

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:

```
30819f300d06092a864886f70d0101050003818d0030818902818100eea39d3e40f737cff2050d40515a4833c80cdd9073d88a3629aef45ee77793c78197eb0f8bbf0688c0672e22ee74e691bd536680
```

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': self.recipient, 'value': self.value, 'time' : self.time})
    def sign_transaction(self):
        private_key = self.sender._private_key
        signer = PKCS1_v1_5.new(private_key)
        h = SHA.new(str(self.to_dict()).encode('utf8'))
        return binascii.hexlify(signer.sign(h)).decode('ascii')
Dinesh = Client()
Ramesh = Client()

t = Transaction(Dinesh, Ramesh.identity, 5.0)

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

Output:

```
0f18c12abcf2904299162ce5c39689df19a9bf583689ec58182cac42617da5b1282d467b6a92ffd358f33052abbeefd502d1bbfcfcde293962e8ecec7b9f6a249df1bfcacf537a19404913e73079fc58c1f
```



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: 30819f300d06092a864886f70d010101050003818d0030818902818100c26c34efd785e255a351885faa617e22dbec4b8d1fdac6f833f00172431f5de6eecfe23c9fdcc14fd4de2dc947e11435fb572d39709f057b3039
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100d31bde02b9759d166f6514250948351f541d716b1ef5d7f3d931de74527050671e57bc3f323940186b105c379f8cd6c28935e43ba5752434a
-----
value: 15.0
-----
time: 2022-07-30 17:35:06.809777
-----
-----
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100c26c34efd785e255a351885faa617e22dbec4b8d1fdac6f833f00172431f5de6eecfe23c9fdcc14fd4de2dc947e11435fb572d39709f057b3039
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100a72cfb527dff9615a33eaf34b0cdf26f68d5d604676786c77ea188380e8412420aee1cde5b6dc9e157b31874d7da7fb40c5c835a0999e2b
-----
value: 6.0
-----
time: 2022-07-30 17:35:06.811866
-----
-----
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100d31bde02b9759d166f6514250948351f541d716b1ef5d7f3d931de74527050671e57bc3f323940186b105c379f8cd6c28935e43ba5752434a814
-----
recipient: 30819f300d06092a864886f70d010101050003818d0030818902818100c7e4e5f5f46085be74cd9c2f0ee1a79942c8a7b10c103a72b542c3246adf0c14f7a3032f5d431942062f5747e2369e6894e51cfbacfb5d578
-----
value: 2.0
-----
time: 2022-07-30 17:35:06.814247
-----
-----
sender: 30819f300d06092a864886f70d010101050003818d0030818902818100a72cfb527dff9615a33eaf34b0cdf26f68d5d604676786c77ea188380e8412420aee1cde5b6dc9e157b31874d7da7fb40c5c835a0999e2b268
-----
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: 11e0a4b57bb76496ecc6ab5a3c126165bc9dbbaef80f664e7e192a422360c3884
after 419 iterations found nonce: 11280dfd9ab05b3dbfa869990153732941408faa4a3b0832819b161f52321c08
after 511 iterations found nonce: 1150944bd7ea429acd052da390b148f27eb881686e0f8bfcdf77f833af878e2e
after 822 iterations found nonce: 110e61d41d94b48f6dfcb6f43f9ceafe2ab7168456dab088e05126d43d0366ef
after 924 iterations found nonce: 11147bf32bafeee4505b5cc40c3b227366d8a41ab3bd740e437c74400b7a8127
'6d80c4cf3cd4fb49aad25deced696c3a48e17974ecfb1441e2128517f0152b2'
```

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: 11a65418aca0201df85c2eeacabbcf0624092abb64bf6e159ccf5539cb1cfe1
after 527 iterations found nonce: 11310267f14590e45be2375fe94e881c65f86020e3fcf7844a849241783680a7
after 718 iterations found nonce: 117ba8286d575afe73007e6f588b6f541580bec444d5165a3f71d9e4b623f3fa
after 308 iterations found nonce: 111b1715b007a9920e76b241e9a4dde15a50b00faeed353bf9279311deb59804
after 398 iterations found nonce: 11a65418aca0201df85c2eeacabbcf0624092abb64bf6e159ccf5539cb1cfe1
after 527 iterations found nonce: 11310267f14590e45be2375fe94e881c65f86020e3fcf7844a849241783680a7
after 718 iterations found nonce: 117ba8286d575afe73007e6f588b6f541580bec444d5165a3f71d9e4b623f3fa
after 308 iterations found nonce: 111b1715b007a9920e76b241e9a4dde15a50b00faeed353bf9279311deb59804
after 398 iterations found nonce: 11a65418aca0201df85c2eeacabbcf0624092abb64bf6e159ccf5539cb1cfe1
after 527 iterations found nonce: 11310267f14590e45be2375fe94e881c65f86020e3fcf7844a849241783680a7
after 718 iterations found nonce: 117ba8286d575afe73007e6f588b6f541580bec444d5165a3f71d9e4b623f3fa
```

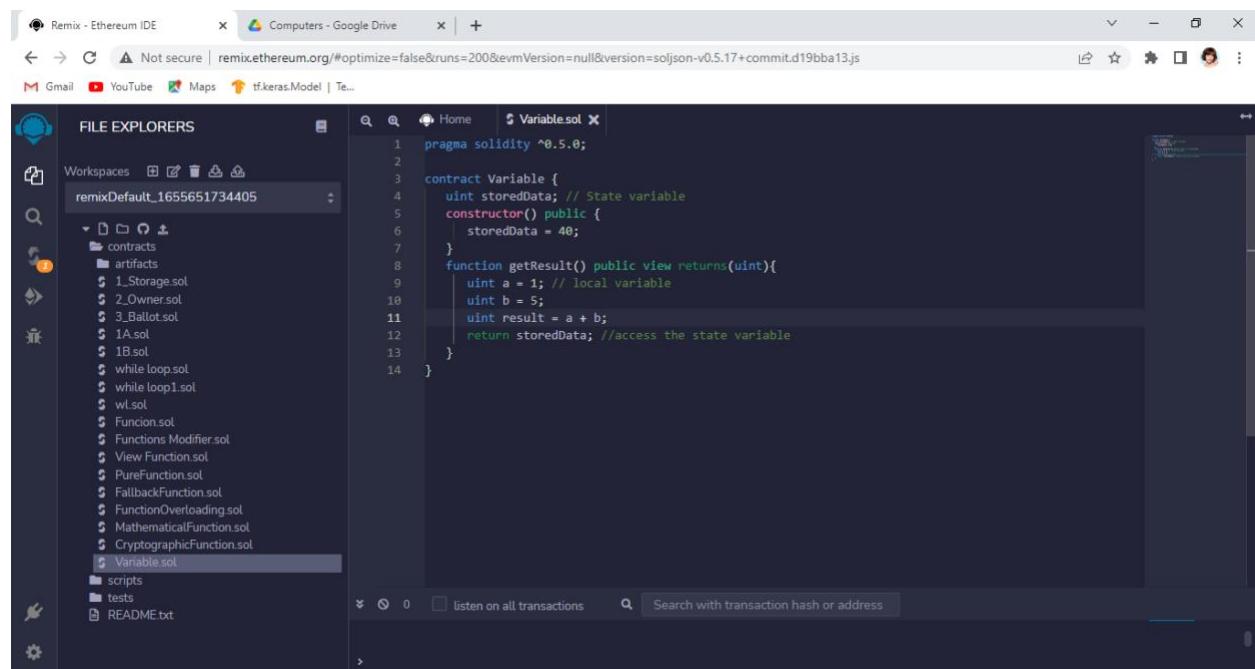
Practical No: 2

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:



The screenshot shows the Remix Ethereum IDE interface. The top bar displays the title "Remix - Ethereum IDE" and the URL "remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.5.17+commit.d19bba13.js". Below the title bar is a toolbar with various icons. The main area is divided into two panes: "FILE EXPLORERS" on the left and the "Contracts" pane on the right. The Contracts pane shows a workspace named "remixDefault_1655651734405" containing several Solidity files under the "contracts" folder, including "artifacts", "1_Storage.sol", "2_Owner.sol", "3_Ballot.sol", "1A.sol", "1B.sol", "while loop.sol", "while loop1.sol", "wl.sol", "Function.sol", "Functions Modifier.sol", "View Function.sol", "PureFunction.sol", "FallbackFunction.sol", "FunctionOverloading.sol", "MathematicalFunction.sol", "CryptographicFunction.sol", and "Variable.sol". The "Variable.sol" file is currently selected and displayed in the central editor pane. The code for "Variable.sol" is as follows:

```
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
    }
}
```

Output:

The screenshot shows the Ethereum IDE (Remix) interface. The top bar displays the title "Remix - Ethereum IDE" and the URL "remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.5.17+commit.d19bba13.js". Below the title bar, there are several browser tabs and icons.

The main area is titled "DEPLOY & RUN TRANSACTIONS". It features a sidebar with icons for location, address, search, and deployment. A message in the sidebar states: "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."

The central code editor window contains the following Solidity code:

```
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
    }
}
```

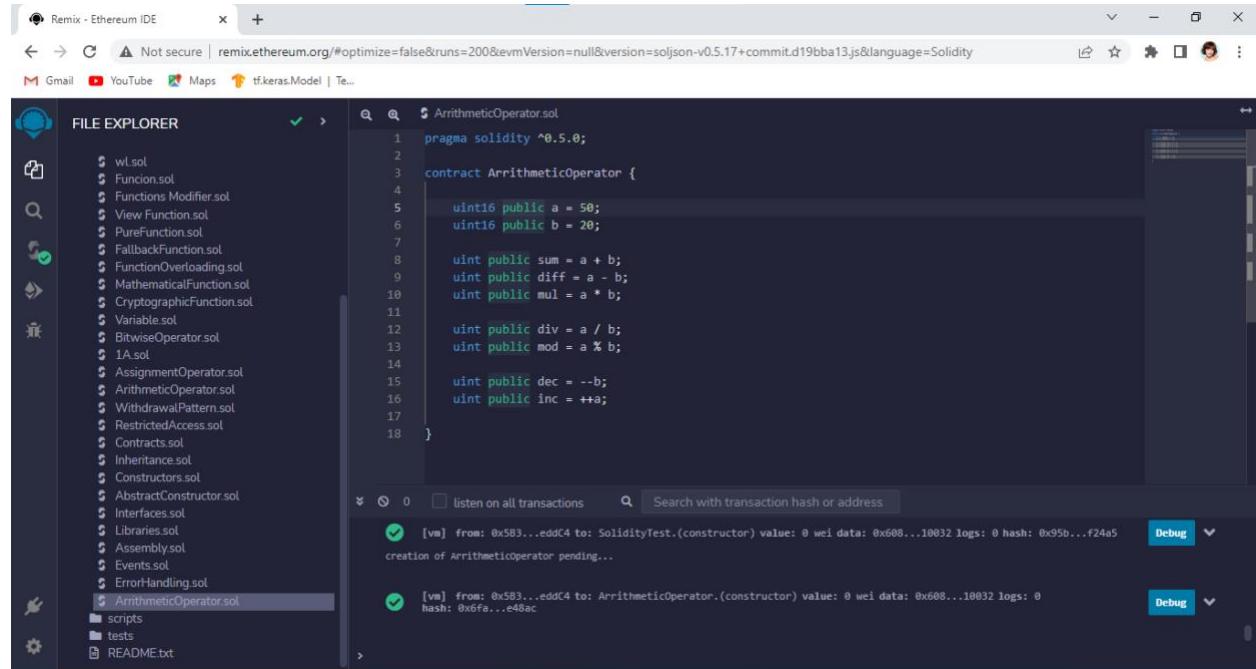
Below the code editor, the "Deployed Contracts" section shows a deployed contract named "VARIABLE AT 0xEF9...10EBF (MEMO)". Under this contract, the "getResult" function is listed with a value of "0: uint256: 40".

The "Low level interactions" section includes a "CALLDATA" button and a "Transact" button.

Operators:

Arithmetic Operator

Code:



The screenshot shows the Remix Ethereum IDE interface. The left sidebar is titled "FILE EXPLORER" and lists several Solidity files: wl.sol, Funcion.sol, Functions Modifier.sol, View Function.sol, PureFunction.sol, FallbackFunction.sol, FunctionOverloading.sol, MathematicalFunction.sol, CryptographicFunction.sol, Variable.sol, BitwiseOperator.sol, 1A.sol, AssignmentOperator.sol, ArithmeticOperator.sol, WithdrawalPattern.sol, RestrictedAccess.sol, Contracts.sol, Inheritance.sol, Constructors.sol, AbstractConstructor.sol, Interfaces.sol, Libraries.sol, Assembly.sol, Events.sol, ErrorHandling.sol, and ArithmeticOperator.sol. The "ArithmeticOperator.sol" file is currently selected. The main editor area displays the following Solidity code:

```
pragma solidity ^0.5.0;

contract ArithmeticOperator {
    uint16 public a = 50;
    uint16 public b = 20;

    uint public sum = a + b;
    uint public diff = a - b;
    uint public mul = a * b;

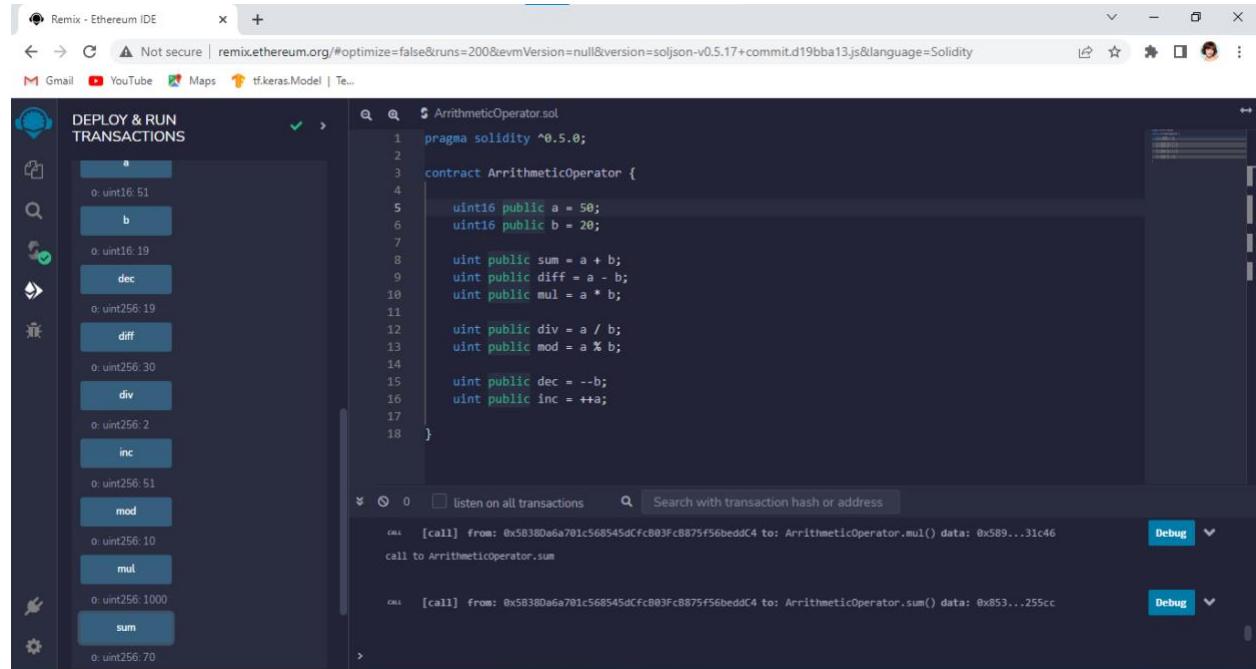
    uint public div = a / b;
    uint public mod = a % b;

    uint public dec = --b;
    uint public inc = ++a;
}
```

The bottom right pane shows two transaction logs:

- [vm] from: 0x583...edd4 to: SolidityTest.(constructor) value: 0 wei data: 0x608...10032 logs: 0 hash: 0x95b...f24a5 creation of ArithmeticOperator pending...
- [vm] from: 0x583...edd4 to: ArithmeticOperator.(constructor) value: 0 wei data: 0x608...10032 logs: 0 hash: 0x6fa...e48ac

Output:

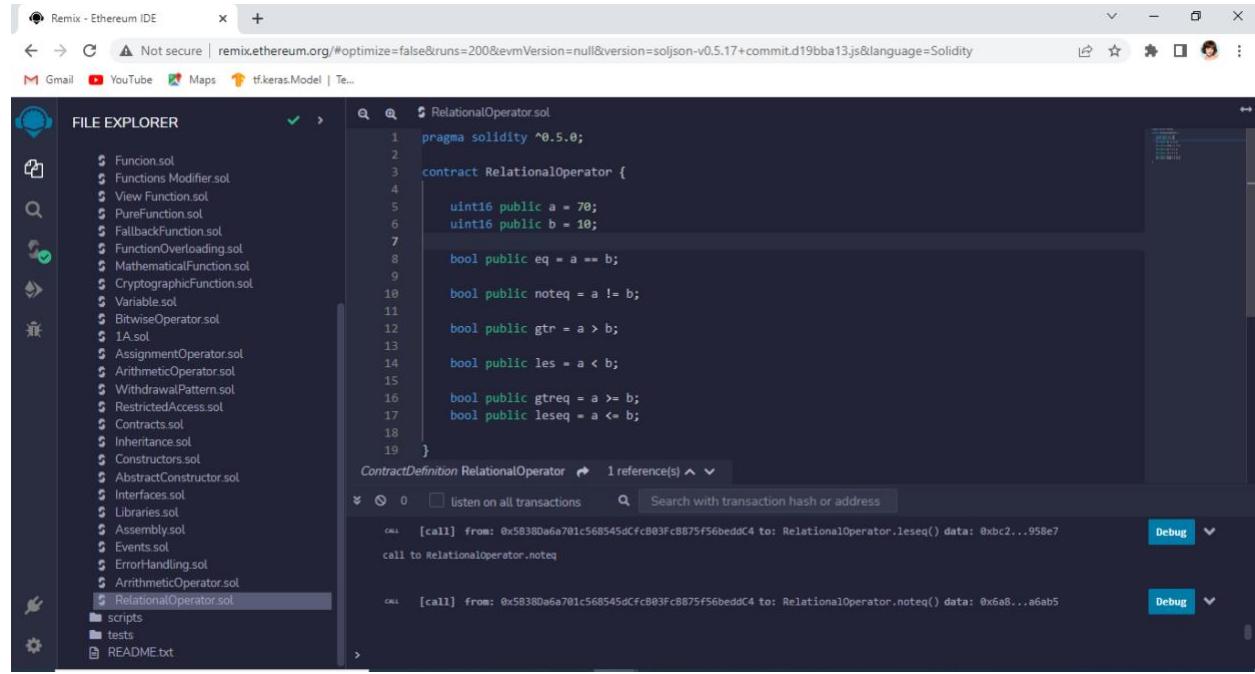


The screenshot shows the Remix Ethereum IDE interface with the "DEPLOY & RUN TRANSACTIONS" tab selected. On the left, there is a sidebar with buttons for "a", "b", "dec", "diff", "div", "inc", "mod", "mul", and "sum". The "mul" button is highlighted. The main editor area shows the same Solidity code as the previous screenshot. The bottom right pane shows transaction logs:

- call [call] from: 0x58380a6a701c568545dCfcB03FcB8875f56beddC4 to: ArithmeticOperator.mul() data: 0x589...31c46 call to ArithmeticOperator.mul
- call [call] from: 0x58380a6a701c568545dCfcB03FcB8875f56beddC4 to: ArithmeticOperator.sum() data: 0x853...255cc call to ArithmeticOperator.sum

Relational Operator

Code:



The screenshot shows the Remix Ethereum IDE interface. The left sidebar is titled "FILE EXPLORER" and lists various Solidity files. The main editor area contains the following Solidity code:

```
pragma solidity ^0.5.0;

contract RelationalOperator {
    uint16 public a = 70;
    uint16 public b = 10;

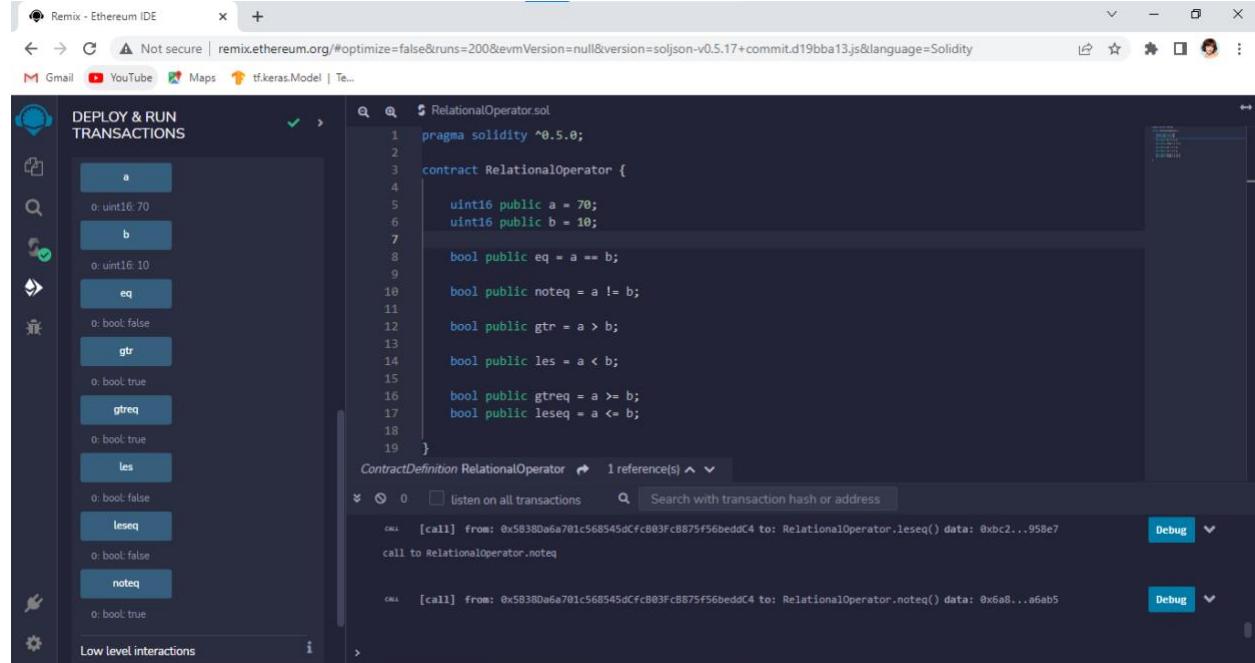
    bool public eq = a == b;
    bool public noteq = a != b;
    bool public gtr = a > b;
    bool public les = a < b;
    bool public gtreq = a >= b;
    bool public leseq = a <= b;
}
```

Below the code, the "ContractDefinition RelationalOperator" section shows two transaction logs:

- [call] from: 0x58380a6a701c568545dCfcB03FcB875f56beddC4 to: RelationalOperator.leseq() data: 0xbc2...958e7 call to RelationalOperator.leseq
- [call] from: 0x58380a6a701c568545dCfcB03FcB875f56beddC4 to: RelationalOperator.noteq() data: 0x6a8...a6ab5 call to RelationalOperator.noteq

At the bottom right, there are "Debug" buttons for each log entry.

Output:



The screenshot shows the Remix Ethereum IDE interface with the "DEPLOY & RUN TRANSACTIONS" sidebar open. The sidebar lists the variables and their assigned values:

- a: uint16: 70
- b: uint16: 10
- eq: bool: false
- gtr: bool: true
- gtreq: bool: true
- les: bool: false
- leseq: bool: false
- noteq: bool: true

The main editor area contains the same Solidity code as the previous screenshot. Below the code, the "ContractDefinition RelationalOperator" section shows the same two transaction logs as before.

Logical Operator

Code:

The screenshot shows the Remix Ethereum IDE interface. The left sidebar is titled "FILE EXPLORER" and lists various Solidity files. The main area displays the code for "LogicalOperator.sol". The code defines a contract named "logicalOperator" with three functions: Logic, which takes two bool inputs and returns three bool outputs; AND, which takes two bool inputs and returns a single bool output; and NOT, which takes a single bool input and returns a single bool output.

```
pragma solidity ^0.5.0;
// Creating a contract
contract logicalOperator{
    function Logic(
        bool a, bool b) public view returns(
        bool, bool, bool){
        // Logical AND operator
        bool and = a&b;
        // Logical OR operator
        bool or = a||b;
        // Logical NOT operator
        bool not = !a;
        return (and, or, not);
    }
}
```

Output:

The screenshot shows the Remix Ethereum IDE interface with the "DEPLOY & RUN TRANSACTIONS" sidebar active. It displays the deployed contract "LOGICALOPERATOR AT 0XB27..." and its methods. The "Logic" method is shown with inputs a: true and b: false, and the output is displayed as 0: bool: false, 1: bool: true, 2: bool: false. The "call" button is highlighted. The right side of the interface shows the same Solidity code as the previous screenshot.

Bitwise Operator

Code:

The screenshot shows the Remix Ethereum IDE interface. On the left, the FILE EXPLORER sidebar lists various Solidity files, with `BitwiseOperator.sol` selected. The main area displays the Solidity code for the `BitwiseOperator` contract. The code defines several public functions: `and`, `or`, `xor`, `leftshift`, `rightshift`, and `not`. Each function takes two `uint16` parameters, `a` and `b`.

```
pragma solidity ^0.5.0;

contract BitwiseOperator {

    uint16 public a = 20;
    uint16 public b = 50;

    uint16 public and = a & b;
    uint16 public or = a | b;
    uint16 public xor = a ^ b;
    uint16 public leftshift = a << b;
    uint16 public rightshift = a >> b;
    uint16 public not = ~a;
}
```

Output:

The screenshot shows the Remix Ethereum IDE interface with the `DEPLOY & RUN TRANSACTIONS` tab active. It displays the deployed contract `BITWISEOPERATOR AT 0xae0...9e`. Below the contract name, a list of functions is shown, each with its return value (`o:`) and operation (`a` or `b`). The functions listed are `and`, `leftshift`, `not`, `or`, `rightshift`, and `xor`. The `not` function returns `o: uint16: 65515`. The `or` function returns `o: uint16: 54`. The `rightshift` function returns `o: uint16: 0`. The `xor` function returns `o: uint16: 38`. At the bottom of the interface, a log message is displayed: `[vm] from: 0x583...addc4 to: LogicalOperator.(constructor) value: 0 wei data: 0x608...10032 logs: 0 hash: 0x5df...62867`.

Assignment Operator

Code:

The screenshot shows the Remix Ethereum IDE interface. The left sidebar is titled "FILE EXPLORER" and lists several Solidity files: contracts, artifacts, 1_Storage.sol, 2_Owner.sol, 3_Ballot.sol, while_loop.sol, while_loop1.sol, wl.sol, Funcion.sol, Functions.Modifier.sol, View.Function.sol, PureFunction.sol, FallbackFunction.sol, FunctionOverloading.sol, MathematicalFunction.sol, CryptographicFunction.sol, Variable.sol, BitwiseOperator.sol, 1A.sol, AssignmentOperator.sol, ArithmeticOperator.sol, WithdrawalPattern.sol, RestrictedAccess.sol, Contracts.sol, Inheritance.sol, Constructors.sol, AbstractConstructor.sol, Interfaces.sol, Libraries.sol. The "AssignmentOperator.sol" file is selected. The main editor area contains the following Solidity code:

```
pragma solidity ^0.5.0;

contract AssignmentOperator {

    uint16 public assignment = 20;
    uint16 public assignment_add = 50;
    uint16 public assignment_sub = 50;
    uint16 public assignment_mul = 10;
    uint16 public assignment_div = 50;
    uint16 public assignment_mod = 32;

    function getResult() public {
        assignment_add += 10;
        assignment_sub -= 20;
        assignment_mul *= 10;
        assignment_div /= 10;
        assignment_mod %= 20;
        return;
    }
}
```

The status bar at the bottom shows a transaction log: "[vm] from: 0x583...eddC4 to: AssignmentOperator.(constructor) value: 0 wei data: 0x608...10032 logs: 0 hash: 0xd1...5c92c".

Output:

The screenshot shows the Remix Ethereum IDE interface with the "DEPLOY & RUN TRANSACTIONS" sidebar open. Under "Deployed Contracts", the "ASSIGNMENTOPERATOR AT 0x9C" contract is listed. Below it, a list of functions is shown with their respective parameters and return types:

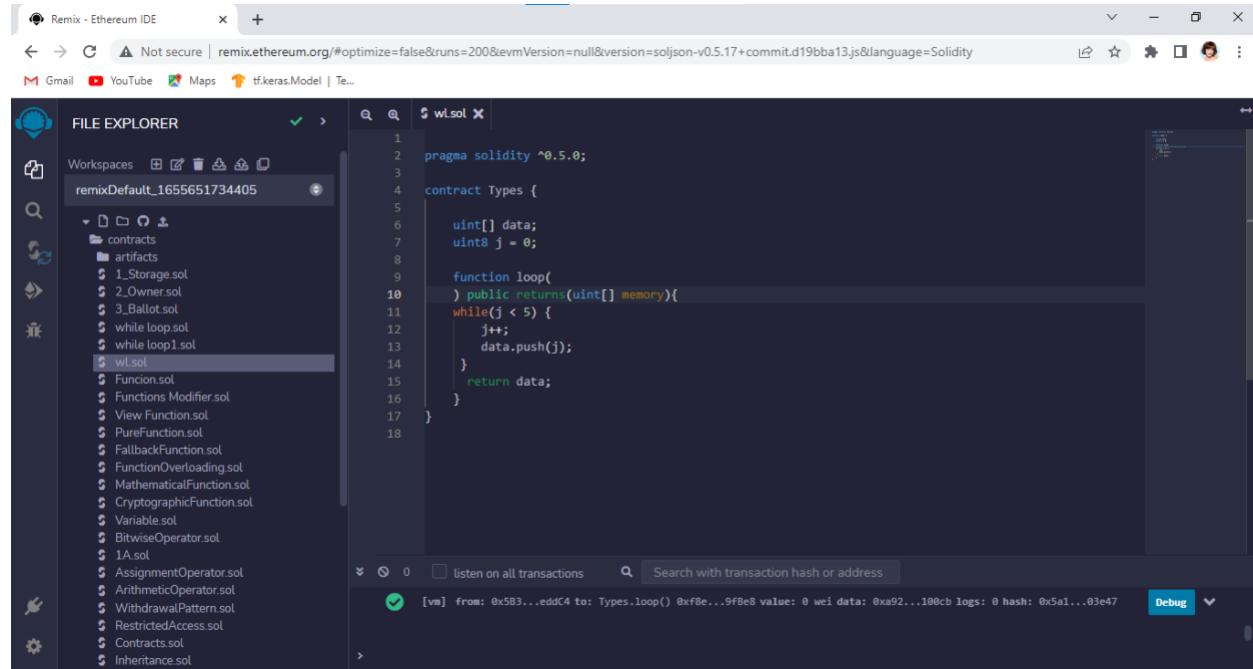
- getResult
- assignment
 - o: uint16:20
 - assignment_a...
o: uint16:60
 - assignment_d...
o: uint16:5
 - assignment_m...
o: uint16:12
 - assignment_m...
o: uint16:100
 - assignment_s...
o: uint16:30

The main editor area contains the same Solidity code as the previous screenshot. The status bar at the bottom shows a transaction log: "[call] [call] from: 0x5830a6a701c568545dCfcB03FcB875f56beddC4 to: AssignmentOperator.assignment_sub() data: 0x656...c8bda".

Loops

While Loop

Code:



The screenshot shows the Remix Ethereum IDE interface. The left sidebar is the File Explorer, displaying a workspace named "remixDefault_1655651734405" containing several Solidity contracts and artifacts. The central area is the code editor with the file "wl.sol" open, which contains the following Solidity code:

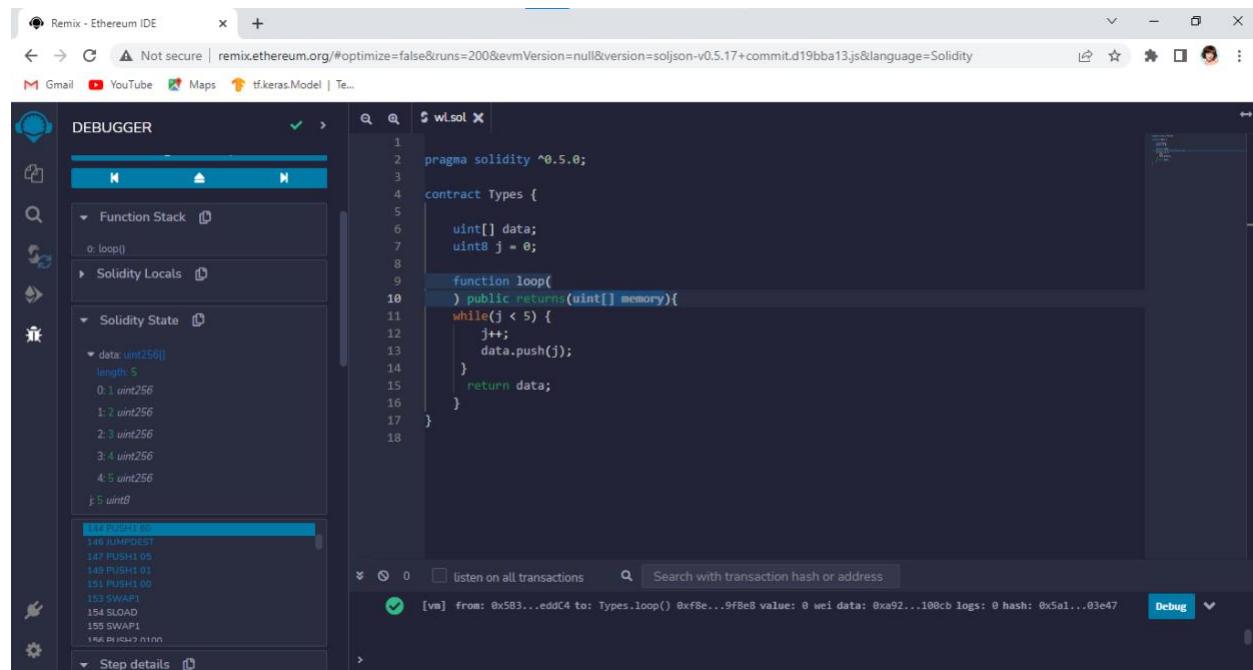
```
pragma solidity ^0.5.0;

contract Types {
    uint[] data;
    uint8 j = 0;

    function loop()
        public returns(uint[] memory)
    {
        while(j < 5) {
            j++;
            data.push(j);
        }
        return data;
    }
}
```

The bottom status bar shows a transaction log: "[vm] from: 0x583...edd4 to: Types.loop() 0xf8e...9f8e8 value: 0 wei data: 0xa92...100cb logs: 0 hash: 0x5a1...03e47". A "Debug" button is also visible.

Output:



The screenshot shows the Remix Ethereum IDE with the "DEBUGGER" tab selected. The left sidebar displays the Function Stack (with entry "0: (loop)") and Solidity Locals. The main code editor area is the same as the previous screenshot. The bottom status bar shows the same transaction log as before: "[vm] from: 0x583...edd4 to: Types.loop() 0xf8e...9f8e8 value: 0 wei data: 0xa92...100cb logs: 0 hash: 0x5a1...03e47". A "Step details" button is visible at the bottom.

Dowhile loop

Code:

The screenshot shows the Remix Ethereum IDE interface. The left sidebar is titled "FILE EXPLORER" and lists several Solidity files. The main editor window is titled "Dowhileloop.sol" and contains the following Solidity code:

```
pragma solidity ^0.5.0;
contract Dowhileloop {
    uint[] data;
    uint8 j = 0;
    function loop() public returns(uint[] memory){
        do{
            j++;
            data.push(j);
        }while(j < 8);
        return data;
    }
}
```

The status bar at the bottom shows transaction details: [vm] from: 0x583...eddC4 to: Dowhileloop.loop() 0xd91...39138 value: 0 wei data: 0xa92...100cb logs: 0 hash: 0x3d7...90655. A "Debug" button is also visible.

Output:

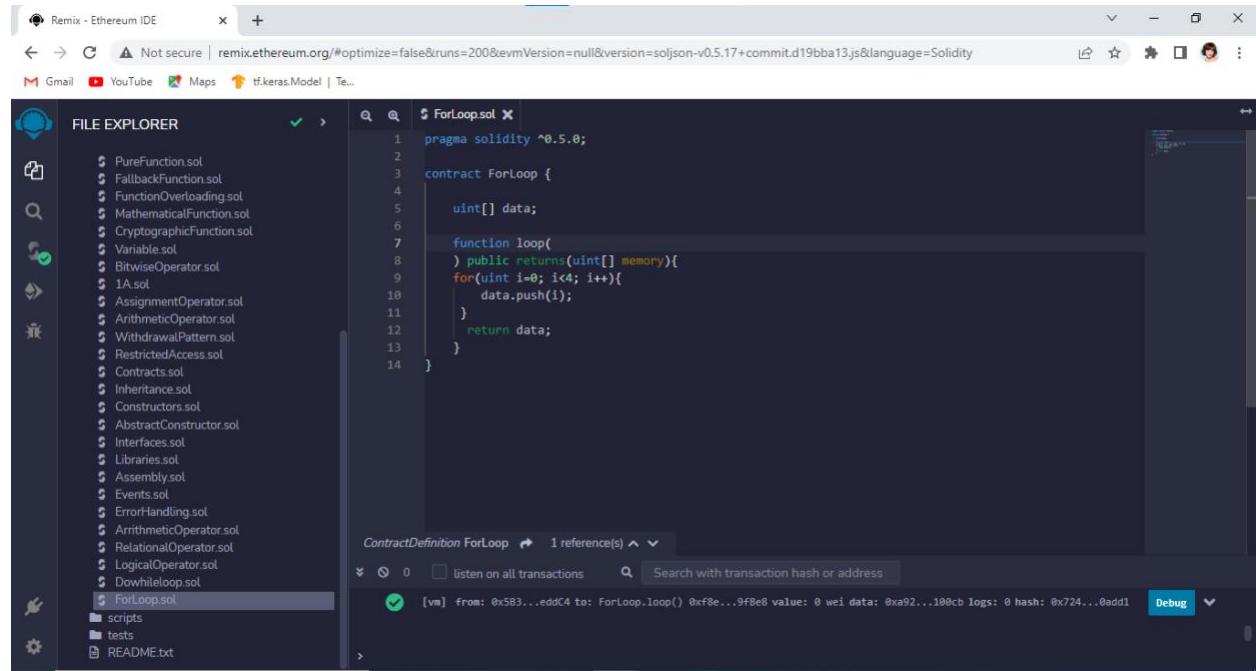
The screenshot shows the Remix Ethereum IDE interface with the "DEBUGGER" tab selected. The left sidebar displays the "Solidity Locals" and "Solidity State" sections. The "Solidity State" section shows the variable "data" as an array of 8 uint256 values, indexed from 0 to 7. The "Step details" section at the bottom shows the assembly trace for the contract, starting with JUMPDEST and PUSH1 instructions.

```
ContractDefinition Dowhileloop
```

```
143 JUMPDEST
144 PUSH1 60
146 JUMPDEST
147 PUSH1 01
149 PUSH1 00
151 DUP2
152 DUP2
```

For loop

Code:

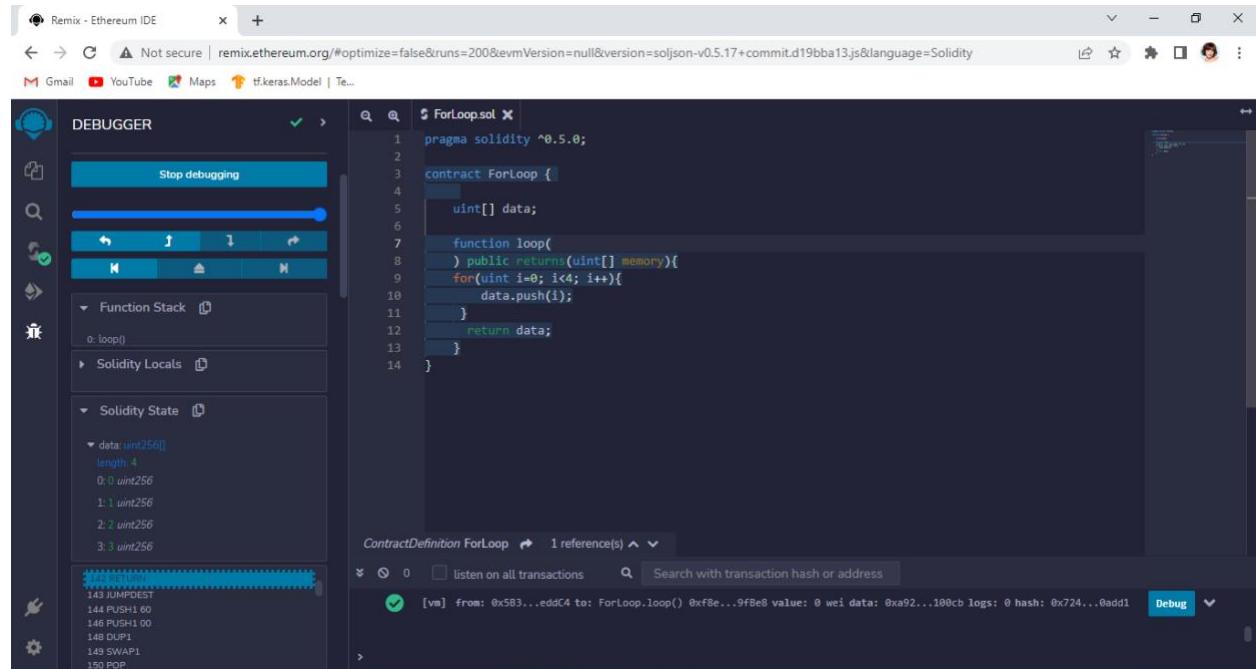


The screenshot shows the Remix Ethereum IDE interface. The left sidebar is titled "FILE EXPLORER" and lists several Solidity files. The main editor window is titled "ForLoop.sol" and contains the following Solidity code:

```
pragma solidity ^0.5.0;
contract ForLoop {
    uint[] data;
    function loop() public returns(uint[] memory){
        for(uint i=0; i<4; i++){
            data.push(i);
        }
        return data;
    }
}
```

The status bar at the bottom indicates a transaction from address 0x583...edd4 to ForLoop.loop() with a value of 0 wei, data of 0xa92...100cb, and a log hash of 0x724...0add1.

Output:



The screenshot shows the Remix Ethereum IDE with the "DEBUGGER" tab selected. The left sidebar has a "Stop debugging" button highlighted. The main editor window is still titled "ForLoop.sol" and displays the same Solidity code as before. The status bar at the bottom shows the same transaction details as the previous screenshot.

Decision making

If statement

Code:

The screenshot shows the Remix Ethereum IDE interface. The left sidebar is the File Explorer, listing various Solidity files. The main editor area contains the following Solidity code:

```
pragma solidity ^0.5.0;
contract IfStatement {
    uint i = 20;
    function decision_making() public returns(bool){
        if(i>0){
            return true;
        }
    }
}
```

The bottom right pane shows transaction logs:

- [vm] from: 0x5B3...edd4 to: ForLoop.loop() 0xf8e...9f8e8 value: 0 wei data: 0xa92...100cb logs: 0 hash: 0x724...0add1 creation of IfStatement pending...
- [vm] from: 0x5B3...edd4 to: IfStatement.(constructor) value: 0 wei data: 0x608...10032 logs: 0 hash: 0x484...7d5c1 transact to IfStatement.decision_making pending ...

Output:

The screenshot shows the Remix Ethereum IDE interface with the Debugger tab selected. The left sidebar includes a Debugger Configuration section with the option "Use generated sources (Solidity >= v0.7.2)" checked. The main editor area contains the same Solidity code as before. The bottom right pane displays the debugger's Function Stack, Solidity Locals, and Solidity State sections. The Function Stack shows a single entry: 0: decision_making(). The Solidity State section shows the variable `i: 20 uint256`. The assembly code at the bottom is:

```
077 JUMPDEST
078 PUSH1 00
080 PUSH1 14
082 PUSH1 00
084 SLOAD
085 LT
```

If...else statement

Code:

The screenshot shows the Remix Ethereum IDE interface. The left sidebar is titled 'FILE EXPLORER' and lists several Solidity files: FunctionOverloading.sol, MathematicalFunction.sol, CryptographicFunction.sol, Variable.sol, BitwiseOperator.sol, 1A.sol, AssignmentOperator.sol, ArithmeticOperator.sol, WithdrawalPattern.sol, RestrictedAccess.sol, Contracts.sol, Inheritance.sol, Constructors.sol, AbstractConstructor.sol, Interfaces.sol, Libraries.sol, Assemblysol, Events.sol, ErrorHandling.sol, ArithmeticOperator.sol, RelationalOperator.sol, LogicalOperator.sol, DoWhileLoop.sol, ForLoop.sol, IfStatement.sol, and IfElse.sol. The file 'IfElse.sol' is currently selected. The main editor area contains the following Solidity code:

```
pragma solidity ^0.5.0;

// Creating a contract
contract IfElse {
    // Declaring state variables
    uint i = 20;
    bool even;

    function decision_making()
        public payable returns(bool){
        if(i%2 == 0){
            even = true;
        }
        else{
            even = false;
        }
        return even;
    }
}
```

Below the code, the status bar indicates 'Execution cost: 22744 gas' and 'FunctionDefinition decision_making'. There is also a note about '1 reference(s)'.

Output:

The screenshot shows the Remix Ethereum IDE interface with the 'DEBUGGER' tab selected. The left sidebar now displays the 'DEBUGGER' section, which includes a 'DEBUGGER CONFIGURATION' panel with a checked checkbox for 'Use generated sources (Solidity >= v0.7.2)' and a 'Stop debugging' button. Below this are buttons for step operations (back, forward, run, etc.). The 'Function Stack' and 'Solidity Locals' sections are also visible. The main editor area shows the same Solidity code as before. The status bar at the bottom shows 'ContractDefinition IfElse' and '1 reference(s)'. The bottom pane displays the assembly code and its corresponding decoded output:

input	0x80...3aF24
decoded input	()
decoded output	{ "0": "bool: true" }

If...else if...else statement

Code:

The screenshot shows the Remix Ethereum IDE interface. The left sidebar is titled 'FILE EXPLORER' and lists several Solidity files. The main editor window is titled '# IfElseIf.sol' and contains the following Solidity code:

```
pragma solidity ^0.5.0;
contract IfElseIf {
    uint i = 10;
    string result;
    function decision_making() public returns(string memory){
        if(i<10){
            result = "less than 10";
        }
        else if(i == 10){
            result = "equal to 10";
        }
        else{
            result = "greater than 10";
        }
        return result;
    }
}
```

Output:

The screenshot shows the Remix Ethereum IDE interface with the 'DEBUGGER' tab selected. The left sidebar shows a 'FUNCTION STACK' with 'decision_making()' at the top. The main editor window is titled '# IfElseIf.sol' and contains the same Solidity code as above. The bottom pane shows the assembly output for the 'decision_making()' function:

```
ContractDefinition IfElseIf 1 reference(s) ^ v
0x3983f1150fa2158e0e5b0075b29c...
0: decision_making()
1: 10 uint256
result: equal to 10 string
2: 179 JUMPDEST
180 PUSH1 60
182 PUSH1 0a
184 PUSH1 00
186 SLOAD
```

String

Code:

The screenshot shows the Remix Ethereum IDE interface. The left sidebar is titled "FILE EXPLORER" and lists several Solidity files. In the center, there are two tabs: "IfElse.sol" and "String.sol". The "String.sol" tab is active, displaying the following Solidity code:

```
pragma solidity ^0.5.0;
contract StringsExample {
    string text;
    function setText() public returns (string memory) {
        text = "Hello World";
        return text;
    }
}
```

Below the code editor, the status bar shows "ContractDefinition StringsExample" and "1 reference(s)". The bottom panel contains sections for "decoded input" and "Search with transaction hash or address".

Output:

The screenshot shows the Remix Ethereum IDE interface with the "DEBUGGER" tab selected. The left sidebar includes a "DEBUGGER CONFIGURATION" section with a checked checkbox for "Use generated sources (Solidity >= v0.7.2)" and a dropdown menu showing the address "0x02b5a7Bec919c54377899e8881ef...". Below this is a toolbar with various debugging buttons. The main area displays the same Solidity code as the previous screenshot. The bottom panel shows the "decoded input" and "decoded output" fields, where "decoded output" contains the JSON object: {"text": "Hello World"}. The assembly code at the bottom is partially visible.

Array

Code:

The screenshot shows the Remix Ethereum IDE interface. The left sidebar is titled "FILE EXPLORER" and lists several Solidity files. The main editor window is titled "Array.sol" and contains the following Solidity code:

```
// Creating a contract
contract ArrayExample {
    uint[] data
    = [10, 20, 30, 40, 50];
    int[] data1;
    function dynamic_array() public returns(
        uint[] memory, int[] memory){
        data1
        = [int(-60), 70, -80, 90, -100, -120, 140];
        return (data, data1);
    }
}
```

Below the code, the "ContractDefinition Types" section shows one reference. The bottom status bar indicates "29°C Rain showers" and the date "7/24/2022".

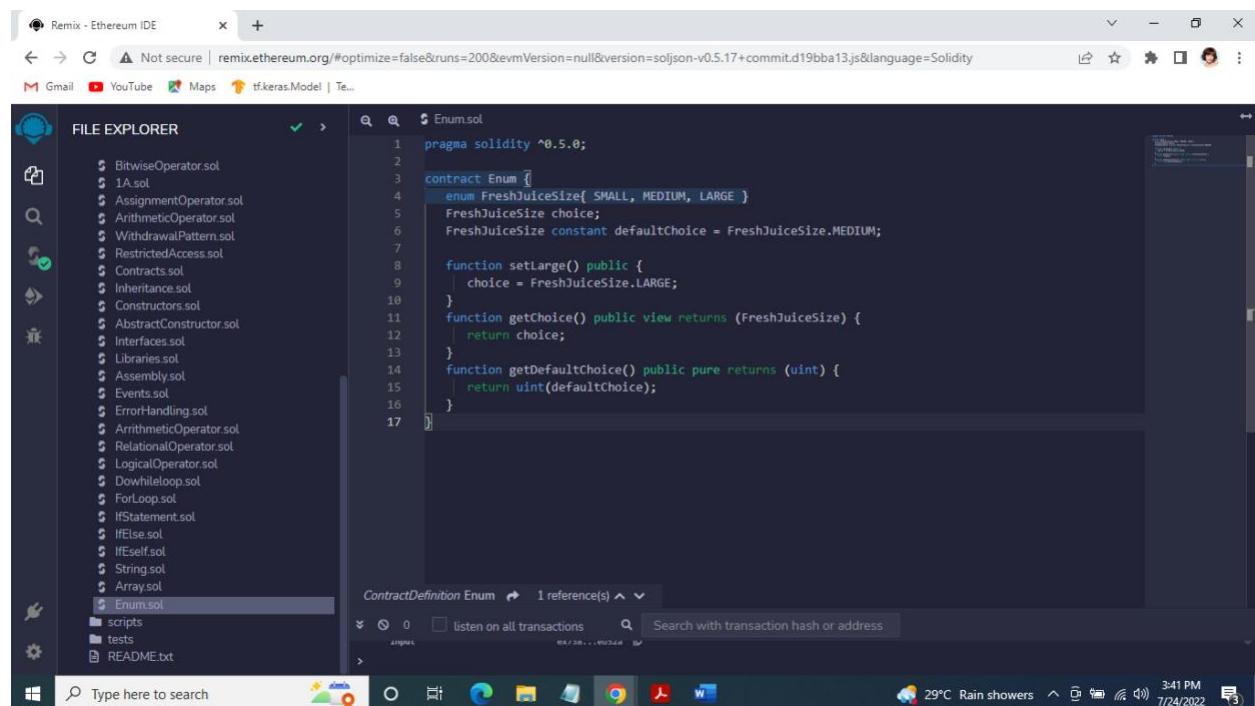
Output:

The screenshot shows the Remix IDE with the "DEBUGGER" tab selected. The left sidebar includes a "DEBUGGER CONFIGURATION" section with a checked checkbox for "Use generated sources (Solidity >= v0.7.2)" and a text input field containing "0x735733decb9b850d092d87c6aea4...". The main editor window is still titled "Array.sol" and displays the same Solidity code as the previous screenshot.

The "Function Stack" panel shows a single entry: "dynamic_array". The "Solidity Locals" panel shows variables "data" and "data1" both set to "[10, 20, 30, 40, 50]". The "Solidity State" panel shows assembly code starting with "215 JUMPDEST". The bottom status bar shows "29°C Rain showers" and the date "7/24/2022".

