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 №1

По дисциплине «Языки программирования»

ВАРИАНТ 10

Выполнил: ст. гр. ТКИ-142

Теканов Федор Андреевич

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

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

Москва 2024

**Оглавление**

[1 ЗАДАНИЕ 1.1 3](#_Toc185131652)

[1.1 Формулировка задания 3](#_Toc185131653)

[1.2 Блок-схема алгоритма 4](#_Toc185131654)

[1.3 Текст программы на языке С 5](#_Toc185131655)

[1.4 Результат выполнения программы 6](#_Toc185131656)

[1.5 Выполнение тестовых примеров 7](#_Toc185131657)

[1.6 Отметка о выполнение задания 8](#_Toc185131658)

[2 ЗАДАНИЕ 1.2 9](#_Toc185131659)

[2.1 Формулировка задания 9](#_Toc185131660)

[2.2 Блок-схема основного алгоритма 10](#_Toc185131661)

[2.3 Код программы на языке С 13](#_Toc185131662)

[2.4 Результат выполнения программы 15](#_Toc185131663)

[2.5 Выполнение тестовых примеров 16](#_Toc185131664)

[2.6 Отметка о выполнение задания 17](#_Toc185131665)

[3 ЗАДАНИЕ 1.3 18](#_Toc185131666)

[3.1 Формулировка задания 18](#_Toc185131667)

[3.2 Блок-схема основного алгоритма 19](#_Toc185131668)

[3.3 Текст программы на языке С 21](#_Toc185131669)

[3.4 Результат выполнения программы 22](#_Toc185131670)

[3.5 Выполнение тестовых примеров 23](#_Toc185131671)

1. ЗАДАНИЕ 1.1

1.1 Формулировка задания

Создать консольное приложение, вычисляющее значения переменных по представленным в таблице формулам (Таблица 1). Расчёт примера осуществить по заданным константам. Вывести на экран значения исходных данных, а также результат вычислений. Дополнить свой отчёт блок-схемой алгоритма.

Таблица 1 – Исходные данные

|  |  |  |
| --- | --- | --- |
| Вариант | Задачи | Интервал |
| 13 |  | x=1.4  y=3.1  z=0.5 |

1.2 Блок-схема алгоритма

Блок-схема основного алгоритма представлена ниже (Рисунок 1). Блок-схема функций расчет (Рисунок 2).

