DEPARTMENT OF INFORMATION TECHNOLOGY

SMT. PARMESHWARI DEVI DURGADUTT TIBREWALA

LIONS JUHU COLLEGE

OF ARTS, COMMERE AND SCIENCE

Affiliated to University of Mumbai

J.B. NAGAR, ANDHERI (E), MUMBAI-400059

Academic Year 2022-2023

For

Semester IV

Submitted By:

Tufail Shaikh

Msc.IT (Sem IV)

Natural Language Processing

DEPARTMENT OF INFORMATION TECHNOLOGY

SMT. PARMESHWARIDEVI DURGADUTT TIBREWALA

LIONS JUHU COLLEGE

OF ARTS, COMMERE AND SCIENCE

Affiliated to University of Mumbai

J.B. NAGAR, ANDHERI (E), MUMBAI-400059

Academic Year 2022-2023

Natural Language Processing

For

Semester IV

Submitted By:

Tufail Shaikh

Msc.IT (Sem IV)

SMT. PARMESHWARIDEVI DURGADUTT TIBREWALA

LIONS JUHU COLLEGE

OF ARTS, COMMERE AND SCIENCE

Affiliated to University of Mumbai

J.B. NAGAR, ANDHERI (E), MUMBAI-400059

DEPARTMENT OF INFORMATION TECHNOLOGY

Certificate of Approval

This is to certify that practical entitled **"Natural Language Processing"** Undertaken at **SMT.PARMESHWARIDEVI DURGADUTT TIBREWALA LIONS JUHU COLLEGEOF ARTS, COMMERECE & SCIENCE.** By **Tufail Shaikh** **Seat No. _____** in partial fulfilment of **M.Sc. (IT) (Semester IV)** Examination had not been submitted for any other examination and does not form of any other course undergone by the candidate. It is further certified that she has completed all required phases of the practical.

_____ **_____**

Internal Examiner **External Examiner**

_____ **_____**

HOD / In-Charge / Coordinator **Signature/**

Principal/Stamp

INDEX

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.		
3. A)	Study of Wordnet Dictionary with methods as synsets, definitions, examples, antonyms		
B)	Study lemmas, hyponyms, hypernyms.		

C)	Write a program using python to find synonyms and antonyms of the word "active" using Wordnet.		
D)	Compare two nouns		
E)	Handling stopword: 1) Using nltk Adding or Removing Stop Words in NLTK's Default Stop Word List 2) Using Gensim Adding and Removing Stop Words in Default Gensim Stop Words List 3) 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.	Import 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.	Finite state automata		
A)	Define grammar using nltk. Analyze a sentence using the same.		
B)	Accept the input string with Regular expression of Finite Automaton: 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. A)	Speech Tagging: 1) Speech tagging using spacy 2) Speech tagging using nltk		
B)	Statistical parsing: 1) Usage of Give and Gave in the Penn Treebank sample 2) Probabilistic parser		
C)	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		

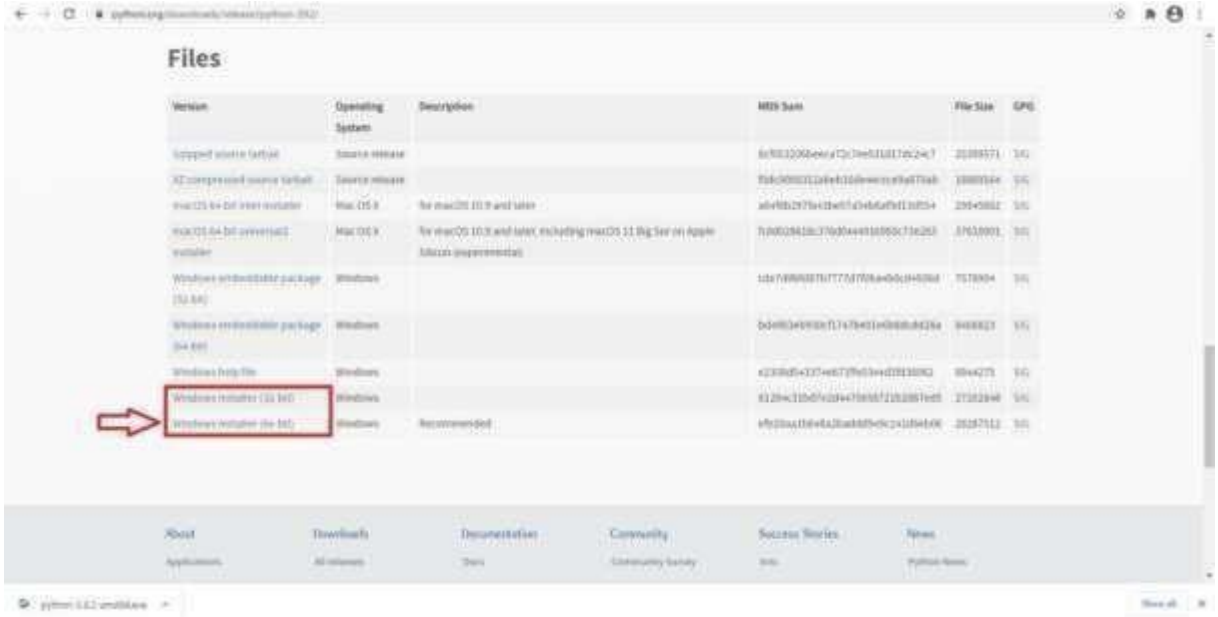
PRACTICAL NO.: 1

[A] Install NLTK.

Python 3.9.2 Installation on Windows

Step 1: Go to link https://www.python.org/downloads/, and select the latest version for windows.

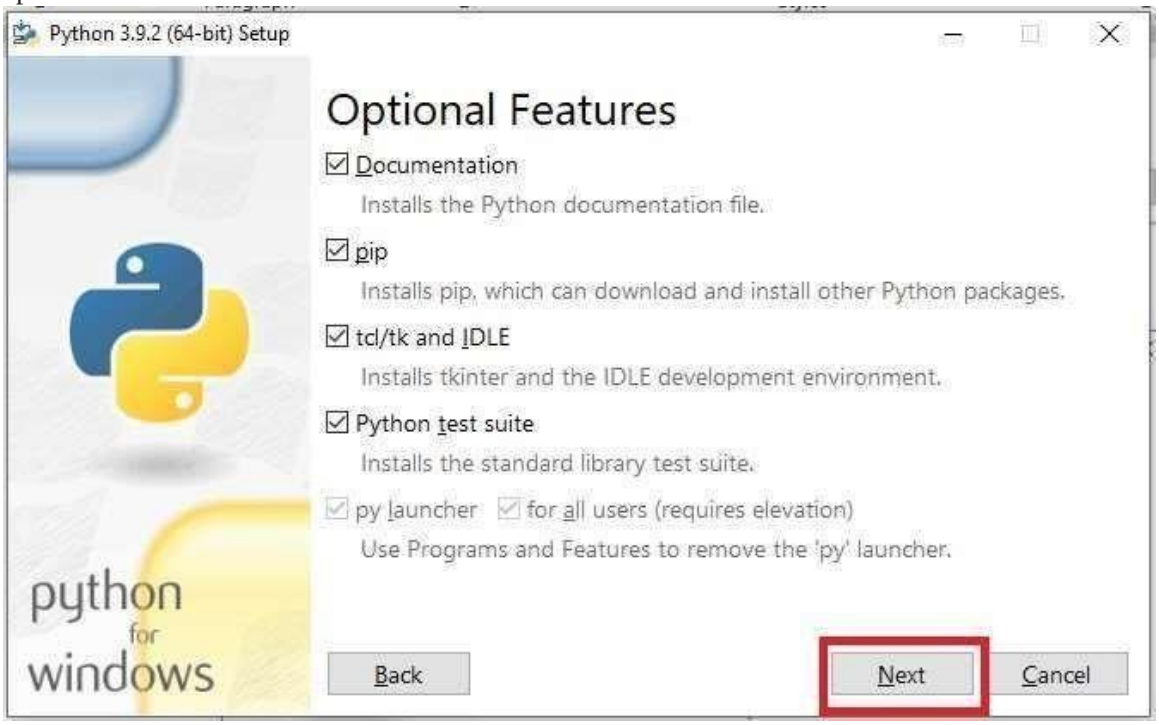
Note: If you don't want to download the latest version, you can visit the download tab and see all releases.



