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 ЛАБОРАТОРНОЙ РАБОТЕ №3

По дисциплине «Языки программирования»

ВАРИАНТ 5

Выполнил: ст. гр. ТКИ-141

Буланый Сергей Евгеньевич

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

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

Москва 2023

Оглавление

[1 Решение задачи 1-1 3](#_Toc151483877)

[1.1 Формулировка задачи 2-1 3](#_Toc151483878)

[1.2 Блок-схема алгоритма задание 2–1 4](#_Toc151483879)

[1.3 Текст программы на языке С задание 2–1 5](#_Toc151483880)

[1.4 Результаты выполнения программы 2–1 6](#_Toc151483881)

[1.5 Выполнение тестовых примеров задание 2–1 7](#_Toc151483882)

[1.6 Отметка о выполнении задания в веб-хостинге системы контроля версий задание 2–1 13](#_Toc151483882)

[2 Решение задачи 2-2 14](#_Toc151483883)

[2.1 Формулировка задачи 2-2 14](#_Toc151483884)

[2.2 Блок-схема алгоритма задания 2-2 15](#_Toc151483885)

[2.3 Текст программы на языке С задание 2–2 19](#_Toc151483886)

[2.4 Результаты выполнения программы 2–2 21](#_Toc151483887)

[2.5 Выполнение тестовых примеров задание 2–2 23](#_Toc151483888)

[2.6 Отметка о выполнении задания в веб-хостинге системы контроля версий задание 2–2 26](#_Toc151483889)

[3 Решение задачи 1-3 27](#_Toc151483890)

[3.1 Формулировка задачи 1-3 27](#_Toc151483891)

[3.2 Блок-схема алгоритма задания 1-3 28](#_Toc151483892)

[3.3 Текст программы на языке С задание 1–3 30](#_Toc151483893)

[3.4 Результаты выполнения программы 1–3 32](#_Toc151483894)

[3.5 Выполнение тестовых примеров задание 1–3 33](#_Toc151483895)

[3.6 Отметка о выполнении задания в веб-хостинге системы контроля версий задание 1–3 34](#_Toc151483896)

1. Решение задачи 3–1

**1.1 Формулировка** **задачи 3–1**

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.

Таблица  – Исходные данные

|  |  |  |
| --- | --- | --- |
| Вариант | Функция | Константы |

|  |  |  |
| --- | --- | --- |
| 5 |  |  |

1.2 Блок-схема алгоритма задание 3–1

Блок-схема основного алгоритма представлена ниже (Рисунок 1). Блок-схемы функций расчета значений y представлены ниже (Рисунок 2).

Изображение выглядит как диаграмма, текст, Технический чертеж, План

Автоматически созданное описание

Рисунок  ­ Блок-схема основного алгоритма

Изображение выглядит как диаграмма, текст, Технический чертеж, План

Автоматически созданное описание

Рисунок  – Блок-схема используемых функций

1.3 Текст программы на языке С задание 3–1

#include <stdlib.h>

#include <math.h>

#include <stdbool.h>

#include <errno.h>

#include <float.h>

#include <stdio.h>

/\*\*

\* @brief Функция проверяющая существование функции в этой точке

\* @param x Входной параметр - точка x

\* @return Возвращает результат

\*/

double result(double x);

/\*\*

\* @brief Функция принимающая и проверяющая значение на ввод

\* @param message - текст сообщения для пользователя

\* @return Значение

\*/

double get\_value(const char\* message);

int main()

{

const double x\_start = get\_value("Start for x: ");

double x = x\_start;

const double x\_finish = get\_value("Finish for x: ");

const double dx = get\_value("Step for x: ");

while ((x - x\_finish) <= DBL\_MIN)

{

double y = result(x);

printf("%lf\t%0.2lf\t\n", x, y);

x += dx;

}

return 0;

}

double result(double x)

{

return ((cos(2 / x)) - 2 \* sin((1 / x) + (1 / x)));

}

double get\_value(const char\* message)

{

double a;

printf("%s", message);

int res = scanf\_s("%lf", &a);

if (res != 1)

{

errno = EIO;

perror("Wrong value");

abort();

}

return a;

}

* 1. Результаты выполнения программы 3–1

Результаты выполнения программы в C представлена ниже (Рисунок 3).

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 3 – Результаты выполнения программы

* 1. Выполнение тестовых примеров задание 3–1

В программе Photomathвыполнены тестовые примеры. Результаты их выполнения представлены ниже (Рисунок 4-14).

