Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Пермский национальный исследовательский**

**политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**Лабораторная работа №1**

**“Методы решения нелинейных уравнений”**

Выполнил:

студент группы ИВТ-24-2б

Ашарапов Р. И.

Проверил:

Доц. каф. ИТАС

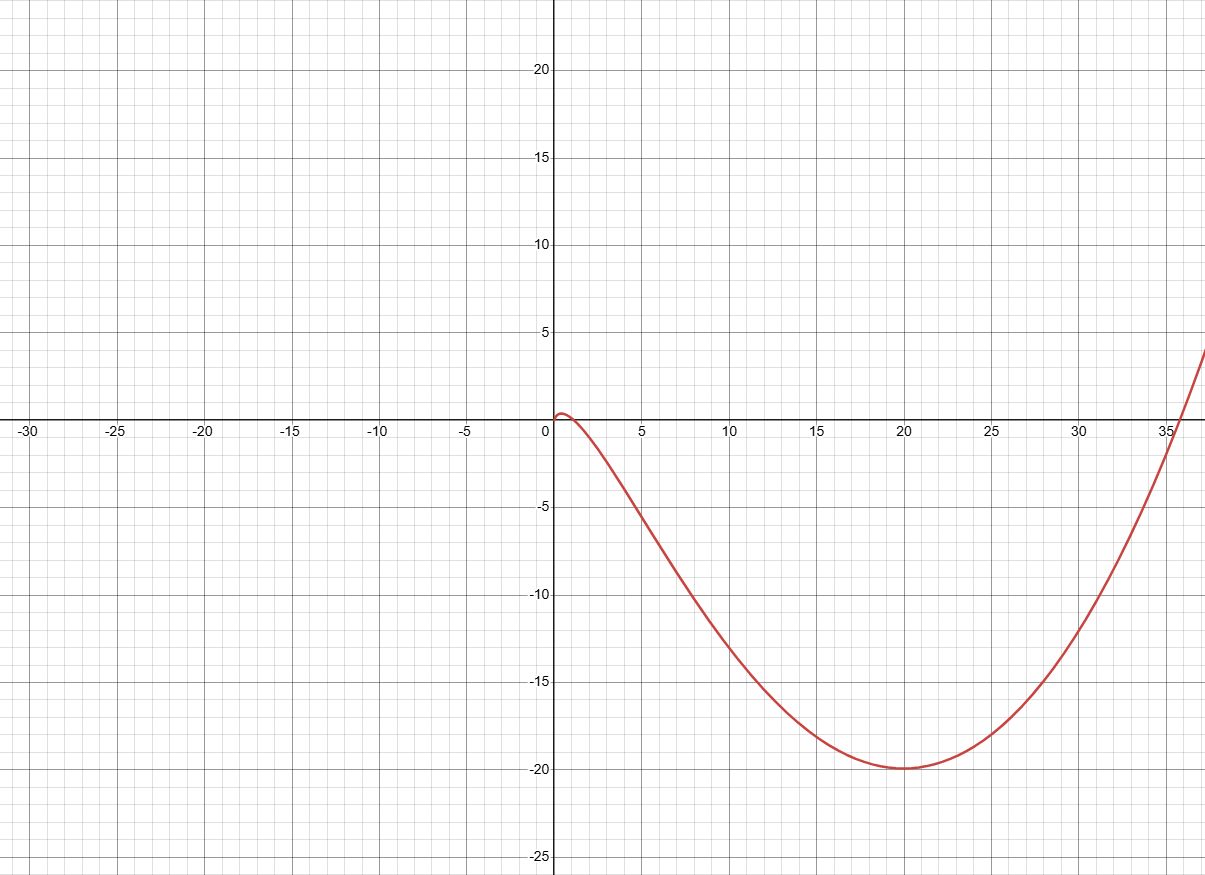
Полякова О.А.

