Тампио Илья Сергеевич 3530904/10002 Лаб. Работа №2 Вариант 5.

**Задание:**

1. Написать функцию для вычисления значения предложенной суммы в заданной точке (x) с указанной абсолютной погрешностью вычисления (absError) и максимальным числом слагаемых (numberMax).



1. Написать программу вывода таблицы вычисленных значений функции на указанном интервале и значений, полученных с использованием стандартных функций C++.

*Замечания:*

* Абсолютная погрешность (absError) представляет собой модуль первого отбрасываемого члена ряда суммы.
* Если номер слагаемого суммы больше numberMax и точность не достигнута, или x не попадает в заданный интервал функция должна инициировать исключение.
* Нельзя использовать функции возведения в степень и вычисление факториала.

Входные данные: Точность вычисления, максимальное число слагаемых, интервал, на котором проводятся вычисления, шаг интервала.

Выходные данные: Таблица вычисленных значений функции на указанном интервале и значений, полученных с использованием стандартных функций C++.

***Детальные требования и тестплан:***

|  |  |  |  |
| --- | --- | --- | --- |
| **Пункт** | **Требование** | **Данные для пункта** | **Ожидаемый результат** |
| 1. Корректность ввода | | | |
| * 1. Абсолютная погрешность   (absError) | Является вещественным ненулевым положительным числом, иначе исключение: «Некорректный ввод: абсолютная погрешность» | Неверные:  А десять 3.Б  (Верные:  0.1 0.001 0.0002) | Если введено неверное число или не число: «Некорректный ввод: Абсолютная погрешность» и завершение программы. |
| 1.2 Максимальное число слагаемых  (numberMax) | Является натуральным числом, иначе исключение: «Некорректный ввод: максимальное число слагаемых» | Неверные:  -1 0 -95 А  (Верные:  256 1024) | Если введено неверное число или не число: «Некорректный ввод: Максимальное число слагаемых» и завершение программы. |
| 1.3 Интервал [x1, x2] | Является парой вещественных чисел x1 x2, причём  -1 < x1 <= x2 < 1.  Если условие не выполнено, исключение.  Примечание: Интервал интерпретируется как строгий, т.е. подходят все значения x  Следовательно, интервал в котором x1 = x2 будет содержать в себе единственный элемент. | Неверные:  0.0A 0.0B  -1 1  0.99 -0.99  (Верные:  0.2 0.5  -0.99 0.99  -0.25 0.5  0.5 0.5) | Если введено не число – Исключение: «Некорректный ввод: Интервал» и завершение программы.  Если не удовлетворяют -1 < x1 <= x2 < 1 - Исключение: «Некорректный ввод: Интервал (не выполнено условие -1 < x1 <= x2 < 1)» и завершение программы. |
| 1.4 Шаг интервала (dx) | Является вещественным положительным ненулевым числом dx. Иначе, исключение.  *Не* запрашивается, если x1 == x2.  Примечание:  Если dx > x2-x1, тогда вычисления произведутся только для x1 как для первого элемента к которому dx не добавляется. | Неверные:  0.0А 0.0B -0.25 0  (Верные:  0.025 0.5 1 256) | Если введено не число или неверное число – Исключение: «Некорректный ввод: Шаг интервала» и завершение программы. |
| 1. Вычисление суммы | | | |
| 2.1 Ограничение Максимальным числом слагаемых | Если заданная точность не достигается после того как было просуммированно больше допустимого числа слагаемых, исключение. | Проверка:  0.0000001 5 -0.5 0.5 0.1 | Вместо значения в таблице приближенное значение и сообщение: «Ошибка вычисления: точность не достигнута». |
| 2.2 Достижение точности | Точность считается достигнутой, когда Абсолютная погрешность больше, чем модуль следующего элемента суммы. При этом, часть числа, меньшая чем погрешность отбрасывается. | Проверка:   * 1. 654321 -0.5 -0.5   2. 654321 -0.5 -0.5   0.001 654321 -0.5 -0.5  0.0001 654321 -0.5 -0.5 | 1.6  1.64  1.648  1.6487 |
| 1. Проверки правильности интервала | | | |
| 3.1 Интервал x1 x2 с шагом dx | В такой интервал входят элементы x1 и x1+dx\*i , где i – натуральное число такое что x1+dx\*i < x2 | Проверка:   |  |  |  | | --- | --- | --- | | x1 | x2 | dX | | -0.5 | 0.5 | 0.2 | | 0 | 0.9 | 0.2 | | Войдут в вычисления:  -0.5 -0.3 -0.1 0.1 0.3 0.5  0 0.2 0.4 0.6 0.8 |
| 3.2 Интервал x1 x2, где x1==x2 | В такой интервал войдёт только x1 | |  |  |  | | --- | --- | --- | | x1 | x2 | dX | | 0 | 0 |  | | 0.5 | 0.5 |  | | Войдут в вычисления:  0  0.5 |

**Ход решения:**

**Рекуррентная формула n-ого члена суммы.**

Код:

#include <iostream>

#include <iomanip>

#include <cmath>

#define DELIMITER '|'

#define MARGIN 3 //Кол-во дополнительно выводимых знаков после запятой

typedef const char\* InputException;

struct CalculationException {

const char\* what;

double approxValue;