Enums

Code:



The screenshot shows the Remix Ethereum IDE interface. The left sidebar is titled "FILE EXPLORER" and lists several Solidity files: BitwiseOperator.sol, 1A.sol, AssignmentOperator.sol, ArithmeticOperator.sol, WithdrawalPattern.sol, RestrictedAccess.sol, Contracts.sol, Inheritance.sol, Constructors.sol, AbstractConstructor.sol, Interfaces.sol, Libraries.sol, Assembly.sol, Events.sol, ErrorHandling.sol, ArithmeticOperator.sol, RelationalOperator.sol, LogicalOperator.sol, Downwhileloop.sol, ForLoop.sol, IfStatement.sol, IfElse.sol, IfEself.sol, String.sol, Array.sol, and Enum.sol. The file "Enum.sol" is currently selected. The main editor area displays the following Solidity code:

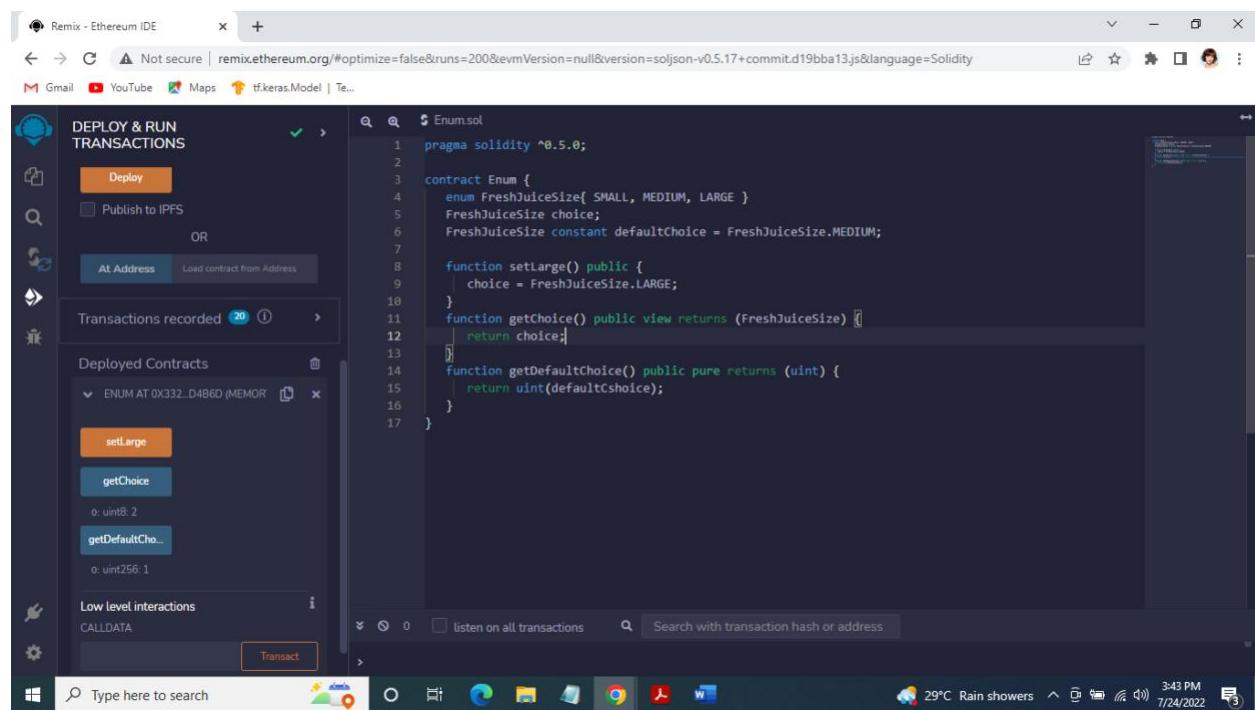
```
pragma solidity ^0.5.0;

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

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

The status bar at the bottom shows the date and time as 7/24/2022 3:41 PM.

Output:



The screenshot shows the Remix Ethereum IDE interface with the "DEPLOY & RUN TRANSACTIONS" sidebar open. The "Deploy" button is highlighted. Below it, there are options to "Publish to IPFS" or "At Address". The "At Address" option is selected, showing the address "0x332...D4B6D". Under "Transactions recorded", there are three buttons: "setLarge", "getChoice", and "getDefaultChoice". The "getChoice" button is currently active, showing the output "0: uint8: 2". The "getDefaultChoice" button shows the output "0: uint256: 1". At the bottom of the sidebar, there is a "Transact" button. The main editor area shows the same Solidity code as the previous screenshot. The status bar at the bottom shows the date and time as 7/24/2022 3:43 PM.

Struct

Code:

The screenshot shows the Remix Ethereum IDE interface. The left sidebar is titled "FILE EXPLORER" and lists several Solidity files: 1A.sol, AssignmentOperator.sol, ArithmeticOperator.sol, WithdrawalPattern.sol, RestrictedAccess.sol, Contracts.sol, Inheritance.sol, Constructors.sol, AbstractConstructor.sol, Interfaces.sol, Libraries.sol, Assembly.sol, Events.sol, ErrorHandling.sol, ArithmeticOperator.sol, RelationalOperator.sol, LogicalOperator.sol, Dowhileloop.sol, ForLoop.sol, IfStatement.sol, IfElse.sol, Ifself.sol, String.sol, Array.sol, Enum.sol, and Struct.sol. The file "Struct.sol" is currently selected and open in the main editor area. The code defines a struct Book with fields title, author, and book_id, and a contract Struct with a function setBook that creates a Book object and a function getBookId that returns its book_id.

```
pragma solidity ^0.5.0;
contract Struct {
    struct Book {
        string title;
        string author;
        uint book_id;
    }
    Book book;
    function setBook() public {
        book = Book('Learn Java', 'TP', 1);
    }
    function getBookId() public view returns (uint) {
        return book.book_id;
    }
}
```

Output:

The screenshot shows the Remix Ethereum IDE interface with the "DEPLOY & RUN TRANSACTIONS" sidebar active. The "Deploy" button is highlighted. Below it, there are options to "Publish to IPFS" and "At Address". The "Transactions recorded" section shows 24 transactions. Under "Deployed Contracts", there is a list item "STRUCT AT 0X93F.C96CC (MEMO)". Below this, two functions are listed: "setBook" and "getBookId". The "Low level interactions" section includes a "TRANSACTION" button. The main editor area shows the same Solidity code as the previous screenshot.

```
pragma solidity ^0.5.0;
contract Struct {
    struct Book {
        string title;
        string author;
        uint book_id;
    }
    Book book;
    function setBook() public {
        book = Book('Learn Java', 'TP', 4);
    }
    function getBookId() public view returns (uint) {
        return book.book_id;
    }
}
```

Mapping

Code:

The screenshot shows the Remix Ethereum IDE interface. The left sidebar is titled "FILE EXPLORER" and lists several Solidity files: AssignmentOperator.sol, ArithmeticOperator.sol, WithdrawalPattern.sol, RestrictedAccess.sol, Contracts.sol, Inheritance.sol, Constructors.sol, AbstractConstructor.sol, Interfaces.sol, Libraries.sol, Assembly.sol, Events.sol, ErrorHandling.sol, ArithmeticOperator.sol, RelationalOperator.sol, LogicalOperator.sol, DoWhileLoop.sol, ForLoop.sol, IfStatement.sol, IfElse.sol, IfSelf.sol, String.sol, Array.sol, Enum.sol, Struct.sol, and Mapping.sol. The file "Mapping.sol" is currently selected. The main editor area displays the following Solidity code:

```
pragma solidity ^0.4.18;

contract mapping_example {

    struct student {
        string name;
        string subject;
        uint8 marks;
    }

    mapping (address => student) result;
    address[] public student_result;

    function adding_values() public {
        var student
            = result[0x0EE7796E89C82C368Add1375076f39D69FafE252];

        student.name = "John";
        student.subject = "Chemistry";
        student.marks = 88;
        student_result.push(
            0x0EE7796E89C82C368Add1375076f39D69FafE252) - 1;
    }
}
```

The status bar at the bottom indicates it's 3:53 PM on 7/24/2022, with a weather forecast of 27°C Rain showers.

Output:

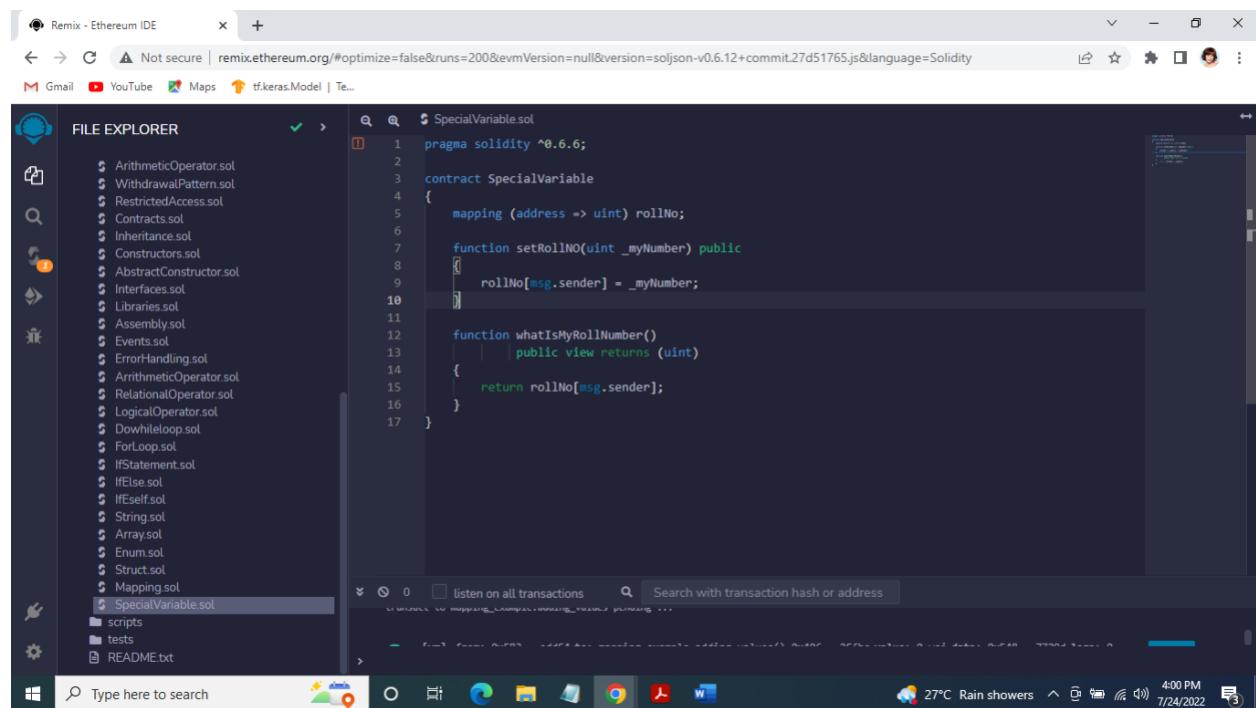
The screenshot shows the Remix Ethereum IDE interface with the "DEBUGGER" tab selected. The left sidebar now includes a "FUNCTION STACK" and "SOLIDITY LOCALS" section, along with the "SOLIDITY STATE" section which displays the state of the "result" mapping. The main editor area shows the same Solidity code as before. The bottom pane displays a transaction history with two entries:

- [vm] from: 0x583...edd4 to: mapping_example.(constructor) value: 0 wei data: 0x608...40029 logs: 0 hash: 0x806...231b7 transact to mapping_example.adding_values pending ...
- [vm] from: 0x583...edd4 to: mapping_example.adding_values() 0x406...2Cfbc value: 0 wei data: 0x548...7729d logs: 0 hash: 0x16b...f8d79 call to mapping_example.student_result errored: Error encoding arguments: Error: invalid BigNumber string (argument="value", value="", code=INVALID_ARGUMENT)

The status bar at the bottom indicates it's 3:55 PM on 7/24/2022, with a weather forecast of 27°C Rain showers.

Special Variable

Code:



The screenshot shows the Remix Ethereum IDE interface. The left sidebar is the File Explorer, listing various Solidity files such as ArithmeticOperator.sol, WithdrawalPattern.sol, RestrictedAccess.sol, Contracts.sol, Inheritance.sol, Constructors.sol, AbstractConstructor.sol, Interfaces.sol, Libraries.sol, Assembly.sol, Events.sol, ErrorHandling.sol, ArithmeticOperator.sol, RelationalOperator.sol, LogicalOperator.sol, DoWhileloop.sol, ForLoop.sol, IfStatement.sol, IfElse.sol, Ifself.sol, String.sol, Arrays.sol, Enum.sol, Struct.sol, Mapping.sol, and SpecialVariable.sol. The right pane displays the Solidity code for `SpecialVariable.sol`:

```
pragma solidity ^0.6.6;

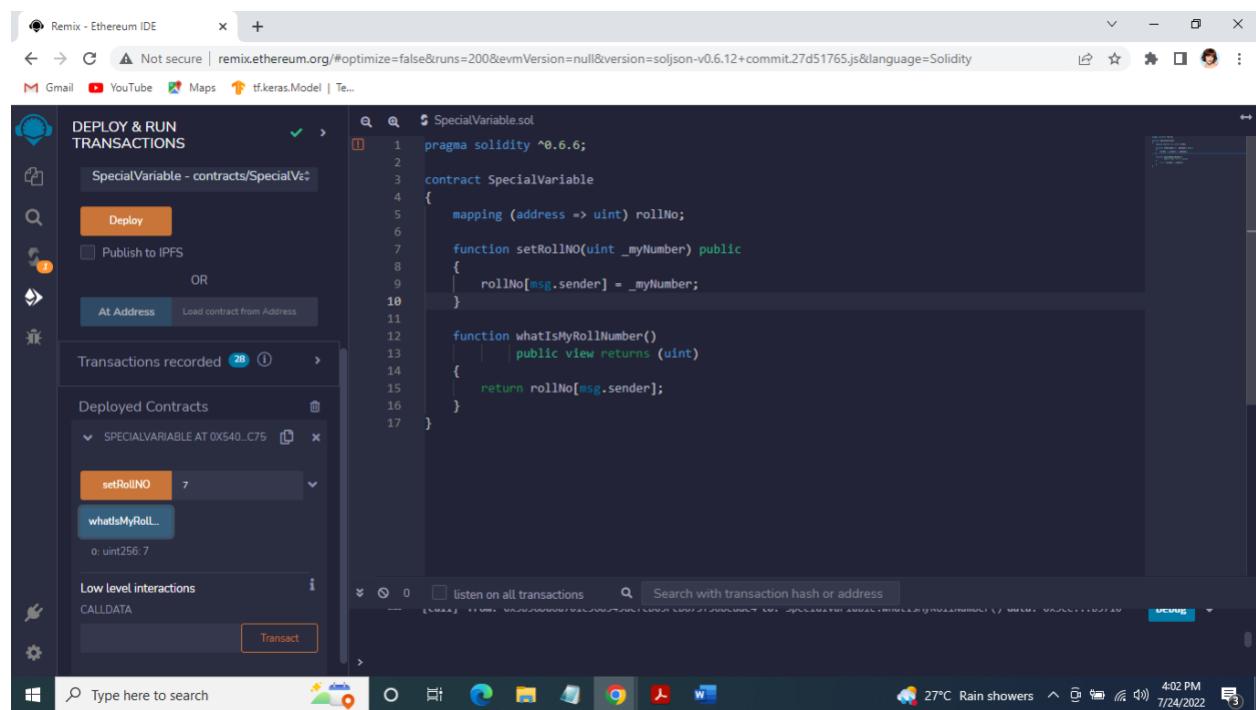
contract SpecialVariable {
    mapping (address => uint) rollNo;

    function setRollNO(uint _myNumber) public
    {
        rollNo[msg.sender] = _myNumber;
    }

    function whatIsMyRollNumber()
    | public view returns (uint)
    {
        return rollNo[msg.sender];
    }
}
```

The status bar at the bottom indicates it's 27°C Rain showers, 4:00 PM, and the date is 7/24/2022.

Output:

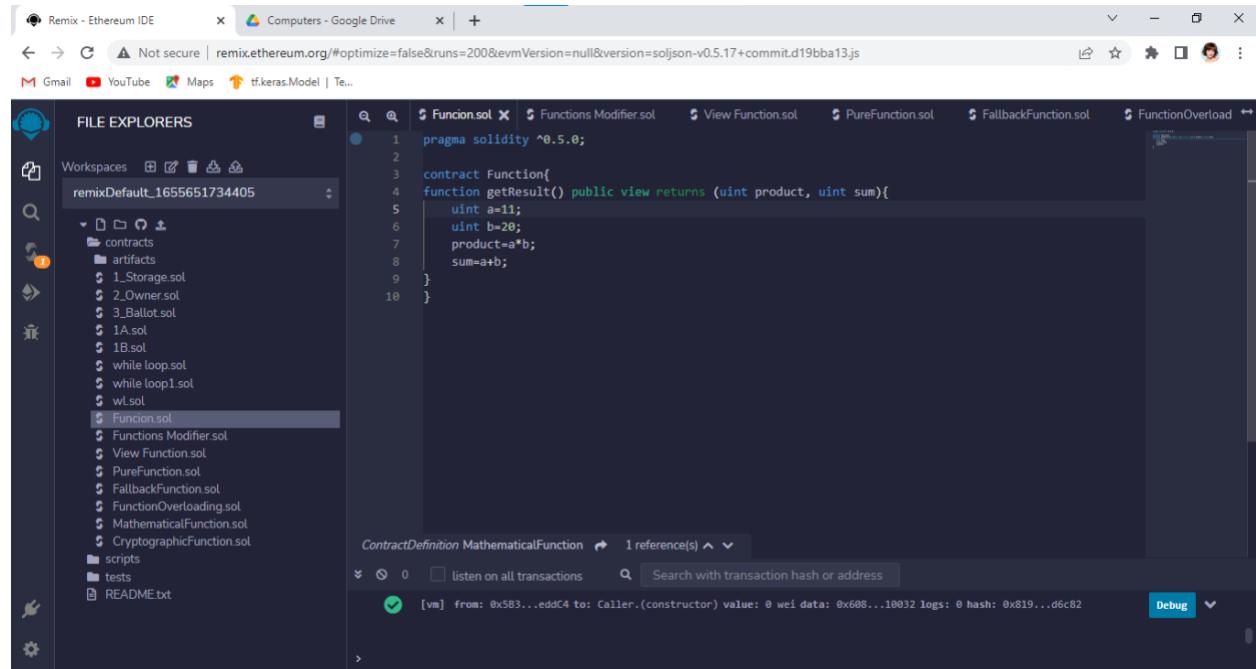


The screenshot shows the Remix Ethereum IDE interface with the "DEPLOY & RUN TRANSACTIONS" sidebar open. It displays the deployed contract `SPECIALVARIABLE` at address `0x540...C75`. Under the "Transactions recorded" section, there are two entries: `setRollNO` with value `7` and `whatIsMyRoll...`. The status bar at the bottom indicates it's 27°C Rain showers, 4:02 PM, and the date is 7/24/2022.

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

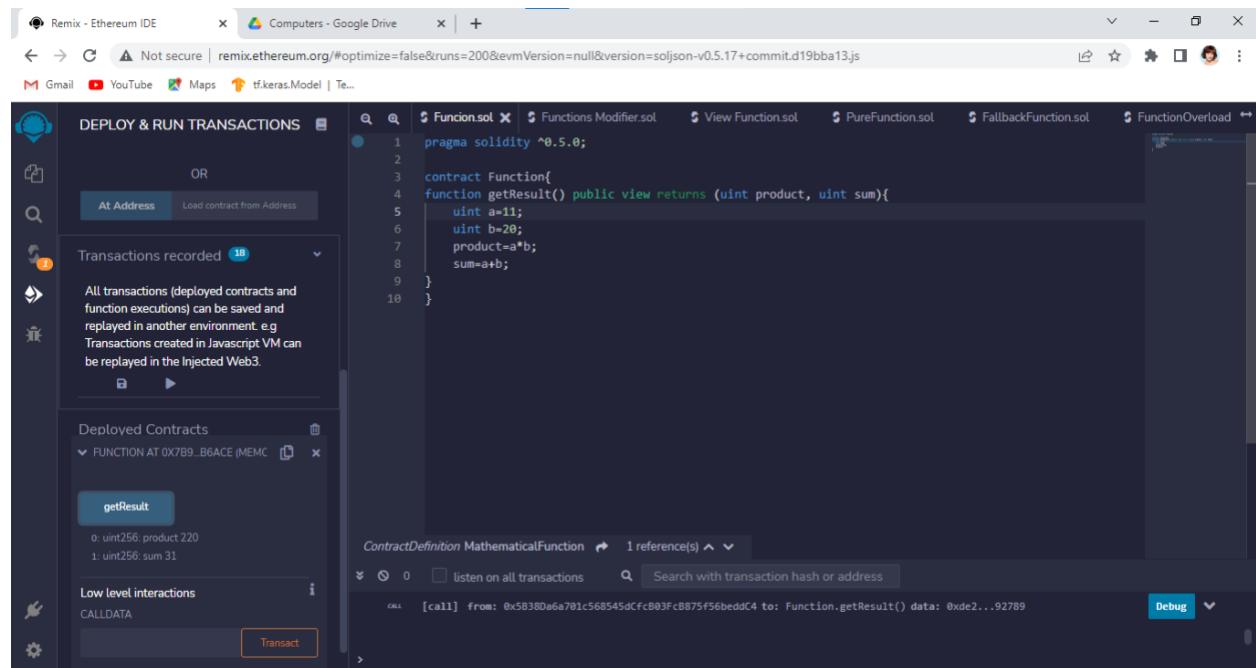
Function:

Code:



```
pragma solidity ^0.5.0;
contract Function{
    function getResult() public view returns (uint product, uint sum){
        uint a=11;
        uint b=20;
        product=a*b;
        sum=a+b;
    }
}
```

Output:



Deploy & Run Transactions

At Address: FUNCTION AT 0X7B9...B6ACE (MEMC)

getResult

0: uint256: product 220
1: uint256: sum 31

Low level interactions

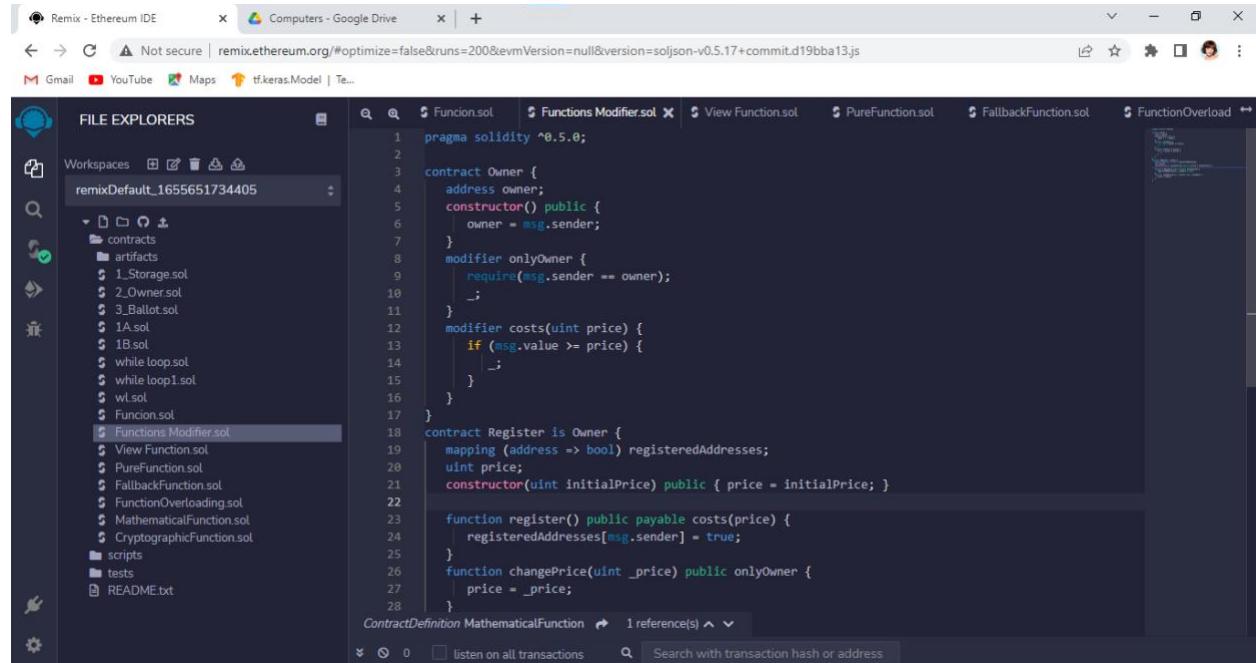
CALLDATA

ContractDefinition MathematicalFunction 1 reference(s) ▾

CALL [call] from: 0x58380a6a701c568545dCfcB03FcB875f56beddC4 to: Function.getResult() data: 0xde2...92789

Functions Modifiers

Code:



The screenshot shows the Remix Ethereum IDE interface. The left sidebar displays a file tree under 'FILE EXPLORERS' for a workspace named 'remixDefault_1655651734405'. The tree includes 'artifacts' (with files 1.Storage.sol, 2.Owner.sol, 3.Ballot.sol, 1A.sol, 1B.sol, while loop.sol, while loop1.sol, wl.sol, Funcion.sol), 'scripts', 'tests', and 'README.txt'. The central editor pane shows the Solidity code for 'Functions Modifier.sol'. The code defines a contract 'Owner' with a constructor that sets the owner to the msg.sender. It includes a modifier 'onlyOwner' that requires the sender to be the owner. Another modifier 'costs(uint price)' checks if the msg.value is greater than or equal to the specified price. The contract also has a constructor taking an initial price, a 'register' function that adds the sender to a mapping of addresses, and a 'changePrice' function that updates the price.

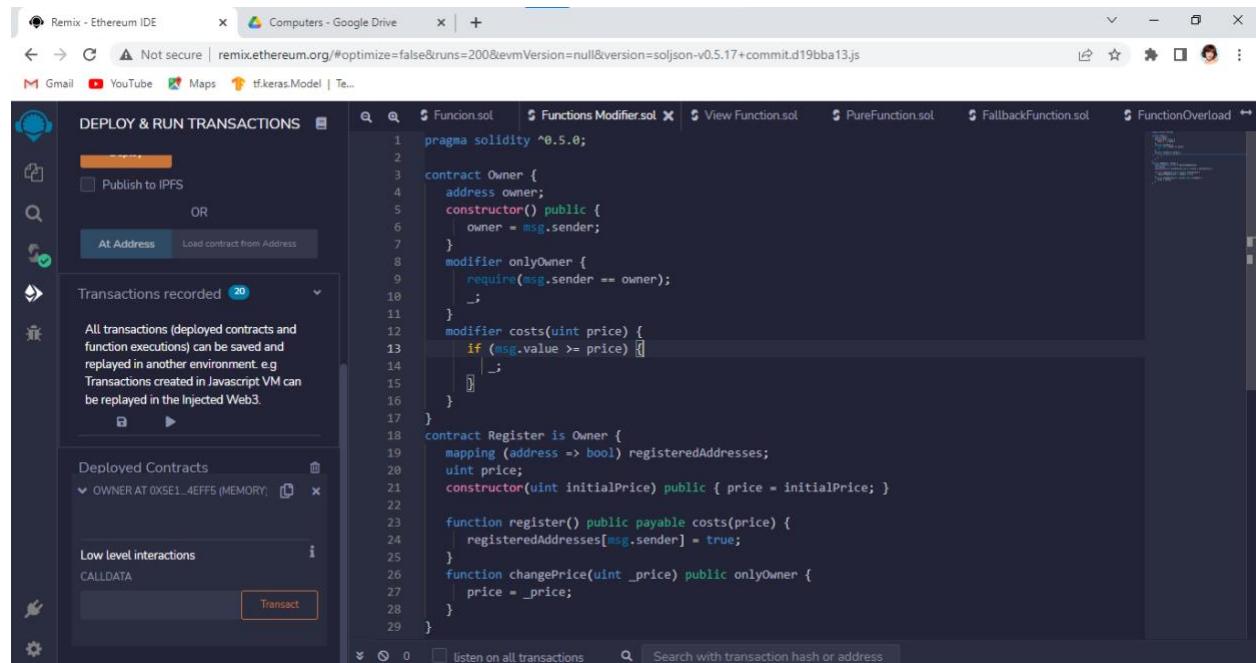
```
pragma solidity ^0.5.0;

contract Owner {
    address owner;
    constructor() public {
        owner = msg.sender;
    }
    modifier onlyOwner {
        require(msg.sender == owner);
    }
    modifier costs(uint price) {
        if (msg.value >= price) {
        }
    }
}

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

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

Output:



The screenshot shows the Remix Ethereum IDE interface with the 'DEPLOY & RUN TRANSACTIONS' tab selected. On the left, there's a sidebar for publishing to IPFS or loading from an address. Below it, a section titled 'Transactions recorded' shows a list of 20 transactions. A note explains that transactions can be saved and replayed. Under 'Deployed Contracts', the 'OWNER' contract is listed at address 0X5E1_4EFFS (MEMORY). A 'Low level interactions' section shows a 'Transact' button. The central editor pane shows the same Solidity code as the previous screenshot, with the 'FUNCTIONS MODIFIER' tab selected. The code defines the 'Owner' contract with its modifiers and functions.

```
pragma solidity ^0.5.0;

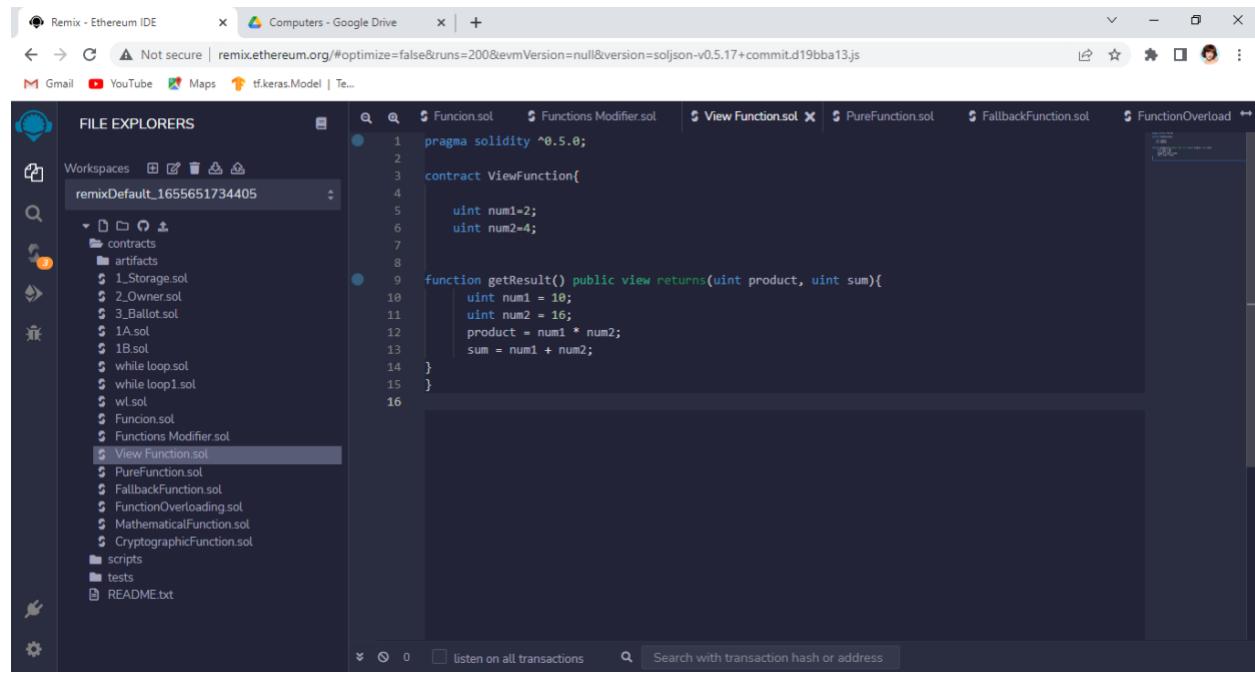
contract Owner {
    address owner;
    constructor() public {
        owner = msg.sender;
    }
    modifier onlyOwner {
        require(msg.sender == owner);
    }
    modifier costs(uint price) {
        if (msg.value >= price) {
        }
    }
}

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

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

View function

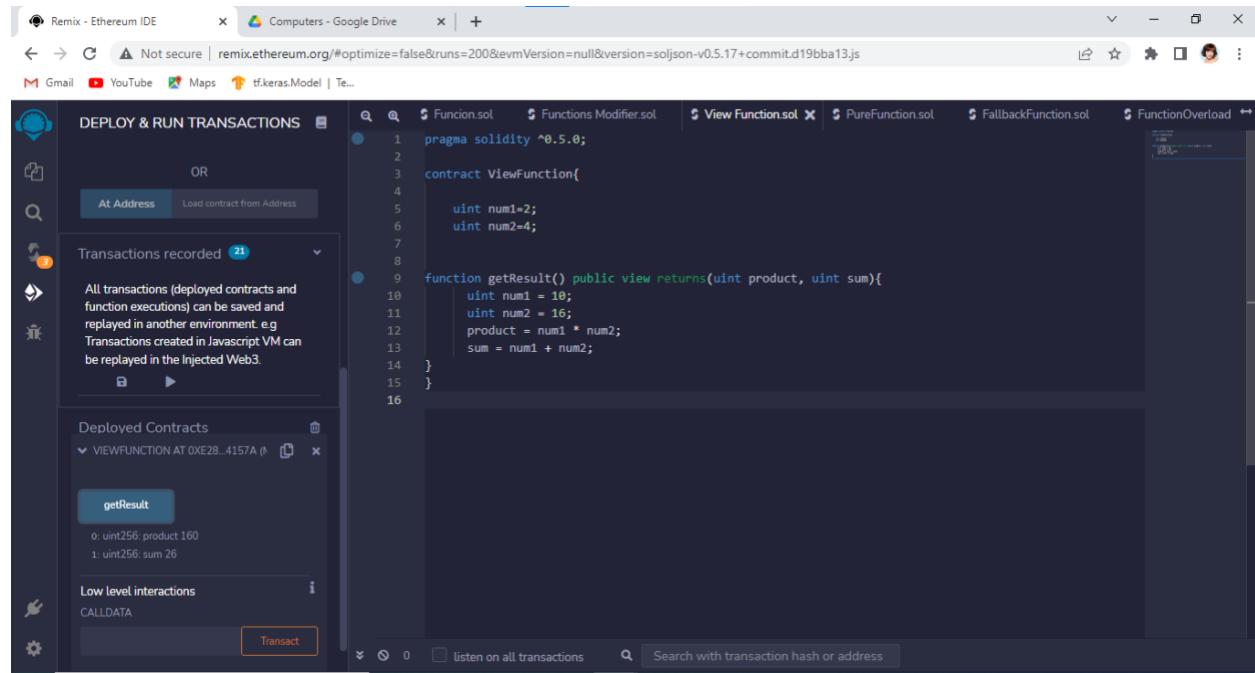
Code:



The screenshot shows the Remix Ethereum IDE interface. The left sidebar is titled "FILE EXPLORERS" and lists several Solidity files: Funcion.sol, Functions Modifier.sol, View Function.sol (which is currently selected), PureFunction.sol, FallbackFunction.sol, FunctionOverloading.sol, MathematicalFunction.sol, CryptographicFunction.sol, scripts, tests, and README.txt. The main workspace displays the source code for "View Function.sol". The code defines a contract named "ViewFunction" with a public view function "getResult" that calculates the product and sum of two uint variables, num1 and num2.

```
pragma solidity ^0.5.0;
contract ViewFunction{
    uint num1=2;
    uint num2=4;
    function getResult() public view returns(uint product, uint sum){
        uint num1 = 10;
        uint num2 = 16;
        product = num1 * num2;
        sum = num1 + num2;
    }
}
```

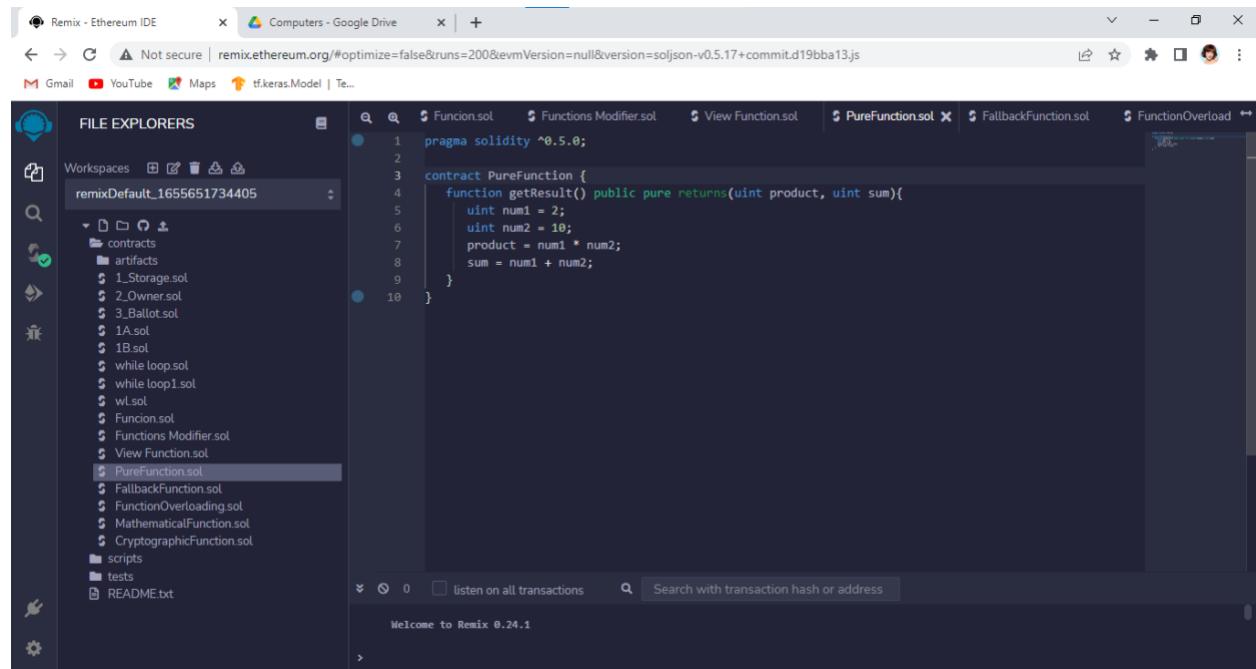
Output:



The screenshot shows the Remix Ethereum IDE interface with the "DEPLOY & RUN TRANSACTIONS" tab selected. It displays the same "View Function.sol" code as the previous screen. On the left, there is a "Transactions recorded" section with a note about saved transactions and a "Deployed Contracts" section showing a deployed contract named "VIEWFUNCTION AT 0xE28...4157A". Below this, the "getResult" function is highlighted, and its results are listed: 0: uint256: product 160 and 1: uint256: sum 26. The bottom of the interface includes buttons for "Transact" and "CALLDATA".

Pure function

Code:



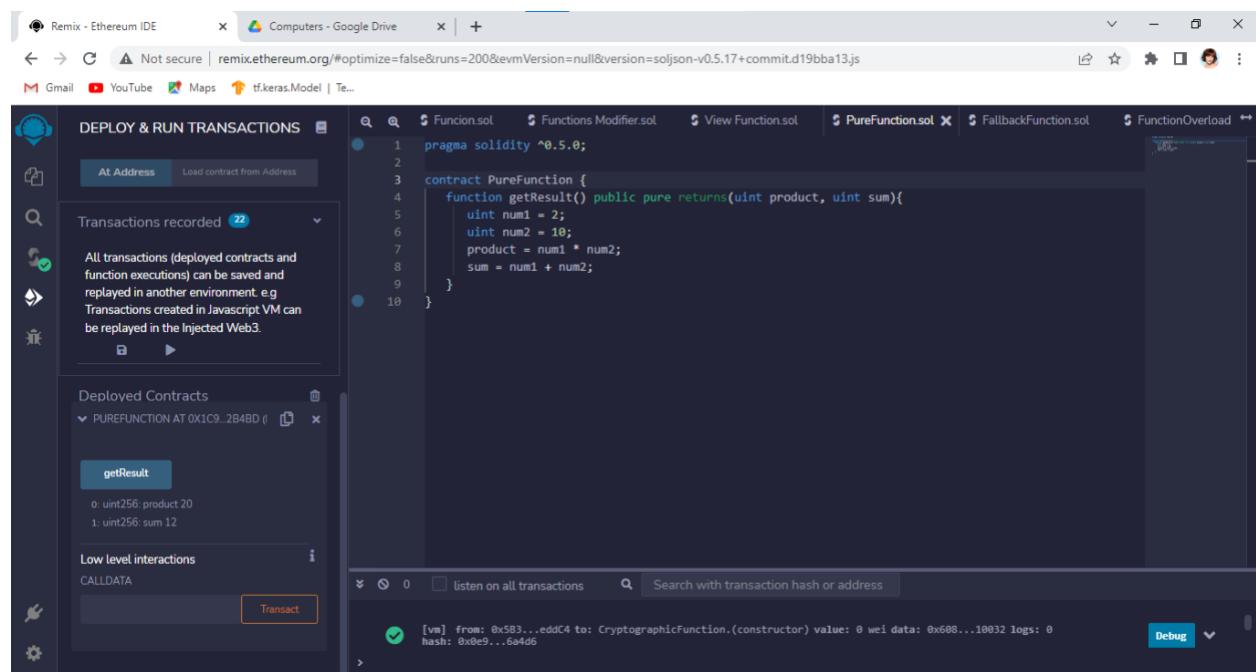
The screenshot shows the Remix Ethereum IDE interface. The left sidebar is titled "FILE EXPLORERS" and lists several Solidity files: remixDefault_1655651734405, contracts, artifacts, 1_Storage.sol, 2_Owner.sol, 3_Ballot.sol, 1A.sol, 1B.sol, while loop.sol, while loop1.sol, wl.sol, Funcion.sol, Functions Modifier.sol, View Function.sol, PureFunction.sol, FallbackFunction.sol, FunctionOverloading.sol, MathematicalFunction.sol, CryptographicFunction.sol, scripts, tests, and README.txt. The file "PureFunction.sol" is currently selected and highlighted in blue. The main editor area displays the following Solidity code:

```
pragma solidity ^0.5.0;

contract PureFunction {
    function getResult() public pure returns(uint product, uint sum){
        uint num1 = 2;
        uint num2 = 10;
        product = num1 * num2;
        sum = num1 + num2;
    }
}
```

The bottom status bar indicates "Welcome to Remix 0.24.1".

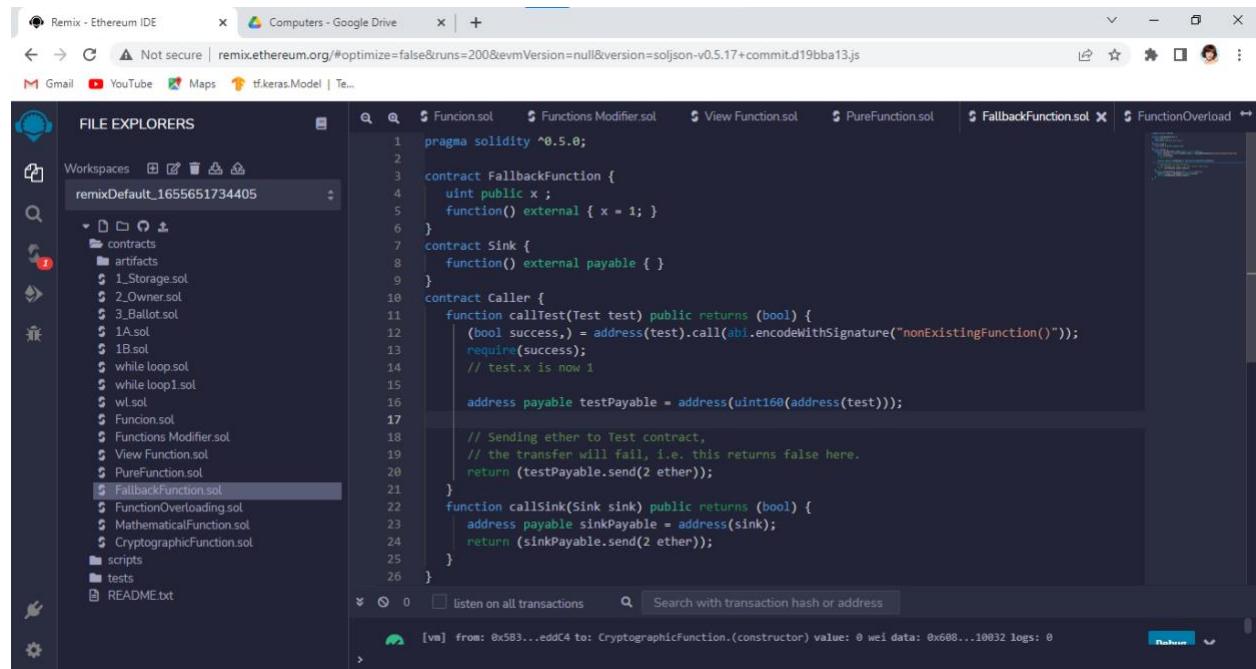
Output:



The screenshot shows the Remix Ethereum IDE interface with the "DEPLOY & RUN TRANSACTIONS" tab active. The left sidebar shows a "Transactions recorded" section with a note: "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." Below it is a "Deployed Contracts" section showing "PUREFUNCTION AT 0x1C9...2B4BD ()". The main editor area displays the same Solidity code as the previous screenshot. The bottom status bar shows a transaction log: "[vm] from: 0x583...edd4 to: CryptographicFunction.(constructor) value: 0 wei data: 0x608...10032 logs: 0 hash: 0xe9...6e4d6". A "Debug" button is visible in the bottom right corner.

Fallback function

Code:



```
pragma solidity ^0.5.0;

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

contract Sink {
    function() external payable {}
}

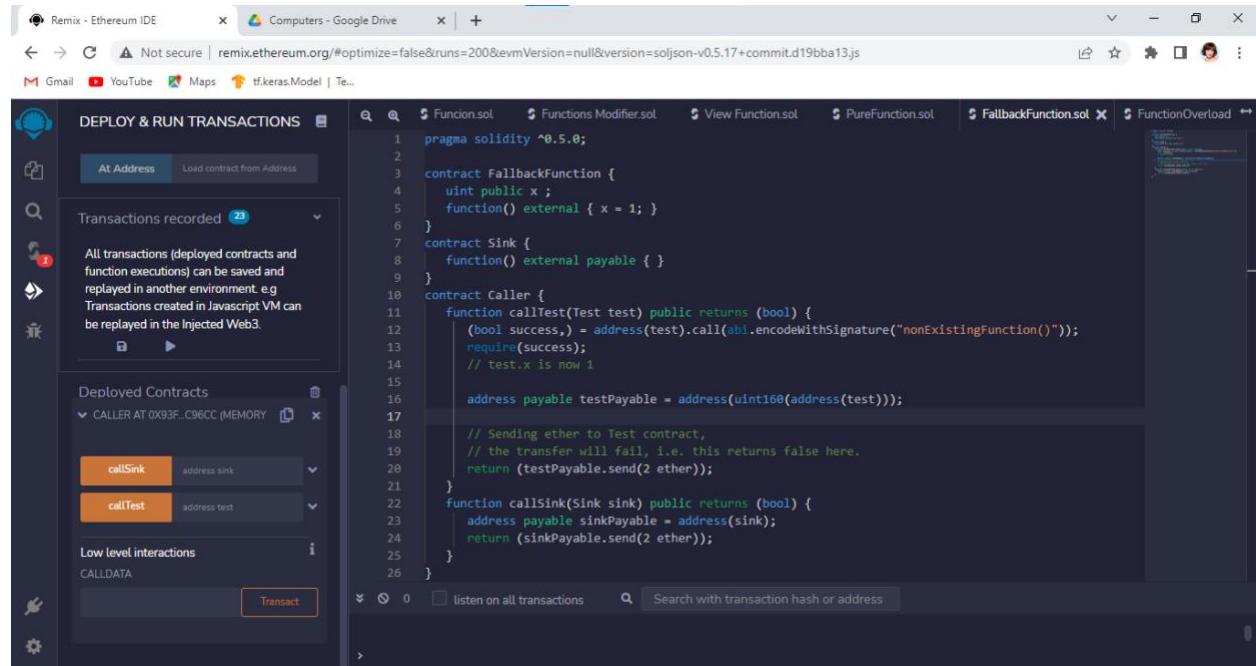
contract Caller {
    function callTest(Test test) public returns (bool) {
        (bool success,) = address(test).call(abi.encodeWithSignature("nonExistingFunction()"));
        require(success);
        // test.x is now 1
    }

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

    // Sending ether to Test contract,
    // the transfer will fail, i.e. this returns false here.
    return (testPayable.send(2 ether));
}

function callSink(Sink sink) public returns (bool) {
    address payable sinkPayable = address(sink);
    return (sinkPayable.send(2 ether));
}
```

Output:



```
pragma solidity ^0.5.0;

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

contract Sink {
    function() external payable {}
}

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

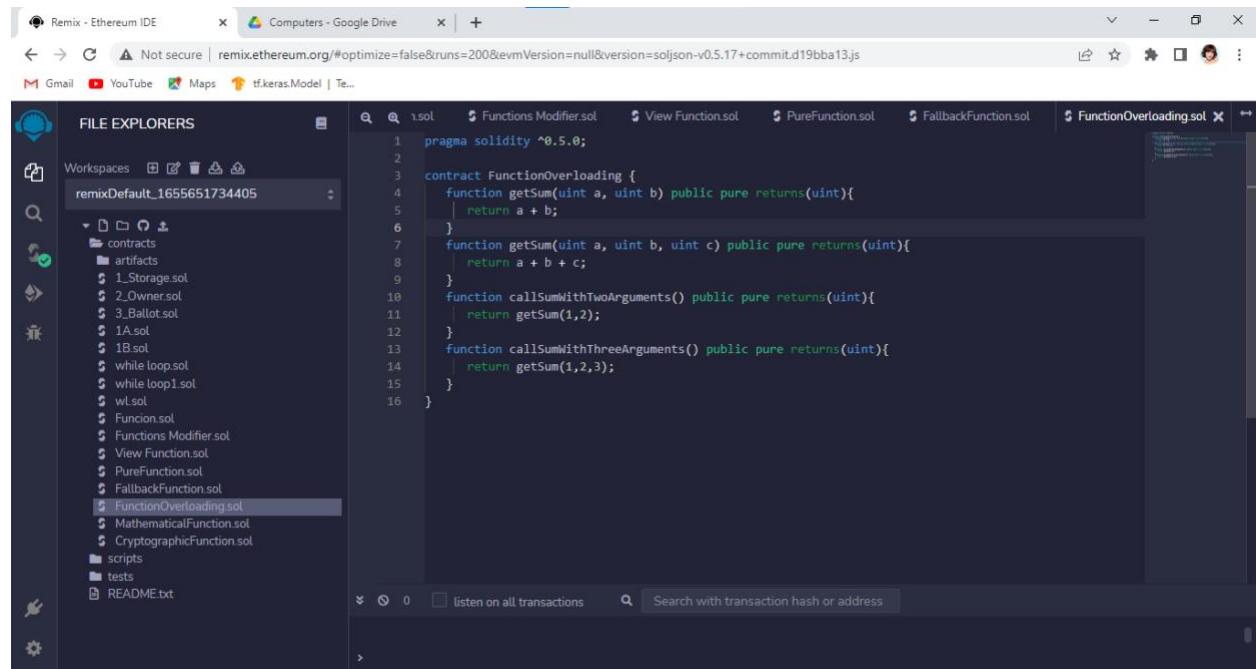
    address payable testPayable = address(uint160(address(test)));

    // Sending ether to Test contract,
    // the transfer will fail, i.e. this returns false here.
    return (testPayable.send(2 ether));
}

function callSink(Sink sink) public returns (bool) {
    address payable sinkPayable = address(sink);
    return (sinkPayable.send(2 ether));
}
```

Function Overloading

Code:

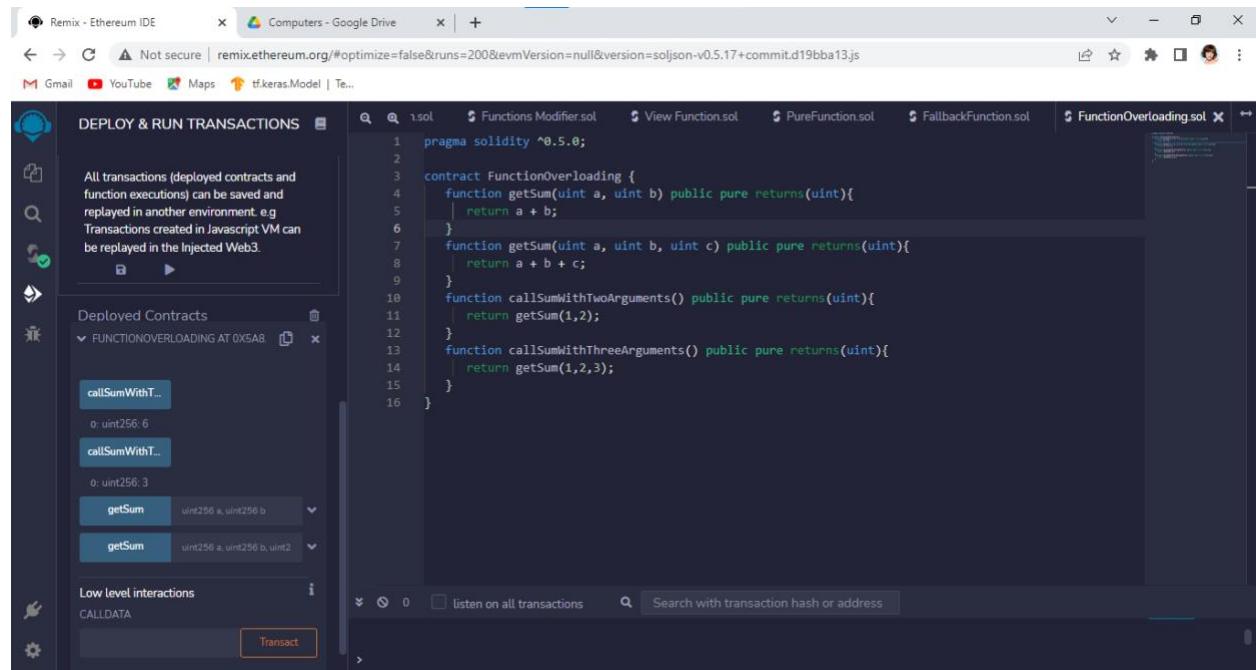


The screenshot shows the Remix Ethereum IDE interface. The left sidebar displays a file tree under 'FILE EXPLORERS' for a workspace named 'remixDefault_1655651734405'. The tree includes contracts like 1.Storage.sol, 2.Owner.sol, 3.Ballot.sol, 1A.sol, 1B.sol, while loop.sol, while loopisolt, wl.sol, Funcion.sol, Functions Modifier.sol, View Function.sol, PureFunction.sol, FallbackFunction.sol, and the current file, FunctionOverloading.sol. Other files like MathematicalFunction.sol and CryptographicFunction.sol are also listed. The right panel shows the Solidity code for the FunctionOverloading contract:

```
pragma solidity ^0.5.0;

contract FunctionOverloading {
    function getSum(uint a, uint b) public pure returns(uint){
        return a + b;
    }
    function getSum(uint a, uint b, uint c) public pure returns(uint){
        return a + b + c;
    }
    function callSumWithTwoArguments() public pure returns(uint){
        return getSum(1,2);
    }
    function callSumWithThreeArguments() public pure returns(uint){
        return getSum(1,2,3);
    }
}
```

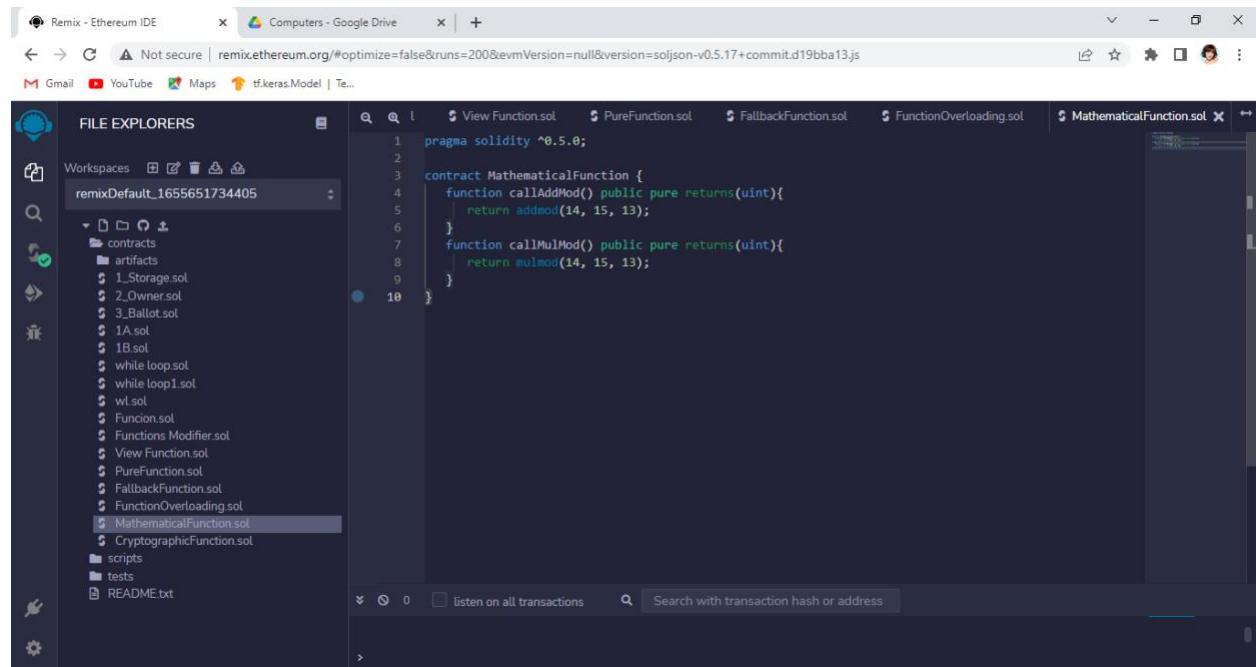
Output:



The screenshot shows the Remix Ethereum IDE interface with the 'DEPLOY & RUN TRANSACTIONS' sidebar open. It displays information about deployed contracts and function executions. Under 'Deployed Contracts', there is one entry: 'FUNCTIONOVERLOADING AT 0x5a8...'. Below it, two functions are listed: 'callSumWithT...' and 'callSumWithT...'. Each function has a dropdown menu showing its parameters: '0: uint256: 6' for the first and '0: uint256: 3' for the second. The right panel shows the same Solidity code as the previous screenshot. The bottom of the interface has a search bar and transaction monitoring controls.

Mathematical Function

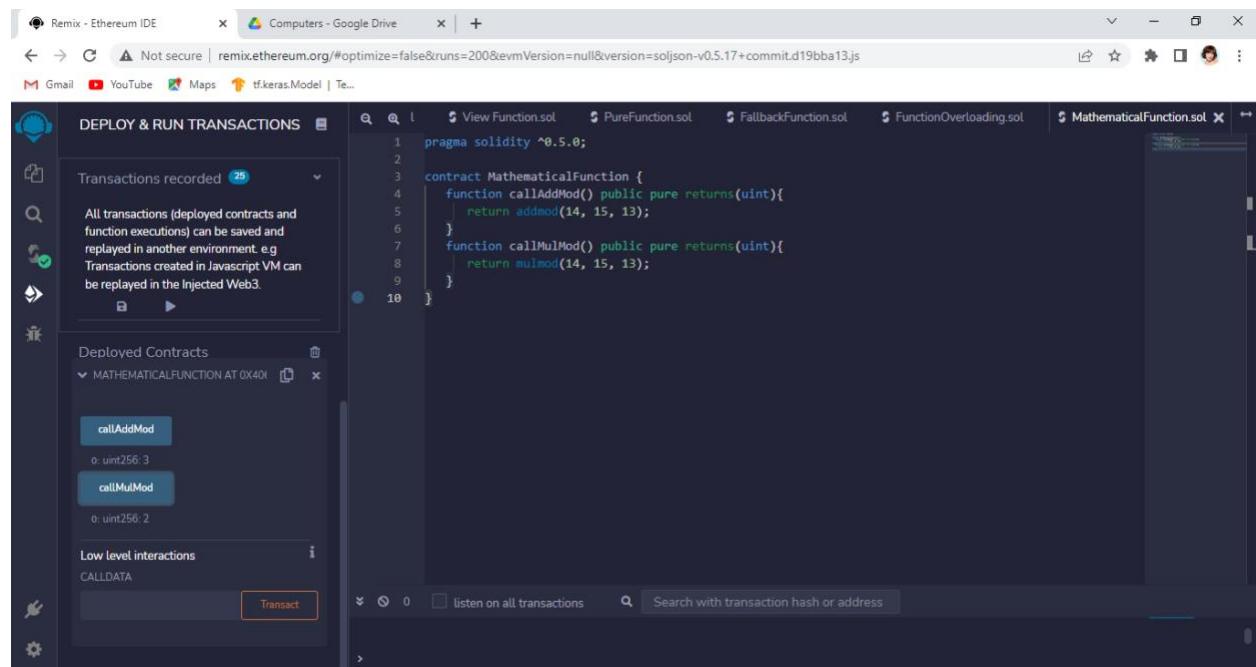
Code:



```
pragma solidity ^0.5.0;

contract MathematicalFunction {
    function callAddMod() public pure returns(uint){
        return addmod(14, 15, 13);
    }
    function callMulMod() public pure returns(uint){
        return mulmod(14, 15, 13);
    }
}
```

Output:



Deploy & Run Transactions

Transactions recorded 25

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

MATHEMATICFUNCTION AT 0x40f

callAddMod

o: uint256: 3

callMulMod

o: uint256: 2

Low level interactions

CALLDATA

Transact

```
pragma solidity ^0.5.0;

contract MathematicalFunction {
    function callAddMod() public pure returns(uint){
        return addmod(14, 15, 13);
    }
    function callMulMod() public pure returns(uint){
        return mulmod(14, 15, 13);
    }
}
```

Cryptographic function

Code:

The screenshot shows the Remix Ethereum IDE interface. The left sidebar is titled "FILE EXPLORERS" and displays a workspace named "remixDefault_1655651734405". Inside this workspace, there is a "contracts" folder containing several Solidity source files: 1.Storage.sol, 2.Owner.sol, 3.Ballot.sol, 1A.sol, 1B.sol, while loop.sol, while loop1.sol, wl.sol, Function.sol, Functions Modifier.sol, View Function.sol, PureFunction.sol, FallbackFunction.sol, FunctionOverloading.sol, MathematicalFunction.sol, and the current file, CryptographicFunction.sol, which is highlighted. Below the workspace, there are "scripts" and "tests" folders, and a "README.txt" file. The main editor area shows the Solidity code for the "CryptographicFunction" contract:

```
pragma solidity ^0.5.0;

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

Output:

The screenshot shows the Remix Ethereum IDE interface with the "DEPLOY & RUN TRANSACTIONS" sidebar open. This sidebar allows selecting a deployment environment (e.g., Javascript VM, Injected Web3) and provides options to "Deploy to Contracts" or "At Address". It also displays a list of "Transactions recorded" (26). The main editor area shows the same Solidity code for the "CryptographicFunction" contract. The bottom section of the interface shows the "Deployed Contracts" list, which includes a entry for "CRYPTOGRAPHICFUNCTION AT 0X4...". Under this entry, the "callKeccak256" function is listed with its output: "0: bytes32 result: 0x1629b9dd060bb30c7908346f6af189c16773fe148d3366701fbaa35d54f3cb". The "Low level interactions" section at the bottom contains "CALLDATA" and a "Transact" button.

