Мова програмування ПІТОН (р)

Лабораторні роботи

Лабораторна робота №1(ру-1)

Використовуючи оператор циклу, знайти двома способами суму елементів ряду, зазначеного у конкретному варіанті. Результат надрукувати, забезпечивши відповідним заголовком.

Варіанти

1) Знайти суму ряду з точністю $\varepsilon=10^{-4}$, загальний член якого

$$a_n = \frac{5^n}{n^n}$$

2) Знайти суму з точністю $\varepsilon=10^{-4}$, загальний член якого

$$a_n = \frac{6^n}{n^n}$$

3) Знайти суму ряду з точністю $\varepsilon=10^{-4}$, загальний член якого

$$a_n = \frac{(n!)^2}{2^{n^2}}$$

4) Знайти суму ряду з точністю $\varepsilon=10^{-4}$, загальний член якого

$$a_n = \frac{(-1)^{n-1}}{n^n}$$

5) Знайти суму ряду з точністю $\varepsilon=10^{-4}$, загальний член якого

$$a_n = \frac{n^3}{(3n-3)!}$$

6) Знайти суму ряду з точністю $\varepsilon=10^{-4}$, загальний член якого

$$a_n = \frac{1}{2^n} + \frac{1}{3^n}$$

7) Знайти суму ряду з точністю $\varepsilon=10^{-4}$, загальний член якого

$$a_n = \frac{1}{((3n-2)(3n+1))}$$

8) Знайти суму ряду з точністю $\varepsilon=10^{-4}$, загальний член якого

$$a_n = \frac{(2n-1)}{2^n}$$

9) Знайти суму ряду з точністю $\varepsilon=10^{-4}$, загальний член якого

$$a_n = \frac{10^n}{n!}$$

10) Знайти суму ряду з точністю $\varepsilon=10^{-4}$, загальний член якого

$$a_n = \frac{n!}{(2n)!}$$

11) Знайти суму з точністю $\varepsilon=10^{-4}$, загальний член якого

$$a_n = \frac{5^n}{n^n}$$

12) Знайти суму з точністю $\varepsilon=10^{-4}$, загальний член якого

$$a_n = \frac{10^n}{(2n)!}$$

13) Знайти суму з точністю $\varepsilon=10^{-4}$, загальний член якого

$$a_n = \frac{6^n}{n^n}$$

14) Знайти суму ряду з точністю $\varepsilon=10^{-4}$, загальний член якого

$$a_n = \frac{5}{n^5}$$

15) Знайти суму ряду з точністю $\varepsilon=10^{-4}$, загальний член якого

$$a_n = \frac{5}{n!}$$

Зміст звіту

1. Постановка задачі.

2. Блок схема програми

3. Текст програми.

4. Результат розв'язання конкретного варіанта.

Методичні вказівки

1. При визначенні суми членів ряду першим способом слід використовувати рекурентну формулу для одержання наступного члена ряду. Наприклад, потрібно знайти суму ряду з точністю $\varepsilon=10^{-4}$, загальний член якого

$$a_n = \frac{2(n!)^2}{(3(2n)!)!}$$

Для отримання рекурентної формули обчислимо відношення:

$$\frac{a_{n+1}}{a_n} = \frac{2((n+1)!)^2 \cdot 3(2n)!}{3(2n+2)! \cdot 2(n!)^2} = \frac{n+1}{2(2n+1)},$$

звідки:

$$a_{n+1} = a_n \cdot \frac{(n+1)}{2(2n+1)}.$$

2. При складанні програми вважати, що точність досягнута, якщо черговий доданок abs(a_n) <ε
3. *Другий спосіб*- виконайте підсумовування без застосування рекурентних формул; - розраховуючи значення кожного доданка безпосередньо за заданою за умовою формулою.

