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 №2

По дисциплине «Языки программирования»

ВАРИАНТ 5

Выполнил: ст. гр. ТКИ-141

Буланый Сергей Евгеньевич

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

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

Москва 2023

Оглавление

[1 Решение задачи 1-1 3](#_Toc151483877)

[1.1 Формулировка задачи 2-1 3](#_Toc151483878)

[1.2 Блок-схема алгоритма задание 2–1 4](#_Toc151483879)

[1.3 Текст программы на языке С задание 2–1 5](#_Toc151483880)

[1.4 Результаты выполнения программы 2–1 9](#_Toc151483881)

[1.5 Выполнение тестовых примеров задание 2–1 12](#_Toc151483882)

[1.6 Отметка о выполнении задания в веб-хостинге системы контроля версий задание 2–1 17](#_Toc151483882)

[2 Решение задачи 2-2 18](#_Toc151483883)

[2.1 Формулировка задачи 2-2 18](#_Toc151483884)

[2.2 Блок-схема алгоритма задания 2-2 19](#_Toc151483885)

[2.3 Текст программы на языке С задание 2–2 20](#_Toc151483886)

[2.4 Результаты выполнения программы 2–2 24](#_Toc151483887)

[2.5 Выполнение тестовых примеров задание 2–2 28](#_Toc151483888)

[2.6 Отметка о выполнении задания в веб-хостинге системы контроля версий задание 2–2 16](#_Toc151483889)

[3 Решение задачи 1-3 29](#_Toc151483890)

[3.1 Формулировка задачи 1-3 29](#_Toc151483891)

[3.2 Блок-схема алгоритма задания 1-3 30](#_Toc151483892)

[3.3 Текст программы на языке С задание 1–3 33](#_Toc151483893)

[3.4 Результаты выполнения программы 1–3 36](#_Toc151483894)

[3.5 Выполнение тестовых примеров задание 1–3 38](#_Toc151483895)

[3.6 Отметка о выполнении задания в веб-хостинге системы контроля версий задание 1–3 41](#_Toc151483896)

1. Решение задачи 2–1

**1.1 Формулировка** **задачи 2–1**

Создать консольное приложение для решения задачи, представленной в таблице, с использованием перечислимого типа. Выбор формулы вычисления зависит от пользователя. Данные для решения задачи так же вводит пользователь. Ввод необходимо проверять на правильность. Все результаты вывести на экран. Дополнить свой отчёт блок-схемой алгоритма.

Таблица 1 – Исходные данные

|  |  |
| --- | --- |
| Вариант | Задачи |
| 5 | Заданы стороны прямоугольника. Определить его:   * периметр; * площадь; * длину диагонали. |

1.2 Блок-схема алгоритма задание 2–1

Блок-схема основного алгоритма представлена ниже (Рисунок 1). Блок-схемы функций расчета значений perimeter, square и diagonal представлены ниже (Рисунок 2).

Изображение выглядит как текст, диаграмма, План, Технический чертеж

Автоматически созданное описание

Рисунок 1 ­ Блок-схема основного алгоритма

Изображение выглядит как текст, диаграмма, снимок экрана, линия

Автоматически созданное описаниеИзображение выглядит как текст, диаграмма, снимок экрана, Шрифт

Автоматически созданное описаниеИзображение выглядит как текст, снимок экрана, Шрифт, диаграмма

Автоматически созданное описание

Рисунок 2 – Блок-схема используемых функций

1.3 Текст программы на языке С задание 2–1

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

#include <errno.h>

#include <float.h>

#include <math.h>

/\*\*

\* @brief Пользовательский ввод

\*/

enum user\_input

{

/\*\*

\* @brief Периметр

\*/

PERIMETER = 1,

/\*\*

\* @brief Площадь

\*/

SQUARE = 2,

/\*\*

\* @brief Длина диагонали

\*/

DIAGONAL = 3

};

/\*\*

\* @brief Получает целое значение

\* @remarks Экстренное завершение программы, в случае неправильного ввода

\* @return Целое значение

\*/

int get\_choice();

/\*\*

\* @brief Функция для проверки ввода стороны фигуры

\* @remarks Экстренное завершение программы, в случае неправильного ввода

\* @return Сторона фигуры

\*/

double get\_side();

