**Московский государственный технический**

**университет им. Н. Э. Баумана.**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Основы программирования»

Отчет по лабораторной работе № 4

«Нахождение корней нелинейного уравнения»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-12Б |  | преподаватель каф. ИУ5 |
| Поляков Леонид |  | Правдина А. Н. |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2023 г.

**Постановка задачи**

## Цель работы

* программирование численных методов решения нелинейных уравнений;
* сравнительный анализ методов простой итерации, половинного деления и метода Ньютона.

## Задание

1. Найти корень уравнения x - \cos(x) = 0 *простой итерацией* (метод №1), *половинным делением* (метод №2) и *методом Ньютона* (метод №3) с погрешностью eps < 0.000001 и для каждого из трех методов определить количество шагов алгоритма.
2. Выполнить п.1 для eps < 0.00000001.
3. Выполнить п.1 для уравнения x - k*\cos(x) = 0 для k = 5 и k = 10 и объяснить результаты.

Результат представить в виде таблицы (без рамок), которая содержит три столбца № метода, x и N, где N - количество итераций.

**Разработка алгоритма**

В данной лабораторной работе методы для нахождения корней нелинейных уравнений оформлены в виде функций (

double simple\_iterations(double k, func\_ref g, double eps, double approximation)

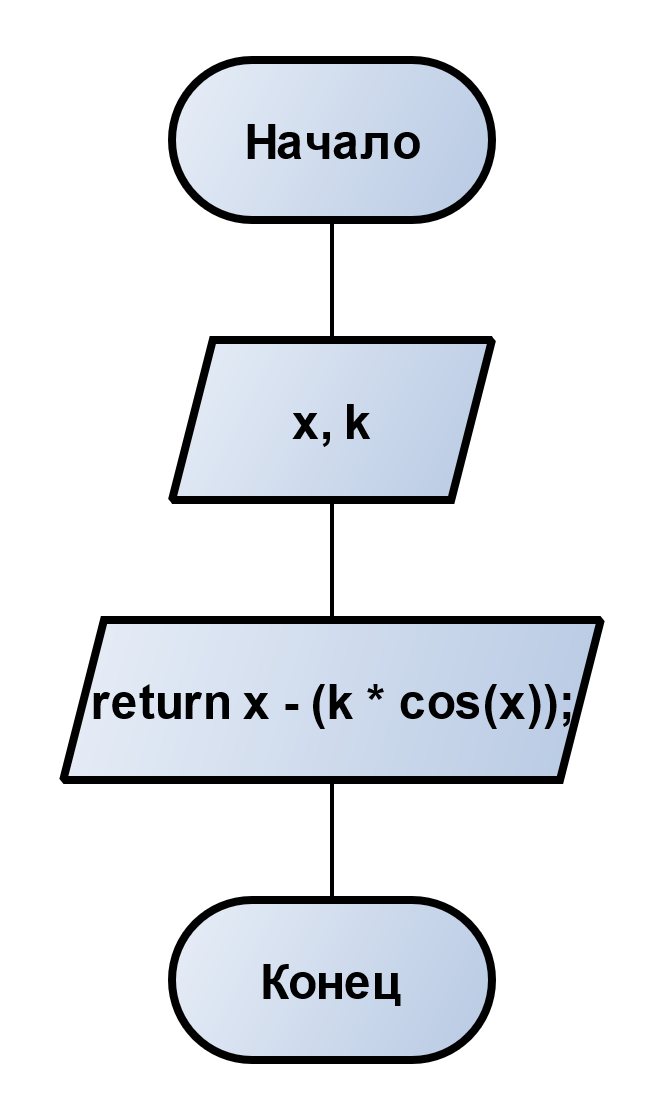
double half\_division(double k, func\_ref f, double eps, double approximation)

double newton\_method(double k, func\_ref f, func\_ref derivative, double eps, double approximation)), принимающих на вход коэффициент k; ссылку на функцию, принимающую в качестве параметров коэффициент k и значение x, и возвращающую значение функции, для которой ведётся поиск корней с данными аргументами (double equation(double x, double k); double transformed\_function(double x, double k); double derivative(double x, double k));

заданную точность(eps), с которой надо посчитать корень и начальное приближение для корня(approximation). Резудьтат на экран выводит функция

void solution(double k, double eps, double approximation), принимающая на вход такие параметры как коэффициент k, фигурирующий в уравнении, точность, с которой требуется посчитать корень и начальное приближение корня. Эта функция последовательно вызывает функции, отвечающие за численное вычисления корня уравнения требуемыми методами.

