Пермский филиал федерального государственного автономного образовательного учреждения высшего образования  
«Национальный исследовательский университет   
Высшая школа экономики»

*Факультет социально-экономических и компьютерных наук*

Мусихин Данил Михайлович

**Название темы**

*Лабораторная работа №*

студента образовательной программы «Разработка информационных систем для бизнеса» по направлению подготовки *09.03.04 Программная инженерия*

Руководитель, к.т.н., Доцент кафедры ИТБ.

Л.Н. Лядова

Пермь, 2023 год

**Оглавление**

[1 Постановка задачи 3](#_Toc147604594)

[2 Анализ 4](#_Toc147604595)

[3 Проектирование 5](#_Toc147604599)

[4 Код программы 7](#_Toc147604600)

# Постановка задачи

Для заданного значения x (вводится с клавиатуры), вычислить функцию f(x) = arctg(x), используя её разложение в степенной ряд в двух случаях:

1. Для заданного n (n вводится с клавиатуры);
2. Для заданной точности (e вводится с клавиатуры);
3. С максимальной возможной точностью.

Для оценки решения нужно найти точное значение функции, используя стандартные функции, реализованные в системе программирования.

Вычисления нужно повторить для разных форматов данных с плавающей точкой, подсчитывая количество итераций при вычислениях с заданной точностью e и с максимальной возможной точностью (нужно сравнить полученные результаты и пояснить их).

Ряд Тейлора для заданной функции представлен в таблице 1.

Таблица 1 – Ряд Тейлора

|  |  |
| --- | --- |
| **Функция** | **Сумма (ряд Тейлора)** |
|  |  |

# Анализ

Попробуем разобраться, как сделать вычисление через итерацию и с заданной точностью.

## Вычисление через итерацию

Для этого у нас есть формула:

Используем ее, где n будет у нас количеством итераций.

## Вычисление с заданной точностью

Оно будет производиться тем же принципом за исключением, что условием выхода будет:

Где e – это заданная точность.

## Определим переменные и их типы

Нам понадобятся переменные, чтобы запоминать факториал и степень, чтобы ускорить работу алгоритма и уменьшить нагрузку на процессор.

|  |  |
| --- | --- |
| **Переменная и что она хранит** | **Ее тип** |
| Fact –факториал числа n! | Double. Можно было бы выбрать целочисленный тип, но при работе с ним будут замечены ошибки в вычислении, так как ему не хватит байтов для хранения. |
| Degree – степень (2x)n | Double. Так как х – вещественное число. |
| S – получаемая сумма | Double. Результаты делений степени на факториал при вычислении дают вещественные числа, поэтому логичнее использовать вещественный тип. |

# Проектирование

Ниже на рисунке приведен алгоритм главной функции, которая объединяет 3 части вычисления: самой функции, ее через итерации и ее через заданную точность.

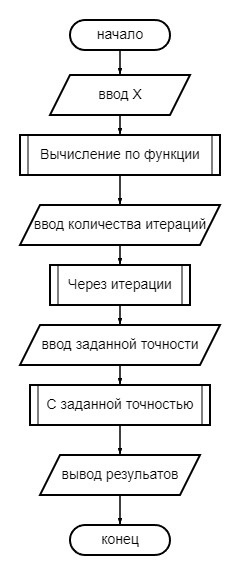


Рисунок 1 - Главная функция

Вычисление по функции производится по формуле:

Вычисление через итерации и с заданной точностью похожи друг на друга, только в одном случае используется итерационный цикл, в другом цикл с условием. Поэтому будет представлен только один вариант вычисления ниже на рисунке 2.

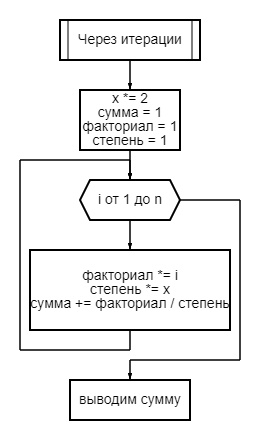


Рисунок 2 - Вычисление функции через итерации

# Код программы

// Функция по вычислению f(x) = e^(2x)

// Вариант 15

#include <iostream>

#include <cmath>

using namespace std;

double f(double x)

{

return pow(M\_E, 2\*x);

}

double fIteration(double x, int n)

{

double s = 1;

double fact = 1;

x \*= 2;

double degree = 1;

for (int i = 1; i <= n; i++)

{

fact \*= i;

degree \*= x;

s += degree / fact;

}

return s;

}

double fWithAccuracy(double x, double E)

{

double s = 0;

double fact = 1;

x \*= 2;

double degree = 1;

int i = 1;

do

{

s += degree / fact;

fact \*= i++;

degree \*= x;

} while(degree / fact > E);

return s;

}

double fAssembler(double x, int n) {

double result;

asm (

"fld1;"

"fldl %1;" // загрузить x в регистр FPU

"fld1;" // загрузить 1 в регистр FPU (для 1!)

"faddp;" // сложить 1 и x

"fld1;" // загрузить 1 в регистр FPU (для начального суммирования)

"movl %2, %%ecx;" // загрузить n в регистр ecx

"test %%ecx, %%ecx;"

"je end;"

"loop\_start:;"

"fld %%st(1);" // загрузить x в регистр FPU

"fmul %%st(0), %%st(1);" // умножить верхний элемент стека на x

"fld1;" // загрузить 1 в регистр FPU

"faddp;" // сложить 1 и x

"fld %%st(0);" // копировать верхний элемент стека (x^n)

"fld %%st(1);" // загрузить n! в регистр FPU

"faddp;" // сложить (2x)^n и n!

"fld %%st(0);" // копировать текущую сумму

"faddp;" // сложить текущую сумму с (2x)^n/n!

"loop loop\_start;"// повторить цикл n раз

"end:;"

"fstp %0;" // сохранить результат в переменной result

: "=m" (result) // вывод: результат сохраняется в переменной result

: "m" (x), "m" (n) // вход: x и n берутся из переменных x и n

: "ecx" // ecx используется как временный регистр

);

return result;

}

int main()

{

double x; int n; double e;

cout << "Введите x: ";

cin >> x;

cout << "Значение функции = " << f(x) << endl;

cout << "Введите количество итераций для вычисления: ";

cin >> n;

cout << "Значение функции через итерационный цикл = " << fIteration(x, n) << endl;

cout << "Введите точность вычисления: ";

cin >> e;

cout << "Значений функции через заданную точность = " << fWithAccuracy(x, e) << endl;

cout << "Значение функции через ассемблерную вставку = " << fAssembler(x, n) << endl;

}