

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»
Кафедра САПР

ЗВІТ

до лабораторної роботи № 4

на тему:

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

ДОСТУП ТА РОБОТА З ЛЕКСИЧНИМИ РЕСУРСАМИ

з дисципліни “Комп’ютерна лінгвістика”

Виконала:

Студентка групи ПРЛм-12

Рибчак Х. В.

Перевірив:

Асистент кафедри САПР

Дупак Б. П.

Львів 2015

МЕТА РОБОТИ

Вивчення основ програмування на мові Python. Вивчення методів доступу до кода та роботи з лексичними ресурсами. Семантичний словник англійської мови WordNet.

КОРОТКІ ТЕОРЕТИЧНІ ВІДОМОСТІ

Поняття функції та модуля

При програмуванні часто необхідно частину програми виконати (використати) декілька разів. Наприклад, потрібно написати програму, яка здійснює утворення множини з однини іменників і вона буде виконуватись в різних місцях програми. Швидше ніж повторювати той самий код декілька разів і більш ефективно і надійно організувати цю роботу через функцію. Функція - це програмна конструкція, яку можна викликати з одним або більше вхідними параметрами, і отримувати результат на виході. Визначаємо функцію, використовуючи ключове слово *def* далі потрібно дати назву функції і визначити вхідні параметри, після двокрапки записується тіло функції. Ключове слово *return* використовується для відображення значення, яке ми хочемо отримати на виході функції.

Множина змінних і функцій збережених у файлі називаються в Python – модулем. Множина пов'язаних між собою модулів називають – пакетом. Програма обробки корпусу Brown це є приклад модуля, а множина програм для роботи зі всіма корпусами це є приклад пакету. NLTK це множина пакетів, яку називають бібліотекою.

Генерація випадкового тексту за допомогою біграмів

Умовний частотний розподіл можна використати для побудови таблиці біграмів (пар слів). Функція NLTK `bigrams()`, як аргумент бере список слів і повертає список послідовних пар слів.

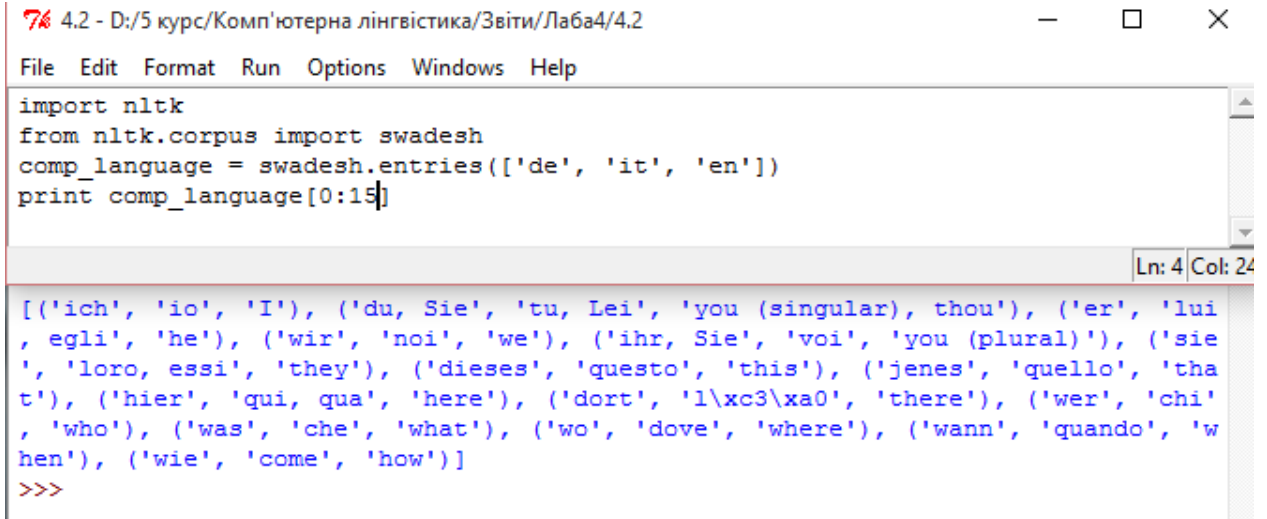
Лексичні ресурси NLTK

Лексичний ресурс або просто словник це набір слів та/або словосполучень, які асоціюються з такою інформацією, як частина мови та опис значення. Лексичні ресурси є вторинними по відношенню до текстів і зазвичай створюються і вдосконалюються з використанням текстів. Наприклад, якщо визначити текст `my_text` тоді `vocab = sorted(set(my_text))` побудує словник тексту `my_text`, `word_freq = FreqDist(my_text)` визначить частоту кожного слова в тексті. `vocab` та `word_freq` – приклад простих лексичних ресурсів. Так само конкорданс дає інформацію про використання слів і ця інформація може бути використана при побудові словників.

