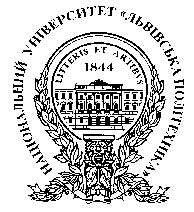
МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ

Національний Університет “Львівська політехніка”



Лабораторна робота № 6

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

***ВИКОРИСТАННЯ РЕГУЛЯРНИХ ВИРАЗІВ ДЛЯ ОБРОБКИ ТЕКСТУ.***

Виконав:

студент групи ПРЛс-11

Форманюк А.А.

Прийняв:

асистент

Дупак Б.П.

Львів 2015

**КОРОТКІ ТЕОРЕТИЧНІ ВІДОМОСТІ**

**Виконанні цієї лабораторної роботи необхідно розпочати з:**

|  |  |
| --- | --- |
|  | **>>> from \_\_future\_\_ import division**  **>>> import nltk, re, pprint** |

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

Багато задач лінгвістичних досліджень передбачають встановлення відповідності заданому шаблону. Наприклад, можна знайти слова, які закінчуються на “ed” використовуючи метод endswith('ed'). Подібні методи перевірки слів перелічені в Таблиці Методичних вказівок до лабораторної роботи №2. Регулярні вирази є більш потужним і гнучким методом опису шаблонів символів, які необхідно виявити у послідовностях символів. **Регулярний вираз (вислів)** (в [програмуванні](http://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)) — це [рядок](http://uk.wikipedia.org/wiki/%D0%A0%D1%8F%D0%B4%D0%BE%D0%BA_%28%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F%29) що описує або збігається з [множиною](http://uk.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B6%D0%B8%D0%BD%D0%B0) рядків, відповідно до набору спеціальних [синтаксичних](http://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%82%D0%B0%D0%BA%D1%81%D0%B8%D1%81) правил. Регулярні вислови використовуються в багатьох [текстових редакторах](http://uk.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D0%B8%D0%B9_%D1%80%D0%B5%D0%B4%D0%B0%D0%BA%D1%82%D0%BE%D1%80) та допоміжних інструментах для пошуку та зміни тексту на основі заданих шаблонів.

Для роботи з регулярними виразами у Python потрібно імпортувати бібліотеку re скориставшись: import re. Корпус слів англійської мови Words Corpus буде використовуватися в якості лінгвістичних даних серед яких буде проводитися пошук. Попередня підготовка списку слів передбачає видалення власних імен.

|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | |  | **>>> import re**  **>>> wordlist = [w for w in nltk.corpus.words.words('en') if w.islower()]** | |

### Використання основних метасимволів (операторів повтору).

Синтаксис регулярних висловів залежить від інтерпретатора, що використовується для їх обробки. Однак, із незначними відхиленнями, майже всі поширені механізми інтерпретатори регулярних висловів мають спільні правила.

Найпростіший регулярний вислів, з якого формуються складні, є звичайний символ. Більшість символів, включаючи усі літери та цифри, є регулярними висловами, що співпадають із відповідними символами в рядках.

Пошук слів із закінченням ed можна здійснити використовуючи регулярний вираз «ed$». Потрібно використати функцію re.search(p, s), яка перевіряє чи може зразок p бути знайдений у будь-якому місці стрічки s. Потрібно визначити символи, які шукаємо та використати символ долара , який в регулярних виразах позначає кінець слова:

|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | |  | **>>> [w for w in wordlist if re.search('ed$', w)]**  **['abaissed', 'abandoned', 'abased', 'abashed', 'abatised', 'abed', 'aborted', ...]** | |

Символ ”.” універсальний символ , якому відповідає будь-який один символ. Нехай потрібно знайти слова з восьми літер, де j – третя літера та t – шоста літера. При створенні регулярного виразу у місцях де може бути будь-який символ вказується крапка. Символ ”^” вказує на початок стрічки:

|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | |  | **>>> [w for w in wordlist if re.search('^..j..t..$', w)]**  **['abjectly', 'adjuster', 'dejected', 'dejectly', 'injector', 'majestic', ...]** | |

***Виконати самостійно***. Повторіть попередній приклад використовуючи регулярний вираз «..j..t..». Результати порівняйте.