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

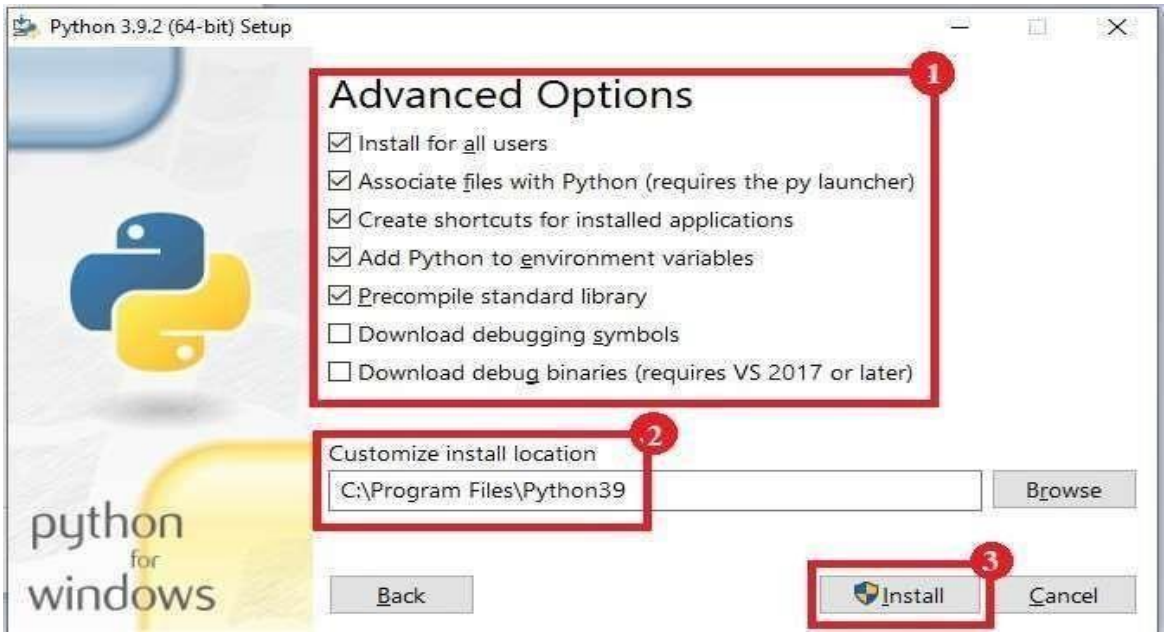
Step 3: Select Customize Installation

Step 4: Click NEXT



Step 5: In next screen

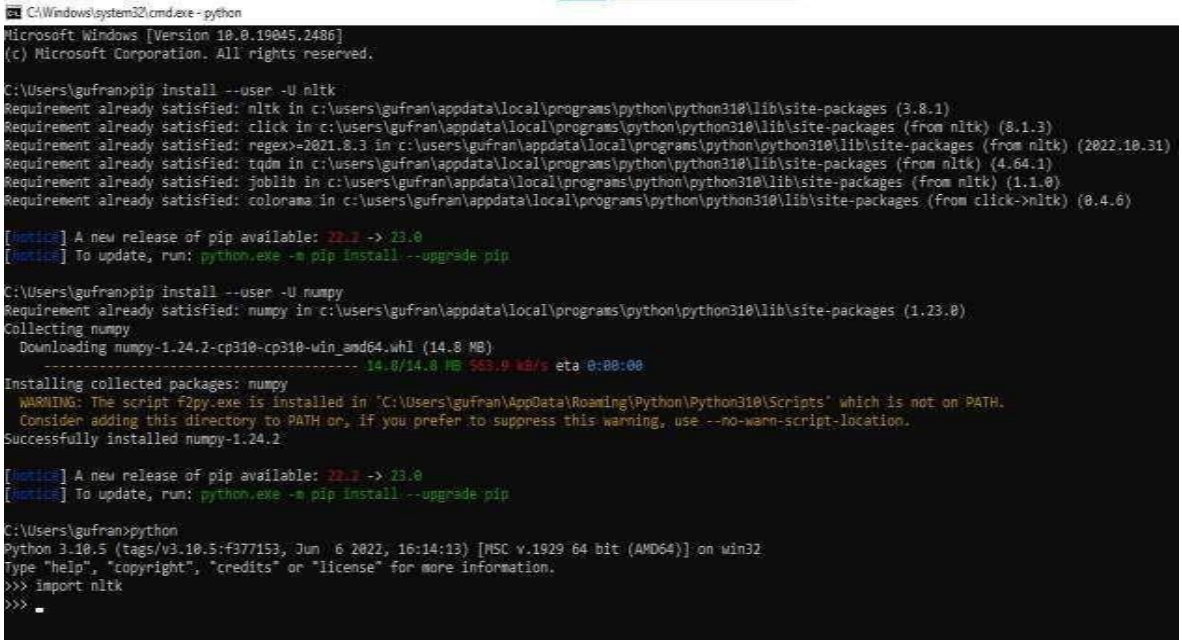
1. Select the advanced options
2. Give a Custom install location. Keep the default folder as c:\Program files\Python39
3. Click Install



Step 6: Click Close button once install is done.

Step 7: Open command prompt window and run the following commands:

>pip install --upgrade pip
>pip install --user -U nltk
>pip install --user –U
>python



[B] Convert the given text to speech.

CODE:

nlp1b.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp1b.

File Edit Format Run Options Window Help

```
from playsound import playsound
from 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")
print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

IDLE Shell 3.11.2

File Edit Shell Debug Options Window Help

```
Python 3.11.2 (tags/v3.11.2:878ead1, Feb  7 2023, 16:38:35) [MSC v.1934 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp1b.py =
Uzma Khan_09_M.L.D.C
>>>
```

[C] Convert audio file Speech to Text.

CODE:

nlp1c.py - C:\Uzma\nlp1c.py (3.11.3)

File Edit Format Run Options Window Help

```
import speech_recognition as sr
filename="myfile.wav"
r=sr.Recognizer()
with sr.AudioFile(filename)as source:
                audio_data=r.record(source)
                text=r.recognize_google(audio_data)
                print(text)

print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

IDLE Shell 3.11.3

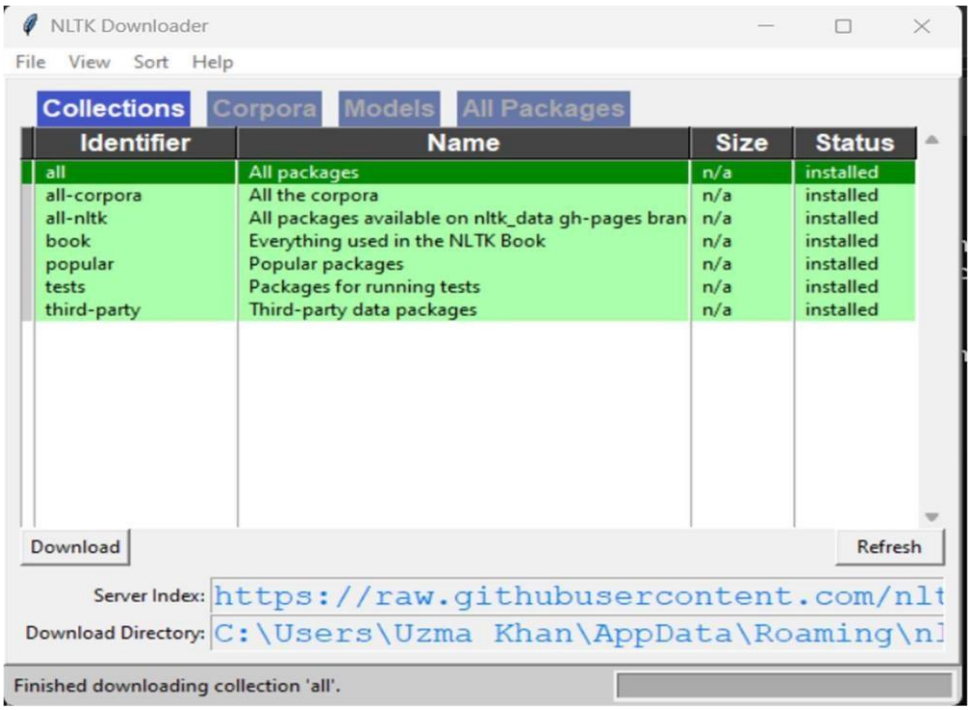
File Edit Shell Debug Options Window Help

```
Python 3.11.3 (tags/v3.11.3:f3909b8, Apr  4 2023, 23:49:59) [MSC v.1934 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:\Uzma\nlp1c.py ==========
talking nonsense
Uzma Khan_09_M.L.D.C
>>>
```

PRACTICAL NO.: 2

[A] Study of various Corpus – Brown, Inaugural, Reuters, udhr with various methods like filelds, raw, words, sents, categories.

```
File  Edit  Shell  Debug  Options  Window  Help
Python 3.11.2 (tags/v3.11.2:878ead1, Feb  7 2023, 16:38:35) [MSC v.1934 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> import nltk
>>> nltk.download()
showing info https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/index.xml
True
>>>
```




```
>>> from nltk.book import *
*** Introductory Examples for the NLTK Book ***
Loading text1, ..., text9 and sent1, ..., sent9
Type the name of the text or sentence to view it.
Type: 'texts()' or 'sents()' to list the materials.
text1: Moby Dick by Herman Melville 1851
text2: Sense and Sensibility by Jane Austen 1811
text3: The Book of Genesis
text4: Inaugural Address Corpus
text5: Chat Corpus
text6: Monty Python and the Holy Grail
text7: Wall Street Journal
text8: Personals Corpus
text9: The Man Who Was Thursday by G . K . Chesterton 1908
>>>
```

CODE:

nlp2a.py - C:\Users\Uzma Khan\AppData\Local\Programs\Python\Python311\nlp2a.py (3.11.2)

File Edit Format Run Options Window Help

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

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())

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)
print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

IDLE Shell 3.11.2

File Edit Shell Debug Options Window Help

```
Python 3.11.2 (tags/v3.11.2:878ead1, Feb  7 2023, 16:38:35) [MSC v.1934 64 bit (A
MD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\Uzma Khan\AppData\Local\Programs\Python\Python311\nlp2a.py =
File ids of brown corpus
```

Squeezed text (50 lines).

```
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
8                       23                      2                       cr09
Uzma Khan_09_M.L.D.C
>>>
```

[B] Create and use your own corpora (plaintext, categorical).

CODE:

nlp2b.py - C:\Users\Uzma Khan\AppData\Local\Programs\Python\Python311\nlp2b.py (3.11....

File Edit Format Run Options Window Help

```
import nltk
from nltk.corpus import PlaintextCorpusReader
corpus_root = r'C:\Users\Uzma Khan\Desktop\NLP'
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
corresponding to the filelist file identifiers listed earlier and then computing
for 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)

print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

IDLE Shell 3.11.2

File Edit Shell Debug Options Window Help

```
Python 3.11.2 (tags/v3.11.2:878ead1, Feb  7 2023, 16:38:35) [MSC v.1934 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\Uzma Khan\AppData\Local\Programs\Python\Python311\nlp2b.py =

 File list:

['safety codes.txt']
C:\Users\Uzma Khan\Desktop\NLP


Statistics for each text:

AvgWordLen      AvgSentenceLen  no.ofTimesEachWordAppearsOnAvg  FileName
6                       37                      7               safety codes.txt
Uzma Khan_09_M.L.D.C
>>>
```

[C] Study Conditional frequency distributions.

CODE:

nlp2c.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp2c.py (3.11.2)

File Edit Format Run Options Window Help

```
#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 brown
fd = 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 inaugural
cfd = 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)
print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

```
>>
== RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp2c.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 (1119 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
Uzma Khan_09_M.L.D.C
>>
```

[D] Study of tagged corpora with methods like tagged_sents, tagged_words.

CODE:

nlp2d.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp2d.py (3.11...

File Edit Format Run Options Window Help

```
import nltk
from nltk import tokenize
nltk.download('punkt')
nltk.download('words')
para = "Hello! My name is Uzma Khan. 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)

print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

IDLE Shell 3.11.2

File Edit Shell Debug Options Window Help

```
Python 3.11.2 (tags/v3.11.2:878ead1, Feb  7 2023, 16:38:35) [MSC v.1934 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp2d.py =
[nltk_data] Downloading package punkt to C:\Users\Uzma
[nltk_data]     Khan\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package words to C:\Users\Uzma
[nltk_data]     Khan\AppData\Roaming\nltk_data...
[nltk_data]   Package words is already up-to-date!

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

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

['Today', 'you', "'ll", 'be', 'learning', 'NLTK', '.']
Uzma Khan_09_M.L.D.C
>>>
```

[E] Write a program to find the most frequent noun tags.

CODE:

nlp2e.py - C:\Users\Uzma Khan\AppData\Local\Programs\Python\Python311\nlp2e.py (3.11.2)

File Edit Format Run Options Window Help

```
import nltk
from collections import defaultdict
text = nltk.word_tokenize("Nick likes to play football. Nick does not like to pl
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))

print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

IDLE Shell 3.11.2

File Edit Shell Debug Options Window Help

```
Python 3.11.2 (tags/v3.11.2:878ead1, Feb  7 2023, 16:38:35) [MSC v.1934 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\Uzma Khan\AppData\Local\Programs\Python\Python311\nlp2e.py =
[('Nick', 'NNP'), ('likes', 'VBZ'), ('to', 'TO'), ('play', 'VB'), ('football', '
NN'), ('.', '.'), ('Nick', 'NNP'), ('does', 'VBZ'), ('not', 'RB'), ('like', 'VB'
), ('to', 'TO'), ('playcricket', 'VB'), ('.', '.')]
['Nick']
Word with maximum frequency : Nick
Uzma Khan_09_M.L.D.C
>>>
```

[F] Map Words to Properties Using Python Dictionaries.

