



**«Московский государственный технический университет  
имени Н.Э. Баумана»  
(национальный исследовательский университет)  
(МГТУ им. Н.Э. Баумана)**

---

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ  
КАФЕДРА КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)

**О т ч е т**

**по домашнему заданию № 1**

**Название домашней работы:** Вычисления. Погрешности вычислений.

**Дисциплина:** Алгоритмизация и программирование

Студент гр. ИУ6-15Б

(Подпись, дата)

В.А. Бирюков

(И.О. Фамилия)

Преподаватель

(Подпись, дата)

А.А. Веселовский

(И.О. Фамилия)

Москва, 2024

## Часть 1

**Задание 1:** Создайте новый проект в отдельной папке и введите программу, представленную ниже, заменив выражения в фигурных скобках соответствующими операторами.

```
#include <iostream>
#include <math.h>

using namespace std;

int main()
{
    float y;
    cout << "До инициализации y = " << y << endl;
    y = 1;
    cout << "После инициализации y = " << y << endl;
    y = y / 6000;
    y = exp(y);
    y = sqrt(y);
    y = y / 14;
    y = 14 * y;
    y = y * y;
    y = log(y);
    y = 6000*y;
    cout << "После преобразований y = " << y << endl;
    return 0;
}
```

2. Выполните оценку абсолютной и относительной погрешности представления числа 1 и вычислений над числами типа *float*. К каким типам относятся данные погрешности (см. список типов погрешностей на предыдущей странице)?
3. Текст программы и результаты занесите в отчет.

Перепишем код программы

```
#include <iostream>
#include <math.h>
using namespace std;

int main()
{
    float y;
    cout << "Before initialization y = " << y << endl;
    y = 1;
    cout << "After initialization y = " << y << endl;
    y = y / 6000;
    y = exp(y);
    y = sqrt(y);
    y = y / 14;
    y = 14 * y;
    y = y * y;
    y = log(y);
    y = 6000*y;
    cout << "After transformation y = " << y << endl;
    return 0;
}
```

Рисунок 1 – код программы

Запустим программу

```
D:\ed\AiP\DZ1\cmake-build-debug\DZ1.exe
Before initialization y = 0
After initialization y = 1
After transformation y = 0.999844
|
Process finished with exit code 0
```

Рисунок 2 – результат работы программы

Вычислим абсолютную погрешность:  $\Delta = |A - a| = |1 - 0.999844| = 0.000156$ .

Вычислим относительную погрешность:  $\delta = \Delta / |A| = 0.000156$ .

Эта погрешность относится к погрешностям, связанным с ограниченным количеством разрядов, используемых для представления чисел (погрешности округления)

**Задание 2:** Из математики известно, что  $\text{ch}^2 x - \text{sh}^2 x = 1$ , где  $\text{ch} x = \frac{e^x + e^{-x}}{2}$ ,  $\text{sh} x = \frac{e^x - e^{-x}}{2}$ . Разработайте программу, которая вычисляет левую часть этого равенства.

*Указание.* Программа должна реализовывать следующую последовательность вычислений:  $y_1 = \text{sh } x$ ,  $y_2 = \text{ch } x$ ,  $y = y_2^2 - y_1^2$ , где  $x$ ,  $y$ ,  $y_1$ ,  $y_2$  – переменные типа *float*.

Полученные значения  $y_1$ ,  $y_2$  и  $y$  вывести на экран, указав ширину поля вывода не менее 20 и количество дробных цифр не менее 16.

