Министерство образования республики Молдова

Технический университет Молдовы

Факультет информатики, вычислительной техники и микроэлектроники

Кафедра Информатики и прикладных иностранных языков

**Отчёт**

Лабораторная работа №1

Тема: Циклические вычислительные процессы

Выполнил студент гр. R-252: Марков Иван

Проверил: Фалько Николай

Кишинёв 2025

**СОДЕРЖАНИЕ**

[Цель работы: 3](#_Toc27030)

[Задание: 3](#_Toc13473)

[Блок-схема 4](#_Toc7480)

[Вариант кода от Искуственного Интелекта 5](#_Toc12153)

[Личный вариант кода 11](#_Toc22804)

[Заключение 15](#_Toc11741)

# Цель работы:

Изучение и практическое применение операторов цикла и условных конструкций в языке программирования C++, освоение методов форматированного вывода числовых данных, а также анализ гибкости и глубины написанного кода.

# Задание:

Вычислить и вывести на экран в виде таблицы значения функции F на интервале от Хнач. до Хкон. с шагом dX.

*ax2 – bx + c при х < 3 и b ≠ 0*

***F = *** *при х > 3 и b =*

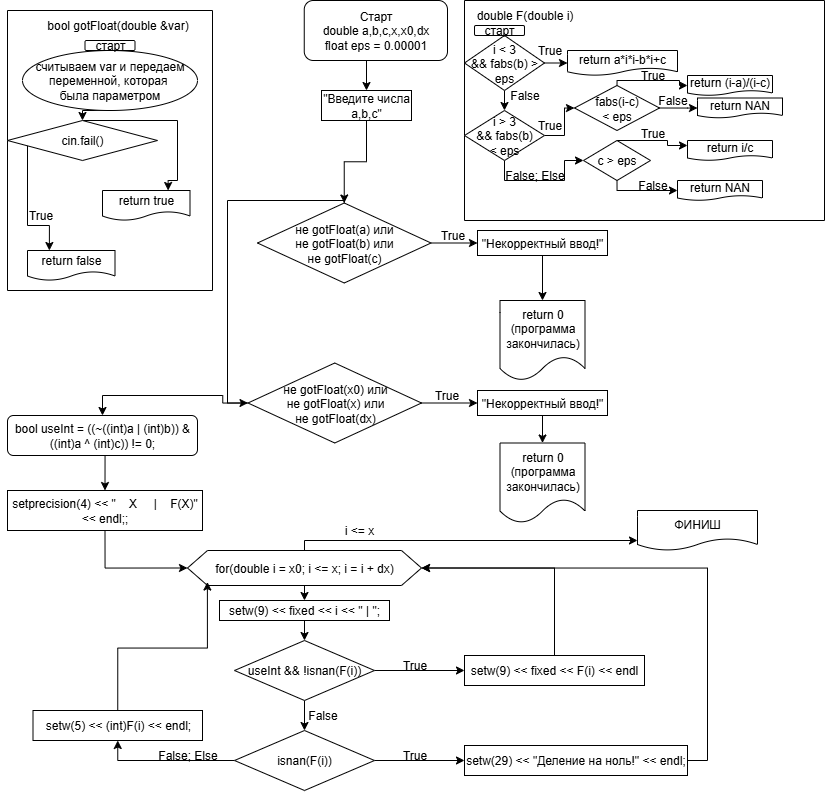
* в остальных случаях*

где а, b, с - действительные числа.

Функция F должна принимать действительное значение, если выражение НЕ(Ац ИЛИ Вц) И (Ац МОД2 Сц) не равно нулю, и целое значение в противном случае. Через Ац, Вц и Сц обозна­чены целые части значений а, b, с, операции НЕ, И, ИЛИ и МОД2 (сложение по модулю 2) - поразрядные. Значения а, b, с, Хнач., Хкон., dХ ввести с клавиа­туры.

(Вариант 10 из методички по лабораторным работам 1)

# Блок-схема



# Вариант кода от Искуственного Интелекта

Первый код(сырой), который был написан при помощи ChatGPT.

