INDEX

List of Practical:	
1.	Write the following programs for Blockchain in Python:
a.	A simple client class that generates the private and public keys by using the built-in Python RSA algorithm and test it.
b.	A transaction class to send and receive money and test it.
c.	Create multiple transactions and display them.
d.	Create a blockchain, a genesis block and execute it.
e.	Create a mining function and test it.
f.	Add blocks to the miner and dump the blockchain.
2.	Implement and demonstrate the use of the following in Solidity:
a.	Variable, Operators, Loops, Decision Making, Strings, Arrays, Enums, Structs, Mappings, Conversions, Ether Units, Special Variables.
b.	Functions, Function Modifiers, View functions, Pure Functions, Fallback Function, Function Overloading, Mathematical functions, Cryptographic functions.
3.	Implement and demonstrate the use of the following in Solidity:
a.	Withdrawal Pattern, Restricted Access.
b.	Contracts, Inheritance, Constructors, Abstract Contracts, Interfaces.
c.	Libraries, Assembly, Events, Error handling.
4.	Install hyperledger fabric and composer. Deploy and execute the application.
5.	Demonstrate the running of the blockchain node.
6.	Demonstrate the use of Bitcoin Core API.
7.	Create your own blockchain and demonstrate its use.

Practical No: 1

Aim: Write the following programs for Blockchain in Python:

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

Code:

```
import hashlib
import random
import string
import json
import binascii
import numpy as np
import pandas as pd
import pylab as pl
import logging
import datetime
import collections
import Crypto
import Crypto.Random
from Crypto.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')
Dinesh = Client()
print (Dinesh.identity)
```

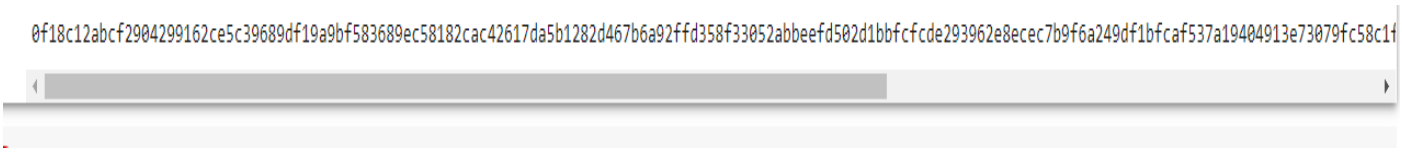
Output:

1b) A transaction class to send and receive money and test it.

Code:

```
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': se
lf.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:



1c) Create Multiple Transaction and display them.

Code:

```
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 ('-----')

  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)

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

Output:

```
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100c26c34efd785e255a351885faa617e22dbec4b8d1fdac6f833f00172431f5de6eecfe23c9fdcc14fd4de2dc947e11435fb572d39709f057b303
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100d31bde02b9759d166f6514250948351f541d716b1ef5d7f3d931de74527050671e57bc3f323940186b105c379f8cd6c28935e43ba5752434
-----
value: 15.0
-----
time: 2022-07-30 17:35:06.809777
-----
--------------
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100c26c34efd785e255a351885faa617e22dbec4b8d1fdac6f833f00172431f5de6eecfe23c9fdcc14fd4de2dc947e11435fb572d39709f057b303
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100a72cfb527dffd9615a33eaf34b0cdf26f68d5d604676786c77ea188380e8412420aee1cde5b6dc9e157b31874d7da7bfb40c5c835a0999e2
-----
value: 6.0
-----
time: 2022-07-30 17:35:06.811866
-----
--------------
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100d31bde02b9759d166f6514250948351f541d716b1ef5d7f3d931de74527050671e57bc3f323940186b105c379f8cd6c28935e43ba5752434a81
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100c7e4e5f5f46085be74cd9c2f0ee1a79942c8a7b10c103a72b542c3246adf0c14f7a3032f5d431942062f5747e2369e6894e51cfbacfb5d578
-----
value: 2.0
-----
time: 2022-07-30 17:35:06.814247
-----
--------------
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100a72cfb527dffd9615a33eaf34b0cdf26f68d5d604676786c77ea188380e8412420aee1cde5b6dc9e157b31874d7da7bfb40c5c835a0999e2b268
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100d31bde02b9759d166f6514250948351f541d716b1ef5d7f3d931de74527050671e57bc3f323940186b105c379f8cd6c28935e43ba5752434a
-----
value: 4.0
-----
time: 2022-07-30 17:35:06.816476
-----
--------------
```

1d) Create a blockchain, a genesis block and execute it.

Code:

```
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:

```
Number of blocks in the chain: 1
block # 0
sender: Genesis
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100a4a79ac0645d8483c7c97fb6b67e1bf90f17d87d1f
-----
value: 500.0
-----
time: 2022-07-30 17:50:16.288802
-----
--------------
====================================
```

1e) Create a mining function and test it.

Code:

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

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

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

Output:

```
after 238 iterations found nonce: 1124711db6185591392c6a06c24a3c2ebbaeca647fb8374824f939ada27c09e9
after 353 iterations found nonce: 11e0a4b57bb76496ecc6ab5a3c126165bc9dbbaf80f664e7e192a422360c3884
after 419 iterations found nonce: 11280dfd9ab05b3dbfa8699901537329414O8faa4a3b0832819b161f52321c08
after 511 iterations found nonce: 1150944bd7ea429acd052da390b148f27eb881686e0f8bfcdf77f833af878e2e
after 822 iterations found nonce: 110e61d41d94b48f6dfcb6f43f9ceafe2ab7168456dab088e05126d43d0366ef
after 924 iterations found nonce: 11147bf32bafeee4505b5cc40c3b227366d8a41ab3bd740e437c74400b7a8127
'6d80c4cf3cd4fb49aad25deced696c3a48e17974ecbfb1441e2128517f0152b2'
```

1f) Add blocks to the miner and dump the blockchain

Code:

