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 8

Выполнил: ст. гр. ТКИ-141

Гулин Игорь Алексеевич

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

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

Москва 2024

Содержание

[**Задание 1.1** 3](#_Toc184517262)

[1.Формулировка задачи. 3](#_Toc184517263)

[**2.Блок-схема алгоритма.** 3](#_Toc184517264)

[3.Решение задачи на языке программирования C. 5](#_Toc184517265)

[**4.Решение тестовых примеров.** 5](#_Toc184517266)

[**5.Решение тестовых примеров в Excel.** 6](#_Toc184517267)

[6. Отметка о выполнении задания в веб-хостинге системы контроля версий. 6](#_Toc184517268)

[**Задание 2.1** 7](#_Toc184517269)

[1.Формулировка задачи. 7](#_Toc184517270)

[**2.Блок-схема алгоритма.** 7](#_Toc184517271)

[3.Решение задачи на языке программирования C. 7](#_Toc184517272)

[**4.Решение тестовых примеров.** 10](#_Toc184517273)

[**5.Решение тестовых примеров в Python.** 10](#_Toc184517274)

[6. Отметка о выполнении задания в веб-хостинге системы контроля версий. 11](#_Toc184517275)

[**Задание 3.1** 12](#_Toc184517276)

[1.Формулировка задачи. 12](#_Toc184517277)

[**2.Блок-схема алгоритма.** 12](#_Toc184517278)

[3.Решение задачи на языке программирования C. 14](#_Toc184517279)

[**4.Решение тестовых примеров.** 15](#_Toc184517280)

[**5.Решение тестовых примеров в Python.** 16](#_Toc184517281)

[6. Отметка о выполнении задания в веб-хостинге системы контроля версий. 17](#_Toc184517282)

[**Задание 4.1** 18](#_Toc184517283)

[1.Формулировка задачи. 18](#_Toc184517284)

[2.Блок-схема алгоритма. 18](#_Toc184517285)

[3.Решение задачи на языке программирования C. 26](#_Toc184517286)

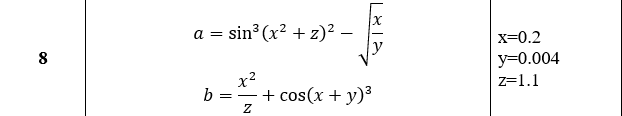
[4.Решение тестовых примеров. 31](#_Toc184517287)

[5.Решение тестовых примеров в Python. 32](#_Toc184517288)

[6. Отметка о выполнении задания в веб-хостинге системы контроля версий 32](#_Toc184517289)

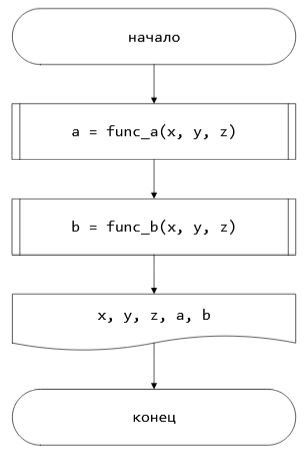
**Задание 1.1**

1.Формулировка задачи.

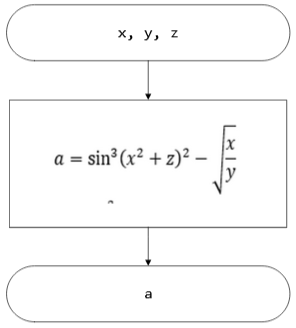


**2.Блок-схема алгоритма.**

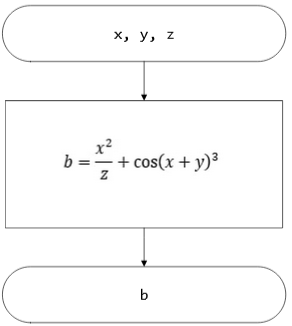
Блок-схемы алгоритмов функций представлены на рисунках (**Рисунок 1,Рисунок 2,Рисунок 3**).



**Рисунок 1.** **Блок-схема функции main ()**



**Рисунок 2. Блок-схема функции func\_a()**





**Рисунок 3. Блок-схема функции func\_b()**



3.Решение задачи на языке программирования C.

