



Міністерство освіти і науки України

Національний технічний університет України

“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

## **Лабораторна робота №2**

**Аналіз текстів з використанням мови Python**

**Тема:** Попередня обробка тексту за допомогою NLTK

**Варіант:** 1

Виконав

студент групи ІП-11:

Панченко С. В.

Перевірів:

Тимофєєва Ю. С

Київ 2023

## ЗМІСТ

1 Мета лабораторної роботи.....	6
2 Завдання.....	7
3 Виконання.....	8
3.1 Перше завдання.....	8
3.2 Друге завдання.....	15
4 Висновок.....	18

## 1 МЕТА ЛАБОРАТОРНОЇ РОБОТИ

Ознайомитись з представленням тексту Python в та регулярними виразами.

## 2 ЗАВДАННЯ

Створити програму, яка виконує завдання відповідно до варіанту, використовуючи бібліотеку NLTK:

Варіант 1.

Зчитати файл text1. а) Порахувати кількість речень в тексті; б) вивести 10 слів, які зустрічаються найчастіше; в) провести лематизацію слів третього речення.

## 3 ВИКОНАННЯ

### 3.1 Перше завдання

Для початку завантажимо текст, імпортуємо модуль `nltk`.

```
In [231]: import nltk
with open('text1.txt', 'r') as file:
    text = ''.join(file.readlines())
print(text)
```

Isa Whitney, brother of the late Elias Whitney, D.D., Principal of the Theological College of St. George's, was much addicted to opium. The habit grew upon him, as I understand, from some foolish freak when he was at college; for having read De Quincey's description of his dreams and sensations, he had drenched his tobacco with laudanum in an attempt to produce the same effects. He found, as so many more have done, that the practice is easier to attain than to get rid of, and for many years he continued to be a slave to the drug, an object of mingled horror and pity to his friends and relatives. I can see him now, with yellow, pasty face, drooping lids, and pin-point pupils, all huddled in a chair, the wreck and ruin of a noble man.

One night—it was in June, '89—there came a ring to my bell, about the hour when a man gives his first yawn and glances at the clock. I sat up in my chair, and my wife laid her needle-work down in her lap and made a little face of disappointment.

Рисунок 3.1 - Завантаження файлу

Розіб'ємо текст на речення за допомогою функції `tokenize.sent_tokenize`.

```
In [232]: sentences = nltk.tokenize.sent_tokenize(text)
sentences
```

```
Out[232]: ['Isa Whitney, brother of the late Elias Whitney, D.D., Principal of the\nTheological Col\nlege of St. George's, was much addicted to opium.',
'The\nhabit grew upon him, as I understand, from some foolish freak when he\nwas at coll\nege; for having read De Quincey's description of his dreams\nand sensations, he had drenc\nhed his tobacco with laudanum in an attempt\nto produce the same effects.',
'He found, as so many more have done, that\nthe practice is easier to attain than to get\nrid of, and for many years\nhe continued to be a slave to the drug, an object of mingled\nhorror\nand pity to his friends and relatives.',
'I can see him now, with yellow,\npasty face, drooping lids, and pin-point pupils, all h\nuddled in a\nchair, the wreck and ruin of a noble man.',
'One night—it was in June, '89—there came a ring to my bell, about the\nhour when a man\ngives his first yawn and glances at the clock.',
'I sat up\nin my chair, and my wife laid her needle-work down in her lap and made\na lit\ntle face of disappointment.']
```

Рисунок 3.2 - Розбиття тексту на речення

Підрахуємо їхню кількість.

```
In [233...] len(sentences)
```

```
Out[233]: 6
```

