

Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет
информатики и радиоэлектроники»

Факультет компьютерных систем и сетей
Кафедра информатики
Дисциплина «Методы численного анализа»

Лабораторная работа №3

на тему:

«Численное решение нелинейных уравнений»
БГУИР 1-40 04 01

Выполнила
студентка гр. 253504:

Лавренова А. С.

Руководитель:

Анисимов В.Я.

Минск 2023

Содержание

1. Цель работы
2. Теоретические сведения
3. Условие
4. Блок схемы
5. Программная реализация
6. Решение задания
7. Тестовые примеры

Заключение

1.Цель работы

- 1) Изучить методы численного решения нелинейных уравнений (метод простой итерации, метод хорд, метод Ньютона)
- 2) Исследовать скорость сходимости итерационных процедур
- 3) Составить программу численного решения нелинейных уравнений методами бисекции, хорд, Ньютона
- 4) Проверить правильность работы программы на тестовых примерах
- 5) Численно решить нелинейное уравнение заданного варианта
- 6) Сравнить число итераций, необходимого для достижения заданной точности вычисления разными методами

2.Теоретический сведения

Численное решение нелинейного уравнения $f(x)=0$ заключается в вычислении с заданной точностью значения всех или некоторых корней уравнения и распадается на несколько задач: *во-первых*, надо исследовать количество и характер корней (вещественные или комплексные, простые или кратные), *во-вторых*, определить их приближенное расположение, т.е. значения начала и конца отрезка, на котором лежит только один корень, *в-третьих*, выбрать интересующие нас корни и вычислить их с требуемой точностью. Вторая задача называется **отделением корней**. Решив ее, по сути дела, находят приближенные значения корней с погрешностью, не превосходящей длины отрезка, содержащего корень. Отметим два простых приема отделения действительных корней уравнения - *табличный* и *графический*. Первый прием состоит в вычислении таблицы значений функции $f(x)$ в заданных точках x_i и использовании следующих теорем математического анализа:

1. Если функция $y=f(x)$ непрерывна на отрезке $[a,b]$ и $f(a)f(b)<0$, то внутри отрезка $[a,b]$ существует по крайней мере один корень уравнения $f(x)=0$.
2. Если функция $y=f(x)$ непрерывна на отрезке $[a,b]$, $f(a)f(b) < 0$ и $f'(x)$ на интервале (a,b) сохраняет знак, то внутри отрезка $[a,b]$ существует единственный корень уравнения $f(x)=0$.

Таким образом, если при некотором k числа $f(x_k)$ и $f(x_{k+1})$ имеют разные знаки, то это означает, что на интервале (x_k, x_{k+1}) уравнение имеет по крайней мере один действительный корень нечетной кратности (точнее – нечетное число корней). Выявить по таблице корень четной кратности очень сложно.

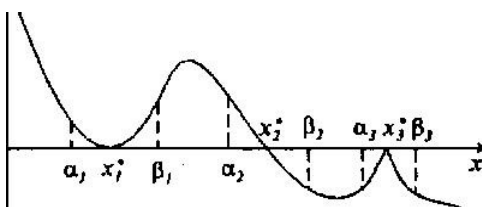


Рис. 1

На рис.1 представлены три наиболее часто встречающиеся ситуации:

а) кратный корень: $f'(x^*)=0, f(a1) * f(b1) > 0$;

б) простой корень: $f'(x^*)=0, f(a2) * f(b2) < 0$;

в) вырожденный корень: $f'(x^*)$ не существует, $f(a3) * f(b3) > 0$.

Как видно из рис.1, в первых двух случаях значение корня совпадает с точкой экстремума функции и для нахождения таких корней рекомендуется использовать методы поиска минимума функции.

Для определения числа корней на заданном промежутке используется **Теорема Штурма** : Если $f(x)$ является многочленом и уравнение $f(x)=0$ не имеет кратных корней на промежутке $[a, b]$, то число корней этого уравнения, лежащих на таком промежутке, совпадает с числом $N(a) - N(b)$, где функция N определяется следующим образом.

Строим ряд Штурма $f_0(x), f_1(x), f_2(x), \dots, f_m(x)$, где

$$f_0(x) = f(x),$$

$$f_1(x) = f'(x),$$

$$f_i(x) = \text{остаток от деления } f_{i-2}(x) \text{ на } f_{i-1}(x), \text{ взятый с обратным}$$

знаком Функция $N(x)$ определяется как число перемен знака в ряде

Штурма,

если подставить в функции ряда значение x

Для отделения корней можно использовать график функции $y=f(x)$.

