

BLOCKCHAIN

A PRACTICAL REPORT
ON
BLOCKCHAIN

SUBMITTED BY
Mr. AYUB SAJJADALI KAZI
SEAT NO:

UNDER THE GUIDANCE OF
PROF. DINAAZ SHAIKH

Submitted in fulfillment of the requirements for qualifying
MSc. IT Part II Semester - IV Examination 2023-2024

University of Mumbai
Department of Information Technology

R.D. & S.H National College of Arts, Commerce & W.A.
Science College Bandra (West), Mumbai – 400 050



R. D. National & W. A. Science College

Bandra (W), Mumbai – 400050.

Department of Information Technology
M.Sc. (IT)

Certificate

*This is to certify that **Blockchain Practicals** performed at **R.D & S.H National & W.A.Science College** by Mr. **AYUB SAJJADALI KAZI** holding Seat No. _____ studying Master of Science in Information Technology Part II Semester – IV has been satisfactorily completed as prescribed by the University of Mumbai, during the year 2023 – 2024.*

Lecturer In charge

External Examiner

Head of Department

College Stamp

INDEX

| Sr. No | Date | Practical | Page No. | Sign |
|--------|------------|--|----------|------|
| 1 | 11-05-2024 | Write the following programs for Blockchain in Python: a. A simple client class that generates the private and public keys by using the built-in Python RSA algorithm and test it. | 1 | |
| | | 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. | | |
| 2 | 18-05-2024 | Install and configure Go Ethereum and the Mist browser. | 16 | |
| 3 | 25-05-2024 | 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. | 18 | |
| | | b. Functions, Function Modifiers, View functions, Pure Functions, Fallback Function, Function Overloading, Mathematical functions, Cryptographic functions. | | |
| 4 | 1-06-2024 | Implement and demonstrate the use of the following in Solidity: a. Contracts, Inheritance, Interfaces. | 48 | |

| | | | | |
|---|------------|--|----|--|
| | | b. Libraries, Assembly, Events, Error handling. | | |
| 5 | 08-06-2024 | Install hyperledger fabric and composer. Deploy and execute the application. | 56 | |
| 6 | 15-06-2024 | Create your own blockchain and demonstrate its use. | 64 | |
| 7 | 22-06-2024 | Demonstrate the use of Bitcoin Core API. | 70 | |
| 8 | 29-06-2024 | Building Dapps with Moralis & MetaMask | 72 | |

Practical No. 01

Writeup:

This image shows a full page of white paper with horizontal blue ruling lines. The lines are evenly spaced and run across the width of the page, providing a template for handwriting practice or general writing. There are no margins, text, or other markings on the page.

Practical No: 01

Aim: Write the following programs for Blockchain in Python

- a. A simple client class that generates the private and public keys by using the built in Python RSA algorithm and test it.**

Code:

```
#import libraries
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

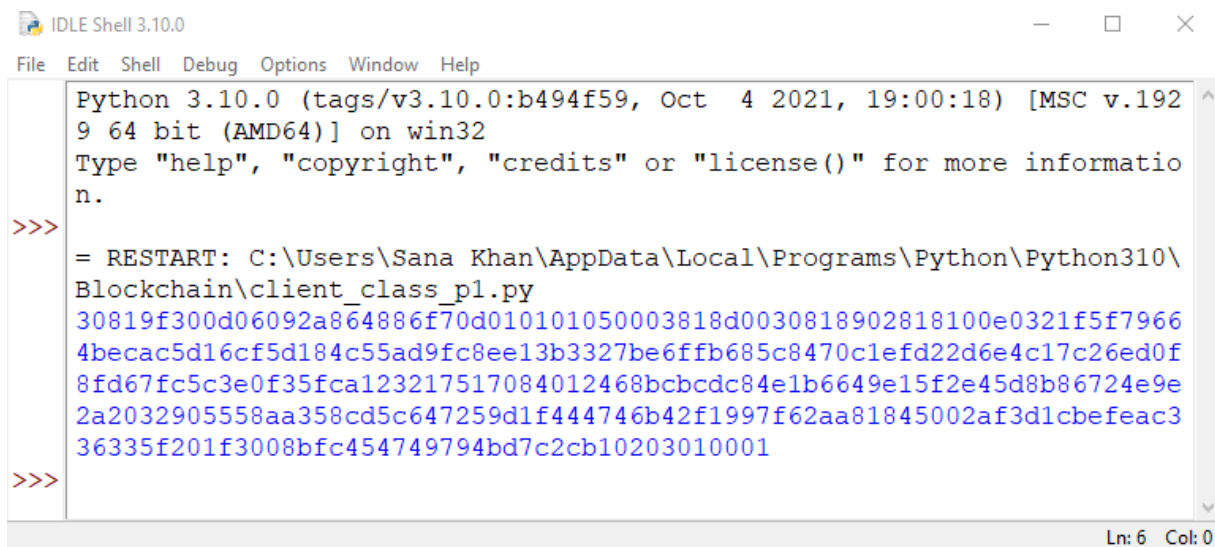
# following imports are required by PKI
import Crypto
import Crypto.Random
from Crypto.Hash import SHA
from Crypto.PublicKey import RSA
from Crypto.Signature import PKCS1_v1_5

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)

    @property
    def identity(self):
        return binascii.hexlify(self._public_key.exportKey(format='DER')).decode('ascii')

Sana = Client()
print(Sana.identity)
```

Output:


```

IDLE Shell 3.10.0
File Edit Shell Debug Options Window Help
Python 3.10.0 (tags/v3.10.0:b494f59, Oct 4 2021, 19:00:18) [MSC v.192
9 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more informatio
n.
>>>
= RESTART: C:\Users\Sana Khan\AppData\Local\Programs\Python\Python310\
Blockchain\client_class_p1.py
30819f300d06092a864886f70d010101050003818d0030818902818100e0321f5f7966
4becac5d16cf5d184c55ad9fc8ee13b3327be6fffb685c8470clefd22d6e4c17c26ed0f
8fd67fc5c3e0f35fca123217517084012468bcbcdc84e1b6649e15f2e45d8b86724e9e
2a2032905558aa358cd5c647259d1f444746b42f1997f62aa81845002af3d1cbefea3
36335f201f3008bfc454749794bd7c2cb10203010001
>>>
Ln: 6 Col: 0

```

b. A transaction class to send and receive money and test it.**Code:**

```

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

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)

```



```
@property
def identity(self):
    return 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()

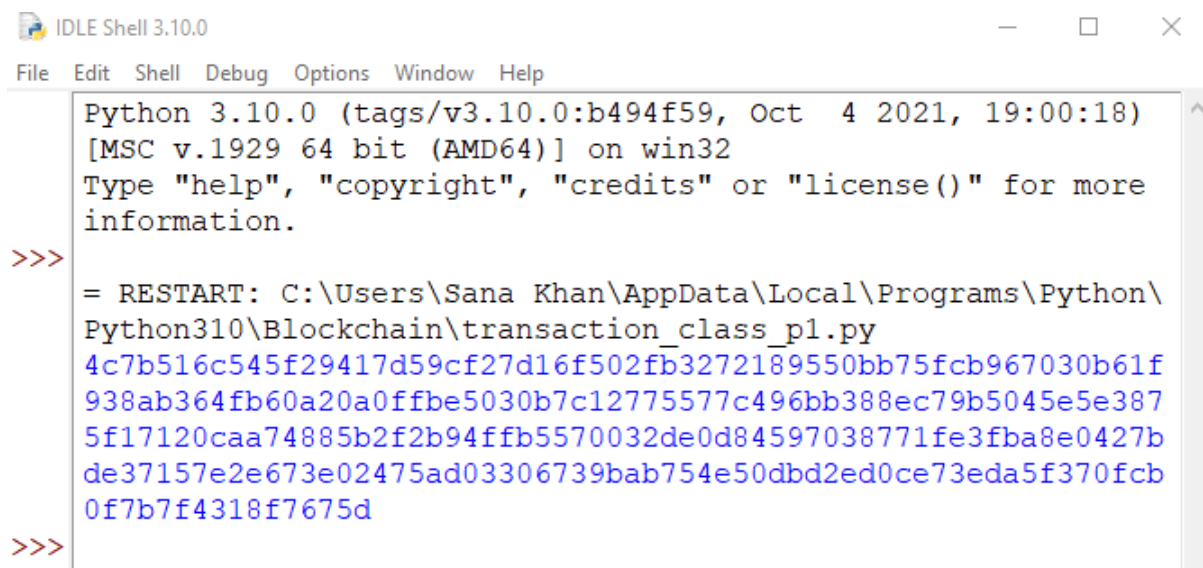
    def to_dict(self):
        if self.sender == "Genesis":
            identity = "Genesis"
        else:
            identity = self.sender.identity
        return collections.OrderedDict({
            'sender': identity,
            'recipient': self.recipient,
            'value': self.value,
            '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')

Sana = Client()
Sarah = Client()

t = Transaction(
    Sana,
    Sarah.identity,
    5.0
)

signature = t.sign_transaction()
print (signature)
```

Output:


```

Python 3.10.0 (tags/v3.10.0:b494f59, Oct 4 2021, 19:00:18)
[MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more
information.
>>>
= RESTART: C:\Users\Sana Khan\AppData\Local\Programs\Python\
Python310\Blockchain\transaction_class_p1.py
4c7b516c545f29417d59cf27d16f502fb3272189550bb75fcb967030b61f
938ab364fb60a20a0ffbe5030b7c12775577c496bb388ec79b5045e5e387
5f17120caa74885b2f2b94ffb5570032de0d84597038771fe3fba8e0427b
de37157e2e673e02475ad03306739bab754e50dbd2ed0ce73eda5f370fcb
0f7b7f4318f7675d
>>>

```

c. Create multiple transactions and display them.**Code:**

```

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

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)

@property
def identity(self):
    return 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')

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 ('.....')
```

Output:

```

>>> ===== RESTART: D:/blockchain/multiple transactions.py =====
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100c89181069a1a4b
c8b00f56170e3b2bd8ebd17e01a718c90ca3113b86660acd6bd160549d2b04ba7d6f4fcd4123e629
edc20f00c545bdbc8899be2265c77a34aca2224604d037c3e6c4e14c3c52bc218147ecef3360760c
8833d9614c61094e9ff5d39a66fdc56de872a9b91499cff6c85d1e79992b035c444fd4a34f724574
890203010001
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100d3f159cd4c4
758ddacad223181a0cdeae1cf85ae89affdc5c34d55e88c458e0985f7af4df52a6c6a08200c5f097
660cc08b6092269ed91390bec05ffcfca52d630d922116445fe941bc41741ea65da10ef3db23d1c0
f2a8c8c79d767a692b3d1b1a75e5a3a202e66d6054bb360d7fa113e3ab1b181009b27e7092739895
722b50203010001
-----
value: 15.0
-----
time: 2022-05-10 18:50:26.221666
-----
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100c89181069a1a4b
c8b00f56170e3b2bd8ebd17e01a718c90ca3113b86660acd6bd160549d2b04ba7d6f4fcd4123e629
edc20f00c545bdbc8899be2265c77a34aca2224604d037c3e6c4e14c3c52bc218147ecef3360760c
8833d9614c61094e9ff5d39a66fdc56de872a9b91499cff6c85d1e79992b035c444fd4a34f724574
890203010001
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100d3e92dafff8
d80c5ee2bed3c0aaf847b810663495d8ee2a630b5a90f3a29be7f269610f69297a569d65f2671d7c
8bd68219ced93780af9ea65cd65e6000a0cfb6922cb7cfb4bbe4275fbaea85d558eec7a7b7dee86
916cba60d1483bdfc3f0e788a27787477e889a95d7a976c684c75ee101f30a505dbb34c5a05cc472
0bc7d0203010001
-----
value: 6.0
-----
time: 2022-05-10 18:50:26.223666
-----
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100d3f159cd4c4758
ddacad223181a0cdeae1cf85ae89affdc5c34d55e88c458e0985f7af4df52a6c6a08200c5f097660
cc08b6092269ed91390bec05ffcfca52d630d922116445fe941bc41741ea65da10ef3db23d1c0f2a
8c8c79d767a692b3d1b1a75e5a3a202e66d6054bb360d7fa113e3ab1b181009b27e7092739895722
b50203010001
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100a76d2fc92bb
d20d127351d92ae1e297dc3686a5f787b103849e0e5d69bac2f180d8e0c6bfe791f9f932c4c7ea05
7af0d9ebc4ca35cf45c945a0dlac9b6af63dd2bdeaf92ece163841d9f3c1cab598a501b2d2882404
bd079b7f35bf12876e0a08fa8dfd9a0c517c6193c348468ba5d1d23c90c347efe9a77459a9618109
514410203010001
-----
value: 2.0
-----
time: 2022-05-10 18:50:26.224665
-----
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100d3e92dafff8d80
c5ee2bed3c0aaf847b810663495d8ee2a630b5a90f3a29be7f269610f69297a569d65f2671d7c8bd
68219ced93780af9ea65cd65e6000a0cfb6922cb7cfb4bbe4275fbaea85d558eec7a7b7dee86916
cba60d1483bdfc3f0e788a27787477e889a95d7a976c684c75ee101f30a505dbb34c5a05cc4720bc
7d0203010001
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100d3f159cd4c4
758ddacad223181a0cdeae1cf85ae89affdc5c34d55e88c458e0985f7af4df52a6c6a08200c5f097
660cc08b6092269ed91390bec05ffcfca52d630d922116445fe941bc41741ea65da10ef3db23d1c0
f2a8c8c79d767a692b3d1b1a75e5a3a202e66d6054bb360d7fa113e3ab1b181009b27e7092739895
722b50203010001
-----
value: 4.0
-----
time: 2022-05-10 18:50:26.225663
-----

