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

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

ОТЧЁТ  
О ПРАКТИЧЕСКОЙ РАБОТЕ №3

По дисциплине «Введение в языки программирования»

Выполнил: ст. гр. ТКИ – 111

Панаргин В.М.

Проверил: к.т.н., доц.

Васильева М.А.

Москва 2021

Оглавление

[Задание 3-1 4](#_Toc91584685)

[Условие задания 4](#_Toc91584686)

[Блок-схема алгоритма 5](#_Toc91584687)

[Программа на языке С++ 6](#_Toc91584688)

[Решение тестовых примеров на языке С++ 7](#_Toc91584689)

[Решение тестовых примеров в Excel 8](#_Toc91584690)

[Задание 3-2а 9](#_Toc91584691)

[Условие задания 9](#_Toc91584692)

[Вычисление рекуррентного члена 10](#_Toc91584693)

[Блок-схема алгоритма 11](#_Toc91584694)

[Программа на языке С++ 12](#_Toc91584695)

[Решение тестовых примеров на языке С++ 14](#_Toc91584696)

[Решение тестовых примеров в Excel 15](#_Toc91584697)

[Задание 3-2б 16](#_Toc91584698)

[Условие задания 16](#_Toc91584699)

[Блок-схема алгоритма 17](#_Toc91584700)

[Программа на языке С++ 18](#_Toc91584701)

[Решение тестовых примеров на языке С++ 20](#_Toc91584702)

[Решение тестовых примеров в Excel 21](#_Toc91584703)

[Задание 3-3 22](#_Toc91584704)

[Условие задания 22](#_Toc91584705)

[Вычисление рекуррентного члена 23](#_Toc91584706)

[Блок-схема алгоритма 24](#_Toc91584707)

[Программа на языке С++ 25](#_Toc91584708)

[Решение тестовых примеров на языке С++ 27](#_Toc91584709)

[Решение тестовых примеров в Excel 28](#_Toc91584710)

# Задание 3-1

## Условие задания

Протабулировать заданную в таблице функцию. Использовать данные в таблице значения шага и интервала в качестве ввода пользователя для решения тестового примера. При невозможности расчёта функции в конкретной точке выводить её значение и надпись, означающую отсутствие решения.

image

## Блок-схема алгоритма

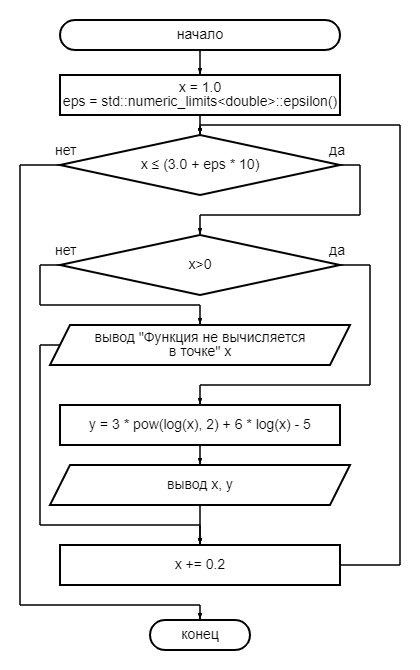


Рисунок 1 - Блок-схема 3-1

## Программа на языке С++

#include <cmath>

#include <iostream>

#include <iomanip>

/\*\*

\* \brief Рассчитывает значение функции в заданной точке \a x.

\* \param x Заданная точка.

\* \return Значение функции в заданной точке \a x.

\*/

double myFunc(double x);

/\*\*

\* \brief Узнает, существует ли функция в заданной точке \a x.

\* \param x Заданная точка.

\* \return true, если значение функции в заданной точке \a x существует.

\*/

bool isCalculated(double x);

/\*\*

\* \brief Точка входа в программу.

\* \return В случаен успеха код 0.