```
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


# 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)
```

Output:

```
after 308 iterations found nonce: 111b1715b007a9920e76b241e9a4dde15a50b00faeed353bf9279311deb59804
after 398 iterations found nonce: 11a65418aca0201df85c2eeacabbcf0624092abb64bf6e159ccf5539cbb1cfe1
after 527 iterations found nonce: 11310267f14590e45be2375fe94e881c65f86020e3fcf7844a849241783680a7
after 718 iterations found nonce: 117ba8286d575afe73007e6f588b6f541580bec444d5165a3f71d9e4b623f3fa
after 308 iterations found nonce: 111b1715b007a9920e76b241e9a4dde15a50b00faeed353bf9279311deb59804
after 398 iterations found nonce: 11a65418aca0201df85c2eeacabbcf0624092abb64bf6e159ccf5539cbb1cfe1
after 527 iterations found nonce: 11310267f14590e45be2375fe94e881c65f86020e3fcf7844a849241783680a7
after 718 iterations found nonce: 117ba8286d575afe73007e6f588b6f541580bec444d5165a3f71d9e4b623f3fa
after 308 iterations found nonce: 111b1715b007a9920e76b241e9a4dde15a50b00faeed353bf9279311deb59804
after 398 iterations found nonce: 11a65418aca0201df85c2eeacabbcf0624092abb64bf6e159ccf5539cbb1cfe1
after 527 iterations found nonce: 11310267f14590e45be2375fe94e881c65f86020e3fcf7844a849241783680a7
after 718 iterations found nonce: 117ba8286d575afe73007e6f588b6f541580bec444d5165a3f71d9e4b623f3fa
```

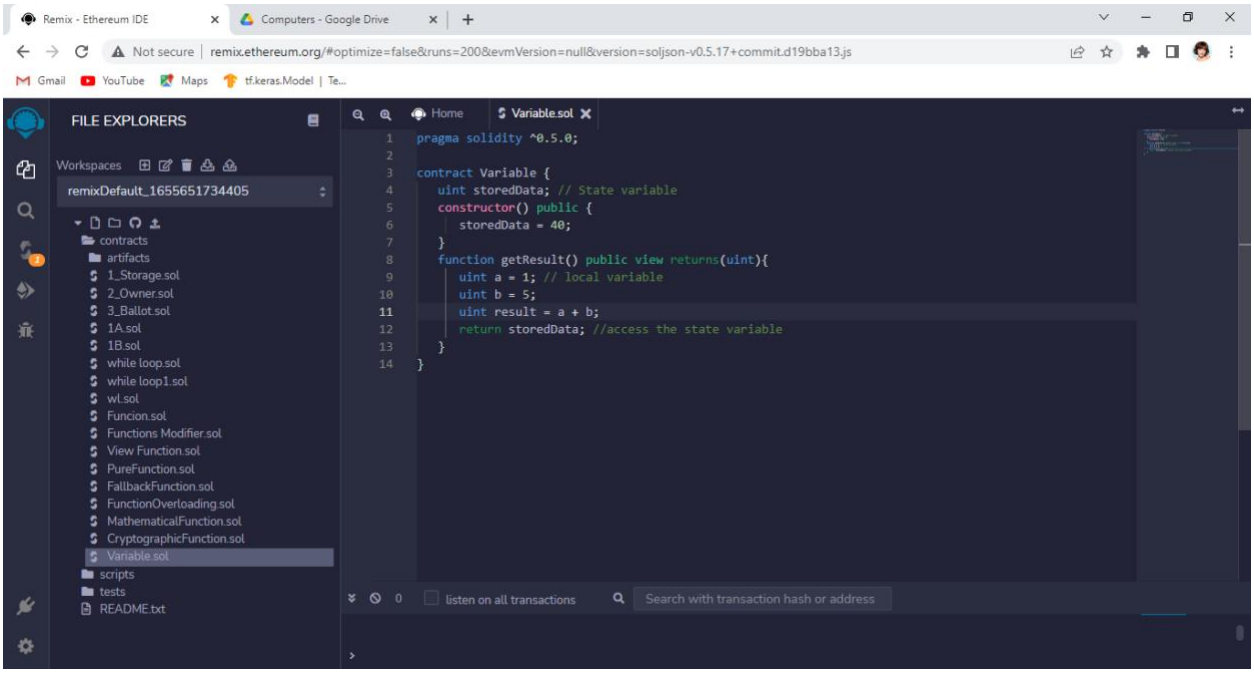
Practical No: 2

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

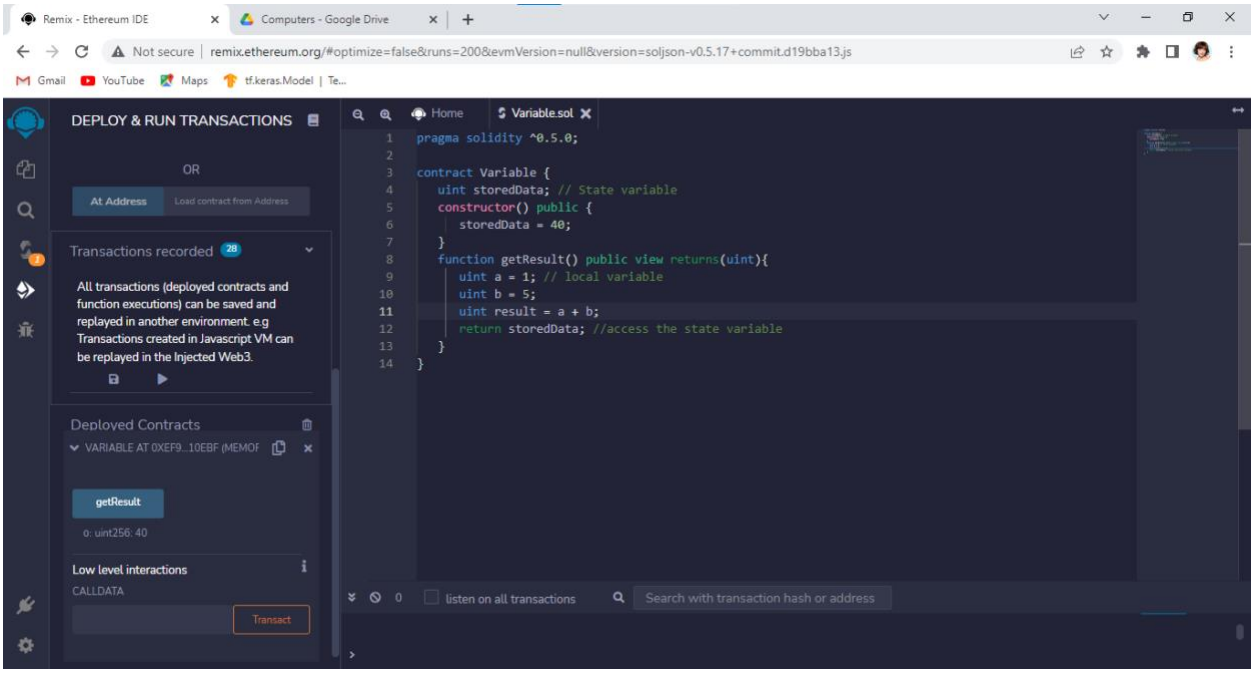
2a) Variable, Operators, Loops, Decision Making, Strings, Arrays, Enums, Structs, Mappings, Conversions, Ether Units, Special Variables. Variable

Variable

Code:



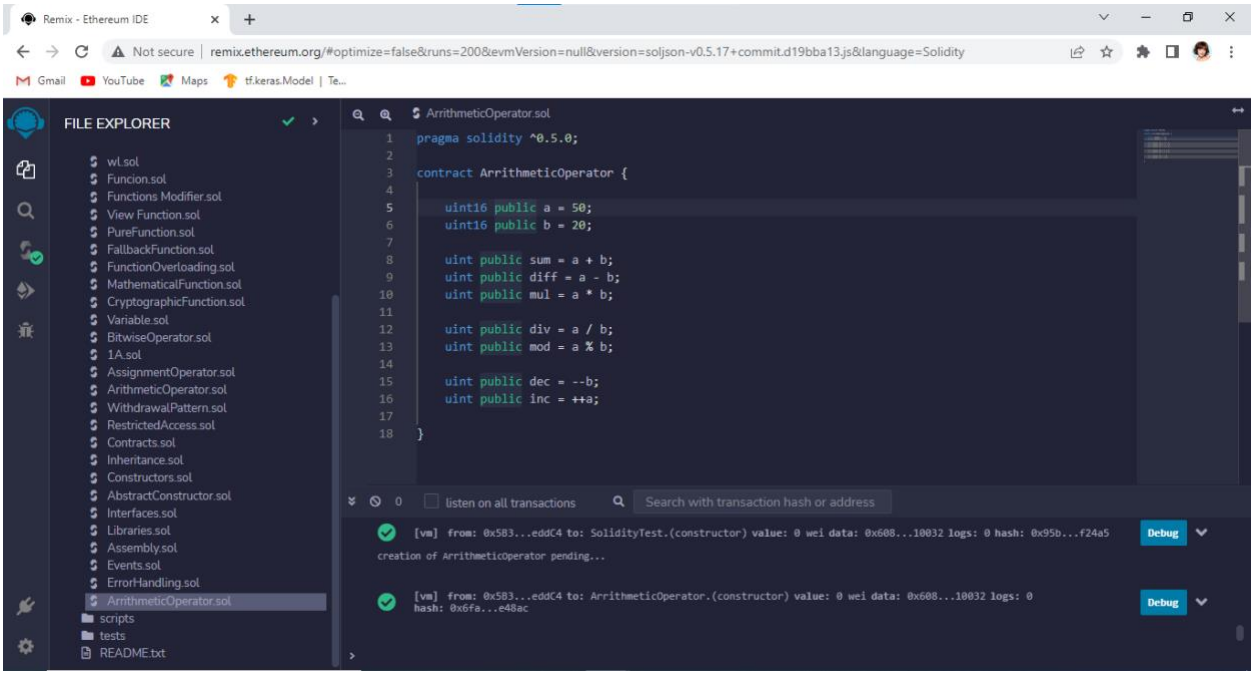
Output:


Remix - Ethereum IDE
Computers - Google Drive
Not secure | remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.5.17+commit.d19bba13.js
Gmail
YouTube
Maps
tf.keras.Model | Te...
DEPLOY & RUN TRANSACTIONS
OR
At Address
Load contract from Address
Transactions recorded 28
All transactions (deployed contracts and function executions) can be saved and replayed in another environment. e.g Transactions created in Javascript VM can be replayed in the Injected Web3.
Deployed Contracts
VARIABLE AT 0XEF9...10EBF (MEMOF
getResult
0: uint256: 40
Low level interactions
CALLDATA
Transact
Home
Variable.sol
pragma solidity ^0.5.0;
contract Variable {
uint storedData; // State variable
constructor() public {
storedData = 40;
}
function getResult() public view returns(uint){
uint a = 1; // local variable
uint b = 5;
uint result = a + b;
return storedData; //access the state variable
}
}
listen on all transactions
Search with transaction hash or address

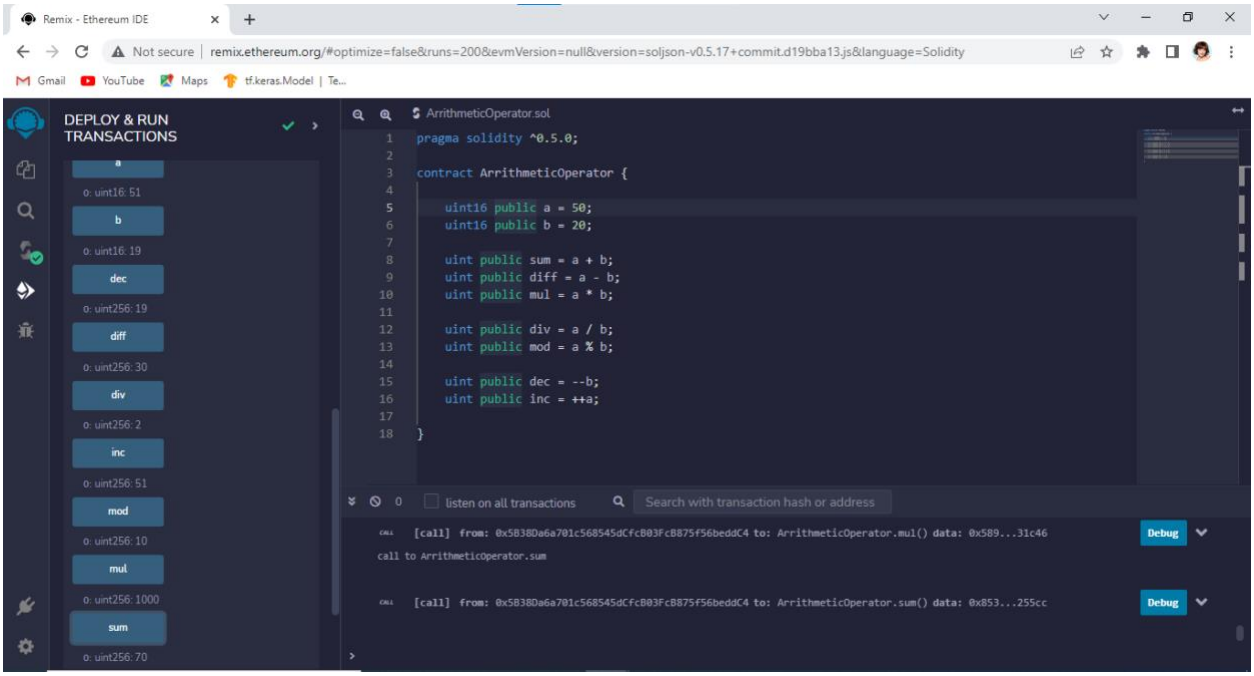
Operators:

Arithmetic Operator

Code:

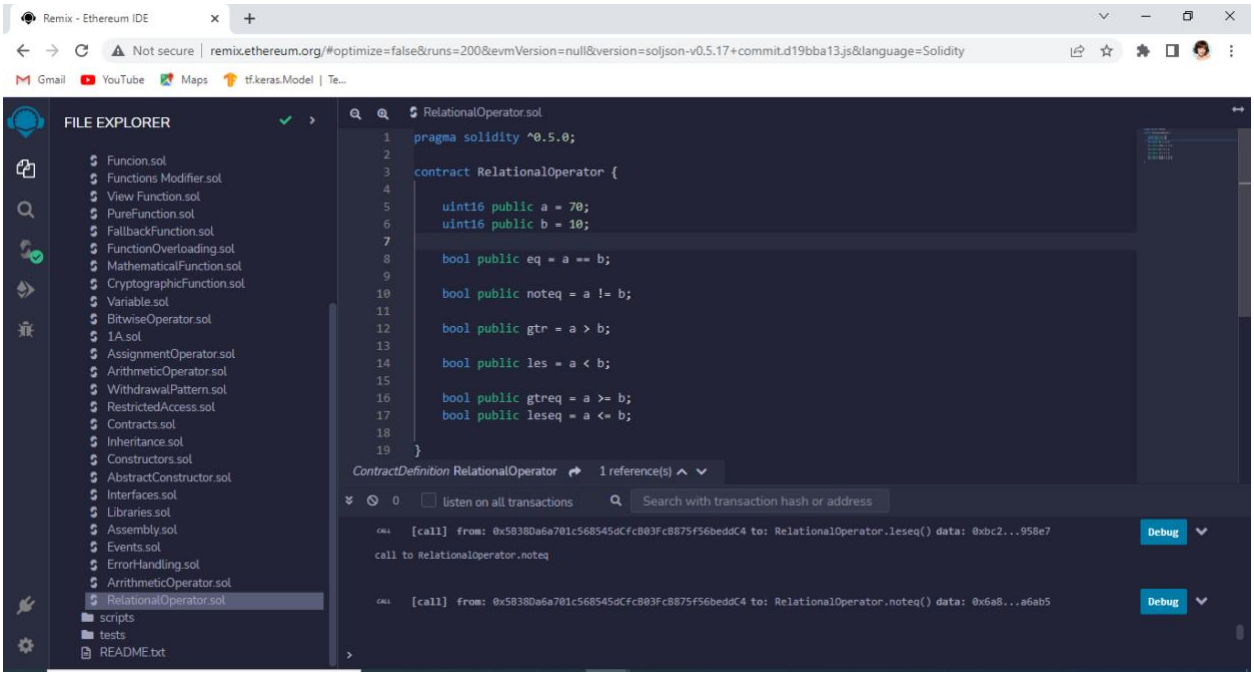


Output:

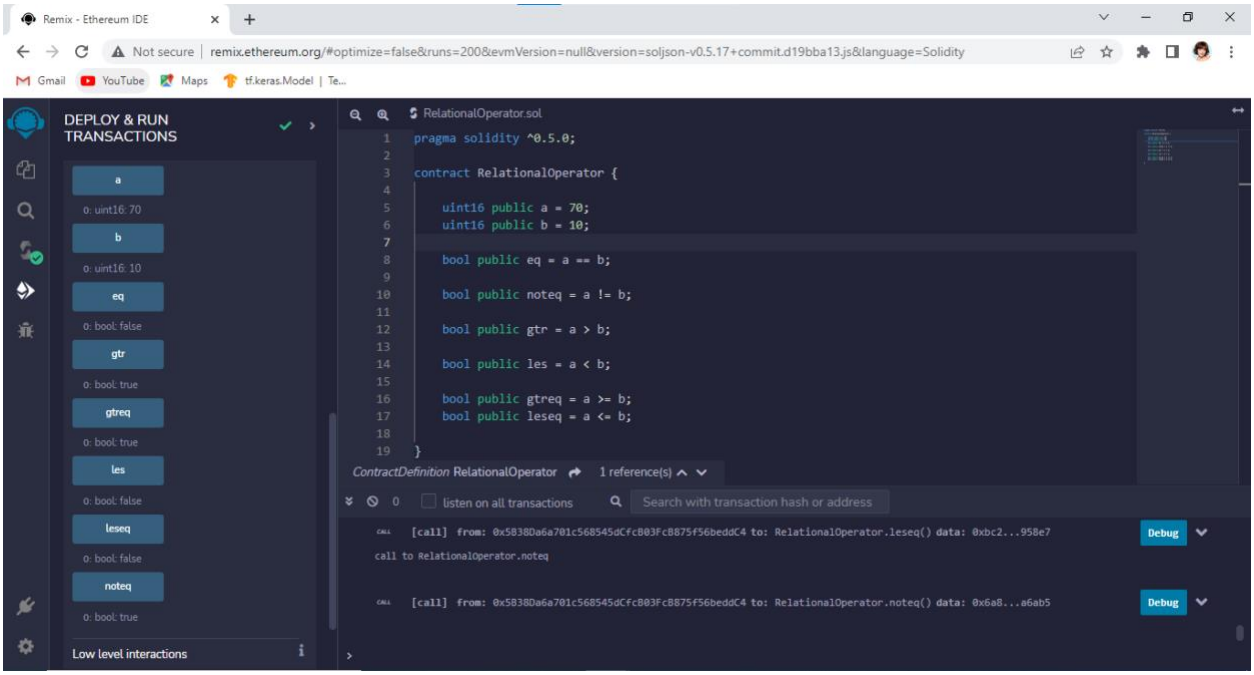


Relational Operator

Code:

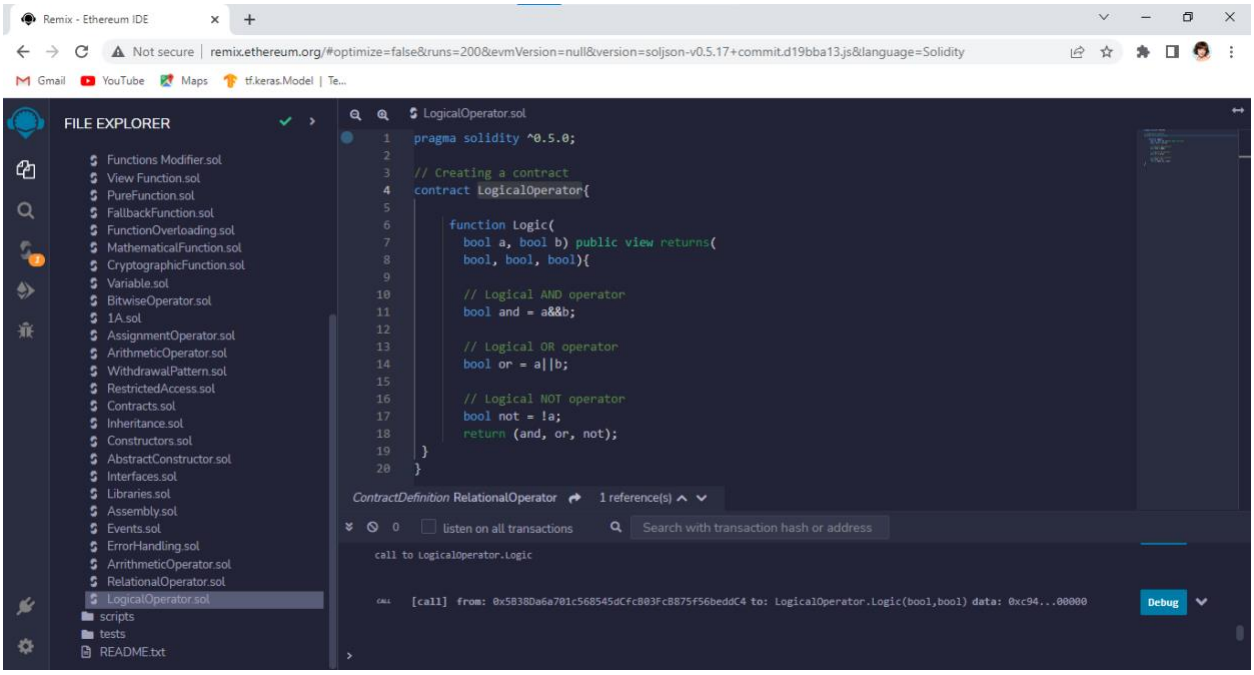


Output:

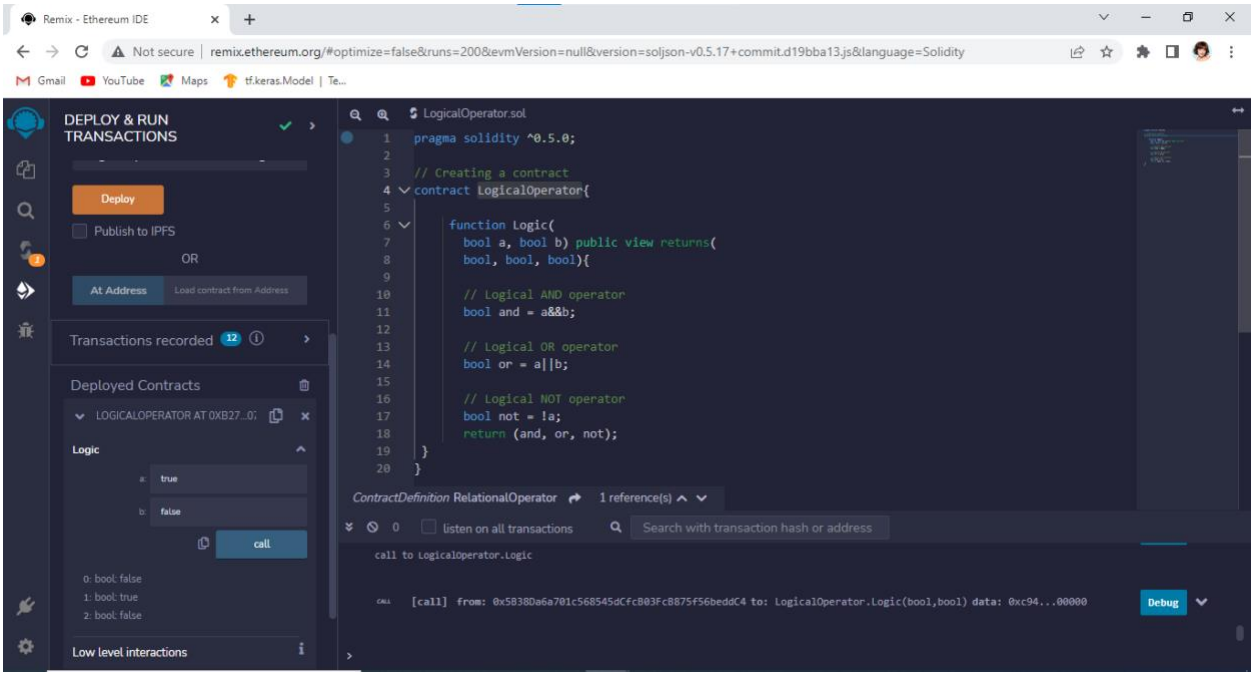


Logical Operator

Code:

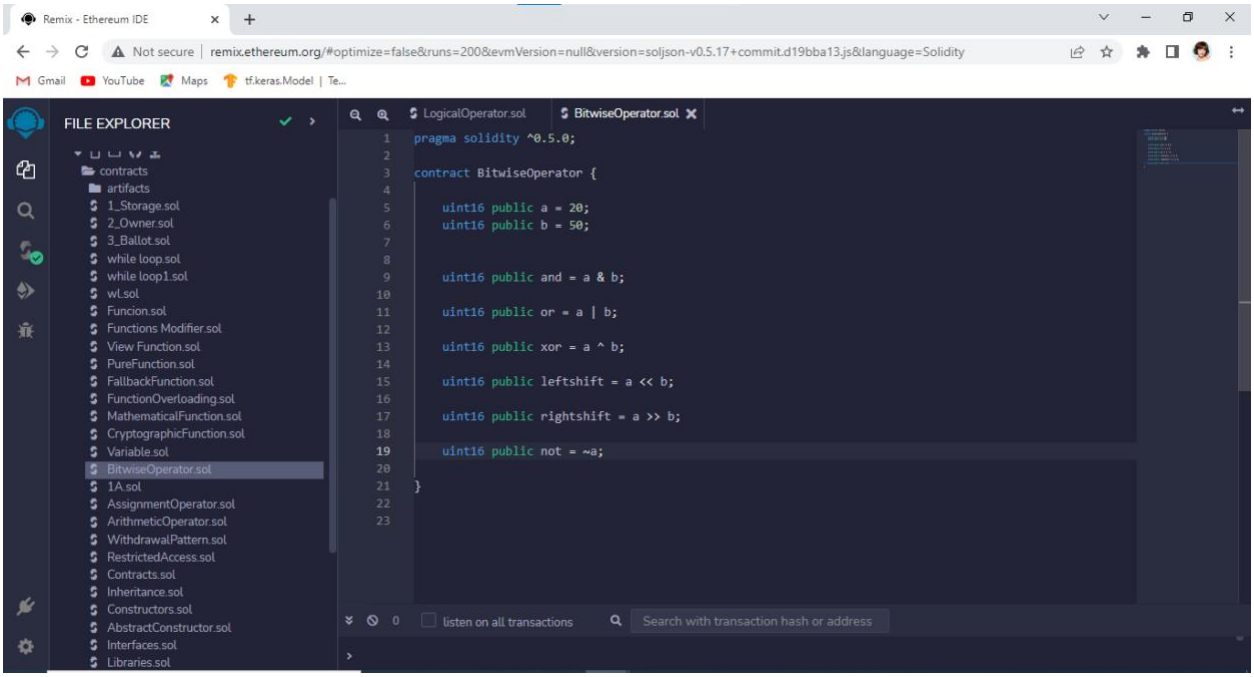


Output:

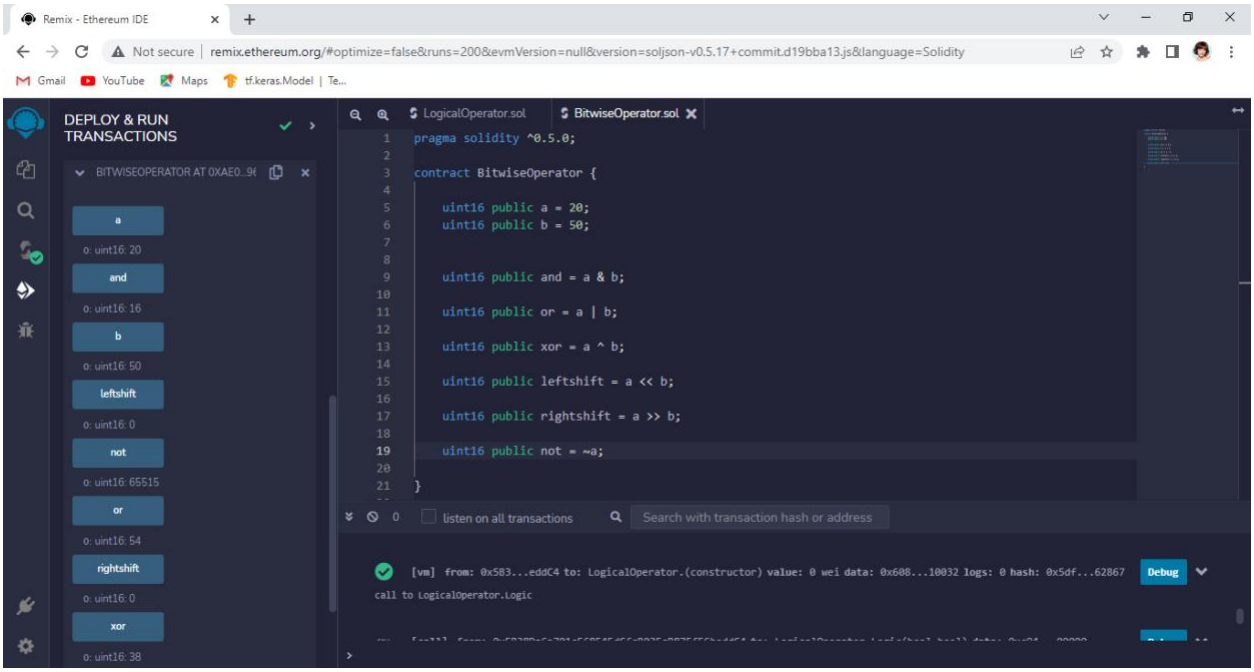


Bitwise Operator

Code:

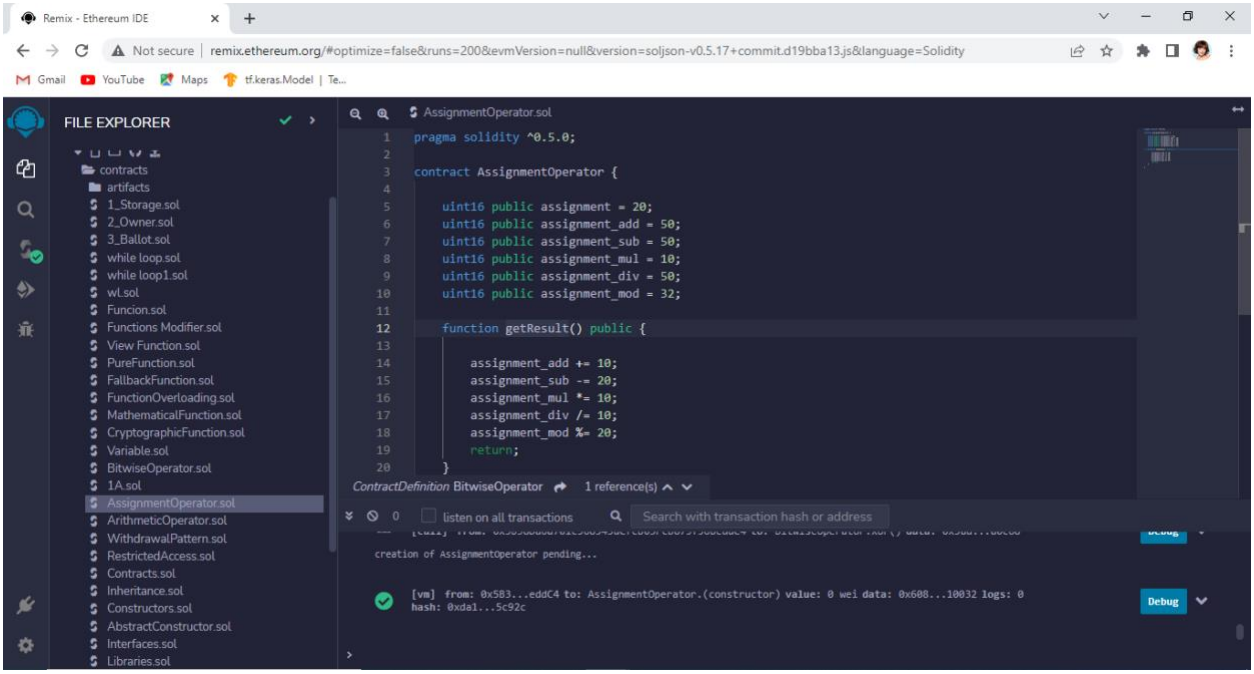


Output:

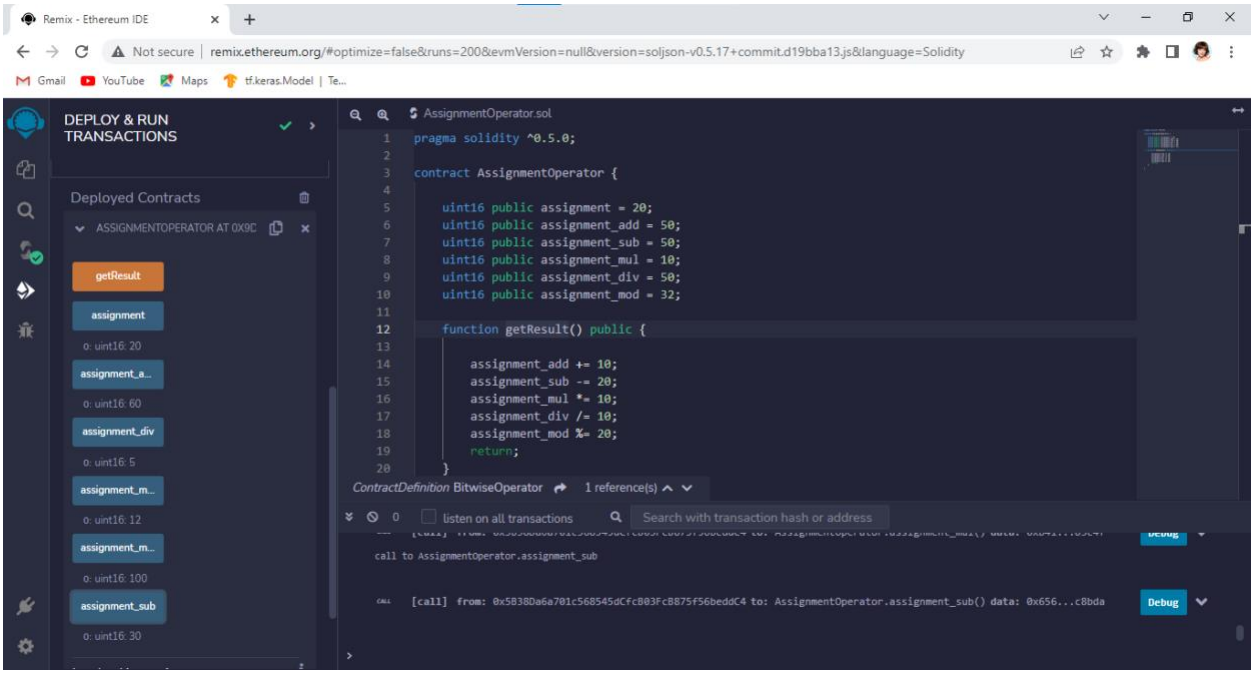


Assignment Operator

Code:



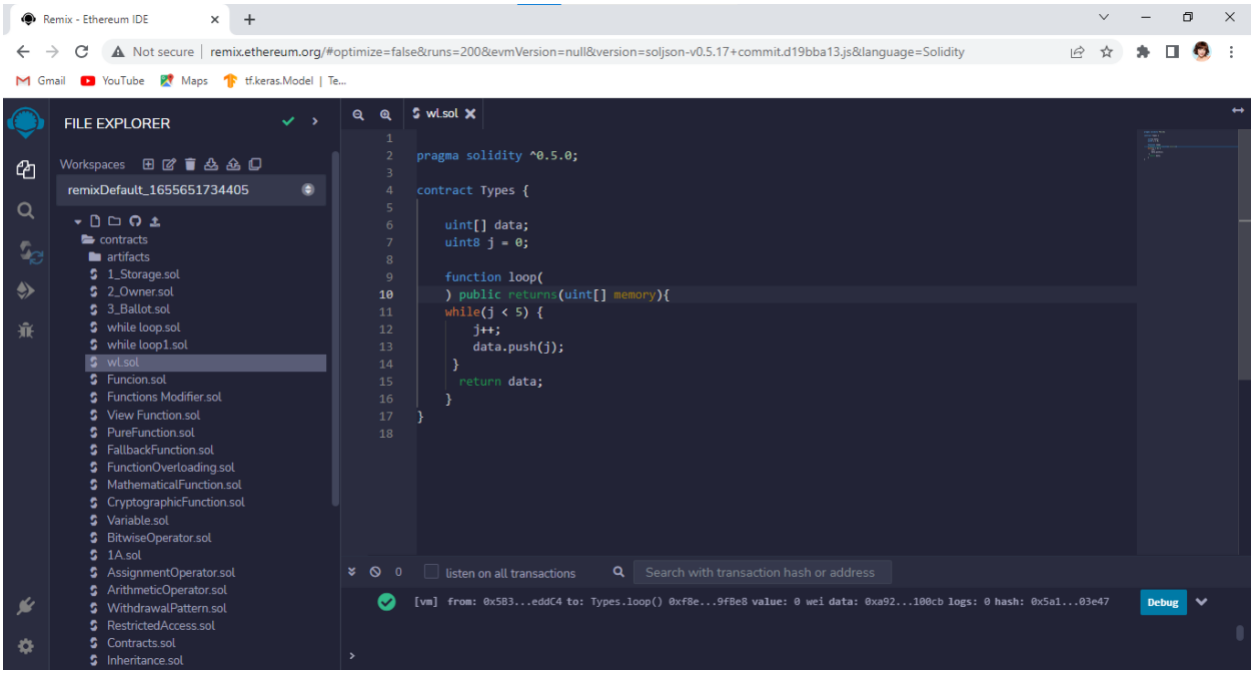
Output:



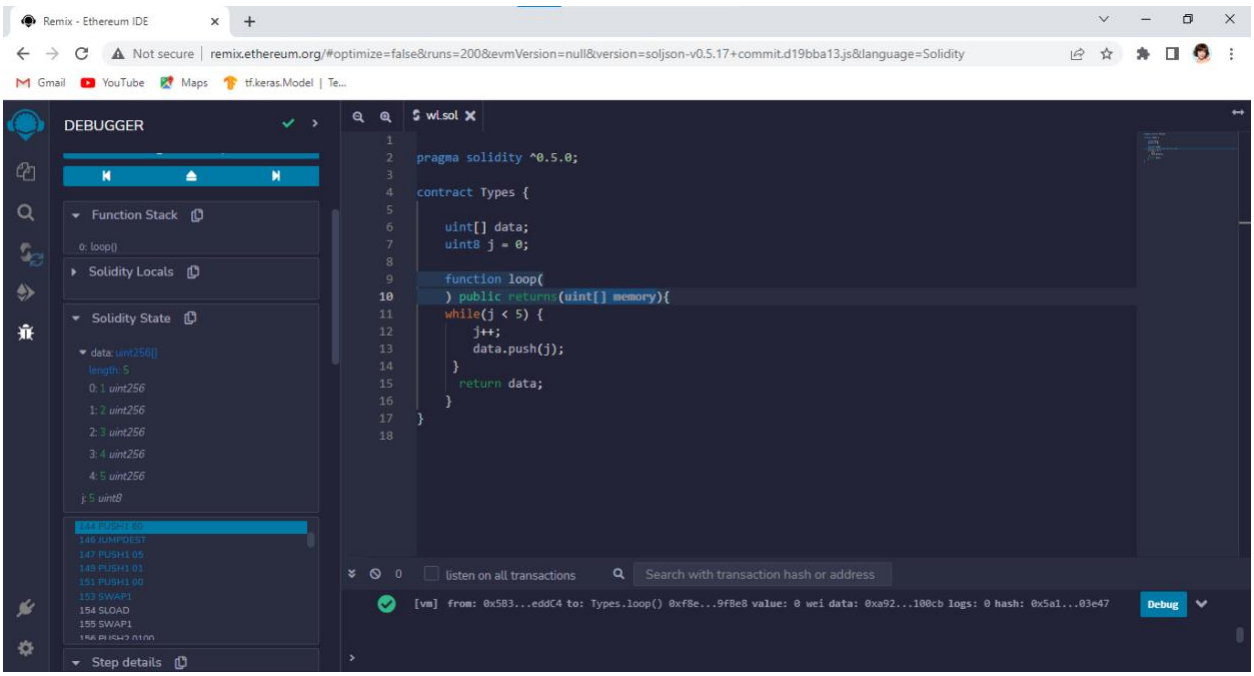
Loops

While Loop

Code:

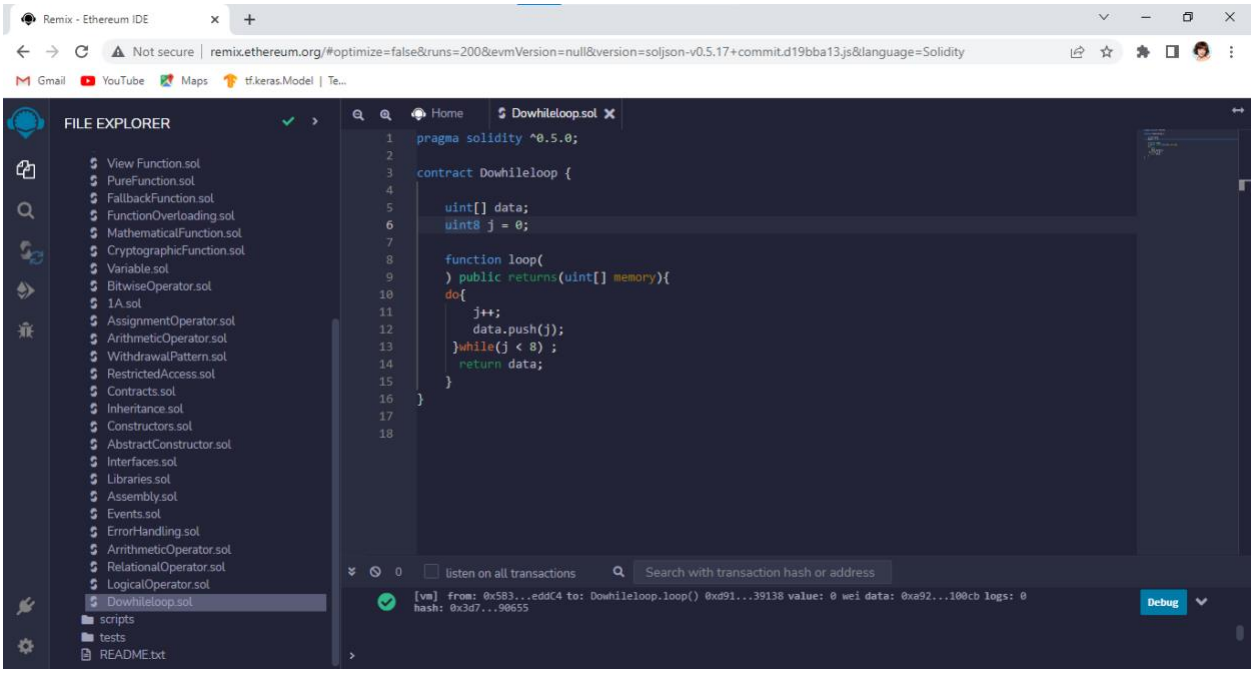


Output:

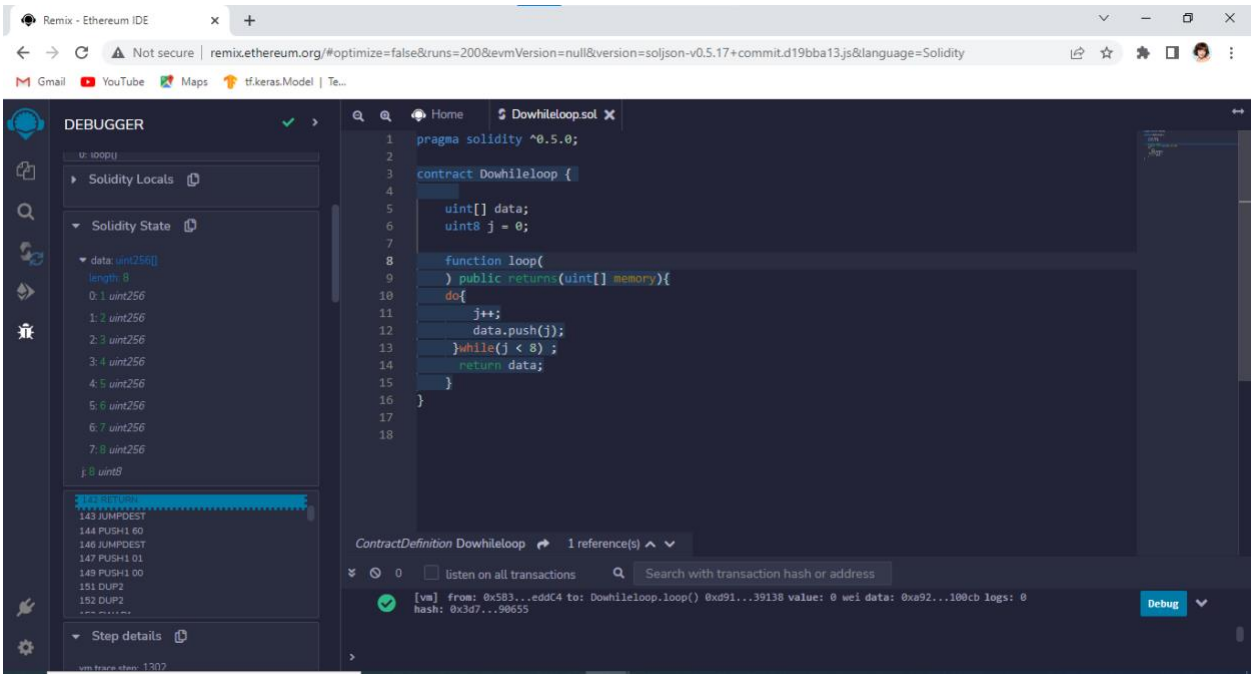


Dowhile loop

Code:

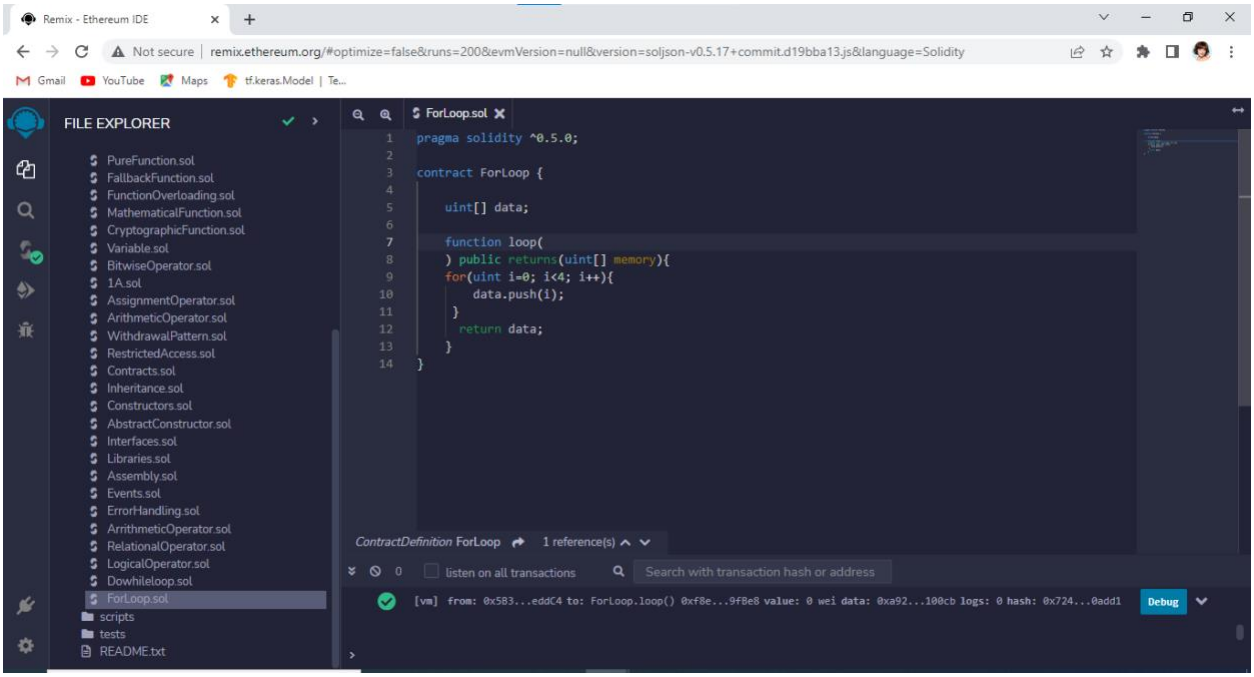


Output:

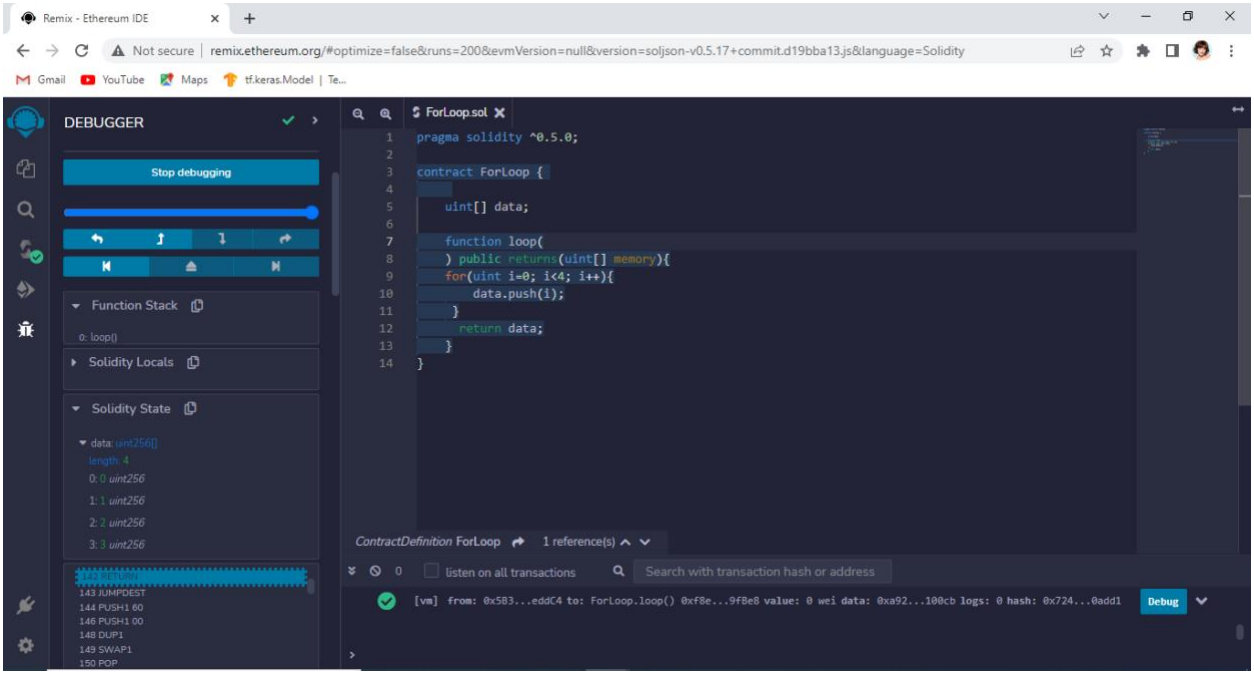


For loop

Code:



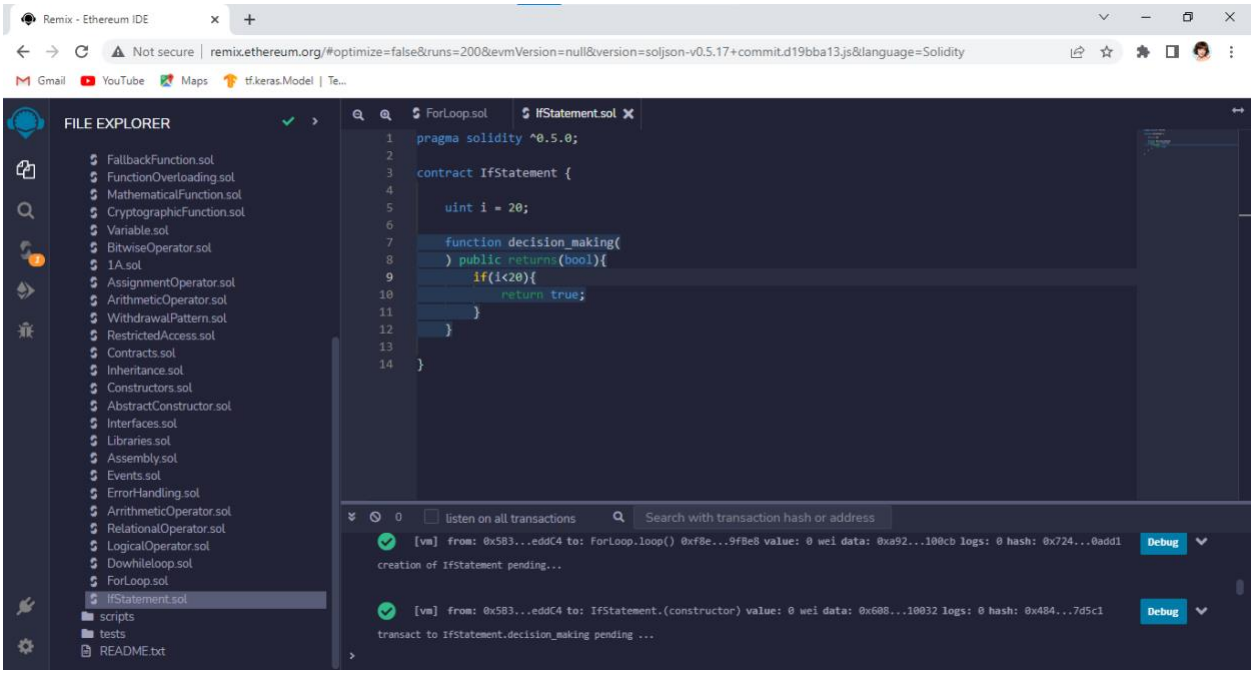
Output:



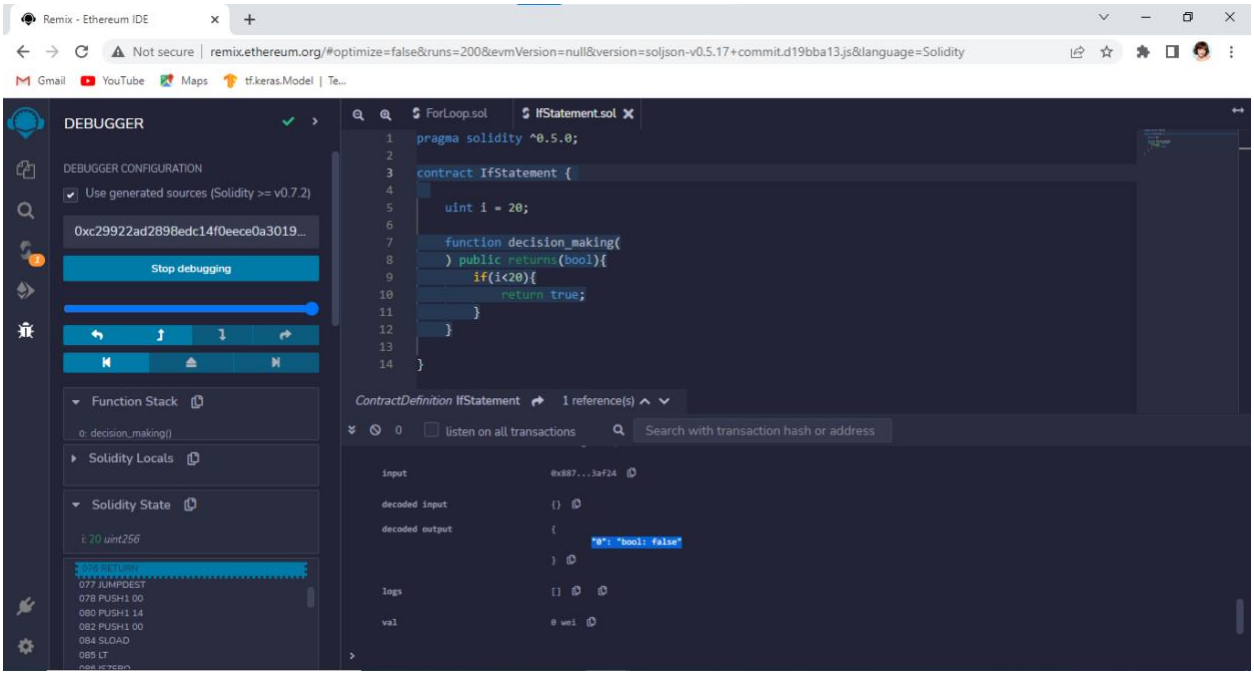
Decision making

If statement

Code:

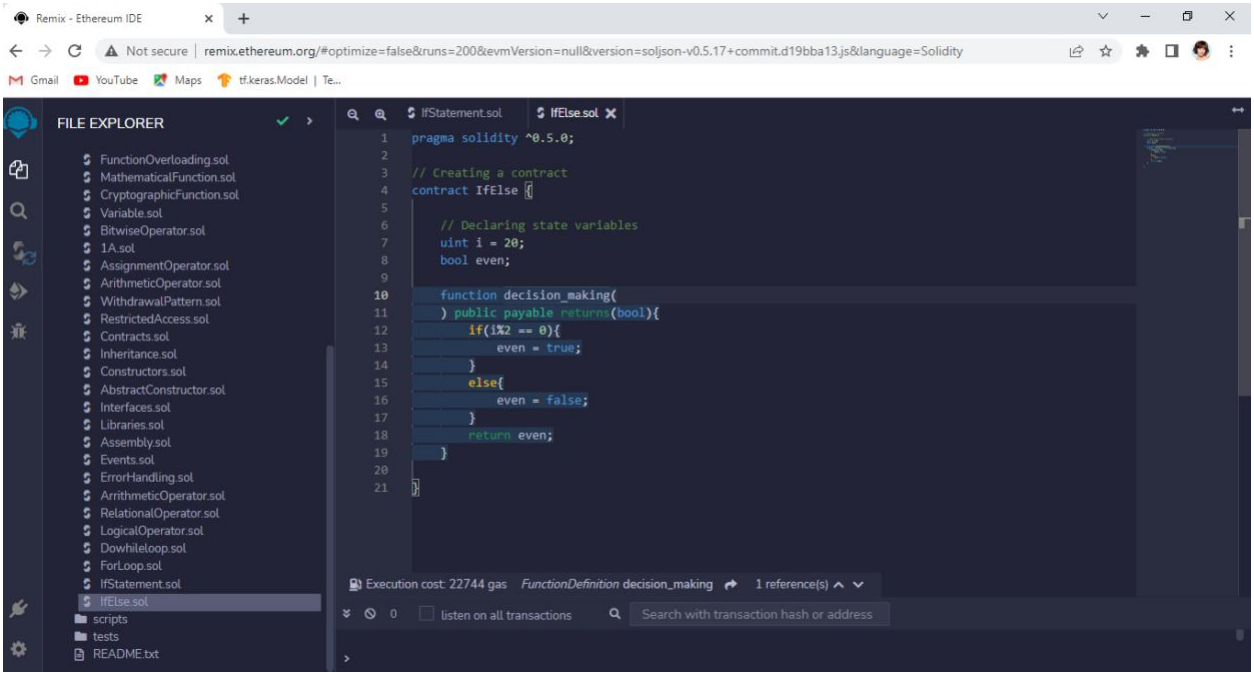


Output:

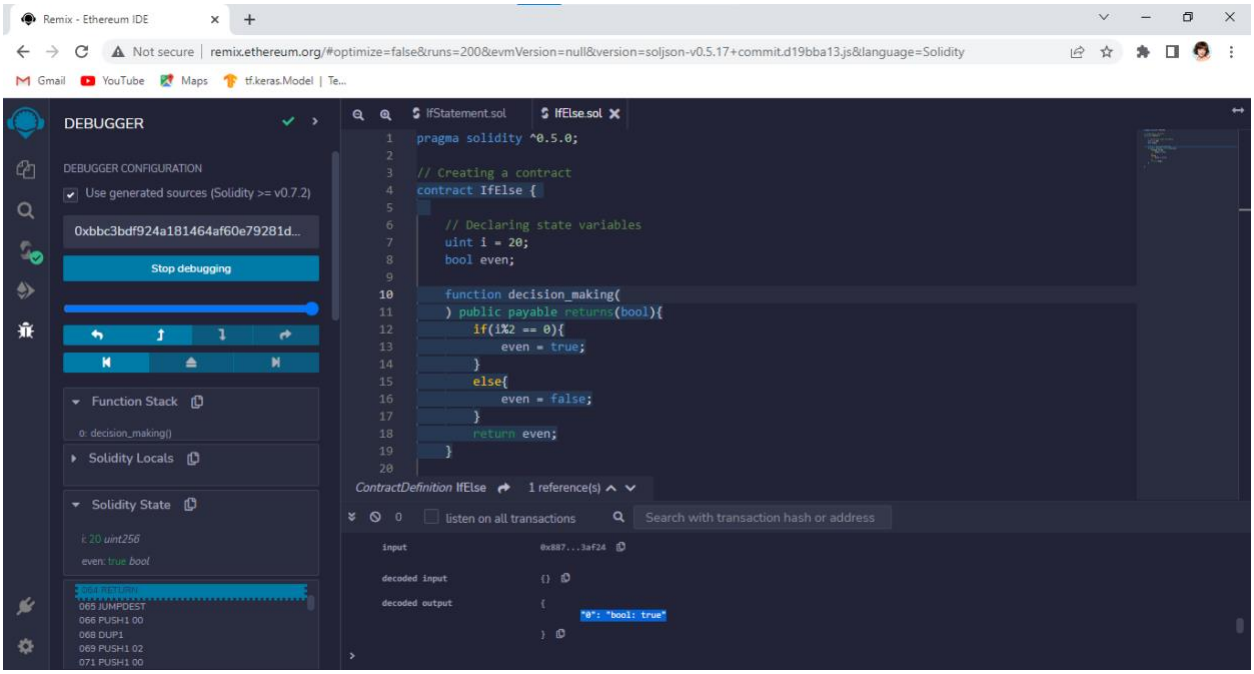


If…else statement

Code:

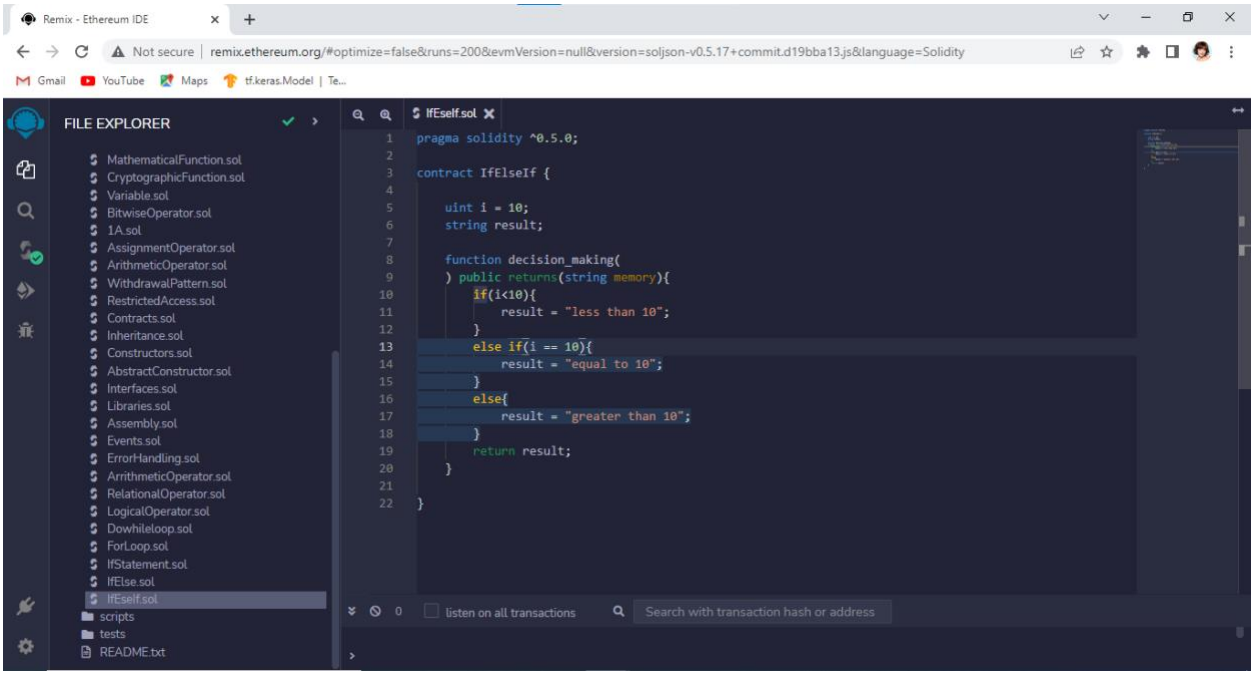


Output:

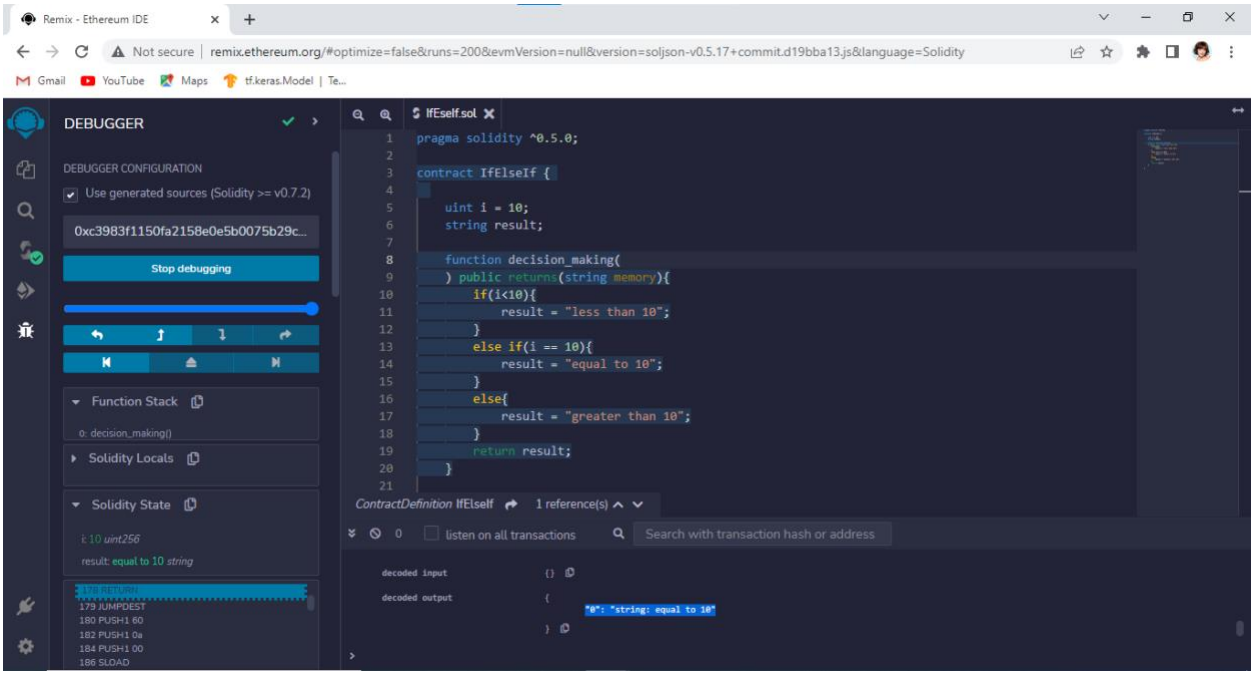


If…else if…else statement

Code:

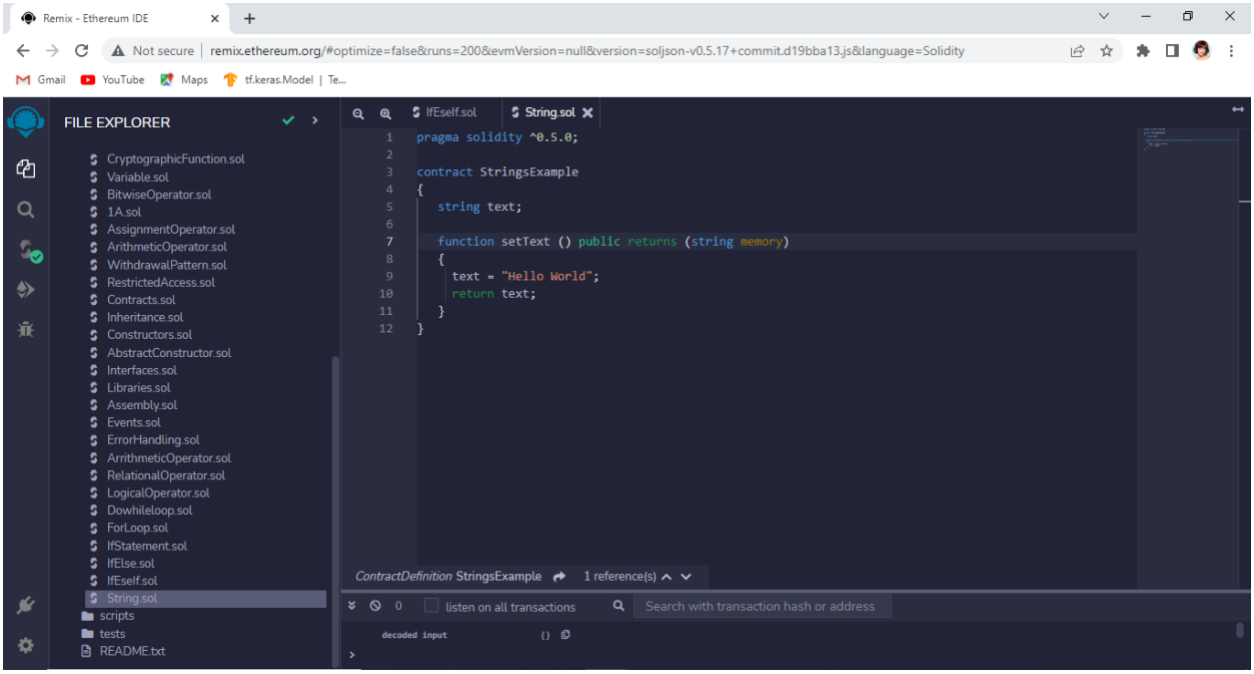


Output:

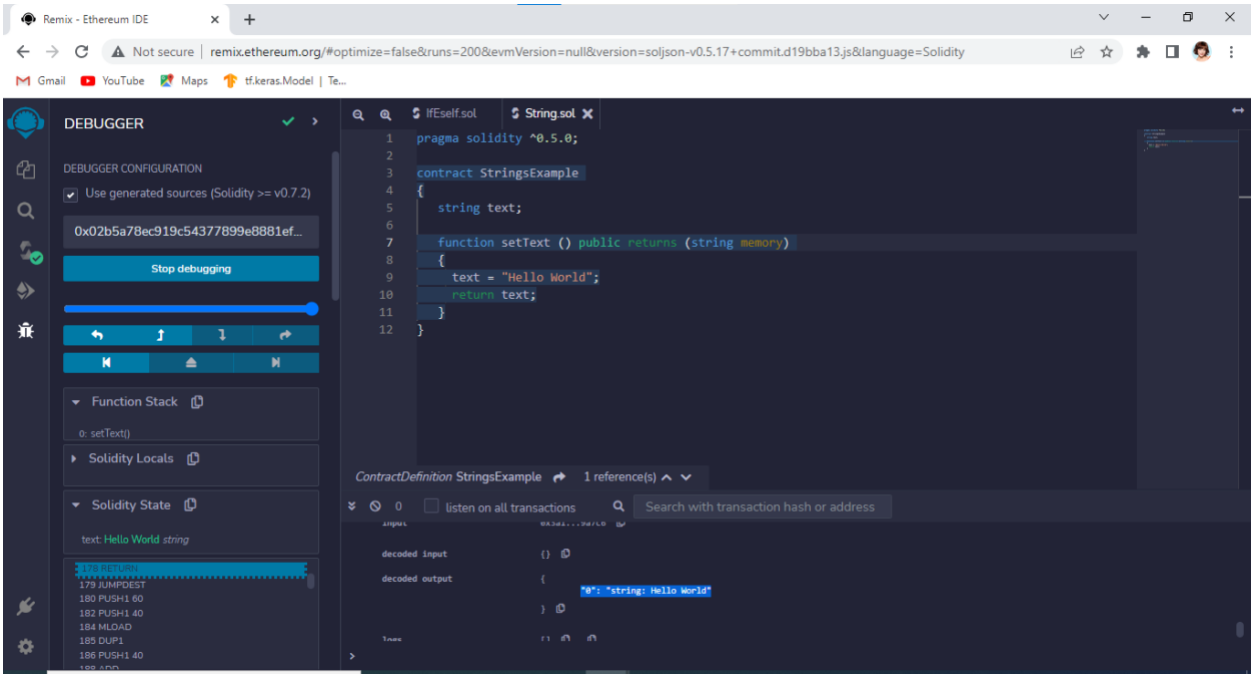


String

Code:

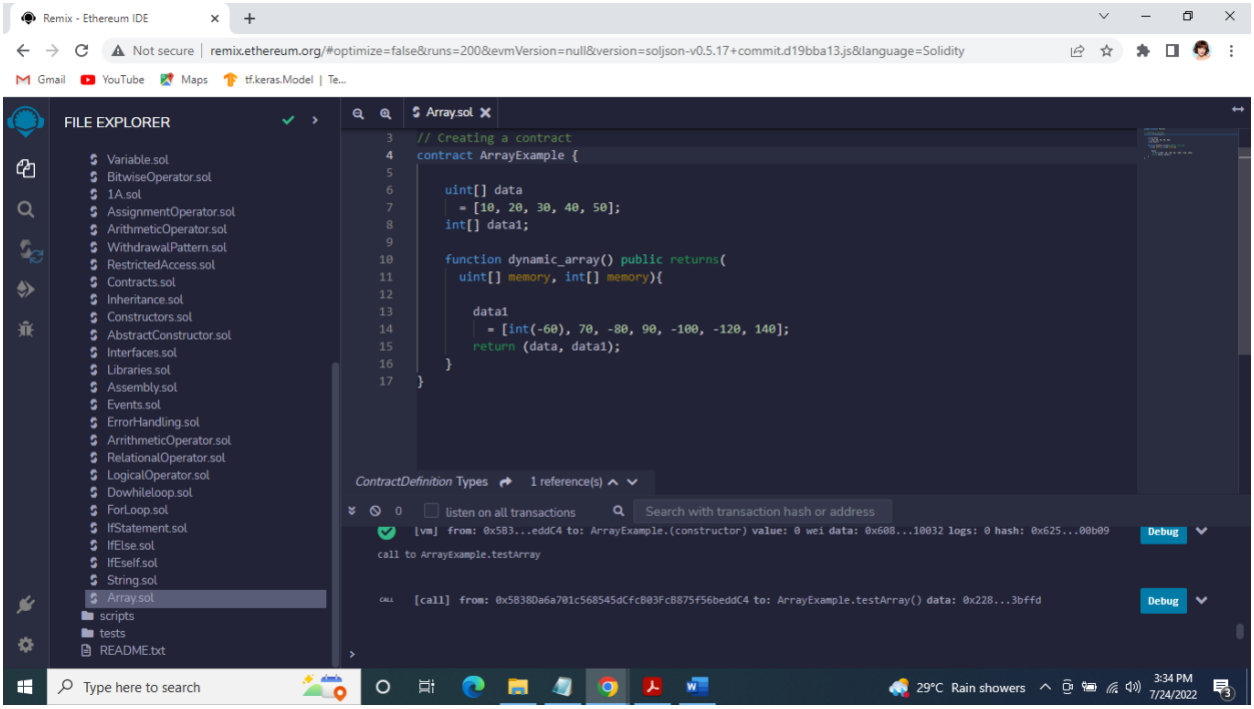


Output:

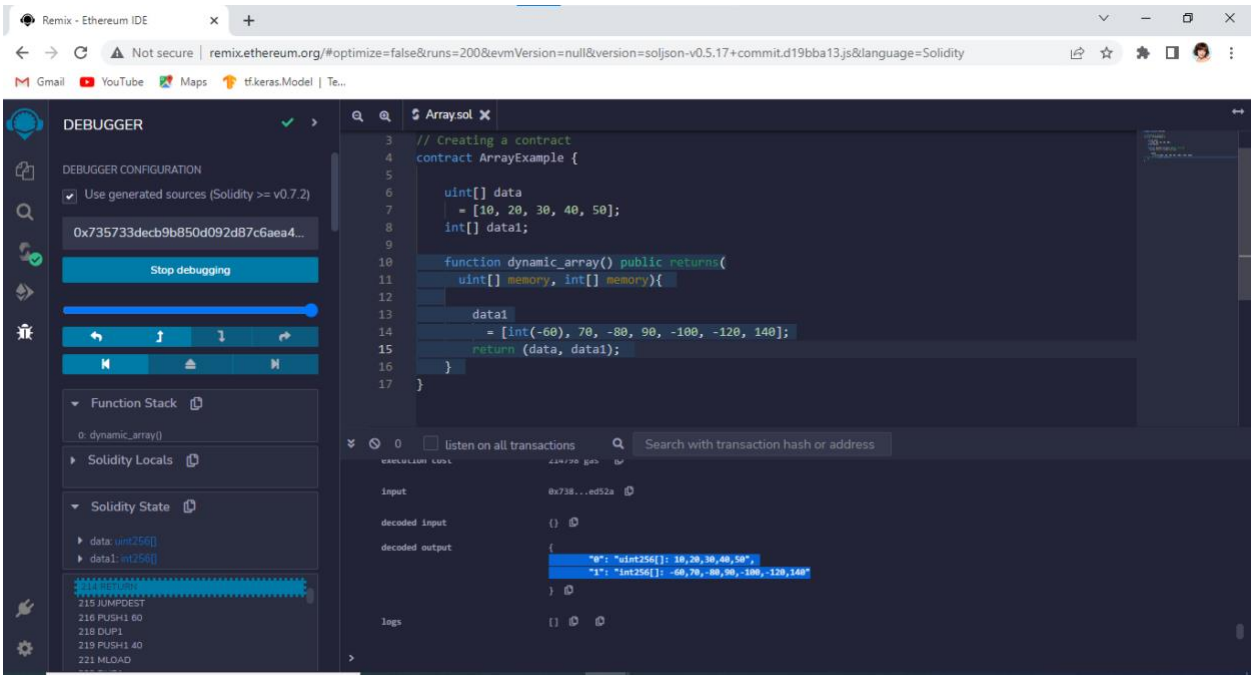


Array

Code:

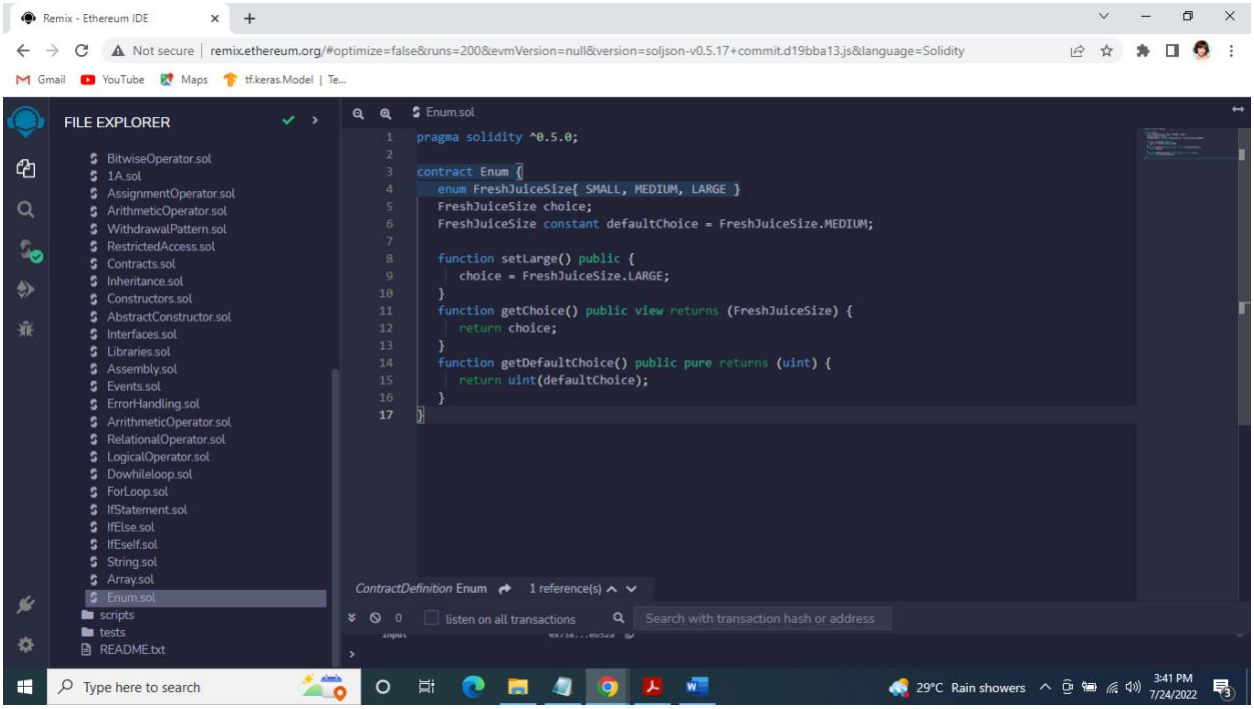


Output:

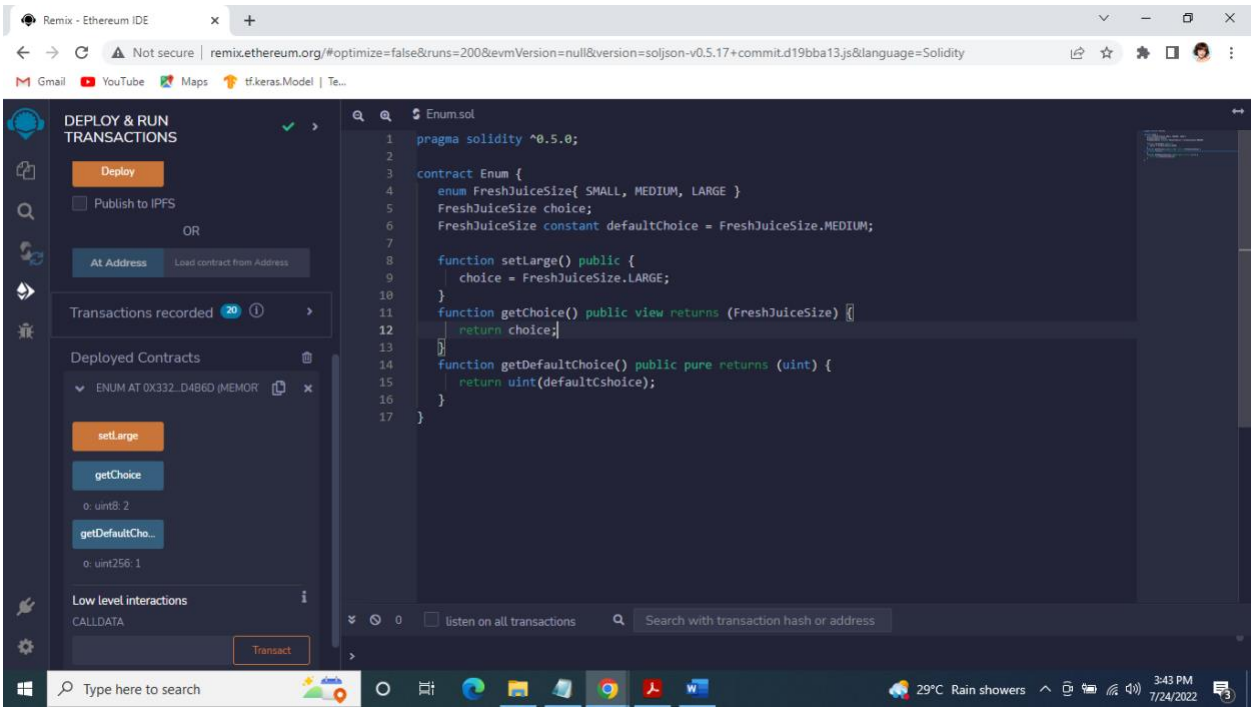


Enums

Code:

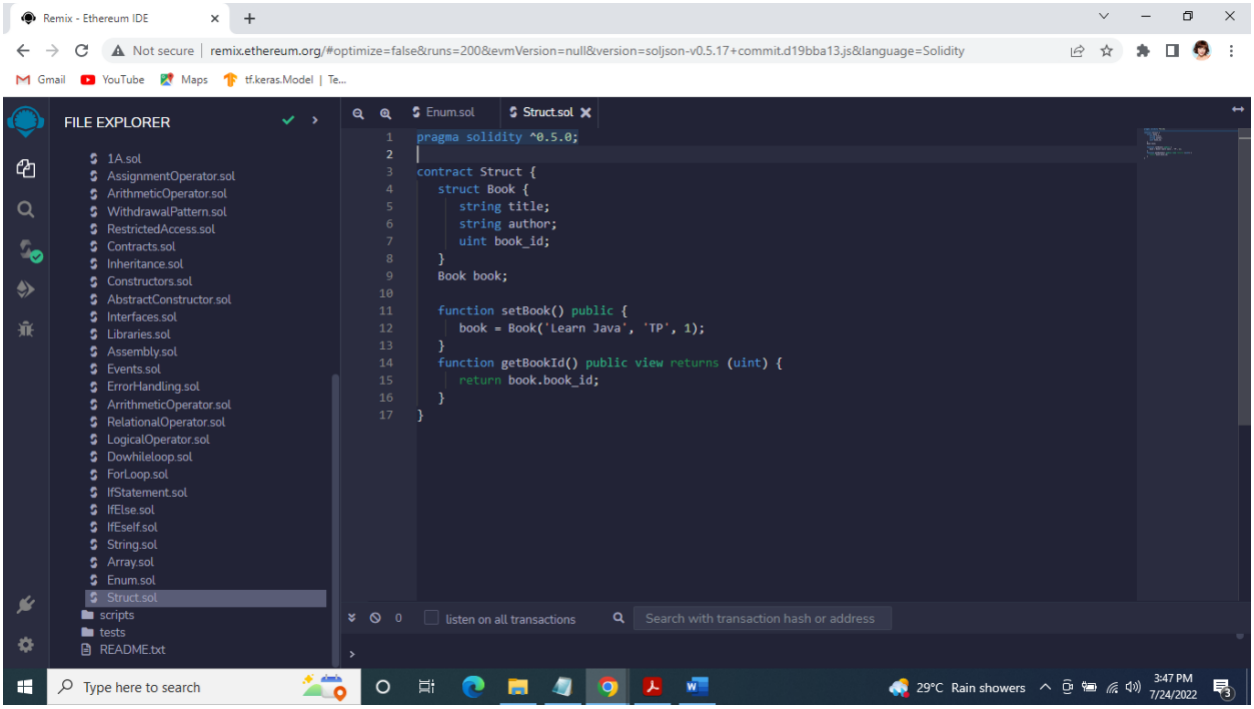


Output:

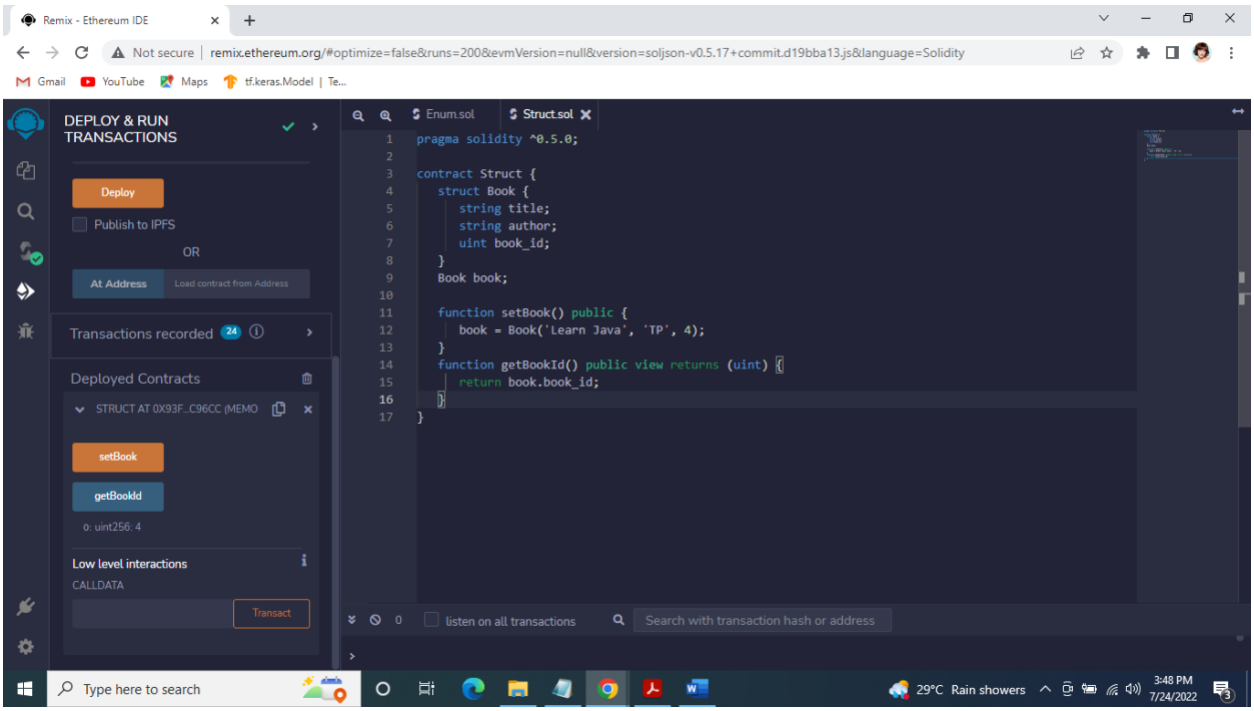


Struct

Code:

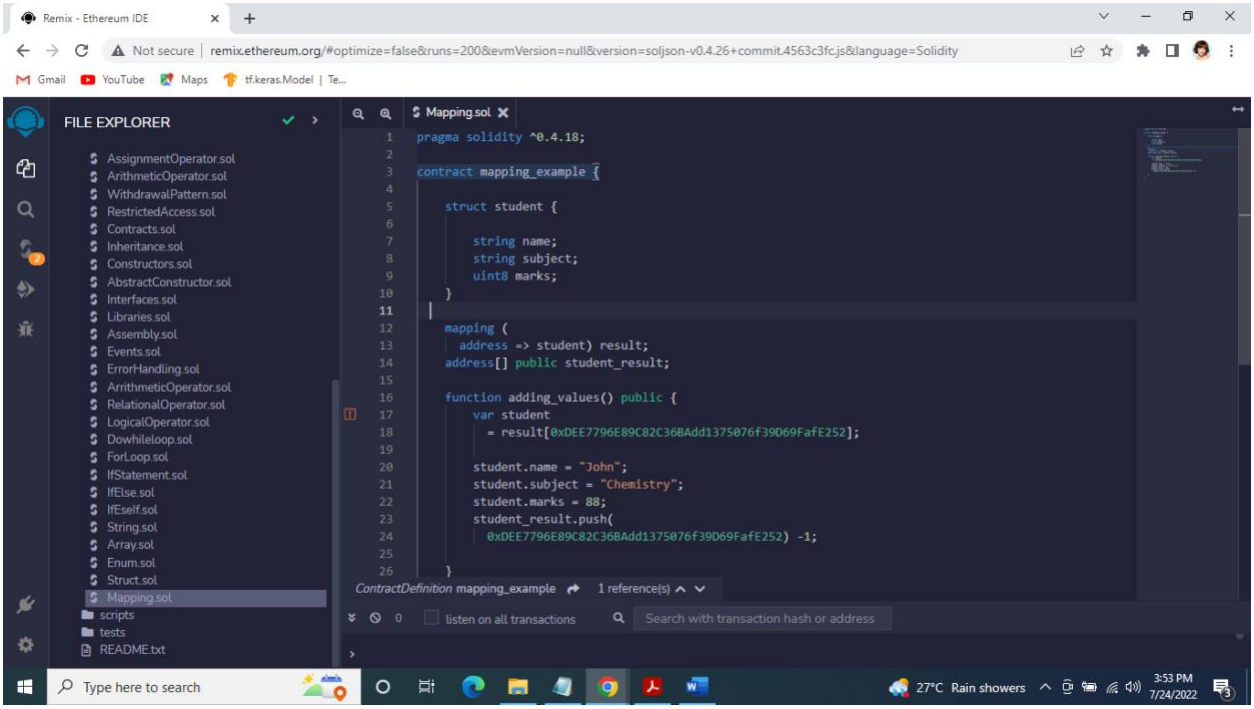


Output:

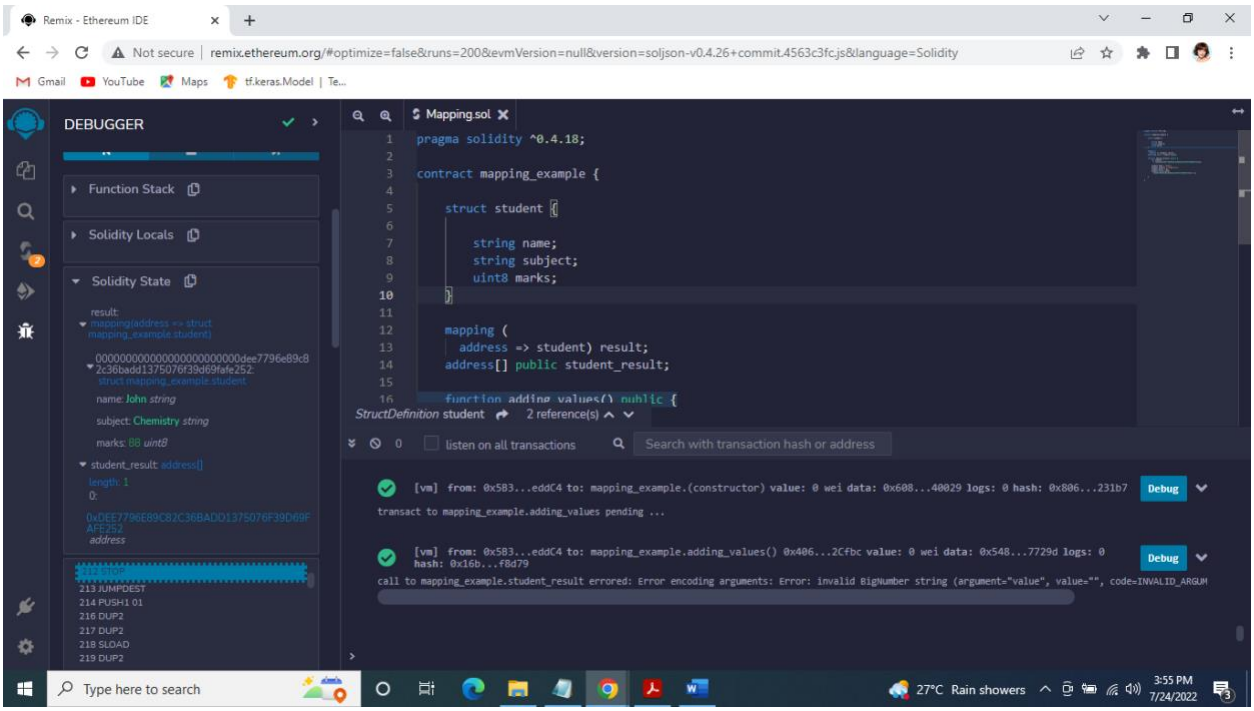


Mapping

Code:

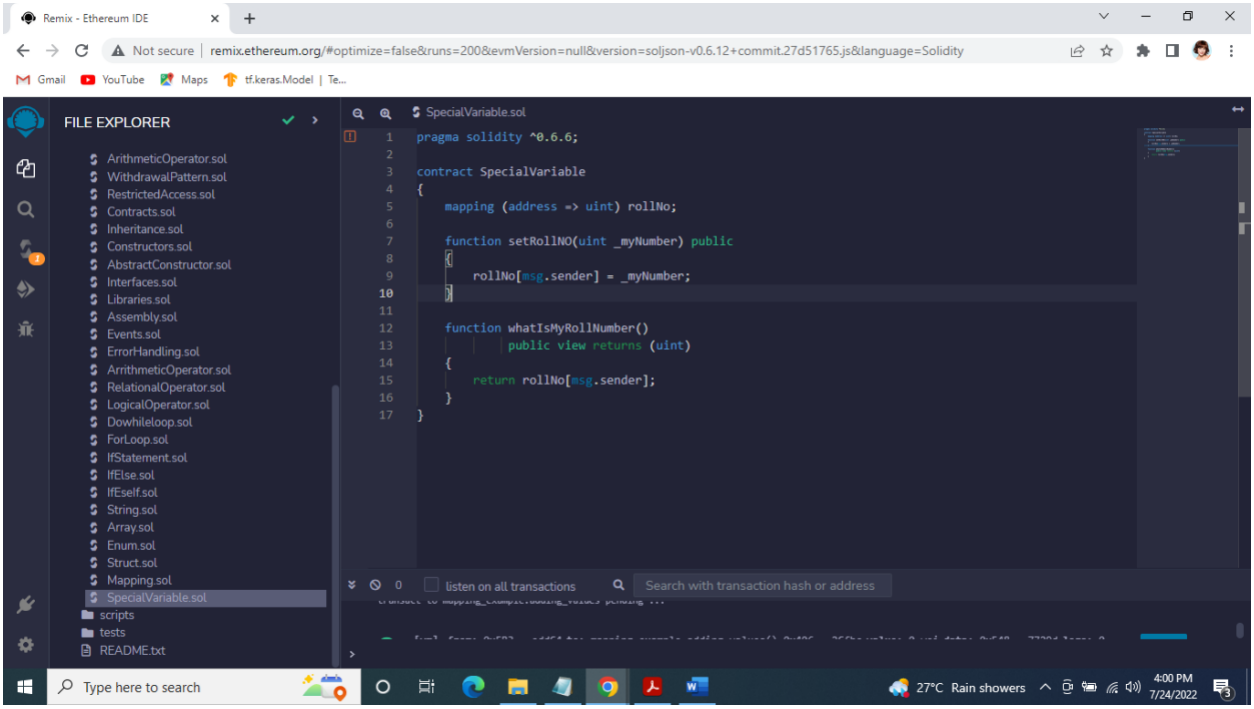


Output:

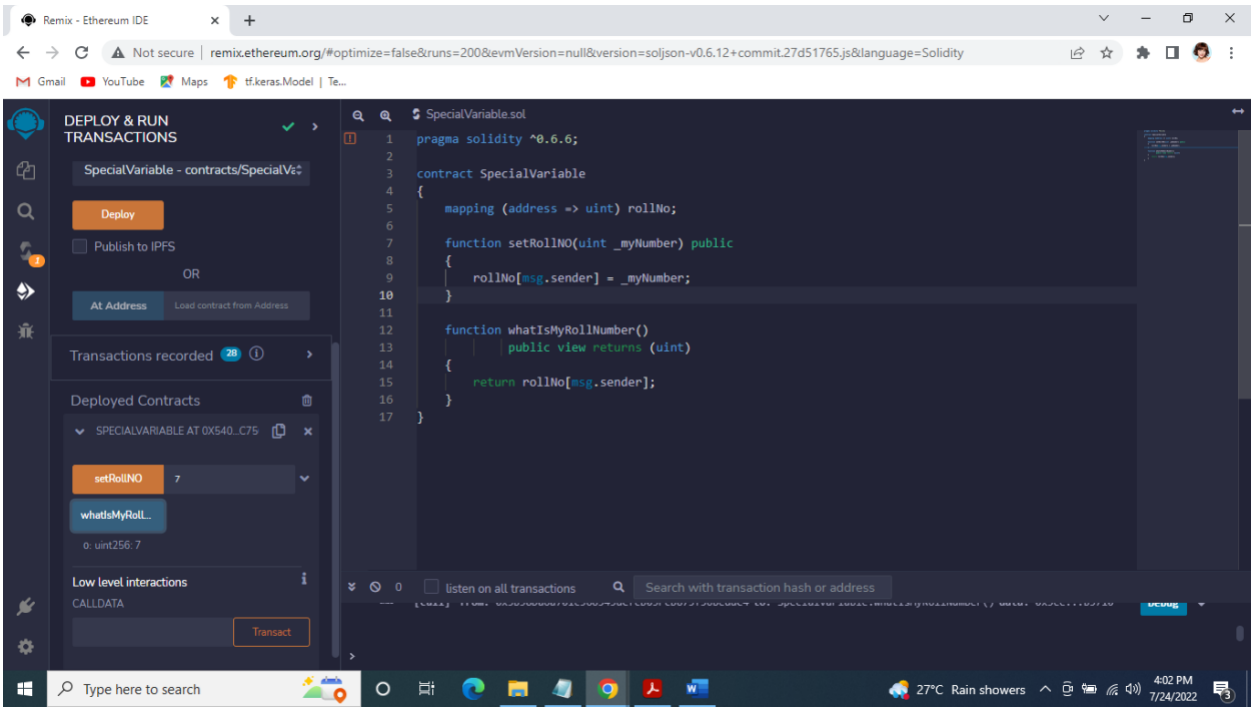


Special Variable

Code:



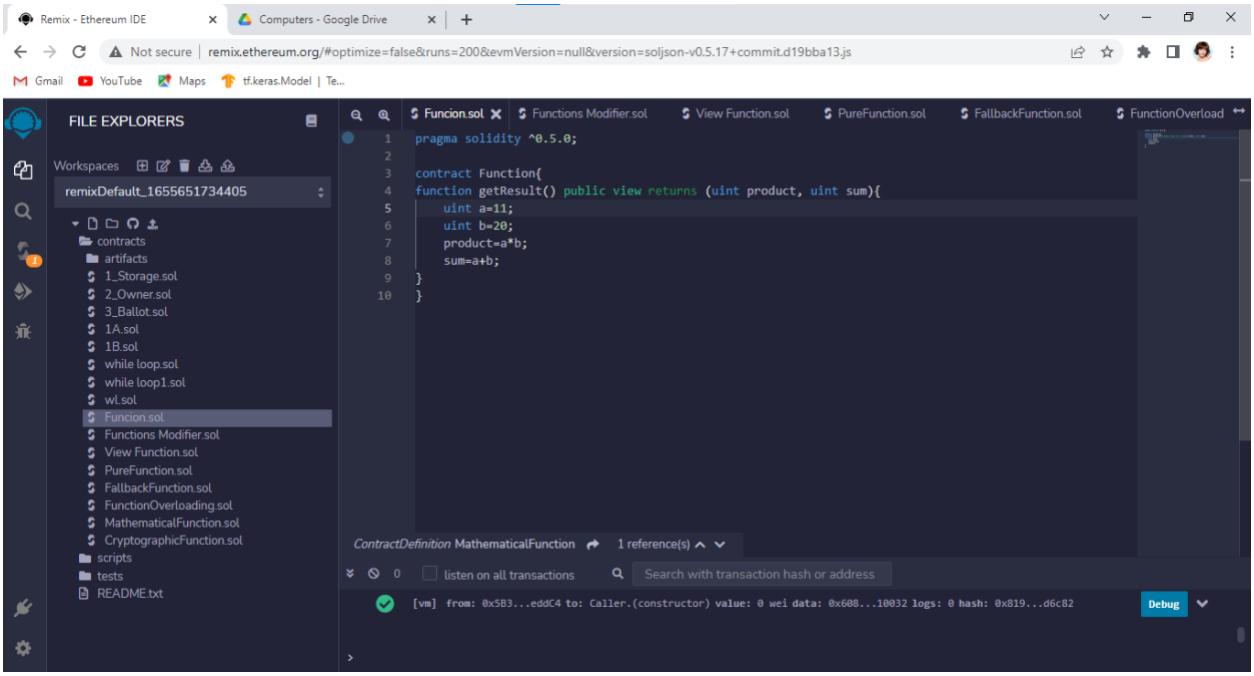
Output:



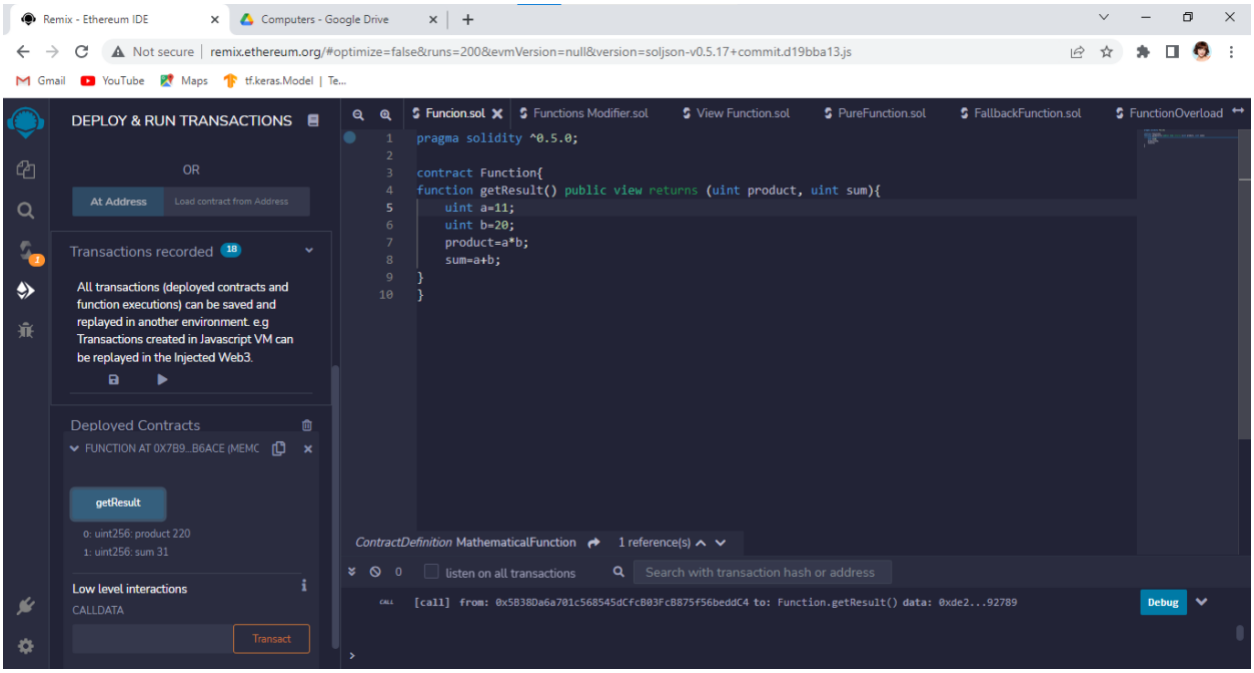
2b) Functions, Function Modifiers, View functions, Pure Functions, Fallback Function, Function Overloading, Mathematical functions, Cryptographic functions.

Function:

Code:

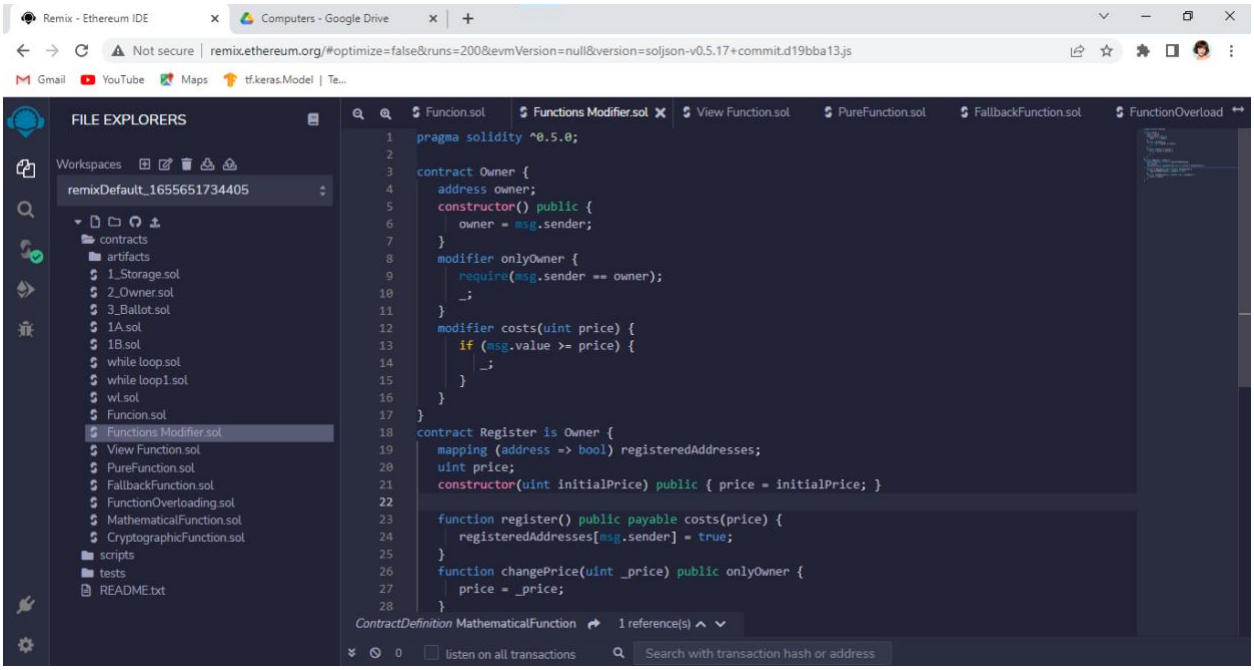


Output:

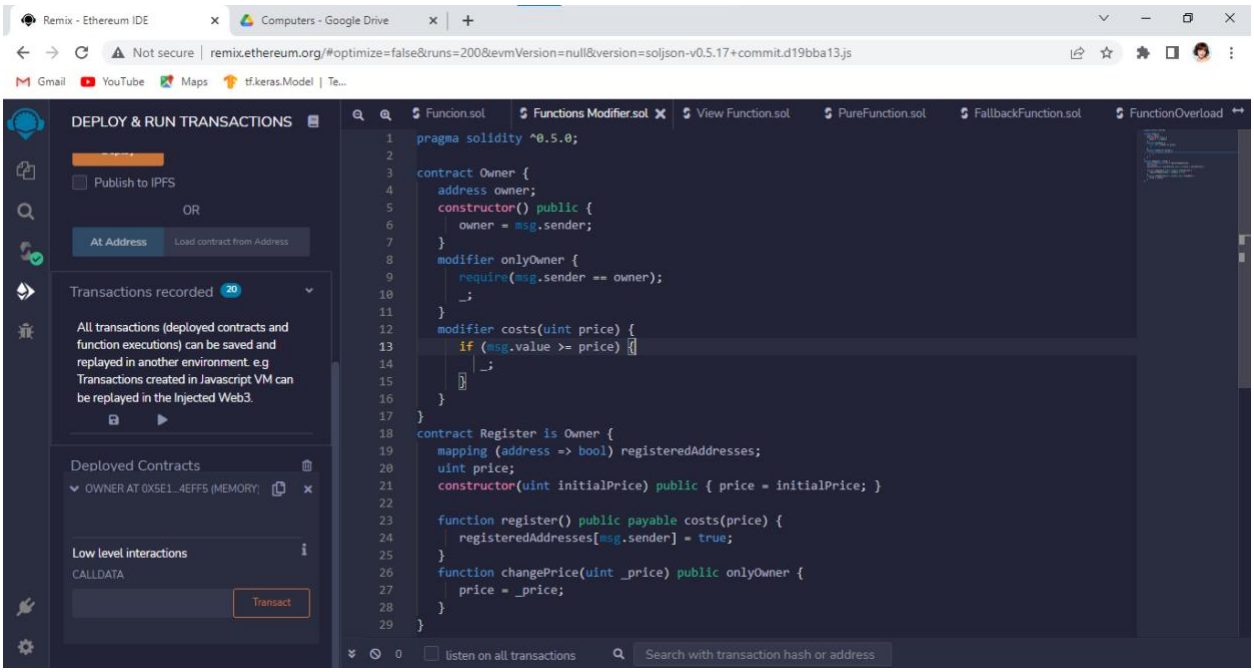


Functions Modifiers

Code:

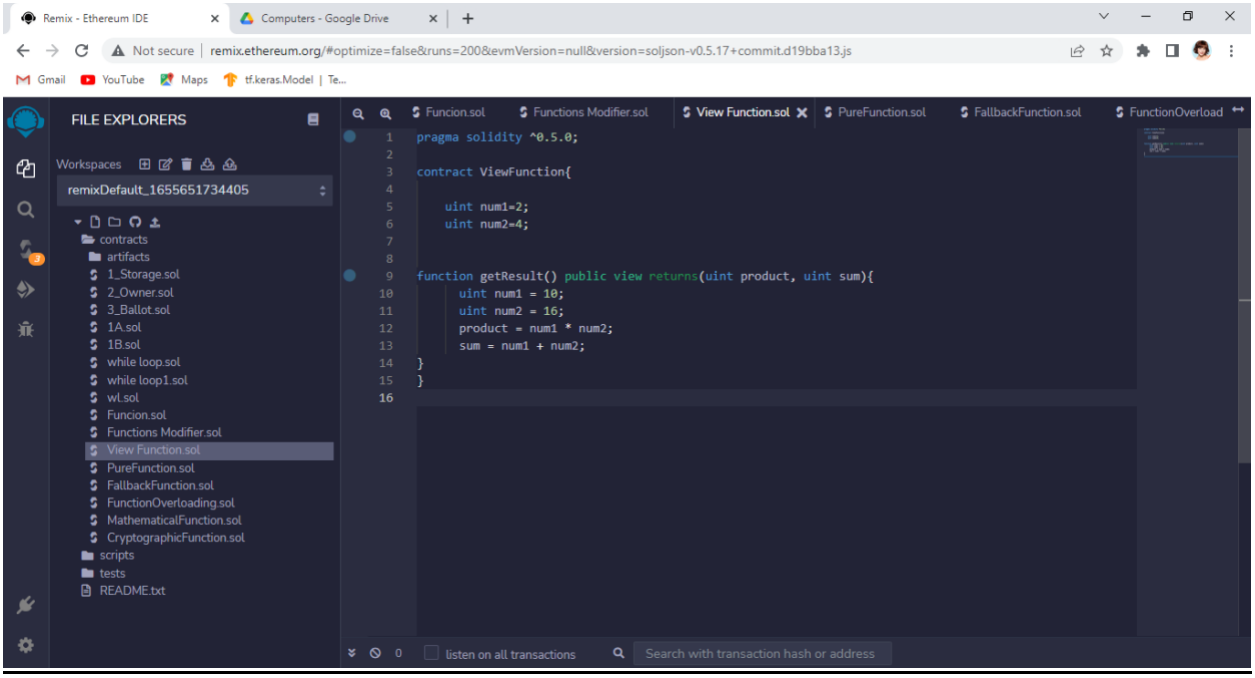


Output:

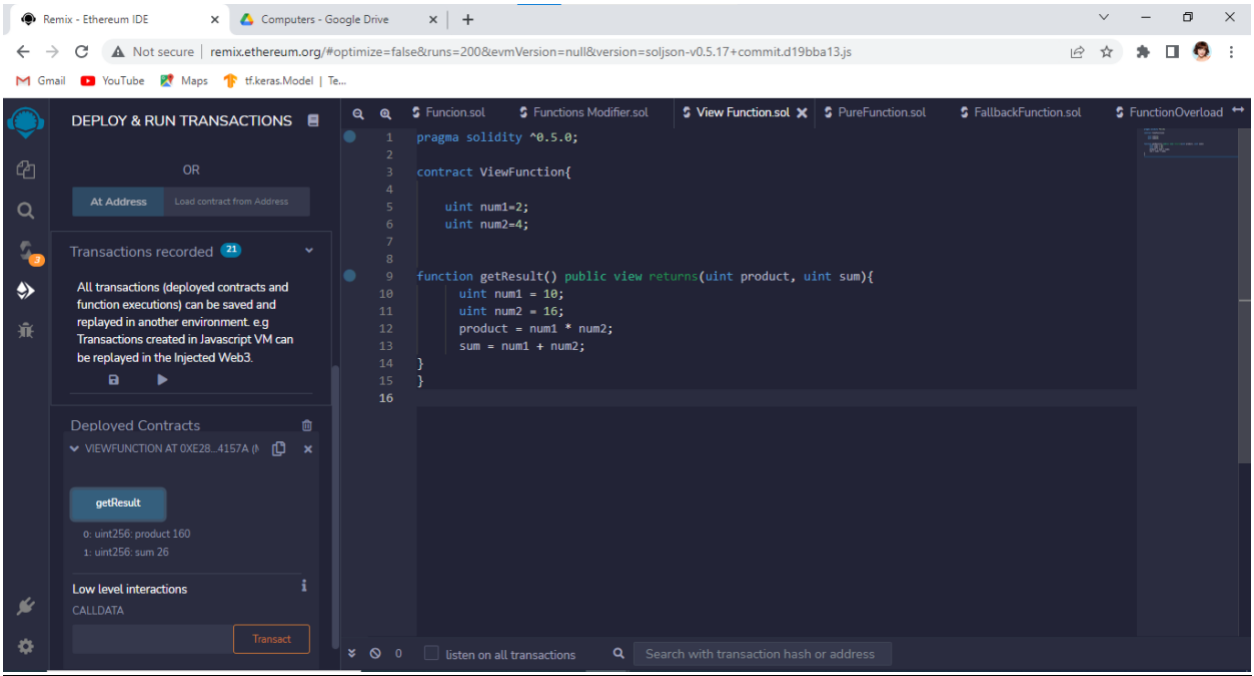


View function

Code:

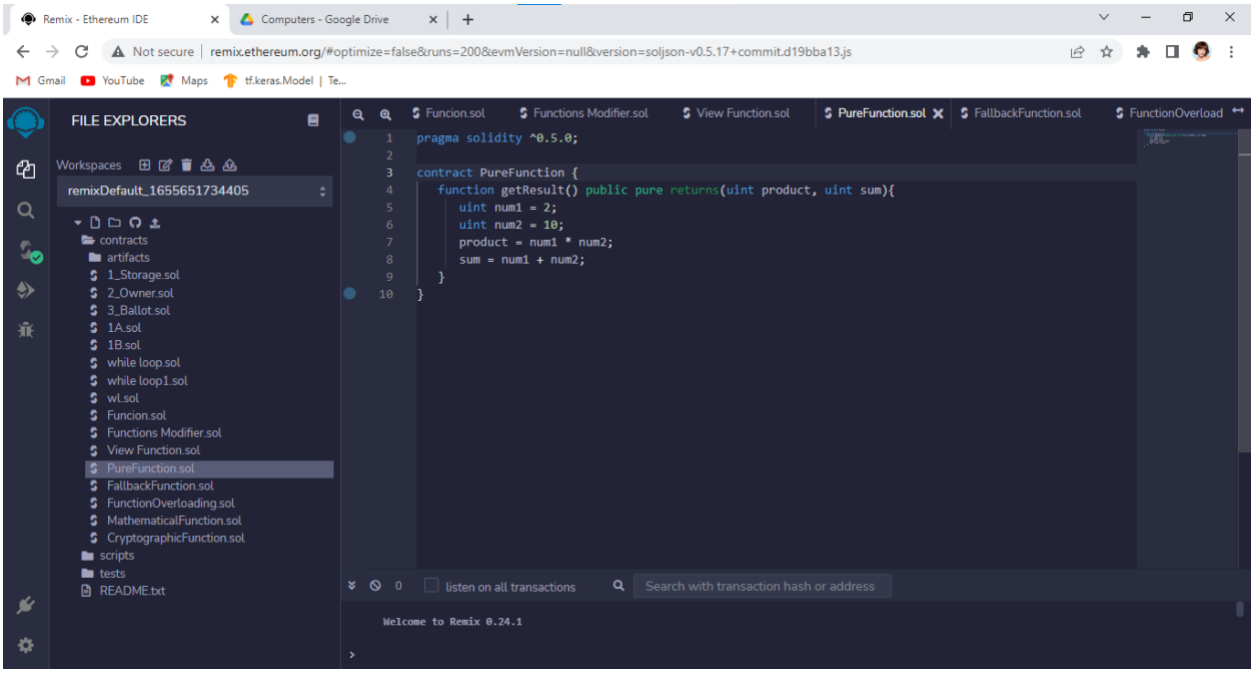


Output:

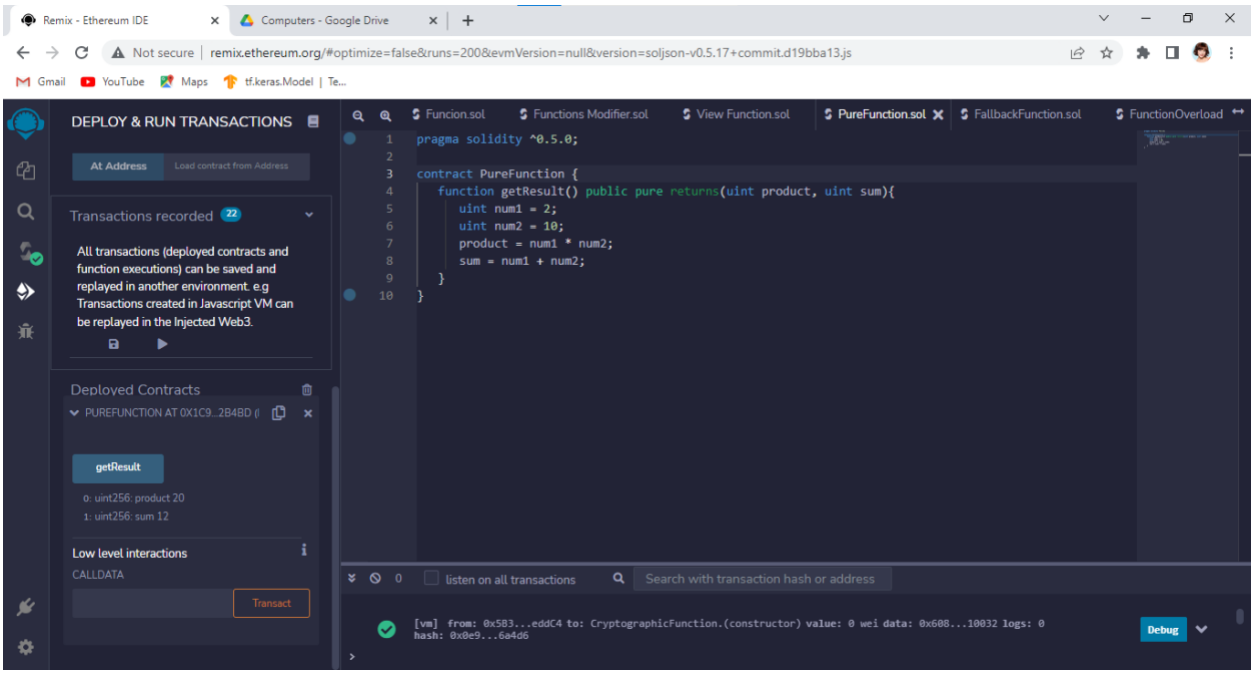


Pure function

Code:

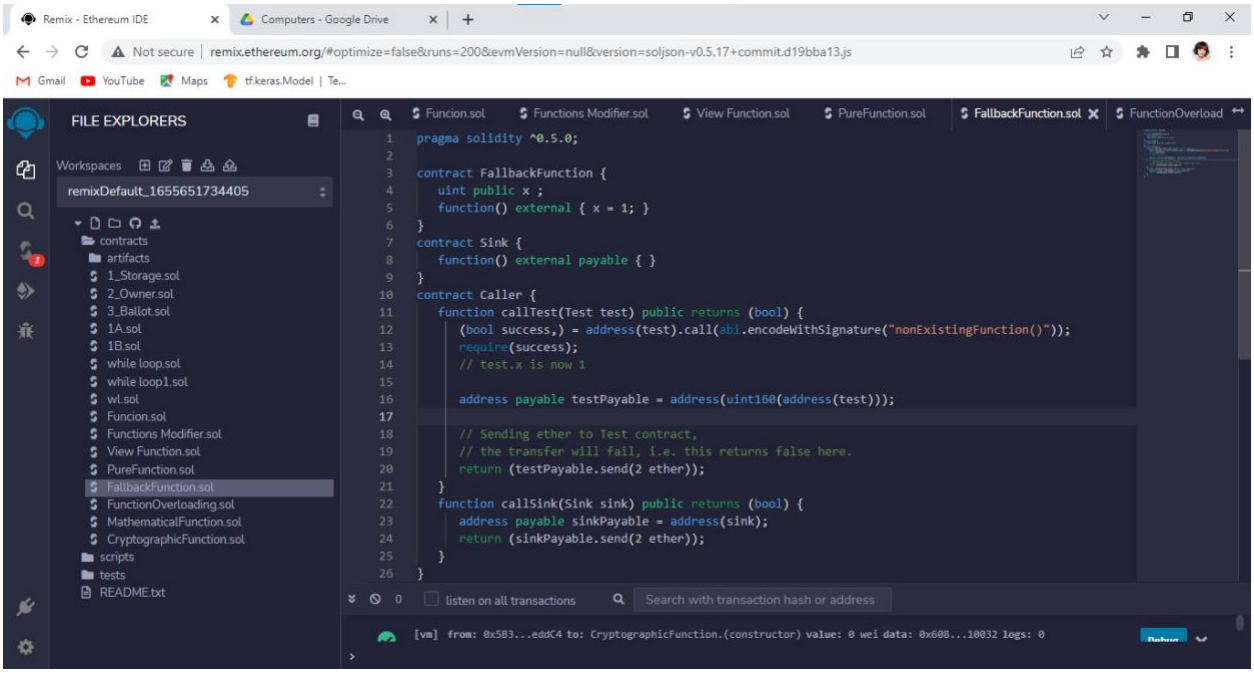


Output:

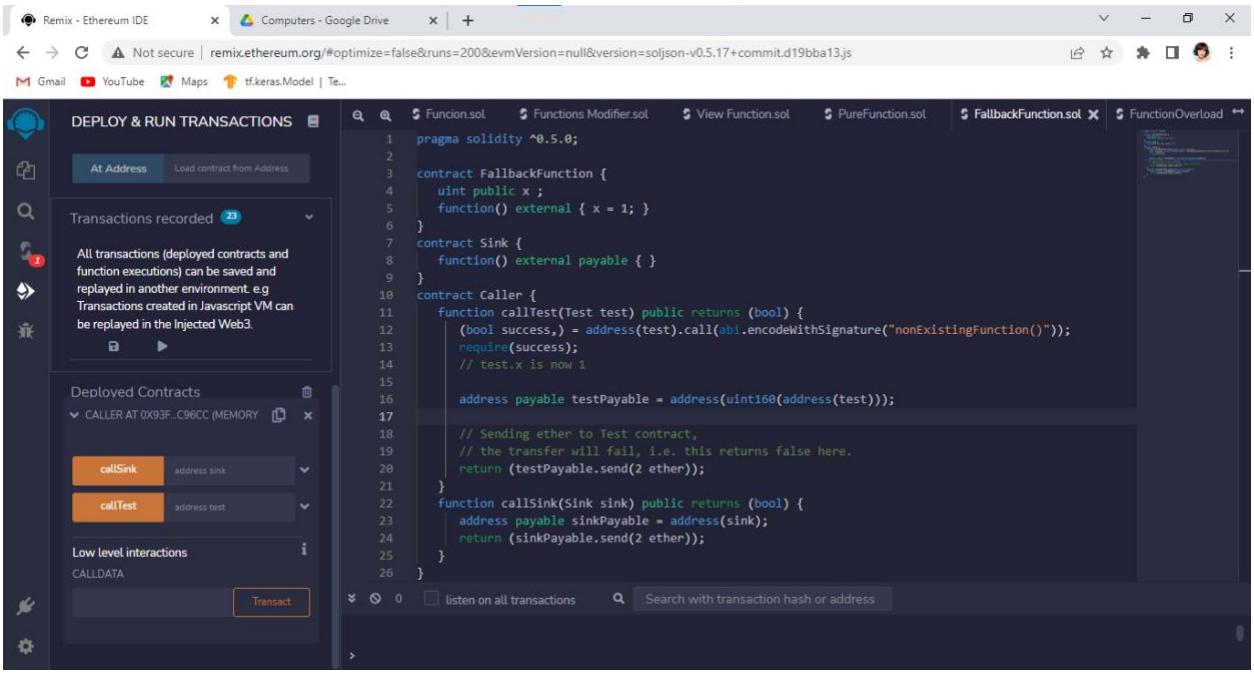


Fallback function

Code:

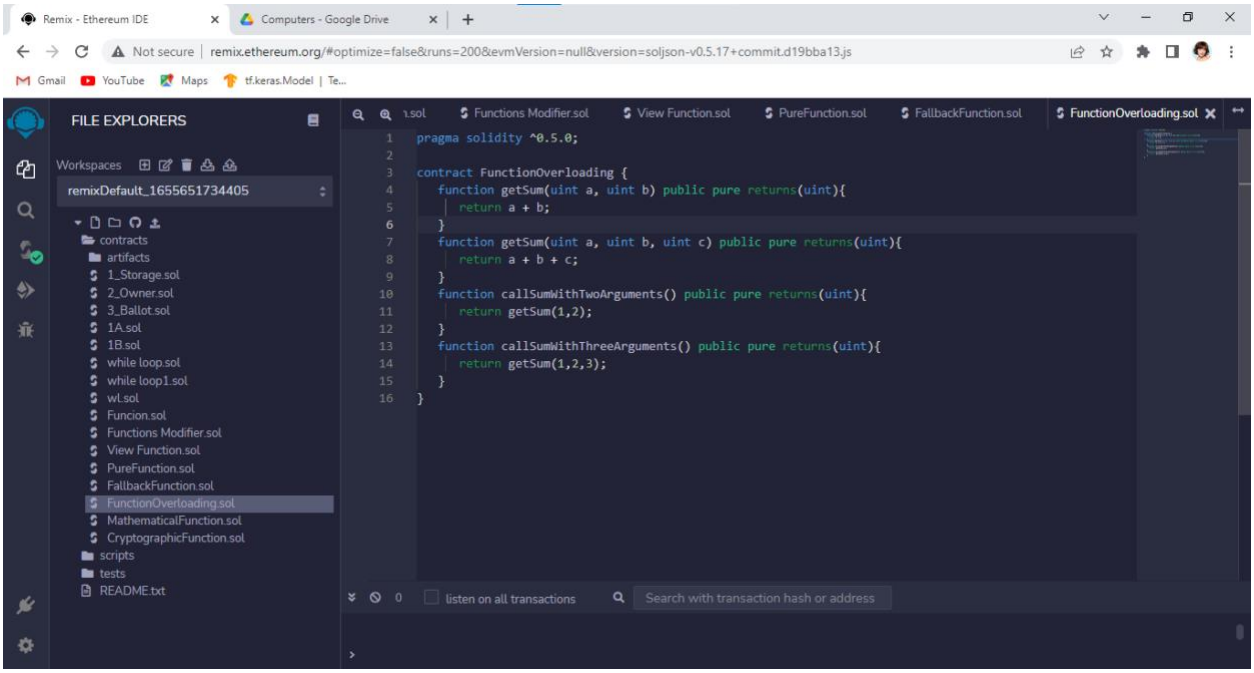


Output:

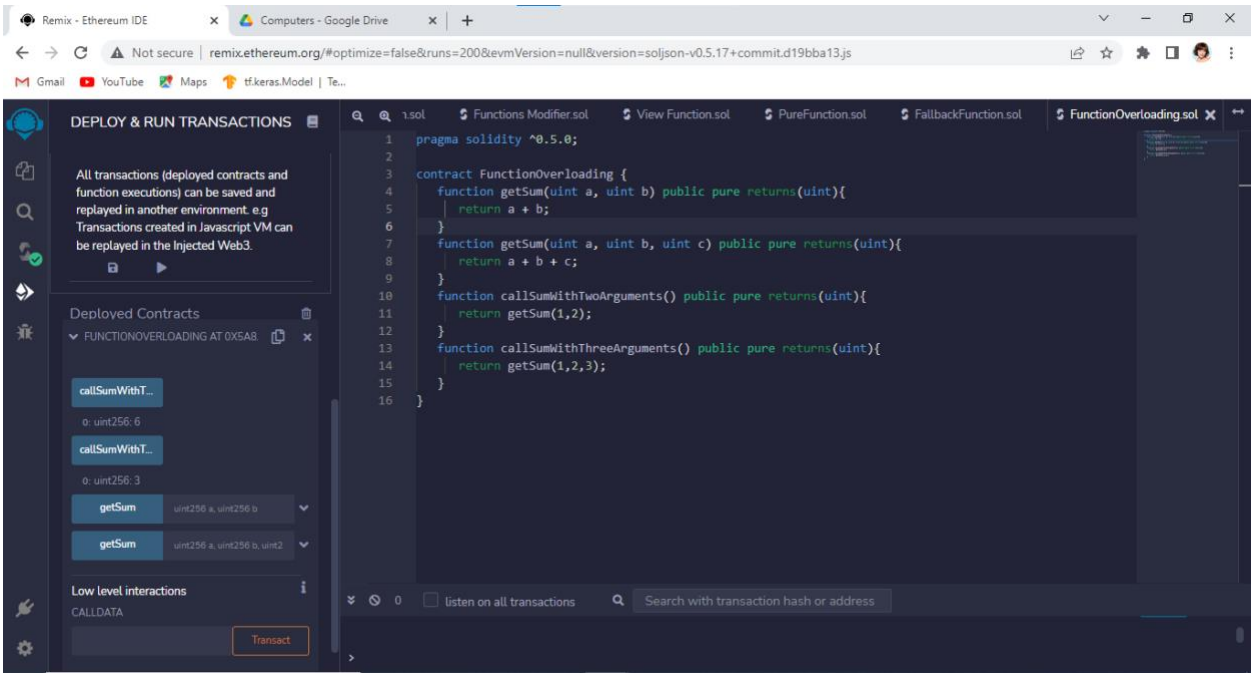


Function Overloading

Code:

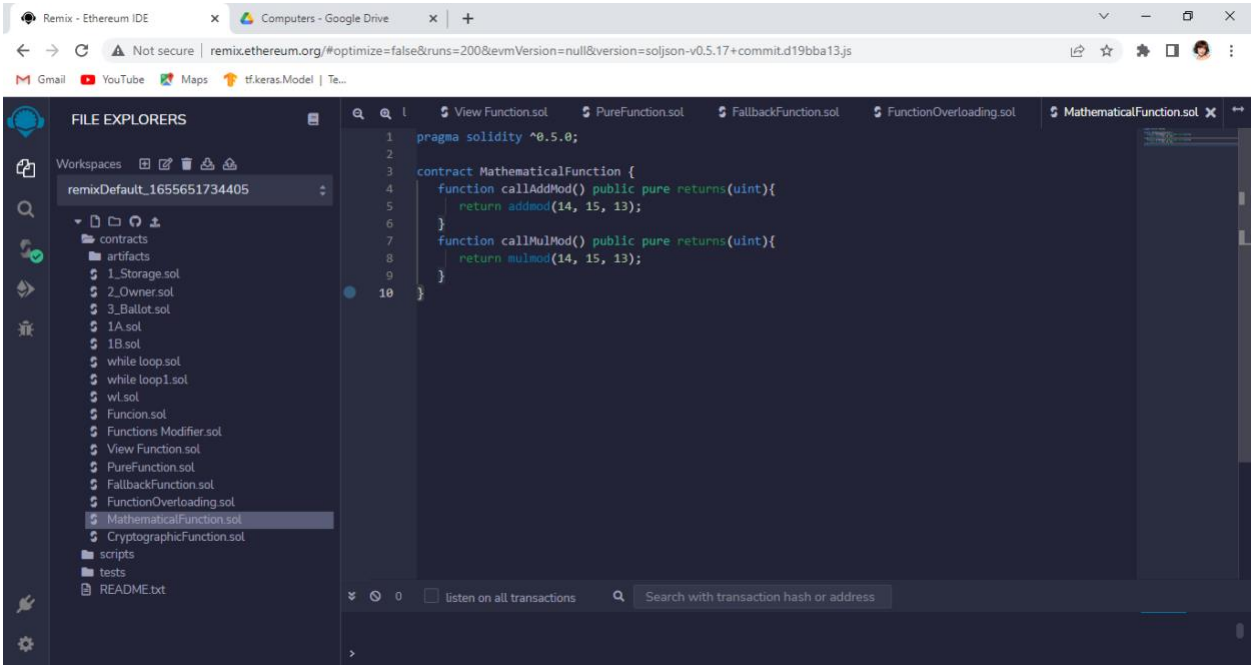


Output:

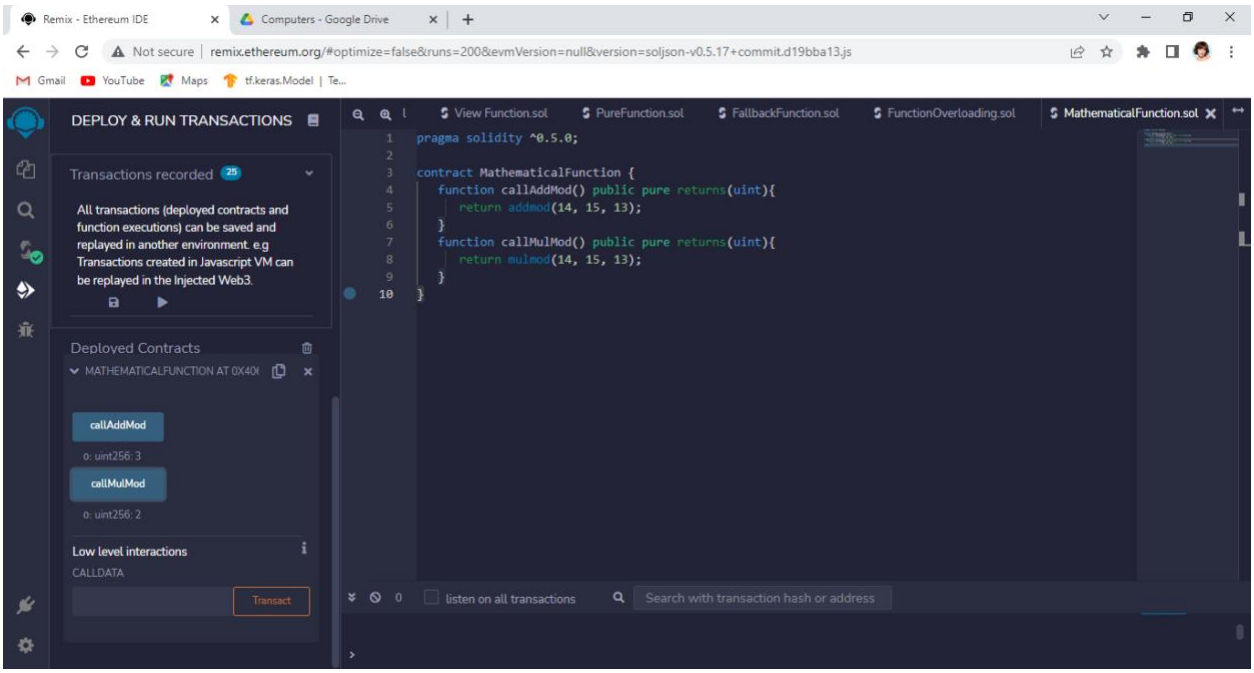


Mathematical Function

Code:

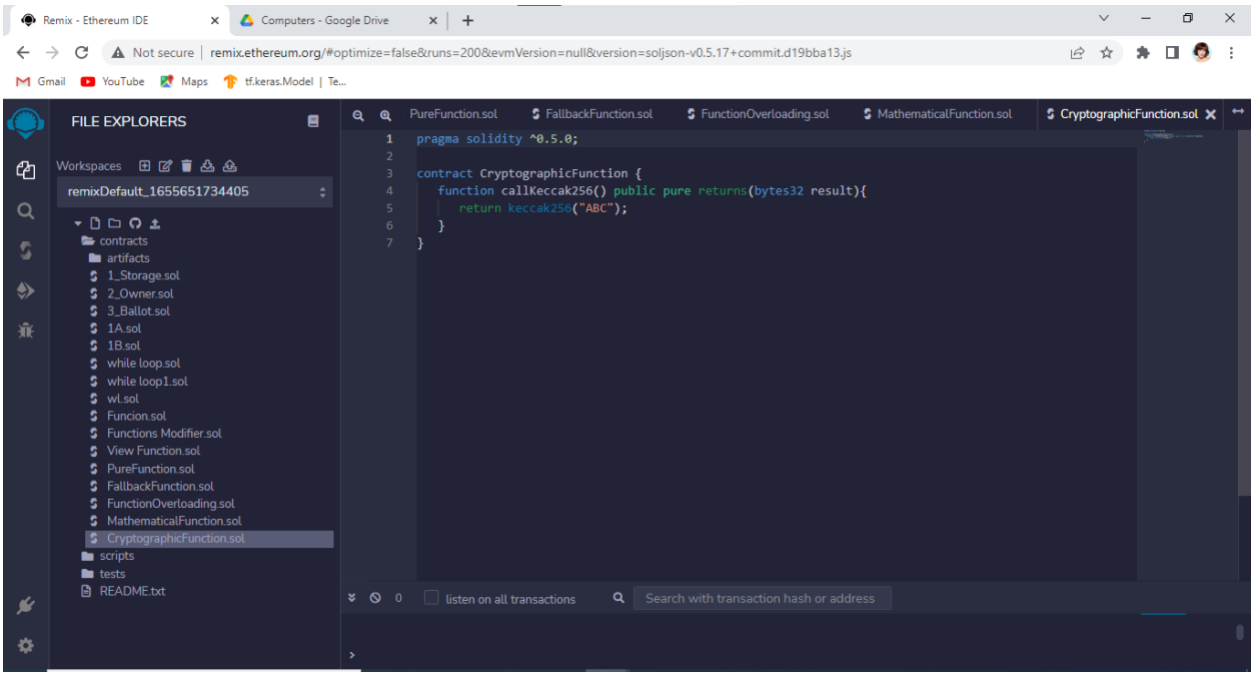


Output:

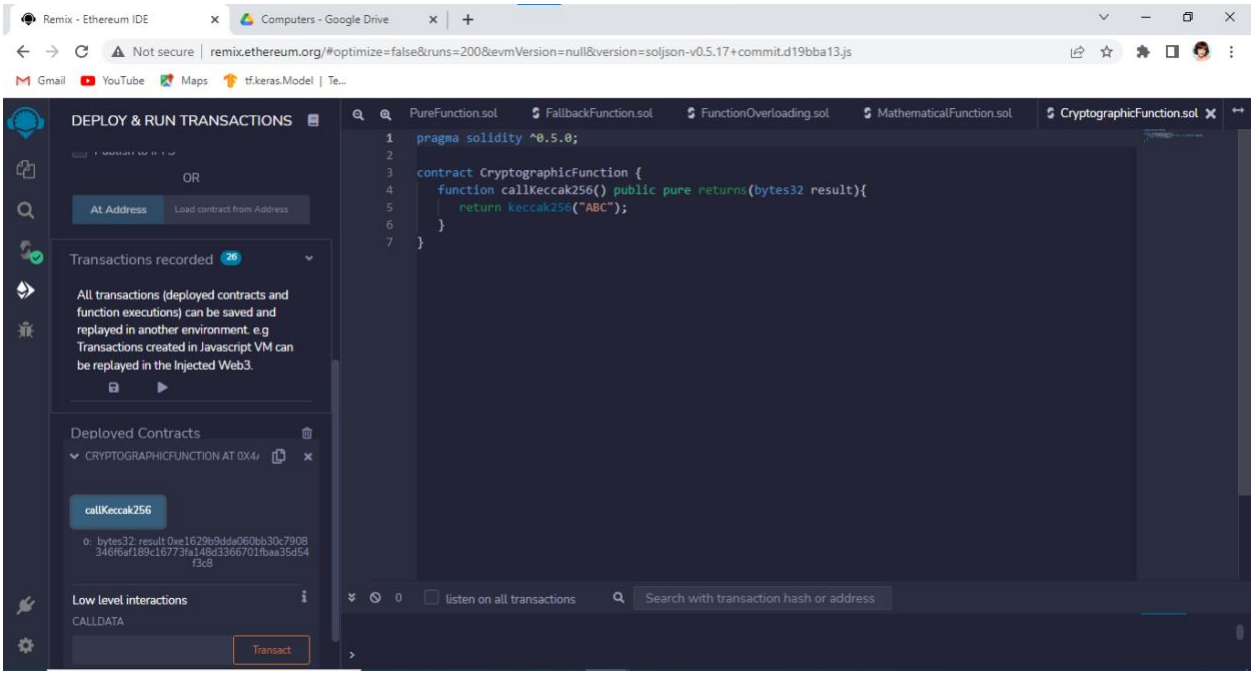


Cryptographic function

Code:



Output:



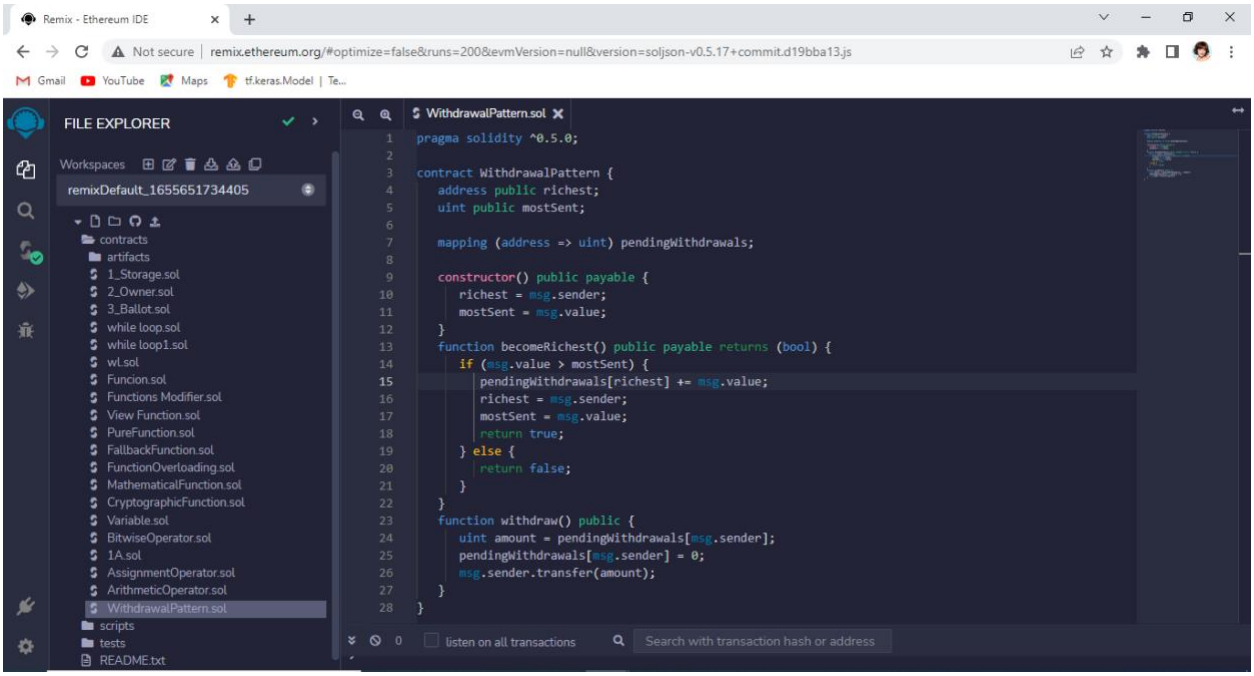
Practical No: 3

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

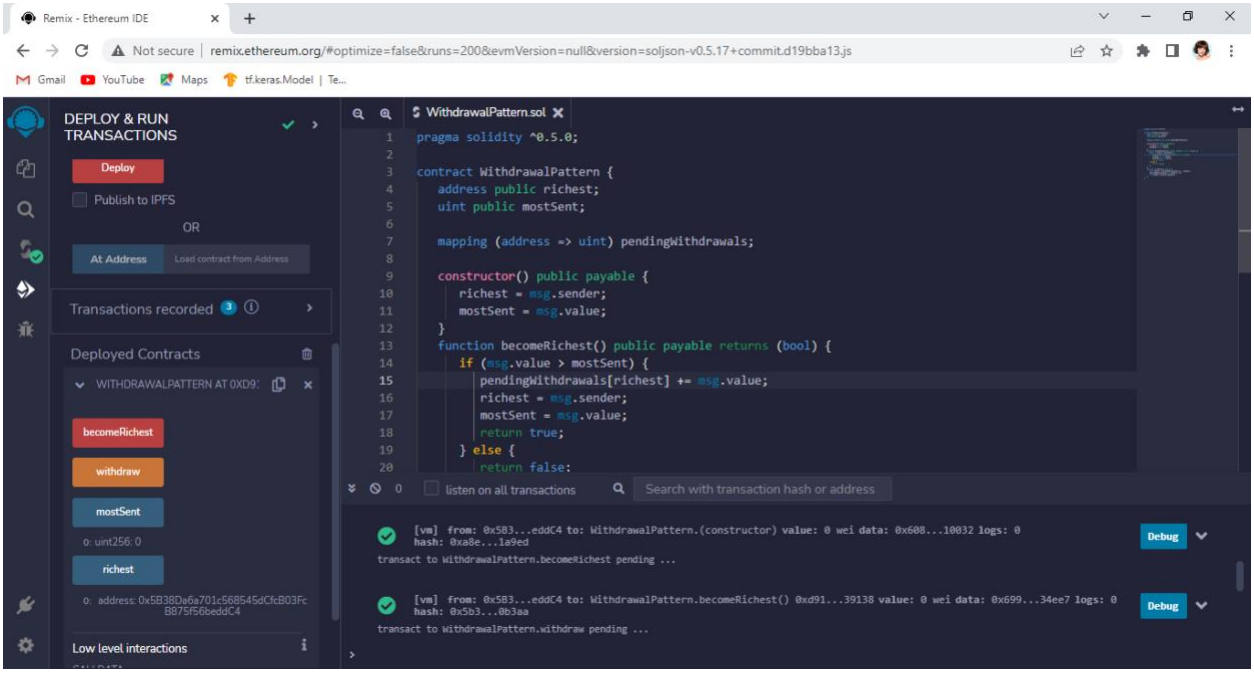
3a) Withdrawal Pattern, Restricted Access.

<u>Withdrawal Pattern</u>

Code:

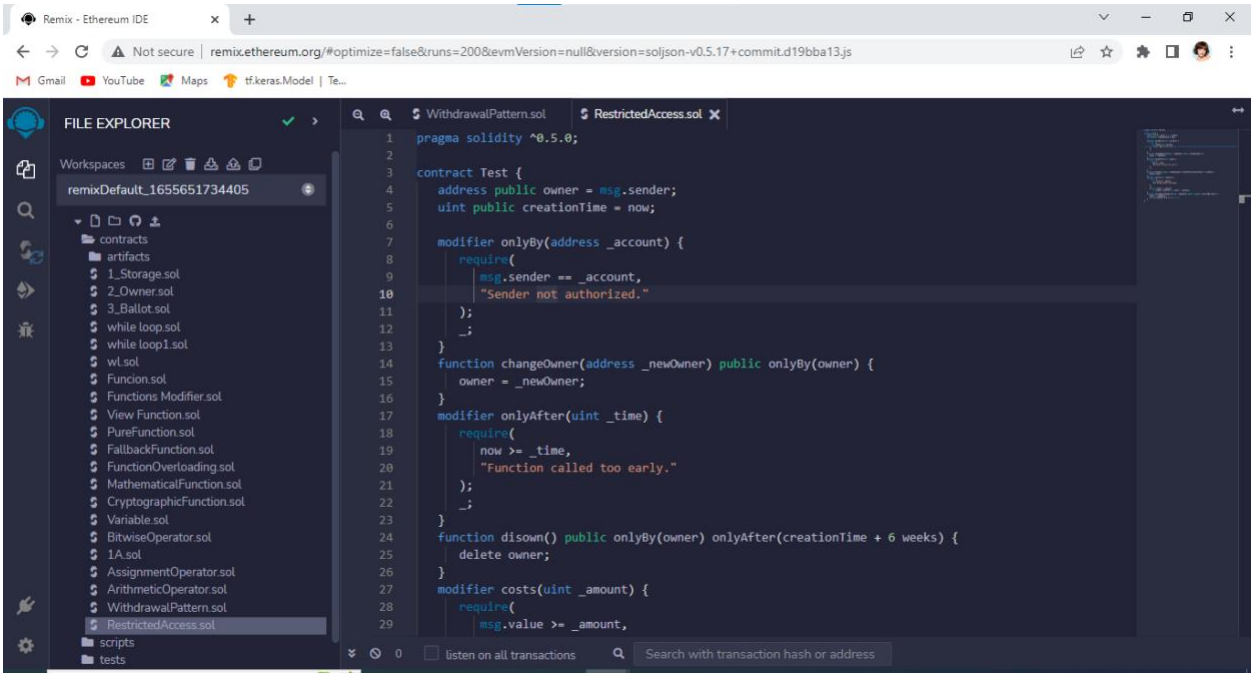


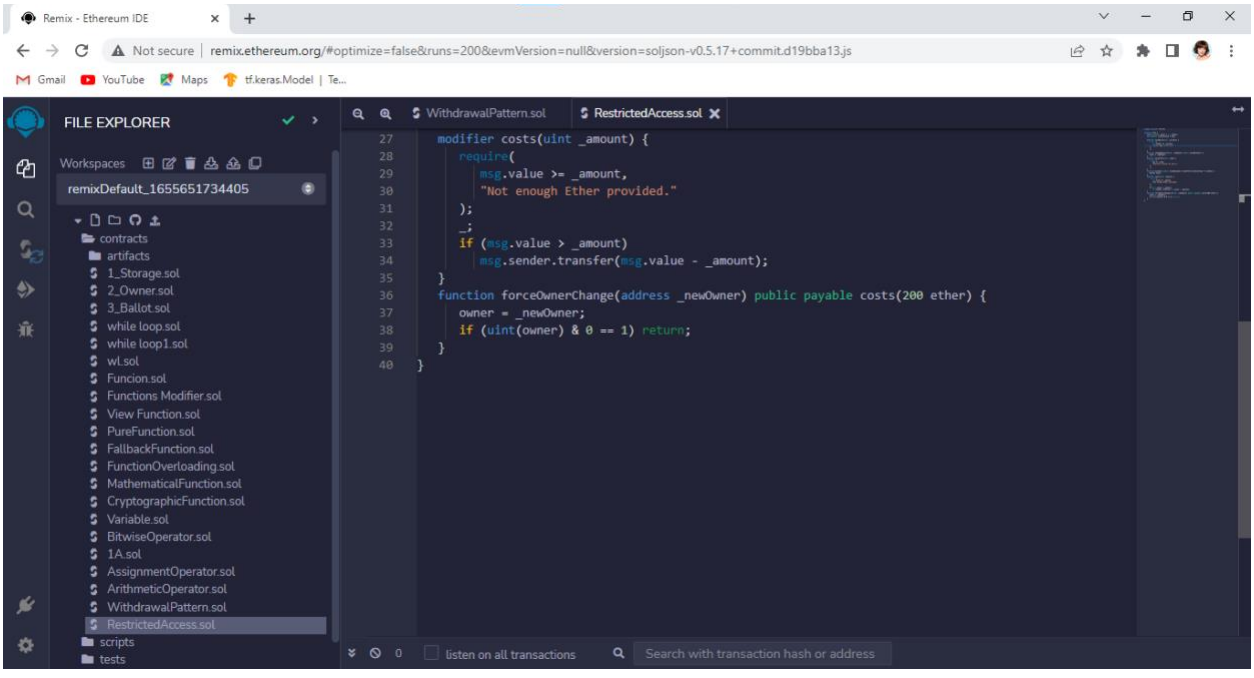
Output:


Remix - Ethereum IDE
Not secure | remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.5.17+commit.d19bba13.js
Gmail
YouTube
Maps
tf.keras.Model | Te...
DEPLOY & RUN TRANSACTIONS
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded
Deployed Contracts
WITHDRAWALPATTERN AT 0XD9:
becomeRichest
withdraw
mostSent
0: uint256: 0
richest
0: address: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
Low level interactions
WithdrawalPattern.sol
pragma solidity ^0.5.0;
contract WithdrawalPattern {
address public richest;
uint public mostSent;
mapping (address => uint) pendingWithdrawals;
constructor() public payable {
richest = msg.sender;
mostSent = msg.value;
}
function becomeRichest() public payable returns (bool) {
if (msg.value > mostSent) {
pendingWithdrawals[richest] += msg.value;
richest = msg.sender;
mostSent = msg.value;
return true;
} else {
return false;
listen on all transactions
Search with transaction hash or address
[vm] from: 0x583...eddC4 to: WithdrawalPattern.(constructor) value: 0 wei data: 0x608...10032 logs: 0
hash: 0xa8e...1a9ed
transact to WithdrawalPattern.becomeRichest pending ...
Debug
[vm] from: 0x583...eddC4 to: WithdrawalPattern.becomeRichest() 0xd91...39138 value: 0 wei data: 0x699...34ee7 logs: 0
hash: 0x5b3...0b3aa
transact to WithdrawalPattern.withdraw pending ...
Debug

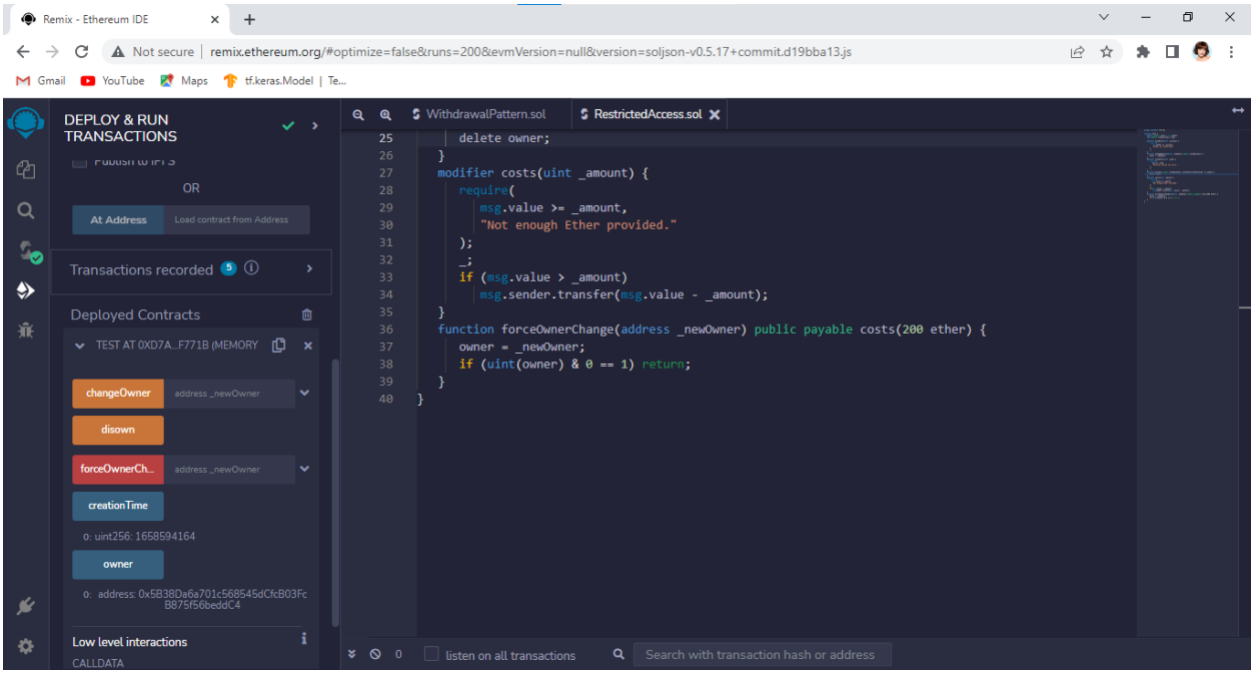
Restricted Access

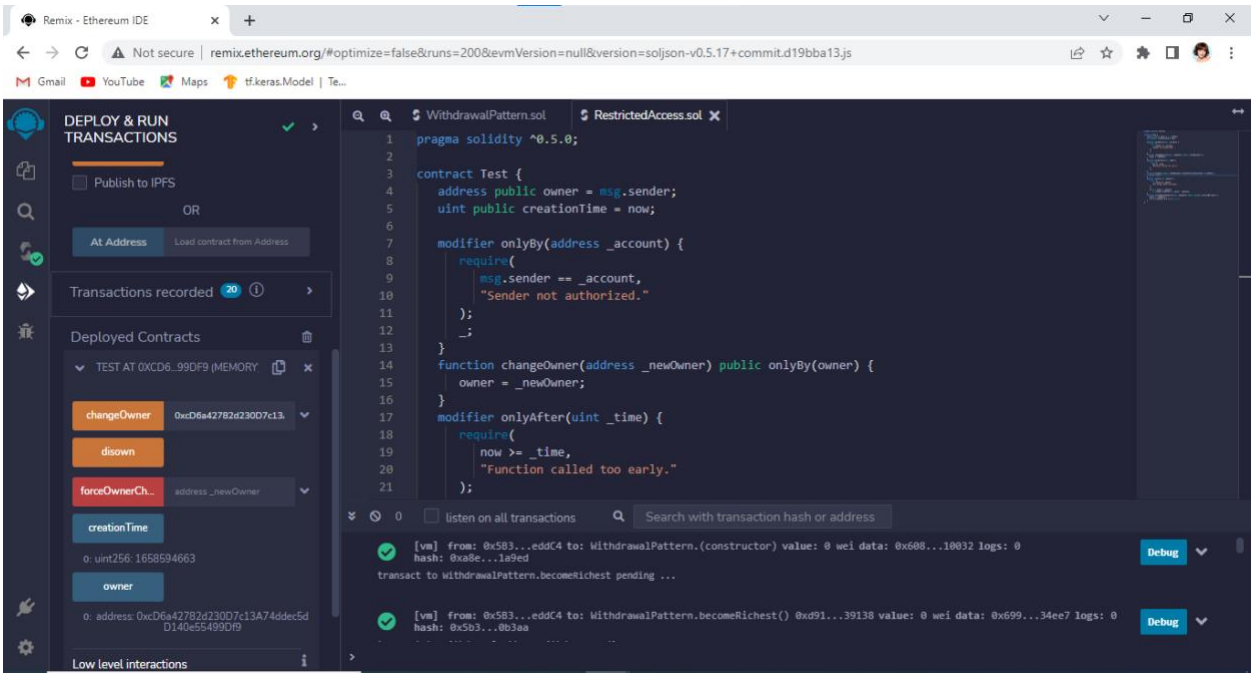
Code:





Output:

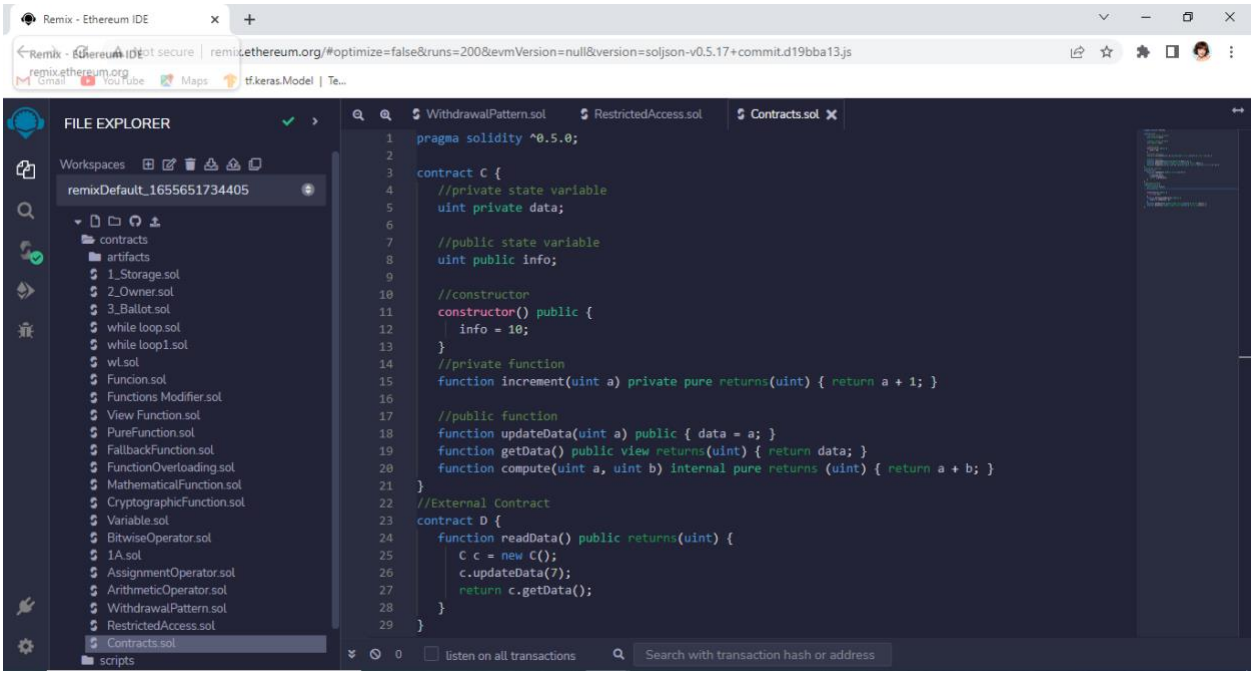

Remix - Ethereum IDE
Not secure | remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.5.17+commit.d19bba13.js
DEPLOY & RUN TRANSACTIONS
OR
At Address
Load contract from Address
Transactions recorded 5
Deployed Contracts
TEST AT 0XD7A...F771B (MEMORY
changeOwner