Символ “ ? “ вказує на те що попередній символ не є обовязковим. Вираз «^e-?mail$» відповідає двом стрічкам email та e-mail. Можна знайти загальну кількість таких стрічок (врахувавши різні способи їх запису) у будь-якому тексті скориставшись sum(1 for w in text if re.search('^e-?mail$', w)).

Рис 1. Система T9 для воду тексту за допомогою 9-ти клавіш.

В мобільних телефонах для вводу тексту використовується система T9 (Рис.1.). Два або більше слів, які можуть бути введені тією самою послідовністю клавіш називають textonyms. Наприклад два слова hole та golf вводяться натисненням послідовності 4653. Які інші слова можуть бути створені натисненням тієї самої послідовності клавіш? Використовуючи регулярний вираз «^[ghi][mno][jlk][def]$» це не складно зробити:

|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | |  | **>>> [w for w in wordlist if re.search('^[ghi][mno][jlk][def]$', w)]**  **['gold', 'golf', 'hold', 'hole']** | |

1. Описати, які класи стрічок відповідають наступному регулярному виразу. [a-zA-Z]+. Результати перевірити використовуючи nltk.re\_show( ).

from \_\_future\_\_ import division

import nltk, re, pprint

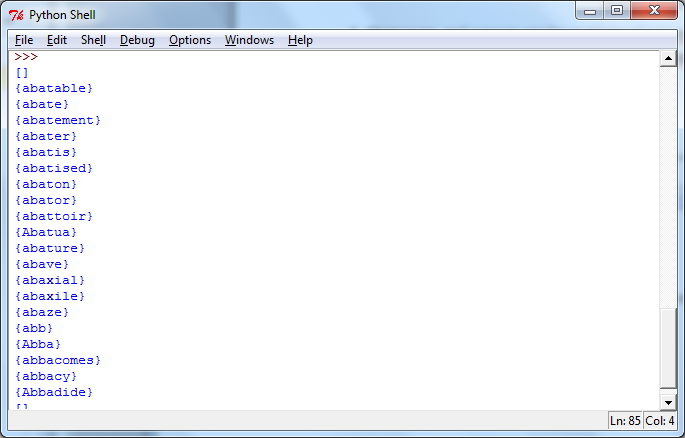
wordlist = [w for w in nltk.corpus.words.words('en')]

str =[w for w in wordlist[80:100] if re.search('[a-zA-Z]+', w)]

print str[80:100]

show =[w for w in wordlist[80:100] if nltk.re\_show('[a-zA-Z]+', w)]

print show[80:100]



Програма виводить всі слова де є великі і малі букви.

2. . Описати, які класи стрічок відповідають наступному регулярному виразу. [A-Z][a-z]\*. Результати перевірити використовуючи nltk.re\_show()

from \_\_future\_\_ import division

import nltk, re, pprint

wordlist = [w for w in nltk.corpus.words.words('en')]

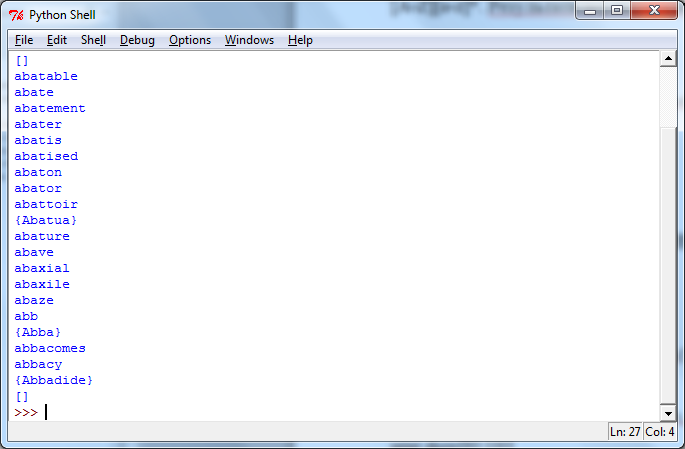
str =[w for w in wordlist[80:100] if re.search('[A-Z][a-z]\*', w)]

print str[80:100]

show =[w for w in wordlist[80:100] if nltk.re\_show('[A-Z][a-z]\*', w)]

print show[80:100]

Програма знаходить слова, що починаються з великої букви.



