Practical 1

Write the following programs for Blockchain in Python

Practical 1 a)

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

Code:

```
#pip install pycryptodome

#1A.- A simple client class that generates the private and public keys by using the built-in
Python RSA algorithm and test it.

import Crypto
import binascii

from Crypto.PublicKey import RSA
from Crypto.Signature import PKCS1_v1_5

class Client:
    def __init__(self):
        # Creating random number for key
        random = Crypto.Random.new().read
        # Creating new public key and private key
        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')
Demo = Client()
print(Demo.identity)
```

Output:

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

30819f300d06092a864886f70d010101050003818d0030818902818100e79bfa63870b87e693fb8bf67e6a4eb8e3a1618ba950f7fb9541
```

+ Code + Text

Practical 1 b)

Aim: A transaction class to send and receive money and test it.

Code:

```
#1B.- A transaction class to send and receive money and test it.
import Crypto
import binascii

import datetime
import collections

from Crypto.PublicKey import RSA
from Crypto.Signature import PKCS1_v1_5
from Crypto.Hash import SHA

class Client:
    def __init__(self):
        # Creating random number for key
        random = Crypto.Random.new().read
        # Creating new public key and private key
        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, receiver, value):
        self.sender = sender
        self.receiver = receiver
        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,
            'receiver': self.receiver,
            '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')

Ninad = Client()
print("-"*50)
print("Ninad Key")
print(Ninad.identity)

KS = Client()
print("-"*50)
print("KS Key")
print(KS.identity)

t = Transaction(Ninad, KS.identity, 10.0)
print("-"*50)
print("Transaction Sign")
signature = t.sign_transaction()
print(signature)
print("-"*50)
```

Output:

```
--------------------------------------------------
Ninad Key
30819f300d06092a864886f70d010101050003818d0030818902818100a99ad7dbf3cbfbb0d340ae1d091387896b0ef9449e032569819ac
--------------------------------------------------
KS Key
30819f300d06092a864886f70d010101050003818d0030818902818100b03818bf884264881b44027dd3e654fdd258339df2d3040ad2122
--------------------------------------------------
Transaction Sign
a7a209a226c7df822f436dae08b64dab0dcc0c744ef7ff198868ee005b3e570969e4cdb817ca36c1d69b999e308c1ae6762755df2101c1e
--------------------------------------------------
```

Practical 1 c)

Aim: Create multiple transactions and display them.

Code:

```
#!pip install pycryptodome

import Crypto
import binascii

from Crypto.PublicKey import RSA
from Crypto.Hash import SHA
from Crypto.Signature import PKCS1_v1_5

import datetime
import collections

import hashlib
from hashlib import sha256

class Client:
    def __init__(self):
        # Creating random number for key
        random = Crypto.Random.new().read
        # Creating new public key and private key
        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, receiver, value):
        self.sender = sender
        self.receiver = receiver
        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,
                "receiver": self.receiver,
                "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 sha256(message):
        return hashlib.sha256(message.encode("ascii")).hexdigest

    def mine(message, difficulty=1):
        assert difficulty >= 1
        prefix = "1" * difficulty
        for i in range(1000):
            digest = sha256(str(hash(message)) + str(i))

        if digest.startwith(prefix):
            print("after" + str(i) + "iteration found nonce:" + digest)
            return digest

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

    last_block_hash = ""

    def display_transaction(transaction):
        dict = transaction.to_dict()
        print("Sender: " + dict["sender"])
        print("-----")
        print("Receiver: " + dict["receiver"])
        print("-----")
        print("Value: " + str(dict["value"]))
        print("-----")
        print("Time: " + str(dict["time"]))
        print("-----")
```

```
    TPCoins = []

    def dump_blockchain(self):
        print("Number of blocks in chain" + str(len(self)))
        for x in range(len(Block.TPCoins)):
            block_temp = Block.TPCoins[x]
            print("block #" + str(x))
            for transaction in block_temp.verified_transactions:
                Block.display_transaction(transaction)
                print("-------")

    last_transaction_index = 0

    transactions = []

    Ninad = Client()
    ks = Client()
    vighnesh = Client()
    sairaj = Client()

    t1 = Transaction(Ninad, ks.identity, 15.0)
    t1.sign_transaction()
    transactions.append(t1)

    t2 = Transaction(Ninad, vighnesh.identity, 6.0)
    t2.sign_transaction()
    transactions.append(t2)

    t3 = Transaction(Ninad, sairaj.identity, 16.0)
    t3.sign_transaction()
    transactions.append(t3)

    t4 = Transaction(vighnesh, Ninad.identity, 8.0)
    t4.sign_transaction()
    transactions.append(t4)

    t5 = Transaction(vighnesh, ks.identity, 19.0)
    t5.sign_transaction()
    transactions.append(t5)

    t6 = Transaction(vighnesh, sairaj.identity, 35.0)
    t6.sign_transaction()
    transactions.append(t6)

    t7 = Transaction(sairaj, vighnesh.identity, 5.0)
    t7.sign_transaction()
    transactions.append(t7)

    t8 = Transaction(sairaj, Ninad.identity, 12.0)
    t8.sign_transaction()
```

```
transactions.append(t8)

t9 = Transaction(sairaj, ks.identity, 25.0)
t9.sign_transaction()
transactions.append(t9)

t10 = Transaction(Ninad, ks.identity, 1.0)
t10.sign_transaction()
transactions.append(t10)

for transaction in transactions:
    display_transaction(transaction)
    print("*" * 50)
```

Output:

```
print("*"*50)
```

```
Sender: 30819f300d06092a864886f70d010101050003818d00308189028181
-----
Receiver: 30819f300d06092a864886f70d010101050003818d003081890281
-----
Value: 15.0
-----
Time: 2024-04-24 11:58:22.603355
-----
**************************************************
Sender: 30819f300d06092a864886f70d010101050003818d00308189028181
-----
Receiver: 30819f300d06092a864886f70d010101050003818d003081890281
-----
Value: 6.0
-----
Time: 2024-04-24 11:58:22.606018
-----
**************************************************
Sender: 30819f300d06092a864886f70d010101050003818d00308189028181
-----
Receiver: 30819f300d06092a864886f70d010101050003818d003081890281
-----
Value: 16.0
-----
Time: 2024-04-24 11:58:22.608726
-----
**************************************************
```

```
-----
**************************************************
Sender: 30819f300d06092a864886f70d010101050003818d0030818902818100a72f9c1fb19a4a4382aeaedc6bea
-----
Receiver: 30819f300d06092a864886f70d010101050003818d0030818902818100bc5003b8a6f5a9a43f7739cd2a
-----
Value: 8.0
-----
Time: 2024-04-24 11:58:22.611128
-----
**************************************************
Sender: 30819f300d06092a864886f70d010101050003818d0030818902818100a72f9c1fb19a4a4382aeaedc6bea
-----
Receiver: 30819f300d06092a864886f70d010101050003818d0030818902818100c61a7aacd1dbedddd4e7a704f
-----
Value: 19.0
-----
Time: 2024-04-24 11:58:22.614112
-----
**************************************************
Sender: 30819f300d06092a864886f70d010101050003818d0030818902818100a72f9c1fb19a4a4382aeaedc6bea
-----
Receiver: 30819f300d06092a864886f70d010101050003818d0030818902818100bf3f7cc5c45ced69ddd45259a
-----
Value: 35.0
-----
Time: 2024-04-24 11:58:22.616541
-----
**************************************************
```

```
-----
**************************************************
Sender: 30819f300d06092a864886f70d010101050003818d0030818902818100bf3f7cc5c45ced69ddd45259a1964757464
-----
Receiver: 30819f300d06092a864886f70d010101050003818d0030818902818100a72f9c1fb19a4a4382aeaedc6bea34e85
-----
Value: 5.0
-----
Time: 2024-04-24 11:58:22.618543
-----
**************************************************
Sender: 30819f300d06092a864886f70d010101050003818d0030818902818100bf3f7cc5c45ced69ddd45259a1964757464
-----
Receiver: 30819f300d06092a864886f70d010101050003818d0030818902818100bc5003b8a6f5a9a43f7739cd2a15f9541
-----
Value: 12.0
-----
Time: 2024-04-24 11:58:22.619900
-----
**************************************************
Sender: 30819f300d06092a864886f70d010101050003818d0030818902818100bf3f7cc5c45ced69ddd45259a1964757464
-----
Receiver: 30819f300d06092a864886f70d010101050003818d0030818902818100c61a7aacd1dbedddd4e7a704ffa0365d1
-----
Value: 25.0
-----
Time: 2024-04-24 11:58:22.622286
-----
**************************************************
Sender: 30819f300d06092a864886f70d010101050003818d0030818902818100bc5003b8a6f5a9a43f7739cd2a15f9541bc
-----
Receiver: 30819f300d06092a864886f70d010101050003818d0030818902818100c61a7aacd1dbedddd4e7a704ffa0365d1
-----
Value: 1.0
-----
Time: 2024-04-24 11:58:22.624531
-----
**************************************************
```

Practical 1 d)

Aim: Create a blockchain, a genesis block and execute it.

Code:

```
# Aim 1D - Create a blockchain, a genesis block and execute it.
#!pip install pycryptodome
import Crypto
import binascii
import datetime
import collections
from Crypto.PublicKey import RSA
from Crypto.Hash import SHA
from Crypto.Signature import PKCS1_v1_5
class Client:
    def __init__(self):
        # Creating random number for key
        random = Crypto.Random.new().read
        # Creating new public key and private key
        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, receiver, value):
        self.sender = sender
        self.receiver = receiver
        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,
                "receiver": self.receiver,
                "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")
class Block:
    def __init__(self):
        self.verified_transactions = []
        self.previous_block_hash = ""
        self.Nonce = ""
    last_block_hash = ""
    def display_transaction(transaction):
        dict = transaction.to_dict()
        print("Sender: " + dict["sender"])
        print("-----")
        print("Receiver: " + dict["receiver"])  # Corrected typo
        print("-----")
        print("Value: " + str(dict["value"]))
        print("-----")
        print("Time: " + str(dict["time"]))
        print("-----")
Ninad = Client()
t0 = Transaction("Genesis", Ninad.identity, 500.0)
block0 = Block()
block0.previous_block_hash = None
Nonce = None
block0.verified_transactions.append(t0)
digest = hash(block0)
last_block_hash = digest
TPCoins = []
def dump_blockchain(self):
    print("Number of blocks in 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:
        Block.display_transaction(transaction)
        print("-" * 20)
    print("=" * 30)


TPCoins.append(block0)
dump_blockchain(TPCoins)
```

Output:

```
Number of blocks in chain: 1
block #0
Sender: Genesis
-----
Receiver: 30819f300d06092a864886f70d010101050003818d003081890281810 0c15f06dc4692a07cbe45da3a867658a08c996d416ab79414f4
-----
Value: 500.0
-----
Time: 2024-04-24 12:54:53.908551
-----
--------------------
==============================
```

Practical 1 e)

Aim: e. Create a mining function and test it and Add blocks to the miner and dump the blockchain.

Code:

```
# -*- coding: utf-8 -*-
import collections
import datetime
import binascii
# !pip install pycryptodome
import Crypto
from Crypto.PublicKey import RSA
from Crypto import Random
from Crypto.Hash import SHA
from Crypto.Signature import PKCS1_v1_5

class Client:
    def __init__(self):
        random=Crypto.Random.new().read
        self._private_key=RSA.generate(1024,random)
        self._public_key=self._private_key.publickey()
        self._signer=PKCS1_v1_5.new(self._private_key)
    @property
    def identity(self):
        return binascii.hexlify(self._public_key.exportKey(format='DER')).decode('ascii')

class Transaction:
    def __init__(self,sender,recipient,value):
        self.sender=sender
        self.recipient=recipient
        self.value=value
        self.time=datetime.datetime.now()
    def to_dict(self):
        if self.sender=="Genesis":
            identity="Genesis"
        else:
            identity=self.sender.identity
        return collections.OrderedDict({
            'sender':identity,
            'recipient':self.recipient,
            'value':self.value,
            'time':self.time})
    def sign_transaction(self):
        private_key=self.sender._private_key
        signer=PKCS1_v1_5.new(private_key)
        h=SHA.new(str(self.to_dict()).encode('utf8'))
        return binascii.hexlify(signer.sign(h)).decode('ascii')
```

```
import hashlib
def sha256(message):
    return hashlib.sha256(message.encode('ascii')).hexdigest()
def mine(message,difficulty=1):
    assert difficulty>=1
    prefix='1'*difficulty
    for i in range(1000):
        digest=sha256(str(hash(message))+str(i))
        if digest.startswith(prefix):
            print("after"+str(i)+"iterationsfoundnonce:"+digest)
            return digest
class Block:
    def __init__(self):
        self.verified_transactions=[]
        self.previous_block_hash=""
        self.Nonce=""

def display_transaction(transaction):

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

last_block_hash=""
TPCoins=[]
last_transaction_index=0
transactions=[]
Raja=Client()
Rani=Client()
Seema=Client()
Reema=Client()

t1=Transaction(Raja,Rani.identity,15.0)
t1.sign_transaction()
```

```
transactions.append(t1)
t2=Transaction(Raja,Seema.identity,6.0)
t2.sign_transaction()
transactions.append(t2)
t3=Transaction(Rani,Reema.identity,2.0)
t3.sign_transaction()
transactions.append(t3)
t4=Transaction(Seema,Rani.identity,4.0)
t4.sign_transaction()
transactions.append(t4)
t5=Transaction(Reema,Seema.identity,7.0)
t5.sign_transaction()
transactions.append(t5)
t6=Transaction(Rani,Seema.identity,3.0)
t6.sign_transaction()
transactions.append(t6)
t7=Transaction(Seema,Raja.identity,8.0)
t7.sign_transaction()
transactions.append(t7)
t8=Transaction(Seema,Rani.identity,1.0)
t8.sign_transaction()
transactions.append(t8)
t9=Transaction(Reema,Raja.identity,5.0)
t9.sign_transaction()
transactions.append(t9)
t10=Transaction(Reema,Rani.identity,3.0)
t10.sign_transaction()
transactions.append(t10)