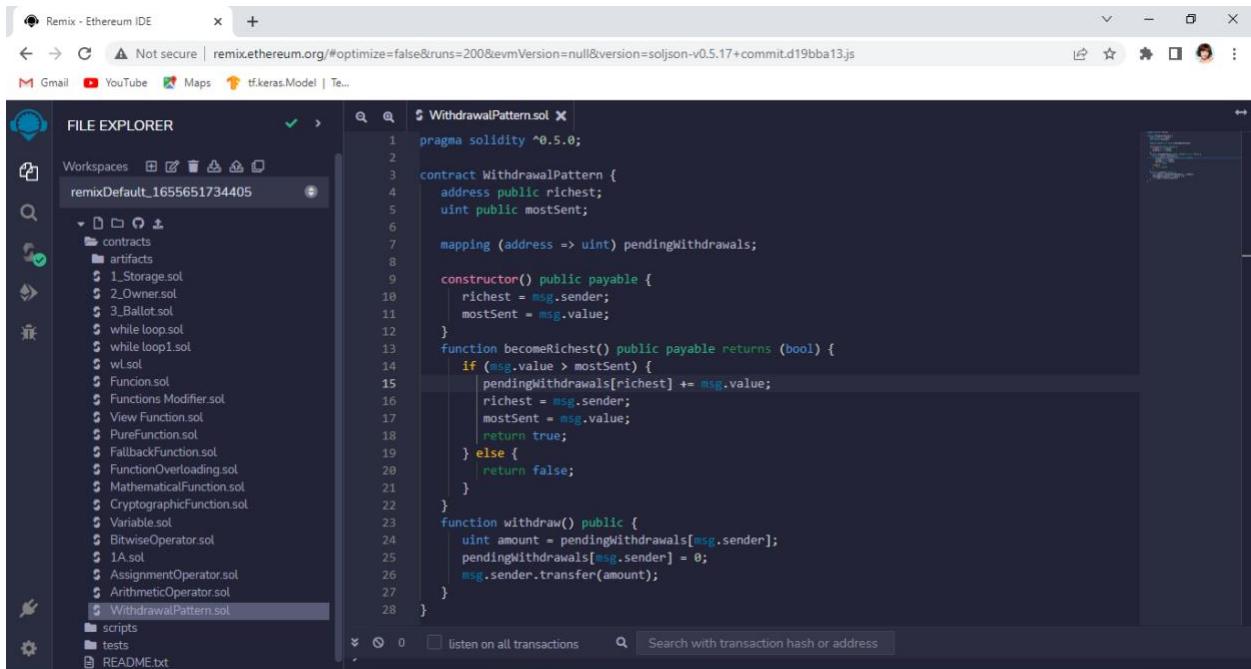
Practical No: 3

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

3a) Withdrawal Pattern, Restricted Access.

Withdrawal Pattern

Code:



The screenshot shows the Remix Ethereum IDE interface. The top bar displays the title "Remix - Ethereum IDE" and the URL "remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.5.17+commit.d19bba13.js". The left sidebar is the "FILE EXPLORER" showing a workspace named "remixDefault_1655651734405" containing several Solidity contracts like "1_Storage.sol", "2_Owner.sol", "3_Ballot.sol", "white_loop.sol", "white_loop1.sol", "wl.sol", "Funcion.sol", "Functions Modifier.sol", "View Function.sol", "PureFunction.sol", "FallbackFunction.sol", "FunctionOverloading.sol", "MathematicalFunction.sol", "CryptographicFunction.sol", "Variable.sol", "BitwiseOperator.sol", "1A.sol", "AssignmentOperator.sol", "ArithmeticOperator.sol", and "WithdrawalPattern.sol". The main central area is a code editor with the file "WithdrawalPattern.sol" open, displaying the following Solidity code:

```
pragma solidity ^0.5.0;

contract WithdrawalPattern {
    address public richest;
    uint public mostSent;

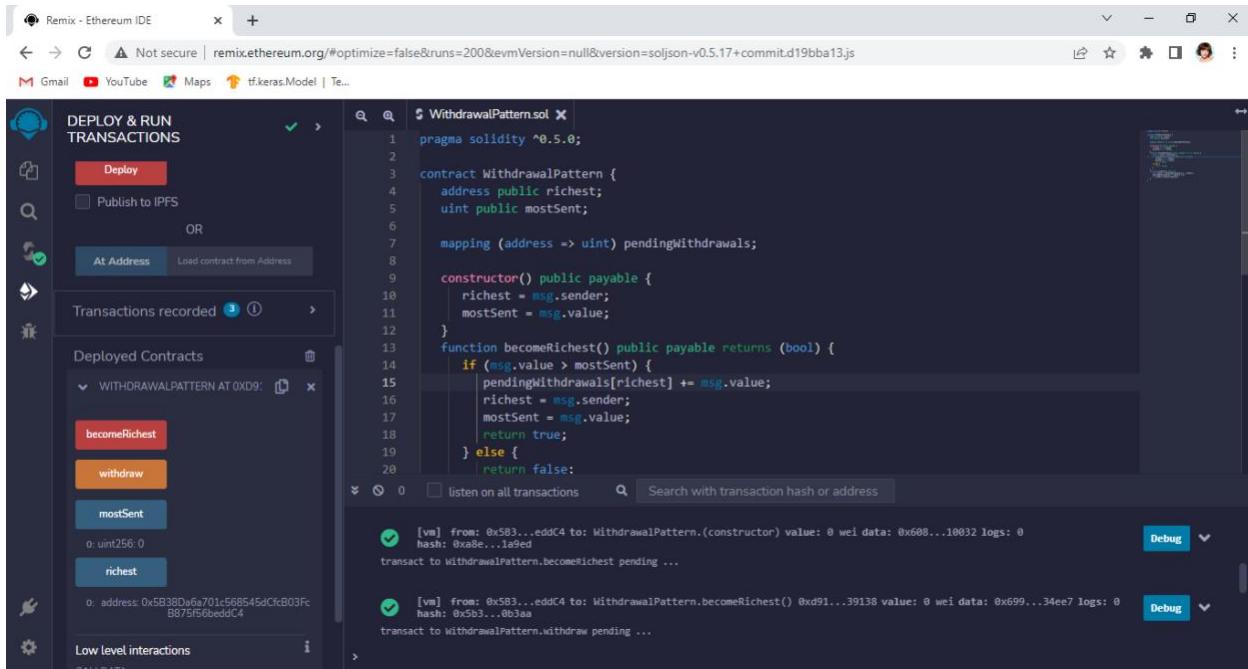
    mapping (address => uint) pendingWithdrawals;

    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;
        }
    }

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

Output:



```
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;
    }

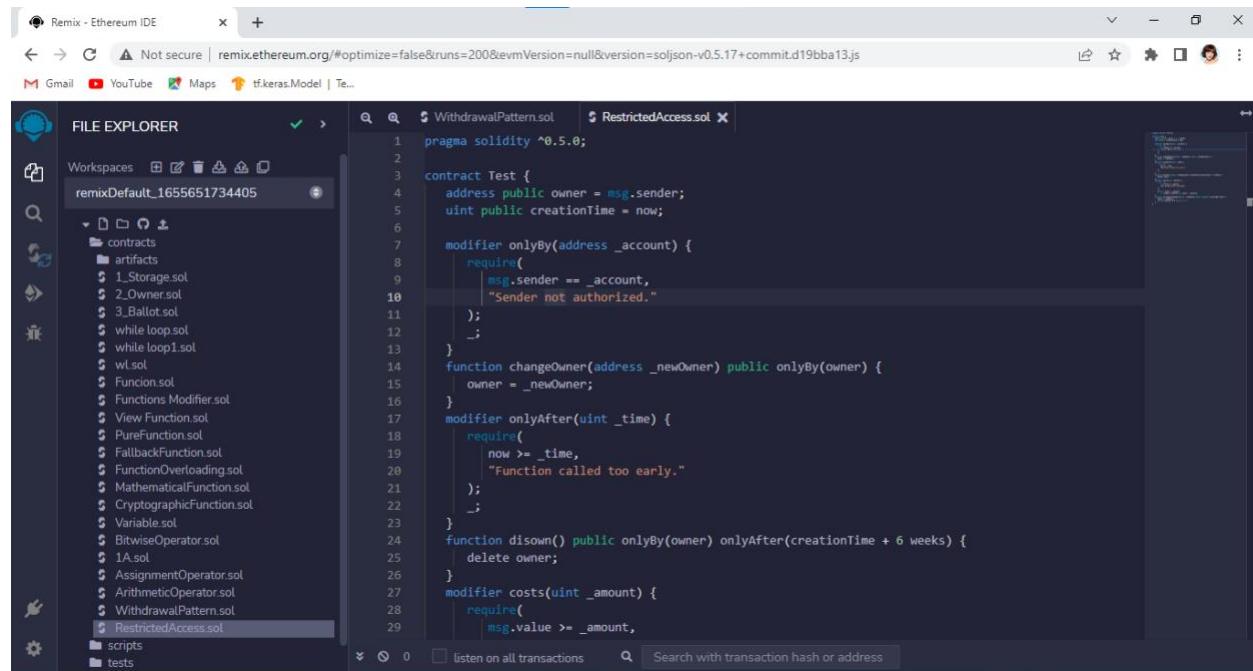
    function becomeRichest() public payable returns (bool) {
        if (msg.value > mostSent) {
            pendingWithdrawals[richest] += msg.value;
            richest = msg.sender;
            mostSent = msg.value;
            return true;
        } else {
            return false;
        }
    }

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

The screenshot shows the Ethereum IDE (Remix) interface. On the left, there's a sidebar with options like "DEPLOY & RUN TRANSACTIONS" (with "Deploy" and "Publish to IPFS" buttons), "Transactions recorded" (listing "becomeRichest", "withdraw", "mostSent", and "richest" functions), and "Low level interactions" (showing address 0xd9...875f56beddC4). The main area displays the Solidity code for the WithdrawalPattern contract. At the bottom, there are two transaction logs: one for the constructor call and another for a call to the becomeRichest function.

Restricted Access

Code:



The screenshot shows the Remix Ethereum IDE interface. The left sidebar displays a file tree under 'remixDefault_1655651734405' workspace, with 'contracts' expanded to show files like 1_Storage.sol, 2_Owner.sol, 3_Ballot.sol, while loop.sol, white loop1.sol, wl.sol, Funcion.sol, Functions Modifier.sol, View Function.sol, PureFunction.sol, FallbackFunction.sol, FunctionOverloading.sol, MathematicalFunction.sol, CryptographicFunction.sol, Variable.sol, BitwiseOperator.sol, I1.sol, AssignmentOperator.sol, ArithmeticOperator.sol, WithdrawalPattern.sol, and RestrictedAccess.sol. The right panel shows the Solidity code for 'RestrictedAccess.sol'. The code defines a contract 'Test' with several modifiers and functions. It includes checks for sender authorization, time constraints, and ether value.

```
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."
        );
    }

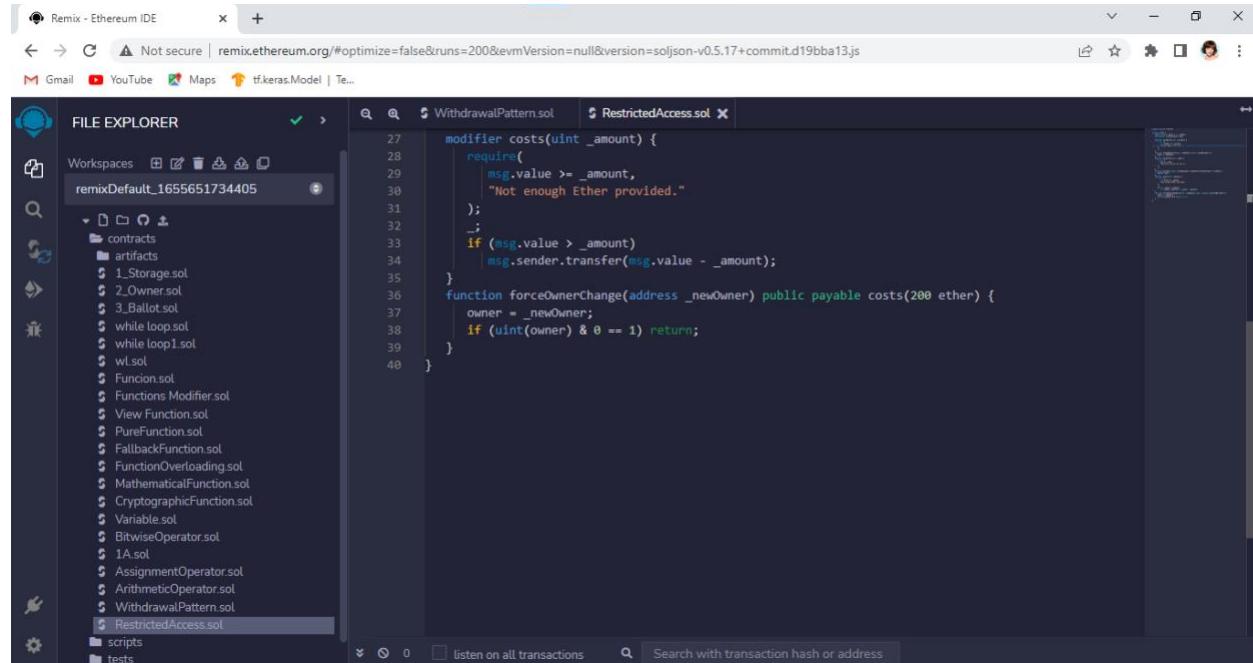
    function changeOwner(address _newOwner) public onlyBy(owner) {
        owner = _newOwner;
    }

    modifier onlyAfter(uint _time) {
        require(
            now >= _time,
            "Function called too early."
        );
    }

    function disown() public onlyBy(owner) onlyAfter(creationTime + 6 weeks) {
        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;
    }
}
```

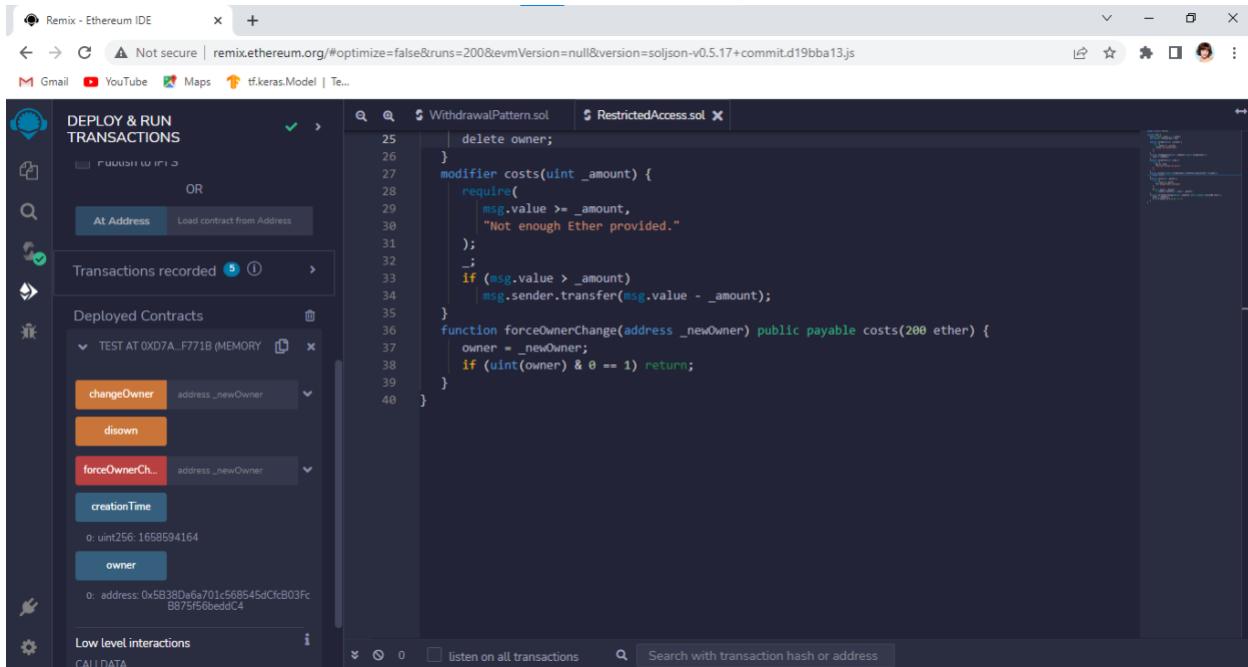


This screenshot shows the continuation of the 'RestrictedAccess.sol' code from the previous screenshot. The code adds a new modifier 'costs' and a function 'forceOwnerChange'. The 'costs' modifier ensures that the caller provides enough Ether. The 'forceOwnerChange' function changes the owner to a specified address and returns if the owner's ID is odd.

```
modifier costs(uint _amount) {
    require(
        msg.value >= _amount,
        "Not enough Ether provided."
    );
    if (msg.value > _amount)
        msg.sender.transfer(msg.value - _amount);
}

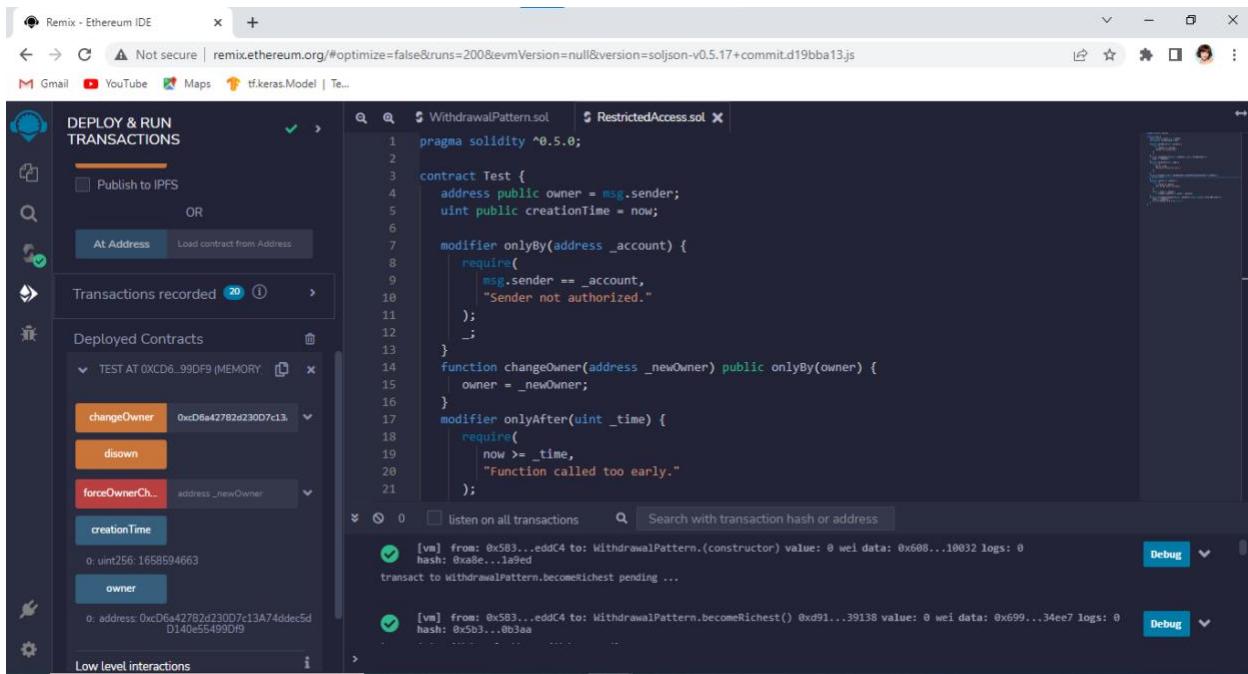
function forceOwnerChange(address _newOwner) public payable costs(200 ether) {
    owner = _newOwner;
    if (uint(owner) & 0 == 1) return;
}
```

Output:



The screenshot shows the Remix Ethereum IDE interface. On the left, the sidebar displays the "DEPLOY & RUN TRANSACTIONS" section with options for "Publish to IPFS" or "At Address". Under "Transactions recorded", there are 5 entries. In the "Deployed Contracts" section, a contract named "TEST AT 0xD7A...F71B (MEMORY)" is listed. The code editor on the right contains the Solidity code for the `RestrictedAccess.sol` contract. The code includes functions for changing ownership and a modifier for restricted access.

```
25 | delete owner;
26 |
27 | modifier costs(uint _amount) {
28 |     require(
29 |         msg.value >= _amount,
30 |         "Not enough Ether provided."
31 |     );
32 |     if (msg.value > _amount)
33 |         msg.sender.transfer(msg.value - _amount);
34 |
35 | }
36 | function forceOwnerChange(address _newOwner) public payable costs(200 ether) {
37 |     owner = _newOwner;
38 |     if (uint(owner) & 0 == 1) return;
39 | }
```



This screenshot shows the same Remix IDE interface after a deployment. The "Deployed Contracts" section now lists a new contract named "TEST AT 0xCD6...99DF9 (MEMORY)". The code editor still displays the `RestrictedAccess.sol` contract, which has been updated to include a `pragma solidity ^0.5.0;` header. The deployment transaction details are visible in the bottom log pane.

```
1  pragma solidity ^0.5.0;
2
3  contract Test {
4      address public owner = msg.sender;
5      uint public creationTime = now;
6
7      modifier onlyBy(address _account) {
8          require(
9              msg.sender == _account,
10             "Sender not authorized."
11         );
12     }
13 }
14 function changeOwner(address _newOwner) public onlyBy(owner) {
15     owner = _newOwner;
16 }
17 modifier onlyAfter(uint _time) {
18     require(
19         now >= _time,
20         "Function called too early."
21     );
22 }
```

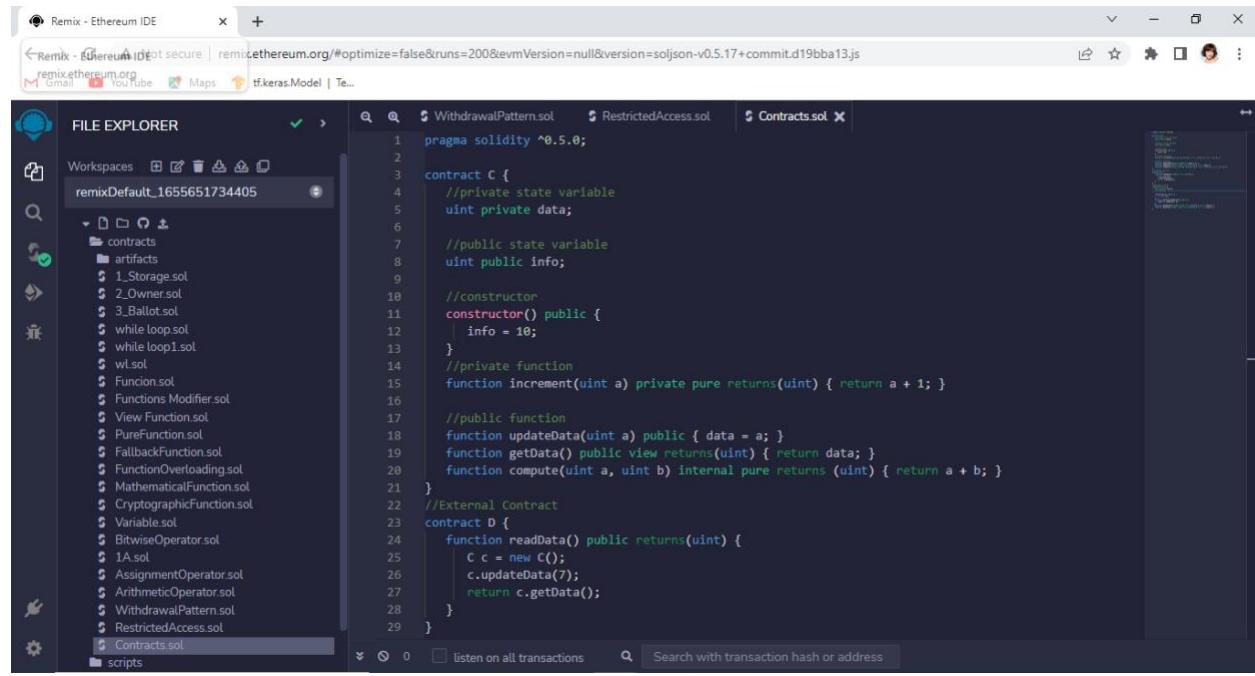
Logs:

- [vm] from: 0x583...eddC4 to: WithdrawalPattern.(constructor) value: 0 wei data: 0x608...10032 logs: 0
- [vm] from: 0x583...eddC4 to: WithdrawalPattern.becomeRichest() 0xd91...39138 value: 0 wei data: 0x699...34ee7 logs: 0

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

Contracts

Code:



The screenshot shows the Remix Ethereum IDE interface. The top bar displays the title "Remix - Ethereum IDE" and the URL "remix.ethereum.org/#optimize=false&runs=200&evmVersion=null&version=soljson-v0.5.17+commit.d19bba13.js". The left sidebar is the "FILE EXPLORER" showing a workspace named "remixDefault_1655651734405" containing several contracts like 1_Storage.sol, 2_Owner.sol, 3_Ballot.sol, etc., and a file named "Contracts.sol" which is currently selected. The main editor area contains the following Solidity code:

```
pragma solidity ^0.5.0;

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

    //public state variable
    uint public info;

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

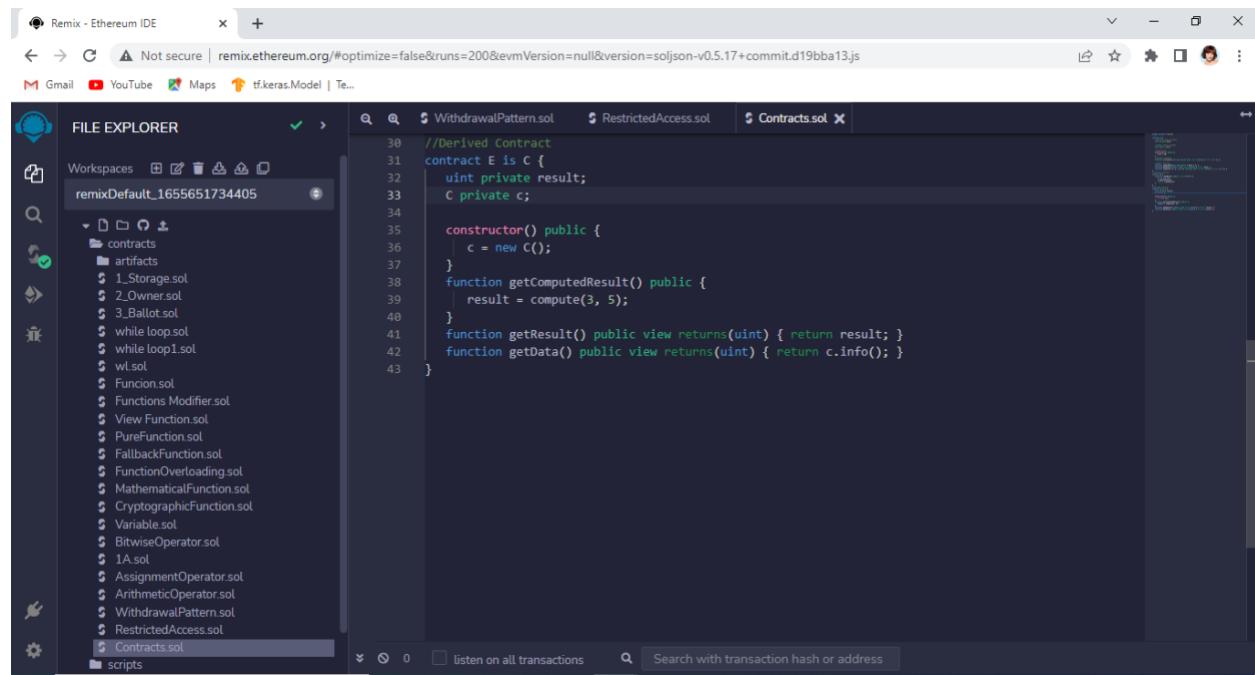
    //private function
    function increment(uint a) private pure returns(uint) { return a + 1; }

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

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

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

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



The screenshot shows the Remix Ethereum IDE interface, similar to the previous one but with different code in the editor. The "Contracts.sol" file now includes inheritance from contract C. The code is as follows:

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

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

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

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

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

Output:

The screenshot shows the Remix Ethereum IDE interface. On the left, the 'DEPLOY & RUN TRANSACTIONS' sidebar is open, showing a list of deployed contracts under 'Deployed Contracts'. One contract, 'C AT 0x5FD...9D88D (MEMORY)', is selected. The 'updateData' function is highlighted, with its parameters: 'a: uint256 a' and 'b: uint256 b'. Below the function list, there's a section for 'Low level interactions' with 'CALLDATA' and a 'Transact' button. The main right panel displays the Solidity code for 'WithdrawalPattern.sol' and 'Contracts.sol'. The transaction history pane at the bottom shows three transactions: a constructor call to 'C' with value 0x0, a call to 'C.updateData' with value 0x0, and a call to 'C.getData'.

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

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

This screenshot is nearly identical to the one above, showing the same Remix interface, deployed contract, and Solidity code. The transaction history pane at the bottom shows three transactions: a constructor call to 'C' with value 0x0, a call to 'C.updateData' with value 0x5FD...9D88D, and a call to 'C.getData'.

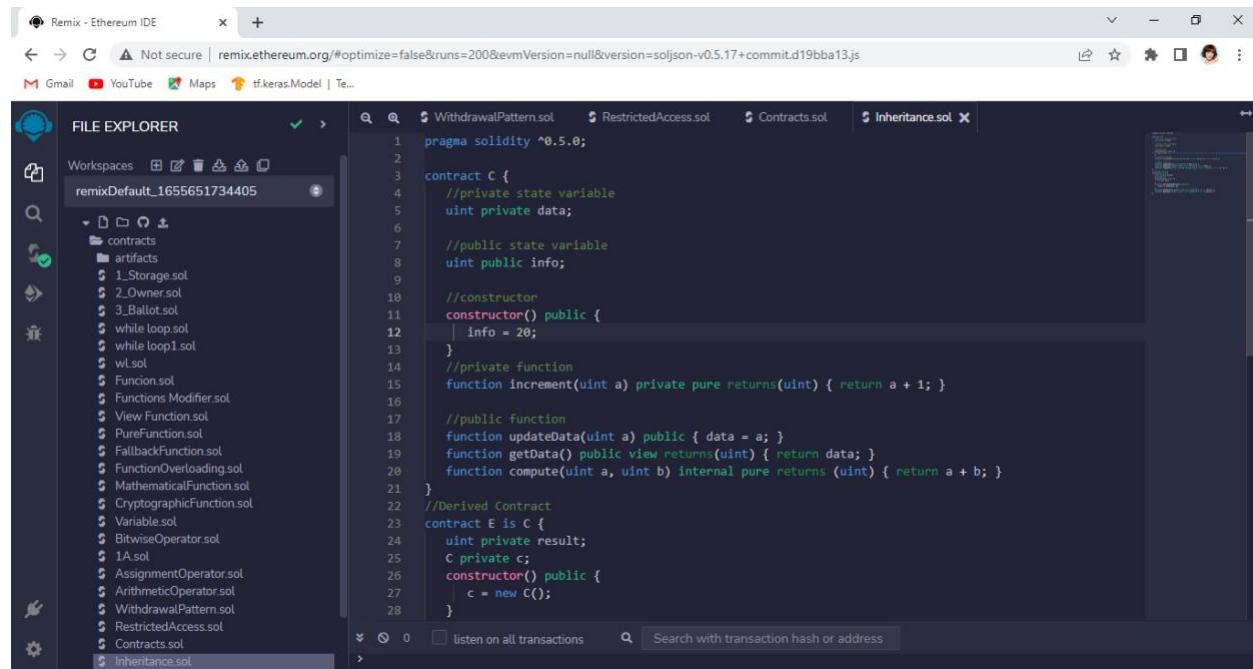
```
[vm] from: 0x583...edd4 to: C.updateData(uint256) 0x5FD...9D88D value: 0 wei data: 0x09e...00007 logs: 0 hash: 0x59c...01391
call to C.updateData

[call] [call] from: 0x58380a6a701c568545dCfcB03FcB875f56bebddC4 to: C.getData() data: 0x3bc...5de30
call to C.getData

[call] [call] from: 0x58380a6a701c568545dCfcB03FcB875f56bebddC4 to: C.info() data: 0x370...158ea
call to C.info
```

Inheritance

Code:



The screenshot shows the Remix Ethereum IDE interface. The left sidebar displays a file tree under 'WORKSPACES' titled 'remixDefault_1655651734405'. The tree includes a 'contracts' folder containing various Solidity files such as 1_Storage.sol, 2_Owner.sol, 3_Ballot.sol, while_loop.sol, while_loop1.sol, wl.sol, Funcion.sol, Functions.Modifier.sol, View.Function.sol, PureFunction.sol, FallbackFunction.sol, FunctionOverloading.sol, MathematicalFunction.sol, CryptographicFunction.sol, Variable.sol, BitwiseOperator.sol, 1A.sol, AssignmentOperator.sol, ArithmeticOperator.sol, WithdrawalPattern.sol, RestrictedAccess.sol, Contracts.sol, and Inheritance.sol. The right panel shows the Solidity code for '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; }

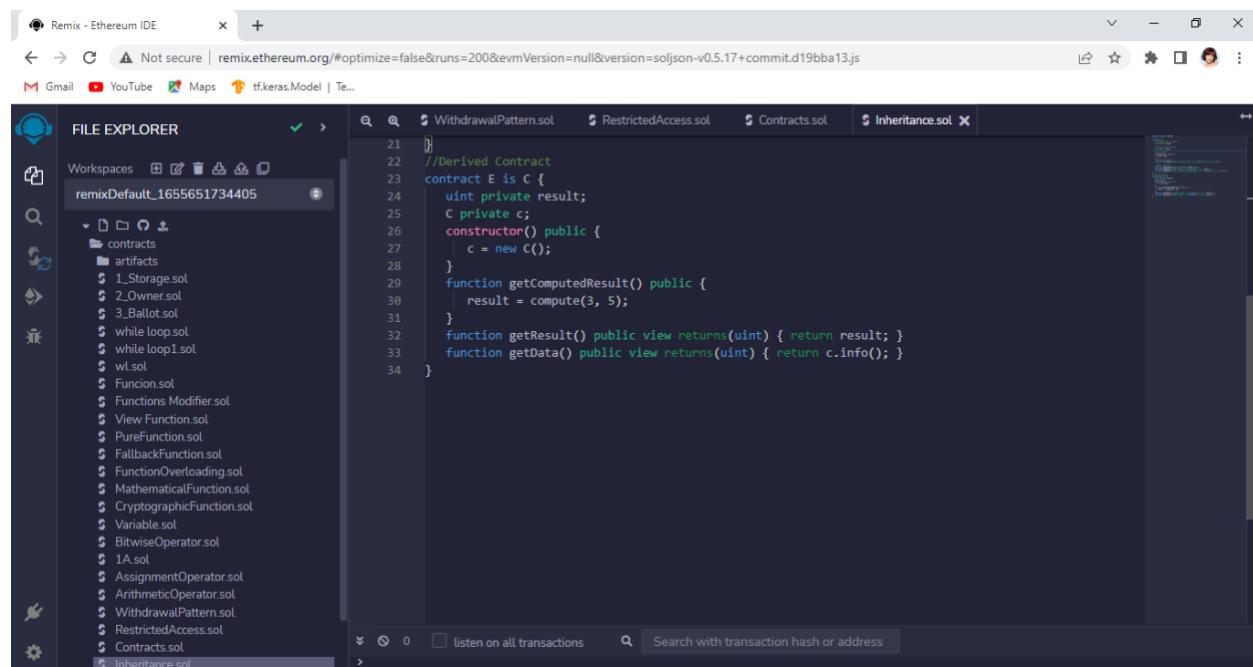
    function compute(uint a, uint b) internal pure returns (uint) { return a + b; }
}

//Derived Contract
contract E is C {
    uint private result;
    C private c;
    constructor() public {
        c = new C();
    }

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

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

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



This screenshot shows the continuation of the Solidity code from the previous screen. The code defines a derived contract 'E' that inherits from 'C'. It includes methods to compute values and retrieve the computed result and data from the base contract 'C'.

```
}

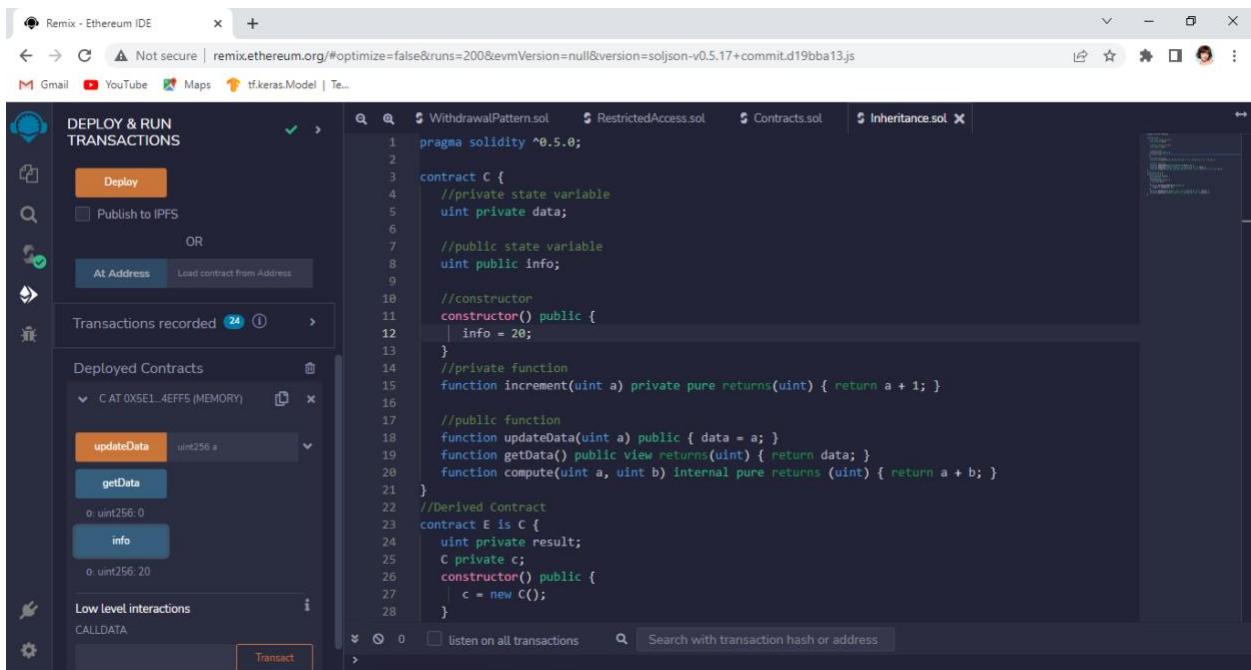
//Derived Contract
contract E is C {
    uint private result;
    C private c;
    constructor() public {
        c = new C();
    }

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

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

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

Output:



```
pragma solidity ^0.5.0;

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

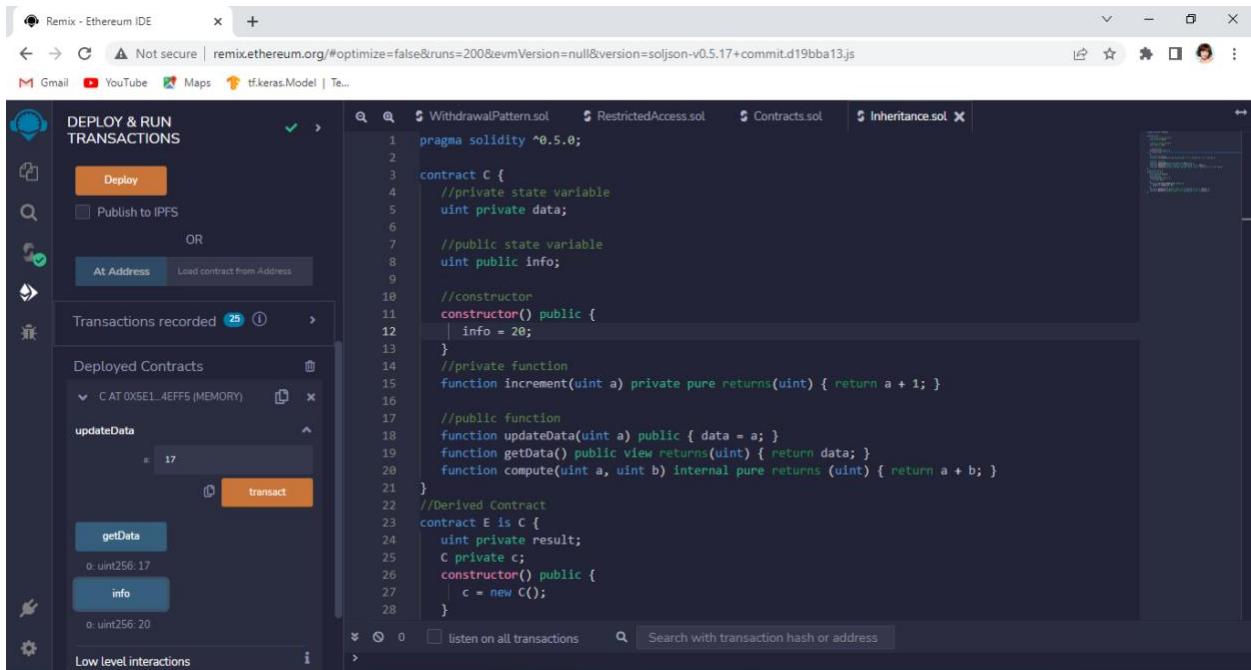
    //public state variable
    uint public info;

    //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();
    }
}
```



```
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();
    }
}
```

Constructors

Code:

The screenshot shows the Remix Ethereum IDE interface. On the left, the FILE EXPLORER panel lists various Solidity contracts and scripts. In the center, the code editor displays the following Solidity code:

```
pragma solidity ^0.5.0;

contract constructorExample {
    string str;
    constructor() public{
        str="This is a example of Constructor";
    }
    function getValue()
    public view returns(
        string memory)
    {
        return str;
    }
}
```

Output:

The screenshot shows the Remix Ethereum IDE interface with the DEPLOY & RUN TRANSACTIONS tab selected. The CONTRACT dropdown shows "constructorExample - contracts/Cons". The Deploy button is highlighted. The code editor on the right remains the same as in the previous screenshot. The bottom pane shows deployment logs and transaction history.

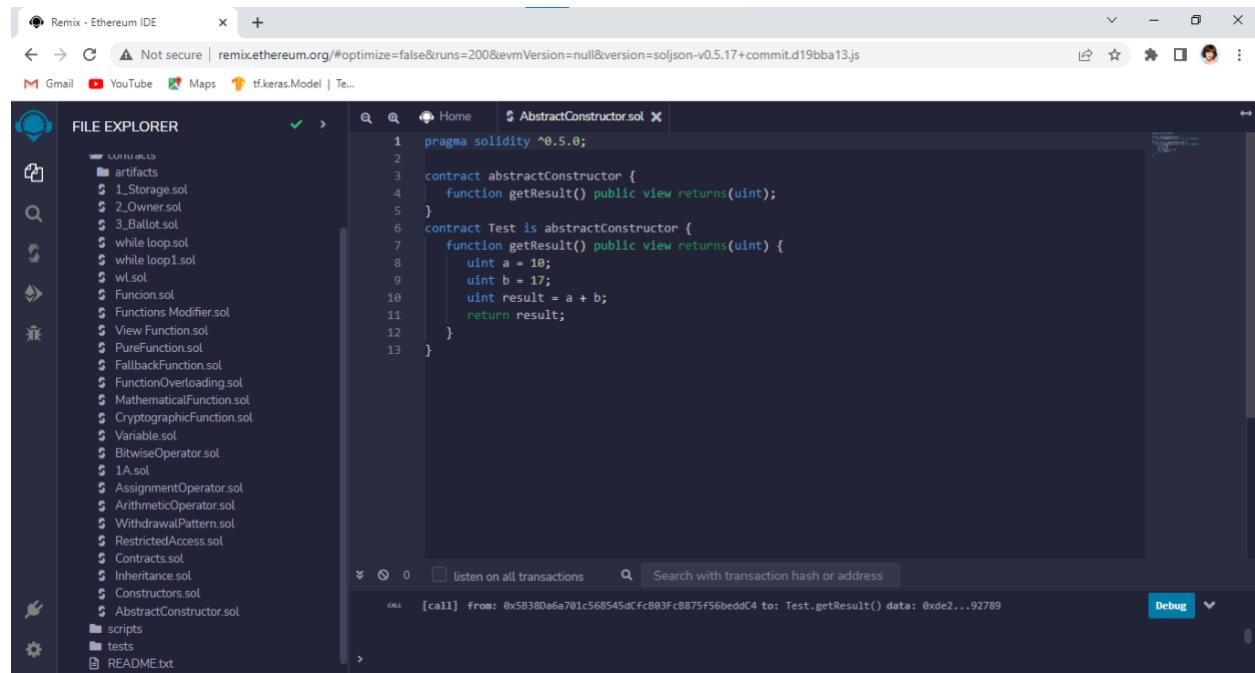
Logs:

```
creation of constructorExample pending...
[vm] from: 0x583...edd4 to: constructorExample.(constructor) value: 0 wei data: 0x608...10032 logs: 0
hash: 0xe3c...1809c
call to constructorExample.getValue()

call [call] from: 0x58380a6a701c568545dCfcB03FcB875f56beddC4 to: constructorExample.getValue() data: 0x209...65255
```

Abstract Contracts

Code:

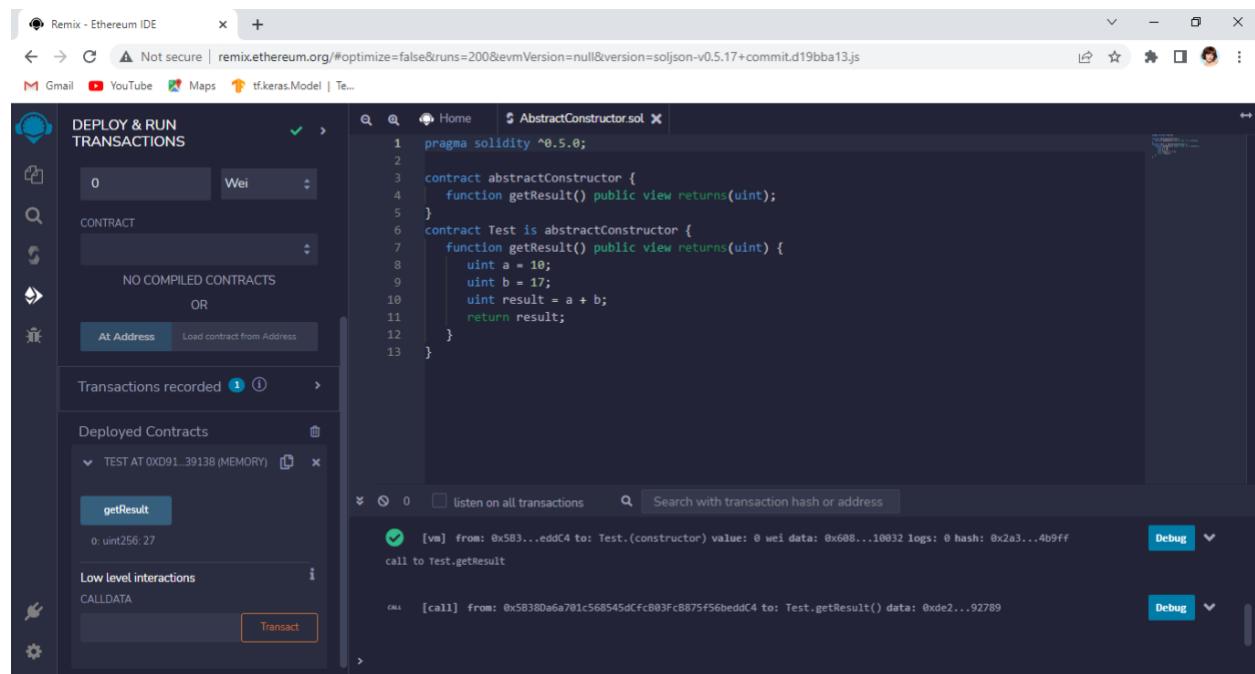


The screenshot shows the Remix Ethereum IDE interface. The left sidebar is titled "FILE EXPLORER" and lists several Solidity files under the "artifacts" folder, including 1_Storage.sol, 2_Owner.sol, 3_Ballot.sol, while loop.sol, while loop1.sol, wl.sol, Funcion.sol, Functions Modifier.sol, View Function.sol, PureFunction.sol, FallbackFunction.sol, FunctionOverloading.sol, MathematicalFunction.sol, CryptographicFunction.sol, Variable.sol, BitwiseOperator.sol, 1A.sol, AssignmentOperator.sol, ArithmeticOperator.sol, WithdrawalPattern.sol, RestrictedAccess.sol, Contracts.sol, Inheritance.sol, Constructors.sol, AbstractConstructor.sol, scripts, tests, and README.txt. The main editor window displays the "AbstractConstructor.sol" file with the following code:

```
1 pragma solidity ^0.5.0;
2
3 contract abstractConstructor {
4     function getResult() public view returns(uint);
5 }
6 contract Test is abstractConstructor {
7     function getResult() public view returns(uint) {
8         uint a = 10;
9         uint b = 17;
10        uint result = a + b;
11        return result;
12    }
13 }
```

The bottom right corner of the editor has a "Debug" button. Below the editor, there is a transaction history section with a single entry: "[call] from: 0x5830a6a701c568545dCfc803Fc8875f56beddC4 to: Test.getResult() data: 0xde2...92789".