address _newOwner
disown
forceOwnerCh...
creationTime
0: uint256: 1658594164
owner
0: address: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
Low level interactions
CALLDATA
WithdrawalPattern.sol
RestrictedAccess.sol
delete owner;
}
modifier costs(uint _amount) {
require(
msg.value >= _amount,
"Not enough Ether provided."
);
_;
if (msg.value > _amount)
msg.sender.transfer(msg.value - _amount);
}
function forceOwnerChange(address _newOwner) public payable costs(200 ether) {
owner = _newOwner;
if (uint(owner) & 0 == 1) return;
}
}
listen on all transactions
Search with transaction hash or address

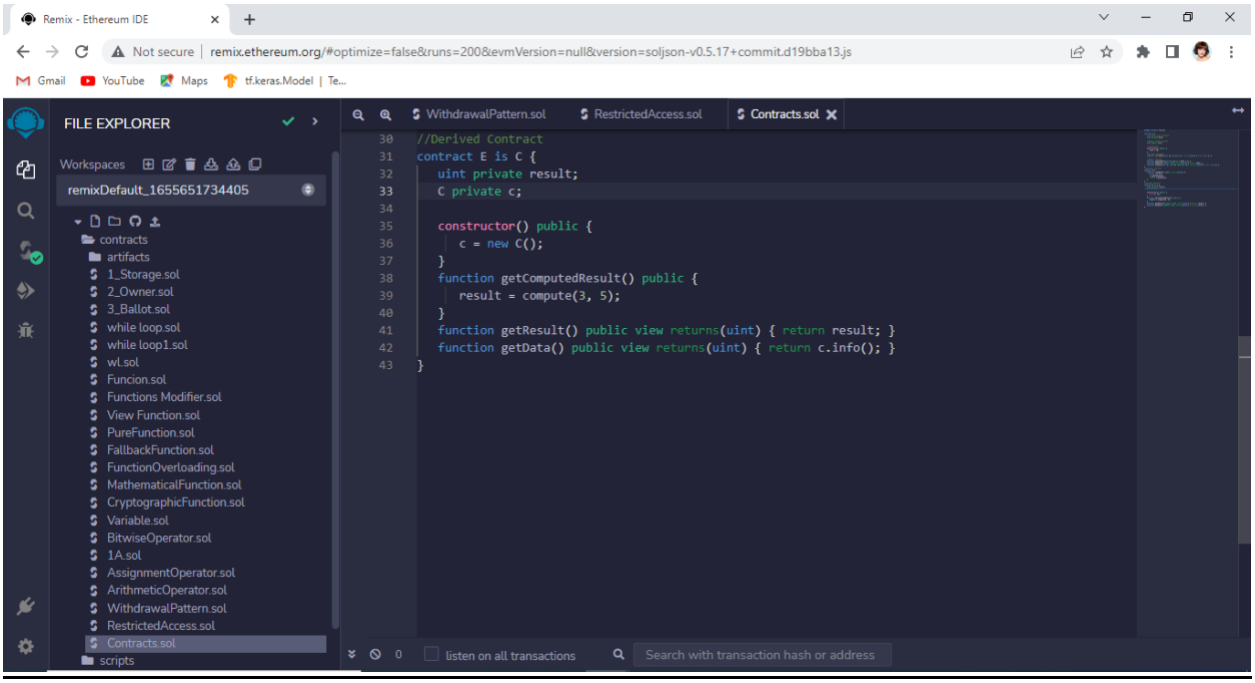

Remix - Ethereum IDE
Not secure | remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.5.17+commit.d19bba13.js
DEPLOY & RUN TRANSACTIONS
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 20
Deployed Contracts
TEST AT 0XCD6...99DF9 (MEMORY
changeOwner
0xcD6a42782d230D7c13
disown
forceOwnerCh...
address _newOwner
creationTime
0: uint256: 1658594663
owner
0: address: 0xcD6a42782d230D7c13A74ddec5dD140e55499Df9
Low level interactions
WithdrawalPattern.sol
RestrictedAccess.sol
pragma solidity ^0.5.0;
contract Test {
address public owner = msg.sender;
uint public creationTime = now;
modifier onlyBy(address _account) {
require(
msg.sender == _account,
"Sender not authorized."
);
_;
}
function changeOwner(address _newOwner) public onlyBy(owner) {
owner = _newOwner;
}
modifier onlyAfter(uint _time) {
require(
now >= _time,
"Function called too early."
);
listen on all transactions
Search with transaction hash or address
[vm] from: 0x583...eddC4 to: WithdrawalPattern.(constructor) value: 0 wei data: 0x608...10032 logs: 0 hash: 0xa8e...1a9ed
transact to WithdrawalPattern.becomeRichest pending ...
[vm] from: 0x583...eddC4 to: WithdrawalPattern.becomeRichest() 0xd91...39138 value: 0 wei data: 0x699...34ee7 logs: 0 hash: 0x5b3...0b3aa
Debug
Debug

3b) Contracts, Inheritance, Constructors, Abstract Contracts, Interfaces.

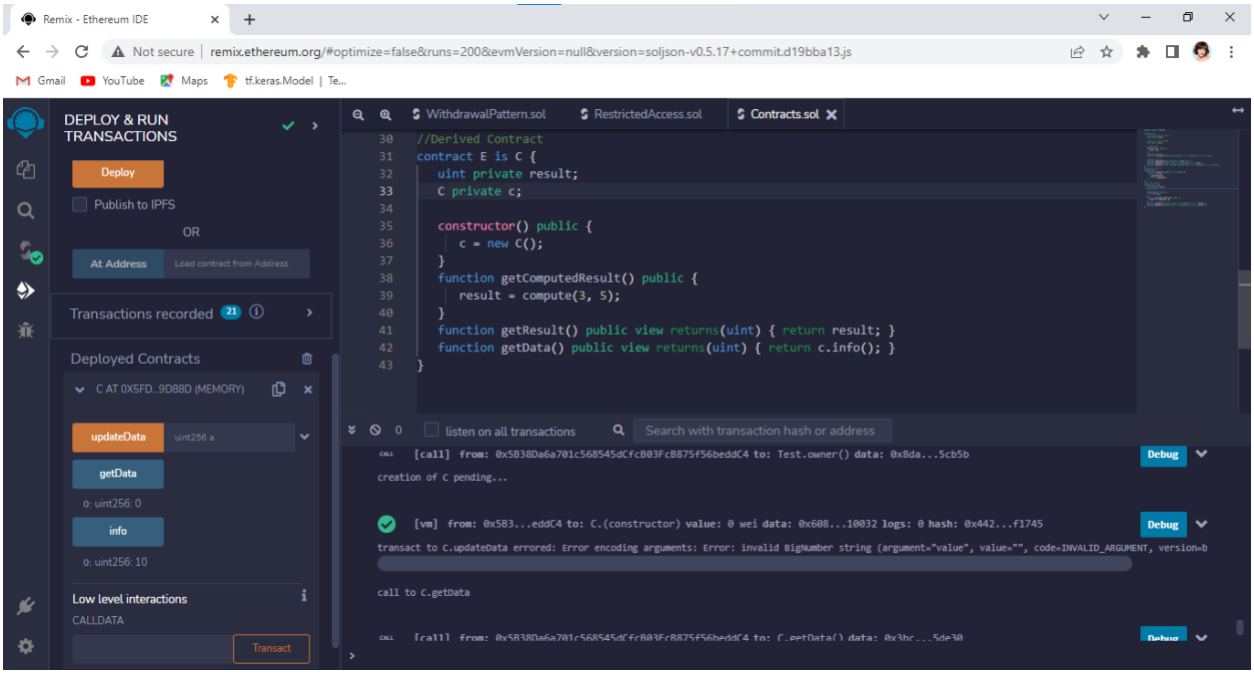
Contracts

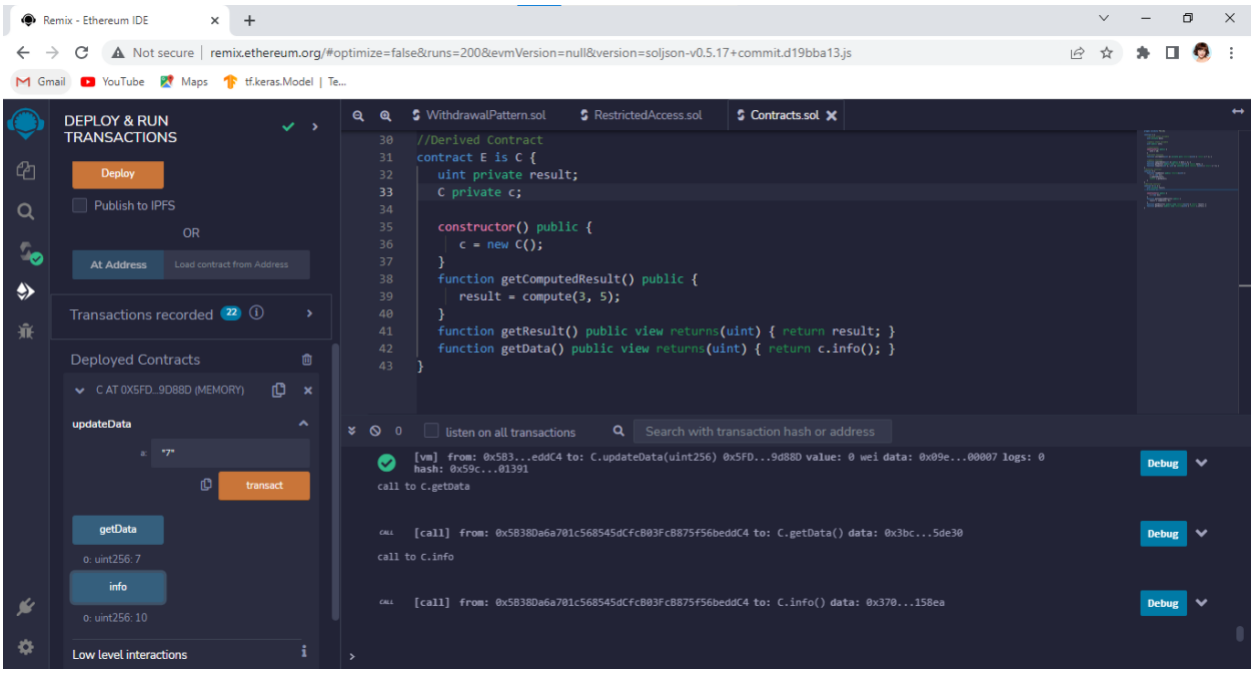
Code:





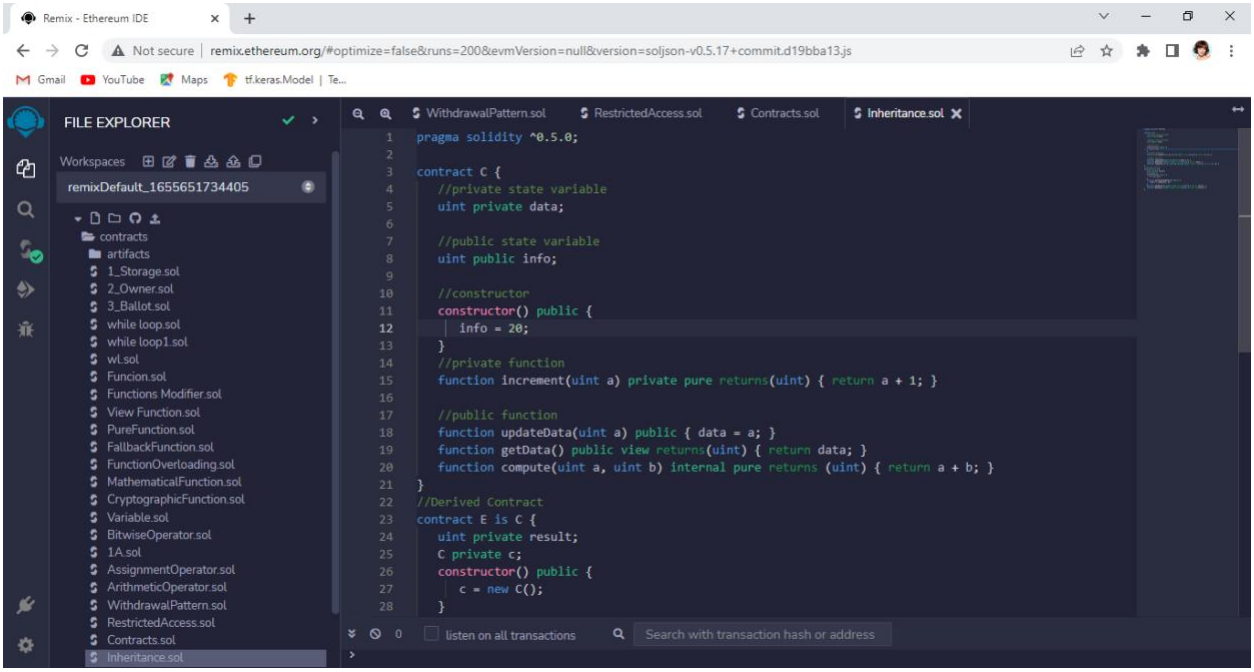
Output:

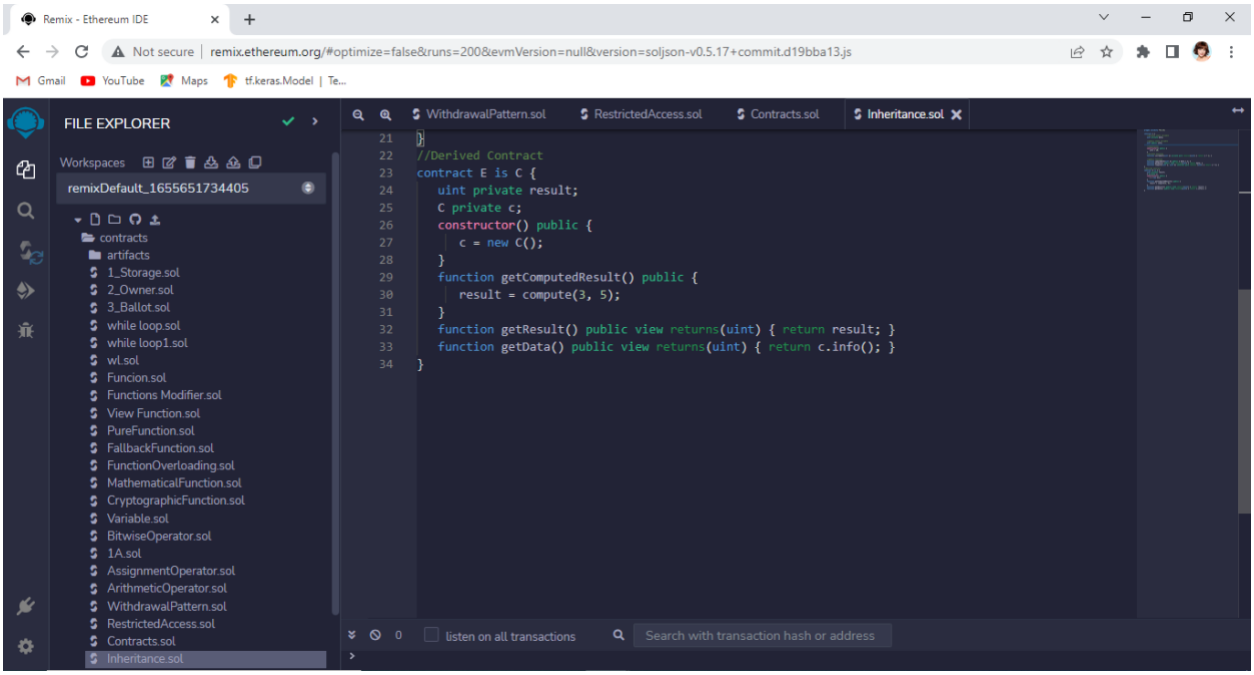

Remix - Ethereum IDE
Not secure | remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.5.17+commit.d19bba13.js
DEPLOY & RUN TRANSACTIONS
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 21
Deployed Contracts
C AT 0X5FD...9D88D (MEMORY)
updateData
uint256 a
getData
0: uint256: 0
info
0: uint256: 10
Low level interactions
CALLDATA
Transact
WithdrawalPattern.sol
RestrictedAccess.sol
Contracts.sol
//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 all transactions
Search with transaction hash or address
[call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: Test.owner() data: 0x8da...5cb5b
Debug
creation of C pending...
[vm] from: 0x583...eddC4 to: C.(constructor) value: 0 wei data: 0x608...10032 logs: 0 hash: 0x442...f1745
Debug
transact to C.updateData errored: Error encoding arguments: Error: invalid BigNumber string (argument="value", value="", code=INVALID_ARGUMENT, version=b
call to C.getData


Remix - Ethereum IDE
Not secure | remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.5.17+commit.d19bba13.js
DEPLOY & RUN TRANSACTIONS
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 22
Deployed Contracts
C AT 0X5FD...9D88D (MEMORY)
updateData
a: "7"
transact
getData
0: uint256: 7
info
0: uint256: 10
Low level interactions
WithdrawalPattern.sol
RestrictedAccess.sol
Contracts.sol
//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 all transactions
Search with transaction hash or address
[vm] from: 0x583...eddC4 to: C.updateData(uint256) 0x5FD...9d88D value: 0 wei data: 0x09e...00007 logs: 0 hash: 0x59c...01391
Debug
call to C.getData
[call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: C.getData() data: 0x3bc...5de30
Debug
call to C.info
[call] from: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 to: C.info() data: 0x370...158ea
Debug

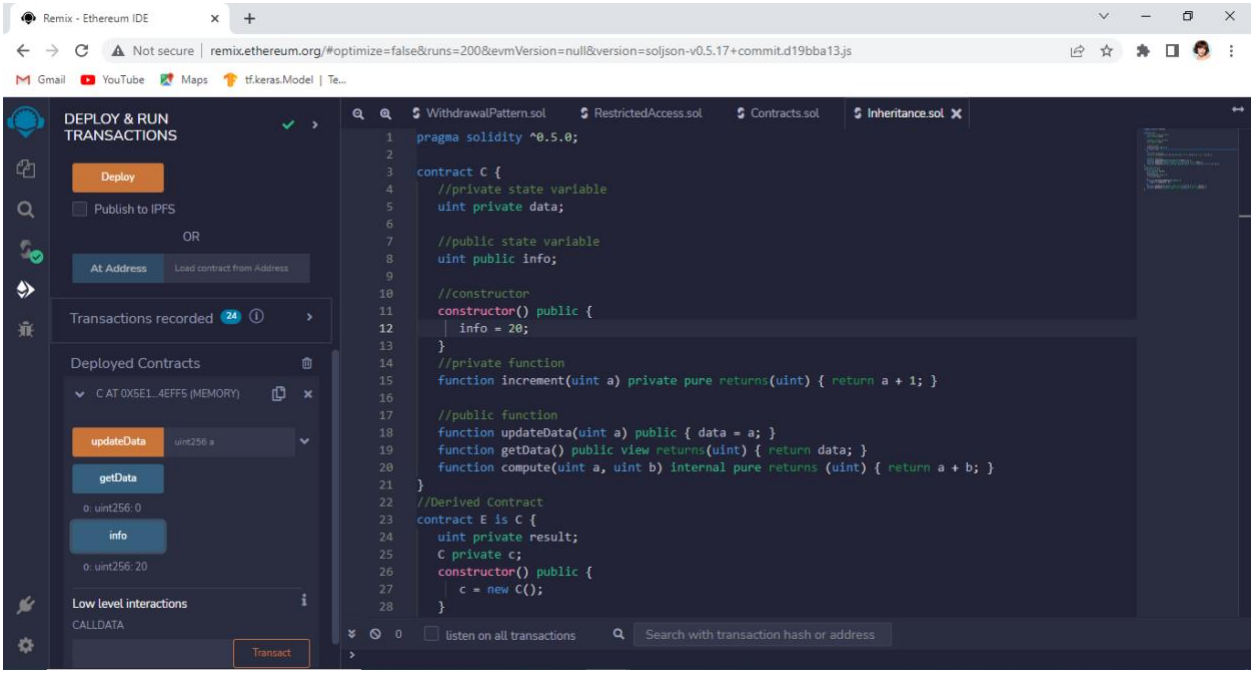
Inheritance

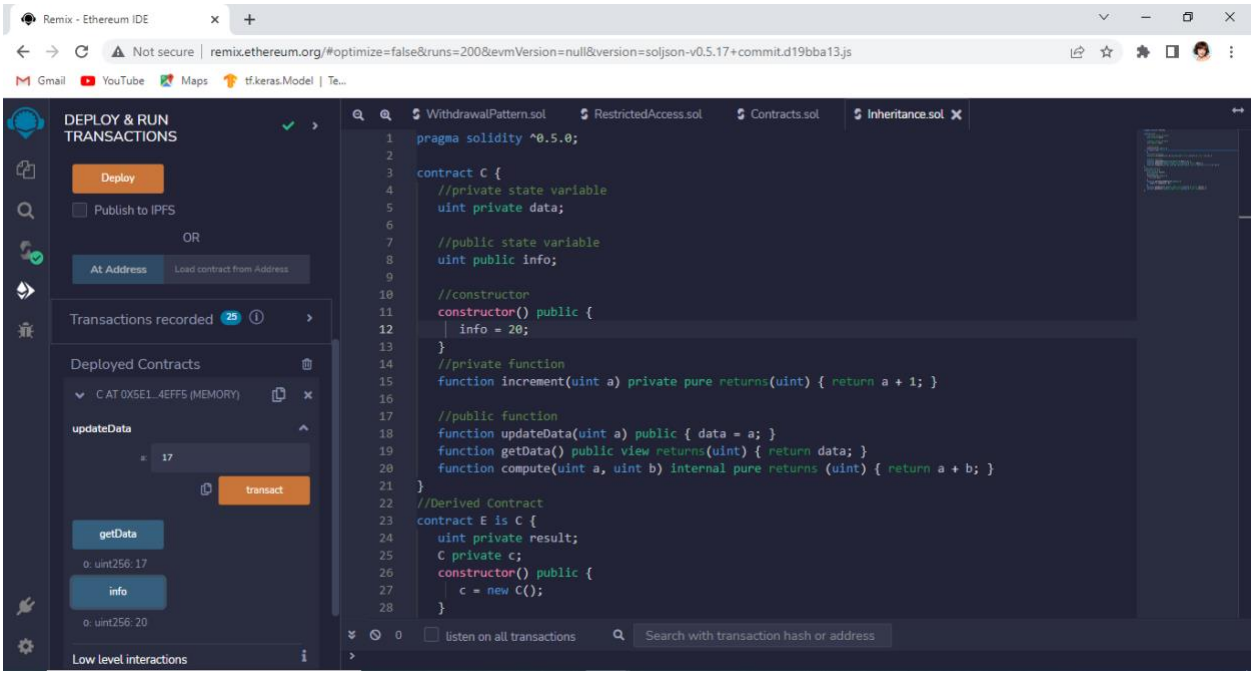
Code:





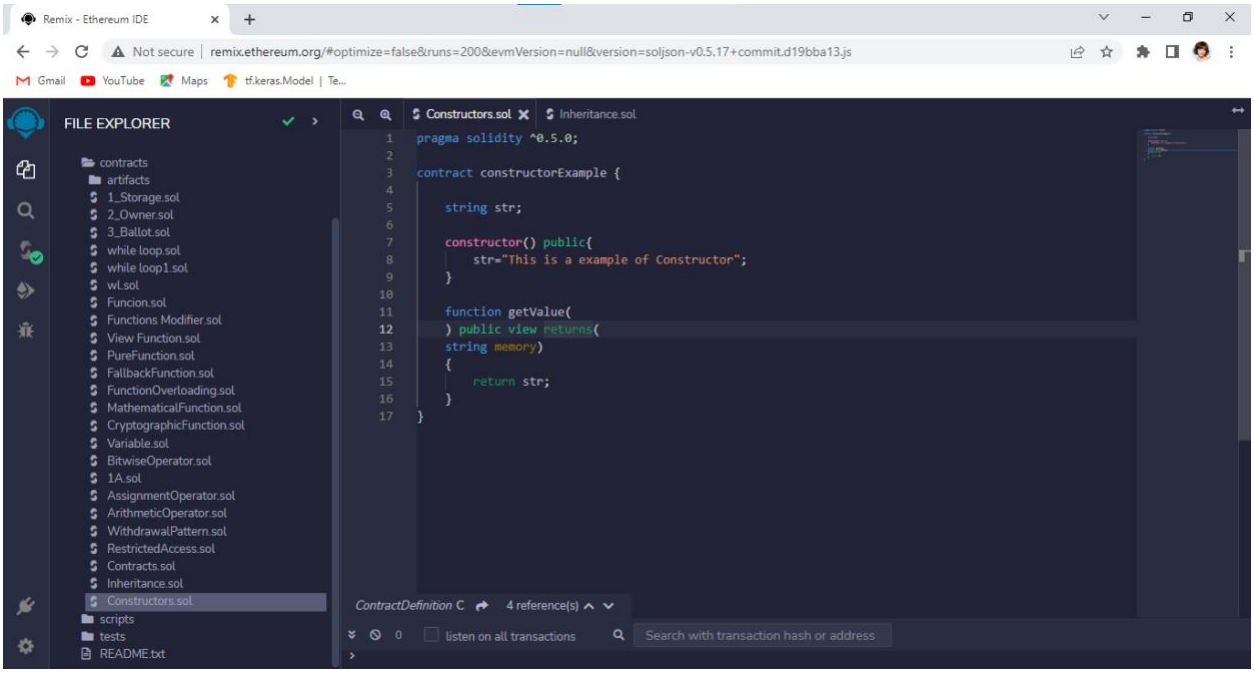
Output:


Remix - Ethereum IDE
Not secure | remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.5.17+commit.d19bba13.js
DEPLOY & RUN TRANSACTIONS
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 24
Deployed Contracts
C AT 0X5E1...4EFF5 (MEMORY)
updateData
uint256 a
getData
0: uint256: 0
info
0: uint256: 20
Low level interactions
CALLDATA
Transact
WithdrawalPattern.sol
RestrictedAccess.sol
Contracts.sol
Inheritance.sol
pragma solidity ^0.5.0;
contract C {
//private state variable
uint private data;
//public state variable
uint public info;
//constructor
constructor() public {
info = 20;
}
//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();
}
listen on all transactions
Search with transaction hash or address

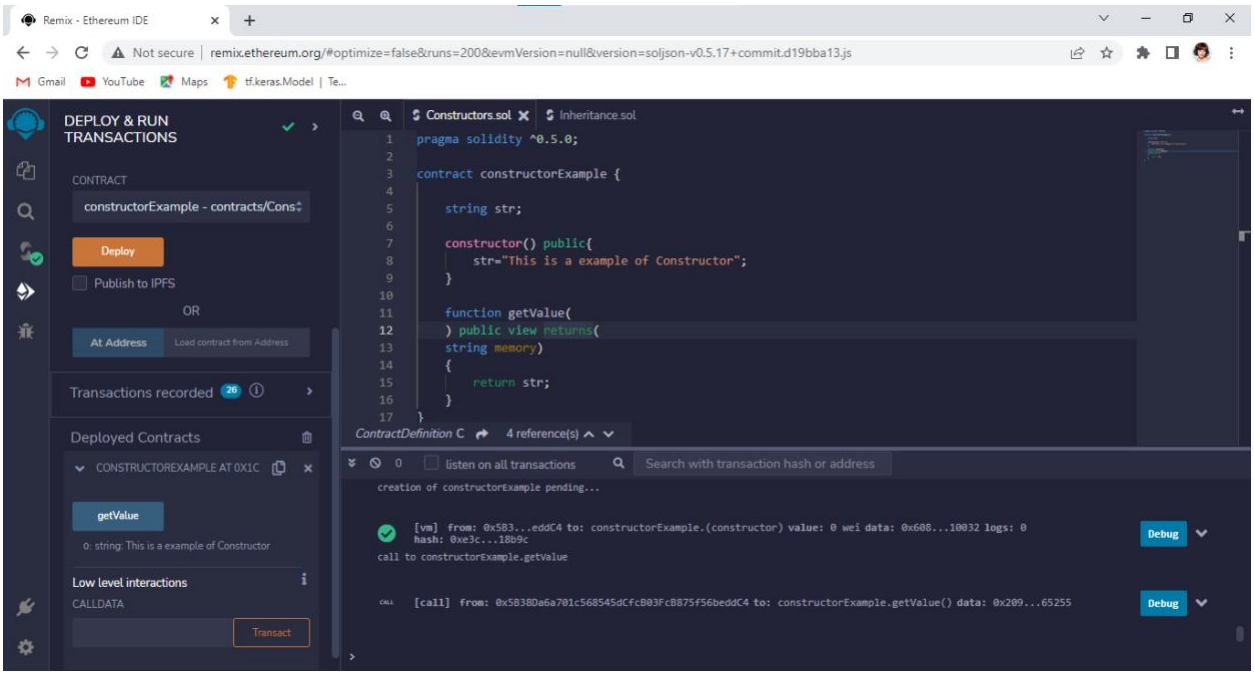

Remix - Ethereum IDE
Not secure | remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.5.17+commit.d19bba13.js
DEPLOY & RUN TRANSACTIONS
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded 25
Deployed Contracts
C AT 0X5E1...4EFF5 (MEMORY)
updateData
17
transact
getData
0: uint256: 17
info
0: uint256: 20
Low level interactions
WithdrawalPattern.sol
RestrictedAccess.sol
Contracts.sol
Inheritance.sol
pragma solidity ^0.5.0;
contract C {
//private state variable
uint private data;
//public state variable
uint public info;
//constructor
constructor() public {
info = 20;
}
//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();
}
listen on all transactions
Search with transaction hash or address

Constructors

Code:

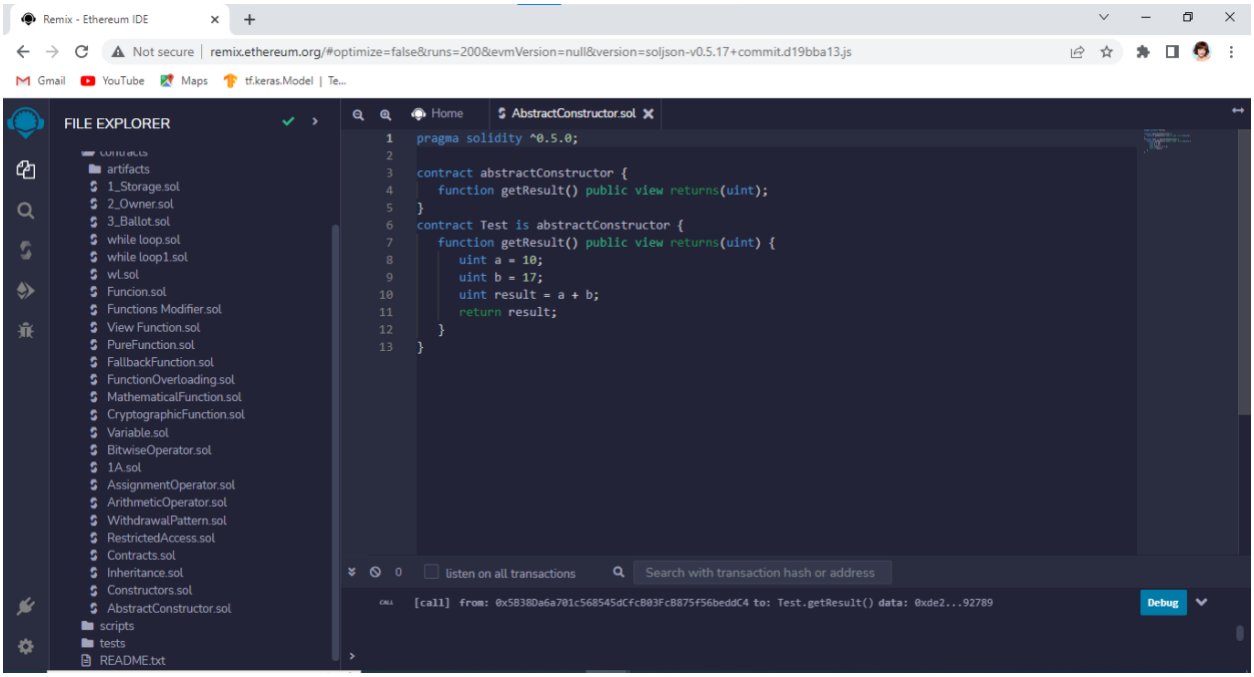


Output:

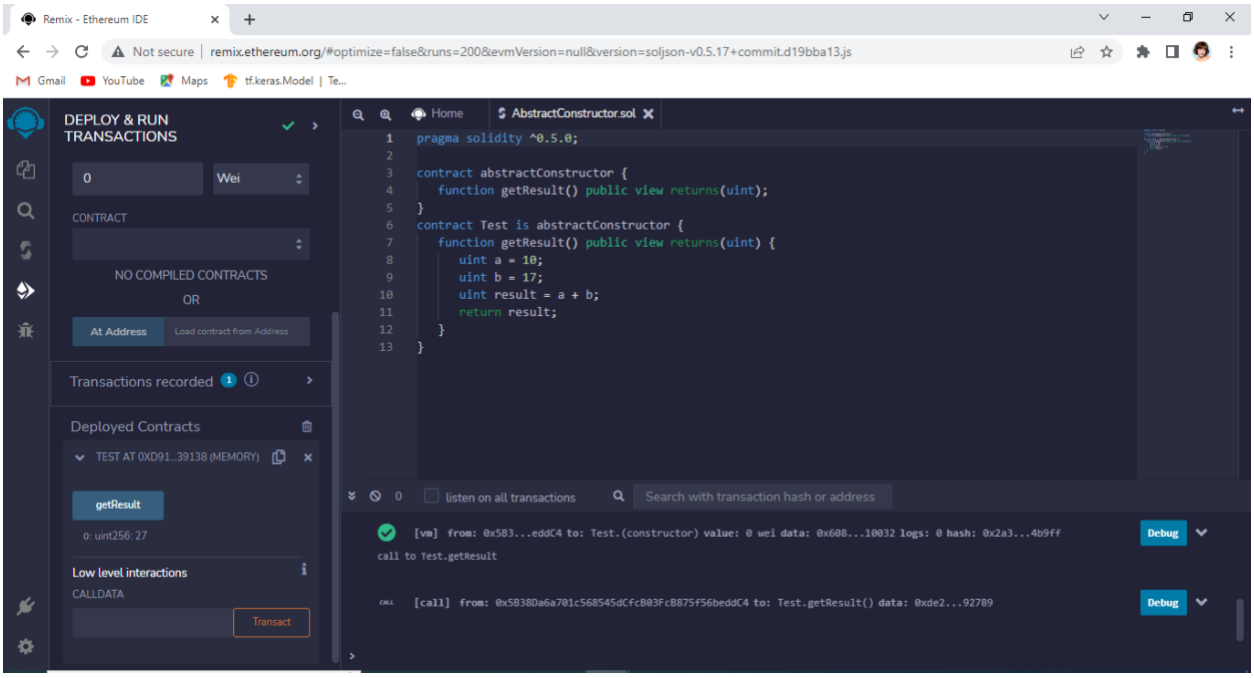


Abstract Contracts

Code:

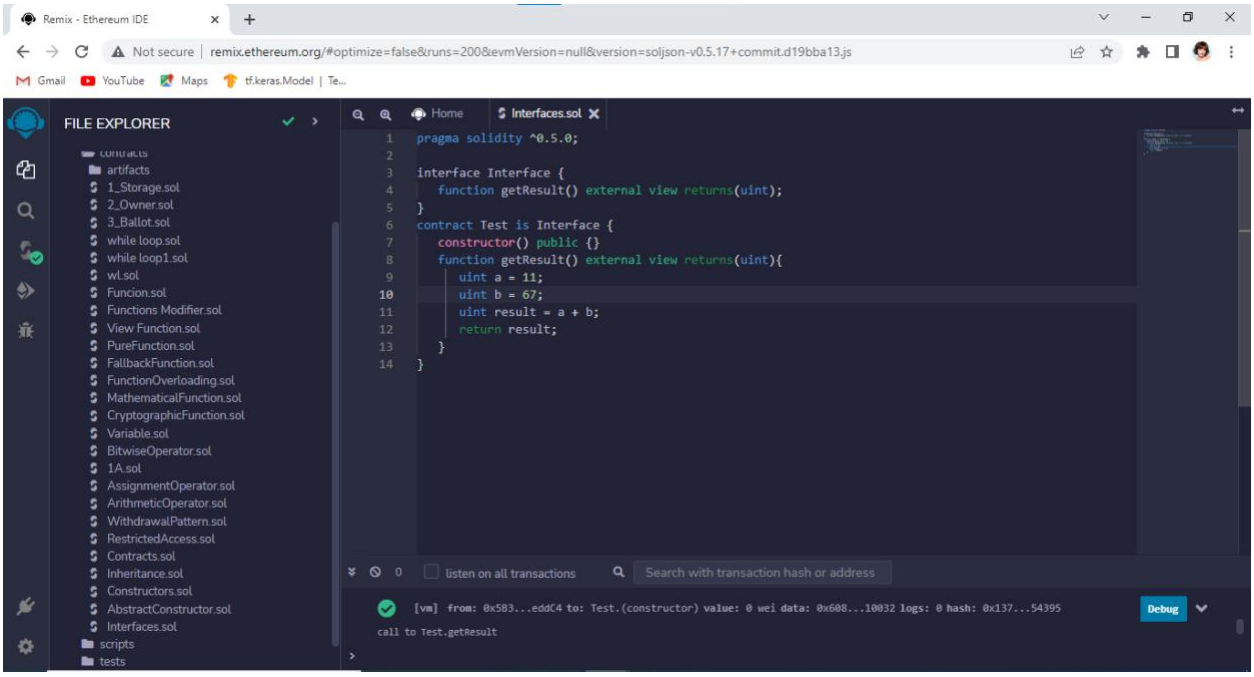


Output:

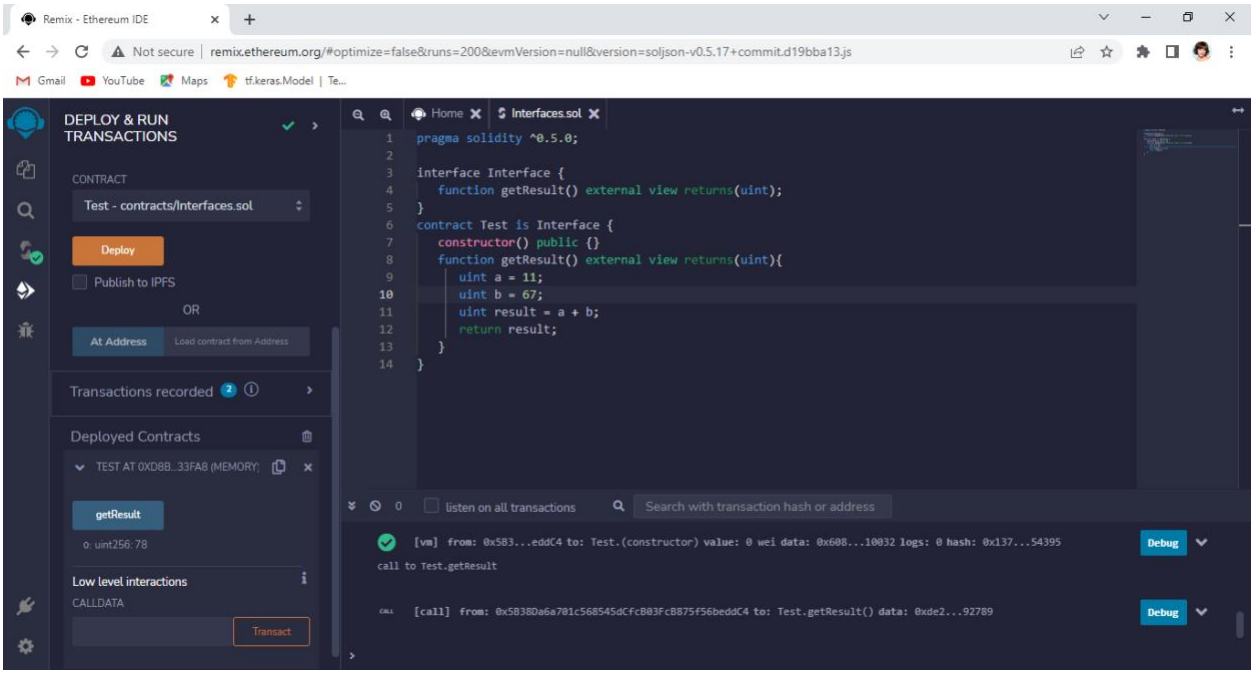


Interfaces

Code:



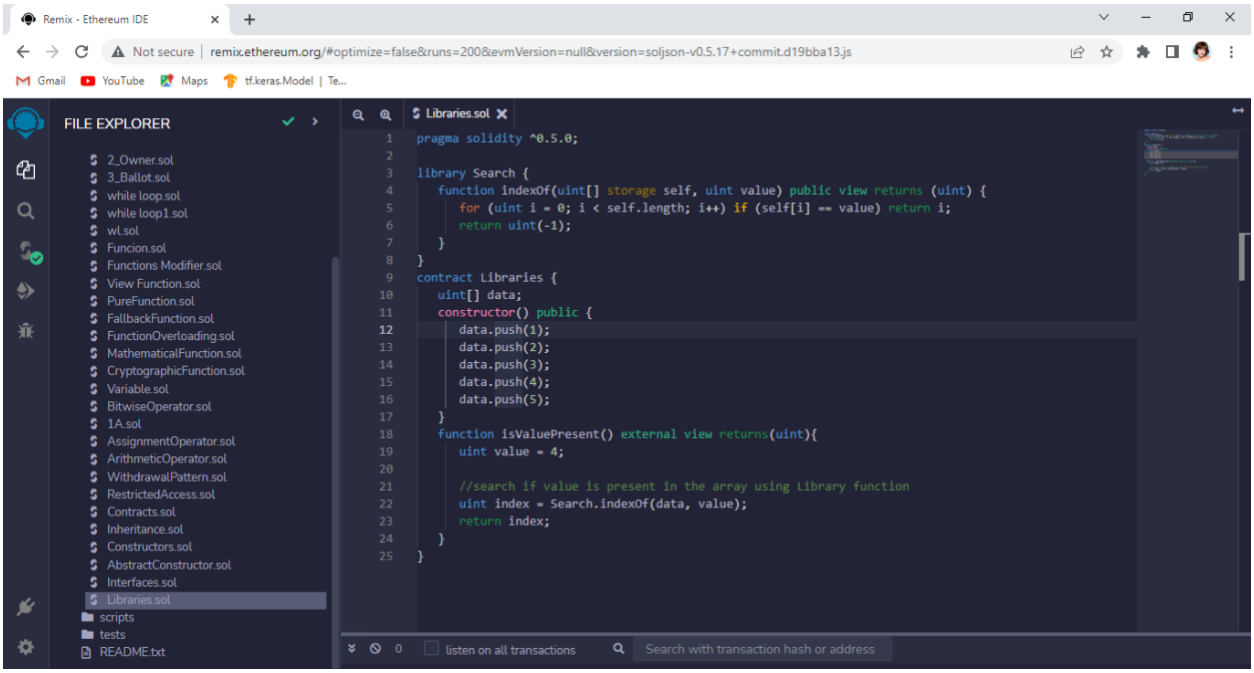
Output:



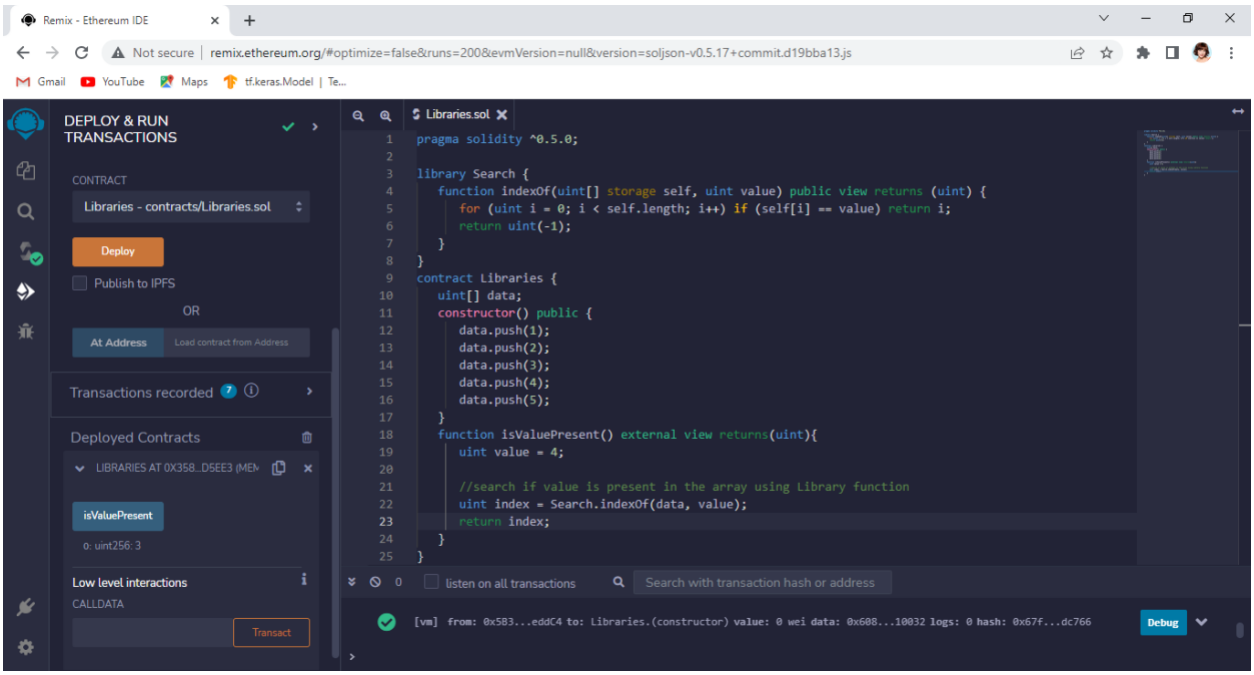
3c) Libraries, Assembly, Events, Error handling.

Libraries

Code:

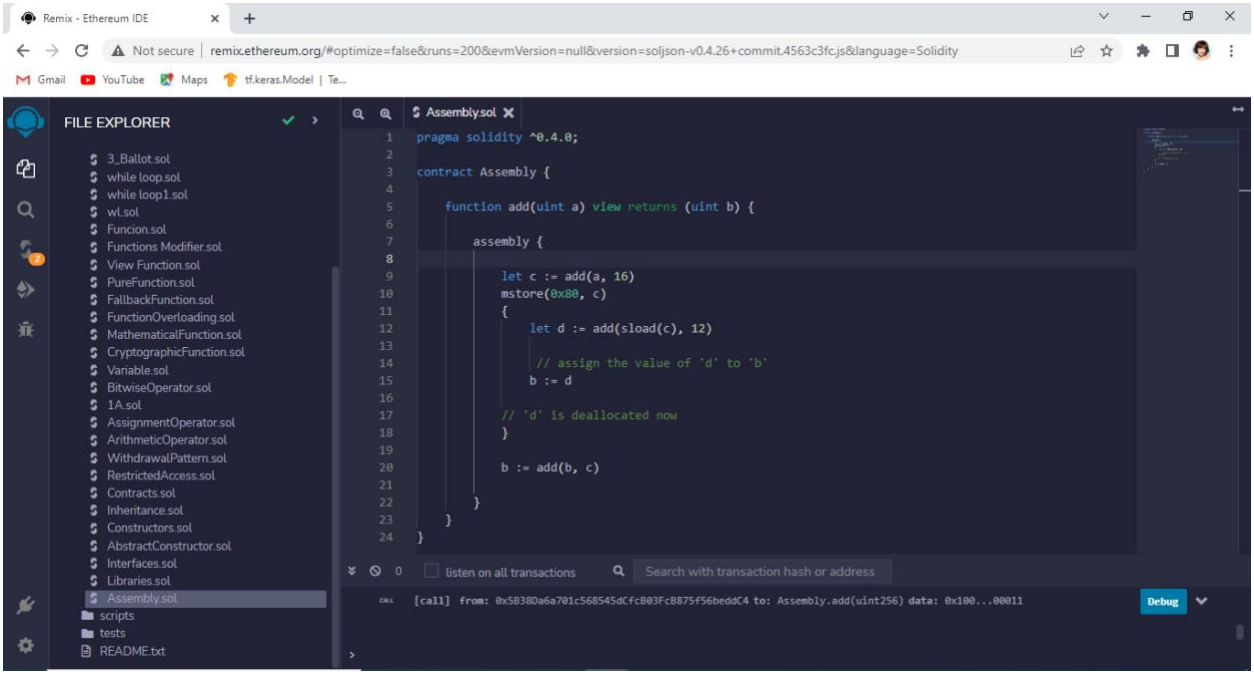


Output:

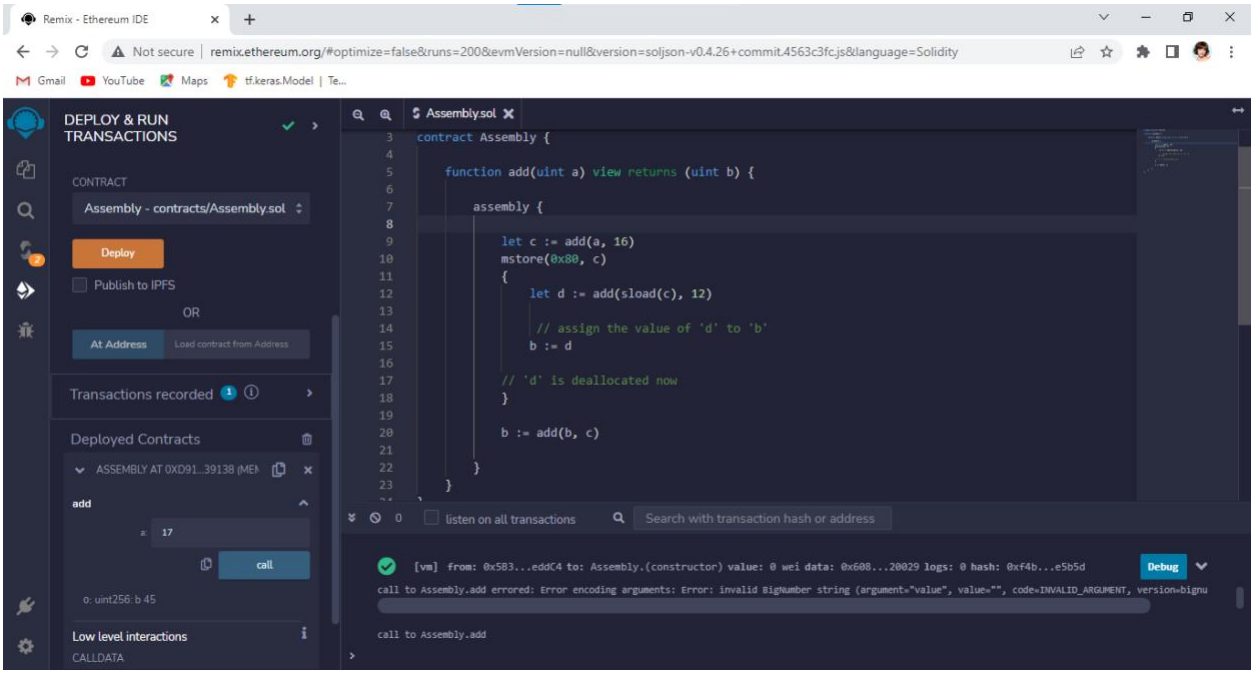


Assembly

Code:

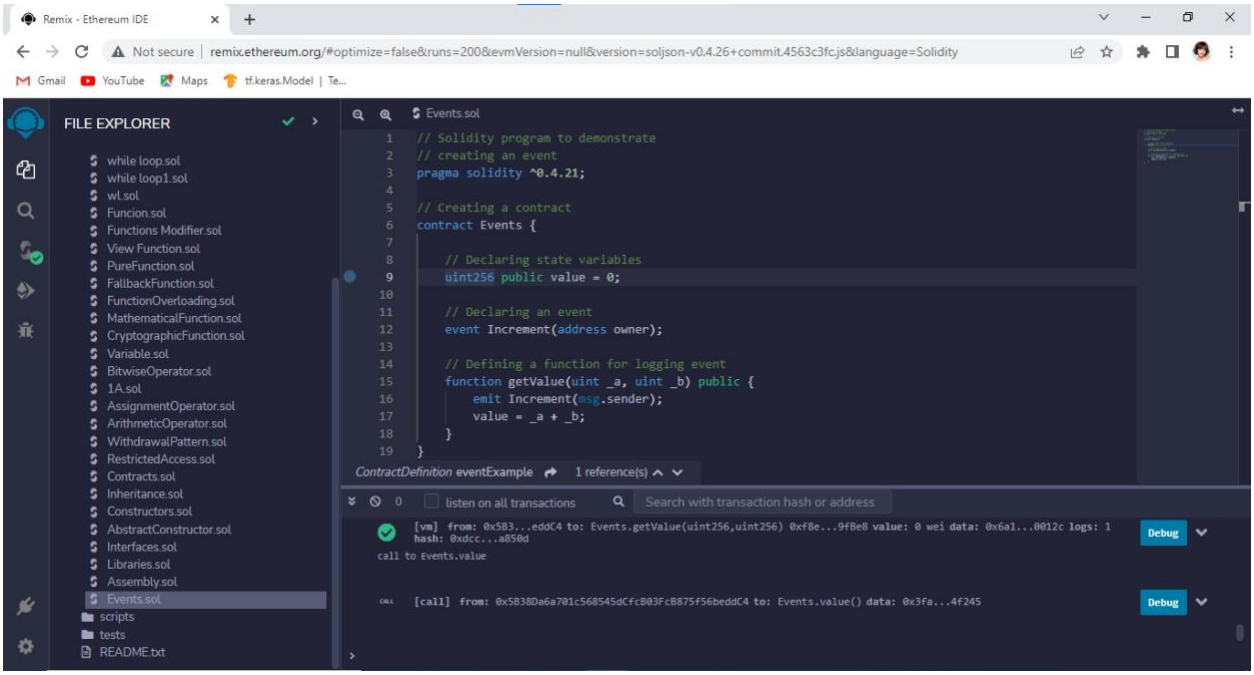


Output:

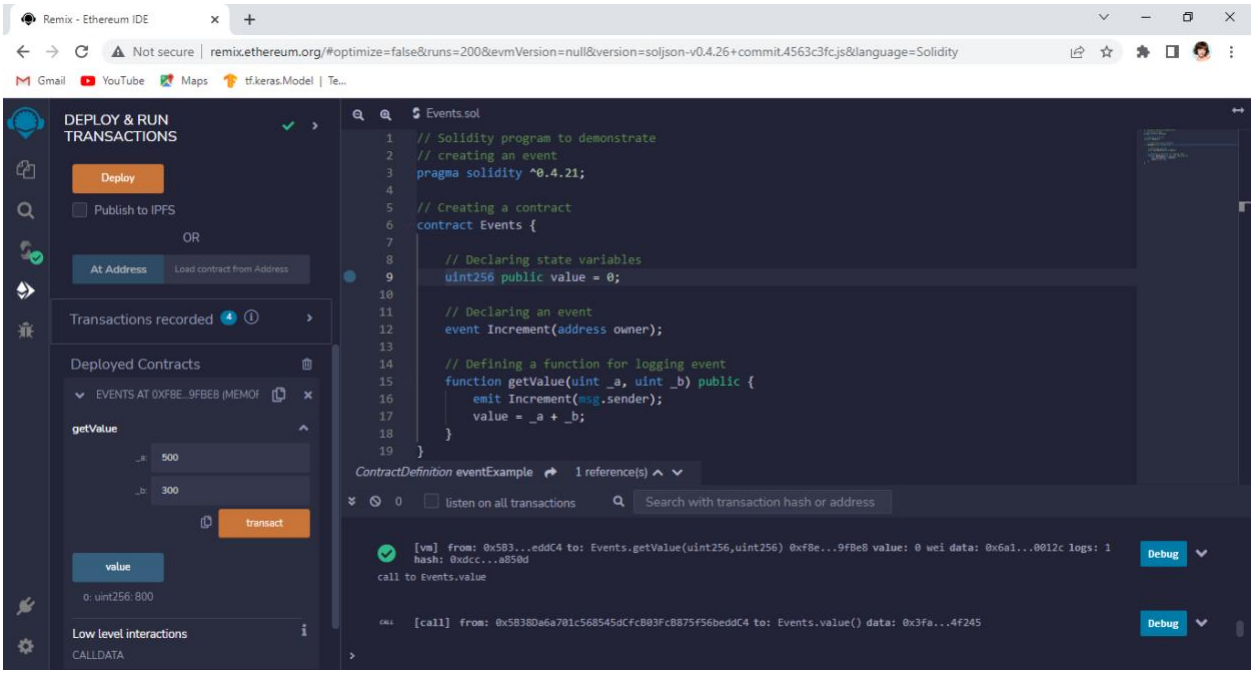


Events

Code:

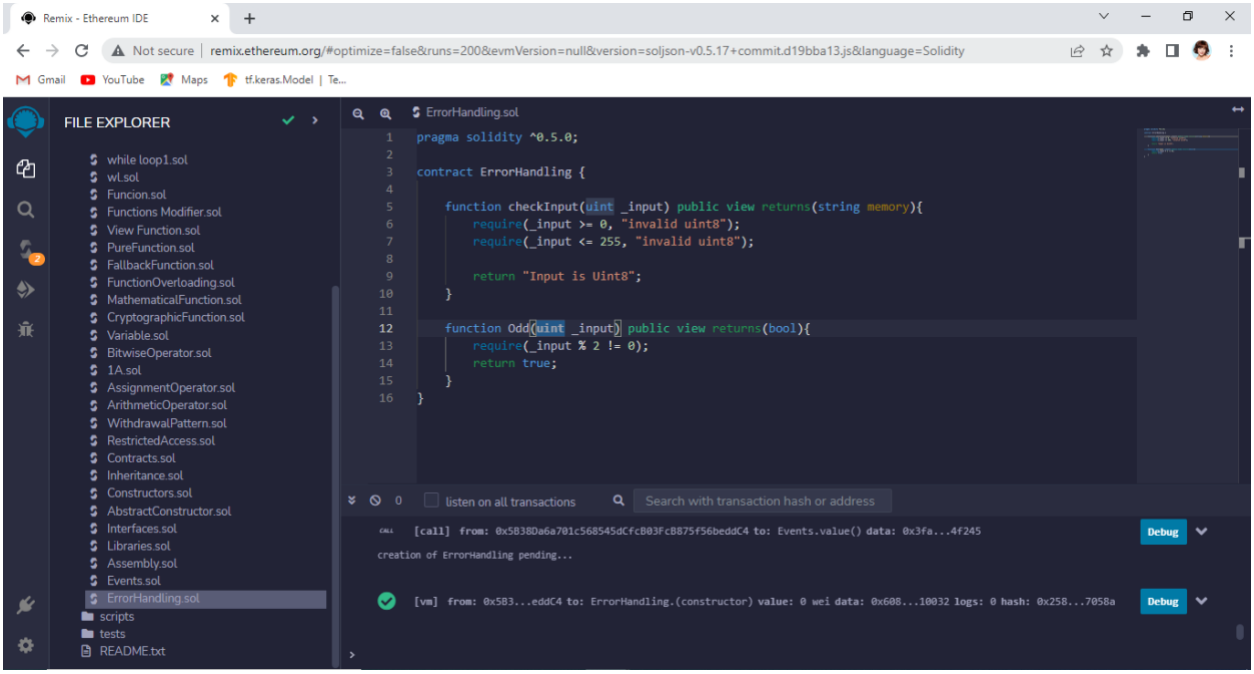


Output:

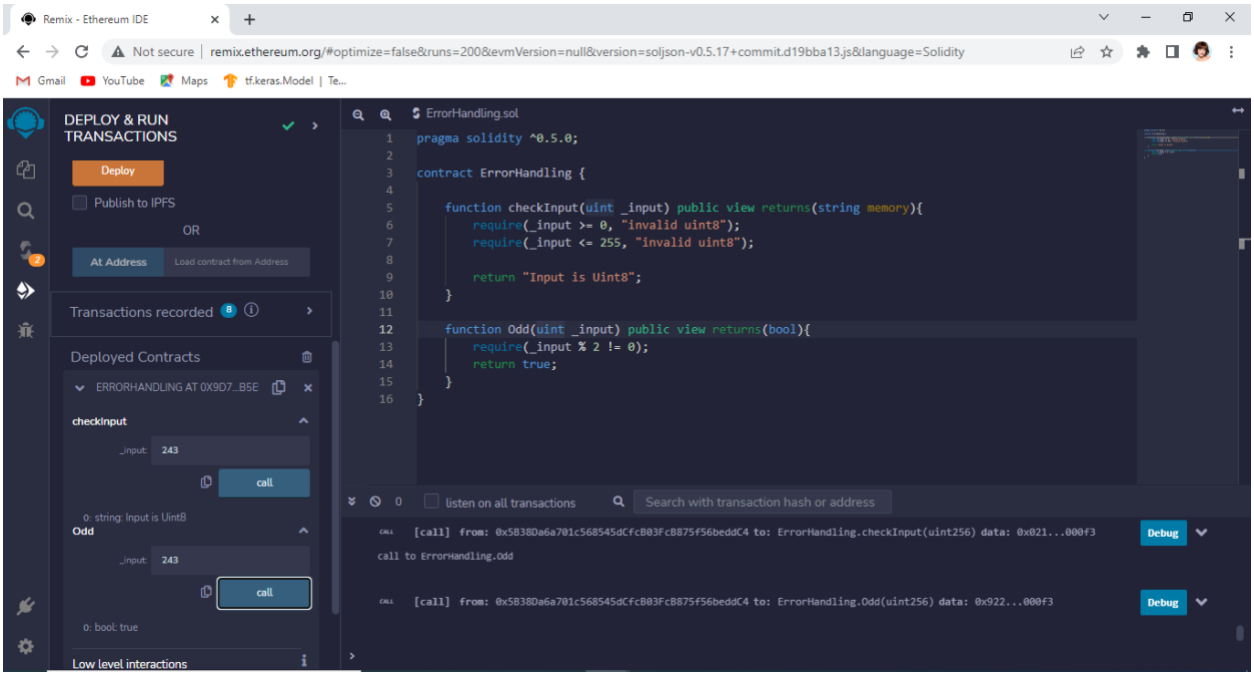


Error Handling

Code:



Output:



Practical No: 4

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

Program Steps:

The following are prerequisites for installing the required development tools:

- Operating Systems: Ubuntu Linux 14.04 / 16.04 LTS (both 64-bit), or Mac OS 10.12
- Docker Engine: Version 17.03 or higher
- Docker-Compose: Version 1.8 or higher
- Node: 8.9 or higher (Note: version 9 is not supported)
- npm: v5.x
- git: 2.9.x or higher
- Python: 2.7.x
- A code editor of your choice, we recommend VSCode.

If installing Hyperledger Composer using Linux, the following points need to be kept in mind:

- Login as a normal user, rather than root.
- Do not use sudo su to root.
- When installing prerequisites, use curl, then unzip using sudo.
- Run prereqs-ubuntu.sh as a normal user. It may prompt for root password as some of its actions are required to be run as root.
- Do not use npm with sudo or su to root to use it.
- Avoid installing node globally as root.

Prerequisites

To download prerequisites for Hyperledger Fabric development, run following command –

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

This command will download and install -

| docker-compose , docker-engine, npm, git, python etc

- Give permissions — chmod u+x prereqs-ubuntu.sh
- Run Script — ./prereqs-ubuntu.sh (restart system after it)
- Essential CLI tools — npm install -g composer-cli@0.20

Steps

1. **Create a directory — mkdir multichain_network**