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 4 – Результат расчета если x=1

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 5 – Результат расчета если x=1.1

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Автоматически созданное описание

Рисунок 6 – Результат расчета если x=1.2

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 7 – Результат расчета если x=1.3

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 8 – Результат расчета если x=1.4

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Автоматически созданное описание

Рисунок 9 – Результат расчета если x=1.5

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 10 – Результат расчета если x=1.6

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Автоматически созданное описание

Рисунок 11 – Результат расчета если x=1.7

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 12 – Результат расчета если x=1.8

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 13 – Результат расчета если x=1.9

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 14 – Результат расчета если x=2

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

Ниже представлено доказательство того, что задание 3–1 было принято. (Рисунок 15)

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 15 – Approve задачи 3–1

1. Решение задачи 3–2

**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  – Исходные данные

|  |  |
| --- | --- |
| Вариант | Ряд |
| 5 |  |

2.2 Блок-схема алгоритма задание 3–2

Блок-схема основного алгоритма представлена ниже (Рисунок 1). Блок-схемы функций расчета значений count, sum, epsilon, recurrent представлены ниже (Рисунок 2-6).

Изображение выглядит как текст, чек, Шрифт, черно-белый

Автоматически созданное описание

Рисунок  ­ Блок-схема основного алгоритма

Изображение выглядит как текст, диаграмма, линия, Технический чертеж

Автоматически созданное описание

Рисунок  – Блок-схема используемой функции

Изображение выглядит как текст, снимок экрана, диаграмма, Шрифт

Автоматически созданное описание

Рисунок 3 – Блок-схема используемой функции

Изображение выглядит как текст, диаграмма, линия, Шрифт

Автоматически созданное описание

Рисунок 4 – Блок-схема используемой функции

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

Рисунок 5 – Блок-схема используемой функции

Изображение выглядит как текст, диаграмма, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 6 – Блок-схема используемой функции

2.3 Текст программы на языке С задание 3–2

#include <stdio.h>

#include <math.h>

#include <stdlib.h>

#include <errno.h>

#include <float.h>

/\*\*

\* @brief Формула для расчета факториала

\* @param k - число от которого факториал должен быть посчитан

\* @return Факториал

\*/

double factorial(double k);

/\*\*

\* @brief Функция принимающая и проверяющая значение на ввод

\* @param message - текст сообщения для пользователя

\* @return Значение

\*/

double get\_value(const char\* message);

/\*\*

\* @brief Основная функция в программе

\* @return 0 если процесс завершился без ошибок

\*/

int main()

{

double answer\_a = 0;

double answer\_b = 0;

double n = get\_value("Enter n: ");

double e = get\_value("Enter e: ");

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

{

answer\_a = answer\_a + factorial(i);

}

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

{

double element = factorial(i);

if (element - e >= DBL\_MIN)

{

answer\_b += element;

}

}

puts("Answer for a: ");

printf("%lf\n", answer\_a);

puts("Answer for b: ");

printf("%lf\n", answer\_b);

return 0;

}

double factorial(double k)

{

if (k < 2)

{

return 1;

}

else

{

return -((pow(k,3)+(3\*pow(k,2))+(3\*k)+1)/(pow(k,4)));

}

}

double get\_value(const char\* message)

{

double a;

printf("%s", message);

int res = scanf\_s("%lf", &a);

if (res != 1)

{

errno = EIO;

perror("Wrong value");

abort();

}

return a;

}

* 1. Результаты выполнения программы 3–2

Результаты выполнения программы в C представлена ниже (Рисунок 7-12)

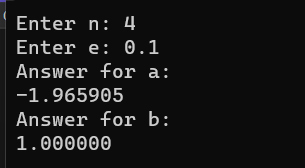


Рисунок 7 – Результаты выполнения программы если n=4 а e=0.1

Изображение выглядит как текст, Шрифт, снимок экрана, дизайн

Автоматически созданное описание

Рисунок 8 – Результаты выполнения программы если n=8 а e=0.2

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 9 – Результаты выполнения программы если n=13 а e=-1

Изображение выглядит как текст, Шрифт, снимок экрана, черный

Автоматически созданное описание

Рисунок 10 – Результаты выполнения программы если n=-5 а e=0.3

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 11 – Результаты выполнения программы если n=d

Изображение выглядит как текст, снимок экрана, Шрифт, мультимедиа

Автоматически созданное описание

Рисунок 12 – Результаты выполнения программы если n=38 а e=a

* 1. Выполнение тестовых примеров задание 3–2

В программе Pythonвыполнены тестовые примеры. Результаты их выполнения представлены ниже (Рисунок 12-17).