Приклад рішення

Знайти суму ряду з точністю Ɛ=10-^{4}, загальний член якого

$$a_n = \frac{1}{n!}$$

Приклад рішення

Виведення рекурентної формули

$$a_{n+1} = \frac{1}{(n+1)!}$$

врахуємо, що

$$(n+1)! = (n+1)n!$$

отримаємо

$$\frac{a_{n+1}}{a_n} = \frac{n!}{(n+1)!} = \frac{n!}{(n+1)n!} = \frac{1}{(n+1)}$$

$$a_{n+1} = a_n \frac{1}{(n+1)}$$

1 спосіб

1 крок: eps=1e-4 s=0, k=0, a=1/0!=1, s=s+a=1

2 крок: поки абсолютна_знач(a) більша за погрішність(eps) ВИП

k=k+1

a=1/fak(k)

s=s+a

якщо k > 10000 зупинитися, напевно ряд не сходиться.

налагоджувальний крок вивести k, a, s

3 крок: висновок s

функція fak(n)

f=1

для i=2 до n вип

f=f*i

повернути f

або

функція fak(n)

якщо n <= 1, то повернути 1

повернути n*fak(n-1)

2 спосіб

1 крок eps=1e-4, s=0, k=0, a=1/0!=1, s=s+a=1

2 крок поки абсолютна_знач(a) більша за похибку ВИП

$$a = a \ \frac{1}{(k+1)}$$

s=s+a

k=k+1

якщо k > 10000 зупинитися, напевно ряд не сходиться.

налагоджувальний крок вивести k, a, s

3 крок виведення s

Текст програм

```
# 1 var

def fak(n : int) -> int:
    if n <=1:
        return 1
    return n*fak(n-1)

eps=1e-4
```

```
s=0; k=0; a=1; s+=a
while abs(a) > eps:
    k+=1
    a=1/fak(k)
    s+=a
    print('{0:3d} {1:16.6g}
{2:16.6g}'.format(k,a,s),sep=' \t')
print('sum=', s)
######################
# 2 var
eps=1e-4
s=0; k=0; a=1; s+=a
while abs(a) > eps:
    a*=1.0/(float(k)+1.0)
    k+=1
    s+=a
    print('{0:3d} {1:16.6g}
{2:16.6g}'.format(k,a,s),sep=' \t')
```

```
======================== RESTART: D:/Te
  1                1                2
  2              0.5              2.5
  3         0.166667          2.66667
  4        0.0416667          2.70833
  5       0.00833333          2.71667
  6       0.00138889          2.71806
  7      0.000198413          2.71825
  8      2.48016e-05          2.71828
sum= 2.71827876984127
  1                1                2
  2              0.5              2.5
  3         0.166667          2.66667
  4        0.0416667          2.70833
  5       0.00833333          2.71667
  6       0.00138889          2.71806
  7      0.000198413          2.71825
  8      2.48016e-05          2.71828
sum= 2.71827876984127
>>>
```

Лабораторна робота №2

Процедури користувача. Обробка символьних рядків.

У наведеному нижче завдання необхідно:

Написати процедуру/функцію користувача мовою програмування Python для виконання завдання, зазначеного у відповідному варіанті.

Обов'язково записати інструкцію щодо використання розробленої процедури/функції та навести приклад її використання (головну тестову програму). Тестова програма повинна вводити з консолі цей текстовий рядок.

Примітка: Використовувати базові операції із символами та рядками.

1. Даний текст. З'ясувати процентний зміст у тексті символів «#» та «@».

2. Даний текст. Чи з'ясувати він є паліндром.

Примітка перед перевіркою, рекомендується викинути з тексту символи ПРОБІЛ.

3. Даний текст містить слова, розділені кількома пробілами. Отримати текст, який містить слова, розділені ОДНИМ пробілом.

4. Даний текст. Отримати рядок, у якому відсутні пробіли, розташовані на початку рядка.

