МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”
ІНСТИТУТ КОМП’ЮТЕРНИХ НАУК ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра “Системи автоматизованого проектування”

Звіт

до лабораторної роботи №8

на тему: ВИВЧЕННЯ БІБЛІОТЕКИ ПРИКЛАДНИХ ПРОГРАМ NLTK, ДЛЯ ОПРАЦЮВАННЯ ТЕКСТІВ ПРИРОДНОЮ МОВОЮ.СТРУКТУРНЕ ПРОГРАМУВАННЯ МОВОЮ PYTHON.
ОСНОВИ ПРОГРАМУВАННЯ НА МОВІ PYTHON(частина 2) ”

з дисципліни “Комп’ютерна лінгвістика”

Виконала:

студентка групи ПРЛм-11

Неїжмак О.А.

Прийняв:

викладач

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.

Варіант №10

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.

```
>>> sent1=['spring','summer','autumn']
>>> sent2=sent1
>>> print sent2
['spring', 'summer', 'autumn']
>>> sent1='winter'
>>> print sent1
winter
>>> print sent2
['spring', 'summer', 'autumn']
>>>
```

Через те, що ми спершу прирівняли sent1 і sent2, а потім провели зміни, тому sent2 не змінився.

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

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

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}
>>> def gematria(word):
        sum=0
        for letter in word:
            sum+=letter_vals[letter]
        return sum

>>> print gematria('morning')
423
```

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?

```
>>> import nltk
>>> text='Pentagon officials are also concerned about any potential run-in with
Russian aircraft in Syrian airspace while the U.S.-led coalition continues its m
ilitary campaign against ISIS. In addition, the U.S. is supporting rebels in fig
hting the terror group, also known as Islamic State.'
>>> def shorten(text,n):
        tokens=nltk.word_tokenize(text)
        fdist=nltk.FreqDist([w.lower()for w in tokens])
        vocab=fdist.keys()[:n]
        b=[]
        for i in tokens:
            if i not in vocab:
                b+=[i]
                import string
                print vocab
                return string.join(b)


>>> print shorten(text,8)
['in', 'the', ',', 'also', '.', 'about', 'addition', 'against']
Pentagon
>>>
```

16.Імпортувати функцію `itemgetter()` модуля `operator` зі стандартної бібліотеки Python(`fromoperatorimportitemgetter`). Створити список `words`, який містить декілька слів. Спробувати виконати: `sorted(words, key=itemgetter(1))`, та `sorted(words, key=itemgetter(-1))`. Пояснити письмово роботу функції `itemgetter()`.

```
import nltk
from operator import itemgetter
words='september', 'oktober', 'november', 'december'
print sorted(words, key=itemgetter(1))
print sorted(words, key=itemgetter(-1))
print sorted(words, key=itemgetter(3))
```

```
>>>
['september', 'december', 'oktober', 'november']
['september', 'oktober', 'november', 'december']
['november', 'december', 'oktober', 'september']
>>> help(itemgetter)
Help on class itemgetter in module operator:

class itemgetter(__builtin__.object)
 |  itemgetter(item, ...) --> itemgetter object
 |
 |  Return a callable object that fetches the given item(s) from its operand.
 |  After, f=itemgetter(2), the call f(r) returns r[2].
 |  After, g=itemgetter(2,5,3), the call g(r) returns (r[2], r[5], r[3])
 |
 |  Methods defined here:
 |
 |  __call__(...)
 |      x.__call__(...) <==> x(...)
 |
```

17. В NLTKреалізовано алгоритм Левінштейна для порівняння стрічок. Спробуйте скористатись цим модулем `nltk.edit_dist()`. Яким чином в цьому модулі використовується динамічне програмування? Який підхід використовується знизу-вверх чи зверху-вниз? Пояснити письмово.

```
>>> import nltk
>>> print nltk.edit_distance('meet','met')
1
>>> print nltk.edit_distance('butter','butterfly')
3
>>>
```

Відстань Левенштайна обраховує відстань між двома стрічками, використовуючи підхід знизу-вверх.

Висновок:

на цій лабораторній роботі я ознайомилася з явищем гематрії, та навчилася обраховувати відстань Левенштайна у мові програмування Python.