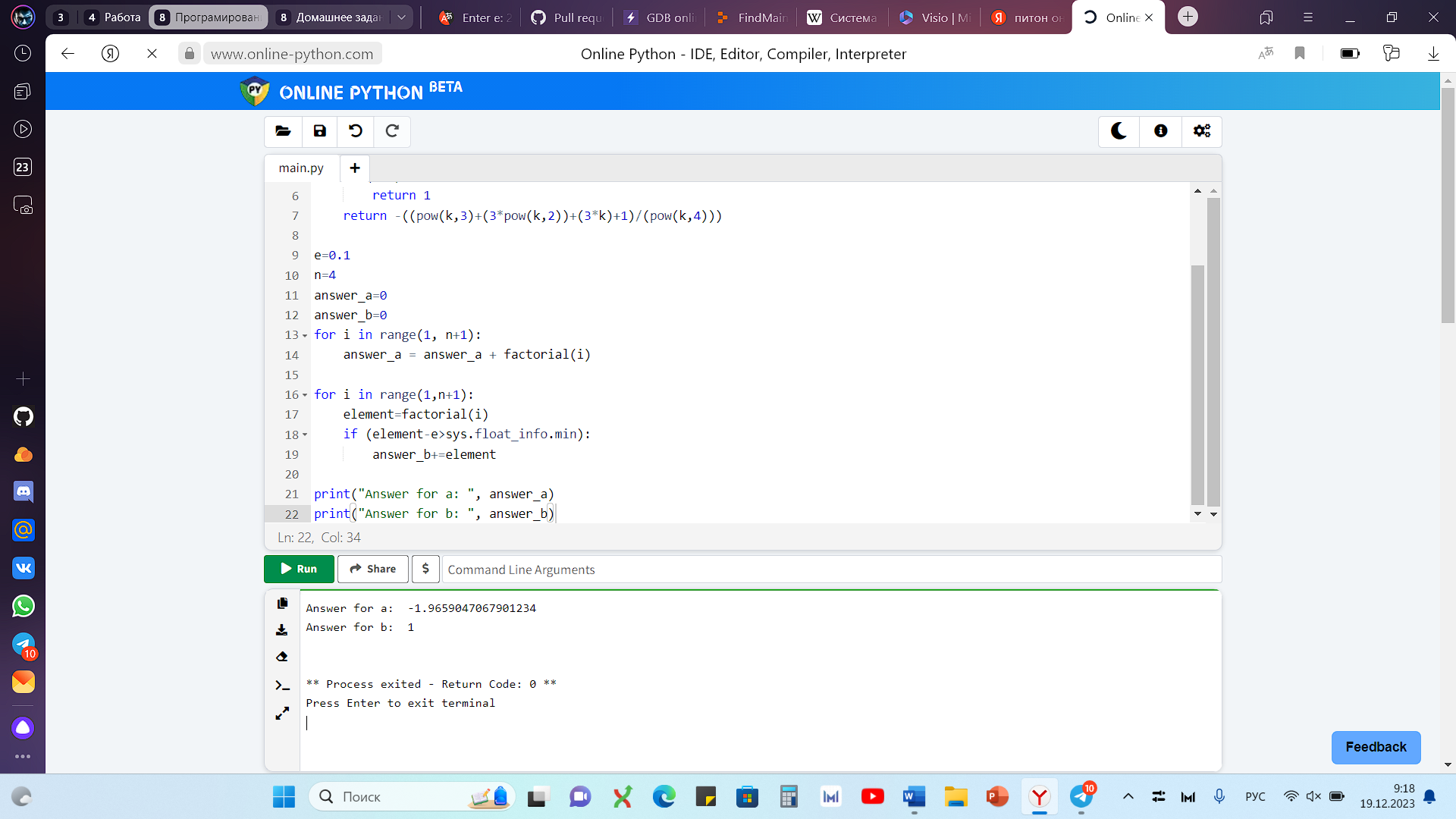


Рисунок 12 – Результат расчета если n=4 а e=0.1

Изображение выглядит как текст, снимок экрана, программное обеспечение, число

Автоматически созданное описание

Рисунок 13 – Результат расчета если n=8 а e=0.2

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 14 – Результат расчета если n=13 а e=-1

