**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра** **САПР**

отчет

**по лабораторной работе №2**

**по дисциплине «Программирование»**

Тема: Итерационные вычисления суммы   
с вещественными числами

|  |  |  |
| --- | --- | --- |
| Студент гр. 4335 |  | Иванов Г.Д. |
| Руководитель |  | Калмычков В.А. |

Санкт-Петербург

2025

**Формулировка задания:**

Написание программы с использованием циклов (итерационные алгоритмы с переходным коэффициентом q)

**Дано:**

Формула для слагаемого частичной суммы:

**Найти:**

Значение частичной суммы , где:

**Способ решения:**

Для решения задачи воспользуемся полученной в «Практическом задании №2» формулой с переходным коэффициентом:

**Особенности компьютерной реализации:**

При реализации программ версий 2 и 3 использовались различные варианты работы с массивами – объявление ограниченного массива в области локальной памяти (стэк/stack) и выделение динамического массива в области динамической памяти (куча/heap) соответственно. Разница между этими двумя подходами представлена в таблице 1:

|  |  |  |
| --- | --- | --- |
| **Версия программы** | 2 | 3 |
| **Тип памяти** | Локальная (стек / stack) | Динамическая (куча / heap) |
| **Способ выделения** | Объявление локальной переменной | Динамическое выделение через оператор new [] |
| **Количество памяти** | Константное, определено на этапе компиляции | Динамическое, определено на этапе исполнения |
| **Обращение к элементам массива** | Индексный оператор обращения (p[i]) | Разыменование указателя (\*(p+i)) |
| **Способ очистки памяти** | Выход за границы области видимости (выход из функции main) | Освобождение памяти оператором delete [] |

Таблица 1. Особенности подходов к использованию различных типов памяти

В версии программы 2 при каждом запуске будет задействовано одинаковое количество оперативной памяти, вне зависимости от введенных пользователем параметров. В версии 3 – будет выделено столько памяти, сколько требуется для конкретного вычисления с конкретными введенными параметрами.

**Организация интерфейса пользователя:**

Интерфейс программы состоит из нескольких макетов ввода/вывода, с небольшими отличиями между 2 и 3 версией программы.

1) MO1:

1.1) Версия 2:

===========================================================

Лабораторная работа №2 по дисциплине "Программирование"

Автор: Иванов Григорий Денисович

Группа: 4335 (подгруппа 1)

Версия: 2

Период выполнения работы: 16.01.2025 - 17.01.2025 ===========================================================

Описание:

В этой программе будет проведен расчёт частичной суммы

S\_n(x) = SUM (u\_1(x), u\_2(x), ... u\_n(x)), номер последнего слагаемого

определяется в соответствии со значением числа e>0.

Формула для слагаемого u\_i(x):

u\_i(x)=(-1)^i \* 2^(2i-1) \* (i-1)! \* i! \* x^(2i+1) / (2i+1)!

Упрощенная формула (формула с переходным коэффициентом):

u\_i+1(x) = (-2x^2\*i) / (2i+3) \* ui(x)\n"

При вычислениях используются объявленные массивы с

определенным числом элементов

1.2) Версия 3:

===========================================================

Лабораторная работа №2 по дисциплине "Программирование"

Автор: Иванов Григорий Денисович

Группа: 4335 (подгруппа 1)

Версия: 3

Период выполнения работы: 16.01.2025 - 17.01.2025

===========================================================

Описание:

В этой программе будет проведен расчёт частичной суммы

S\_n(x) = SUM (u\_1(x), u\_2(x), ... u\_n(x)), номер последнего слагаемого

определяется в соответствии со значением числа e>0.

Формула для слагаемого u\_i(x):

u\_i(x)=(-1)^i \* 2^(2i-1) \* (i-1)! \* i! \* x^(2i+1) / (2i+1)!