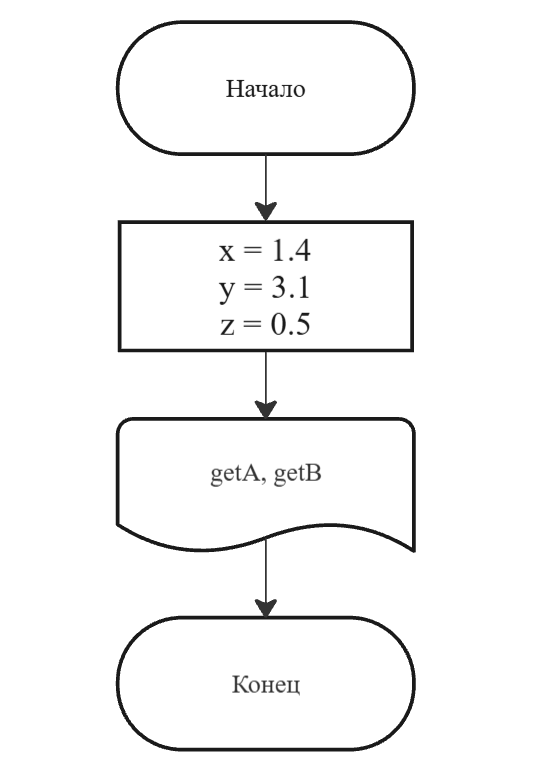


Рисунок 1 – Блок-схема основного алгоритма

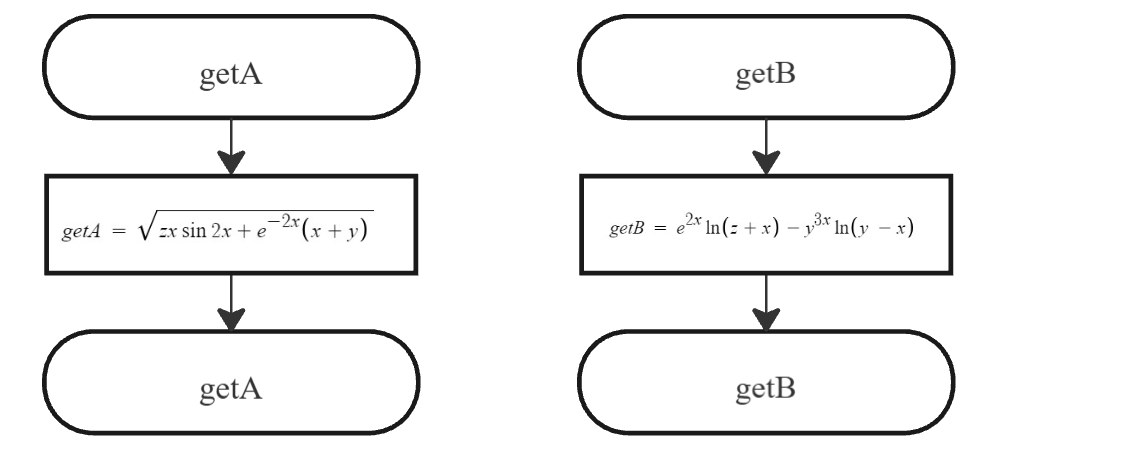


Рисунок 2 – Блок-схема функции getA, getB

1.3 Текст программы на языке С

#include <stdio.h>

#include <math.h>

/\*\* @brief рассчитывает выражение A по формуле

\* @param x значение константы x

\* @param y значение константы y

\* @param z значение константы z

\* @return рассчитанное значение

\*/

double getA(const double x, const double y, const double z);

/\*\* @brief рассчитывает выражение B по формуле

\* @param x значение константы x

\* @param y значение константы y

\* @param z значение константы z

\* @return рассчитанное значение.

\*/

double getB(const double x, const double y, const double z);

/\*\* @brief Точка входа в программу

\* @return 0 в случае успеха

\*/

int main(void)

{

const double x = 1.4;

const double y = 3.1;

const double z = 0.5;

printf("%lf\n%lf", getA(x, y, z), getB(x, y, z));

return 0;

}

double getA(const double x, const double y, const double z)

{

return sqrt(z \* x \* sin(2 \* x) + exp(-2 \* x) \* (x + y));

}

double getB(const double x, const double y, const double z)

{

return exp(2\*x) \* log(z + x) - pow(y, 3\*x) \* log(y - x);

}

1.4 Результат выполнения программы

Ниже представлен результат выполнения программы (Рисунок 3).



Рисунок 3 – Результат выполнения программы

1.5 Выполнение тестовых примеров

Ниже представлен результат выполнения тестового примера на языке Python (Рисунок 4).



Рисунок 4 – Результат выполнения программы на языке Python

1.6 Отметка о выполнение задания

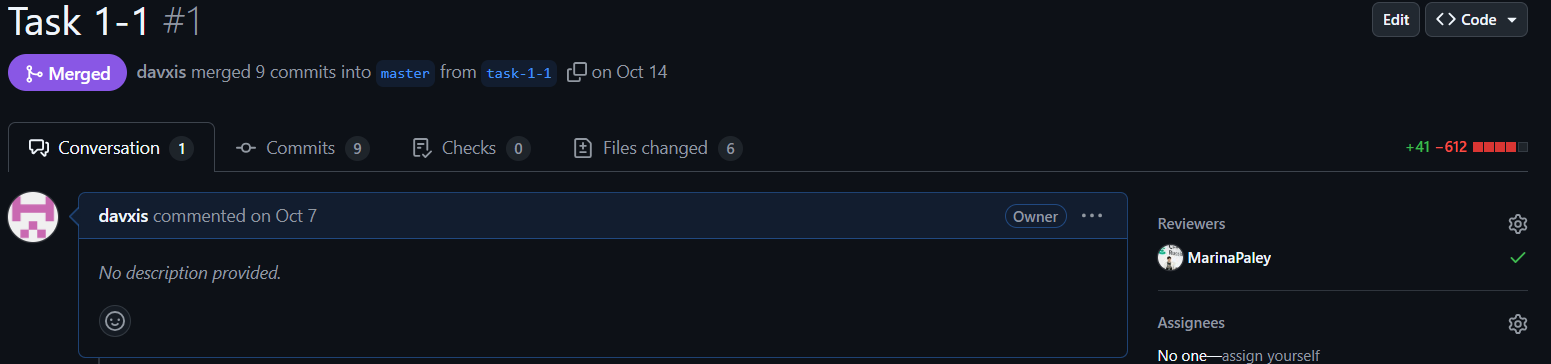
[](file:///C:\Users\MCRPO\Downloads\Telegram%20Desktop\Лаба%20№1.docx)

Рисунок 5

1. ЗАДАНИЕ 1.2
   1. Формулировка задания

Создать консольное приложение для решения задачи, представленной в таблице (Таблица 2). Данные для решения вводит пользователь. Вывести результат вычислений на экран. При необходимости дополнить свой отчёт поясняющими формулами, помогающими решить задачу. Дополнить свой отчёт блок-схемой алгоритма.

Таблица 2 – Исходные данные

|  |  |
| --- | --- |
| **Вариант** | **Задачи** |
| **10** | Вычислить площади геометрических фигур: прямоугольника и треугольника по заданным сторонам. |

* 1. Блок-схема основного алгоритма

Блок-схема основного алгоритма представлена ниже (Рисунок 5). Блок-схемы функций расчета (Рисунок 6, Рисунок 7, Рисунок 8).

