Practical Journal

BLOCKCHAIN

Submitted by

Name: Akash Shitut

SAP ID: 53004220033

Class: MSC IT PART II

Department of Information Technology

Shri Vile Parle Kelavani Mandal’s

Usha Pravin Gandhi College of Arts, Science & Commerce

(Affiliated to University of Mumbai)

NAAC Reaccredited ‘A+’ Grade

Vile Parle (West), Mumbai 400056

2023-24

BLOCKCHAIN Practical Journal

Academic Year: 2023-2024

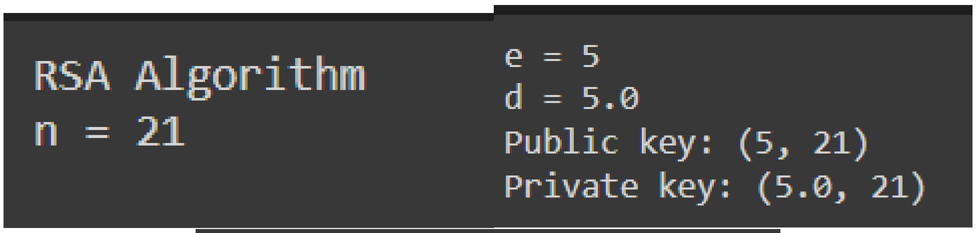
Practical 1

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

Code and Output:

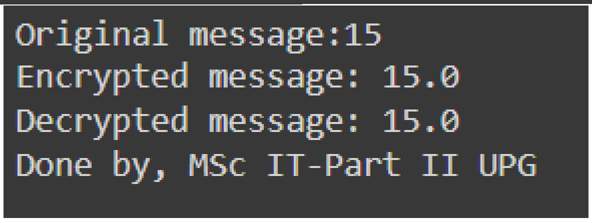
```
# pip install Crypto
# pip install pycryptodome
import math
# step 1
p = 3
q = 7
print("RSA Algorithm ")
# step 2
n = p*q
print("n =", n)
# step 3
phi = (p-1)*(q-1)
# step 4
e = 2 #change value
while(e<phi):
    if (math.gcd(e, phi) == 1):
        break
    else:
        e += 1
print("e =", e)
# step 5
k = 2
d = ((k*phi)+1)/e
print("d =", d)
print(f'Public key: {e, n}')
print(f'Private key: {d, n}')
# plain text
msg = 15
print(f'Original message:{msg}')
# encryption
C = pow(msg, e)
C = math.fmod(C, n)
print(f'Encrypted message: {C}')
# decryption
M = pow(C, d)
M = math.fmod(M, n)
print(f'Decrypted message: {M}')
print('Done by, MSc IT-Part II UPG')
```

Output:



```
RSA Algorithm
n = 21
```

```
e = 5
d = 5.0
Public key: (5, 21)
Private key: (5.0, 21)
```



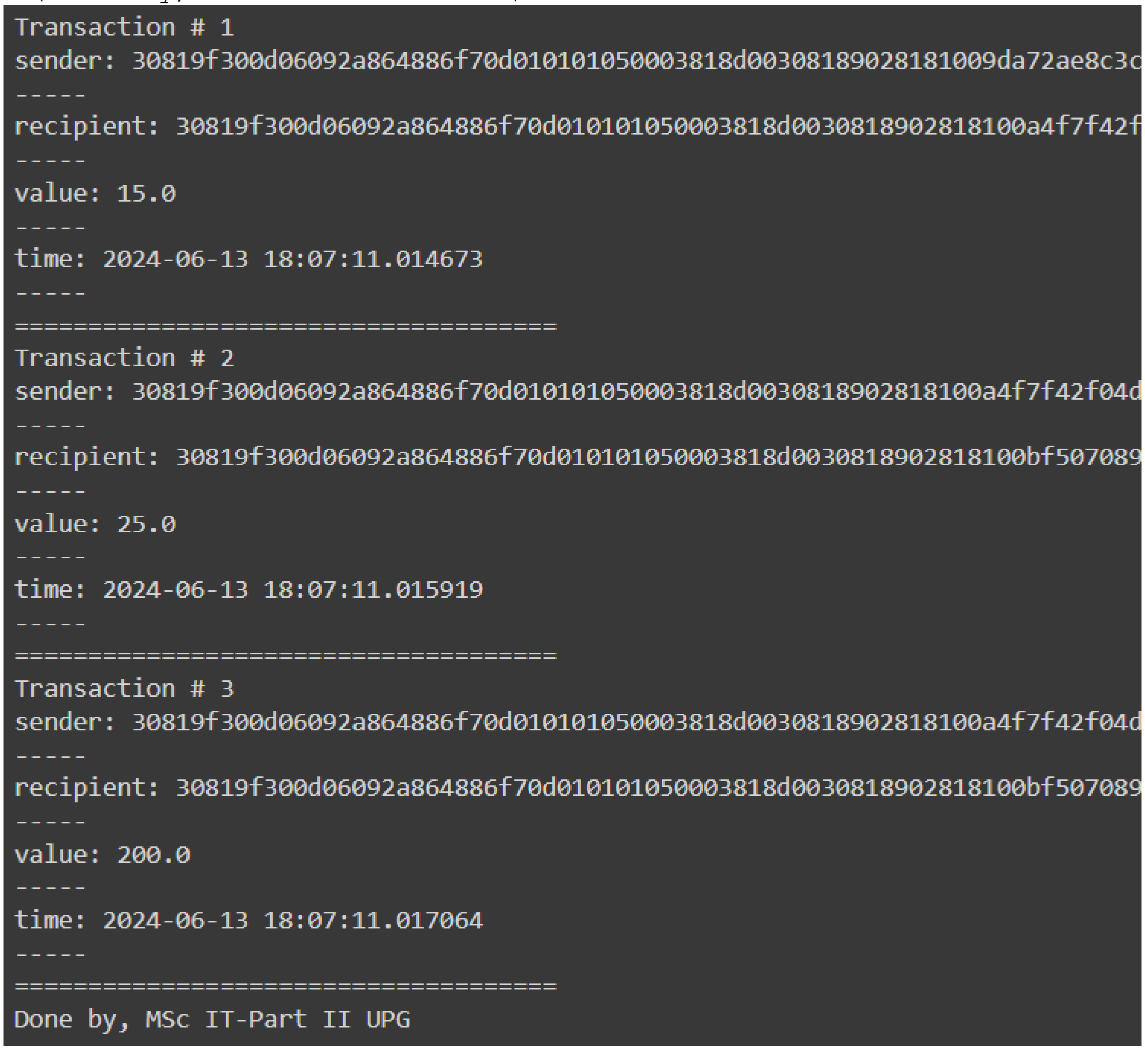
```
Original message:15
Encrypted message: 15.0
Decrypted message: 15.0
Done by, MSc IT-Part II UPG
```

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

Code and Output:

```
#Practical 1 B
# following imports are required by PKI
import hashlib
import random
import binascii
import datetime
import collections
from Crypto.PublicKey import RSA
from Crypto import Random
from Crypto.Cipher import PKCS1_v1_5
from collections import OrderedDict
import Crypto
import Crypto.Random
from Crypto.Hash import SHA
from Crypto.Signature import PKCS1_v1_5
class Client:
    def __init__(self):
     random = Random.new().read
     self._private_key = RSA.generate(1024, random)
     self._public_key = self._private_key.publickey()
     self._signer = PKCS1_v1_5.new(self._private_key)
    @property
    def identity(self):
return
binascii.hexlify(self._public_key.exportKey(format='DER')).decode('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()
Suresh = Client()

t1 = Transaction( Dinesh, Ramesh.identity, 15.0)

t1.sign_transaction()
transactions.append(t1)

t2 = Transaction( Ramesh, Suresh.identity, 25.0)
t2.sign_transaction()
transactions.append(t2)

t3 = Transaction( Ramesh, Suresh.identity,200.0)
t3.sign_transaction()
transactions.append(t3)

tn=1
for t in transactions:
  print("Transaction #",tn)
  display_transaction (t)
  tn=tn+1
  print ('=====================================')
print('Done by, MSc IT-Part II UPG')
```



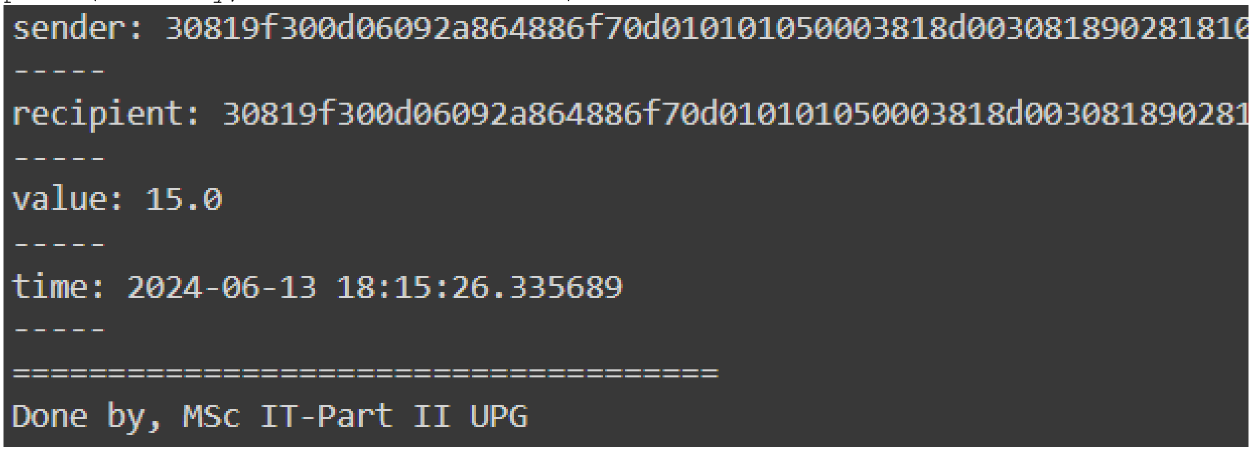
```
Transaction # 1
sender: 30819f300d06092a864886f70d010101050003818d00308189028181009da72ae8c3c
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100a4f7f42f
-----
value: 15.0
-----
time: 2024-06-13 18:07:11.014673
-----
=====================================
Transaction # 2
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100a4f7f42f04d
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100bf507089
-----
value: 25.0
-----
time: 2024-06-13 18:07:11.015919
-----
=====================================
Transaction # 3
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100a4f7f42f04d
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100bf507089
-----
value: 200.0
-----
time: 2024-06-13 18:07:11.017064
-----
=====================================
Done by, MSc IT-Part II UPG
```

C. Create multiple transactions and display them.

Code and Output:

```
#Practical 1 C
# following imports are required by PKI
import hashlib
import random
import binascii
import datetime
import collections
from Crypto.PublicKey import RSA
from Crypto import Random
from Crypto.Cipher import PKCS1_v1_5
from collections import OrderedDict
import Crypto
import Crypto.Random
from Crypto.Hash import SHA
from Crypto.Signature import PKCS1_v1_5
class Client:
    def __init__(self):
     random = Random.new().read
     self._private_key = RSA.generate(1024, random)
     self._public_key = self._private_key.publickey()
     self._signer = PKCS1_v1_5.new(self._private_key)
    @property
    def identity(self):
                                                                          return
binascii.hexlify(self._public_key.exportKey(format='DER')).decode('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 = []
```

```
Kartik = Client()
Rakesh = Client()
Suraj = Client()

t1 = Transaction( Kartik, Rakesh.identity, 15.0)

t1.sign_transaction()
display_transaction (t1)
print ('====================================')
print('Done by, MSc IT-Part II UPG')
```



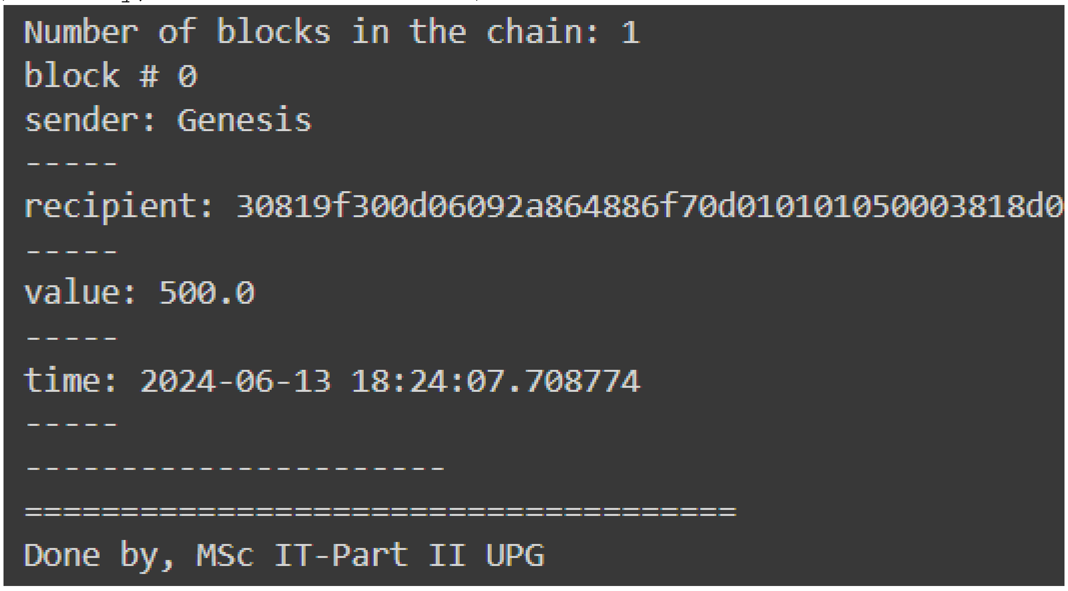
```
sender: 30819f300d06092a864886f70d010101050003818d00308189028181
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818
-----
value: 15.0
-----
time: 2024-06-13 18:15:26.335689
-----
====================================
Done by, MSc IT-Part II UPG
```

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

Code and Output:

```
#Practical 1D
# following imports are required by PKI
import hashlib
import random
import binascii
import datetime
import collections
from Crypto.PublicKey import RSA
from Crypto import Random
from Crypto.Cipher import PKCS1_v1_5
from collections import OrderedDict
import Crypto
import Crypto.Random
from Crypto.Hash import SHA
from Crypto.Signature import PKCS1_v1_5
class Client:
    def __init__(self):
     random = Random.new().read
     self._private_key = RSA.generate(1024, random)
     self._public_key = self._private_key.publickey()
     self._signer = PKCS1_v1_5.new(self._private_key)
    @property
    def identity(self):
        return
binascii.hexlify(self._public_key.exportKey(format='DER')).decode('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 ('-----')
```

```
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 ('=====================================')

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

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)
print('Done by, MSc IT-Part II UPG')
```

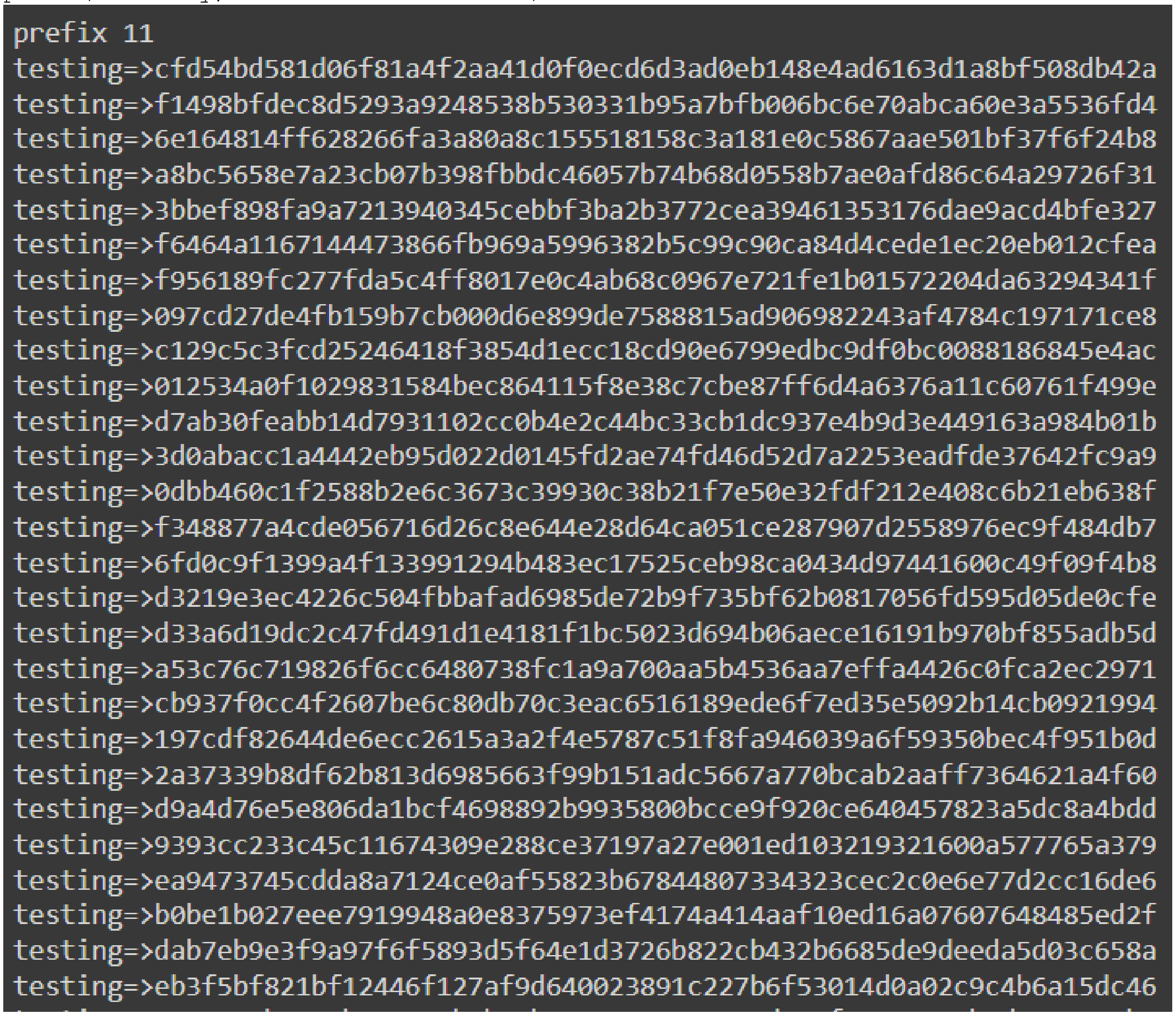


```
Number of blocks in the chain: 1
block # 0
sender: Genesis
-----
recipient: 30819f300d06092a864886f70d010101050003818d0
-----
value: 500.0
-----
time: 2024-06-13 18:24:07.708774
-----
----------------------
=====================================
Done by, MSc IT-Part II UPG
```

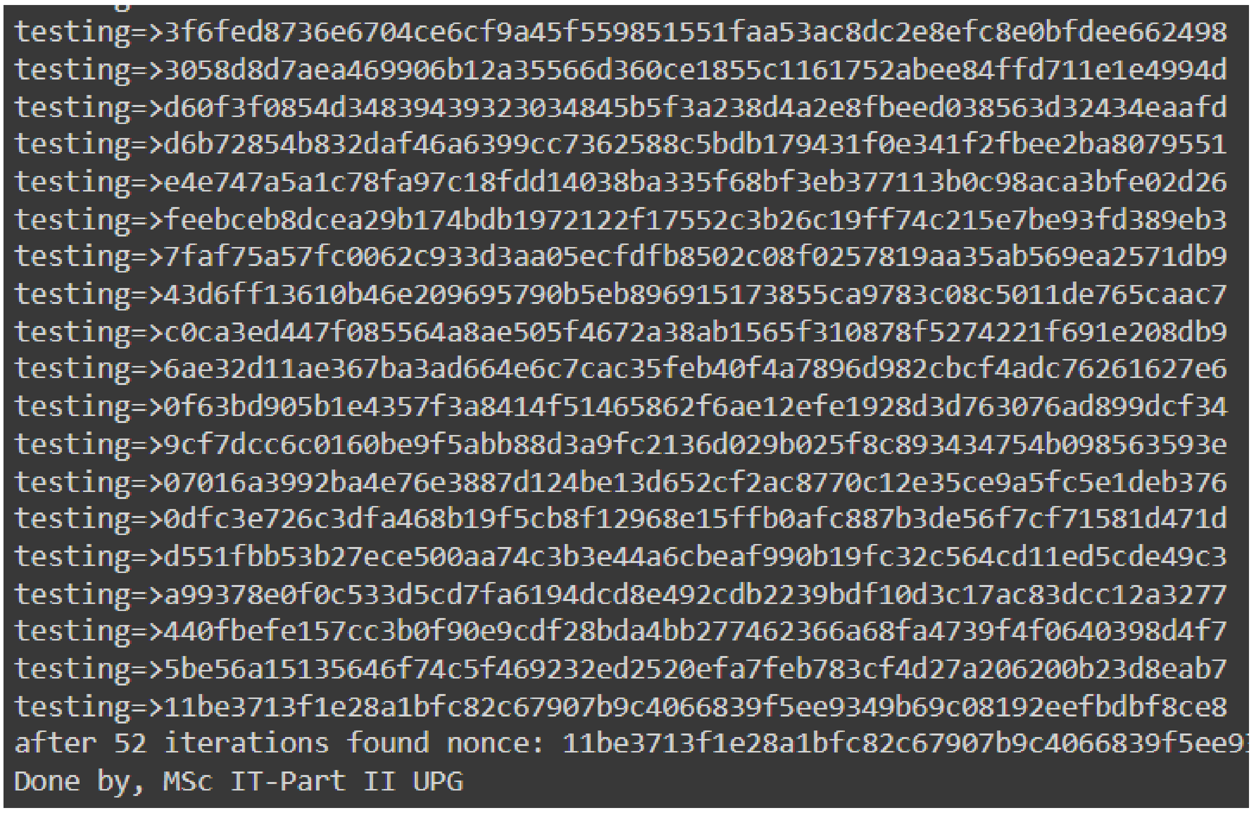
E. Create a mining function and test it.

Code and Output:

```
#Practical 1E
import hashlib
def sha256(message):
  return hashlib.sha256(message.encode('ascii')).hexdigest()
def mine(message, difficulty=1):
  assert difficulty >= 1
  #if(difficulty <1):
  # return #'1'*2=> '11'
  prefix = '1' * difficulty
  print("prefix",prefix)
  for i in range(1000):
    digest = sha256(str(hash(message)) + str(i))
    print("testing=>"+digest)
    if digest.startswith(prefix):
      print ("after " + str(i) + " iterations found nonce: "+ digest)
      return i #i= nonce value
mine ("Test message",2)
print('Done by, MSc IT-Part II UPG')
```



```
prefix 11
testing=>cfd54bd581d06f81a4f2aa41d0f0ecd6d3ad0eb148e4ad6163d1a8bf508db42a
testing=>f1498bfdec8d5293a9248538b530331b95a7bfb006bc6e70abca60e3a5536fd4
testing=>6e164814ff628266fa3a80a8c155518158c3a181e0c5867aae501bf37f6f24b8
testing=>a8bc5658e7a23cb07b398fbbdc46057b74b68d0558b7ae0afd86c64a29726f31
testing=>3bbef898fa9a7213940345cebbf3ba2b3772cea39461353176dae9acd4bfe327
testing=>f6464a1167144473866fb969a5996382b5c99c90ca84d4cede1ec20eb012cfea
testing=>f956189fc277fda5c4ff8017e0c4ab68c0967e721fe1b01572204da63294341f
testing=>097cd27de4fb159b7cb000d6e899de7588815ad906982243af4784c197171ce8
testing=>c129c5c3fcd25246418f3854d1ecc18cd90e6799edbc9df0bc0088186845e4ac
testing=>012534a0f1029831584bec864115f8e38c7cbe87ff6d4a6376a11c60761f499e
testing=>d7ab30feabb14d7931102cc0b4e2c44bc33cb1dc937e4b9d3e449163a984b01b
testing=>3d0abacc1a4442eb95d022d0145fd2ae74fd46d52d7a2253eadfde37642fc9a9
testing=>0dbb460c1f2588b2e6c3673c39930c38b21f7e50e32fdf212e408c6b21eb638f
testing=>f348877a4cde056716d26c8e644e28d64ca051ce287907d2558976ec9f484db7
testing=>6fd0c9f1399a4f133991294b483ec17525ceb98ca0434d97441600c49f09f4b8
testing=>d3219e3ec4226c504fbbafad6985de72b9f735bf62b0817056fd595d05de0cfe
testing=>d33a6d19dc2c47fd491d1e4181f1bc5023d694b06aece16191b970bf855adb5d
testing=>a53c76c719826f6cc6480738fc1a9a700aa5b4536aa7effa4426c0fca2ec2971
testing=>cb937f0cc4f2607be6c80db70c3eac6516189ede6f7ed35e5092b14cb0921994
testing=>197cdf82644de6ecc2615a3a2f4e5787c51f8fa946039a6f59350bec4f951b0d
testing=>2a37339b8df62b813d6985663f99b151adc5667a770bcab2aaff7364621a4f60
testing=>d9a4d76e5e806da1bcf4698892b9935800bcce9f920ce640457823a5dc8a4bdd
testing=>9393cc233c45c11674309e288ce37197a27e001ed103219321600a577765a379
testing=>ea9473745cdda8a7124ce0af55823b67844807334323cec2c0e6e77d2cc16de6
testing=>b0be1b027eee7919948a0e8375973ef4174a414aaf10ed16a07607648485ed2f
testing=>dab7eb9e3f9a97f6f5893d5f64e1d3726b822cb432b6685de9deeda5d03c658a
testing=>eb3f5bf821bf12446f127af9d640023891c227b6f53014d0a02c9c4b6a15dc46
```



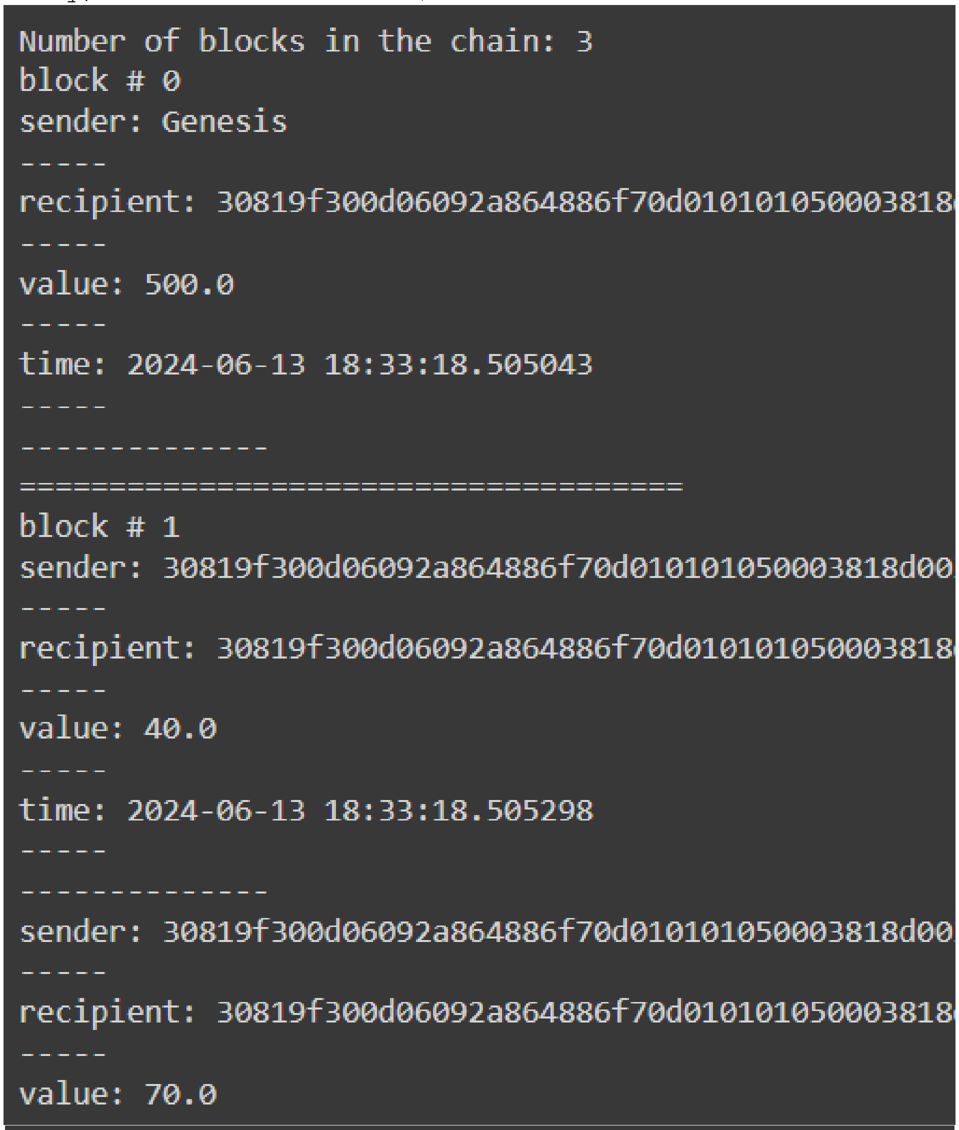
```
testing=>3f6fed8736e6704ce6cf9a45f559851551faa53ac8dc2e8efc8e0bfdee662498
testing=>3058d8d7aea469906b12a35566d360ce1855c1161752abee84ffd711e1e4994d
testing=>d60f3f0854d34839439323034845b5f3a238d4a2e8fbeed038563d32434eaafd
testing=>d6b72854b832daf46a6399cc7362588c5bdb179431f0e341f2fbee2ba8079551
testing=>e4e747a5a1c78fa97c18fdd14038ba335f68bf3eb377113b0c98aca3bfe02d26
testing=>feebceb8dcea29b174bdb1972122f17552c3b26c19ff74c215e7be93fd389eb3
testing=>7faf75a57fc0062c933d3aa05ecfdfb8502c08f0257819aa35ab569ea2571db9
testing=>43d6ff13610b46e209695790b5eb896915173855ca9783c08c5011de765caac7
testing=>c0ca3ed447f085564a8ae505f4672a38ab1565f310878f5274221f691e208db9
testing=>6ae32d11ae367ba3ad664e6c7cac35feb40f4a7896d982cbcf4adc76261627e6
testing=>0f63bd905b1e4357f3a8414f51465862f6ae12efe1928d3d763076ad899dcf34
testing=>9cf7dcc6c0160be9f5abb88d3a9fc2136d029b025f8c893434754b098563593e
testing=>07016a3992ba4e76e3887d124be13d652cf2ac8770c12e35ce9a5fc5e1deb376
testing=>0dfc3e726c3dfa468b19f5cb8f12968e15ffb0afc887b3de56f7cf71581d471d
testing=>d551fbb53b27ece500aa74c3b3e44a6cbeaf990b19fc32c564cd11ed5cde49c3
testing=>a99378e0f0c533d5cd7fa6194dcd8e492cdb2239bdf10d3c17ac83dcc12a3277
testing=>440fbefe157cc3b0f90e9cdf28bda4bb277462366a68fa4739f4f0640398d4f7
testing=>5be56a15135646f74c5f469232ed2520efa7feb783cf4d27a206200b23d8eab7
testing=>11be3713f1e28a1bfc82c67907b9c4066839f5ee9349b69c08192eefbdbf8ce8
after 52 iterations found nonce: 11be3713f1e28a1bfc82c67907b9c4066839f5ee9
Done by, MSc IT-Part II UPG
```

