Лабораторна робота №3(ру-3)

Обробка текстової інформації.

У наведеному нижче завдання необхідно:

1. читати вихідний текст, символьні рядки тощо. із зовнішнього символьного файлу.

2. результат виконання програми виводити в новий символьний файл та/або на консоль.

Примітка

- Ви можете скористатися, наприклад, https://www.ukrlib.com.ua/ для отримання вихідного файлу символу.

- Ви можете скористатися будь-яким текстовим редактором для створення та отримання вихідного символьного файлу.

- Файл повинен містити не менше 1000 слів

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;

==========*(факультативно)*==========

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);

Методичні вказівки.

Приклади роботи з файлами та символами дивись, наприклад, у «Збірник задач і вправ»

Приклад вирішення варіанта роботи

```
def toster_word_split(s, SEPARATORS=' \n\r\t.;,?!\t:'):
    '''
    Отримання списку слів,
    розділених заданими роздільниками
    із вхідного рядка s отримуємо list слів,
    розділених символами роздільниками з рядка 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

    if current_word:
```

```
            result.append(current_word)
    return result
# тест функції
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):
    '''
    словник - це багато пар (ключ: значення)
    ключ key - це слово (або літера)
    значення item – це ціле число – скільки разів зустрілося слово

    до словника dic (key : item )містимо
     key-це слово або літера : +1 до старого значення item
      '''
    if key in dic: # чи є слово у словнику
        dic[key]+=1 # якщо ТАК, то до значення додамо 1
    else:
        dic[key]=1 # якщо слова у словнику немає, то помістимо (слово: 1)
####################################
# тест функції
txt='111 222 111 222 333 444;444;555.555'
print(txt)
words=toster_word_split(txt) # отримали список слів
i=0
for w in words:
    i+=1
    print(str(i),'\t', w)
#######################
import sys
# Програма підрахунку слів у файлі
# Найпростіша версія без доп. перевірок
filename="labirint0.txt" #текст у кодуванні windows
```

```
#filename="test_text.txt" #текст у кодуванні windows
try:
    # відкриваємо файл
    h=open(filename, encoding="cp1251") #текст у кодуванні windows
except Exception as e:
    print('e=',e)
    print('помилка', e, '\nвідкриття файлу', filename)
    sys.exit(4)
i=0          # лічильник рядків
dword= {} # словник частоти слів (слово: кількість)
k_word=0 # лічильник слів
dleter={} # словник частоти букв ( літера : кількість)
k_leter = 0 # лічильник букв
while True:
    читаємо черговий рядок у зміну txt
    txt=h.readline()
    if len(txt) == 0: # рядки у файлі закінчилися
            break
    txt1=txt[:-1] # прибираємо "переклади рядків" та рядок у txt1
    i+=1
    #print(i, '\t рядок: <',txt1,'>') # для налагодження можна розкоментувати
    # заповнюємо словники
    words=toster_word_split(txt1) # отримали список слів у черговому записі
    for w in words: # для кожного слова (слово в змінній w)
        k_word+=1 # вважаємо слова
        toD(dword,w) # слово в словник dword
      # саме ТУТ можна ПОРІВНИТИ чергове слово (знаходиться в змінній w)
      # із ЗАДАНИМ словом і у разі складання збільшити відповідний лічильник
    for let in txt: # для кожної літери з рядка ( літера в змінній let)
        k_leter+=1 # вважаємо літери
        toD(dleter,let) # буква в словник
     # саме ТУТ можна ПОРІВНИТИ чергову букву (знаходиться в змінній let)
     # із ЗАДАНИМ словом і у разі складання збільшити відповідний лічильник
h.close()
```

```
#1 print(dword)
#1 print(dleter)
#1 print('\n\ndword:')
#друк таблицею dword
#print('кількість \t слово')
#1 for kk, vv in dword.items():
#1 print(vv,'\t',kk)
#можна надрукувати статистику
print("У файлі", filename)
print("Кількість букв: %d" % k_leter)
print("Унікальні літери : %d" % len(dleter))
print('\n\ndleter:')
#друк таблицею dleter (красиво)
print('символ \t кількість % змісту')
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'bl'
    else:
        kk='' +kk
    print(kk, '\t',vv, '\t', "{0:6.2f}%".format(100*vv/k_leter) )
########################
#можна надрукувати статистику
print("У файлі", filename)
print("Кількість слів: %d" % k_word)
print("Унікальні слова: %d" % len(dword))

print("Всі використані слова:") # це для краси....
####for word in dword:
#### print(word.ljust(20), words_dict[word])
```

```
#сформуємо список D1 зі словника та
# та його сортуємо за ключом словника key=lambda x:x[0]
# якщо хочемо за значенням, пишіть key=lambda x:x[1]
# якщо хочемо у зворотному порядку, то пишіть key=lambda x:-x[1]
D1=sorted(dword.items(),key=lambda x:-x[1]) # за ключами
#print(D1) # друк списку
# відсортований список у словник D2 та його друк
D2={D1[i][0] : D1[i][1] for i in range(len(D1)) }
#друк таблицею D2
for kk, vv in D2.items():
    print(vv,'\t',kk)
```

