Blockchain

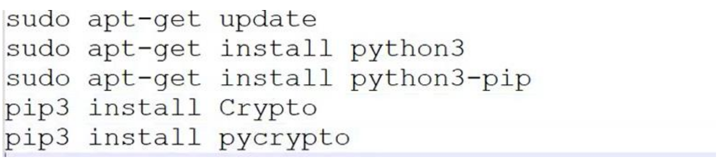
Index

Sr. No	Title	Date	Sign
1.	A . simple client class that generates the private and public keys by using the built-in Python RSA algorithm and test it. B. Create multiple transactions and display them C. Create a transaction class to send and receive money and test it. 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.	write a solidity program for variables, operators, loops, decision making and string.		
3.	A. write a solidity program for string, arrays, enums, structure & mappings. B. write a solidity program for function, view function, pure function & fallback function.		
4.	A. write a solidity program for function overloading, mathematical function & cryptographic functions B. write a solidity program for contract, inheritance, constructors, abstract contracts, interfaces, libraries, assembly, events, error handling		
5.	Install hyperledger-Irhoa		
6.	Demonstrate the running of the blockchain node.		
7.	Demonstrate the running of the blockchain node.		
8.	Demonstrate the use of Bitcoin Core API.		
9.	Build Dapps with angular[using truffle and ganache cli		

Practical 1

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

After Creating Ubuntu VM-> Login -> Open Terminal -> Install below packages



```
sudo apt-get update
sudo apt-get install python3
sudo apt-get install python3-pip
pip3 install Crypto
pip3 install pycrypto
```

Code :

```
pip3 install Crypto

pip3 install pycrypto

import hashlib
import random
import binascii
import datetime
import collections
from Crypto.PublicKey import RSA
from Crypto import Random
from Crypto.Cipher import PKCS1_v1_5
class Client:
  def __init__(self):
    random = Random.new().read
    self._private_key = RSA.generate(1024, random)
    self._public_key = self._private_key.publickey()
    self._signer = PKCS1_v1_5.new(self._private_key)
  @property
  def identity(self):
    return binascii.hexlify(self._public_key.exportKey(format='DER')).decode('ascii
Dinesh = Client()
print ("sender ",Dinesh.identity)
```

Output:

```
sender  30819f300d06092a864886f.........
```

B. Create multiple transactions and display them

Code:

```
import hashlib
import binascii
import datetime
import collections
from Crypto.PublicKey import RSA
from Crypto import Random
from Crypto.Cipher import PKCS1_v1_5
from collections import OrderedDict
import Crypto
import Crypto.Random
from Crypto.Hash import SHA
from Crypto.Signature import PKCS1_v1_5
class Client:
   def __init__(self):
     random = Random.new().read
     self._private_key = RSA.generate(1024, random)
     self._public_key = self._private_key.publickey()
     self._signer = PKCS1_v1_5.new(self._private_key)
   @property
   def identity(self):
     return binascii.hexlify(self._public_key.exportKey(format='DER')).decode('ascii')

class Transaction:
   def __init__(self, sender, recipient, value):
      self.sender = sender
      self.recipient = recipient
      self.value = value
      self.time = datetime.datetime.now()

   def to_dict(self):
      if self.sender == "Genesis":
         identity = "Genesis"
      else:
         identity = self.sender.identity
      return collections.OrderedDict({
         'sender': identity,
         'recipient': self.recipient,
         'value': self.value,
         'time' : self.time})
   def sign_transaction(self):
      private_key = self.sender._private_key
```

```
        signer = PKCS1_v1_5.new(private_key)
        h = SHA.new(str(self.to_dict()).encode('utf8'))
        return binascii.hexlify(signer.sign(h)).decode('ascii')
def display_transaction(transaction):
        #for transaction in transactions:
        dict = transaction.to_dict()
        print ("sender: " + dict['sender'])
        print ('-----')
        print ("recipient: " + dict['recipient'])
        print ('-----')
        print ("value: " + str(dict['value']))
        print ('-----')
        print ("time: " + str(dict['time']))
        print ('-----')
transactions = []
A = Client()
B = Client()
t1 = Transaction(
   A,
   B.identity,
   15.0
)
t1.sign_transaction()
display_transaction (t1)
```

Output:

```
sender: 30819f300d0609…….
-----
recipient: 30819f300d06…..
-----
value: 15.0
-----
time: 2022-04-26 04:00:21.070283
-----
```

C. Create a transaction class to send and receive money and test it

Code:

```
# following imports are required by PKI
import hashlib
import binascii
import datetime
import collections

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

class Transaction:
   def __init__(self, sender, recipient, value):
      self.sender = sender
      self.recipient = recipient
      self.value = value
      self.time = datetime.datetime.now()

   def to_dict(self):
      if self.sender == "Genesis":
         identity = "Genesis"
      else:
         identity = self.sender.identity

      return collections.OrderedDict({
         'sender': identity,
         'recipient': self.recipient,
         'value': self.value,
        'time' : self.time})

    def sign_transaction(self):
      private_key = self.sender._private_key
      signer = PKCS1_v1_5.new(private_key)
      h = SHA.new(str(self.to_dict()).encode('utf8'))
      return binascii.hexlify(signer.sign(h)).decode('ascii')

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

transactions = []

Dinesh = Client()
Ramesh = Client()
Suresh = Client()

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

t1.sign_transaction()
transactions.append(t1)

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

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

tn=1
for t in transactions:
  print("Transaction #",tn)
  display_transaction (t)
  tn=tn+1
  print ('--------------')
```

Output:

```
Transaction # 1
sender: 30819f300d060…
-----
recipient: 30819f300d02a864….
-----
value: 15.0
-----
time: 2022-04-26 04:07:59.162213
-----
--------------
Transaction # 2
sender: 30819f300d06092a8…..
-----
recipient: 30819f300d06092a8…..
-----
value: 25.0
-----
time: 2022-04-26 04:07:59.165396
-----
--------------
Transaction # 3
sender: 30819f300d06092a8648….
-----
recipient: 30819f300d06092a86488…
-----
value: 200.0
-----
time: 2022-04-26 04:07:59.168579
-----
```

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

Code:

```
import hashlib
import binascii
import datetime
import collections

from Crypto.PublicKey import RSA
from Crypto import Random
from Crypto.Cipher import PKCS1_v1_5
from collections import OrderedDict
import Crypto
import Crypto.Random
from Crypto.Hash import SHA
from Crypto.Signature import PKCS1_v1_5

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

class Transaction:
  def __init__(self, sender, recipient, value):
      self.sender = sender
      self.recipient = recipient
      self.value = value
      self.time = datetime.datetime.now()

  def to_dict(self):
      if self.sender == "Genesis":
         identity = "Genesis"
      else:
         identity = self.sender.identity
      return collections.OrderedDict({
        'sender': identity,
        'recipient': self.recipient,
        'value': self.value,
        'time' : self.time})

    def sign_transaction(self):
      private_key = self.sender._private_key
      signer = PKCS1_v1_5.new(private_key)
      h = SHA.new(str(self.to_dict()).encode('utf8'))
      return binascii.hexlify(signer.sign(h)).decode('ascii')

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

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

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

Dinesh = Client()

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

block0 = Block()
block0.previous_block_hash = None
Nonce = None
block0.verified_transactions.append (t0)
digest = hash (block0)
last_block_hash = digest

TPCoins = []
TPCoins.append (block0)

dump_blockchain(TPCoins)
```

Output:

```
Number of blocks in the chain: 1
block # 0
sender: Genesis
-----
recipient: 30819f300d06092.....
-----
value: 500.0
-----
time: 2022-04-26 04:24:05.232662
```

<u>E. Create a mining function and test it.</u>

Code:

```
import hashlib

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

def mine(message, difficulty=1):
  assert difficulty >= 1
  #if(difficulty <1):
  #       return
  #'1'*2=> '11'
  prefix = '1' * difficulty
  print("prefix",prefix)
  for i in range(1000):
    digest = sha256(str(hash(message)) + str(i))
    print("testing=>"+digest)
    if digest.startswith(prefix):
      print ("after " + str(i) + " iterations found nonce: "+ digest)
      return i #i= nonce value

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

Output:

```
prefix 11
testing=>ab7d1f2b4ba63486a274d7a8c5e4dde793c2d47069ae19ab832dc1177622a182
testing=>cf0a36c4f0c3107cba7a8ebe690db004a01f659bc0aed3b327f01fab0065bf41
testing=>fb0eac040f5f40cd4a39373ca0e6165c07a36db3df510b4c0ad4d45654caeabb
testing=>a298e97de6df74e3856aabbd5aeed9807652d98a9911a6431bdb3bad0ad2a7bd
testing=>7ff8aa3e5b40e1b5bed59ab464c9b98ceff64b2445cc446cc89ecd93330cba1e
…….
testing=>1cddb5b7e9af6eda960e734606c33f0ce676a7e557a22ba4d7b9af557b0c0360
testing=>29d2f56130e7b276b3cfb94687ff3b1d5c79b6dc8238fe259aae1f5af19fd8b2
testing=>3a5f4dcfed5301f36be80fd7d42573b1585ea4ef9037e96853affe66d68f8a04
testing=>ddb4d9dc8c7f20443eedc9ac798aebb2c080cc46926dc0151760e37097bf2dcf
testing=>4fb1010880723ce012526941ae6236260852c8e995583d0d2f65b6f9ff655c61
testing=>11038c5fc4f90108f4198097c76c9af5d38c92b48fe27968eacbd89324fe9d2a
after 21 iterations found nonce:
11038c5fc4f90108f4198097c76c9af5d38c92b48fe27968eacbd89324fe9d2a
21
```

<u>F. Add blocks to the miner and dump the blockchain.</u>

```
# following imports are required by PKI
import hashlib
import random
import binascii
import datetime
import collections

from Crypto.PublicKey import RSA
from Crypto import Random
from Crypto.Cipher import PKCS1_v1_5
from collections import OrderedDict
import Crypto
import Crypto.Random
```

```
from Crypto.Hash import SHA
from Crypto.Signature import PKCS1_v1_5

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

class Transaction:
   def __init__(self, sender, recipient, value):
      self.sender = sender
      self.recipient = recipient
      self.value = value
      self.time = datetime.datetime.now()

   def to_dict(self):
      if self.sender == "Genesis":
         identity = "Genesis"
      else:
         identity = self.sender.identity

      return collections.OrderedDict({
         'sender': identity,
         'recipient': self.recipient,
         'value': self.value,
         'time' : self.time})

   def sign_transaction(self):
      private_key = self.sender._private_key
      signer = PKCS1_v1_5.new(private_key)
      h = SHA.new(str(self.to_dict()).encode('utf8'))
      return binascii.hexlify(signer.sign(h)).decode('ascii')

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

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

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

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

def mine(message, difficulty=1):
  assert difficulty >= 1
  #if(difficulty <1):
  #       return
  #'1'*3=> '111'
  prefix = '1' * difficulty
  for i in range(1000):
    digest = sha256(str(hash(message)) + str(i))
    if digest.startswith(prefix):
      return i #i= nonce value


A = Client()
B =Client()
C =Client()
t0 = Transaction (
  "Genesis",
  A.identity,
  500.0
)

t1 = Transaction (
  A,
  B.identity,
  40.0
)
t2 = Transaction (
  A,
  C.identity,
  70.0
)
t3 = Transaction (
  B,
  C.identity,
  700.0
)
#blockchain
TPCoins = []

block0 = Block()
block0.previous_block_hash = None
Nonce = None
block0.verified_transactions.append (t0)
digest = hash (block0)
last_block_hash = digest #last_block_hash it is hash of block0
TPCoins.append (block0)

block1 = Block()
block1.previous_block_hash = last_block_hash
block1.verified_transactions.append (t1)
block1.verified_transactions.append (t2)
block1.Nonce=mine (block1, 2)
digest = hash (block1)
last_block_hash = digest
TPCoins.append (block1)


block2 = Block()
block2.previous_block_hash = last_block_hash
block2.verified_transactions.append (t3)
Nonce = mine (block2, 2)
block2.Nonce=mine (block2, 2)
digest = hash (block2)
```

```
last_block_hash = digest
TPCoins.append (block2)


dump_blockchain(TPCoins)
```

Output:

```
Number of blocks in the chain: 3
block # 0
sender: Genesis
-----
recipient: 30819f300d0609.....
-----
value: 500.0
-----
time: 2022-04-26 04:30:59.070952
-----
--------------
=====================================
block # 1
sender: 30819f300d06092a86.....
-----
recipient: 30819f300d06092a.....
-----
value: 40.0
-----
time: 2022-04-26 04:30:59.071076
-----
--------------
sender: 30819f300d06092a86....
-----
recipient: 30819f300d06092a....
-----
value: 70.0
-----
time: 2022-04-26 04:30:59.071174
-----
--------------
=====================================
block # 2
sender: 30819f300d06092a....
-----
recipient: 30819f300d06092a....
-----
value: 700.0
-----
time: 2022-04-26 04:30:59.071272
-----
```

```
---------------
========================================
```

Practical 2

AIM: write a solidity program for variables, operators, loops, decision making and string.

A)Variables:

supports three types of variables.

State Variables – Variables whose values are permanently stored in a contract storage.

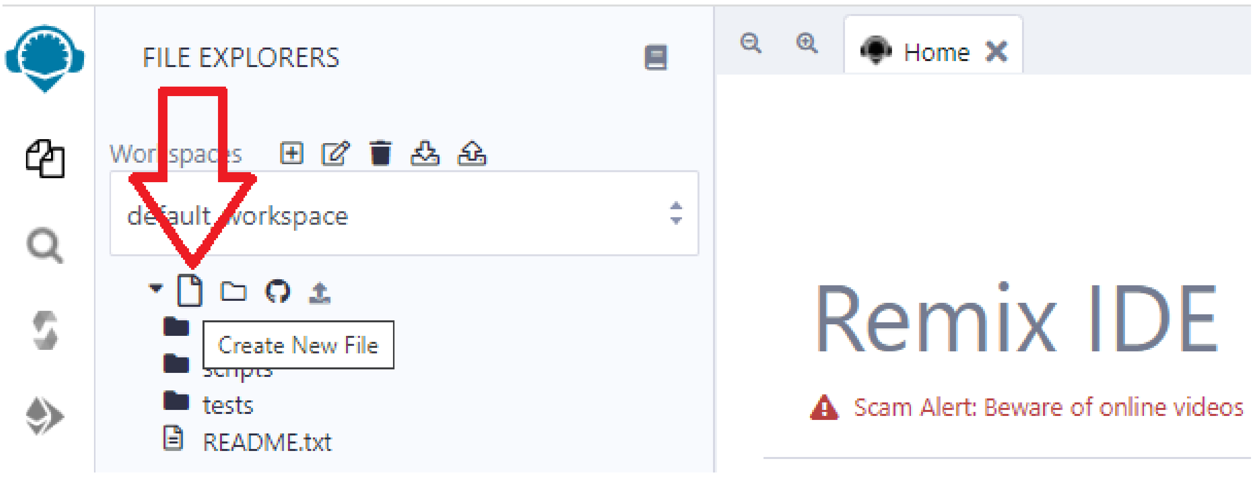
Local Variables – Variables whose values are present till function is executing.

Global Variables – Special variables exists in the global namespace used to get information about the blockchain.i.e. blockhash(uint blockNumber) returns (bytes32), block.coinbase (address payable), block.difficulty (uint)…..and many more

Step 1: Open this website

https://remix.ethereum.org/

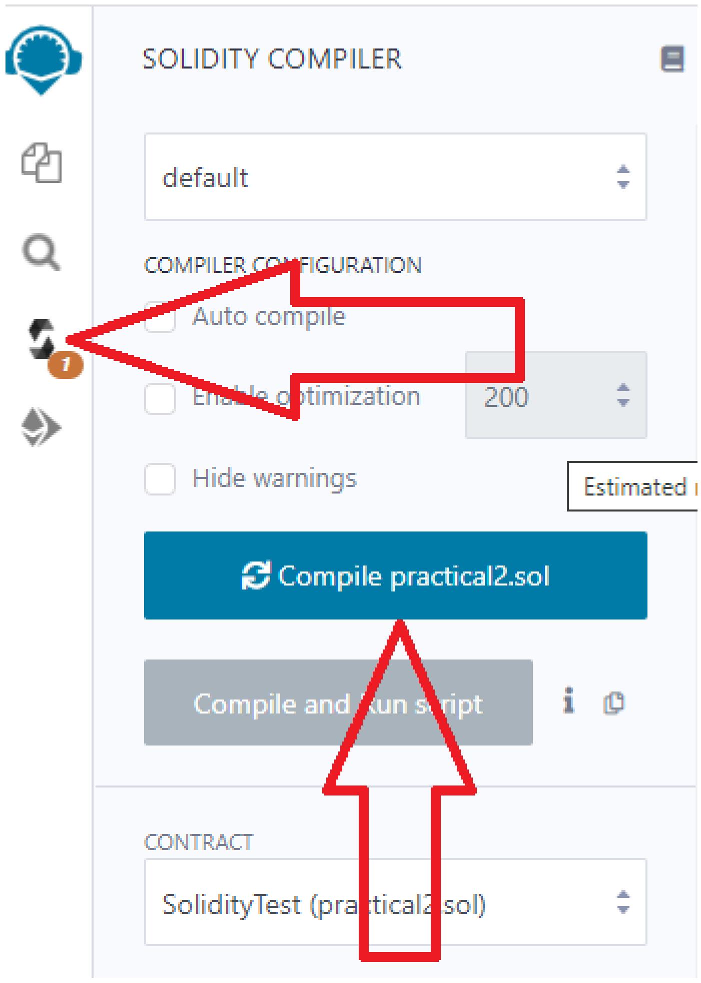
Step 2: Create new file – practical.sol



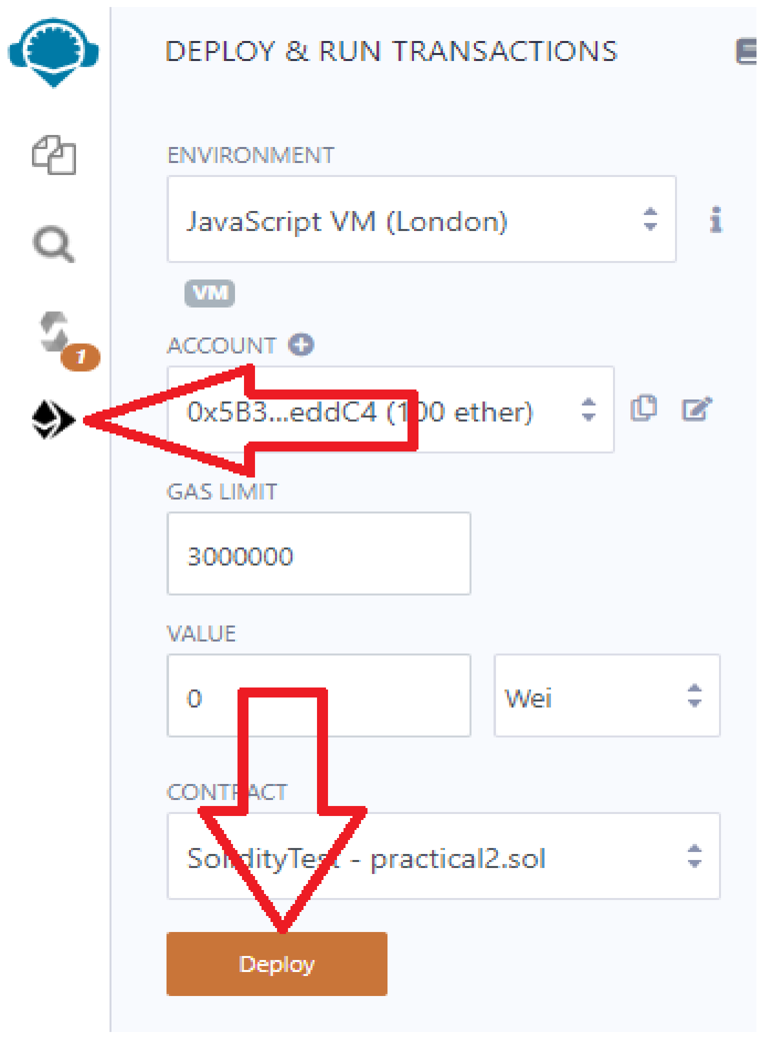
Step 3: Write this program in the new file

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

Step 4: Compile contract



Step 5: Deploy contract



DEPLOY & RUN TRANSACTIONS

ENVIRONMENT: JavaScript VM (London)

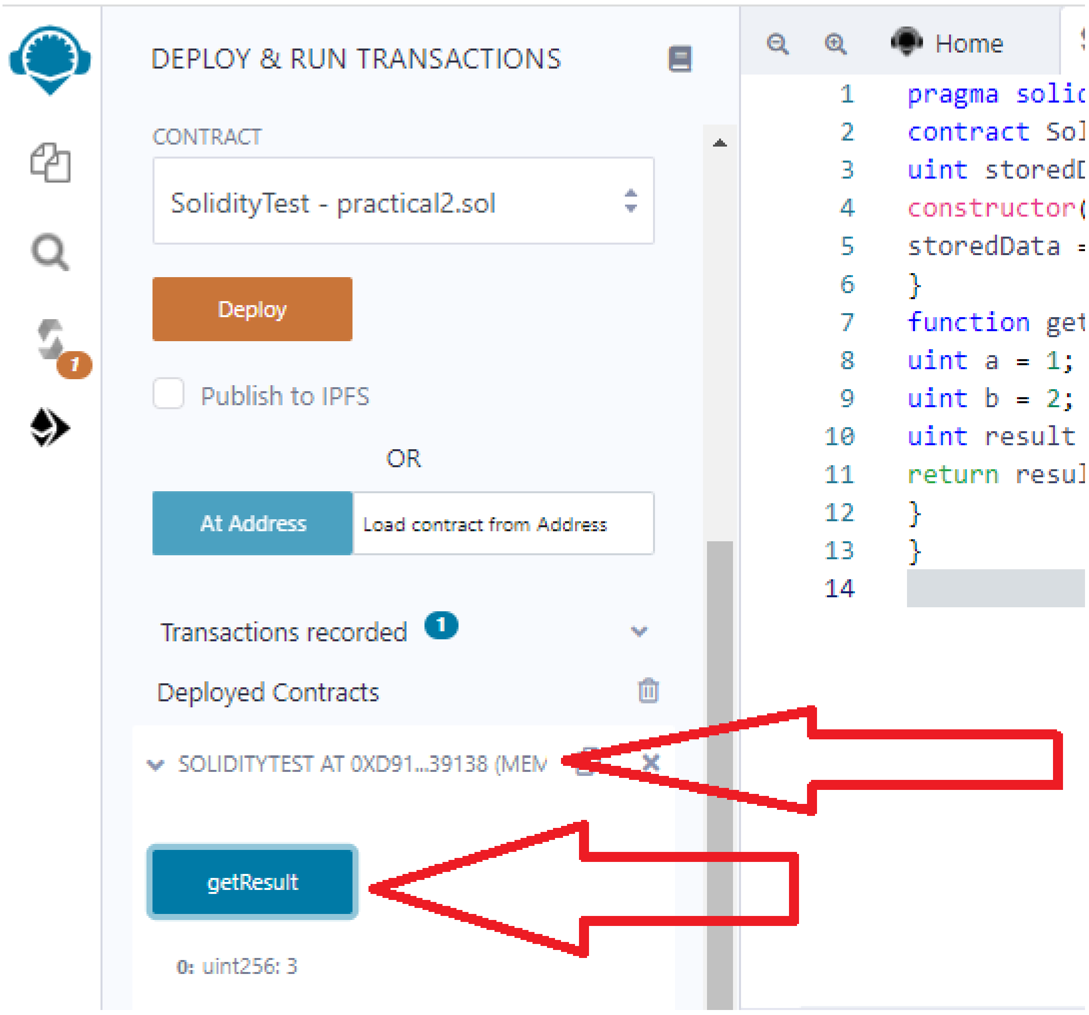
ACCOUNT: 0x5B3...eddC4 (100 ether)

GAS LIMIT: 3000000

VALUE: 0 Wei

CONTRACT: SolidityTest - practical2.sol

Step 6: Select the contract and click button



DEPLOY & RUN TRANSACTIONS

CONTRACT

SolidityTest - practical2.sol

Deploy

Publish to IPFS

OR

At Address | Load contract from Address

Transactions recorded 1

Deployed Contracts

SOLIDITYTEST AT 0XD91...39138 (MEM

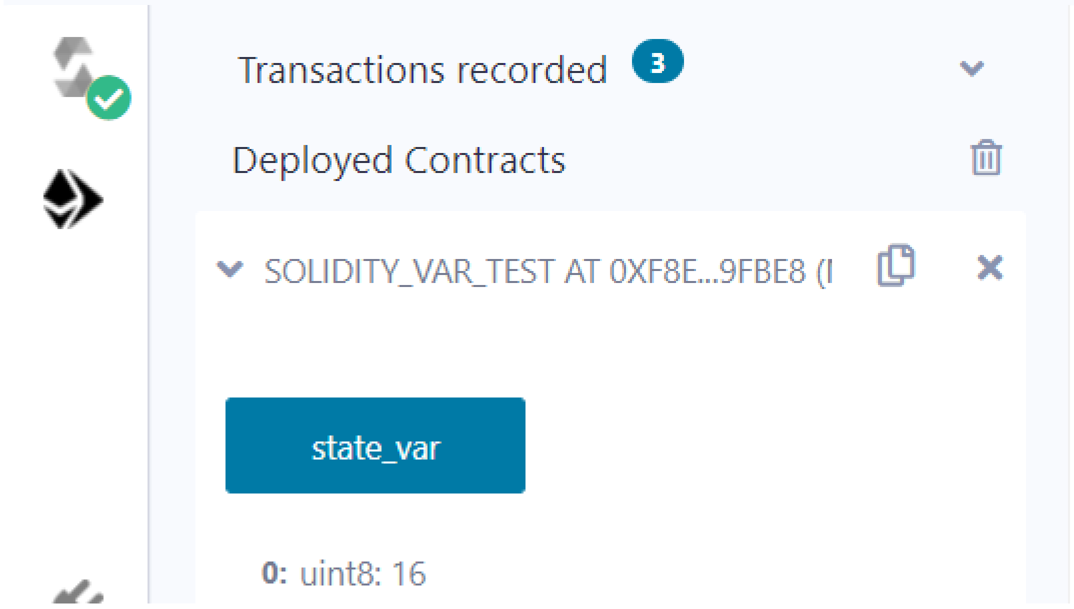
getResult

0: uint256: 3

```
pragma solic
contract Sol
uint storedI
constructor(
storedData =
}
function get
uint a = 1;
uint b = 2;
uint result
return resul
}
}
```

1.State Variable:

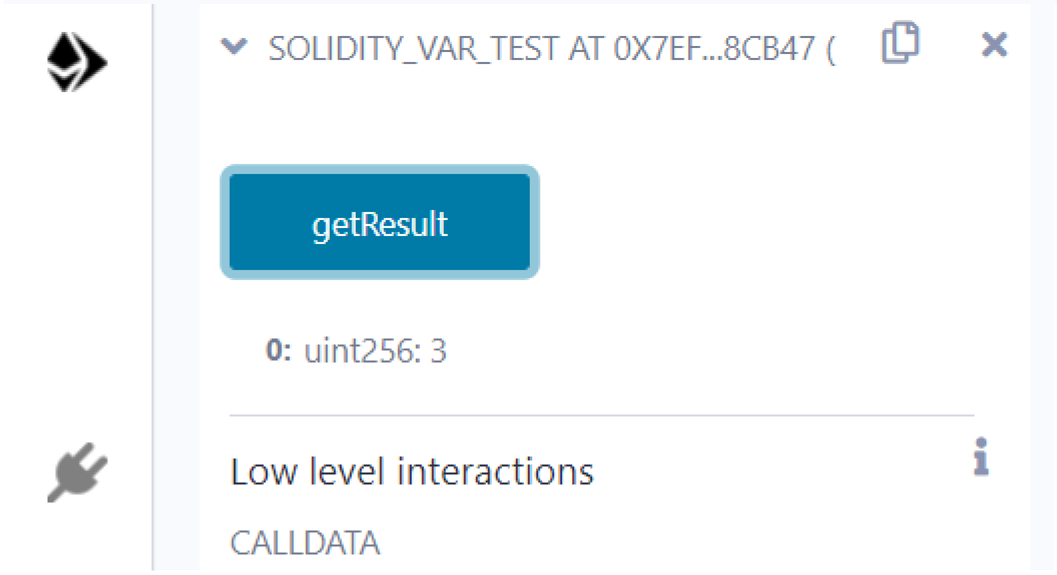
```
// Solidity program to
// demonstrate state
// variables
pragma solidity ^0.5.0;
// Creating a contract
contract Solidity_var_Test {
// Declaring a state variable
uint8 public state_var;
// Defining a constructor
constructor() public {
state_var = 16;
}
}
```



```
0: uint8: 16
```

2.Local Variable:

```
// Solidity program to demonstrate
// local variables
pragma solidity ^0.5.0;
// Creating a contract
contract Solidity_var_Test {
// Defining function to show the declaration and
// scope of local variables
function getResult() public view returns(uint){
// Initializing local variables
uint local_var1 = 1;
uint local_var2 = 2;
uint result = local_var1 + local_var2;
// Access the local variable
return result;
}
}
```

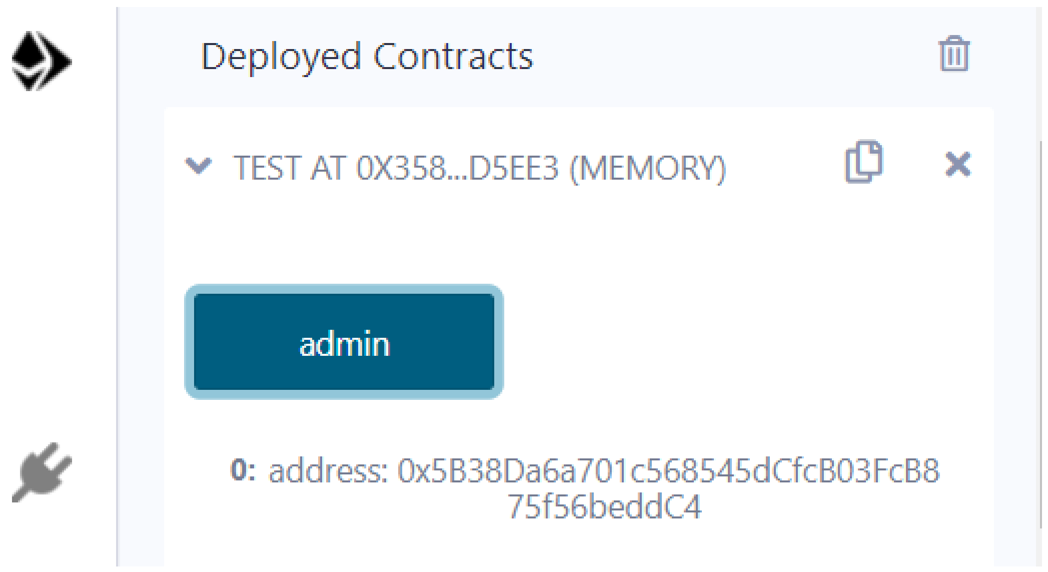


SOLIDITY_VAR_TEST AT 0X7EF...8CB47 (

```
0: uint256: 3
```

3.Global variable:

```
// Solidity program to
// show Global variables
pragma solidity ^0.5.0;
// Creating a contract
contract Test {
// Defining a variable
address public admin;
// Creating a constructor to
// use Global variable
constructor() public {
admin = msg.sender;
}
}
```



0: address: 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4

Scope of local variables is limited to function in which they are defined but State variables can have three types of scopes.

Public – Public state variables can be accessed internally as well as via messages. For a public state variable, an automatic getter function is generated.

Internal – Internal state variables can be accessed only internally from the current contract or contract deriving from it without using this.

Private – Private state variables can be accessed only internally from the current contract they are defined not in the derived contract from it.

B)Operators

Solidity supports the following types of operators.

Arithmetic Operators

Comparison Operators

Logical (or Relational) Operators

Assignment Operators

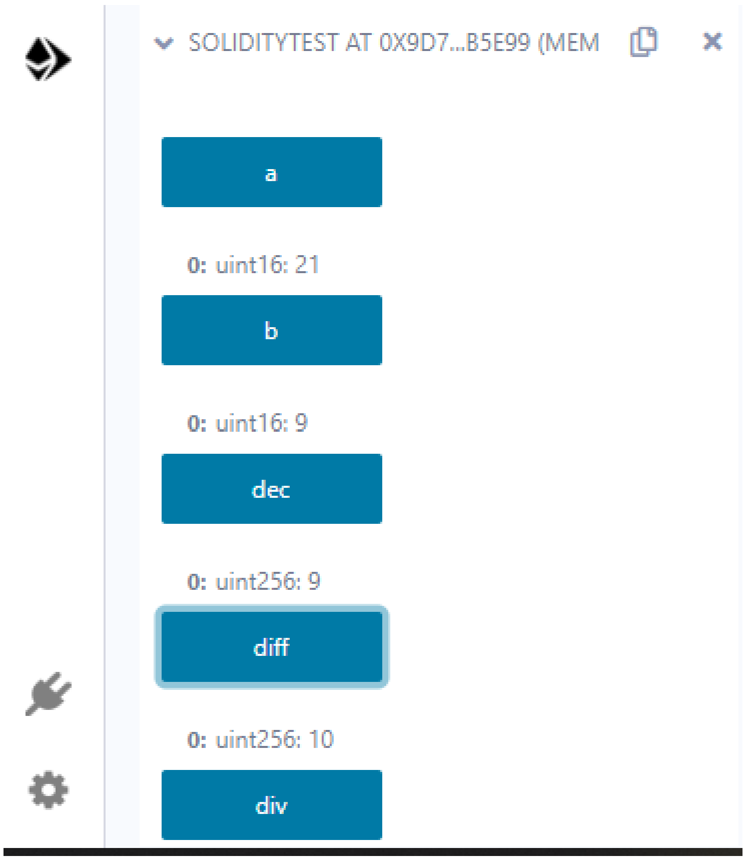
Conditional (or ternary) Operators

1. Arithematic Operator

// Solidity contract to demonstrate

// Arithematic Operator

```
pragma solidity ^0.5.0;
// Creating a contract
contract SolidityTest {
// Initializing variables
uint16 public a = 20;
uint16 public b = 10;
// Initializing a variable
// with sum
uint public sum = a + b;
// Initializing a variable
// with the difference
uint public diff = a - b;
// Initializing a variable
// with product
uint public mul = a * b;
// Initializing a variable
// with quotient
uint public div = a / b;
// Initializing a variable
// with modulus
uint public mod = a % b;
// Initializing a variable
// decrement value
uint public dec = --b;
// Initializing a variable
// with increment value
uint public inc = ++a;
}
```



```
0: uint16: 21
0: uint16: 9
0: uint256: 9
0: uint256: 10
```

2.Relational Operator

// Solidity program to demonstrate

// Relational Operator

```
pragma solidity ^0.5.0;

// Creating a contract
contract SolidityTest {

// Declaring variables
uint16 public a = 20;
uint16 public b = 10;

// Initializing a variable
// with bool equal result
bool public eq = a == b;

// Initializing a variable
// with bool not equal result
bool public noteq = a != b;

// Initializing a variable
// with bool greater than result
bool public gtr = a > b;

// Initializing a variable
// with bool less than result
bool public les = a < b;

// Initializing a variable
// with bool greater than equal to result
bool public gtreq = a >= b;

// Initializing a variable
// bool less than equal to result
bool public leseq = a <= b;
}
```

3.Logical Operators

// Solidity program to demonstrate

// Logical Operators

```
pragma solidity ^0.5.0;

// Creating a contract
contract logicalOperator{

// Defining function to demonstrate
// Logical operator
function Logic(
bool a, bool b) public view returns(
bool, bool, bool){

// Logical AND operator
bool and = a&&b;

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

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

4.Bitwise Operators

// Solidity program to demonstrate

// Bitwise Operator

```
pragma solidity ^0.5.0;

// Creating a contract
contract SolidityTest {

// Declaring variables
uint16 public a = 20;
uint16 public b = 10;

// Initializing a variable
// to '&' value
uint16 public and = a & b;

// Initializing a variable
// to '|' value
uint16 public or = a | b;

// Initializing a variable
// to '^' value
uint16 public xor = a ^ b;

// Initializing a variable
// to '<<' value
uint16 public leftshift = a << b;

// Initializing a variable
// to '>>' value
uint16 public rightshift = a >> b;

// Initializing a variable
// to '~' value
uint16 public not = ~a ;
}
```

5.Assignment Operator

// Solidity program to demonstrate

// Assignment Operator

```
pragma solidity ^0.5.0;

// Creating a contract
contract SolidityTest {

// Declaring variables
uint16 public assignment = 20;
uint public assignment_add = 50;
uint public assign_sub = 50;
uint public assign_mul = 10;
uint public assign_div = 50;
uint public assign_mod = 32;

// Defining function to
// demonstrate Assignment Operator
function getResult() public{
assignment_add += 10;
assign_sub -= 20;
assign_mul *= 10;
assign_div /= 10;
assign_mod %= 20;
return ;
}
}
```

6.**Conditional Operators**

// Solidity program to demonstrate

// Conditional Operator

```
pragma solidity ^0.5.0;

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

C)Loops:

1.While loop: The most basic loop in Solidity is the **while** loop which would be discussed in this chapter. The purpose of a **while** loop is to execute a statement or code block repeatedly as long as an **expression** is true. Once the expression becomes **false,** the loop terminates.

2.do-while loop: The **do...while** loop is similar to the **while** loop except that the condition check happens at the end of the loop. This means that the loop will always be executed at least once, even if the condition is **false**.

3.for loop: The **for** loop is the most compact form of looping. It includes the following three important parts –

The **loop initialization** where we initialize our counter to a starting value. The initialization statement is executed before the loop begins.

The **test statement** which will test if a given condition is true or not. If the condition is true, then the code given inside the loop will be executed, otherwise the control will come out of the loop.

The **iteration statement** where you can increase or decrease your counter.

4.loop control: Solidity provides full control to handle loops and switch statements. There may be a situation when you need to come out of a loop without reaching its bottom. There may also be a situation when you want to skip a part of your code block and start the next iteration of the loop.To handle all such situations, Solidity provides **break** and **continue** statements. These statements are used to immediately come out of any loop or to start the next iteration of any loop respectively.

<u>1.While Loop</u>

```
pragma solidity ^0.5.0;
contract SolidityTest {
uint storedData;
constructor() public{
storedData = 10;
}
function getResult() public view returns(string memory){
uint a = 10;
uint b = 2;
uint result = a + b;
return integerToString(result);
}
function integerToString(uint _i) internal pure
returns (string memory) {
if (_i == 0) {
return "0";
}
uint j = _i;
uint len;
while (j != 0) {
len++;
j /= 10;
}
bytes memory bstr = new bytes(len);
uint k = len - 1;
while (_i != 0) { // while loop
bstr[k--] = byte(uint8(48 + _i % 10));
_i /= 10;
}
return string(bstr);
}
}
```

<u>2.Do-while loop:</u>

```
pragma solidity ^0.5.0;
contract SolidityTest {
uint storedData;
constructor() public{
storedData = 10;
}
function getResult() public view returns(string memory){
uint a = 10;
uint b = 2;
uint result = a + b;
return integerToString(result);
}
function integerToString(uint _i) internal pure
returns (string memory) {
if (_i == 0) {
return "0";
}
uint j = _i;
uint len;
while (j != 0) {
len++;
j /= 10;
}
bytes memory bstr = new bytes(len);
uint k = len - 1;
do {                // do while loop
bstr[k--] = byte(uint8(48 + _i % 10));
_i /= 10;
}
while (_i != 0);
return string(bstr);
}
}
```

<u>3.For Loop:</u>

```
pragma solidity ^0.5.0;

contract SolidityTest {
uint storedData;
constructor() public{
storedData = 10;
}

function getResult() public view returns(string memory){
```

```
uint a = 10;
uint b = 2;
uint result = a + b;
return integerToString(result);
}

function integerToString(uint _i) internal pure
returns (string memory) {
if (_i == 0) {
return "0";
}
uint j=0;
uint len;
for (j = _i; j != 0; j /= 10) {  //for loop example
len++;
}
bytes memory bstr = new bytes(len);
uint k = len - 1;
while (_i != 0) {
bstr[k--] = byte(uint8(48 + _i % 10));
_i /= 10;
}
return string(bstr);//access local variable
}}
```

4.loop Control: (Break statement)

```
pragma solidity ^0.5.0;

contract SolidityTest {
uint storedData;
constructor() public{
storedData = 10;
}
function getResult() public view returns(string memory){
uint a = 1;
uint b = 2;
uint result = a + b;
return integerToString(result);
}
function integerToString(uint _i) internal pure
returns (string memory) {

if (_i == 0) {
```

```
return "0";
}
uint j = _i;
uint len;

while (true) {
len++;
j /= 10;
if(j==0){
break;   //using break statement
}
}
bytes memory bstr = new bytes(len);
uint k = len - 1;

while (_i != 0) {
bstr[k--] = byte(uint8(48 + _i % 10));
_i /= 10;
}
return string(bstr);
}
}
```

(continue statement)

```
pragma solidity ^0.5.0;
contract SolidityTest {
uint storedData;
constructor() public{
storedData = 10;
}
function getResult() public view returns(string memory){
uint n = 1;
uint sum = 0;

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

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

while (true) {
len++;
j /= 10;
if(j==0){
break;   //using break statement
}
}
bytes memory bstr = new bytes(len);
uint k = len - 1;

while (_i != 0) {
bstr[k--] = byte(uint8(48 + _i % 10));
_i /= 10;
}
return string(bstr);
}
}
```

D) Decision Making:

While writing a program, there may be a situation when you need to adopt one out of a given set of paths. In such cases, you need to use conditional statements that allow your program to make correct decisions and perform right actions.Solidity supports conditional statements which are used to perform different actions based on different conditions. Here we will explain the **if..else** statement.

1.if statement: The **if** statement is the fundamental control statement that allows Solidity to make decisions and execute statements conditionally.

```
pragma solidity ^0.5.0;

contract SolidityTest {
uint storedData;
constructor() public {
```

```
storedData = 10;
}
function getResult() public view returns(string memory){
uint a = 1;
uint b = 2;
uint result = a + b;
return integerToString(result);
}
function integerToString(uint _i) internal pure
returns (string memory) {
if (_i == 0) {   // if statement
return "0";
}
uint j = _i;
uint len;

while (j != 0) {
len++;
j /= 10;
}
bytes memory bstr = new bytes(len);
uint k = len - 1;

while (_i != 0) {
bstr[k--] = byte(uint8(48 + _i % 10));
_i /= 10;
}
return string(bstr);//access local variable
}}
```

<u>**2.if-else statement:**</u> The **'if...else'** statement is the next form of control statement that allows Solidity to execute statements in a more controlled way.

```
pragma solidity ^0.5.0;

// Creating a contract
contract Types {
// Declaring state variables
uint i = 10;
bool even;

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

3.if-else..if statement: The **if...else if...** statement is an advanced form of **if...else** that allows Solidity to make a correct decision out of several conditions.

```
pragma solidity ^0.5.0;

// Creating a contract
contract Types {
// Declaring state variables
uint i = 12;
string result;
// Defining function to
// demonstrate the use
// of 'if...else if...else
// statement'
function decision_making (
) public returns(string memory){
if(i<10){
result = "less than 10";
}
else if(i == 10){
result = "equal to 10";
}
else{
result = "greater than 10";
}
return result;
}
}
```

String:

```
// Solidity program to demonstrate

// how to create a contract
pragma solidity ^0.4.23;

// Creating a contract
contract Test {
// Declaring variable
string  str;

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

Practical 3

AIM: write a solidity program for variables, operators, loops, decision making and string.

A) String:

Solidity supports String literal using both double quote (") and single quote ('). It provides string as a data type to declare a variable of type String.(Int to str)

```
pragma solidity ^0.5.0;

contract SolidityTest {
constructor() public{
}
function getResult() public view returns(string memory){
uint a = 1;
uint b = 2;
uint result = a + b;
return integerToString(result);
}
function integerToString(uint _i) internal pure
returns (string memory) {

if (_i == 0) {
```

```
return "0";
}
uint j = _i;
uint len;

while (j != 0) {
len++;
j /= 10;
}
bytes memory bstr = new bytes(len);
uint k = len - 1;

while (_i != 0) {
bstr[k--] = byte(uint8(48 + _i % 10));
_i /= 10;
}
return string(bstr);
}
}
```

B)Array:

Array is a data structure, which stores a fixed-size sequential collection of elements of the same type. An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.

// Solidity program to demonstrate

// accessing elements of an array

```
pragma solidity ^0.5.0;

// Creating a contract
contract Types {

// Declaring an array
uint[6] data;
uint x;

// Defining function to
// assign values to array
function array_example() public returns (uint[6] memory)
{

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

C)Enums:

Enums restrict a variable to have one of only a few predefined values. The values in this enumerated list are called enums. With the use of enums it is possible to reduce the number of bugs in your code.

// Solidity program to demonstrate

// how to use 'enumerator'

```
pragma solidity ^0.5.0;

// Creating a contract
contract Types {

// Creating an enumerator
enum week_days
{
Monday,
Tuesday,
Wednesday,
Thursday,
Friday,
Saturday,
Sunday
}
// Declaring variables of
// type enumerator
week_days week;

week_days choice;

// Setting a default value
week_days constant default_value
= week_days.Sunday;

// Defining a function to
// set value of choice
function set_value() public {
choice = week_days.Thursday;
}
// Defining a function to
// return value of choice
function get_choice(
) public view returns (week_days) {
return choice;
}
// Defining function to
// return default value
function getdefaultvalue(
) public pure returns(week_days) {
return default_value;
}
}
```

D)Structure:

Struct types are used to represent a record.

```
pragma solidity ^0.5.0;

contract test {
struct Book {
string title;
string author;
uint book_id;
}
Book book;

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

E)Mappings:

Mapping is a reference type as arrays and structs. Following is the syntax to declare a mapping type.

mapping(_KeyType => _ValueType) where ,

_KeyType – can be any built-in types plus bytes and string. No reference type or complex objects are allowed.

_ValueType – can be any type.

```
pragma solidity ^0.5.0;

contract LedgerBalance {
mapping(address => uint)  balance;

function updateBalance() public  returns(uint) {
balance[msg.sender]=30;
return balance[msg.sender];
}
}
```

Mapping program for String.

```
pragma solidity ^0.5.0;

contract LedgerBalance {
mapping(address => string)  name;

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

3B. WRITE A SOLIDITY PROGRAM FOR FUNCTION, VIEW FUNCTION, PURE FUNCTION & FALLBACK FUNCTION.

A)Function:

A function is a group of reusable code which can be called anywhere in your program. This eliminates the need of writing the same code again and again. It helps programmers in writing modular codes. Functions allow a programmer to divide a big program into a number of small and manageable functions.

```
pragma solidity ^0.5.0;

contract SolidityTest {
constructor() public{
}
function getResult() public view returns(string memory){
uint a = 1;
uint b = 2;
uint result = a + b;
return integerToString(result);
}
function integerToString(uint _i) internal pure
returns (string memory) {

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

while (j != 0) {
len++;
j /= 10;
}
bytes memory bstr = new bytes(len);
uint k = len - 1;

while (_i != 0) {
bstr[k--] = byte(uint8(48 + _i % 10));
_i /= 10;
}
return string(bstr);//access local variable

}
}
```

B)View Function:

View functions ensure that they will not modify the state. A function can be declared as **view**. Getter method are by default view functions.

```
pragma solidity ^0.5.0;
contract Test {
function getResult() public view returns(uint product, uint sum){
uint a = 1; // local variable
uint b = 2;
product = a * b;
sum = a + b;
}
}
```

C)Pure Function:

Pure functions ensure that they not read or modify the state. A function can be declared as **pure**. Pure functions can use the revert() and require() functions to revert potential state changes if an error occurs.

```
pragma solidity ^0.5.0;

contract Test {
function getResult() public pure returns(uint product, uint sum){
uint a = 1;
uint b = 2;
product = a * b;
sum = a + b;
}
}
```

D)Fallback Function:

Fallback function is a special function available to a contract.

```
pragma solidity ^0.5.0;
contract Test {
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));
}
}
```

Practical 4

A. write a solidity program for function overloading, mathematical function & cryptographic functions

Function Overloading:

The definition of the function must differ from each other by the types and/or the number of arguments in the argument list. You cannot overload function declarations that differ only by return type.

```
pragma solidity ^0.5.0;

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

}
```

Mathematical Function:

Solidity provides inbuilt mathematical functions as well.

```
pragma solidity ^0.5.0;

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

Cryptographic Function:

Solidity provides inbuilt cryptographic functions as well.

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

4B. WRITE A SOLIDITY PROGRAM FOR CONTRACT, INHERITANCE, CONSTRUCTORS, ABSTRACT CONTRACTS, INTERFACES, LIBRARIES, ASSEMBLY, EVENTS, ERROR HANDLING.

A)Contract:

Contract in Solidity is similar to a Class in C++. A Contract have following properties.

Constructor – A special function declared with constructor keyword which will be executed once per contract and is invoked when a contract is created.

State Variables – Variables per Contract to store the state of the contract.

Functions – Functions per Contract which can modify the state variables to alter the state of a contract.

// Calling function from external contract

```
pragma solidity ^0.5.0;
contract C {
//private state variable
uint private data;

//public state variable
uint public info;

//constructor
constructor() public {
info = 10;
}
//private function
function increment(uint a) private pure returns(uint) { return a + 1; }
//public function
function updateData(uint a) public { data = a; }
function getData() public view returns(uint) { return data; }
function compute(uint a, uint b) internal pure returns (uint) { return a + b; }
}
//Derived Contract
contract E is C {
uint private result;
C private c;

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

B)Inheritance:

Inheritance is a way to extend functionality of a contract. Solidity supports both single as well as multiple inheritance.

