МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ

БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ФАКУЛЬТЕТ АВТОМАТИКИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ

КАФЕДРА РАДИОЭЛЕКТРОННЫХ СРЕДСТВ

Лабораторная работа №4

по дисциплине «Цифровые устройства и микропроцессоры»

Вариант 11

«МОДУЛЬНОЕ

ПРОГРАММИРОВАНИЕ»

Выполнил студент группы ИНБб-3301-02-00 \_\_\_\_\_\_\_\_\_\_/ К.А. Чашечников/

Проверил: преподаватель \_\_\_\_\_\_\_\_\_\_\_\_/М.А. Земцов/

Киров 2025

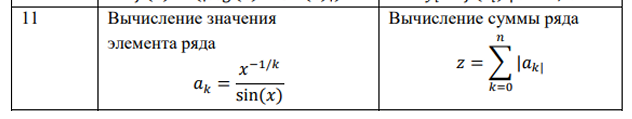
**1.** **Цель работы:**

знакомство с технологией применения языка ассемблера

при разработке программного обеспечения на языках высокого уровня.

**2. Ход работы:**

**Вариант задания:**



**Текст программы:**

**Main.cpp:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include <math.h>

#include<stdlib.h>

using namespace std;

extern "C" float SumR(int k, float x);

void main()

{

int n;

float x;

cout << "Input x ,n" << endl;

cin >> x >> n;

double R = 0.0;

R = SumR(n, x); //вызов внешней функции на ассемблере

cout << "Result" << R << endl;

}

extern "C" float fun\_sum(int i, float z)

{

float f;

f = powf(z,-1/(float)i)/sinf(z); //расчеты

return f;

}

**Function.asm:**

.586

.MODEL flat,C ; плоская модель в совместимости с C

.DATA

SUM DD 0.0

i\_local DD 0

.CODE

extern fun\_sum:near ; внешняя функция на C

public SumR ; функция на ассемблере для вызова из Cpp файла

SumR proc C

push ebp ; сохраняем регистр в памяти

mov ebp,esp ; новый кадр стека

mov i\_local,1

mov ecx, dword ptr [ebp+8] ; первый аргумент из стека

@@for\_i:

push ecx

push dword ptr [ebp + 12] ; второй элемент из стека

push i\_local

call fun\_sum ; вызов внешней функции

fabs ; по модулю

fld SUM ; заносим в fpu переменную

add esp,8

fadd

pop ecx

inc i\_local

fstp SUM

loop @@for\_i

fld SUM

mov esp,ebp

pop ebp

ret

SumR endp

End

**Верификация программы:**

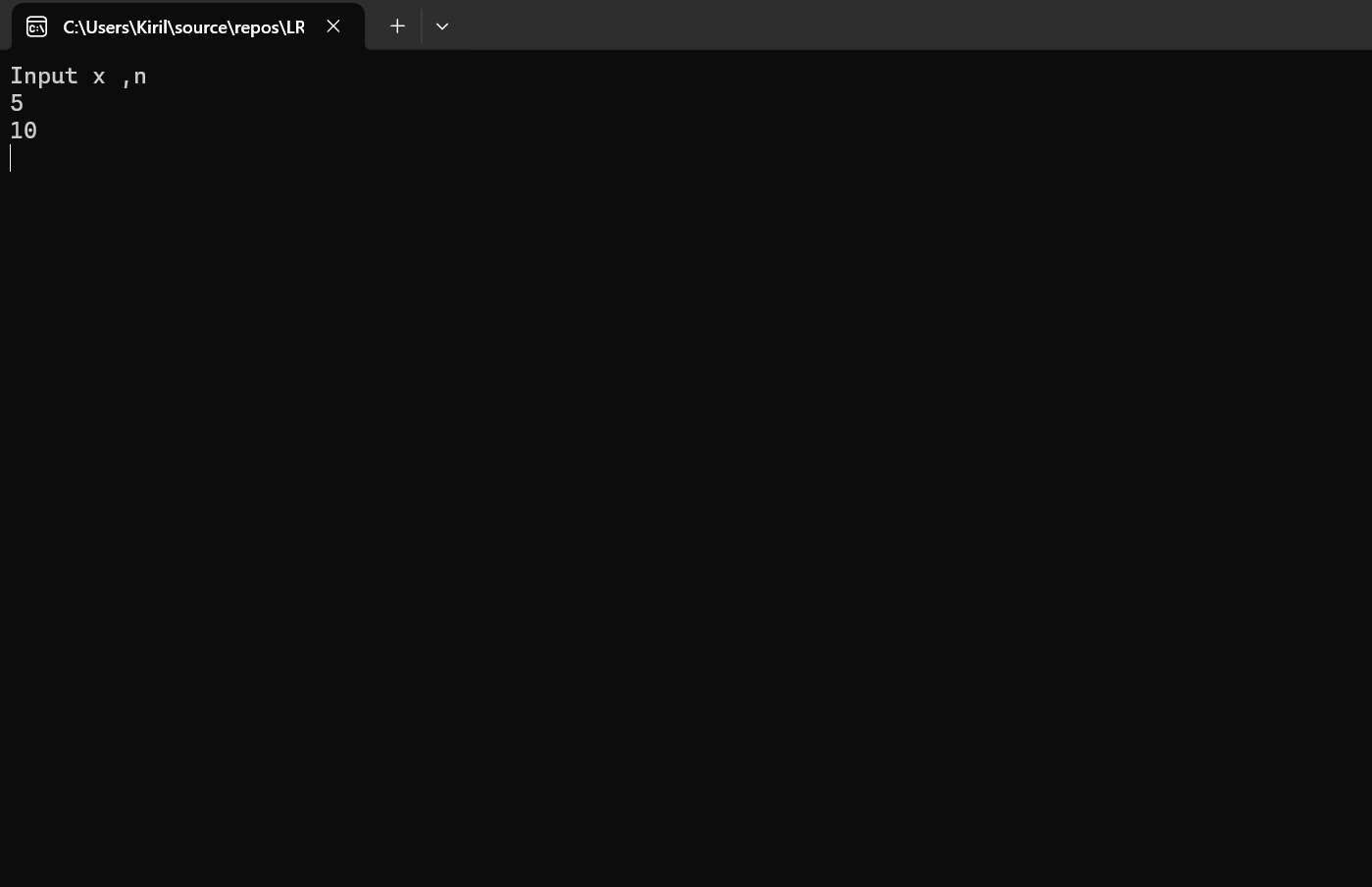
****

Рисунок 1 – Ввод начальных данных

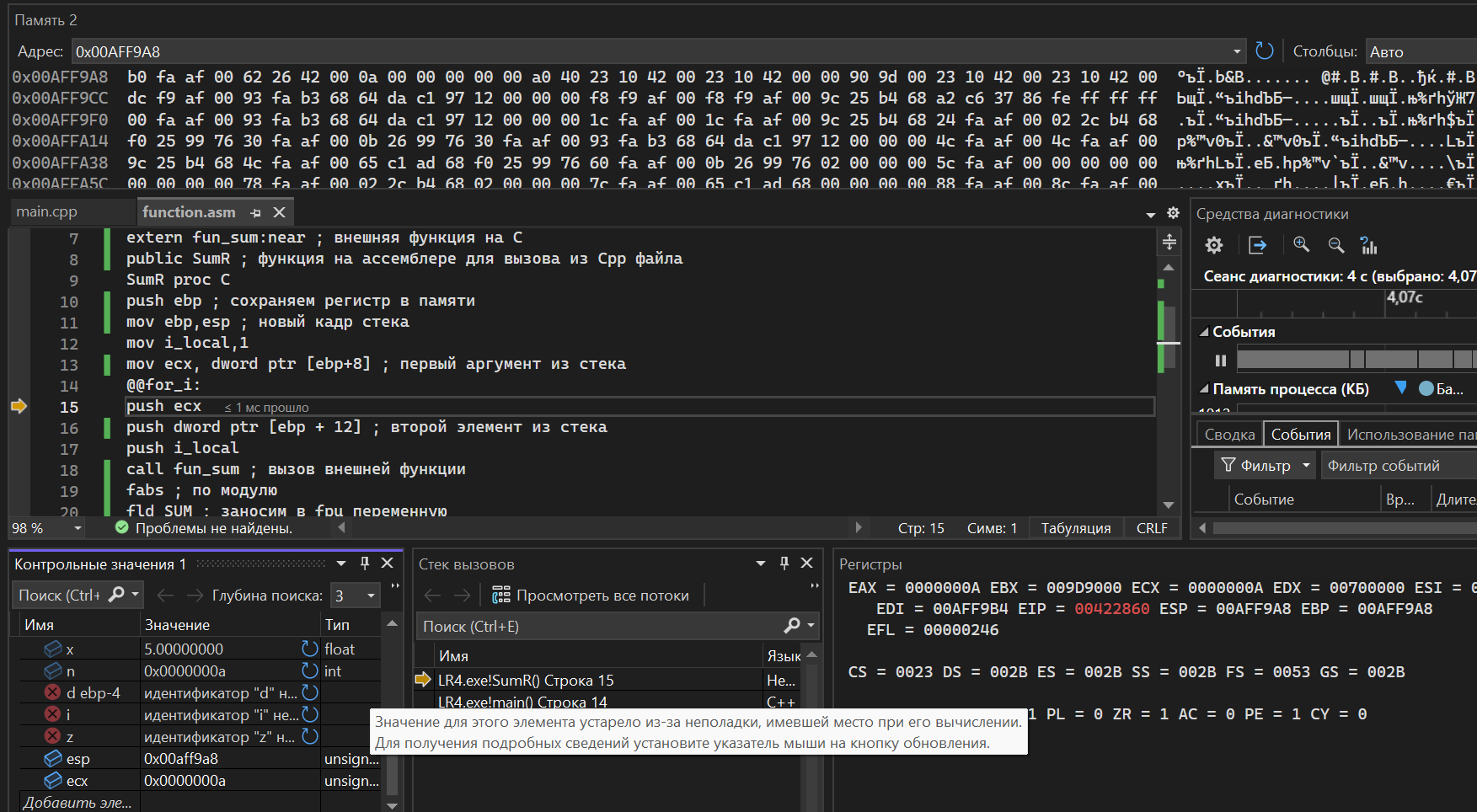


Рисунок 2 – Содержимое регистров после начала исполнения подпрограммы на ассемблере и копирования переменной n из стека в регистр счетчика

