**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ**

**імені ІГОРЯ СІКОРСЬКОГО»**

**Інститут прикладного системного аналізу**

**Кафедра системного проектування**

**Звіт**

**про виконання лабораторної роботи №3**

**з дисципліни «Алгоритмізація та програмування»**

Виконав:  
студент I курсу, групи ДА-02

Рудік Андрій Миколайович  
Прийняв:

к.т.н., доцент Безносик О. Ю.

Київ – 2020

Завдання №1

Обчислення суми довільної низки чисел

Аналіз умови задачі

Для виконання задачі слід використати нескінченний цикл, для виходу з якого користувачу потрібно ввести певне значення. Кожне число введене користувачем повинно додаватись до існуючої змінної, в якій повинна зберігатись сума усіх вже введених чисел. Блок-схема алгоритму представлена на рисунку 1.

Код програми

#include <locale.h>

#include <stdio.h>

*int* main() {

  setlocale(LC\_ALL, "rus");

*float* sum = 0;

*float* n;

  printf("Вводите произвольные числа. Чтобы узнать минимальное из уже введенных чисел, введите 0 или что-то, что не есть числом.\n");

  do {

    n = 0;

    scanf("%f", &n);

    sum += n;

  } while(n);

  printf("Сумма всех введенных чисел равна %f", sum);

  return 0;

}

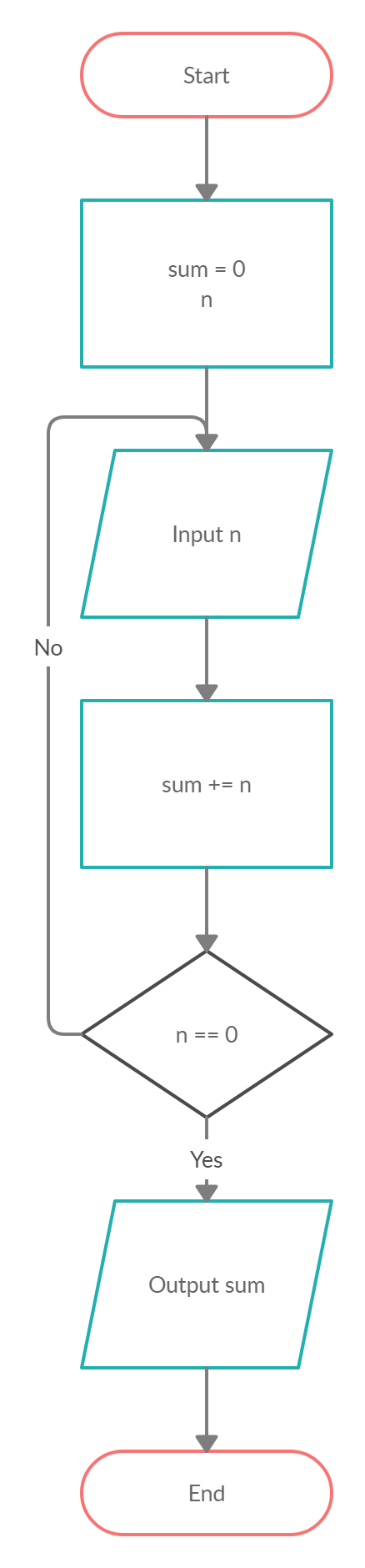


Рисунок 1 – Блок-схема алгоритму №1

Результат роботи програми представлений на рисунку 2

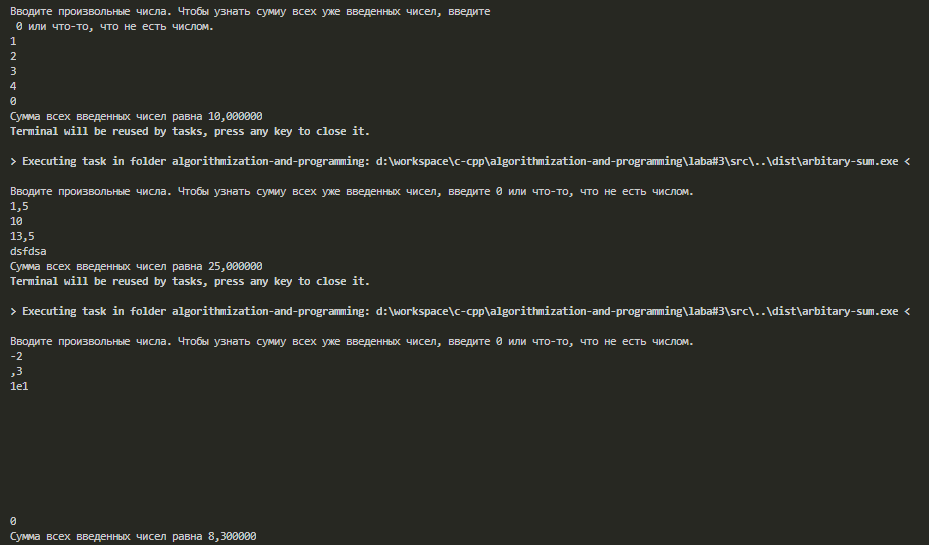


Рисунок 2 – Результат виконання програми №1

Завдання №2

Обчислення добутку:

; 

Аналіз умови задачі

Задача аналогічна попередній. Блок-схема алгоритму представлена на рисунку 3.

Код програми

#include <locale.h>

#include <stdio.h>

#include <math.h>

*float* getElement(*float* *x*) {

  return pow(*x*, 2) / (1 - *x*);

}

*int* main() {

  setlocale(LC\_ALL, "rus");

*float* res = 1;

*float* n;

  printf("Вводите произвольные числа. Чтобы узнать произведение всех уже введенных чисел согласно формуле, введите 1 или что-то, что не есть числом.\n");

  while (1) {

    n = 1;

    scanf("%f", &n);

    if (n == 1) break;

    res \*= getElement(n);

  };

  printf("Произведение всех введенных чисел согласно формуле равно %f", res);

  return 0;

}

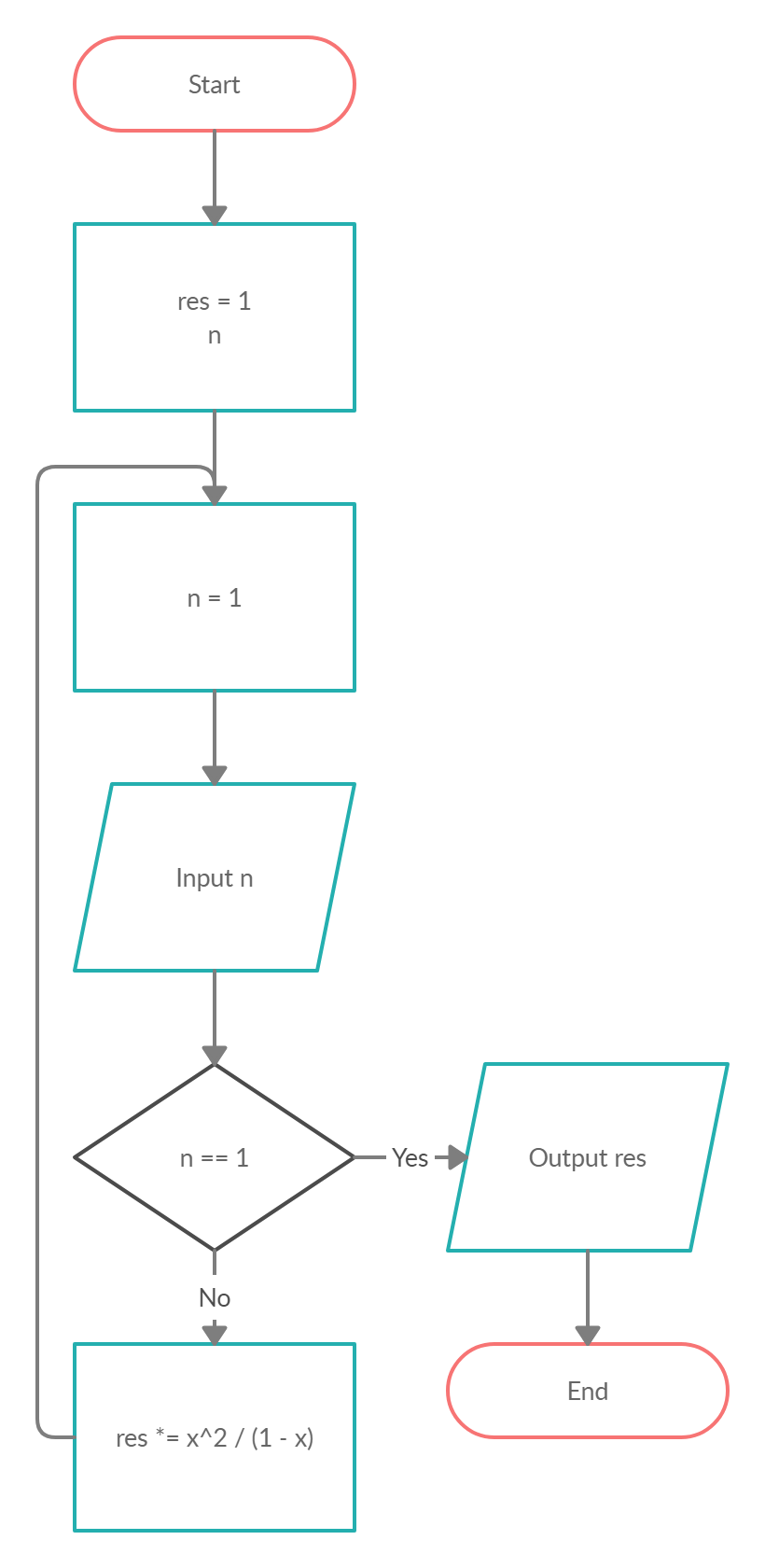


Рисунок 3 – Блок-схема алгоритму №1

Результат роботи програми представлений на рисунку 4