Пермь, 2024

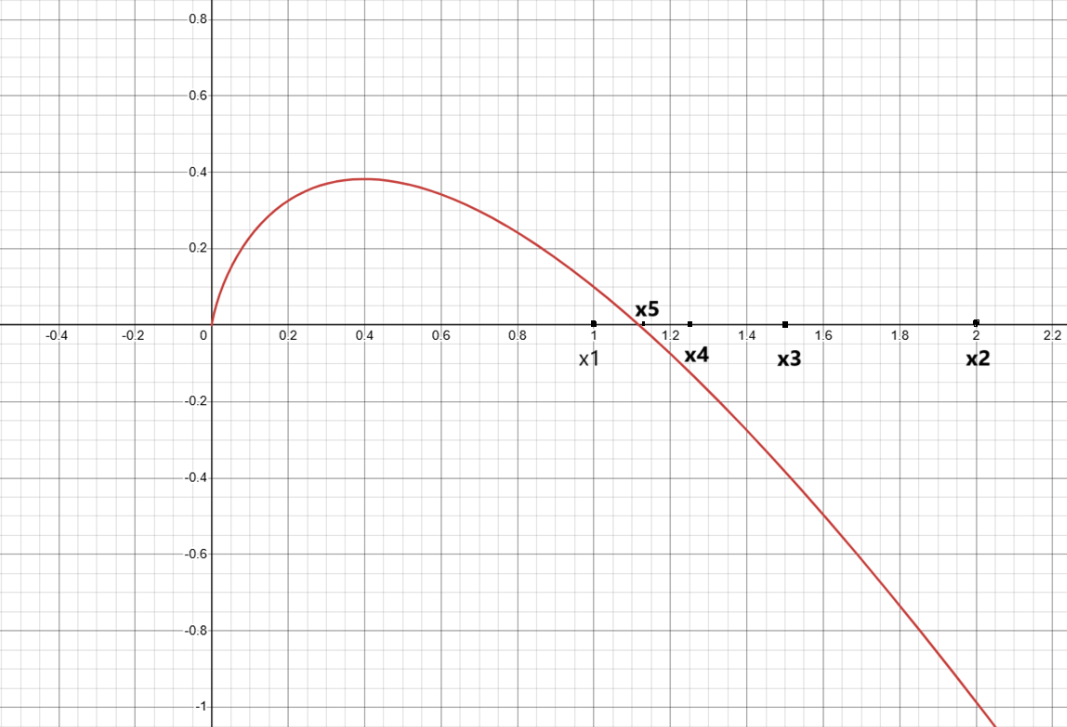
Постановка задачи

Решить нелинейное уравнение 0,1x2 - x ln x = 0 3 способами: методом Ньютона, методом итерации, методом половинного деления на интервале [1,2]. Точность вычисления вводит пользователь.

