МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное   
образовательное учреждение высшего образования  
«Самарский национальный исследовательский университет   
имени академика С.П. Королева»

(Самарский университет)

Институт информатики и кибернетики

Кафедра технической кибернетики

ЛАБОРАТОРНАЯ РАБОТА № 2

**Вычисление суммы ряда**

по курсу   
Основы программирования

Группа 6101

Студент А. И. Афанасенко

(*подпись*)

Преподаватель,

асс. каф. В.Д. Зайцев

(*подпись*)

Самара 2023

Задание

Задание на лабораторную работу приведено в таблице 1.

**Таблица 1 –** Задание на ЛР № 2

|  |  |
| --- | --- |
| Основное задание на ЛР | Реализовать консольное приложение, предназначенное для приближённого вычисления суммы бесконечного числового ряда, выбранного согласно номеру варианта. **Входные данные**: параметр x и параметр α, характеризующий точность вычисления ряда. **Выходные данные**: номер итерации (n), последний просуммированный член ряда (an), текущая частичная сумма (Sn), точность вычисления заданной текущей частичной суммы (αn). |

Вариант для задания приведен в таблице 2.

**Таблица 2 –** Вариант основного задания на ЛР № 2

|  |  |
| --- | --- |
| Вариант 9 |  |

Ход выполнения работы

В рамках выполнения задания я сначала написал функции чтения ввода от пользователя readDouble и readDoubleBetween0and1.

Обе функции читают строку из консоли и затем используют третью функцию parseRealNum, чтобы преобразовать строку в число типа double.

Функция parseRealNum использует функцию из стандартной библиотеки stod чтобы получить double из строки. stod может выкинуть ошибки конвертации, которые затем обрабатываются конструкцией try...catch. Таким образом, parseRealNum обрабатывает случаи, когда вместо числа вводится обычная строка или число вводися слишком большое, как показано на Рисунке 1.

Интересной особенностью stod является то, что он может без ошибок конвертировать такую строку как, например, 140afd – когда за числами следуют буквы. Для того чтобы выкидывать ошибку в таком случае, parseRealNum использует регулярное выражение [+-]?[0-9]+([,.][0-9]+ чтобы проверить, что строка является числом и ничем больше.

Функция readDoubleBetween0and1 также добавляет проверку на то, чтобы parseRealNum возвращал число между -1 и 1, согласно ограничениям в задании.

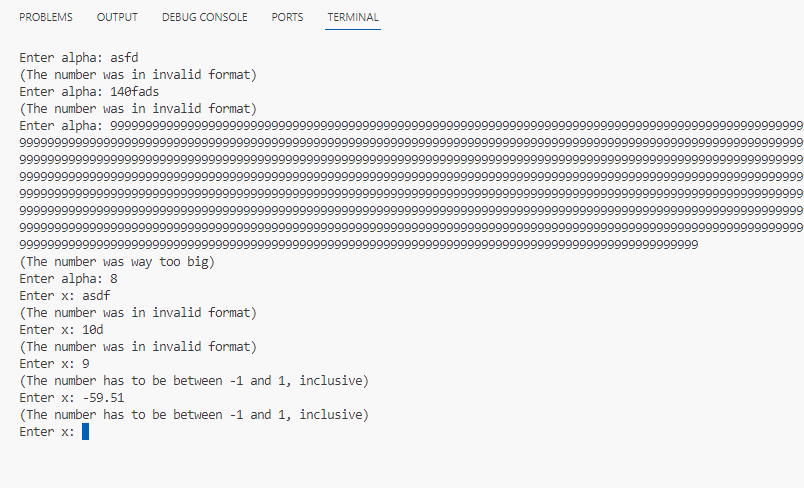
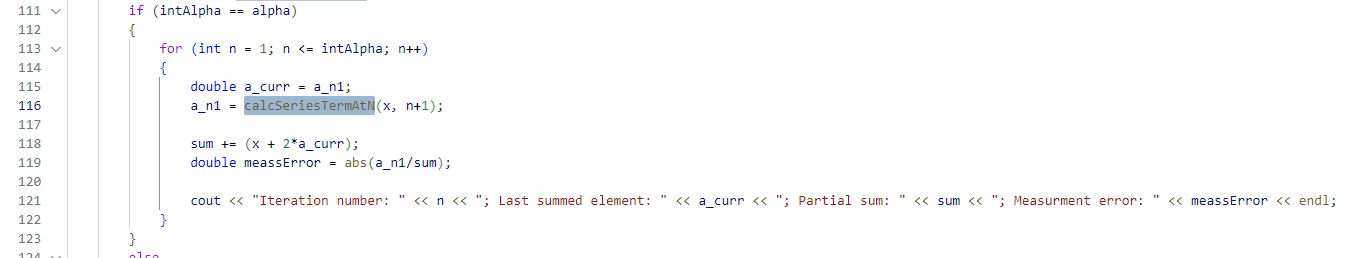


