

```
ëëËËË444€€€ÏÏÏÏÏÏÏ7777
old txt= ëëËËË444€€€ÏÏÏÏÏÏÏ7777
new txt= &&&&444€€€&&&&&7777
введіть текст (Enter - stop pgm):
```

Лабораторна робота №3_(ру-3)

Обробка текстової інформації.

У наведеному нижче завдання необхідно:

1. читати вихідний текст, символні рядки тощо. із зовнішнього символного файлу.
2. результат виконання програми виводити в новий символний файл та/або на консоль.

Примітка

- Ви можете скористатися будь-яким текстовим редактором, щоб отримати вихідний символний файл.

-Ви можете "взяти" файл з "інтернету"

Завдання лабораторної роботи

З'ясувати:

1. скільки слів у тексті;
2. скільки літер у тексті;
3. визначити скільки разів у тексті зустрічається задана літера;
4. визначити скільки разів у тексті зустрічається задане слово;

=====

5. побудувати таблицю частот літер та символів тесту (кілька разів у тексті зустрічається кожна літера алфавіту);

6. побудувати таблицю частот слів (кілька разів у тексті зустрічається кожне слово з тексту);

Приклад вирішення варіанта роботи

```
def toster_word_split(s, SEPARATORS=' \n\r\t.,;?!\\t:'):
    """
    Получения списка слов,
    разделенных заданными разделителями
    из входной строки s получаем list слов,
    разделённых символами разделителями из строки SEPARATORS
    """
    result=[]
    current_word=''
    for char in s:
        if char in SEPARATORS:
            if current_word:
                result.append(current_word)
                current_word=''
            else:
                current_word+=char
        else:
            current_word+=char
    if current_word:
        result.append(current_word)
    return result
# тест функции
print('w1=toster_word_split("слово1      слово2....слово3!!
44444;;;55555")')
w1=toster_word_split("слово1      слово2....слово3!!
44444;;;55555")
print(w1)
#####
def toD(dic, key):
    """
    словарь - это много пар (ключ : значение)
    ключ key      - это слово (или буква)
    значение item - это целое число - сколько раз встретилось
    слово

    в словарь dic (key : item ) помещаем
    key-это слово или буква : +1 к старому значению item
    """
    if key in dic:      # есть ли слово в словаре
        dic[key]+=1    # если да, то к значению прибавим 1
    else:
        dic[key]=1     # если слова в словаре нет, то поместим
        (слово : 1)
    #####
# тест функции
txt='111 222 111 222 333 444;444;555.555'
print(txt)
words=toster_word_split(txt) # получили список слов
i=0
for w in words:
    i+=1
    print(str(i), '\\t', w)
#####
```

```

import sys
# Программа подсчета слов в файле
# простейшая версия без доп. проверок
filename="labirint0.txt" #текст в кодировке windows
#filename="test_text.txt" #текст в кодировке windows
try:
    # открываем файл
    h=open(filename, encoding="cp1251") #текст в кодировке
windows
except Exception as e:
    print('e=',e)
    print('ошибка', e, '\nоткрытие файла ', filename)
    sys.exit(4)
i=0 # счетчик строк
dword= {} # словарь частоты слов (слово : количество)
k_word=0 # счетчик слов
dleter={ } # словарь частоты букв ( буква : количество)
kleter = 0 # счетчик букв
while True:
    # читаем очередную строку в переменную txt
    txt=h.readline()
    if len(txt) == 0: # строки в файле закончились
        break
    txt1=txt[:-1] # убираем "переводы строк" и строку в txt1
    i+=1
    #print(i, '\t строка: <',txt1,'>') # для отладка можно
раскомментировать
    # заполняем словари
    words=toster_word_split(txt1) # получили список слов в
очередной записи
    for w in words: # для каждого слова (слово в переменной
w)
        k_word+=1 # считаем слова
        toD(dword,w) # слово в словарь dword
        # именно СДЕСЬ можно СРАВНИТЬ очередное слово (находится в
переменной w)
        # с ЗАДАННЫМ словом и в случае совадения увеличить
соответствующий счетчик
        for let in txt: # для каждой буквы из строки ( буква в
переменной let)
            kleter+=1 # считаем буквы
            toD(dleter,let) # буква в словарь deleter
            # именно СДЕСЬ можно СРАВНИТЬ очередную букву (находится в
переменной let)
            # с ЗАДАННЫМ словом и в случае совадения увеличить
соответствующий счетчик
h.close()
#1 print(dword)
#1 print(dleter)
#1 print('\n\ndword:')
#печать таблицей dword
#print('количество \t слово')
#1 for kk, vv in dword.items():
#1 print(vv, '\t',kk)
#можно напечатать статистику
print("В файле ", filename)
print("Кол-во букв: %d" % kleter)
print("Кол-во уникальных букв : %d" % len(dleter))
print('\n\ndleter:')
#печать таблицей deleter (красиво)