#include <iostream>

#include <iomanip>

#include <cmath>

using namespace std;

// Функция для операции MOD2 (XOR)

int mod2(int x, int y) {

return x ^ y;

}

int main() {

setlocale(LC\_ALL, "Russian");

double a, b, c;

double Xnach, Xkon, dX;

cout << "Введите a, b, c: ";

cin >> a >> b >> c;

cout << "Введите Xнач, Xкон, dX: ";

cin >> Xnach >> Xkon >> dX;

// Целые части

int Aц = (int)floor(a);

int Bц = (int)floor(b);

int Cц = (int)floor(c);

cout << fixed << setprecision(4);

cout << " X\t\tF(X)" << endl;

for (double x = Xnach; x <= Xkon; x += dX) {

double F;

// Вычисление функции по условиям

if (x <= 3 && b != 0) {

F = a \* x \* x - b \* x + c;

}

else if (x >= 3 && b == 0) {

if (fabs(x - c) < 1e-9) { // защита от деления на 0

cout << x << "\tделение на 0!" << endl;

continue;

}

F = (x - a) / (x - c);

}

else {

if (fabs(c) < 1e-9) { // защита от деления на 0

cout << x << "\tделение на 0!" << endl;

continue;

}

F = x / c;

}

// Логическое условие

bool condition = ( !(Aц || Bц) && (mod2(Aц, Cц) != 0) );

if (!condition) {

F = floor(F); // берём целую часть

}

cout << setw(6) << x << "\t" << setw(10) << F << endl;

}

return 0;

}

ИИ решил назвать все переменные в точности, как это было указано в задании, однако на строках

int Aц = (int)floor(a);

int Bц = (int)floor(b);

int Cц = (int)floor(c);

редактор кода выдавал ошибку, не понимая русских букв. Поэтому везде, где была буква “ц”, была заменена на английскую “c”. Также ИИ применил для возведения в модуль функцию fabs(). Данная функция применяется на версиях меньше C++11, но в данный момент работает и классическая функция abs(). Кроме того, ChatGPT с первого раза догадался сделать защиту от деления на 0, однако, если вместо цифры ввести букву, программа уйдет в бесконечный цикл, предупреждая, что идет деление на 0. И главная ошибка, при a = 1.5, b = 0, c = 2.7 должны выводиться целые числа. Однако, из-за того что ИИ поставил cout << fixed << setprecision(4), оно всегда выдает действительные числа с 4 цифрами после запятой. Кроме того, неправильно записана поразрядная операция с переменными Ац, Вц, Сц.

#include <iostream>

#include <iomanip>

#include <cmath>

using namespace std;

// Функция для операции MOD2 (XOR)

int mod2(int x, int y) {

return x ^ y;

}

// Функция вычисления F(x) по кусочной формуле

double computeF(double x, double a, double b, double c) {

if (x < 3 && b != 0) {

return a \* x \* x - b \* x + c;

}

else if (x > 3 && b == 0) {

if (fabs(x - c) < 1e-9) return NAN; // защита от деления на 0

return (x - a) / (x - c);

}

else {

if (fabs(c) < 1e-9) return NAN; // защита от деления на 0

return x / c;

}

}

// Функция для безопасного ввода числа

bool safeInput(double &var) {

cin >> var;

if (cin.fail()) {

return false; // введено не число

}

return true;

}

int main() {

setlocale(LC\_ALL, "Russian");

double a, b, c;

double Xnach, Xkon, dX;

cout << "Введите a, b, c: ";

if (!safeInput(a) || !safeInput(b) || !safeInput(c)) {

cout << "Ошибка ввода! Введено не число." << endl;

return 1;

}

cout << "Введите Xнач, Xкон, dX: ";

if (!safeInput(Xnach) || !safeInput(Xkon) || !safeInput(dX)) {

cout << "Ошибка ввода! Введено не число." << endl;

return 1;

}

// Целые части

int Ac = static\_cast<int>(floor(a));

int Bc = static\_cast<int>(floor(b));

int Cc = static\_cast<int>(floor(c));

cout << " X\tF(X)" << endl;

for (double x = Xnach; x <= Xkon; x += dX) {

double F = computeF(x, a, b, c);

if (isnan(F)) {

cout << setw(6) << x << "\tделение на 0!" << endl;

continue;

}

// Условие: НЕ(Ац ИЛИ Вц) И (Ац МОД2 Сц) != 0

// Условие: все операции поразрядные

bool condition = ((~(Ac | Bc)) & (Ac ^ Cc)) != 0;

cout << setprecision(3) << setw(6) << x << "\t";

if (condition) {

cout << fixed << setprecision(3) << F; // действительное

} else {

cout << static\_cast<int>(F); // целое

}

cout << endl;

}

return 0;

}

Теперь этот код выдает правильный формат чисел, реагирует если введено что-то кроме цифр и защищен от деления на 0. Но, этот вариант кода я добивался долго, потому что ИИ не понимал условие проверки с Ац, Вц, Сц. Он сравнивал, не равняется ли нулю, только часть со сложением по модулю 2. Также, пока он изменял сложение по модулю, забыл стереть функцию, которую он для этого заготовил. Также, из-за того что после первой итерации, когда функция принимает действительные значение, X тоже становится действительным и получается немного криво; первая итерация X пишется с целыми числами, а вторая уже с дробным числом. Это можно будет увидеть чуть ниже.

