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 №3.3

По дисциплине «Языки программирования»

ВАРИАНТ 3

Выполнил: ст. гр. ТКИ-142

Величко Иван Сергеевич

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

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

Москва 2023

1. Формулировка 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.

Таблица 1 – Исходные данные

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Вар | Функция y | Сумма | Интервал | ɛ |
| 3 |  |  | 1≤𝑥≤2 |  |

1. Блок-схема алгоритма

Ниже представлены:

Блок-схема основного алгоритма (Рисунок 1).

Блок-схема функции input() (Рисунок 2).

Блок-схема функции get\_step() (Рисунок 3).

Блок-схема функции calc() (Рисунок 4).

Блок-схема функции calc\_y() (Рисунок 5).

Блок-схема функции sum() (Рисунок 6).

Блок-схема функции reccurent() (Рисунок 7).



Рисунок 1– Блок-схема основного алгоритма

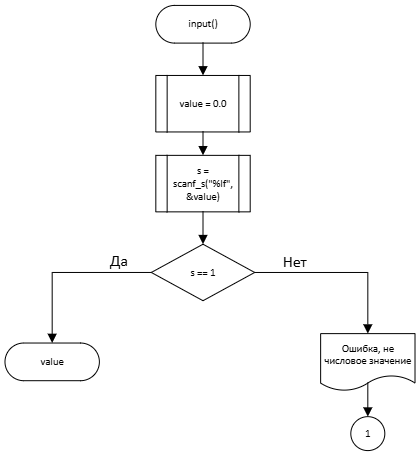


Рисунок 2 – Блок-схема функции input()

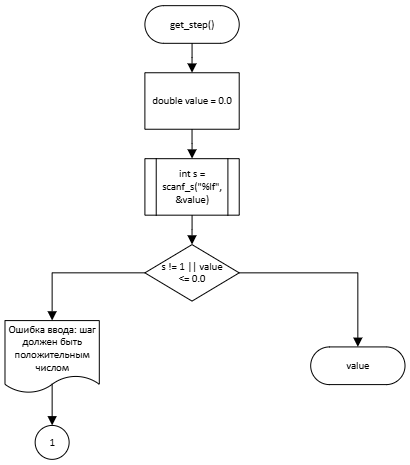


Рисунок 3 – Блок-схема функции get\_step()

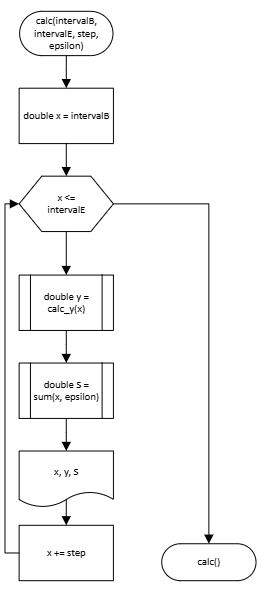


Рисунок 4 – Блок-схема функции calc()

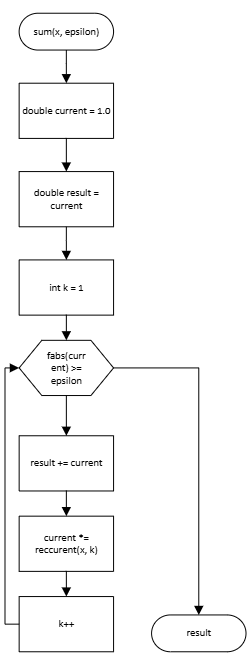


Рисунок 5 – Блок-схема функции sum()



Рисунок 6 – Блок-схема функции calc\_y()

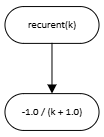


Рисунок 7 – Блок-схема recurent()

1. Текст программы на языке C

#include <stdio.h>

#include <errno.h>

#include <math.h>

#include <float.h>

#include <stdlib.h>

/\*\*

\* @brief принимает числовое значение из stdin

\* @return числовое значение полученное из stdin

\*/

double input(void);

/\*\*

\* @brief принимает положительное числовое значение из stdin

\* @return положительное числовое значение полученное из stdin

\*/

double get\_step(void);

/\*\*

\* @brief табулирует функцию в пределах ОДЗ, а так же расчитывает сумму ряда

\* @param intervalB передает значение начала интервала

\* @param intervalE передает значение конца интервала

\* @param step передает значение шага

\* @param epsilon передает значение точности

\*/

void calc(const double intervalB, const double intervalE, const double step, const double epsilon);

/\*\*

\* @brief расчитывает значение функции

\* @param x передает значение аргумента функции

\* @return возвращает значение функции (y)

\*/

double calc\_y(const double x);

/\*\*

\* @brief расчитывает сумму ряда

\* @param x передает значение аргумента функции

\* @param epsilon передает значение точности

\* @return возвращает сумму ряда

\*/

double sum(const double x, const double epsilon);

/\*\*

\* @brief возвращает следующий член последовательности по рекуррентной формуле

\* @param x передает значение аргумента функции

\* @param k актуальное порядковое значение элемента последовательности

\* @return возвращает следующий член последовательности по рекуррентной формуле

\*/

double reccurent(const double x, const double k);