```

```

print('символ \t количество % содержания')
for kk, vv in dleter.items():
    if (kk == '\n'):
        kk=r'\n'
    elif (kk == '\r'):
        kk=r'\r'
    elif (kk == '\t'):
        kk=r'\t'
    elif (kk == ' '):
        kk=r' |'
    else:
        kk=' '+kk
    print(kk, '\t', vv, '\t', "{0:6.2f}%".format(100*vv/k_leter)
)
#####
#можно напечатать статистику
print("В файле ", filename)
print("Кол-во слов: %d" % k_word)
print("Кол-во уникальных слов: %d" % len(dword))

print("Все использованные слова:") # это для красоты....
####for word in dword:
####    print(word.ljust(20), words_dict[word])
#сформируем список D1 из словаря и
# и его сортируем по ключу словаря key=lambda x:x[0]
# если хотим по значению, то пишите key=lambda x:x[1]
# если хотим в обратном порядке, то пишите key=lambda x:-x[1]
D1=sorted(dword.items(),key=lambda x:-x[1]) # по ключам
#print(D1) # печать списка
# отсортированный список в словарь D2 и его печать
D2={D1[i][0] : D1[i][1] for i in range(len(D1)) }
#печать таблицей D2
for kk, vv in D2.items():
    print(vv, '\t', kk)

```

Приклад 1

Виведення таблиці символів (Python)

У Python v.3 для рядків використовується кодування Unicode. (Слід пам'ятати, що в Python, на відміну від інших мов програмування, взагалі немає такого типу як одиночний символ; будь-який символ це рядок, довжина якого дорівнює 1.)

Перші 128 символів за таблицею Unicode такі самі як і таблиці символів ASCII. Виведемо їх (починаючи з пробілу – 32-й символ). Щоб привести висновок до табличної форми, переходитимемо на новий рядок після кожного десятого символу (інструкція if у коді нижче).

Функція chr() повертає символ із таблиці Unicode, що відповідає переданому коду-числу.

```

For i in range(32,128):

```

```

print(chr(i), end='')
if (i-2) %10 == 0:
    print()

print()

```

Результат виконання коду:

```

! " # $ % & ' ( ) *
+ , - . / 0 1 2 3 4
5 6 7 8 9 : ; < = >
? @ABCDEFGHIJ
KLMNOPQR
STUVWXYZ [ \
] ^ _ ` abcdef
ghijklmnop
qrstuvwxyz
{| } ~

```

Але припустимо, нам захотілося чи знадобилося дізнатися коди символів російських букв (кирилиці). Таблиця Unicode дуже велика і включає майже всі алфавіти Землі. Однак припустимо, що кирилиця має бути закодована десь на початку таблиці (адже російська - одна з найпоширеніших мов світу). Переберемо коди символів від 256 до 10000, і якщо який-небудь код із цього діапазону відповідає російській літері (великій або малій), то виведемо на екран сам код і літеру, якій він відповідає.

```

foriin range(256,10000):
    if 'a'<=chr(i)<='я' or 'A'<=chr(i)<='Я':
        print(i, '-', chr(i))

