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

Кафедра “Системи автоматизованого проектування”

Звіт

до лабораторної роботи №3

на тему: “ ВИВЧЕННЯ БІБЛІОТЕКИ ПРИКЛАДНИХ ПРОГРАМ NLTK ДЛЯ ОПРАЦЮВАННЯ ТЕКСТІВ ПРИРОДНОЮ МОВОЮ.
ДОСТУП ТА РОБОТА З КОРПУСАМИ ТЕКСТІВ.”

з дисципліни “Комп’ютерна лінгвістика”

Виконала:
студентка групи ПРЛм-11
Зварич О.І.
Прийняв:
викладач
Дупак Б.П.

Львів-2015

1. Мета.

- Вивчення основ програмування на мові Python.
- Вивчення методів доступу до корпусів текстів.
- Вивчення класу ConditionalFreqDist.

2. Теоретичні відомості.

При роботі з корпусами важливо мати засоби доступу як до окремих текстів так і до окремих частин цих тексів, а також і до окремих слів.

В NLTK входить невелика частина текстів з електронного архіву текстів Project Gutenberg, який містить 25000 безкоштовних електронних книжок різних авторів (http://www.gutenberg.org/). Тексти творів в окремих файлах. Для одержання назв файлів (ідентифікаторів файлів) в яких зберігаються текстів потрібно використати наступну функцію:

```
>>> import nltk

>>> nltk.corpus.gutenberg.fileids()
```

Функція .raw()дозволяє доступитися до вмісту файла без будь-якої його попередньої лінгвістичної обробки. Тому використання len(gutenberg.raw('blake-poems.txt') дозволяє встановити скільки символів (разом з пробілами) є в тексті. Функція sents() ділить текст на оремі речення і кожне речення представляється, як список стрічок, де стічки – окремі слова.

Для роботи з менш формальною мовою, NLTK містить набір текстів з Інтернету: тексти з форуму, тексти з фільму Пірати карибського моря, тексти особистих оголошень,телефонні розмови, огляд новин:

```
>>> from nltk.corpus import webtext

>>> for fileid in webtext.fileids():
```

Також в NLTK входить корпус повідомлень з чатів, створений в Naval Postgraduate School для досліджень з метою автоматичного виявлення Інтернет злочинців. Цей корпус містить 10000 анонімних повідомлень в яких імена користувачів замінені за шаблоном "UserNNN" а також видалена інша персональна інформація. Корпус організований ,як 15 окремих файлів, кожен з яких містить декілька сотень повідомлень з певною датою створення та вікових даних авторів (підлітки, 20ти, 30ти та 40ка річні, дорослі). Назва файла містить інформацію про дату, вікову групу та кількість повідомлень, наприклад файл 10-19-20s_706posts.xml містить 706 повідомлень двадцятирічних дописувачів від 19 жовтня 2006 року.

Використовуючи засоби NLTK можна отримати доступ до корпусу Браун, як до списку слів або списку речень (кожне речення – список слів). Також доступна можливість вибору текстів окремої категорії або з окремого файлу.

```
>>> from nltk.corpus import brown

>>> brown.categories()

['adventure', 'belles_lettres', 'editorial', 'fiction', 'government', 'hobbies',

'humor', 'learned', 'lore', 'mystery', 'news', 'religion', 'reviews', 'romance',
```

'science_fiction']

Корпус Reuters. Засобами NLTK можна звернутися до тем, яких торкаються в одному або декількох текстах або навпаки дізнатися весь перелік текстів, які належать до певної категорії.

Найпростіший корпус текстів не має структури, це набір текстів. Інші корпуси це набори текстів поділених за категоріями мови, жанру, автора. В багатьох випадках категорії текстів можуть перетинатися між собою, оскільки тексти можуть належати різним категоріям. Окремий випадок це коли набори текстів розподілені за часовими параметрами.