```
cd multichain_network

curl -sSL http://bit.ly/2ysbOFE | bash -s 1.2.0

export PATH=<path to download location>/multichain_network/fabric-
samples/first-network/bin:$PATH
```

2. **Generate Certificates**

```
cd first-network

export FABRIC_CFG_PATH=$PWD

cryptogen generate — config=./crypto-config.yaml
```

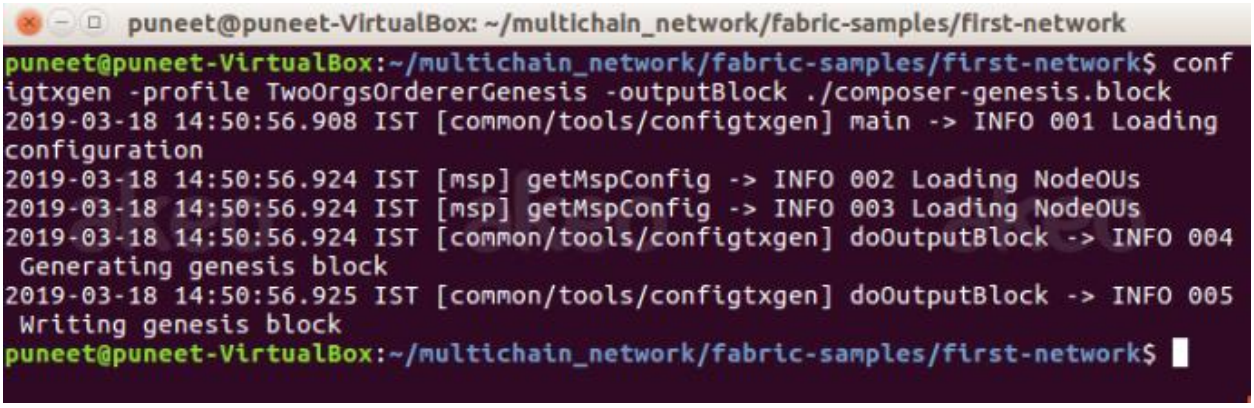
This will create all certificates for orderers and peers in crypto-config folder.

Copy certificates for all peers and orderer to temporary folder.

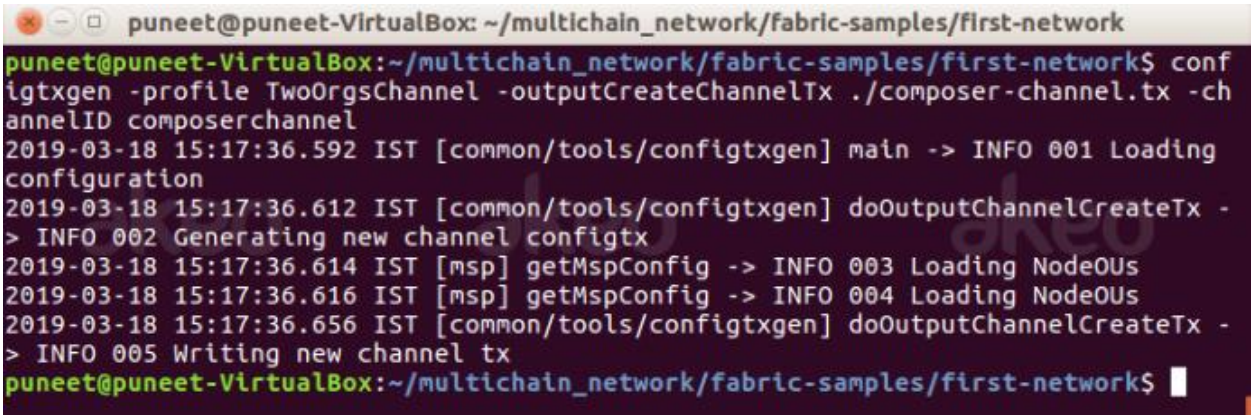
```
In first-network folder run this command — mkdir -p tmp/composer/org1

awk 'NF {sub(/\r/, ""); printf "%s\\n",$0;}' ./crypto-
config/peerOrganizations/org1.example.com/peers/peer0.org1.example.com
/tls/ca.crt > ./tmp/composer/org1/ca-org1.txt

awk 'NF {sub(/\r/, ""); printf "%s\\n",$0;}' ./crypto-
config/ordererOrganizations/example.com/orderers/orderer.example.com/t
ls/ca.crt > ./tmp/composer/ca-orderer.txt

export ORG1=./crypto-
config/peerOrganizations/org1.example.com/users/Admin@org1.example.com
/msp

cp -p $ORG1/signcerts/A*.pem ./tmp/composer/org1

cp -p $ORG1/keystore/*_sk ./tmp/composer/org1
```

3. **Create genesis block and channeltx**

```
configtxgen -profile TwoOrgsOrdererGenesis -outputBlock ./composer-genesis.block
```




```
configtxgen -profile TwoOrgsChannel -outputCreateChannelTx ./composer-channel.tx -channelID composerchannel
```



4. Update CA keys in docker composer file

Open project in code editor.

We have to change CA keys in “docker-compose-e2e-template.yaml” file, therefore navigate to this file in vscode.

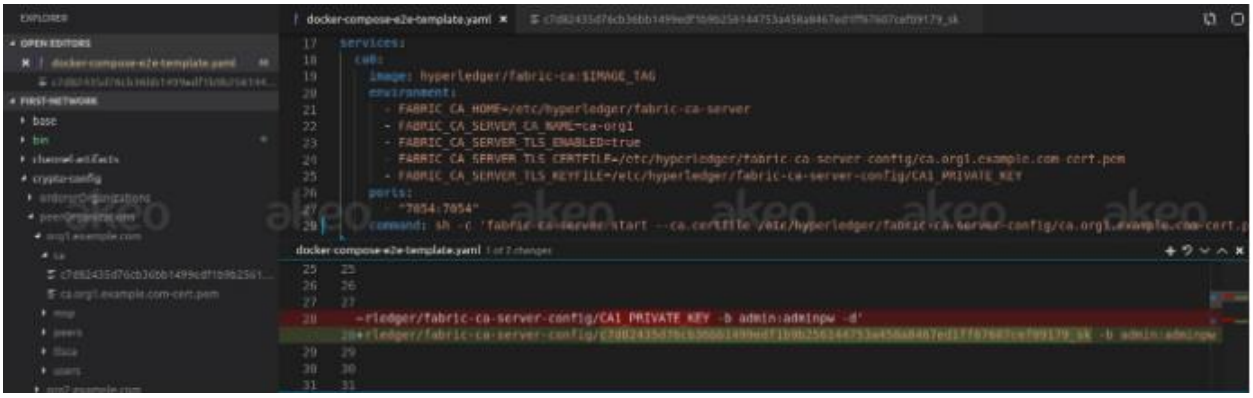
Under services section we have two certificate authorities named “ca0” and “ca1”.

For ca0 — go to command section under ca0 and find CA1_PRIVATE_KEY and replace it with private key –

```
C7d82435d76cb36bb1499edf1b9b256144753a458a8467ed1ff67607cef09179_sk
```

This key can be found in –

```
"first-network/crypto-config/peerOrganizations/org1.example.com/ca/c7d82435d76cb36bb1499edf1b9b256144753a458a8467ed1ff67607cef09179_sk"
```

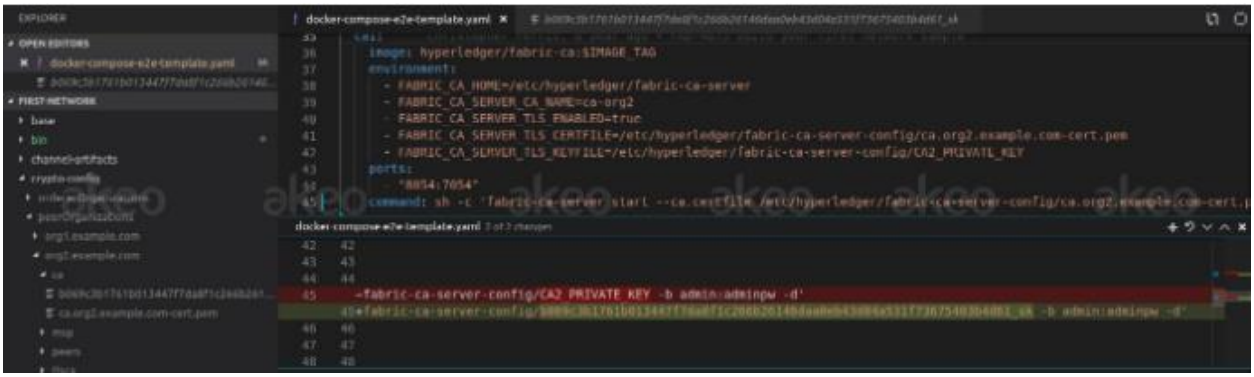


For ca1 — goto command section under ca1 and find CA2_PRIVATE_KEY and replace it with private key –

```
B069c3b1761b013447f7da8f1c266b26146daa0eb43d04a531f73675403b4d61_sk
```

This key can be found in –

```
"first-network/crypto-config/peerOrganizations/org1.example.com/ca/b069c3b1761b013447f7da8f1c266b26146daa0eb43d04a531f73675403b4d61_sk
```



Steps of Starting Hyperledger Fabric

1. Open docker-compose-base.yaml file which is present in the bin folder and introduce following changes.

```
Change orderer volume binding to -

../composer-
genesis.block:/var/hyperledger/orderer/orderer.genesis.block
```

As shown in the figure below –

```
orderer.example.com:
  container_name: orderer.example.com
  image: hyperledger/fabric-orderer:$IMAGE_TAG
  environment:
    - ORDERER_GENERAL_LOGLEVEL=INFO
    - ORDERER_GENERAL_LISTENADDRESS=0.0.0.0
    - ORDERER_GENERAL_GENESISMETHOD=file
    - ORDERER_GENERAL_GENESISFILE=/var/hyperledger/orderer/orderer.genesis.block
    - ORDERER_GENERAL_LOCALMSPID=OrdererMSP
    - ORDERER_GENERAL_LOCALMSPDIR=/var/hyperledger/orderer/msp
    # enabled TLS
    - ORDERER_GENERAL_TLS_ENABLED=true
    - ORDERER_GENERAL_TLS_PRIVATEKEY=/var/hyperledger/orderer/tls/server.key
    - ORDERER_GENERAL_TLS_CERTIFICATE=/var/hyperledger/orderer/tls/server.crt
    - ORDERER_GENERAL_TLS_ROOTCAS=[/var/hyperledger/orderer/tls/ca.crt]
  working_dir: /opt/gopath/src/github.com/hyperledger/fabric
  command: orderer
  volumes:
  - ../composer-genesis.block:/var/hyperledger/orderer/orderer.genesis.block
  - ../crypto-config/ordererOrganizations/example.com/orderers/orderer.example.com/msp:/var/hyperledger/orderer/msp
  - ../crypto-config/ordererOrganizations/example.com/orderers/orderer.example.com/tls/:/var/hyperledger/orderer/tls
  - orderer.example.com:/var/hyperledger/production/orderer
  ports:
    - 7050:7050
```

Change peer volume binding to (for all 4 peers)-

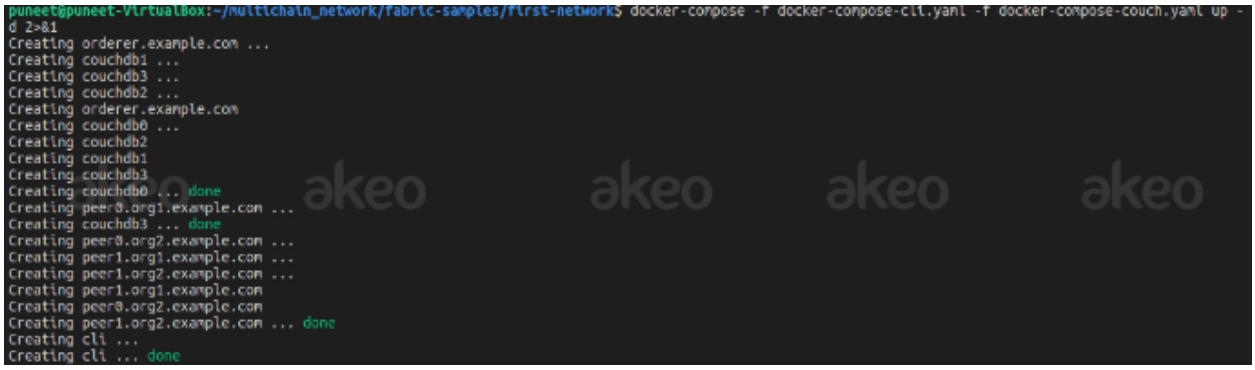
```
- ../:/etc/configtx
```

```
peer0.org1.example.com:
  extends:
    file: peer-base.yaml
    service: peer-base
  environment:
    - CORE_PEER_ID=peer0.org1.example.com
    - CORE_PEER_ADDRESS=peer0.org1.example.com:7051
    - CORE_PEER_GOSSIP_BOOTSTRAP=peer1.org1.example.com:7051
    - CORE_PEER_GOSSIP_EXTERNALENDPOINT=peer0.org1.example.com:7051
    - CORE_PEER_LOCALMSPID=Org1MSP
  volumes:
      - ../:/etc/configtx
      - /var/run/:/host/var/run/
      - ../crypto-config/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/msp:/etc/hyperledger/fabric/msp
      - ../crypto-config/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls:/etc/hyperledger/fabric/tls
      - peer0.org1.example.com:/var/hyperledger/production
  ports:
    - 7051:7051
    - 7053:7053
```

2. To start fabric, run the following command –

```
docker-compose -f docker-compose-cli.yaml -f docker-compose-couch.yaml
up -d 2>&1
```

The output of the above command is shown below –



3. After completing all these steps, you can run command –

```
docker ps -a
```

This will list all running containers regarding our network setup, in our case it will list 10 containers.

Proposed network setup is complete, our network have -

- One orderer
- Two Organizations
- Four peers (two peers on each organization)

- Couchdb for all peers

4. Creating channel

```
docker exec peer0.org1.example.com peer channel create -o
orderer.example.com:7050 -c composerchannel -f /etc/configtx/composer-
channel.tx — tls true — cafile /etc/configtx/crypto-
config/ordererOrganizations/example.com/orderers/orderer.example.com/m
sp/tlscacerts/tlsca.example.com-cert.pem
```

5. Joining first peer where channel is created —

```
docker exec -e

"CORE_PEER_MSPCONFIGPATH=/etc/configtx/tmp/composer/org1/Admin@org1.ex
ample.com/msp" peer0.org1.example.com peer channel join -b composer-
genesis.block
```

6. For other peers, we have to first fetch the config of block from orderer

   For Fetching –

```
docker exec -e

"CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/msp/users/Admin@org1.example
.com/msp"

peer1.org1.example.com peer channel fetch config -o

orderer.example.com:7050 -c TwoOrgsChannel
```

   For joining channel –

```
docker exec -e

"CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/msp/users/Admin@org1.example
.com/msp"

peer1.org1.example.com peer channel join -b
composerchannel_config.block
```

Install chainCode on every machine

```
composer network install -a device-network.bna -c

PeerAdmin@multi_org1
```

Start chaincode on one machine

```
composer network start -n device-network -V 0.0.2-
deploy.${bna_deployment_version} -A admin -S adminpw -c
PeerAdmin@multi_org1
```

Conclusion

We learnt about installing Hyperledger Fabric development tools and Hyperledger Composer using Linux. We also successfully deployed a Hyperledger Fabric network having one orderer, two organizations and two peers in each organization.

Practical No: 5

Aim: Demonstrate the running of the blockchain node.

Refer Practical No: 1, 2 3

Practical No: 6

Aim: Demonstrate the use of Bitcoin Core API.

Code:

```
from hashlib import sha256
MAX_NONCE = 100000000000

def SHA256(text):
    return sha256(text.encode("ascii")).hexdigest()

def mine(block_number, transactions, previous_hash, prefix_zeros):
    prefix_str = '0'*prefix_zeros
    for nonce in range(MAX_NONCE):
        text = str(block_number) + transactions + previous_hash + str(nonc
e)
        new_hash = SHA256(text)
        if new_hash.startswith(prefix_str):
            print(f"Yay! Successfully mined bitcoins with nonce value:{non
ce}")
            return new_hash

    raise BaseException(f"Couldn't find correct has after trying {MAX_NONC
E} times")

if __name__=='__main__':
    transactions='''
    Dhaval->Bhavin->20,
    Mando->Cara->45
    '''
    difficulty=4 # try changing this to higher number and you will see it
will take more time for mining as difficulty increases
    import time
    start = time.time()
    print("start mining")
```

```
    new_hash = mine(5,transactions,'0000000xa036944e29568d0cff17edbe038f81
208fecf9a66be9a2b8321c6ec7', difficulty)
    total_time = str((time.time() - start))
    print(f"end mining. Mining took: {total_time} seconds")
    print(new_hash)
```

Output:

```
start mining
Yay! Successfully mined bitcoins with nonce value:2425
end mining. Mining took: 0.011358022689819336 seconds
0000de957fbfdfc77582e0d0b20c53d2d1d83d8bb8cfe3693521f672bf2a6021
```

Practical No: 7

Aim: Create your own blockchain and demonstrate its use.

Refer Practical No: 1, 2 3

Practical No	Title
1	Performing matrix multiplication and finding eigen vectors and eigen values using TensorFlow
2	Solving XOR problem using deep feed forward network.
3	Implementing deep neural network for performing binary classification task.
4	a) Aim: Using deep feed forward network with two hidden layers for performing multiclass classification and predicting the class. b) Aim: Using a deep feed forward network with two hidden layers for performing classification and predicting the probability of class. c) Aim: Using a deep feed forward network with two hidden layers for performing linear regression and predicting values.
5	a)Evaluating feed forward deep network for regression using KFold cross validation. b)Evaluating feed forward deep network for multiclass Classification using KFold cross-validation.
6	Implementing regularization to avoid overfitting in binary classification.
7	Demonstrate recurrent neural network that learns to perform sequence analysis for stock price.
8	Performing encoding and decoding of images using deep autoencoder.
9	Implementation of convolutional neural network to predict numbers from number images
10	Denoising of images using autoencoder.

Practical No:1

Aim: Performing matrix multiplication and finding eigen vectors and eigen values using TensorFlow.

```
import tensorflow as tf

print("Matrix Multiplication Demo")

x=tf.constant([1,2,3,4,5,6],shape=[2,3])

print(x)

y=tf.constant([7,8,9,10,11,12],shape=[3,2])

print(y)

z=tf.matmul(x,y)

print("Product:",z)

e_matrix_A=tf.random.uniform([2,2],minval=3,maxval=10,dtype=tf.float32,name="matrixA")

print("Matrix A:\n{}\n\n".format(e_matrix_A))

eigen_values_A,eigen_vectors_A=tf.linalg.eigh(e_matrix_A)

print("Eigen Vectors:\n{}\n\nEigen Values:\n{}\n".format(eigen_vectors_A,eigen_values_A))
```

OUTPUT:

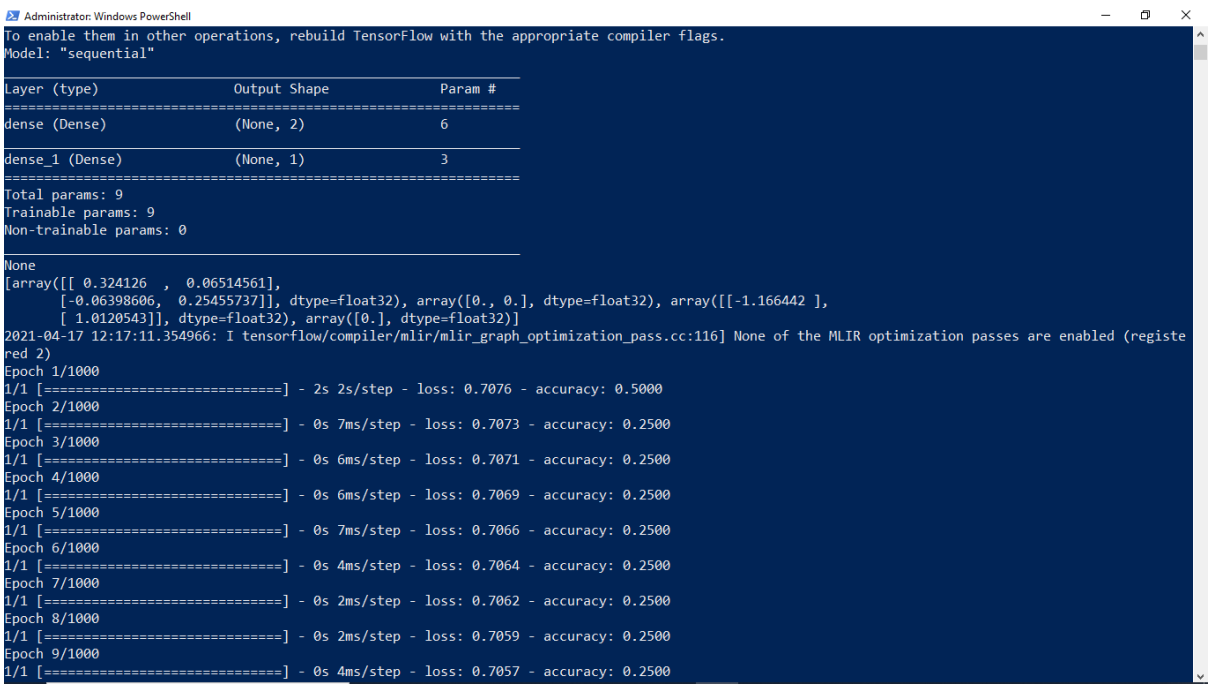
```
tf.Tensor(
[[1 2 3]
 [4 5 6]], shape=(2, 3), dtype=int32)
tf.Tensor(
[[ 7  8]
 [ 9 10]
 [11 12]], shape=(3, 2), dtype=int32)
Product: tf.Tensor(
[[ 58  64]
 [139 154]], shape=(2, 2), dtype=int32)
Matrix A:
[[7.791751  6.3527837]
 [6.8659496 5.229142 ]]


Eigen Vectors:
[[-0.63896394  0.7692366 ]
 [ 0.7692366   0.63896394]]

Eigen Values:
[-0.47403672 13.494929  ]

(venv) PS D:\keras>
```

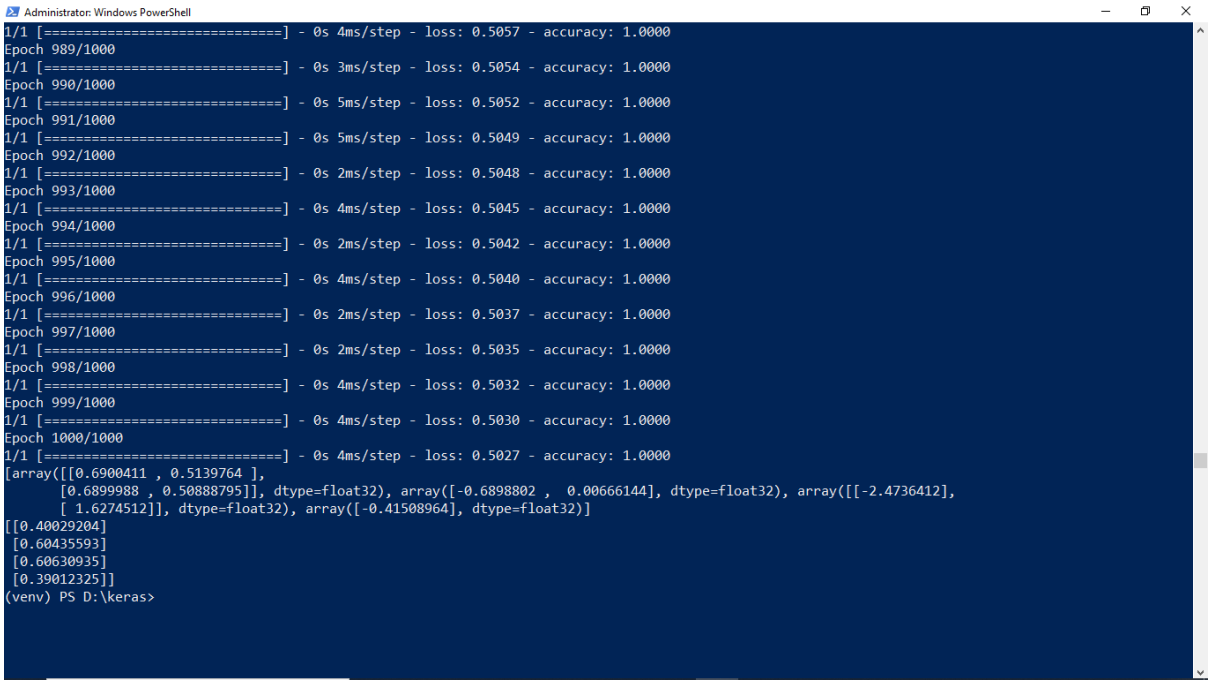
Practical No:2

Aim: Solving XOR problem using deep feed forward network.

```
import numpy as np
from keras.layers import Dense
from keras.models import Sequential
model=Sequential()
model.add(Dense(units=2,activation='relu',input_dim=2))
model.add(Dense(units=1,activation='sigmoid'))
model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])
print(model.summary())
print(model.get_weights())
X=np.array([[0.,0.],[0.,1.],[1.,0.],[1.,1.]])
Y=np.array([0.,1.,1.,0.])
model.fit(X,Y,epochs=1000,batch_size=4)
print(model.get_weights())
print(model.predict(X,batch_size=4))
```

OUTPUT:



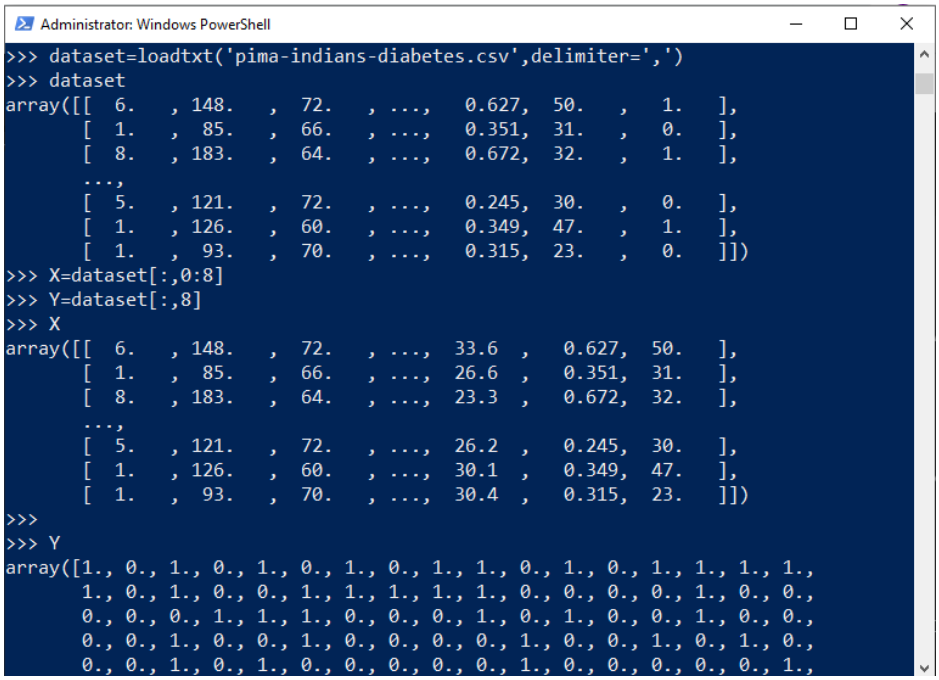


Practical No:3

Aim: Implementing deep neural network for performing classification task.

Problem statement: the given dataset comprises of health information about diabetic women patient. we need to create deep feed forward network that will classify women suffering from diabetes mellitus as 1.

```
>>> from numpy import loadtxt
>>> from keras.models import Sequential
>>> from keras.layers import Dense
>>>
```



Creating model:

```
>>> model=Sequential()
```

```
>>> model.add(Dense(12,input_dim=8,activation='relu'))
>>> model.add(Dense(8,activation='relu'))
>>> model.add(Dense(1,activation='sigmoid'))
>>>
```

Compiling and fitting model:

```
>>> model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])
>>> model.fit(X,Y,epochs=150,batch_size=10)
```

```
Administrator: Windows PowerShell
>>> model.add(Dense(1,activation='sigmoid'))
>>> model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])
>>> model.fit(X,Y,epochs=150,batch_size=10)
2021-04-05 17:40:32.289557: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:116] None of the MLIR optimization passes are enabled (registe
red 2)
Epoch 1/150
77/77 [==============================] - 2s 2ms/step - loss: 2.6770 - accuracy: 0.4399
Epoch 2/150
77/77 [==============================] - 0s 1ms/step - loss: 1.1332 - accuracy: 0.5064
Epoch 3/150
77/77 [==============================] - 0s 2ms/step - loss: 0.8624 - accuracy: 0.5592
Epoch 4/150
77/77 [==============================] - 0s 2ms/step - loss: 0.8135 - accuracy: 0.5700
Epoch 5/150
77/77 [==============================] - 0s 2ms/step - loss: 0.7369 - accuracy: 0.6089
Epoch 6/150
77/77 [==============================] - 0s 1ms/step - loss: 0.7405 - accuracy: 0.6269
Epoch 7/150
77/77 [==============================] - 0s 2ms/step - loss: 0.7157 - accuracy: 0.6060
Epoch 8/150
77/77 [==============================] - 0s 1ms/step - loss: 0.6852 - accuracy: 0.6354
Epoch 9/150
77/77 [==============================] - 0s 2ms/step - loss: 0.6585 - accuracy: 0.6398
Epoch 10/150
77/77 [==============================] - 0s 2ms/step - loss: 0.6524 - accuracy: 0.6330
Epoch 11/150
77/77 [==============================] - 0s 2ms/step - loss: 0.6671 - accuracy: 0.6584
Epoch 12/150
77/77 [==============================] - 0s 2ms/step - loss: 0.6216 - accuracy: 0.6857
Epoch 13/150
77/77 [==============================] - 0s 2ms/step - loss: 0.6656 - accuracy: 0.6469
Epoch 14/150
77/77 [==============================] - 0s 2ms/step - loss: 0.6304 - accuracy: 0.6870
Epoch 15/150
77/77 [==============================] - 0s 2ms/step - loss: 0.6290 - accuracy: 0.6594
Epoch 16/150
77/77 [==============================] - 0s 2ms/step - loss: 0.6033 - accuracy: 0.6722
```

Evaluating the accuracy:

```
>>> _,accuracy=model.evaluate(X,Y)
24/24 [==============================] - 0s 1ms/step - loss: 0.4849 - accuracy: 0
.7591
>>> print('Accuracy of model is',(accuracy*100))
Accuracy of model is 75.91145634651184
>>>
```

Using model for prediction class:

```
>>> prediction=model.predict_classes(X)
```

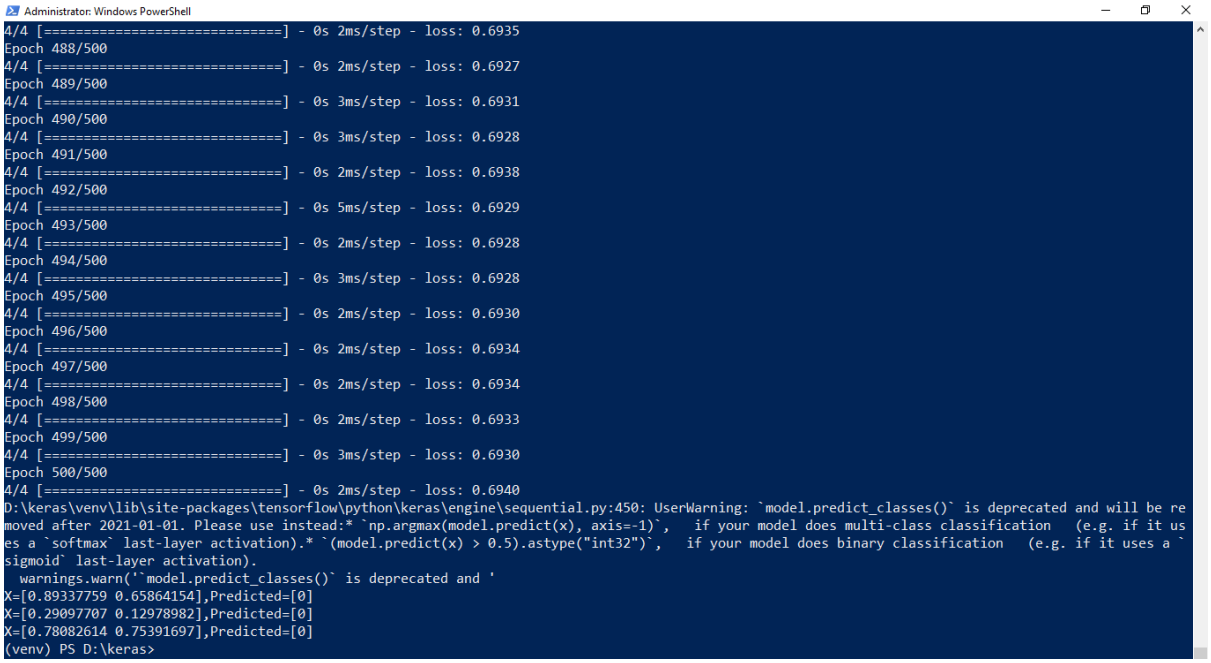
```
>>> exec("for i in range(5):print(X[i].tolist(),prediction[i],Y[i])")
[6.0, 148.0, 72.0, 35.0, 0.0, 33.6, 0.627, 50.0] [1] 1.0
[1.0, 85.0, 66.0, 29.0, 0.0, 26.6, 0.351, 31.0] [0] 0.0
[8.0, 183.0, 64.0, 0.0, 0.0, 23.3, 0.672, 32.0] [1] 1.0
[1.0, 89.0, 66.0, 23.0, 94.0, 28.1, 0.167, 21.0] [0] 0.0
[0.0, 137.0, 40.0, 35.0, 168.0, 43.1, 2.288, 33.0] [1] 1.0
>>>
```

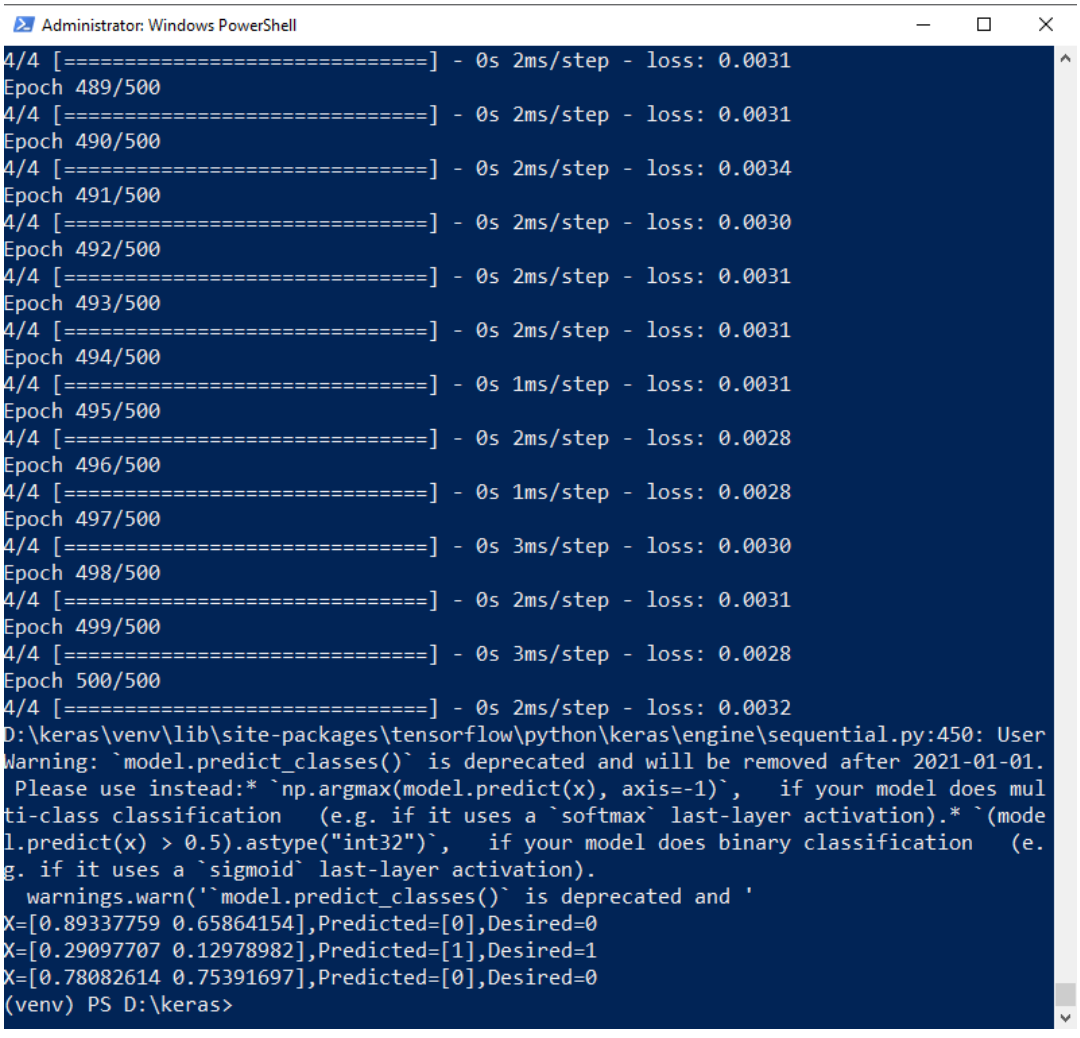
Practical No:4

a) Aim: Using deep feed forward network with two hidden layers for performing classification and predicting the class.

```
from keras.models import Sequential
from keras.layers import Dense
from sklearn.datasets import make_blobs
from sklearn.preprocessing import MinMaxScaler

X,Y=make_blobs(n_samples=100,centers=2,n_features=2,random_state=1)
scalar=MinMaxScaler()
scalar.fit(X)
X=scalar.transform(X)

model=Sequential()
model.add(Dense(4,input_dim=2,activation='relu'))
model.add(Dense(4,activation='relu'))
model.add(Dense(1,activation='sigmoid'))
model.compile(loss='binary_crossentropy',optimizer='adam')
model.fit(X,Y,epochs=500)

Xnew,Yreal=make_blobs(n_samples=3,centers=2,n_features=2,random_state=1)
Xnew=scalar.transform(Xnew)

Ynew=model.predict_classes(Xnew)
for i in range(len(Xnew)):
    print("X=%s,Predicted=%s,Desired=%s"%(Xnew[i],Ynew[i],Yreal[i]))
```

OUTPUT:

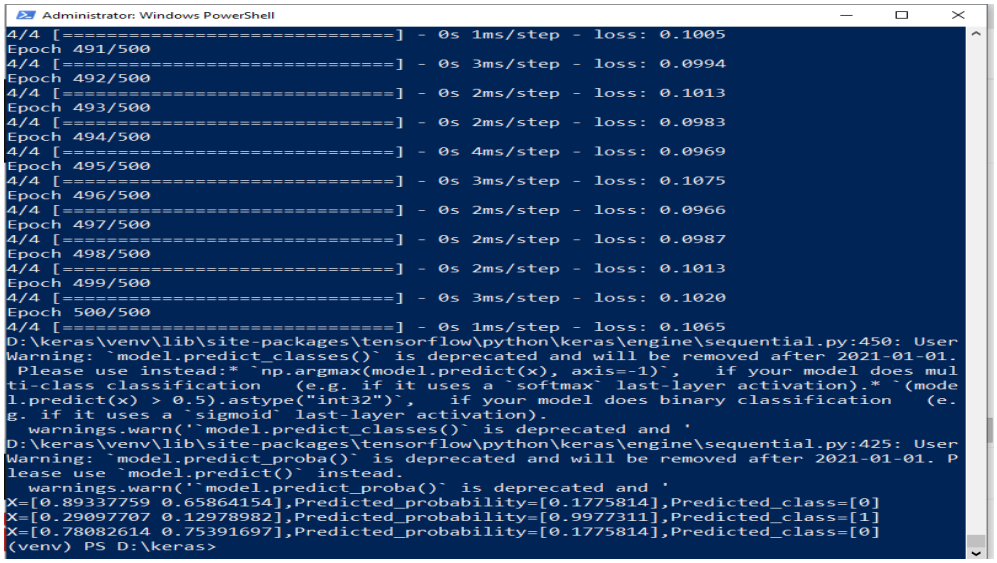

Administrator: Windows PowerShell
4/4 [==============================] - 0s 2ms/step - loss: 0.6935
Epoch 488/500
4/4 [==============================] - 0s 2ms/step - loss: 0.6927
Epoch 489/500
4/4 [==============================] - 0s 3ms/step - loss: 0.6931
Epoch 490/500
4/4 [==============================] - 0s 3ms/step - loss: 0.6928
Epoch 491/500
4/4 [==============================] - 0s 2ms/step - loss: 0.6938
Epoch 492/500
4/4 [==============================] - 0s 5ms/step - loss: 0.6929
Epoch 493/500
4/4 [==============================] - 0s 2ms/step - loss: 0.6928
Epoch 494/500
4/4 [==============================] - 0s 3ms/step - loss: 0.6928
Epoch 495/500
4/4 [==============================] - 0s 2ms/step - loss: 0.6930
Epoch 496/500
4/4 [==============================] - 0s 2ms/step - loss: 0.6934
Epoch 497/500
4/4 [==============================] - 0s 2ms/step - loss: 0.6934
Epoch 498/500
4/4 [==============================] - 0s 2ms/step - loss: 0.6933
Epoch 499/500
4/4 [==============================] - 0s 3ms/step - loss: 0.6930
Epoch 500/500
4/4 [==============================] - 0s 2ms/step - loss: 0.6940
D:\keras\venv\lib\site-packages\tensorflow\python\keras\engine\sequential.py:450: UserWarning: `model.predict_classes()` is deprecated and will be removed after 2021-01-01. Please use instead:* `np.argmax(model.predict(x), axis=-1)`, if your model does multi-class classification (e.g. if it uses a `softmax` last-layer activation).* `(model.predict(x) > 0.5).astype("int32")`, if your model does binary classification (e.g. if it uses a `sigmoid` last-layer activation).
warnings.warn('`model.predict_classes()` is deprecated and '
X=[0.89337759 0.65864154],Predicted=[0]
X=[0.29097707 0.12978982],Predicted=[0]
X=[0.78082614 0.75391697],Predicted=[0]
(venv) PS D:\keras>


Administrator: Windows PowerShell
4/4 [==============================] - 0s 2ms/step - loss: 0.0031
Epoch 489/500
4/4 [==============================] - 0s 2ms/step - loss: 0.0031
Epoch 490/500
4/4 [==============================] - 0s 2ms/step - loss: 0.0034
Epoch 491/500
4/4 [==============================] - 0s 2ms/step - loss: 0.0030
Epoch 492/500
4/4 [==============================] - 0s 2ms/step - loss: 0.0031
Epoch 493/500
4/4 [==============================] - 0s 2ms/step - loss: 0.0031
Epoch 494/500
4/4 [==============================] - 0s 1ms/step - loss: 0.0031
Epoch 495/500
4/4 [==============================] - 0s 2ms/step - loss: 0.0028
Epoch 496/500
4/4 [==============================] - 0s 1ms/step - loss: 0.0028
Epoch 497/500
4/4 [==============================] - 0s 3ms/step - loss: 0.0030
Epoch 498/500
4/4 [==============================] - 0s 2ms/step - loss: 0.0031
Epoch 499/500
4/4 [==============================] - 0s 3ms/step - loss: 0.0028
Epoch 500/500
4/4 [==============================] - 0s 2ms/step - loss: 0.0032
D:\keras\venv\lib\site-packages\tensorflow\python\keras\engine\sequential.py:450: UserWarning: `model.predict_classes()` is deprecated and will be removed after 2021-01-01. Please use instead:* `np.argmax(model.predict(x), axis=-1)`, if your model does multi-class classification (e.g. if it uses a `softmax` last-layer activation).* `(model.predict(x) > 0.5).astype("int32")`, if your model does binary classification (e.g. if it uses a `sigmoid` last-layer activation).
warnings.warn('`model.predict_classes()` is deprecated and '
X=[0.89337759 0.65864154],Predicted=[0],Desired=0
X=[0.29097707 0.12978982],Predicted=[1],Desired=1
X=[0.78082614 0.75391697],Predicted=[0],Desired=0
(venv) PS D:\keras>

b) Aim: Using a deep field forward network with two hidden layers for performing classification and predicting the probability of class.

```
from keras.models import Sequential

from keras.layers import Dense

from sklearn.datasets import make_blobs

from sklearn.preprocessing import MinMaxScaler

X,Y=make_blobs(n_samples=100,centers=2,n_features=2,random_state=1)

scalar=MinMaxScaler()

scalar.fit(X)

X=scalar.transform(X)

model=Sequential()

model.add(Dense(4,input_dim=2,activation='relu'))

model.add(Dense(4,activation='relu'))

model.add(Dense(1,activation='sigmoid'))

model.compile(loss='binary_crossentropy',optimizer='adam')

model.fit(X,Y,epochs=500)

Xnew,Yreal=make_blobs(n_samples=3,centers=2,n_features=2,random_state=1)

Xnew=scalar.transform(Xnew)

Yclass=model.predict_classes(Xnew)

Ynew=model.predict_proba(Xnew)

for i in range(len(Xnew)):

    print("X=%s,Predicted_probability=%s,Predicted_class=%s"%(Xnew[i],Ynew[i],Yclass[i]))
```

OUTPUT:



c) Aim: Using a deep field forward network with two hidden layers for performing linear regression and predicting values.

```
from keras.models import Sequential
from keras.layers import Dense
from sklearn.datasets import make_regression
from sklearn.preprocessing import MinMaxScaler

X,Y=make_regression(n_samples=100,n_features=2,noise=0.1,random_state=1)
scalarX,scalarY=MinMaxScaler(),MinMaxScaler()

scalarX.fit(X)
scalarY.fit(Y.reshape(100,1))
X=scalarX.transform(X)
Y=scalarY.transform(Y.reshape(100,1))

model=Sequential()
model.add(Dense(4,input_dim=2,activation='relu'))
model.add(Dense(4,activation='relu'))
model.add(Dense(1,activation='sigmoid'))
model.compile(loss='mse',optimizer='adam')
model.fit(X,Y,epochs=1000,verbose=0)

Xnew,a=make_regression(n_samples=3,n_features=2,noise=0.1,random_state=1)
Xnew=scalarX.transform(Xnew)

Ynew=model.predict(Xnew)

for i in range(len(Xnew)):
    print("X=%s,Predicted=%s"%(Xnew[i],Ynew[i]))
```

OUTPUT:

```
X=[0.29466096 0.30317302],Predicted=[0.18255734]
X=[0.39445118 0.79390858],Predicted=[0.7581165]
X=[0.02884127 0.6208843 ],Predicted=[0.3932857]
(venv) PS D:\keras>
```

Practical No:5(a)

Aim: Evaluating feed forward deep network for regression using KFold cross validation.

```
import pandas as pd
from keras.models import Sequential
from keras.layers import Dense
from keras.wrappers.scikit_learn import KerasRegressor
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import KFold
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
dataframe=pd.read_csv("housing.csv",delim_whitespace=True,header=None)
dataset=dataframe.values
X=dataset[:,0:13]
Y=dataset[:,13]
def wider_model():
    model=Sequential()
    model.add(Dense(15,input_dim=13,kernel_initializer='normal',activation='relu'))
    model.add(Dense(13,kernel_initializer='normal',activation='relu'))
    model.add(Dense(1,kernel_initializer='normal'))
    model.compile(loss='mean_squared_error',optimizer='adam')
    return model
estimators=[]
estimators.append(('standardize',StandardScaler()))
estimators.append(('mlp',KerasRegressor(build_fn=wider_model,epochs=100,batch_size=5)))
pipeline=Pipeline(estimators)
kfold=KFold(n_splits=10)
results=cross_val_score(pipeline,X,Y,cv=kfold)
print("Wider: %.2f (%.2f) MSE" % (results.mean(), results.std()))
```

OUTPUT:

```
Wider: -20.88 (24.29) MSE
(venv) PS D:\keras>
```

(After changing neuron)

model.add(Dense(20, input_dim=13,kernel_initializer='normal',activation='relu'))

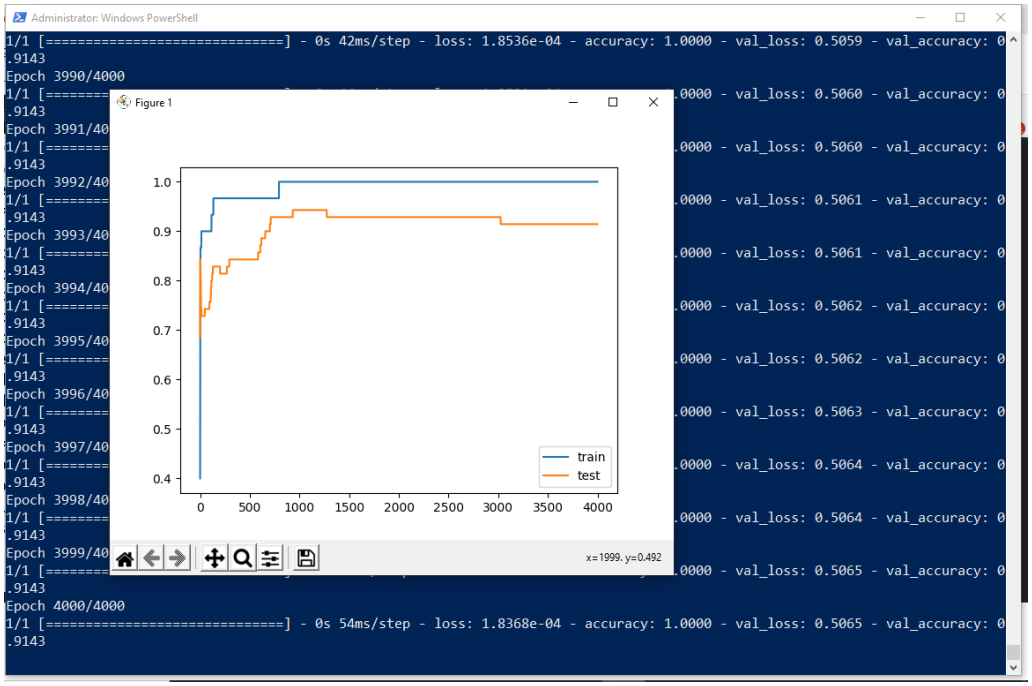
```
Wider: -22.17 (24.38) MSE
(venv) PS D:\keras>
```

Practical No :6

Aim: implementing regularization to avoid overfitting in binary classification.