// Solidity program to

// demonstrate

// Single Inheritance

```
pragma solidity >=0.4.22 <0.6.0;

// Defining contract
contract parent{
// Declaring internal
// state variable
uint internal sum;

// Defining external function
// to set value of internal
// state variable sum
function setValue() external {
uint a = 20;
uint b = 20;
sum = a + b;
}
}
// Defining child contract
contract child is parent{

// Defining external function
// to return value of
// internal state variable sum
function getValue() external view returns(uint) {
return sum;
}
}
// Defining calling contract
```

```
contract caller {

// Creating child contract object
child cc = new child();

// Defining function to call
// setValue and getValue functions
function testInheritance() public {
cc.setValue();
}
function result() public view returns(uint ){
return cc.getValue();
}
}
```

C)Constructors:

Constructor is a special function declared using constructor keyword. It is an optional function and is used to initialize state variables of a contract. Following are the key characteristics of a constructor.

A contract can have only one constructor.

A constructor code is executed once when a contract is created and it is used to initialize contract state.

A constructor can be either public or internal.

An internal constructor marks the contract as abstract.

In case, no constructor is defined, a default constructor is present in the contract.

```
pragma solidity ^0.5.0;
contract Base {
uint data;
constructor(uint _data) public {
data = _data;
}
function getresult()public view returns(uint){
return data;
}
}
contract Derived is Base (5) {
constructor() public {}
}
```

```
// Indirect Initialization of Base Constructor
pragma solidity ^0.5.0;

contract Base {
uint data;
constructor(uint _data) public {
data = _data;
}
function getresult()public view returns(uint){
return data;
}
}
contract Derived is Base {
constructor(uint _info) Base(_info * _info) public {}
}
```

D)Abstract Contracts:

Abstract Contract is one which contains at least one function without any implementation. Such a contract is used as a base contract. Generally an abstract contract contains both implemented as well as abstract functions. Derived contract will implement the abstract function and use the existing functions as and when required.

```
pragma solidity ^0.5.0;

contract Calculator {
function getResult() public view returns(uint);
}
contract Test is Calculator {
function getResult() public view returns(uint) {
uint a = 4;
uint b = 2;
uint result = a + b;
return result;
}
}
```

E)Interfaces:

Interfaces are similar to abstract contracts and are created using interface keyword. Following are the key characteristics of an interface.

Interface can not have any function with implementation.

Functions of an interface can be only of type external.

Interface can not have constructor.

Interface can not have state variables.

```
pragma solidity ^0.5.0;

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

F)Libraries:

Libraries are similar to Contracts but are mainly intended for reuse. A Library contains functions which other contracts can call. Solidity have certain restrictions on use of a Library.

```
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 Test {
uint[] data;
uint value;
uint index;
constructor() public {
data.push(6);
data.push(7);
data.push(8);
data.push(9);
data.push(10);
}
function isValuePresent() external {
value = 9;
//search if value is present in the array using Library function
index = Search.indexOf(data, value);
}
function getresult() public view returns(uint){
return index;
}}
```

G)Assembly:

Solidity provides an option to use assembly language to write inline assembly within Solidity source code. We can also write a standalone assembly code which then be converted to bytecode. Standalone Assembly is an intermediate language for a Solidity compiler and it converts the Solidity code into a Standalone Assembly and then to byte code. We can used the same language used in Inline Assembly to write code in a Standalone assembly.

```
pragma solidity ^0.5.0;

library Sum {
function sumUsingInlineAssembly(uint[] memory _data) public pure returns (uint o_sum) {
for (uint i = 0; i < _data.length; ++i) {
assembly {
o_sum := add(o_sum, mload(add(add(_data, 0x20), mul(i, 0x20))))
}}
}
}
contract Test {
uint[] data;
constructor() public {
data.push(1);
data.push(2);
data.push(3);
data.push(4);
data.push(5);
}
function sum() external view returns(uint){
return Sum.sumUsingInlineAssembly(data);
}
}
```

H)Events:

Event is an inheritable member of a contract. An event is emitted, it stores the arguments passed in transaction logs. These logs are stored on blockchain and are accessible using address of the contract till the contract is present on the blockchain. An event generated is not accessible from within contracts, not even the one which have created and emitted them.

// Solidity program to demonstrate

// creating an event

```
pragma solidity ^0.4.21;

// Creating a contract
contract eventExample {

// Declaring state variables
uint256 public value = 0;

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

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

I)Error Handling:

Solidity provides various functions for error handling. Generally when an error occurs, the state is reverted back to its original state. Other checks are to prevent unauthorized code access.

Solidity program to demonstrate require statement.

// Solidity program to

// demonstrate require

// statement

```
pragma solidity ^0.5.0;
// Creating a contract
contract requireStatement {
// Defining function to
// check input
function checkInput(uint8 _input) public view returns(string memory){
require(_input >= 0, "invalid uint");
require(_input <= 255, "invalid uint8");

return "Input is Uint8";
}
// Defining function to
// use require statement
function Odd(uint _input) public view returns(bool){
require(_input % 2 != 0);
return true;
}
}
```

Solidity program to demonstrate assert statement.

// Solidity program to

// demonstrate assert

// statement

```
pragma solidity ^0.5.0;

// Creating a contract
contract assertStatement {
// Defining a state variable
bool result;
// Defining a function
// to check condition
function checkOverflow(uint8 _num1, uint8 _num2) public {
uint8 sum = _num1 + _num2;
assert(sum<=255);
result = true;
}
// Defining a function to
// print result of assert
// statement
function getResult() public view returns(string memory){
if(result == true){
return "No Overflow";
}
else{
return "Overflow exist";
}
}
}
```

Solidity program to demonstrate revert statement.

// Solidity program to

// demonstrate revert

```
pragma solidity ^0.5.0;
// Creating a contract
contract revertStatement {
// Defining a function
// to check condition
function checkOverflow(uint _num1, uint _num2) public view returns(
string memory, uint) {
uint sum = _num1 + _num2;
if(sum < 0 || sum > 255){
revert(" Overflow Exist");
}
else{
return ("No Overflow", sum);
}
}
}
```

Practical 5

Aim: Install hyperledger-Irhoa

Step 1: install docker

```
sudo apt-get install curl

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu
$(lsb_release -cs) stable"

sudo apt-get update

apt-cache policy docker-ce

sudo apt-get install -y docker-ce
```

Step 2: create docker network

```
sudo docker network create mithilesh-iroha-network
```

Step 3:add PostgreSQL to our network

```
sudo docker run --name some-postgres -e POSTGRES_USER=postgres -e
POSTGRES_PASSWORD=mysecretpassword -p 5432:5432 --network=mithilesh-iroha-network -
d postgres:9.5
```

Step 4:create a volume of persistant storage named "blockstore" to store the blocks for our blockchain

```
sudo docker volume create blockstore
```

Step 5: Download the Iroha code from github.

```
sudo apt-get install git
git clone -b develop https://github.com/hyperledger/iroha --depth=1
```

Step 6: run the Iroha docker container

```
sudo docker run -it --name iroha \
-p 50051:50051 \
-v $(pwd)/iroha/example:/opt/iroha_data \
-v blockstore:/tmp/block_store \
--network=mithilesh-iroha-network \
--entrypoint=/bin/bash \
hyperledger/iroha:latest
```

Step 7: run Iroha
irohad --config config.docker --genesis_block genesis.block --keypair_name node0

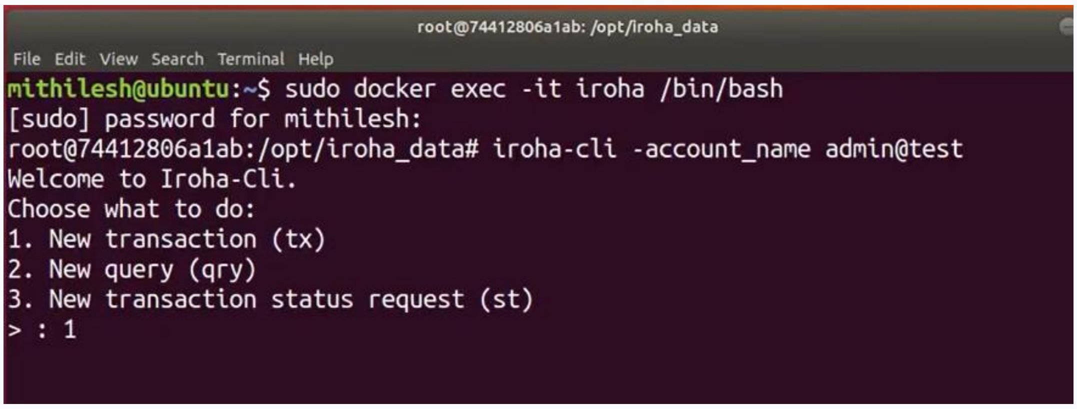
Step 8:Open a new terminal

Step 9:attach the docker container to our terminal
sudo docker exec -it iroha /bin/bash

Step 10:Launch the iroha-cli tool and login as admin@test.

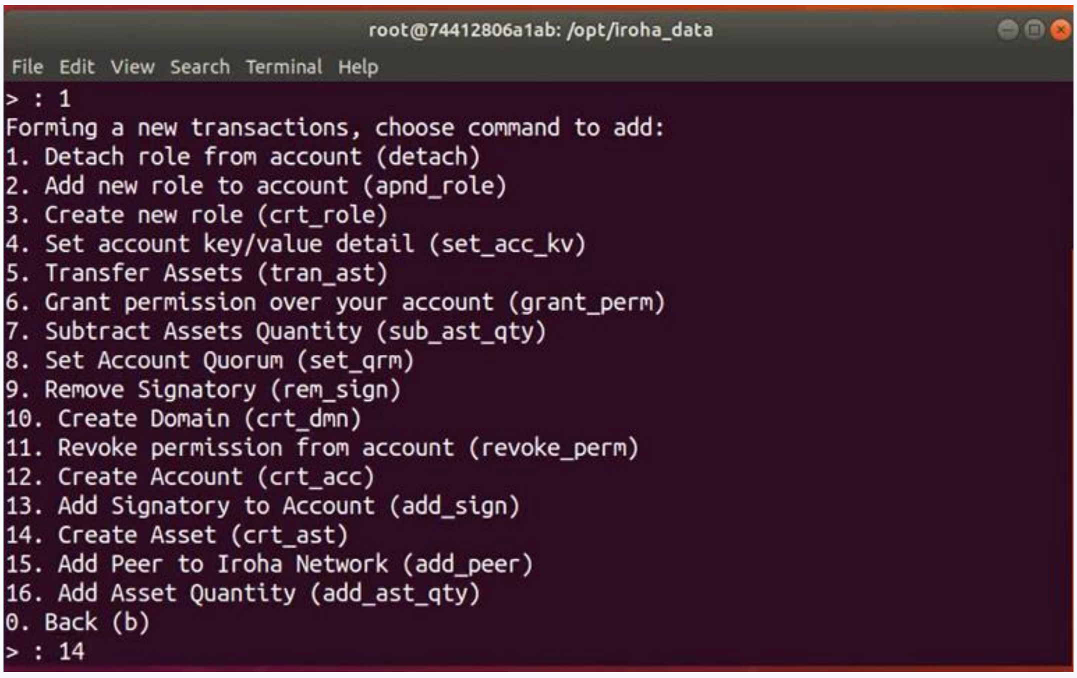
iroha-cli -account_name admin@test

Select 1 – for new transaction



```
mithilesh@ubuntu:~$ sudo docker exec -it iroha /bin/bash
[sudo] password for mithilesh:
root@74412806a1ab:/opt/iroha_data# iroha-cli -account_name admin@test
Welcome to Iroha-Cli.
Choose what to do:
1. New transaction (tx)
2. New query (qry)
3. New transaction status request (st)
> : 1
```

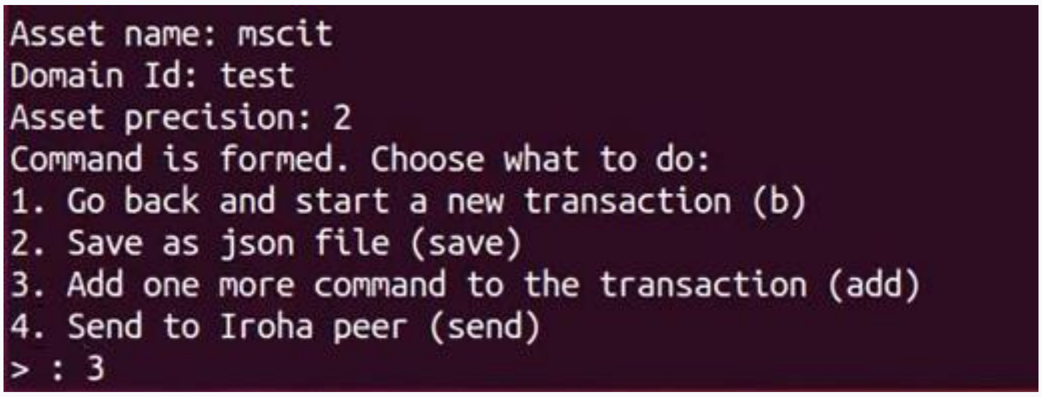
Select 14- for creating new coin



```
> : 1
Forming a new transactions, choose command to add:
1. Detach role from account (detach)
2. Add new role to account (apnd_role)
3. Create new role (crt_role)
4. Set account key/value detail (set_acc_kv)
5. Transfer Assets (tran_ast)
6. Grant permission over your account (grant_perm)
7. Subtract Assets Quantity (sub_ast_qty)
8. Set Account Quorum (set_qrm)
9. Remove Signatory (rem_sign)
10. Create Domain (crt_dmn)
11. Revoke permission from account (revoke_perm)
12. Create Account (crt_acc)
13. Add Signatory to Account (add_sign)
14. Create Asset (crt_ast)
15. Add Peer to Iroha Network (add_peer)
16. Add Asset Quantity (add_ast_qty)
0. Back (b)
> : 14
```

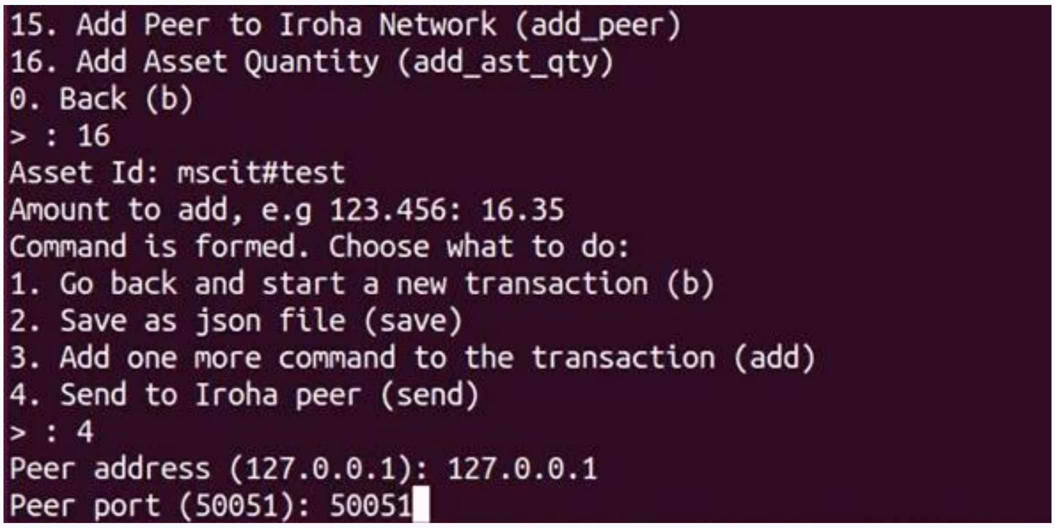
Now type Asset name: mscit
Domain id: test
Asset precision: 2

And select option 3 to add more command

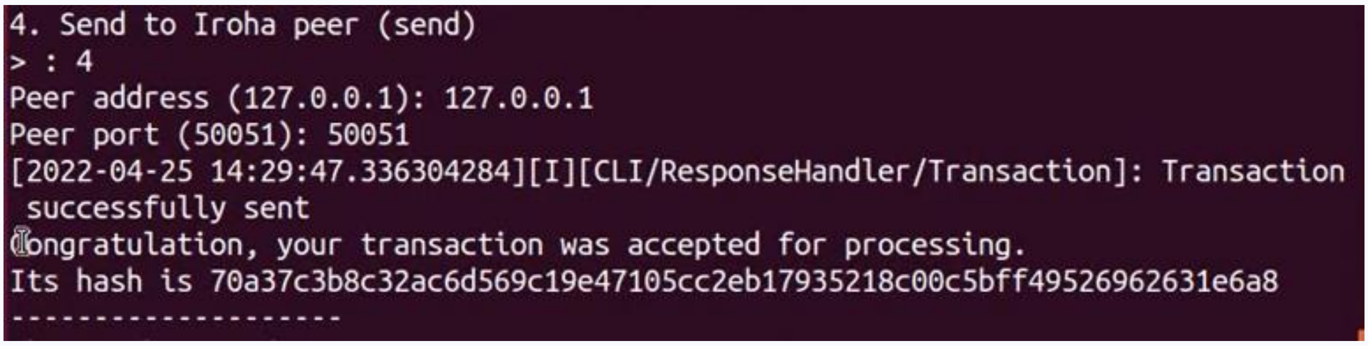


```
Asset name: mscit
Domain Id: test
Asset precision: 2
Command is formed. Choose what to do:
1. Go back and start a new transaction (b)
2. Save as json file (save)
3. Add one more command to the transaction (add)
4. Send to Iroha peer (send)
> : 3
```

Now select option 16 to add asset quantity
Asset id: mscit#test
Amount: 16.35
Select option 4- send request to Iroha peer

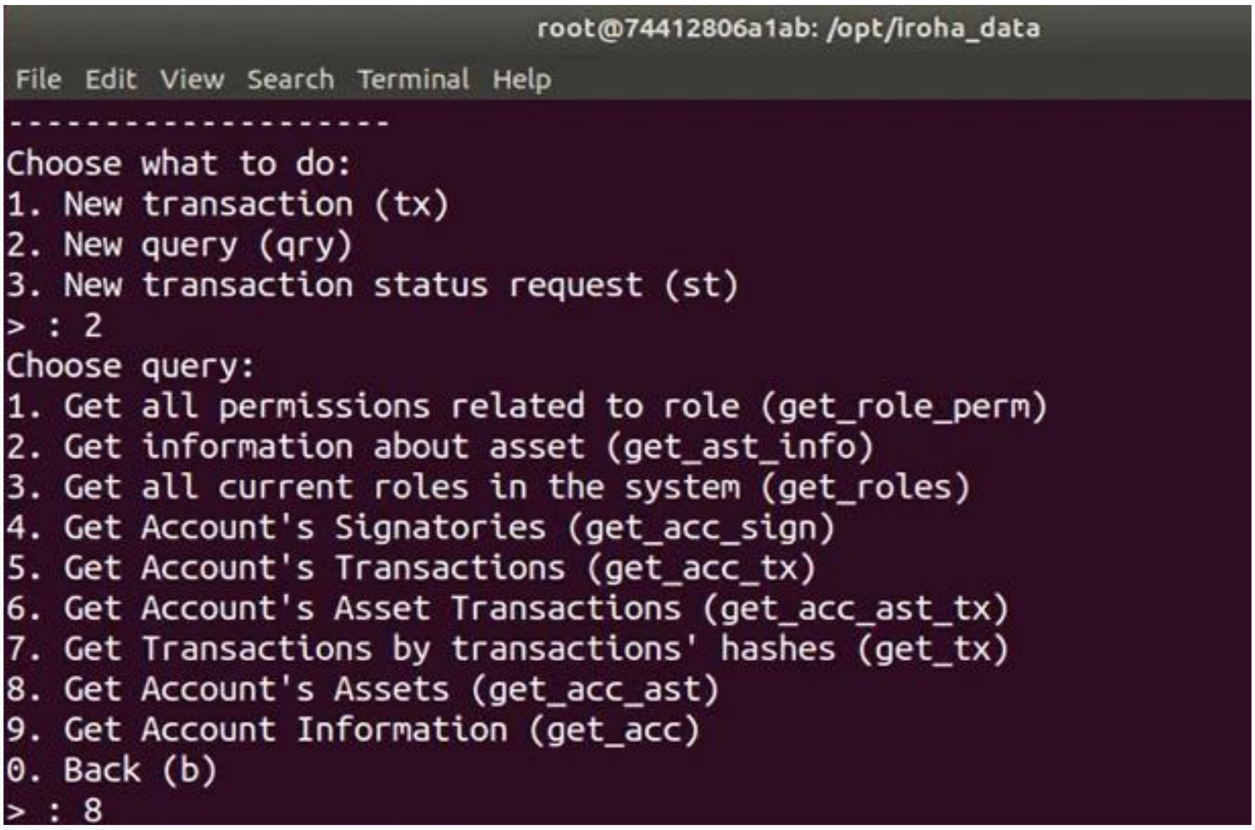


```
15. Add Peer to Iroha Network (add_peer)
16. Add Asset Quantity (add_ast_qty)
0. Back (b)
> : 16
Asset Id: mscit#test
Amount to add, e.g 123.456: 16.35
Command is formed. Choose what to do:
1. Go back and start a new transaction (b)
2. Save as json file (save)
3. Add one more command to the transaction (add)
4. Send to Iroha peer (send)
> : 4
Peer address (127.0.0.1): 127.0.0.1
Peer port (50051): 50051
```



```
4. Send to Iroha peer (send)
> : 4
Peer address (127.0.0.1): 127.0.0.1
Peer port (50051): 50051
[2022-04-25 14:29:47.336304284][I][CLI/ResponseHandler/Transaction]: Transaction
 successfully sent
Congratulation, your transaction was accepted for processing.
Its hash is 70a37c3b8c32ac6d569c19e47105cc2eb17935218c00c5bff49526962631e6a8
--------------------
```

Select option 2 –for query
Select option 8- for assets

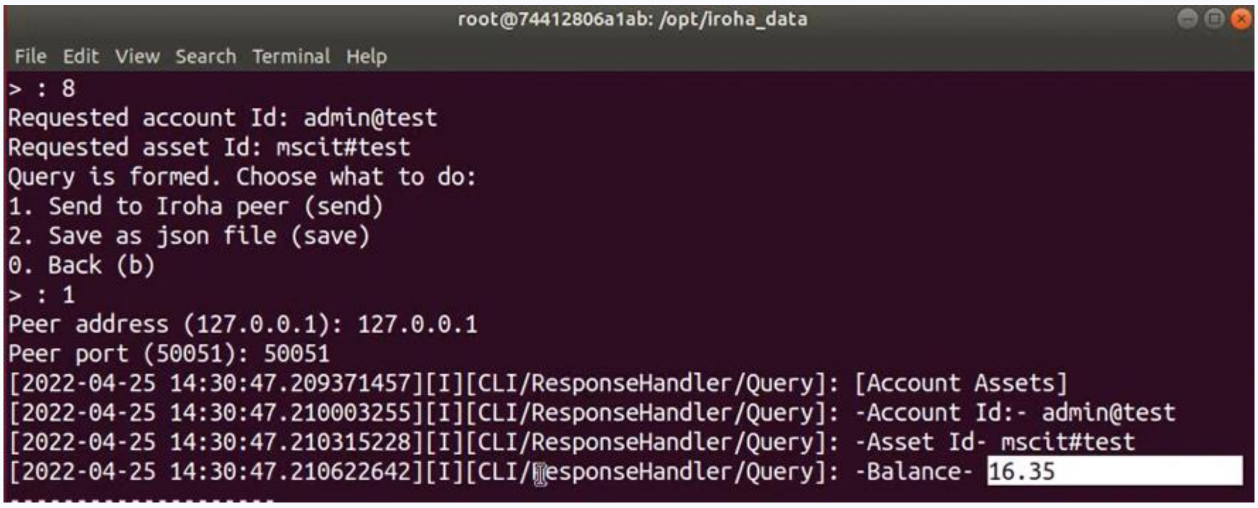

root@74412806a1ab: /opt/iroha_data
File Edit View Search Terminal Help

```
--------------------
Choose what to do:
1. New transaction (tx)
2. New query (qry)
3. New transaction status request (st)
> : 2
Choose query:
1. Get all permissions related to role (get_role_perm)
2. Get information about asset (get_ast_info)
3. Get all current roles in the system (get_roles)
4. Get Account's Signatories (get_acc_sign)
5. Get Account's Transactions (get_acc_tx)
6. Get Account's Asset Transactions (get_acc_ast_tx)
7. Get Transactions by transactions' hashes (get_tx)
8. Get Account's Assets (get_acc_ast)
9. Get Account Information (get_acc)
0. Back (b)
> : 8
```

Select option 1

Enter peer : 127.0.0.1

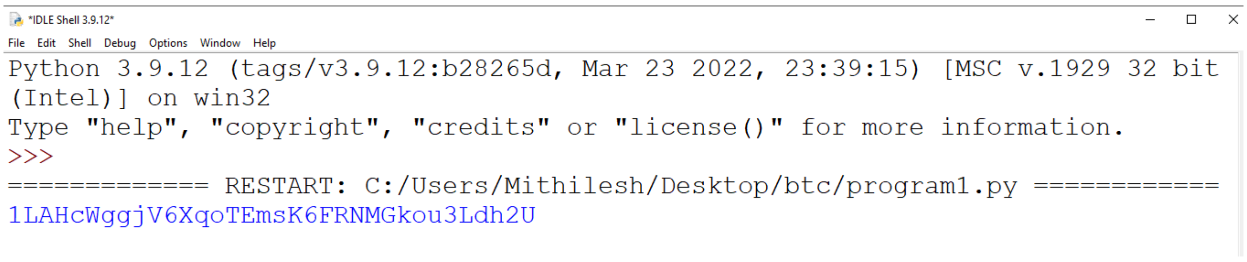
Port: 50051


root@74412806a1ab: /opt/iroha_data
File Edit View Search Terminal Help

```
> : 8
Requested account Id: admin@test
Requested asset Id: mscit#test
Query is formed. Choose what to do:
1. Send to Iroha peer (send)
2. Save as json file (save)
0. Back (b)
> : 1
Peer address (127.0.0.1): 127.0.0.1
Peer port (50051): 50051
[2022-04-25 14:30:47.209371457][I][CLI/ResponseHandler/Query]: [Account Assets]
[2022-04-25 14:30:47.210003255][I][CLI/ResponseHandler/Query]: -Account Id:- admin@test
[2022-04-25 14:30:47.210315228][I][CLI/ResponseHandler/Query]: -Asset Id- mscit#test
[2022-04-25 14:30:47.210622642][I][CLI/ResponseHandler/Query]: -Balance- 16.35
```

Aim: Demonstrate the use of Bitcoin Core API.

```
#pip install bitcoinlib
from bitcoinlib.wallets import Wallet
w = Wallet.create('Wallet1')
key1 = w.get_key()
print(key1.address)
w.scan()
print(w.info())
```



```
Python 3.9.12 (tags/v3.9.12:b28265d, Mar 23 2022, 23:39:15) [MSC v.1929 32 bit
(Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
============= RESTART: C:/Users/Mithilesh/Desktop/btc/program1.py ============
1LAHcWggjV6XqoTEmsK6FRNMGkou3Ldh2U
```

More detail on: https://pypi.org/project/bitcoinlib/

Practical 6

Aim: Create your own blockchain and demonstrate its use.

Install on Ubuntu via PPAs

The easiest way to install go-ethereum on Ubuntu-based distributions is with the built-in launchpad PPAs (Personal Package Archives). We provide a single PPA repository that contains both our stable and development releases for Ubuntu versions trusty, xenial, zesty and artful.

linux:

To enable our launchpad repository run:

Step 1: open new terminal

Step 2: on terminal type this command

sudo add-apt-repository -y ppa:ethereum/ethereum

#if above command gives error then run

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

Step 3: install the stable version of go-ethereum:

sudo apt-get update

sudo apt-get install ethereum

Step 4: create new directory for storing blockchain data

mkdir myblockchain

cd myblockchain

Step 5:Create genesis.json file

sudo nano genesis.json

```
{
  "config": {
    "chainId": 10,
    "homesteadBlock": 0,
    "eip155Block": 0,
    "eip158Block": 0,
    "eip150Block": 0,
    "eip150Hash": "0x0000000000000000000000000000000000000000000000000000000000000000"
  },
  "alloc": {},
  "coinbase": "0x0000000000000000000000000000000000000000",
  "difficulty": "0x02000000",
  "extraData": "",
  "gasLimit": "0x2fefd8",
  "nonce": "0x0000000000000042",
  "mixhash": "0x0000000000000000000000000000000000000000000000000000000000000000",
  "parentHash": "0x0000000000000000000000000000000000000000000000000000000000000000",
  "timestamp": "0x00"
}
```

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

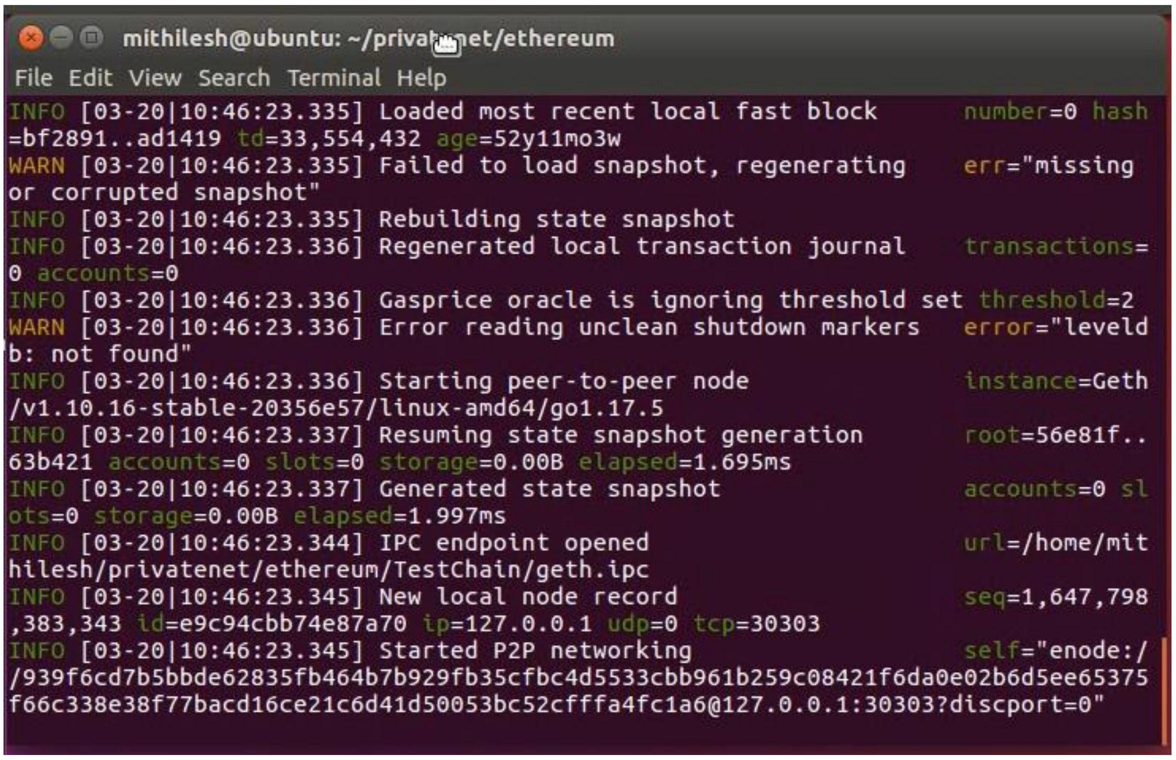
Step 6: initialize the block

sudo geth --datadir TestChain init genesis.json

Step 7: create network

sudo geth –datadir TestChain –networkid 1234

[do not close this terminal]



```
mithilesh@ubuntu: ~/privatenet/ethereum
File  Edit  View  Search  Terminal  Help
INFO [03-20|10:46:23.335] Loaded most recent local fast block      number=0 hash
=bf2891..ad1419 td=33,554,432 age=52y11mo3w
WARN [03-20|10:46:23.335] Failed to load snapshot, regenerating    err="missing
or corrupted snapshot"
INFO [03-20|10:46:23.335] Rebuilding state snapshot
INFO [03-20|10:46:23.336] Regenerated local transaction journal    transactions=
0 accounts=0
INFO [03-20|10:46:23.336] Gasprice oracle is ignoring threshold set threshold=2
WARN [03-20|10:46:23.336] Error reading unclean shutdown markers   error="leveld
b: not found"
INFO [03-20|10:46:23.336] Starting peer-to-peer node               instance=Geth
/v1.10.16-stable-20356e57/linux-amd64/go1.17.5
INFO [03-20|10:46:23.337] Resuming state snapshot generation       root=56e81f..
63b421 accounts=0 slots=0 storage=0.00B elapsed=1.695ms
INFO [03-20|10:46:23.337] Generated state snapshot                 accounts=0 sl
ots=0 storage=0.00B elapsed=1.997ms
INFO [03-20|10:46:23.344] IPC endpoint opened                      url=/home/mit
hilesh/privatenet/ethereum/TestChain/geth.ipc
INFO [03-20|10:46:23.345] New local node record                    seq=1,647,798
,383,343 id=e9c94cbb74e87a70 ip=127.0.0.1 udp=0 tcp=30303
INFO [03-20|10:46:23.345] Started P2P networking                   self="enode:/
/939f6cd7b5bbde62835fb464b7b929fb35cfbc4d5533cbb961b259c08421f6da0e02b6d5ee65375
f66c338e38f77bacd16ce21c6d41d50053bc52cfffa4fc1a6@127.0.0.1:30303?discport=0"
```

///

Step 8: open new terminal 2:

cd myblockchain

Step 9: attach geth to the network

sudo geth attach TestChain/geth.ipc

Step 10: on geth terminal type these commands

personal.newAccount("123456")

miner.start()

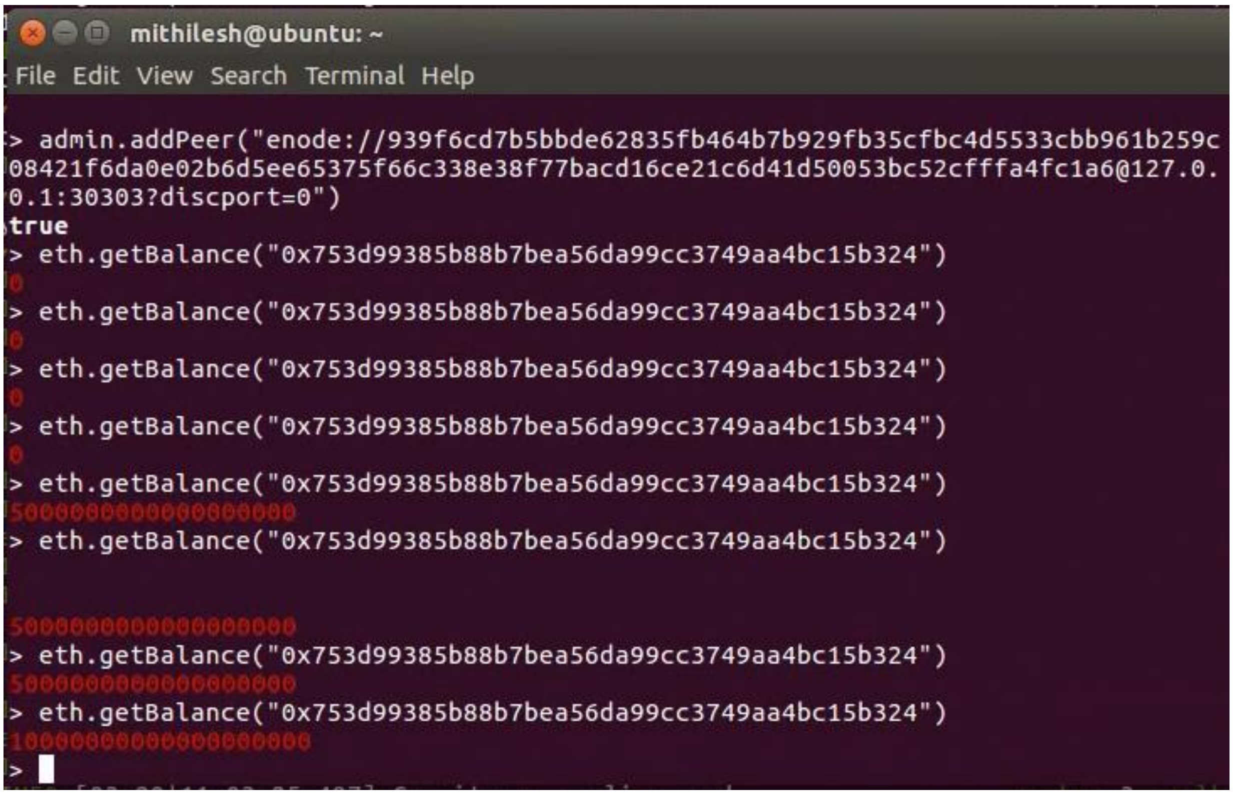
miner.setEtherbase(eth.accounts[0])

admin.addPeer(admin.nodeInfo.enode)

Step 10: Wait for 10-20 minutes and check balance

eth.getBalance(eth.accounts[0])

if ether balance is 0 wait for 10-20minutes for mining process to get complete and run **eth.getBalance(eth.accounts[0]) again.**



```
mithilesh@ubuntu: ~
File  Edit  View  Search  Terminal  Help

> admin.addPeer("enode://939f6cd7b5bbde62835fb464b7b929fb35cfbc4d5533cbb961b259c
08421f6da0e02b6d5ee65375f66c338e38f77bacd16ce21c6d41d50053bc52cfffa4fc1a6@127.0.
0.1:30303?discport=0")
true
> eth.getBalance("0x753d99385b88b7bea56da99cc3749aa4bc15b324")
0

> eth.getBalance("0x753d99385b88b7bea56da99cc3749aa4bc15b324")
0

> eth.getBalance("0x753d99385b88b7bea56da99cc3749aa4bc15b324")
0

> eth.getBalance("0x753d99385b88b7bea56da99cc3749aa4bc15b324")
0

> eth.getBalance("0x753d99385b88b7bea56da99cc3749aa4bc15b324")
5000000000000000000
> eth.getBalance("0x753d99385b88b7bea56da99cc3749aa4bc15b324")



5000000000000000000
> eth.getBalance("0x753d99385b88b7bea56da99cc3749aa4bc15b324")
5000000000000000000
> eth.getBalance("0x753d99385b88b7bea56da99cc3749aa4bc15b324")
10000000000000000000
> 
```

After balance is updated you can check current block height

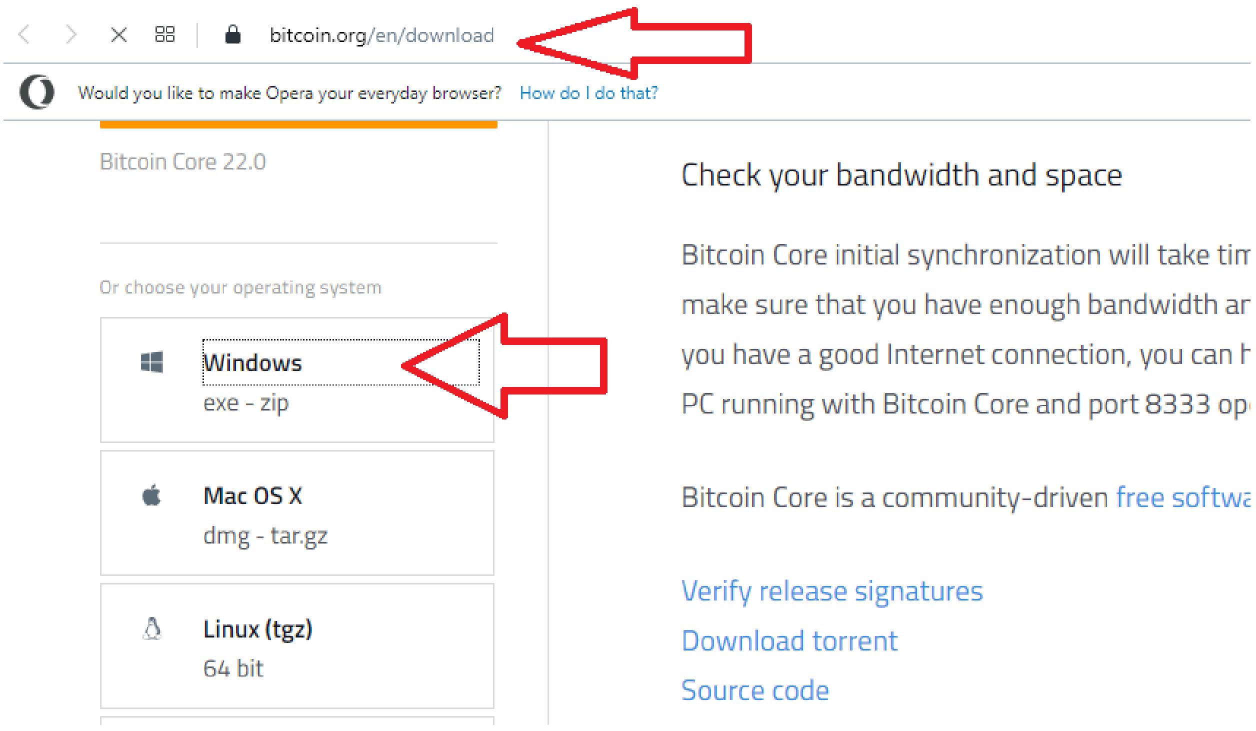
eth.blockNumber

Practical 7

Aim: Demonstrate the running of the blockchain node.

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

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


bitcoin.org/en/download

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

Bitcoin Core 22.0

Or choose your operating system

Windows
exe - zip

Mac OS X
dmg - tar.gz

Linux (tgz)
64 bit

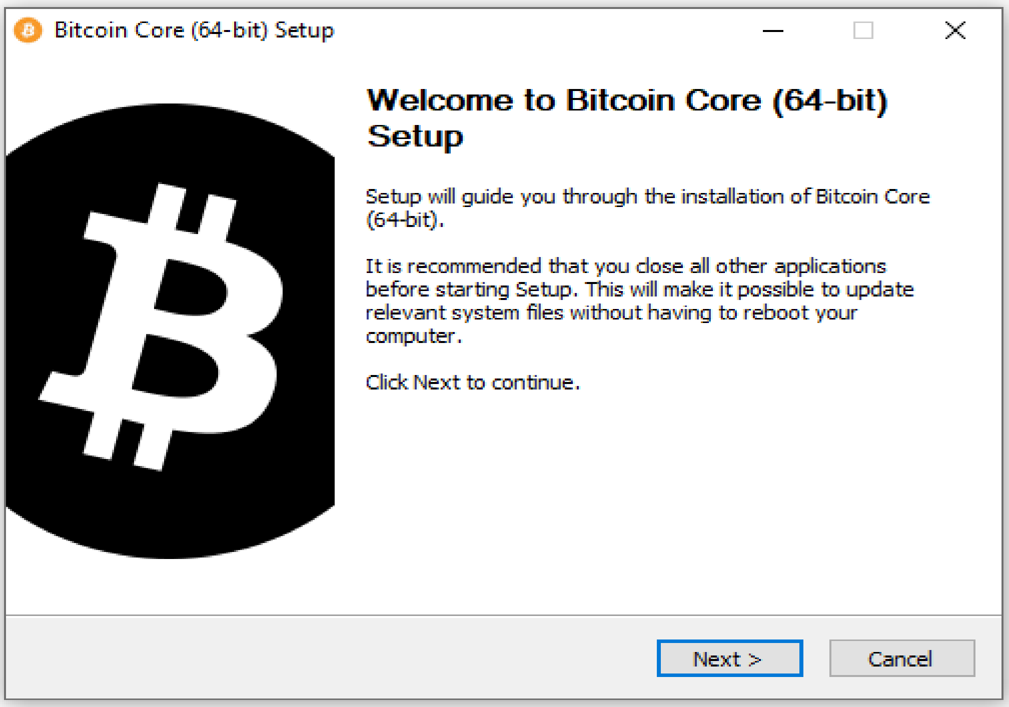
Check your bandwidth and space

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

Bitcoin Core is a community-driven free softwa

Verify release signatures
Download torrent
Source code

Step 3: Run the setup file-> click next


Bitcoin Core (64-bit) Setup

Welcome to Bitcoin Core (64-bit) Setup

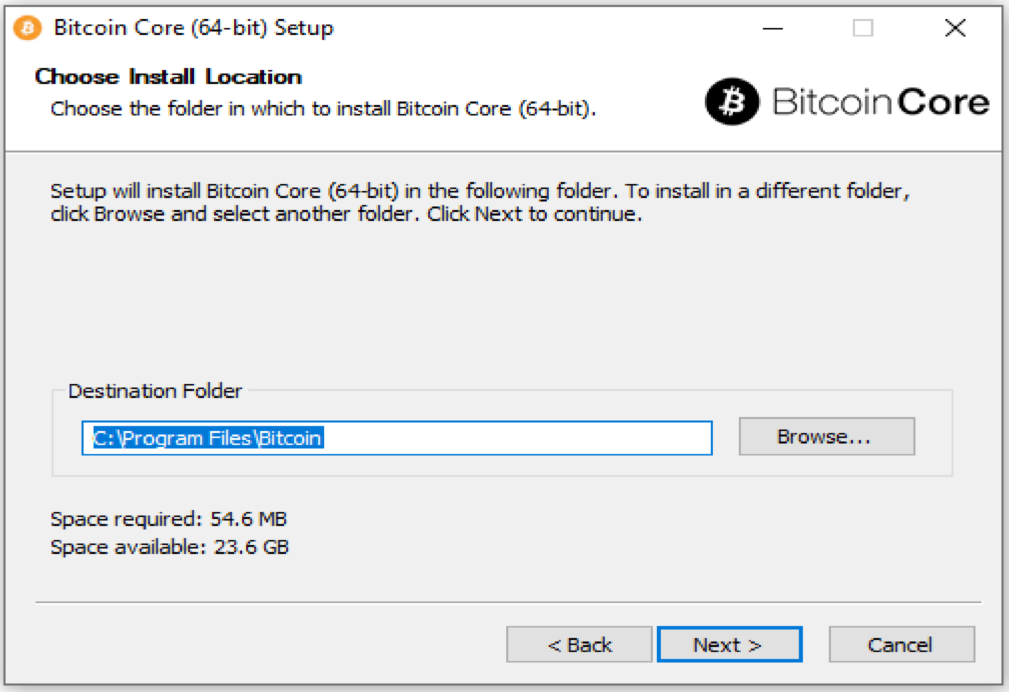
Setup will guide you through the installation of Bitcoin Core (64-bit).

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

Click Next to continue.

Next > Cancel

Step 4: Click Next



Bitcoin Core (64-bit) Setup

Choose Install Location

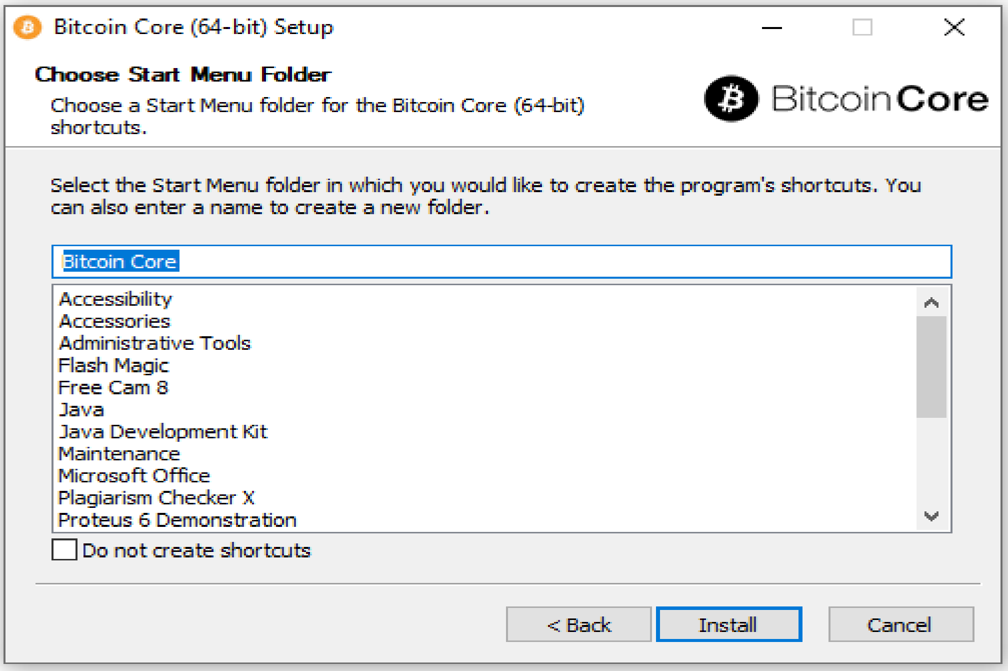
Choose the folder in which to install Bitcoin Core (64-bit).

