ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»  
(РУТ (МИИТ))

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

ОТЧЁТ  
ПО ЛАБОРАТОРНОЙ РАБОТЕ №3

По дисциплине «Языки программирования»

ВАРИАНТ 12

Выполнил: ст. гр. ТКИ-141

Комаров Константин Сергеевич

Проверил: к.т.н., доц. Васильева М. А.

(Проверил: к.т.н, доц. Балакина Е. П.)

Москва 2024

**РЕШЕНИЕ ЗАДАЧИ 3-2**

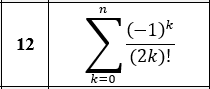
* 1. **Формулировка задачи 3-2**

Составьте две программы:

1. вычислить сумму первых n членов последовательности (k = 1, 2, 3 ..., n).
2. вычислить сумму всех членов последовательности, не меньших заданного числа e.

Помните о проверке пользовательского ввода. Все результаты вывести на экран. Отчёт дополнить блок-схемой. При вычислении факториалов рекомендуется отказаться от использования рекурсивных методов.

Таблица 1 – Исходные данные

****

* 1. **Блок-схема алгоритма задания 3-2**

Блок-схема основного алгоритма представлена ниже (Рисунок 1). Блок – схемы функций ***SummaN*** и ***SummaE*** представлены ниже (Рисунок 2). Блок – схема функции ***Recurrent*** (Рисунок 3). Блок - схемы проверки ввода (Рисунок 4).

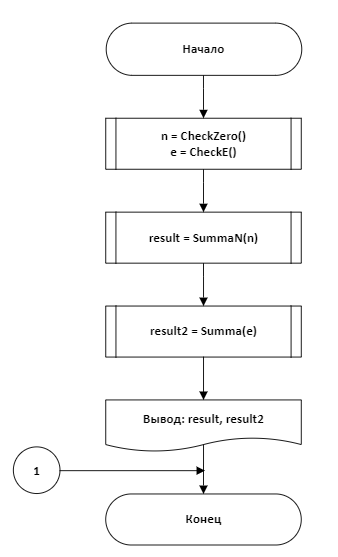


Рисунок 1 – Блок-схема основного алгоритма

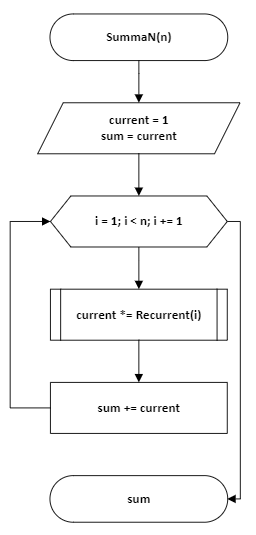
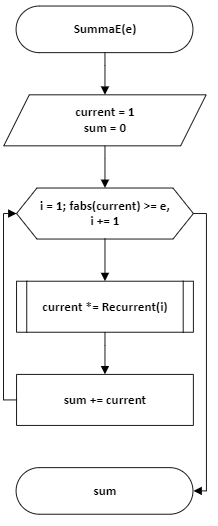
 

Рисунок 2 - Блок – схемы функций ***SummaN*** и ***SummaE***

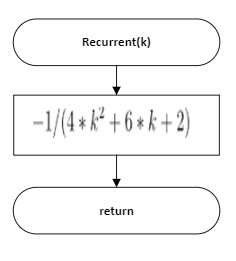
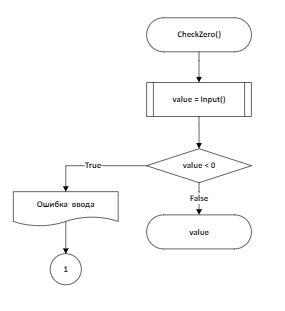
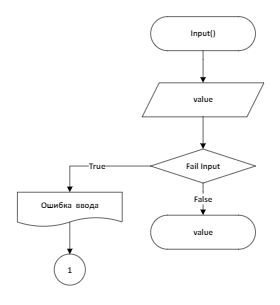


Рисунок 3 - Блок – схема функции ***Recurrent***



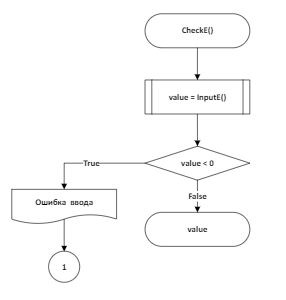
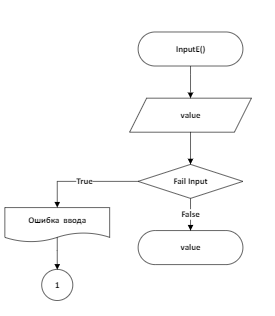


Рисунок 4 - Блок - схемы проверки ввода

* 1. **Текст программы на языки СИ задание 3-2**

#include <stdio.h>

#include <math.h>

#include <stdlib.h>

#include <locale.h>

#include <float.h>

/\*\*

\*@brief Функция подсчета реккурентной функции.

