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 5

Выполнил: ст. гр. ТКИ - 142

Горковец Анна Сергеевна

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

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

Москва 2023

**Решение задачи 1.1**

**Формулировка задачи**

Создать консольное приложение, вычисляющее значения переменных по представленным в таблице формулам. Расчёт примера осуществить по заданным константам. Вывести на экран значения исходных данных, а также результат вычислений

Таблица 1 – Условие задачи 1.1

|  |  |  |
| --- | --- | --- |
| **Вариант** | **Формулы** | **Константы** |
| **5** |  | **x=-2.9**  **y=15.5**  **z=1.5** |

**Блок-схема алгоритма**

Блок-схема алгоритма представлена ниже (Рисунок 1, Рисунок 2, Рисунок 3)

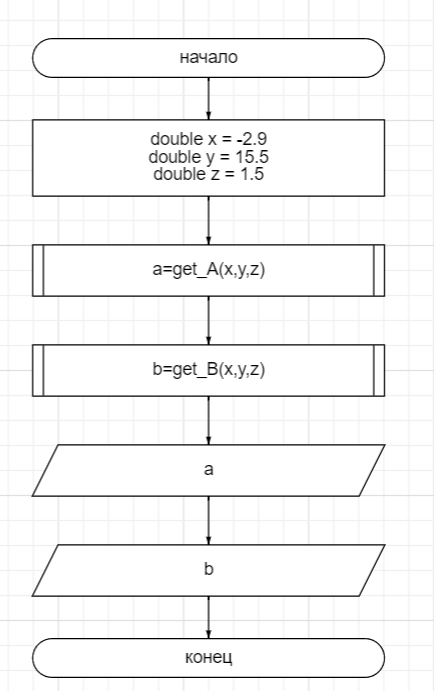


Рисунок 1 – блок-схема функции main()

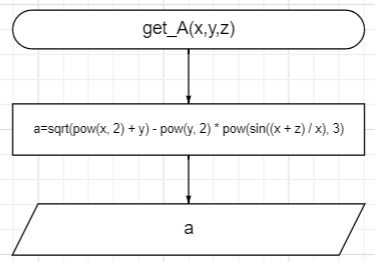


Рисунок 2 – блок-схема функции get\_A()

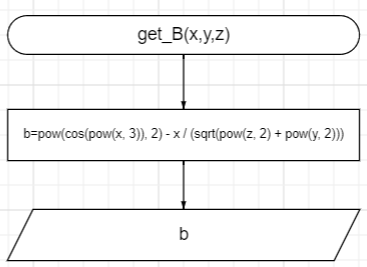


Рисунок 3 – блок-схема функции get\_B()

**Код для задания 1.1**

﻿#include <stdio.h>

#include <math.h>

/\*\*

\* @breaf Рассчитывает значение переменной а

\* @param x значение константы x

\* @param y значение константы y

\* @param z значение константы z

\* @return возвращает значение переменной а

\*/

double get\_A(const double x, const double y, const double z);

/\*\*

\*@breaf Рассчитывает значение переменной b

\* @param x значение константы x

\* @param y значение константы y

\* @param z значение константы z

\* @return возвращает значение переменной а

\*/

double get\_B(const double x, const double y, const double z);

int main()

{

double a, b;

const double x = -2.9;

const double y = 15.5;

const double z = 1.5;

a = get\_A(x, y, z);

b = get\_B(x, y, z);

printf\_s("a=%lf\n", a);

printf\_s("b=%lf\n", b);

return 0;

}

double get\_A(const double x, const double y, const double z)

{

return sqrt(pow(x, 2) + y) - pow(y, 2) \* pow(sin((x + z) / x), 3);

}

double get\_B(const double x, const double y, const double z)

{

return pow(cos(pow(x, 3)), 2) - x / (sqrt(pow(z, 2) + pow(y, 2)));

}

**Решение различных тестовых примеров на С**

Ниже представлен вывод программы 1.1 в С (Рисунок 4)



Рисунок 4 - Вывод программы 1.1

**Решение различных тестовых** примеров в Excel

Ниже представлено решение задания 1.1 в Excel (Рисунок 5)

