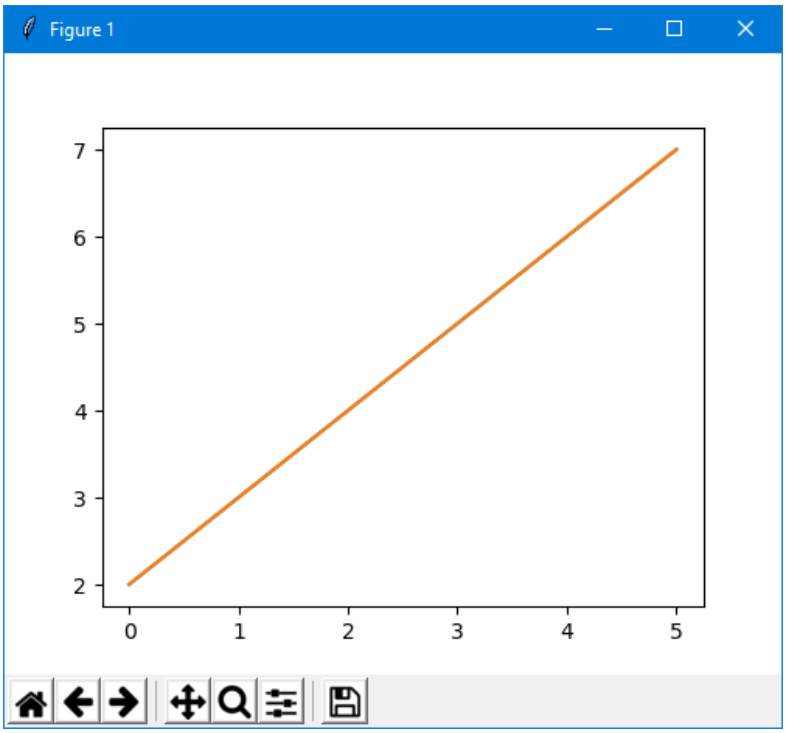
Тема 4. Побудова графіків функцій

П1. Список координат, x координати числа 0, 1, 2, 3...

Список y[2,3,4,5,6,7] координат, x координати утворюють послідовність 0, 1, 2, ...:

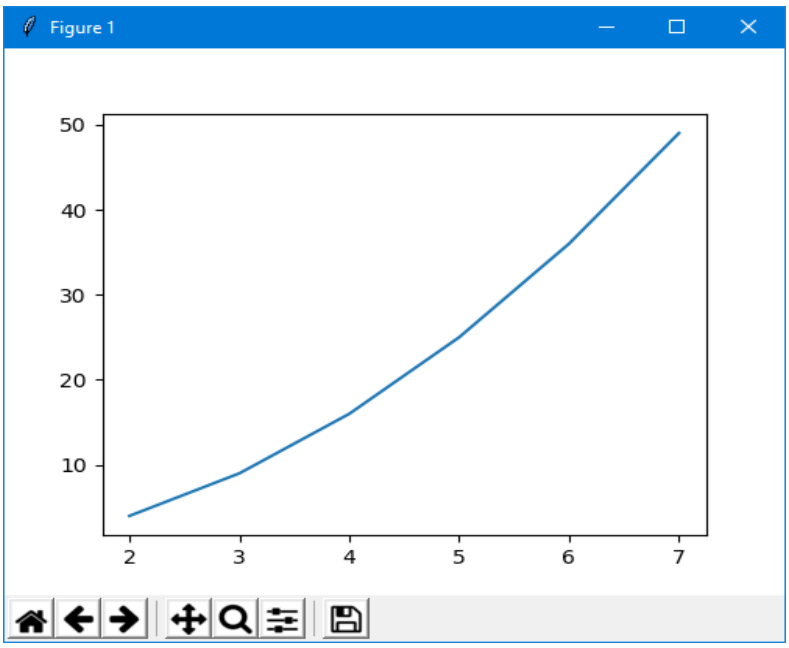
```
plt.plot([2,3,4,5,6,7]);plt.show()
```



П2. Списоки x та y-координат

Список x та y координат - [2,3,4,5,6,7], [4,9,16,25,36,49]);

```
plt.plot([2,3,4,5,6,7],[4,9,16,25,36,49]);plt.show()
```

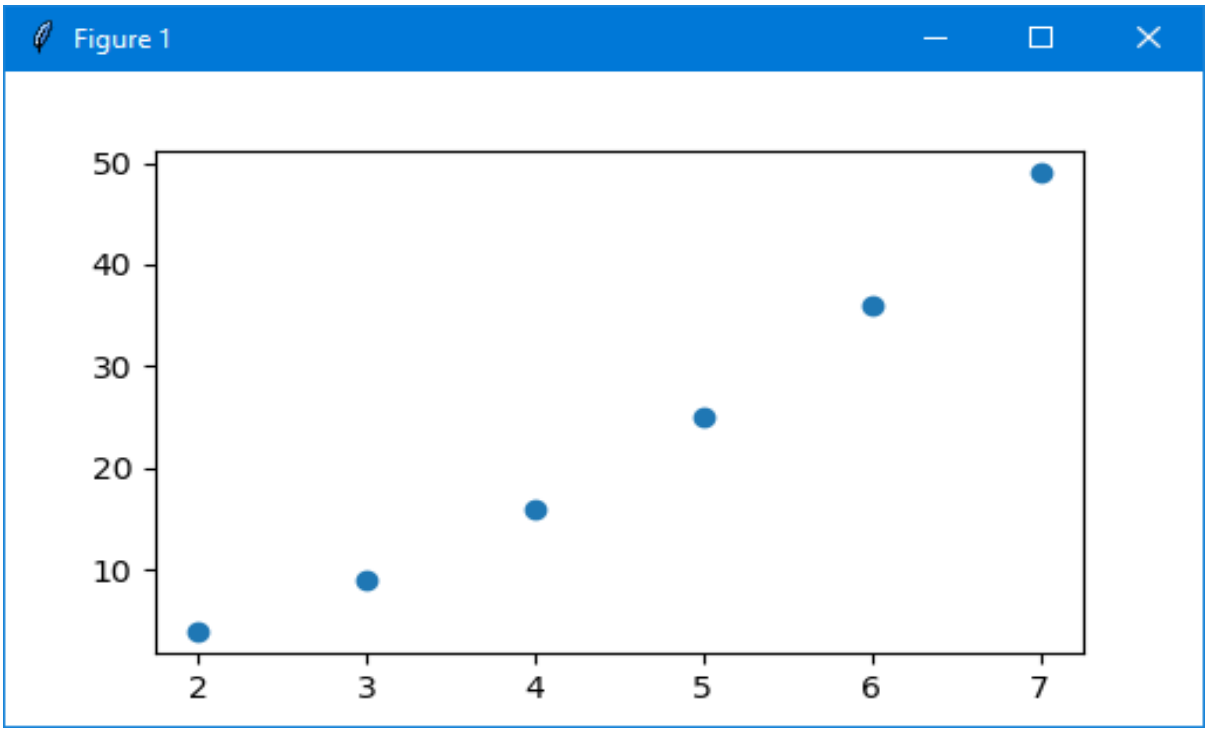


П3. Зображаємо крапки

Список x та y координат - [2,3,4,5,6,7], [4,9,16,25,36,49]);

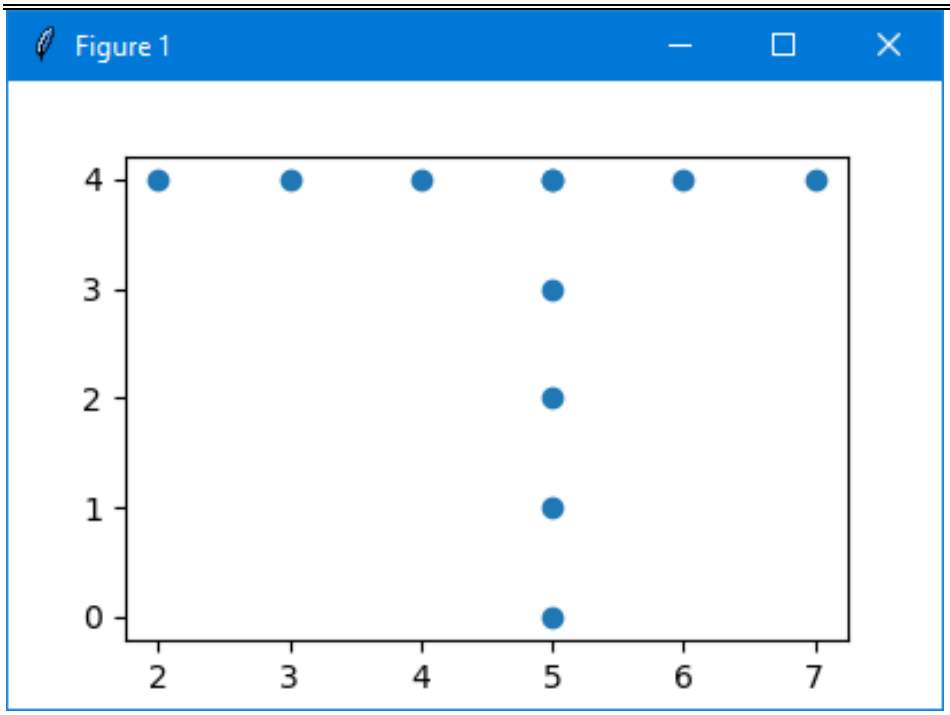
Зображаємо лише крапки

```
plt.scatter([2,3,4,5,6,7],[4,9,16,25,36,49]);plt.show()
```