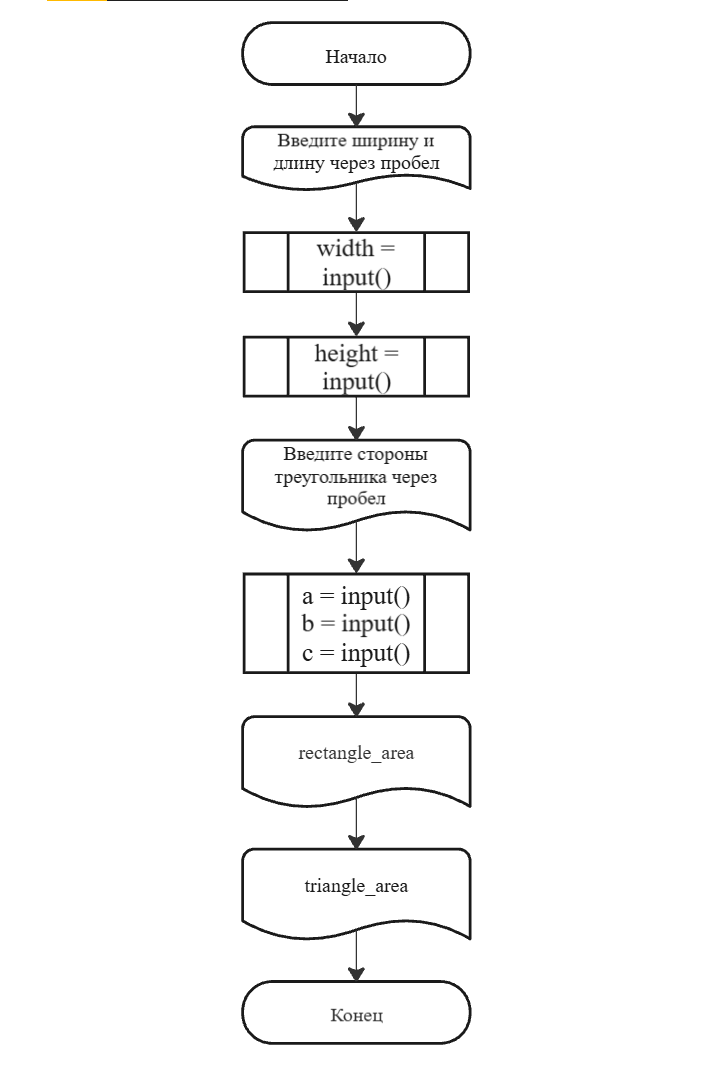


Рисунок 6 – Блок-схема основного алгоритма

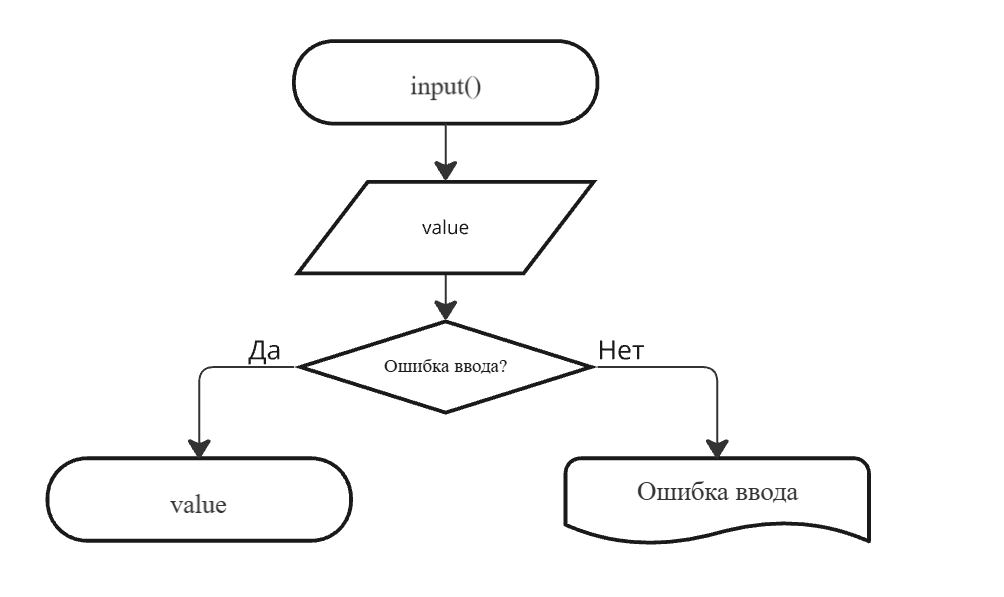


Рисунок 7 – Блок-схема функции input()