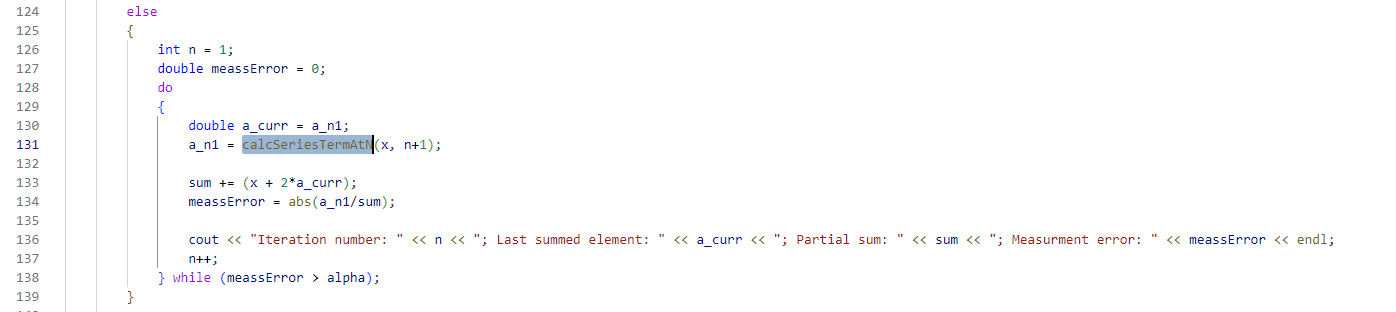
Рисунок 1 — обработка ввода

После ввода значений, код проверяет является ли alpha целым числом, присваивая значение alpha к переменной int и сравнивая значения исходной переменной и int переменной.

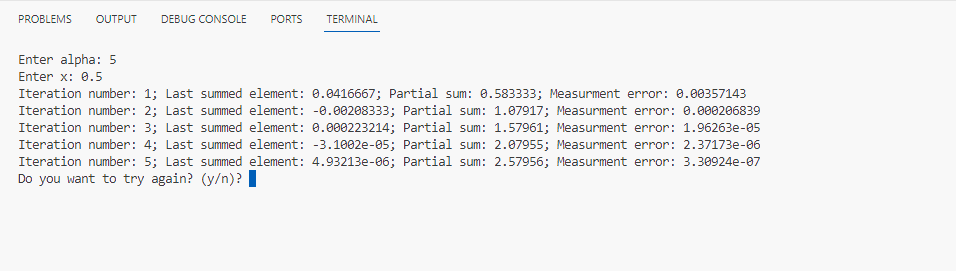
В случае, если alpha – целое число, вызывается цикл for, итерирующийся alpha раз. На каждой итерации цикла вызывается функция calcSeriesTermAtN, высчитывающая по формуле следющий элемент ряда. Результат, возвращенный функцией, присваивается переменной a\_n1. Затем предыдующее значение a\_n1 добавляется к частичной сумме, а текущее значение a\_n1 используется для вычисления погрешности (Рисунок 2).

Рисунок 2 — цикл, когда alpha это цилое число

В случае если alpha не является целым числом, вызывается цикл while, итерирующийся пока alpha меньше вычисляемой на каждой итерации погрешности. Он также вызывает calcSeriesTermAtN для получения следующего элемента ряда, и вычисляет погрешность с частичной суммой основываясь на этом (Рисунок 3).

Рисунок 3 — цикл, когда alpha не является целым числом

Оба цикла выводят порядковый номер итерации, текущий элемент ряда, частичную сумму и погрешность, как можно увидеть на рисунке 4.

Рисунок 4 — результат выполнения программы

Заключение

Цель лабораторной работы – написать программу, предназначенную для приближённого вычисления суммы бесконечного числового ряда, достигнута. В ходе выполнения лабораторной работы я изучил базовые циклы C++, приобрел навыки по использованию функций, написанию математических выражений и применению регулярных выражений. Самой интересной частью работы было вынесение вычисления текущего элемента ряда в отдельную функцию и использование результата этой функции в теле цикла.