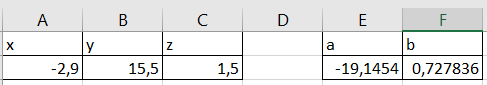


Рисунок 5 - Решение задания 1.1 в Excel

**Подтверждение approve 1.1**

Ниже представлено доказательство того, что задание 1.1 было принято. (Рисунок 6)

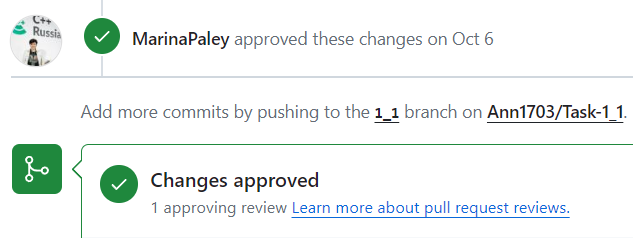


Рисунок 6 – Approve задачи 1.1

**Решение задачи 1.2**

**Формулировка задачи**

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 2 – Условие задачи 1.2

|  |  |
| --- | --- |
| **Вариант** | **Задача** |
| **5** | Перевести значение веса, выраженное в граммах, в унции (1 унция =28.3 г) |

# Блок-схема алгоритма

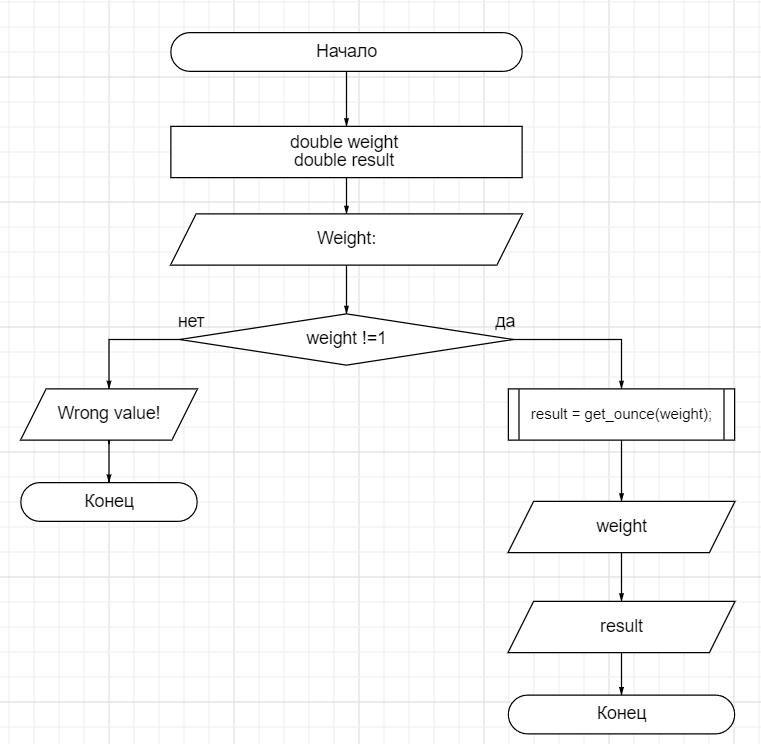
Ниже представлена блок-схема алгоритма (Рисунок 7, Рисунок 8) 

Рисунок 7 – блок-схема функции main()

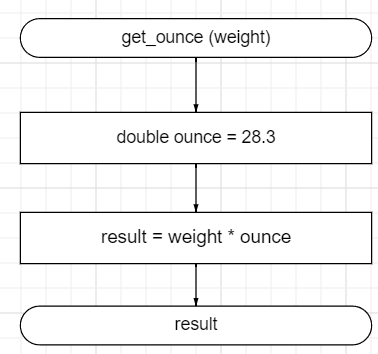


Рисунок 8 – блок-схема функции get\_ounce ()

**Поясняющие формулы**

– где:

ounce – 1 унция =28.3

weight – вес в граммах

**Код для задания 1.2**

#define EXIT\_SUCCESS /\*implementation defined\*/

#define EXIT\_FAILURE /\*implementation defined\*/

#include <stdio.h>

/\*\*

\* @breaf Перевод введеного значения из грамм в унции

\* @param weight значение в граммах

\* @param ounce значение константы ounce

\* @return Возвращение результата в унциях

\*/

double get\_ounce(double weight);

