MSC.IT PART II

BLOCKCHAIN PRACTICAL JOURNAL

Index

Sr.No	Topic		Date	Remark
1	**Practical 1 : Write the following programs for Blockchain in Python :**			
	(I). A simple client class that generates the private and public keys by using the built in Python RSA algorithm and test it (II). A transaction class to send and receive money and test it .			
2	**Practical 2 : Write the following programs for Blockchain in Python :** (I).Create multiple transactions and display them . (II). Create a blockchain, a genesis block and execute it .			
3	**Practical 3 : Write the following programs for Blockchain in Python :** (I).Create a mining function and test it . (II).Add blocks to the miner and dump the blockchain .			
4	**Practical 4 : Implement and demonstrate the use of the following in Solidity :** (I).Varaible (II).Operators (III).Loops (IV).Decision Making (V).Strings			
5	**Practical 5 : Implement and demonstrate the use of the following in Solidity :** (I).Arrays (II).Enums (III).Structs (IV).Mappings (V).Coversations (VI).Ether Units (VII).Special Variables			

Index

Practical 1

Write the following programs for Blockchain in Python :

(I) A simple client class that generates the private and public keys by using the built in Python RSA algorithm and test it

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

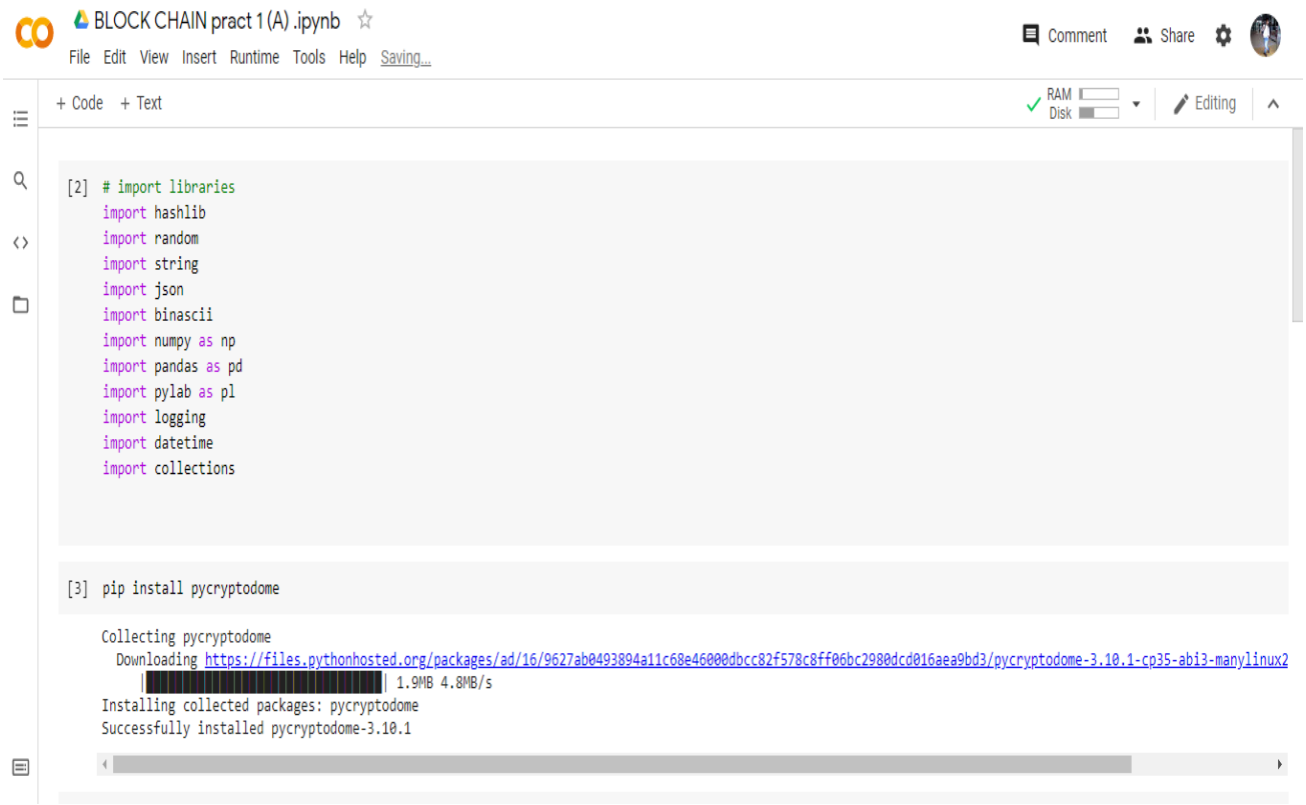
→

```
# import libraries
import hashlib
import random
import string
import json
import binascii
import numpy as np
import pandas as pd
import pylab as pl
import logging
import datetime
import collections
 pip install pycryptodome
 # following imports are required by PKI
import Crypto
import Crypto.Random
from Crypto.Hash import SHA
from Crypto.PublicKey import RSA
from Crypto.Signature import PKCS1_v1_5


import binascii
```

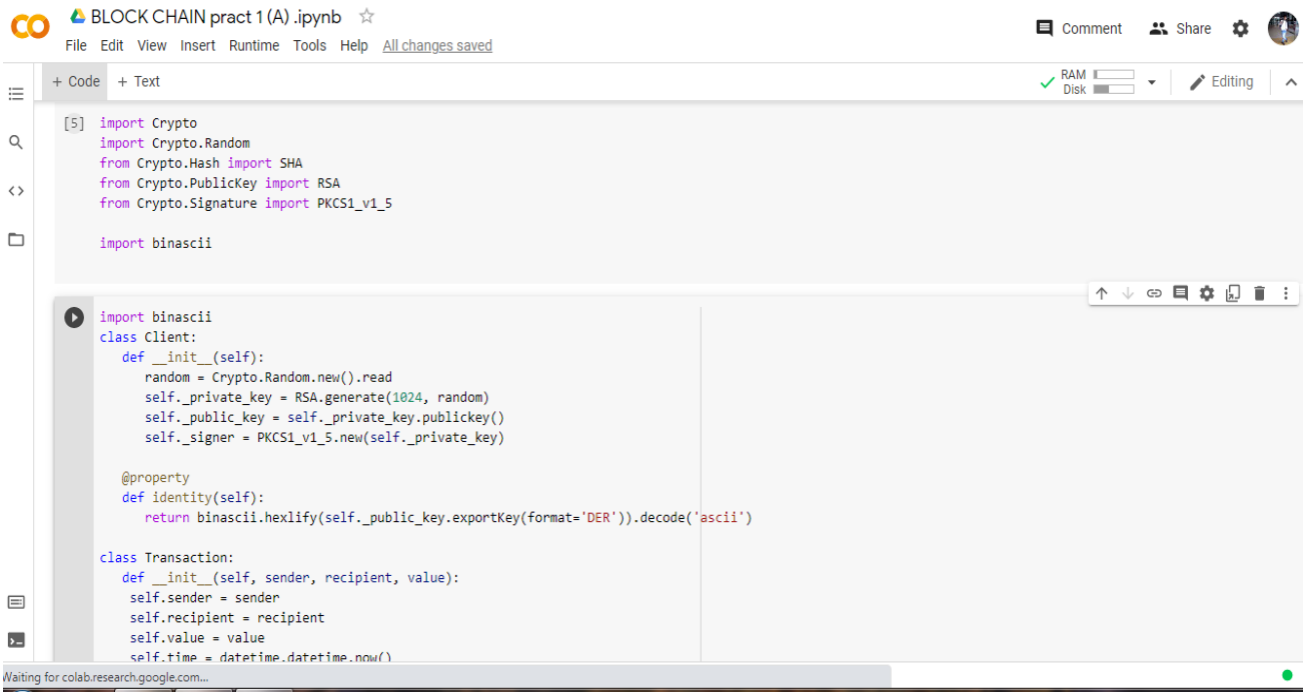
```
class Client:
  def __init__(self):
    random = Crypto.Random.new().read
    self._private_key = RSA.generate(1024, random)
    self._public_key = self._private_key.publickey()
    self._signer = PKCS1_v1_5.new(self._private_key)

  @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')
Dinesh = Client()
    Ramesh = Client()
t = Transaction(Dinesh,Ramesh.identity,5.0)
signature = t.sign_transaction()
print (signature)
```

Output:



:


BLOCK CHAIN pract 1 (A) .ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code
+ Text
RAM
Disk
Editing
[5] import Crypto
import Crypto.Random
from Crypto.Hash import SHA
from Crypto.PublicKey import RSA
from Crypto.Signature import PKCS1_v1_5
import binascii
import binascii
class Client:
def __init__(self):
random = Crypto.Random.new().read
self._private_key = RSA.generate(1024, random)
self._public_key = self._private_key.publickey()
self._signer = PKCS1_v1_5.new(self._private_key)
@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
Waiting for colab.research.google.com...


BLOCK CHAIN pract 1 (A) .ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code
+ Text
RAM
Disk
Editing
self.time = datetime.datetime.now()
def to_dict(self):
if self.sender == "Genesis":
identity = "Genesis"
else:
identity = self.sender.identity
return collections.OrderedDict({
'sender': identity,
'recipient': self.recipient,
'value': self.value,
'time' : self.time})
def sign_transaction(self):
private_key = self.sender._private_key
signer = PKCS1_v1_5.new(private_key)
h = SHA.new(str(self.to_dict()).encode('utf8'))
return binascii.hexlify(signer.sign(h)).decode('ascii')
Dinesh = Client()
Ramesh = Client()
t = Transaction(Dinesh,Ramesh.identity,5.0)
signature = t.sign_transaction()
print (signature)
656f0e37653d1bc3c538de01595a9c7e4a603e03740d1ed450f1ea32f568663cebff7ec9757cdd35d4f8b1a3524341528a48674981265b68a477c06d57271d6250f45d6e3946a4f8166fff50251956fa
Waiting for colab.research.google.com...

Practical 2

Write the following programs for Blockchain in Python :

(I).Create multiple transactions and display them .

(II). Create a blockchain, a genesis block and execute it .

→

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

transactions = []

Dinesh = Client()
Ramesh = Client()
Seema = Client()
Vijay = Client()

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

```
)

t1.sign_transaction()
transactions.append(t1)

t2 = Transaction(
   Dinesh,
   Seema.identity,
   6.0
)
t2.sign_transaction()
transactions.append(t2)
t3 = Transaction(
   Ramesh,
   Vijay.identity,
   2.0
)
t3.sign_transaction()
transactions.append(t3)
t4 = Transaction(
   Seema,
   Ramesh.identity,
   4.0
)
t4.sign_transaction()
transactions.append(t4)
t5 = Transaction(
   Vijay,
   Seema.identity,
```

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

```
)
t9.sign_transaction()
transactions.append(t9)
t10 = Transaction(
   Vijay,
   Ramesh.identity,
   3.0
)
t10.sign_transaction()
transactions.append(t10)

for transaction in transactions:
   display_transaction (transaction)
   print ('--------------')

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

last_block_hash = ""

Dinesh = Client()

t0 = Transaction (
   "Genesis",
   Dinesh.identity,
   500.0
```

```
)

block0 = Block()

block0.previous_block_hash = None
Nonce = None

block0.verified_transactions.append (t0)

digest = hash (block0)
last_block_hash = digest

TPCoins = []

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

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

Output:

```
[10] 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 ('-----')
```

```
[11] transactions = []
```

```
[12] Dinesh = Client()
     Ramesh = Client()
     Seema = Client()
     Vijay = Client()
```

```
[13] t1 = Transaction(
        Dinesh,
        Ramesh.identity,
        15.0
     )
```

```
[15] t1.sign_transaction()
     transactions.append(t1)
```

```
[16] t2 = Transaction(
        Dinesh,
        Seema.identity,
        6.0
     )
     t2.sign_transaction()
     transactions.append(t2)
     t3 = Transaction(
        Ramesh,
        Vijay.identity,
        2.0
     )
     t3.sign_transaction()
     transactions.append(t3)
     t4 = Transaction(
        Seema,
        Ramesh.identity,
        4.0
     )
     t4.sign_transaction()
     transactions.append(t4)
     t5 = Transaction(
        Vijay,
        Seema.identity,
        7.0
     )
     t5.sign_transaction()
     transactions.append(t5)
     t6 = Transaction(
        Ramesh,
        Seema.identity,
        3.0
     )
     t6.sign_transaction()
     transactions.append(t6)
     t7 = Transaction(
        Seema,
        Dinesh.identity,
        8.0
     )
     t7.sign_transaction()
     transactions.append(t7)
     t8 = Transaction(
        Seema,
        Ramesh.identity,
        1.0
     )
     t8.sign_transaction()
     transactions.append(t8)
     t9 = Transaction(
        Vijay,
        Dinesh.identity,
        5.0
     )
```

```
[16] t8.sign_transaction()
     transactions.append(t8)
     t9 = Transaction(
        Vijay,
        Dinesh.identity,
        5.0
     )
     t9.sign_transaction()
     transactions.append(t9)
     t10 = Transaction(
        Vijay,
        Ramesh.identity,
        3.0
     )
     t10.sign_transaction()
     transactions.append(t10)
```


```
[17] for transaction in transactions:
        display_transaction (transaction)
        print ('--------------')

     sender: 30819f300d06092a864886f70d010101050003818d0030818                f2feecba6d
     -----
     recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100b5c36acef717c85079d5
     -----
     value: 15.0
     -----
     time: 2021-05-05 07:37:46.751762
     -----
     --------------
     sender: 30819f300d06092a864886f70d010101050003818d0030818902818100096e3e436b160bf2feecba6d
     -----
     recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100b5c36acef717c85079d5
     -----
     value: 15.0
     -----
     time: 2021-05-05 07:37:46.751762
     -----
     --------------
     sender: 30819f300d06092a864886f70d010101050003818d0030818902818100096e3e436b160bf2feecba6d
     -----
     recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100a3b073014e3d9a1e3353
     -----
     value: 6.0
     -----
     time: 2021-05-05 07:38:17.042489
     -----
     --------------
     sender: 30819f300d06092a864886f70d010101050003818d0030818902818100b5c36acef717c85079d5eb8
     -----
     recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100967bf6965c7e25f7c711
     -----
     value: 2.0
     -----
     time: 2021-05-05 07:38:17.044049
     -----
     --------------
     sender: 30819f300d06092a864886f70d010101050003818d0030818902818100a3b073014e3d9a1e335365f
     -----
     recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100b5c36acef717c85079d5
     -----
     value: 4.0
     -----
     time: 2021-05-05 07:38:17.045503
     -----
     --------------
     sender: 30819f300d06092a864886f70d010101050003818d0030818902818100967bf6965c7e25f7c711a30
     -----
     recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100a3b073014e3d9a1e3353
     -----
     value: 7.0
     -----
```

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

```
[19] last_block_hash = ""
```

+ Code + Text

RAM Disk

Editing

```
[20] Dinesh = Client()
```

```
[21] t0 = Transaction (
        "Genesis",
        Dinesh.identity,
        500.0
     )
```

```
[22] block0 = Block()
```

```
[23] block0.previous_block_hash = None
     Nonce = None
```

```
[24] block0.verified_transactions.append (t0)
```

```
[25] digest = hash (block0)
     last_block_hash = digest
```

```
[26] 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 ('======================================')
```

```
[27] TPCoins = []
```

```
[30] 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 ('======================================')
```

```
[31] TPCoins.append (block0)
```

```
[32] dump_blockchain(TPCoins)
```

```
Number of blocks in the chain: 1
block # 0
sender: Genesis
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100df7d100622cc76eab161
-----
value: 500.0
-----
time: 2021-05-05 07:41:01.767902
-----
--------------
======================================
```

Practical 3

Write the following programs for Blockchain in Python :

(I).Create a mining function and test it .

(II).Add blocks to the miner and dump the blockchain .

→

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


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



last_transaction_index = 0


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

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

# Miner 2 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
# Miner 3 adds a block
block = Block()

for i in range(3):
  temp_transaction = transactions[last_transaction_index]
  #display_transaction (temp_transaction)
  # 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)
```

Full Output:

```
# import libraries
import hashlib
import random
import string
import json
import binascii
import numpy as np
import pandas as pd
import pylab as pl
import logging
import datetime
import collections
```

```
[ ] pip install pycryptodome
```

```
Collecting pycryptodome
  Downloading https://files.pythonhosted.org/packages/ad/16/9627ab0493894a11c68e46000dbcc
     |████████████████████████████████| 1.9MB 5.6MB/s
Installing collected packages: pycryptodome
Successfully installed pycryptodome-3.10.1
```

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

```
[ ] import binascii
class Client:
    def __init__(self):
        random = Crypto.Random.new().read
        self._private_key = RSA.generate(1024, random)
        self._public_key = self._private_key.publickey()
        self._signer = PKCS1_v1_5.new(self._private_key)

    @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')
```

```
[ ]
```

```
[ ] Dinesh = Client()
Ramesh = Client()
```

```
[ ] t = Transaction(Dinesh,Ramesh.identity,5.0)
```

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

```
251f28f6f523733739979611b404c755210b5b6a70f28d5eb3e3049f7dceea873e472f0df41a55152c71bb6f5
```

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

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

```
[ ] transactions = []
```

```
[ ] Dinesh = Client()
    Ramesh = Client()
    Seema = Client()
    Vijay = Client()
```

```
[ ] t1 = Transaction(
       Dinesh,
       Ramesh.identity,
       15.0
    )
```

```
[ ] t1.sign_transaction()
    transactions.append(t1)
```

```
[ ] t2 = Transaction(
       Dinesh,
       Seema.identity,
       6.0
    )
    t2.sign_transaction()
    transactions.append(t2)
    t3 = Transaction(
       Ramesh,
       Vijay.identity,
       2.0
    )
    t3.sign_transaction()
    transactions.append(t3)
    t4 = Transaction(
       Seema,
       Ramesh.identity,
       4.0
    )
    t4.sign_transaction()
    transactions.append(t4)
    t5 = Transaction(
       Vijay,
       Seema.identity,
       7.0
    )
    t5.sign_transaction()
    transactions.append(t5)
    t6 = Transaction(
       Ramesh,
       Seema.identity,
       3.0
    )
    t6.sign_transaction()
    transactions.append(t6)
    t7 = Transaction(
       Seema,
       Dinesh.identity,
       8.0
    )
    t7.sign_transaction()
    transactions.append(t7)
    t8 = Transaction(
       Seema,
       Ramesh.identity,
       1.0
    )
    t8.sign_transaction()
    transactions.append(t8)
    t9 = Transaction(
       Vijay,
       Dinesh.identity,
       5.0
    )
    t9.sign_transaction()
    transactions.append(t9)
    t10 = Transaction(
       Vijay,
       Ramesh.identity,
       3.0
    )
    t10.sign_transaction()
    transactions.append(t10)
```

```
[ ] for transaction in transactions:
        display_transaction (transaction)
        print ('--------------')
```

```
sender: 30819f300d06092a864886f70d010101050003818d00308189028181008f5e89df98216eb2c1e4713
-----
recipient: 30819f300d06092a864886f70d010101050003818d00308189028181008 2e40f44f9cef42cac49
-----
value: 15.0
-----
time: 2021-04-08 06:10:52.172517
-----
--------------
sender: 30819f300d06092a864886f70d010101050003818d00308189028181008f5e89df98216eb2c1e4713
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100a0b9b51bb0c81cbbad86
-----
value: 6.0
-----
time: 2021-04-08 06:11:40.567785
-----
--------------
sender: 30819f300d06092a864886f70d010101050003818d00308189028181008 2e40f44f9cef42cac492d1
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100d776af73071682cb494e
-----
value: 2.0
-----
time: 2021-04-08 06:11:40.569278
-----
--------------
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100a0b9b51bb0c81cbbad86384
-----
recipient: 30819f300d06092a864886f70d010101050003818d00308189028181008 2e40f44f9cef42cac49
-----
value: 4.0
-----
time: 2021-04-08 06:11:40.570658
-----
--------------
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100d776af73071682cb494e752
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100a0b9b51bb0c81cbbad86
-----
value: 7.0
-----
time: 2021-04-08 06:11:40.572173
-----
--------------
sender: 30819f300d06092a864886f70d010101050003818d00308189028181008 2e40f44f9cef42cac492d1
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100a0b9b51bb0c81cbbad86
-----
value: 3.0
-----
```

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

```
[ ] last_block_hash = ""
```

```
[ ] Dinesh = Client()
```

```
[ ] t0 = Transaction (
        "Genesis",
        Dinesh.identity,
        500.0
    )
```

```
[ ] block0 = Block()
```

```
[ ] block0.previous_block_hash = None
    Nonce = None
```

```
[ ] block0.verified_transactions.append (t0)
```

+ Code + Text Copy to Drive Connect Editing

```
[ ] digest = hash (block0)
    last_block_hash = digest
```

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

```
[ ] TPCoins = []
```

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

```
[ ] TPCoins.append (block0)
```

```
[ ] dump_blockchain(TPCoins)
```

```
Number of blocks in the chain: 1
block # 0
sender: Genesis
-----
recipient: 30819f300d06092a864886f70d010101050003818d003081802818100bae5c5a46a1591af94c0
-----
value: 500.0
-----
time: 2021-04-08 06:16:28.464142
-----
--------------
====================================
```

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

```
[ ] def mine(message, difficulty=1):
       assert difficulty >= 1
       prefix = '1' * difficulty
       for i in range(1000):
          digest = sha256(str(hash(message)) + str(i))
          if digest.startswith(prefix):
             print ("after " + str(i) + " iterations found nonce: "+ digest)
          return digest
```

```
[ ] mine ("test message", 2)
```

```
'938c72d2197fa6c2294df7bc8cea6be98769aad888e3cfa15558c78469394ebd'
```

```
[ ] last_transaction_index = 0
```

```
[ ] 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
```

+ Code + Text | Copy to Drive | Connect | Editing

```
[ ] # Miner 2 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
    # Miner 3 adds a block
    block = Block()

    for i in range(3):
       temp_transaction = transactions[last_transaction_index]
       #display_transaction (temp_transaction)
       # 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)
```

```
time: 2021-04-08 06:11:40.570658
-----
--------------
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100d776af73071682cb494e752
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100a0b9b51bb0c81cbbad86
-----
value: 7.0
-----
time: 2021-04-08 06:11:40.572173
-----
--------------
sender: 30819f300d06092a864886f70d010101050003818d003081890281810082e40f44f9cef42cac492d1
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100a0b9b51bb0c81cbbad86
-----
value: 3.0
-----
time: 2021-04-08 06:11:40.574036
-----
--------------
====================================
block # 3
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100a0b9b51bb0c81cbbad86384
-----
recipient: 30819f300d06092a864886f70d010101050003818d003081902818100 8f5e89df98216eb2c1e4
-----
value: 8.0
-----
time: 2021-04-08 06:11:40.575310
-----
--------------
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100a0b9b51bb0c81cbbad86384
-----
recipient: 30819f300d06092a864886f70d010101050003818d003081890281810082e40f44f9cef42cac49
-----
value: 1.0
-----
time: 2021-04-08 06:11:40.576645
-----
--------------
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100d776af73071682cb494e752
-----
recipient: 30819f300d06092a864886f70d010101050003818d003081902818100 8f5e89df98216eb2c1e4
-----
value: 5.0
-----
time: 2021-04-08 06:11:40.578007
-----
--------------
====================================
```

Practical 4

Implement and demonstrate the use of the following in Solidity :

(I).Varaible

(II).Operators

(III).Loops

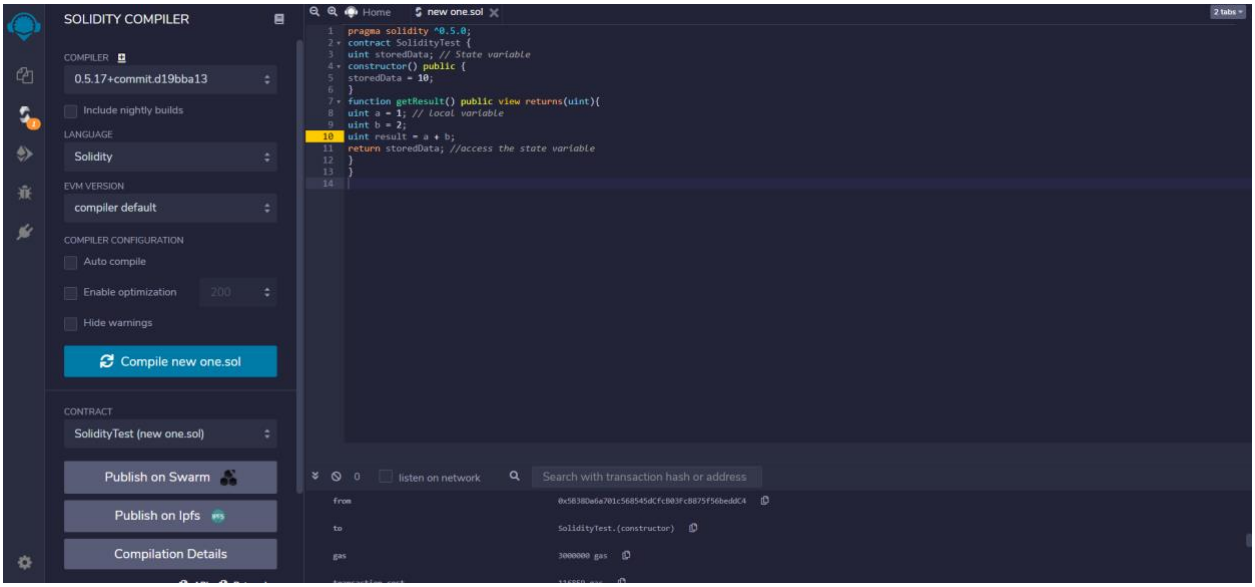
(IV).Decision Making

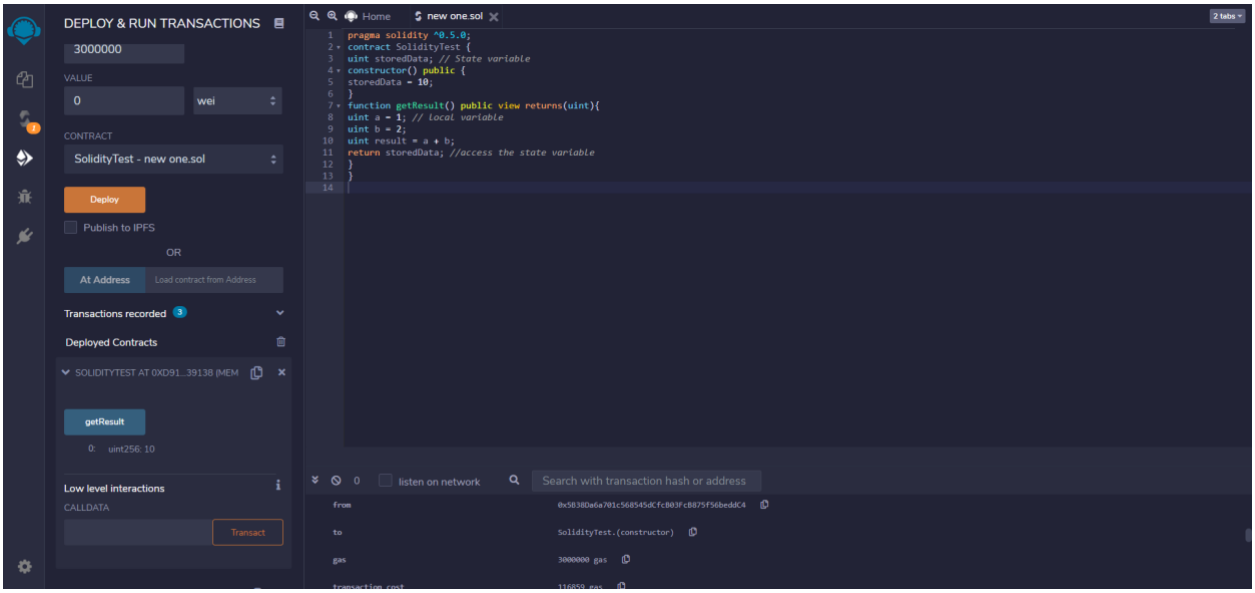
(V).Strings

(I).Variable

```
pragma solidity ^0.5.0;
contract SolidityTest {
uint storedData; // State variable
constructor() public {
storedData = 10;
}
function getResult() public view returns(uint){
uint a = 1; // local variable
uint b = 2;
uint result = a + b;
return storedData; //access the state variable
}
}
```