\*@param k значение параметра k.

\*@return Возвращает k.

\*/

double Recurrent(const double k);

/\*\*

\*@brief Функция счета суммы по n-ому члену

\*@param n - переменна заданная пользователем

\*@return Фукнция возвращает значение суммы по n-ому члену

\*/

double SummaN(const double n);

/\*\*

\*@brief Функция проверки ввода.

\*@return Возвращает значение, если верное, иначе выдает ошибку.

\*/

int Input(void);

/\*\*

\*@brief Функция проверки на ноль.

\*@return Возвращает значение, если верное, иначе выдает ошибку.

\*/

int CheckZero(void);

/\*\*

\*@brief Функция подсчёта суммы ряда.

\*@param n значение параметр n.

\*@return Возвращает значение суммы.

\*/

double SummaE(const int e);

/\*\*

\*@brief Фукнция счета суммы с погрешностью e

\*@return Возвращает значение посчитанной функции

\*/

double InputE(void);

/\*\*

\*@brief Функция проверки ввода.

\*@return Возвращает значение, если верное, иначе выдает ошибку.

\*/

double CheckE(void);

/\*\*

\*@brief Точка входа.

\*@return Возвращает значение функции с кодом ошибки 0.

\*/

int main(void)

{

setlocale(LC\_ALL, "Russian");

puts("Введите верхний порог");

int n = CheckZero();

puts("Введите погрешность");

double e = CheckE();

printf("Сумма %d членов: %lf\n", n, SummaN(n));

printf("Сумма членов с погрешностью: %lf\n", SummaE(e));

return 0;

}

double Recurrent(const double k)

{

return -1 / (4 \* pow(k, 2) + 6 \* k + 2);

}

double SummaN(const double n)

{

double current = 1;

double sum = current;

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

{

current \*= Recurrent(i);

sum += current;

}

return sum;

}

double SummaE(const int e)

{

double current = 1;

double sum = 0;

for (int i = 1; fabs(current) >= e - DBL\_EPSILON; ++i)

{

current \*= Recurrent(i);

sum += current;

}

return sum;

}

int Input(void)

{

Int value = 0;

if (scanf\_s("%d", &value) != 1)

{

printf("Ошибка ввода");

exit(EXIT\_FAILURE);

}

return value;

}

int CheckZero(void)

{

int value = Input();

if (value < DBL\_EPSILON)

{

printf("n должно быть больше нуля");

exit(EXIT\_FAILURE);

}

return value;

}

double InputE(void)

{

double value = 0.0;

if (scanf\_s("%lf", &value) != 1)

{

printf("Ошибка ввода");

exit(EXIT\_FAILURE);

}

return value;

}

double CheckE(void)

{

double value = InputE();

if (value < DBL\_EPSILON)

{

puts("e должно быть больше нуля");

exit(EXIT\_FAILURE);

}

return value;

}

**1.4 Результаты выполнения программы 3-2**

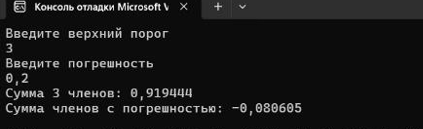


Рисунок 5 – вывод суммы 3 членов и 3 членов с погрешностью

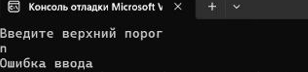


Рисунок 6 – вывод ошибки ввода

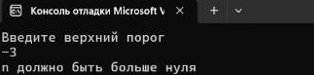


Рисунок 7 – вывод ошибки ввода значения n

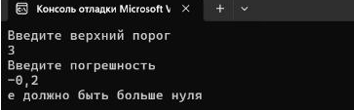


Рисунок 8 – вывод ошибки ввода значения e

* 1. **Выполнение тестовых примеров задание 3-2**

В программе Pythonвыполнены тестовые примеры. Результаты их выполнения представлены ниже (Рисунок 9).