```
from matplotlib import pyplot
from sklearn.datasets import make_moons
from keras.models import Sequential
from keras.layers import Dense
X,Y=make_moons(n_samples=100,noise=0.2,random_state=1)
n_train=30
trainX,testX=X[:n_train,:],X[n_train:]
trainY,testY=Y[:n_train],Y[n_train:]
#print(trainX)
#print(trainY)
#print(testX)
#print(testY)
model=Sequential()
model.add(Dense(500,input_dim=2,activation='relu'))
model.add(Dense(1,activation='sigmoid'))
model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])
history=model.fit(trainX,trainY,validation_data=(testX,testY),epochs=4000)
pyplot.plot(history.history['accuracy'],label='train')
pyplot.plot(history.history['val_accuracy'],label='test')
pyplot.legend()
pyplot.show()
```

OUTPUT:

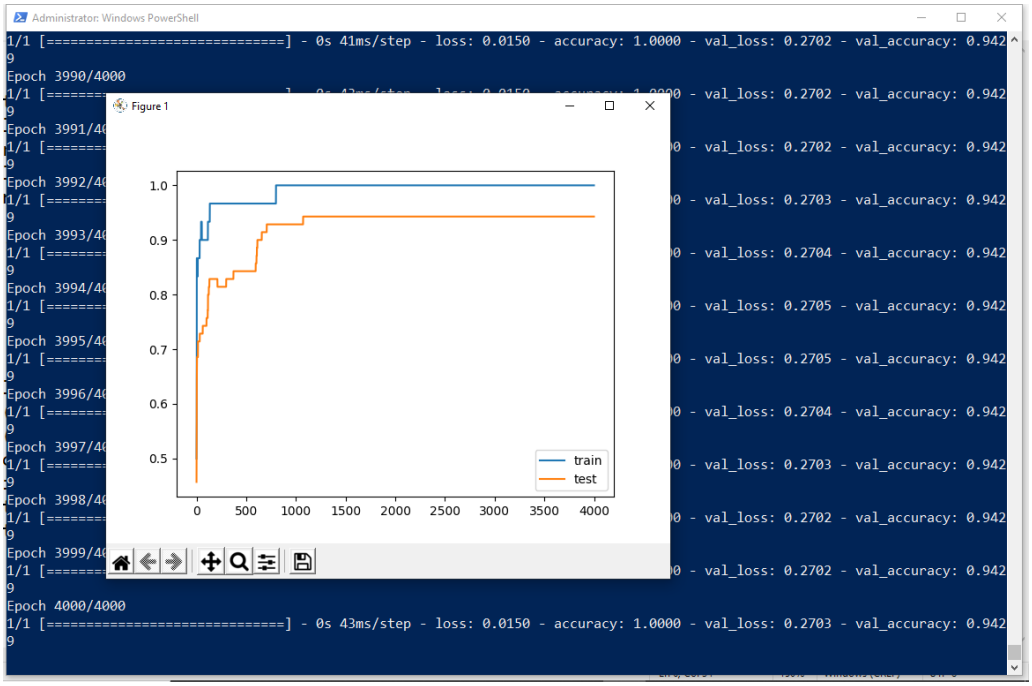


The above code and resultant graph demonstrate overfitting with accuracy of testing data less than accuracy of training data also the accuracy of testing data increases once and then start decreases gradually.to solve this problem we can use regularization

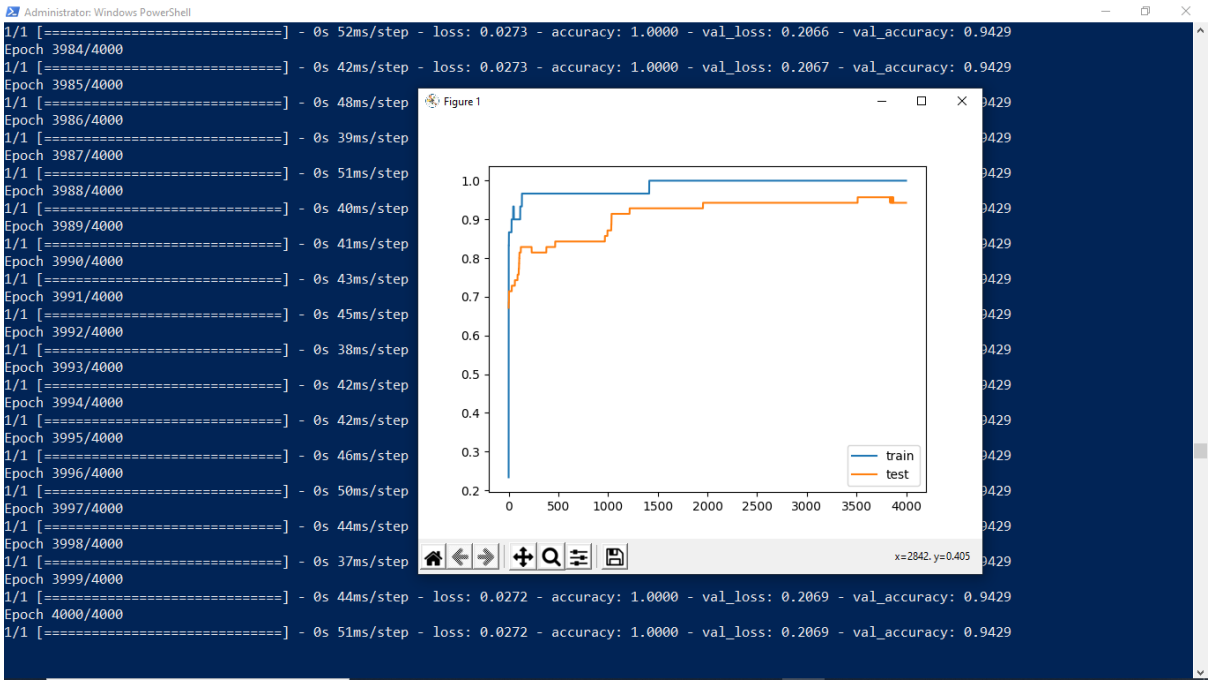
Hence, we will add two lines in the above code as highlighted below to implement l2 regularization with alpha=0.001

```
from matplotlib import pyplot
from sklearn.datasets import make_moons
from keras.models import Sequential
from keras.layers import Dense
from keras.regularizers import l2
X,Y=make_moons(n_samples=100,noise=0.2,random_state=1)
n_train=30
trainX,testX=X[:n_train,:],X[n_train:]
trainY,testY=Y[:n_train],Y[n_train:]
#print(trainX)
#print(trainY)
#print(testX)
#print(testY)
model=Sequential()
model.add(Dense(500,input_dim=2,activation='relu',kernel_regularizer=l2(0.001)))
model.add(Dense(1,activation='sigmoid'))
```

```
model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])
history=model.fit(trainX,trainY,validation_data=(testX,testY),epochs=4000)
pyplot.plot(history.history['accuracy'],label='train')
pyplot.plot(history.history['val_accuracy'],label='test')
pyplot.legend()
pyplot.show()
```



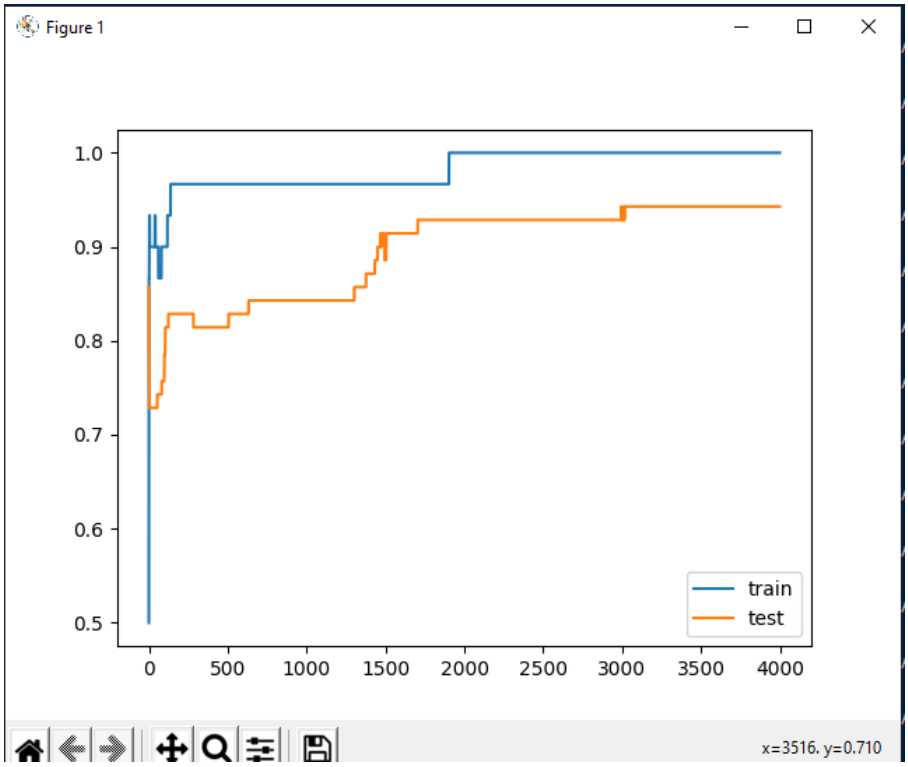
By replacing l2 regularizer with l1 regularizer at the same learning rate 0.001 we get the following output.



By applying l1 and l2 regularizer we can observe the following changes in accuracy of both trainig and testing data. The changes in code are also highlighted.

```
from matplotlib import pyplot
from sklearn.datasets import make_moons
from keras.models import Sequential
from keras.layers import Dense
from keras.regularizers import l1_l2
X,Y=make_moons(n_samples=100,noise=0.2,random_state=1)
n_train=30
trainX,testX=X[:n_train,:],X[n_train:]
trainY,testY=Y[:n_train],Y[n_train:]
#print(trainX)
#print(trainY)
#print(testX)
#print(testY)
model=Sequential()
model.add(Dense(500,input_dim=2,activation='relu',kernel_regularizer=l1_l2(l1=0.001,l2=0.001)))
model.add(Dense(1,activation='sigmoid'))
model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])
history=model.fit(trainX,trainY,validation_data=(testX,testY),epochs=4000)
pyplot.plot(history.history['accuracy'],label='train')
pyplot.plot(history.history['val_accuracy'],label='test')
pyplot.legend()
pyplot.show()
```

OUTPUT:



Practical No:7

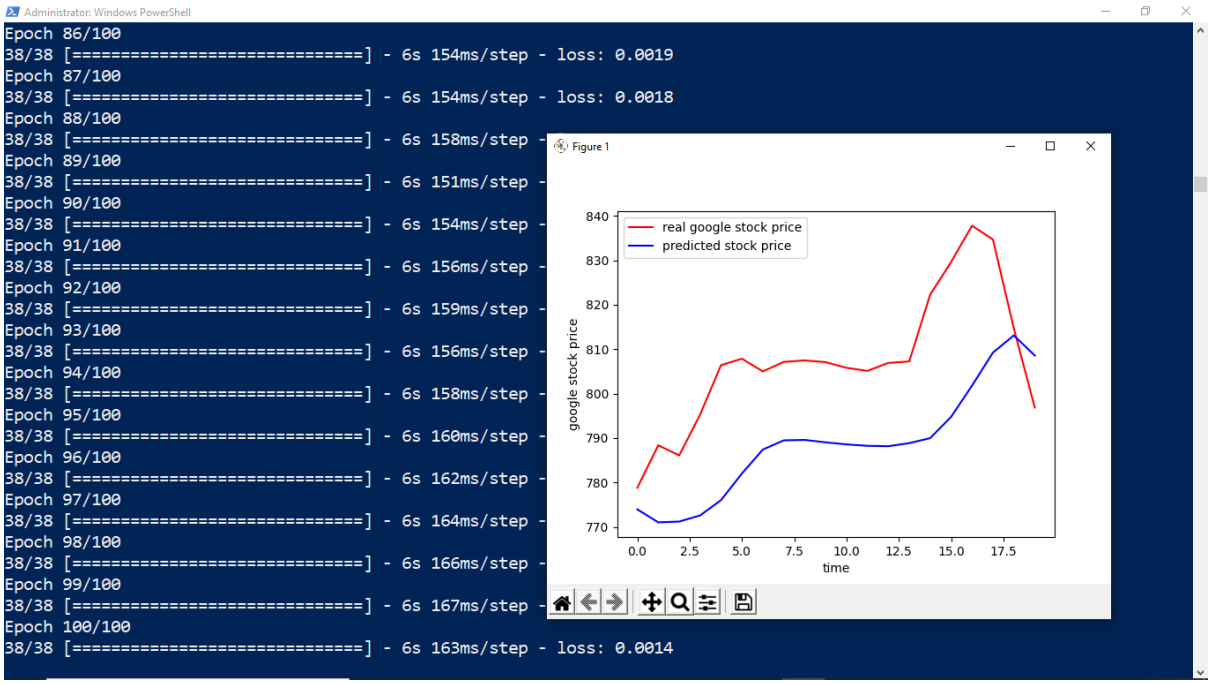
Aim: Demonstrate recurrent neural network that learns to perform sequence analysis for stock price.

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.layers import Dropout
from sklearn.preprocessing import MinMaxScaler
dataset_train=pd.read_csv('Google_Stock_price_train.csv')
#print(dataset_train)
training_set=dataset_train.iloc[:,1:2].values

#print(training_set)
sc=MinMaxScaler(feature_range=(0,1))
training_set_scaled=sc.fit_transform(training_set)
#print(training_set_scaled)
```

```
X_train=[]
Y_train=[]
for i in range(60,1258):
    X_train.append(training_set_scaled[i-60:i,0])
    Y_train.append(training_set_scaled[i,0])
X_train,Y_train=np.array(X_train),np.array(Y_train)
print(X_train)
print('**********************************************')
print(Y_train)
X_train=np.reshape(X_train,(X_train.shape[0],X_train.shape[1],1))
print('**********************************************')
print(X_train)
regressor=Sequential()
regressor.add(LSTM(units=50,return_sequences=True,input_shape=(X_train.shape[1],1)))
regressor.add(Dropout(0.2))
regressor.add(LSTM(units=50,return_sequences=True))
regressor.add(Dropout(0.2))
regressor.add(LSTM(units=50,return_sequences=True))
regressor.add(Dropout(0.2))
regressor.add(LSTM(units=50))
regressor.add(Dropout(0.2))
regressor.add(Dense(units=1))

regressor.compile(optimizer='adam',loss='mean_squared_error')
regressor.fit(X_train,Y_train,epochs=100,batch_size=32)

dataset_test=pd.read_csv('Google_Stock_price_Test.csv')
real_stock_price=dataset_test.iloc[:,1:2].values
dataset_total=pd.concat((dataset_train['Open'],dataset_test['Open']),axis=0)
inputs=dataset_total[len(dataset_total)-len(dataset_test)-60:].values
inputs=inputs.reshape(-1,1)
inputs=sc.transform(inputs)
X_test=[]
for i in range(60,80):
```

```
    X_test.append(inputs[i-60:i,0])
X_test=np.array(X_test)
X_test=np.reshape(X_test,(X_test.shape[0],X_test.shape[1],1))
predicted_stock_price=regressor.predict(X_test)

predicted_stock_price=sc.inverse_transform(predicted_stock_price)
plt.plot(real_stock_price,color='red',label='real google stock price')
plt.plot(predicted_stock_price,color='blue',label='predicted stock price')
plt.xlabel('time')
plt.ylabel('google stock price')
plt.legend()
plt.show()
```

OUTPUT:



Practical No:8

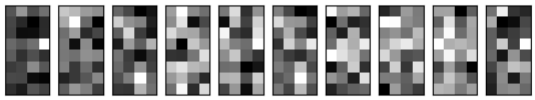
Aim: Performing encoding and decoding of images using deep autoencoder.

```
import keras
from keras import layers
from keras.datasets import mnist
```

```
import numpy as np
encoding_dim=32
#this is our input image
input_img=keras.Input(shape=(784,))
#"encoded" is the encoded representation of the input
encoded=layers.Dense(encoding_dim, activation='relu')(input_img)
#"decoded" is the lossy reconstruction of the input
decoded=layers.Dense(784, activation='sigmoid')(encoded)
#creating autoencoder model
autoencoder=keras.Model(input_img,decoded)
#create the encoder model
encoder=keras.Model(input_img,encoded)
encoded_input=keras.Input(shape=(encoding_dim,))
#Retrive the last layer of the autoencoder model
decoder_layer=autoencoder.layers[-1]
#create the decoder model
decoder=keras.Model(encoded_input,decoder_layer(encoded_input))
autoencoder.compile(optimizer='adam',loss='binary_crossentropy')
#scale and make train and test dataset
(X_train,_),(X_test,_)=mnist.load_data()
X_train=X_train.astype('float32')/255.
X_test=X_test.astype('float32')/255.
X_train=X_train.reshape((len(X_train),np.prod(X_train.shape[1:])))
X_test=X_test.reshape((len(X_test),np.prod(X_test.shape[1:])))
print(X_train.shape)
print(X_test.shape)
#train autoencoder with training dataset
autoencoder.fit(X_train,X_train,
                epochs=50,
                batch_size=256,
                shuffle=True,
                validation_data=(X_test,X_test))
encoded_imgs=encoder.predict(X_test)
decoded_imgs=decoder.predict(encoded_imgs)
```

```
import matplotlib.pyplot as plt
n = 10 # How many digits we will display
plt.figure(figsize=(40, 4))
for i in range(10):
    # display original
    ax = plt.subplot(3, 20, i + 1)
    plt.imshow(X_test[i].reshape(28, 28))
    plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)
    # display encoded image
    ax = plt.subplot(3, 20, i + 1 + 20)
    plt.imshow(encoded_imgs[i].reshape(8,4))
    plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)
    # display reconstruction
    ax = plt.subplot(3, 20, 2*20 +i+ 1)
    plt.imshow(decoded_imgs[i].reshape(28, 28))
    plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)
plt.show()
```

OUTPUT:



Practical No:5b

Aim: Evaluating feed forward deep network for multiclass Classification using KFold cross-validation.

#loading libraries

import pandas

from keras.models import Sequential

```
from keras.layers import Dense
from keras.wrappers.scikit_learn import KerasClassifier
from keras.utils import np_utils
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import KFold
from sklearn.preprocessing import LabelEncoder
#loading dataset
df=pandas.read_csv('Flower.csv',header=None)
print(df)
#splitting dataset into input and output variables
X = df.iloc[:,0:4].astype(float)
y=df.iloc[:,4]
#print(X)
#print(y)
#encoding string output into numeric output
encoder=LabelEncoder()
encoder.fit(y)
encoded_y=encoder.transform(y)
print(encoded_y)
dummy_Y=np_utils.to_categorical(encoded_y)
print(dummy_Y)
def baseline_model():
        # create model
        model = Sequential()
        model.add(Dense(8, input_dim=4, activation='relu'))
        model.add(Dense(3, activation='softmax'))
        # Compile model
        model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
        return model
estimator=baseline_model()
estimator.fit(X,dummy_Y,epochs=100,shuffle=True)
action=estimator.predict(X)
```

```
for i in range(25):
    print(dummy_Y[i])
print('^^^^^^^^^^^^^^^^^^^^^^^^')
for i in range(25):
    print(action[i])
```

OUTPUT:

```
      0    1    2    3              4
0    5.1  3.5  1.4  0.2     Iris-setosa
1    4.9  3.0  1.4  0.2     Iris-setosa
2    4.7  3.2  1.3  0.2     Iris-setosa
3    4.6  3.1  1.5  0.2     Iris-setosa
4    5.0  3.6  1.4  0.2     Iris-setosa
..   ...  ...  ...  ...             ...
145  6.7  3.0  5.2  2.3  Iris-virginica
146  6.3  2.5  5.0  1.9  Iris-virginica
147  6.5  3.0  5.2  2.0  Iris-virginica
148  6.2  3.4  5.4  2.3  Iris-virginica
149  5.9  3.0  5.1  1.8  Iris-virginica

[150 rows x 5 columns]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2]
[[1. 0. 0.]
 [1. 0. 0.]
 [1. 0. 0.]
 [1. 0. 0.]
 [1. 0. 0.]
 [1. 0. 0.]
 [1. 0. 0.]
 [1. 0. 0.]
 [1. 0. 0.]
 [1. 0. 0.]
 [1. 0. 0.]
```

```
Epoch 98/100
5/5 [==============================] - 0s 0s/step - loss: 0.3899 - accuracy: 0.9313
Epoch 99/100
5/5 [==============================] - 0s 0s/step - loss: 0.3896 - accuracy: 0.9230
Epoch 100/100
5/5 [==============================] - 0s 0s/step - loss: 0.3682 - accuracy: 0.9361
[1. 0. 0.]
[1. 0. 0.]
[1. 0. 0.]
[1. 0. 0.]
[1. 0. 0.]
[1. 0. 0.]
[1. 0. 0.]
[1. 0. 0.]
[1. 0. 0.]
[1. 0. 0.]
[1. 0. 0.]
[1. 0. 0.]
[1. 0. 0.]
[1. 0. 0.]
[1. 0. 0.]
[1. 0. 0.]
[1. 0. 0.]
[1. 0. 0.]
[1. 0. 0.]
[1. 0. 0.]
[1. 0. 0.]
```

```
^^^^^^^^^^^^^^^^^^^^^
[0.9145307  0.08423453 0.00123477]
[0.88751584 0.1100563  0.00242792]
[0.8999843  0.09803853 0.00197715]
[0.858188   0.13759544 0.00421653]
[0.9138275  0.08489472 0.00127787]
[0.8994011  0.09916449 0.0014343 ]
[0.8872866  0.11023647 0.00247695]
[0.89339536 0.10458492 0.00201967]
[0.8545533  0.14064151 0.00480518]
[0.87742513 0.11963753 0.00293737]
[0.9203753  0.07866727 0.00095734]
[0.8665611  0.1300417  0.00339716]
[0.88403696 0.11323617 0.0027269 ]
[0.9008803  0.09682965 0.00229002]
[9.5539063e-01 4.4350266e-02 2.5906262e-04]
[9.4327897e-01 5.6333560e-02 3.8754733e-04]
[9.3672138e-01 6.2714875e-02 5.6370755e-04]
[0.91191673 0.08680107 0.00128225]
[0.9100969  0.08882014 0.00108295]
[0.91078293 0.08794734 0.00126965]
[0.8827079  0.11510085 0.00219123]
[0.9060573  0.09255142 0.00139134]
[9.3434143e-01 6.4821333e-02 8.3730859e-04]
[0.85551745 0.14102885 0.00345369]
[0.80272377 0.1895675  0.00770868]
```

Code 2:

```
import pandas
from keras.models import Sequential
from keras.layers import Dense
from keras.wrappers.scikit_learn import KerasClassifier
from keras.utils import np_utils
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import KFold
from sklearn.preprocessing import LabelEncoder

dataset=pandas.read_csv("Flower.csv",header=None)
```

```
dataset1=dataset.values
X=dataset1[:,0:4].astype(float)
Y=dataset1[:,4]
print(Y)
encoder=LabelEncoder()
encoder.fit(Y)
encoder_Y=encoder.transform(Y)
print(encoder_Y)
dummy_Y=np_utils.to_categorical(encoder_Y)
print(dummy_Y)
def baseline_model():
        model=Sequential()
        model.add(Dense(8,input_dim=4,activation='relu'))
        model.add(Dense(3,activation='softmax'))
        model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
        return model
estimator=KerasClassifier(build_fn=baseline_model,epochs=100,batch_size=5)
kfold = KFold(n_splits=10, shuffle=True)
results = cross_val_score(estimator, X, dummy_Y, cv=kfold)
print("Baseline: %.2f%% (%.2f%%)" % (results.mean()*100, results.std()*100))
```

```
3/3 [==============================] - 0s 2ms/step - loss: 0.2491 - accuracy: 0.9333
Baseline: 96.00% (4.42%)
```

(Changing neuron)

```
model.add(Dense(10,input_dim=4,activation='relu'))
```

```
3/3 [==============================] - 0s 999us/step - loss: 0.1436 - accuracy: 1.0000
Baseline: 98.67% (2.67%)
```

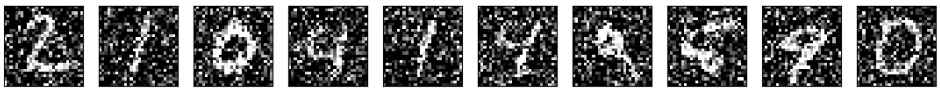
Practical No:10

Aim: Denoising of images using autoencoder.

```
import keras
from keras.datasets import mnist
from keras import layers
import numpy as np
from keras.callbacks import TensorBoard
import matplotlib.pyplot as plt
(X_train,_),(X_test,_)=mnist.load_data()
X_train=X_train.astype('float32')/255.
X_test=X_test.astype('float32')/255.
X_train=np.reshape(X_train,(len(X_train),28,28,1))
X_test=np.reshape(X_test,(len(X_test),28,28,1))
noise_factor=0.5
X_train_noisy=X_train+noise_factor*np.random.normal(loc=0.0,scale=1.0,size=X_train.shape)
X_test_noisy=X_test+noise_factor*np.random.normal(loc=0.0,scale=1.0,size=X_test.shape)
X_train_noisy=np.clip(X_train_noisy,0.,1.)
X_test_noisy=np.clip(X_test_noisy,0.,1.)
n=10
plt.figure(figsize=(20,2))
for i in range(1,n+1):
    ax=plt.subplot(1,n,i)
    plt.imshow(X_test_noisy[i].reshape(28,28))
    plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)
plt.show()
input_img=keras.Input(shape=(28,28,1))
x=layers.Conv2D(32,(3,3),activation='relu',padding='same')(input_img)
x=layers.MaxPooling2D((2,2),padding='same')(x)
x=layers.Conv2D(32,(3,3),activation='relu',padding='same')(x)
encoded=layers.MaxPooling2D((2,2),padding='same')(x)
x=layers.Conv2D(32,(3,3),activation='relu',padding='same')(encoded)
```

```
x=layers.UpSampling2D((2,2))(x)
x=layers.Conv2D(32,(3,3),activation='relu',padding='same')(x)
x=layers.UpSampling2D((2,2))(x)
decoded=layers.Conv2D(1,(3,3),activation='sigmoid',padding='same')(x)
autoencoder=keras.Model(input_img,decoded)
autoencoder.compile(optimizer='adam',loss='binary_crossentropy')
autoencoder.fit(X_train_noisy,X_train,
                epochs=3,
                batch_size=128,
                shuffle=True,
                validation_data=(X_test_noisy,X_test),
callbacks=[TensorBoard(log_dir='/tmo/tb',histogram_freq=0,write_graph=False)])
predictions=autoencoder.predict(X_test_noisy)
m=10
plt.figure(figsize=(20,2))
for i in range(1,m+1):
    ax=plt.subplot(1,m,i)
    plt.imshow(predictions[i].reshape(28,28))
    plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)
plt.show()
```

OUTPUT:



After 3 epochs:

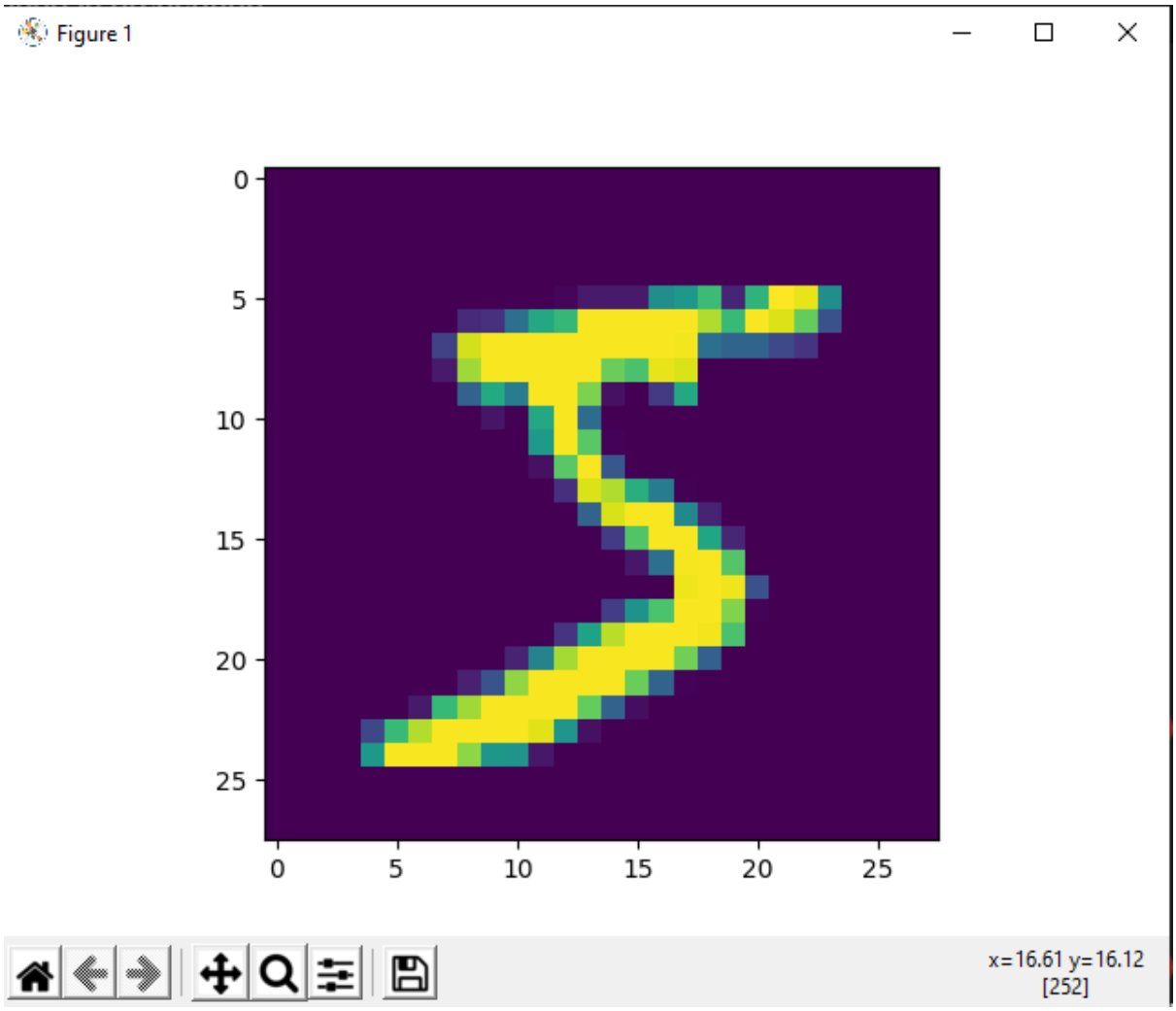
Practical No:9

Aim: Implementation of convolutional neural network to predict numbers from number images

```
from keras.datasets import mnist
from keras.utils import to_categorical
from keras.models import Sequential
from keras.layers import Dense,Conv2D,Flatten
import matplotlib.pyplot as plt

#download mnist data and split into train and test sets
(X_train,Y_train),(X_test,Y_test)=mnist.load_data()

#plot the first image in the dataset
plt.imshow(X_train[0])
plt.show()

print(X_train[0].shape)
X_train=X_train.reshape(60000,28,28,1)
X_test=X_test.reshape(10000,28,28,1)

Y_train=to_categorical(Y_train)
Y_test=to_categorical(Y_test)
Y_train[0]
print(Y_train[0])

model=Sequential()
#add model layers
#learn image features
model.add(Conv2D(64,kernel_size=3,activation='relu',input_shape=(28,28,1)))
model.add(Conv2D(32,kernel_size=3,activation='relu'))

model.add(Flatten())
model.add(Dense(10,activation='softmax'))
```

```
model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])
#train
model.fit(X_train,Y_train,validation_data=(X_test,Y_test),epochs=3)

print(model.predict(X_test[:4]))
#actual results for 1st 4 images in the test set
print(Y_test[:4])
```

OUTPUT:



(28, 28)

[0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]

```
(venv) PS D:\keras> python pract6.py
(28, 28)
[0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]
```

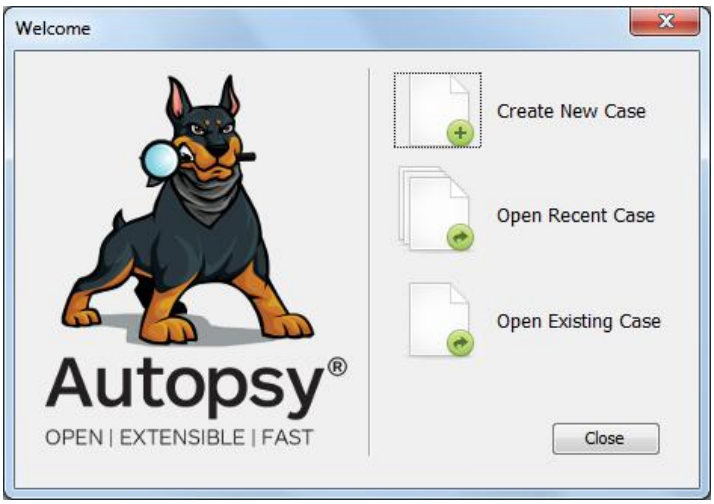
```
Epoch 1/3
1875/1875 [==============================] - 235s 124ms/step - loss: 0.9714 - accuracy: 0.9111
- val_loss: 0.1084 - val_accuracy: 0.9661
Epoch 2/3
1875/1875 [==============================] - 242s 129ms/step - loss: 0.0663 - accuracy: 0.9789
- val_loss: 0.0787 - val_accuracy: 0.9758
Epoch 3/3
1875/1875 [==============================] - 241s 128ms/step - loss: 0.0458 - accuracy: 0.9854
- val_loss: 0.0904 - val_accuracy: 0.9751
[[8.5066381e-09 1.9058415e-15 1.5103029e-09 6.2544638e-07 4.8599115e-14
  3.8009873e-13 8.0967405e-13 9.9999940e-01 2.3813423e-10 1.8504194e-09]
 [4.6695381e-10 4.9075446e-09 1.0000000e+00 1.4425230e-12 5.5351397e-15
  1.4244286e-16 4.9031729e-10 2.1196991e-15 8.1773255e-13 2.7225001e-19]
 [1.4877173e-06 9.9855584e-01 1.0760028e-04 1.4199993e-07 1.0726219e-03
  6.1853432e-05 5.0982948e-05 6.4035441e-05 8.5100648e-05 3.5164564e-07]
 [9.9999988e-01 7.7231385e-13 9.2269055e-08 2.9055267e-10 1.8901826e-10
  2.9204628e-09 8.1175129e-09 4.1387605e-12 6.0085120e-10 1.4425010e-08]]
[[0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]]
(venv) PS D:\keras>
```

Index

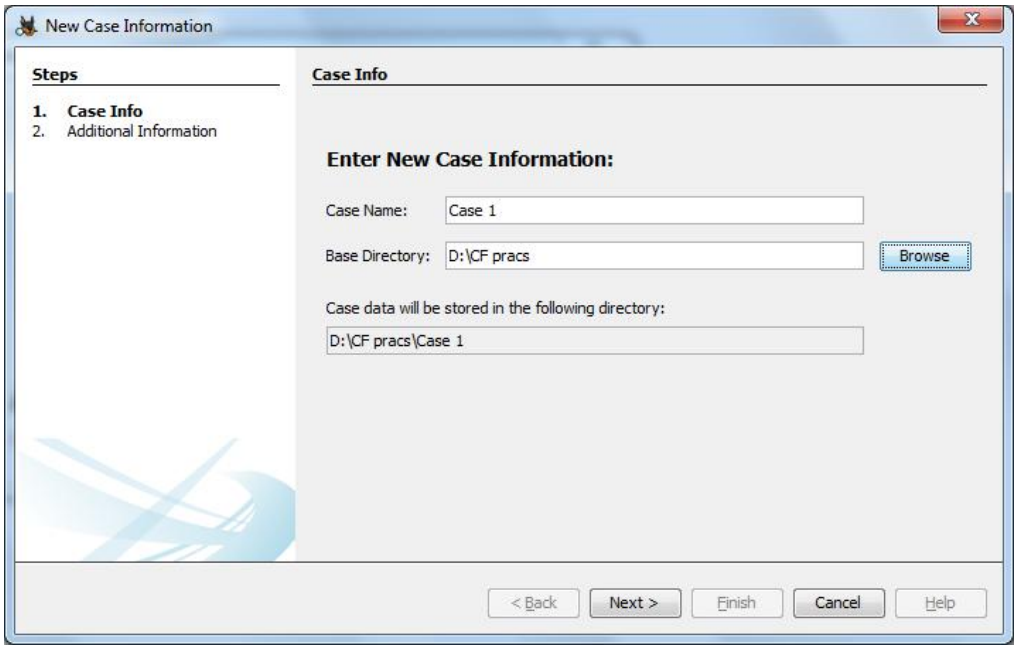
Practical No: 01

How to Start a Case.

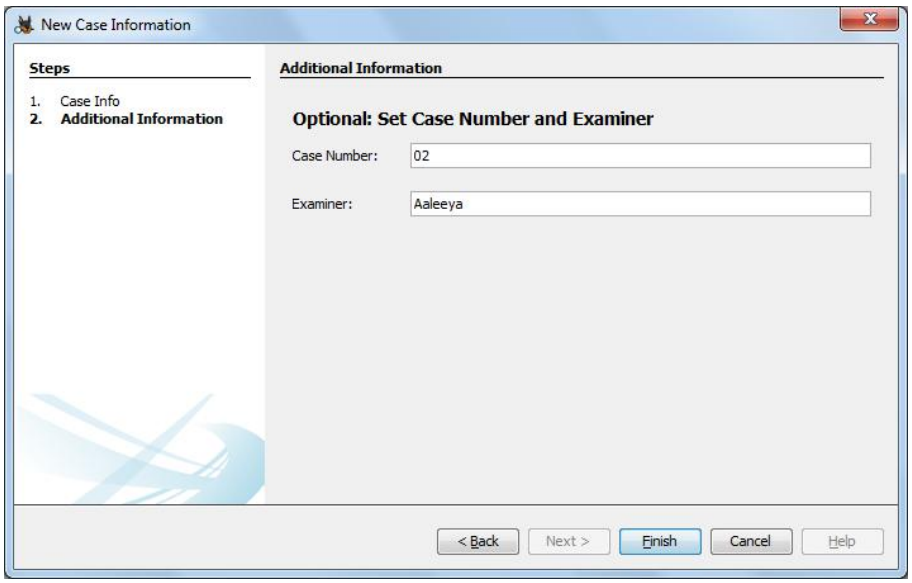
Upon starting Autopsy 3.1.2, a window will open with three selections to make: create a new case, open existing case, or to open a recent case.



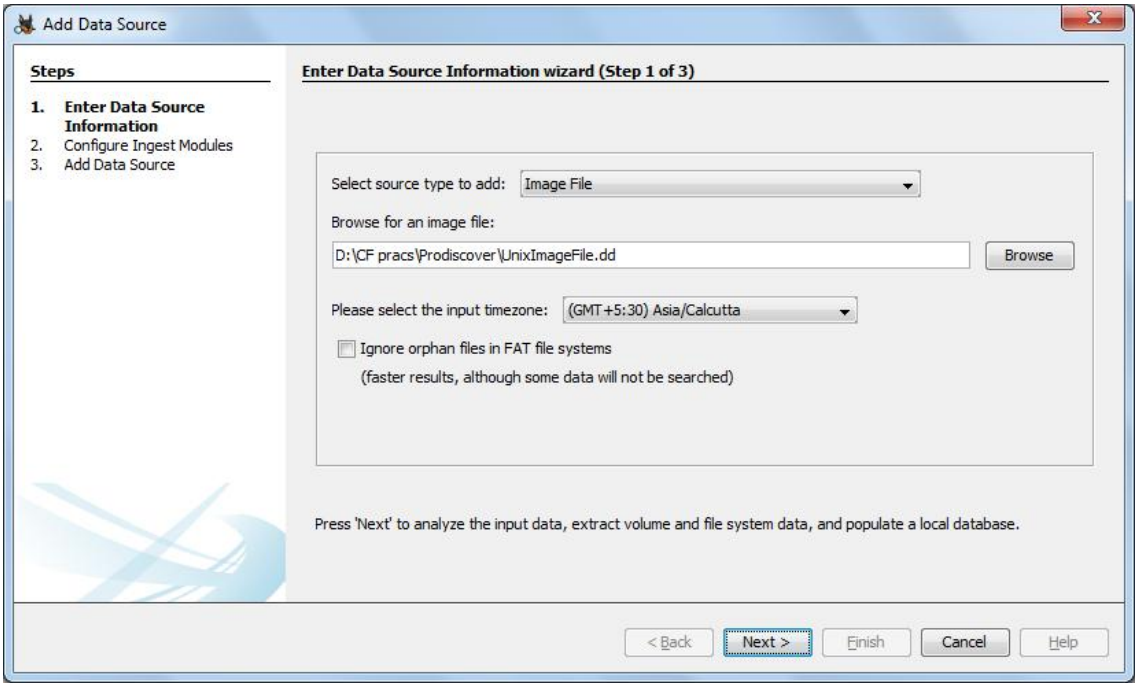
Step 1) Select the "Create New Case" option and be directed to a new window that will have information to fill in, we will be naming the case "Test."



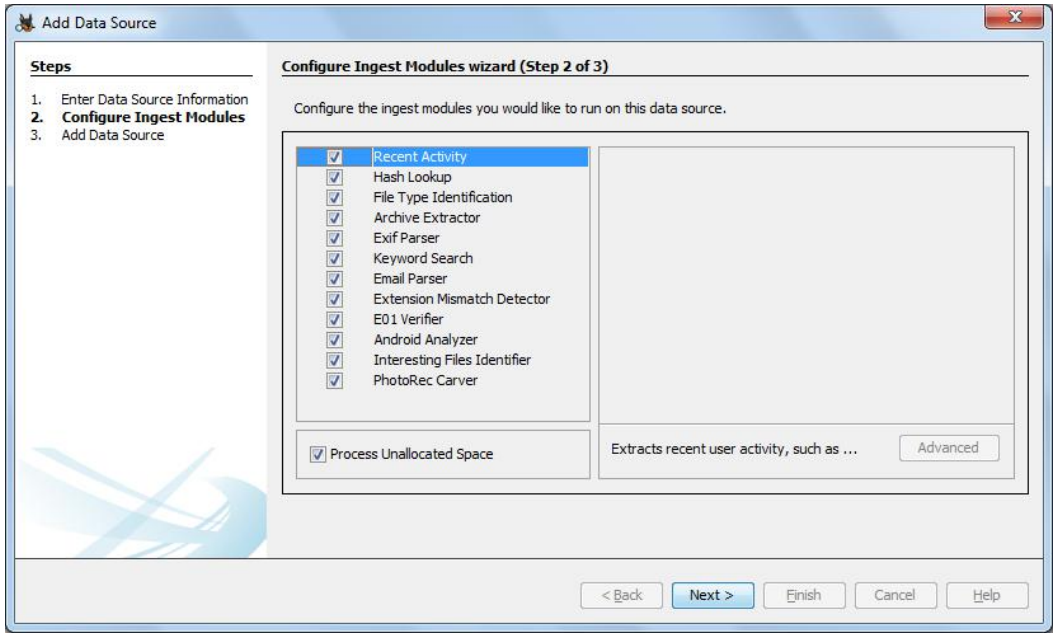
Step 2) After the information has been filled in select the next button. The next window will allow the investigator to fill in the case number and examiner name. This is for the purpose of creating better documentation and logging. After the information is filled in select the finish button to continue.



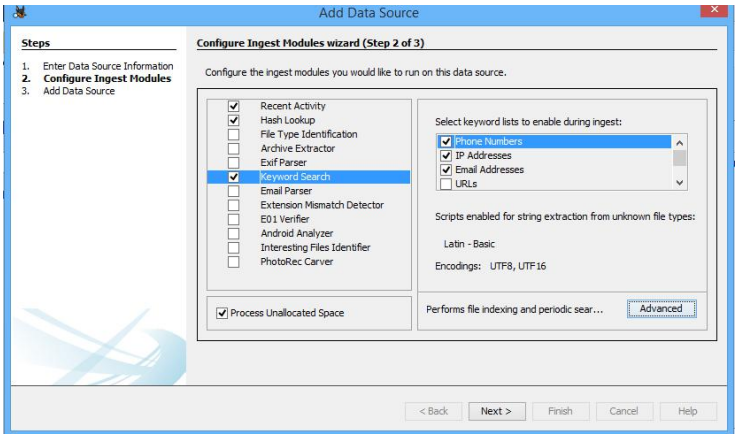
Step 3) The next step in the investigation will be to add an image file to the case. The image file can be chosen from a wide variety of formats including: img, dd, 001, aa, and e01. Use the browse button to find the image that is desired to work with and select add. Options to choose the timezone of where the image came from as well as to ignore orphan files in FAT file systems are available to be selected based on the investigators preference and situation.



Step 4) After selecting the next button the image will be added to the case and the next button should be selected again if there are no errors.

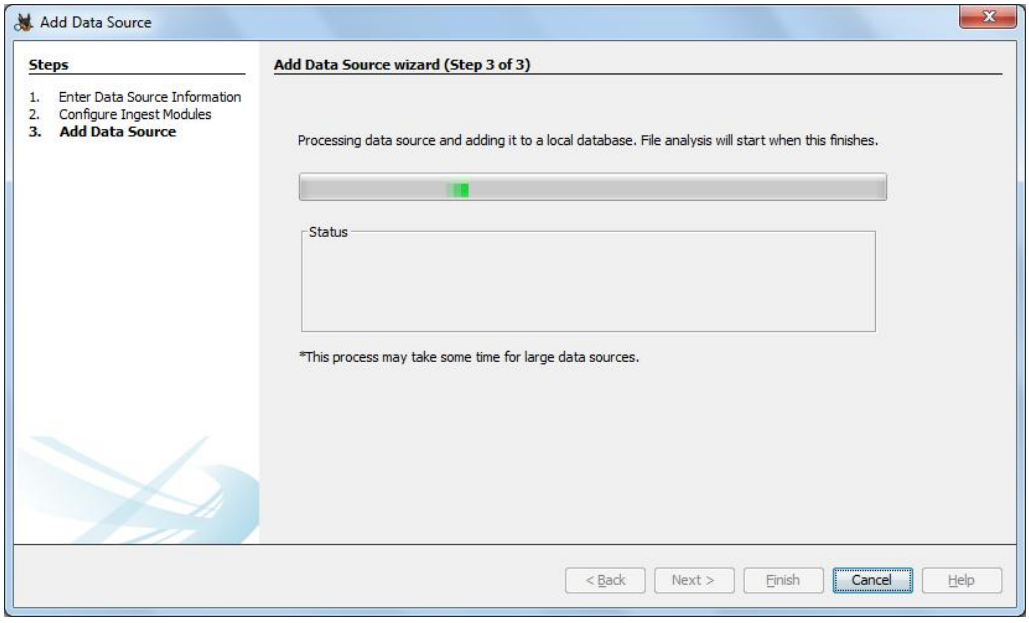


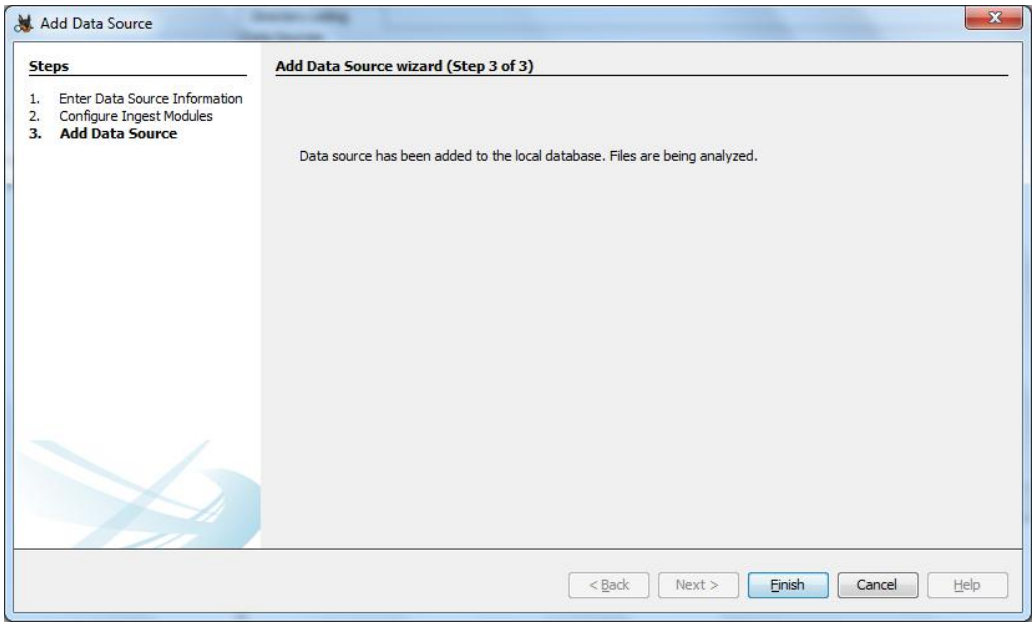
Step 5) The following window will bring the investigator to the Ingest wizard panel, which is one of the new features offered in Autopsy. There are three options in the first box: Recent Activity, Hash lookup, and Keyword Searches.

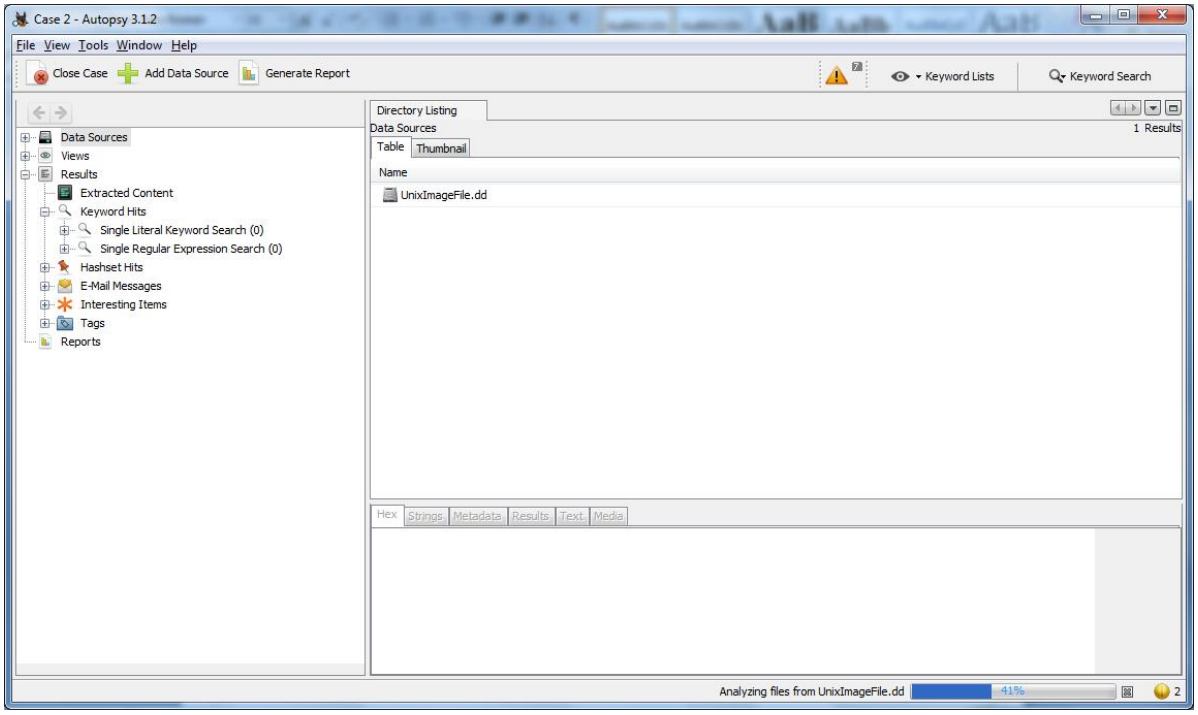


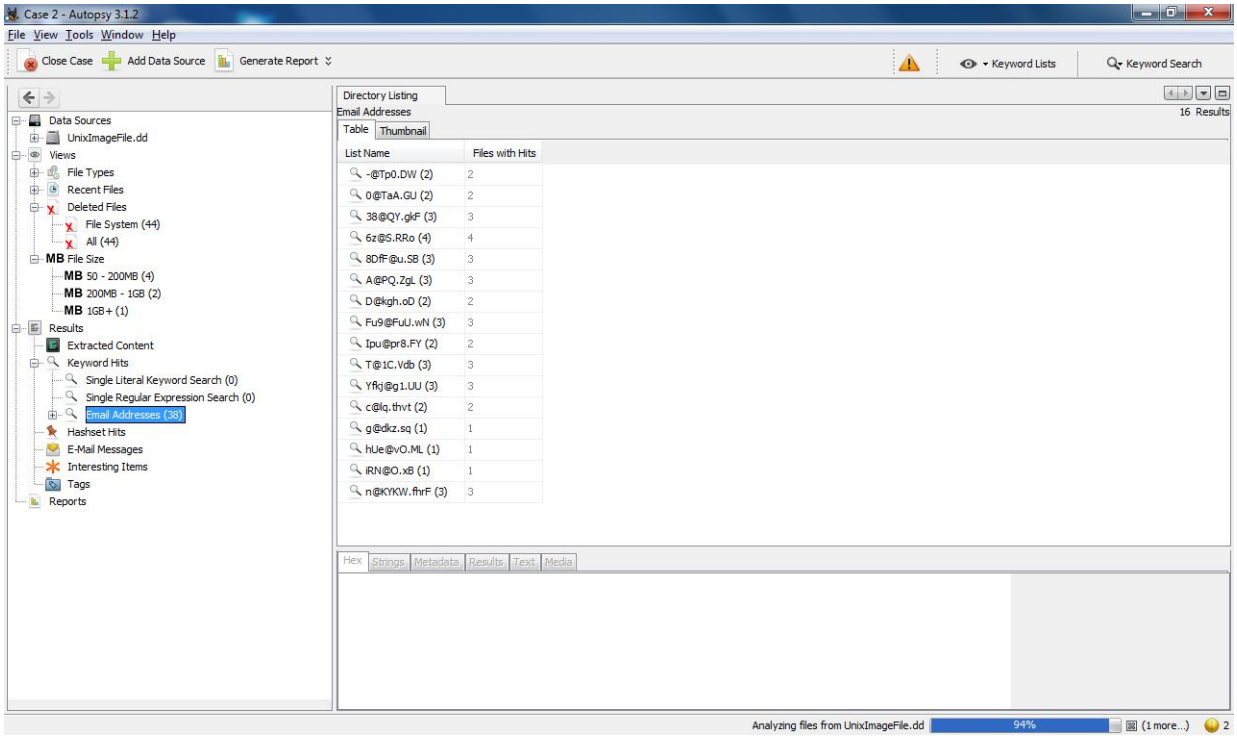
By selecting any of the options advanced settings can be set to increase the capabilities of the search. Under the Hash Lookup option there is the advanced option to add databases of known hashes.

Under the Keyword Search option are many different lists that can be used to search for information. By default, Phone Numbers, IP Addresses, Email Addresses, and URL's are available. Select the Advanced button and a Keyword List Configuration window will open. In this new window select New List and type the name that is desired for the list. This makes it easier to search by subject matter or other organizational methods. For now the list Test keywords will be used to create a list. In the adjacent pane there is a blank section with a word bar and an Add button next to it. Type the keyword desired (case sensitive) and select Add to add the word to the list. There is also the option to select Regular Expression. This allows the investigator to further narrow the field to search in by selecting what the keyword is that is being searched for including: passwords, emails, text file name, domains, and many more options.

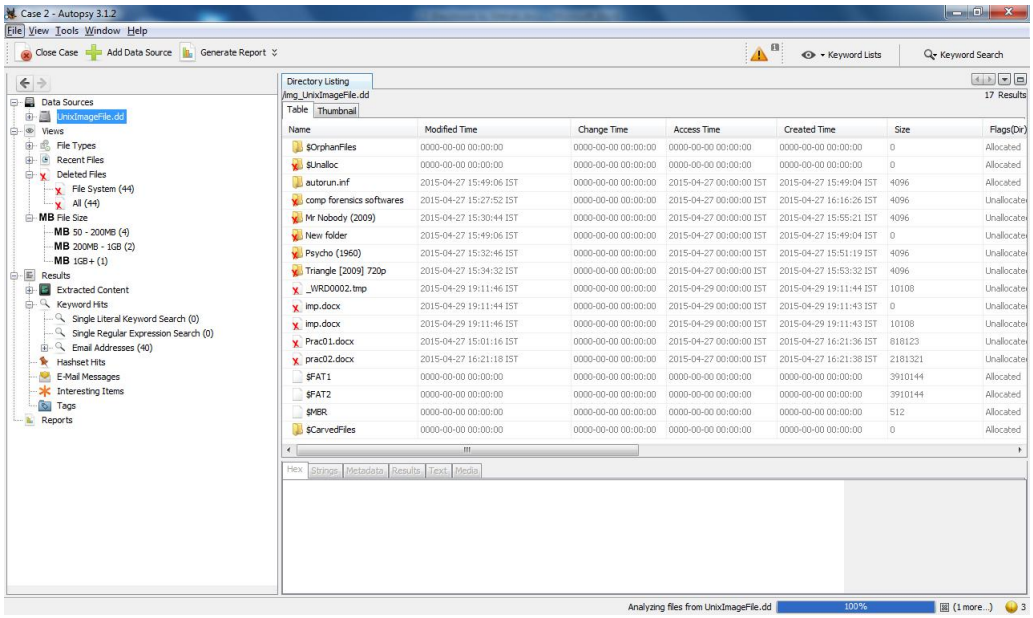

Add Data Source
Steps
1. Enter Data Source Information
2. Configure Ingest Modules
3. Add Data Source
Add Data Source wizard (Step 3 of 3)
Processing data source and adding it to a local database. File analysis will start when this finishes.
Status
*This process may take some time for large data sources.
< Back
Next >
Finish
Cancel
Help


Add Data Source
Steps
1. Enter Data Source Information
2. Configure Ingest Modules
3. Add Data Source
Add Data Source wizard (Step 3 of 3)
Data source has been added to the local database. Files are being analyzed.
< Back
Next >
Finish
Cancel
Help

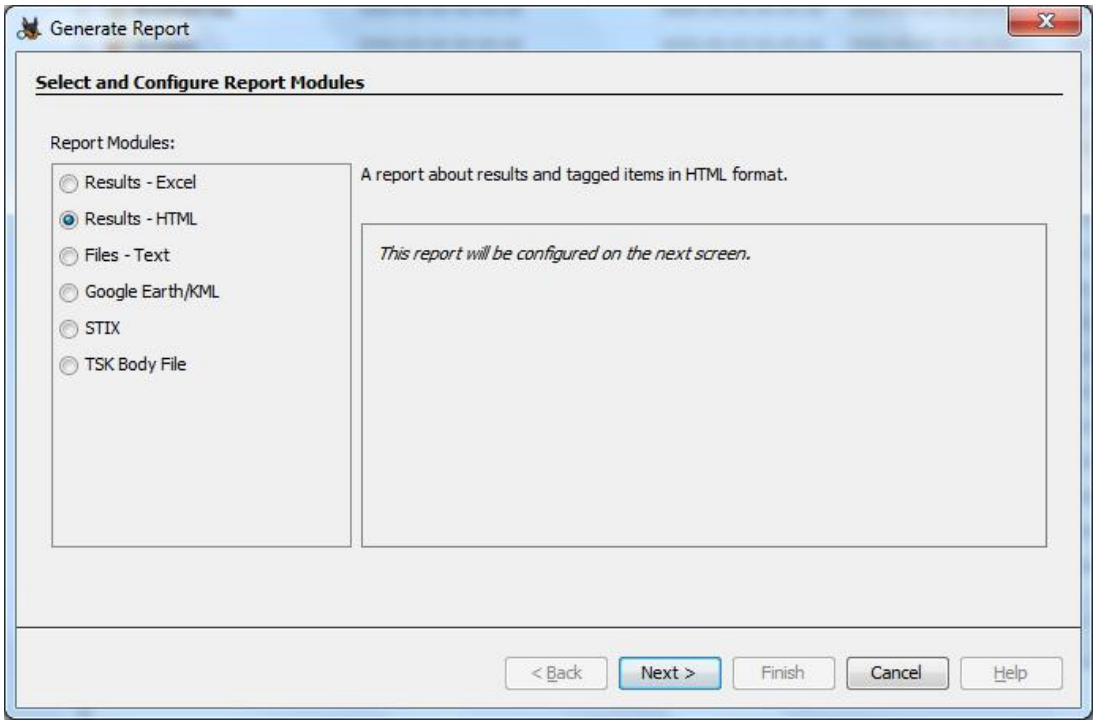

Case 2 - Autopsy 3.1.2
File View Tools Window Help
Close Case
Add Data Source
Generate Report
Keyword Lists
Keyword Search
Data Sources
Views
Results
Extracted Content
Keyword Hits
Single Literal Keyword Search (0)
Single Regular Expression Search (0)
Hashset Hits
E-Mail Messages
Interesting Items
Tags
Reports
Directory Listing
Data Sources
1 Results
Table
Thumbnail
Name
UnixImageFile.dd
Hex
Strings
Metadata
Results
Text
Media
Analyzing files from UnixImageFile.dd
41%

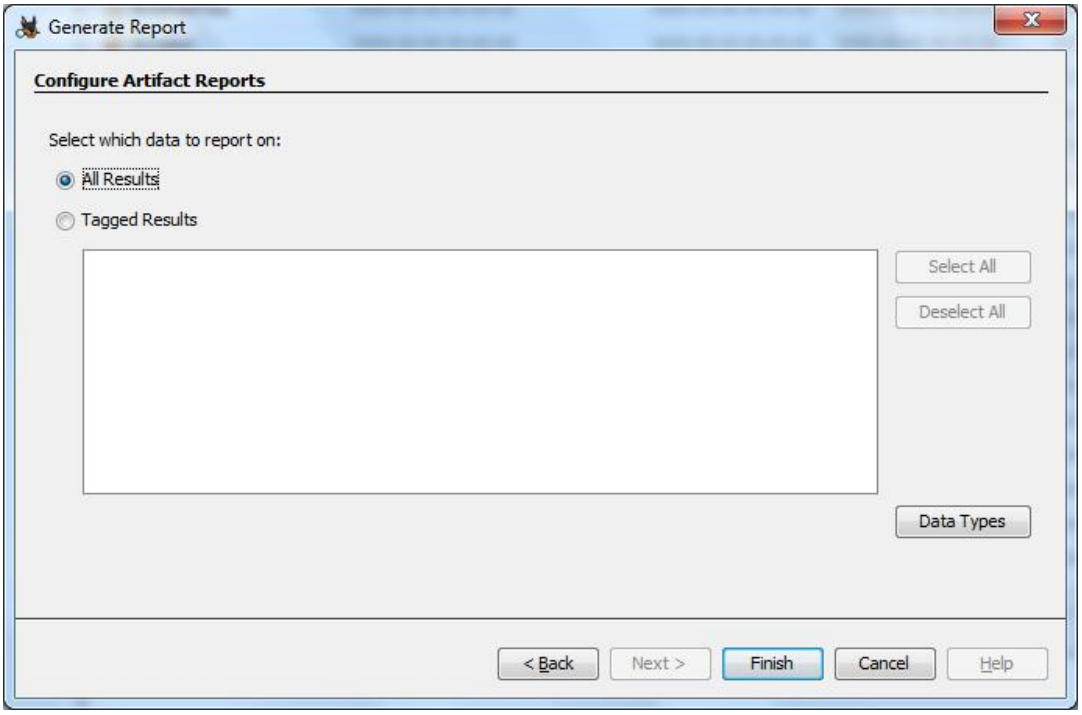

Case 2 - Autopsy 3.1.2
File View Tools Window Help
Close Case
Add Data Source
Generate Report
Keyword Lists
Keyword Search
Data Sources
UnixImageFile.dd
Views
File Types
Recent Files
Deleted Files
File System (44)
All (44)
File Size
50 - 200MB (4)
200MB - 1GB (2)
1GB+ (1)
Results
Extracted Content
Keyword Hits
Single Literal Keyword Search (0)
Single Regular Expression Search (0)
Email Addresses (38)
Hashset Hits
E-Mail Messages
Interesting Items
Tags
Reports
Directory Listing
Email Addresses
16 Results
Table
Thumbnail
List Name
Files with Hits
-@Tp0.DW (2) 2
0@TaA.GU (2) 2
38@QY.gkF (3) 3
6z@S.RRo (4) 4
8DfF@u.SB (3) 3
A@PQ.ZgL (3) 3
D@kgh.oD (2) 2
Fu9@FuU.wN (3) 3
Ipu@pr8.FY (2) 2
T@1C.Vdb (3) 3
Yfkj@g1.UU (3) 3
c@lq.thvt (2) 2
g@dkz.sq (1) 1
hUe@vO.ML (1) 1
iRN@O.xB (1) 1
n@KYKW.fhrF (3) 3
Hex
Strings
Metadata
Results
Text
Media
Analyzing files from UnixImageFile.dd
94%
(1 more...)

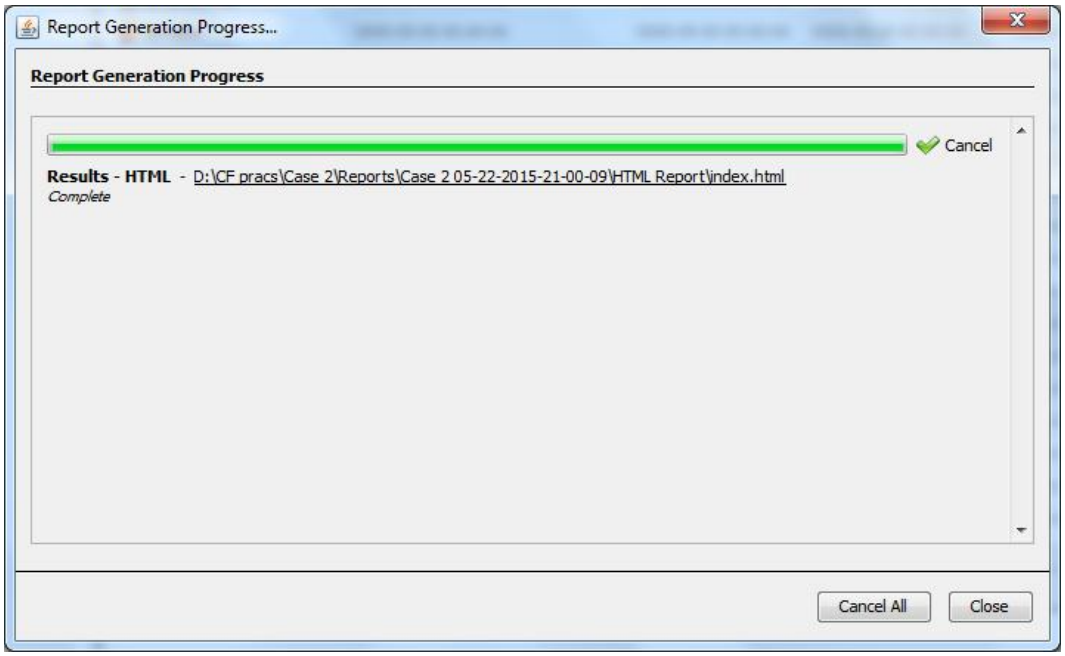
Step 6) After finishing the keyword parameters the screen will be laid out for the user.



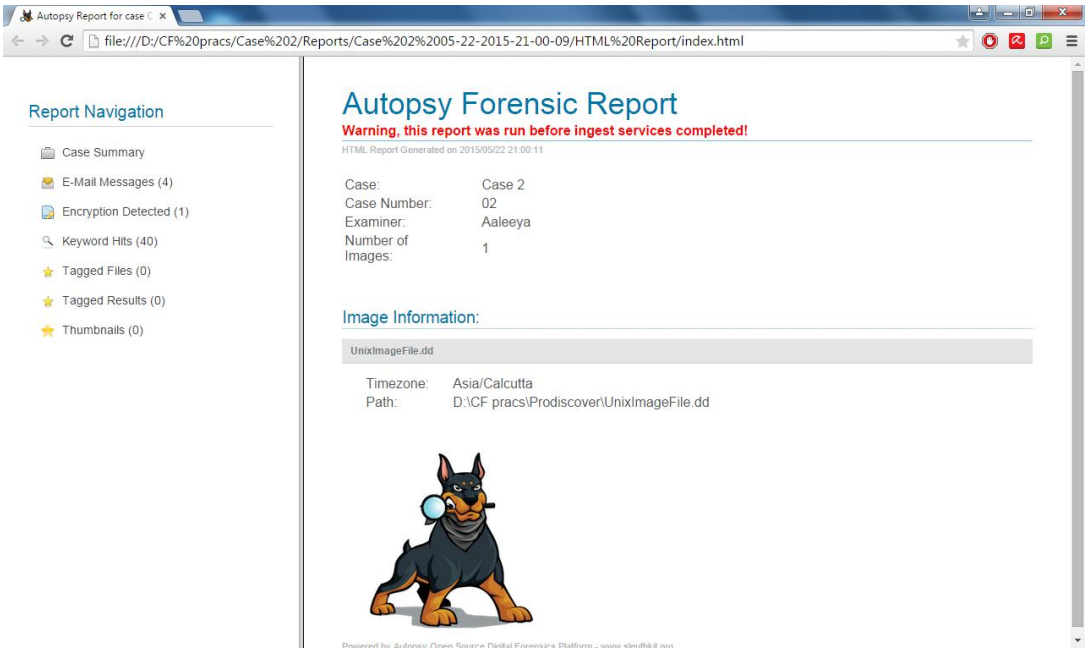
Step 7) After the image is indexed the tree will be populated by the file system, extracted content, keyword searches, and the hash list (if any were used).
the investigator should generate a report. This will allow the investigator to have an idea of what type of information is available and what to expect. The report can be generated in three formats: Excel, XML, and HTML. It also has the ability to select what information to display with choices that can be seen in the image below.



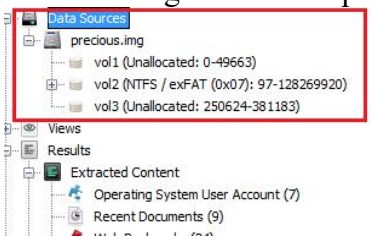

Generate Report
Configure Artifact Reports
Select which data to report on:
All Results
Tagged Results
Select All
Deselect All
Data Types
< Back
Next >
Finish
Cancel
Help


Report Generation Progress...
Report Generation Progress
Cancel
Results - HTML - D:\CF pracs\Case 2\Reports\Case 2 05-22-2015-21-00-09\HTML Report\index.html
Complete
Cancel All
Close

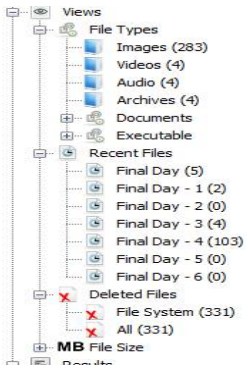
Step 8) With the report on hand the investigator will have an idea of what to expect as well as a list of programs that are installed on the machine. This can help investigators gather all the evidence they need to perform a complete investigation.



Looking at the tree, the top selection is titled “Data Sources” this is where the acquired image is located and the bulk of the investigation, will take place. If the Images tab is expanded the investigator will see each image that was added to the investigation. By expanding an images tab the volumes of the image will be seen including the file system and unallocated space. Expanding the tab that contains the Operating System will give the investigator a look at the root directory and the tree that contains most of the relevant information. This is the same as if the investigator would open the default drive when browsing through a system.

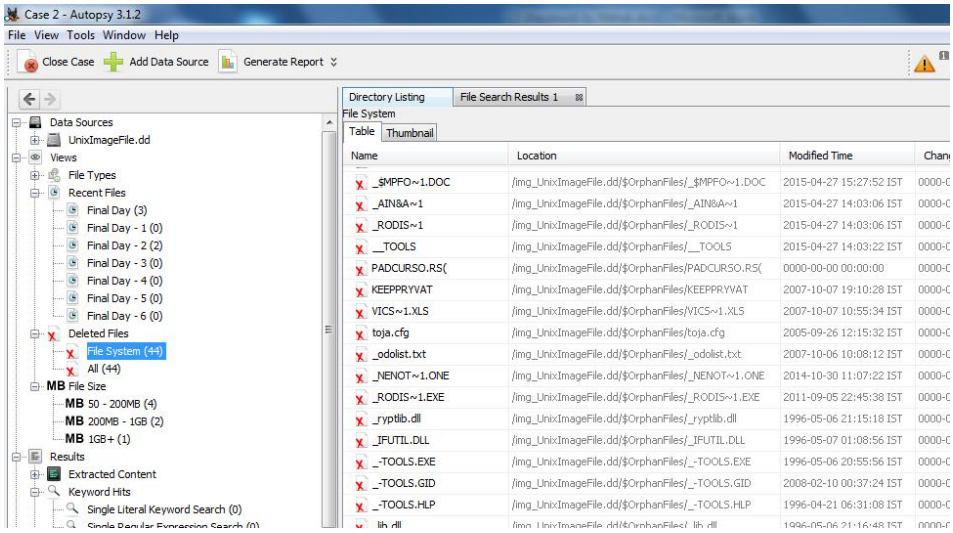


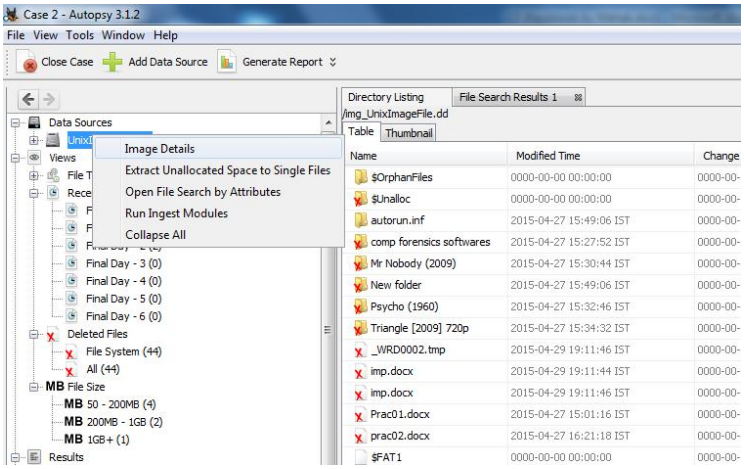
Below the Images tab is the “Views” tab that will allow the investigator to separate the information in the image into different categories such as by file types and by recent documents. The file type can be broken down into: images, video, audio, and documents which includes the major text formats. Another section in the Views tab is a new feature in Autopsy 3, the Recent Files tab. This tab allows the investigator to get a rough outline of what happened in the last 6 days of use by the suspect. The results include registry files, documents opened, and programs run.



The next tab that is seen is the Results tab, this is a new feature that displays all the information from the ingest process. This uses the program BEViewer to look for certain information inside of the data and separate it into sections that make it easier to search for specific data instead of going through all of the information manually. Although this simplifies the investigation process, it does not mean that this is all of the information that is able to be gained through an investigation.

There are 4 main categories when separating the Results tab: Extracted Content, Keyword Hits, Hashset Hits, and E-mail Messages. Each of these sections has subsections that allow for more specific information divisions. In the Extracted Content tab there are sections for: Bookmarks, Cookies, Web History, Downloads, Recent Documents, Installed Programs, and Device Attached.