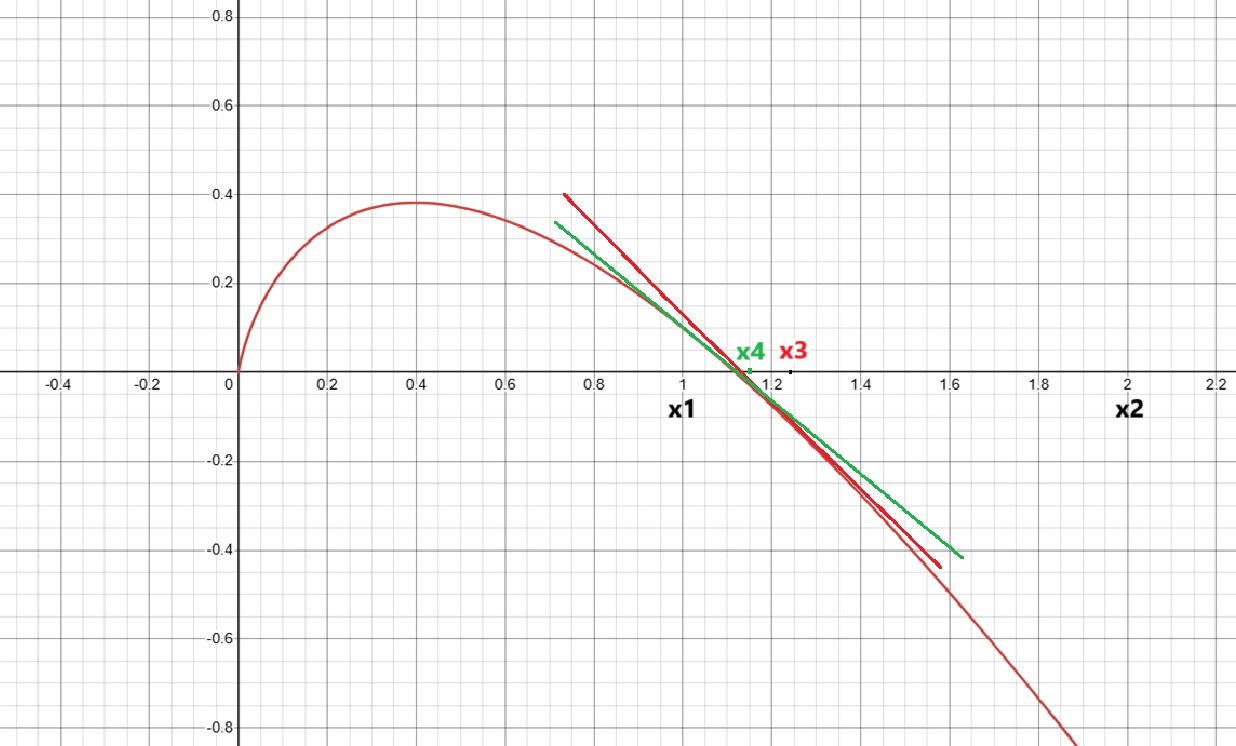
Геометрическая интерпретация



1. Метод половинного деления



2. Метод Ньютона



Анализ задачи

Метод половинного деления

1. Функция f(x) = 0,1x2 - x ln x

2. Функция монотонна и непрерывна

3. Проверяем наличие корня на интервале ab, он существует, поэтому функция f(x) пересекает ось ох

4. Находим среднюю точку интервала *x*1:

5. Если , обновляем интервал: , иначе:

6. Каждый раз после нахождения нового корня уравнения сравниваем abs(x1-x2) с эпсилон, как только станет меньше мы получим самое точное значение х