CODE:

nlp2f.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp2f.py (3.11.2)

File Edit Format Run Options Window Help

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

print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

IDLE Shell 3.11.2

File Edit Shell Debug Options Window Help

```
Python 3.11.2 (tags/v3.11.2:878ead1, Feb  7 2023, 16:38:35) [MSC v.1934 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp2f.py =
{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}
Ford
3
<class 'dict'>
Uzma Khan_09_M.L.D.C
>>>
```

[G] Study i) DefaultTagger, ii) Regular expression tagger, iii) UnigramTagger. i) DefaultTagger:

CODE:

nlp2g1.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp2g1.py (3....

File Edit Format Run Options Window Help

```
import nltk
from nltk.tag import DefaultTagger
exptagger = DefaultTagger('NN')
from nltk.corpus import treebank
testsentences = treebank.tagged_sents() [1000:]
print(exptagger.accuracy (testsentences))
#Tagging a list of sentences
import nltk
from nltk.tag import DefaultTagger
exptagger = DefaultTagger('NN')
print(exptagger.tag_sents([['Hi', ','], ['How', 'are', 'you', '?']]))
print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

```
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp2g1.py =
0.13198749536374715
[[('Hi', 'NN'), (',', 'NN')], [('How', 'NN'), ('are', 'NN'), ('you', 'NN'), ('?',
'NN')]]
Uzma Khan_09_M.L.D.C
>>>
```

ii) Regular expression tagger:

CODE:

nlp2g2.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp2

File Edit Format Run Options Window Help

```
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))
print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

```
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp2g2.py =
<Regexp Tagger: size=9>
[('The', 'AT'), ('Fulton', 'NN'), ('County', 'NN'), ('Grand', 'NN'), ('Jury', 'NN
'), ('said', 'NN'), ('Friday', 'NN'), ('an', 'AT'), ('investigation', 'NN'), ('of
', 'NN'), ("Atlanta's", 'NNS'), ('recent', 'NN'), ('primary', 'NN'), ('election',
'NN'), ('produced', 'VBD'), ('``', 'NN'), ('no', 'NN'), ('evidence', 'NN'), ("''"
, 'NN'), ('that', 'NN'), ('any', 'NN'), ('irregularities', 'NNS'), ('took', 'NN')
, ('place', 'NN'), ('.', 'NN')]
Uzma Khan_09_M.L.D.C
>>>
```

iii) UnigramTagger.

CODE:

nlp2g3.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp2g3.py (3....

File Edit Format Run Options Window Help

```
# Loading Libraries
from nltk.tag import UnigramTagger
from 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]))
print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

```
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp2g3.py =
['Pierre', 'Vinken', ',', '61', 'years', 'old', ',', 'will', 'join', 'the', 'board',
'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), ('board', N
one), ('as', None), ('a', None), ('nonexecutive', None), ('director', None), ('Nov.'
, None), ('29', None), ('.', None)]
Uzma Khan_09_M.L.D.C
>>>
```

[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. Also find 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) CODE:

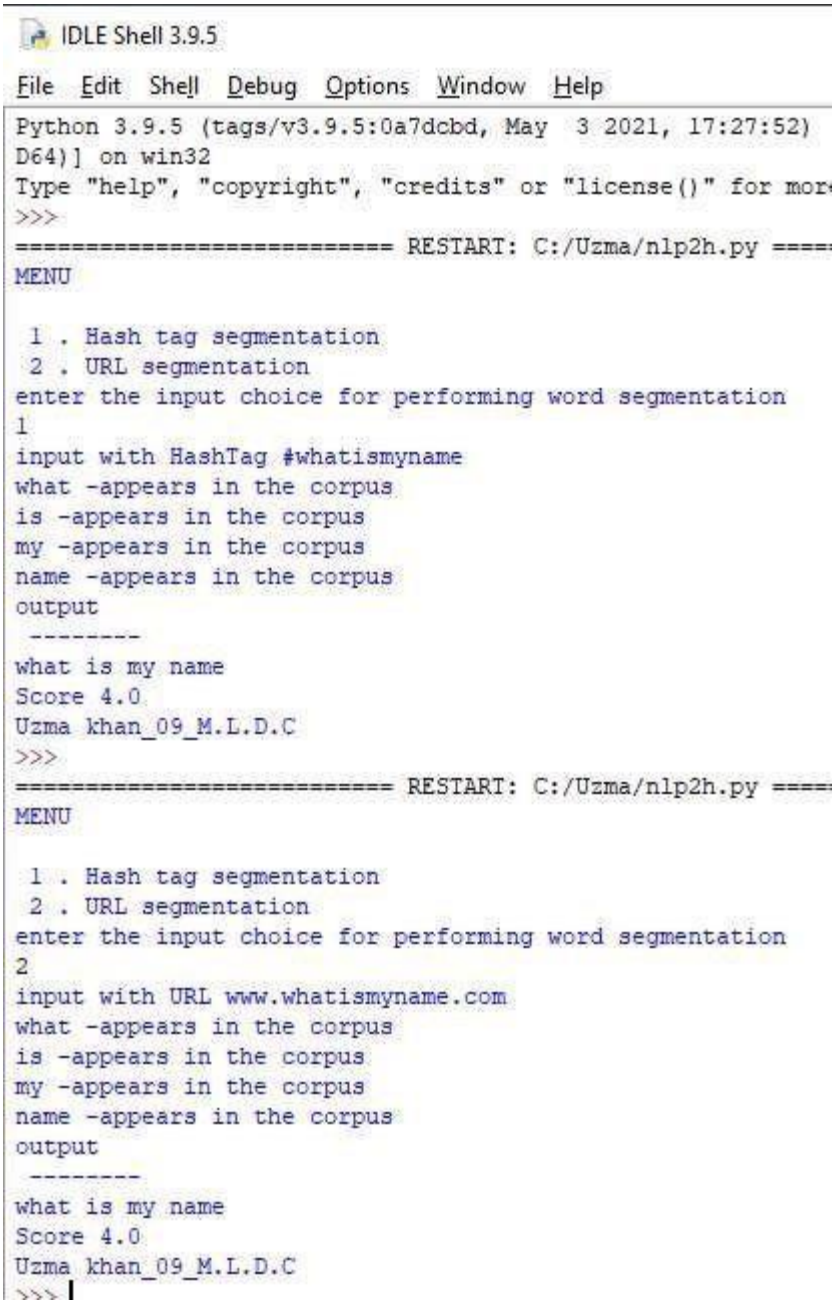
nlp2h.py - C:\Uzma\nlp2h.py (3.11.3)

File Edit Format Run Options Window Help

```
import future #with statement for reading file
import re # Regular expression
words = [] # corpus file words
testword = [] # 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 Score
C = len(answer)
score = C * N / N # Calculate the score
print("Score",score)
print('Uzma khan_09_M.L.D.C')
```

OUTPUT:


IDLE Shell 3.9.5
File Edit Shell Debug Options Window Help
Python 3.9.5 (tags/v3.9.5:0a7dcbd, May 3 2021, 17:27:52)
D64)] on win32
Type "help", "copyright", "credits" or "license()" for mor
>>>
========== RESTART: C:/Uzma/nlp2h.py ====
MENU
1 . Hash tag segmentation
2 . URL segmentation
enter the input choice for performing word segmentation
1
input with HashTag #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
Uzma khan_09_M.L.D.C
>>>
========== RESTART: C:/Uzma/nlp2h.py ====
MENU
1 . Hash tag segmentation
2 . URL segmentation
enter the input choice for performing word segmentation
2
input with URL www.whatismyname.com
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
Uzma khan_09_M.L.D.C
>>>

PRACTICAL NO.: 3

[A] Study of Wordnet Dictionary with methods as synsets, definitions, examples, antonyms.

CODE:

nlp3a.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp3a.py (3.11.2)

File Edit Format Run Options Window Help

```
'''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())
print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

IDLE Shell 3.11.2

File Edit Shell Debug Options Window Help

```
Python 3.11.2 (tags/v3.11.2:878ead1, Feb  7 2023, 16:38:35) [MSC v.1934 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp3a.py =
[Synset('computer.n.01'), Synset('calculator.n.01')]
a machine for performing calculations automatically
Examples: []
[Lemma('sell.v.01.sell')]
Uzma Khan_09_M.L.D.C
>>>
```

[B] Study lemmas, hyponyms, hypernyms.

CODE:

nlp3b.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp3b.py (3.11...

File Edit Format Run Options Window Help

```
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))
print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

```
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp3b.py =
[Synset('computer.n.01'), Synset('calculator.n.01')]
['computer', 'computing_machine', 'computing_device', 'data_processor', 'electroni
c_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'), Lemma(
'computer.n.01.computing_device'), Lemma('computer.n.01.data_processor'), Lemma('c
omputer.n.01.electronic_computer'), Lemma('computer.n.01.information_processing_sy
stem')]
Synset('computer.n.01')
computing_device
<bound method _WordNetObject.hyponyms of Synset('computer.n.01')>
['analog_computer', 'analogue_computer', 'digital_computer', 'home_computer', 'nod
e', 'client', 'guest', 'number_cruncher', 'pari-mutuel_machine', 'totalizer', 'tot
aliser', 'totalizator', 'totalisator', 'predictor', 'server', 'host', 'Turing_mach
ine', 'web_site', 'website', 'internet_site', 'site']
[Synset('vehicle.n.01')]
Uzma Khan_09_M.L.D.C
>>
```

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

CODE:

nlp3c.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp3c.py (3.11.2)

File Edit Format Run Options Window Help

```
from nltk.corpus import wordnet
print( wordnet.synsets("active"))
print(wordnet.lemma('active.a.01.active').antonyms())
print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

```
== RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp3c.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')]
Uzma Khan_09_M.L.D.C
>>>
```

[D] Compare two nouns.

CODE:

nlp3d.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp3d.py (3.11.2)

File Edit Format Run Options Window Help

```
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 word1 with 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()
print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

IDLE Shell 3.11.2

File Edit Shell Debug Options Window Help

```
Python 3.11.2 (tags/v3.11.2:878ead1, Feb  7 2023, 16:38:35) [MSC v.1934 64 bit (AMD64)] on
win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp3d.py =
Path similarity of:
Synset('football.n.01') ( n ) [ any of various games played with a ball (round or oval) in
which two teams try to kick or carry or propel the ball into each other's goal ]
Synset('soccer.n.01') ( n ) [ a football game in which two teams of 11 players try 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 American football
]
Synset('soccer.n.01') ( n ) [ a football game in which two teams of 11 players try to kick
or head a ball into the opponents' goal ]
 is 0.05

Uzma Khan_09_M.L.D.C
>>>
```

[E] Handling stopword:

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

CODE:

nlp3e1.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp3e1.py (3.11.2)

File Edit Format Run Options Window Help

```
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)
print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

IDLE Shell 3.11.2

File Edit Shell Debug Options Window Help

```
Python 3.11.2 (tags/v3.11.2:878ead1, Feb  7 2023, 16:38:35) [MSC v.1934 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp3e1.py
[nltk_data] Downloading package stopwords to C:\Users\Uzma
[nltk_data]     Khan\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
['Yashesh', 'likes', 'play', 'football', ',', 'fond', 'tennis', '.']
['Yashesh', 'likes', 'football', ',', 'however', 'fond', 'tennis', '.']
['Yashesh', 'likes', 'football', ',', 'however', 'not', 'fond', 'tennis', '.']
Uzma Khan_09_M.L.D.C
>>>
```

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

CODE:

nlp3e2.py - C:\Users\HP\AppData\Local\Programs\Python\Python311\nlp3e2.py (3.11...