5. Даний текст. Виконати просте шифрування тексту. Кожен символ, код якого є x, замінити символом із кодом F(x). Вибрати F(x)= 32767 - x Тестова програма повинна виконати перекодування вихідного рядка на кодований, потім отриманий кодований рядок «закодувати» ще раз. Порівняйте та поясніть отриманий результат.

6. Даний текст. З'ясувати процентний зміст у тексті літери a та A

7. Даний текст. З'ясувати відсотковий зміст у тексті заданого слова.

8. Даний текст. З'ясувати відсотковий зміст у тексті заданої літери.

9. Даний текст. З'ясувати відсотковий зміст у тексті «здвоєних» прогалин.

10. Даний текст. З'ясувати процентний зміст у тексті символів «%» та «*»

11. Даний текст. З'ясувати відсотковий зміст у тексті символів «&» та «?»

12. Даний текст. З'ясувати процентний зміст у тексті символів «@» та «!»

13. Даний текст. З'ясувати відсотковий зміст у тексті символу «мінус» та «(«

14. Даний текст. З'ясувати відсотковий зміст у тексті символу "(" і ")"

15. Даний текст. З'ясувати відсотковий зміст у тексті символів "цифра в десятковій системі числення"

16. Даний текст. Отримати рядок, у якому відсутні прогалини, розташовані в кінці рядка.

17. Даний текст. З'ясувати, чи цей текст є ідентифікатором змінної;

18. Даний текст. З'ясувати, чи є текст текстовим записом цілого числа.

19. Даний текст. Групи символів, розділені пробілами (одним або декількома) і не містять пробілів у собі, називатимемо словами. Випередити кількість слів у тексті.

20. Даний текст. Якщо в тексті немає символу *, залишити цей текст без зміни, інакше кожну з англійських літер, що передують першому входженню символу *, замінити на символ #.

21. Даний текст. Якщо в тексті немає символу +, то залишити текст без зміни, інакше кожну цифру, що

передує першому входженню символу +, замінити символом @.

22. Даний текст. Замінити у тексті усі літери Я літерою Q.

23. Даний текст. Замінити у тексті всі літери англійського алфавіту символом ТІЛЬДА.

24. Даний текст. Замінити у тексті всі символи «цифра» символом #

25. Даний текст. Замінити у тексті всі ЗАДАННІ у вхідному рядку символи символом &

26. Даний текст. Замінити текст Задане слово на символ *

27. Даний текст. Замінити в тексті Задане(1) слово на задане(2) слово

28. Даний текст. Кожне слово тексту записати у дзеркальному вигляді

29. Даний текст. Знайти скільки разів у тексті зустрічається задане поєднання букв

30. Даний текст; визначити, чи містить він символи, відмінні від англійських літер та пробілу.

Приклад рішення

```
# вариант 25
# Дан текст. Заменить в тексте все ЗАДАННЫЕ
# в входной строке символы символом &.
def schange(text_in, str_change):
    """
    В текстовой строке text_in заменяются все
    символы, которые записаны в заданной строке
str_change
    на символ &
    """
    str=""
    for st in text_in:
        out_str=st
        for ch in str_change:
            if ch==out_str:
                out_str='&'
                break
        str+=out_str
    return str
#####################################################
ch=input("введите заменяемые в строке символы: ")
while True:
    txt=input("введите текст (Enter - stop pgm):\n")
    if len(txt)==0:
        break
    txt1=schange(txt, ch)
    print("old txt=",txt)
    print("new txt=",txt1)

введите заменяемые в строке символы: ЁёiIЇї5
введите текст (Enter - stop pgm):
```

```
ёёЁЁЁ444ЄЄЄЇЇїїїї7777
old txt= ёёЁЁЁ444ЄЄЄЇЇїїїї7777
new txt= &&&&&444ЄЄЄ&&&&&&&7777
введите текст (Enter - stop pgm):
```

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 будь-яким текстовим редактором, щоб отримати вихідний символьний 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

