Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

**«Пермский национальный исследовательский политехнический университет»**

«Кафедра Информационные технологии и автоматизированные системы»

**Лабораторная работа №7**

**Решение не линейных уравнений численными методами приближенного нахождения корней**

Выполнил студент гр. ИВТ-24-2б

Пономарев Кирилл Денисович

Проверил:

Доц. каф. ИТАС

Ольга Андреевна Полякова

г. Пермь 2024

Вариант 4

Дано:

0,25x3 + x - 1,2502 = 0

Отрезок, содержащий корень: [0;2]

Точность 0.00001

Анализ:

Метод Ньютона

1. **Определение функции** f(x) = 0,25x3 + x — 1,2502
2. **Вычисление производных**

* f`(x) = 0,25\*3\*x2 +1
* f``(x) = 0,25\*6\*x

1. **Инициализация**:
2. **Итерации:** продолжаем итерации пока f(x) > 0,00001

Метод половинного деления

1. **Определение функции** f(x) = 0,25x3 + x — 1,2502
2. **Инициализация**: устанавливаем начальные значения интервала 0 и *2*
3. **Итерации:**

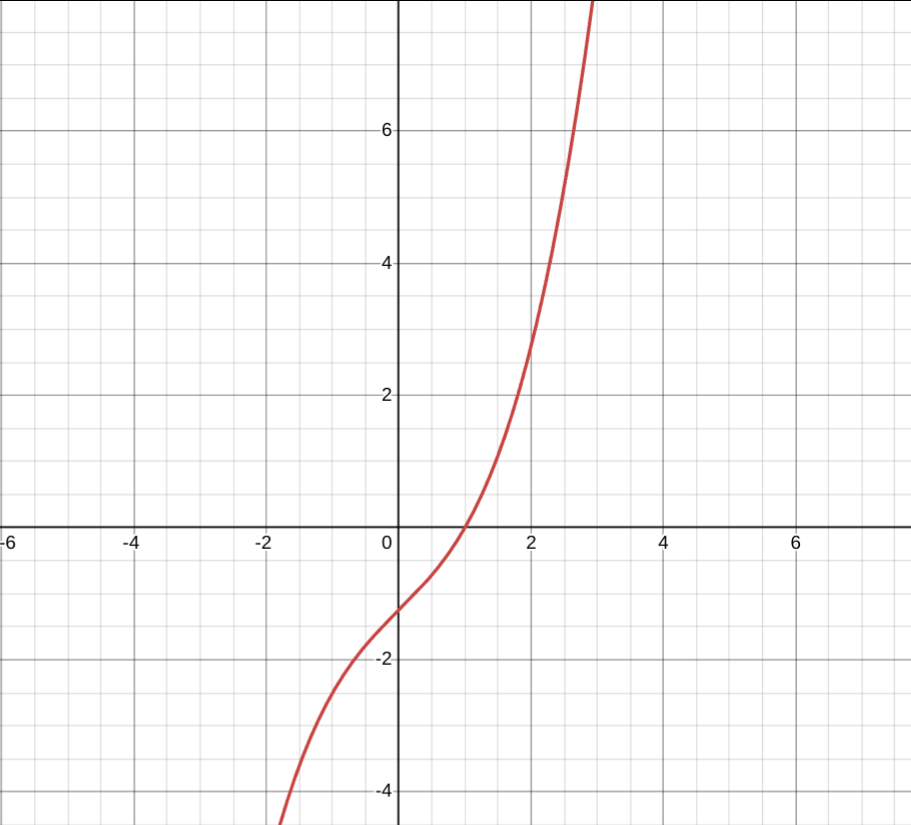
* Находим среднюю точку интервала *x*:
* Если , обновляем интервал:
* Иначе:
* Продолжаем итерации, пока длина интервала больше 0.00001