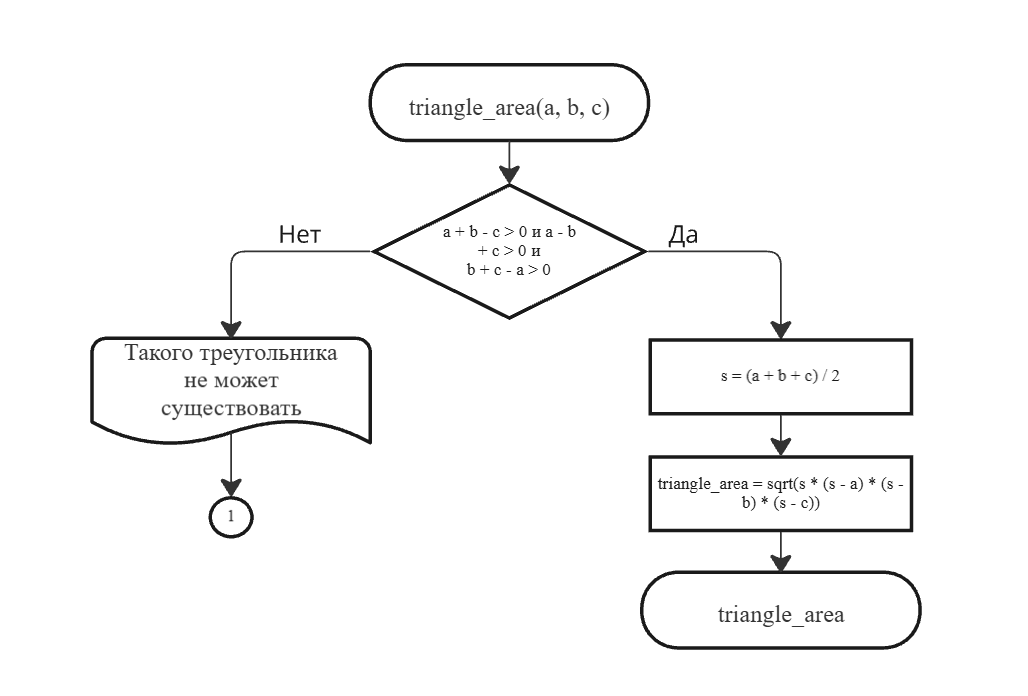


Рисунок 8 – Блок-схема функции triangle\_area

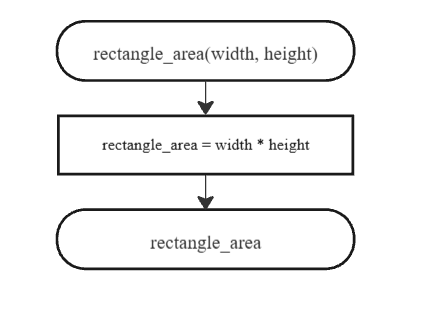


Рисунок 9 – Блок-схема функции rectangle\_area

* 1. Код программы на языке С

#include <stdio.h>

#include <errno.h>

#include <stdlib.h>

#include <math.h>

/\*\*

\* @brief Вычисляет площадь прямоугольника.

\*

\* @param width Ширина прямоугольника.

\* @param height Высота прямоугольника.

\* @return Площадь прямоугольника.

\*/

double rectangle\_area(const double width, const double height);

/\*\*

\* @brief Вычисляет площадь треугольника по формуле Герона.

\*

\* @param a Длина первой стороны треугольника.

\* @param b Длина второй стороны треугольника.

\* @param c Длина третьей стороны треугольника.

\* @return Площадь треугольника.

\*/

double triangle\_area(const double a, const double b, const double c);

/\*\*

\* @brief считывает вещественное число

\* @return возвращает вещественное число

\*/

double input(void);

/\*\*

\* @brief Точка входа в программу.

\* @return 0 в случае успеха.

\*/

int main(void)

{

printf("Введите ширину и длину через пробел\n");

const double width = input(), height = input();

printf("Введите стороны треугольника через пробел\n");

const double a = input(), b = input(), c = input();

// Вычисление и вывод площади прямоугольника

printf("Площадь прямоугольника: %.2lf\n", rectangle\_area(width, height));

// Вычисление и вывод площади треугольника

printf("Площадь треугольника: %.2lf\n", triangle\_area(a, b, c));

return 0;

}

double input(void)

{

double value = 0.0;

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

if (result != 1)

{

errno = EIO;

perror("Ошибка ввода");

exit(EXIT\_FAILURE);

}

return value;

}

double triangle\_area(const double a, const double b, const double c)

{

if (a + b - c > 0 && a - b + c > 0 && b + c - a > 0)

{

const double s = (a + b + c) / 2; // Полупериметр

return sqrt(s \* (s - a) \* (s - b) \* (s - c)); // Формула Герона

}

else

{

printf("Такого треугольника не может существовать\n");

errno = EIO;

exit(EXIT\_FAILURE);

}

}

double rectangle\_area(const double width, const double height)

{

return width \* height;

}

* 1. Результат выполнения программы

Ниже представлены результаты выполнения программы (Рисунок 9, Рисунок 10, Рисунок 11).

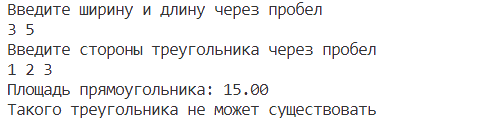


Рисунок 10 – Первый результат выполнения программы

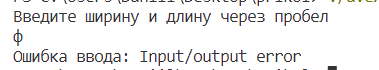


Рисунок 11 – Второй результат выполнения программы

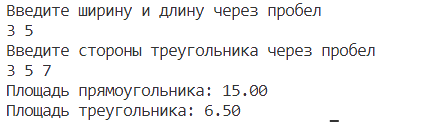


Рисунок 12 – Третий результат выполнения программы

* 1. Выполнение тестовых примеров

Ниже представлено выполнение тестового примера на языке Python (Рисунок 12).