/\*\*

\* @brief Функция для подсчёта периметра прямоугольника

\* @param a - первая сторона прямоугольника

\* @param b - вторая сторона прямоугольника

\* @return Периметр прямоугольника

\*/

double get\_perimeter(double a, double b);

/\*\*

\* @brief Функция для подсчёта площади прямоугольника

\* @param a - первая сторона прямоугольника

\* @param b - вторая сторона прямоугольника

\* @return Площадь прямоугольника

\*/

double get\_square(double a, double b);

/\*\*

\* @brief Функция для подсчёта периметра прямоугольника

\* @param a - первая сторона прямоугольника

\* @param b - вторая сторона прямоугольника

\* @return Периметр прямоугольника

\*/

double get\_diagonal(double a, double b);

/\*\*

\* @brief Точка входа в программу

\* @return Возврящает 0, если программа работает верно, иначе 1

\*/

int main()

{

setlocale(LC\_ALL, "RU");

puts("Введите номер соответствующий заданию:\n");

printf\_s("%d - для вычисления пириметра прямоугольника\n", PERIMETER);

printf\_s("%d - для вычисления площади прямоугольника\n", SQUARE);

printf\_s("%d - для вычисления длины диагонали прямоугольника\n", DIAGONAL);

int choice = get\_choice();

enum UserInput figure = (enum user\_input)choice;

puts("Введите первую сторону прямоугольника: ");

double a = get\_side();

puts("Введите вторую сторону прямоугольника: ");

double b = get\_side();

switch (figure)

{

case PERIMETER:

{

double perimeter = get\_perimeter(a, b);

printf\_s("%lf - периметр прямоугольника", perimeter);

break;

}

case SQUARE:

{

double squar = get\_square(a, b);

printf\_s("%lf - площадь прямоугольника", squar);

break;

}

case DIAGONAL:

{

double diagonal = get\_diagonal(a, b);

printf\_s("%lf - длина диагонали треугольника", diagonal);

break;

}

default:

{

errno = EDOM;

perror("Ошибка ввода");

return EXIT\_FAILURE;

}

}

return EXIT\_SUCCESS;

}

int get\_choice()

{

int choice;

int result = scanf\_s("%d", &choice);

if (result != 1)

{

errno = EIO;

perror("Ошибка ввода");

abort();

}

return choice;

}

double get\_side()

{

double side;

int result = scanf\_s("%lf", &side);

if (result != 1 || side <= 0)

{

errno = EIO;

perror("Введено неправильное значение");

abort();

}

return side;

}

double get\_perimeter(double a, double b)

{

return (a + b) \* 2;

}

double get\_square(double a, double b)

{

return a \* b;

}

double get\_diagonal(double a, double b)

{

return sqrt((a \* a) + (b \* b));

}

* 1. Результаты выполнения программы 2–1

Результаты выполнения программы в C представлена ниже (Рисунок 3-11).

Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 3 – Результаты выполнения программы если a =2 и b=0,4

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание

Рисунок 4 – Результаты выполнения программы если a =1 и b=-4

Изображение выглядит как текст, снимок экрана, программное обеспечение, Операционная система

