Министерство образования науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский авиационный институт (национальный исследовательский университет)»

«Информационные технологии и прикладная математика»

**Курсовой проект**

**По курсу «Вычислительные системы»**

**1 семестр**

**Задание 3:**

**«Приближённые значения. Табулирование функции»**

|  |  |
| --- | --- |
| Группа: | М8О-106Б-21 |
| Студент: | Орусский В.Р. |
| Преподаватель: | Дубинин А.В. |
| Оценка: |  |
| Дата: |  |

Оглавление

[Введение 3](#_Toc92476756)

[Постановка задачи 4](#_Toc92476757)

[Вариант 8: 4](#_Toc92476758)

[Теоретическая часть 5](#_Toc92476759)

[Ряд Тейлора 5](#_Toc92476760)

[Машинный ноль (Машинный нуль) 5](#_Toc92476761)

[Машинный эпсилон 5](#_Toc92476762)

[IEEE 754 6](#_Toc92476763)

[Описание алгоритма 7](#_Toc92476764)

[Исходный код программы: 8](#_Toc92476765)

[Переменные в программе: 9](#_Toc92476766)

[Входные данные: 9](#_Toc92476767)

[Выходные данные 9](#_Toc92476768)

[Протокол исполнения и тесты 10](#_Toc92476769)

[Тест №1 10](#_Toc92476770)

[Тест №2 10](#_Toc92476771)

[Тест №3 11](#_Toc92476772)

[Тест №4 12](#_Toc92476773)

[Вывод 13](#_Toc92476774)

[Список литературы 14](#_Toc92476775)

# Введение

Изучить ряды Тейлора и определение, и нахождение машинного эпсилона, с определённой точностью которого найти приближённое значение функции данной нам функции.

# Постановка задачи

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 ε \* 10k, где ε - 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 ε и обеспечивать корректные размеры генерируемой таблицы.

## Вариант 8:

Ряд Тейлора:



Функция:

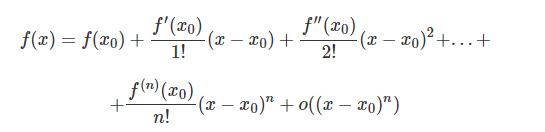


Значения a и b соответственно:



# Теоретическая часть

Ряд Тейлора — разложение функции в бесконечную сумму степенных функций. Многочлен Тейлора применяется при аппроксимации функции многочленами. Сама она является следствием теоремы Лагранжа о среднем значении дифференцируемой функции. В случае x0=0 формула называется рядом Маклорена.



Машинный ноль (Машинный нуль) — числовое значение с таким отрицательным порядком, которое воспринимается машиной как ноль.

Машинный эпсилон - числовое значение, меньше которого невозможно задавать относительную точность для любого алгоритма, возвращающего вещественные числа. Абсолютное значение для машинного эпсилон зависит от разрядности сетки применяемой ЭВМ и от разрядности используемых при расчёте типа данных. Формально машинный эпсилон определяется, как минимальное число ε, удовлетворяющее равенству 1 + ε > 1. Альтернативное определение — максимальное ε, для которого справедливо равенство 1+ε=1. Фактически, два отличных от нуля числа являются равными с точки зрения машинной арифметики, если их модуль разности меньше или не превосходит машинный эпсилон.

В языке Си существуют специальные константы FLT\_EPSILON, DBL\_EPSILON и LDBL\_EPSILON являющиеся «машинным эпсилон», для определённого типа данных, сокращение которого пишется в начале константы. FLT=Float => м.э.(машинный эпсилон) для типа float = 2−23 ≈ 1.19e-07, DBL=Double => м.э. для типа double = 2−52 ≈ 2.20e-16 и LDBL=long double => м.э. для типа long double = 2−63 ≈ 1.08e-19.

IEEE 754 — широко используемый стандарт двоичной арифметики чисел с плавающей точкой (запятой), описывающий формат представления действительных чисел.

Стандарт описывает:

* формат чисел с плавающей точкой: мантисса, экспонента (показатель), знак числа;
* представление положительного и отрицательного нуля, положительной и отрицательной бесконечностей, а также нечисла́ (англ. Not-a-Number, NaN);
* исключительные ситуации: деление на ноль, переполнение, потеря значимости, работа с денормализованными числами и другие;
* операции: арифметические и другие.

Представление числа в памяти по стандарту IEEE 754 основывается на нормализованной экспоненциальной записи числа, через мантиссу и порядок, где мантисса принимает значение от 1 (включительно) до основания системы счисления (не включая). Для представления числа типа double в памяти отводится 64 бита (8 байт) и имеет диапазон представления от 4,94⋅10−324 до 1.79 \* 10308:

1 бит знак + 10 бит — показатель (порядок),

1 бит знак + 52 бита — знак и мантисса.

# Описание алгоритма

Первым делом необходимо найти машинный эпсилон, на котором будет основываться точность вычисления. Это делается до тех пор пока 1 + ε > 1, при том что ε изначально равен 1.0, а на каждой итерации мы его делим пополам.