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

Destination Folder: C:\Program Files\Bitcoin

Space required: 54.6 MB
Space available: 23.6 GB

Step 5: Finally click on Install



Bitcoin Core (64-bit) Setup

Choose Start Menu Folder

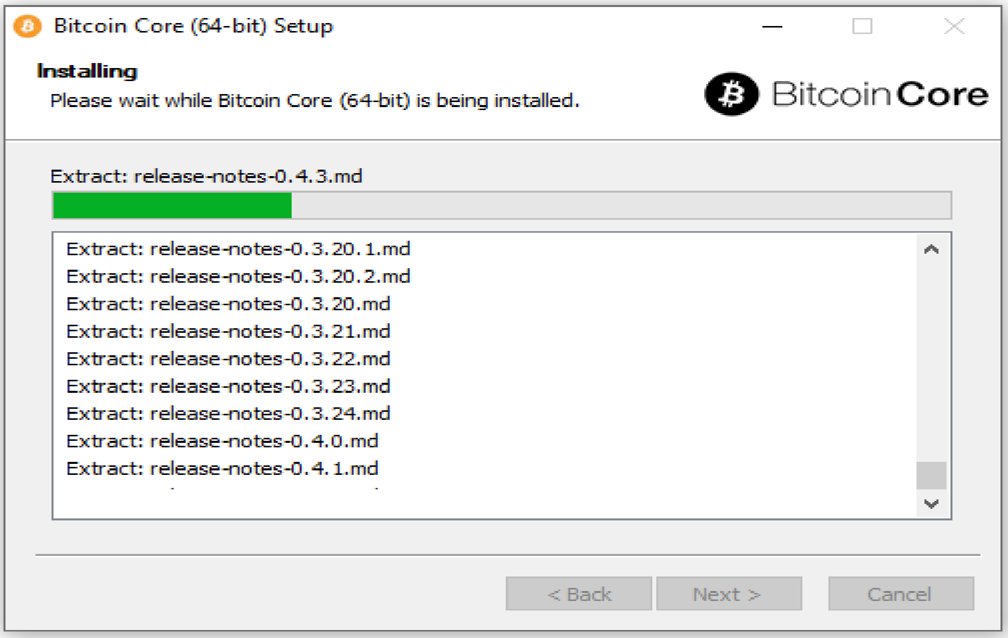
Choose a Start Menu folder for the Bitcoin Core (64-bit) shortcuts.

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

Bitcoin Core

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

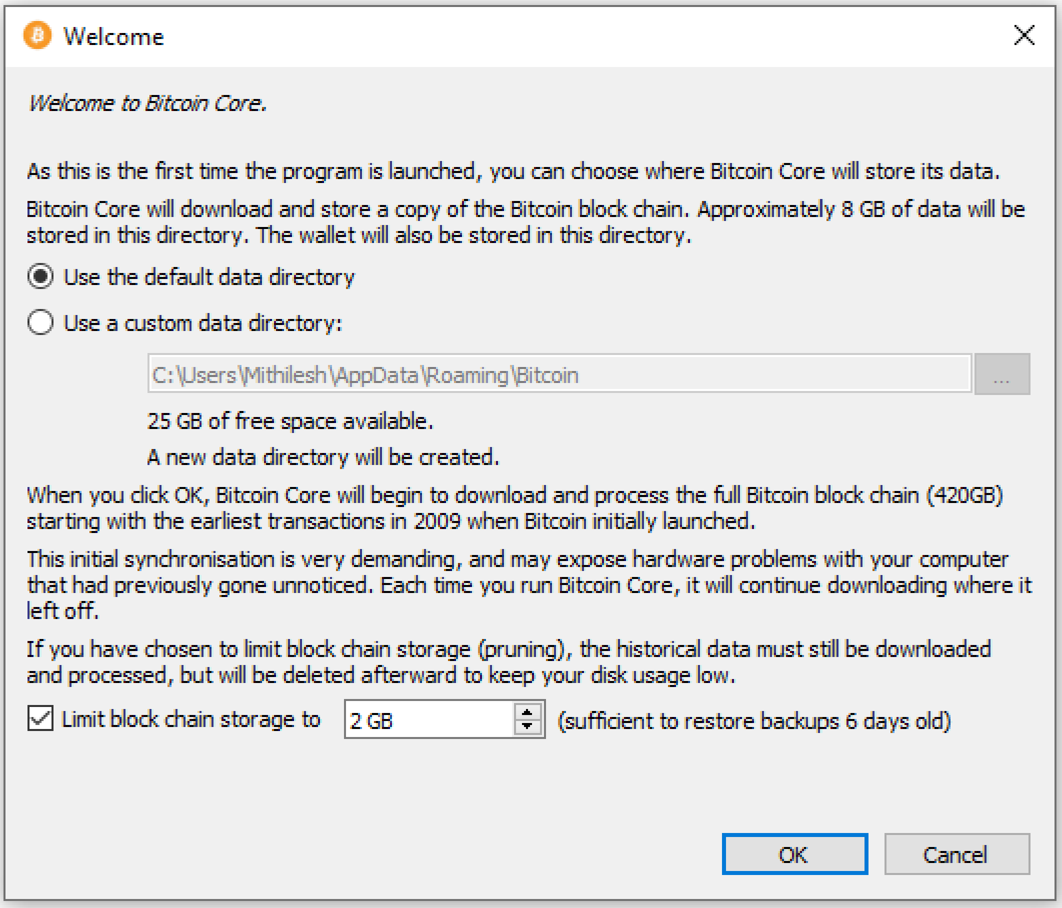
☐ Do not create shortcuts



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

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

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

Launch Bitcoin Core-> Click OK.



Welcome to Bitcoin Core.

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

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

[✓ Use the default data directory]

Use a custom data directory:

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

25 GB of free space available.

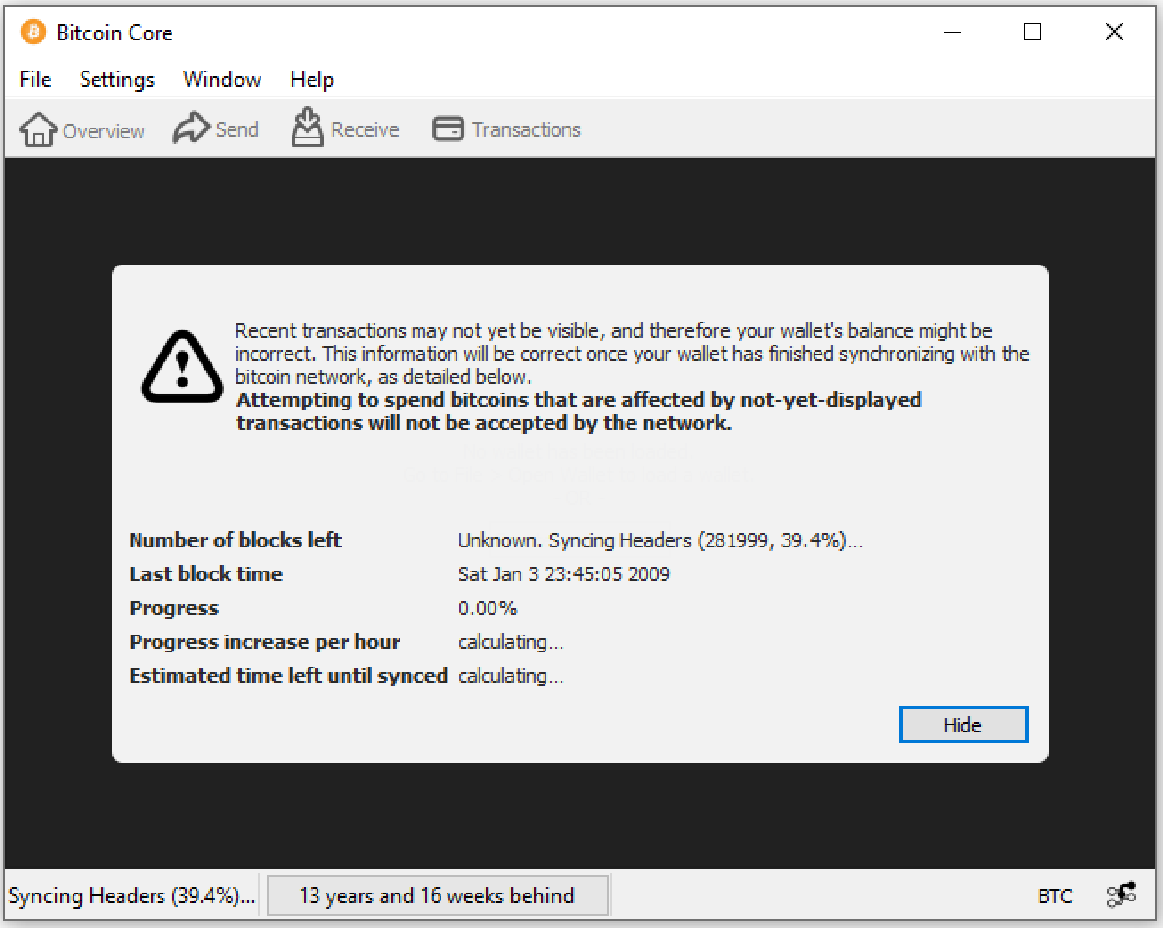
A new data directory will be created.

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

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

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

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

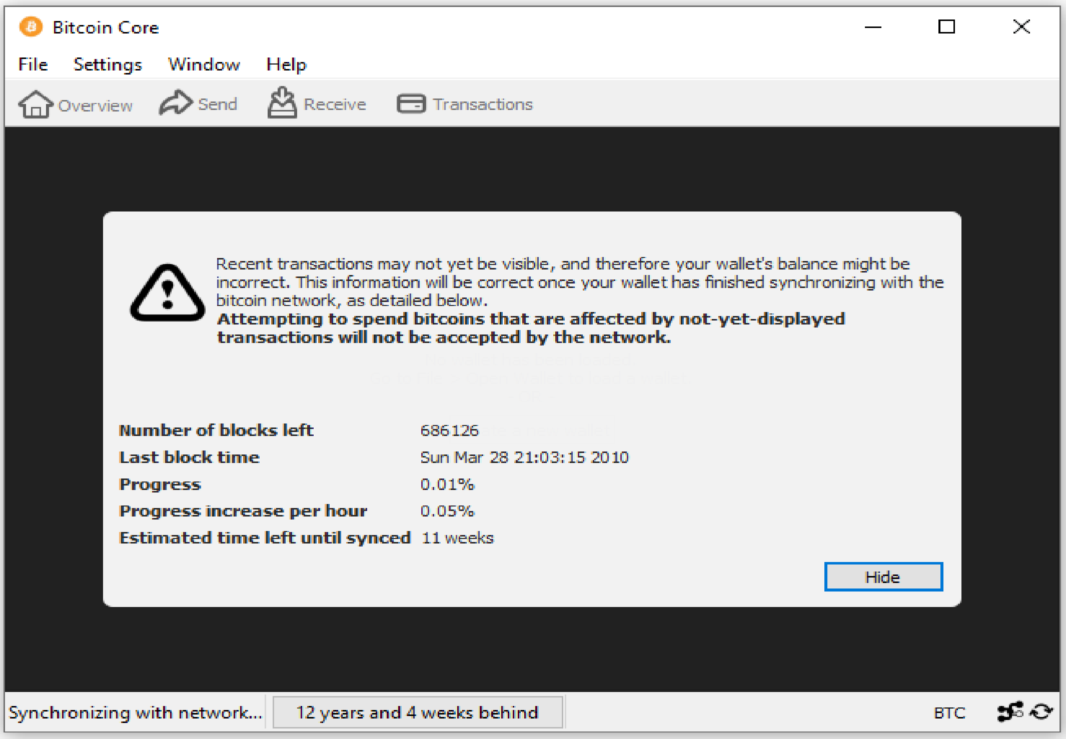


Recent transactions may not yet be visible, and therefore your wallet's balance might be incorrect. This information will be correct once your wallet has finished synchronizing with the bitcoin network, as detailed below.
Attempting to spend bitcoins that are affected by not-yet-displayed transactions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

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

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

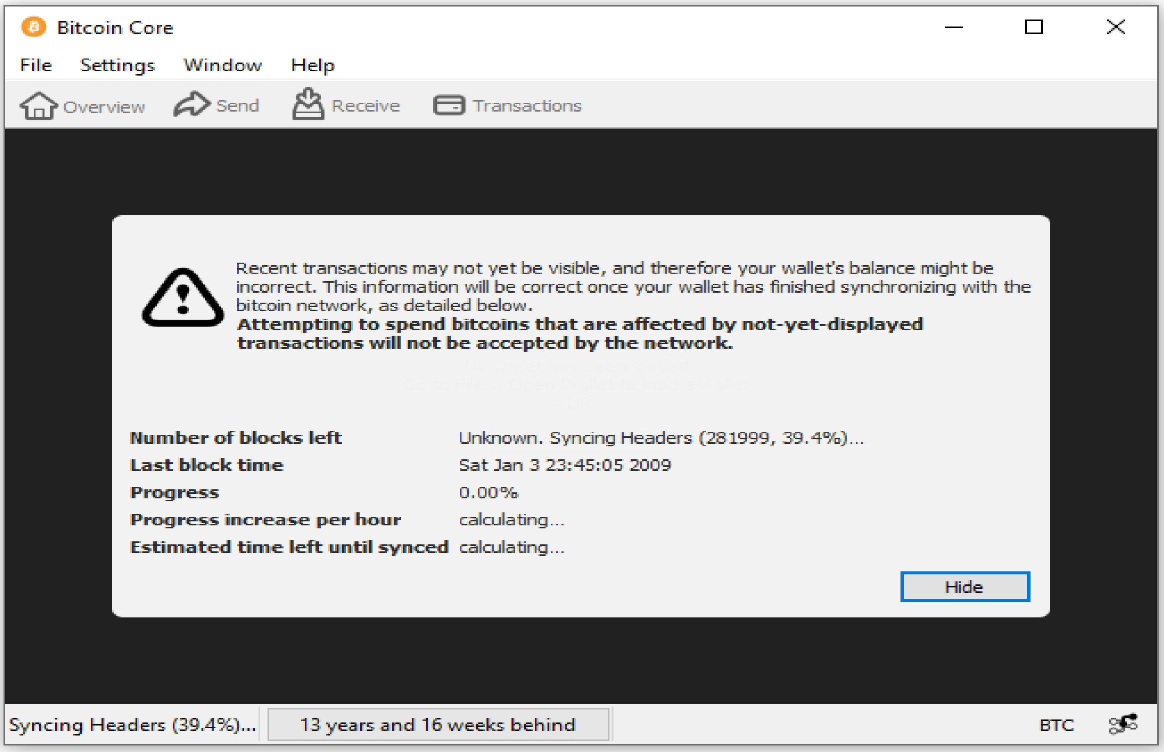
Click on Hide button [Synchronization take place in background]



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

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

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

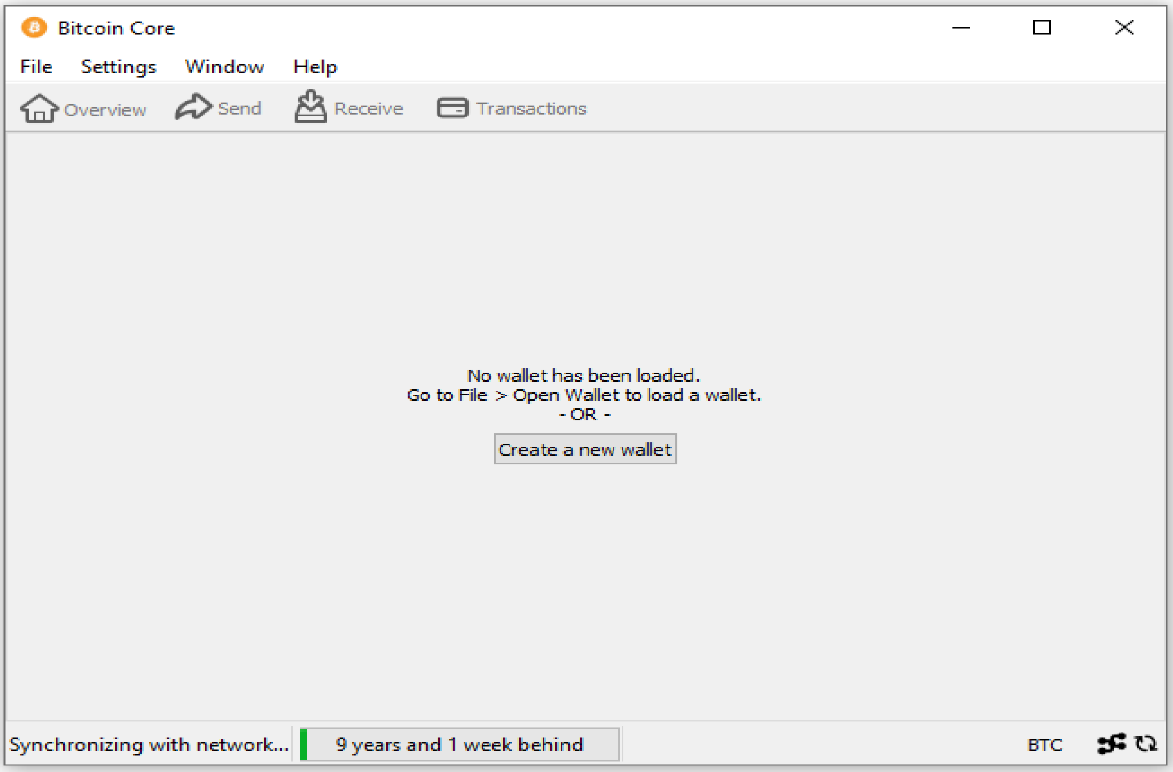


Recent transactions may not yet be visible, and therefore your wallet's balance might be incorrect. This information will be correct once your wallet has finished synchronizing with the bitcoin network, as detailed below.
Attempting to spend bitcoins that are affected by not-yet-displayed transactions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

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

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

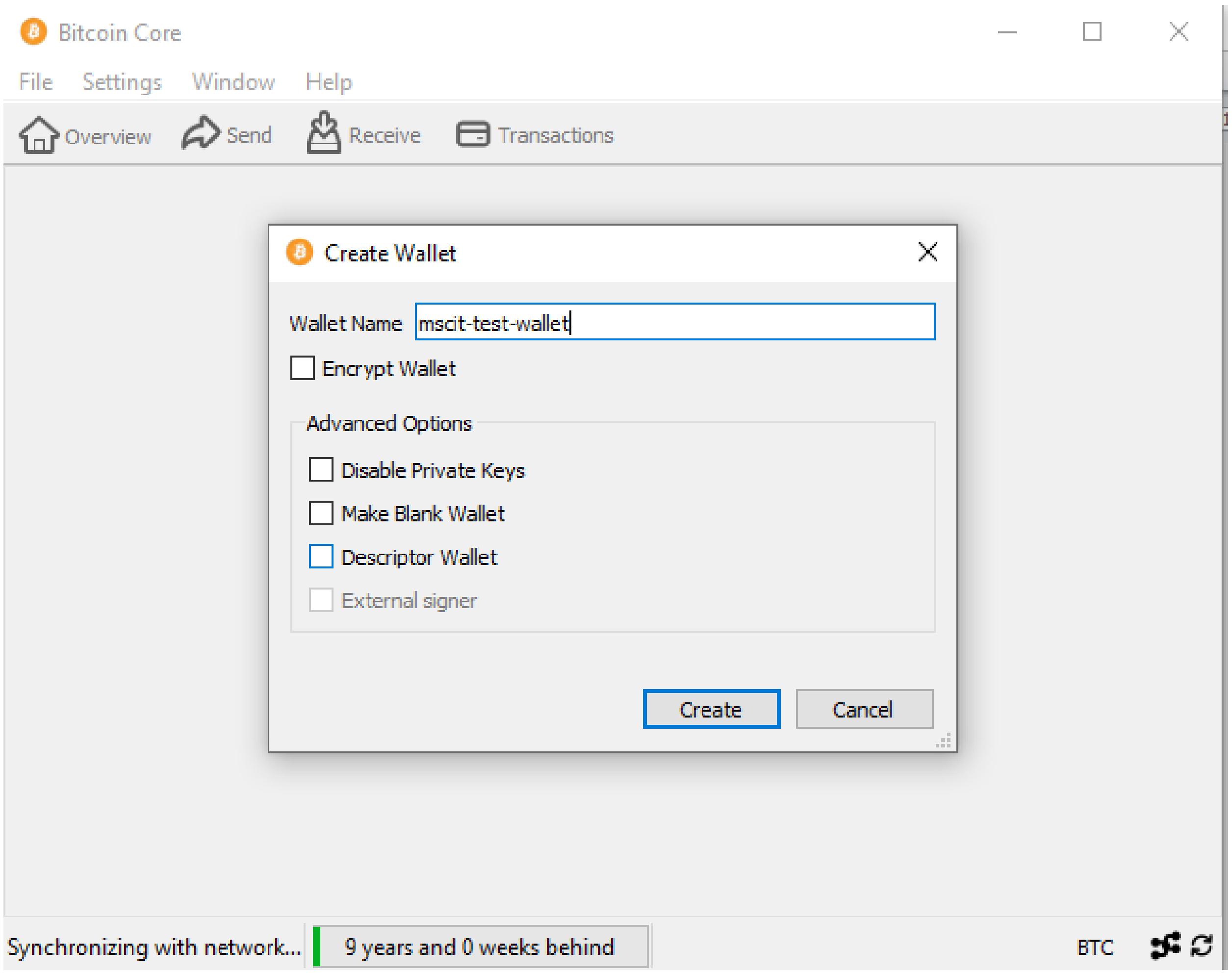
You can create a wallet -> Create a new wallet



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

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

Enter Wallet name



Wallet Name: mscit-test-wallet

☐ Encrypt Wallet

Advanced Options

☐ Disable Private Keys

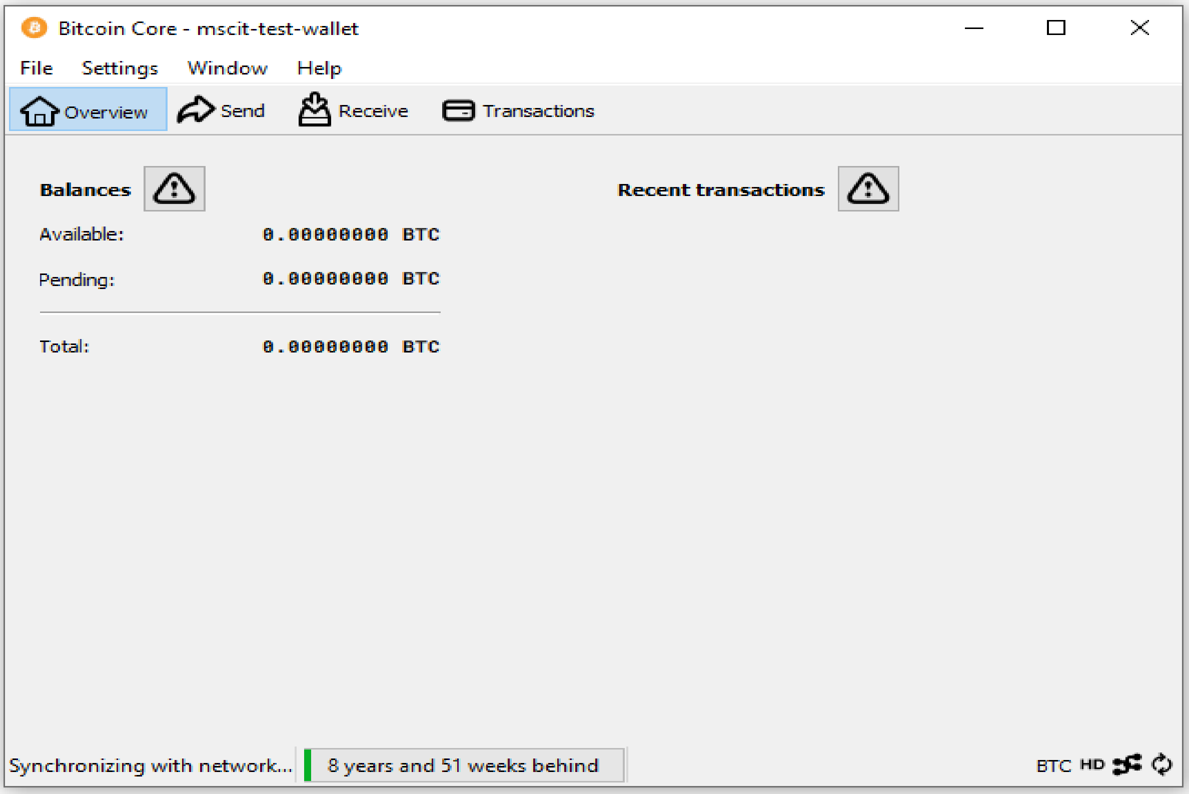
☐ Make Blank Wallet

☐ Descriptor Wallet

☐ External signer

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

Finally Account is setup


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

Balances

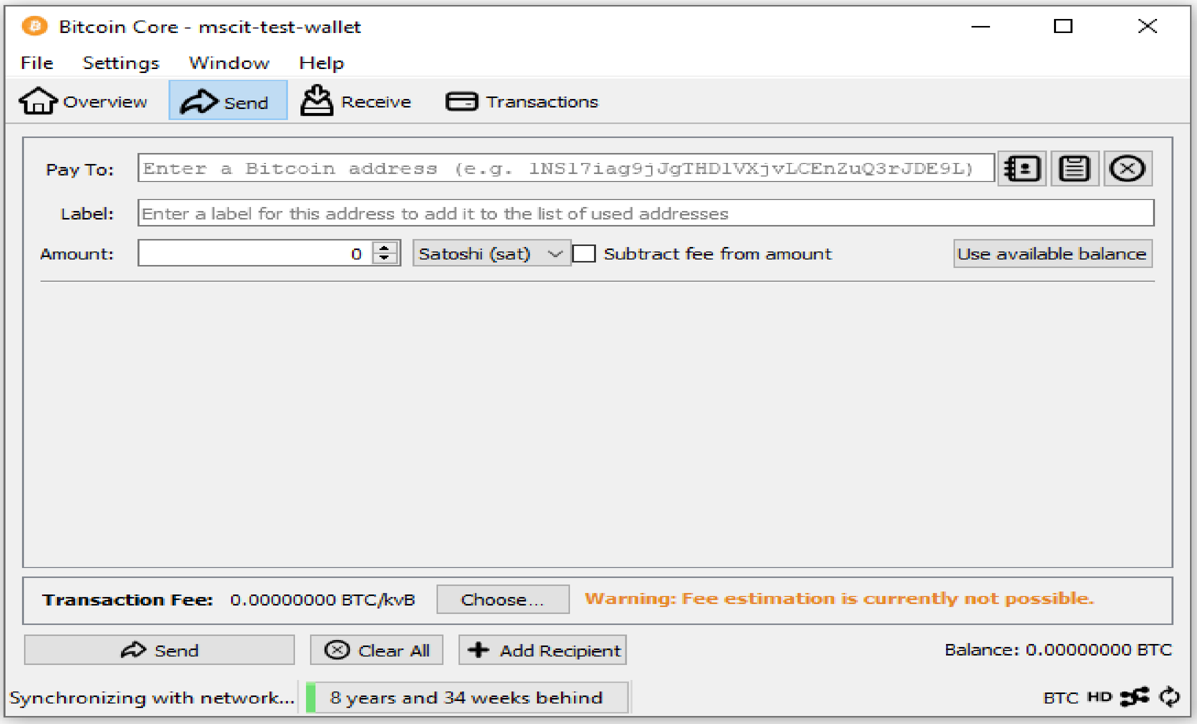
Available: 0.00000000 BTC

Pending: 0.00000000 BTC

Total: 0.00000000 BTC

Recent transactions

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


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

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

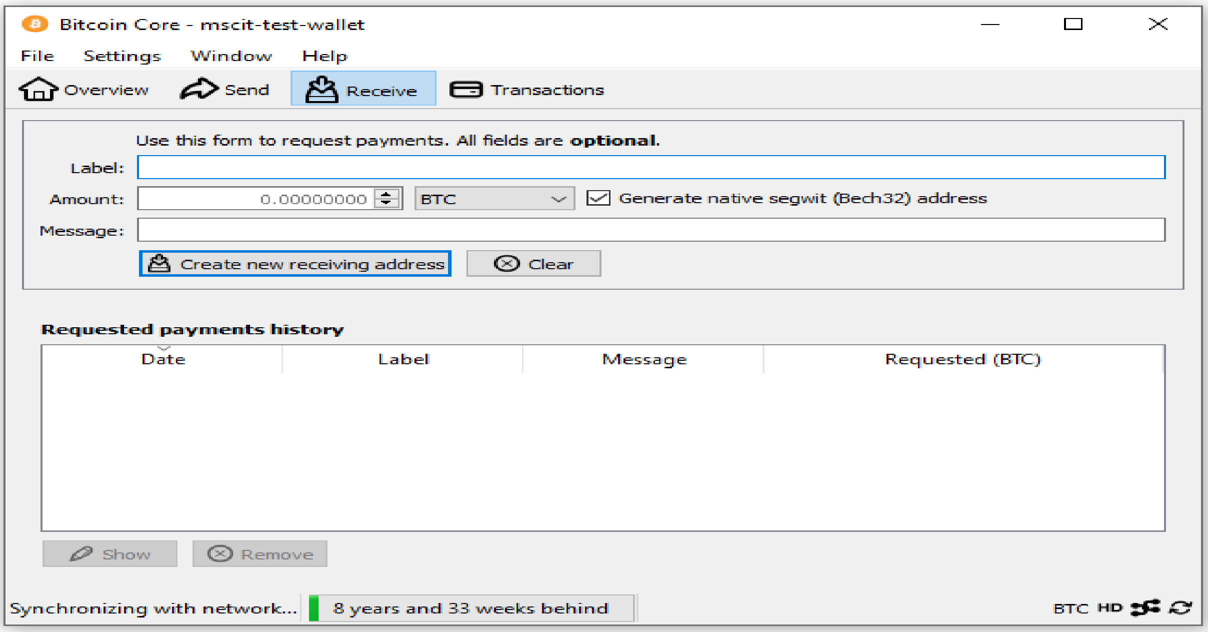
Label: Enter a label for this address to add it to the list of used addresses

Amount: 0 Satoshi (sat) Subtract fee from amount Use available balance

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

Send Clear All Add Recipient Balance: 0.00000000 BTC

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



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

Label:

Amount: 0.00000000 BTC

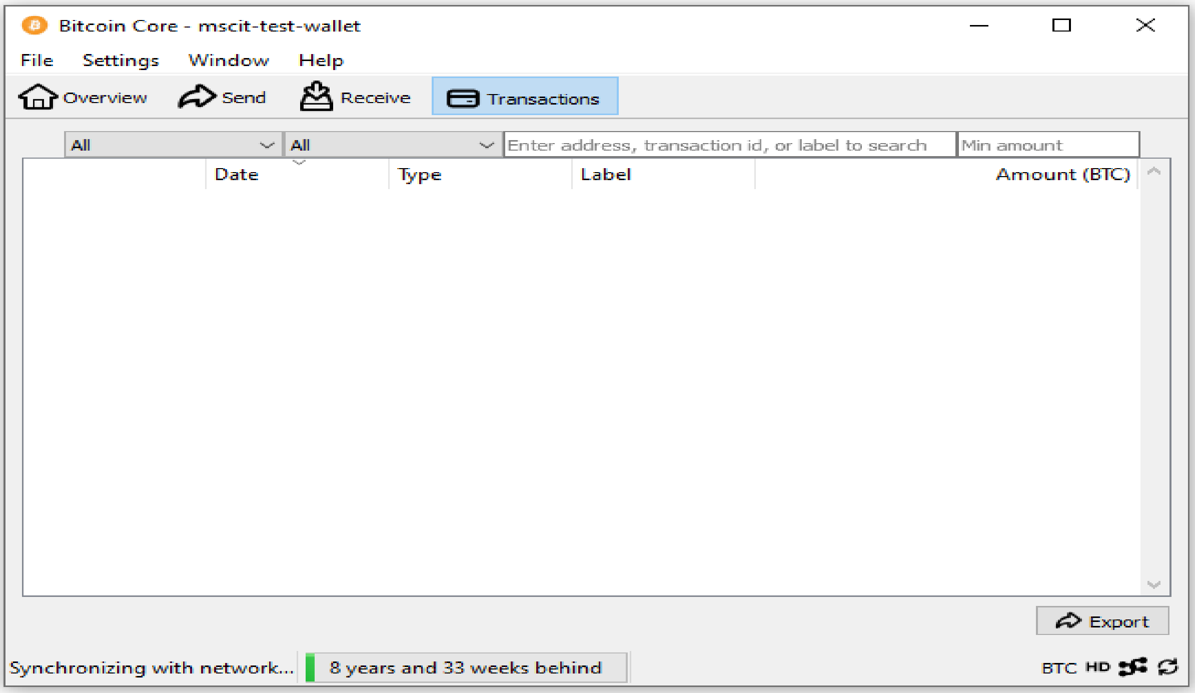
[✓] Generate native segwit (Bech32) address

Message:

Requested payments history

Date	Label	Message	Requested (BTC)

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


Bitcoin Core - mscit-test-wallet
File
Settings
Window
Help
Overview
Send
Receive
Transactions
All
All
Enter address, transaction id, or label to search
Min amount
Export
BTC
HD

Date	Type	Label	Amount (BTC)

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

Practical 8

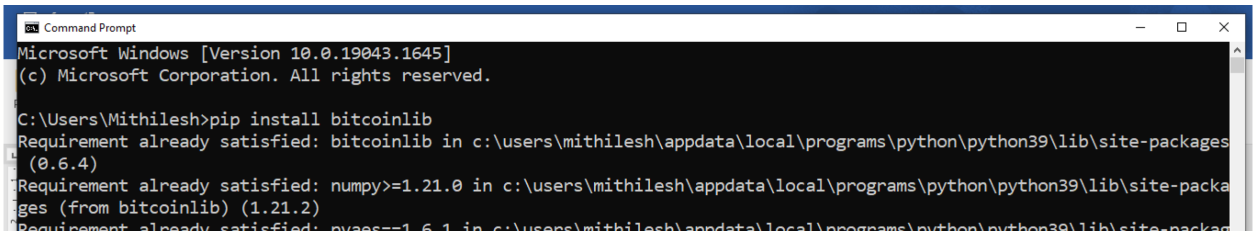
Aim: Demonstrate the use of Bitcoin Core API.

Open Python IDLE and create new Script.

########################

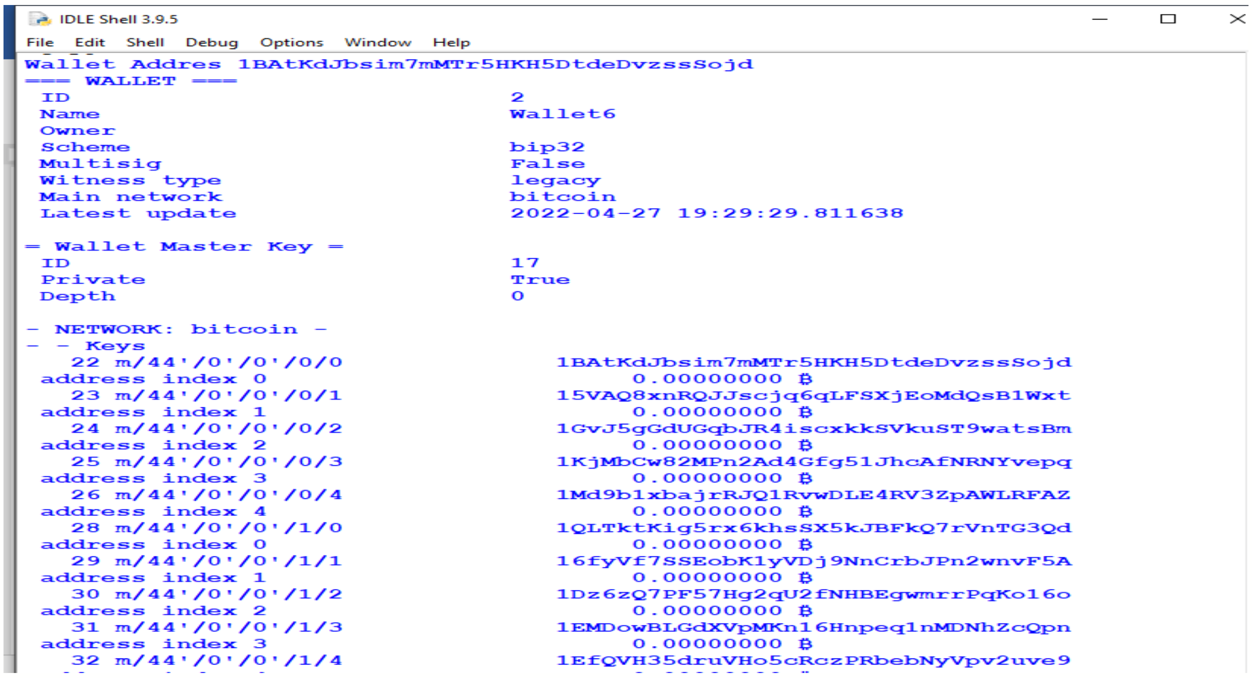
```
from bitcoinlib.wallets import Wallet
w = Wallet.create('Wallet6')
key1 = w.get_key()
print('Wallet Address:',key1.address)
w.scan()
print(w.info())
```

####################
Open CMD and install bitcoinlib package
pip install bitcoinlib



```
Microsoft Windows [Version 10.0.19043.1645]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Mithilesh>pip install bitcoinlib
Requirement already satisfied: bitcoinlib in c:\users\mithilesh\appdata\local\programs\python\python39\lib\site-packages
 (0.6.4)
Requirement already satisfied: numpy>=1.21.0 in c:\users\mithilesh\appdata\local\programs\python\python39\lib\site-packa
ges (from bitcoinlib) (1.21.2)
```

After installing package run the program from Python IDLE



```
Wallet Addres 1BAtKdJbsim7mMTr5HKH5DtdeDvzssSojd
=== WALLET ===
 ID                              2
 Name                            Wallet6
 Owner
 Scheme                          bip32
 Multisig                        False
 Witness type                    legacy
 Main network                    bitcoin
 Latest update                   2022-04-27 19:29:29.811638

= Wallet Master Key =
 ID                              17
 Private                         True
 Depth                           0

- NETWORK: bitcoin -
- - Keys
   22 m/44'/0'/0'/0/0               1BAtKdJbsim7mMTr5HKH5DtdeDvzssSojd
 address index 0                         0.00000000 ₿
   23 m/44'/0'/0'/0/1               15VAQ8xnRQJJscjq6qLFSXjEoMdQsB1Wxt
 address index 1                         0.00000000 ₿
   24 m/44'/0'/0'/0/2               1GvJ5gGdUGqbJR4iscxkkSVkuST9watsBm
 address index 2                         0.00000000 ₿
   25 m/44'/0'/0'/0/3               1KjMbCw82MPn2Ad4Gfg51JhcAfNRNYvepq
 address index 3                         0.00000000 ₿
   26 m/44'/0'/0'/0/4               1Md9b1xbajrRJQ1RvwDLE4RV3ZpAWLRFAZ
 address index 4                         0.00000000 ₿
   28 m/44'/0'/0'/1/0               1QLTktKig5rx6khsSX5kJBFkQ7rVnTG3Qd
 address index 0                         0.00000000 ₿
   29 m/44'/0'/0'/1/1               16fyVf7SSEobK1yVDj9NnCrbJPn2wnvF5A
 address index 1                         0.00000000 ₿
   30 m/44'/0'/0'/1/2               1Dz6zQ7PF57Hg2qU2fNHBEgwmrrPqKo16o
 address index 2                         0.00000000 ₿
   31 m/44'/0'/0'/1/3               1EMDowBLGdXVpMKn16Hnpeq1nMDNhZcQpn
 address index 3                         0.00000000 ₿
   32 m/44'/0'/0'/1/4               1EfQVH35druVHo5cRczPRbebNyVpv2uve9
```

More Detail: https://pypi.org/project/bitcoinlib/

Practical 9

Aim: Build Dapps with angular[using truffle and ganache cli]

Step 1:Install the required package –on new **terminal 1** type these commands

sudo apt-get -y install curl git vim build-essential

sudo apt-get install curl software-properties-common

sudo apt install npm

sudo npm install -g web3

sudo apt-get install nodejs

sudo apt install python3.9

curl -sL https://deb.nodesource.com/setup_10.x | sudo bash -

sudo npm install --global node-sass@latest

sudo npm install -g truffle@latest

sudo npm install -g ganache-cli

export NODE_OPTIONS=--openssl-legacy-provider

Step 2: Create a new directory

mkdir myproject

cd myproject

Step 3:Initialize the project folder

truffle init

########################

////to update npm//

sudo npm cache clean -f

sudo npm install -g n

sudo n latest

######################

Step 4: Now create a new contract

nano contracts/HelloWorld.sol

Step 5:Add the following code in HelloWorld.sol

```
pragma solidity ^0.8.0;

contract HelloWorld {

   function sayHello() public pure returns(string memory){

      return("hello world");

   }

}
```

Step 6:Edit default configuration file

nano migrations/1_initial_migration.js

Step 7: Edit this line in the file

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

module.exports = function (deployer) {

  deployer.deploy(Migrations,"hello");

};
```

Step 8: Edit network configuration file

sudo nano truffle-config.js

Remove all line(press CTRL +K) from the file and add the following lines

#########################

```
module.exports = {

  networks: {
   development: {
     host: "127.0.0.1",
     port: 8545,
     network_id: "*",
   }
 }
}
```

#########################

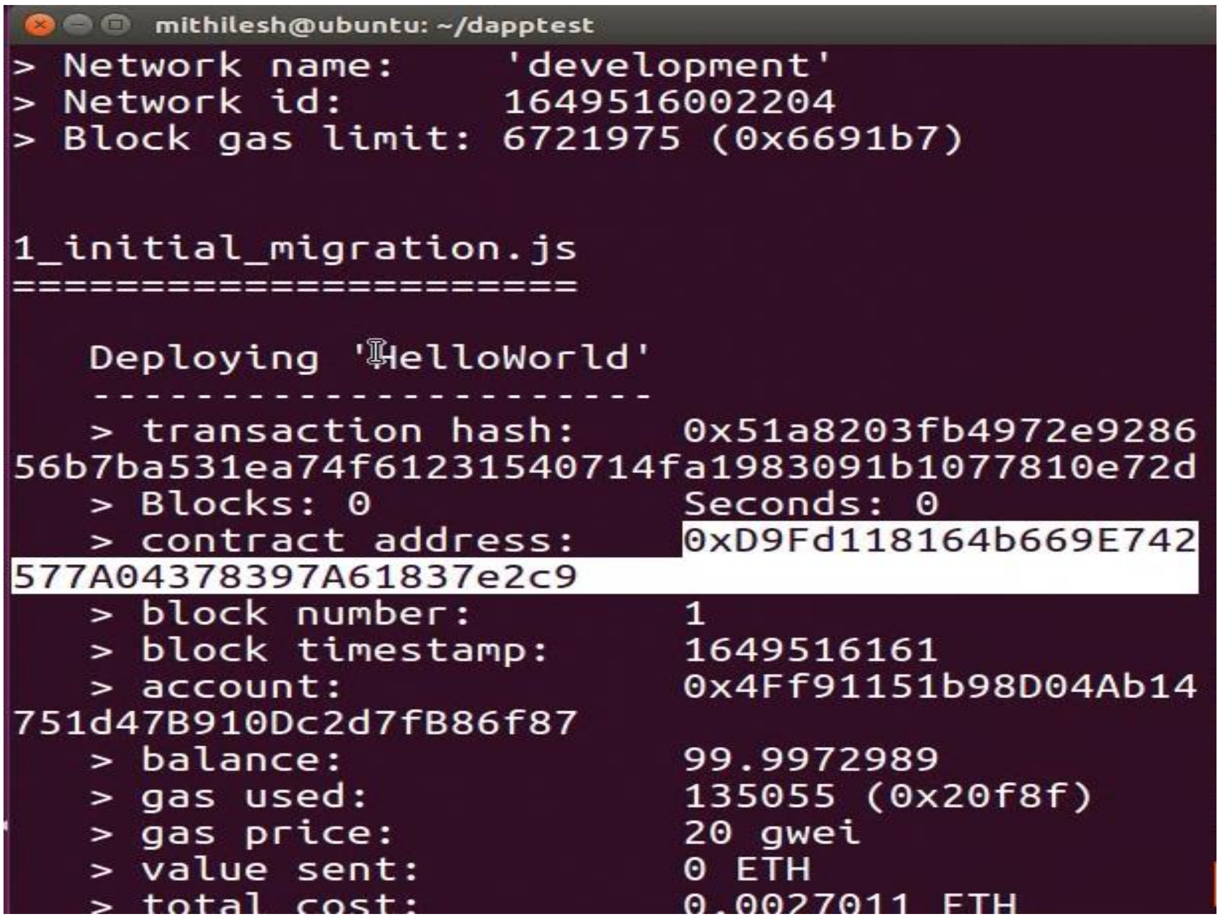
Step 9: start ganache-cli –Switch/Open to **terminal 2**

ganache-cli

Step 10: deploy the truffle deploy- On **terminal 1**

truffle deploy

[Note contract address]



```
> Network name:    'development'
> Network id:      1649516002204
> Block gas limit: 6721975 (0x6691b7)


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

   Deploying 'HelloWorld'
   ----------------------
   > transaction hash:    0x51a8203fb4972e9286
56b7ba531ea74f61231540714fa1983091b1077810e72d
   > Blocks: 0            Seconds: 0
   > contract address:    0xD9Fd118164b669E742
577A04378397A61837e2c9
   > block number:        1
   > block timestamp:     1649516161
   > account:             0x4Ff91151b98D04Ab14
751d47B910Dc2d7fB86f87
   > balance:             99.9972989
   > gas used:            135055 (0x20f8f)
   > gas price:           20 gwei
   > value sent:          0 ETH
   > total cost:          0.0027011 ETH
```

Step 11: Open truffle console - On **terminal 1**

truffle console

Step 11: Get reference of contact

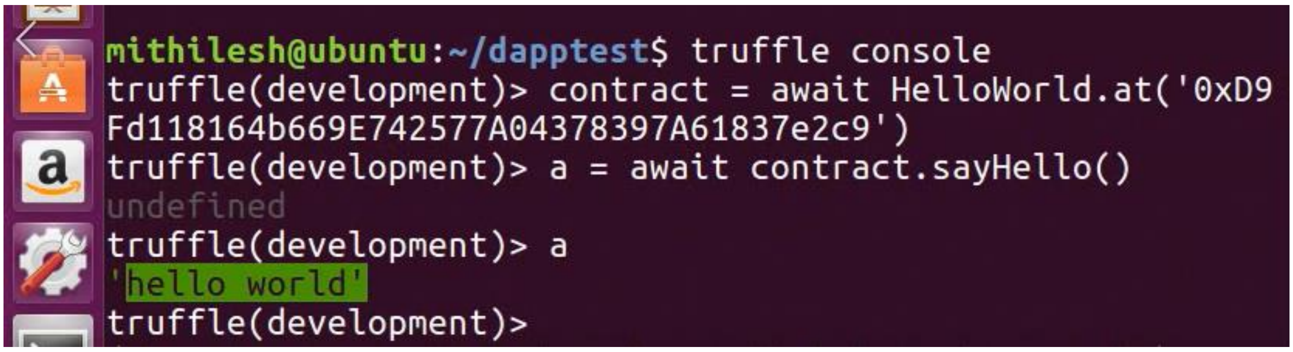
contract = await HelloWorld.at('0x2C403EE1b30F56C0c773089c1Eb9DddF1499C969')

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

Step 12: Call the function from the contract

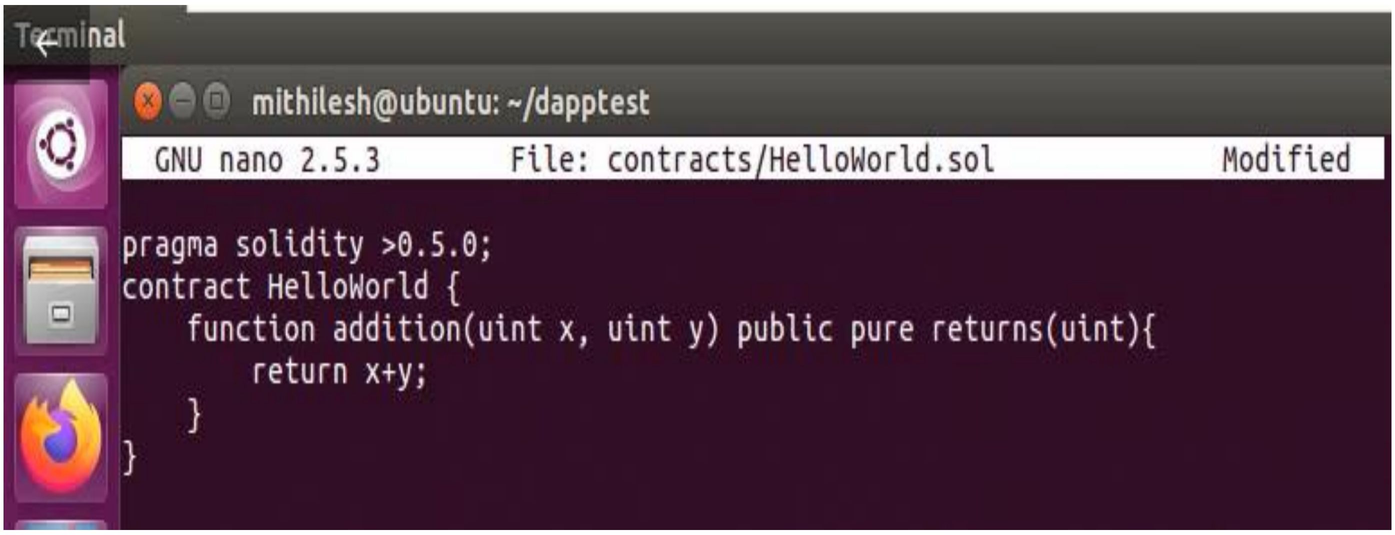
a = await contract.sayHello()

Step 13:Print output on the screen

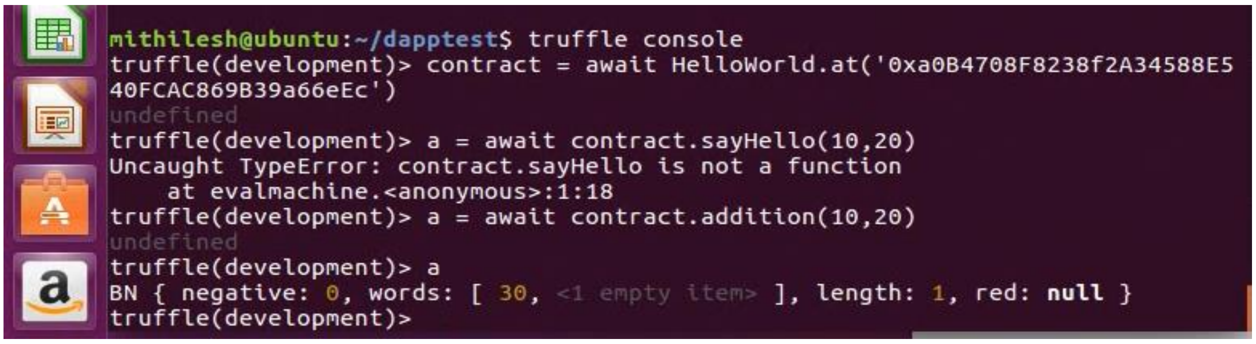

