Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное   
учреждение высшего образования

Национальный исследовательский Нижегородский государственный университет им. Н.И. Лобачевского

Институт информационных технологий, математики и механики

**Отчет по лабораторной работе**

**«Ряды Тейлора»**

**Выполнила**:

студентка группы 3822Б1ПМ1

Шарутина В. Ф.

**Проверил**:

преподаватель каф. МОСТ,

Волокитин В.Д.

Нижний Новгород

2023

**Содержание**

[Постановка задачи 3](#_Toc26962563)

[Метод решения 4](#_Toc26962564)

[Руководство пользователя 5](#_Toc26962565)

[Описание программной реализации 6](#_Toc26962566)

[Подтверждение корректности 9](#_Toc26962567)

[Результаты экспериментов 10](#_Toc26962568)

[Заключение 11](#_Toc26962569)

[Приложение 12](#_Toc26962562)

# Постановка задачи

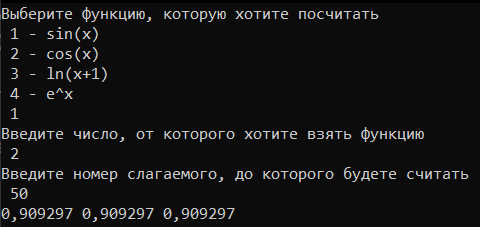
В данной лабораторной работе было необходимо написать программу на языке программирования Си, состоящую из функций, которые вычисляют слагаемые отдельно для каждой математической функции(sin(x), cos(x), ln(x+1), exp(x)), и из функций прямого и обратного суммирования этих слагаемых, чтобы в результате выходило значение рядов Тейлора, а также нужно было провести эксперимент на вычисление погрешности между реальными значениями математических функций и значениями, получившимися в результате прямой и обратной суммы, сравнить эти результаты и сделать вывод.

# Метод решения

В данной программе были реализованы функции, вычисляющее слагаемые рядов Тейлора для sin(x), cos(x), ln(x+1), exp(x). В начале считались слагаемые для прямой суммы, каждый следующий из которых выражался через предыдущий, потом считались слагаемые для обратной суммы, каждый предыдущий из который выражался через последующий. Помимо этого, были реализованы функция прямого суммирования, где складывались слагаемые, начиная с первого, и функция обратного суммирования, где уже складывались слагаемые, начиная с последнего.

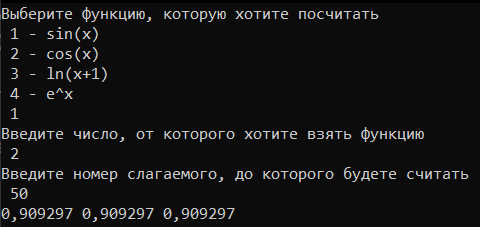
# Руководство пользователя

Когда пользователь запустит программу, перед ним на консоли появится подобие меню, в котором можно выбрать одну из четырёх представленных математических функций. После чего ему необходимо будет нажать на клавишу с номером функции, результат которой он хочет посчитать. Далее пользователь должен будет ввести значение, от которого будет браться функция, и ввести номер слагаемого, до которого будет воспроизводиться функция суммирование прямое и обратное.



**Рис.1 Меню и ввод значения и кол-ва слагаемых**

Затем пользователь после нажатия клавиши «enter» увидит на консоли получившиеся значения в результате прямого суммирования, обратного и значение получившееся, в результате использования встроенной математической функции(math.h). После чего программа прекращает работу, и при желании поменять функцию, вводимое значение или количество слагаемых её нужно перезапустить.



**Рис.2 Результат работы программы**

# Описание программной реализации

Программа состоит из одного файла teilor.cpp, в котором есть 12 функций, одна из которых main.

* **float sinslg(float prev, int n, float x)**

Функция принимает на входе числа типа float, первое из которых - значение предыдущего элемента, второе - число, от которого будет считаться функция, и номер элемента, значение которого мы хотим посчитать, типа int. Она вычисляет член ряда Тейлора для sin(x) при прямом суммировании по формуле: an= ((-1) \* x \* x \* an-1) / ((2 \* n) \* (2 \* n + 1)) и возвращает его получившееся значения.