Метод Ньютона

1. Функция f(x) = 0,1x2 - x ln x

2. Производные:

=0.2\*x-lnx-1

=0.2-(1/x)

3. Берем за значение а приблизительное значение корня, f(a)\*f’’(a)>0,

4. Функция монотонна и непрерывна

5. По формуле находим х, до тех пор пока abs(x1-x2) не станет меньше эпсилон

Метод итерации

1. Функция f(x) = 0,1x2 - x ln x

2. Выражаем х :

f(x)=e^(0.1\*x)

3. Находим производную

f’(x)=(e^(x/10)/10)

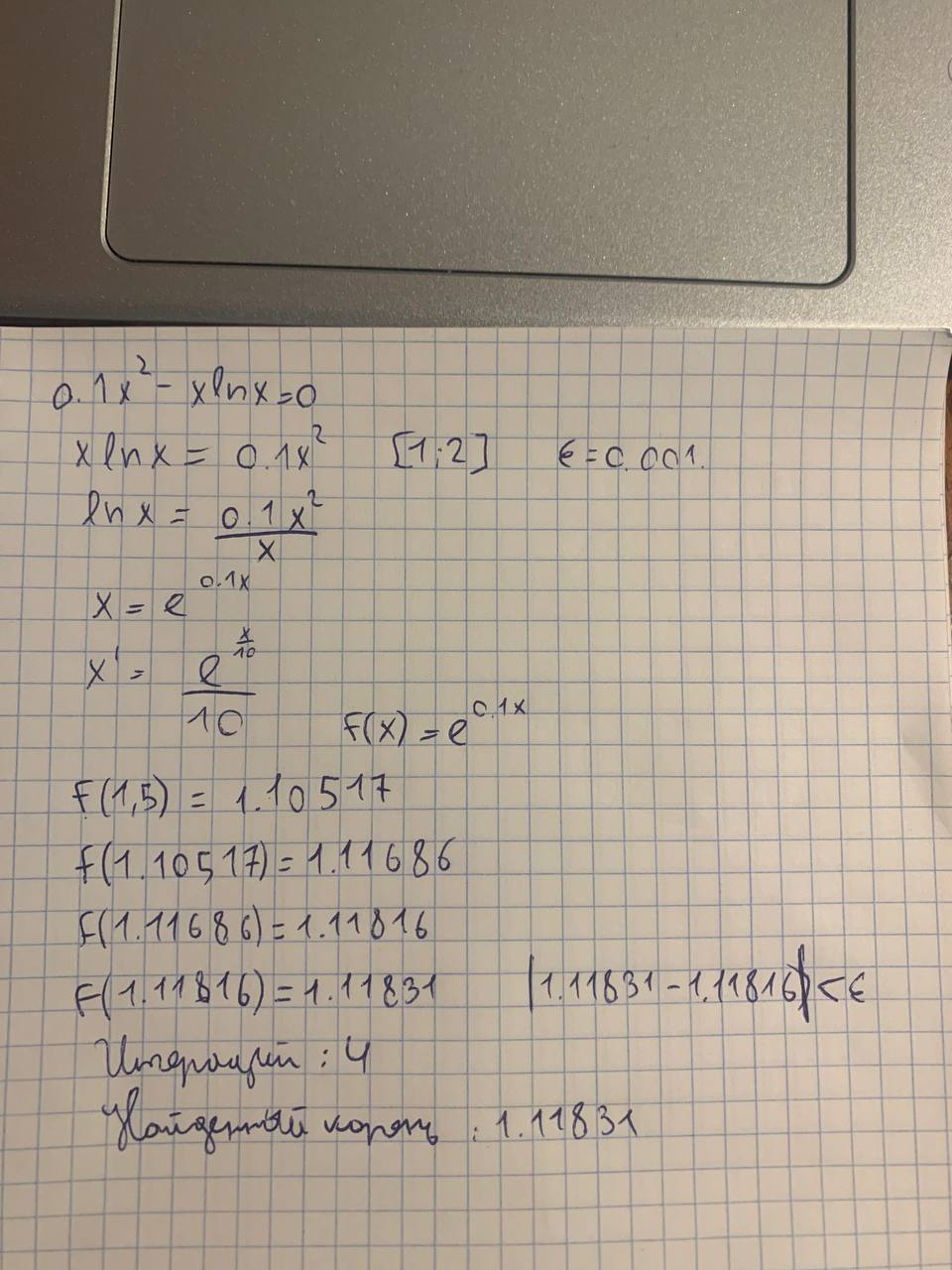
4. Подставляем значение в производную, результат должен быть меньше единицы:

f’(1)=0.11<1, следовательно с этим уравнением можно работать

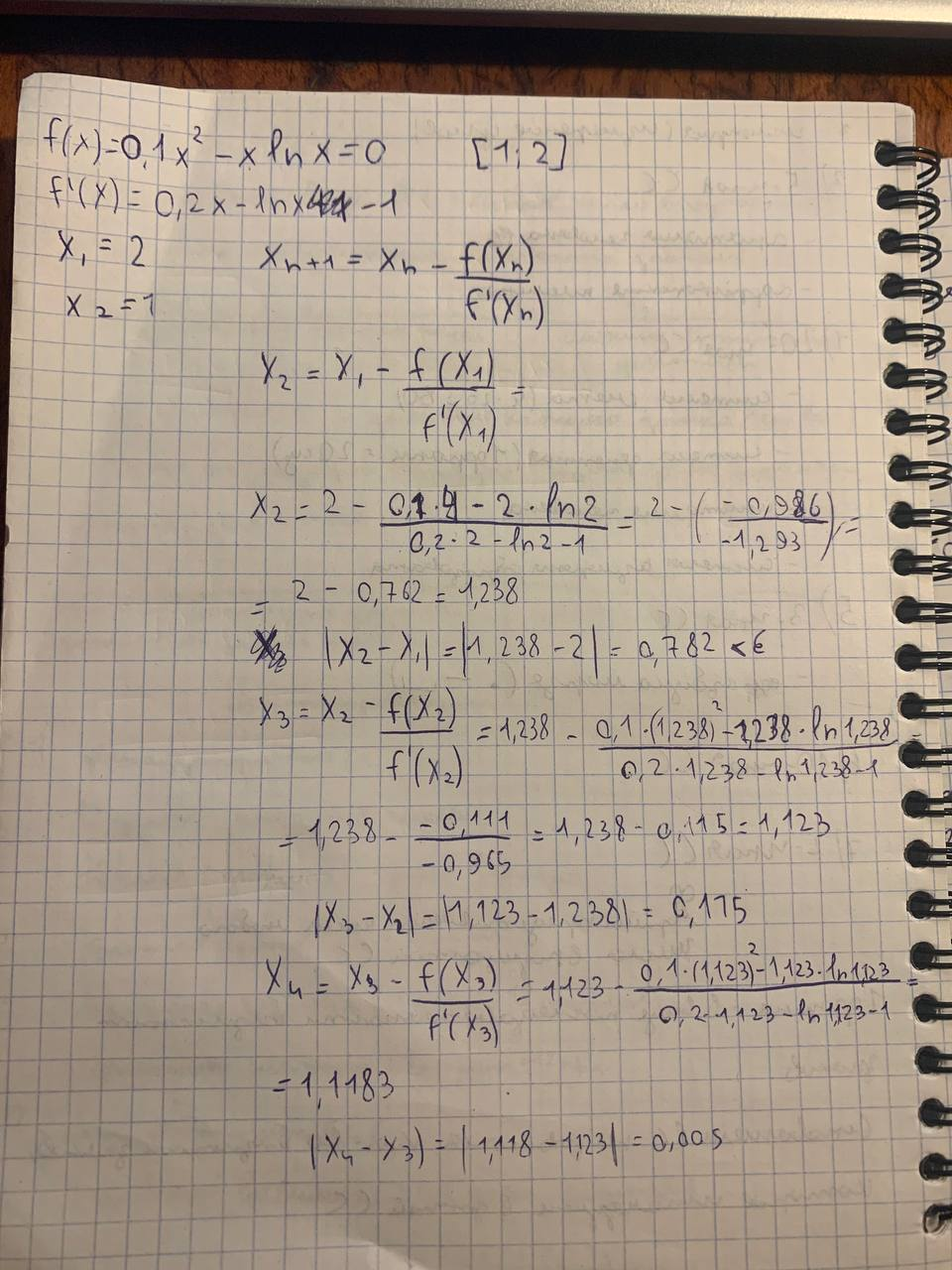
5. Находим х, подставляя полученный корень в f(x)=e^(0.1\*x), продолжаем итерации, пока разница между значениями по модулю не станет меньше заданной эпсилон

