Московский авиационный институт

(национальный исследовательский университет)

Институт №8 «Информационные технологии и прикладная математика»

Кафедра 806 «Вычислительная математика и программирование»

Курсовой проект

по курсу «Вычислительные системы»

1 семестр

Задание 7.

Тема: «Вещественный тип. Приближённые вычисления. Табулирование функций.»

|  |  |
| --- | --- |
| **Студент:** | Терентьев М.А. |
| **Группа:** | М8О - 101Б - 23 |
| **Преподаватель:** | Крылов С.С |
| **Подпись:** |  |
| **Оценка:** |  |

Москва 2023

**Оглавление**

[Задание. 2](#_heading=h.gjdgxs)

[Формулировка. 2](#_heading=h.30j0zll)

[Вариант задания. 3](#_heading=h.1fob9te)

[Работа. 3](#_heading=h.3znysh7)

[Код программы. 3](#_heading=h.2et92p0)

[Описание алгоритма. 4](#_heading=h.tyjcwt)

[Протокол выполнения программы. 5](#_heading=h.3dy6vkm)

[Вывод. 6](#_heading=h.1t3h5sf)

# Задание.

## Формулировка.

Составить программу на Си, которая печатает таблицу значений элементарной функции, вычисленной двумя способами: по формуле Тейлора и с помощью встроенных функций языка программирования. В качестве аргументов таблицы взять точки разбиения отрезка [a, b] на n равных частей (n + 1 точка включая концы отрезка), находящихся в рекомендованной области хорошей точности формулы Тейлора. Вычисления по формуле Тейлора проводить по экономной в сложностном смысле схеме с точностью EPS\*k, где EPS — машинное эпсилон аппаратно реализованного вещественного типа для данной ЭВМ, а k — экспериментально подбираемый коэффициент, обеспечивающий приемлемую сходимость. Число итераций должно ограничиваться сверху числом порядка 100. Программа должна сама определять машинное EPS и обеспечивать корректные размеры генерируемой таблицы.

## Вариант задания.



# Работа.

## Код программы.

##include <stdio.h>

#include <math.h>

const float a = 0.0;

const float b = 1.0;

const int fieldSize = 18;

const int columnsCount = 4;

float getEps() {

    float epsilon = 1.0;

    while ((float)(1.0 + (epsilon / 2.0)) != 1.0) {

        epsilon /= 2.0;

    }

    return epsilon;

}

float myPow(float x, int n) {

    float res = 1.0;

    if (n < 0) return 1 / myPow(x, -n);

    for (int i = 0; i < n; i++) res \*= x;

    return res;

}

float calcF(float x) {

    return (1 + x) \* exp(-x);

}

float calcFTaylor(float x, int\* pn, float eps) {

    int iter = 1, currN = 2;

    float res = 1, izm = 0;

    int degree = 1, fact = 1;

    do {

        fact \*= currN;

        iter += 1;

        izm = (-1 \* degree) \* ((currN - 1) / (float)fact) \* myPow(x, currN);

        res += izm;

        currN += 1;

        degree \*= -1;

    } while (fabs(izm) > eps);

    \*pn = iter - 1;

    return res;

}

void typeSplitter() {

    int columnWidth = fieldSize + 3;

    for (int i = 0; i < columnWidth \* columnsCount; i++) {

        if (i % columnWidth == 0) {

            printf("+");

            continue;

        }

        printf("-");

    }

    printf("+\n");

}

void typeHeader() {

    printf("| %18s | %18s | %18s | %18s |\n", "x value", "f(x) - function", "f(x) - Taylor", "Iteration");

}

int main() {

    int steps = 100;

    float step = (b - a) / steps;

    float x = a;

    typeSplitter();

    typeHeader();

    typeSplitter();

    float eps = getEps();

    for (int i = 0; i <= steps; i++) {

        int n = 0;

        int\* pn = &n;

        float fVal = calcF(x);

        float fTaylorVal = calcFTaylor(x, pn, eps);

        printf("| %18.3f | %18.15f | %18.15f | %18.3d |\n", x, fVal, fTaylorVal, n);

        x += step;

    }

    typeSplitter();

    return 0;

}

## Описание алгоритма.

1. Определение констант: Задаются константы a, b, fieldSize и columnsCount.

2. Функция getEps(): Эта функция вычисляет машинное эпсилон, используемое для оценки точности при вычислении ряда Тейлора.

3. Функция myPow(): Эта функция реализует возведение числа в степень.

4. Функция calcF(): Эта функция вычисляет значение функции (1 + x) \* exp(-x).

5. Функция calcFTaylor(): Эта функция вычисляет значение ряда Тейлора для заданного значения x с использованием итеративного подхода.

Цикл do-while: выполняется цикл, в котором вычисляются последовательные члены ряда Тейлора до тех пор, пока абсолютное значение текущего члена (izm) больше заданной точности eps. В каждой итерации:

- Вычисляется факториал текущего значения n (fact).

- Вычисляется изменение (izm) как (-1)^n \* ((n-1)! / n!) \* x^n.

- Результат (res) увеличивается на значение изменения (izm).

- Текущее значение n (currN) увеличивается на 1.

- Степень (-1)^n обновляется для следующей итерации.

6. Функция typeSplitter(): Эта функция выводит разделительную линию для форматирования вывода.

7. Функция typeHeader(): Эта функция выводит заголовок таблицы для форматированного вывода значений x, f(x), f(x) - Taylor и количества итераций.

8. В функции main():

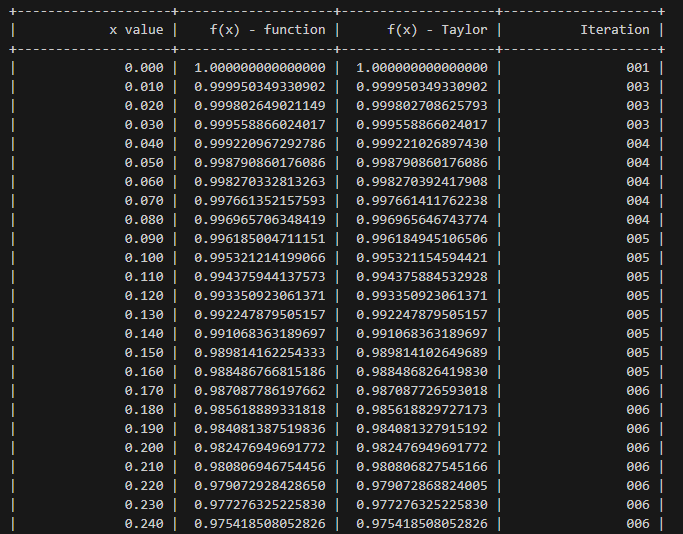
- Вычисляется шаг изменения x на основе заданного количества шагов.

- Выполняется цикл для вычисления значений функции и ее ряда Тейлора для различных значений x в заданном диапазоне.

- Результаты выводятся в виде таблицы.

Таким образом, программа вычисляет значения функции и ее ряда Тейлора для различных значений x в заданном диапазоне с использованием машинного эпсилон для оценки точности.

## Протокол выполнения программы.



# 

# Вывод

В ходе выполнения данного задания курсового проекта я научился реализовывать программную версию вычисления значений функции пользуясь рядом Тейлора для этой функции. Научился выводить данные в виде таблицы, оптимизировать вывод значений переменных с плавающей точкой. Научился с помощью алгоритма определять машинное эпсилон. Закрепил свои навыки работы в языке программирования Си.