```

```

sender: 30819f300d06092a864886f70d010101050003818d0030818902818100a76d2fc92bbd20
d127351d92ae1e297dc3686a5f787b103849e0e5d69bac2f180d8e0c6bfe791f9f932c4c7ea057af
0d9ebc4ca35cf45c945a0dlac9b6af63dd2bdeaf92ece163841d9f3c1cab598a501b2d2882404bd0
79b7f35bf12876e0a08fa8dfd9a0c517c6193c348468ba5d1d23c90c347efe9a77459a9618109514
410203010001
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100d3e92dafff8
d80c5ee2bed3c0aaf847b810663495d8ee2a630b5a90f3a29be7f269610f69297a569d65f2671d7c
8bd68219ced93780af9ea65cd65e6000a0cfb6922cb7cfb4bbe4275fbaea85d558eec7a7b7dee86
916cha60d1483bdfc3f0e788a27787477e889a95d7a976c684c75ee101f30a505dbb34c5a05cc472
0bc7d0203010001
-----
value: 7.0
-----
time: 2022-05-10 18:50:26.227657
-----
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100d3f159cd4c4758
ddacdad223181a0cdeae1cf85ae89affdc5c34d55e88c458e0985f7af4df52a6c6a08200c5f097660
cc08b6092269ed91390bec05ffcfca52d630d922116445fe941bc41741ea65da10ef3db23d1c0f2a
8c8c79d767a692b3d1b1a75e5a3a202e66d6054bb360d7fa113e3ab1b181009b27e7092739895722
b50203010001
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100d3e92dafff8
d80c5ee2bed3c0aaf847b810663495d8ee2a630b5a90f3a29be7f269610f69297a569d65f2671d7c
8bd68219ced93780af9ea65cd65e6000a0cfb6922cb7cfb4bbe4275fbaea85d558eec7a7b7dee86
916cha60d1483bdfc3f0e788a27787477e889a95d7a976c684c75ee101f30a505dbb34c5a05cc472
0bc7d0203010001
-----
value: 3.0
-----
time: 2022-05-10 18:50:26.228624
-----
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100a76d2fc92bbd20
d127351d92ae1e297dc3686a5f787b103849e0e5d69bac2f180d8e0c6bfe791f9f932c4c7ea057af
0d9ebc4ca35cf45c945a0dlac9b6af63dd2bdeaf92ece163841d9f3c1cab598a501b2d2882404bd0
79b7f35bf12876e0a08fa8dfd9a0c517c6193c348468ba5d1d23c90c347efe9a77459a9618109514
410203010001
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100c89181069a1
a4bc8b00f56170e3b2bd8ebd17e01a718c90ca3113b86660acd6bd160549d2b04ba7d6f4fcd4123e
629edc20f00c545bdbc8899be2265c77a34aca2224604d037c3e6c4e14c3c52bc218147ecef33607
60c8833d9614c61094e9ff5d39a66fdc56de872a9b91499cff6c85d1e79992b035c444fd4a34f724
574890203010001
-----
value: 5.0
-----
time: 2022-05-10 18:50:26.232644
-----
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100a76d2fc92bbd20
d127351d92ae1e297dc3686a5f787b103849e0e5d69bac2f180d8e0c6bfe791f9f932c4c7ea057af
0d9ebc4ca35cf45c945a0dlac9b6af63dd2bdeaf92ece163841d9f3c1cab598a501b2d2882404bd0
79b7f35bf12876e0a08fa8dfd9a0c517c6193c348468ba5d1d23c90c347efe9a77459a9618109514
410203010001
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100d3f159cd4c4
758ddacdad223181a0cdeae1cf85ae89affdc5c34d55e88c458e0985f7af4df52a6c6a08200c5f097
660cc08b6092269ed91390bec05ffcfca52d630d922116445fe941bc41741ea65da10ef3db23d1c0
f2a8c8c79d767a692b3d1b1a75e5a3a202e66d6054bb360d7fa113e3ab1b181009b27e7092739895
722b50203010001
-----
value: 3.0
-----
time: 2022-05-10 18:50:26.233641
-----

```

```

sender: 30819f300d06092a864886f70d010101050003818d0030818902818100d3e92dafff8d80
c5ee2bed3c0aaf847b810663495d8ee2a630b5a90f3a29be7f269610f69297a569d65f2671d7c8bd
68219ced93780af9ea65cd65e6000a0cfb6922cb7cfb4bbe4275fbaea85d558eec7a7b7dee86916
cba60d1483bdfc3f0e788a27787477e889a95d7a976c684c75ee101f30a505dbb34c5a05cc4720bc
7d0203010001
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100c89181069a1
a4bc8b00f56170e3b2bd8ebd17e01a718c90ca3113b86660acd6bd160549d2b04ba7d6f4fcd4123e
629edc20f00c545bdbc8899be2265c77a34aca2224604d037c3e6c4e14c3c52bc218147ecef33607
60c8833d9614c61094e9ff5d39a66fdc56de872a9b91499cff6c85d1e79992b035c444fd4a34f724
574890203010001
-----
value: 8.0
-----
time: 2022-05-10 18:50:26.229652
-----
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100d3e92dafff8d80
c5ee2bed3c0aaf847b810663495d8ee2a630b5a90f3a29be7f269610f69297a569d65f2671d7c8bd
68219ced93780af9ea65cd65e6000a0cfb6922cb7cfb4bbe4275fbaea85d558eec7a7b7dee86916
cba60d1483bdfc3f0e788a27787477e889a95d7a976c684c75ee101f30a505dbb34c5a05cc4720bc
7d0203010001
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100d3f159cd4c4
758ddacac223181a0cdea1cf85ae89affdc5c34d55e88c458e0985f7af4df52a6c6a08200c5f097
660cc08b6092269ed91390bec05ffcfa52d630d922116445fe941bc41741ea65da10ef3db23d1c0
f2a8c8c79d767a692b3d1b1a75e5a3a202e66d6054bb360d7fa113e3ab1b181009b27e7092739895
722b50203010001
-----
value: 1.0
-----
time: 2022-05-10 18:50:26.230649
-----

```

d. Create a blockchain, a genesis block and execute it.

Code:

```

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

```



```
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)

    @property
    def identity(self):
        return 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')

def display_transaction(transaction):
    #for transaction in transactions:
    dict = transaction.to_dict()
    print ("sender: " + dict['sender'])
    print (' ----')
    print ("recipient: " + dict['recipient'])
    print (' ----')
    print ("value: " + str(dict['value']))
    print (' ----')
```

```

    print ("time: " + str(dict['time']))
    print (' ----')

class Block:
    def __init__(self):
        self.verified_transactions = []
        self.previous_block_hash = ""
        self.Nonce = ""
last_block_hash = ""

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 ('=====')

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 = []
TPCoins.append (block0)
dump_blockchain(TPCoins)

```

Output:

```

>>>
===== RESTART: D:/blockchain/addingBlockchain-addingGenesisBlock.py =====
Number of blocks in the chain: 1
block # 0
sender: Genesis
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100a567f72c685
1771a0239036e044b7bdc83819fbedc5a6479b5c180c414c4dfb036e3ed0eb472ebfbel7ef61d358
29f2280006307b23c72321b8e3de8f972e887b31080ef495b0c2e03f7ed7e62e05e826ca523386c7
863f29c37b6c3d94c621823c7771dac2370578efe2e73820934238968bleefdd038474dfcf971505
c76d70203010001
-----
value: 500.0
-----
time: 2022-05-10 20:16:05.127628
-----
=====

```

e. Create a mining function 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

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

def mine(message, difficulty=1):
    assert difficulty >= 1
    prefix = '1' * difficulty

```

```
for i in range(1000):
    digest = sha256(str(hash(message)) + str(i))
    if digest.startswith(prefix):
        print ("after " + str(i) + " iterations found nonce: " + digest)
    return digest
mine ("test message", 2)
```

Output:

```
>>> = RESTART: C:\Users\Sana Khan\AppData\Local\Programs\Python\Python
310\Blockchain\miningfunc.py
after 16 iterations found nonce: 11dfa4d4222c51d9c3c85a64c146327c9
73d799be08dd80a1f6e7122736a9bc0
>>>
```

Writeup:

This image shows a blank sheet of white paper with horizontal ruling lines. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.

Practical No: 02

Aim: Install and configure Go Ethereum and the Mist browser.

Installing GETH (Go Ethereum)

Step 1: Go to website <https://geth.ethereum.org/downloads/>

Step 2: From stable releases Geth 1.5.8 (kind = installer)

Step 3: once downloaded run it then click next

Step 4: Select Geth and Development tools click next

Step 5: Select location to install click next

Step 6: Once Installation is finished Click Close and its done

Installing Mist Browser

Step 1: <https://github.com/ethereum/mist/releases>

Step 2: Under Ethereum Wallet and Mist 0.8.9 - "The Wizard" download mist-installer-0-8-9.exe

Step 3: For installation click, I agree -> next -> install

Run Mist

Step 1: Open the Mist from the start menu

Step 2: It will start downloading Blockchain data once you open it

Step 3: Once it finishes downloading it is ready to use

Run Geth

Step 1: Open CMD

Step 2: Type GETH and press enter

Step 3: After it finishes loading press ctrl+c to exit the process.

Step 4: Now it's ready to use

Writeup:

This image shows a single sheet of white paper with horizontal ruling lines. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.

Practical No: 03

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

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

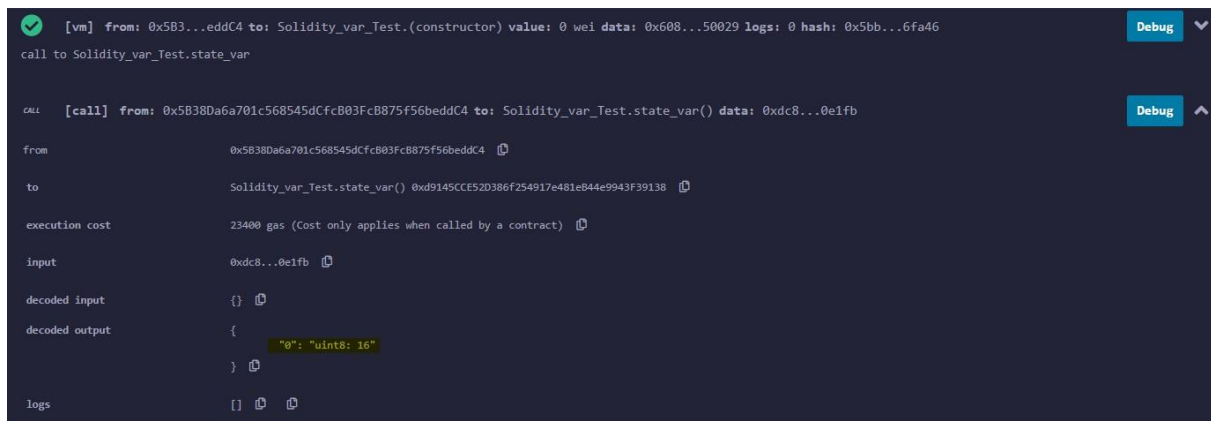
Variable

Code:

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

```
// Creating a contract
contract Solidity_var_Test {
// Declaring a state variable
uint8 public state_var;
// Defining a constructor
constructor() public {
    state_var = 16;
}
}
```

Output:



Operators

a. Arithmetic Operator

Code:

```
// Solidity contract to demonstrate
// Arithmetic Operator
pragma solidity ^0.5.0;