Output:


SOLIDITY COMPILER
COMPILER
0.5.17+commit.d19bba13
Include nightly builds
LANGUAGE
Solidity
EVM VERSION
compiler default
COMPILER CONFIGURATION
Auto compile
Enable optimization
200
Hide warnings
Compile new one.sol
CONTRACT
SolidityTest (new one.sol)
Publish on Swarm
Publish on Ipfs
Compilation Details
Home
new one.sol
2 tabs
pragma solidity ^0.5.0;
contract SolidityTest {
uint storedData; // State variable
constructor() public {
storedData = 10;
}
function getResult() public view returns(uint){
uint a = 1; // local variable
uint b = 2;
uint result = a + b;
return storedData; //access the state variable
}
}
listen on network
Search with transaction hash or address
from
0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to
SolidityTest.(constructor)
gas
3000000 gas


DEPLOY & RUN TRANSACTIONS
3000000
VALUE
0
wei
CONTRACT
SolidityTest - new one.sol
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded
3
Deployed Contracts
SOLIDITYTEST AT 0XD91...39138 (MEM
getResult
0: uint256: 10
Low level interactions
CALLDATA
Transact
Home
new one.sol
2 tabs
pragma solidity ^0.5.0;
contract SolidityTest {
uint storedData; // State variable
constructor() public {
storedData = 10;
}
function getResult() public view returns(uint){
uint a = 1; // local variable
uint b = 2;
uint result = a + b;
return storedData; //access the state variable
}
}
listen on network
Search with transaction hash or address
from
0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to
SolidityTest.(constructor)
gas
3000000 gas

// **Solidity program to demonstrate state variables**

pragma solidity ^0.5.0;

// Creating a contract

contract Solidity_var_Test {

// Declaring a state variable

uint8 public state_var;

// Defining a constructor

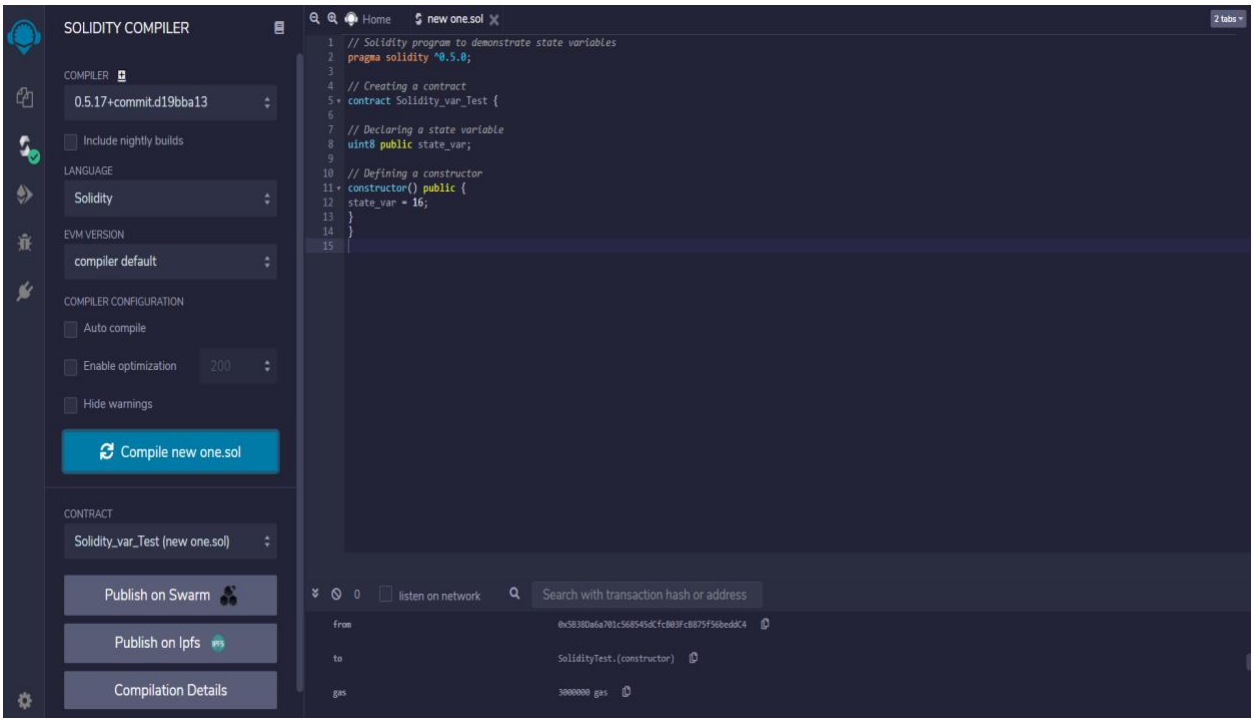
constructor() public {

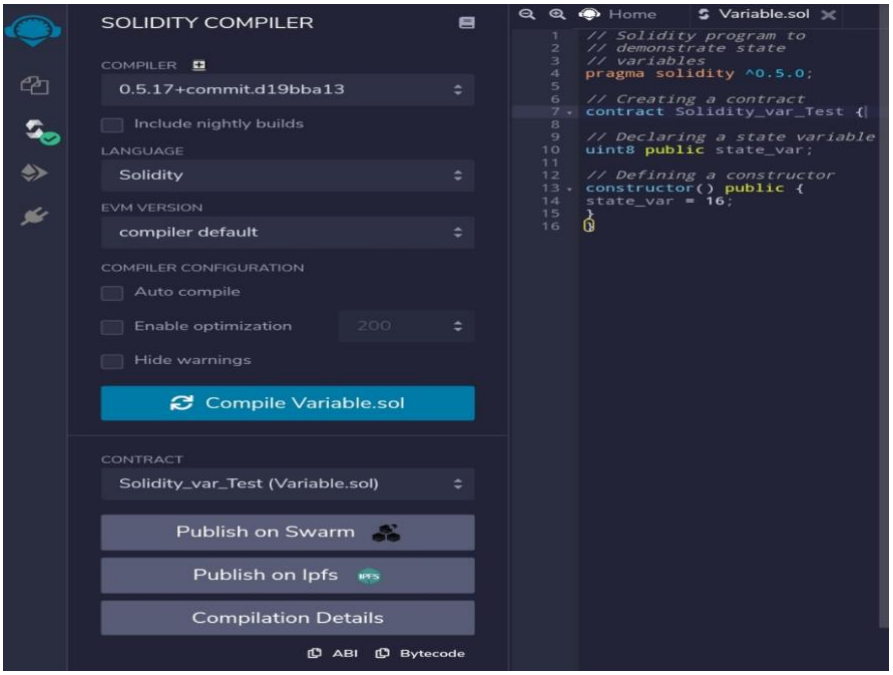
state_var = 16;

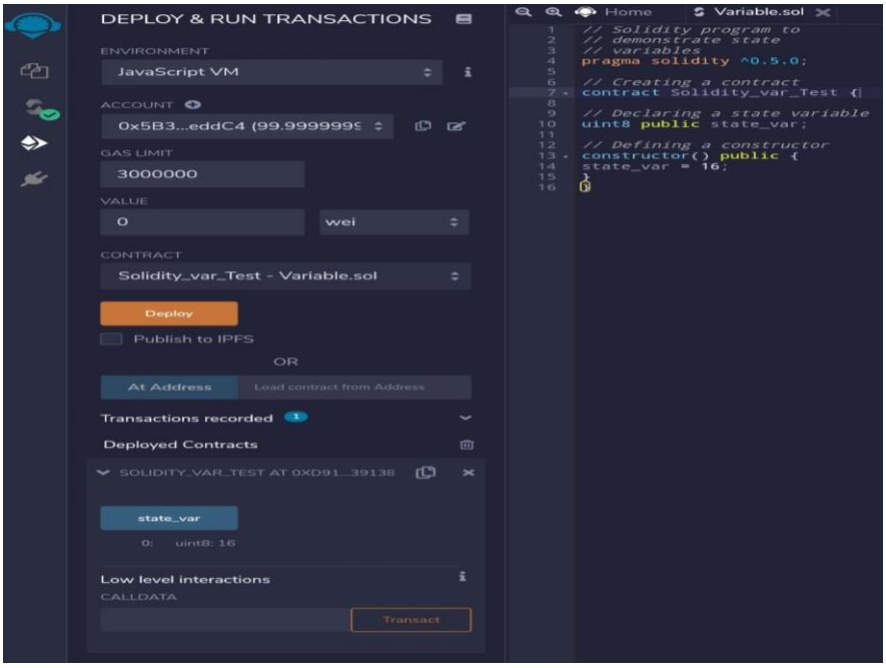
}

}

Output:




SOLIDITY COMPILER
COMPILER
0.5.17+commit.d19bba13
Include nightly builds
LANGUAGE
Solidity
EVM VERSION
compiler default
COMPILER CONFIGURATION
Auto compile
Enable optimization
200
Hide warnings
Compile Variable.sol
CONTRACT
Solidity_var_Test (Variable.sol)
Publish on Swarm
Publish on Ipfs
Compilation Details
ABI
Bytecode
Home
Variable.sol
// Solidity program to
// demonstrate state
// variables
pragma solidity ^0.5.0;
// Creating a contract
contract Solidity_var_Test {
// Declaring a state variable
uint8 public state_var;
// Defining a constructor
constructor() public {
state_var = 16;
}


DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
JavaScript VM
ACCOUNT
0x5B3...eddC4 (99.999999
GAS LIMIT
3000000
VALUE
0
wei
CONTRACT
Solidity_var_Test - Variable.sol
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded
Deployed Contracts
SOLIDITY_VAR_TEST AT 0XD91...39138
state_var
0: uint8: 16
Low level interactions
CALLDATA
Transact
Home
Variable.sol
// Solidity program to
// demonstrate state
// variables
pragma solidity ^0.5.0;
// Creating a contract
contract Solidity_var_Test {
// Declaring a state variable
uint8 public state_var;
// Defining a constructor
constructor() public {
state_var = 16;
}

// **Solidity program to show Global variables**

pragma solidity ^0.5.0;

// Creating a contract

contract Test {

// Defining a variable

address public admin;

// Creating a constructor to

// use Global variable

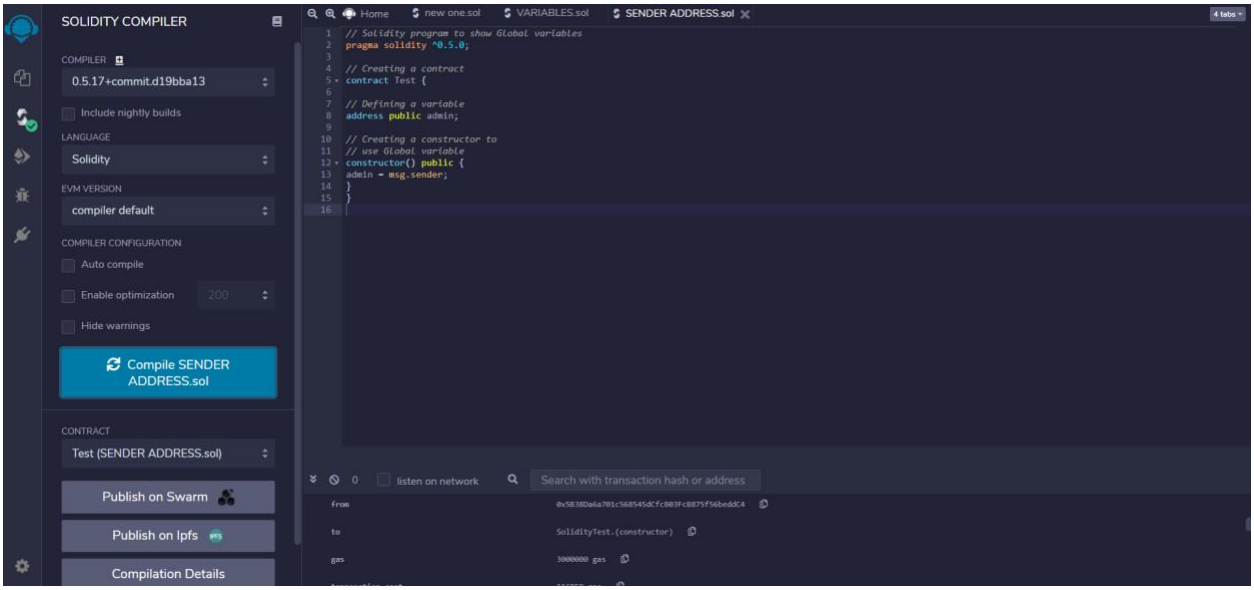
constructor() public {

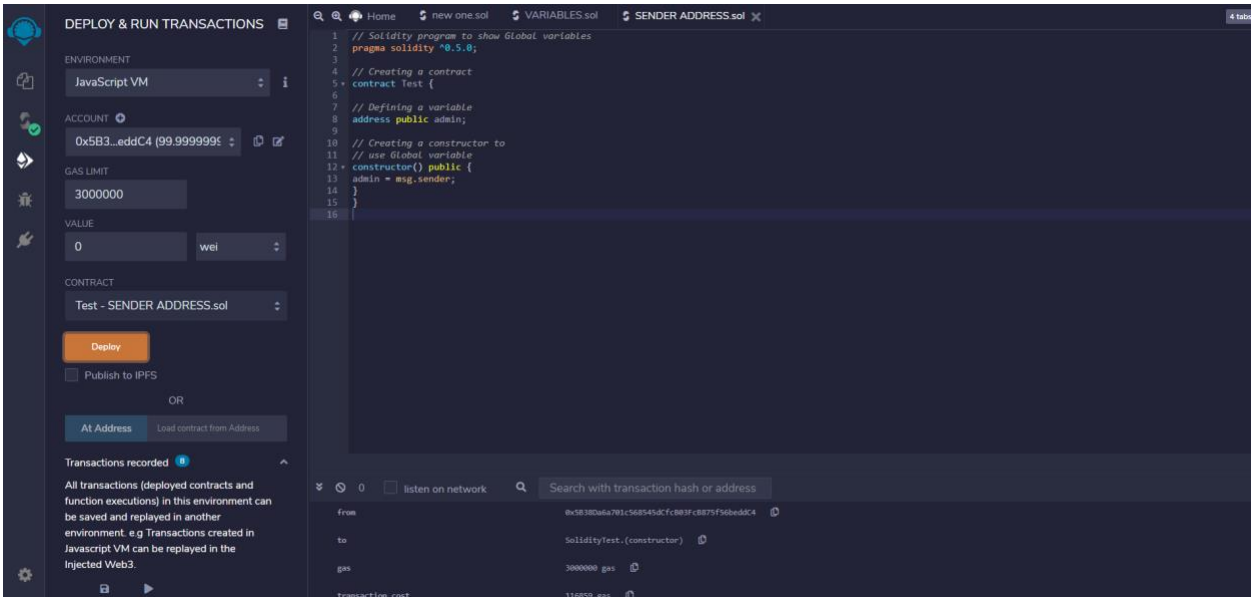
admin = msg.sender;

}

}

Output:



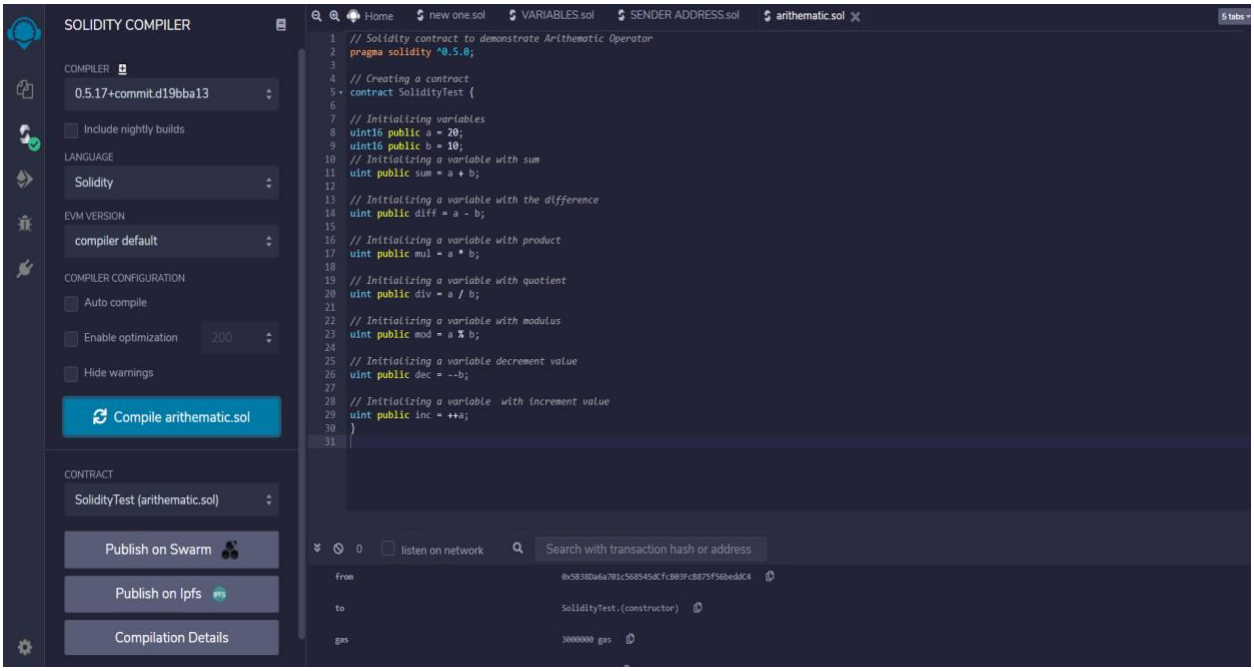


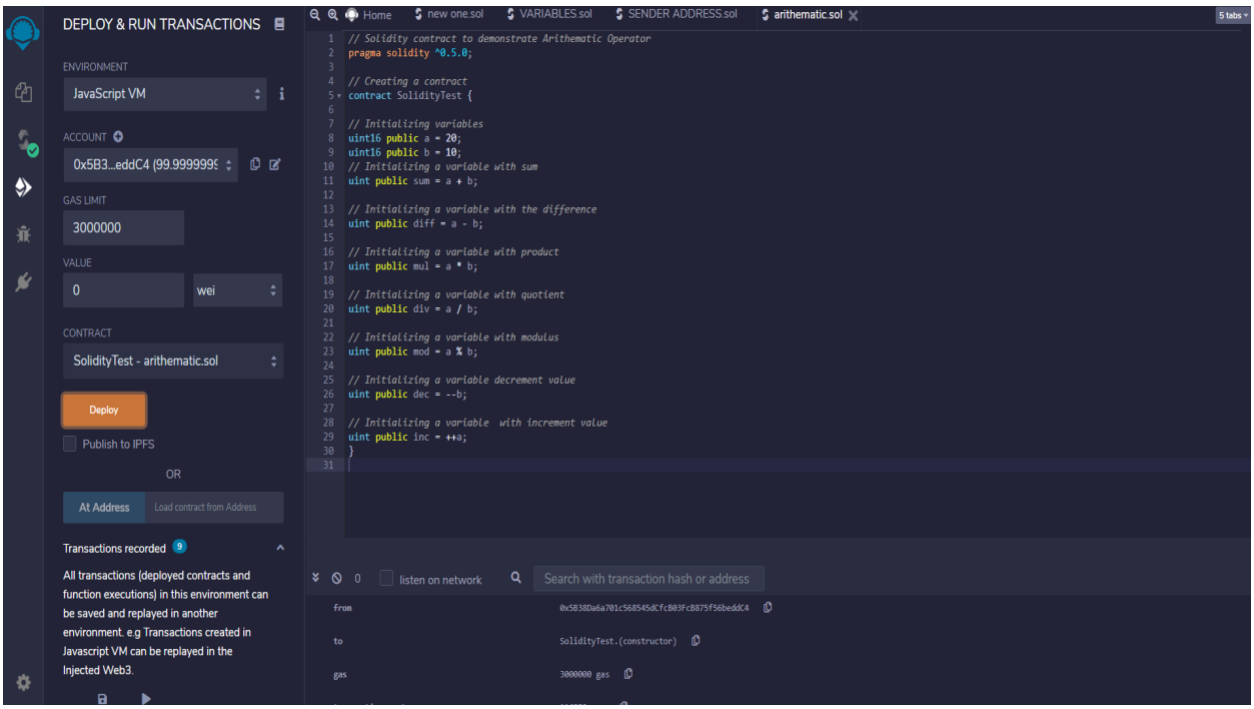
(II).Operators

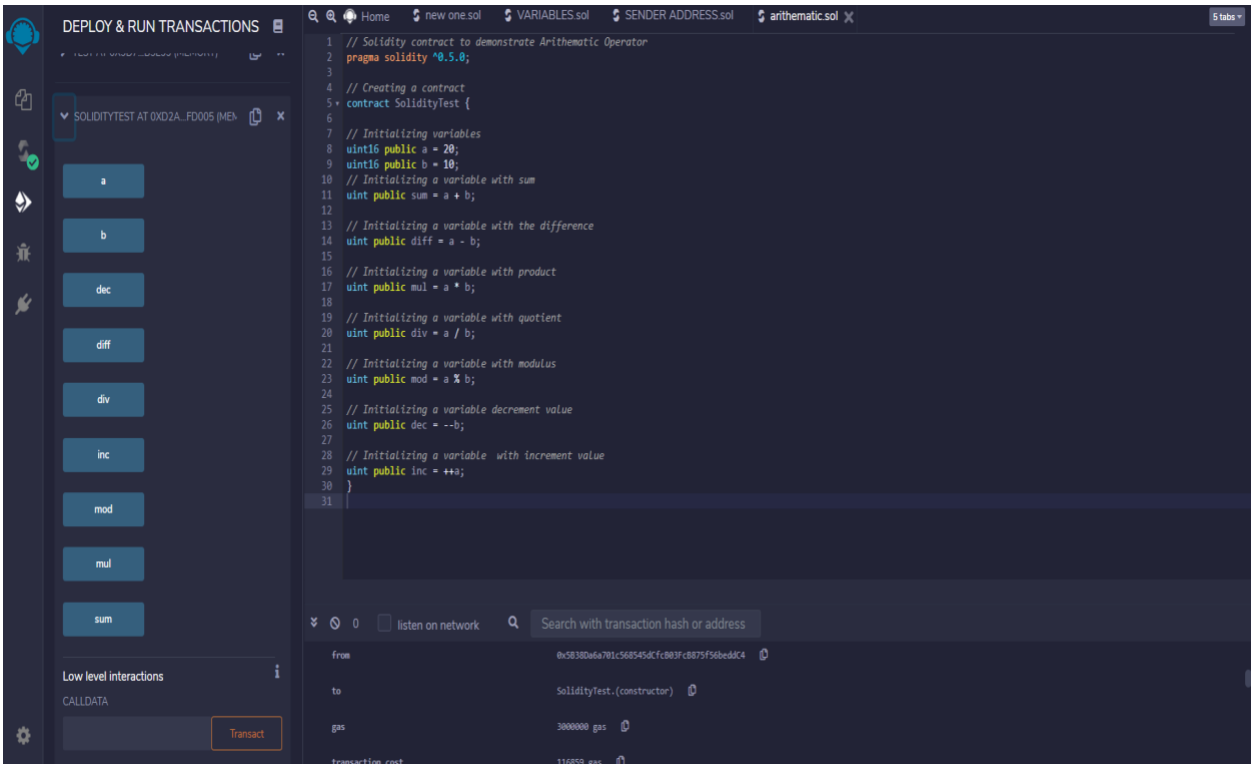
```
// Solidity contract to demonstrate Arithematic Operator
pragma solidity ^0.5.0;

// Creating a contract
contract SolidityTest {

// Initializing variables
uint16 public a = 20;
uint16 public b = 10;
// Initializing a variable with sum
uint public sum = a + b;

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

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

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

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

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

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

Output:



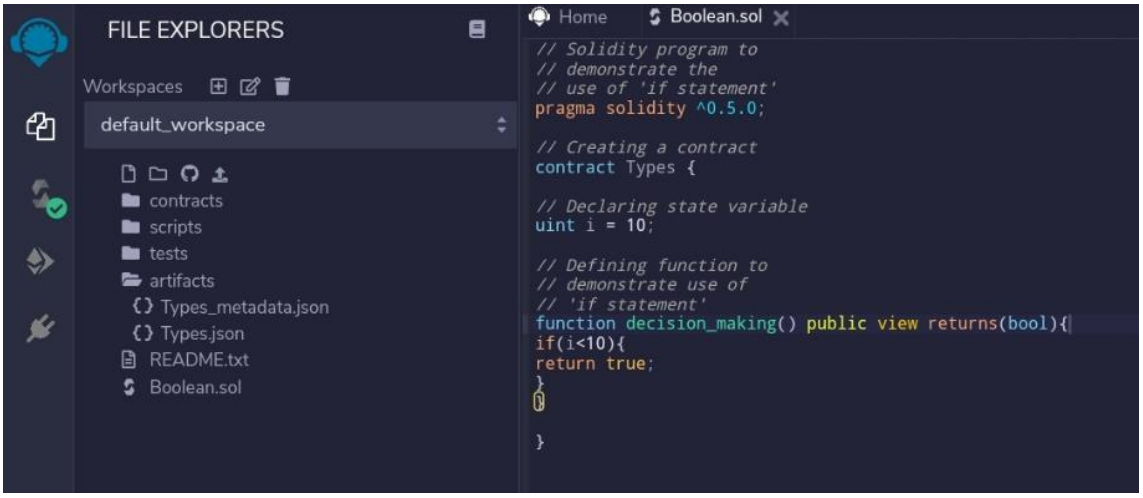

DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
JavaScript VM
ACCOUNT
0x5B3...eddC4 (99.999999
GAS LIMIT
3000000
VALUE
0
wei
CONTRACT
SolidityTest - arithematic.sol
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 9
All transactions (deployed contracts and function executions) in this environment can be saved and replayed in another environment. e.g Transactions created in Javascript VM can be replayed in the Injected Web3.
Home
new one.sol
VARIABLES.sol
SENDER ADDRESS.sol
arithematic.sol
5 tabs
// Solidity contract to demonstrate Arithematic Operator
pragma solidity ^0.5.0;
// Creating a contract
contract SolidityTest {
// Initializing variables
uint16 public a = 20;
uint16 public b = 10;
// Initializing a variable with sum
uint public sum = a + b;
// Initializing a variable with the difference
uint public diff = a - b;
// Initializing a variable with product
uint public mul = a * b;
// Initializing a variable with quotient
uint public div = a / b;
// Initializing a variable with modulus
uint public mod = a % b;
// Initializing a variable decrement value
uint public dec = --b;
// Initializing a variable with increment value
uint public inc = ++a;
}
0
listen on network
Search with transaction hash or address
from
0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to
SolidityTest.(constructor)
gas
3000000 gas

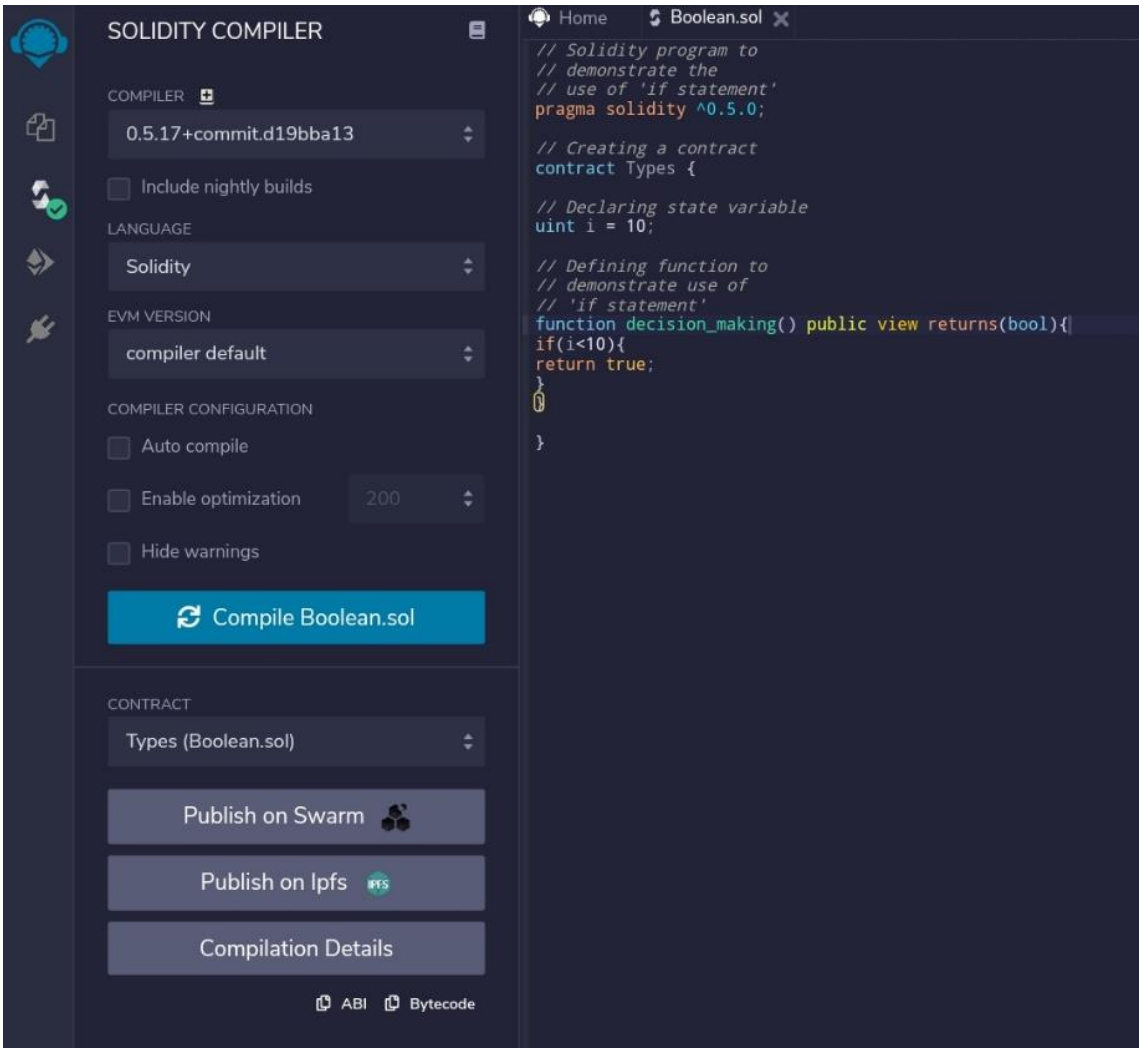

DEPLOY & RUN TRANSACTIONS
SOLIDITYTEST AT 0XD2A...FD005 (MEM
a
b
dec
diff
div
inc
mod
mul
sum
Low level interactions
CALLDATA
Transact
Home
new one.sol
VARIABLES.sol
SENDER ADDRESS.sol
arithematic.sol
5 tabs
// Solidity contract to demonstrate Arithematic Operator
pragma solidity ^0.5.0;
// Creating a contract
contract SolidityTest {
// Initializing variables
uint16 public a = 20;
uint16 public b = 10;
// Initializing a variable with sum
uint public sum = a + b;
// Initializing a variable with the difference
uint public diff = a - b;
// Initializing a variable with product
uint public mul = a * b;
// Initializing a variable with quotient
uint public div = a / b;
// Initializing a variable with modulus
uint public mod = a % b;
// Initializing a variable decrement value
uint public dec = --b;
// Initializing a variable with increment value
uint public inc = ++a;
}
0
listen on network
Search with transaction hash or address
from
0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to
SolidityTest.(constructor)
gas
3000000 gas

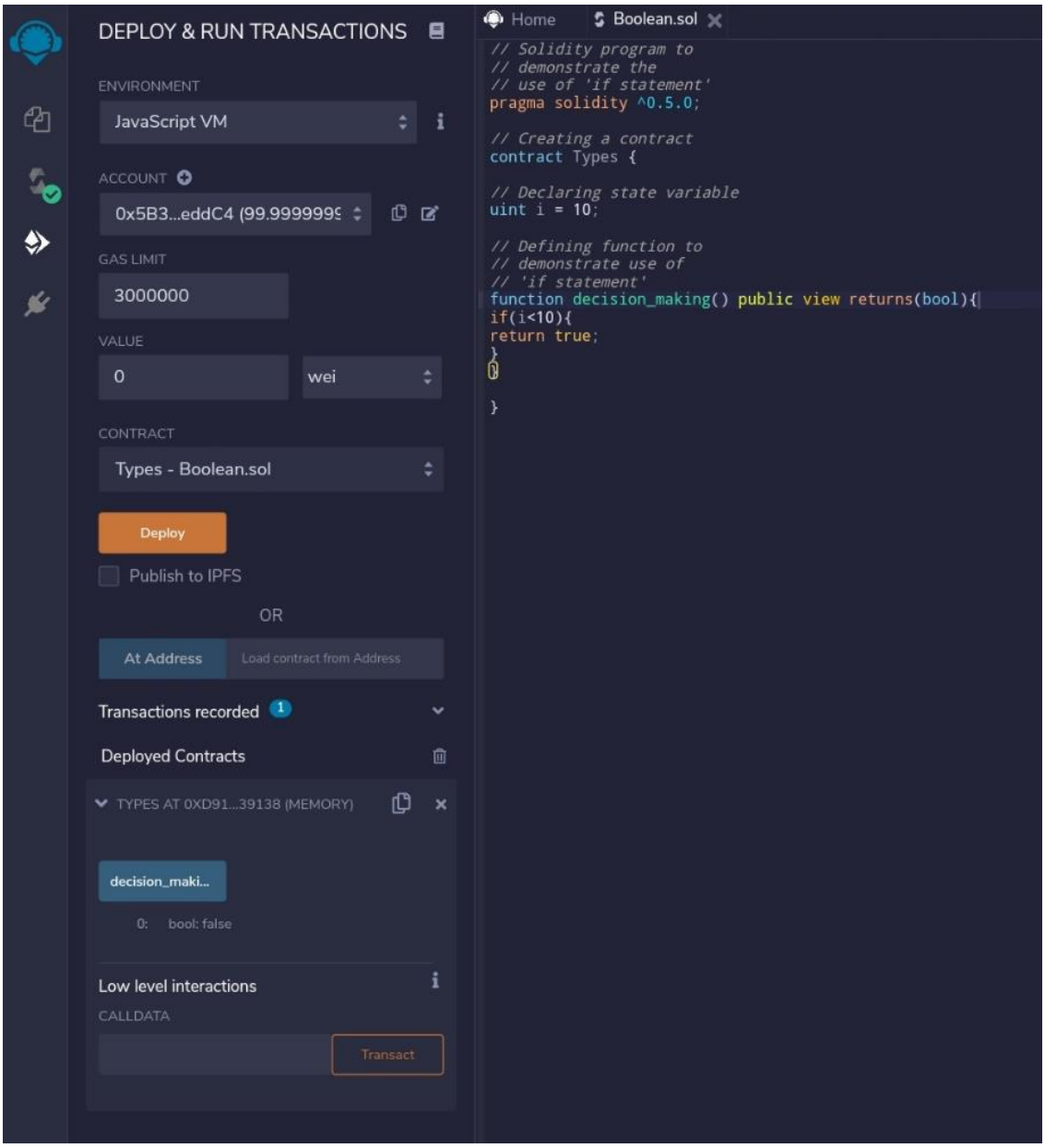
(IV).Decision Making

```
// Solidity program to demonstrate the use of 'if statement'
pragma solidity ^0.5.0;

// Creating a contract
contract Types {
// Declaring state variable
uint i = 10;

// Defining function to demonstrate use of 'if statement'
function decision_making() public view returns(bool){
if(i<10){
return true;
}
}
}
```

Output:




SOLIDITY COMPILER
COMPILER
0.5.17+commit.d19bba13
Include nightly builds
LANGUAGE
Solidity
EVM VERSION
compiler default
COMPILER CONFIGURATION
Auto compile
Enable optimization
200
Hide warnings
Compile Boolean.sol
CONTRACT
Types (Boolean.sol)
Publish on Swarm
Publish on Ipfs
Compilation Details
ABI
Bytecode
Home
Boolean.sol
// Solidity program to
// demonstrate the
// use of 'if statement'
pragma solidity ^0.5.0;
// Creating a contract
contract Types {
// Declaring state variable
uint i = 10;
// Defining function to
// demonstrate use of
// 'if statement'
function decision_making() public view returns(bool){
if(i<10){
return true;
}
}



// **Solidity program to demonstrate the use of 'if...else' statement**

pragma solidity ^0.5.0;

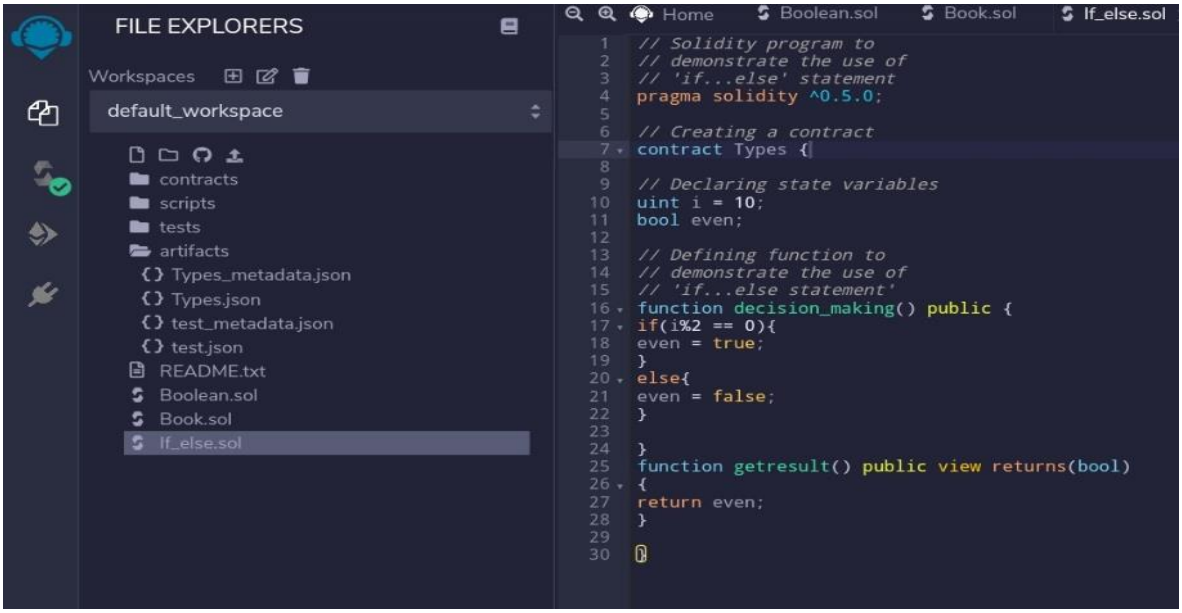
// Creating a contract

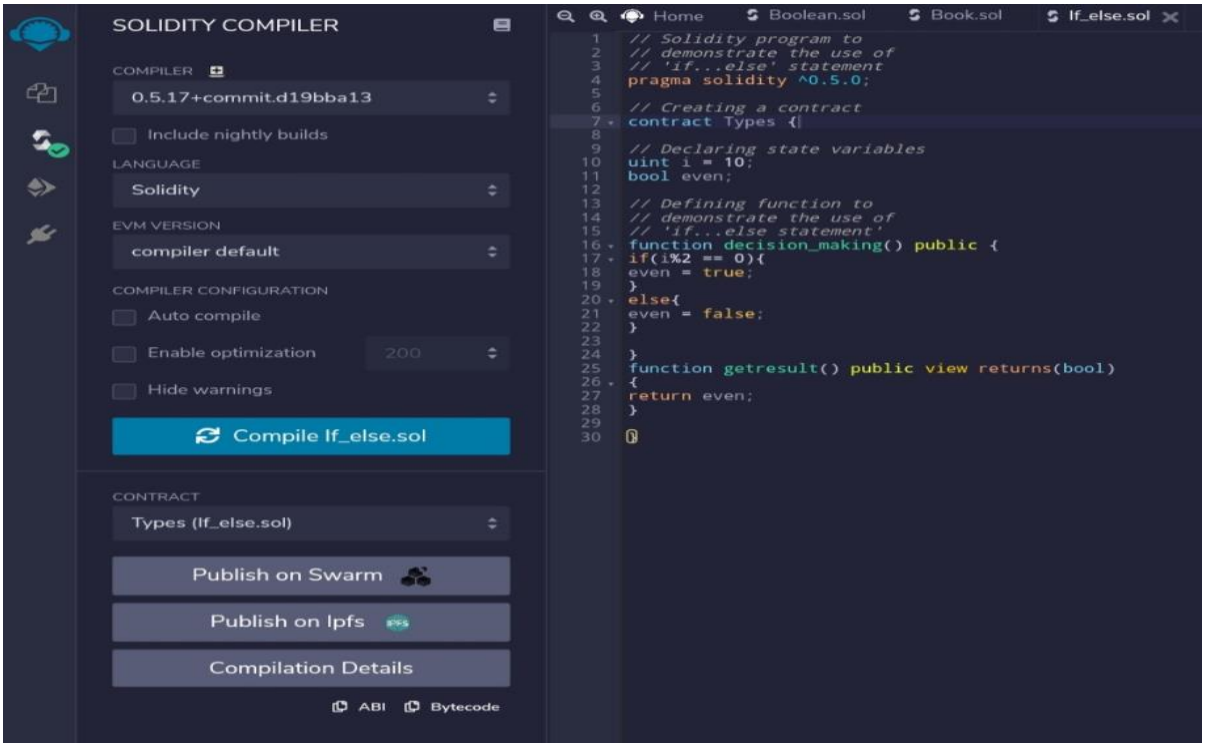
contract Types {

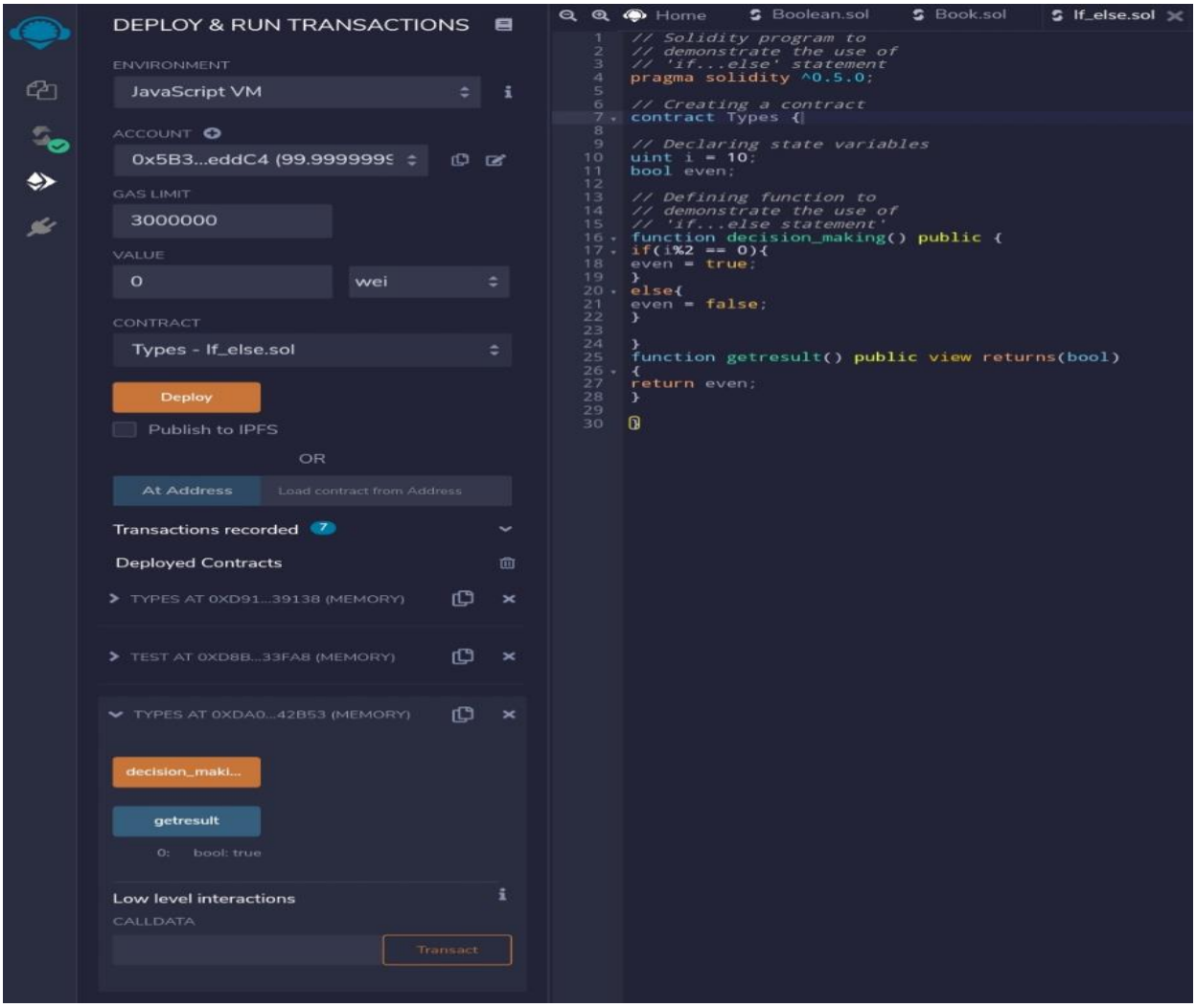
// Declaring state variables

```
uint i = 10;
bool even;

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

}
```

Output:


FILE EXPLORERS
Workspaces
default_workspace
contracts
scripts
tests
artifacts
Types_metadata.json
Types.json
test_metadata.json
test.json
README.txt
Boolean.sol
Book.sol
If_else.sol
Home
Boolean.sol
Book.sol
If_else.sol
// Solidity program to
// demonstrate the use of
// 'if...else' statement
pragma solidity ^0.5.0;
// Creating a contract
contract Types {
// Declaring state variables
uint i = 10;
bool even;
// Defining function to
// demonstrate the use of
// 'if...else statement'
function decision_making() public {
if(i%2 == 0){
even = true;
}
else{
even = false;
}
}
function getresult() public view returns(bool)
{
return even;
}

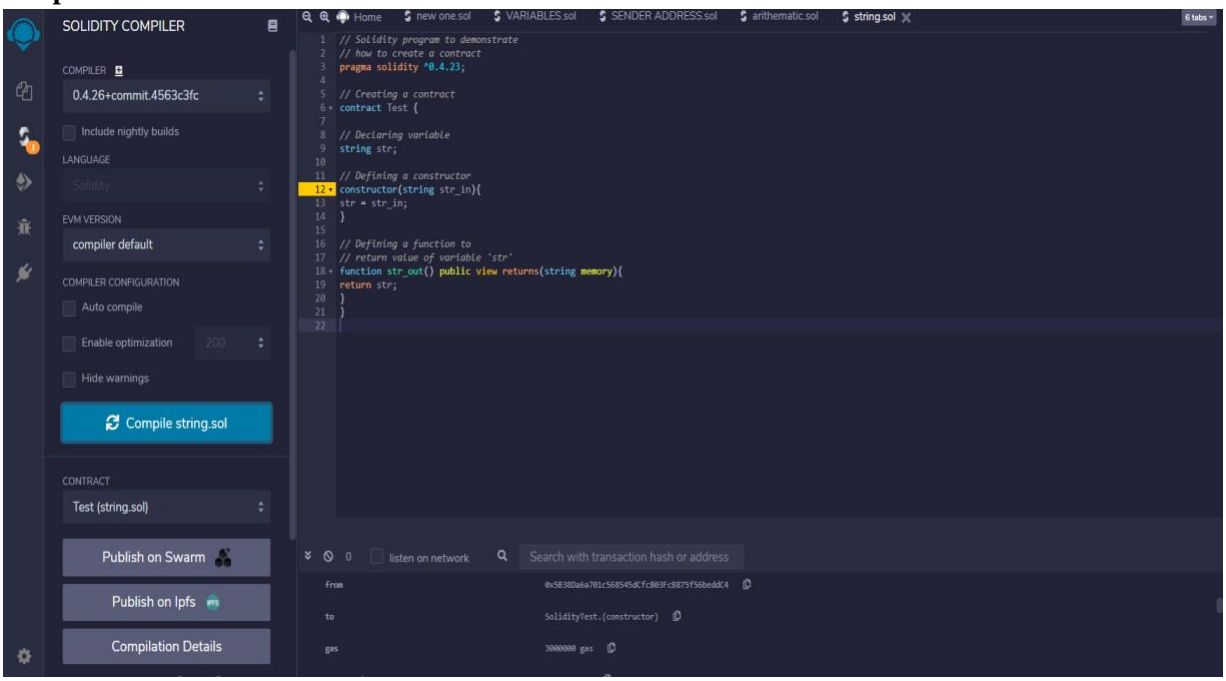

SOLIDITY COMPILER
COMPILER
0.5.17+commit.d19bba13
Include nightly builds
LANGUAGE
Solidity
EVM VERSION
compiler default
COMPILER CONFIGURATION
Auto compile
Enable optimization
200
Hide warnings
Compile If_else.sol
CONTRACT
Types (If_else.sol)
Publish on Swarm
Publish on Ipfs
Compilation Details
ABI
Bytecode
Home
Boolean.sol
Book.sol
If_else.sol
// Solidity program to
// demonstrate the use of
// 'if...else' statement
pragma solidity ^0.5.0;
// Creating a contract
contract Types {
// Declaring state variables
uint i = 10;
bool even;
// Defining function to
// demonstrate the use of
// 'if...else statement'
function decision_making() public {
if(i%2 == 0){
even = true;
}
else{
even = false;
}
}
function getresult() public view returns(bool)
{
return even;
}

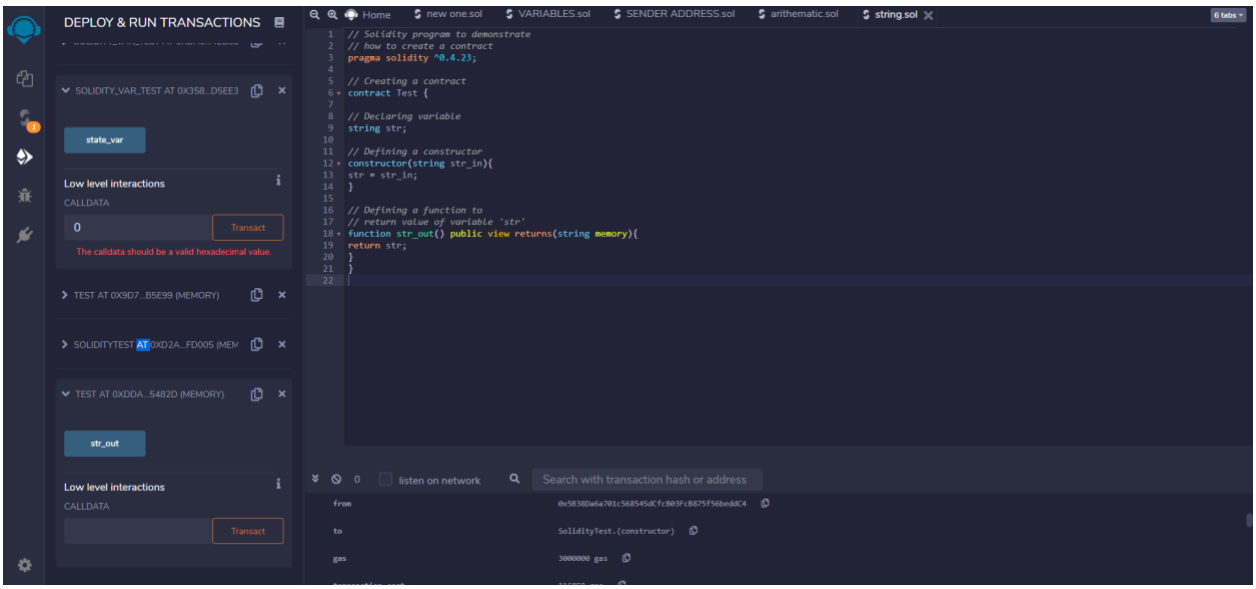

DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
JavaScript VM
ACCOUNT
0x5B3...eddC4 (99.999999
GAS LIMIT
3000000
VALUE
0
wei
CONTRACT
Types - If_else.sol
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 7
Deployed Contracts
TYPES AT 0XD91...39138 (MEMORY)
TEST AT 0XD8B...33FA8 (MEMORY)
TYPES AT 0XDA0...42B53 (MEMORY)
decision_maki...
getresult
0: bool: true
Low level interactions
CALLDATA
Transact
Home
Boolean.sol
Book.sol
If_else.sol
// Solidity program to
// demonstrate the use of
// 'if...else' statement
pragma solidity ^0.5.0;
// Creating a contract
contract Types {
// Declaring state variables
uint i = 10;
bool even;
// Defining function to
// demonstrate the use of
// 'if...else statement'
function decision_making() public {
if(i%2 == 0){
even = true;
}
else{
even = false;
}
}
function getresult() public view returns(bool)
{
return even;
}

(III) Strings

