

Розв'язання нелінійних рівнянь

Метод Ньютона

Алгоритми

Лабораторна робота №3

Варіант 7

Виконав: Конча Вадим

## Завдання:

### Вимоги до виконання роботи

1. Складіть програму, що реалізує алгоритм розв'язання рівняння  $f(x) = 0$  методом Ньютона. Фрагмент програми, що власне розв'язує рівняння, оформіть у вигляді окремої процедури.
2. Корені рівняння відокремте графічно і уточніть один з них вказаним методом з точністю до  $\varepsilon = 10^{-4}$ . Результат виведіть на екран.
3. Введіть у програму проміжний друк номера ітерації  $k$ ; на кожній ітерації виведіть значення наближення кореня  $x_k$  та значення  $\Delta_N^{(k)} = |x^{(k)} - x^{(k-1)}|$ , що характеризує досягнуту точність поточного наближення. Виведені результати повинні мати вигляд охайної таблиці.
4. Дослідіть, як похибки поточного наближення до кореня  $\Delta_N^{(k)}$  залежать від номера ітерації. Побудуйте графік залежності  $\lg \Delta_N^{(k)}$  від  $k$  і на його основі з'ясуйте порядок збіжності методу Ньютона.

$$7. e^x + e^{-3x} - 4 = 0.$$

### Код:

```
#include <iostream>
#include <vector>
#include <iomanip>    // std::setprecision

using namespace std;

vector <double> iterations, accuracy;
double a=-1, b=0; //границы
double ab = -0.75; //начальный икс в условиях где то между границ

double func(double x) {
    double rez = exp(x) + exp(-3 * x) - 4; //розрахунок самой функции
    return rez;
}

double func_2(double x) {
    double rez = exp(x) - exp(-3 * x) * 3; //подсчёт похидной
    return rez;
}

double func_3(double x) {
```

```

double rez = exp(x) + exp(-3 * x) * 9; //подсчёт второй похидной
return rez;
}

```

```

void NewTone()
{

```

```

    double x0; //ща будем определять чё из границ - наша точка
отправления

```

```

    if (
        func_2(ab) < 0 && func_3(ab) < 0
        || //or
        func_2(ab) > 0 && func_3(ab) > 0
    )
    {
        x0 = b;
    }
    else if (
        func_2(ab) > 0 && func_3(ab) < 0
        || //or
        func_2(ab) < 0 && func_3(ab) > 0
    )
    {
        x0 = a;
    }

```

```

iterations.push_back(x0); //вектора для вывода
accuracy.push_back(0);
double x1; //доп.

```

```

for (int i = 0; i < 10; i++)
{
    x1 = x0 - func(x0) / func_2(x0);
    iterations.push_back(x1);
    accuracy.push_back(x1 - x0);
    x0 = x1;
}
for (int i = 0; i < iterations.size(); i++)
{

```

```

        cout << i << setprecision(4) << " iteration \t" << iterations[i] << " - value\t"
        << "accuracy: " << accuracy[i] << endl;
    }
}

int main()
{
    NewTone();
}

```

Вивід:

Консоль отладки Microsoft Visual Studio

```

0 iteration      -1 - value      accuracy: 0
1 iteration      -0.7253 - value accuracy: 0.2747
2 iteration      -0.5212 - value accuracy: 0.204
3 iteration      -0.4215 - value accuracy: 0.09976
4 iteration      -0.4017 - value accuracy: 0.01978
5 iteration      -0.401 - value  accuracy: 0.000668
6 iteration      -0.401 - value  accuracy: 7.343e-07
7 iteration      -0.401 - value  accuracy: 8.864e-13
8 iteration      -0.401 - value  accuracy: -5.551e-17
9 iteration      -0.401 - value  accuracy: 0
10 iteration     -0.401 - value  accuracy: 0

```

Також при  $x_0 = 0$  (штучно)

