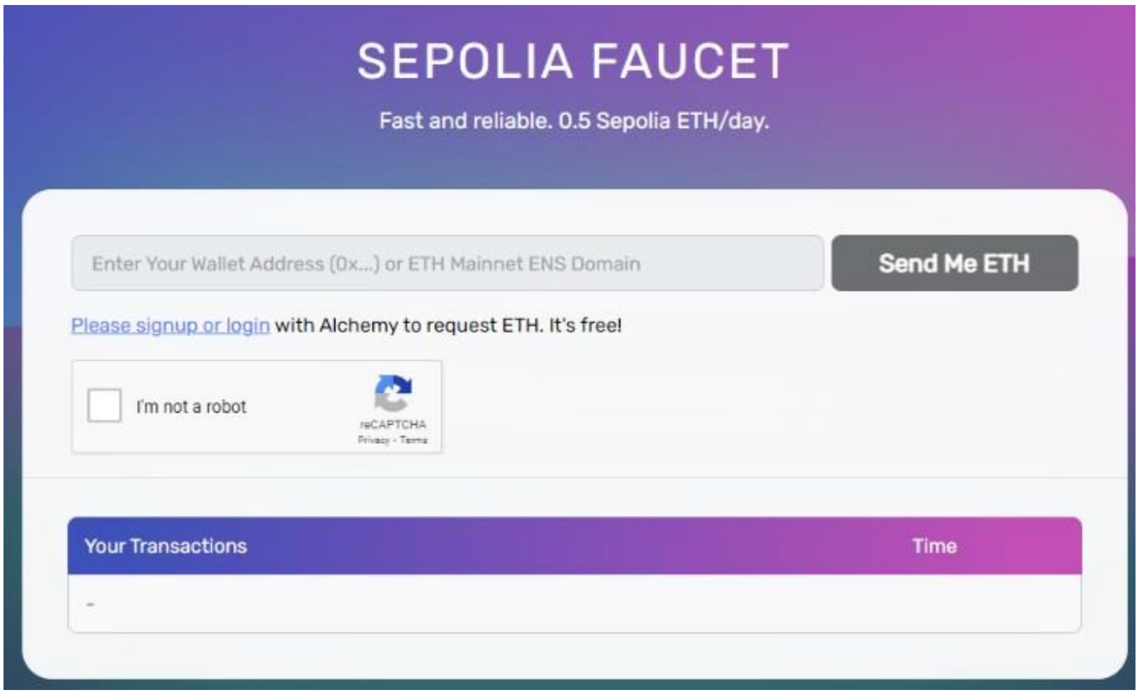
Backup your data
You can backup user settings containing preferences and account addresses into a JSON file.
Backup

Step 11-> Check if tesnets are shown by clicking on Etherum Mainnet button. Clickon Sepolia test network.

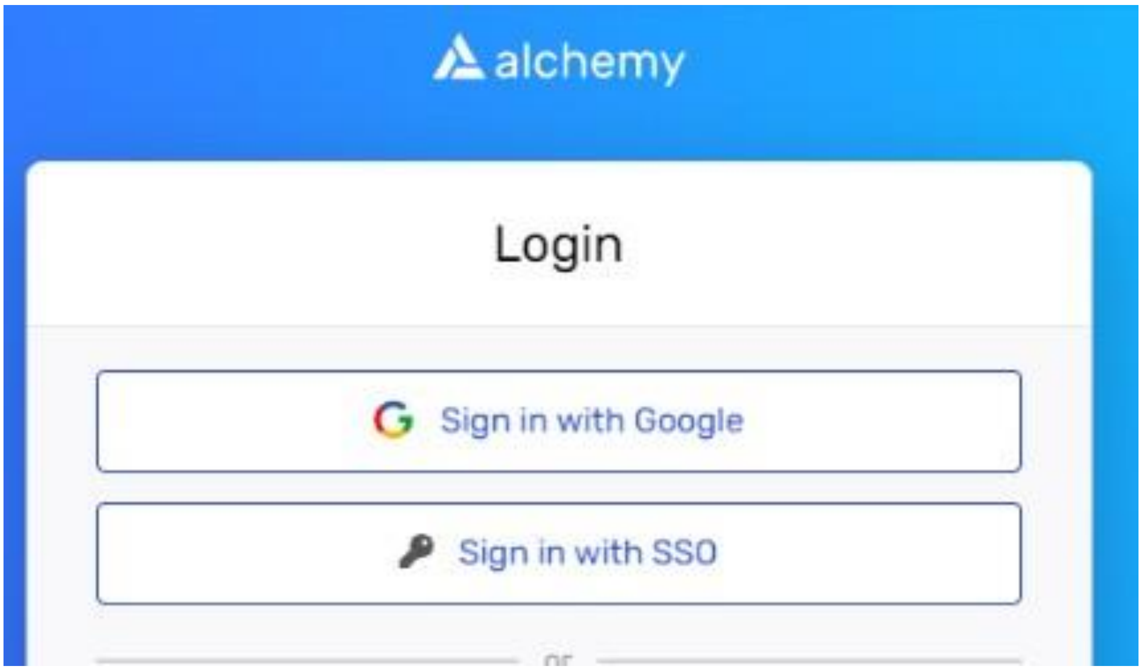




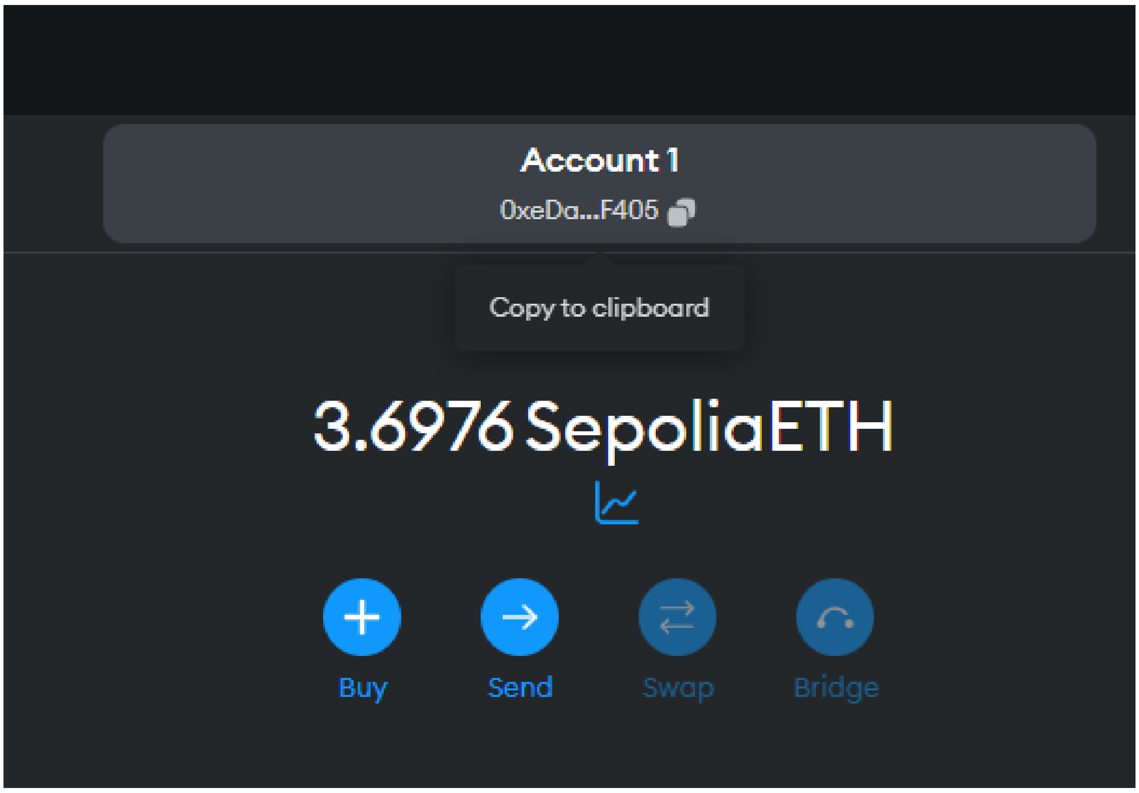
Step 12-> Go to https://sepoliafaucet.com/ and Click on Alchemy Login button.


SEPOLIA FAUCET
Fast and reliable. 0.5 Sepolia ETH/day.
Enter Your Wallet Address (0x...) or ETH Mainnet ENS Domain
Send Me ETH
Please signup or login with Alchemy to request ETH. It's free!
I'm not a robot
reCAPTCHA
Privacy - Terms
Your Transactions
Time
-

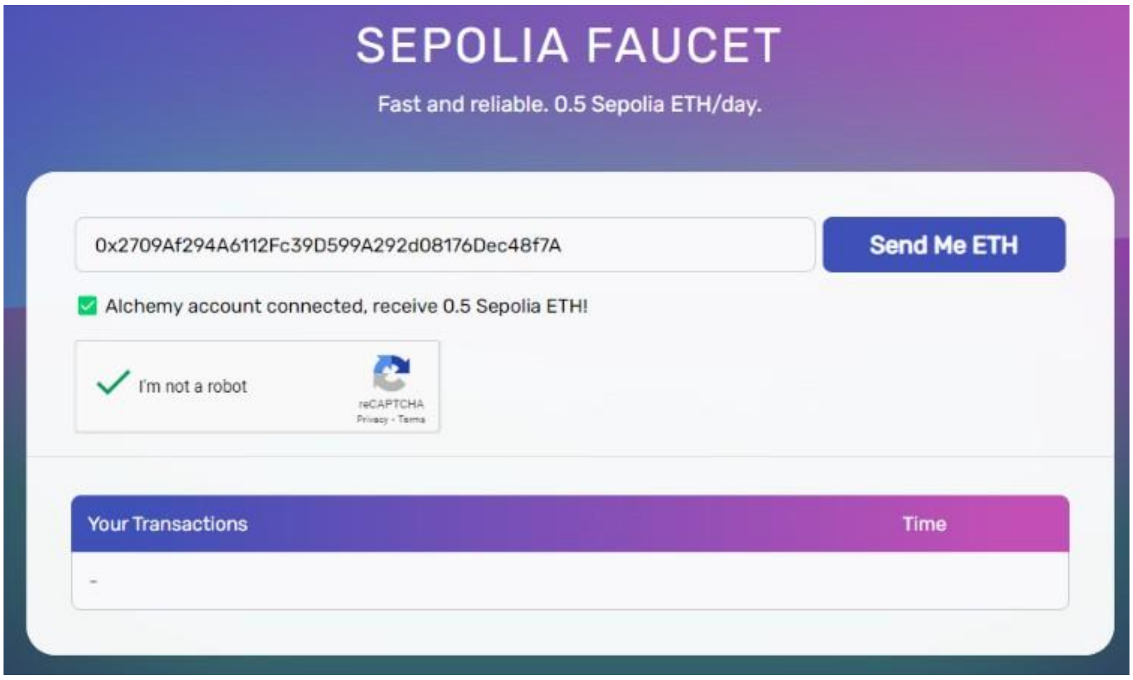
Step 13-> Login to a gmail account in another browser tab and click on Sign in withGoogle



Step 14-> Now go to MetaMask and copy the account address.



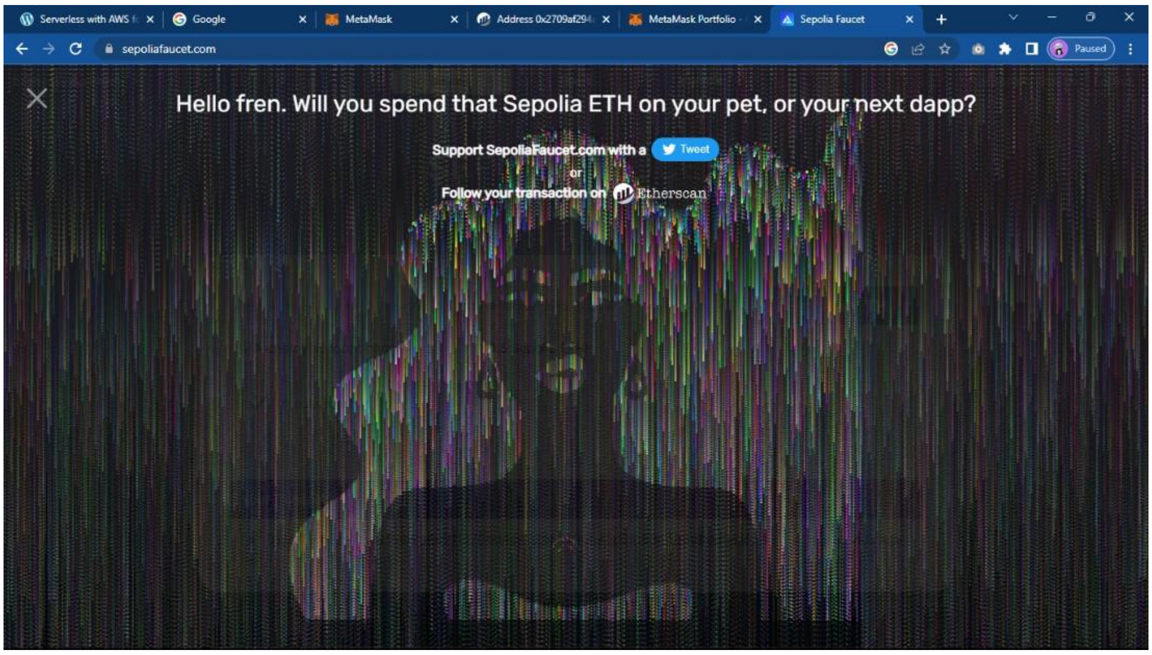
Step 15-> Paste the address and click on Send Me ETH.



0x2709Af294A6112Fc39D599A292d08176Dec48f7A

Alchemy account connected, receive 0.5 Sepolia ETH!

Step 16-> Your ETH transfer is succesfull. You should see a similar animation.



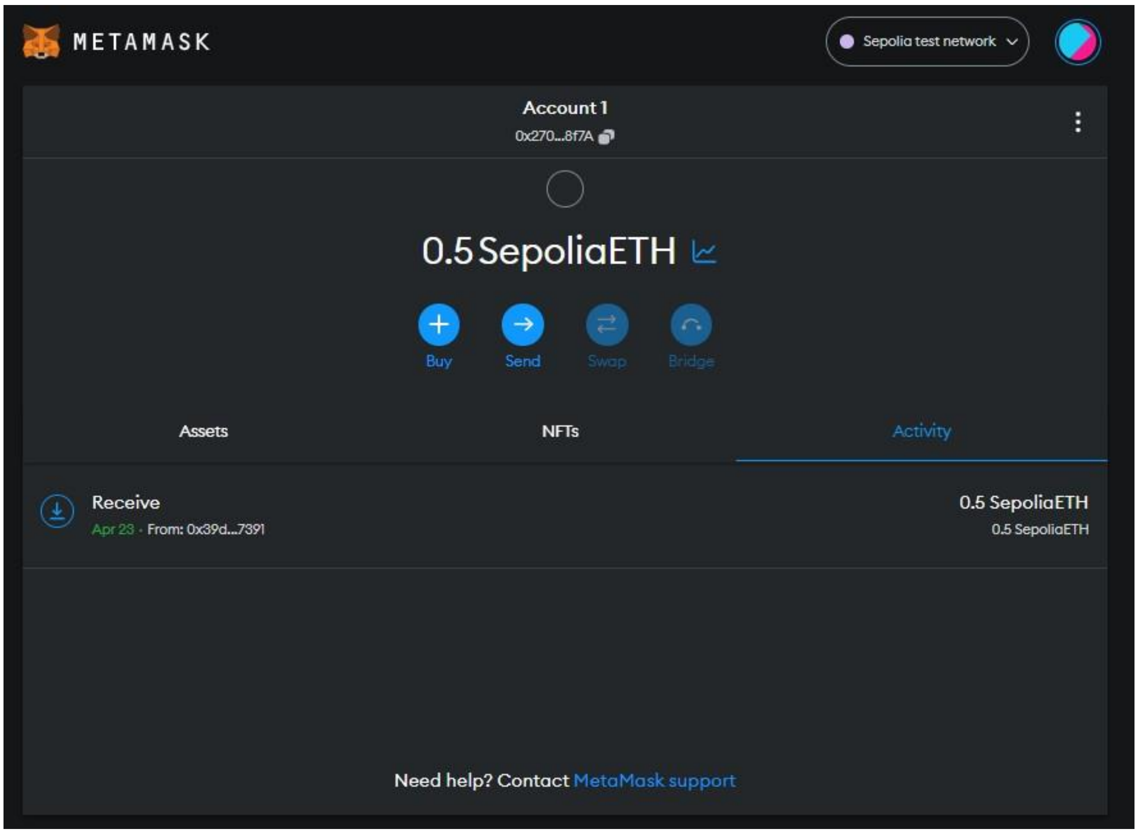
Hello fren. Will you spend that Sepolia ETH on your pet, or your next dapp?

Support SepoliaFaucet.com with a Tweet

or

Follow your transaction on Etherscan

Step 17-> Check your MetaMask account for Sepolia test network. 0.5 ETH will beadded.



Account 1

0x270...8f7A

0.5 SepoliaETH

Receive 0.5 SepoliaETH

Apr 23 · From: 0x39d...7391 0.5 SepoliaETH

Need help? Contact MetaMask support

PRACTICAL-3

Aim:IMPLEMENT AND DEMONSTRATE THE USE OF THE FOLLOWING IN SOLIDITY

1. To execute Solidity scripts go to ->https://remix.ethereum.org/

2. Open contracts folder and starting writing scripts. The scripts are compiled using Solidity compiler.

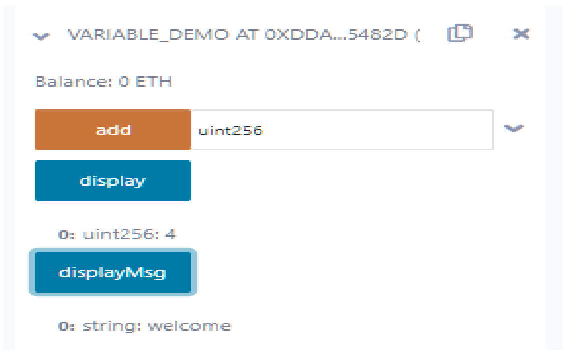
3. The following scripts were compiled using 0.5.0+commit.1d4f565a Solidity compiler

4. Deploy the scripts to execute code

A) Variable, Operators, Loops, Decision Making, Strings, Arrays, Enums, Structs,Mappings, Conversions, Ether Units, Special Variables

1. Variable

```
pragma solidity ^0.5.0;

contract variable_demo {
    uint256 sum = 4; //state variable
    uint256 x;
    address a;
    string s = "welcome";

    function add(uint256) public {
        uint256 y = 2; //local variable sum = sum+x+y:
        sum = sum + x + y;
    }

    function display() public view returns (uint256) {
        return sum;
    }

    function displayMsg() public view returns (string memory) {
        return s;
    }
}
```

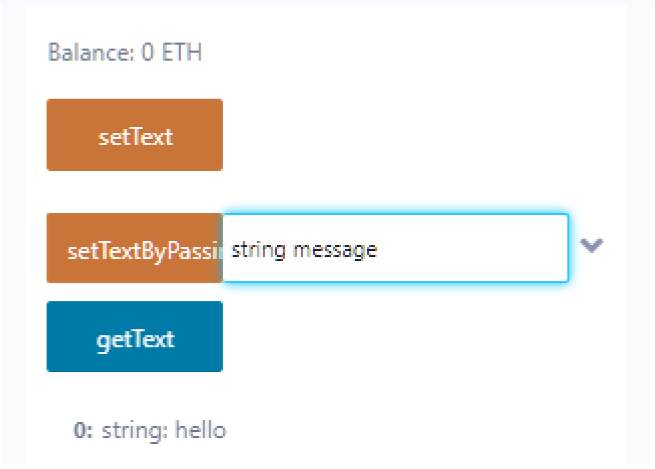


```
0: uint256: 4
0: string: welcome
```

