МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ

ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное

учреждение высшего образования

«Московский Авиационный Институт

(Национальный Исследовательский Университет)»

Институт №8 «Компьютерные науки и прикладная математика»

Кафедра «Вычислительная математика и программирование»

**Курсовой проект**

по курсу “Фундаментальная информатика” 1 семестра

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

Выполнил Епифанов Е. В., №9 по списку

Группа М8О-112Б-22

Проверил Никулин С. П., каф. 806

Москва, 2022

1. **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 × k, 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 №17**



Изображение выглядит как стол

Автоматически созданное описание

1. **Метод решения:**

Программа высчитывает машинное эпсилон с помощью деления числа e, равному 1, на 2. Далее это значение эпсилон будет использоваться для отслеживания точности расчета суммы элементов последовательности Тейлора. Потом программа просит пользователя ввести число, которое равно числу подотрезков исходного отрезка. Из исходного отрезка берутся x для будущих вычислений в программе, подотрезки как раз нужны для точного определения этих x. Параллельно с этим поэтапно выводится таблица – ответ на задачу. В цикле считается значение функции от x и сумма ряда Тейлора от того же x. Каждая итерация – вывод новой строки таблицы. В конце пользователь получает законченную таблицу.

1. **Сведения о программе**

Программа написана на языке Си с помощью редактора Emacs в OC Linux. Редактор текстов: GNU Emacs 28.2.

Компилятор gcc Linux.

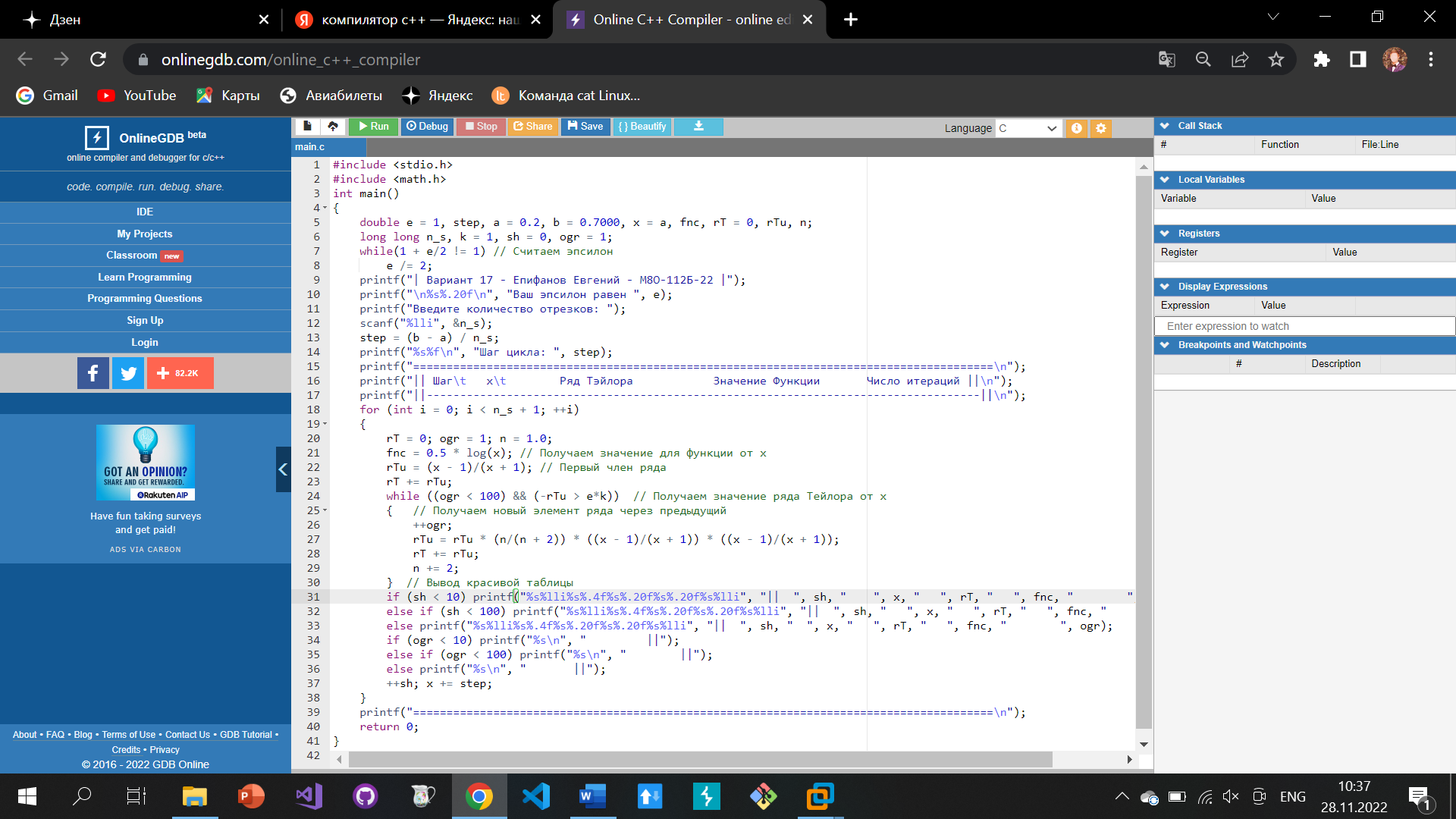
Операционная система семейства UNIX, наименование: Linux Ubuntu (22.04) 5.15.0-48-generic #54-Ubuntu SMP Fri Aug 26 13:26:29 UTC 2022 x86\_64 x86\_64 x86\_64 GNU/Linux.