#include <math.h>

#include <stdio.h>

/\*\*

\* @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 func\_b(const double x, const double y, const double z);

/\*\*

\* @brief функция для вычисления значения а.

\* @param x Значение переменной x.

\* @param y Значение переменной y.

\* @param z Значение переменной z.

\* @return Результат вычисления а.

\*/

double func\_a(const double x, const double y, const double z);

/\*\*

\* @brief Точка входа в программу.

\* @return Возвращает в случае успеха.

\*/

int main(void)

{

const double x = 0.2;

const double y = 0.004;

const double z = 1.1;

printf("a = %f b = %f", func\_a(x, y, z), func\_b(x, y, z));

return 0;

}

double func\_a(const double x, const double y, const double z)

{

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

}

double func\_b(const double x, const double y, const double z)

{

return x \* x / z + cos(pow((x + y), 3));

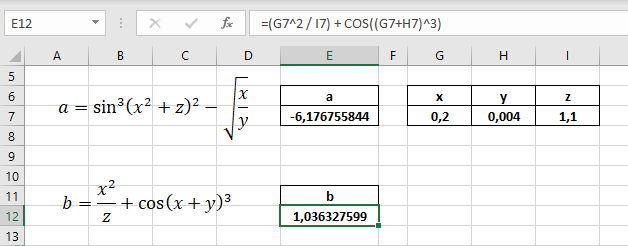
}

**4.Решение тестовых примеров.**

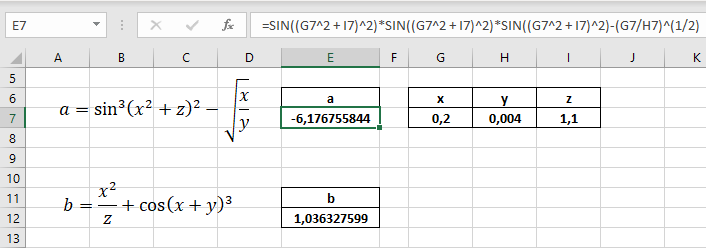


**Рисунок 4.Решение тестовых примеров**

**5.Решение тестовых примеров в Excel.**

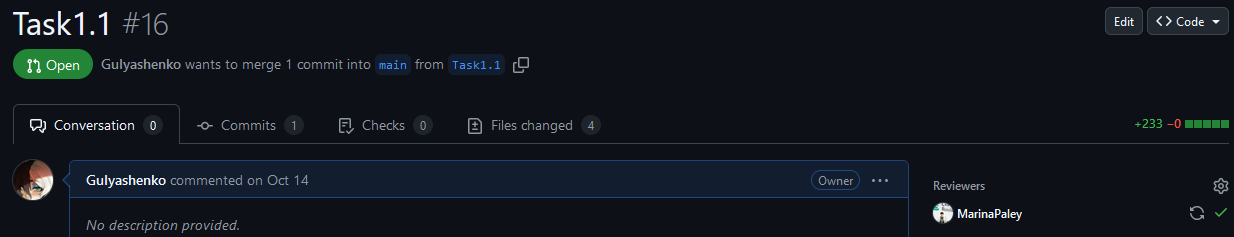


**Рисунок 5.Расчет значения b**



**Рисунок 6. Расчет значения a**

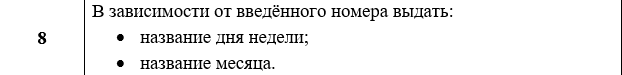
6. Отметка о выполнении задания в веб-хостинге системы контроля версий.