Упрощенная формула (формула с переходным коэффициентом):

u\_i+1(x) = (-2x^2\*i) / (2i+3) \* ui(x)

При вычислении используются инструменты для динамического

выделения необходимой памяти

2) MI2:

Введите X из диапазона (-1, 1): <вещественное число в обычном или научном представлении>

3) MI3:

Введите e из диапазона (0, 1): <вещественное число в обычном или научном представлении>

3) MO4:

| i| u\_i(x)| s\_i(x)|

| 1| <число с точностью до 10 знаков>|<число с точностью до 10 знаков>|

| 2| <число с точностью до 10 знаков>|<число с точностью до 10 знаков>|

…

…

| n-1| <число с точностью до 10 знаков>|<число с точностью до 10 знаков>|

| n| <число с точностью до 10 знаков>|<число с точностью до 10 знаков>|

4) MO5:

Результат: <число с точностью до 10 знаков>

Для реализации макетов использовались такие модификаторы потока вывода как:

1. std::setw – для указания ширины выводимого слова
2. std::setprecision – для указания максимального количества знаков после запятой

**Описание алгоритма работы программы:**

Версия 2:

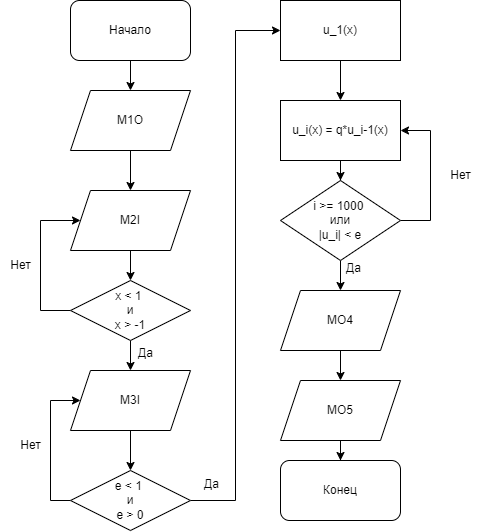


Рисунок 1. Алгоритм работы программы версии 2.

Версия 3:

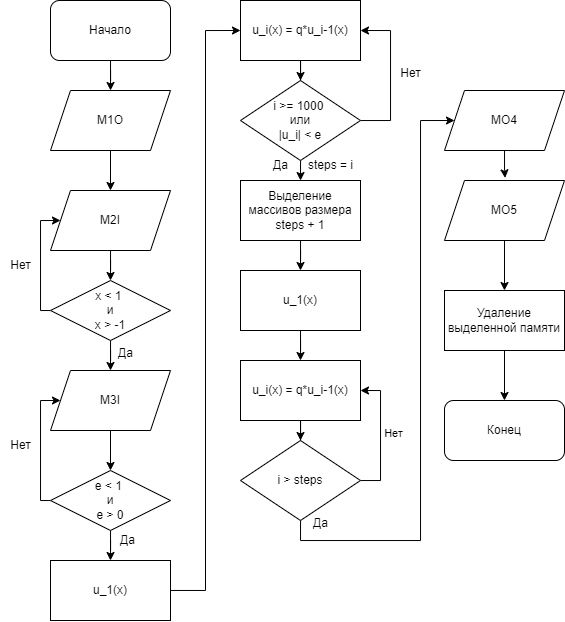


Рисунок 2. Алгоритм работы программы версии 3.

**Текст программы:**

Версия 2:

#include <iostream>

#include <iomanip>

namespace {

const char credits [] = "===========================================================\n"

"Лабораторная работа №2 по дисциплине \"Программирование\"\n"

"Автор: Иванов Григорий Денисович\n"

"Группа: 4335 (подгруппа 1)\n"

"Версия: 2\n"

"Период выполнения работы: 16.01.2025 - 17.01.2025\n"

"===========================================================";

const char description [] = "Описание:\n"

"В этой программе будет проведен расчёт частичной суммы\n"

"S\_n(x) = SUM (u\_1(x), u\_2(x), ... u\_n(x)), номер последнего слагаемого\n"

"определяется в соответствии со значением числа e>0.\n"

"Формула для слагаемого u\_i(x):\n"

" u\_i(x)=(-1)^i \* 2^(2i-1) \* (i-1)! \* i! \* x^(2i+1) / (2i+1)!\n"

"Упрощенная формула (формула с переходным коэффициентом):\n"