Figure 1 -Displaying variable value

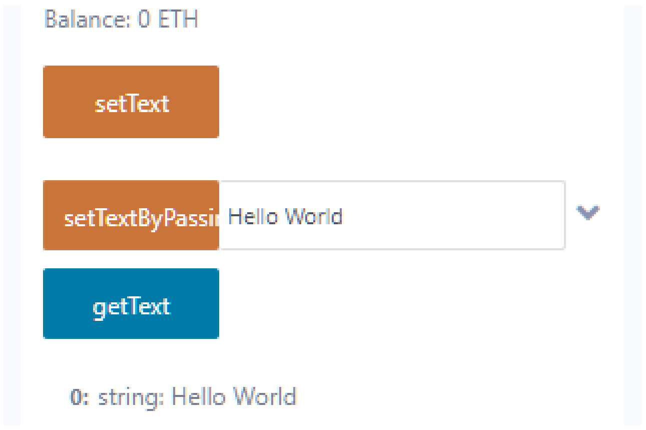
Strings

```
pragma solidity ^0.5.0;

contract LearningStrings {
    string text;

    function getText() public view returns (string memory) {
        return text;
    }

    function setText() public {
        text = "hello";
    }

    function setTextByPassing(string memory message) public {
        text = message;
    }
}
```



```
0: string: hello
```

Figure 2 - Before setting new string value

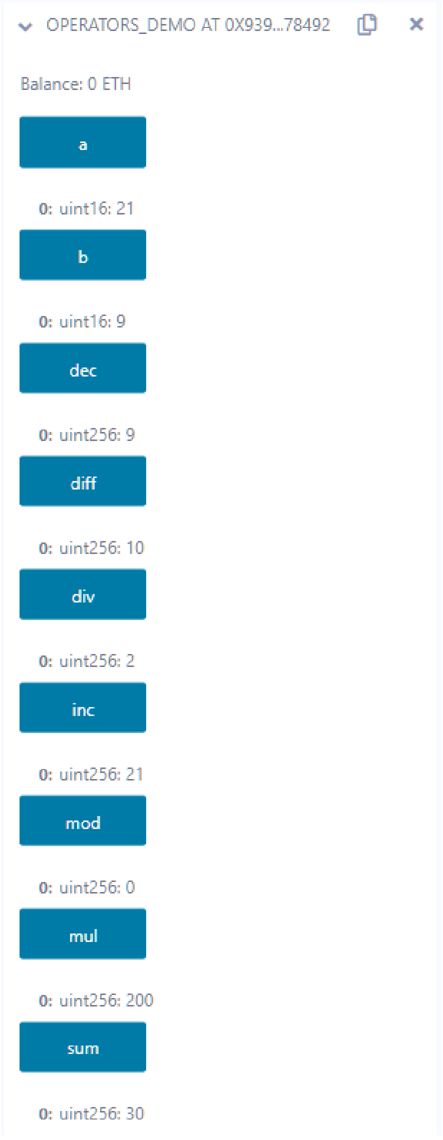


```
0: string: Hello World
```

Figure 3 - after setting string value

2. Operators

```
pragma solidity ^0.5.0;

contract SolidityTest {
    uint16 public a = 20;
    uint16 public b = 10;
    uint256 public sum = a + b;
    uint256 public diff = a - b;
    uint256 public mul = a * b;
    uint256 public div = a / b;
    uint256 public mod = a % b;
    uint256 public dec = --b;
    uint256 public inc = ++a;
}
```

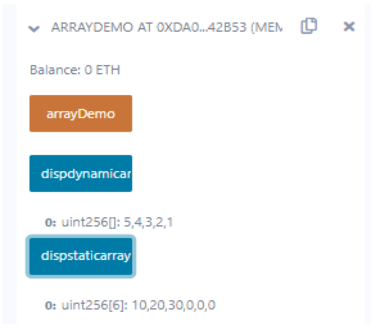


```
OPERATORS_DEMO AT 0X939...78492

Balance: 0 ETH

a
0: uint16: 21
b
0: uint16: 9
dec
0: uint256: 9
diff
0: uint256: 10
div
0: uint256: 2
inc
0: uint256: 21
mod
0: uint256: 0
mul
0: uint256: 200
sum
0: uint256: 30
```

FIGURE 4 - ALL OPERATORS OF SOLIDITY DISPLAYED

3. Array

```
 pragma solidity ^0.5.0;contract

 arraydemo

 {

         //Static Array

         uint[6] arr2=[10,20,30];

         function dispstaticarray() public view returns(uint[6] memory)

         {

                 return arr2;

         }

         //Dynamic Array

         uint x=5;

         uint [] arr1;

         function arrayDemo() public

         {
              while(x>0)

                     {
                      arr1.push(x);x=x1;

}

         }

function dispdynamicarray()

 public  viewreturns(uint[]memory)

{

return arr1;

}

 }
```



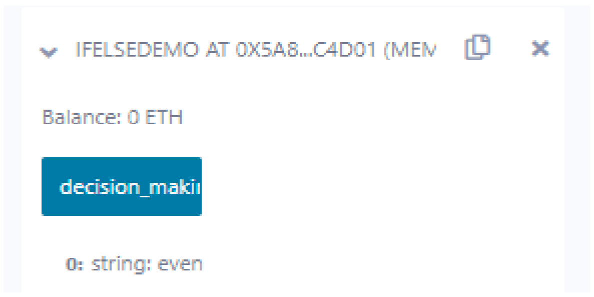
```
ARRAYDEMO AT 0XDA0...42B53 (MEM
Balance: 0 ETH
arrayDemo
dispdynamicar
0: uint256[]: 5,4,3,2,1
dispstaticarray
0: uint256[6]: 10,20,30,0,0,0
```

FIGURE 5 - ARRAY DISPLAYED

4. Decision Making

If Else

```
pragma solidity ^0.5.0;contract

ifelsedemo

{

    uint i=10;

    function decision_making() public view returns(string memory)

    {

        if(i%2==0)

        {

            return "even";

        }

        else

        {

            return "Odd";

        }

    }

}
```



```
IFELSEDEMO AT 0X5A8...C4D01 (MEM

Balance: 0 ETH

decision_maki

0: string: even
```

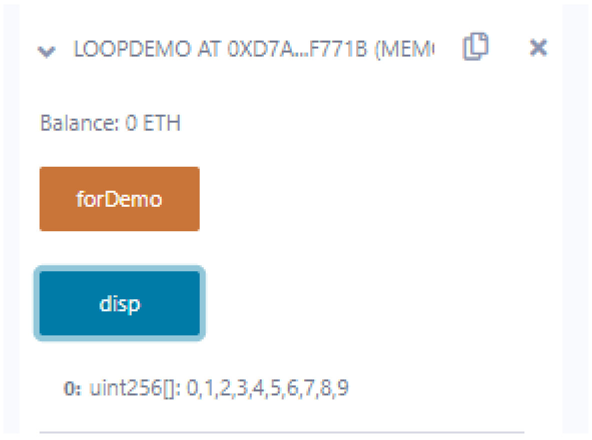
Figure 6 - if else output

5. Loops

For Loop

For Loop

```
pragma solidity ^0.5.0;contract
loopDemo
{
    uint [] data;
    function forDemo() public returns(uint[] memory)
    {
        for(uint i=0; i<10; i++){
            data.push(i);
        }
        return data;
    }
    function disp() public view returns(uint[] memory)
    {
        return data;
    }
}
```

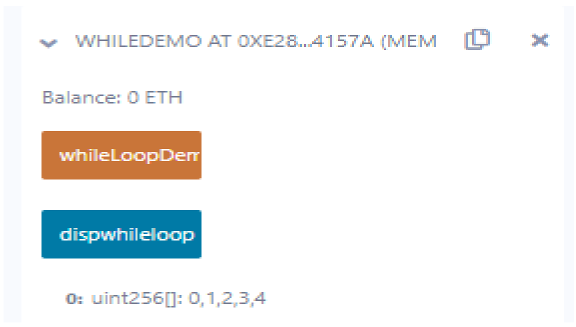


```
0: uint256[]: 0,1,2,3,4,5,6,7,8,9
```

FIGURE 7 - APPENDING VALUES TO ARRAY USING FOR LOOP

While Loop

```
pragma solidity ^0.5.0;contract

whiledemo

{

    uint [] data;uint

    x=0;



    function whileLoopDemo() public

    {

        while(x<5)

        {

            data.push(x);

            x=x+1;

        }

    }



    function dispwhileloop() public view returns(uint[] memory)

    {

        return data;

    }

}
```

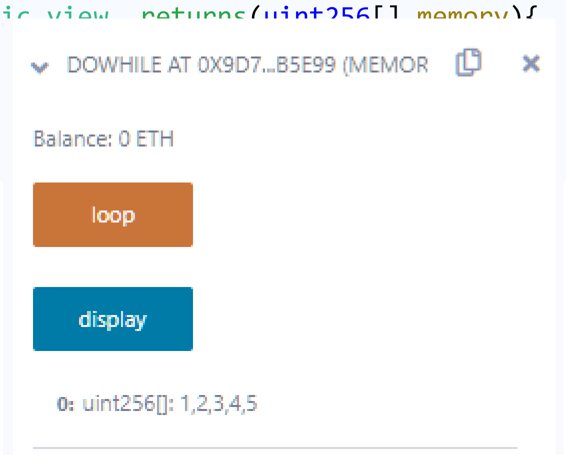


```
Balance: 0 ETH
0: uint256[]: 0,1,2,3,4
```

Figure 8 - appending values to array using while loop

Do While

```
pragma solidity ^0.5.0;

// Creating a contract
contract DoWhile {
    // Declaring a dynamic array
    uint256[] data;

    // Declaring state variable
    uint8 j = 0;

    // Defining function to demonstrate
    // 'Do-While loop'
    function loop() public returns (uint256[] memory) {
        do {
            j++;
            data.push(j);
        } while (j < 5);
        return data;
    }
    function display() public view  returns(uint256[] memory){
        return data;
    }
}
```


DOWHILE AT 0X9D7...B5E99 (MEMOR

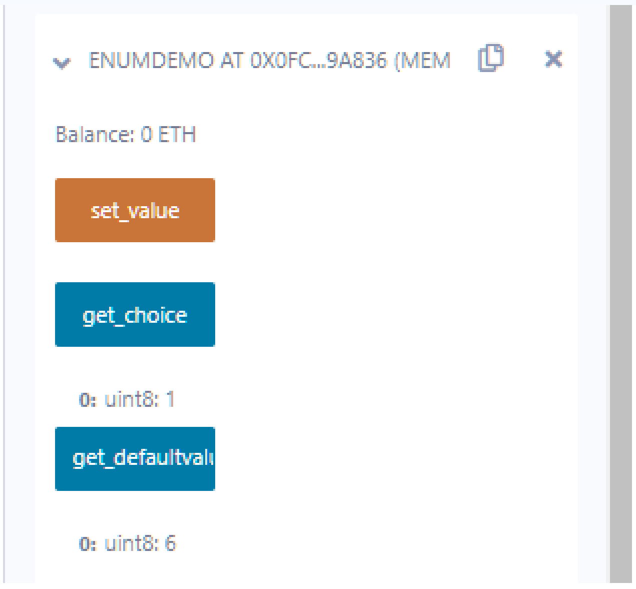
Balance: 0 ETH

```
0: uint256[]: 1,2,3,4,5
```

Figure 9 appending values to array using do while loop

6. Enums

```
pragma solidity ^0.5.0;

contract enumdemo {
    enum week_days {
        Monday,
        Tuesday,
        Wednesday,
        Thursday,
        Friday,
        Saturday,
        Sunday
    }

    week_days week;
    week_days choice;
    week_days constant default_value = week_days.Sunday;

    function set_value() public {
        choice = week_days.Tuesday;
    }

    function get_choice() public view returns (week_days) {
        return choice;
    }

    function get_defaultvalue() public view returns (week_days) {
        return default_value;
    }
}
```



Balance: 0 ETH

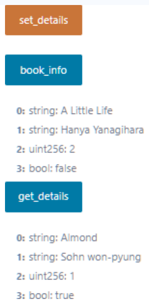
```
0: uint8: 1
```

```
0: uint8: 6
```

FIGURE 10 - ACCESSING ENUM VALUES

7. Structs

```
pragma solidity ^0.5.0;

contract structdemo {
    struct Book {
        string name;
        string author;
        uint256 id;
        bool availability;
    }
    Book book2;
    Book book1 = Book("A Little Life", "Hanya Yanagihara", 2, false);

    function set_details() public {
        book2 = Book("Almond", "Sohn won-pyung", 1, true);
    }

    function book_info()
        public
        view
        returns (
            string memory,
            string memory,
            uint256,
            bool
        )
    {
        return (book1.name, book1.author, book1.id, book1.availability);
    }

    function get_details()
        public
        view
        returns (
            string memory, string memory, uint256, bool
        )
    {
        return (book2.name, book2.author, book2.id, book2.availability);
    }
}
```



```
0: string: A Little Life
1: string: Hanya Yanagihara
2: uint256: 2
3: bool: false
```

```
0: string: Almond
1: string: Sohn won-pyung
2: uint256: 1
3: bool: true
```

FIGURE 11- STRUCTURE DATATYPE IN SOLIDITY

8. Mappings

```
pragma solidity ^0.5.0;

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();
        return ledgerBalance.balances(address(this));
    }
}
```

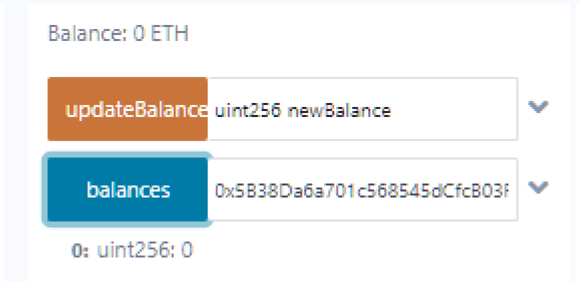


FIGURE 12 - BEFORE UPDATING BALANCE

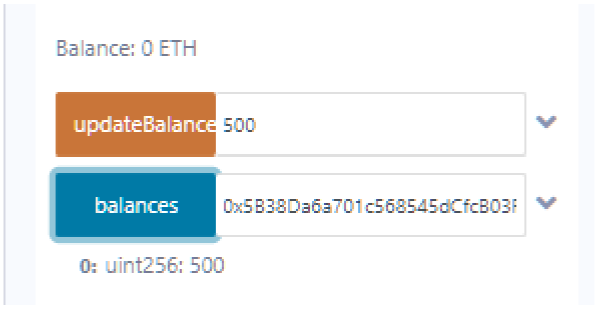
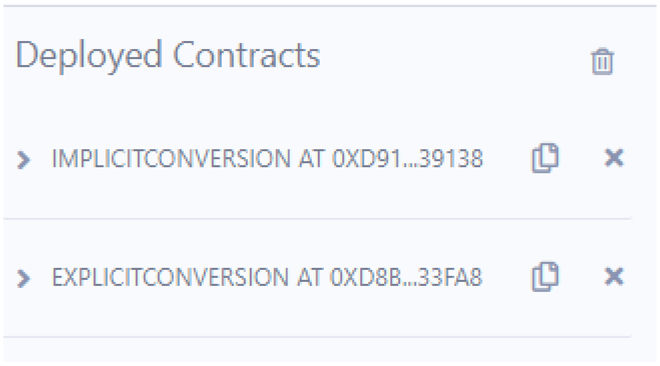


FIGURE 13 - AFTER UPDATING BALANCE

9. Conversions

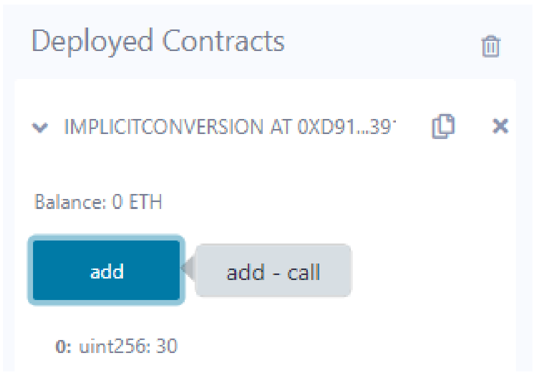
```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract ImplicitConversion {
    function add() public pure returns (uint256) {
        uint256 a = 10;
        uint256 b = 20;
        return a + b;
    }
}

contract ExplicitConversion {
    function convert() public pure returns (bytes memory) {
        string memory str = "Hello World";
        bytes memory b = bytes(str);
        return b;
    }
}
```

Step 1-> Deploy both contracts



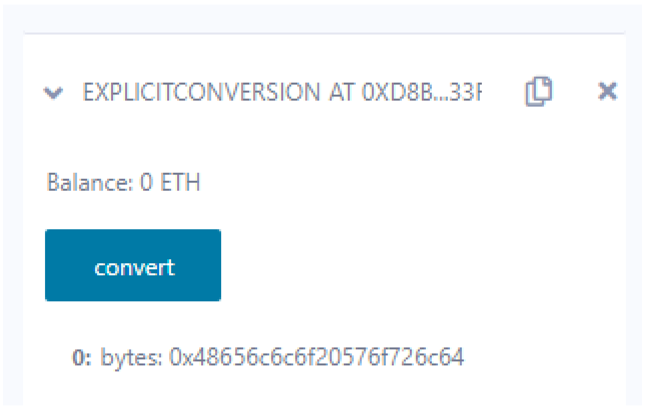
Deployed Contracts
IMPLICITCONVERSION AT 0XD91...39138
EXPLICITCONVERSION AT 0XD8B...33FA8

Step 2-> Open Implicit Conversion and click on add button to sum and displayvalue



```
0: uint256: 30
```

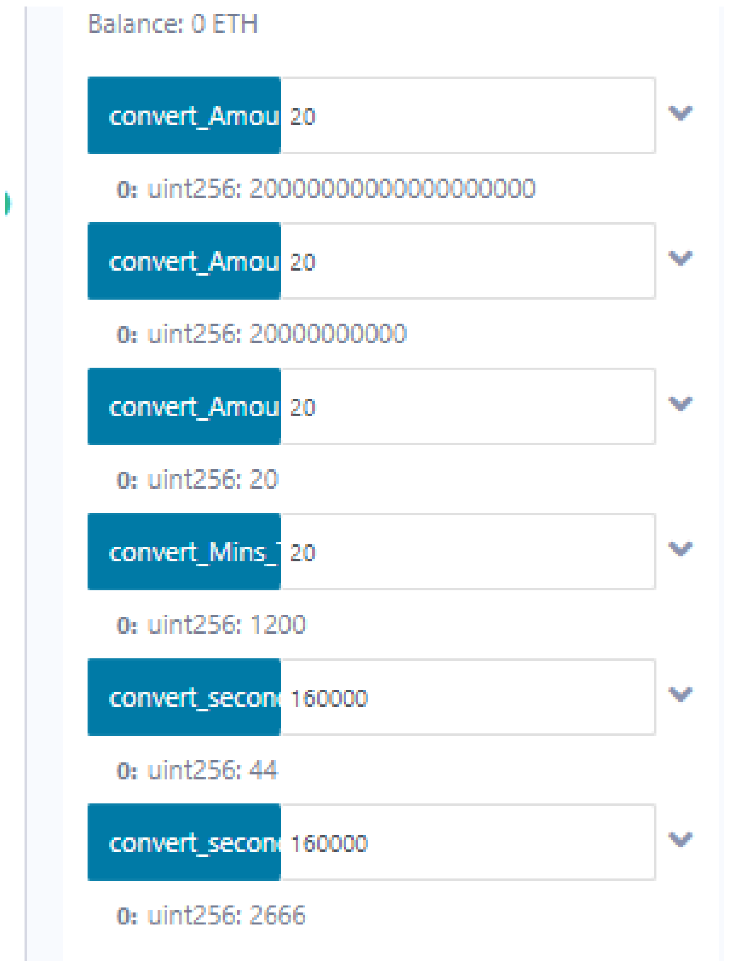
Step 3-> Open Explicit Conversion and click on convert button