2. Текст программы и ее результаты занесите в отчет.
3. Последовательно вводя указанные значения аргумента и рассчитывая погрешности вычислений, заполните таблицу.

x	y1	y2	y	$\Delta$	$\delta$
5					
10					
15					
20					
25					

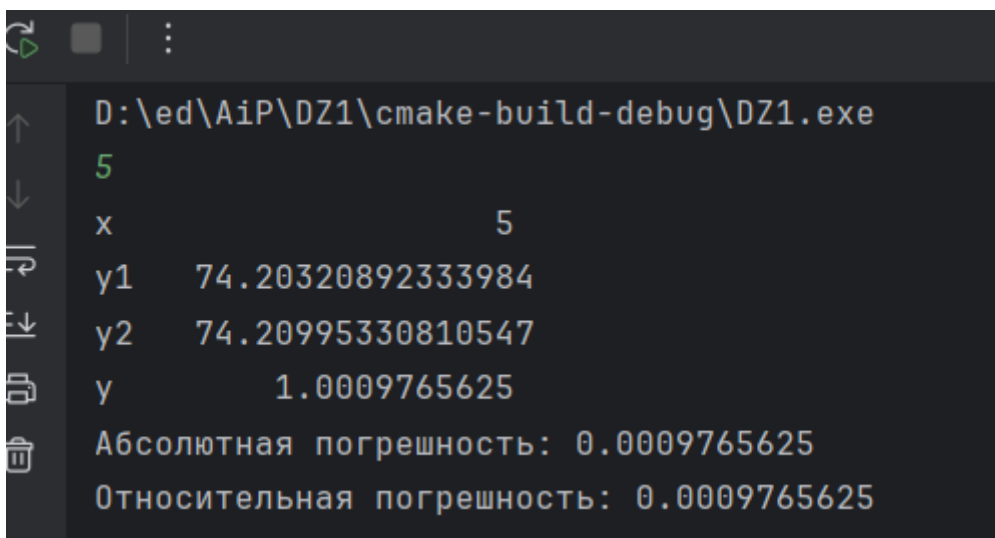
4. Поясните полученный результат и объяснения включите в отчет.
5. Измените в программе типы переменных  $x$ ,  $y$ ,  $y_1$ ,  $y_2$  на *double*. Повторите опыт и заполните аналогичную таблицу. Повторите опыт с типом *long double*. Сравните три таблицы и объясните полученные результаты.
6. Ответьте на вопрос: изменение типа данных каких переменных (из  $x$ ,  $y$ ,  $y_1$ ,  $y_2$ ) реально влияет на точность результата и почему?

Напишем программу на языке C++

```
1  #include <iostream>
2  #include <math.h>
3  #include <Windows.h>
4  #include <iomanip>
5
6  int main() {
7      SetConsoleOutputCP( wCodePageID: CP_UTF8);
8      float x,y,y1,y2;
9      std::cin >> x;
10     std::cout.precision(16);
11     y1 = (exp(x) - exp(-x)) / 2;
12     y2 = (exp(x) + exp(-x)) / 2;
13     y = y2*y2 - y1*y1;
14     std::cout << "x" << std::setw(n:20) << x << std::endl;
15     std::cout << "y1" << std::setw(n:20) << y1 << std::endl;
16     std::cout << "y2" << std::setw(n:20) << y2 << std::endl;
17     std::cout << "y" << std::setw(n:20) << y << std::endl;
18     std::cout << "Абсолютная погрешность: " << abs(x:1-y) << std::endl;
19     std::cout << "Относительная погрешность: " << abs(x:1-y) / 1 << std::endl;
20     return 0;
21 }
```

Рисунок 3 – код программы

Запустим программу и посмотрим на результат



```
D:\ed\AiP\DZ1\cmake-build-debug\DZ1.exe
5
x          5
y1  74.20320892333984
y2  74.20995330810547
y    1.0009765625
Абсолютная погрешность: 0.0009765625
Относительная погрешность: 0.0009765625
```

Рисунок 4 – результат работы программы

Заполним таблицу с результатами

x	y1	y2	y	$\Delta$	$\delta$
5	74.20320892 33398 4	74.20995330 81054 7	1.0009765625	0.0009765625	0.0009765625
10	11013.23242 1875	11013.23242 1875	0	1	1
15	1634508.625	1634508.625	0	1	1
20	242582592	242582592	0	1	1
25	36002451456	36002451456	0	1	1

