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 2

Выполнил: ст. гр. ТКИ - 142

Бахров Кирилл Сергеевич

Проверил: к.т.н., доц. Васильева М. А Проверил: к.т.н., доц. Балакина Е. П.

Москва 2023

Оглавление

**Решение задачи 3.13**

Формулировка задания3

Блок-схема алгоритма4-5

Текст программы на языке C6-8

Результат выполнения программы9

Выполнение тестовых примеров10

Отметка о выполнении задания в веб-хостинге системы контроля версий11

**Решение задачи 3.1 доп12**

Формулировка задания12

Блок-схема алгоритма13

Текст программы на языке C14

Результат выполнения программы15

Выполнение тестовых примеров16

Отметка о выполнении задания в веб-хостинге системы контроля версий17

**Решение задачи 3.2**1**8**

Формулировка задания18

Блок-схема алгоритма19-20

Текст программы на языке C21-24

Результат выполнения программы25

Выполнение тестовых примеров26-27

Отметка о выполнении задания в веб-хостинге системы контроля версий28

**Решение задачи 3.2 доп**29

Формулировка задания29

Блок-схема алгоритма30

Текст программы на языке C31-32

Результат выполнения программы33

Выполнение тестовых примеров34

Отметка о выполнении задания в веб-хостинге системы контроля версий35

1. РЕШЕНИЕ ЗАДАЧИ 3.1
   1. Формулировка задания

Протабулировать заданную в таблице функцию. (Таблица 1) 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. При решении данного задания в MatLab необходимо построить график!

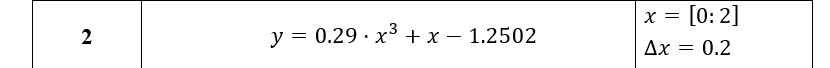


Таблица  – Исходные данные

* 1. Блок-схема алгоритма

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

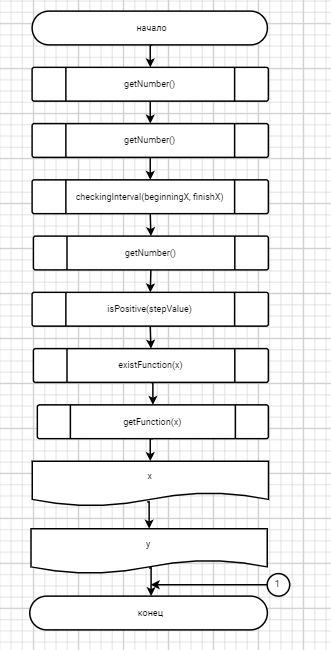


Рисунок  ­ Блок-схема основного алгоритма

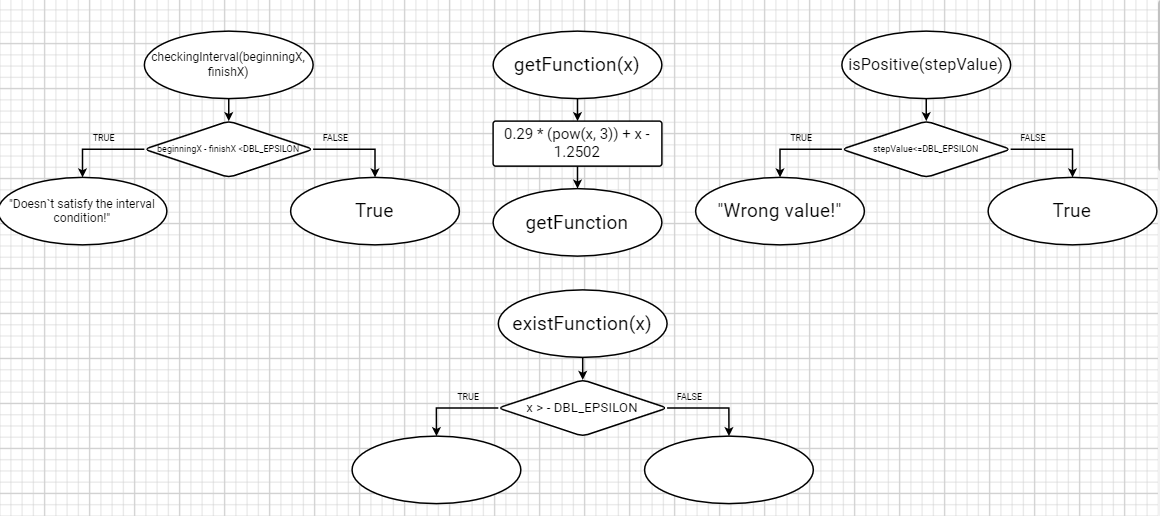


Рисунок  – Блок-схема используемых функций

* 1. Текст программы на языке C

#include <stdio.h>

#include <math.h>

#include <stdlib.h>

#include <stdbool.h>

#include <float.h>

/\*\*

@brief считывает и проверияет ввод переменной

@return возвращает значение переменной

\*/

double getNumber();