File Edit Format Run Options Window Help

```
import nltk
from nltk.tokenize import word_tokenize
import 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)


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)

print("\n Uzma Khan_09_M.L.D.C")
```

OUTPUT:

```
==== RESTART: C:\Users\HP\AppData\Local\Programs\Python\Python311\nlp3e2.py ====
Yashesh likes play football, fond tennis.
frozenset({'detail', 'you', 'between', 'all', 'has', 'beside', 'becomes', 'everyone', 'besides', 'some
how', 'fire', 'find', 'moreover', 'yours', 'per', 'about', 'thin', 'six', 'ten', 'most', 'part', 'show
', 'nor', 'whoever', 'eight', 'their', 'during', 'seeming', 'etc', 'is', 'she', 'to', 'might', 'for',
'through', 'km', 'own', 'would', 'eg', 'thus', 'whenever', 'interest', 'un', 'whose', 'just', 'no', 's
ide', 'will', 'already', 'due', 'also', 'whole', 'alone', 'often', 'why', 'cannot', 'while', 'regardin
g', 'more', 'toward', 'when', 'de', 'put', 'latter', 'them', 'over', 'by', 'his', 'almost', 'we', 'was
', 'on', 'make', 'one', 'myself', 'top', 'beyond', 'there', 'above', 'whereby', 'meanwhile', 'as', 'ev
er', 'first', 'using', 'sometimes', 'does', 'where', 'thru', 'thence', 'kg', 'from', 'empty', 'an', 'a
gain', 'such', 'were', 'at', 'him', 'nevertheless', 'formerly', 're', 'me', 'but', 'they', 'everywhere
', 'below', 'namely', 'together', 'seems', 'hereupon', 'everything', 'could', 'anyone', 'down', 'thems
elves', 'see', 'something', 'please', 'take', 'someone', 'across', 'move', 'did', 'hers', 'cant', 'any
how', 'never', 'he', 'have', 'otherwise', 'upon', 'had', 'along', 'even', 'enough', 'into', 'be', 'say
', 'amount', 'off', 'this', 'before', 'seem', 'became', 'forty', 'couldnt', 'whereas', 'many', 'rather
', 'hence', 'being', 'noone', 'nine', 'anywhere', 'front', 'same', 'whence', 'get', 'because', 'neithe
r', 'computer', 'herein', 'former', 'it', 'onto', 'been', 'these', 'next', 'bill', 'nobody', 'around',
 'sincere', 'another', 'serious', 'thick', 'itself', 'whatever', 'con', 'eleven', 'cry', 'made', 'wher
ever', 'become', 'which', 'don', 'although', 'back', 'done', 'fifty', 'whither', 'her', 'hereafter', '
whether', 'fifteen', 'after', 'in', 'so', 'only', 'others', 'how', 'somewhere', 'am', 'under', 'yet',
'third', 'well', 'elsewhere', 'amongst', 'with', 'yourselves', 'co', 'several', 'two', 'any', 'then',
'other', 'four', 'too', 'less', 'behind', 'thereafter', 'the', 'five', 'our', 'must', 'unless', 'via',
 'full', 'bottom', 'go', 'within', 'us', 'sixty', 'here', 'seemed', 'latterly', 'anything', 'wherein',
 'that', 'out', 'really', 'either', 'ours', 'both', 'ie', 'whereupon', 'except', 'nothing', 'some', 'e
ach', 'inc', 'therein', 'system', 'amoungst', 'still', 'among', 'various', 'those', 'i', 'quite', 'my'
, 'much', 'twelve', 'keep', 'may', 'of', 'hasnt', 'nowhere', 'doesn', 'since', 'throughout', 'herself'
, 'always', 'if', 'mostly', 'call', 'himself', 'not', 'used', 'its', 'therefore', 'your', 'should', 'b
eforehand', 'against', 'hundred', 'further', 'now', 'up', 'very', 'a', 'becoming', 'few', 'doing', 'so
metime', 'what', 'didn', 'ourselves', 'until', 'else', 'yourself', 'hereby', 'thereupon', 'thereby', '
least', 'mill', 'fill', 'however', 'found', 'three', 'towards', 'can', 'anyway', 'describe', 'once', '
without', 'every', 'none', 'though', 'ltd', 'whom', 'than', 'and', 'last', 'perhaps', 'afterwards', 'n
ame', 'do', 'indeed', 'are', 'twenty', 'whereafter', 'or', 'mine', 'give', 'who'})
['Yashesh', 'football', ',', 'fond', 'tennis', '.']
['Yashesh', 'likes', 'play', 'football', ',', 'not', 'fond', 'tennis', '.']

 Uzma Khan_09_M.L.D.C
>>>
```

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

CODE:

nlp3e3.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp3e3.py (3....

File Edit Format Run Options Window Help

```
import spacy
import nltk
from nltk.tokenize import word_tokenize
sp = 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)
print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

IDLE Shell 3.11.2

File Edit Shell Debug Options Window Help

```
Python 3.11.2 (tags/v3.11.2:878ead1, Feb  7 2023, 16:38:35) [MSC v.1934 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp3e3.py
['Yashesh', 'likes', 'football', ',', 'fond', 'tennis', '.']
['Yashesh', 'likes', 'football', ',', 'not', 'fond', 'tennis', '.']
Uzma Khan_09_M.L.D.C
>>>
```

PRACTICAL NO.: 4

[A] Tokenization using Python's split() function.

CODE:

nlp4a.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp4a.py (3.11.2)

File Edit Format Run Options Window Help

```
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 Briefing
model, 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)

print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

IDLE Shell 3.11.2

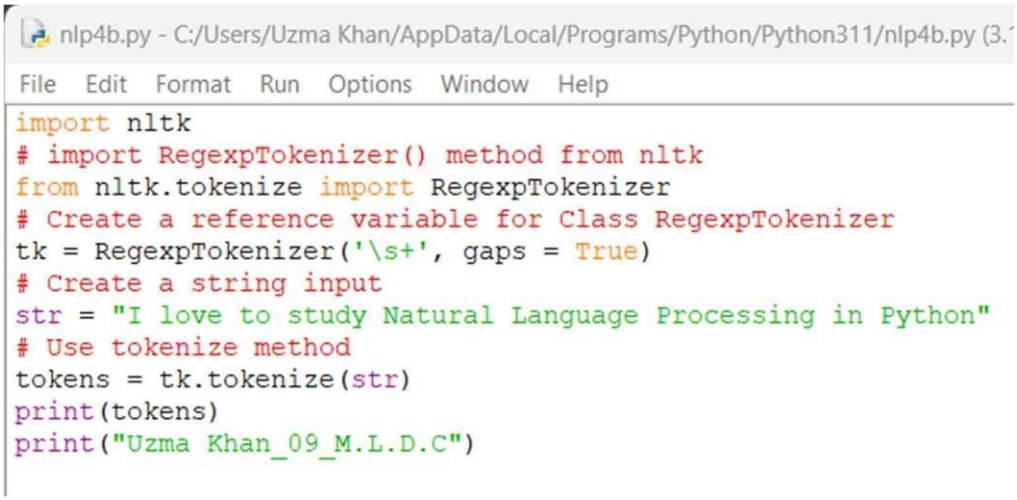
File Edit Shell Debug Options Window Help

```
Python 3.11.2 (tags/v3.11.2:878ead1, Feb  7 2023, 16:38:35) [MSC v.1934 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp4a.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 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

Uzma Khan_09_M.L.D.C
>>>
```

[B] Tokenization using Regular Expressions (RegEx).

CODE:

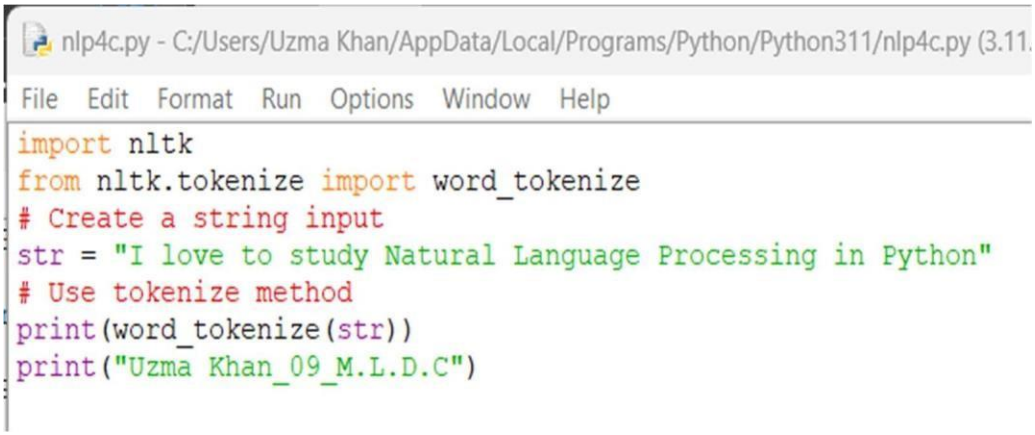


OUTPUT:

```
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp4b.py
['I', 'love', 'to', 'study', 'Natural', 'Language', 'Processing', 'in', 'Python']
Uzma Khan_09_M.L.D.C
>>
```

[C] Tokenization using NLTK.

CODE:



OUTPUT:

```
==== RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp4c.py ====
['I', 'love', 'to', 'study', 'Natural', 'Language', 'Processing', 'in', 'Python']
Uzma Khan_09_M.L.D.C
>>>
```

[D] Tokenization using the spaCy library.

CODE:

nlp4d.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp4d.py (3

File Edit Format Run Options Window Help

```
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)
print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

```
==== RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp4d.py ====
['I', 'love', 'to', 'study', 'Natural', 'Language', 'Processing', 'in', 'Python']
Uzma Khan_09_M.L.D.C
>>>
```

[E] Tokenization using Keras CODE:

nlp4e.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp4e.py (3.11....

File Edit Format Run Options Window Help

```
import 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)

print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

```
==== RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp4e.py ====
['i', 'love', 'to', 'study', 'natural', 'language', 'processing', 'in', 'python']
Uzma Khan_09_M.L.D.C
>>>
```

[F] Tokenization using Gensim.