Автоматически созданное описание

Рисунок 5 – Результаты выполнения программы если a =1 и b=g

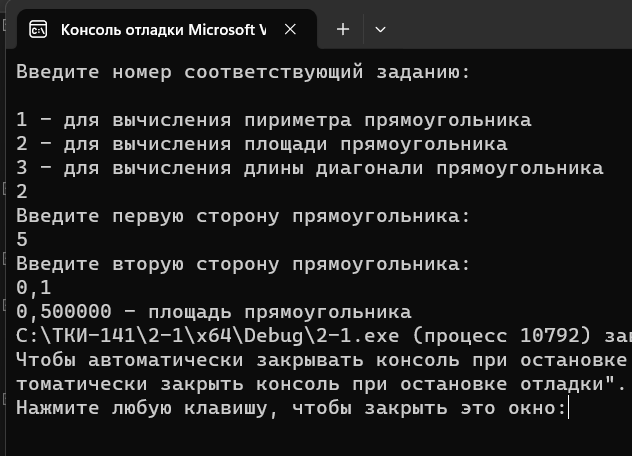


Рисунок 6 – Результаты выполнения программы если a =5 и b=0,1

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание

Рисунок 7 – Результаты выполнения программы если a =-6

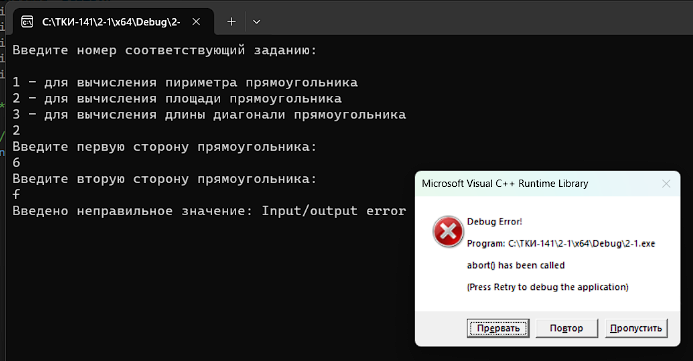


Рисунок 8 – Результаты выполнения программы если a =6 и b=f

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 9 – Результаты выполнения программы если a =4 и b=0,9

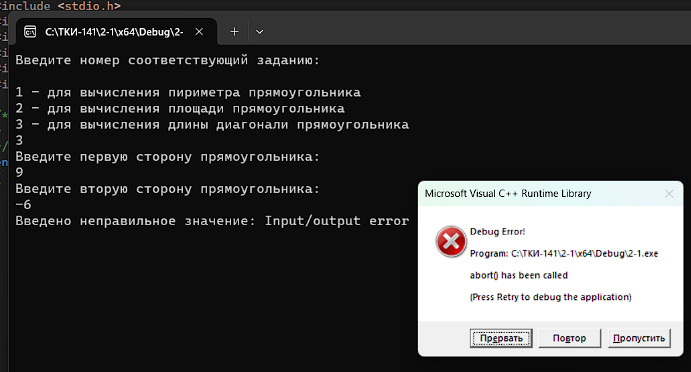


Рисунок 10 – Результаты выполнения программы если a =9 и b=-6

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 11 – Результаты выполнения программы если a =1 и b=l

* 1. Выполнение тестовых примеров задание 2–1

В программе Pythonвыполнены тестовые примеры. Результаты их выполнения представлены ниже (Рисунок 12-20).

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 12 – Результат расчета если a =2 и b=0,4

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Рисунок 13 – Результат расчета если a =1 и b=-4

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 14 – Результат расчета если a =1 и b=g

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Рисунок 15 – Результат расчета если a =5 и b=0,5

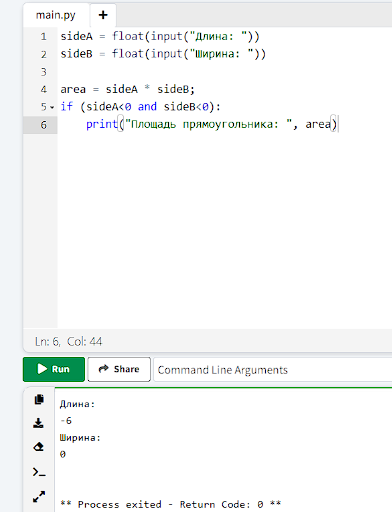


Рисунок 16 – Результат расчета если a =-6

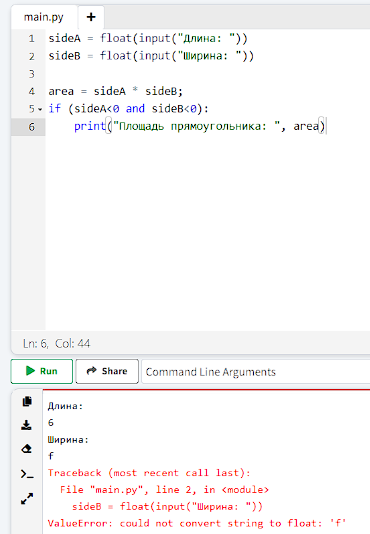


Рисунок 17 – Результат расчета если a =6 и b=f

Изображение выглядит как текст, снимок экрана, дисплей, программное обеспечение

Автоматически созданное описание

Рисунок 18 – Результат расчета если a =4 и b=0,9

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 19 – Результат расчета если a =9 и b=-6