* **float sinslgrev(float next, int n, float x)**

Функция принимает на входе числа типа float, первое из которых - значение последнего элемента, второе - число, от которого будет считаться функция, и номер элемента, значение которого мы хотим посчитать, типа int. Она вычисляет члены ряда Тейлора для sin(x) при обратном суммировании по формуле: an-1= (-1) \* an \* (2 \* n) \* (2 \* n + 1) / (x \* x) и возвращает его получившееся значения.

* **float cosslg(float prev, int n, float x)**

Функция принимает на входе числа типа float, первое из которых - значение предыдущего элемента, второе - число, от которого будет считаться функция, и номер элемента, значение которого мы хотим посчитать, типа int. Она вычисляет члены ряда Тейлора для cos(x) при прямом суммировании по формуле: an= ((-1) \* (x \* x) \* an-1) / (2 \* n \* (2 \* n - 1)) и возвращает его получившееся значения.

* **float cosslgrev(float next, int n, float x)**

Функция принимает на входе числа типа float, первое из которых - значение последнего элемента, второе - число, от которого будет считаться функция, и номер элемента, значение которого мы хотим посчитать, типа int. Она вычисляет члены ряда Тейлора для cos(x) при обратном суммировании по формуле: an-1= ((-1) \* an \* 2 \* n \* (2 \* n - 1) / (x \* x)) и возвращает его получившееся значения.

* **float lnslg(float prev, int n, float x)**

Функция принимает на входе числа типа float, первое из которых - значение предыдущего элемента, второе - число, от которого будет считаться функция (в данном случае необходимо вводить значения |x|<1), и номер элемента, значение которого мы хотим посчитать, типа int. Она вычисляет члены ряда Тейлора для ln(x+1) при прямом суммировании по формуле: an= ((-1) \* x \* an-1 \* n) / (n + 1) и возвращает его получившееся значения.

* **float lnslgrev(float next, int n, float x)**

Функция принимает на входе числа типа float, первое из которых - значение последнего элемента, второе – число (в данном случае необходимо вводить значения |x|<1) , от которого будет считаться функция, и номер элемента, значение которого мы хотим посчитать, типа int. Она вычисляет члены ряда Тейлора для ln(x+1) при обратном суммировании по формуле: an-1= ((-1) \* an \* (n + 1) / (n \* x)) и возвращает его получившееся значения.

* **float expslg(float prev, int n, float x)**

Функция принимает на входе числа типа float, первое из которых - значение предыдущего элемента, второе - число, от которого будет считаться функция, и номер элемента, значение которого мы хотим посчитать, типа int. Она вычисляет члены ряда Тейлора для exp(x) при прямом суммировании по формуле: an= an-1 \* x / n и возвращает его получившееся значения.

* **float expslgrev(float next, int n, float x)**

Функция принимает на входе числа типа float, первое из которых - значение последнего элемента, второе - число, от которого будет считаться функция, и номер элемента, значение которого мы хотим посчитать, типа int. Она вычисляет члены ряда Тейлора для exp(x) при обратном суммировании по формуле: an-1= ( an \* n / x) и возвращает его получившееся значения.

* **float smfn(float x, int n, float(\*slg)(float, int, float))**

Функция принимает на входе значение x типа float, от которого будет браться sin(x), cos(x), ln(x+1) или exp(x), количество слагаемых типа int, а также указатель на функцию для вычисления слагаемых ряда Тейлора конкретной функции (в данном случае будут использоваться функции, вычисляющие элементы для прямого суммирования). Внутри имеется цикл, в котором подсчитывается каждый следующий элемент, а потом они складываются друг с другом, начиная с первого, то есть вычисляется прямая сумма, чьё значение возвращает данная функция.

* **float lastslg(float x, int n, float(\*slg)(float, int, float))**

Функция принимает на входе значение x типа float, от которого будет браться sin(x), cos(x), ln(x+1) или exp(x), количество слагаемых типа int, а также указатель на функцию для вычисления слагаемых ряда Тейлора конкретной функции (в данном случае будут использоваться функции, вычисляющие элементы для прямого суммирования). Внутри имеется цикл, который вычисляет значение последнего элемента (в зависимости от числа слагаемых, который ввёл пользователь), в результате функция будет возвращать это значение.