F. Add blocks to the miner and dump the blockchain.

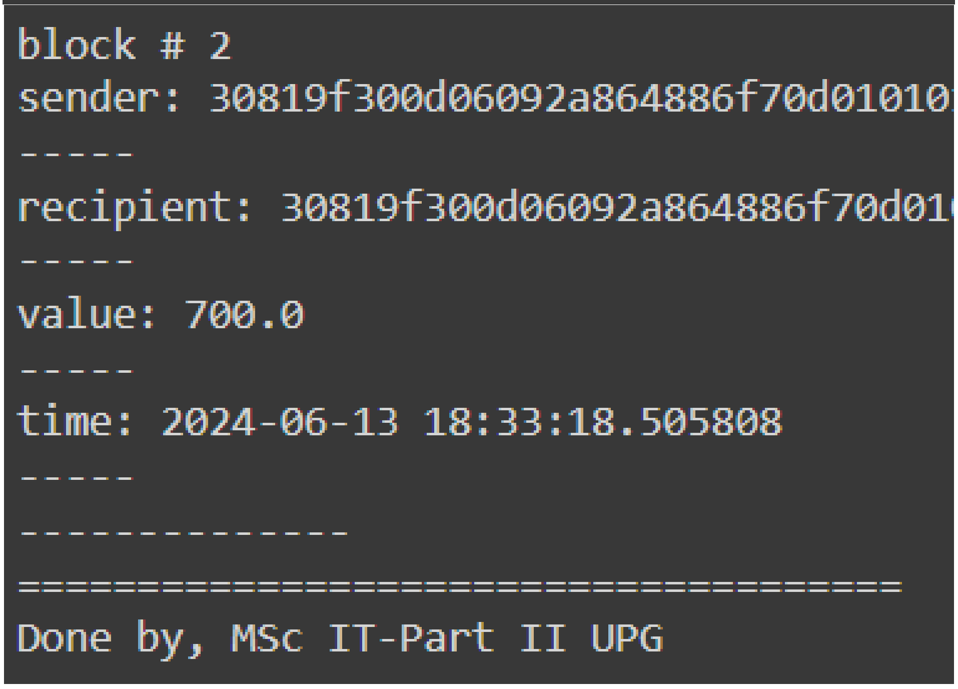
Code and Output:

```
#Practical 1 F
# following imports are required by PKI
import hashlib
import random
import binascii
import datetime
import collections
from Crypto.PublicKey import RSA
from Crypto import Random
from Crypto.Cipher import PKCS1_v1_5
from collections import OrderedDict
import Crypto
import Crypto.Random
from Crypto.Hash import SHA
from Crypto.Signature import PKCS1_v1_5
class Client:
    def __init__(self):
     random = Random.new().read
     self._private_key = RSA.generate(1024, random)
     self._public_key = self._private_key.publickey()
     self._signer = PKCS1_v1_5.new(self._private_key)
    @property
    def identity(self):
        return
binascii.hexlify(self._public_key.exportKey(format='DER')).decode('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 ('-----')
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 ('=====================================')
class Block:
    def __init__(self):
        self.verified_transactions = []
        self.previous_block_hash = ""
        self.Nonce = ""
def sha256(message):
    return hashlib.sha256(message.encode('ascii')).hexdigest()
def mine(message, difficulty=1):
    assert difficulty >= 1
 #if(difficulty <1):
 # return
 #'1'*3=> '111'
    prefix = '1' * difficulty
    for i in range(1000):
        digest = sha256(str(hash(message)) + str(i))
    if digest.startswith(prefix):
        return i #i= nonce value
A = Client()
B =Client()
C =Client()
t0 = Transaction (
        "Genesis",
        A.identity,
        500.0)
t1 = Transaction (
        A,
        B.identity,
        40.0)
t2 = Transaction (
        A,
        C.identity,
        70.0)
t3 = Transaction (
        B,
        C.identity,
        700.0)
TPCoins = []
block0 = Block()
block0.previous_block_hash = None
Nonce = None
block0.verified_transactions.append (t0)
digest = hash (block0)
last_block_hash = digest #last_block_hash it is hash of block0
TPCoins.append (block0)
block1 = Block()
block1.previous_block_hash = last_block_hash
block1.verified_transactions.append (t1)
block1.verified_transactions.append (t2)
block1.Nonce=mine (block1, 2)
digest = hash (block1)
last_block_hash = digest
TPCoins.append (block1)
block2 = Block()
block2.previous_block_hash = last_block_hash
block2.verified_transactions.append (t3)
Nonce = mine (block2, 2)
block2.Nonce=mine (block2, 2)
digest = hash (block2)
last_block_hash = digest
```

```
TPCoins.append (block2)
dump_blockchain(TPCoins)
print('Done by, MSc IT-Part II UPG')
```



```
Number of blocks in the chain: 3
block # 0
sender: Genesis
-----
recipient: 30819f300d06092a864886f70d01010105000381
-----
value: 500.0
-----
time: 2024-06-13 18:33:18.505043
-----
--------------
=====================================
block # 1
sender: 30819f300d06092a864886f70d010101050003818d00
-----
recipient: 30819f300d06092a864886f70d01010105000381
-----
value: 40.0
-----
time: 2024-06-13 18:33:18.505298
-----
--------------
sender: 30819f300d06092a864886f70d010101050003818d00
-----
recipient: 30819f300d06092a864886f70d01010105000381
-----
value: 70.0
```



```
block # 2
sender: 30819f300d06092a864886f70d01010
-----
recipient: 30819f300d06092a864886f70d01
-----
value: 700.0
-----
time: 2024-06-13 18:33:18.505808
-----
--------------
========================================
Done by, MSc IT-Part II UPG
```

Practical 2

A. Variable, Operators, Loops, Decision Making, Strings, Arrays, Enums, Structs, Mappings.

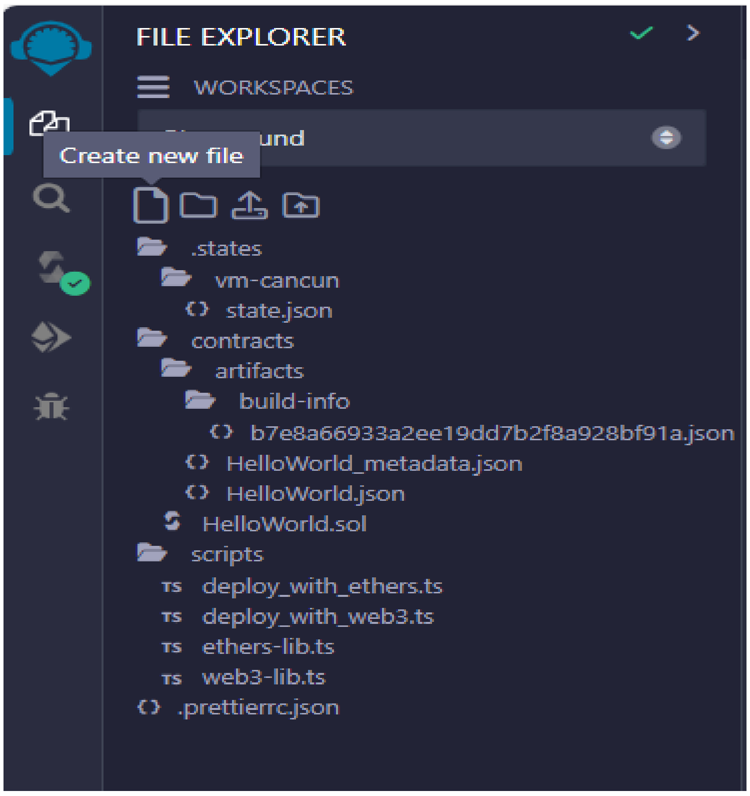
Code and Output:

i. Variables

1. State Variable

Step 1: Open https://remix.ethereum.org/ website

Step 2: Create a new file and name it.

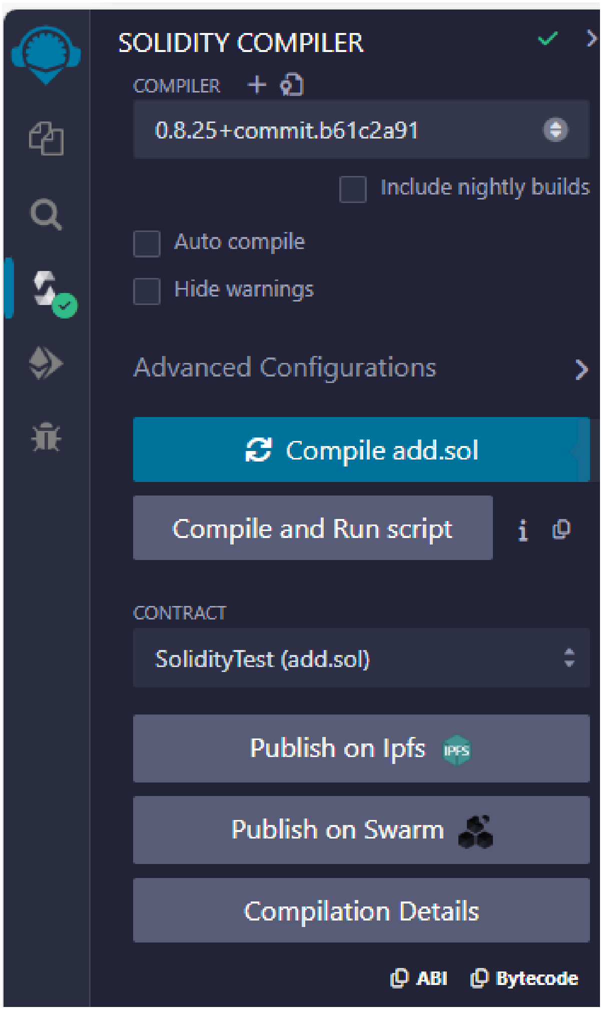


```
.states
  vm-cancun
    state.json
contracts
  artifacts
    build-info
      b7e8a66933a2ee19dd7b2f8a928bf91a.json
    HelloWorld_metadata.json
    HelloWorld.json
  HelloWorld.sol
scripts
  deploy_with_ethers.ts
  deploy_with_web3.ts
  ethers-lib.ts
  web3-lib.ts
.prettierrc.json
```

Step 3: Write this program in new file

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.6.12 <0.9.0;
contract SolidityTest{uint storedData; // State variable
constructor()  {
storedData = 10;
}
function getResult() public pure returns(uint){
uint a = 1; // local variable
uint b = 5;
uint result = a + b;
return result; //access the state variable
}
}
```

Step 4: Compile contract



SOLIDITY COMPILER

COMPILER

0.8.25+commit.b61c2a91

Include nightly builds

Auto compile

Hide warnings

Advanced Configurations

Compile add.sol

Compile and Run script

CONTRACT

SolidityTest (add.sol)

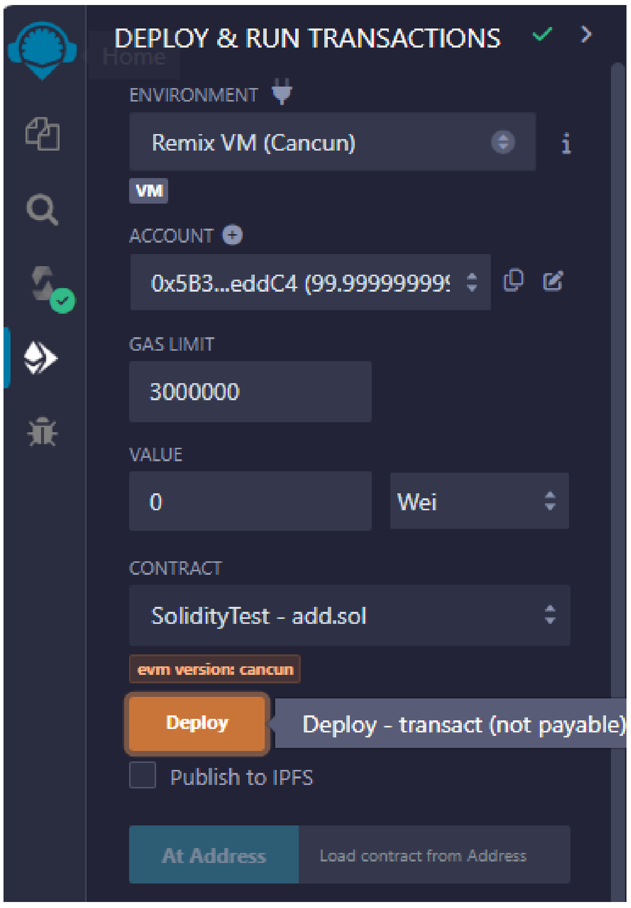
Publish on Ipfs

Publish on Swarm

Compilation Details

ABI Bytecode

Step 5: Deploy contract



DEPLOY & RUN TRANSACTIONS

ENVIRONMENT

Remix VM (Cancun)

VM

ACCOUNT

0x5B3...eddC4 (99.99999999

GAS LIMIT

3000000

VALUE

0 Wei

CONTRACT

SolidityTest - add.sol

evm version: cancun

Deploy

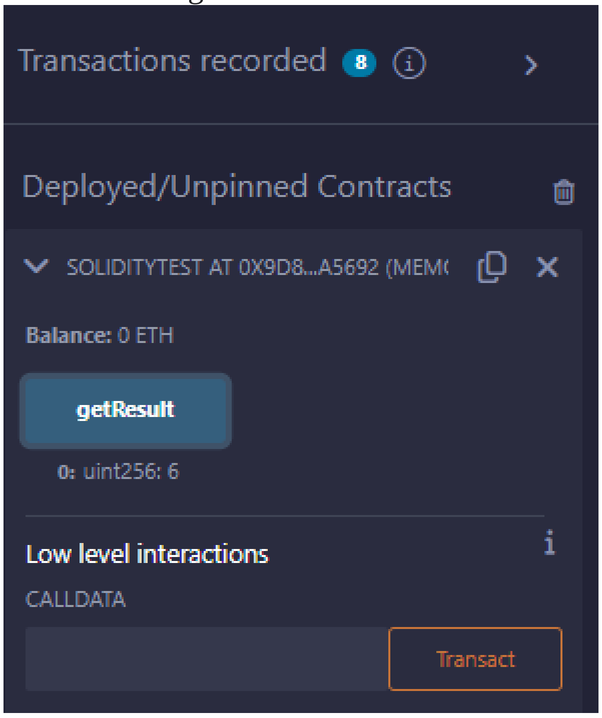
Deploy - transact (not payable)

Publish to IPFS

At Address

Load contract from Address

Step 6: Select contract and click button getResult



Transactions recorded 8

Deployed/Unpinned Contracts

SOLIDITYTEST AT 0X9D8...A5692 (MEM(

Balance: 0 ETH

0: uint256: 6

Low level interactions

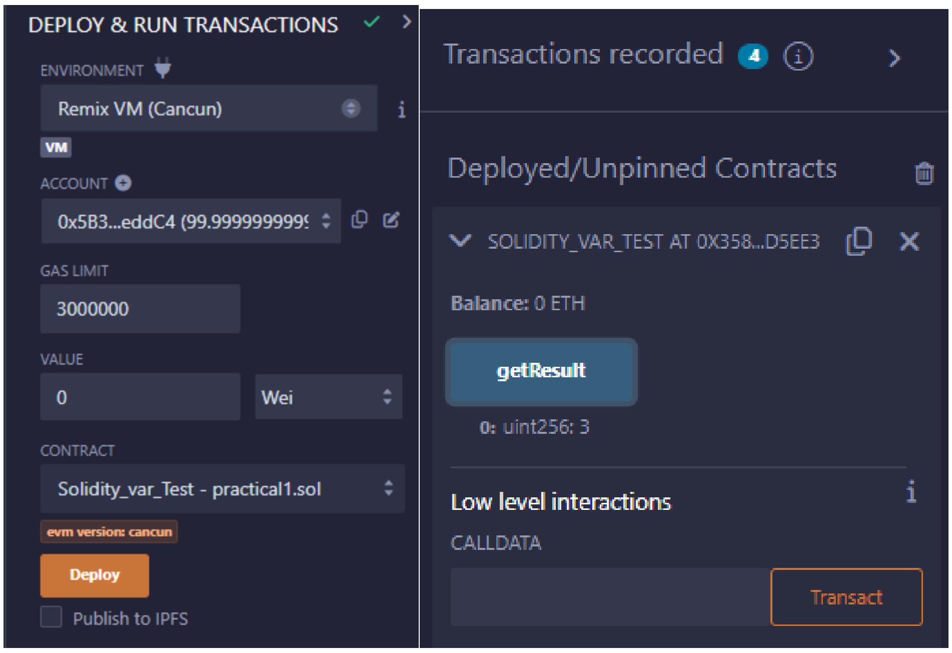
CALLDATA

2. **Local Variable**

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.6.12 <0.9.0;
// Creating a contract
contract Solidity_var_Test {
// Defining function to show the declaration and
// scope of local variables
function getResult() public pure returns(uint){
     // Initializing local variables
     uint local_var1 = 1;
     uint local_var2 = 2;
     uint result = local_var1 + local_var2;
     // Access the local variable
     return result;
}
}
```

Output:



DEPLOY & RUN TRANSACTIONS

ENVIRONMENT: Remix VM (Cancun)

VM

ACCOUNT: 0x5B3...eddC4 (99.99999999!

GAS LIMIT: 3000000

VALUE: 0 Wei

CONTRACT: Solidity_var_Test - practical1.sol

evm version: cancun

Transactions recorded 4

Deployed/Unpinned Contracts

SOLIDITY_VAR_TEST AT 0X358...D5EE3

Balance: 0 ETH

0: uint256: 3

Low level interactions

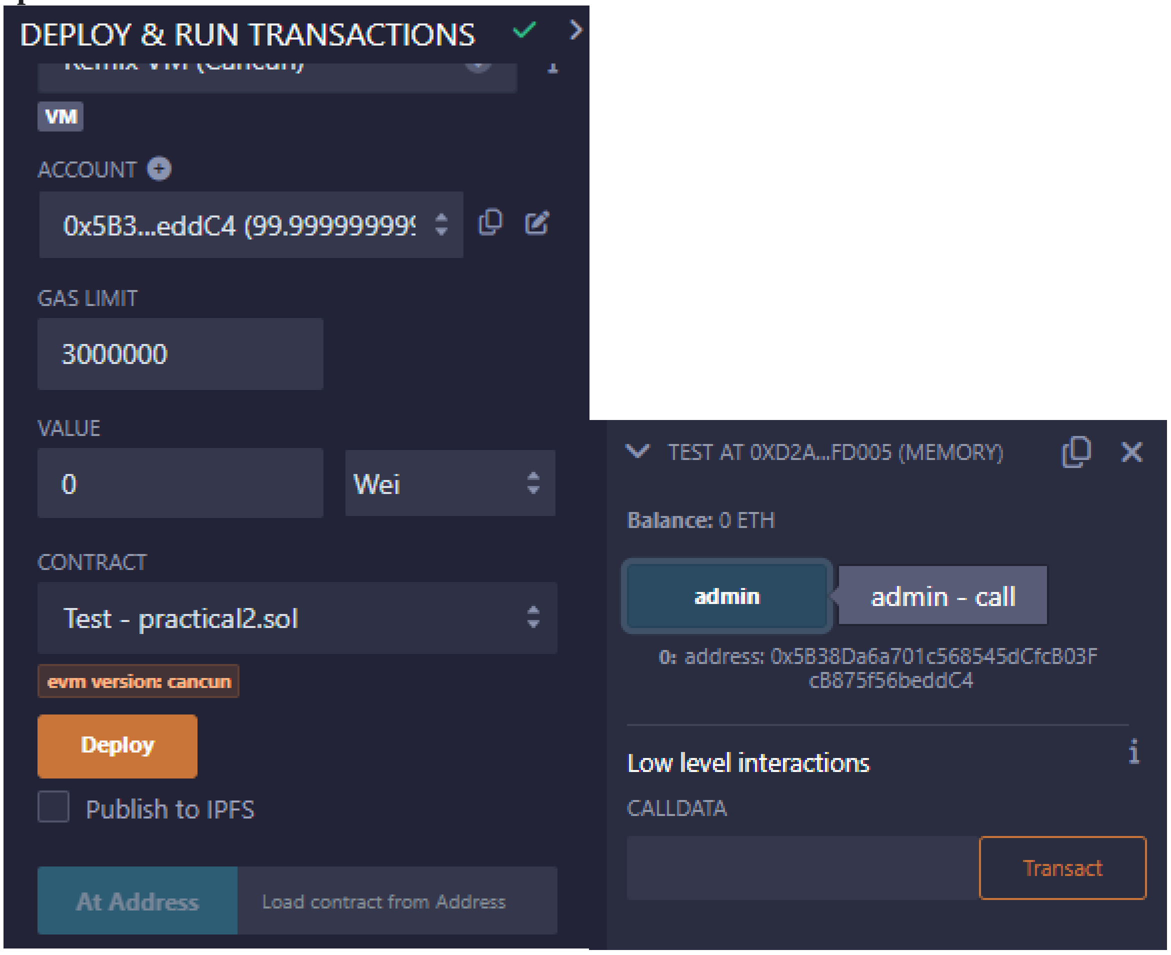
CALLDATA

3. Global Variable

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.6.12 <0.9.0;
// Creating a contract
contract Test {
    // Defining a variable
    address public admin;

    // Creating a constructor to use Global variable
    constructor() {
        admin = msg.sender;
    }
}
```

Output:



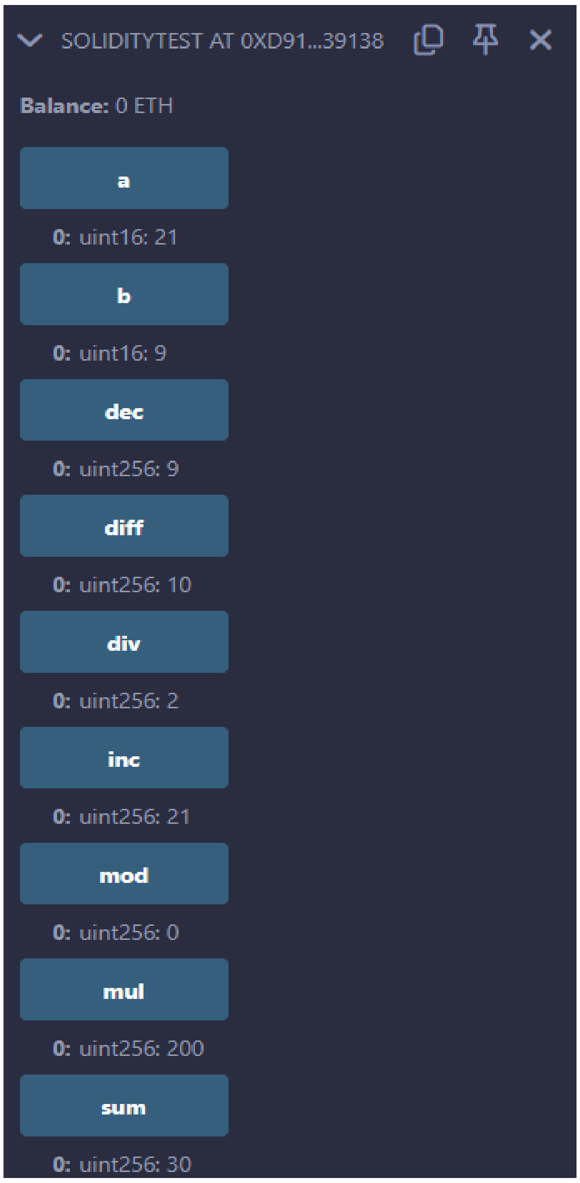
```
DEPLOY & RUN TRANSACTIONS
VM
ACCOUNT
0x5B3...eddC4 (99.999999999
GAS LIMIT
3000000
VALUE
0    Wei
CONTRACT
Test - practical2.sol
evm version: cancun
Deploy
Publish to IPFS
At Address    Load contract from Address

TEST AT 0XD2A...FD005 (MEMORY)
Balance: 0 ETH
admin    admin - call
0: address: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
Low level interactions
CALLDATA
Transact
```

ii. Operators

1. Arithmetic Operator

```
// SPDX-License-Identifier: MIT
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 variabl decrement value
uint public dec = --b;
// Initializing a variable with increment value
uint public inc = ++a;
}
```

Output:



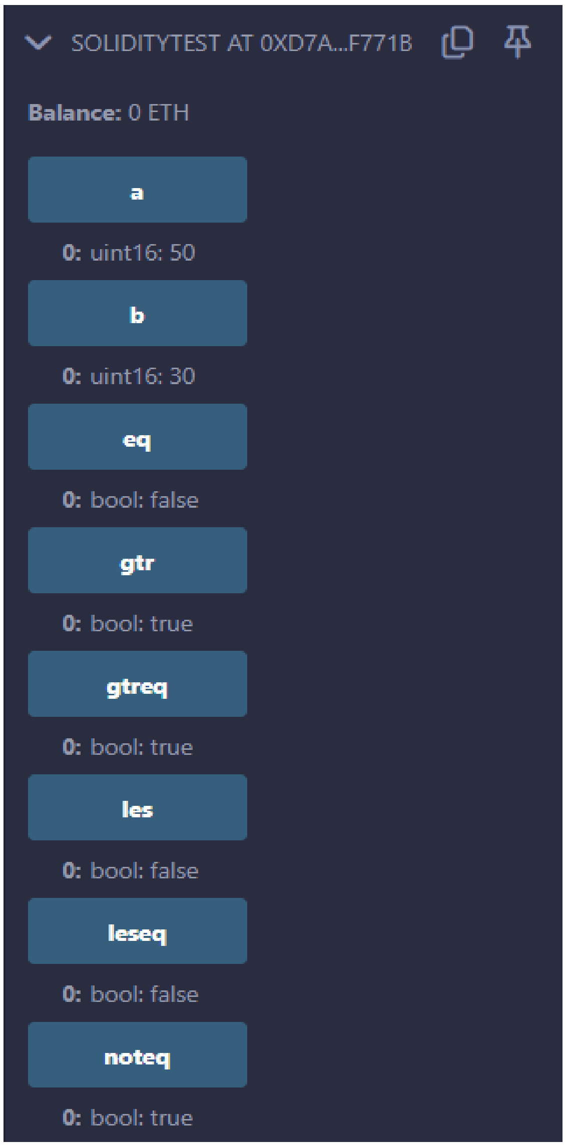
```
SOLIDITYTEST AT 0XD91...39138
Balance: 0 ETH
a
0: uint16: 21
b
0: uint16: 9
dec
0: uint256: 9
diff
0: uint256: 10
div
0: uint256: 2
inc
0: uint256: 21
mod
0: uint256: 0
mul
0: uint256: 200
sum
0: uint256: 30
```

2. Relational Operator

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.0;
// Creating a contract
contract SolidityTest {
// Declaring variables
uint16 public a = 50;
uint16 public b = 30;
// 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:



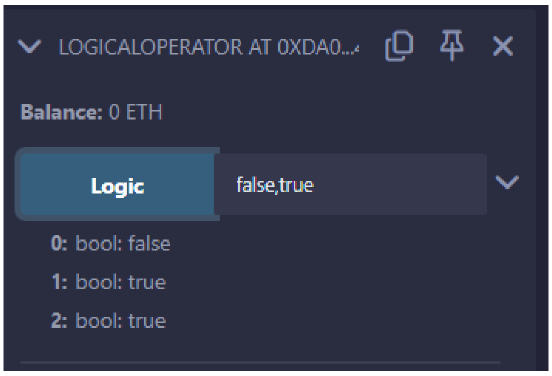
```
SOLIDITYTEST AT 0XD7A...F771B
Balance: 0 ETH
a
0: uint16: 50
b
0: uint16: 30
eq
0: bool: false
gtr
0: bool: true
gtreq
0: bool: true
les
0: bool: false
leseq
0: bool: false
noteq
0: bool: true
```

3. **Logical Operator**

Code:

```
pragma solidity ^0.5.0;
// Creating a contract
contract logicalOperator{
// Defining function to demonstrate Logical operator
function Logic(bool a, bool b) public view returns(
    bool, bool, bool){
        // Logical AND operator
        bool and = a&&b;

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

        // Logical NOT operator
        bool not = !a;
        return (and, or, not);
    }
}
```

Output:



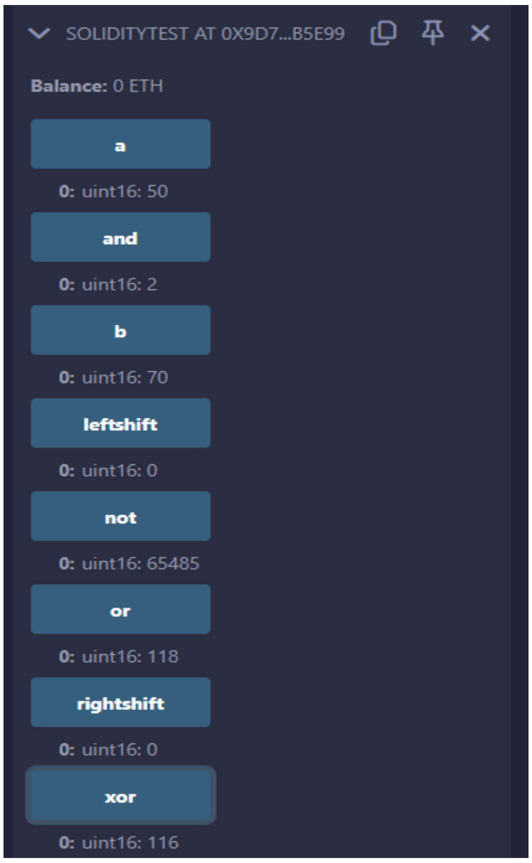
```
LOGICALOPERATOR AT 0XDA0...
Balance: 0 ETH
Logic    false,true
0: bool: false
1: bool: true
2: bool: true
```

4. **Bitwise Operator**

Code:

```
pragma solidity ^0.5.0;
// Creating a contract
contract SolidityTest {
// Declaring variables
uint16 public a = 50;
uint16 public b = 70;
// 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:



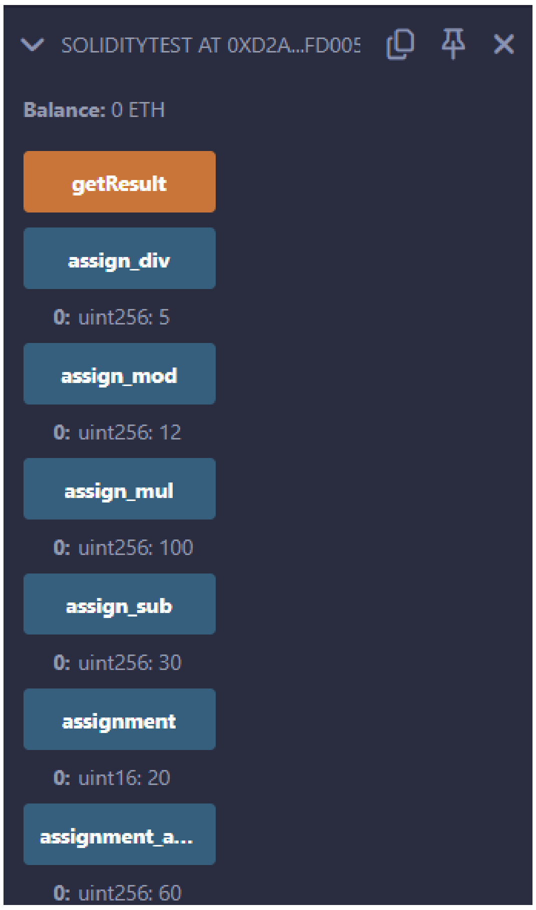
```
SOLIDITYTEST AT 0X9D7...B5E99
Balance: 0 ETH
a
0: uint16: 50
and
0: uint16: 2
b
0: uint16: 70
leftshift
0: uint16: 0
not
0: uint16: 65485
or
0: uint16: 118
rightshift
0: uint16: 0
xor
0: uint16: 116
```

5. Assignment Operator

Code:

```
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:



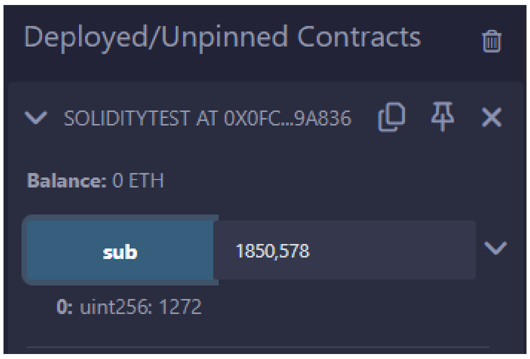
```
Balance: 0 ETH
0: uint256: 5
0: uint256: 12
0: uint256: 100
0: uint256: 30
0: uint16: 20
0: uint256: 60
```

6. Conditional Operator

Code:

```
pragma solidity ^0.5.0;
// Creating a contract
contract SolidityTest{
// Defining function to demonstrate conditional operator
function sub(
uint a, uint b) public view returns( uint){
    uint result = (a > b? a-b : b-a);
    return result;
}}
```

Output:



```
Deployed/Unpinned Contracts
SOLIDITYTEST AT 0X0FC...9A836
Balance: 0 ETH
sub    1850,578
0: uint256: 1272
```

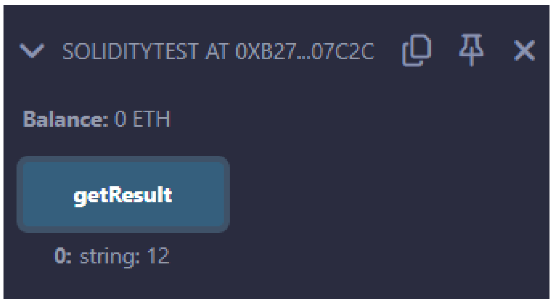
iii. Loops

1. **While Loop**

Code:

```
pragma solidity ^0.5.0;
contract SolidityTest { uint storedData; constructor() public{ storedData =
10;
}
function getResult() public view returns(string memory){
    uint a = 10;
    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:



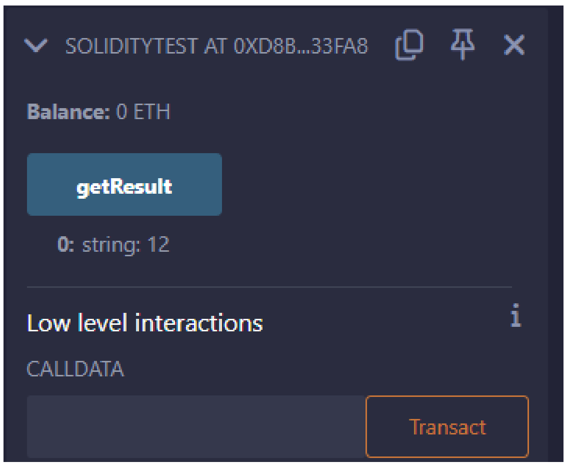
```
SOLIDITYTEST AT 0XB27...07C2C
Balance: 0 ETH
getResult
0: string: 12
```

2. **Do-while Loop**

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.5.12 <0.8.0;
contract SolidityTest {
uint storedData;
constructor() public{
storedData = 10;
}
function getResult() public view returns(string memory){
uint a = 10;
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=0;
uint len;
for (j = _i; j != 0; j /= 10) {  //for loop example
len++;
}
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:

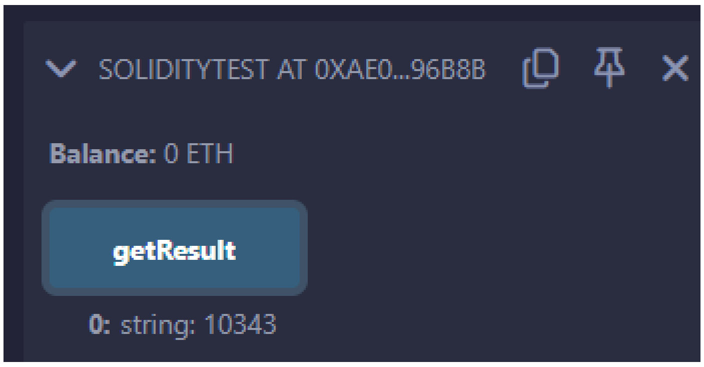


```
SOLIDITYTEST AT 0XD8B...33FA8
Balance: 0 ETH
getResult
0: string: 12
Low level interactions
CALLDATA
Transact
```