    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, '\nоткрытия 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)
k_leter = 0   #  счетчик букв
while True:
    # читаем очередную строку  в переменую 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
      # имеено 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)
        k_leter+=1       # считаем буквы
        toD(dleter,let) # буква в словарь dleter
      # имеено 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" % k_leter)
print("Кол-во уникальных букв : %d" % len(dleter))
print('\n\ndleter:')
#печать таблицей dleter (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'bl'
    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 :; < = >
? @ABCDEFGH
IJKLMNOPQR
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 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 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)=='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 2
"цікави" програми

1. Частота букв у тексті

```
text = 'hello world Привет ЙёЁй hello world Привет ЙёЁй '
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 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. 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].split(' '):
     # k1+=1
     # print(k, k1, w)
f.close()
###exit()
```
приклад 2 підрахунок рядків коментарів
```
col = 0
col1 = 0
f1=open("file1.py", "r", encoding='UTF-8')
for x in f1.readlines():
   col1+=1
   if '#' in x:
     col += 1
     continue
print("Количество строк с кометариями: ", col, "\n количество
строк: ", 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 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 работы команды:')
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

Необхідно використовуючи пакет matplotlib і можливо numpy графічно дослідити рішення нелінійного рівняння і отримати все (якщо можливо) рішення. Зберегти малюнки графіків на окремий файл. Вставити цей файл у звіт роботи

Рішення уточнити шляхом половинного поділу з точністю $eps=10^{-8}$. Для цього скласти програму.