Метод итераций

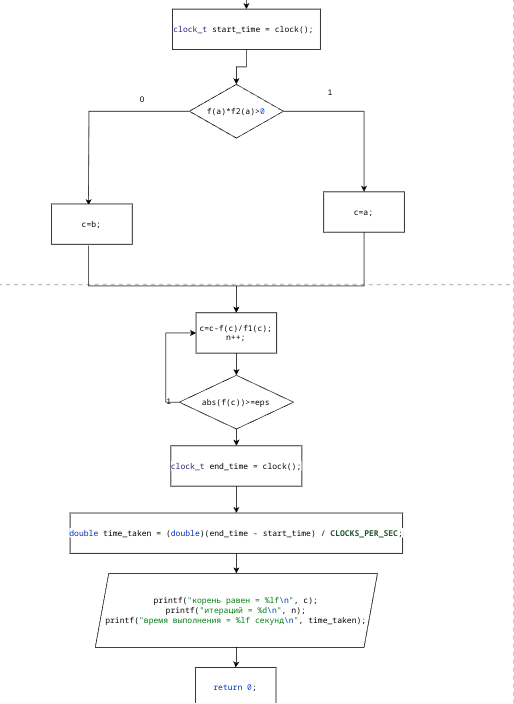
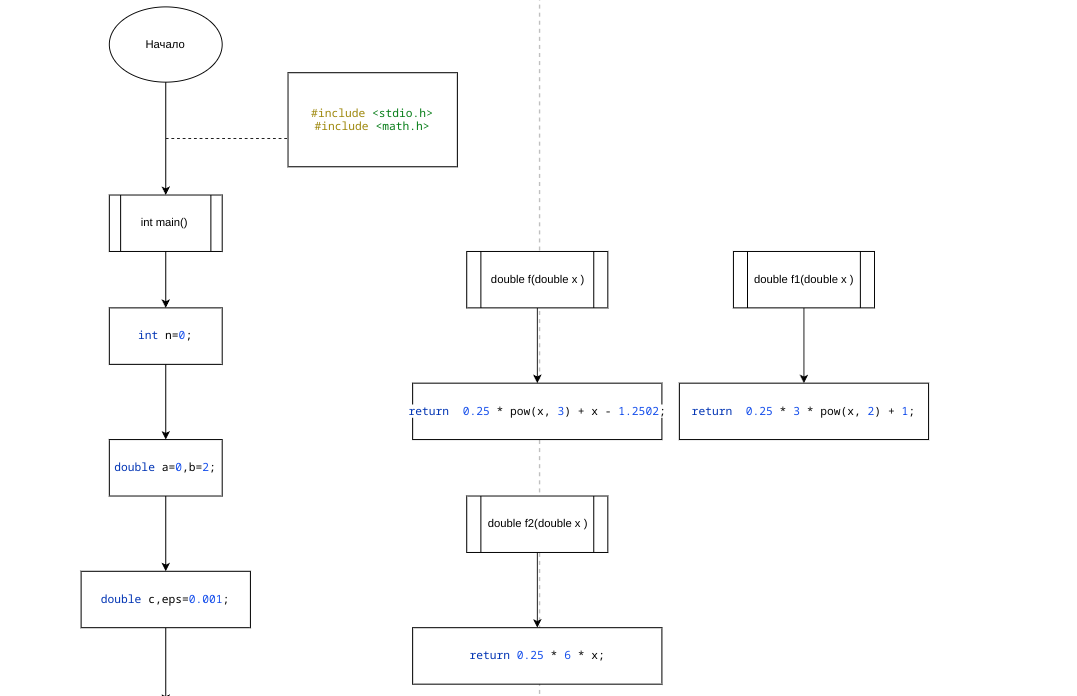
1. **Определение функции:** f(x) = 0,25x3 + x — 1,2502
2. **Переписывание функции для итерационного метода**: g(x) = 1.2502 — 0.25 \*x3
3. **Инициализация**: устанавливаем начальное значение x0
4. **Итерации**:

* Находим новое значение *x1*: x n+1= g(xn)
* Продолжаем итерации, пока

Геометрическая интерпретация

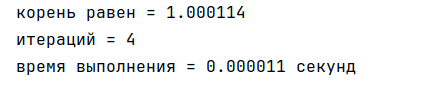


Метод Ньютона

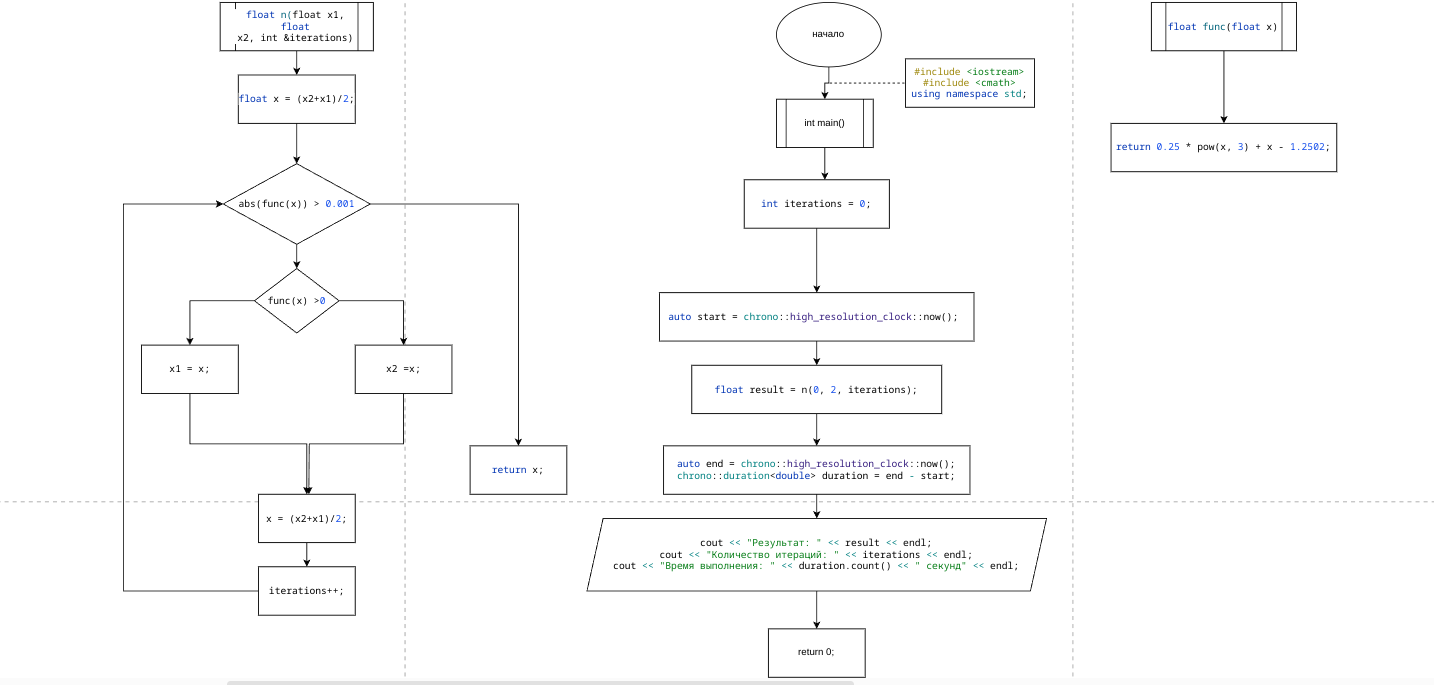


Код

#include <stdio.h>  
#include <math.h>  
#include <time.h>  
  
double f(double x)  
{  
 return 0.25 \* pow(x, 3) + x - 1.2502;  
}  
  
double f1(double x)  
{  
 return 0.25 \* 3 \* pow(x, 2) + 1;  
}  
  
double f2(double x)  
{  
 return 0.25 \* 6 \* x;  
}  
  
int main()  
{  
 int n = 0;  
 double a = 0, b = 2;  
 double c, eps = 0.00001;  
  
 clock\_t start\_time = clock();  
  
 if (f(a) \* f2(a) > 0)  
 c = a;  
 else  
 c = b;  
  
 do  
 {  
 c = c - f(c) / f1(c);  
 n++;  
 }  
 while (abs(f(c)) >= eps);  
  
 clock\_t end\_time = clock();  
  
 double time\_taken = (double)(end\_time - start\_time) / **CLOCKS\_PER\_SEC**;  
  
 printf("корень равен = %lf\n", c);  
 printf("итераций = %d\n", n);  
 printf("время выполнения = %lf секунд\n", time\_taken);  
  