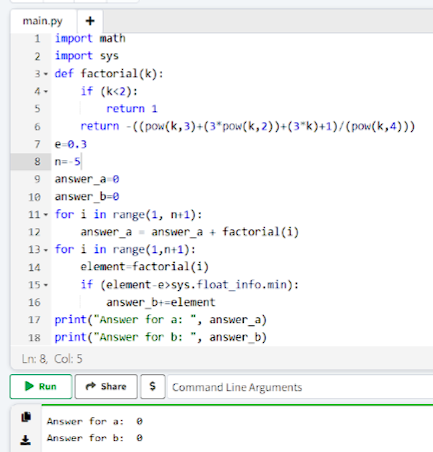


Рисунок 15 – Результат расчета если n=-5 а e=0.3

Изображение выглядит как текст, программное обеспечение, число, веб-страница

Автоматически созданное описание

Рисунок 16 – Результат расчета если n=d

Изображение выглядит как текст, снимок экрана, программное обеспечение, число

Автоматически созданное описание

Рисунок 17 – Результат расчета если n=38 а e=a

* 1. Отметка о выполнении задания в веб-хостинге системы контроля версий задание 3–2

Ниже представлено доказательство того, что задание 3–2 было принято. (Рисунок 18)

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 18 – Approve задачи 3–2

1. Решение задачи 3–3

**3.1 Формулировка** **задачи 2–3**

Протабулировать заданную функцию и сумму функционального ряда разложения этой функции на интервале [a, b] и с шагом h (шаг и интервал задается в константах). Функциональный 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.

Таблица  – Исходные данные

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Вариант | Функция y | Сумма | Интервал | ɛ |
| 5 |  |  |  | 30-5 |

3.2 Блок-схема алгоритма задание 3–3

Блок-схема основного алгоритма представлена ниже (Рисунок 1). Блок-схемы функций расчета значений function, summ представлены ниже (Рисунок 2-3).