0: bytes: 0x48656c6c6f20576f726c64

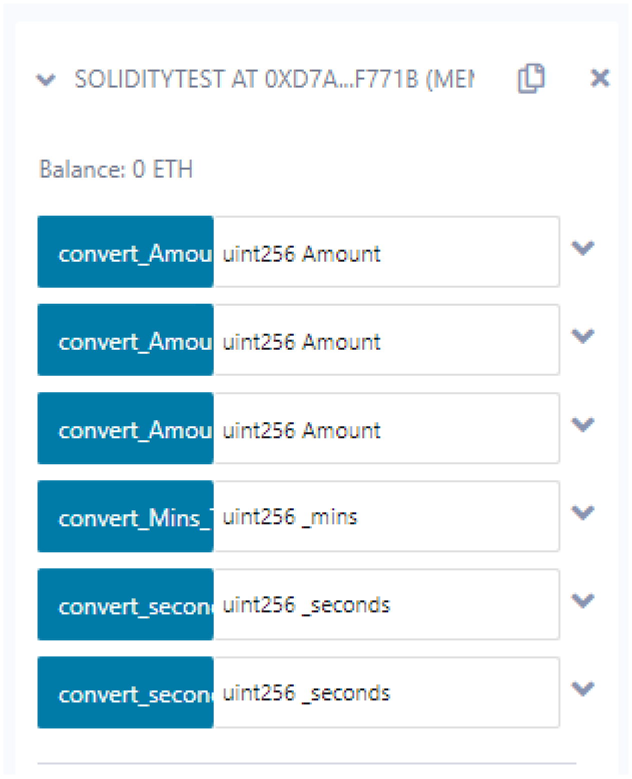
10.Ether Units

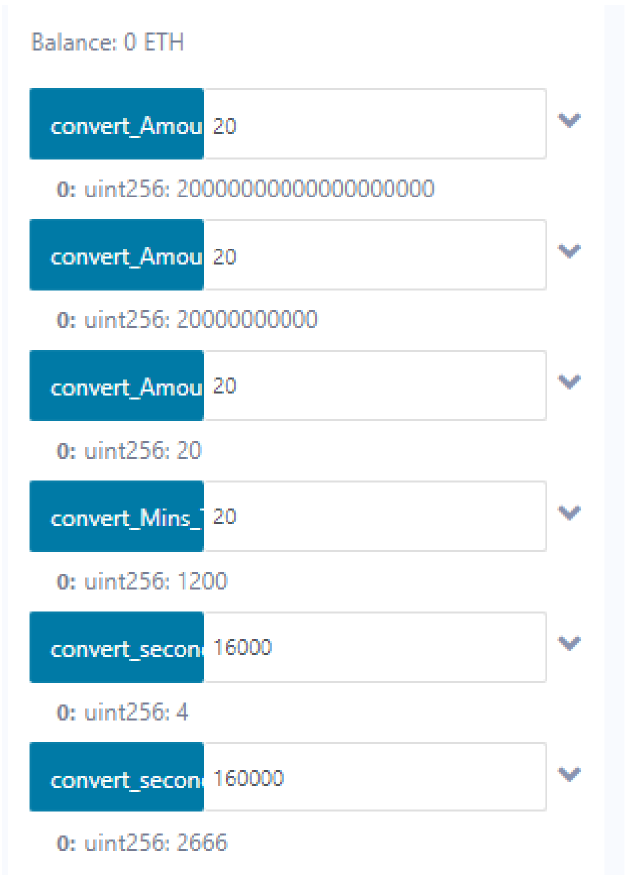
```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract SolidityTest {
    function convert_Amount_to_Wei(uint256 Amount)
        public
        pure
        returns (uint256)
    {
        return Amount * 1 wei;
    }

    function convert_Amount_To_Ether(uint256 Amount)
        public
        pure
        returns (uint256)
    {
        return Amount * 1 ether;
    }

    function convert_Amount_To_Gwei(uint256 Amount)
        public
        pure
        returns (uint256)
    {
        return Amount * 1 gwei;
    }

    function convert_seconds_To_mins(uint256 _seconds)
        public
        pure
        returns (uint256)
    {
        return _seconds / 60;
    }

    function convert_seconds_To_Hours(uint256 _seconds)
        public
        pure
        returns (uint256)
    {
        return _seconds / 3600;
    }

    function convert_Mins_To_Seconds(uint256 _mins)
        public
        pure
        returns (uint256)
    {
        return _mins * 60;
    }
}
```



```
Balance: 0 ETH

convert_Amou  20
0: uint256: 20000000000000000000
convert_Amou  20
0: uint256: 20000000000
convert_Amou  20
0: uint256: 20
convert_Mins_T  20
0: uint256: 1200
convert_secon  160000
0: uint256: 44
convert_secon  160000
0: uint256: 2666
```

Step 1-> Provide values to each function and click on them


SOLIDITYTEST AT 0XD7A...F771B (MEI
Balance: 0 ETH
convert_Amou uint256 Amount
convert_Amou uint256 Amount
convert_Amou uint256 Amount
convert_Mins_ uint256 _mins
convert_secon uint256 _seconds
convert_secon uint256 _seconds


Balance: 0 ETH

convert_Amou 20

0: uint256: 20000000000000000000

convert_Amou 20

0: uint256: 20000000000

convert_Amou 20

0: uint256: 20

convert_Mins_ 20

0: uint256: 1200

convert_secon 16000

0: uint256: 4

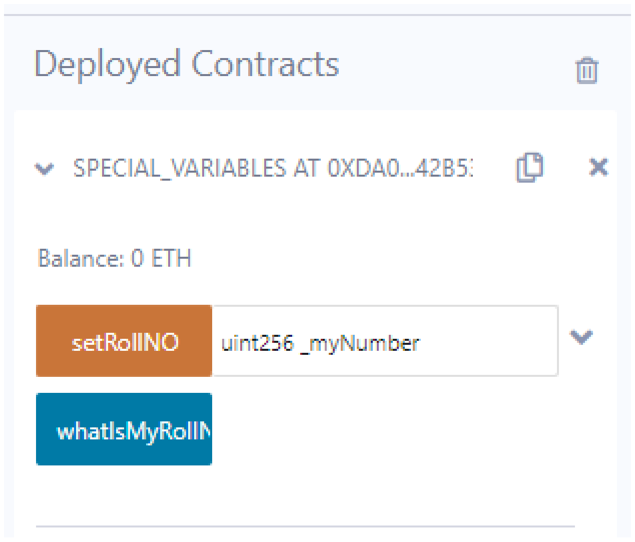
convert_secon 160000

0: uint256: 2666

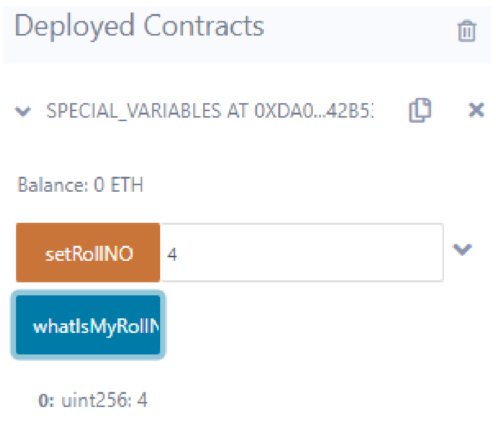
11.Special Variables

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

contract Special_Variables {
    mapping(address => uint256) rollNo;

    function setRollNO(uint256 _myNumber) public {
        rollNo[msg.sender] = _myNumber;
    }

    function whatIsMyRollNumber() public view returns (uint256) {
        return rollNo[msg.sender];
    }
}
```

Step 1-> Deploy contract Special Variables



Step 2-> Input a number for setRollNO function and click on it &whatIsMyRollNumber button

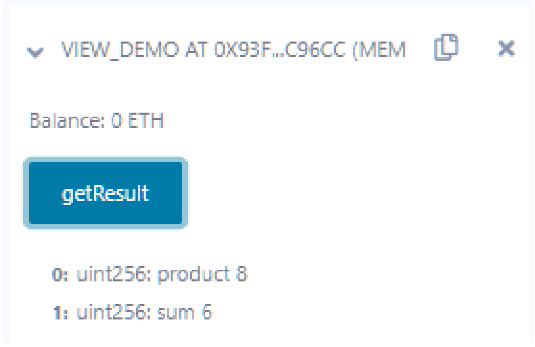


```
0: uint256: 4
```

B) Functions, Function Modifiers, View functions, Pure Functions, Fallback Function,Function Overloading, Mathematical functions, Cryptographic functions

1. View Functions

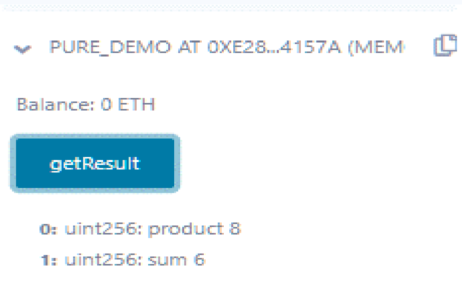
```
pragma solidity ^0.5.0;

contract view_demo {
    uint256 num1 = 2;
    uint256 num2 = 4;

    function getResult() public view returns (uint256 product, uint256 sum) {
        product = num1 * num2;
        sum = num1 + num2;
    }
}
```



```
VIEW_DEMO AT 0X93F...C96CC (MEM

Balance: 0 ETH

getResult

0: uint256: product 8
1: uint256: sum 6
```

Pure Functions

```
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;
    }
```

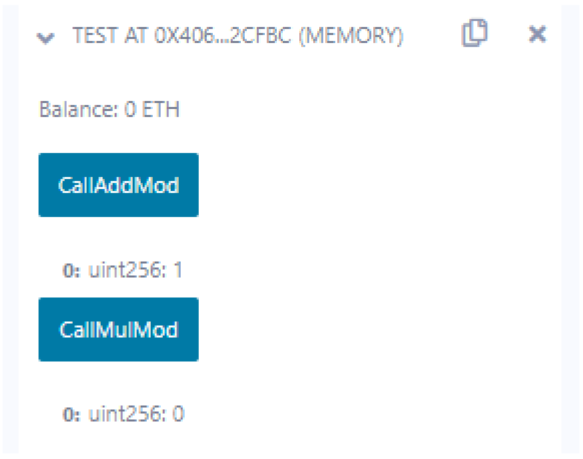


```
PURE_DEMO AT 0XE28...4157A (MEM

Balance: 0 ETH

getResult

0: uint256: product 8
1: uint256: sum 6
```

Figure 15 - pure function output

2. Mathematical Functions

```
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);

        }

}
```

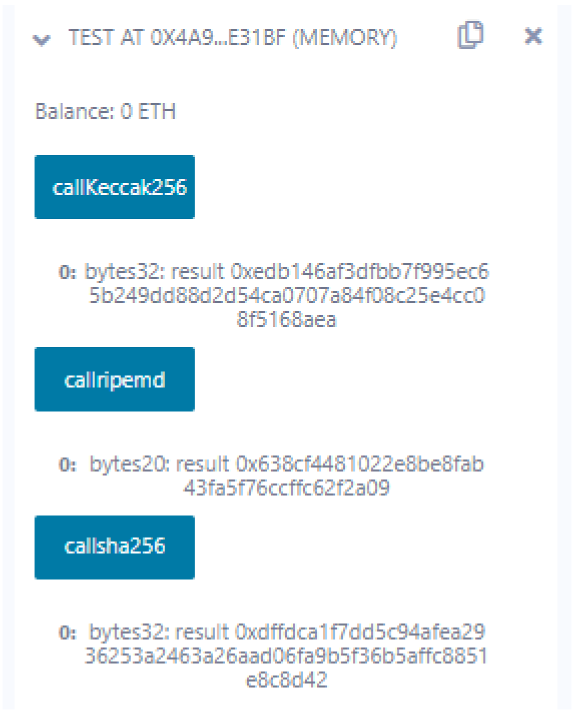


```
Balance: 0 ETH
0: uint256: 1
0: uint256: 0
```

FIGURE 16 - MATHEMATICAL FUNCTIONS IN SOLIDITY

3. Cryptographic Functions

```
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");

        }

}
```



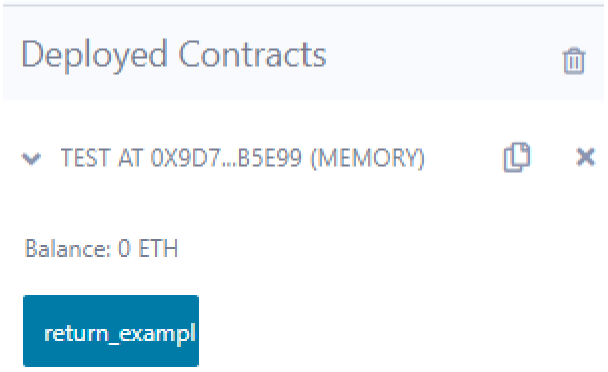
```
TEST AT 0X4A9...E31BF (MEMORY)

Balance: 0 ETH

callKeccak256

0: bytes32: result 0xedb146af3dfbb7f995ec6
   5b249dd88d2d54ca0707a84f08c25e4cc0
   8f5168aea

callripemd

0: bytes20: result 0x638cf4481022e8be8fab
   43fa5f76ccffc62f2a09

callsha256

0: bytes32: result 0xdffdca1f7dd5c94afea29
   36253a2463a26aad06fa9b5f36b5affc8851
   e8c8d42
```

FIGURE 17 - CRYPTOGRAPHY ALGORITHMS IN SOLIDITY

4. Functions

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.4.22 <0.9.0;

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);
    }
}
```

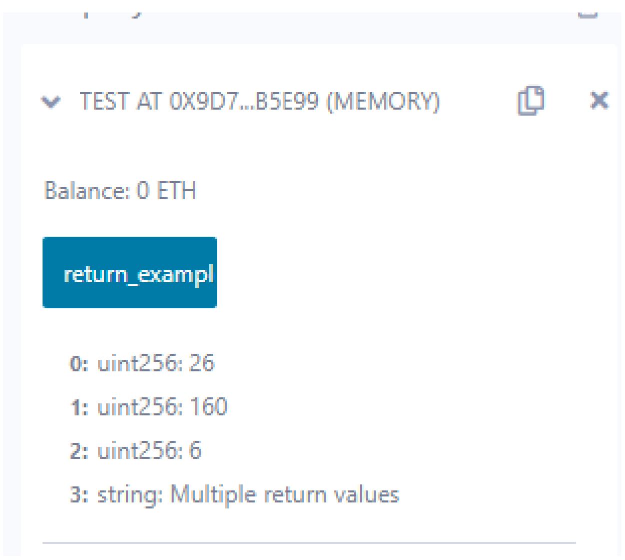
Step 1-> Deploy Test Contract


Deployed Contracts

TEST AT 0X9D7...B5E99 (MEMORY)

Balance: 0 ETH

Step 2-> Click on return_example button to display all values

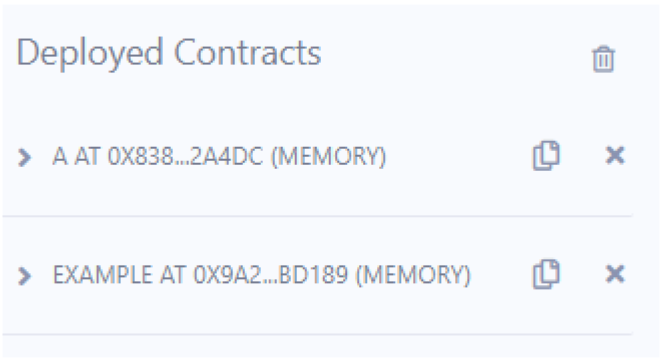


```
Balance: 0 ETH

0: uint256: 26
1: uint256: 160
2: uint256: 6
3: string: Multiple return values
```

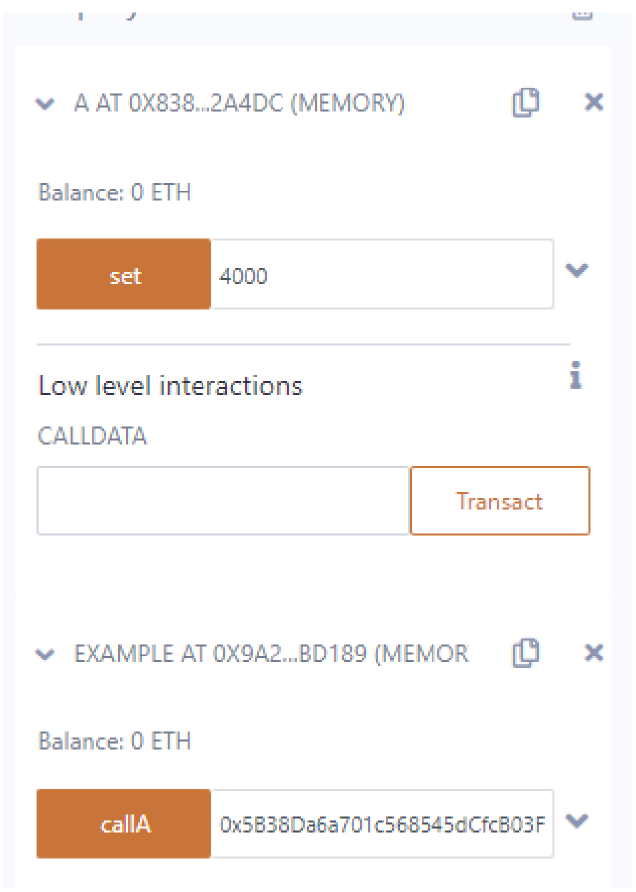
5. Fallback Function

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.12;

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));
    }
}
```

Step 1-> Deploy both A & example contracts


Deployed Contracts
A AT 0X838...2A4DC (MEMORY)
EXAMPLE AT 0X9A2...BD189 (MEMORY)

Step 2-> Provide values to both deployed contracts accordingly(use any address)


A AT 0X838...2A4DC (MEMORY)

Balance: 0 ETH

set 4000

Low level interactions

CALLDATA

Transact

EXAMPLE AT 0X9A2...BD189 (MEMOR

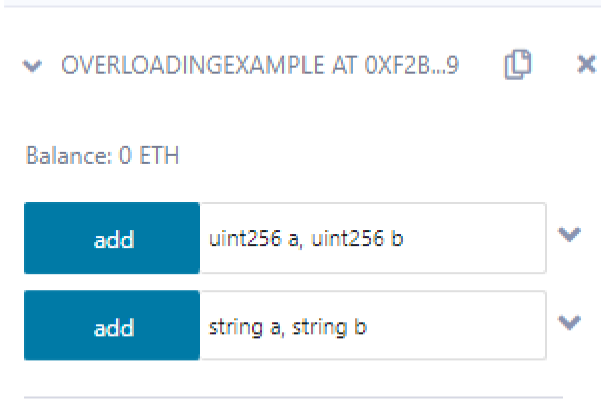
Balance: 0 ETH

callA 0x5B38Da6a701c568545dCfcB03F

6. Function Overloading

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

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));
    }
}
```