/\*\*

@brief проверяет условие, чтобы начальное значение интервала было меньше конечного

@param beginningX - значение начала интервала

@param finishX - значение rjywf интервала

@return возвращает 1 если условие выполнено

\*/

double checkingInterval(double beginningX, double finishX);

/\*\*

@brief проверка значение на положительное число

@param stepValue - значение переменной stepValue

@return возвращает значение переменной

\*/

void isPositive(double stepValue);

/\*\*

@brief проверяет существует ли в функция в заданной точке

@param x - значение переменной x

@return возвращает true если функция существует

\*/

bool existFunction(double x);

/\*\*

@brief рассчитывает значение функции

@param x - значение переменной x

@return возвращает посчитанное значение

\*/

double getFunction(double x);

/\*\*

@brief точка входа в программу

@return возвращает 0, если программа выполнена верно

\*/

int main()

{

printf("Enter the beginning of the interval: ");

double beginningX = getNumber();

printf("Enter the end of the interval: ");

double finishX = getNumber();

checkingInterval(beginningX, finishX);

printf("Enter the step value: ");

double stepValue = getNumber();

isPositive(stepValue);

double x = beginningX;

while(fabs(x) - fabs(finishX) > - DBL\_EPSILON)

{

if (existFunction(x))

{

double y = getFunction(x);

printf(" x = %lf", x);

printf(" y = %lf\n", y);

}

else

{

printf("The function does not exist, when x = %lf\n", x);

}

x = x + stepValue;

}

return 0;

}

double getNumber()

{

double entered\_number;

if (scanf("%lf", &entered\_number) != 1 )

{

printf("Wrong value");

abort();

}

return entered\_number;

}

void checkingInterval(double beginningX, double finishX)

{

if(beginningX - finishX < DBL\_EPSILON)

{

printf("Does not satisfy the interval condition!");

abort;

}

}

void isPositive(double stepValue)

{

if(stepValue <= DBL\_EPSILON){

puts("Wrong value");

abort;

}

}

bool existFunction(double x)

{

return x > - DBL\_EPSILON;

}

double getFunction(double x)

{

return( (0.29 \* ( pow(x, 3))) + x - 1.2502 );

}

* 1. Результаты выполнения программы

Результаты выполнения программы представлены ниже (Рисунок 3).

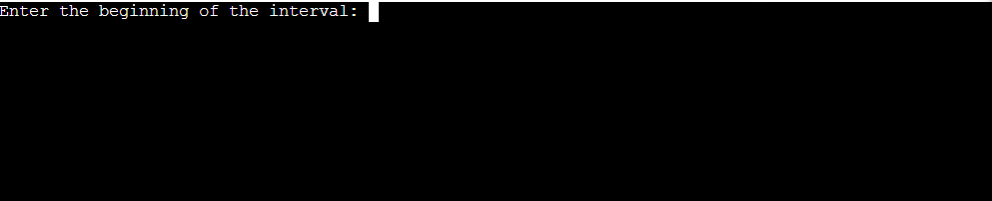


Рисунок 3 – Результаты выполнения программы

* 1. Выполнение тестовых примеров

В программе MS Excel выполнены тестовые примеры. Результаты их выполнения представлены ниже (Рисунок 4, Рисунок 5).

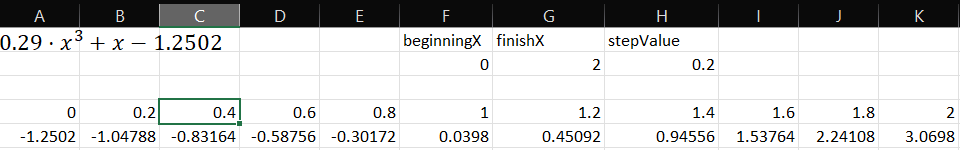


Рисунок 4 – Результат расчета значений функции

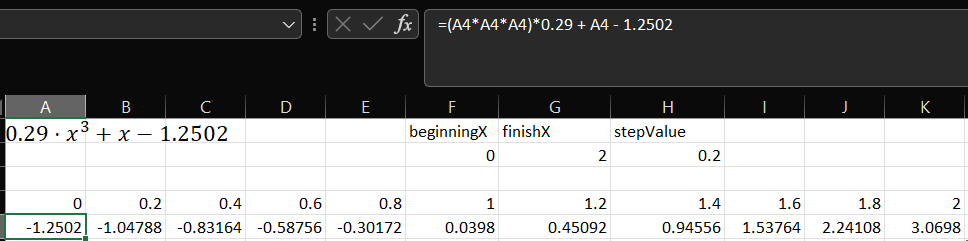


Рисунок 5 – Результат расчета значений функции

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



1. РЕШЕНИЕ ЗАДАЧИ 3.1 доп

2.1 Формулировка задания

При решении данных задач не забывайте дополнять свой отчёт пояснениями, почему выбран тот или иной метод решения.



