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 9

Выполнил: ст. гр. ТКИ-141

Ткачев Вадим Евгеньевич

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

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

Москва 2024

Оглавление

[1 Решение задачи 1–1 3](#_Toc182431052)

[1.2 Блок-схема алгоритма задание 1–1 4](#_Toc182431053)

[1.3 Текст программы на языке С задание 1–1 5](#_Toc182431054)

[1.4 Результаты выполнения программы 1–1 6](#_Toc182431055)

[1.5 Выполнение тестовых примеров задание 1–1 6](#_Toc182431056)

[1.6 Отметка о выполнении задания в веб-хостинге системы контроля версий задание 1–1 7](#_Toc182431057)

[2 Решение задачи 1–2 8](#_Toc182431058)

[2.1 Формулировка задачи 1–2 8](#_Toc182431059)

[2.2 Блок-схема алгоритма задание 1–2 8](#_Toc182431060)

[2.3 Текст программы на языке С задание 1–2 9](#_Toc182431061)

[2.4 Результаты выполнения программы 1–2 11](#_Toc182431062)

[2.5 Выполнение тестовых примеров задание 1–2 11](#_Toc182431063)

[2.6 Отметка о выполнении задания в веб-хостинге системы контроля версий задание 1–2 11](#_Toc182431064)

[3 Решение задачи 1–3 13](#_Toc182431065)

[3.1 Формулировка задачи 1–3 13](#_Toc182431066)

[3.2 Блок-схема алгоритма задание 1–3 14](#_Toc182431067)

[3.3 Текст программы на языке С задание 1–3 15](#_Toc182431068)

[3.4 Результаты выполнения программы 1–3 16](#_Toc182431069)

[3.5 Выполнение тестовых примеров задание 1–3 17](#_Toc182431070)

[3.6 Отметка о выполнении задания в веб-хостинге системы контроля версий задание 1–3 17](#_Toc182431071)

1. Решение задачи 1–1
   1. **Формулировка задачи 1–1**

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 | Формулы | Константы |
| 9 |  | x=2  y=0.7  z=-1 |

* 1. Блок-схема алгоритма задание 1–1

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

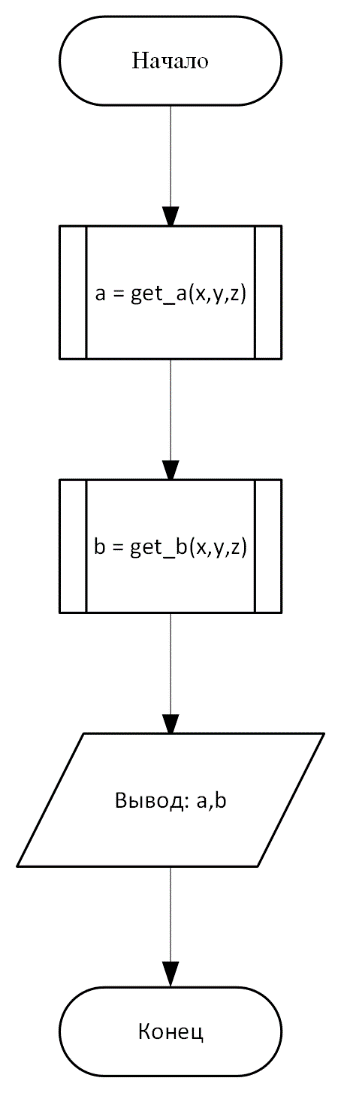


Рисунок 1 ­­–­ Блок-схема основного алгоритма

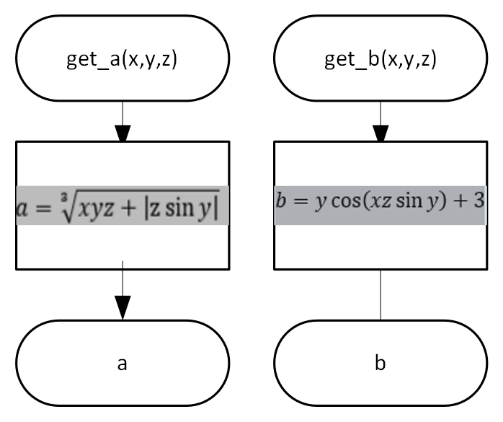


Рисунок 2 – Блок-схема для вычисления a и b

* 1. Текст программы на языке С задание 1–1

#include <stdio.h>

#include <math.h>

/\*\*

\* @brief функция для вычисления значения a.