Суммируем члены разложения ряда Тейлора до тех пор, пока количество членов не достигнет 100, либо разница между k-ым и k+1-ым членом ряда Тейлора не станет меньше машинного эпсилона.

# Исходный код программы:

*/\*  
 КП №3  
 Вариант №8  
 \*/*#include <stdio.h>  
#include <math.h>  
  
*double* get\_machine\_epsilon() {  
 *double* epsilon = 1.0;  
 *while* ((1.0 + epsilon / 2.0) > 1.0) {  
 epsilon /= 2.0;  
 }  
 *return* epsilon;  
}  
  
*double* func(*double* x) {  
 *return* 1.0 / (2.0 \* x - 5.0);  
}  
  
*int* main() {  
 *double* start = 0.0, end = 2.0;  
 *int* n, accuracy\_factor;  
 scanf("%d %d", &n, &accuracy\_factor);  
  
 *double* interval = (end - start) / n;  
 *double* epsilon = get\_machine\_epsilon();  
 epsilon = epsilon \* pow(10, 16 - accuracy\_factor);  
 printf("Epsilon is: %.32f \n", epsilon);  
 *double* value\_by\_Tailor, value\_by\_Ctools;  
 *double* prev\_value\_by\_Tailor;  
 *int* n\_for\_Tailor;  
 *double* x\_for\_polynomial = start;  
 *for* (*int* i = 0; i <= n; i++) {  
 value\_by\_Ctools = func(x\_for\_polynomial);  
 value\_by\_Tailor = -1.0 / 5.0;  
 prev\_value\_by\_Tailor = 0;  
 n\_for\_Tailor = 1;  
 *int* count\_of\_iter = 0;  
 *while* (fabs(value\_by\_Tailor - prev\_value\_by\_Tailor) > epsilon && count\_of\_iter < 100){  
 n\_for\_Tailor++;  
 count\_of\_iter++;  
 prev\_value\_by\_Tailor = value\_by\_Tailor;  
 value\_by\_Tailor -= (pow(2, n\_for\_Tailor - 1) \* pow(x\_for\_polynomial, n\_for\_Tailor - 1)) / pow(5, n\_for\_Tailor);  
 printf("%d of Teilor: %.\*f\n", n\_for\_Tailor, accuracy\_factor, value\_by\_Tailor);  
 }  
 printf("x|%.2f|\t Tailor|%.\*f|\t C tools|%.\*f|\t count of iteration|%d|\t difference|%.\*f|\t\n", x\_for\_polynomial, accuracy\_factor + 1 , value\_by\_Tailor, accuracy\_factor + 1, value\_by\_Ctools, count\_of\_iter, accuracy\_factor + 1,  
 fabs(value\_by\_Ctools - value\_by\_Tailor));  
 x\_for\_polynomial += interval;  
 }  
}

# Переменные в программе:

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | За что отвечает |
| start | double | Начальное значение отрезка |
| end | double | Конечное значение отрезка |
| n | integer | Кол-во равных кусков отрезка [a;b] |
| accuracy\_factor | integer | Точность |
| interval | double | Разница между разделёнными частями отрезка |
| epsilon | double | Машинный эпсилон |
| value\_by\_Tailor | double | Значение k+1-ого члена по Тейлору |
| prev\_value\_by\_Tailor | double | Значение k-ого члена по Тейлору |
| value\_by\_Ctools | double | Значение вычисленное средствами Си |
| n\_for\_Tailor | integer | n-ый член Тейлора |
| count\_of\_iter | integer | Количество итераций (на 1 меньше последнего члена) |
| x\_for\_polynomial | double | Текущее значение x из отрезка для функций |

# Входные данные:

На вход с клавиатуры подаются два целых числа n (0≤n≤100) – число разбиений отрезка на равные части, accuracy\_factor (0≤ accuracy\_factor ≤16) — коэффициент для вычисления точности формулы Тейлора.

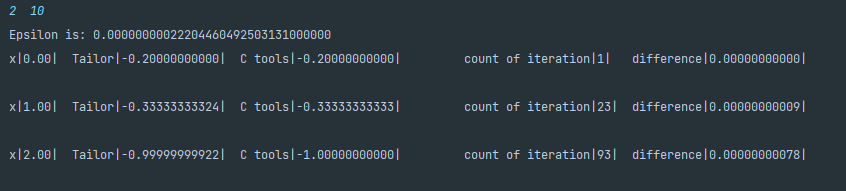
# Выходные данные

В начале программа выводит вычисленное нами значение машинного эпсилона, а затем N+1 строку, содержащую сведения о вычислении n-ого значения x с помощью ряда Тейлора и функций Си.