Таблица 2 – Исходные данные

2.2 Блок-схема алгоритма

Блок-схема основного алгоритма представлена ниже (Рисунок 6).

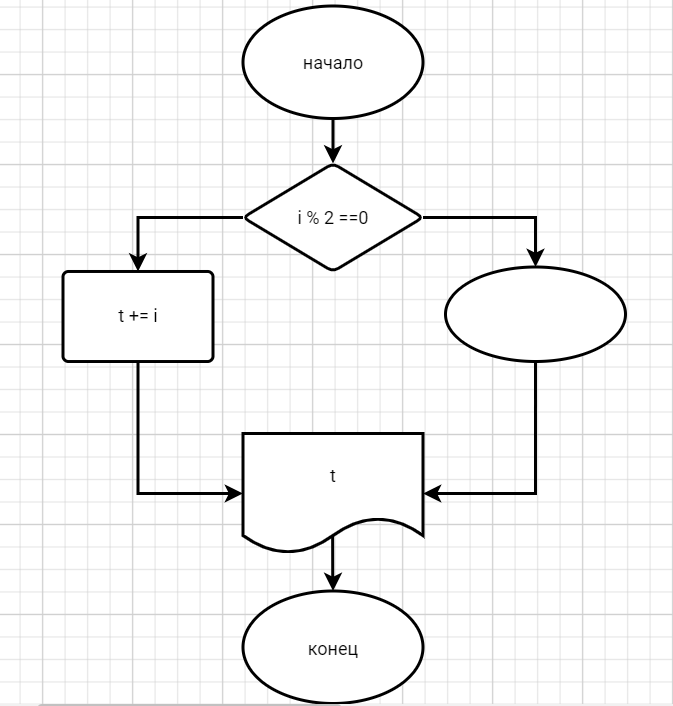


Рисунок 6 ­ Блок-схема основного алгоритма

2.3 Текст программы на языке C

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

/\*\*

@brief считает сумму всех четных чисел от 1 до 100

@return возвращает посчитанную сумму

\*/

int sumEvenNumbers();

/\*\*

@brief точка входа в программу

@return возвращает 0, если программа выполнена верно

\*/

int main()

{

int t = 0;

for(int i = 0; i <= 100; i++){

if(i % 2 == 0){

t += i;

}

}

printf("%d", t);

return 0;

}

2.4 Результаты выполнения программы

Результаты выполнения программы представлены ниже (Рисунок 7).

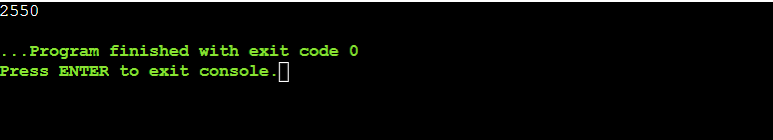


Рисунок 7– Результаты выполнения программы

2.5 Выполнение тестовых примеров

В программе MS Excel выполнены тестовые примеры. Результаты их выполнения представлены ниже (Рисунок 8).