// Creating a contract
contract SolidityTest {

    // Initializing variables
    uint16 public a = 20;
    uint16 public b = 10;

    // Initializing a variable
    // with sum
    uint public sum = a + b;

    // Initializing a variable
    // with the difference
    uint public diff = a - b;

    // Initializing a variable
    // with product
    uint public mul = a * b;

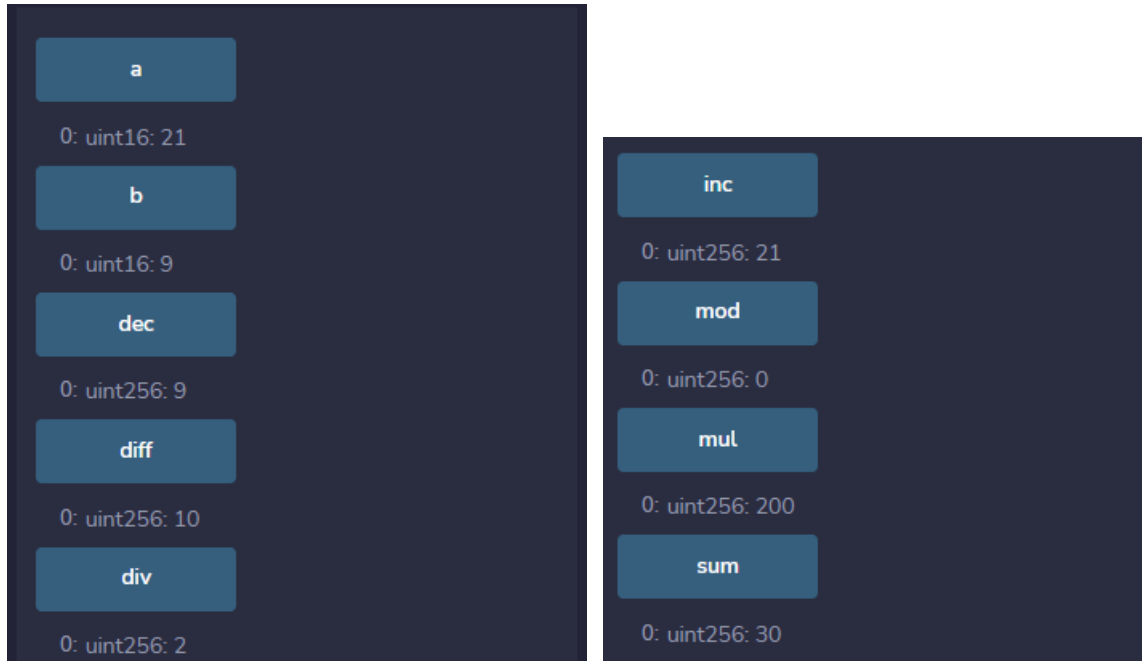
    // Initializing a variable
    // with quotient
    uint public div = a / b;

    // Initializing a variable
    // with modulus
    uint public mod = a % b;

    // Initializing a variable
    // decrement value
    uint public dec = --b;

    // Initializing a variable
    // with increment value
    uint public inc = ++a;
```

```
}
```

Output:**b. Relational Operator****Code:**

```
// Solidity program to demonstrate
// Relational Operator
pragma solidity ^0.5.0;
// Creating a contract
contract SolidityTest {
    // Declaring variables
    uint16 public a = 20;
    uint16 public b = 10;

    // Initializing a variable
    // with bool equal result
    bool public eq = a == b;

    // Initializing a variable
    // with bool not equal result
    bool public noteq = a != b;

    // Initializing a variable
    // with bool greater than result
    bool public gtr = a > b;
```

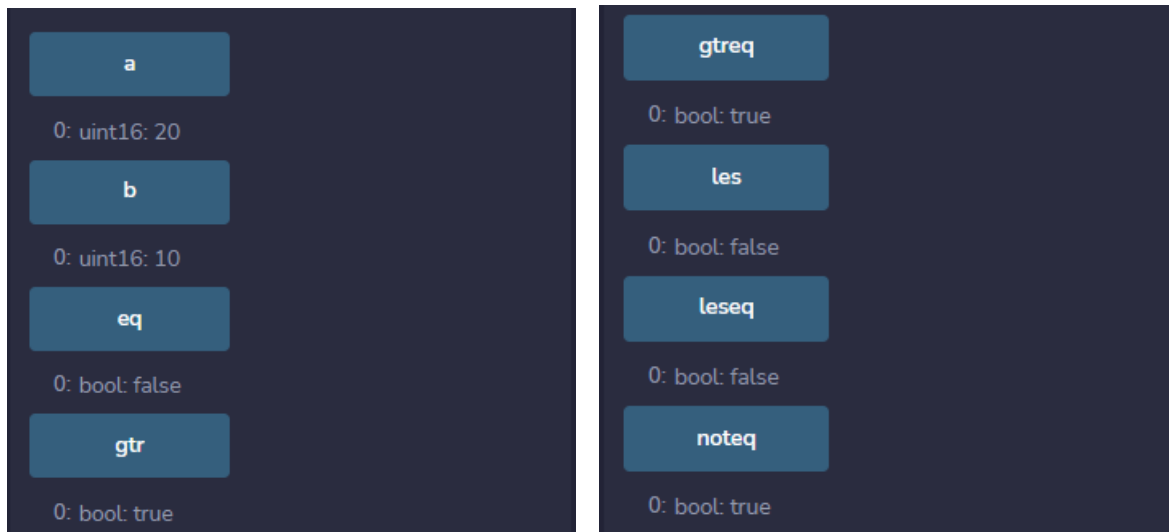
```

// Initializing a variable
// with bool less than result
bool public les = a < b;

// Initializing a variable
// with bool greater than equal to result
bool public gtreq = a >= b;

// Initializing a variable
// bool less than equal to result
bool public leseq = a <= b;
}

```

Output:**c. Logical Operator****Code:**

```

// Solidity program to demonstrate Logical Operators
pragma solidity ^0.5.0;

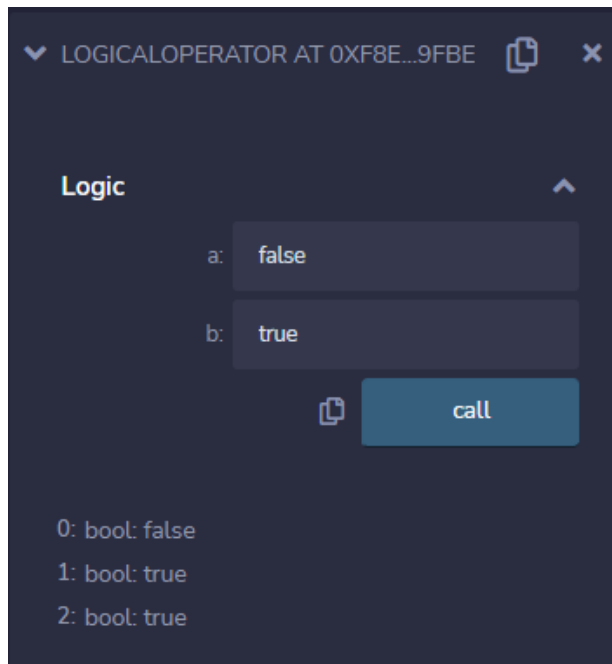
// Creating a contract
contract LogicalOperator {
    // Defining function to demonstrate Logical operators
    function logic(bool a, bool b) public pure returns (bool, bool, bool) {
        // Logical AND operator
        bool andResult = a && b;

        // Logical OR operator
        bool orResult = a || b;

        // Logical NOT operator
        bool notResult = !a;

        return (andResult, orResult, notResult);
    }
}

```

Output:**d. Bitwise Operator.****Code:**

```
// Solidity program to demonstrate
// Bitwise Operator
pragma solidity ^0.5.0;

// Creating a contract
contract SolidityTest {

    // Declaring variables
    uint16 public a = 20;
    uint16 public b = 10;
```

```
// Initializing a variable
// to '&' value
uint16 public and = a & b;

// Initializing a variable
// to '|' value
uint16 public or = a | b;

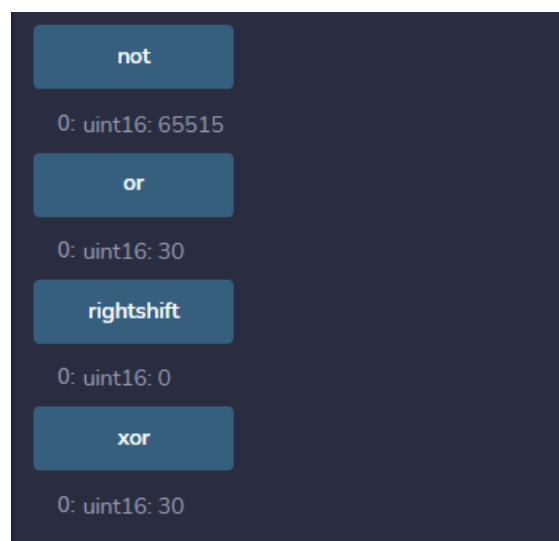
// Initializing a variable
// to '^' value
uint16 public xor = a ^ b;

// Initializing a variable
// to '<<' value
uint16 public leftshift = a << b;

// Initializing a variable
// to '>>' value
uint16 public rightshift = a >> b;

// Initializing a variable
// to '~' value
uint16 public not = ~a ;

}
```

Output:

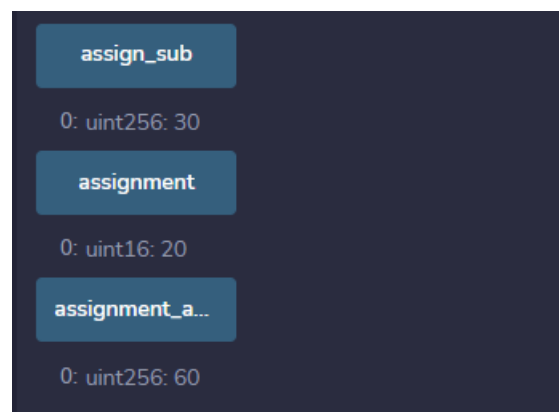
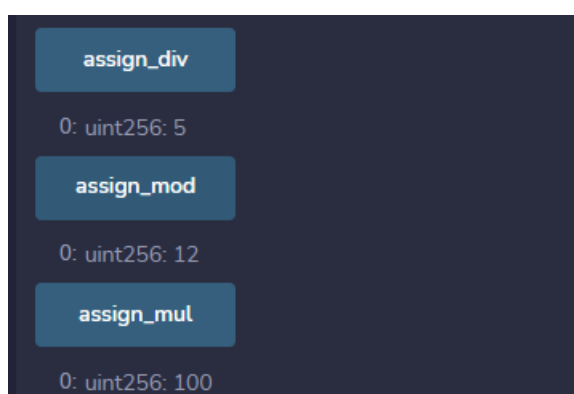
e. Assignment Operator.**Code:**

```
// Solidity program to demonstrate
// Assignment Operator
pragma solidity ^0.5.0;

// Creating a contract
contract SolidityTest {

    // Declaring variables
    uint16 public assignment = 20;
    uint public assignment_add = 50;
    uint public assign_sub = 50;
    uint public assign_mul = 10;
    uint public assign_div = 50;
    uint public assign_mod = 32;

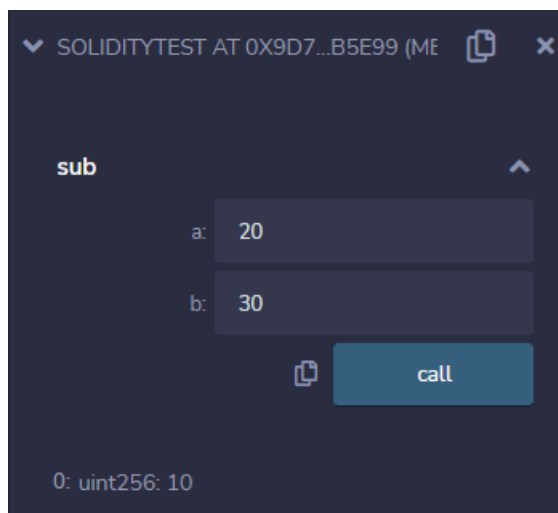
    // Defining function to
    // demonstrate Assignment Operator
    function getResult() public{
        assignment_add += 10;
        assign_sub -= 20;
        assign_mul *= 10;
        assign_div /= 10;
        assign_mod %= 20;
        return ;
    }
}
```

Output:

f. Conditional Operator.**Code:**