**Рисунок 7. Отметка о выполнении задания**

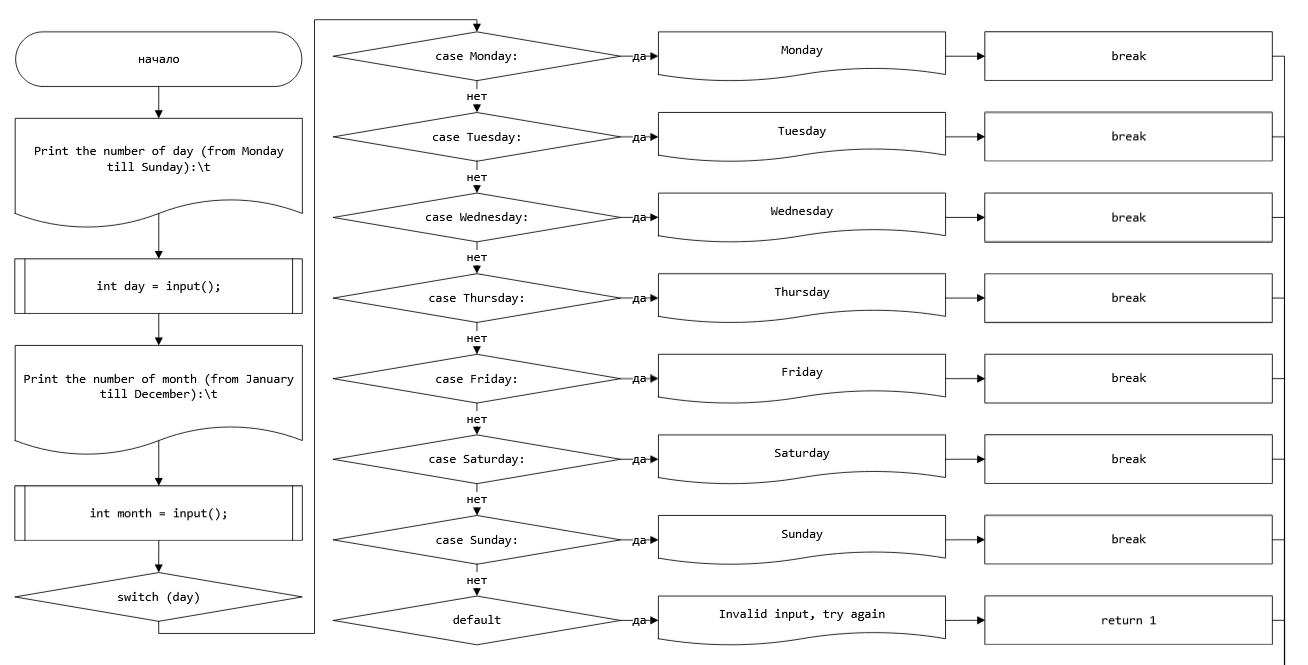
**Задание 2.1**

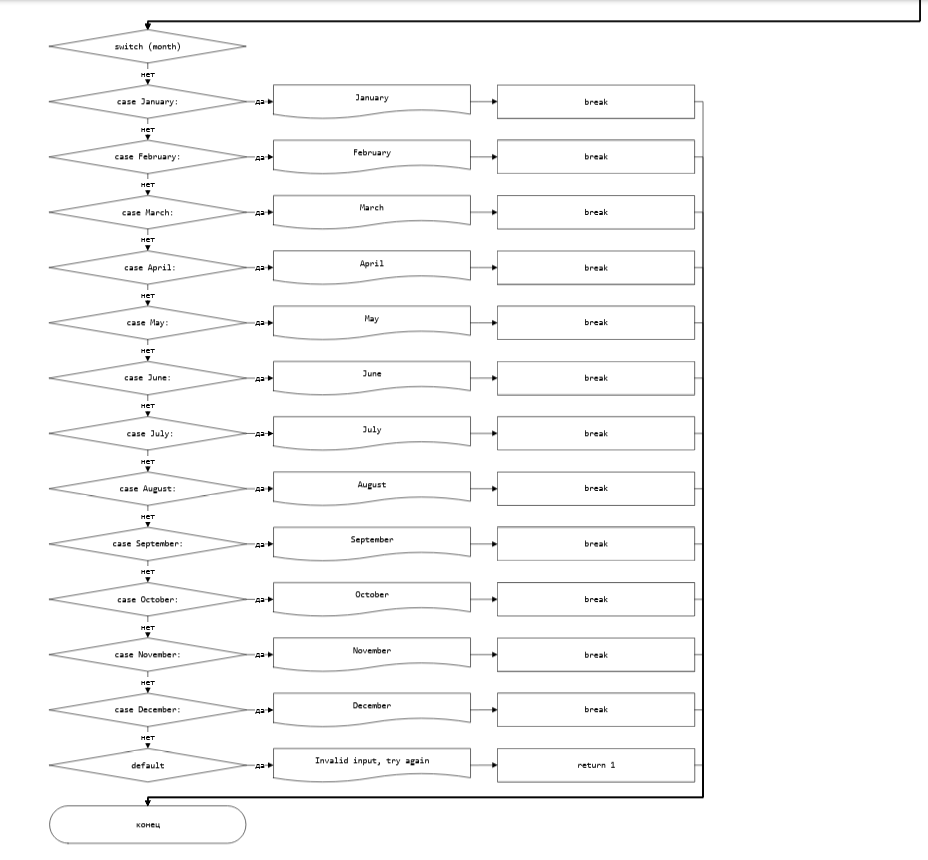
1.Формулировка задачи.