Output:

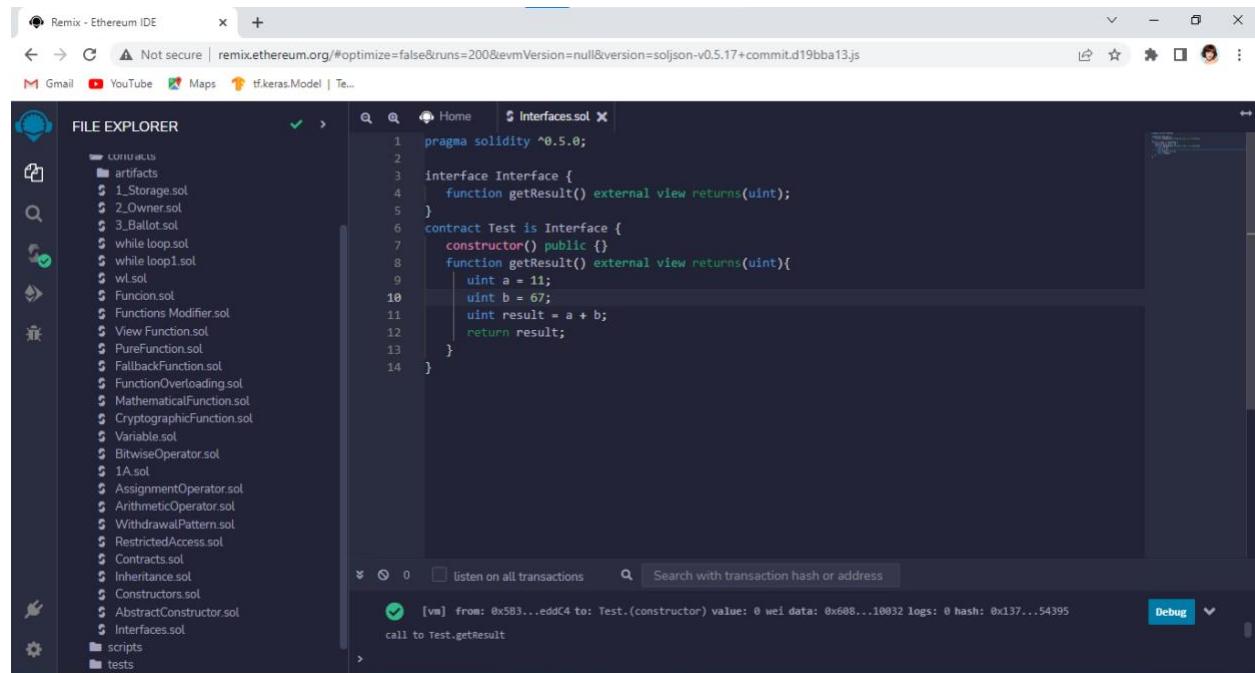


The screenshot shows the Remix Ethereum IDE interface with the "DEPLOY & RUN TRANSACTIONS" sidebar active. The sidebar includes fields for "Wei" and "At Address". It also shows a "CONTRACT" dropdown set to "NO COMPILED CONTRACTS" and a "Transactions recorded" section. The main editor window is identical to the one in the previous screenshot, displaying the "AbstractConstructor.sol" code.

The bottom right corner of the editor has a "Debug" button. Below the editor, there are two transaction entries. The first is a constructor call: "[vm] from: 0x583...eddC4 to: Test.(constructor) value: 0 wei data: 0x608...10032 logs: 0 hash: 0x2a3...4b9ff". The second is a function call: "[call] from: 0x5830a6a701c568545dCfc803Fc8875f56beddC4 to: Test.getResult() data: 0xde2...92789".

Interfaces

Code:



The screenshot shows the Remix Ethereum IDE interface. On the left, the FILE EXPLORER sidebar lists various Solidity files and contracts. The main code editor window displays the following Solidity code:

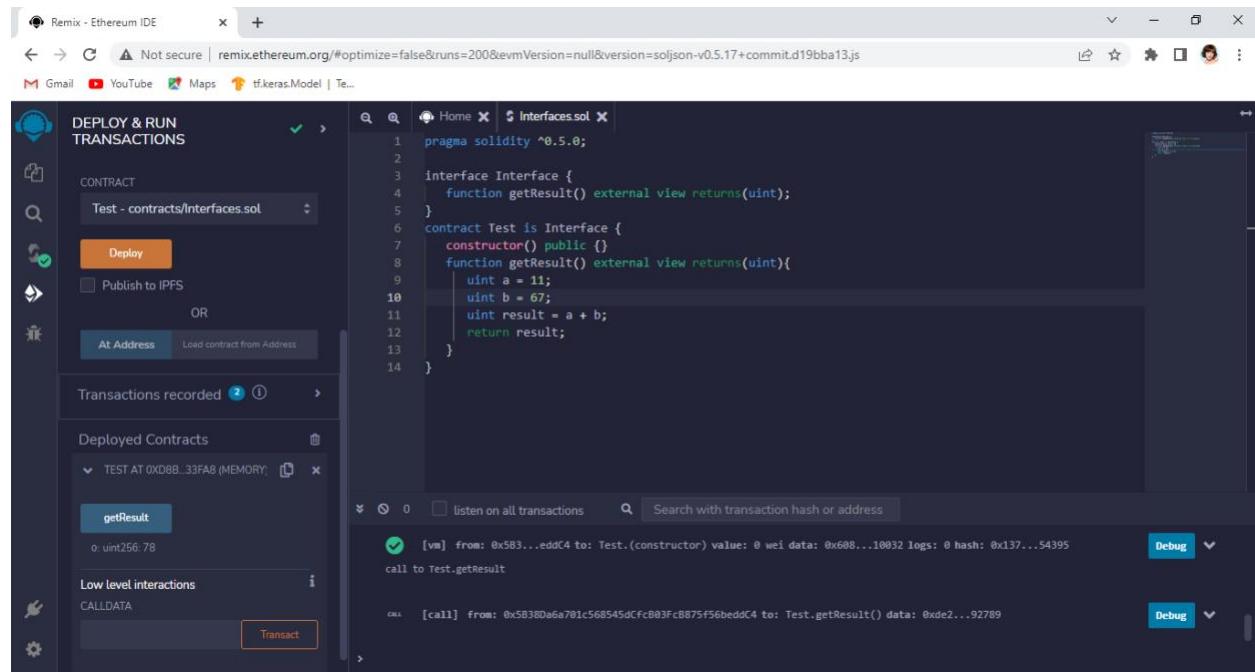
```
pragma solidity ^0.5.0;

interface Interface {
    function getResult() external view returns(uint);
}

contract Test is Interface {
    constructor() public {}
    function getResult() external view returns(uint){
        uint a = 11;
        uint b = 67;
        uint result = a + b;
        return result;
    }
}
```

The bottom status bar shows transaction logs and a debug button.

Output:



The screenshot shows the Remix Ethereum IDE interface with the DEPLOY & RUN TRANSACTIONS sidebar active. It displays the deployed contract details and transaction history. The code editor window shows the same Solidity code as before. The bottom status bar shows transaction logs and a debug button.

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

Libraries

Code:

The screenshot shows the Remix Ethereum IDE interface. On the left, the FILE EXPLORER sidebar lists several Solidity files: 2_Owners.sol, 3_Ballot.sol, whileloop.sol, whileloop1.sol, wl.sol, Funcion.sol, Functions.Modifier.sol, View.Function.sol, PureFunction.sol, FallbackFunction.sol, FunctionOverloading.sol, MathematicalFunction.sol, CryptographicFunction.sol, Variable.sol, BitwiseOperator.sol, 1A.sol, AssignmentOperator.sol, ArithmeticOperator.sol, WithdrawalPattern.sol, RestrictedAccess.sol, Contracts.sol, Inheritance.sol, Constructors.sol, AbstractConstructor.sol, Interfaces.sol, Libraries.sol, scripts, tests, and README.txt. The Libraries.sol file is currently selected and open in the main code editor window. The code implements a library named Search with a function to find the index of a value in an array, and a contract named Libraries with a constructor that initializes an array of uints from 1 to 5, and a function to check if a value is present in the array using the library function.

```
pragma solidity ^0.5.0;

library Search {
    function indexOf(uint[] storage self, uint value) public view returns (uint) {
        for (uint i = 0; i < self.length; i++) if (self[i] == value) return i;
        return uint(-1);
    }
}

contract Libraries {
    uint[] data;
    constructor() public {
        data.push(1);
        data.push(2);
        data.push(3);
        data.push(4);
        data.push(5);
    }
    function isValuePresent() external view returns(uint){
        uint value = 4;
        //search if value is present in the array using Library function
        uint index = Search.indexOf(data, value);
        return index;
    }
}
```

Output:

The screenshot shows the Remix Ethereum IDE interface with the DEPLOY & RUN TRANSACTIONS tab active. The CONTRACT section shows the Libraries - contracts/Libraries.sol file selected. Below it, there are buttons for Deploy, Publish to IPFS, and At Address. The At Address button is highlighted. The Deployed Contracts section shows a deployed contract named LIBRARIES AT 0X358...D5EE3 (METHANE). Under this contract, the isValuePresent function is listed with its output: o: uint256: 3. The code editor window on the right remains the same as in the previous screenshot, displaying the Solidity code for the Libraries contract.

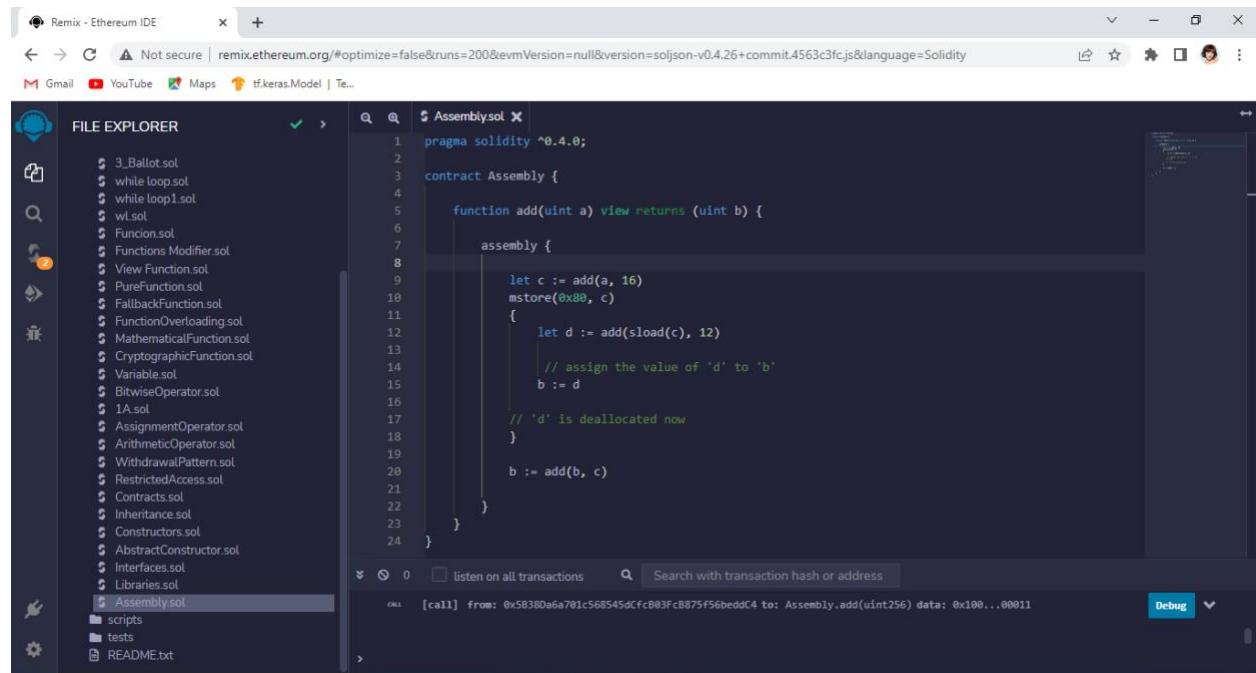
```
pragma solidity ^0.5.0;

library Search {
    function indexOf(uint[] storage self, uint value) public view returns (uint) {
        for (uint i = 0; i < self.length; i++) if (self[i] == value) return i;
        return uint(-1);
    }
}

contract Libraries {
    uint[] data;
    constructor() public {
        data.push(1);
        data.push(2);
        data.push(3);
        data.push(4);
        data.push(5);
    }
    function isValuePresent() external view returns(uint){
        uint value = 4;
        //search if value is present in the array using Library function
        uint index = Search.indexOf(data, value);
        return index;
    }
}
```

Assembly

Code:

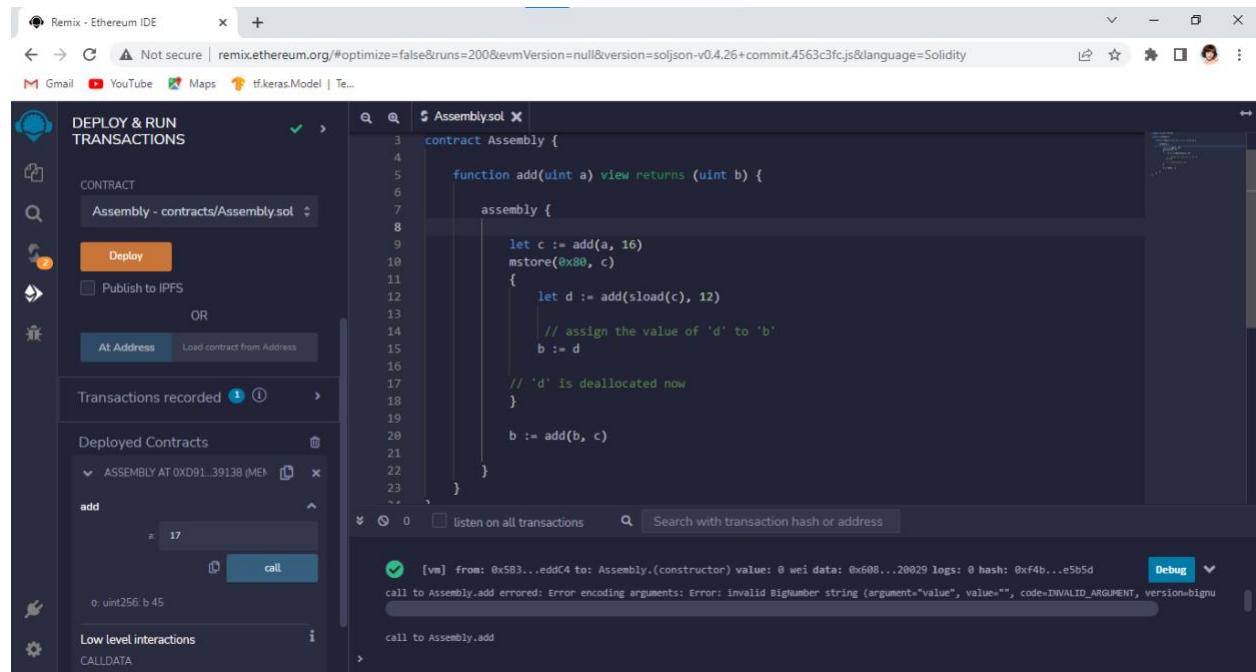


The screenshot shows the Remix Ethereum IDE interface. The left sidebar is titled "FILE EXPLORER" and lists several Solidity files. The main editor window is titled "# Assembly.sol" and contains the following Solidity code:

```
pragma solidity ^0.4.0;
contract Assembly {
    function add(uint a) view returns (uint b) {
        assembly {
            let c := add(a, 16)
            mstore(0x80, c)
            {
                let d := add(sload(c), 12)
                // assign the value of 'd' to 'b'
                b := d
                // 'd' is deallocated now
            }
            b := add(b, c)
        }
    }
}
```

The status bar at the bottom shows a transaction log: "[call] from: 0x5830da6a701c568545dCfcB03FcB875f56beddC4 to: Assembly.add(uint256) data: 0x100...00011".

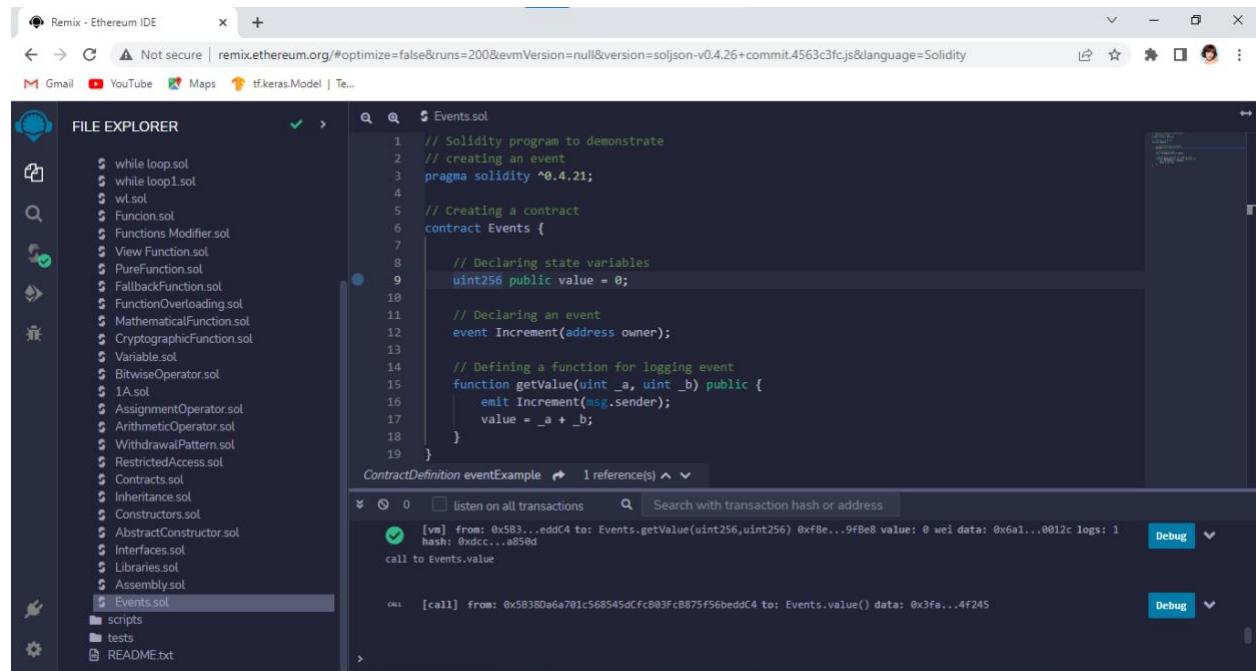
Output:



The screenshot shows the Remix Ethereum IDE interface with the "DEPLOY & RUN TRANSACTIONS" sidebar open. The "CONTRACT" dropdown is set to "Assembly - contracts/Assembly.sol". The "Deploy" button is highlighted. Below it, there are options for "Publish to IPFS" and "At Address". The "Transactions recorded" section shows one entry: "ASSEMBLY AT 0xD91...3913B (METH)". The main editor window is the same as in the previous screenshot. The status bar at the bottom shows a transaction log: "[vm] from: 0x583...eddC4 to: Assembly.(constructor) value: 0 wei data: 0x600...20029 logs: 0 hash: 0xf4b..e5b5d".

Events

Code:



The screenshot shows the Remix Ethereum IDE interface. The left sidebar is titled "FILE EXPLORER" and lists several Solidity files. The main editor window contains the following Solidity code:

```
// Solidity program to demonstrate
// creating an event
pragma solidity ^0.4.21;

// Creating a contract
contract Events {

    // Declaring state variables
    uint256 public value = 0;

    // Declaring an event
    event Increment(address owner);

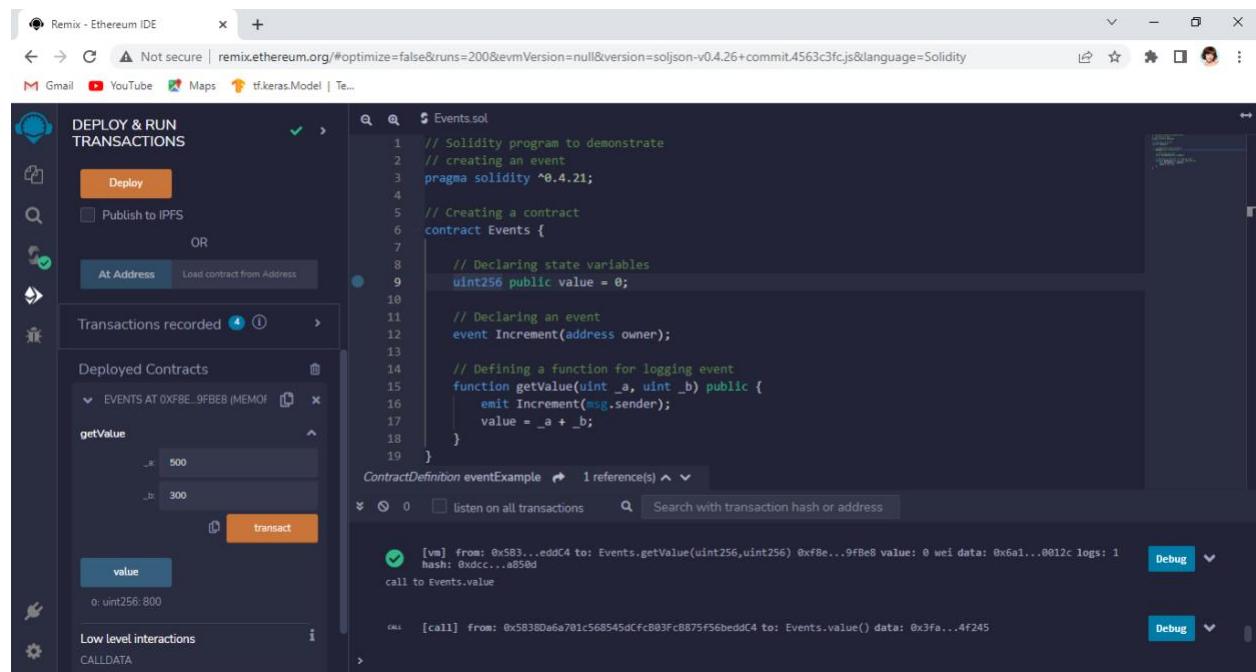
    // Defining a function for logging event
    function getValue(uint _a, uint _b) public {
        emit Increment(msg.sender);
        value = _a + _b;
    }
}
```

The "ContractDefinition" section at the bottom shows "eventExample" with 1 reference(s). The "Logs" section displays a log entry:

[vm] from: 0x583...edd4 to: Events.getValue(uint256,uint256) 0xf8e...9f8e8 value: 0 wei data: 0x6a1...0012c logs: 1
hash: 0xdcc...a850d
call to Events.value

The "CALL" section shows a call to "Events.value()".

Output:



The screenshot shows the Remix Ethereum IDE interface with the "DEPLOY & RUN TRANSACTIONS" sidebar open. The "Deploy" button is highlighted. The "Transactions recorded" section shows a recent transaction:

getValues
_a: 500
_b: 300
transact
value
0: uint256: 800

The main editor window contains the same Solidity code as the previous screenshot. The "ContractDefinition" section shows "eventExample" with 1 reference(s). The "Logs" section displays a log entry:

[vm] from: 0x583...edd4 to: Events.getValue(uint256,uint256) 0xf8e...9f8e8 value: 0 wei data: 0x6a1...0012c logs: 1
hash: 0xdcc...a850d
call to Events.value

The "CALL" section shows a call to "Events.value()".

Error Handling

Code:

The screenshot shows the Remix Ethereum IDE interface. The left sidebar is titled "FILE EXPLORER" and lists several Solidity files: whileloop1.sol, wl.sol, Funcion.sol, Functions Modifier.sol, View Function.sol, PureFunction.sol, FallbackFunction.sol, FunctionOverloading.sol, MathematicalFunction.sol, CryptographicFunction.sol, Variable.sol, BitwiseOperator.sol, I1.sol, AssignmentOperator.sol, ArithmeticOperator.sol, WithdrawalPattern.sol, RestrictedAccess.sol, Contracts.sol, Inheritance.sol, Constructors.sol, AbstractConstructor.sol, Interfaces.sol, Libraries.sol, Assembly.sol, Events.sol, ErrorHandling.sol (which is selected), scripts, tests, and README.txt. The main editor area contains the following Solidity code:

```
pragma solidity ^0.5.0;

contract ErrorHandling {

    function checkInput(uint _input) public view returns(string memory){
        require(_input >= 0, "invalid uint8");
        require(_input <= 255, "invalid uint8");

        return "Input is Uint8";
    }

    function Odd(uint _input) public view returns(bool){
        require(_input % 2 != 0);
        return true;
    }
}
```

The bottom status bar shows transaction logs and debug buttons.

Output:

The screenshot shows the Remix Ethereum IDE interface with the "DEPLOY & RUN TRANSACTIONS" sidebar open. The "Deploy" button is highlighted. The "Transactions recorded" section shows a deployment log for "ERRORHANDLING AT 0X9D7...B5E". The "checkInput" function is called with input 243, and the output is "Input is Uint8". The "Odd" function is also called with input 243, and the output is "bool: true". The bottom status bar shows transaction logs and debug buttons.

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/2ysbOFF | bash -s 1.2.0

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

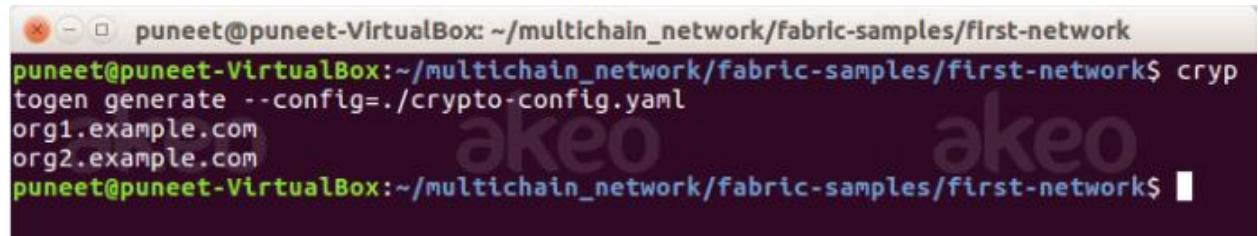
2. Generate Certificates

```
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.



The screenshot shows a terminal window with the following text:

```
puneet@puneet-VirtualBox:~/multichain_network/fabric-samples/first-network
puneet@puneet-VirtualBox:~/multichain_network/fabric-samples/first-network$ cryp
togen generate --config=./crypto-config.yaml
org1.example.com
org2.example.com
puneet@puneet-VirtualBox:~/multichain_network/fabric-samples/first-network$
```

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
```

```
puneet@puneet-VirtualBox:~/multichain_network/fabric-samples/first-network$ configtxgen -profile TwoOrgsOrdererGenesis -outputBlock ./composer-genesis.block
2019-03-18 14:50:56.908 IST [common/tools/configtxgen] main -> INFO 001 Loading configuration
2019-03-18 14:50:56.924 IST [msp] getMspConfig -> INFO 002 Loading NodeOUs
2019-03-18 14:50:56.924 IST [msp] getMspConfig -> INFO 003 Loading NodeOUs
2019-03-18 14:50:56.924 IST [common/tools/configtxgen] doOutputBlock -> INFO 004 Generating genesis block
2019-03-18 14:50:56.925 IST [common/tools/configtxgen] doOutputBlock -> INFO 005 Writing genesis block
puneet@puneet-VirtualBox:~/multichain_network/fabric-samples/first-network$
```

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

```
puneet@puneet-VirtualBox:~/multichain_network/fabric-samples/first-network$ configtxgen -profile TwoOrgsChannel -outputCreateChannelTx ./composer-channel.tx -channelID composerchannel
2019-03-18 15:17:36.592 IST [common/tools/configtxgen] main -> INFO 001 Loading configuration
2019-03-18 15:17:36.612 IST [common/tools/configtxgen] doOutputChannelCreateTx -> INFO 002 Generating new channel configtx
2019-03-18 15:17:36.614 IST [msp] getMspConfig -> INFO 003 Loading NodeOUs
2019-03-18 15:17:36.616 IST [msp] getMspConfig -> INFO 004 Loading NodeOUs
2019-03-18 15:17:36.656 IST [common/tools/configtxgen] doOutputChannelCreateTx -> INFO 005 Writing new channel tx
puneet@puneet-VirtualBox:~/multichain_network/fabric-samples/first-network$
```

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/c7d82435d76cb36bb1499edf1
b9b256144753a458a8467ed1ff67607cef09179_sk"
```

```
services:
  ca1:
    image: hyperledger/fabric-ca:1.4.0
    environment:
      - FABRIC_CA_HOME=/etc/hyperledger/fabric-ca-server
      - FABRIC_CA_SERVER_NAME=ca-org1
      - FABRIC_CA_SERVER_TLS_ENABLED=true
      - FABRIC_CA_SERVER_TLS_CERTFILE=/etc/hyperledger/fabric-ca-server-config/ca.org1.example.com-cert.pem
      - FABRIC_CA_SERVER_TLS_KEYFILE=/etc/hyperledger/fabric-ca-server-config/ca1_PRIVATE_KEY
    ports:
      - "7054:7054"
    command: sh -c '/Fabric-Ca-server start --ca.certfile /etc/hyperledger/fabric-ca-server-config/ca.org1.example.com-cert.pem --admincertfile /etc/hyperledger/fabric-ca-server-config/ca1_PRIVATE_KEY -b admin:adminpw -d'
```

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/b069c3b1761b013447f7da8f1
c266b26146daa0eb43d04a531f73675403b4d61_sk
```

```
services:
  ca2:
    image: hyperledger/fabric-ca:1.4.0
    environment:
      - FABRIC_CA_HOME=/etc/hyperledger/fabric-ca-server
      - FABRIC_CA_SERVER_NAME=ca-org2
      - FABRIC_CA_SERVER_TLS_ENABLED=true
      - FABRIC_CA_SERVER_TLS_CERTFILE=/etc/hyperledger/fabric-ca-server-config/ca.org2.example.com-cert.pem
      - FABRIC_CA_SERVER_TLS_KEYFILE=/etc/hyperledger/fabric-ca-server-config/ca2_PRIVATE_KEY
    ports:
      - "7054:7054"
    command: sh -c '/Fabric-Ca-server start --ca.certfile /etc/hyperledger/fabric-ca-server-config/ca.org2.example.com-cert.pem --admincertfile /etc/hyperledger/fabric-ca-server-config/ca2_PRIVATE_KEY -b admin:adminpw -d'
```

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 –

```
10  orderer.example.com:
11    container_name: orderer.example.com
12    image: hyperledger/fabric-orderer:$IMAGE_TAG
13    environment:
14      - ORDERER_GENERAL_LOGLEVEL=INFO
15      - ORDERER_GENERAL_LISTENADDRESS=0.0.0.0
16      - ORDERER_GENERAL_GENESISMETHOD=file
17      - ORDERER_GENERAL_GENESISFILE=/var/hyperledger/orderer/orderer.genesis.block
18      - ORDERER_GENERAL_LOCALMSPID=OrdererMSP
19      - ORDERER_GENERAL_LOCALMSPDIR=/var/hyperledger/orderer/msp
20    # enabled TLS
21    - ORDERER_GENERAL_TLS_ENABLED=true
22    - ORDERER_GENERAL_TLS_PRIVATEKEY=/var/hyperledger/orderer/tls/server.key
23    - ORDERER_GENERAL_TLS_CERTIFICATE=/var/hyperledger/orderer/tls/server.crt
24    - ORDERER_GENERAL_TLS_ROOTCAS=[/var/hyperledger/orderer/tls/ca.crt]
25    working_dir: /opt/gopath/src/github.com/hyperledger/fabric
26    command: orderer
27    volumes:
28      - .../composer-genesis.block:/var/hyperledger/orderer/orderer.genesis.block
29      - .../crypto-config/ordererOrganizations/example.com/orderers/orderer.example.com/msp:/var/hyperledger/orderer/msp
30      - .../crypto-config/ordererOrganizations/example.com/orderers/orderer.example.com/tls:/var/hyperledger/orderer/tls
31    ports:
32      - 7050:7050
33
```

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

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

```
35  peer0.org1.example.com:
36    extends:
37      file: peer-base.yaml
38      service: peer-base
39    environment:
40      - CORE_PEER_ID=peer0.org1.example.com
41      - CORE_PEER_ADDRESS=peer0.org1.example.com:7051
42      - CORE_PEER_GOSSIP_BOOTSTRAP=peer1.org1.example.com:7051
43      - CORE_PEER_GOSSIP_EXTERNALENDPOINT=peer0.org1.example.com:7051
44      - CORE_PEER_LOCALMSPID=Org1MSP
45    volumes:
46      - ...:/etc/configtx
47      - /var/run/:/host/var/run/
48      - .../crypto-config/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/msp:/etc/hyperledger/fabric/msp
49      - .../crypto-config/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls:/etc/hyperledger/fabric/tls
50    ports:
51      - 7051:7051
52      - 7053:7053
53
```

- 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 –

```
puneet@puneet-VirtualBox:~/Multichain_Network/fabric-samples/first-network$ docker-compose -f docker-compose-cli.yaml -f docker-compose-couch.yaml up -
d 2>&1
Creating orderer.example.com ...
Creating couchdb1 ...
Creating couchdb3 ...
Creating couchdb2 ...
Creating orderer.example.com
Creating couchdb0 ...
Creating couchdb2
Creating couchdb1
Creating couchdb3
Creating couchdb0 ... done
Creating peer0.org1.example.com ...
Creating couchdb3 ... done
Creating peer0.org2.example.com ...
Creating peer1.org1.example.com ...
Creating peer1.org2.example.com ...
Creating peer0.org1.example.com
Creating peer0.org2.example.com
Creating peer1.org2.example.com ... done
Creating cli ...
Creating cli ... done
```

- 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.

CONTAINER ID PORTS	IMAGE	NAMES	COMMAND	CREATED	STATUS
badfe5ea8a92	hyperledger/fabric-peer:latest	peer1.org1.example.com	"peer node start"	4 hours ago	Up 3 hours
50d9fa8f5dfb	hyperledger/fabric-peer:latest	peer0.org2.example.com	"peer node start"	4 hours ago	Up 3 hours
ee4b501b7d00	hyperledger/fabric-peer:latest	peer0.org1.example.com	"peer node start"	4 hours ago	Up 3 hours
32f0f4426a22	hyperledger/fabric-peer:latest	peer1.org2.example.com	"peer node start"	4 hours ago	Up 3 hours
ec8f9df17258	hyperledger/fabric-couchdb 4369/tcp, 9100/tcp, 0.0.0.0:8984->5984/tcp	couchdb3	"tini -- /docker-entrypoint-init.d/couchdb"	4 hours ago	Up 4 hours
2b4be64efcfe	hyperledger/fabric-couchdb 4369/tcp, 9100/tcp, 0.0.0.0:7984->5984/tcp	couchdb2	"tini -- /docker-entrypoint-init.d/couchdb"	4 hours ago	Up 4 hours
f922625a027e	hyperledger/fabric-couchdb 4369/tcp, 9100/tcp, 0.0.0.0:5984->5984/tcp	couchdb0	"tini -- /docker-entrypoint-init.d/couchdb"	4 hours ago	Up 4 hours
8de729e174b0	hyperledger/fabric-couchdb 4369/tcp, 9100/tcp, 0.0.0.0:6984->5984/tcp	couchdb1	"tini -- /docker-entrypoint-init.d/couchdb"	4 hours ago	Up 4 hours
18830d7ccf5e	hyperledger/fabric-tools:latest	cli	"/bin/bash"	11 hours ago	Up 11 hours
c7ef15b68cd5	hyperledger/fabric-orderer:latest 0.0.0.0:7050->7050/tcp	orderer.example.com	"orderer"	11 hours ago	Up 11 hours

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(nonce)
        new_hash = SHA256(text)
        if new_hash.startswith(prefix_str):
            print(f"Yay! Successfully mined bitcoins with nonce value:{nonce}")
            return new_hash

    raise BaseException(f"Couldn't find correct has after trying {MAX_NONCE} 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	<p>a) Aim: Using deep feed forward network with two hidden layers for performing multiclass classification and predicting the class.</p> <p>b) Aim: Using a deep feed forward network with two hidden layers for performing classification and predicting the probability of class.</p> <p>c) Aim: Using a deep feed forward network with two hidden layers for performing linear regression and predicting values.</p>
5	<p>a) Evaluating feed forward deep network for regression using KFold cross validation.</p> <p>b) Evaluating feed forward deep network for multiclass Classification using KFold cross-validation.</p>
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".format(e_matrix_A))
eigen_values_A,eigen_vectors_A=tf.linalg.eigh(e_matrix_A)
print("Eigen Vectors:\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:

```
Administrator: Windows PowerShell
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
Model: "sequential"
+-----+
| Layer (type)        | Output Shape      | Param # |
+=====+=====+=====+
| dense  (Dense)      | (None, 2)          | 6         |
+=====+=====+=====+
| dense_1 (Dense)     | (None, 1)          | 3         |
+=====+=====+=====+
Total params: 9
Trainable params: 9
Non-trainable params: 0
+-----+
None
[array([[ 0.324126 ,  0.06514561],
       [-0.06398606,  0.25455737]], dtype=float32), array([0., 0.], dtype=float32), array([-1.166442 ],
       [ 1.0120543]), dtype=float32), array([0.], dtype=float32)]
2021-04-17 12:17:11.354966: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:116] None of the MLIR optimization passes are enabled (registered 2)
Epoch 1/1000
1/1 [=====] - 2s 2ms/step - loss: 0.7076 - accuracy: 0.5000
Epoch 2/1000
1/1 [=====] - 0s 7ms/step - loss: 0.7073 - accuracy: 0.2500
Epoch 3/1000
1/1 [=====] - 0s 6ms/step - loss: 0.7071 - accuracy: 0.2500
Epoch 4/1000
1/1 [=====] - 0s 6ms/step - loss: 0.7069 - accuracy: 0.2500
Epoch 5/1000
1/1 [=====] - 0s 7ms/step - loss: 0.7066 - accuracy: 0.2500
Epoch 6/1000
1/1 [=====] - 0s 4ms/step - loss: 0.7064 - accuracy: 0.2500
Epoch 7/1000
1/1 [=====] - 0s 2ms/step - loss: 0.7062 - accuracy: 0.2500
Epoch 8/1000
1/1 [=====] - 0s 2ms/step - loss: 0.7059 - accuracy: 0.2500
Epoch 9/1000
1/1 [=====] - 0s 4ms/step - loss: 0.7057 - accuracy: 0.2500
```

```

Administrator: Windows PowerShell
1/1 [=====] - 0s 4ms/step - loss: 0.5057 - accuracy: 1.0000
Epoch 989/1000
1/1 [=====] - 0s 3ms/step - loss: 0.5054 - accuracy: 1.0000
Epoch 990/1000
1/1 [=====] - 0s 5ms/step - loss: 0.5052 - accuracy: 1.0000
Epoch 991/1000
1/1 [=====] - 0s 5ms/step - loss: 0.5049 - accuracy: 1.0000
Epoch 992/1000
1/1 [=====] - 0s 2ms/step - loss: 0.5048 - accuracy: 1.0000
Epoch 993/1000
1/1 [=====] - 0s 4ms/step - loss: 0.5045 - accuracy: 1.0000
Epoch 994/1000
1/1 [=====] - 0s 2ms/step - loss: 0.5042 - accuracy: 1.0000
Epoch 995/1000
1/1 [=====] - 0s 4ms/step - loss: 0.5040 - accuracy: 1.0000
Epoch 996/1000
1/1 [=====] - 0s 2ms/step - loss: 0.5037 - accuracy: 1.0000
Epoch 997/1000
1/1 [=====] - 0s 2ms/step - loss: 0.5035 - accuracy: 1.0000
Epoch 998/1000
1/1 [=====] - 0s 4ms/step - loss: 0.5032 - accuracy: 1.0000
Epoch 999/1000
1/1 [=====] - 0s 4ms/step - loss: 0.5030 - accuracy: 1.0000
Epoch 1000/1000
1/1 [=====] - 0s 4ms/step - loss: 0.5027 - accuracy: 1.0000
[array([[0.6900411, 0.5139764],
       [0.6899988, 0.50888795]], dtype=float32), array([-0.6898802, 0.00666144], dtype=float32), array([-2.4736412],
       [1.6274512]], dtype=float32), array([-0.41508964], dtype=float32)]
[[0.40029204]
[0.60435593]
[0.60630935]
[0.39012325]]
(venv) PS D:\keras>

```

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
>>>

```

```

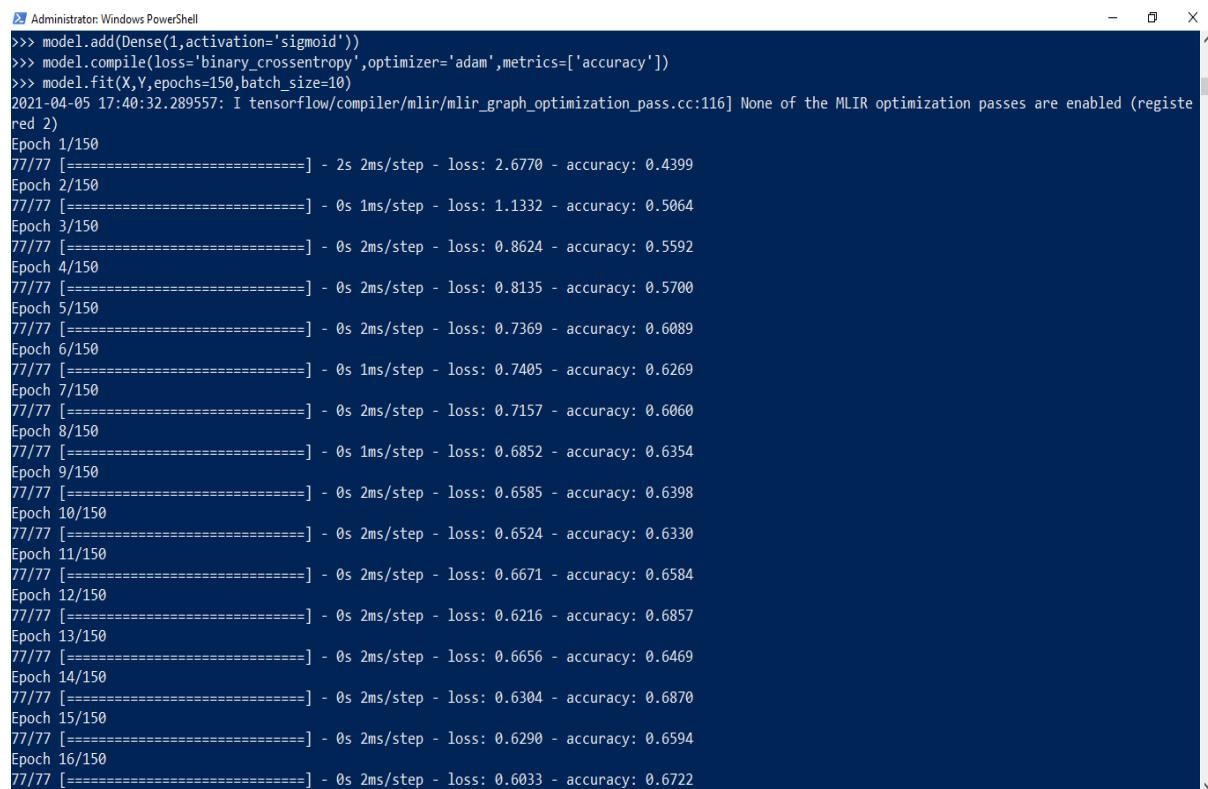
Administrator: Windows PowerShell
>>> dataset=loadtxt('pima-indians-diabetes.csv',delimiter=',')
>>> dataset
array([[ 6. , 148. , 72. , ..., 0.627, 50. , 1. ],
       [ 1. , 85. , 66. , ..., 0.351, 31. , 0. ],
       [ 8. , 183. , 64. , ..., 0.672, 32. , 1. ],
       ...,
       [ 5. , 121. , 72. , ..., 0.245, 30. , 0. ],
       [ 1. , 126. , 60. , ..., 0.349, 47. , 1. ],
       [ 1. , 93. , 70. , ..., 0.315, 23. , 0. ]])
>>> X=dataset[:,0:8]
>>> Y=dataset[:,8]
>>> X
array([[ 6. , 148. , 72. , ..., 33.6 , 0.627, 50. ],
       [ 1. , 85. , 66. , ..., 26.6 , 0.351, 31. ],
       [ 8. , 183. , 64. , ..., 23.3 , 0.672, 32. ],
       ...,
       [ 5. , 121. , 72. , ..., 26.2 , 0.245, 30. ],
       [ 1. , 126. , 60. , ..., 30.1 , 0.349, 47. ],
       [ 1. , 93. , 70. , ..., 30.4 , 0.315, 23. ]])
>>>
>>> Y
array([1., 0., 1., 0., 1., 0., 1., 0., 1., 0., 1., 1., 1., 1.,
       1., 0., 1., 0., 1., 1., 1., 1., 0., 0., 0., 0., 1., 0., 0.,
       0., 0., 0., 1., 1., 0., 0., 0., 1., 0., 0., 1., 0., 0., 0.,
       0., 0., 1., 0., 0., 1., 0., 0., 0., 1., 0., 0., 1., 0., 0.,
       0., 0., 1., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 0., 1.,
```

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 (registered 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:

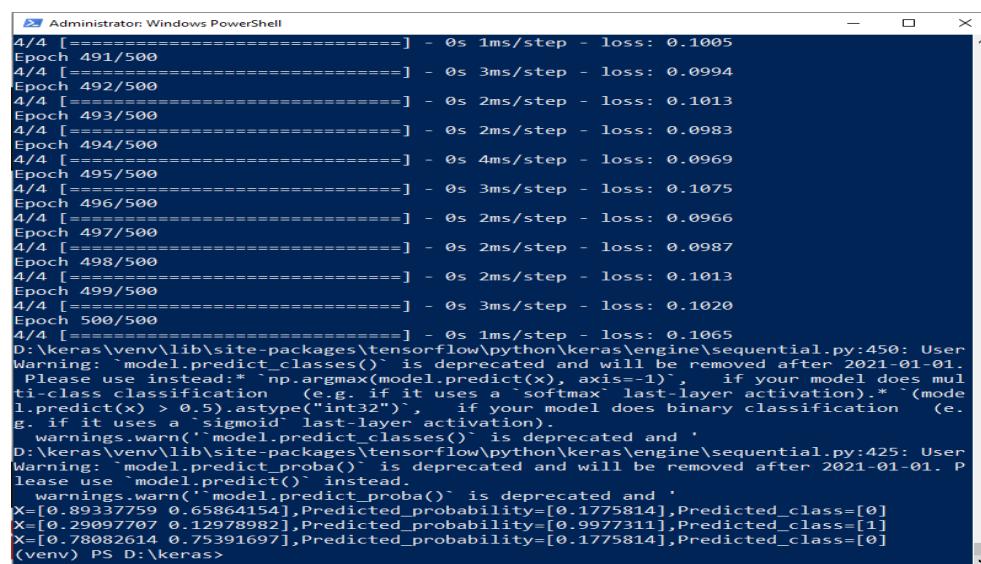
```
Administrator: Windows PowerShell
4/4 [=====] - 0s 2ms/step - loss: 0.6935
Epoch 488/500
4/4 [=====] - 0s 2ms/step - loss: 0.6927
Epoch 489/500
4/4 [=====] - 0s 3ms/step - loss: 0.6931
Epoch 490/500
4/4 [=====] - 0s 3ms/step - loss: 0.6928
Epoch 491/500
4/4 [=====] - 0s 2ms/step - loss: 0.6938
Epoch 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: User
Warning: `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 mul
ti-class classification (e.g. if it uses a `softmax` last-layer activation). * `(mode
l.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:



```
Administrator: Windows PowerShell
4/4 [=====] - 0s 1ms/step - loss: 0.1005
Epoch 491/500
4/4 [=====] - 0s 3ms/step - loss: 0.0994
Epoch 492/500
4/4 [=====] - 0s 2ms/step - loss: 0.1013
Epoch 493/500
4/4 [=====] - 0s 2ms/step - loss: 0.0983
Epoch 494/500
4/4 [=====] - 0s 4ms/step - loss: 0.0969
Epoch 495/500
4/4 [=====] - 0s 3ms/step - loss: 0.1075
Epoch 496/500
4/4 [=====] - 0s 2ms/step - loss: 0.0966
Epoch 497/500
4/4 [=====] - 0s 2ms/step - loss: 0.0987
Epoch 498/500
4/4 [=====] - 0s 2ms/step - loss: 0.1013
Epoch 499/500
4/4 [=====] - 0s 3ms/step - loss: 0.1020
Epoch 500/500
4/4 [=====] - 0s 1ms/step - loss: 0.1065
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 '
D:\keras\venv\lib\site-packages\tensorflow\python\keras\engine\sequential.py:425: UserWarning: `model.predict_proba()` is deprecated and will be removed after 2021-01-01. Please use `model.predict()` instead.
  warnings.warn(`model.predict_proba()` is deprecated and '
X=[0.89337759 0.65864154],Predicted_probability=[0.1775814],Predicted_class=[0]
X=[0.29097707 0.12978982],Predicted_probability=[0.9977311],Predicted_class=[1]
X=[0.78082614 0.75391697],Predicted_probability=[0.1775814],Predicted_class=[0]
(venv) PS D:\keras>
```

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,noisy=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,noisy=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'))
```

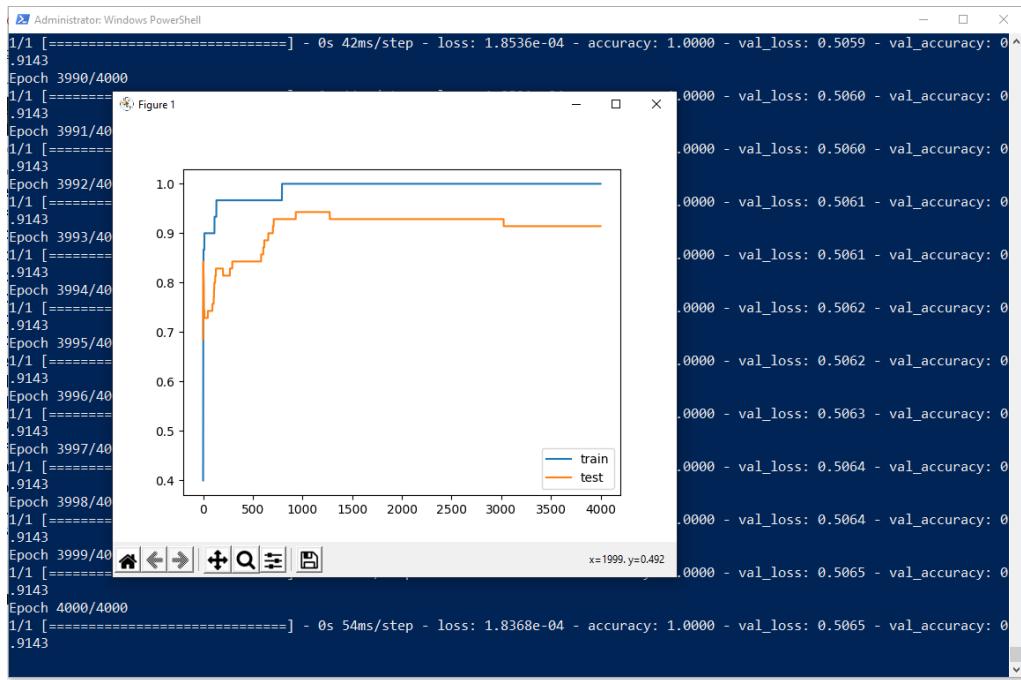
```
Wider: -22.17 (24.38) MSE  
(venv) PS D:\keras>
```

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,noisy=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)

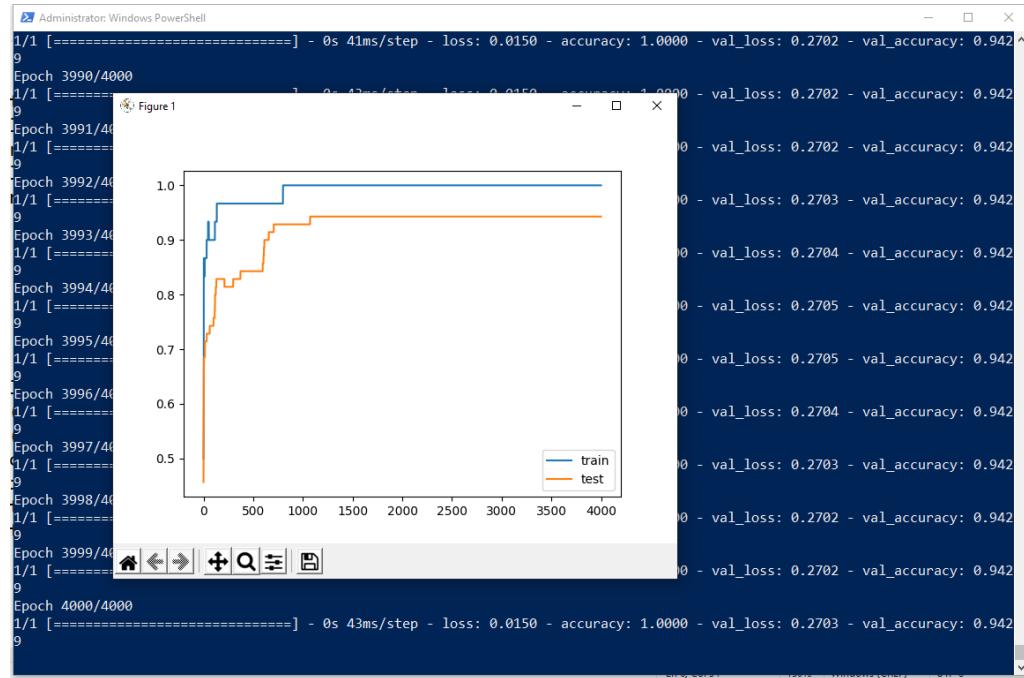
pyplot.plot(history.history['accuracy'],label='train')

pyplot.plot(history.history['val_accuracy'],label='test')

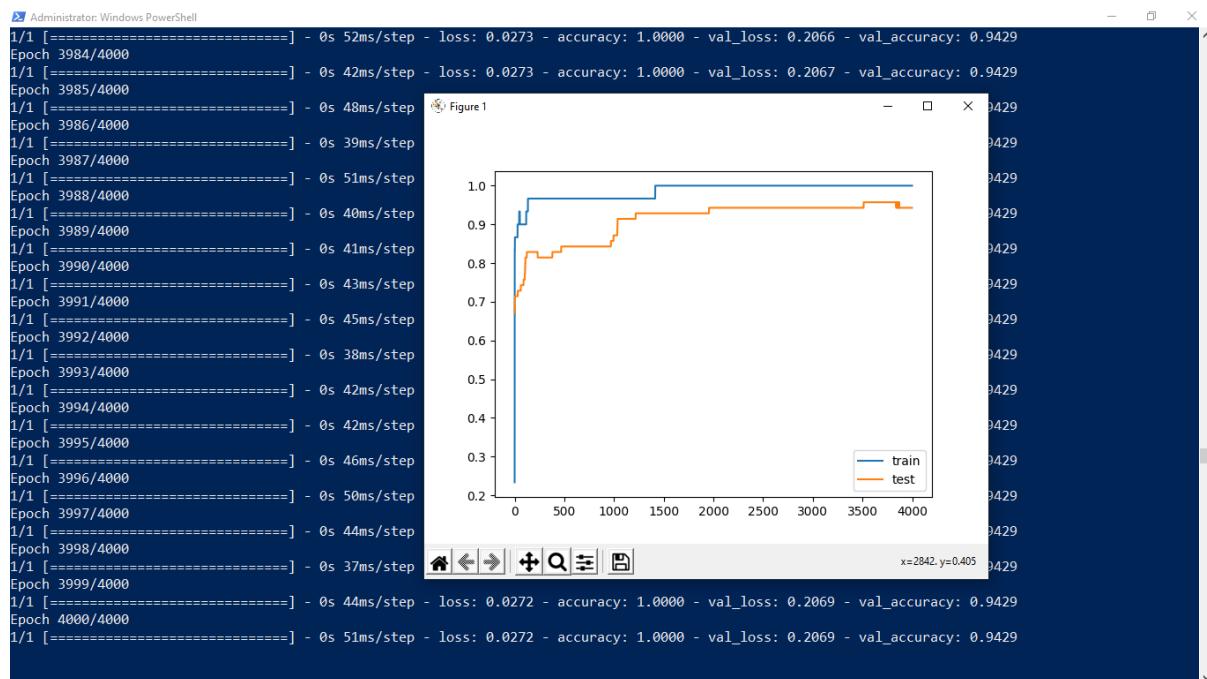
pyplot.legend()

pyplot.show()

```



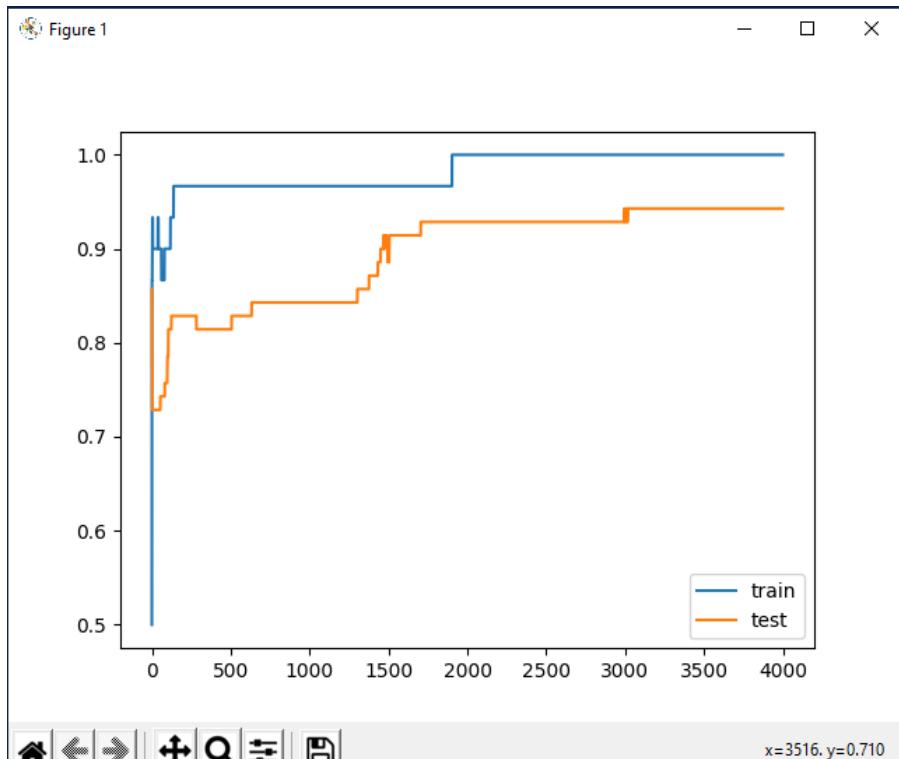
By replacing l2 regularizer with l1 regularizer at the same learning rate 0.001 we get the following output.