3. For Loop

Code and Output:

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.5.12 <0.8.0;
contract SolidityTest {
uint storedData;
constructor() public{
storedData = 10;
}
function getResult() public view returns(string memory){
uint a = 1587;
uint b = 8756;
uint result = a + b;
return integerToString(result);
}
function integerToString(uint _i) internal pure
returns (string memory) {
if (_i == 0) {
return "0";
}
uint j=0;
uint len;
for (j = _i; j != 0; j /= 10) {  //for loop example
len++;
}
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
}}
```



SOLIDITYTEST AT 0XAE0...96B8B

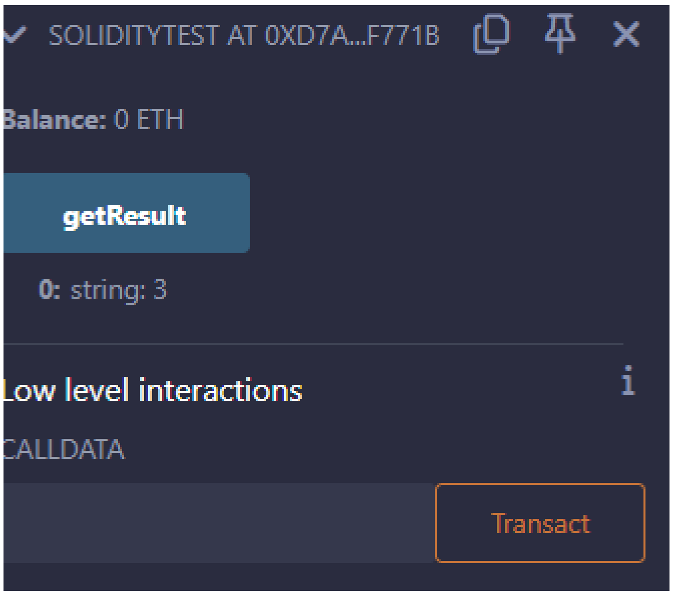
Balance: 0 ETH

```
0: string: 10343
```

4. loop Control (Break statement)

Code and Output:

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.5.12 <0.8.0;
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 (true) {
len++;
j /= 10;
if(j==0){
break;   //using break statement
}
}
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);
}
}
```

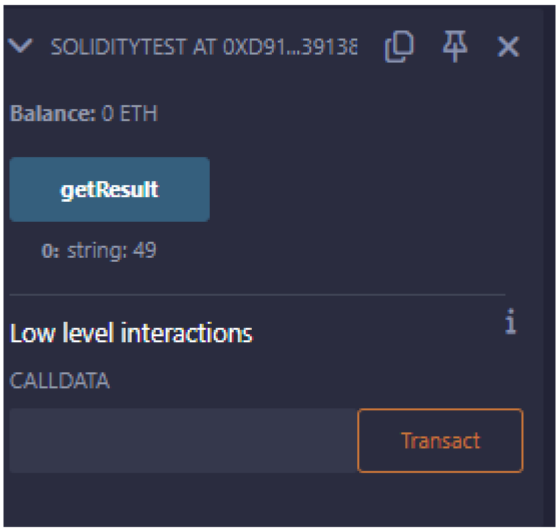


0: string: 3

5. Continue statement

Code and Output:

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.5.12 <0.8.0;
contract SolidityTest {
uint storedData;
constructor() public{
storedData = 10;
}
function getResult() public view returns(string memory){
uint n = 1;
uint sum = 0;

while( n < 10){
n++;
if(n == 5){
continue; // skip n in sum when it is 5.
}
sum = sum + n;
}
return integerToString(sum);
}
function integerToString(uint _i) internal pure
returns (string memory) {

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

while (true) {
len++;
j /= 10;
if(j==0){
break;   //using break statement
}
}
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);
}
}
```



SOLIDITYTEST AT 0XD91...39138

Balance: 0 ETH

0: string: 49

Low level interactions

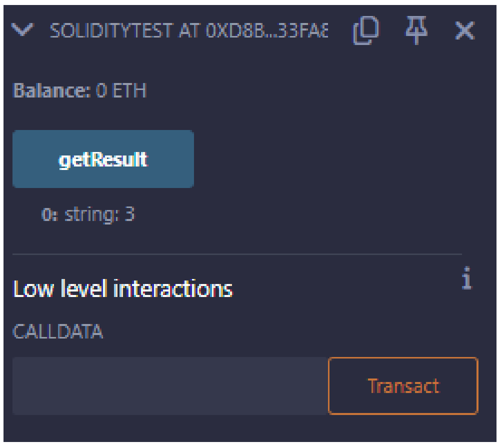
CALLDATA

iv. Decision Making

1. If-statement

Code and Output:

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.5.12 <0.8.0;
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) {   // if statement
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
}}
```

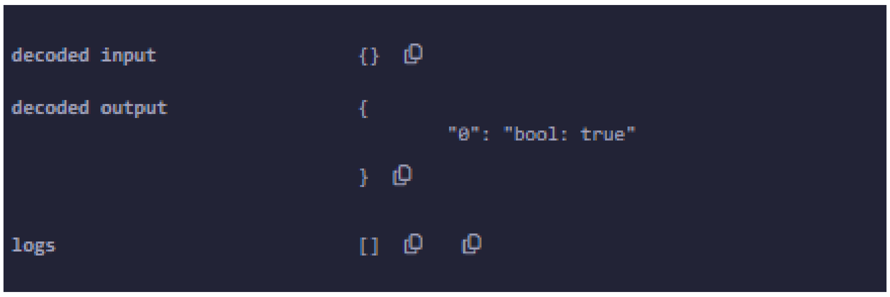


```
SOLIDITYTEST AT 0XD8B...33FA8
Balance: 0 ETH
getResult
0: string: 3
Low level interactions
CALLDATA
Transact
```

2.If-else statement:

Code and Output:

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.5.12 <0.8.0;
contract Types {
// Declaring state variables
uint i = 10;
bool even;

// Defining function to
// demonstrate the use of
// 'if...else statement'
function decision_making(
) public payable returns(bool){
if (i%2 == 0){
even = true;
}
else{
even = false;
}
return even;
}}
```

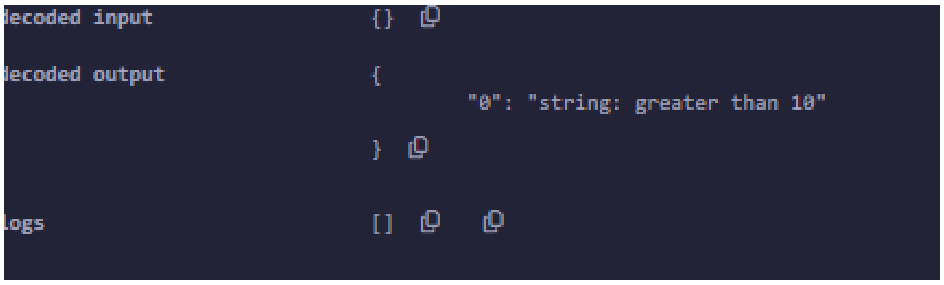


```
decoded input          {}
decoded output         {
                           "0": "bool: true"
                       }
logs                   []
```

3.If-else-if statement:

Code and Output:

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.5.12 <0.8.0;
contract Types {
// Declaring state variables
uint i = 12;
string result;
// Defining function to
// demonstrate the use
// of 'if...else if...else
// statement'
function decision_making (
) public returns(string memory){
if(i<10){
result = "less than 10";
}
else if(i == 10){
result = "equal to 10";
}
else{
result = "greater than 10";
}
return result;
}
}
```

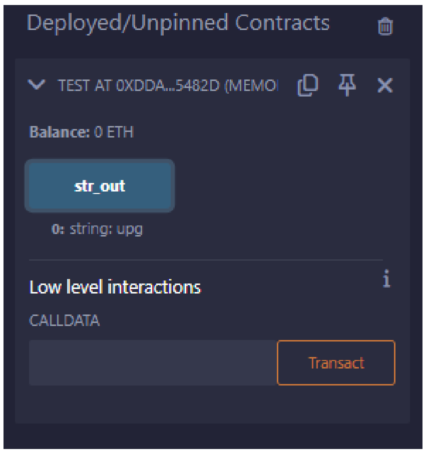


```
decoded input          {}
decoded output         {
                            "0": "string: greater than 10"
                       }
logs                   []
```

v. String

Code and Output:

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.6.12 <0.9.0;
// Creating a contract
contract Test {
// Declaring variable
string  str;
// Defining a constructor
constructor(string memory str_in){
    str = str_in;
}
// Defining a function to
// return value of variable 'str'
function str_out() public view returns(string memory){
return str;
}}
```

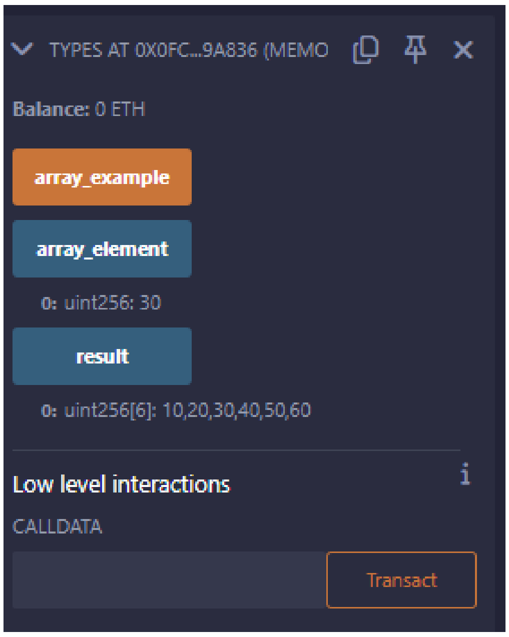


```
Deployed/Unpinned Contracts
TEST AT 0XDDA...5482D (MEMO
Balance: 0 ETH
str_out
0: string: upg
Low level interactions
CALLDATA
Transact
```

vi. Arrays:

Code and Output:

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.6.12 <0.9.0;
// Creating a contract
contract Types {
// Declaring an array
uint[6] data;
uint x;
// Defining function to
// assign values to array
function array_example() public returns (uint[6] memory)
{

data  = [uint(10), 20, 30, 40, 50, 60];
}
function result() public view returns(uint[6] memory){
return data;
}
// Defining function to access
// values from the array
// from a specific index
function array_element() public view returns (uint){
uint x = data[2];
return x;
}
}
```

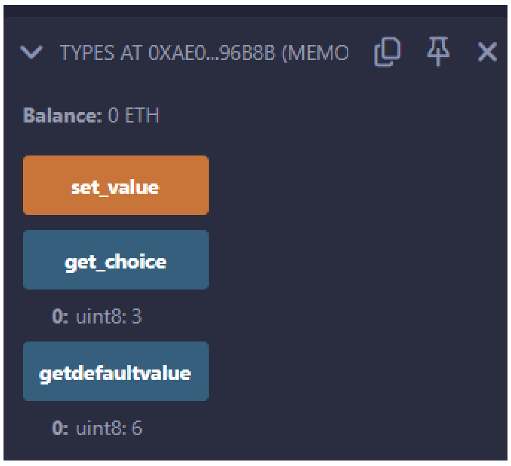


```
TYPES AT 0X0FC...9A836 (MEMO
Balance: 0 ETH
array_example
array_element
0: uint256: 30
result
0: uint256[6]: 10,20,30,40,50,60
Low level interactions
CALLDATA
Transact
```

vii. Enums

Code and Output:

```
pragma solidity >=0.5.12 <0.8.0;
// Creating a contract
contract Types
{
// Creating an enumerator
enum week_days {
Monday
,
Tuesday
,
Wednesday
,
Thursday
,
Friday
,
Saturday
,
Sunday }
// Declaring variables of
// type enumerator
week_days week
;
week_days choice
;
// Setting a default value
week_days constant default_value = week_days.Sunday;
// Defining a function to
// set value of choice
function set_value() public
{
choice
= week_days
.Thursday
;
}
// Defining a function to
// return value of choice
function get_choice
(
) public view returns
(week_days
)
{
return choice
;
}
// Defining function to
// return default value
function getdefaultvalue
(
) public pure returns
(week_days
)
{
return default_value
;
}}
```

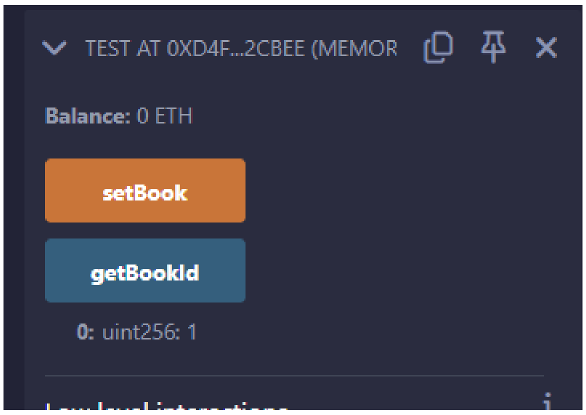


```
Balance: 0 ETH
0: uint8: 3
0: uint8: 6
```

viii. Structure
Code and Output:

```
pragma solidity >=0.5.12 <0.8.0;
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) {
```

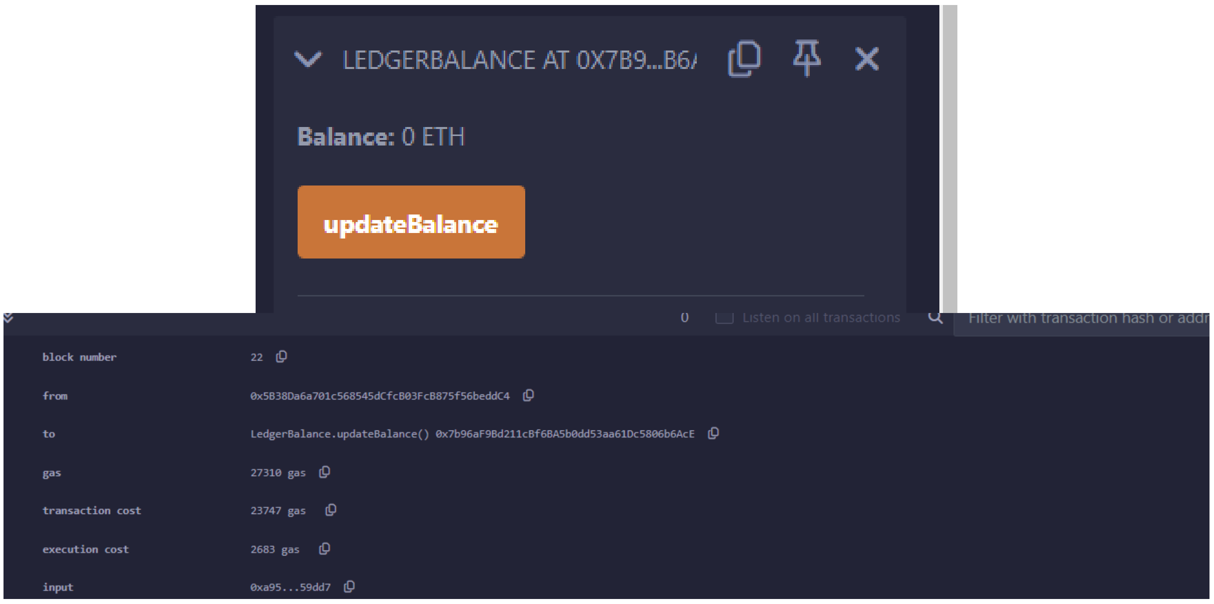
```
return book.book_id;
}
}
```



ix. Mappings

Code and Output:

```
pragma solidity >=0.5.12 <0.8.0;
contract LedgerBalance {
mapping(address => uint) balance;
function updateBalance() public returns(uint) {
balance[msg.sender]=30;
return balance[msg.sender];
}
}
```


LEDGERBALANCE AT 0X7B9...B6/
Balance: 0 ETH
updateBalance
0
Listen on all transactions
Filter with transaction hash or addr

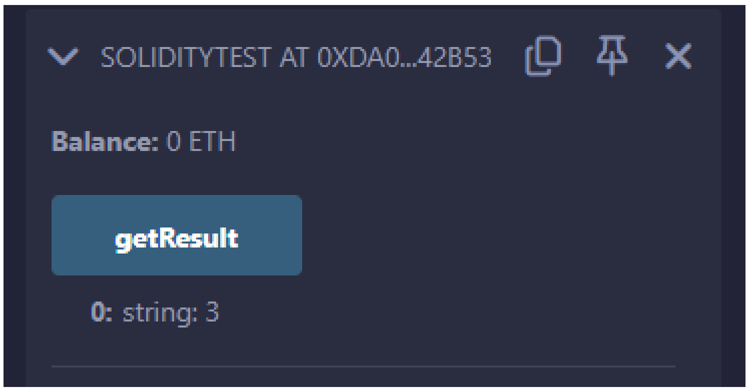
```
block number          22
from                  0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to                    LedgerBalance.updateBalance() 0x7b96aF9Bd211cBf6BA5b0dd53aa61Dc5806b6AcE
gas                   27310 gas
transaction cost      23747 gas
execution cost        2683 gas
input                 0xa95...59dd7
```

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

i. Functions

Code and Output:

```
pragma solidity >=0.5.12 <0.8.0;
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
}}
```

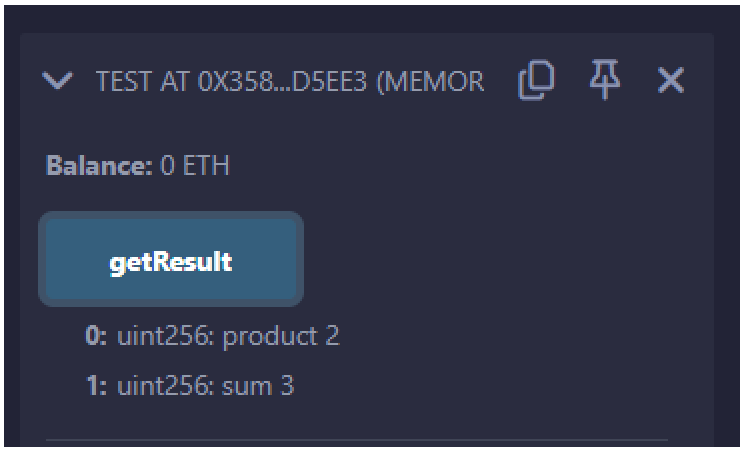


```
Balance: 0 ETH
0: string: 3
```

ii. View Function

Code and Output:

```
pragma solidity >=0.5.12 <0.8.0;
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;
}}
```

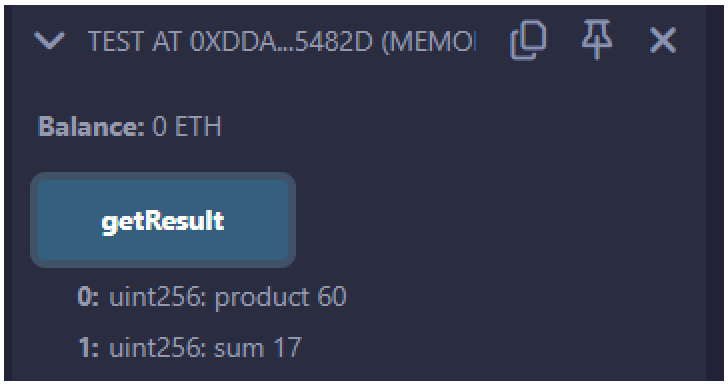


```
TEST AT 0X358...D5EE3 (MEMOR
Balance: 0 ETH
getResult
0: uint256: product 2
1: uint256: sum 3
```

iii. Pure Functions

Code and Output:

```
pragma solidity >=0.5.12 <0.8.0;
contract Test {
function getResult() public pure returns(uint product, uint sum){
uint a = 12;
uint b = 5;
product = a * b;
sum = a + b;
}}
```

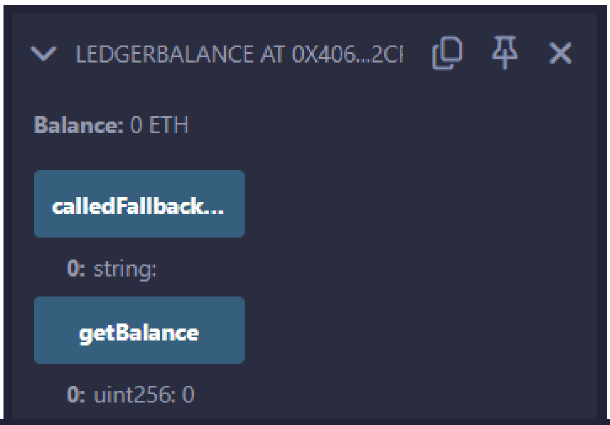


```
TEST AT 0XDDA...5482D (MEMO
Balance: 0 ETH
getResult
0: uint256: product 60
1: uint256: sum 17
```

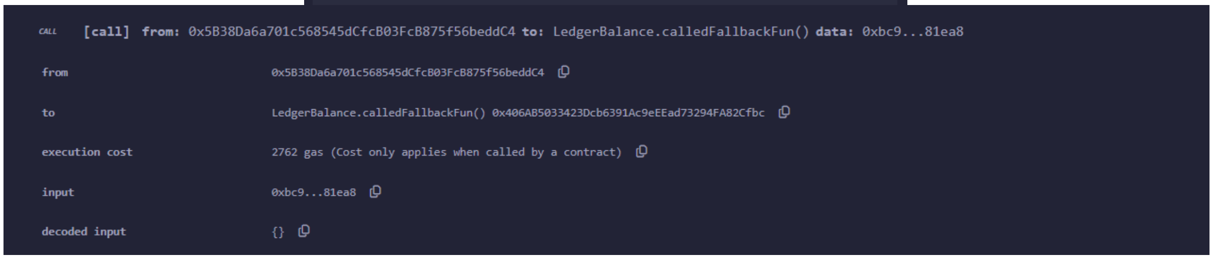
iv. Fallback Function

Code and Output:

```
pragma solidity >=0.5.12 <0.8.0;
contract LedgerBalance {
string public calledFallbackFun;
fallback() external payable{
calledFallbackFun="Fallback function is executed!";
}
function getBalance() public view returns (uint) {
return address(this).balance;
}
}
contract Sender
{
function transferEther() public payable
{
(bool sent, ) =
payable(0xD4Fc541236927E2EAf8F27606bD7309C1Fc2cbee).call{value: 2
ether}("Transaction Completed!");
require(sent, "Transaction Failed!");
}
function getBalance() public view returns (uint) {
return address(this).balance;
}}
```



```
LEDGERBALANCE AT 0X406...2CI
Balance: 0 ETH
calledFallback...
0: string:
getBalance
0: uint256: 0
```

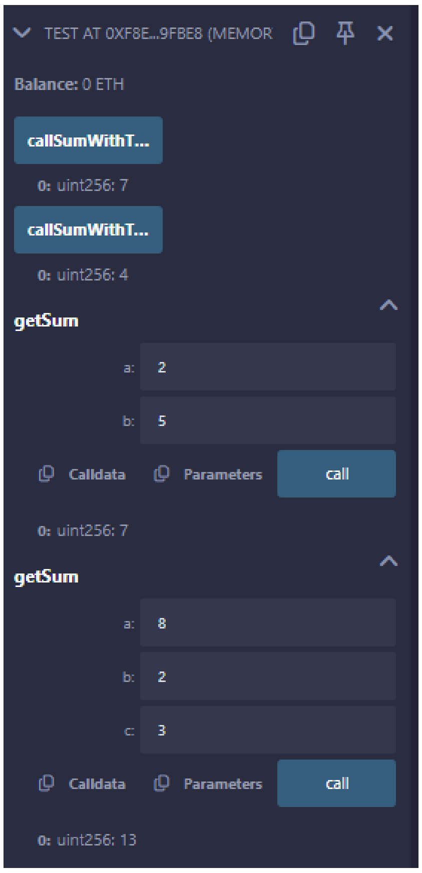


```
CALL  [call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: LedgerBalance.calledFallbackFun() data: 0xbc9...81ea8
from            0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to              LedgerBalance.calledFallbackFun() 0x406AB5033423Dcb6391Ac9eEEad73294FA82Cfbc
execution cost  2762 gas (Cost only applies when called by a contract)
input           0xbc9...81ea8
decoded input   {}
```

v. Function Overloading

Code and Output:

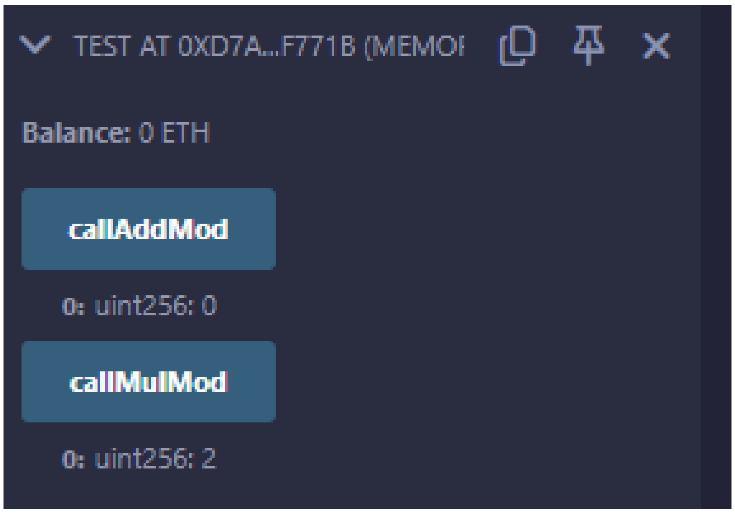
```
contract Test {
function getSum(uint a, uint b) public pure returns(uint){
return a + b;}
function getSum(uint a, uint b, uint c ) public pure returns(uint){
return a + b + c;}
function callSumWithTwoArguments() public pure returns(uint){
return getSum(2,2);}
function callSumWithThreeArguments() public pure returns(uint){
return getSum(1,2,4);
}}
```



```
TEST AT 0XF8E...9FBE8 (MEMOR
Balance: 0 ETH
callSumWithT...
0: uint256: 7
callSumWithT...
0: uint256: 4
getSum
a: 2
b: 5
Calldata  Parameters  call
0: uint256: 7
getSum
a: 8
b: 2
c: 3
Calldata  Parameters  call
0: uint256: 13
```

vi. Mathematical Function
Code and Output:

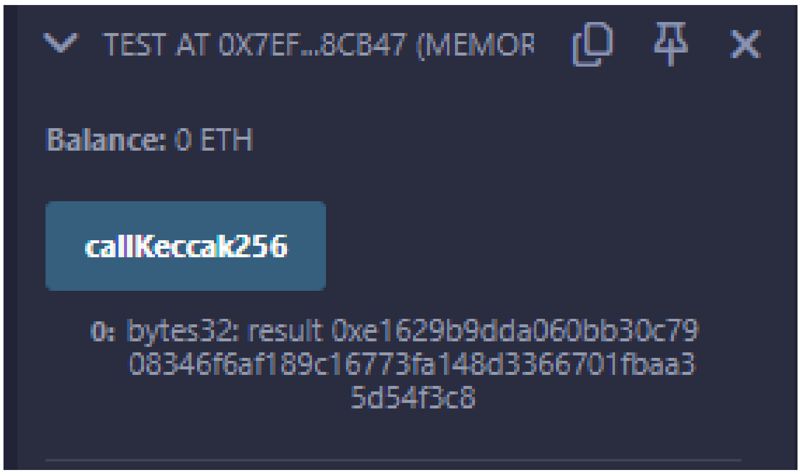
```
contract Test {
function callAddMod() public pure returns(uint){
return addmod(4, 5, 3);
}
function callMulMod() public pure returns(uint){
return mulmod(4, 5, 3);
}}
```



```
TEST AT 0XD7A...F771B (MEMOI
Balance: 0 ETH
callAddMod
0: uint256: 0
callMulMod
0: uint256: 2
```

vii. Cryptographic Function
Code and Output:

```
pragma solidity ^0.5.0;
contract Test {
function callKeccak256() public pure returns(bytes32 result){
return keccak256("ABC");
}
}
```



```
TEST AT 0X7EF...8CB47 (MEMOR
Balance: 0 ETH
callKeccak256
0: bytes32: result 0xe1629b9dda060bb30c79
   08346f6af189c16773fa148d3366701fbaa3
   5d54f3c8
```

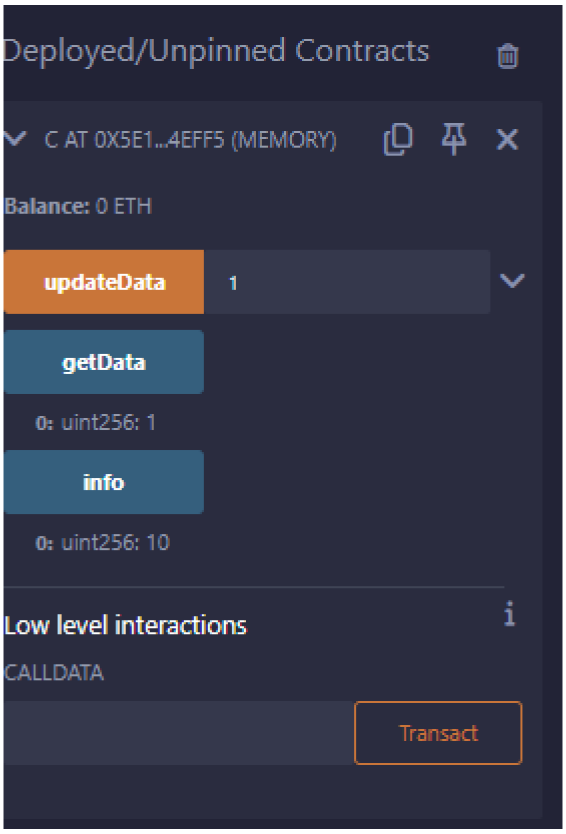
Practical 3

A. Contracts, Inheritance, Constructors, Abstract Contracts, Interfaces.

i. Contract

Code and Output:

```
pragma solidity >=0.5.0 <0.7.0;
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() virtual public view returns(uint) { return data; }
function compute(uint a, uint b) internal pure returns (uint) { return a + b;
}}
//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(); }
}
```

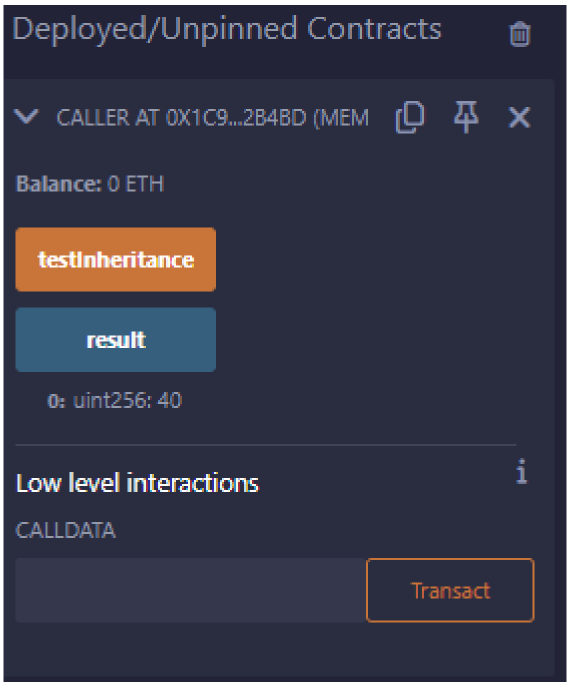


```
Balance: 0 ETH
updateData 1
0: uint256: 1
0: uint256: 10
```

ii. Inheritance

Code and Output:

```
pragma solidity >=0.5.0 <0.7.0;
// Defining contract
contract parent{
// Declaring internal
// state variable
uint internal sum;
// Defining external function
// to set value of internal
// state variable sum
function setValue() external {
uint a = 20;
uint b = 20;
sum = a + b;
}
}
// Defining child contract
contract child is parent{
// Defining external function
// to return value of
// internal state variable sum
function getValue() external view returns(uint) {
return sum;
}
}
// Defining calling contract
contract caller {
// Creating child contract object
child cc = new child();
// Defining function to call
// setValue and getValue functions
function testInheritance() public {
cc.setValue();
}
function result() public view returns(uint ){
return cc.getValue();
}}
```



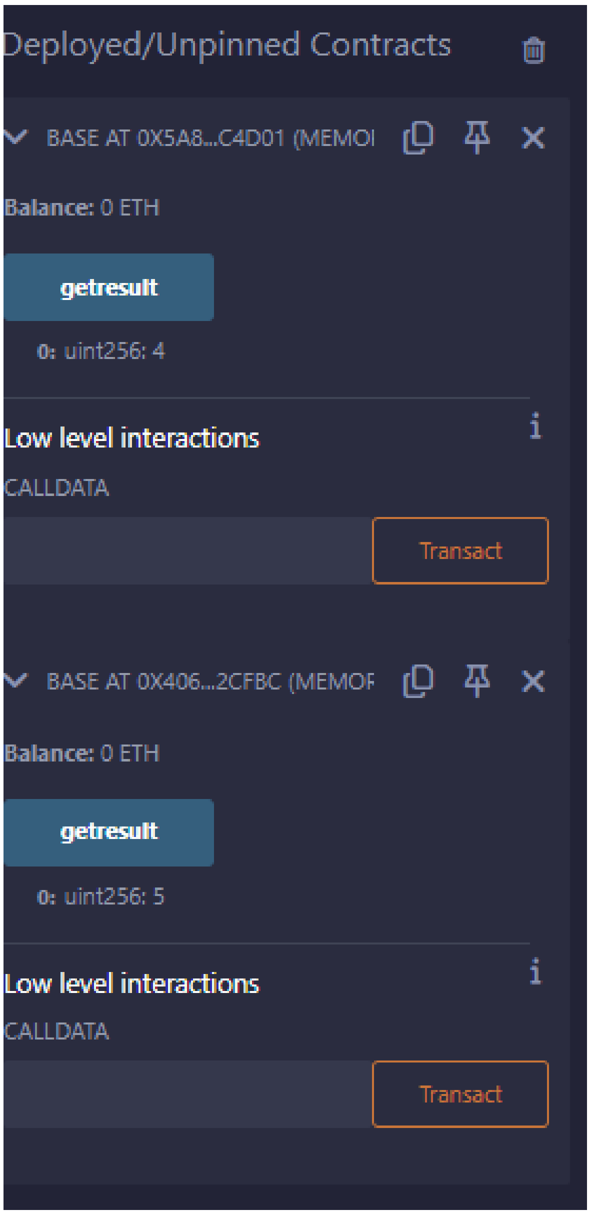
```
0: uint256: 40
```

iii. Constructors

Code and Output:

```
pragma solidity >=0.5.0 <0.7.0;
```

```
contract Base {
uint data;
constructor(uint _data) public {
data = _data;
}
function getresult()public view returns(uint){
return data;
}}
contract Derived is Base (5) {
constructor() public {}
}
// Indirect Initialization of Base Constructor
pragma solidity >=0.5.0 <0.7.0;
contract Base {
uint data;
constructor(uint _data) public {
data = _data;
}
function getresult()public view returns(uint){
return data;
}}
contract Derived is Base {
constructor(uint _info) Base(_info * _info) public {}
}
```