Якщо тексти в корпусі поділені на різні категорії, (за жанром, тематикою, авторами) то можна побудувати частотні розподіли для кожної з категорій. Такі дані дозволяють досліджувати відмінності між жанрами. Умовний частотний розподіл це набір частотних розподілів, кожен з яких відповідає певній «умові». Такою умовою може бути категорія тексту.

Варіант 4:

4. Використовуючи конкорданси поясніть відмінності у вживанні слова however на початку речення ("in whatever way", "to whatever extent", або "nevertheless").

```
>>> import nltk
>>> from nltk.corpus import gutenberg
>>> data = nltk.Text(nltk.corpus.gutenberg.words())
>>> data.concordance("however")
Building index...
Displaying 25 of 673 matches:
 her many enjoyments . The danger , however , was at present so unperceived , t
ion would offend . Miss Churchill , however , being of age , and with the full
n . From the expense of the child , however , he was soon relieved . The boy ha
 -- and been very well brought up . However , I do not mean to set up my opinio
f and predict . It was not likely , however , that any body should have equalle
to be borne . We will not despair , however . Weston may grow cross from the wa
is so very handsome and agreeable . However , I do really think Mr . Martin a v
 accepted after all . This letter , however , was written , and sealed , and se
e him ." " And if I did , ( which , however , I am far from allowing ) I should
 slightingly . Waiving that point , however , and supposing her to be , as you
e was not so materially cast down , however , but that a little time and the re
ld inspire him ." The very next day however produced some proof of inspiration
and staid to look through herself ; however , she called me back presently , an
t turn up . His ostensible reason , however , was to ask whether Mr . Woodhouse
l and cross . This does not apply , however , to Miss Bates ; she is only too g
and sufferings of the poor family , however , were the first subject on meeting
ting for her . She gained on them , however , involuntarily : the child ' s pac
ould close it . It was not closed , however , it still remained ajar ; but by e
 believes himself secure ." Still , however , though every thing had not been a
ght advance rapidly if they would , however ; they must advance somehow or othe
 offence came not . The beginning , however , of every visit displayed none but
eed !-- and my memory is very bad . However , it was an exceeding good , pretty
first day . Emma ' s sense of right however had decided it ; and besides the co
le fatigued . I could have wished , however , as you know , that you had seen M
" Our little friend Harriet Smith , however , is just such another pretty kind
>>>
```

6. Проаналізуйте таблицю частот модальних дієслів для різних жанрів. Спробуйте її пояснити. Знайдіть інші класи слів вживання яких також відрізняються в різних жанрах.

```
import nltk
from nltk.corpus import brown
cdf = nltk.ConditionalFreqDist(
          (genre, word)
          for genre in brown.categories()
          for word in brown.words(categories=genre))
genres = ['news', 'religion', 'hobbies', 'science_fiction', 'romance', 'humor']
modals = ['can', 'could', 'may', 'might', 'must', 'will']
cdf.tabulate(conditions=genres, samples=modals)
```

```
>>>
                  can could  may might must will
            news   93    86   66    38   50  389
        religion   82    59   78    12   54   71
         hobbies  268    58  131    22   83  264
science_fiction   16    49    4    12    8   16
         romance   74   193   11    51   45   43
           humor   16    30    8     8    9   13
```

Проаналізувавши таблицю частот модальних дієслів для різних жанрів, можна зробити висновок, що: у жанрі news слово will зустрічається найчастіше. Can – найчастотніше слово у жанрах religion та hobbies. У жанрах science_fiction, romance, humour – could.