ТЕКСТИ ПРОГРАМ НА МОВІ PYTHON

ВАРІАНТ №8

2. Використовуючи компаративний словник знайти близькі слова для німецької, італійської та англійської мов. Чи можуть отримані результати використовуватися для здійснення перекладу?



```
4.2 - D:/5 курс/Комп'ютерна лінгвістика/Звіти/Лаба4/4.2
File Edit Format Run Options Windows Help
import nltk
from nltk.corpus import swadesh
comp_language = swadesh.entries(['de', 'it', 'en'])
print comp_language[0:15]

[('ich', 'io', 'I'), ('du', 'Sie', 'tu', 'Lei', 'you (singular), thou'), ('er', 'lui', 'egli', 'he'), ('wir', 'noi', 'we'), ('ihr', 'Sie', 'voi', 'you (plural)'), ('sie', 'loro', 'essi', 'they'), ('dieses', 'questo', 'this'), ('jenes', 'quello', 'that'), ('hier', 'qui, qua', 'here'), ('dort', 'l\xc3\xa0', 'there'), ('wer', 'chi', 'who'), ('was', 'che', 'what'), ('wo', 'dove', 'where'), ('wann', 'quando', 'when'), ('wie', 'come', 'how')]
>>>
```

Рис. 1. Текст програми №2.

3. Побудувати умовний частотний розподіл для корпусу імен. Знайти які перші літери частіше використовуються в чоловічих та жіночих іменах.

7% 4.3.py - D:\5 курс\Комп'ютерна лінгвістика\Звіти\Лаба4\4.3.py

File Edit Format Run Options Windows Help

```
from nltk.corpus import names
from nltk.probability import ConditionalFreqDist
cfd = ConditionalFreqDist(
    (fileid, name[0])
    for fileid in names.fileids()
    for name in names.words(fileid))
cfd.plot()
```