Deployed/Unpinned Contracts

BASE AT 0X5A8...C4D01 (MEMOI

Balance: 0 ETH

getresult

0: uint256: 4

Low level interactions

CALLDATA

Transact

BASE AT 0X406...2CFBC (MEMOF

Balance: 0 ETH

getresult

0: uint256: 5

Low level interactions

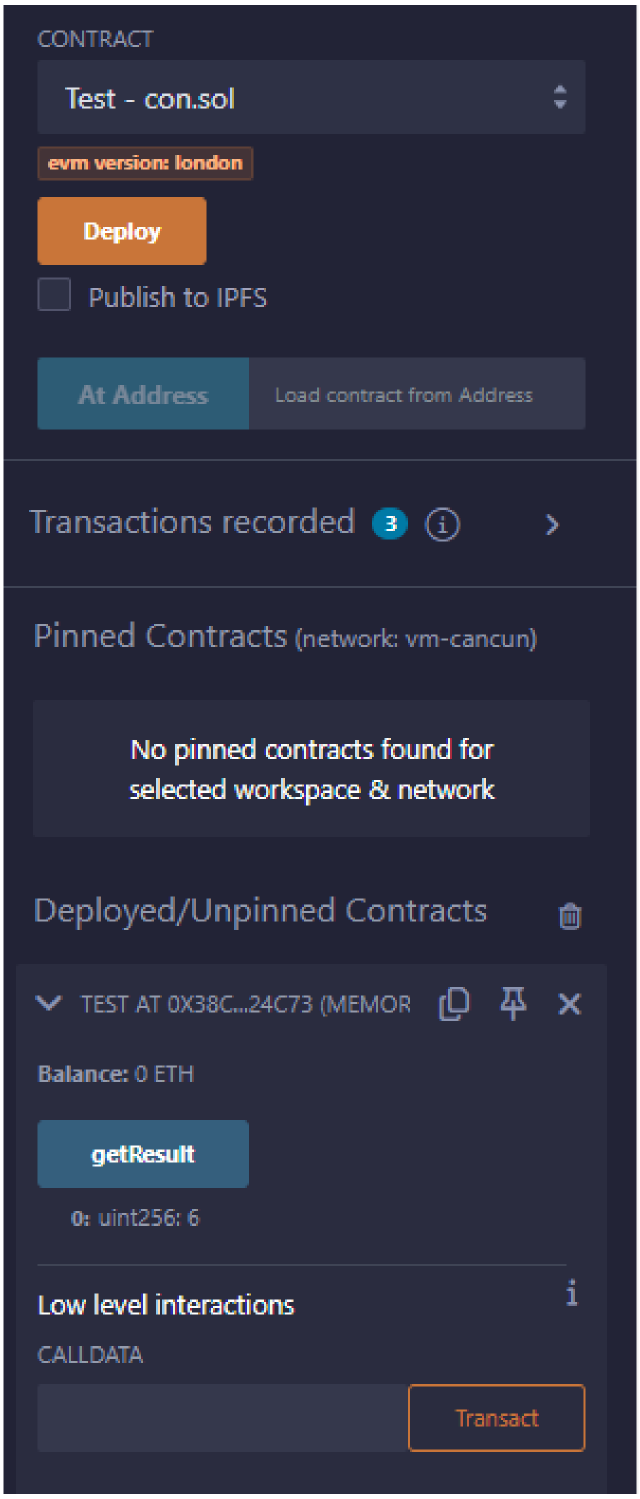
CALLDATA

Transact

iv. Abstract Contract

Code and Output:

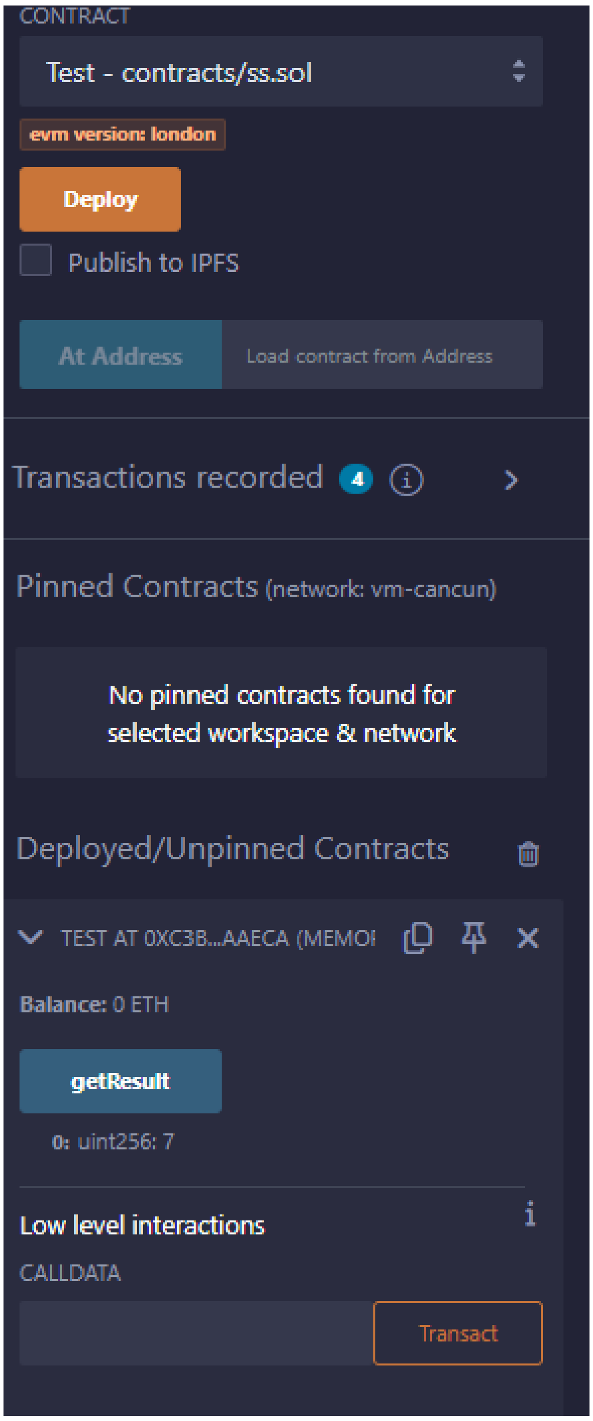
```
pragma solidity ^0.8.0;
abstract contract Calculator {
function getResult() public view virtual  returns(uint);
}
contract Test is Calculator {
function getResult() public view override returns(uint) {
uint a = 4;
uint b = 2;
uint result = a + b;
return result;
}}
```


CONTRACT
Test - con.sol
evm version: london
Deploy
Publish to IPFS
At Address
Load contract from Address
Transactions recorded 3
Pinned Contracts (network: vm-cancun)
No pinned contracts found for selected workspace & network
Deployed/Unpinned Contracts
TEST AT 0X38C...24C73 (MEMOR
Balance: 0 ETH
getResult
0: uint256: 6
Low level interactions
CALLDATA
Transact

v. Interfaces

Code and Output:

```
pragma solidity ^0.8.0;
interface Calculator {
function getResult() external view returns(uint);
}
contract Test is Calculator {
constructor() public {}
function getResult() external view returns(uint){
uint a = 5;
uint b = 2;
uint result = a + b;
return result;
}}
```



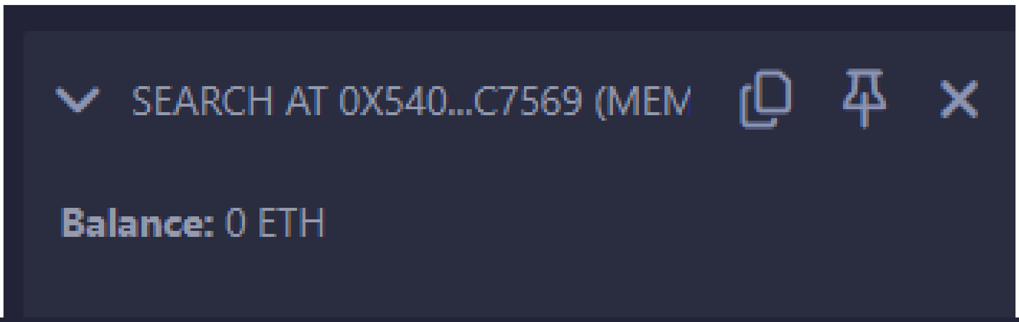
```
CONTRACT
Test - contracts/ss.sol
evm version: london
Deploy
Publish to IPFS
At Address    Load contract from Address
Transactions recorded 4
Pinned Contracts (network: vm-cancun)
No pinned contracts found for
selected workspace & network
Deployed/Unpinned Contracts
TEST AT 0XC3B...AAECA (MEMOI
Balance: 0 ETH
getResult
0: uint256: 7
Low level interactions
CALLDATA
Transact
```

B. Libraries, Assembly, Events, Error handling.

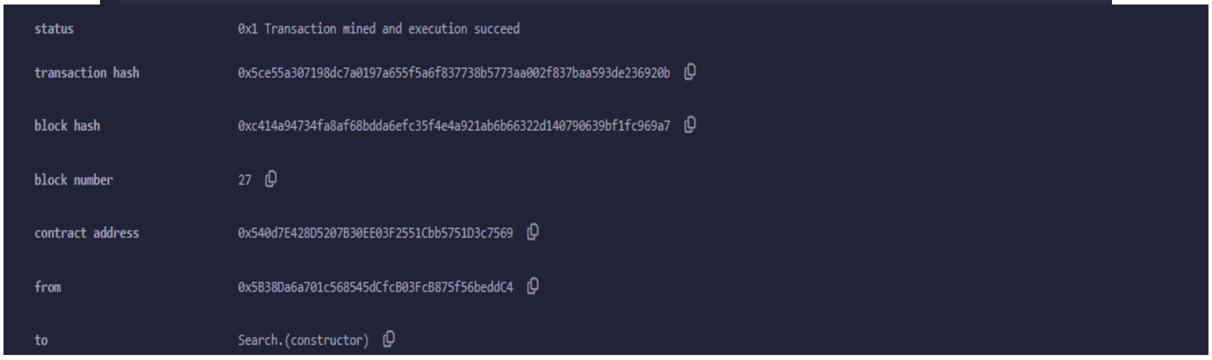
i. Libraries

Code and Output:

```
pragma solidity >=0.5.12 <0.8.0;
library Search {
function indexOf(uint[] storage self, uint value) public view returns (uint)
{
for (uint i = 0; i < self.length; i++)
if (self[i] == value) return i;
return uint(-1);}
}
contract Test {
uint[] data;
uint value;
uint index;
constructor() public {
data.push(6);
data.push(7);
data.push(8);
data.push(9);
data.push(10);
}
function isValuePresent() external {
value = 9;
//search if value is present in the array using Library function
index = Search.indexOf(data, value);
}
function getresult() public view returns(uint){
return index;
}
}
```



Balance: 0 ETH

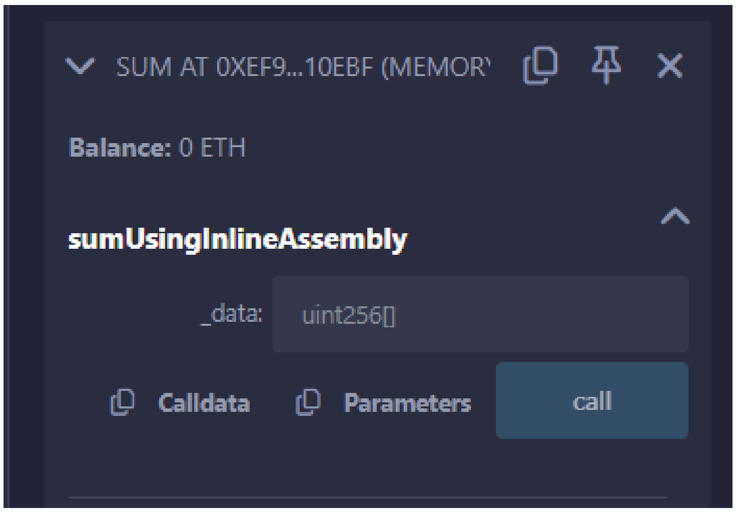


```
status              0x1 Transaction mined and execution succeed
transaction hash    0x5ce55a307198dc7a0197a655f5a6f837738b5773aa002f837baa593de236920b
block hash          0xc414a94734fa8af68bdda6efc35f4e4a921ab6b66322d140790639bf1fc969a7
block number        27
contract address    0x540d7E428D5207B30EE03F2551Cbb5751D3c7569
from                0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to                  Search.(constructor)
```

ii. Assembly

Code and Output:

```
pragma solidity >=0.5.12 <0.8.0;
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);
}}
```



SUM AT 0XEF9...10EBF (MEMORY

Balance: 0 ETH

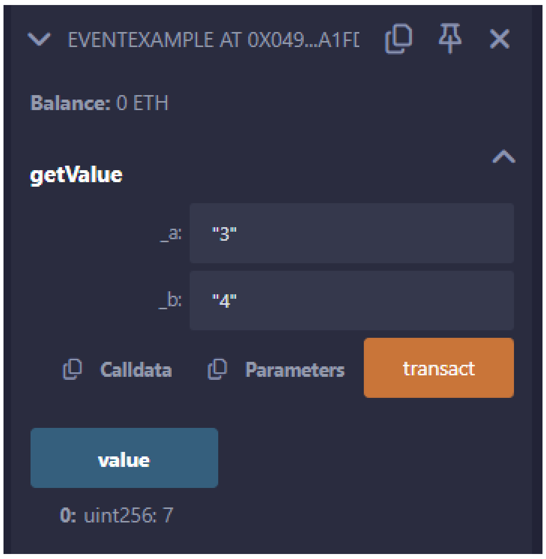
sumUsingInlineAssembly

_data: uint256[]

iii. Events

Code and Output:

```
pragma solidity >=0.5.12 <0.8.0;
// 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(uint _a, uint _b) public {
emit Increment(msg.sender);
value = _a + _b;
}}
```



EVENTEXAMPLE AT 0X049...A1FD

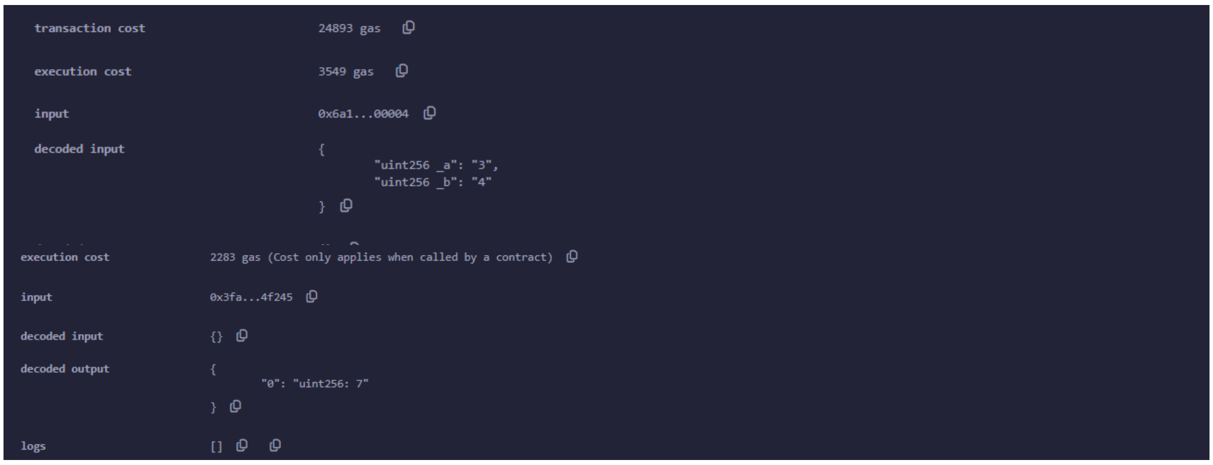
Balance: 0 ETH

getValue

_a: "3"

_b: "4"

0: uint256: 7

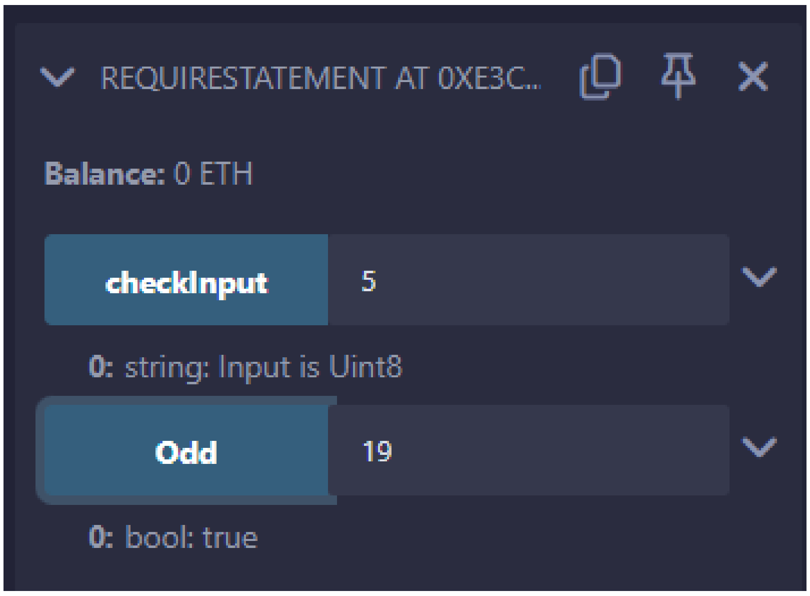


```
transaction cost        24893 gas

execution cost          3549 gas

input                   0x6a1...00004

decoded input           {
                               "uint256 _a": "3",
                               "uint256 _b": "4"
                        }

execution cost     2283 gas (Cost only applies when called by a contract)

input              0x3fa...4f245

decoded input      {}

decoded output     {
                          "0": "uint256: 7"
                   }

logs               []
```

iv. Error Handling

Code and Output:

```
pragma solidity >=0.5.12 <0.8.0;
// Creating a contract
contract requireStatement {
// Defining function to
// check input
function checkInput(uint8 _input) public view returns(string memory){
require(_input >= 0, "invalid uint");
require(_input <= 255, "invalid uint8");
return "Input is Uint8";
}
// Defining function to
// use require statement
function Odd(uint _input) public view returns(bool){
require(_input % 2 != 0);
return true;
}
}
```



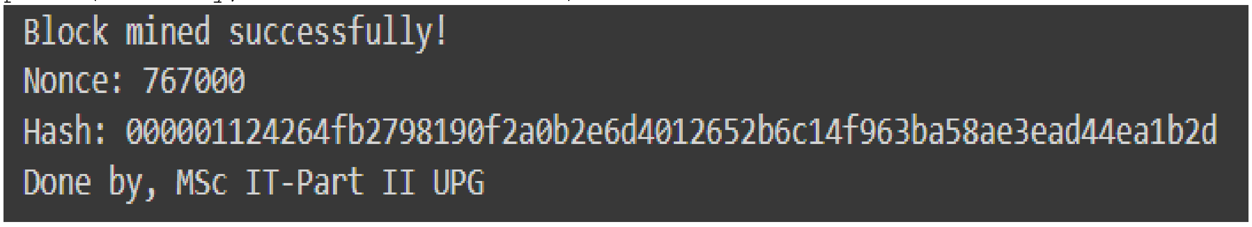
```
REQUIRESTATEMENT AT 0XE3C...

Balance: 0 ETH

checkInput   5
0: string: Input is Uint8

Odd   19
0: bool: true
```

Practical 4

A. Write a program to demonstrate mining of Ether.

Code and Output:

```
#Practical 4
import hashlib
def mine_block(previous_hash, transactions, difficulty):
  nonce = 0
  prefix = '0' * difficulty
  while True:
    data = str(nonce) + previous_hash + transactions
    hash_result = hashlib.sha256(data.encode()).hexdigest()
    if hash_result.startswith(prefix):
      print("Block mined successfully!")
      print("Nonce:", nonce)
      print("Hash:", hash_result)
      return hash_result
    nonce += 1
def main():
                                                                            previous_hash
='0000000000000000000000000000000000000000000000000000000000000000' # Initial
hash
  transactions = 'A sends 1 Ether to B' # Example transaction data
  difficulty = 5 # Number of leading zeros required in the hash
  mined_hash = mine_block(previous_hash, transactions, difficulty)
  # Add code here to broadcast the mined hash to the Ethereum network
if __name__ == '__main__':
  main()
print('Done by, MSc IT-Part II UPG')
```



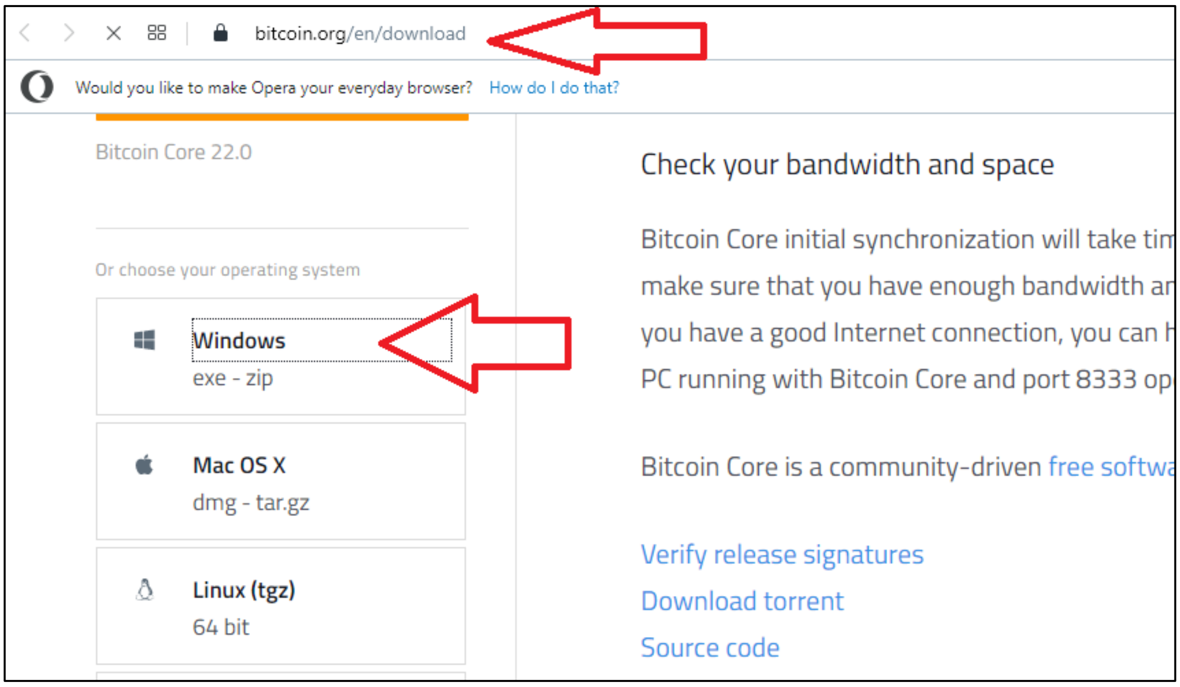
```
Block mined successfully!
Nonce: 767000
Hash: 000001124264fb2798190f2a0b2e6d4012652b6c14f963ba58ae3ead44ea1b2d
Done by, MSc IT-Part II UPG
```

Practical 5

A. Demonstrate the running of the blockchain node.

Step 1: Visit: https://bitcoin.org/en/download

Step 2: Download windows setup [use and try with Linux version as well]



bitcoin.org/en/download

Would you like to make Opera your everyday browser? How do I do that?

Bitcoin Core 22.0

Or choose your operating system

Windows
exe - zip

Mac OS X
dmg - tar.gz

Linux (tgz)
64 bit

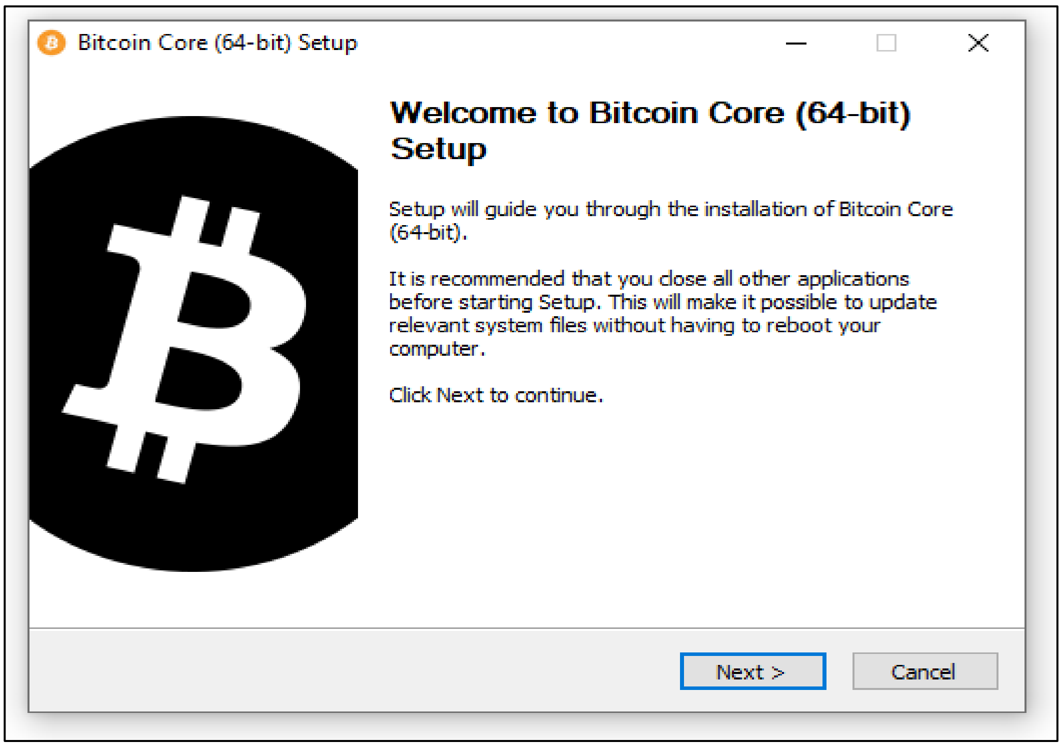
Check your bandwidth and space

Bitcoin Core initial synchronization will take tim
make sure that you have enough bandwidth ar
you have a good Internet connection, you can h
PC running with Bitcoin Core and port 8333 op

Bitcoin Core is a community-driven free softwa

Verify release signatures
Download torrent
Source code

Step 3: Run the setup file-> click next



Bitcoin Core (64-bit) Setup

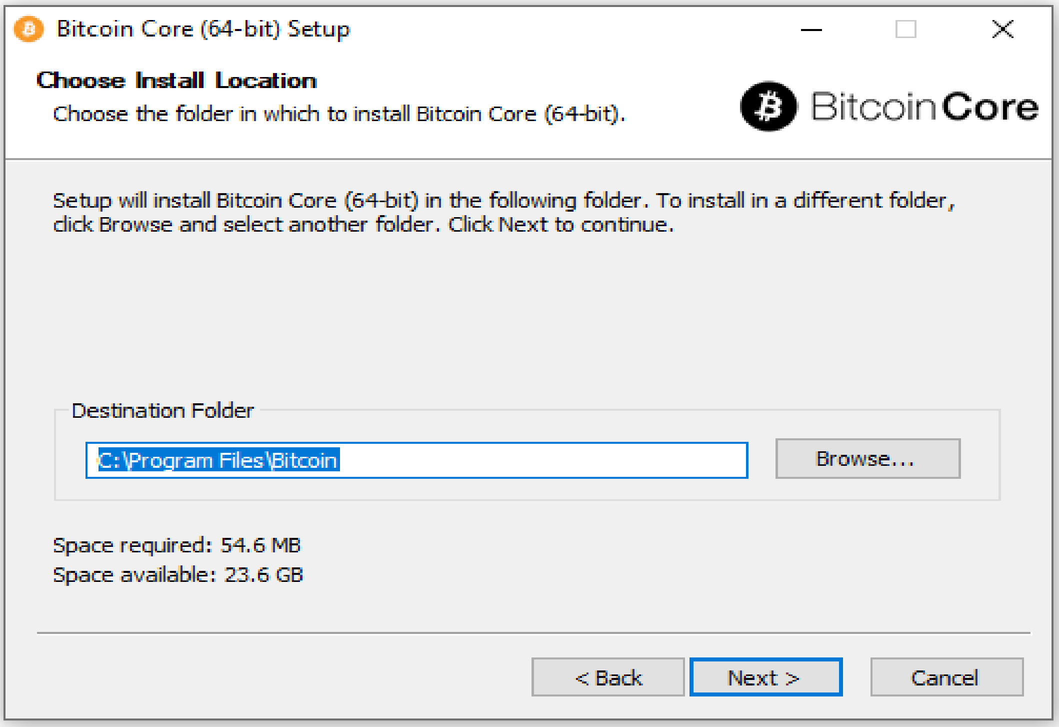
Welcome to Bitcoin Core (64-bit) Setup

Setup will guide you through the installation of Bitcoin Core (64-bit).

It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.

Click Next to continue.

Step 4: Click Next


Bitcoin Core (64-bit) Setup

Choose Install Location

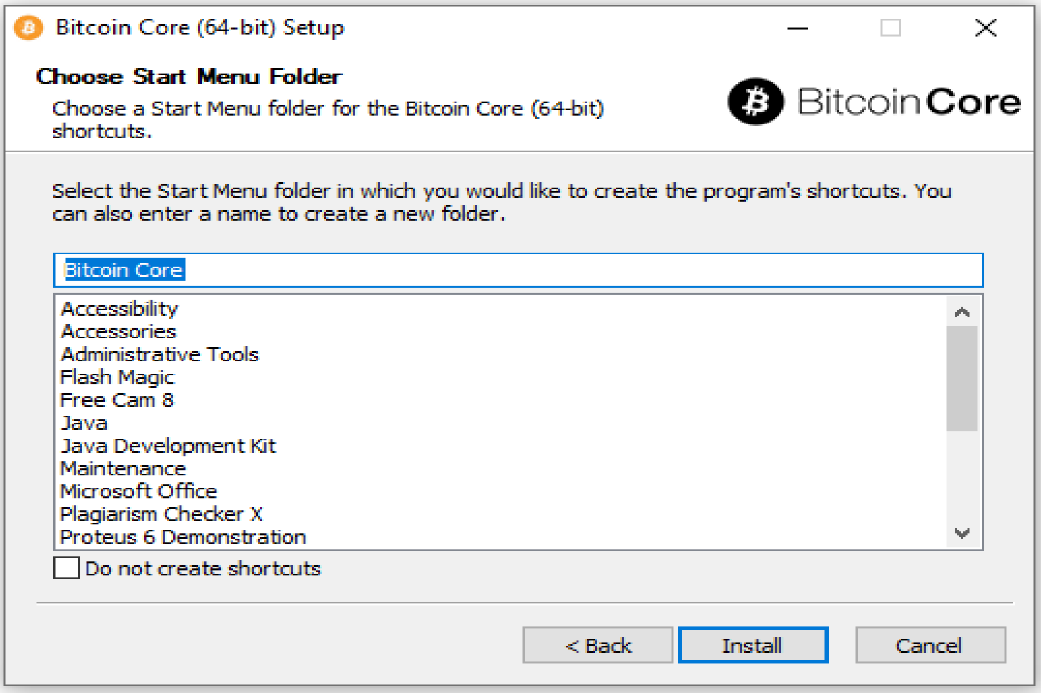
Choose the folder in which to install Bitcoin Core (64-bit).

Setup will install Bitcoin Core (64-bit) in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.