```
mithilesh@ubuntu:~/dapptest$ truffle console
truffle(development)> contract = await HelloWorld.at('0xD9
Fd118164b669E742577A04378397A61837e2c9')
truffle(development)> a = await contract.sayHello()
undefined
truffle(development)> a
'hello world'
truffle(development)>
```

[In case you are getting any error for version; change the solidity version in HelloWorld.sol]

Example 2:


```
mithilesh@ubuntu: ~/dapptest
  GNU nano 2.5.3        File: contracts/HelloWorld.sol                Modified

pragma solidity >0.5.0;
contract HelloWorld {
    function addition(uint x, uint y) public pure returns(uint){
        return x+y;
    }
}
```

Save-> Deploy ->Open console


```
mithilesh@ubuntu:~/dapptest$ truffle console
truffle(development)> contract = await HelloWorld.at('0xa0B4708F8238f2A34588E5
40FCAC869B39a66eEc')
undefined
truffle(development)> a = await contract.sayHello(10,20)
Uncaught TypeError: contract.sayHello is not a function
    at evalmachine.<anonymous>:1:18
truffle(development)> a = await contract.addition(10,20)
undefined
truffle(development)> a
BN { negative: 0, words: [ 30, <1 empty item> ], length: 1, red: null }
truffle(development)>
```

Natural Language Processing

Index

Sr. No	Title	Date	Sign
1.	A. Install NLTK. B. Convert the given text to speech. C. Convert audio file Speech to Text. 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.	A. Study of various Corpus – Brown, Inaugural, Reuters, udhr with various methods like fields, raw, words, sents, categories. B. Create and use your own corpora (plaintext, categorical). C. Study Conditional frequency distributions Study of tagged corpora with methods like tagged_sents, tagged_words. D. Write a program to find the most frequent noun tags. E. Map Words to Properties Using Python Dictionaries F. Study DefaultTagger, Regular expression tagger, UnigramTagger G. Find different words from a given plain text without any space by comparing this text with a given corpus of words. Also find the score of words.		
3.	A. Study of Wordnet Dictionary with methods as synsets, definitions, examples, antonyms. B. Study lemmas, hyponyms, hypernyms, entailments.		

	C. Write a program using python to find synonym and antonym of word "active" using Wordnet. D. Compare two nouns. E. Handling stopword. Using nltk Adding or Removing Stop Words in NLTK's Default Stop Word List. Using Gensim Adding and Removing Stop Words in Default Gensim Stop Words List. Using Spacy Adding and Removing Stop Words in Default Spacy Stop Words List.		
4.	Text Tokenization A. Tokenization using Python's split() function. B. Tokenization using Regular Expressions (RegEx). C. Tokenization using NLTK. D. Tokenization using the spaCy library. E. Tokenization using Keras. F. Tokenization using Gensim.		
5.	Important NLP Libraries for Indian Languages and perform: A. word tokenization in Hindi. B. Generate similar sentences from a given Hindi text input. C. Identify the Indian language of a text.		
6.	Illustrate part of speech tagging. A. Part of speech Tagging and chunking of user defined text. B. Named Entity recognition of user defined text. C. Named Entity recognition with diagram using NLTK corpus – treebank.		
7.	A. Define grammer using nltk. Analyze a sentence using the same.		

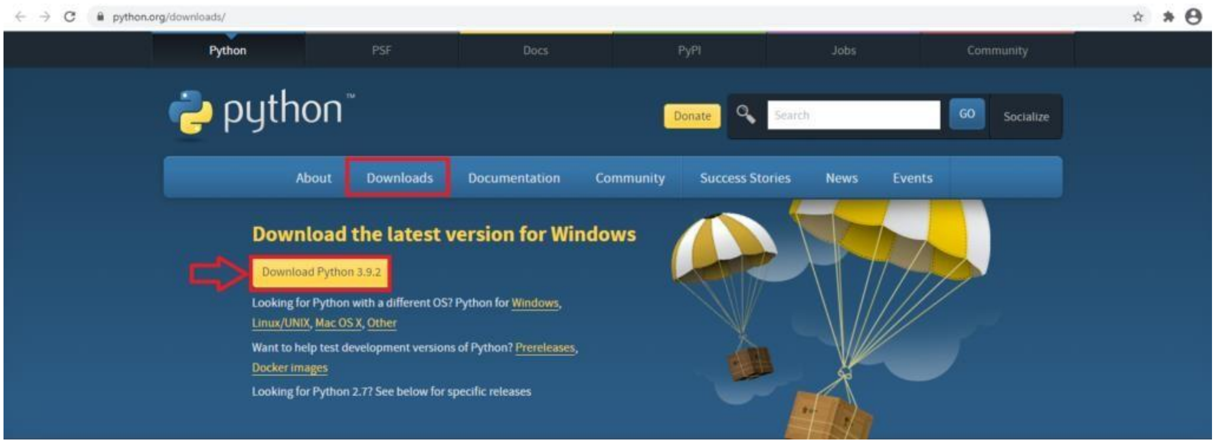
	B. Accept the input string with Regular expression of FA: 101+ C. Accept the input string with Regular expression of FA: (a+b)*bba D. Implementation of Deductive Chart Parsing using context free grammar and a given sentence.		
8.	Study PorterStemmer, LancasterStemmer, RegexpStemmer, SnowballStemmer Study WordNetLemmatizer.		
9.	Implement Naive Bayes classifier.		
10.	Speech Tagging: A. Speech tagging using spacy B. Speech tagging using nktl Statistical parsing: A. Usage of Give and Gave in the Penn Treebank sample B. probabilistic parser Malt parsing: Parse a sentence and draw a tree using malt parsing.		
11.	A. Multiword Expressions in NLP B. Normalized Web Distance and Word Similarity C. Word Sense Disambiguation		

Practical No. 1

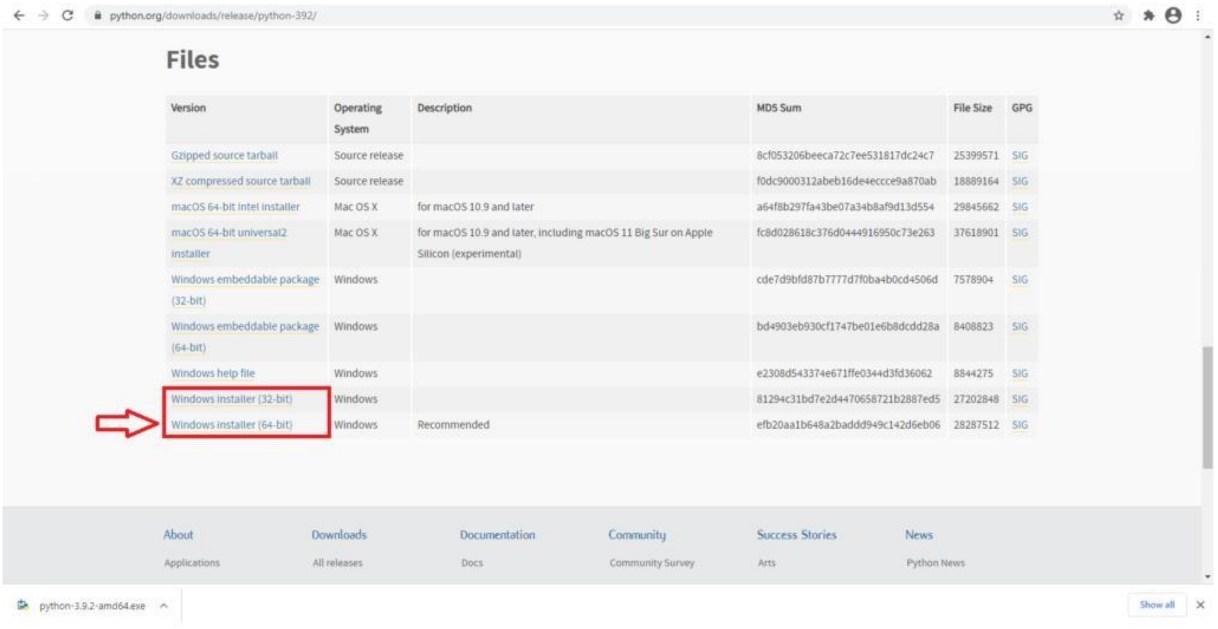
A. Install NLTK

Python 3.9.2 Installation on Windows

Step 1) **Go to link** https://www.python.org/downloads/, **and select the latestversion for windows.**


python.org/downloads/
Python
PSF
Docs
PyPI
Jobs
Community
python
Donate
Search
GO
Socialize
About
Downloads
Documentation
Community
Success Stories
News
Events
Download the latest version for Windows
Download Python 3.9.2
Looking for Python with a different OS? Python for Windows, Linux/UNIX, Mac OS X, Other
Want to help test development versions of Python? Prereleases, Docker images
Looking for Python 2.7? See below for specific releases

Note: If you don't want to download the latest version, you can visit thedownload tab and see all releases.

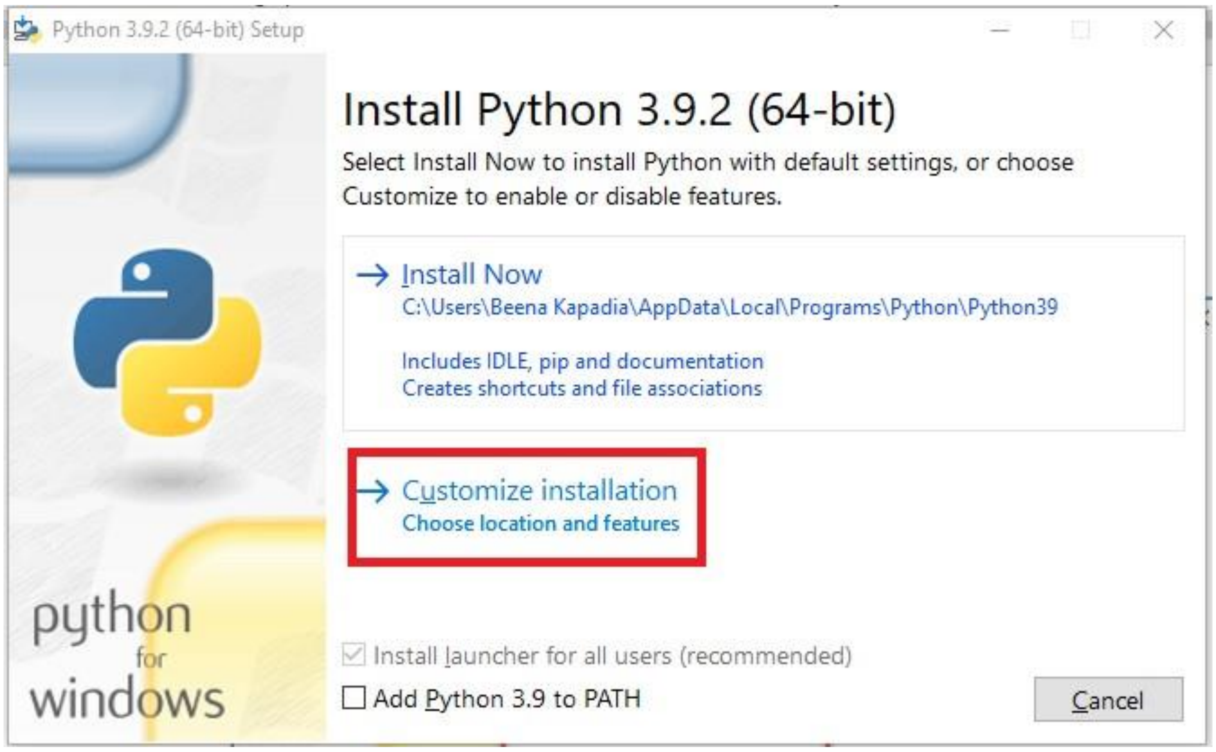


Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		8cf053206beeca72c7ee531817dc24c7	25399571	SIG
XZ compressed source tarball	Source release		f0dc9000312abeb16de4eccce9a870ab	18889164	SIG
macOS 64-bit Intel installer	Mac OS X	for macOS 10.9 and later	a64f8b297fa43be07a34b8af9d13d554	29845662	SIG
macOS 64-bit universal2 installer	Mac OS X	for macOS 10.9 and later, including macOS 11 Big Sur on Apple Silicon (experimental)	fc8d028618c376d0444916950c73e263	37618901	SIG
Windows embeddable package (32-bit)	Windows		cde7d9bfd87b7777d7f0ba4b0cd4506d	7578904	SIG
Windows embeddable package (64-bit)	Windows		bd4903eb930cf1747be01e6b8dcdd28a	8408823	SIG
Windows help file	Windows		e2308d543374e671ffe0344d3fd36062	8844275	SIG
Windows installer (32-bit)	Windows		81294c31bd7e2d4470658721b2887ed5	27202848	SIG
Windows installer (64-bit)	Windows	Recommended	efb20aa1b648a2baddd949c142d6eb06	28287512	SIG

Step 2) Click on the Windows installer (64 bit)

Step 3) Select Customize Installation


Python 3.9.2 (64-bit) Setup

Install Python 3.9.2 (64-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

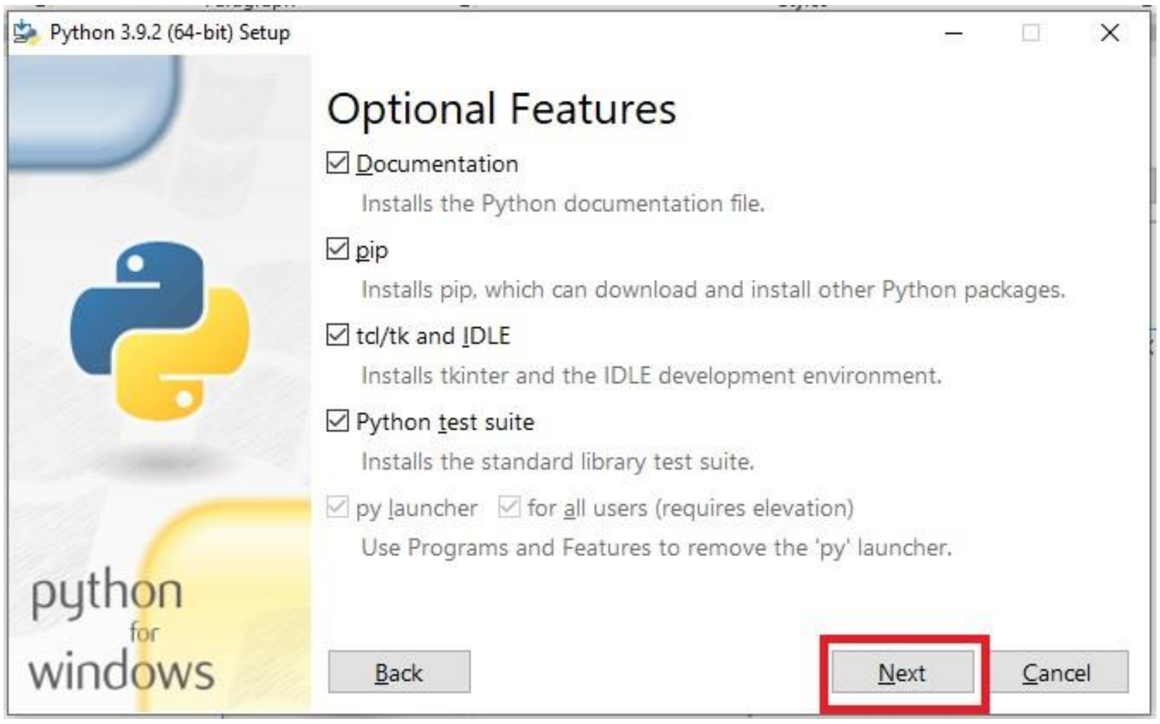
→ Install Now
C:\Users\Beena Kapadia\AppData\Local\Programs\Python\Python39

Includes IDLE, pip and documentation
Creates shortcuts and file associations

→ Customize installation
Choose location and features

☑ Install launcher for all users (recommended)
☐ Add Python 3.9 to PATH

Step 4) Click NEXT


Python 3.9.2 (64-bit) Setup

Optional Features

☑ Documentation
Installs the Python documentation file.

☑ pip
Installs pip, which can download and install other Python packages.

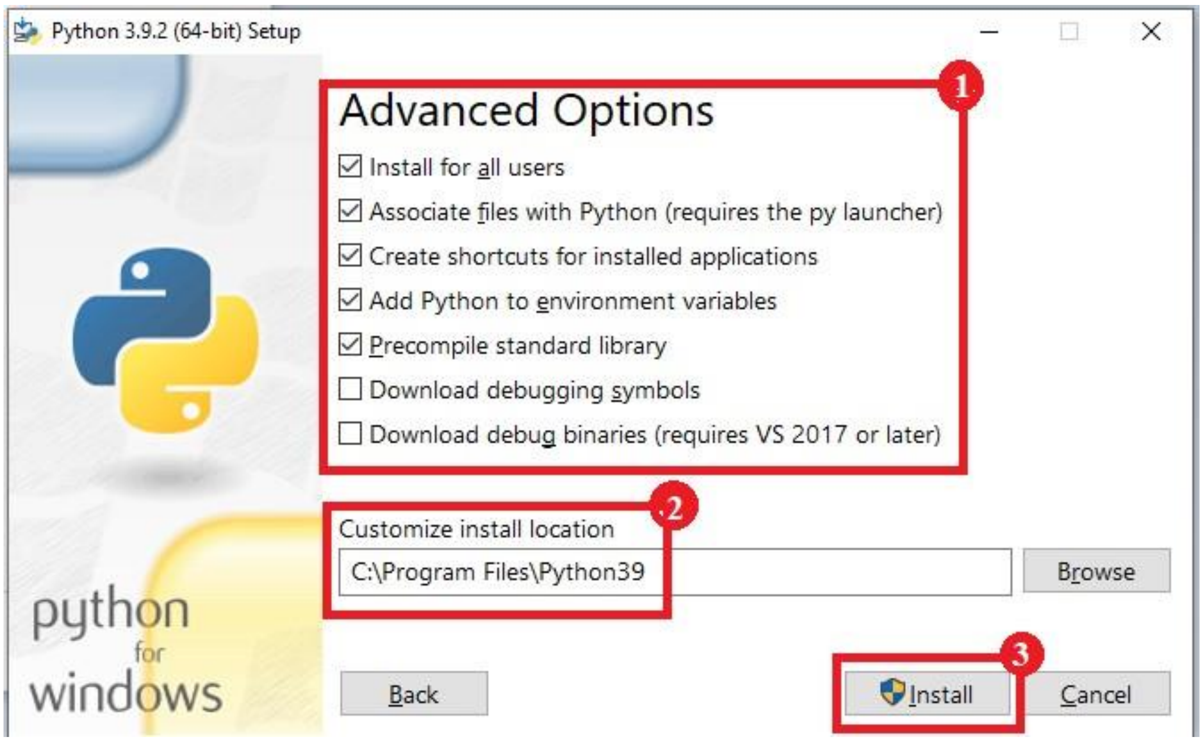
☑ tcl/tk and IDLE
Installs tkinter and the IDLE development environment.

☑ Python test suite
Installs the standard library test suite.

☑ py launcher ☑ for all users (requires elevation)
Use Programs and Features to remove the 'py' launcher.

Step 5) In next screen

1. Select the advanced options
2. Give a Custom install location. Keep the default folder as c:\Program files\Python39
3. Click Install



Advanced Options

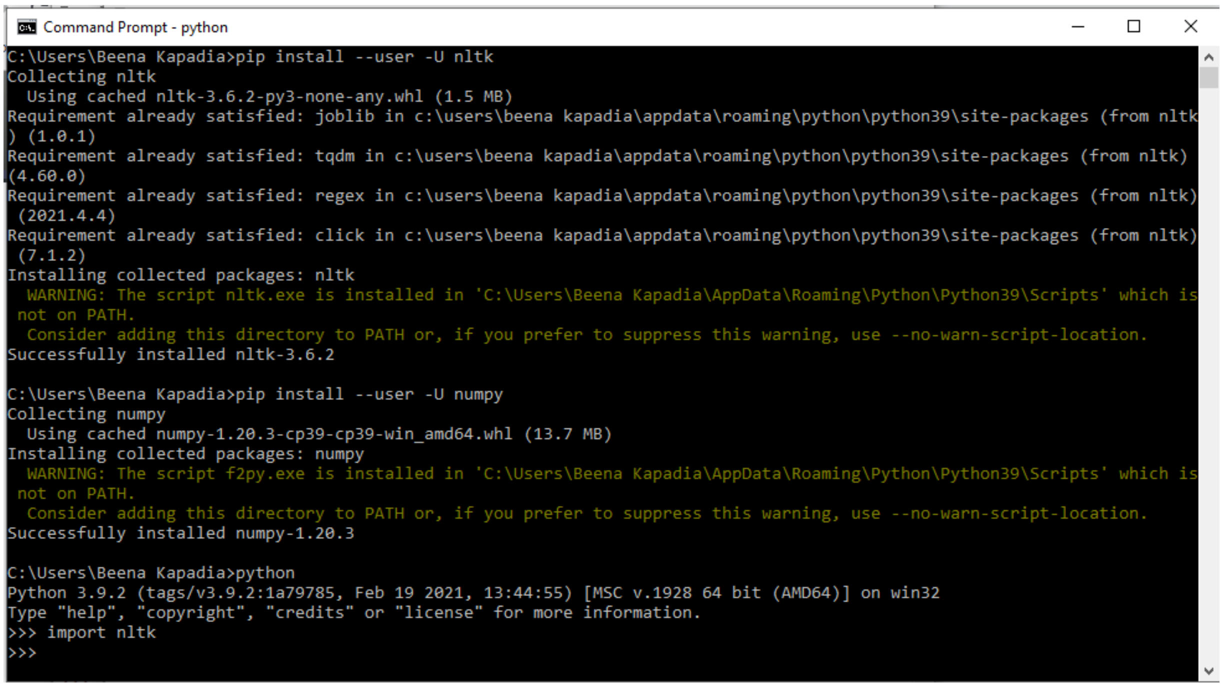
- ☑ Install for all users
- ☑ Associate files with Python (requires the py launcher)
- ☑ Create shortcuts for installed applications
- ☑ Add Python to environment variables
- ☑ Precompile standard library
- ☐ Download debugging symbols
- ☐ Download debug binaries (requires VS 2017 or later)

Customize install location

C:\Program Files\Python39

Step 6) Click Close button once install is done.

Step 7) open command prompt window and run the following commands:
C:\Users\Beena Kapadia>pip install --upgrade pip
C:\Users\Beena Kapadia> pip install --user -U nltk C:\Users\Beena
Kapadia> >pip install --user -U numpyC:\Users\Beena
Kapadia>python
>>> import nltk
>>>



```
C:\Users\Beena Kapadia>pip install --user -U nltk
Collecting nltk
  Using cached nltk-3.6.2-py3-none-any.whl (1.5 MB)
Requirement already satisfied: joblib in c:\users\beena kapadia\appdata\roaming\python\python39\site-packages (from nltk
) (1.0.1)
Requirement already satisfied: tqdm in c:\users\beena kapadia\appdata\roaming\python\python39\site-packages (from nltk)
(4.60.0)
Requirement already satisfied: regex in c:\users\beena kapadia\appdata\roaming\python\python39\site-packages (from nltk)
 (2021.4.4)
Requirement already satisfied: click in c:\users\beena kapadia\appdata\roaming\python\python39\site-packages (from nltk)
 (7.1.2)
Installing collected packages: nltk
  WARNING: The script nltk.exe is installed in 'C:\Users\Beena Kapadia\AppData\Roaming\Python\Python39\Scripts' which is
 not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Successfully installed nltk-3.6.2

C:\Users\Beena Kapadia>pip install --user -U numpy
Collecting numpy
  Using cached numpy-1.20.3-cp39-cp39-win_amd64.whl (13.7 MB)
Installing collected packages: numpy
  WARNING: The script f2py.exe is installed in 'C:\Users\Beena Kapadia\AppData\Roaming\Python\Python39\Scripts' which is
 not on PATH.
  Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Successfully installed numpy-1.20.3

C:\Users\Beena Kapadia>python
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import nltk
>>>
```

(Browse https://www.nltk.org/install.html for more details)

b) Convert the given text to speech.
Source code:

```
# text to speech# pip

install gtts
# pip install playsound

from playsound import playsound

# import required for text to speech conversionfrom

gtts import gTTS
mytext = "Welcome to Natural Language programming"
language = "en"
myobj = gTTS(text=mytext, lang=language, slow=False)
myobj.save("myfile.mp3")
playsound("myfile.mp3")
```

Output:
welcomeNLP.mp3 audio file is getting created and it plays the file with playsound()method, while running the program.

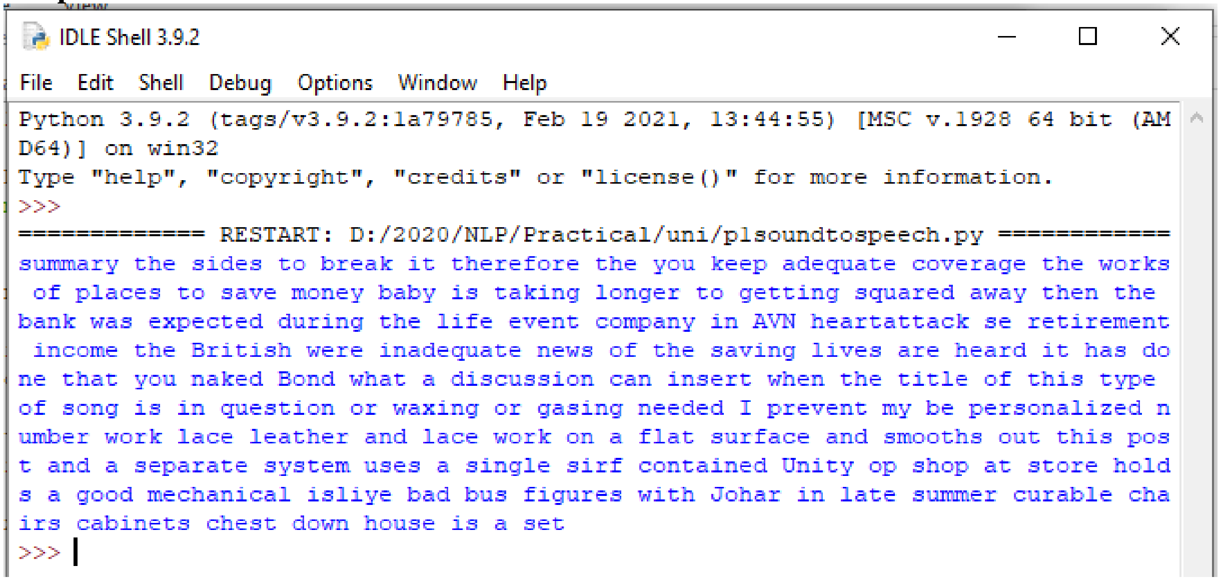
c) Convert audio file Speech to Text.
Source code:

Note: required to store the input file "male.wav" in the current folder before running the program.

```
#pip3 install SpeechRecognition pydub

import speech_recognition as sr
filename = "male.wav"

# initialize the recognizerr =
sr.Recognizer()

# open the file
with sr.AudioFile(filename) as source:
# listen for the data (load audio to memory)audio_data
= r.record(source)
# recognize (convert from speech to text)text =
r.recognize_google(audio_data) print(text)
```

Input:
male.wav (any wav file)

Output:



```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 
============= RESTART: D:/2020/NLP/Practical/uni/p1soundtospeech.py ============
summary the sides to break it therefore the you keep adequate coverage the works
 of places to save money baby is taking longer to getting squared away then the
bank was expected during the life event company in AVN heartattack se retirement
 income the British were inadequate news of the saving lives are heard it has do
ne that you naked Bond what a discussion can insert when the title of this type
of song is in question or waxing or gasing needed I prevent my be personalized n
umber work lace leather and lace work on a flat surface and smooths out this pos
t and a separate system uses a single sirf contained Unity op shop at store hold
s a good mechanical isliye bad bus figures with Johar in late summer curable cha
irs cabinets chest down house is a set
>>> 
```

Practical No. 2

a. **Study of various Corpus – Brown, Inaugural, Reuters, udhr with various methods like filelds, raw, words, sents, categories.**
b. **Create and use your own corpora (plaintext, categorical)**
c. **Study Conditional frequency distributions**
d. **Study of tagged corpora with methods like tagged_sents, tagged_words.**
e. **Write a program to find the most frequent noun tags.**
f. **Map Words to Properties Using Python Dictionaries**
g. **Study DefaultTagger, Regular expression tagger, UnigramTagger**
h. **Find different words from a given plain text without any space by comparing this text with a given corpus of words. Also find the score of words.**

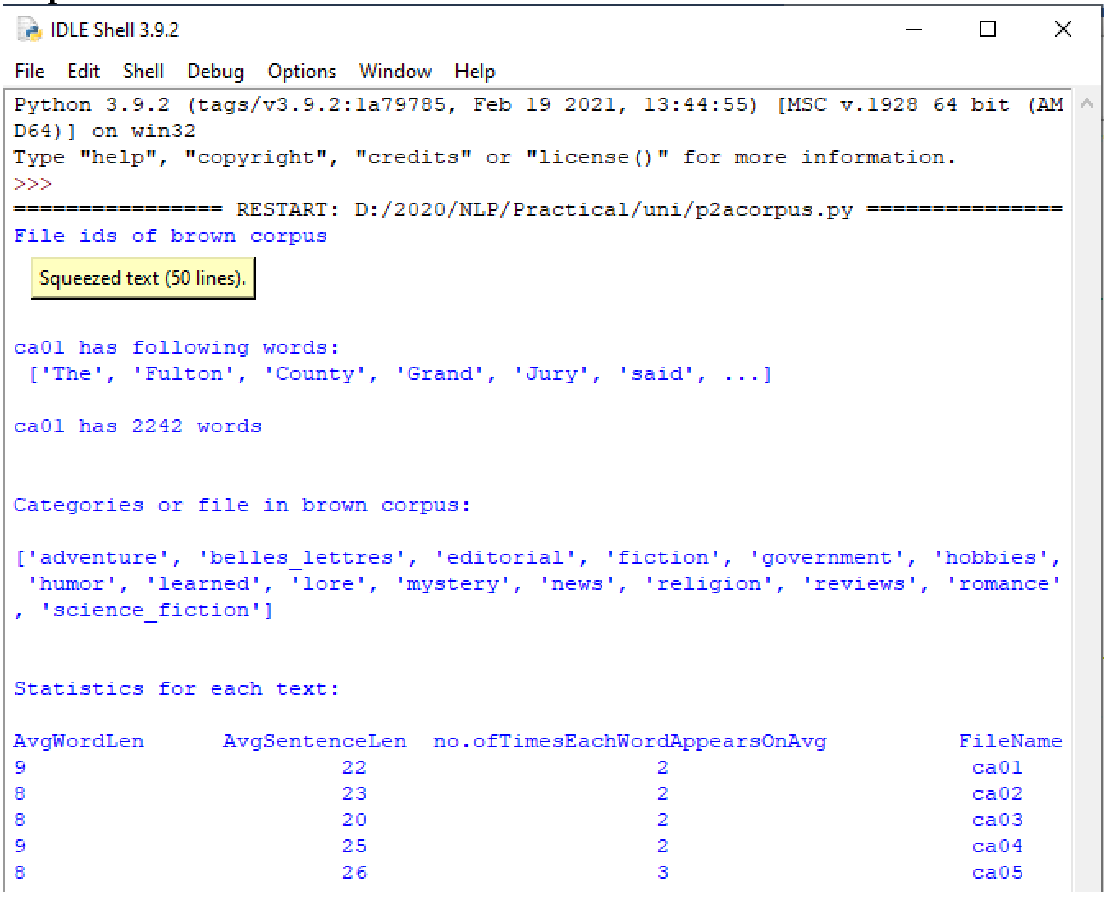
a. Study of various Corpus – Brown, Inaugural, Reuters, udhr with various methods like fields, raw, words, sents, categories,

source code:

```
'''NLTK includes a small selection of texts from the Project brown electronic text archive, which
contains some 25,000 free electronic books, hosted at http://www.brown.org/. We begin by
getting the Python interpreter to load the NLTK package, then ask to see
nltk.corpus.brown.fileids(), the file identifiers in this corpus:'''

import nltk
from nltk.corpus import brown
print ('File ids of brown corpus\n',brown.fileids())

'''Let's pick out the first of these texts — Emma by Jane Austen — and give it a shortname,
emma, then find out how many words it contains:'''
ca01 = brown.words('ca01')

# display first few words
print('\nca01 has following words:\n',ca01)

# total number of words in ca01 print('\nca01
has',len(ca01),'words')



#categories or files
print ('\n\nCategories or file in brown corpus:\n')print
(brown.categories())

'''display other information about each text, by looping over all the values of fileid
corresponding to the brown file identifiers listed earlier and then computing statisticsfor each
text.'''
print ('\n\nStatistics for each text:\n')print
('AvgWordLen\tAvgSentenceLen\tno.ofTimesEachWordAppearsOnAvg\t\tFileName') for
fileid in brown.fileids():
num_chars = len(brown.raw(fileid)) num_words
= len(brown.words(fileid))num_sents =
len(brown.sents(fileid))
num_vocab = len(set([w.lower() for w in brown.words(fileid)]))
```

print (int(num_chars/num_words),'\t\t\t', int(num_words/num_sents),'\t\t\t', int(num_words/num_vocab),'\t\t\t', fileid)

output:



IDLE Shell 3.9.2
File Edit Shell Debug Options Window Help
Squeezed text (50 lines).

```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================ RESTART: D:/2020/NLP/Practical/uni/p2acorpus.py ===============
File ids of brown corpus

ca01 has following words:
 ['The', 'Fulton', 'County', 'Grand', 'Jury', 'said', ...]

ca01 has 2242 words


Categories or file in brown corpus:

['adventure', 'belles_lettres', 'editorial', 'fiction', 'government', 'hobbies',
 'humor', 'learned', 'lore', 'mystery', 'news', 'religion', 'reviews', 'romance'
, 'science_fiction']


Statistics for each text:

AvgWordLen      AvgSentenceLen  no.ofTimesEachWordAppearsOnAvg          FileName
9                        22                       2                      ca01
8                        23                       2                      ca02
8                        20                       2                      ca03
9                        25                       2                      ca04
8                        26                       3                      ca05
```

b. Create and use your own corpora (plaintext, categorical)

source code:

'''NLTK includes a small selection of texts from the Project filelist electronic text archive, which contains some 25,000 free electronic books, hosted at http://www.filelist.org/. We begin by getting the Python interpreter to load the NLTK package, then ask to see nltk.corpus.filelist.fileids(), the file identifiers in this corpus:'''

import nltk
from nltk.corpus import PlaintextCorpusReader

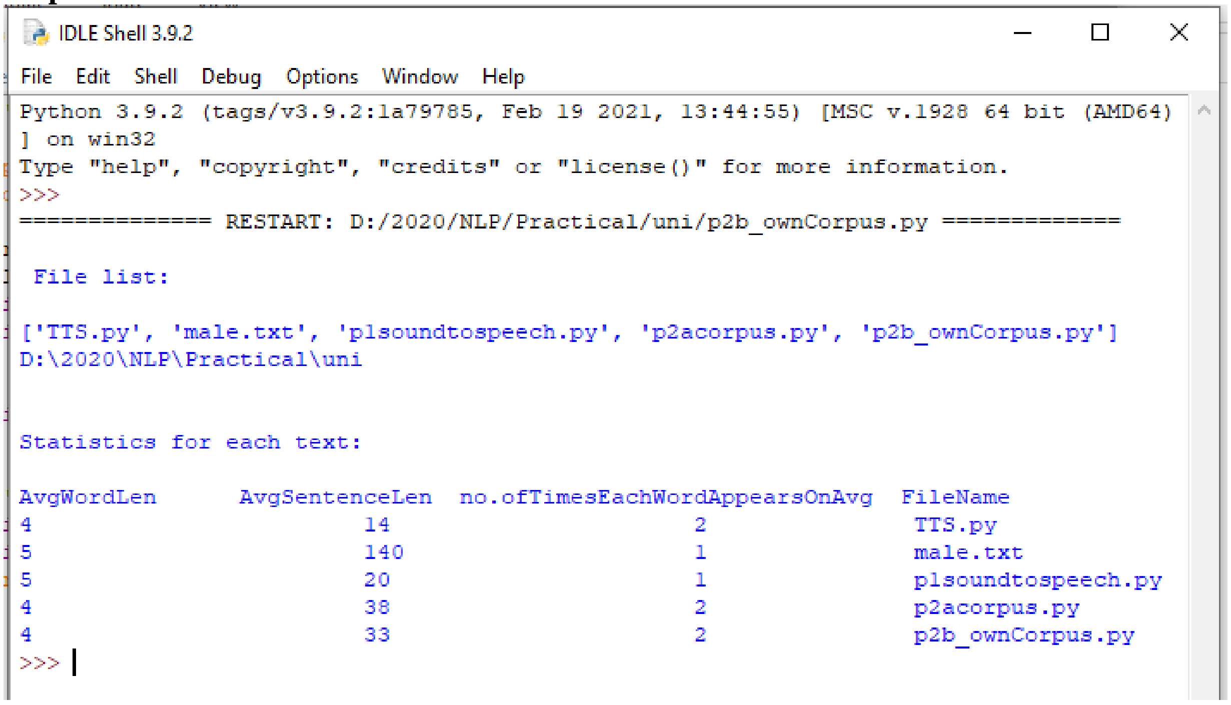
corpus_root = 'D:/2020/NLP/Practical/uni'
filelist = PlaintextCorpusReader(corpus_root, '.*')print ('\n
File list: \n')
print (filelist.fileids())print

(filelist.root)

'''display other information about each text, by looping over all the values of fileid corresponding to the filelist file identifiers listed earlier and then computing statisticsfor each text.'''

print ('\n\nStatistics for each text:\n')print
('AvgWordLen\tAvgSentenceLen\tno.ofTimesEachWordAppearsOnAvg\tFileName')
for fileid in filelist.fileids():
num_chars = len(filelist.raw(fileid)) num_words
= len(filelist.words(fileid))num_sents =
len(filelist.sents(fileid))
num_vocab = len(set([w.lower() for w in filelist.words(fileid)]))
print (int(num_chars/num_words),'\t\t\t', int(num_words/num_sents),'\t\t\t',
int(num_words/num_vocab),'\t\t', fileid)

output:


```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AMD64)
] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
============== RESTART: D:/2020/NLP/Practical/uni/p2b_ownCorpus.py =============

 File list:

['TTS.py', 'male.txt', 'p1soundtospeech.py', 'p2acorpus.py', 'p2b_ownCorpus.py']
D:\2020\NLP\Practical\uni


Statistics for each text:

AvgWordLen      AvgSentenceLen  no.ofTimesEachWordAppearsOnAvg  FileName
4                        14                        2                TTS.py
5                        140                       1                male.txt
5                        20                        1                p1soundtospeech.py
4                        38                        2                p2acorpus.py
4                        33                        2                p2b_ownCorpus.py
>>>
```

c. Study Conditional frequency distributions
source code:

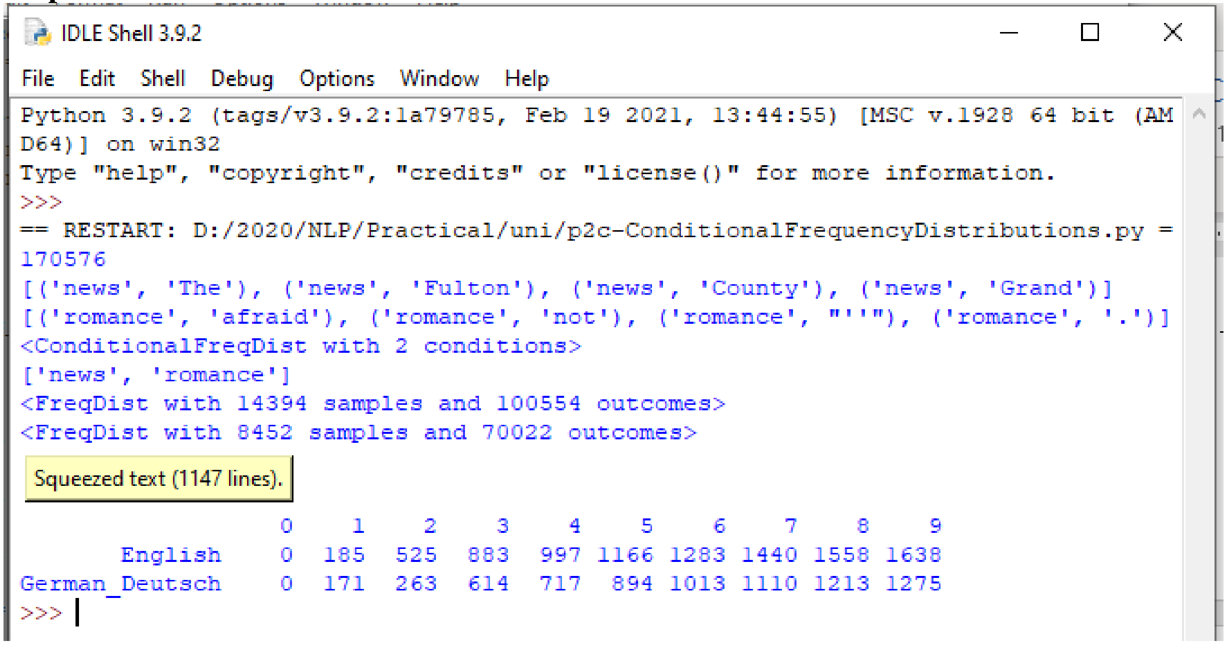
#process a sequence of pairs
text = ['The', 'Fulton', 'County', 'Grand', 'Jury', 'said', ...]
pairs = [('news', 'The'), ('news', 'Fulton'), ('news', 'County'), ...]import
nltk
from nltk.corpus import brownfd =
nltk.ConditionalFreqDist(
(genre, word)
for genre in brown.categories()
for word in brown.words(categories=genre))

genre_word = [(genre, word)
for genre in ['news', 'romance']
for word in brown.words(categories=genre)]

print(len(genre_word))

```
print(genre_word[:4])

print(genre_word[-4:])

cfd = nltk.ConditionalFreqDist(genre_word)

print(cfd)

print(cfd.conditions())

print(cfd['news'])
print(cfd['romance'])
print(list(cfd['romance']))

from nltk.corpus import inauguralcfd =
nltk.ConditionalFreqDist(
(target, fileid[:4])
for fileid in inaugural.fileids() for w in
inaugural.words(fileid)for target in ['america',
'citizen']if w.lower().startswith(target))

from nltk.corpus import udhr
languages = ['Chickasaw', 'English', 'German_Deutsch',
'Greenlandic_Inuktikut', 'Hungarian_Magyar', 'Ibibio_Efik']
cfd = nltk.ConditionalFreqDist((lang,
len(word))
for lang in languages
for word in udhr.words(lang + '-Latin1'))

cfd.tabulate(conditions=['English', 'German_Deutsch'],
samples=range(10), cumulative=True)
```

output:



```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
== RESTART: D:/2020/NLP/Practical/uni/p2c-ConditionalFrequencyDistributions.py =
170576
[('news', 'The'), ('news', 'Fulton'), ('news', 'County'), ('news', 'Grand')]
[('romance', 'afraid'), ('romance', 'not'), ('romance', "''"), ('romance', '.')]
<ConditionalFreqDist with 2 conditions>
['news', 'romance']
<FreqDist with 14394 samples and 100554 outcomes>
<FreqDist with 8452 samples and 70022 outcomes>
Squeezed text (1147 lines).
                  0    1    2    3    4    5    6    7    8    9
       English    0  185  525  883  997 1166 1283 1440 1558 1638
German_Deutsch    0  171  263  614  717  894 1013 1110 1213 1275
>>>
```

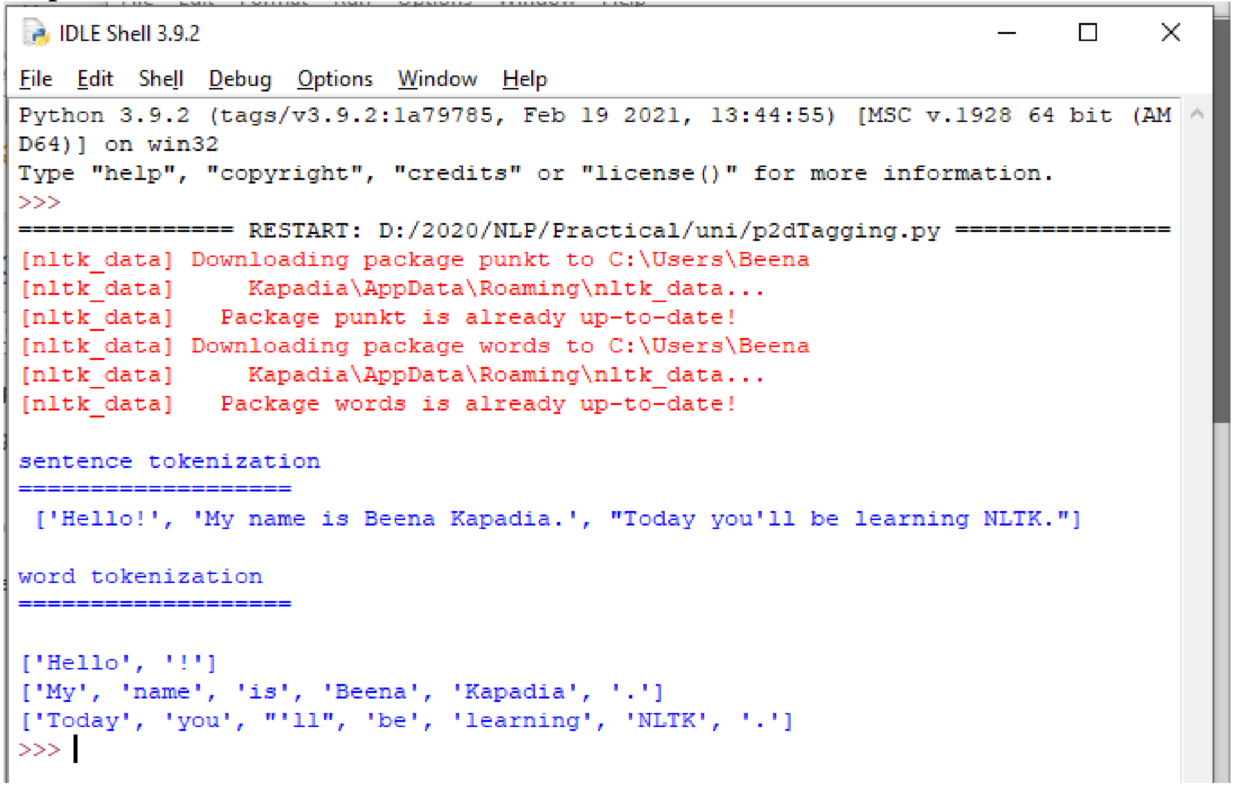
d. Study of tagged corpora with methods like tagged_sents, tagged_words.

Source code:
import nltk
from nltk import tokenize
nltk.download('punkt')
nltk.download('words')

para = "Hello! My name is Beena Kapadia. Today you'll be learning NLTK."sents =
tokenize.sent_tokenize(para)
print("\nsentence tokenization\n===================\n",sents)

# word tokenization
print("\nword tokenization\n===================\n")for
index in range(len(sents)):
words = tokenize.word_tokenize(sents[index])
print(words)

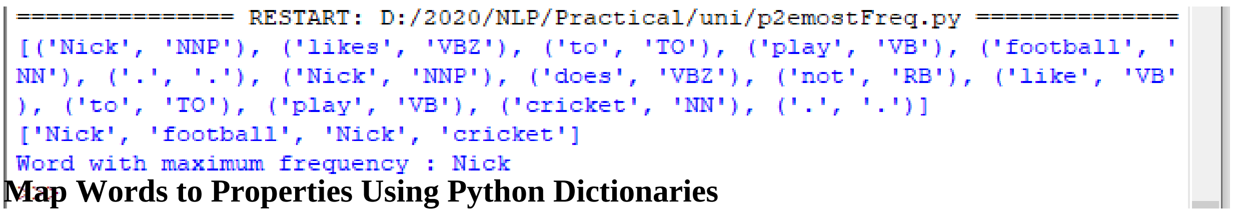
output:



```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
=============== RESTART: D:/2020/NLP/Practical/uni/p2dTagging.py ===============
[nltk_data] Downloading package punkt to C:\Users\Beena
[nltk_data]     Kapadia\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package words to C:\Users\Beena
[nltk_data]     Kapadia\AppData\Roaming\nltk_data...
[nltk_data]   Package words is already up-to-date!

sentence tokenization
===================
 ['Hello!', 'My name is Beena Kapadia.', "Today you'll be learning NLTK."]

word tokenization
===================

['Hello', '!']
['My', 'name', 'is', 'Beena', 'Kapadia', '.']
['Today', 'you', "'ll", 'be', 'learning', 'NLTK', '.']
>>>
```

e. Write a program to find the most frequent noun tags.
Code:
import nltk
from collections import defaultdict
text = nltk.word_tokenize("Nick likes to play football. Nick does not like to play
cricket.")
tagged = nltk.pos_tag(text)
print(tagged)