7% Figure 1

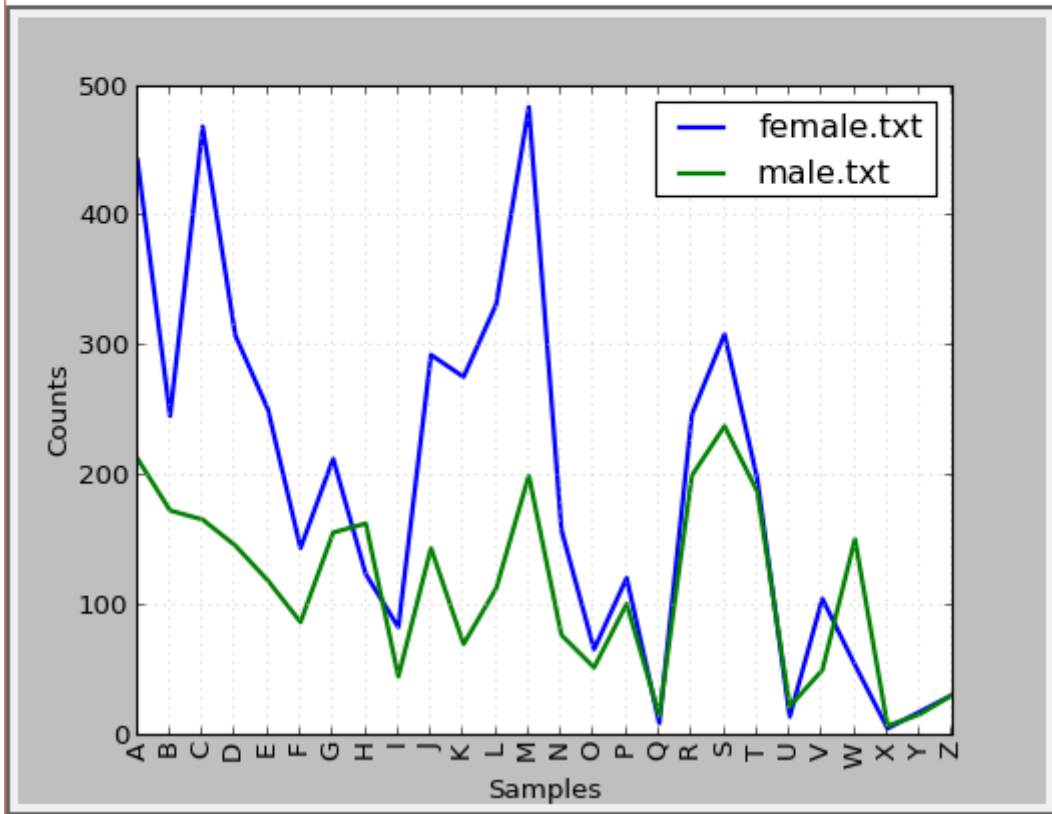


Рис. 2. Текст програми №3.

4. Здійснити аналіз словника вимов. Знайти скільки різних слів він містить. Який відсоток слів з цього словника можуть мати різну вимову?

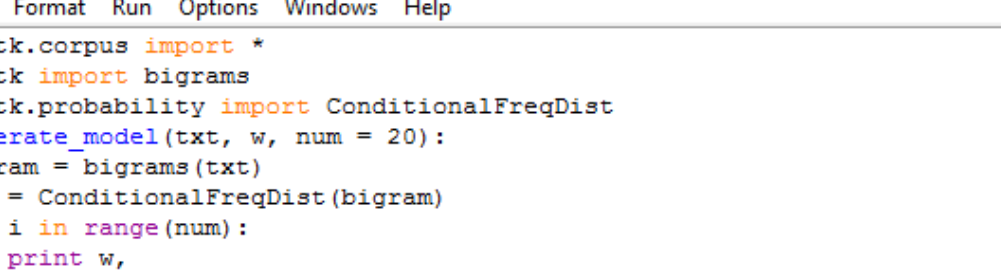
File Edit Format Run Options Windows Help

```
from __future__ import division
from nltk.corpus import cmudict
num = cmudict.entries()
print len(num)
all_words = []
poly = []
for word, pron in num:
    if word not in all_words:
        all_words.append(word)
    else:
        poly.append(word)
print 'Number of words in the dictionary: ', len(all_words)
poly = set(poly)
print 'Words with different pron: ', len(poly)
percent = len(poly)*100/len(all_words)
print percent, '% of words have different pronunciation.'
```

```
>>>
133737
Number of words in the dictionary: 123455
Words with different pron: 9241
7.48531853712 % of words have different pronunciation.
>>>
```

Рис. 3. Текст програми №4.

8. Модифікувати програму генерації випадкового тексту для виконання наступного: тренувати програму на текстах різних жанрів та різних корпусів. Генерацію тексту провести з 5-ма різними початковими словами. Результати проаналізувати та порівняти.

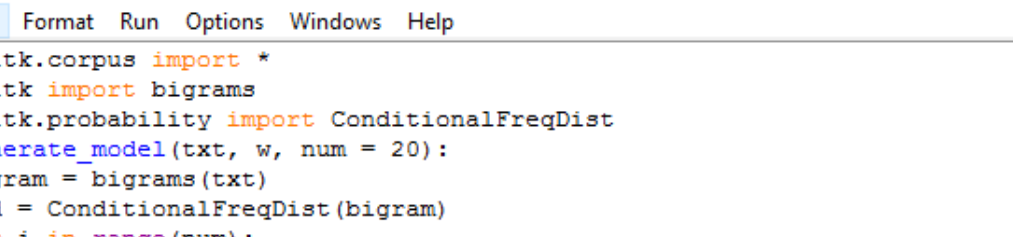


```
4.8.py - D:\5 курс\Комп'ютерна лінгвістика\Звіти\Лаба4\4.8.py
File Edit Format Run Options Windows Help

from nltk.corpus import *
from nltk import bigrams
from nltk.probability import ConditionalFreqDist
def generate_model(txt, w, num = 20):
    bigram = bigrams(txt)
    cfd = ConditionalFreqDist(bigram)
    for i in range(num):
        print w,
        w = cfd[w].max()
text = genesis.words()
word = raw_input('Введіть початкове слово ')
print generate_model(text, word)

Ln: 10 Col: 2
>>>
Введіть початкове слово land
land of the land of the land of the land of the land of the land of
None
>>>
```

Рис. 4а. Текст програми №8, корпус genesis, початкове слово land.