Destination Folder: C:\Program Files\Bitcoin

Space required: 54.6 MB
Space available: 23.6 GB

Step 5: Finally click on Install


Bitcoin Core (64-bit) Setup

Choose Start Menu Folder

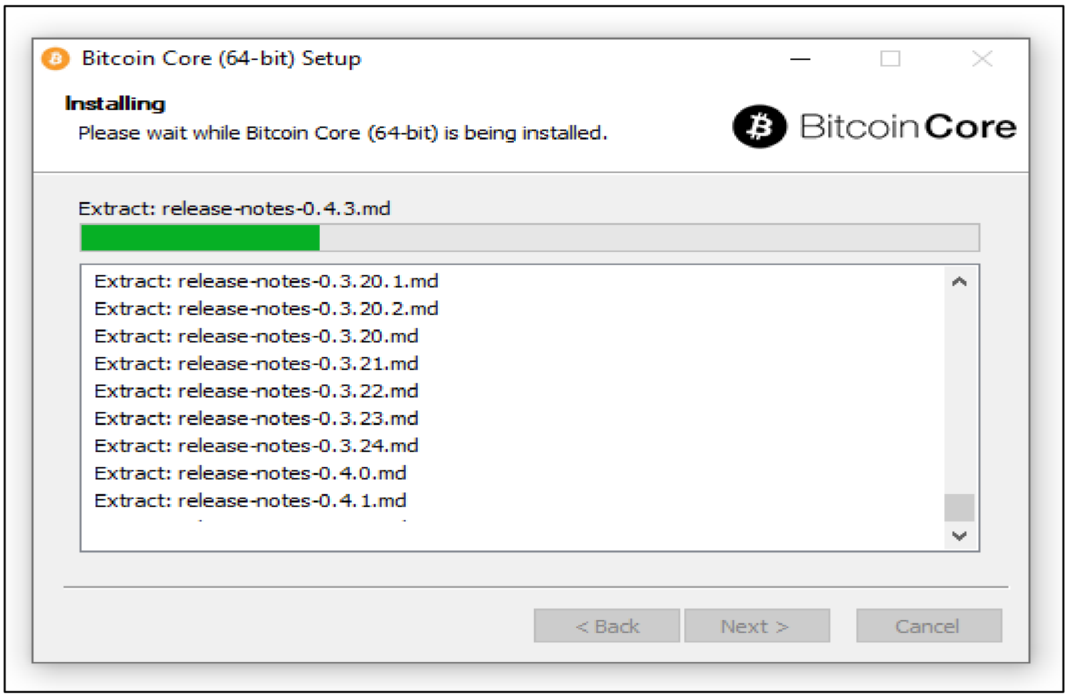
Choose a Start Menu folder for the Bitcoin Core (64-bit) shortcuts.

Select the Start Menu folder in which you would like to create the program's shortcuts. You can also enter a name to create a new folder.

Bitcoin Core

Accessibility
Accessories
Administrative Tools
Flash Magic
Free Cam 8
Java
Java Development Kit
Maintenance
Microsoft Office
Plagiarism Checker X
Proteus 6 Demonstration

☐ Do not create shortcuts

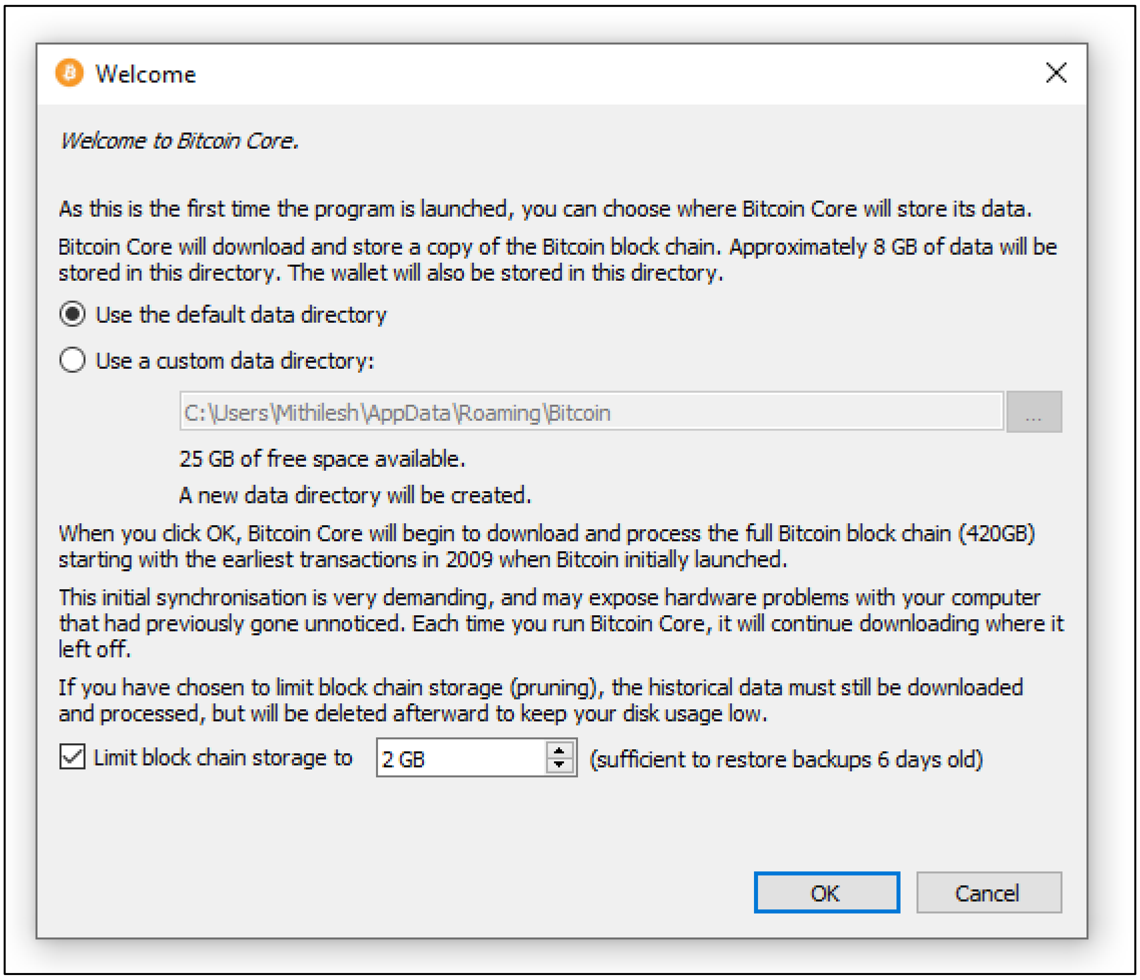


Installing

Please wait while Bitcoin Core (64-bit) is being installed.

```
Extract: release-notes-0.4.3.md

Extract: release-notes-0.3.20.1.md
Extract: release-notes-0.3.20.2.md
Extract: release-notes-0.3.20.md
Extract: release-notes-0.3.21.md
Extract: release-notes-0.3.22.md
Extract: release-notes-0.3.23.md
Extract: release-notes-0.3.24.md
Extract: release-notes-0.4.0.md
Extract: release-notes-0.4.1.md
```

Step 6: Launch Bitcoin Core-> Click OK.



Welcome to Bitcoin Core.

As this is the first time the program is launched, you can choose where Bitcoin Core will store its data.

Bitcoin Core will download and store a copy of the Bitcoin block chain. Approximately 8 GB of data will be stored in this directory. The wallet will also be stored in this directory.

[✓ Use the default data directory]

Use a custom data directory:

C:\Users\Mithilesh\AppData\Roaming\Bitcoin

25 GB of free space available.

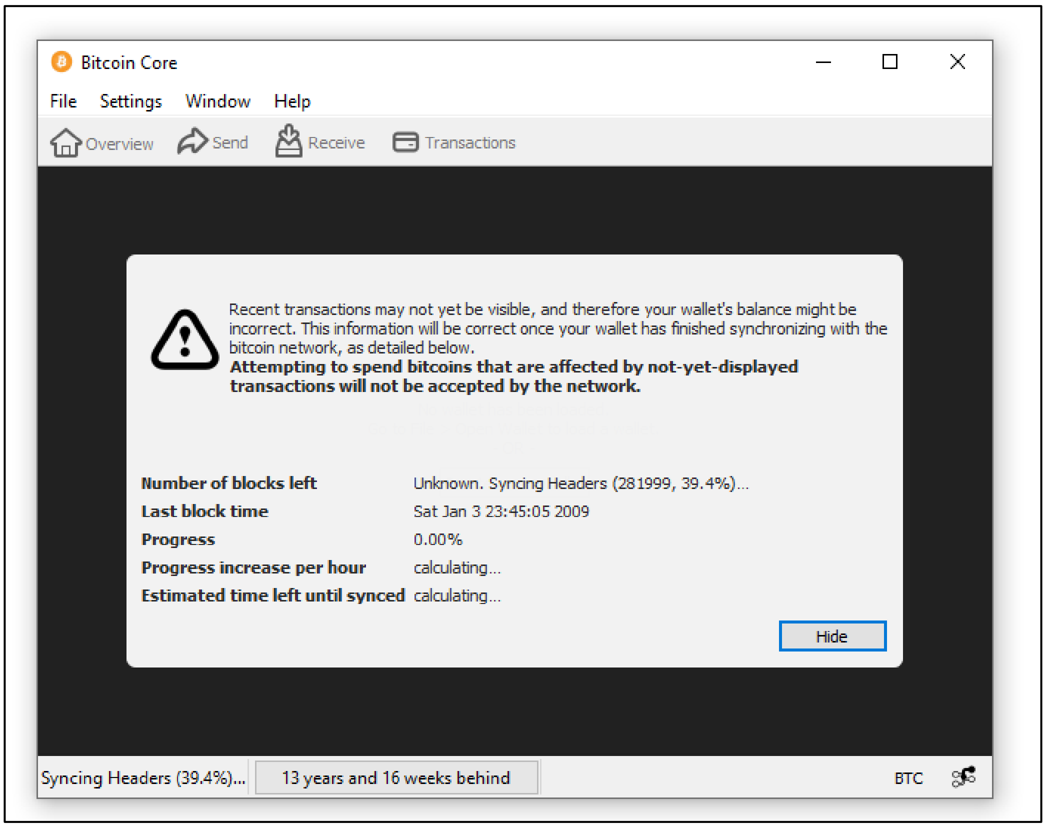
A new data directory will be created.

When you click OK, Bitcoin Core will begin to download and process the full Bitcoin block chain (420GB) starting with the earliest transactions in 2009 when Bitcoin initially launched.

This initial synchronisation is very demanding, and may expose hardware problems with your computer that had previously gone unnoticed. Each time you run Bitcoin Core, it will continue downloading where it left off.

If you have chosen to limit block chain storage (pruning), the historical data must still be downloaded and processed, but will be deleted afterward to keep your disk usage low.

☑ Limit block chain storage to 2 GB (sufficient to restore backups 6 days old)

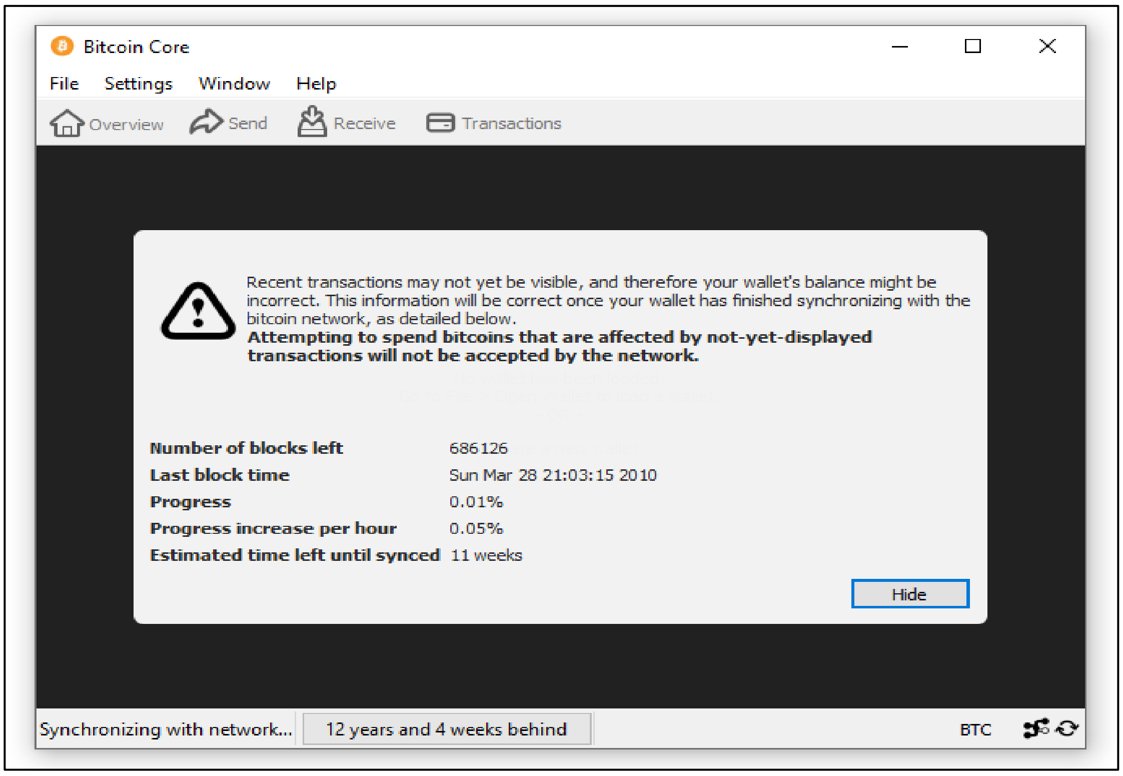


Recent transactions may not yet be visible, and therefore your wallet's balance might be incorrect. This information will be correct once your wallet has finished synchronizing with the bitcoin network, as detailed below.
Attempting to spend bitcoins that are affected by not-yet-displayed transactions will not be accepted by the network.

Number of blocks left	Unknown. Syncing Headers (281999, 39.4%)...
Last block time	Sat Jan 3 23:45:05 2009
Progress	0.00%
Progress increase per hour	calculating...
Estimated time left until synced	calculating...

Syncing Headers (39.4%)... 13 years and 16 weeks behind

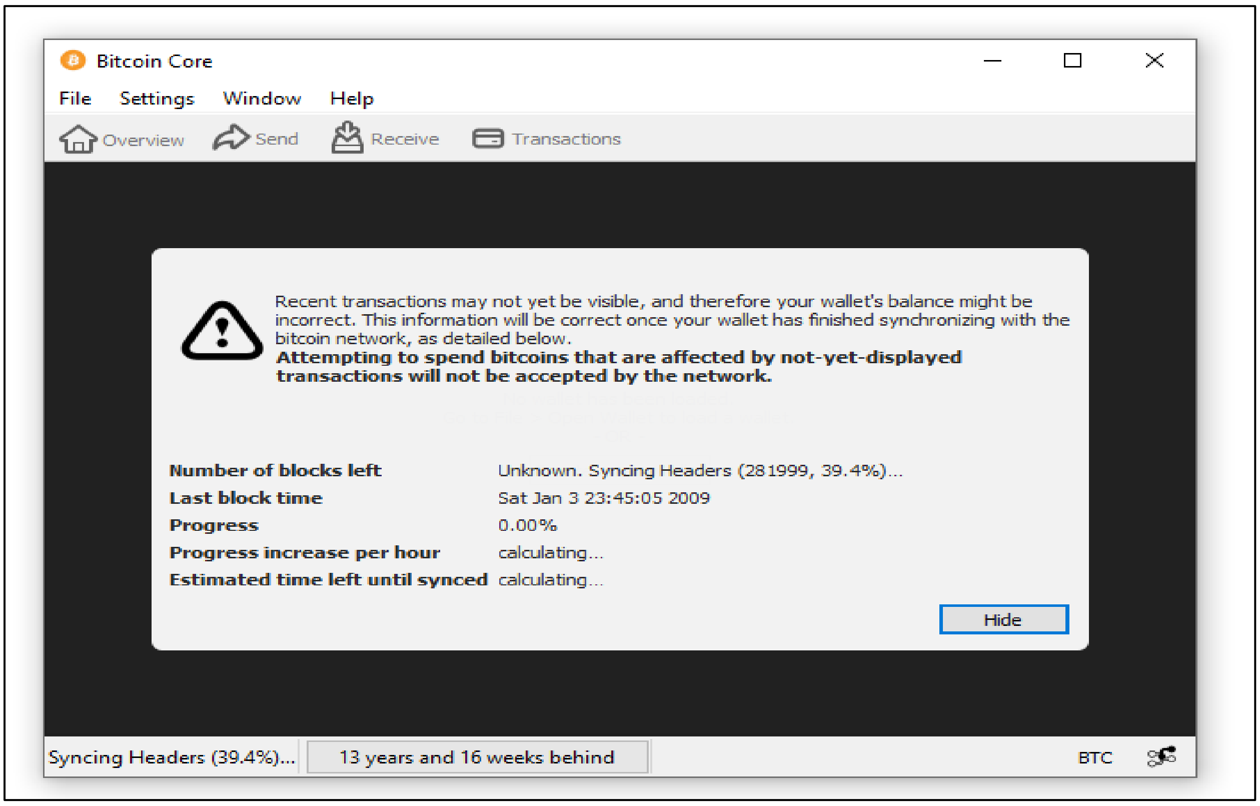
Step 7: Click on Hide button [Synchronization take place in background]



Recent transactions may not yet be visible, and therefore your wallet's balance might be incorrect. This information will be correct once your wallet has finished synchronizing with the bitcoin network, as detailed below.
Attempting to spend bitcoins that are affected by not-yet-displayed transactions will not be accepted by the network.

Number of blocks left	686126
Last block time	Sun Mar 28 21:03:15 2010
Progress	0.01%
Progress increase per hour	0.05%
Estimated time left until synced	11 weeks

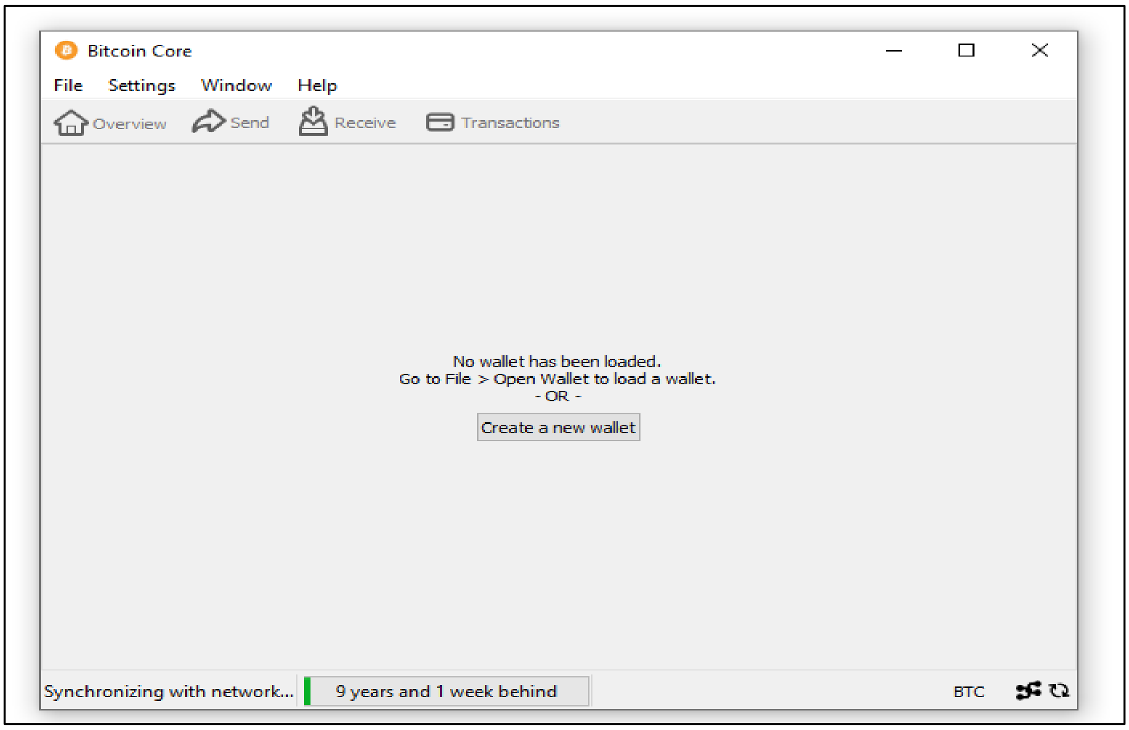
Synchronizing with network... 12 years and 4 weeks behind


Recent transactions may not yet be visible, and therefore your wallet's balance might be incorrect. This information will be correct once your wallet has finished synchronizing with the bitcoin network, as detailed below.
Attempting to spend bitcoins that are affected by not-yet-displayed transactions will not be accepted by the network.

Number of blocks left	Unknown. Syncing Headers (281999, 39.4%)...
Last block time	Sat Jan 3 23:45:05 2009
Progress	0.00%
Progress increase per hour	calculating...
Estimated time left until synced	calculating...

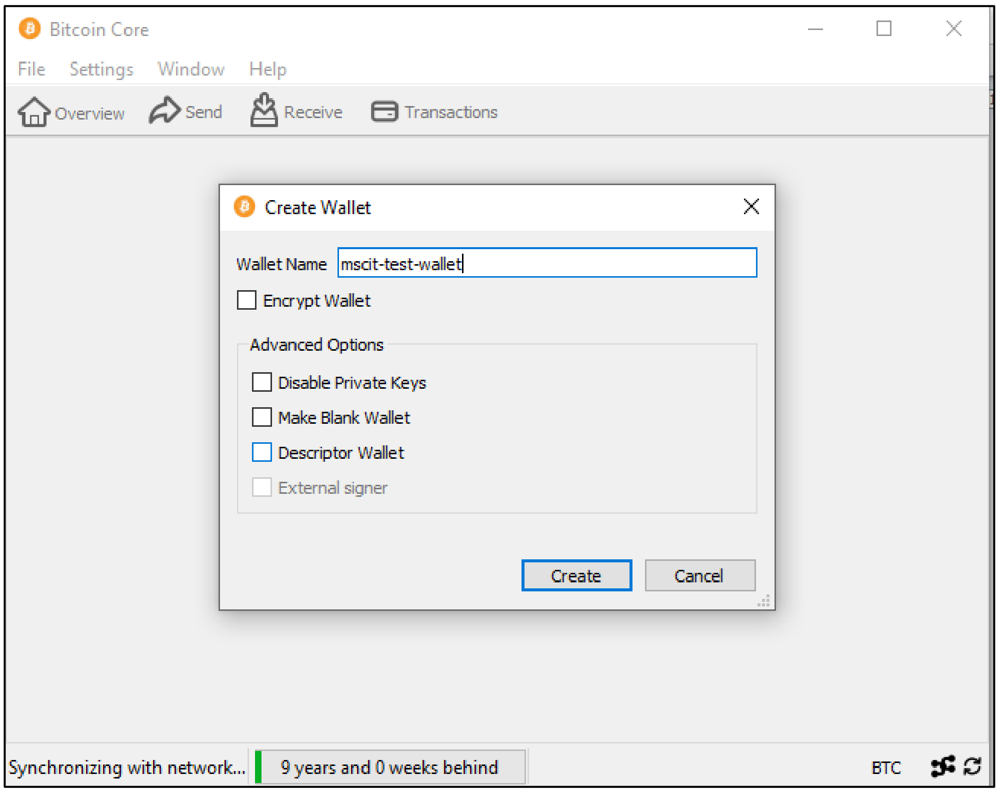
Syncing Headers (39.4%)... 13 years and 16 weeks behind

Step 8: You can create a wallet -> Create a new wallet


No wallet has been loaded.
Go to File > Open Wallet to load a wallet.
- OR -

Synchronizing with network... 9 years and 1 week behind

Step 9: Enter Wallet name



Wallet Name: mscit-test-wallet

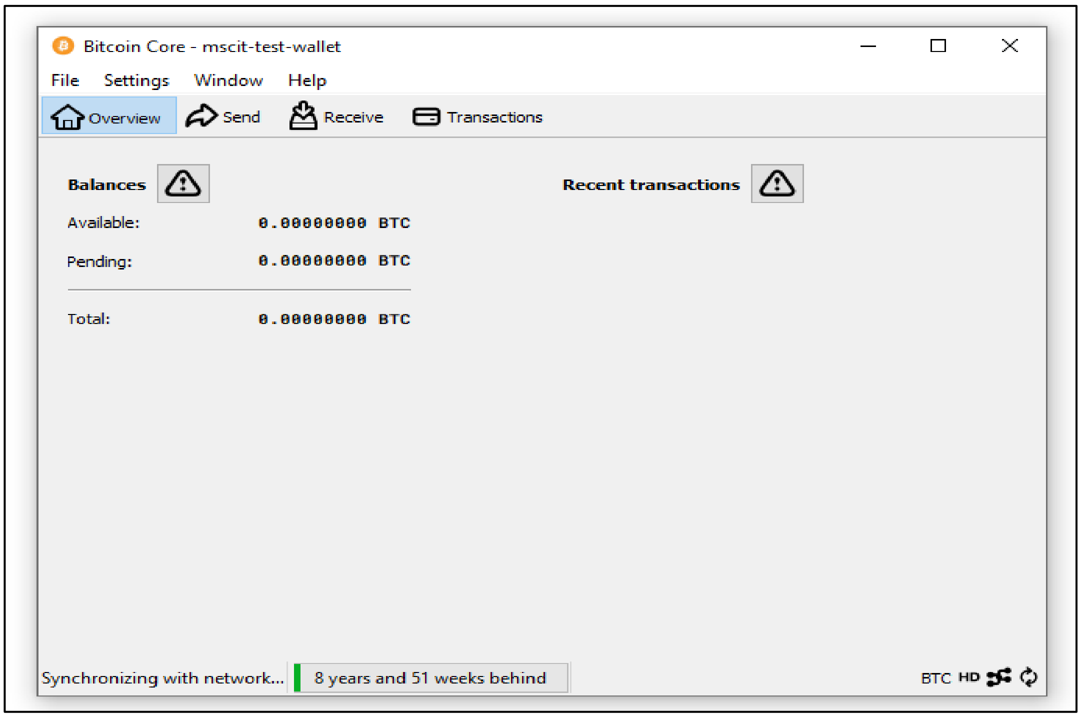
- [ ] Encrypt Wallet

Advanced Options

- [ ] Disable Private Keys
- [ ] Make Blank Wallet
- [ ] Descriptor Wallet
- [ ] External signer

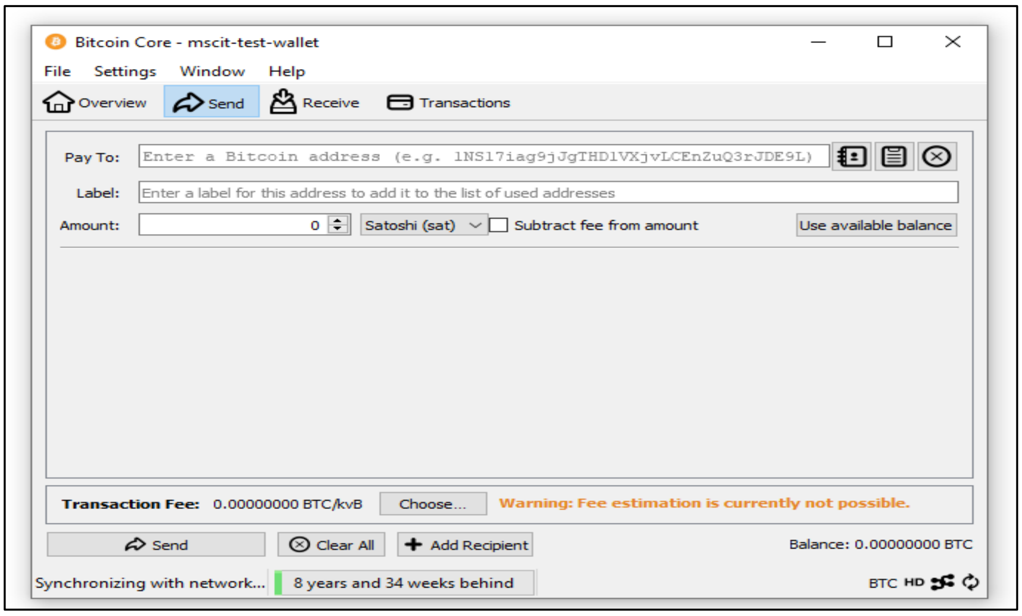
Synchronizing with network... 9 years and 0 weeks behind

Step 10: Finally, Account is setup


Bitcoin Core - mscit-test-wallet
File Settings Window Help
Overview Send Receive Transactions
Balances
Recent transactions
BTC HD

Available:	0.00000000 BTC
Pending:	0.00000000 BTC
Total:	0.00000000 BTC

Synchronizing with network... 8 years and 51 weeks behind


Bitcoin Core - mscit-test-wallet
File Settings Window Help
Overview Send Receive Transactions
Send Clear All Add Recipient
Choose...
Use available balance

Pay To: Enter a Bitcoin address (e.g. 1NS17iag9jJgTHD1VXjvLCEnZuQ3rJDE9L)

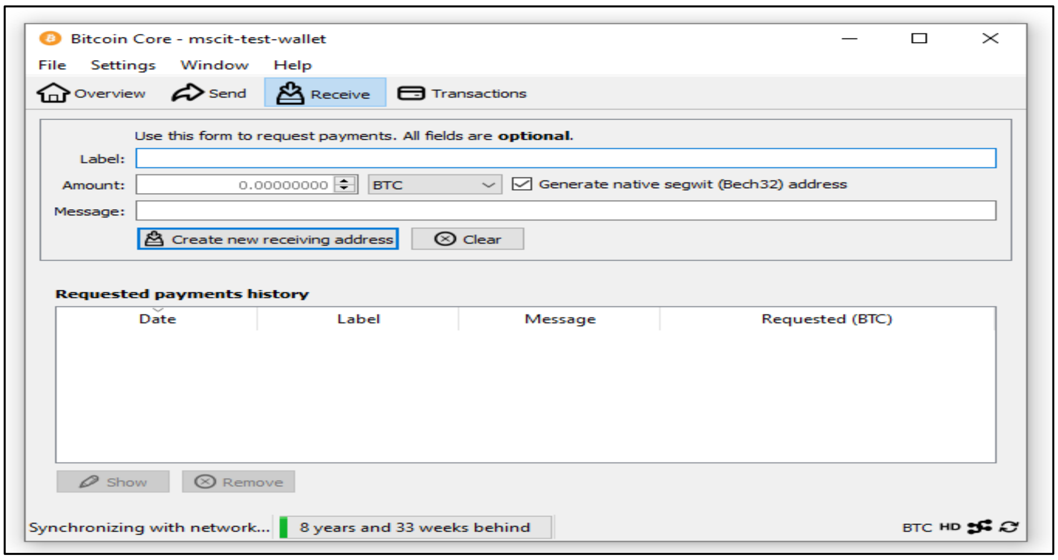
Label: Enter a label for this address to add it to the list of used addresses

Amount: 0 Satoshi (sat) ☐ Subtract fee from amount

Transaction Fee: 0.00000000 BTC/kvB Warning: Fee estimation is currently not possible.

Balance: 0.00000000 BTC

Synchronizing with network... 8 years and 34 weeks behind BTC HD


Bitcoin Core - mscit-test-wallet
File Settings Window Help
Overview Send Receive Transactions
Create new receiving address
Clear
Show
Remove

Use this form to request payments. All fields are **optional**.