**Тестирование программы от ChatGPT:**

Входные результаты: a = 1, b = 2, c = 3, Хнач = 1, Хкон = 10, dX = 1;  
Вывод:

Введите a, b, c: 1 2 3

Введите Xнач, Xкон, dX: 1 10 1

X F(X)

1 2

2 3

3 1

4 1

5 1

6 2

7 2

8 2

9 3

10 3

Входные результаты: a = 0, b = 0, c = 1, Хнач = 1, Хкон = 10, dX = 1;  
Вывод:

Введите a, b, c: 0 0 1

Введите Xнач, Xкон, dX: 1 10 1

X F(X)

1 1.000 (вот тут он выводит целый X, а дальше дробный)

2.000 2.000

3.000 3.000

4.000 1.333

5.000 1.250

6.000 1.200

7.000 1.167

8.000 1.143

9.000 1.125

10.000 1.111

**Итог:** ChatGPT дается сложно понимание сложных логических функций, особенно если побитово. Также есть небольшие проблемы с форматированием вывода.

# Личный вариант кода

#include <iostream>

#include <iomanip>

#include <cmath>

using namespace std;

double a, b, c, x0, x, dx;

float eps = 0.00001;

double F(double i){

if( i < 3 && fabs(b) > eps){

return a\*i\*i - b\*i + c;

}else if (i > 3 && fabs(b) < eps){

if(fabs(i - c) > eps){

return (i-a)/(i-c);

}else{

return NAN;

}

}else{

if(c > eps){

return i/c;

}else{

return NAN;

}

}

}

bool gotFloat(double &var) {

cin >> var;

if (cin.fail()) {

return false;

}

return true;

}

int main(){

setlocale(LC\_ALL, "Russian");

cout << "Введите числа a, b, c: ";

if(!gotFloat(a) || !gotFloat(b) || !gotFloat(c)){

cout << "Некорректный ввод!";

return 0;

}

cout << "Введите числа Хнач, Хкон, dX: ";

if(!gotFloat(x0) || !gotFloat(x) || !gotFloat(dx)){

cout << "Некорректный ввод!";

return 0;

}

cout << endl;

bool useInt = ((~((int)a | (int)b)) & ((int)a ^ (int)c)) != 0;

cout << setprecision(4) << " X | F(X)" << endl;

for(double i = x0; i <= x; i = i + dx){

cout << setw(9) << fixed << i << " | ";

if (useInt && !isnan(F(i))){

cout << setw(9) << fixed << F(i) << endl;

}else if(isnan(F(i))){

cout << setw(29) << "Деление на ноль!" << endl;

}else{

cout << setw(5) << (int)F(i) << endl;

}

}

}

Свой вариант кода я старался сделать лучше, чем у ИИ, но сильных отличий не получилось. Во первых, я просил ChatGPT сделать конкретные изменения в его коде и использовал это же в своем. Во вторых, увидев его код, он мне показался самым разумным и я не смог придумать что-то лучше, чем уже было. Из улучшений кода получилось только немного укоротить его, сделать лучше оформление вывода и возможно сделать быстрее. Во время создания своего кода, я не подсматривал в первый вариант!

**Тестирование программы**

Входные результаты: a = 1, b = 2, c = 3, Хнач = 1, Хкон = 10, dX = 1;  
Вывод:

Введите числа a, b, c: 1 2 3

Введите числа Хнач, Хкон, dX: 1 10 1

X | F(X)

1.0000 | 2

2.0000 | 3

3.0000 | 1

4.0000 | 1

5.0000 | 1

6.0000 | 2

7.0000 | 2

8.0000 | 2

9.0000 | 3

10.0000 | 3

Входные результаты: a = 0, b = 0, c = 1, Хнач = 1, Хкон = 10, dX = 1;  
Вывод:

Введите числа a, b, c: 0 0 1

Введите числа Хнач, Хкон, dX: 1 10 1

X | F(X)

1.0000 | 1.0000

2.0000 | 2.0000

3.0000 | 3.0000

4.0000 | 1.3333

5.0000 | 1.2500

6.0000 | 1.2000

7.0000 | 1.1667

8.0000 | 1.1429

9.0000 | 1.1250

10.0000 | 1.1111

**Итог:**

Результаты получились такие же, как и у ChatGPT, значит программа выполняет вычисления верно. Также программа делает более красивый вывод результатов и выдает правильный формат чисел в зависимости от входных переменных. Этот код был написан практически с первого раза, в отличии от ИИ, которому долго пришлось объяснять условия.

# Заключение

Во время лабораторной работы, я научился работать с циклом for, оператором условия if, форматами вывода, логическими операторами, нестандартными типами данных, функциями на языке программирования С++. Также, выяснил, что ИИ пишет код не намного лучше, чем человек.