\* @param x Значение переменной x.

\* @param y Значение переменной y.

\* @param z Значение переменной z.

\* @return Результат вычисления a.

\*/

double get\_a(double x, double y, double z);

/\*\*

\* @brief функция для вычисления значения b.

\* @param x Значение переменной x.

\* @param y Значение переменной y.

\* @param z Значение переменной z.

\* @return Результат вычисления b.

\*/

double get\_b(double x, double y, double z);

/\*\*

\* @brief Точка входа в программу.

\* @return Возвращает в случае успеха.

\*/

int main() {

const double x = 2.0;

const double y = 0.7;

const double z = -1.0;

double a = get\_a(x, y, z);

double b = get\_b(x, y, z);

printf("a = %f \nb = %f", a, b);

return 0;

}

double get\_a(double x, double y, double z) {

return cbrt(x \* y \* z + fabs(z \* sin(y)));

}

double get\_b(double x, double y, double z) {

return y \* cos(x \* z \* sin(y)) + 3;

}

* 1. Результаты выполнения программы 1–1

Результат выполнения программы в C представлен ниже (Рисунок 3).

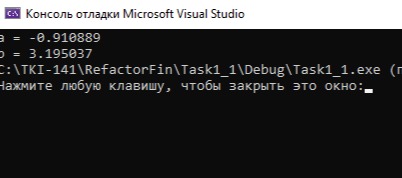


Рисунок 3 – Результат выполнения программы

* 1. Выполнение тестовых примеров задание 1–1

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

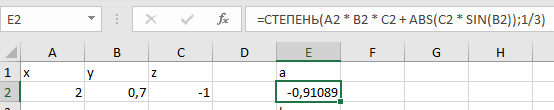


Рисунок 4 – Результат расчета переменной a

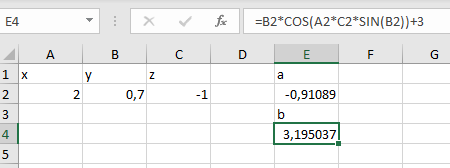


Рисунок 5 – Результат расчета переменной b

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

Ниже представлено доказательство того, что задание 1–1 было принято. (Рисунок 6)

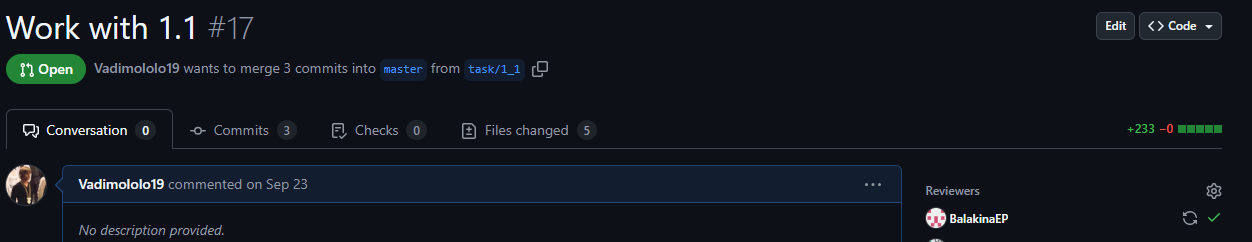


Рисунок 6 – Approve задачи 1–1

1. Решение задачи 1–2
   1. Формулировка задачи 1–2

Создать консольное приложение для решения задачи, представленной в таблице. Данные для решения вводит пользователь. Помните, что ввод необходимо проверять на правильность (только числа). Вывести результат вычислений на экран. При необходимости дополнить свой отчёт поясняющими формулами, помогающими решить задачу. Дополнить свой отчёт блок-схемой алгоритма.