Корнями уравнения являются те значения x , при которых график функции пересекает ось абсцисс. Построение графика функции даже с малой точностью обычно дает представление о расположении и характере корней уравнения (иногда позволяет выявить даже корни четной кратности). Если построение графика функции $y=f(x)$ вызывает затруднение, следует преобразовать исходное уравнение к виду $\varphi_1(x)=\varphi_2(x)$ таким образом, чтобы графики функций $y=\varphi_1(x)$ и $y=\varphi_2(x)$ были достаточно просты. Абсциссы точек пересечения этих графиков и будут корнями уравнения.

Допустим, что искомый корень уравнения отделен, т.е. найден отрезок $[a, b]$, на котором имеется только один корень уравнения. Для вычисления корня с требуемой точностью ε обычно применяют какую-либо итерационную процедуру **уточнения корня**, строящую числовую последовательность значений x_n , сходящуюся к искомому корню уравнения. Начальное приближение x_0 выбирают на отрезке $[a, b]$, продолжают вычисления, пока не выполнится неравенство $|x_{n-1} - x_n| < \varepsilon$, и считают, что x_n есть корень уравнения, найденный с заданной точностью. Имеется множество различных методов построения таких последовательностей и выбор алгоритма – весьма важный момент при практическом решении задачи. Немалую роль при этом играют такие свойства метода, как простота, надежность, экономичность, важнейшей характеристикой является его

скорость сходимости. Последовательность x_n , сходящаяся к пределу x^* ,

имеет скорость сходимости порядка α , если при

$$n \rightarrow \infty: |x_{n+1} - x^*| = O(|x_n - x^*|^\alpha). \text{ При } \alpha=1 \text{ сходимость называется}$$

линейной, при $1 < \alpha < 2$ – сверхлинейной, при $\alpha=2$ – квадратичной. С ростом α алгоритм, как правило, усложняется и условия сходимости становятся более жесткими. Рассмотрим наиболее распространенные итерационные методы

уточнения корня.

Метод простых итераций. Вначале уравнение $f(x)=0$ преобразуется к эквивалентному уравнению вида $x=\varphi(x)$. Это можно сделать многими способами, например, положив $\varphi(x)=x+\diamond(x)f(x)$, где $\diamond(x)$ – произвольная непрерывная знакопостоянная функция. Выбираем некоторое начальное приближение x_0 и вычисляем дальнейшие приближения по формуле

$$x_k = \varphi(x_{k-1}), k=1, 2, \dots$$

Метод простых итераций не всегда обеспечивает сходимость к корню уравнения. Достаточным условием сходимости этого метода является выполнение неравенства $|\varphi'(x)| \leq q < 1$ на отрезке, содержащем корень и все приближения x_n . Метод имеет линейную скорость сходимости и справедливы следующие оценки:

$$\left| x_n - x^{**} \right| < q \left| x_n - x_{n-1} \right|, \text{ если } \varphi'(x) > 0,$$

$$\left| x_n - x^{**} \right| < \frac{1-q}{q} \left| x_n - x_{n-1} \right|, \text{ если } \varphi'(x) < 0.$$

Метод имеет простую геометрическую интерпретацию: нахождение корня уравнения $f(x)=0$ равносильно обнаружению неподвижной точки функции $x=\varphi(x)$, т.е. точки пересечения графиков функций $y=\varphi(x)$ и $y=x$. Если производная $\varphi'(x) < 0$, то последовательные приближения колеблются около корня, если же производная $\varphi'(x) > 0$, то последовательные приближения сходятся к корню монотонно.