```
// Solidity program to demonstrate
// how to create a contract
pragma solidity ^0.4.23;

// Creating a contract
contract Test {

// Declaring variable
string str;

// Defining a constructor
constructor(string str_in){
str = str_in;
}

// Defining a function to
// return value of variable 'str'
function str_out() public view returns(string memory){
return str;
}
}
```

Output


SOLIDITY COMPILER
COMPILER
0.4.26+commit.4563c3fc
Include nightly builds
LANGUAGE
Solidity
EVM VERSION
compiler default
COMPILER CONFIGURATION
Auto compile
Enable optimization
200
Hide warnings
Compile string.sol
CONTRACT
Test (string.sol)
Publish on Swarm
Publish on Ipfs
Compilation Details
Home
new one.sol
VARIABLES.sol
SENDER ADDRESS.sol
arithematic.sol
string.sol
6 tabs
// Solidity program to demonstrate
// how to create a contract
pragma solidity ^0.4.23;
// Creating a contract
contract Test {
// Declaring variable
string str;
// Defining a constructor
constructor(string str_in){
str = str_in;
}
// Defining a function to
// return value of variable 'str'
function str_out() public view returns(string memory){
return str;
}
}
listen on network
Search with transaction hash or address
from
0x5838Da6a701c568545dCfcB03FcB875f56beddC4
to
SolidityTest.(constructor)
gas
3000000 gas


DEPLOY & RUN TRANSACTIONS
SOLIDITY_VAR_TEST AT 0X358...D5EE3
state_var
Low level interactions
CALLDATA
0
Transact
The calldata should be a valid hexadecimal value.
TEST AT 0X9D7...B5E99 (MEMORY)
SOLIDITYTEST AT 0XD2A...FD005 (MEM
TEST AT 0XDDA...5482D (MEMORY)
str_out
Low level interactions
CALLDATA
Transact
Home
new one.sol
VARIABLES.sol
SENDER ADDRESS.sol
arithematic.sol
string.sol
6 tabs
// Solidity program to demonstrate
// how to create a contract
pragma solidity ^0.4.23;
// Creating a contract
contract Test {
// Declaring variable
string str;
// Defining a constructor
constructor(string str_in){
str = str_in;
}
// Defining a function to
// return value of variable 'str'
function str_out() public view returns(string memory){
return str;
}
}
listen on network
Search with transaction hash or address
from
0x5838Da6a701c568545dCfcB03FcB875f56beddC4
to
SolidityTest.(constructor)
gas
3000000 gas

Practical 5

Implement and demonstrate the use of the following in Solidity :

(I).Arrays

(II).Enums

(III).Structs

(IV).Mappings

(V).Coversations

(VI).Ether Units

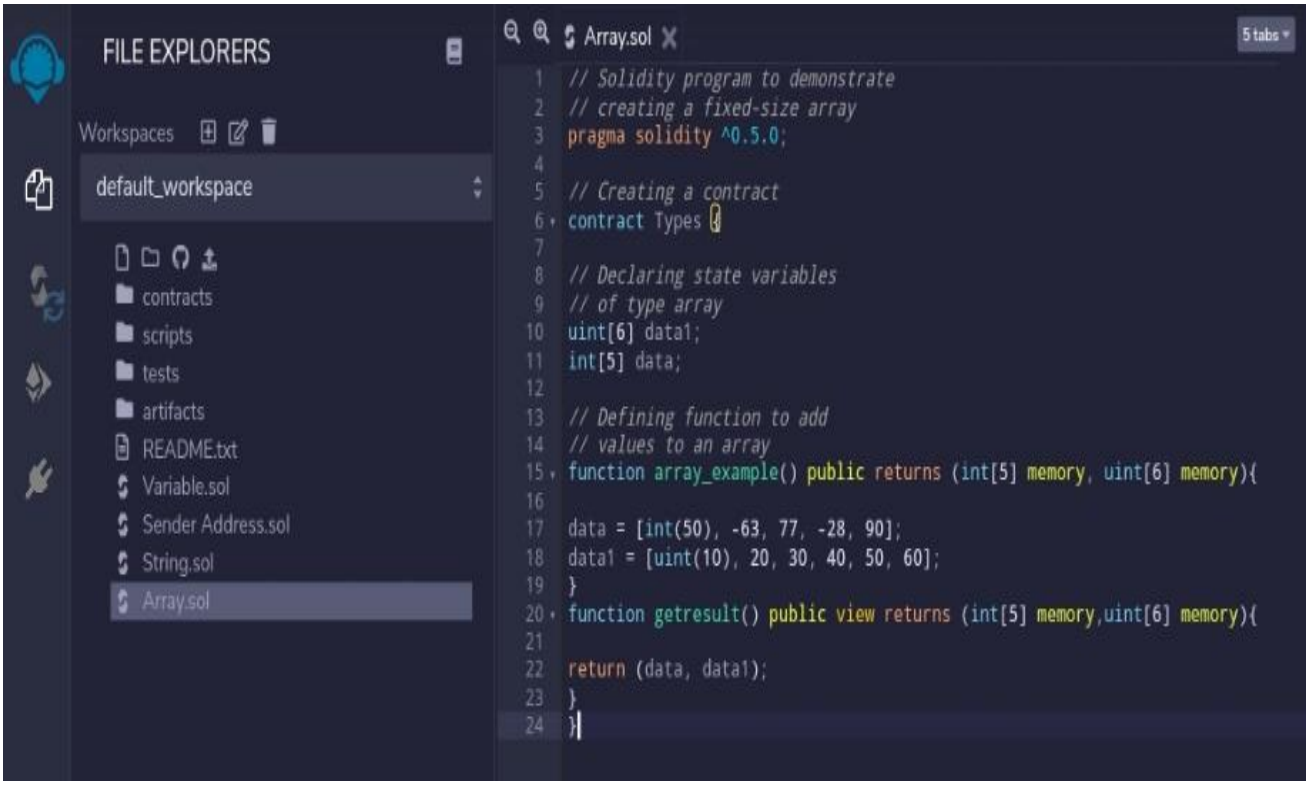
(VII).Special Varaibles

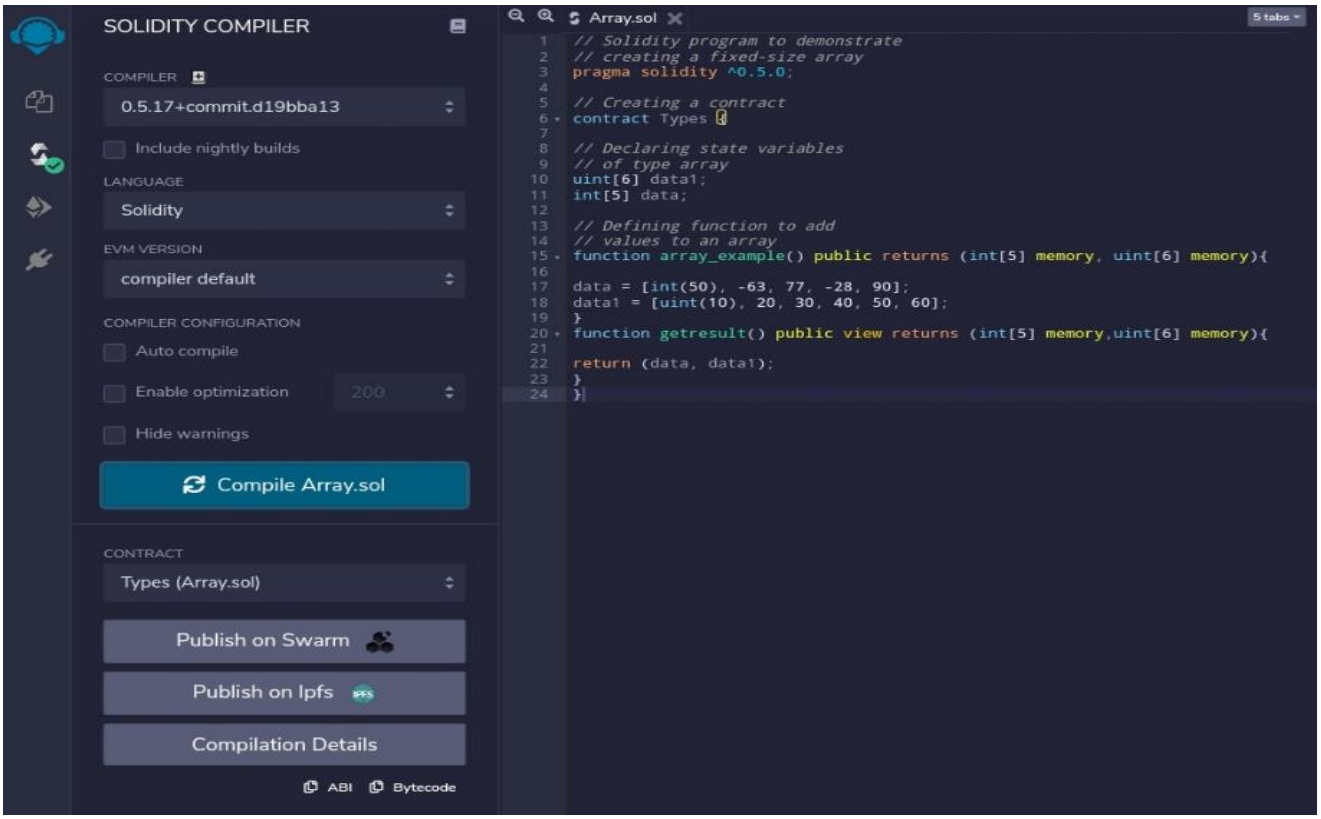
(I).Arrays

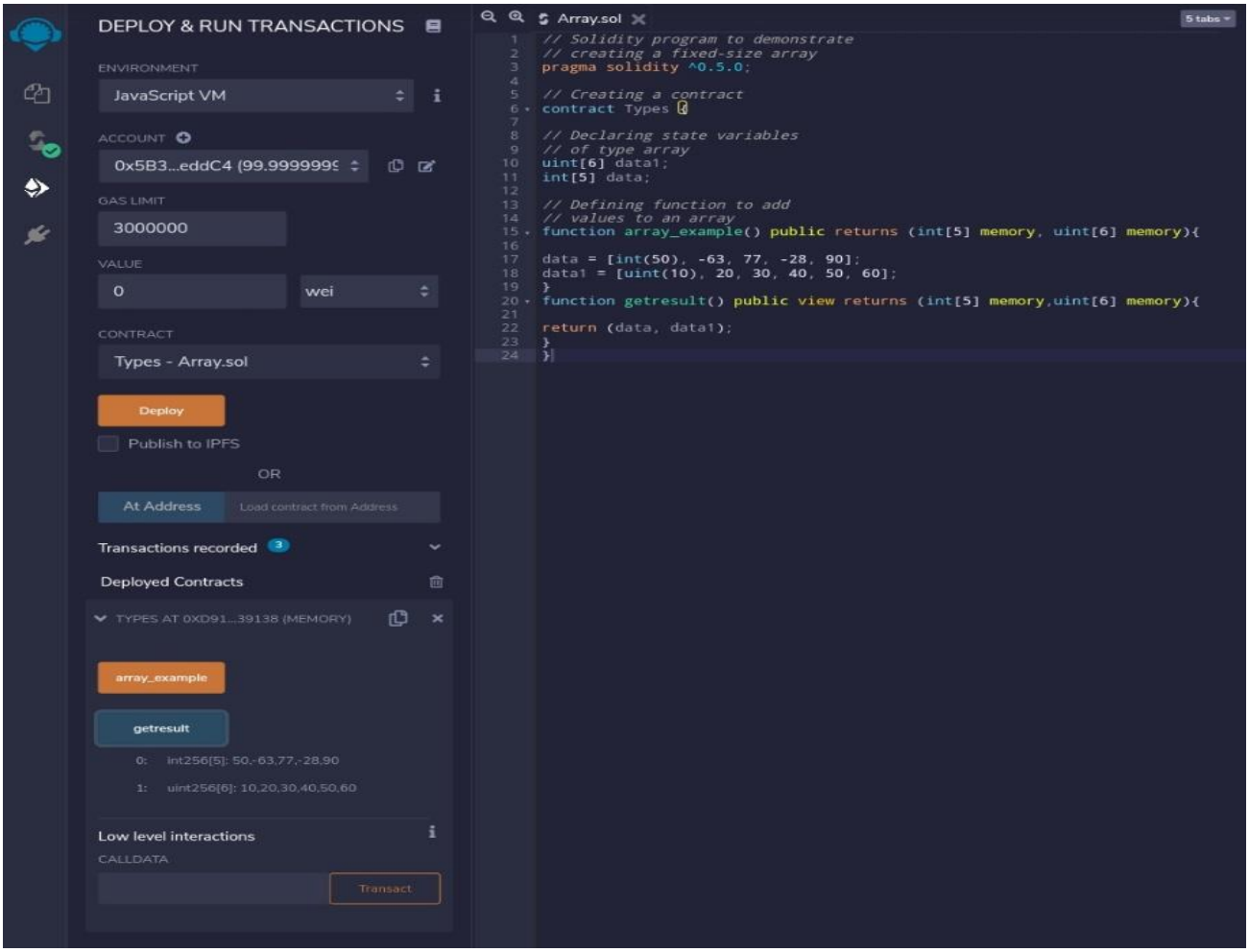
```
// Solidity program to demonstrate
// creating a fixed-size array
pragma solidity ^0.5.0;

// Creating a contract
contract Types {

// Declaring state variables
// of type array
uint[6] data1;
int[5] data;

// Defining function to add
// values to an array
function array_example() public returns (int[5] memory, uint[6] memory){
data = [int(50), -63, 77, -28, 90];
```

```
data1 = [uint(10), 20, 30, 40, 50, 60];
}
function getresult() public view returns (int[5] memory,uint[6] memory){
return (data, data1);
}
}
```

Output:




SOLIDITY COMPILER
COMPILER
0.5.17+commit.d19bba13
Include nightly builds
LANGUAGE
Solidity
EVM VERSION
compiler default
COMPILER CONFIGURATION
Auto compile
Enable optimization
200
Hide warnings
Compile Array.sol
CONTRACT
Types (Array.sol)
Publish on Swarm
Publish on Ipfs
Compilation Details
ABI
Bytecode
Array.sol
5 tabs
// Solidity program to demonstrate
// creating a fixed-size array
pragma solidity ^0.5.0;
// Creating a contract
contract Types {
// Declaring state variables
// of type array
uint[6] data1;
int[5] data;
// Defining function to add
// values to an array
function array_example() public returns (int[5] memory, uint[6] memory){
data = [int(50), -63, 77, -28, 90];
data1 = [uint(10), 20, 30, 40, 50, 60];
}
function getresult() public view returns (int[5] memory,uint[6] memory){
return (data, data1);
}
}

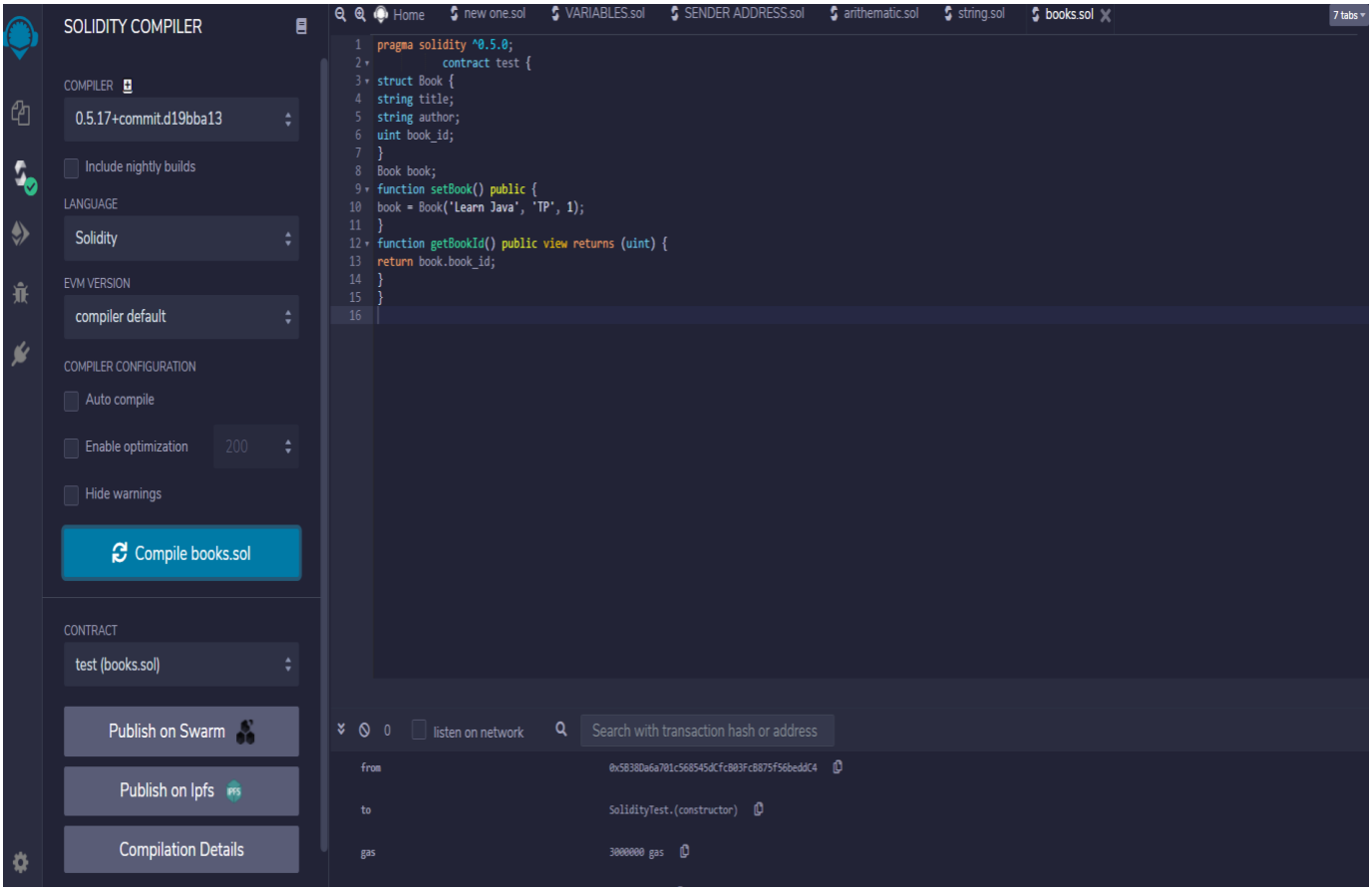


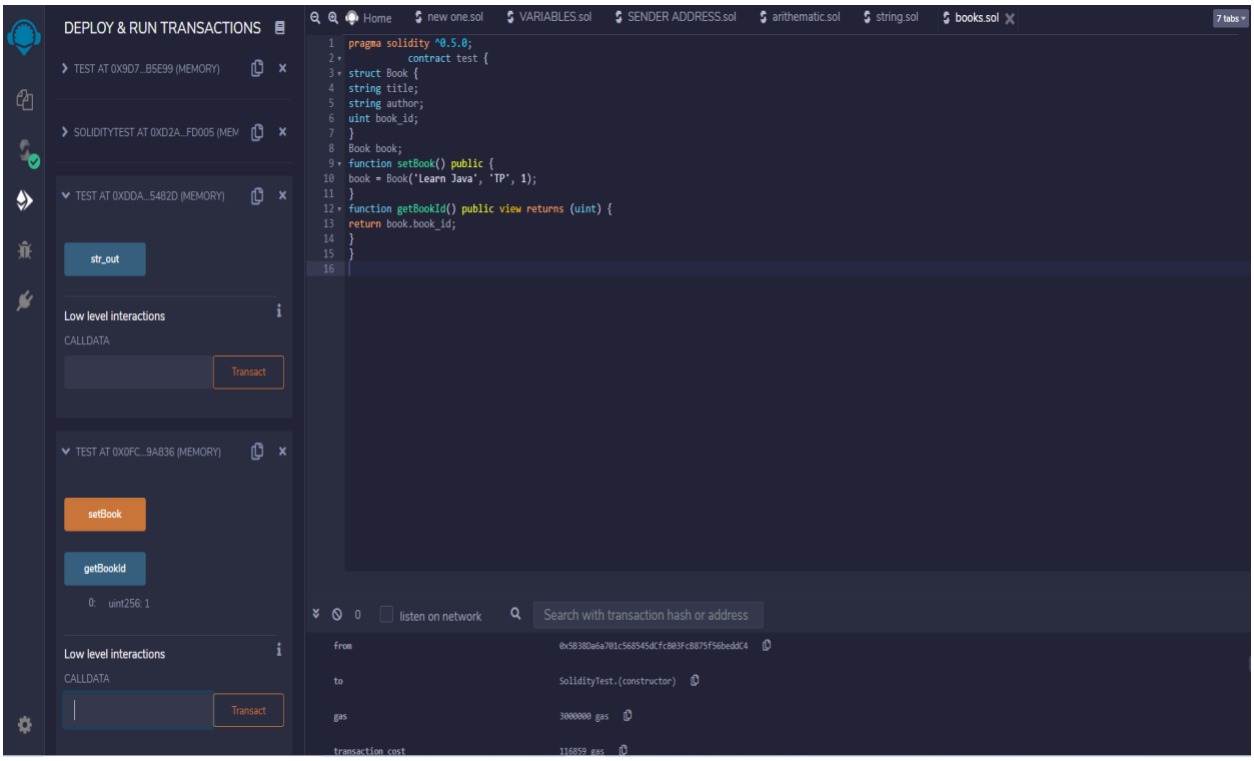
(III).Structs

```
pragma solidity ^0.5.0;
 contract test {
struct Book {
string title;
string author;
uint book_id;
}
Book book;
```

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

Output:

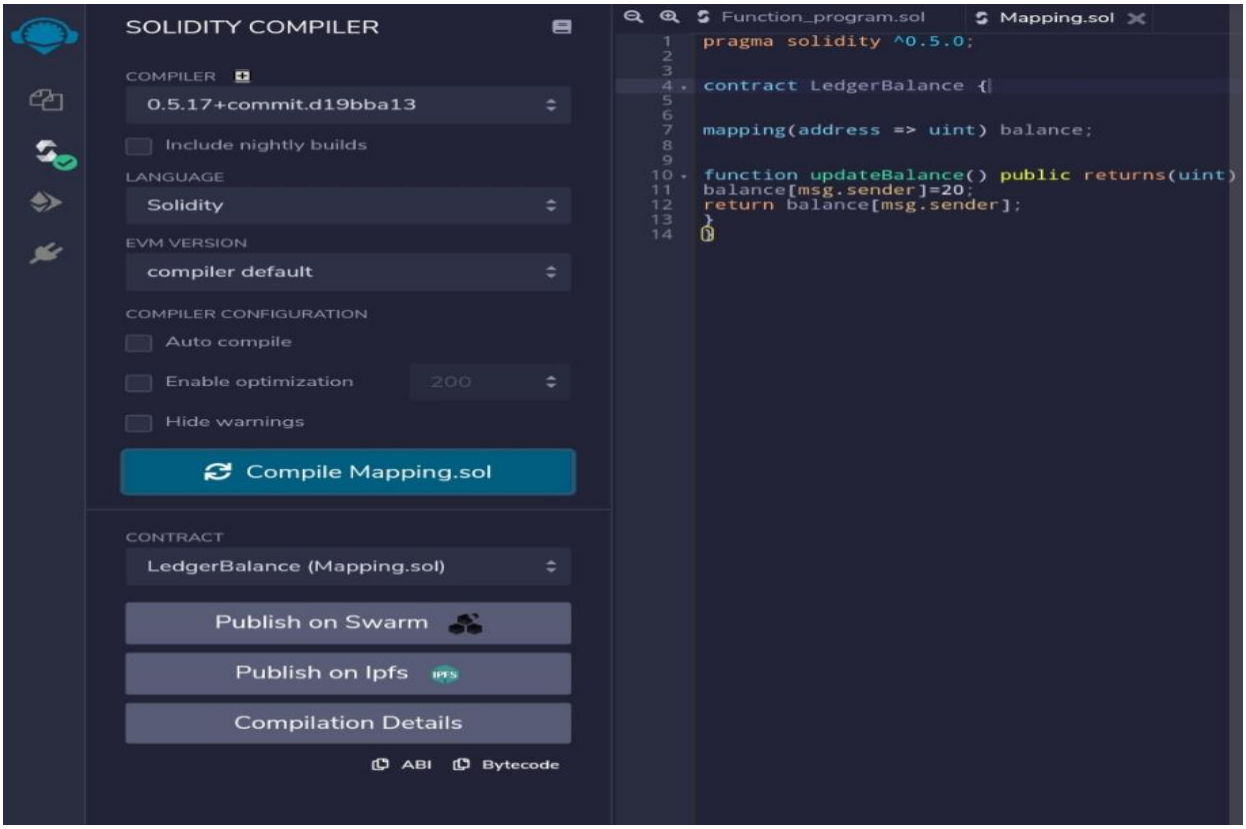


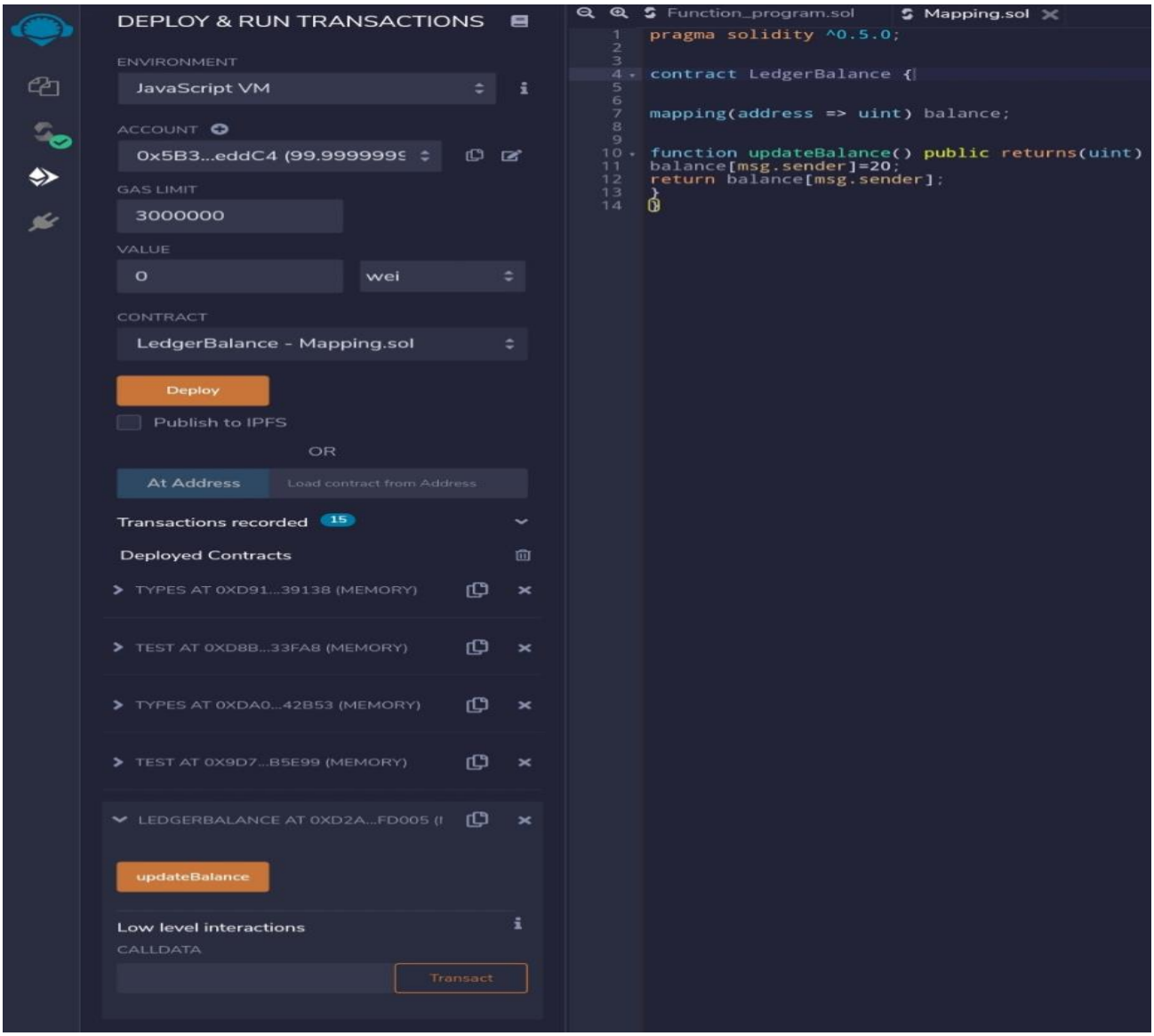


(IV).Mappings

