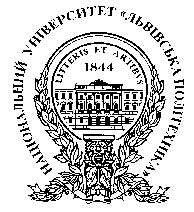
МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ

Національний Університет “Львівська політехніка”



Лабораторна робота № 2

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

***ОСНОВИ ПРОГРАМУВАННЯ НА МОВІ PYTHON(частина 2).***

Виконав:

студент групи ПРЛс-11

Форманюк А.А.

Прийняв:

асистент

Дупак Б.П.

Львів 2015

**Мета:** вивчити основи програмування на мові Python та 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.

***Засоби здійснення простого керування.***

Більшість мов програмування дозволяють виконання окремих блоків програми, коли використовуються умовні вирази або *if* оператори. В наступній програмі ми створили змінну *word*, яка містить значення *‘cat’* типу стрічка. *If*- оператор перевіряє умову чи довжина слова <5, чи ні. Якщо умова виразу справджується, то виконується тіло *if* оператора і виконується оператор print .

***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*.

***Колокації та біграми***.

Колокація це словосполучення яке зустрічається дуже часто. red wine –це колокація а the wine – ні . Характерною рисою колокацій є те що вони є стійкі до заміни одного зі слів на інше подібне за змістом (maroon wine). Для того щоб побудувати колокації спочатку потрібно побудувати на основі тексту пари слів, або біграми. Для цього можна використати функцію bigrams():

>>> bigrams(['more', 'is', 'said', 'than', 'done'])

[('more', 'is'), ('is', 'said'), ('said', 'than'), ('than', 'done')]

>>>

**Задачі**

1. Створіть змінну *sentence* і присвойте їй значення *‘she sells sea shells by the sea shore’* та напишіть фрагмент програми, яка генерує нову стрічку додаючи *‘like’* перед кожним зі слів , яке починається з *‘se’*.

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

*sentence = sentence.split()*

*a = ''*

*for w in sentence:*

*if w.startswith('se'):*

*a = a+ ' like ' + str(w)*

*else:*

*a = a + ' ' + str(w)*

*print a*

*>>>*

*she like sells like sea shells by the like sea shore*

1. Перегляньте результати виконання умовних виразів: ’row’ in ’brown’ та ’row’ in [’ brown’, ’cow’]. Напишіть програму для перевірки наявності в стрічці *sent=’ ’colorless green ideas sleep furiously’* окремих слів та підстрічок.

\*

>>> word = 'brown'

>>> 'row' in word

True

\*

>>> msg = 'brown cow'

>>> msg = msg.split()

>>> msg

['brown', 'cow']

>>> 'row' in msg

False

\*

>>> sent = 'colorless green ideas sleep furiously'

>>> sent = sent.split()

>>> sent

['colorless', 'green', 'ideas', 'sleep', 'furiously']

>>> 'sleep' in sent

True

*>>> sent*

*['colorless', 'green', 'ideas', 'sleep', 'furiously']*

*>>> for word in sent:*

*if word.startswith('c'):*

*print word*

*colorless*

*>>> sent*

*['colorless', 'green', 'ideas', 'sleep', 'furiously']*

*>>> 'green'.endswith('en')*

*True*

*>>> 'furiously'.startswith('f')*

*True*

1. Виконати наступні приклади і пояснити різницю між ними

w.isupper()

not w.islower()

*>>> sent = 'THE Science minister LORD Sainsburu'*

*>>> sent1 = sent.split()*

*>>> sent*

*'THE Science minister LORD Sainsburu'*

*>>> for word in sent1:*

*[w.isupper() for w in sent1]*

*[True, False, False, True, False]*

*>>> sent*

*'THE Science minister LORD Sainsburu'*

*>>> for word in sent1:*

*[not w.islower() for w in sent1]*

*[True, True, False, True, True]*

1. Використайте вираз sum([len(w) for w in text1]) для знаходження середньої довжини слів в тексті.

*>>> import nltk*

*>>> from nltk.book import(text1)*

*\*\*\* 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*

*>>> sum = sum([len(w) for w in text1])*

*>>> l = len(text1)*

*>>> print 'amount of letters' , sum*

*amount of letters 999044*

*>>> print 'amount of words', l*

*amount of words 260819*

*>>> print 'average amount', sum/float(l)*

*average amount 3.83041112802*

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

*>>> import nltk*

*>>> 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*

*>>> vyraz = set(sent3) < set(text1)*

*>>> print vyraz*

*True*

*>>> print sent3*

*['In', 'the', 'beginning', 'God', 'created', 'the', 'heaven', 'and', 'the', 'earth', '.']*

*>>> print text1*

*<Text: Moby Dick by Herman Melville 1851>*

**Зі зміною аргумента фунуції**

*>>> vyraz = set(sent3) > set(text1)*

*>>> print vyraz*

*False*

1. Побудуйте колокації для текстів №1 та №2. Результати порівняйте.

*>>> import nltk*

*>>> 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*

*text1.collocations()*

*Building collocations list*

*Sperm Whale; Moby Dick; White Whale; old man; Captain Ahab; sperm*

*whale; Right Whale; Captain Peleg; New Bedford; Cape Horn; cried Ahab;*

*years ago; lower jaw; never mind; Father Mapple; cried Stubb; chief*

*mate; white whale; ivory leg; one hand*

*>>> text2.collocations()*

*Building collocations list*

*Colonel Brandon; Sir John; Lady Middleton; Miss Dashwood; every thing;*

*thousand pounds; dare say; Miss Steeles; said Elinor; Miss Steele;*

*every body; John Dashwood; great deal; Harley Street; Berkeley Street;*

*Miss Dashwoods; young man; Combe Magna; every day; next morning*

*>>>*

*>>> text1.collocations() > text2.collocations()*

*False*

**Висновок:** на лабораторній роботі я освоїв основи програмування на мові Python та ознайомився з контрольними структурами та класом FreqDist.