Варіанти

1) $\sqrt{(25 - x^2)} = arctg\ (2x)$

2) $(x^3 - 2x^2 - 3x + 4) = 2\cos(2x + 1)$

3) $arctg\,2x - \frac{(x-1)^4}{(5)} + \sin^2(5x) = 0$

4) $\sin^2 x \cdot \sqrt{(81 - x^2)} = 5e^{-x^2}$

5) $\frac{(x^2 - 9)}{(x^2 + 4)} = \sqrt{x^2 + 1}e^{x\cos x}$

6) $\ln^2(x-1) = 3\cos(2x) + 1$

7) $\sin(x)\sqrt{(81-x^2)} = 5arctg(x)$

8) $\frac{10}{(1+x^2)} - 2\cos 2x + x = 0$

9) $\frac{(10x-2)}{(3+x^2)} - 2\cos 2\ x \cdot \sqrt[4]{x} = 0$

10) $\frac{10}{(1-x^2)} = 2\sin(2x\quad) + x$

11) $\sqrt{(25-x^2)} = arctg\ (2x)$

12) $(x^3 - 2x^2 - 3x + 4) = 2\cos(2x+1)$

13) $arctg\,2x - \frac{(x-1)^4}{(5)} + \sin^2(5x) = 0$

14) $\sin^2 x \cdot \sqrt{(81 - x^2)} = 5e^{-x^2}$

15) $\frac{(x^2 - 9)}{(x^2 + 4)} = \sqrt{x^2 + 1}e^{x\cos x}$

16) $\ln^2(x-1) = 3\cos(2x) + 1$

17) $\left| \sin x \cdot \sqrt{(81 - x^2)} - 5arctgx \right| = 0$

18) $\frac{10}{(1 + x^2)} - 2\cos 2x + x = 0$

19) $\frac{(10x - 2)}{(3 + x^2)} - 2\cos 2\ x \cdot \sqrt[4]{x} = 0$

20) $\frac{10}{(1 - x^2)} = 2\sin(2x\quad) + x$

21) $\sqrt{(25 - x^2)} = arctg\ (2x)$

22) $\sin^2 x \cdot \sqrt{(81 - x^2)} = 5e^{-x^2}$

23) $(x^3 - 2x^2 - 3x + 4) = 2\cos(2x + 1)$

24) $\cos(x)^2 \sqrt{(81 - x^2)} = 5arctg(x)$

25) $4\cos(x)^2 = \sqrt{(25 - x^2)}$

****

Приклад рішення

Графічно дослідити рішення нелінійного рівняння та отримати всі (якщо можливо) рішення.

Зберегти малюнки графіків на окремий файл.

Рішення уточнити шляхом половинного поділу з точністю eps=10-8. Для цього скласти програму.

$$4\cos(x)^2 = \sqrt{(25 - x^2)}$$

Рішення

Для рівняння Ф1(x) = Ф2(x) значення x буде розв'язанням у точці перетину графіків функцій
y1(x)=Ф1(x) та y2(x)=Ф2(x)

Для нашого випадку Ф1(x) – це графік функцій

$$y1(x) = 4\cos(x)^2$$

Ф2(x) – це графік функцій

$$y2a(x) = +\sqrt{(25 - x^2)}$$

$$y2b(x) = -\sqrt{(25 - x^2)}$$

Т.к. підкорене вираз $\sqrt{(25 - x^2)}$ має бути більше нуля, то

$25 - x^2 \geq 0$ звідки слідує $x \in [-5,+5]$

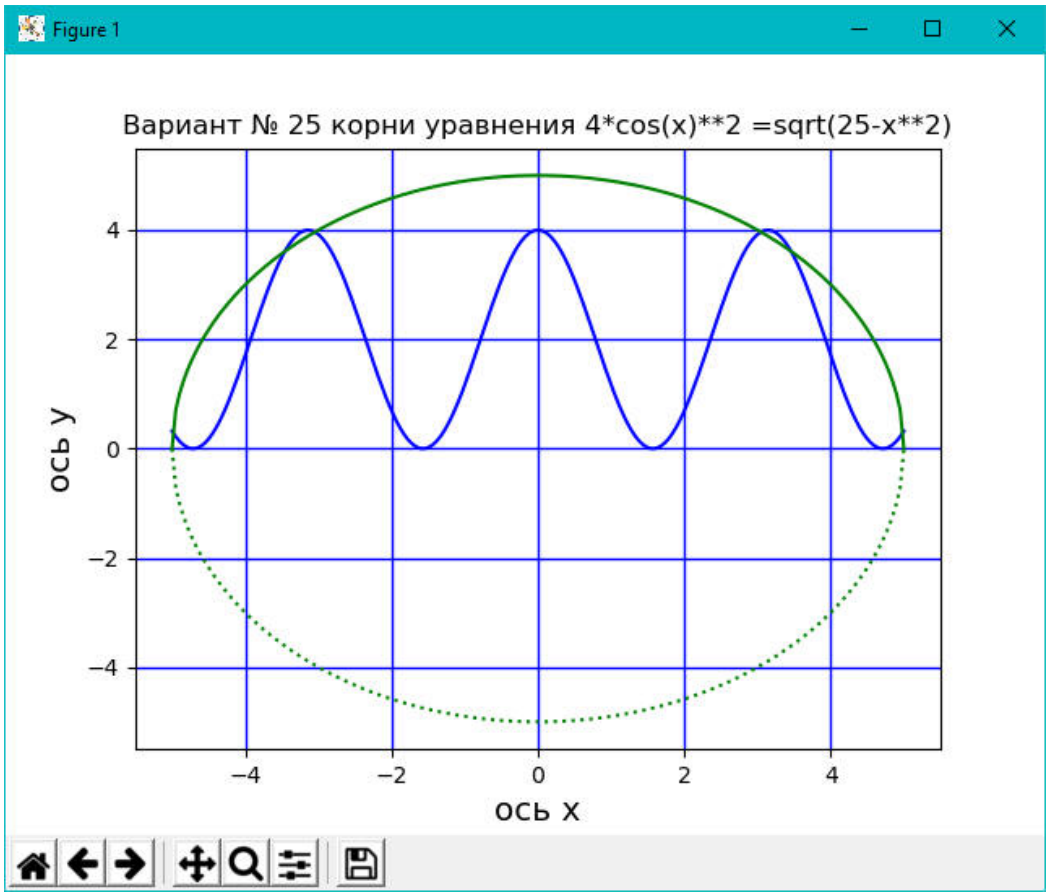
Запишемо програму

(виводити легенду не будемо)