Поитерационное решение.

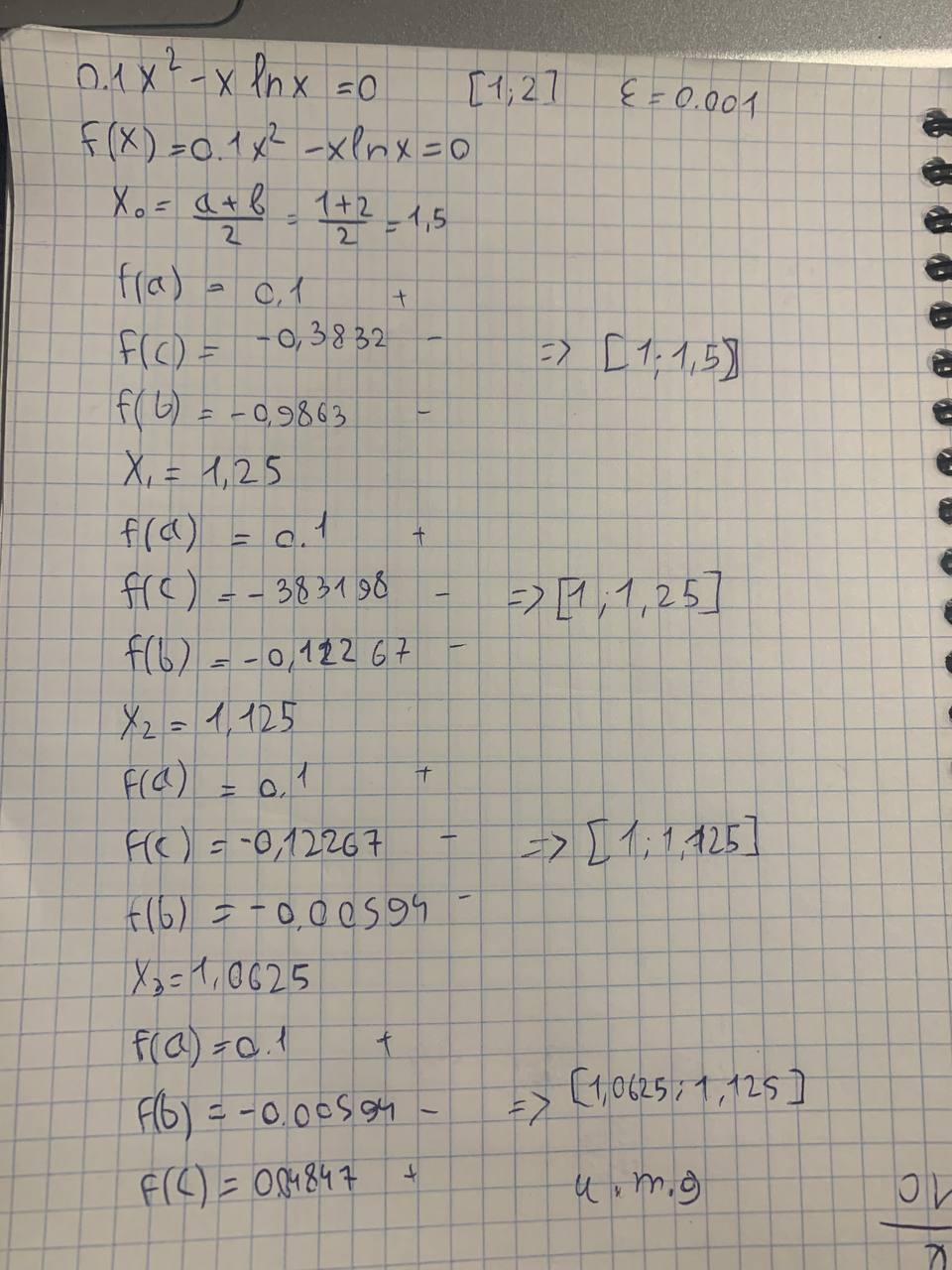
1. Метод итерации



2. Метод Ньютона

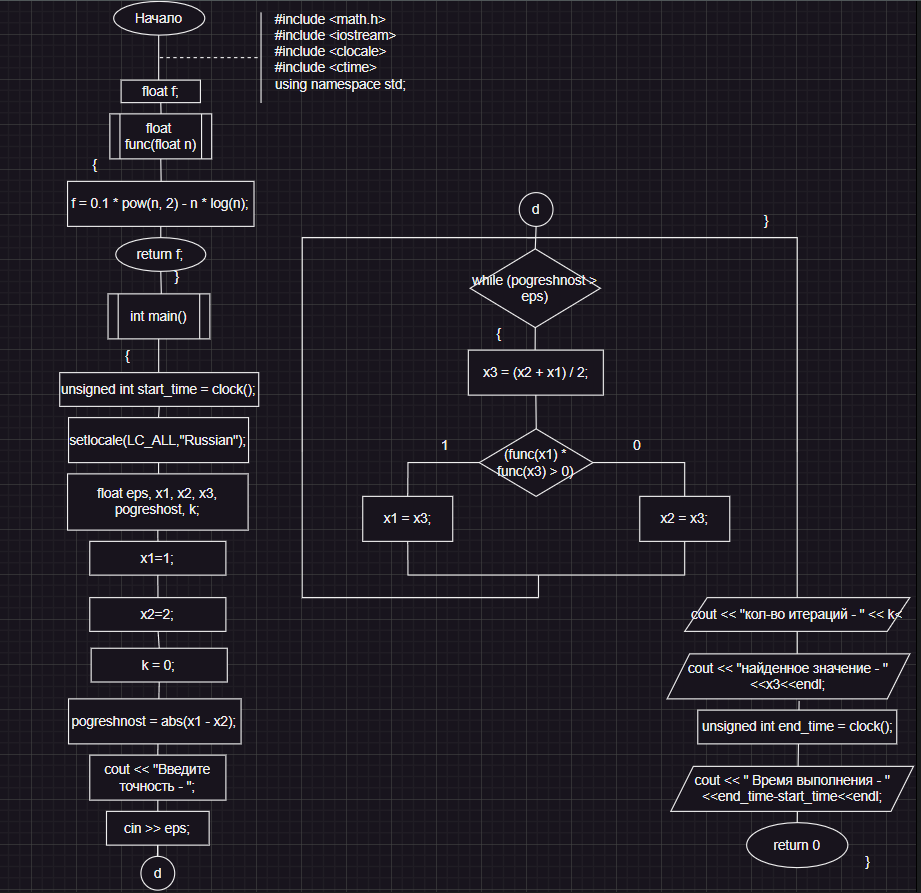


3. Метод половинного деления

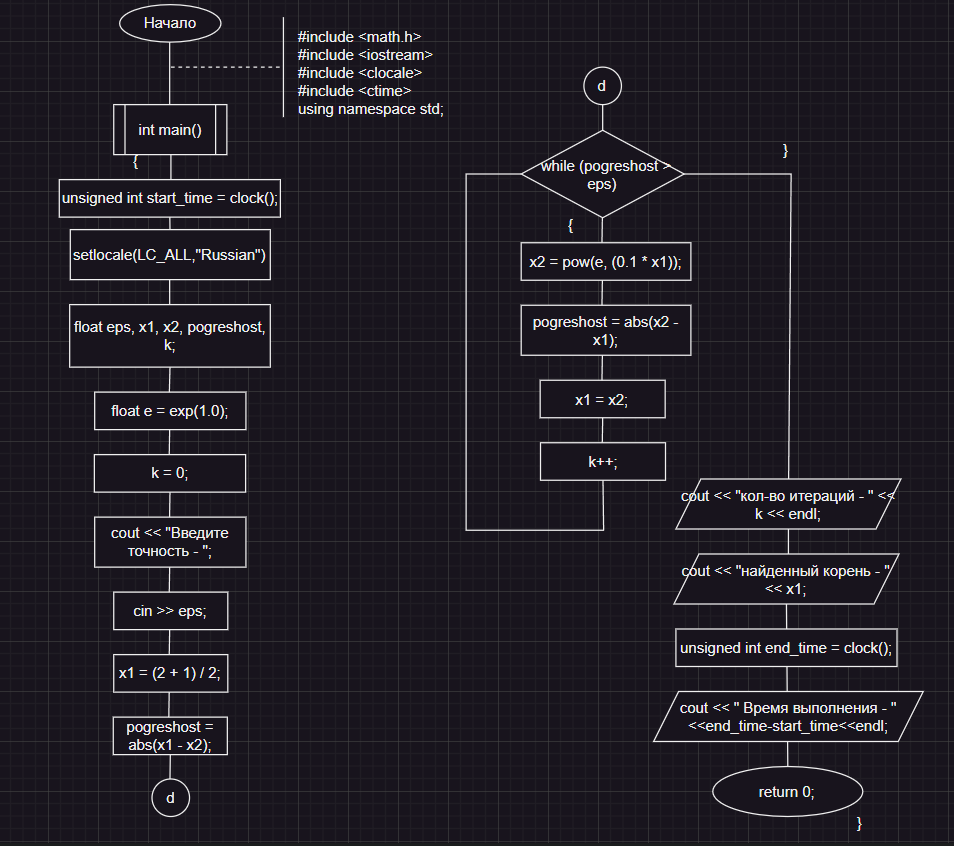


Блок-схемы

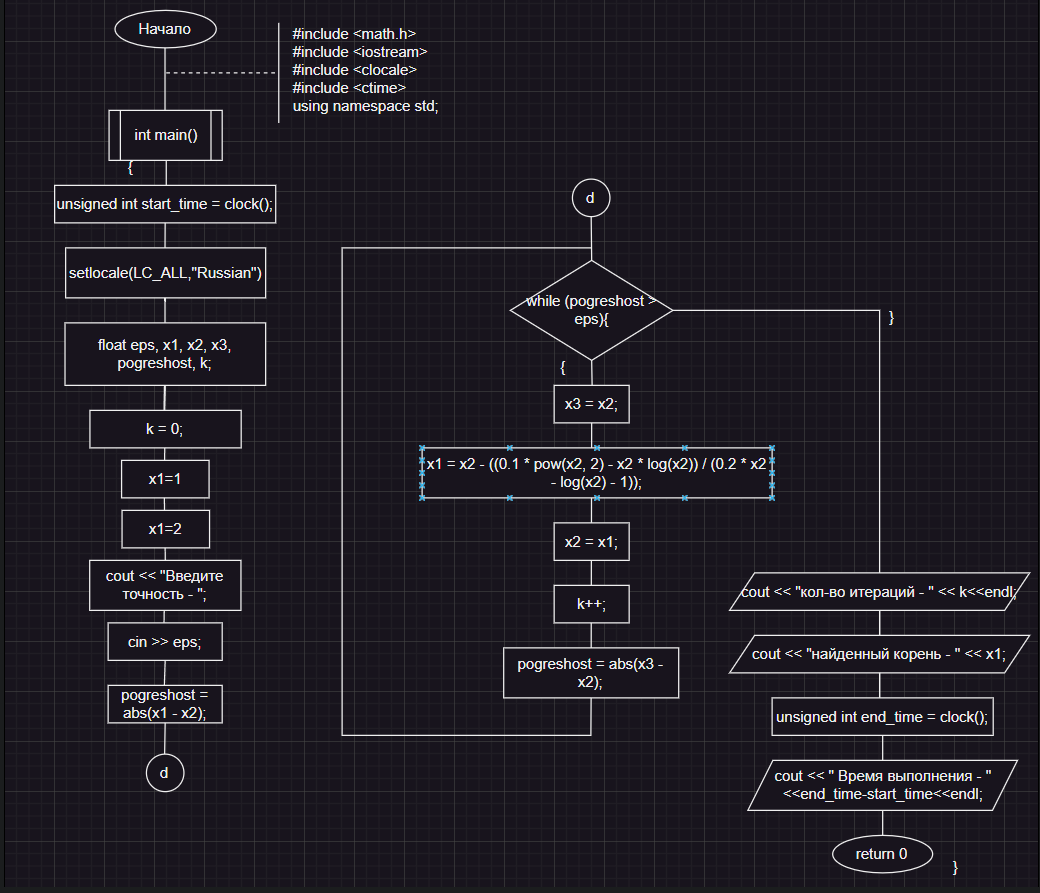
1.Метод половинного деления



2. Метод итерации



3.Метод Ньютона



Код программы c++

1. Метод половинного деления.

#include<iostream>

#include<clocale>

#include<math.h>

#include<ctime>

using namespace std;

float f;

float func(float n) {

f = 0.1 \* pow(n, 2) - n \* log(n);

return f;

}

int main()

{

unsigned int start\_time = clock();

setlocale(LC\_ALL, "RU");

float x1, x2, x3, eps, pogreshnost,k;

x1 = 1;

x2 = 2;

k = 0;

pogreshnost = abs(x1 - x2);

cout << "Введите точность - ";

cin >> eps;

while (pogreshnost > eps){

x3 = (x2 + x1) / 2;

if (func(x1) \* func(x3) > 0){

x1 = x3;

}

else {

x2 = x3;

}