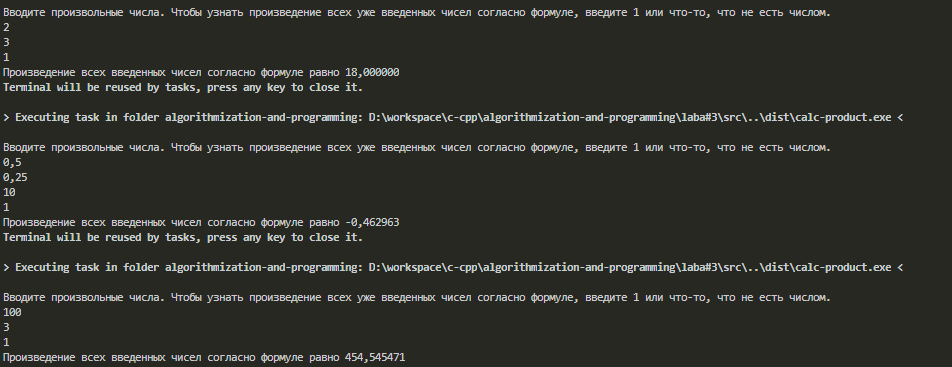


Рисунок 4 – Результат виконання програми №2

Завдання №3

Обчислення числа сполучень з m чисел по n.

;

Аналіз умови задачі

Розрахування числа сполучень за формулою. Блок-схема алгоритму представлена на рисунку 5.

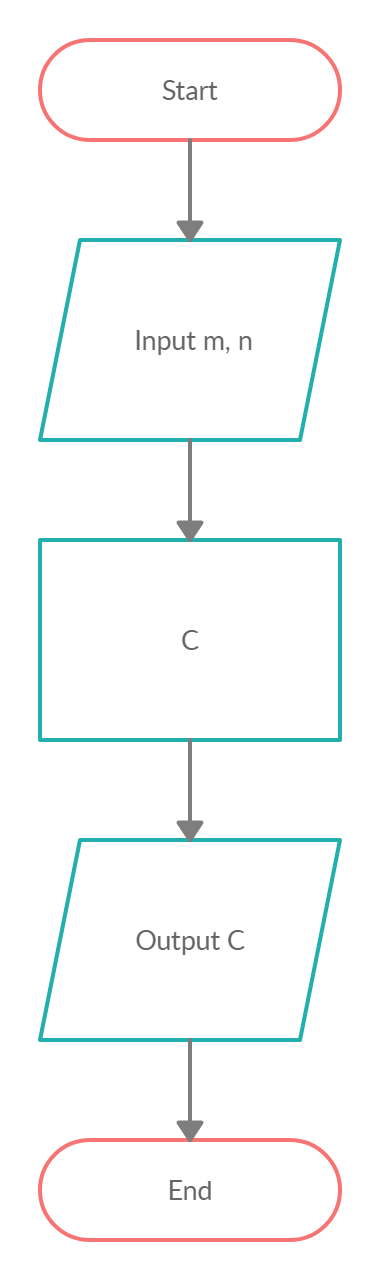


Рисунок 5 – Блок-схема алгоритму №3

Код програми

#include <stdio.h>

#include <math.h>

#include <locale.h>

*unsigned* *int* getFactorial(*unsigned* *int* *n*) {

*unsigned* *int* factorial = 1;

  for (*unsigned* *int* i = 2; i <= *n*; i++) {

    factorial \*= i;

  }

  return factorial;

}

*unsigned* *int* getCombinations(*unsigned* *int* *m*, *unsigned* *int* *n*) {

  return getFactorial(*m*) / (getFactorial(*n*) \* getFactorial(*m* - *n*));

}

*int* main() {

  setlocale(LC\_ALL, "rus");