```
import matplotlib.pyplot as plt
import numpy as np
# независимая переменная
x = np.linspace(-5, +5, 200) # 200 точек
# название осей
xlab='ось x'
ylab='ось y'
# первый график
y1= 4*np.cos(x)**2
# легенда
leg1='cos(x)**2'
# второй график - a
y2a= +np.sqrt(25-x**2)
leg2a='+sqrt(25-x**2)'
# второй график - b
y2b= -np.sqrt(25-x**2)
leg2b='-sqrt(25-x**2)'
# Включаем сетку по оси X и оси Y.
# Задаем цвет толщину сетки
plt.grid(color = 'b',linewidth = 1)
# Задаем подписи к осям X и оси Y и размер шрифта
```

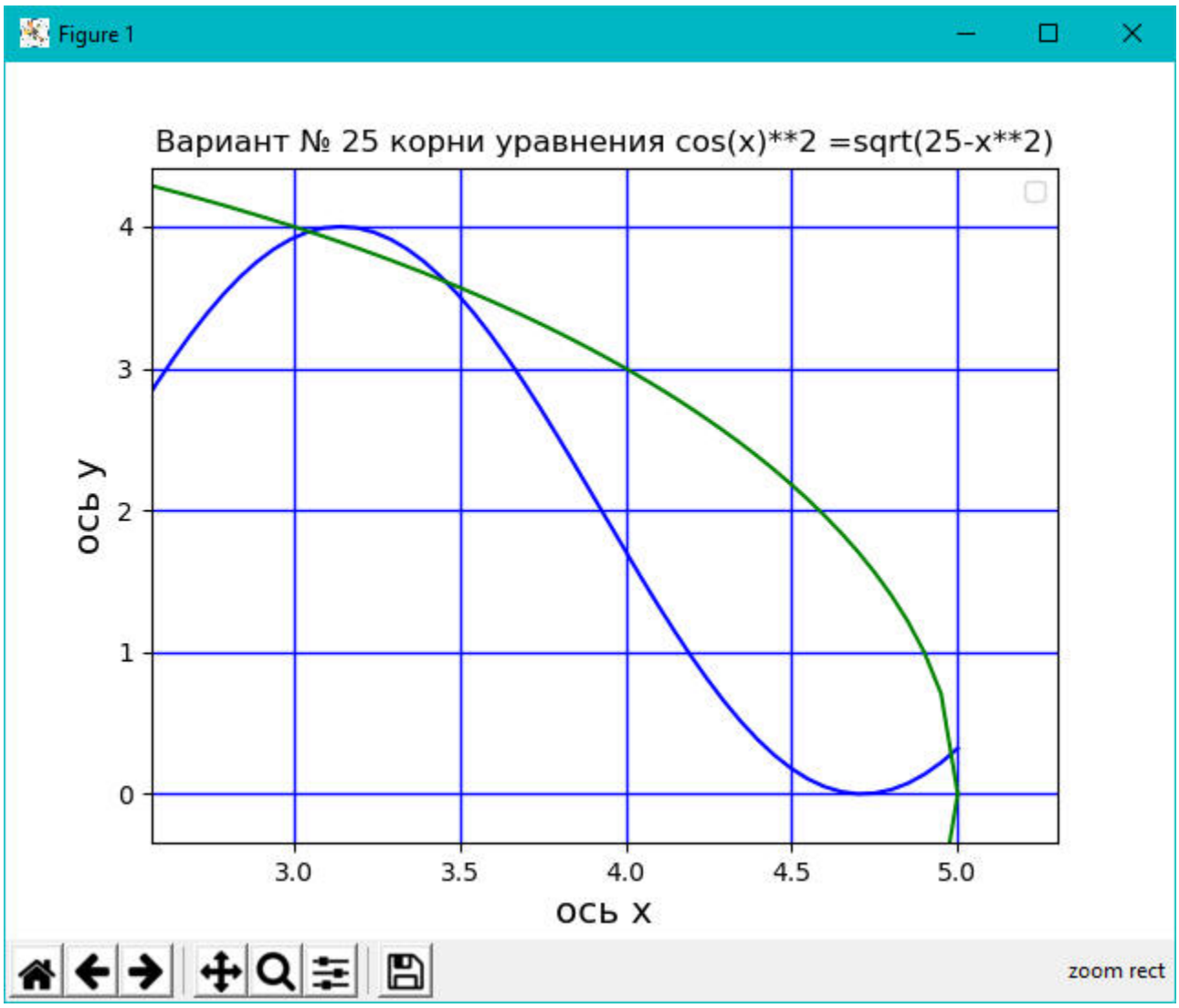
```
plt.xlabel(xlab, fontsize = 'x-large')
plt.ylabel(ylab, fontsize = 'x-large')
# Задаем заголовок диаграммы
plt.title('корни уравнения cos(x)**2 =sqrt(25-x**2)')
# строим графики
plt.plot(x,y1,'b-') # ,label=leg1)
plt.plot(x,y2a,'g-') #,label=leg2a)
plt.plot(x,y2b,'g-') #,label=leg2b)
# задаем вывод легенды и ее расположение
# plt.legend(loc='best')
# Включаем сетку
plt.grid(True)
# Сохраняем построенную диаграмму в файл
# Задаем имя файла и его тип
plt.savefig('var25.png', format = 'png')
# визуализируем графики
plt.show()
```

Результат роботи програми



Бачимо ШІСТЬ точок перетину графіків

використовуючи ZOOM



знаходимо

$x1 \in [3.0,\ \ 3.25]$

$x2 \in [3.25,\ \ 3.5]$

$x3 \in [4.75,\ \ 5.0]$

Значення x4, x5 і x6 з симетрії дорівнюють -x1, -x2 і -x3

Складемо програму уточнення коренів методом половинного поділу