Рисунок 3.3 - Кількість речень у тексті

Розділимо текст на слова за допомогою функції `nlk.tokenize.regexp_tokenize`. Використаємо шаблон розділення як `r'\w+'` для розбиття речення на незалежні алфавітні токени. Використаємо `nlk.FreqDist` для знаходження частоти tokenів.

```
In [234...] import pandas as pd
words = nltk.tokenize.regexp_tokenize(text, pattern=r'\w+')
df = pd.DataFrame(nltk.FreqDist(words).items(), columns=['words', 'counts'])
df.set_index('words', inplace=True)
df
```

```
Out[234]:
```

	counts
words	

Рисунок 3.4 - Розбиття тексту на слова

За допомогою `idxmax` виведемо найбільш уживане слово та кількість разів його використання.

```
In [235...] df.counts.idxmax(), df.loc[df.counts.idxmax(), 'counts']
```

```
Out[235]: ('and', 9)
```

Рисунок 3.5 - Найбільш уживане слово

Виділимо слова третього речення, приберемо повтори.

```
In [236... words = nltk.tokenize.regexp_tokenize(sentences[2], pattern='\w+')
df = pd.DataFrame(set(words), columns=['words'])
df
```

```
Out[236]:
```

	words
0	for
1	easier
2	a
3	pity
4	years
5	of
6	many
7	is
8	slave
9	found
10	he
11	be
12	have
13	an
14	his
15	so
16	practice
17	relatives
18	rid
19	to
20	attain
21	that
22	and
23	get
24	more
25	mingled
26	drug
27	continued
28	friends
29	the
30	horror
31	He
32	than
33	as
34	object
35	done

Рисунок 3.6 - Слова третьего речення

Визначимо частину мови для кожного слова. Використаємо функцію `nltk.pos_tag`.

```
In [237]: df['ps'] = [tag for _, tag in nltk.pos_tag(df.words)]
df
```

```
Out[237]:
```

	words	ps
0	for	IN
1	easier	JJR
2	a	DT
3	pity	NN
4	years	NNS
5	of	IN
6	many	JJ
7	is	VBZ
8	slave	JJ
9	found	VBN
10	he	PRP
11	be	VB
12	have	VBP
13	an	DT
14	his	PRP\$
15	so	NN
16	practice	NN
17	relatives	NNS
18	rid	VBP
19	to	TO
20	attain	VB
21	that	DT
22	and	CC
23	get	VB
24	more	JJR
25	mingled	JJ
26	drug	NN
27	continued	VBD
28	friends	VBZ
29	the	DT
30	horror	NN
31	He	PRP
32	than	IN
33	as	IN
34	object	JJ
35	done	VBN

Рисунок 3.7 - Визначення частини мови

Узагальнимо частини мови до звичайних: noun, adjective, verb тощо.



```
In [238... from nltk.tag import map_tag
df['sps'] = [map_tag('en-ptb', 'universal', tag) for tag in df.ps]
df
```

```
Out[238]:
```

	words	ps	sps
0	for	IN	ADP
1	easier	JJR	ADJ
2	a	DT	DET
3	pity	NN	NOUN
4	years	NNS	NOUN
5	of	IN	ADP
6	many	JJ	ADJ
7	is	VBZ	VERB
8	slave	JJ	ADJ
9	found	VBN	VERB
10	he	PRP	PRON
11	be	VB	VERB
12	have	VBP	VERB
13	an	DT	DET
14	his	PRP\$	PRON
15	so	NN	NOUN
16	practice	NN	NOUN
17	relatives	NNS	NOUN
18	rid	VBP	VERB
19	to	TO	PRT
20	attain	VB	VERB
21	that	DT	DET
22	and	CC	CONJ
23	get	VB	VERB
24	more	JJR	ADJ
25	mingled	JJ	ADJ
26	drug	NN	NOUN
27	continued	VBD	VERB
28	friends	VBZ	VERB
29	the	DT	DET
30	horror	NN	NOUN
31	He	PRP	PRON
32	than	IN	ADP
33	as	IN	ADP
34	object	JJ	ADJ
35	done	VBN	VERB