**Функция, считающая исходную функцию** double equation(double x, double k)



x – переменная функции, у которой ищется корень

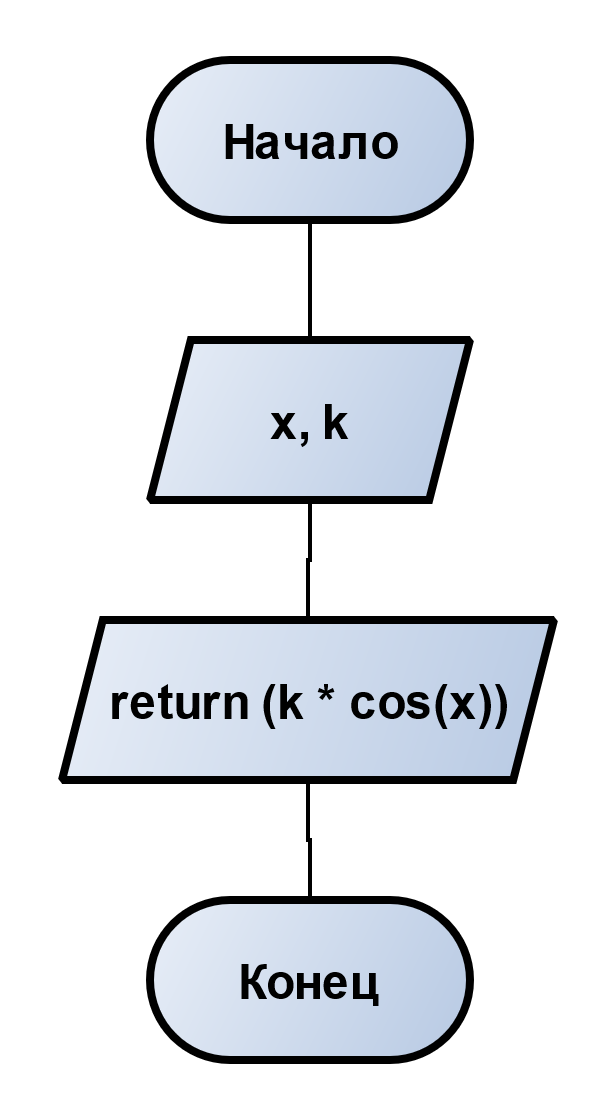
k – коэффициент исходного уравнения

Функция возвращает значение исходной функции

с этими аргументами

**Функция, считающая функцию, приведённую к виду**  x = \varphi(x) \quad (2) 

double transformed\_function(double x, double k)