Step 1-> Deploy Overloading Example contract



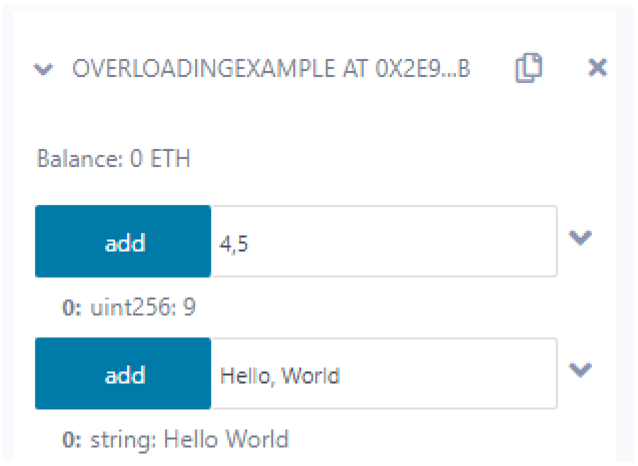
OVERLOADINGEXAMPLE AT 0XF2B...9

Balance: 0 ETH

add uint256 a, uint256 b

add string a, string b

Step 2-> Give integer and string values to both add functions as below



OVERLOADINGEXAMPLE AT 0X2E9...B

Balance: 0 ETH

add 4,5

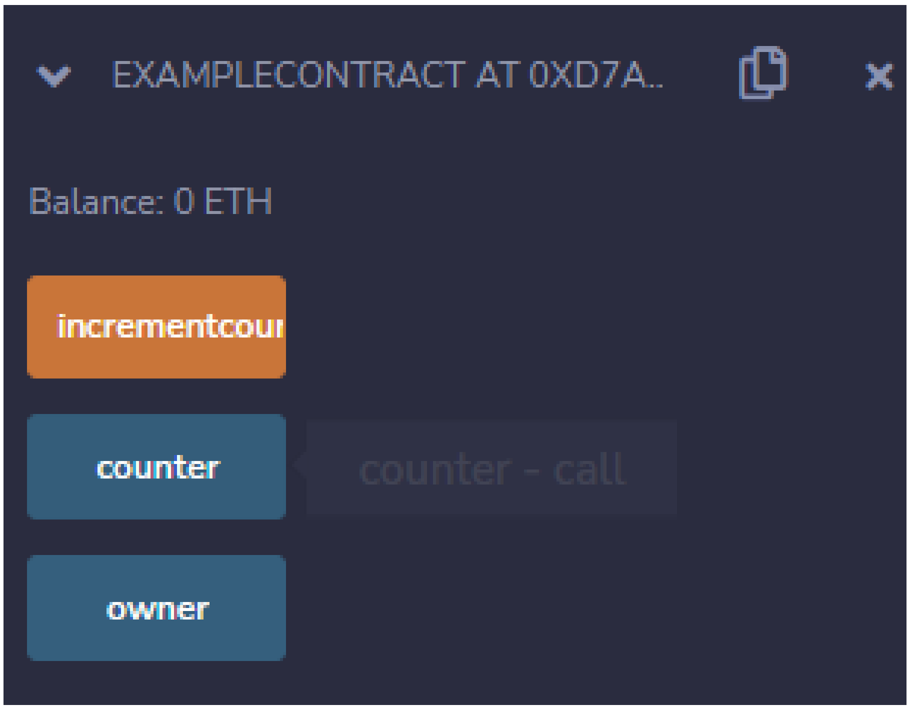
0: uint256: 9

add Hello, World

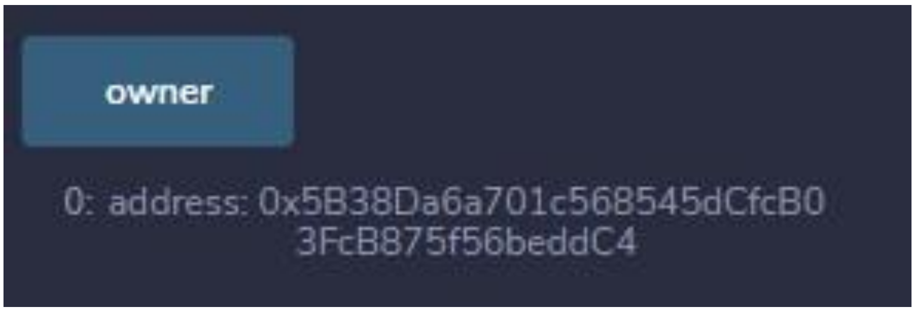
0: string: Hello World

7. 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++;
    }
}
```

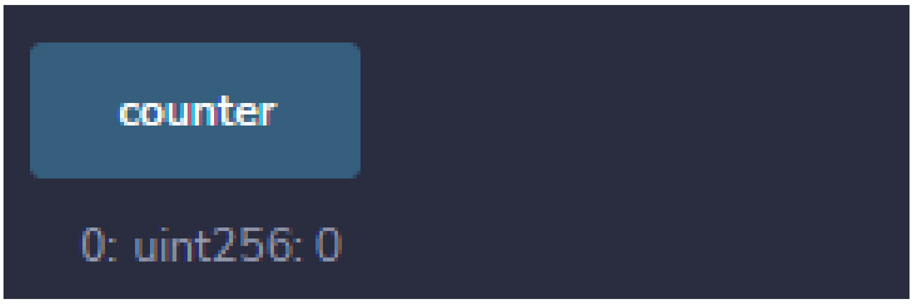


Step 1-> Click on owner button



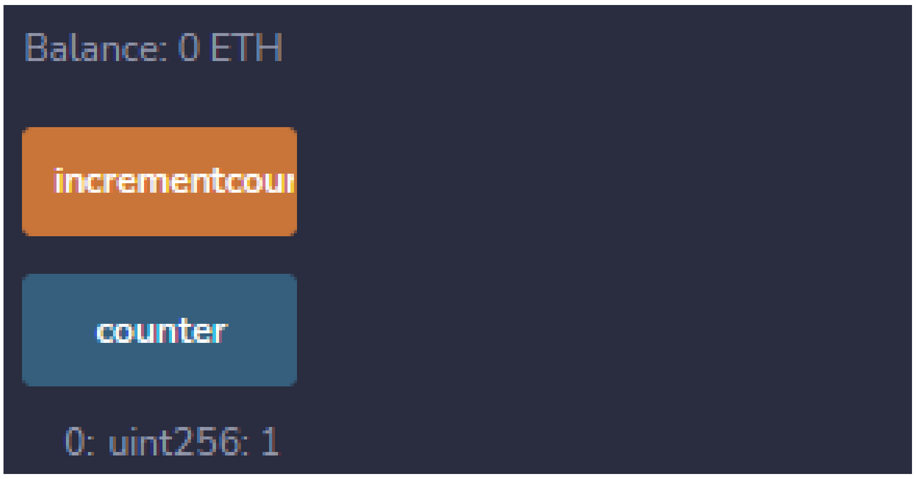
0: address: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4

Step 2-> Click on counter button initially it is 0.



0: uint256: 0

Step 3-> Then click on increment counter button and again click on counterbutton, the counter has been increased



Balance: 0 ETH

0: uint256: 1

PRACTICAL-4

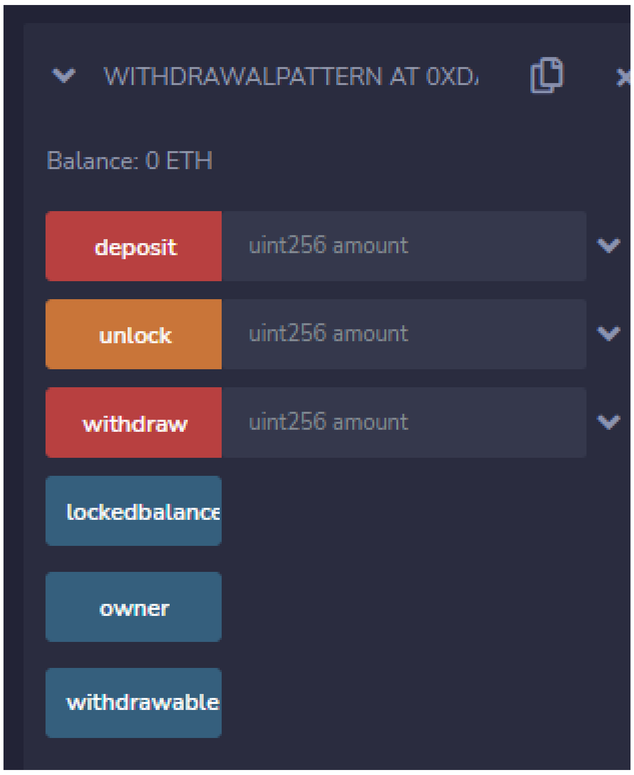
Aim: IMPLEMENT AND DEMONSTRATE THE USE OF THE FOLLOWING IN SOLIDITY

A) Withdrawal Pattern, Restricted Access

1) Withdrawal Pattern

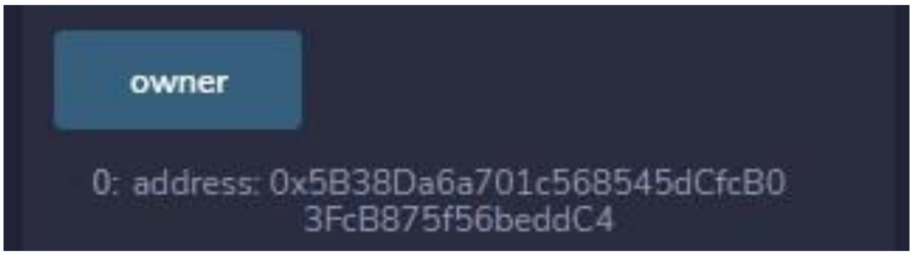
```
// SPDX-License-Identifier: MIT
pragma solidity 0.8.18;

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;
    }
}
```

Outputs:



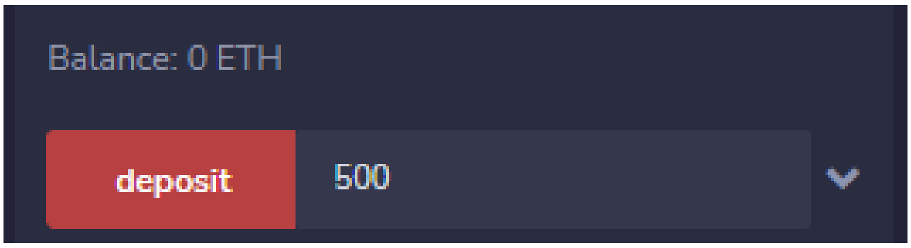
Flow of execution

Step 1-> Click on owner

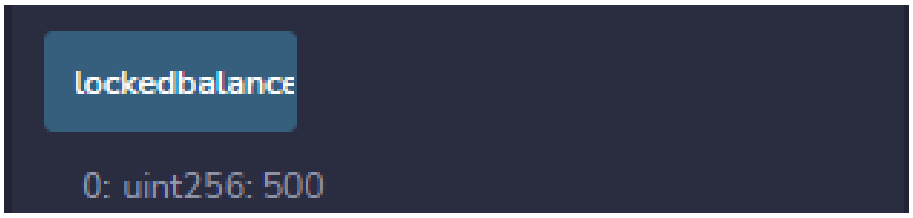


0: address: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4

Step 2-> Enter an amount and click on deposit

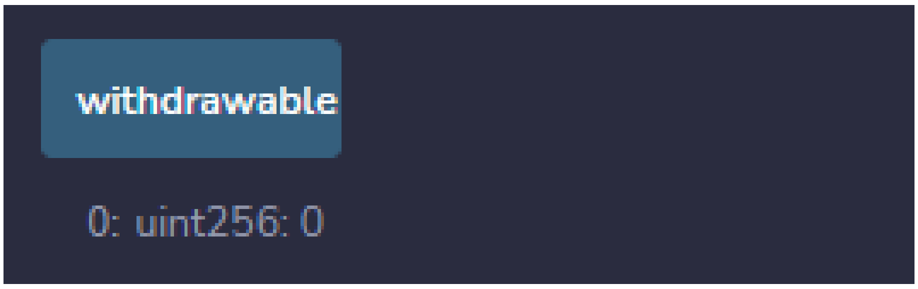


Step 3-> Click on locked balance button to display the locked amount in the account



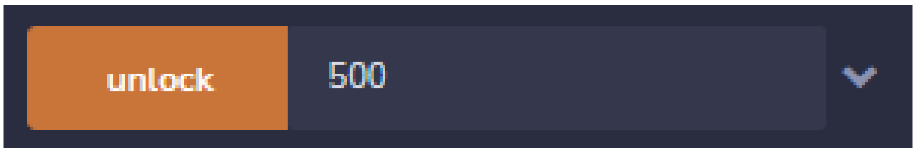
```
0: uint256: 500
```

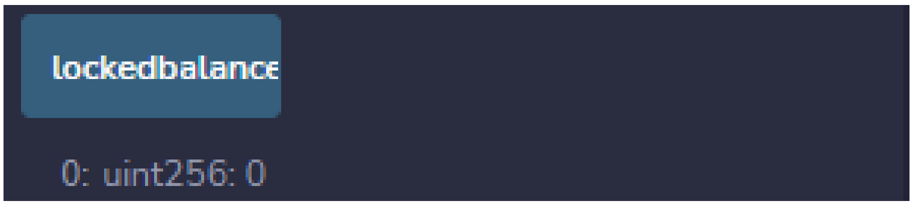
Step 4-> Click on withdrawable balance button



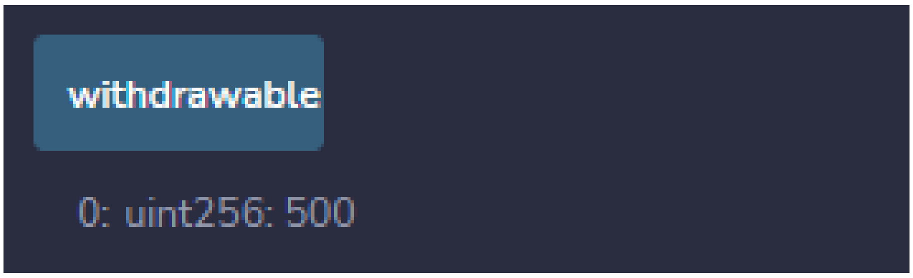
```
0: uint256: 0
```

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



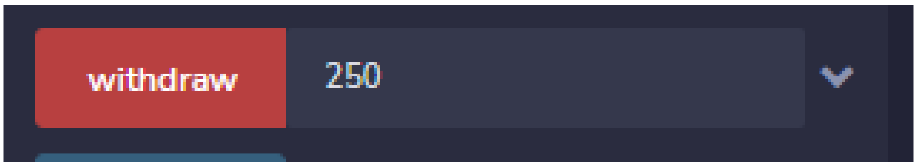


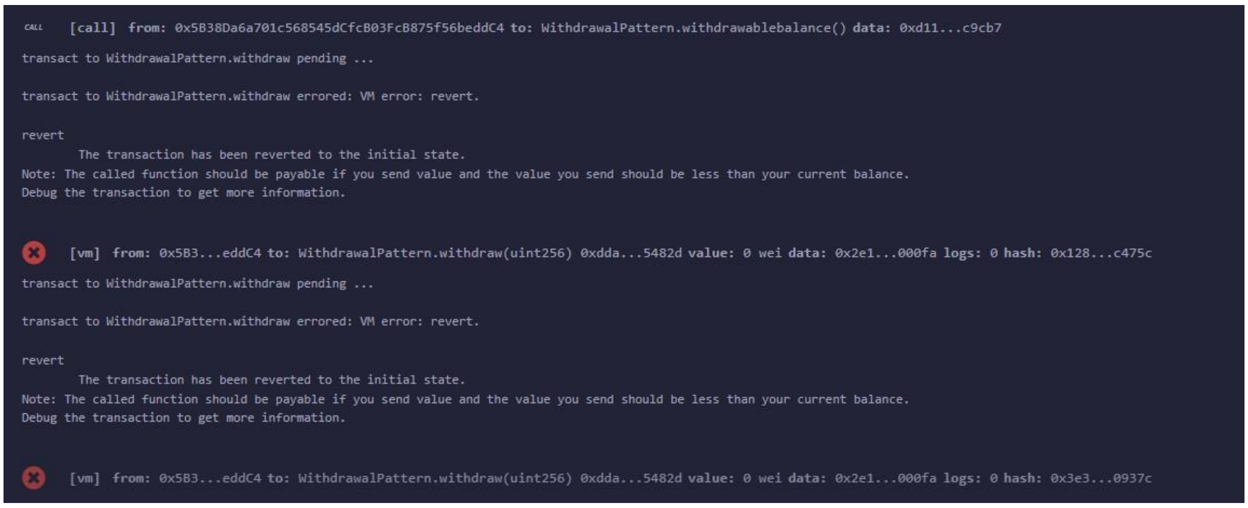
```
0: uint256: 0
```



```
0: uint256: 500
```

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.



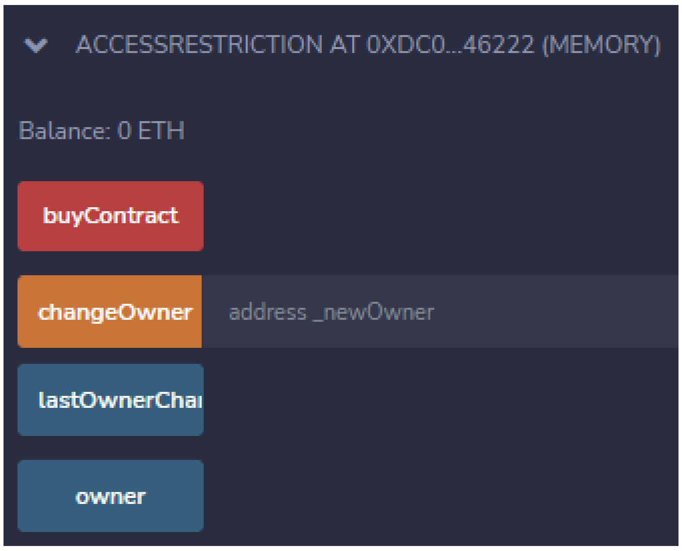


```
CALL  [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: WithdrawalPattern.withdrawablebalance() data: 0xd11...c9cb7
transact to WithdrawalPattern.withdraw pending ...

transact to WithdrawalPattern.withdraw errored: VM error: revert.

revert
        The transaction has been reverted to the initial state.
Note: The called function should be payable if you send value and the value you send should be less than your current balance.
Debug the transaction to get more information.


      [vm] from: 0x5B3...eddC4 to: WithdrawalPattern.withdraw(uint256) 0xdda...5482d value: 0 wei data: 0x2e1...000fa logs: 0 hash: 0x128...c475c
transact to WithdrawalPattern.withdraw pending ...

transact to WithdrawalPattern.withdraw errored: VM error: revert.

revert
        The transaction has been reverted to the initial state.
Note: The called function should be payable if you send value and the value you send should be less than your current balance.
Debug the transaction to get more information.


      [vm] from: 0x5B3...eddC4 to: WithdrawalPattern.withdraw(uint256) 0xdda...5482d value: 0 wei data: 0x2e1...000fa logs: 0 hash: 0x3e3...0937c
```

2) Restricted Access

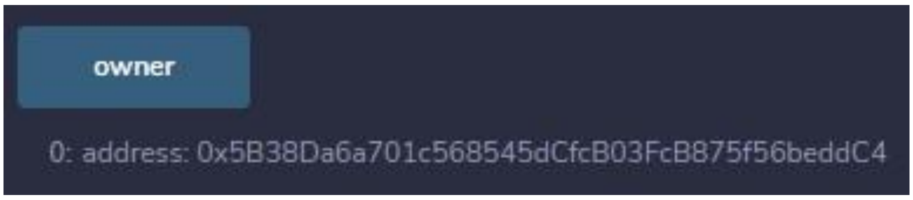
```solidity
//SPDX-License-Identifier: MIT
pragma solidity ^0.8.18;

contract RestrictedAccess {
    address public owner = msg.sender;
    uint256 public creationTime = block.timestamp;

    modifier onlyBy(address _account) {
        require(msg.sender == _account, "Sender not authorized!");
        _;
    }

    modifier onlyAfter(uint256 _time) {
        require(block.timestamp >= _time, "Function was called too early!");
        _;
    }

    modifier costs(uint256 _amount) {
        require(msg.value >= _amount, "Not enough Ether provided!");
        _;
    }

    function forceOwnerChange(address _newOwner)
        public
        payable
        costs(200 ether)
    {
        owner = _newOwner;
    }

    function changeOwner(address _owner) public onlyBy(owner) {
        owner = _owner;
    }

    function disown() public onlyBy(owner) onlyAfter(creationTime + 3 weeks) {
        delete owner;
    }
}
```

Output



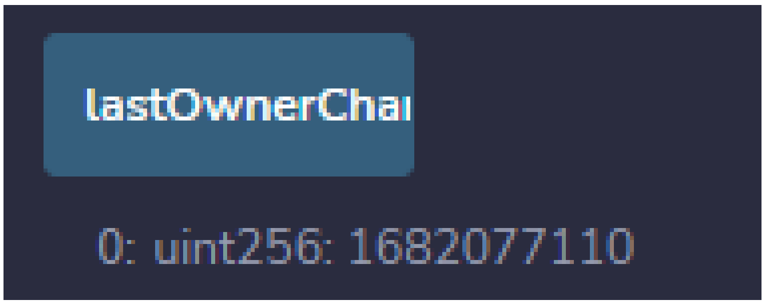
Flow of execution

Step 1-> Click on owner to create an owner object



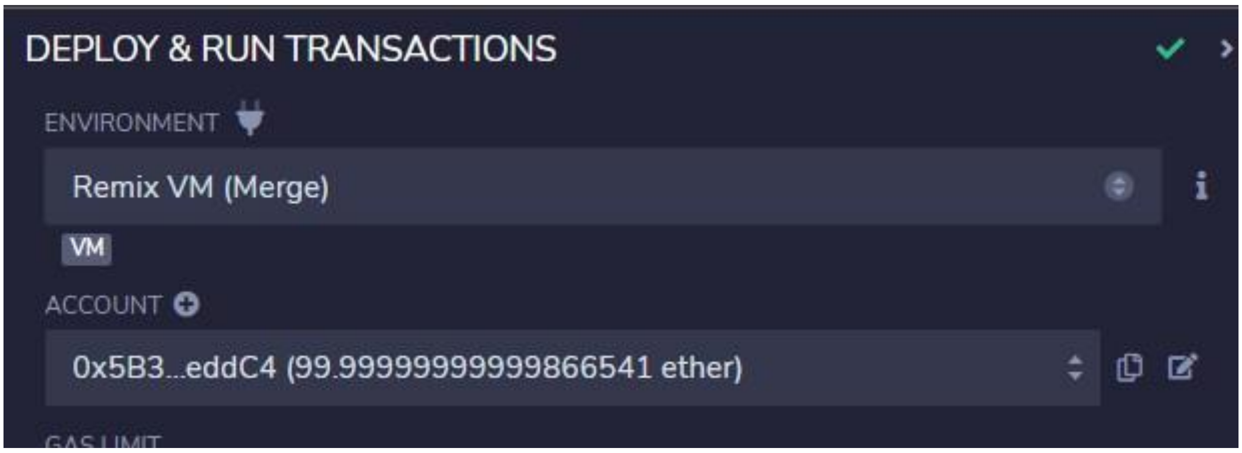
```
0: address: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
```

Step 2-> Click on lastOwnerChange button