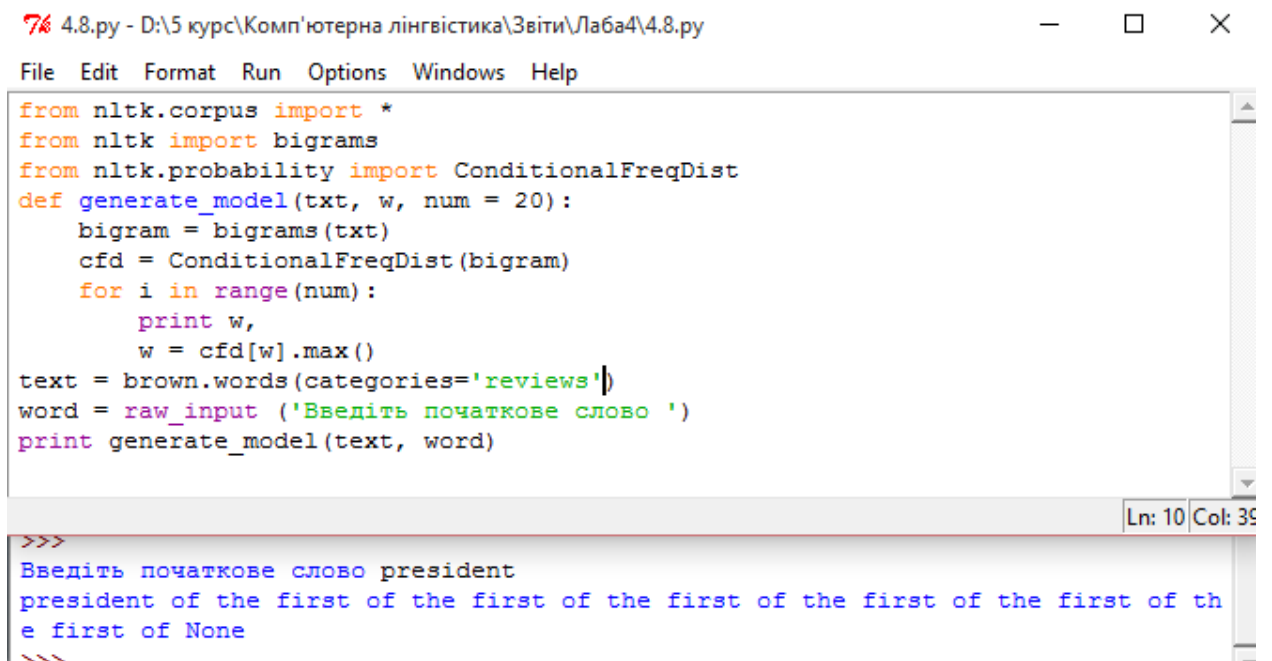


```
7% 4.8.py - D:\5 курс\Комп'ютерна лінгвістика\Звіти\Лаба4\4.8.py
File Edit Format Run Options Windows Help

from nltk.corpus import *
from nltk import bigrams
from nltk.probability import ConditionalFreqDist
def generate_model(txt, w, num = 20):
    bigram = bigrams(txt)
    cfd = ConditionalFreqDist(bigram)
    for i in range(num):
        print w,
        w = cfd[w].max()
text = gutenberg.words()
word = raw_input ('Введіть початкове слово ')
print generate_model(text, word)

>>>
Введіть початкове слово news
news , and the LORD , and the LORD , and the LORD , and the None
>>>
```

Рис. 4б. Текст програми №8, корпус gutenberг, початкове слово news.



