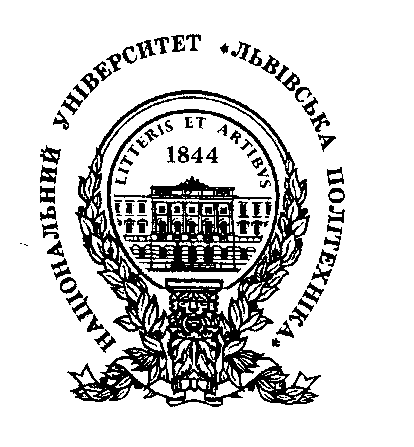
Міністерство освіти та науки України

Національний університет «Львівська Політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра САПР



Лабораторна робота №8

на тему:

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

Виконав:

студент ПРЛм-12

Білик Д.Г.

Перевірив:

Дупак Б.П.

Львів 2015

**МЕТА РОБОТИ**

* Вивчення основ програмування на мові *Python*.
* Вивчення основ структурного програмування мовою *Python*.
* Повторення та закріплення знань отриманих при виконанні попередніх лабораторних робіт.
* Покращення загальних навичок у програмуванні.

**КОРОТКІ ТЕОРЕТИЧНІ ВІДОМОСТІ**

## Використання регулярних виразів для виявлення слів за заданими шаблонами.

## *1.   Складні випадки використання функцій*

### *1.1 Функція, як аргумент*

Аргументи функцій, які розглядалися в попередніх лабораторних роботах, були простими об’єктами, такими як стрічка, або структурованими, такими як список. В Python аргументом функції також може бути і інша функція.В наступному прикладі показано, яким чином вбудована функція len() або розроблена функція last\_letter() передаються, як аргументи іншій функції:

|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | |  | **>>> sent = ['Take', 'care', 'of', 'the', 'sense', ',', 'and', 'the',**  **... 'sounds', 'will', 'take', 'care', 'of', 'themselves', '.']**  **>>> def extract\_property(prop):**  **... return [prop(word) for word in sent]**  **...**  **>>> extract\_property(len)**  **[4, 4, 2, 3, 5, 1, 3, 3, 6, 4, 4, 4, 2, 10, 1]**  **>>> def last\_letter(word):**  **... return word[-1]**  **>>> extract\_property(last\_letter)**  **['e', 'e', 'f', 'e', 'e', ',', 'd', 'e', 's', 'l', 'e', 'e', 'f', 's', '.']** | |

Об’єкти len та last\_letter передаються у функцію як списки та словники (тип даних словник буде розглянуто пізніше). Зауважимо, що дужки після імені функції використовуються тільки при її виклику, а якщо функція трактується, як об’єкт (аргумент іншої функції) то дужки опускаються.

### *1.2 Функції накопичення*

Виконання функцій накопичення починається з виділення певного об’єму пям’яті. Під час здійснення ітерацій над вхідними даними, цей об’єм заповнюється і тільки після цього функція повертає результат (велику структуру або узагальнений результат). Стандартний спосіб реалізувати таку функцію це створити пустий список, в списку накопичити дані і повернути цей список , як показано у функції search1()(наступний приклад).

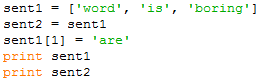
|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | |  | **def search1(substring, words):**  **result = []**  **for word in words:**  **if substring in word:**  **result.append(word)**  **return result**  **def search2(substring, words):**  **for word in words:**  **if substring in word:**  **yield word**  **print "search1:"**  **for item in search1('zz', nltk.corpus.brown.words()):**  **print item**  **print "search2:"**  **for item in search2('zz', nltk.corpus.brown.words()):**  **print item** | |
|  |

Функція search2() – це генератор. Після першого виклику цієї функції, вона виконується аж до yield твердження і зупиняється. Програма, з якої викликалася ця функція, отримує перше слово і виконує всі необхідні дії. Тільки коли програма готова до обробки наступного слова, виконання функції продовжується з того самого місця в якому вона була зупинена і функція знову виконується аж до yield твердження. Такий підхід є типовим і більш ефективним, оскільки функція генерує дані тільки тоді коли їх потребує основна програма і не потребує виділення додаткового об’єму памяті для збереження результатів.