Программа в моем случае умещается в 40 строк. Программа компилируется с помощью компилятора gcc.

1. **Функциональное назначение­­**

Программа предназначена для обработки вещественных и целых чисел (double и long long). Также в программе используются строковые переменные, но никакой обработке они не подвергаются. Ограничения по вводимым данным соответствуют допустимым размерам для вышеуказанных типов данных. Дополнительное ограничение – n\_s < 1000, на функционал программы не влияет. Числа, большие или равные 1000, незначительно портят косметику выводимой таблицы.

1. **Описание алгоритма**



**Программа в виде текста:**

#include <stdio.h>

#include <math.h>

int main()

{

double e = 1, step, a = 0.2, b = 0.7000, x = a, fnc, rT = 0, rTu, n;

long long n\_s, k = 1, sh = 0, ogr = 1;

while(1 + e/2 != 1) // Считаем эпсилон

e /= 2;

printf("| Вариант 17 - Епифанов Евгений - М8О-112Б-22 |");

printf("\n%s%.20f\n", "Ваш эпсилон равен ", e);

printf("Введите количество отрезков: ");

scanf("%lli", &n\_s);

step = (b - a) / n\_s;

printf("%s%f\n", "Шаг цикла: ", step);

printf("=======================================================================================\n");

printf("|| Шаг\t x\t Ряд Тэйлора Значение Функции Число итераций ||\n");

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

for (int i = 0; i < n\_s + 1; ++i)

{

rT = 0; ogr = 1; n = 1.0;

fnc = 0.5 \* log(x); // Получаем значение для функции от x

rTu = (x - 1)/(x + 1); // Первый член ряда

rT += rTu;

while ((ogr < 100) && (-rTu > e\*k)) // Получаем значение ряда Тейлора от x

{ // Получаем новый элемент ряда через предыдущий

++ogr;

rTu = rTu \* (n/(n + 2)) \* ((x - 1)/(x + 1)) \* ((x - 1)/(x + 1));

rT += rTu;

n += 2;

} // Вывод красивой таблицы

if (sh < 10) printf("%s%lli%s%.4f%s%.20f%s%.20f%s%lli", "|| ", sh, " ", x, " ", rT, " ", fnc, " ", ogr);

else if (sh < 100) printf("%s%lli%s%.4f%s%.20f%s%.20f%s%lli", "|| ", sh, " ", x, " ", rT, " ", fnc, " ", ogr);

else printf("%s%lli%s%.4f%s%.20f%s%.20f%s%lli", "|| ", sh, " ", x, " ", rT, " ", fnc, " ", ogr);

if (ogr < 10) printf("%s\n", " ||");

else if (ogr < 100) printf("%s\n", " ||");

else printf("%s\n", " ||");

++sh; x += step;

}

printf("=======================================================================================\n");

return 0;

}

**Алгоритм:**

1) Вычисление машинного эпсилон с помощью деления 1 на 2. Деление производится, пока полученное значение больше 0. Вывод полученного результата.

2) Запрос на ввод числа n\_s (кол-во подотрезков) от пользователя. Вывод шага программы.

3) 1-ый этап печати таблицы – шапка.

4) 1-ый цикл для расчёта значения функции от x. Начало получения суммы ряда Тейлора.

5) 2-ой цикл (while) для расчёта уже самой суммы ряда Тейлора. Каждый следующий элемент получается из предыдущего путем умножения его на шаг, высчитанный логически. В моем случае шаг – умножение на (x-1/x+1)^2 и на (n/n+2).

6) Анализ количества полученных строк, чтобы выводимая таблица выглядела презентабельно.

7) Вывод завершающей части таблицы – её дно.