Таблица 1 – Исходные данные

|  |  |
| --- | --- |
| Вариант | Задачи |
| 9 | Известен объем информации в байтах. Выразить его в мегабайтах и гигабайтах. |

* 1. Блок-схема алгоритма задание 1–2

Блок-схема основного алгоритма представлена ниже (Рисунок 7). Блок-схемы функций get\_mb и get\_gb, а также проверка представлены ниже (Рисунок 8 и Рисунок 9).

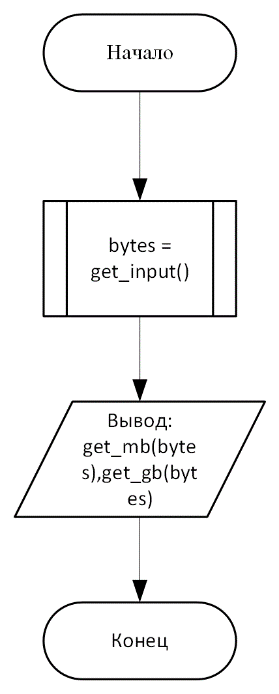


Рисунок 7 –­ Блок-схема основного алгоритма

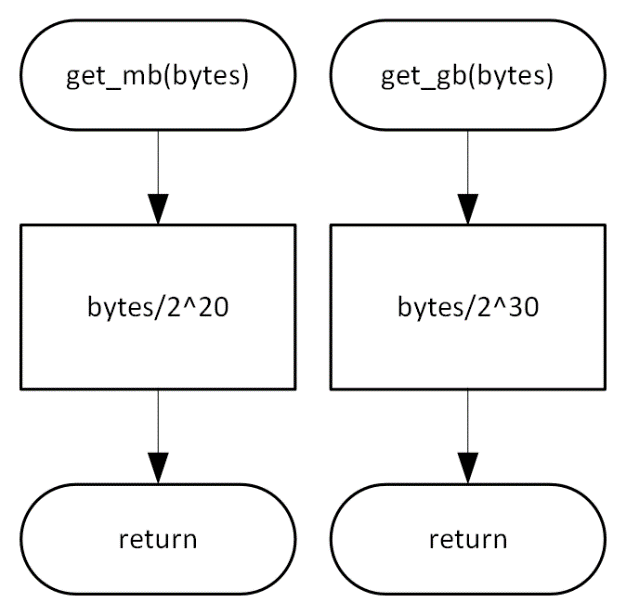


Рисунок 8 – Блок-схема функций get\_mb и get\_gb

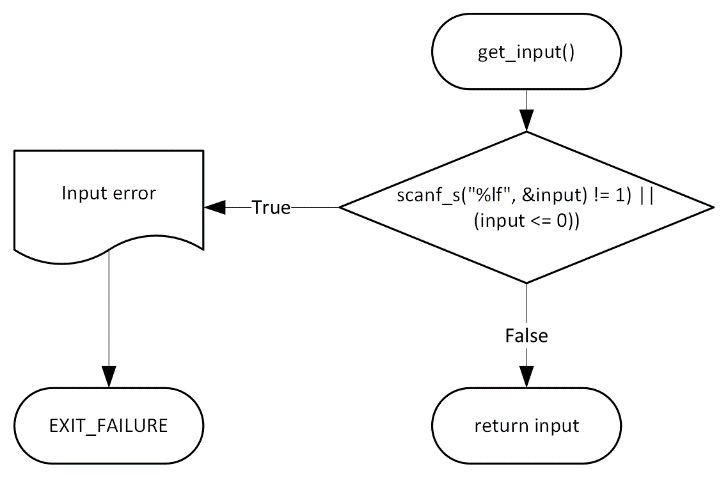


Рисунок 9 – Блок-схема проверки ввода

* 1. Текст программы на языке С задание 1–2

#include <stdio.h>

#include <math.h>

#include <stdlib.h>

/\*\*

\* @brief Функция проверки введенных данных.