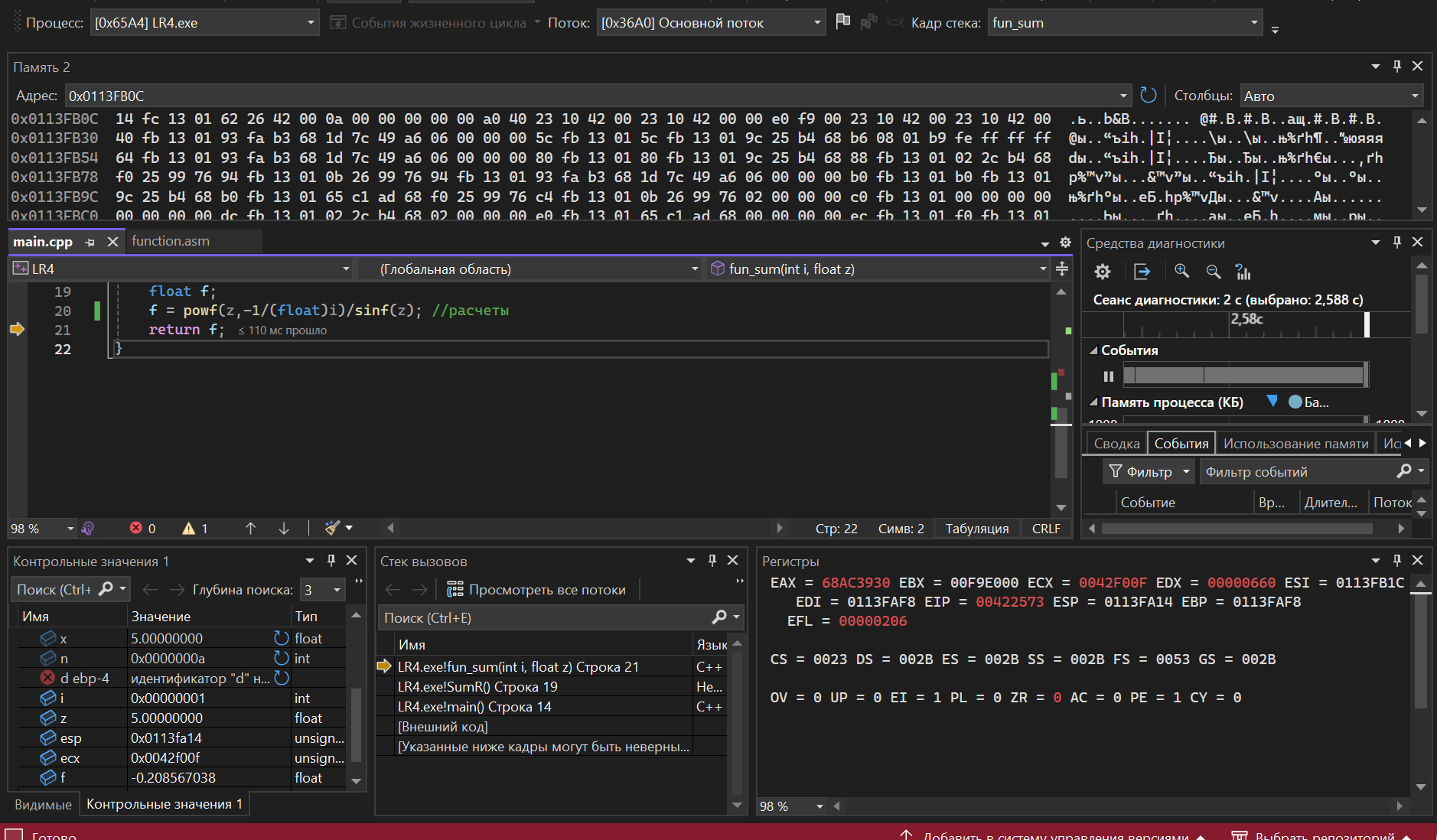


Рисунок 3 – Расчет первого значения суммы

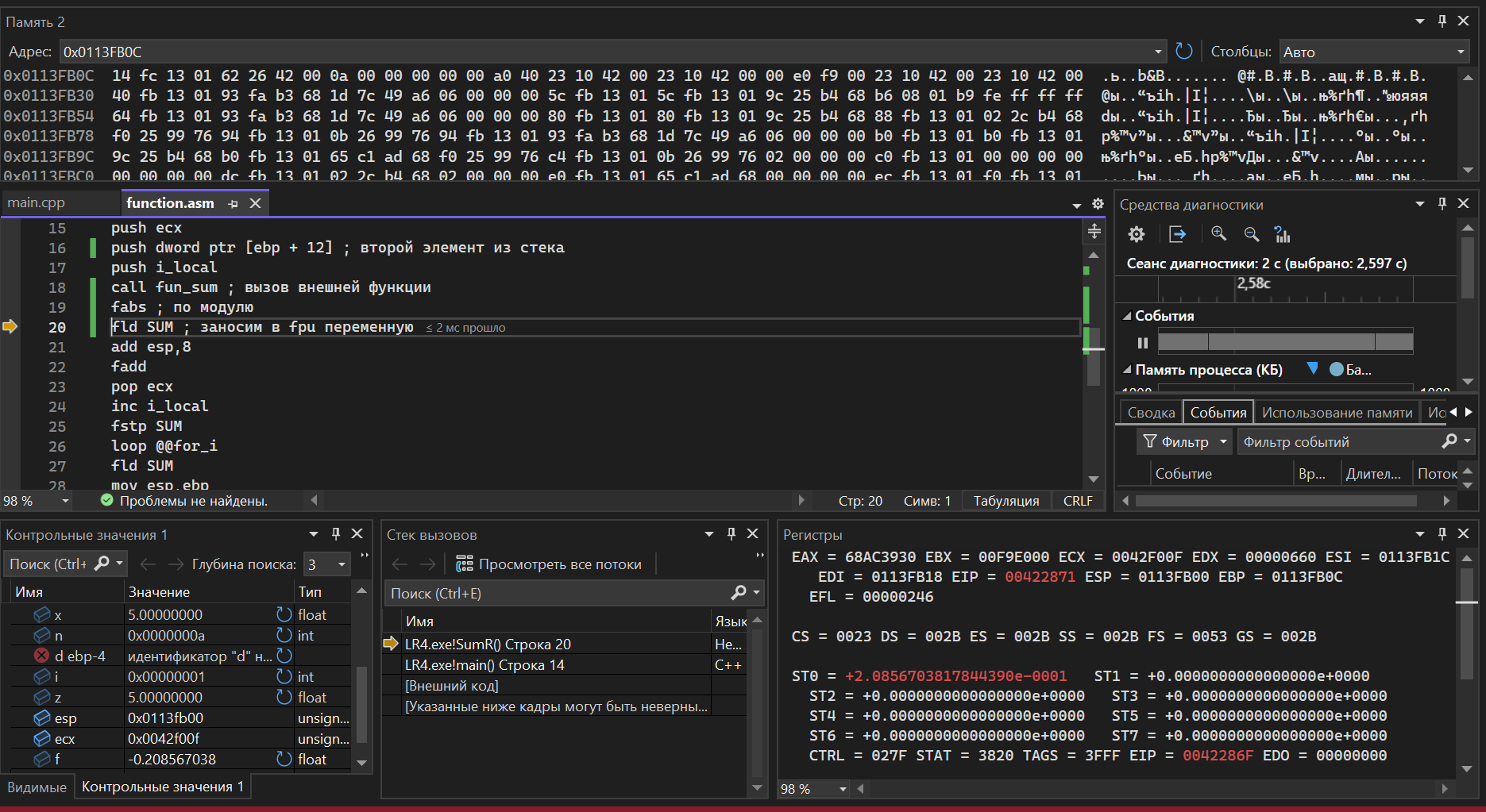


