



### Цель работы:

- а) освоение методов решения нелинейных уравнений;
- б) совершенствование навыков по алгоритмизации и программированию вычислительных задач.

### Задание:

3.	$tg(ax+b)+cx^2=0$	Ньютона $\varepsilon = 10^{-3}$	$a = 3.01; b = 4; c = -1$
----	-------------------	------------------------------------	---------------------------

### Математическая часть:

Пусть определен интервал  $[a, b]$ , в пределах которого расположен единственный корень  $x^*$  уравнения  $f(x)=0$ , и функция  $f(x)$  удовлетворяет требованиям ("а", "b", "с", см. п.2.1). Пусть в качестве начального приближения выбрано значение  $x_0$  и отвечающая ему точка  $A_0 [x_0, f(x_0)]$  (рис. 2.5).

Найдем следующее приближение как точку пересечения с осью касательной к кривой  $y=f(x)$ , проведенной в точке  $A_0$ . Определив таким образом новую точку  $A_1 [x_1, f(x_1)]$ , найдем следующее приближение  $x_2$  как абсциссу пересечения касательной в точке  $A_1$ .

Повторяя подобные операции, построим последовательность  $x_1, x_2, \dots$ , сходящуюся (при выполнении определенных условий) к искомому корню  $x^*$ .

Для некоторого приближения  $x_n$  уравнение касательной, проведенной в точке  $A_n [x_n, f(x_n)]$ , имеет вид  $y-f(x_n) = f'(x_n)(x-x_n)$ , и значение  $x_{n+1}$ , отвечающее следующему приближению, находится из условия  $y(x_{n+1})=0$ , отвечающего пересечению касательной с осью абсцисс. Отсюда

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, \quad n=0, 1, 2, \dots, \quad (2.11)$$

что и определяет последовательность расчетных формул метода. Можно показать, что если  $f(x)$  удовлетворяет упомянутым требованиям "а", "b", "с" и начальное приближение выбрано так, что

$$f(x_0) \cdot f''(x_0) > 0, \quad (2.12)$$

где  $f''(x_0)=f''(x)$  при  $x=x_0$  то последовательные приближения  $x_0, x_1, \dots$  сходятся к единственному на интервале  $[a, b]$  корню  $x^*$ , который может быть вычислен с любой необходимой точностью.

Точность вычислений может быть оценена с помощью соотношения

$$|x_{n+1} - x^*| \leq |x_{n+1} - x_n|, \quad (2.13)$$

из которого следует, что вычисления могут быть прекращены с получением заданной точности  $|x_{n+1} - x^*| \leq \varepsilon$ , когда выполняется неравенство

$$|x_{n+1} - x_n| \leq \varepsilon, \quad (2.14)$$

аналогичное условию (2.10) метода последовательных приближений.

При несоблюдении условия (2.12) выбора начальной точки  $x_0$  метод может дать неудовлетворительный результат. Например, при выборе  $x_0=a$  (рис. 2.5) последующее приближение  $x_1$  может оказаться вне интервала  $[a, b]$  отделения данного корня  $x^*$ . При этом в соответствии с (2.12) в качестве начальной следует выбрать ту из граничных точек, в которой знаки функции и ее второй производной совпадают.

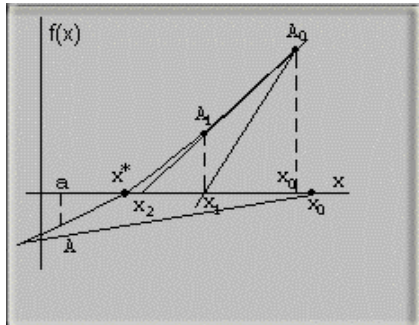


Рис.2.5. Метод касательных

#### Пример 2.4.

Вычислить наименьший положительный корень уравнения  $e^x - 3x = 0$  с точностью  $\varepsilon = 0.0001$ .

Решение. Для отделения возможных положительных корней перепишем уравнение в виде  $e^x = 3x$  и построим графики функций  $y = e^x$ ,  $y = 3x$  (рис. 2.7), показывающие, что имеются два положительных корня, наименьший из которых  $x_1^*$  заключен в интервале  $[0, 1]$ .