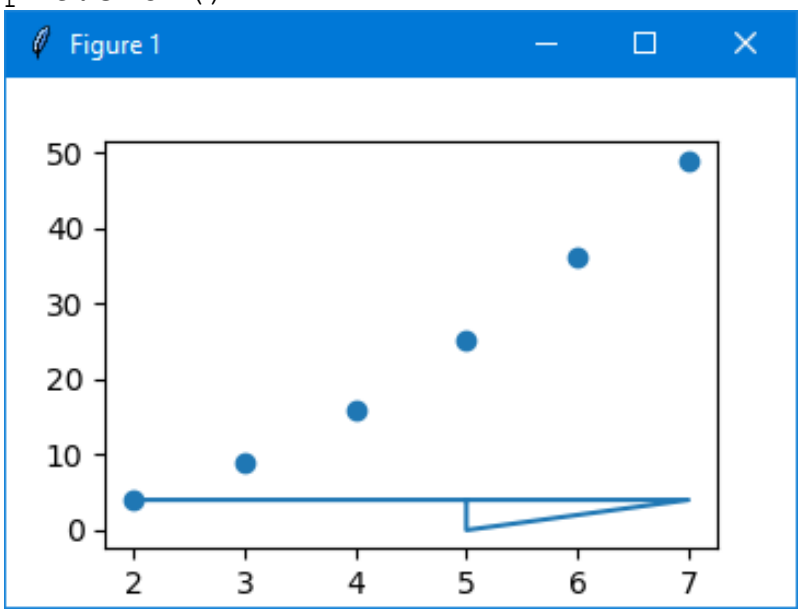
П4. «Довільні» координати

plt.scatter([2,3,4,5,6,7,5,5,5,5,5],[4,4,4,4,4,4,0,1,2,3,4,]);plt.show()



П5. Декілька графіків на одному аркуші

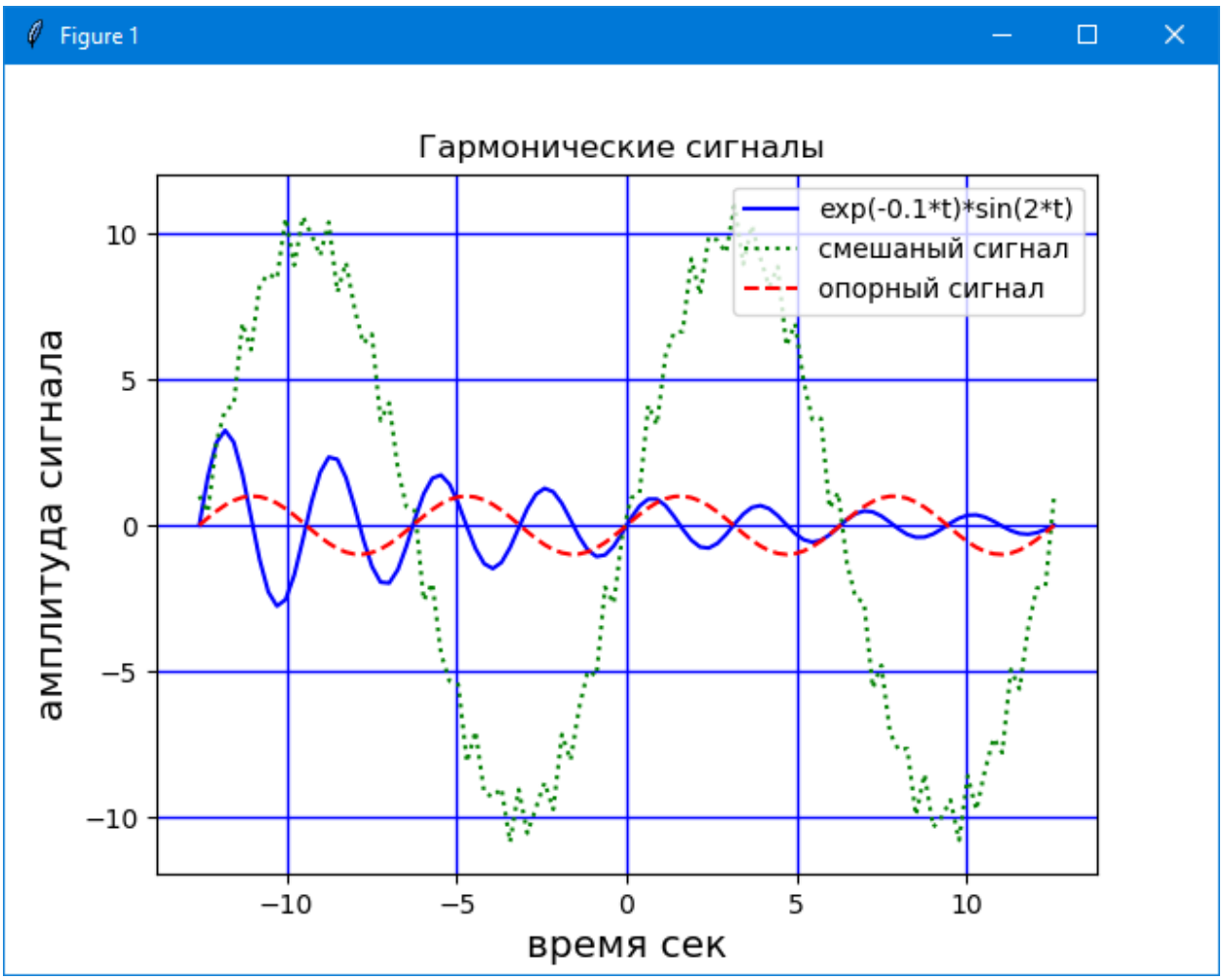
```
plt.scatter([2,3,4,5,6,7],[4,9,16,25,36,49])
plt.plot([2,3,4,5,6,7,5,5,5,5,5],
[4,4,4,4,4,4,0,1,2,3,4,])

plt.show()
```



П5. «Прикрашання» та багато графіків на діаграмі

Малюємо та зберігаємо у файлі три графіки на одній діаграмі в одному масштабі. Дивіться коментарі скрипту.

```
importmatplotlib.pyplot as plt
importnumpy as np
# незалежна змінна
x = np.linspace (-4 * np.pi, 4 * np.pi, 100)
назва осей
xlab='час сек'
ylab='амплітуда сигналу'
перший графік
y1= np.sin(2*x)*np.exp(-0.1*x)
# Легенда
leg1='exp(-0.1*t)*sin(2*t)'
# другий графік
y2=10*np.sin(0.5*x)+np.cos(10*x)
leg2='змішані сигнал'
# третій графік
y3=np.sin(x)
leg3='опорний сигнал'
# Включаємо сітку по осі X та осі Y.
# Задаємо колір товщину сітки
plt.grid(color = 'b',linewidth = 1)
# Задаємо підписи до осей X та осі Y та розмір шрифту
plt.xlabel(xlab, fontsize = 'x-large')
plt.ylabel(ylab, fontsize = 'x-large')
# Задаємо заголовок діаграми
plt.title('Гармонічні сигнали')
# будуємо графіки
plt.plot(x,y1,'b-',label=leg1)
plt.plot(x,y2,'g:',label=leg2)
plt.plot(x,y3,'r--',label=leg3)

# задаємо висновок легенди та її розташування
plt.legend(loc='best')
# Включаємо сітку
plt.grid(True)
# Зберігаємо побудовану діаграму у файл
# Задаємо ім'я файлу та його тип
plt.savefig('signal3.png', format = 'png')
візуалізуємо графіки
plt.show()
```



Як задаються стилі ліній, маркери та кольори ліній дивіться у документації або скористайтеся «демо скриптом»:

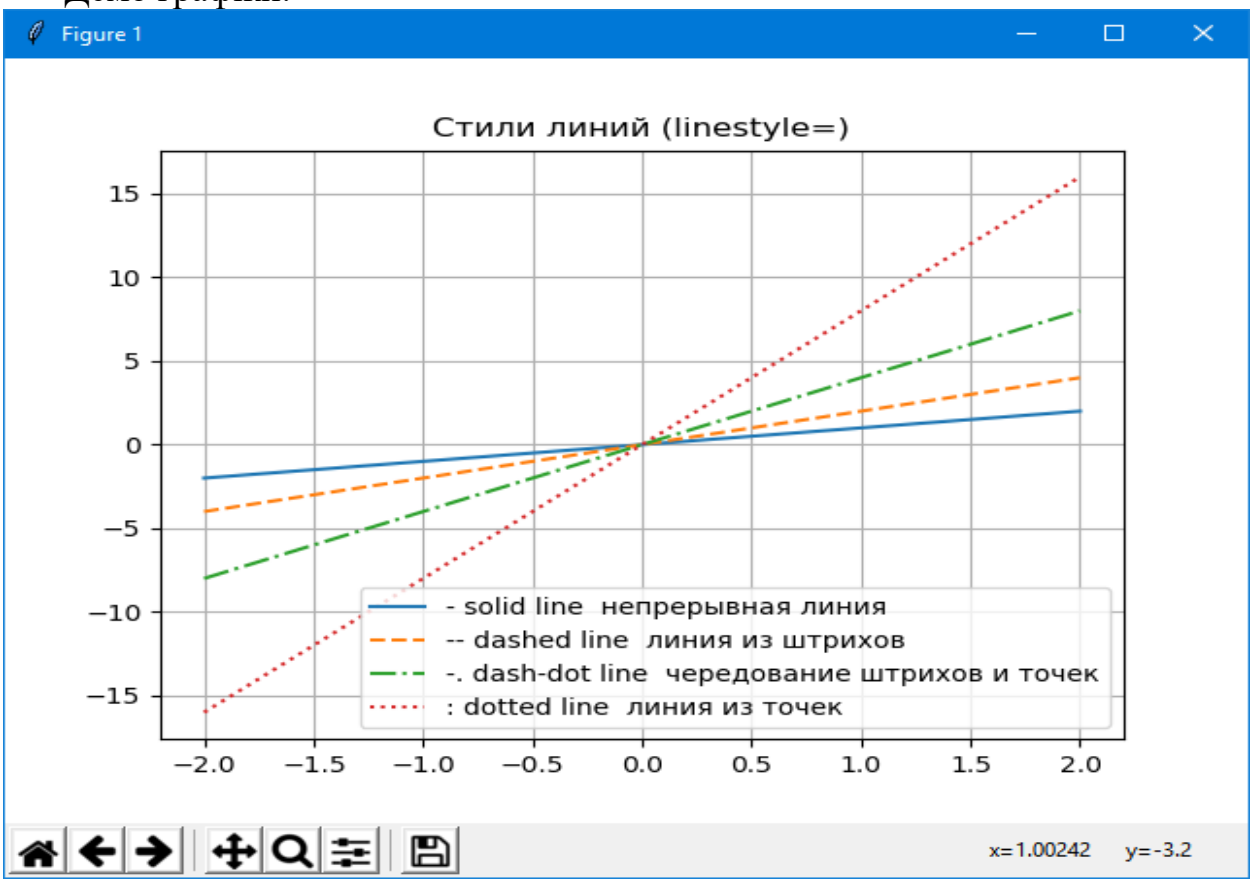
```
importsys
importmath
importos
importmatplotlib.pyplot as plt
importnumpy as np
##sys.exit(0)
importmatplotlib as mpl
# Виведення на екран поточної версії бібліотеки
matplotlib
print('Current version on matplotlib library is',
mpl.__version__)

x = np.linspace(-2, 2, 10)
y1=1*x
y2=2*x
y3=4*x
y4=8*x

s1= ['-', 'solid line', 'безперервна лінія']
```