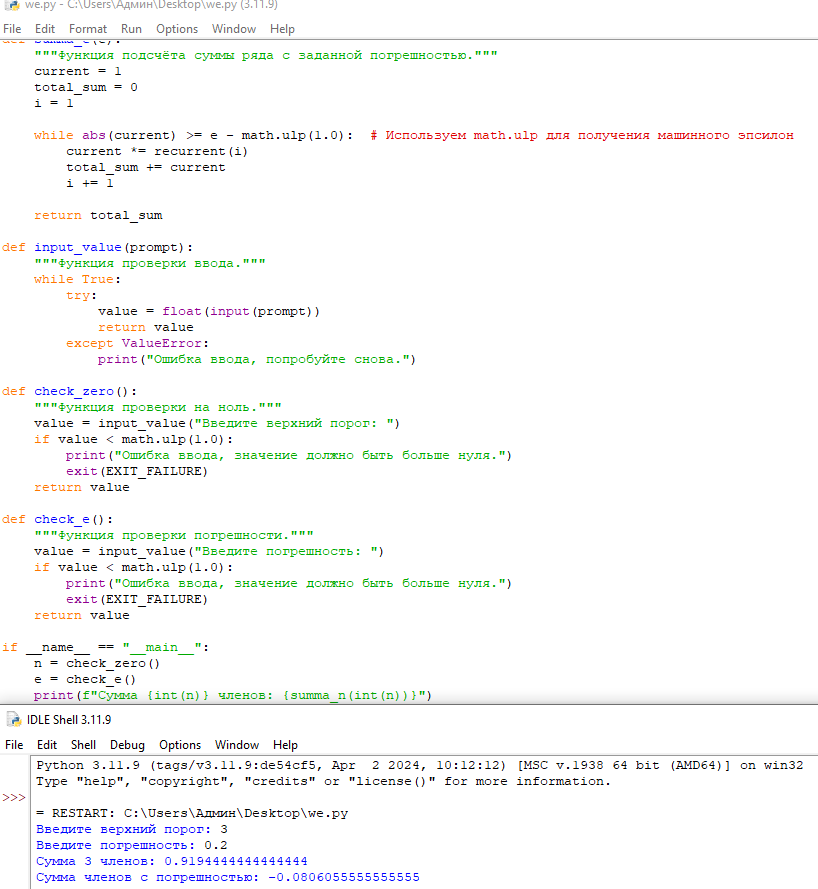


Рисунок 9 – Тестовый пример

**1.6 Отметка о выполнении задания в веб-хостинге системы контроля версий задание 3-1**

Ниже представлено доказательство того, что задание 3–1 было принято. (10)

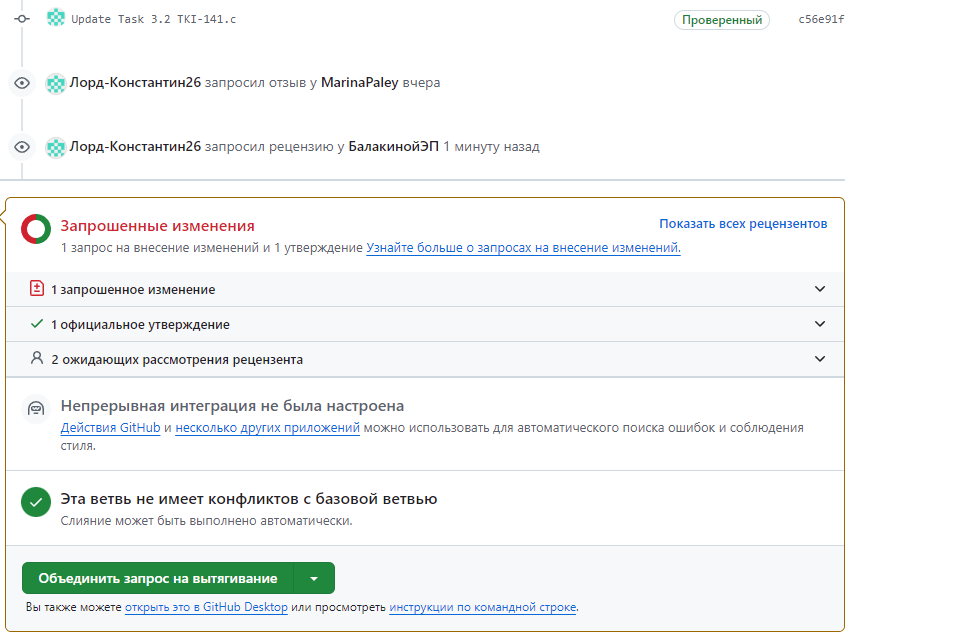


Рисунок 10 – Approve задачи 3–2