```
# checking if it is a noun or not
addNounWords = []
count=0
for words in tagged:
val = tagged[count][1]
if(val == 'NN' or val == 'NNS' or val == 'NNPS' or val == 'NNP'):
addNounWords.append(tagged[count][0])
count+=1

print (addNounWords)temp =

defaultdict(int)

# memoizing count
 for sub in addNounWords:for wrd
    in sub.split(): temp[wrd] += 1

# getting max frequency
res = max(temp, key=temp.get)

# printing result
print("Word with maximum frequency : " + str(res))output:
```



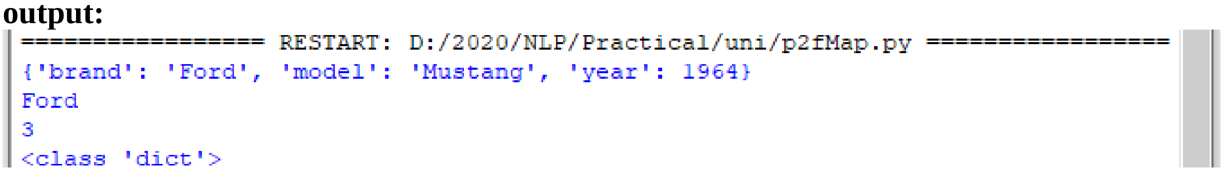
```
=============== RESTART: D:/2020/NLP/Practical/uni/p2emostFreq.py ==============
[('Nick', 'NNP'), ('likes', 'VBZ'), ('to', 'TO'), ('play', 'VB'), ('football', '
NN'), ('.', '.'), ('Nick', 'NNP'), ('does', 'VBZ'), ('not', 'RB'), ('like', 'VB'
), ('to', 'TO'), ('play', 'VB'), ('cricket', 'NN'), ('.', '.')]
['Nick', 'football', 'Nick', 'cricket']
Word with maximum frequency : Nick
```

f. Map Words to Properties Using Python Dictionaries

code:

```
#creating and printing a dictionay by mapping word with its propertiesthisdict
= {
"brand": "Ford",
"model": "Mustang","year":
1964
}
print(thisdict)
print(thisdict["brand"])
print(len(thisdict))
print(type(thisdict))
```

output:



```
================= RESTART: D:/2020/NLP/Practical/uni/p2fMap.py =================
{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}
Ford
3
<class 'dict'>
```

g. Study i) DefaultTagger, ii) Regular expression tagger, iii) UnigramTagger

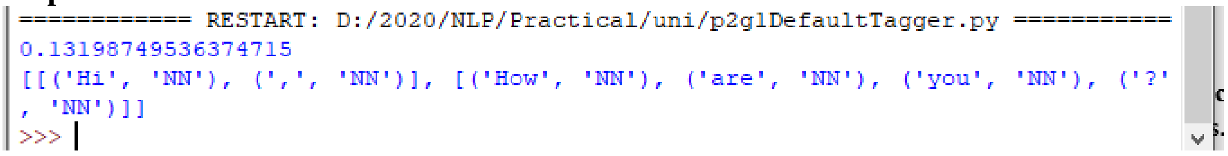
i) DefaultTagger

code:

import nltk
from nltk.tag import DefaultTagger
exptagger = DefaultTagger('NN') from
nltk.corpus import treebank
testsentences = treebank.tagged_sents() [1000:]
print(exptagger.evaluate (testsentences))

#Tagging a list of sentencesimport
nltk
from nltk.tag import DefaultTagger
exptagger = DefaultTagger('NN')
print(exptagger.tag_sents([['Hi', ','], ['How', 'are', 'you', '?']]))

output

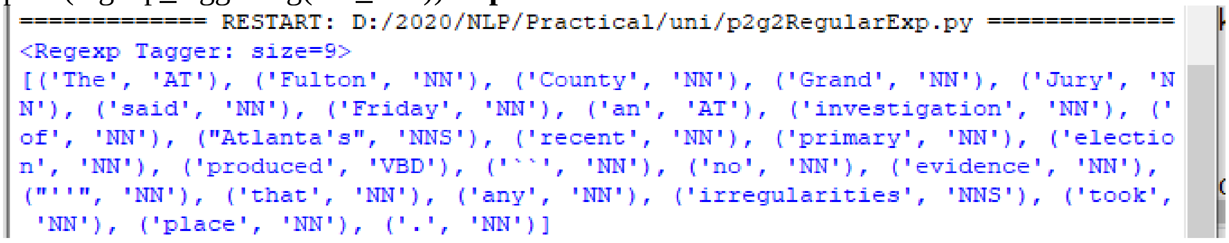


```
============ RESTART: D:/2020/NLP/Practical/uni/p2g1DefaultTagger.py ===========
0.13198749536374715
[[('Hi', 'NN'), (',', 'NN')], [('How', 'NN'), ('are', 'NN'), ('you', 'NN'), ('?'
, 'NN')]]
>>> 
```

ii) Regular expression tagger,

code:

from nltk.corpus import brown from
nltk.tag import RegexpTagger
test_sent = brown.sents(categories='news')[0]
regexp_tagger = RegexpTagger(
[(r'^-?[0-9]+(.[0-9]+)?$', 'CD'), # cardinal numbers
(r'(The|the|A|a|An|an)$', 'AT'), # articles (r'.*able$', 'JJ'), #
adjectives
(r'.*ness$', 'NN'), # nouns formed from adjectives
(r'.*ly$', 'RB'), # adverbs
(r'.*s$', 'NNS'), # plural nouns
(r'.*ing$', 'VBG'), # gerunds
(r'.*ed$', 'VBD'), # past tense verbs
(r'.*', 'NN') # nouns (default)
])
print(regexp_tagger)
print(regexp_tagger.tag(test_sent))**output:**



```
============= RESTART: D:/2020/NLP/Practical/uni/p2g2RegularExp.py =============
<Regexp Tagger: size=9>
[('The', 'AT'), ('Fulton', 'NN'), ('County', 'NN'), ('Grand', 'NN'), ('Jury', 'N
N'), ('said', 'NN'), ('Friday', 'NN'), ('an', 'AT'), ('investigation', 'NN'), ('
of', 'NN'), ("Atlanta's", 'NNS'), ('recent', 'NN'), ('primary', 'NN'), ('electio
n', 'NN'), ('produced', 'VBD'), ('``', 'NN'), ('no', 'NN'), ('evidence', 'NN'),
("''", 'NN'), ('that', 'NN'), ('any', 'NN'), ('irregularities', 'NNS'), ('took',
 'NN'), ('place', 'NN'), ('.', 'NN')]
```

iii) UnigramTagger

code:

# Loading Libraries
from nltk.tag import UnigramTaggerfrom
nltk.corpus import treebank

# Training using first 10 tagged sentences of the treebank corpus as data.# Using
data
train_sents = treebank.tagged_sents()[:10]

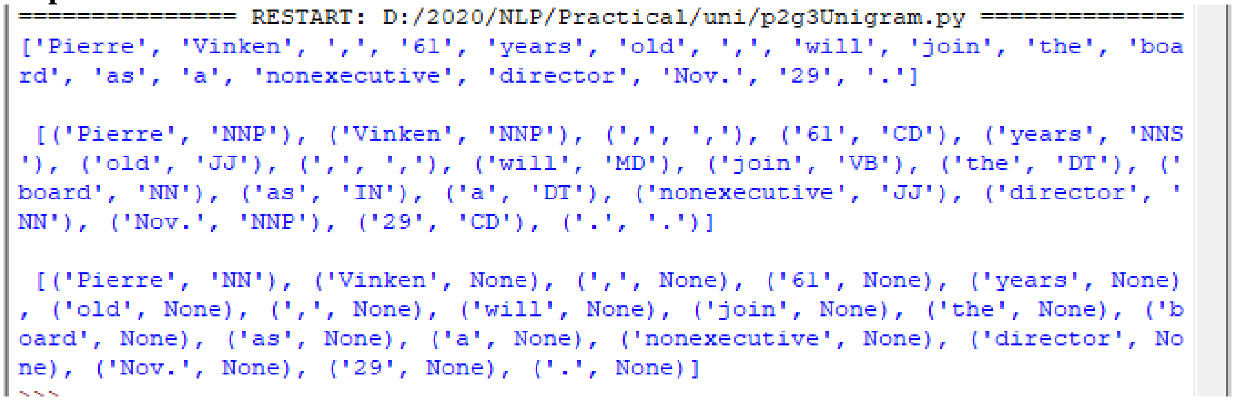
# Initializing
tagger = UnigramTagger(train_sents)

# Lets see the first sentence
# (of the treebank corpus) as list
print(treebank.sents()[0])
print('\n',tagger.tag(treebank.sents()[0]))

#Finding the tagged results after training.
tagger.tag(treebank.sents()[0])

#Overriding the context model
tagger = UnigramTagger(model ={'Pierre': 'NN'})
print('\n',tagger.tag(treebank.sents()[0]))

output:



```
================ RESTART: D:/2020/NLP/Practical/uni/p2g3Unigram.py ==============
['Pierre', 'Vinken', ',', '61', 'years', 'old', ',', 'will', 'join', 'the', 'boa
rd', 'as', 'a', 'nonexecutive', 'director', 'Nov.', '29', '.']

 [('Pierre', 'NNP'), ('Vinken', 'NNP'), (',', ','), ('61', 'CD'), ('years', 'NNS
'), ('old', 'JJ'), (',', ','), ('will', 'MD'), ('join', 'VB'), ('the', 'DT'), ('
board', 'NN'), ('as', 'IN'), ('a', 'DT'), ('nonexecutive', 'JJ'), ('director', '
NN'), ('Nov.', 'NNP'), ('29', 'CD'), ('.', '.')]

 [('Pierre', 'NN'), ('Vinken', None), (',', None), ('61', None), ('years', None)
, ('old', None), (',', None), ('will', None), ('join', None), ('the', None), ('b
oard', None), ('as', None), ('a', None), ('nonexecutive', None), ('director', No
ne), ('Nov.', None), ('29', None), ('.', None)]
```

h. Find different words from a given plain text without any space by comparing this text with a given corpus of words. Also find the score of words.

Question:

Initialize the hash tag test data or URL test data and convert to plain text without any space.. Read a text file of different words and compare the plain text data with the words exist in that text file and find out different words available in that plain text. Alsofind out how many words could be found. (for example, text = "#whatismyname" or text = www.whatismyname.com. Convert that to plain text without space as: whatismyname and read text file as words.txt. Now compare plain text with words given in a file and find the words form the plain text and the count of words which could be found)

Source code:

from___future__________import with_statement #with statement for reading file
import re # Regular expression

```
words = [] # corpus file wordstestword
= [] # test words
ans = []                # words matches with corpus

print("MENU")
print("...............................")
print(" 1 . Hash tag segmentation ")print(" 2
. URL segmentation ")
print("enter the input choice for performing word segmentation")choice =
int(input())

if choice == 1:
text = "#whatismyname"                        # hash tag test data to segment
print("input with HashTag",text) pattern=re.compile("[^\w']")
a = pattern.sub('', text)elif choice
== 2:
text = "www.whatismyname.com"                      # url test data to segment
print("input with URL",text)
a=re.split('\s|(?<!\d)[,.](?!\d)', text)
splitwords = ["www","com","in"]                    # remove the words which is containg in the list
a ="".join([each for each in a if each not in splitwords])
else:
print("wrong choice...try again")print(a)

for each in a:
testword.append(each)  #test word
test_lenth = len(testword)                  # lenth of the test data

# Reading the corpus
with open('words.txt', 'r') as f:lines =
f.readlines()
words =[(e.strip()) for e in lines]

def Seg(a,lenth):ans =[]
for k in range(0,lenth+1):  # this loop checks char by char in the corpus

if a[0:k] in words:
print(a[0:k],"-appears in the corpus")
ans.append(a[0:k])
break if ans != []:
g = max(ans,key=len)return g

test_tot_itr = 0                #each iteration value
answer = []                 # Store the each word contains the corpus
Score = 0  # initial value for score
```

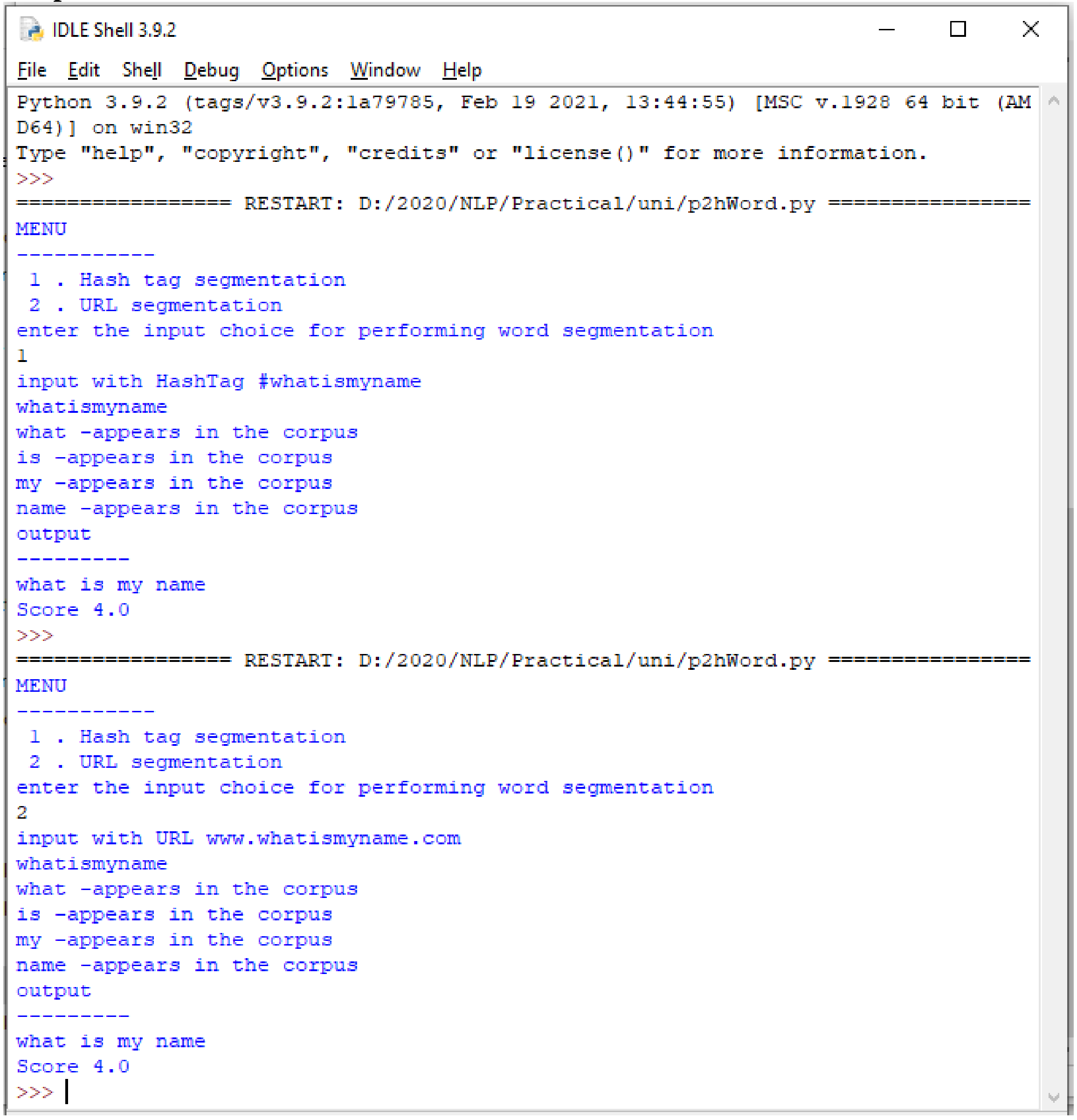
```
N = 37                # total no of corpus
M = 0
C = 0
while test_tot_itr < test_lenth: ans_words
= Seg(a,test_lenth)if ans_words != 0:
test_itr = len(ans_words)
answer.append(ans_words)a =
a[test_itr:test_lenth] test_tot_itr +=
test_itr

Aft_Seg = " ".join([each for each in answer])# print
segmented words in the list print("output")
print("------------------")
print(Aft_Seg)  # print After segmentation the input

# Calculating ScoreC =
len(answer)
score = C * N / N                # Calculate the score
print("Score",score)
```

Input:

Words.txt

```
         ---------
check domain
big rocks name
cheap being
human current
rates ought to
go down apple
domainshonesty
hour follow
back social
media30
secondsearth
this
is insaneit
time what is
my namelet
usgo
```

Output:



```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================= RESTART: D:/2020/NLP/Practical/uni/p2hWord.py ================
MENU
-----------
 1 . Hash tag segmentation
 2 . URL segmentation
enter the input choice for performing word segmentation
1
input with HashTag #whatismyname
whatismyname
what -appears in the corpus
is -appears in the corpus
my -appears in the corpus
name -appears in the corpus
output
---------
what is my name
Score 4.0
>>>
================= RESTART: D:/2020/NLP/Practical/uni/p2hWord.py ================
MENU
-----------
 1 . Hash tag segmentation
 2 . URL segmentation
enter the input choice for performing word segmentation
2
input with URL www.whatismyname.com
whatismyname
what -appears in the corpus
is -appears in the corpus
my -appears in the corpus
name -appears in the corpus
output
---------
what is my name
Score 4.0
>>>
```

Practical No. 3

a. **Study of Wordnet Dictionary with methods as synsets, definitions, examples,antonyms**

Source code:

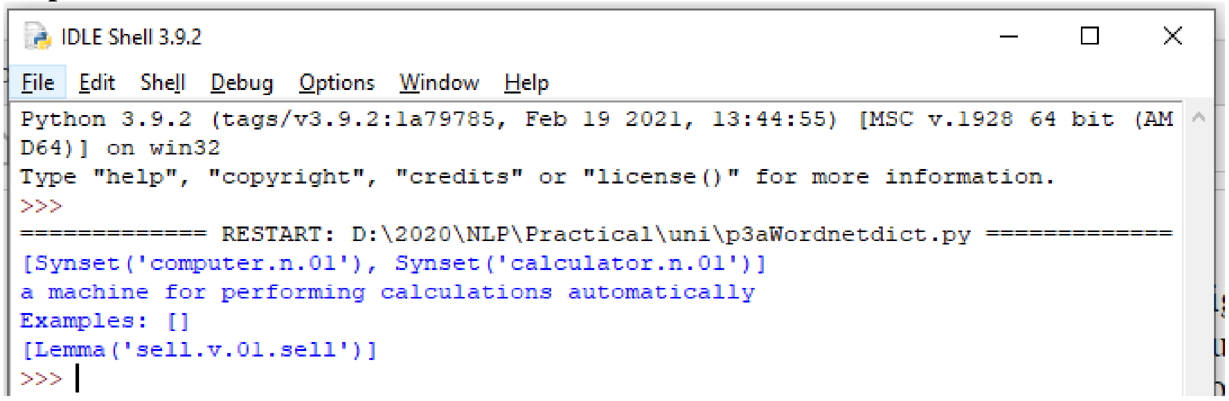
'''WordNet provides synsets which is the collection of synonym words also called “lemmas”'''
import nltk
from nltk.corpus import wordnet
print(wordnet.synsets("computer"))

# definition and example of the word ‘computer’
print(wordnet.synset("computer.n.01").definition())

#examples
print("Examples:", wordnet.synset("computer.n.01").examples())

#get Antonyms print(wordnet.lemma('buy.v.01.buy').antonyms())

output:



```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
============= RESTART: D:\2020\NLP\Practical\uni\p3aWordnetdict.py =============
[Synset('computer.n.01'), Synset('calculator.n.01')]
a machine for performing calculations automatically
Examples: []
[Lemma('sell.v.01.sell')]
>>> |
```

b. Study lemmas, hyponyms, hypernyms.
Source code:

import nltk
from nltk.corpus import wordnet
print(wordnet.synsets("computer"))
print(wordnet.synset("computer.n.01").lemma_names())#all lemmas for each synset.
for e in wordnet.synsets("computer"): print(f'{e} --> {e.lemma_names()}')

#print all lemmas for a given synset
print(wordnet.synset('computer.n.01').lemmas())

#get the synset corresponding to lemma print(wordnet.lemma('computer.n.01.computing_device').synset())

#Get the name of the lemma
print(wordnet.lemma('computer.n.01.computing_device').name())

#Hyponyms give abstract concepts of the word that are much more specific#the list
of hyponyms words of the computer

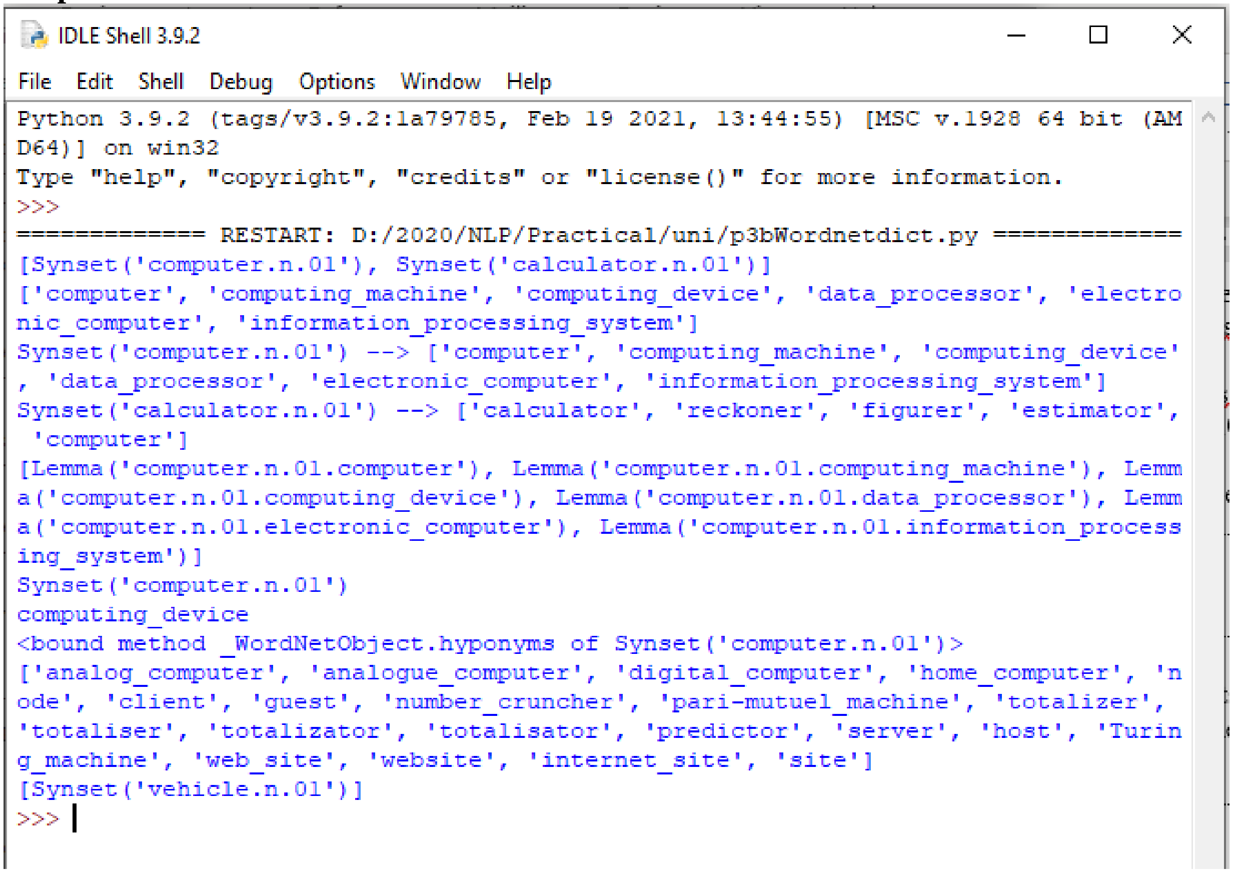
syn = wordnet.synset('computer.n.01')
print(syn.hyponyms)

print([lemma.name() for synset in syn.hyponyms() for lemma in synset.lemmas()])#the

semantic similarity in WordNet
vehicle = wordnet.synset('vehicle.n.01')car =
wordnet.synset('car.n.01')

print(car.lowest_common_hypernyms(vehicle))

Output:



```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
============= RESTART: D:/2020/NLP/Practical/uni/p3bWordnetdict.py =============
[Synset('computer.n.01'), Synset('calculator.n.01')]
['computer', 'computing_machine', 'computing_device', 'data_processor', 'electro
nic_computer', 'information_processing_system']
Synset('computer.n.01') --> ['computer', 'computing_machine', 'computing_device'
, 'data_processor', 'electronic_computer', 'information_processing_system']
Synset('calculator.n.01') --> ['calculator', 'reckoner', 'figurer', 'estimator',
 'computer']
[Lemma('computer.n.01.computer'), Lemma('computer.n.01.computing_machine'), Lemm
a('computer.n.01.computing_device'), Lemma('computer.n.01.data_processor'), Lemm
a('computer.n.01.electronic_computer'), Lemma('computer.n.01.information_process
ing_system')]
Synset('computer.n.01')
computing_device
<bound method _WordNetObject.hyponyms of Synset('computer.n.01')>
['analog_computer', 'analogue_computer', 'digital_computer', 'home_computer', 'n
ode', 'client', 'guest', 'number_cruncher', 'pari-mutuel_machine', 'totalizer',
'totaliser', 'totalizator', 'totalisator', 'predictor', 'server', 'host', 'Turin
g_machine', 'web_site', 'website', 'internet_site', 'site']
[Synset('vehicle.n.01')]
>>> |
```

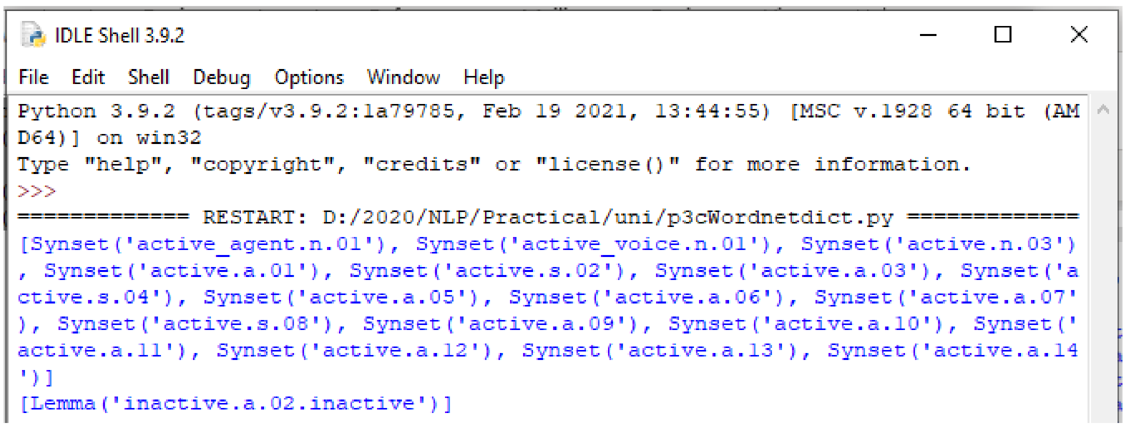
c. Write a program using python to find synonym and antonym of word "active" using Wordnet.

Source code:

from nltk.corpus import wordnetprint(
wordnet.synsets("active"))

print(wordnet.lemma('active.a.01.active').antonyms())

Output:

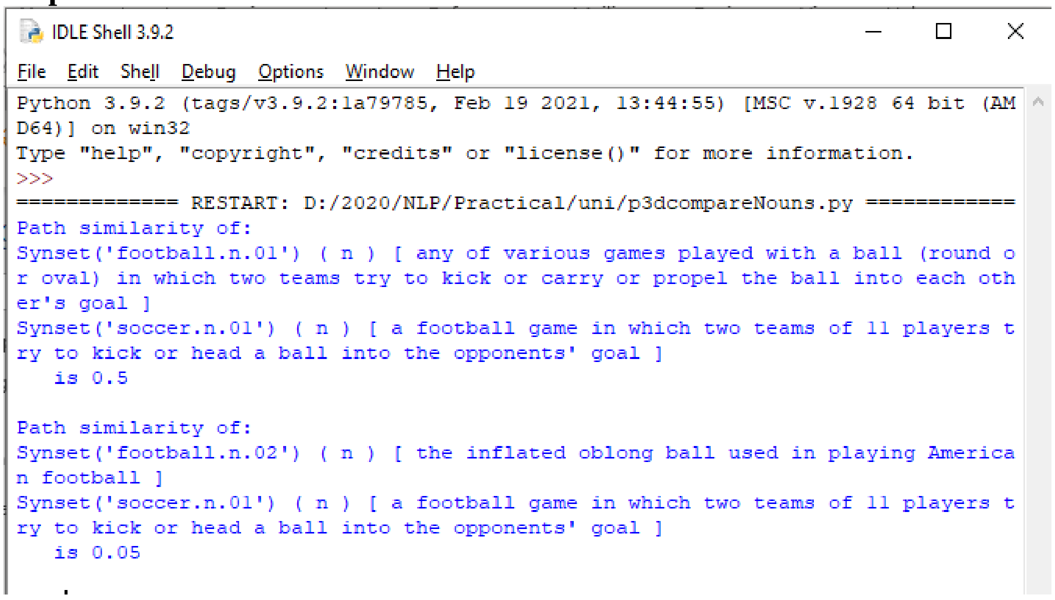


```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
============= RESTART: D:/2020/NLP/Practical/uni/p3cWordnetdict.py =============
[Synset('active_agent.n.01'), Synset('active_voice.n.01'), Synset('active.n.03')
, Synset('active.a.01'), Synset('active.s.02'), Synset('active.a.03'), Synset('a
ctive.s.04'), Synset('active.a.05'), Synset('active.a.06'), Synset('active.a.07'
), Synset('active.s.08'), Synset('active.a.09'), Synset('active.a.10'), Synset('
active.a.11'), Synset('active.a.12'), Synset('active.a.13'), Synset('active.a.14
')]
[Lemma('inactive.a.02.inactive')]
```

d. Compare two nouns
source code:

```
import nltk
from nltk.corpus import wordnet

syn1 = wordnet.synsets('football')syn2 =
wordnet.synsets('soccer')

# A word may have multiple synsets, so need to compare each synset of word1with synset
of word2
for s1 in syn1: for s2 in syn2:
print("Path similarity of: ")
print(s1, '(', s1.pos(), ')', '[', s1.definition(), ']')
print(s2, '(', s2.pos(), ')', '[', s2.definition(), ']')print("  is",
s1.path_similarity(s2))
print()
```

output:



```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
============= RESTART: D:/2020/NLP/Practical/uni/p3dcompareNouns.py ============
Path similarity of:
Synset('football.n.01') ( n ) [ any of various games played with a ball (round o
r oval) in which two teams try to kick or carry or propel the ball into each oth
er's goal ]
Synset('soccer.n.01') ( n ) [ a football game in which two teams of 11 players t
ry to kick or head a ball into the opponents' goal ]
   is 0.5

Path similarity of:
Synset('football.n.02') ( n ) [ the inflated oblong ball used in playing America
n football ]
Synset('soccer.n.01') ( n ) [ a football game in which two teams of 11 players t
ry to kick or head a ball into the opponents' goal ]
   is 0.05
```

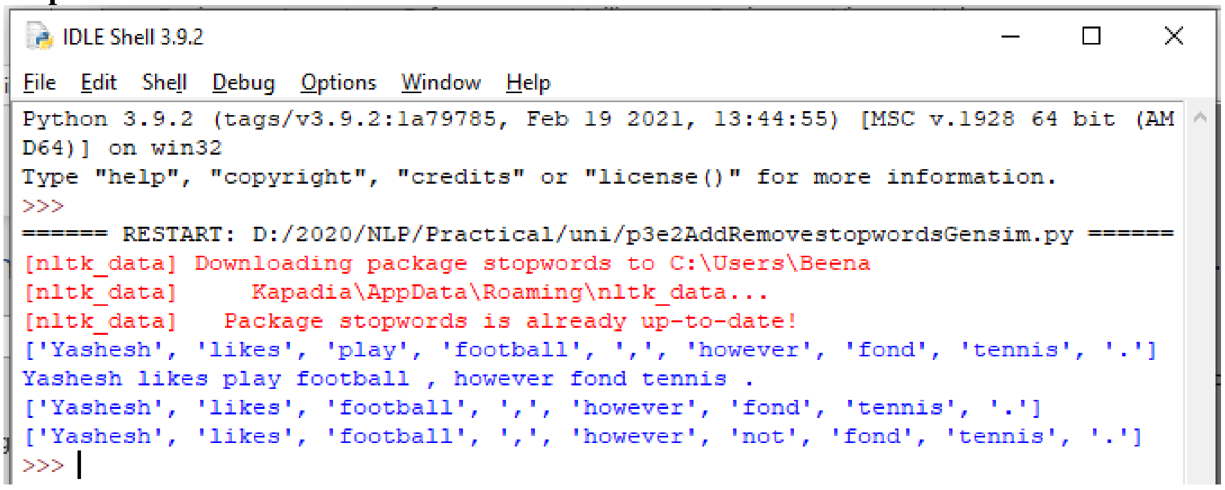
e. Handling stopword:

i) Using nltk Adding or Removing Stop Words in NLTK's Default Stop Word List

code:

```
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')
from nltk.tokenize import word_tokenize

text = "Yashesh likes to play football, however he is not too fond of tennis."text_tokens
= word_tokenize(text)

tokens_without_sw = [word for word in text_tokens if not word in
stopwords.words()]

print(tokens_without_sw)

#add the word play to the NLTK stop word collection
all_stopwords = stopwords.words('english')
all_stopwords.append('play')

text_tokens = word_tokenize(text)
tokens_without_sw = [word for word in text_tokens if not word in all_stopwords]

print(tokens_without_sw)

#remove ‘not’ from stop word collection
all_stopwords.remove('not')

text_tokens = word_tokenize(text)
tokens_without_sw = [word for word in text_tokens if not word in all_stopwords]

print(tokens_without_sw)
```

output



```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
====== RESTART: D:/2020/NLP/Practical/uni/p3e2AddRemovestopwordsGensim.py ======
[nltk_data] Downloading package stopwords to C:\Users\Beena
[nltk_data]     Kapadia\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
['Yashesh', 'likes', 'play', 'football', ',', 'however', 'fond', 'tennis', '.']
Yashesh likes play football , however fond tennis .
['Yashesh', 'likes', 'football', ',', 'however', 'fond', 'tennis', '.']
['Yashesh', 'likes', 'football', ',', 'however', 'not', 'fond', 'tennis', '.']
>>> 
```

ii) Using Gensim Adding and Removing Stop Words in Default Gensim Stop Words List

code:

```
#pip install gensimimport gensim
from gensim.parsing.preprocessing import remove_stopwords

text = "Yashesh likes to play football, however he is not too fond of tennis."
filtered_sentence = remove_stopwords(text)

print(filtered_sentence)

all_stopwords = gensim.parsing.preprocessing.STOPWORDS
print(all_stopwords)

'''The following script adds likes and play to the list of stop words in Gensim:'''from

gensim.parsing.preprocessing import STOPWORDS all_stopwords_gensim =

STOPWORDS.union(set(['likes', 'play']))

text = "Yashesh likes to play football, however he is not too fond of tennis."text_tokens =
word_tokenize(text)
tokens_without_sw = [word for word in text_tokens if not word in
all_stopwords_gensim]

print(tokens_without_sw)'''Output:

['Yashesh', 'football', ',', 'fond', 'tennis', '.']

The following script removes the word "not" from the set of stop words inGensim:'''

from gensim.parsing.preprocessing import STOPWORDS

all_stopwords_gensim = STOPWORDS
sw_list = {"not"}
all_stopwords_gensim = STOPWORDS.difference(sw_list)

text = "Yashesh likes to play football, however he is not too fond of tennis."text_tokens =
word_tokenize(text)
tokens_without_sw = [word for word in text_tokens if not word in
all_stopwords_gensim]

print(tokens_without_sw)
```

output

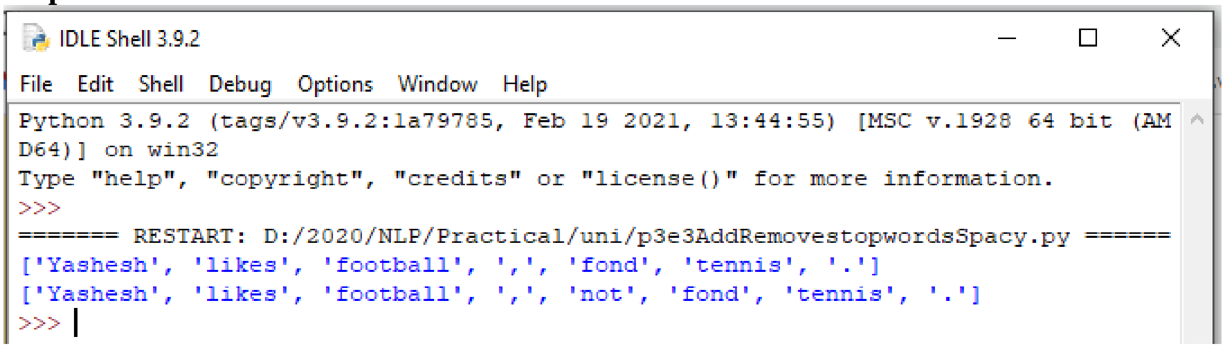
Microsoft Visual C++ 14.0 is required. Get it with "Build Tools for Visual Studio": https://visualstudio.microsoft.com/downloads/

iii) Using Spacy Adding and Removing Stop Words in Default Spacy Stop Words List

code:

```
#pip install spacy
#python -m spacy download en_core_web_sm#python -
m spacy download en

import spacyimport
nltk
from nltk.tokenize import word_tokenizesp =

spacy.load('en_core_web_sm')

#add the word play to the NLTK stop word collection
all_stopwords = sp.Defaults.stop_words
all_stopwords.add("play")

text = "Yashesh likes to play football, however he is not too fond of tennis."
text_tokens = word_tokenize(text)
tokens_without_sw = [word for word in text_tokens if not word in all_stopwords]

print(tokens_without_sw)

#remove 'not' from stop word collection
all_stopwords.remove('not')

tokens_without_sw = [word for word in text_tokens if not word in all_stopwords]

print(tokens_without_sw)
```

output:



```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
======= RESTART: D:/2020/NLP/Practical/uni/p3e3AddRemovestopwordsSpacy.py ======
['Yashesh', 'likes', 'football', ',', 'fond', 'tennis', '.']
['Yashesh', 'likes', 'football', ',', 'not', 'fond', 'tennis', '.']
>>> |
```

Practical No. 4

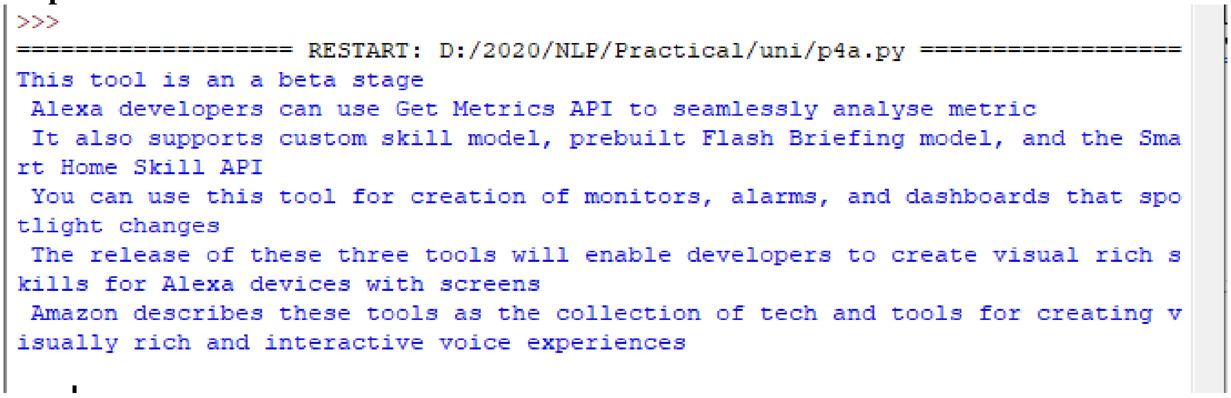
Text Tokenization

a. Tokenization using Python's split() function

code:

text = """ This tool is an a beta stage. Alexa developers can use Get Metrics API to seamlessly analyse metric. It also supports custom skill model, prebuilt Flash Briefingmodel, and the Smart Home Skill API. You can use this tool for creation of monitors, alarms, and dashboards that spotlight changes. The release of these three tools will enable developers to create visual rich skills for Alexa devices with screens. Amazon describes these tools as the collection of tech and tools for creating visually rich and interactive voice experiences. """
data = text.split('.')for i in
data:
print (i)

output:



```
>>>
=================== RESTART: D:/2020/NLP/Practical/uni/p4a.py ==================
This tool is an a beta stage
 Alexa developers can use Get Metrics API to seamlessly analyse metric
 It also supports custom skill model, prebuilt Flash Briefing model, and the Sma
rt Home Skill API
 You can use this tool for creation of monitors, alarms, and dashboards that spo
tlight changes
 The release of these three tools will enable developers to create visual rich s
kills for Alexa devices with screens
 Amazon describes these tools as the collection of tech and tools for creating v
isually rich and interactive voice experiences
```

b. Tokenization using Regular Expressions (RegEx)

code:

import nltk
# import RegexpTokenizer() method from nltkfrom
nltk.tokenize import RegexpTokenizer

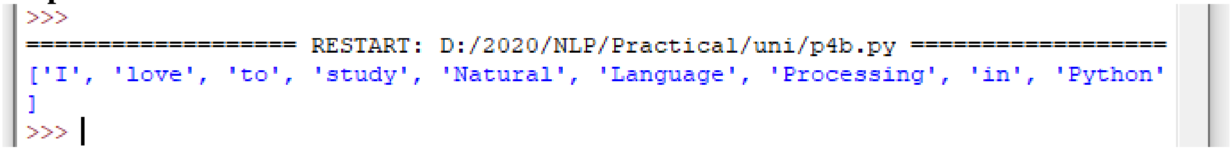
# Create a reference variable for Class RegexpTokenizertk =
RegexpTokenizer('\s+', gaps = True)

# Create a string input
str = "I love to study Natural Language Processing in Python"

# Use tokenize method tokens =
tk.tokenize(str)

print(tokens)

output:



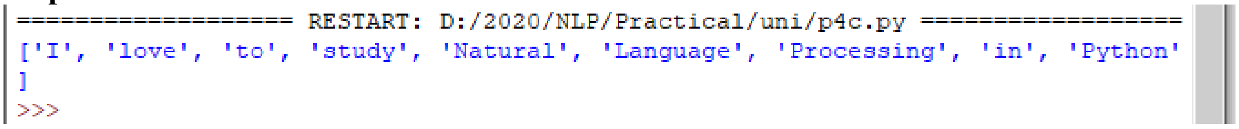
```
>>>
=================== RESTART: D:/2020/NLP/Practical/uni/p4b.py ==================
['I', 'love', 'to', 'study', 'Natural', 'Language', 'Processing', 'in', 'Python'
]
>>>
```

c. Tokenization using NLTK

code:

```
import nltk
from nltk.tokenize import word_tokenize

# Create a string input
str = "I love to study Natural Language Processing in Python"

# Use tokenize method
print(word_tokenize(str))
```

output:



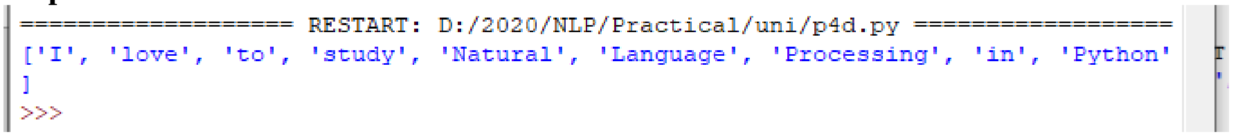
```
==================== RESTART: D:/2020/NLP/Practical/uni/p4c.py ===================
['I', 'love', 'to', 'study', 'Natural', 'Language', 'Processing', 'in', 'Python'
]
>>>
```

d. Tokenization using the spaCy library

code:

```
import spacy
nlp = spacy.blank("en")

# Create a string input
str = "I love to study Natural Language Processing in Python"

# Create an instance of document;
# doc object is a container for a sequence of Token objects.doc =
nlp(str)

# Read the words; Print the words#
words = [word.text for word in doc]
print(words)
```

output:



```
==================== RESTART: D:/2020/NLP/Practical/uni/p4d.py ===================
['I', 'love', 'to', 'study', 'Natural', 'Language', 'Processing', 'in', 'Python'
]
>>>
```

e. Tokenization using Keras

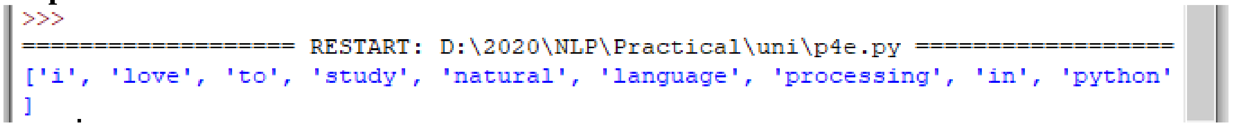
code:

```
#pip install keras
#pip install tensorflowimport
keras
from keras.preprocessing.text import text_to_word_sequence# Create

a string input
```

str = "I love to study Natural Language Processing in Python"

# tokenizing the text
tokens = text_to_word_sequence(str)
print(tokens)

output:



```
>>> 
=================== RESTART: D:\2020\NLP\Practical\uni\p4e.py ==================
['i', 'love', 'to', 'study', 'natural', 'language', 'processing', 'in', 'python'
]
```

f. **Tokenization using Gensim**

code:

#pip install gensim

from gensim.utils import tokenize# Create

a string input
str = "I love to study Natural Language Processing in Python"

# tokenizing the text
list(tokenize(str))

output:
Microsoft Visual C++ 14.0 is required. Get it with "Build Tools for Visual Studio": https://visualstudio.microsoft.com/downloads/

Practical No. 5

Import NLP Libraries for Indian Languages and perform:

Note: Execute this practical in https://colab.research.google.com/

a) word tokenization in Hindi

Source code:

```
!pip install torch==1.3.1+cpu -f https://download.pytorch.org/whl/torch_stable.html

!pip install inltk

!pip install tornado==4.5.3

from inltk.inltk import setup
setup('hi')

from inltk.inltk import tokenize

hindi_text = """प्राकृ तिक भाषा सीखना बहुि तिलचस्प है।"""

# tokenize(input text, language code)
tokenize(hindi_text, "hi")
```

output

```
['▁प्राकृ तिक', '▁भाषा', '▁सीखना', '▁बहुि', '▁तिलचस्प', '▁है', '।']
```

b) Generate similar sentences from a given Hindi text input

Source code:

```
!pip install torch==1.3.1+cpu -f https://download.pytorch.org/whl/torch_stable.html

!pip install inltk

!pip install tornado==4.5.3

from inltk.inltk import setup
setup('hi')

from inltk.inltk import get_similar_sentences

# get similar sentences to the one given in hindi
output = get_similar_sentences('मैं आज बहुि खुश हूं', 5, 'hi')

print(output)
```

Output:

```
['मैं आजकल बहुि खुश हूं', 'मैं आज अत्यतिक खुश हूं', 'मैं अभी बहुि खुश
हूं', 'मैं वतिमान बहुि खुश हूं', 'मैं वतिमान बहुि खुश हूं']
```

c) Identify the Indian language of a text

Source code:

```
!pip install torch==1.3.1+cpu -f https://download.pytorch.org/whl/torch_stable.html
```

```
!pip install inltk

!pip install tornado==4.5.3

from inltk.inltk import setup
setup('gu')

from inltk.inltk import identify_language
#Identify the Lnaguage of given text
identify_language('બીના કાપડિયા')
```

Output:

```
gujarati
```

Practical No. 6

Illustrate part of speech tagging.
g. Part of speech Tagging and chunking of user defined text.
h. Named Entity recognition of user defined text.
i. Named Entity recognition with diagram using NLTK corpus – treebank

POS Tagging, chunking and NER:

a) sentence tokenization, word tokenization, Part of speech Tagging and chunking of user defined text.

Source code:

```
import nltk
from nltk import tokenize
nltk.download('punkt') from nltk
import  tag from nltk import
chunk
nltk.download('averaged_perceptron_tagger')
nltk.download('maxent_ne_chunker') nltk.download('words')

para = "Hello! My name is Beena Kapadia. Today you'll be learning NLTK."sents =
tokenize.sent_tokenize(para)
print("\nsentence tokenization\n===================\n",sents)

# word tokenization
print("\nword tokenization\n===================\n")for
index in range(len(sents)):
words = tokenize.word_tokenize(sents[index])
print(words)



# POS Tagging tagged_words = []
for index in range(len(sents)):
tagged_words.append(tag.pos_tag(words))
print("\nPOS Tagging\n===========\n",tagged_words)


# chunkingtree = []
for index in range(len(sents)): tree.append(chunk.ne_chunk(tagged_words[index]))
print("\nchunking\n========\n")
print(tree)
```

Output:

```
sentence tokenization
===================
['Hello!', 'My name is Beena Kapadia.', "Today you'll be learning NLTK."]
```

```
word tokenization
===================

['Hello', '!']
['My', 'name', 'is', 'Beena', 'Kapadia', '.']
['Today', 'you', "'ll", 'be', 'learning', 'NLTK', '.']