**2.Блок-схема алгоритма.**

Блок-схема алгоритма представлена на рисунке (***Рисунок 8***).





**Рисунок 8.Блок-схема алгоритма**

3.Решение задачи на языке программирования C.

#include <stdio.h>

#include <stdlib.h>

/\*\*

@brief перечисление названий дней недели

\*/

enum days {

Monday = 1,

Tuesday = 2,

Wednesday = 3,

Thursday = 4,

Friday = 5,

Saturday = 6,

Sunday = 7

};

/\*\*

@brief перечисление названий месяцев

\*/

enum months {

January = 1,

February = 2,

March = 3,

April = 4,

May = 5,

June = 6,

July = 7,

August = 8,

September = 9,

October = 10,

November = 11,

December = 12

};

int input(void);

/\*\*

@brief Точка входа в программу

@return Возвращает в случае успеха

\*/

int main(void) {

printf("Print the number of day (from Monday till Sunday):\t");

int day = input();

printf("Print the number of month (from January till December):\t");

int month = input();

switch (day) {

case Monday:

printf("Monday\n");

break;

case Tuesday:

printf("Tuesday\n");

break;

case Wednesday:

printf("Wednesday\n");

break;

case Thursday:

printf("Thursday\n");

break;

case Friday:

printf("Friday\n");

break;

case Saturday:

printf("Saturday\n");

break;

case Sunday:

printf("Sunday\n");

break;

default:

printf("Invalid input, try again\n");

return 1;

break;

}

switch (month) {

case January:

printf("January");

break;

case February:

printf("February");

break;

case March:

printf("March");

break;

case April:

printf("April");

break;

case May:

printf("May");

break;

case June:

printf("June");

break;

case July:

printf("July");

break;

case August:

printf("August");

break;

case September:

printf("September");

break;

case October:

printf("October");

break;

case November:

printf("November");

break;

case December:

printf("December");

break;

default:

printf("Invalid input, try again");

return 1;

break;

}

return 0;

}

int input(void) {

int number = 0;

if (scanf\_s("%d", &number) != 1) {

printf("Your input isn't a numbern");

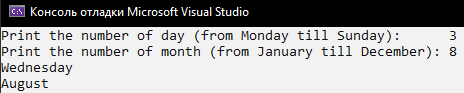
exit(EXIT\_FAILURE);

}

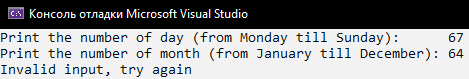
return number;