```
pragma solidity ^0.5.0;
contract LedgerBalance {
mapping(address => uint) balance;
function updateBalance() public returns(uint) {
balance[msg.sender]=20;
return balance[msg.sender];
}
}
```

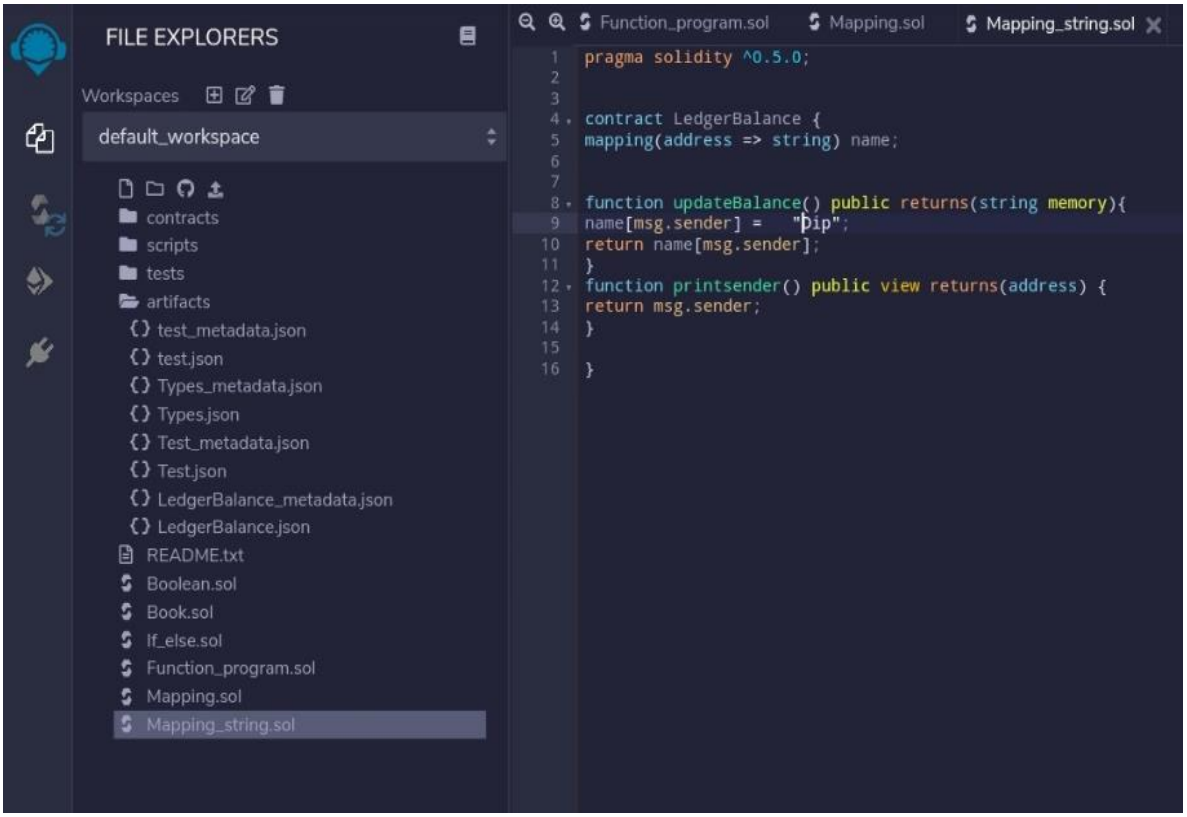
Output:

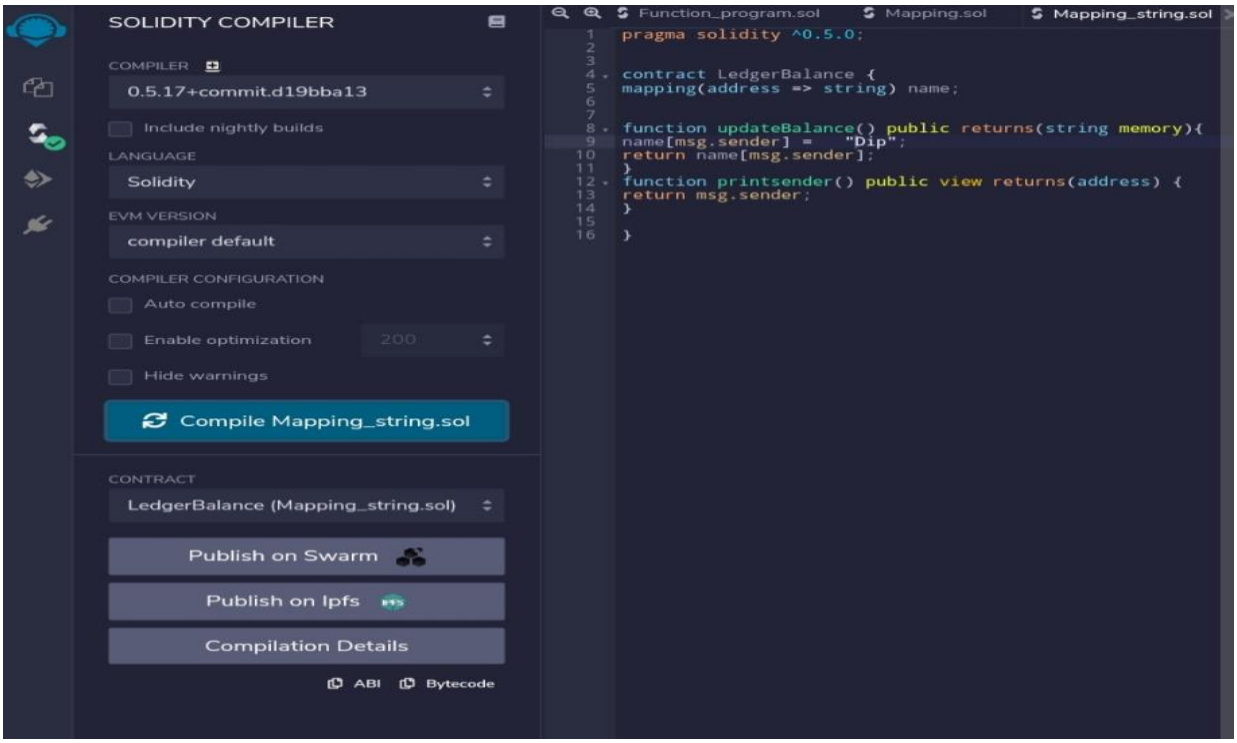

SOLIDITY COMPILER
COMPILER
0.5.17+commit.d19bba13
Include nightly builds
LANGUAGE
Solidity
EVM VERSION
compiler default
COMPILER CONFIGURATION
Auto compile
Enable optimization
200
Hide warnings
Compile Mapping.sol
CONTRACT
LedgerBalance (Mapping.sol)
Publish on Swarm
Publish on Ipfs
Compilation Details
ABI
Bytecode
Function_program.sol
Mapping.sol
pragma solidity ^0.5.0;
contract LedgerBalance {
mapping(address => uint) balance;
function updateBalance() public returns(uint)
balance[msg.sender]=20;
return balance[msg.sender];
}

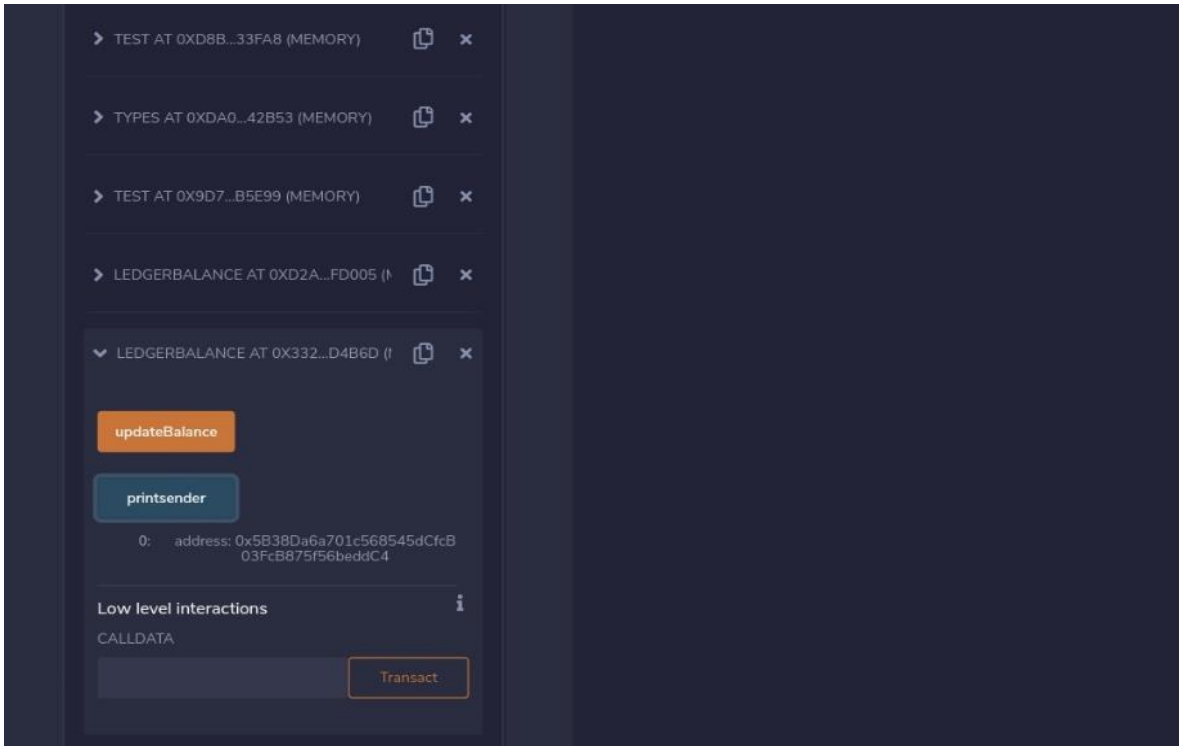

DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
JavaScript VM
ACCOUNT
0x5B3...eddC4 (99.999999
GAS LIMIT
3000000
VALUE
0
wei
CONTRACT
LedgerBalance - Mapping.sol
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 15
Deployed Contracts
TYPES AT 0XD91...39138 (MEMORY)
TEST AT 0XD8B...33FA8 (MEMORY)
TYPES AT 0XDA0...42B53 (MEMORY)
TEST AT 0X9D7...B5E99 (MEMORY)
LEDGERBALANCE AT 0XD2A...FD005 (I
updateBalance
Low level interactions
CALLDATA
Transact
Function_program.sol
Mapping.sol
pragma solidity ^0.5.0;
contract LedgerBalance {
mapping(address => uint) balance;
function updateBalance() public returns(uint)
balance[msg.sender]=20;
return balance[msg.sender];
}

```
pragma solidity ^0.5.0;
contract LedgerBalance {
mapping(address => string) name;

function updateBalance() public returns(string memory){
name[msg.sender] = "Dip";
return name[msg.sender];
}
function printsender() public view returns(address) {
return msg.sender;
}
}
```

Output:




SOLIDITY COMPILER
COMPILER
0.5.17+commit.d19bba13
Include nightly builds
LANGUAGE
Solidity
EVM VERSION
compiler default
COMPILER CONFIGURATION
Auto compile
Enable optimization
200
Hide warnings
Compile Mapping_string.sol
CONTRACT
LedgerBalance (Mapping_string.sol)
Publish on Swarm
Publish on Ipfs
Compilation Details
ABI
Bytecode
Function_program.sol
Mapping.sol
Mapping_string.sol
pragma solidity ^0.5.0;
contract LedgerBalance {
mapping(address => string) name;
function updateBalance() public returns(string memory){
name[msg.sender] = "Dip";
return name[msg.sender];
}
function printsender() public view returns(address) {
return msg.sender;
}
}


TEST AT 0XD8B...33FA8 (MEMORY)
TYPES AT 0XDA0...42B53 (MEMORY)
TEST AT 0X9D7...B5E99 (MEMORY)
LEDGERBALANCE AT 0XD2A...FD005 (
LEDGERBALANCE AT 0X332...D4B6D (
updateBalance
printsender
0: address: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
Low level interactions
CALLDATA
Transact

Practical 6

Implement and demonstrate the use of the following in Solidity :

(I).Functions

(II).View Functions

(III).Pure Functions

(IV).Fallback Functions

(V).Function Overloading

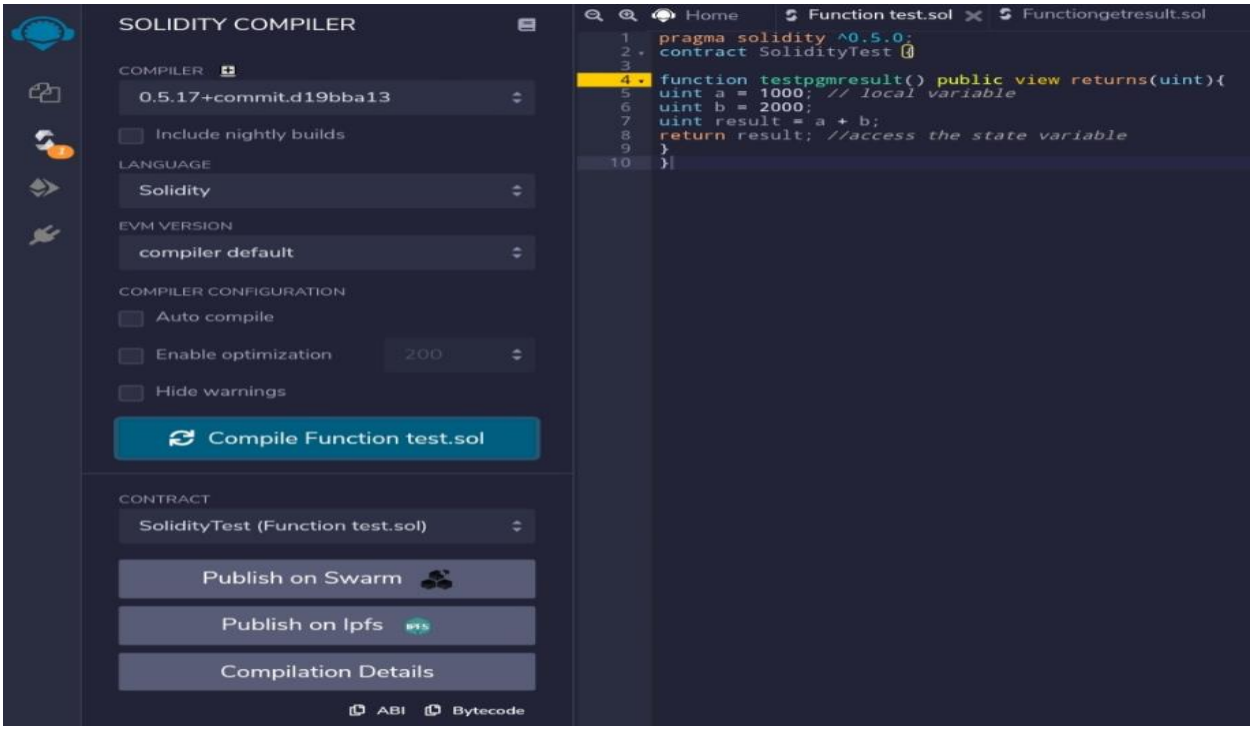
(VI).Mathematical Functions

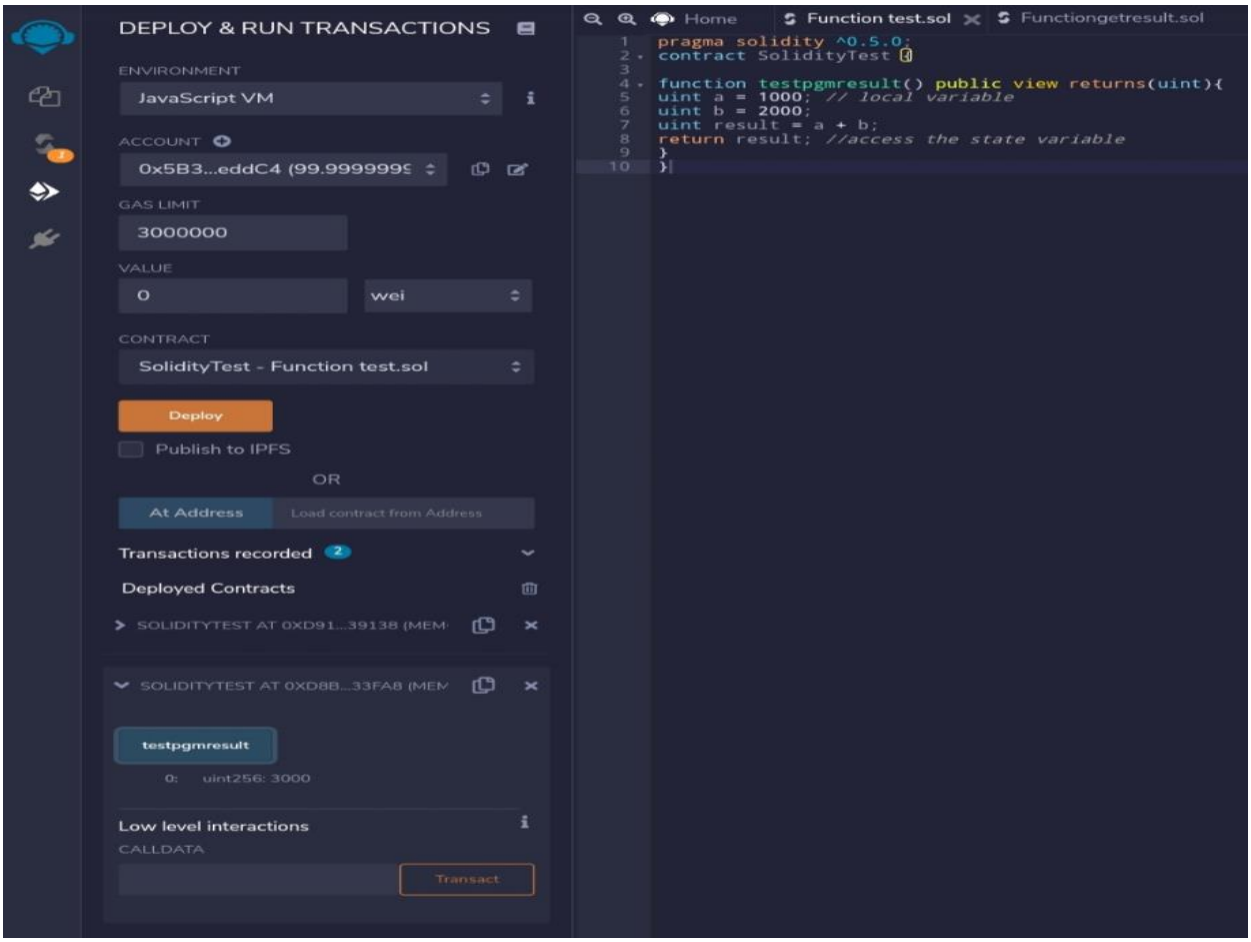
(VII).Cryptographic Functions

(I).Functions