\*/

int main()

{

const double X\_START = 1.0;

const double X\_FINISH = 3.0;

const double STEP = 0.2;

double x = X\_START;

while (x <= X\_FINISH + (std::numeric\_limits<double>::epsilon() \* 10))

{

if (isCalculated(x))

{

const double y = myFunc(x);

std::cout << "x = " << std::setw(5) << std::left << std::setprecision(5) << x << " y = " << y << "\n";

}

else

{

std::cout << "x = " << std::setw(5) << std::left << std::setprecision(5) << x << " y = NaN" << "\n";

}

x += STEP;

}

return 0;

}

double myFunc(const double x)

{

return 3 \* pow(log(x), 2) + 6 \* log(x) - 5;

}

bool isCalculated(const double x)

{

return x > 0;

}

## Решение тестовых примеров на языке С++

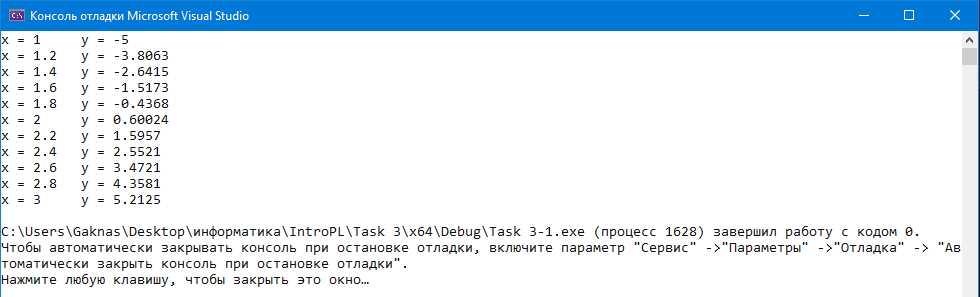


Рисунок 2 - Решение тестовых примеров для задания 3-1 на языке С++

## Решение тестовых примеров в Excel

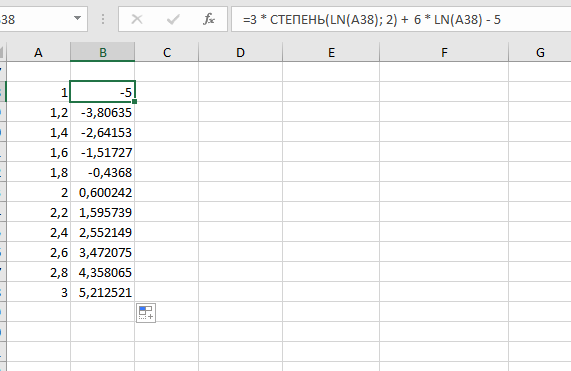


Рисунок 3 - Решение тестовых примеров для 3-1 в Excel

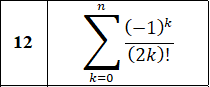
# Задание 3-2а

## Условие задания

Составьте две программы:

a) вычислить сумму первых n членов последовательности (k = 1, 2, 3 ..., n).

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



## Вычисление рекуррентного члена

## Блок-схема алгоритма

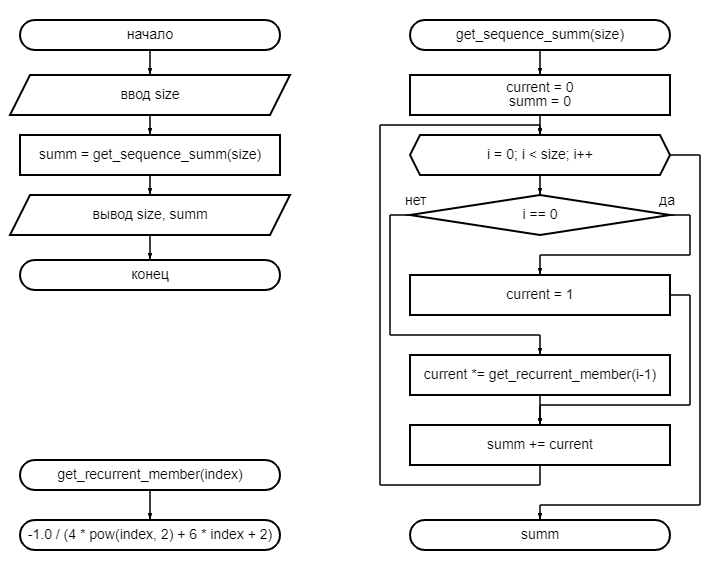


Рисунок 4 - Блок-схема 3-2а

## Программа на языке С++

#include <iostream>

/\*\*

\* \brief Пользовательский ввод количества членов последовательности.