Label:

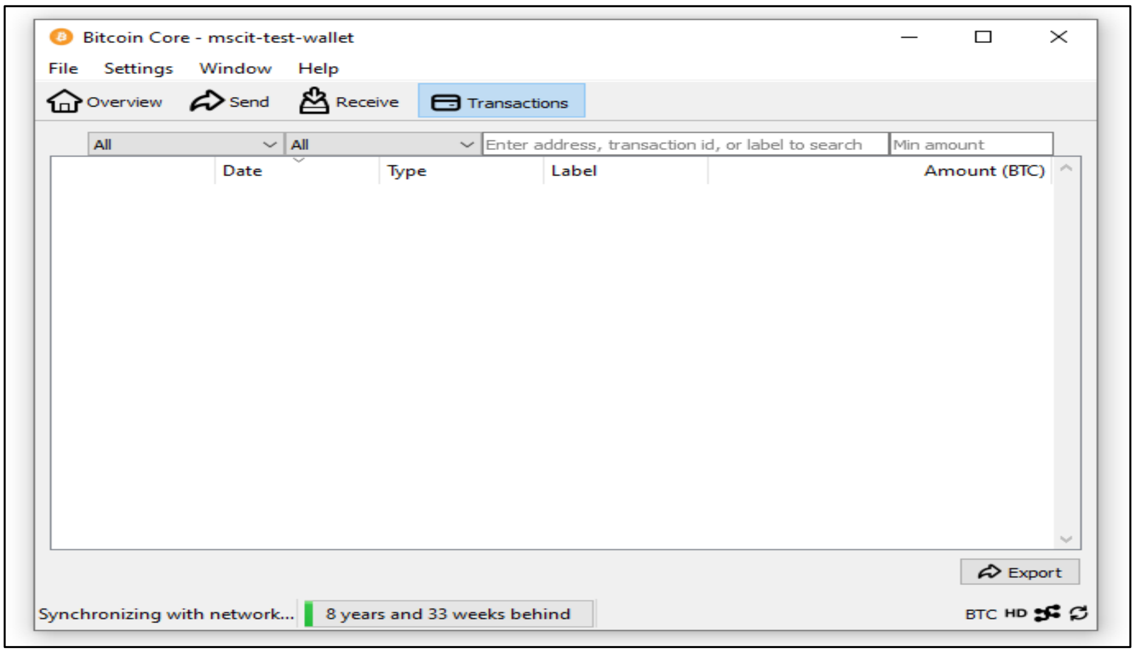
Amount: 0.00000000 BTC ☑ Generate native segwit (Bech32) address

Message:

Requested payments history

Date	Label	Message	Requested (BTC)

Synchronizing with network... 8 years and 33 weeks behind BTC HD


Bitcoin Core - mscit-test-wallet
File Settings Window Help
Overview Send Receive Transactions
Export

All | All | Enter address, transaction id, or label to search | Min amount

Date	Type	Label	Amount (BTC)

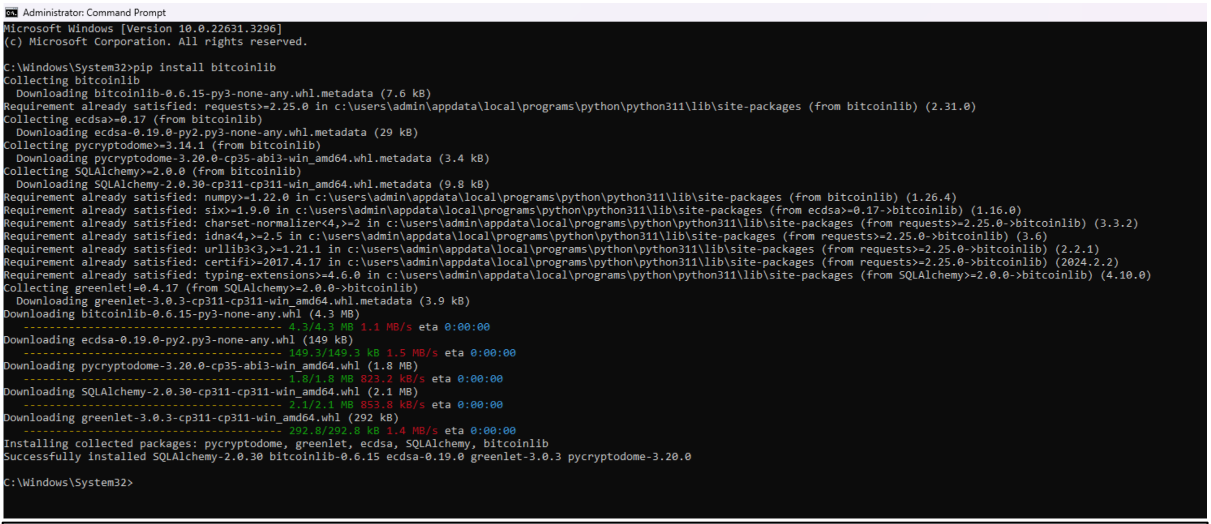
Synchronizing with network... 8 years and 33 weeks behind BTC HD

Practical 6

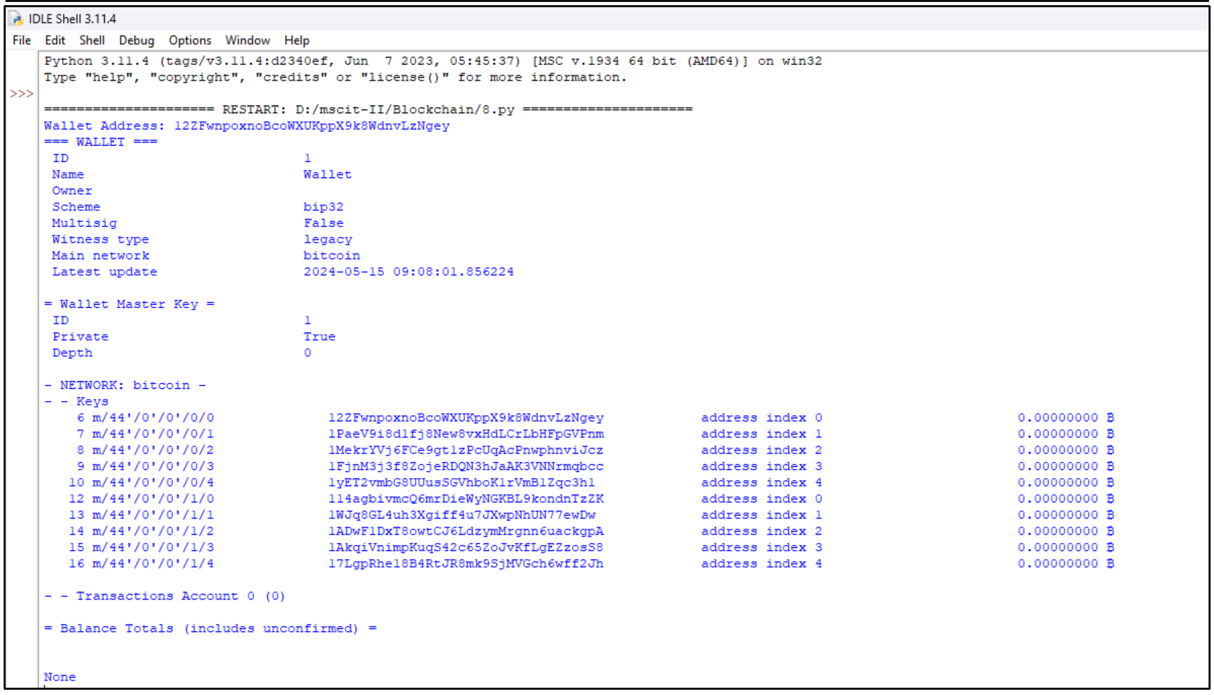
A. Demonstrate the use of Bitcoin Core API.

Code and Output:

```
!pip install bitcoin
from bitcoinlib.wallets import Wallet
w = Wallet.create('Wallet6')
key1 = w.get_key()
print('Wallet Address:',key1.address)
w.scan()
print(w.info())
```



```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.22631.3296]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32>pip install bitcoinlib
Collecting bitcoinlib
  Downloading bitcoinlib-0.6.15-py3-none-any.whl.metadata (7.6 kB)
Requirement already satisfied: requests>=2.25.0 in c:\users\admin\appdata\local\programs\python\python311\lib\site-packages (from bitcoinlib) (2.31.0)
Collecting ecdsa>=0.17 (from bitcoinlib)
  Downloading ecdsa-0.19.0-py2.py3-none-any.whl.metadata (29 kB)
Collecting pycryptodome>=3.14.1 (from bitcoinlib)
  Downloading pycryptodome-3.20.0-cp35-abi3-win_amd64.whl.metadata (3.4 kB)
Collecting SQLAlchemy>=2.0.0 (from bitcoinlib)
  Downloading SQLAlchemy-2.0.30-cp311-cp311-win_amd64.whl.metadata (9.8 kB)
Requirement already satisfied: numpy>=1.22.0 in c:\users\admin\appdata\local\programs\python\python311\lib\site-packages (from bitcoinlib) (1.26.4)
Requirement already satisfied: six>=1.9.0 in c:\users\admin\appdata\local\programs\python\python311\lib\site-packages (from ecdsa>=0.17->bitcoinlib) (1.16.0)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\admin\appdata\local\programs\python\python311\lib\site-packages (from requests>=2.25.0->bitcoinlib) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in c:\users\admin\appdata\local\programs\python\python311\lib\site-packages (from requests>=2.25.0->bitcoinlib) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\admin\appdata\local\programs\python\python311\lib\site-packages (from requests>=2.25.0->bitcoinlib) (2.2.1)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\admin\appdata\local\programs\python\python311\lib\site-packages (from requests>=2.25.0->bitcoinlib) (2024.2.2)
Requirement already satisfied: typing-extensions>=4.6.0 in c:\users\admin\appdata\local\programs\python\python311\lib\site-packages (from SQLAlchemy>=2.0.0->bitcoinlib) (4.10.0)
Collecting greenlet!=0.4.17 (from SQLAlchemy>=2.0.0->bitcoinlib)
  Downloading greenlet-3.0.3-cp311-cp311-win_amd64.whl.metadata (3.9 kB)
Downloading bitcoinlib-0.6.15-py3-none-any.whl (4.3 MB)
   ---------------------------------------- 4.3/4.3 MB 1.1 MB/s eta 0:00:00
Downloading ecdsa-0.19.0-py2.py3-none-any.whl (149 kB)
   ---------------------------------------- 149.3/149.3 kB 1.5 MB/s eta 0:00:00
Downloading pycryptodome-3.20.0-cp35-abi3-win_amd64.whl (1.8 MB)
   ---------------------------------------- 1.8/1.8 MB 823.2 kB/s eta 0:00:00
Downloading SQLAlchemy-2.0.30-cp311-cp311-win_amd64.whl (2.1 MB)
   ---------------------------------------- 2.1/2.1 MB 853.8 kB/s eta 0:00:00
Downloading greenlet-3.0.3-cp311-cp311-win_amd64.whl (292 kB)
   ---------------------------------------- 292.8/292.8 kB 1.4 MB/s eta 0:00:00
Installing collected packages: pycryptodome, greenlet, ecdsa, SQLAlchemy, bitcoinlib
Successfully installed SQLAlchemy-2.0.30 bitcoinlib-0.6.15 ecdsa-0.19.0 greenlet-3.0.3 pycryptodome-3.20.0

C:\Windows\System32>
```



```
Python 3.11.4 (tags/v3.11.4:d2340ef, Jun  7 2023, 05:45:37) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
==================== RESTART: D:/mscit-II/Blockchain/8.py ====================
Wallet Address: 12ZFwnpoxnoBcoWXUKppX9k8WdnvLzNgey
=== WALLET ===
 ID                             1
 Name                           Wallet
 Owner
 Scheme                         bip32
 Multisig                       False
 Witness type                   legacy
 Main network                   bitcoin
 Latest update                  2024-05-15 09:08:01.856224

= Wallet Master Key =
 ID                             1
 Private                        True
 Depth                          0

- NETWORK: bitcoin -
- - Keys
    6 m/44'/0'/0'/0/0            12ZFwnpoxnoBcoWXUKppX9k8WdnvLzNgey           address index 0                        0.00000000 ₿
    7 m/44'/0'/0'/0/1            1PaeV9i8dlfj8New8vxHdLCrLbHFpGVPnm           address index 1                        0.00000000 ₿
    8 m/44'/0'/0'/0/2            1MekrYVj6FCe9gtlzPcUqAcPnwphnviJcz           address index 2                        0.00000000 ₿
    9 m/44'/0'/0'/0/3            1FjnM3j3f8ZojeRDQN3hJaAK3VNNrmqbcc           address index 3                        0.00000000 ₿
   10 m/44'/0'/0'/0/4            1yET2vmbG8UUusSGVhboK1rVmBlZqc3hl            address index 4                        0.00000000 ₿
   12 m/44'/0'/0'/1/0            114agbivmcQ6mrDieWyNGKBL9kondnTzZK           address index 0                        0.00000000 ₿
   13 m/44'/0'/0'/1/1            1WJq8GL4uh3Xgiff4u7JXwpNhUN77ewDw            address index 1                        0.00000000 ₿
   14 m/44'/0'/0'/1/2            1ADwF1DxT8owtCJ6LdzymMrgnn6uackgpA           address index 2                        0.00000000 ₿
   15 m/44'/0'/0'/1/3            1AkqiVnimpKuqS42c65ZoJvKfLgEZzosS8           address index 3                        0.00000000 ₿
   16 m/44'/0'/0'/1/4            17LgpRhe18B4RtJR8mk9SjMVGch6wff2Jh           address index 4                        0.00000000 ₿

- - Transactions Account 0 (0)

= Balance Totals (includes unconfirmed) =


None
```

Practical 7

A. Create your own blockchain and demonstrate its use.

Steps:

1. Install VMWare Workstation

2. Download Linux Virtual Machine

3. Open Linux VM in VMWare

4. If creating new VM. Follow this command before starting step 5. Open new Terminal, on terminal type this command
>sudo add-apt-repository -y ppa:ethereum/ethereum

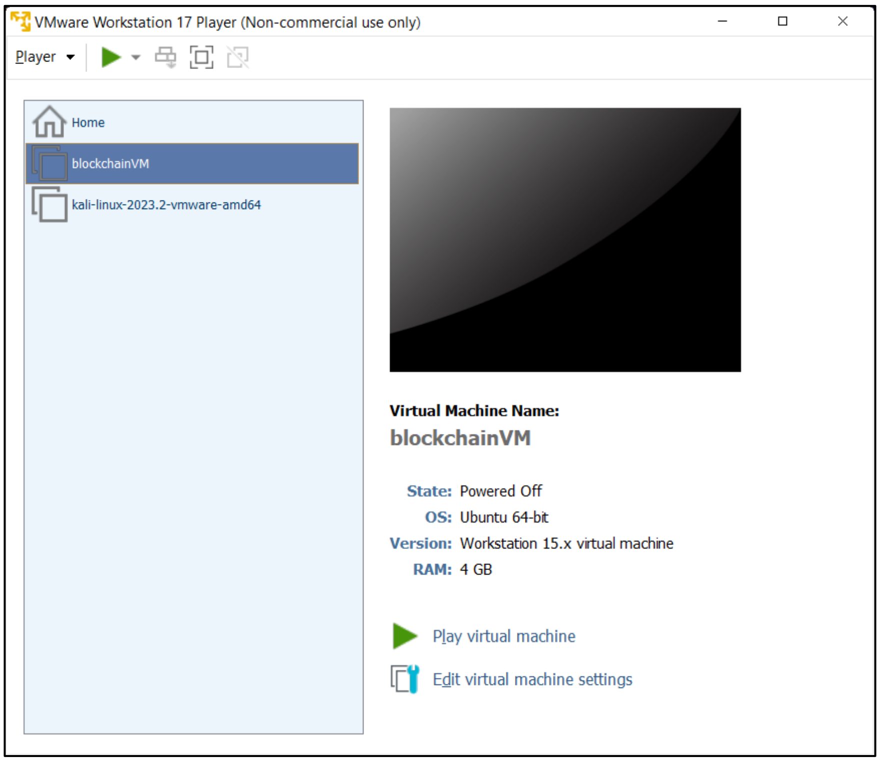
#if error encountered then run

# sudo apt-get install --reinstall ca-certificates

Install stable version of go-etherium

>sudo apt-get update

>sudo apt-get install ethereum


VMware Workstation 17 Player (Non-commercial use only)
Player
Home
blockchainVM
kali-linux-2023.2-vmware-amd64
Play virtual machine
Edit virtual machine settings

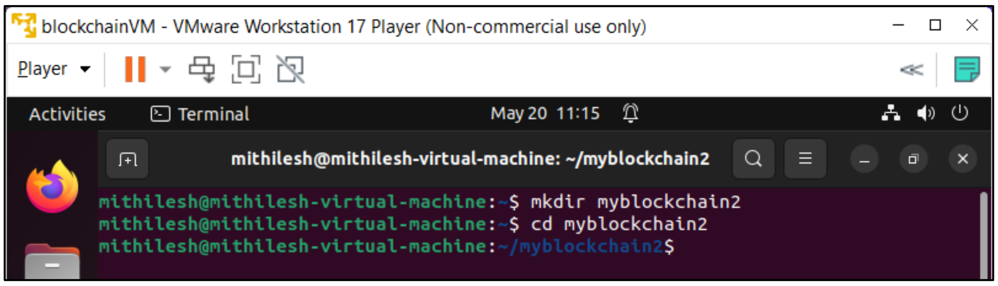
Virtual Machine Name:
blockchainVM

State: Powered Off
OS: Ubuntu 64-bit
Version: Workstation 15.x virtual machine
RAM: 4 GB

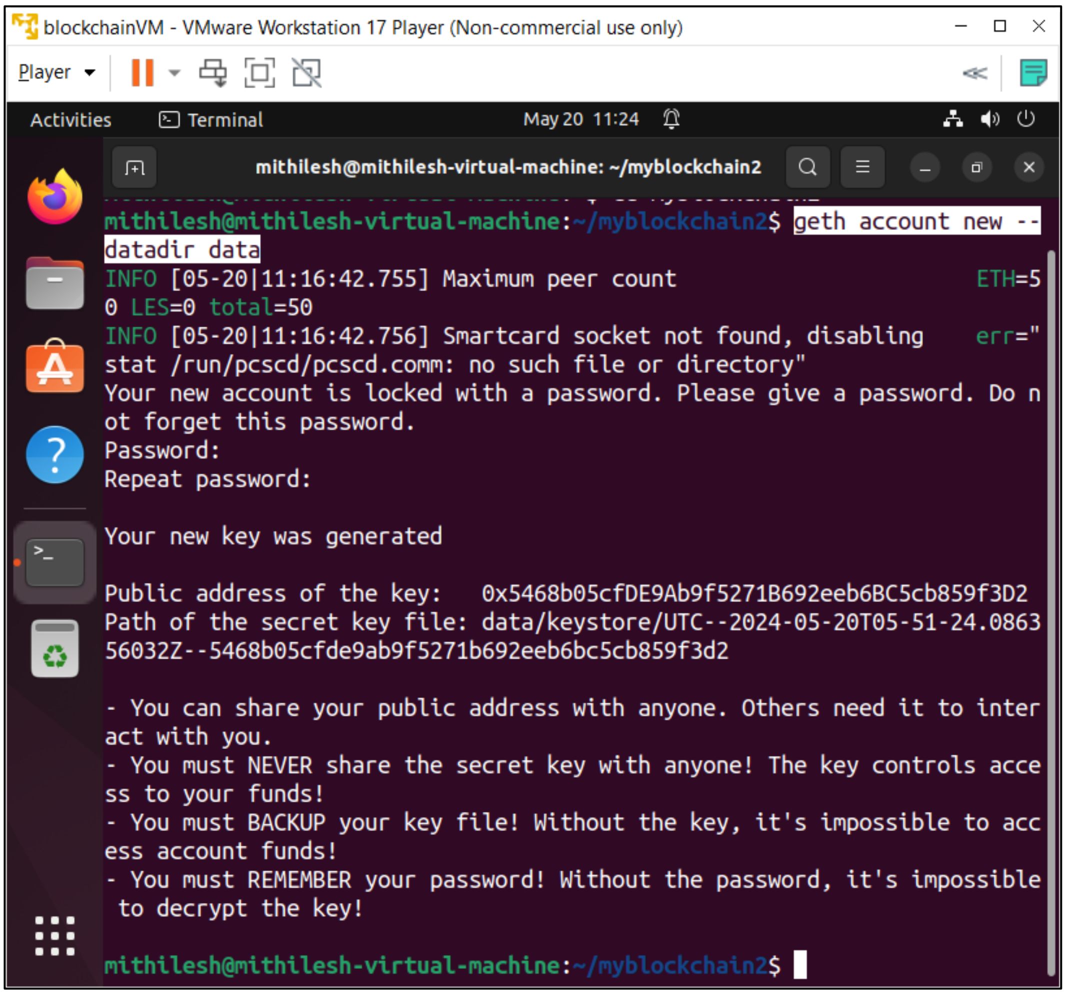
5. Open new Terminal and type this command for creating new directory for storing blockchain data: >**mkdir myblockchain**

>cd myblockchain

>geth account new --datadir data


blockchainVM - VMware Workstation 17 Player (Non-commercial use only)
Player
Activities
Terminal
May 20 11:15
mithilesh@mithilesh-virtual-machine: ~/myblockchain2

```
mithilesh@mithilesh-virtual-machine:~$ mkdir myblockchain2
mithilesh@mithilesh-virtual-machine:~$ cd myblockchain2
mithilesh@mithilesh-virtual-machine:~/myblockchain2$
```


blockchainVM - VMware Workstation 17 Player (Non-commercial use only)
Player
Activities
Terminal
May 20 11:24
mithilesh@mithilesh-virtual-machine: ~/myblockchain2

```
mithilesh@mithilesh-virtual-machine:~/myblockchain2$ geth account new --
datadir data
INFO [05-20|11:16:42.755] Maximum peer count                       ETH=5
0 LES=0 total=50
INFO [05-20|11:16:42.756] Smartcard socket not found, disabling    err="
stat /run/pcscd/pcscd.comm: no such file or directory"
Your new account is locked with a password. Please give a password. Do n
ot forget this password.
Password:
Repeat password:

Your new key was generated

Public address of the key:   0x5468b05cfDE9Ab9f5271B692eeb6BC5cb859f3D2
Path of the secret key file: data/keystore/UTC--2024-05-20T05-51-24.0863
56032Z--5468b05cfde9ab9f5271b692eeb6bc5cb859f3d2

- You can share your public address with anyone. Others need it to inter
act with you.
- You must NEVER share the secret key with anyone! The key controls acce
ss to your funds!
- You must BACKUP your key file! Without the key, it's impossible to acc
ess account funds!
- You must REMEMBER your password! Without the password, it's impossible
 to decrypt the key!

mithilesh@mithilesh-virtual-machine:~/myblockchain2$
```

6. Create genesis.json file

sudo nano genesis.json

```
{
"config": {
"chainId": 12345,
"homesteadBlock": 0,
"eip150Block": 0,
"eip155Block": 0,
"eip158Block": 0,
```

"byzantiumBlock": 0,

"constantinopleBlock": 0,

"petersburgBlock": 0,

"istanbulBlock": 0,

"berlinBlock": 0, "ethash": {}

},

"difficulty": "1",

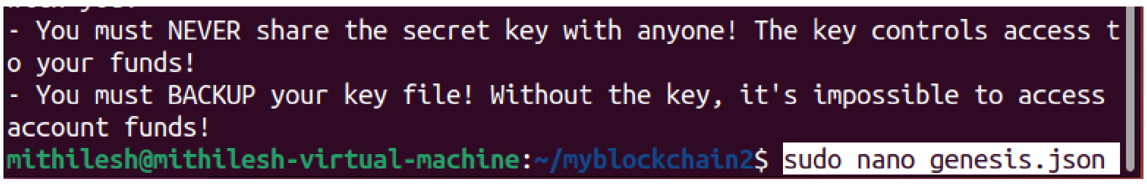
"gasLimit": "8000000", "alloc": {

"7df9a875a174b3bc565e6424a0050ebc1b2d1d82": { "balance": "300000" }, "Efaf4df069211972a7D2C3306d1F778a1603F10F": { "balance": "400000" }

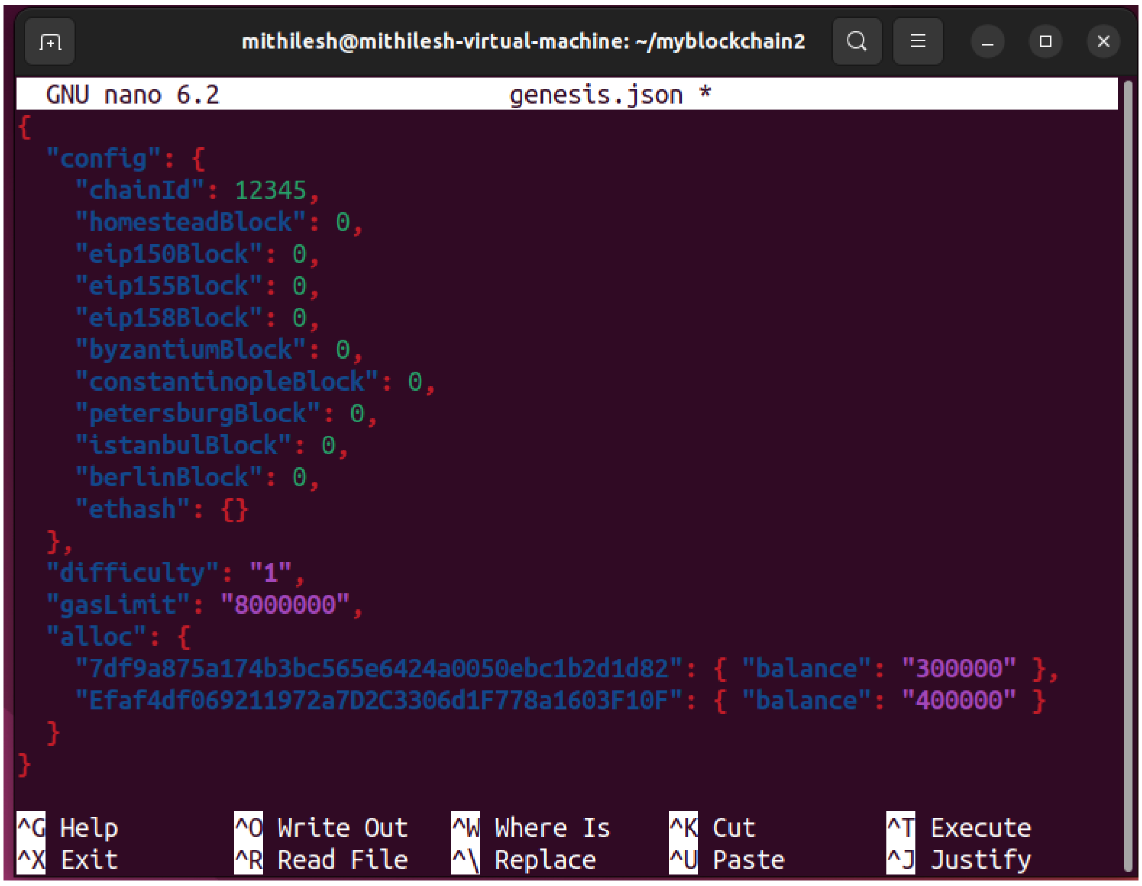
}

}

save the file -> “ctrl +o to write” and “{enter} save -> ctrl +x” to exit

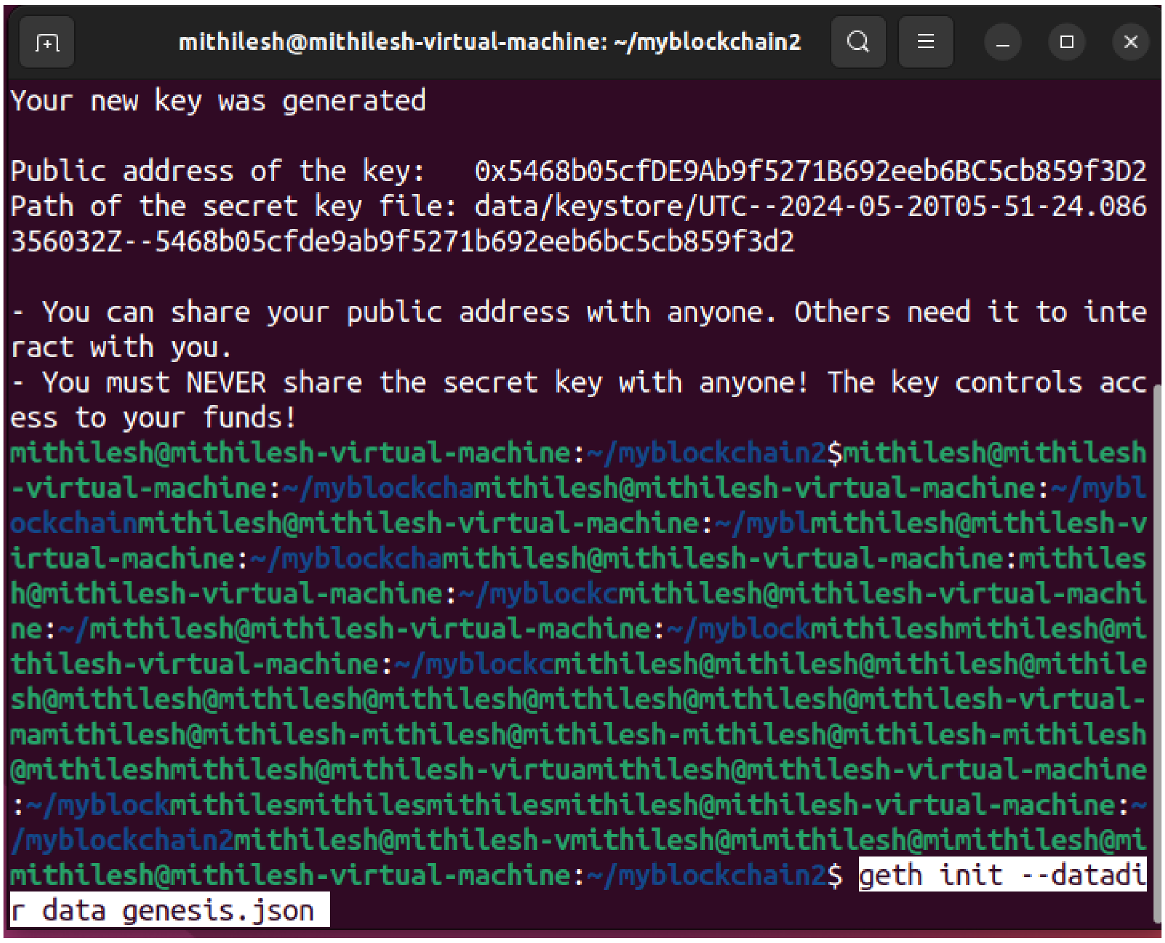


```
- You must NEVER share the secret key with anyone! The key controls access t
o your funds!
- You must BACKUP your key file! Without the key, it's impossible to access
account funds!
mithilesh@mithilesh-virtual-machine:~/myblockchain2$ sudo nano genesis.json
```

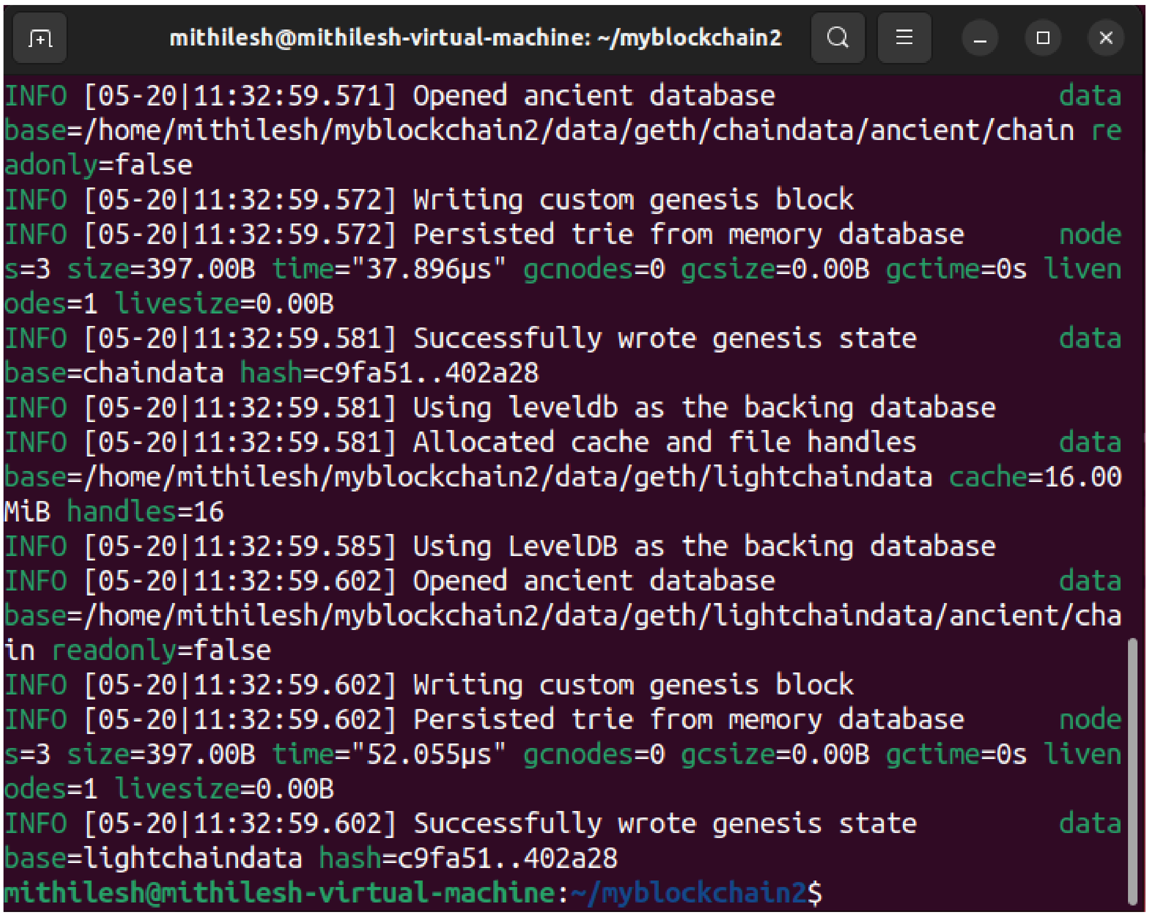


```
mithilesh@mithilesh-virtual-machine: ~/myblockchain2
  GNU nano 6.2                          genesis.json *
{
  "config": {
    "chainId": 12345,
    "homesteadBlock": 0,
    "eip150Block": 0,
    "eip155Block": 0,
    "eip158Block": 0,
    "byzantiumBlock": 0,
    "constantinopleBlock": 0,
    "petersburgBlock": 0,
    "istanbulBlock": 0,
    "berlinBlock": 0,
    "ethash": {}
  },
  "difficulty": "1",
  "gasLimit": "8000000",
  "alloc": {
    "7df9a875a174b3bc565e6424a0050ebc1b2d1d82": { "balance": "300000" },
    "Efaf4df069211972a7D2C3306d1F778a1603F10F": { "balance": "400000" }
  }
}

^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify
```