}

**4.Решение тестовых примеров.**

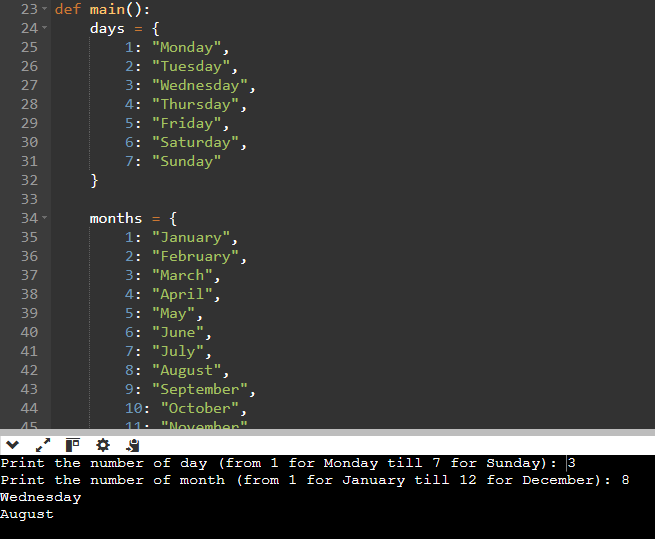


**Рисунок 9.Решение тестовых примеров**



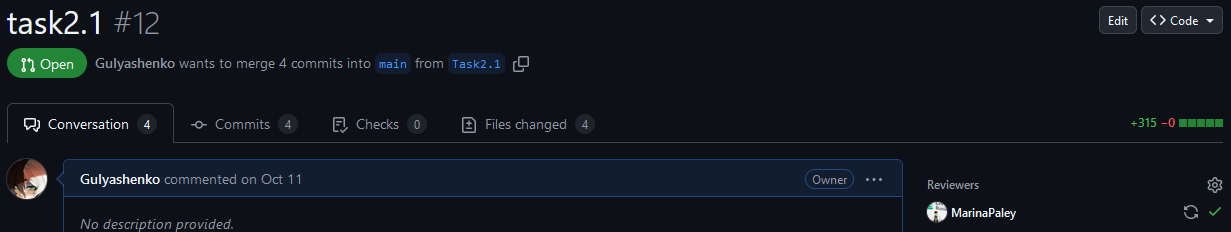
**Рисунок 10.Решение тестовых примеров**

**5.Решение тестовых примеров в Python.**



**Рисунок 11. Решение тестового примера**

6. Отметка о выполнении задания в веб-хостинге системы контроля версий.

****

**Рисунок 12.Отметка о выполнении задания**

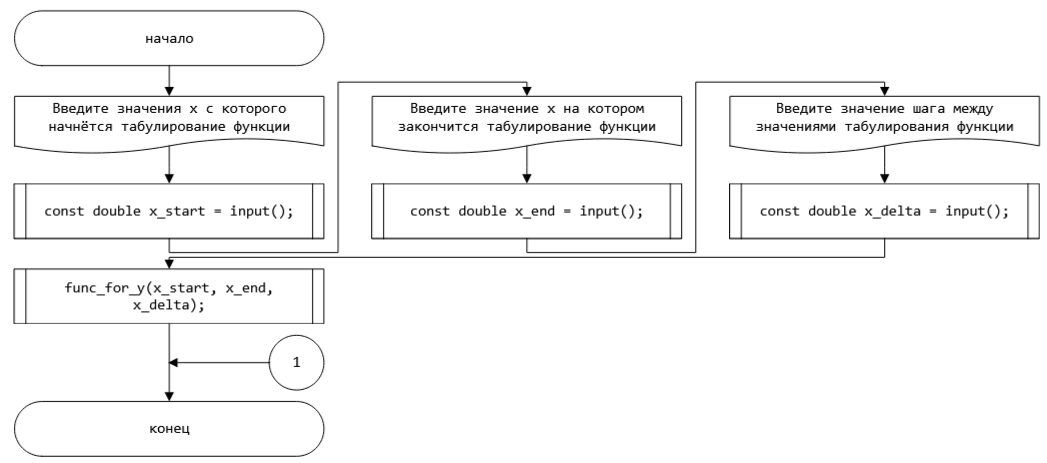
**Задание 3.1**

1.Формулировка задачи.