\* \param message Мотивационное сообщение.

\* \return Размер последовательности.

\* \exception std::invalid\_argument В случае, если размер последовательности меньше 1.

\*/

size\_t get\_sequence\_size(const std::string& message);

/\*\*

\* \brief Рассчитывает сумму последовательность с количеством членов (\a size).

\* \param size Количество членов последовательности.

\* \return Сумма последовательности.

\*/

double get\_sequence\_summ(const size\_t size);

/\*\*

\* \brief Рассчитывает значение рекуррентного члена в зависимости от индекса (\a index).

\* \param index Индекс, номер итерации.

\* \return Значение рекуррентного члена.

\*/

double get\_recurrent\_member(const size\_t index) noexcept;

/\*\*

\* \brief Точка входа в программу.

\* \return 0 – в случае успеха,<br/>1 – в противном случае.

\*/

int main() noexcept

{

try

{

setlocale(LC\_ALL, "Russian");

const auto size = get\_sequence\_size("Введите количество членов последовательности ");

const double summ = get\_sequence\_summ(size);

std::cout << "Сумма последовательности из " << size << " членов равна: " << summ << std::endl;

return 0;

}

catch (const std::exception& e)

{

std::cerr << e.what() << std::endl;

return 1;

}

}

double get\_sequence\_summ(const size\_t size) {

double current = 0;

double summ = 0;

for (size\_t i = 0; i < size; i++) {

if (i == 0) {

current = 1;

}

else {

current \*= get\_recurrent\_member(i - 1);

}

summ += current;

}

return summ;

}

size\_t get\_sequence\_size(const std::string& message)

{

std::cout << message;

int size = 0;

std::cin >> size;

if (size < 1)

{

throw std::invalid\_argument("Значение должно быть больше 0!");

}

return size;

}

double get\_recurrent\_member(const size\_t index) noexcept

{

return -1.0 / (4 \* pow(index, 2) + 6 \* index + 2);

}

## Решение тестовых примеров на языке С++

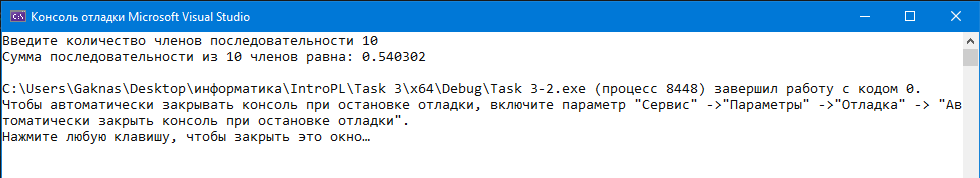


Рисунок 5- Решение тестовых примеров для задания 3-2а на языке С++

## Решение тестовых примеров в Excel

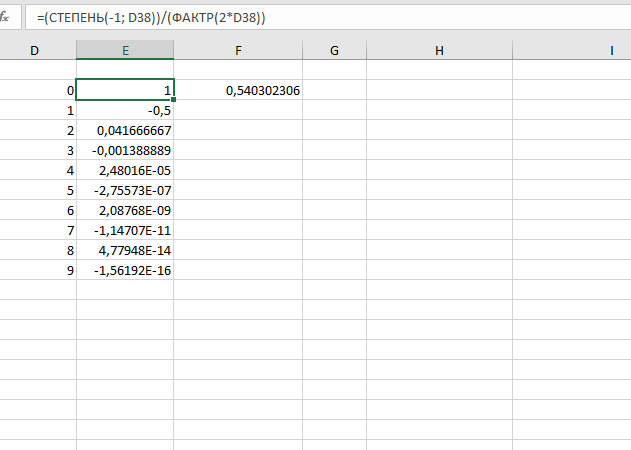


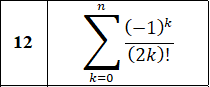
Рисунок 6 - Решение тестовых примеров для задания 3-2а в Excel

# Задание 3-2б

## Условие задания

Составьте программу:  
b) 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.