Case 2 - Autopsy 3.1.2
File View Tools Window Help
Close Case
Add Data Source
Generate Report
Directory Listing
File Search Results 1
/img_UnixImageFile.dd
Table
Thumbnail
Name
Modified Time
Change
Data Sources
Views
Image Details
Extract Unallocated Space to Single Files
Open File Search by Attributes
Run Ingest Modules
Collapse All
Final Day - 3 (0)
Final Day - 4 (0)
Final Day - 5 (0)
Final Day - 6 (0)
Deleted Files
File System (44)
All (44)
MB File Size
MB 50 - 200MB (4)
MB 200MB - 1GB (2)
MB 1GB+ (1)
Results
$OrphanFiles
$Unalloc
autorun.inf
comp forensics softwares
Mr Nobody (2009)
New folder
Psycho (1960)
Triangle [2009] 720p
_WRD0002.tmp
imp.docx
imp.docx
Prac01.docx
prac02.docx
$FAT1

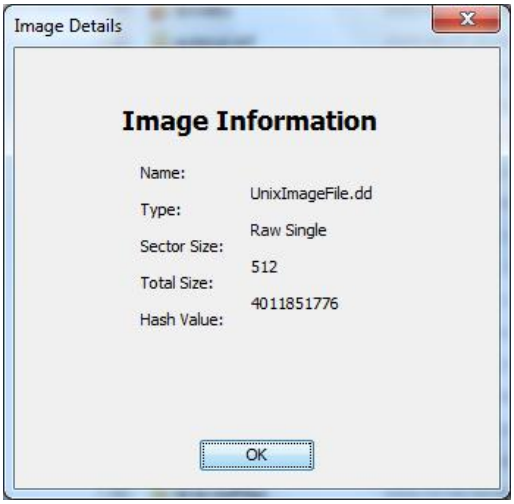
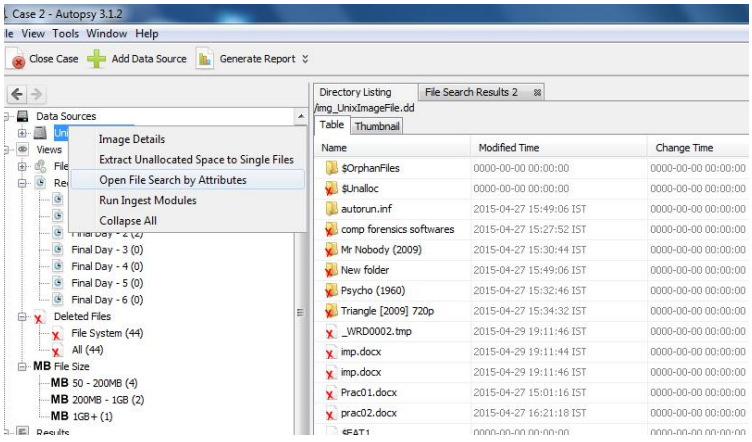
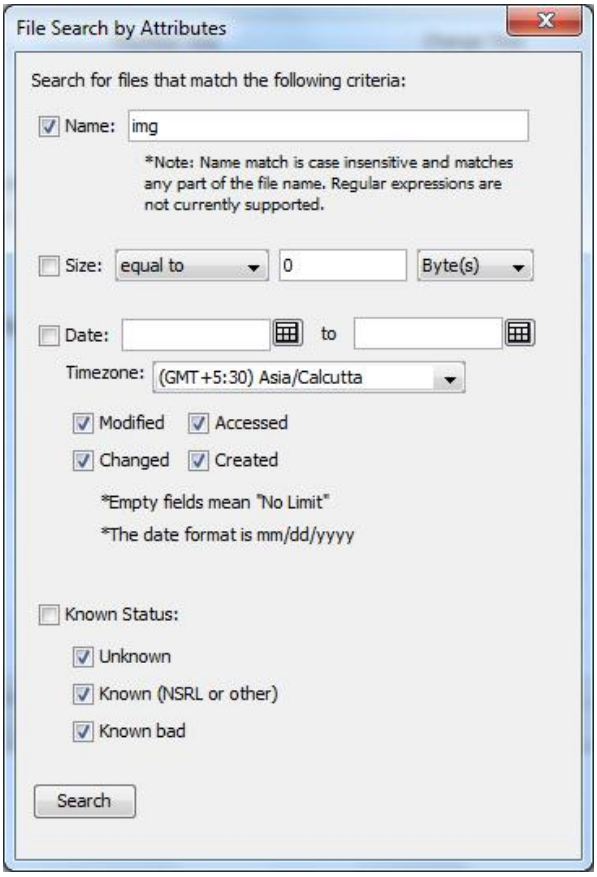
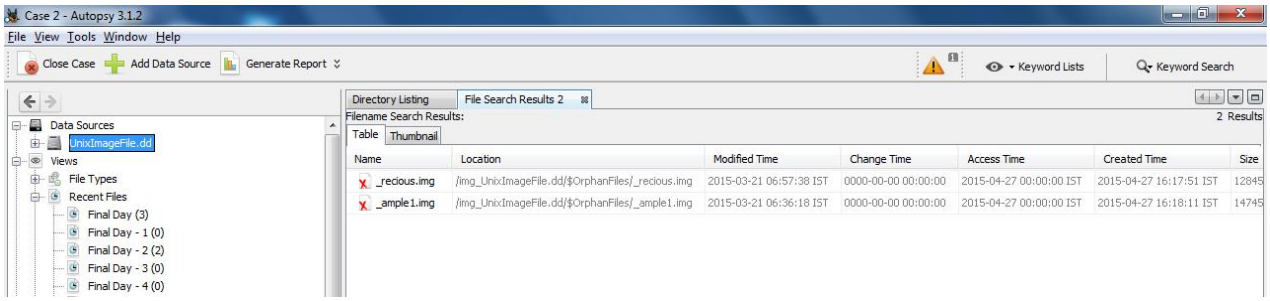
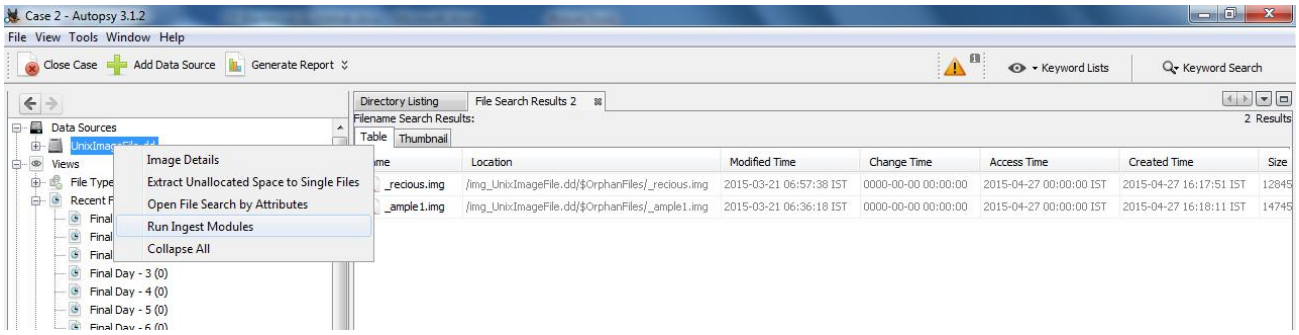


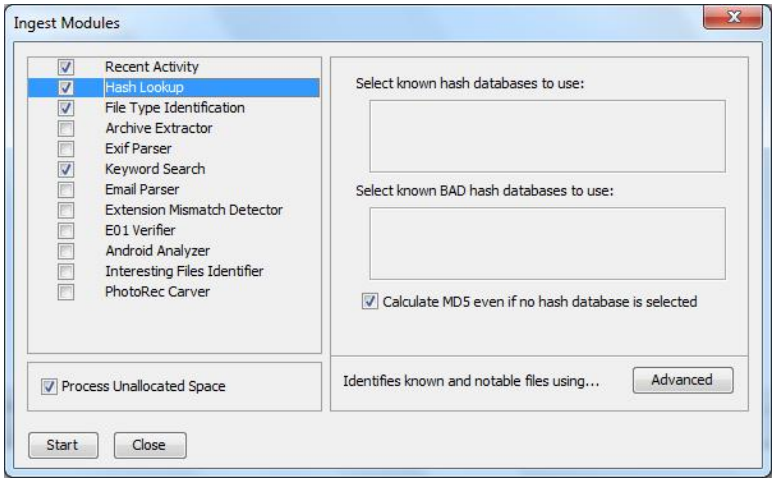
Image Details
Image Information
Name:
UnixImageFile.dd
Type:
Raw Single
Sector Size:
512
Total Size:
4011851776
Hash Value:
OK

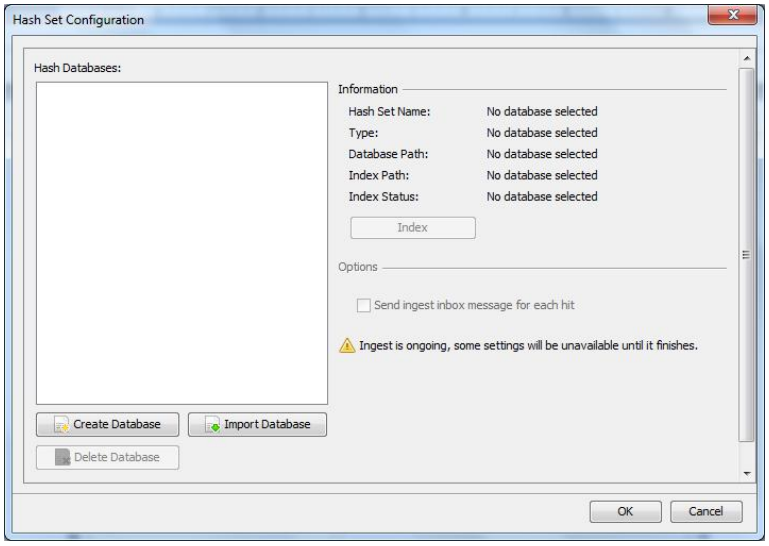

Case 2 - Autopsy 3.1.2
Close Case
Add Data Source
Generate Report
Directory Listing
File Search Results 2
/img_UnixImageFile.dd
Table
Thumbnail
Name
Modified Time
Change Time
Image Details
Extract Unallocated Space to Single Files
Open File Search by Attributes
Run Ingest Modules
Collapse All

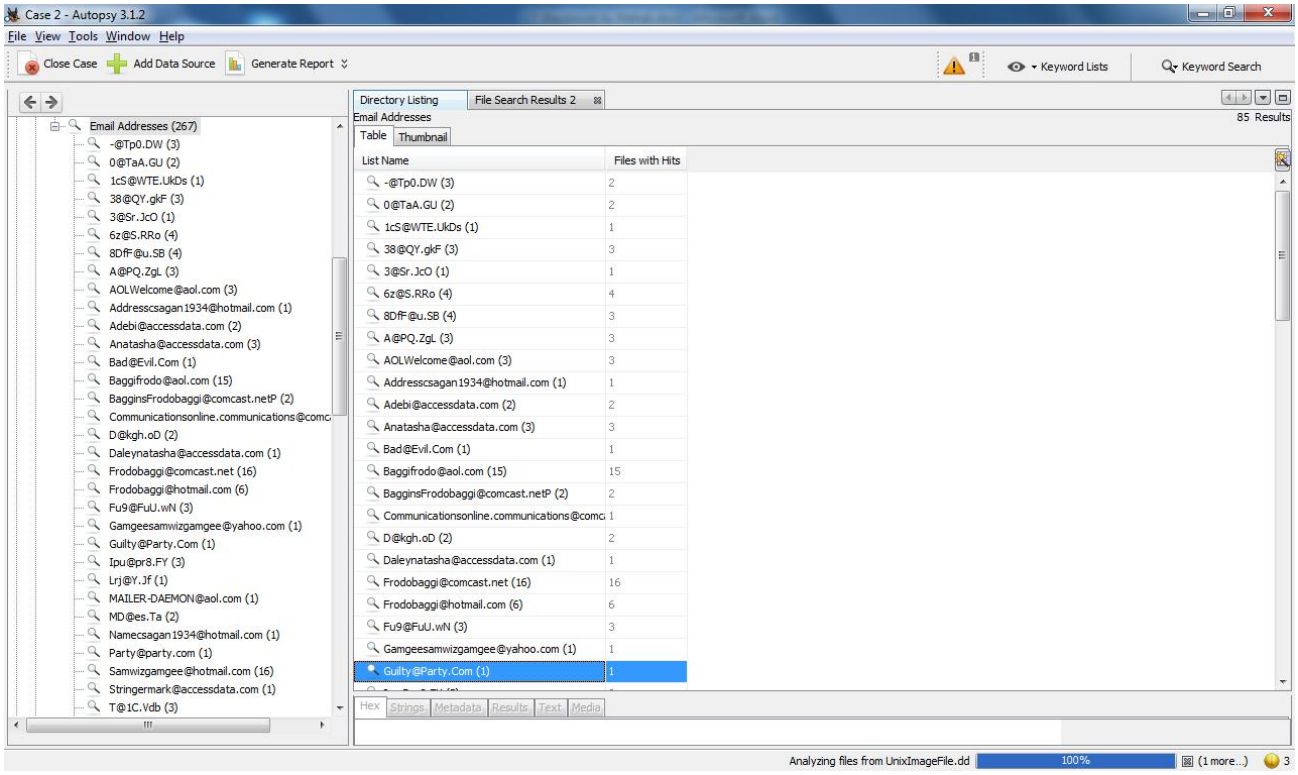

File Search by Attributes
Search for files that match the following criteria:
Name: img
*Note: Name match is case insensitive and matches any part of the file name. Regular expressions are not currently supported.
Size: equal to 0 Byte(s)
Date: to
Timezone: (GMT+5:30) Asia/Calcutta
Modified Accessed
Changed Created
*Empty fields mean "No Limit"
*The date format is mm/dd/yyyy
Known Status:
Unknown
Known (NSRL or other)
Known bad
Search

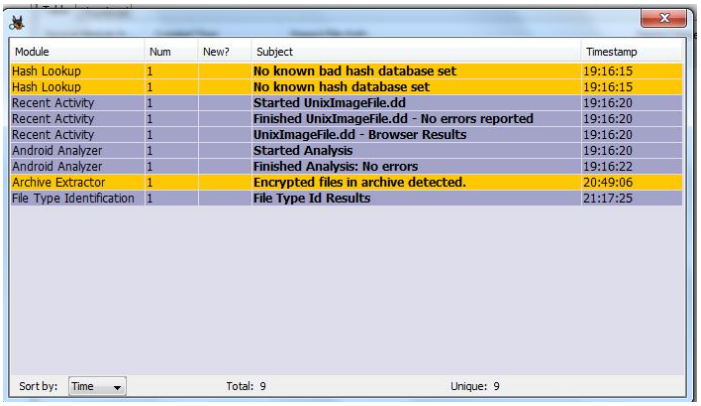

Case 2 - Autopsy 3.1.2
File View Tools Window Help
Close Case
Add Data Source
Generate Report
Keyword Lists
Keyword Search
Directory Listing
File Search Results 2
Filename Search Results:
2 Results
Table
Thumbnail
Name
Location
Modified Time
Change Time
Access Time
Created Time
Size
_recious.img
/img_UnixImageFile.dd/$OrphanFiles/_recious.img
2015-03-21 06:57:38 IST
0000-00-00 00:00:00
2015-04-27 00:00:00 IST
2015-04-27 16:17:51 IST
_ample1.img
/img_UnixImageFile.dd/$OrphanFiles/_ample1.img
2015-03-21 06:36:18 IST
2015-04-27 16:18:11 IST
Data Sources
UnixImageFile.dd
Views
File Types
Recent Files
Final Day (3)
Final Day - 1 (0)
Final Day - 2 (2)
Final Day - 3 (0)
Final Day - 4 (0)

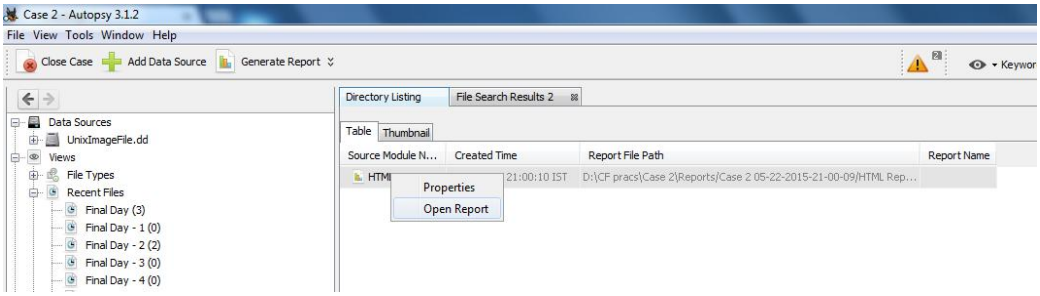

Case 2 - Autopsy 3.1.2
File View Tools Window Help
Close Case
Add Data Source
Generate Report
Keyword Lists
Keyword Search
Directory Listing
File Search Results 2
Filename Search Results:
2 Results
Table
Thumbnail
Location
Modified Time
Change Time
Access Time
Created Time
Size
_recious.img
/img_UnixImageFile.dd/$OrphanFiles/_recious.img
2015-03-21 06:57:38 IST
0000-00-00 00:00:00
2015-04-27 00:00:00 IST
2015-04-27 16:17:51 IST
_ample1.img
/img_UnixImageFile.dd/$OrphanFiles/_ample1.img
2015-03-21 06:36:18 IST
2015-04-27 16:18:11 IST
Data Sources
Views
File Type
Recent F
Image Details
Extract Unallocated Space to Single Files
Open File Search by Attributes
Run Ingest Modules
Collapse All
Final Day - 3 (0)
Final Day - 4 (0)
Final Day - 5 (0)

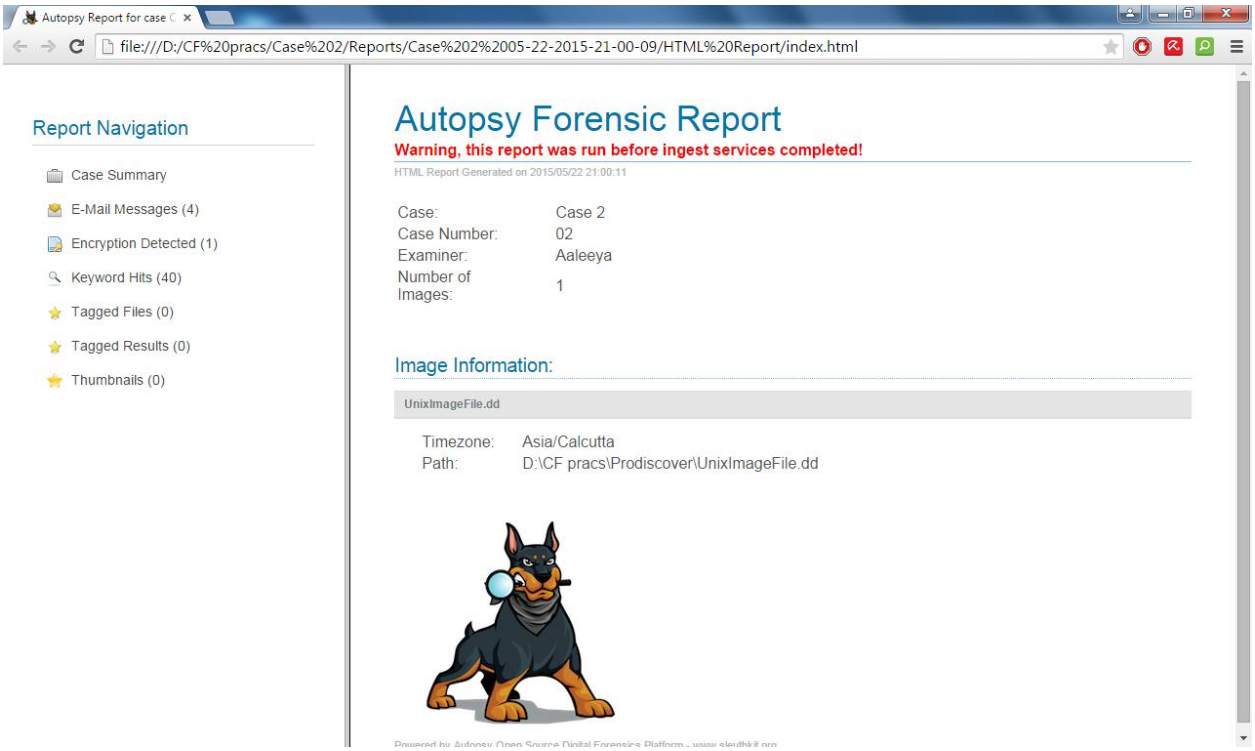

Ingest Modules
Recent Activity
Hash Lookup
File Type Identification
Archive Extractor
Exif Parser
Keyword Search
Email Parser
Extension Mismatch Detector
E01 Verifier
Android Analyzer
Interesting Files Identifier
PhotoRec Carver
Select known hash databases to use:
Select known BAD hash databases to use:
Calculate MD5 even if no hash database is selected
Process Unallocated Space
Identifies known and notable files using...
Advanced
Start
Close


Hash Set Configuration
Hash Databases:
Information
Hash Set Name: No database selected
Type: No database selected
Database Path: No database selected
Index Path: No database selected
Index Status: No database selected
Index
Options
Send ingest inbox message for each hit
Ingest is ongoing, some settings will be unavailable until it finishes.
Create Database
Import Database
Delete Database
OK
Cancel


Case 2 - Autopsy 3.1.2
File View Tools Window Help
Close Case
Add Data Source
Generate Report
Keyword Lists
Keyword Search
Email Addresses (267)
Directory Listing
File Search Results 2
Email Addresses
85 Results
Table
Thumbnail
List Name
Files with Hits
Analyzing files from UnixImageFile.dd
100%


Module Num New? Subject Timestamp
Hash Lookup 1 No known bad hash database set 19:16:15
Hash Lookup 1 No known hash database set 19:16:15
Recent Activity 1 Started UnixImageFile.dd 19:16:20
Recent Activity 1 Finished UnixImageFile.dd - No errors reported 19:16:20
Recent Activity 1 UnixImageFile.dd - Browser Results 19:16:20
Android Analyzer 1 Started Analysis 19:16:20
Android Analyzer 1 Finished Analysis: No errors 19:16:22
Archive Extractor 1 Encrypted files in archive detected. 20:49:06
File Type Identification 1 File Type Id Results 21:17:25
Sort by: Time
Total: 9
Unique: 9

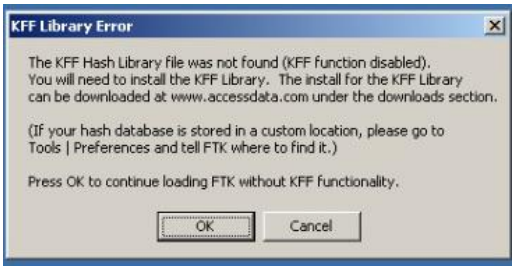

Case 2 - Autopsy 3.1.2
File View Tools Window Help
Close Case
Add Data Source
Generate Report
Directory Listing
File Search Results 2
Table
Thumbnail
Source Module N...
Created Time
Report File Path
Report Name
Data Sources
UnixImageFile.dd
Views
File Types
Recent Files
Final Day (3)
Final Day - 1 (0)
Final Day - 2 (2)
Final Day - 3 (0)
Final Day - 4 (0)
Properties
Open Report


Autopsy Report for case
file:///D:/CF%20pracs/Case%202/Reports/Case%202%2005-22-2015-21-00-09/HTML%20Report/index.html
Report Navigation
Case Summary
E-Mail Messages (4)
Encryption Detected (1)
Keyword Hits (40)
Tagged Files (0)
Tagged Results (0)
Thumbnails (0)
Autopsy Forensic Report
Warning, this report was run before ingest services completed!
HTML Report Generated on 2015/05/22 21:00:11
Case: Case 2
Case Number: 02
Examiner: Aaleeya
Number of Images: 1
Image Information:
UnixImageFile.dd
Timezone: Asia/Calcutta
Path: D:\CF pracs\Prodiscover\UnixImageFile.dd

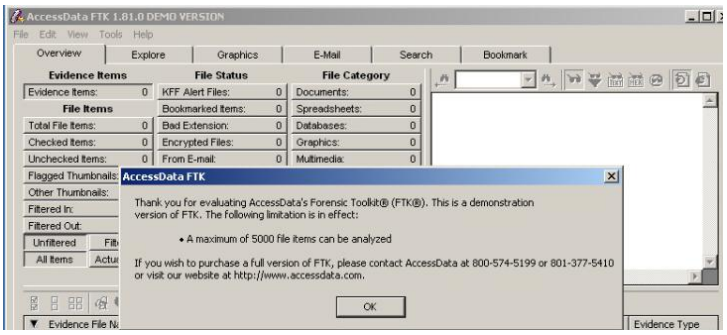
Practical No: 2 A

Step 1: Start Forensic Toolkit.

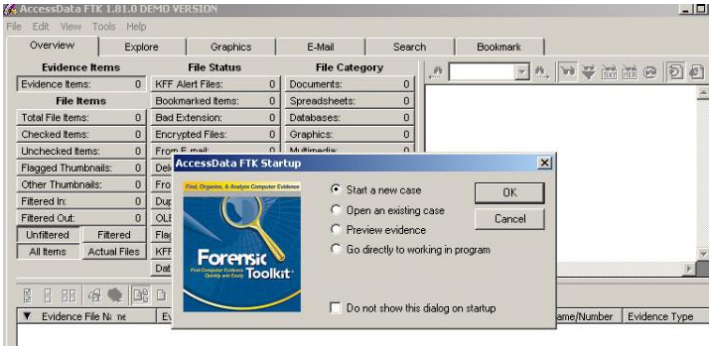
Step 2: Here, prompted with a warning dialog box, click on OK to continue.



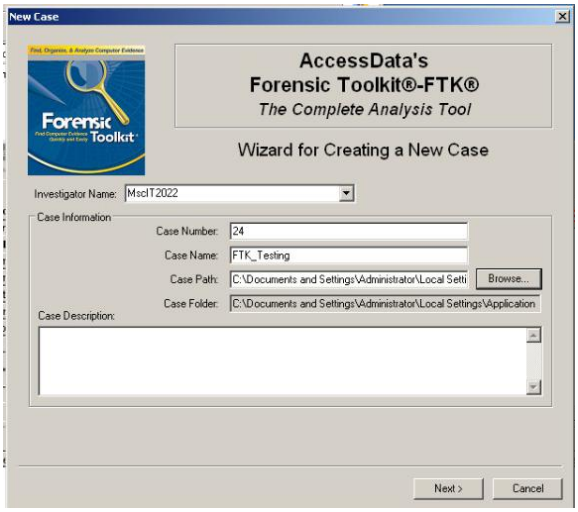
Step 3: click on OK button.



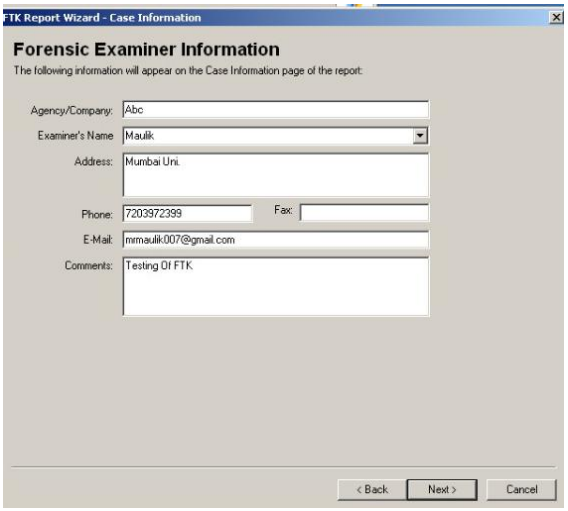
Step 4: Now select Start New Case option and click on ok.



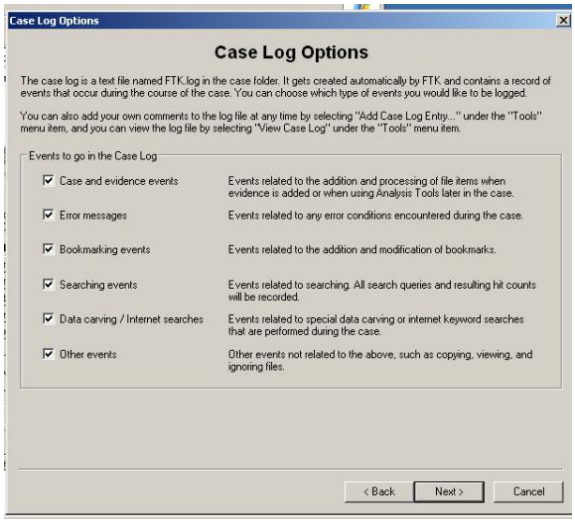
Step 5: Enter the detail for a New case.



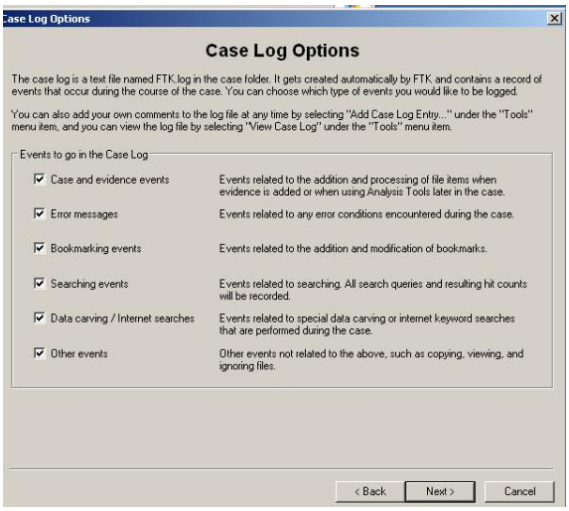
Step 6: Fill the information in Forensic Examiner Information dialog box.



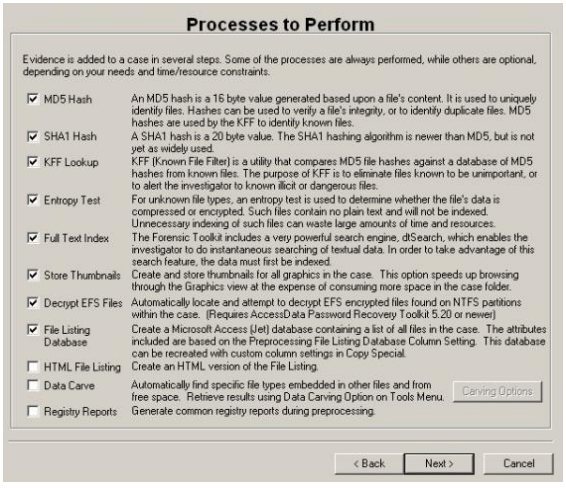
Step 7: leave the default settings and click on next.



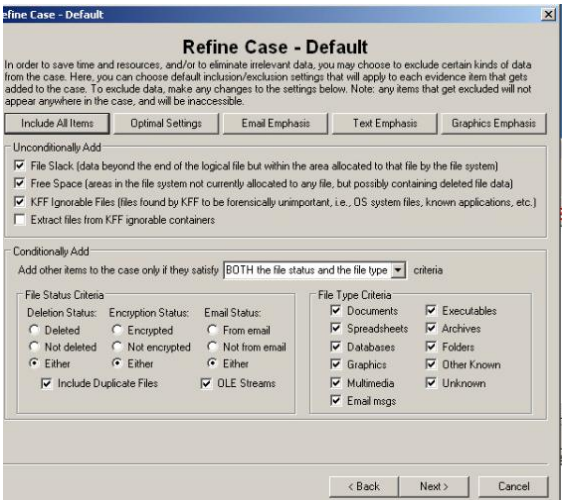
Step 8: Now again leave the default settings and click on next.



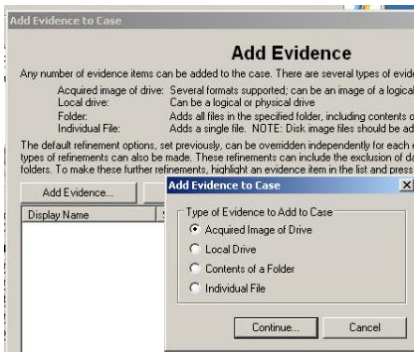
Step 9: In the Refine Case-Default, click the Include All items button and then click Next.



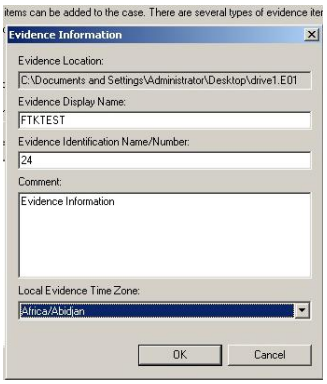
Step 10: In Refine Index-Default, accept the default settings and click Next.



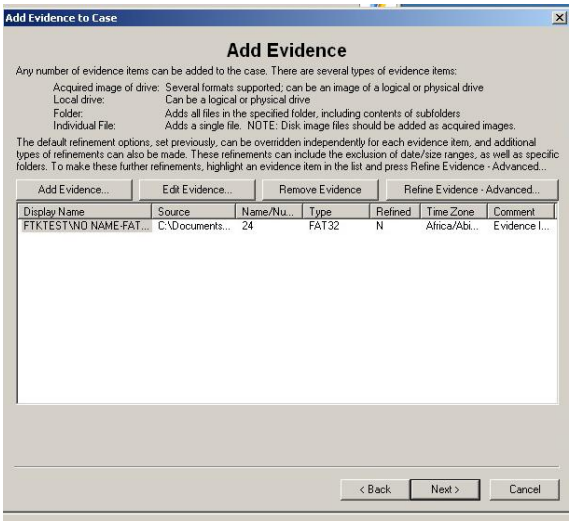
Step 11: Now here Click on add Evidence button.



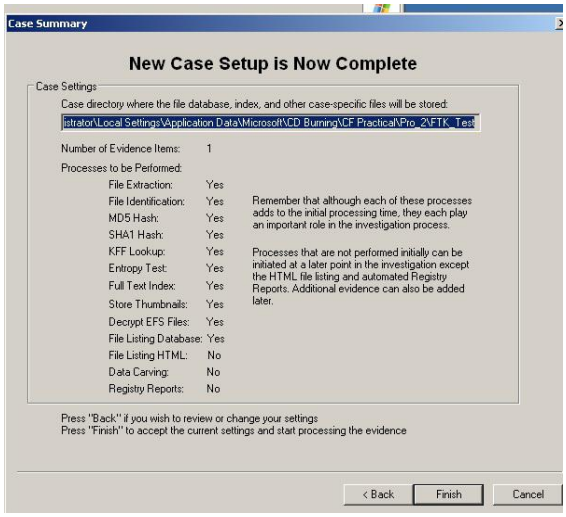
Step 12: Enter Evidence Information and click on OK button.



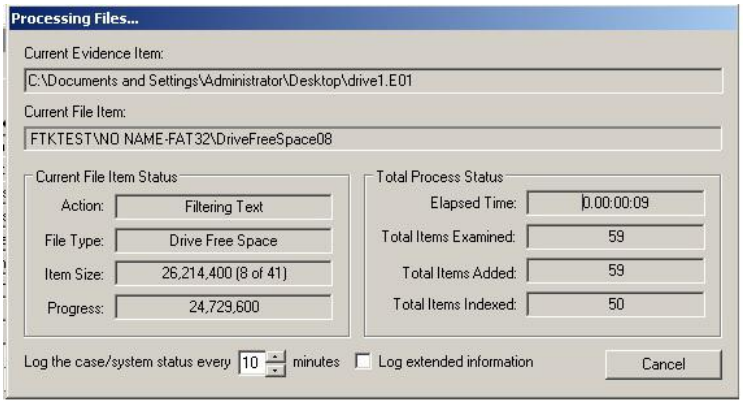
Step 13: Now click on Next.



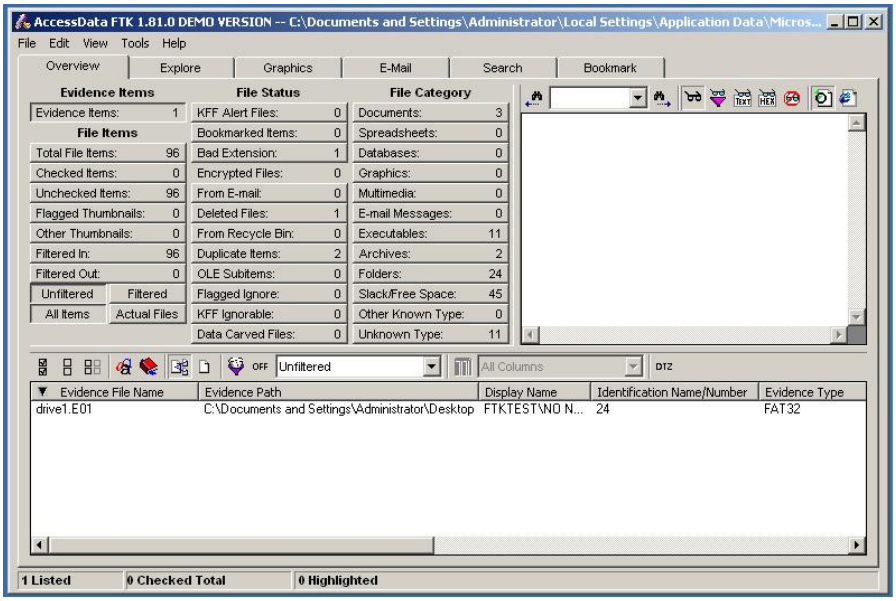
Step 14: Click on Finish to initiate the analysis.



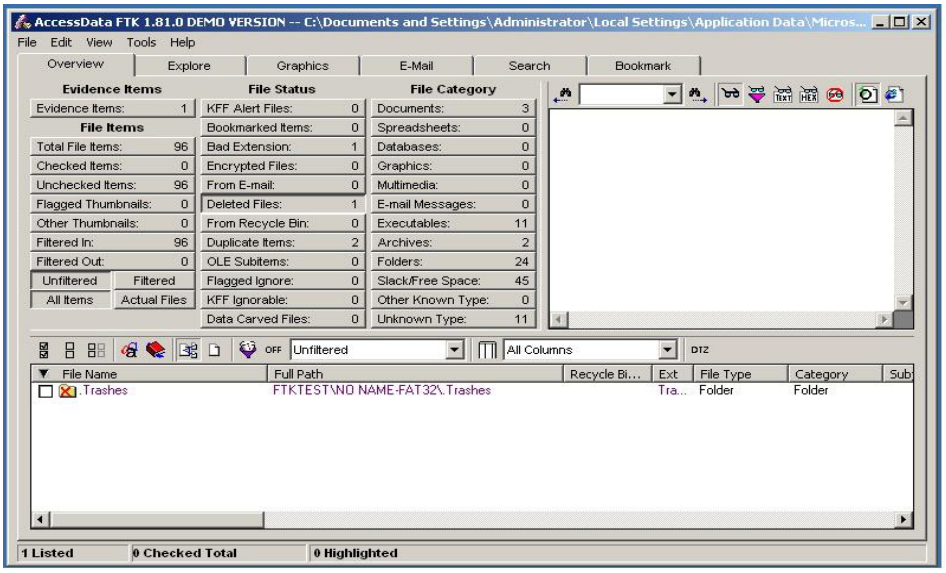
Step 15: Now Processing Will Start.



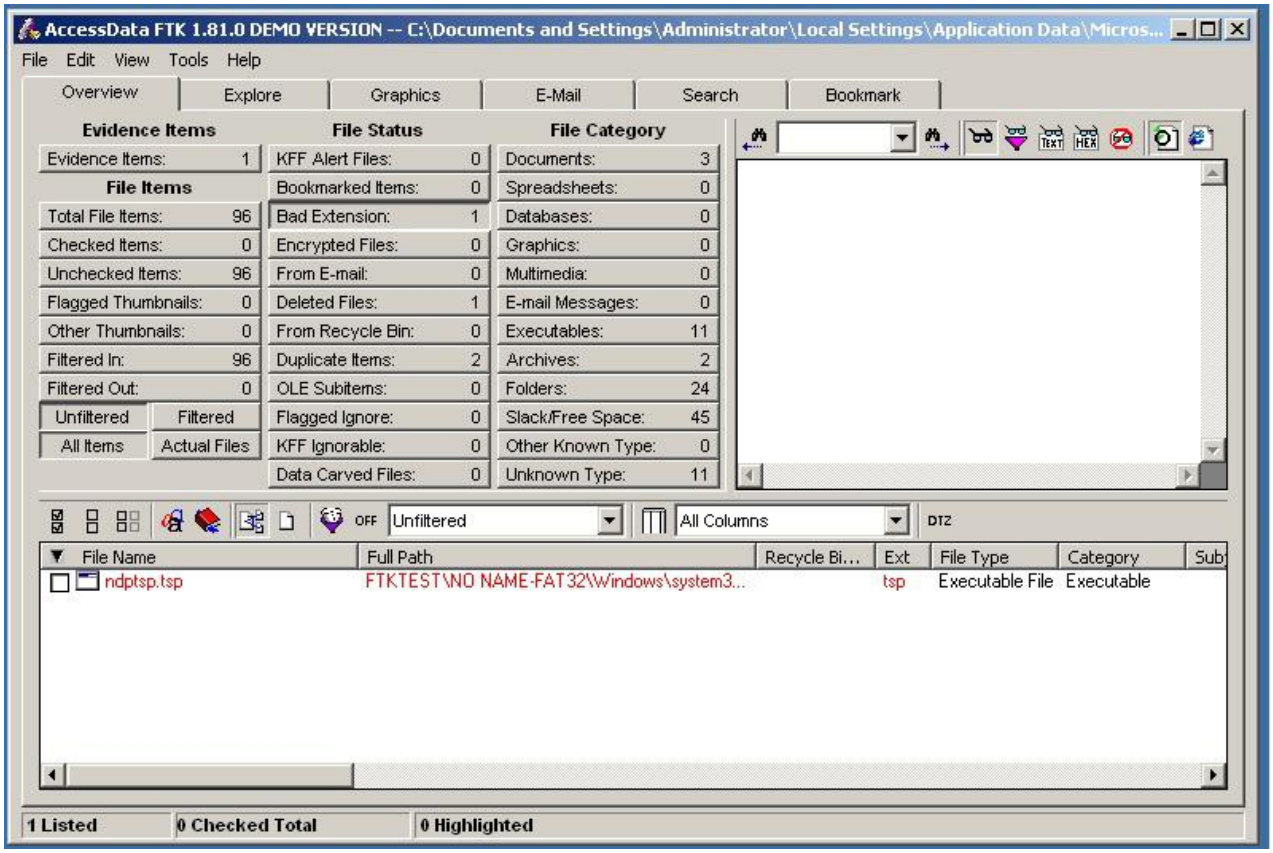
Step 16: when FTK finishes the processing part, the FTK window opens to the Overview tab.



Step 17: Select Deleted Files option to explore the evidence items.



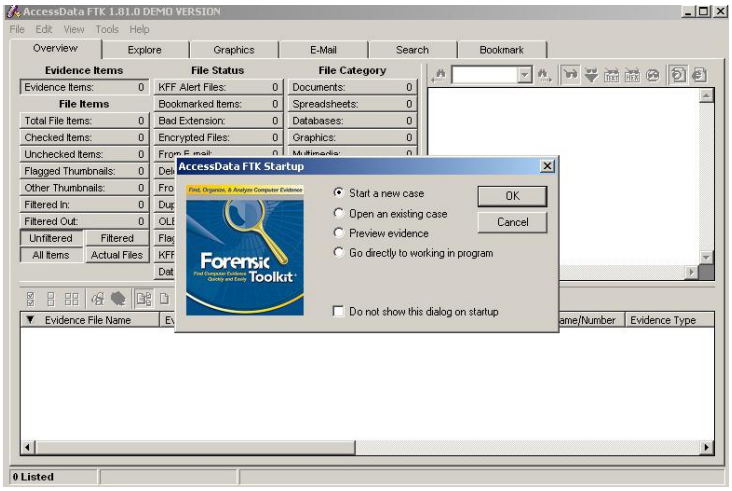
Step 18: Select Bad Extension Files to view.



Practical No: 3

Aim: Using Forensic Toolkit FTK & Write a report using FTK(AccessData FTK).

Step 1: First we need to Create a Case.



To provide the new case information:

1 In the Investigator Name field, type the name of the investigator. The drop-down list contains the name of investigators that have been entered in prior cases. If the investigator has worked on other cases in FTK, select the name from the list.

2: In the Case Number field, enter the case number for reference.

3 In the Case Name field, enter the name of the case. The name cannot contain the following characters: " > ? / : \ | < The case name also becomes the name of the folder where all case information will be stored.

4 Next to the Case Path field, click Browse to select the path where the evidence will be stored. By default, all FTK cases are stored in that directory.

5 Verify that the Case Folder field lists the folder where you want the case to be stored. Each case is stored in a separate folder and should be kept distinct from other cases. The Case Folder field is based on the Case Name and Case Path fields. To make changes to the Case Folder, change the Case Name and Case Path fields.

6 (Optional) In the Case Description field, add information that will be helpful to the analysis of the case. This field is particularly useful if several people work on the case. This field is included in the report created at the end of the case investigation. Click Next.

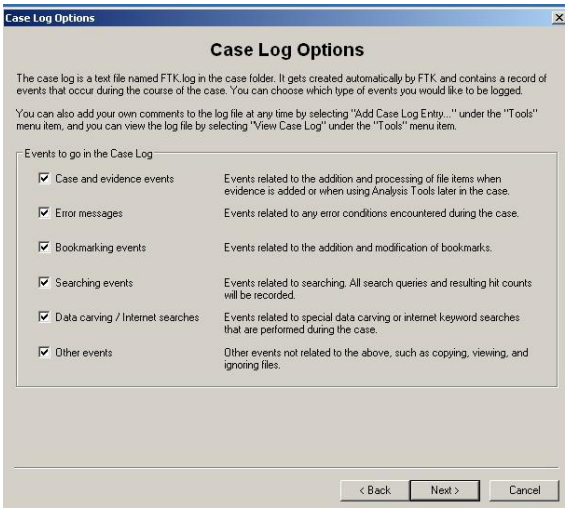
Entering Forensic Examiner Information:

Selecting Case Log Options:

The Case Log Options form allows you to select which events you want FTK to log for the current case. FTK maintains a log file of FTK events such as bookmarking items, searches, and error messages for each case.

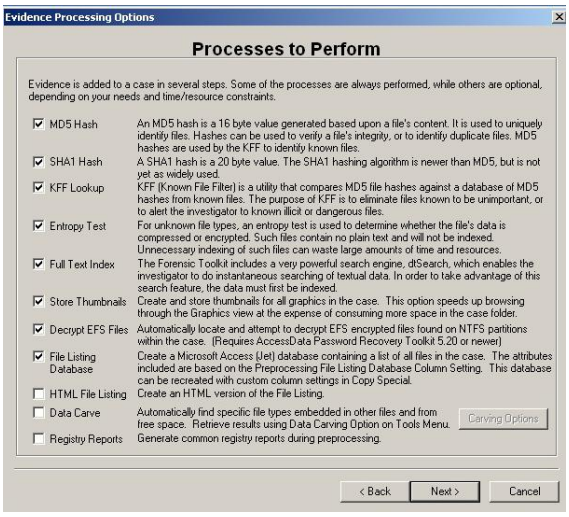
The following table outlines the Case Log Options form:

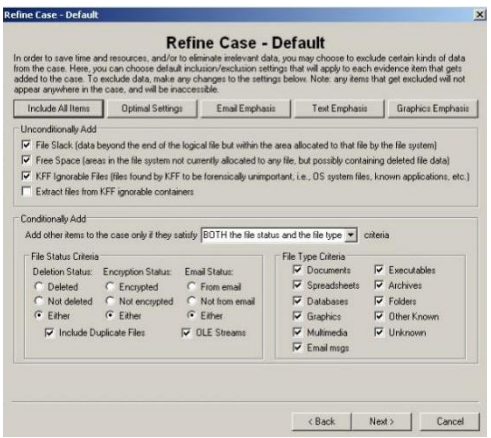
Option	Description
Bookmarking Events	Events related to the addition and modification of bookmarks.
Case and Evidence Events	Events related to the addition and processing of file items when evidence is added or when using the Analysis Tools.
Data Carving/Internet Searches	Events related to data carving or Internet keyword searches that are performed during the case.
Error Messages	Events related to any errors encountered during the case.
Other Events	The following events: • Copy special • Exporting files • Viewing items in the detached viewer • Ignoring and unignoring files
Searching Events	All search queries and resulting hit counts.



Selecting Evidence Processes:

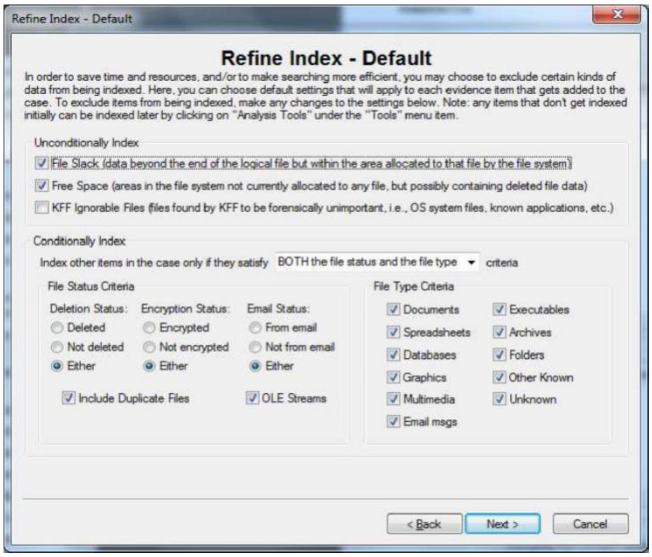
The Evidence Processing Options form allows you to select which processes you want to perform on the current evidence. You only need to select those processes that are relevant to the evidence you are adding to the case. For example, if your case is primarily a graphics case, there is no need to index the evidence.





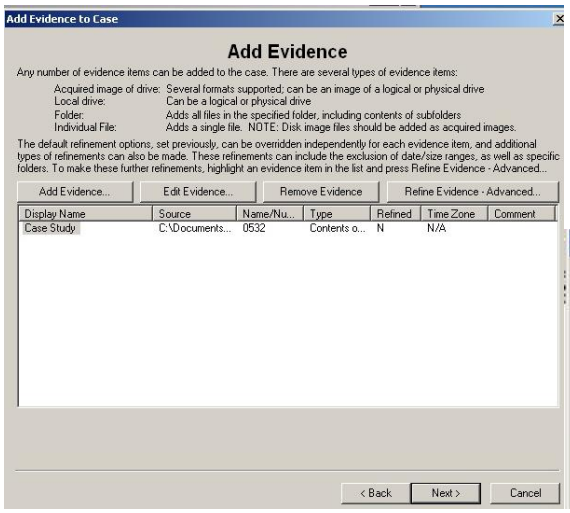
Refining the Index:

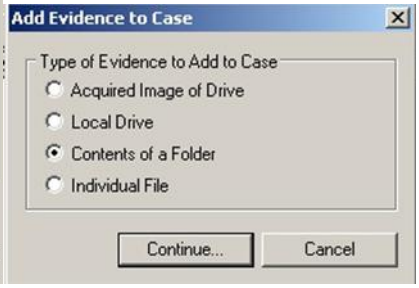
The Refine Index form allows you to specify types of data that you do not want to index. You might choose to exclude data to save time and resources and to increase searching efficiency.

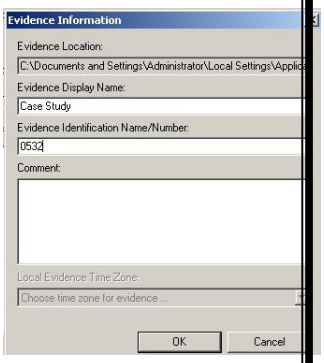


Managing Evidence :

As your investigation progresses, you will want to edit the information you entered for your evidence. Evidence is managed through the Add Evidence forms.

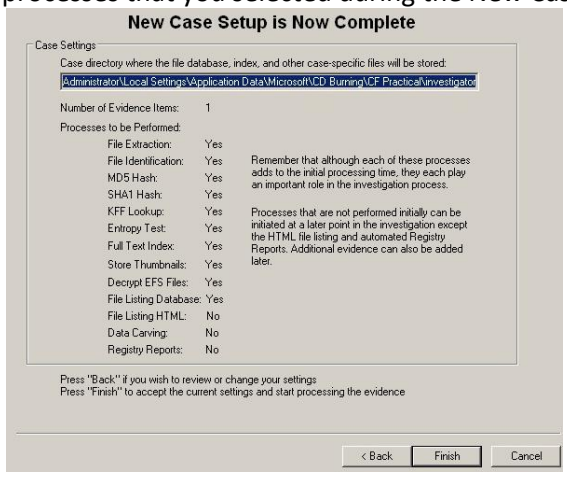






Reviewing Case Summary:

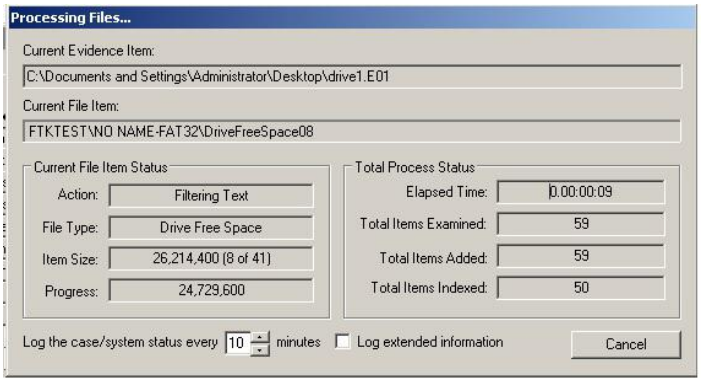
The Case Summary form allows you to review the evidence directory, number of evidence items, and evidence processes that you selected during the New Case Wizard.

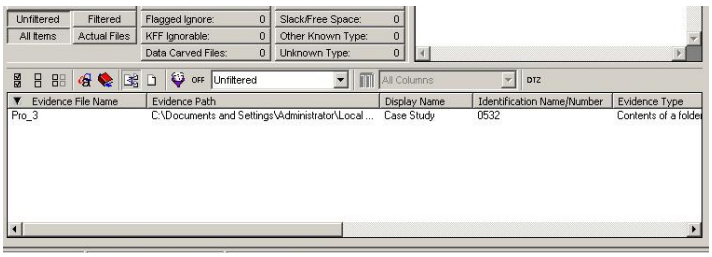


Processing the Evidence :

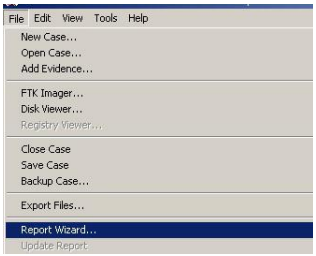
After you click Finish, the Processing Files form appears and displays the status of the processes you selected in the wizard.

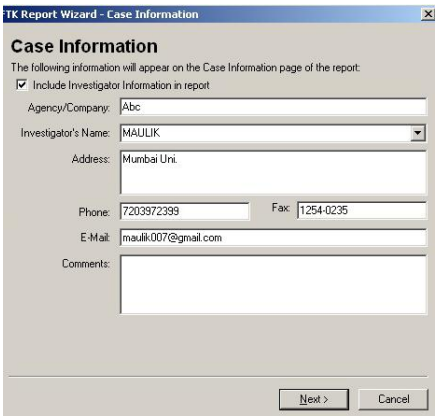
1.From the menu, select Report, and then Generate Report or click the button on the toolbar.



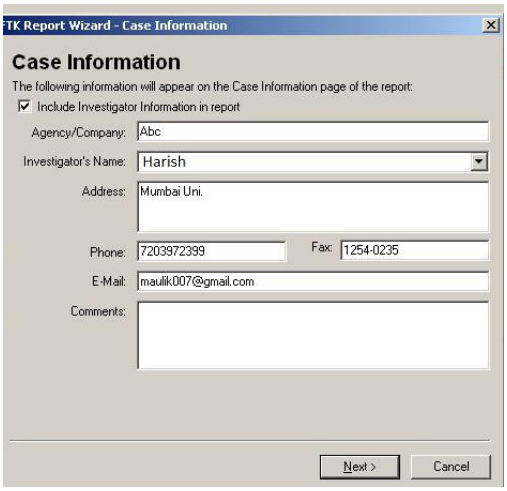


2.The Case Information dialog appears.

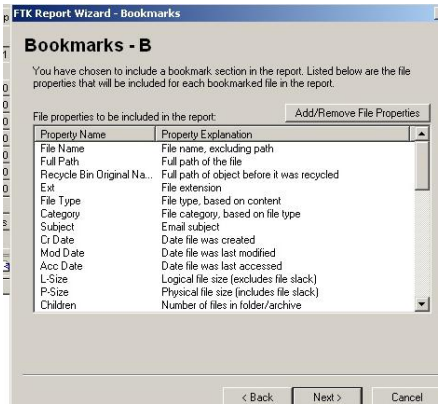




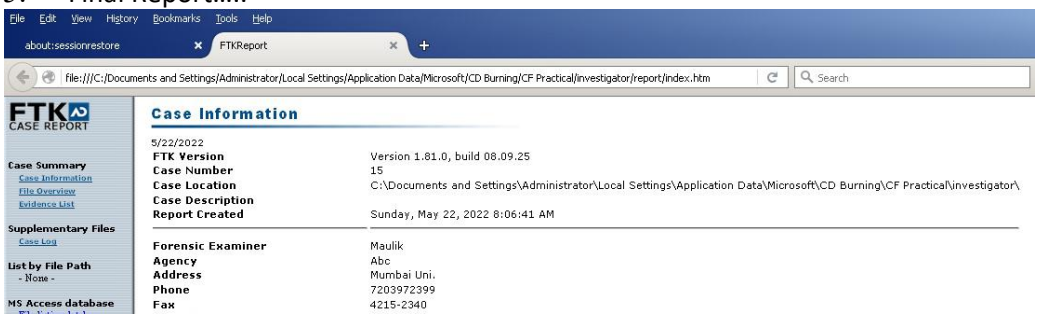
3. The Bookmarks-A dialog appears.



4. The Bookmarks-B dialog appears



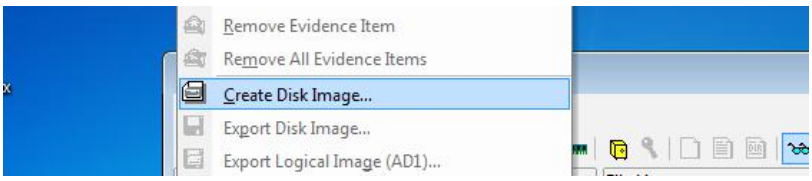
5. Final Report…..



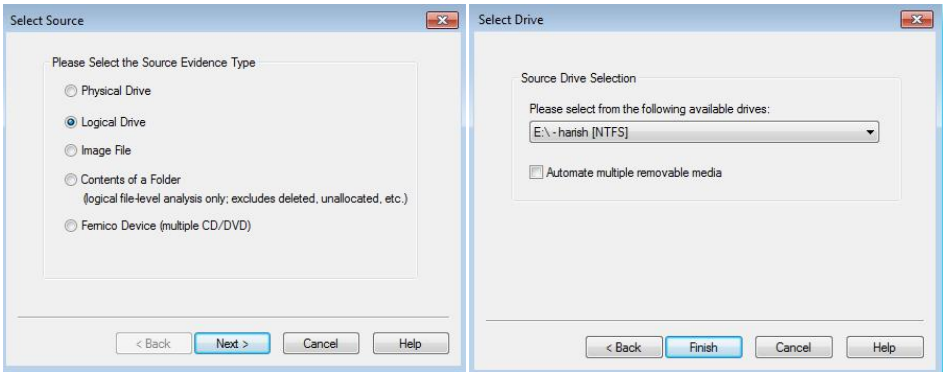
Practical No. 4A

Step 1:Download and install Accessdata FTK Imager on windows virtual machine.

Step 2: Click On File > Create Disk Image



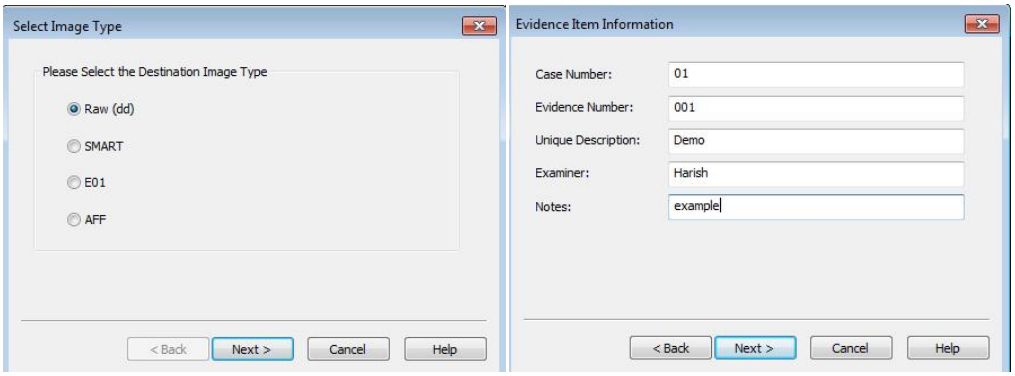
Step 3: Select logical Drive and Select the drive.

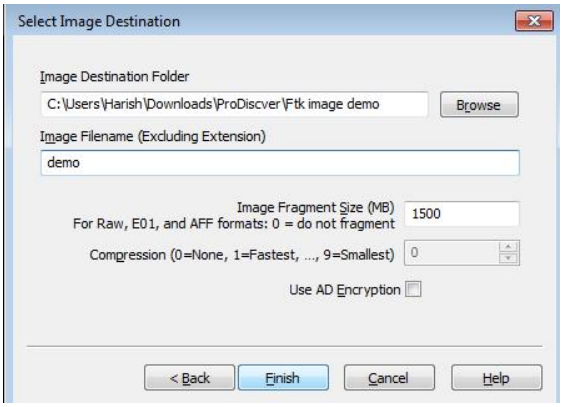


Step 4: Select the Destination for image file to be saved.In that Choose the appropriate destination image type. For our guide, we'll be deploying 'dd' which stands for disk dump.

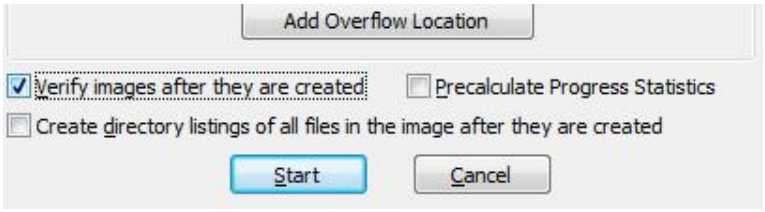
After that we need to enter the case number and other details as show below.

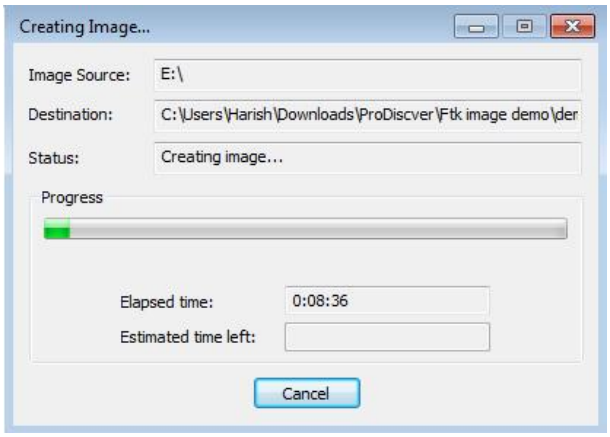
At last select the destination for saving the file.





Step 5: At last we need to select the checkbox for checking the image file after creating. Then start creating Image.





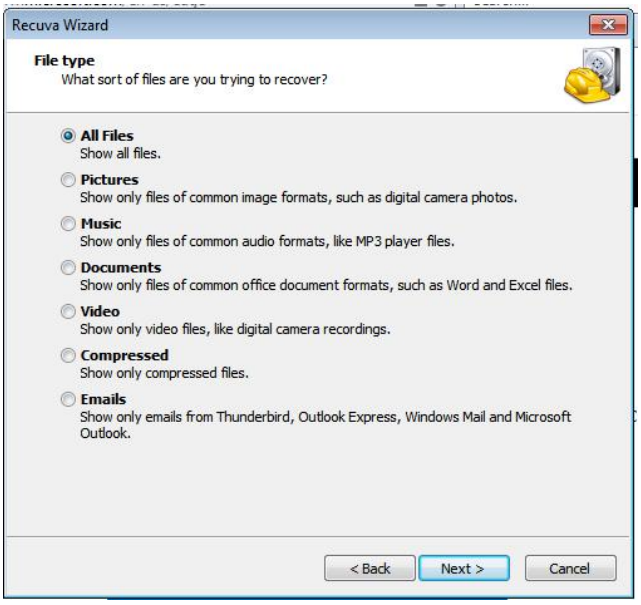
Practical No. 4B

Using Recuva

Step 1: Open the Recuva Software.

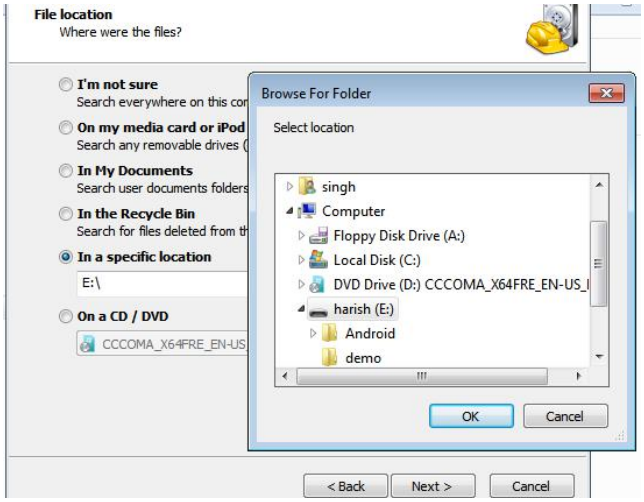
Step 2: click on next.

Step 3: Select the type of files you want to recover.

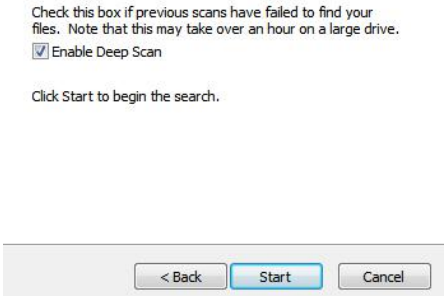


Step 4: Click the next button.

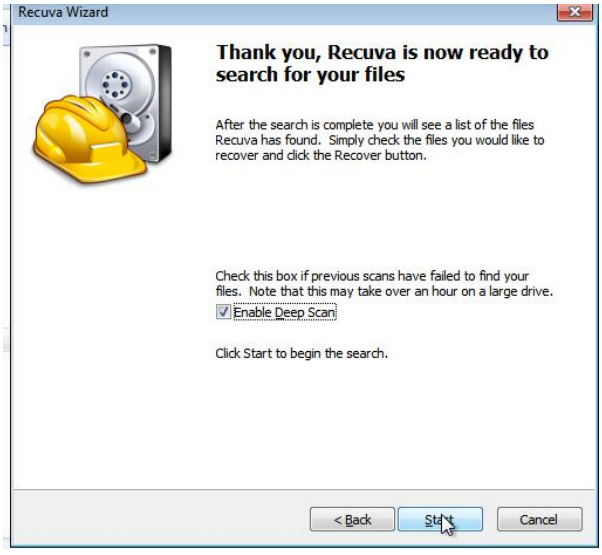
Step 5: Select the drive which we want to recover.



Step 6: check for deep scan for files.

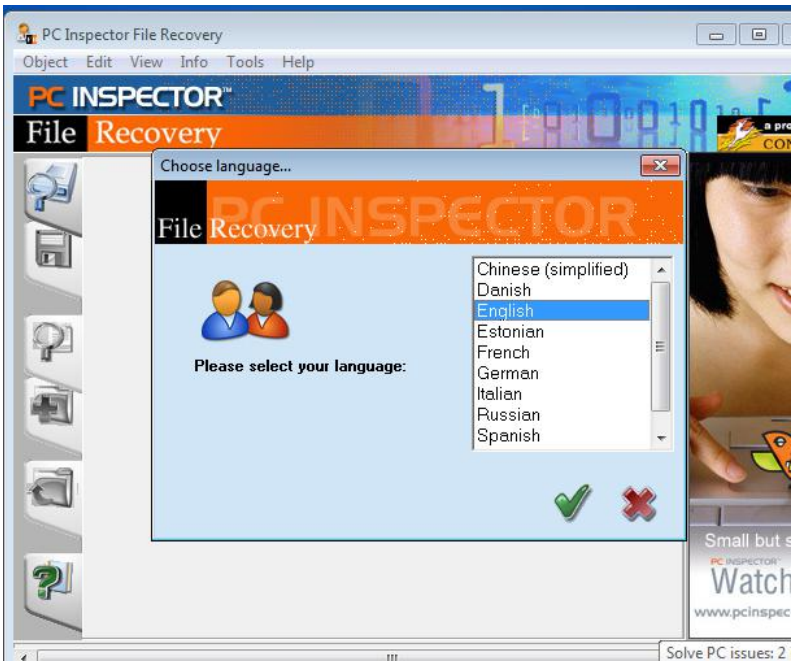


Step 7:Click start Button. It will start recovering deleted files.

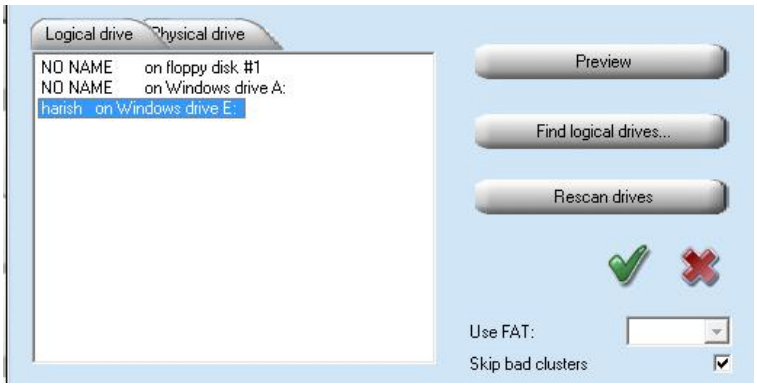


Using PC Inspector File Recovery

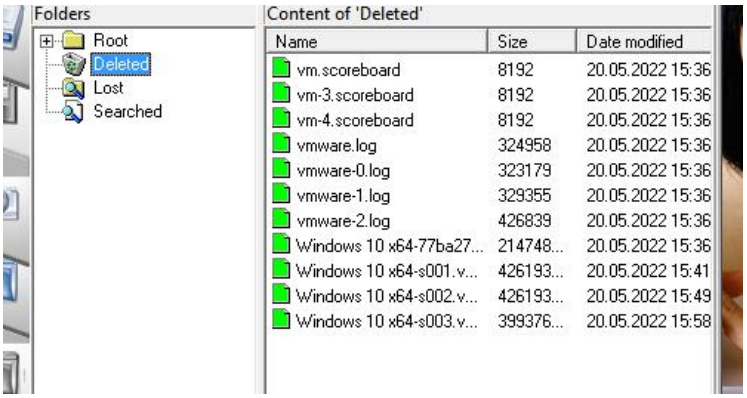
Step 1:Download and install PC inspector and open it.



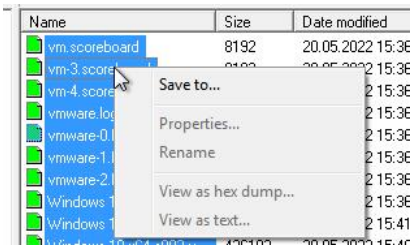
Step 2: Select the drive you want to recover files from. Ensure you've selected the right drive.