Таблица 1 – результаты работы программы

Изменим тип переменных x,y,y1,y2 на double и заполним аналогичную таблицу

x	y1	y2	y	$\Delta$	$\delta$
5	74.20321057 77887 5	74.20994852 47878 5	1.00000000000 1819	1.8189894035 45856 e-12	1.8189894035 45856 e-12
10	11013.23287 47033 9	11013.23292 01033 2	1.00000002980 2322	2.9802322387 69531e-08	2.9802322387 69531 e-08
15	1634508.686 23590 2	1634508.686 23620 8	1	0	0
20	242582597.7 04895 1	242582597.7 04895 1	0	1	1
25	36002449668	36002449668	0	1	1

	.6929 4	.6929 4			
--	------------	------------	--	--	--

Таблица 2 – результаты работы программы с типом double

Изменим тип переменных на long double и ещё раз заполним аналогичную таблицу

x	y1	y2	y	$\Delta$	$\delta$
5	74.20321057 77887 6	74.20994852 47878 4	0.99999999999 99991	8.8817841970 01252 e-16	8.8817841970 01252 e-16
10	11013.23287 47033 9	11013.23292 01033 2	1	0	0
15	1634508.686 23590 2	1634508.686 23620 8	1.00000023841 8579	2.3841857910 15625 e-07	2.3841857910 15625 e-07
20	242582597.7 04895 1	242582597.7 04895 1	1.00390625	0.00390625	0.00390625
25	36002449668 .6929 4	36002449668 .6929 4	0	1	1

Таблица 3 – результат работы программы с типом long double

На основе полученных результатов, можем сделать вывод, что увеличение количества байт для записи числа увеличивает точность вычислений

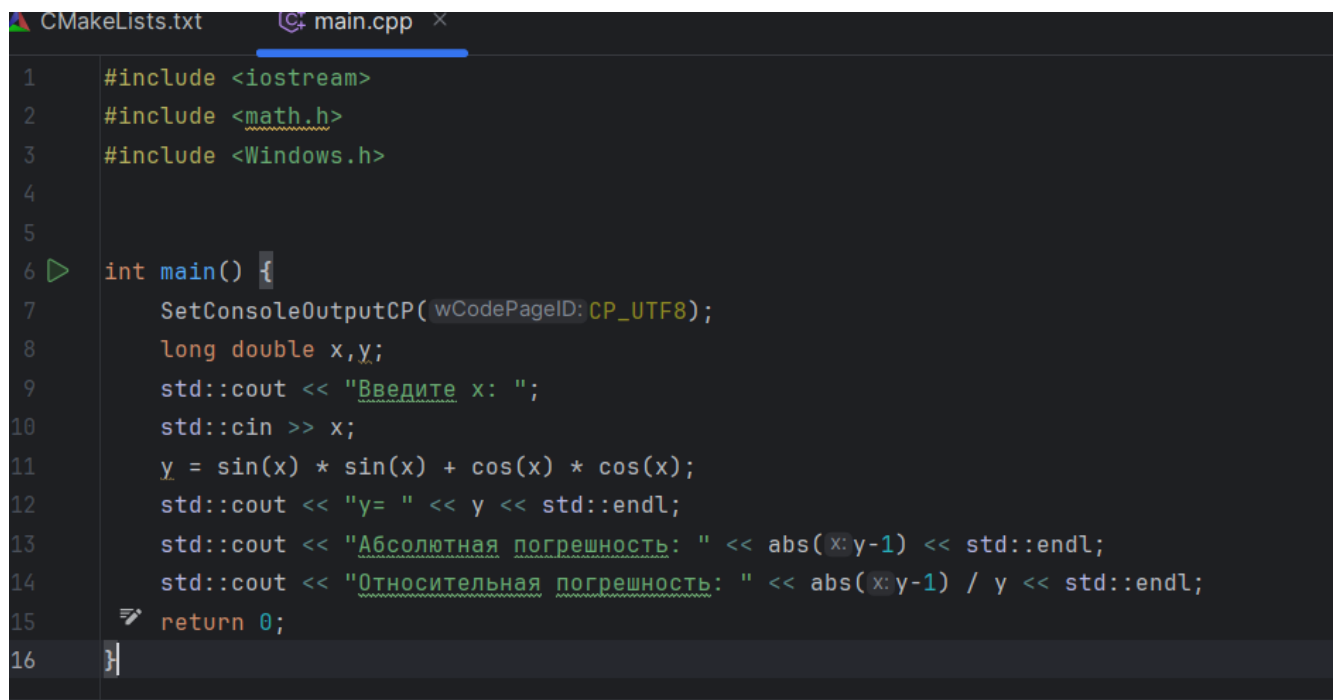
Действительно влияет на точность вычислений изменений типа переменных y1 и y2, так как операция деления, в которой они участвуют, менее точна

