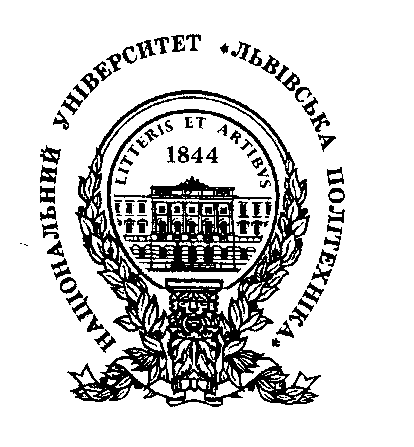
Міністерство освіти та науки України

Національний університет «Львівська Політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра САПР



Лабораторна робота №1

на тему:

ВИВЧЕННЯ БІБЛІОТЕКИ ПРИКЛАДНИХ ПРОГРАМ NLTK, ДЛЯ ОПРАЦЮВАННЯ ТЕКСТІВ ПРИРОДНОЮ МОВОЮ.

Виконав:

студент ПРЛм-12

Білик Д.Г.

Перевірив:

Дупак Б.П.

Львів 2015

**МЕТА РОБОТИ**

* Вивчення основ програмування на мові *Python*.
* Ознайомлення з контрольними структурами та класом FreqDist.

**КОРОТКІ ТЕОРЕТИЧНІ ВІДОМОСТІ**

Засоби керування логікою.

Програми, які згадувалися в лабораторній роботі №1, доволі прості, але дозволяють працювати з мовою та полегшити роботу людини шляхом її автоматизації. Основна особливість програмування – це спосібність програми приймати рішення від імені людини, виконуючи інструкції коли справджуються певні умови або послідовно обробляти текстові дані до тих пір поки не задоволена певна умова. Таким засобом керування поведінкою програми є контрольні структури.

Умовні вирази

Python підтримує широкий набір операторів для встановлення взаємозв’язків між змінними (значеннями). Звичайно ми використовуємо умовні оператори, як частину If операторів. Для перевірки властивостей окремих слів існує набір наступних функцій (Таблиця 2.).

Таблиця 2.

| Функція | Пояснення |
| --- | --- |
| s.startswith(t) | чи починається s з t |
| s.endswith(t) | чи закінчується s на t |
| t in s | Чи t міститься в s |
| s.islower() | Чи всі символи в s є малі |
| s.isupper() | Чи всі символи в s є великі |
| s.isalpha() | Чи всі символи в s є букви |
| s.isalnum() | Чи всі символи в s є букви і цифри |
| s.isdigit() | Чи всі символи в s є цифри |
| s.istitle() | Чи всі слова в s є з великої літери |

Клас FreqDist для простих статистичних досліджень

Для автоматичного визначення слів, які є найбільш інформативними для текстів певного жанру або певної тематики спочатку інтуітивно виникає думка побудувати частотний список або частотний розподіл. Частотний розподіл вказує на частоту з якою в тексті зустрічається кожне зі слів. Такий частотний список називають розподілом тому, що він вказує яким чином загальна кількість слів розподіляється між словниковими статями (оригінальні слова) в тексті. Враховуючи що побудова частотних розподілів часто необхідна при обробці природної мови в NLTK реалізовано окремий клас FreqDist в модулі nltk.probability . Застосуємо цей клас для знаходження 50 найчастотніших слів в тексті Moby Dick.

>>> fdist1 = FreqDist(text1)  #1

>>> fdist1  #2

<FreqDist with 260819 outcomes>

>>> vocabulary1 = fdist1.keys() #3

>>> vocabulary1[:50]  #4

[',', 'the', '.', 'of', 'and', 'a', 'to', ';', 'in', 'that', "'", '-',

'his', 'it', 'I', 's', 'is', 'he', 'with', 'was', 'as', '"', 'all', 'for',

'this', '!', 'at', 'by', 'but', 'not', '--', 'him', 'from', 'be', 'on',

'so', 'whale', 'one', 'you', 'had', 'have', 'there', 'But', 'or', 'were',

'now', 'which', '?', 'me', 'like']

>>> fdist1['whale']

906

>>>

При першому виклику FreqDist назва текста вказується як аргумент класа #1. Можна дізнатися загальну кількість слів, які були підраховані #2. Вираз keys() дозволяє встановити список оригінальних слів текста #3 і можна переглянути перші 50 з них #4. Серед цих 50 слів тільки одне дає певну інформацію про текст (whale) і це слово зустрічається в тексті 906 раз. Всі інші слова є не інформативними, або службовими. Для визначення, яку частину тексту займають ці службові слова знайдемо їх сумарну частоту побудувавши графік (Рис.1.) за допомогою fdist1.plot(50, cumulative=True)