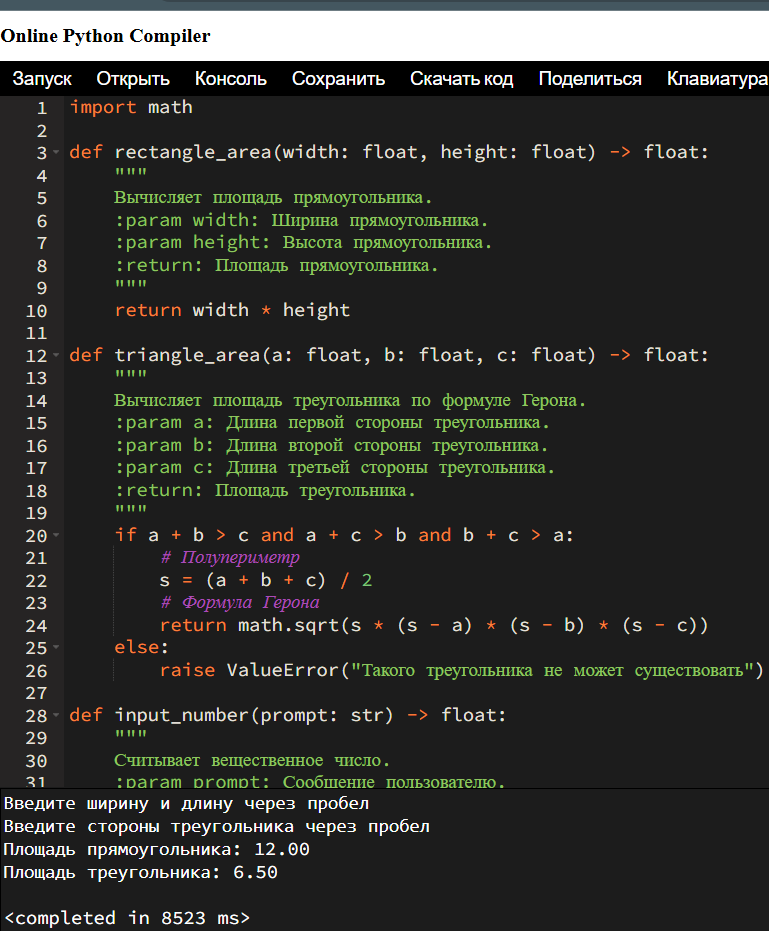


Рисунок 13 – Результат выполнения тестового примера

* 1. Отметка о выполнение задания

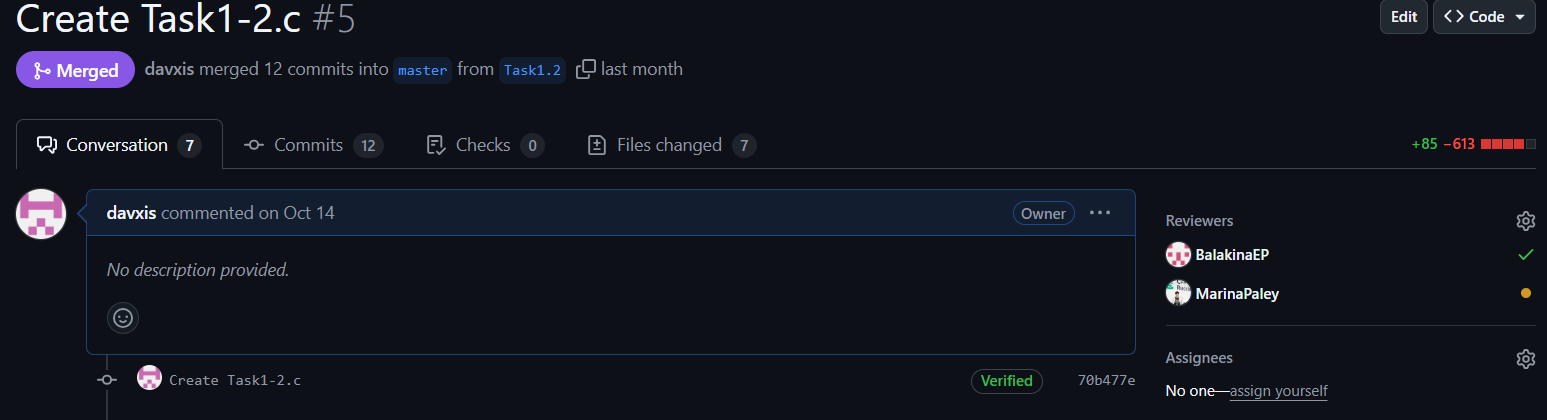


Рисунок 14

1. ЗАДАНИЕ 1.3
   1. Формулировка задания

Создать консольное приложение для решения задачи, представленной в таблице (Таблица 3). Данные для решения вводит пользователь. Вывести результат вычислений на экран. При необходимости дополнить свой отчёт поясняющими формулами, помогающими решить задачу. Дополнить свой отчёт блок-схемой алгоритма.