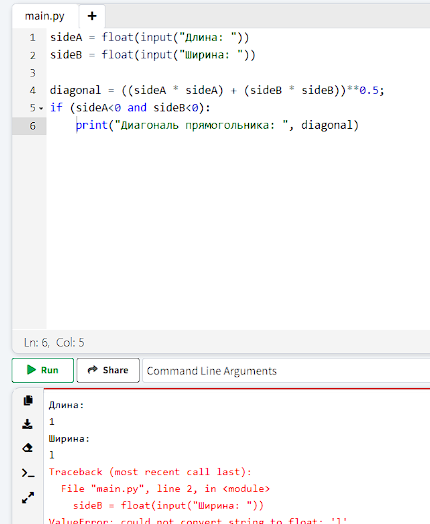


Рисунок 20 – Результат расчета если a =1 и b=l

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

Ниже представлено доказательство того, что задание 2–1 было принято. (Рисунок 21)

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описаниеРисунок 21 – Approve задачи 2–1

1. Решение задачи 2–2

**2.1 Формулировка** **задачи 2–2**

Создать консольное приложение с организацией разветвлённого процесса для решения заданной в таблице функции. Данные для решения вводит пользователь. Ввод необходимо проверять на правильность. Все результаты вывести на экран. Дополнить свой отчёт блок-схемой алгоритма.

Таблица 1 – Исходные данные

|  |  |  |
| --- | --- | --- |
| Вариант | Функция | Константы |
| 5 |  |  |

2.2 Блок-схема алгоритма задание 2–2

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

Изображение выглядит как текст, диаграмма, линия, Технический чертеж

Автоматически созданное описание

Рисунок 1 ­ Блок-схема основного алгоритма

2.3 Текст программы на языке С задание 2–2

#include <math.h>

#include <stdlib.h>

#include <errno.h>

#include <float.h>

#include <stdio.h>

/\*\*

\* @brief Функция расчитывающая значение y при x < 2.

\* @param x Входная величина.

\* @return Значение функции.

\*/

double get\_y\_1(double x);

/\*\*

\* @brief Функция расчитывающая значение y при x >= 2.

\* @param x Входная величина.

\* @return Значение функции.

\*/

double get\_y\_2(double x);

/\*\*

\* @brief Функция принимающая и проверяющая значение на ввод.

\* @return Значение.

\*/

double get\_value();

/\*\*

\* @brief Точка входа в программу

\* @return Возврящает 0, если программа работает верно, иначе 1

\*/

int main()

{

double x;

puts("Enter x: ");

x = get\_value();

double result = 0;

if (x < 2)

{

result = get\_y\_1(x);

}

else if (x >= 2)

{

result = get\_y\_2(x);

}

else

{

errno = EIO;

perror("Wrong value");

return 1;

}

printf("%lf", result);

return 0;

}

double get\_y\_1(double x)

{

double y = 1.5 \* (cos(x) \* cos(x));

return y;

}

double get\_y\_2(double x)

{

double y = ((x - 2) \* (x - 2)) + 6;

return y;

}

double get\_value()

{

double a;

int res = scanf\_s("%lf", &a);

if (res != 1)

{

errno = EIO;

perror("Wrong value");

abort();

}

return a;

}

* 1. Результаты выполнения программы 2–2

Результаты выполнения программы в C представлена ниже (Рисунок 2-9)

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 2 – Результаты выполнения программы если x =3

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 3 – Результаты выполнения программы если x =2

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 4 – Результаты выполнения программы если x =0,9

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

Рисунок 5 – Результаты выполнения программы если x =0

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 6 – Результаты выполнения программы если x =-5

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

Рисунок 7 – Результаты выполнения программы если x =-0,3

Изображение выглядит как текст, снимок экрана, Шрифт, мультимедиа

Автоматически созданное описание

Рисунок 8 – Результаты выполнения программы если x =100

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Автоматически созданное описание

Рисунок 9 – Результаты выполнения программы если x =h

* 1. Выполнение тестовых примеров задание 2–2

В программе Pythonвыполнены тестовые примеры. Результаты их выполнения представлены ниже (Рисунок 10-17).