/\*\* @breaf точка входа в программу

@return EXIT\_FAILURE если неверный ввод

@return EXIT\_SUCCESS если ввод верный

\*/

int main()

{

double weight=0.0;

double result = 0.0;

printf\_s("% s", "Weight=");

if (scanf\_s("%lf", &weight) != 1)

{

printf\_s("%s", "Wrong value");

return EXIT\_FAILURE

abort();

}

result = get\_ounce(weight);

printf\_s("weight = %lf", weight);

printf\_s("\nresult=%lf", result);

return EXIT\_SUCCESS;

}

double get\_ounce(double weight)

{

const double ounce = 28.3;

return weight \* ounce;

}

**Решение различных тестовых примеров на С**

Ниже представлены выводы программы 1.2 в С (Рисунок 9, Рисунок 10)



Рисунок 9 – Вывод программы 1.2, если значения угла заданно корректно

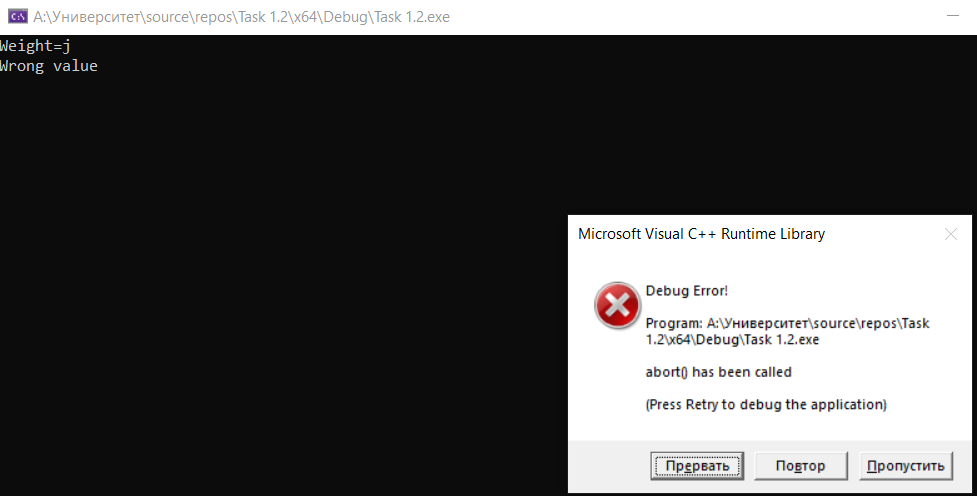


Рисунок 10 – Вывод программы 1.2, если значение угла задано буквой

**Решение различных тестовых примеров в** Excel

Ниже представлено решение задания 1.2 в Excel (Рисунок 11)



Рисунок 11 - Решение тестового примера задания 1.2 в Excel

**Подтверждение approve 1.2**

Ниже представлено доказательство того, что задание 1.2 было принято (Рисунок 12)

