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

Кафедра “Системи автоматизованого проектування”

Звіт

до лабораторної роботи №6

на тему: «ВИВЧЕННЯ БІБЛІОТЕКИ ПРИКЛАДНИХ ПРОГРАМ NLTK, ДЛЯ ОПРАЦЮВАННЯ ТЕКСТІВ ПРИРОДНОЮ МОВОЮ. ВИКОРИСТАННЯ РЕГУЛЯРНИХ ВИРАЗІВ ДЛЯ ОБРОБКИ ТЕКСТУ»

з дисципліни “Комп’ютерна лінгвістика”

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

Львів-2015

МЕТА РОБОТА

- Вивчення основ програмування на мові *Python*.
- Використання регулярних виразів для обробки текстів.

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

У цій програмі ми шукаємо всі слова, які складаються з малих та великих літер англійського алфавіту. Слова, в яких є повторення будь-якої літери один або більше разів.

```
from __future__ import division
import nltk
import re
text = 'Three Israelis have been killed and more than 20 injured in shooting and
wordlist=nltk.word_tokenize(text)
print wordlist
print [w for w in wordlist if nltk.re_show('[a-zA-Z]+', w)]
```

```
['Three', 'Israelis', 'have', 'been', 'killed', 'and', 'more', 'than', '20', 'i
njured', 'in', 'shooting', 'and', 'stabbing', 'attacks', 'in', 'Jerusalem', 'an
d', 'central', 'Israel', ',', 'Israeli', 'police', 'say', '.']
{Three}
{Israelis}
{have}
{been}
{killed}
{and}
{more}
{than}
20
{injured}
{in}
{shooting}
{and}
{stabbing}
{attacks}
{in}
{Jerusalem}
{and}
{central}
{Israel}
,
{Israeli}
{police}
{say}
.
[]
```

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

У цій програмі ми шукаємо послідовності символів, що починаються з великої літери англійського алфавіту

```
from __future__ import division
import nltk
import re
wordlist = 'Three Israelis have been killed and more than 20 injured in shooting
print wordlist
print nltk.re_show('[A-Z][a-z]*', wordlist)
```

```
Three Israelis have been killed and more than 20 injured in shooting and stabbi
ng attacks in Jerusalem and central Israel, Israeli police say.
{Three} {Israelis} have been killed and more than 20 injured in shooting and st
abbing attacks in {Jerusalem} and central {Israel}, {Israeli} police say.
None
```

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

У цій програмі ми шукаємо послідовності символів, що складаються з цифр, які повторюються з 1 і більше разів; (\.\d+)? – послідовність цифр після крапки не є обов'язковою

```
from __future__ import division
import nltk
import re
wordlist = 'Three Israelis have been killed and more than 20 injured in shooting
print wordlist
print nltk.re_show('\d+(\.\d+)?', wordlist)
```

```
Three Israelis have been killed and more than 20 injured in shooting and stabbi
ng attacks in Jerusalem and central Israel, Israeli police say.
Three Israelis have been killed and more than {20} injured in shooting and stab
bing attacks in Jerusalem and central Israel, Israeli police say.
None
```

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

У цій програмі ми шукаємо послідовності символів, що складаються з трьох символів, перший і третій з яких не є голосною, а другий – будь-яка голосна з набору [aeiou] і зустрічаються 0 і більше разів. Якщо послідовність не знайдено – виводиться {}

```
from __future__ import division
import nltk
import re
wordlist = 'Three Israelis have been killed and more than 20 injured in shooting
print wordlist
print nltk.re_show('([^aeiou][aeiou][^aeiou])*' , wordlist)
```

```
Three Israelis have been killed and more than 20 injured in shooting and stabbin
g attacks in Jerusalem and central Israel, Israeli police say.
{}T{}h{}r{}e{}e{} {}I{}s{}r{}a{}e{lis} {hav}e{} {}b{}e{}e{}n{} {killed an}d{} {m
or}e{} {}t{han} {}2{}0{ injur}e{}d{ in} {}s{}h{}o{}o{tin}g{ an}d{} {}s{tabbin}g{
 attac}k{}s{ in} {Jer}u{sal}e{}m{ an}d{} {cen}t{ral} {}I{}s{}r{}a{}e{}l{},{} {}I
{}s{}r{}a{}e{li pol}i{ce say}.{}
None
```