*(В программе присутствуют комментарии, которые помогут дополнительно разобраться с алгоритмом)*

1. **Описание переменных**

|  |  |
| --- | --- |
| Константы | Значение |
| a (double) | 0.2 |
| b (double) | 0.7 |
| k (long long) | 1 |

a и b – границы отрезка, с которого берутся x. k – константа для корректировки ограничивающего значения эпсилон.

|  |  |
| --- | --- |
| Переменная | Описание |
| e (double) | Начальное значение эпсилон |
| step (double) | Шаг для x |
| x (double) | Значение, по которому будут найдены значения функции и суммы ряда Тейлора |
| fnc (double) | Значение функции по x |
| rT (double) | Сумма ряда Тейлора |
| rTu (double) | n-ный элемент ряда Тейлора |
| n (double) | Параметр для формулы ряда Тейлора |
| n\_s (long long) | Кол-во подотрезков отрезка [a, b] |
| sh (long long) | Шаг вывода для таблицы |
| ogr (long long) | Переменная для проверки ограничения по точности для суммы ряда Тейлора |

1. **Подпрограммы**

Подпрограммы в программе отсутствуют. Все тело программы находится в int (main).

1. **Входные данные**

Единственные входные данные – переменная n\_s. Ввод реализован через scanf("%lli", &n\_s);. Переменная заранее инициализирована. Ввод сопровождается пользовательским интерфейсом.

1. **Выходные данные**

В начале программы выводится плашка с данными о работе и её авторе. Далее будет выведен текст, определяющий переменную e (эпсилон). После выводится предложение ввести переменную n\_s. Затем поэтапно выводится таблица: шапка, строки с данными, дно. Выводится в основном строчный текст, не считая некоторых переменных, объявленных выше.

1. **Тесты**
2. Ввод: 5

evgeniy2@evgeniy2:~$ emacs krs3.c

evgeniy2@evgeniy2:~$ cc krs3.c -lm

evgeniy2@evgeniy2:~$ ./a.out

| Вариант 17 - Епифанов Евгений - М8О-112Б-22 |

Ваш эпсилон равен 0.00000000000000022204

Введите количество отрезков: 5

Шаг цикла: 0.100000

=======================================================================================

|| Шаг x Ряд Тэйлора Значение Функции Число итераций ||

||-----------------------------------------------------------------------------------||

|| 0 0.2000 -0.80471895621705025192 -0.80471895621705014090 40 ||

|| 1 0.3000 -0.60198640216296772643 -0.60198640216296805949 27 ||

|| 2 0.4000 -0.45814536593707760881 -0.45814536593707755330 20 ||

|| 3 0.5000 -0.34657359027997253209 -0.34657359027997269862 16 ||

|| 4 0.6000 -0.25541281188299524985 -0.25541281188299536087 13 ||

|| 5 0.7000 -0.17833747196936622359 -0.17833747196936622359 11 ||

=======================================================================================

Изображение выглядит как текст, снимок экрана, монитор, серебряный

Автоматически созданное описание

1. Ввод: 10

Изображение выглядит как текст, снимок экрана, монитор

Автоматически созданное описание

1. Ввод: 1

Изображение выглядит как текст, снимок экрана, монитор, серебряный

Автоматически созданное описание

1. Ввод: 321 (добавлено для примера отображения таблицы, включающей шаг таблицы, равный трехзначному числу)

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

1. **Дневник отладки**

Работа выполнена 21.11.2022 в период с 18:03 до 21:48. Критических ошибок не возникало, сначала была неточность работы программы, из-за чего выводились некорректные значения сумм ряда Тейлора – проблема была исправлена путем дополнительного обнуления переменных rT и rTu в цикле for.

26.11.2022, 23:20 – корректировка программы, добавлен 4 столбец с кол-вом итераций для ряда Тейлора для конкретного x.

28.11.2022, 10:04 – корректировка условия выхода из цикла while.

Многие ошибки, связанные с неточностью работы программы, легко заметить, так как в таблице должны выводиться относительно похожие значения – если присутствует ошибка, данные в столбцах будут сильно различаться. Для определения проблемы удобно использовать отслеживание значений переменных – так можно понять, на каком этапе программа выдала неверный ответ, и исправить проблему.

1. **Заключение**

Проанализировав таблицу, можно заметить, что значения ряда Тейлора имеют небольшие отличия от встроенной функции примерно после 16 знака после запятой (в идеальном случае, когда точность максимальная). Это означает, что, несмотря на точность данного метода задания функций, он не является полностью достоверным.

Такого рода программы не имеют прикладного применения, так как они требуют большого количества времени при компиляции и являются бессмысленными при наличии встроенных функций. В защиту программ можно сказать, что они дают пользователю представление о том, как задаются математические функции на языке Си и прочих языках программирования.

Во время выполнения данного задания я подробнее изучил язык программирования Си, получил навыки работы с математическими функциями и пользовательским интерфейсом. Также я улучшил умение работать с табулированием функций.