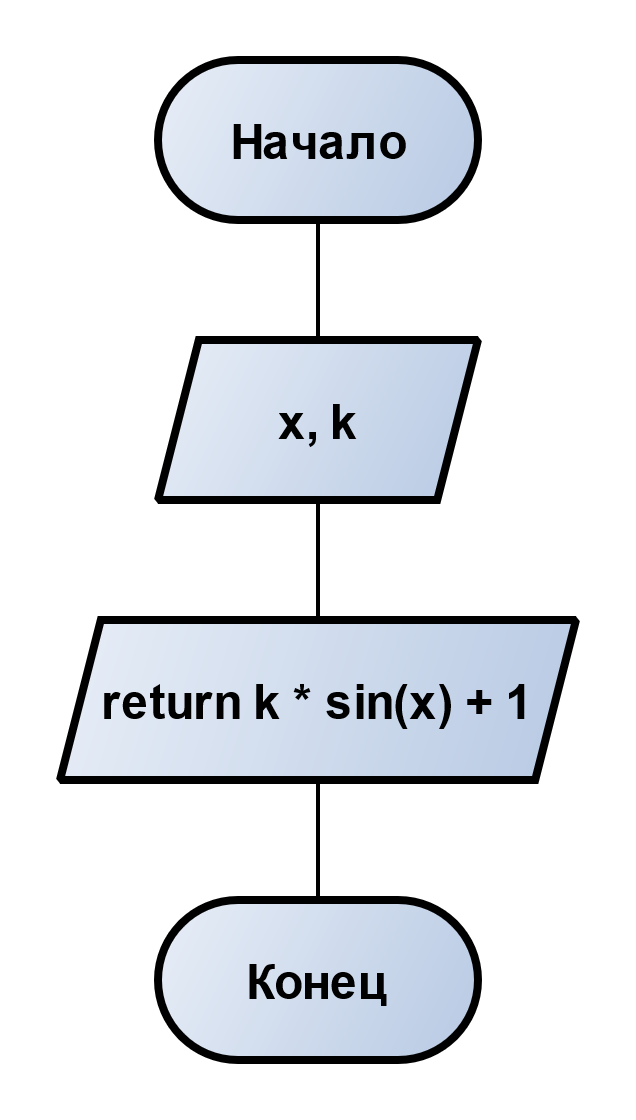
x – переменная функции, у которой ищется корень

k – коэффициент исходного уравнения

Функция возвращает значение функции, преобразованной для метода простых итераций

**Функция, считающая производную исходной функции**

double derivative(double x, double k)