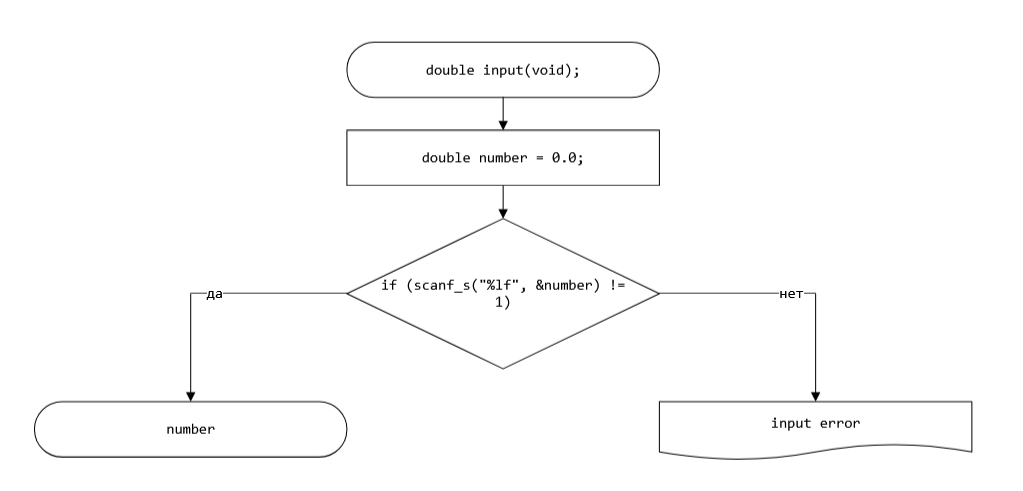


**2.Блок-схема алгоритма.**

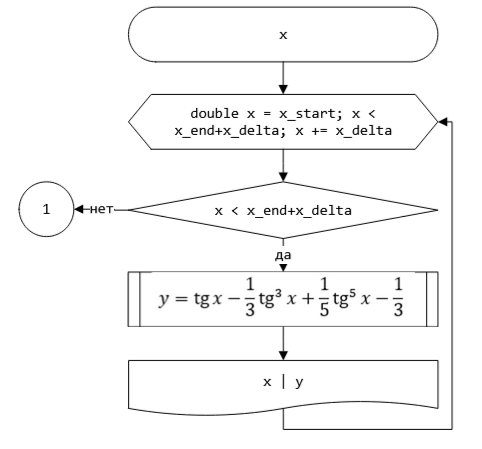
Блок-схема алгоритма представлена на рисунке (***Рисунок 13***,***Рисунок 14***,***Рисунок 15***).



**Рисунок 13.****Блок-схема функции main()**



**Рисунок 14.Блок-схема функции Input(void)**



**Рисунок 15.Блок-схема функции func\_for\_y(x\_start, x\_end, x\_delta)**

3.Решение задачи на языке программирования C.

#include <stdio.h>

#include <stdlib.h>

/\*\*

\* @brief проверяет вводимое значение(является ли числом)

\* @return возвращает значение если оно является числом

\*/

double input(void);

/\*\*

\* @brief считаеn значение y относительно значения х и выводит значение х и у

\* @param x\_start значение х с которого начнётся табулирование функции

\* @param x\_end значение х на котором закончится табулирование функции

\* @param x\_delta значение шага между значениями табулирования функции

\*/

void func\_for\_y(const double x\_start,const double x\_end,const double x\_delta);

/\*\*

\* @brief точка входа в программу

\* @return возвращает в случае успеха

\*/

int main(void) {

printf("Введите значения х с которого начнётся табулирование функции\t");

const double x\_start = input();

printf("Введите значение х на котором закончится табулирование функции\t");

const double x\_end = input();

printf("Введите значение шага между значениями табулирования функции\t");

const double x\_delta = input();

printf("x\t|\ty\n");

func\_for\_y(x\_start, x\_end, x\_delta);

return 0;

}

double input(void) {

double number;

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

printf("input error");

exit(EXIT\_FAILURE);

}

return number;

}

void func\_for\_y(const double x\_start,const double x\_end,const double x\_delta) {

for (double x = x\_start; x < x\_end+x\_delta; x += x\_delta) {

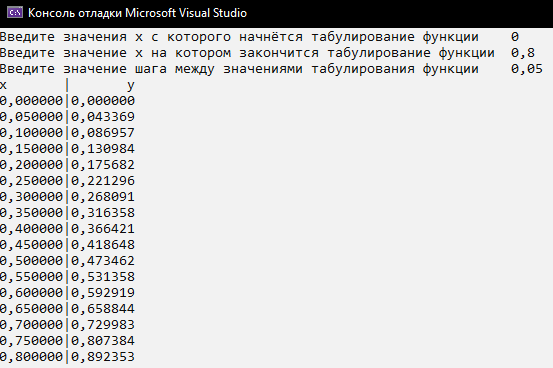
const double y = x - x / 3 + x / 5 - 1 / 3;

printf("%lf|%lf\n", x\_start, y);