*unsigned* *int* m;

*unsigned* *int* n;

  printf("Введите числа m и n через пробел, для получения количества сочетаний с m по n\n");

  scanf("%u %u", &m, &n);

  printf("Количества сочетаний с m по n равно %u", getCombinations(m, n));

  return 0;

}

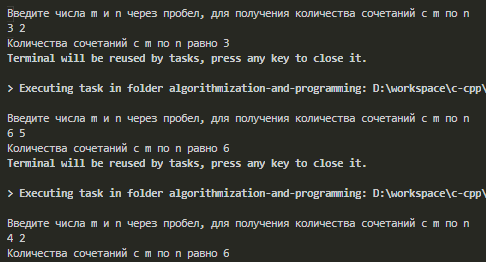
Результат роботи програми представлений на рисунку 6

Рисунок 6 – Результат виконання програми №3

Завдання №4

Скласти програму пошуку мінімального елемента з низки довільних чисел N, що вводяться з клавіатури.

Аналіз умови задачі

Аналогічно до двох перших задач.

Код програми

#include <locale.h>

#include <stdio.h>

*int* main() {

  setlocale(LC\_ALL, "rus");

*float* min;

*float* n;

  printf("Вводите произвольные числа. Чтобы узнать минимальное среди введенных чисел, введите 0 или что-то, что не есть числом.\n");

  while (1) {

    n = 0;

    scanf("%f", &n);

    if (n == 0) break;

    min = min > n ? n : min;

  };

  printf("Минимальное среди введенных чисел равно %f", );

  return 0;

}

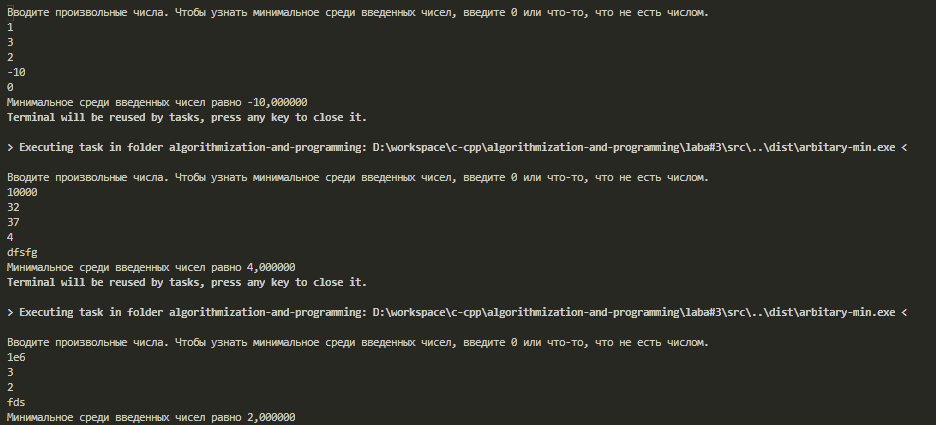
Результат роботи програми представлений на рисунку 7

Рисунок 7 – Результат виконання програми №4

Завдання №5

Скласти схему алгоритму і програму обчислень суми ряду з точністю **eps**1 = 0.01 і **eps**2 = 0.0001.Вивести в двох випадках на друк значення **s, n** (число ітерацій), **еps**

Аналіз умови задачі

Для обчислення суми ряду з певною точністю нам слід визначити коефіцієнт q. У моєму випадку він буде рівним . У цілому обчислення суми зводиться до нескінченного циклу, з котрого слід виходити в разі, якщо модуль нового коефіцієнту q стане меншим за епсилон. Блок-схема алгоритму представлена на рисунку 8.