CODE:

nlp4f.py - C:/Users/HP/AppData/Local/Programs/Python/Python311/nlp4f.py..

File Edit Format Run Options Window Help

```
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))
print("Uzma khan_09_M.L.D.C")
```

OUTPUT:

```
[2]: ['I',
      'love',
      'to',
      'study',
      'Natural',
      'Language',
      'Processing',
      'in',
      'Python']
```

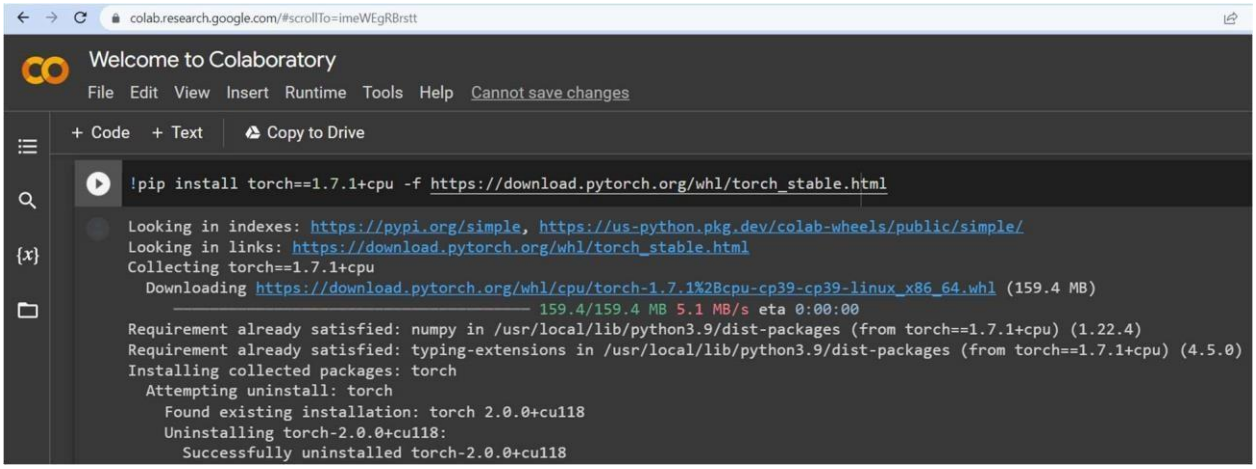
+ Code + Markdown

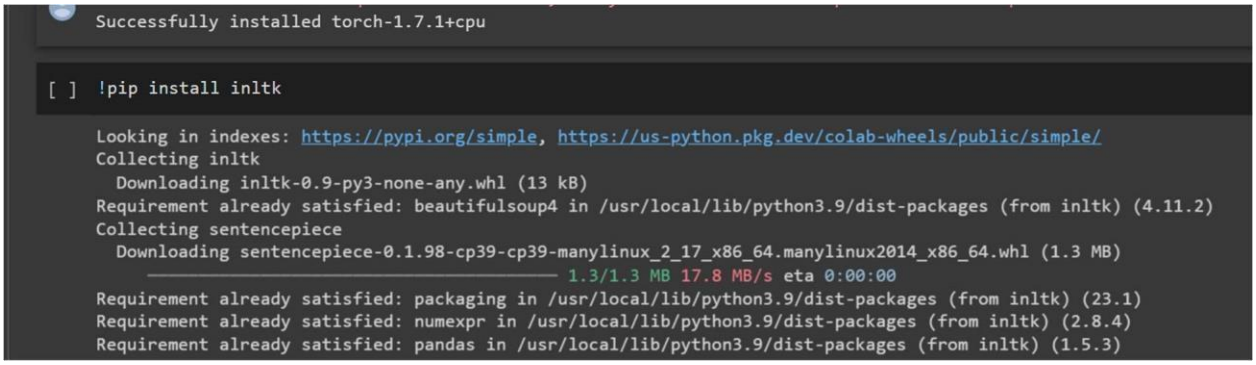
PRACTICAL NO.: 5

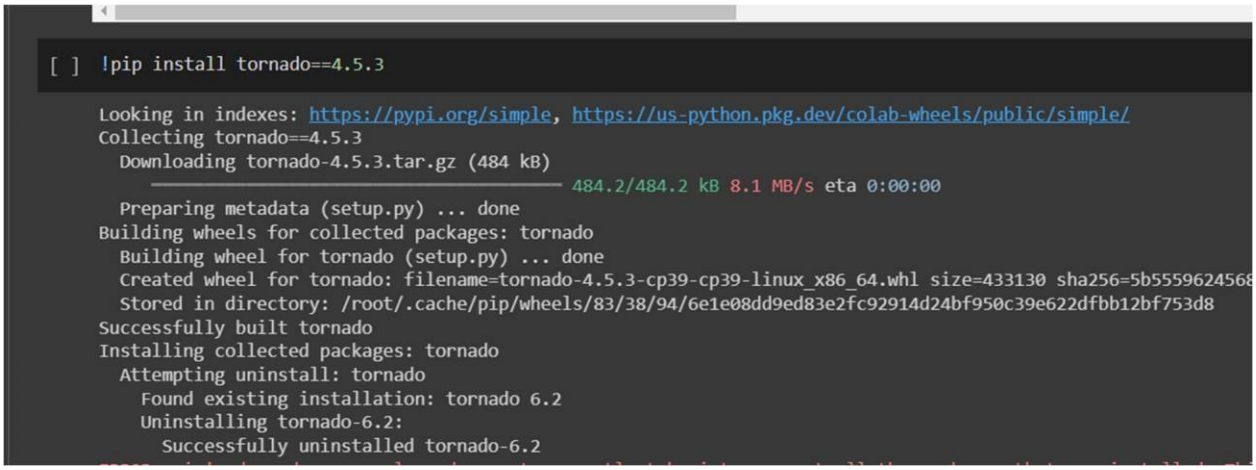
Import NLP Libraries for Indian Languages and perform:

[A] word tokenization in Hindi

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

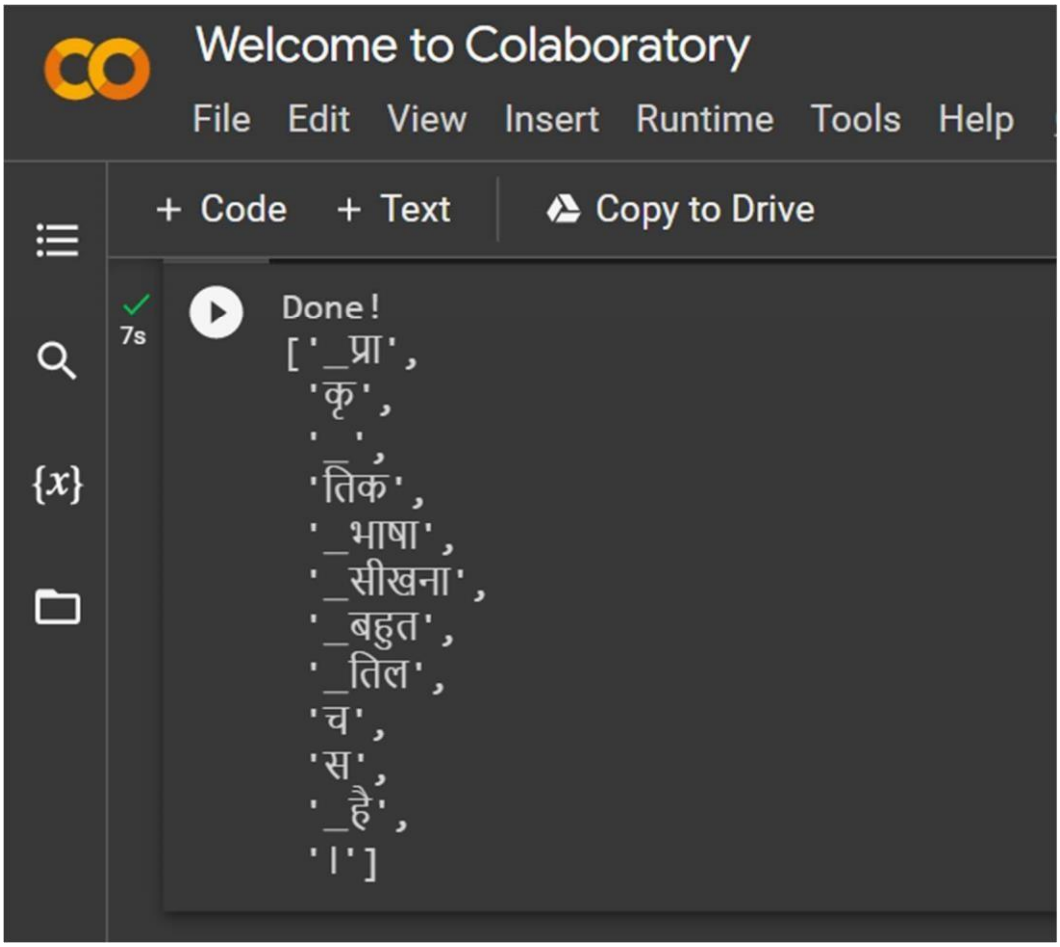






```
[1]  from inltk.inltk import setup
     setup('hi')
     from inltk.inltk import tokenize
     hindi_text ="""प्राकृ तिक भाषा सीखना बहुत तिलचस है।"""
     tokenize(hindi_text, "hi")
```

OUTPUT:



[B] Generate similar sentences from a given Hindi text input

CODE:

NLP5b.py - C:/Users/Uzma Khan/Desktop/NLP/NLP5b.py (3.7.4)

File Edit Format Run Options Window Help

```
#pip install torch==1.7.1+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 बहुत खुश हूं', 5, 'hi')
print(end="\n")
print(output)
```

OUTPUT:

```
============== RESTART: C:/Users/Uzma Khan/Desktop/NLP/NLP5b.py ==============
Done!
  |----------------------------------------| 0.00% [0/1 00:00<?] |
  | 100.00% [1/1 00:00<00:00]
  |----------------------------------------| 0.00% [0/1 00:00<?] |
  | 100.00% [1/1 00:00<00:00]
['मैआज बहाुं नाराज़ हूं', 'मैअमृत बहाुं खुश हूं', 'मैआज अन्तराुं खुश हूं', 'मैआज बहाुं आश्वस्त हूं', 'मैआज साश्लाष्ट खुश हूं']
>>>
```

[C] Identify the Indian language of a text.

CODE:

NLP5c.py - C:/Users/Uzma Khan/Desktop/NLP/NLP5c.py (3.7.4)

File Edit Format Run Options Window Help

```
from inltk.inltk import setup
setup('gu')
from inltk.inltk import identify_language
#Identify the Lnaguage of given text
identify_language('બોના કાપોડયા')
```

OUTPUT:

>>>gujarati

PRACTICAL NO.: 6

Illustrate part of speech tagging.

[A] Part of speech Tagging and chunking of user defined text.