/\*\*

\* @brief - точка входа в программу

\* @return 0 при успешном выполнении программы

\*/

int main(void) {

double epsilon = pow(15, -4);

puts("Пожалуйста, введите значения начала и конца интервала:");

double intervalB = input();

double intervalE = input();

puts("Пожалуйста, введите значение шага табулирования:");

double step = get\_step();

calc(intervalB, intervalE, step, epsilon);

return 0;

}

double input(void) {

double value = 0.0;

int s = scanf\_s("%lf", &value);

if (s != 1) {

errno = EIO;

perror("Ошибка, не числовое значение");

exit(EXIT\_FAILURE);

}

return value;

}

double get\_step(void) {

double value = 0.0;

int s = scanf\_s("%lf", &value);

if (s != 1 || value <= 0.0) {

errno = EIO;

perror("Ошибка ввода: шаг должен быть положительным числом");

exit(EXIT\_FAILURE);

}

return value;

}

void calc(const double intervalB, const double intervalE, const double step, const double epsilon) {

for (double x = intervalB; x <= intervalE + DBL\_EPSILON; x += step) {

double fx = calc\_y(x);

double sx = sum(x, epsilon);

printf("x = %.2f f(x) = %.6f S(x) = %.6f\n", x, fx, sx);

}

}

double calc\_y(const double x) {

return exp(x);

}

double sum(const double x, const double epsilon) {

double current = 1.0;

double result = current;

int k = 1;

while (fabs(current) >= epsilon) {

current \*= x / k;

result += current;

k++;

}

return result;

}

double reccurent(const double x, const double k) {

return x / k;

}

1. Результаты выполнения программы

Результаты выполнения программы представлены ниже (Рисунок 8).

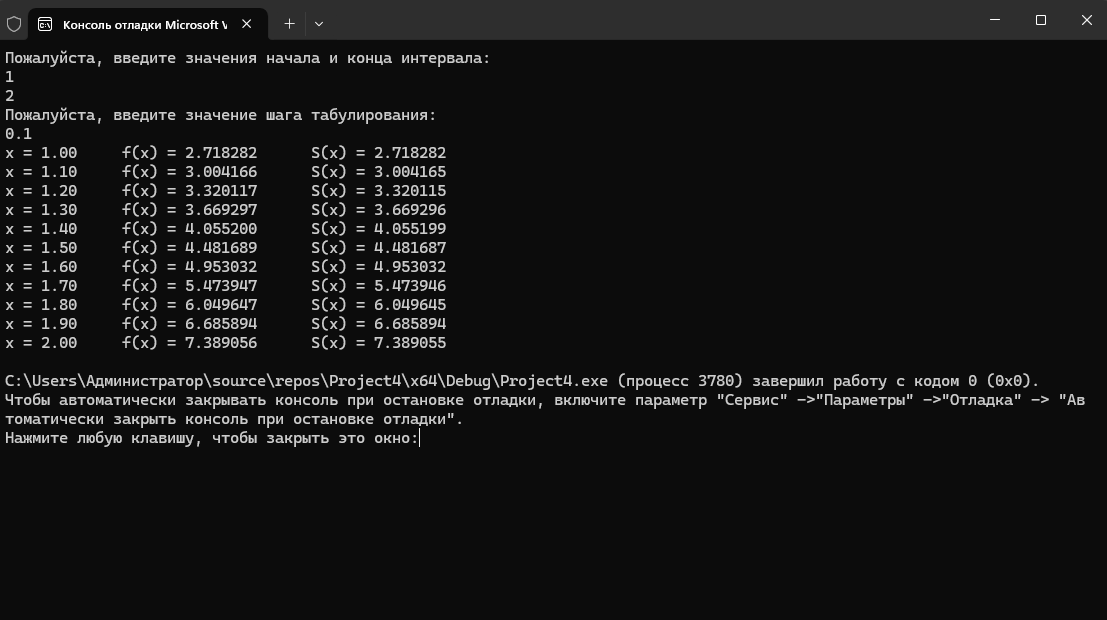


Рисунок 8 – Результат выполнения программы для первых 10 членов последовательности.

1. Выполнение тестовых примеров

Результат табулирования функции на интервале 1≤𝑥≤2 с точностью в Mathlab (Рисунок 7, Рисунок 10, Рисунок 11, Рисунок 12).

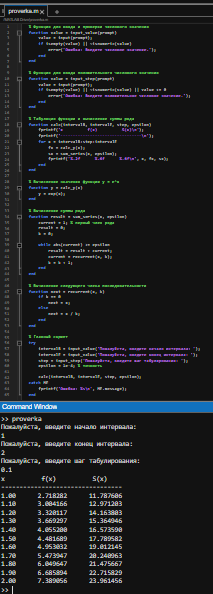


Рисунок 9 – Результат выполнения программы в Mathlab.