**ТЕКСТИ ПРОГРАМ НА МОВІ PYTHON**

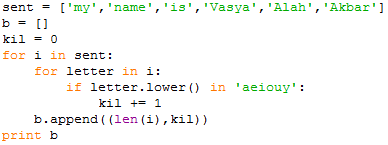
**Варіант 13**

3.1. Створити список слів і зберегти їх в змінній sent1. Здійснити операцію присвоювання sent2 = sent1. Змінити один з елементів в sent1 і перевірити чи змінився sent2. Результат письмово пояснити.





3.6. Написати програму для створення двовимірного масиву word\_vowels елементами якого є набори. Програма повинна обробити список слів і додати результати обробки до word\_vowels[l][v] де l – довжина слова, v – кількість голосних у слові.

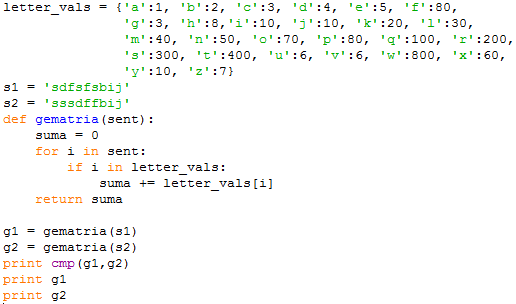




3.9. Гематрія – метод виявлення прихованого змісту слів на основі порівняння чисел, які відповідають словам. Слова з однаковими числами мають однаковий зміст. Число слова визначається сумуванням чисел, як відповідають його літерам. Написати функцію gematria() для сумування числових значень літер в слові згідно наступних значень letter\_vals:

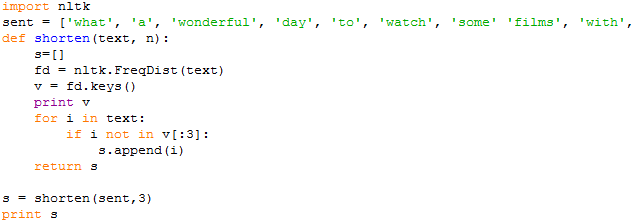
**>>> letter\_vals = {'a':1, 'b':2, 'c':3, 'd':4, 'e':5, 'f':80, 'g':3, 'h':8,'i':10, 'j':10, 'k':20, 'l':30, 'm':40, 'n':50, 'o':70, 'p':80, 'q':100, 'r':200, 's':300, 't':400, 'u':6, 'v':6, 'w':800, 'x':60, 'y':10, 'z':7}**

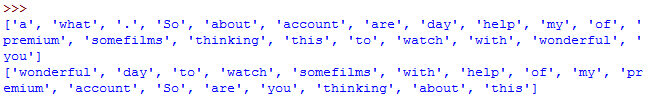
|  |
| --- |
|  |



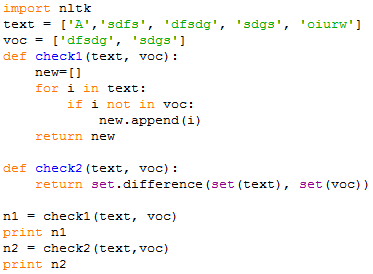


3.12. Написати функцію shorten(text, n) обробки тексту , для вилучення *n* найбільш частотних слів в тексті. Яким чином змінилась читабельність тексту, після вилучення цих слів?





3.15. Написати функцію, яка приймає текст і словник, як аргументи і повертає набір слів, які є у тексті але відсутні у словнику. Аргументи повинні бути представлені, як списки стрічок. Чи може функція мати один рядок при використанні set.difference()?





**ВИСНОВОК**

На цій лабораторній роботі, ми вивчили основи програмування на мові Python, основи структурного програмування мовою Python, повторили та закріпили знання отримані при виконанні попередніх лабораторних робіт та покращили загальні навички у програмуванні.