7. Initialize block “geth init --datadir data genesis.json”

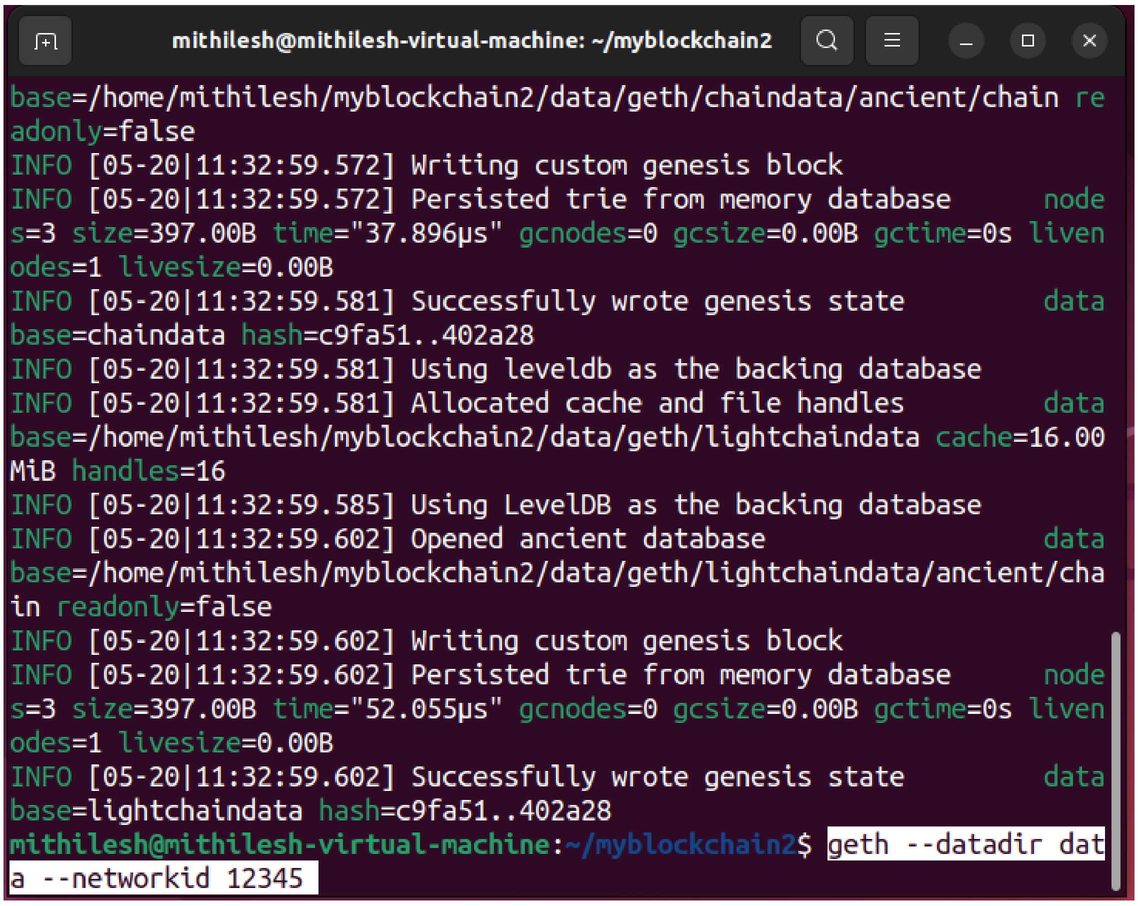


```
Your new key was generated

Public address of the key:   0x5468b05cfDE9Ab9f5271B692eeb6BC5cb859f3D2
Path of the secret key file: data/keystore/UTC--2024-05-20T05-51-24.086
356032Z--5468b05cfde9ab9f5271b692eeb6bc5cb859f3d2

- You can share your public address with anyone. Others need it to inte
ract with you.
- You must NEVER share the secret key with anyone! The key controls acc
ess to your funds!
mithilesh@mithilesh-virtual-machine:~/myblockchain2$mithilesh@mithilesh
-virtual-machine:~/myblockchamithilesh@mithilesh-virtual-machine:~/mybl
ockchainmithilesh@mithilesh-virtual-machine:~/myblmithilesh@mithilesh-v
irtual-machine:~/myblockchamithilesh@mithilesh-virtual-machine:mithiles
h@mithilesh-virtual-machine:~/myblockcmithilesh@mithilesh-virtual-machi
ne:~/mithilesh@mithilesh-virtual-machine:~/myblockmithileshmithilesh@mi
thilesh-virtual-machine:~/myblockcmithilesh@mithilesh@mithilesh@mithile
sh@mithilesh@mithilesh@mithilesh@mithilesh@mithilesh@mithilesh-virtual-
mamithilesh@mithilesh-mithilesh@mithilesh-mithilesh@mithilesh-mithilesh
@mithileshmithilesh@mithilesh-virtuamithilesh@mithilesh-virtual-machine
:~/myblockmithilesmithilesmithilesmithilesh@mithilesh-virtual-machine:~
/myblockchain2mithilesh@mithilesh-vmithilesh@mimithilesh@mimithilesh@mi
mithilesh@mithilesh-virtual-machine:~/myblockchain2$ geth init --datadi
r data genesis.json
```



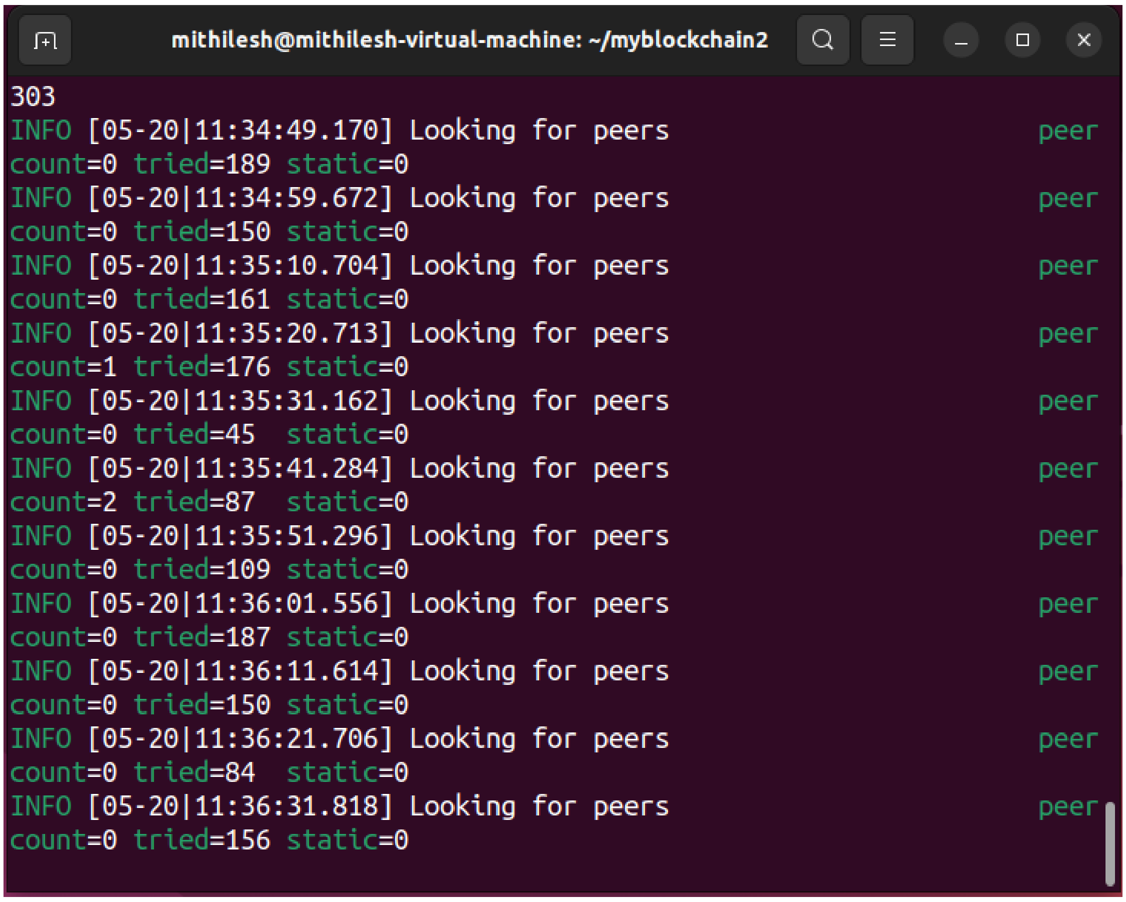
```
INFO [05-20|11:32:59.571] Opened ancient database                  data
base=/home/mithilesh/myblockchain2/data/geth/chaindata/ancient/chain re
adonly=false
INFO [05-20|11:32:59.572] Writing custom genesis block
INFO [05-20|11:32:59.572] Persisted trie from memory database      node
s=3 size=397.00B time="37.896µs" gcnodes=0 gcsize=0.00B gctime=0s liven
odes=1 livesize=0.00B
INFO [05-20|11:32:59.581] Successfully wrote genesis state         data
base=chaindata hash=c9fa51..402a28
INFO [05-20|11:32:59.581] Using leveldb as the backing database
INFO [05-20|11:32:59.581] Allocated cache and file handles         data
base=/home/mithilesh/myblockchain2/data/geth/lightchaindata cache=16.00
MiB handles=16
INFO [05-20|11:32:59.585] Using LevelDB as the backing database
INFO [05-20|11:32:59.602] Opened ancient database                  data
base=/home/mithilesh/myblockchain2/data/geth/lightchaindata/ancient/cha
in readonly=false
INFO [05-20|11:32:59.602] Writing custom genesis block
INFO [05-20|11:32:59.602] Persisted trie from memory database      node
s=3 size=397.00B time="52.055µs" gcnodes=0 gcsize=0.00B gctime=0s liven
odes=1 livesize=0.00B
INFO [05-20|11:32:59.602] Successfully wrote genesis state         data
base=lightchaindata hash=c9fa51..402a28
mithilesh@mithilesh-virtual-machine:~/myblockchain2$
```

8. Create Network "geth --datadir data --networkid 12345" and **do not close this terminal**



mithilesh@mithilesh-virtual-machine: ~/myblockchain2

```
base=/home/mithilesh/myblockchain2/data/geth/chaindata/ancient/chain re
adonly=false
INFO [05-20|11:32:59.572] Writing custom genesis block
INFO [05-20|11:32:59.572] Persisted trie from memory database      node
s=3 size=397.00B time="37.896µs" gcnodes=0 gcsize=0.00B gctime=0s liven
odes=1 livesize=0.00B
INFO [05-20|11:32:59.581] Successfully wrote genesis state         data
base=chaindata hash=c9fa51..402a28
INFO [05-20|11:32:59.581] Using leveldb as the backing database
INFO [05-20|11:32:59.581] Allocated cache and file handles         data
base=/home/mithilesh/myblockchain2/data/geth/lightchaindata cache=16.00
MiB handles=16
INFO [05-20|11:32:59.585] Using LevelDB as the backing database
INFO [05-20|11:32:59.602] Opened ancient database                  data
base=/home/mithilesh/myblockchain2/data/geth/lightchaindata/ancient/cha
in readonly=false
INFO [05-20|11:32:59.602] Writing custom genesis block
INFO [05-20|11:32:59.602] Persisted trie from memory database      node
s=3 size=397.00B time="52.055µs" gcnodes=0 gcsize=0.00B gctime=0s liven
odes=1 livesize=0.00B
INFO [05-20|11:32:59.602] Successfully wrote genesis state         data
base=lightchaindata hash=c9fa51..402a28
mithilesh@mithilesh-virtual-machine:~/myblockchain2$ geth --datadir dat
a --networkid 12345
```



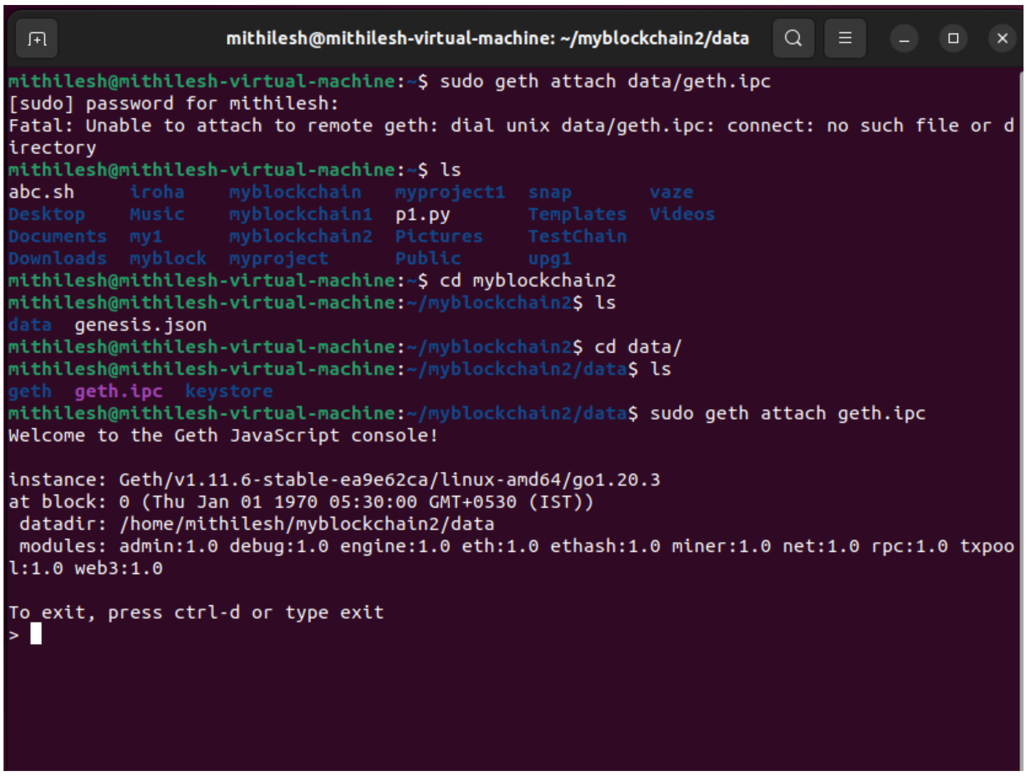
mithilesh@mithilesh-virtual-machine: ~/myblockchain2

```
303
INFO [05-20|11:34:49.170] Looking for peers                        peer
count=0 tried=189 static=0
INFO [05-20|11:34:59.672] Looking for peers                        peer
count=0 tried=150 static=0
INFO [05-20|11:35:10.704] Looking for peers                        peer
count=0 tried=161 static=0
INFO [05-20|11:35:20.713] Looking for peers                        peer
count=1 tried=176 static=0
INFO [05-20|11:35:31.162] Looking for peers                        peer
count=0 tried=45  static=0
INFO [05-20|11:35:41.284] Looking for peers                        peer
count=2 tried=87  static=0
INFO [05-20|11:35:51.296] Looking for peers                        peer
count=0 tried=109 static=0
INFO [05-20|11:36:01.556] Looking for peers                        peer
count=0 tried=187 static=0
INFO [05-20|11:36:11.614] Looking for peers                        peer
count=0 tried=150 static=0
INFO [05-20|11:36:21.706] Looking for peers                        peer
count=0 tried=84  static=0
INFO [05-20|11:36:31.818] Looking for peers                        peer
count=0 tried=156 static=0
```

9. Open new terminal

“sudo geth attach data/geth.ipc eth.getBalance(eth.accounts[0]) miner.setEtherbase(eth.accounts[0])”

“miner.start() admin.addPeer(admin.nodeInfo.enode) eth.getBalance(eth.accounts[0])”

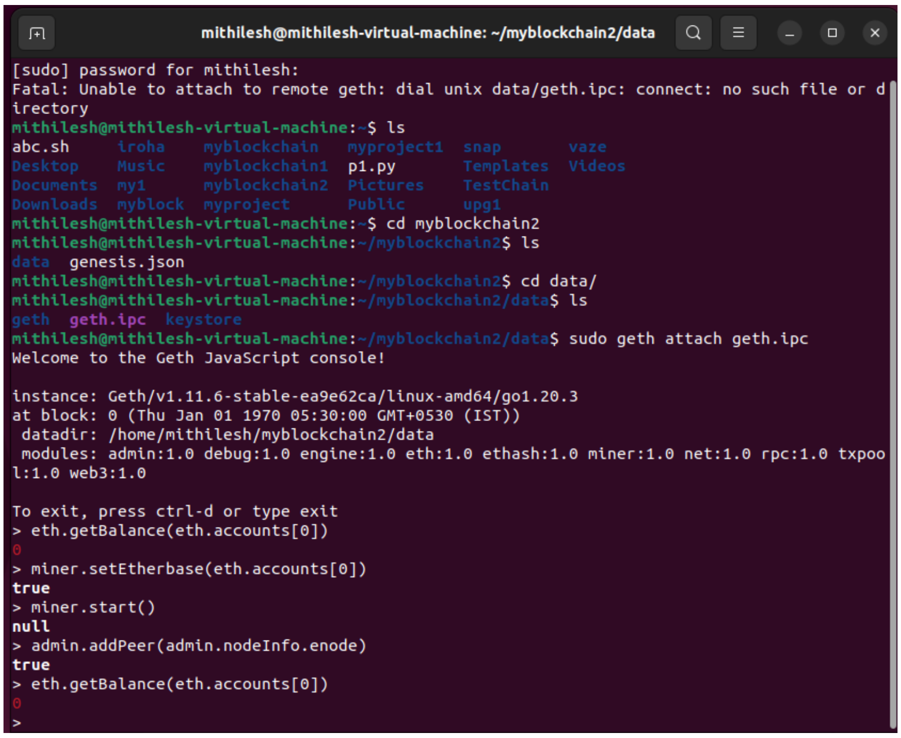


```
mithilesh@mithilesh-virtual-machine: ~/myblockchain2/data
mithilesh@mithilesh-virtual-machine:~$ sudo geth attach data/geth.ipc
[sudo] password for mithilesh:
Fatal: Unable to attach to remote geth: dial unix data/geth.ipc: connect: no such file or d
irectory
mithilesh@mithilesh-virtual-machine:~$ ls
abc.sh     iroha    myblockchain   myproject1  snap       vaze
Desktop    Music    myblockchain1  p1.py       Templates  Videos
Documents  my1      myblockchain2  Pictures    TestChain
Downloads  myblock  myproject      Public      upg1
mithilesh@mithilesh-virtual-machine:~$ cd myblockchain2
mithilesh@mithilesh-virtual-machine:~/myblockchain2$ ls
data  genesis.json
mithilesh@mithilesh-virtual-machine:~/myblockchain2$ cd data/
mithilesh@mithilesh-virtual-machine:~/myblockchain2/data$ ls
geth  geth.ipc  keystore
mithilesh@mithilesh-virtual-machine:~/myblockchain2/data$ sudo geth attach geth.ipc
Welcome to the Geth JavaScript console!

instance: Geth/v1.11.6-stable-ea9e62ca/linux-amd64/go1.20.3
at block: 0 (Thu Jan 01 1970 05:30:00 GMT+0530 (IST))
 datadir: /home/mithilesh/myblockchain2/data
 modules: admin:1.0 debug:1.0 engine:1.0 eth:1.0 ethash:1.0 miner:1.0 net:1.0 rpc:1.0 txpoo
l:1.0 web3:1.0

To exit, press ctrl-d or type exit
> 
```

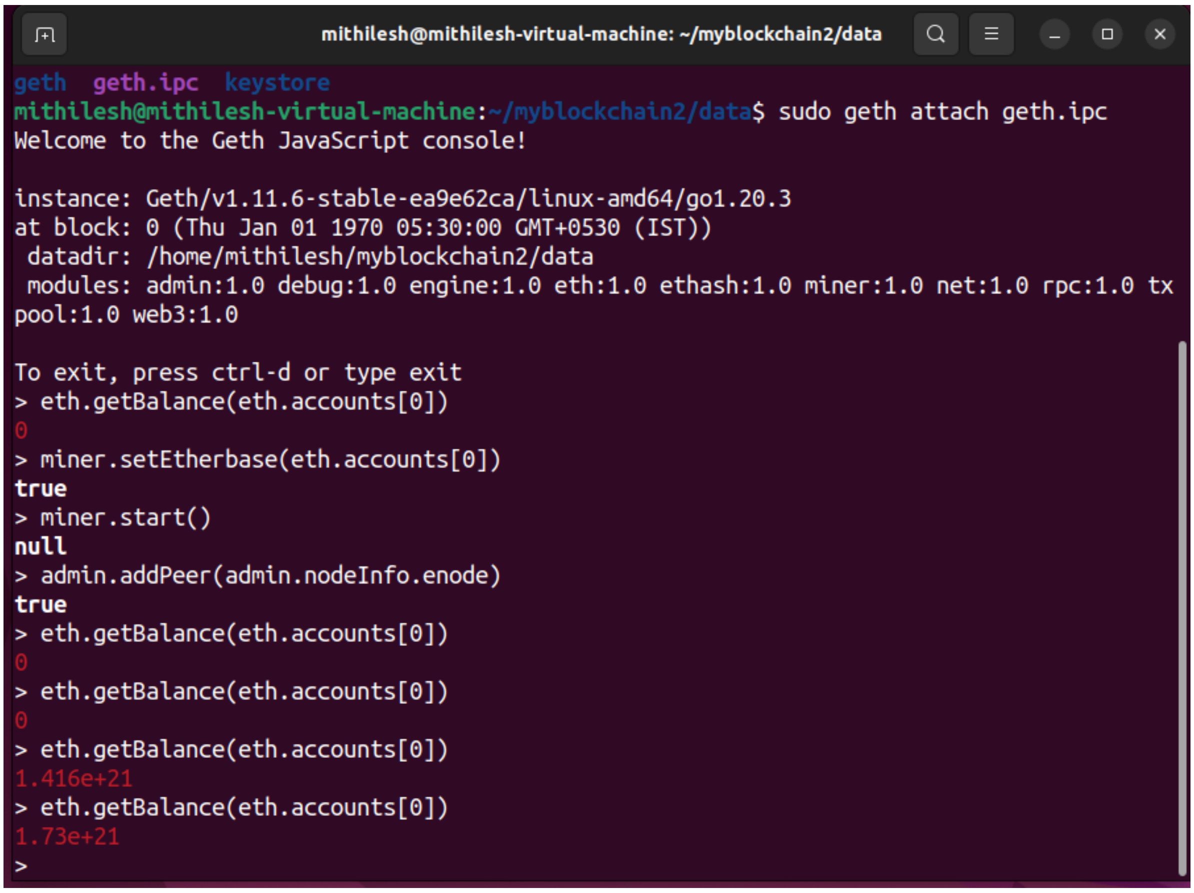
10. **Wait for 10-20 minutes and check balance “eth.getBalance(eth.accounts[0])”**

After balance is updated, you can check current block height

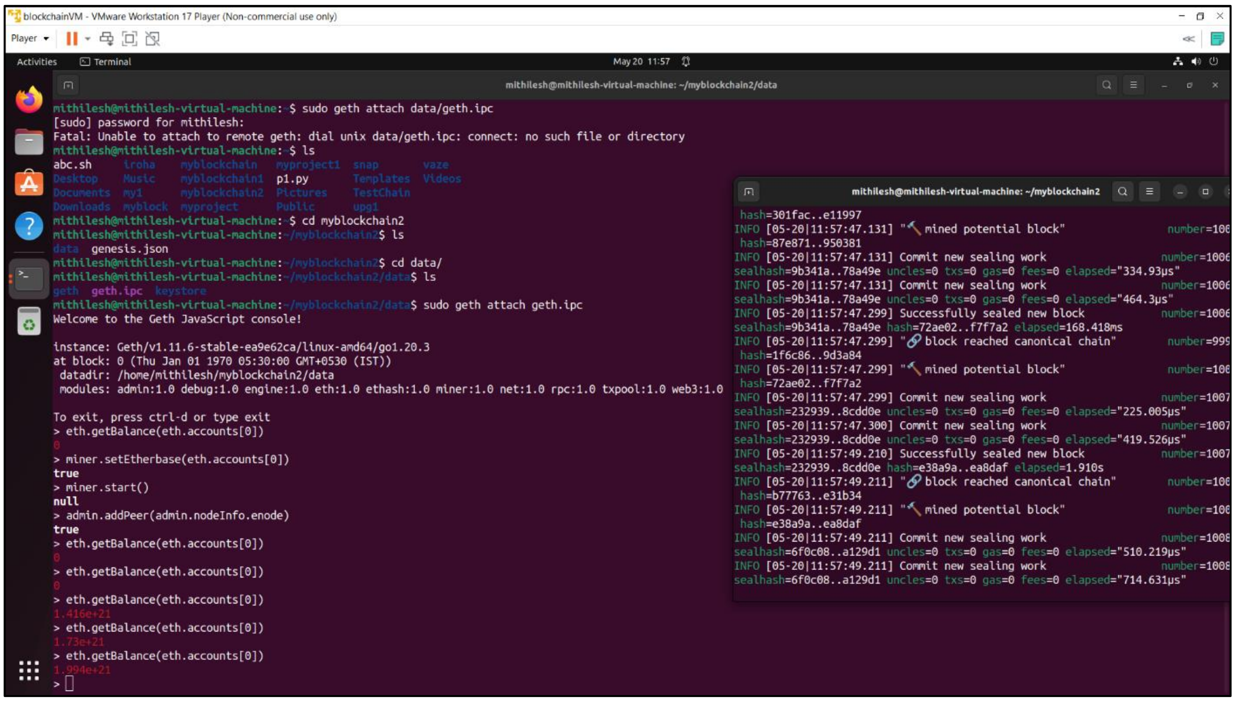
>eth.blockNumber



```
mithilesh@mithilesh-virtual-machine: ~/myblockchain2/data
[sudo] password for mithilesh:
Fatal: Unable to attach to remote geth: dial unix data/geth.ipc: connect: no such file or d
irectory
mithilesh@mithilesh-virtual-machine:~$ ls
abc.sh     iroha    myblockchain   myproject1  snap       vaze
Desktop    Music    myblockchain1  p1.py       Templates  Videos
Documents  my1      myblockchain2  Pictures    TestChain
Downloads  myblock  myproject      Public      upg1
mithilesh@mithilesh-virtual-machine:~$ cd myblockchain2
mithilesh@mithilesh-virtual-machine:~/myblockchain2$ ls
data  genesis.json
mithilesh@mithilesh-virtual-machine:~/myblockchain2$ cd data/
mithilesh@mithilesh-virtual-machine:~/myblockchain2/data$ ls
geth  geth.ipc  keystore
mithilesh@mithilesh-virtual-machine:~/myblockchain2/data$ sudo geth attach geth.ipc
Welcome to the Geth JavaScript console!

instance: Geth/v1.11.6-stable-ea9e62ca/linux-amd64/go1.20.3
at block: 0 (Thu Jan 01 1970 05:30:00 GMT+0530 (IST))
 datadir: /home/mithilesh/myblockchain2/data
 modules: admin:1.0 debug:1.0 engine:1.0 eth:1.0 ethash:1.0 miner:1.0 net:1.0 rpc:1.0 txpoo
l:1.0 web3:1.0

To exit, press ctrl-d or type exit
> eth.getBalance(eth.accounts[0])
0
> miner.setEtherbase(eth.accounts[0])
true
> miner.start()
null
> admin.addPeer(admin.nodeInfo.enode)
true
> eth.getBalance(eth.accounts[0])
0
>
```



```
mithilesh@mithilesh-virtual-machine: ~/myblockchain2/data
geth  geth.ipc  keystore
mithilesh@mithilesh-virtual-machine:~/myblockchain2/data$ sudo geth attach geth.ipc
Welcome to the Geth JavaScript console!

instance: Geth/v1.11.6-stable-ea9e62ca/linux-amd64/go1.20.3
at block: 0 (Thu Jan 01 1970 05:30:00 GMT+0530 (IST))
 datadir: /home/mithilesh/myblockchain2/data
 modules: admin:1.0 debug:1.0 engine:1.0 eth:1.0 ethash:1.0 miner:1.0 net:1.0 rpc:1.0 tx
pool:1.0 web3:1.0

To exit, press ctrl-d or type exit
> eth.getBalance(eth.accounts[0])
0
> miner.setEtherbase(eth.accounts[0])
true
> miner.start()
null
> admin.addPeer(admin.nodeInfo.enode)
true
> eth.getBalance(eth.accounts[0])
0
> eth.getBalance(eth.accounts[0])
0
> eth.getBalance(eth.accounts[0])
1.416e+21
> eth.getBalance(eth.accounts[0])
1.73e+21
>
```



```
blockchainVM - VMware Workstation 17 Player (Non-commercial use only)
Activities   Terminal                May 20 11:57
mithilesh@mithilesh-virtual-machine: ~/myblockchain2/data
mithilesh@mithilesh-virtual-machine:~$ sudo geth attach data/geth.ipc
[sudo] password for mithilesh:
Fatal: Unable to attach to remote geth: dial unix data/geth.ipc: connect: no such file or directory
mithilesh@mithilesh-virtual-machine:~$ ls
abc.sh     iroha     myblockchain    myproject1  snap       vaze
Desktop    Music     myblockchain1   p1.py       Templates  Videos
Documents  my1       myblockchain2   Pictures    TestChain
Downloads  myblock   myproject       Public      upg1
mithilesh@mithilesh-virtual-machine:~$ cd myblockchain2
mithilesh@mithilesh-virtual-machine:~/myblockchain2$ ls
data  genesis.json
mithilesh@mithilesh-virtual-machine:~/myblockchain2$ cd data/
mithilesh@mithilesh-virtual-machine:~/myblockchain2/data$ ls
geth  geth.ipc  keystore
mithilesh@mithilesh-virtual-machine:~/myblockchain2/data$ sudo geth attach geth.ipc
Welcome to the Geth JavaScript console!

instance: Geth/v1.11.6-stable-ea9e62ca/linux-amd64/go1.20.3
at block: 0 (Thu Jan 01 1970 05:30:00 GMT+0530 (IST))
 datadir: /home/mithilesh/myblockchain2/data
 modules: admin:1.0 debug:1.0 engine:1.0 eth:1.0 ethash:1.0 miner:1.0 net:1.0 rpc:1.0 txpool:1.0 web3:1.0

To exit, press ctrl-d or type exit
> eth.getBalance(eth.accounts[0])
0
> miner.setEtherbase(eth.accounts[0])
true
> miner.start()
null
> admin.addPeer(admin.nodeInfo.enode)
true
> eth.getBalance(eth.accounts[0])
0
> eth.getBalance(eth.accounts[0])
0
> eth.getBalance(eth.accounts[0])
1.416e+21
> eth.getBalance(eth.accounts[0])
1.73e+21
> eth.getBalance(eth.accounts[0])
1.994e+21
>
```

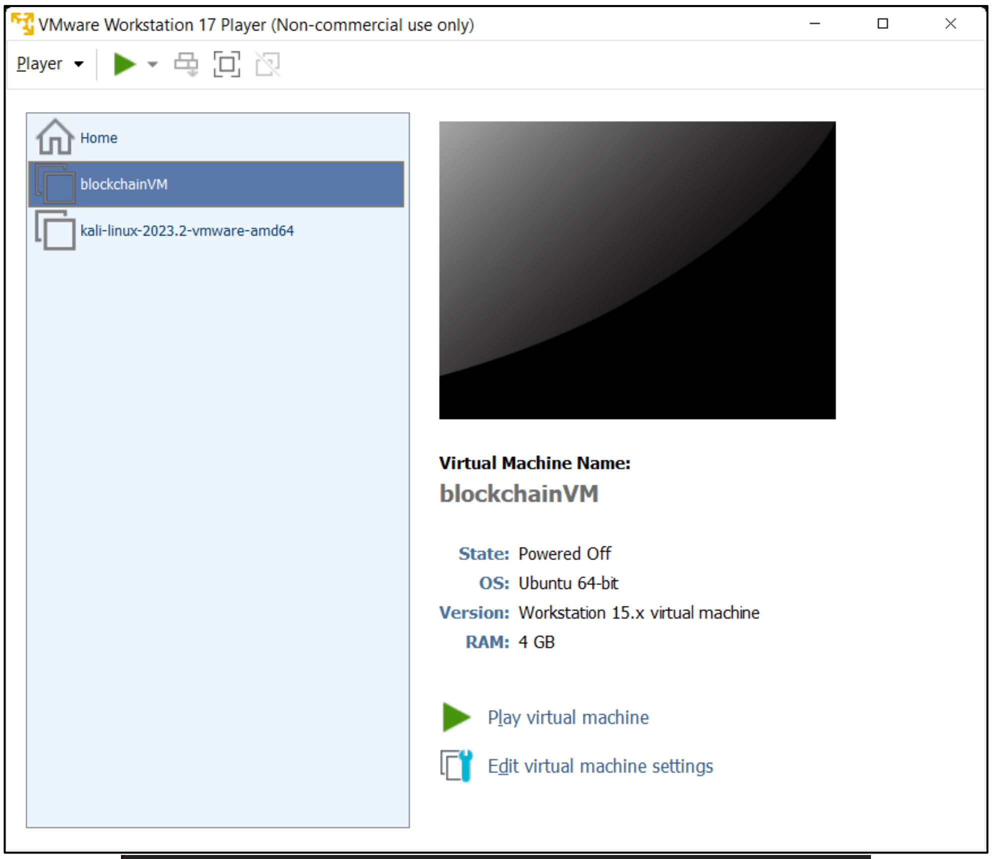
```
mithilesh@mithilesh-virtual-machine: ~/myblockchain2
 hash=301fac..e11997
INFO [05-20|11:57:47.131] "🔨 mined potential block"                number=100
 hash=87e871..950381
INFO [05-20|11:57:47.131] Commit new sealing work                  number=1006
sealhash=9b341a..78a49e uncles=0 txs=0 gas=0 fees=0 elapsed="334.93µs"
INFO [05-20|11:57:47.131] Commit new sealing work                  number=1006
sealhash=9b341a..78a49e uncles=0 txs=0 gas=0 fees=0 elapsed="464.3µs"
INFO [05-20|11:57:47.299] Successfully sealed new block            number=1006
sealhash=9b341a..78a49e hash=72ae02..f7f7a2 elapsed=168.418ms
INFO [05-20|11:57:47.299] "🔗 block reached canonical chain"         number=999
 hash=1f6c86..9d3a84
INFO [05-20|11:57:47.299] "🔨 mined potential block"                number=100
 hash=72ae02..f7f7a2
INFO [05-20|11:57:47.299] Commit new sealing work                  number=1007
sealhash=232939..8cdd0e uncles=0 txs=0 gas=0 fees=0 elapsed="225.005µs"
INFO [05-20|11:57:47.300] Commit new sealing work                  number=1007
sealhash=232939..8cdd0e uncles=0 txs=0 gas=0 fees=0 elapsed="419.526µs"
INFO [05-20|11:57:49.210] Successfully sealed new block            number=1007
sealhash=232939..8cdd0e hash=e38a9a..ea8daf elapsed=1.910s
INFO [05-20|11:57:49.211] "🔗 block reached canonical chain"         number=100
 hash=b77763..e31b34
INFO [05-20|11:57:49.211] "🔨 mined potential block"                number=100
 hash=e38a9a..ea8daf
INFO [05-20|11:57:49.211] Commit new sealing work                  number=1008
sealhash=6f0c08..a129d1 uncles=0 txs=0 gas=0 fees=0 elapsed="510.219µs"
INFO [05-20|11:57:49.211] Commit new sealing work                  number=1008
sealhash=6f0c08..a129d1 uncles=0 txs=0 gas=0 fees=0 elapsed="714.631µs"
```