```
pragma solidity ^0.5.0;

contract SolidityTest {
    function sub(uint a, uint b) public pure returns (uint) {
        uint result = (a > b) ? a - b : b - a;
        return result;
    }
}
```

Output:

Loops

a. While Loop.

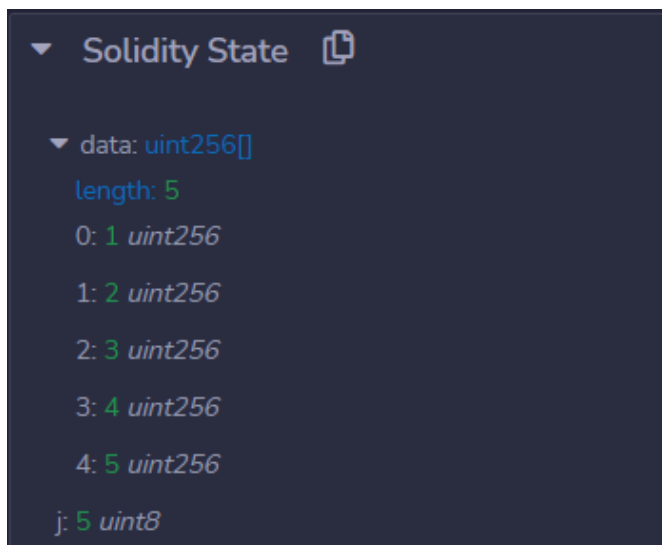
```
pragma solidity ^0.5.0;

// Creating a contract
contract Types {
    // Declaring a dynamic array
    uint[] data;

    // Declaring state variable
    uint8 j = 0;

    // Defining a function to demonstrate the use of 'While loop'
    function loop() public returns (uint[] memory) {
        while (j < 5) {
            j++;
            data.push(j);
        }
        return data;
    }
}
```

Output:



b. Do-While Loop.**Code:**

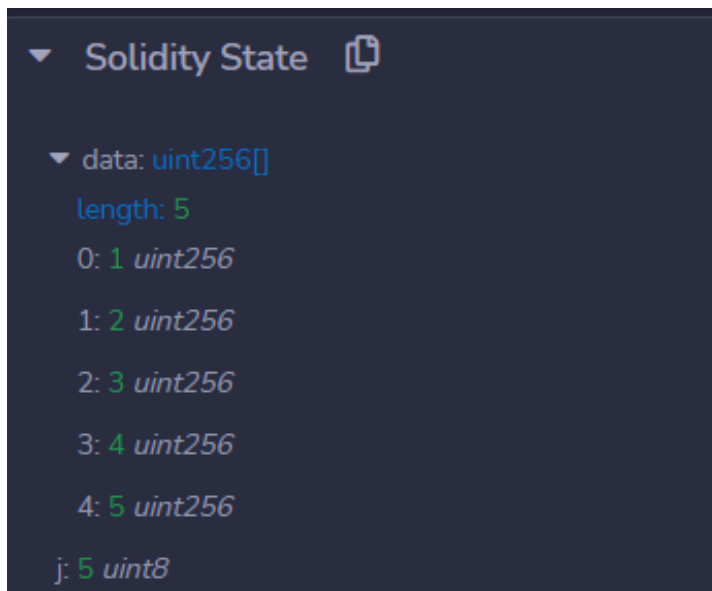
```
pragma solidity ^0.5.0;

// Creating a contract
contract Types {
    // Declaring a dynamic array
    uint[] data;

    // Declaring state variable
    uint8 j = 0;

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

        return data;
    }
}
```

Output:

c. For Loop.**Code:**

```
pragma solidity ^0.5.0;

// Creating a contract
contract Types {
    // Declaring a dynamic array
    uint[] data;

    // Defining a function to demonstrate 'For loop'
    function loop() public returns (uint[] memory) {
        for (uint i = 0; i < 5; i++) {
            data.push(i);
        }
        return data;
    }
}
```

Output:

Decision Making

If Statement

Code:

```
pragma solidity ^0.5.0;

contract SolidityTest {
    uint storedData;
    constructor() public {
        storedData = 10;
    }
    function getResult() public view returns (string memory) {
        uint a = 1;
        uint b = 2;
        uint result = a + b;
        return integerToString(result);
    }
    function integerToString(uint _i) internal pure returns (string memory) {
        if (_i == 0) {
            return "0";
        }
        uint j = _i;
        uint len;
        while (j != 0) {
            len++;
            j /= 10;
        }
        bytes memory bstr = new bytes(len);
        uint k = len - 1;
        while (_i != 0) {
            bstr[k--] = byte(uint8(48 + _i % 10));
            _i /= 10;
        }
        return string(bstr);
    }
}
```

Output:**If else statement****Code:**

```
pragma solidity ^0.5.0;
contract SolidityTest {
    uint storedData;
    constructor() public{
        storedData = 10;
    }
    function getResult() public view returns(string memory){
        uint a = 1;
        uint b = 2;
        uint result;
        if( a > b) { // if else statement
            result = a;
        }
        else {
            result = b;
        }
        return integerToString(result);
    }
    function integerToString(uint _i) internal pure
```

```

returns (string memory) {
    if (_i == 0) {
        return "0";
    }
    uint j = _i;
    uint len;

    while (j != 0) {
        len++;
        j /= 10;
    }
    bytes memory bstr = new bytes(len);
    uint k = len - 1;

    while (_i != 0) {
        bstr[k--] = byte(uint8(48 + _i % 10));
        _i /= 10;
    }
    return string(bstr); //access local variable
}
}

```

Output:

```

CALL [call] from: 0x5B38Da6a701c568545dCfc803Fc8875f56beddC4 to: SolidityTest.getResult() data: 0xde2...92789

from 0x5B38Da6a701c568545dCfc803Fc8875f56beddC4
to SolidityTest.getResult() 0xf8e81D47203A594245E36C48e151709F0C19fBe8
execution cost 22314 gas (Cost only applies when called by a contract)
input 0xde2...92789
decoded input {}
decoded output {
  "0": "string: 2"
}
logs

```

If-else-If statement**Code:**

```
pragma solidity ^0.5.0;
contract SolidityTest {
    uint storedData; // State variable
    constructor() public {
        storedData = 10;
    }
    function getResult() public view returns(string memory) {
        uint a = 1;
        uint b = 2;
        uint c = 3;
        uint result

        if( a > b && a > c) { // if else statement
            result = a;
        } else if( b > a && b > c ){
            result = b;
        } else {
            result = c;
        }
        return integerToString(result);
    }
    function integerToString(uint _i) internal pure
        returns (string memory) {

        if (_i == 0) {
            return "0";
        }
        uint j = _i;
        uint len;

        while (j != 0) {
```

```
len++;  
j /= 10;  
}  
bytes memory bstr = new bytes(len);  
uint k = len - 1;  
  
while (_i != 0) {  
    bstr[k--] = byte(uint8(48 + _i % 10));  
    _i /= 10;  
}  
return string(bstr); //access local variable  
}  
}
```

Output:

Strings

Code:

```
pragma solidity ^0.5.0;

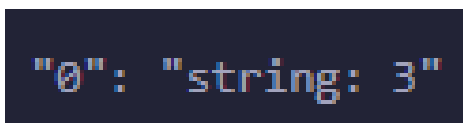
contract SolidityTest {
    constructor() public{
    }
    function getResult() public view returns(string memory){
        uint a = 1;
        uint b = 2;
        uint result = a + b;
        return integerToString(result);
    }
    function integerToString(uint _i) internal pure
        returns (string memory) {

        if (_i == 0) {
            return "0";
        }
        uint j = _i;
        uint len;

        while (j != 0) {
            len++;
            j /= 10;
        }
        bytes memory bstr = new bytes(len);
        uint k = len - 1;

        while (_i != 0) {
            bstr[k--] = byte(uint8(48 + _i % 10));
            _i /= 10;
        }
        return string(bstr);
    }
}
```

Output:

A terminal window with a dark background and light-colored text. The text displays the output of a Solidity test, showing a mapping from the value '0' to the string 'string: 3'.

Arrays

Code:

```
// Solidity program to demonstrate
// creating a fixed-size array
pragma solidity ^0.5.0;

// Creating a contract
contract Types {

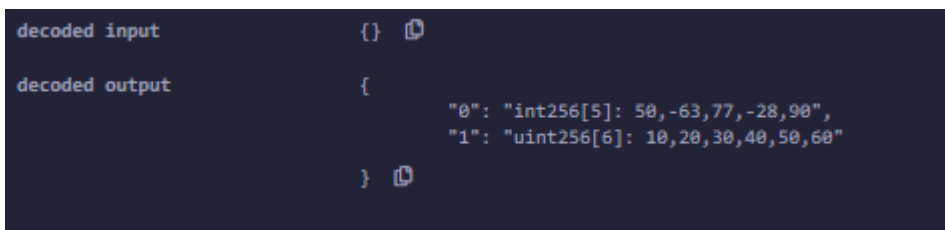
    // Declaring state variables
    // of type array
    uint[6] data1;

    // Defining function to add
    // values to an array
    function array_example() public returns (
        int[5] memory, uint[6] memory){

        int[5] memory data
        = [int(50), -63, 77, -28, 90];
        data1
        = [uint(10), 20, 30, 40, 50, 60];

        return (data, data1);
    }
}
```

Output:



```
decoded input      {}
decoded output     {
                    "0": "int256[5]: 50,-63,77,-28,90",
                    "1": "uint256[6]: 10,20,30,40,50,60"
                    }
```

Enums**Code:**

```
pragma solidity ^0.5.0;

contract test {
    enum FreshJuiceSize{ SMALL, MEDIUM, LARGE }
    FreshJuiceSize choice;
    FreshJuiceSize constant defaultChoice = FreshJuiceSize.MEDIUM;

    function setLarge() public {
        choice = FreshJuiceSize.LARGE;
    }
    function getChoice() public view returns (FreshJuiceSize) {
        return choice;
    }
    function getDefaultChoice() public pure returns (uint) {
        return uint(defaultChoice);
    }
}
```

Output:

On clicking getChoice

```
{
  "0": "uint8: 2"
}
```

On clicking getDefaultChocie

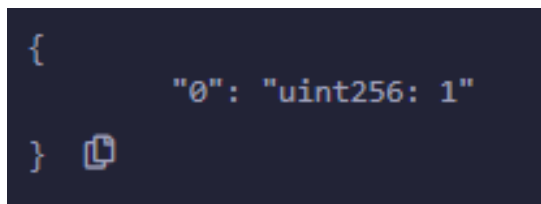
```
{
  "0": "uint256: 1"
}
```

Structs**Code:**

```
pragma solidity ^0.5.0;

contract test {
    struct Book {
        string title;
        string author;
        uint book_id;
    }
    Book book;

    function setBook() public {
        book = Book('Learn Java', 'TP', 1);
    }
    function getBookId() public view returns (uint) {
        return book.book_id;
    }
}
```

Output:

```
{
  "0": "uint256: 1"
}
```

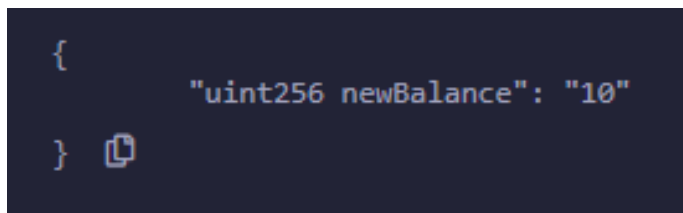
Mapping**Code:**

```
pragma solidity ^0.5.0;

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

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

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

Output:

```
{
  "uint256 newBalance": "10"
}
```

b. Functions, Function Modifiers, View functions, Pure Functions, Fallback Function, Function Overloading, Mathematical functions, Cryptographic functions.**Functions****Code:**

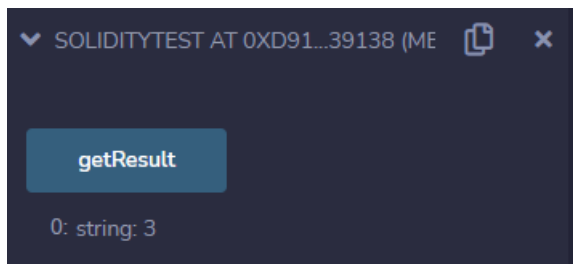
```
pragma solidity ^0.5.0;

contract SolidityTest {
    constructor() public{
    }
    function getResult() public view returns(string memory){
        uint a = 1;
        uint b = 2;
        uint result = a + b;
        return integerToString(result);
    }
    function integerToString(uint _i) internal pure
        returns (string memory) {

        if (_i == 0) {
            return "0";
        }
        uint j = _i;
        uint len;

        while (j != 0) {
            len++;
            j /= 10;
        }
        bytes memory bstr = new bytes(len);
        uint k = len - 1;

        while (_i != 0) {
            bstr[k--] = byte(uint8(48 + _i % 10));
            _i /= 10;
        }
        return string(bstr);//access local variable
    }
}
```