#Miner1addsablock
block=Block()
for i in range(3):
    temp_transaction=transactions[last_transaction_index]
    #validatetransaction
    #if valid
    block.verified_transactions.append(temp_transaction)
    last_transaction_index+=1
block.previous_block_hash=last_block_hash
block.Nonce=mine(block,2)
digest=hash(block)
TPCoins.append(block)
last_block_hash=digest
#Miner2 adds a block
block=Block()
for i in range(3):
    temp_transaction=transactions[last_transaction_index]
    #validate transaction
    #if valid
    block.verified_transactions.append(temp_transaction)
    last_transaction_index+=1
```

```
block.previous_block_hash=last_block_hash
block.Nonce=mine(block,2)
digest=hash(block)
TPCoins.append(block)
last_block_hash=digest
#Miner3 adds a block
block=Block()
for i in range(3):
    temp_transaction=transactions[last_transaction_index]
    #validate transaction
    #if valid
    block.verified_transactions.append(temp_transaction)
    last_transaction_index+=1

block.previous_block_hash=last_block_hash
block.Nonce=mine(block,2)
digest=hash(block)
TPCoins.append(block)
last_block_hash=digest

dump_blockchain(TPCoins)
```

Output:

```
dump_blockchain(TPCoins)

Number of blocks in the chain :3
Block # 0
sender : 30819f300d06092a864886f70d010101050003818d0030818902818100af5e7c2d295bb7f3c955377812f881b4496426daa95c318437988bed26a56a8e8518
-----
recipient : 30819f300d06092a864886f70d010101050003818d0030818902818100bb469cf8e4898e9ebd0e53759f98044e8d5a0f85fec696817afe9071cea83d1d2
-----
value : 15.0
-----
time : 2024-05-15 18:57:35.383272
-----
--------------
====================================

--------------
====================================
Block # 2
sender : 30819f300d06092a864886f70d010101050003818d0030818902818100b9d6fa2fdadfed622b63b4314b31e96ed51be0ade187fb6291f8f6
-----
recipient : 30819f300d06092a864886f70d010101050003818d0030818902818100af5e7c2d295bb7f3c955377812f881b4496426daa95c3184379
-----
value : 8.0
-----
time : 2024-05-15 18:57:35.405888
-----
--------------
====================================
sender : 30819f300d06092a864886f70d010101050003818d0030818902818100b9d6fa2fdadfed622b63b4314b31e96ed51be0ade187fb6291f8f6
-----
recipient : 30819f300d06092a864886f70d010101050003818d0030818902818100bb469cf8e4898e9ebd0e53759f98044e8d5a0f85fec696817af
-----
value : 1.0
-----
time : 2024-05-15 18:57:35.412450
-----
--------------
```

Practical 2

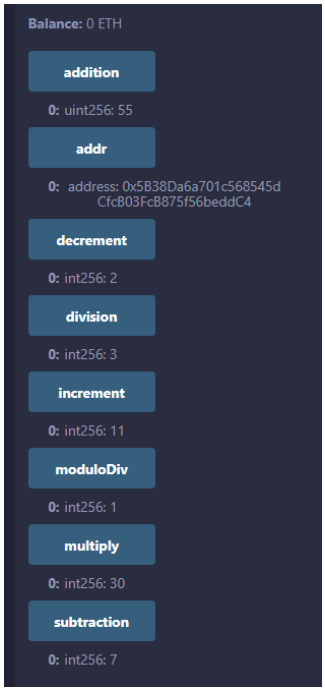
Implement and demonstrate the use of the following in Solidity:

Aim: A) Variable and Operators.

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;
contract PrimitiveDataTypes {
    uint8   a = 20;
    uint256 b = 35;
    int     c = 10;
    int8    d = 3;
    bool    flag = true;
    address public addr;
    constructor() {
        addr = msg.sender;
    }
    uint public addition    = a + b;
    int  public subtraction = c - d;
    int  public multiply    = d * c;
    int  public division    = c / d;
    int  public moduloDiv   = c % d;
    int  public increment   = ++c;
    int  public decrement
```

Output:



Aim: B)Loops.

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;
contract Loop {
    function summation(uint256 n) public pure returns (uint256) {
        uint256 sum = 0;
        for (uint256 i = 1; i <= n; i++) {
            sum += i;
        }
        return sum;
    }

    function sumWhile(uint256 n) public pure returns (uint256) {
        uint256 sum = 0;
        uint256 i = 1;
        while (i <= n) {
            sum += i;
            i++;
        }
        return sum;
    }

    function sumDoWhile(uint256 n) public pure returns (uint256) {
        uint256 sum = 0;
        uint256 i = 1;
        do {
            sum += i;
            i++;
        } while (i <= n);
        return sum;
    }
}
```

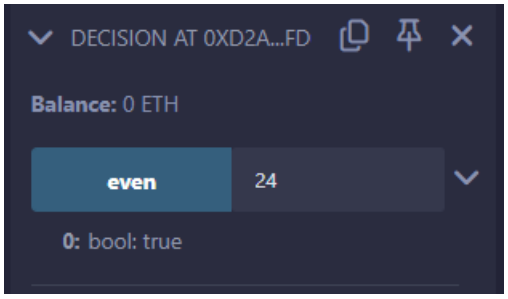
Output:

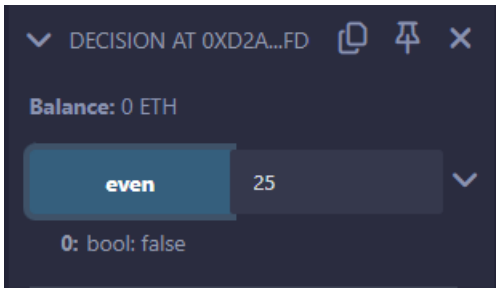
Aim: C) Decision Making.

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;

contract decision{

    function even(uint n) public pure returns(bool){
        if(n%2==0){
            return true;
        }
        else{
            return false;
        }
    }
}
```

Output:





Aim: D) Arrays.

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;

contract Arrays {

    // Declaring an array
    uint[] public array1 = [1, 2, 3, 4, 5];

    function fetch(uint index) public view returns (uint) {
        require(index < array1.length, "Index out of bounds");
        return array1[index];
    }
}
```

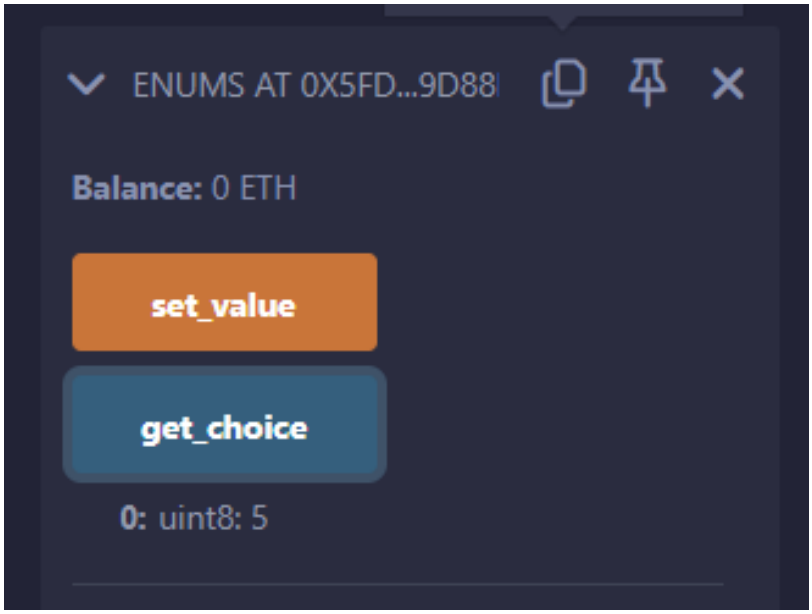
Output:

Aim: E) Enums.

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;

contract Enums {
    enum week_days {Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday}
    week_days choice;

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

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

Output:

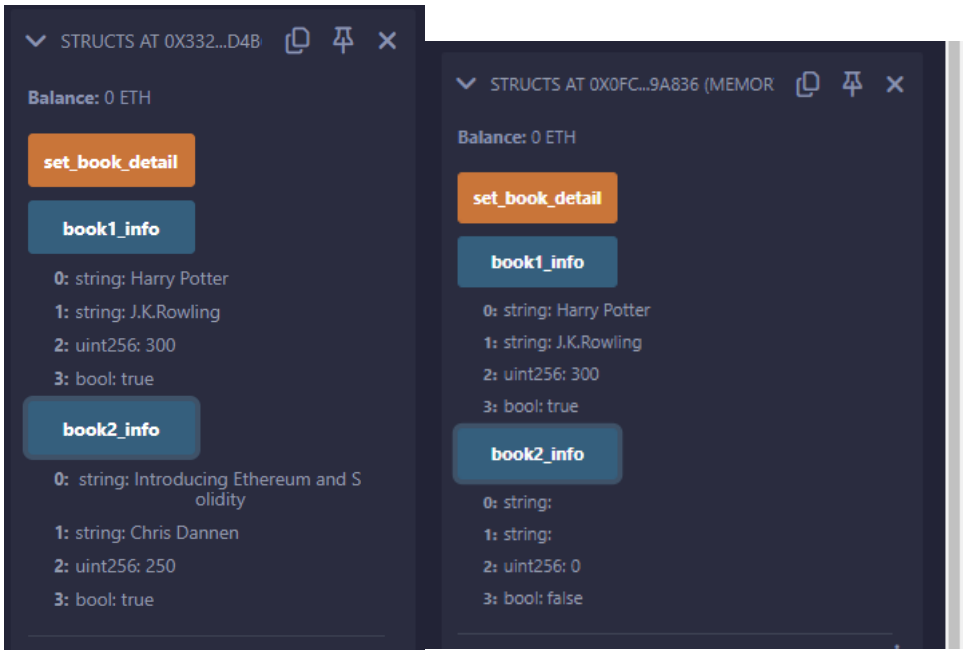


Aim: F) Structs.

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;

contract Structs {

    struct Book {
        string name;
        string writer;
        uint price;
        bool available;
    }
    Book book1;
    Book book2 = Book("Harry Potter", "J.K.Rowling", 300, true);

    function set_book_detail() public {
        book1 = Book("Introducing Ethereum and Solidity", "Chris Dannen", 250, true);
    }

    function book1_info() public view returns (string memory, string memory, uint, bool) {
        return(book2.name, book2.writer, book2.price, book2.available);
    }

    function book2_info() public view returns (string memory, string memory, uint, bool) {
        return (book1.name, book1.writer, book1.price, book1.available);
    }
}
```

Output:

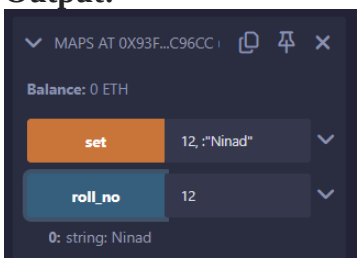


Aim: G) Mappings.

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;
contract maps {
    mapping (uint => string) public roll_no;

    function set(uint keys, string memory value) public {
        roll_no[keys] = value;
    }
}
```

Output:

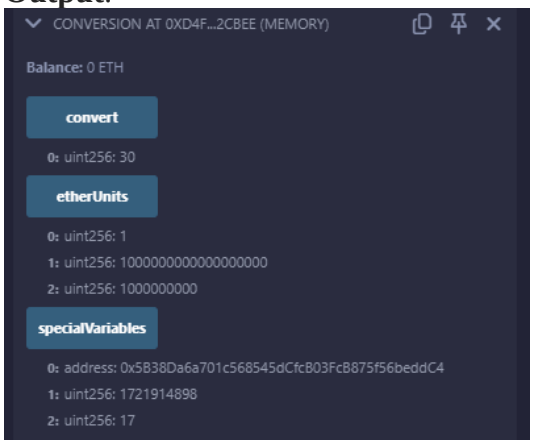


Aim: H) Conversions, Ether Units, Special Variables.

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;
contract conversion {
    uint   a = 5;    // Default unsigned integer type
    uint8  b = 10;   // 8-bit unsigned integer
    uint16 c = 15;   // 16-bit unsigned integer

    function convert() public view returns (uint) {
        uint result = a + uint(b) + uint(c); // Convert b and c to uint and add them to a
        return result; // Return the result
    }
}
```

Output:

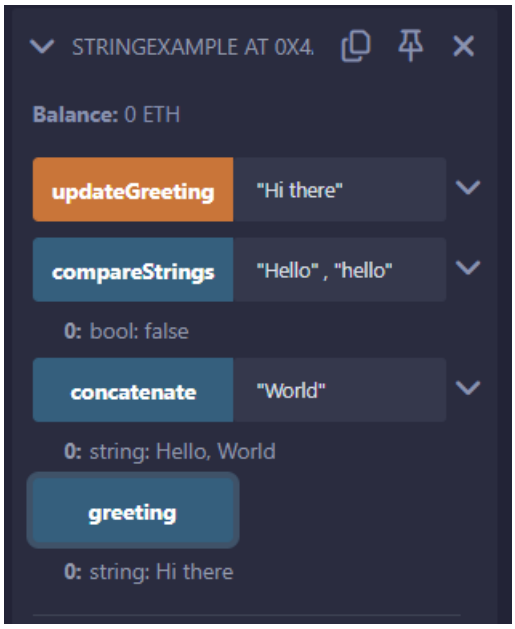


Aim: I) Strings.