```
pragma solidity ^0.5.0;
contract SolidityTest {
function testpgmresult() public view returns(uint){
uint a = 1000; // local variable
uint b = 2000;
uint result = a + b;
return result; //access the state variable
}
}
```

Output:


SOLIDITY COMPILER
COMPILER
0.5.17+commit.d19bba13
Include nightly builds
LANGUAGE
Solidity
EVM VERSION
compiler default
COMPILER CONFIGURATION
Auto compile
Enable optimization
200
Hide warnings
Compile Function test.sol
CONTRACT
SolidityTest (Function test.sol)
Publish on Swarm
Publish on Ipfs
Compilation Details
ABI
Bytecode
Home
Function test.sol
Functiongetresult.sol
pragma solidity ^0.5.0;
contract SolidityTest {
function testpgmresult() public view returns(uint){
uint a = 1000; // local variable
uint b = 2000;
uint result = a + b;
return result; //access the state variable
}
}



(II).View Functions

pragma solidity ^0.5.0;

contract Test {

function getResult() public view returns(uint product, uint sum){

uint a = 1; // local variable

uint b = 2;

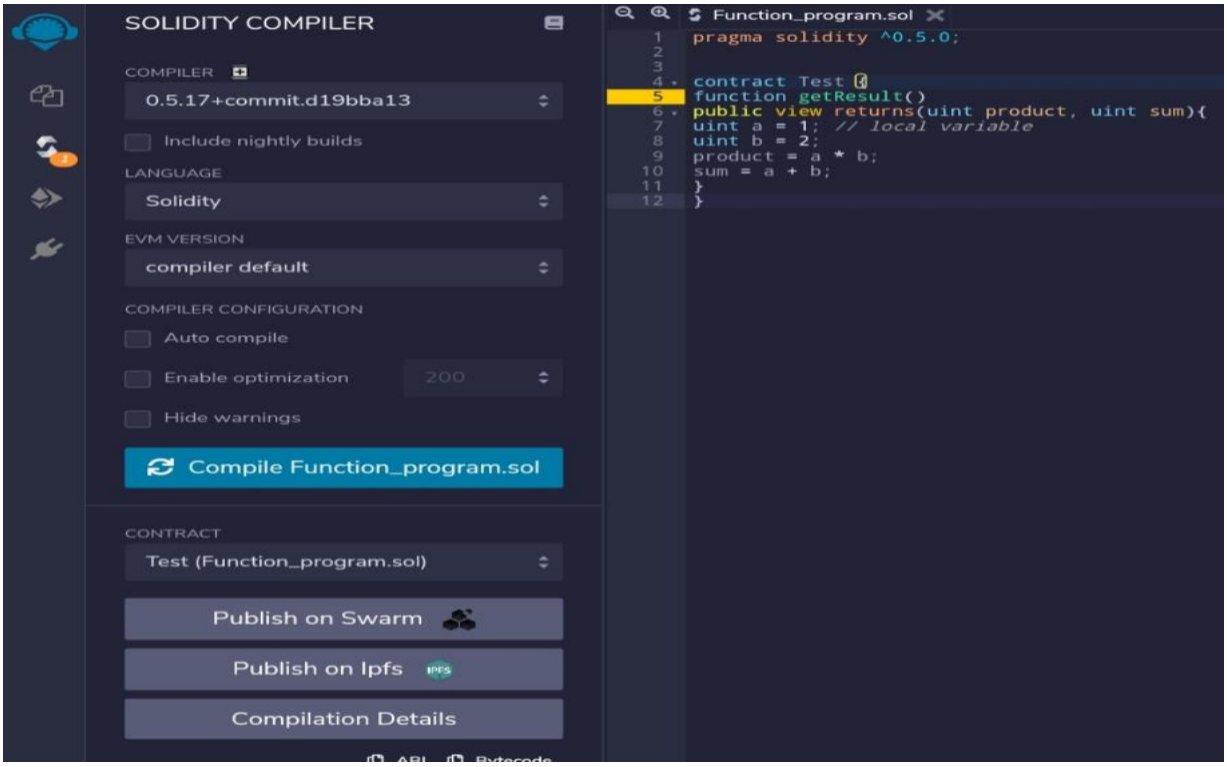
product = a * b;

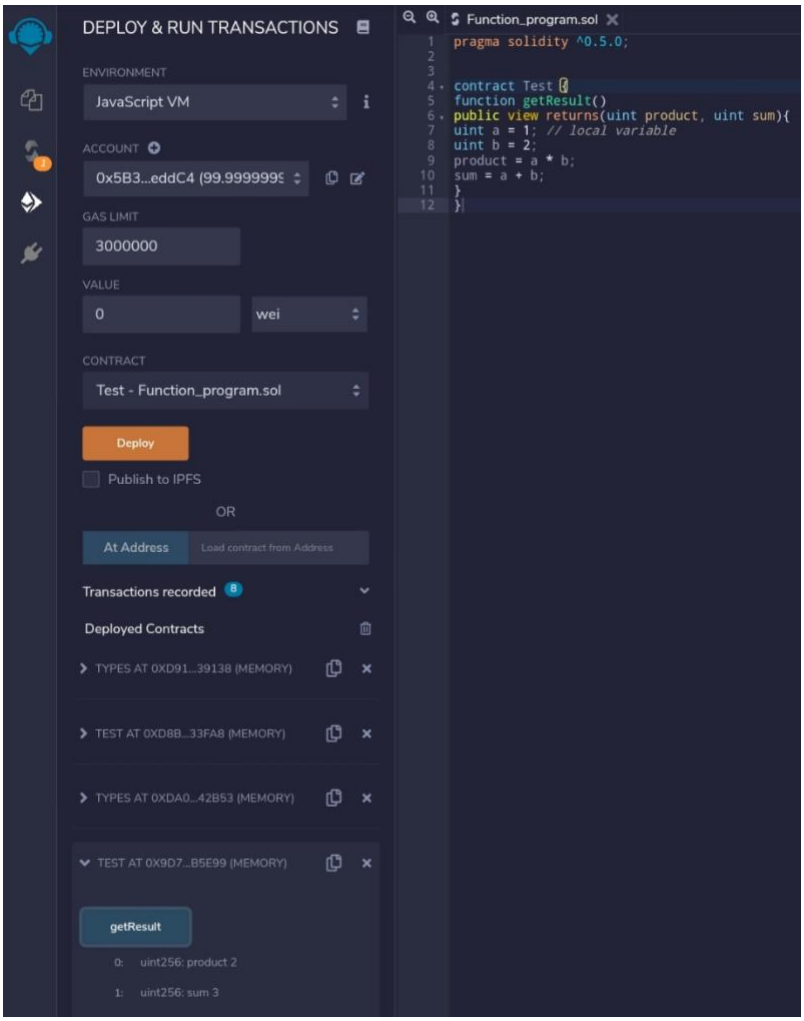
sum = a + b;

}

}

Output:


SOLIDITY COMPILER
COMPILER
0.5.17+commit.d19bba13
Include nightly builds
LANGUAGE
Solidity
EVM VERSION
compiler default
COMPILER CONFIGURATION
Auto compile
Enable optimization
200
Hide warnings
Compile Function_program.sol
CONTRACT
Test (Function_program.sol)
Publish on Swarm
Publish on Ipfs
Compilation Details
Function_program.sol
pragma solidity ^0.5.0;
contract Test {
function getResult()
public view returns(uint product, uint sum){
uint a = 1; // local variable
uint b = 2;
product = a * b;
sum = a + b;
}
}



(III).Pure Functions

pragma solidity ^0.5.0;

contract C {

//private state variable

uint private data;

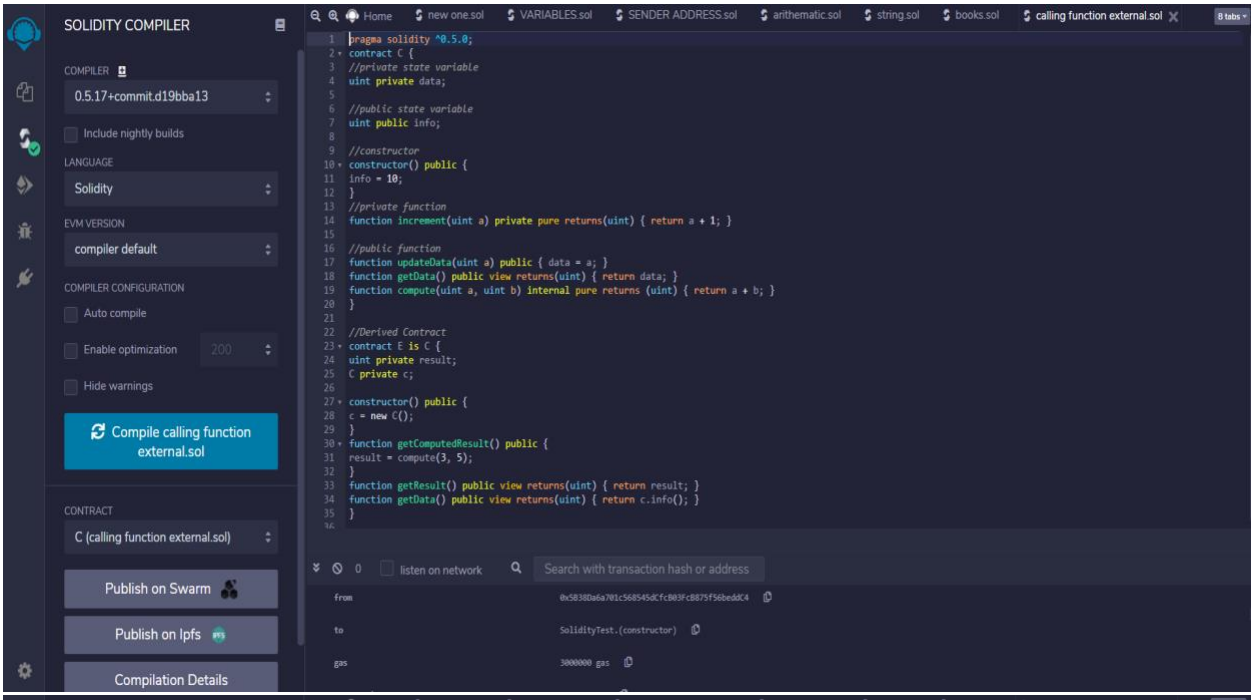
//public state variable

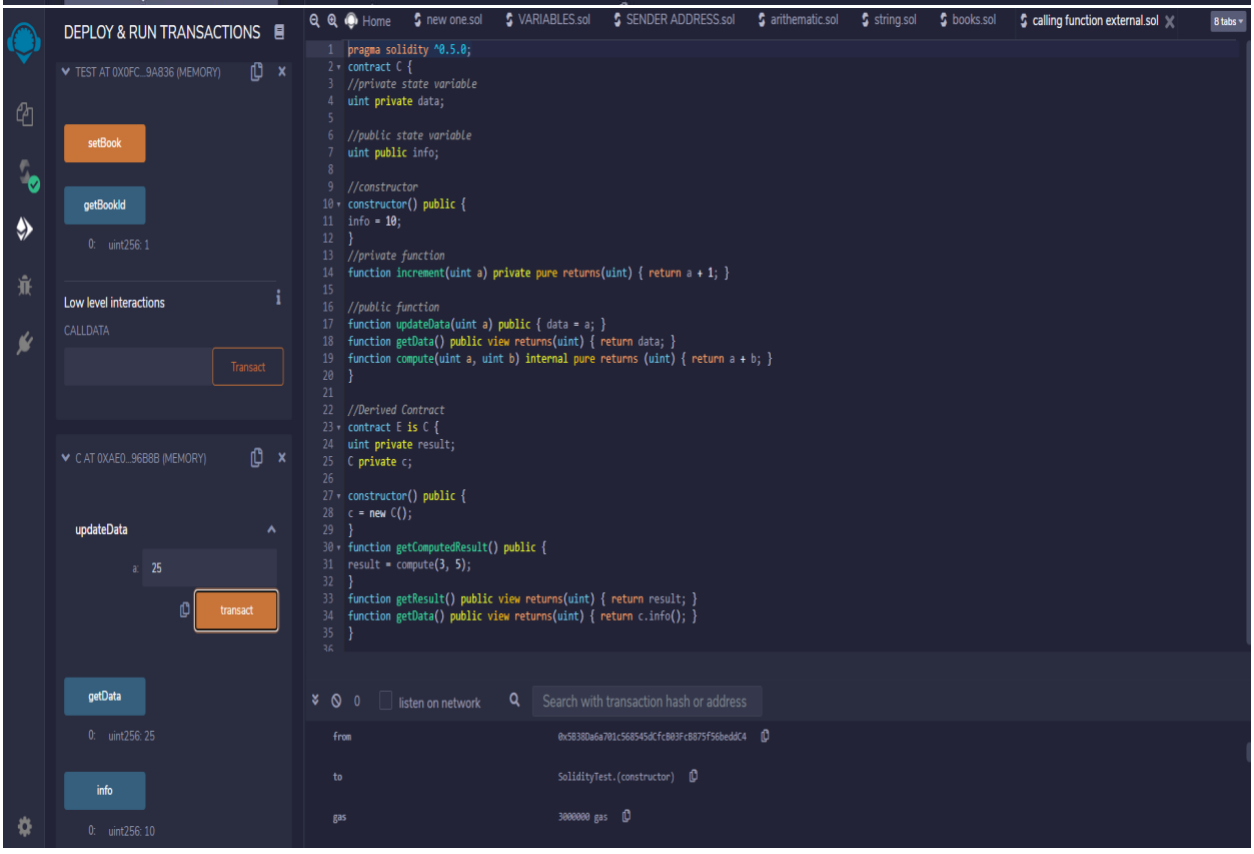
uint public info;

//constructor

```
    constructor() public {
    info = 10;
    }
    //private function
    function increment(uint a) private pure returns(uint) { return a + 1; }

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

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

    constructor() public {
    c = new C();
    }
    function getComputedResult() public {
    result = compute(3, 5);
    }
    function getResult() public view returns(uint) { return result; }
    function getData() public view returns(uint) { return c.info(); }
    }
```

Output:

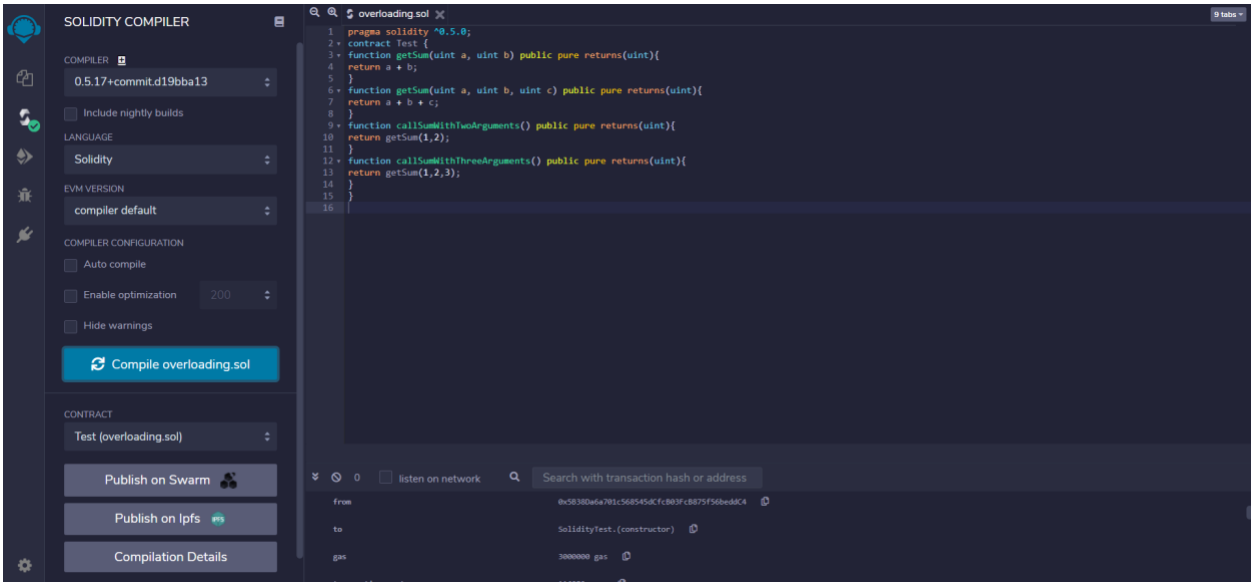

SOLIDITY COMPILER
COMPILER
0.5.17+commit.d19bba13
Include nightly builds
LANGUAGE
Solidity
EVM VERSION
compiler default
COMPILER CONFIGURATION
Auto compile
Enable optimization
200
Hide warnings
Compile calling function external.sol
CONTRACT
C (calling function external.sol)
Publish on Swarm
Publish on Ipfs
Compilation Details
Home
new one.sol
VARIABLES.sol
SENDER ADDRESS.sol
arithematic.sol
string.sol
books.sol
calling function external.sol
8 tabs
pragma solidity ^0.5.0;
contract C {
//private state variable
uint private data;
//public state variable
uint public info;
//constructor
constructor() public {
info = 10;
}
//private function
function increment(uint a) private pure returns(uint) { return a + 1; }
//public function
function updateData(uint a) public { data = a; }
function getData() public view returns(uint) { return data; }
function compute(uint a, uint b) internal pure returns (uint) { return a + b; }
}
//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(); }
}
listen on network
Search with transaction hash or address
from
0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to
SolidityTest.(constructor)
gas
3000000 gas

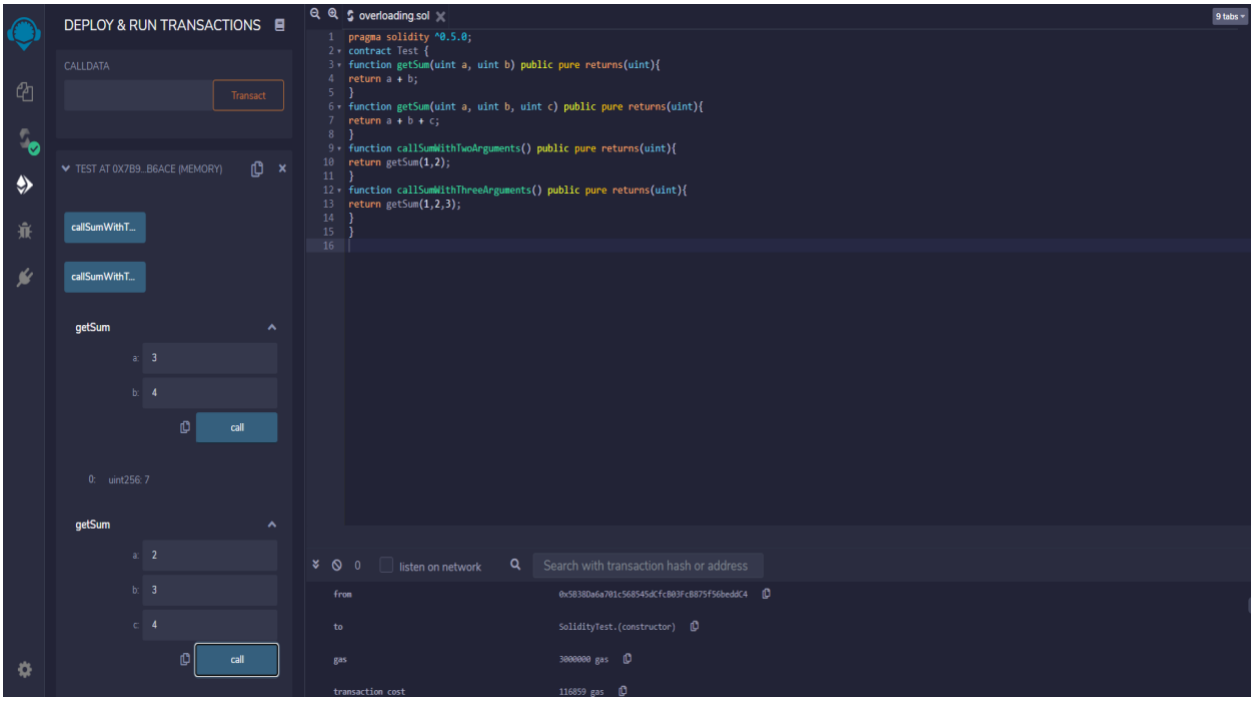

DEPLOY & RUN TRANSACTIONS
TEST AT 0X0FC...9A836 (MEMORY)
setBook
getBookId
0: uint256: 1
Low level interactions
CALLDATA
Transact
C AT 0XAE0...96B8B (MEMORY)
updateData
a: 25
transact
getData
0: uint256: 25
info
0: uint256: 10
Home
new one.sol
VARIABLES.sol
SENDER ADDRESS.sol
arithematic.sol
string.sol
books.sol
calling function external.sol
8 tabs
pragma solidity ^0.5.0;
contract C {
//private state variable
uint private data;
//public state variable
uint public info;
//constructor
constructor() public {
info = 10;
}
//private function
function increment(uint a) private pure returns(uint) { return a + 1; }
//public function
function updateData(uint a) public { data = a; }
function getData() public view returns(uint) { return data; }
function compute(uint a, uint b) internal pure returns (uint) { return a + b; }
}
//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(); }
}
listen on network
Search with transaction hash or address
from
0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to
SolidityTest.(constructor)
gas
3000000 gas

(V).Function Overloading

```
pragma solidity ^0.5.0;
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(1,2);
}
function callSumWithThreeArguments() public pure returns(uint){
return getSum(1,2,3);
}
}
```

Output:

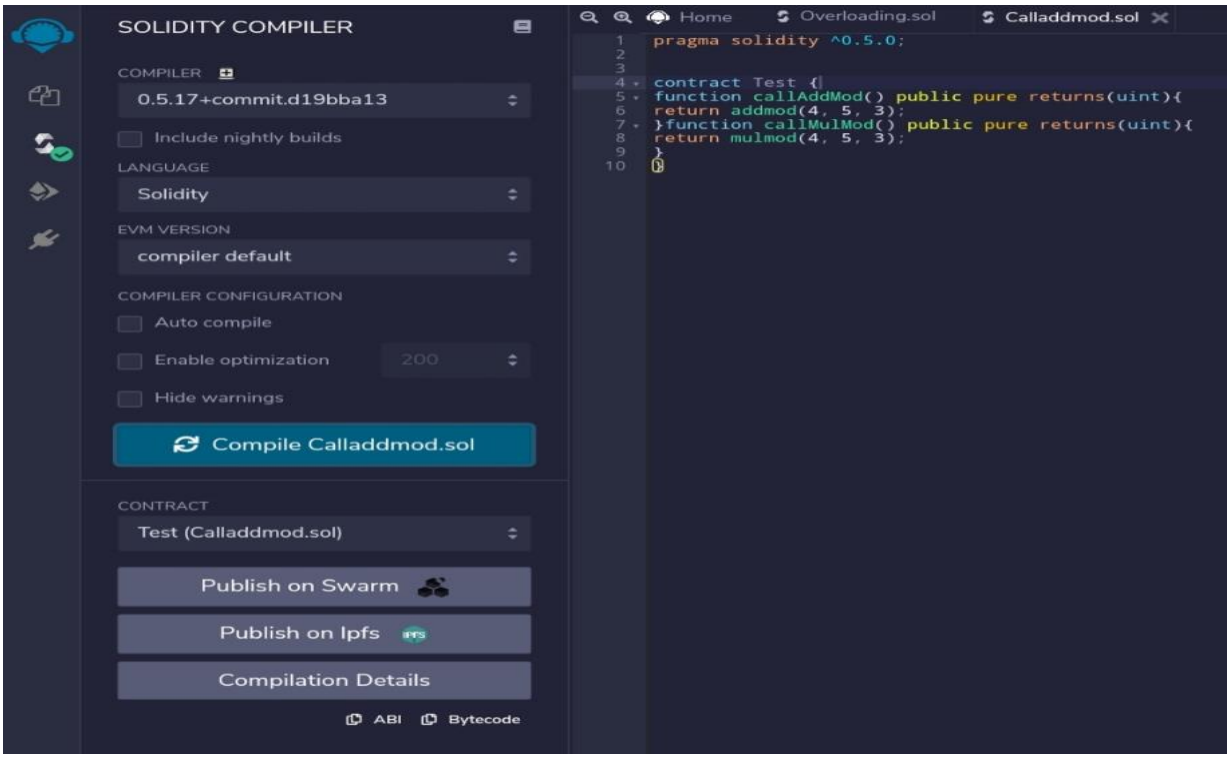


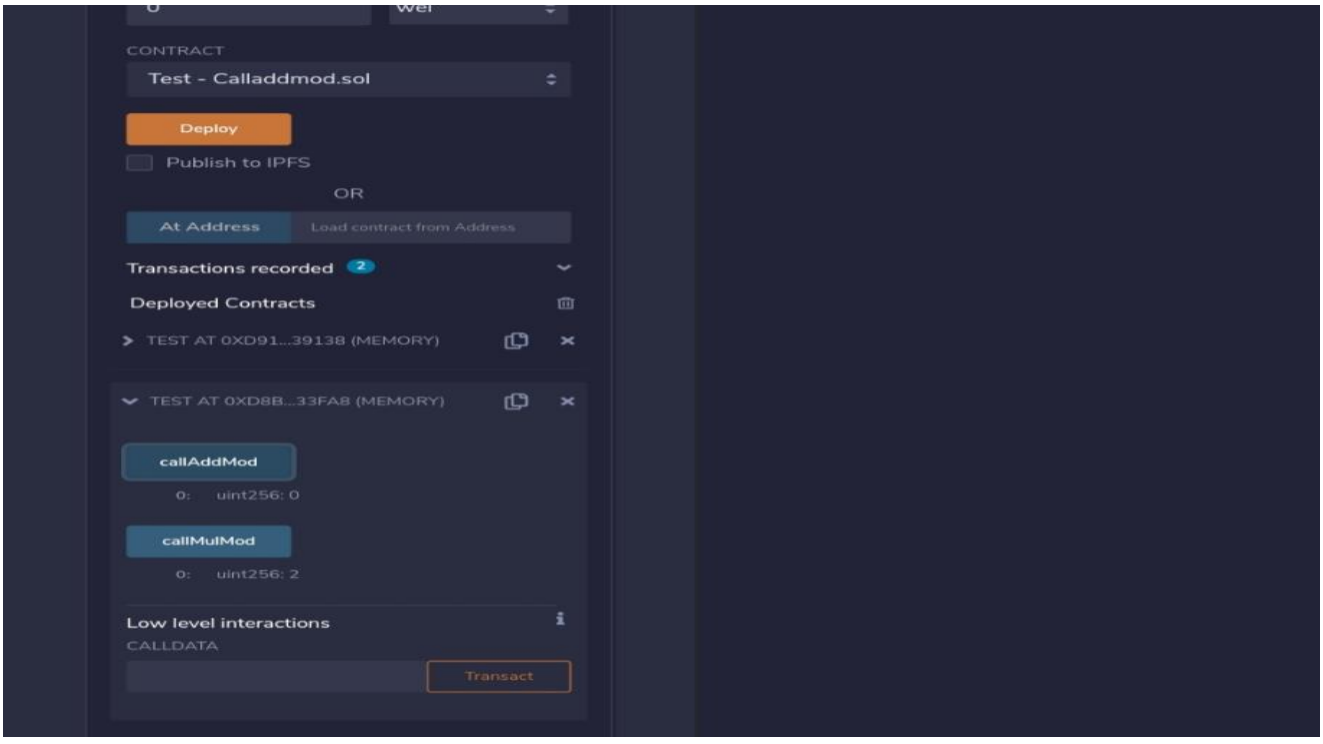


(VI).Mathematical Functions

```
pragma solidity ^0.5.0;
contract Test {
function callAddMod() public pure returns(uint){
return addmod(4, 5, 3);
}
function callMulMod() public pure returns(uint){
return mulmod(4, 5, 3);
}
}
```

Output:

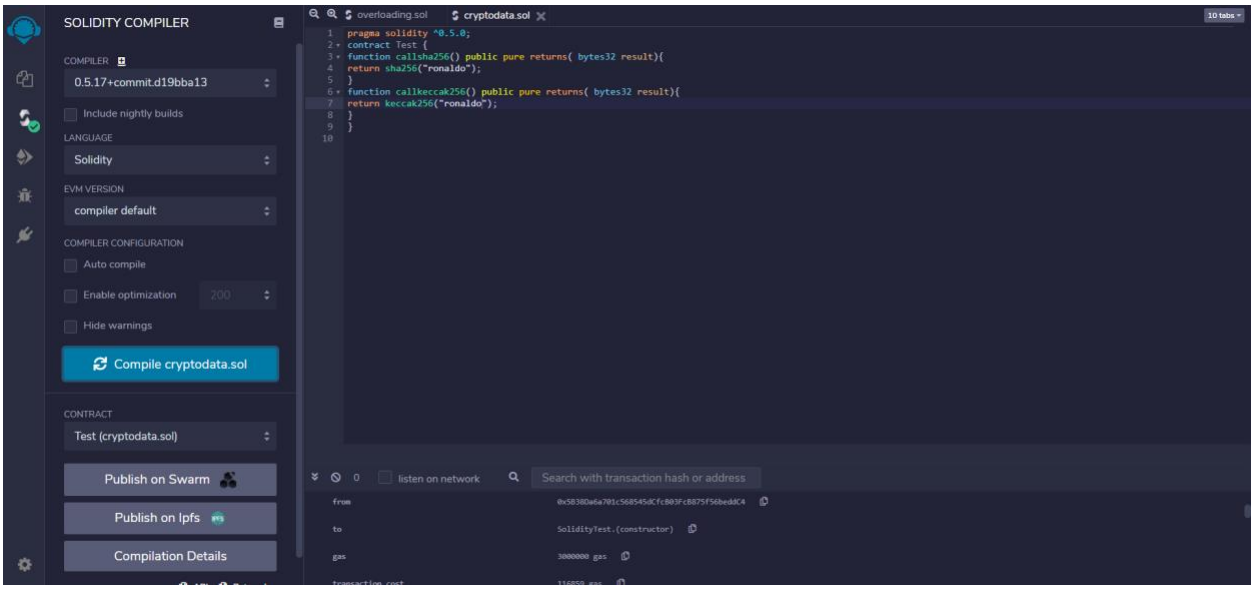

SOLIDITY COMPILER
COMPILER
0.5.17+commit.d19bba13
Include nightly builds
LANGUAGE
Solidity
EVM VERSION
compiler default
COMPILER CONFIGURATION
Auto compile
Enable optimization
200
Hide warnings
Compile Calladdmod.sol
CONTRACT
Test (Calladdmod.sol)
Publish on Swarm
Publish on Ipfs
Compilation Details
ABI
Bytecode
Home
Overloading.sol
Calladdmod.sol
pragma solidity ^0.5.0;
contract Test {
function callAddMod() public pure returns(uint){
return addmod(4, 5, 3);
}function callMulMod() public pure returns(uint){
return mulmod(4, 5, 3);
}

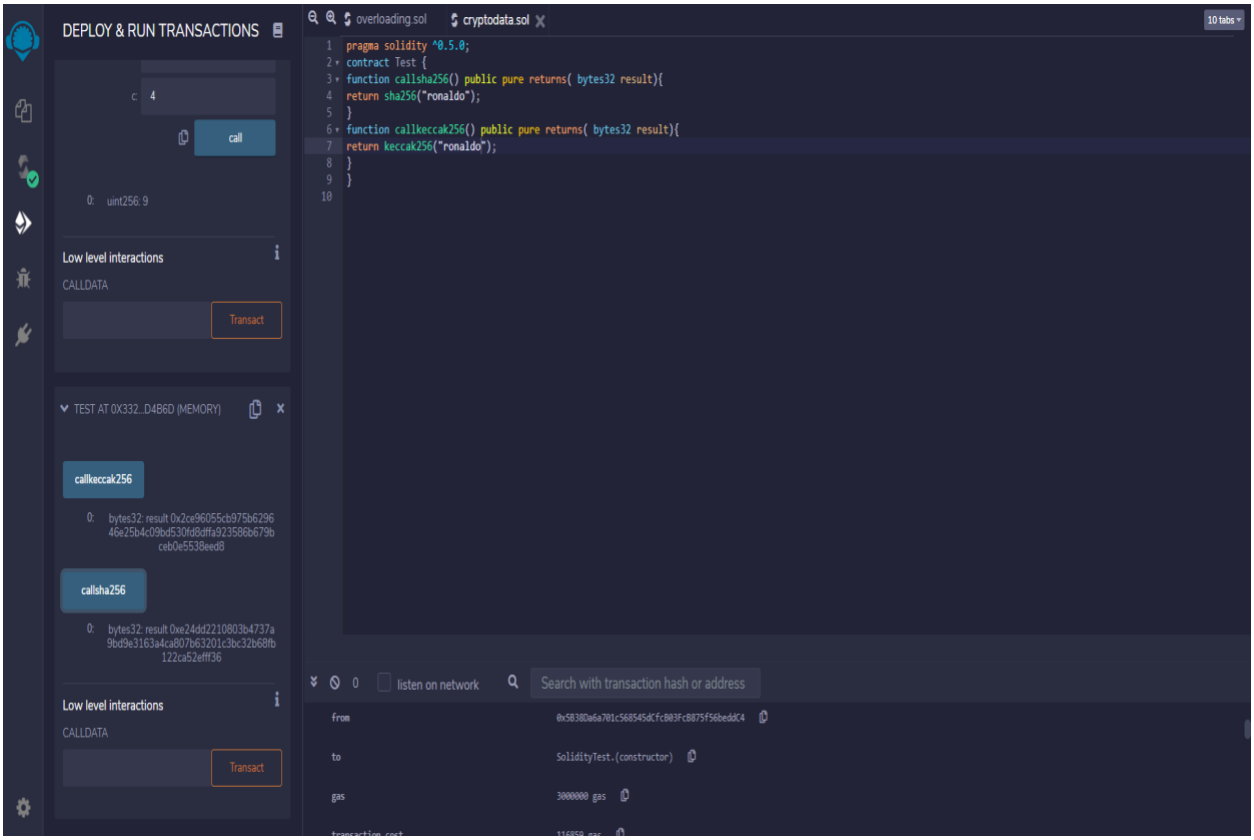

CONTRACT
Test - Calladdmod.sol
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded
2
Deployed Contracts
TEST AT 0XD91...39138 (MEMORY)
TEST AT 0XD8B...33FA8 (MEMORY)
callAddMod
0: uint256: 0
callMulMod
0: uint256: 2
Low level interactions
CALLDATA
Transact

(VII).Cryptographic Functions