* **float smfnrev(float x, int n, float(\*slgrev)(float, int, float), float prev)**

Функция принимает на входе значение x типа float, от которого будет браться sin(x), cos(x), ln(x+1) или exp(x), количество слагаемых типа int, указатель на функцию для вычисления слагаемых ряда Тейлора конкретной функции (в данном случае будут использоваться функции, вычисляющие элементы для обратного суммирования), а также значение последнего элемента типа float. Внутри имеется цикл, в котором подсчитывается каждый предыдущий элемент, а потом они складываются друг с другом, начиная с последнего, то есть вычисляется обратная сумма, чьё значение возвращает данная функция.

* **int main()** - основная функция, которая ничего на входе не принимает и в конце возвращает по сути 0. Она отвечает за выбор функции и её работу, а также вывод результата прямой, обратной суммы и реального значения.

# Подтверждение корректности

В данной программе для подтверждения корректности используются встроенные математические функции из math.h, которые принимают число, от которого необходимо узнать значение функции, после чего можно сравнить реальное значение с получившимся, в результате прямого и обратного суммирования членов ряда Тейлора определённой функции.

# Результаты экспериментов

Суть эксперимента в данной работе заключается в том, чтобы сравнить полученные результаты при подстановке различных точек в функции, вычисляющие результаты разными способами суммирования, с реальными значениями математических функций.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | кол-во слагаемых | значение x | прямая сумма | обратная сумма | реальное значение | относительная погрешность при прямой сумме | абсолютная погрешность при прямой сумме | относительная погрешность при обратной сумме | абсолютная погрешность при прямой сумме |
|  | 10 | 2 | 0,909297 | 0,909297 | 0,909297 | -4,69402E-07 | -4,26826E-07 | -4,69402E-07 | -4,2683E-07 |
| sin(x) | 30 | 15 | 0,650288 | 0,650288 | 0,650288 | 2,45803E-07 | 1,59843E-07 | 2,45803E-07 | 1,5984E-07 |
|  | 60 | 26 | 0,762559 | 0,762558 | 0,762558 | 7,20627E-07 | 5,4952E-07 | -5,90748E-07 | -4,5048E-07 |
|  | 15 | 5 | 0,283662 | 0,283662 | 0,283662 | -6,53817E-07 | -1,85463E-07 | -6,53817E-07 | -1,8546E-07 |
| cos(x) | 50 | 25 | 0,991202 | 0,991203 | 0,991203 | -8,19069E-07 | -8,11863E-07 | 1,89806E-07 | 1,8814E-07 |
|  | 100 | 30 | 0,154191 | 0,154331 | 0,154251 | -0,000391892 | -6,04499E-05 | 0,000515717 | 7,955E-05 |
|  | 17 | -0,64 | -1,02161 | -1,02161 | -1,02165 | -4,52674E-05 | 4,62475E-05 | -4,52674E-05 | 4,6248E-05 |
| ln(x+1) | 40 | 0,03 | 0,029559 | 0,029559 | 0,029559 | 6,69034E-06 | 1,97758E-07 | 6,69034E-06 | 1,9776E-07 |
|  | 30 | 0,7 | 0,530628 | 0,530628 | 0,530628 | -4,73141E-07 | -2,51062E-07 | -4,73141E-07 | -2,5106E-07 |
|  | 10 | 0,5 | 0,606531 | 0,606531 | 0,606531 | 5,61039E-07 | 3,40287E-07 | 5,61039E-07 | 3,4029E-07 |
| exp(x) | 13 | 2 | 7,389055 | 7,389055 | 7,389056 | -1,48724E-07 | -1,09893E-06 | -1,48724E-07 | -1,0989E-06 |
|  | 21 | 5,5 | 244,6918 | 244,6918 | 244,6919 | -3,72976E-07 | -9,12642E-05 | -3,72976E-07 | -9,1264E-05 |

**Рис.3 Результаты измерений**

Из таблицы можно увидеть, что два представленных способа не всегда дают точные результаты, но при этом погрешность не так велика. Кроме этого, можно заметить, что при достаточно больших значениях подставляемых точек обратное суммирование выдаёт более точный результат, чем прямое.