Рисунок 3.8 - Узагальнення чатин мов

Перетворимо узагальнені частини мови на абревіатури для лематизації.

```
In [239]: abbr = {'NOUN': 'n', 'VERB': 'v', 'ADJ': 'a', 'ADV': 'r'}
def to_abbr(el):
    res = abbr.get(el)
    if res is not None:
        return res
    return ''

df['abbr'] = [to_abbr(x) for x in df['sps']]
df
```

```
Out[239]:
```

	words	ps	sps	abbr
0	for	IN	ADP	
1	easier	JJR	ADJ	a
2	a	DT	DET	
3	pity	NN	NOUN	n
4	years	NNS	NOUN	n
5	of	IN	ADP	
6	many	JJ	ADJ	a
7	is	VBZ	VERB	v
8	slave	JJ	ADJ	a
9	found	VBN	VERB	v
10	he	PRP	PRON	
11	be	VB	VERB	v
12	have	VBP	VERB	v
13	an	DT	DET	
14	his	PRP\$	PRON	
15	so	NN	NOUN	n
16	practice	NN	NOUN	n
17	relatives	NNS	NOUN	n
18	rid	VBP	VERB	v
19	to	TO	PRT	
20	attain	VB	VERB	v
21	that	DT	DET	
22	and	CC	CONJ	
23	get	VB	VERB	v
24	more	JJR	ADJ	a
25	mingled	JJ	ADJ	a
26	drug	NN	NOUN	n
27	continued	VBD	VERB	v
28	friends	VBZ	VERB	v
29	the	DT	DET	
30	horror	NN	NOUN	n
31	He	PRP	PRON	
32	than	IN	ADP	
33	as	IN	ADP	
34	object	JJ	ADJ	a
35	done	VBN	VERB	v

Рисунок 3.9 - Приведення загальних частин мов до абревіатур

Проведемо лематизацію кожного слова за допомогою методу `lemmatize` об'єкта класу `nlk.stem.WordNetLemmatizer`.

```
In [240]: from nltk.stem import WordNetLemmatizer
wlem = WordNetLemmatizer()
def my_lem(word, abbr):
    if abbr == '':
        return wlem.lemmatize(word)
    return wlem.lemmatize(word, pos=abbr)
df['lemms'] = [my_lem(row[1]['words'], row[1]['abbr'])
               for row in df.loc[:, ['words', 'abbr']].iterrows()]
df
```

```
Out[240]:
```

	words	ps	sps	abbr	lemms
0	for	IN	ADP		for
1	easier	JJR	ADJ	a	easy
2	a	DT	DET		a
3	pity	NN	NOUN	n	pity
4	years	NNS	NOUN	n	year
5	of	IN	ADP		of
6	many	JJ	ADJ	a	many
7	is	VBZ	VERB	v	be
8	slave	JJ	ADJ	a	slave
9	found	VBN	VERB	v	find
10	he	PRP	PRON		he
11	be	VB	VERB	v	be
12	have	VBP	VERB	v	have
13	an	DT	DET		an
14	his	PRP\$	PRON		his
15	so	NN	NOUN	n	so
16	practice	NN	NOUN	n	practice
17	relatives	NNS	NOUN	n	relative
18	rid	VBP	VERB	v	rid
19	to	TO	PRT		to
20	attain	VB	VERB	v	attain
21	that	DT	DET		that
22	and	CC	CONJ		and
23	get	VB	VERB	v	get
24	more	JJR	ADJ	a	more
25	mingled	JJ	ADJ	a	mingled
26	drug	NN	NOUN	n	drug
27	continued	VBD	VERB	v	continue
28	friends	VBZ	VERB	v	friends
29	the	DT	DET		the
30	horror	NN	NOUN	n	horror
31	He	PRP	PRON		He
32	than	IN	ADP		than
33	as	IN	ADP		a
34	object	JJ	ADJ	a	object
35	done	VBN	VERB	v	do

Рисунок 3.10 - Лематизація слів

Бачимо, що під час лематизації та тегування сталася одна помилка: "friends" - це "noun", або іменник, тому мало б бути "friend". Однак, "mingled" правильно ідентифікувало, адже це дієприкметник, тобто спрощено "ajective".