```

Результат виконання коду:

```

1040 - А
1041 - Б
1042 - В
1043 - Г
1044 - Д
1045 - Е
1046 - Ж
1047 - З
1048 - І
1049 - Й
1050 - До
1051 - Л
1052 - М
1053 - Н
1054 - Про
1055 - П
1056 - Р
1057 - З
1058 - Т
1059 - У

```

1060 - Ф
1061 - Х
1062 - Ц
1063 - Ч
1064 - Ш
1065 - Щ
1066 - Ъ
1067 - Ы
1068 - Ь
1069 - Е
1070 - Ю
1071 - Я
1072 - а
1073 - б
1074 - в
1075 р
1076 - д
1077 - е
1078 - ж
1079 - з
1080 - і
1081-й
1082 - до
1083 - л
1084 - м
1085 - н
1086 - про
1087 - п
1088 - р
1089 - з
1090 - т
1091 - у
1092 - ф
1093 - х
1094 - ц
1095 - год
1096 - ш
1097 - щ
1098 - ъ
1099 -
1100 - ь
1101 - е
1102 - ю
1103 - я

Тепер ми знаємо коди російських букв за таблицею Unicode. Але висновок вийшов якийсь некомпактний. До того ж бачимо, що літери алфавіту йдуть одна одною. Тому достатньо спочатку дізнатися тільки код першої великої літери алфавіту (великі символи йдуть завжди попереду малих) і код останньої маленької літери алфавіту. Крім того, якщо ми знайшли коди символів, то нема чого далі продовжувати цикл. Тому перепишемо програму так:

```
first = 0
last = 0
foriin range(255,10000):
    if chr(i)=='А':
        first = i
    elif chr(i)=='я':
        last = i
    break вихід з циклу
```

```

j = 0
for i in range(first, last+1):
    print(i, '-', chr(i), end='')
    j += 1
    if j%10 == 0: print()

print()

```

Результат виконання коду:

```

1040 - А 1041 - Б 1042 - У 1043 - Г 1044 - Д 1045 - Е 1046 - Ж 1047 - З 1048
- І 1049 - Й
1050 - К 1051 - Л 1052 - М 1053 - Н 1054 - Про 1055 - П 1056 - Р 1057 - З
1058 - Т 1059 - У
1060 - Ф 1061 - Х 1062 - Ц 1063 - Ч 1064 - Ш 1065 - Щ 1066 - Ъ 1067 - Ы 1068
- Ь 1069 - Е
1070 - Ю 1071 - Я 1072 - а 1073 - б 1074 - у 1075 - г 1076 - д 1077 - е 1078
- ж 1079 - з
1080 - і 1081 - й 1082 - до 1083 - л 1084 - м 1085 - н 1086 - про 1087 - п
1088 - р 1089 - с
1090 - т 1091 - у 1092 - ф 1093 - х 1094 - ц 1095 - ч 1096 - ш 1097 - щ 1098
- ъ 1099 - ы
1100 - 1101 - е 1102 - ю 1103 - я

```

Вивчіть текст 2

Приклади 2 "цікави" програми

1. Частота букв у тексті

```

text = 'hello world Привет ЙёЁй hello world Привет ЙёЁй '
unique_letters = set(text)
analyze = {}
for letter in unique_letters:
    analyze[letter] = text.count(letter)
print("1 var ", analyze)
#####

indecies = set(text)
values = (text.count(letter) for letter in indecies)
analyze = dict(zip(indecies, values))
print("2 var ", analyze)

```

2. Розбиття на слова

```
import re
s='123 4456 678;ffff;5555; zzz'
re.sub('\W', ' ', s).split()
#####

SEPARATORS=".,;?! "
def toster_word_split(s):
    result=[]
    current_word=''
    for char in s:
        if char in SEPARATORS:
            if current_word: result.append(current_word)
            current_word=''
        else:
            current_word+=char
    if current_word:
        result.append(current_word)
    return result
print( toster_word_split("Lorem,ipsum;bingo.Bongo? King of
Kongo.") )
#####

def msplit(str, raz=' '):
    ''' возвращает список слов из str, разделённых raz '''
    import re
    __r="["
    for __b in raz:
        __r=__r + __b + "|"
    #print(__b)
    __r=__r[:-1]+"]"
    #print("==>", __r)
    #return
    __split_regex = re.compile(__r) ###r'[.?!|?|...]'
    __sentences = filter(lambda t: t, [t.strip() for t in
__split_regex.split(str)])
    __w1=[]
    for __s in __sentences:
        __w1.append(__s)
    return __w1
```