### Задание 3:

Разработайте программу, которая проверяет равенство  $\sin^2 x + \cos^2 x = 1$ .

Убедитесь, что погрешность достаточно мала. Поясните полученный результат.

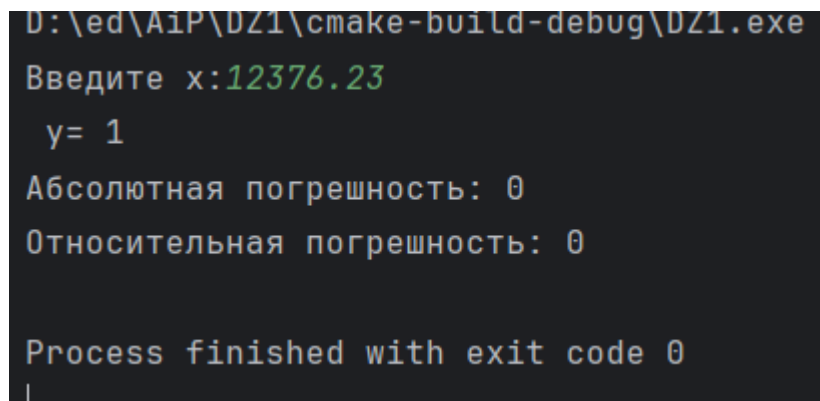
Напишем код программы на языке C++



```
1  #include <iostream>
2  #include <math.h>
3  #include <Windows.h>
4
5
6  int main() {
7      SetConsoleOutputCP( wCodePageID: CP_UTF8);
8      long double x,y;
9      std::cout << "Введите x: ";
10     std::cin >> x;
11     y = sin(x) * sin(x) + cos(x) * cos(x);
12     std::cout << "y= " << y << std::endl;
13     std::cout << "Абсолютная погрешность: " << abs(x*y-1) << std::endl;
14     std::cout << "Относительная погрешность: " << abs(x*y-1) / y << std::endl;
15     return 0;
16 }
```

Рисунок 5 – код программы

Запустим программу и посмотрим на результат



```
D:\ed\A1P\021\cmake-build-debug\021.exe
Введите x:12376.23
y= 1
Абсолютная погрешность: 0
Относительная погрешность: 0

Process finished with exit code 0
```

Рисунок 6 – результат работы программы



Погрешность в вычисления синуса и косинуса достаточно мала, чтобы не повлиять на результат вычислений

## Часть 2

### Вариант 2

**Задание:** Заданы три действительных числа  $x$ ,  $y$  и  $z$ . Присвоить логической переменной значение «true», если только одно из этих чисел положительно. В противном случае логической переменной присвоить значение «false». Вывести результат на экран.

