Московский авиационный институт

(Национальный исследовательский университет)

Институт №8 «Компьютерные науки и прикладная математика»

Кафедра «Вычислительная математика и программирование»

**Курсовой проект**

**по курсу**

**«Фундаментальная информатика»**

**I семестр**

**Задание 3**

**«Вещественный тип. Приближенные вычисления. Табулирование функций»**

Студент: Горин Н.А.

Группа: М8О-113Б-22

Руководитель: Никулин С.П.

Оценка:

Дата:

Москва

2022 г.

Оглавление

[Задача 3](#_Toc122357752)

[Идея решения 3](#_Toc122357753)

[Протокол 3](#_Toc122357754)

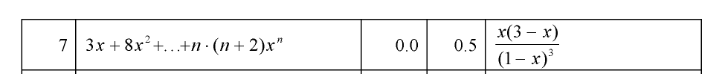
[Вывод 5](#_Toc122357755)

# Задача

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 ɛ\*k, где ɛ - 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 ɛ и обеспечивать корректные размеры генерируемой таблицы.

Вариант:



# Идея решения

Напишем вычисленные значения функции на отрезке [0.0;0.5] двумя способами: с помощью ряда Тейлора и программных средств на языке Си

# Протокол

user@user-Katana:~$ gcc KP3.c -lm

user@user-Katana:~$ .cat KP3.c

/\* Курсовой проект по фундаментальной информатике

Задание 3, Вариант 7

М8О-113Б-22, Горин Никита

\*/

#include <string.h>

#include <stdio.h>

#include <stdbool.h>

#include <ctype.h>

#include <locale.h>

#include <math.h>

int main() {

double T = 0, f = 0;

double a = 0, b = 0.5;

double x = 0;

double e = 1.0;

double sum = 0;

int n = 1;

double d = 0;

int steps;

while (1 + e / 2 > 1) e /= 2;

printf("Машинное эпсилон = %.16f\n", e);

printf("Введите число отрезков: ");

scanf("%d", &steps);

printf("Функция: x(3-x)/(1-x)^3\n");

printf("Ряд Тейлора: 3x + 8x^2 +...+ n(n+2)x^n\n");

printf("Таблица значений ряда Тейлора и обычной функции\n");

printf("|\tx\t|\tCумма для ряда\t\t|\tЗначения функции\t|Число слагаемых ряда\t|\n");

double delta = (b - a) / steps;

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

f = (x \* (3 - x)) / pow((1 - x), 3);

do {

d = n \* (n + 2) \* pow(x, n);

sum += d;

n++;

} while ((d > e || d < -e) && n <= 1000);

printf("--------------------------------------------------------------------------------------------------\n");

printf("|\t%.2f\t|\t%.16f\t|\t%.16f\t|\t", x, sum, f);

printf("%d\t|\n", n - 1);

x += delta;

sum = 0.0;

n = 1;

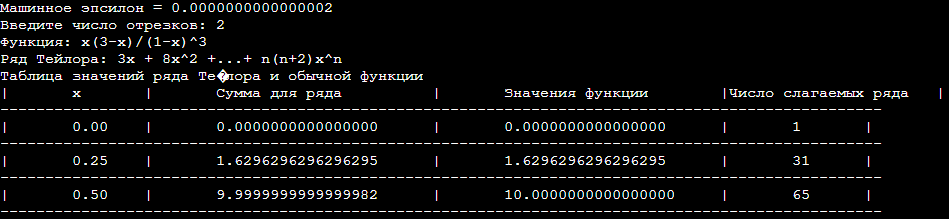
}

printf("--------------------------------------------------------------------------------------------------\n");

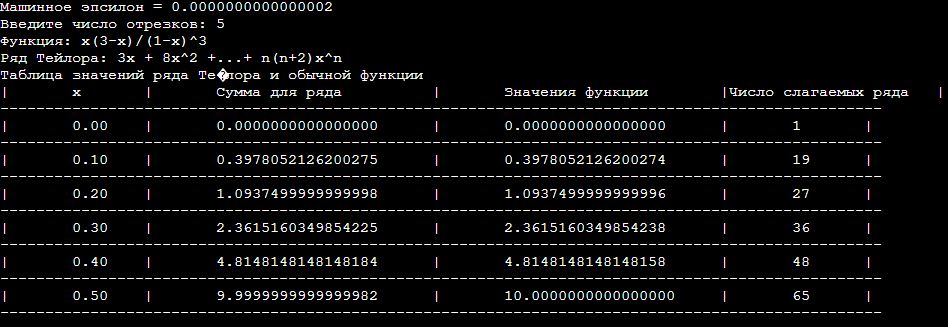
}

(Сделаем проверку для n = 2, n = 5, n = 10)

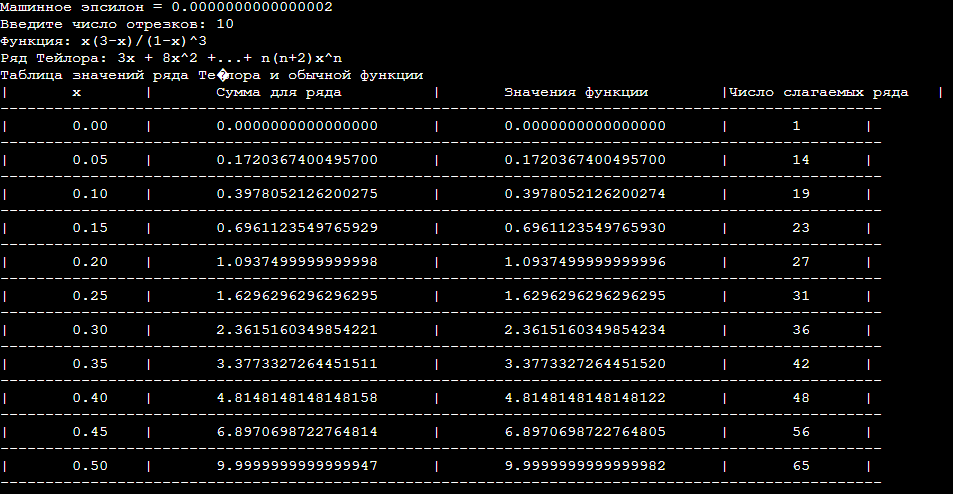
user@user-Katana:~$ ./a.out



user@user-Katana:~$ ./a.out



user@user-Katana:~$ ./a.out



# Вывод

Сравнительная таблица показывает, что значения ряда Тейлора отличаются от значений самой функции примерно на 10^-15. Это означает,

что несмотря на точность задания функции, метод имеет ряд недостатков, в том числе затрачивается много времени при компиляции.

В то же время такие методы дают представление о том, как задаются некоторые математические функции в языках программирования, в частности, Си