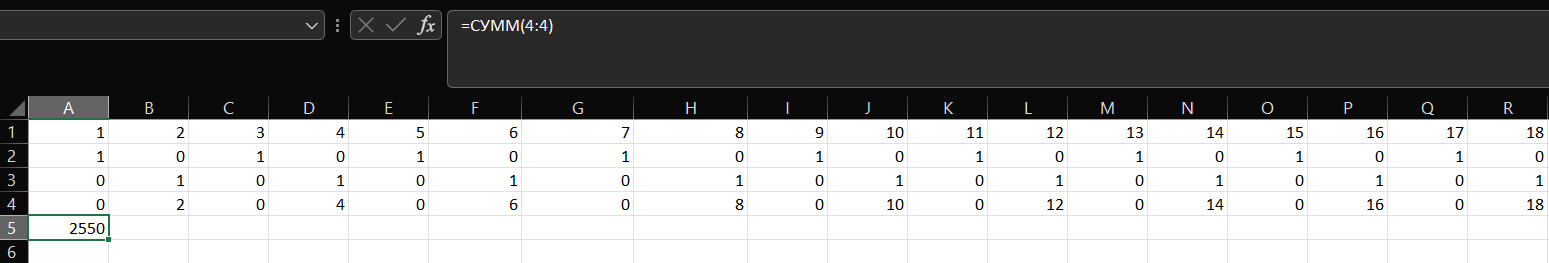


Рисунок 8 – Результат выполнения программы

2.6 Отметка о выполнении задания в веб-хостинге системы контроля версий



Рисунок 9 – Результат выполнения задания в веб хостинге контроля версий

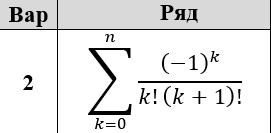
РЕШЕНИЕ ЗАДАЧИ 3.2  
3.1 Формулировка задания

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 3 – Исходные данные



3.2 Блок-схема алгоритма

Блок-схема основного алгоритма представлена ниже (Рисунок 10). Блок-схемы функций расчета значений a и b представлены ниже (11).

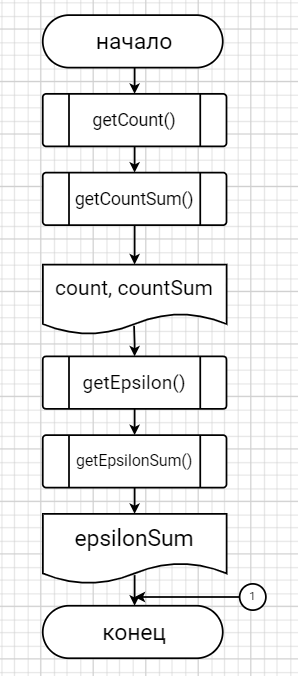


Рисунок 0 ­ Блок-схема основного алгоритма

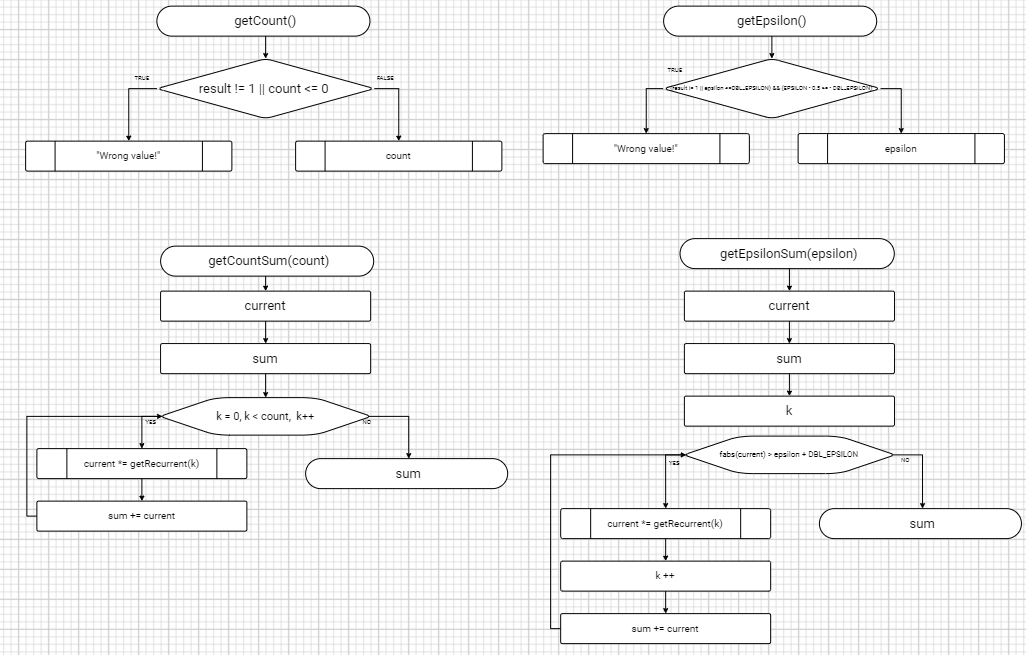


Рисунок 11 – Блок-схема используемых функций

3.3 Текст программы на языке C

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <float.h>

#include <errno.h>

#include <locale.h>

/\*\*

\* @brief проверяет на положительное число

\* @remarks Экстренное завершение программы, в случае неправильного ввода

\* @return возвращает Количество членов последовательности

\*/

int getCount();

/\*\*

\* @brief проверяет на функцию, считывающую точность вычисления суммы последовательности

\* @remarks Экстренное завершение программы, в случае неправильного ввода

\* @return возвращает точность вычисления суммы последовательности

\*/

double getEpsilon();