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 10 – Результат расчета если x =3

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 11 – Результат расчета если x =2

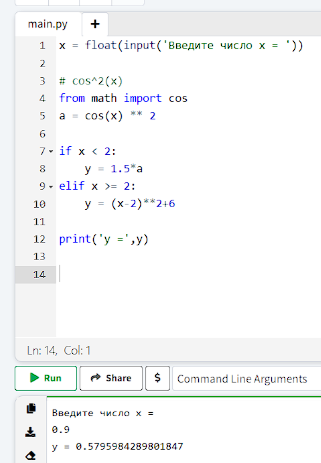


Рисунок 12 – Результат расчета если x =0.9

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 13 – Результат расчета если x =0

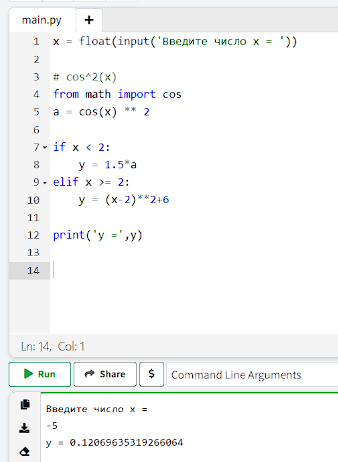


Рисунок 14 – Результат расчета если x =-5

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 15 – Результат расчета если x =-0,3

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 16 – Результат расчета если x = 100

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 17 – Результат расчета если x = h

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

Ниже представлено доказательство того, что задание 2–2 было принято. (Рисунок 18)

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 18 – Approve задачи 2–2

1. Решение задачи 2–3

**3.1 Формулировка** **задачи 2–3**

Создать консольное приложение с организацией разветвлённого процесса для решения задачи из таблицы. Данные для решения вводит пользователь. Ввод необходимо проверять на правильность. Все результаты вывести на экран. Дополнить свой отчёт блок-схемой алгоритма.

Таблица 1 – Исходные данные

|  |  |
| --- | --- |
| Вариант | Задачи |
| 5 | Путник двигался *t1* часов со скоростью *v1*, а затем *t2* часов – со скоростью *v2* и *t3* часов – со скоростью *v3*. За какое время он одолел первую половину пути, после чего запланировал привал? |

3.2 Блок-схема алгоритма задание 2–3

Блок-схема основного алгоритма представлена ниже (Рисунок 1). Блок-схемы функций расчета значений interval, t, way, def\_t и проверка представлены ниже (Рисунок 2-5).

Изображение выглядит как текст, снимок экрана, Шрифт, чек

Автоматически созданное описание

Рисунок 1 ­ Блок-схема основного алгоритма

Изображение выглядит как текст, диаграмма, линия, снимок экрана

Автоматически созданное описание

Рисунок 2 – Блок-схема используемой функции

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 3 – Блок-схема используемой функции

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описаниеИзображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 4 – Блок-схема используемой функции

Изображение выглядит как текст, диаграмма, снимок экрана, линия

Автоматически созданное описание

Рисунок 5 – Блок-схема используемой функции

3.3 Текст программы на языке С задание 2–3

# include <stdio.h>

# include <locale.h>

# include <stdlib.h>

# include <errno.h>

# include <stdbool.h>

# include <float.h>

/\*\*

\* @brief Функция принимающая и проверяющая значение на ввод

\* @param message - текст сообщения для пользователя

\* @return Значение

\*/

double get\_value(const char\* message);

/\*\*

\* @brief Функция для нахождения времени за которое путник одолел первую половину пути

\* @param v1 - первая скорость

\* @param v2 - вторая скорость

\* @param v3 - третья скорость

\* @param t1 - первое время

\* @param t2 - второе время

\* @param t3 - треть время

\* @return Время за которое путник преодолел первую половину пути

\*/

double def\_t(double v1, double v2, double v3, double t1, double t2, double t3);

/\*\*

\* @brief Функция для нахождения растояния

\* @param v - скорость

\* @param t - время

\* @return Растояние которое прошёл путник

\*/

double get\_way(double v, double t);

/\*\*

\* @brief Функция для нахождения времени

\* @param s - растояние

\* @param v - скорость

\* @return Время которое прошёл путник

\*/

double get\_time(double s, double v);

/\*\*

\* @brief Функция для приверки интервала

\* @param s - растояние

\* @param half\_way - пол пути

\* @return Интервал

\*/

bool belong\_to\_interval(double s, double half\_way);