Напишем код программы на языке C++

```
int main() {
    SetConsoleOutputCP( wCodePageID: CP_UTF8);
    float x,y,z;
    int k = 0;
    bool b = false;
    std::cout << "Введите 3 числа" << std::endl;
    std::cin >> x >> y >> z;
    if (x>0) {
        k += 1;
    }
    if (y>0) {
        k += 1;
    }
    if (z>0) {
        k += 1;
    }
    if (k==1) {
        b = true;
    }
    std::cout << "Результат: " << b << std::endl;
    std::cout << "0 - положительное число не одно/отсутствует" << "\n" << "1 - положительное число одно";
    return 0;
}
```

Рисунок 7 – код программы

Запустим программу и посмотрим на результат

```
D:\ed\AiP\DZ1\cmake-build-debug\DZ1.exe
Введите 3 числа
-213 0 35
Результат: 1
0 - положительное число не одно/отсутствует
1 - положительное число одно
Process finished with exit code 0
|
```

Рисунок 8 – результат работы программы

Составим схему алгоритма для этой программы

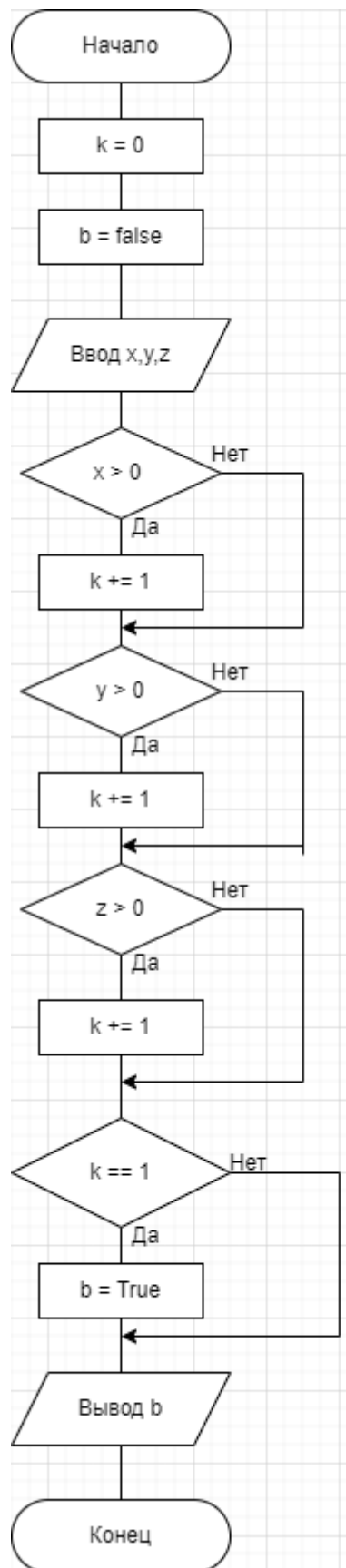
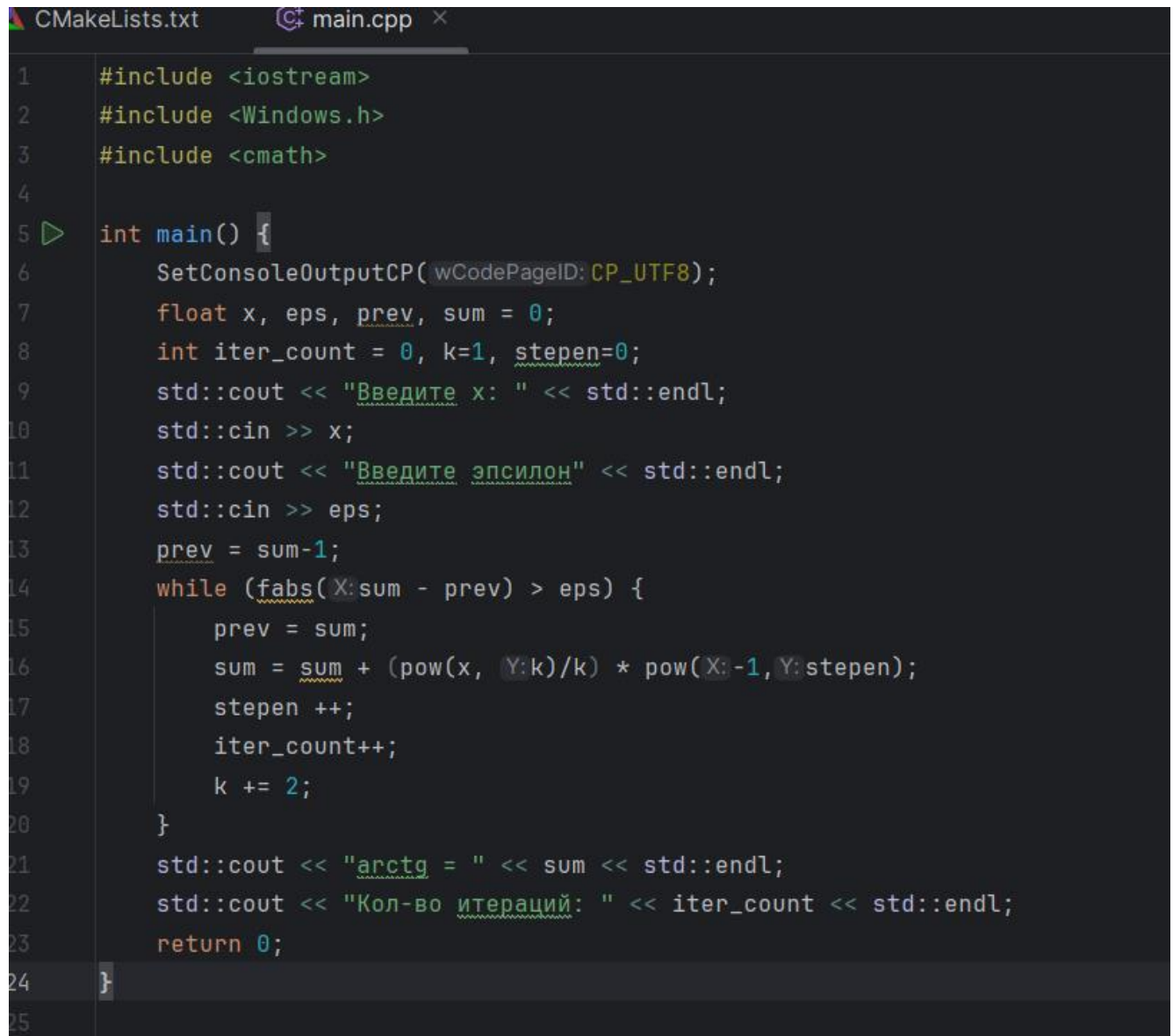


