Practical no 3

Name: Shivam Tawari
Roll no: A-58

Aim: Write a python program for tokenizer

Theory:

Tokenization:
It may be defined as the process of breaking up a piece of text into smallest parts, such as sentences and words. These smaller parts are called tokens.

NLTK Package:
nltk.tokenize is the package provided by NLTK module to achieve the process of tokenization.

Tokenizing sentences into words splitting the sentence into words or creating a list of words or creating a list of words from a string is an essential part of every text processing activity. Let us understand it with the help of various functions / modules provided by nltk tokenize package.

Word Punkt Tokenizer class:
An alterative word tokenizer that splits all punctuation into separate tokens.

Why it is required:
An obvious question that comes in our mind is that when we have word tokenizer then why do we need to count the average words in the sentence, how can we do this? For accomplishing this, we need both sentence tokenization and word tokenization.

The tokenization without NLTK would take hours and hours of coding with regular expressions.

Limitations:

One of the major issues with word tokens is dealing with out of vocabulary words (OOV). OOV words refer to the new words which are encountered at testing. These new words do not exist in the vocabulary.

Wordnet :
Wordnet is a lexical database for the English language. It groups English words into sets of ~~English words into sets~~ of synonyms called synsets, provides short definitions and usage examples, and records a number of relations among these Synonym sets or their members.

Conclusion: Hence, we have successfully programmed tokenizer in python.

Practical 3

Name: Shivam Tawari

Roll no: A-58

Shivam Tawari A-58

```
[3]  #1st Example

     import nltk
     nltk.download('punkt')
     from nltk.tokenize import word_tokenize
     word_tokenize('Raisoni.net provides high quality technical knowledge for free')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
['Raisoni.net',
 'provides',
 'high',
 'quality',
 'technical',
 'knowledge',
 'for',
 'free']
```

```
import nltk
from nltk.stem import PorterStemmer
from nltk.tokenize import word_tokenize
from nltk.corpus import wordnet

nltk.download('punkt')
nltk.download('wordnet')

ps = PorterStemmer()

sentence = "Raisoni.net provides high quality technical knowledge for free"
words = word_tokenize(sentence)

for w in words:
    syn = list()
    for synset in wordnet.synsets(w):
        for lemma in synset.lemmas():
            syn.append(lemma.name())

    print(w, " : ", ps.stem(w))
    print('Synonyms: ' + str(syn))
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
Raisoni.net  :  raisoni.net
```

```
[4]     print('Synonyms: ' + str(syn))
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
Raisoni.net  :  raisoni.net
Synonyms: []
provides  :  provid
Synonyms: ['supply', 'provide', 'render', 'furnish', 'provide', 'supply', 'ply', 'cater', 'provide', 'put_up
high  :  high
Synonyms: ['high', 'high', 'high', 'high', 'high', 'heights', 'senior_high_school', 'senior_high', 'high', '
quality  :  qualiti
Synonyms: ['quality', 'quality', 'caliber', 'calibre', 'quality', 'character', 'lineament', 'timbre', 'timbe
technical  :  technic
Synonyms: ['technical', 'technical_foul', 'technical', 'technical', 'proficient', 'technical', 'technical',
knowledge  :  knowledg
Synonyms: ['cognition', 'knowledge', 'noesis']
for  :  for
Synonyms: []
free  :  free
Synonyms: ['free', 'free_people', 'free', 'liberate', 'release', 'unloose', 'unloosen', 'loose', 'rid', 'fre
```