```
pragma solidity ^0.5.0;
contract Test {
function callsha256() public pure returns( bytes32 result){
return sha256("ronaldo");
}
function callkeccak256() public pure returns( bytes32 result){
return keccak256("ronaldo");
}
}
```

Output:





Practical 7

Implement and demonstrate the use of the following in Solidity :

(I).Contracts

(II).Inheritance

(III).Constructors

(IV).Abstract Class

(V).Interfaces

(I).Contracts

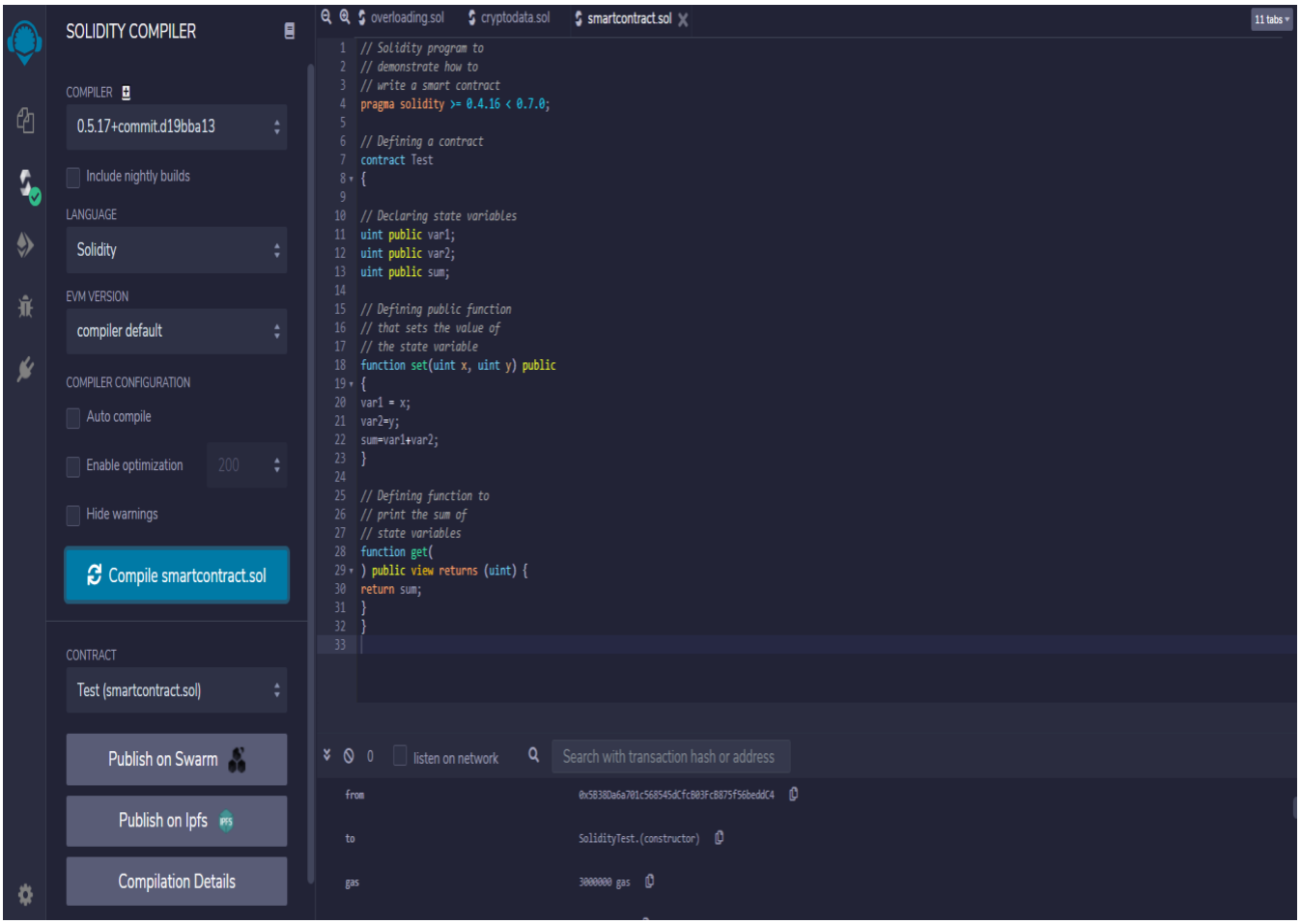
```
// Solidity program to
// demonstrate how to
// write a smart contract
pragma solidity >= 0.4.16 < 0.7.0;

// Defining a contract
contract Test
{

// Declaring state variables
uint public var1;
uint public var2;
uint public sum;

// Defining public function
// that sets the value of
// the state variable
function set(uint x, uint y) public
{
var1 = x;
var2=y;
sum=var1+var2;
}

// Defining function to
// print the sum of
// state variables
function get(
) public view returns (uint) {
```

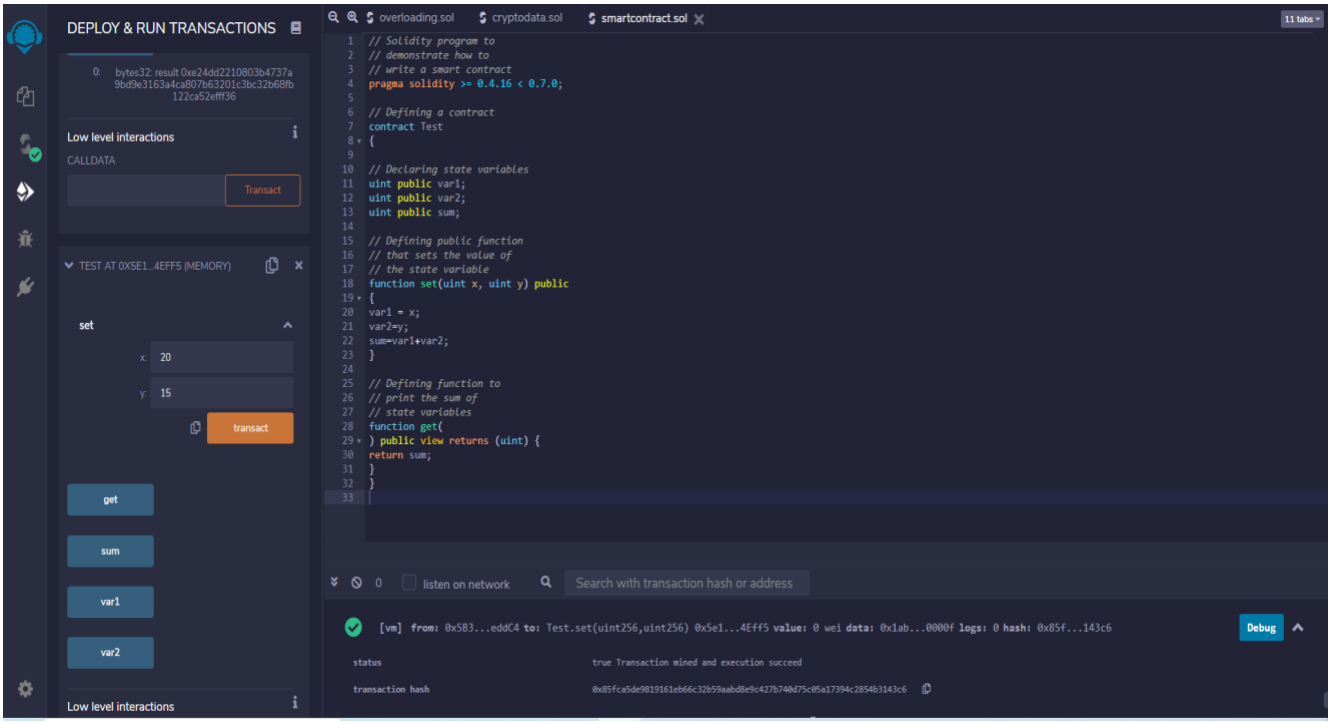
return sum;

}

}

Output:



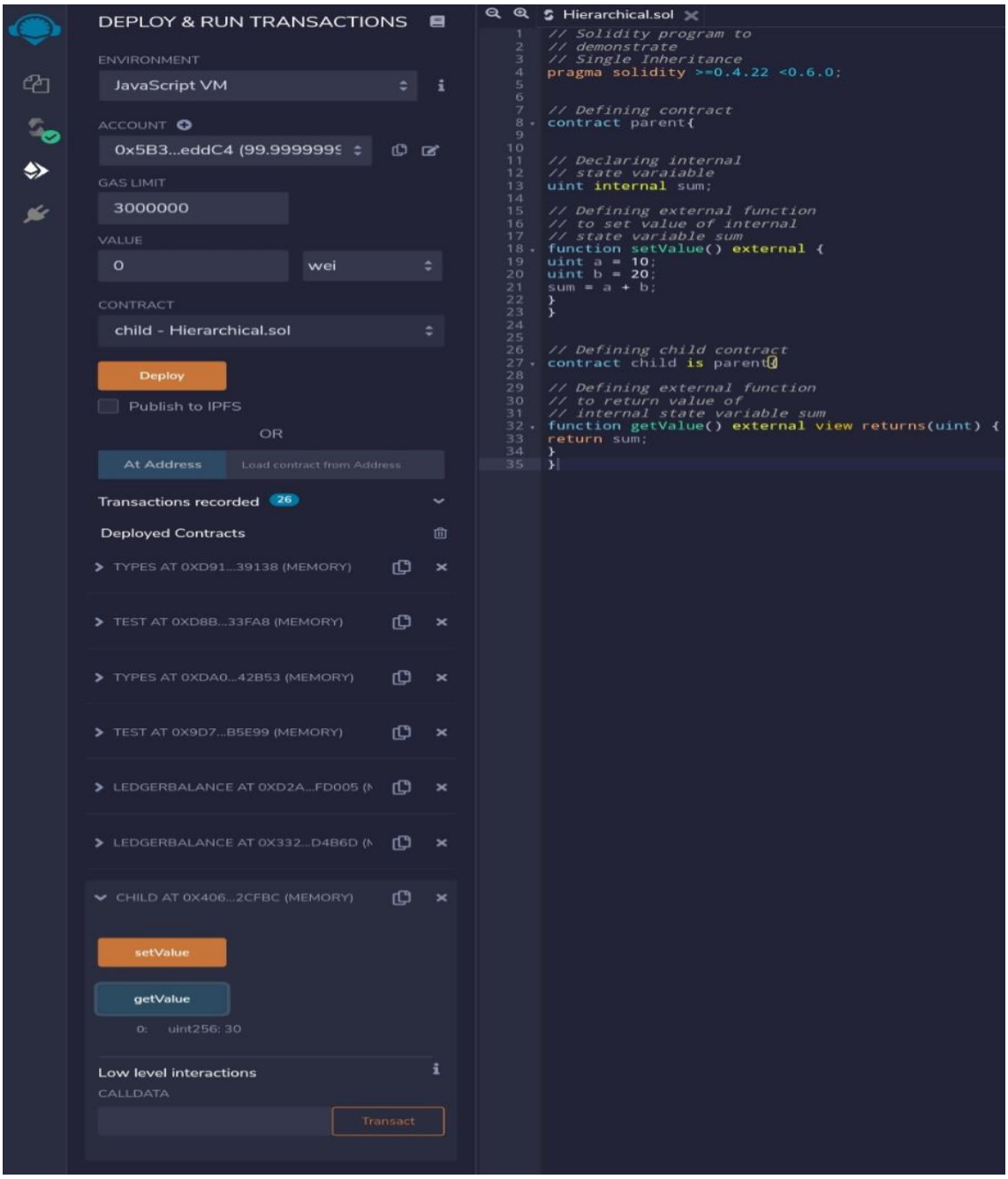


(II).Inheritance

```
// Solidity program to demonstrate Single Inheritance
pragma solidity >=0.4.22 <0.6.0;

// Defining contract
contract parent{

// Declaring internal state varaiable
uint internal sum;

// Defining external function to set value of internal state variable sum
function setValue() external {
uint a = 10;
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();
}
}
```

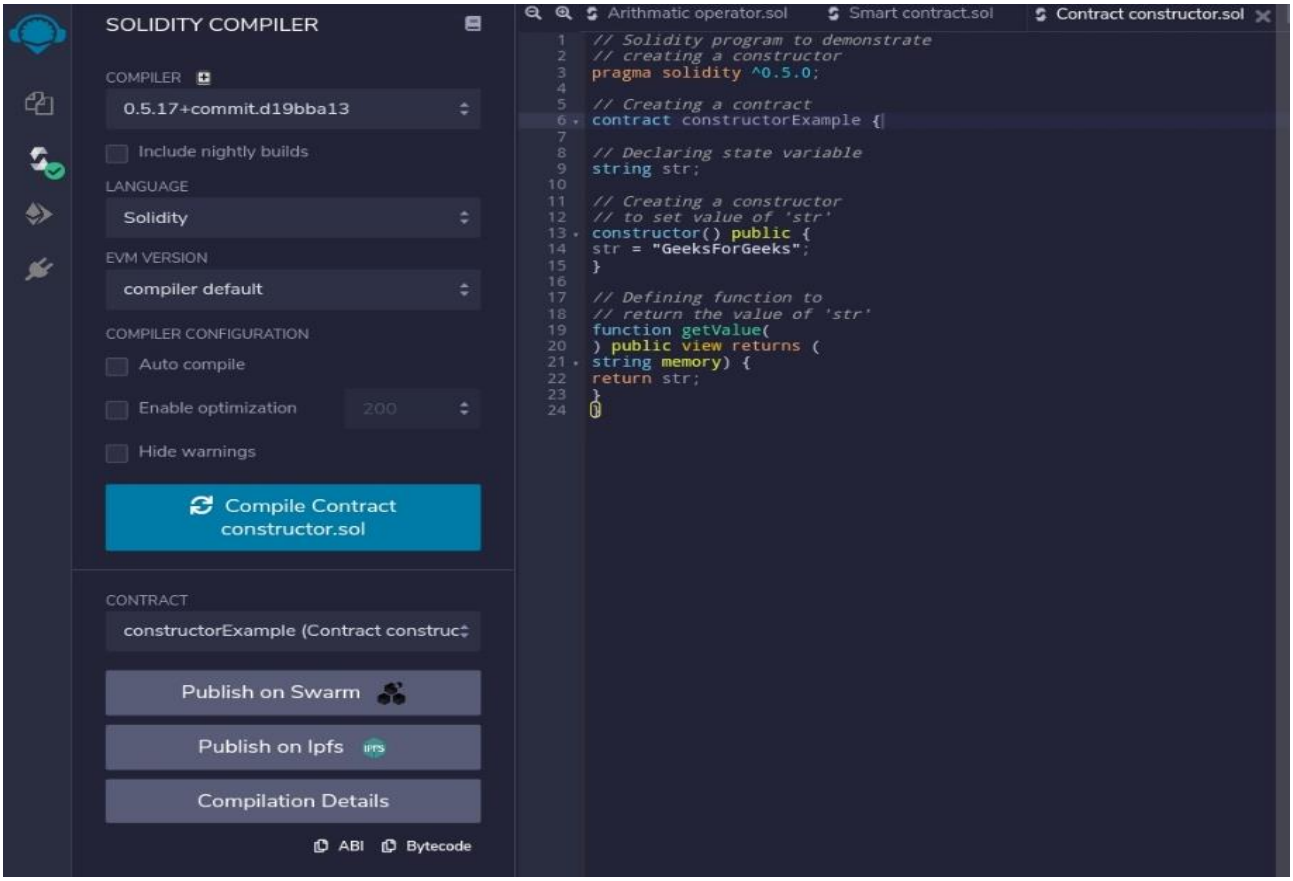
Output:

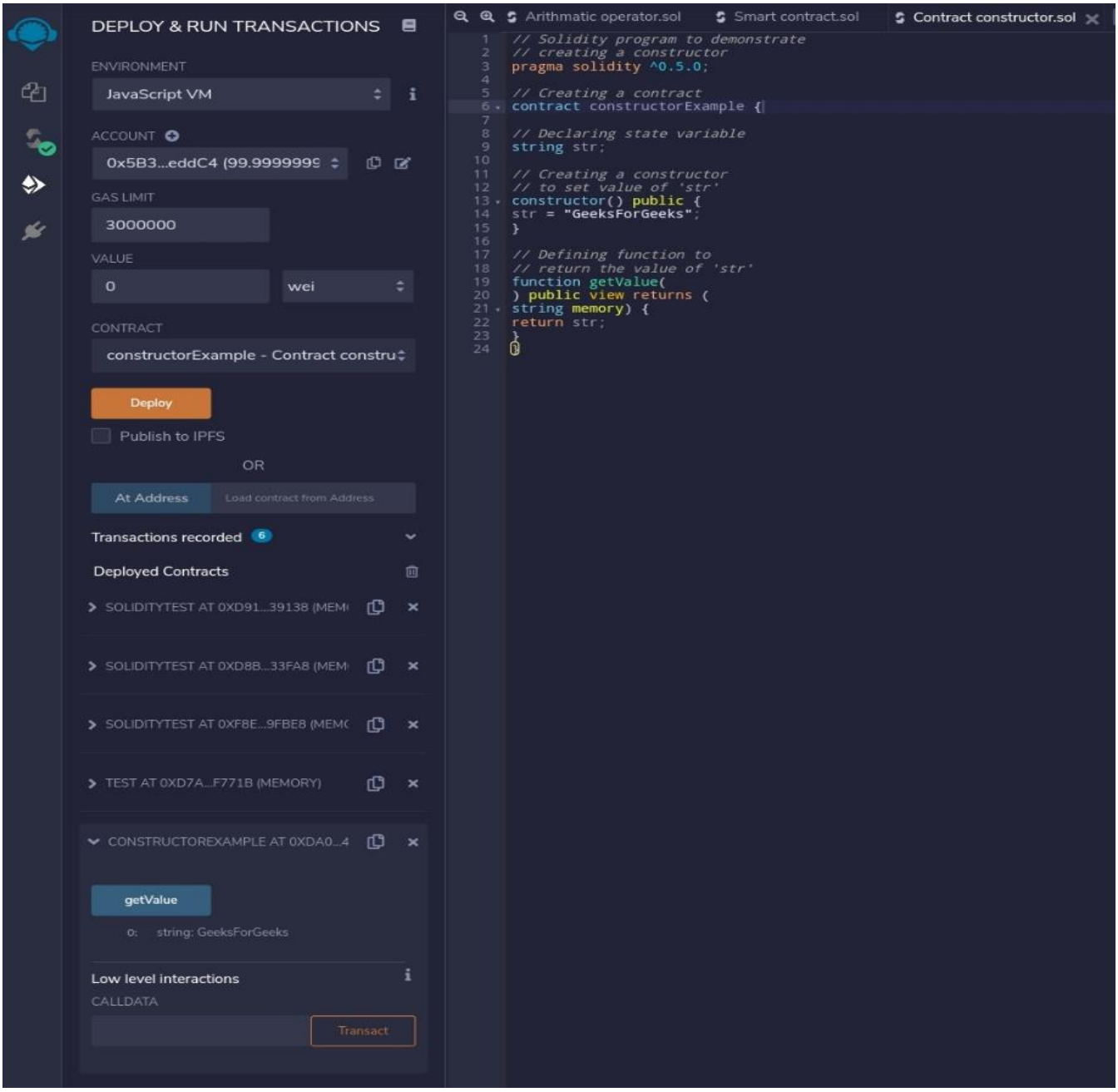

DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
JavaScript VM
ACCOUNT
0x5B3...eddC4 (99.999999
GAS LIMIT
3000000
VALUE
0
wei
CONTRACT
child - Hierarchical.sol
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 26
Deployed Contracts
TYPES AT 0XD91...39138 (MEMORY)
TEST AT 0XD8B...33FA8 (MEMORY)
TYPES AT 0XDA0...42B53 (MEMORY)
TEST AT 0X9D7...B5E99 (MEMORY)
LEDGERBALANCE AT 0XD2A...FD005 (
LEDGERBALANCE AT 0X332...D4B6D (
CHILD AT 0X406...2CFBC (MEMORY)
setValue
getValue
0: uint256: 30
Low level interactions
CALLDATA
Transact
Hierarchical.sol
// Solidity program to
// demonstrate
// Single Inheritance
pragma solidity >=0.4.22 <0.6.0;
// Defining contract
contract parent{
// Declaring internal
// state varaiable
uint internal sum;
// Defining external function
// to set value of internal
// state variable sum
function setValue() external {
uint a = 10;
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;
}
}

(III).Constructors

```
// Solidity program to demonstrate
// creating a constructor
pragma solidity ^0.5.0;

// Creating a contract
contract constructorExample {

// Declaring state variable
string str;

// Creating a constructor
// to set value of 'str'
constructor() public {
str = "GeeksForGeeks";
}