3. Описати, які класи стрічок відповідають наступному регулярному виразу. \d+(\.\d+)?. Результати перевірити використовуючи nltk.re\_show()

from \_\_future\_\_ import division

import nltk, re, pprint

list = ['68', 'r2r', '96 cars', '18 years', 'ree4', 'u2', '2b']

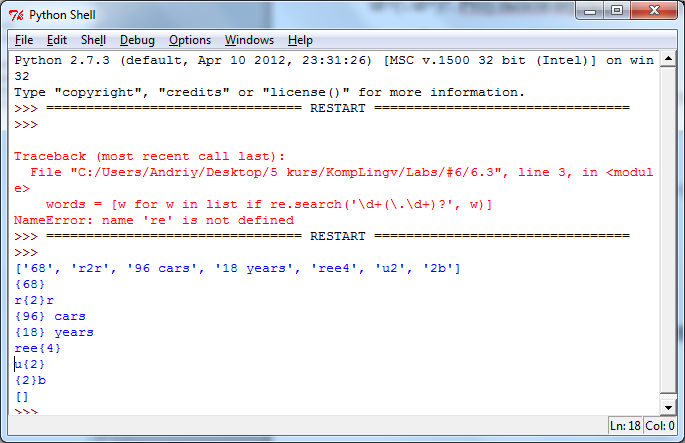
words = [w for w in list if re.search('\d+(\.\d+)?', w)]

print words

show = [w for w in list if nltk.re\_show('\d+(\.\d+)?', w)]

print show

Програма шукає слова, що містять числа.



4. Описати, які класи стрічок відповідають наступному регулярному виразу. ([^aeiou][aeiou][^aeiou])\*. Результати перевірити використовуючи nltk.re\_show()

from \_\_future\_\_ import division

import nltk, re, pprint

wordlist = [w for w in nltk.corpus.words.words('en') if w.islower()]

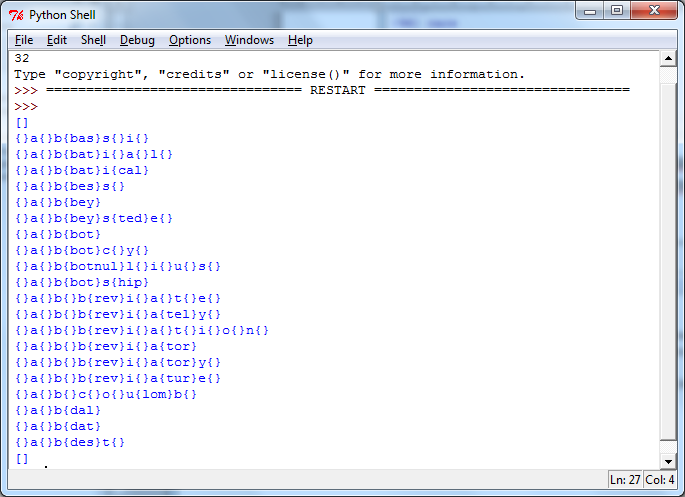
str =[w for w in wordlist[80:100] if re.search('([^aeiou][aeiou][^aeiou])\*', w)]

print str[80:100]

show =[w for w in wordlist[80:100] if nltk.re\_show('([^aeiou][aeiou][^aeiou])\*', w)]

print show[80:100]

Програма шукає слова за шаблоном «голосна-голосна-неголосна.