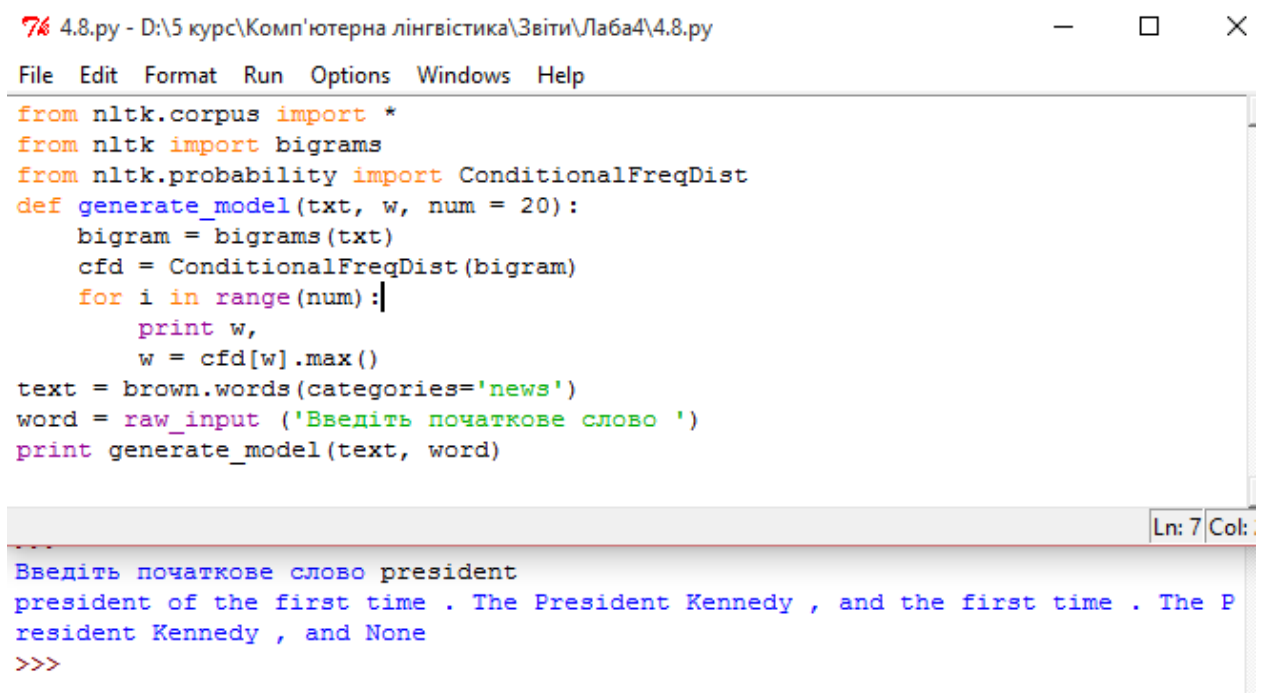
```
4.8.py - D:\5 курс\Комп'ютерна лінгвістика\Звіти\Лаба4\4.8.py
File Edit Format Run Options Windows Help

from nltk.corpus import *
from nltk import bigrams
from nltk.probability import ConditionalFreqDist
def generate_model(txt, w, num = 20):
    bigram = bigrams(txt)
    cfd = ConditionalFreqDist(bigram)
    for i in range(num):
        print w,
        w = cfd[w].max()
text = brown.words(categories='reviews')
word = raw_input ('Введіть початкове слово ')
print generate_model(text, word)

Ln: 10 Col: 39

>>>
Введіть початкове слово president
president of the first of the first of the first of the first of th
e first of None
>>>
```

Рис. 4в. Текст програми №8, корпус brown, жанр reviews, початкове слово president.



```
4.8.py - D:\5 курс\Комп'ютерна лінгвістика\Звіти\Лаба4\4.8.py
File Edit Format Run Options Windows Help

from nltk.corpus import *
from nltk import bigrams
from nltk.probability import ConditionalFreqDist
def generate_model(txt, w, num = 20):
    bigram = bigrams(txt)
    cfd = ConditionalFreqDist(bigram)
    for i in range(num):
        print w,
        w = cfd[w].max()
text = brown.words(categories='news')
word = raw_input ('Введіть початкове слово ')
print generate_model(text, word)

Ln: 7 Col:

Введіть початкове слово president
president of the first time . The President Kennedy , and the first time . The P
resident Kennedy , and None
>>>
```