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

У цій програмі ми шукаємо послідовності, що складаються з літер чи цифр, які повторюються 1 і більше разів

```
from __future__ import division
import nltk
import re
wordlist = 'Three Israelis have been killed and more than 20 injured in shooting
print wordlist
print nltk.re_show('\w+|[^\w\s]+' , wordlist)
```

```
Three Israelis have been killed and more than 20 injured in shooting and stabbin
g attacks in Jerusalem and central Israel, Israeli police say.
{Three} {Israelis} {have} {been} {killed} {and} {more} {than} {20} {injured} {in
} {shooting} {and} {stabbing} {attacks} {in} {Jerusalem} {and} {central} {Israel
}{,} {Israeli} {police} {say}{.}
None
```

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

У цій програмі ми шукаємо стрічки, які складаються з букви “p”, жодної, одної або двох голосних з набору і літери “t”.

```
from __future__ import division
import nltk
import re
wordlist = 'Three Israelis have been killed and more than 20 injured in shooting
print wordlist
print nltk.re_show('p[aeiou]{,2}t' , wordlist)
```

```
Three Israelis have been killed and more than 20 injured in shooting and stabbin
g attacks in Jerusalem and central Israel, Israeli police say.
Three Israelis have been killed and more than 20 injured in shooting and stabbin
g attacks in Jerusalem and central Israel, Israeli police say.
None
```

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

```
from __future__ import division
import nltk
import re
chat_words = sorted(set(w for w in nltk.corpus.nps_chat.words()))
article = [w for w in chat_words if re.search('^(an?|the)$', w)]
print article
```

```
['a', 'an', 'the']
>>>
```

9. Зберегти довільний текст у файлі corpus.txt. Визначити функцію для читання з цього файлу (назва файлу аргумент функції) і повертає стрічку, яка містить текст з файлу. Використовуючи nltk.regexp_tokenize() розробити токенізатор для токенізації різних типів пунктуації в цьому тексті. Використовувати багаторядковий запис регулярного виразу з коментарями та «verbose flag»

```
>>> import nltk
>>> import re
>>> def Mytext(tx):
        s=''
        for line in tx:
                p= line.strip()
                sl=p
        print sl


>>> f=open('E:\Text\corpus.txt')
>>> Mytext(f)
Israeli Prime Minister Benjamin Netanyahu convened an emergency session of the secu
rity cabinet to discuss how to prevent further attacks.
>>> f=open('E:\Text\corpus.txt')
>>> raw = f.read()
>>> print raw
Three Israelis have been killed and more than 20 injured in shooting and stabbing a
ttacks in Jerusalem and central Israel, Israeli police say.
Two were killed when two assailants, who were identified as Arabs, shot and stabbed
passengers on a bus in Jerusalem before being shot by police.
Another Israeli died after being run down and stabbed elsewhere in the city.
Near-daily stabbings by Palestinians have left dozens of Israelis dead and wounded
over the past fortnight.
Several attackers and at least 17 other Palestinians have been killed in the upsurg
e of violence.
Israeli Prime Minister Benjamin Netanyahu convened an emergency session of the secu
rity cabinet to discuss how to prevent further attacks.
>>> pattern = r'''(?x)    # set flag to allow verbose regexps
...     ([A-Z]\.)+        # abbreviations, e.g. U.S.A.
...   | \w+(-\w+)*        # words with optional internal hyphens
...   | \$?\d+(\.\d+)?%?  # currency and percentages, e.g. $12.40, 82%
...   | \.\.\.            # ellipsis
...   | [][.,;"'?():-_`]  # these are separate tokens
... '''
>>> print nltk.regexp_tokenize(raw, pattern)
['Three Is', 'raelis ha', 've be', 'en ki', 'lled an', 'd mo', 're th', 'an 20', 'i
njured in', 'shooting an', 'd st', 'abbing at', 'tacks in', 'Jerusalem an', 'd ce',
'ntral Is', 'rael, I', 'sraeli po', 'lice sa', 'y.\nT', 'wo we', 're ki', 'lled wh'
, 'en tw', 'o as', 'sailants, w', 'ho we', 're id', 'entified as', 'Arabs, s', 'hot
an', 'd st', 'abbed pa', 'ssengers on', 'a bu', 's in', 'Jerusalem be', 'fore be',
'ing sh', 'ot by', 'police.\nA', 'nother Is', 'raeli di', 'ed af', 'ter be', 'ing r
u', 'n do', 'wn an', 'd st', 'abbed el', 'sewhere in', 'the ci', 'ty.\nN', 'ear-dai
ly st', 'abbings by', 'Palestinians ha', 've le', 'ft do', 'zens of', 'Israelis de'
, 'ad an', 'd wo', 'unded ov', 'er th', 'e pa', 'st fo', 'rtnight.\nS', 'everal at'
, 'tackers an', 'd at', 'least 17', 'other Pa', 'lestinians ha', 've be', 'en ki',
'lled in', 'the up', 'surge of', 'violence.\nI', 'sraeli Pr', 'ime Mi', 'nister Be'
, 'njamin Ne', 'tanyahu co', 'nvened an', 'emergency se', 'ssion of', 'the se', 'cu
rity ca', 'binet to', 'discuss ho', 'w to', 'prevent fu', 'rther at', 'tacks.']
```