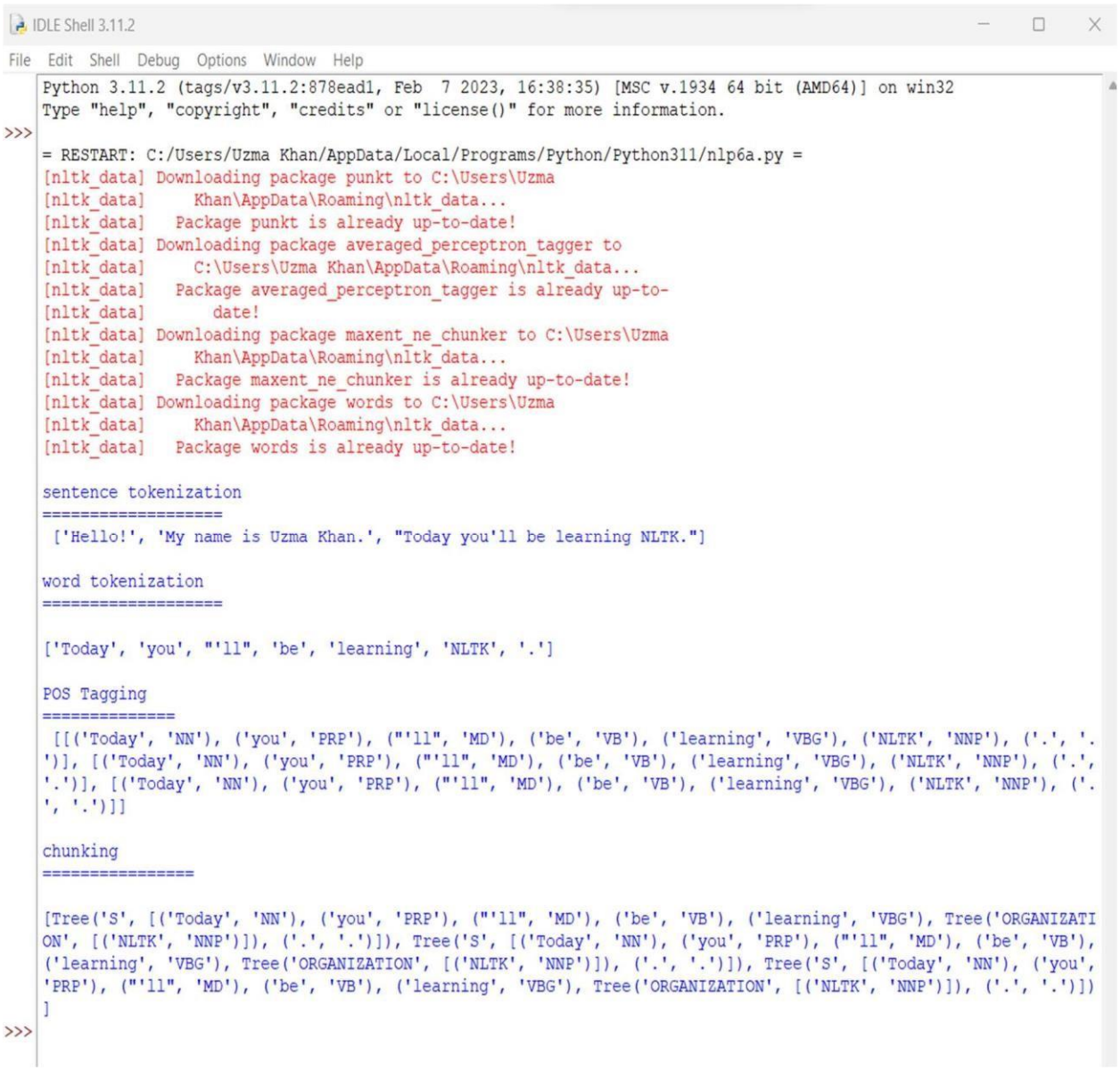
CODE:

nlp6a.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp6a.py (3.11....

File Edit Format Run Options Window Help

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

OUTPUT:


IDLE Shell 3.11.2
File Edit Shell Debug Options Window Help
Python 3.11.2 (tags/v3.11.2:878ead1, Feb 7 2023, 16:38:35) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp6a.py =
[nltk_data] Downloading package punkt to C:\Users\Uzma
[nltk_data] Khan\AppData\Roaming\nltk_data...
[nltk_data] Package punkt is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data] C:\Users\Uzma Khan\AppData\Roaming\nltk_data...
[nltk_data] Package averaged_perceptron_tagger is already up-to-
[nltk_data] date!
[nltk_data] Downloading package maxent_ne_chunker to C:\Users\Uzma
[nltk_data] Khan\AppData\Roaming\nltk_data...
[nltk_data] Package maxent_ne_chunker is already up-to-date!
[nltk_data] Downloading package words to C:\Users\Uzma
[nltk_data] Khan\AppData\Roaming\nltk_data...
[nltk_data] Package words is already up-to-date!
sentence tokenization
['Hello!', 'My name is Uzma Khan.', "Today you'll be learning NLTK."]
word tokenization
['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.

CODE:

nlp6b.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp6b.py (3.11...

File Edit Format Run Options Window Help

```
import spacy
# Load English tokenizer, tagger, parser and NER
nlp = 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"])
print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

```
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp6b.py =
Noun phrases: ['Sebastian Thrun', 'self-driving cars', 'Google', 'few people', 'the
company', 'him', 'I', 'you', 'very senior CEOs', 'major American car companies', 'my
hand', 'I', 'Thrun', 'an interview', 'Recode']
Verbs: ['start', 'work', 'drive', 'take', 'tell', 'shake', 'turn', 'talk', 'say']
Uzma Khan_09_M.L.D.C
>>>
```

[C] Named Entity recognition with diagram using NLTK corpus – treebank.

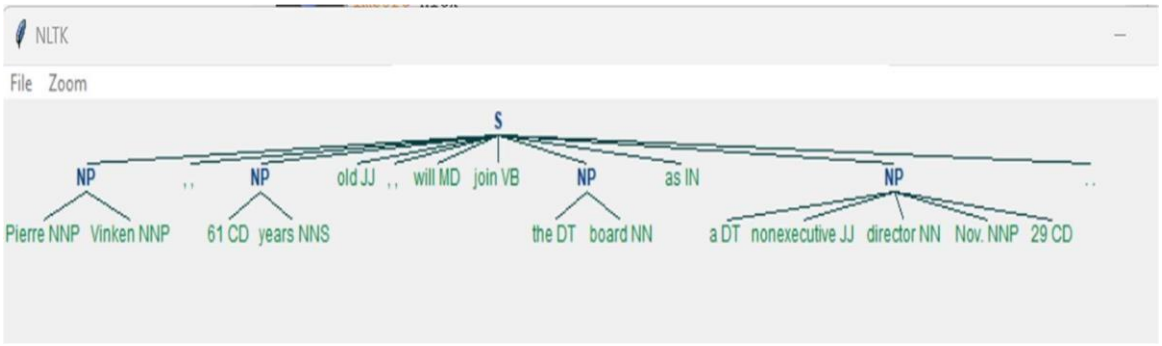
CODE:

nlp6c.py - C:/Users/Uzma Khan/AppData/Local/Programs/Pyt

File Edit Format Run Options Window Help

```
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()
print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:




```
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp6c.py =
[nltk_data] Downloading package treebank to C:\Users\Uzma
[nltk_data]     Khan\AppData\Roaming\nltk_data...
[nltk_data]   Package treebank is already up-to-date!
Uzma Khan_09_M.L.D.C
>>>
```

PRACTICAL NO.: 7

Finite state automata [A]

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

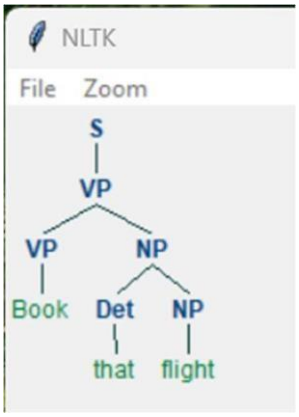
CODE:

nlp7a.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Pytho

File Edit Format Run Options Window Help

```
import nltk
from nltk import tokenize
grammar1 = nltk.CFG.fromstring("""
S -> VP
VP -> VP NP
NP -> Det NP
Det -> 'that'
NP -> singular Noun
NP -> '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()
print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:




```
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp7a.py =
['Book', 'that', 'flight']
(S (VP (VP Book) (NP (Det that) (NP flight))))
Uzma Khan_09_M.L.D.C
>>
```

[B] Accept the input string with Regular expression of Finite Automaton: 101+.

CODE:

nlp7b.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp7b.py (3.11.2)

File Edit Format Run Options Window Help

```
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 index
    if s[0]=='1':
        if s[1]=='0':
            if s[2]=='1':
# After index 2 only "1" can appear. Therefore break the process if any other character is detected
                for i in range(3,len(s)):
                    if s[i]!='1':
                        return "Rejected"
                    return "Accepted" # if all 4 nested if true
                return "Rejected" # else of 3rd if
            return "Rejected" # else of 2nd if
        return "Rejected" # else of 1st if
inputs=['1','10101','101','10111','01010','100','','10111101','1011111']
for i in inputs:
    print(FA(i))
print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

```
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp7b.py =
Rejected
Rejected
Rejected
Accepted
None
Rejected
Rejected
Accepted
Accepted
Uzma Khan_09_M.L.D.C
>>>
```

[C] Accept the input string with Regular expression of FA: (a+b)*bba.

CODE:

OUTPUT:

```
>>>
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp7c.py =
Rejected
Rejected
Rejected
Rejected
Rejected
Rejected
None
Uzma Khan_09_M.L.D.C
>>>
```

[D] Implementation of Deductive Chart Parsing using context free grammar and a given sentence.