3. Читаемо/пишемо у файл

```
# приклад 1 відкриття файлу та перебору рядків у ньому
try:
f = open("file1.py", 'r', encoding='UTF-8')
```

```

except IOError:
print ("No file")
exit(0)
k=0
for line in f.readlines():
    #print (line[:-1])
    k+=1
    k1=0
    print(k, line.rstrip('\n') )
    #for w in line[:-1].split(' '):
        # k1+=1
        # print(k, k1, w)
f.close()
###exit()
### приклад 2 підрахунок рядків коментарів
col = 0
coll = 0
f1=open("file1.py", "r", encoding='UTF-8')
for x in f1.readlines():
    coll+=1
    if '#' in x:
        col += 1
        continue
print("Количество строк с коментариями: ", col, "\n количество
строк: ", coll)
f1.close()
#####exit(0)
# Приклад 3:
with open('file1.py', encoding='utf-8') as f:
nblank, nint = 0, 0
for line in f:
    line = line.rstrip('\n')
    if line in '# ':
        nblank+=1
    for c in line:
        if c in '0123456789':
            nint+=1
print(nblank, nint)
# Приклад 4
import math
def grad_to_rad(grad):
    return grad/360*math.pi*2
f2=open("file1xxx.txt", "w", encoding='UTF-8')
#x=0
for grad in range(0,720,5):
    x=grad_to_rad(grad)
    y,z)=(math.sin(x), math.cos(2*x) )
print(x,y,z)
f2.write(str(x)+' \t'+str(y)+' \t'+str(z)+'\n' )
# x+=(5*math.pi)/360
f2.close()

```

```

f2=open("file1xxx.txt", "r", encoding='UTF-8')
line=f2.readline()
line=line.rstrip('\n')
kk=0
while line:
    kk+=1
    print(kk, '<',line, '>')
    x=float( line.split(" \t")[0] )
    y=float( line.split(" \t")[1] )
    z=float( line.split(" \t")[2] )
    print(x, ' - - ', y, ' - - ', z)
    (x,y,z)=map(float, line.split(" \t") )
    print(x, ' -*- ', y, ' -*- ', z)
    line=f2.readline()
    line=line.rstrip('\n')
f2.close()

```

4. Виконати програму.

Отримання тексту з його появи:

```

cmd = 'ping google.com' # -c 3'
import subprocess
PIPE = subprocess.PIPE
p = subprocess.Popen(cmd, shell=True, stdin=PIPE,
stdout=PIPE,
stderr=subprocess.STDOUT) #, close_fds=True)
while True:
    s = p.stdout.readline()
    if not s: break
    print( s.decode("cp866")

```

Отримання всього результату після завершення програми:

```

cmd = 'dir'
print(cmd)
import subprocess
PIPE = subprocess.PIPE
p = subprocess.Popen(cmd, shell=True, stdin=PIPE,
stdout=PIPE,
stderr=subprocess.STDOUT )
###close_fds=True,
#cwd='/home/'
print( p.stdout.read().decode("cp866") ) #utf-8" )

```

Найпростіший Запуск програми

```

import os

```

```

os.system("dir & pause")
#cmd='cmd'
#os.system(cmd)
#####
import subprocess
cmd = 'ping google.com & pause'
subprocess.Popen(cmd) #, shell = True)

```

```
#####
```

#Приклад виклику зовнішньої програми або команди.