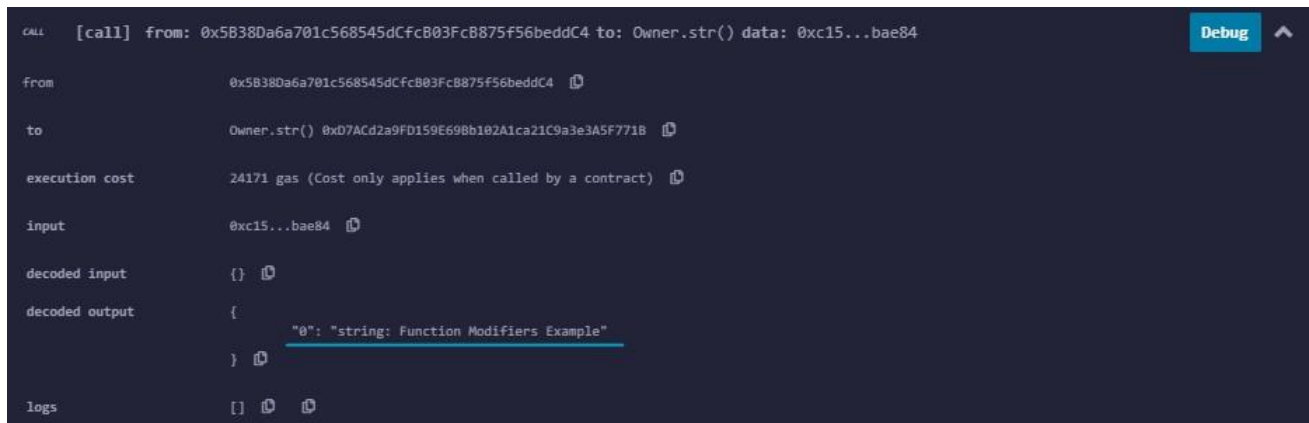
Output:**Function Modifiers****Code:**

```
pragma solidity ^0.5.0;

contract Owner {
    address owner;
    string public str = "Function Modifiers Example";
    constructor() public {
        owner = msg.sender;
    }
    modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }
    modifier costs(uint price) {
        if (msg.value >= price) {
            _;
        }
    }
}

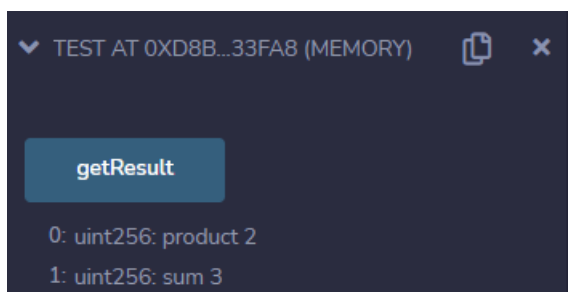
contract Register is Owner {
    mapping (address => bool) registeredAddresses;
    uint price;
    constructor(uint initialPrice) public { price = initialPrice; }

    function register() public payable costs(price) {
        registeredAddresses[msg.sender] = true;
    }
    function changePrice(uint _price) public onlyOwner {
        price = _price;
    }
}
```

Output:**View Function****Code:**

```
pragma solidity ^0.5.0;
```

```
contract Test {
    function getResult() public view returns(uint product, uint sum){
        uint a = 1; // local variable
        uint b = 2;
        product = a * b;
        sum = a + b;
    }
}
```

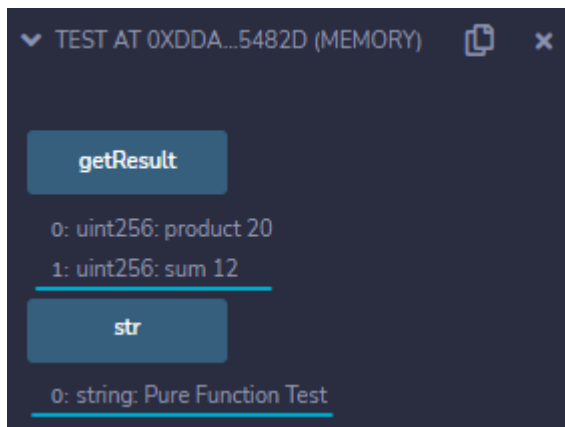
Output:

Pure Function**Code:**

```
pragma solidity ^0.5.0;

contract Test {

    function getResult() public pure returns(uint product, uint sum){
        uint a = 10;
        uint b = 2;
        product = a * b;
        sum = a + b;
    }
    string public str = "Pure Function Test";
}
```

Output:**Fallback Function:****Code:**

```
pragma solidity ^0.5.0;

contract Test {
    uint public x ;
    function() external { x = 1; }
}

contract Sink {
    function() external payable { }
}

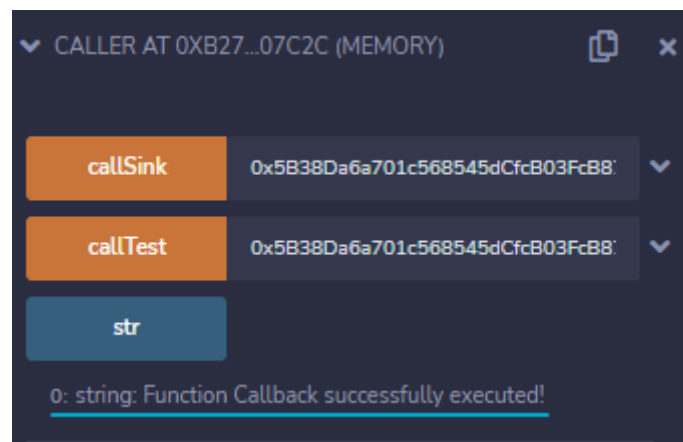
contract Caller {
```

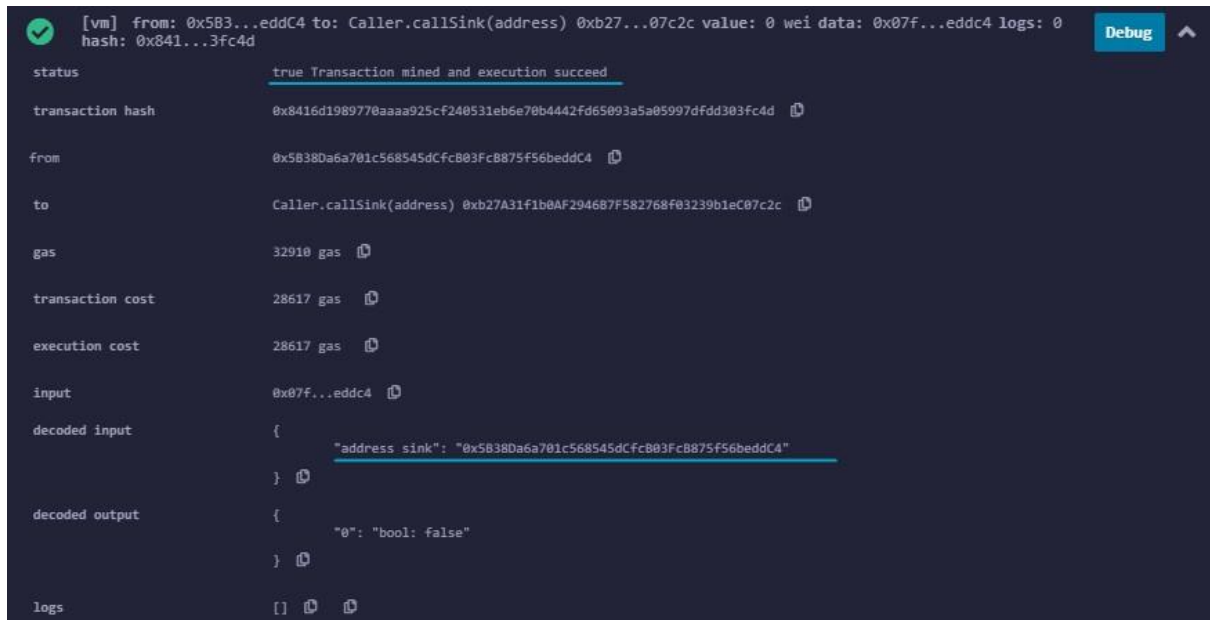


```
function callTest(Test test) public returns (bool) {
    (bool success,) = address(test).call(abi.encodeWithSignature("nonExistingFunction()"));
    require(success);
    // test.x is now 1

    address payable testPayable = address(uint160(address(test)));

    // Sending ether to Test contract,
    // the transfer will fail, i.e. this returns false here.
    return (testPayable.send(2 ether));
}
function callSink(Sink sink) public returns (bool) {
    address payable sinkPayable = address(sink);
    return (sinkPayable.send(2 ether));
}
string public str = "Function Callback successfully executed!";
}
```

Output:



[vm] from: 0x583...eddc4 to: Caller.callSink(address) 0xb27...07c2c value: 0 wei data: 0x07f...eddc4 logs: 0
hash: 0x841...3fc4d

status: true Transaction mined and execution succeed

transaction hash: 0x8416d1989770aaaa925cf240531eb6e70b4442fd65093a5a05997dfdd303fc4d

from: 0x5838Da6a701c568545dCfcB03FcB875f56beddC4

to: Caller.callSink(address) 0xb27A31f1b0AF294687F582768f03239b1eC07c2c

gas: 32910 gas

transaction cost: 28617 gas

execution cost: 28617 gas

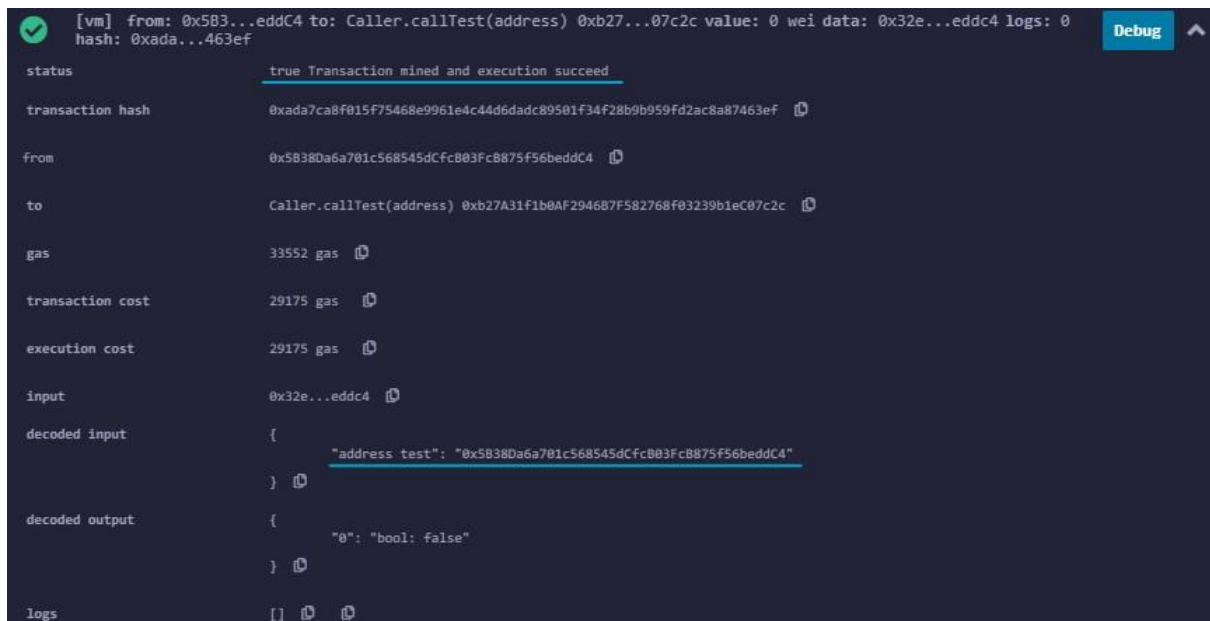
input: 0x07f...eddc4

decoded input: {
 "address sink": "0x5838Da6a701c568545dCfcB03FcB875f56beddC4"
}

decoded output: {
 "0": "bool: false"
}

logs: []

