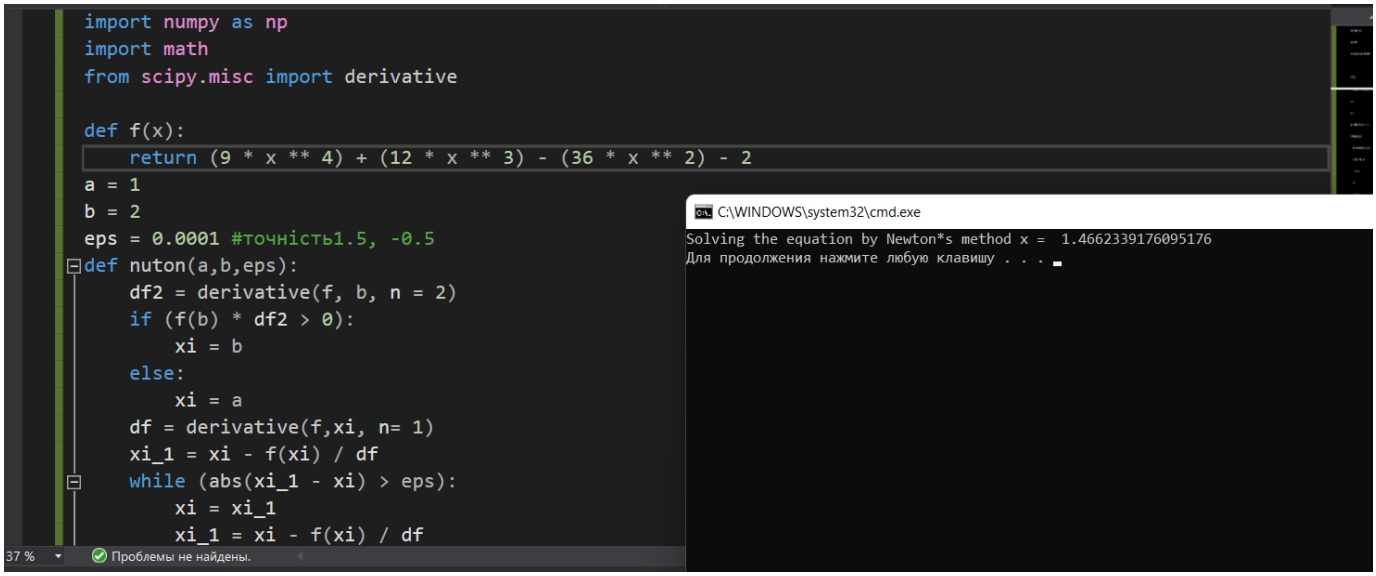
Завдання 1

```
import numpy as np
import math
from scipy.misc import derivative


def f(x):
    return (9 * x ** 4) + (12 * x ** 3) - (36 * x ** 2) - 2
a = 1
b = 2
eps = 0.0001 #точність1.5, -0.5
def nuton(a,b,eps):
    df2 = derivative(f, b, n = 2)
    if (f(b) * df2 > 0):
        xi = b
    else:
        xi = a
    df = derivative(f,xi, n= 1)
    xi_1 = xi - f(xi) / df
    while (abs(xi_1 - xi) > eps):
        xi = xi_1
        xi_1 = xi - f(xi) / df
    return print('Solving the equation by Newton*s method x = ', xi_1)


nuton(a,b,eps)
```



Завдання 2

```
import numpy as np
import math
from scipy.misc import derivative
def f(x):
    return (9 * x ** 4) + (12 * x ** 3) - (36 * x ** 2) - 2
a = 1
b = 2
eps = 0.00001


def komb(a,b,eps):
    if(derivative(f, a, n = 1) * derivative(f, a, n = 2) > 0):
        an = a
        bn = b
        an_1 = an - f(an) * (bn - an) / (f(bn) - f(an))
        bn_1 = bn - f(bn) / derivative(f, bn, n = 1)
    else:
        an = a
        bn = b
        an_1 = an - f(an) / derivative(f, an, n = 1)
        bn_1 = bn - f(bn) * (bn - an) / (f(bn) - f(an))
```

```
    while(abs(bn_1 - an_1) > eps):
        an = an_1
        b1 = bn_1
        if (derivative(f, an, n = 1) * derivative(f, an, n = 2) > 0):
            an_1 = an - f(an) * (bn - an) / (f(bn) - f(an))
            bn_1 = bn - f(bn) - f(bn) / derivative(f, bn, n = 1)
        else:
            an_1 = an - f(an) - f(an) / derivative(f, an, n = 1)
            bn_1 = bn - f(bn) * (bn - an) / (f(bn) - f(an))
    return print("Корень = ", bn_1)

komb(a,b,eps)
```