## Блок-схема алгоритма

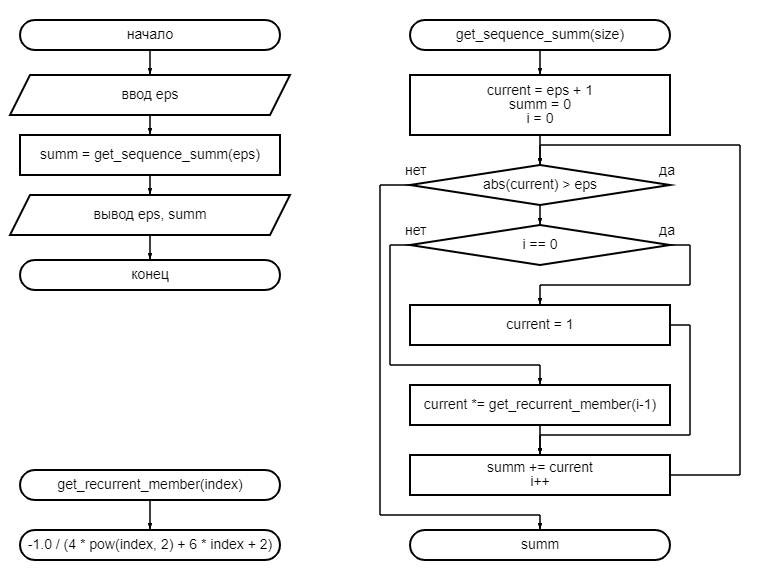


Рисунок 7- Блок-схема 3-2б

## Программа на языке С++

#include <iostream>

/\*\*

\* \brief Пользовательский ввод необходимой точности вычисления.

\* \param message Мотивационное сообщение.

\* \return Точность вычисления.

\* \exception std::invalid\_argument В случае, если размер последовательности меньше 1.

\*/

double get\_epsilon(const std::string& message);

/\*\*

\* \brief Рассчитывает сумму последовательность с точностью (\a eps).

\* \param eps Точность вычисления.

\* \return Сумма последовательности.

\*/

double get\_sequence\_summ(const double eps);

/\*\*

\* \brief Рассчитывает значение рекуррентного члена в зависимости от индекса (\a index).

\* \param index Индекс, номер итерации.

\* \return Значение рекуррентного члена.

\*/

double get\_recurrent\_member(const size\_t index) noexcept;

/\*\*

\* \brief Точка входа в программу.

\* \return 0 – в случае успеха,<br/>1 – в противном случае.

\*/

int main() noexcept

{

try

{

setlocale(LC\_ALL, "Russian");

const auto eps = get\_epsilon("Введите точность вычислений ");

const double summ = get\_sequence\_summ(eps);

std::cout << "Сумма последовательности c точностью " << eps << " равна: " << summ << std::endl;

return 0;

}

catch (const std::exception& e)

{

std::cerr << e.what() << std::endl;

return 1;

}

}

double get\_sequence\_summ(const double eps) {

size\_t i = 0;

double summ = 0;

double current = eps + 1;

while (abs(current) >= eps) {

if (i == 0) {

current = 1;

}

else {

current \*= get\_recurrent\_member(i - 1);

}

summ += current;

i++;

}

return summ;

}

double get\_epsilon(const std::string& message)

{

std::cout << message;

double eps = 0;

std::cin >> eps;

if (eps < 0)

{

throw std::invalid\_argument("Значение должно быть больше 0!");

}

return eps;

}

double get\_recurrent\_member(const size\_t index) noexcept

{

return -1.0 / (4 \* pow(index, 2) + 6 \* index + 2);