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;

contract StringExample {
    string public greeting = "Hello, ";

    function concatenate(string memory _name) public view returns (string memory) {
        return string(abi.encodePacked(greeting, _name));
    }

    function compareStrings(string memory _a, string memory _b) public pure returns (bool) {
        return keccak256(abi.encodePacked(_a)) == keccak256(abi.encodePacked(_b));
    }

    function updateGreeting(string memory _newGreeting) public {
        greeting = _newGreeting;
    }
}
```

Output:



Practical 3

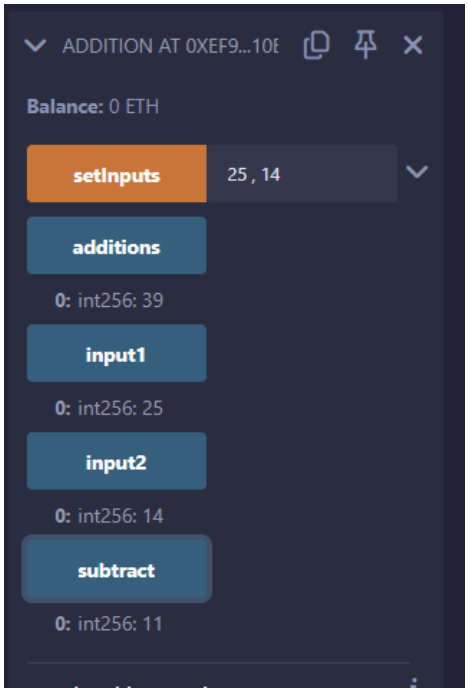
Implement and demonstrate the use of the following in Solidity:

Aim: A) Function

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;

contract Addition {

    int public input1;
    int public input2;

    function setInputs(int _input1, int _input2) public {
        input1 = _input1;
        input2 = _input2;
    }

    function additions() public view returns(int) {
        return input1 + input2;
    }

    function subtract() public view returns(int) {
        return input1 - input2;
    }
}
```

Output:



Aim: B) Fallback Function

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;

contract fallbackfn {

    // Event to log details when fallback or receive function is called
    event Log(string func, address sender, uint value, bytes data);

    // Fallback function to handle calls to the contract with data or no matching function
    fallback() external payable {
        emit Log("fallback", msg.sender, msg.value, msg.data); // Emit log with details
    }

    // Receive function to handle plain ether transfers
    receive() external payable {
        emit Log("receive", msg.sender, msg.value, ""); // Emit log with details (msg.data is
empty)
        //msg.data is empty hence no need to specify it and mark it as empty string

    }
}
```

Output:

logs

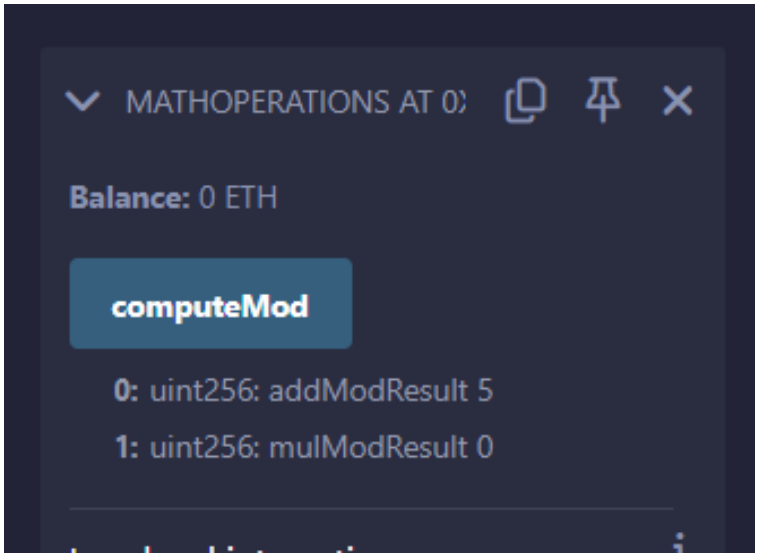

```
[
    {
        "from": "0xb31BA5cDC07A2EaFAF77c95294fd4aE27D04E9CA",
        "topic": "0xf7f75251dee7d7fc22deac3247729ebe7c86541f35930bf10c2a4207479a3b6c",
        "event": "Log",
        "args": {
            "0": "receive",
            "1": "0x5B38Da6a701c568545dCfcB03FcB875f56beddC4",
            "2": "0",
            "3": "0x",
            "func": "receive",
            "sender": "0x5B38Da6a701c568545dCfcB03FcB875f56beddC4",
            "value": "0",
            "data": "0x"
        }
    }
]
```

Aim: C) c. Mathematical functions.

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;

contract MathOperations {
    // addMod computes (x + y) % k
    // mulMod computes (x * y) % k

    // Function to compute modular addition and multiplication
    // @return addModResult: Result of (x + y) % k
    // @return mulModResult: Result of (x * y) % k
    function computeMod() public pure returns (uint addModResult, uint mulModResult) {
        uint x = 3;
        uint y = 2;
        uint k = 6;
        addModResult = addmod(x, y, k); // Compute (x + y) % k
        mulModResult = mulmod(x, y, k); // Compute (x * y) % k
    }
}
```

Output:

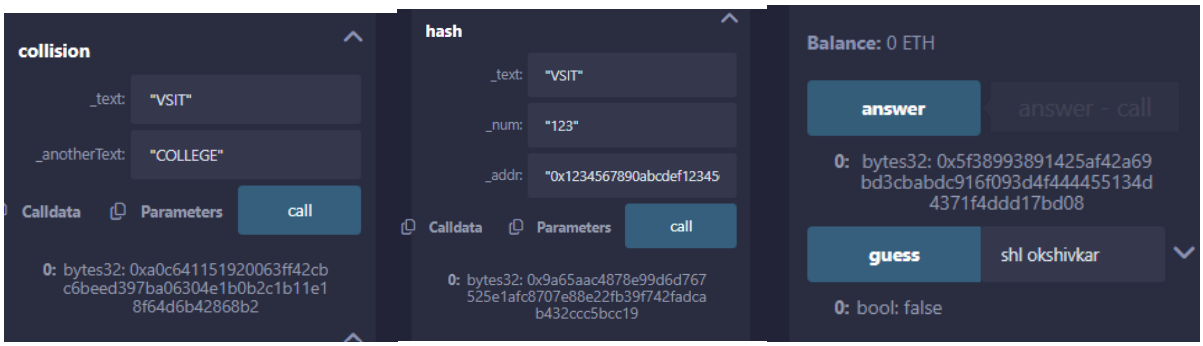


Aim: D) d. Cryptographic functions.

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;

contract Crypto {
   function hash(string memory _text,uint _num,address _addr) public pure returns (bytes32)
{
         return keccak256(abi.encodePacked(_text, _num, _addr));
         }

   function collision(string memory _text, string memory _anotherText)public pure returns
(bytes32){
            return keccak256(abi.encodePacked(_text, _anotherText));
         }
}

contract GuessTheWord {
   bytes32 public answer =
0x5f38993891425af42a69bd3cbabdc916f093d4f444455134d4371f4ddd17bd08;

   function guess(string memory _word) public view returns (bool) {
    return keccak256(abi.encodePacked(_word)) == answer;
   }
}
```

Output:

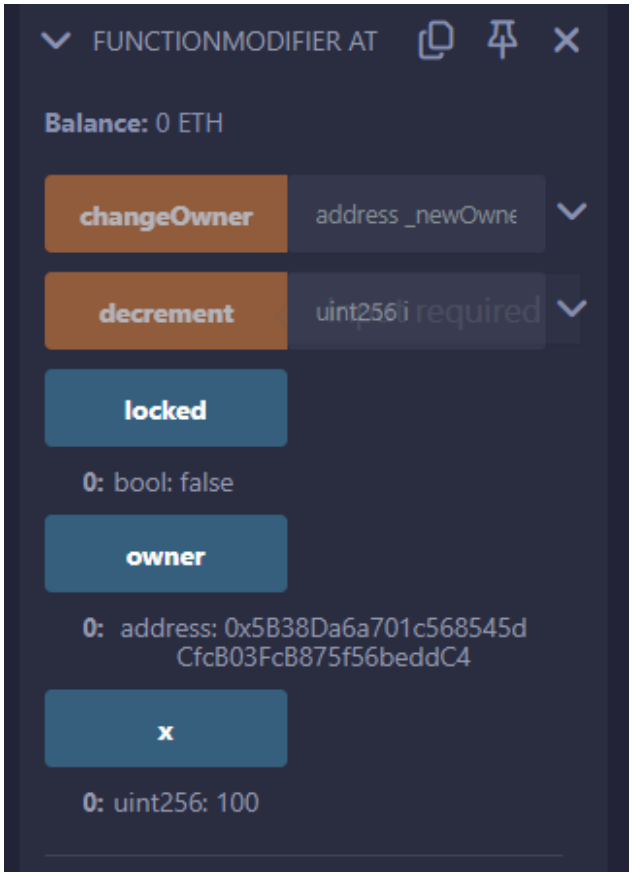


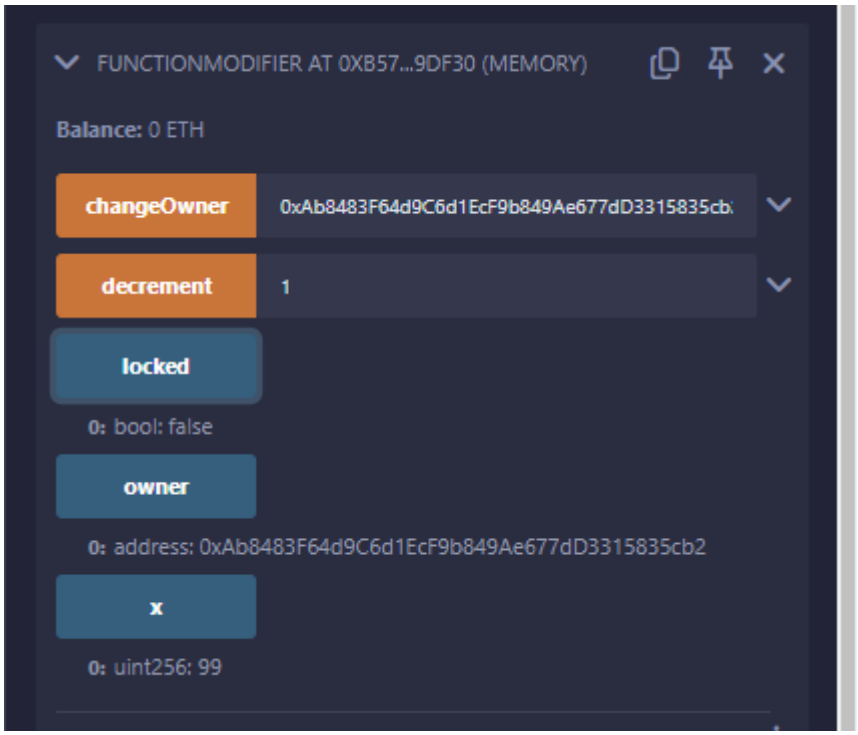
Aim: E) e. Function Modifiers.

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.13;

contract FunctionModifier {
    address public owner;
    uint256 public x = 100;
    bool public locked;

    constructor() {
        // Set the transaction sender as the owner of the contract.
        owner = msg.sender;
    }

    modifier onlyOwner() {
        require(msg.sender == owner, "Not owner");
        _;
    }

    modifier validAddress(address _addr) {
        require(_addr != address(0), "Not valid address");
        _;
    }

    function changeOwner(address _newOwner)
        public
        onlyOwner
        validAddress(_newOwner)
    {
        owner = _newOwner;
    }

    modifier noReentrancy() {
        require(!locked, "No reentrancy");
        locked = true;
        _;
        locked = false;
    }
    function decrement(uint256 i) public noReentrancy {
        x -= i;
        if (i > 1) {
            decrement(i - 1);
        }
    }
}
```

Output:


FUNCTIONMODIFIER AT
Balance: 0 ETH
changeOwner
address _newOwne
decrement
uint256 i required
locked
0: bool: false
owner
0: address: 0x5B38Da6a701c568545d
CfcB03FcB875f56beddC4
x
0: uint256: 100

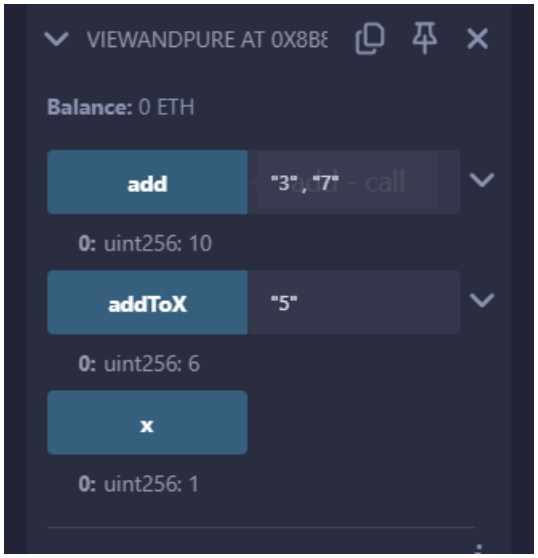

FUNCTIONMODIFIER AT 0XB57...9DF30 (MEMORY)
Balance: 0 ETH
changeOwner
0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb
decrement
1
locked
0: bool: false
owner
0: address: 0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2
x
0: uint256: 99

Aim: F) f. View and Pure Functions.