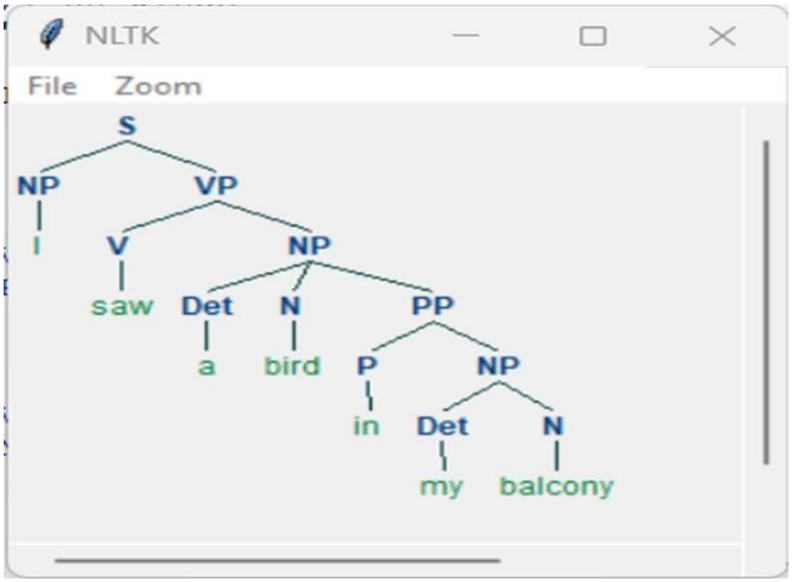
CODE:

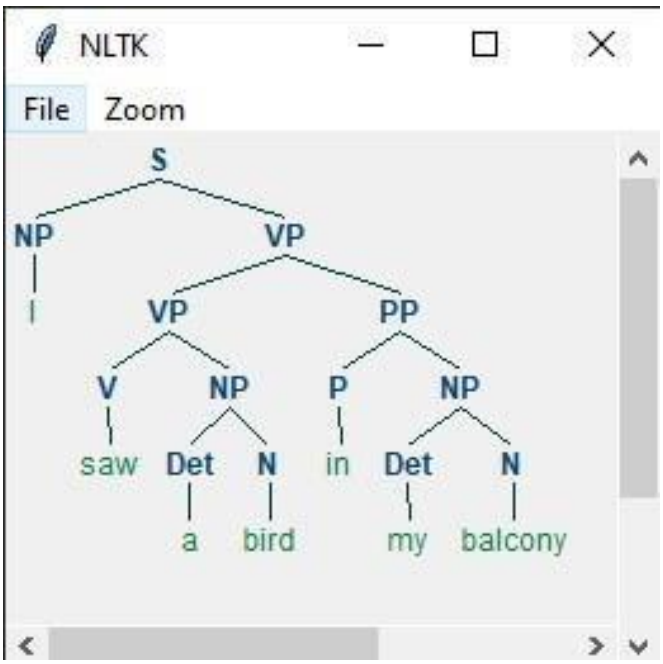
nlp7d.py - C:\Users\Uzma Khan\AppData\Local\Programs\Python\Python311\nlp7d.py (3.11...

File Edit Format Run Options Window Help

```
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()
print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:






```
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp7d.py =
['I', 'saw', 'a', 'bird', 'in', 'my', 'balcony']
(S
  (NP I)
  (VP
    (VP (V saw) (NP (Det a) (N bird)))
    (PP (P in) (NP (Det my) (N balcony)))))
(S
  (NP I)
  (VP
    (V saw)
    (NP (Det a) (N bird) (PP (P in) (NP (Det my) (N balcony))))))
Uzma Khan_09_M.L.D.C
>>
```

PRACTICAL NO.: 8

Study PorterStemmer, LancasterStemmer, RegexpStemmer, SnowballStemmer Study WordNetLemmatizer.

CODE:

PorterStemmer:

nlp8.py - C:/Users/Uzma Khan/AppData/Local/Programs,

File Edit Format Run Options Window Help

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

print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

```
== RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp8.py =
write
Uzma Khan_09_M.L.D.C
>>>
```

CODE:

LancasterStemmer

nlp8.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/

File Edit Format Run Options Window Help

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

print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

```
== RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp8.py =
writ
Uzma Khan_09_M.L.D.C
>>>
```

CODE:

RegexpStemmer:

nlp8.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nl

File Edit Format Run Options Window Help

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

print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

```
== RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp8.py =
writ
Uzma Khan_09_M.L.D.C
>
```

CODE:

SnowballStemmer:

nlp8.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Pyt

File Edit Format Run Options Window Help

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

print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

```
== RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp8.py =
write
Uzma Khan_09_M.L.D.C
```

CODE:

WordNetLemmatizer:

nlp8.py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp8.py (3.

File Edit Format Run Options Window Help

```
#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"))

print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

```
== RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp8.py =
word :  lemma
rocks : rock
corpora : corpus
better : good
Uzma Khan_09_M.L.D.C
```

PRACTICAL NO.: 9

Implement Naive Bayes classifier

CODE:

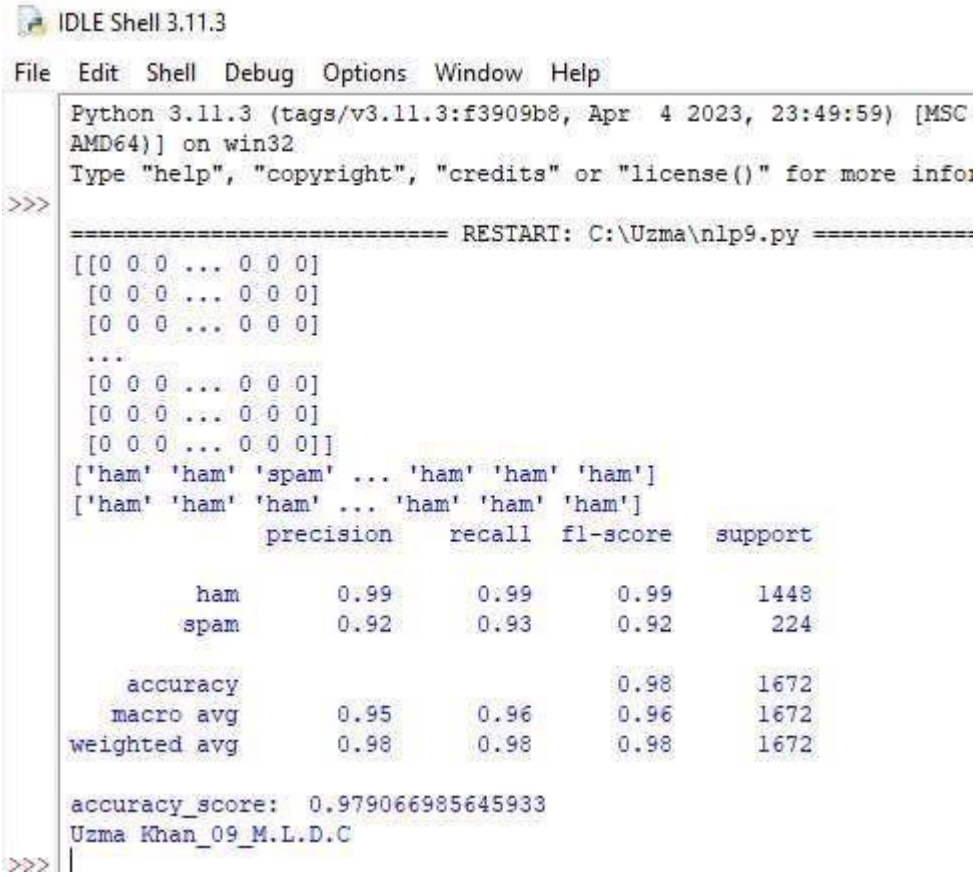
nlp9.py - C:\Uzma\nlp9.py (3.11.3)

File Edit Format Run Options Window Help

```
import pandas as pd
import 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'].values
print(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 Model
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.metrics import accuracy_score
print(classification_report(y_test,y_pred))
print("accuracy_score: ",accuracy_score(y_test,y_pred))

print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:


IDLE Shell 3.11.3
File Edit Shell Debug Options Window Help
Python 3.11.3 (tags/v3.11.3:f3909b8, Apr 4 2023, 23:49:59) [MSC
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more infor
>>>
========== RESTART: C:\Uzma\nlp9.py ==========
[[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]]
['ham' 'ham' 'spam' ... 'ham' 'ham' 'ham']
['ham' 'ham' 'ham' ... 'ham' 'ham' 'ham']
precision recall f1-score support
ham 0.99 0.99 0.99 1448
spam 0.92 0.93 0.92 224
accuracy 0.98 1672
macro avg 0.95 0.96 0.96 1672
weighted avg 0.98 0.98 0.98 1672
accuracy_score: 0.979066985645933
Uzma Khan_09_M.L.D.C
>>>

PRACTICAL NO.: 10

[A] Speech Tagging:

i) Speech tagging using spacy:

CODE:

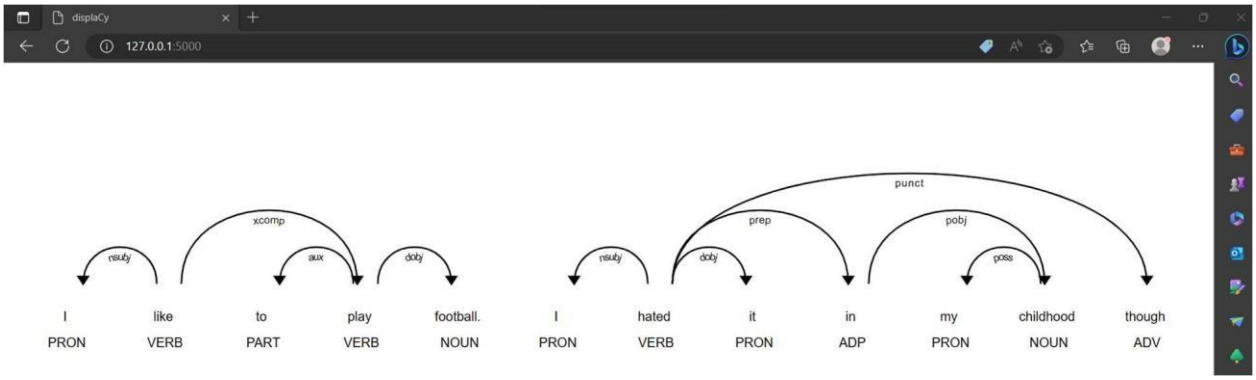
nlp10a(i).py - C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp10a(i).py (3.11.2)

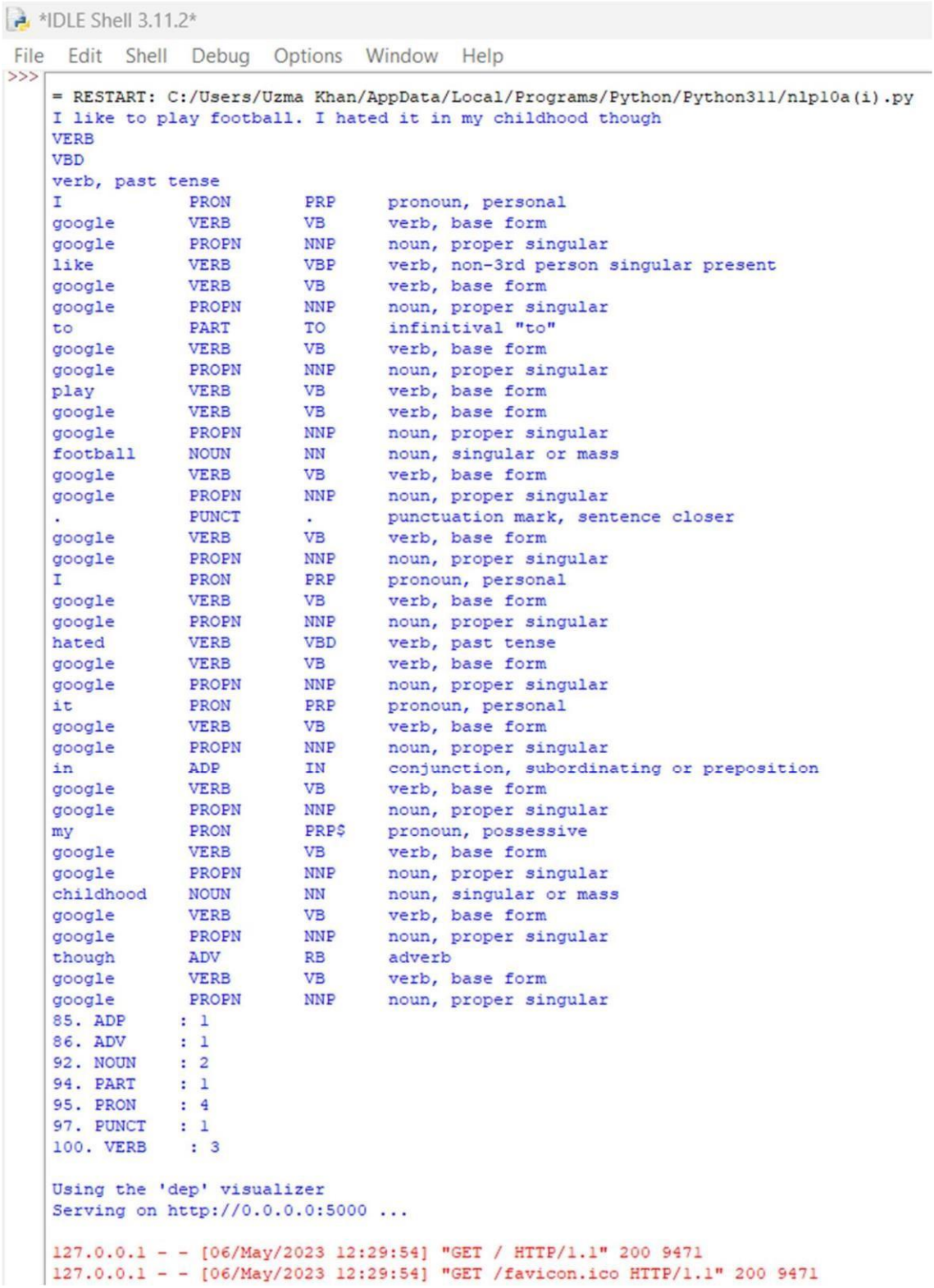
File Edit Format Run Options Window Help

