Павшук Ярослав Денисович, ФІТ 3-14, варіант 21  
  
**Метод Ньютона**  
  
import numpy as np

from scipy.misc import derivative

def f(x):

return 3 \* x\*\*4 - 4 \* x\*\*3 + x\*\*2 - 2 \* x - 3

eps = 0.001 # точність

def newton(a, b, eps):

xi = (a + b) / 2 # початкове припущення, можна вибрати інше початкове значення за необхідності

while True:

df = derivative(f, xi, dx=1e-6, n=1) # обчислення першої похідної

d2f = derivative(f, xi, dx=1e-6, n=2) # обчислення другої похідної

if d2f == 0:

print("Друга похідна дорівнює нулю. Метод Ньютона не працює.")

return

xi\_1 = xi - df / d2f # ітерація методом Ньютона

if abs(xi\_1 - xi) < eps: # перевірка на збіжність

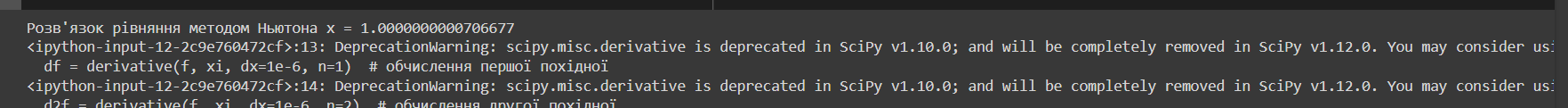
break

xi = xi\_1 # оновлення xi для наступної ітерації

print("Розв'язок рівняння методом Ньютона x =", xi\_1)

a = -2.0

b = -0.5

newton(a, b, eps)  
  
  
**Комбінований метод**  
  
import numpy as np

from scipy.misc import derivative

def f(x):

return 6 \* x\*\*4 + 4 \* x\*\*3 - x\*\*2 - x - 10

def komb(a, b, eps):

fa = f(a)

fb = f(b)

if fa \* derivative(f, a, n=2) > 0:

a0, b0 = a, b

else:

a0, b0 = b, a

ai = a0

bi = b0

while abs(ai - bi) > eps:

ai\_1 = ai - f(ai) \* (bi - ai) / (f(bi) - f(ai))

bi\_1 = bi - f(bi) / derivative(f, bi, n=1)

ai = ai\_1

bi = bi\_1

x = (ai\_1 + bi\_1) / 2

return print('Комбінований метод =', x)

komb(-2, -1/2, 0.0001)  