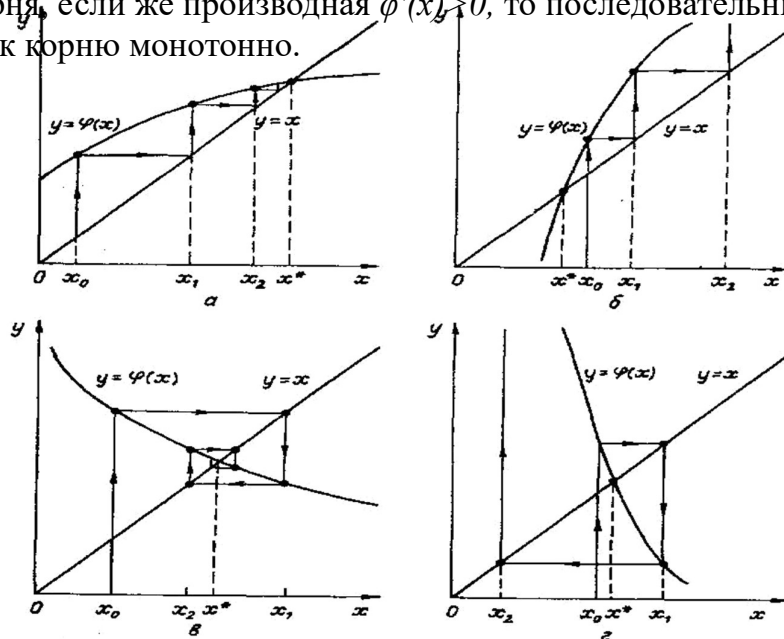


Рис. 2. Метод простых итераций: а - односторонний сходящийся

процесс; б - односторонний расходящийся процесс; в - двухсторонний сходящийся процесс; г - двухсторонний расходящийся процесс

Рассмотрим процесс графически (рис. 2). Из графиков видно, что при $\varphi'(x) < 0$ и при $\varphi'(x) > 0$ возможны как сходящиеся, так и расходящиеся итерационные процессы. Скорость сходимости зависит от абсолютной величины производной $\varphi(x)$. Чем меньше $|\varphi'(x)|$ вблизи корня, тем быстрее сходится процесс.

Метод хорд. Пусть дано уравнение $f(x) = 0$, $a \leq x \leq b$, где $f(x)$ – дважды непрерывно дифференцируемая функция. Пусть выполняется условие $f(a) \cdot f(b) < 0$ и проведено отделение корней, то есть на данном интервале (a, b) находится один корень уравнения. При этом, не ограничивая общности, можно считать, что $f(b) > 0$.

Пусть функция f выпукла на интервале (a, b) (см. рис. 3).

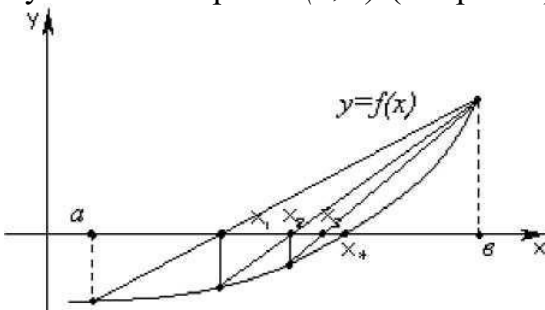


Рис.3

Заменим график функции хордой (прямой), проходящей через точки $M_0(a, f(a))$ и $M_1(b, f(b))$. Уравнение прямой, проходящей через две заданные точки,

можно записать в виде
$$\frac{y - y_1}{y_2 - y_1} = \frac{x - x_1}{x_2 - x_1}$$
. В нашем случае получим:

$$\frac{y - f(a)}{f(b) - f(a)} = \frac{x - a}{b - a}$$
. Найдем точку пересечения хорды с осью Ox .

Полагая $y = 0$, получаем из предыдущего уравнения:

$$x_1 = a - \frac{f(a)}{f(b) - f(a)} \cdot (b - a)$$
. Теперь возьмем интервал (x_1, b) в качестве

исходного и повторим вышеописанную процедуру (см. рис. 3). Получим

$$x_2 = x_1 - \frac{f(x_1)}{f(b) - f(x_1)} \cdot (b - x_1)$$
. Продолжим процесс. Каждое

последующее приближение вычисляется по рекуррентной формуле

$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f(b) - f(x_{n-1})} \cdot (b - x_{n-1}), \quad n = 1, 2, \dots$$

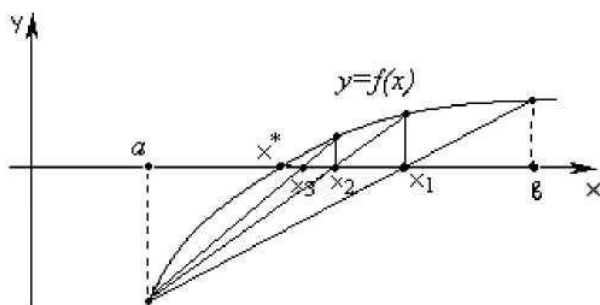


Рис. 4

уравнение прямой соединяющей точки $M_0(a, f(a))$ и $M_1(b, f(b))$ запишем в виде $\frac{y-f(b)}{f(a)-f(b)} = \frac{x-b}{a-b}$. Найдем точку пересечения хорды с осью Ох:

$x_1 = b - \frac{f(b)}{f(a)-f(b)}(a-b)$. Теперь возьмем интервал (a, x_1) в качестве

исходного и найдем точки пересечения хорды, соединяющей точки $(a, f(a))$ и $(x_1, f(x_1))$, с осью абсцис (с м. р и с. 4). Получим

$x_2 = x_1 - \frac{f(x_1)}{f(a)-f(x_1)}(a-x_1)$. Повторяя данную процедуру, получаем

рекуррентную формулу:

$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f(a)-f(x_{n-1})}(a-x_{n-1}), n = 1, 2, \dots$$

(3.2)

Описанный выше метод построения рекуррентных последовательностей (3.1) и (3.2) называется методом хорд. Для использования метода хорд нужно было бы предварительно найти точки перегиба и выделить участки, на которых функция не меняет характер выпуклости. Однако на практике поступают проще: в случае $f(b) f'(b) > 0$ для построения рекуррентной последовательности применяются формулы (3.1), а в случае, когда $f(a) f'(a) > 0$, применяют формулы (3.2).

Метод Ньютона (касательных). Для начала вычислений требуется задание одного начального приближения x_0 , последующие приближения вычисляются по формуле

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, f'(x_n) \neq 0$$

Метод имеет квадратичную скорость сходимости для простого корня, но очень чувствителен к выбору начального приближения. При произвольном начальном приближении итерации сходятся, если всюду

$|f(x)f''(x)| < (f'(x))^2$, в противном случае сходимость будет только при x_0 , достаточно близком к корню. Существует несколько достаточных условий сходимости. Если производные $f'(x)$ и $f''(x)$ сохраняют знак в окрестности корня, рекомендуется выбирать x_0 так, чтобы $f(x)f''(x) > 0$. Если, кроме

этого, для отрезка $[a, b]$, содержащего корень, выполняются условия

$\left| \frac{f(a)}{f'(a)} \right| < b - a$, $\left| \frac{f(b)}{f'(b)} \right| < b - a$, то метод сходится для любых $a \leq x_0 \leq b$.

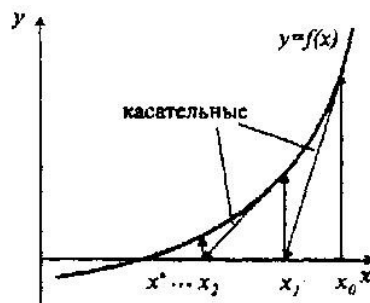


Рис. 5

Метод Ньютона получил также второе название *метод касательных* благодаря геометрической иллюстрации его сходимости, представленной на рис. 5.

Метод Ньютона позволяет находить как простые, так и кратные корни.

Основной его недостаток — малая область сходимости и необходимость вычисления производной.

Также хорошей практикой считается построение графиков, ведь имея перед глазами график мы сразу сможем найти приблизительные промежутки, где находится корень или даже возможность сразу найти корень. Ну и это просто очень сильный инструмент для получения подробной информации о поведении функции на определенном промежутке.

3. Условие:

ЗАДАНИЕ.

1) Используя теорему Штурма определить число корней уравнения:

$x^3 + ax^2 + bx + c = 0$ на отрезке $[-10, 10]$. Значения коэффициентов уравнения взять из таблицы.

