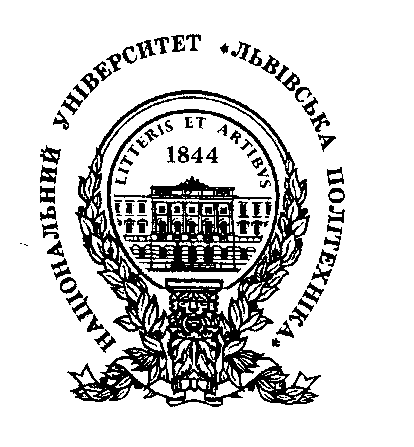
Міністерство освіти та науки України

Національний університет «Львівська Політехніка»

Інститут комп’ютерних наук та інформаційних технологій

Кафедра САПР



Лабораторна робота №7

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**

## *1. Основи програмування*

### *1.1 Присвоювання*

Присвоювання – найпростіше поняття програмування, але навіть і йому властиві певні тонкощі. Розглянемо приклад:

|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | |  | **>>> foo = 'Monty'**  **>>> bar = foo**  **#1**  **>>> foo = 'Python'**  **#2**  **>>> bar**  **'Monty'** | |

Коли записується bar = foo (рядок програми #1), то значення змінної foo (стрічка 'Monty') присвоюється змінній bar. Таким чином, bar є копією foo. Якщо змінити foo на нову стрічку 'Python' (рядок програми #2), то змінної bar ці зміни не торкнулася.

За допомогою операції присвоєння відбувається копіювання значення виразу іншій змінній, хоча насправді, значення структурованого об’єкту, наприклад такого, як список, це є посилання на цей об’єкт. В наступному прикладі (рядок програми #1) відбувається присвоєння посилання на значення змінної foo новій змінній bar. Якщо відбуваються зміни в foo (рядок програми #2) то ці зміни також торкаються і bar.

|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | |  | **>>> foo = ['Monty', 'Python']**  **>>> bar = foo** **#1**  **>>> foo[1] = 'Bodkin'** **#2**  **>>> bar**  **['Monty', 'Bodkin']** | |

### *1.2 Порівняння*

Python підтримує два способи порівняння. Оператор is перевіряє об’єкти на ідентичність. Створивши список з декількох копій одного і того самого об’єкту не складно переконатися, що елементи цього списку не тільки ідентичні, згідно ==, але і є одним і тим самим об’єктом:

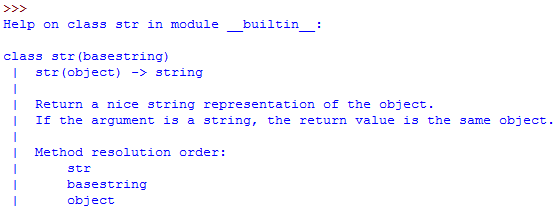
|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | |  | **>>> size = 5**  **>>> python = ['Python']**  **>>> snake\_nest = [python] \* size**  **>>> snake\_nest[0] == snake\_nest[1] == snake\_nest[2] == snake\_nest[3] == snake\_nest[4]**  **True**  **>>> snake\_nest[0] is snake\_nest[1] is snake\_nest[2] is snake\_nest[3] is snake\_nest[4]**  **True**  **>>> [id(snake) for snake in snake\_nest]**  **[13275984, 13275984, 13275984, 13275984, 13275984]** | |

**ТЕКСТИ ПРОГРАМ НА МОВІ PYTHON**

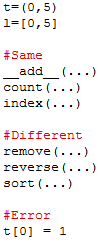
**Варіант 13**

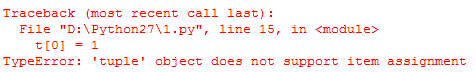
3.1. Знайти в Python's help додаткову інформацію про послідовності. В інтерпретаторі, набрати по черзі help(str), help(list), та help(tuple). На екрані буде відображено повний список функцій властивих кожному з типів. Деякі функції мають спеціальні імена з подвійними підкреслюваннями. Кожній такій функції відповідає і інший запис показаний в документації. Наприклад x.\_\_getitem\_\_(y) відповідає x[y]





3.2. Знайти три операції, які можна здійснювати і зі списками та із кортежами. Знайти три операції, які не можна здійснювати над кортежами. Знайдіть коли використання списку замість кортежу приводить до Python помилки.