Проверяя выполнение требований "a", "b", "c", предъявляемых к функции  $f(x) = e^x - 3x$ , находим  $f'(x) = e^x - 3$ ,  $f''(x) = e^x$ ,  $f'(x) < 0$ ,  $f''(x) > 0$

при  $0 \leq x \leq 1$  производные непрерывны и сохраняют знаки на рассматриваемом интервале. При этом  $f(x)$  так же непрерывна, но  $f(0) \cdot f(1) < 0$ .

Выбирая начальную точку  $x_0$ , находим  $f(0) \cdot f''(0) = (e^x - 3x) e^x \big|_{x=0} = 1 > 0$ ,

$$f(1) \cdot f''(1) = (e^x - 3x) e^x \big|_{x=1} = (e-3)e < 0.$$

Следовательно, необходимо принять  $x_0=0$ . Последующие приближения вычисляются по формулам (11), которые в данном случае принимают вид

$$x_{n+1} = x_n - \frac{e^{x_n} - 3x_n}{e^{x_n} - 3} \quad (n=0,1,2,\dots).$$

Результаты следующих отсюда вычислений сведены в табл. 2.1 При этом величина

$$\varepsilon_n = \frac{f(x_n)}{f'(x_n)} = x_{n+1} - x_n$$

в соответствии с (2.13) и (2.14) определяет точность приближения  $x_{n+1}$  к искомому корню  $x^*$ .

Таблица 2.1

n	$x_n$	$e^x$	$3x$	$f(x_n)$	$f'(x_n)$	$\varepsilon_n$
0	0.00000	1.0000	0.0000	1.0000	-2.0	-0.500000
1	0.50000	1.6500	1.5000	0.1500	-1.35	-0.110000
2	0.61000	1.8404	1.8300	0.0104	-1.16	-0.009000
3	0.61900	1.8571	1.8570	0.0001	-1.14	-0.000088
4	0.61909	1.8573	1.8573	0.0000	-	-

В качестве искомого значения корня принимаем  $x \approx 0.619$ .

## Аналитические расчеты:

### Решение шаг 1

 показать шаги

Итак, нам необходимо найти корень уравнения

$$f(x) = \operatorname{tg}(3.01 \cdot x + 4) - x^2 = 0$$

Формула для поиска корня уравнения по методу Ньютона имеет вид:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

В этой формуле:

$$f'(x_n) = \left( \frac{df(x)}{dx} \right) \bigg|_{x=x_n} \quad \text{— значение производной функции } f(x) \text{ в точке } x_n.$$

=> Находим  $\frac{df(x)}{dx}$ :

$$\frac{df(x)}{dx} = 3.01 \cdot \sec^2(3.01 \cdot x + 4) - 2.0 \cdot x$$

#### Шаг 6

Вычисления представим в виде таблицы:

iter	$x_n$	$f(x_n)$	$f'(x_n)$	$ f(x_n) $	$ x_n - x_{n-1} $
0	0.0	1.15782128234958	7.0450558668035063	1.15782128234958	—
1	-0.1643452236915913	0.3536559347922388	3.774857690955678	0.353655934792239	0.164345223691591
2	-0.2580324468257444	0.0153314030903857	3.5462607889828474	0.0153314030903857	0.0936872231341532
3	-0.2623557056052497	$-5.4794336549 \cdot 10^{-6}$	3.5489694371513959	$5.4794336549 \cdot 10^{-6}$	0.00432325877950536

#### Шаг 7

На 3 итерации сработал критерий останова вычислений по отличию значения функции от нуля, т.е.  $|f(x_3)| < \varepsilon$ :

$$5.4794336549 \cdot 10^{-6} < 0.001$$

#### Ответ

Итак, мы нашли корень нашего уравнения с заданной точностью:

$$x = -0.2623557056052497$$

#### Шаг 2

Теперь у нас есть все необходимые данные для того, чтобы начать поиск корня по методу Ньютона.

Запишем формулу Ньютона для нашего уравнения:

$$x_{n+1} = x_n - \frac{\operatorname{tg}(3.01 \cdot x_n + 4) - x_n^2}{3.01 \cdot \sec^2(3.01 \cdot x_n + 4) - 2.0 \cdot x_n}$$

#### Шаг 3

Покажем как используя приведённую выше формулу и зная значение  $x_0$ , вычислить значение  $x_1$ .