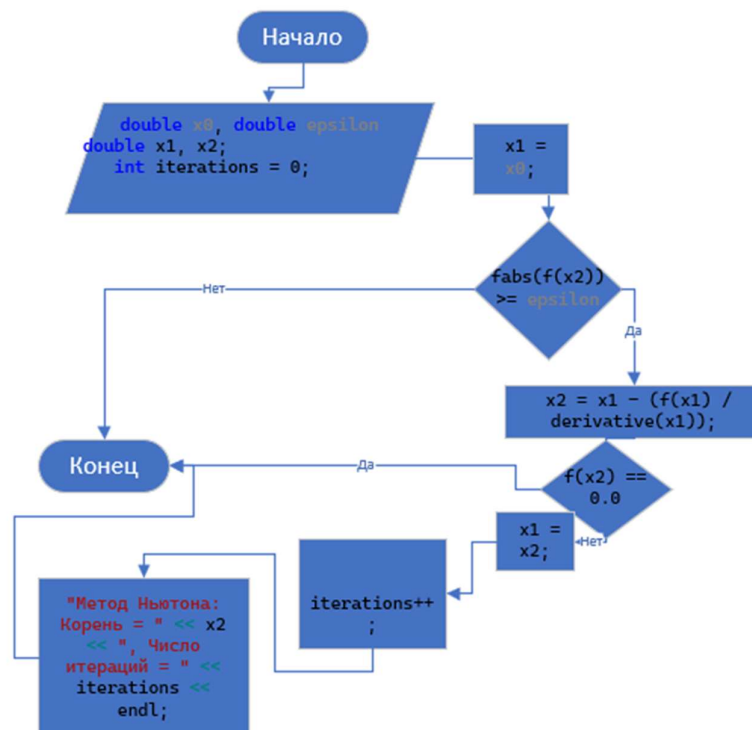
2) Отделить все корни, лежащие на данном отрезке.

3) Вычислить наименьший из корней сначала методом половинного деления, а затем методом хорд и методом Ньютона. Сравнить число необходимых итераций в обоих методах. Точность до 0.0001.