Call Sink



[vm] from: 0x583...eddc4 to: Caller.callTest(address) 0xb27...07c2c value: 0 wei data: 0x32e...eddc4 logs: 0
hash: 0xada...463ef

status: true Transaction mined and execution succeed

transaction hash: 0xada7ca8f015f75468e9961e4c44d6dadcb9501f34f28b9b959fd2ac8a87463ef

from: 0x5838Da6a701c568545dCfcB03FcB875f56beddC4

to: Caller.callTest(address) 0xb27A31f1b0AF294687F582768f03239b1eC07c2c

gas: 33552 gas

transaction cost: 29175 gas

execution cost: 29175 gas

input: 0x32e...eddc4

decoded input: {
 "address test": "0x5838Da6a701c568545dCfcB03FcB875f56beddC4"
}

decoded output: {
 "0": "bool: false"
}

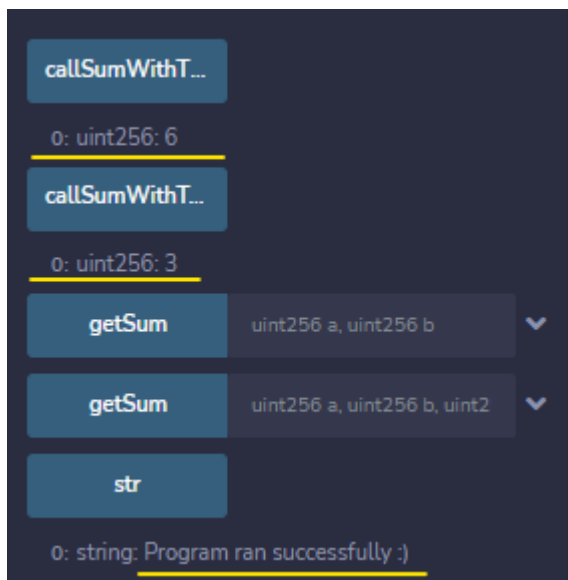
logs: []

Call Test

Function Overloading:**Code:**

```
pragma solidity ^0.5.0;
```

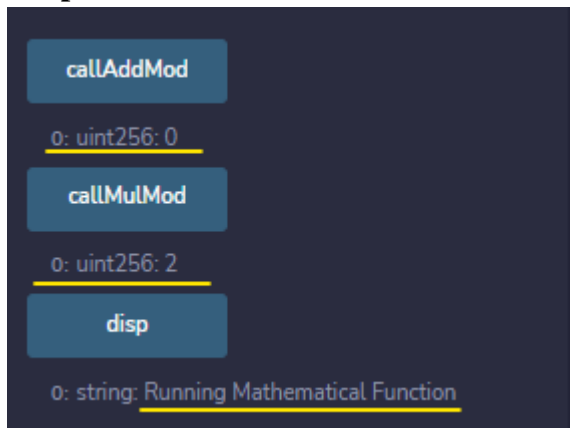
```
contract Test {  
    string public str="Program ran successfully :)";  
    function getSum(uint a, uint b) public pure returns(uint){  
        return a + b;  
    }  
    function getSum(uint a, uint b, uint c) public pure returns(uint){  
        return a + b + c;  
    }  
    function callSumWithTwoArguments() public pure returns(uint){  
        return getSum(1,2);  
    }  
    function callSumWithThreeArguments() public pure returns(uint){  
        return getSum(1,2,3);  
    }  
}
```

Output:

Mathematical Functions:**Code:**

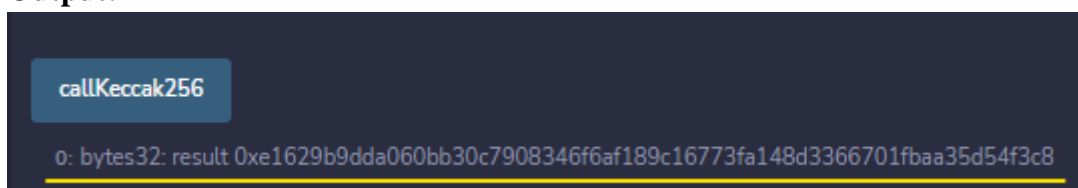
```
pragma solidity ^0.5.0;

contract Test {
    string public disp="Running Mathematical Function";
    function callAddMod() public pure returns(uint){
        return addmod(4, 5, 3);
    }
    function callMulMod() public pure returns(uint){
        return mulmod(4, 5, 3);
    }
}
```

Output:**Cryptographic Functions:****Code:**

```
pragma solidity ^0.5.0;

contract Test {
    function callKeccak256() public pure returns(bytes32 result){
        return keccak256("ABC");
    }
}
```

Output:

Writeup:

This image shows a single sheet of white paper with horizontal ruling lines. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.

Practical No: 04

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

a. Contracts, Inheritance, Constructors, Abstract Contracts, Interfaces.

Contracts:

Code:

```
pragma solidity ^0.5.0;

contract C {
    // Private state variable
    uint private data;

    // Public state variable
    uint public info;

    // Constructor
    constructor() public {
        info = 10;
    }

    // Private function
    function increment(uint a) private pure returns (uint) {
        return a + 1;
    }

    // Public function
    function updateData(uint a) public {
        data = a;
    }

    function getData() public view returns (uint) {
        return data;
    }

    function compute(uint a, uint b) internal pure returns (uint) {
        return a + b;
    }
}

// External Contract
contract D {
    function readData() public returns (uint) {
        C c = new C();
        c.updateData(7);
        return c.getData();
    }
}
```

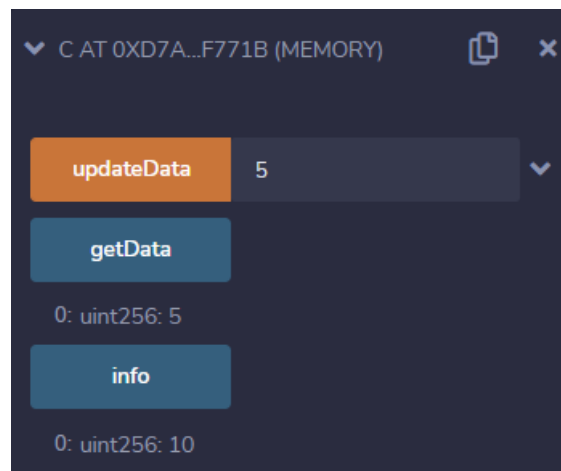
```
// Derived Contract
contract E is C {
    uint private result;
    C private c;

    constructor() public {
        c = new C();
    }

    function getComputedResult() public {
        result = compute(3, 5);
    }

    function getResult() public view returns (uint) {
        return result;
    }

    function getData() public view returns (uint) {
        return c.info();
    }
}
```

Output:

Inheritance

Code:

```
pragma solidity ^0.5.0;

contract C {
    // Private state variable
    uint private data;

    // Public state variable
    uint public info;

    // Constructor
    constructor() public {
        info = 10;
    }

    // Private function
    function increment(uint a) private pure returns (uint) {
        return a + 1;
    }

    // Public function
    function updateData(uint a) public {
        data = a;
    }

    function getData() public view returns (uint) {
        return data;
    }

    function compute(uint a, uint b) internal pure returns (uint) {
        return a + b;
    }
}

// External Contract
contract D {
    function readData() public returns (uint) {
        C c = new C();
        c.updateData(7);
        return c.getData();
    }
}

// Derived Contract
contract E is C {
    uint private result;
    C private c;

    constructor() public {
        c = new C();
    }

    function getComputedResult() public {
        result = compute(3, 5);
    }

    function getResult() public view returns (uint) {
        return result;
    }
}
```

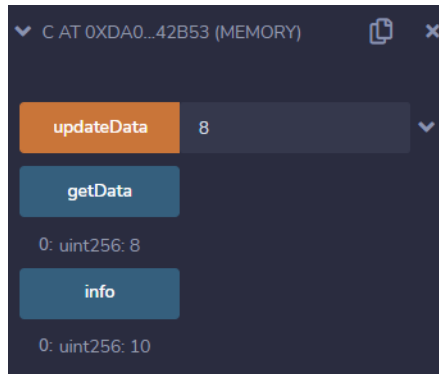


```

}

function getData() public view returns (uint) {
    return c.info();
}
}

```

Output:**Interfaces****Code:**

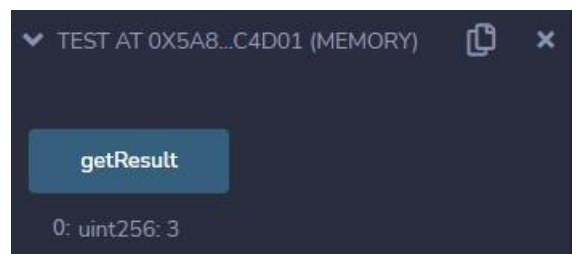
```

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;
    }
}

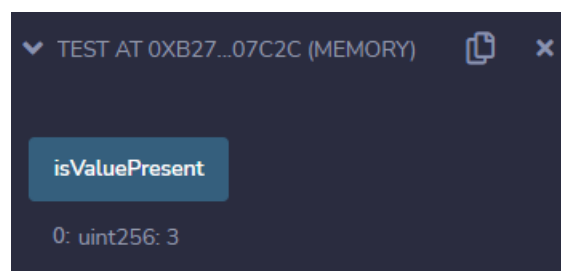
```

Output:

b. Libraries, Assembly, Events, Error handling.**Libraries****Code:**

```
pragma solidity ^0.5.0;
```

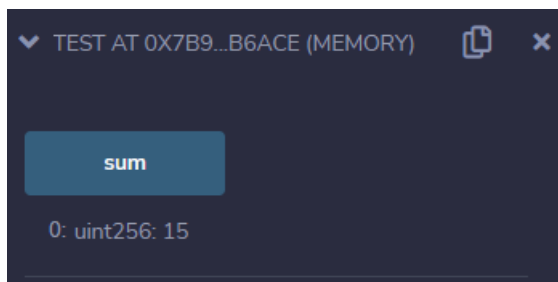
```
library Sum {  
    function sumUsingInlineAssembly(uint[] memory _data) public pure returns (uint o_sum) {  
        for (uint i = 0; i < _data.length; ++i) {  
            assembly {  
                o_sum := add(o_sum, mload(add(add(_data, 0x20), mul(i, 0x20))))  
            }  
        }  
    }  
}  
  
contract Test {  
    uint[] data;  
  
    constructor() public {  
        data.push(1);  
        data.push(2);  
        data.push(3);  
        data.push(4);  
        data.push(5);  
    }  
    function sum() external view returns(uint){  
        return Sum.sumUsingInlineAssembly(data);  
    }  
}
```

Output:

Assembly**Code:**

```
pragma solidity ^0.5.0;
```

```
library Sum {  
    function sumUsingInlineAssembly(uint[] memory _data) public pure returns (uint o_sum) {  
        for (uint i = 0; i < _data.length; ++i) {  
            assembly {  
                o_sum := add(o_sum, mload(add(add(_data, 0x20), mul(i, 0x20))))  
            }  
        }  
    }  
}  
  
contract Test {  
    uint[] data;  
  
    constructor() public {  
        data.push(1);  
        data.push(2);  
        data.push(3);  
        data.push(4);  
        data.push(5);  
    }  
    function sum() external view returns(uint){  
        return Sum.sumUsingInlineAssembly(data);  
    }  
}
```

Output:

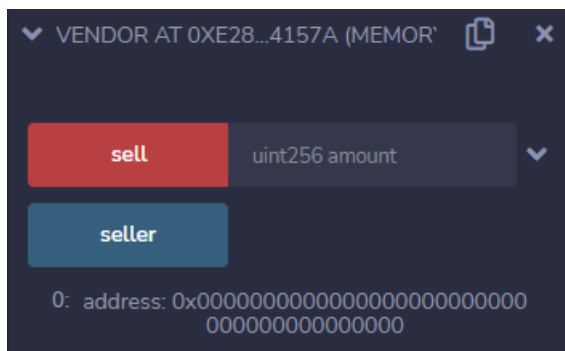
Error Handling

Code:

```
pragma solidity ^0.5.0;

contract Vendor {
    address public seller;
    modifier onlySeller() {
        require(
            msg.sender == seller,
            "Only seller can call this."
        );
    }
    function sell(uint amount) public payable onlySeller {
        if (amount > msg.value / 2 ether)
            revert("Not enough Ether provided.");
        // Perform the sell operation.
    }
}
```

Output:



Writeup:

This image shows a single sheet of white paper with horizontal ruling lines. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.

Practical No: 05

Aim Install hyperledger fabric and composer. Deploy and execute the application.