pogreshnost = abs(x1 - x2);

k++;

}

cout << "кол-во итераций - " << k<<endl;

cout << "найденное значение - "<<x3<<endl;

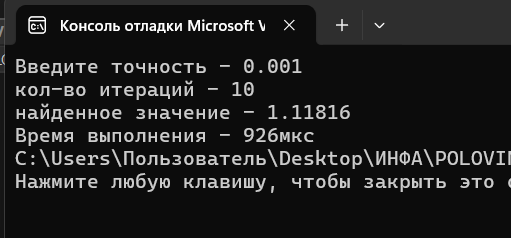
unsigned int end\_time = clock();

cout << "Время выполнения - " << end\_time - start\_time << "мкс";

return 0;

}

Вывод в консоли:



2. Метод итерации

#include<iostream>

#include<clocale>

#include<math.h>

#include<ctime>

using namespace std;

int main()

{

unsigned int start\_time = clock();

setlocale(LC\_ALL, "RU");

float eps, x1, x2, pogreshost, k;

float e = exp(1.0);

k = 0;

cout << "Введите точность - ";

cin >> eps;

x2=2;

x1 = (2 + 1) / 2;

pogreshost = abs(x1 - x2);

while (pogreshost > eps) {

x2 = pow(e, (0.1 \* x1));

pogreshost = abs(x2 - x1);

x1 = x2;

k++;

}

cout << "кол-во итераций - " << k << endl;

cout << "найденный корень - " << x1;

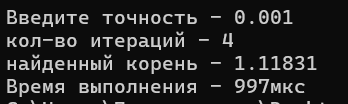
unsigned int end\_time = clock();

cout << "Время выполнения - " << end\_time - start\_time << "мкс";

return 0;

}

Вывод в консоль:



3. Метод Ньютона

#include<iostream>

#include<clocale>

#include<math.h>

#include<ctime>

using namespace std;

int main()

{

unsigned int start\_time = clock();

setlocale(LC\_ALL, "RU");

double eps, x1, x2, x3, pogreshost, k;

k = 0;

x1 = 1;

x2 = 2;

cout << "Введите точность - ";

cin >> eps;

pogreshost = abs(x1 - x2);

while (pogreshost > eps){

x3 = x2;

x1 = x2 - ((0.1 \* pow(x2, 2) - x2 \* log(x2)) / (0.2 \* x2 - log(x2) - 1));

x2 = x1;

k++;

pogreshost = abs(x3 - x2);

}

cout << "кол-во итераций - " << k<<endl;

cout << "найденный корень - " << x1<<endl;

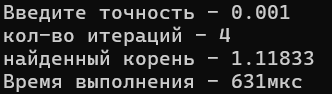
unsigned int end\_time = clock();

cout <<"Время выполнения - "<< end\_time - start\_time <<"мкс";

return 0;

}

Вывод в консоль:



Ссылка на GitHub: https://github.com/wenaly