Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра 806 “Вычислительная математика и программирование”

**Курсовая работа**

**по курсу “Архитектура компьютера”**

**1 семестр**

**Задание 3. Вещественный тип.**

**Приближенные вычисления.**

**Табулирование функций**

Студент: Старостина А.А.

Группа: М8О-108Б-22

№ по списку: 19

Руководитель: Сахарин Н.А.

Дата: 08.01.2023

Оценка: \_\_\_\_\_\_

г. Москва, 2023

**Содержание**

ЗАДАНИЕ ……………………………………………………………………. 3

ВАРИАНТ ………………………………………………………………..&..... 3

ОБЩИЙ МЕТОД РЕШЕНИЯ ……………………………………………… 3

ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММЕ ……………………………………. 3

ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ……………………………………. 3

ОПИСАНИЕ ЛОГИЧЕСКОЙ СТРУКТУРЫ ……………………………… 4

ОПИСАНИЕ ПЕРЕМЕННЫХ, КОНСТАНТ И ПОДПРОГРАММ ………. 4

ПРОТОКОЛ ………………………………………………………………….. 5

ВХОДНЫE ДАННЫЕ ……………………………………………………..... 6

ВЫХОДНЫЕ ДАННЫЕ ………………………………………………...…... 7

ВЫВОДЫ …………………………………………………………………….. 8

**1. Задание**

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.

**2. Вариант 19**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Ряд | a | b | Функция |
| 19 |  | 0.1 | 0.6 |  |

**3. Общий метод решения**

Вычисление значения функции в некоторой точке на отрезке от 0.1 до 0.6 двумя способами:

1: использование программных средств, встроенных в стандартную математическую библиотеку языка Си “math.c”.

2 : при помощи ряда Тейлора.

**4. Общие сведения о программе**

Аппаратное обеспечение: домашний ноутбук

Операционная система: Linux Ubuntu, версия 22.04.1 LTS

Язык и система программирования: С, GNU

Местонахождение файлов: /home/ann

Компиляция программы: gcc -lm kp3.c

Вызов программы: ./a.out

**5. Функциональное назначение**

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

**6. 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 мы преобразуем в функцию, которая вычисляет слагаемые ряда. Далее мы складываем полученные слагаемые, пока их количество не превысит 100 или значение одного из них не станет совсем мало (меньше ε\*k по модулю). В конце мы выводим таблицу с значением аргумента, значением функции, вычисленным с помощью ряда Тейлора и с использованием программной библиотеки, и номером итерации.

**7. Описание переменных, констант и подпрограмм**

|  |  |  |
| --- | --- | --- |
| **Функция** | **Входные аргументы** | **Описание** |
| machine\_epsilon | long double x | Функция для подсчета машинного ε. Сравниваем 1+ε с 1. Последнее число, при стремлении ε к нулю, при котором 1+ε > 1 и будет машинным ε |
| function | long double x | Вычисляет значение входной функции |
| taylor | long double x, int n, | Считает сумму ряда по формуле Тейлора |
| factorial | long long n | Вычисляет факториал |

Таблица 1. Описание функций программы

|  |  |
| --- | --- |
| **Переменная** | **Значение** |
| long double abs\_epsilon | Машинный эпсилон(абсолютный) |
| long double otn\_epsilon | Корень из машинного эпсилона, выбрали сами это число для сравнения(это относительный эпсилон) |
| const long double k | Эмпирический коэффициент для эпсилон |
| int MAX\_ITER | Максимальное число итераций |
| long double a,b | Границы отрезка |
| long double step | Количество отрезков |
| long double n | Количество частей, на которые разбивается отрезок [a, b] |
| long double result, sum | Сумма ряда |
| long double x | Значение аргумента функции |
| int n | Текущая итерация |
| const long double e | Число е |
| long double ans | Результат факториала |

Таблица 2. Описание переменных

**8. Протокол**

Код программы:

#include <stdio.h>

#include <math.h>

const long double e = 2.71828182846;

int MAX\_ITER = 100;

long double machine\_epsilon() {

    long double eps = 1;

    while (1<1+eps)

        eps /= 2;

    return eps;

}

int factorial(long long n) {

    long double ans = 1;

    for (long long i = 2; i <= n; i += 1)

        ans \*= i;

    return ans;

}

long double func(long double x) {

    return ( (pow(e,x) + pow(e,-x)) /2 );

}

long double taylor(long double x, int n) {

    long double sum = 0;

    for (long double i = 0.0; i <= n; i += 1)

        sum += (pow(x,2\*i)/factorial(2\*i));

    return sum;

}

int main() {

    long double a = 0.1, b = 0.6, step, result;

    int n;

    printf("Enter the iterations: ");

    scanf("%d",&n);

    printf("\n");

    step = (b-a)/n;

    long double abs\_epsilon = machine\_epsilon();

long double otn\_epsilon = sqrt(abs\_epsilon);

    printf("Machine epsilon: %.40Lf\n",abs\_epsilon);

    printf("Taylor series value table for f(x) = ch x\n");