/\*\*

\* @brief рассчитывает функцию сумму членов последовательности

\* @param count - количество членов последовательности

\* @return возвращает сумму последовательности

\*/

double getCountSum(int count);

/\*\*

\* @brief рассчитывает сумму членов последовательности

\* @param epsilon - точность вычисления суммы последовательности

\* @return возвращает посчитанную сумму последовательности

\*/

double getEpsilonSum(double epsilon);

/\*\*

\* @brief расчитывает рекурентный член последовательности

\* @param k - номер члена последовательности

\* @return возвращает значение рекурентного члена последовательности

\*/

double getRecurrent(int k);

/\*\*

\*@brief Точка входа в программу

\*@return Возврящает 0, если программа работает верно, иначе 1

\*/

int main()

{

setlocale(LC\_ALL, "RU");

printf("Введите количество членов последовательности: ");

int count = getCount();

double countSum = getCountSum(count);

printf("Суммы %d членов последовательности равны: %.20lf \n", count, countSum);

printf("Введите точность вычисления суммы последовательности: ");

double epsilon = getEpsilon();

double epsilonSum = getEpsilonSum(epsilon);

printf("Сумма последовательности с заданной точностью равна: %.20lf", epsilonSum);

return EXIT\_SUCCESS;

}

int getCount()

{

int count;

int result = scanf("%d", &count);

if(result != 1 || count <= 0)

{

errno = EIO;

perror("Wrong value");

abort;

}

return count;

}

double getEpsilon()

{

double epsilon;

int result = scanf("%lf", &epsilon);

if((result !=1 || epsilon <= DBL\_EPSILON) && (epsilon - 0.5 >= -DBL\_EPSILON))

{

errno = EIO;

perror("Wrong value");

abort;

}

return epsilon;

}

double getCountSum(int count)

{

double current = 1.0; //

double sum = 0; // \*\*\*

for(int k = 0; k < count + 1; k++)

{

current \*= getRecurrent(k);

sum += current;

}

return sum;

}

double getEpsilonSum(double epsilon)

{

double current = 1.0; ///

double sum = 0;

int k = 0;

while (fabs(current) > epsilon + DBL\_EPSILON) /// &&&&

{

current \*= getRecurrent(k);

k++;

sum += current;

}

return sum;

}

double getRecurrent(int k)

{

return((-1)/((k+2)\*(k+1)));

}

3.4 Результаты выполнения программы

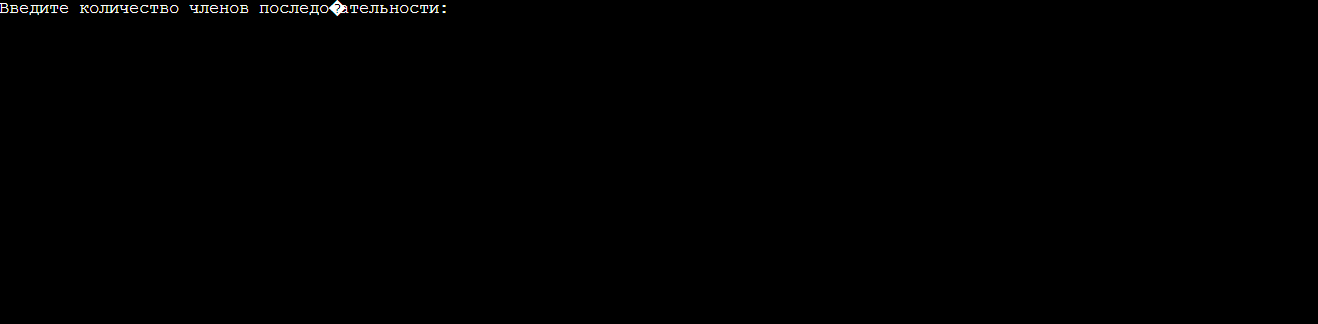


Рисунок 12 - Результаты выполнения программы

3.5 Выполнение тестовых примеров

В программе MS Excel выполнены тестовые примеры. Результаты их выполнения представлены ниже (Рисунок 13 – Рисунок 18).