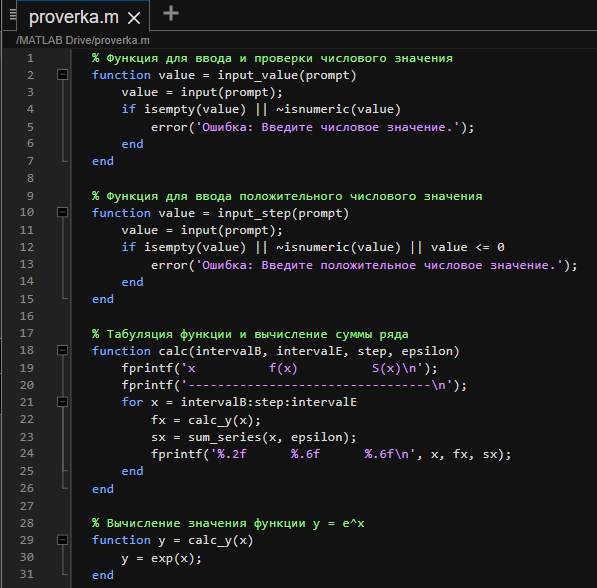


Рисунок – Первая часть программы

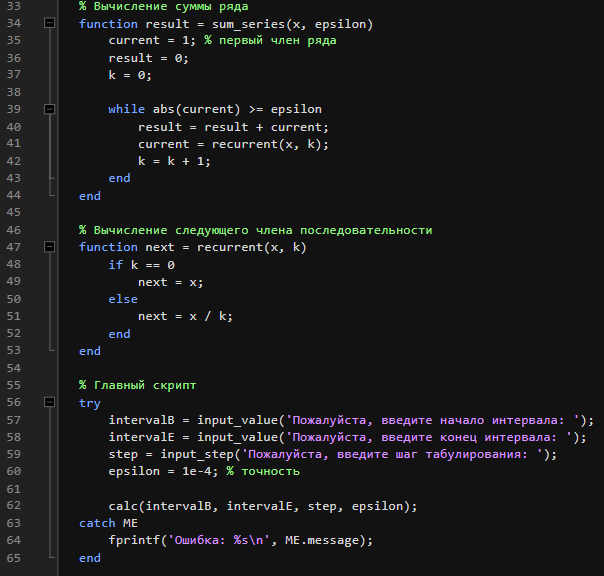


Рисунок 11 – Вторая часть программы

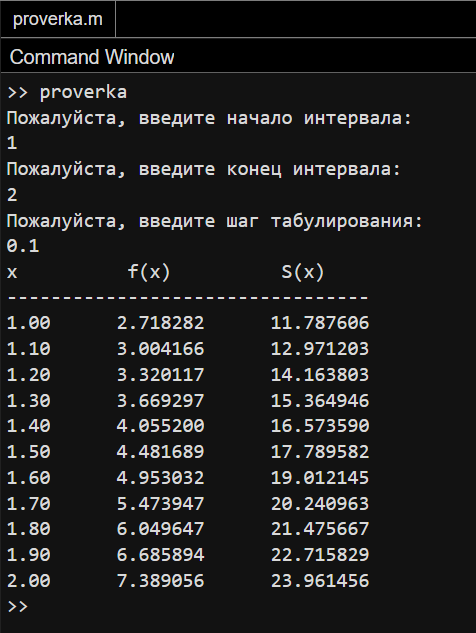


Рисунок 12 – Значения полученные в Mathlab.

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

Отметка о выполнение задания представлена ниже (Рисунок 10)

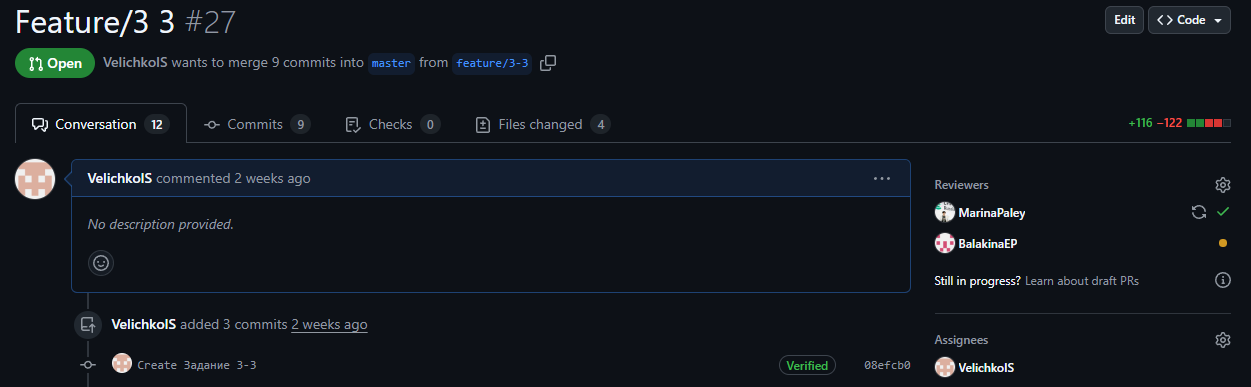


Рисунок 13 – Подтверждение выполнения задания