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,nois=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)
print('*****')

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):

```

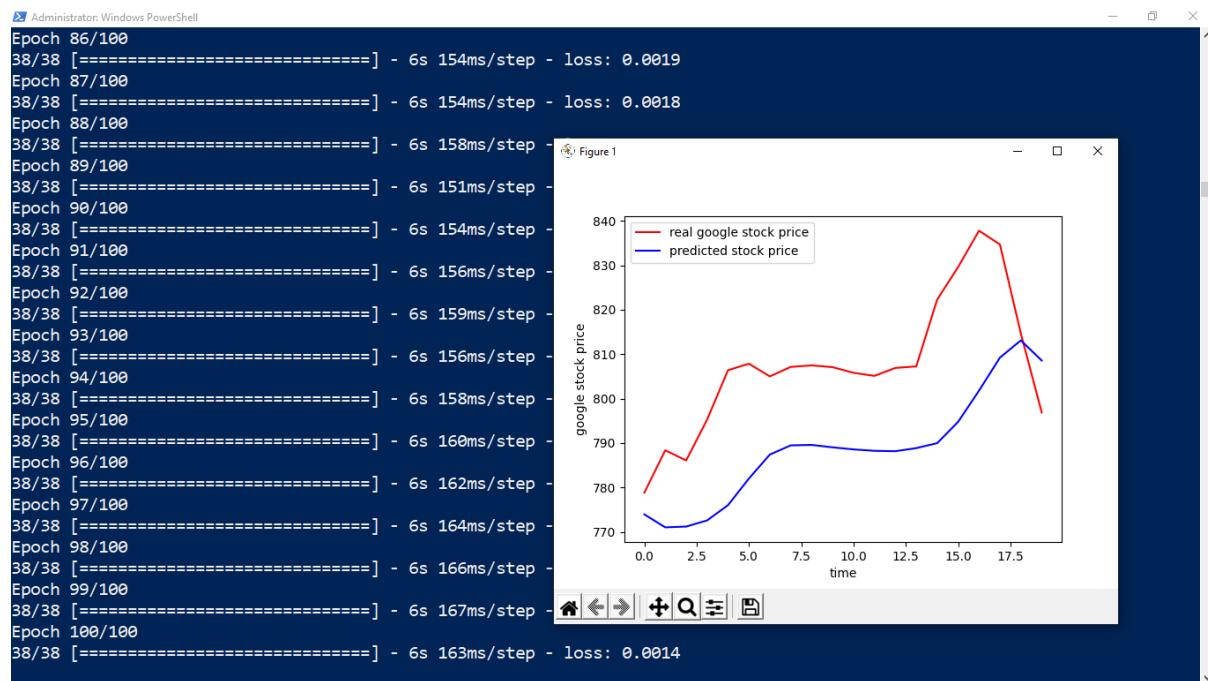
```

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.

```

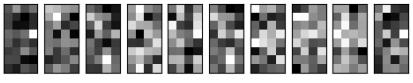
import keras
from keras import layers
from keras.datasets import mnist

```

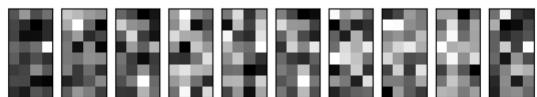
```
import numpy as np
encoding_dim=32
#this is our input image
input_img=keras.Input(shape=(784,))
#"encoded" is the encoded representation of the input
encoded=layers.Dense(encoding_dim, activation='relu')(input_img)
#"decoded" is the lossy reconstruction of the input
decoded=layers.Dense(784, activation='sigmoid')(encoded)
#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:

```
Administrator: Windows PowerShell
235/235 [=====] - 3s 13ms/step - loss: 0.0929 - val_loss: 0.0918
Epoch 34/50
235/235 [=====] - 3s 13ms/step - loss: 0.0926 - val_loss: 0.0917
Epoch 35/50
235/235 [=====] - 3s 13ms/step - loss: 0.0928 - val_loss: 0.0917
Epoch 36/50
235/235 [=====]
Figure 1
7 2 1 0 4 1 4 9 5 9
[]
7 2 1 0 4 1 4 9 5 9
[]
7 2 1 0 4 1 4 9 5 9
[]
235/235 [=====] - 3s 14ms/step - loss: 0.0929 - val_loss: 0.0919
Epoch 48/50
235/235 [=====] - 3s 14ms/step - loss: 0.0927 - val_loss: 0.0915
Epoch 49/50
235/235 [=====] - 3s 13ms/step - loss: 0.0927 - val_loss: 0.0915
Epoch 50/50
235/235 [=====] - 3s 15ms/step - loss: 0.0926 - val_loss: 0.0914
```

7 2 1 0 4 1 4 9 5 9

[]

7 2 1 0 4 1 4 9 5 9

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:

```
^^^^^^^^^^^^^^^^^^^^^
[0.9145307  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='/tmp/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)

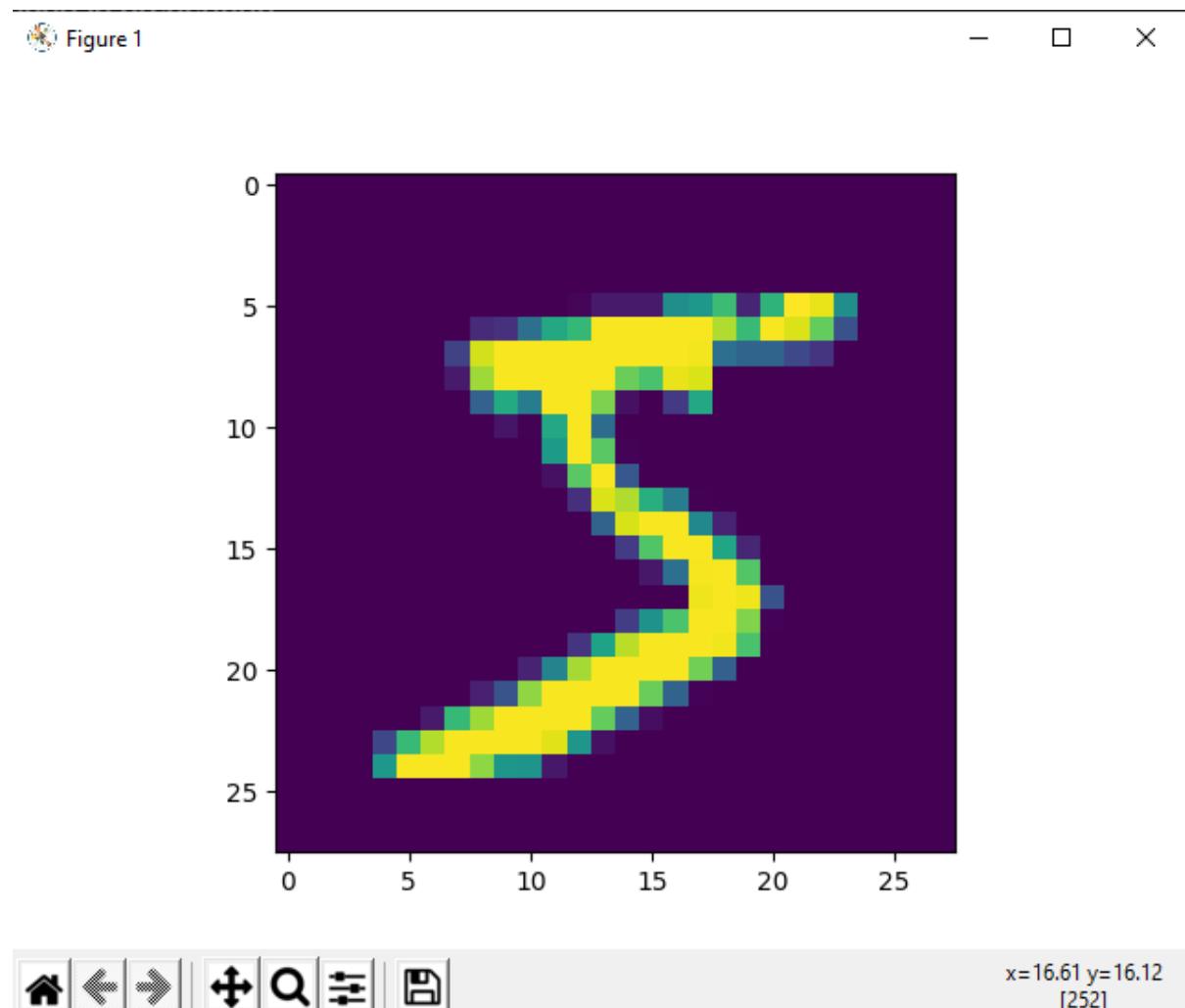
print(model.predict(X_test[:4]))