POS Tagging
===========
[[('Today', 'NN'), ('you', 'PRP'), ("'ll", 'MD'), ('be', 'VB'), ('learning', 'VBG'), ('NLTK',
'NNP'), ('.', '.')], [('Today', 'NN'), ('you', 'PRP'), ("'ll", 'MD'), ('be', 'VB'), ('learning',
'VBG'), ('NLTK', 'NNP'), ('.', '.')], [('Today', 'NN'), ('you', 'PRP'), ("'ll", 'MD'), ('be','VB'),
('learning', 'VBG'), ('NLTK', 'NNP'), ('.', '.')]]

chunking
========

[Tree('S', [('Today', 'NN'), ('you', 'PRP'), ("'ll", 'MD'), ('be', 'VB'), ('learning', 'VBG'),
Tree('ORGANIZATION', [('NLTK', 'NNP')]), ('.', '.')]), Tree('S', [('Today', 'NN'), ('you',
'PRP'), ("'ll", 'MD'), ('be', 'VB'), ('learning', 'VBG'), Tree('ORGANIZATION', [('NLTK',
'NNP')]), ('.', '.')]), Tree('S', [('Today', 'NN'), ('you', 'PRP'), ("'ll", 'MD'), ('be', 'VB'), ('learning',
'VBG'), Tree('ORGANIZATION', [('NLTK', 'NNP')]), ('.', '.')])]
```

b) Named Entity recognition using user defined text.
Source code:

```
!pip install -U spacy
!python -m spacy download en_core_web_smimport
spacy

# Load English tokenizer, tagger, parser and NERnlp =
spacy.load("en_core_web_sm")

# Process whole documents
text = ("When Sebastian Thrun started working on self-driving cars at ""Google
in 2007, few people outside of the company took him " "seriously. “I can tell
you very senior CEOs of major American "
"car companies would shake my hand and turn away because I wasn’t ""worth
talking to,” said Thrun, in an interview with Recode earlier " "this week.")
doc = nlp(text)

# Analyse syntax
print("Noun phrases:", [chunk.text for chunk in doc.noun_chunks]) print("Verbs:",
[token.lemma_ for token in doc if token.pos_ == "VERB"])
```

Output:

```
Noun phrases: ['Sebastian Thrun', 'self-driving cars', 'Google', 'few people', 'the company',
'him', 'I', 'you', 'very senior CEOs', 'major American car companies', 'myhand', 'I', 'Thrun',
'an interview', 'Recode']
```

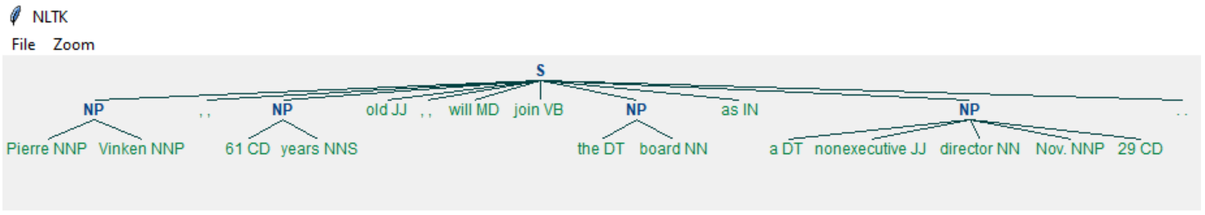
Verbs: ['start', 'work', 'drive', 'take', 'tell', 'shake', 'turn', 'be', 'talk', 'say']

c) Named Entity recognition with diagram using NLTK corpus – treebank.
Source code:

Note: It runs on Python IDLE

```
import nltk
nltk.download('treebank')
from nltk.corpus import treebank_chunk
treebank_chunk.tagged_sents()[0]

treebank_chunk.chunked_sents()[0]
treebank_chunk.chunked_sents()[0].draw()
```

Output:



Practical No. 7

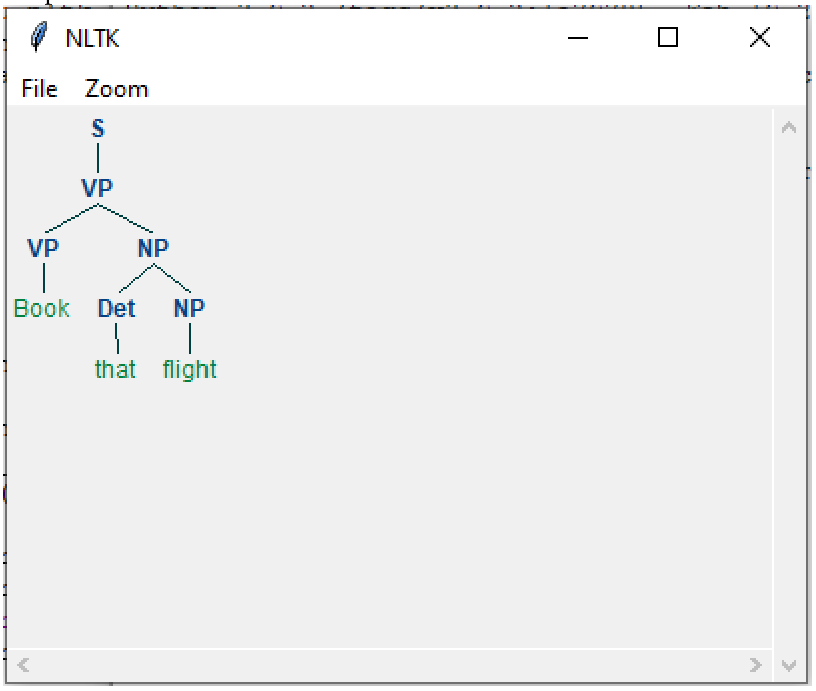
Finite state automata

a) Define grammar using nltk. Analyze a sentence using the same.

Code:

```
import nltk
from nltk import tokenize
grammar1 = nltk.CFG.fromstring("""S -> VP
VP -> VP NP NP -> Det  NP
Det -> 'that'
NP -> singular NounNP -> 'flight'
VP -> 'Book'""")
sentence = "Book that flight"

for index in range(len(sentence)):
all_tokens = tokenize.word_tokenize(sentence)
print(all_tokens)

parser = nltk.ChartParser(grammar1)for tree in
parser.parse(all_tokens):
print(tree) tree.draw()
```

output:



b) Accept the input string with Regular expression of Finite Automaton: 101+.

Source code:

```
def FA(s):
#if the length is less than 3 then it can't be accepted, Therefore end the process.if
len(s)<3:
```

```
return "Rejected"
#first three characters are fixed. Therefore, checking them using indexif
s[0]=='1':
if s[1]=='0':
if s[2]=='1':
# After index 2 only "1" can appear. Therefore break the process if any othercharacter is
detected
for i in range(3,len(s)):if s[i]!='1':
return "Rejected"
return "Accepted" # if all 4 nested if truereturn "Rejected"
# else of 3rd if
return "Rejected" # else of 2nd ifreturn
"Rejected" # else of 1st if
inputs=['1','10101','101','10111','01010','100',",'10111101','1011111']
for i in inputs:
print(FA(i))
```

Output:

```
Rejected Rejected
Accepted
Accepted
Rejected Rejected
Rejected Rejected
Accepted
```

c) **Accept the input string with Regular expression of FA: (a+b)*bba.**
Code:

```
def FA(s):size=0
#scan complete string and make sure that it contains only 'a' & 'b'for i in s:
if i=='a' or i=='b':size+=1
else:
return "Rejected"
#After checking that it contains only 'a' & 'b'#check it's
length it should be 3 atleast
if size>=3:
#check the last 3 elementsif s[size-
3]=='b':
if s[size-2]=='b':
if s[size-1]=='a':
return "Accepted" # if all 4 if truereturn "Rejected" # else
of 4th if
return "Rejected" # else of 3rd ifreturn "Rejected" #
else of 2nd if
```

```
return "Rejected" # else of 1st if

inputs=['bba', 'ababbba', 'abba','abb', 'baba','bbb','']for i in
inputs:
print(FA(i))
```

output:

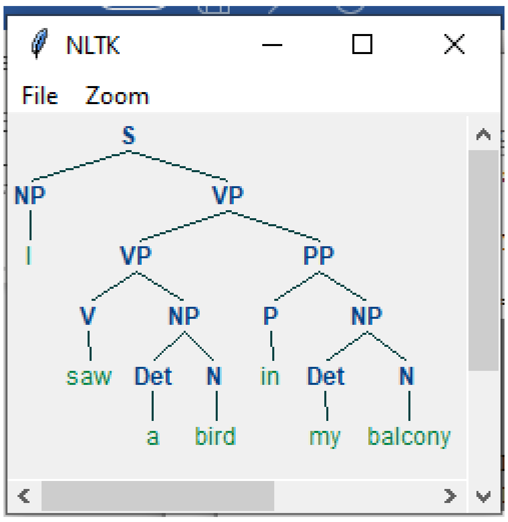
```
Rejected      Rejected
Accepted    Accepted
Rejected      Rejected
Rejected      Rejected
Accepted
```

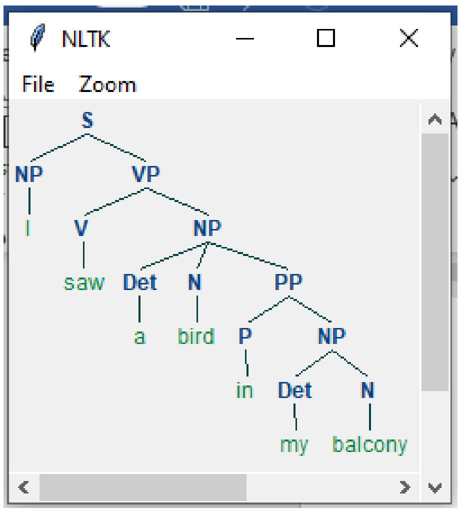
d) Implementation of Deductive Chart Parsing using context free grammar and a given sentence.

Source code:

```
import nltk
from nltk import tokenize
grammar1 = nltk.CFG.fromstring("""S -> NP
VP
PP -> P NP
NP -> Det N | Det N PP | 'I'VP -> V NP
| VP PP
Det -> 'a' | 'my'
N -> 'bird' | 'balcony'V -> 'saw'
P -> 'in'
""")
sentence = "I saw a bird in my balcony"

for index in range(len(sentence)):
all_tokens = tokenize.word_tokenize(sentence)
print(all_tokens)

# all_tokens = ['I', 'saw', 'a', 'bird', 'in', 'my', 'balcony']parser =
nltk.ChartParser(grammar1)
for tree in parser.parse(all_tokens):
print(tree)
tree.draw()
```

output:


NLTK
File Zoom
S
NP
VP
I
VP
PP
V
NP
P
NP
saw
Det
N
in
Det
N
a
bird
my
balcony


NLTK
File Zoom
S
NP
VP
I
V
NP
saw
Det
N
PP
a
bird
P
NP
in
Det
N
my
balcony

Practical No. 8

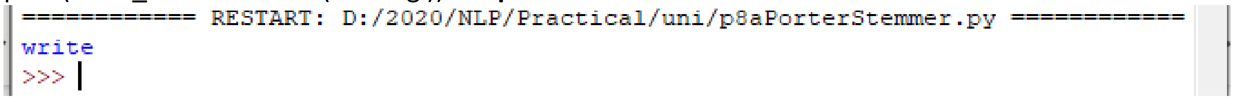
Study PorterStemmer, LancasterStemmer, RegexpStemmer, SnowballStemmer Study WordNetLemmatizer

Code:

# **PorterStemmer**

```
import nltk
from nltk.stem import PorterStemmer
word_stemmer = PorterStemmer()
print(word_stemmer.stem('writing'))
```

Output:

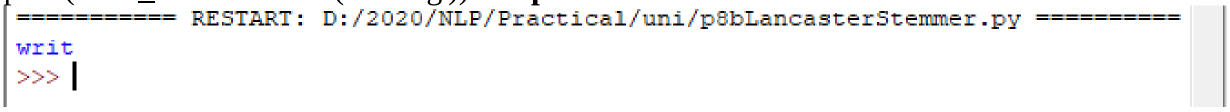


```
============ RESTART: D:/2020/NLP/Practical/uni/p8aPorterStemmer.py ============
write
>>> 
```

#LancasterStemmer

```
import nltk
from nltk.stem import LancasterStemmer
Lanc_stemmer = LancasterStemmer()
print(Lanc_stemmer.stem('writing'))
```

Output:

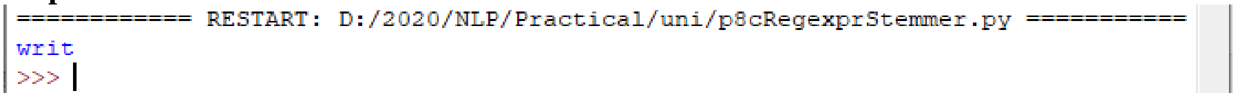


```
=========== RESTART: D:/2020/NLP/Practical/uni/p8bLancasterStemmer.py ==========
writ
>>> 
```

#RegexpStemmer

```
import nltk
from nltk.stem import RegexpStemmer
Reg_stemmer = RegexpStemmer('ing$|s$|e$|able$', min=4)
print(Reg_stemmer.stem('writing'))
```

output

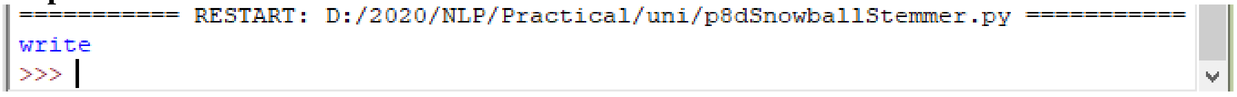


```
============ RESTART: D:/2020/NLP/Practical/uni/p8cRegexprStemmer.py ===========
writ
>>> 
```

#SnowballStemmer

```
import nltk
from nltk.stem import SnowballStemmer
english_stemmer = SnowballStemmer('english')
print(english_stemmer.stem ('writing'))
```

output



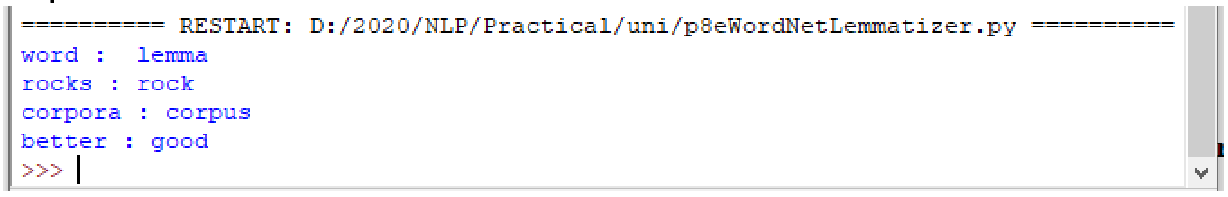
```
=========== RESTART: D:/2020/NLP/Practical/uni/p8dSnowballStemmer.py ===========
write
>>> 
```

#WordNetLemmatizer

```
from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()
```

```
print("word :\tlemma")
print("rocks :", lemmatizer.lemmatize("rocks"))
print("corpora :", lemmatizer.lemmatize("corpora"))

# a denotes adjective in "pos"
print("better :", lemmatizer.lemmatize("better", pos ="a"))
```

Output:



```
========== RESTART: D:/2020/NLP/Practical/uni/p8eWordNetLemmatizer.py ==========
word :  lemma
rocks : rock
corpora : corpus
better : good
>>> 
```

Practical No. 9

Implement Naive Bayes classifier

Code:

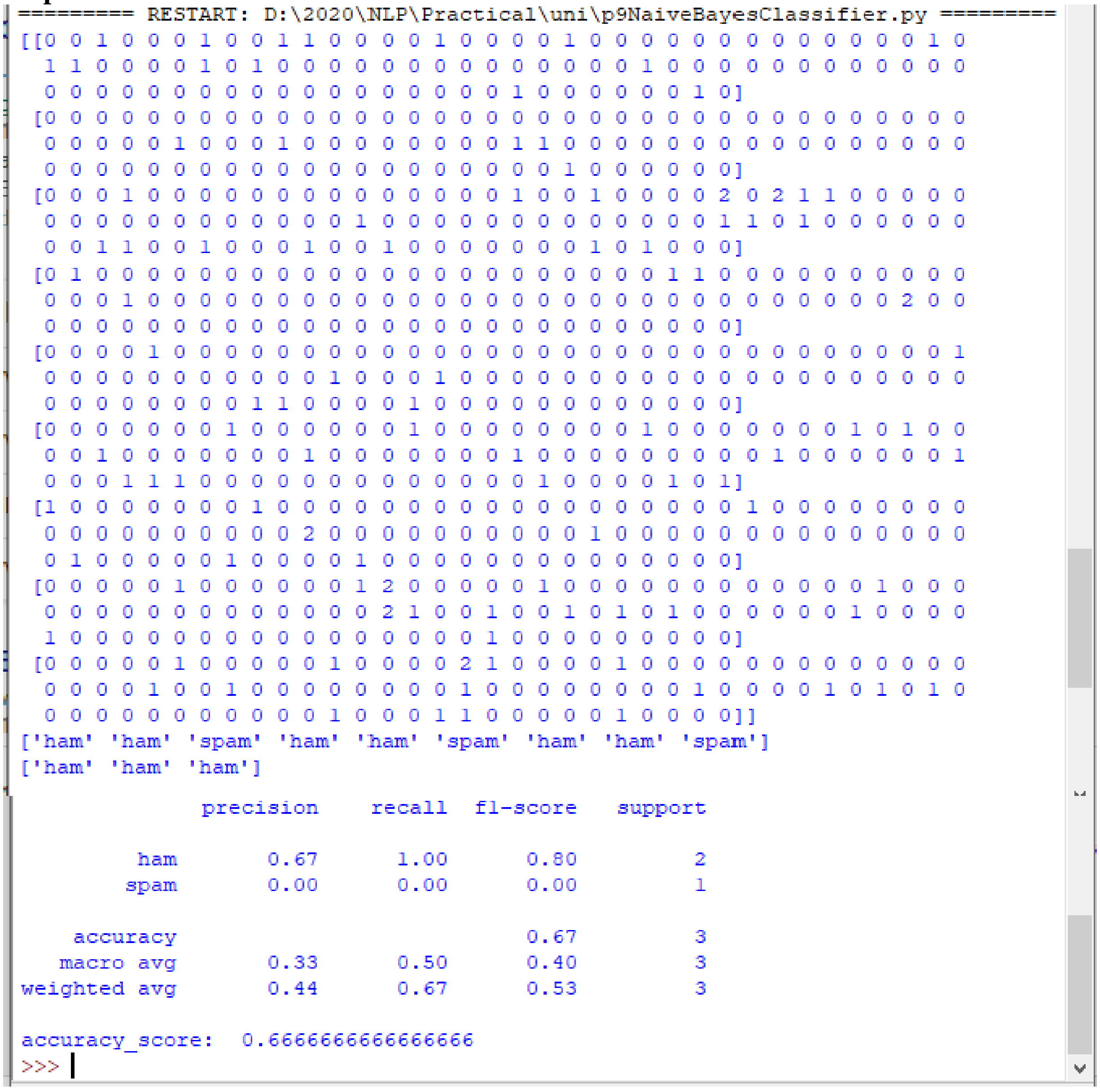
```
#pip install pandas #pip install
sklearn

import pandas as pdimport
numpy as np

sms_data = pd.read_csv("spam.csv", encoding='latin-1')import

re
import nltk
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer

stemming = PorterStemmer()corpus =
[]
for i in range (0,len(sms_data)):
s1 = re.sub('[^a-zA-Z]',repl = ' ',string = sms_data['v2'][i])s1.lower()
s1 = s1.split()
s1 = [stemming.stem(word) for word in s1 if word not in
set(stopwords.words('english'))]
s1 = ' '.join(s1)
corpus.append(s1)

from sklearn.feature_extraction.text import CountVectorizer
countvectorizer =CountVectorizer()

x = countvectorizer.fit_transform(corpus).toarray()print(x)

y = sms_data['v1'].valuesprint(y)

from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.3,
stratify=y,random_state=2)

#Multinomial Naïve Bayes.
from sklearn.naive_bayes import MultinomialNB
multinomialnb = MultinomialNB()
multinomialnb.fit(x_train,y_train)

# Predicting on test data:

y_pred = multinomialnb.predict(x_test)
print(y_pred)

#Results of our Models
```

from sklearn.metrics import classification_report, confusion_matrixfrom
sklearn.metrics import accuracy_score

print(classification_report(y_test,y_pred))
print("accuracy_score: ",accuracy_score(y_test,y_pred))

input:
spam.csv file from github

output:



```
========= RESTART: D:\2020\NLP\Practical\uni\p9NaiveBayesClassifier.py =========
[[0 0 1 0 0 0 1 0 0 1 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
  1 1 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 2 0 2 1 1 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 0 0 0 0 0 0
  0 0 1 1 0 0 1 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 1 0 0 0]
 [0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0
  0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0
  0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
  0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 1 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 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 0 1 0 0 0 0 0 0 1
  0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 1]
 [1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
  0 1 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 1 0 0 0 0 0 0 1 2 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
  0 0 0 0 0 0 0 0 0 0 0 0 0 2 1 0 0 1 0 0 1 0 1 0 1 0 0 0 0 0 0 1 0 0 0 0
  1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 2 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
  0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 1 0
  0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 1 0 0 0 0 0 1 0 0 0 0]]
['ham' 'ham' 'spam' 'ham' 'ham' 'spam' 'ham' 'ham' 'spam']
['ham' 'ham' 'ham']
              precision    recall  f1-score   support

         ham       0.67      1.00      0.80         2
        spam       0.00      0.00      0.00         1

    accuracy                           0.67         3
   macro avg       0.33      0.50      0.40         3
weighted avg       0.44      0.67      0.53         3

accuracy_score:  0.6666666666666666
>>> 
```

Practical No. 10

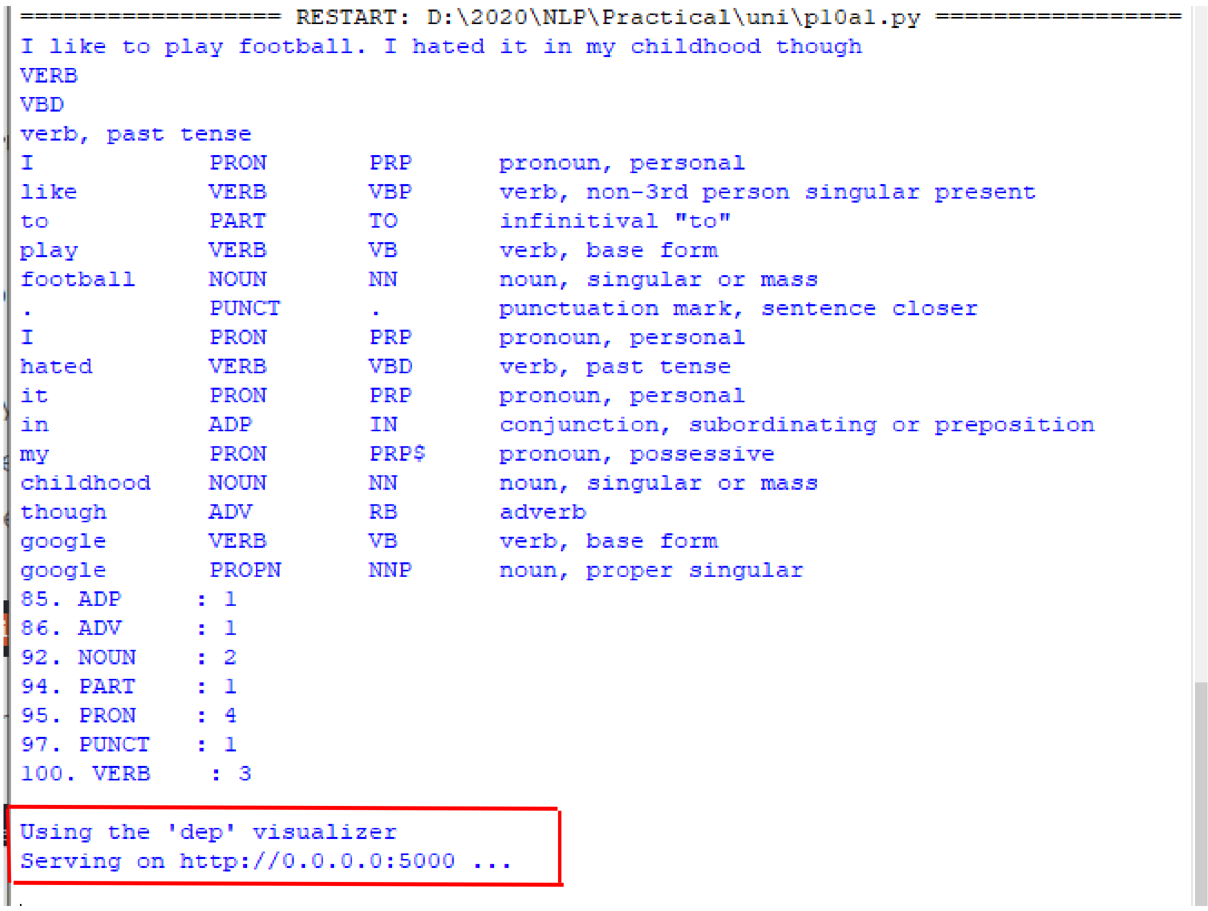
a. Speech Tagging:

i. Speech tagging using spacy

code

```
import spacy
sp = spacy.load('en_core_web_sm')
sen = sp(u"I like to play football. I hated it in my childhood though")print(sen.text)
print(sen[7].pos_) print(sen[7].tag_)
print(spacy.explain(sen[7].tag_))for word in sen:
print(f'{word.text:{12}} {word.pos_:{10}} {word.tag_:{8}}
{spacy.explain(word.tag_)}')


sen = sp(u'Can you google it?')word = sen[2]

print(f'{word.text:{12}} {word.pos_:{10}} {word.tag_:{8}}
{spacy.explain(word.tag_)}')
sen = sp(u'Can you search it on google?')word = sen[5]

print(f'{word.text:{12}} {word.pos_:{10}} {word.tag_:{8}}
{spacy.explain(word.tag_)}')

#Finding the Number of POS Tags
sen = sp(u"I like to play football. I hated it in my childhood though")

num_pos = sen.count_by(spacy.attrs.POS)num_pos

for k,v in sorted(num_pos.items()): print(f'{k}.
{sen.vocab[k].text:{8}}: {v}')

#Visualizing Parts of Speech Tagsfrom spacy import
displacy

sen = sp(u"I like to play football. I hated it in my childhood though")
displacy.serve(sen, style='dep', options={'distance': 120})
```

output:



```
================== RESTART: D:\2020\NLP\Practical\uni\p10a1.py =================
I like to play football. I hated it in my childhood though
VERB
VBD
verb, past tense
I            PRON      PRP     pronoun, personal
like         VERB      VBP     verb, non-3rd person singular present
to           PART      TO      infinitival "to"
play         VERB      VB      verb, base form
football     NOUN      NN      noun, singular or mass
.            PUNCT     .       punctuation mark, sentence closer
I            PRON      PRP     pronoun, personal
hated        VERB      VBD     verb, past tense
it           PRON      PRP     pronoun, personal
in           ADP       IN      conjunction, subordinating or preposition
my           PRON      PRP$    pronoun, possessive
childhood    NOUN      NN      noun, singular or mass
though       ADV       RB      adverb
google       VERB      VB      verb, base form
google       PROPN     NNP     noun, proper singular
85. ADP     : 1
86. ADV     : 1
92. NOUN    : 2
94. PART    : 1
95. PRON    : 4
97. PUNCT   : 1
100. VERB    : 3

Using the 'dep' visualizer
Serving on http://0.0.0.0:5000 ...
```

To view the dependency tree, type the following address in your browser: http://127.0.0.1:5000/. You will see the following dependency tree:

ii. Speech tagging using nktl

code:

import nltk
from nltk.corpus import state_union
from nltk.tokenize import PunktSentenceTokenizer

#create our training and testing data:
train_text = state_union.raw("2005-GWBush.txt") sample_text = state_union.raw("2006-GWBush.txt")

#train the Punkt tokenizer like:
custom_sent_tokenizer = PunktSentenceTokenizer(train_text)

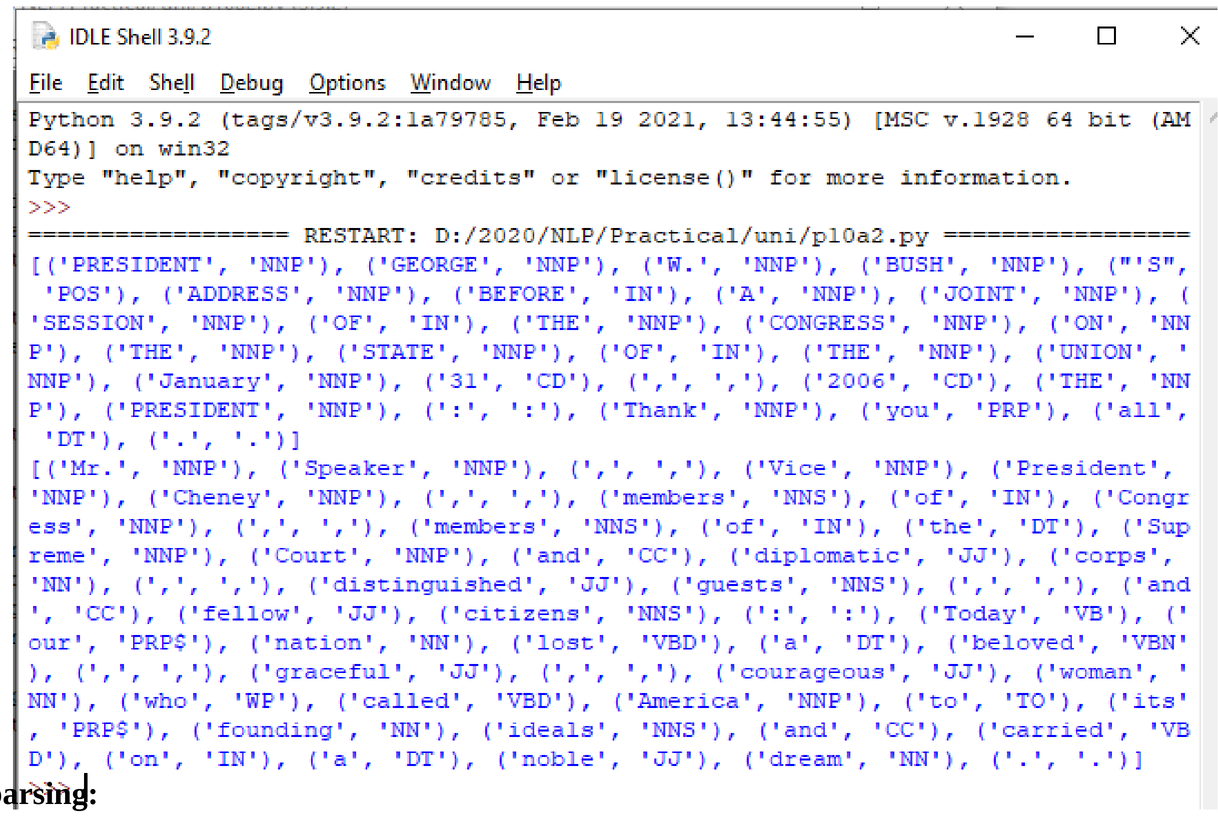
# tokenize:
tokenized = custom_sent_tokenizer.tokenize(sample_text)

def process_content():try:
for i in tokenized[:2]:
words = nltk.word_tokenize(i)tagged =
nltk.pos_tag(words)

print(tagged)

except Exception as e:print(str(e))

process_content()output:



```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
================== RESTART: D:/2020/NLP/Practical/uni/p10a2.py =================
[('PRESIDENT', 'NNP'), ('GEORGE', 'NNP'), ('W.', 'NNP'), ('BUSH', 'NNP'), ("'S",
 'POS'), ('ADDRESS', 'NNP'), ('BEFORE', 'IN'), ('A', 'NNP'), ('JOINT', 'NNP'), (
'SESSION', 'NNP'), ('OF', 'IN'), ('THE', 'NNP'), ('CONGRESS', 'NNP'), ('ON', 'NN
P'), ('THE', 'NNP'), ('STATE', 'NNP'), ('OF', 'IN'), ('THE', 'NNP'), ('UNION', '
NNP'), ('January', 'NNP'), ('31', 'CD'), (',', ','), ('2006', 'CD'), ('THE', 'NN
P'), ('PRESIDENT', 'NNP'), (':', ':'), ('Thank', 'NNP'), ('you', 'PRP'), ('all',
 'DT'), ('.', '.')]
[('Mr.', 'NNP'), ('Speaker', 'NNP'), (',', ','), ('Vice', 'NNP'), ('President',
'NNP'), ('Cheney', 'NNP'), (',', ','), ('members', 'NNS'), ('of', 'IN'), ('Congr
ess', 'NNP'), (',', ','), ('members', 'NNS'), ('of', 'IN'), ('the', 'DT'), ('Sup
reme', 'NNP'), ('Court', 'NNP'), ('and', 'CC'), ('diplomatic', 'JJ'), ('corps',
'NN'), (',', ','), ('distinguished', 'JJ'), ('guests', 'NNS'), (',', ','), ('and
', 'CC'), ('fellow', 'JJ'), ('citizens', 'NNS'), (':', ':'), ('Today', 'VB'), ('
our', 'PRP$'), ('nation', 'NN'), ('lost', 'VBD'), ('a', 'DT'), ('beloved', 'VBN'
), (',', ','), ('graceful', 'JJ'), (',', ','), ('courageous', 'JJ'), ('woman', '
NN'), ('who', 'WP'), ('called', 'VBD'), ('America', 'NNP'), ('to', 'TO'), ('its'
, 'PRP$'), ('founding', 'NN'), ('ideals', 'NNS'), ('and', 'CC'), ('carried', 'VB
D'), ('on', 'IN'), ('a', 'DT'), ('noble', 'JJ'), ('dream', 'NN'), ('.', '.')]
>>>
```

b. Statistical parsing:

i. Usage of Give and Gave in the Penn Treebank sample

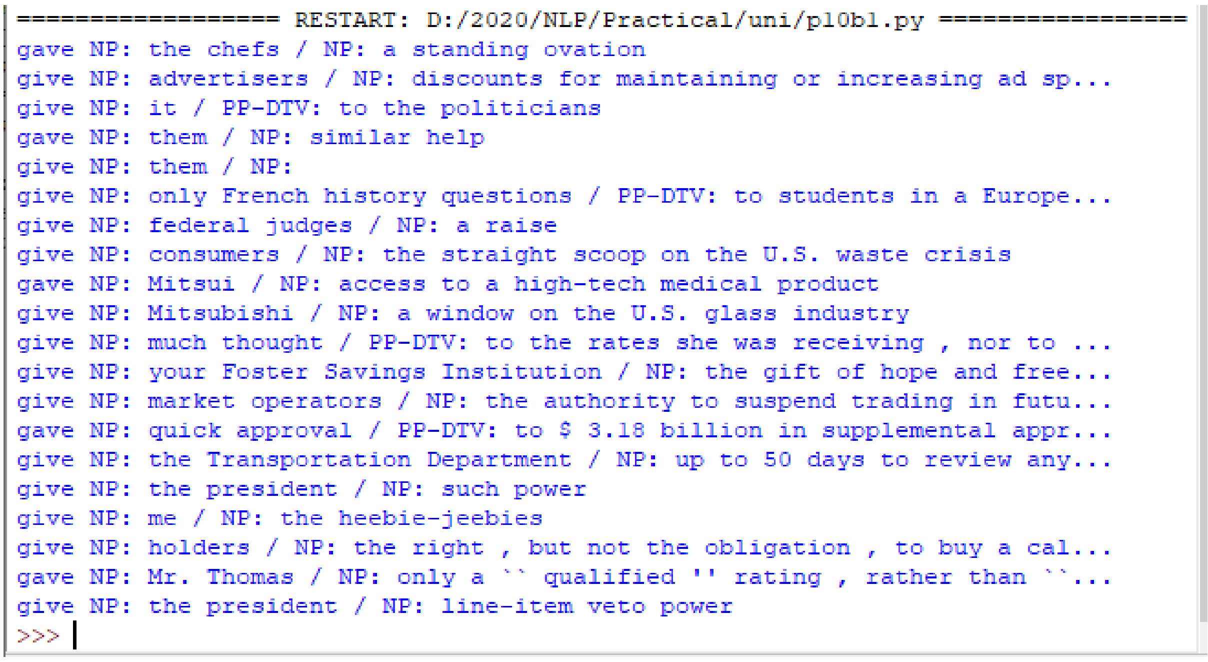
Source code:

```
#probabilitistic parser
#Usage of Give and Gave in the Penn Treebank sample

import nltk
import nltk.parse.viterbiimport
nltk.parse.pchart

def give(t):
return t.label() == 'VP' and len(t) > 2 and t[1].label() == 'NP'\and (t[2].label()
== 'PP-DTV' or t[2].label() == 'NP')\
and ('give' in t[0].leaves() or 'gave' in t[0].leaves())


def sent(t):
return ' '.join(token for token in t.leaves() if token[0] not in '*-0')
```

def print_node(t, width):
output = "%s %s: %s / %s: %s" %\
(sent(t[0]), t[1].label(), sent(t[1]), t[2].label(), sent(t[2]))if len(output) >
width:
output = output[:width] + "..."print (output)

for tree in nltk.corpus.treebank.parsed_sents():for t in
tree.subtrees(give):
print_node(t, 72)

Output:



```
================== RESTART: D:/2020/NLP/Practical/uni/p10b1.py ==================
gave NP: the chefs / NP: a standing ovation
give NP: advertisers / NP: discounts for maintaining or increasing ad sp...
give NP: it / PP-DTV: to the politicians
gave NP: them / NP: similar help
give NP: them / NP:
give NP: only French history questions / PP-DTV: to students in a Europe...
give NP: federal judges / NP: a raise
give NP: consumers / NP: the straight scoop on the U.S. waste crisis
gave NP: Mitsui / NP: access to a high-tech medical product
give NP: Mitsubishi / NP: a window on the U.S. glass industry
give NP: much thought / PP-DTV: to the rates she was receiving , nor to ...
give NP: your Foster Savings Institution / NP: the gift of hope and free...
give NP: market operators / NP: the authority to suspend trading in futu...
gave NP: quick approval / PP-DTV: to $ 3.18 billion in supplemental appr...
give NP: the Transportation Department / NP: up to 50 days to review any...
give NP: the president / NP: such power
give NP: me / NP: the heebie-jeebies
give NP: holders / NP: the right , but not the obligation , to buy a cal...
gave NP: Mr. Thomas / NP: only a `` qualified '' rating , rather than ``...
give NP: the president / NP: line-item veto power
>>> |
```

ii. probabilistic parser

Source code:
import nltk
from nltk import PCFG

grammar = PCFG.fromstring('''
NP -> NNS [0.5] | JJ NNS [0.3] | NP CC NP [0.2]
NNS -> "men" [0.1] | "women" [0.2] | "children" [0.3] | NNS CC NNS [0.4]JJ -> "old" [0.4]
| "young" [0.6]
CC -> "and" [0.9] | "or" [0.1]''')

print(grammar)

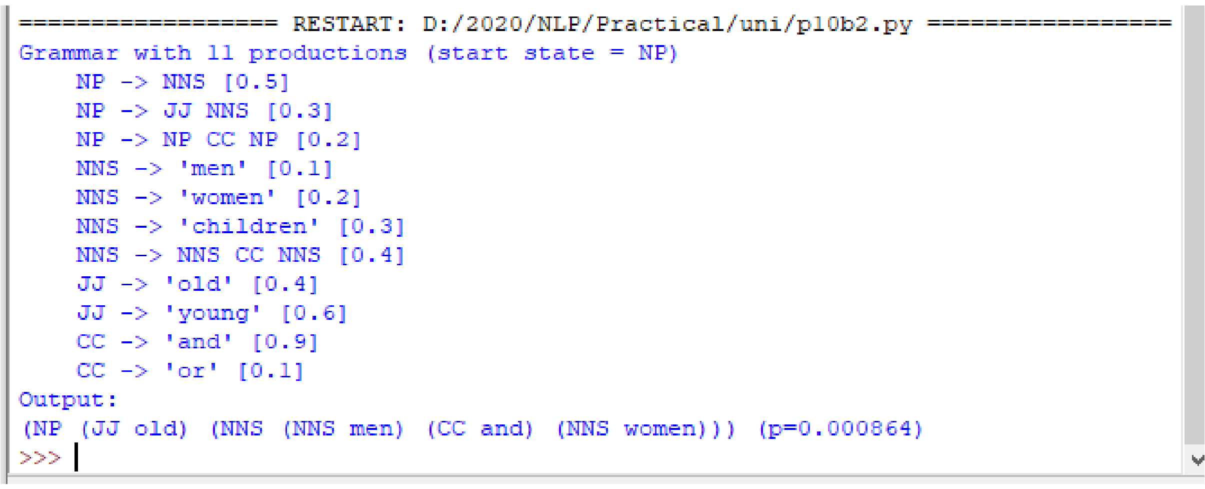
viterbi_parser = nltk.ViterbiParser(grammar)token = "old

men and women".split()

obj = viterbi_parser.parse(token)

print("Output: ")for x in obj:
print(x)

Output:



```
================== RESTART: D:/2020/NLP/Practical/uni/p10b2.py ==================
Grammar with 11 productions (start state = NP)
    NP -> NNS [0.5]
    NP -> JJ NNS [0.3]
    NP -> NP CC NP [0.2]
    NNS -> 'men' [0.1]
    NNS -> 'women' [0.2]
    NNS -> 'children' [0.3]
    NNS -> NNS CC NNS [0.4]
    JJ -> 'old' [0.4]
    JJ -> 'young' [0.6]
    CC -> 'and' [0.9]
    CC -> 'or' [0.1]
Output:
(NP (JJ old) (NNS (NNS men) (CC and) (NNS women))) (p=0.000864)
>>> 
```

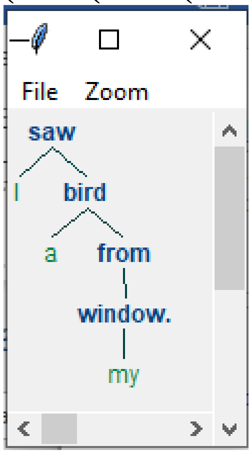
c. Malt parsing:
Parse a sentence and draw a tree using malt parsing.
Note: 1) Java should be installed.
2) maltparser-1.7.2 zip file should be copied in C:\Users\Beena Kapadia\AppData\Local\Programs\Python\Python39 folder and should be extracted in the same folder.
3) engmalt.linear-1.7.mco file should be copied to C:\Users\Beena Kapadia\AppData\Local\Programs\Python\Python39 folder
Source code:

# copy maltparser-1.7.2(unzipped version) and engmalt.linear-1.7.mco files toC:\Users\Beena Kapadia\AppData\Local\Programs\Python\Python39 folder
# java should be installed
# environment variables should be set - MALT_PARSER - C:\Users\Beena Kapadia\AppData\Local\Programs\Python\Python39\maltparser-1.7.2 and MALT_MODEL - C:\Users\Beena Kapadia\AppData\Local\Programs\Python\Python39\engmalt.linear-1.7.mco

from nltk.parse import malt
mp = malt.MaltParser('maltparser-1.7.2', 'engmalt.linear-1.7.mco')#filet = mp.parse_one('I saw a bird from my window.'.split()).tree()
print(t) t.draw()

Output:

```
(saw I (bird a (from (window. my))))
```



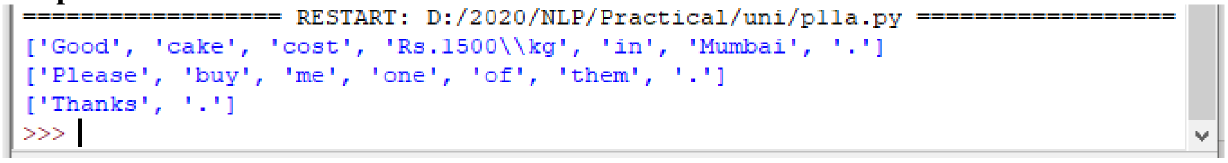
Practical No. 11

a) Multiword Expressions in NLP

Source code:

```
# Multiword Expressions in NLP

from nltk.tokenize import MWETokenizer from nltk
import sent_tokenize, word_tokenize
s = '''Good cake cost Rs.1500\kg in Mumbai. Please buy me one of them.\n\nThanks.'''mwe =
MWETokenizer([('New', 'York'), ('Hong', 'Kong')], separator='_')
for sent in sent_tokenize(s):
print(mwe.tokenize(word_tokenize(sent)))
```

Output:



```
================== RESTART: D:/2020/NLP/Practical/uni/p11a.py ==================
['Good', 'cake', 'cost', 'Rs.1500\\kg', 'in', 'Mumbai', '.']
['Please', 'buy', 'me', 'one', 'of', 'them', '.']
['Thanks', '.']
>>>
```

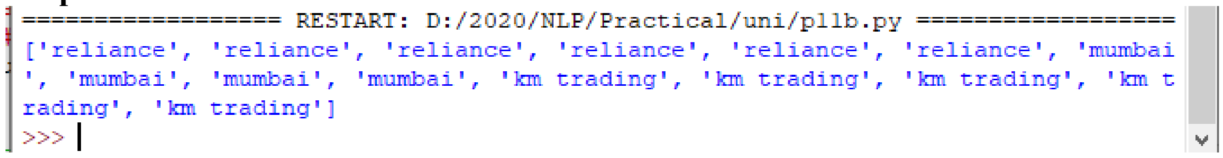
b) Normalized Web Distance and Word Similarity

Source code:

```
# Normalized Web Distance and Word Similarity

#convert

#Reliance            supermarket
#Reliance            hypermarket
#Reliance
#Reliance
#Reliance downtown
#Relianc market #Mumbai
#Mumbai Hyper
#Mumbai dxb #mumbai
airport#k.m trading #KM
Trading #KM trade
#K.M. Trading
#KM.Trading

#into

#Reliance
#Reliance
#Reliance
#Reliance
#Reliance
#Reliance
#Mumbai
#Mumbai
#Mumbai
#Mumbai
```

```
#KM    Trading  #KM
Trading#KM  Trading
#KM    Trading  #KM
Trading

import numpy as npimport
re
import textdistance # pip install textdistance# we
will need scikit-learn>=0.21
import sklearn #pip install sklearn
from sklearn.cluster import AgglomerativeClustering

texts = [
'Reliance supermarket', 'Reliance hypermarket', 'Reliance', 'Reliance', 'Reliance
downtown', 'Relianc market',
'Mumbai', 'Mumbai Hyper', 'Mumbai dxb', 'mumbai airport',
'k.m trading', 'KM Trading', 'KM trade', 'K.M.  Trading', 'KM.Trading'
]

def normalize(text):
""" Keep only lower-cased text and numbers"""return
re.sub('[^a-z0-9]+', ' ', text.lower())

def group_texts(texts, threshold=0.4):
""" Replace each text with the representative of its cluster"""
normalized_texts = np.array([normalize(text) for text in texts])distances
= 1 - np.array([
[textdistance.jaro_winkler(one, another) for one in normalized_texts]for another
in normalized_texts
])
clustering = AgglomerativeClustering(
distance_threshold=threshold, # this parameter needs to be tuned carefully
affinity="precomputed", linkage="complete", n_clusters=None
).fit(distances) centers =
dict()
for cluster_id in set(clustering.labels_):
index = clustering.labels_ == cluster_id
centrality = distances[:, index][index].sum(axis=1) centers[cluster_id] =
normalized_texts[index][centrality.argmin()] return [centers[i] for i in
clustering.labels_]

print(group_texts(texts))
```

Output:



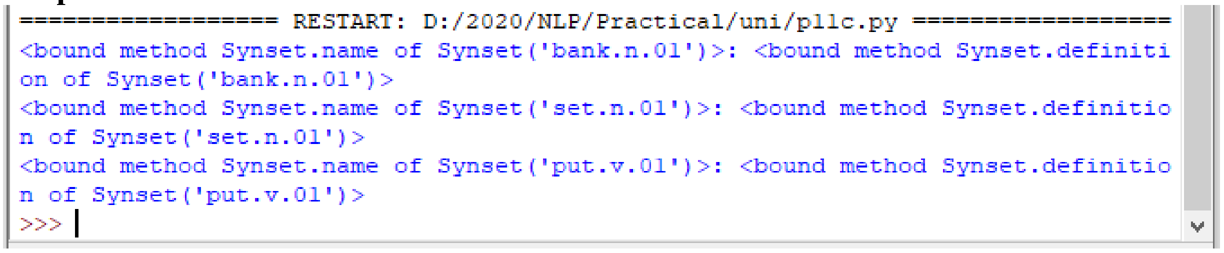
```
================== RESTART: D:/2020/NLP/Practical/uni/p11b.py ==================
['reliance', 'reliance', 'reliance', 'reliance', 'reliance', 'reliance', 'mumbai
', 'mumbai', 'mumbai', 'mumbai', 'km trading', 'km trading', 'km trading', 'km t
rading', 'km trading']
>>> 
```

c) Word Sense Disambiguation

Source code:

```
#Word Sense Disambiguation
from nltk.corpus import wordnet as wn

def get_first_sense(word,
pos=None):if pos:
synsets = wn.synsets(word,pos)
else:
synsets = wn.synsets(word)
return synsets[0]

best_synset = get_first_sense('bank')
print ('%s: %s' % (best_synset.name,
best_synset.definition))best_synset =
get_first_sense('set','n')
print ('%s: %s' % (best_synset.name,
best_synset.definition))best_synset =
get_first_sense('set','v')
print ('%s: %s' % (best_synset.name, best_synset.definition))
```

Output:



```
================== RESTART: D:/2020/NLP/Practical/uni/p11c.py ==================
<bound method Synset.name of Synset('bank.n.01')>: <bound method Synset.definiti
on of Synset('bank.n.01')>
<bound method Synset.name of Synset('set.n.01')>: <bound method Synset.definitio
n of Synset('set.n.01')>
<bound method Synset.name of Synset('put.v.01')>: <bound method Synset.definitio
n of Synset('put.v.01')>
>>> 
```

Deep Learning

Index

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

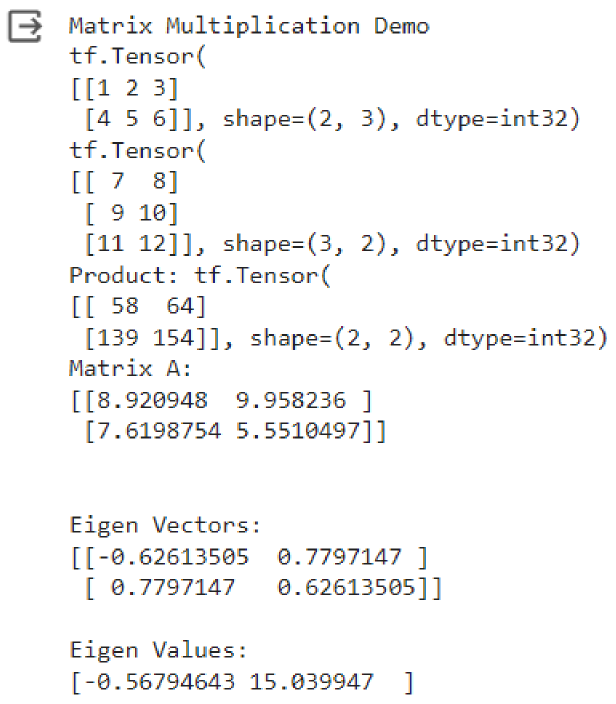
Practical No: 1

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

Code:

```
import tensorflow as tf

print("Matrix Multiplication Demo")

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

print(x)

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

print(y)

z=tf.matmul(x,y)

print("Product:",z)

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

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

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

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

Output:



```
Matrix Multiplication Demo
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:
[[8.920948  9.958236 ]
 [7.6198754 5.5510497]]


Eigen Vectors:
[[-0.62613505  0.7797147 ]
 [ 0.7797147   0.62613505]]

Eigen Values:
[-0.56794643 15.039947  ]
```

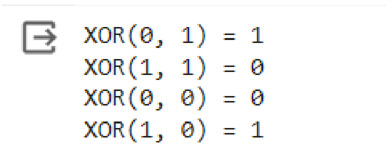
Practical No: 2

Aim: Solving XOR problem using deep feed forward network.