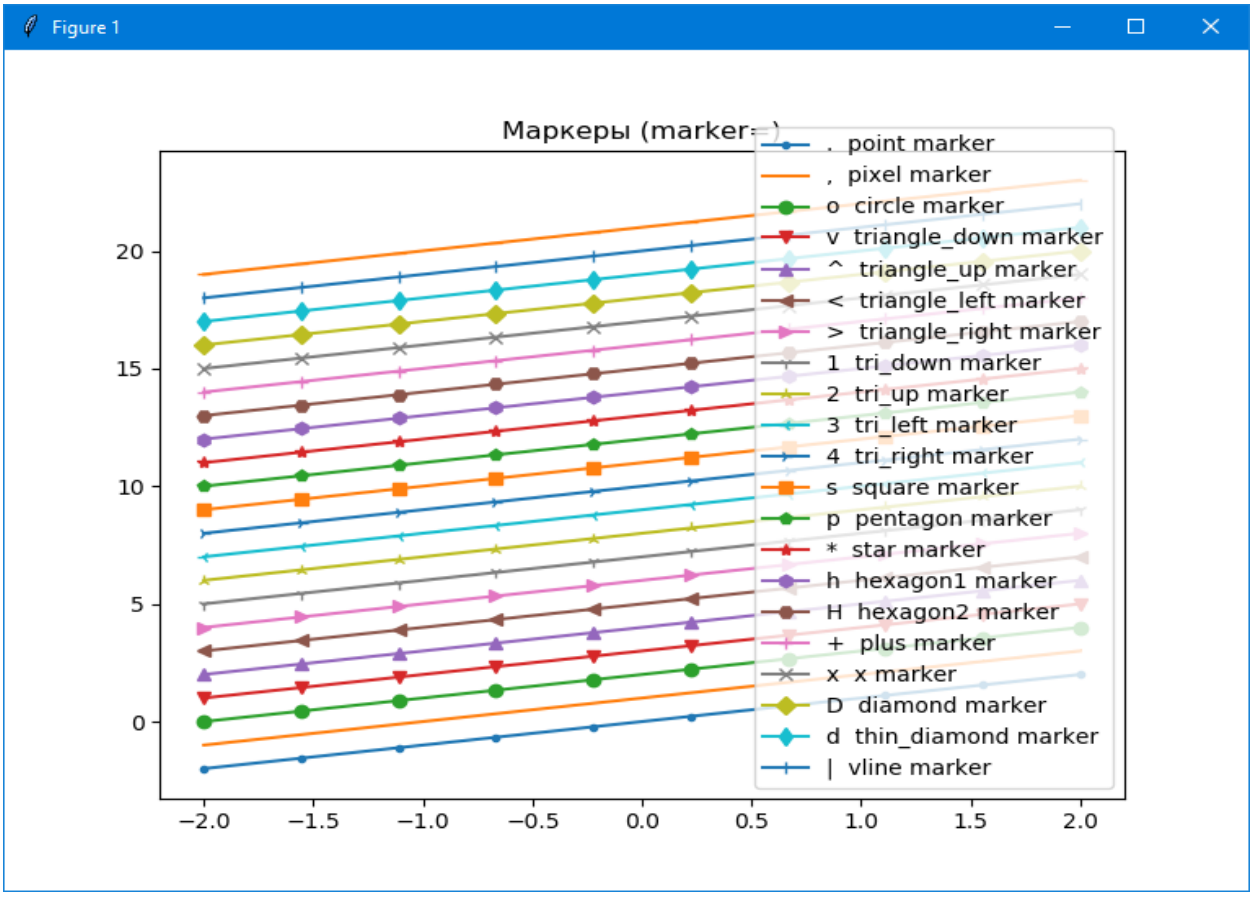
```
s2= ['--', 'dashed line ', 'лінія зі штрихів']
s3= ['-.', 'dash-dot line ', 'чергування штрихів і
точок']
s4= [':', 'dotted line', 'лінія з точок']

leg1=s1[0]+' '+s1[1]+' '+s1[2]
leg2=s2[0]+' '+s2[1]+' '+s2[2]
leg3=s3[0]+' '+s3[1]+' '+s3[2]
leg4=s4[0]+' '+s4[1]+' '+s4[2]

ax=plt.subplot(111)
# можна поміняти розміри вікна графіка у граф. вікні
box=ax.get_position()
ax.set_position([box.x0, box.y0, box.width*1.0 ,
box.height*1.0])


p1 = plt.plot (x, y1, linestyle = s1 [0], label = leg1)
p2=plt.plot(x,y2,linestyle=s2[0], label=leg2)
p3 = plt.plot (x, y3, linestyle = s3 [0], label = leg3)
p4 = plt.plot (x, y4, linestyle = s4 [0], label = leg4)

plt.title('Стилі ліній (linestyle=)')
#ax.legend(loc=(1.0,0.5), mode='expand' )
#ax.legend(mode='expand', bbox_to_anchor=(1, 0.05),loc
# ='upper center' ) #left best
ax.legend(loc='lower right')
# Включаємо сітку
plt.grid()
plt.show()

#sys.exit(0)
# marker
plt.close()
mar=[
'.', 'point marker',
',', 'pixel marker',
'o', 'circle marker',
'v', 'triangle_down marker',
'^', 'triangle_up marker',
'<', 'triangle_left marker',
'>', 'triangle_right marker',
'1', 'tri_down marker',
'2', 'tri_up marker',
'3', 'tri_left marker',
```