#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. 1. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0. 0.]
 [1. 0. 0. 0. 0. 0. 0. 0. 0.]]]
(venv) PS D:\keras>
```


Index

SR No	Title	Page No	Sign
1	File System Analysis using the sleuth kit		
2	Explore windows forensic tool (OS Forensics)		
3	Using forensic toolkit (FTK) & writing report using FTK		
4	A. Using file recovery tool (FTK Imager) Creating image.		
	B. Recover deleted file using recuva, Pc inspector file recovery, Recover my file, R-studio.		
5.	A. Perform TCP SYN flood attack with kali linux and HPing3		
	B. Using log and traffic capturing and analyzing tool (Wireshark)		
	C. Using network forensics analysis tool (Network Miner)		
6.	Dump memory content using PMdump		
7.	Using data acquisition tool (Pro discovery)		
8.	A. Using Steganography tool (S-tool)		
	B. Using whitespace steganography tool SNOW		
9.	Performing password cracking (Cain & Abel)		

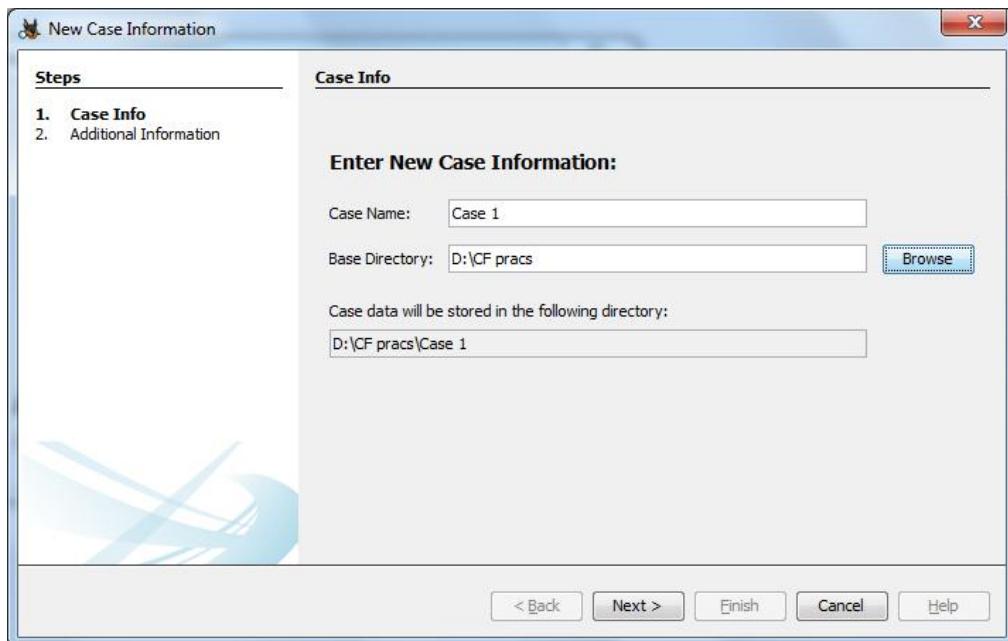
Practical No: 01

How to Start a Case.

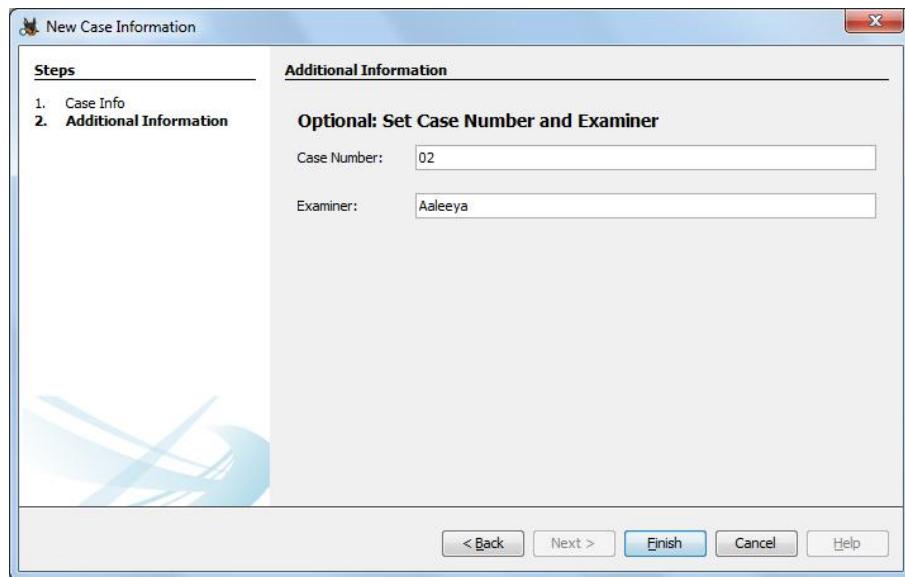
Upon starting Autopsy 3.1.2, a window will open with three selections to make: create a new case, open existing case, or to open a recent case.



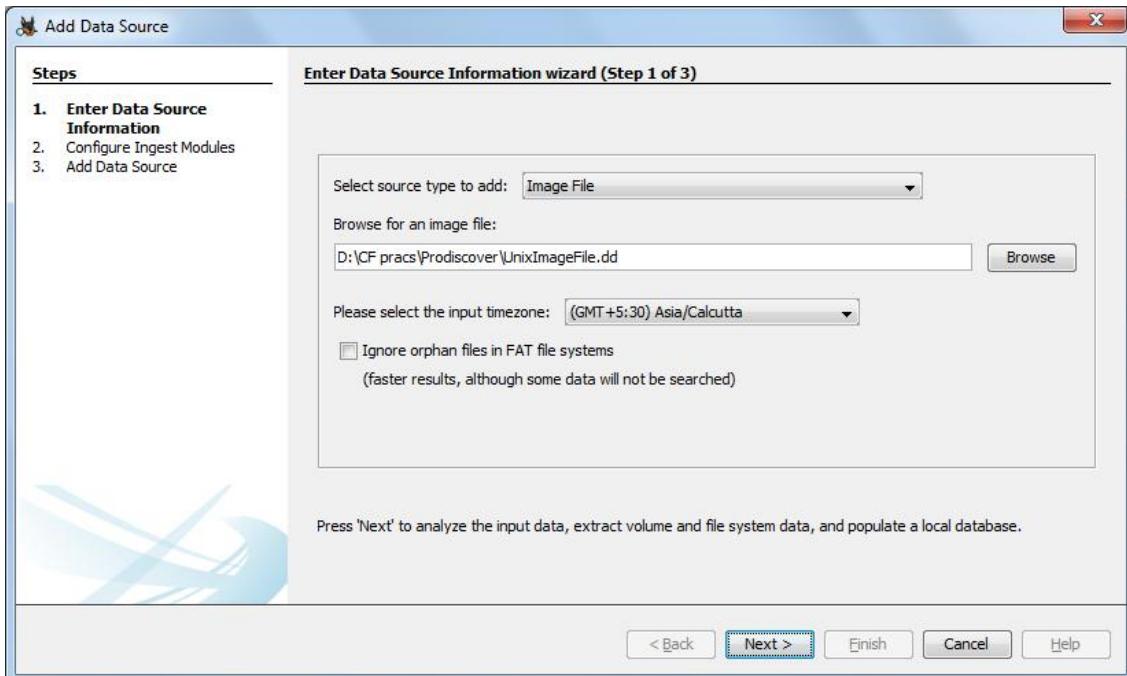
Step 1) Select the “Create New Case” option and be directed to a new window that will have information to fill in, we will be naming the case “Test.”



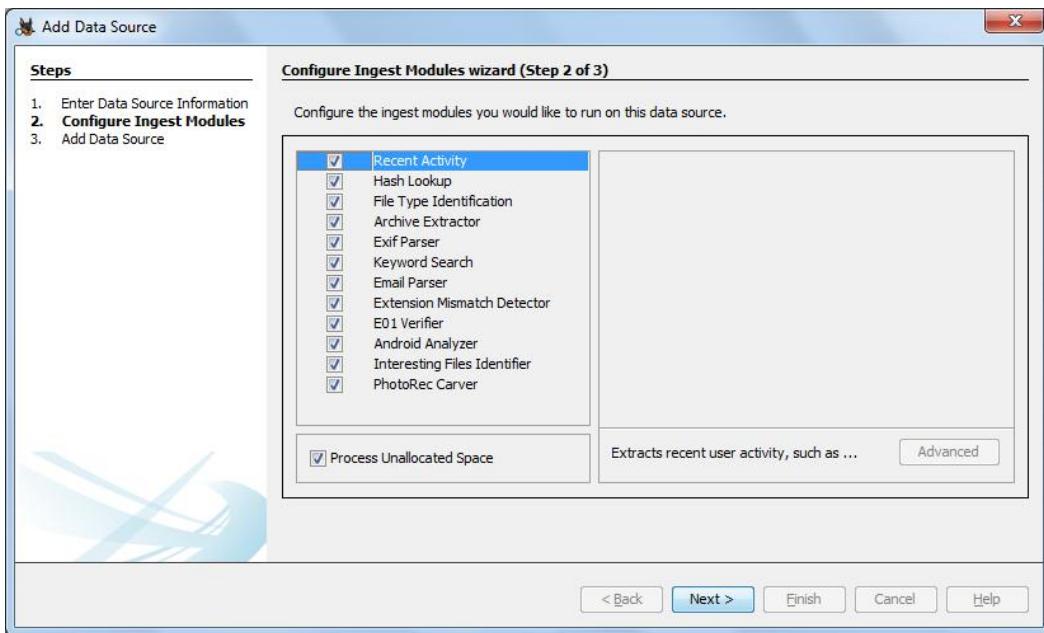
Step 2) After the information has been filled in select the next button. The next window will allow the investigator to fill in the case number and examiner name. This is for the purpose of creating better documentation and logging. After the information is filled in select the finish button to continue.



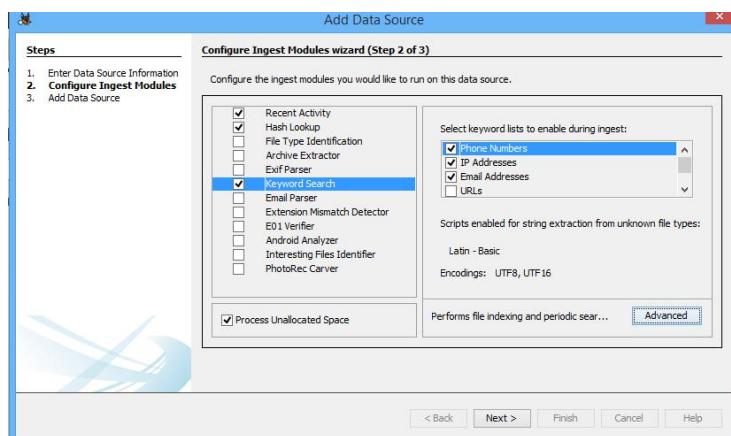
Step 3) The next step in the investigation will be to add an image file to the case. The image file can be chosen from a wide variety of formats including: img, dd, 001, aa, and e01. Use the browse button to find the image that is desired to work with and select add. Options to choose the timezone of 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.

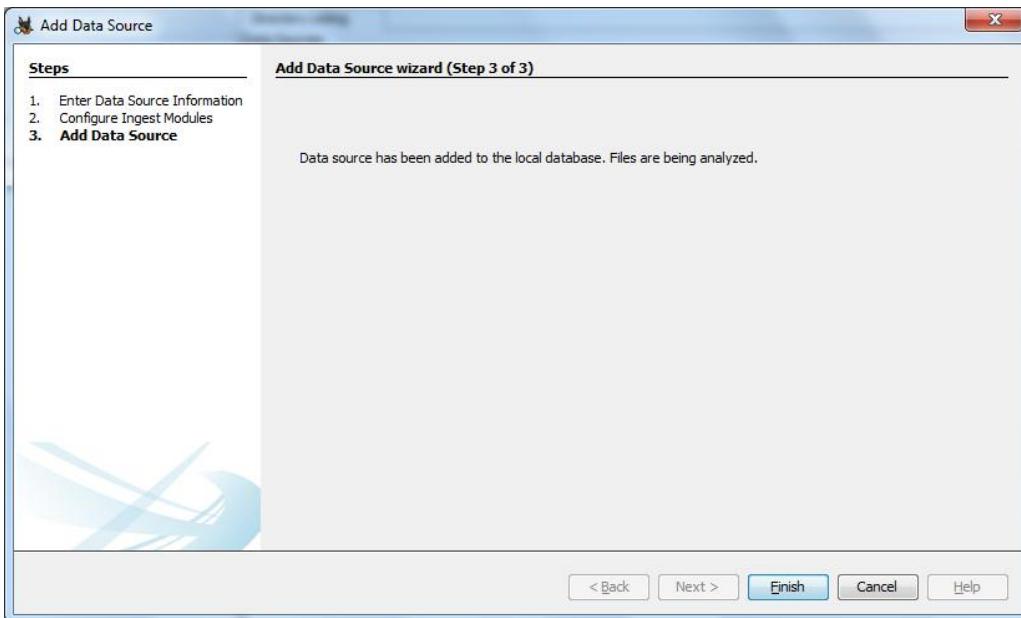
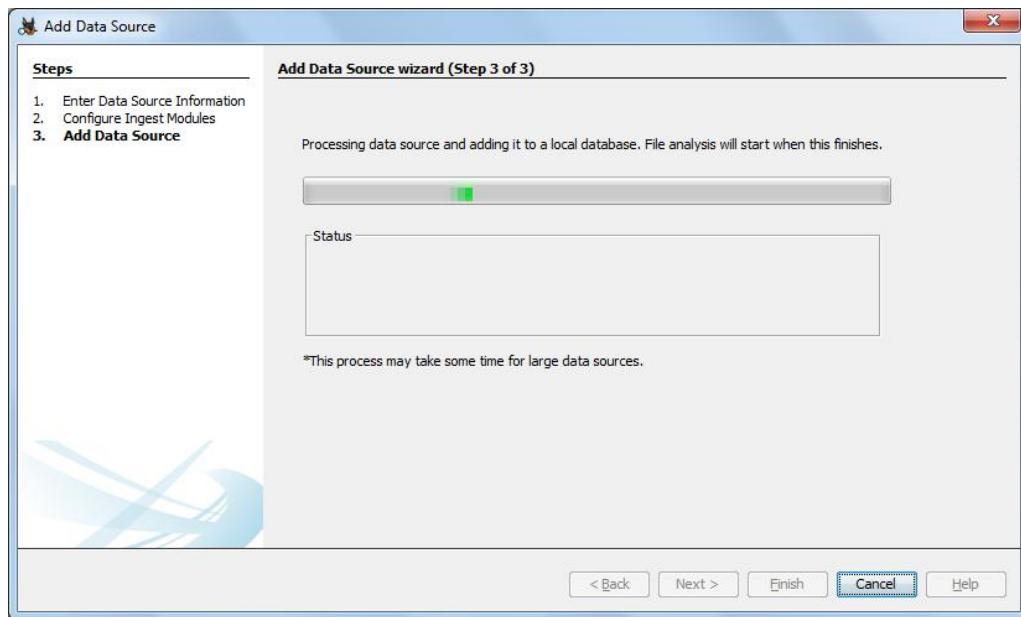


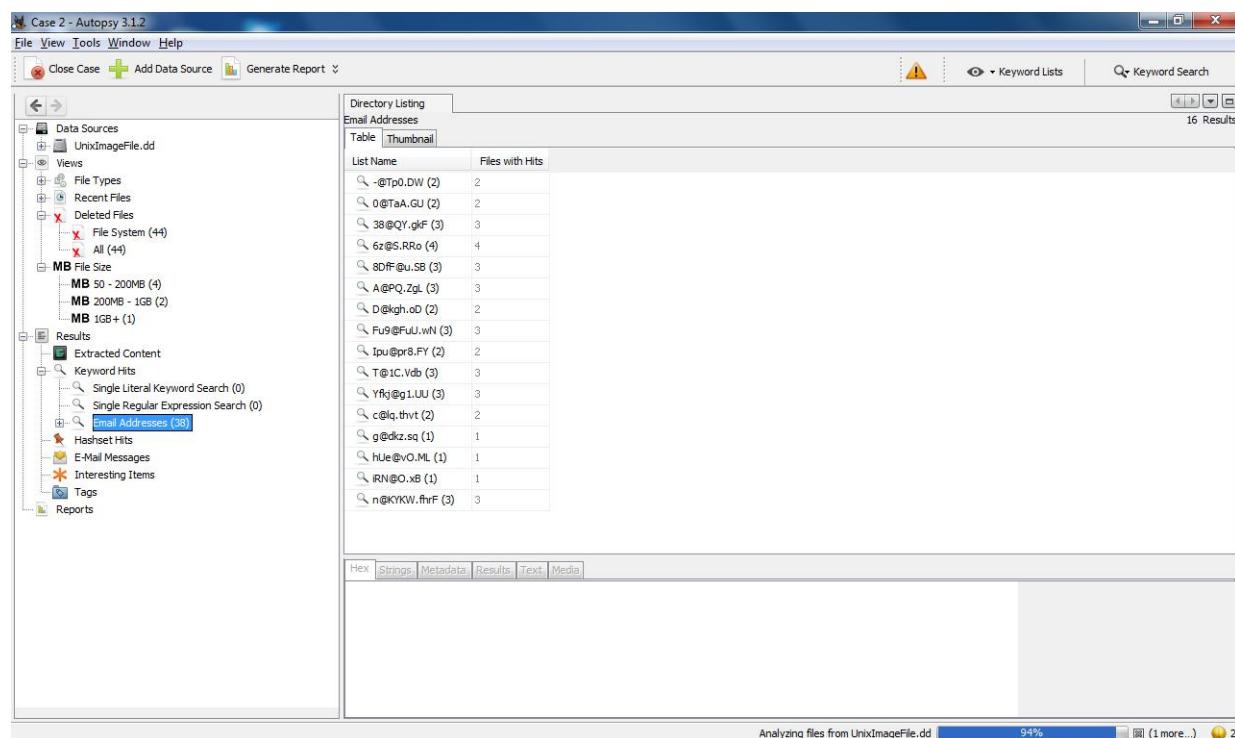
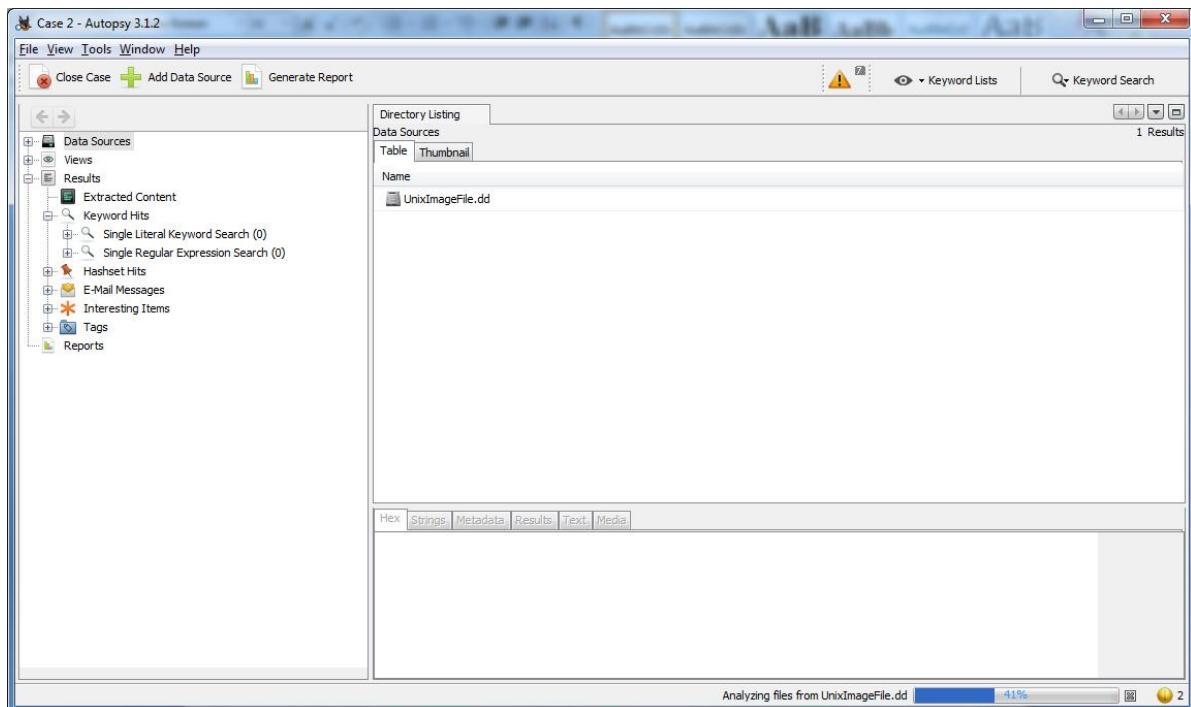
Step 5) The following window will bring the investigator to the Ingest wizard panel, which is one of the new features offered in Autopsy. There are three options in the first box: Recent Activity, Hash lookup, and Keyword Searches.



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.





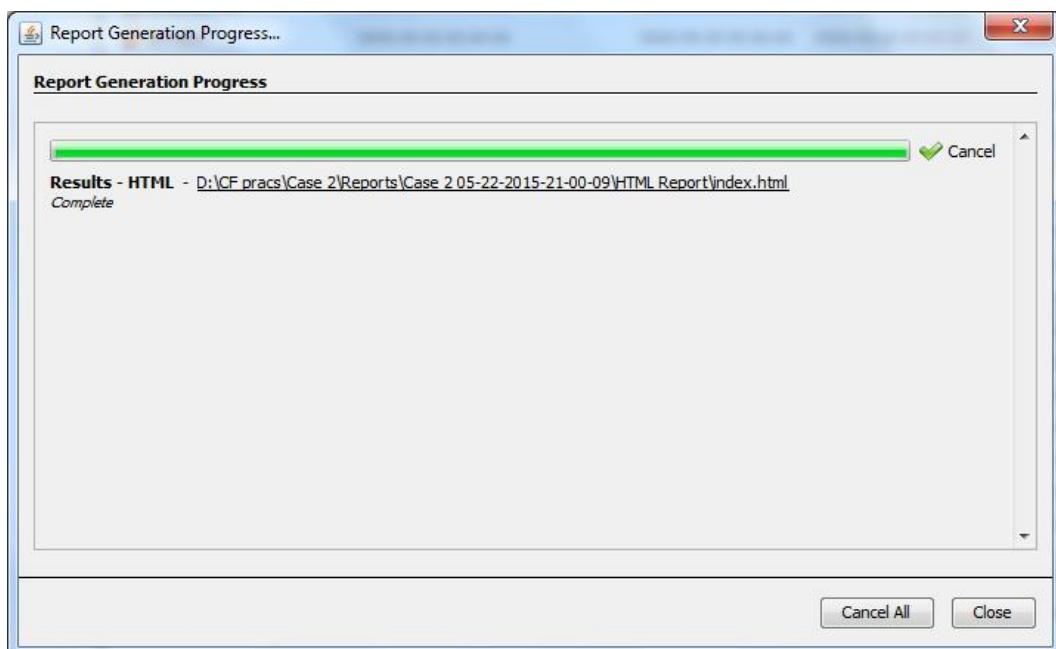
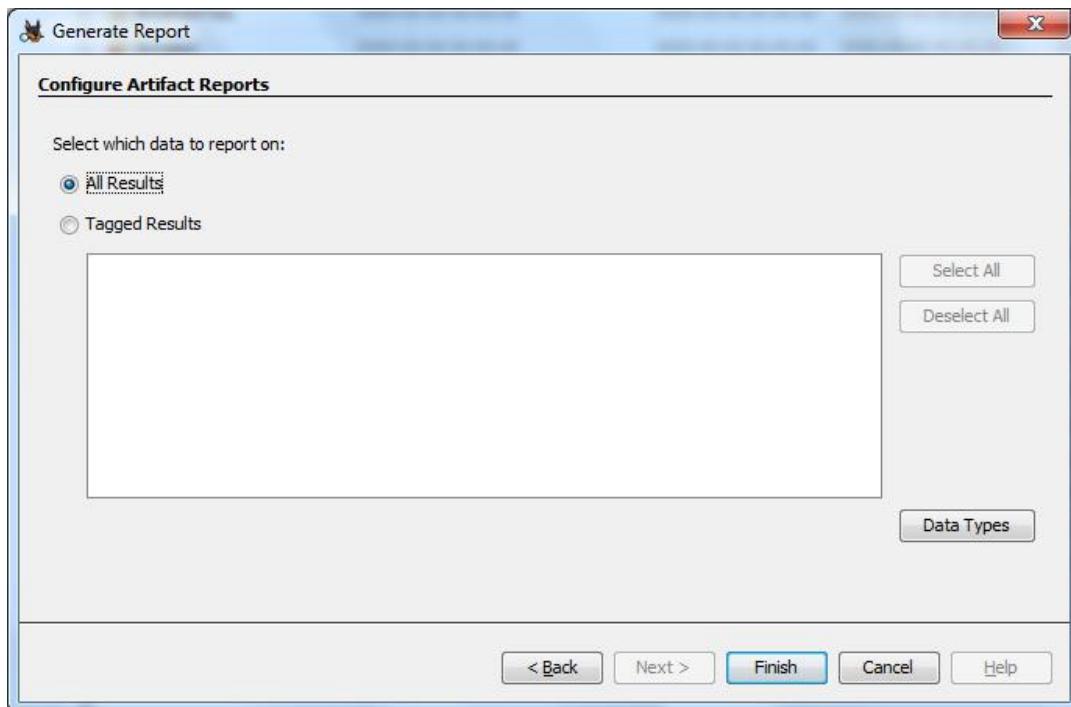
Step 6) After finishing the keyword parameters the screen will be laid out for the user.

The screenshot shows the Autopsy 3.1.2 interface with the title bar "Case 2 - Autopsy 3.1.2". The main window displays a "Directory Listing" for the file "Img_UinxImageFile.dd". The table view shows 17 results with columns: Name, Modified Time, Change Time, Access Time, Created Time, Size, and Flags(Dr). The results include various files like ".OrphanFiles", "autorun.inf", "comp forensics softwares", "Mr Nobody (2009)", "New folder", "Psycho (1960)", "Triangle [2009] 720p", and several Microsoft Word documents ("imp.docx", "prao01.docx", "prao02.docx"). The interface also includes a sidebar with sections like Data Sources, Views, File Types, Recent Files, Deleted Files, MB File Sets, Results, Extracted Content, and Reports. At the bottom, there are tabs for Hex, Strings, Metadata, Results, Text, and Media, along with a progress bar indicating "Analyzing files from UnixImageFile.dd" at 100% completion.

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.

The screenshot shows the "Generate Report" dialog box with the title "Select and Configure Report Modules". On the left, under "Report Modules:", there is a list of options with radio buttons: "Results - Excel", "Results - HTML" (which is selected), "Files - Text", "Google Earth/KML", "STIX", and "TSK Body File". To the right of the list, there is a descriptive text: "A report about results and tagged items in HTML format." Below this, another text box contains the message: "This report will be configured on the next screen." At the bottom of the dialog are buttons for navigation: "< Back", "Next >" (highlighted in blue), "Finish", "Cancel", and "Help".



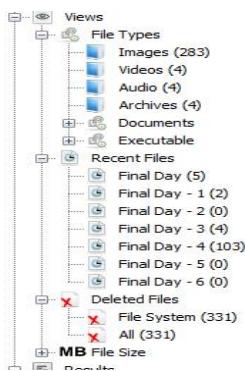
Step 8) With the report on hand the investigator will have an idea of what to expect as well as a list of programs that are installed on the machine. This can help investigators gather all the evidence they need to perform a complete investigation.

The screenshot shows the 'Autopsy Report for case' window. The left sidebar is titled 'Report Navigation' and lists several items: Case Summary, E-Mail Messages (4), Encryption Detected (1), Keyword Hits (40), Tagged Files (0), Tagged Results (0), and Thumbnails (0). The main content area is titled 'Autopsy Forensic Report' and includes a warning: 'Warning, this report was run before ingest services completed!'. It shows the following details: Case: Case 2, Case Number: 02, Examiner: Aleeya, and Number of Images: 1. Below this is a section titled 'Image Information:' with a sub-section for 'UnixImageFile.dd'. It shows the Timezone: Asia/Calcutta and Path: D:\ICF pracs\ProDiscover\UnixImageFile.dd. At the bottom of the main content area is a cartoon dog icon.

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.

The screenshot shows the 'Data Sources' tree view. A red box highlights the 'precious.img' node, which has three child nodes: 'vol1 (Unallocated: 0-49663)', 'vol2 (NTFS / exFAT (0x07): 97-128269920)', and 'vol3 (Unallocated: 250624-381183)'. Below this are other tabs: 'Views', 'Results', 'Extracted Content' (which is expanded to show 'Operating System User Account (7)' and 'Recent Documents (9)'), and 'Recent Files'.

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.

Name	Location	Modified Time	Chanc
\$_MPFO~1.DOC	/img_UinxImageFile.dd/\$OrphanFiles/_\$MPFO~1.DOC	2015-04-27 15:27:52 IST	0000-C
_AIN&A~1	/img_UinxImageFile.dd/\$OrphanFiles/_AIN&A~1	2015-04-27 14:03:06 IST	0000-C
_RODIS~1	/img_UinxImageFile.dd/\$OrphanFiles/_RODIS~1	2015-04-27 14:03:06 IST	0000-C
_TOOLS	/img_UinxImageFile.dd/\$OrphanFiles/_TOOLS	2015-04-27 14:03:22 IST	0000-C
PADCURSO.RS(/img_UinxImageFile.dd/\$OrphanFiles/PADCURSO.RS(0000-00-00 00:00:00	0000-C
KEEPPRYVAT	/img_UinxImageFile.dd/\$OrphanFiles/KEEPPRYVAT	2007-10-07 19:10:28 IST	0000-C
VICS~1.XLS	/img_UinxImageFile.dd/\$OrphanFiles/VICS~1.XLS	2007-10-07 10:55:34 IST	0000-C
toja.cfg	/img_UinxImageFile.dd/\$OrphanFiles/toja.cfg	2005-09-26 12:15:32 IST	0000-C
_odolist.txt	/img_UinxImageFile.dd/\$OrphanFiles/_odolist.txt	2007-10-06 10:08:12 IST	0000-C
_NENOT~1.ONE	/img_UinxImageFile.dd/\$OrphanFiles/_NENOT~1.ONE	2014-10-30 11:07:22 IST	0000-C
_RODIS~1.EXE	/img_UinxImageFile.dd/\$OrphanFiles/_RODIS~1.EXE	2011-09-05 22:45:38 IST	0000-C
_ryptlib.dll	/img_UinxImageFile.dd/\$OrphanFiles/_ryptlib.dll	1996-05-06 21:15:18 IST	0000-C
_IFUTIL.DLL	/img_UinxImageFile.dd/\$OrphanFiles/_IFUTIL.DLL	1996-05-07 01:06:56 IST	0000-C
_TOOLS.EXE	/img_UinxImageFile.dd/\$OrphanFiles/_TOOLS.EXE	1996-05-06 20:55:56 IST	0000-C
_TOOLS.GID	/img_UinxImageFile.dd/\$OrphanFiles/_TOOLS.GID	2008-02-10 00:37:24 IST	0000-C
_TOOLS.HLP	/img_UinxImageFile.dd/\$OrphanFiles/_TOOLS.HLP	1996-04-21 06:31:08 IST	0000-C
lh_all	lh_all	1996-05-06 21:16:48 IST	0000-C

Case 2 - Autopsy 3.1.2

File View Tools Window Help

Close Case Add Data Source Generate Report

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
- All (44)
- MB File Size
- MB 50 - 200MB (4)
- MB 200MB - 1GB (2)
- MB 1GB+ (1)
- Results

Directory Listing /img_UinxImageFile.dd

Table Thumbnail

Name	Modified Time	Change Time
\$OrphanFiles	0000-00-00 00:00:00	0000-00-00 00:00:00
\$Unalloc	0000-00-00 00:00:00	0000-00-00 00:00:00
autorun.inf	2015-04-27 15:49:06 IST	0000-00-00 00:00:00
comp forensics softwares	2015-04-27 15:27:52 IST	0000-00-00 00:00:00
Mr Nobody (2009)	2015-04-27 15:30:44 IST	0000-00-00 00:00:00
New folder	2015-04-27 15:49:06 IST	0000-00-00 00:00:00
Psycho (1960)	2015-04-27 15:32:46 IST	0000-00-00 00:00:00
Triangle [2009] 720p	2015-04-27 15:34:32 IST	0000-00-00 00:00:00
_WRD0002.tmp	2015-04-29 19:11:46 IST	0000-00-00 00:00:00
imp.docx	2015-04-29 19:11:46 IST	0000-00-00 00:00:00
Prac01.docx	2015-04-27 15:01:16 IST	0000-00-00 00:00:00
prac02.docx	2015-04-27 16:21:18 IST	0000-00-00 00:00:00
\$FAT1	0000-00-00 00:00:00	0000-00-00 00:00:00

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

File View Tools Window Help

Close Case Add Data Source Generate Report

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
- All (44)
- MB File Size
- MB 50 - 200MB (4)
- MB 200MB - 1GB (2)
- MB 1GB+ (1)
- Results

Directory Listing /img_UinxImageFile.dd

Table Thumbnail

Name	Modified Time	Change Time
\$OrphanFiles	0000-00-00 00:00:00	0000-00-00 00:00:00
\$Unalloc	0000-00-00 00:00:00	0000-00-00 00:00:00
autorun.inf	2015-04-27 15:49:06 IST	0000-00-00 00:00:00
comp forensics softwares	2015-04-27 15:27:52 IST	0000-00-00 00:00:00
Mr Nobody (2009)	2015-04-27 15:30:44 IST	0000-00-00 00:00:00
New folder	2015-04-27 15:49:06 IST	0000-00-00 00:00:00
Psycho (1960)	2015-04-27 15:32:46 IST	0000-00-00 00:00:00
Triangle [2009] 720p	2015-04-27 15:34:32 IST	0000-00-00 00:00:00
_WRD0002.tmp	2015-04-29 19:11:46 IST	0000-00-00 00:00:00
imp.docx	2015-04-29 19:11:46 IST	0000-00-00 00:00:00
Prac01.docx	2015-04-27 15:01:16 IST	0000-00-00 00:00:00
prac02.docx	2015-04-27 16:21:18 IST	0000-00-00 00:00:00
\$FAT1	nnnn-nn-nn nn-nn-nn	nnnn-nn-nn nn-nn-nn



Case 2 - Autopsy 3.1.2

File View Tools Window Help

Close Case Add Data Source Generate Report

Directory Listing File Search Results 2

Filename Search Results:

Name	Location	Modified Time	Change Time	Access Time	Created Time	Size
X _recious.img	/img_UinxImageFile.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	12845
X _ample1.img	/img_UinxImageFile.dd/\$OrphanFiles/_ample1.img	2015-03-21 06:36:18 IST	0000-00-00 00:00:00	2015-04-27 00:00:00 IST	2015-04-27 16:18:11 IST	14745

Case 2 - Autopsy 3.1.2

File View Tools Window Help

Close Case Add Data Source Generate Report

Directory Listing File Search Results 2

Filename Search Results:

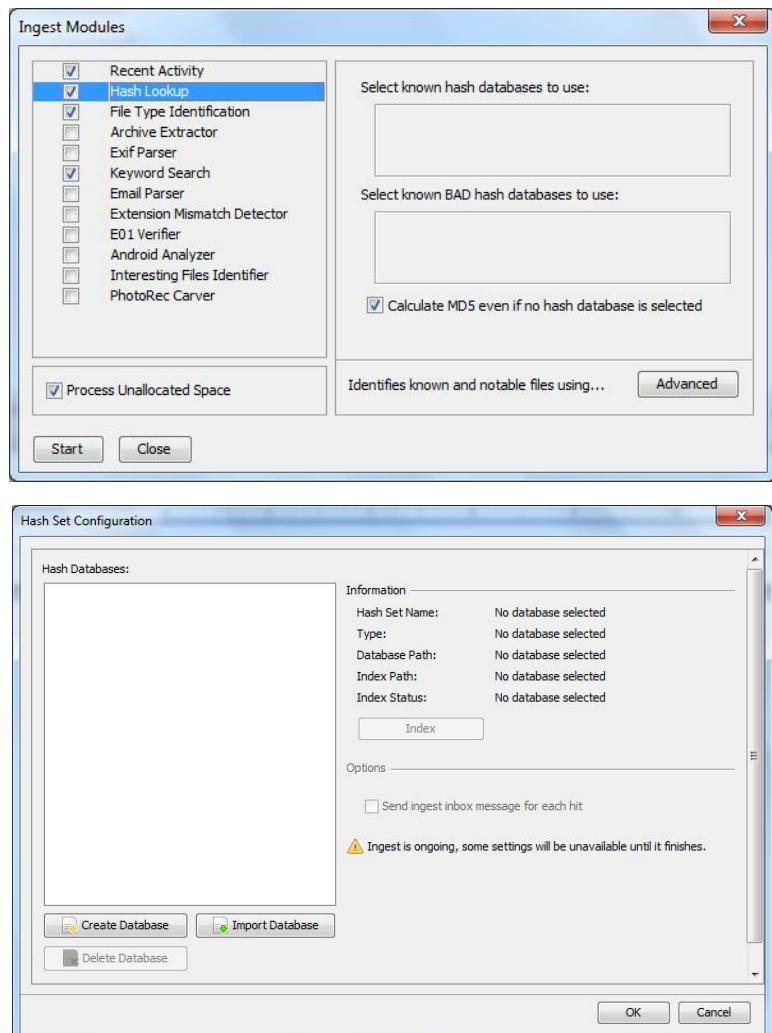
Name	Location	Modified Time	Change Time	Access Time	Created Time	Size
X _recious.img	/img_UinxImageFile.dd/\$OrphanFiles/_recious.img	2015-03-21 06:57:38 IST	0000-00-00 00:00:00	2015-04-27 00:00:00 IST	2015-04-27 16:17:51 IST	12845
X _ample1.img	/img_UinxImageFile.dd/\$OrphanFiles/_ample1.img	2015-03-21 06:36:18 IST	0000-00-00 00:00:00	2015-04-27 00:00:00 IST	2015-04-27 16:18:11 IST	14745

Views

- File Types
- Recent Files
 - Final Day (3)
 - Final Day - 1 (0)
 - Final Day - 2 (2)
 - Final Day - 3 (0)
 - Final Day - 4 (0)

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)



The main interface shows a "Case 2 - Autopsy 3.1.2" title bar with standard menu options: File, View, Tools, Window, Help. The toolbar includes Close Case, Add Data Source, Generate Report, Keyword Lists, and Keyword Search. The main pane displays a "Directory Listing" and "File Search Results 2" tab. On the left, a tree view shows "Email Addresses (257)" expanded, listing various email addresses such as "-@Tp0.DW (3)", "0@TAA.GU (2)", "1cS@WTE.UkDs (1)", etc. The right pane shows a "Table" view for "Email Addresses" with columns "List Name" and "Files with Hits". The results are as follows:

List Name	Files with Hits
-@Tp0.DW (3)	2
0@TAA.GU (2)	2
1cS@WTE.UkDs (1)	1
38@QY.gkF (3)	3
3@Sr.JcO (1)	1
6z@S.RRo (4)	4
8DfF@u.SB (4)	3
A@PQ.Zgl. (3)	3
AOLWelcome@aol.com (3)	3
Addressscagan1934@hotmail.com (1)	1
Adeb@accessdata.com (2)	2
Anatasha@accessdata.com (3)	3
Bad@Evil.Com (1)	1
Baggifrodo@aol.com (15)	15
BagginsFrobaggip@comcast.netP (2)	2
Communicationsonline.communications@comcast.net (1)	1
D@kgh.oD (2)	2
Daleynatasha@accessdata.com (1)	1
Frobobaggip@comcast.net (16)	16
Frobobaggip@hotmail.com (6)	6
Fu9@FuJ.wN (3)	3
Gamgeesamwizgamgee@yahoo.com (1)	1
Guilty@Party.Com (1)	1
Ipu@pr8.FY (3)	1
Lrj@Y.jf (1)	1
MAILER-DAEMON@aol.com (1)	1
MD@es.Ta (2)	2
Namecsagan1934@hotmail.com (1)	1
Party@party.com (1)	1
Samwizgamgee@hotmail.com (16)	16
Stringermark@accessdata.com (1)	1
T@1C.Vdb (3)	3

At the bottom, tabs for Hex, Strings, Metadata, Results, Text, and Media are visible. A status bar at the bottom right indicates "Analyzing files from UnixImageFile.dd" at 100%, "(1 more...)" with a count of 3.

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

Data Sources: UnixImageFile.dd

Views: File Types, Recent Files

File Types: Final Day (3), Final Day - 1 (0), Final Day - 2 (2), Final Day - 3 (0), Final Day - 4 (0)

Table: HTML Properties Open Report

Autopsy Report for case C

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



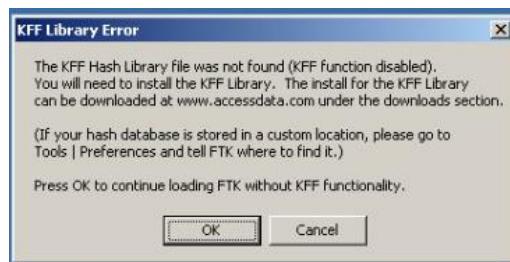
Powered by Autopsy Open Source Digital Forensic Platform, www.autopsy.org

Practical No: 2 A

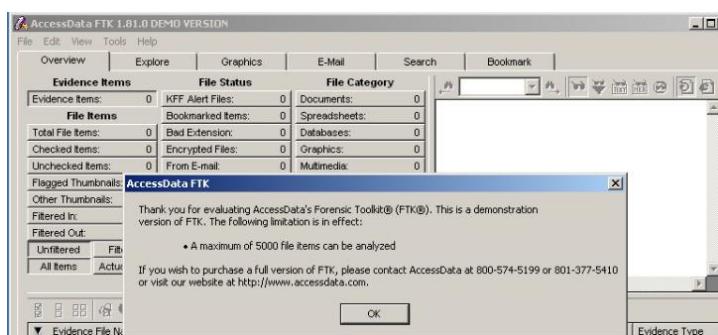
Step 1: Start Forensic Toolkit.



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



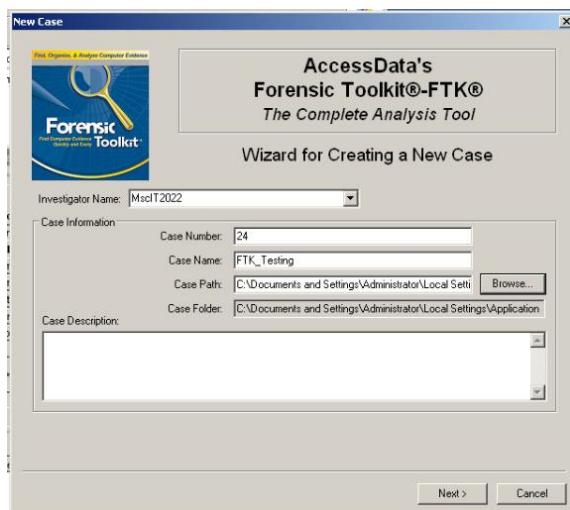
Step 3: click on OK button.



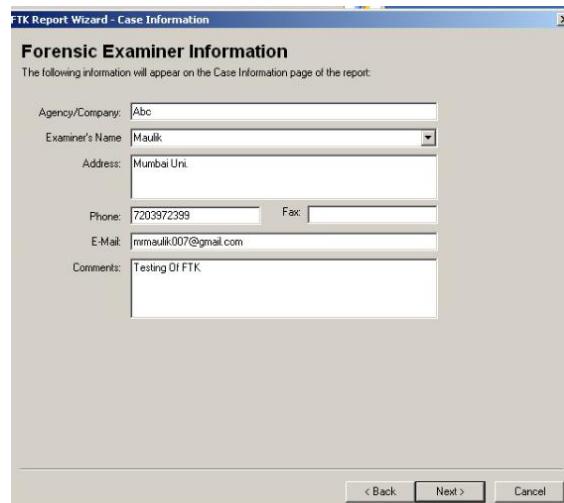
Step 4: Now select Start New Case option and click on ok.



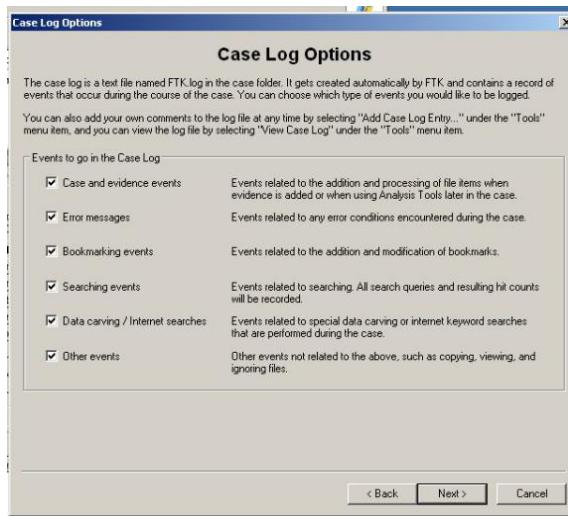
Step 5: Enter the detail for a New case.



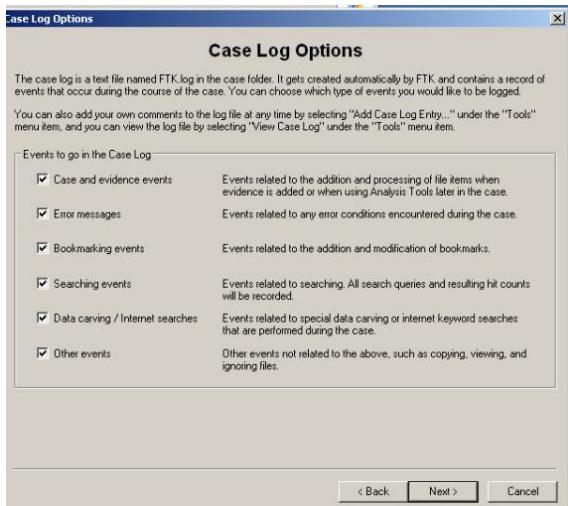
Step 6: Fill the information in Forensic Examiner Information dialog box.



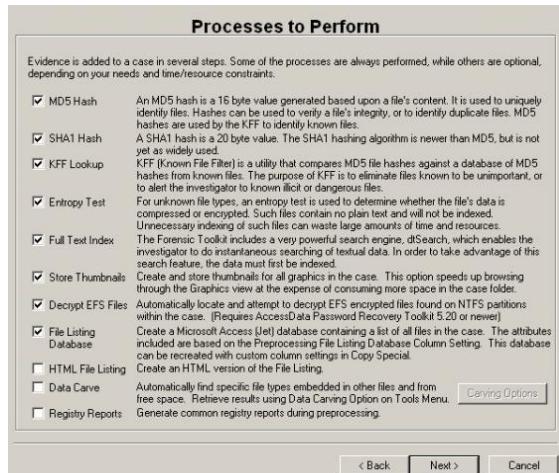
Step 7: leave the default settings and click on next.



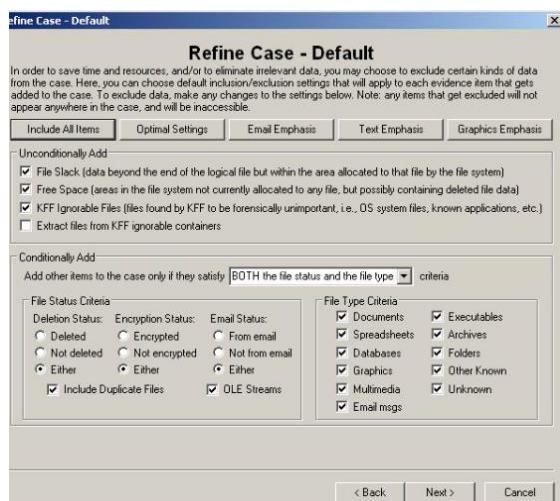
Step 8: Now again leave the default settings and click on next.



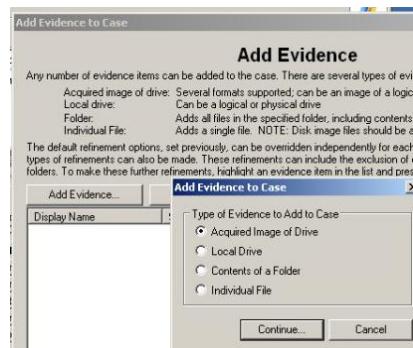
Step 9: In the Refine Case-Default, click the Include All items button and then click Next.



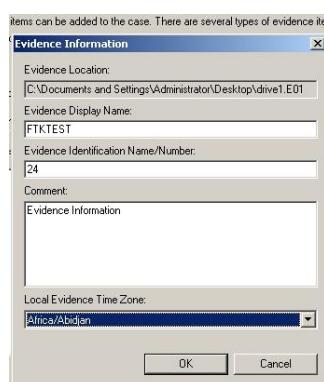
Step 10: In Refine Index-Default, accept the default settings and click Next.

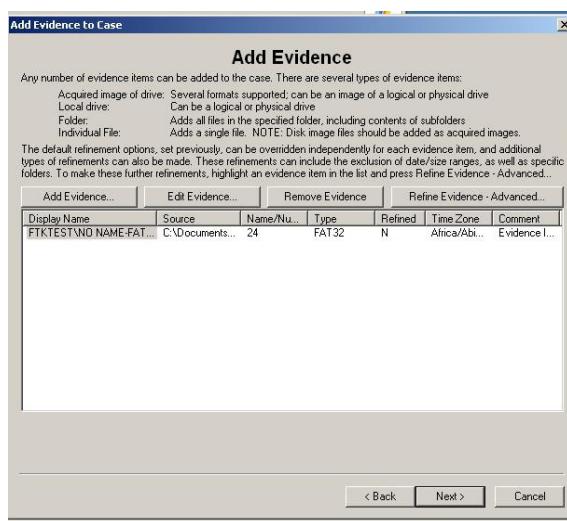
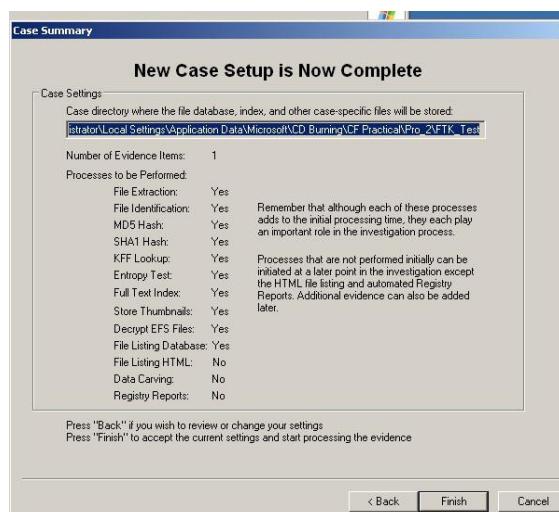
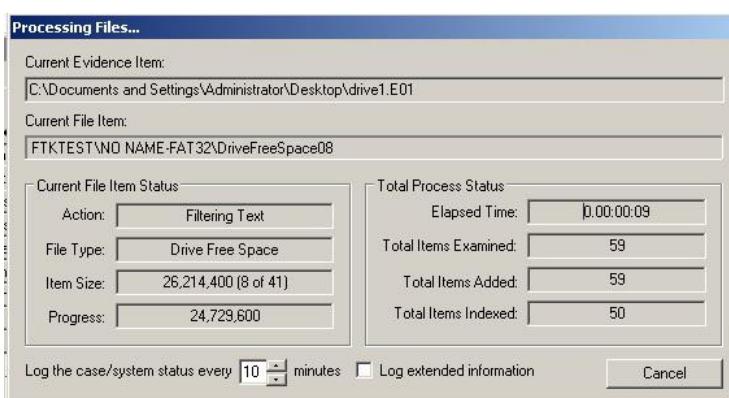


Step 11: Now here Click on add Evidence button.

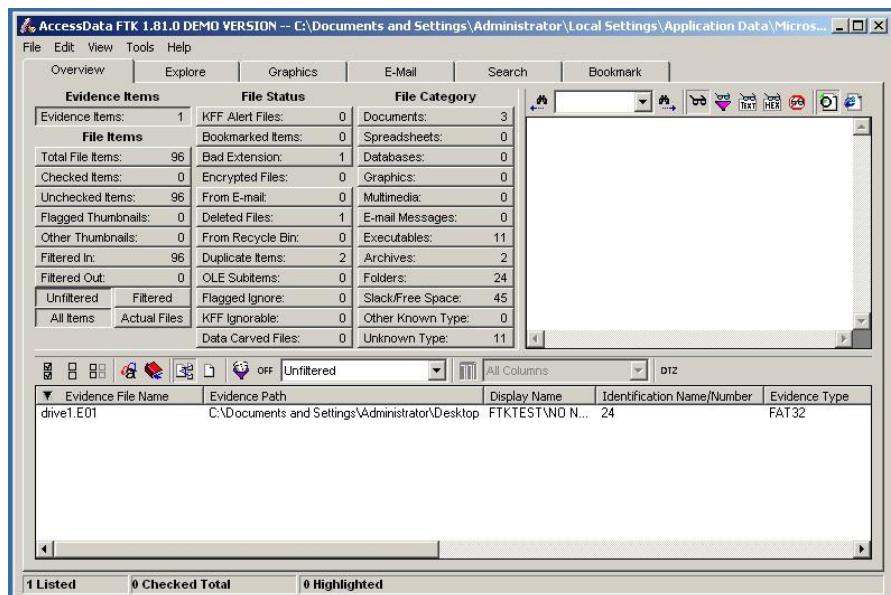


Step 12: Enter Evidence Information and click on OK button.

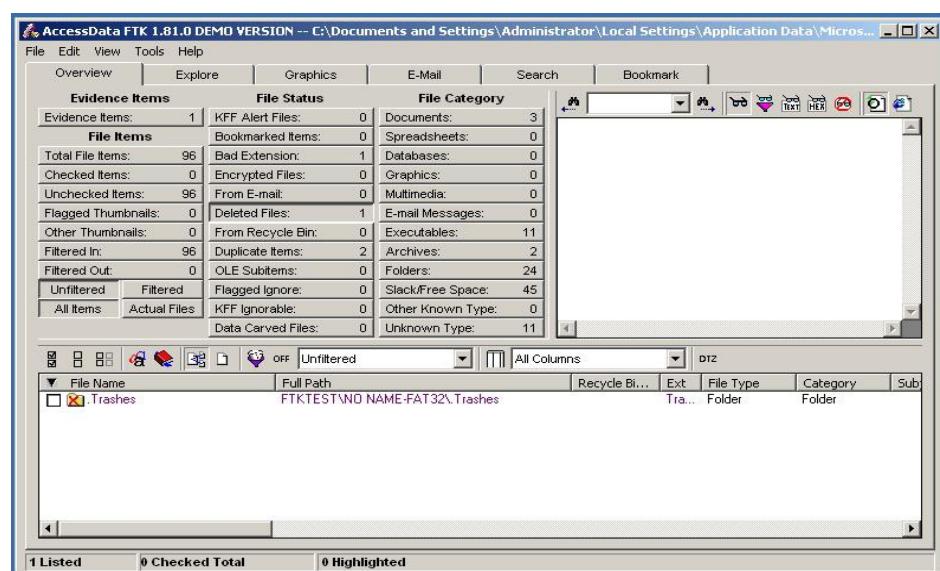


Step 13: Now click on Next.**Step 14:** Click on Finish to initiate the analysis.**Step 15:** Now Processing Will Start.

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.

The screenshot shows the AccessData FTK 1.81.0 DEMO VERSION interface. The title bar displays the path: C:\Documents and Settings\Administrator\Local Settings\Application Data\Micros... . The menu bar includes File, Edit, View, Tools, and Help. Below the menu is a toolbar with icons for Overview, Explore, Graphics, E-Mail, Search, and Bookmark. A large central pane displays a table of evidence items categorized by status and type. The 'File Status' column highlights one item as 'Bad Extension'. The 'File Category' column lists various file types with their counts. At the bottom, a detailed file list table shows a single entry: ndptsp.tsp, located at FTKTEST\NO NAME-FAT32\Windows\system3..., with a file type of Executable File and category of Executable. The status bar at the bottom indicates 1 Listed, 0 Checked Total, and 0 Highlighted.

Evidence Items		File Status	File Category
Evidence Items:	1	KFF Alert Files:	0
		Bookmarked Items:	0
Total File Items:	96	Bad Extension:	1
Checked Items:	0	Encrypted Files:	0
Unchecked Items:	96	From E-mail:	0
Flagged Thumbnails:	0	Deleted Files:	1
Other Thumbnails:	0	From Recycle Bin:	0
Filtered In:	96	Duplicate Items:	2
Filtered Out:	0	OLE Subitems:	0
Unfiltered	Filtered	Flagged Ignore:	0
All Items	Actual Files	KFF Ignorable:	0
		Data Carved Files:	0

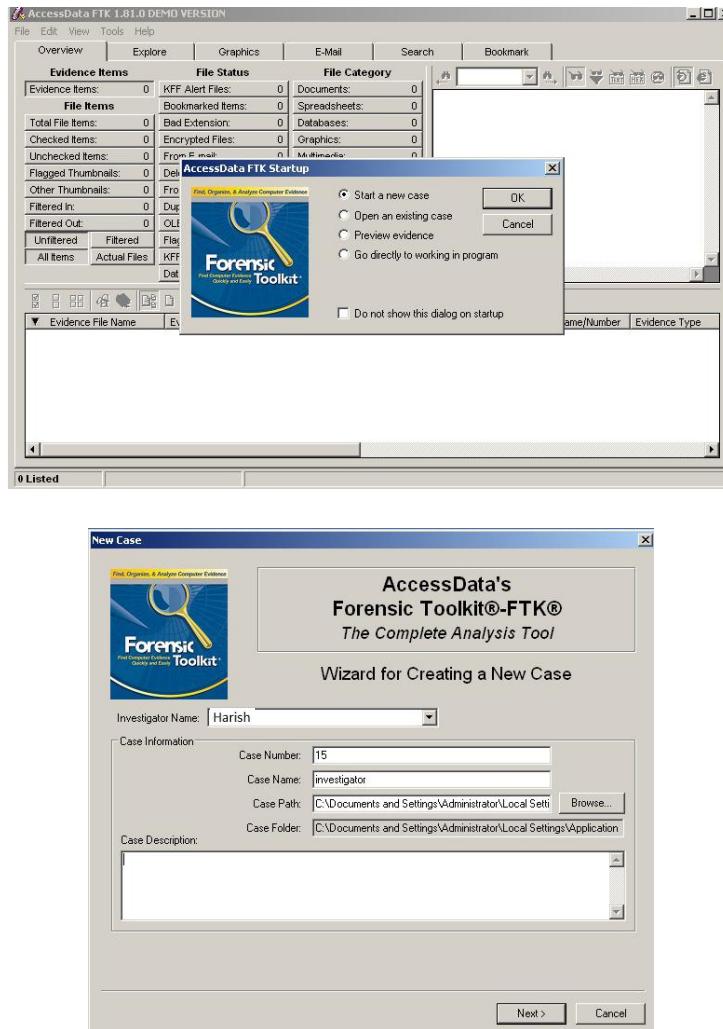
File Name	Full Path	Recycle Bi...	Ext	File Type	Category	Sub
ndptsp.tsp	FTKTEST\NO NAME-FAT32\Windows\system3...		tsp	Executable File	Executable	

1 Listed | 0 Checked Total | 0 Highlighted

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:

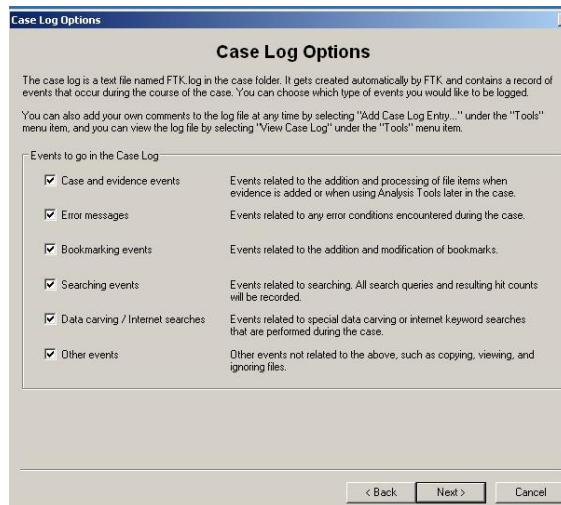
Selecting Case Log Options:

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:

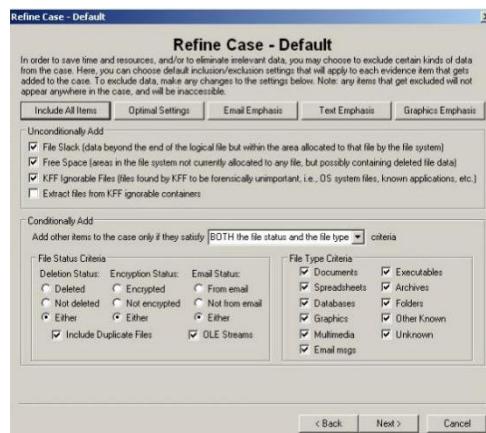
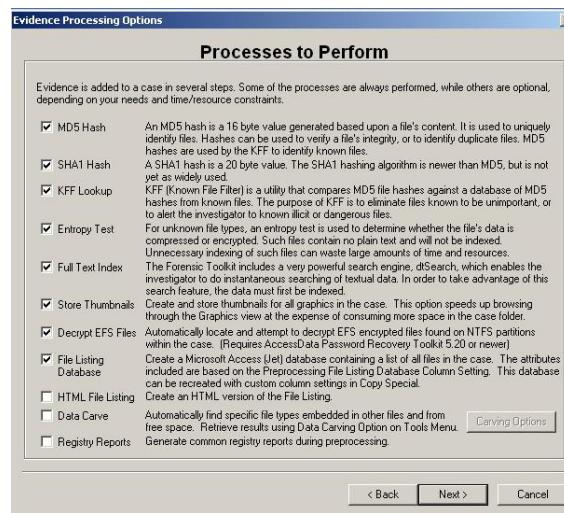
Option	Description
Bookmarking Events	Events related to the addition and modification of bookmarks.
Case and Evidence Events	Events related to the addition and processing of file items when evidence is added or when using the Analysis Tools.
Data Carving/Internet Searches	Events related to data carving or Internet keyword searches that are performed during the case.

Option	Description
Error Messages	Events related to any errors encountered during the case.
Other Events	The following events: <ul style="list-style-type: none"> ♦ 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.

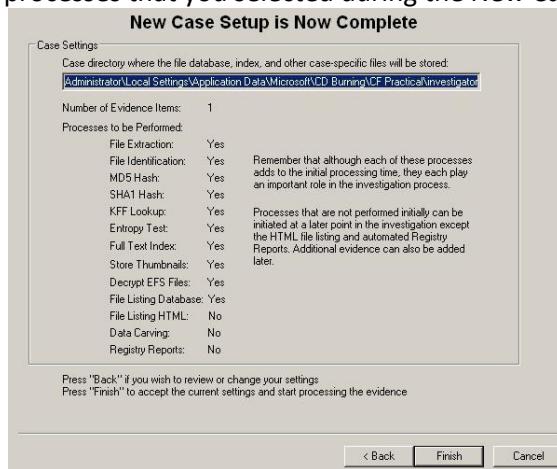


Managing Evidence :

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:

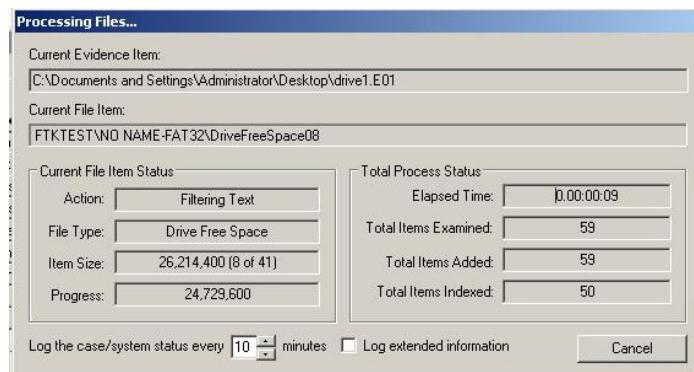
The Case Summary form allows you to review the evidence directory, number of evidence items, and evidence processes that you selected during the New Case Wizard.



Processing the Evidence :

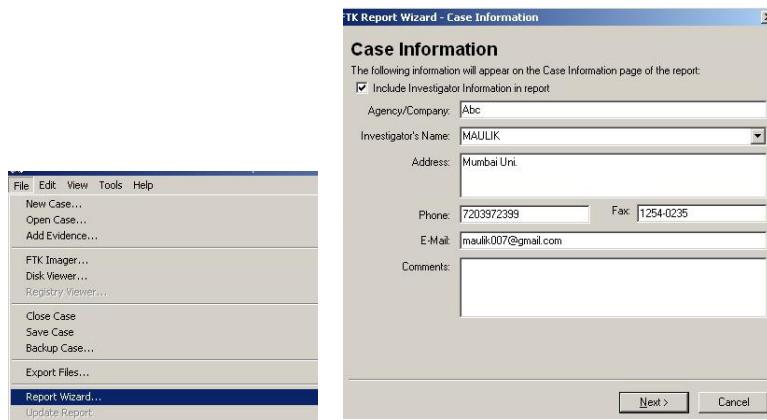
After you click Finish, the Processing Files form appears and displays the status of the processes you selected in the wizard.

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

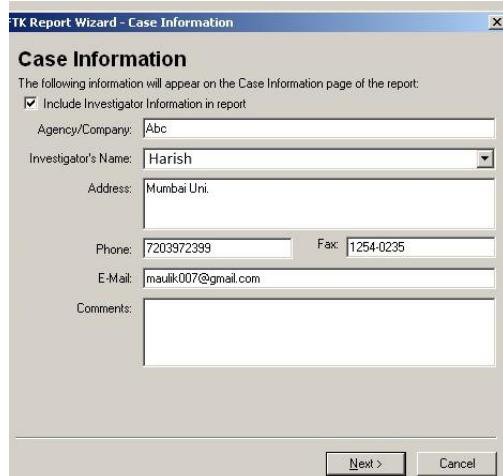


Unfiltered	Filtered	Flagged Ignore	0	Slack/Free Space	0
All Items	Actual Files	KFF Ignorable	0	Other Known Type	0
		Data Carved Files	0	Unknown Type	0
<hr/>					
Evidence File Name	Evidence Path	Display Name	Identification Name/Number	Evidence Type	
Pro_3	C:\Documents and Settings\Administrator\Local ...	Case Study	0532	Contents of a folder	

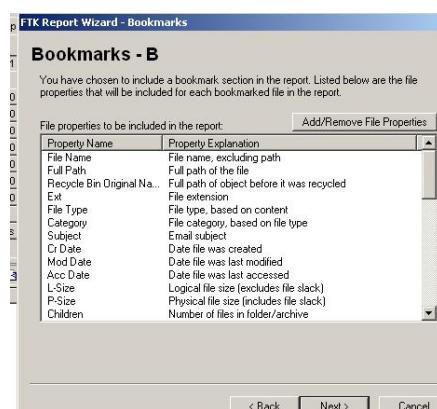
2.The Case Information dialog appears.



3. The Bookmarks-A dialog appears.



4. The Bookmarks-B dialog appears



5. Final Report.....

The screenshot shows the FTK Case Report interface. The main window title is "FTKReport". The URL in the address bar is "file:///C:/Documents and Settings/Administrator/Local Settings/Application Data/Microsoft/CD Burning/CF Practical/investigator/report/index.htm". The left sidebar has links for Case Summary, Case Information (which is selected), File Overview, and Evidence List. The right panel is titled "Case Information" and displays the following data:

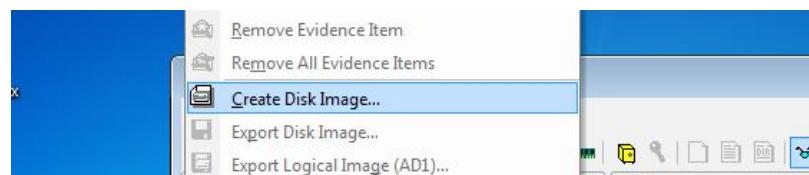
Information Type	Value
FTK Version	Version 1.81.0, build 08.09.25
Case Number	15
Case Location	C:\Documents and Settings\Administrator\Local Settings\Application Data\Microsoft\CD Burning\CF Practical\investigator\
Case Description	
Report Created	Sunday, May 22, 2022 8:06:41 AM
Forensic Examiner	Maulik
Agency	Abc
Address	Mumbai Uni.
Phone	7203972399
Fax	4215-2340

Practical No. 4A

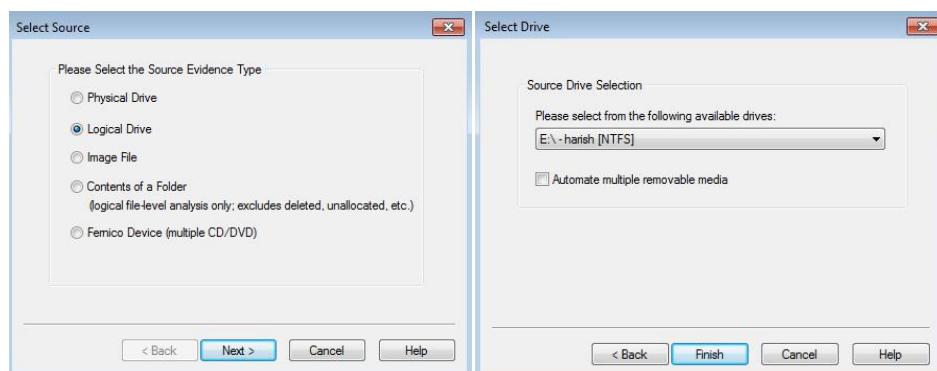
Step 1: Download and install Accessdata FTK Imager on windows virtual machine.



Step 2: Click On File > Create Disk Image



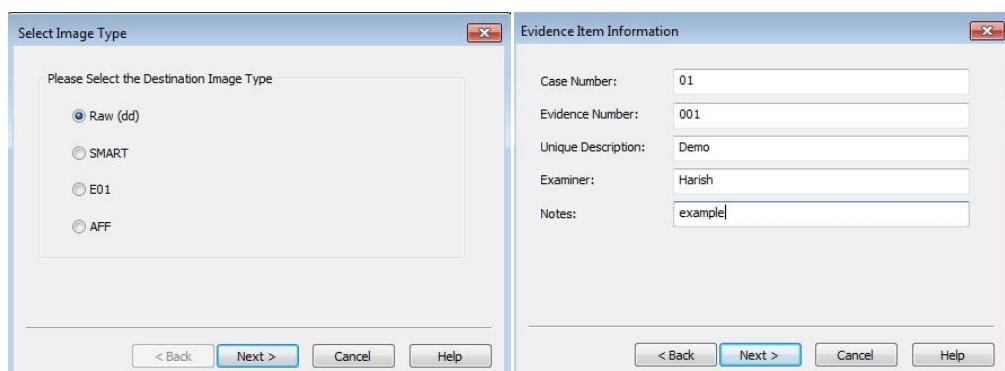
Step 3: Select logical Drive and Select the drive.

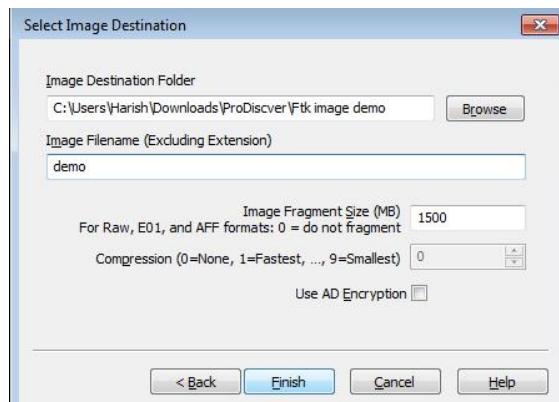


Step 4: Select the Destination for image file to be saved. In that Choose the appropriate destination image type. For our guide, we'll be deploying 'dd' which stands for disk dump.

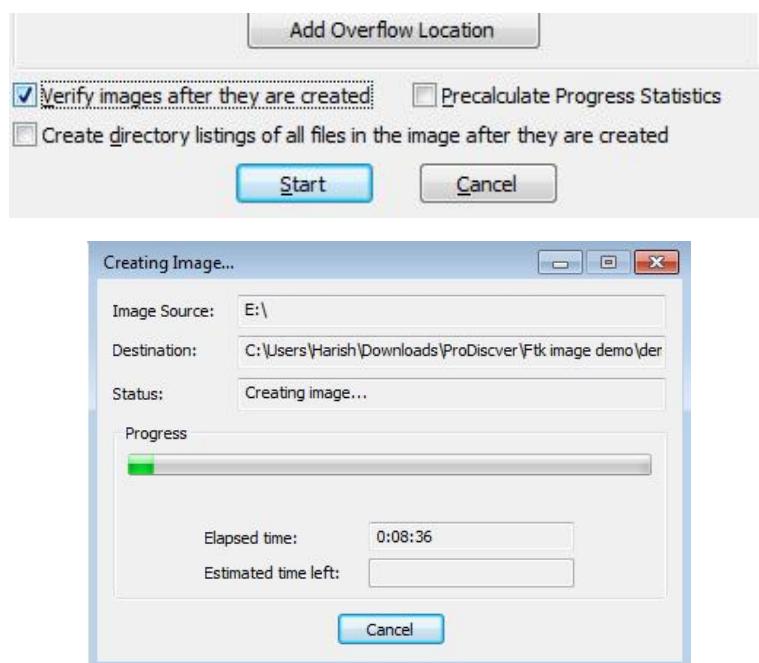
After that we need to enter the case number and other details as show below.

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

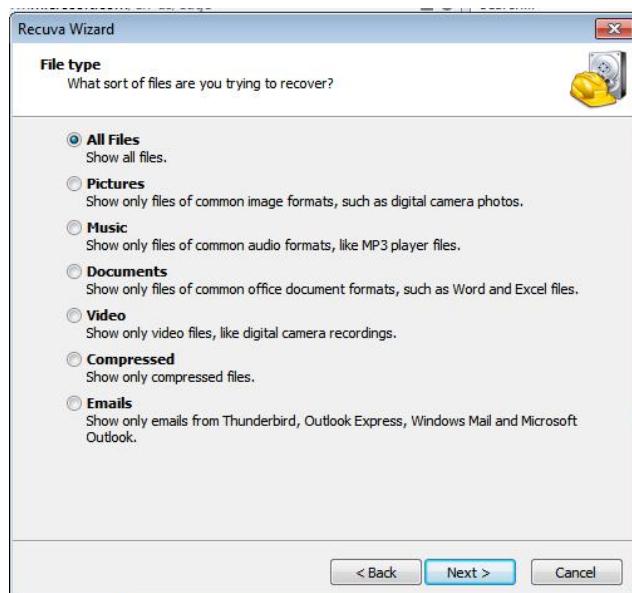
Using Recuva

Step 1: Open the Recuva Software.



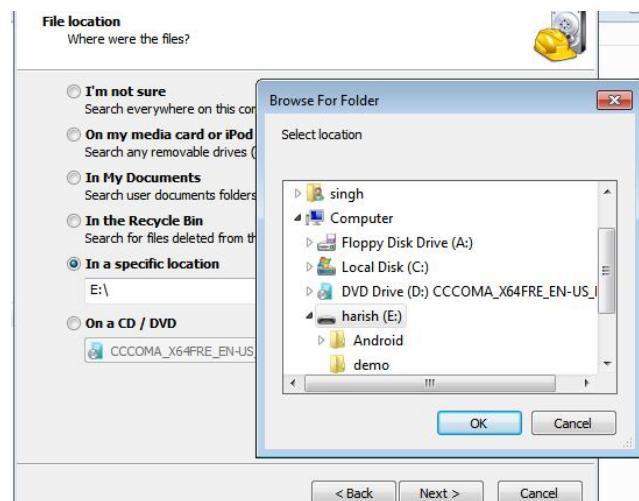
Step 2: click on next.

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.



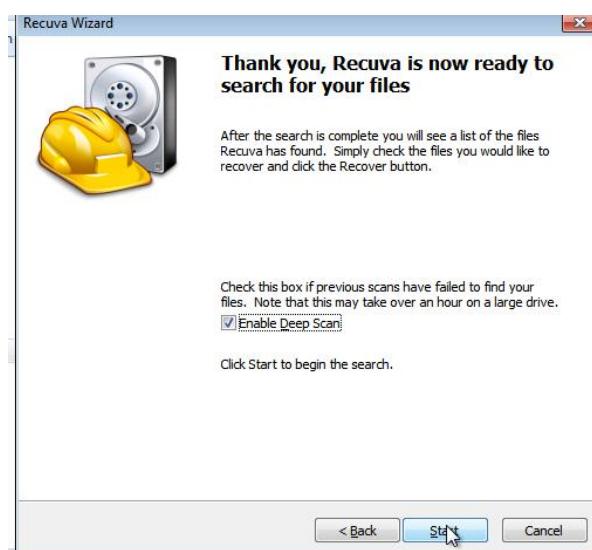
Step 6: check for deep scan for files.

Check this box if previous scans have failed to find your files. Note that this may take over an hour on a large drive.
 Enable Deep Scan

Click Start to begin the search.



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

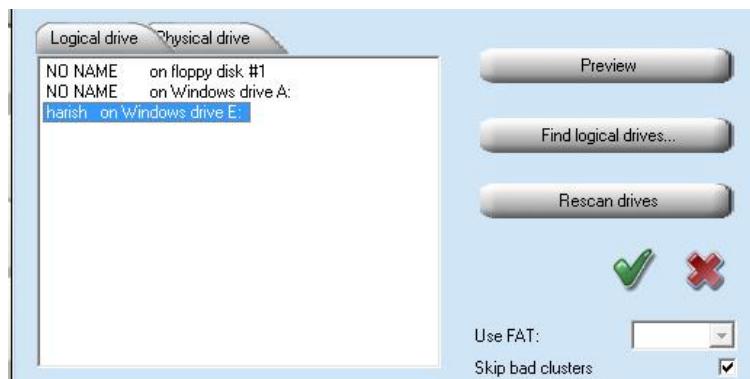


Using PC Inspector File Recovery

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



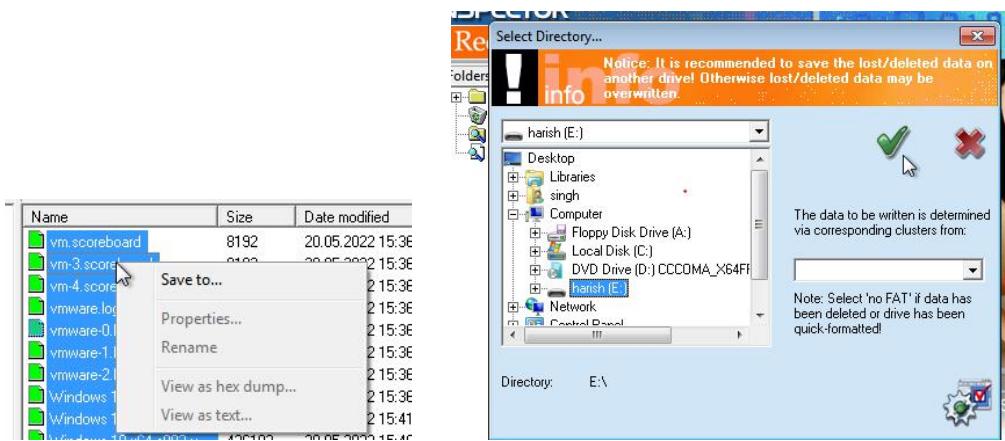
Step 2: Select the drive you want to recover files from. Ensure you've selected the right drive.



Step 3: Scan the selected drive for deleted files.

Folders		Content of 'Deleted'		
		Name	Size	Date modified
[+]	Root	vm.scoreboard	8192	20.05.2022 15:36
	Deleted	vm-3.scoreboard	8192	20.05.2022 15:36
	Lost	vm-4.scoreboard	8192	20.05.2022 15:36
	Searched	vmware.log	324958	20.05.2022 15:36
		vmware-0.log	323179	20.05.2022 15:36
		vmware-1.log	329355	20.05.2022 15:36
		vmware-2.log	426839	20.05.2022 15:36
		Windows 10 x64-77ba27...	214748...	20.05.2022 15:36
		Windows 10 x64-s001.v...	426193...	20.05.2022 15:41
		Windows 10 x64-s002.v...	426193...	20.05.2022 15:49
		Windows 10 x64-s003.v...	399376...	20.05.2022 15:58

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

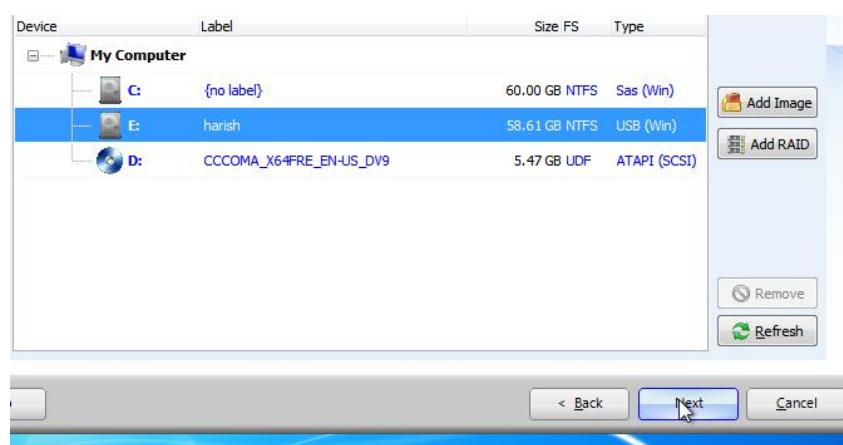


Using Recover my file.

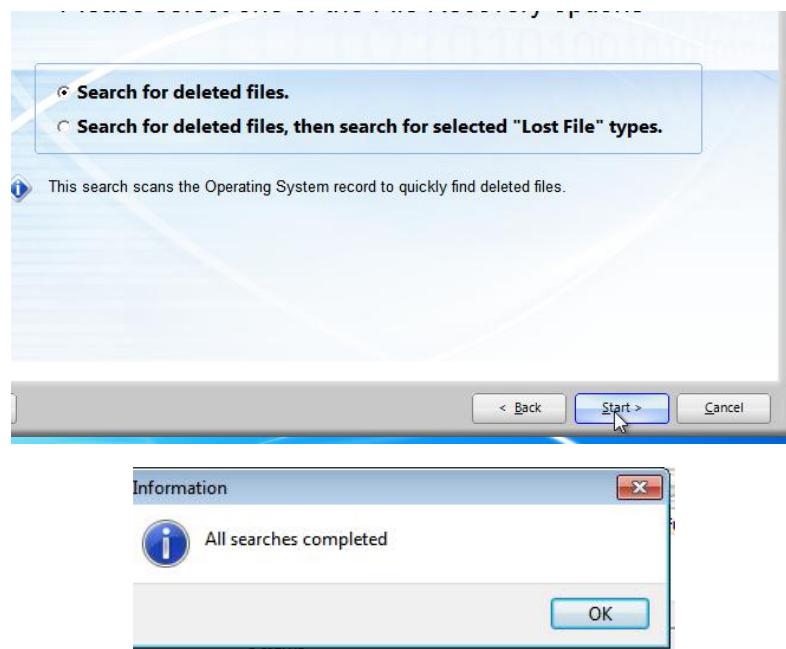
Step 1: Download and install the tool and open then select the type of file you want to recover.



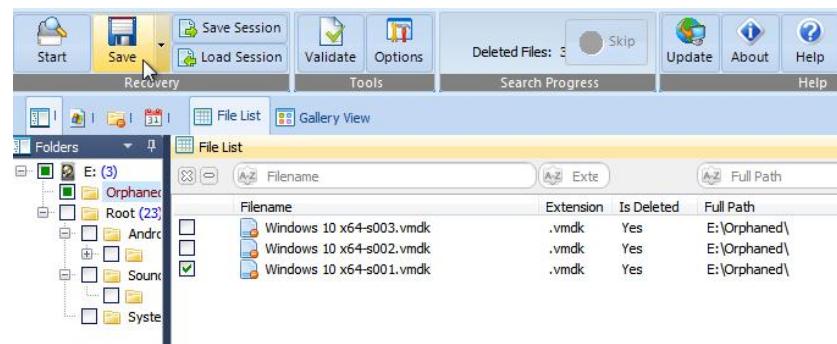
Step 2: Select the file or drive you want to recover.



Step 3: Start search for deleted file.

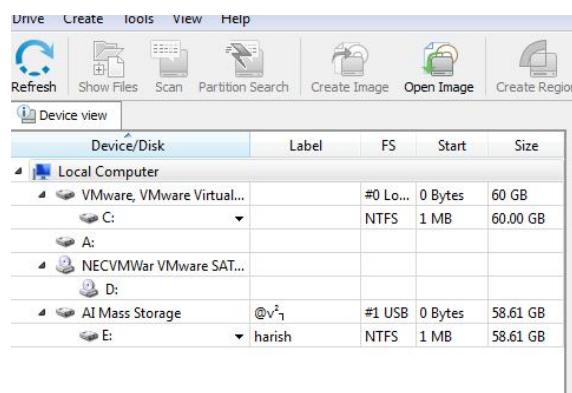


Step 4: Now you can view and recover the deleted file. Then save that original file.

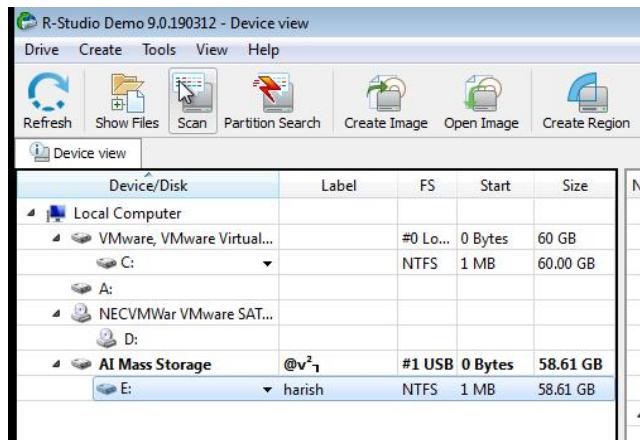


Using R-studio

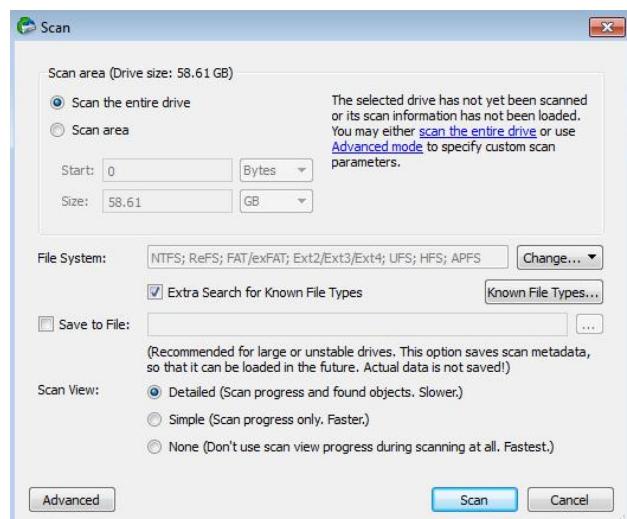
Step 1: Start R-Studio and locate the damaged disk.



Step 2: Scan the damaged disk.



Step 3: View the search results.

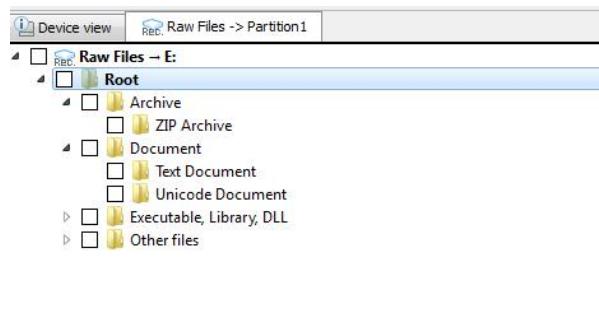


Step 4: Double-click the partition to browse its contents.

Offset in sectors: 0
Size in sectors: 333084

Name	Offset (in sectors)	Size (in sectors)
NTFS MFT Extents	16	8
NTFS Directory Entries	288	8
NTFS Boot Sectors	0	1
FAT Table Entries	304	128
FAT Table Entries	437	371
FAT Table Entries	821	123
Unicode Document	279	6

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



Practical No. 5A

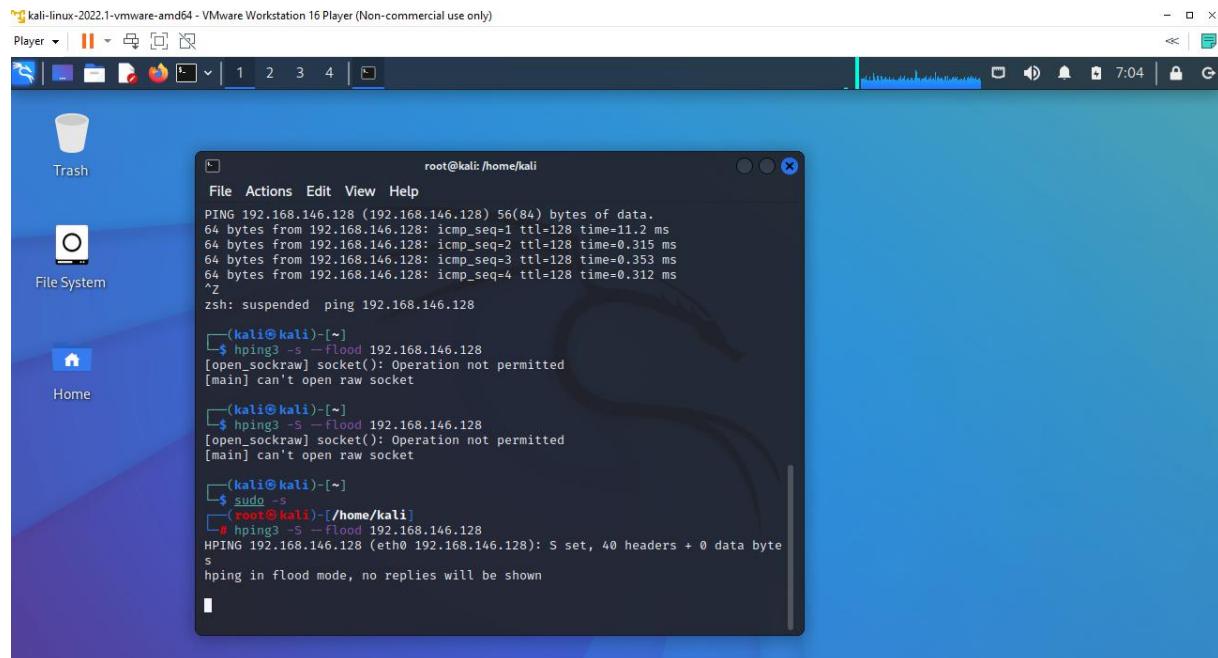
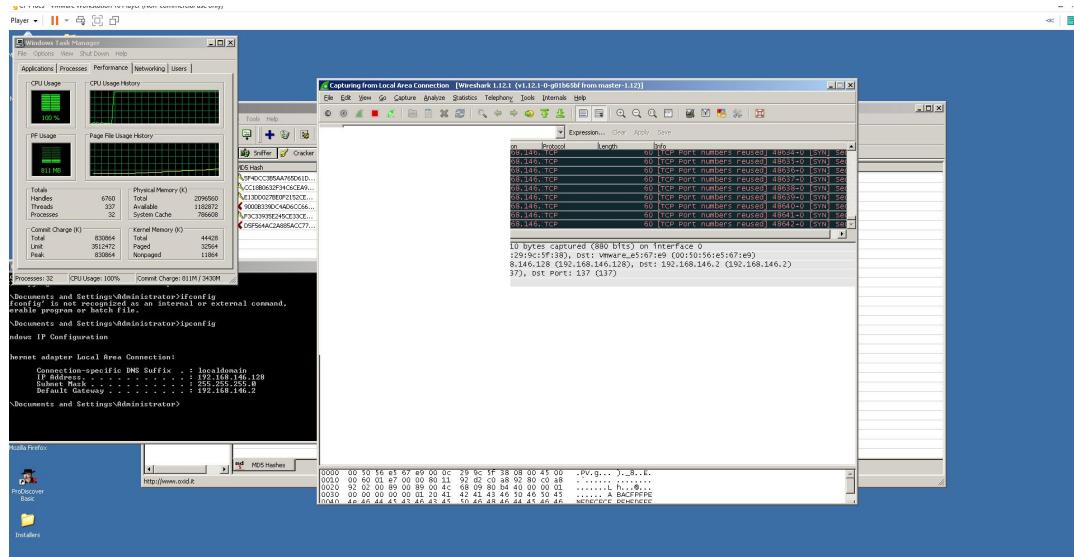
Step 1: check the ip address of virtual testing pc. And open the Wireshark Network Analyser tool.

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

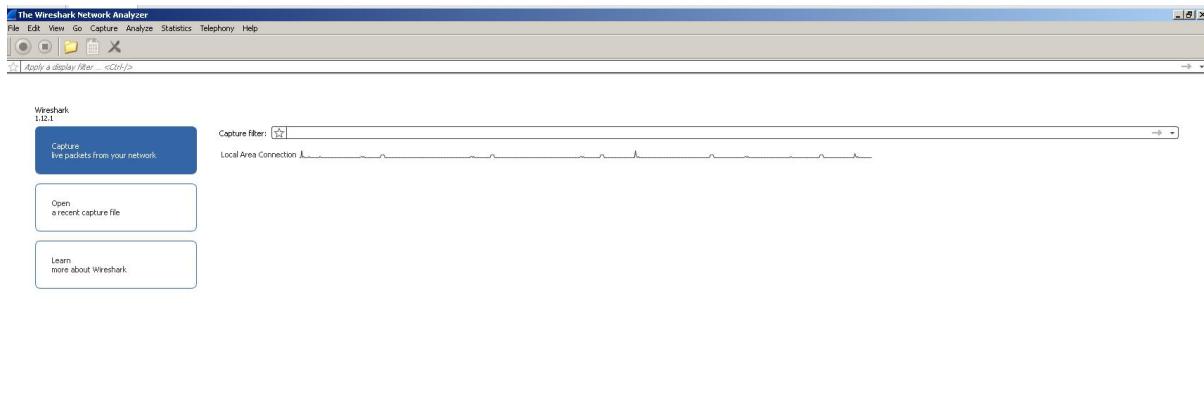
Step 5: hping3 -c 15000 -d 120 -S -w 64 -p 80 --flood --rand-source 192.168.1.159



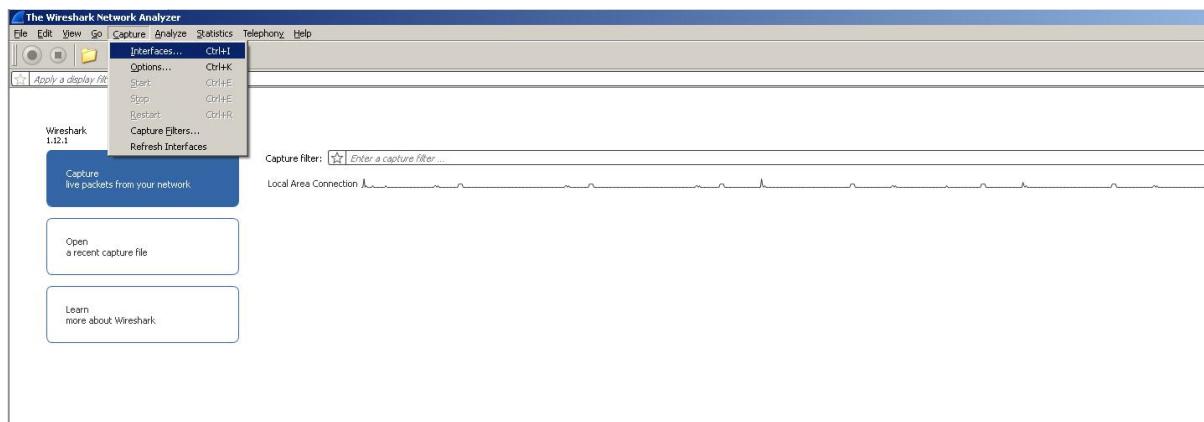
Practical No. 5B

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

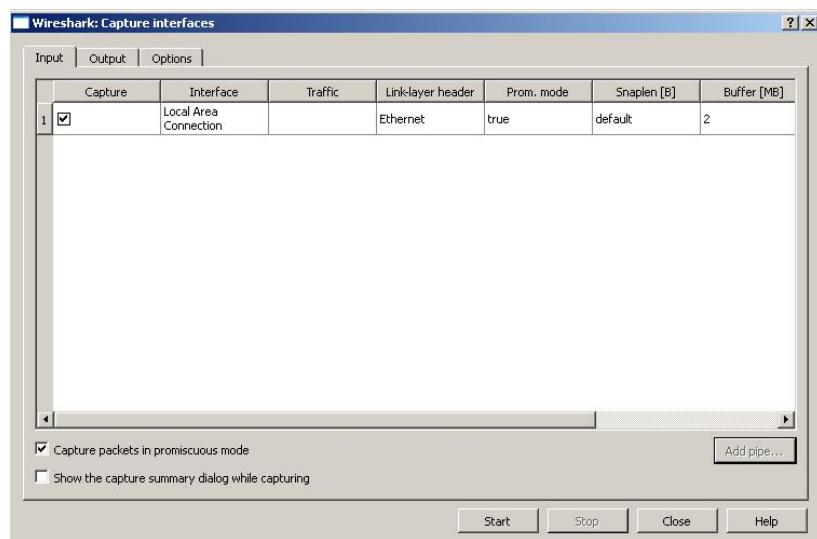
Step 1: open the wireshark

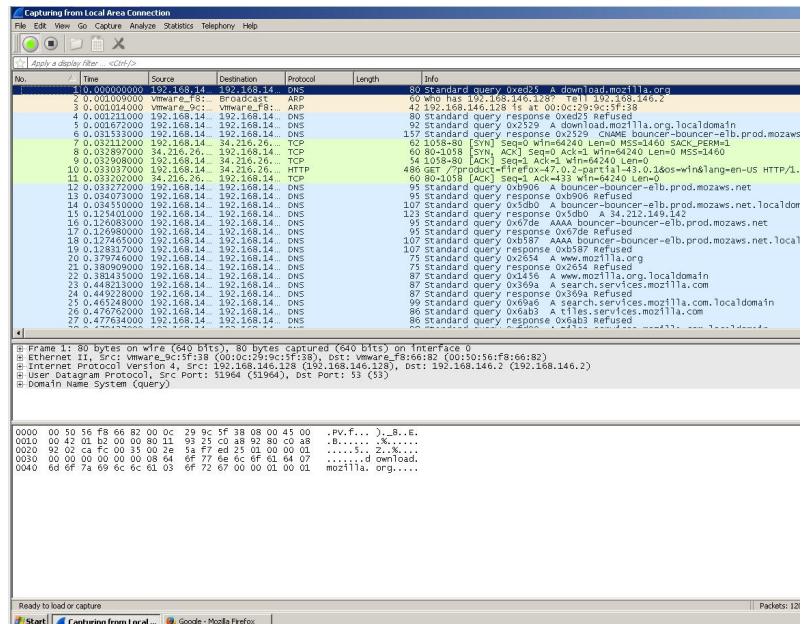


Step 2: On menu bar select Capture. Select interfaces.

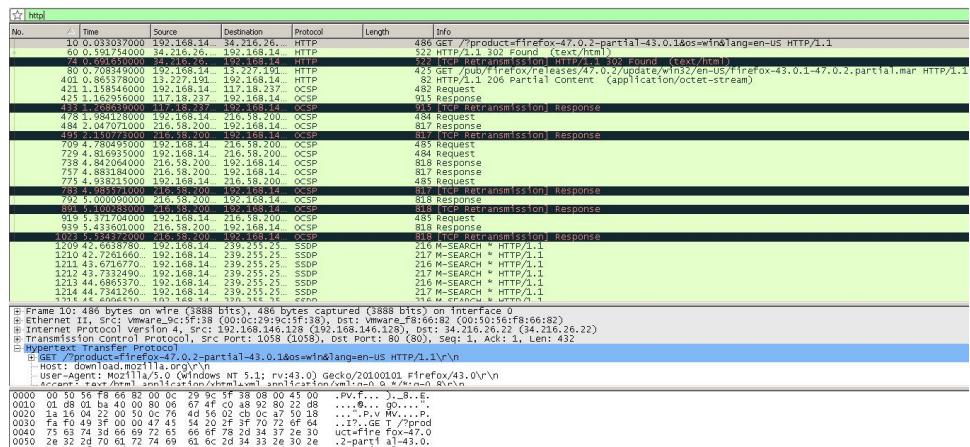


Step 3: Select Once you click on start, then Wireshark starts to capture the packets on that interface.





Step 4: Filter packets with HTTP protocol.



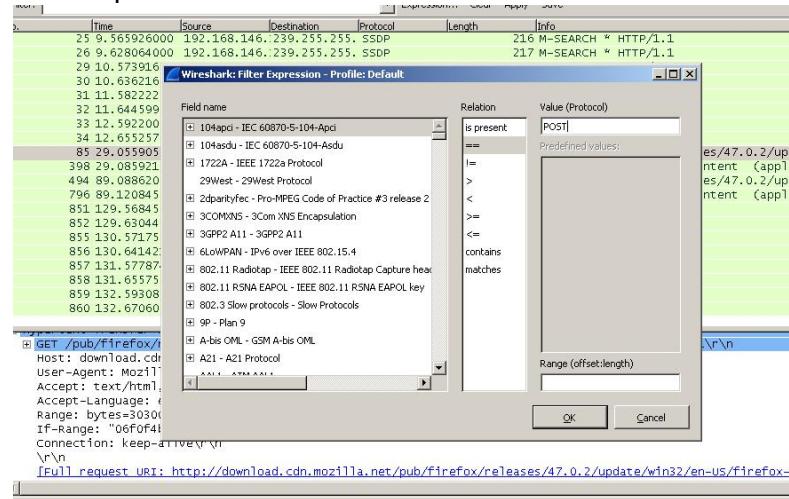
Step 5: A file with only text:

<https://vulms.vu.edu.pk/Courses/SOC101/Downloads/Introduction%20to%20Sociology%20-%20SOC101.pdf>



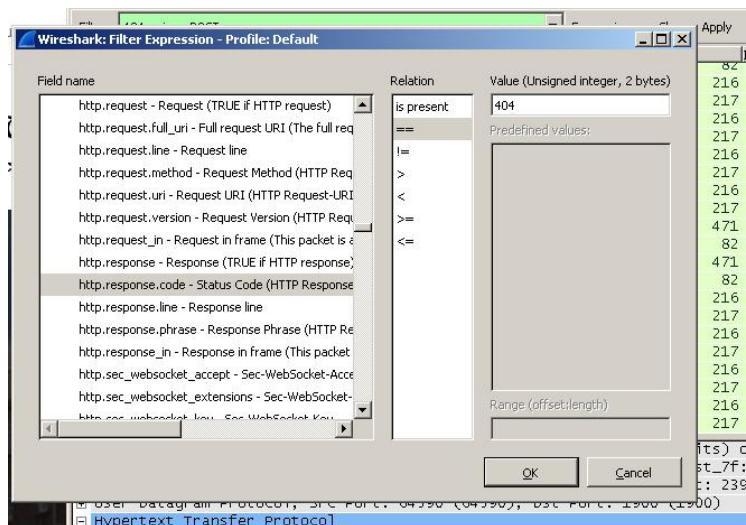
Step 6: Applying different filters using expressions.

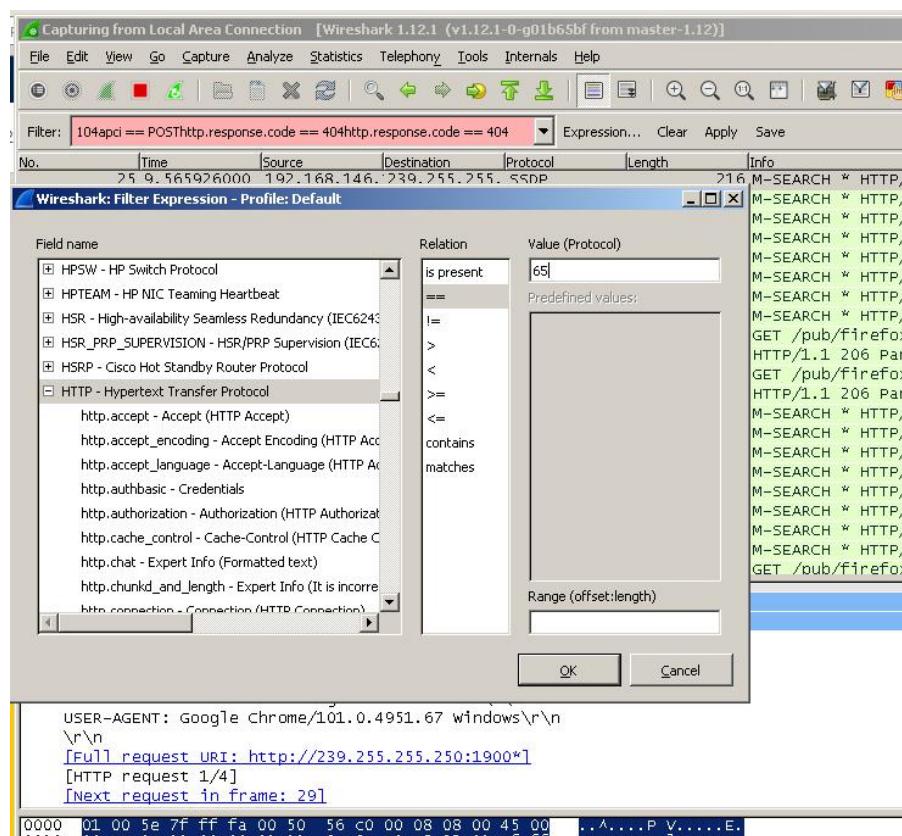
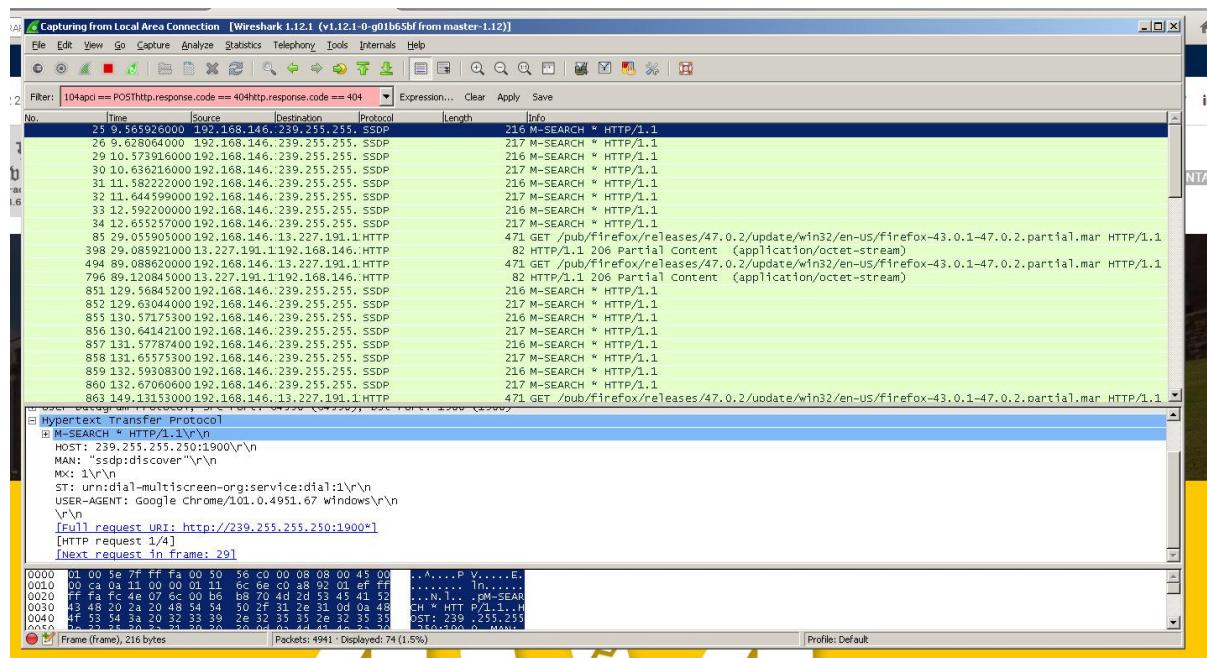
1) Filtering HTTP POST request

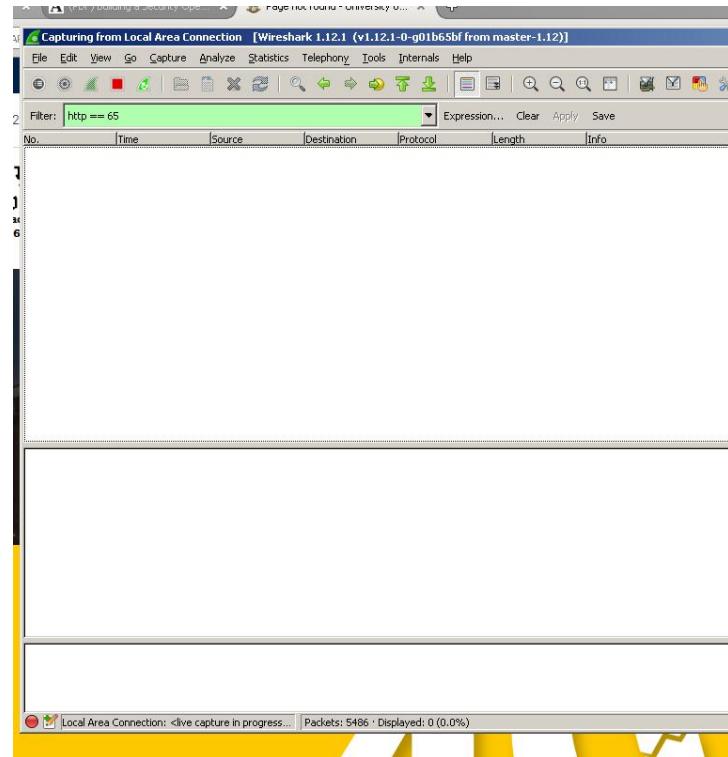


2) Filtering 404 not found error

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







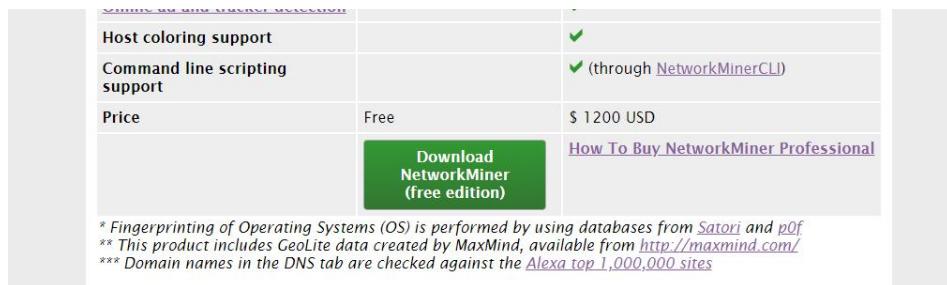
Practical No. 5C

Aim: Using Network Forensic Analysis Tool NetworkMiner

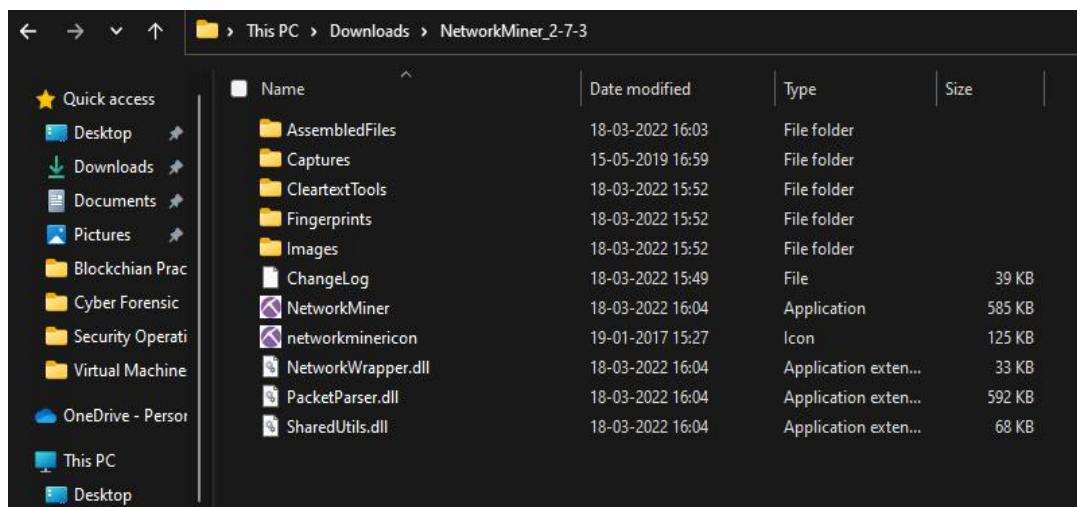
Part 1: Install NetworkMiner.

Step 1: download the zip file from the official web link:

<https://www.netresec.com/?page=NetworkMiner>

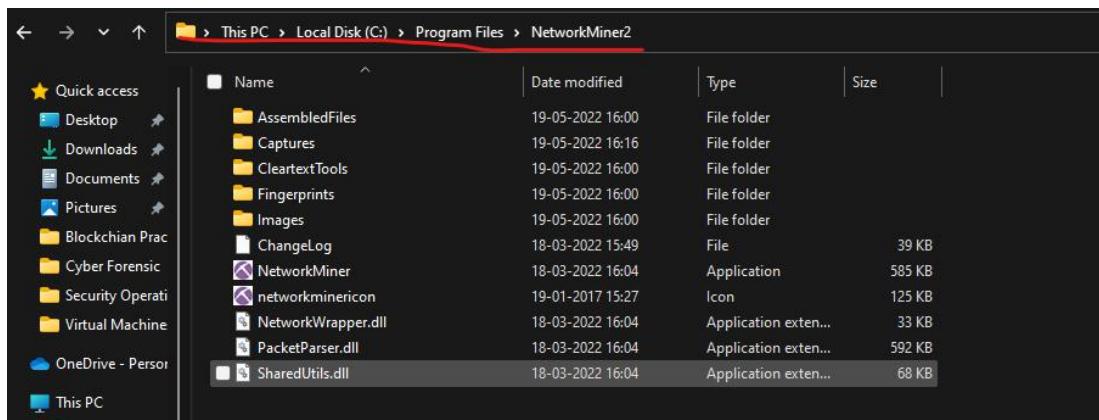


Step 2: after download we need to extract the file. subfiles look like this

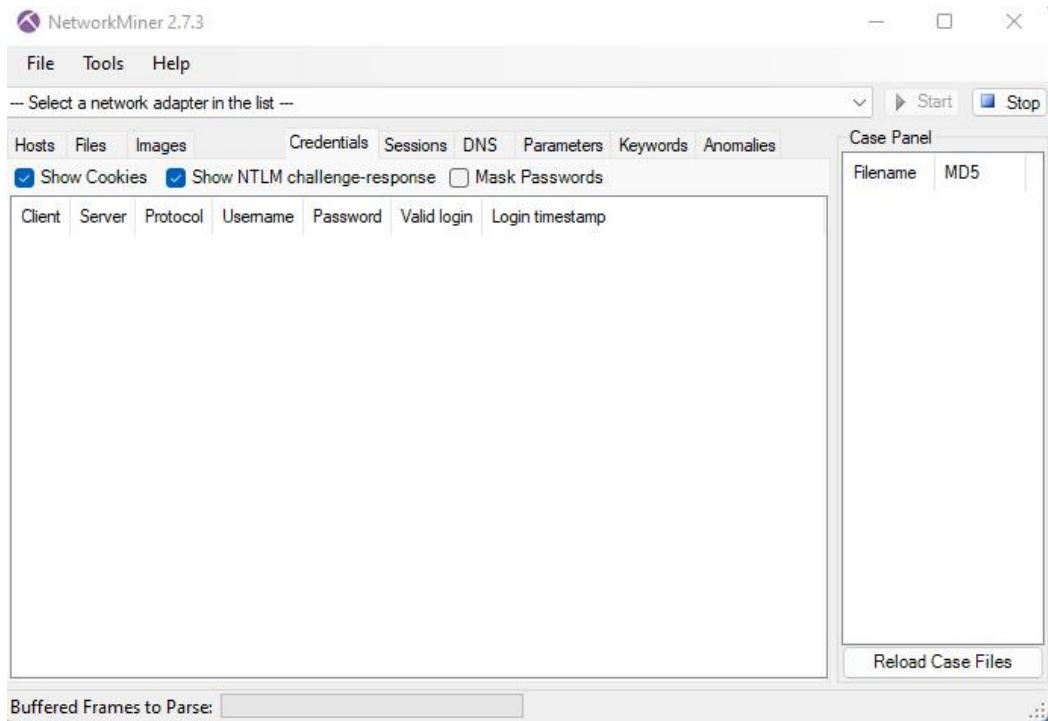


Step 3: Now we need to copy all the files and copy the move to the

C:\Program Files\NetworkMiner2 (Create new folder name NetworkMiner) and paste it that folder.

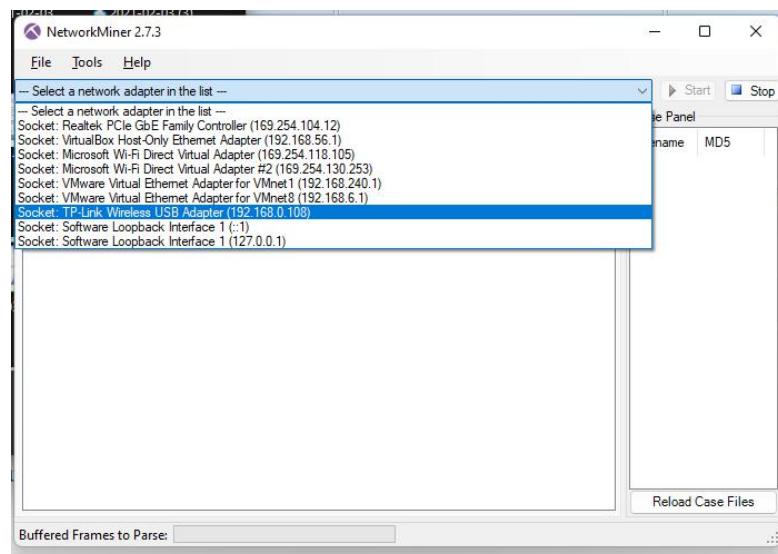


Step 4: Now open the application with open as Admin and interface look like this.

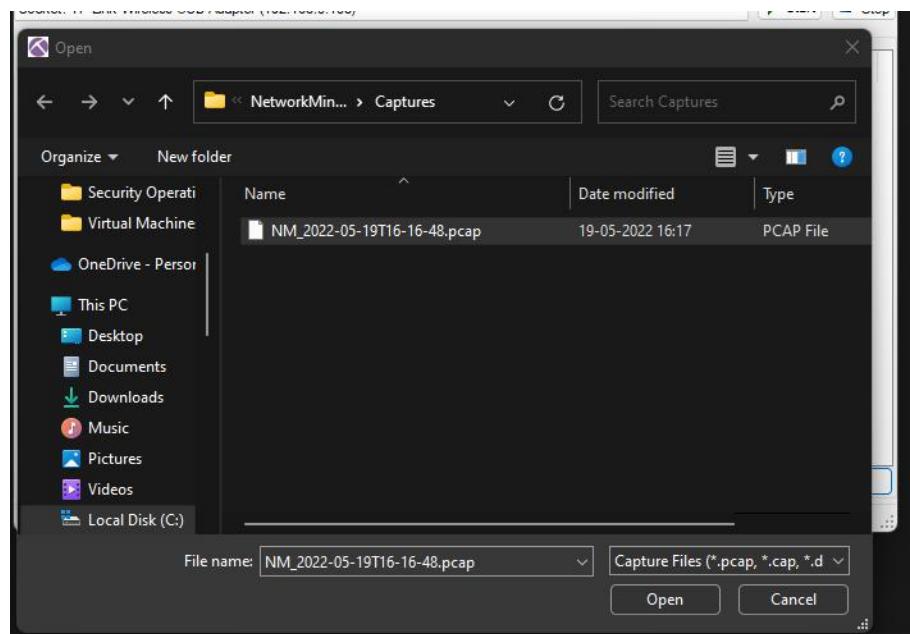


Part 2: Using the NetworkMiner for 1st use.

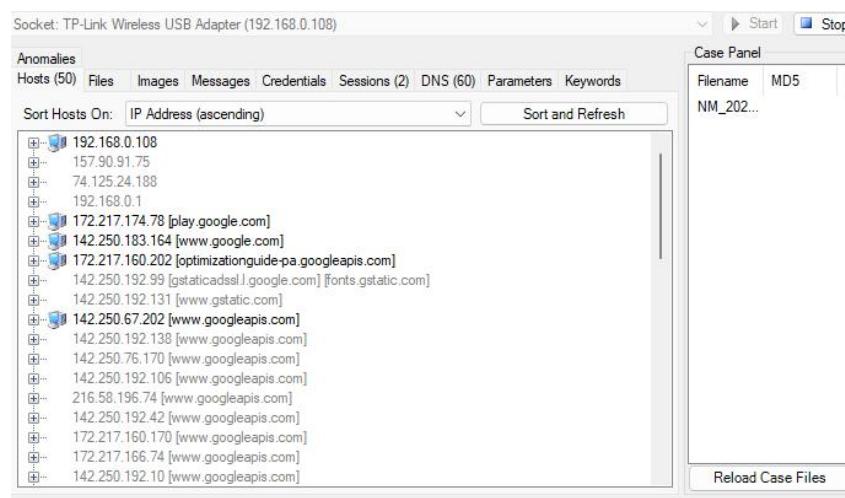
Step1: choose your network adapter from the top of the window.



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.



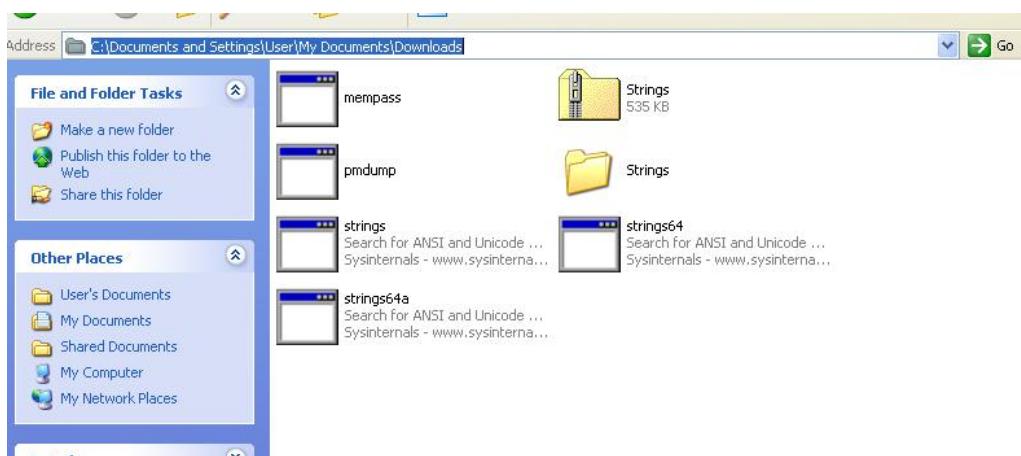
Practical No. 6

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.

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\User>cd C:\Documents and Settings\User\My Documents\Downloads
C:\Documents and Settings\User\My Documents\Downloads>
```