Code:

```
import numpy as np


def unitStep(v):
   if v >= 0:
      return 1
   else:
      return 0
def perceptronModel(x, w, b):
   v = np.dot(w, x) + b
   y = unitStep(v)
   return y
def NOT_logicFunction(x):
   wNOT = -1
   bNOT = 0.5
   return perceptronModel(x, wNOT, bNOT)
def AND_logicFunction(x):
   w = np.array([1, 1])
   bAND = -1.5
   return perceptronModel(x, w, bAND)
def OR_logicFunction(x):
    w = np.array([1, 1])

    bOR = -0.5

    return perceptronModel(x, w, bOR)

def XOR_logicFunction(x):

  y1 = AND_logicFunction(x)

  y2 = OR_logicFunction(x)

  y3 = NOT_logicFunction(y1)

  final_x = np.array([y2, y3])

  finalOutput = AND_logicFunction(final_x)

  y3 = NOT_logicFunction(y1)

  return finalOutput


test1 = np.array([0, 1])

test2 = np.array([1, 1])

test3 = np.array([0, 0])

test4 = np.array([1, 0])


print("XOR({}, {}) = {}".format(0, 1, XOR_logicFunction(test1)))

print("XOR({}, {}) = {}".format(1, 1, XOR_logicFunction(test2)))

print("XOR({}, {}) = {}".format(0, 0, XOR_logicFunction(test3)))

print("XOR({}, {}) = {}".format(1, 0, XOR_logicFunction(test4)))
```

Output:



```
XOR(0, 1) = 1
XOR(1, 1) = 0
XOR(0, 0) = 0
XOR(1, 0) = 1
```

Practical No: 3

Aim: Implementing deep neural network for performing binary classification task.

Code:

```
import pandas as pd

from keras.models import Sequential

from keras.layers import Dense

from scikeras.wrappers import KerasClassifier

from sklearn.model_selection import cross_val_score

from sklearn.preprocessing import LabelEncoder

from sklearn.model_selection import StratifiedKFold

from sklearn.preprocessing import StandardScaler

from sklearn.pipeline import Pipeline


# load dataset

dataframe = pd.read_csv("//content//sonar.all-data", header=None)

dataset = dataframe.values

# split into input (X) and output (Y) variables

X = dataset[:,0:60].astype(float)

Y = dataset[:,60]

# encode class values as integers

encoder = LabelEncoder()

encoder.fit(Y)

encoded_Y = encoder.transform(Y)
```

```
# baseline model
def create_baseline():
    # create model
    model = Sequential()
    model.add(Dense(60, input_dim=60, activation='relu'))
    model.add(Dense(1, activation='sigmoid'))
    # Compile model
    model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
    return model

# evaluate model with standardized dataset
estimator = KerasClassifier(build_fn=create_baseline, epochs=100, batch_size=5, verbose=0)
kfold = StratifiedKFold(n_splits=10, shuffle=True)
results = cross_val_score(estimator, X, encoded_Y, cv=kfold)
print("Baseline: %.2f%% (%.2f%%)" % (results.mean()*100, results.std()*100))
# evaluate baseline model with standardized dataset
estimators = []
estimators.append(('standardize', StandardScaler()))
estimators.append(('mlp', KerasClassifier(build_fn=create_baseline, epochs=100, batch_size=5,
verbose=0)))
pipeline = Pipeline(estimators)
kfold = StratifiedKFold(n_splits=10, shuffle=True)
results = cross_val_score(pipeline, X, encoded_Y, cv=kfold)
print("Standardized: %.2f%% (%.2f%%)" % (results.mean()*100, results.std()*100))
def create_smaller():
    # create model
    model = Sequential()
```

```
	model.add(Dense(30, input_dim=60, activation='relu'))
	model.add(Dense(1, activation='sigmoid'))
	model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
	return model

estimators = []
estimators.append(('standardize', StandardScaler()))
estimators.append(('mlp', KerasClassifier(build_fn=create_smaller, epochs=100, batch_size=5,
verbose=0)))
pipeline = Pipeline(estimators)
kfold = StratifiedKFold(n_splits=10, shuffle=True)
results = cross_val_score(pipeline, X, encoded_Y, cv=kfold)
print("Smaller: %.2f%% (%.2f%%)" % (results.mean()*100, results.std()*100))
# larger model
def create_larger():
	# create model
	model = Sequential()
	model.add(Dense(60, input_dim=60, activation='relu'))
	model.add(Dense(30, activation='relu'))
	model.add(Dense(1, activation='sigmoid'))
	# Compile model
	model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
	return model

estimators = []
estimators.append(('standardize', StandardScaler()))
estimators.append(('mlp', KerasClassifier(build_fn=create_larger, epochs=100, batch_size=5,
verbose=0)))
```

```
pipeline = Pipeline(estimators)

kfold = StratifiedKFold(n_splits=10, shuffle=True)

results = cross_val_score(pipeline, X, encoded_Y, cv=kfold)

print("Larger: %.2f%% (%.2f%%)" % (results.mean()*100, results.std()*100))
```

Output:

```
Baseline: 82.69% (9.24%)
Standardized: 87.52% (7.73%)
Smaller: 83.12% (5.11%)
Larger: 86.10% (7.49%)
```

Practical No: 4

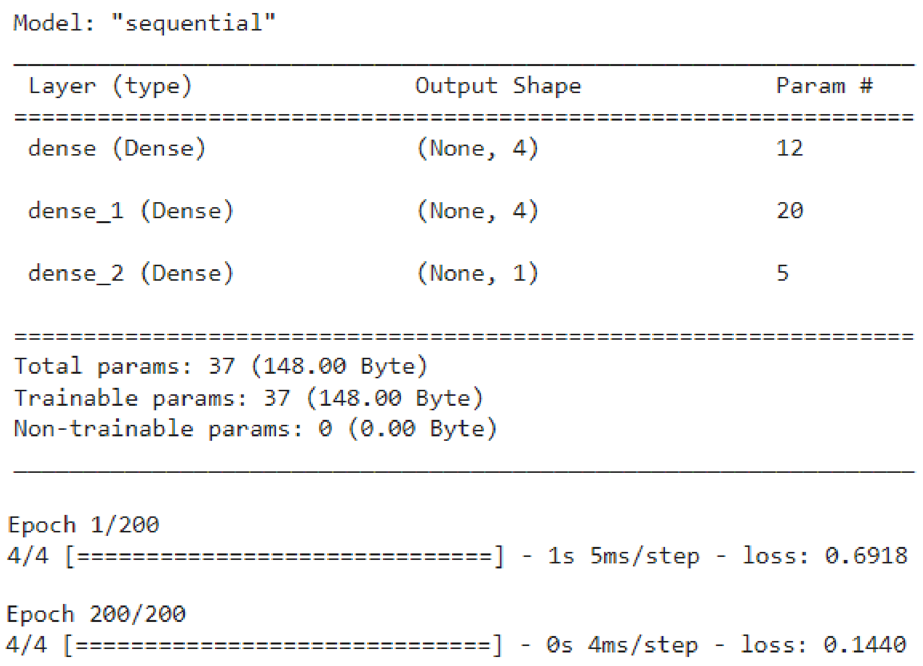
Aim: A] Using deep feed forward network with two hidden layers for performing multiclass classification and predicting the class.

Code:

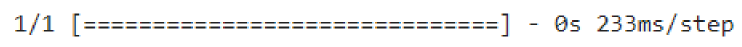
```
from keras.models import Sequential

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.summary()

model.fit(X,Y,epochs=200)


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

```
Xnew=scalar.transform(Xnew)

Yclass=model.predict(Xnew)


import numpy as np

def predict_prob(number):

  return [number[0],1-number[0]]

y_prob = np.array(list(map(predict_prob, model.predict(Xnew))))

y_prob

for i in range(len(Xnew)):

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

 #second way

predict_prob=model.predict([Xnew])

predict_classes=np.argmax(predict_prob,axis=1)

predict_classes
```

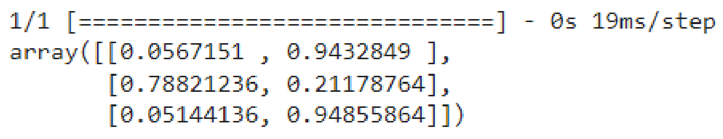
Output:



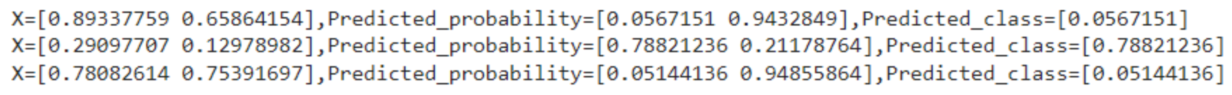
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 4)                 12

 dense_1 (Dense)             (None, 4)                 20

 dense_2 (Dense)             (None, 1)                 5

=================================================================
Total params: 37 (148.00 Byte)
Trainable params: 37 (148.00 Byte)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________

Epoch 1/200
4/4 [==============================] - 1s 5ms/step - loss: 0.6918

Epoch 200/200
4/4 [==============================] - 0s 4ms/step - loss: 0.1440
```



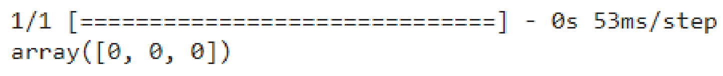
```
1/1 [==============================] - 0s 233ms/step
```



```
1/1 [==============================] - 0s 19ms/step
array([[0.0567151 , 0.9432849 ],
       [0.78821236, 0.21178764],
       [0.05144136, 0.94855864]])
```



```
X=[0.89337759 0.65864154],Predicted_probability=[0.0567151 0.9432849],Predicted_class=[0.0567151]
X=[0.29097707 0.12978982],Predicted_probability=[0.78821236 0.21178764],Predicted_class=[0.78821236]
X=[0.78082614 0.75391697],Predicted_probability=[0.05144136 0.94855864],Predicted_class=[0.05144136]
```



```
1/1 [==============================] - 0s 53ms/step
array([0, 0, 0])
```

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

Code:

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'))

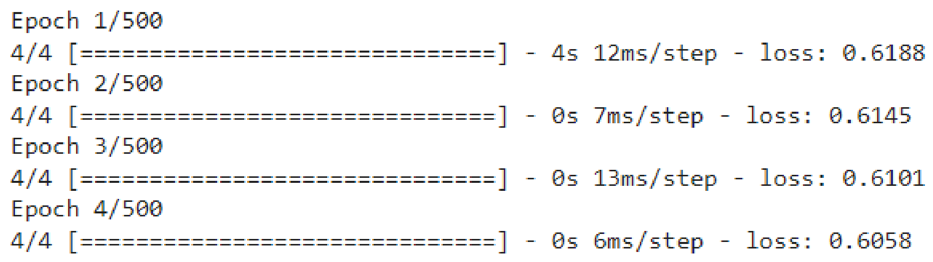
model.add(Dense(1,activation='sigmoid'))

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

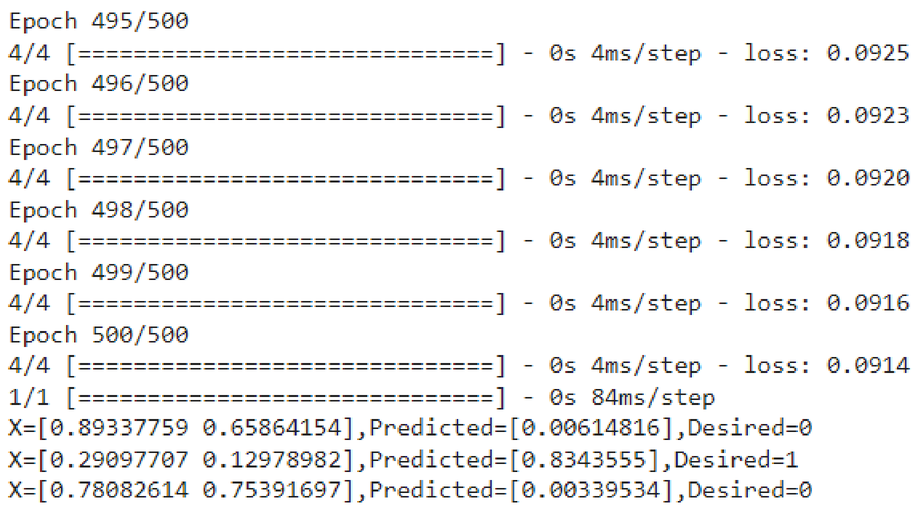
model.fit(X,Y,epochs=500)

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

Xnew=scalar.transform(Xnew)

Ynew=model.predict(Xnew)

for i in range(len(Xnew)):

 print("X=%s,Predicted=%s,Desired=%s"%(Xnew[i],Ynew[i],Yreal[i]))
```

Output:



```
Epoch 1/500
4/4 [==============================] - 4s 12ms/step - loss: 0.6188
Epoch 2/500
4/4 [==============================] - 0s 7ms/step - loss: 0.6145
Epoch 3/500
4/4 [==============================] - 0s 13ms/step - loss: 0.6101
Epoch 4/500
4/4 [==============================] - 0s 6ms/step - loss: 0.6058
```



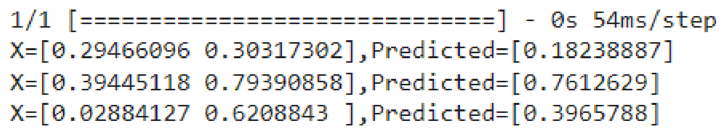
```
Epoch 495/500
4/4 [==============================] - 0s 4ms/step - loss: 0.0925
Epoch 496/500
4/4 [==============================] - 0s 4ms/step - loss: 0.0923
Epoch 497/500
4/4 [==============================] - 0s 4ms/step - loss: 0.0920
Epoch 498/500
4/4 [==============================] - 0s 4ms/step - loss: 0.0918
Epoch 499/500
4/4 [==============================] - 0s 4ms/step - loss: 0.0916
Epoch 500/500
4/4 [==============================] - 0s 4ms/step - loss: 0.0914
1/1 [==============================] - 0s 84ms/step
X=[0.89337759 0.65864154],Predicted=[0.00614816],Desired=0
X=[0.29097707 0.12978982],Predicted=[0.8343555],Desired=1
X=[0.78082614 0.75391697],Predicted=[0.00339534],Desired=0
```

Aim: C] Using a deep feed forward network with two hidden layers for performing linear regression and predicting values.

Code:

```
from keras.models import Sequential

from keras.layers import Dense
from sklearn.datasets import make_regression

from sklearn.preprocessing import MinMaxScaler

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

scalarX,scalarY=MinMaxScaler(),MinMaxScaler()

scalarX.fit(X)

scalarY.fit(Y.reshape(100,1))

X=scalarX.transform(X)

Y=scalarY.transform(Y.reshape(100,1))

model=Sequential()

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

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

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

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

model.fit(X,Y,epochs=1000,verbose=0)

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

Xnew=scalarX.transform(Xnew)

Ynew=model.predict(Xnew)

for i in range(len(Xnew)):

 print("X=%s,Predicted=%s"%(Xnew[i],Ynew[i]))
```

Output:



```
1/1 [==============================] - 0s 54ms/step
X=[0.29466096 0.30317302],Predicted=[0.18238887]
X=[0.39445118 0.79390858],Predicted=[0.7612629]
X=[0.02884127 0.6208843 ],Predicted=[0.3965788]
```

Practical No: 5

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

Code:

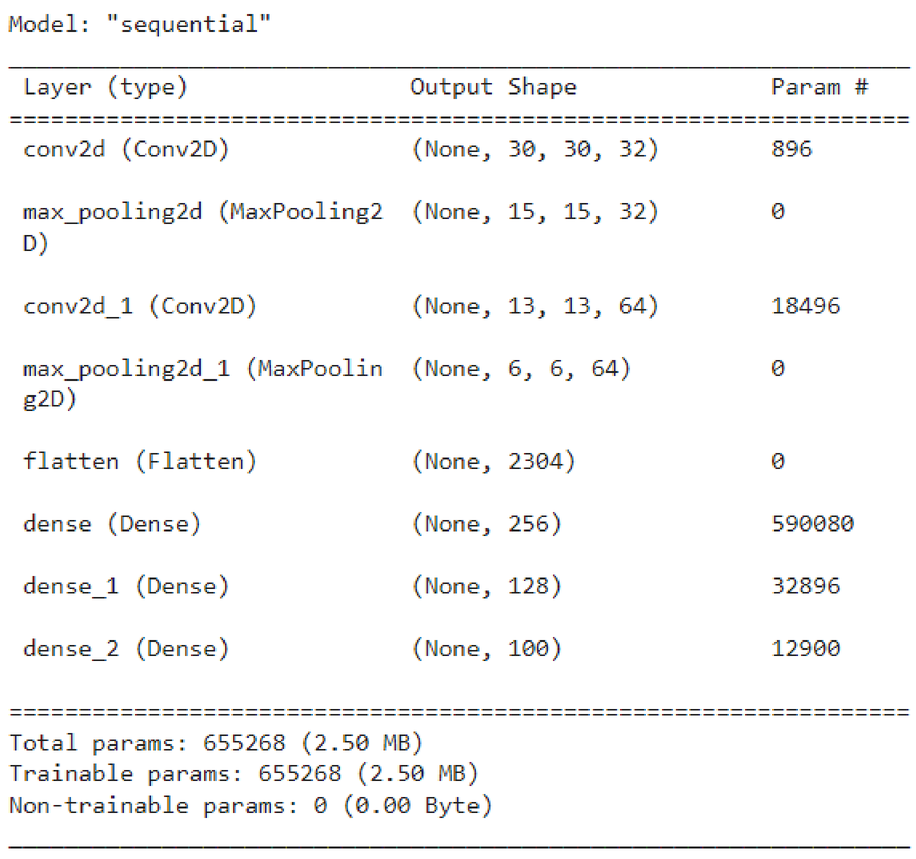
```
from tensorflow.keras.datasets import cifar10

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense, Flatten, Conv2D, MaxPooling2D

from tensorflow.keras.losses import sparse_categorical_crossentropy

from tensorflow.keras.optimizers import Adam

import matplotlib.pyplot as plt

# Model configuration

batch_size = 50

img_width, img_height, img_num_channels = 32, 32, 3

loss_function = sparse_categorical_crossentropy

no_classes = 100

no_epochs = 10   # you can increase it to 20,50,70, 100

optimizer = Adam()

verbosity = 1

# Load CIFAR-10 data

(input_train, target_train), (input_test, target_test) = cifar10.load_data()

# Determine shape of the data

input_shape = (img_width, img_height, img_num_channels)

# Parse numbers as floats

input_train = input_train.astype('float32')
```

```
input_test = input_test.astype('float32')

# Normalize data

input_train = input_train / 255

input_test = input_test / 255

# Create the model

model = Sequential()

model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=input_shape))

model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))

model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Flatten())

model.add(Dense(256, activation='relu'))

model.add(Dense(128, activation='relu'))

model.add(Dense(no_classes, activation='softmax'))

model.summary()

# Compile the model

model.compile(loss=loss_function, optimizer=optimizer,metrics=['accuracy'])

# Fit data to model (this will take little time to train)

history = model.fit(input_train, target_train, batch_size=batch_size, epochs=no_epochs,
verbose=verbosity)

# Generate generalization metrics

score = model.evaluate(input_test, target_test, verbose=0)

print(f'Test loss: {score[0]} / Test accuracy: {score[1]}')

# Visualize history

# Plot history: Loss

plt.plot(history.history['loss'])

plt.title('Validation loss history')
plt.ylabel('Loss value')

plt.xlabel('No. epoch')

plt.show()

# Plot history: Accuracy

plt.plot(history.history['accuracy'])

plt.title('Validation accuracy history')

plt.ylabel('Accuracy value (%)')

plt.xlabel('No. epoch')

plt.show()


# By Adding k fold cross validation

from tensorflow.keras.datasets import cifar10

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense, Flatten, Conv2D, MaxPooling2D

from tensorflow.keras.losses import sparse_categorical_crossentropy

from tensorflow.keras.optimizers import Adam

from sklearn.model_selection import KFold

KFold

import numpy as np

# Model configuration

batch_size = 50

img_width, img_height, img_num_channels = 32, 32, 3

loss_function = sparse_categorical_crossentropy

no_classes = 100

no_epochs = 10

optimizer = Adam()

verbosity = 1
```

```
num_folds = 5

# Load CIFAR-10 data

(input_train, target_train), (input_test, target_test) = cifar10.load_data()

# Determine shape of the data

input_shape = (img_width, img_height, img_num_channels)

# Parse numbers as floats

input_train = input_train.astype('float32')

input_test = input_test.astype('float32')

# Normalize data

input_train = input_train / 255

input_test = input_test / 255

# Define per-fold score containers

acc_per_fold = []

loss_per_fold = []

# Merge inputs and targets

inputs = np.concatenate((input_train, input_test), axis=0)

targets = np.concatenate((target_train, target_test), axis=0)

# Define the K-fold Cross Validator

kfold = KFold(n_splits=num_folds, shuffle=True)


# K-fold Cross Validation model evaluation

fold_no = 1

for train, test in kfold.split(inputs, targets):

  # Define the model architecture

  model = Sequential()

  model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=input_shape))

  model.add(MaxPooling2D(pool_size=(2, 2)))
  model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))

  model.add(MaxPooling2D(pool_size=(2, 2)))

  model.add(Flatten())

  model.add(Dense(256, activation='relu'))

  model.add(Dense(128, activation='relu'))

  model.add(Dense(no_classes, activation='softmax'))

  # Compile the model

  model.compile(loss=loss_function,

                optimizer=optimizer,

                metrics=['accuracy'])

  # Generate a print

  print('------------------------------------------------------------------------')

  print(f'Training for fold {fold_no} ...')

  # Fit data to model

  history = model.fit(inputs[train], targets[train],

              batch_size=batch_size,

              epochs=no_epochs,

              verbose=verbosity)

  # Generate generalization metrics

  scores = model.evaluate(inputs[test], targets[test], verbose=0)

  print(f'Score for fold {fold_no}: {model.metrics_names[0]} of {scores[0]};
{model.metrics_names[1]} of {scores[1]*100}%')

  acc_per_fold.append(scores[1] * 100)

  loss_per_fold.append(scores[0])

  # Increase fold number

  fold_no = fold_no + 1
```

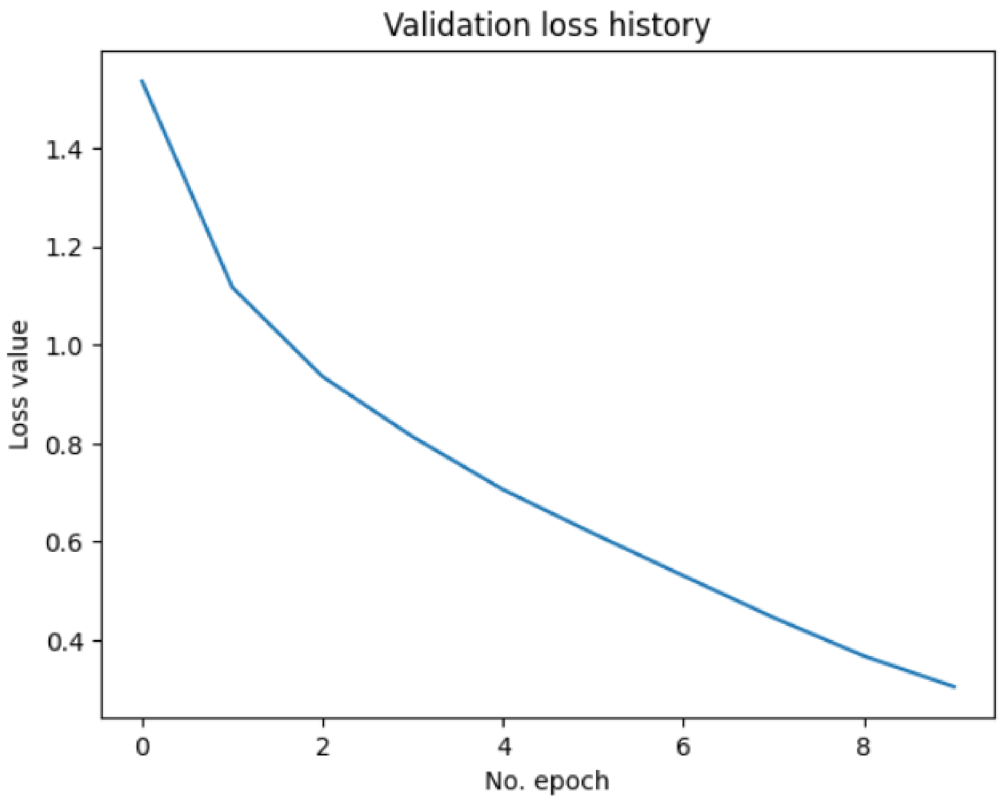
```
# == Provide average scores ==
print('----------------------------------------------------------------------')
print('Score per fold')
for i in range(0, len(acc_per_fold)):
  print('----------------------------------------------------------------------')
  print(f'> Fold {i+1} - Loss: {loss_per_fold[i]} - Accuracy: {acc_per_fold[i]}%')
print('----------------------------------------------------------------------')
print('Average scores for all folds:')
print(f'> Accuracy: {np.mean(acc_per_fold)} (+- {np.std(acc_per_fold)})')
print(f'> Loss: {np.mean(loss_per_fold)}')
print('----------------------------------------------------------------------')
```

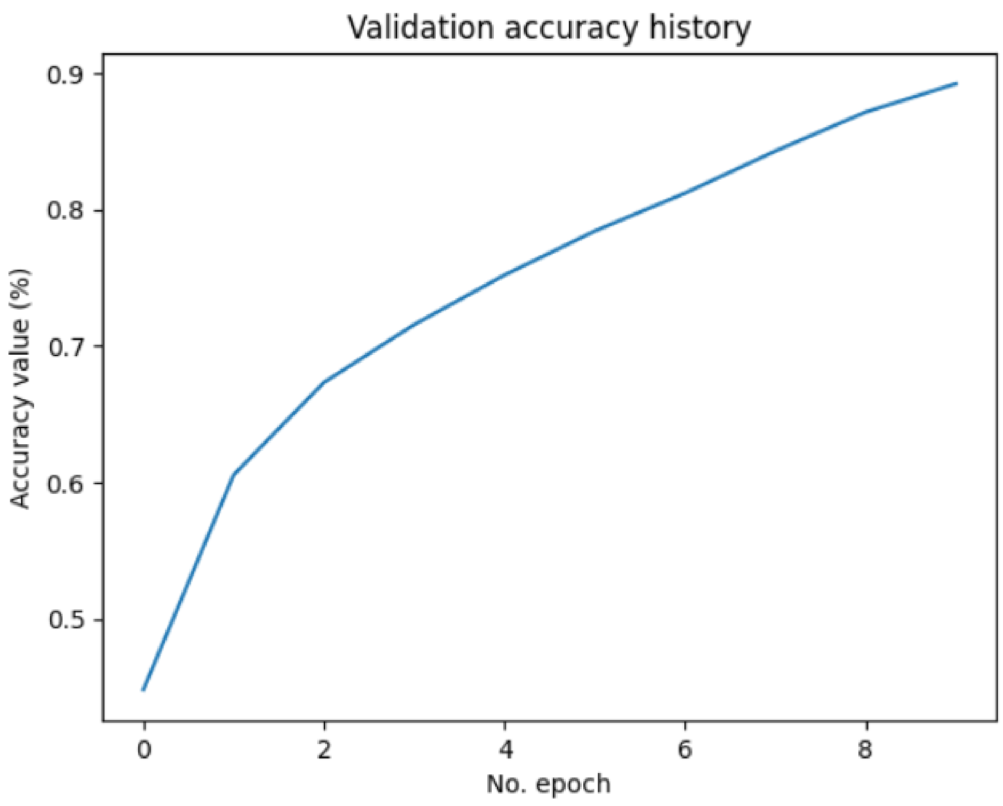
Output:



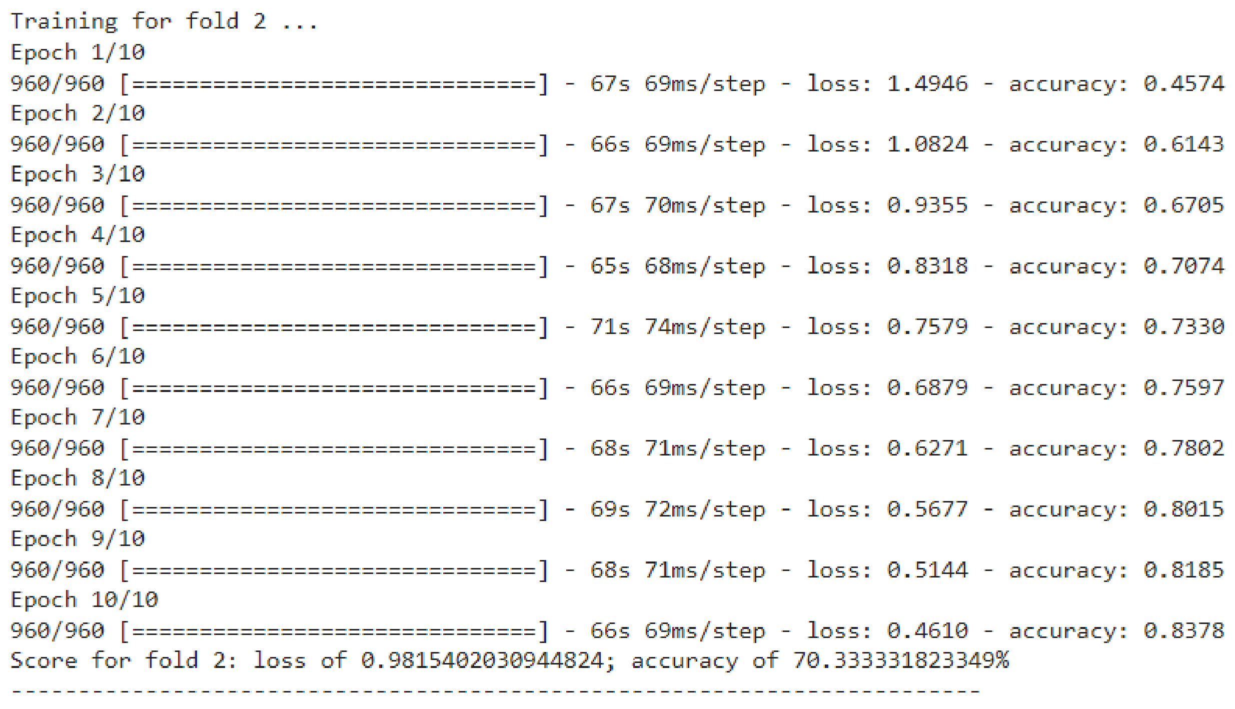
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 30, 30, 32)        896

 max_pooling2d (MaxPooling2  (None, 15, 15, 32)        0
 D)

 conv2d_1 (Conv2D)           (None, 13, 13, 64)        18496

 max_pooling2d_1 (MaxPoolin  (None, 6, 6, 64)          0
 g2D)

 flatten (Flatten)           (None, 2304)              0

 dense (Dense)               (None, 256)               590080

 dense_1 (Dense)             (None, 128)               32896

 dense_2 (Dense)             (None, 100)               12900

=================================================================
Total params: 655268 (2.50 MB)
Trainable params: 655268 (2.50 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

```
Test loss: 1.137102484703064 / Test accuracy: 0.6966999769210815
```

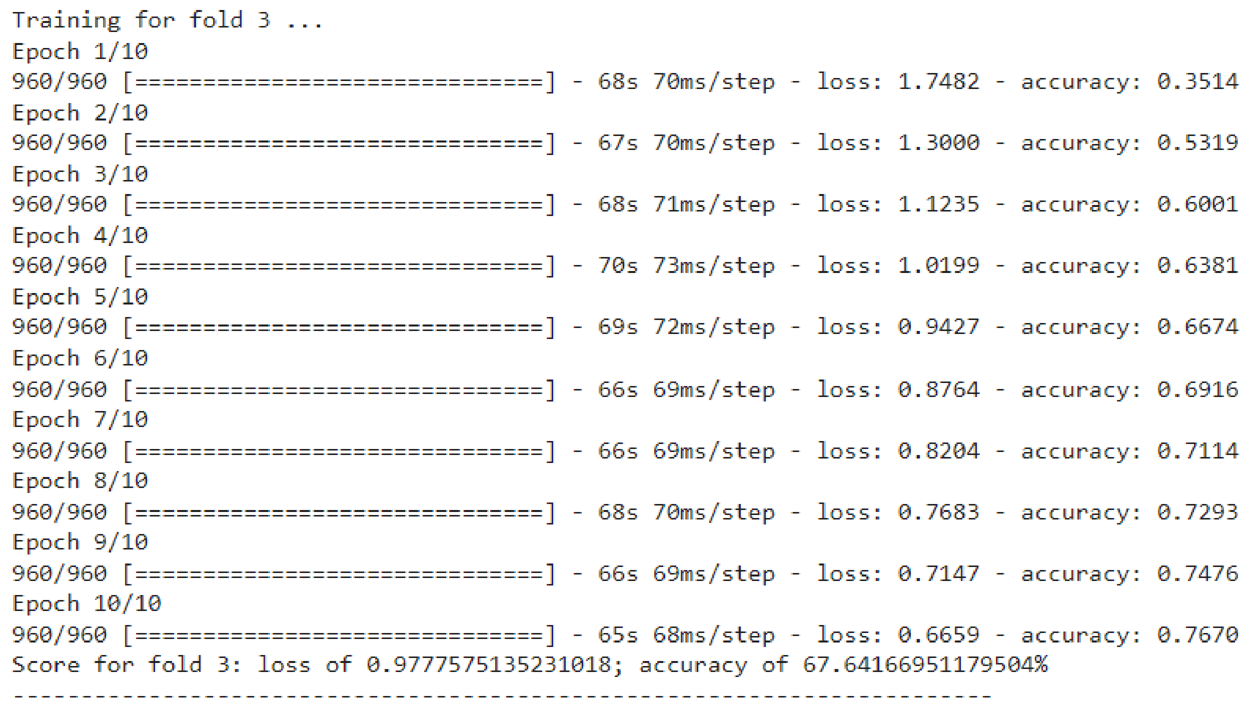




```
Training for fold 1 ...
Epoch 1/10
960/960 [==============================] - 71s 73ms/step - loss: 1.5497 - accuracy: 0.4419
Epoch 2/10
960/960 [==============================] - 71s 74ms/step - loss: 1.1217 - accuracy: 0.6035
Epoch 3/10
960/960 [==============================] - 72s 75ms/step - loss: 0.9531 - accuracy: 0.6672
Epoch 4/10
960/960 [==============================] - 70s 73ms/step - loss: 0.8358 - accuracy: 0.7075
Epoch 5/10
960/960 [==============================] - 69s 72ms/step - loss: 0.7359 - accuracy: 0.7411
Epoch 6/10
960/960 [==============================] - 73s 76ms/step - loss: 0.6461 - accuracy: 0.7727
Epoch 7/10
960/960 [==============================] - 68s 70ms/step - loss: 0.5593 - accuracy: 0.8024
Epoch 8/10
960/960 [==============================] - 69s 72ms/step - loss: 0.4746 - accuracy: 0.8320
Epoch 9/10
960/960 [==============================] - 69s 72ms/step - loss: 0.4027 - accuracy: 0.8556
Epoch 10/10
960/960 [==============================] - 68s 71ms/step - loss: 0.3315 - accuracy: 0.8819
Score for fold 1: loss of 1.1447182893753052; accuracy of 69.01666522026062%
------------------------------------------------------------------------
```

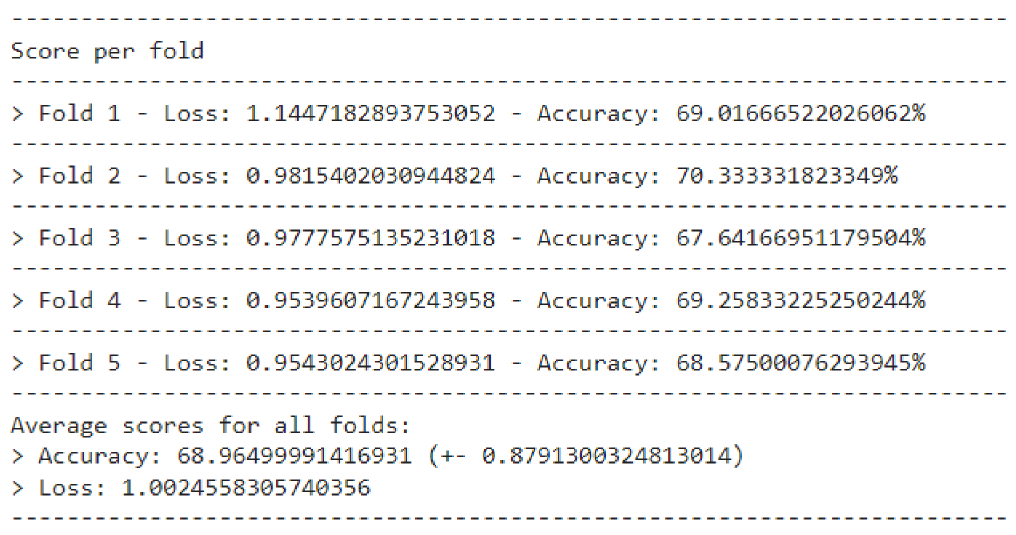


```
Training for fold 2 ...
Epoch 1/10
960/960 [==============================] - 67s 69ms/step - loss: 1.4946 - accuracy: 0.4574
Epoch 2/10
960/960 [==============================] - 66s 69ms/step - loss: 1.0824 - accuracy: 0.6143
Epoch 3/10
960/960 [==============================] - 67s 70ms/step - loss: 0.9355 - accuracy: 0.6705
Epoch 4/10
960/960 [==============================] - 65s 68ms/step - loss: 0.8318 - accuracy: 0.7074
Epoch 5/10
960/960 [==============================] - 71s 74ms/step - loss: 0.7579 - accuracy: 0.7330
Epoch 6/10
960/960 [==============================] - 66s 69ms/step - loss: 0.6879 - accuracy: 0.7597
Epoch 7/10
960/960 [==============================] - 68s 71ms/step - loss: 0.6271 - accuracy: 0.7802
Epoch 8/10
960/960 [==============================] - 69s 72ms/step - loss: 0.5677 - accuracy: 0.8015
Epoch 9/10
960/960 [==============================] - 68s 71ms/step - loss: 0.5144 - accuracy: 0.8185
Epoch 10/10
960/960 [==============================] - 66s 69ms/step - loss: 0.4610 - accuracy: 0.8378
Score for fold 2: loss of 0.9815402030944824; accuracy of 70.333331823349%
------------------------------------------------------------------------
```



```
Training for fold 3 ...
Epoch 1/10
960/960 [==============================] - 68s 70ms/step - loss: 1.7482 - accuracy: 0.3514
Epoch 2/10
960/960 [==============================] - 67s 70ms/step - loss: 1.3000 - accuracy: 0.5319
Epoch 3/10
960/960 [==============================] - 68s 71ms/step - loss: 1.1235 - accuracy: 0.6001
Epoch 4/10
960/960 [==============================] - 70s 73ms/step - loss: 1.0199 - accuracy: 0.6381
Epoch 5/10
960/960 [==============================] - 69s 72ms/step - loss: 0.9427 - accuracy: 0.6674
Epoch 6/10
960/960 [==============================] - 66s 69ms/step - loss: 0.8764 - accuracy: 0.6916
Epoch 7/10
960/960 [==============================] - 66s 69ms/step - loss: 0.8204 - accuracy: 0.7114
Epoch 8/10
960/960 [==============================] - 68s 70ms/step - loss: 0.7683 - accuracy: 0.7293
Epoch 9/10
960/960 [==============================] - 66s 69ms/step - loss: 0.7147 - accuracy: 0.7476
Epoch 10/10
960/960 [==============================] - 65s 68ms/step - loss: 0.6659 - accuracy: 0.7670
Score for fold 3: loss of 0.9777575135231018; accuracy of 67.64166951179504%
------------------------------------------------------------------------
```

```
Training for fold 4 ...
Epoch 1/10
960/960 [==============================] - 66s 68ms/step - loss: 1.5605 - accuracy: 0.4300
Epoch 2/10
960/960 [==============================] - 67s 69ms/step - loss: 1.1548 - accuracy: 0.5909
Epoch 3/10
960/960 [==============================] - 72s 75ms/step - loss: 1.0053 - accuracy: 0.6465
Epoch 4/10
960/960 [==============================] - 68s 71ms/step - loss: 0.9051 - accuracy: 0.6813
Epoch 5/10
960/960 [==============================] - 67s 70ms/step - loss: 0.8169 - accuracy: 0.7136
Epoch 6/10
960/960 [==============================] - 69s 71ms/step - loss: 0.7510 - accuracy: 0.7357
Epoch 7/10
960/960 [==============================] - 66s 69ms/step - loss: 0.6871 - accuracy: 0.7579
Epoch 8/10
960/960 [==============================] - 68s 71ms/step - loss: 0.6304 - accuracy: 0.7791
Epoch 9/10
960/960 [==============================] - 67s 70ms/step - loss: 0.5766 - accuracy: 0.7966
Epoch 10/10
960/960 [==============================] - 68s 71ms/step - loss: 0.5261 - accuracy: 0.8164
Score for fold 4: loss of 0.9539607167243958; accuracy of 69.25833225250244%
------------------------------------------------------------------------

Training for fold 5 ...
Epoch 1/10
960/960 [==============================] - 67s 69ms/step - loss: 1.5813 - accuracy: 0.4209
Epoch 2/10
960/960 [==============================] - 73s 76ms/step - loss: 1.2035 - accuracy: 0.5685
Epoch 3/10
960/960 [==============================] - 68s 70ms/step - loss: 1.0382 - accuracy: 0.6306
Epoch 4/10
960/960 [==============================] - 68s 71ms/step - loss: 0.9473 - accuracy: 0.6646
Epoch 5/10
960/960 [==============================] - 66s 68ms/step - loss: 0.8742 - accuracy: 0.6905
Epoch 6/10
960/960 [==============================] - 66s 69ms/step - loss: 0.8167 - accuracy: 0.7120
Epoch 7/10
960/960 [==============================] - 69s 71ms/step - loss: 0.7613 - accuracy: 0.7337
Epoch 8/10
960/960 [==============================] - 66s 69ms/step - loss: 0.7087 - accuracy: 0.7507
Epoch 9/10
960/960 [==============================] - 68s 71ms/step - loss: 0.6611 - accuracy: 0.7652
Epoch 10/10
960/960 [==============================] - 67s 70ms/step - loss: 0.6132 - accuracy: 0.7837
Score for fold 5: loss of 0.9543024301528931; accuracy of 68.57500076293945%
```



```
------------------------------------------------------------------------
Score per fold
------------------------------------------------------------------------
> Fold 1 - Loss: 1.1447182893753052 - Accuracy: 69.01666522026062%
------------------------------------------------------------------------
> Fold 2 - Loss: 0.9815402030944824 - Accuracy: 70.333331823349%
------------------------------------------------------------------------
> Fold 3 - Loss: 0.9777575135231018 - Accuracy: 67.64166951179504%
------------------------------------------------------------------------
> Fold 4 - Loss: 0.9539607167243958 - Accuracy: 69.25833225250244%
------------------------------------------------------------------------
> Fold 5 - Loss: 0.9543024301528931 - Accuracy: 68.57500076293945%
------------------------------------------------------------------------
Average scores for all folds:
> Accuracy: 68.96499991416931 (+- 0.8791300324813014)
> Loss: 1.0024558305740356
------------------------------------------------------------------------
```

Practical No: 6

Aim: Implementing regularization to avoid overfitting in binary classification.

Code:

```
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.shape)

print(trainY.shape)

print(testX.shape)

print(testY.shape)


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'])

model.summary()

history=model.fit(trainX,trainY,validation_data=(testX,testY),epochs=100)


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

pyplot.legend()

pyplot.show()


from keras.regularizers import l2

model=Sequential()

model.add(Dense(500,input_dim=2,activation='relu',kernel_regularizer=l2(0.001)))

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

model.summary()


model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])

history=model.fit(trainX,trainY,validation_data=(testX,testY),epochs=100)


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

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

pyplot.legend()

pyplot.show()


from keras.regularizers import l1_l2

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'])

model.summary()

history=model.fit(trainX,trainY,validation_data=(testX,testY),epochs=100)
```

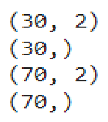
pyplot.plot(history.history['accuracy'],label='train')

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

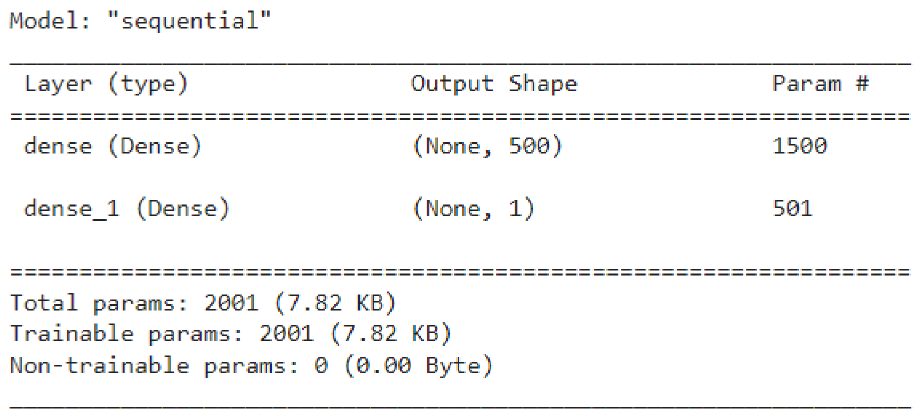
pyplot.legend()

pyplot.show()

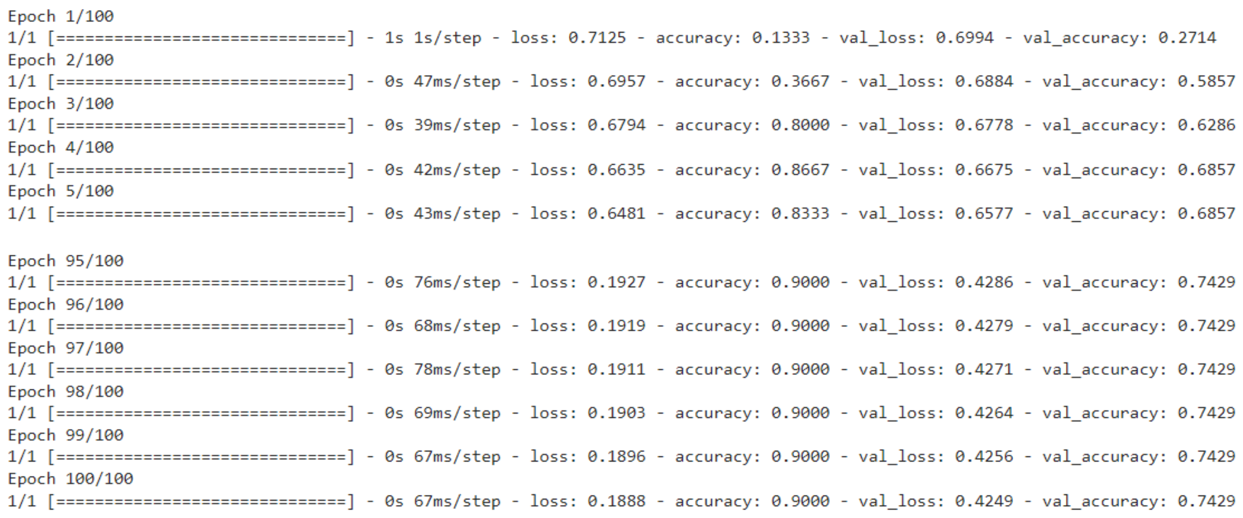
Output:



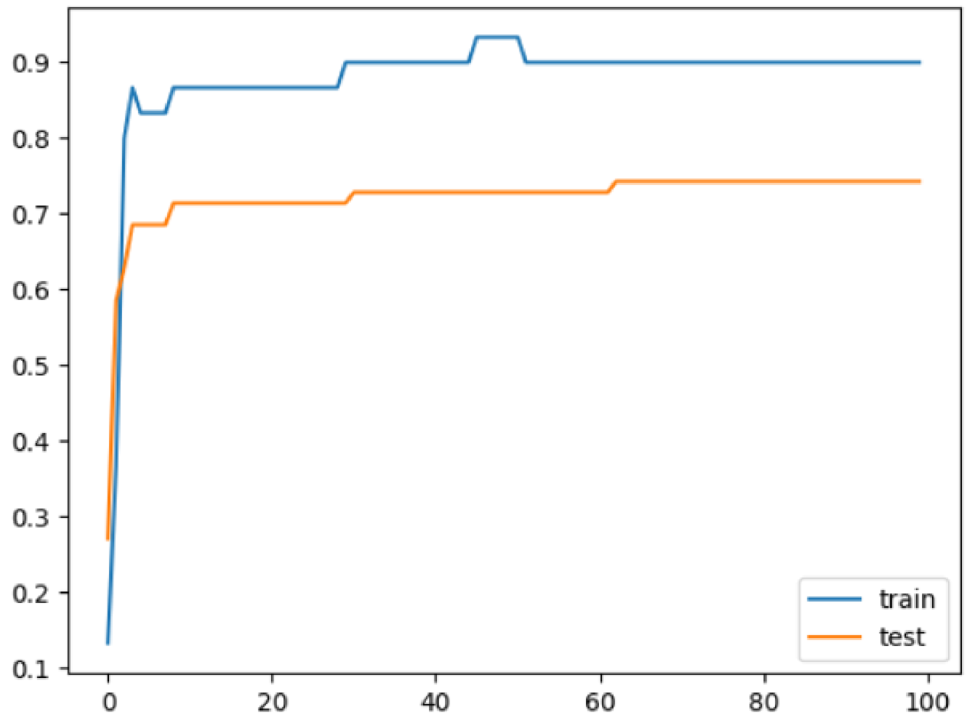
```
(30, 2)
(30,)
(70, 2)
(70,)
```