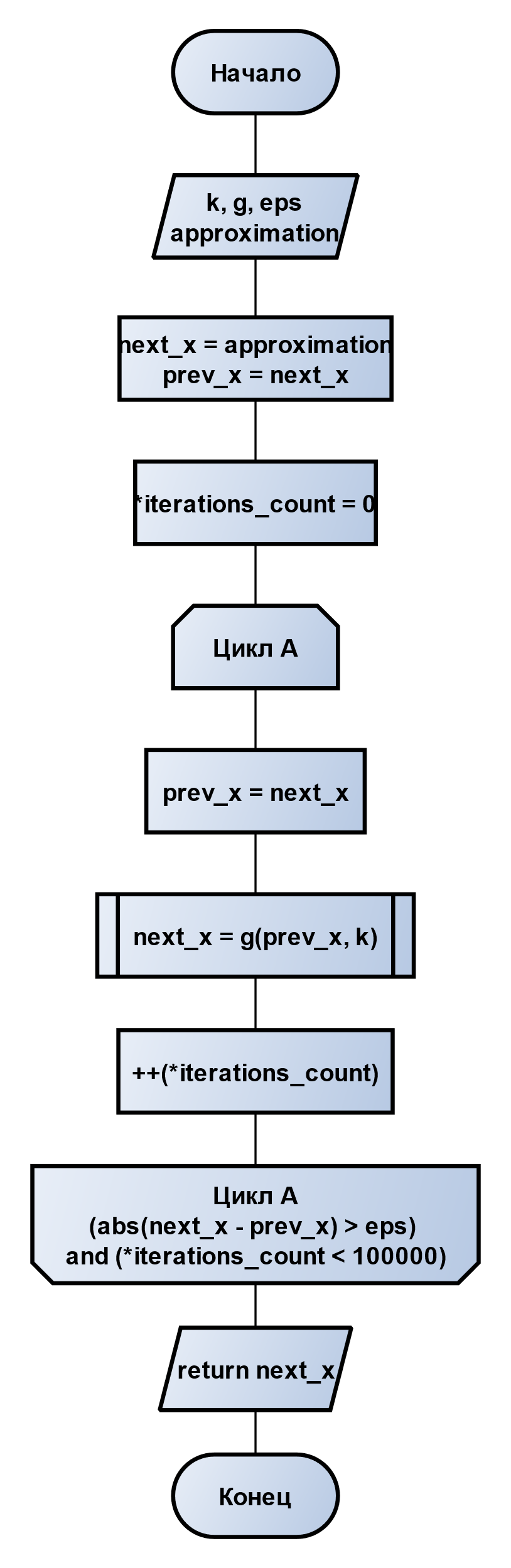
****

x – переменная функции, у которой ищется корень

k – коэффициент исходного уравнения

Функция возвращает значение производной функции, у которой вычисляется корень

**Метод простых итераций**

double simple\_iterations(double k, func\_ref g, double eps, double approximation)

k – коэффициент исходного уравнения

g – ссылка на приведённую функцию вида x = g(x)

eps – требуемая точность для вычисления корня

approximation – начальное приближение корня

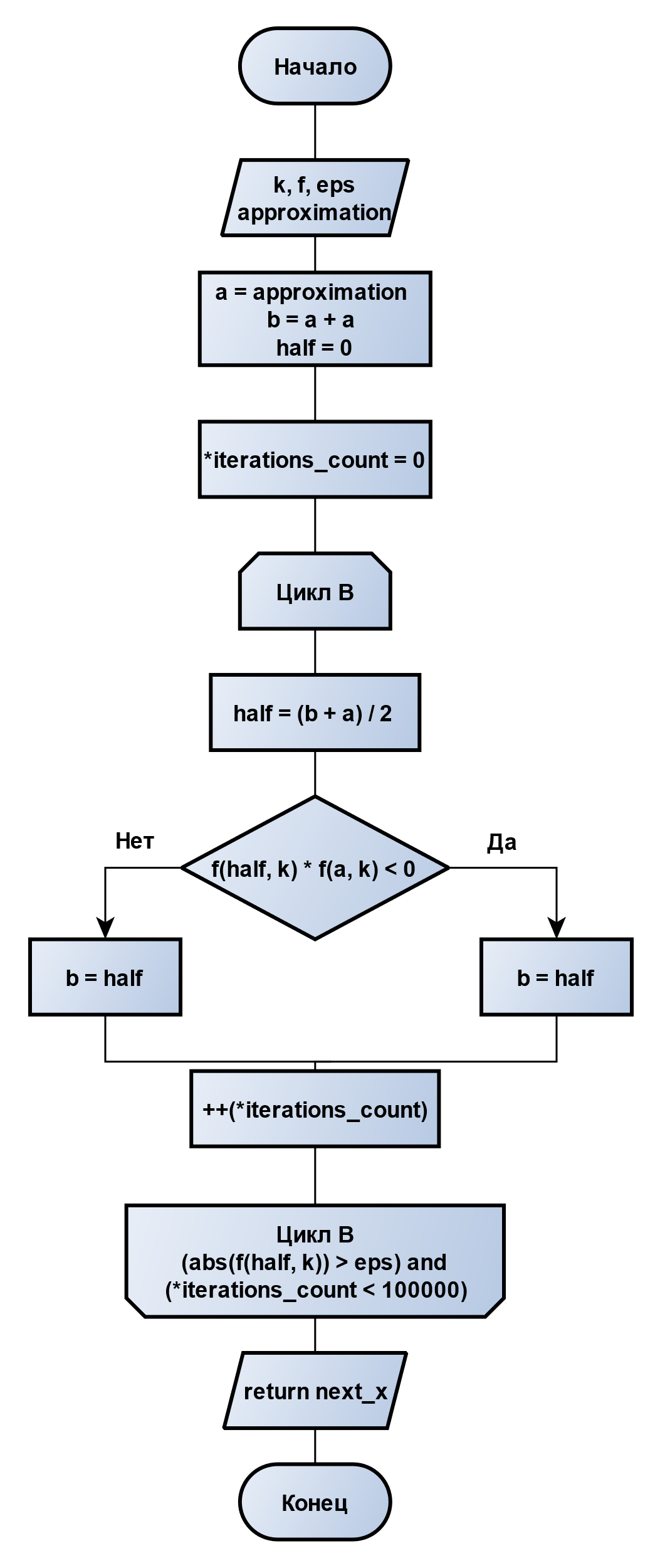
prev\_x, next\_x – предыдущее и нынешнее приближение корня на каждом шаге

iterations\_count – указатель; указывает на переменную, хранящую текущее количество итераций, осуществлённых для вычисления корня