// Defining function to
// return the value of 'str'
function getValue(
) public view returns (
string memory) {
return str;
}
}
```

Output:


SOLIDITY COMPILER
COMPILER
0.5.17+commit.d19bba13
Include nightly builds
LANGUAGE
Solidity
EVM VERSION
compiler default
COMPILER CONFIGURATION
Auto compile
Enable optimization
200
Hide warnings
Compile Contract
constructor.sol
CONTRACT
constructorExample (Contract construc
Publish on Swarm
Publish on Ipfs
Compilation Details
ABI
Bytecode
Arithmatic operator.sol
Smart contract.sol
Contract constructor.sol
// Solidity program to demonstrate
// creating a constructor
pragma solidity ^0.5.0;
// Creating a contract
contract constructorExample {
// Declaring state variable
string str;
// Creating a constructor
// to set value of 'str'
constructor() public {
str = "GeeksForGeeks";
}
// Defining function to
// return the value of 'str'
function getValue(
) public view returns (
string memory) {
return str;
}


DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
JavaScript VM
ACCOUNT
0x5B3...eddC4 (99.9999999
GAS LIMIT
3000000
VALUE
0
wei
CONTRACT
constructorExample - Contract constru
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 6
Deployed Contracts
SOLIDITYTEST AT 0XD91...39138 (MEM
SOLIDITYTEST AT 0XD8B...33FA8 (MEM
SOLIDITYTEST AT 0XF8E...9FBE8 (MEM
TEST AT 0XD7A...F771B (MEMORY)
CONSTRUCTOREXAMPLE AT 0XDA0...4
getValue
0: string: GeeksForGeeks
Low level interactions
CALLDATA
Transact
Arithmatic operator.sol
Smart contract.sol
Contract constructor.sol
// Solidity program to demonstrate
// creating a constructor
pragma solidity ^0.5.0;
// Creating a contract
contract constructorExample {
// Declaring state variable
string str;
// Creating a constructor
// to set value of 'str'
constructor() public {
str = "GeeksForGeeks";
}
// Defining function to
// return the value of 'str'
function getValue(
) public view returns (
string memory) {
return str;
}
}

Practical 8

Implement and demonstrate the use of the following in Solidity :

(I).Libraries

(II).Assembly

(III).Events

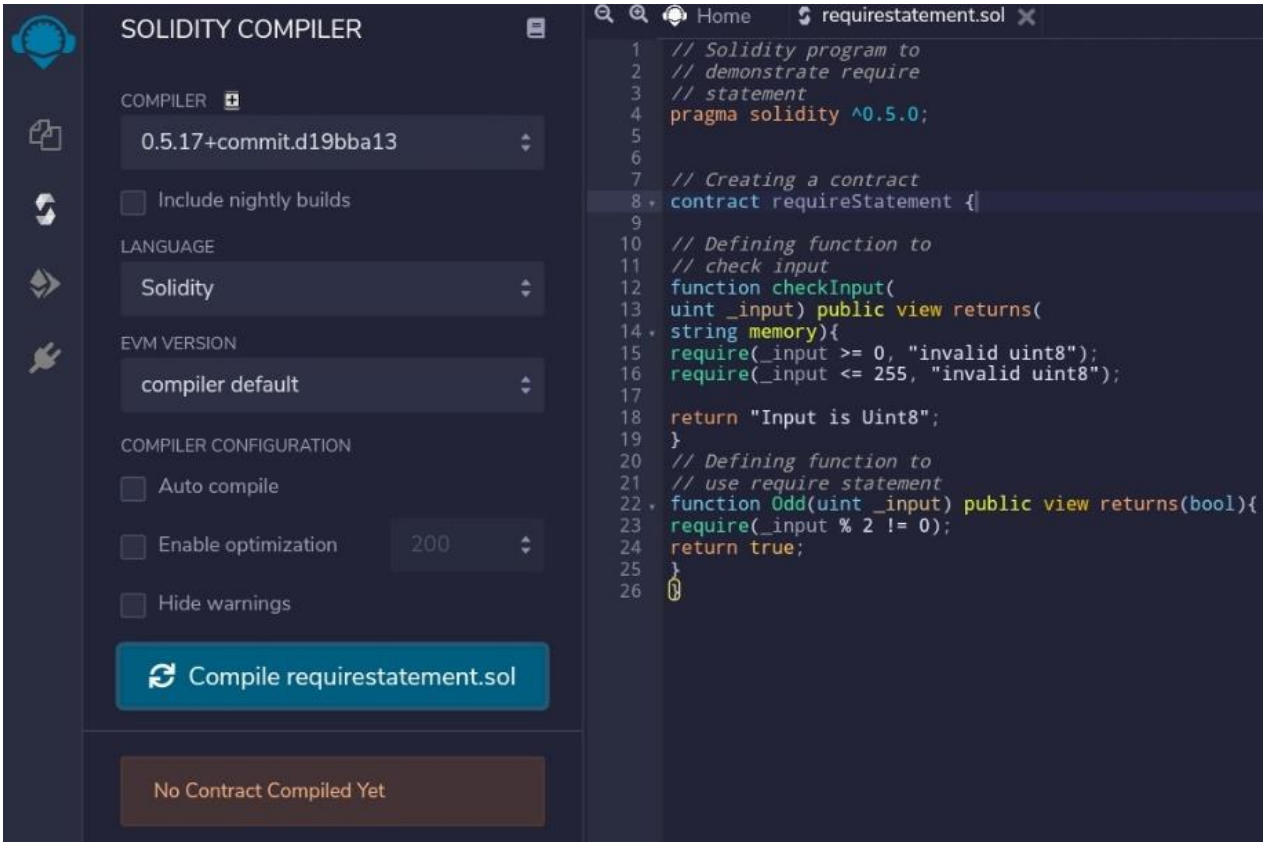
(IV).Error Handling

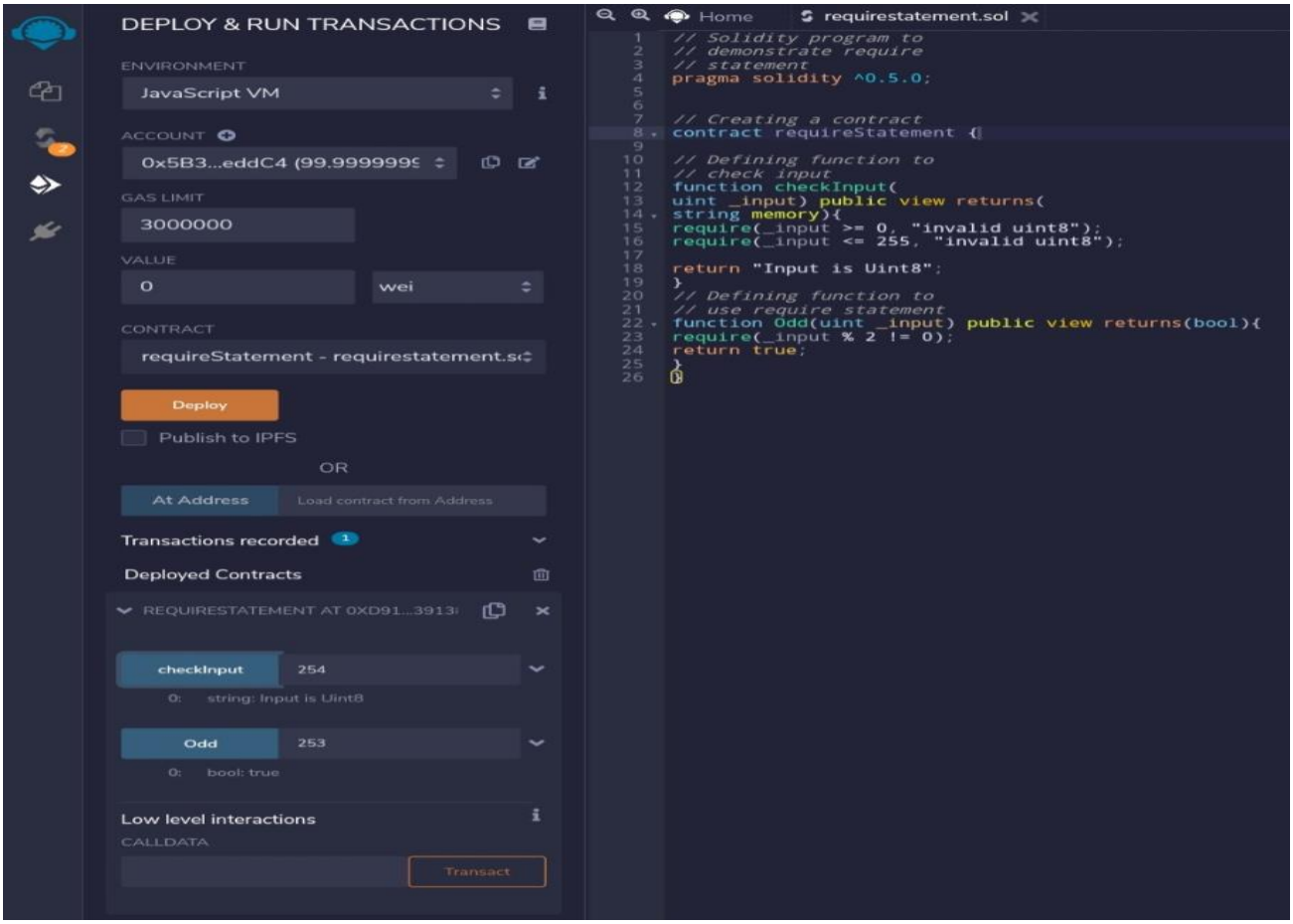
(IV).Error Handling

```
// Solidity program to demonstrate require statement
pragma solidity ^0.5.0;

// Creating a contract
contract requireStatement {

// Defining function to check input
function checkInput(
uint _input) public view returns(
string memory){
require(_input >= 0, "invalid uint8");
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;
}
```

}

Output:


SOLIDITY COMPILER
COMPILER
0.5.17+commit.d19bba13
Include nightly builds
LANGUAGE
Solidity
EVM VERSION
compiler default
COMPILER CONFIGURATION
Auto compile
Enable optimization
200
Hide warnings
Compile requirestatement.sol
No Contract Compiled Yet
Home
requirestatement.sol
// Solidity program to
// demonstrate require
// statement
pragma solidity ^0.5.0;
// Creating a contract
contract requireStatement {
// Defining function to
// check input
function checkInput(
uint _input) public view returns(
string memory){
require(_input >= 0, "invalid uint8");
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;
}
}



// **Solidity program to demonstrate assert statement**

pragma solidity ^0.5.0;

// Creating a contract

contract assertStatement {

// Defining a state variable

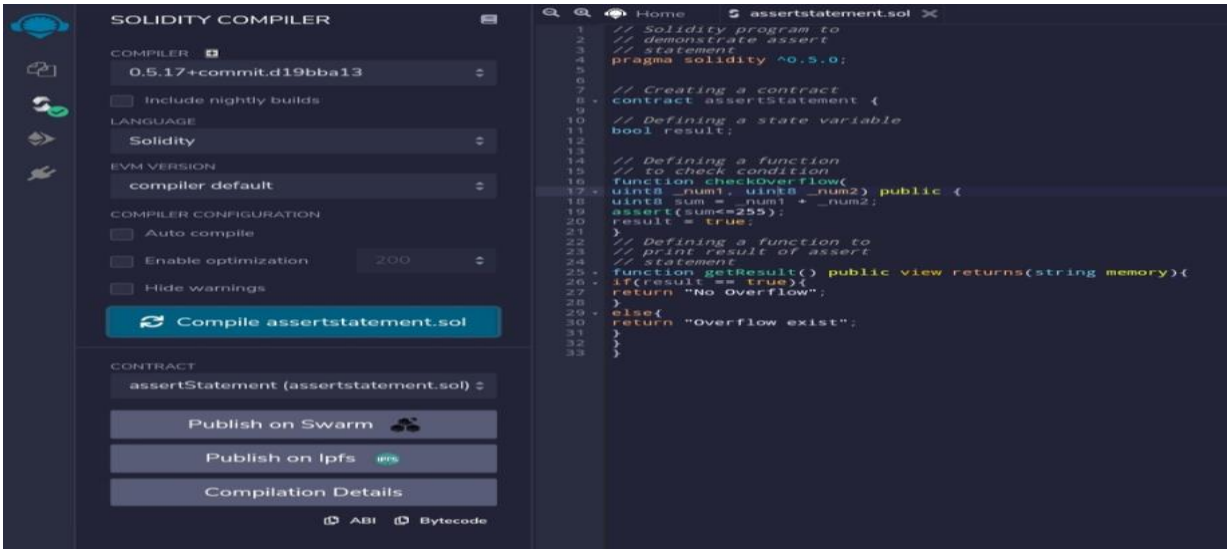
bool result;

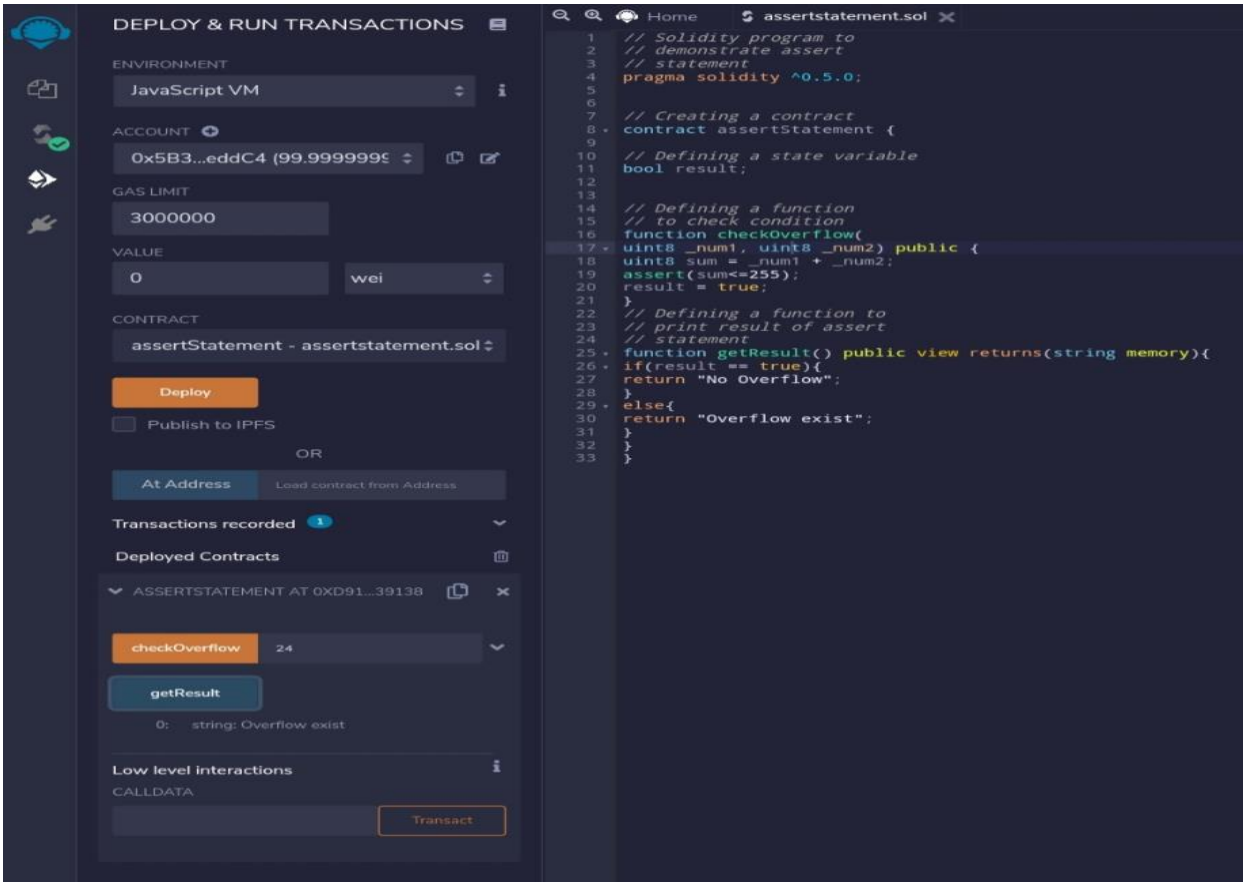
// Defining a function to check condition

function checkOverflow(

```
uint _num1, uint _num2) public {
uint8 sum = _num1 + _num2;
assert(sum<=255);
result = true;
}

// Defining a function to print result of assert statement
function getResult() public view returns(string memory){
if(result == true){
return "No Overflow";
}
else{
return "Overflow exist";
}
}
}
```

Output:





// **Solidity program to demonstrate assert statement**

pragma solidity ^0.5.0;

// Creating a contract

contract assertStatement {

// Defining a state variable

bool result;

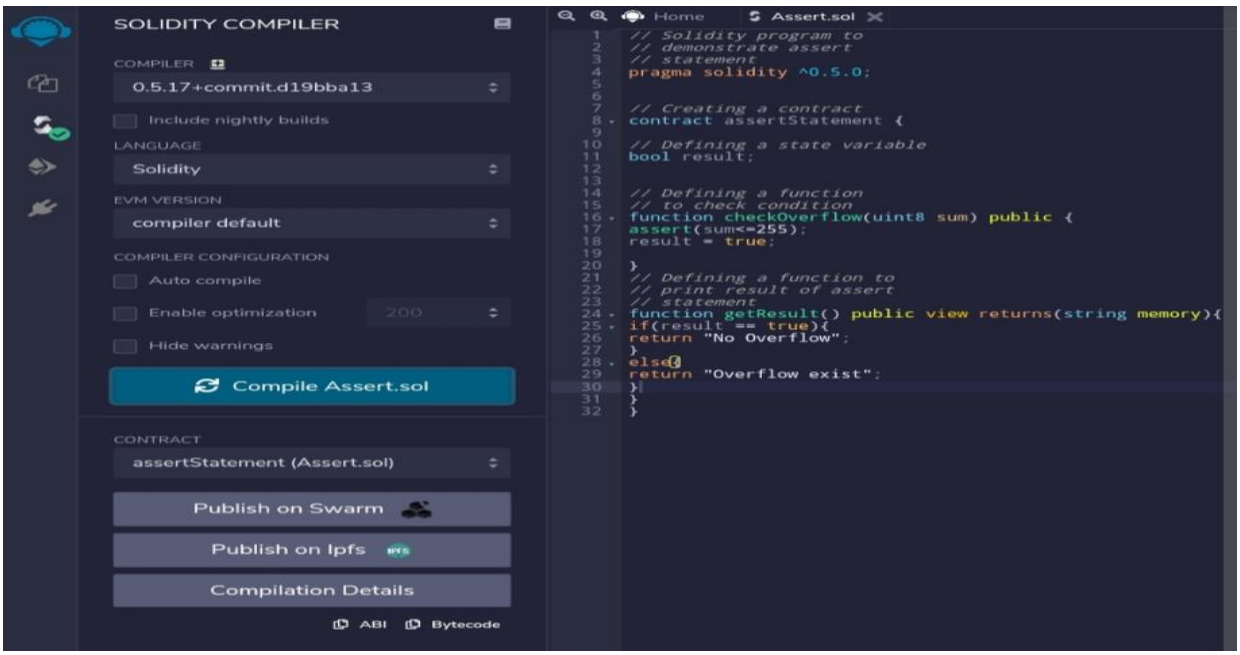
// Defining a function

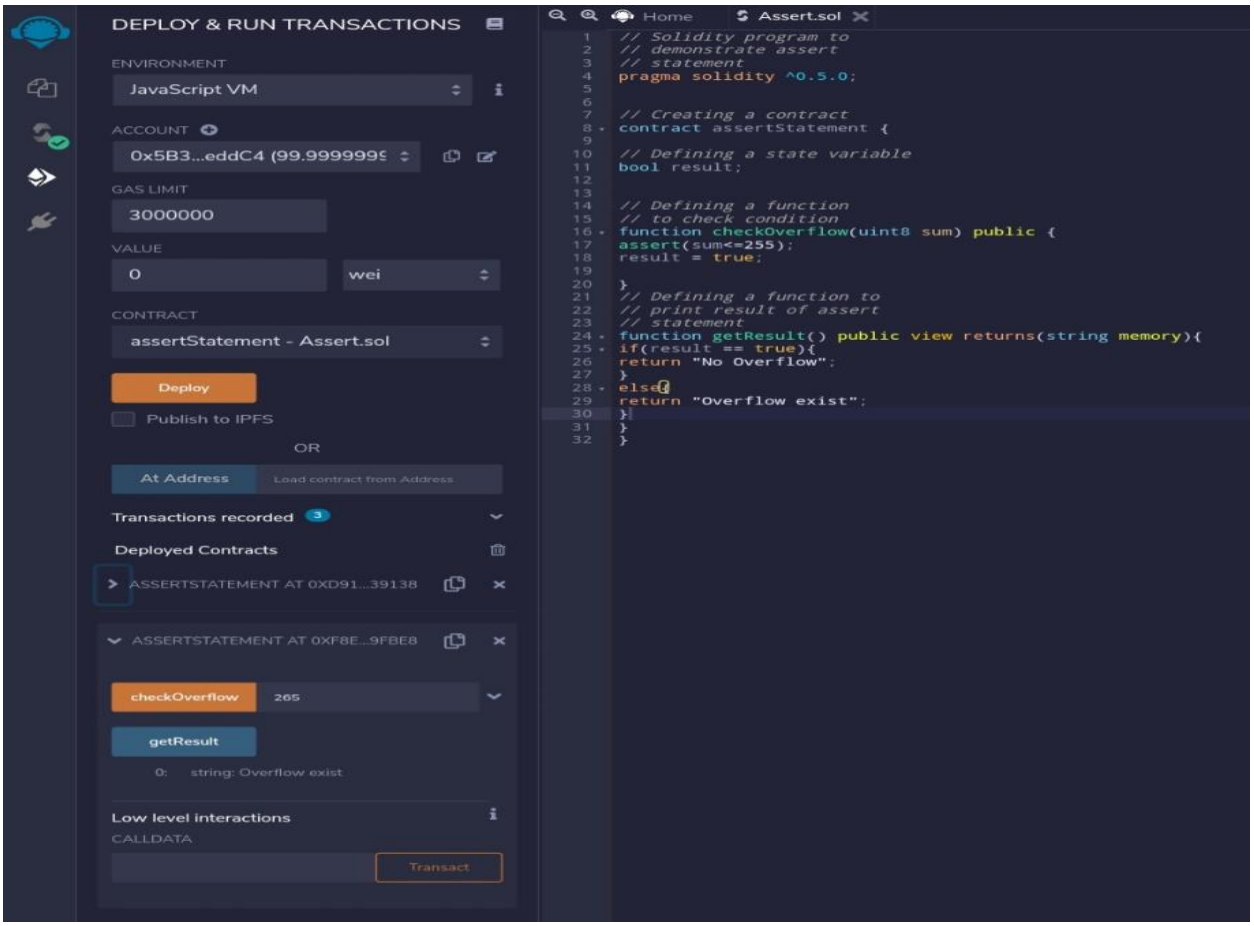
// to check condition

function checkOverflow(uint8 sum) public {

```
assert(sum<=255);
result = true;
}

// Defining a function to print result of assert statement
function getResult() public view returns(string memory){
if(result == true){
return "No Overflow";
}
else{
return "Overflow exist";
}
}
}
```

Output:





// **Solidity program to demonstrate revert statement**

pragma solidity ^0.5.0;

// Creating a contract

contract revertStatement {

// Defining a function to check condition

function checkOverflow(

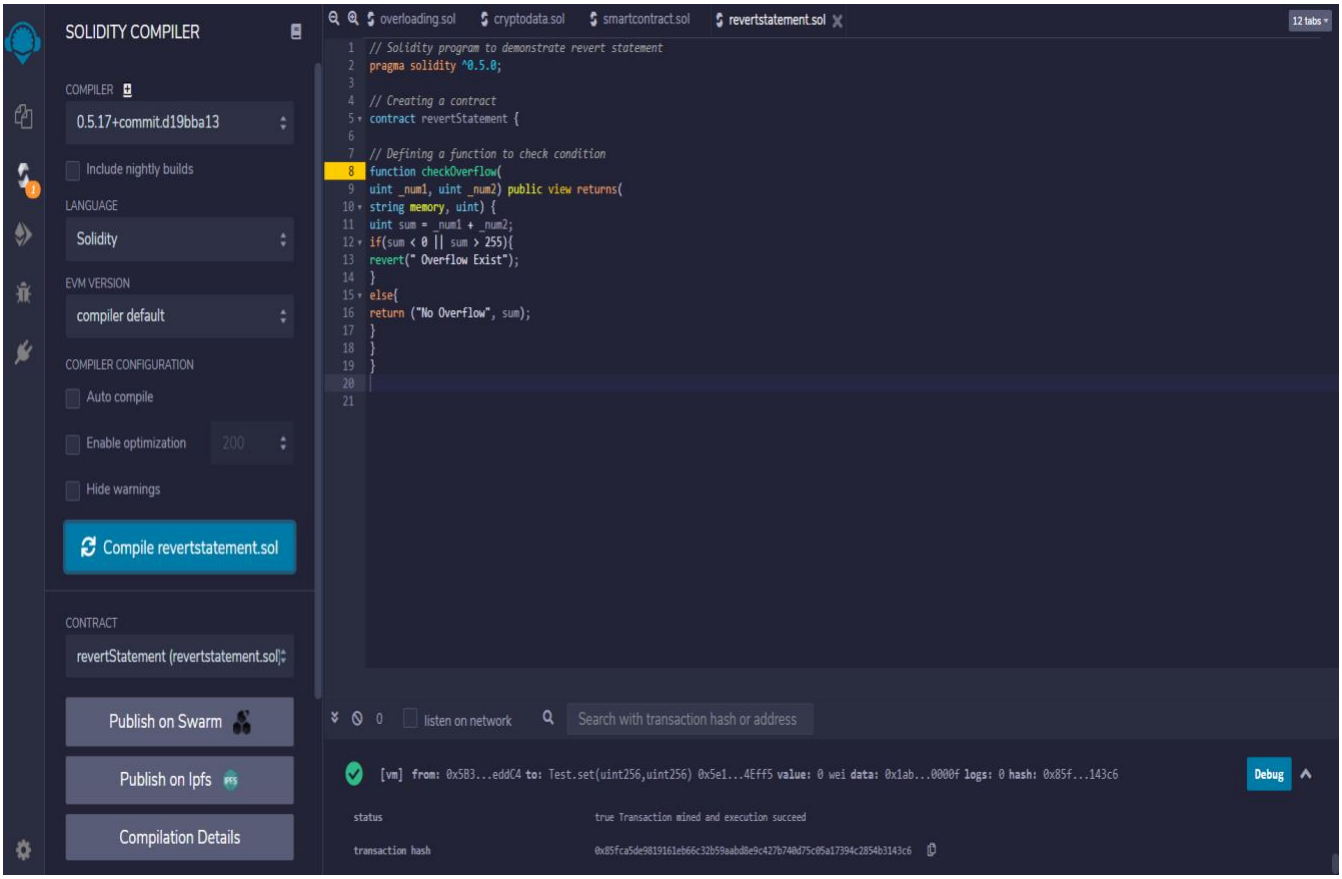
uint _num1, uint _num2) public view returns(

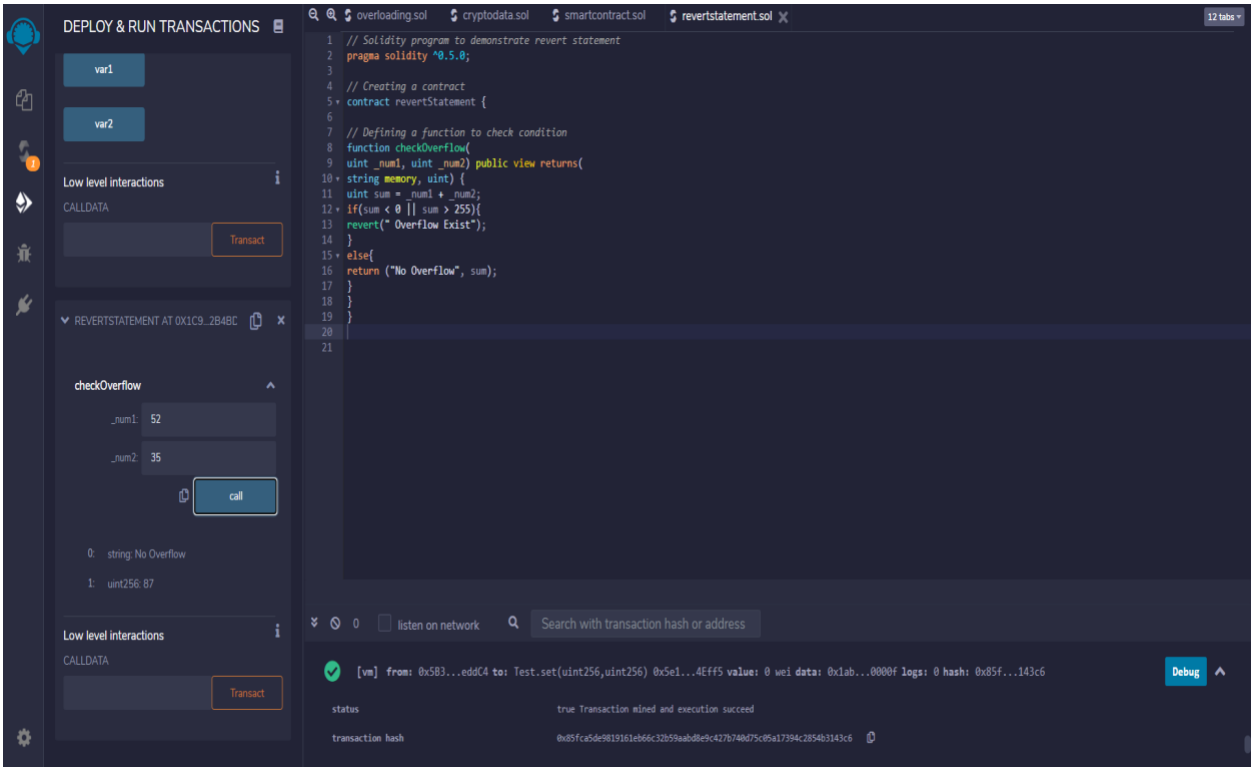
string memory, uint) {

uint sum = _num1 + _num2;

```
if(sum < 0 || sum > 255){
revert(" Overflow Exist");
}
else{
return ("No Overflow", sum);
}
}
}
}
```

Output:




DEPLOY & RUN TRANSACTIONS
var1
var2
Low level interactions
CALLDATA
Transact
REVERTSTATEMENT AT 0X1C9...2B4BC
checkOverflow
_num1: 52
_num2: 35
call
0: string: No Overflow
1: uint256: 87
Low level interactions
CALLDATA
Transact
overloading.sol
cryptodata.sol
smartcontract.sol
revertstatement.sol
12 tabs
// Solidity program to demonstrate revert statement
pragma solidity ^0.5.0;
// Creating a contract
contract revertStatement {
// Defining a function to check condition
function checkOverflow(
uint _num1, uint _num2) public view returns(
string memory, uint) {
uint sum = _num1 + _num2;
if(sum < 0 || sum > 255){
revert(" Overflow Exist");
}
else{
return ("No Overflow", sum);
}
}
}
listen on network
Search with transaction hash or address
[vm] from: 0x5B3...eddC4 to: Test.set(uint256,uint256) 0x5e1...4Eff5 value: 0 wei data: 0x1ab...0000f logs: 0 hash: 0x85f...143c6
Debug
status
true Transaction mined and execution succeed
transaction hash
0x85fca5de9819161eb66c32b59aabd8e9c427b740d75c05a17394c2854b3143c6