/\*\*

\* @brief Точка входа в программу

\* @return Возвращает 0, если программа работает верно, иначе 1

\*/

int main()

{

setlocale(LC\_ALL, "RU");

double v1 = get\_value("Введите первую скорость (v1): ");

double v2 = get\_value("Введите вторую скорость (v2): ");

double v3 = get\_value("Введите третью скорость (v3): ");

double t1 = get\_value("Введите первое время (t1): ");

double t2 = get\_value("Введите второе время (t2): ");

double t3 = get\_value("Введите третье время (t3): ");

printf("Время за которое путник преодолел первую половину пути: %lf", def\_t(v1, v2, v3, t1, t2, t3));

return 0;

}

double get\_value(const char\* message)

{

double a;

printf("%s", message);

int res = scanf\_s("%lf", &a);

if (res != 1 || a <= 0)

{

errno = EIO;

perror("Wrong value");

abort();

}

return a;

}

double get\_way(double v, double t)

{

return v \* t;

}

double def\_t(double v1, double v2, double v3, double t1, double t2, double t3)

{

const double s1 = get\_way(v1, t1);

const double s2 = get\_way(v2, t2);

const double s3 = get\_way(v3, t3);

double half\_way = (s1 + s2 + s3) / 2;

if (belong\_to\_interval(s1, half\_way))

{

return get\_time(half\_way, v1);

}

if (belong\_to\_interval(s1 + s2, half\_way))

{

return t1 + get\_time(half\_way - s1, v2);

}

if (belong\_to\_interval(s1 + s2 + s3, half\_way))

{

return t1 + t2 + get\_time(half\_way - s1 - s2, v3);

}

}

double get\_time(double s, double v)

{

return s / v;

}

bool belong\_to\_interval(double s, double half\_way)

{

return s - half\_way > -DBL\_EPSILON;

}

3.4 Результаты выполнения программы 2–3

Результаты выполнения программы в C представлена ниже (Рисунок 6-10)

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 6 – Результаты выполнения программы если v1=1, v2=2, v3=3, t1=1, t2=2, t3=3

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Автоматически созданное описание

Рисунок 7 – Результаты выполнения программы если v1=4, v2=6, v3=8, t1=1, t2=5, t3=-3,4

Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание

Рисунок 8 – Результаты выполнения программы если v1=10, v2=-8,2

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 9 – Результаты выполнения программы если v1=4, v2=7, v3=12, t1=16, t2=f

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание

Рисунок 10 – Результаты выполнения программы если v1=13, v2=24, v3=s

3.5 Выполнение тестовых примеров задание 2–3

В программе Pythonвыполнены тестовые примеры. Результаты их выполнения представлены ниже (Рисунок 11-15).

Изображение выглядит как текст, снимок экрана, программное обеспечение, компьютер

Автоматически созданное описание

Рисунок 11 – Результат расчета если v1=1, v2=2, v3=3, t1=1, t2=2, t3=3

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 12 – Результат расчета если v1=4, v2=6, v3=8, t1=1, t2=5, t3=-3,4

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 13 – Результат расчета если v1=10, v2=-8,2

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описание

Рисунок 14 – Результаты выполнения программы если v1=4, v2=7, v3=12, t1=16, t2=f

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 15 – Результаты выполнения программы если v1=13, v2=24, v3=s

3.6 Отметка о выполнении задания в веб-хостинге системы контроля версий задание 2–3

Ниже представлено доказательство того, что задание 2–3 было принято. (Рисунок 16)

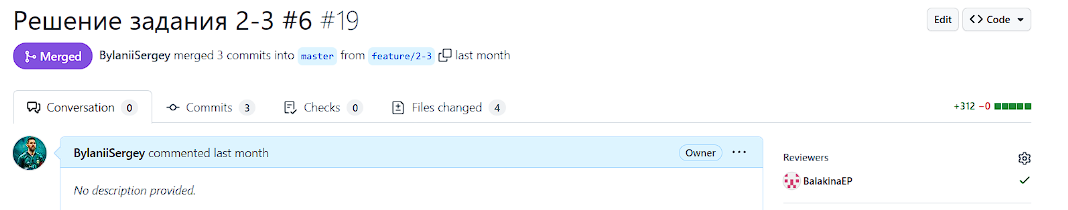


Рисунок 16 – Approve задачи 2–3