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

    printf("|  x    |       sum of row       |        function        |   iter |\n");

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

    for (long double x = a; x <= b; x += step) {

                for (n = 0; n < MAX\_ITER; n += 1) {

            result = taylor(x, n);

            if (fabs(result) <= fmax(otn\_epsilon \* fabs(result), abs\_epsilon))

                break;

        }

        printf("| %.3Lf | %.20Lf | %.20Lf |  %d   |\n",x,result,function(x),n);

        result = 0.0;

    }

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

    return 0;

}

**9. Входные данные**

На вход подается одно число n (4, 9, 13)

**10. Выходные данные**

ann@ann:~$ gcc kp3.c -lm

ann@ann:~$ ./a.out

Enter the iterations: 4

Machine epsilon: 0.0000000000000000000000000000000000962965

Taylor series value table for f(x) = ch x

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  x    |       sum of row       |        function        |   iter |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| 0.100 | 1.00500416805580359996 | 1.00500416805580705670 |  100   |

| 0.225 | 1.02541946747562069597 | 1.02541946747563850195 |  100   |

| 0.350 | 1.06187781915598554806 | 1.06187781915602919014 |  100   |

| 0.475 | 1.11494962697541900804 | 1.11494962697548372077 |  100   |

| 0.600 | 1.18546521824282886838 | 1.18546521824240169707 |  100   |

|\_\_\_\_ \_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_|

ann@ann:~$ ./a.out

Enter the iterations: 9

Machine epsilon: 0.0000000000000000000000000000000000962965

Taylor series value table for f(x) = ch x

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  x    |       sum of row       |        function        |   iter |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| 0.100 | 1.00500416805580359996 | 1.00500416805580705670 |  100   |

| 0.156 | 1.01212318180608733352 | 1.01212318180609583962 |  100   |

| 0.211 | 1.02236683607593229389 | 1.02236683607594791567 |  100   |

| 0.267 | 1.03576675521480039484 | 1.03576675521482552966 |  100   |

| 0.322 | 1.05236430763619390785 | 1.05236430763623101114 |  100   |

| 0.378 | 1.07221073353078743111 | 1.07221073353083795610 |  100   |

| 0.433 | 1.09536730305584677419 | 1.09536730305591012424 |  100   |

| 0.489 | 1.12190550548930526999 | 1.12190550548936585074 |  100   |

| 0.544 | 1.15190726993247685265 | 1.15190726993245551313 |  100   |

| 0.600 | 1.18546521824282886838 | 1.18546521824240169707 |  100   |

|\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_|

ann@ann:~$ ./a.out

Enter the iterations: 13

Machine epsilon: 0.0000000000000000000000000000000000962965

Taylor series value table for f(x) = ch x

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  x    |       sum of row       |        function        |   iter |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| 0.100 | 1.00500416805580359996 | 1.00500416805580705670 |  100   |

| 0.138 | 1.00960112319658304086 | 1.00960112319658978919 |  100   |

| 0.177 | 1.01569175524130606770 | 1.01569175524131716237 |  100   |

| 0.215 | 1.02328507511142135520 | 1.02328507511143773634 |  100   |

| 0.254 | 1.03239231691340289173 | 1.03239231691342592256 |  100   |

| 0.292 | 1.04302695455929943150 | 1.04302695455932981083 |  100   |

| 0.331 | 1.05520472170101423210 | 1.05520472170105339416 |  100   |

| 0.369 | 1.06894363500780754697 | 1.06894363500785605225 |  100   |

| 0.408 | 1.08426402082146028772 | 1.08426402082151840389 |  100   |

| 0.446 | 1.10118854522853490890 | 1.10118854522860010192 |  100   |

| 0.485 | 1.11974224759422623283 | 1.11974224759428864218 |  100   |

| 0.523 | 1.13995257760741697584 | 1.13995257760744506115 |  100   |

| 0.562 | 1.16184943589175203644 | 1.16184943589166156386 |  100   |

| 0.600 | 1.18546521824282886838 | 1.18546521824240169707 |  100   |

|\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_|

**11. Вывод**

В ходе выполнения данной работы были получены навыки вычисления заданной функции при помощи разложения по ряду Тейлора и с помощью встроенной библиотеки “math.c”, было изучено вычисление и использование машинного эпсилон., как абсолютного, так и относительного. При сравнении данных в таблице, можно увидеть, что значения различаются приблизительно после 13 знака после запятой. Это происходит из-за ограниченного количества памяти в компьютере для представлении вещественных чисел, что приводит к тому, что в окрестностях границ данного диапазона возникают погрешности.

И что важно, вычисление трансцендентных функций при помощи формулы Тейлора не применяется на практике ввиду большой ресурсоемкости и значительной погрешности.