 return 0;  
}

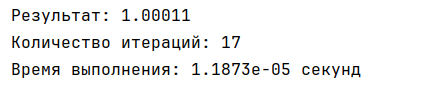


Метод половинного деления

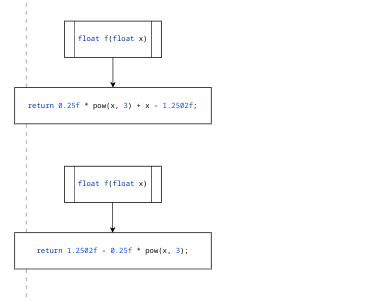
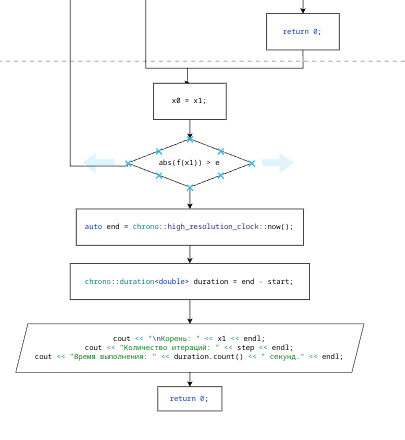
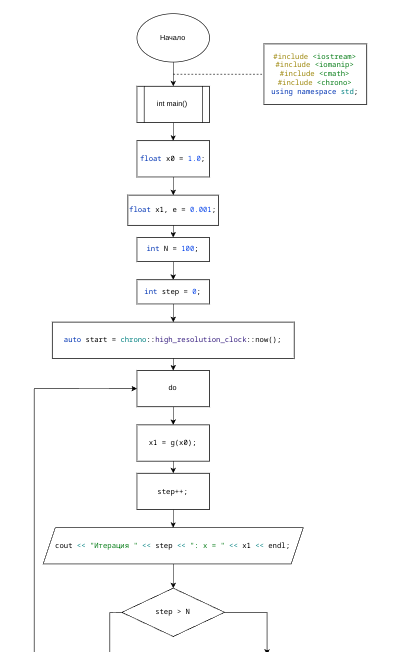


Код

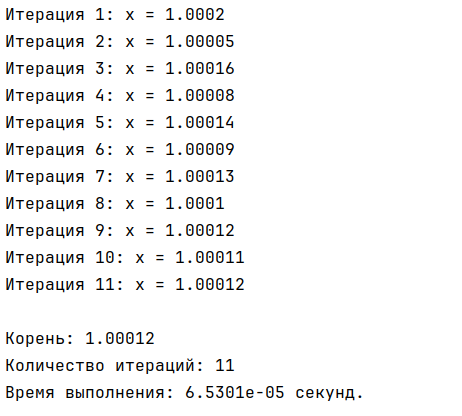
#include <iostream>  
#include <cmath>  
#include <chrono>  
using namespace std;  
  
float func(float x) {  
 return 0.25 \* pow(x, 3) + x - 1.2502;  
}  
  
float n(float x1, float x2, int &iterations) {  
 float x = (x2 + x1) / 2;  
 while (abs(func(x)) > 0.00001) {  
 if (func(x) > 0) x2 = x;  
 else x1 = x;  
 x = (x2 + x1) / 2;  
 iterations++;  
 }  
 return x;  
}  
  
int main() {  
 int iterations = 0;  
 auto start = chrono::high\_resolution\_clock::now();  
  
 float result = n(0, 2, iterations);  
 auto end = chrono::high\_resolution\_clock::now();  
 chrono::duration<double> duration = end - start;  
  
 cout << "Результат: " << result << endl;  
 cout << "Количество итераций: " << iterations << endl;  
 cout << "Время выполнения: " << duration.count() << " секунд" << endl;  
  
 return 0;  
}



Метод итераций



код

#include <iostream>  
#include <iomanip>  
#include <cmath>  
#include <chrono>  
using namespace std;  
  
float f(float x) {  
 return 0.25f \* pow(x, 3) + x - 1.2502f;  
}  
float g(float x) {  
 return 1.2502f - 0.25f \* pow(x, 3);  
}  
  
int main() {  
 float x0 = 1.0;  
 float x1, e = 0.00001;  
 int N = 100;  
 int step = 0;  
  
 auto start = chrono::high\_resolution\_clock::now();  
  
 do {  
 x1 = g(x0);  
 step++;  
  
 cout << "Итерация " << step << ": x = " << x1 << endl;  
 if (step > N) {  
 return 0;  
 }  
 x0 = x1;  
  
 } while (abs(f(x1)) > e);  
  
 auto end = chrono::high\_resolution\_clock::now();  
 chrono::duration<double> duration = end - start;  
  
 cout << "\nКорень: " << x1 << endl;  
 cout << "Количество итераций: " << step << endl;  
 cout << "Время выполнения: " << duration.count() << " секунд." << endl;  
  
 return 0;}