}

## Решение тестовых примеров на языке С++

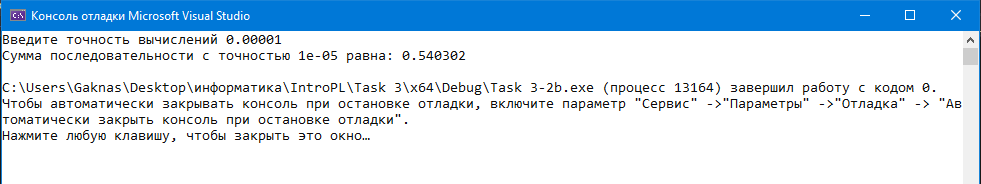


Рисунок 8 - Решение тестовых примеров для задания 3-2б на языке С++

## Решение тестовых примеров в Excel

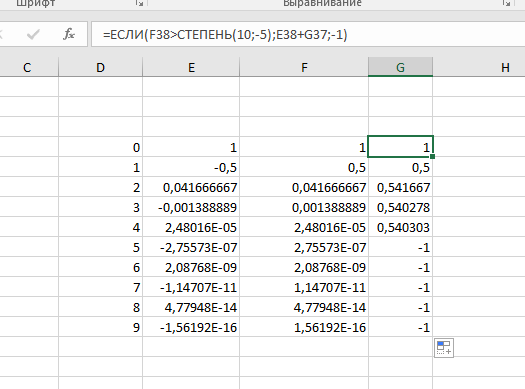
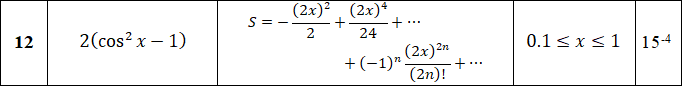


Рисунок 9 - Решение тестовых примеров для задания 3-2б на языке С++

# Задание 3-3

## Условие задания

Протабулировать заданную функцию и сумму функционального ряда разложения этой функции на интервале [a,b] и с шагом h (шаг и интервал задается в константах). Функциональнй ряд вычисляется по соответствующей рекуррентной формуле с заданной точностью ɛ. В результате показать три столбца: значение аргумента, значение функции в данной точке и значение суммы ряда, вычисленное с заданной точностью в данной точке. Два последних столбца должны иметь близкие результаты.



## Вычисление рекуррентного члена

## Блок-схема алгоритма

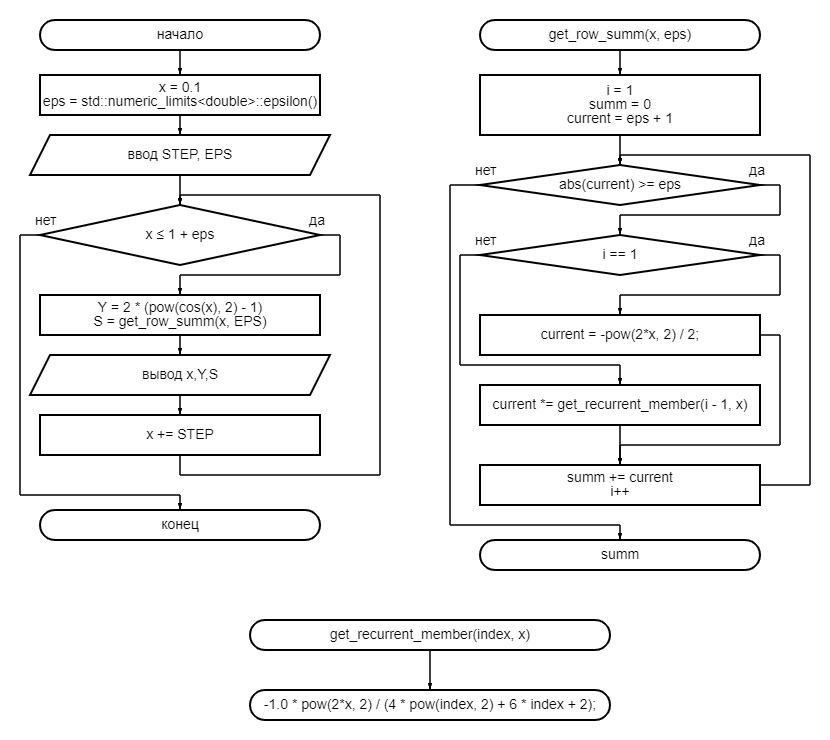


Рисунок 10 - Блок-схема 3-3

## Программа на языке С++

#include <iostream>

#include <cmath>

#include <iomanip>

/\*\*

\* \brief Рассчитывает значение функции в заданной точке (\a x).

\* \param x Абсцисса заданной точки.

\* \return Значение фунции.

\*/

double get\_function\_value(const double x);

/\*\*

\* \brief Рассчитывает сумму функционального ряда в заданной точке (\a x) с точностью (\a eps).

\* \param x Абсцисса заданной точки.