```
0: uint256: 1682077110
```

Step 3-> Change the address of the account from Account dropdown in Deploytab of Remix IDE.


DEPLOY & RUN TRANSACTIONS

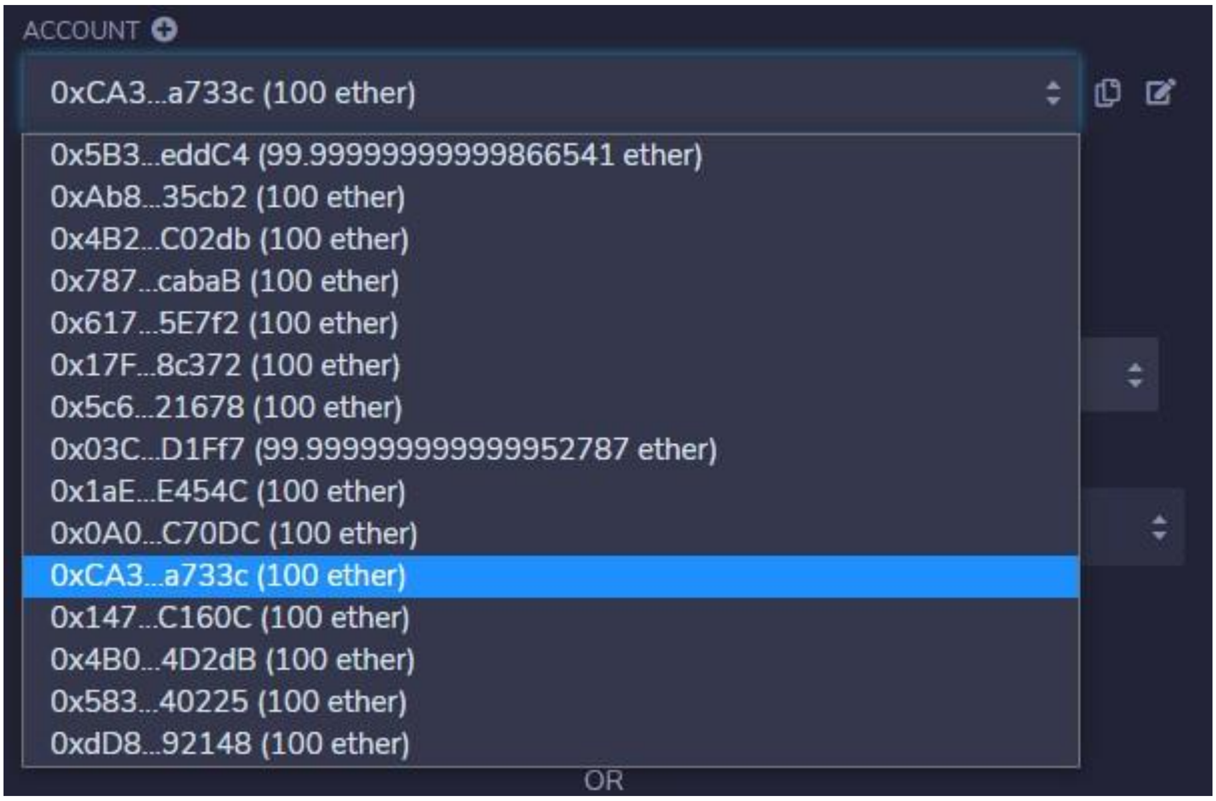
ENVIRONMENT

Remix VM (Merge)

VM

ACCOUNT

0x5B3...eddC4 (99.99999999999866541 ether)

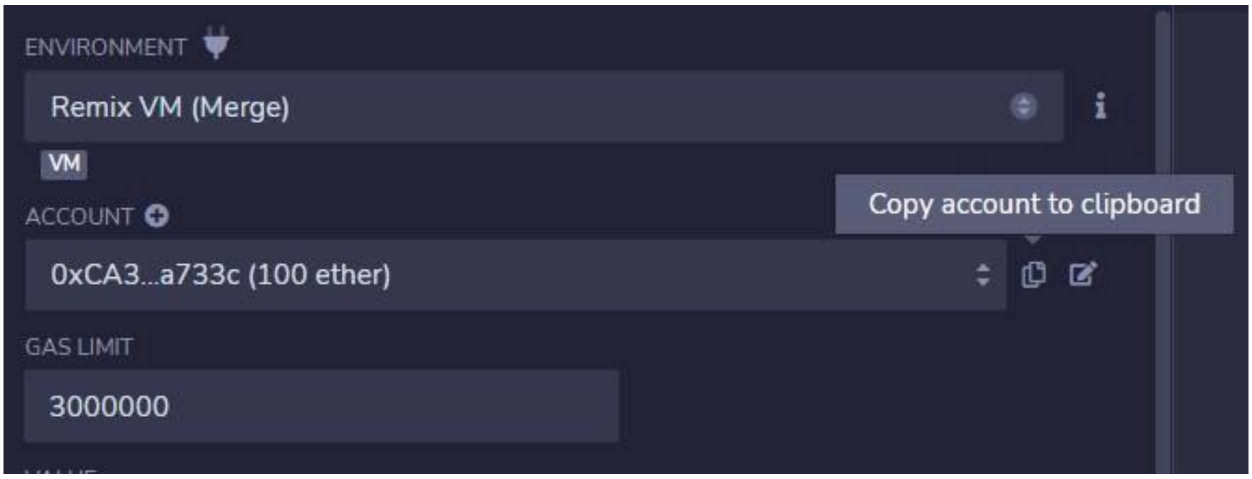

ACCOUNT

0xCA3...a733c (100 ether)

```
0x5B3...eddC4 (99.99999999999866541 ether)
0xAb8...35cb2 (100 ether)
0x4B2...C02db (100 ether)
0x787...cabaB (100 ether)
0x617...5E7f2 (100 ether)
0x17F...8c372 (100 ether)
0x5c6...21678 (100 ether)
0x03C...D1Ff7 (99.9999999999999952787 ether)
0x1aE...E454C (100 ether)
0x0A0...C70DC (100 ether)
0xCA3...a733c (100 ether)
0x147...C160C (100 ether)
0x4B0...4D2dB (100 ether)
0x583...40225 (100 ether)
0xdD8...92148 (100 ether)
```

OR

Step 4-> Copy the address


ENVIRONMENT

Remix VM (Merge)

VM

ACCOUNT

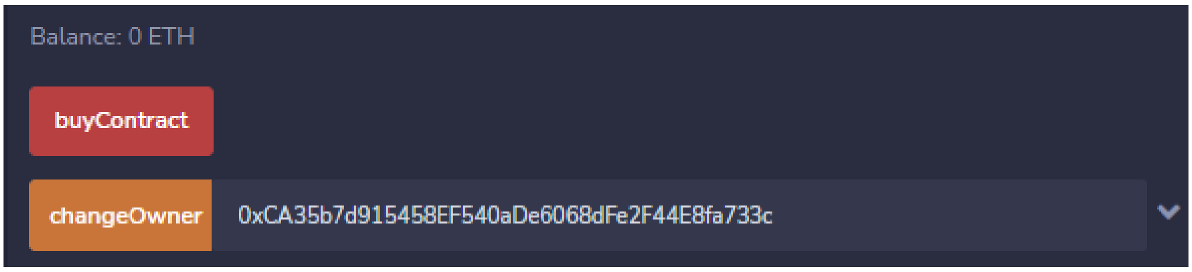
0xCA3...a733c (100 ether)

Copy account to clipboard

GAS LIMIT

3000000

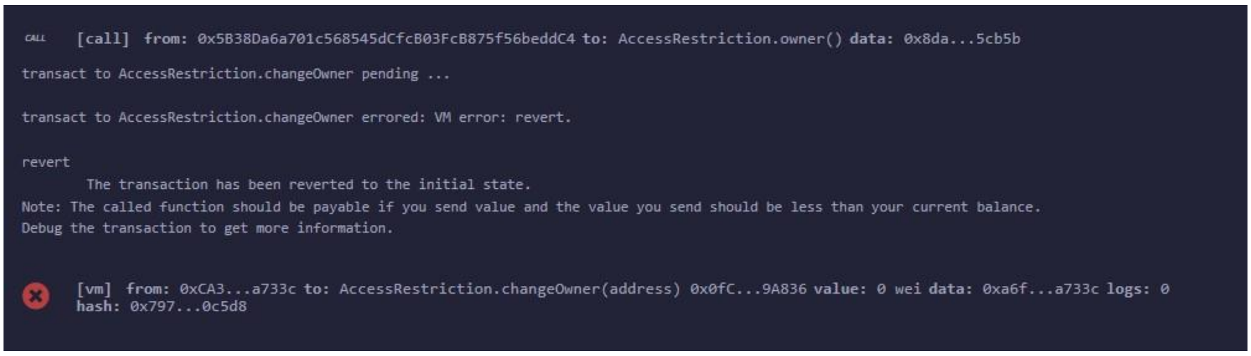
Step 5-> Paste the address in changeOwner input and click on changeOwner.


Balance: 0 ETH

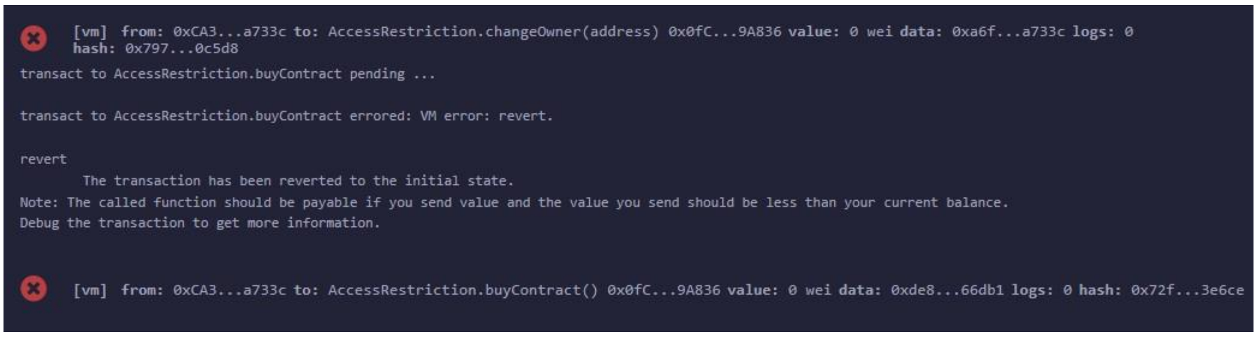
buyContract

changeOwner 0xCA35b7d915458EF540aDe6068dFe2F44E8fa733c

Step 6-> You should get an error as following

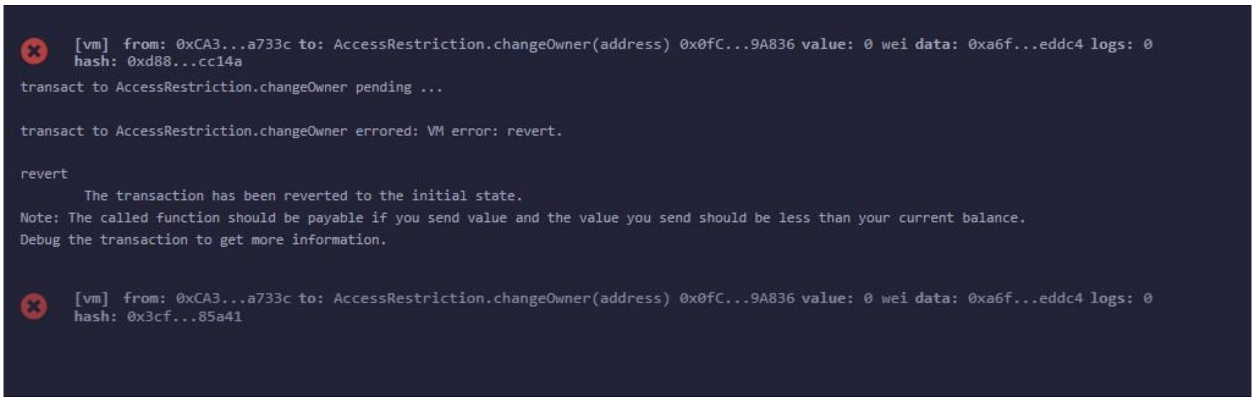

```
CALL  [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: AccessRestriction.owner() data: 0x8da...5cb5b
transact to AccessRestriction.changeOwner pending ...

transact to AccessRestriction.changeOwner errored: VM error: revert.

revert
        The transaction has been reverted to the initial state.
Note: The called function should be payable if you send value and the value you send should be less than your current balance.
Debug the transaction to get more information.

      [vm] from: 0xCA3...a733c to: AccessRestriction.changeOwner(address) 0x0fC...9A836 value: 0 wei data: 0xa6f...a733c logs: 0
      hash: 0x797...0c5d8
```

Step 7-> If you click on buycontract it should give an error as follows



```
[vm] from: 0xCA3...a733c to: AccessRestriction.changeOwner(address) 0x0fC...9A836 value: 0 wei data: 0xa6f...a733c logs: 0
hash: 0x797...0c5d8
transact to AccessRestriction.buyContract pending ...

transact to AccessRestriction.buyContract errored: VM error: revert.

revert
        The transaction has been reverted to the initial state.
Note: The called function should be payable if you send value and the value you send should be less than your current balance.
Debug the transaction to get more information.


[vm] from: 0xCA3...a733c to: AccessRestriction.buyContract() 0x0fC...9A836 value: 0 wei data: 0xde8...66db1 logs: 0 hash: 0x72f...3e6ce
```

Step 8-> Now, paste the actual address of the account in the changeowner inputand click on changeowner



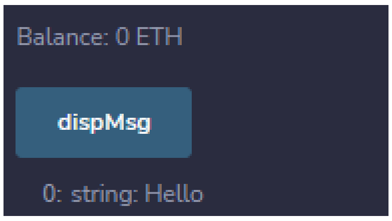
```
[vm] from: 0xCA3...a733c to: AccessRestriction.changeOwner(address) 0x0fC...9A836 value: 0 wei data: 0xa6f...eddc4 logs: 0
hash: 0xd88...cc14a
transact to AccessRestriction.changeOwner pending ...

transact to AccessRestriction.changeOwner errored: VM error: revert.

revert
        The transaction has been reverted to the initial state.
Note: The called function should be payable if you send value and the value you send should be less than your current balance.
Debug the transaction to get more information.


[vm] from: 0xCA3...a733c to: AccessRestriction.changeOwner(address) 0x0fC...9A836 value: 0 wei data: 0xa6f...eddc4 logs: 0
hash: 0x3cf...85a41
```

B) Contracts, Inheritance, Constructors, Abstract Contracts, Interfaces

1) Contracts

```
pragma solidity ^0.5.0;

contract Contract_demo {
    string message = "Hello";

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

Output

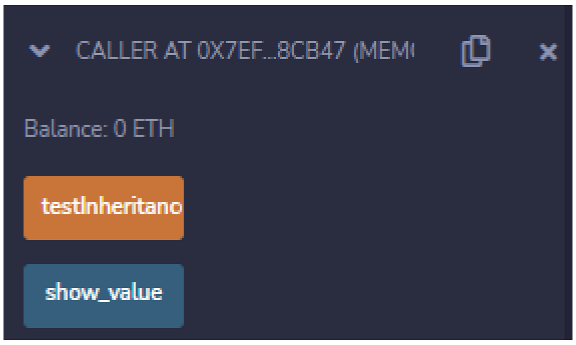


```
Balance: 0 ETH
dispMsg
0: string: Hello
```

2) Inheritance

```
pragma solidity >=0.4.22 <0.6.0;

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();
    }
}
```

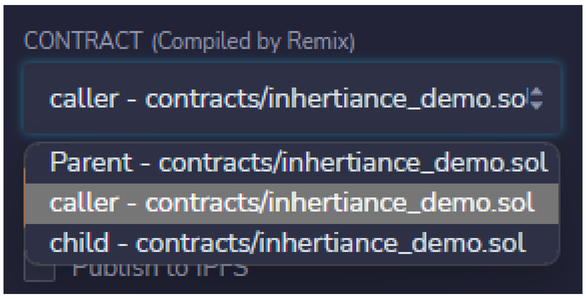
Outputs


CALLER AT 0X7EF...8CB47 (MEM

Balance: 0 ETH

Flow of execution

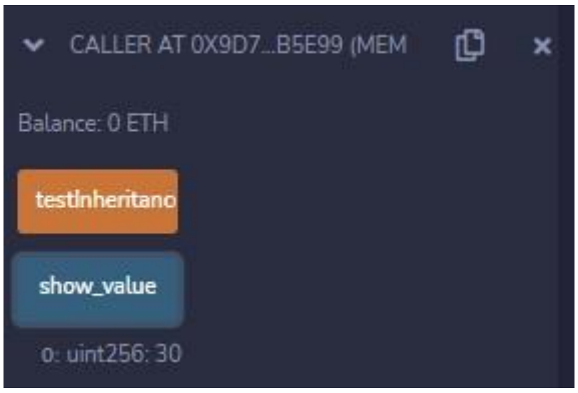
Step 1-> Select caller contract to deploy in Contract and deploy


CONTRACT (Compiled by Remix)

caller - contracts/inhertiance_demo.sol

Parent - contracts/inhertiance_demo.sol
caller - contracts/inhertiance_demo.sol
child - contracts/inhertiance_demo.sol

Step 2-> Click test Inheritance and then click on show_value to view value


CALLER AT 0X9D7...B5E99 (MEM

Balance: 0 ETH

```
0: uint256: 30
```

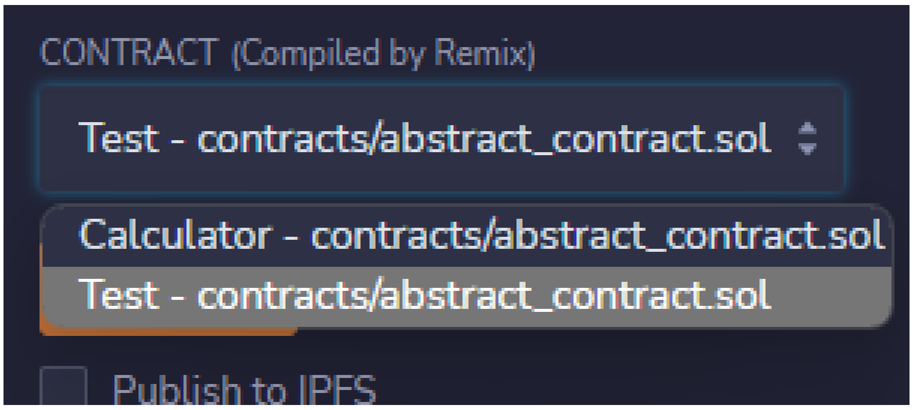
3) Abstract Contracts

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.17;