## 3.2 Друге завдання

Використаємо корпус brown. Виведемо перші 5 речень.

```
In [241]: from nltk.corpus import brown
text = [' '.join(sent) for sent in brown.sents(categories='fiction')[:5]]
text

Out[241]: ['Thirty-three',
'Scotty did not go back to school .',
'His parents talked seriously and lengthily to their own doctor and to a specialist at t
he University Hospital -- Mr. McKinley was entitled to a discount for members of his fami
ly -- and it was decided it would be best for him to take the remainder of the term off ,
spend a lot of time in bed and , for the rest , do pretty much as he chose -- provided ,
of course , he chose to do nothing too exciting or too debilitating .',
'His teacher and his school principal were conferred with and everyone agreed that , if
he kept up with a certain amount of work at home , there was little danger of his losing
a term .',
'Scotty accepted the decision with indifference and did not enter the arguments .']
```

Рисунок 3.11 - Виведення перших п'яти речень

Виведемо протеговані слова з категорії "fiction".

```
In [242]: words_tags = brown.tagged_sents(categories='fiction')
words_tags_flat = []
for sent in words_tags:
    for w_t in sent:
        words_tags_flat.append(w_t)
df = pd.DataFrame(words_tags_flat, columns=['words', 'ps'])
df
```

```
Out[242]:
```

	words	ps
0	Thirty-three	CD-HL
1	Scotty	NP
2	did	DOD
3	not	*
4	go	VB
...	...	...
68483	heart	NN
68484	almost	RB
68485	stopped	VBD
68486	beating	VBG
68487	.	.

68488 rows × 2 columns

Рисунок 3.12 - Протеговані слова

Узагальнимо частини мови.

```
In [243]: df['sps'] = [map_tag('en-ptb', 'universal', tag) for tag in df.ps]
df
```

```
Out[243]:
```

	words	ps	sps
0	Thirty-three	CD-HL	X
1	Scotty	NP	NOUN
2	did	DOD	X
3	not	*	X
4	go	VB	VERB
...	...	...	...
68483	heart	NN	NOUN
68484	almost	RB	ADV
68485	stopped	VBD	VERB
68486	beating	VBG	VERB
68487	.	.	.

68488 rows × 3 columns

Рисунок 3.13 - Узагальнення частин мови

Виділимо лише іменники, згрупуємо їх та застосуємо метод count для підрахунку кожного.

```
In [244]: df = df[df.sps == 'NOUN'].groupby(by='words')['words'].count()
df
```

```
Out[244]: words
$2,500      1
$500        1
8-Balls     1
Aaron        1
Abbe         1
..          ..
yelling      1
youth        5
zenith        1
zlotys        1
zoo           2
Name: words, Length: 4198, dtype: int64
```

Рисунок 3.14 - Підрахунок іменників

Визначимо десять іменників, що зустрічаються найчастіше.

```
In [246]: df.sort_values(ascending=False, inplace=True)  
df.head(10)
```

```
Out[246]: words  
man      111  
time      99  
men       72  
room      63  
way       62  
eyes      60  
face      55  
house     54  
head      54  
night     53  
Name: words, dtype: int64
```

Рисунок 3.15 - Іменники, що зустрічаються найчастіше

## 4 ВИСНОВОК

Під час виконання даної лабораторної роботи я ознайомився з представленням тексту Python в та регулярними виразами.

У першому завданні було розбито текст на речення, підраховано їхню кількість, розділено на слова та підраховано кількість слів. Також було проведено лематизацію.

У другому завданні зчитано перші п'ять речень відповідної категорії та виведено десять найбільш уживаних іменників.