```
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 Tags
from 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})

print("Uzma_Khan_09_M.L.D.C")
```

OUTPUT:




IDLE Shell 3.11.2
File Edit Shell Debug Options Window Help
>>>
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp10a(i).py
I like to play football. I hated it in my childhood though
VERB
VBD
verb, past tense
I PRON PRP pronoun, personal
google VERB VB verb, base form
google PROPN NNP noun, proper singular
like VERB VBP verb, non-3rd person singular present
google VERB VB verb, base form
google PROPN NNP noun, proper singular
to PART TO infinitival "to"
google VERB VB verb, base form
google PROPN NNP noun, proper singular
play VERB VB verb, base form
google VERB VB verb, base form
google PROPN NNP noun, proper singular
football NOUN NN noun, singular or mass
google VERB VB verb, base form
google PROPN NNP noun, proper singular
. PUNCT . punctuation mark, sentence closer
google VERB VB verb, base form
google PROPN NNP noun, proper singular
I PRON PRP pronoun, personal
google VERB VB verb, base form
google PROPN NNP noun, proper singular
hated VERB VBD verb, past tense
google VERB VB verb, base form
google PROPN NNP noun, proper singular
it PRON PRP pronoun, personal
google VERB VB verb, base form
google PROPN NNP noun, proper singular
in ADP IN conjunction, subordinating or preposition
google VERB VB verb, base form
google PROPN NNP noun, proper singular
my PRON PRP$ pronoun, possessive
google VERB VB verb, base form
google PROPN NNP noun, proper singular
childhood NOUN NN noun, singular or mass
google VERB VB verb, base form
google PROPN NNP noun, proper singular
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 ...
127.0.0.1 - - [06/May/2023 12:29:54] "GET / HTTP/1.1" 200 9471
127.0.0.1 - - [06/May/2023 12:29:54] "GET /favicon.ico HTTP/1.1" 200 9471

ii) Speech tagging using nltk:

CODE:

10a(ii).py - C:\Users\HP\AppData\Local\Programs\Python\Python39\10a(ii)

File Edit Format Run Options Window Help

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

train_text = state_union.raw("2005-GWBush.txt")
sample_text = state_union.raw("2006-GWBush.txt")

custom_sent_tokenizer = PunktSentenceTokenizer(train_text)

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()

print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

```
==== RESTART: C:\Users\HP\AppData\Local\Programs\Python\Python39\10a(ii).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'), ('.', '.')]
Uzma Khan_09_M.L.D.C
>>>
```

[B] Statistical parsing:

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

CODE:

10b(i).py - C:/Uzma/10b(i).py (3.9.5)

File Edit Format Run Options Window Help

```
import nltk
import nltk.parse.viterbi
import 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)
print("Uzma khan_09_M.L.D.C")
```

OUTPUT:

IDLE Shell 3.9.5

File Edit Shell Debug Options Window Help

```
Python 3.9.5 (tags/v3.9.5:0a7dcbd, May  3 2021, 17:27:52) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========================== RESTART: C:/Uzma/10b(i).py ==========================
give NP: advertisers / NP: discounts for maintaining or increasing ad sp...
give NP: only French history questions / PP-DTV: to students in a Europe...
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: holders / NP: the right , but not the obligation , to buy a cal...
gave NP: Mr. Thomas / NP: only a `` qualified '' rating , rather than ``...
Uzma khan_09_M.L.D.C
>>>
```

ii) probabilistic parser.

CODE:

10b(ii).py - C:/Uzma/10b(ii).py (3.9.5)

File Edit Format Run Options Window Help

```
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)

print("Uzma khan_09_M.L.D.C")
```

OUTPUT:

```
>>>
========================== RESTART: C:/Uzma/10b(ii).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)
Uzma khan_09_M.L.D.C
>>>
```

[C] Malt parsing:

Parse a sentence and draw a tree using malt parsing.

CODE:

nlp10c.py - C:/Users/Uzma Khan/Desktop/NLP/nlp10c.py (3.7.4)

File Edit Format Run Options Window Help

```
from nltk.parse import malt
mp = malt.MaltParser('maltparser-1.7.2', 'engmalt.linear-1.7.mco')#file
t = mp.parse_one('I saw a bird from my window.'.split()).tree()
print(t)
t.draw()
print("Uzma Khan_09_M.L.D.C")
```

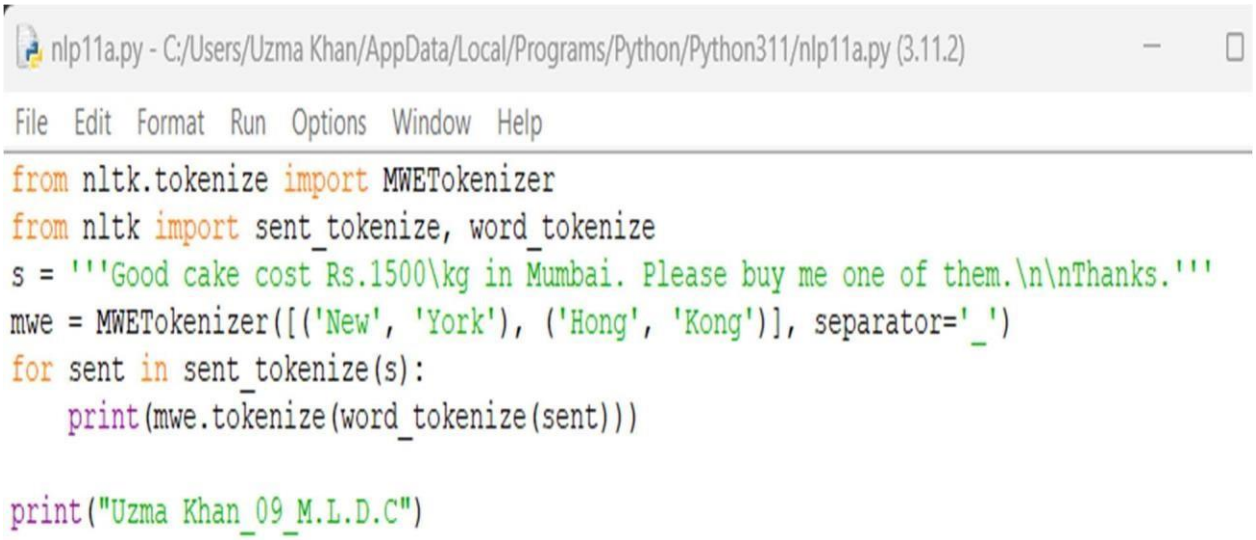
OUTPUT:


```
============= RESTART: C:/Users/Uzma Khan/Desktop/NLP/nlp10c.py =============
(saw I (bird a (from (window. my))))
Uzma Khan_09_M.L.D.C
>>>
```

PRACTICAL NO.: 11

[A] Multiword Expressions in NLP.

CODE:



OUTPUT:

```
= RESTART: C:/Users/Uzma Khan/AppData/Local/Programs/Python/Python311/nlp11a.py
['Good', 'cake', 'cost', 'Rs.1500\\kg', 'in', 'Mumbai', '.']
['Please', 'buy', 'me', 'one', 'of', 'them', '.']
['Thanks', '.']
Uzma Khan_09_M.L.D.C
```

[B] Normalized Web Distance and Word Similarity.

CODE:

nlp11b.py - C:/Uzma/nlp11b.py (3.9.5)

File Edit Format Run Options Window Help

```
import numpy as np
import 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,
        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))

print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

```
========================== RESTART: C:/Uzma/nlp11b.py ==========================

Warning (from warnings module):
  File "C:\Users\HP\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn
\cluster\_agglomerative.py", line 983
    warnings.warn(
FutureWarning: Attribute `affinity` was deprecated in version 1.2 and will be remove
d in 1.4. Use `metric` instead
['reliance', 'reliance', 'reliance', 'reliance', 'reliance', 'reliance', 'mumbai', '
mumbai', 'mumbai', 'mumbai', 'km trading', 'km trading', 'km trading', 'km trading',
'km trading']
Uzma Khan_09_M.L.D.C
>>>
```

[C] Word Sense Disambiguation

CODE:

nlp11c.py - C:/Uzma/nlp11c.py (3.9.5)

File Edit Format Run Options Window Help

```
#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))

print("Uzma Khan_09_M.L.D.C")
```

OUTPUT:

```
========== RESTART: C:/Uzma/nlp11c.py ==========
<bound method Synset.name of Synset('bank.n.01')>: <bound method Synset.definition o
f Synset('bank.n.01')>
<bound method Synset.name of Synset('set.n.01')>: <bound method Synset.definition of
Synset('set.n.01')>
<bound method Synset.name of Synset('put.v.01')>: <bound method Synset.definition of
Synset('put.v.01')>
Uzma Khan_09_M.L.D.C
>>>
```