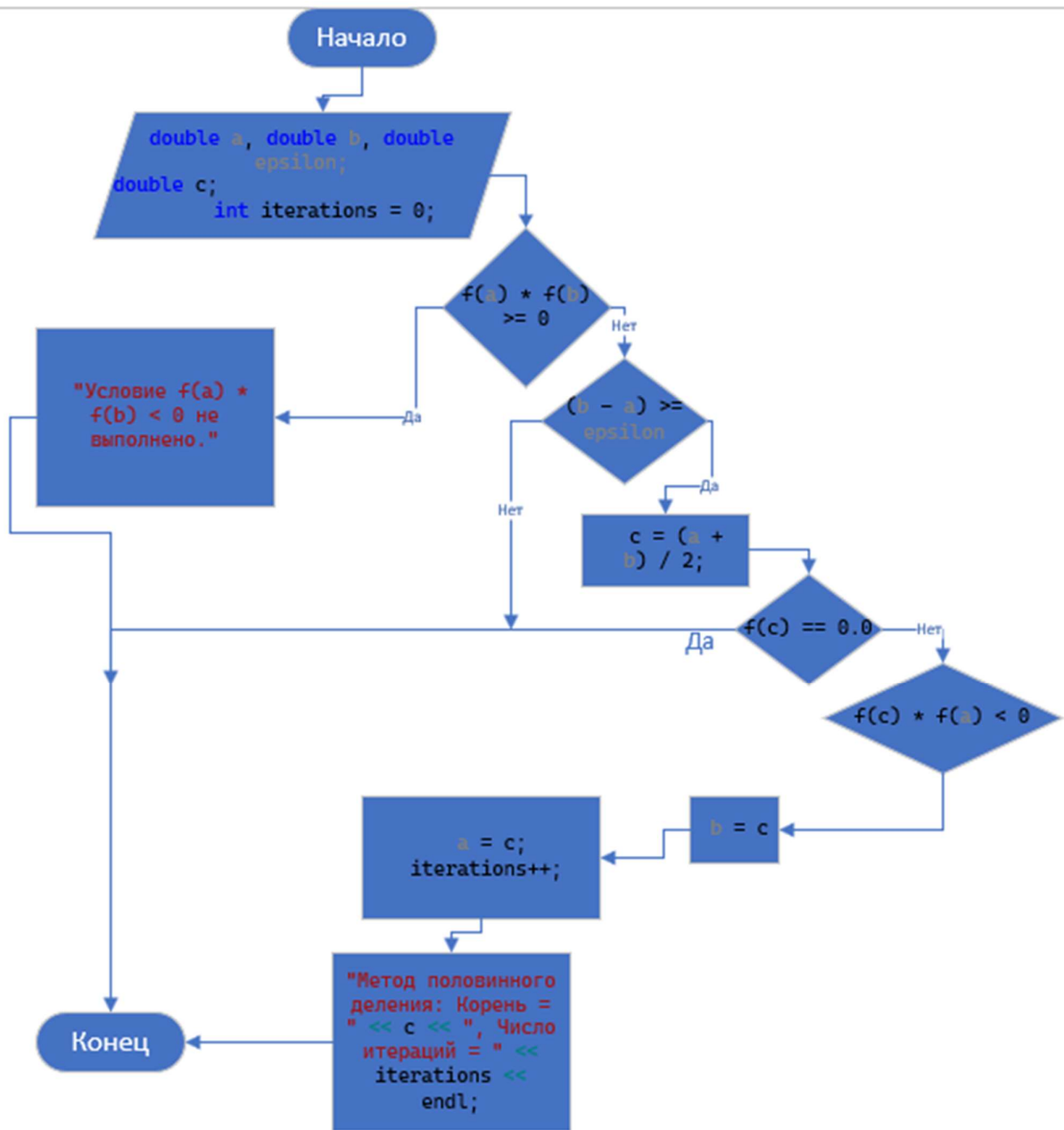
4. Блок схемы

Решение задачи реализовано на языке C++. Основное решение представлено тремя функциями в которых реализовано три метода.

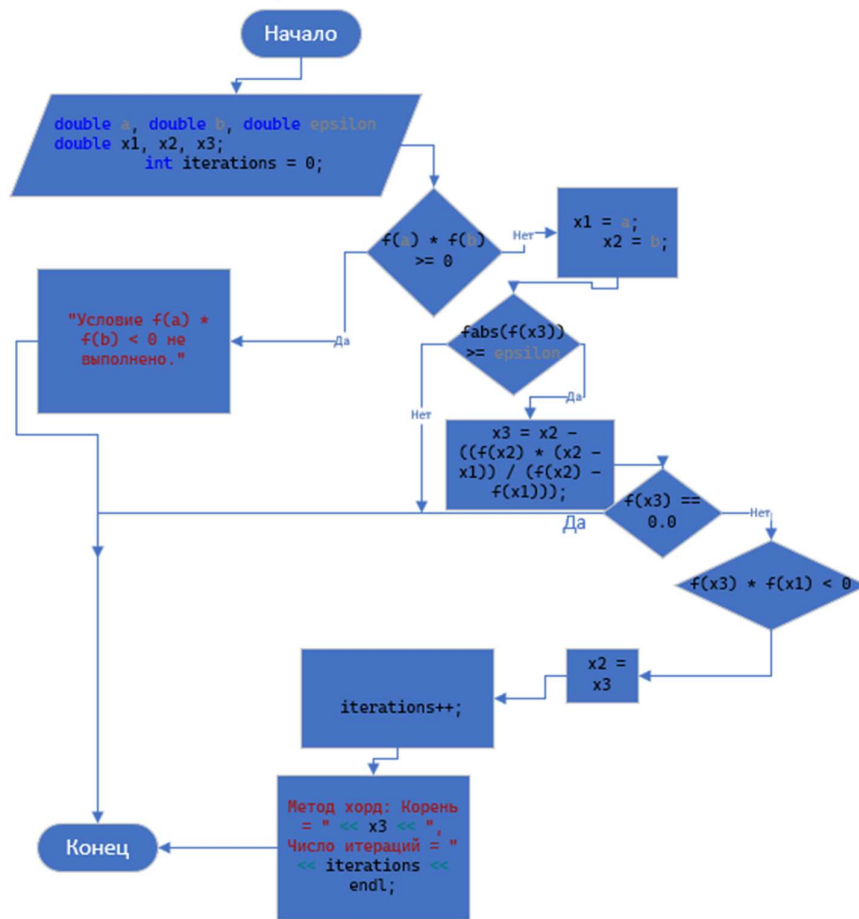
Блок схема метода Ньютона:



Блок схема метода половинного деления:



Блок схема метода хорд:



5. Программная реализация

Метода половинного деления:

```

void bisectionMethod(double a, double b, double epsilon) {
    double c;
    int iterations = 0;

    if (f(a) * f(b) >= 0) {
        cout << "Условие f(a) * f(b) < 0 не выполнено." << endl;
        return;
    }

    while ((b - a) >= epsilon) {
        c = (a + b) / 2;

        if (f(c) == 0.0) {
            break;
        }
        else if (f(c) * f(a) < 0) {
            b = c;
        }
        else {
            a = c;
        }

        iterations++;
    }

    cout << "Метод половинного деления: Корень = " << c << ", Число итераций = " << iterations << endl;
}
  