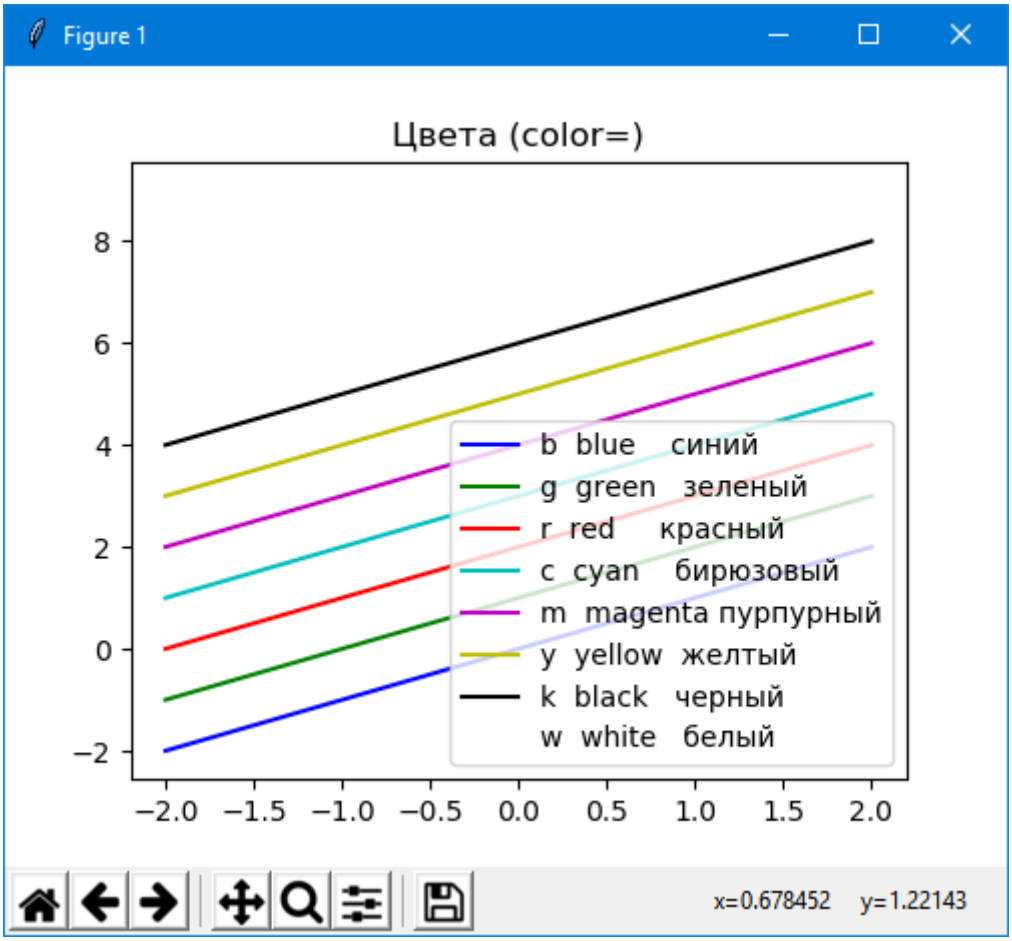
```
'4', 'tri_right marker',
's', 'square marker',
'p', 'pentagon marker',
'*', 'star marker',
'h', 'hexagon1 marker',
'H', 'hexagon2 marker',
'+', 'plus marker',
'x', 'x marker',
'D', 'diamond marker',
'd', 'thin_diamond marker',
'|', 'vline marker',
'_', 'hline marker'
]
leg=[]
forі in range(0,len(mar),2):
 leg.append(mar[i]+' '+mar[i+1])
mpl.rcParams['figure.figsize'] = (8.0, 6.0)
kk = int (len (mar) / 2)
forі in range(kk):
    plt.plot(x, x+i, marker=leg[i][0], label=leg[i])
plt.legend(loc='lower right')
plt.title('Маркери (marker=)')
plt.show()

#color
col=[
'b', 'blue синій',
'g', 'green зелений',
'r', 'red червоний',
'c', 'cyan бірюзовий',
'm', 'magenta пурпуровий',
'y', 'yellow жовтий',
'k', 'black чорний',
'w', 'white білий'
]

leg=[]
forі in range(0,len(col),2):
 leg.append(col[i]+' '+col[i+1])
mpl.rcParams['figure.figsize'] = (5.0, 4.0)
kk = int (len (col) / 2)
forі in range(kk):
    plt.plot(x, x+i, color=leg[i][0], label=leg[i])
plt.legend(loc='lower right')
plt.title('Кольори (color=)')
```

```
plt.show()
```

Демо графіки:







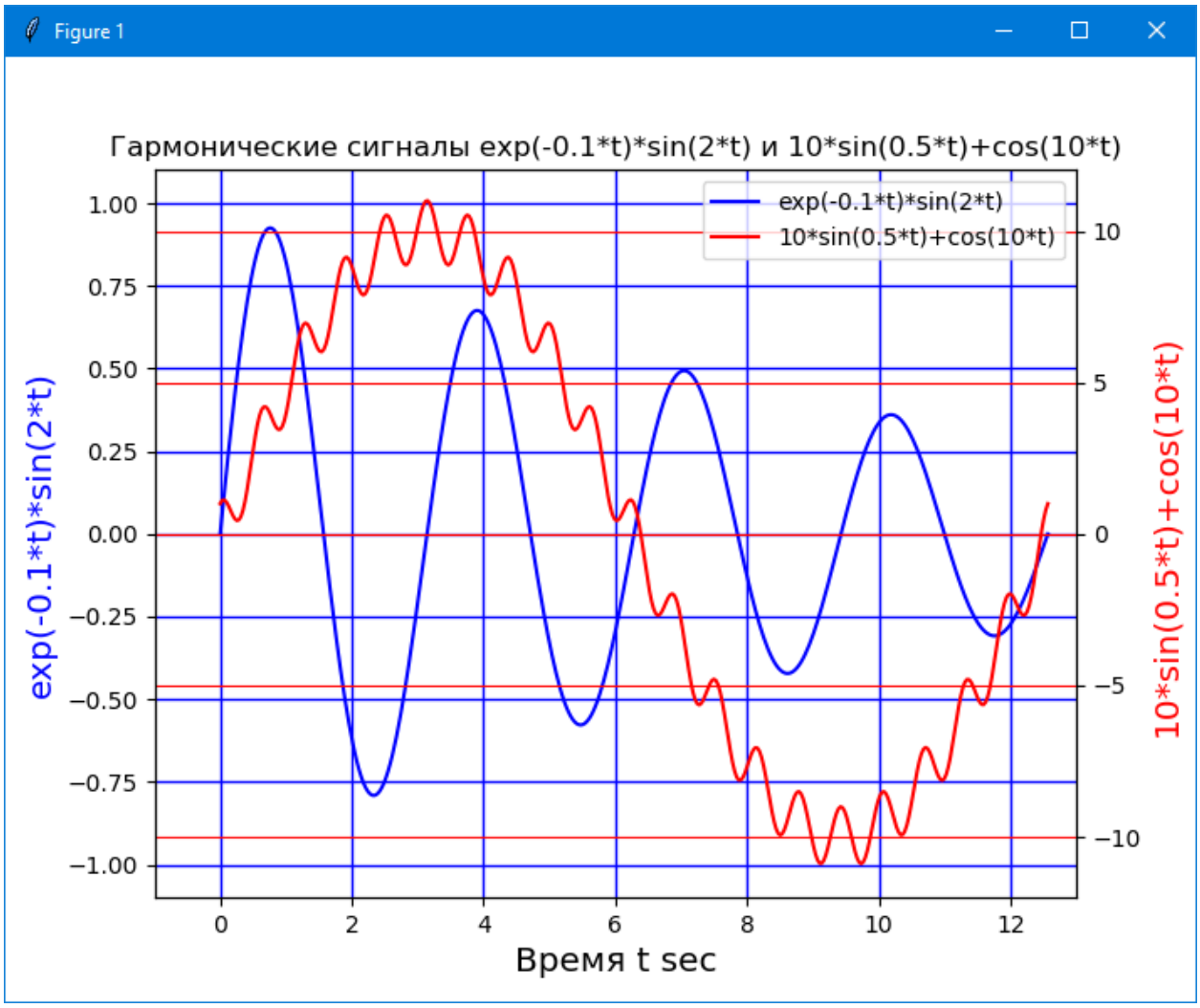
П6. Два графіки на діаграмі в індивідуальних масштабах

Зверніть увагу на «кому» після імені змінної
line_01,= ax_01.plot(X, Y_01, 'b-')

```
# -*- coding: UTF-8 -*-
importnumpy as np
importmatplotlib as mpl
importmatplotlib.pyplot as plt
# Значення по осі X час, t
X = np.linspace (0, 4 * np.pi, 400)
# Значення по осях Y_01 & Y_02
leg_01='exp(-0.1*t)*sin(2*t)'
Y_01 = np.sin(2*X)*np.exp(-0.1*X)
# "Модульований" сигнал
leg_02="10*sin(0.5*t)+cos(10*t)"
Y_02=10*np.sin(0.5*X)+np.cos(10*X)
# Задамо розміри зображення діаграми
mpl.rcParams['figure.figsize'] = (8.0, 6.0)
# Будуємо діаграму
```

```
# Отримуємо посилання на об'єкт типу
matplotlib.axes.AxesSubplot, поточну діаграму
ax_01 = plt.axes()
# Задаємо вихідні дані для першої лінії діаграми #
(лінії ступеня розкладання), зовнішній вигляд лінії та
# маркера.
Функція plot() повертає посилання на list,
перший елемент, якого є об'єкт класу #
matplotlib.lines.Line2D
line_01 = ax_01.plot(X, Y_01, 'b-') #, label=leg_01)
# якщо потрібно надалі використовувати об'єкт класу #
matplotlib.lines.Line2D

# то застосовують ДВА варіанти
# 1) line_01 = ax_01.plot(X, Y_01, 'b-') # прийом
першого елемента списку

# 2) line_01 (без коми) = … використовувати line_01 [0]
# Задаємо інтервали значень по осях X та # основної осі
Y
ax_01.axis([-1, 13, -1.1, 1.1])
# Включаємо сітку по осі X та основної осі Y. # Задаємо
колір сітки
ax_01.grid(color = 'b',linewidth = 1)
# Задаємо підписи до осей X та основної осі Y # та
розмір фонту
ax_01.set_xlabel('Час t sec', fontsize = 'x-large')
ax_01.set_ylabel(leg_01, color = 'b',
                                fontsize = 'x-large')
# Задаємо заголовок діаграми
ax_01.set_title('Гармонічні сигнали '+ leg_01+" та "+
leg_02)
#ax_01.legend( loc='upper right')
# Включаємо додаткову вісь Y
ax_02 = ax_01.twinx()
# Задаємо вихідні дані для другої лінії діаграми,
зовнішній вигляд лінії і маркера.
# Функція plot() повертає посилання на об'єкт класу #
matplotlib.lines.Line2D (див. вище)
line_02, = ax_02.plot(X, Y_02, 'r-') #, label=leg_02)
# Задаємо інтервали значень по осях X# та додаткової
осі Y
ax_02.axis([-1, 13, -12, 12])
# Задаємо підпис до додаткової осі Y
```

```
ax_02.set_ylabel(leg_02,color='r', fontsize = 'x-
large')

# Задаємо вихідні дані для легенди та місце її
розміщення
ax_02.legend((line_01, line_02), (leg_01, leg_02), loc
= 'best')
#ax_02.legend(loc='lower left')
# Включаємо сітку додаткової осі Y. # Задаємо колір
сітки
ax_02.grid(color = 'r')
# Зберігаємо побудовану діаграму у файл
# Задаємо ім'я файлу та його тип
#plt.savefig('trigo.png', format = 'png')
візуалізуємо
plt.show()
```



П7 Кілька графіків в одному та в різних графічних вікнах

Робота з matplotlib заснована на використанні графічних вікон та осей (осі дозволяють задати деяку графічну область). Усі побудови застосовуються до поточних осей. Це дозволяє зображувати декілька графіків в одному графічному вікні.