Литература

1. Безус, Е.А. Лабораторные работы по курсу «Методы программирования» (язык программирования C++): Методические указания [Текст] / Сост. Е.А. Безус, Е.С. Козлова, С.С. Стафеев Самара, 2019. – 56 с.
2. Страуструп, Б. Язык программирования С++ / Б. Страуструп. – М:Бином, 2011 –1136 с.
3. Шилдт. Г. С++. Полное руководство / Г. Шилдт. – М: "ООО И.Д. Вильямс", 2019 – 800 с.
4. Страуструп, Б. Программирование: принципы и практика использования С++, испр. изд.: Пер. с англ. / Б. Страуструп. – М: "ООО И.Д. Вильямс", 2011 –1244 с.
5. Прата, С. Язык программирования С++. Лекции и упражения , 6-е изд.: Пер. с англ. / С. Прата. – Киев: "Диалектика", 2018 –1248 с.
6. Библиографическое описание документа. Общие требования и правила составления [Электронный ресурс] / сост.: В.С. Крылова, С.М. Григорьевская, Е.Ю. Кичигина // Официальный интернет-сайт научной библиотеки Томского государственного университета. – Электрон. дан. – Томск, [2010]. – <http://www.lib.tsu.ru/win/produkzija/metodichka/metodich.html> (дата обращения: 10.09.2019).

Приложение

#include <iostream>

#include <regex>

using namespace std;

/// @brief Tries to convert string double, throws invalid\_argument with readable text if it goes wrong

/// @return converted strNumber of type double

double parseRealNum(string strNumber)

{

regex reg\_matchRealNums = regex(R"(^[+-]?[0-9]+([,.][0-9]+)?$)");

double result = 0;

try

{

result = stod(strNumber);

//regex to block the case when input is smth like "100ffas"

if(!regex\_match(strNumber, reg\_matchRealNums))

throw invalid\_argument("");

}

catch(const invalid\_argument& e)

{

throw invalid\_argument("(The number was in invalid format)");

}

catch(const out\_of\_range& e)

{

throw invalid\_argument("(The number was way too big)");

}

return result;

}

/// @brief shows greetingText to user, then reads entered number of type double

/// @return entered double

double readDouble(string greetingText)

{

string inp;

double result;

bool readValueSuccessfully = false;

do

{

cout << greetingText;

cin >> inp;

try

{

result = parseRealNum(inp);

readValueSuccessfully = true;

}

catch(invalid\_argument& e)

{

cout << e.what() << endl;

}

} while(!readValueSuccessfully);

return result;

}

/// @brief shows greetingText to user, then reads entered number of type double

/// @return entered double

double readDoubleBetween0and1(string greetingText)

{

string inp;

double result;

bool readValueSuccessfully = false;

do

{

cout << greetingText;

cin >> inp;

try

{

result = parseRealNum(inp);

if (result < -1 || result > 1)

{

throw invalid\_argument("(The number has to be between -1 and 1, inclusive)");

}

readValueSuccessfully = true;

}

catch(invalid\_argument& e)

{

cout << e.what() << endl;

}

} while(!readValueSuccessfully);

return result;

}

double calcSeriesTermAtN(double x, int n)

{

return ( pow(-1, n+1) \* pow(x, 2\*n+1) ) / (4 \* pow(n, 2)-1);

}

int main() {

//loop for repeating algo if user needs it

bool doContinue = false;

do

{

doContinue = false;

double alpha, x;

alpha = readDouble("Enter alpha: ");

x = readDoubleBetween0and1("Enter x: ");

int intAlpha = alpha;

double sum = 0;

double a\_n1 = calcSeriesTermAtN(x, 1);

if (intAlpha == alpha)

{

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

{

double a\_curr = a\_n1;

a\_n1 = calcSeriesTermAtN(x, n+1);

sum += (x + 2\*a\_curr);

double meassError = abs(a\_n1/sum);

cout << "Iteration number: " << n << "; Last summed element: " << a\_curr << "; Partial sum: " << sum << "; Measurment error: " << meassError << endl;

}

}

else

{

int n = 1;

double meassError = 0;

do

{

double a\_curr = a\_n1;

a\_n1 = calcSeriesTermAtN(x, n+1);

sum += (x + 2\*a\_curr);

meassError = abs(a\_n1/sum);

cout << "Iteration number: " << n << "; Last summed element: " << a\_curr << "; Partial sum: " << sum << "; Measurment error: " << meassError << endl;

n++;

} while (meassError > alpha);

}

//asks whether to execute algo again or not

char char\_doContinue = ' ';

while (char\_doContinue != 'y' && char\_doContinue != 'Y' && char\_doContinue != 'n' && char\_doContinue != 'N')

{

cout << "Do you want to try again? (y/n)? ";

cin >> char\_doContinue;

if (char\_doContinue == 'y' || char\_doContinue == 'Y')

doContinue = true;

}

} while (doContinue);

}