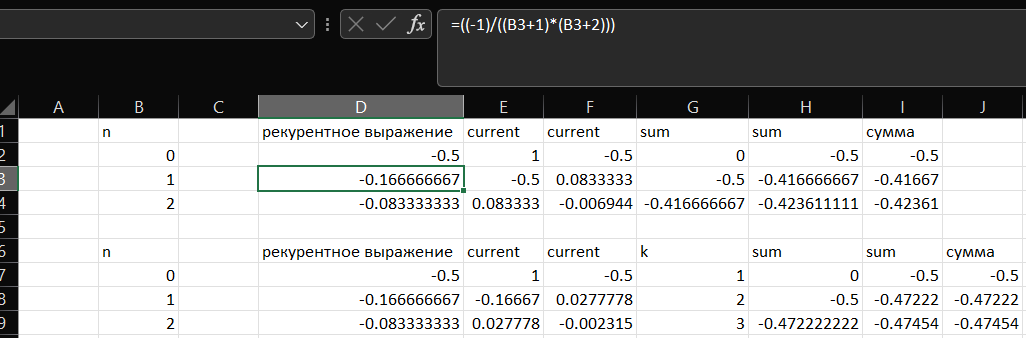
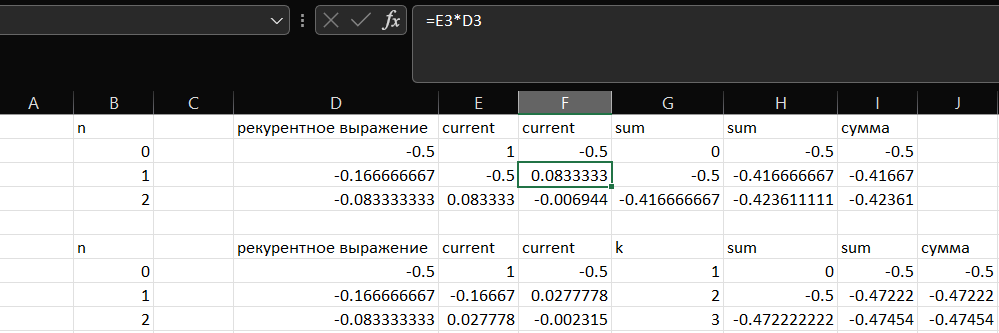
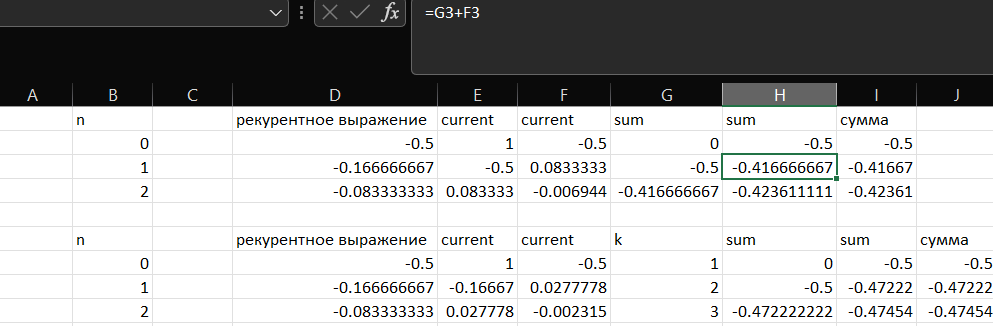
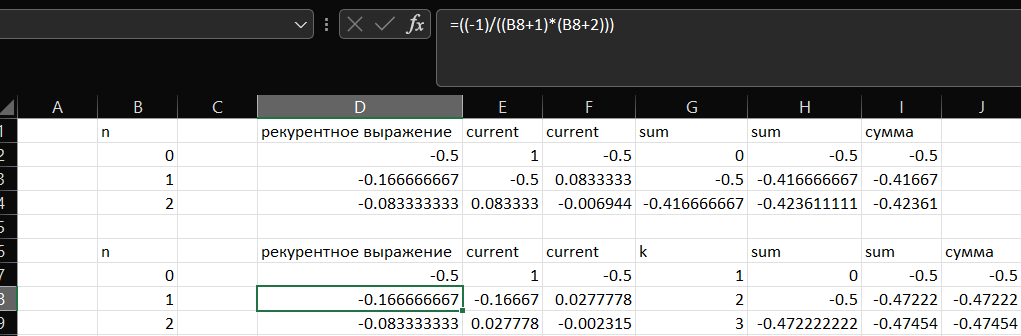
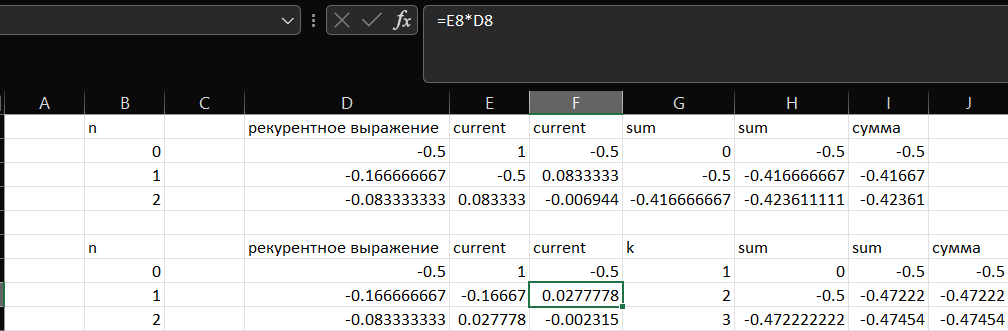