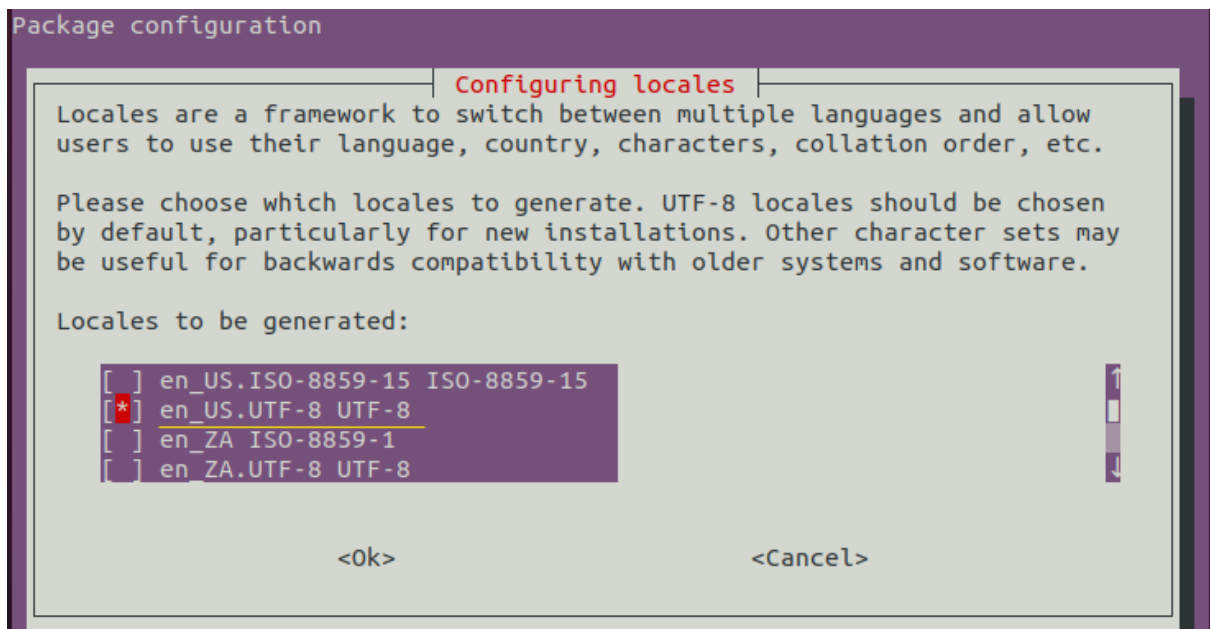
Create VM

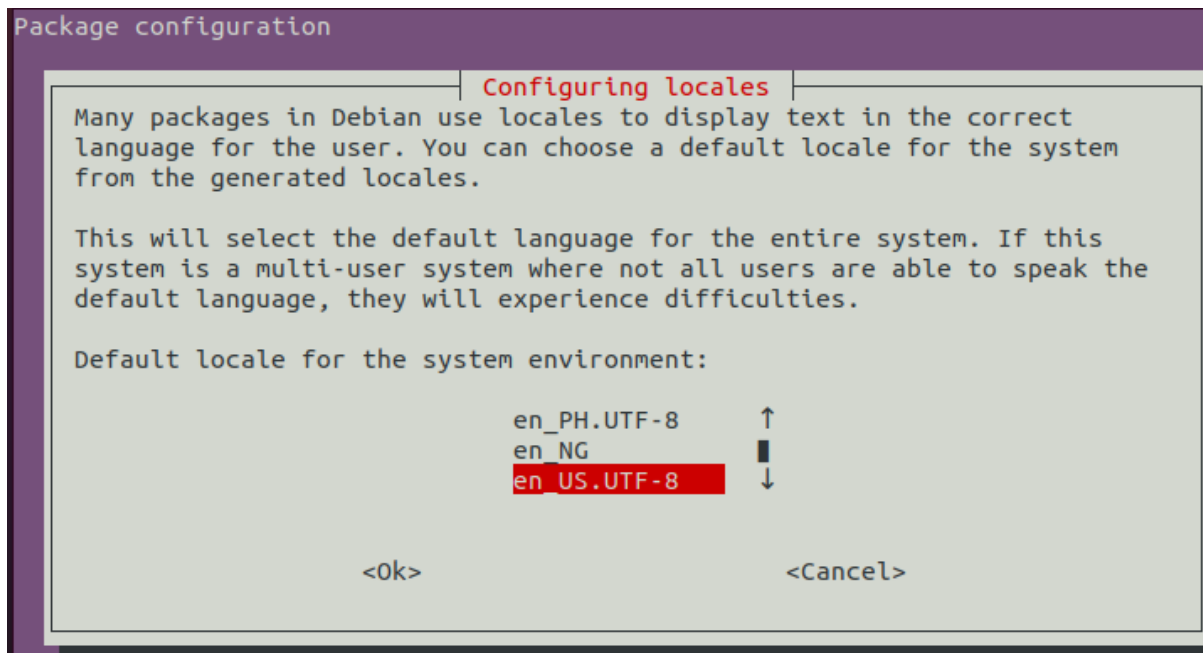
1. Download VMware Player.
2. Download Ubuntu ISO
3. Install vmware player
4. Create VM of Ubuntu using vmware player

\$ sudo dpkg-reconfigure locales // choose en_US.UTF-8 if in doubt

```
student@ubuntu:~/Desktop$ sudo dpkg-reconfigure locales
[sudo] password for student:
Generating locales (this might take a while)...
  en_AG.UTF-8... done
  en_AU.UTF-8...
```

\$ sudo apt-get update





\$ sudo apt-get upgrade

```
student@ubuntu:~/Desktop$ sudo apt-get update
Get:1 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Hit:2 http://us.archive.ubuntu.com/ubuntu focal InRelease
Get:3 http://us.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:4 http://us.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Fetched 336 kB in 2s (139 kB/s)
Reading package lists... Done
student@ubuntu:~/Desktop$
```

Install pre-requisites

\$ sudo apt-get install curl git docker.io docker-compose golang nodejs npm

```
student@ubuntu:~/Desktop$ sudo apt-get install curl git docker.io docker-compose
golang nodejs npm
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu bridge-utils
  build-essential containerd cpp-9 dpkg-dev fakeroot g++ g++-9 gcc gcc-9
  gcc-9-base git-man golang-1.13 golang-1.13-doc golang-1.13-go
  golang-1.13-race-detector-runtime golang-1.13-src golang-doc golang-go
  golang-race-detector-runtime golang-src gyp javascript-common
```

Type Y for yes

```
Do you want to continue? [Y/n] y
Get:1 http://us.archive.ubuntu.com/ubuntu focal-updates/universe amd64 libpython2
.7-minimal amd64 2.7.18-1~20.04.1 [335 kB]
Get:2 http://us.archive.ubuntu.com/ubuntu focal-updates/universe amd64 python2.7-
minimal amd64 2.7.18-1~20.04.1 [1,285 kB]
Get:3 http://us.archive.ubuntu.com/ubuntu focal/universe amd64 python2-minimal am
d64 2.7.17-2ubuntu4 [27.5 kB]
Get:4 http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 libc6-dbg amd6
4 2.31-0ubuntu9.9 [20.0 MB]
5% [4 libc6-dbg 5,078 kB/20.0 MB 25%]
```

Install Docker

```
$ sudo usermod -a -G docker $USER
```

```
$ sudo systemctl start docker
```

```
$ sudo systemctl enable docker
```

```
$ sudo chmod 666 /var/run/docker.sock
```

```
student@ubuntu:~/Desktop$ sudo usermod -a -G docker $USER
[sudo] password for student:
student@ubuntu:~/Desktop$ sudo systemctl start docker
student@ubuntu:~/Desktop$ sudo systemctl enable docker
student@ubuntu:~/Desktop$ sudo chmod 666 /var/run/docker.sock
student@ubuntu:~/Desktop$
```

Install Hyperledger Fabric

1. Check the latest version of fabric repository
2. Install Fabric

```
$ curl -sSL http://bit.ly/2ysb0FE | bash -s 1.4.0
```

```
student@ubuntu:~/Desktop$ curl -sSL http://bit.ly/2ysb0FE | bash -s 1.4.0

Clone hyperledger/fabric-samples repo

==> Cloning hyperledger/fabric-samples repo
Cloning into 'fabric-samples'...
remote: Enumerating objects: 10222, done.
Receiving objects: 30% (3067/10222), 1.67 MiB | 402.00 KiB/s
```


3. Check if fabric is installed, you should see big "END" once done

```
$ cd fabric-samples/first-network
```

```
$ ./byfn.sh generate
```

```
student@ubuntu:~/Desktop$ cd fabric-samples/first-network
student@ubuntu:~/Desktop/fabric-samples/first-network$ ./byfn.sh generate
Generating certs and genesis block for channel 'mychannel' with CLI timeout of '10' seconds and CLI delay of '3' seconds
Continue? [Y/n] y
proceeding ...
/home/student/Desktop/fabric-samples/first-network/./bin/cryptogen

#####
##### Generate certificates using cryptogen tool #####
#####
+ cryptogen generate --config=./crypto-config.yaml
```

```
$ ./byfn.sh up
```

```
student@ubuntu:~/Desktop/fabric-samples/first-network$ ./byfn.sh up
Starting for channel 'mychannel' with CLI timeout of '10' seconds and CLI delay of '3' seconds
Continue? [Y/n] y
proceeding ...
```

4. Check if fabric docker is running smoothly

```
$ docker ps -a
```

```
student@ubuntu:~/Desktop/fabric-samples/first-network$ docker ps -a
```

| CONTAINER ID | IMAGE | PORTS | COMMAND | CREATED | STATUS | NAME |
|--------------|-----------------------------------|---|-------------------|---------------|---------------------------|---------------------|
| a9e202ca7c49 | hyperledger/fabric-tools:latest | | "/bin/bash" | 2 minutes ago | Up 2 minutes | cli |
| 54fd7c6969af | hyperledger/fabric-orderer:latest | 0.0.0.0:7050->7050/tcp, :::7050->7050/tcp | "orderer" | 3 minutes ago | Up 2 minutes | orderer.example.com |
| 3c57c8c912e0 | hyperledger/fabric-peer:latest | | "peer node start" | 3 minutes ago | Exited (2) 49 seconds ago | peer |
| becc638f5a5f | hyperledger/fabric-peer:latest | | "peer node start" | 3 minutes ago | Exited (2) 47 seconds ago | peer |
| 7f026872358a | hyperledger/fabric-peer:latest | | "peer node start" | 3 minutes ago | Exited (2) 48 seconds ago | peer |
| bb783f92ffb6 | hyperledger/fabric-peer:latest | | "peer node start" | 3 minutes ago | Exited (2) 50 seconds ago | peer |

5. Stop the network

```
$ ./byfn.sh down
```

```
student@ubuntu:~/Desktop/fabric-samples/first-network$ ./byfn.sh down
Stopping for channel 'mychannel' with CLI timeout of '10' seconds and CLI delay o
f '3' seconds
Continue? [Y/n] y
proceeding ...
Stopping cli ... done
Stopping orderer.example.com ... done
Removing cli ... done
Removing orderer.example.com ... done
Removing peer1.org2.example.com ... done
Removing peer0.org2.example.com ... done
Removing peer1.org1.example.com ... done
Removing peer0.org1.example.com ... done
Removing network net_byfn
Removing volume net_orderer.example.com
Removing volume net_peer0.org1.example.com
Removing volume net_peer1.org1.example.com
Removing volume net_peer0.org2.example.com
Removing volume net_peer1.org2.example.com
Removing volume net_peer0.org3.example.com
```

Install Composer

1. Create new user, when asked about the full name, use something different than the full name used of the main user, to avoid confusion next time you are logging on.

```
$ sudo adduser playground
```

```
student@ubuntu:~/Desktop/fabric-samples/first-network$ sudo adduser playground
[sudo] password for student:
Adding user `playground' ...
Adding new group `playground' (1002) ...
Adding new user `playground' (1002) with group `playground' ...
Creating home directory `/home/playground' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for playground
Enter the new value, or press ENTER for the default
    Full Name []: user
    Room Number []: user
    Work Phone []: 2865302263
    Home Phone []: 2284550367
    Other []: 17454007647
Is the information correct? [Y/n] y
student@ubuntu:~/Desktop/fabric-samples/first-network$
```

2. Set permission for the new user

```
$ sudo usermod -aG sudo playground
```

3. Login as the new user

```
$ su - playground
```

```
student@ubuntu:~/Desktop/fabric-samples/first-network$ sudo usermod -aG sudo playground
[sudo] password for student:
student@ubuntu:~/Desktop/fabric-samples/first-network$ su - playground
Password:
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

playground@ubuntu:~$
```

4. Install the prerequisites by getting and running the script from github. It will ask for the password of “playground” account to proceed.

```
$ curl -O https://hyperledger.github.io/composer/latest/prereqs-ubuntu.sh
```

```
$ chmod u+x prereqs-ubuntu.sh
```

```
playground@ubuntu:~$ curl -O https://hyperledger.github.io/composer/latest/prereqs-ubuntu.sh
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total     Spent    Left     Speed
100 4001  100 4001    0     0  6713      0 --:--:-- --:--:-- --:--:-- 6701
playground@ubuntu:~$ chmod u+x prereqs-ubuntu.sh
```

```
$ ./prereqs-ubuntu.sh
```

5. Logout and login with the new user to get things activated properly

```
$ exit
```

```
$ su - playground
```

```
playground@ubuntu:~$ ./prereqs-ubuntu.sh
Error: Ubuntu focal is not supported
playground@ubuntu:~$ exit
logout
student@ubuntu:~/Desktop/fabric-samples/first-network$ su - playground
Password:
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

playground@ubuntu:~$
```

6. Install components needed for running Hyperledger Fabric

```
$ curl -sSL http://bit.ly/2ysb0FE | bash -s 1.4.0
```

```
playground@ubuntu:~$ curl -sSL http://bit.ly/2ysb0FE | bash -s 1.4.0
Clone hyperledger/fabric-samples repo
====> Cloning hyperledger/fabric-samples repo
Cloning into 'fabric-samples'...
r ShowApplications ing objects: 10222, done.
Receiving objects: 19% (1943/10222), 500.01 KiB | 323.00 KiB/s
```

7. Install components needed for running Hyperledger Composer

```
$ npm install -g composer-cli composer-rest-server generator-hyperledger-composer
yo composer-playground
```

8. Start Composer

```
$ composer-playground
```

9. Open your browser and check it:

```
http://localhost:8080
```

Writeup:

This image shows a single sheet of white paper with horizontal ruling lines. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.