5. Описати, які класи стрічок відповідають наступному регулярному виразу. \w+|[^\w\s]+.. Результати перевірити використовуючи nltk.re\_show()

from \_\_future\_\_ import division

import nltk, re, pprint

wordlist = [w for w in nltk.corpus.words.words('en') if w.islower()]

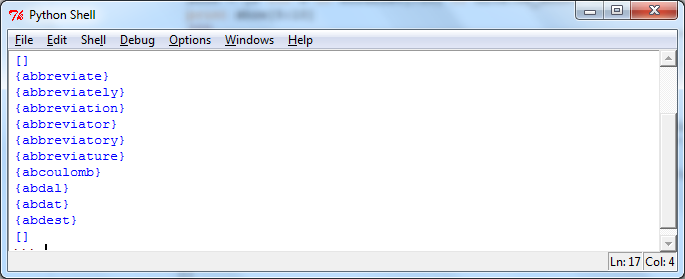
str =[w for w in wordlist[90:100] if re.search('\w+|[^\w\s]+', w)]

print str[90:100]

show =[w for w in wordlist[90:100] if nltk.re\_show('\w+|[^\w\s]+', w)]

print show[90:100]

Програма показує всі символи, цифри і слова.



6. Описати, які класи стрічок відповідають наступному регулярному виразу. p[aeiou]{,2}t Результати перевірити використовуючи nltk.re\_show()

from \_\_future\_\_ import division

import nltk, re, pprint

wordlist = [w for w in nltk.corpus.words.words('en') if w.islower()]

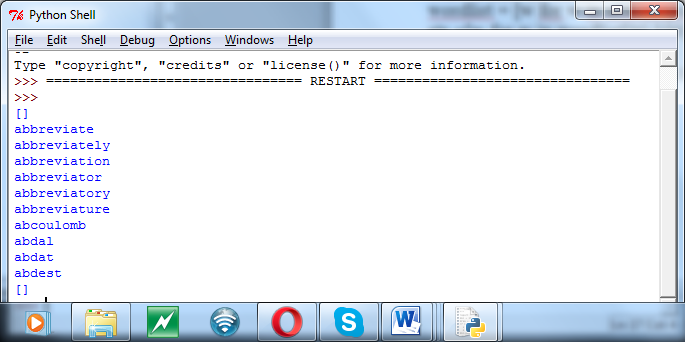
str =[w for w in wordlist[90:100] if re.search('p[aeiou]{,2}t', w)]

print str[90:100]

show =[w for w in wordlist[90:100] if nltk.re\_show('p[aeiou]{,2}t', w)]

print show[90:100]

Прогпама показує слова де між буквами «p» та «t» мужуть бути до 2х голосних.



7. Написати регулярний вираз, який встановлює відповідність наступному класу стрічок: всі артиклі *(a, an, the*).

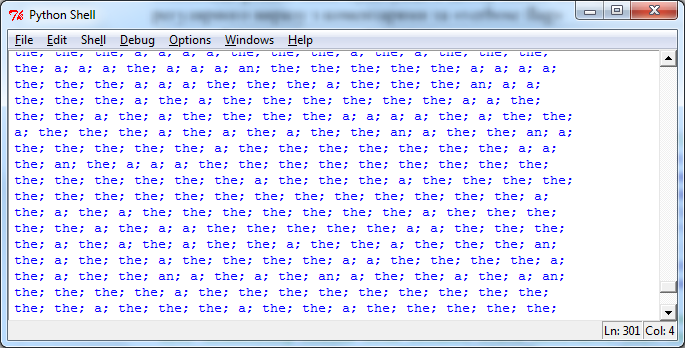
import nltk, re, pprint

from nltk.corpus import brown

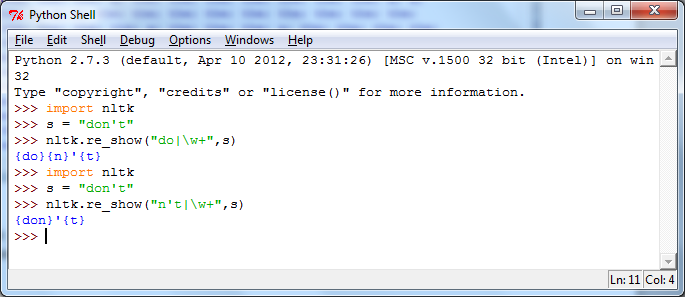
wordlist = nltk.Text(brown.words(categories='mystery'))

article = wordlist.findall(r'<a>|<an>|<the>')

print article



8. Написати регулярний вираз для токенізації такого тексту, як don't до do та n't? Пояснити чому цей регулярний вираз не працює: «n't|\w+».



