Московский Авиационный Институт (Национальный Исследовательский Университет)

Курсовой проект  
по курсу «Вычислительные системы»

1 семестр

Задание 3 «Вещественный тип. Приближенные вычисления. Табулирование функций»

Выполнил: Бровина В.Д.

Группа: М8О-111Б-22

Руководитель: Аносова Н.П.

Оценка: \_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_

Москва, 2022

Оглавление

[Задание 3](#_Toc122711545)

[Алгоритм решения 3](#_Toc122711546)

[Листинг программного кода 4](#_Toc122711547)

[Выходные данные 6](#_Toc122711548)

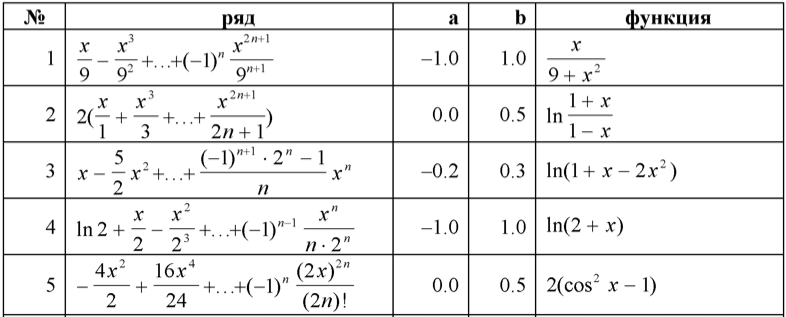
[Вывод 7](#_Toc122711550)

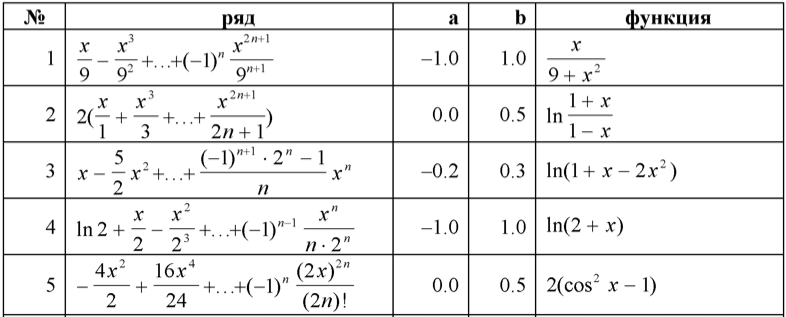
[Заключение 7](#_Toc122711551)

[Список литературы 7](#_Toc122711552)

# **Задание**

Составить программу на языке 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 на равных частей, находящихся в рекомендованной области хорошей точности формулы Тейлора. Вычисления по формуле Тейлора проводить по экономной в сложностном смысле схеме с точностью , где – машинное эпсилон, аппаратно-реализованного вещественного типа для данной ЭВМ, а – экспериментально подбираемый коэффициент, обеспечивающий приемлемую сходимость. Число итераций должно сверху ограничиваться числом порядка 100. Программа должна сама определять машинное эпсилон и обеспечивать корректные размеры генерируемой таблицы.

**Вариант №5**

****

# **Алгоритм решения**

1. Найти машинное эпсилон
2. Задать число N – на сколько отрезков нужно разбить изначальный отрезок
3. Для каждого заданного N вычислять шаг изменения переменной “x”
4. В цикле найти значение суммы ряда и значение функции
5. Выводить значения: “x”, значение суммы ряда Тейлора, значение функции, число итераций.

# **Листинг программного кода**

#include <math.h>

#include <stdio.h>

#include <stdbool.h>

int main(void) {

printf("%s", "Input the number of segmentations: ");

int N;

scanf("%d", &N);

long double a = 0.0, b = 0.5;

long double step = (b - a) / N;

long double x = a;

long double eps = 1.0;

while (1.0 + eps / 2.0 > 1) {

eps /= 2;

}

printf(" ");

for (int x = 0; x < 96; x++) {

printf("\_");

}

printf("\n|\tx\t|\t Taylor's Function \t|\t Function \t| Iterations\t|");

printf("\n");

printf("|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\n");

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

long double row0 = -((4 \* x \* x) / 2);

long double taylor\_function = row0;

int x0 = 2, num\_of\_operations = 0;

bool flag = true;

for (int n = 0; n < 100 && flag; n++) {

row0 = (-1) \* row0 \* ((4 \* x \* x) / ((x0 + 1) \* (x0 + 2)));

if (abs(row0) < eps) {

flag = false;

num\_of\_operations = n;

}

taylor\_function += row0;

x0 += 2;

}

long double function = 2 \* (cos(x) \* cos(x) - 1);

printf("|\t%.3Lf\t|\t%.20Lf\t|\t%.20Lf\t|\t%d\t| \n", x, taylor\_function, function, num\_of\_operations);

x += step;

}

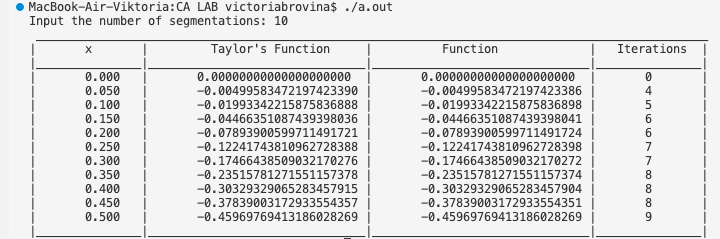
printf("|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\n");

return 0;

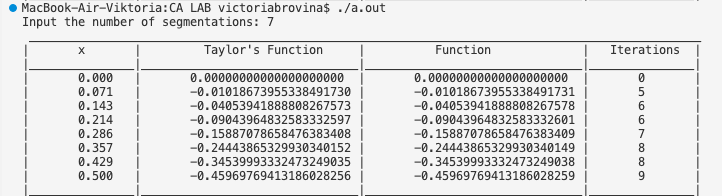
}

# **Выходные данные**

## Для N = 10:



Для N = 7:



# 

# **Вывод**

В ходе работы над данной курсовой работы я научилась вычислять сумму ряда эффективным цикличным методом по предыдущему значению суммы ряда. Навыки оформления таблицы помогут мне в дальнейшем понятно представлять результаты сложных вычислений в терминале.

# **Заключение**

Из таблицы заметим, что значение ряда Тейлора и встроенной функции начинают различаться с 19-20 знака после запятой, что означает, что метод задания функции не является практичным при необходимости получить точное значение более 20 знаков после запятой.

# **Список литературы**

* [**https://www.desmos.com/calculator?lang=ru**](https://www.desmos.com/calculator?lang=ru)
* [**https://en.cppreference.com/w/**](https://en.cppreference.com/w/)
* [**https://core.ac.uk/download/pdf/162891844.pdf**](https://core.ac.uk/download/pdf/162891844.pdf)