Функция возвращает найденное с заданной точностью приближение корня

Данная функция принимает на вход такие параметры как коэффициент k, фигурирующий в уравнении, указатель на функцию, принимающую два параметра (double k, double x), и возвращающую значение типа double (double (\*g)(double, double)), точность(eps), с которой требуется посчитать корень и начальное приближение корня(approximation). Указатель на переменную iterations\_count отвечаeт за подсчёт общего числа итераций, произведённых для вычисления корня с заданной точностью. При каждом новом вызове функции значение в нём обнуляется. Переменные prev\_x и next\_x хранят в себе на каждой итерации предыдущее и новое значение корня соответственно.

**Метод половинного деления**

double half\_division(double k, func\_ref f, double eps, double approximation)

k – коэффициент исходного уравнения

f – ссылка на функцию, для которой ищется корень

eps – требуемая точность для вычисления корня

approximation – начальное приближение корня

a – нижняя граница интервала на каждой итерации цикла, в котором ищется корень

b – верхняя граница интервала на каждой итерации цикла, в котором ищется корень

half – середина интервала на каждой итерации

цикла

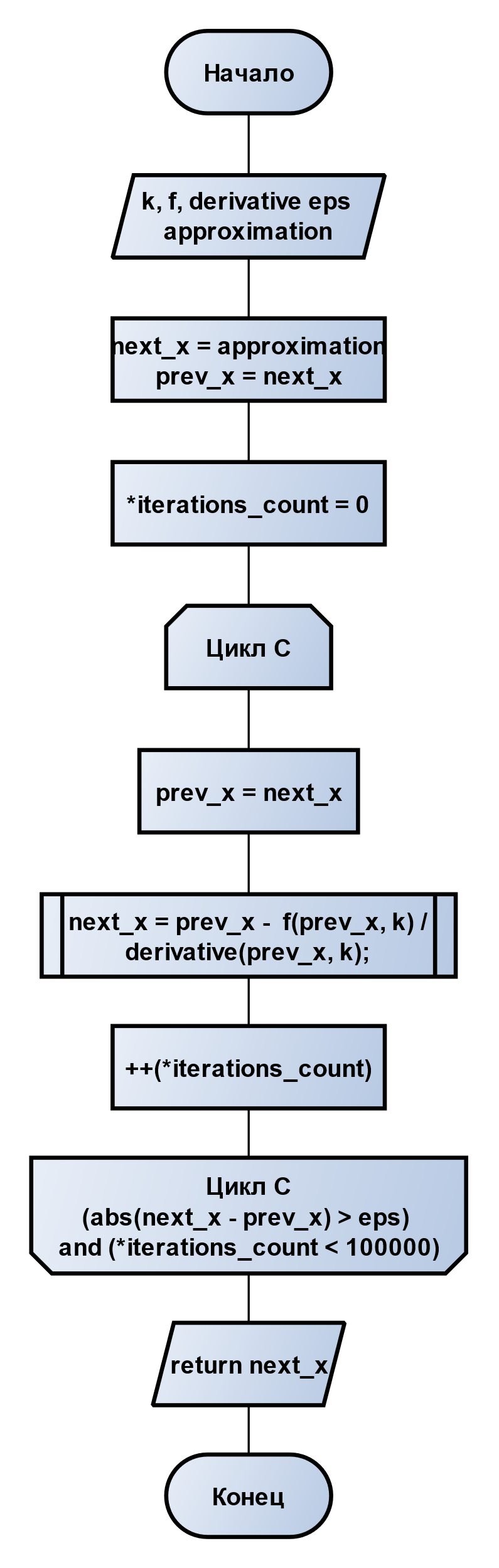
iterations\_count – указатель; указывает на переменную, хранящую текущее количество итераций, осуществлённых для вычисления корня