11. Написати функцію unknown(), яка приймає інтернет адресу як аргумент і повертає не відомі слова, які зустрічаються в тексті. При розробці функції використовувати re.findall() для виявлення всіх підстрічок та корпус Words Corpus (nltk.corpus.words) для виявлення не відомих слів.

```
import nltk
import re
import urllib, nltk
from urllib import urlopen
def unknown(url):
    html=urlopen(url).read()
    raw=nltk.clean_html(html)

    unknown_words=nltk.corpus.words.words()
    if re.findall(r'[A-Za-z]*', raw) not in unknown_words:
        return re.findall(r'[A-Za-z]*',raw)
print unknown("http://www.bbc.com/news/world-europe-34501617")
```

```
['Turkish', '', 'air', '', 'strikes', '', 'on', '', 'Kurdish', '', 'PKK', '', 'r
ebels', '', 'as', '', 'mourning', '', 'continues', '', '', '', 'BBC', '', 'News'
, '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', ''
, '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', ''
, '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', ''
, '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', ''
, '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', ''
, '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', ''
, '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', ''
, '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', ''
, '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', ''
, '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', ''
, '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', ''
, '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', ''
, '', '', '', '', '', '', '', '', 'Accessibility', '', 'links', '', '', 'Skip',
'', 'to', '', 'content', '', '', 'Accessibility', '', 'Help', '', '', '', '', ''
, '', 'BBC', '', 'iD', '', '', '', '', '', '', '', '', '', 'BBC', '', 'navigatio
n', '', '', '', 'News', '', '', '', 'News', '', '', '', 'Sport', '', '', '', 'We
ather', '', '', '', 'Shop', '', '', '', 'Earth', '', '', '', 'Travel', '', '', '
', 'Capital', '', '', '', 'iPlayer', '', '', '', 'Culture', '', '', '', 'Autos',
 '', '', '', 'Future', '', '', '', 'TV', '', '', '', 'Radio', '', '', '', 'CBBC'
, '', '', '', 'CBeebies', '', '', '', 'Food', '', '', '', 'Arts', '', '', '', 'M
```

15. 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 gutenberg, nps_chat
text = nltk.Text(gutenberg.words('melville-moby_dick.t
print text.findall(r'<the> <best> <\w*> <can>')
print text.findall(r'<as> <best> <\w*> <can>')
print text.findall(r'<as> <best> <as> <\w*> <can>')
print text.findall(r'<the> <best> <that> <\w*> <can>')
chat=nltk.Text(nps_chat.words())
print chat.findall(r'<the> <best> <\w*> <can>')
print text.findall(r'<as> <best> <\w*> <can>')
print text.findall(r'<as> <best> <as> <\w*> <can>')
print text.findall(r'<the> <best> <that> <\w*> <can>')
```

```
the best we can
None

None

None

None

None

None

None

None
```

Висновок: під час виконання цієї лабораторної роботи я навчилася використовувати регулярні вирази для обробки текстів.