Рис. 4г. Текст програми №8, корпус brown, жанр news, початкове слово president.


```
7% 4.8.py - D:\5 курс\Комп'ютерна лінгвістика\Звіти\Лаба4\4.8.py
File Edit Format Run Options Windows Help
from nltk.corpus import *
from nltk import bigrams
from nltk.probability import ConditionalFreqDist
def generate_model(txt, w, num = 20):
    bigram = bigrams(txt)
    cfd = ConditionalFreqDist(bigram)
    for i in range(num):
        print w,
        w = cfd[w].max()
text = brown.words(categories='government')
word = raw_input ('Введіть початкове слово ')
print generate_model(text, word)

Ln: 10 Col: 1

>>>
Введіть початкове слово president
president can be made to the United States , and the United States , and the Uni
ted States , and None
...
```

Рис. 4д. Текст програми №8, корпус brown, жанр government, початкове слово president.

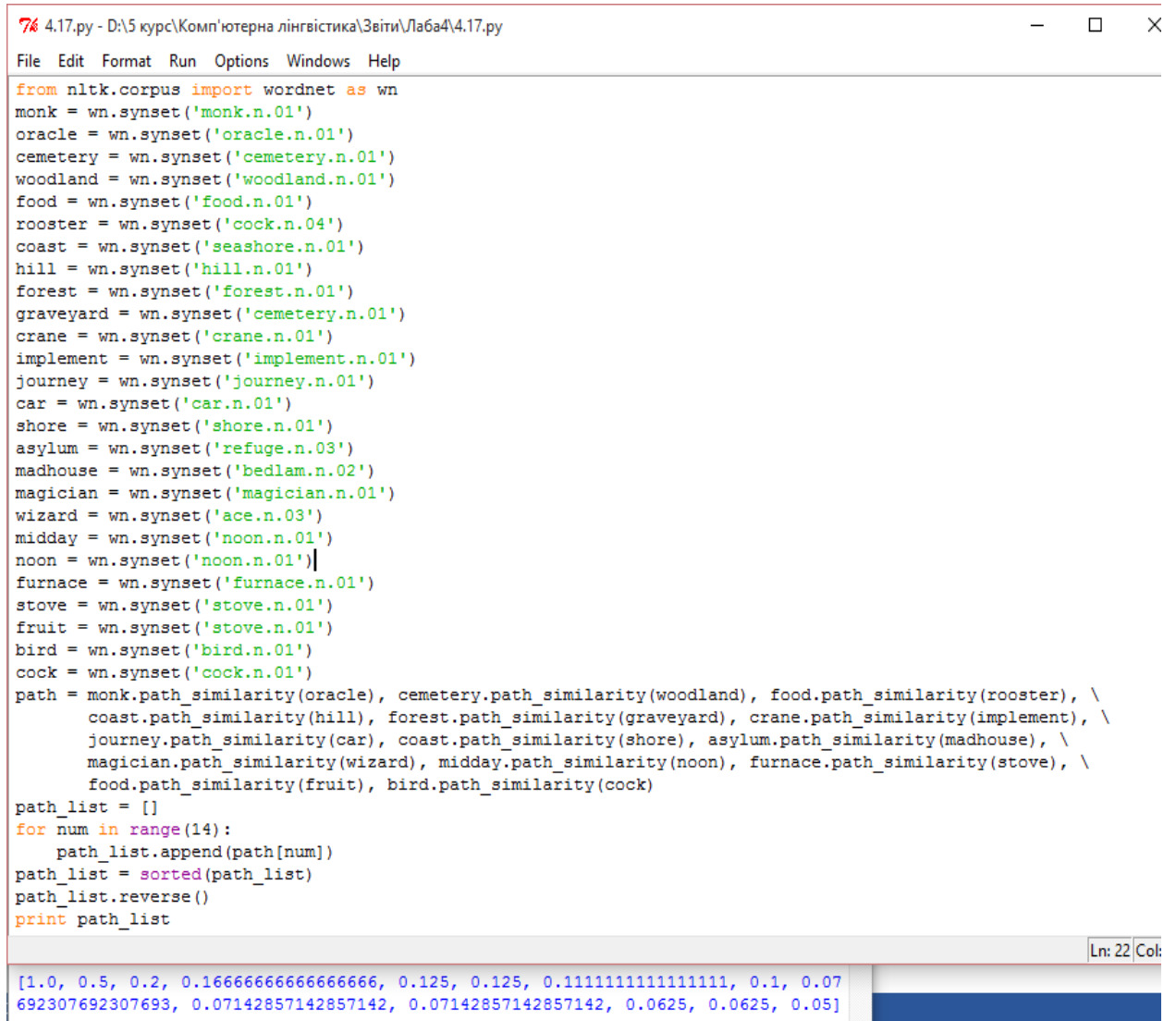
13. Полісемія – це явище коли одне слово має декілька значень (іменник dog має 7 значень, кількість яких визначити можна як `len(wn.synsets('dog', 'n'))`). Знайдіть середнє значення полісемії для прислівників.

```
File Edit Format Run Options Windows Help
from __future__ import division
from nltk.corpus import wordnet as wn
all_adverbs = []
for i in wn.all_synsets('r'):
    for j in i.lemma_names:
        all_adverbs.append(j)
num_adverbs = len(set(all_adverbs))
print 'Number of all the adverbs =', num_adverbs
poly_adverbs = 0
for i in set(all_adverbs):
    poly_adverbs += len(wn.synsets(i, 'r'))
print 'Number of all the meanings =', poly_adverbs
mean_value = poly_adverbs/num_adverbs
print 'Mean value for adverbs equals ', mean_value

>>>
Number of all the adverbs = 4481
Number of all the meanings = 5616
Mean value for adverbs equals  1.25329167597
>>>
```