Функция возвращает найденное с заданной точностью приближение корня

Данная функция принимает на вход такие параметры как коэффициент k, фигурирующий в уравнении, указатель на функцию, принимающую два параметра (double k, double x), и возвращающую значение типа double (double (\*f)(double, double)), точность(eps), с которой требуется посчитать корень и начальное приближение корня(approximation). Указатель на переменную iterations\_count отвечаeт за подсчёт общего числа итераций, произведённых для вычисления корня с заданной точностью. При каждом новом вызове функции значение в нём обнуляется. Переменные a и b хранят в себе на каждой итерации границы промежутка, на котором ищется корень. Переменная half хранит в себе текущую середину отрезка, на котором производится поиск корня.

**Метод Ньютона**

double newton\_method(double k, func\_ref f, func\_ref derivative, double eps, double approximation)



k – коэффициент исходного уравнения

f – ссылка на функцию, для которой ищется корень

derivative – ссылка на производную функции, для которой ищется корень

eps – требуемая точность для вычисления корня

approximation – начальное приближение корня

prev\_x, next\_x – предыдущее и нынешнее приближение корня на каждом шаге

iterations\_count – указатель; указывает на переменную, хранящую текущее количество итераций, осуществлённых для вычисления корня

Функция возвращает найденное с заданной точностью приближение корня

Данная функция принимает на вход такие параметры как коэффициент k, фигурирующий в уравнении, два указателя на функции, принимающие два параметра (double k, double x), и возвращающие значение типа double (double (\*g)(double, double), double (\*derivative)(double, double)), точность(eps), с которой требуется посчитать корень и начальное приближение корня(approximation). Указатель на переменную iterations\_count отвечаeт за подсчёт общего числа итераций, произведённых для вычисления корня с заданной точностью. При каждом новом вызове функции значение в нём обнуляется. Переменные prev\_x и next\_x хранят в себе на каждой итерации предыдущее и новое значение корня соответственно.

**Текст программы**

**constants.h**Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

**Function.h**

**Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание**

**Methods.hИзображение выглядит как текст, Шрифт, линия, снимок экрана

Автоматически созданное описание**

**Solution.hИзображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описаниеFunction.cppИзображение выглядит как текст, снимок экрана, программное обеспечение, число

Автоматически созданное описание**

**Methods.cppИзображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание**

**Solution.cppИзображение выглядит как текст, снимок экрана, линия, Шрифт

Автоматически созданное описание**

**main.cppИзображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание**

**Анализ результатовИзображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание**

**Вывод**

Нетрудно заметить, что во всех четырёх случаях метод Ньютона показал себя лучше двух других методов. Во всех случаях этому методу хватило 4 итерации, чтобы вычислить корень уравнения с заданной точностью. Сравнивая же два оставшихся метода (метод простых итераций и метод половинного деления) можно убедиться, что для данного уравнения и начальных приближений, метод половинного деления считает корни эффективнее считает корни. Это заключается и в количестве необходимых итераций, для достижения требуемой точности, и в том, что для уравнений вида x - k*\cos(x) = 0 метод простых итераций расходится (это видно по количеству итераций (100000) в двух последних таблицах), несмотря на довольно точное начальное приближение, что является явным недостатком вышеупомянутого метода.

**Использованные источники**

* **Веб-сайт кафедры ИУ5, курс «основы программирования»**

[**https://cpp1.wiki.iu5edu.ru**](https://cpp1.wiki.iu5edu.ru)

* **Веб-сайт кафедры ИУ5 «автоматизированные системы обработки информации и управления»**

**https://e-learning.bmstu.ru/iu5/mod/folder/view.php?id=277**