" u\_i+1(x) = (-2x^2\*i) / (2i+3) \* ui(x)\n"

"При вычислениях используются объявленные массивы с\n"

"определенным числом элементов\n";

}

int main()

{

double x, ep;

double tmps[1001], sums[1001];

unsigned int i {1}, steps;

std::cout << credits << std::endl;

std::cout << description << std::endl;

do

{

std::cout << "Введите X из диапазона (-1, 1):";

std::cin >> x;

}while(x <= -1 || x >= 1);

do

{

std::cout << "Введите e из диапазона (0, 1):";

std::cin >> ep;

}while(ep <= 0 || ep >= 1);

tmps[0] = -x\*x\*x / 3;

sums[0] = tmps[0];

for(;;i++)

{

tmps[i] = tmps[i-1] \* -2 \* x \* x \* i;

tmps[i] /= 2 \* i + 3;

sums[i] = sums[i-1] + tmps[i];

if(i >= 1000 || (tmps[i] < 0? -tmps[i] : tmps[i]) < ep)

break;

}

steps = i;

std::cout << "|" << std::setw(10) << "i" //"Шаг"

<< "|" << std::setw(20) << "u\_i(x)" //"Слагаемое"

<< "|" << std::setw(20) << "s\_i(x)" //"Сумма"

<< "|" << std::endl;

for(i = 0; i < steps; i++)

{

std::cout << "|" << std::setw(10) << i + 1

<< "|" << std::setw(20) << std::setprecision(10) << tmps[i]

<< "|" << std::setw(20) << std::setprecision(10) << sums[i]

<< "|" << std::endl;

}

std::cout << "Результат: " << sums[steps-1] << std::endl;

return 0;

}

Версия 3:

#include <iostream>

#include <iomanip>

namespace {

const char credits [] = "===========================================================\n"

"Лабораторная работа №2 по дисциплине \"Программирование\"\n"

"Автор: Иванов Григорий Денисович\n"

"Группа: 4335 (подгруппа 1)\n"

"Версия: 3\n"

"Период выполнения работы: 16.01.2025 - 17.01.2025\n"

"===========================================================";

const char description [] = "Описание:\n"

"В этой программе будет проведен расчёт частичной суммы\n"

"S\_n(x) = SUM (u\_1(x), u\_2(x), ... u\_n(x)), номер последнего слагаемого\n"

"определяется в соответствии со значением числа e>0.\n"

"Формула для слагаемого u\_i(x):\n"

" u\_i(x)=(-1)^i \* 2^(2i-1) \* (i-1)! \* i! \* x^(2i+1) / (2i+1)!\n"

"Упрощенная формула (формула с переходным коэффициентом):\n"

" u\_i+1(x) = (-2x^2\*i) / (2i+3) \* ui(x)\n"

"При вычислении используются инструменты для динамического\n"

"выделения необходимой памяти\n";

}

int main()

{

double x, tmp, ep;

double \* tmps { nullptr }, \* sums { nullptr };

unsigned int i {1}, steps;

std::cout << credits << std::endl;

std::cout << description << std::endl;

do

{

std::cout << "Введите X из диапазона (-1, 1):";

std::cin >> x;

}while(x <= -1 || x >= 1);

do

{

std::cout << "Введите e из диапазона (0, 1):";

std::cin >> ep;

}while(ep <= 0 || ep >= 1);

tmp = -x\*x\*x / 3;

for(;;i++)

{

tmp \*= -2 \* x \* x \* i;

tmp /= 2 \* i + 3;

if(i >= 1000 || (tmp < 0? -tmp : tmp) < ep)

break;

}

steps = i;

tmps = new double[steps + 1];

sums = new double[steps + 1];

\*tmps = -x\*x\*x / 3;

\*sums = \*tmps;

for(i = 1; i <= steps; i++)

{

\*(tmps + i) = \*(tmps + i - 1) \* -2 \* x \* x \* i;

\*(tmps + i) /= 2 \* i + 3;

\*(sums + i) = \*(sums + i - 1) + \*(tmps + i);

}

std::cout << "|" << std::setw(10) << "i" //"Шаг"

<< "|" << std::setw(20) << "u\_i(x)" //"Слагаемое"