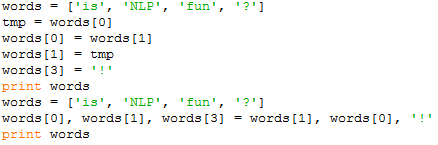
3.3. Яким чином можна створити кортеж з одного елемента. Продемонструвати два різні способи.

|  |
| --- |
|  |



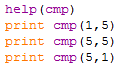


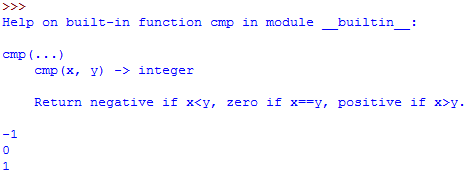
3.4. Створити список words = ['is', 'NLP', 'fun', '?']. Використовуючи операції присвоювання подібні до words[1] = words[2] та тимчасову змінну tmp перетворити цей список в список ['NLP', 'is', 'fun', '!']. Здійснити аналогічні перетворення використовуючи присвоювання в кортежах.



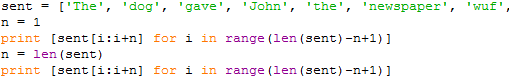


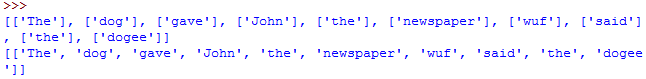
3.5. Прочитати про вбудовану функцію здійснення порівнянь cmp, набравши help(cmp). Продемонструвати чим поведінка цієї функції відрізняється від поведінки операторів порівняння.



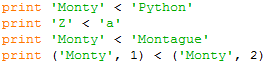


3.6. Написати програму для коректного виділення в тексті n-грамів з врахуванням граничних випадків: *n* = 1, та *n* = len(sent)?



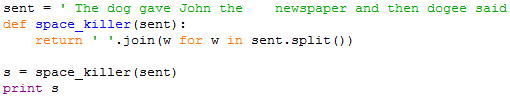


3.7. Використати оператори нерівності для порівняння стрічок, наприклад. 'Monty' < 'Python'. Що станеться, якщо виконати 'Z' < 'a'? Порівняти стрічки,як мають однаковий префікс, наприклад 'Monty' < 'Montague'. Спробувати порівняти структуровані об’єкти ,наприклад. ('Monty', 1) < ('Monty', 2). Чи отримали очікувані результати?



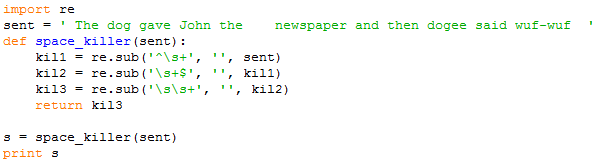


3.8. Написати програму видалення пробілів на початку і в кінці стрічки та для видалення зайвих пробілів між словами. Використовувати split() та join(). Оформити у вигляді функції. Функція повинна містити повну стрічку документування.



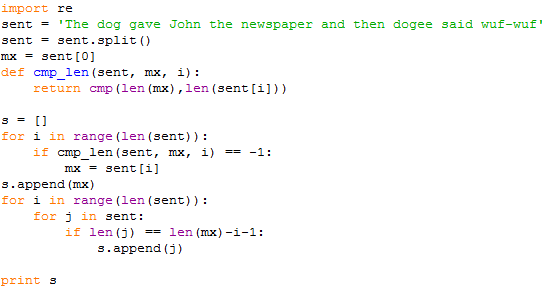


3.9. Написати програму видалення пробілів на початку і в кінці стрічки та для видалення зайвих пробілів між словами. Використовувати re.sub() . Оформити у вигляді функції. Функція повинна містити повну стрічку документування.





3.10. Написати програму сортування слів за їх довжиною. Визначити допоміжну функцію cmp\_len, яка буде використовувати функцію cmp для порівняння довжин слів. Функція повинна містити повну стрічку документування.





**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.