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

```
C:\Documents and Settings\User\My Documents\Downloads>mempass.exe
Please enter your password:crackpass
```

Step 5: then type pmdump

```
C:\Documents and Settings\User\My Documents\Downloads>pmdump
pmdump 1.2 - (c) 2002, Arne Vidstrom <arne.vidstrom@ntsecurity.nu>
- http://ntsecurity.nu/toolbox/pmdump/
Usage: pmdump <pid> <filename>
      - dumps the process memory contents to a file
      pmdump -list
      - lists all running processes and their PID's
C:\Documents and Settings\User\My Documents\Downloads>
```

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

```
1148 - svchost.exe  
1500 - explorer.exe  
1612 - spoolsv.exe  
1692 - UBoxTray.exe  
892 - wsctnfy.exe  
1192 - alg.exe  
632 - cmd.exe  
532 - firefox.exe  
1264 - mempass.exe  
1744 - cmd.exe  
1856 - cmd.exe  
1484 - pmdump.exe
```

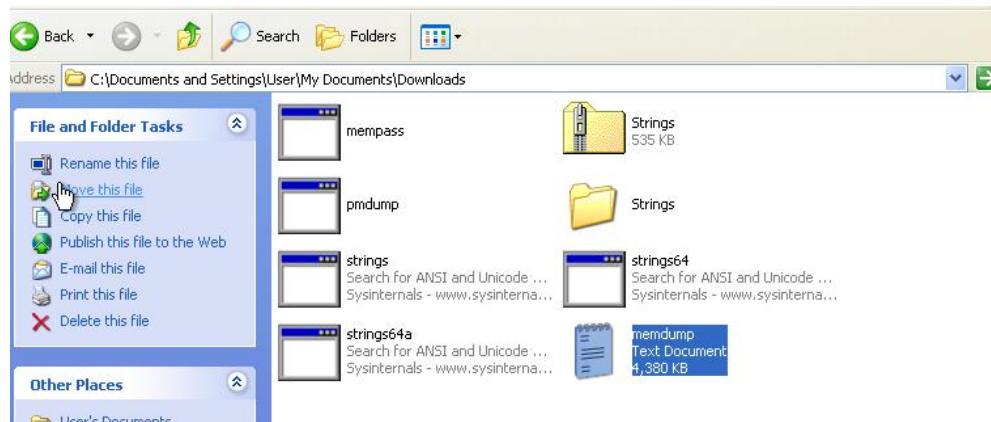
Step 7: type “pmdump 1264 memdump.txt” it will save the log into text file.

```
C:\WINDOWS\system32\cmd.exe
532 - firefox.exe
1264 - mempass.exe
1744 - cmd.exe
1856 - cmd.exe
1484 - pmdump.exe

C:\Documents and Settings\User\My Documents\Downloads>pmdump 1264 memdump.txt
pmdump 1.2 - (c) 2002, Arne Vidstrom <arne.vidstrom@ntsecurity.nu>
              - http://ntsecurity.nu/toolbox/pmdump/

```

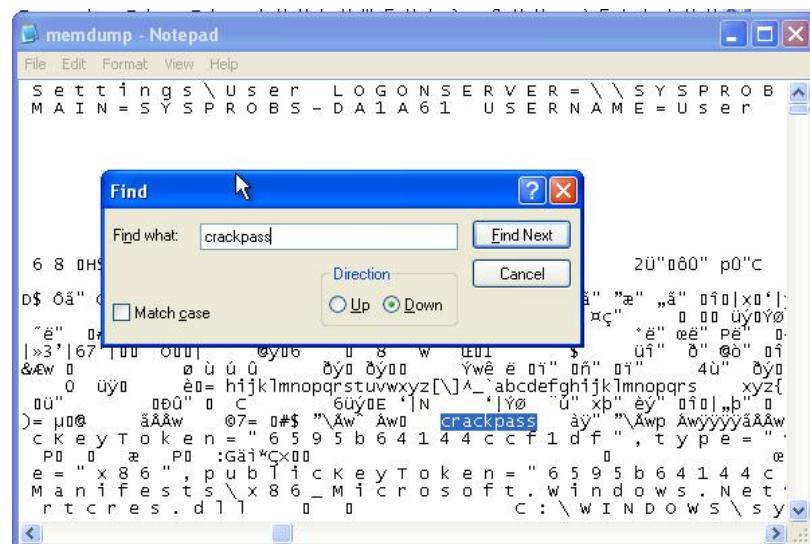
Step 8: txt file is saved in local drive.



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

```
possible to go with this process.  
1484 - pmdump.exe  
C:\Documents and Settings\User\My Documents\Downloads>pmdump 1264 memdump.txt  
pmdump 1.2 - (c) 2002, Arne Vidstrom <arne.vidstrom@ntsecurity.nu>  
- http://ntsecurity.nu/toolbox/pmdump/  
  
C:\Documents and Settings\User\My Documents\Downloads>strings  
Access is denied.  
C:\Document:  
Access is d C:\Documents and Settings\User\My Documents\Downloads\strings.exe X  
C:\Document:  
Access is d C:\Documents and Settings\User\My Documents\Downloads\strings.exe is not a valid  
Win32 application.  
C:\Document:  
Access is d  
C:\Document:  
Access is denied.  
C:\Documents and Settings\User\My Documents\Downloads>strings
```

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



Practical No. 7

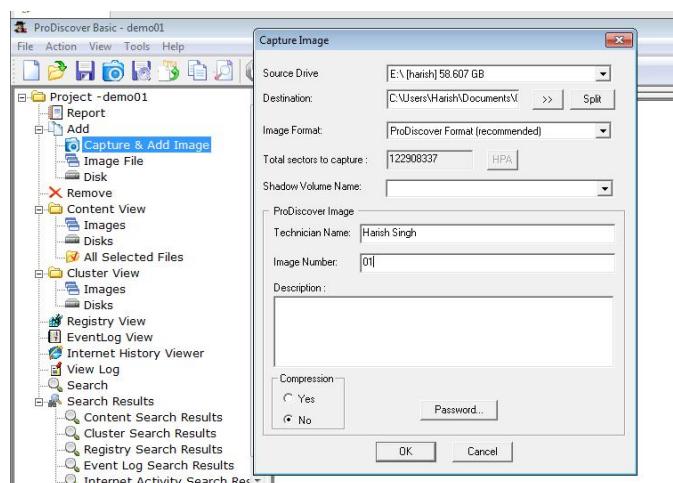
Aim: Using Data Acquisition Tools [ProDiscover Pro]

Solution:

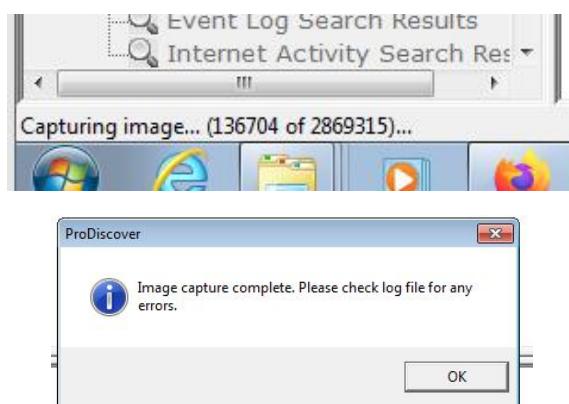
Step 1: open the ProDiscover and add the case number and name.



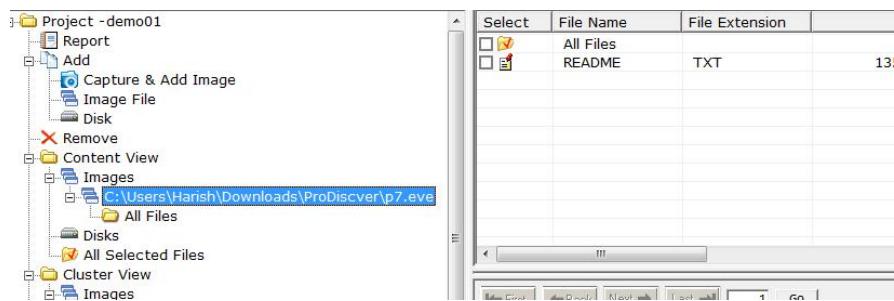
Step 2: open the project and click on Add then select the capture&Add image.fil all the detail as shown in the snap.



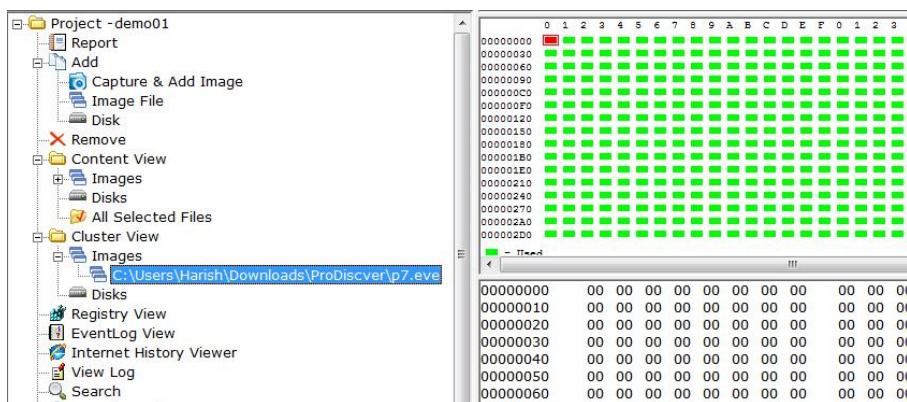
Step 3: After clicking ok image should be created.



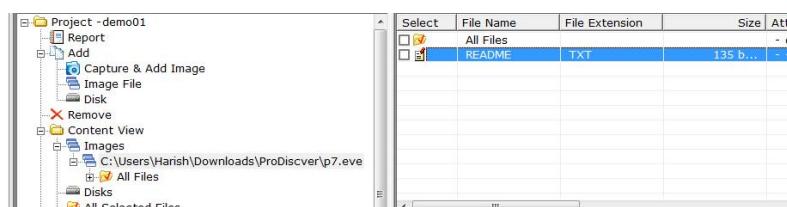
Step 4: After complete of image creation. please check the log file for any error. For that go to content view and click image.



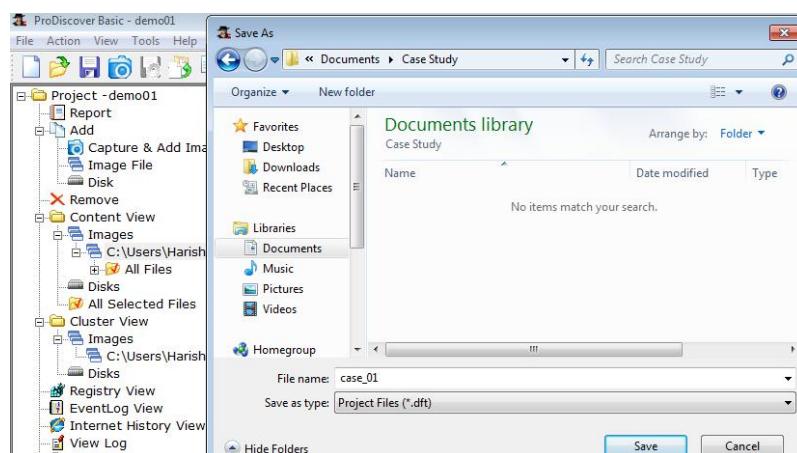
Step 5: you can also see the cluster view file has been created.



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



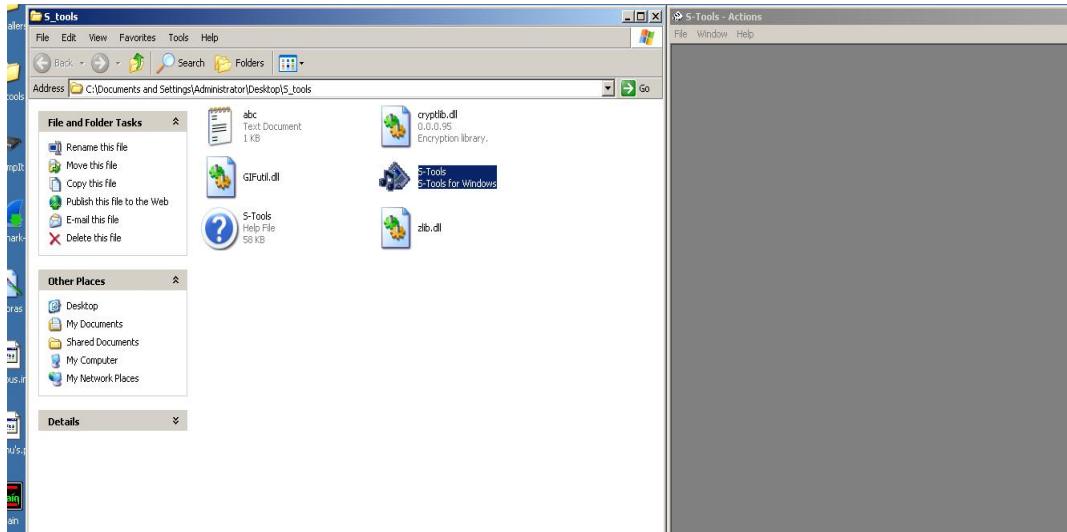
Step 7: At last we need to save the CaseFile.



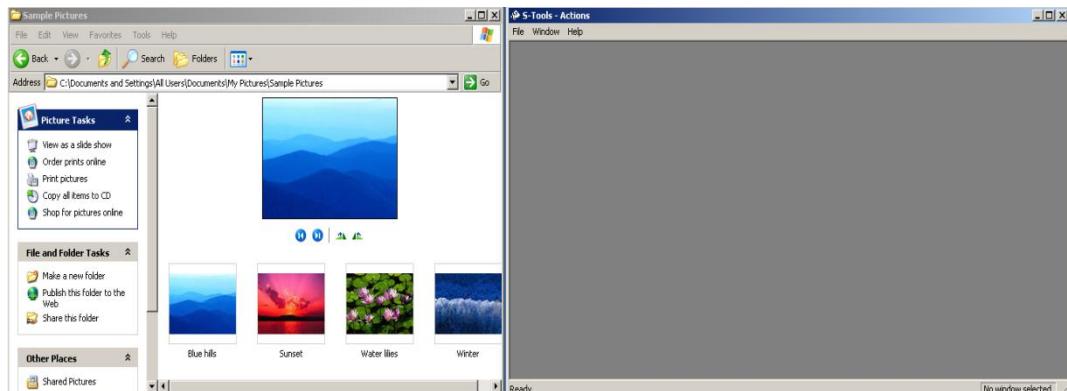
Practical No. 8 A

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



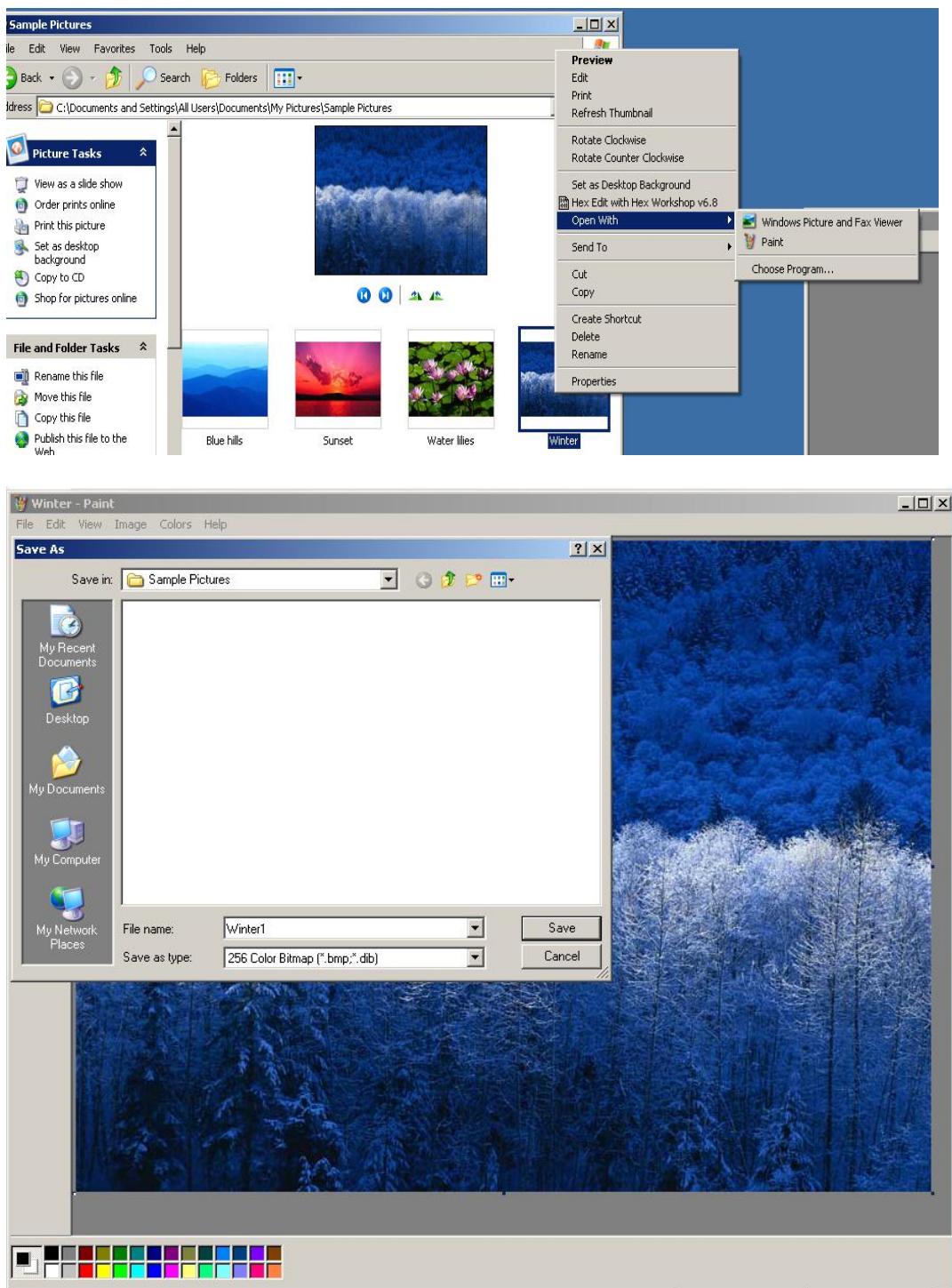
The S-Tools program is a drag and drop software. The files used to create the steganography file can be dragged from the directory into the S-Tools program.

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

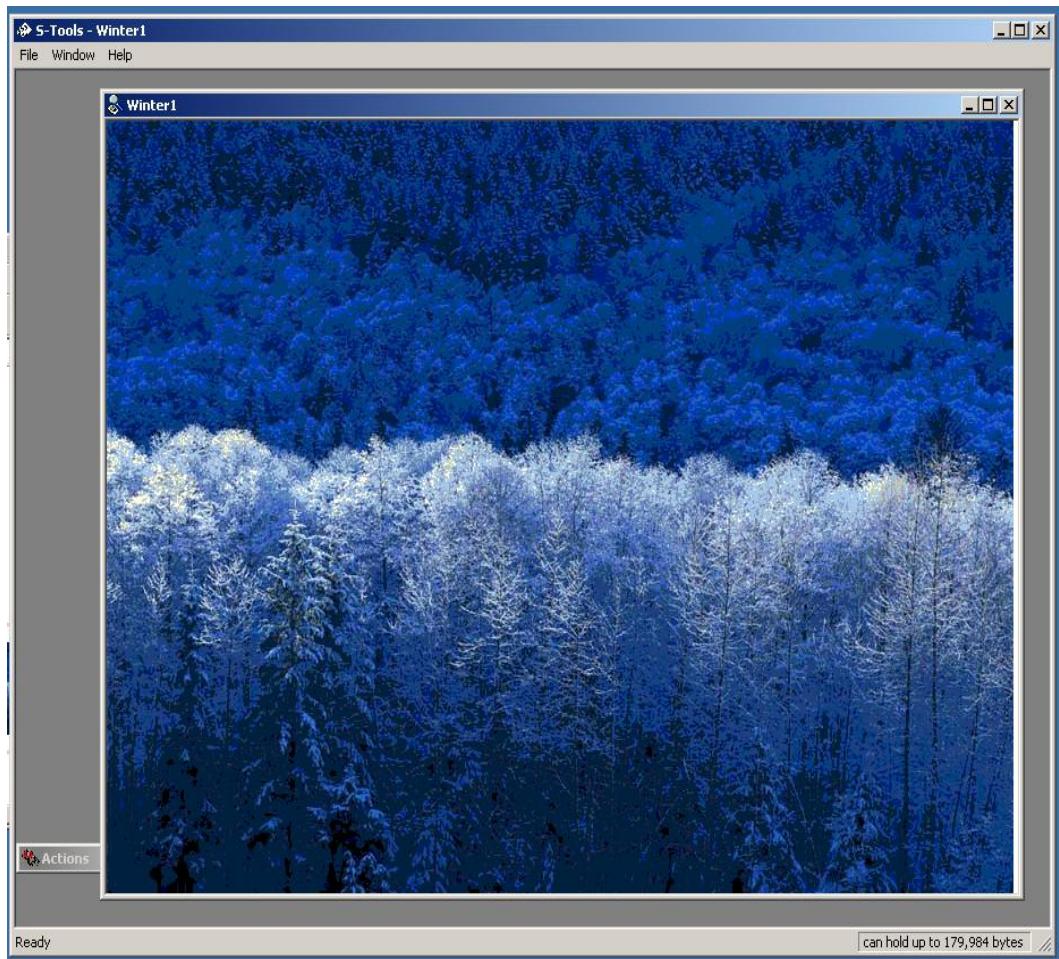
A dialogue box appears indicating that the file type is unknown. Supported file types for audio and image files are shown below:

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

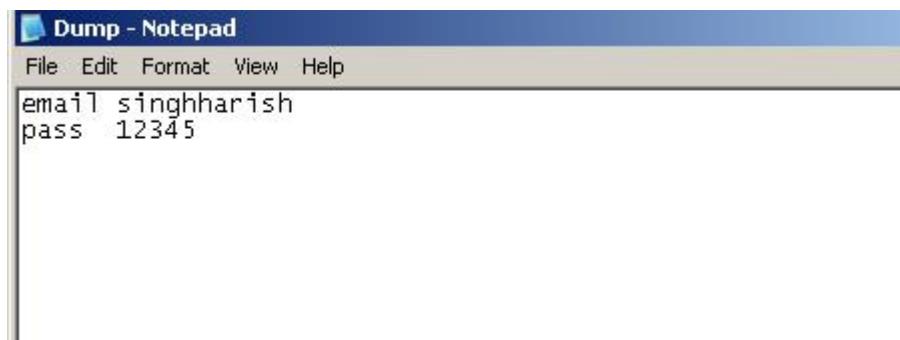
If your image is in .jpg format, convert it to .bmp format by doing the following steps using Paint:



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.

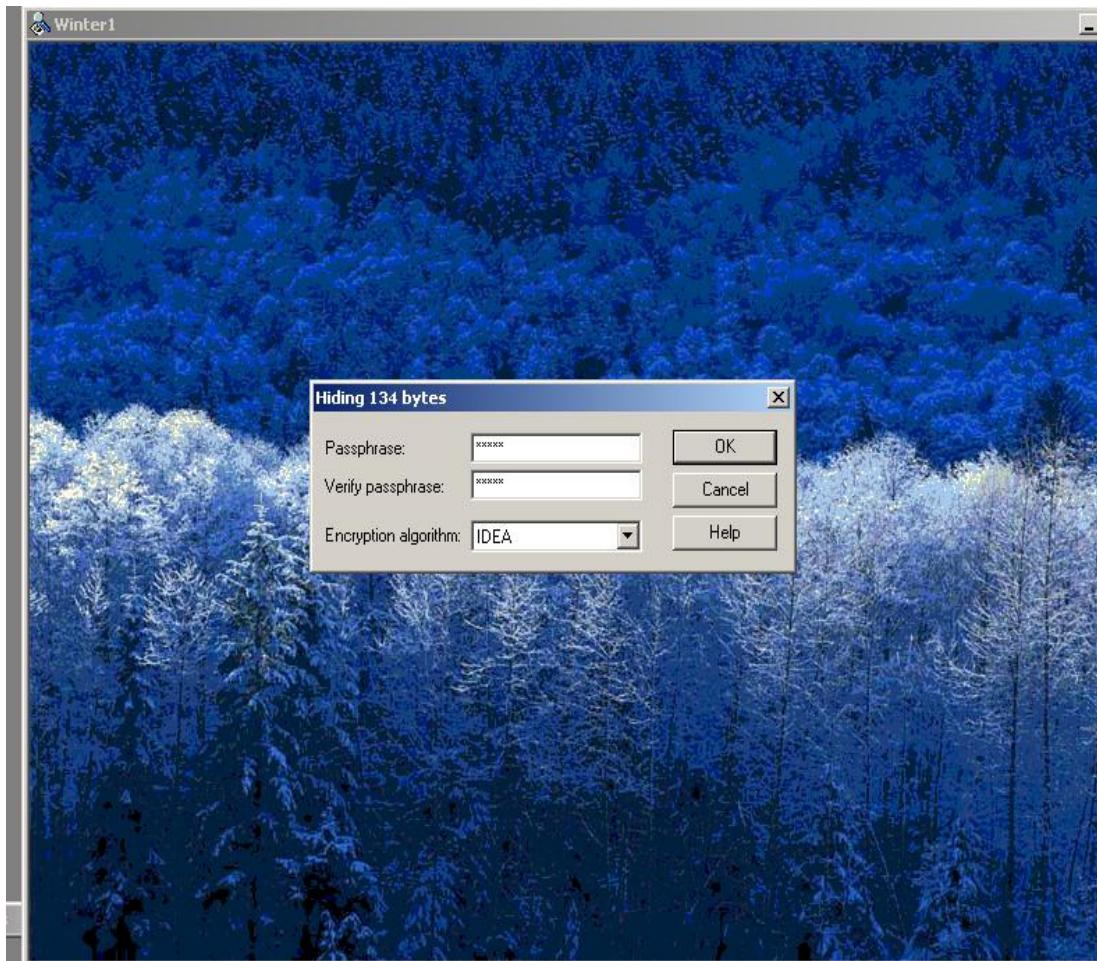


Step 5) Select a file to hide within the base file. If it's not there, create a txt file and Save the file.