Таблица 3 – Исходные данные

|  |  |
| --- | --- |
| **Вариант** | **Задача** |
| **10** | Сколько времени нагревалась проволока с сопротивлением *R* Ом, если при силе тока в *I* А в ней выделилось *Q* кДж теплоты? |

* 1. Блок-схема основного алгоритма

Ниже представлена блок-схема основного алгоритма (Рисунок 13). Блок-схемы функций расчета (Рисунок 14, Рисунок 15).

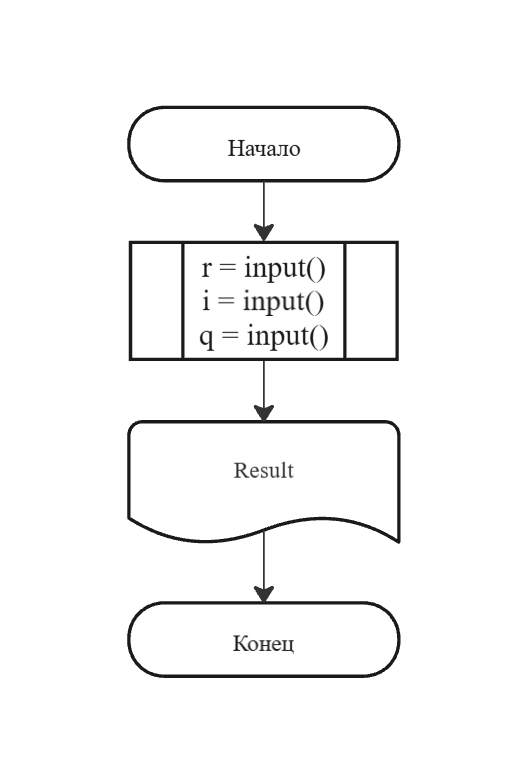


Рисунок 15 – блок-схема основного алгоритма

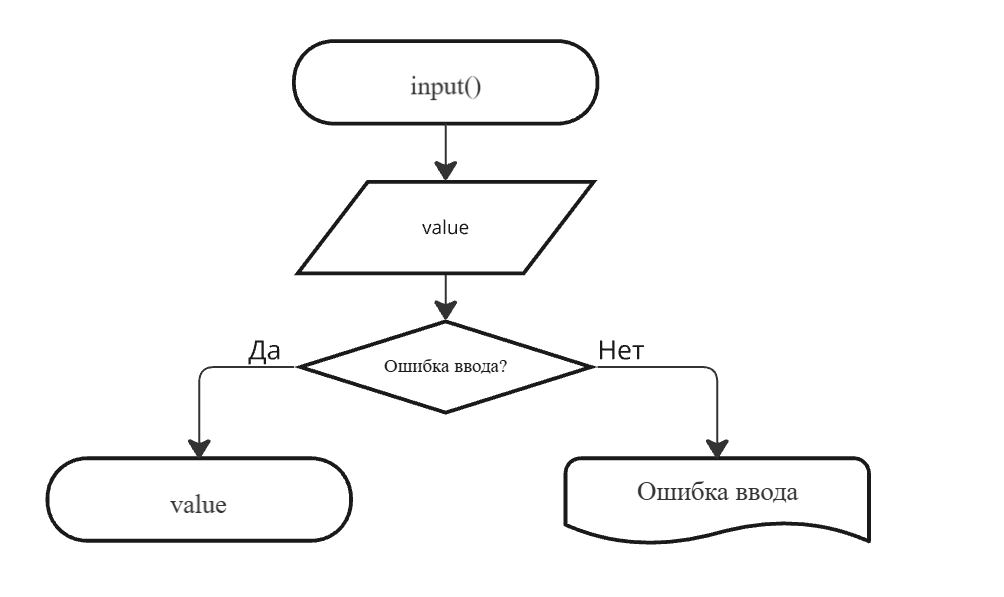


Рисунок 16 – блок-схема функции input()

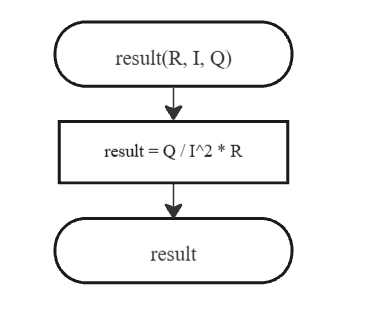


Рисунок 17 – блок-схема функции result

* 1. Текст программы на языке С

#include <stdio.h>

#include <math.h>

#include <stdlib.h>

#include <errno.h>

/\*\*

\* @brief считывает вещественное число

\* @return возвращает вещественное число

\*/

double input(void);