Рис. 5. Текст програми №13.

17. Використовуючи один з методів визначення подібності слів побудуйте відсортований по спаданню список значень подібності для наступних пар слів: monk-oracle, cemetery-woodland, food-rooster, coast-hill, forest-graveyard, crane-implement, journey-car, coast-shore, asylum-madhouse, magician-wizard, midday-noon, furnace-stove, food-fruit, bird-cock.



```

4.17.py - D:\5 курс\Комп'ютерна лінгвістика\Звіти\Лаба4\4.17.py
File Edit Format Run Options Windows Help

from nltk.corpus import wordnet as wn
monk = wn.synset('monk.n.01')
oracle = wn.synset('oracle.n.01')
cemetery = wn.synset('cemetery.n.01')
woodland = wn.synset('woodland.n.01')
food = wn.synset('food.n.01')
rooster = wn.synset('cock.n.04')
coast = wn.synset('seashore.n.01')
hill = wn.synset('hill.n.01')
forest = wn.synset('forest.n.01')
graveyard = wn.synset('cemetery.n.01')
crane = wn.synset('crane.n.01')
implement = wn.synset('implement.n.01')
journey = wn.synset('journey.n.01')
car = wn.synset('car.n.01')
shore = wn.synset('shore.n.01')
asylum = wn.synset('refuge.n.03')
madhouse = wn.synset('bedlam.n.02')
magician = wn.synset('magician.n.01')
wizard = wn.synset('ace.n.03')
midday = wn.synset('noon.n.01')
noon = wn.synset('noon.n.01')
furnace = wn.synset('furnace.n.01')
stove = wn.synset('stove.n.01')
fruit = wn.synset('stove.n.01')
bird = wn.synset('bird.n.01')
cock = wn.synset('cock.n.01')
path = monk.path_similarity(oracle), cemetery.path_similarity(woodland), food.path_similarity(rooster), \
      coast.path_similarity(hill), forest.path_similarity(graveyard), crane.path_similarity(implement), \
      journey.path_similarity(car), coast.path_similarity(shore), asylum.path_similarity(madhouse), \
      magician.path_similarity(wizard), midday.path_similarity(noon), furnace.path_similarity(stove), \
      food.path_similarity(fruit), bird.path_similarity(cock)
path_list = []
for num in range(14):
    path_list.append(path[num])
path_list = sorted(path_list)
path_list.reverse()
print path_list

Ln: 22 Col:
[1.0, 0.5, 0.2, 0.16666666666666666, 0.125, 0.125, 0.11111111111111111, 0.1, 0.07
692307692307693, 0.07142857142857142, 0.07142857142857142, 0.0625, 0.0625, 0.05]

```

Рис. 6. Текст програми №17.

ВИСНОВОК

У цій лабораторній роботі я вивчила основи програмування на Python. Ознайомилася з прикладами корпусів текстів та методами доступу до них, навчилася будувати умовний частотний розподіл. Дізналася про поняття функції та модуля.