Practical No: 06**Aim: Create your own blockchain and demonstrate its use.****Code:**

```
import hashlib

import time

class Block(object):
    def __init__(self, index, proof_number, previous_hash, data, timestamp=None):

        self.index = index

        self.proof_number = proof_number

        self.previous_hash = previous_hash

        self.data = data

        self.timestamp = timestamp or time.time()

    @property

    def compute_hash(self):

        string_block = "{}{}{}{}{}{}".format(self.index, self.proof_number, self.previous_hash,
        self.data, self.timestamp)

        return hashlib.sha256(string_block.encode()).hexdigest()

    def __repr__(self):

        return "{} - {} - {} - {} - {}".format(self.index, self.proof_number, self.previous_hash,
        self.data, self.timestamp)

class Blockchain(object):

    def __init__(self):

        self.chain = []

        self.current_data = []
```

```
self.nodes = set()

self.build_genesis()

def build_genesis(self):

    self.build_block(proof_number=0, previous_hash=0)

def build_block(self, proof_number, previous_hash):

    block = Block(

        index=len(self.chain),

        proof_number=proof_number,

        previous_hash=previous_hash,

        data=self.current_data

    )

    self.current_data = []

    self.chain.append(block)

    return block

    @staticmethod

    def confirm_validity(block, previous_block):

        if previous_block.index + 1 != block.index:

            return False

        elif previous_block.compute_hash != block.previous_hash:

            return False

        elif block.timestamp <= previous_block.timestamp:

            return False
```

```
        return True

    def get_data(self, sender, receiver, amount):

        self.current_data.append({

            'sender': sender,

            'receiver': receiver,

            'amount': amount

        })

        return True

    @staticmethod

    def proof_of_work(last_proof):

        pass

    @property

    def latest_block(self):

        return self.chain[-1]

    def chain_validity(self):

        pass

    def block_mining(self, details_miner):

        self.get_data(

            sender="0", #it implies that this node has created a new block

            receiver=details_miner,

            quantity=1, #creating a new block (or identifying the proof number) is awarded with 1

        )
```



```
        last_block = self.latest_block

        last_proof_number = last_block.proof_number

        proof_number = self.proof_of_work(last_proof_number)

        last_hash = last_block.compute_hash

        block = self.build_block(proof_number, last_hash)

        return vars(block)

def create_node(self, address):

    self.nodes.add(address)

    return True

    @staticmethod

def get_block_object(block_data):

    return Block(

        block_data['index'],

        block_data['proof_number'],

        block_data['previous_hash'],

        block_data['data'],

        timestamp=block_data['timestamp']

    )

blockchain = Blockchain()

print("GET READY MINING ABOUT TO START")

print(blockchain.chain)

last_block = blockchain.latest_block
```

```
last_proof_number = last_block.proof_number

proof_number = blockchain.proof_of_work(last_proof_number)

blockchain.get_data(

    sender="0", #this means that this node has constructed another block

    receiver="Sana",

    amount=1, #building a new block (or figuring out the proof number) is awarded with 1

)

last_hash = last_block.compute_hash

block = blockchain.build_block(proof_number, last_hash)

print("WOW, MINING HAS BEEN SUCCESSFUL!")

print(blockchain.chain)
```

Output:

```
>>> = RESTART: C:/Users/Sana Khan/AppData/Local/Programs/Python/Python310/Blockchain
/blockchain.py
GET READY MINING ABOUT TO START
[0 - 0 - 0 - [] - 1652524638.0712283]
WOW, MINING HAS BEEN SUCCESSFUL!
[0 - 0 - 0 - [] - 1652524638.0712283, 1 - None - bcd916c6d08bf57103648c4197ebb44
473da2b02c60717b8bab7accf4b97a4fa - [{'sender': '0', 'receiver': 'Sana', 'amount
': 1}]] - 1652524638.0901768]
>>>
```

Practical No. 07

Writeup:

[illegible]

Practical No: 07**Aim: Demonstrate the use of Bitcoin Core API.****Code:**

```
#pip install bitcoinlib
#Use Python 3.9.12

from bitcoinlib.wallets import Wallet
w = Wallet.create('Wallet3')
key1 = w.get_key()
print(key1.address)

# Send a small transaction to your wallet and use the scan() method to update transactions
and UTXO's
w.scan()
print(w.info())
```

Output:

```
===== RESTART: C:/Users/RDNC/AppData/Local/Programs/Python/Python39/bitcoincoreapi.py =====
1K4gyCkQNPjeNv9UJk9ZkFmZ5yaY5l7qDL
=== WALLET ===
ID                               1
Name                             Wallet3
Owner
Scheme                           bip32
Multisig                          False
Witness type                       legacy
Main network                       bitcoin
Latest update                     2022-05-21 12:04:00.155937

= Wallet Master Key =
ID                               1
Private                          True
Depth                            0

- NETWORK: bitcoin -
-- Keys
  6 m/44'/0'/0'/0/0             1K4gyCkQNPjeNv9UJk9ZkFmZ5yaY5l7qDL          address index 0          0.00000000 B
  7 m/44'/0'/0'/0/1             1FKpqr5JLBpsFTj5NUdq9xvCUUEcAqQPCk          address index 1          0.00000000 B
  8 m/44'/0'/0'/0/2             1R7CpuLSv6CSwdNXoJVHgQDVwNeuDGsXE          address index 2          0.00000000 B
  9 m/44'/0'/0'/0/3             1L7yAqGnQZzH5Rd75ZttsNKQ52ewUSRz2L          address index 3          0.00000000 B
 10 m/44'/0'/0'/0/4             19dn2JAarrktgsx9wZxfq9eX3KRAXmS2ff3          address index 4          0.00000000 B
 12 m/44'/0'/0'/1/0             124HAnsolkVUUsUrYBtLJSFKSUCDuKAnMjM          address index 0          0.00000000 B
 13 m/44'/0'/0'/1/1             1JExXuETZcGKfwGw6nYGxUPL13cMB65DTx          address index 1          0.00000000 B
 14 m/44'/0'/0'/1/2             1Bp84cUlzxtffkzSMtmArPhXaifych6uqD          address index 2          0.00000000 B
 15 m/44'/0'/0'/1/3             17BtY4FnpYFv6YmjQGHgyb4nt8nfiTg6Fb          address index 3          0.00000000 B
 16 m/44'/0'/0'/1/4             17FSCz4iptwjSeFVTASNSD9S4mpyi1BRFY          address index 4          0.00000000 B

-- Transactions Account 0 (0)

= Balance Totals (includes unconfirmed) =

None
>>>
```

Writeup:

This image shows a single sheet of white paper with horizontal ruling lines. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.

Practical No: 08**Aim: Building Dapps with Moralis & MetaMask****Code:****index.html:**

```
<html>
<head>

  <!-- Moralis SDK code -->
  <script src="https://cdn.jsdelivr.net/npm/web3@latest/dist/web3.min.js"></script>
  <script src="https://unpkg.com/moralis/dist/moralis.js"></script>
</head>
<body>
  <h1>Moralis Gas Stats</h1>

  <button id="btn-login">Moralis Login</button>
  <button id="btn-logout">Logout</button>
  <button id="btn-get-stats">Refresh Stats</button>

  <!-- stats will go here -->
  <ul id="gas-stats"></ul>

  <script>
    // connect to Moralis server
    const serverUrl = "https://gjjgzccgef6d.usemoralis.com:2053/server";
    const appId = "8IqxNy07hfIUf9lc3P27wEPcup6uaIj592FKIu1D";
    Moralis.start({ serverUrl, appId });

    // LOG IN WITH METAMASK
    async function login() {
      let user = Moralis.User.current();
      if (!user) {
        user = await Moralis.authenticate();
      }
      console.log("logged in user:", user);
      getStats();
    }

    // LOG OUT
    async function logOut() {
      await Moralis.User.logOut();
      console.log("logged out");
    }
  </script>
</body>
</html>
```

```
// bind button click handlers
document.getElementById("btn-login").onclick = login;
document.getElementById("btn-logout").onclick = logOut;
document.getElementById("btn-get-stats").onclick = getStats;

// refresh stats
function getStats() {
  const user = Moralis.User.current();
  if (user) {
    getUserTransactions(user);
  }
  getAverageGasPrices();
}

// HISTORICAL TRANSACTIONS
async function getUserTransactions(user) {
  // create query
  const query = new Moralis.Query("EthTransactions");
  query.equalTo("from_address", user.get("ethAddress"));

  // subscribe to query updates
  const subscription = await query.subscribe();
  handleNewTransaction(subscription);

  // run query
  const results = await query.find();
  console.log("user transactions:", results);
}

// REAL-TIME TRANSACTIONS
async function handleNewTransaction(subscription) {
  // log each new transaction
  subscription.on("create", function (data) {
    console.log("new transaction: ", data);
  });
}

// CLOUD FUNCTION
async function getAverageGasPrices() {
  const results = await Moralis.Cloud.run("getAvgGas");
  console.log("average user gas prices:", results);
  renderGasStats(results);
}
```

```

function renderGasStats(data) {
  const container = document.getElementById("gas-stats");
  container.innerHTML = data
    .map(function (row, rank) {
      return `<li>#${rank + 1}: ${Math.round(row.avgGas)} gwei</li>`;
    })
    .join("");
}

//get stats on page load
getStats();
</script>
</body>
</html>

```

Cloud Function file on Moralis Server:

```

Moralis.Cloud.define("getAvgGas", async function (request) {
  const query = new Moralis.Query("EthTransactions");
  const pipeline = [
    {
      group: {
        // group by "from_address"
        objectId: "$from_address",
        // add computed property avgGas
        // get average and convert wei to gwei
        avgGas: { $avg: { $divide: ["$gas_price", 10000000000] } },
      },
    },
    { sort: { avgGas: -1 } }, // sort by avgGas high to low
    { limit: 10 }, // only return top 10 results
  ];

  // the master key is required for aggregate queries
  const results = await query.aggregate(pipeline, { useMasterKey: true });
  return results;
});

```


Output:

Moralis Gas Stats

Moralis LoginLogoutRefresh Stats

ElementsConsoleSourcesNetworkPerformanceMemoryApplicationSecurity

topFilter

Live reload enabled.
average user gas prices: ▶ Array(0)
logged in user: ▶ ParseUser {id: 'W4WzefD2vc5KazILvzAjiwoV', _objCount: 0, className: '_User'}
average user gas prices: ▶ []
user transactions: ▶ []
average user gas prices: ▶ []
user transactions: ▶ []
logged out
logged in user: ▶ ParseUser {id: 'W4WzefD2vc5KazILvzAjiwoV', _objCount: 1, className: '_User'}
average user gas prices: ▶ []
user transactions: ▶ []
average user gas prices: ▶ []
user transactions: ▶ []
logged out

>