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.3;

contract ViewAndPure {
    uint public x = 1;

    // Promise not to modify the state.
    function addToX(uint y) public view returns (uint) {
        return x + y;
    }

    // Promise not to modify or read from the state.
    function add(uint i, uint j) public pure returns (uint) {
        return i + j;
    }
}
```

Output:

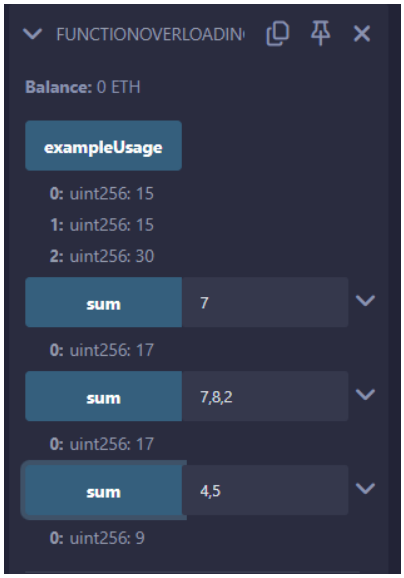


Aim: G) g. Function Overloading.

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;

contract FunctionOverloading {
    // Function with one parameter
    function sum(uint a) public pure returns (uint) {
        return a + 10;
    }
    // Overloaded function with two parameters
    function sum(uint a, uint b) public pure returns (uint) {
        return a + b;
    }
    // Overloaded function with three parameters
    function sum(uint a, uint b, uint c) public pure returns (uint) {
        return a + b + c;
    }
    // Examples of calling overloaded functions
    function exampleUsage() public pure returns (uint, uint, uint) {
        uint result1 = sum(5);          // Calls the first sum function
        uint result2 = sum(5, 10);      // Calls the second sum function
        uint result3 = sum(5, 10, 15);  // Calls the third sum function

        return (result1, result2, result3);
    }
}
```

Output:



Practical 4

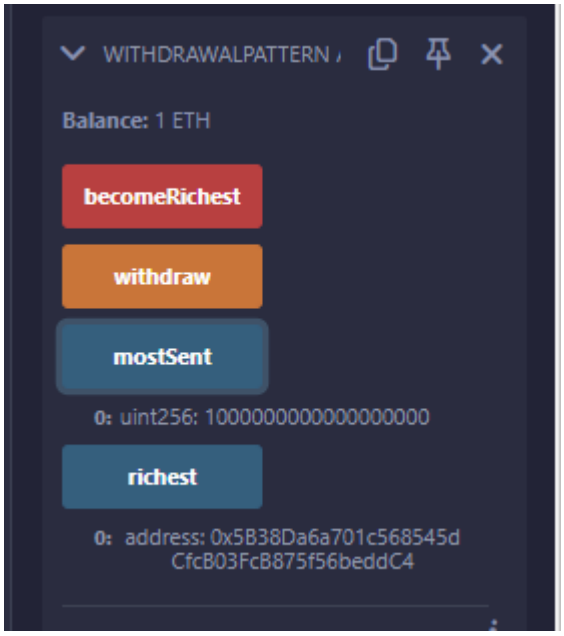
Implement and demonstrate the use of the following in Solidity:

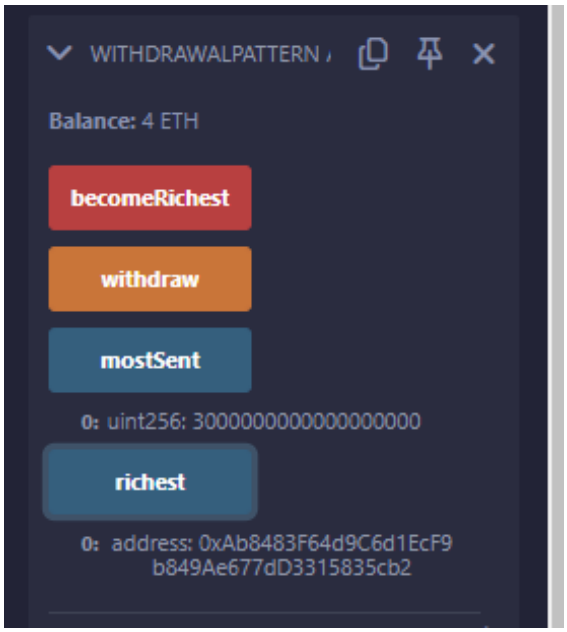
Aim: A) a. Withdrawal Pattern.

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.13;

contract withdrawalPattern{
    address public richest;
    uint public mostSent;

    mapping (address=>uint) pendingWithdrawals;
    error NotEnoughEther();

    constructor() payable{
        richest = msg.sender;
        mostSent = msg.value;
    }

    function becomeRichest() public payable{
        if (msg.value <= mostSent) revert NotEnoughEther();
        pendingWithdrawals[richest] += msg.value;
        richest = msg.sender;
        mostSent = msg.value;
    }

    function withdraw() public {
        uint amount = pendingWithdrawals[msg.sender];
        pendingWithdrawals[msg.sender] = 0;
        payable (msg.sender).transfer(amount);
    }

}
```

Output:


WITHDRAWALPATTERN
Balance: 1 ETH
becomeRichest
withdraw
mostSent
0: uint256: 1000000000000000000
richest
0: address: 0x5B38Da6a701c568545d
CfcB03FcB875f56beddC4

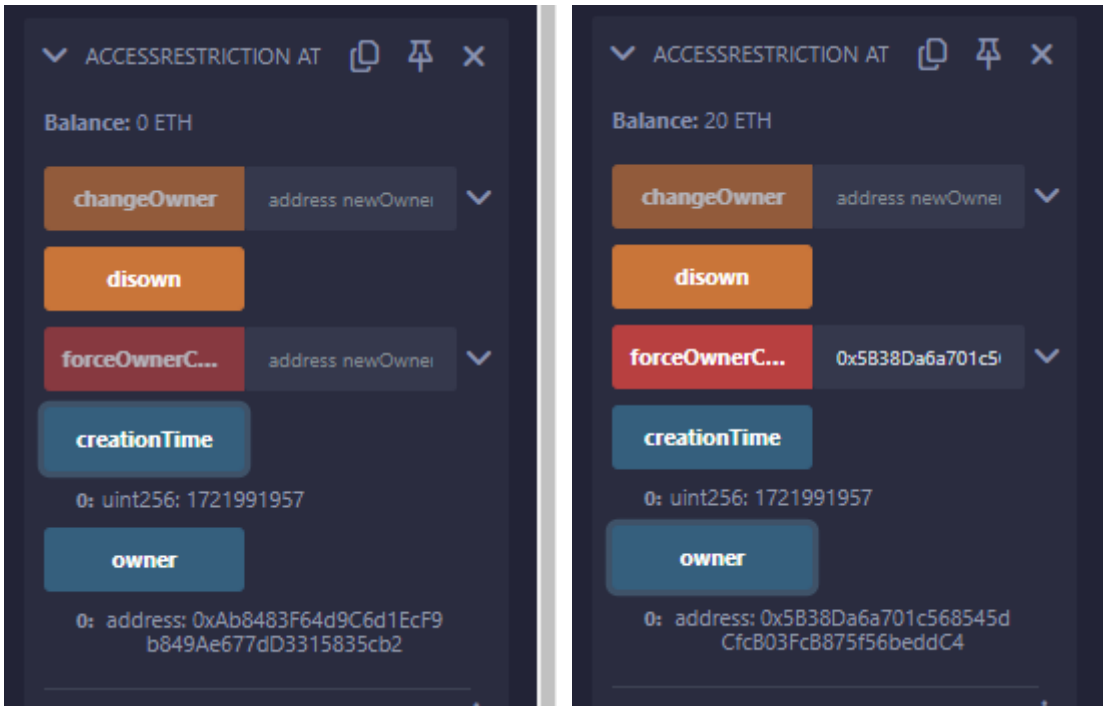

WITHDRAWALPATTERN
Balance: 4 ETH
becomeRichest
withdraw
mostSent
0: uint256: 3000000000000000000
richest
0: address: 0xAb8483F64d9C6d1EcF9
b849Ae677dD3315835cb2

Aim: B) b. Restricted Access.

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;
contract AccessRestriction {

    address public owner = msg.sender;
    uint public creationTime = block.timestamp;

    error Unauthorized();
    error TooEarly();
    error NotEnoughEther();

    modifier onlyBy(address account){
        if (msg.sender != account)
        revert Unauthorized();
        _;
    }

    modifier costs(uint amount) {
        if (msg.value < amount)
            revert NotEnoughEther();
            _;
        if (msg.value > amount)
            payable(msg.sender).transfer(msg.value - amount);
    }

    modifier onlyAfter(uint time) {
        if (block.timestamp < time)
            revert TooEarly();
            _;
    }

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

    function disown()public onlyBy(owner) onlyAfter(creationTime + 6 weeks){
        delete owner;
    }
    function forceOwnerChange(address newOwner)public payable costs(200 ether){
        owner = newOwner;
        // just some example condition
        if (uint160(owner) & 0 == 1)
            return;
    }
}
```

Output:


ACCESSRESTRICTION AT
Balance: 0 ETH
changeOwner
address newOwner
disown
forceOwnerC...
address newOwner
creationTime
0: uint256: 1721991957
owner
0: address: 0xAb8483F64d9C6d1EcF9
b849Ae677dD3315835cb2
ACCESSRESTRICTION AT
Balance: 20 ETH
changeOwner
address newOwner
disown
forceOwnerC...
0x5B38Da6a701c5
creationTime
0: uint256: 1721991957
owner
0: address: 0x5B38Da6a701c568545d
CfcB03FcB875f56beddC4

Practical 5

Implement and demonstrate the use of the following in Solidity:

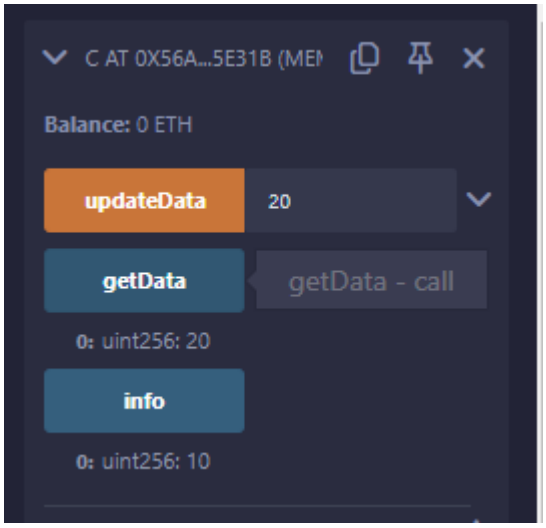
Aim: A) a. Contracts and Inheritance.

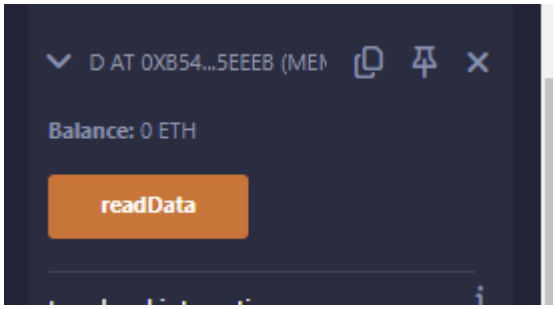
Code:

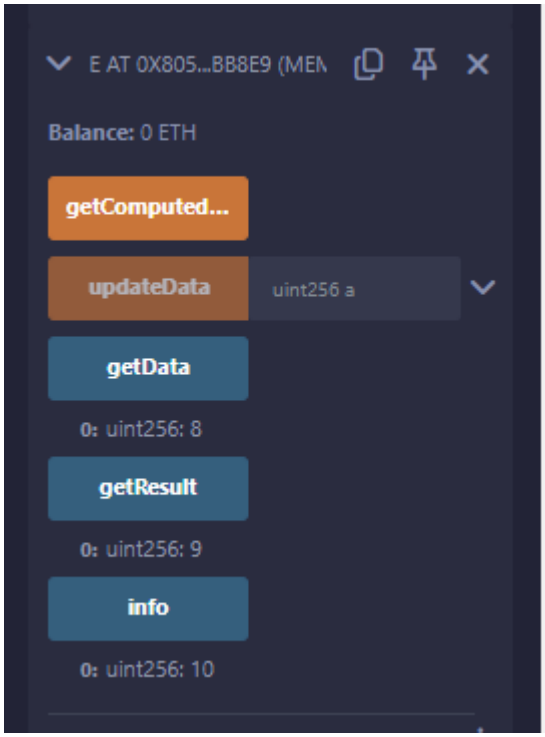
```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;

contract C{

    uint private data;
    uint public info;

    constructor()  {
        info = 10;
        }

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

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

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



contract D {

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

```
contract E is C {

    uint private result;
    C private c;

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

    function getComputedResult() public {
        result = compute(3, 6);
    }

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

Output:





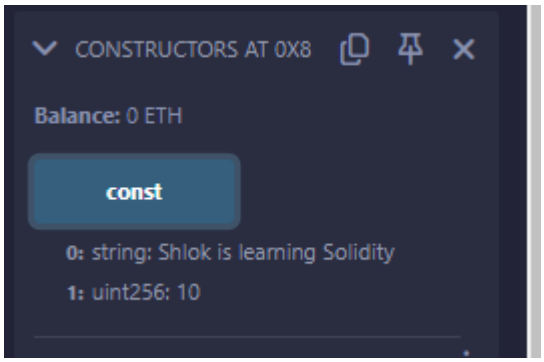

E AT 0X805...BB8E9 (MEM
Balance: 0 ETH
getComputed...
updateData
uint256 a
getData
0: uint256: 8
getResult
0: uint256: 9
info
0: uint256: 10

Aim: B) b. Constructors

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;

contract constructors{

    string str;
    uint amount;

    constructor(){
        str  = "Shlok is learning Solidity";
        amount = 10;
    }

    function const()public view returns(string memory,uint){
        return (str,amount);

    }

}
```

Output:

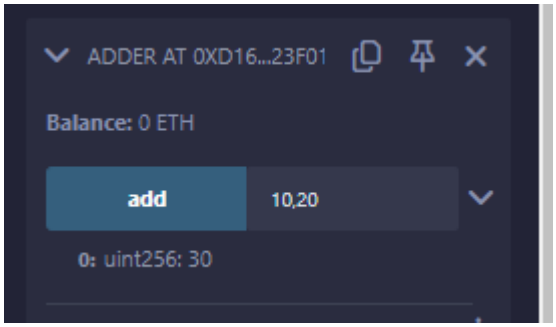


Aim: C) c. Abstract Contracts.

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;

abstract contract Main {
    // Define an abstract function that can be overridden
    function add(uint a, uint b) public virtual pure returns (uint);
}

contract Adder is Main {
    // Override the add function from the Main contract
    function add(uint a, uint b) public override pure returns (uint) {
        return a + b;
    }
}
```