Practical 8

A. Build D-Apps with angular.

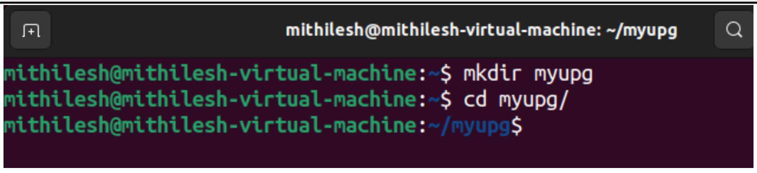
Step 1:Install the required package –on new terminal 1 type these commands
sudo apt-get -y install curl git vim build-essential sudo apt-get install curl software-properties-common sudo apt install npm
sudo npm install -g web3 sudo apt-get install nodejs sudo apt install python3.9
curl -sL https://deb.nodesource.com/setup_10.x | sudo bash - sudo npm install --global node-sass@latest
sudo npm install -g truffle@latest sudo npm install -g ganache-cli
export NODE_OPTIONS=--openssl-legacy-provider
To update npm>sudo npm cache clean -f sudo npm install -g n sudo n latest

Step 2: Start from step 2 if you have VM configured. Create a new directory
mkdir myupg
cd myupg

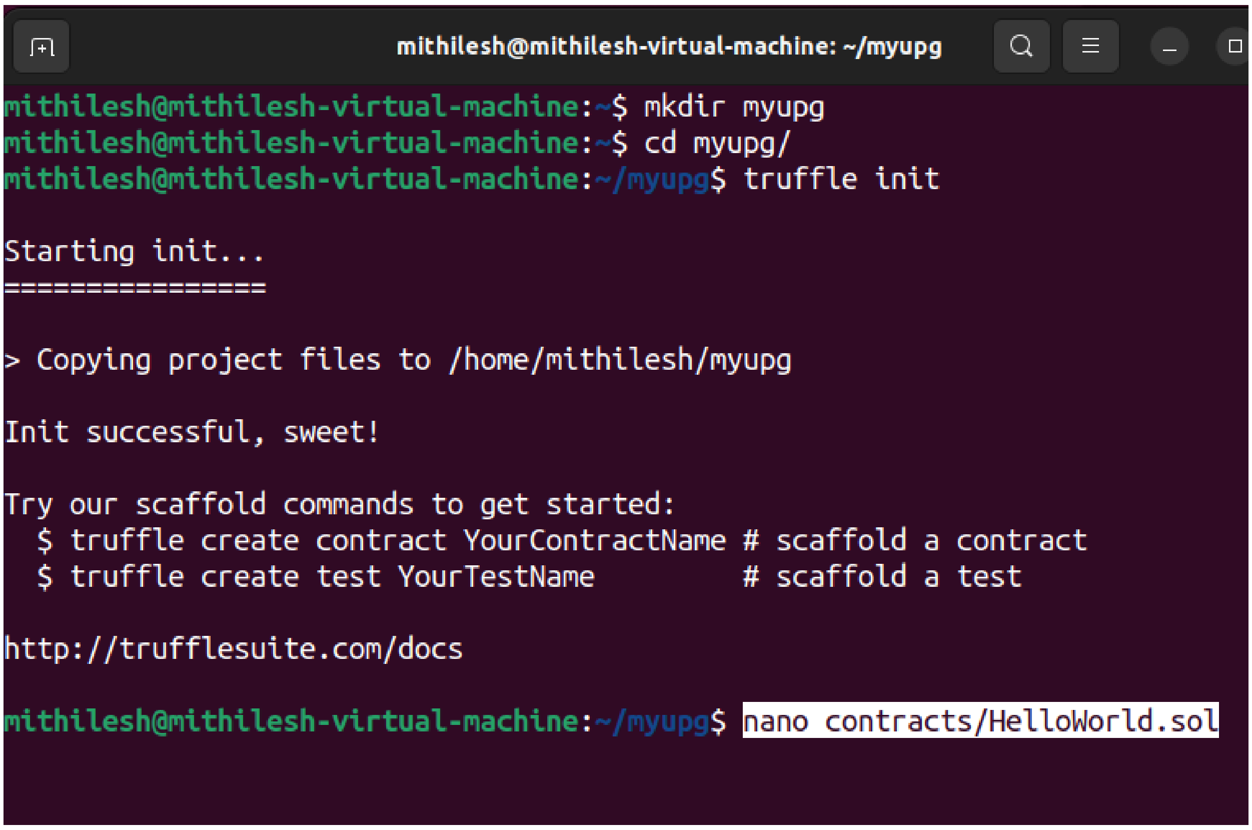

VMware Workstation 17 Player (Non-commercial use only)

Virtual Machine Name:
blockchainVM

State: Powered Off
OS: Ubuntu 64-bit
Version: Workstation 15.x virtual machine
RAM: 4 GB

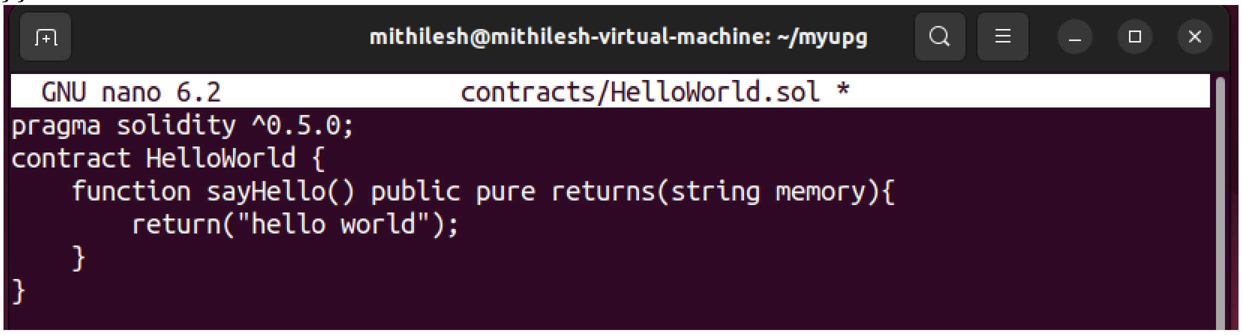

```
mithilesh@mithilesh-virtual-machine: ~/myupg
mithilesh@mithilesh-virtual-machine:~$ mkdir myupg
mithilesh@mithilesh-virtual-machine:~$ cd myupg/
mithilesh@mithilesh-virtual-machine:~/myupg$
```

Step 3: Initialize the project folder
truffle init
Step 4: Now create a new contract
nano contracts/HelloWorld.sol



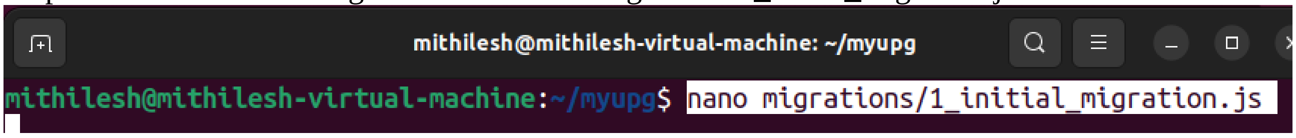
```
mithilesh@mithilesh-virtual-machine: ~/myupg
mithilesh@mithilesh-virtual-machine:~$ mkdir myupg
mithilesh@mithilesh-virtual-machine:~$ cd myupg/
mithilesh@mithilesh-virtual-machine:~/myupg$ truffle init

Starting init...
================

> Copying project files to /home/mithilesh/myupg

Init successful, sweet!

Try our scaffold commands to get started:
  $ truffle create contract YourContractName # scaffold a contract
  $ truffle create test YourTestName         # scaffold a test

http://trufflesuite.com/docs

mithilesh@mithilesh-virtual-machine:~/myupg$ nano contracts/HelloWorld.sol
```

Step 5:Add the following code in HelloWorld.sol pragma solidity ^0.5.0;
contract HelloWorld {
function sayHello() public pure returns(string memory){ return("hello world");
}}



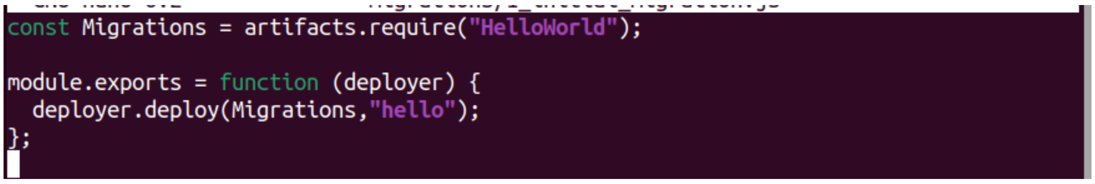
```
mithilesh@mithilesh-virtual-machine: ~/myupg
  GNU nano 6.2                  contracts/HelloWorld.sol *
pragma solidity ^0.5.0;
contract HelloWorld {
    function sayHello() public pure returns(string memory){
        return("hello world");
    }
}
```

Step 6: create default configuration file: nano migrations/1_initial_migration.js



```
mithilesh@mithilesh-virtual-machine: ~/myupg
mithilesh@mithilesh-virtual-machine:~/myupg$ nano migrations/1_initial_migration.js
```

Step 7: Edit this line in the file
const Migrations = artifacts.require("HelloWorld"); module.exports = function (deployer) { deployer.deploy(Migrations,"hello");
};

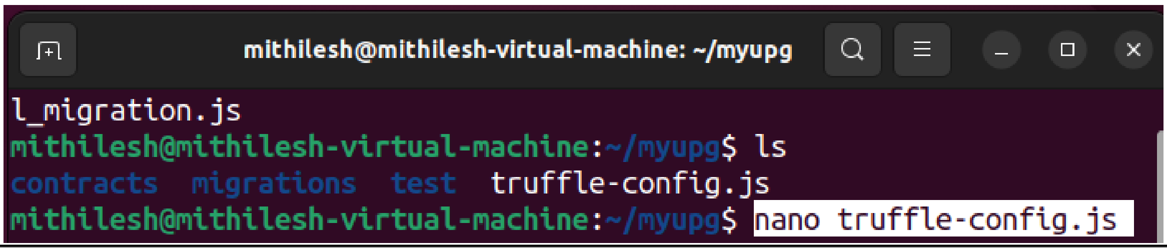


```
const Migrations = artifacts.require("HelloWorld");

module.exports = function (deployer) {
  deployer.deploy(Migrations,"hello");
};
```

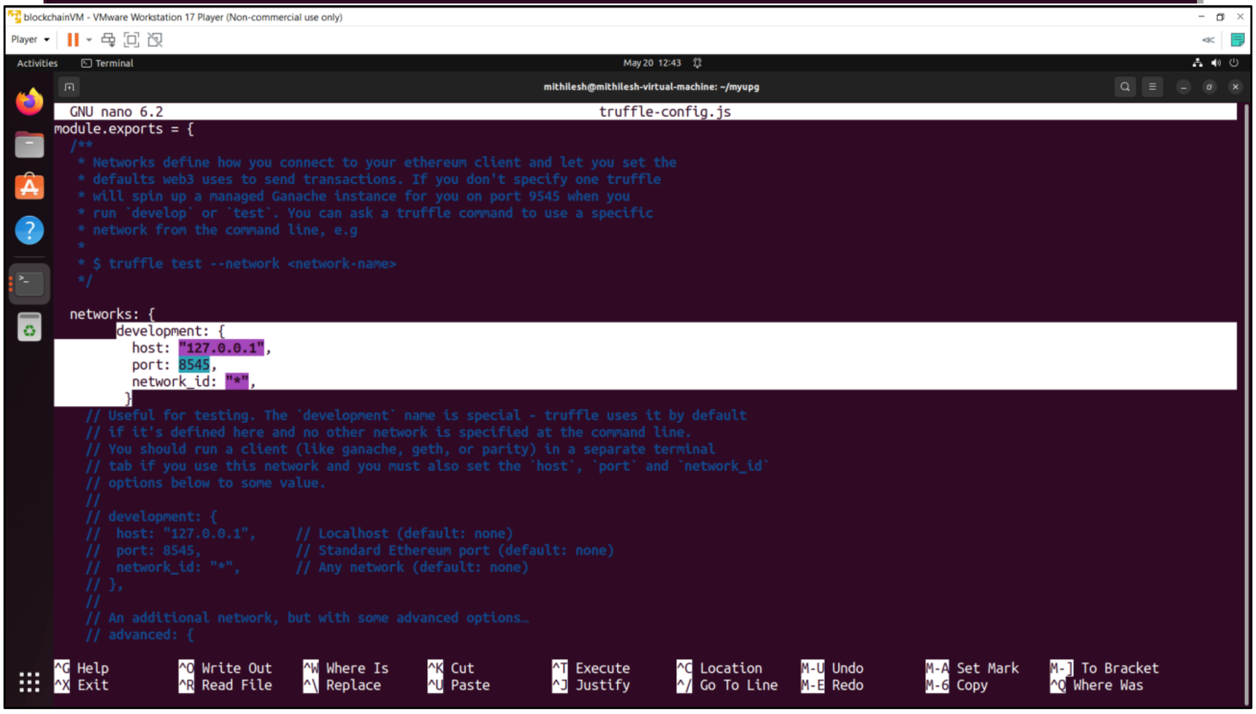
Step 8: Edit network configuration file

sudo nano truffle-config.js

Remove all line(press CTRL +K) from the file and add the following lines
###########################
module.exports = { networks: { development: {
host: "127.0.0.1",
port: 8545, network_id: "*",
}
}}

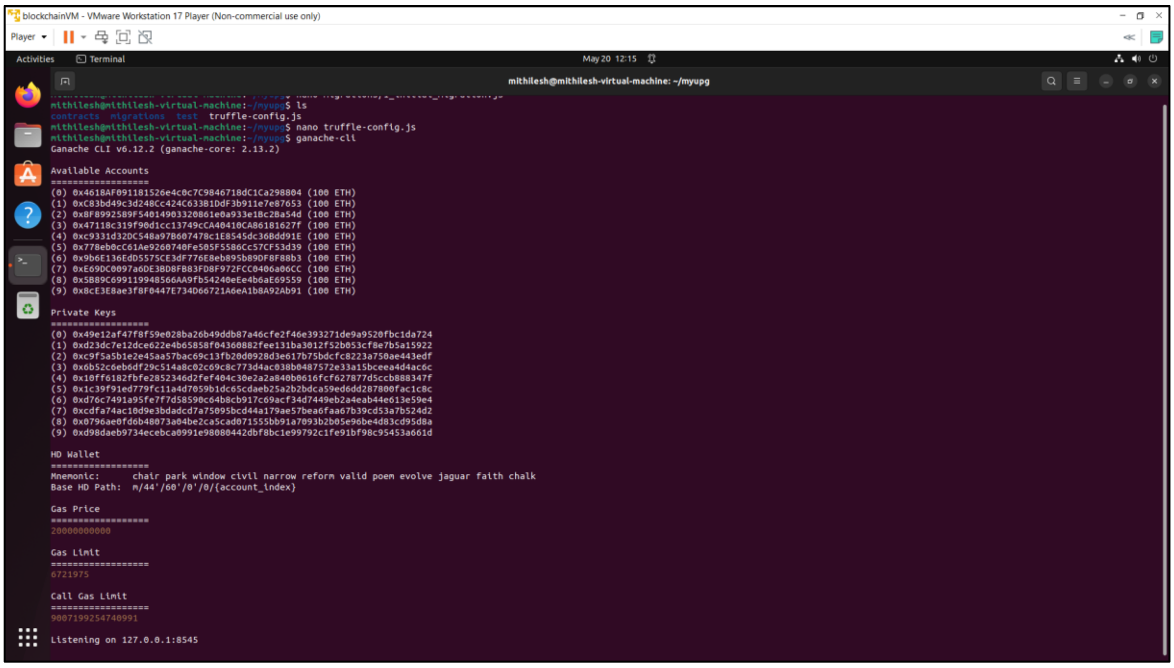


```
l_migration.js
mithilesh@mithilesh-virtual-machine:~/myupg$ ls
contracts  migrations  test  truffle-config.js
mithilesh@mithilesh-virtual-machine:~/myupg$ nano truffle-config.js
```



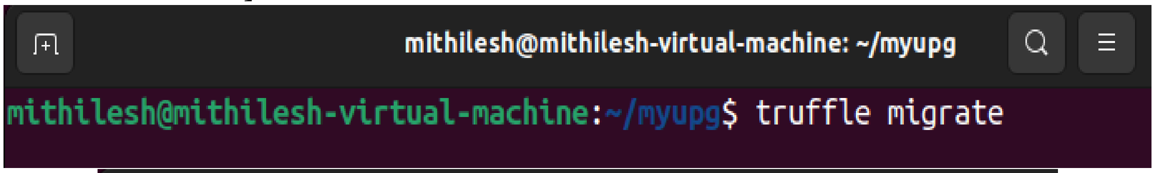
```
GNU nano 6.2                          truffle-config.js
module.exports = {
  /**
   * Networks define how you connect to your ethereum client and let you set the
   * defaults web3 uses to send transactions. If you don't specify one truffle
   * will spin up a managed Ganache instance for you on port 9545 when you
   * run `develop` or `test`. You can ask a truffle command to use a specific
   * network from the command line, e.g
   *
   * $ truffle test --network <network-name>
   */

  networks: {
        development: {
          host: "127.0.0.1",
          port: 8545,
          network_id: "*",
        }
    // Useful for testing. The `development` name is special - truffle uses it by default
    // if it's defined here and no other network is specified at the command line.
    // You should run a client (like ganache, geth, or parity) in a separate terminal
    // tab if you use this network and you must also set the `host`, `port` and `network_id`
    // options below to some value.
    //
    // development: {
    //  host: "127.0.0.1",     // Localhost (default: none)
    //  port: 8545,            // Standard Ethereum port (default: none)
    //  network_id: "*",       // Any network (default: none)
    // },
    //
    // An additional network, but with some advanced options…
    // advanced: {
^G Help       ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo      M-A Set Mark  M-] To Bracket
^X Exit       ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line M-E Redo      M-6 Copy      ^Q Where Was
```

Step 9: start ganache-cli –Switch/Open to terminal 2 ganache-cli



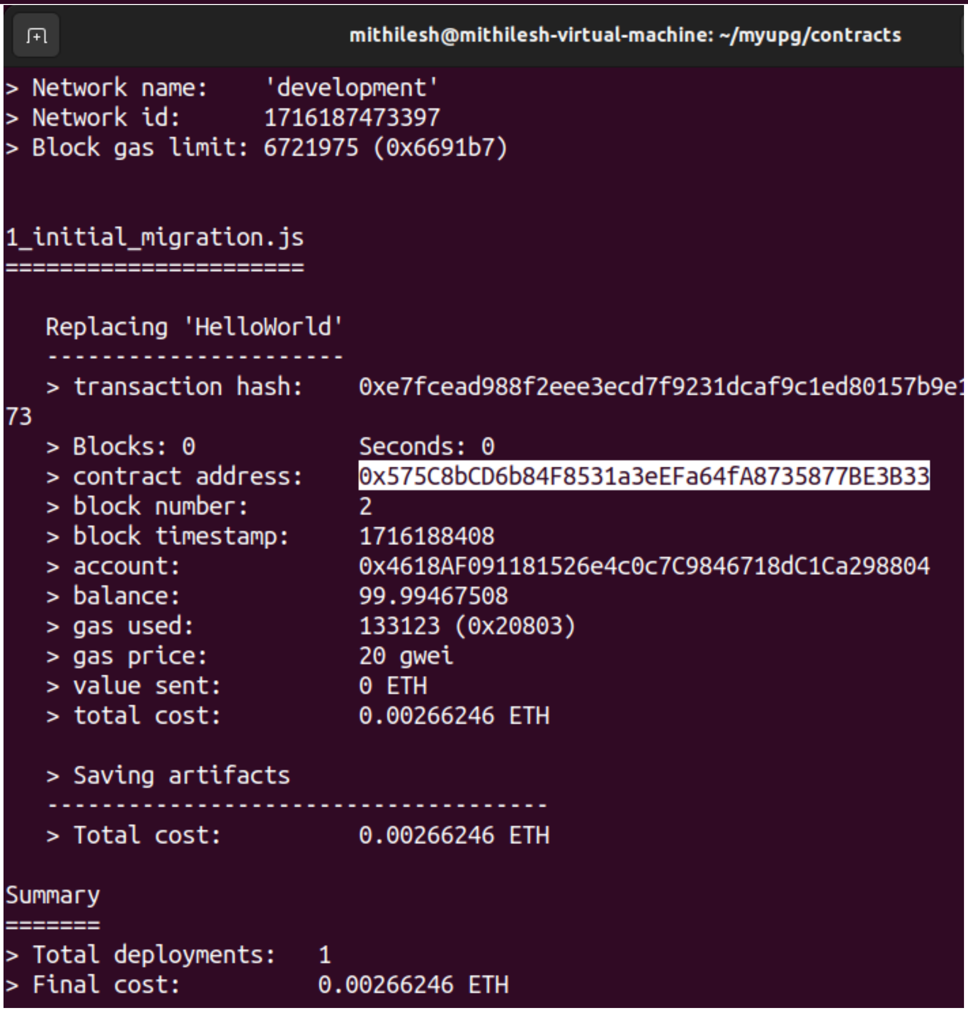
```
mithilesh@mithilesh-virtual-machine:~/myupg$ ls
contracts  migrations  test  truffle-config.js
mithilesh@mithilesh-virtual-machine:~/myupg$ nano truffle-config.js
mithilesh@mithilesh-virtual-machine:~/myupg$ ganache-cli
Ganache CLI v6.12.2 (ganache-core: 2.13.2)

Available Accounts
==================
(0) 0x4618AF091181526e4c0c7C9846718dC1Ca298804 (100 ETH)
(1) 0xC83bd49c3d248Cc424C633B1DdF3b911e7e87653 (100 ETH)
(2) 0x8F8992589F54014903320861e0a933e1Bc2Ba54d (100 ETH)
(3) 0x47118c319f90d1cc13749cCA40410CA86181627f (100 ETH)
(4) 0xc9331d32DC548a97B607478c1E8545dc36Bdd91E (100 ETH)
(5) 0x778eb0cC61Ae9260740Fe505F5586Cc57CF53d39 (100 ETH)
(6) 0x9b6E136EdD5575CE3dF776E8eb895b89DF8F88b3 (100 ETH)
(7) 0xE69DC0097a6DE3BD8FB83FD8F972FCC0406a06CC (100 ETH)
(8) 0x5B89C699119948566AA9fb54240eEe4b6aE69559 (100 ETH)
(9) 0x8cE3E8ae3f8F0447E734D66721A6eA1b8A92Ab91 (100 ETH)

Private Keys
==================
(0) 0x49e12af47f8f59e028ba26b49ddb87a46cfe2f46e393271de9a9520fbc1da724
(1) 0xd23dc7e12dce622e4b65858f04360882fee131ba3012f52b053cf8e7b5a15922
(2) 0xc9f5a5b1e2e45aa57bac69c13fb20d0928d3e617b75bdcfc8223a750ae443edf
(3) 0x6b52c6eb6df29c514a8c02c69c8c773d4ac038b0487572e33a15bceea4d4ac6c
(4) 0x10ff6182fbfe2852346d2fef404c30e2a2a840b0616fcf627877d5ccb888347f
(5) 0x1c39f91ed779fc11a4d7059b1dc65cdaeb25a2b2bdca59ed6dd287800fac1c8c
(6) 0xd76c7491a95fe7f7d58590c64b8cb917c69acf34d7449eb2a4eab44e613e59e4
(7) 0xcdfa74ac10d9e3bdadcd7a75095bcd44a179ae57bea6faa67b39cd53a7b524d2
(8) 0x0796ae0fd6b48073a04be2ca5cad071555bb91a7093b2b05e96be4d83cd95d8a
(9) 0xd98daeb9734ecebca0991e98080442dbf8bc1e99792c1fe91bf98c95453a661d

HD Wallet
==================
Mnemonic:      chair park window civil narrow reform valid poem evolve jaguar faith chalk
Base HD Path:  m/44'/60'/0'/0/{account_index}

Gas Price
==================
20000000000

Gas Limit
==================
6721975

Call Gas Limit
==================
9007199254740991

Listening on 127.0.0.1:8545
```

Step 10: deploy the truffle deploy- On terminal 1 truffle deploy
[Note contract address]



mithilesh@mithilesh-virtual-machine: ~/myupg

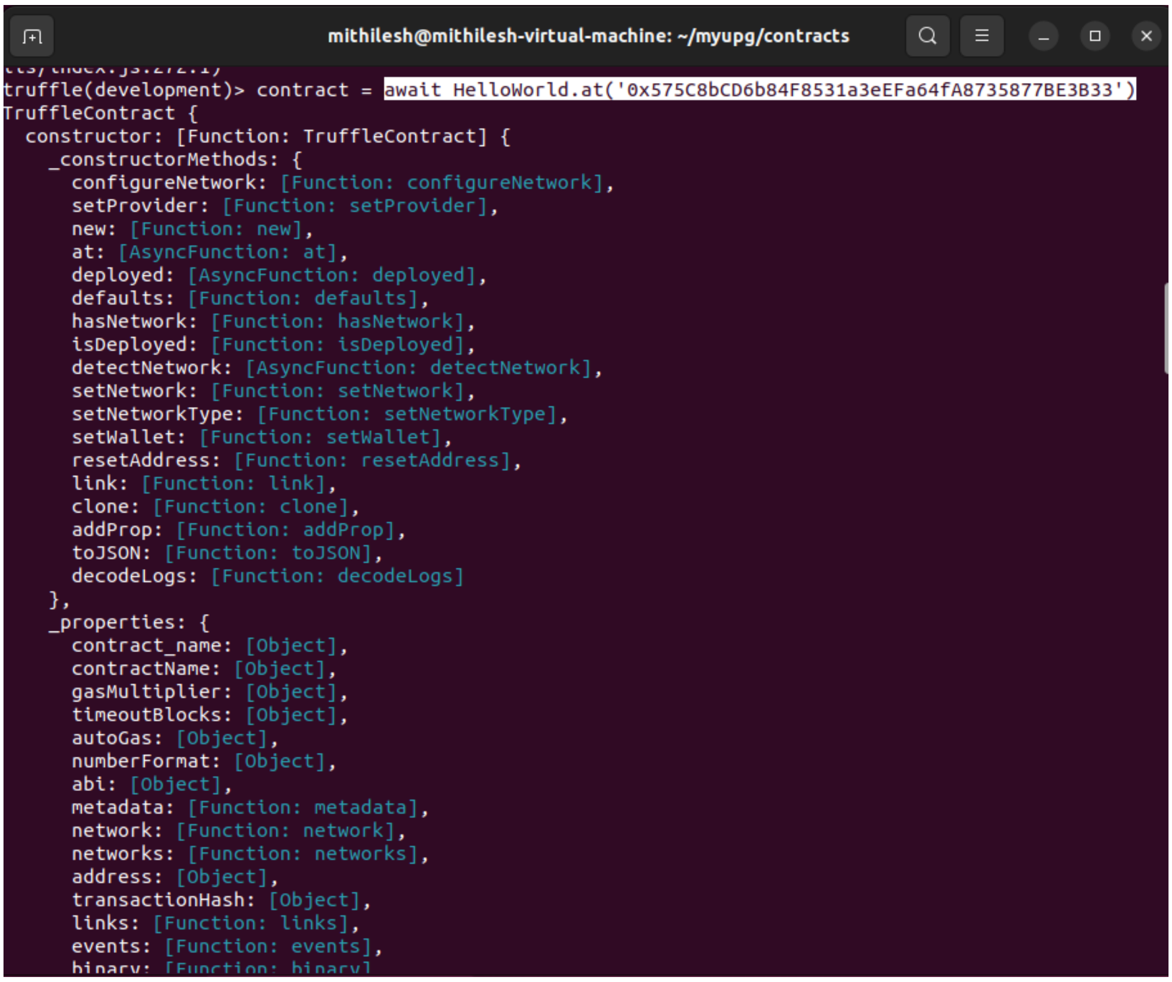
```
mithilesh@mithilesh-virtual-machine:~/myupg$ truffle migrate
```



mithilesh@mithilesh-virtual-machine: ~/myupg/contracts

```
> Network name:    'development'
> Network id:      1716187473397
> Block gas limit: 6721975 (0x6691b7)


1_initial_migration.js
======================

   Replacing 'HelloWorld'
   ----------------------
   > transaction hash:    0xe7fcead988f2eee3ecd7f9231dcaf9c1ed80157b9e1
73
   > Blocks: 0            Seconds: 0
   > contract address:    0x575C8bCD6b84F8531a3eEFa64fA8735877BE3B33
   > block number:        2
   > block timestamp:     1716188408
   > account:             0x4618AF091181526e4c0c7C9846718dC1Ca298804
   > balance:             99.99467508
   > gas used:            133123 (0x20803)
   > gas price:           20 gwei
   > value sent:          0 ETH
   > total cost:          0.00266246 ETH

   > Saving artifacts
   -------------------------------------
   > Total cost:          0.00266246 ETH

Summary
=======
> Total deployments:   1
> Final cost:          0.00266246 ETH
```

Step 11: Open truffle console - On terminal 1 truffle console
Step 11: Get reference of contact
contract = await HelloWorld.at('0x2C403EE1b30F56C0c773089c1Eb9DddF1499C969')


mithilesh@mithilesh-virtual-machine: ~/myupg/contracts

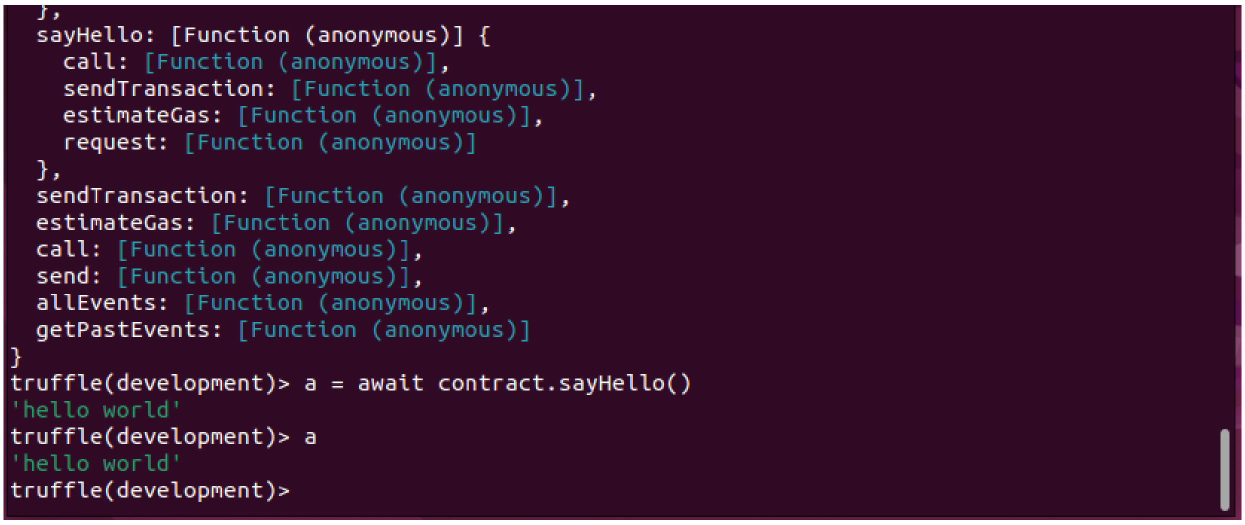
```
truffle(development)> contract = await HelloWorld.at('0x575C8bCD6b84F8531a3eEFa64fA8735877BE3B33')
TruffleContract {
  constructor: [Function: TruffleContract] {
    _constructorMethods: {
      configureNetwork: [Function: configureNetwork],
      setProvider: [Function: setProvider],
      new: [Function: new],
      at: [AsyncFunction: at],
      deployed: [AsyncFunction: deployed],
      defaults: [Function: defaults],
      hasNetwork: [Function: hasNetwork],
      isDeployed: [Function: isDeployed],
      detectNetwork: [AsyncFunction: detectNetwork],
      setNetwork: [Function: setNetwork],
      setNetworkType: [Function: setNetworkType],
      setWallet: [Function: setWallet],
      resetAddress: [Function: resetAddress],
      link: [Function: link],
      clone: [Function: clone],
      addProp: [Function: addProp],
      toJSON: [Function: toJSON],
      decodeLogs: [Function: decodeLogs]
    },
    _properties: {
      contract_name: [Object],
      contractName: [Object],
      gasMultiplier: [Object],
      timeoutBlocks: [Object],
      autoGas: [Object],
      numberFormat: [Object],
      abi: [Object],
      metadata: [Function: metadata],
      network: [Function: network],
      networks: [Function: networks],
      address: [Object],
      transactionHash: [Object],
      links: [Function: links],
      events: [Function: events],
```

[Replace '0x2C403EE1b30F56C0c773089c1Eb9DddF1499C969' with your contact address; every time you compile/deploy a new contract address will be generated]

Step 12: Call the function from the contract.

a = await contract.sayHello()

a



```
  sayHello: [Function (anonymous)] {
    call: [Function (anonymous)],
    sendTransaction: [Function (anonymous)],
    estimateGas: [Function (anonymous)],
    request: [Function (anonymous)]
  },
  sendTransaction: [Function (anonymous)],
  estimateGas: [Function (anonymous)],
  call: [Function (anonymous)],
  send: [Function (anonymous)],
  allEvents: [Function (anonymous)],
  getPastEvents: [Function (anonymous)]
}
truffle(development)> a = await contract.sayHello()
'hello world'
truffle(development)> a
'hello world'
truffle(development)>
```