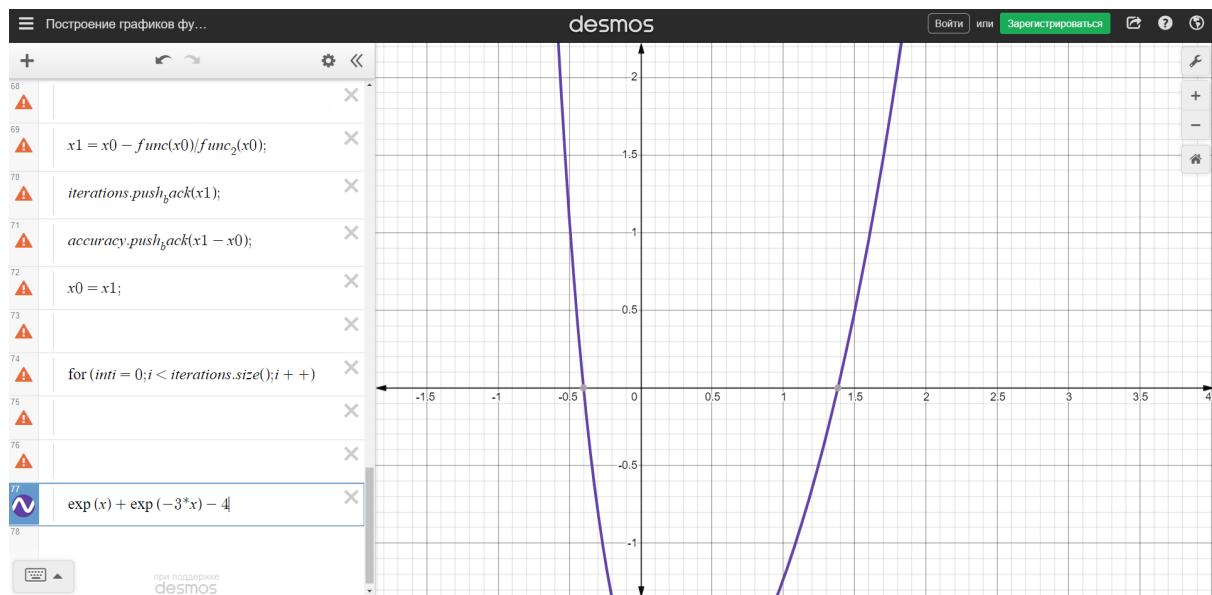
Выбрать Консоль отладки Microsoft Visual Studio

```

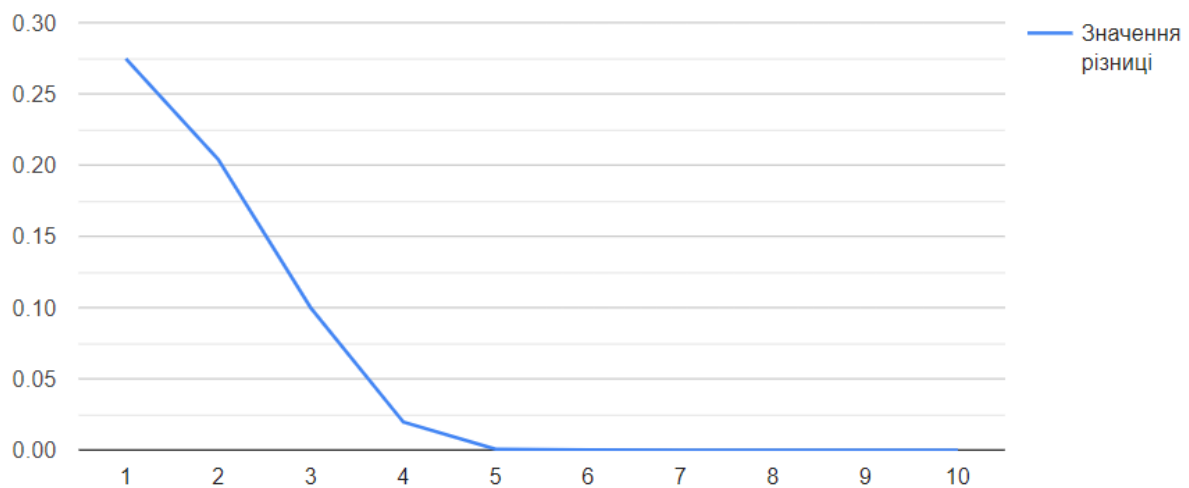
0 iteration      -1 - value      accuracy: 0
1 iteration      -1 - value      accuracy: -1
2 iteration      -0.7253 - value accuracy: 0.2747
3 iteration      -0.5212 - value accuracy: 0.204
4 iteration      -0.4215 - value accuracy: 0.09976
5 iteration      -0.4017 - value accuracy: 0.01978
6 iteration      -0.401 - value  accuracy: 0.000668
7 iteration      -0.401 - value  accuracy: 7.343e-07
8 iteration      -0.401 - value  accuracy: 8.864e-13
9 iteration      -0.401 - value  accuracy: -5.551e-17
10 iteration     -0.401 - value  accuracy: 0

```

## Перевірка:



## Графік:



## Джерела:

[Построение графиков функций \(лекция для старших классов\) \(desmos.com\)](https://desmos.com)

[Метод касательных \(метод Ньютона\) - YouTube](#)

[Метод Ньютона \(дотичних\) \(studfile.net\)](https://studfile.net)

І посібники