Output:

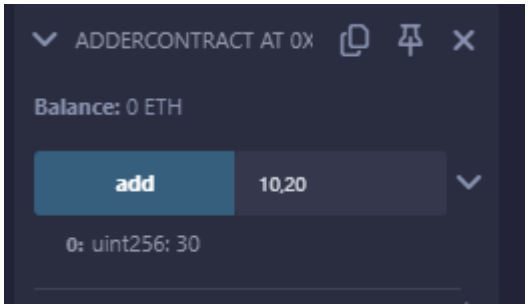


Aim: D) d. Interfaces

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;

interface adder{
    function add(uint a, uint b)external pure returns(uint);
}

contract adderContract is adder{
    function add(uint a, uint b)external pure returns(uint){
        return a+b;
    }
}
```

Output:



Practical 6

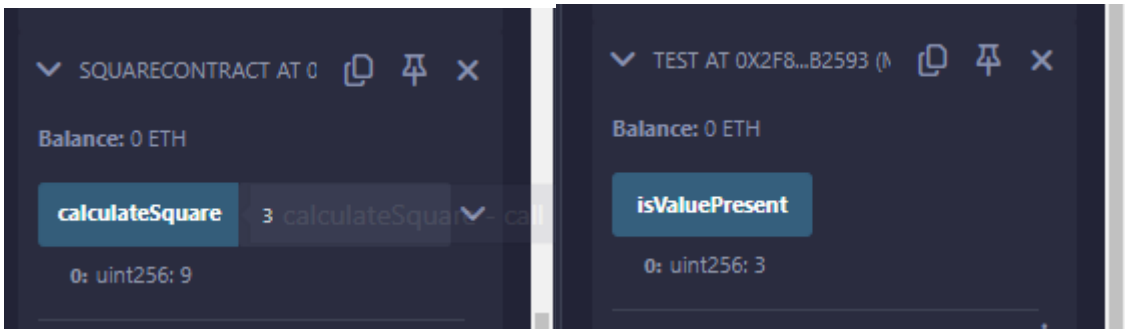
Implement and demonstrate the use of the following in Solidity:

Aim: a. Libraries.

Code:

```
pragma solidity ^0.8.17;
library Search {
  function indexOf(uint[] storage self, uint value) internal view returns (uint) {
    for (uint i = 0; i < self.length; i++) {
      if (self[i] == value) {
        return i;
      }
    }
    return type(uint).max;
  }
}
contract Test {
  uint[] data;

  constructor() {
    data.push(1);
    data.push(2);
    data.push(3);
    data.push(4);
    data.push(5);
  }
  function isValuePresent() external view returns (uint) {
    uint value = 4;
    uint index = Search.indexOf(data, value);
    return index;
  }
}
library MathLibrary {
  function square(uint num) internal pure returns (uint) {
    return num * num;
  }
}
contract SquareContract {
  using MathLibrary for uint;
  function calculateSquare(uint num) external pure returns (uint) {
    return num.square();
  }
}
```

Output:

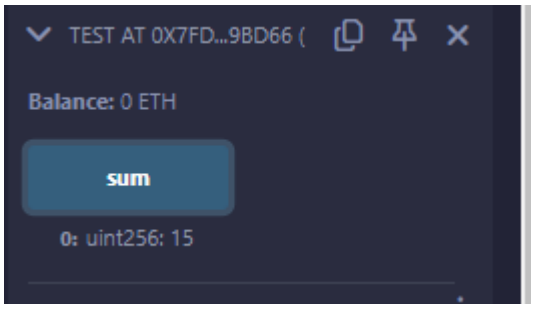

SQUARECONTRACT AT 0
Balance: 0 ETH
calculateSquare
3
0: uint256: 9
TEST AT 0X2F8...B2593
Balance: 0 ETH
isValuePresent
0: uint256: 3

Aim: b. Assembly

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;

library Sum {
  function sumUsingInlineAssembly(uint[] memory _data) public pure returns (uint sum) {
    for (uint i = 0; i < _data.length; ++i) {
      assembly {
        // Load the value from memory at the current index
        let value := mload(add(add(_data, 0x20), mul(i, 0x20)))
        sum := add(sum, value)
      }
    }
    return sum;
  }
}

contract Test {
  uint[] data;

  constructor() {
    data.push(1);
    data.push(2);
    data.push(3);
    data.push(4);
    data.push(5);
  }

  function sum() external view returns (uint) {
    return Sum.sumUsingInlineAssembly(data);
  }
}
```

Output:

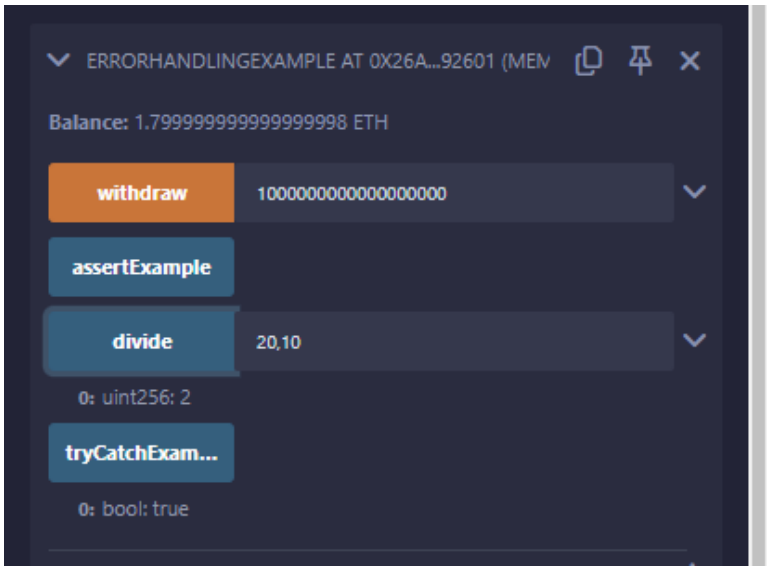


Aim: c. Error handling.

Code:

```
pragma solidity ^0.8.17;
contract ErrorHandlingExample {
    constructor() payable {
        // Allow the contract to receive Ether during deployment
    }

    function divide(uint256 numerator, uint256 denominator) external pure returns (uint256) {
        require(denominator != 0, "Division by zero is not allowed");
        return numerator / denominator;
    }

    function withdraw(uint256 amount) external {
        require(amount <= address(this).balance, "Insufficient balance");

        payable(msg.sender).transfer(amount);
    }

    function assertExample() external pure {
        uint256 x = 5;
        uint256 y = 10;
        assert(x < y);
    }
    function tryCatchExample() external view returns (bool) {
        try this.divide(10, 5) returns (uint256 result) {
            return true;
        } catch {
            return false;
        }
    }
}
```

Output:

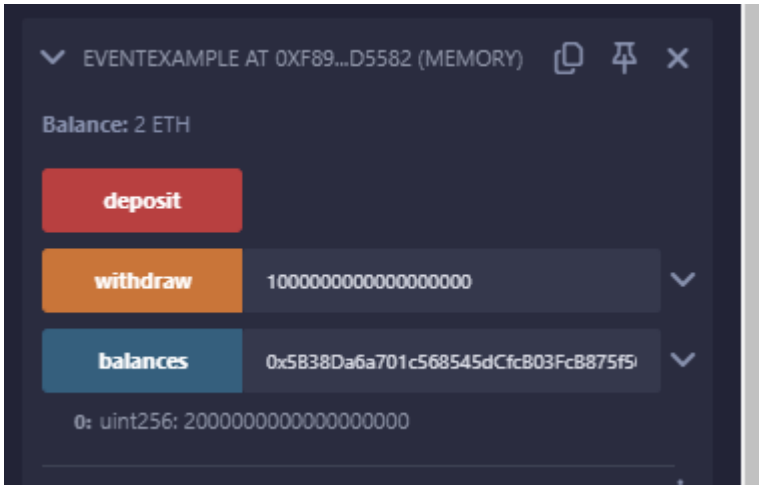

ERRORHANDLINGEXAMPLE AT 0X26A...92601 (MEM
Balance: 1.799999999999999998 ETH
withdraw
1000000000000000000
assertExample
divide
20,10
0: uint256: 2
tryCatchExam...
0: bool: true

Aim: d. Events.

Code:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;
contract EventExample {
    event Deposit(address indexed from, uint256 amount);
    event Withdraw(address indexed to, uint256 amount);

    mapping(address => uint256) public balances;

    function deposit() public payable {
        require(msg.value > 0, "Must deposit more than 0 ether");

        balances[msg.sender] += msg.value;

        emit Deposit(msg.sender, msg.value);
    }

    function withdraw(uint256 amount) public {
        require(balances[msg.sender] >= amount, "Insufficient balance");

        balances[msg.sender] -= amount;

        payable(msg.sender).transfer(amount);

        emit Withdraw(msg.sender, amount);
    }
}
```

Output:



Practical 7

Install Hyperledger fabric:

Aim: Install hyperledger fabric

Commands and Output:

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS 2

pcs@Pranav:~/BC_Pract/fabric$ git --version
git version 2.34.1
pcs@Pranav:~/BC_Pract/fabric$ curl --version
curl 7.81.0 (x86_64-pc-linux-gnu) libcurl/7.81.0 OpenSSL/3.
sl/zlib nghttp2/1.43.0 librtmp/2.3 OpenLDAP/2.5.16
Release-Date: 2022-01-05
Protocols: dict file ftp ftps gopher gophers http https ima
Features: alt-svc AsynchDNS brotli GSS-API HSTS HTTP2 HTTPS
pcs@Pranav:~/BC_Pract/fabric$
```

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS 2

pcs@Pranav:~/BC_Pract/fabric$ docker --version
Docker version 24.0.5, build 24.0.5-0ubuntu1~22.04.1
pcs@Pranav:~/BC_Pract/fabric$ jq --version
jq-1.6
pcs@Pranav:~/BC_Pract/fabric$
```

Download fabric samples

curl -sSLO https://raw.githubusercontent.com/hyperledger/fabric/main/scripts/install-fabric.sh && chmod +x install-fabric.sh

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS 2   bash

pcs@Pranav:~/BC_Pract/fabric$ curl -sSLO https://raw.githubusercontent.com/hyperledger/fabric/main/scripts/install-fabric.sh && chmod +x install-fabric.sh
pcs@Pranav:~/BC_Pract/fabric$ ls
install-fabric.sh
pcs@Pranav:~/BC_Pract/fabric$
```

Pull the docker containers

./install-fabric.sh

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

be49257713f2: Pull complete
Digest: sha256:09a67ee71cfdb2861475d37cfcc822f00545dc6852a43a6326e608b5926da1b5
Status: Downloaded newer image for hyperledger/fabric-ca:1.5.10
docker.io/hyperledger/fabric-ca:1.5.10
===> List out hyperledger images
hyperledger/fabric-peer      2.5       4b70009a7773   4 weeks ago   141MB
hyperledger/fabric-peer      2.5.7     4b70009a7773   4 weeks ago   141MB
hyperledger/fabric-peer      latest    4b70009a7773   4 weeks ago   141MB
hyperledger/fabric-orderer   2.5       3209e74fbdbb   4 weeks ago   110MB
hyperledger/fabric-orderer   2.5.7     3209e74fbdbb   4 weeks ago   110MB
hyperledger/fabric-orderer   latest    3209e74fbdbb   4 weeks ago   110MB
hyperledger/fabric-ccenv     2.5       682214ab2caa   4 weeks ago   629MB
hyperledger/fabric-ccenv     2.5.7     682214ab2caa   4 weeks ago   629MB
hyperledger/fabric-ccenv     latest    682214ab2caa   4 weeks ago   629MB
hyperledger/fabric-baseos    2.5       f8ac867caa68   4 weeks ago   128MB
hyperledger/fabric-baseos    2.5.7     f8ac867caa68   4 weeks ago   128MB
hyperledger/fabric-baseos    latest    f8ac867caa68   4 weeks ago   128MB
hyperledger/fabric-ca        1.5       da516cafd70e   4 weeks ago   206MB
hyperledger/fabric-ca        1.5.10    da516cafd70e   4 weeks ago   206MB
hyperledger/fabric-ca        latest    da516cafd70e   4 weeks ago   206MB
pcs@Pranav:~/BC_Pract/fabric$
L: Ubuntu   ⊗ 0 ⚠ 0   0
```

Navigate to test network directory

ls

cd fabric-samples

ls

```
hyperledger/fabric-ca        latest    da516cafd70e   4 weeks ago   206MB
pcs@Pranav:~/BC_Pract/fabric$ ls
fabric-samples  install-fabric.sh
pcs@Pranav:~/BC_Pract/fabric$ cd fabric-samples/
pcs@Pranav:~/BC_Pract/fabric/fabric-samples$ ls
CHANGELOG.md        README.md                    asset-transfer-private-data      builders                        off_chain_data          token-erc-20
CODEOWNERS          SECURITY.md                  asset-transfer-sbe               ci                              test-application        token-erc-721
CODE_OF_CONDUCT.md  asset-transfer-abac          asset-transfer-secured-agreement config                          test-network            token-sdk
CONTRIBUTING.md     asset-transfer-basic         auction-dutch                    full-stack-asset-transfer-guide test-network-k8s        token-utxo
LICENSE             asset-transfer-events        auction-simple                   hardware-security-module        test-network-nano-bash
MAINTAINERS.md      asset-transfer-ledger-queries bin                             high-throughput                 token-erc-1155
pcs@Pranav:~/BC_Pract/fabric/fabric-samples$
WSL: Ubuntu   ⊗ 0 ⚠ 0   0
```

cd test-network

ls

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$ ls
CHAINCODE_AS_A_SERVICE_TUTORIAL.md  addOrg3     compose   monitordocker.sh  network.sh     prometheus-grafana  setOrgEnv.sh
README.md                           bft-config  configtx  network.config    organizations  scripts             system-genesis-block
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$
```