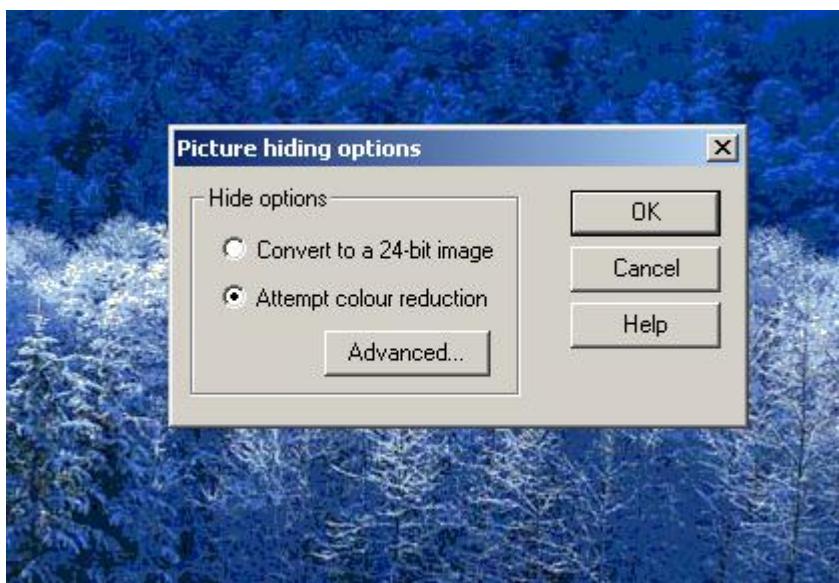
Step 6) The *.txt text file is selected and dragged on top of the base image. Release the file while the cursor is still on top of the base file.

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.

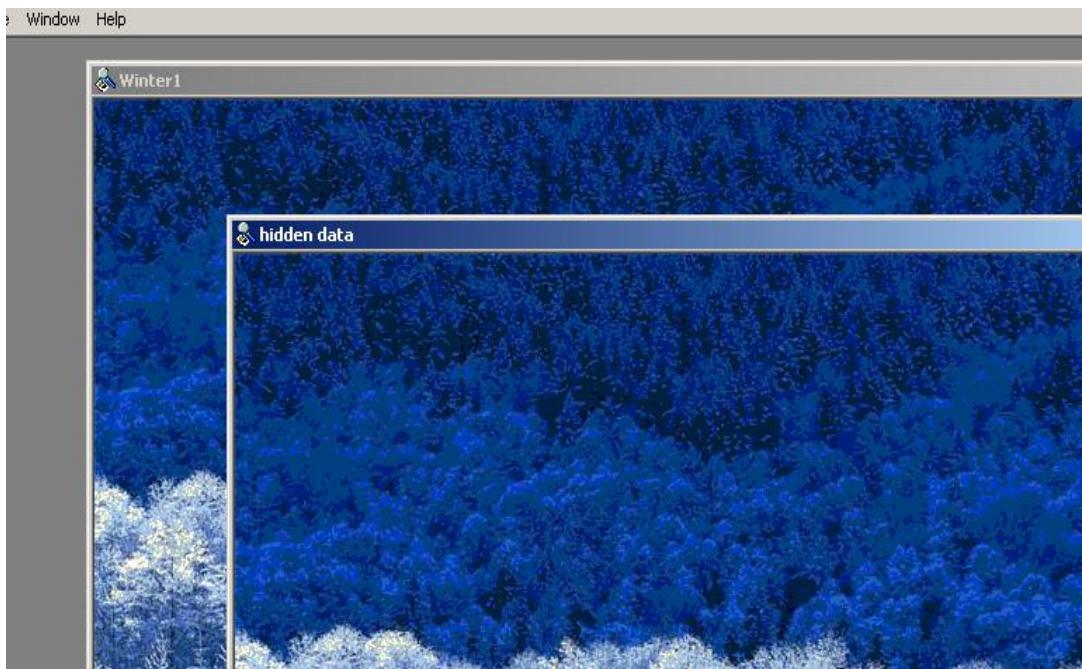


Step 8) Enter a passphrase in both the passphrase and verify passphrase text boxes. If the same passphrase is not entered in both text boxes the 'OK' button will be grayed out and the user will not be able to proceed to creating the steganography file.

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

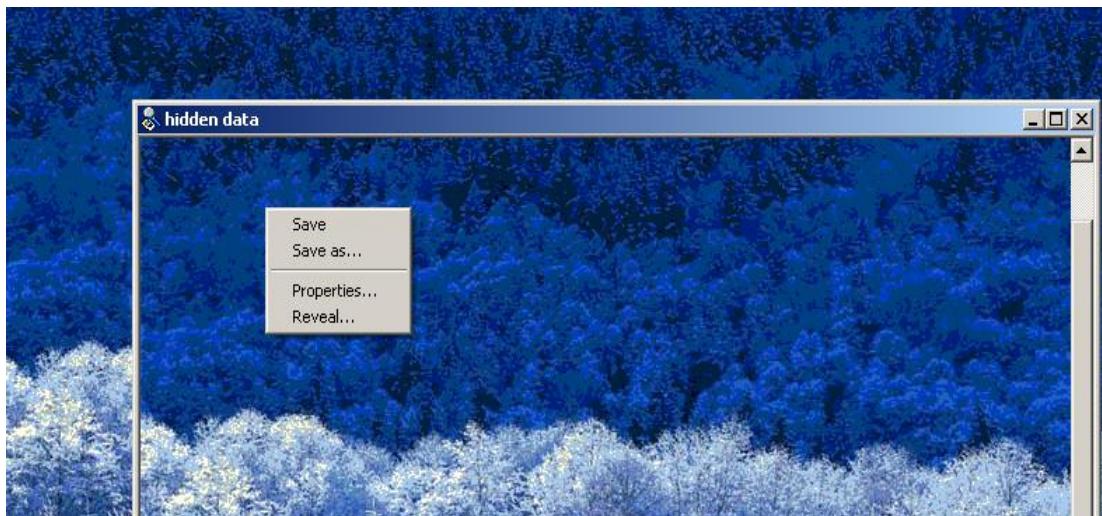


Step 10) The S-Tools main window will appear and a new file will be visible. The name of the file will be called hidden_data by default.



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:

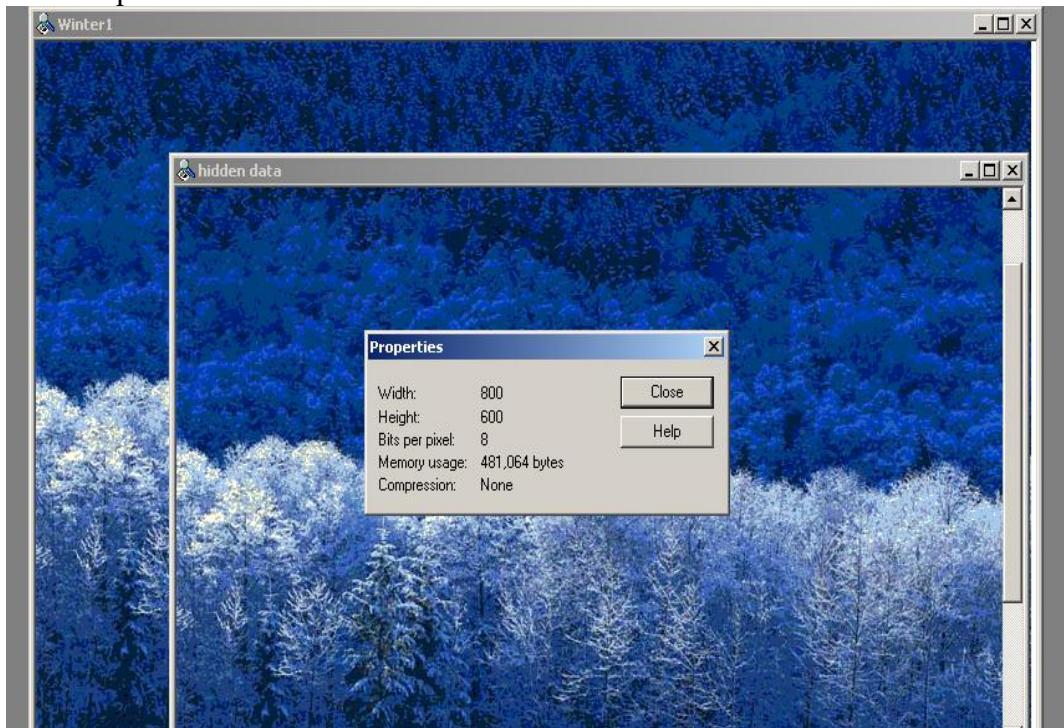
- Save
- Save As
- Properties
- Reveal



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

- Width and Height of the image

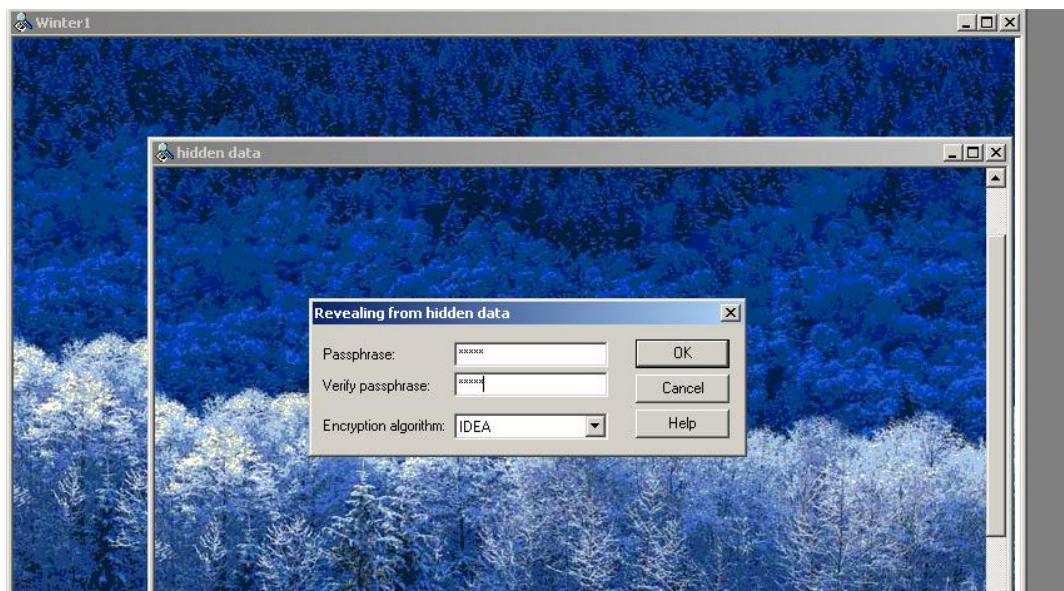
- Bits per pixel
- Memory Usage (file size in bytes)
- Compression



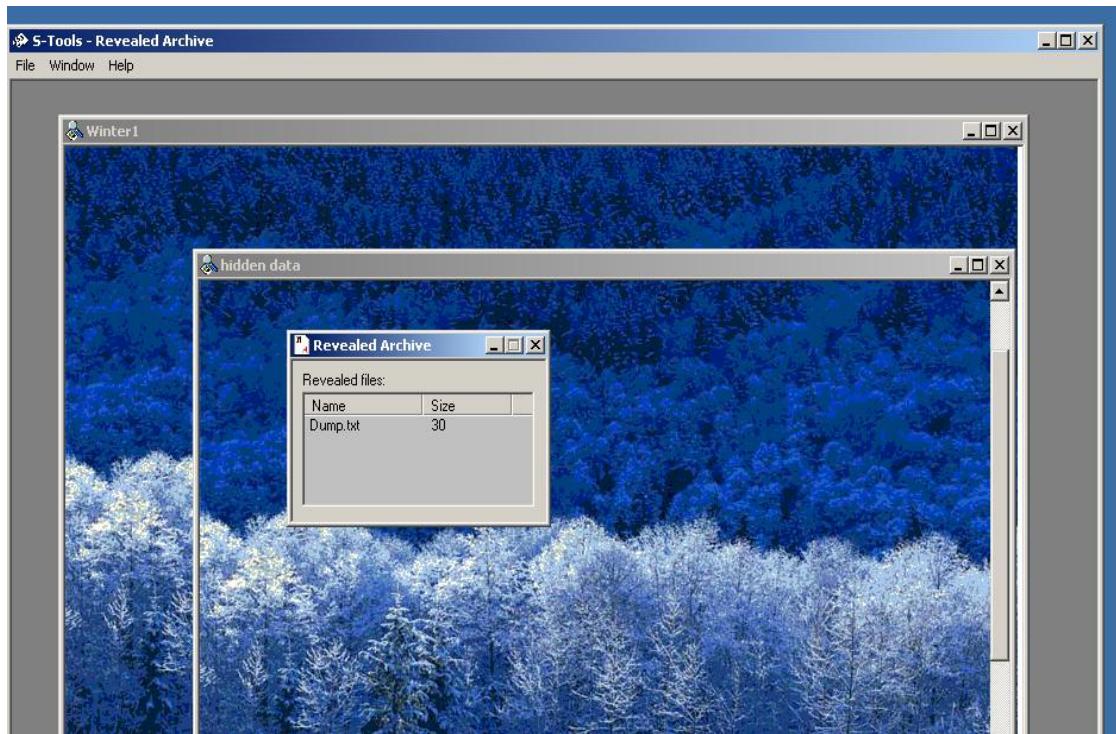
Step 13) Selecting the ‘Reveal’ button will display a passphrase dialogue box. A passphrase must be entered twice in the dialogue box and the correct encryption algorithm must be selected.

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.



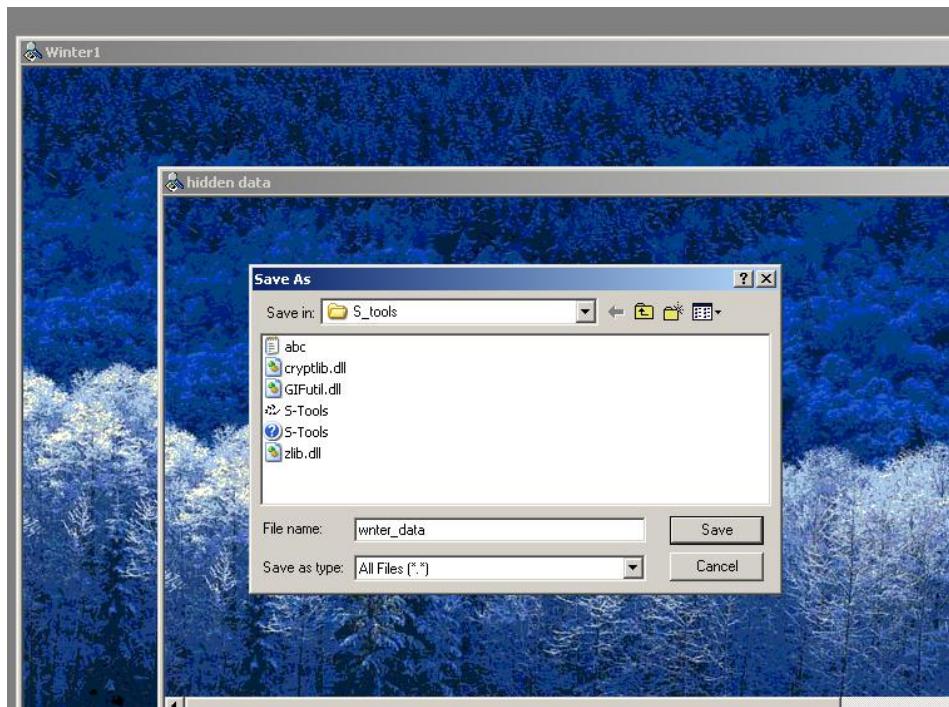
Step 15) A ‘Revealed Archive’ dialogue box will display which contains the file name and size of the hidden file.



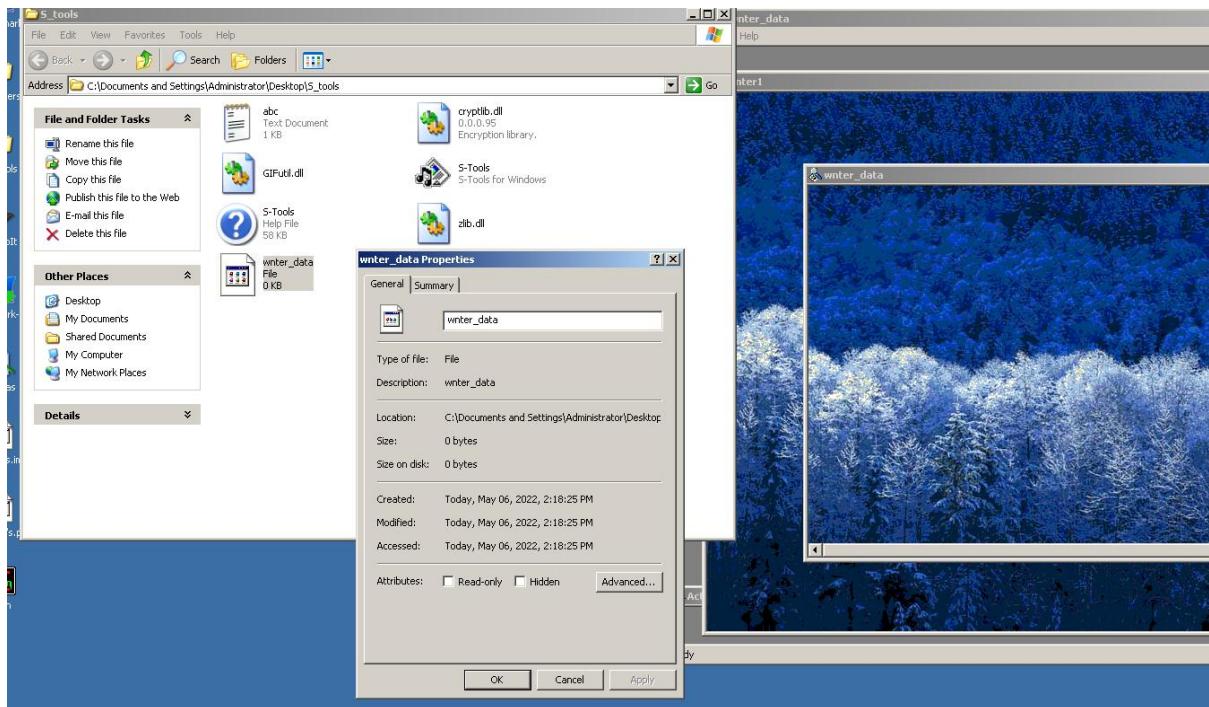
Step 16) Select the ‘Save As’ button.

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.



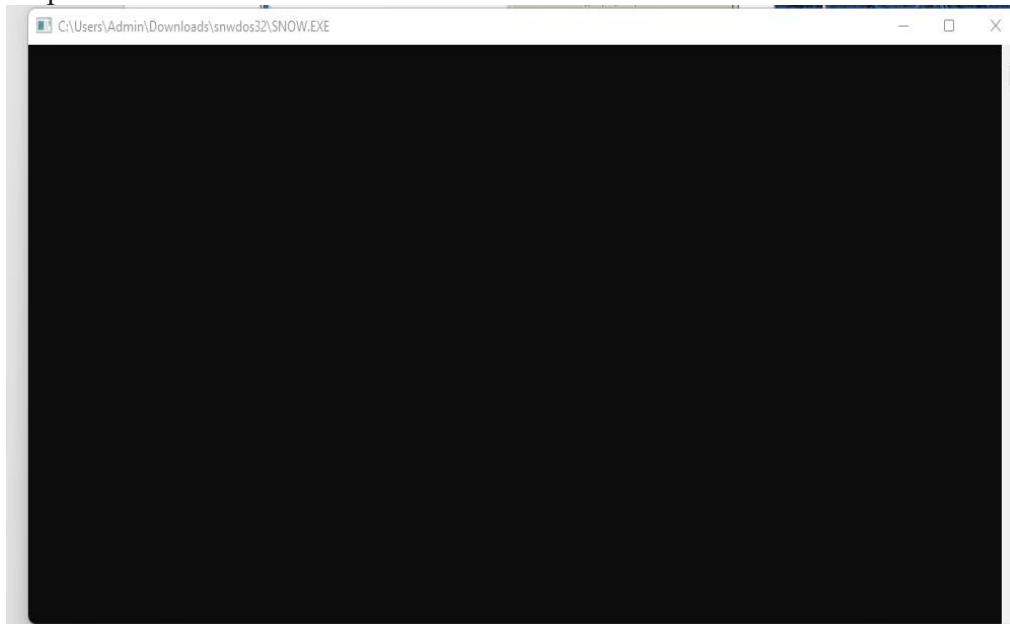
Practical No. 8B

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.

```
C:\> Command Prompt  
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>
```

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.

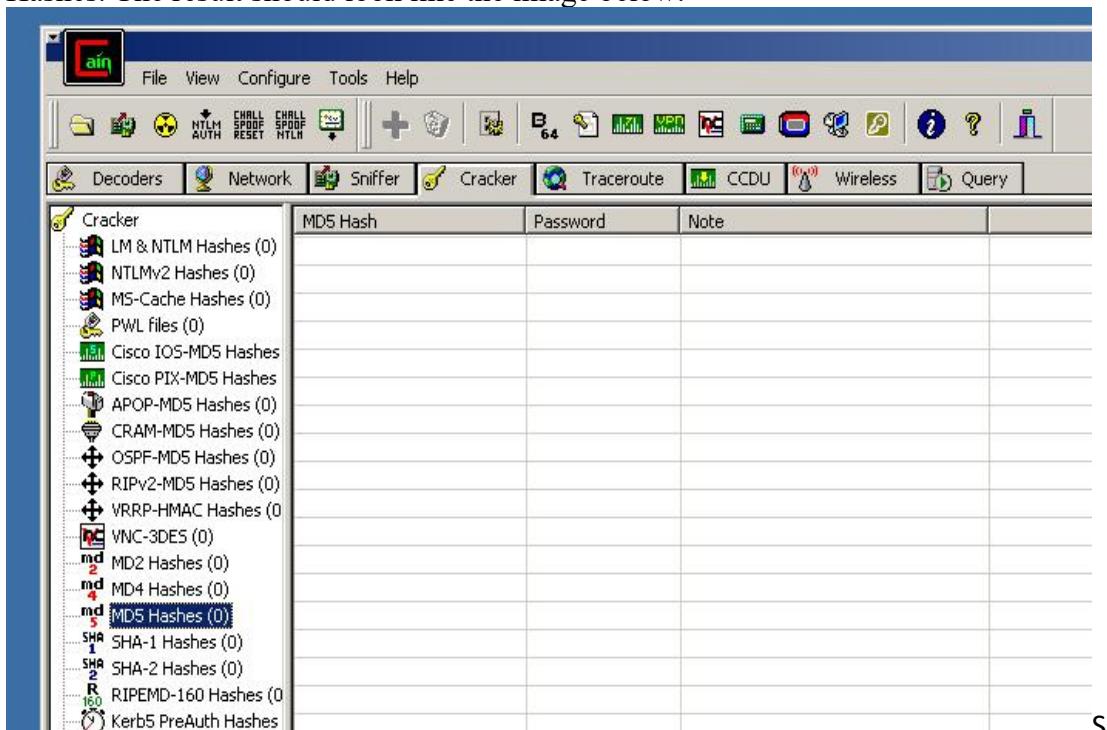
This exercise demonstrates how password could be cracked through various methods, specifically regarding MD5 encrypted passwords

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.

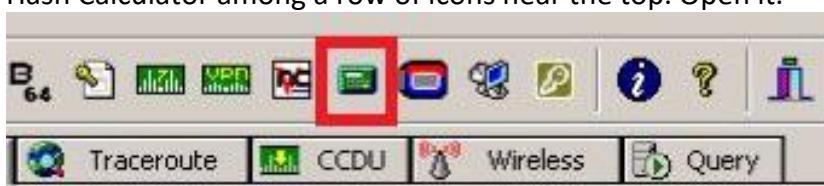
Step 1: Start Cain & Abel via the Desktop Shortcut ‘Cain’ or Start menu . a. (Start > Programs > Cain > Cain).

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

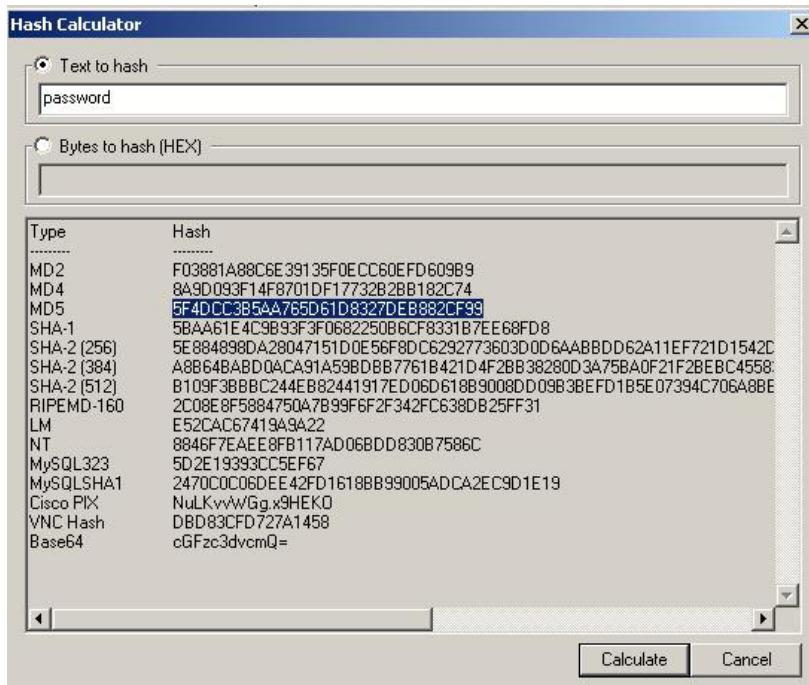
Step 3: Once on, select the ‘Cracker’ tab with the key symbol, then click on MD5 Hashes. The result should look like the image below.



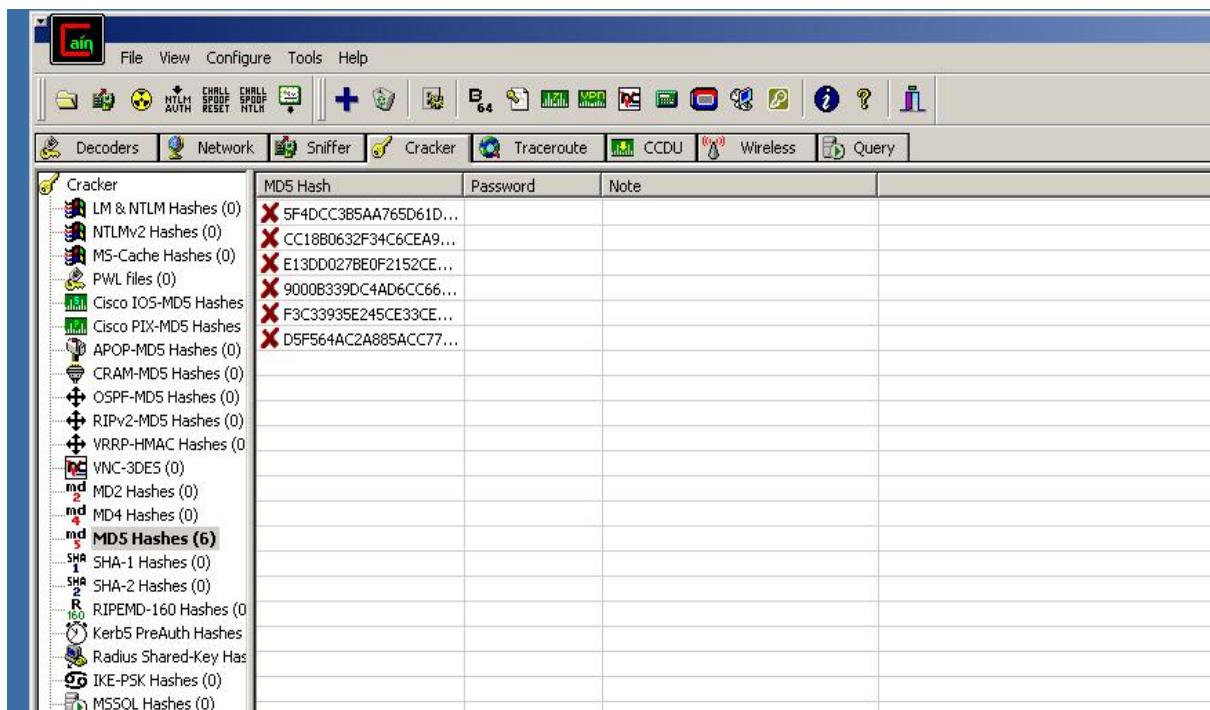
Step 4: As you might have noticed we don't have any passwords to crack, thus for the next few steps we will create our own MD5 encrypted passwords. First, locate the Hash Calculator among a row of icons near the top. Open it.



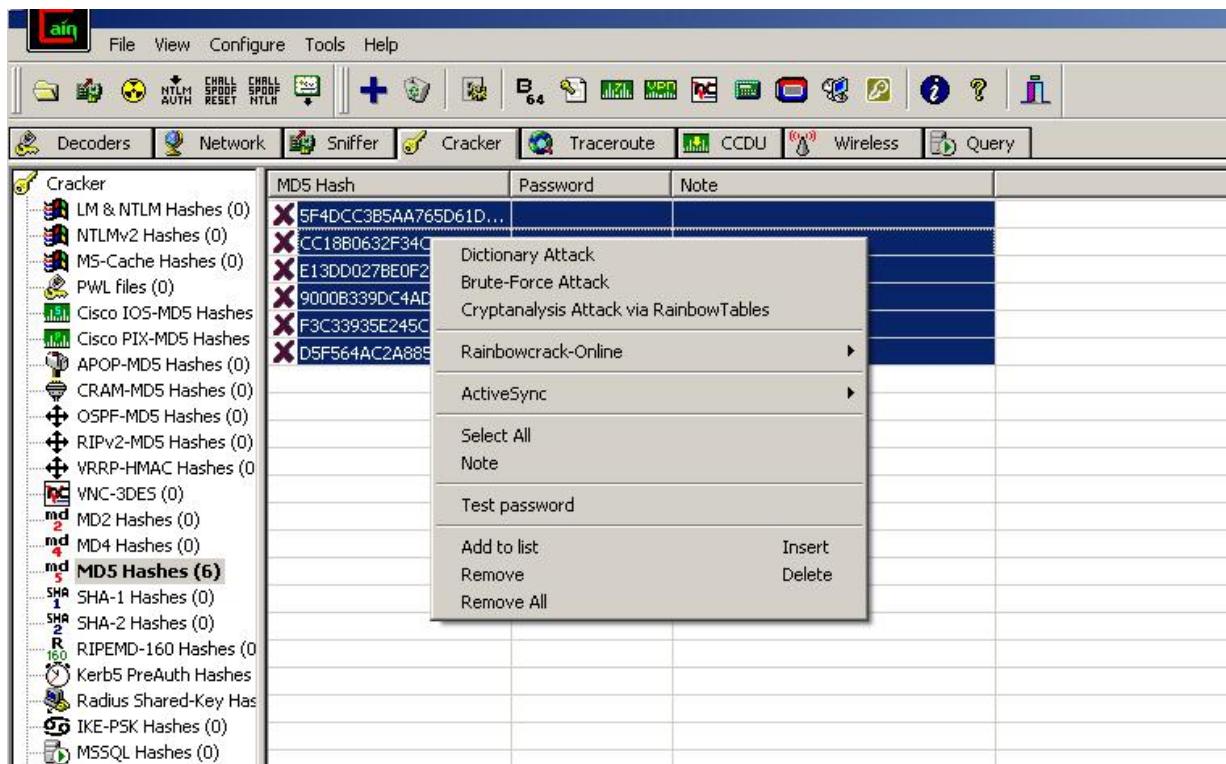
Step 5: Next, type into 'Text to Hash' the word password. It will generate a list of hashes pertaining to different types of hash algorithms. We will be focusing on MD5 hash so copy it. Then exit calculator by clicking 'Cancel'



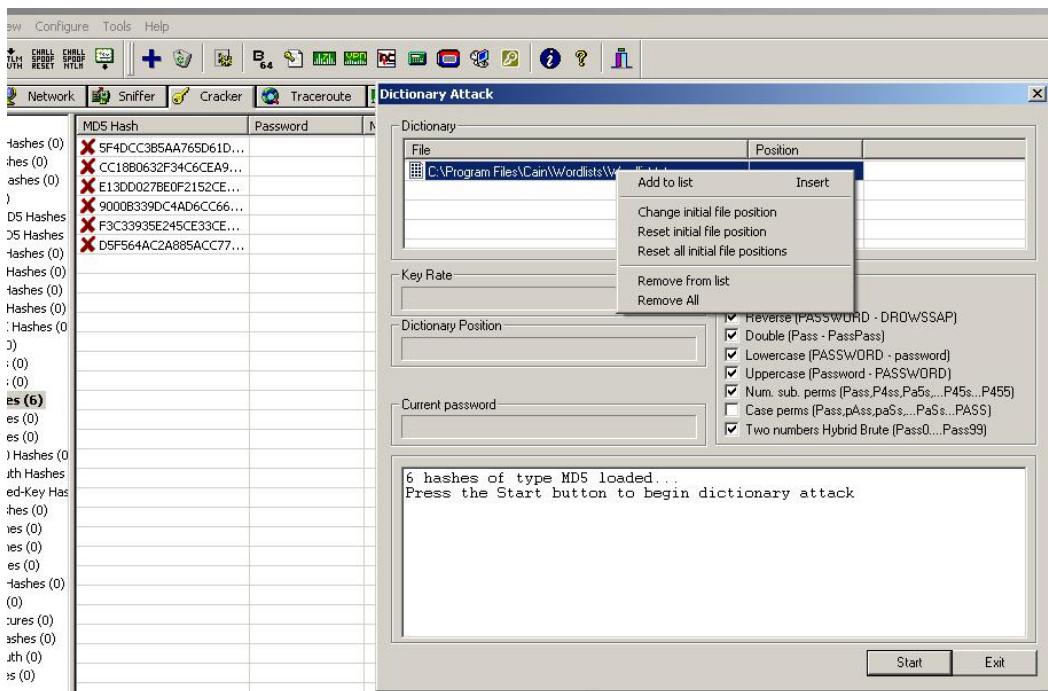
Step 6: After you exit, right click and select 'Add to list' , paste your hash then click OK . Your first encrypted password! But don't stop there, add the following MD5 hashes from the words PaSS , 13579 , 15473 , sunshine89, and c@t69



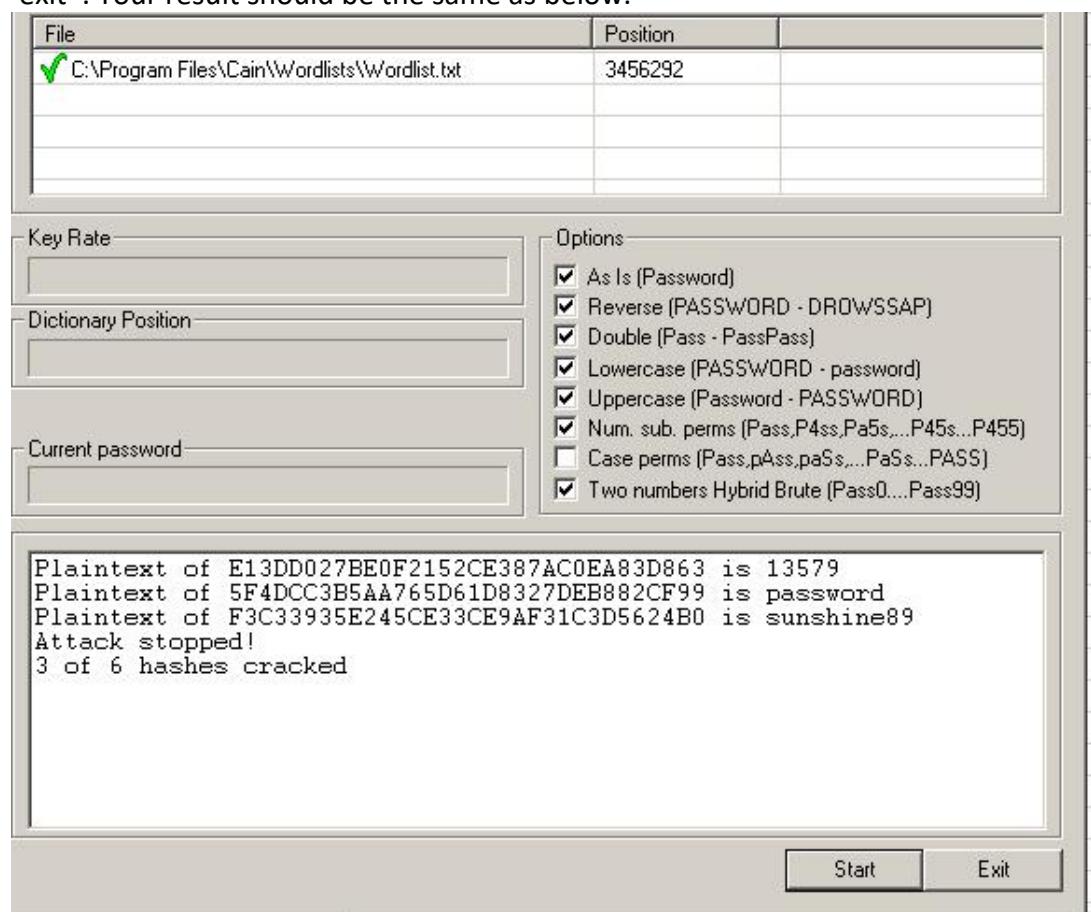
Step 7: With all the encrypted MD5 passwords on hand, we can finally start! Move your cursor and select all six passwords, then right click and press 'Dictionary Attack'.



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!

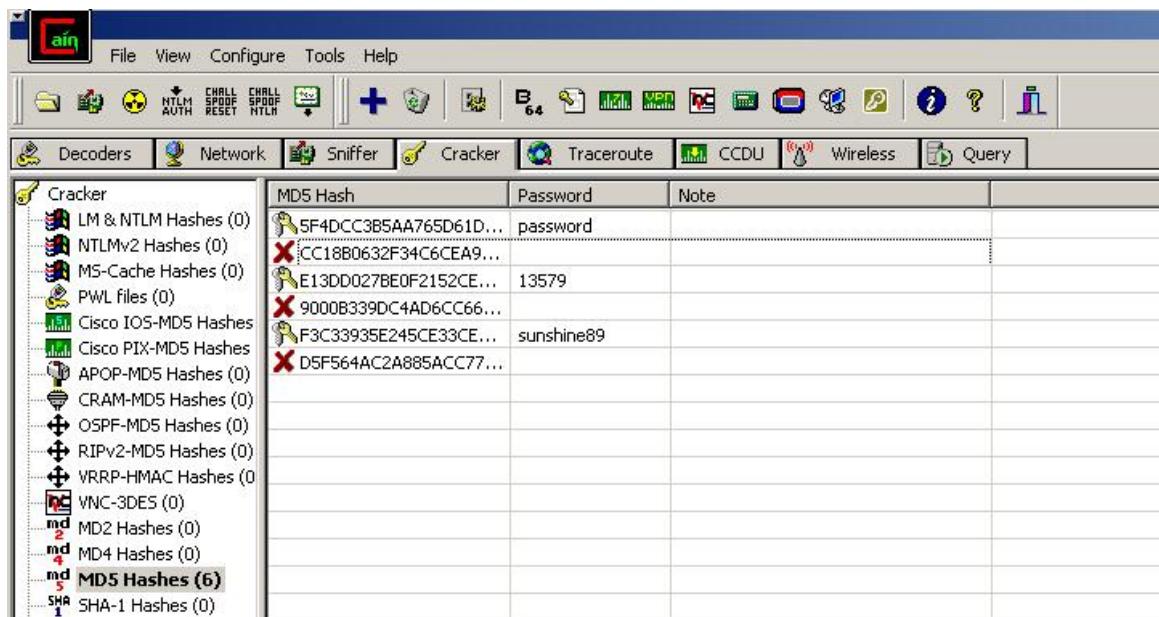


Step 9: Click 'start' and watch the magic happens before your eyes! Once it ends 'exit' . Your result should be the same as below.

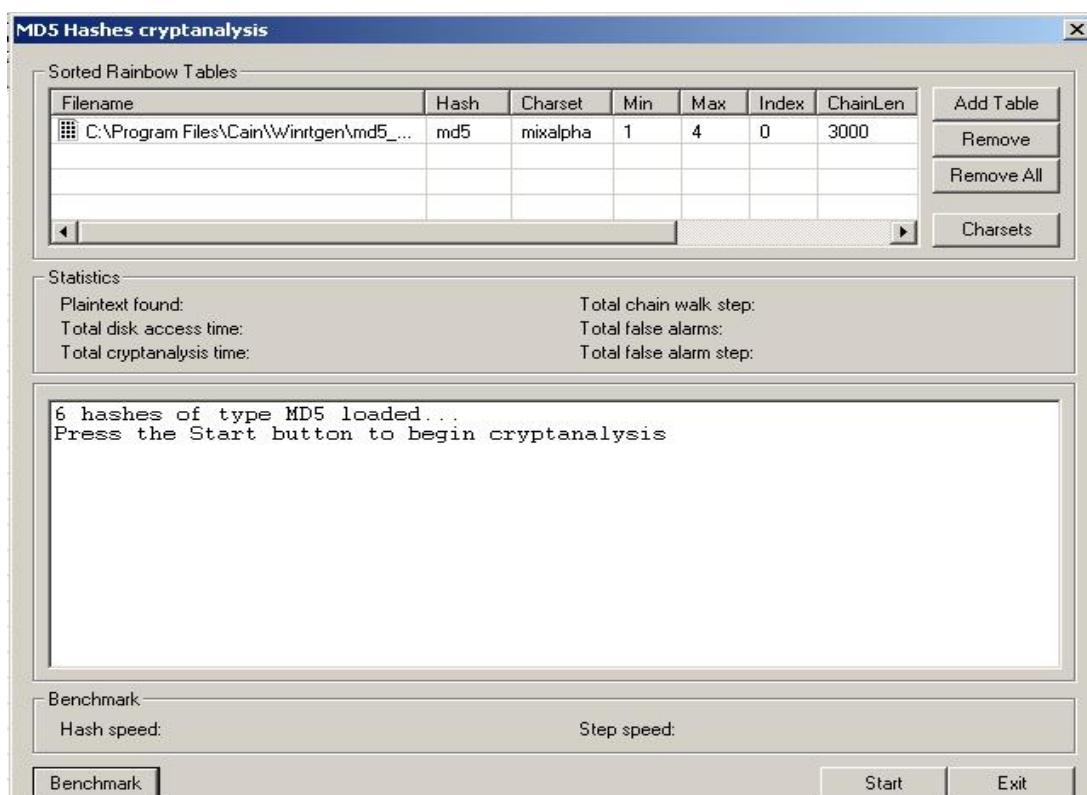


Part 2 : Rainbow Tables - Rainbow tables use pre-calculated MD5 hashes sorted on a table(s) to compare to encrypted MD5 files in order to find a match thus cracking the password. This type of password cracking trades time and storage capacity.

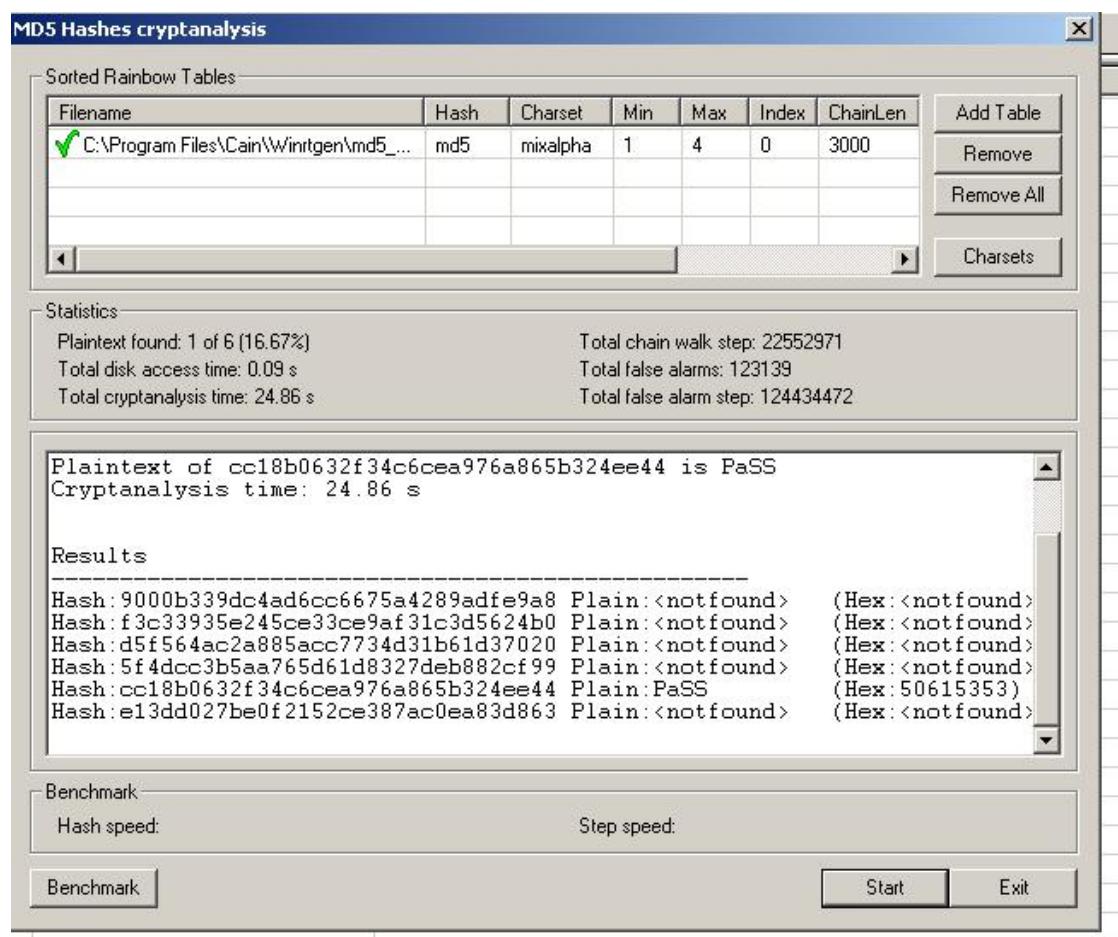
Step 1: Continuation from the previous 'Dictionary Attack' section. Cain & Abel should already be opened with following MD5 encrypted passwords.



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'.

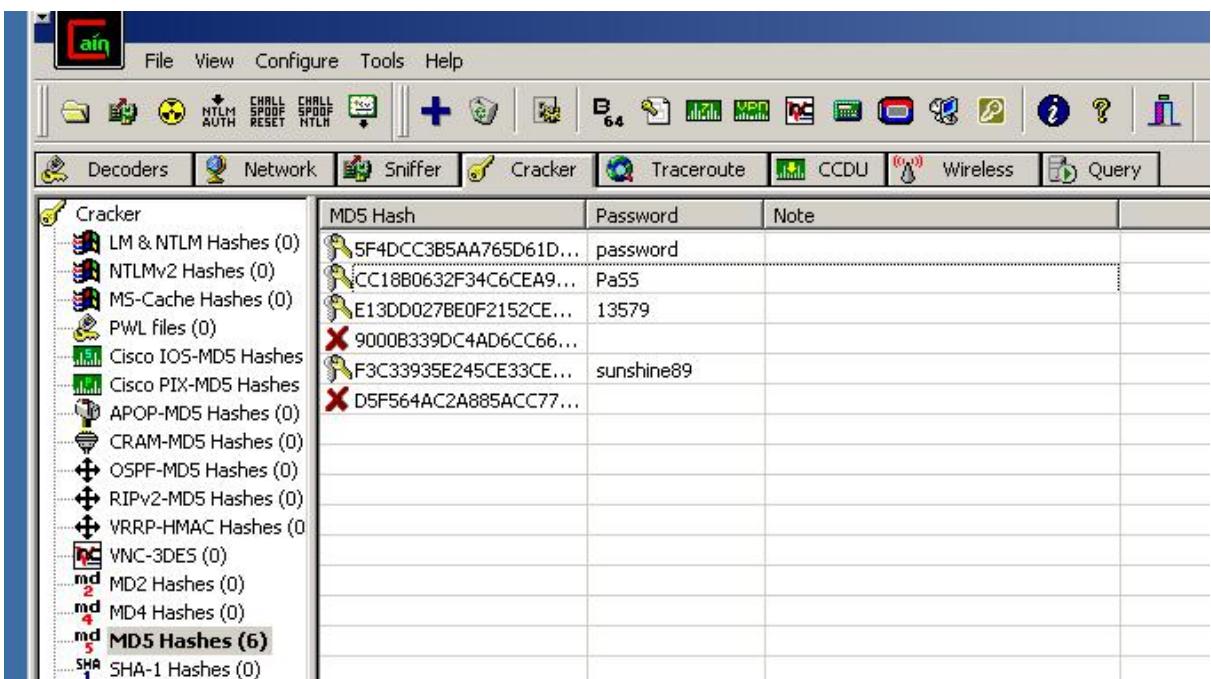


Step 3: A window will pop up and you could see under 'Sorted Rainbow Tables' there is already a MD5 rainbow table already added. Notice the specifications for that specific rainbow table. Click 'Start' when ready. 'Exit' when done.

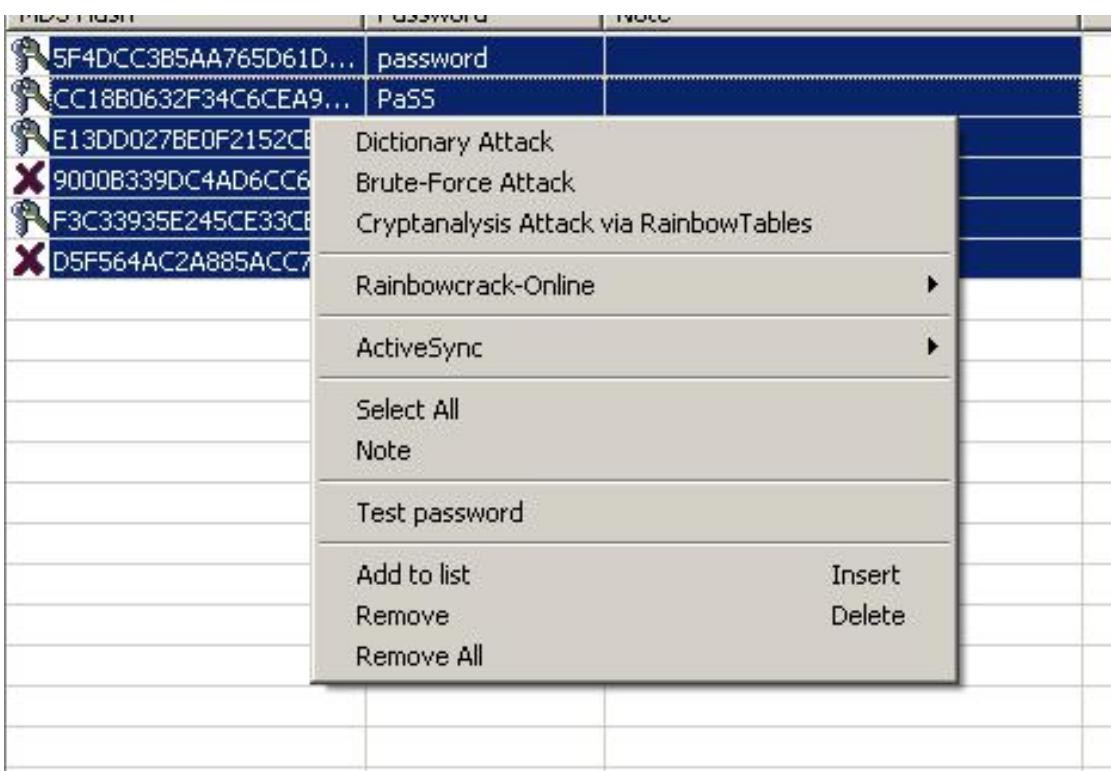


Part 3: Brute Force - Brute force attacks uses a finite but enormous number of combinations involving alphabet, numbers, and symbols in order to crack a password. This type of password cracking is usually used as a last resort as it's the most time consuming overall.

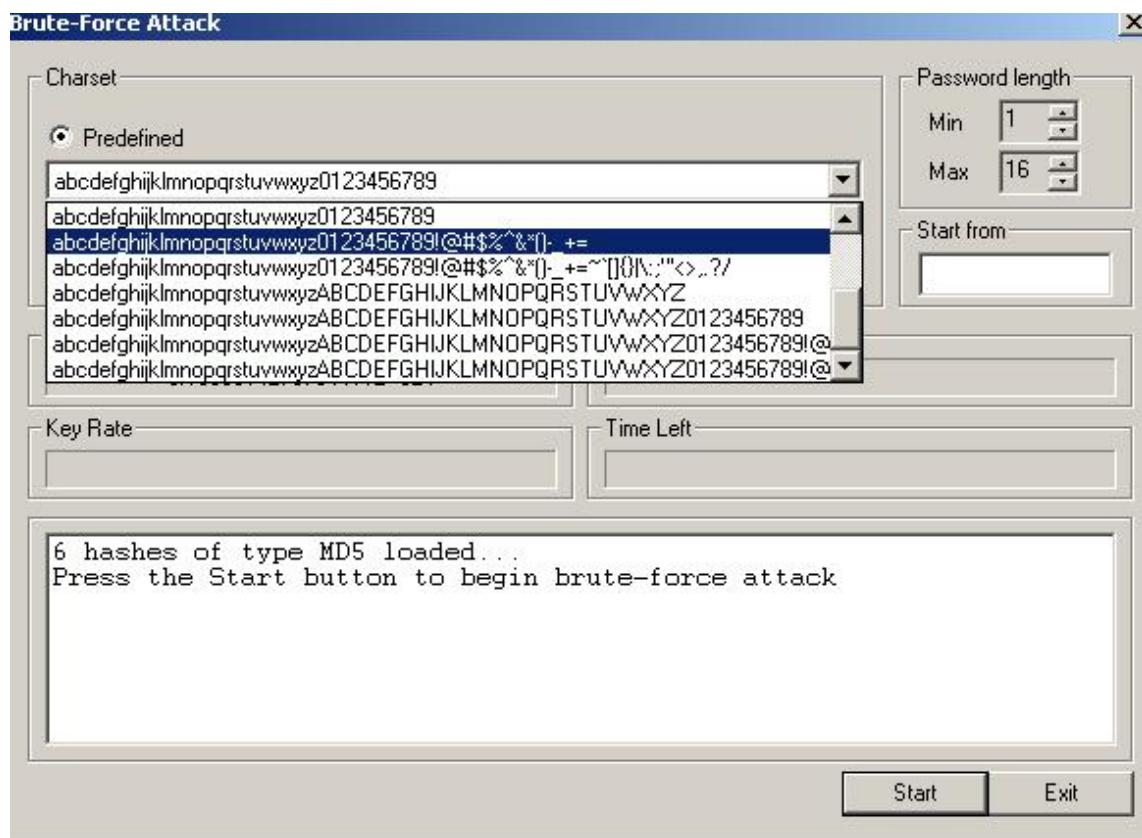
Step 1: Continuation from the previous 'Rainbow Tables' section. Cain & Abel should already be opened with the following MD5 encrypted passwords.



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!