Если  $n = 1$ , то формула принимает вид:

$$x_1 = x_0 - \frac{\operatorname{tg}(3.01 \cdot x_0 + 4) - x_0^2}{3.01 \cdot \sec^2(3.01 \cdot x_0 + 4) - 2.0 \cdot x_0}$$

#### Шаг 4

Подставляем значение  $x_0 = 0$  в приведённую выше формулу:

$$x_1 = -\frac{\operatorname{tg}(4)}{3.01 \cdot \sec^2(4)}$$

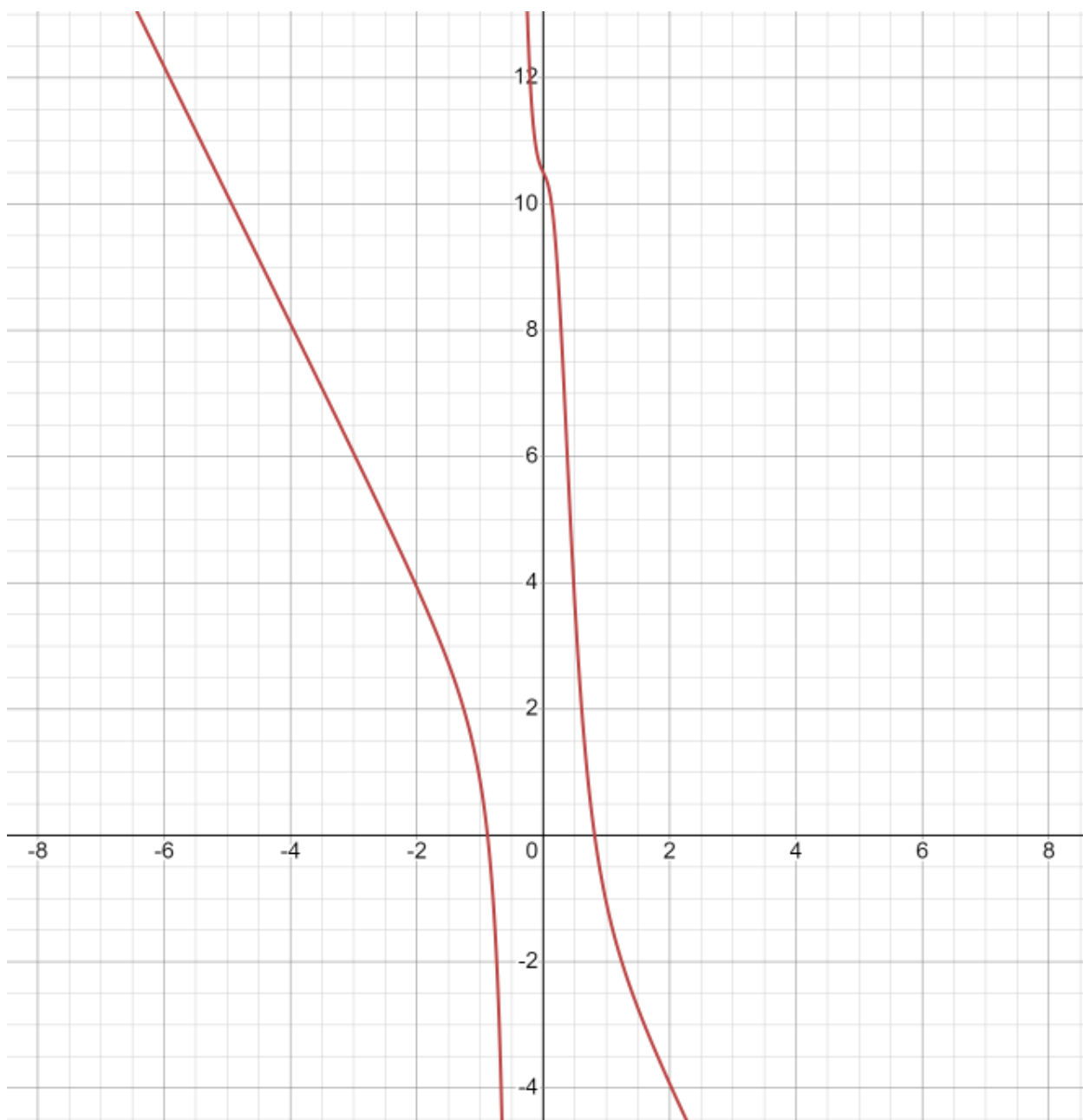
#### Шаг 5

 показать шаги

Упрощая, получаем значение для  $x_1$ :