Remove any containers or artifacts

./network.sh down

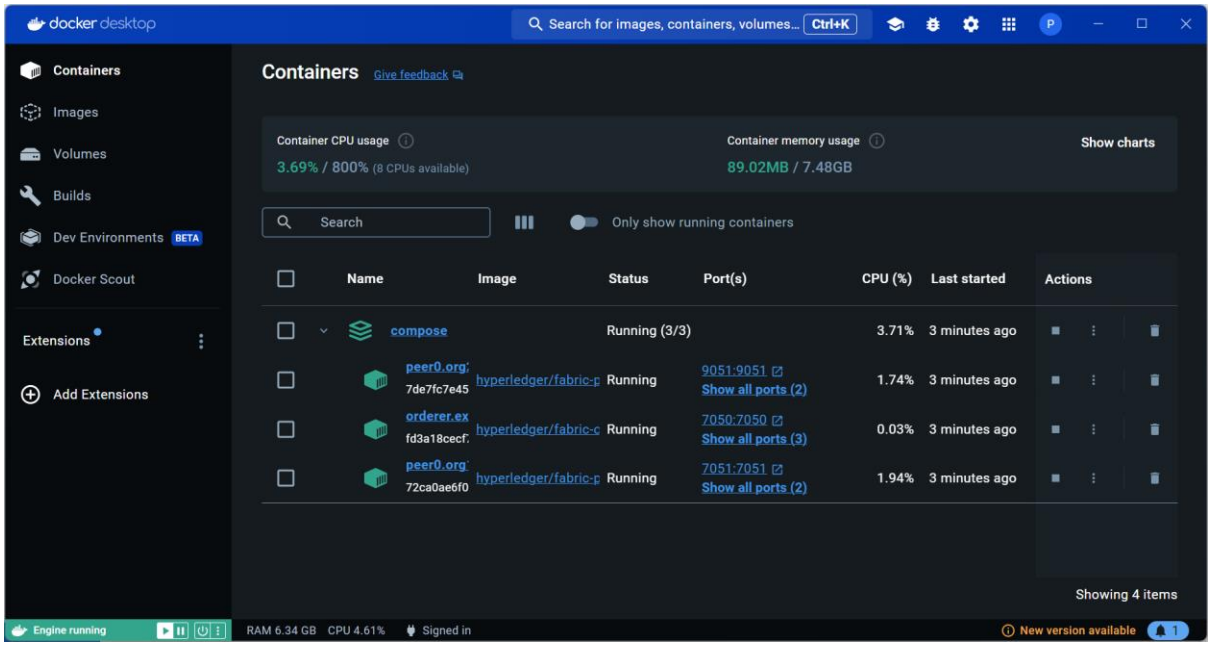
```
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$ ./network.sh down
Using docker and docker-compose
Stopping network
WARN[0000] /home/pcs/BC_Pract/fabric/fabric-samples/test-network/compose/compose-bft-test-net.yaml: `version` is obsolete
WARN[0000] /home/pcs/BC_Pract/fabric/fabric-samples/test-network/compose/docker/docker-compose-bft-test-net.yaml: `version` is obsolete
WARN[0000] /home/pcs/BC_Pract/fabric/fabric-samples/test-network/compose/compose-couch.yaml: `version` is obsolete
WARN[0000] /home/pcs/BC_Pract/fabric/fabric-samples/test-network/compose/docker/docker-compose-couch.yaml: `version` is obsolete
WARN[0000] /home/pcs/BC_Pract/fabric/fabric-samples/test-network/compose/compose-ca.yaml: `version` is obsolete
WARN[0000] /home/pcs/BC_Pract/fabric/fabric-samples/test-network/compose/docker/docker-compose-ca.yaml: `version` is obsolete
WARN[0000] /home/pcs/BC_Pract/fabric/fabric-samples/test-network/addOrg3/compose/compose-org3.yaml: `version` is obsolete
WARN[0000] /home/pcs/BC_Pract/fabric/fabric-samples/test-network/addOrg3/compose/docker/docker-compose-org3.yaml: `version` is obsolete
WARN[0000] /home/pcs/BC_Pract/fabric/fabric-samples/test-network/addOrg3/compose/compose-couch-org3.yaml: `version` is obsolete
WARN[0000] /home/pcs/BC_Pract/fabric/fabric-samples/test-network/addOrg3/compose/docker/docker-compose-couch-org3.yaml: `version` is obsolete
WARN[0000] /home/pcs/BC_Pract/fabric/fabric-samples/test-network/addOrg3/compose/compose-ca-org3.yaml: `version` is obsolete
WARN[0000] /home/pcs/BC_Pract/fabric/fabric-samples/test-network/addOrg3/compose/docker/docker-compose-ca-org3.yaml: `version` is obsolete
[+] Running 7/0
 ✓ Volume compose_peer0.org1.example.com  Removed
 ✓ Volume compose_peer0.org2.example.com  Removed
 ✓ Volume compose_peer0.org3.example.com  Removed
 ✓ Volume compose_orderer4.example.com    Removed
 ✓ Volume compose_orderer.example.com     Removed
 ✓ Volume compose_orderer2.example.com    Removed
 ✓ Volume compose_orderer3.example.com    Removed
Error response from daemon: get docker_orderer.example.com: no such volume
Error response from daemon: get docker_peer0.org1.example.com: no such volume
Error response from daemon: get docker_peer0.org2.example.com: no such volume
Removing remaining containers
Removing generated chaincode docker images
Unable to find image 'busybox:latest' locally
```

Up the network

./network.sh up

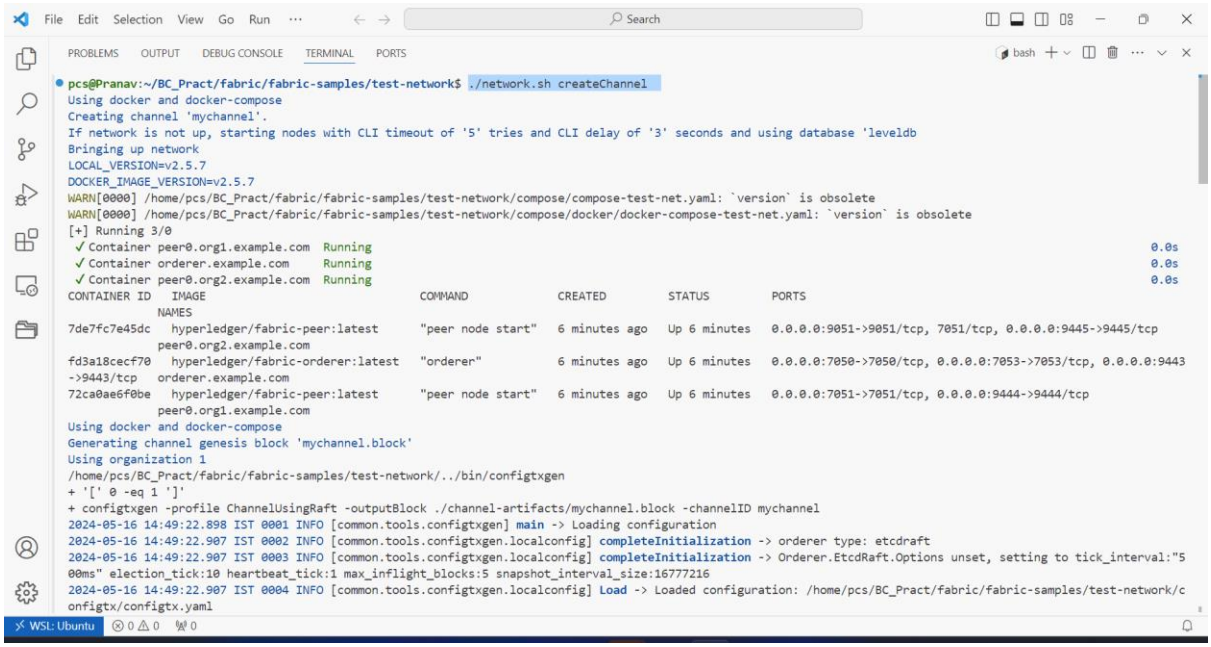
```
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$ ./network.sh up
Using docker and docker-compose
Starting nodes with CLI timeout of '5' tries and CLI delay of '3' seconds and using database 'leveldb' with crypto from 'cryptogen'
LOCAL_VERSION=v2.5.7
DOCKER_IMAGE_VERSION=v2.5.7
/home/pcs/BC_Pract/fabric/fabric-samples/test-network/../bin/cryptogen
Generating certificates using cryptogen tool
Creating Org1 Identities
+ cryptogen generate --config=./organizations/cryptogen/crypto-config-org1.yaml --output=organizations
org1.example.com
+ res=0
Creating Org2 Identities
+ cryptogen generate --config=./organizations/cryptogen/crypto-config-org2.yaml --output=organizations
org2.example.com
+ res=0
Creating Orderer Org Identities
+ cryptogen generate --config=./organizations/cryptogen/crypto-config-orderer.yaml --output=organizations
+ res=0
Generating CCP files for Org1 and Org2
WARN[0000] /home/pcs/BC_Pract/fabric/fabric-samples/test-network/compose/compose-test-net.yaml: `version` is obsolete
WARN[0000] /home/pcs/BC_Pract/fabric/fabric-samples/test-network/compose/docker/docker-compose-test-net.yaml: `version` is obsolete
[+] Running 7/7
 ✓ Network fabric_test                    Created                                                    0.1s
 ✓ Volume "compose_orderer.example.com"   Created                                                    0.0s
 ✓ Volume "compose_peer0.org1.example.com" Created                                                   0.0s
 ✓ Volume "compose_peer0.org2.example.com" Created                                                   0.0s
 ✓ Container peer0.org1.example.com       Started                                                    0.3s
 ✓ Container peer0.org2.example.com       Started                                                    0.3s
 ✓ Container orderer.example.com          Started                                                    0.4s
```

```
 ✓ Container peer0.org1.example.com       Started                                                    0.3s
 ✓ Container peer0.org2.example.com       Started                                                    0.3s
 ✓ Container orderer.example.com          Started                                                    0.4s
CONTAINER ID   IMAGE                          COMMAND             CREATED        STATUS                 PORTS
                 NAMES
7de7fc7e45dc   hyperledger/fabric-peer:latest      "peer node start"   1 second ago   Up Less than a second   0.0.0.0:9051->9051/tcp, 7051/tcp, 0.0.0.0:9445->9445
/tcp             peer0.org2.example.com
fd3a18cecf70   hyperledger/fabric-orderer:latest   "orderer"           1 second ago   Up Less than a second   0.0.0.0:7050->7050/tcp, 0.0.0.0:7053->7053/tcp, 0.0.
0.0:9443->9443/tcp   orderer.example.com
72ca0ae6f0be   hyperledger/fabric-peer:latest      "peer node start"   1 second ago   Up Less than a second   0.0.0.0:7051->7051/tcp, 0.0.0.0:9444->9444/tcp
                 peer0.org1.example.com
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$
Ubuntu  ⊗ 0 ⚠ 0  0
```