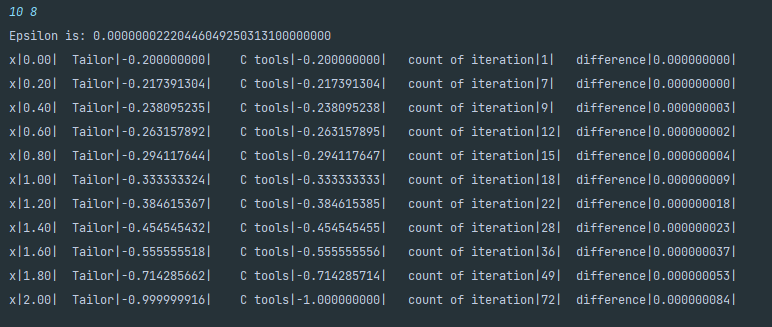
Каждая строка содержит значение x, для которого вычисляется функция, число, вычисленное с помощью формулы Тейлора, число, полученное с помощью встроенных функций языка, количество итераций, потребовавшихся для вычисления функции рядом Тейлора (последний член ряда), и разница двух полученных значений.

# Протокол исполнения и тесты

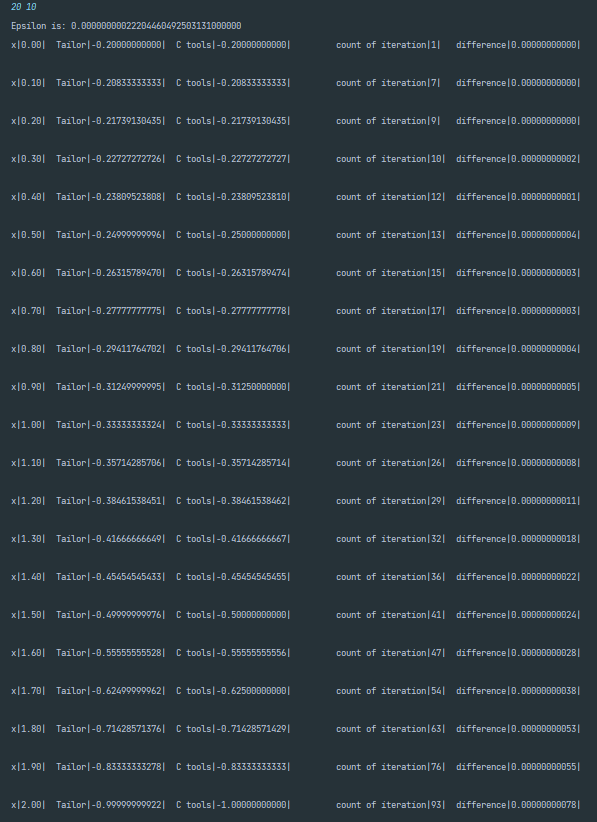
## Тест №1



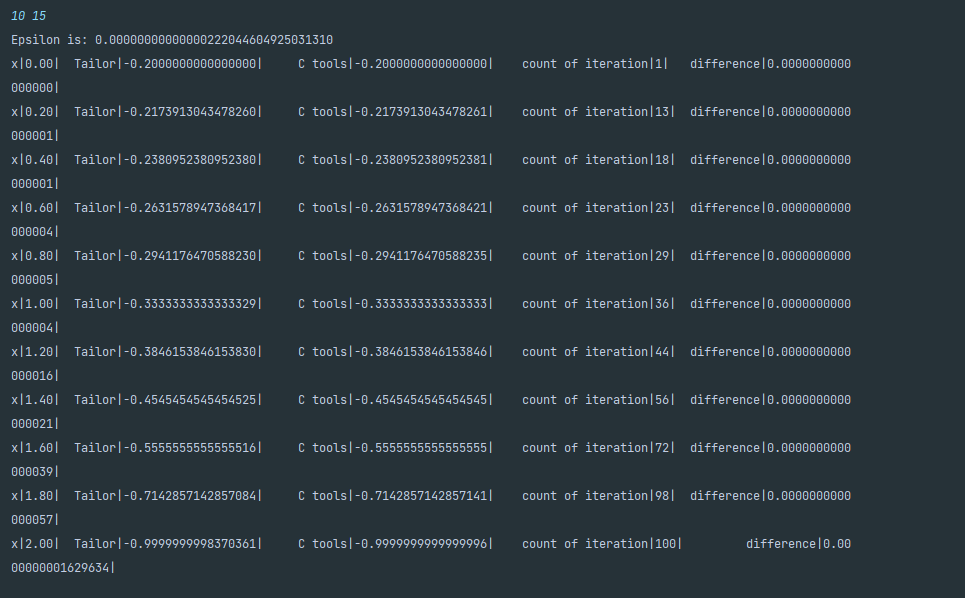
## Тест №2

****

## Тест №3



## Тест №4



# Вывод

В работе описано определение машинного эпсилон, описана формула Тейлора и составлен алгоритм реализации вычисления значения функции с заданной точностью для определённого заранее числа точек на отрезке. Было очень полезно и интересно использовать знания математического анализа на практике в программировании.

# Список литературы

1. <https://ru.wikipedia.org/wiki/Число_с_плавающей_запятой> - статья про представление чисел с плавающей точкой (запятой) в ПК.
2. <https://ru.wikipedia.org/wiki/IEEE_754-2008> - статья про стандарт IEEE.
3. <https://ru.wikipedia.org/wiki/Машинный_ноль> – машинный ноль и эпсилон.
4. <https://portal.tpu.ru/SHARED/k/KONVAL/Sites/Russian_sites/Calc1-ru/6/01.htm> - Ряд Тейлора.