```
 Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 500)               1500

 dense_1 (Dense)             (None, 1)                 501

=================================================================
Total params: 2001 (7.82 KB)
Trainable params: 2001 (7.82 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```



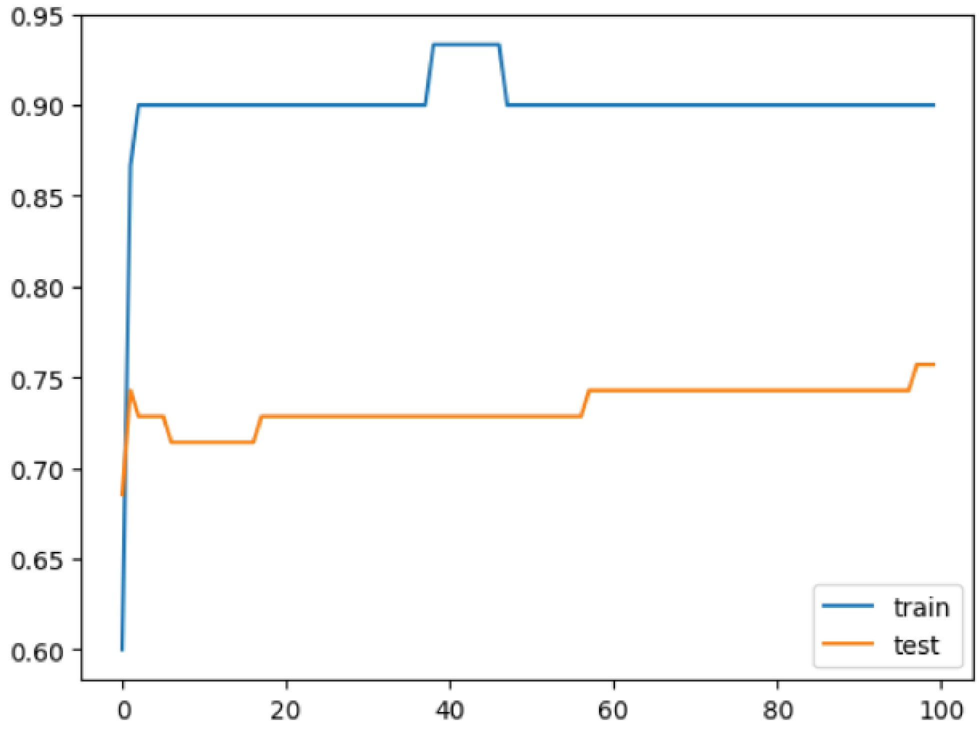
```
Epoch 1/100
1/1 [==============================] - 1s 1s/step - loss: 0.7125 - accuracy: 0.1333 - val_loss: 0.6994 - val_accuracy: 0.2714
Epoch 2/100
1/1 [==============================] - 0s 47ms/step - loss: 0.6957 - accuracy: 0.3667 - val_loss: 0.6884 - val_accuracy: 0.5857
Epoch 3/100
1/1 [==============================] - 0s 39ms/step - loss: 0.6794 - accuracy: 0.8000 - val_loss: 0.6778 - val_accuracy: 0.6286
Epoch 4/100
1/1 [==============================] - 0s 42ms/step - loss: 0.6635 - accuracy: 0.8667 - val_loss: 0.6675 - val_accuracy: 0.6857
Epoch 5/100
1/1 [==============================] - 0s 43ms/step - loss: 0.6481 - accuracy: 0.8333 - val_loss: 0.6577 - val_accuracy: 0.6857

Epoch 95/100
1/1 [==============================] - 0s 76ms/step - loss: 0.1927 - accuracy: 0.9000 - val_loss: 0.4286 - val_accuracy: 0.7429
Epoch 96/100
1/1 [==============================] - 0s 68ms/step - loss: 0.1919 - accuracy: 0.9000 - val_loss: 0.4279 - val_accuracy: 0.7429
Epoch 97/100
1/1 [==============================] - 0s 78ms/step - loss: 0.1911 - accuracy: 0.9000 - val_loss: 0.4271 - val_accuracy: 0.7429
Epoch 98/100
1/1 [==============================] - 0s 69ms/step - loss: 0.1903 - accuracy: 0.9000 - val_loss: 0.4264 - val_accuracy: 0.7429
Epoch 99/100
1/1 [==============================] - 0s 67ms/step - loss: 0.1896 - accuracy: 0.9000 - val_loss: 0.4256 - val_accuracy: 0.7429
Epoch 100/100
1/1 [==============================] - 0s 67ms/step - loss: 0.1888 - accuracy: 0.9000 - val_loss: 0.4249 - val_accuracy: 0.7429
```



```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_2 (Dense)             (None, 500)               1500

 dense_3 (Dense)             (None, 1)                 501

=================================================================
Total params: 2001 (7.82 KB)
Trainable params: 2001 (7.82 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

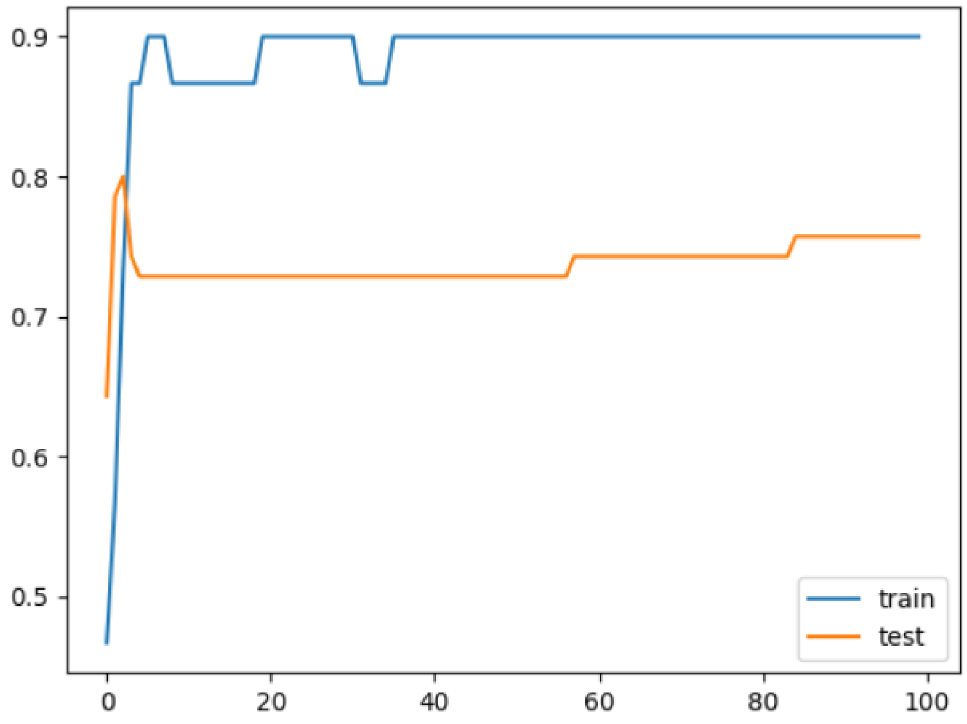
```
Epoch 1/100
1/1 [==============================] - 1s 1s/step - loss: 0.6955 - accuracy: 0.6000 - val_loss: 0.6837 - val_accuracy: 0.6857
Epoch 2/100
1/1 [==============================] - 0s 43ms/step - loss: 0.6789 - accuracy: 0.8667 - val_loss: 0.6729 - val_accuracy: 0.7429
Epoch 3/100
1/1 [==============================] - 0s 47ms/step - loss: 0.6627 - accuracy: 0.9000 - val_loss: 0.6625 - val_accuracy: 0.7286
Epoch 4/100
1/1 [==============================] - 0s 57ms/step - loss: 0.6470 - accuracy: 0.9000 - val_loss: 0.6525 - val_accuracy: 0.7286
Epoch 5/100
1/1 [==============================] - 0s 55ms/step - loss: 0.6317 - accuracy: 0.9000 - val_loss: 0.6429 - val_accuracy: 0.7286
```

```
Epoch 95/100
1/1 [==============================] - 0s 43ms/step - loss: 0.1986 - accuracy: 0.9000 - val_loss: 0.4315 - val_accuracy: 0.7429
Epoch 96/100
1/1 [==============================] - 0s 42ms/step - loss: 0.1979 - accuracy: 0.9000 - val_loss: 0.4309 - val_accuracy: 0.7429
Epoch 97/100
1/1 [==============================] - 0s 56ms/step - loss: 0.1972 - accuracy: 0.9000 - val_loss: 0.4303 - val_accuracy: 0.7429
Epoch 98/100
1/1 [==============================] - 0s 58ms/step - loss: 0.1965 - accuracy: 0.9000 - val_loss: 0.4297 - val_accuracy: 0.7571
Epoch 99/100
1/1 [==============================] - 0s 57ms/step - loss: 0.1958 - accuracy: 0.9000 - val_loss: 0.4290 - val_accuracy: 0.7571
Epoch 100/100
1/1 [==============================] - 0s 56ms/step - loss: 0.1951 - accuracy: 0.9000 - val_loss: 0.4284 - val_accuracy: 0.7571
```



```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_4 (Dense)             (None, 500)               1500

 dense_5 (Dense)             (None, 1)                 501

=================================================================
Total params: 2001 (7.82 KB)
Trainable params: 2001 (7.82 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

```
Epoch 1/100
1/1 [==============================] - 1s 1s/step - loss: 0.7606 - accuracy: 0.4667 - val_loss: 0.7435 - val_accuracy: 0.6429
Epoch 2/100
1/1 [==============================] - 0s 58ms/step - loss: 0.7445 - accuracy: 0.5667 - val_loss: 0.7329 - val_accuracy: 0.7857
Epoch 3/100
1/1 [==============================] - 0s 57ms/step - loss: 0.7288 - accuracy: 0.7333 - val_loss: 0.7225 - val_accuracy: 0.8000
Epoch 4/100
1/1 [==============================] - 0s 58ms/step - loss: 0.7135 - accuracy: 0.8667 - val_loss: 0.7125 - val_accuracy: 0.7429
Epoch 5/100
1/1 [==============================] - 0s 42ms/step - loss: 0.6985 - accuracy: 0.8667 - val_loss: 0.7029 - val_accuracy: 0.7286

Epoch 95/100
1/1 [==============================] - 0s 153ms/step - loss: 0.2591 - accuracy: 0.9000 - val_loss: 0.4713 - val_accuracy: 0.7571
Epoch 96/100
1/1 [==============================] - 0s 136ms/step - loss: 0.2582 - accuracy: 0.9000 - val_loss: 0.4705 - val_accuracy: 0.7571
Epoch 97/100
1/1 [==============================] - 0s 91ms/step - loss: 0.2574 - accuracy: 0.9000 - val_loss: 0.4698 - val_accuracy: 0.7571
Epoch 98/100
1/1 [==============================] - 0s 40ms/step - loss: 0.2566 - accuracy: 0.9000 - val_loss: 0.4690 - val_accuracy: 0.7571
Epoch 99/100
1/1 [==============================] - 0s 43ms/step - loss: 0.2558 - accuracy: 0.9000 - val_loss: 0.4682 - val_accuracy: 0.7571
Epoch 100/100
1/1 [==============================] - 0s 43ms/step - loss: 0.2550 - accuracy: 0.9000 - val_loss: 0.4675 - val_accuracy: 0.7571
```



Practical No: 7

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

Code:

```
import numpy as np

import tensorflow_datasets as tfds

import tensorflow as tf

tfds.disable_progress_bar()

import matplotlib.pyplot as plt

def plot_graphs(history, metric):

  plt.plot(history.history[metric])

  plt.plot(history.history['val_'+metric], '')

  plt.xlabel("Epochs")

  plt.ylabel(metric)

  plt.legend([metric, 'val_'+metric])

dataset, info = tfds.load('imdb_reviews', with_info=True,

                          as_supervised=True)

train_dataset, test_dataset = dataset['train'], dataset['test']

train_dataset.element_spec


for example, label in train_dataset.take(5):

  print('text: ', example.numpy())

  print('label: ', label.numpy())


BUFFER_SIZE = 10000
```

```
BATCH_SIZE = 64

train_dataset =
train_dataset.shuffle(BUFFER_SIZE).batch(BATCH_SIZE).prefetch(tf.data.AUTOTUNE)

test_dataset = test_dataset.batch(BATCH_SIZE).prefetch(tf.data.AUTOTUNE)


for example, label in train_dataset.take(1):

  print('texts: ', example.numpy()[:3])

  print()

  print('labels: ', label.numpy()[:3])


VOCAB_SIZE = 1000

encoder = tf.keras.layers.TextVectorization(max_tokens=VOCAB_SIZE)

encoder.adapt(train_dataset.map(lambda text, label: text))


vocab = np.array(encoder.get_vocabulary())

vocab[:20]


encoded_example = encoder(example)[:3].numpy()

encoded_example


for n in range(3):

  print("Original: ", example[n].numpy())

  print("Round-trip: ", " ".join(vocab[encoded_example[n]]))

  print()


model = tf.keras.Sequential([

    encoder,
    tf.keras.layers.Embedding(
        input_dim=len(encoder.get_vocabulary()),
        output_dim=64,
        # Use masking to handle the variable sequence lengths
        mask_zero=True),
    tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(64)),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(1)
])
print([layer.supports_masking for layer in model.layers])

# predict on a sample text without padding.
sample_text = ('The movie was cool. The animation and the graphics '
               'were out of this world. I would recommend this movie.')
predictions = model.predict(np.array([sample_text]))
print(predictions[0])

# predict on a sample text with padding
padding = "the " * 2000
predictions = model.predict(np.array([sample_text, padding]))
print(predictions[0])

model.compile(loss=tf.keras.losses.BinaryCrossentropy(from_logits=True),
              optimizer=tf.keras.optimizers.Adam(1e-4),
              metrics=['accuracy'])
history = model.fit(train_dataset, epochs=10,
                    validation_data=test_dataset,
                    validation_steps=30)

test_loss, test_acc = model.evaluate(test_dataset)

print('Test Loss:', test_loss)
print('Test Accuracy:', test_acc)

plt.figure(figsize=(16, 8))
plt.subplot(1, 2, 1)
plot_graphs(history, 'accuracy')
plt.ylim(None, 1)
plt.subplot(1, 2, 2)
plot_graphs(history, 'loss')
plt.ylim(0, None)

sample_text = ('The movie was cool. The animation and the graphics '
               'were out of this world. I would recommend this movie.')
predictions = model.predict(np.array([sample_text]))
predictions

model = tf.keras.Sequential([
    encoder,
    tf.keras.layers.Embedding(len(encoder.get_vocabulary()), 64, mask_zero=True),
    tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(64,  return_sequences=True)),
    tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(32)),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dropout(0.5),
    tf.keras.layers.Dense(1)
])
model.compile(loss=tf.keras.losses.BinaryCrossentropy(from_logits=True),
              optimizer=tf.keras.optimizers.Adam(1e-4),
              metrics=['accuracy'])
history = model.fit(train_dataset, epochs=10,
                    validation_data=test_dataset,
                    validation_steps=30)

test_loss, test_acc = model.evaluate(test_dataset)
print('Test Loss:', test_loss)
print('Test Accuracy:', test_acc)

# predict on a sample text without padding.
sample_text = ('The movie was not good. The animation and the graphics '
               'were terrible. I would not recommend this movie.')
predictions = model.predict(np.array([sample_text]))
print(predictions)

plt.figure(figsize=(16, 6))
plt.subplot(1, 2, 1)
plot_graphs(history, 'accuracy')
plt.subplot(1, 2, 2)
plot_graphs(history, 'loss')
```

Output:

```
text:  b"This was an absolutely terrible movie. Don't be lured in by Christopher Walken or Michael Ironside. Both are great actors, but this must simply be their worst role in history. Even their great act
label:  0
text:  b'I have been known to fall asleep during films, but this is usually due to a combination of things including, really tired, being warm and comfortable on the sette and having just eaten a lot. Howe
label:  0
text:  b'Mann photographs the Alberta Rocky Mountains in a superb fashion, and Jimmy Stewart and Walter Brennan give enjoyable performances as they always seem to do. <br /><br />But come on Hollywood - a
label:  0
text:  b'This is the kind of film for a snowy Sunday afternoon when the rest of the world can go ahead with its own business as you descend into a big arm-chair and mellow for a couple of hours. Wonderful
label:  1
text:  b'As others have mentioned, all the women that go nude in this film are mostly absolutely gorgeous. The plot very ably shows the hypocrisy of the female libido. When men are around they want to be p
label:  1
```

```
array([[ 10,   1, 442, ...,   0,   0,   0],
       [  1,  16,  49, ...,   0,   0,   0],
       [  4, 220,  12, ...,   0,   0,   0]])
```

```
array(['', '[UNK]', 'the', 'and', 'a', 'of', 'to', 'is', 'in', 'it', 'i',
       'this', 'that', 'br', 'was', 'as', 'for', 'with', 'movie', 'but'],
      dtype='<U14')
```

```
Original:  b'I voted 3 for this movie because it looks great as does all of Greenaways output. However it was his usual mix of "art" sex and pretentious crap.I know lots of people like this film but I grew
Round-trip:  i [UNK] 3 for this movie because it looks great as does all of [UNK] [UNK] however it was his usual [UNK] of art sex and [UNK] [UNK] know lots of people like this film but i [UNK] [UNK] of it

Original:  b'Justifications for what happened to his movie in terms of distributors and secondary directors, drunks and receptionists doing script rewrites aside, let\'s just take this movie as it\'s offer
Round-trip:  [UNK] for what happened to his movie in [UNK] of [UNK] and [UNK] directors [UNK] and [UNK] doing script [UNK] [UNK] lets just take this movie as its [UNK] without [UNK] [UNK] br this movie is

Original:  b'A comedy that worked surprisingly well was the little British effort "The Divorce Of Lady X (1938)" . It marks the first pairing of Laurence Olivier and Merle Oberon, before that little film a
Round-trip:  a comedy that worked [UNK] well was the little british effort the [UNK] of lady [UNK] [UNK] it [UNK] the first [UNK] of [UNK] [UNK] and [UNK] [UNK] before that little film about [UNK] [UNK] or
```

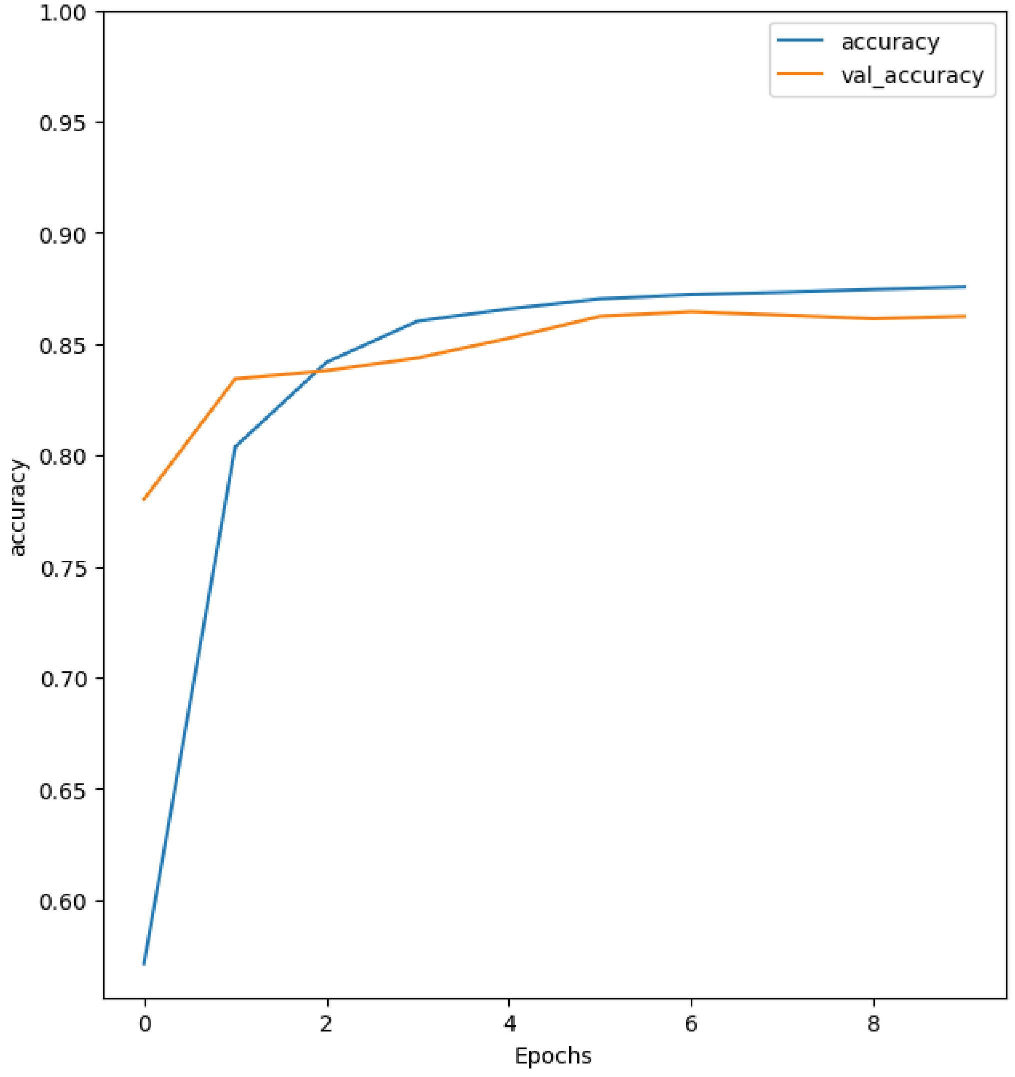
```
[False, True, True, True, True]
```

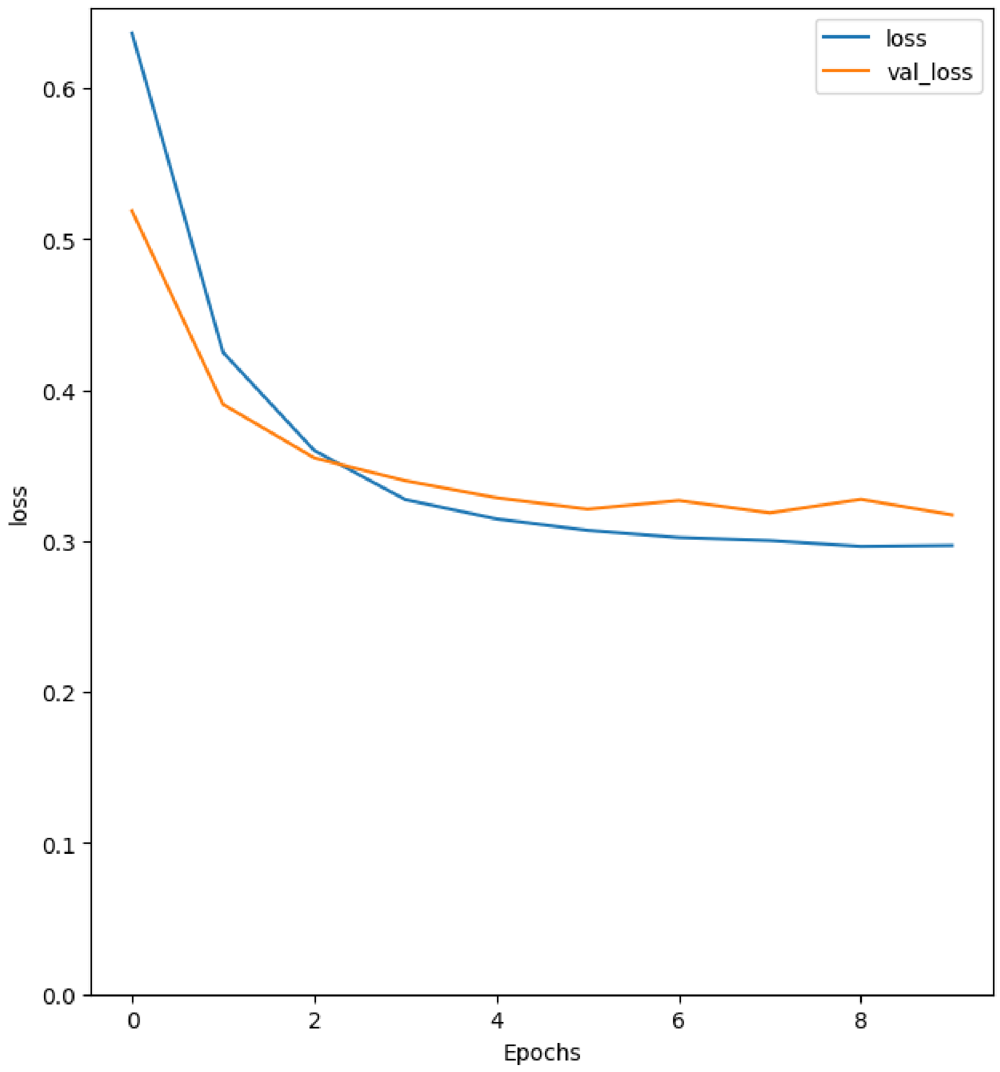
```
1/1 [==============================] - 4s 4s/step
[-0.01484437]
```

```
Epoch 1/10
391/391 [==============================] - 50s 103ms/step - loss: 0.6363 - accuracy: 0.5712 - val_loss: 0.5186 - val_accuracy: 0.7802
Epoch 2/10
391/391 [==============================] - 26s 67ms/step - loss: 0.4250 - accuracy: 0.8037 - val_loss: 0.3905 - val_accuracy: 0.8344
Epoch 3/10
391/391 [==============================] - 27s 69ms/step - loss: 0.3600 - accuracy: 0.8418 - val_loss: 0.3551 - val_accuracy: 0.8380
Epoch 4/10
391/391 [==============================] - 26s 67ms/step - loss: 0.3275 - accuracy: 0.8604 - val_loss: 0.3400 - val_accuracy: 0.8438
Epoch 5/10
391/391 [==============================] - 25s 65ms/step - loss: 0.3147 - accuracy: 0.8658 - val_loss: 0.3287 - val_accuracy: 0.8526
Epoch 6/10
391/391 [==============================] - 26s 66ms/step - loss: 0.3071 - accuracy: 0.8703 - val_loss: 0.3212 - val_accuracy: 0.8625
Epoch 7/10
391/391 [==============================] - 25s 64ms/step - loss: 0.3024 - accuracy: 0.8722 - val_loss: 0.3269 - val_accuracy: 0.8646
Epoch 8/10
391/391 [==============================] - 25s 64ms/step - loss: 0.3003 - accuracy: 0.8733 - val_loss: 0.3187 - val_accuracy: 0.8630
Epoch 9/10
391/391 [==============================] - 25s 64ms/step - loss: 0.2965 - accuracy: 0.8746 - val_loss: 0.3277 - val_accuracy: 0.8615
Epoch 10/10
391/391 [==============================] - 26s 65ms/step - loss: 0.2971 - accuracy: 0.8757 - val_loss: 0.3174 - val_accuracy: 0.8625
```

```
391/391 [==============================] - 18s 45ms/step - loss: 0.3137 - accuracy: 0.8618
Test Loss: 0.3137068450450897
Test Accuracy: 0.8617600202560425
```

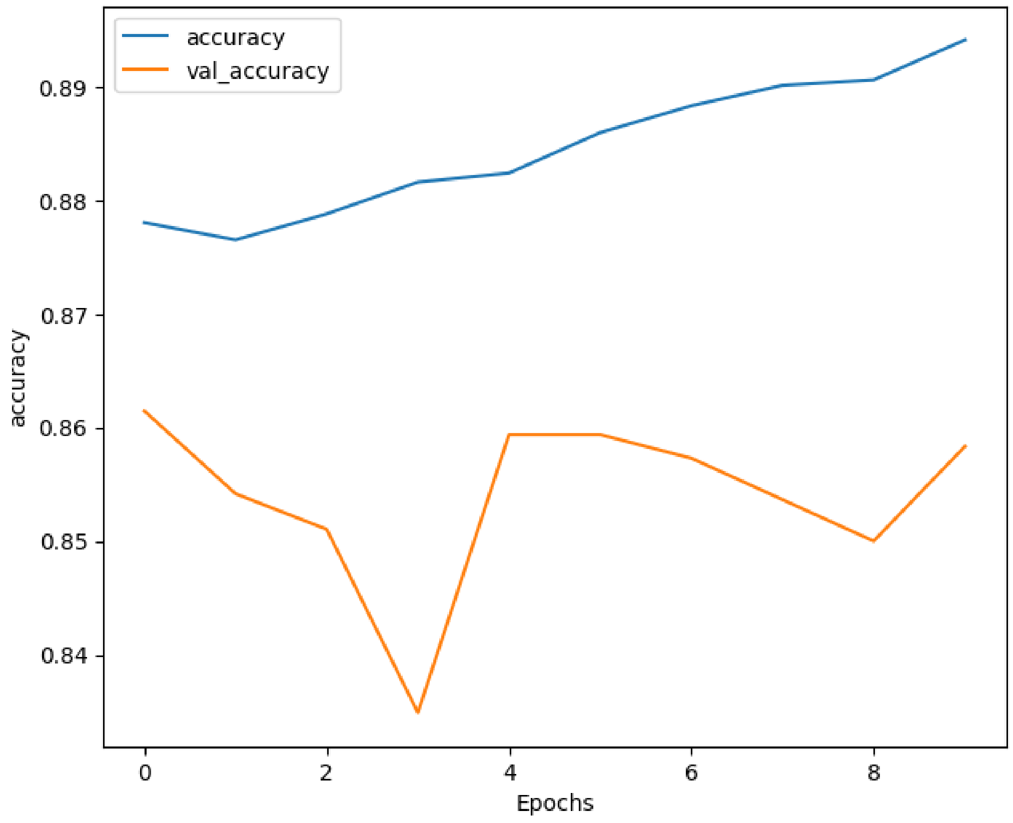
```
(0.0, 0.6532905504107476)
```

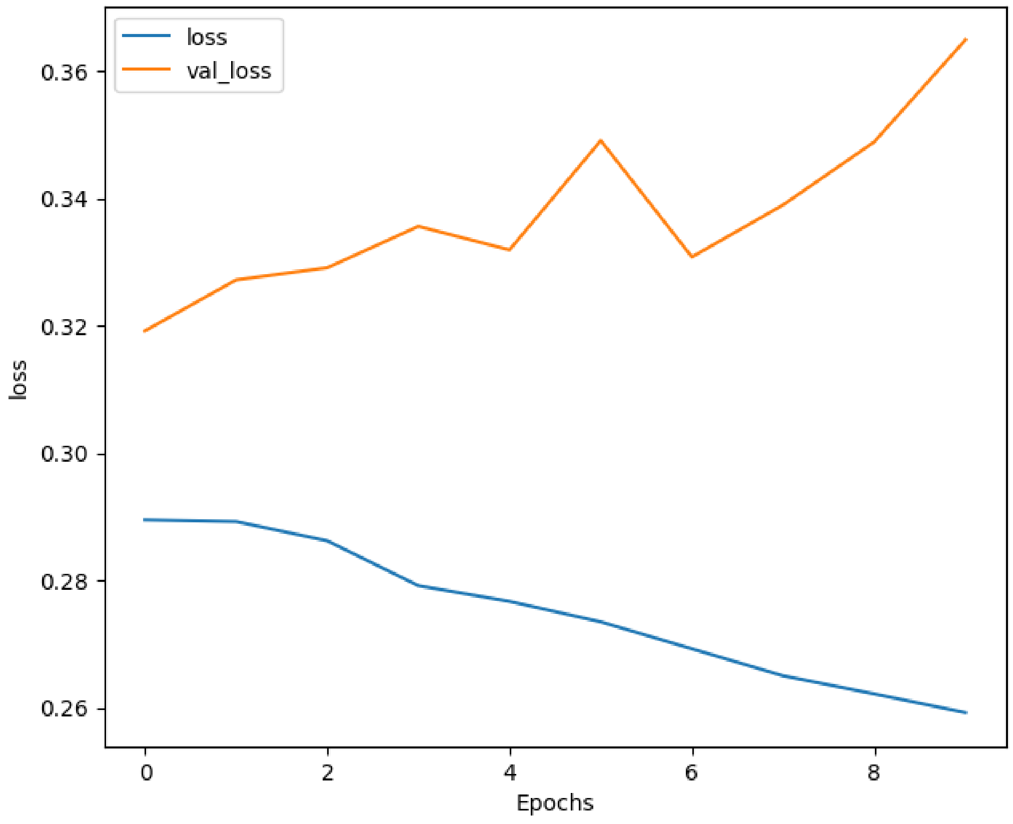




```
Epoch 1/10
391/391 [==============================] - 56s 142ms/step - loss: 0.2895 - accuracy: 0.8780 - val_loss: 0.3192 - val_accuracy: 0.8615
Epoch 2/10
391/391 [==============================] - 48s 123ms/step - loss: 0.2892 - accuracy: 0.8765 - val_loss: 0.3272 - val_accuracy: 0.8542
Epoch 3/10
391/391 [==============================] - 47s 120ms/step - loss: 0.2862 - accuracy: 0.8788 - val_loss: 0.3291 - val_accuracy: 0.8510
Epoch 4/10
391/391 [==============================] - 47s 119ms/step - loss: 0.2792 - accuracy: 0.8816 - val_loss: 0.3356 - val_accuracy: 0.8349
Epoch 5/10
391/391 [==============================] - 46s 117ms/step - loss: 0.2767 - accuracy: 0.8824 - val_loss: 0.3319 - val_accuracy: 0.8594
Epoch 6/10
391/391 [==============================] - 48s 121ms/step - loss: 0.2735 - accuracy: 0.8860 - val_loss: 0.3491 - val_accuracy: 0.8594
Epoch 7/10
391/391 [==============================] - 46s 117ms/step - loss: 0.2692 - accuracy: 0.8883 - val_loss: 0.3308 - val_accuracy: 0.8573
Epoch 8/10
391/391 [==============================] - 45s 115ms/step - loss: 0.2650 - accuracy: 0.8901 - val_loss: 0.3390 - val_accuracy: 0.8536
Epoch 9/10
391/391 [==============================] - 47s 120ms/step - loss: 0.2622 - accuracy: 0.8906 - val_loss: 0.3489 - val_accuracy: 0.8500
Epoch 10/10
391/391 [==============================] - 47s 119ms/step - loss: 0.2592 - accuracy: 0.8941 - val_loss: 0.3649 - val_accuracy: 0.8583
```

```
391/391 [==============================] - 20s 51ms/step - loss: 0.3604 - accuracy: 0.8529
Test Loss: 0.3603912889957428
Test Accuracy: 0.8529199957847595
```


accuracy
val_accuracy
0.89
0.88
0.87
0.86
0.85
0.84
accuracy
0
2
4
6
8
Epochs


loss
val_loss
0.36
0.34
0.32
0.30
0.28
0.26
loss
0
2
4
6
8
Epochs

Practical No: 8

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

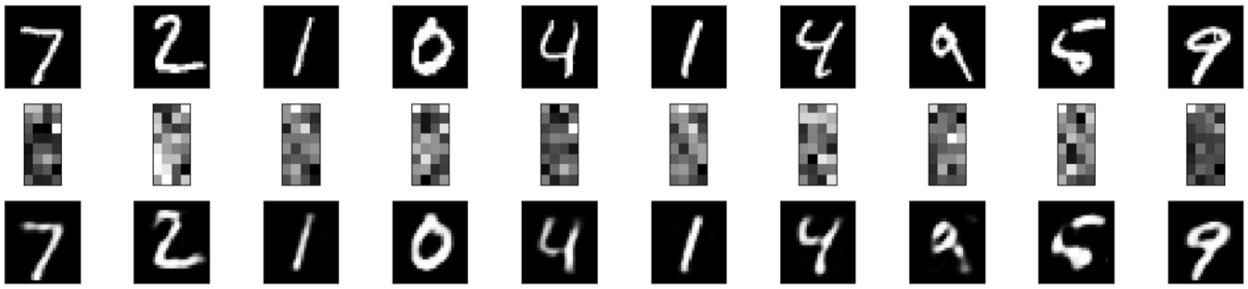
Code:

```
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:

```
Epoch 1/50
235/235 [==============================] - 4s 12ms/step - loss: 0.2743 - val_loss: 0.1868
Epoch 2/50
235/235 [==============================] - 4s 16ms/step - loss: 0.1695 - val_loss: 0.1527
Epoch 3/50
235/235 [==============================] - 3s 11ms/step - loss: 0.1440 - val_loss: 0.1334
Epoch 4/50
235/235 [==============================] - 3s 12ms/step - loss: 0.1283 - val_loss: 0.1212
Epoch 5/50
235/235 [==============================] - 3s 11ms/step - loss: 0.1180 - val_loss: 0.1127

Epoch 45/50
235/235 [==============================] - 3s 12ms/step - loss: 0.0927 - val_loss: 0.0917
Epoch 46/50
235/235 [==============================] - 3s 12ms/step - loss: 0.0927 - val_loss: 0.0915
Epoch 47/50
235/235 [==============================] - 4s 16ms/step - loss: 0.0927 - val_loss: 0.0915
Epoch 48/50
235/235 [==============================] - 3s 12ms/step - loss: 0.0926 - val_loss: 0.0915
Epoch 49/50
235/235 [==============================] - 3s 12ms/step - loss: 0.0926 - val_loss: 0.0915
Epoch 50/50
235/235 [==============================] - 3s 12ms/step - loss: 0.0926 - val_loss: 0.0915
```



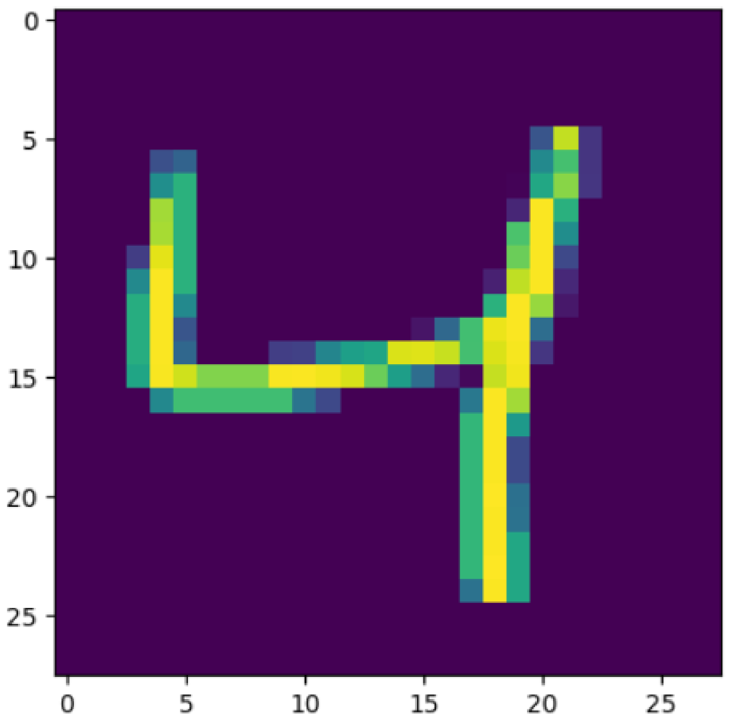
Practical No: 9

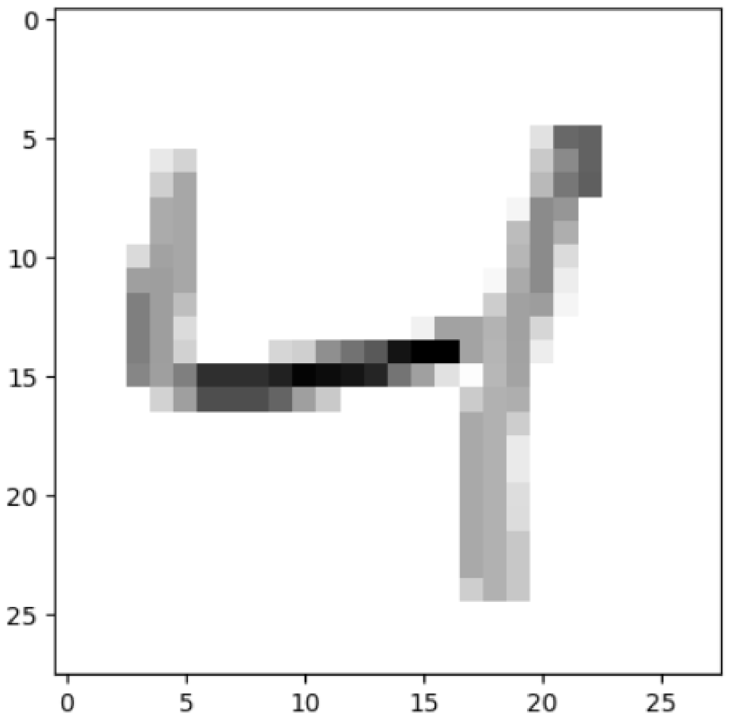
Aim: Implementation of convolutional neural network to predict numbers from number images.

Code:

```
import tensorflow as tf
mnist = tf.keras.datasets.mnist
(X_train, y_train), (X_test, y_test) = mnist.load_data()

X_train.shape
y_train.shape
X_test.shape
y_test.shape

import matplotlib.pyplot as plt
plt.imshow(X_train[2])
plt.show()
plt.imshow(X_train[2], cmap=plt.cm.binary)

X_train[2]
X_train = tf.keras.utils.normalize(X_train, axis=1)
X_test = tf.keras.utils.normalize(X_test, axis=1)
plt.imshow(X_train[2], cmap=plt.cm.binary)
print(X_train[2])

import tensorflow as tf
import tensorflow.keras.layers  as KL

import tensorflow.keras.models  as KM


inputs = KL.Input(shape=(28, 28, 1))

c = KL.Conv2D(32, (3, 3), padding="valid", activation=tf.nn.relu)(inputs)

m = KL.MaxPool2D((2, 2), (2, 2))(c)

d = KL.Dropout(0.5)(m)

c = KL.Conv2D(64, (3, 3), padding="valid", activation=tf.nn.relu)(d)

m = KL.MaxPool2D((2, 2), (2, 2))(c)

d = KL.Dropout(0.5)(m)

c = KL.Conv2D(128, (3, 3), padding="valid", activation=tf.nn.relu)(d)

f = KL.Flatten()(c)

outputs = KL.Dense(10, activation=tf.nn.softmax)(f)

model = KM.Model(inputs, outputs)

model.summary()

model.compile(optimizer="adam", loss="sparse_categorical_crossentropy",
metrics=["accuracy"])


model.fit(X_train, y_train, epochs=5)

test_loss, test_acc = model.evaluate(X_test, y_test)

print("Test Loss: {0} - Test Acc: {1}".format(test_loss, test_acc))
```

Output:


0
5
10
15
20
25


0
5
10
15
20
25

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 28, 28, 1)]       0

 conv2d (Conv2D)             (None, 26, 26, 32)        320

 max_pooling2d (MaxPooling2  (None, 13, 13, 32)        0
 D)

 dropout (Dropout)           (None, 13, 13, 32)        0

 conv2d_1 (Conv2D)           (None, 11, 11, 64)        18496

 max_pooling2d_1 (MaxPoolin  (None, 5, 5, 64)          0
 g2D)

 dropout_1 (Dropout)         (None, 5, 5, 64)          0

 conv2d_2 (Conv2D)           (None, 3, 3, 128)         73856

 flatten (Flatten)           (None, 1152)              0

 dense (Dense)               (None, 10)                11530

=================================================================
Total params: 104202 (407.04 KB)
Trainable params: 104202 (407.04 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________

Epoch 1/5
1875/1875 [==============================] - 65s 34ms/step - loss: 0.2675 - accuracy: 0.9175
Epoch 2/5
1875/1875 [==============================] - 62s 33ms/step - loss: 0.0983 - accuracy: 0.9699
Epoch 3/5
1875/1875 [==============================] - 64s 34ms/step - loss: 0.0755 - accuracy: 0.9767
Epoch 4/5
1875/1875 [==============================] - 62s 33ms/step - loss: 0.0654 - accuracy: 0.9796
Epoch 5/5
1875/1875 [==============================] - 64s 34ms/step - loss: 0.0568 - accuracy: 0.9822
313/313 [==============================] - 3s 9ms/step - loss: 0.0309 - accuracy: 0.9897
Test Loss: 0.03090468980371952 - Test Acc: 0.9897000193595886
```

Practical No: 10

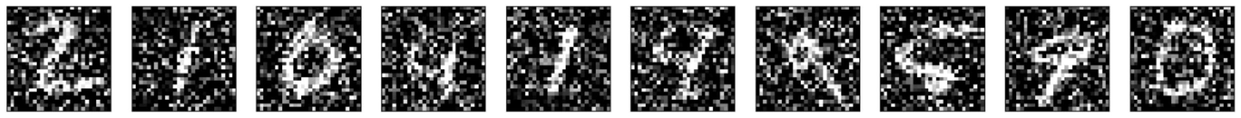
Aim: Denoising of images using autoencoder.

Code:

```
import keras

from keras.datasets import mnist

from keras import layers

import numpy as np

from keras.callbacks import TensorBoard

import matplotlib.pyplot as plt


(X_train, _), (X_test, _) = mnist.load_data()

X_train = X_train.astype('float32') / 255.

X_test = X_test.astype('float32') / 255.

X_train = np.reshape(X_train, (len(X_train), 28, 28, 1))

X_test = np.reshape(X_test, (len(X_test), 28, 28, 1))


noise_factor = 0.5

X_train_noisy = X_train + noise_factor * np.random.normal(loc=0.0, scale=1.0,
size=X_train.shape)

X_test_noisy = X_test + noise_factor * np.random.normal(loc=0.0, scale=1.0, size=X_test.shape)

X_train_noisy = np.clip(X_train_noisy, 0., 1.)

X_test_noisy = np.clip(X_test_noisy, 0., 1.)


n = 10

plt.figure(figsize=(20, 2))
```

```
for i in range(1, n + 1):
    ax = plt.subplot(1, n, i)
    plt.imshow(X_test_noisy[i].reshape(28, 28))
    plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)
plt.show()

input_img = keras.Input(shape=(28, 28, 1))
x = layers.Conv2D(32, (3, 3), activation='relu', padding='same')(input_img)
x = layers.MaxPooling2D((2, 2), padding='same')(x)
x = layers.Conv2D(32, (3, 3), activation='relu', padding='same')(x)
encoded = layers.MaxPooling2D((2, 2), padding='same')(x)

x = layers.Conv2D(32, (3, 3), activation='relu', padding='same')(encoded)
x = layers.UpSampling2D((2, 2))(x)
x = layers.Conv2D(32, (3, 3), activation='relu', padding='same')(x)
x = layers.UpSampling2D((2, 2))(x)
decoded = layers.Conv2D(1, (3, 3), activation='sigmoid', padding='same')(x)

autoencoder = keras.Model(input_img, decoded)
autoencoder.compile(optimizer='adam', loss='binary_crossentropy')
autoencoder.fit(X_train_noisy, X_train,
                epochs=3,
                batch_size=128,
                shuffle=True,
                validation_data=(X_test_noisy, X_test),
        callbacks=[TensorBoard(log_dir='/tmo/tb', histogram_freq=0, write_graph=False)])

predictions = autoencoder.predict(X_test_noisy)
m = 10
plt.figure(figsize=(20, 2))
for i in range(1, m + 1):
    ax = plt.subplot(1, m, i)
    plt.imshow(predictions[i].reshape(28, 28))
    plt.gray()
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)
plt.show()
```

Output:



```
Epoch 1/3
469/469 [==============================] - 133s 281ms/step - loss: 0.1604 - val_loss: 0.1171
Epoch 2/3
469/469 [==============================] - 115s 245ms/step - loss: 0.1126 - val_loss: 0.1077
Epoch 3/3
469/469 [==============================] - 114s 242ms/step - loss: 0.1073 - val_loss: 0.1063
313/313 [==============================] - 5s 15ms/step
```