\* @return Возвращает значение, если правильное, закрывает программу, если иначе.

\*/

double get\_input();

/\*\*

\* @brief Функция перевода байтов в мегабайты.

\* @params bytes - байты, которые ввел пользователь.

\* @return Возвращает значение в Мегабайтах.

\*/

double get\_mb(double bytes);

/\*\*

\* @brief Функция перевода байтов в гигабайты.

\* @params bytes - байты, которые ввел пользователь.

\* @return Возвращает значение в Гигабайтах.

\*/

double get\_gb(double bytes);

int main()

{

puts("Введите значение в байтах");

double bytes = get\_input(), mb = get\_mb(bytes), gb = get\_gb(bytes);

printf("mb = %lf\ngb = %lf", mb, gb);

return 0;

}

double get\_input()

{

double input;

if ((scanf\_s("%lf", &input) != 1) || (input <= 0)) {

printf("Input error");

exit(EXIT\_FAILURE);

}

return input;

}

double get\_mb(double bytes)

{

const double mb\_value = powl(2,20);

return bytes / mb\_value;

}

double get\_gb(double bytes)

{

const double gb\_value = powl(2,30);

return bytes / gb\_value;

}

* 1. Результаты выполнения программы 1–2

Результаты выполнения программы в C представлена ниже (Рисунок 10).

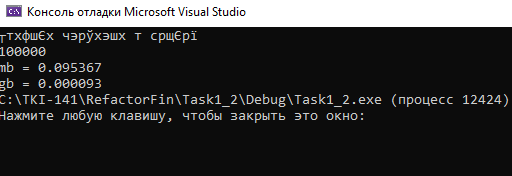


Рисунок 10 – Результаты выполнения программы

* 1. Выполнение тестовых примеров задание 1–2

В программе Pythonвыполнены тестовые примеры. Результаты их выполнения представлены ниже (Рисунок 11).

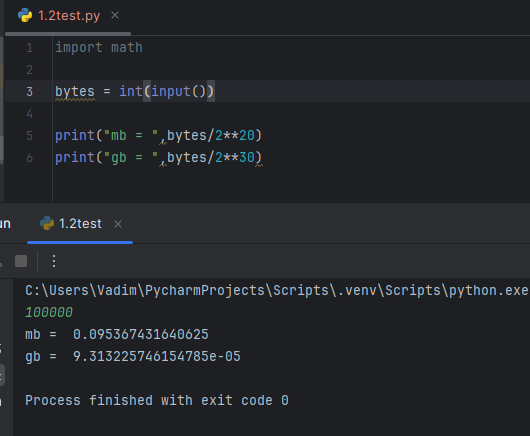


Рисунок 11 – Результат выполнения тестовых примеров в Python

* 1. Отметка о выполнении задания в веб-хостинге системы контроля версий задание 1–2