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;
    }
}
```

Outputs

Flow of execution

Step 1-> Select Test contract and deploy



CONTRACT (Compiled by Remix)

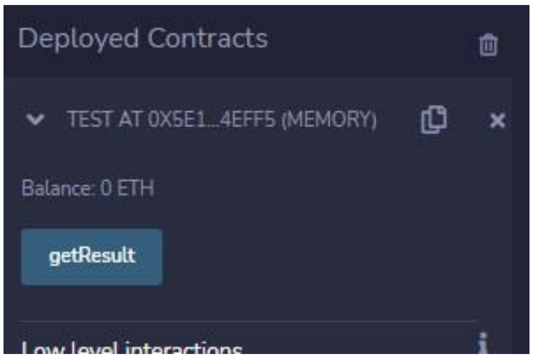
Test - contracts/abstract_contract.sol

Calculator - contracts/abstract_contract.sol

Test - contracts/abstract_contract.sol

Publish to IPFS

Step 2-> The contact will deploy as below



Deployed Contracts

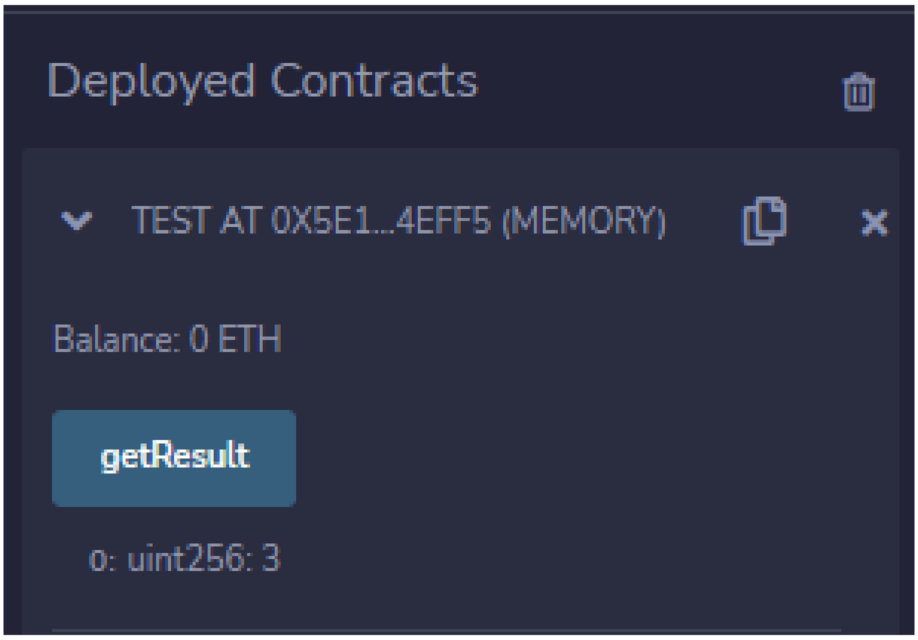
TEST AT 0X5E1...4EFF5 (MEMORY)

Balance: 0 ETH

getResult

Low level interactions

Step 3-> Click on getResult to get sum of a+b


Deployed Contracts

TEST AT 0X5E1...4EFF5 (MEMORY)

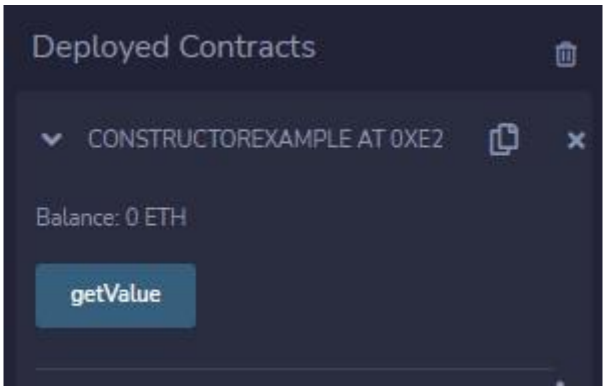
Balance: 0 ETH

0: uint256: 3

4) Constructors

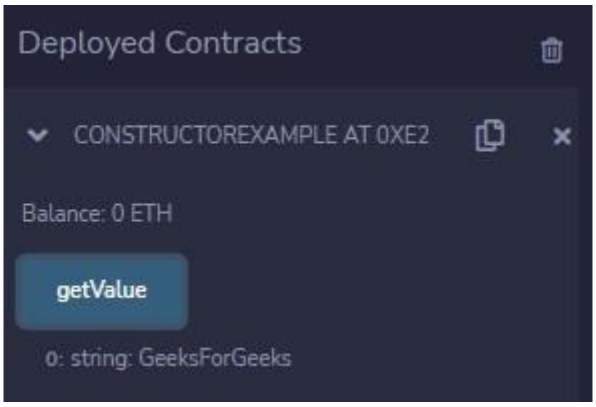
```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

// Creating a contract
contract constructorExample {
    string str;

    constructor() public {
        str = "GeeksForGeeks";
    }

    function getValue() public view returns (string memory) {
        return str;
    }
}
```

Outputs



Flow of execution

Step 1-> Click on getValue to print string



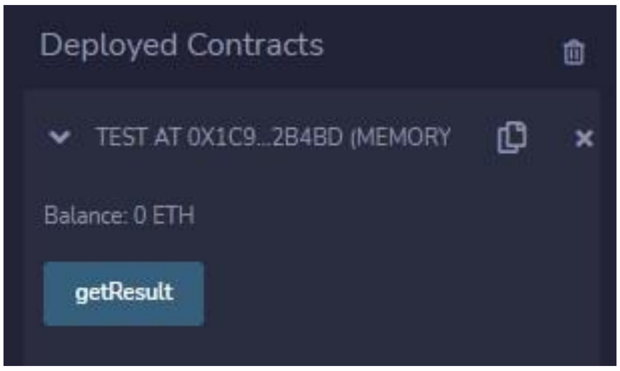
```
0: string: GeeksForGeeks
```

5) Interfaces

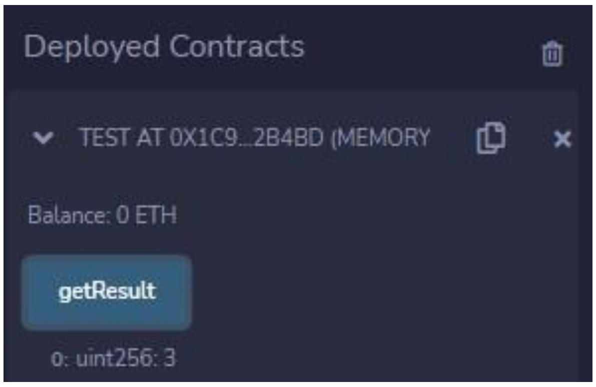
```
pragma solidity ^0.5.0;

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;
   }
}
```

Outputs

Flow of execution



Step 1-> Click on getResult to display sum



```
0: uint256: 3
```

C) Libraries, Assembly, Events, Error handling.

1) Libraries

myLib.sol Code

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.7.0 <0.9.0;

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;
    }
}
```

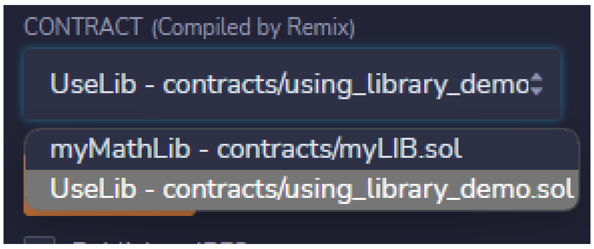
using_library.sol Code

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.7.0 <0.9.0;

import "contracts/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);
    }
}
```

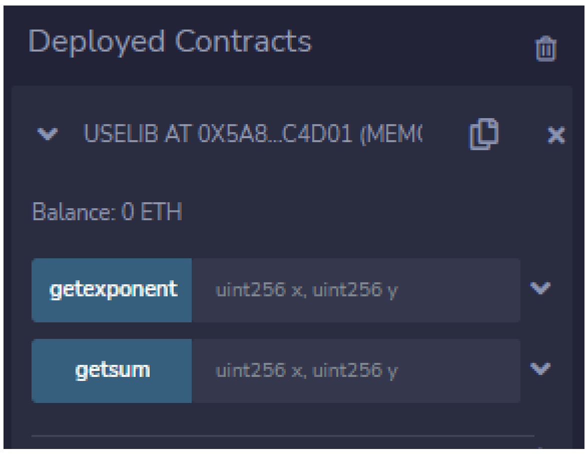
Outputs

Flow of execution

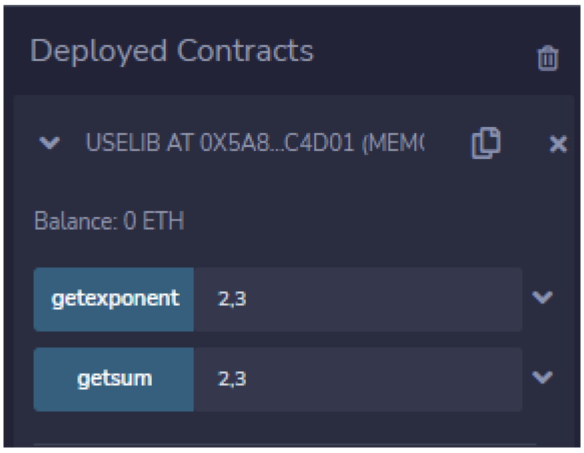
Step 1-> Change contract to UseLib and deploy.


CONTRACT (Compiled by Remix)
UseLib - contracts/using_library_demo
myMathLib - contracts/myLIB.sol
UseLib - contracts/using_library_demo.sol

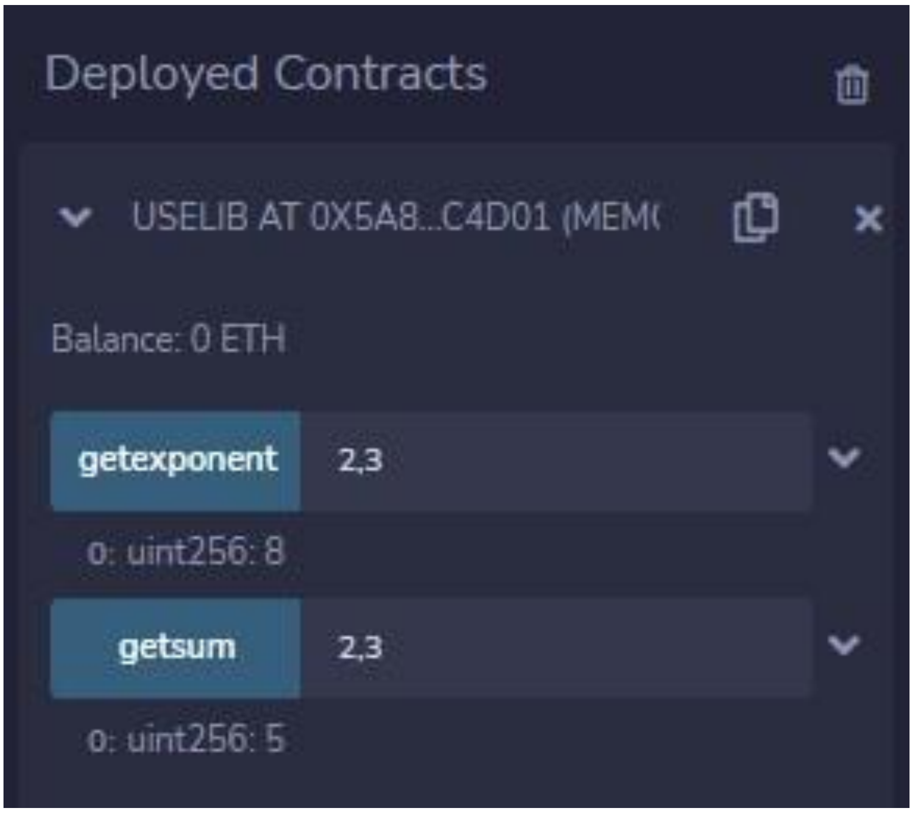
Step 2-> The deployed contract should be same as below


Deployed Contracts
USELIB AT 0X5A8...C4D01 (MEM(
Balance: 0 ETH
getexponent uint256 x, uint256 y
getsum uint256 x, uint256 y

Step 3-> Input values to both getexponent and getsum functions as below


Deployed Contracts
USELIB AT 0X5A8...C4D01 (MEM(
Balance: 0 ETH
getexponent 2,3
getsum 2,3

Step 4-> Execute both functions. You will get below output



Balance: 0 ETH

getexponent 2,3

0: uint256: 8

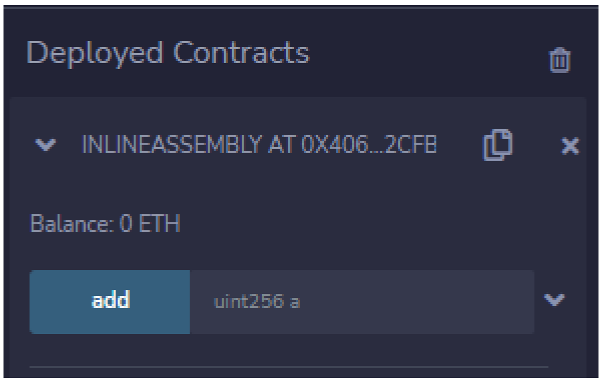
getsum 2,3

0: uint256: 5

2) Assembly

```
// SPDX-License-Identifier: GPL-3.0
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)
        }
    }
}
```

Outputs

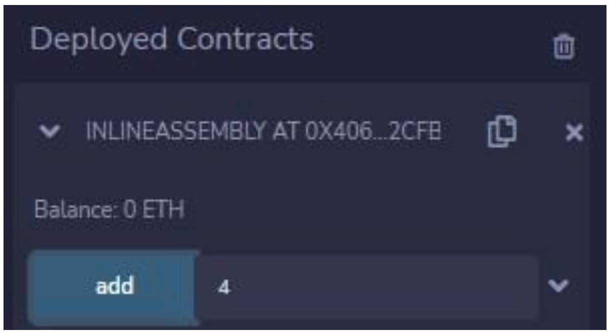

Deployed Contracts

INLINEASSEMBLY AT 0X406...2CFB

Balance: 0 ETH

Flow of execution

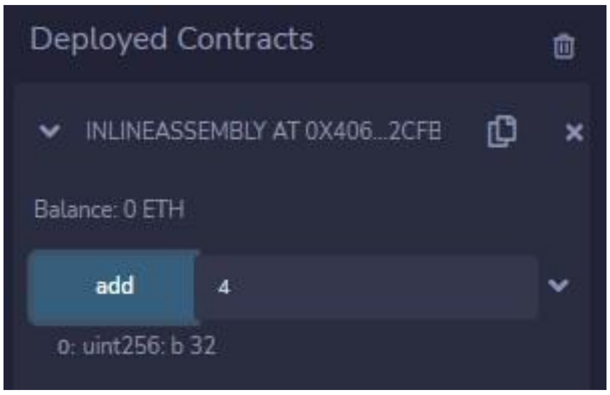
Step 1-> Input a number for add function


Deployed Contracts

INLINEASSEMBLY AT 0X406...2CFB

Balance: 0 ETH

Step 2-> Click add to output sum



Balance: 0 ETH

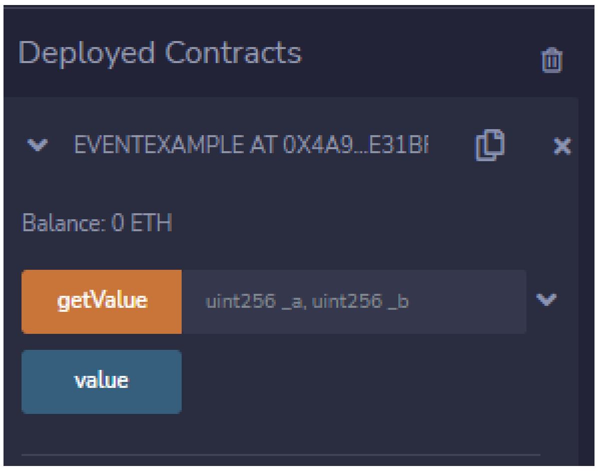
4

```
0: uint256: b 32
```

3) Events

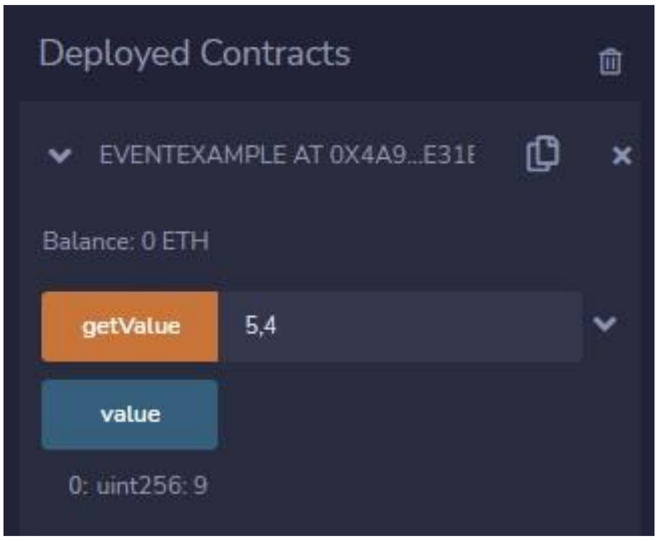
```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;

// 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;
    }
}
```

Outputs



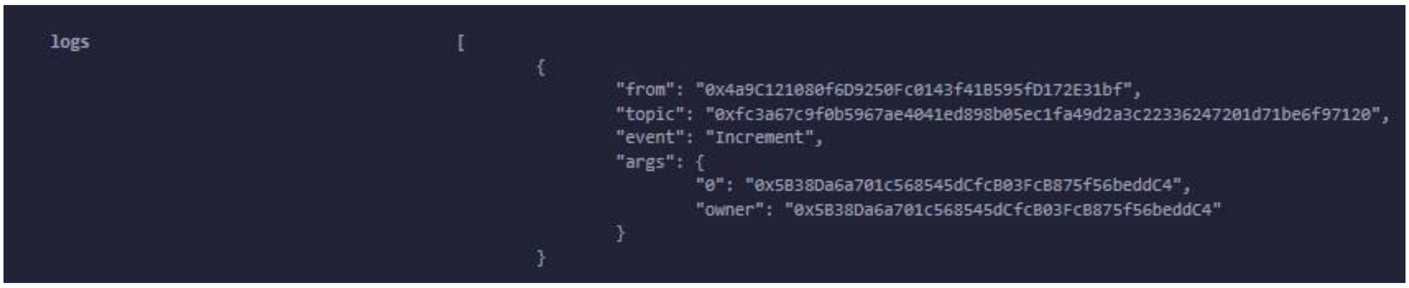
Flow of execution

Step 1-> Provide values to getValue function and click on it.