```

Метод хорд:

```
void chordMethod(double a, double b, double epsilon) {
    double x1, x2, x3;
    int iterations = 0;

    if (f(a) * f(b) >= 0) {
        cout << "Условие f(a) * f(b) < 0 не выполнено." << endl;
        return;
    }

    x1 = a;
    x2 = b;

    do {
        x3 = x2 - ((f(x2) * (x2 - x1)) / (f(x2) - f(x1)));

        if (f(x3) == 0.0) {
            break;
        }
        else if (f(x3) * f(x1) < 0) {
            x2 = x3;
        }
        else {
            x1 = x3;
        }

        iterations++;
    } while (fabs(f(x3)) >= epsilon);

    cout << "Метод хорд: Корень = " << x3 << ", Число итераций = " << iterations << endl;
}
```

Метод Ньютона:

```
void newtonMethod(double x0, double epsilon) {
    double x1, x2;
    int iterations = 0;

    x1 = x0;

    do {
        x2 = x1 - (f(x1) / derivative(x1));

        if (f(x2) == 0.0) {
            break;
        }

        x1 = x2;
        iterations++;
    } while (fabs(f(x2)) >= epsilon);

    cout << "Метод Ньютона: Корень = " << x2 << ", Число итераций = " << iterations << endl;
}
```

6. Решение задания

Введём изначальные данные:

```
double f(double x) {  
    return x * x * x + 38.4621 * x * x + 364.594 * x + 914.196; // уравнение, для которого ищем корни  
}  
  
double proizvodnai(double x) {  
    return 3 * x * x + 76.9242 * x + 364.594; // производная уравнения  
}
```

Получаем:

```
Число корней на отрезке [-10, 10] = 2  
Условие f(a) * f(b) < 0 не выполнено.  
Условие f(a) * f(b) < 0 не выполнено.  
Метод Ньютона: Корень = -8.7648, Число итераций = 4
```

7. Тестовые примеры

Пример 1:

Условие:

```
double f(double x) {  
    return x * x * x * x - 2 * x * x * x - 7 * x * x + 8 * x + 1; // уравнение, для которого ищем корни  
}  
  
double proizvodnai(double x) {  
    return 4 * x * x * x - 6 * x * x - 14 * x; // производная уравнения  
}
```

Ответ:

```
Число корней на отрезке [-10, 10] = 4  
Условие f(a) * f(b) < 0 не выполнено.  
Условие f(a) * f(b) < 0 не выполнено.  
Метод Ньютона: Корень = -2.35009, Число итераций = 13
```

Пример 2:

Условие:

```
double f(double x) {  
    return 10 * x - 4; // уравнение, для которого ищем корни  
}  
  
double proizvodnai(double x) {  
    return 10; // производная уравнения  
}
```

Ответ:

```
Число корней на отрезке [-10, 10] = 1  
Метод половинного деления: Корень = 0.400009, Число итераций = 18  
Метод хорд: Корень = 0.4, Число итераций = 1  
Метод Ньютона: Корень = 0.4, Число итераций = 1
```

Пример 3:

Условие:

```
double f(double x) {  
    return x * x * x + 8 * x + 10; // уравнение, для которого ищем корни  
}  
  
double proizvodnai(double x) {  
    return 3 * x * x + 8; // производная уравнения  
}
```

Ответ:

```
Число корней на отрезке [-10, 10] = 1  
Метод половинного деления: Корень = -1.08864, Число итераций = 18  
Метод хорд: Корень = -1.08869, Число итераций = 118  
Метод Ньютона: Корень = -1.0887, Число итераций = 7
```

ЗАКЛЮЧЕНИЕ

Таким образом, в ходе выполнения лабораторной работы были изучены методы численного решения нелинейных уравнений (метод простой итерации, метод хорд(метод Биссекции), метод Ньютона), исследована скорость сходимости итерационных процедур, составлена программа численного решения нелинейных уравнений методами бисекции/хорд, Ньютона, проверена правильность работы программы на тестовых примерах, численно решено нелинейное уравнение заданного варианта, сравнены количества итераций, необходимых для достижения заданной точности вычисления разными методами.

Оптимальным способом численного решения нелинейных уравнений является применение метода Ньютона, так скорость сходимости в этом методе почти всегда квадратичная, но иногда не квадратичная.

В ходе работы были исполнены 3 функций, имеющих несколько корней на заданном промежутке $(-10, 10)$, и в ходе решения результат был проверен с помощью онлайн калькулятора, что дает знать о том, что реализованные методы успешно справляются с решением нелинейных уравнений.