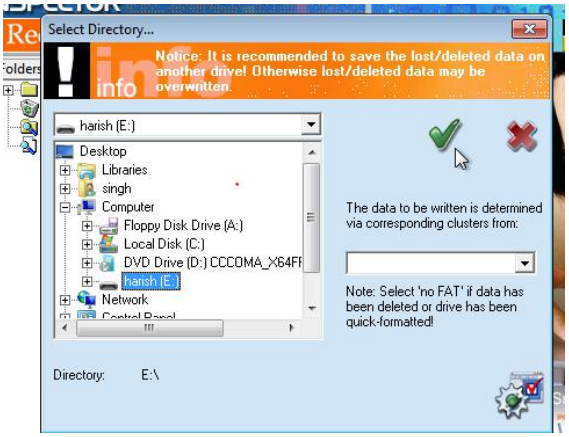


Step 3: Scan the selected drive for deleted files.



Step 4: Sort through the potentially recoverable files.And save the deleted file.

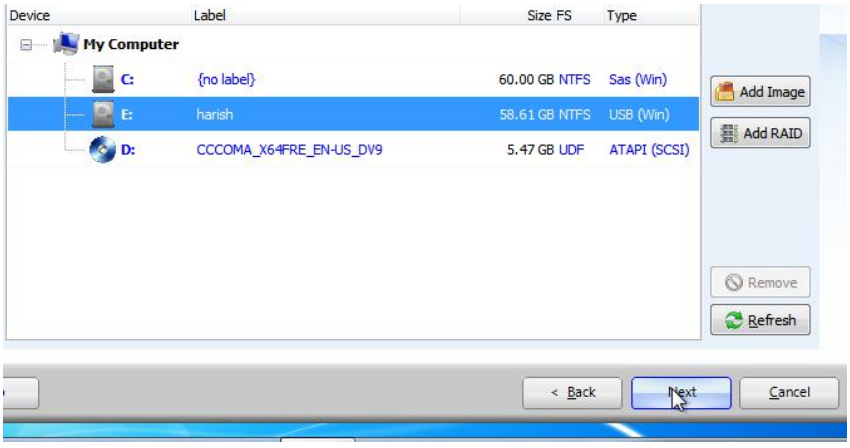




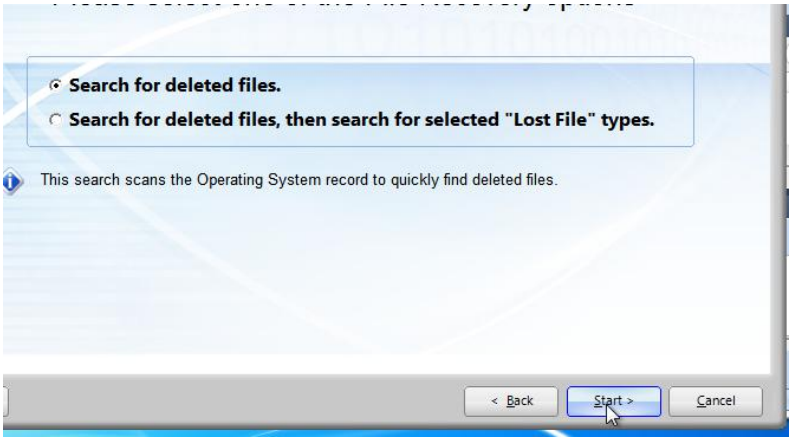
Using Recover my file.

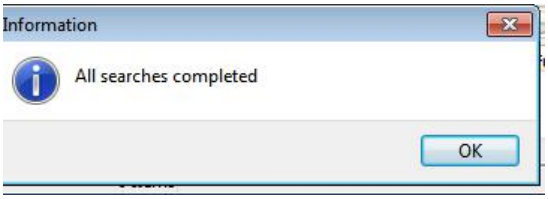
Step 1:Download and install the tool and open then select the type of file you want to recover.

Step 2: Select the file or drive you want to recover.

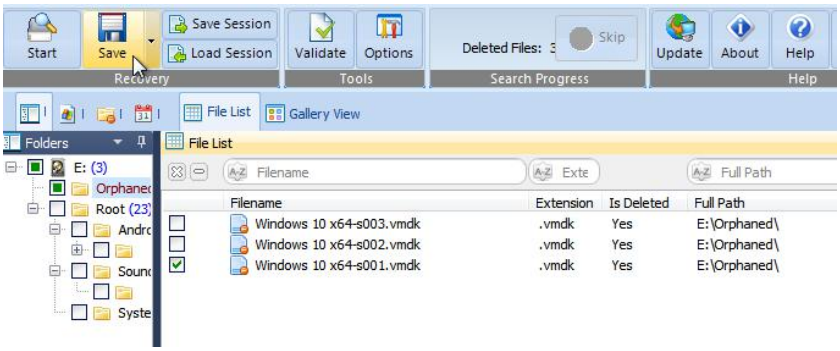


Step 3: Start search for deleted file.



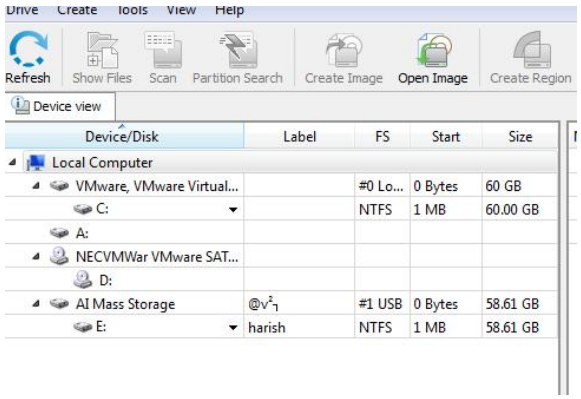


Step 4: Now you can view and recover the deleted file. Then save that original file.

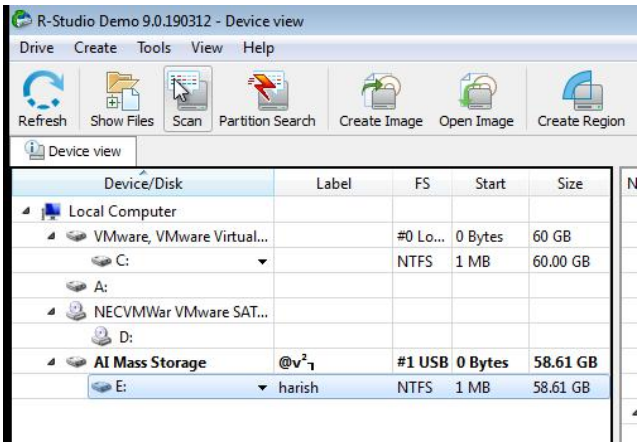


Using R-studio

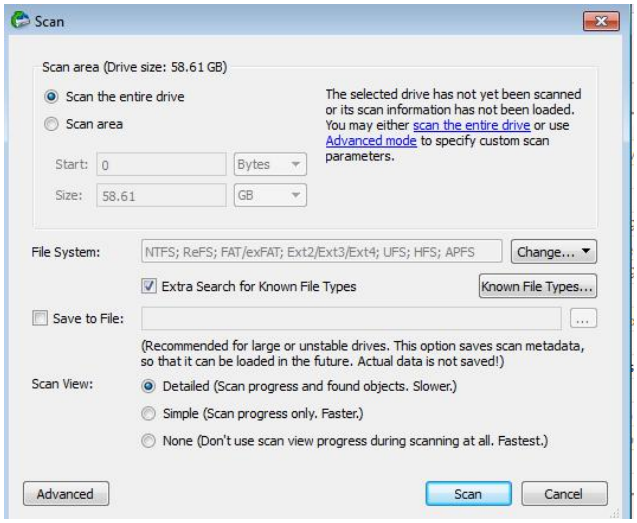
Step 1: Start R-Studio and locate the damaged disk.



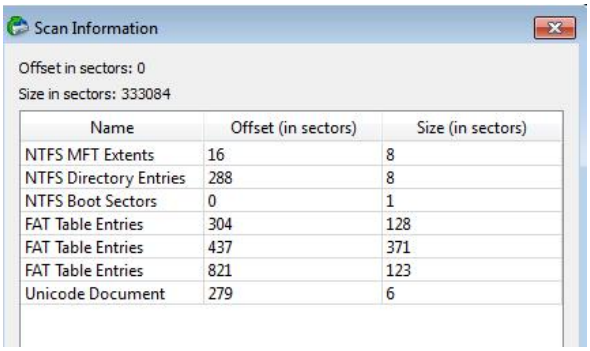
Step 2: Scan the damaged disk.



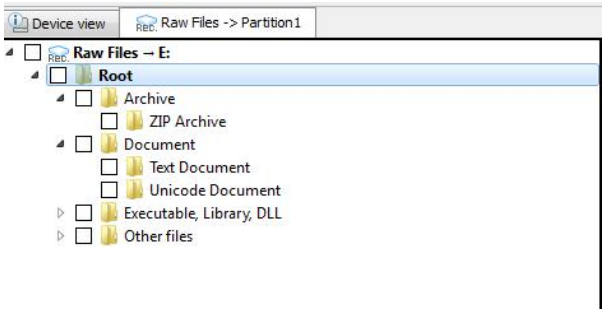
Step 3: View the search results.



Step 4: Double-click the partition to browse its contents.

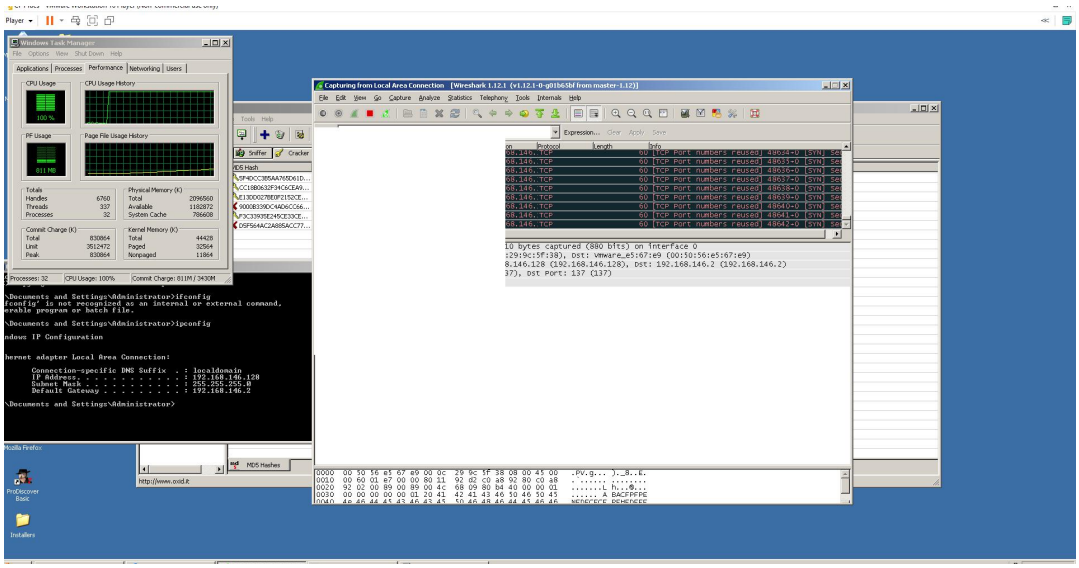


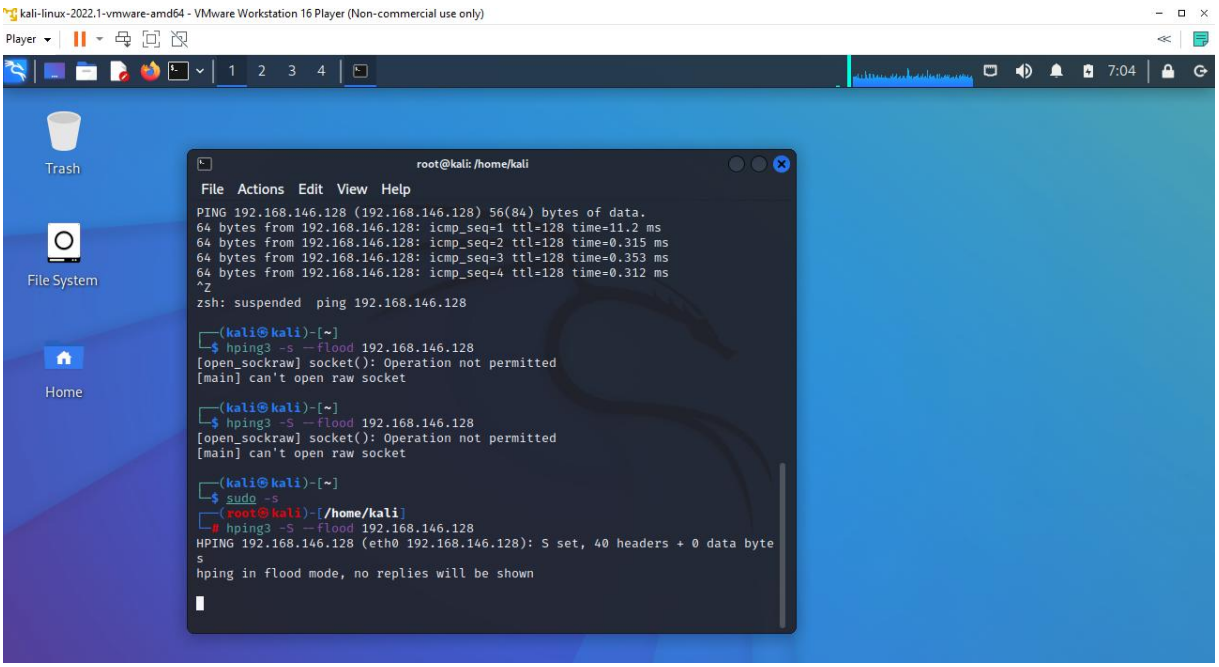
Step 5: Preview the files by double-clicking them.



Practical No. 5A

Step 1: check the ip address of virtual testing pc. And open the Wireshark Network Analyser tool.
Step 2: open another virtual installed kali linux on it.
Step 3: write command to do a Dos attack by network flood.
Step 4: sudo apt-get install hping3
Step 5: hping3 -c 15000 -d 120 -S -w 64 -p 80 --flood --rand-source 192.168.1.159

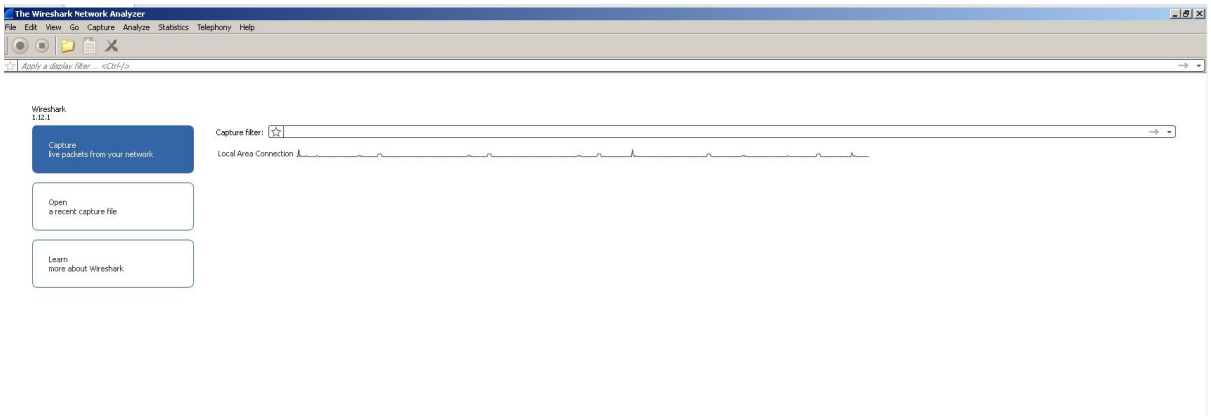




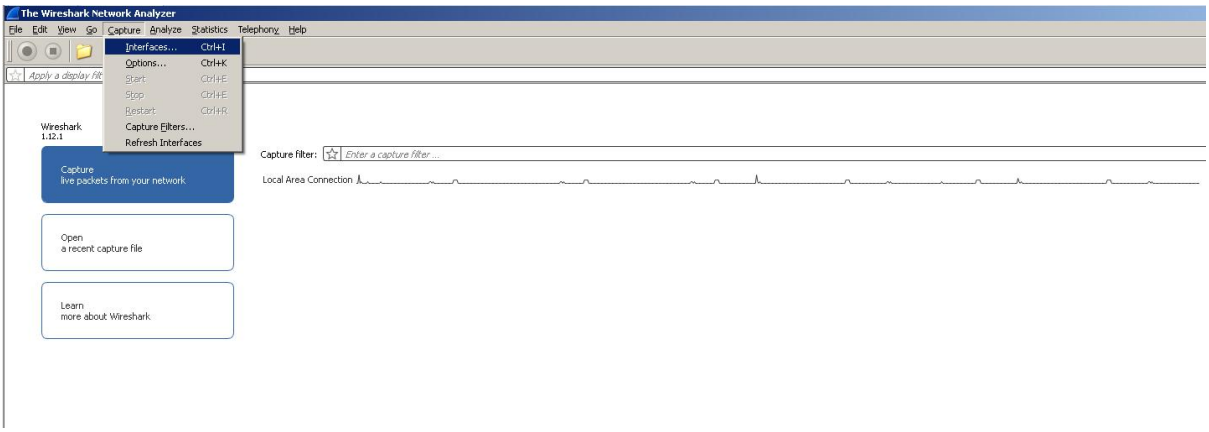
Practical No. 5B

Aim: Using Traffic Capturing and Analysis tools . [wireshark]

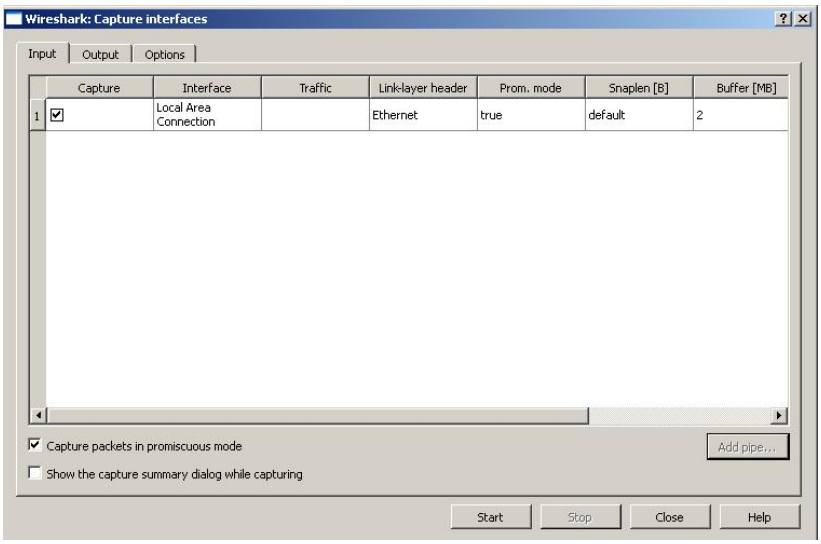
Step 1: open the wireshark

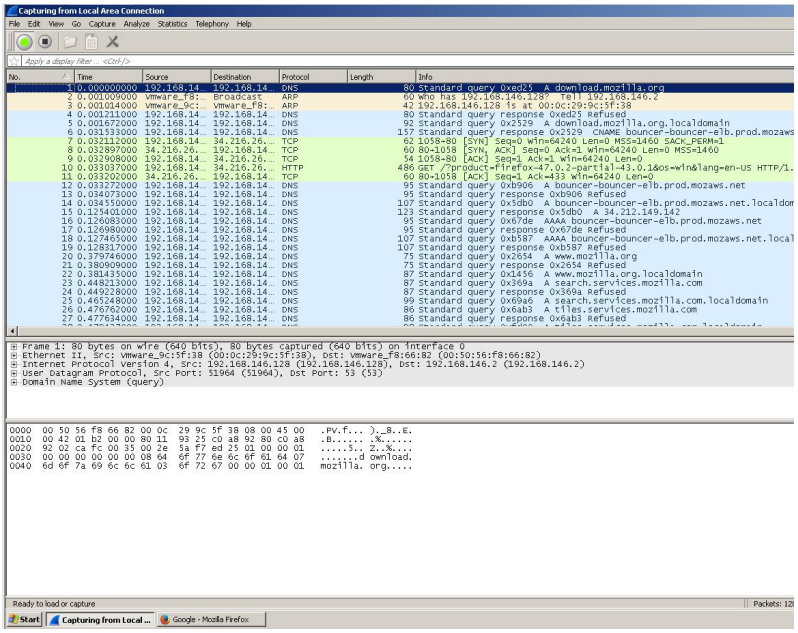


Step 2: On menu bar select Capture. Select interfaces.

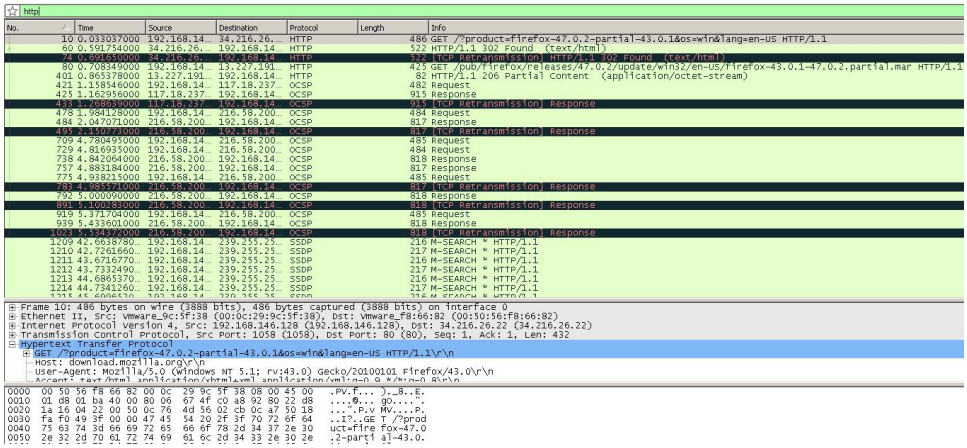


Step 3: Select Once you click on start, then Wireshark starts to capture the packets on that interface.

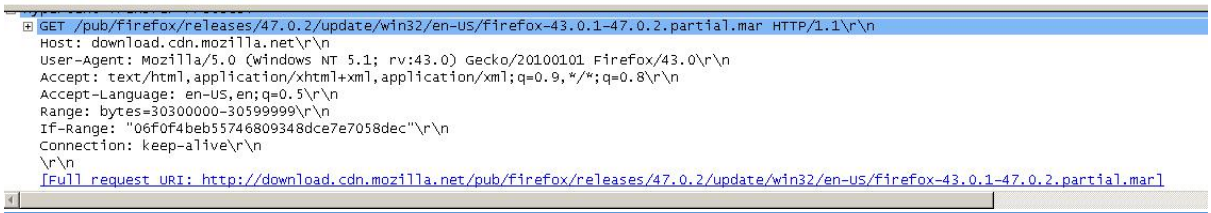




Step 4: Filter packets with HTTP protocol.

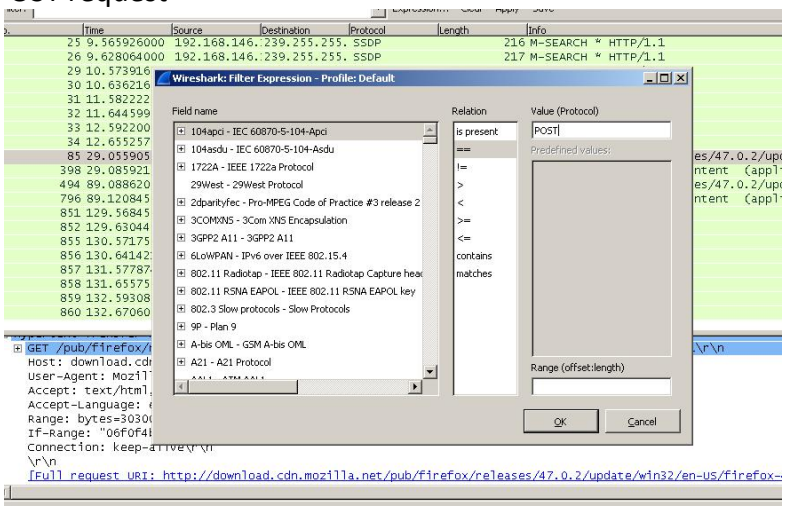


Step 5: A file with only text:
https://vulms.vu.edu.pk/Courses/SOC101/Downloads/Introduction%20to%20Sociology%20-%20SOC101.pdf



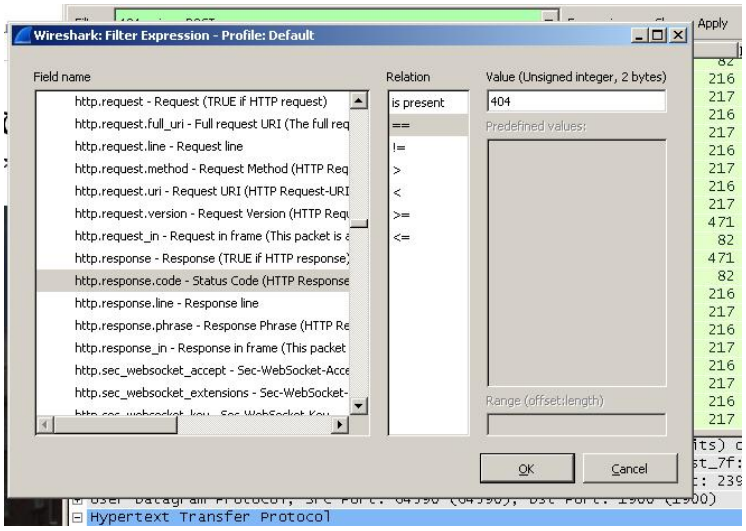
Step 6: Applying different filters using expressions.

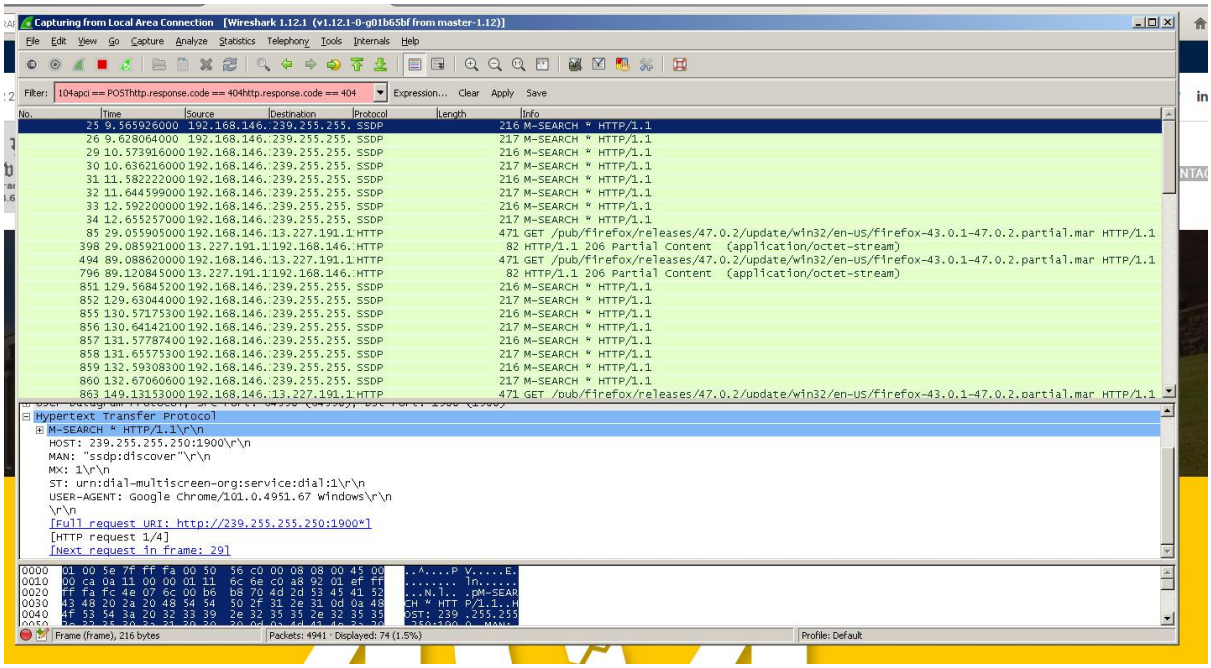
1) Filtering HTTP POST request

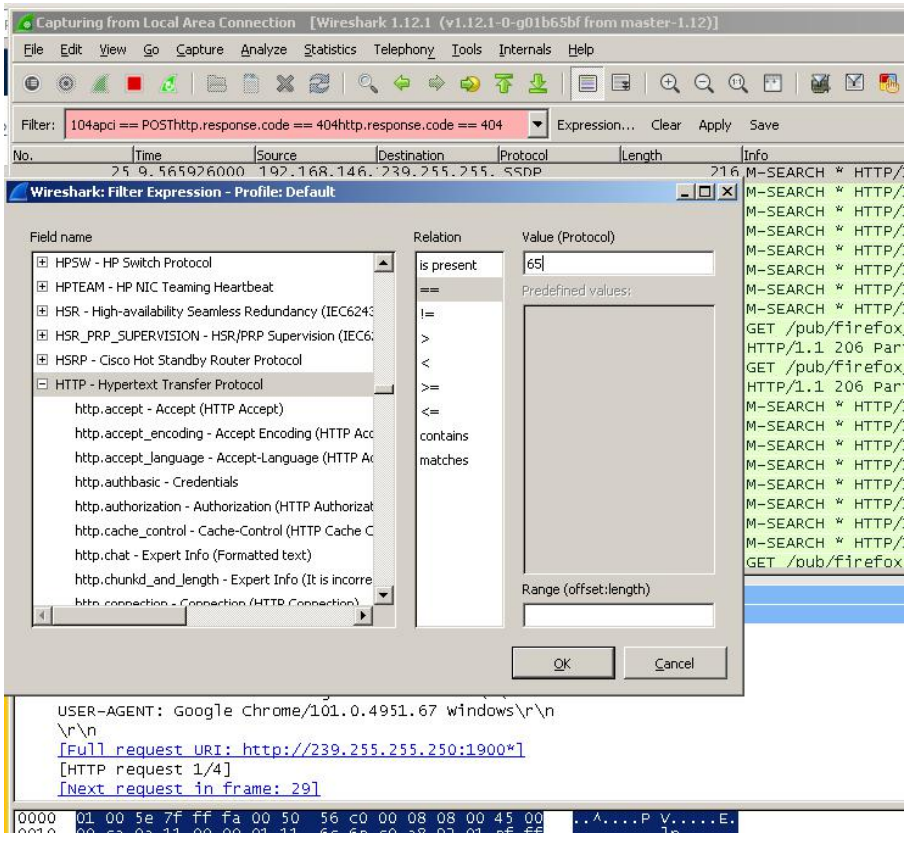


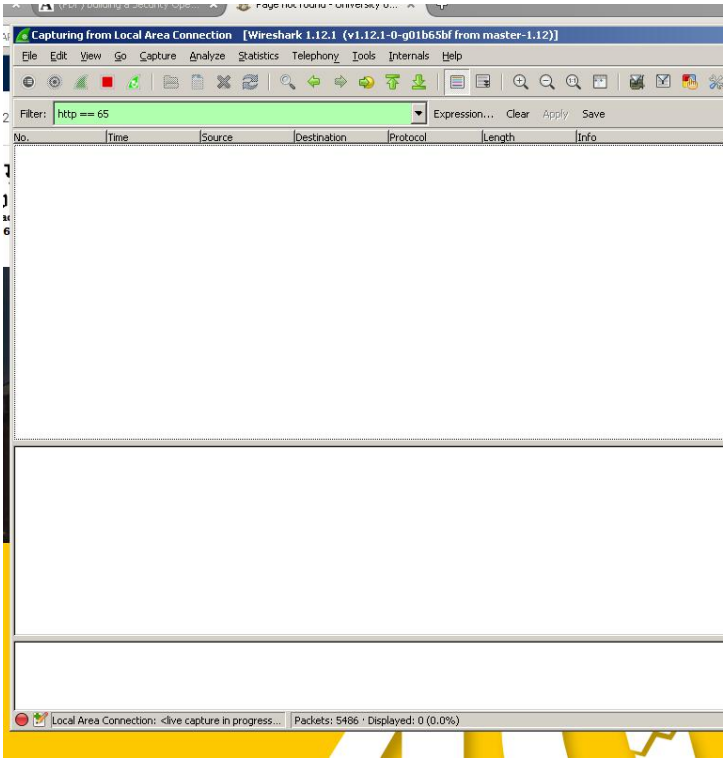
2) Filtering 404 not found error

Same under expression go to HTTP and expand that and select the status code.




Capturing from Local Area Connection [Wireshark 1.12.1 (v1.12.1-0-g01b65bf from master-1.12)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: 104apci == POSThttp.response.code == 404http.response.code == 404
Expression... Clear Apply Save
No. Time Source Destination Protocol Length Info
25 9.565926000 192.168.146.239.255.255.255. SSDP 216 M-SEARCH * HTTP/1.1
26 9.628064000 192.168.146.239.255.255.255. SSDP 217 M-SEARCH * HTTP/1.1
29 10.573916000 192.168.146.239.255.255.255. SSDP 216 M-SEARCH * HTTP/1.1
30 10.636216000 192.168.146.239.255.255.255. SSDP 217 M-SEARCH * HTTP/1.1
31 11.582222000 192.168.146.239.255.255.255. SSDP 216 M-SEARCH * HTTP/1.1
32 11.644599000 192.168.146.239.255.255.255. SSDP 217 M-SEARCH * HTTP/1.1
33 12.592200000 192.168.146.239.255.255.255. SSDP 216 M-SEARCH * HTTP/1.1
34 12.655257000 192.168.146.239.255.255.255. SSDP 217 M-SEARCH * HTTP/1.1
85 29.055905000 192.168.146.13.227.191.1 HTTP 471 GET /pub/firefox/releases/47.0.2/update/win32/en-US/firefox-43.0.1-47.0.2.partial.mar HTTP/1.1
398 29.085921000 13.227.191.1 192.168.146. HTTP 82 HTTP/1.1 206 Partial Content (application/octet-stream)
494 89.088620000 192.168.146.13.227.191.1 HTTP 471 GET /pub/firefox/releases/47.0.2/update/win32/en-US/firefox-43.0.1-47.0.2.partial.mar HTTP/1.1
796 89.120845000 13.227.191.1 192.168.146. HTTP 82 HTTP/1.1 206 Partial Content (application/octet-stream)
851 129.56845200 192.168.146.239.255.255.255. SSDP 216 M-SEARCH * HTTP/1.1
852 129.63044000 192.168.146.239.255.255.255. SSDP 217 M-SEARCH * HTTP/1.1
855 130.57175300 192.168.146.239.255.255.255. SSDP 216 M-SEARCH * HTTP/1.1
856 130.64142100 192.168.146.239.255.255.255. SSDP 217 M-SEARCH * HTTP/1.1
857 131.57787400 192.168.146.239.255.255.255. SSDP 216 M-SEARCH * HTTP/1.1
858 131.65575300 192.168.146.239.255.255.255. SSDP 217 M-SEARCH * HTTP/1.1
859 132.59308300 192.168.146.239.255.255.255. SSDP 216 M-SEARCH * HTTP/1.1
860 132.67060600 192.168.146.239.255.255.255. SSDP 217 M-SEARCH * HTTP/1.1
863 149.13153000 192.168.146.13.227.191.1 HTTP 471 GET /pub/firefox/releases/47.0.2/update/win32/en-US/firefox-43.0.1-47.0.2.partial.mar HTTP/1.1
Hypertext Transfer Protocol
M-SEARCH * HTTP/1.1\r\n
HOST: 239.255.255.250:1900\r\n
MAN: "ssdp:discover"\r\n
MX: 1\r\n
ST: urn:dial-multiscreen-org:service:dial:1\r\n
USER-AGENT: Google Chrome/101.0.4951.67 Windows\r\n
\r\n
[Full request URI: http://239.255.255.250:1900*]
[HTTP request 1/4]
[Next request in frame: 29]
0000 01 00 5e 7f ff fa 00 50 56 c0 00 08 08 00 45 00 ..^....P V.....E.
0010 00 ca 0a 11 00 00 01 11 6c 6e c0 a8 92 01 ef ff .... ln.....
0020 ff fa fc 46 07 6c 00 b6 b8 70 4d 20 53 45 41 52 ...N.l.. .pM-SEAR
0030 43 48 20 2a 20 48 54 54 50 2f 31 2e 31 0d 0a 48 CH * HTT P/1.1..H
0040 4f 53 54 3a 20 32 33 39 2e 32 35 35 2e 32 35 35 OST: 239 .255.255
Frame (frame), 216 bytes
Packets: 4941 · Displayed: 74 (1.5%)
Profile: Default


Capturing from Local Area Connection [Wireshark 1.12.1 (v1.12.1-0-g01b65bf from master-1.12)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: 104apci == POSThttp.response.code == 404http.response.code == 404
Expression... Clear Apply Save
No. Time Source Destination Protocol Length Info
Wireshark: Filter Expression - Profile: Default
Field name
Relation
Value (Protocol)
HPSW - HP Switch Protocol
HPTEAM - HP NIC Teaming Heartbeat
HSR - High-availability Seamless Redundancy (IEC624
HSR_PRP_SUPERVISION - HSR/PRP Supervision (IEC6
HSRP - Cisco Hot Standby Router Protocol
HTTP - Hypertext Transfer Protocol
http.accept - Accept (HTTP Accept)
http.accept_encoding - Accept Encoding (HTTP Acc
http.accept_language - Accept-Language (HTTP A
http.authbasic - Credentials
http.authorization - Authorization (HTTP Authorizat
http.cache_control - Cache-Control (HTTP Cache C
http.chat - Expert Info (Formatted text)
http.chunkd_and_length - Expert Info (It is incorre
is present
==
!=
>
<
>=
<=
contains
matches
65
Predefined values:
Range (offset:length)
OK
Cancel
USER-AGENT: Google Chrome/101.0.4951.67 Windows\r\n
\r\n
[Full request URI: http://239.255.255.250:1900*]
[HTTP request 1/4]
[Next request in frame: 29]
0000 01 00 5e 7f ff fa 00 50 56 c0 00 08 08 00 45 00 ..^....P V.....E.

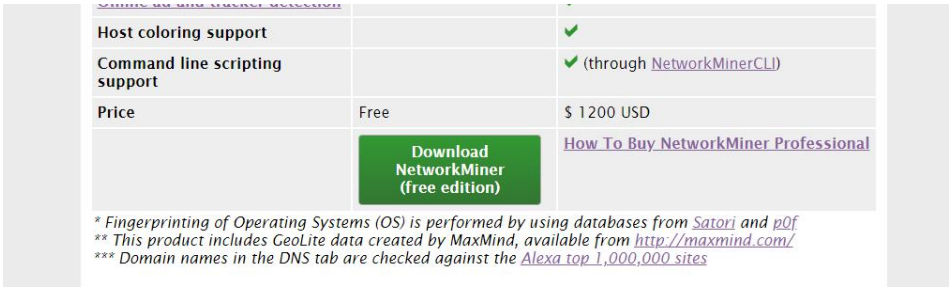

Capturing from Local Area Connection [Wireshark 1.12.1 (v1.12.1-0-g01b65bf from master-1.12)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: http == 65
Expression... Clear Apply Save
No. Time Source Destination Protocol Length Info
Local Area Connection: <live capture in progress...
Packets: 5486 · Displayed: 0 (0.0%)

Practical No. 5C

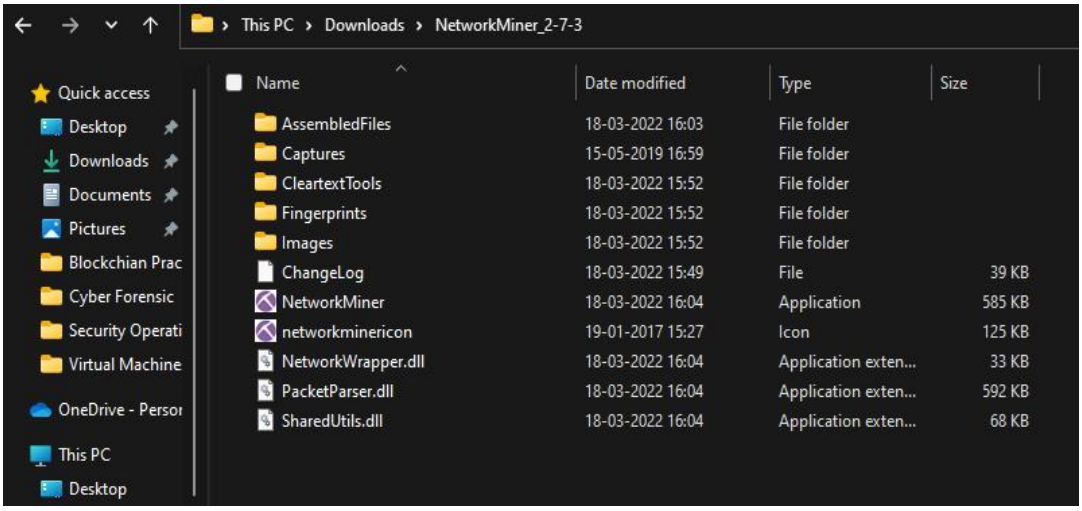
Aim: Using Network Forensic Analysis Tool NetworMiner

Part 1: Install NetworkMiner.

Step 1: download the zip file from the official web link: https://www.netresec.com/?page=NetworkMiner

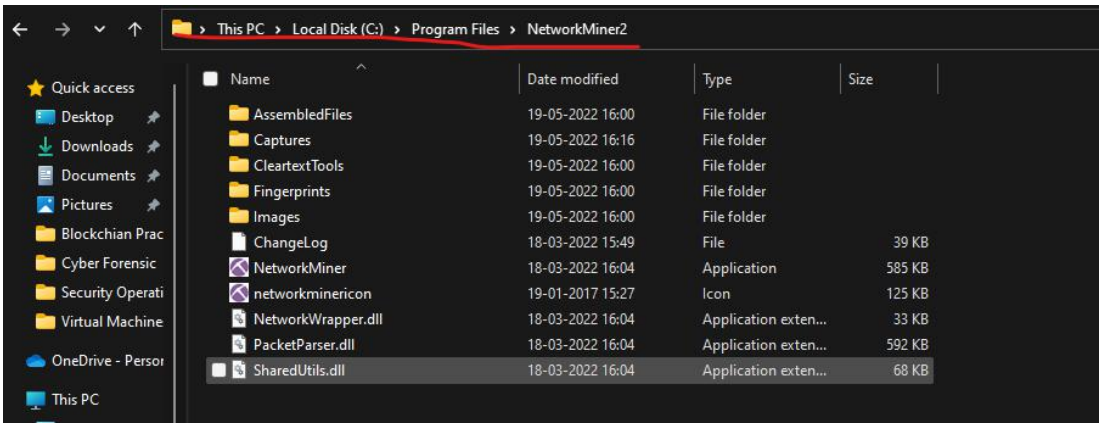


Step 2: after download we need to extract the file.subfiles look like this

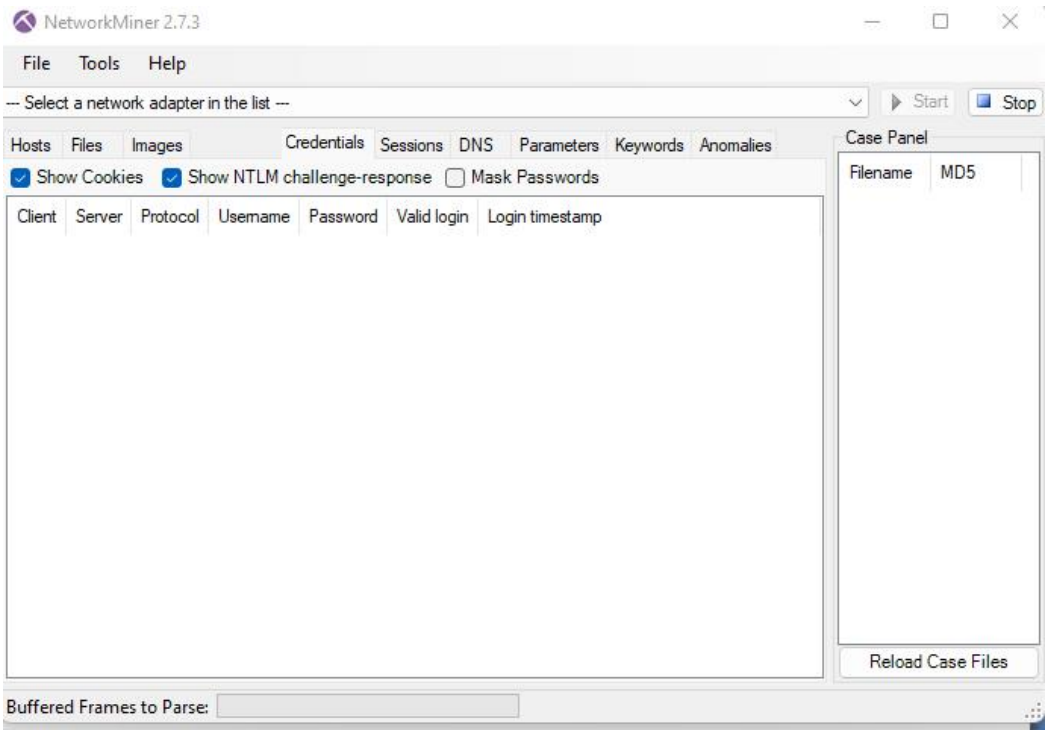


Step 3: Now we need to copy all the files and copy the move to the

C:\Program Files\NetworkMine2 (Create new folder name NetwoekMiner) and paste it that folder.

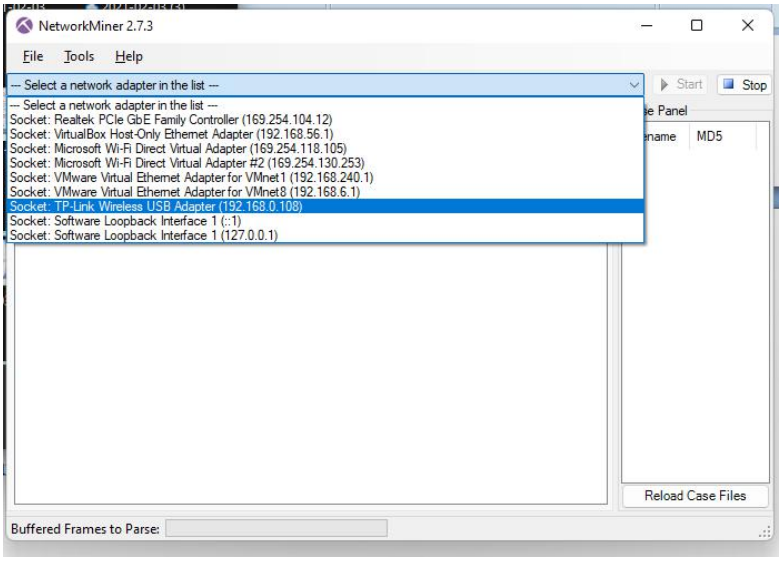


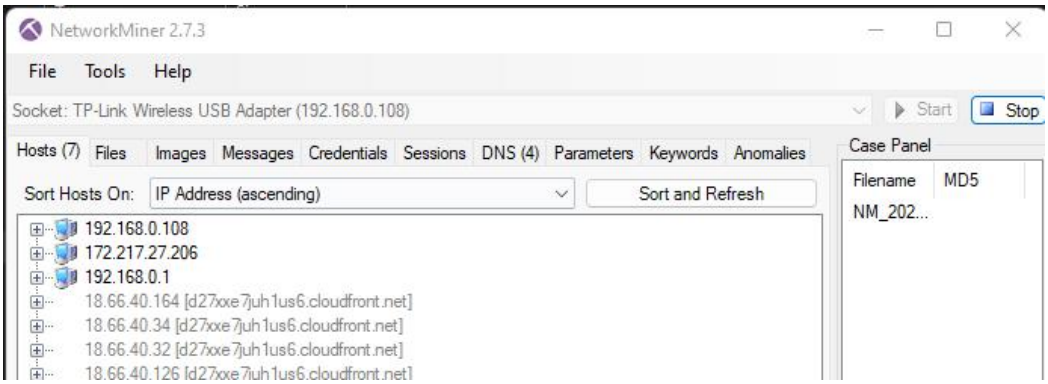
Step 4: Now open the application with open as Admin and interface look ike this.



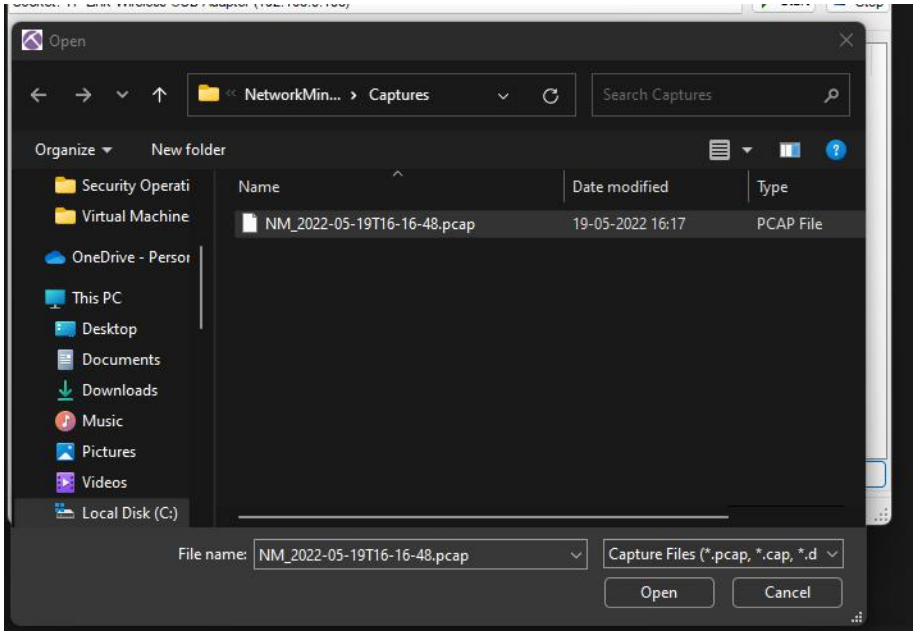
Part 2: Using the NetworkMiner for 1st use.

Step1: choose your network adapter from the top of the window.

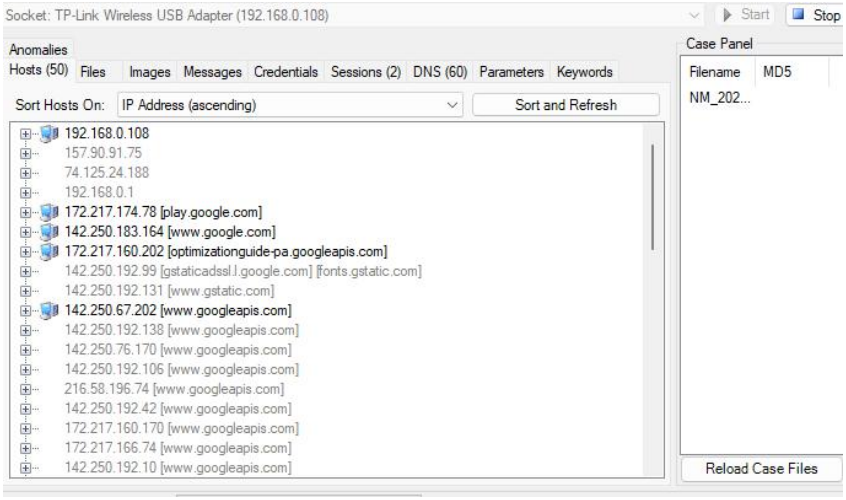




Step 2: Or if you want to open the previous packet captured so open the pcap file.



Step 3: after open the pcap file it will load all packet information which is captured.



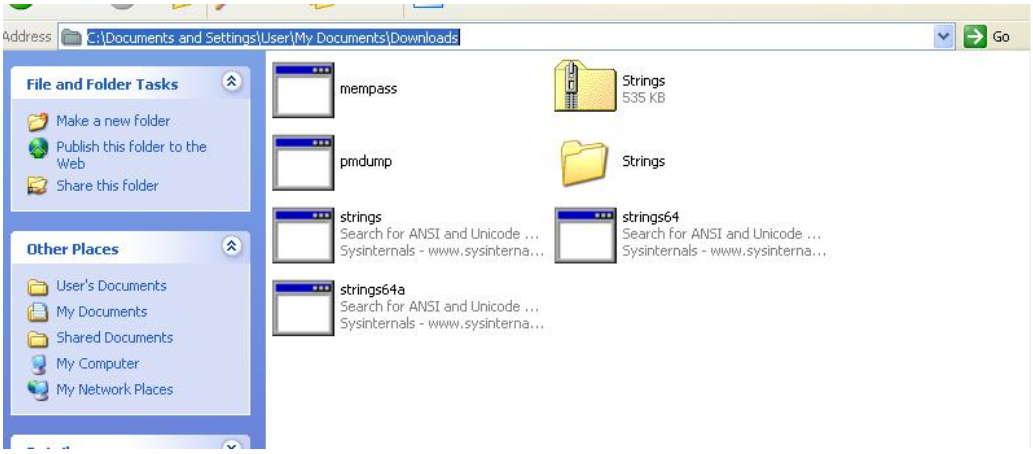
Practical No. 6

Aim: Dump Memory contents using PMdump.

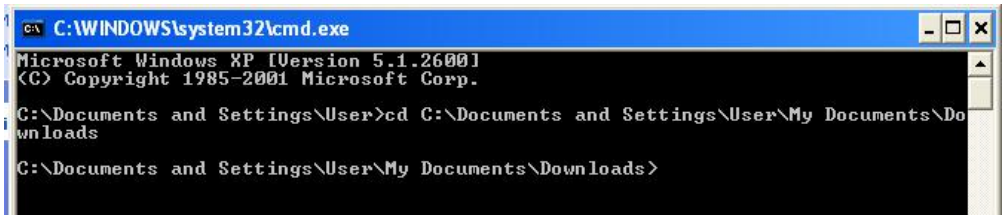
Step 1: Download all the required files. Links are given bellow:

Pmdump – https://dl.packetstormsecurity.net/Win2k/pmdump.exe
Mempass - http://code.securitytube.net/mempass.exe
Strings - https://download.sysinternals.com/files/Strings.zip

Step 2: Save all the files in same folder.

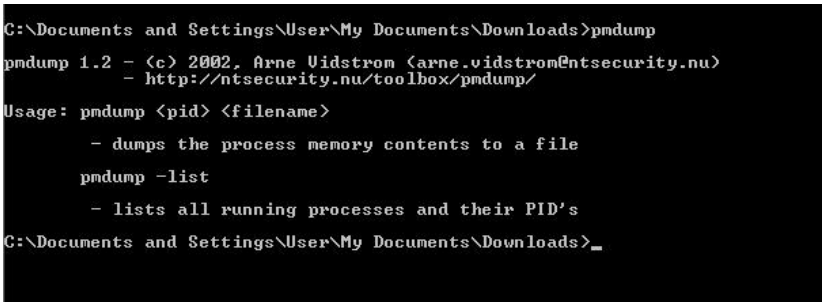


Step 3: open cmd and go to the folder were you have downloaded all the files.



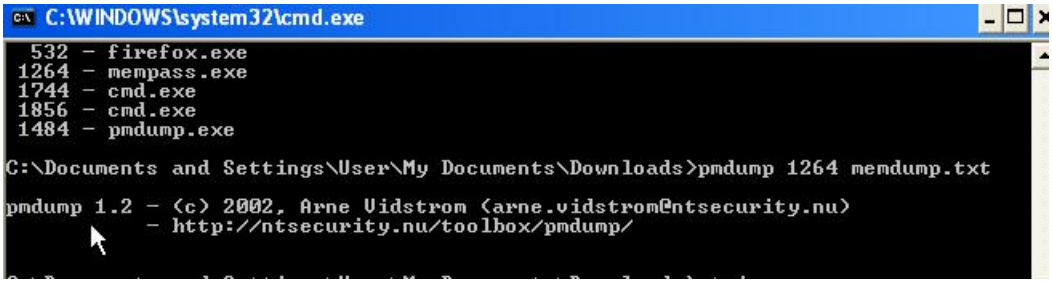
Step 4: run the command “mempass.exe” and create password

Step 5: then type pmdump

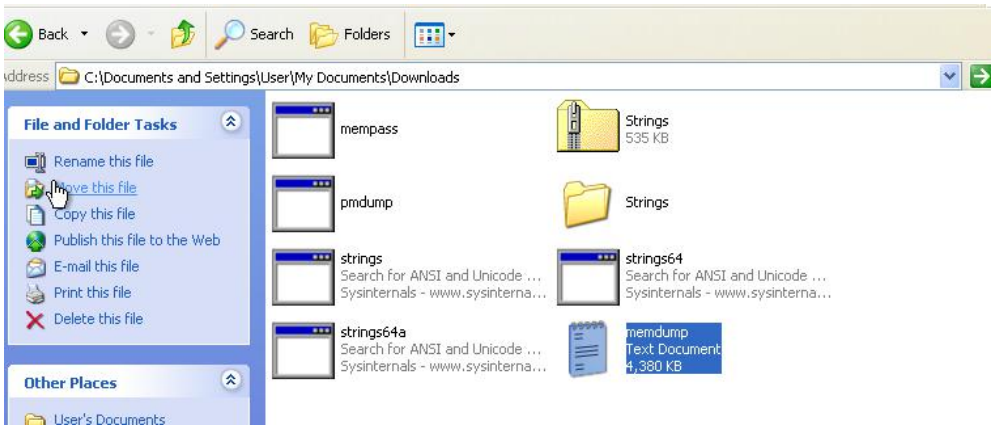


Step 6: type “pmdump –list” and chech for the process id for mempass.exe

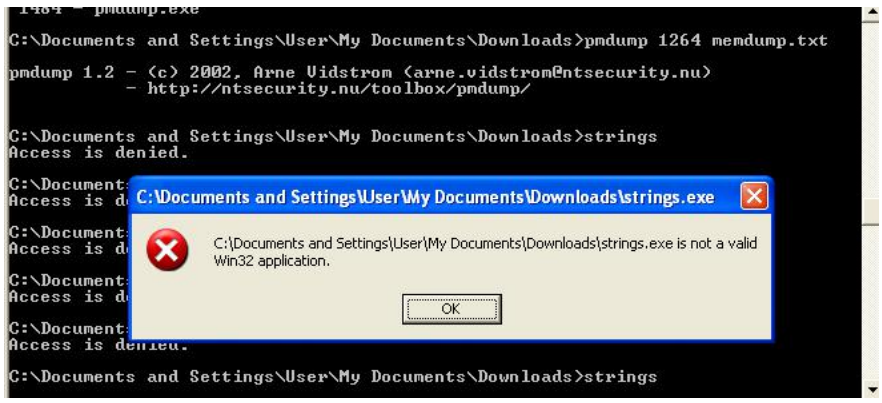
Step 7: type “pmdump 1264 memdump.txt” it will save the log into text file.



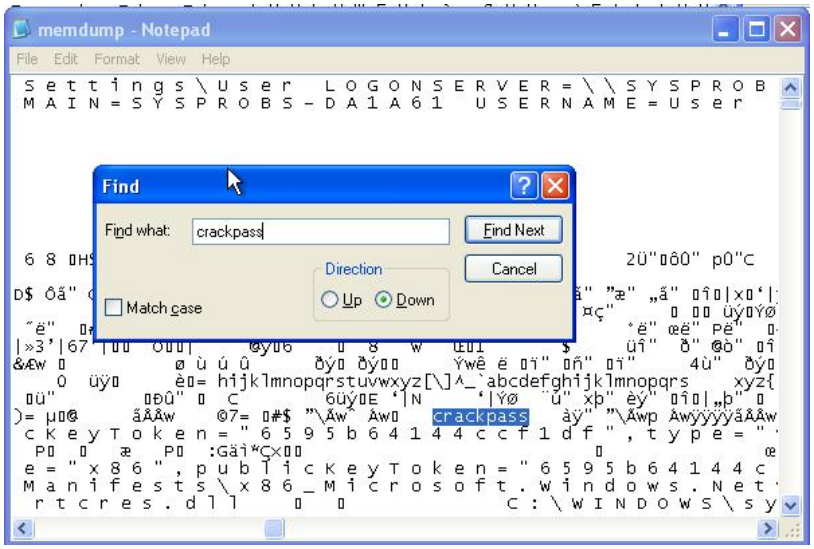
Step 8: txt file is saved in local drive.



Step 9: now we run the strings command but here we are getting some error. It is not possible to go with this process.



Step 10: So simply we open the txt file and search for the password we createdat the time of mempass.exe file running in cmd.

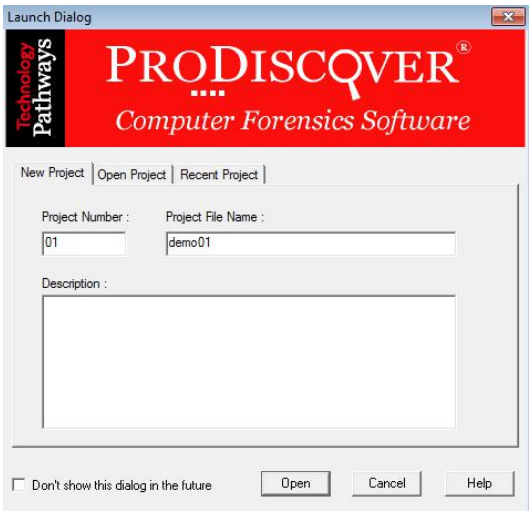


Practical No. 7

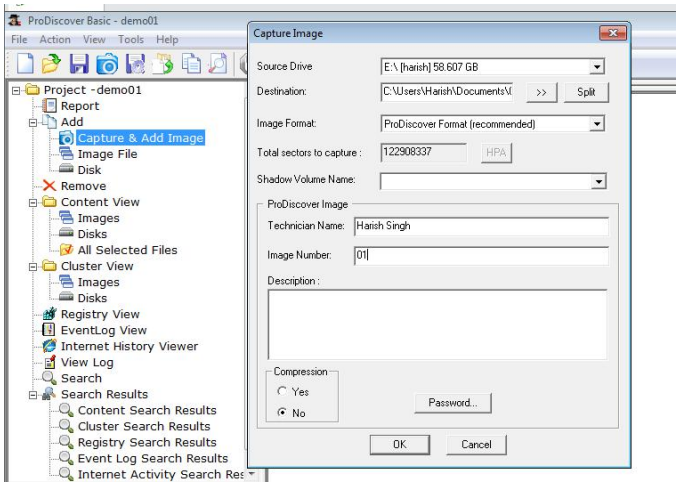
Aim: Using Data Acquisition Tools [ProDiscover Pro]

Solution:

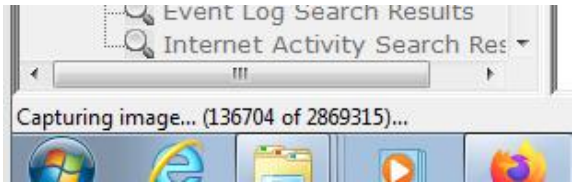
Step 1: open the ProDiscover and add the case number and name.

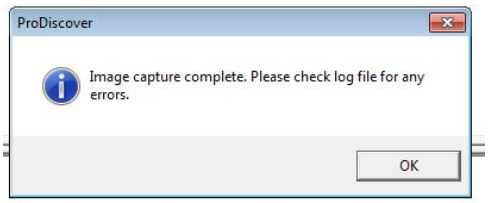


Step 2:open the project and click on Add then select the capture&Add image.fil all the detail as shown in the snap.

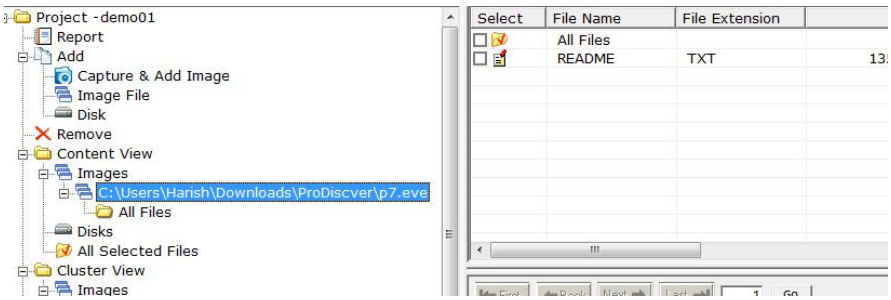


Step 3: After clicking ok image should be created.

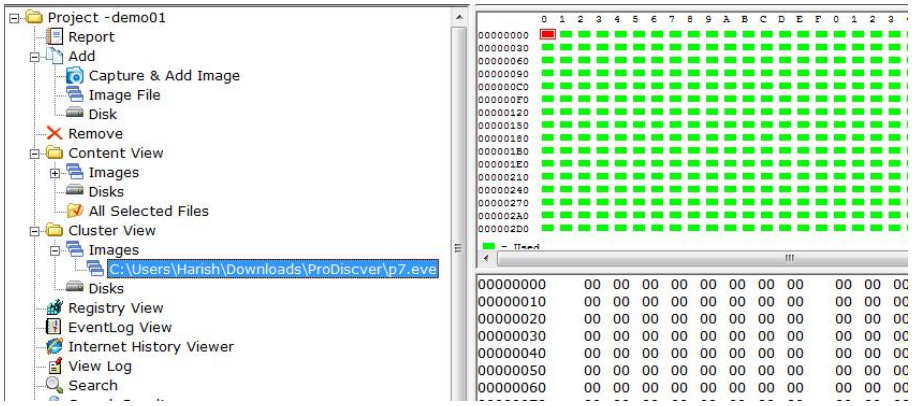




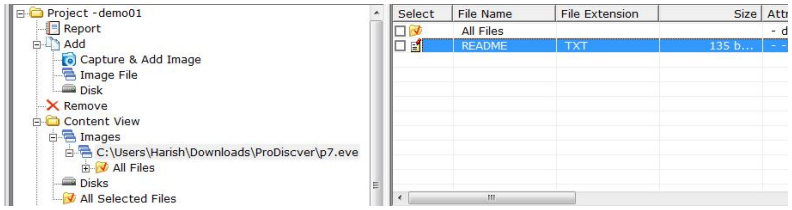
Step 4: After complete of image creation. please check the log file for any error. For that go to content view and click image.



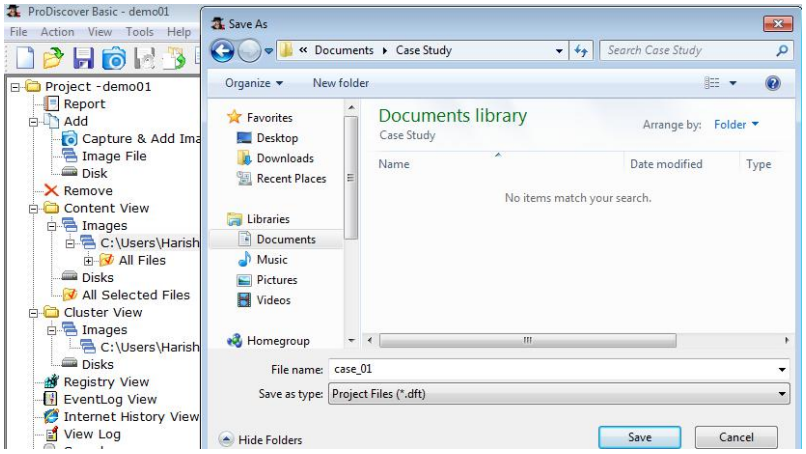
Step 5: you can also see the cluster view file has been created.



Step 6: we can view the file of PanDrive as we created he image.



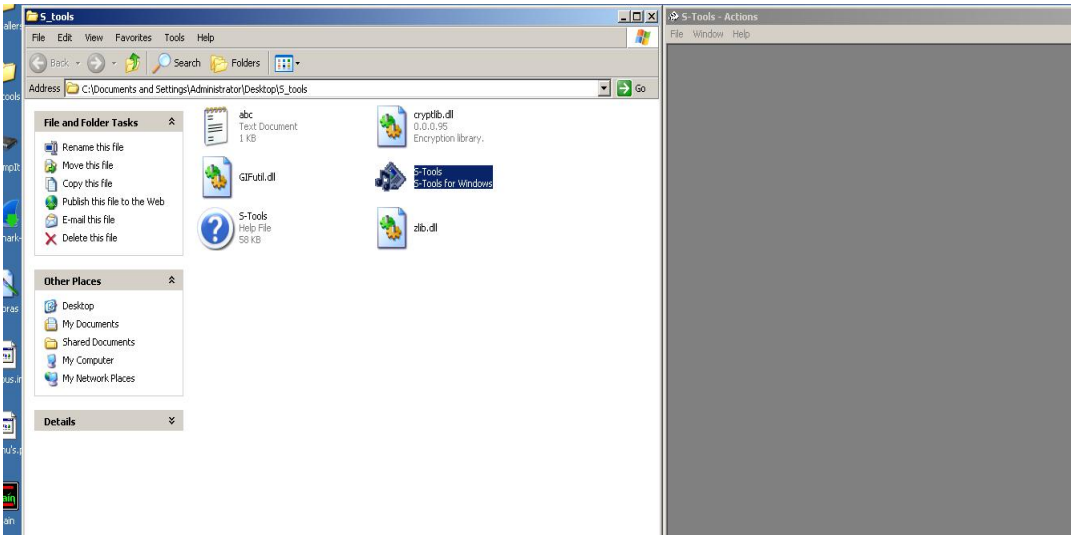
Step 7: At last we need to save the CaseFile.



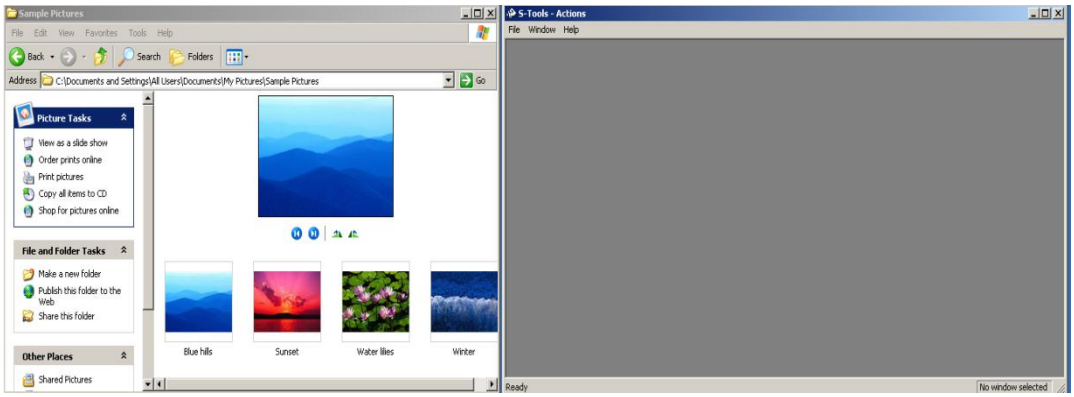
Practical No. 8 A

Following steps Show how to use freeware S-Tools utility to hide and reveal files inside pictures

Step 1) Select the S-Tools.exe file and open the steganography software tool.



Step 2) With both the working directory and the S-Tools program open minimize both windows and place side-by-side



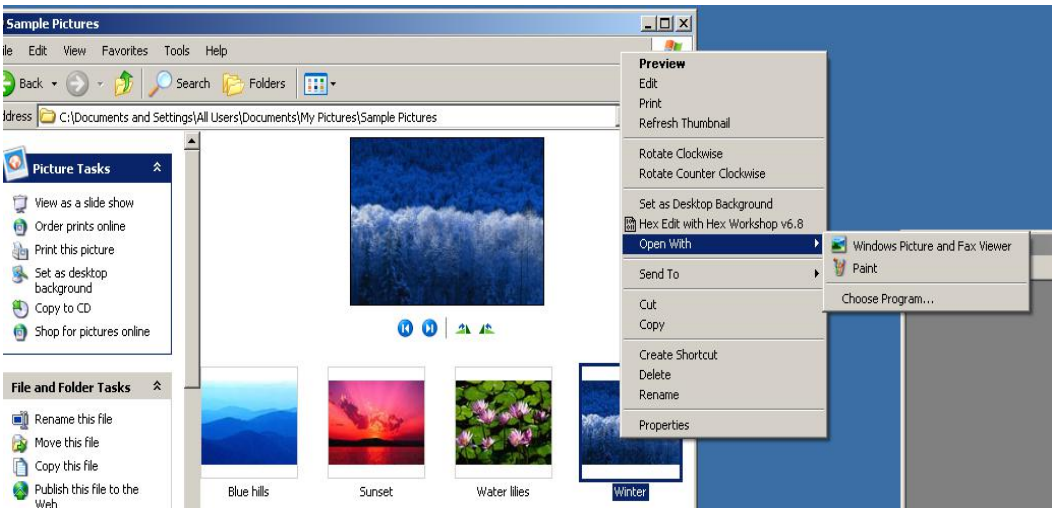
The S-Tools program is a drag and drop software. The files used to create the steganography file can be dragged from the directory into the S-Tools program.