За замовчуванням створюється одне графічне вікно figure(1) та одна графічна область subplot(111) у цьому вікні.

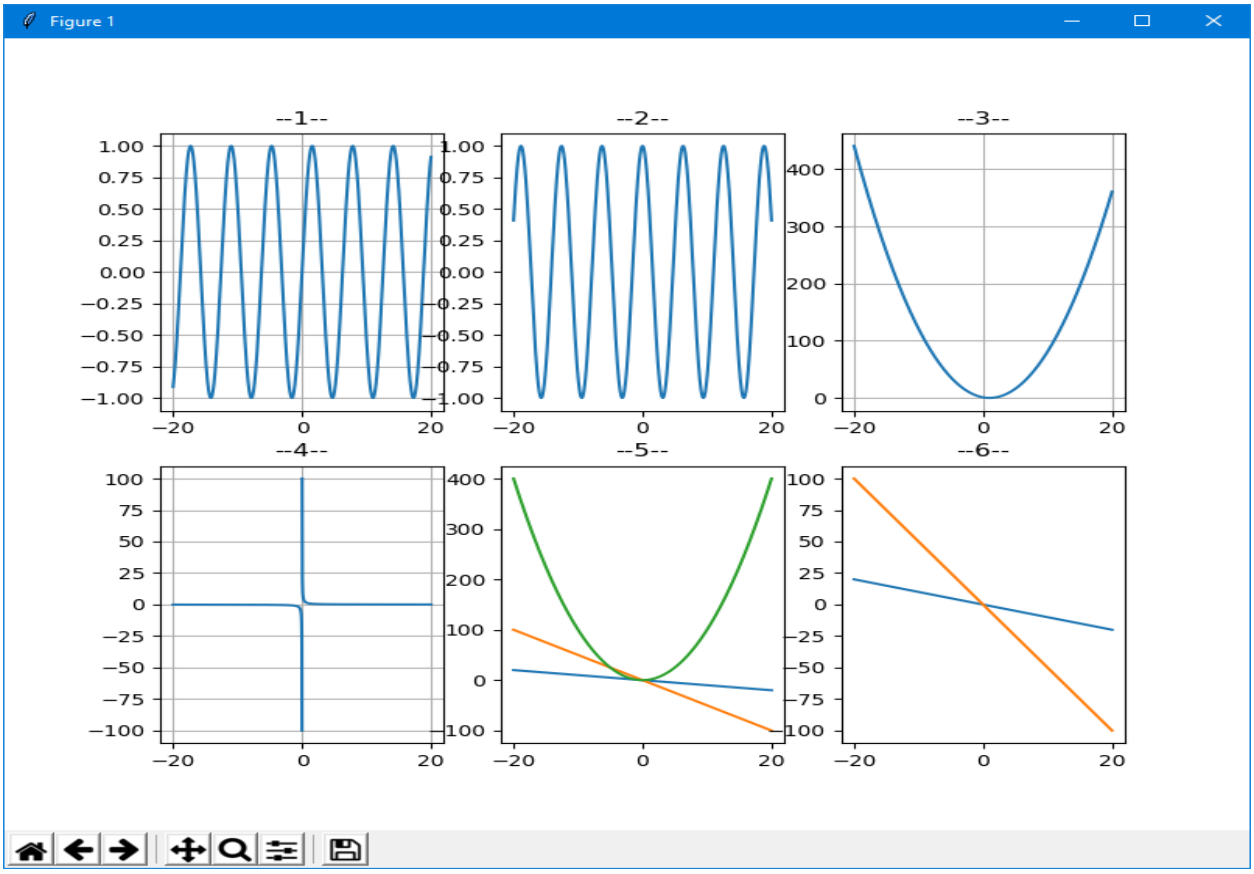
Команда subplot дозволяє розбити графічне вікно на кілька областей. Вона має три параметри: nr; nc; np:

- параметри nr та nc визначають кількість рядків та стовпців на які розбивається графічна область
- параметр np визначає номер поточної області (np набуває значення від 1 до nr*nc).

Якщо nr*nc<10, передавати параметри nr,nc,np можна без використання коми. Наприклад, допустимі форми subplot(2,2,1) та subplot(221).

```
importmath
importnumpy as np
importmatplotlib.pyplot as plt

# Імпортуємо пакет із допоміжними функціями
importmatplotlib.mlab as ml

# Малюватимемо графік цієї функції
deffunc (x):
    """
    sinc(x)
    """
    if x == 0:
        return1.0
    returnmath.sin(x)/x

# Інтервал зміни змінної по осі X
xmin = -20.0
xmax = 20.0
# Крок між точками
dx = 0.01
# Створимо список координат по осі X на відрізку [-
xmin; xmax], включаючи кінці
xlist = ml. frange (xmin, xmax, dx)
# Обчислимо значення функції у заданих точках
ylist = [func(x) for x in xlist]
```

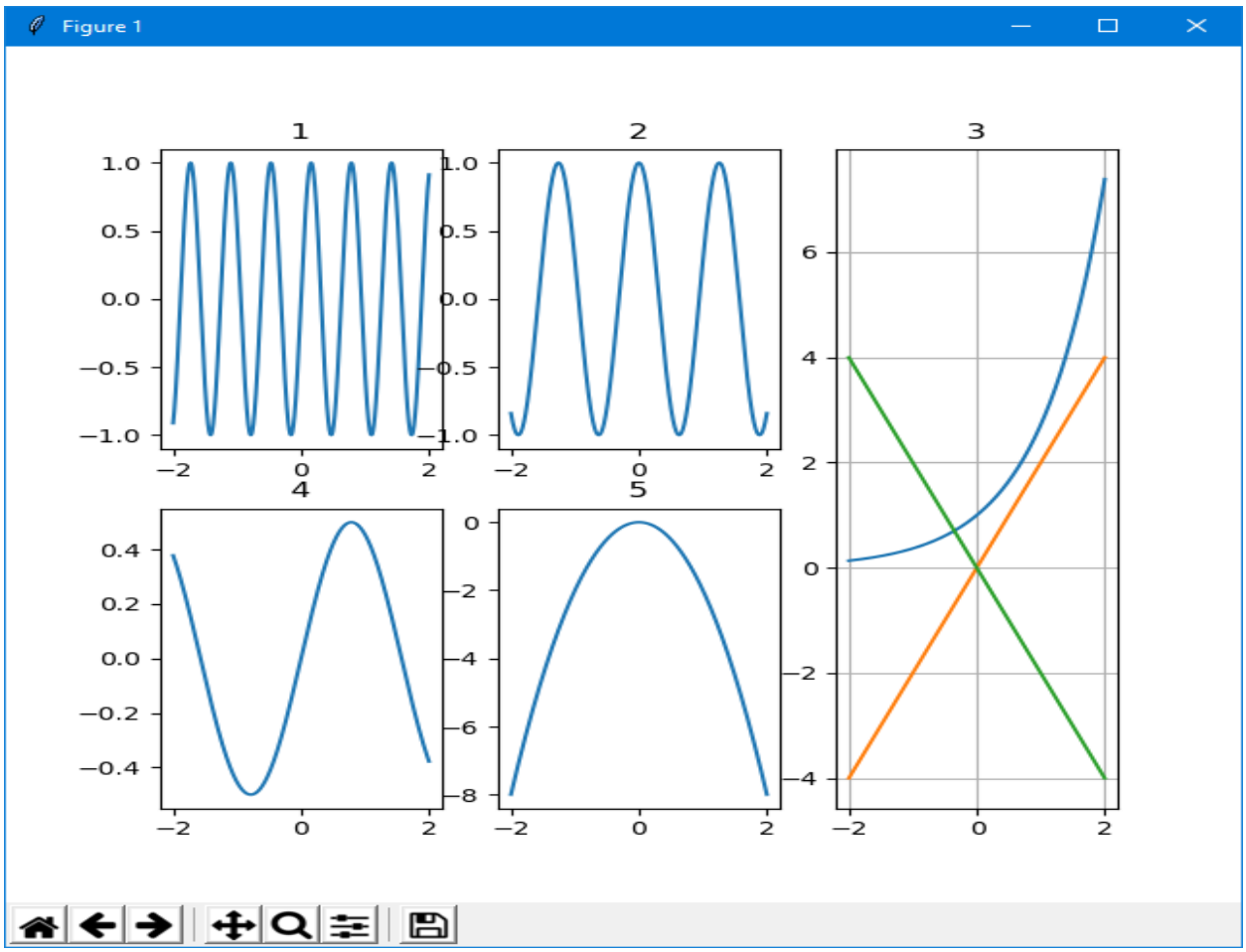
```
# !!! Два рядки, три стовпці.
# !!! Поточний осередок - 1
plt.subplot (2, 3, 1)
plt.plot (xlist, np.sin(xlist) ) #ylist)
plt.grid(True)
plt.title ("--1--")

# !!! Два рядки, три стовпці.
# !!! Поточний осередок - 2
plt.subplot (2, 3, 2)
plt.plot (xlist, np.cos(xlist))
plt.title ("--2--")

# !!! Два рядки, три стовпці.
# !!! Поточний осередок - 3
plt.subplot (2, 3, 3)
plt.plot (xlist, (xlist-1)*(xlist-1))
plt.grid(True)
plt.title ("--3--")

# !!! Два рядки, три стовпці.
# !!! Поточний осередок - 4
plt.subplot (2, 3, 4)
plt.plot (xlist, 1/xlist)
plt.grid(True)
plt.title ("--4--")

# !!! Два рядки, три стовпці.
# !!! Поточний осередок - 5
plt.subplot (2, 3, 5)
plt.plot (xlist, -xlist)
plt.plot (xlist, -5*xlist)
plt.plot (xlist, xlist * xlist)
plt.title ("--5--")

# !!! Два рядки, три стовпці.
# !!! Поточний осередок - 6
plt.subplot (2, 3, 6)
plt.plot (xlist, -xlist)
plt.plot (xlist, -5*xlist)
plt.title ("--6--")

# Покажемо вікно з намальованим графіком
plt.show()
```