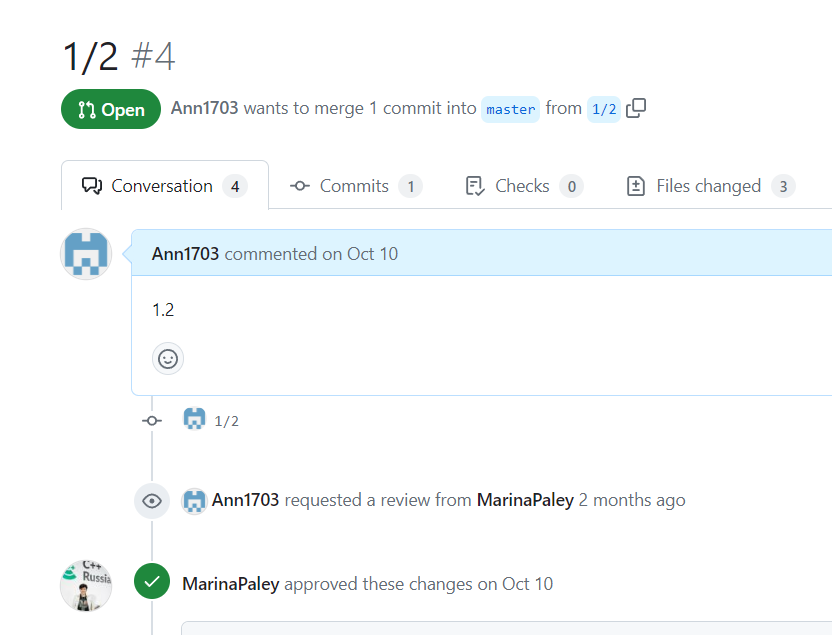


Рисунок 12 – Approve задачи 1.2

**Решение задания 1.3**

**Формулировка задачи**

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 3 – Условие задачи 1.3

|  |  |
| --- | --- |
| **Вариант** | **Задача** |
| **5** | Какую работу совершит электрический ток в электродвигателе вентилятора за 20 минут, если сила тока в цепи равна I А, а напряжение равно U В? |

**Блок-схема алгоритма**

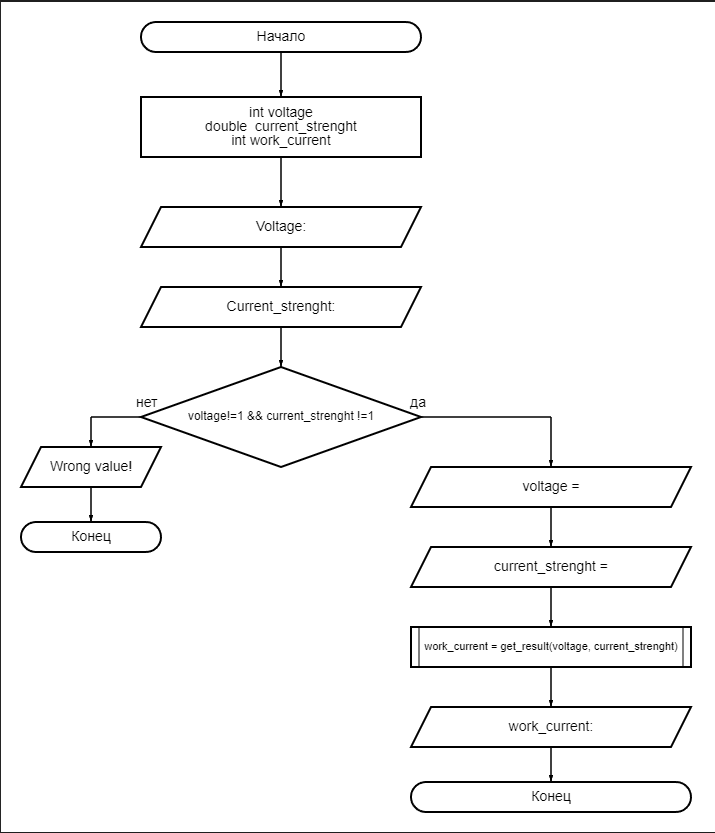
Ниже представлена блок-схема алгоритма (Рисунок 13, Рисунок 14)

Рисунок 13 – блок-схема функции main()

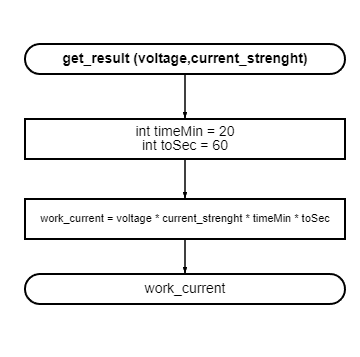


Рисунок 14 – блок-схема функции get\_result()

**Поясняющие формулы**

– где:

A=work\_current – работа электродвигателя

I=voltage – Сила тока

U=current\_strenght – Напряжение

T= timeMin\*toSec – Время

timeMin ­ – ­­ время в минутах

toSec – 1 минута в секундах = 60

**Код для задания 1.3**

#define EXIT\_SUCCESS /\*implementation defined\*/

#define EXIT\_FAILURE /\*implementation defined\*/

#include <stdio.h>

/\*\*

\* @breaf Расчёт работы электрического тока

\* @param voltage значение напряжения в вольтах

\* @param current\_strenght значения силы тока в амперах

\* @param work\_current значение работы электрического тока

\* @return Вывод полученного результата

\*/

double get\_result(int voltage, double current\_strenght);

/\*\* @breaf точка входа в программу

@return EXIT\_FAILURE если неверный ввод

@return EXIT\_SUCCESS если ввод верный

\*/

int main()