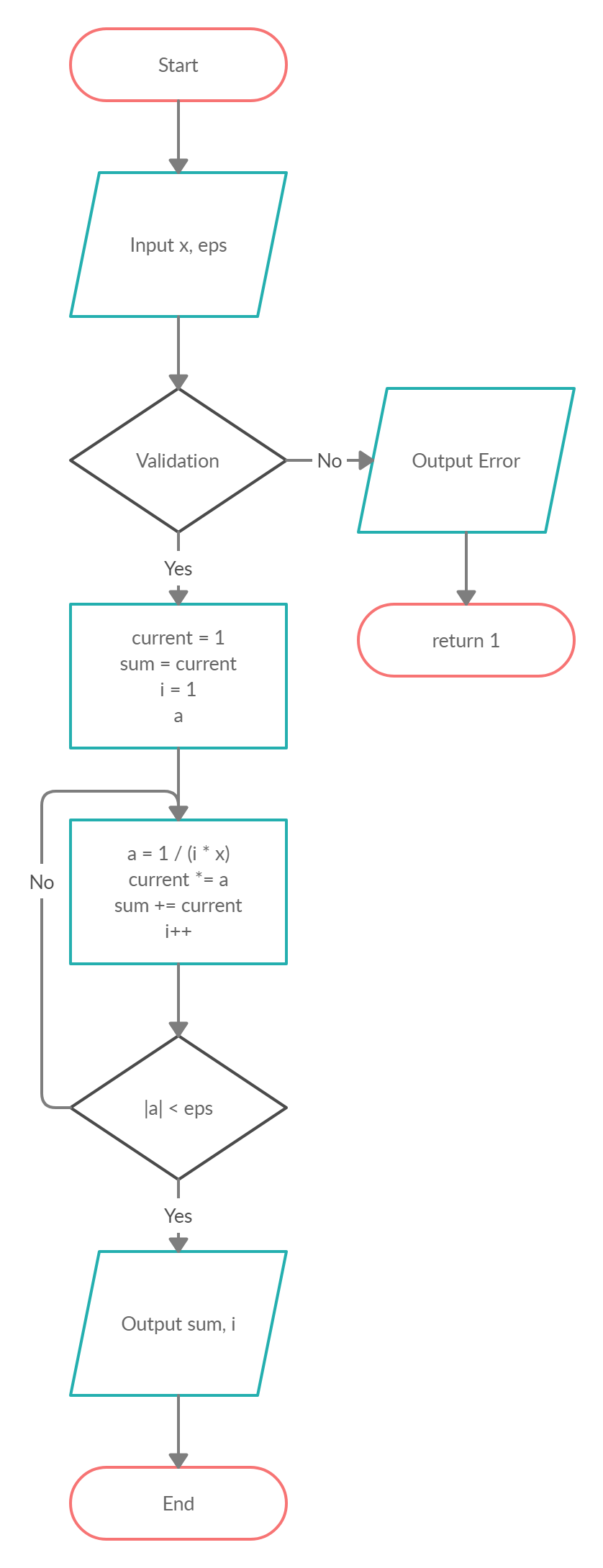


Рисунок 8 – Блок-схема алгоритму №5

Код програми

#include <locale.h>

#include <stdio.h>

#include <math.h>

*int* main() {

  setlocale(LC\_ALL, "rus");

*double* x;

  printf("Введите произвольное число х, такое что х != 0 и -1 < x < 1\n");

  scanf("%lf", &x);

*float* eps;

  printf("Введите натуральное число эпсилон которое отвечает за точность\n");

  scanf("%f", &eps);

  if (x == 0 || x >= 1 || x <= -1 || eps <= 0) {

    printf("Некоректный ввод\n");

    return 1;

  }

*long* *double* current = 1;

*long* *double* sum = current;

*int* i = 1;

*long* *double* q;

  do {

    q = 1. / (i \* x);

    current \*= q;

    sum += current;

    i++;

  } while(fabs(q) > eps);

  printf("Сумма ряда, вычисленная с точностью %f равна %Lf, на это пришлось потратить %d. Для сравнения e^(1/%lf) = %lf",

    eps, sum, i, x, exp(1. / x));

  return 0;

}

Результат роботи програми представлений на рисунку 9

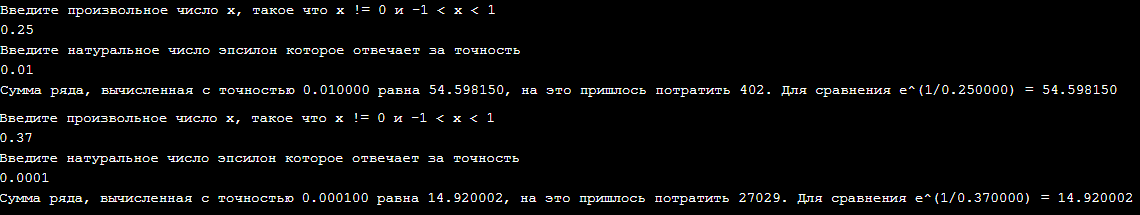


Рисунок 9 – Результат виконання програми №5

Висновки

У ході виконання лабораторної роботи я закріпив навички програмної реалізації алгоритмів розгалужених та циклічних структур, оволодів навичками реалізації найпростіших алгоритмів, включаючи алгоритм ітераційної циклічної структури: обчислив суми кінцевого числа доданків, обчислив добуток кінцевого числа множників, визначив найменше значення з усіх даних, що вводяться, обчислив суму збіжної послідовності із заданою точністю.