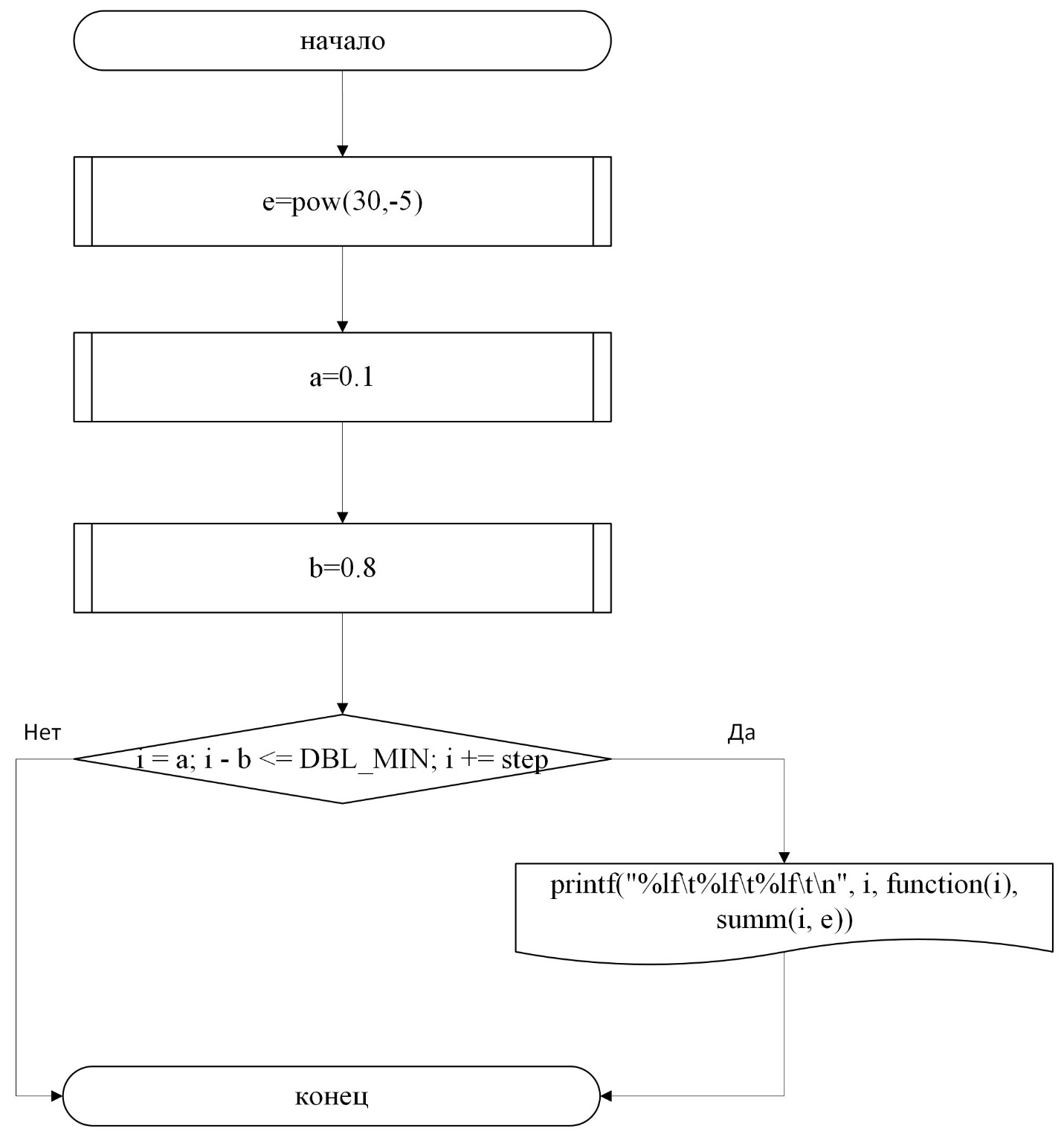


Рисунок  ­ Блок-схема основного алгоритма

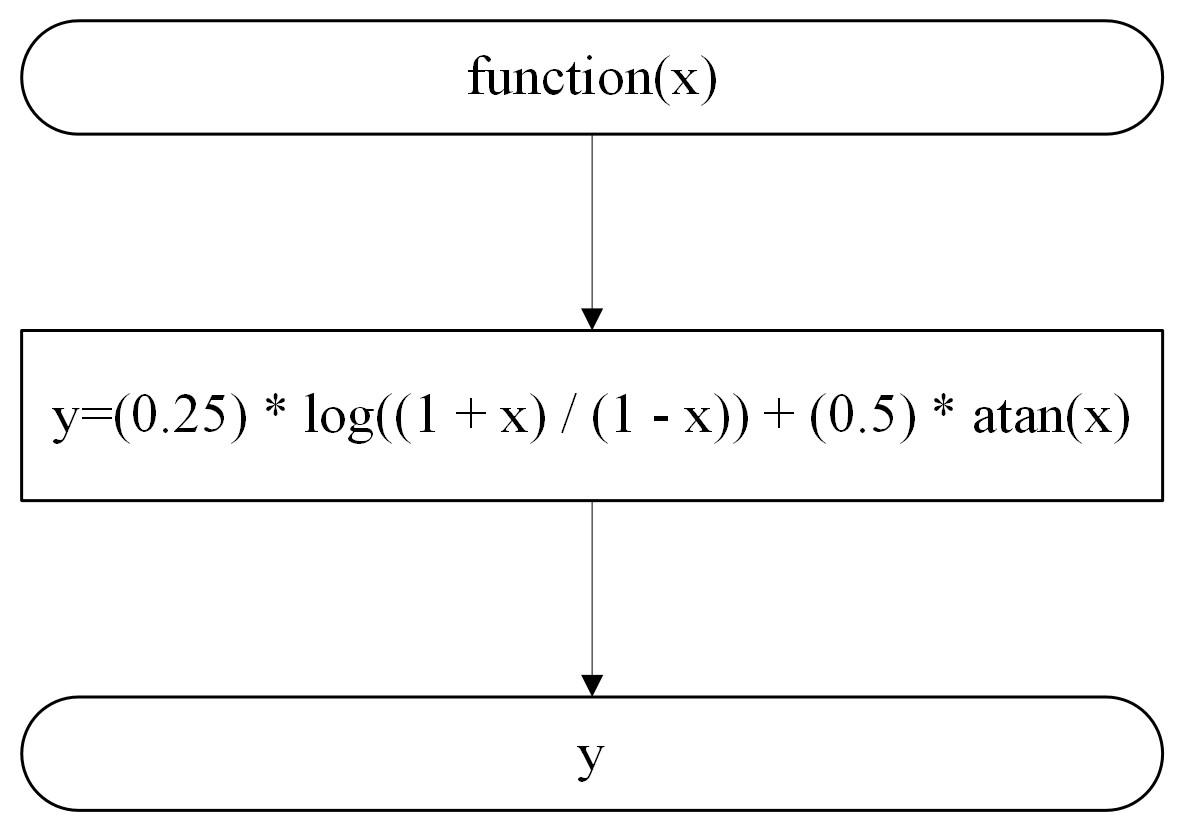


Рисунок  – Блок-схема используемой функции

Изображение выглядит как текст, диаграмма, линия, Параллельный

Автоматически созданное описание

Рисунок 3 – Блок-схема используемой функции

3.3 Текст программы на языке С задание 3–3

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <float.h>

#include <errno.h>

/\*\*

\* @brief Функция принимающая и проверяющая значение на ввод

\* @param message - текст сообщения для пользователя

\* @return Значение

\*/

double get\_step(const char\* message);