# Заключение

В результате данной работы была выполнена выше поставленная цель: была реализована программа на языке программирования Си, содержащая в себе алгоритмы, вычисляющие значения рядов Тейлора различных функций для прямой суммы и обратной, а также был проведён эксперимент, направленный на сравнение получившихся значений с реальными.

# Приложение

#include "stdio.h"

#include "math.h"

#include "stdlib.h"

#include "locale.h"

float sinslg(float prev, int n, float x)

{

float next;

if (n == 0) return x;

else

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

return next;

}

float sinslgrev(float next, int n, float x)

{

float prev;

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

return prev;

}

float cosslg(float prev, int n, float x)

{

float next;

if (n == 0) return 1;

else

next = ((-1) \* (x \* x) \* prev) / (2 \* n \* (2 \* n - 1));

return next;

}

float cosslgrev(float next, int n, float x)

{

float prev;

prev = ((-1) \* next \* 2 \* n \* (2 \* n - 1) / (x \* x));

return prev;

}

float lnslg(float prev, int n, float x)

{

float next;

if (n == 0) return x;

else

next = ((-1) \* x \* prev \* n) / (n + 1);

return next;

}

float lnslgrev(float next, int n, float x)

{

float prev;

prev = ((-1) \* next \* (n + 1) / (n \* x));

return prev;

}

float expslg(float prev, int n, float x)

{

float next;

if (n == 0) return 1;

else

next = prev \* x / n;

return next;

}

float expslgrev(float next, int n, float x)

{

float prev;

prev = ( next \* n / x);

return prev;

}

float smfn(float x, int n, float(\*slg)(float, int, float))

{

float next = 0;

float sm = 0;

for (int i = 1; i <= n; i++)

{

next = slg(next, i-1, x);

sm += next;

}

return sm;

}

float lastslg(float x, int n, float(\*slg)(float, int, float))

{

float last=0;

float k = 0;

for (int i = 1; i <= n; i++)

{

last = slg(last, i-1, x);

k = last;

}

return k;

}

float smfnrev(float x, int n, float(\*slgrev)(float, int, float), float prev)

{

float sm = 0;

for (int i = n; i > 0; i--)

{

sm += prev;

prev = slgrev(prev,i-1,x);

}

return sm;

}

int main()

{

int k, o;

float x;

float last;

setlocale(LC\_ALL, "Rus");

printf("Выберите функцию, которую хотите посчитать\n ");

printf("1 - sin(x)\n ");

printf("2 - cos(x)\n ");

printf("3 - ln(x+1)\n ");

printf("4 - e^x\n ");

scanf\_s("%d", &k);

printf("Введите число, от которого хотите взять функцию\n ");

scanf\_s("%lf", &x);

printf("Ввeдите номер слагаемого, до которого будете считать\n ");

scanf\_s("%d", &o);

if (k == 1)

{

printf("%lf ", smfn(x, o, sinslg));

last = lastslg(x, o, sinslg);

printf("%lf ", smfnrev(x, o, sinslgrev, last));

printf("%lf ", sin(x));

}

if (k == 2)

{

printf("%lf ", smfn(x, o, cosslg));

last = lastslg(x, o, cosslg);

printf("%lf ", smfnrev(x, o, cosslgrev, last));

printf("%lf ", cos(x));

}

if (k == 3)

{

printf("%lf ", smfn(x, o, lnslg));

last = lastslg(x, o, lnslg);

printf("%lf ", smfnrev(x, o, lnslgrev, last));

printf("%lf ", log(x + 1));

}

if (k == 4)

{

printf("%lf ", smfn(x, o, expslg));

last = lastslg(x, o, expslg);

printf("%lf ", smfnrev(x, o, expslgrev, last));

printf("%lf ", exp(x));

}

}