**ТЕКСТИ ПРОГРАМ НА МОВІ PYTHON**

**Варіант 13**

3.1. Створіть змінну sentence і присвойте їй значення ‘she sells sea shells by the sea shore’ та напишіть фрагмент програми для виведення на екран всіх слів які починаються з ‘sh’.

sentence = 'she sells sea shells by the sea shore'

words = sentence.split()

for i in words:

if i.startswith('sh'):

print i



3.4. Напишіть програму, яка видаляє всі голосні зі стрічки, яка відповідає імені, по батькові та прізвищу студента. Програма повинна здійснювати наступну послідовність дій: створення початкової стрічки; створення стрічки, у якій буде зберігатися результат; for цикл для обробки стрічки символ за символом і запису неголосних символів в результуючу стрічку.

name1 = 'Bilyk Dmytro Haiyovich'

name2 = ''

for i in name1:

if 'a' not in i:

name2+=i

name1=''

for i in name2:

if 'o' not in i:

name1+=i

name2=''

for i in name1:

if 'i' not in i:

name2+=i

name1=''

for i in name2:

if 'u' not in i:

name1+=i

name2=''

for i in name1:

if 'y' not in i:

name2+=i

print name2



3.7. Виконати наступні приклади і пояснити чому отримані різні результати (різні значення змінних)

sorted(set([w.lower() for w in text1]))

sorted([w.lower() for w in set(text1)])

3.13. Перевірте виконання виразу set(sent3) < set(text1). Змініть аргументи функції. Результати поясніть.

import nltk

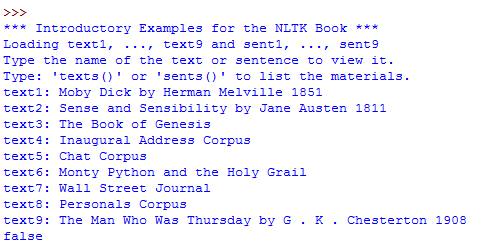
from nltk.book import\*

if set(sent3) < set(text1):

print 'true'

else:

print 'false'



Речення меньше за розміром ніж цілий текст.

3.11. Напишіть вираз для знаходження в тексті №6 всіх слів які відповідають наступним вимогам: закінчуються на ize; містять літеру z; містять послідовність літер pt; написані з великої літери . Результат представити, як список слів.

import nltk

from nltk.book import\*

result1 = sorted([w for w in set(text6) if w.endswith('ize')])

for word in result1:

print word

print '=========================================='

result2 = sorted([w for w in set(text6) if 'z' in w])

for word in result2:

print word

print '=========================================='

result3 = sorted([w for w in set(text6) if 'pt' in w])

for word in result3:

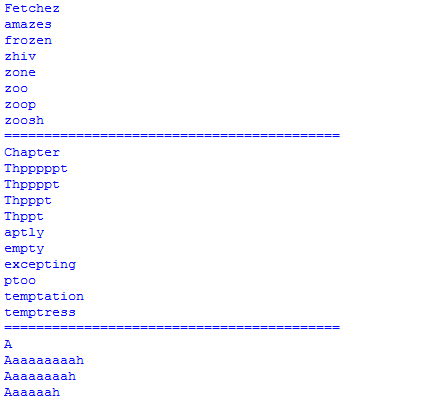
print word

print '=========================================='

result4 = sorted([w for w in set(text6) if w.istitle()])

for word in result4:

print word



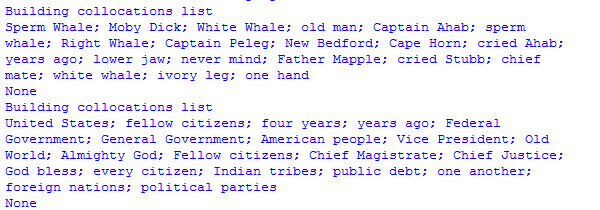
3.15. Побудуйте колокації для текстів №1 та №4. Результати порівняйте.

import nltk

from nltk.book import\*

print text1.collocations()

print text4.collocations()



**ВИСНОВОК**

На цій лабораторній роботі, ми вивчили основи програмування на мові Python та ознайомились з контрольними структурами та класом FreqDist.