};

int calcNumbersAfterPoint(double n) {

//Подсчитывает кол-во знаков после запятой до 30

const int MAXCOUNTER = 30;

int counter = 0;

for (; int(n)!=n && counter < MAXCOUNTER; n \*= 10) {

counter++;

}

return counter;

}

double truncate(double value, int n) {

//Отбрасывает все цифры после запятой после n-ой

return std::trunc(value \* pow(10, n)) / pow(10, n);

}

double ePowerMinusXApprox(double& x, double& absError, int& numberMax) {

const char\* PRECISION\_NOT\_REACHED\_EXCEPTION = "Точность не достигнута";

double result = 1.0;

double elemI = 1;

double factorial = 1;

int factorialMultiply = 2;

double xPower = -x;

int nSignsAfterPoint = calcNumbersAfterPoint(absError);

for (int i = 1; absError < abs(elemI); i++) {

if (i > numberMax) {

throw CalculationException {PRECISION\_NOT\_REACHED\_EXCEPTION, truncate(result, nSignsAfterPoint)};

}

elemI = xPower / factorial;

result += elemI;

xPower \*= -x;

factorial \*= factorialMultiply;

factorialMultiply += 1;

}

return truncate(result, nSignsAfterPoint);

}

void printTableLine(

const double& x,

const double& val1,

const double& val2,

const int& numbersAfterPoint,

const int& xFieldWidth,

const char& delimiter='|')

{

std::cout << delimiter << std::setw(xFieldWidth) << std::setprecision(xFieldWidth-2) << x

<< delimiter << std::setw(numbersAfterPoint + 2) << std::setprecision(numbersAfterPoint) << val1

<< delimiter << std::setw(numbersAfterPoint + 2) << std::setprecision(numbersAfterPoint) << val2

<< delimiter << std::endl; // +2 для ячеек под первую цифру числа и точку.

}

int main() {

setlocale(LC\_ALL, "RUSSIAN");

InputException ABSERROR\_GENERAL\_EXCEPTION = "Абсолютная погрешность";

InputException NUMBERMAX\_GENERAL\_EXCEPTION = "Максимальное число слагаемых";

InputException INTERVAL\_GENERAL\_EXCEPTION = "Интервал";

InputException INTERVAL\_LIMITS\_EXCEPTION = "Интервал (не выполнено условие -1 < x1 <= x2 < 1)";

InputException INTERVAL\_STEP\_GENERAL\_EXCEPTION = "Шаг интервала";

try {

std::cout << "Абсолютная погрешность: ";

double absError = 0.0;

std::cin >> absError;

if (!std::cin.good() || absError <= 0) {

throw ABSERROR\_GENERAL\_EXCEPTION;

}

std::cout << "Максимальное число слагаемых: ";

int numberMax = 0;

std::cin >> numberMax;

if (!std::cin.good() || numberMax <= 0) {

throw NUMBERMAX\_GENERAL\_EXCEPTION;

}

std::cout << "Интервал (два числа такие, что -1 < x1 <= x2 < 1): ";

double x1 = 0.0;

double x2 = 0.0;

std::cin >> x1 >> x2;

if (!std::cin.good()) {

throw INTERVAL\_GENERAL\_EXCEPTION;

}

else if(!(x1 <= x2 && x1 > -1 && x2 < 1)){

throw INTERVAL\_LIMITS\_EXCEPTION;

}

bool onlyOneIntervalElem = x1 == x2;

double dx = 2;

if (!onlyOneIntervalElem) {

std::cout << "Шаг интервала: ";

std::cin >> dx;

if (!std::cin.good() || dx <= 0) {

throw INTERVAL\_STEP\_GENERAL\_EXCEPTION;

}

}

int numbersAfterPointX1 = calcNumbersAfterPoint(x1);

int numbersAfterPointX2 = calcNumbersAfterPoint(x2);

int maxNumbersAfterPointX = (numbersAfterPointX1 > numbersAfterPointX2) ? numbersAfterPointX1 : numbersAfterPointX2;

int numbersAfterPoint = calcNumbersAfterPoint(absError) + MARGIN;

const int xFieldWidth = maxNumbersAfterPointX + 2; // +2 для вывода "0."

std::cout << std::endl << DELIMITER << std::setw(xFieldWidth) << "x"

<< DELIMITER << std::setw(numbersAfterPoint + 2) << "Approx"

<< DELIMITER << std::setw(numbersAfterPoint + 2) << "exp(-x)"

<< DELIMITER << std::endl;

//Меньшевизна эпсилон зависит от количества знаков в пределах интервала

double epsilon = dx / pow(10, maxNumbersAfterPointX);

for (double x = x1; x < x2 + epsilon; x += dx) {

double val1 = 0;

try {

val1 = ePowerMinusXApprox(x, absError, numberMax);

}

catch (CalculationException exception){

std::cerr << "Ошибка вычисления: " << exception.what << std::endl;

val1 = exception.approxValue;

}

double val2 = exp(-x);

printTableLine(x, val1, val2, numbersAfterPoint, xFieldWidth, DELIMITER);

}

}

catch (InputException what) {

std::cerr << "Ошибка ввода: " << what << std::endl;

return 1;

}

return 0;

}