Ниже представлено доказательство того, что задание 1–2 было принято. (Рисунок 12

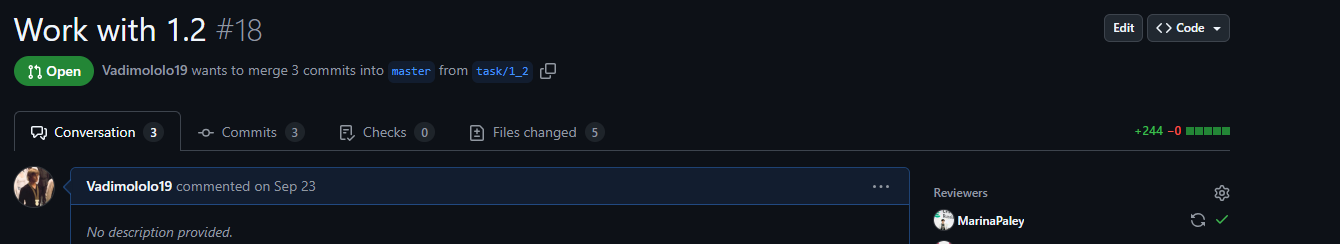


Рисунок 12 – Approve задачи 1–2

1. Решение задачи 1–3
   1. Формулировка задачи 1–3

Создать консольное приложение для решения задачи, представленной в таблице. Данные для решения вводит пользователь. Помните, что ввод необходимо проверять на правильность (только числа). Вывести результат вычислений на экран. При необходимости дополнить свой отчёт поясняющими формулами, помогающими решить задачу. Дополнить свой отчёт блок-схемой алгоритма.

Таблица 1 – Исходные данные

|  |  |
| --- | --- |
| Вариант | Задачи |
| 9 | Три резистора (*R1 R2 R3*) соединены последовательно. Найдите сопротивление соединения. |

* 1. Блок-схема алгоритма задание 1–3

Блок-схема основного алгоритма представлена ниже (Рисунок 13). Блок-схемы функции get\_r (Рисунок 14) и проверки представлены ниже (Рисунок 15).

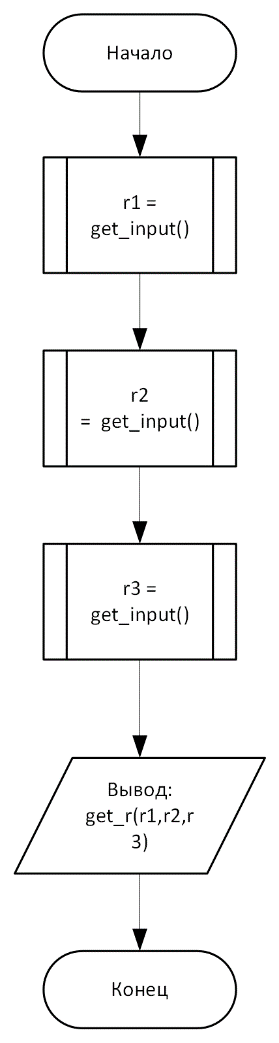


Рисунок 13 ­ Блок-схема основного алгоритма

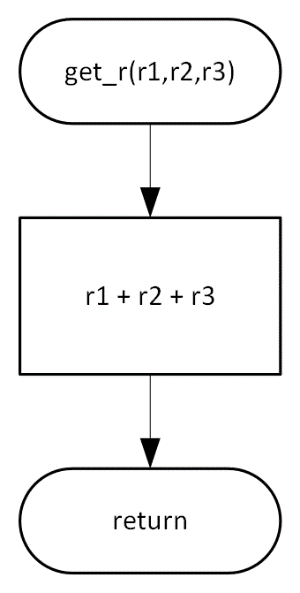


Рисунок 14 – Блок-схема функции get\_r

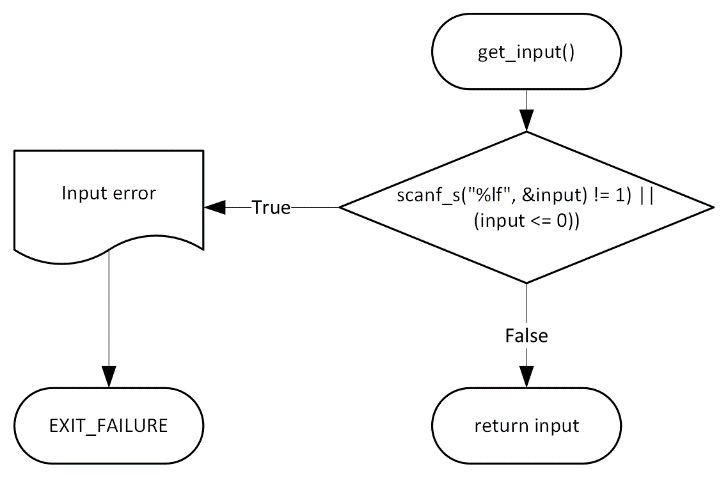


Рисунок 15 – Блок-схема проверки ввода

* 1. Текст программы на языке С задание 1–3

#include <stdio.h>

#include <math.h>

#include <stdlib.h>

/\*\*

\* @brief Функция для расчета сопротивления соединения.