Всі ці осередки суто умовні, тому для кожного виклику subplot() розбиття може бути своє, що дозволяє зробити, наприклад, наступне розташування графіків (зверніть увагу на нумерацію осередків):

```
importmath
importnumpy as np
importmatplotlib.pyplot as plt
# Імпортуємо пакет із допоміжними функціями
importmatplotlib.mlab as ml
# Інтервал зміни змінної по осі X
xmin = -2.0
xmax = 2.0
# Крок між точками
dx = 0.001
# Створимо список координат по осі X на відрізку [-
xmin; xmax], включаючи кінці
x = ml. frange (xmin, xmax, dx)
# !!! Два рядки, три стовпці.
# !!! Поточний осередок - 1
plt.subplot (2, 3, 1)
plt.plot (x, np.sin(10*x))
plt.title ("1")
# !!! Два рядки, три стовпці.
```

```
# !!! Поточний осередок - 2
plt.subplot (2, 3, 2)
plt.plot (x, np.cos(5*x) )
plt.title ("2")
# !!! Два рядки, три стовпці.
# !!! Поточний осередок - 4
plt.subplot (2, 3, 4)
plt.plot (x, np.sin(x)*np.cos(x))
plt.title ("4")
# !!! Два рядки, три стовпці.
# !!! Поточний осередок - 5
plt.subplot (2, 3, 5)
plt.plot (x, -2*x*x)
plt.title ("5")
# !!! Один рядок, три стовпці.
# !!! Поточний осередок - 3
plt.subplot (1, 3, 3)
plt.plot (x, np.exp(x))
plt.plot (x, 2*x)
plt.plot (x, -2*x)
plt.grid(True)
plt.title ("3")
# Покажемо вікно з намальованим графіком
plt.show()
```



figure() та axes()

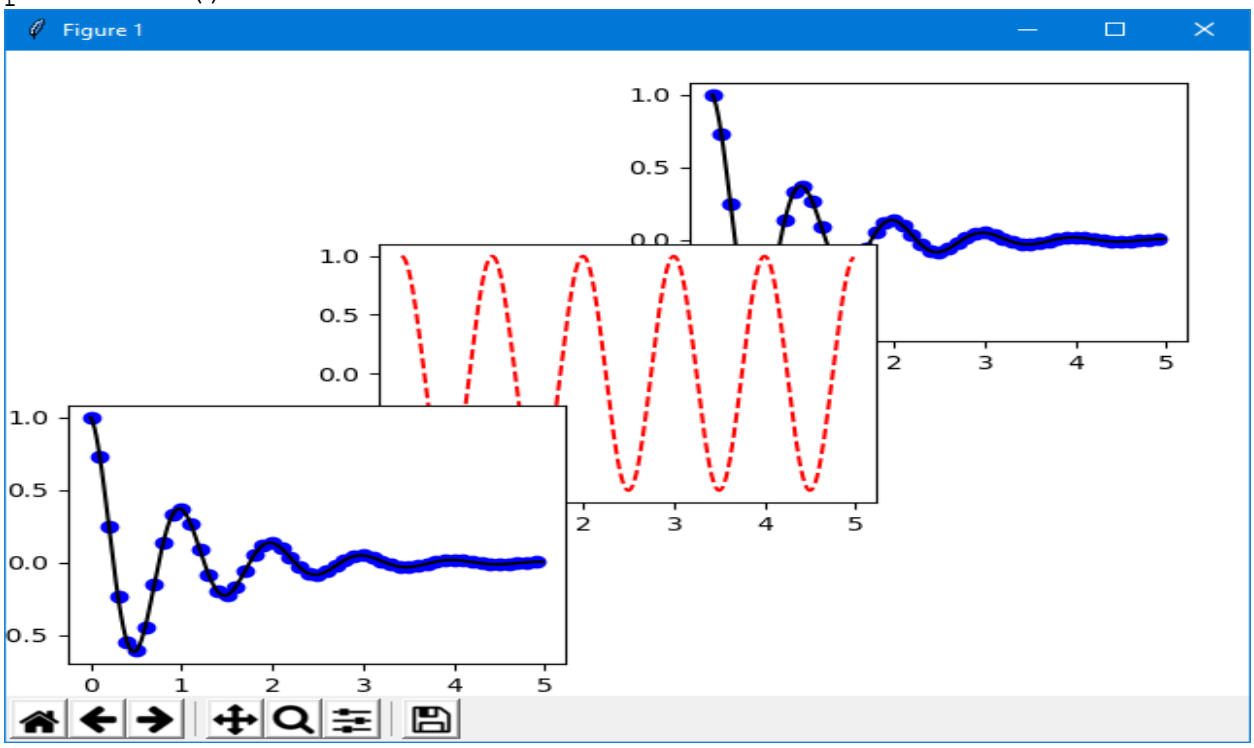
Є можливість розмістити осі (підвіконня) вручну, наприклад не на прямокутній сітці, а як-небудь ялинкою, використовуйте команду axes(), яка дозволяє визначити положення осей як

axes([Left, bottom, width, height]),

де всі значення змінюються від 0 до 1. Виглядати це може так:

```
importmath
importnumpy as np
importmatplotlib.pyplot as plt
# Імпортуємо пакет із допоміжними функціями
importmatplotlib.mlab as ml
importnumpy as np
importmatplotlib.pyplot as plt
def f(t):
    returnnp.exp(-t) * np.cos(2*np.pi*t)

t1 = np.arange(0.0, 5.0, 0.1)
t2 = np.arange(0.0, 5.0, 0.02)
plt.figure(1)
plt.axes([0.55, 0.55, 0.4, 0.4])
plt.plot(t1, f(t1), 'bo', t2, f(t2), 'k')
plt.axes([0.3, 0.3, 0.4, 0.4])
```

```
plt.plot(t2, np.cos(2*np.pi*t2), 'r--')
plt.axes([0.05, 0.05, 0.4, 0.4])
plt.plot(t1, f(t1), 'bo', t2, f(t2), 'k')
# Покажемо вікно з намальованим графіком
plt.show()
```

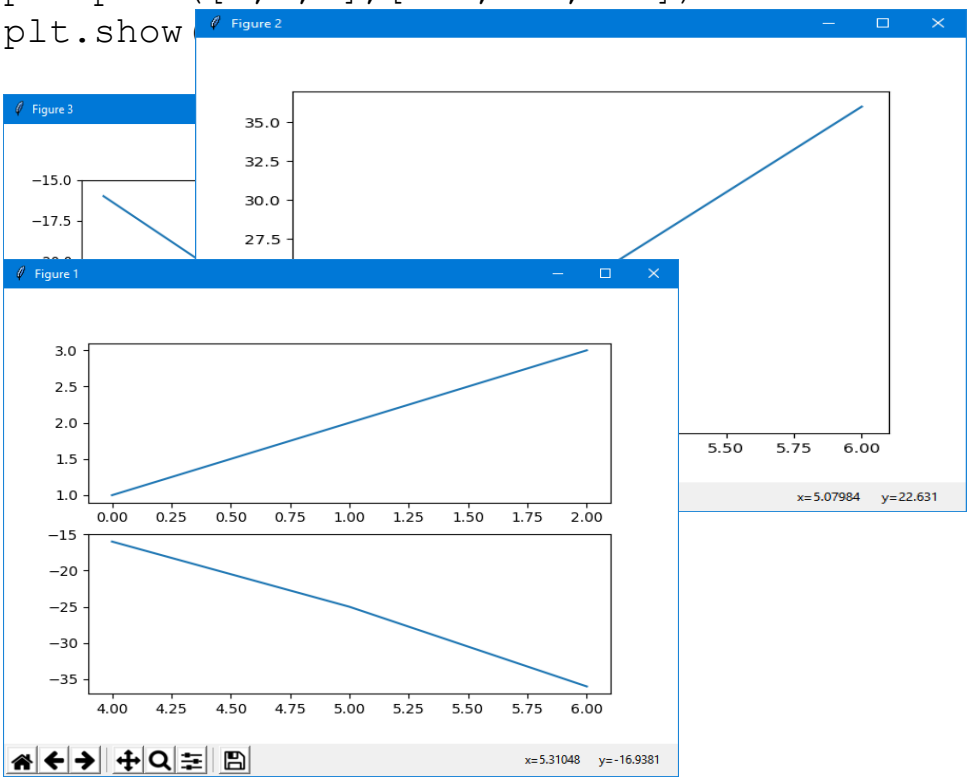


Є можливість створити кілька вікон, викликаючи figure() з номером вікна, що збільшується. Звичайно, кожне вікно може містити кілька осей та підвікон.