Create a channel

./network.sh createChannel



Deploy chaincode on peers and channel

./network.sh deployCC -ccn basic -ccp ../asset-transfer-basic/chaincode-javascript -ccl javascript

```
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$ ./network.sh deployCC -ccn basic -ccp ../asset-transfer-basic/chaincode-javascript -ccl javascript
Using docker and docker-compose
deploying chaincode on channel 'mychannel'
executing with the following
- CHANNEL_NAME: mychannel
- CC_NAME: basic
- CC_SRC_PATH: ../asset-transfer-basic/chaincode-javascript
- CC_SRC_LANGUAGE: javascript
- CC_VERSION: 1.0.1
- CC_SEQUENCE: auto
- CC_END_POLICY: NA
- CC_COLL_CONFIG: NA
- CC_INIT_FCN: NA
- DELAY: 3
- MAX_RETRY: 5
- VERBOSE: false
executing with the following
- CC_NAME: basic
- CC_SRC_PATH: ../asset-transfer-basic/chaincode-javascript
- CC_SRC_LANGUAGE: javascript
- CC_VERSION: 1.0.1
+ '[' false = true ']'
+ peer lifecycle chaincode package basic.tar.gz --path ../asset-transfer-basic/chaincode-javascript --lang node --label basic_1.0.1
+ res=0
Chaincode is packaged
Installing chaincode on peer0.org1...
Using organization 1
+ peer lifecycle chaincode queryinstalled --output json
+ jq -r 'try (.installed_chaincodes[].package_id)'
+ grep '^basic_1.0.1:f28a294429ebb36f96bab0d39e72a12c165b73705584e1c8239b8bb73c33ac24$'
+ test 1 -ne 0
+ peer lifecycle chaincode install basic.tar.gz
```

```
{
        "approvals": {
                "Org1MSP": true,
                "Org2MSP": true
        }
}
Checking the commit readiness of the chaincode definition successful on peer0.org1 on channel 'mychannel'
Using organization 2
Checking the commit readiness of the chaincode definition on peer0.org2 on channel 'mychannel'...
Attempting to check the commit readiness of the chaincode definition on peer0.org2, Retry after 3 seconds.
+ peer lifecycle chaincode checkcommitreadiness --channelID mychannel --name basic --version 1.0.1 --sequence 1 --output json
+ res=0
{
        "approvals": {
                "Org1MSP": true,
                "Org2MSP": true
        }
}
Checking the commit readiness of the chaincode definition successful on peer0.org2 on channel 'mychannel'
Using organization 1
Using organization 2
+ peer lifecycle chaincode commit -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls --cafile /home/pcs/BC_Pract/fabric/fabric-samples/test-
network/organizations/ordererOrganizations/example.com/tlsca/tlsca.example.com-cert.pem --channelID mychannel --name basic --peerAddresses localhost:7051 --tlsRoo
tCertFiles /home/pcs/BC_Pract/fabric/fabric-samples/test-network/organizations/peerOrganizations/org1.example.com/tlsca/tlsca.org1.example.com-cert.pem --peerAddr
esses localhost:9051 --tlsRootCertFiles /home/pcs/BC_Pract/fabric/fabric-samples/test-network/organizations/peerOrganizations/org2.example.com/tlsca/tlsca.org2.ex
ample.com-cert.pem --version 1.0.1 --sequence 1
+ res=0
2024-05-16 14:53:29.792 IST 0001 INFO [chaincodeCmd] ClientWait -> txid [316a394ecf3a3b61aadc1b3baff65f6a549df752b0f16dc0f28020662fd64946] committed with status (
VALID) at localhost:9051
2024-05-16 14:53:29.797 IST 0002 INFO [chaincodeCmd] ClientWait -> txid [316a394ecf3a3b61aadc1b3baff65f6a549df752b0f16dc0f28020662fd64946] committed with status (
VALID) at localhost:7051
Chaincode definition committed on channel 'mychannel'
Using organization 1
```

```
}
Checking the commit readiness of the chaincode definition successful on peer0.org2 on channel 'mychannel'
Using organization 1
Using organization 2
+ peer lifecycle chaincode commit -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls --cafile /home/pcs/BC_Pract/fabric/fabric-samples/test-
network/organizations/ordererOrganizations/example.com/tlsca/tlsca.example.com-cert.pem --channelID mychannel --name basic --peerAddresses localhost:7051 --tlsRoo
tCertFiles /home/pcs/BC_Pract/fabric/fabric-samples/test-network/organizations/peerOrganizations/org1.example.com/tlsca/tlsca.org1.example.com-cert.pem --peerAddr
esses localhost:9051 --tlsRootCertFiles /home/pcs/BC_Pract/fabric/fabric-samples/test-network/organizations/peerOrganizations/org2.example.com/tlsca/tlsca.org2.ex
ample.com-cert.pem --version 1.0.1 --sequence 1
+ res=0
2024-05-16 14:53:29.792 IST 0001 INFO [chaincodeCmd] ClientWait -> txid [316a394ecf3a3b61aadc1b3baff65f6a549df752b0f16dc0f28020662fd64946] committed with status (
VALID) at localhost:9051
2024-05-16 14:53:29.797 IST 0002 INFO [chaincodeCmd] ClientWait -> txid [316a394ecf3a3b61aadc1b3baff65f6a549df752b0f16dc0f28020662fd64946] committed with status (
VALID) at localhost:7051
Chaincode definition committed on channel 'mychannel'
Using organization 1
Querying chaincode definition on peer0.org1 on channel 'mychannel'...
Attempting to Query committed status on peer0.org1, Retry after 3 seconds.
+ peer lifecycle chaincode querycommitted --channelID mychannel --name basic
+ res=0
Committed chaincode definition for chaincode 'basic' on channel 'mychannel':
Version: 1.0.1, Sequence: 1, Endorsement Plugin: escc, Validation Plugin: vscc, Approvals: [Org1MSP: true, Org2MSP: true]
Query chaincode definition successful on peer0.org1 on channel 'mychannel'
Using organization 2
Querying chaincode definition on peer0.org2 on channel 'mychannel'...
Attempting to Query committed status on peer0.org2, Retry after 3 seconds.
+ peer lifecycle chaincode querycommitted --channelID mychannel --name basic
+ res=0
Committed chaincode definition for chaincode 'basic' on channel 'mychannel':
Version: 1.0.1, Sequence: 1, Endorsement Plugin: escc, Validation Plugin: vscc, Approvals: [Org1MSP: true, Org2MSP: true]
Query chaincode definition successful on peer0.org2 on channel 'mychannel'
Chaincode initialization is not required
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$
```

Interacting with the network

Set the path for peer binary and config for core.yaml

export PATH=${PWD}/../bin:$PATH

export FABRIC_CFG_PATH=$PWD/../config/

Set the environment variables to operate Peer as Org1

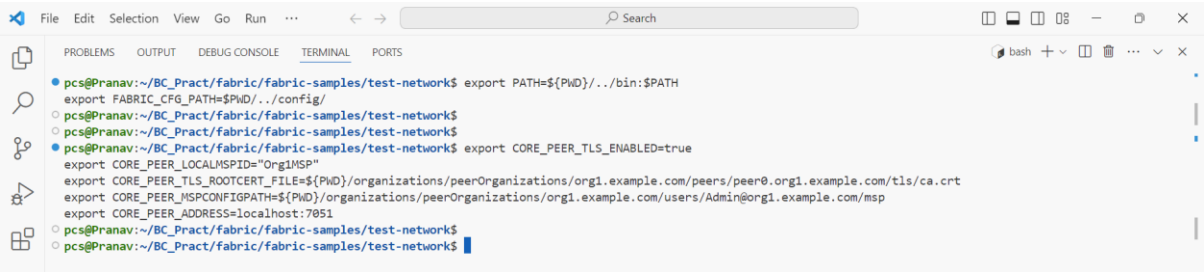
export CORE_PEER_TLS_ENABLED=true

export CORE_PEER_LOCALMSPID="Org1MSP"

export CORE_PEER_TLS_ROOTCERT_FILE=${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt

export CORE_PEER_MSPCONFIGPATH=${PWD}/organizations/peerOrganizations/org1.example.com/users/Admin@org1.example.com/msp

export CORE_PEER_ADDRESS=localhost:7051



Command to initialize the ledger with assets

peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n basic --peerAddresses localhost:7051 --tlsRootCertFiles "${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt" --peerAddresses localhost:9051 --tlsRootCertFiles "${PWD}/organizations/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt" -c '{"function":"InitLedger","Args":[]}'

```
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$ peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls --cafile
"${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n basic --peerAddress
es localhost:7051 --tlsRootCertFiles "${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt" --peerAddresses localhost:9
051 --tlsRootCertFiles "${PWD}/organizations/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt" -c '{"function":"InitLedger","Args":[]}'
2024-05-16 15:45:16.528 IST 0001 INFO [chaincodeCmd] chaincodeInvokeOrQuery -> Chaincode invoke successful. result: status:200
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$
WSL: Ubuntu
```

Query the ledger

peer chaincode query -C mychannel -n basic -c '{"Args":["GetAllAssets"]}'

```
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$ peer chaincode query -C mychannel -n basic -c '{"Args":["GetAllAssets"]}'
[{"AppraisedValue":300,"Color":"blue","ID":"asset1","Owner":"Tomoko","Size":5,"docType":"asset"},{"AppraisedValue":400,"Color":"red","ID":"asset2","Owner":"Brad",
"Size":5,"docType":"asset"},{"AppraisedValue":500,"Color":"green","ID":"asset3","Owner":"Jin Soo","Size":10,"docType":"asset"},{"AppraisedValue":600,"Color":"yell
ow","ID":"asset4","Owner":"Max","Size":10,"docType":"asset"},{"AppraisedValue":700,"Color":"black","ID":"asset5","Owner":"Adriana","Size":15,"docType":"asset"},{"
AppraisedValue":800,"Color":"white","ID":"asset6","Owner":"Michel","Size":15,"docType":"asset"}]
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$
```

Transfer the asset

peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n basic --peerAddresses localhost:7051 --tlsRootCertFiles "${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt" --peerAddresses localhost:9051 --tlsRootCertFiles "${PWD}/organizations/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt" -c '{"function":"TransferAsset","Args":["asset6","Christopher"]}'

```
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$ peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls --cafile
"${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n basic --peerAddress
es localhost:7051 --tlsRootCertFiles "${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt" --peerAddresses localhost:9
051 --tlsRootCertFiles "${PWD}/organizations/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt" -c '{"function":"TransferAsset","Args":["
asset6","Christopher"]}'
2024-05-16 15:49:13.048 IST 0001 INFO [chaincodeCmd] chaincodeInvokeOrQuery -> Chaincode invoke successful. result: status:200 payload:"Michel"
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$ peer chaincode query -C mychannel -n basic -c '{"Args":["GetAllAssets"]}'
[{"AppraisedValue":300,"Color":"blue","ID":"asset1","Owner":"Tomoko","Size":5,"docType":"asset"},{"AppraisedValue":400,"Color":"red","ID":"asset2","Owner":"Brad",
"Size":5,"docType":"asset"},{"AppraisedValue":500,"Color":"green","ID":"asset3","Owner":"Jin Soo","Size":10,"docType":"asset"},{"AppraisedValue":600,"Color":"yell
ow","ID":"asset4","Owner":"Max","Size":10,"docType":"asset"},{"AppraisedValue":700,"Color":"black","ID":"asset5","Owner":"Adriana","Size":15,"docType":"asset"},{"
AppraisedValue":800,"Color":"white","ID":"asset6","Owner":"Christopher","Size":15,"docType":"asset"}]
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$
```

Lets query the ledger from Org2 peer

Set the environment variables to operate Peer as Org2

export CORE_PEER_TLS_ENABLED=true

export CORE_PEER_LOCALMSPID="Org2MSP"

export CORE_PEER_TLS_ROOTCERT_FILE=${PWD}/organizations/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt

export CORE_PEER_MSPCONFIGPATH=${PWD}/organizations/peerOrganizations/org2.example.com/users/Admin@org2.example.com/msp

export CORE_PEER_ADDRESS=localhost:9051

```
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$ export CORE_PEER_TLS_ENABLED=true
export CORE_PEER_LOCALMSPID="Org2MSP"
export CORE_PEER_TLS_ROOTCERT_FILE=${PWD}/organizations/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt
export CORE_PEER_MSPCONFIGPATH=${PWD}/organizations/peerOrganizations/org2.example.com/users/Admin@org2.example.com/msp
export CORE_PEER_ADDRESS=localhost:9051
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$ peer chaincode query -C mychannel -n basic -c '{"Args":["GetAllAssets"]}'
[{"AppraisedValue":300,"Color":"blue","ID":"asset1","Owner":"Tomoko","Size":5,"docType":"asset"},{"AppraisedValue":400,"Color":"red","ID":"asset2","Owner":"Brad",
"Size":5,"docType":"asset"},{"AppraisedValue":500,"Color":"green","ID":"asset3","Owner":"Jin Soo","Size":10,"docType":"asset"},{"AppraisedValue":600,"Color":"yell
ow","ID":"asset4","Owner":"Max","Size":10,"docType":"asset"},{"AppraisedValue":700,"Color":"black","ID":"asset5","Owner":"Adriana","Size":15,"docType":"asset"},{"
AppraisedValue":800,"Color":"white","ID":"asset6","Owner":"Christopher","Size":15,"docType":"asset"}]
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$
```

Query the ledger

peer chaincode query -C mychannel -n basic -c '{"Args":["ReadAsset","asset6"]}'

```
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$ peer chaincode query -C mychannel -n basic -c '{"Args":["ReadAsset","asset6"]}'
{"AppraisedValue":800,"Color":"white","ID":"asset6","Owner":"Christopher","Size":15,"docType":"asset"}
pcs@Pranav:~/BC_Pract/fabric/fabric-samples/test-network$
```

Bring the network down

./network.sh down

Practical 8

Demonstrate the running of the blockchain node (create node using solidity and run).

To check if the prerequisites (Node.js, npm, and Truffle) are installed, you can run the following commands:

Step 1: Prerequisites

Install Node.js

https://nodejs.org/en/download/prebuilt-installer

Execute the following Commands:

npm install -g truffle
npm install -g ganache-cli

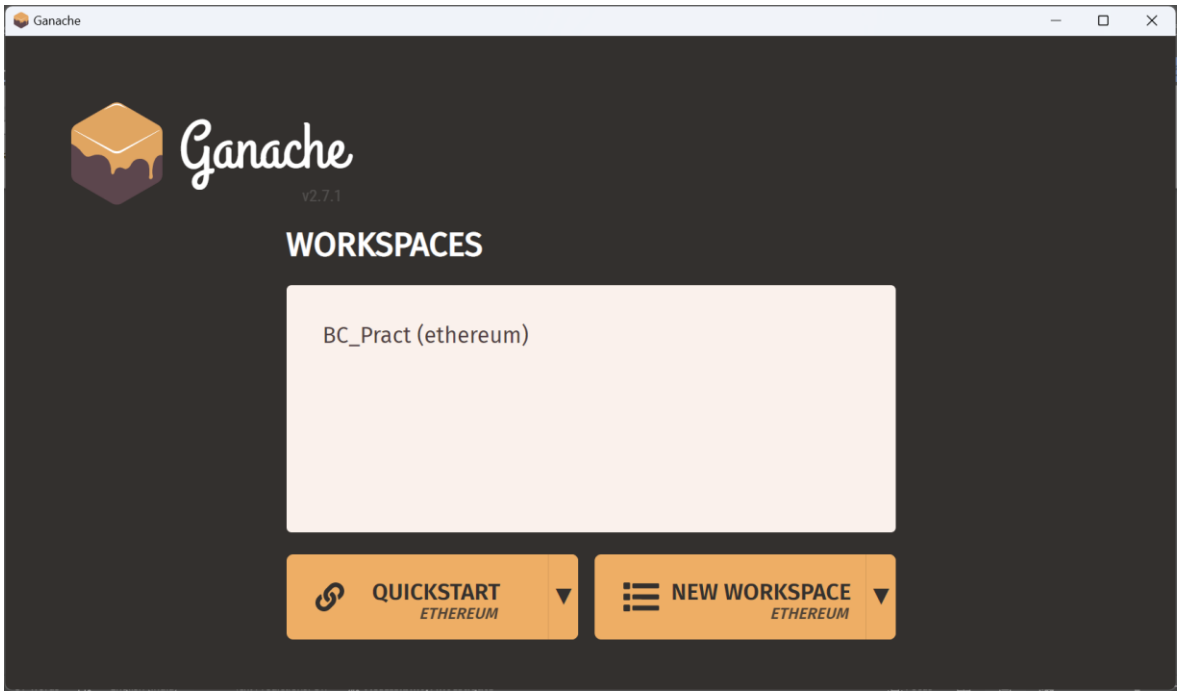
1) **Check Node.js and npm installation:**
   node -v
   npm -v