9. Прочитати файл допомоги про функцію re.sub() використовуючи help(re.sub) . Використовуючи re.sub напишіть програму видалення HTML розмітки замінивши її на пробіли.

import urllib, nltk, re

from urllib import urlopen

print help(re.sub)

url = "http://www.bbc.com/news/world-europe-34840943"

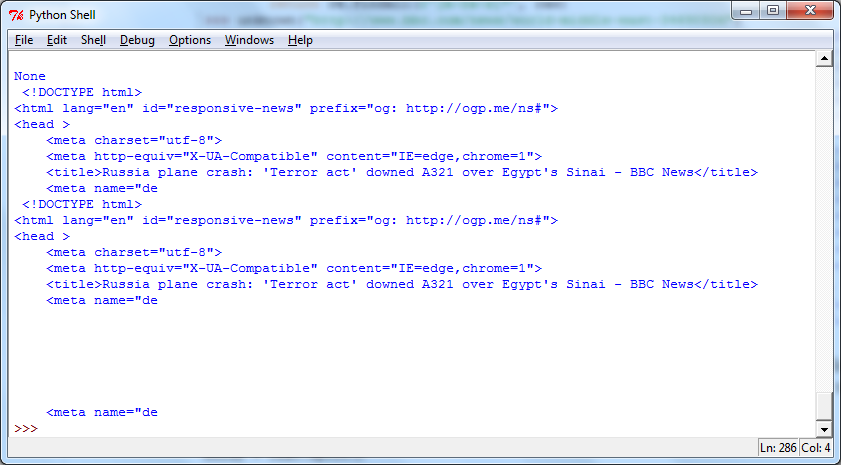
html = urlopen(url).read()

print html[:500]

q = html[:500]

print q

print re.sub(r"<.\*>",' ',q)



10. Прочитати Додаток А. Дослідити явища описані у Додатку А використовуючи корпуси текстів та метод findall()для пошуку в токенізованому тексті.

import nltk, re

from nltk.corpus import brown

text = brown.raw()

words = text.split()

i = 0

q = []

for word in words:

i = word.find('/')

word = word[0:i]

q.append(word)

text = nltk.Text(q)

print 'as best \_ can: ',

text.findall (r'<as best>(<.\*>)<can>')

print 'as best as \_ can: ',

text.findall (r'<as best as>(<.\*>)<can>')

print 'the best \_ can: ',

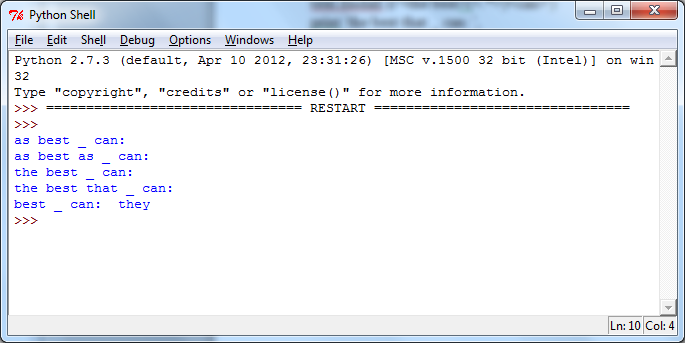
text.findall (r'<the best>(<.\*>)<can>')

print 'the best that \_ can: ',

text.findall (r'<the best that>(<.\*>)<can>')

print 'best \_ can: ',

text.findall (r'<best>(<.\*>)<can>')



**Висновок:** на лабораторній роботі я освоїв використання регулярних виразів для обробки текстів.