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**

**“** **Вычисление функций с использованием их разложения в степенной ряд”**

Выполнил:

студент группы ИВТ-24-2б

Ашарапов Р. И.

Проверил:

Доц. каф. ИТАС

Полякова О.А.

Пермь, 2024

1. Постановка задачи: Решить уравнение методом половинного деления, используя рекурсию.

Анализ задачи

Метод половинного деления

1. Функция f(x) = 0,1x2 - x ln x

2. Функция монотонна и непрерывна

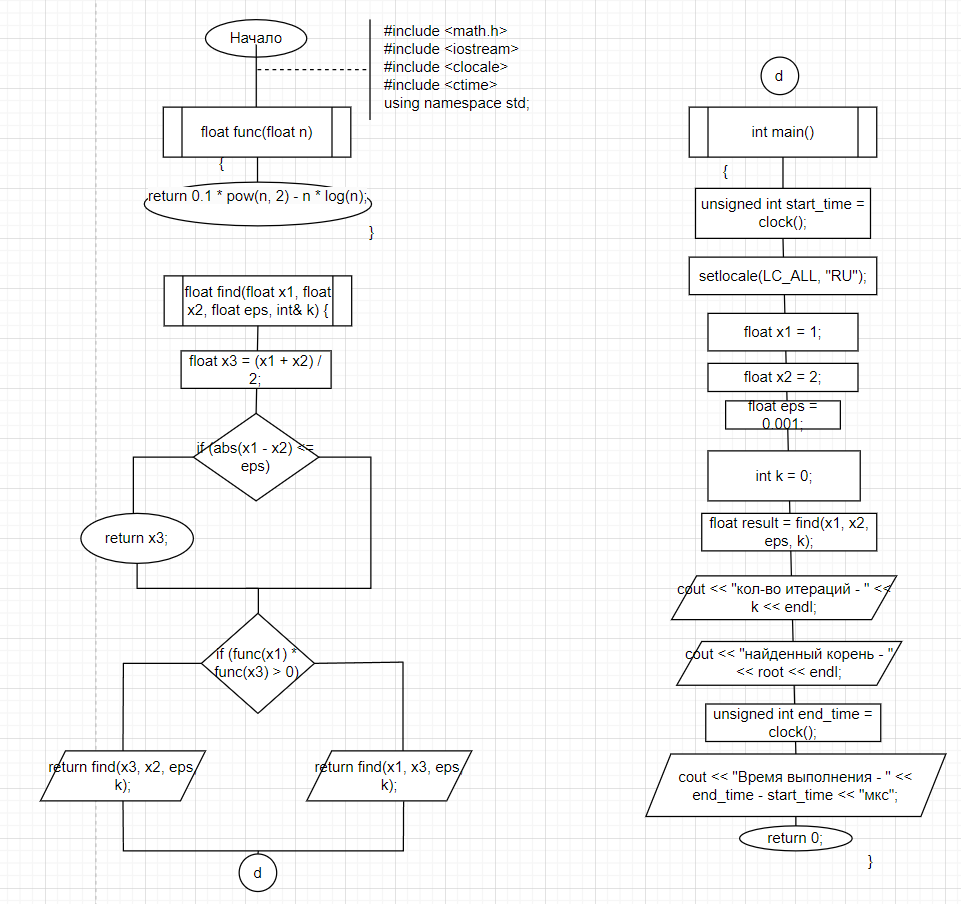
3. Проверяем наличие корня на интервале ab, он существует, поэтому функция f(x) пересекает ось ох

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

Блок-схема



Код в c++

#include<ctime>

#include <iostream>

#include <clocale>

#include <cmath>

using namespace std;

float func(float n) {

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

}

float find(float x1, float x2, float eps, int& k) {

float x3 = (x1 + x2) / 2;

if (abs(x1 - x2) <= eps) {

return x3; // Возвращаем найденное значение

}

k++;

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

return find(x3, x2, eps, k); // Ищем в правой половине

}

else {

return find(x1, x3, eps, k); // Ищем в левой половине

}

}

int main() {

unsigned int start\_time = clock();

setlocale(LC\_ALL, "RU");

float x1 = 1;

float x2 = 2;

float eps = 0.001;

int k = 0; // Счетчик итераций

float result = find(x1, x2, eps, k); // Запускаем рекурсивный поиск

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

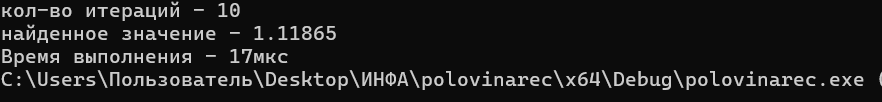
cout << "найденное значение - " << result << endl;

unsigned int end\_time = clock();

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

return 0;

}



2. Постановка задачи: найти девятое число последовательности Фибоначчи с помощью рекурсии

Анализ задачи

1.При реализации рекурсивной функции для вычисления чисел Фибоначчи необходимо определить базовые случаи:

F(0) : возвращает 0.