\* \param eps Точность вычисления.

\* \return Сумма ряда.

\*/

double get\_row\_summ(const double x, const double eps);

/\*\*

\* \brief Рассчитывает значение рекуррентного члена в зависимости от индекса (\a index) в заданной точке (\a x).

\* \param index Индекс, номер итерации.

\* \param x Абсцисса заданной точки.

\* \return Значение рекуррентного члена.

\*/

double get\_recurrent\_member(const size\_t index, const double x);

/\*\*

\* \brief Пользовательский ввод значения.

\* \param message Мотивационное сообщение.

\* \return необходимое значение.

\* \exception std::invalid\_argument В случае, если введенное значения несоответствует условиям.

\*/

double get\_value(const std::string& message);

/\*\*

\* \brief Точка входа в программу.

\* \return 0 – в случае успеха,<br/>1 – в противном случае.

\*/

int main()

{

try {

setlocale(LC\_ALL, "Russian");

const double START = 0.1;

double x = START;

const double END = 1;

const double STEP = get\_value("введите шаг h ");;

const double EPS = get\_value("введите точность для вычисления функционального ряда e ");

while (x <= END + (std::numeric\_limits<double>::epsilon()))

{

const double Y = get\_function\_value(x);

const double S = get\_row\_summ(x, EPS);

std::cout << "x = " << std::setprecision(10) << x << " y = " << Y << " S = " << S << '\n';

x += STEP;

}

return 0;

}

catch (const std::exception& e)

{

std::cerr << e.what() << std::endl;

return 1;

}

}

double get\_value(const std::string& message)

{

std::cout << message;

double value = 0;

std::cin >> value;

if (value <= 0)

{

throw std::invalid\_argument("Значение должно быть больше 0!");

}

return value;

}

double get\_function\_value(const double x)

{

return 2 \* (pow(cos(x), 2) - 1);

}

double get\_row\_summ(const double x, const double eps)

{

int i = 1;

double summ = 0;

double current = eps + 1;

while (abs(current) >= eps) {

if (i == 1) {

current = -pow(2\*x, 2) / 2;

}

else {

current \*= get\_recurrent\_member(i - 1, x);

}

summ += current;

i++;

}

return summ;

}

double get\_recurrent\_member(const size\_t index, const double x)

{

return -1.0 \* pow(2\*x, 2) / (4 \* pow(index, 2) + 6 \* index + 2);

}

## Решение тестовых примеров на языке С++

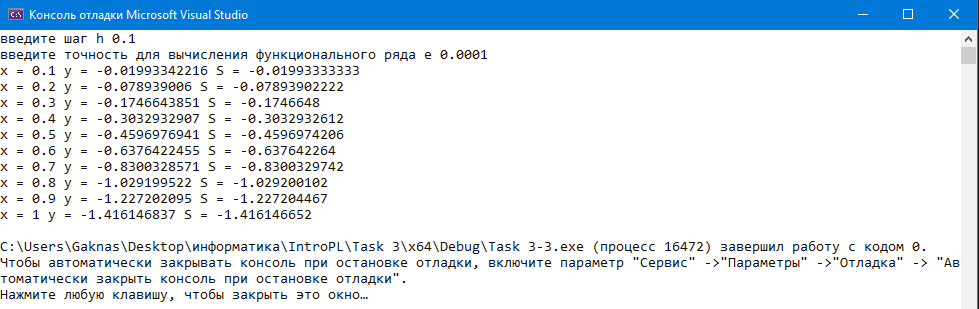


Рисунок 11 - Решение тестовых примеров для задания 3-3 на языке С++

## Решение тестовых примеров в Excel

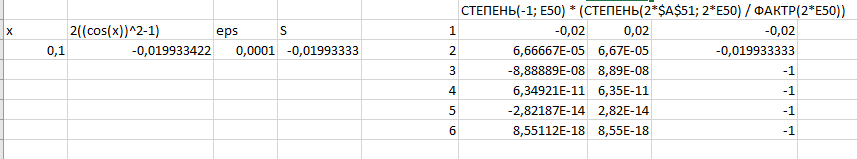


Рисунок 12 - Решение тестовых примеров для задания 3-3 в Excel