Вивчіть текст 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.

```
foriin 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 :; < = >
? @ABCDEFGH
IJKLMNOPQR
STUVWXYZ [ \
] ^ _ ` abcdef
ghijklmnop
qrstuvwxyz
{| } ~
```

Але припустимо, нам захотілося чи знадобилося дізнатися коди символів букв кирилиці. Таблиця Unicode дуже велика і включає майже всі алфавіти Землі. Однак припустимо, що кирилиця має бути закодована десь на початку таблиці. Переберемо коди символів від 256 до 10000), і якщо який-небудь код із цього діапазону відповідає літері кирилиці (великій або малі), 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 кирилиці 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)=='A':
            first = i
      elif chr(i)=='я':
            last = i
            break вихід з циклу

j = 0
foriin range(first,last+1):
      print(i,'-',chr(i), end ='')
      j += 1
      ifj%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 "цікавих" програм

1. Частота букв у тексті

```
text = 'hello world Привіт Їй hello world Привіт Їй '
unique_letters = set(text)
analize = {}
for letter in unique_letters:
  analize[letter] = text.count(letter)
print("1 var ", analize)##########################

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

2. Розбиття на слова

```
import res='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=''):
''' повертає список слів з str, розділених 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. Читаємо/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].
f.close()###exit()
### приклад 2 підрахунок рядків коментарів
col = 0
col1 = 0f1=open("file1.py", "r", encoding='UTF-8')
for x in f1.readlines(): col1+=1 if '#' in x: col += 1 continue
print("Кількість рядків з кометаріями: ", col, "\n кількість рядків: ",
col1)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 line: if c in
'0123456789': nint+=1
print(nblank, nint)
# Приклад 4
import math
def grad_to_rad(grad): return grad/360*math.pi*2f2=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)/360f2.close()f2=open("file1xxx.txt", "r", encoding='UTF
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,', (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 subprocessPIPE = subprocess.PIPEp = 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 subprocessPIPE = subprocess.PIPEp = 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( 'Висновок помилок:')
if stderrdata == None: print (stderrdata)
else: print (stderrdata.decode("cp866") ) print ('Результат роботи команди, рядково:')
інкрементz=[]z=stdoutdata.decode("cp866").split("\n")
print('z=')
print(z)
print( 'Кількість рядків в результаті роботи команди: '+str( len(z) ) )
print( 'Кількість символів в результаті роботи команди: '+str( len(stdoutdata.decode("cp866")) )
)
```

5. Варіанти – розбиття на слова, підрахунок частоти слів

```
import re
text = '''
Перша пропозиція.
Друга пропозиція!
Третя пропозиція?
Четверта пропозиція...
П'ята пропозиція.
П'яте плюс Друга пропозиція
Складне слово якось?
Ще одна пропозиція ... '''
print("***** за пропозиціями")
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***** за словами")
split_w = re.compile(r'[""\n""| |.|!|?|…]')
sentences = filter(lambda t: t, [t.strip() for t in split_w.split(text)])w1=[] # списокw2=set()
# безліч;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 друкується -----", s2) # фільтр спрацював.

print("до сортування список")
for ww in w1: print("##### ",ww) w1.sort()
print("після сортування список")
for ww in w1: print(ww)w3=[]
print("множина до сортування....")
for ww in w2: print("@@@ ",ww) w3.append(ww)w3.sort()
print("множина після сортування")
for ww in w3: print(ww)


class xxx:n=0w=""
def isYESp(wp):
'''пошук та збільшення лічильника'''
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()
```

```
zw=ww
zn=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 слів")
for axxx in zw:
   print(axxx.n,'', axxx.w)#z=xxx()#zn=5#zw="qqqq"#print(zn,'', zw)
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() )
```