\* @param R1 - значение первого резистора.

\* @param R2 - значение второго резистора.

\* @param R3 - значение третьего резистора.

\* return Рассчет сопротивления всего соединения

\*/

double get\_r(double R1, double R2, double R3);

/\*\*

\* @brief Функция проверки введенных значений.

\* @return возвращает 0 и значения, если проверки выполнены, или ошибку, если иначе

\*/

double get\_input();

/\*\*

\* @brief Точка входа

\* @return Возвращает результат с кодом ошибки 0

\*/

int main()

{

puts("Enter the value of the three resistors");

double R1 = get\_input(), R2 = get\_input(), R3 = get\_input();

double r = get\_r(R1, R2, R3);

printf("Final R = %lf", r);

return 0;

}

double get\_input()

{

double input;

if (scanf\_s("%lf", &input) != 1 || (input <= 0))

{

printf("Input error");

exit(EXIT\_FAILURE);

}

return input;

}

double get\_r(double R1, double R2, double R3)

{

return R1 + R2 + R3;

}

* 1. Результаты выполнения программы 1–3

Результаты выполнения программы в C представлена ниже (Рисунок 16).

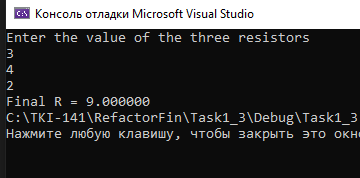


Рисунок 16 – Результат выполнения программы

* 1. Выполнение тестовых примеров задание 1–3

В программе Pythonвыполнено решение тестового примера. Результат его выполнения представлен ниже (Рисунок 17).

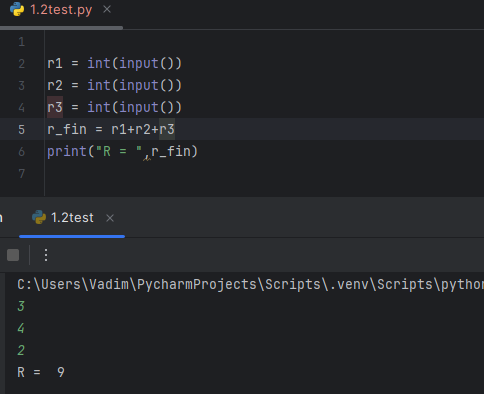


Рисунок 17 – Результат выполнения тестового примера

* 1. Отметка о выполнении задания в веб-хостинге системы контроля версий задание 1–3

Ниже представлено доказательство того, что задание 1–3 было принято. (Рисунок 18)

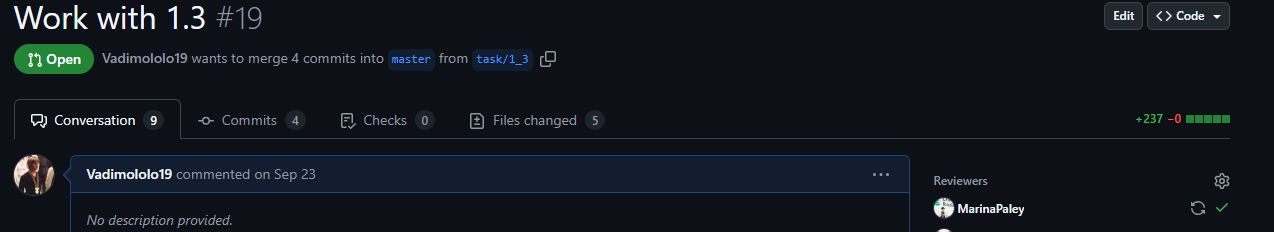


Рисунок 18 – Approve задачи 1–3