<< "|" << std::setw(20) << "s\_i(x)" //"Сумма"

<< "|" << std::endl;

for(i = 0; i < steps; i++)

{

std::cout << "|" << std::setw(10) << i + 1

<< "|" << std::setw(20) << std::setprecision(10) << \*(tmps + i)

<< "|" << std::setw(20) << std::setprecision(10) << \*(sums + i)

<< "|" << std::endl;

}

std::cout << "Результат: " << \*(sums + steps - 1) << std::endl;

delete [] tmps;

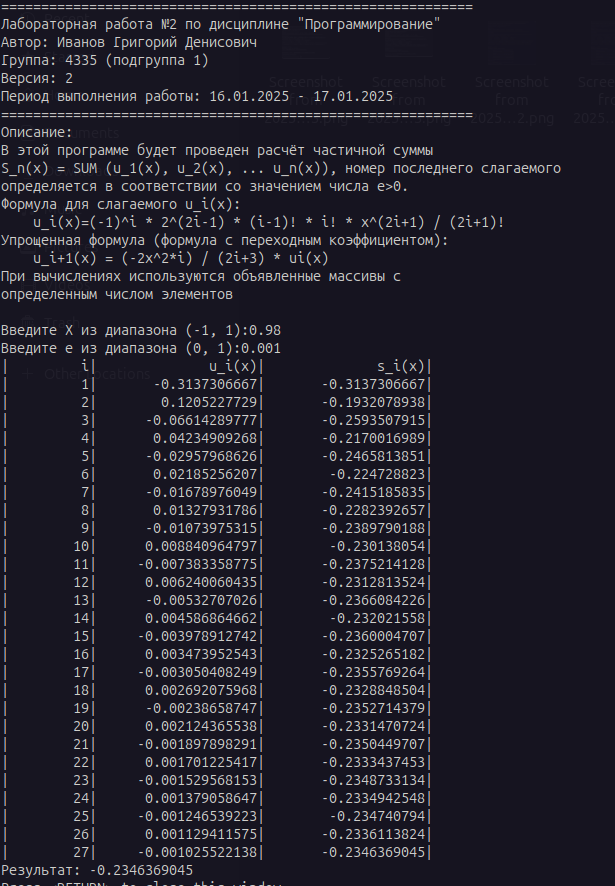
delete [] sums;

return 0;

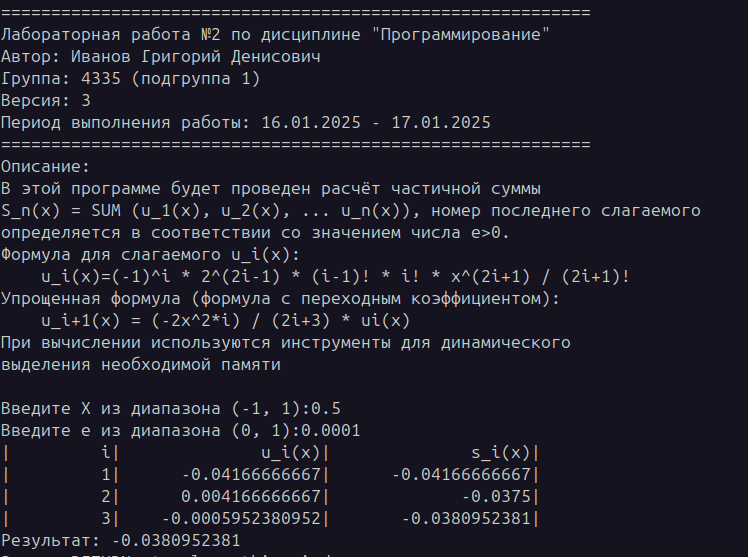
}

**Результат работы программы:**

Версия 2:



Версия 3:



**Выводы:**

В этой лабораторной работе мы научились работать с различными видами циклов, освоили два разных способа работы с массивами (локальное определение и динамическое выделение), а также воспользовались модификаторами стандартного потока вывода с++. В ходе работы мы провели реализацию итерационного алгоритма нахождения частичной суммы математического выражения.