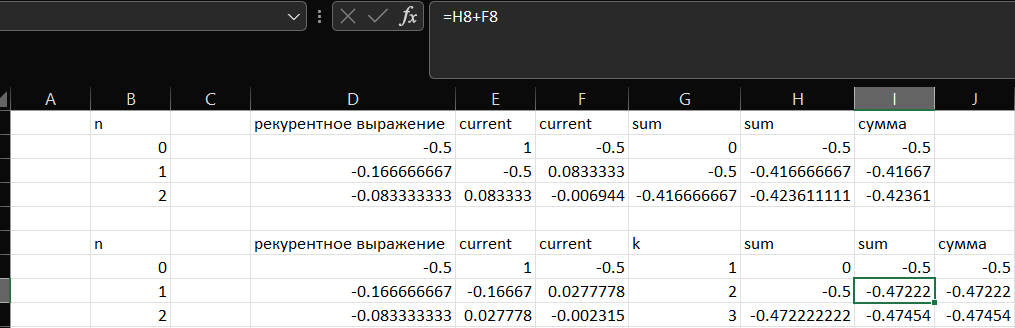
Рисунок 13 – Результат расчета функции

****Рисунок 14 – Результат расчета функции

****Рисунок 15 – Результат расчета функции

****Рисунок 16 – Результат расчета функции

****Рисунок 17 – Результат расчета функции

****Рисунок 18 – Результат расчета функции

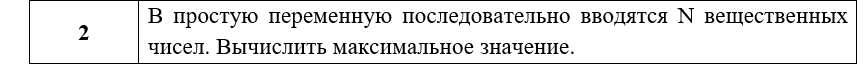
3.6 Отметка о выполнении задания в веб-хостинге системы контроля версий

Рисунок 19 – Результат выполнения задания в веб хостинге

РЕШЕНИЕ ЗАДАЧИ 3.2 доп  
4.1 Формулировка задания

При решении данных задач не забывайте дополнять свой отчёт пояснениями, почему выбран тот или иной метод решения.

Таблица 4 – Исходные данные



4.2 Блок-схема алгоритма

Блок-схема основного алгоритма представлена ниже (20). Блок-схемы функций расчета значений a и b представлены ниже (21).