Рисунок 4 – модуль числа в fpu

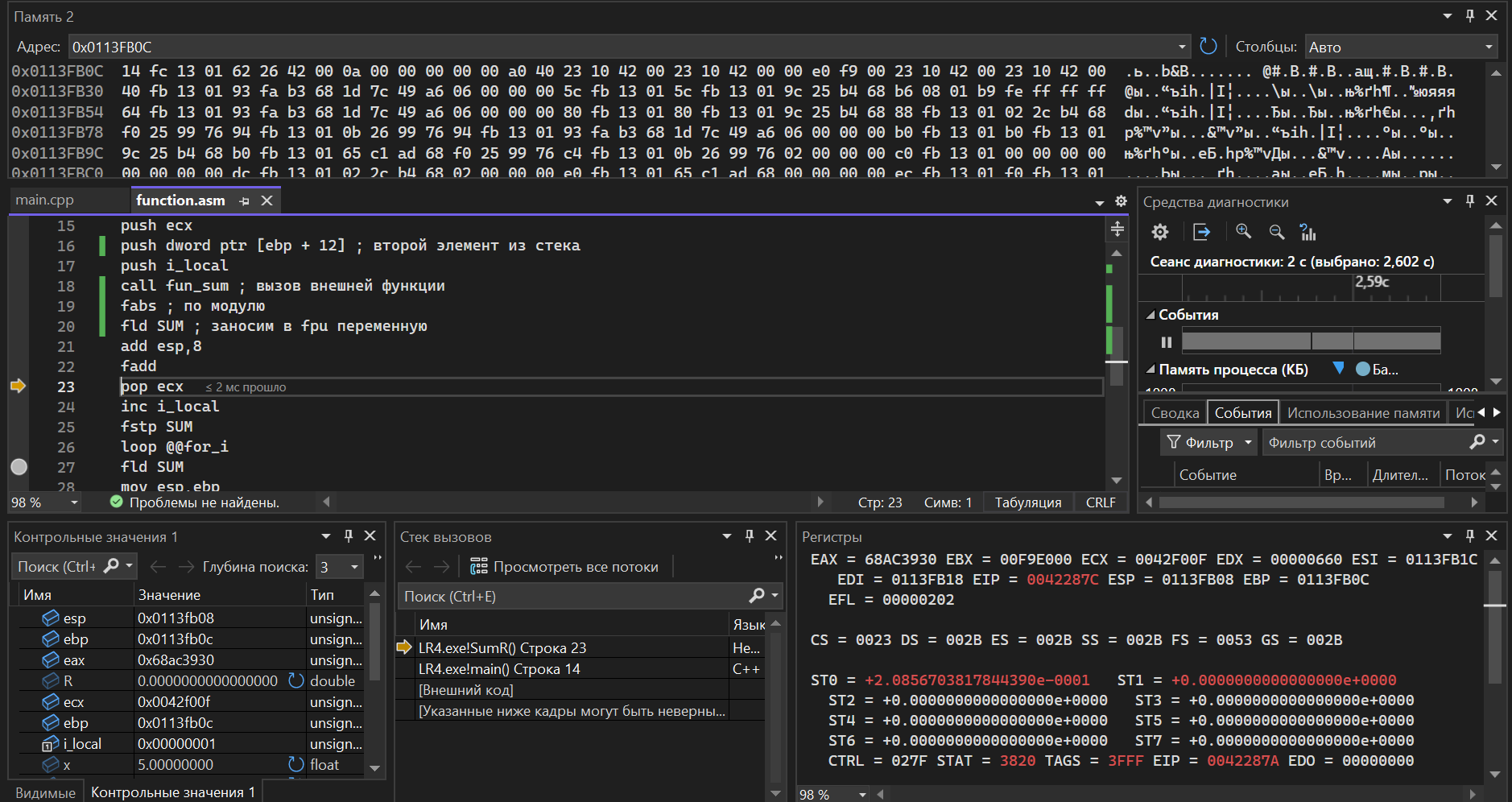


Рисунок 5 – Добавление числа к сумме

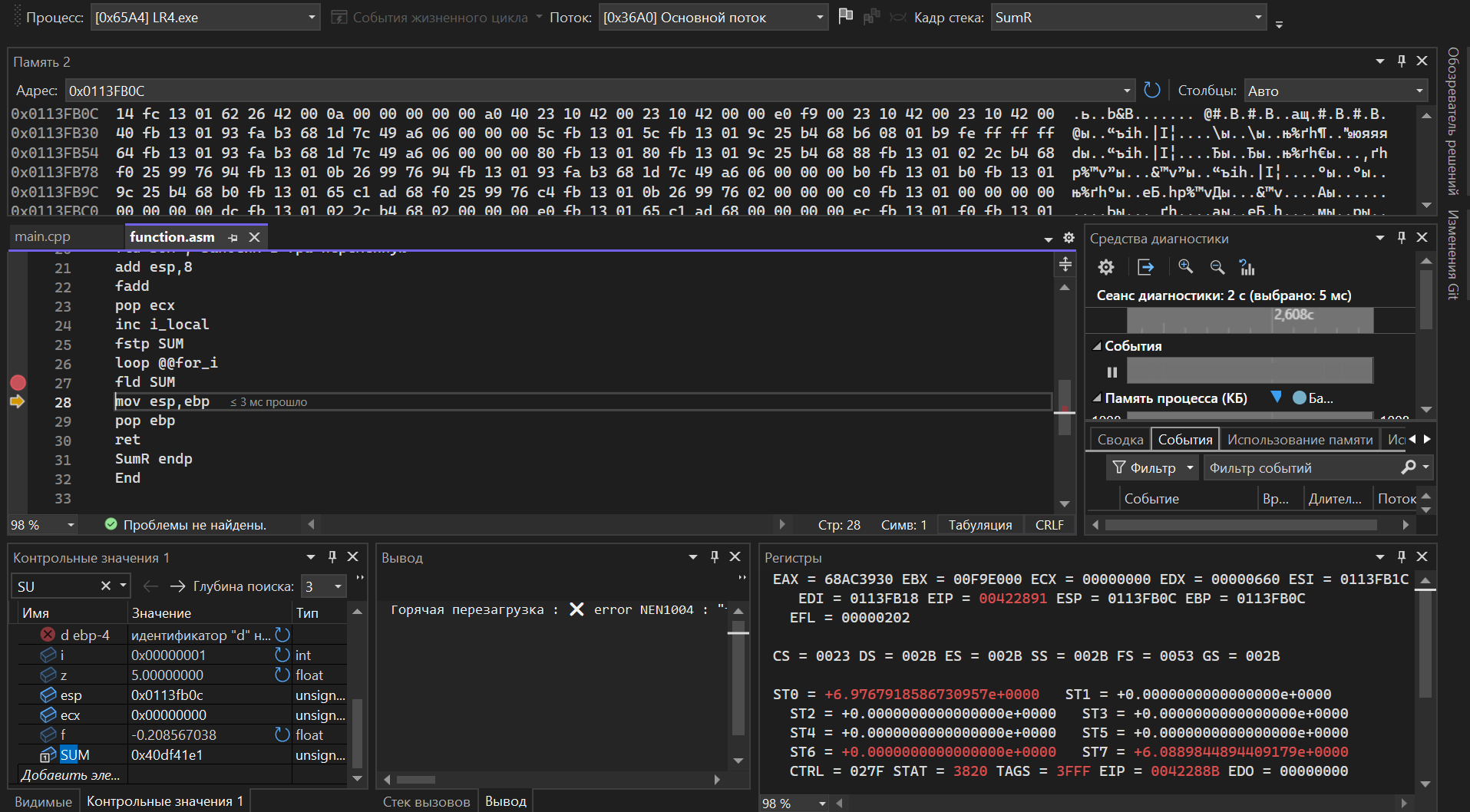


