

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)
```

```
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 #точність 1.5, -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
```

C:\WINDOWS\system32\cmd.exe
Solving the equation by Newton's method x = 1.4662339176095176
Для продолжения нажмите любую клавишу . . .

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)

The screenshot shows a Python IDE with a dark theme. On the left, the code for the `komb` function is written. It imports `numpy` as `np` and `math`, and uses `derivative` from `scipy.misc`. The function `f(x)` is defined as $9x^4 + 12x^3 - 36x^2 - 2$. The `komb` function takes parameters `a`, `b`, and `eps`. It initializes `an = a` and `bn = b`. It then enters a `while` loop that continues as long as `abs(bn_1 - an_1) > eps`. Inside the loop, it checks if `derivative(f, an, n = 1) * derivative(f, an, n = 2) > 0`. If true, it updates `an_1` and `bn_1` using the first set of formulas. If false, it updates them using the second set of formulas. After the loop, it prints the root value. On the right, a terminal window shows the output: "Корень = 1.4662339176095176" and a prompt for the user to press any key to continue.

```

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

112 % Проблемы не найдены.

```

C:\WINDOWS\system32\cmd.exe
Корень = 1.4662339176095176
Для продолжения нажмите любую клавишу . . .