```
0: uint256: 9
```

Step 2-> In the terminal check for logs



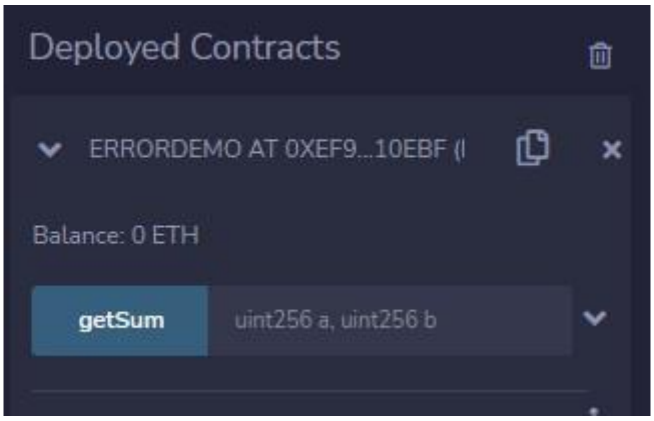
```
logs                                    [
                                                {
                                                        "from": "0x4a9C121080f6D9250Fc0143f41B595fD172E31bf",
                                                        "topic": "0xfc3a67c9f0b5967ae4041ed898b05ec1fa49d2a3c22336247201d71be6f97120",
                                                        "event": "Increment",
                                                        "args": {
                                                                "0": "0x5B38Da6a701c568545dCfcB03FcB875f56beddC4",
                                                                "owner": "0x5B38Da6a701c568545dCfcB03FcB875f56beddC4"
                                                        }
                                                }
```

4) Error Handling

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.17;
```

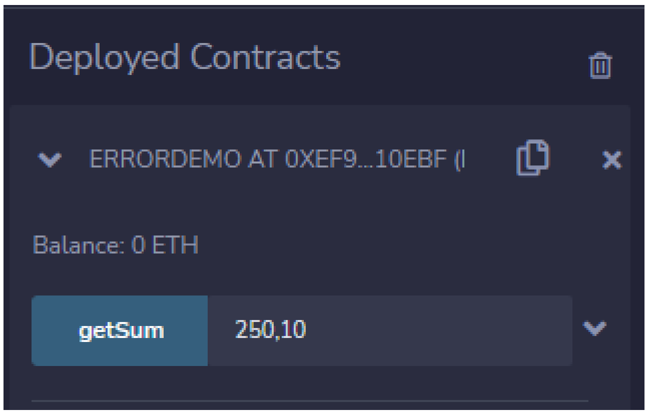
```
contract ErrorDemo {
    function getSum(uint256 a, uint256 b) public pure returns (uint256) {
        uint256 sum = a + b;
        // require(sum < 255, "Invalid");
        assert(sum<255);
        return sum;
    }
}
```

Output

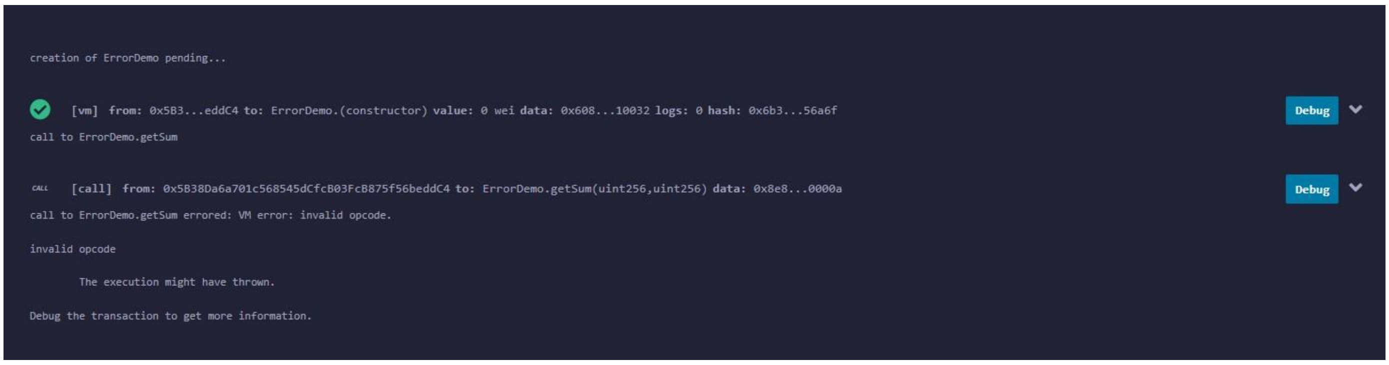


Flow of execution

Step 1-> Provide some values and press on getSum



Step 2-> Check terminal panel



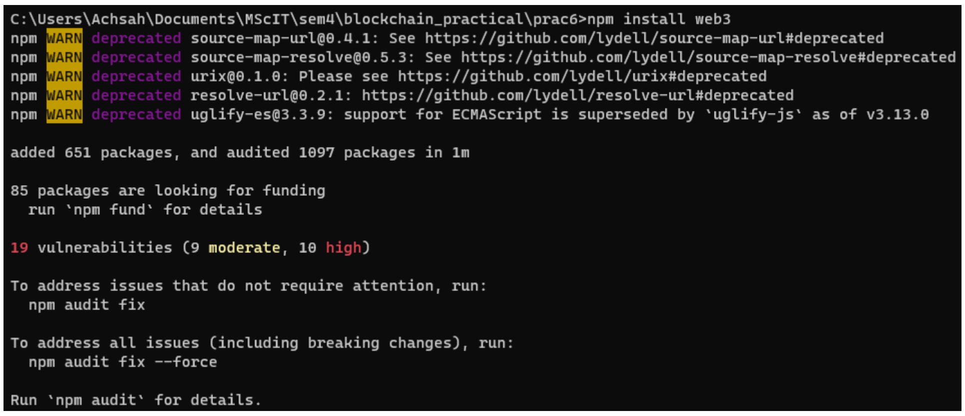
```
creation of ErrorDemo pending...

[vm] from: 0x5B3...eddC4 to: ErrorDemo.(constructor) value: 0 wei data: 0x608...10032 logs: 0 hash: 0x6b3...56a6f
call to ErrorDemo.getSum

CALL [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: ErrorDemo.getSum(uint256,uint256) data: 0x8e8...0000a
call to ErrorDemo.getSum errored: VM error: invalid opcode.

invalid opcode

        The execution might have thrown.

Debug the transaction to get more information.
```

PRACTICAL-5

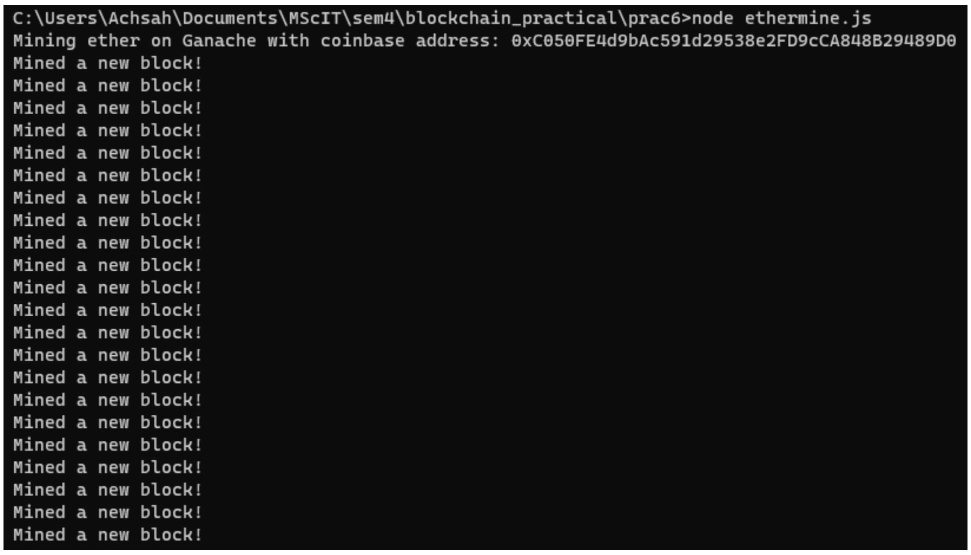
Aim: WRITE A PROGRAM TO DEMONSTRATE MINING OF ETHER

```
const Web3 = require('web3');

const web3 = new Web3(new
Web3.providers.HttpProvider('http:   127.0.0.1:7545'));   Replace with yourGanache HTTP provider


async function mine() {
        const accounts = await web3.eth.getAccounts();const coinbaseacc1 =
        accounts[0];
        const coinbaseacc2 = accounts[1];
        console.log(`Mining ether on Ganache with coinbase address:
${coinbaseacc1}`);

        while (true) {try {
                        await web3.eth.sendTransaction({from: coinbaseacc1,
                                to: coinbaseacc2,value:
                                50,
                        });
                        console.log(`Mined a new block!`);
                } catch (err) { console.error(err);
                }
        }
}

mine();
```

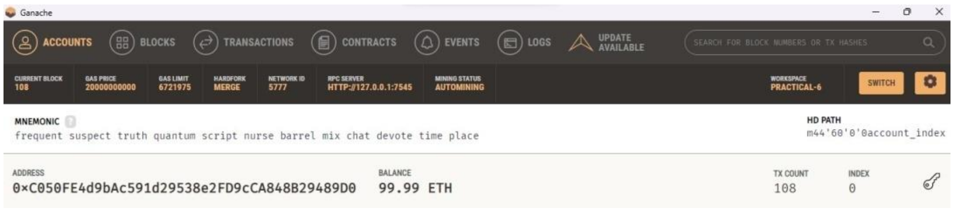
Output



```
C:\Users\Achsah\Documents\MScIT\sem4\blockchain_practical\prac6>npm install web3
npm WARN deprecated source-map-url@0.4.1: See https://github.com/lydell/source-map-url#deprecated
npm WARN deprecated source-map-resolve@0.5.3: See https://github.com/lydell/source-map-resolve#deprecated
npm WARN deprecated urix@0.1.0: Please see https://github.com/lydell/urix#deprecated
npm WARN deprecated resolve-url@0.2.1: https://github.com/lydell/resolve-url#deprecated
npm WARN deprecated uglify-es@3.3.9: support for ECMAScript is superseded by `uglify-js` as of v3.13.0

added 651 packages, and audited 1097 packages in 1m

85 packages are looking for funding
  run `npm fund` for details

19 vulnerabilities (9 moderate, 10 high)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
```



```
C:\Users\Achsah\Documents\MScIT\sem4\blockchain_practical\prac6>node ethermine.js
Mining ether on Ganache with coinbase address: 0xC050FE4d9bAc591d29538e2FD9cCA848B29489D0
Mined a new block!
Mined a new block!
Mined a new block!
Mined a new block!
Mined a new block!
Mined a new block!
Mined a new block!
Mined a new block!
Mined a new block!
Mined a new block!
Mined a new block!
Mined a new block!
Mined a new block!
Mined a new block!
Mined a new block!
Mined a new block!
Mined a new block!
Mined a new block!
Mined a new block!
Mined a new block!
Mined a new block!
Mined a new block!
```



CURRENT BLOCK 108 | GAS PRICE 20000000000 | GAS LIMIT 6721975 | HARDFORK MERGE | NETWORK ID 5777 | RPC SERVER HTTP://127.0.0.1:7545 | MINING STATUS AUTOMINING | WORKSPACE PRACTICAL-6

MNEMONIC
frequent suspect truth quantum script nurse barrel mix chat devote time place

HD PATH
m44'60'0'0account_index

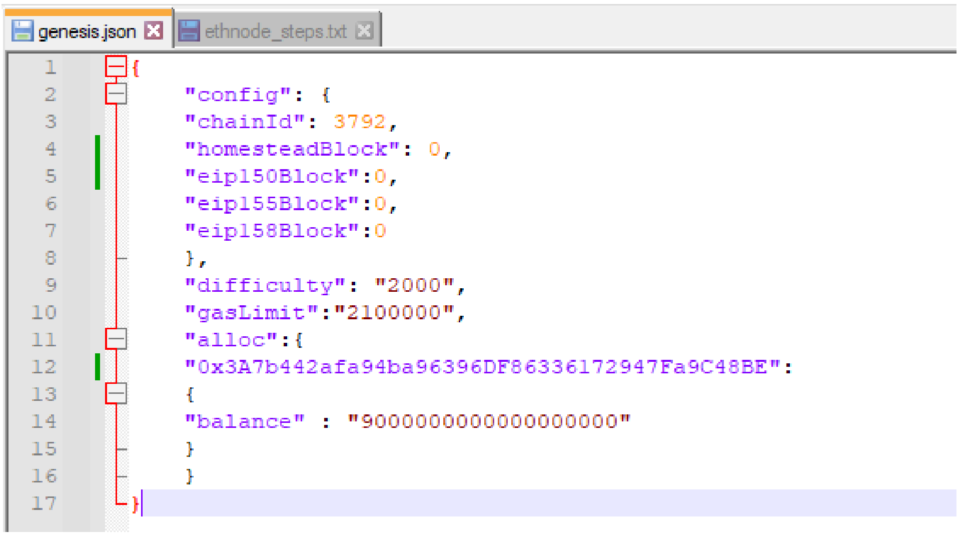
ADDRESS	BALANCE	TX COUNT	INDEX
0×C050FE4d9bAc591d29538e2FD9cCA848B29489D0	99.99 ETH	108	0

PRACTICAL-6

Aim: DEMONSTRATE THE RUNNING OF THE BLOCKCHAIN NODE

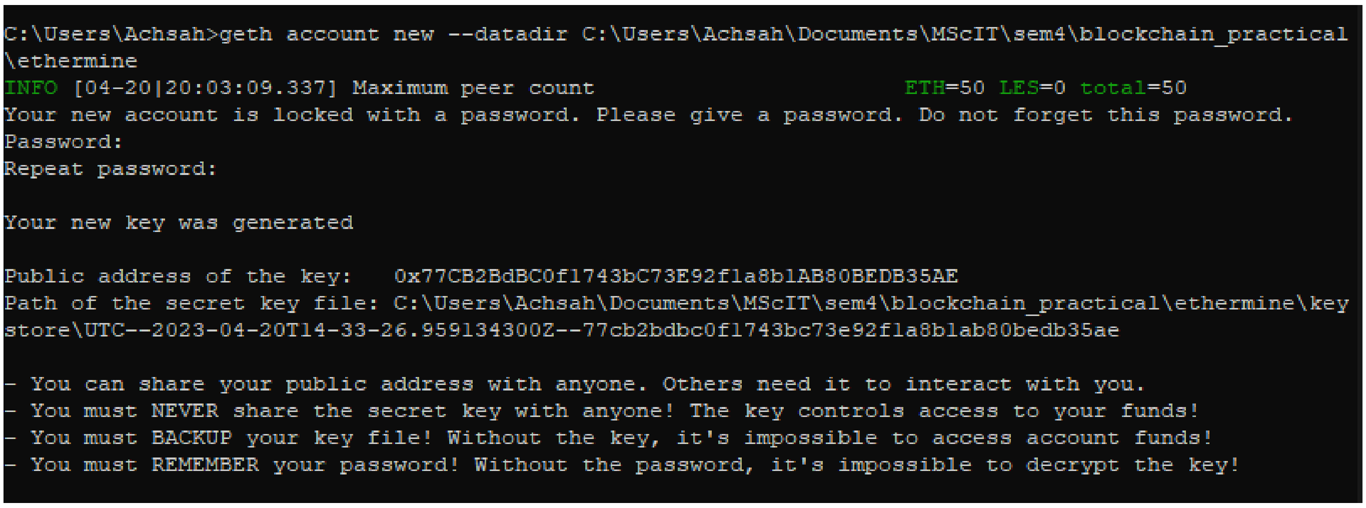
Step 1-> Create a folder named ethermine and a JSON file named genesis.jsonand write the following lines in it.

```
{
        "config": {
                "chainId": 3792,
                "homesteadBlock": 0,
                "eip150Block": 0,
                "eip155Block": 0,
                "eip158Block": 0
        },
        "difficulty": "2000",
        "gasLimit": "2100000","alloc": {
                "0×0b6C4c81f58B8d692A7B46AD1e16a1147c25299F": {"balance":
                        "9000000000000000000"
                }
        }
}
```



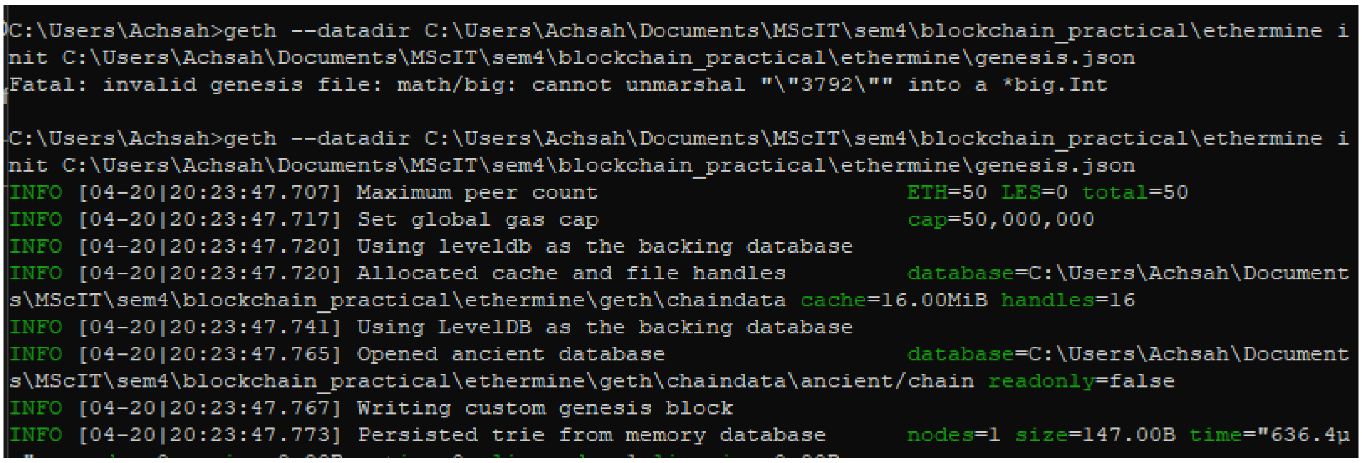
```
{
    "config": {
    "chainId": 3792,
    "homesteadBlock": 0,
    "eip150Block":0,
    "eip155Block":0,
    "eip158Block":0
    },
    "difficulty": "2000",
    "gasLimit":"2100000",
    "alloc":{
    "0x3A7b442afa94ba96396DF86336172947Fa9C48BE":
    {
    "balance" : "9000000000000000000"
    }
    }
}
```

Step 2-> Run command **geth account new –datadir C:\Users\Achsah\Documents\MScIT\sem4\blockchain_practical\etherminetestnet-blockchain**



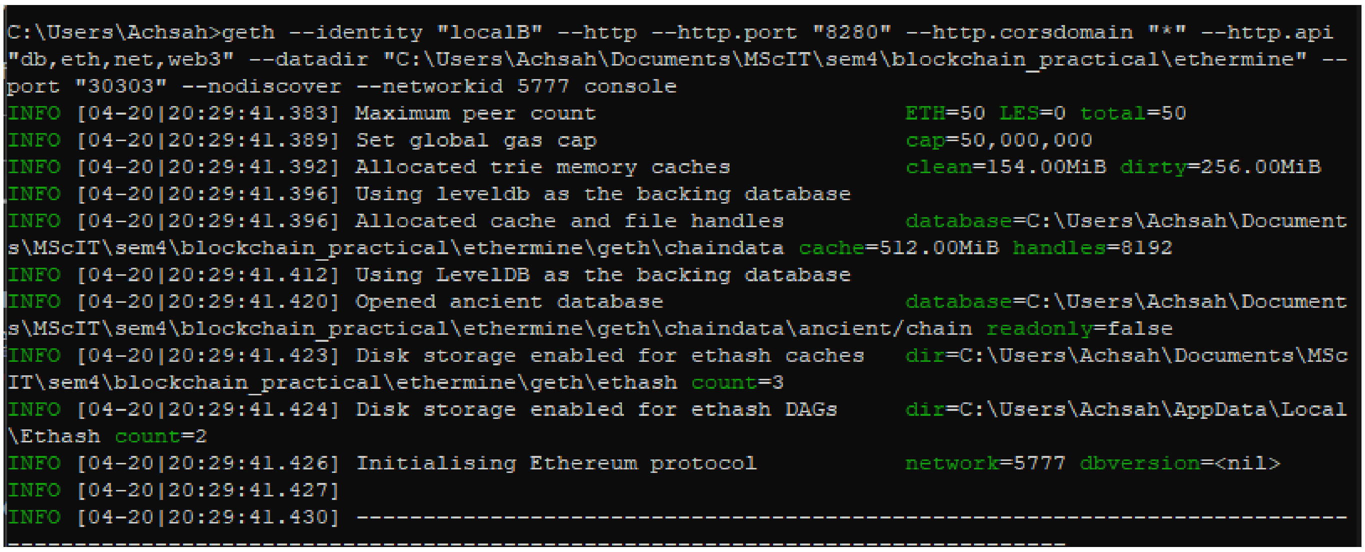
```
C:\Users\Achsah>geth account new --datadir C:\Users\Achsah\Documents\MScIT\sem4\blockchain_practical
\ethermine
INFO [04-20|20:03:09.337] Maximum peer count                       ETH=50 LES=0 total=50
Your new account is locked with a password. Please give a password. Do not forget this password.
Password:
Repeat password:

Your new key was generated

Public address of the key:   0x77CB2BdBC0f1743bC73E92f1a8b1AB80BEDB35AE
Path of the secret key file: C:\Users\Achsah\Documents\MScIT\sem4\blockchain_practical\ethermine\key
store\UTC--2023-04-20T14-33-26.959134300Z--77cb2bdbc0f1743bc73e92f1a8b1ab80bedb35ae

- You can share your public address with anyone. Others need it to interact with you.
- You must NEVER share the secret key with anyone! The key controls access to your funds!
- You must BACKUP your key file! Without the key, it's impossible to access account funds!
- You must REMEMBER your password! Without the password, it's impossible to decrypt the key!
```

Step 3-> Run command **geth account new --datadir C:\Users\Achsah\Documents\MScIT\sem4\blockchain_practical\ethermine**



```
C:\Users\Achsah>geth --datadir C:\Users\Achsah\Documents\MScIT\sem4\blockchain_practical\ethermine i
nit C:\Users\Achsah\Documents\MScIT\sem4\blockchain_practical\ethermine\genesis.json
Fatal: invalid genesis file: math/big: cannot unmarshal "\"3792\"" into a *big.Int

C:\Users\Achsah>geth --datadir C:\Users\Achsah\Documents\MScIT\sem4\blockchain_practical\ethermine i
nit C:\Users\Achsah\Documents\MScIT\sem4\blockchain_practical\ethermine\genesis.json
INFO [04-20|20:23:47.707] Maximum peer count                       ETH=50 LES=0 total=50
INFO [04-20|20:23:47.717] Set global gas cap                       cap=50,000,000
INFO [04-20|20:23:47.720] Using leveldb as the backing database
INFO [04-20|20:23:47.720] Allocated cache and file handles         database=C:\Users\Achsah\Document
s\MScIT\sem4\blockchain_practical\ethermine\geth\chaindata cache=16.00MiB handles=16
INFO [04-20|20:23:47.741] Using LevelDB as the backing database
INFO [04-20|20:23:47.765] Opened ancient database                  database=C:\Users\Achsah\Document
s\MScIT\sem4\blockchain_practical\ethermine\geth\chaindata\ancient/chain readonly=false
INFO [04-20|20:23:47.767] Writing custom genesis block
INFO [04-20|20:23:47.773] Persisted trie from memory database      nodes=1 size=147.00B time="636.4µ
```