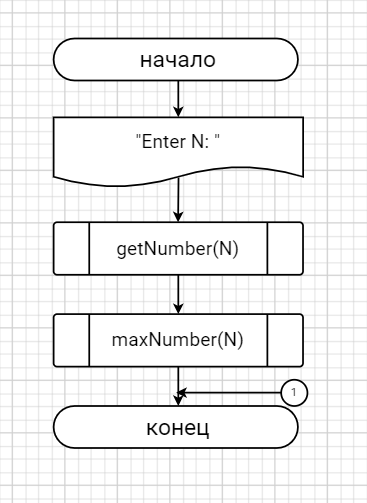


Рисунок 20 ­ Блок-схема основного алгоритма

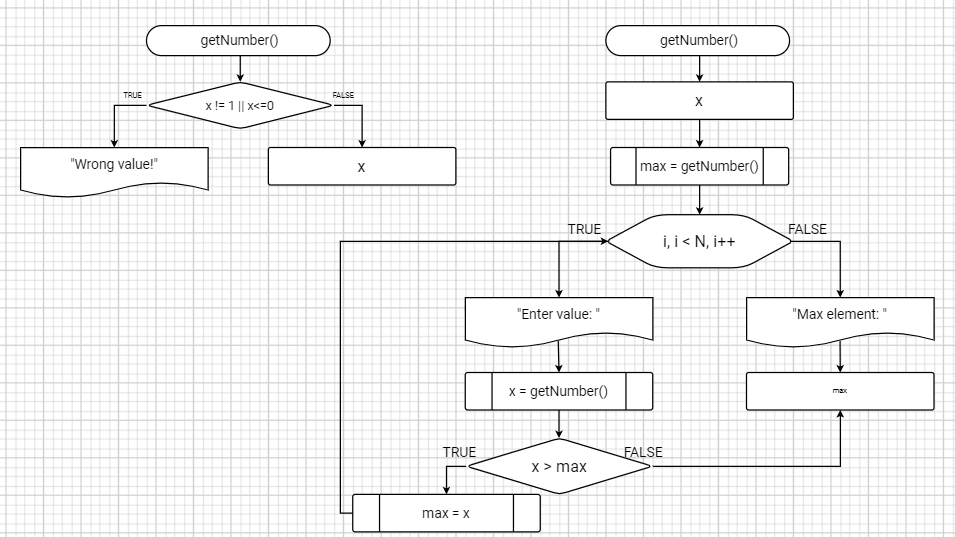
****

Рисунок 21 ­ Блок-схема функций

4.3 Текст программы на языке C

#include <stdio.h>

#include <stdlib.h>

/\*\*

\* @brief считывает и проверяет ввод переменной типа double

\* @return возвращает считанное значение

\*/

double getNumber();

/\*\*

@brief считает максимальное число из N введенных чисел

@return возвращает максимальное число из N введенных

\*/

double maxNumber(int N);

/\*\*

@brief точка входа в программу

@return возвращает 0, если программа выполнена верно

\*/

int main()

{

int N;

printf("Enter N: ");

getNumber(N);

maxNumber(N);

return 0;

}

double getNumber()

{

double x;

if (scanf("%lf", &x) !=1 || x <= 0)

{

puts("Wrong value");

abort();

}

return x;

}

double maxNumber(int N)

{

double x;

double max = getNumber();

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

{

printf("Enter value: ");

x = getNumber();

if (x > max)

{

max = x;

}

}

printf("Max element: %lf\n", max);

}

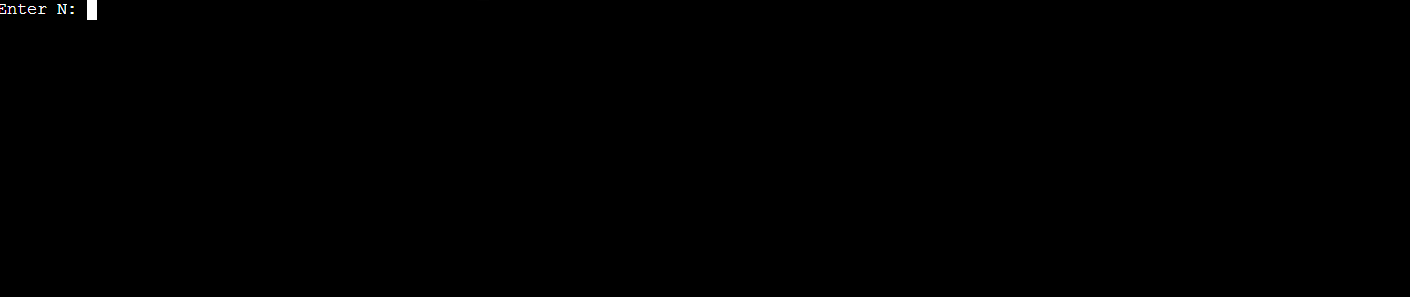
4.4 Результаты выполнения программы  


Рисунок 22 - Результаты выполнения программы

4.5 Выполнение тестовых примеров

В программе MS Excel выполнены тестовые примеры. Результаты их выполнения представлены ниже (Рисунок 23).

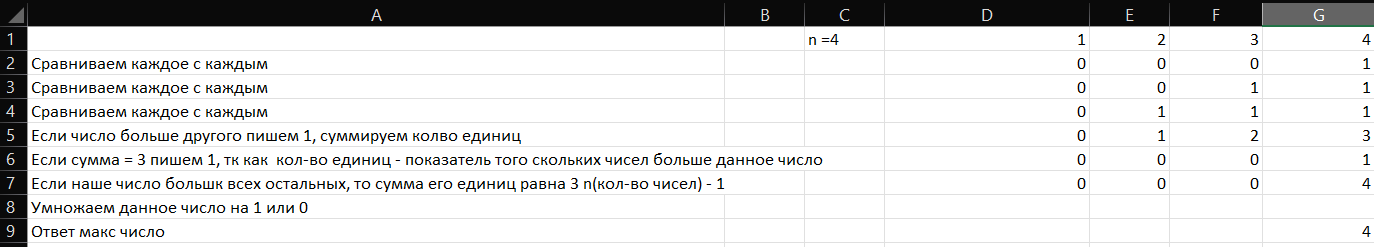


Рисунок 23 – Результат расчета функции

4.6 Отметка о выполнении задания в веб-хостинге системы контроля версий

Рисунок 24 – Результат выполнения задания в веб хостинге