Рисунок 6 – Результат вычисления суммы после всего цикла

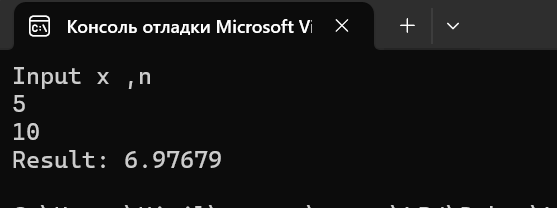


Рисунок 7 – Вывод результата для начального значения



Рисунок 8 – Результат вычислений для других данных

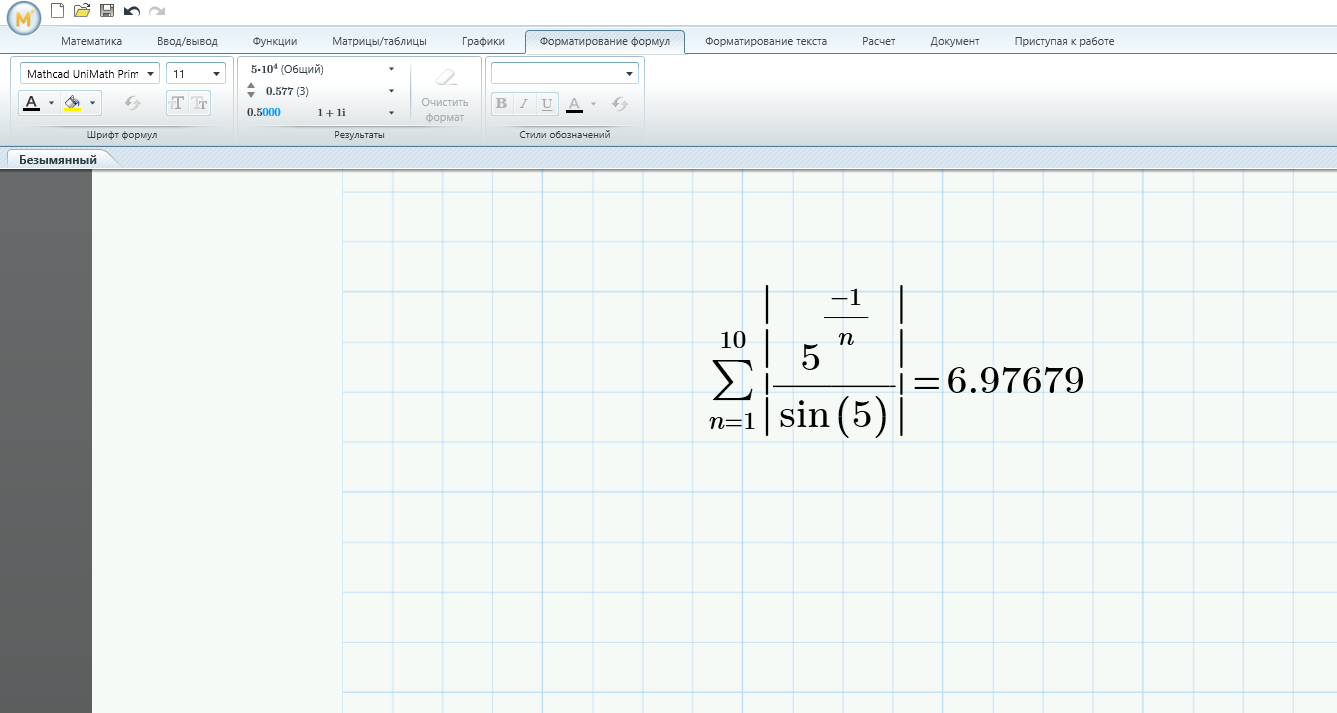


Рисунок 9 – Проверка в mathcad результата для начального значения

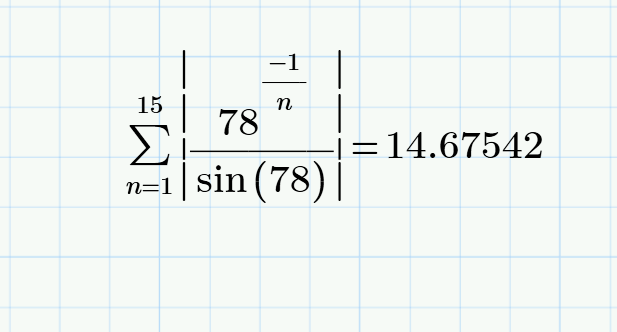


Рисунок 10 – Проверка в mathcad результата для других данных