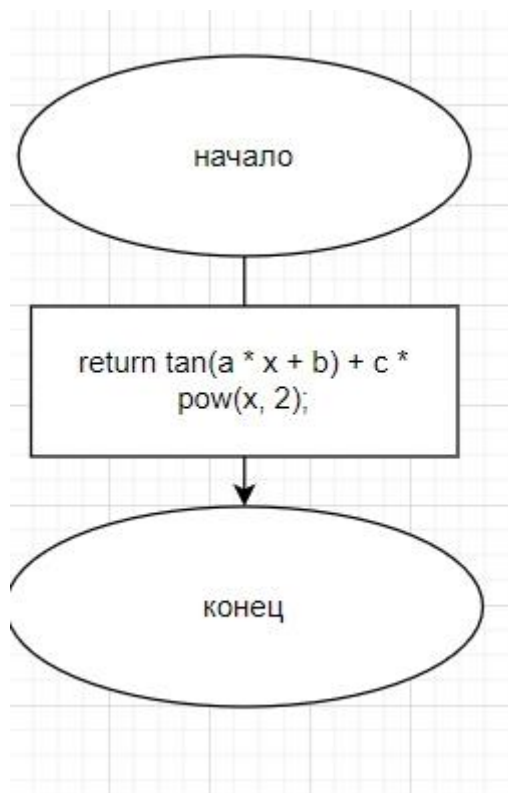
$$x_1 = -0.1643452236915913$$

График функции:

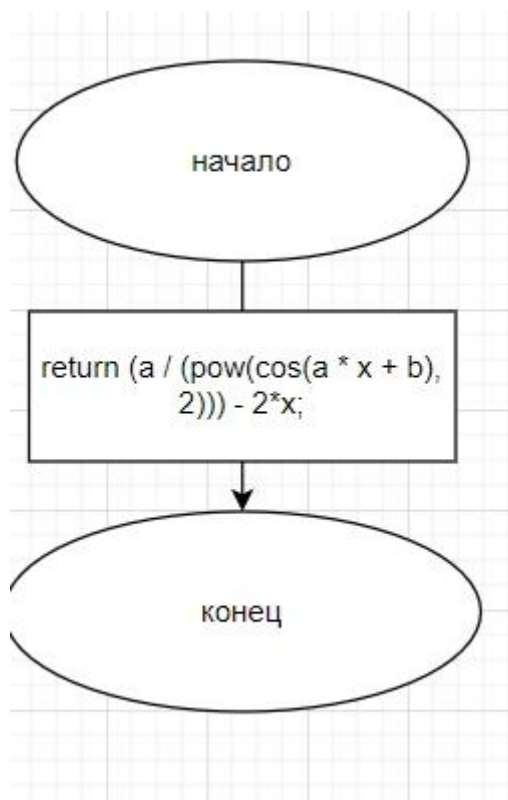


**Схема решения задачи:**

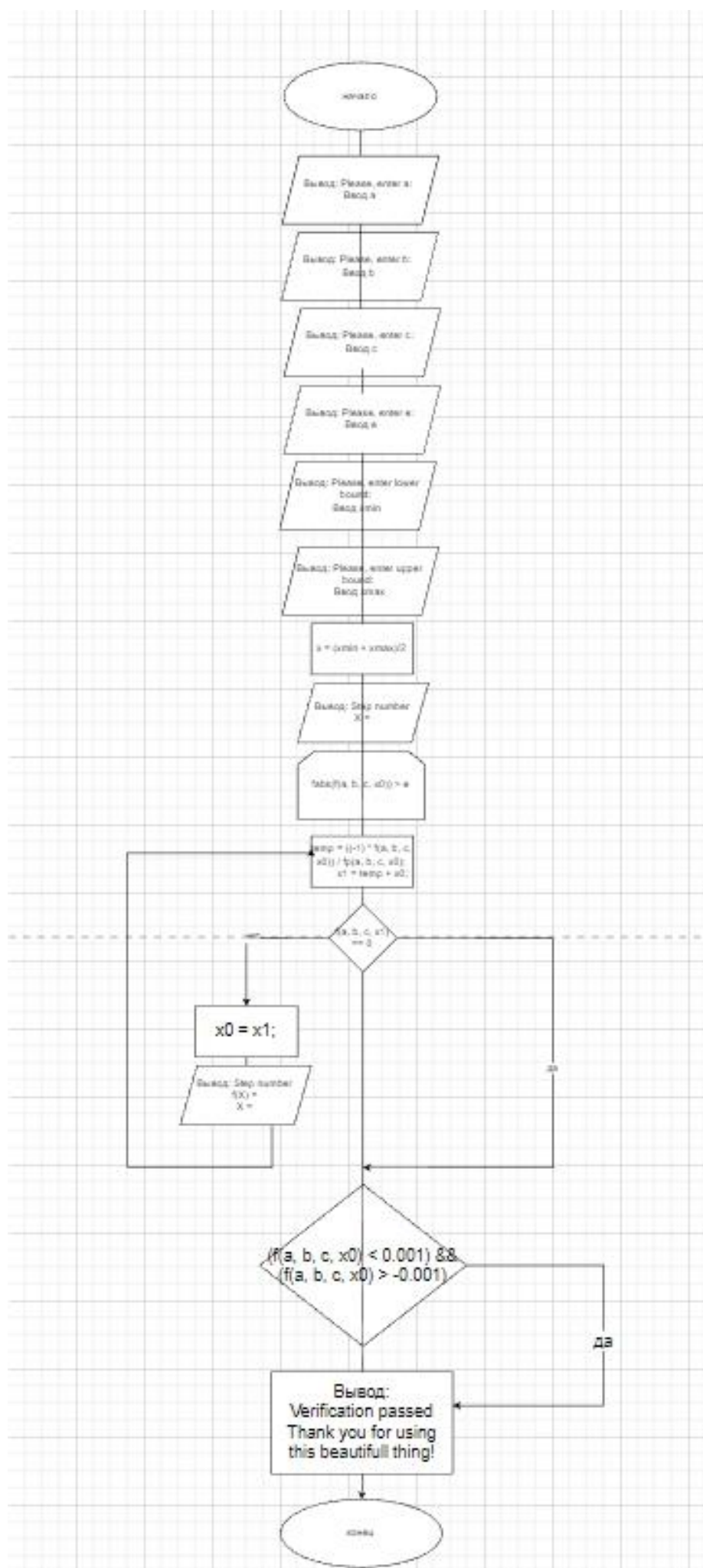
1) Блок схема функции  $f$ :



2) Блок схема функции fr:



3) Блок схема функции main:





## Листинг кода программы:

```
#include <iostream>

#include <cmath>

#include <conio.h>

using namespace std;

double f(double a, double b, double c, double x) {
    return tan(a * x + b) + c * pow(x, 2);
}

double fp(double a, double b, double c, double x) {
    return (a / (pow(cos(a * x + b), 2))) - 2*x;
}

int main()
{
    double a, b, c, x0, x1, e, count, temp, xmin, xmax;

    //a = 3.01;
    //b = 4;
    //c = -1;
    //e = 0.001;

    count = 1;

    cout << "PR1 Bobrovich 4136" << endl;

    cout << "3 var:\ntg(ax+b)+cx2=0\nNewton's method\n" << endl;
    cout << endl;

    cout << "Press any key to continue." << endl;
    _getch();
    system("cls");

    cout << "Please, enter a: ";

    cin >> a;

    cout << "Please, enter b: ";

    cin >> b;

    cout << "Please, enter c: ";
```

```

cin >> c;

cout << "Please, enter e: ";

cin >> e;


cout << "Please, enter lower bound: ";

cin >> xmin;

cout << "Please, enter upper bound: ";

cin >> xmax;


x0 = (xmin + xmax) / 2;


cout << "Step number: " << count << endl;

count++;

cout << "X: " << x0 << endl;

cout << endl;


while (fabs(f(a, b, c, x0)) > e) {
    temp = ((-1) * f(a, b, c, x0)) / fp(a, b, c, x0);
    x1 = temp + x0;
    if (f(a, b, c, x1) == 0) {
        break;
    }
    x0 = x1;

    cout << "Step number: " << count << endl;

    count++;

    cout << "f(X): " << f(a, b, c, x0) << endl;

    cout << "X: " << x0 << endl;

    cout << endl;
}

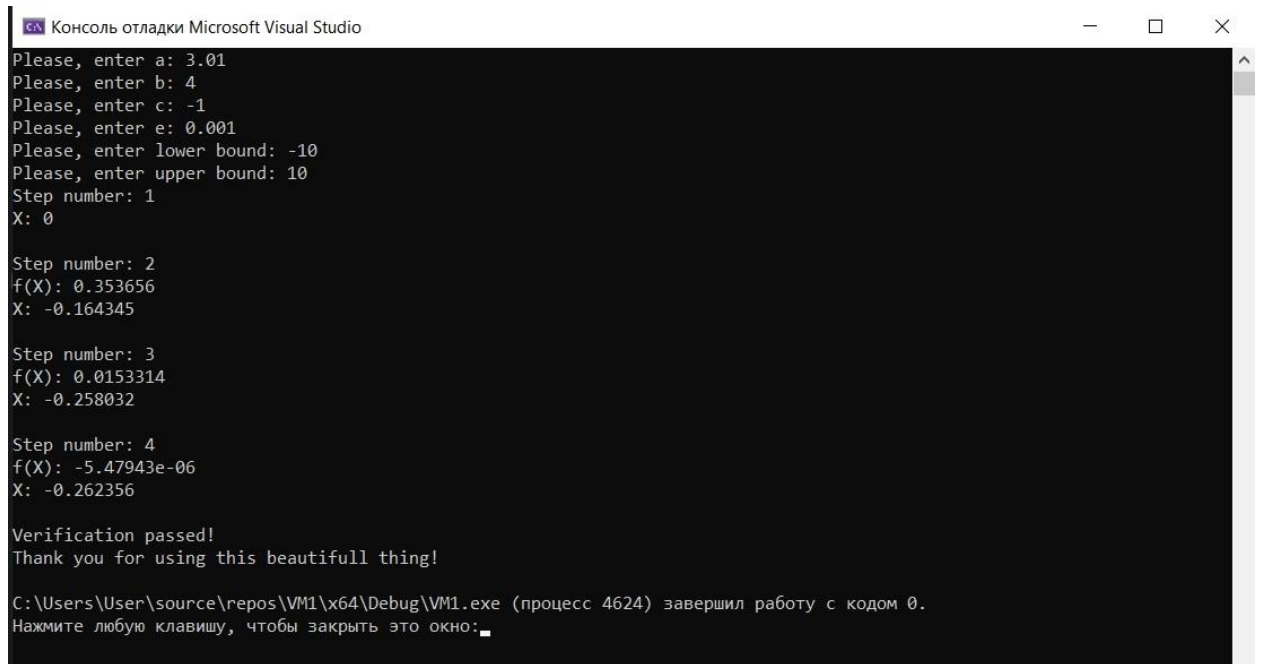
if ((f(a, b, c, x0) < 0.001) && (f(a, b, c, x0) > -0.001)) {
    cout << "All is right!" << endl;
}

cout << "Thank you for using this beautifull thing!" << endl;

return 0;
}

```

## Результаты программных расчетов:



```
Консоль отладки Microsoft Visual Studio
Please, enter a: 3.01
Please, enter b: 4
Please, enter c: -1
Please, enter e: 0.001
Please, enter lower bound: -10
Please, enter upper bound: 10
Step number: 1
X: 0

Step number: 2
f(X): 0.353656
X: -0.164345

Step number: 3
f(X): 0.0153314
X: -0.258032

Step number: 4
f(X): -5.47943e-06
X: -0.262356

Verification passed!
Thank you for using this beautifull thing!

C:\Users\User\source\repos\VM1\x64\Debug\VM1.exe (процесс 4624) завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно: _
```

## Сравнение результатов программных и аналитических расчетов:

Исходя из результатов мы видим, что результаты не совсем сходятся, но это не означает, что программа производит неверный расчет. В аналитических данных не записан самый первый интервал  $[-10, 10]$ . Это первая причина несхождения. Вторая - округление. В аналитических расчетах, когда точность равна 0.00122, остановили расчет, так как при округлении это будет равно 0.001, а значит по условию дальнейшие шаги не выполняются, так как задана точность. В программе же не идет округление, поэтому рассчитывается еще один шаг. Из-за расхождения последних интервалов, искомые корни, так же отличаются.

## Вывод

В ходе выполнения практической работы №1 был освоен метод решения нелинейных уравнений – метод Ньютона. Также были улучшены навыки по алгоритмизации и программированию вычислительной задачи на языке C++ в программе Microsoft Visual Studio.