/\*\*

\* @brief рассчитывает работу эл.тока по формуле

\* @param I - сила тока

\* @param U - напряжение

\* @param time - время переведенное в секунды

\* return рассчитанное значение работы силы тока

\*/

double result(const double R, const double I, const double Q);

/\*\*

\* @brief точка входу в прогамму

\* return 0 в случае успехa

\*/

int main(void)

{

const double R = input();

const double I = input();

const double Q = input();

printf("Проволка нагревалась %lf секунд\n", result(R, I, Q));

return 0;

}

double input(void)

{

double value = 0.0;

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

if (result != 1)

{

errno = EIO;

perror("Ошибка ввода");

exit(EXIT\_FAILURE);

}

return value;

}

double result(const double R, const double I, const double Q)

{

return Q / pow(I, 2) \* R;

}

* 1. Результат выполнения программы

Ниже представлены результаты выполнения программы (Рисунок 16, Рисунок 17, Рисунок 18).



Рисунок 18 – первый результат выполнения программы

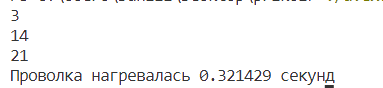


Рисунок 19 – второй результат выполнение программы



Рисунок 20 – третий результат выполнения программы

* 1. Выполнение тестовых примеров

Ниже представлены результаты выполнения тестовых примеров на языке Python (Рисунок 19, Рисунок 20).