Рисунок 9 – схема алгоритма для программы

**Задание:** Вычислить  $\arctg x = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \dots$  в точке  $x=0,5$  и с точностью

$\xi$ . Значение точности вводится с клавиатуры. Определить, как изменяется число итераций с изменением точности. Проверить программу для  $\xi=10^{-3}, 10^{-4}$ .

Напишем программу на языке C++



```
1  #include <iostream>
2  #include <Windows.h>
3  #include <cmath>
4
5  int main() {
6      SetConsoleOutputCP( wCodePageID: CP_UTF8);
7      float x, eps, prev, sum = 0;
8      int iter_count = 0, k=1, stepen=0;
9      std::cout << "Введите x: " << std::endl;
10     std::cin >> x;
11     std::cout << "Введите эпсилон" << std::endl;
12     std::cin >> eps;
13     prev = sum-1;
14     while (fabs(X:sum - prev) > eps) {
15         prev = sum;
16         sum = sum + (pow(x, Y:k)/k) * pow(X:-1, Y:stepen);
17         stepen ++;
18         iter_count++;
19         k += 2;
20     }
21     std::cout << "arctg = " << sum << std::endl;
22     std::cout << "Кол-во итераций: " << iter_count << std::endl;
23     return 0;
24 }
```

Рисунок 10 – код программы для расчета арктангенса

Запустим программу и посмотрим на результат

```
D:\ed\AiP\DZ1\cmake-build-debug\DZ1.exe
Введите x:
0.9
Введите эпсилон
0.001
arctg = 0.73322
Кол-во итераций: 17

Process finished with exit code 0
|
```

Рисунок 11 – результат работы программы

Проверим программу для  $\xi = 10^{-3}, 10^{-4}$

```
D:\ed\AiP\DZ1\cmake-build-debug\DZ1.exe
Введите x:
0.6
Введите эпсилон
0.001
arctg = 0.540343
Кол-во итераций: 6

Process finished with exit code 0
```

Рисунок 12 – результат работы программы для  $\xi = 10^{-3}$

```
D:\ed\AiP\DZ1\cmake-build-debug\DZ1.exe
Введите x:
0.6
Введите эпсилон
0.0001
arctg = 0.540412
Кол-во итераций: 8
```

Рисунок 13 – результат работы программы для  $\xi = 10^{-4}$

Видим, что при увеличении точности, увеличивается и количество необходимых итераций

Составим схему алгоритма для этой программы

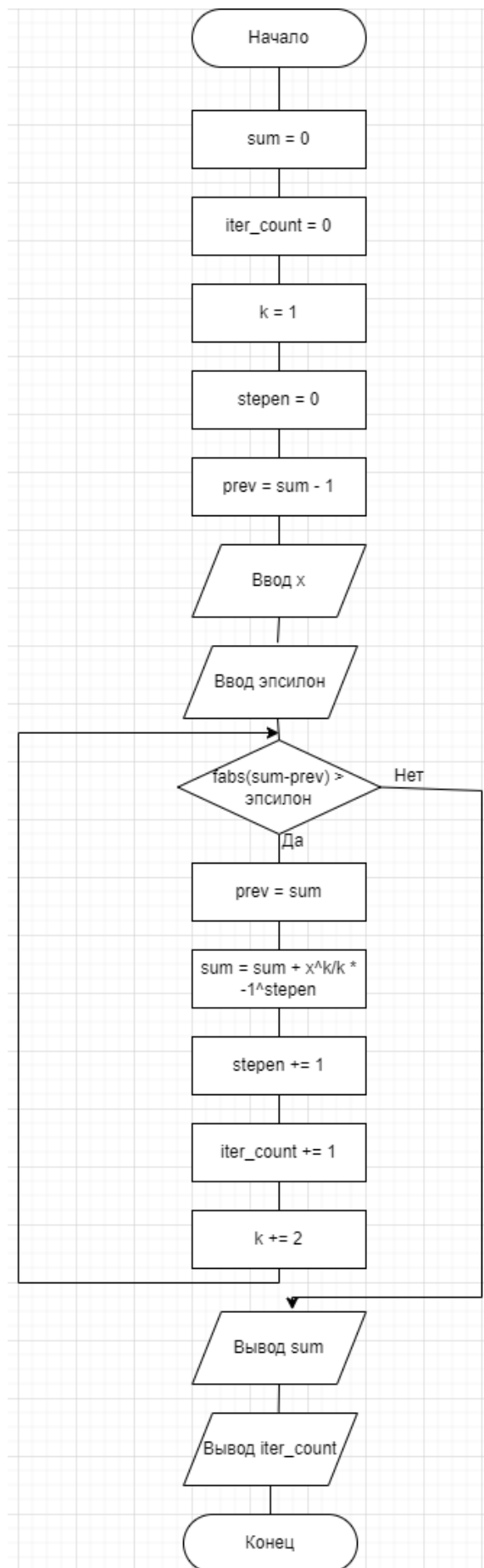


Рисунок 14 – схема алгоритма для программы по вычислению арктангенса