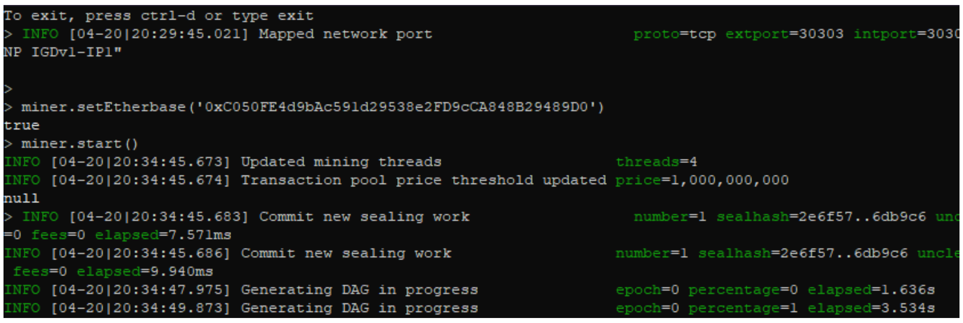
Step 4-> Run command **geth --identity "localB" --http --http.port "8280" --http.corsdomain "*" --http.api "db,eth,net,web3" --datadir "C:\Users\Achsah\Documents\MScIT\sem4\blockchain_practical\ethermine" --port "30303" --nodiscover --networkid 5777 console**. This command willenable geth console.



```
C:\Users\Achsah>geth --identity "localB" --http --http.port "8280" --http.corsdomain "*" --http.api
"db,eth,net,web3" --datadir "C:\Users\Achsah\Documents\MScIT\sem4\blockchain_practical\ethermine" --
port "30303" --nodiscover --networkid 5777 console
INFO [04-20|20:29:41.383] Maximum peer count                       ETH=50 LES=0 total=50
INFO [04-20|20:29:41.389] Set global gas cap                       cap=50,000,000
INFO [04-20|20:29:41.392] Allocated trie memory caches             clean=154.00MiB dirty=256.00MiB
INFO [04-20|20:29:41.396] Using leveldb as the backing database
INFO [04-20|20:29:41.396] Allocated cache and file handles         database=C:\Users\Achsah\Document
s\MScIT\sem4\blockchain_practical\ethermine\geth\chaindata cache=512.00MiB handles=8192
INFO [04-20|20:29:41.412] Using LevelDB as the backing database
INFO [04-20|20:29:41.420] Opened ancient database                  database=C:\Users\Achsah\Document
s\MScIT\sem4\blockchain_practical\ethermine\geth\chaindata\ancient/chain readonly=false
INFO [04-20|20:29:41.423] Disk storage enabled for ethash caches   dir=C:\Users\Achsah\Documents\MSc
IT\sem4\blockchain_practical\ethermine\geth\ethash count=3
INFO [04-20|20:29:41.424] Disk storage enabled for ethash DAGs     dir=C:\Users\Achsah\AppData\Local
\Ethash count=2
INFO [04-20|20:29:41.426] Initialising Ethereum protocol           network=5777 dbversion=<nil>
INFO [04-20|20:29:41.427]
INFO [04-20|20:29:41.430] ------------------------------------------------------------------------
------------------------------------------------------------------------------
```

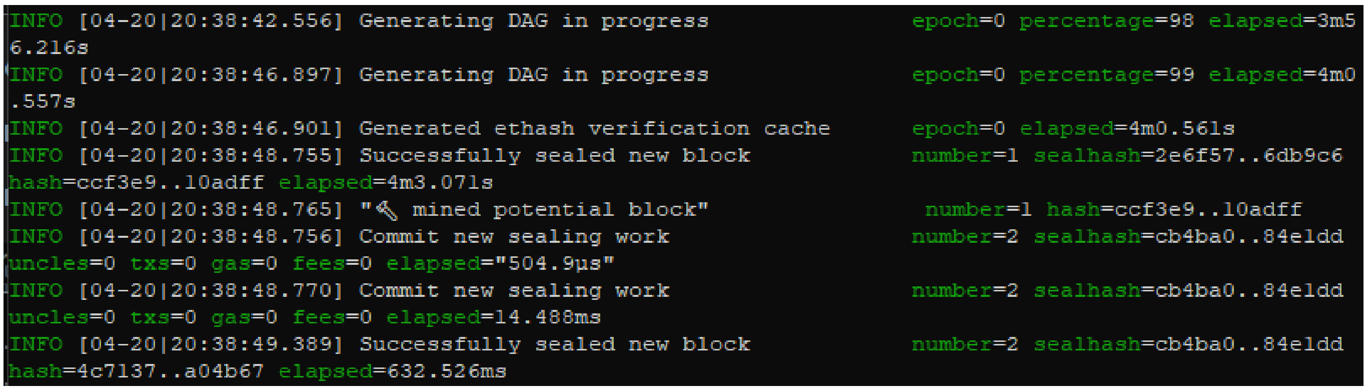
Step 5-> Run the command **miner.setEtherbase('0xC050FE4d9bAc591d29538e2FD9cCA848B29489D0')**in the geth console

Step 6-> Run the command **miner.start()** to start mining



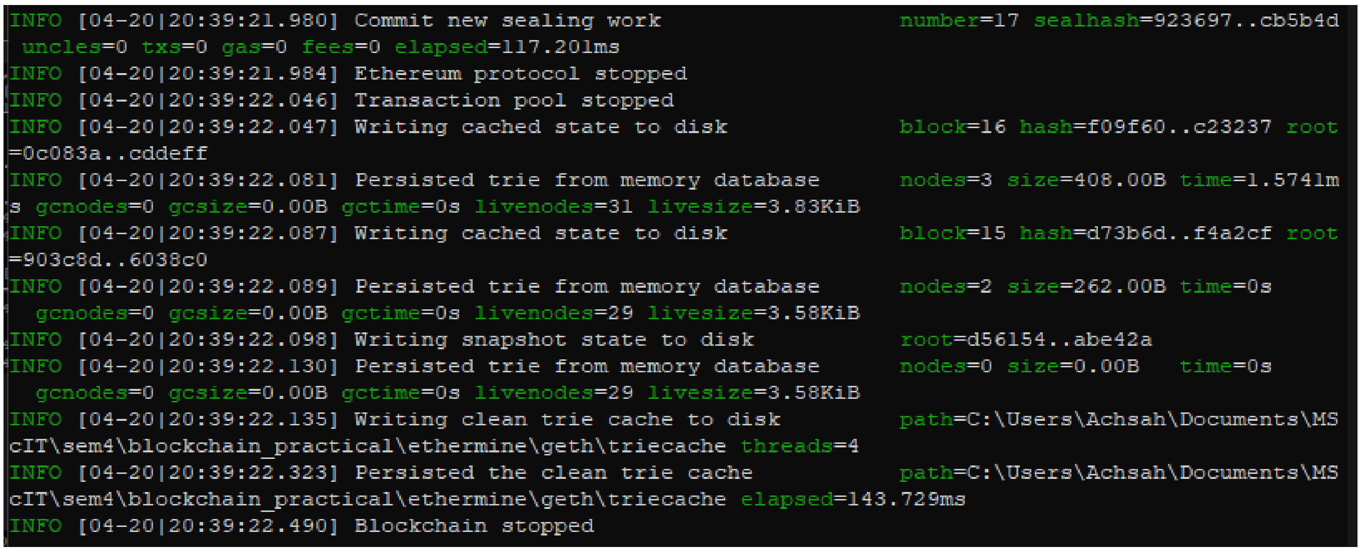
```
To exit, press ctrl-d or type exit
> INFO [04-20|20:29:45.021] Mapped network port                     proto=tcp extport=30303 intport=3030
NP IGDv1-IP1"

>
> miner.setEtherbase('0xC050FE4d9bAc591d29538e2FD9cCA848B29489D0')
true
> miner.start()
INFO [04-20|20:34:45.673] Updated mining threads                  threads=4
INFO [04-20|20:34:45.674] Transaction pool price threshold updated price=1,000,000,000
null
> INFO [04-20|20:34:45.683] Commit new sealing work                 number=1 sealhash=2e6f57..6db9c6 unc
=0 fees=0 elapsed=7.571ms
INFO [04-20|20:34:45.686] Commit new sealing work                 number=1 sealhash=2e6f57..6db9c6 uncle
 fees=0 elapsed=9.940ms
INFO [04-20|20:34:47.975] Generating DAG in progress              epoch=0 percentage=0 elapsed=1.636s
INFO [04-20|20:34:49.873] Generating DAG in progress              epoch=0 percentage=1 elapsed=3.534s
```

Step 7-> Below screenshots are the mining processes running on your local machine.



```
INFO [04-20|20:38:42.556] Generating DAG in progress               epoch=0 percentage=98 elapsed=3m5
6.216s
INFO [04-20|20:38:46.897] Generating DAG in progress               epoch=0 percentage=99 elapsed=4m0
.557s
INFO [04-20|20:38:46.901] Generated ethash verification cache      epoch=0 elapsed=4m0.561s
INFO [04-20|20:38:48.755] Successfully sealed new block            number=1 sealhash=2e6f57..6db9c6
hash=ccf3e9..10adff elapsed=4m3.071s
INFO [04-20|20:38:48.765] "🔨 mined potential block"                 number=1 hash=ccf3e9..10adff
INFO [04-20|20:38:48.756] Commit new sealing work                  number=2 sealhash=cb4ba0..84e1dd
uncles=0 txs=0 gas=0 fees=0 elapsed="504.9µs"
INFO [04-20|20:38:48.770] Commit new sealing work                  number=2 sealhash=cb4ba0..84e1dd
uncles=0 txs=0 gas=0 fees=0 elapsed=14.488ms
INFO [04-20|20:38:49.389] Successfully sealed new block            number=2 sealhash=cb4ba0..84e1dd
hash=4c7137..a04b67 elapsed=632.526ms
```

Step 8-> To stop the mining press **Ctrl+D**



```
INFO [04-20|20:39:21.980] Commit new sealing work                  number=17 sealhash=923697..cb5b4d
 uncles=0 txs=0 gas=0 fees=0 elapsed=117.201ms
INFO [04-20|20:39:21.984] Ethereum protocol stopped
INFO [04-20|20:39:22.046] Transaction pool stopped
INFO [04-20|20:39:22.047] Writing cached state to disk             block=16 hash=f09f60..c23237 root
=0c083a..cddeff
INFO [04-20|20:39:22.081] Persisted trie from memory database      nodes=3 size=408.00B time=1.5741m
s gcnodes=0 gcsize=0.00B gctime=0s livenodes=31 livesize=3.83KiB
INFO [04-20|20:39:22.087] Writing cached state to disk             block=15 hash=d73b6d..f4a2cf root
=903c8d..6038c0
INFO [04-20|20:39:22.089] Persisted trie from memory database      nodes=2 size=262.00B time=0s
  gcnodes=0 gcsize=0.00B gctime=0s livenodes=29 livesize=3.58KiB
INFO [04-20|20:39:22.098] Writing snapshot state to disk           root=d56154..abe42a
INFO [04-20|20:39:22.130] Persisted trie from memory database      nodes=0 size=0.00B   time=0s
  gcnodes=0 gcsize=0.00B gctime=0s livenodes=29 livesize=3.58KiB
INFO [04-20|20:39:22.135] Writing clean trie cache to disk         path=C:\Users\Achsah\Documents\MS
cIT\sem4\blockchain_practical\ethermine\geth\triecache threads=4
INFO [04-20|20:39:22.323] Persisted the clean trie cache           path=C:\Users\Achsah\Documents\MS
cIT\sem4\blockchain_practical\ethermine\geth\triecache elapsed=143.729ms
INFO [04-20|20:39:22.490] Blockchain stopped
```

PRACTICAL-7

Aim:CREATE YOUR OWN BLOCKCHAIN AND DEMONSTRATE ITS USE

Create a javascript folder with the following code in any folder of your choice.

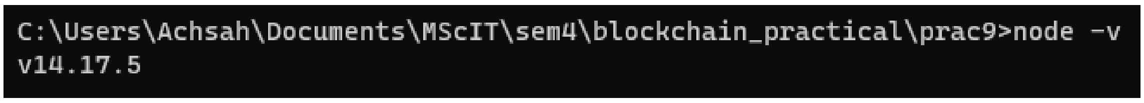
JavaScript Code

```
const SHA256 = require("crypto-js/sha256");class Block {
    constructor(index, timestamp, data, previousHash = "") {this.index = index;
        this.timestamp = timestamp; this.data = data;
        this.previousHash = previousHash;this.hash =
        this.calculateHash();
    }

    calculateHash() {return
        SHA256(
            this.index + this.previousHash +
                this.timestamp +
                JSON.stringify(this.data)
        ).toString();
    }
}

class Blockchain {
    constructor() {
        this.chain = [this.createGenesisBlock()];
    }

    createGenesisBlock() {
        return new Block(0, "21/04/2023", "Genesis Block", "0");
    }

    getLatestBlock() {
        return this.chain[this.chain.length - 1];
    }

    addBlock(newBlock) {
        newBlock.previousHash = this.getLatestBlock().hash;
        newBlock.hash = newBlock.calculateHash();this.chain.push(newBlock);
    }

    isChainValid() {
        for (let i = 1; i < this.chain.length; i +) {const currentBlock =
            this.chain[i];
            const previousBlock = this.chain[i - 1];

            if (currentBlock.hash                    currentBlock.calculateHash()) {return false;
            }

            if (currentBlock.previousHash                      previousBlock.hash) {return false;
            }
        }

        return true;
    }
}

   Blockchain Implementation

let myCoin = new Blockchain();
myCoin.addBlock(new Block(1, "22/04/2023", { amount: 4 }));myCoin.addBlock(new Block(2, "22/04/2023", {
amount: 8 }));
   console.log('Is blockchain valid? ' + myCoin.isChainValid());console.log(JSON.stringify(myCoin, null, 4));
```

Output

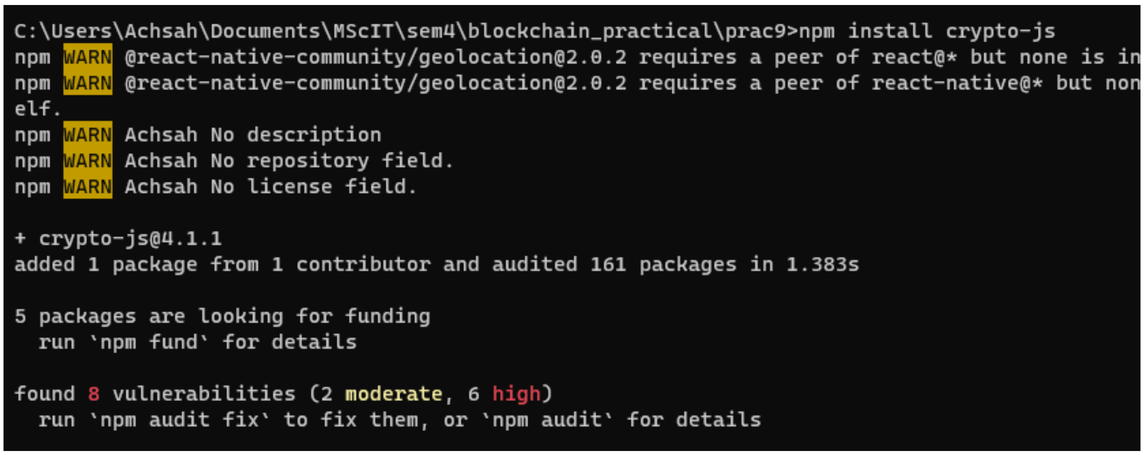
Flow of execution

Step 1-> Make sure you have installed nodejs in your system



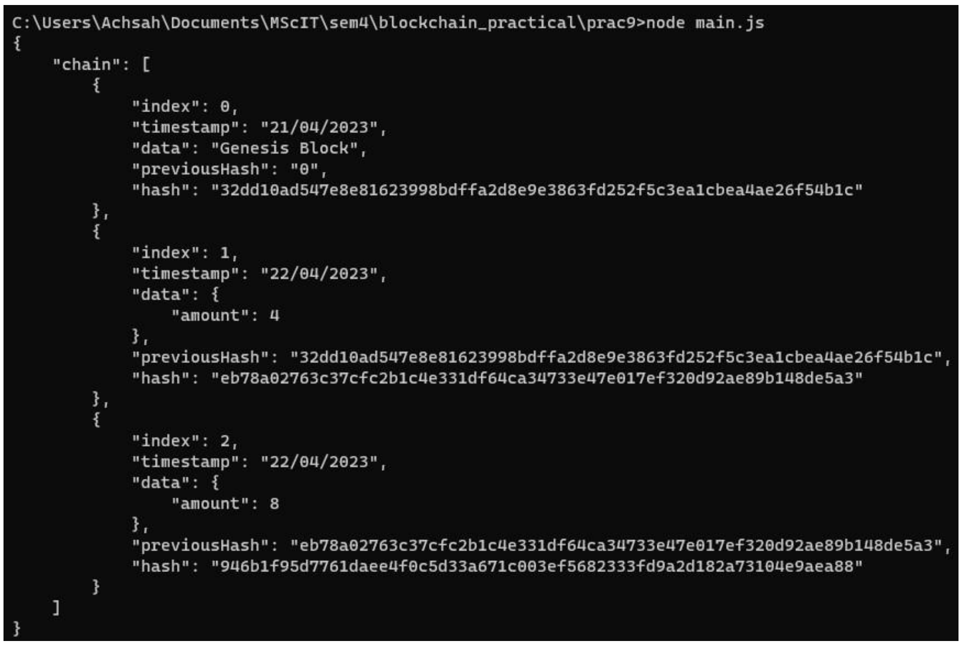
```
C:\Users\Achsah\Documents\MScIT\sem4\blockchain_practical\prac9>node -v
v14.17.5
```

Step 2-> We need **crypto –js** node module to make our own blockchain. Soinstall it as following



```
C:\Users\Achsah\Documents\MScIT\sem4\blockchain_practical\prac9>npm install crypto-js
npm WARN @react-native-community/geolocation@2.0.2 requires a peer of react@* but none is in
npm WARN @react-native-community/geolocation@2.0.2 requires a peer of react-native@* but non
elf.
npm WARN Achsah No description
npm WARN Achsah No repository field.
npm WARN Achsah No license field.

+ crypto-js@4.1.1
added 1 package from 1 contributor and audited 161 packages in 1.383s

5 packages are looking for funding
  run `npm fund` for details

found 8 vulnerabilities (2 moderate, 6 high)
  run `npm audit fix` to fix them, or `npm audit` for details
```

Step 3-> Run the above code in command line using command: node main.js



```
C:\Users\Achsah\Documents\MScIT\sem4\blockchain_practical\prac9>node main.js
{
    "chain": [
        {
            "index": 0,
            "timestamp": "21/04/2023",
            "data": "Genesis Block",
            "previousHash": "0",
            "hash": "32dd10ad547e8e81623998bdffa2d8e9e3863fd252f5c3ea1cbea4ae26f54b1c"
        },
        {
            "index": 1,
            "timestamp": "22/04/2023",
            "data": {
                "amount": 4
            },
            "previousHash": "32dd10ad547e8e81623998bdffa2d8e9e3863fd252f5c3ea1cbea4ae26f54b1c",
            "hash": "eb78a02763c37cfc2b1c4e331df64ca34733e47e017ef320d92ae89b148de5a3"
        },
        {
            "index": 2,
            "timestamp": "22/04/2023",
            "data": {
                "amount": 8
            },
            "previousHash": "eb78a02763c37cfc2b1c4e331df64ca34733e47e017ef320d92ae89b148de5a3",
            "hash": "946b1f95d7761daee4f0c5d33a671c003ef5682333fd9a2d182a73104e9aea88"
        }
    ]
}
```