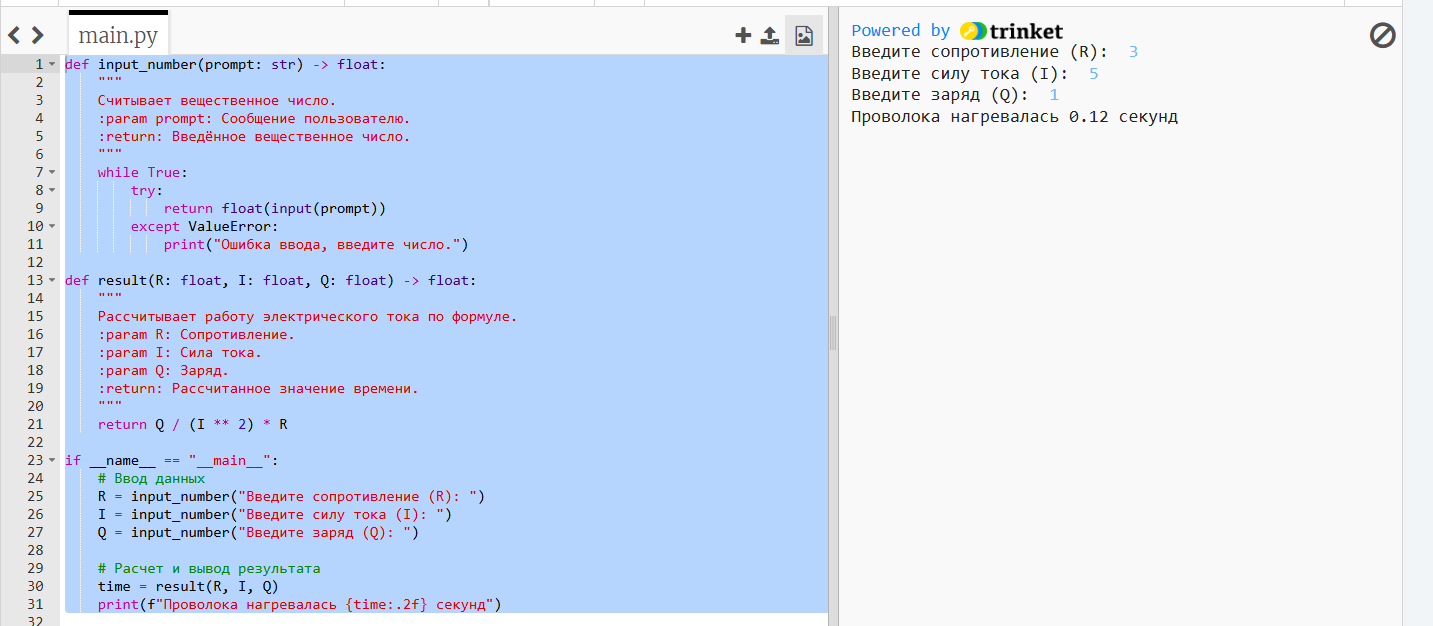


Рисунок 21 – результат выполнения первого тестового примера

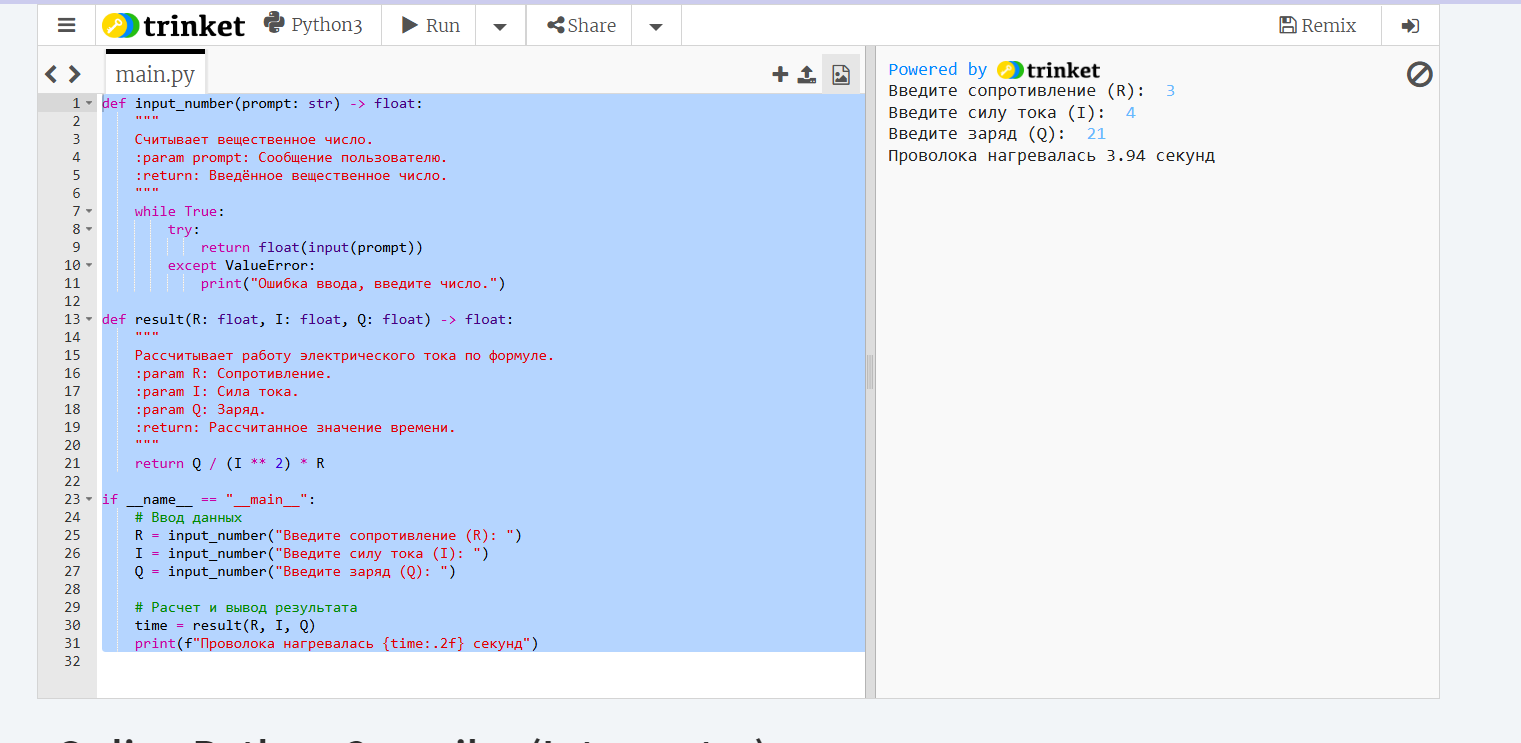


Рисунок 22 – результат выполнения второго тестового примера

* 1. Отметка о выполнении задания

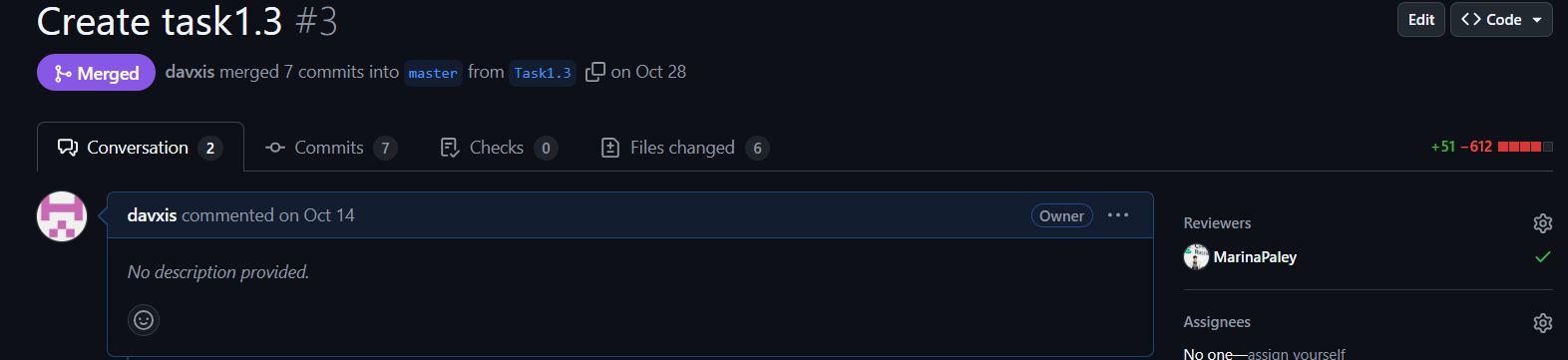
****

Рисунок 23