

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ
ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
“Московский Авиационный Институт
(Национальный Исследовательский Университет)”
Факультет информационных технологий и прикладной математики
Кафедра 806 “Вычислительная математика и программирование”

Курсовой проект
по курсу “Фундаментальная информатика”
1 семестр
Задание 3. Вещественный тип. Приближенные вычисления. Табулирование
функций

Студент: Сибирцев Р. Д.
Группа: М8О-108Б-2
Руководитель: Сахарин Н.А.
Дата: 08.01.23
Оценка:

Москва, 2023

ОГЛАВЛЕНИЕ

ОГЛАВЛЕНИЕ	2
ЗАДАЧА	3
ВАРИАНТ	3
ОБЩИЙ МЕТОД РЕШЕНИЯ	3
ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММЕ	4
НАЗНАЧЕНИЕ	4
ОПИСАНИЕ ЛОГИЧЕСКОЙ СТРУКТУРЫ	4
ЛИСТИНГ ПРОГРАММЫ	6
ДНЕВНИК ОТЛАДКИ	9
ВЫВОД	9

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 $\varepsilon * k$, 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

№	ряд	a	b	функция
17	$\frac{x-1}{x+1} + \frac{1}{3}\left(\frac{x-1}{x+1}\right)^3 + \dots + \frac{1}{2n+1}\left(\frac{x-1}{x+1}\right)^{2n+1}$	0.2	0.7	$\frac{1}{2}\ln(x)$

ОБЩИЙ МЕТОД РЕШЕНИЯ

Вычислить значения функции в некоторой точке на отрезке от 0.2 до 0.7 двумя способами.

- 1) При помощи использования программных средств, встроенных в стандартную математическую библиотеку языка Си “math.h”.
- 2) При помощи ряда Тейлора.

ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММЕ

Аппаратное обеспечение: ПК

Операционная система: Linux , Windows10

Язык и система программирования: C, VSCode

Компиляция программы: gcc -std=c2x kp3.c -lm

Вызов программы: ./a.out

НАЗНАЧЕНИЕ

Программа предназначена для высокоточного вычисления вещественного значения трансцендентных функций в алгебраической форме с использованием ряда Тейлора и при помощи встроенных программных функций библиотеки языка Си.

ОПИСАНИЕ ЛОГИЧЕСКОЙ СТРУКТУРЫ

Программа вычисляет значение функции в данной точке при помощи разложения по ряду Тейлора и с использованием программных средств языка программирования СИ. Ряд Тейлора описывается функцией, которая вычисляет слагаемые ряда, складывает их, по количеству слагаемых не превысит 100 или значение одного из них не станет совсем мало (меньше ϵ по модулю). В конце выводится таблица со значениями аргумента, функции, вычисленной с помощью ряда Тейлора и с использованием программной библиотеки и номером итерации.

ОПИСАНИЕ ПЕРЕМЕННЫХ И ФУНКЦИЙ

Имя	Тип	Назначение
eps	double	Машинный эпсилон
n	int	Максимальное количество итераций
a	double	Левая граница отрезка
b	double	Правая граница отрезка
step	double	шаг
num	int	Количество символов после точки в машинном эпсилоне
res1	double	Результат вычислений с помощью ряда Тейлора
res2	double	Результат подсчета функции
N	int	Количество шагов

ЛИСТИНГ ПРОГРАММЫ

```
#include <stdio.h>
#include <math.h>

typedef struct {
    double res;
    int N;
} result;

double mech_eps(void) {
    double e = 1.0f;

    while (1.0f + e / 2.0f > 1.0f)
        e /= 2.0f;
    return e;
}

int number_d(double x) {
    int k = 0;
    while ((int) x == 0) {
        k += 1;
        x *= 10;
    }
    return k;
}

result teylor(double x, int n, double eps) {
    double res = 0.0, res_old;
    double r, r2 ;
    result A;

    for (int i = 0; i < n; ++i) {
```

```

    r = (2 * i + 1);
    r = 1 / r;
    r = r * pow(((x - 1) / (x + 1)), (2 * i + 1));
    res_old = res;
    res = res + r;
    if (res_old - res < eps) {
        A.res = res;
        A.N = i;
        return A;
    }
}
A.res = res;
A.N = n;
return A;
}

```

```

double func(double x) {
    double res;

    res = 0.5 * log(x);
    return res;
}

```

```

int main(void) {
    double eps = mech_eps();
    int n = 100;
    double a = 0.2, b = 0.7;
    int num = number_d(eps);
    double res1, res2, step;
    int N;
    scanf("%d", &N);
    step = (b - a) / N;

    printf("%.*fn", num, eps);
}

```

```

for (double x = a; b - x + step >= eps; x = x + step) {
    result res1 = teylor(x, n, eps);
    double res2 = func(x);
    printf("%.2f %.*f %.*f %d\n", x, num, res1.res, num, res2, res1.N);
}
return 0;
}

```

ВХОДНЫЕ ДАННЫЕ

N = 11

РЕЗУЛЬТАТ

Linux

0.000000000000000002

0.20	-0.8047189562170503	-0.8047189562170501	39
0.25	-0.6931471805599456	-0.6931471805599453	32
0.30	-0.6019864021629677	-0.6019864021629681	26
0.35	-0.5249110622493387	-0.5249110622493389	22
0.40	-0.4581453659370776	-0.4581453659370776	19
0.45	-0.3992538481088859	-0.3992538481088859	17
0.50	-0.3465735902799725	-0.3465735902799727	15
0.55	-0.2989185003778104	-0.2989185003778103	13
0.60	-0.2554128118829952	-0.2554128118829954	12
0.65	-0.2153914580462272	-0.2153914580462271	11
0.70	-0.1783374719693661	-0.1783374719693661	10

Windows 10

0.0000000000000000001

0.20	-0.8047189562170501400	-0.8047189562170501400	41
0.25	-0.6931471805599456200	-0.6931471805599452900	33
0.30	-0.6019864021629677300	-0.6019864021629680600	27
0.35	-0.5249110622493387000	-0.5249110622493389200	23
0.40	-0.4581453659370775500	-0.4581453659370775500	20

0.45 -0.3992538481088858700 -0.3992538481088858700 18
 0.50 -0.3465735902799725900 -0.3465735902799727000 16
 0.55 -0.2989185003778103700 -0.2989185003778102600 14
 0.60 -0.2554128118829952500 -0.2554128118829953600 13
 0.65 -0.2153914580462271400 -0.2153914580462271100 12
 0.70 -0.1783374719693661400 -0.1783374719693661400 10

ДНЕВНИК ОТЛАДКИ

Дата	Место	Событие	Действие по исправлению

ВЫВОД

В результате выполненного курсового проекта были получены результаты работы программы в разных операционных системах. Сравнивая эти результаты можно заметить, что машинный эпсилон в Windows 10 на несколько порядков меньше, чем в Linux. Из этого можно сделать вывод, что в операционной системе Windows можно проводить более точные расчеты. Также можно заметить, что результаты вычислений с помощью ряда Тейлора и с помощью стандартных функций различаются, начиная с 15-16 знака после запятой. Это обусловлено тем, что в языке программирования СИ тип данных double может без погрешности хранить только 16 знаков после запятой.