```
import math
def f(x):
   return 4*math.cos(x)**2 - math.sqrt(25-x**2)

############
def pd(a, b, eps=1e-8):
   if f(a)*f(b) >0:
      print("error f(a)*f(b) > 0")
```

```
        return -1
    if a >= b:
        print("error a < b")
        return -1
    while True:
        x=a+(b-a)/2
        fx=f(x)
        #print(a,b,x,f(a), fx, f(b)); input()
        if abs(fx) < eps:
            return x
        if f(a)*fx < 0:
            b=x
        else:
            a=x
        pass

##############
x1=pd(3.0, 3.25)
x2=pd(3.25, 3.5)
x3=pd(4.75, 5.0)
print('x1=', x1)
print('x2=', x2)
print('x3=', x3)
x4=-x1; x5=-x2; x6=-x3;
print('x4=', x4)
print('x5=', x5)
print('x6=', x6)
print(10*'-')
for x in [x1,x2,x3,x4,x5,x6]:
print('f(',x,')=',f(x) )



================================ RESTART:
D:/Test/Python/lab3_pd.py ==============================
x1= 3.04699943959713
x2= 3.4589415714144707
x3= 4.990856625139713
x4= -3.04699943959713
x5= -3.4589415714144707
x6= -4.990856625139713
----------
f( 3.04699943959713 )= -4.768277772626561e-09
f( 3.4589415714144707 )= -5.146310044779057e-10
f( 4.990856625139713 )= 8.167858944752027e-09
f( -3.04699943959713 )= -4.768277772626561e-09
f( -3.4589415714144707 )= -5.146310044779057e-10
```

```
f( -4.990856625139713 )= 8.167858944752027e-09
>>>
```