/\*\*

\* @brief Функция возвращающая значение функции в данной точке

\* @param x - точка

\* @return Значение функции в точке x

\*/

double function(double x);

/\*\*

\* @brief Функция возвращающая значение суммы ряда в этой точке

\* @param x - точка

\* @param e - точка

\* @return Значение суммы ряда в точке x

\*/

double summ(double x, double e);

/\*\*

\* @brief Основная функция в программе

\* @return 0 если процесс завершился без ошибок

\*/

int main()

{

double const a = 0.1;

double x = a;

double const b = 0.8;

double step = get\_step("Enter step: ");

double e = pow(30, -5);

for (double i = a; i - b <= DBL\_MIN; i += step)

{

printf("%lf\t%lf\t%lf\t\n", i, function(i), summ(i, e));

}

return 0;

}

double function(double x)

{

return (0.25) \* log((1 + x) / (1 - x)) + (0.5) \* atan(x);

}

double summ(double x, double e)

{

double sum = 0;

int i = 1;

double s\_i = -(4 \* i \* pow(x, 4) + pow(x, 4)) / (4 \* i + 5);

sum += s\_i;

for (int i = 2; s\_i > e; i++)

{

s\_i += -(4 \* i \* pow(x, 4) + pow(x, 4)) / (4 \* i + 5);

sum += s\_i;

}

return sum;

}

double get\_step(const char\* message)

{

double a;

printf("%s", message);

int res = scanf\_s("%lf", &a);

if (res != 1 || a <= 0)

{

errno = EIO;

perror("Wrong value");

abort();

}

return a;

}

3.4 Результаты выполнения программы 3–3

Результаты выполнения программы в C представлена ниже (Рисунок 4-10)

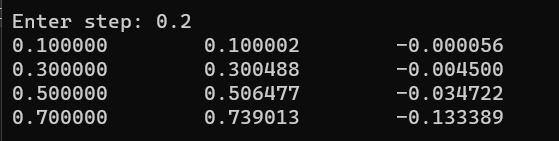


Рисунок 4 – Результаты выполнения программы если step=0.2

3.5 Выполнение тестовых примеров задание 3–3

В программе Pythonвыполнены тестовые примеры. Результаты их выполнения представлены ниже (Рисунок 5).

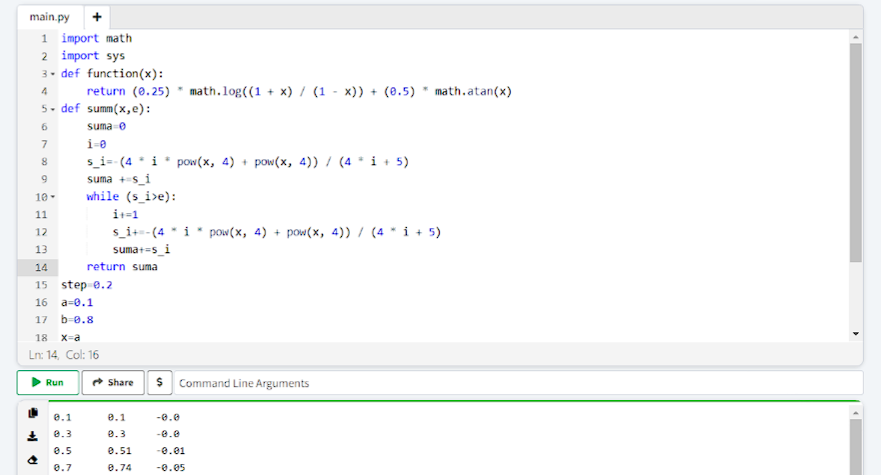


Рисунок 5 – Результат расчета если step=0.2

3.6 Отметка о выполнении задания в веб-хостинге системы контроля версий задание 3–3

Ниже представлено доказательство того, что задание 3–3 было принято. (Рисунок 6)

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 6 – Approve задачи 3–3