F(1) : возвращает 1.

Эти базовые случаи предотвращают бесконечную рекурсию и позволяют функции корректно завершаться.

2. Для любого значения n > 1 функция будет вызывать себя дважды:

• Один раз для n - 1

• Другой раз для n - 2

Это приводит к следующей рекурсивной формуле:

F(n) = F(n-1) + F(n-2)

Таким образом, чтобы вычислить F(n) , нам нужно знать два предыдущих числа: F(n-1) и F(n-2) .

• F(2) = F(1) + F(0) = 1 + 0 = 1

• F(3) = F(2) + F(1) = 1 + 1 = 2

• F(4) = F(3) + F(2) = 2 + 1 = 3

• F(5) = F(4) + F(3) = 3 + 2 = 5

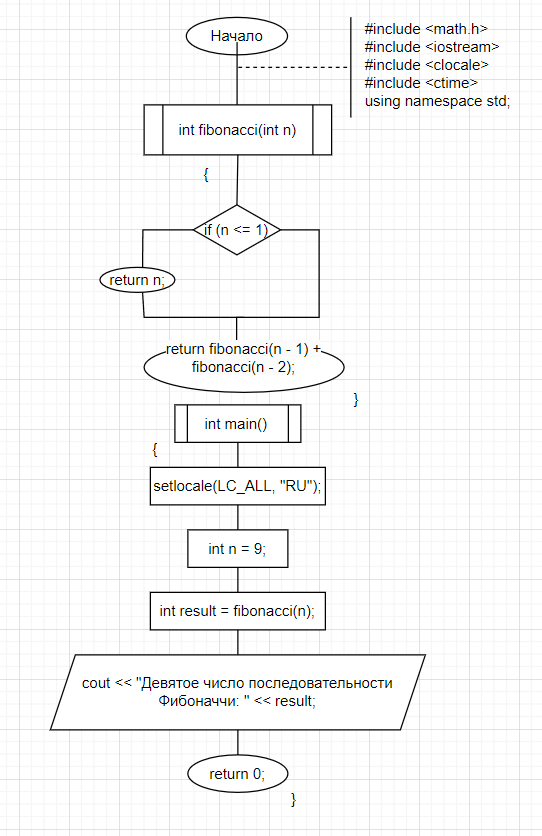
• F(6) = F(5) + F(4) = 5 + 3 = 8

• F(7) = F(6) + F(5) = 8 + 5 = 13

• F(8) = F(7) + F(6) = 13 + 8 = 21

• F(9) = F(8) + F(7) = 21 + 13 = 34

Блок-схема



Код c++

#include <iostream>

#include <clocale>

using namespace std;

int fibonacci(int n) {

if (n <= 1) {

return n;

}

return fibonacci(n - 1) + fibonacci(n - 2);

}

int main() {

setlocale(LC\_ALL, "RU");

int n = 9;

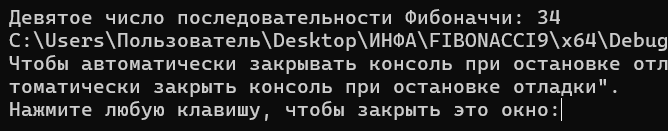
int result = fibonacci(n);

cout << "Девятое число последовательности Фибоначчи: " << result;

return 0;

}

Вывод в консоли



3. Постановка задачи

Для х изменяющегося от a до b с шагом (b-a)/k, где (k=10), вычислить функцию f(x), используя ее разложение в степенной ряд в двух случаях:

а) для заданного n;

б) для заданной точности ε (ε=0.0001).

Для сравнения найти точное значение функции.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 10 |  |  | 20 |  |

Код

#include <iostream>

#include <cmath>

using namespace std;

float factorial(float n) {

if (n == 1) {

return 1;

}

return n \* factorial(n - 1);

}

float sum(float n, float x) {

if (n == 1) {

return 1;

}

return cos(n \* x) / (factorial(n)) + sum(n - 1, x);

}

int main() {

setlocale(LC\_ALL, "RU");

int n,x;

cout << "Введите значение x (в радианах): ";

cin >> x;

cout << "Введите значение n: ";

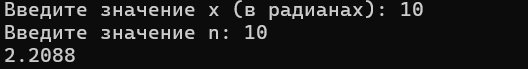
cin >> n;

cout << sum(n, x)+1;

return 0;

}

Вывод в консоль



Блок-схема