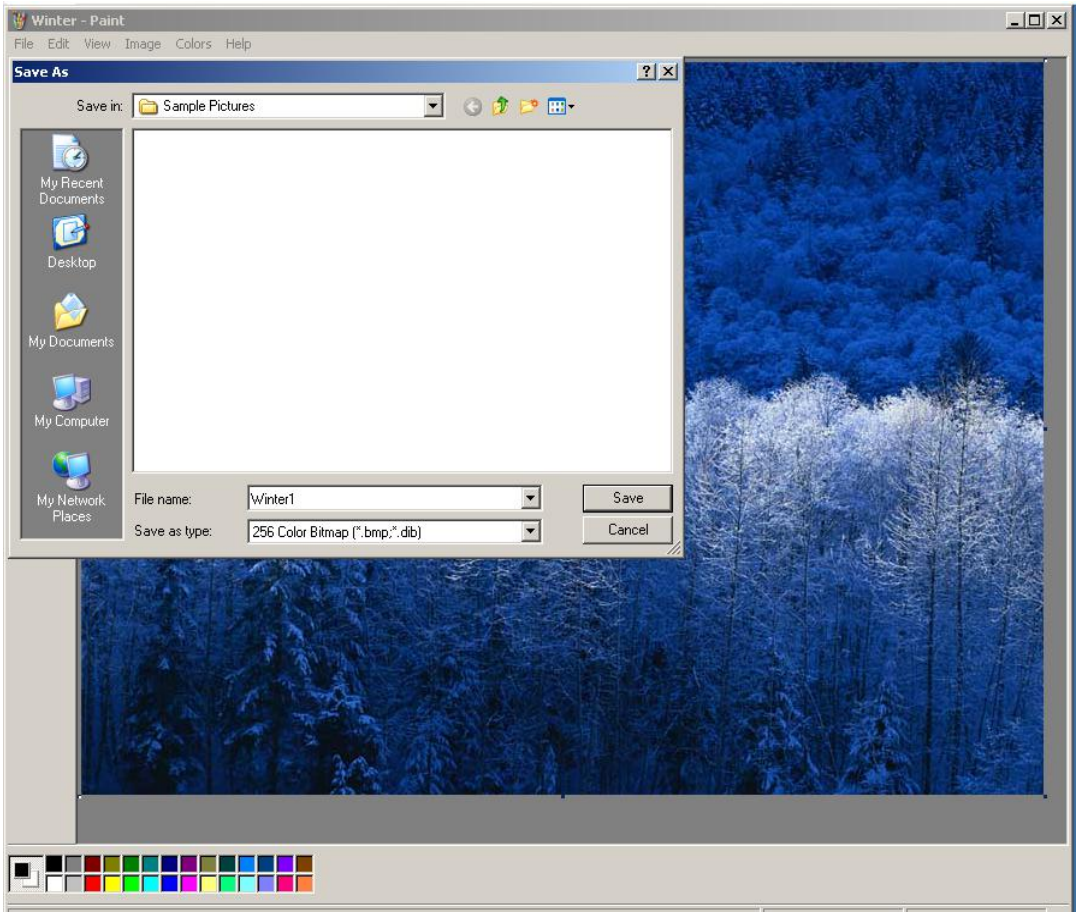
Step 3) Select the file from the directory and drag it over the S-Tools main window and release the file.

A dialogue box appears indicating that the file type is unknown. Supported file types for audio and image files are shown below:

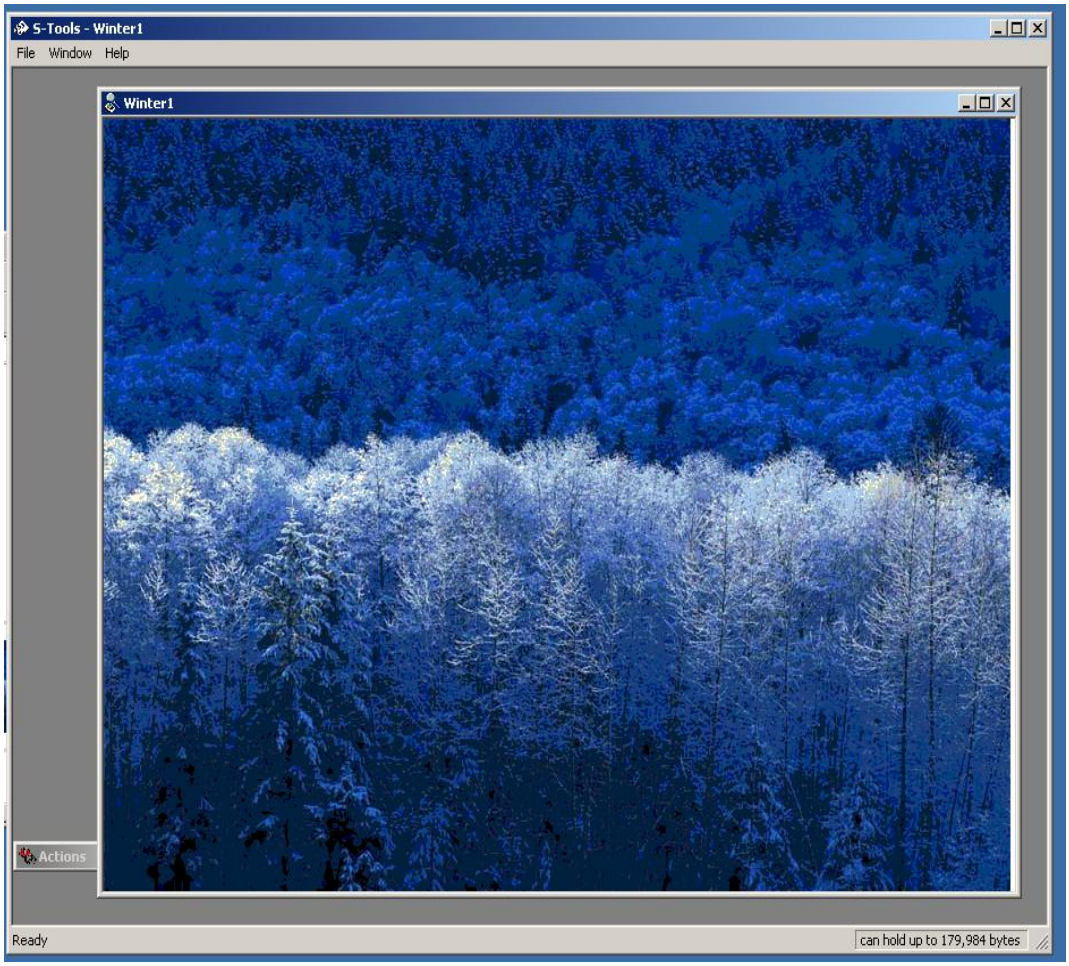
- Audio - *.wav
- Image - *.bmp and *.gif

If your image is in .jpg format, convert it to .bmp format by doing the following steps using Paint:

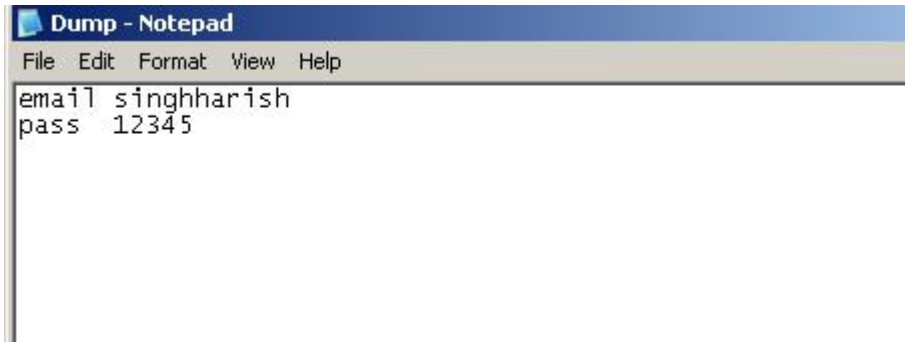




Step 4) Select a valid audio file or image as the base file for the steganography file. The Winter1.bmp was selected and dragged onto the main window of the S-Tools program. The image is opened.

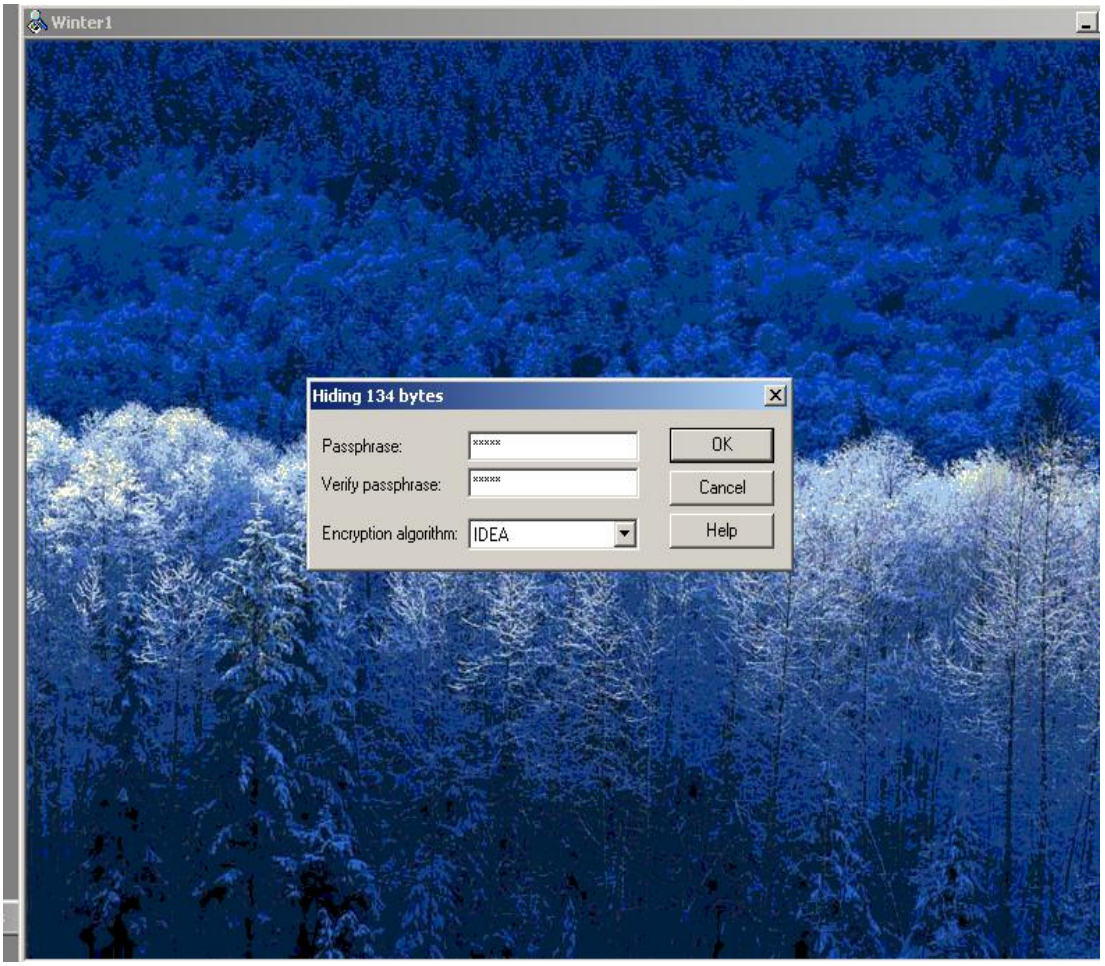


Step 5) Select a file to hide within the base file. If it's not there, create a txt file and Save the file.



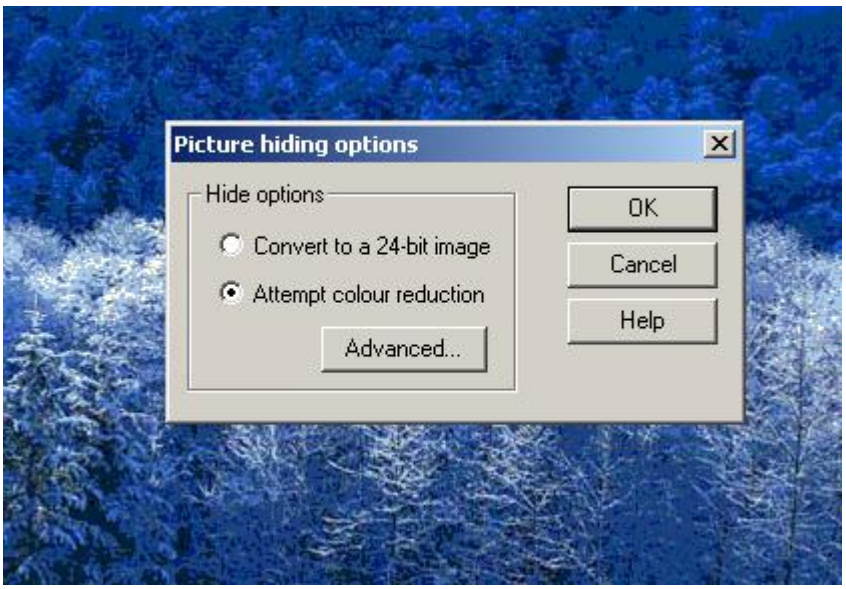
Step 6) The *.txt text file is selected and dragged on top of the base image. Release the file while the cursor is still on top of the base file.

Step 7) A dialogue box will appear asking the user to enter and verify a passphrase. Additionally, the user will have to select an encryption algorithm.

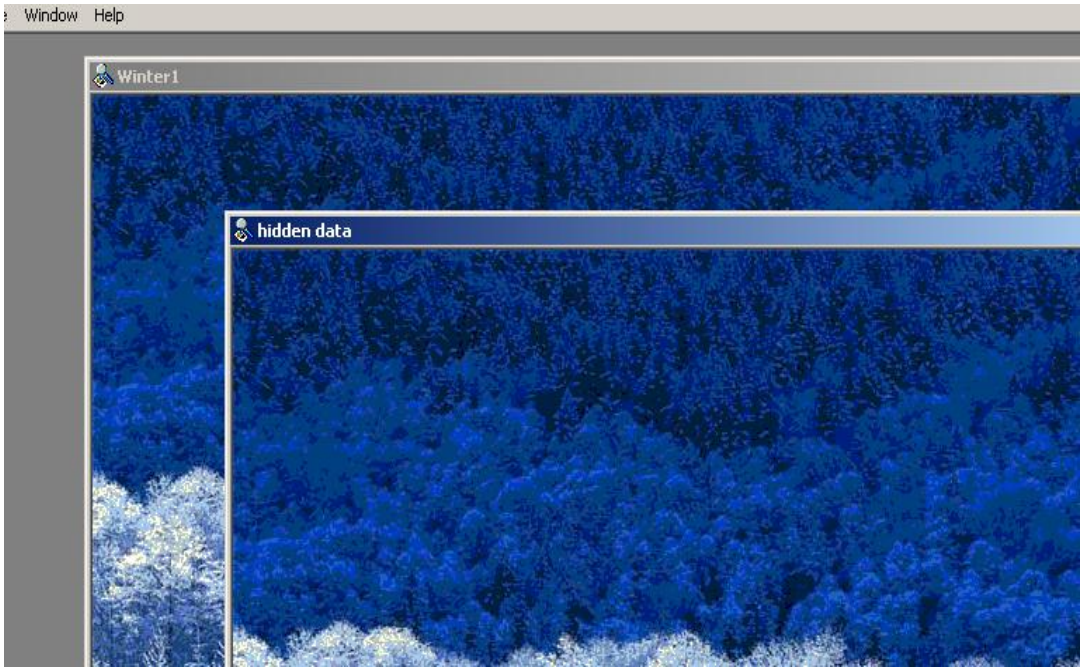


Step 8) Enter a passphrase in both the passphrase and verify passphrase text boxes. If the same passphrase is not entered in both text boxes the 'OK' button will be grayed out and the user will not be able to proceed to creating the steganography file.

Step 9) Select the 'OK' button after entering a valid passphrase.

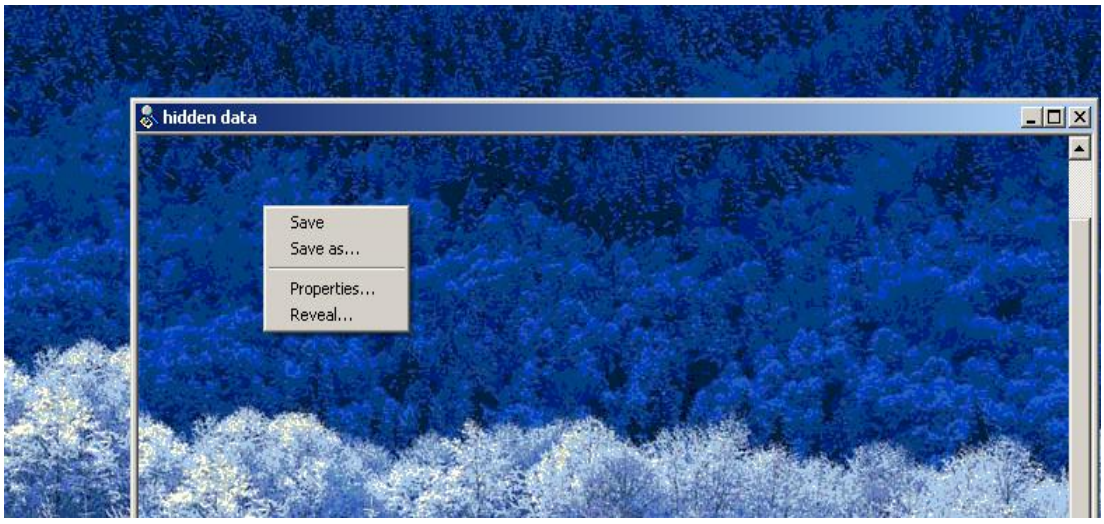


Step 10) The S-Tools main window will appear and a new file will be visible. The name of the file will be called hidden_data by default.



Step 11) Place the cursor on top of the hidden data image and select the right mouse button. The user will
have four options available to them:

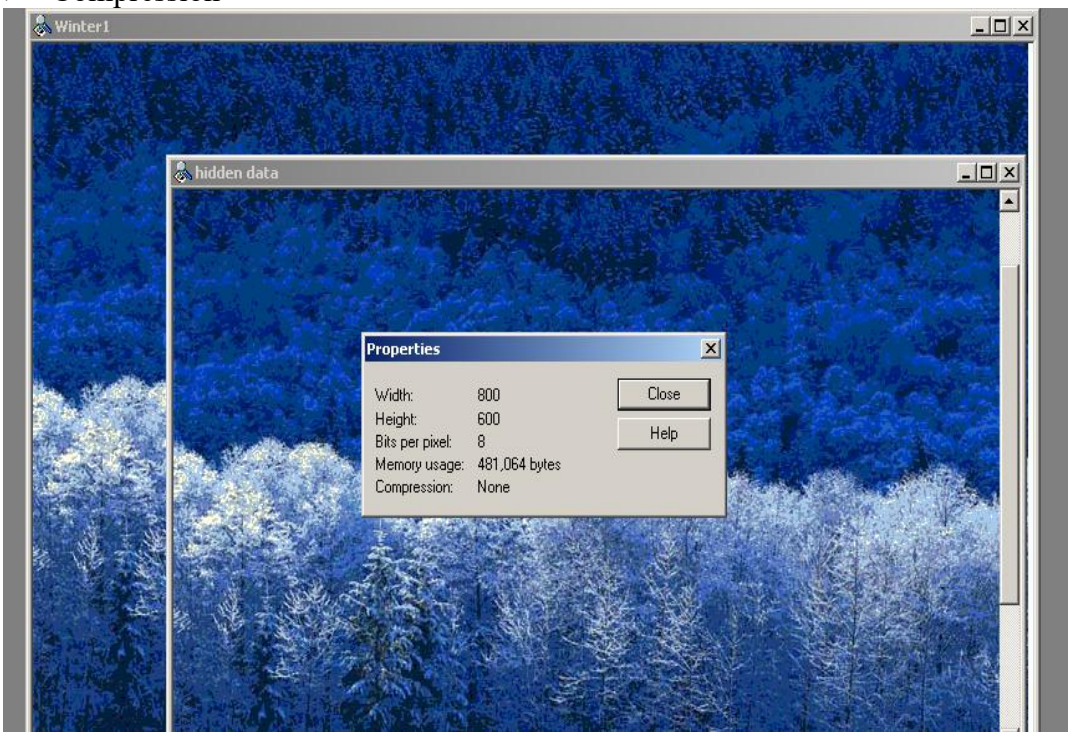
- Save
- Save As
- Properties
- Reveal



Step 12) Selecting the 'Properties' button while the cursor is over any image will display the following properties:

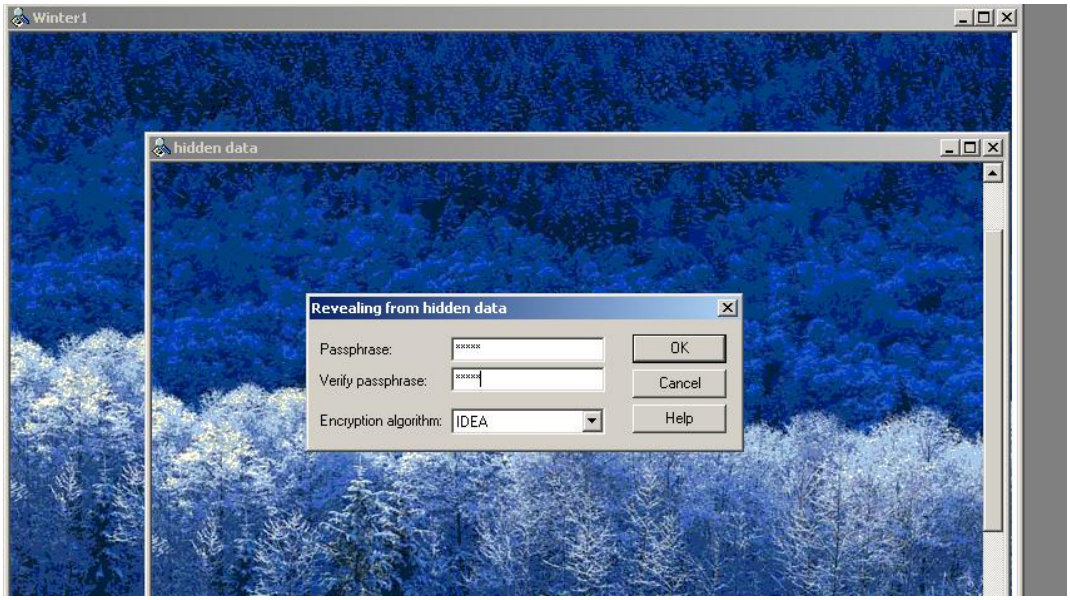
- Width and Height of the image

- Bits per pixel
- Memory Usage (file size in bytes)
- Compression

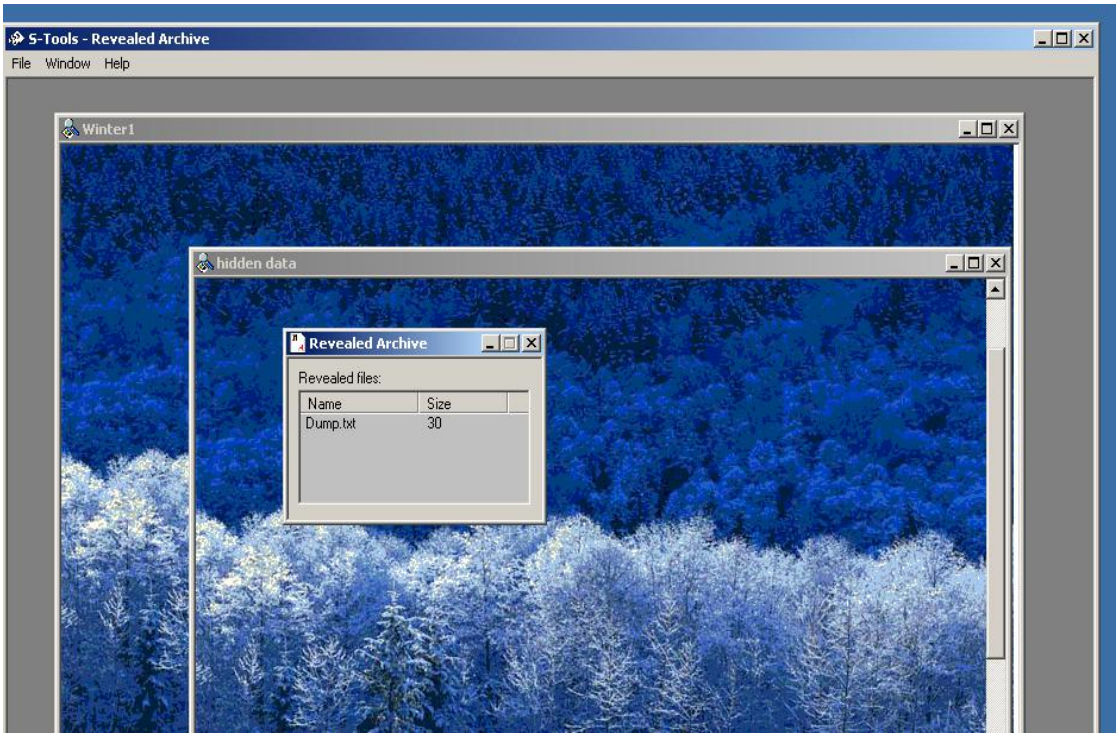


Step 13) Selecting the 'Reveal' button will display a passphrase dialogue box. A passphrase must be entered twice in the dialogue box and the correct encryption algorithm must be selected.
Notice that the title of the dialogue box has changed to 'Revealing from Tulips.bmp'

Step 14) Enter a passphrase twice, select the encryption algorithm, and select the 'OK' button.



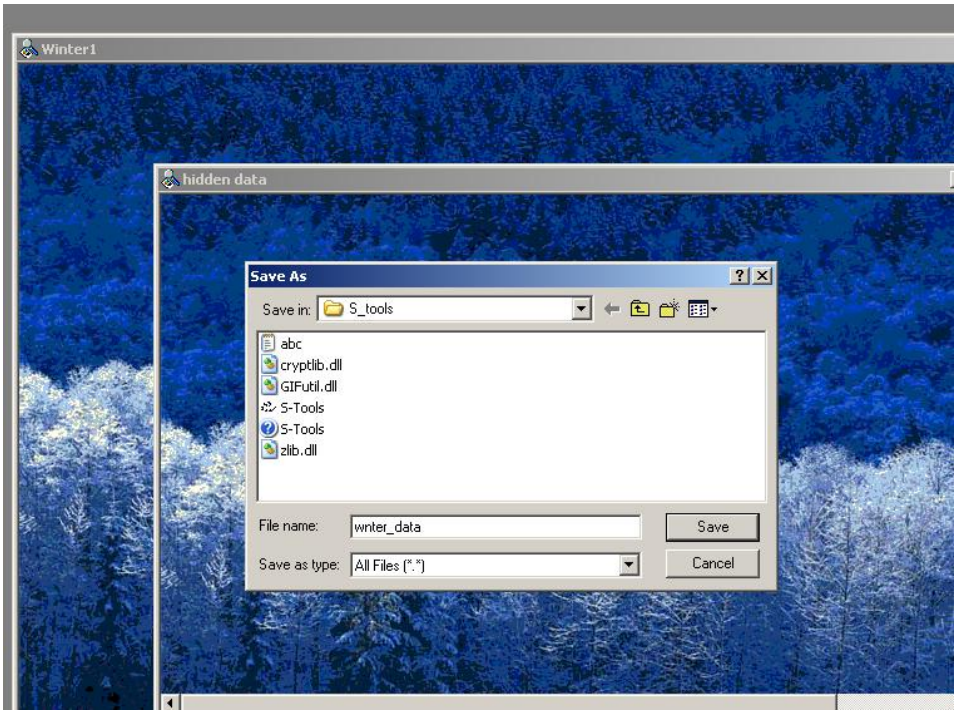
Step 15) A ‘Revealed Archive’ dialogue box will display which contains the file name and size of the hidden file.



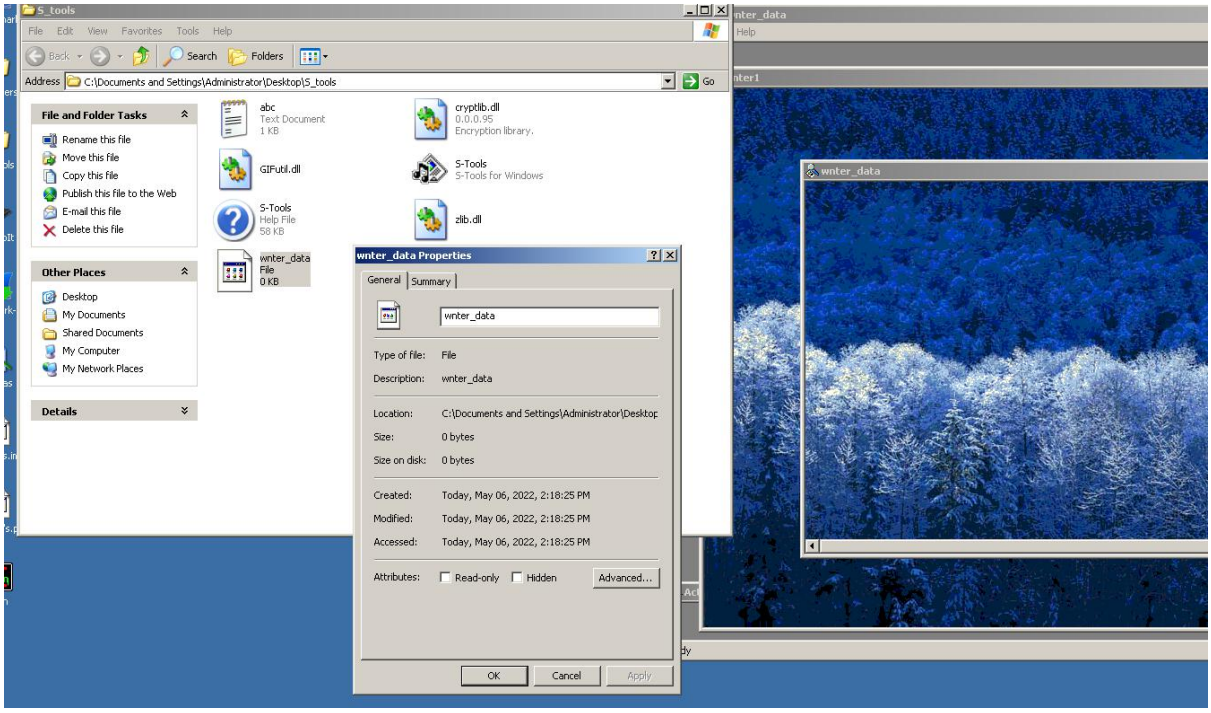
Step 16) Select the ‘Save As’ button.

Step 17) A ‘Save As’ dialogue box will appear. Enter a valid file name, select the working directory and select the ‘Save’ button.

Step 18) Locate the files in the working directory.



Step 19) Open the files using a multimedia software program and ensure that the files were extracted from the steganography file successfully.

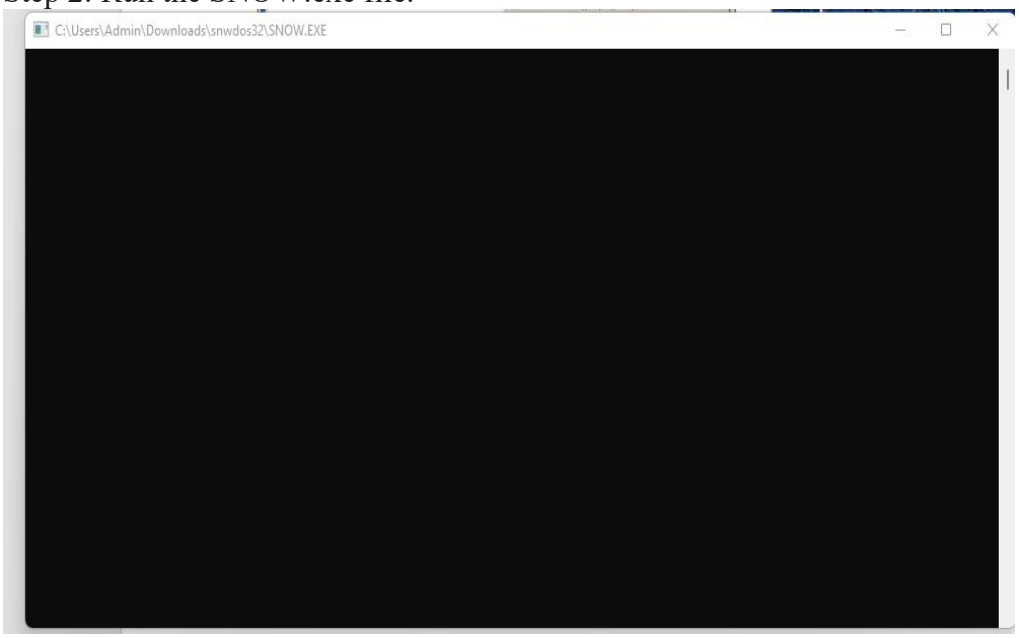


Practical No. 8B

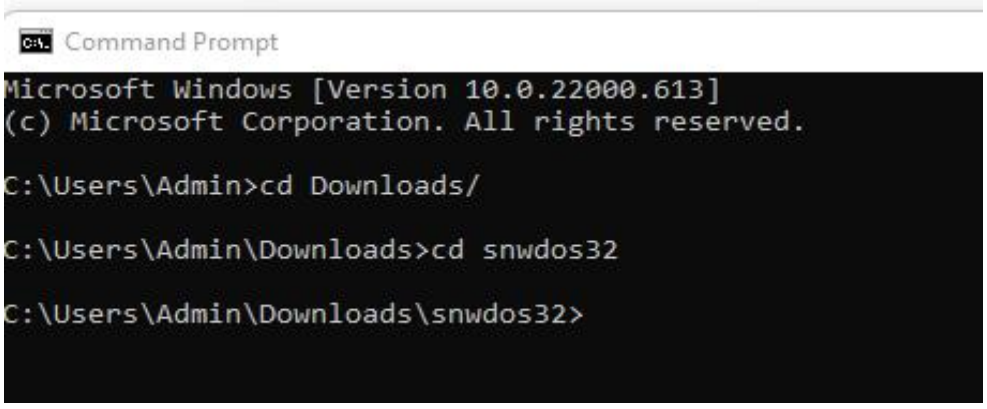
Aim: Stegnography tool SNOW.exe

SNOW is a tool that is the abbreviation of Steganographic Nature Of Whitespace which uses ICE encryption algorithm. SNOW is a whitespace steganography tool that is used to embed hidden messages in ASCII format by extending the whitespace to the end of lines.

Step 1: Open the uncompressed file.
Step 2: Run the SNOW.exe file.



Step 3: Open CMD and reach the file that you want to hide the message within.



Step 4: Put this command on cmd
“SNOW.EXE -C -p 1234 -m "hidden message" input.txt output.txt”

```
Microsoft Windows [Version 10.0.22000.613]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin>cd Downloads/

C:\Users\Admin\Downloads>cd snwdos32

C:\Users\Admin\Downloads\snwdos32>SNOW.EXE -C -p 1234 -m "hidden message" input.txt output.txt
Compressed by 44.64%
Message exceeded available space by approximately 1.#J%.
An extra 3 lines were added.

C:\Users\Admin\Downloads\snwdos32>
```

Step 5: to extract the hidden message type the following command
"SNOW.EXE -C -p 1234 output.txt"

```
C:\Users\Admin\Downloads\snwdos32>SNOW.EXE -C -p 1234 -m "hidden message" input.txt output.txt
Compressed by 44.64%
Message exceeded available space by approximately 1.#J%.
An extra 3 lines were added.

C:\Users\Admin\Downloads\snwdos32>SNOW.EXE -C -p 1234 output.txt
hidden message
C:\Users\Admin\Downloads\snwdos32>
```

Practical No. 9A

Aim: Password Cracking using Cain and Abel.

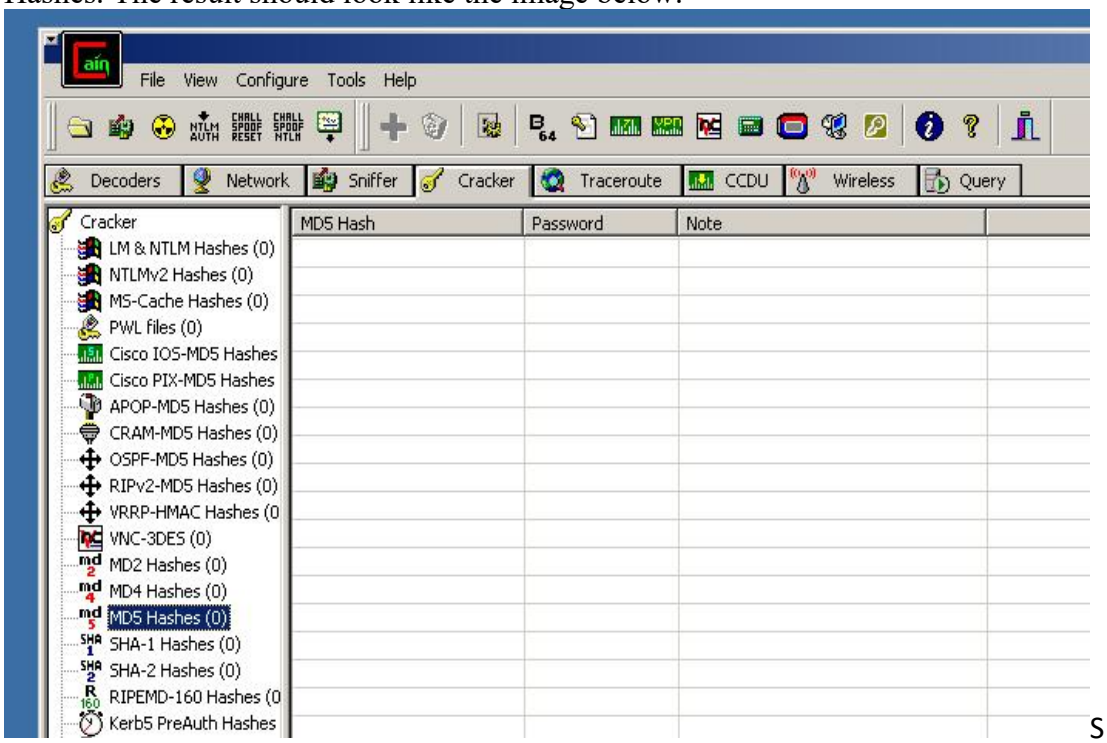
This exercise demonstrates how password could be cracked through various methods, specifically regarding MD5 encrypted passwords

Part 1: Dictionary attack - Dictionary attack uses a predetermined list of words from a dictionary to generate possible passwords that may match the MD5 encrypted password. This is one of the easiest and quickest way to obtain any given password.

Step 1: Start Cain & Abel via the Desktop Shortcut 'Cain' or Start menu . a. (Start > Programs > Cain > Cain).

Step 2: Choose 'Yes' to proceed when a 'User Account Control' notification pops up regarding software authorization

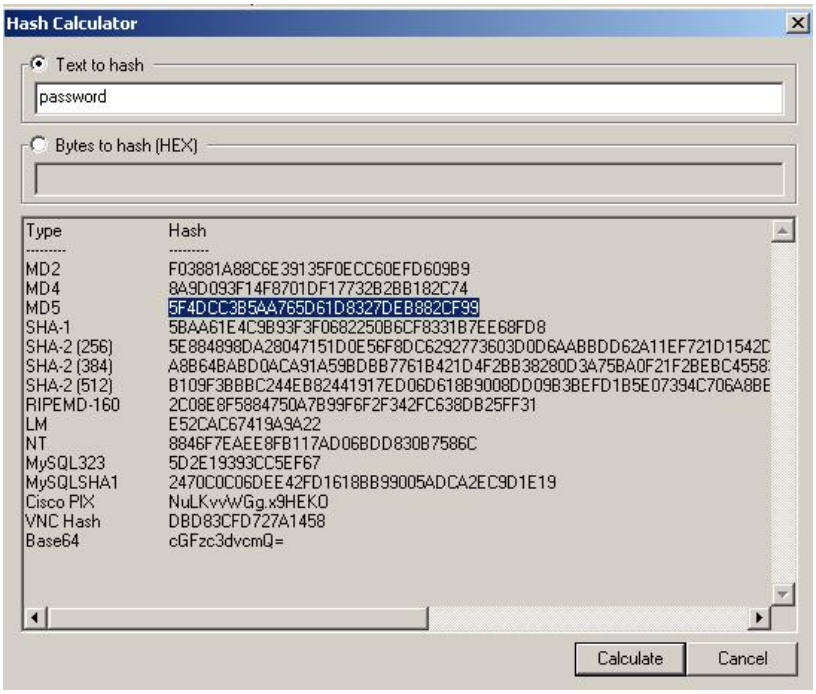
Step 3: Once on, select the 'Cracker' tab with the key symbol, then click on MD5 Hashes. The result should look like the image below.



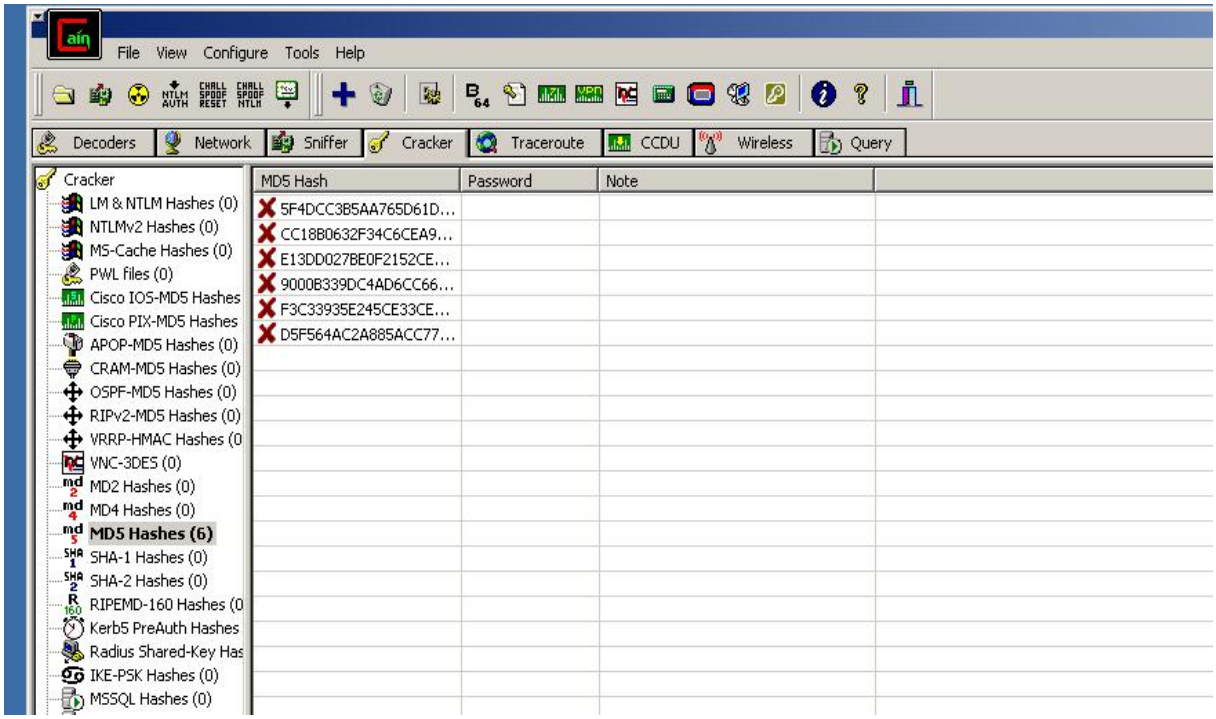
S

tep 4: As you might have noticed we don't have any passwords to crack, thus for the next few steps we will create our own MD5 encrypted passwords. First, locate the Hash Calculator among a row of icons near the top. Open it.

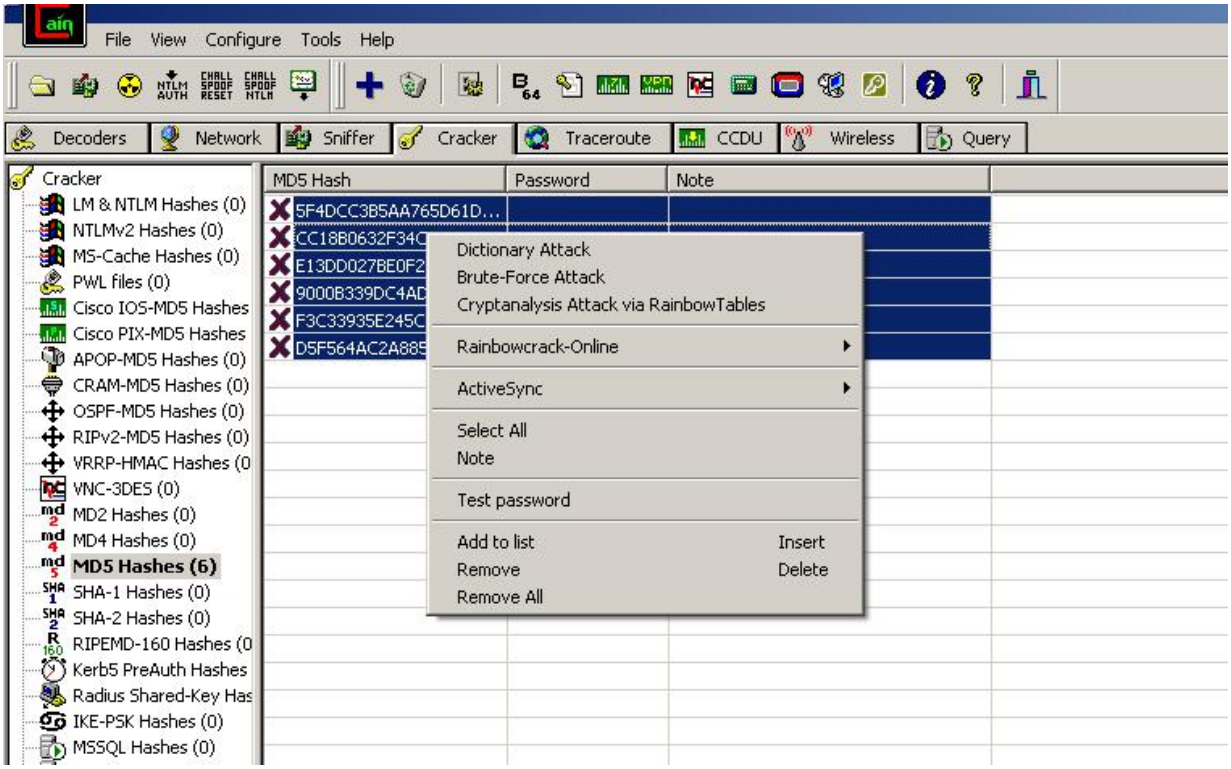
Step 5: Next, type into 'Text to Hash' the word password. It will generate a list of hashes pertaining to different types of hash algorithms. We will be focusing on MD5 hash so copy it. Then exit calculator by clicking 'Cancel'



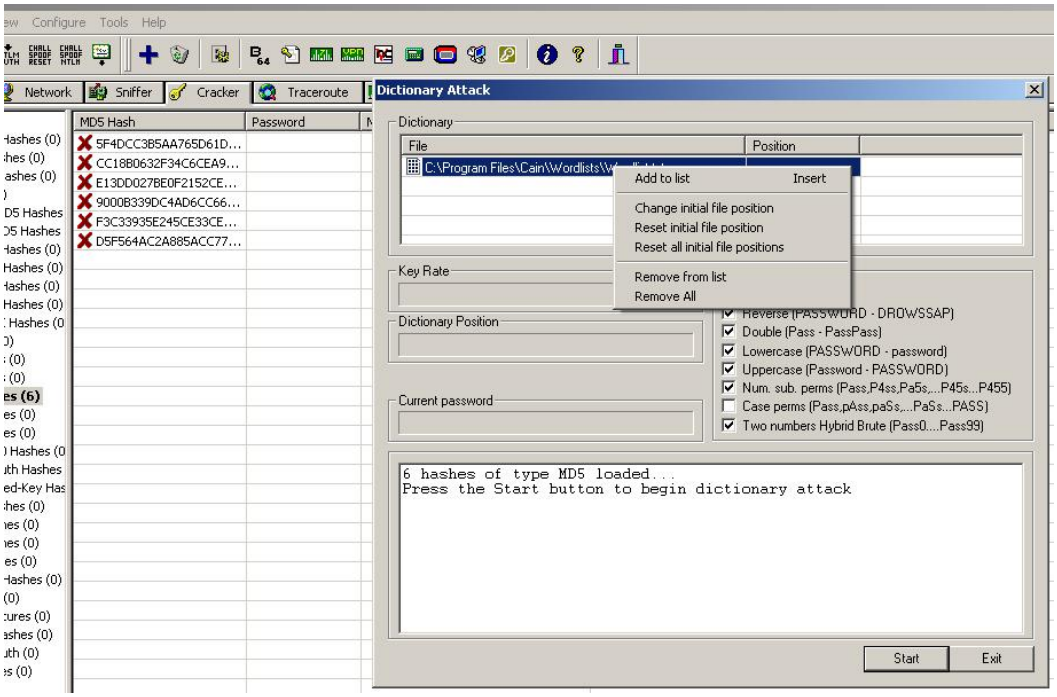
Step 6: After you exit, right click and select 'Add to list' , paste your hash then click OK . Your first encrypted password! But don't stop there, add the following MD5 hashes from the words PaSS , 13579, 15473 , sunshine89, and c@t69



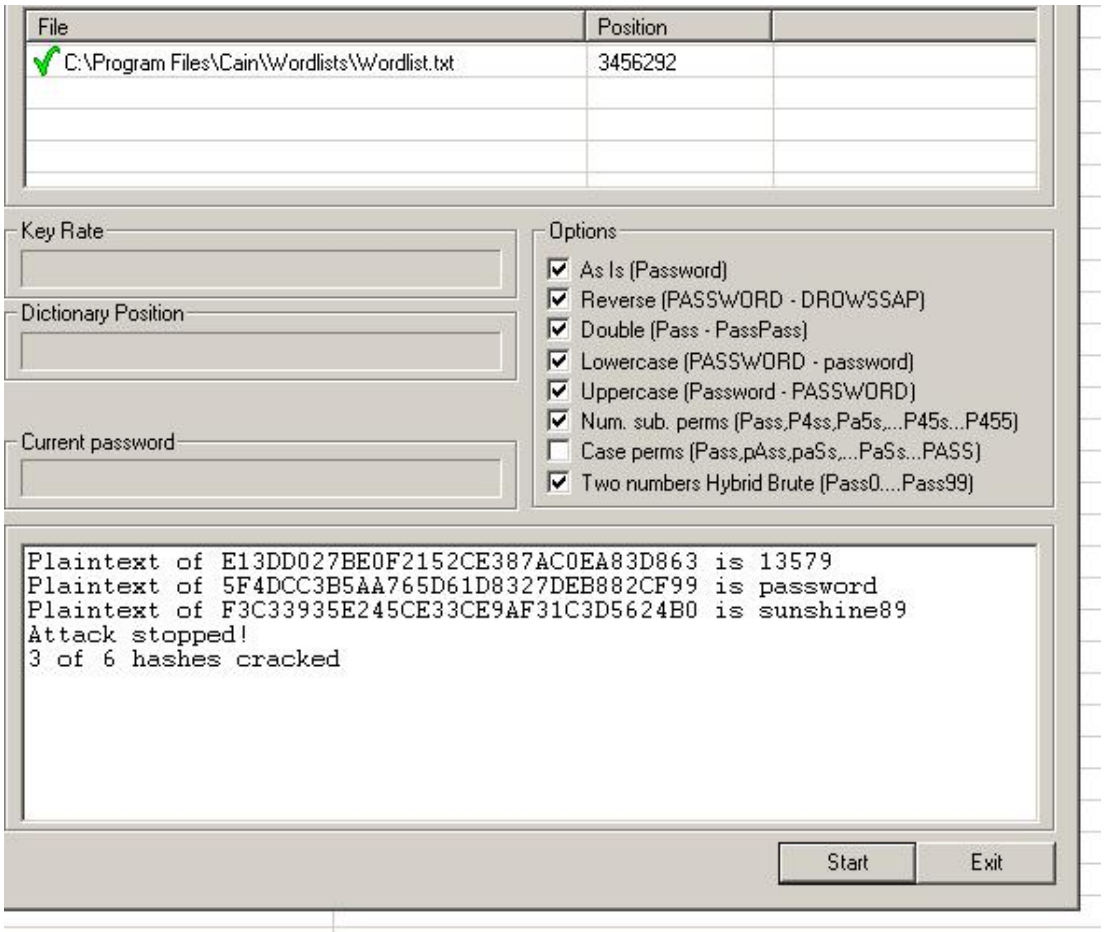
Step 7: With all the encrypted MD5 passwords on hand, we can finally start! Move your cursor and select all six passwords, then right click and press 'Dictionary Attack'.



Step 8: Once the window opens, go up to the dictionary and select 'Wordlist.txt' , right click and select 'Reset initial file position'. You'll know you've resetted when there's nothing under the position column. Note: Make sure to do this every time you want to restart a dictionary attack!

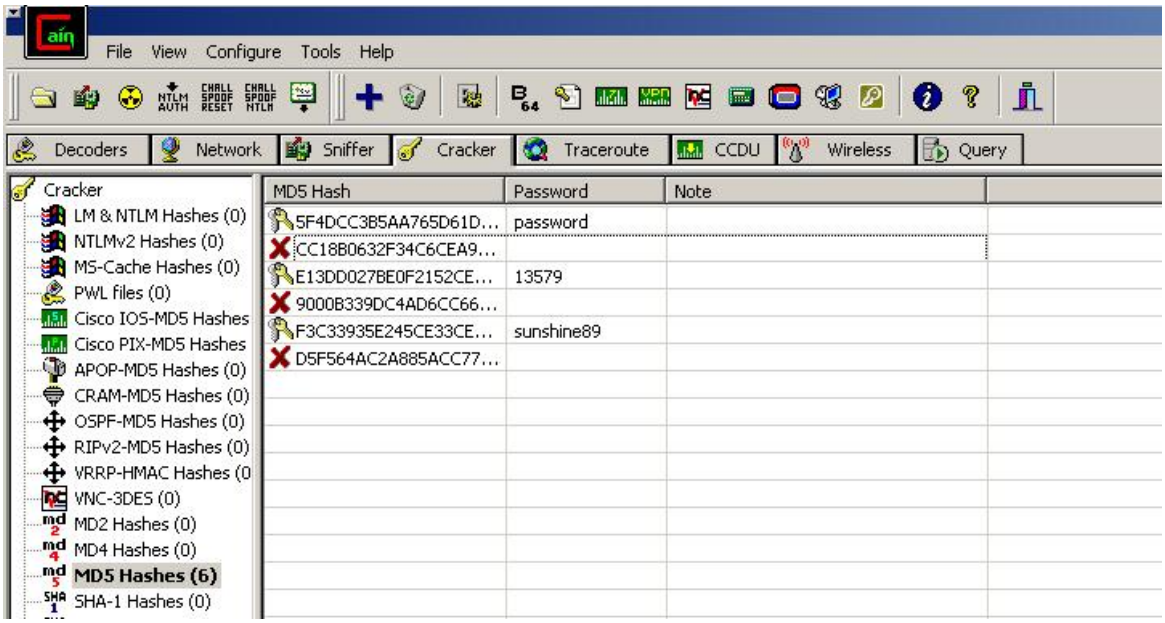


Step 9: Click ‘start’ and watch the magic happens before your eyes! Once it ends ‘exit’ . Your result should be the same as below.

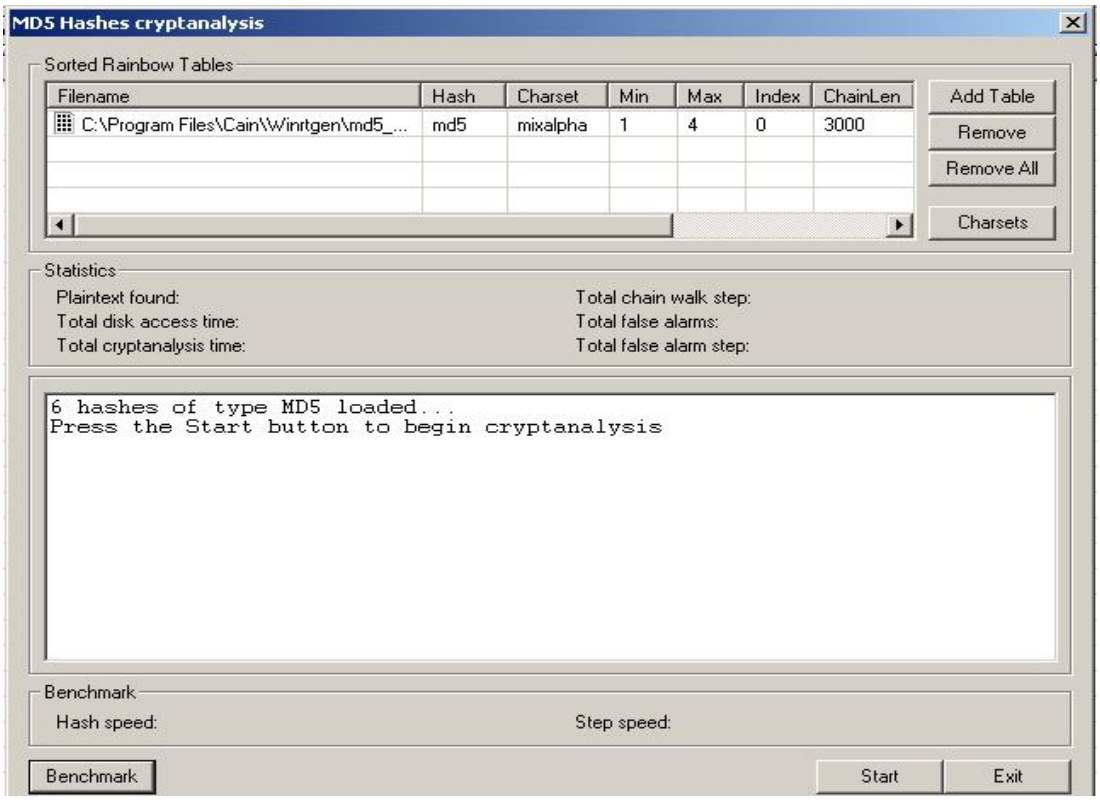


Part 2 : Rainbow Tables - Rainbow tables use pre-calculated MD5 hashes sorted on a table(s) to compare to encrypted MD5 files in order to find a match thus cracking the password. This type of password cracking trades time and storage capacity.

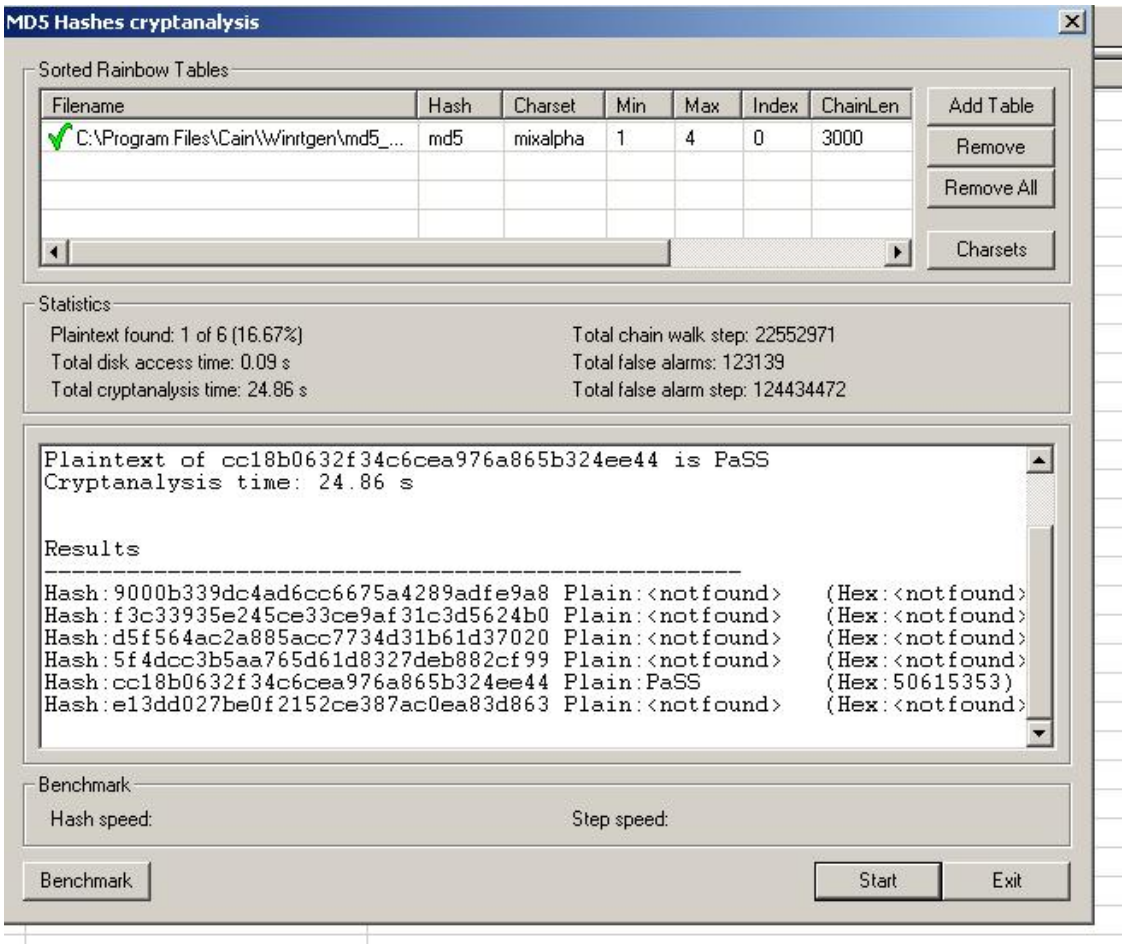
Step 1: Continuation from the previous 'Dictionary Attack' section. Cain & Abel should already be opened with following MD5 encrypted passwords.



Step 2: Now with the other half of the passwords still encrypted, we will be using rainbow table attacking to see if we can finally crack them. Select all six passwords, right click, and select 'Cryptanalysis Attack via RainbowTables'.

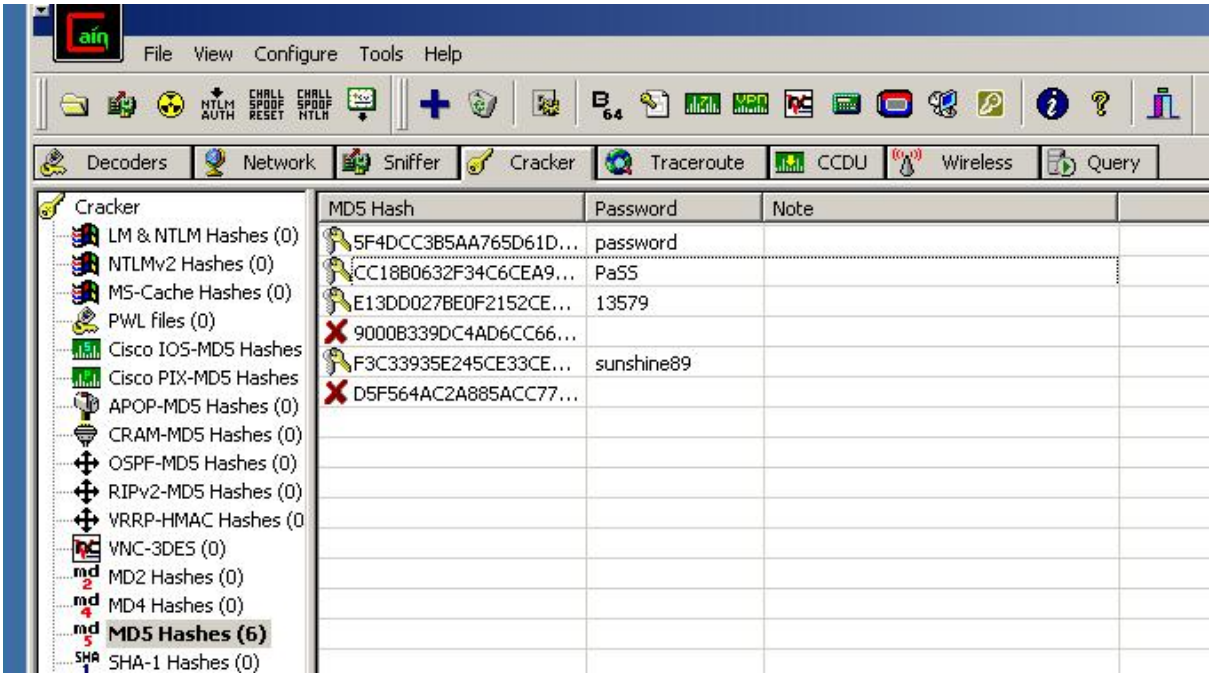


Step 3: A window will pop up and you could see under 'Sorted Rainbow Tables' there is already a MD5 rainbow table already added. Notice the specifications for that specific rainbow table. Click 'Start' when ready. 'Exit' when done.

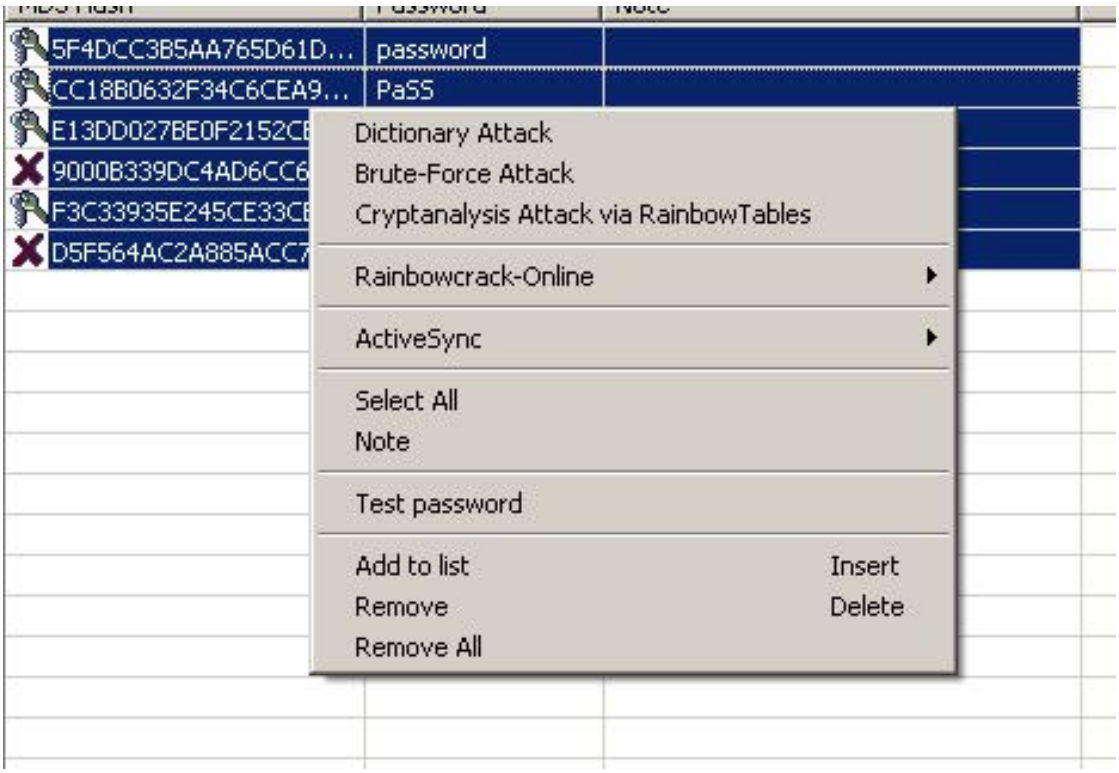


Part 3: Brute Force - Brute force attacks uses a finite but enormous number of combinations involving alphabet, numbers, and symbols in order to crack a password. This type of password cracking is usually used as a last resort as it's the most time consuming overall.

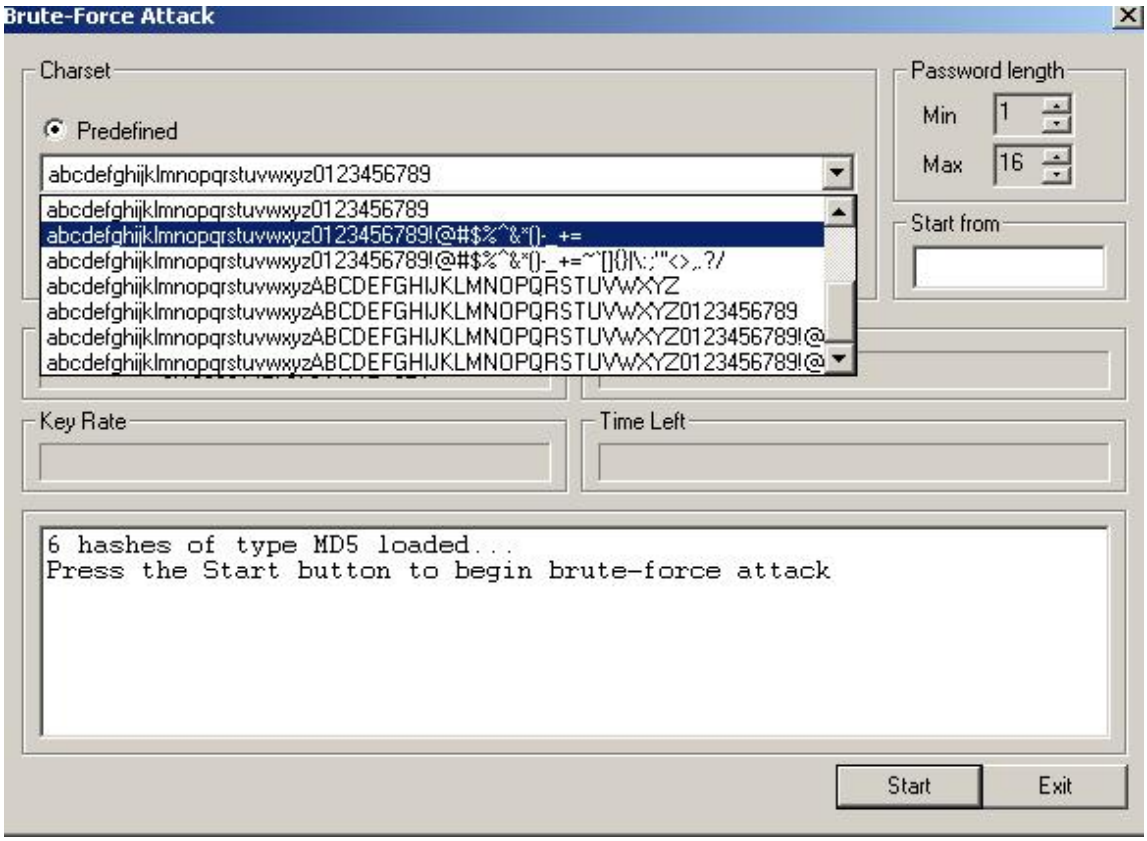
Step 1: Continuation from the previous 'Rainbow Tables' section. Cain & Abel should already be opened with the following MD5 encrypted passwords.



Step 2: Now with only two more passwords still encrypted, we will be using brute force attack to see if we can finally crack them. Select all six passwords, right click, and select 'Brute-Force Attack'.



Step 3 : Once a window appears we will have to adjust some settings to fit our requirements. Under Charset and Predefined selected, open the drop down bar and select the one below the initially selected one. Next, under Password length turn Max down to 5.



Step 4: When ready click 'Start'. Once it's done calculating 'Exit' . Your final results should be the same as below. All of them should be cracked! Yay!