{

int voltage = 0;

double current\_strenght = 0.0;

int work\_current = 0;

printf\_s("% s", "Voltage=");

if (scanf\_s("%d", &voltage) != 1 && scanf\_s("%lf", &current\_strenght) != 1)

{

printf\_s("%s", "Wrong value");

return EXIT\_FAILURE;

}

printf\_s("\n% s", "Current\_strenght=");

if (scanf\_s("%lf", &current\_strenght) != 1)

{

printf\_s("%s", "Wrong value");

return EXIT\_FAILURE;

abort();

}

printf\_s("voltage = %d\n", voltage);

printf\_s("current\_strenght = %lf\n", current\_strenght);

work\_current = get\_result(voltage, current\_strenght);

printf\_s("\nwork\_current=%d", work\_current);

return EXIT\_SUCCESS;

}

double get\_result(int voltage, double current\_strenght)

{

const int timeMin = 20;

int toSec = 60;

return voltage \* current\_strenght \* timeMin \* toSec;

}

**Решение различных тестовых примеров на С**

Ниже представлены выводы программы 1.3 в С (Рисунок 14, Рисунок 15, Рисунок 16)

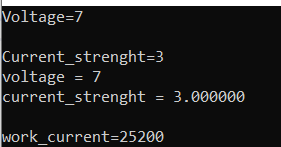


Рисунок 14 – Вывод программы 1.3 если сила тока в цепи и напряжение заданы корректно



Рисунок 15 – Вывод программы 1.3 если сила тока в цепи задана буквой

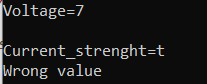


Рисунок 16 – Вывод программы 1.3 если напряжение задано буквой

**Решение различных тестовых примеров в Excel**

Ниже представлено решение задания 1.3 в Excel (Рисунок 17)

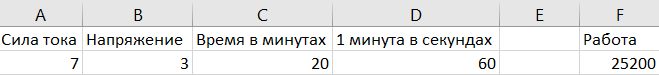


Рисунок 17 - Решение тестового примера задания 1.3 в Excel

**Подтверждение approve 1.3**

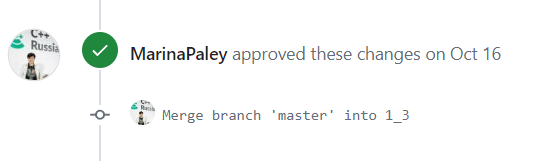
Ниже представлено доказательство того, что задание 1.3 было принято (Рисунок 18) 

Рисунок 18 – Approve задачи 1.3

Оглавление

[1 Решение задачи 1.1 1](#_Toc152272586)

[1.1 Формулировка задачи 2](#_Toc152272587)

[1.2 Блок-схема алгоритма 2](#_Toc152272588)

[1.3 Код для задания 1.1 3](#_Toc152272589)

[1.4 Решение различных тестовых примеров на С 4](#_Toc152272590)

[1.5 Решение различных тестовых примеров в Excel 4](#_Toc152272591)

[1.6 Подтверждение approve 1.1 4](#_Toc152272592)

[2 Решение задачи 1.2 6](#_Toc152272593)

[2.1 Формулировка задачи 6](#_Toc152272594)

[2.2 Блок-схема алгоритма 6](#_Toc152272595)

[2.3 Поясняющие формулы 7](#_Toc152272596)

[2.4 Код для задания 1.2 7](#_Toc152272597)

[2.5 Решение различных тестовых примеров на С 8](#_Toc152272598)

[2.6 Решение различных тестовых примеров в Excel 8](#_Toc152272599)

[2.7 Подтверждение approve 1.2 9](#_Toc152272600)

[3 Решение задания 1.3 10](#_Toc152272601)

[3.1 Формулировка задачи 10](#_Toc152272602)

[3.2 Блок-схема алгоритма 11](#_Toc152272603)

[3.3 Поясняющие формулы 11](#_Toc152272604)

[3.4 Код для задания 1.3 12](#_Toc152272605)

[3.5 Решение различных тестовых примеров на С 13](#_Toc152272606)

[3.6 Решение различных тестовых примеров в Excel 13](#_Toc152272607)

[3.7 Подтверждение approve 1.3 14](#_Toc152272608)