2) **Check Truffle installation:**
   truffle version

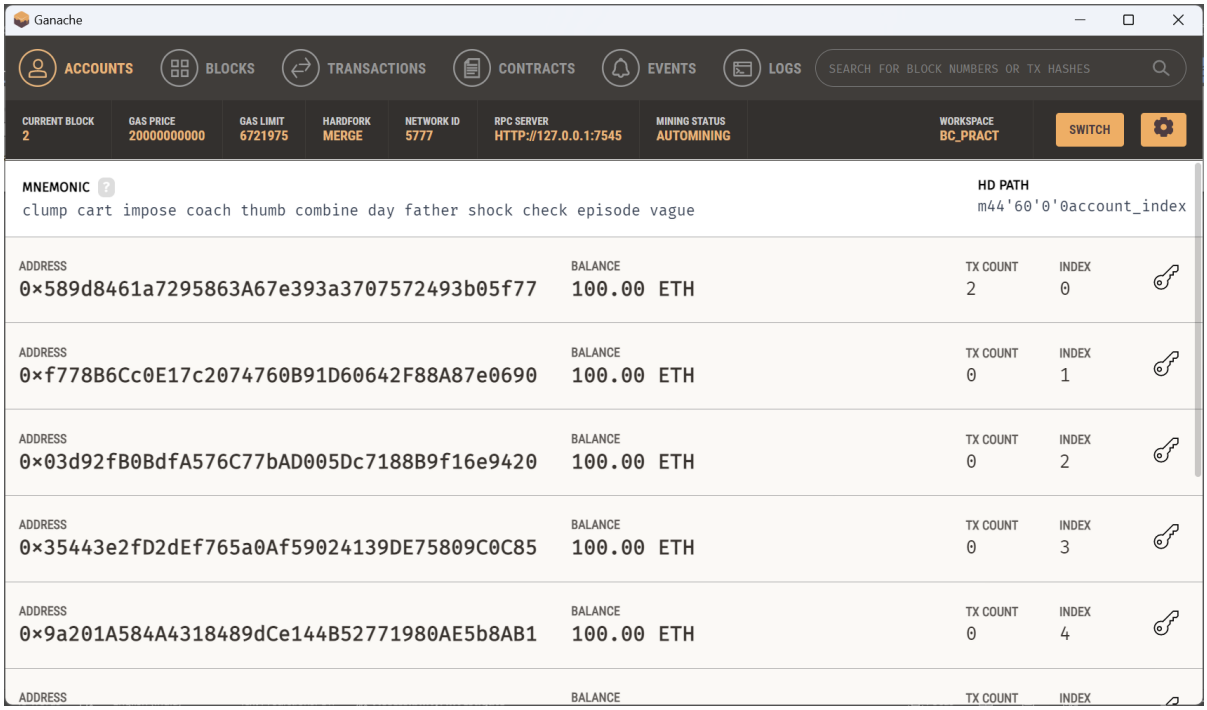
```
PROBLEMS 4   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

Microsoft Windows [Version 10.0.22631.3447]
(c) Microsoft Corporation. All rights reserved.

C:\Users\prana\Desktop\BC_Pract>node -v
v20.14.0

C:\Users\prana\Desktop\BC_Pract>npm -v
10.7.0

C:\Users\prana\Desktop\BC_Pract>truffle version
Truffle v5.11.5 (core: 5.11.5)
Ganache v7.9.1
Solidity v0.5.16 (solc-js)
Node v20.14.0
Web3.js v1.10.0

C:\Users\prana\Desktop\BC_Pract>
```

3) Install Ganache

https://archive.trufflesuite.com/ganache/



4) Create a new Workspace (BC_Pract)



Step 2: Initialize a Truffle Project

1) **Create a new directory for your project:**
   mkdir myProj
   cd myProj

2) **Initialize the Truffle project:**
   truffle init

Step 3: Create a Solidity Smart Contract

1) **Navigate to the Contracts directory(myProj/contracts):**

SimpleStorage.sol

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

contract SimpleStorage {
   uint256 public storedData;

   function set(uint256 x) public {
      storedData = x;
   }

   function get() public view returns (uint256) {
      return storedData;
   }
}
```

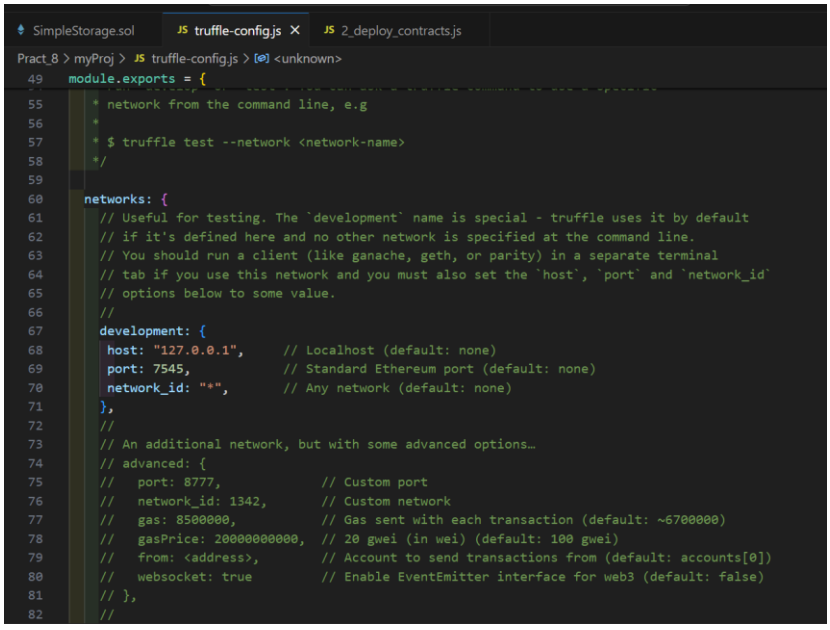
2) **Compile the Smart Contract**
   Command: truffle compile

C:\Users\prana\Desktop\BC_Pract\Pract_8\myProj>truffle compile

Step 4: Configure Truffle to Use Ganache

Open the truffle-config.js file and configure the development network to use Ganache. Update the networks section:

```
module.exports = {
  networks: {
    development: {
      host: "127.0.0.1",
      port: 7545,  // Match the port Ganache is using
      network_id: "*" // Match any network id
    }
  },
  compilers: {
    solc: {
      version: "0.8.0"  // Specify the Solidity compiler version
    }
  }
};
```

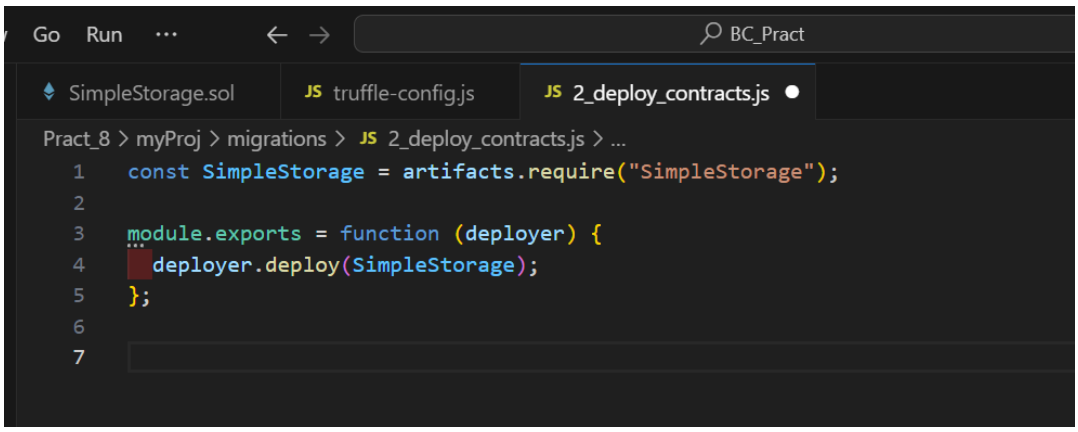


Step 5: Migrate the Smart Contract to Ganache

1) **Start Ganache (open the Ganache application and start a new workspace(BC_Pract)).**
2) **Create a migration script in the migrations directory (e.g., deploy_contracts.js):**

Pract_8\myProj\migrations\2_deploy_contracts.js

```
const SimpleStorage = artifacts.require("SimpleStorage");

module.exports = function (deployer) {
  deployer.deploy(SimpleStorage);
};
```



3) **Run the migration:**
   Command: truffle migrate

C:\Users\prana\Desktop\BC_Pract\Pract_8\myProj>truffle migrate

```
C:\Users\prana\Desktop\BC_Pract\Pract_8\myProj>truffle migrate

Compiling your contracts...
===========================
✓ Fetching solc version list from solc-bin. Attempt #1
✓ Downloading compiler. Attempt #1.
> Compiling .\contracts\SimpleStorage.sol
> Artifacts written to C:\Users\prana\Desktop\BC_Pract\Pract_8\myProj\build\contracts
> Compiled successfully using:
   - solc: 0.8.0+commit.c7dfd78e.Emscripten.clang


Starting migrations...
======================
> Network name:    'development'
> Network id:      5777
> Block gas limit: 6721975 (0x6691b7)
```

```
2_deploy_contracts.js
=====================

   Deploying 'SimpleStorage'
   -------------------------
   > transaction hash:    0xe6f72fa4e5dfe58ae8d45d96b8619cc88f79d07edc96964f872cf565528d7827
   > Blocks: 0            Seconds: 0
   > contract address:    0x06Bb10be4AdccA7BcFB491f9151d8c4c1600c22F
   > block number:        1
   > block timestamp:     1717939680
   > account:             0x589d8461a7295863A67e393a3707572493b05f77
   > balance:             99.999548117875
   > gas used:            133891 (0x20b03)
   > gas price:           3.375 gwei
   > value sent:          0 ETH
   > total cost:          0.000451882125 ETH

   > Saving artifacts
   -------------------------------------
   > Total cost:     0.000451882125 ETH

Summary
=======
> Total deployments:   1
> Final cost:          0.000451882125 ETH


C:\Users\prana\Desktop\BC_Pract\Pract_8\myProj>
```

Step 6: Interact with the Deployed Contract

1) Open the new command prompt:

Command: truffle console

C:\Users\prana\Desktop\BC_Pract\Pract_8\myProj>truffle console

2) Interact with the deployed contract:

Execute the following commands one-by-one

```
let instance = await SimpleStorage.deployed()
await instance.set(42)
let value = await instance.get()
value.toString()  // Output should be '42'
```

```
C:\Users\prana\Desktop\BC_Pract\Pract_8\myProj>truffle console
truffle(development)> let instance = await SimpleStorage.deployed()
undefined
truffle(development)> await instance.set(42)
{
  tx: '0xfa113e5fe6f6a4a47000d9490a02732f220c99df34568910723822d5daf9ac5f',
  receipt: {
    transactionHash: '0xfa113e5fe6f6a4a47000d9490a02732f220c99df34568910723822d5daf9ac5f',
    transactionIndex: 0,
    blockNumber: 2,
    blockHash: '0xef54b0b98dc1b4b30f61fc358e3b1b43a11f7bd9faf1db7000d7d3ef89a2c4b8',
    from: '0x589d8461a7295863a67e393a3707572493b05f77',
    to: '0x06bb10be4adcca7bcfb491f9151d8c4c1600c22f',
    cumulativeGasUsed: 43724,
    gasUsed: 43724,
    contractAddress: null,
    logs: [],
    logsBloom: '0x00000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000
    status: true,
    effectiveGasPrice: 3269982152,
    type: '0x2',
    rawLogs: []
  },
  logs: []
}
truffle(development)> let value = await instance.get()
undefined
truffle(development)>
undefined
```

```
truffle(development)>
undefined
truffle(development)> value.toString()
'42'
truffle(development)>
(To exit, press Ctrl+C again or Ctrl+D or type .exit)
truffle(development)>

C:\Users\prana\Desktop\BC_Pract\Pract_8\myProj>
```

Practical 9

Demonstrate the use of Bitcoin API:

Aim:

```
import requests

# Task 1: Get information regarding the current block
def get_current_block_info():
    response = requests.get("https://blockchain.info/latestblock")
    block_info = response.json()
    print("Current block information:")
    print("Block height:", block_info['height'])
    print("Block hash:", block_info['hash'])
    print("Block index:", block_info['block_index'])
    print("Timestamp:", block_info['time'])

# Task 3: Get balance of an address
def get_address_balance(address):
    response = requests.get(f"https://blockchain.info/q/addressbalance/{address}")
    balance = float(response.text) / 10**8
    print("Balance of address", address, ":", balance, "BTC")

# Example usage
if __name__ == "__main__":
    # Task 1: Get information regarding the current block
    get_current_block_info()

    # Task 3: Get balance of an address
    address = "3Dh2ft6UsqjbTNzs5zrp7uK17Gqg1Pg5u5"
    get_address_balance(address)
```

Output:

```
Current block information:
Block height: 854032
Block hash: 0000000000000000000627e8cc662c4cdfe178f0f43875dd8dcfff5b548b547
Block index: 854032
Timestamp: 1721999061
Balance of address 3Dh2ft6UsqjbTNzs5zrp7uK17Gqg1Pg5u5 : 0.0 BTC
```