Наступний код:

```

import subprocess
import asyncio, sys
import shlex, subprocess

```

```

cmd='dir ' # echo A; echo B; echo C'
print ('Команда: ',cmd)
p=subprocess.Popen(cmd, stdout=PIPE, shell=True)
stdoutdata, stderrdata = p.communicate()
errorCode=p.returncode
print( 'Результат роботи команди:')
print( stdoutdata.decode("cp866"))
print( 'Вывод ошибок:')
if stderrdata == None:
    print (stderrdata)
else:
    print (stderrdata.decode("cp866") )
    print ('Результат работы команды, построчно:')
    numLine=1
    for line in stdoutdata.decode("cp866").split("\n"):
        print( str(numLine)+' '+line )
        numLine+=1 # инкремент
z=[]
z=stdoutdata.decode("cp866").split("\n")
print('z=')
print(z)
print( 'Количество строк в результате работы команды:
'+str( len(z) ) )
print( 'Количество символов в результате работы команды:
'+str( len(stdoutdata.decode("cp866")) ) )

```

5. Варіанти – розбиття на слова, підрахунок частоти слів

```

import retext = '''Перша пропозиція.
Друга пропозиція!
Третя пропозиція?

```

Четверта пропозиція...
П'ята пропозиція.
П'ята плюс Друга пропозиція
Складне слово якось?
Ще одна пропозиція...
'''

```
import re

print(" ***** по предложениям")

split_regex = re.compile(r'[.?!|?|...|]')

sentences = filter(lambda t: t, [t.strip() for t in
split_regex.split(text)])

sentences1=sentences

for s in sentences:

    print(s)

#####

print("\n ***** по словам ")

split_w = re.compile(r'[""\n"'| |.?!|?|...|]')

sentences = filter(lambda t: t, [t.strip() for t in
split_w.split(text)])

w1=[] # список

w2=set() # множество;

w5=[]

print( "++", sentences)

for s in sentences:

    print(s)

    w1.append(s)
```

```
w2.add(s)

w5.append(s)

#print(w1)

##в список w

print("--", sentences )

for s2 in sentences1:

    print("---- не печатается -----", s2) # фильтр
сработал....

print("до сортировки список")

for ww in w1:

    print("##### ",ww)

w1.sort()

print("после сортировки список")

for ww in w1:

    print(ww)

w3=[]

print("множество до сортировки....")

for ww in w2:

    print("@@@ ",ww)

    w3.append(ww)

w3.sort()

print("множество после сортировки")
```

```
for ww in w3:
    print(ww)

class xxx:
    n=0
    w=""
    def isYESp(wp) :
        ''' поиск и увеличение счетчика'''
        jj=0
        for ww1 in zw:
            if ww1.w == wp:
                zw[jj].n+=1
                return True
            jj+=1
        return False

zw=[] ## =set()
kk=1
for ww in w5:
    print("++++##### ww=",ww, ' kk=',kk)

z=xxx()
z.w=ww
z.n=1
```

```
#if ( isYESp(ww)):  
# print(ww, " --- YES")  
#else:  
# print(ww, " ---NO")  
if ( isYESp(ww)):  
    pass  
else:  
    zw.append(z) ###add(z)  
    kk+=1  
def skey(xs):  
    return -xs.n #xs.n  
  
zw.sort(key=skey)  
print(" частота слов")  
for axxx in zw:  
    print(axxx.n, ' ', axxx.w)  
#z=xxx()  
#z.n=5  
#z.w="qqqq"  
#print(z.n, ' ', z.w)  
print("+++++")  
for www in text.split():  
    print("---> ", www)
```

```
#####  
  
## f = open("C:/war&peace.txt")  
  
arr = []  
  
###for s in f:  
  
t = re.split("[\s;:\-_*\".,?!()]", text) #s)  
  
t = [a for a in t if a != '']  
  
arr.extend(t)  
  
print("----\n", arr)  
  
#####  
  
import sys  
  
pattern = re.compile("([\w]+[-'])*[\w']+?", re.U)  
  
###????line = unicode(text, 'cp1251')  
  
line=text  
  
line = line.replace('--', ' -- ')  
  
for token in line.split(' '):  
  
    m = pattern.match(token)  
  
    if m:  
  
        print( m.group() )
```

Лабораторна робота №4 (ру-4)

Рішення нелінійного рівняння