Вживання питальних прийменників у різних жанрах.

```
>>> import nltk
>>> from nltk.corpus import brown
>>> cdf=nltk.ConditionalFreqDist(
        (genre, word)
        for genre in brown.categories()
        for word in brown.words(categories=genre))
genres=['news', 'religion', 'hobbies', 'science_fiction', 'romance', 'humor']
>>> pronouns=['what', 'when', 'where', 'who', 'why']
>>> cdf.tabulate(conditions=genres, samples=pronouns)
                what when where  who  why
           news   76  128    58  268    9
       religion   64   53    20  100   14
        hobbies   78  119    72  103   10
science_fiction   27   21    10   13    4
        romance  121  126    54   89   34
          humor   36   52    15   48    9
>>>
```

7. Напишіть програму для знаходження всіх слів в корпусі Brown, які зустрічаються не менш ніж три рази.

```
>>> import nltk
>>> from nltk.corpus import brown
>>> words = brown.words()
>>> fdlist=nltk.FreqDist(words)
>>> list=[w for w in set(words) if fdlist[w]>=3]
>>> print (list[1:50])
['woods', 'hanging', 'over/under', 'originality', 'opener', 'Western', 'Elec', 'co
-operation', 'pigment', 'appropriation', 'bringing', 'wooded', 'wooden', 'ultra-vi
olet', 'non-violent', "Hammarskjold's", 'inevitably', 'feasibility', '275', 'Honor
able', 'scraped', 'snuggled', 'errors', 'Initially', 'self-reliant', 'cooking', 'H
amilton', 'designing', 'College', 'shocks', 'crouch', 'Foundation', 'china', 'affi
liated', 'confronts', 'kids', 'climbed', 'controversy', 'natures', 'Isles', 'rebel
', 'golden', 'Dexter', 'topography', 'projection', 'Harvey', 'Matilda', 'lower-mid
dle', 'stern']
>>>
```

8. Напишіть програму генерації таблиці відношень кількість слів/кількість оригінальних слів для всіх жанрів корпуса Brown.

```
>>> import nltk
>>> from nltk.corpus import brown
>>> for category in brown.categories():
        num_words = len(brown.words(categories=category))
        num_vocab = len(set([w.lower() for w in brown.words(categories=category)]))
        print num_words, num_vocab, (num_words/num_vocab), category


69342 8289 8 adventure
173096 17058 10 belles_lettres
61604 9109 6 editorial
68488 8680 7 fiction
70117 7361 9 government
82345 10824 7 hobbies
21695 4755 4 humor
181888 15476 11 learned
110299 13403 8 lore
57169 6463 8 mystery
100554 13112 7 news
39399 5931 6 religion
40704 8069 5 reviews
70022 7883 8 romance
14470 3032 4 science_fiction
>>>
```

9. Напишіть програму для знаходження 50 найчастотніших слів в тексті, за виключенням незначущих слів.

```
import nltk
from nltk.corpus import gutenberg

persuasion=gutenberg.words('austen-persuasion.txt')

fdist=nltk.FreqDist([w.lower() for w in persuasion])
fdist=nltk.FreqDist(persuasion)

vocabulary = fdist.keys()
vocabulary[:50]
print vocabulary[:50]
```

```
[',', 'the', 'to', '.', 'and', 'of', 'a', 'in', 'was', ';', 'had', 'her', 'I', '
be', 'not', 'it', 'that', 'she', '"', 'as', 'he', 'for', 'with', 'his', 'have',
'but', 'you', "'", 'at', 'all', 'Anne', 'been', 's', 'him', 'could', 'were', 've
ry', 'which', 'by', 'is', 'on', '."', 'would', 'so', 'She', 'they', 'no', '-', '
Captain', 'Mrs']
```

13. Визначити функцію hedge(text), яка обробляє текст і створює нову версію цього тексту додаючи слово ‘like’ перед кожним третім словом.

```
textt='The truce, which the OSCE is monitoring, was agreed in the Belarussian ca
def hedge(text):
    a=[]
    text = text.split()
    for i in text:
        if text.index(i) in range(2, len(text), 2):
            a.append('like ' + i)
        else:
            a.append(i)
        new_text=''
        for word in a:
            new_text+=word + ' '
    print new_text
hedge(textt)
```

```
The truce, like which the like OSCE is like monitoring, was like agreed
```

Висновок: я ознайомилась з основами програмування на мові Python.Освоїла методів доступу до корпусів текстів. Та опанувала клас ConditionalFreqDist.