Очистити активне вікно можна clf() , а активні осі за допомогою cla().

```
importmath
importnumpy as np
importmatplotlib.pyplot as plt
# Імпортуємо пакет із допоміжними функціями
importmatplotlib.mlab as ml
importnumpy as np
importmatplotlib.pyplot as plt

plt.figure(1) # the first figure
plt.subplot(211) # 1
plt.plot([1,2,3])
plt.subplot(212) # second subplot in first figure
plt.plot([4,5,6],[-16,-25,-36])
plt.figure(2) # a second figure
plt.plot([4,5,6],[16,25,36]) # creates a subplot(111)
by default
plt.figure(3)
```

```
plt.plot([4,5,6],[-16,-25,-36])
plt.show
```



Встановити свій діапазон осей

```
plt.xlim(right=xmax) #xmax is your value
plt.xlim(left=xmin) #xmin is your value
plt.ylim(top=ymax) #ymax is your value
plt.ylim(bottom=ymin) #ymin is your value
```

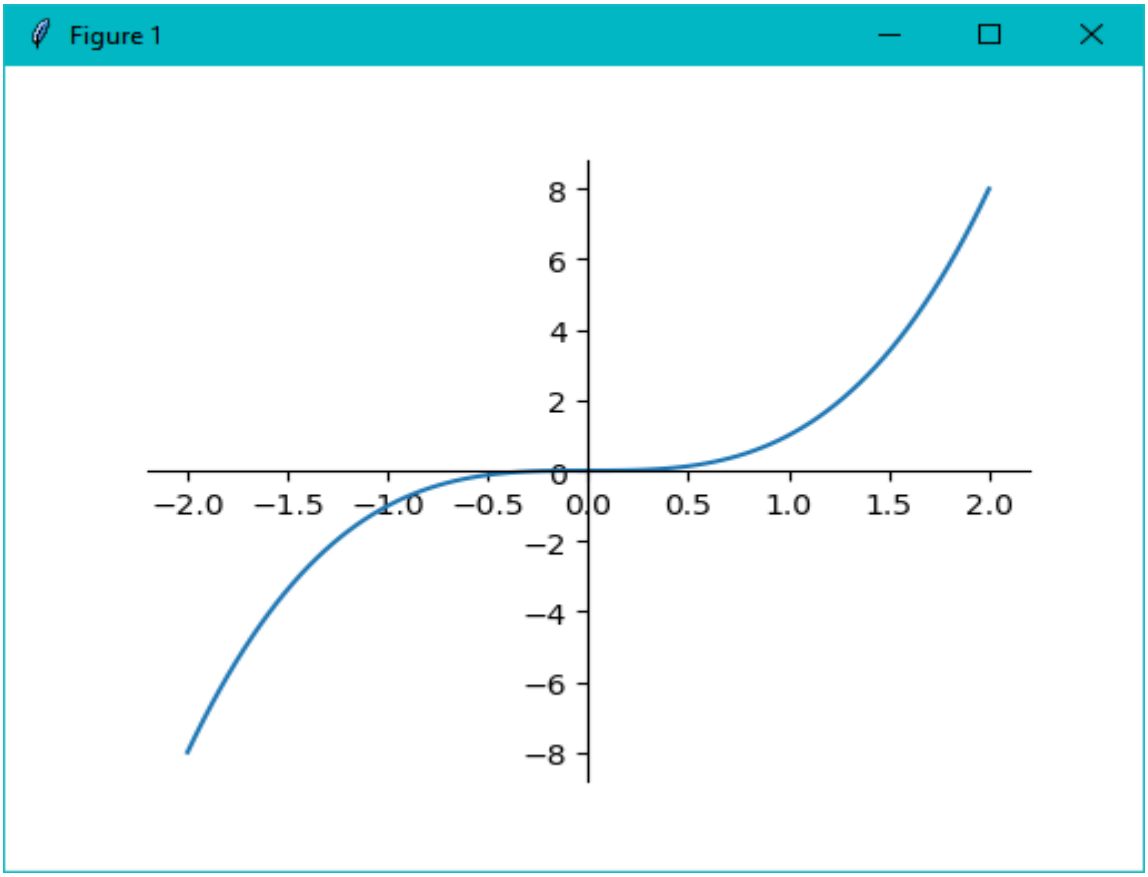
Або так:

```
plt.ylim(ymin, ymax)
plt.xlim(xmin, xmax)
```

Перенесення координатних осей до центру графіка

Якщо необхідно перенести координатні осі в центр малюнка, можна вчинити так:

```
importmatplotlib.pyplot as plt
importnumpy as np
X = np.linspace(-2,2,100)
Y = X**3
plt.plot(X, Y)
ax = plt.gca()
ax.spines['left'].set_position('center')
ax.spines['bottom'].set_position('center')
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
plt.show()
```

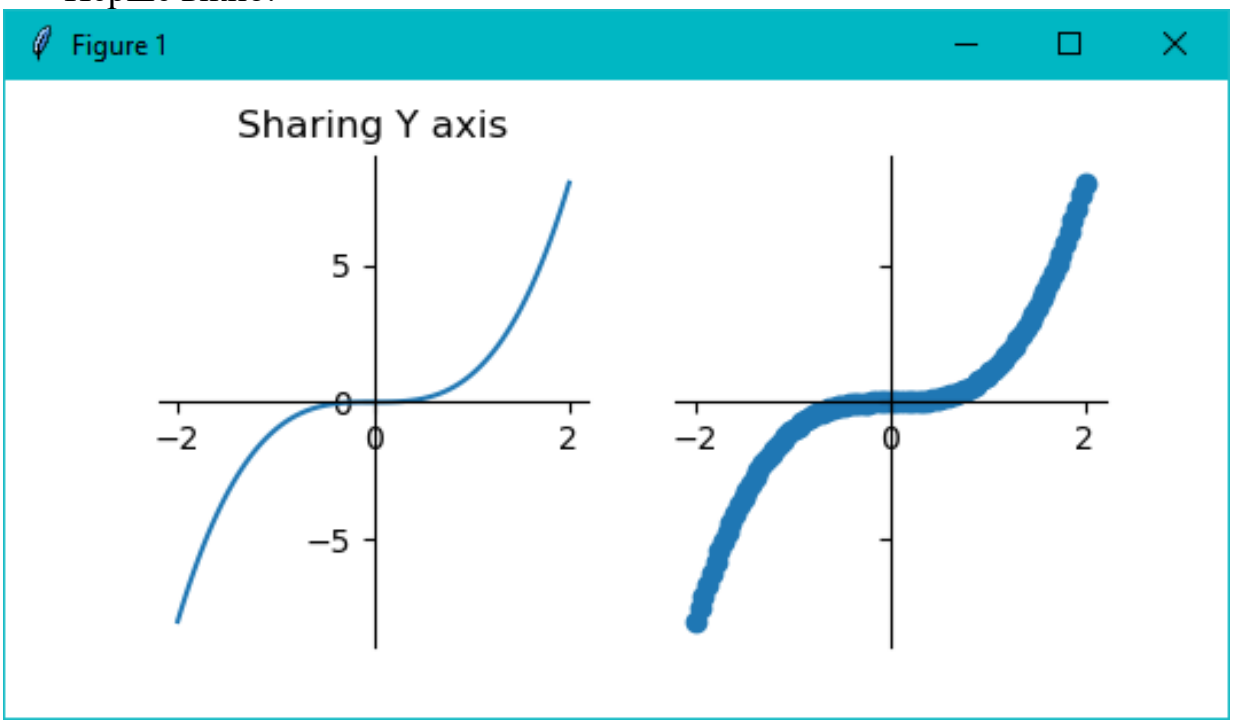


Приклад перенесення осей для subplot:

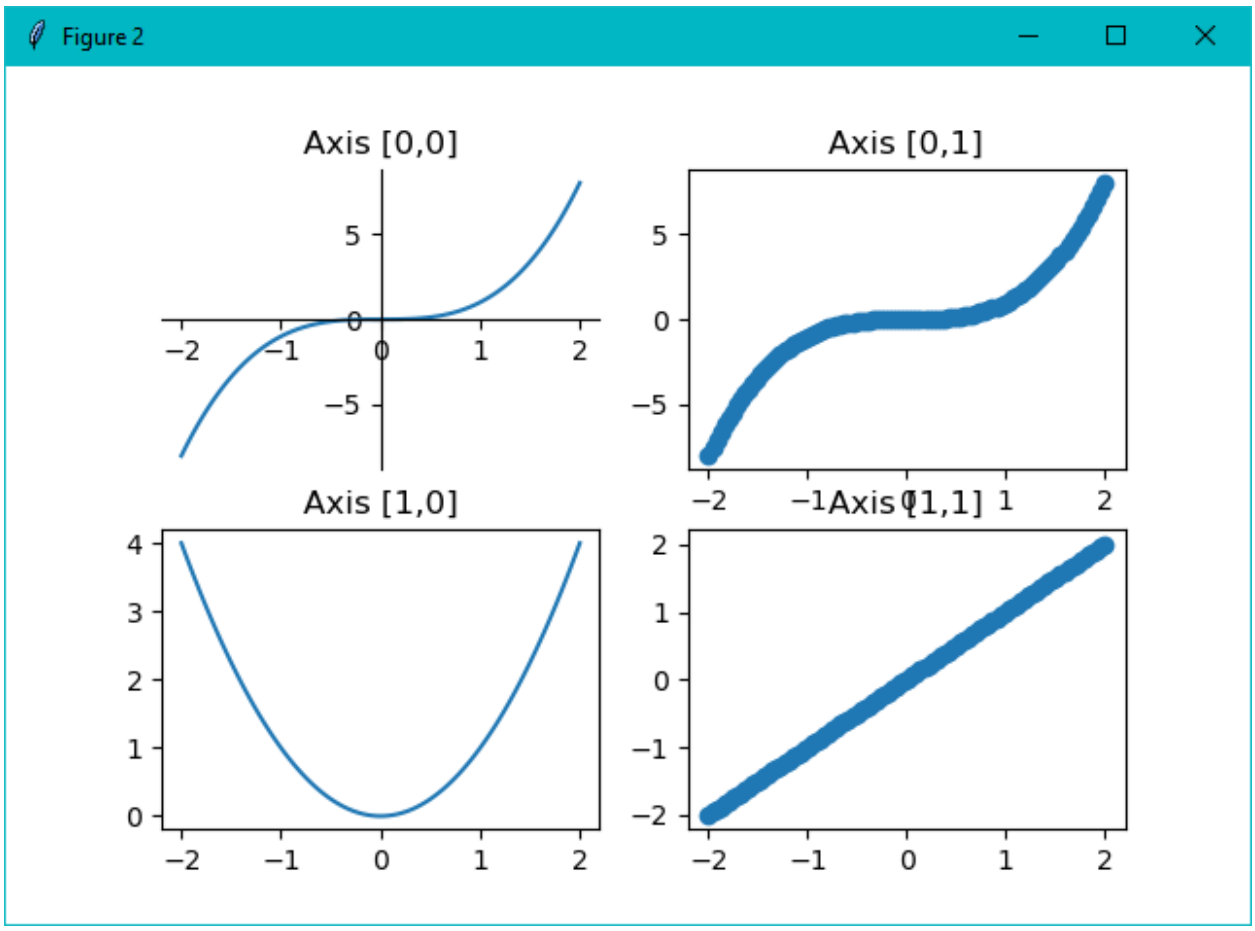
```
#Приклад отримання осей для subplots:
# Two subplots, unpack the axes array immediately
f, (ax1, ax2) = plt.subplots(1, 2, sharey=True)
ax1.plot(X, Y)
ax1.set_title('Sharing Y axis')
ax1.spines['left'].set_position('center')
ax1.spines['bottom'].set_position('center')
ax1.spines['top'].set_visible(False)
ax1.spines['right'].set_visible(False)

ax2.scatter(X, Y)
ax2.spines['left'].set_position('center')
ax2.spines['bottom'].set_position('center')
ax2.spines['top'].set_visible(False)
ax2.spines['right'].set_visible(False)

f, axarr = plt.subplots(2, 2) # Four axes, returned as
a 2-d array


axarr[0, 0].plot(X, Y)
axarr[0, 0].spines['left'].set_position('center')
```

```
axarr[0, 0].spines['bottom'].set_position('center')
axarr[0, 0].spines['top'].set_visible(False)
axarr[0, 0].spines['right'].set_visible(False)
axarr[0, 0].set_title('Axis [0,0]')

axarr[0, 1].scatter(X, Y)
axarr[0, 1].set_title('Axis [0,1]')
axarr[1, 0].plot(X, X ** 2)
axarr[1, 0].set_title('Axis [1,0]')
axarr[1, 1].scatter(X, X ** 1)
axarr[1, 1].set_title('Axis [1,1]')

plt.show()
```

Перше вікно:



Друге вікно:

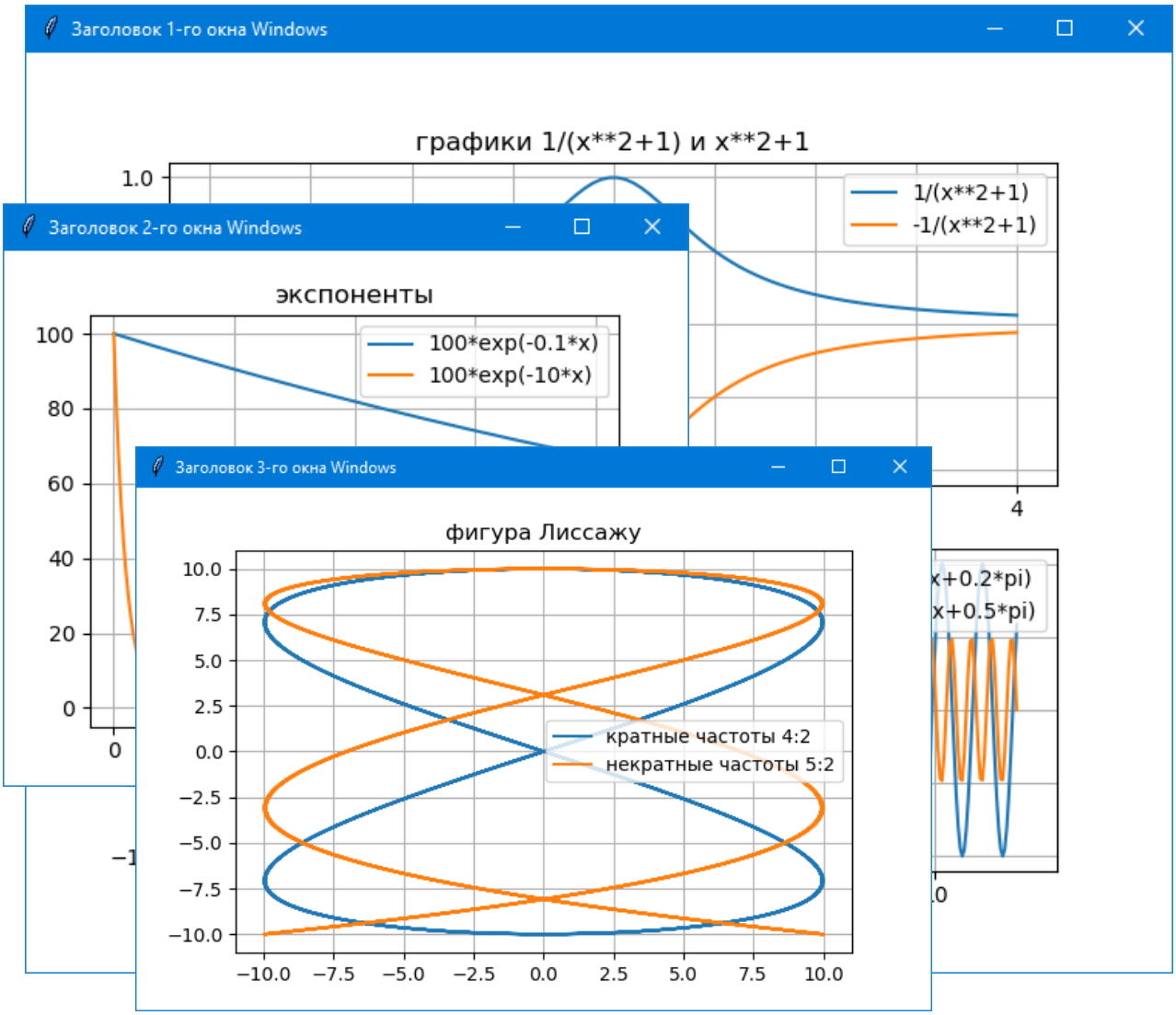


П8 Написи на вікнах діаграм і вікнах Windows, зміна розміру вікна

Наведемо приклад, аналогічний розглянутому вище. Для всіх вікон та діаграм задані титульні (тульбарні) написи:

```
importmath
importnumpy as np
importmatplotlib.pyplot as plt
# Імпортуємо пакет із допоміжними функціями
importmatplotlib as mpl
importnumpy as np
importmatplotlib.pyplot as plt

x1=np.linspace(-4,4,100)
y1=1/(x1**2+1)
y11=-1/(x1**2+1)
# Задамо розміри зображення діаграми
mpl.rcParams['figure.figsize'] = (8.0, 6.0)
plt.figure("Заголовок 1-го вікна Windows") # перше #
вікно windows

plt.subplot(211) # перші графіки у цьому вікні
```

```
plt.plot(x1,y1,label='1/(x**2+1)')
plt.plot(x1,y11,label='-1/(x**2+1)')
plt.title("графіки 1/(x**2+1) та x**2+1" )
plt.grid(True)
plt.legend()
plt.subplot(212) # другі графіки у цьому вікні
x2=np.linspace(-4*np.pi,4*np.pi,400)
plt.plot(x2,10*np.sin(5*x2+0.2*np.pi),label='10*sin(5*x
+0.2*pi)')
plt.plot(x2,5*np.cos(10*x2+0.5*np.pi),label="5*cos(10*x
+0.5*pi)")
plt.title("тригонометричні функції")
plt.grid(True)
plt.legend(loc='best')

plt.figure("Заголовок 2-го вікна Windows") # a second
figure
# або так
#fig = plt.figure()
#fig.canvas.set_window_title("Заголовок 2-го вікна
Windows")
x3=np.linspace(0,4,400)
plt.plot(x3, 100*np.exp(-0.1*x3), label='100*exp(-
0.1*x)')
plt.plot(x3, 100*np.exp(-10*x3), label='100*exp(-
10*x)')
plt.title("експоненти")
plt.grid(True)
plt.legend(loc='best')
fig3=plt.figure()
fig3.canvas.set_window_title("Заголовок 3-го вікна
Windows")
plt.grid(True)
t=np.linspace(0,8*np.pi,500)
plt.plot(10*np.sin(4*t),10*np.cos(2*t), label="кратні
частоти 4:2")

plt.plot(10*np.sin(5*t),10*np.cos(2*t),
                        label="неразові частоти 5:2")
plt.title("фігура Лісажу")
plt.grid(True)
plt.grid(True)
plt.legend(loc='best')

plt.show()
```


Заголовок 1-го окна Windows
графики 1/(x**2+1) и x**2+1
1/(x**2+1)
-1/(x**2+1)
Заголовок 2-го окна Windows
экспоненты
100*exp(-0.1*x)
100*exp(-10*x)
Заголовок 3-го окна Windows
фигура Лиссажу
кратные частоты 4:2
некратные частоты 5:2