}

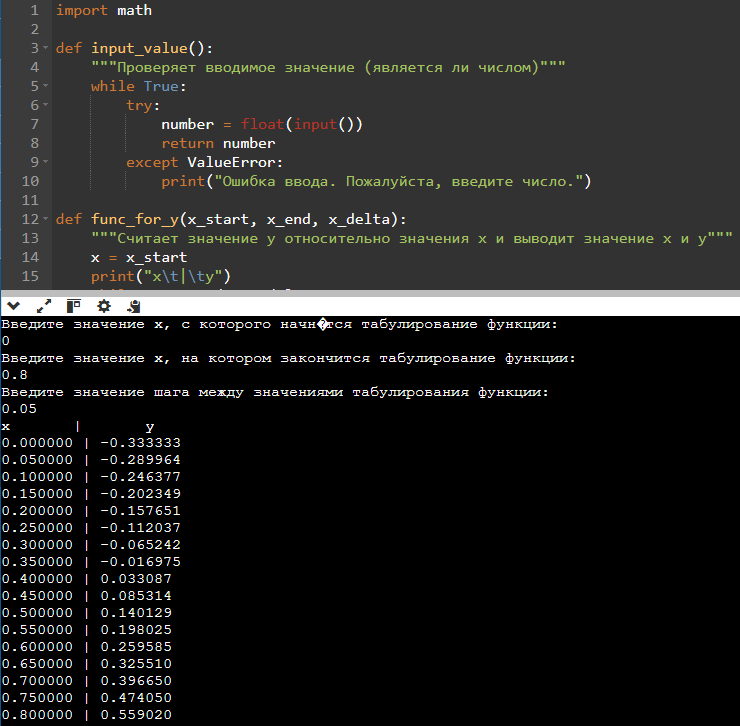
}

**4.Решение тестовых примеров.**



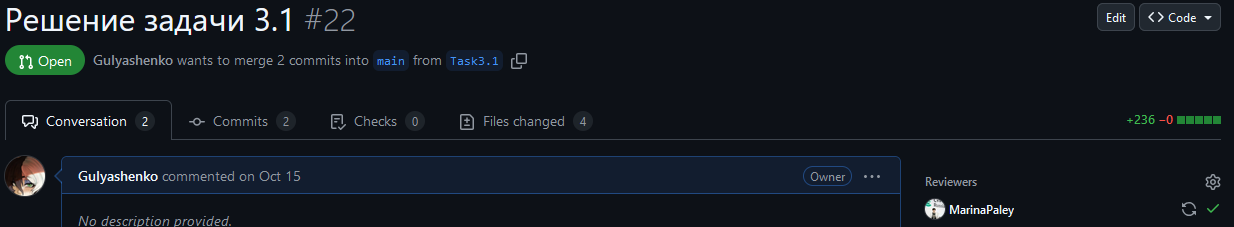
**Рисунок 16.Решение тестовых примеров**

**5.Решение тестовых примеров в Python.**



**Рисунок 17. Решение тестового примера**

6. Отметка о выполнении задания в веб-хостинге системы контроля версий.

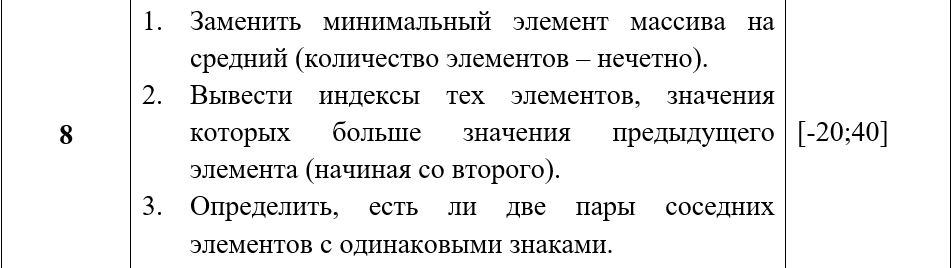
****

**Рисунок 18.Отметка о выполнении задания**

**Задание 4.1**

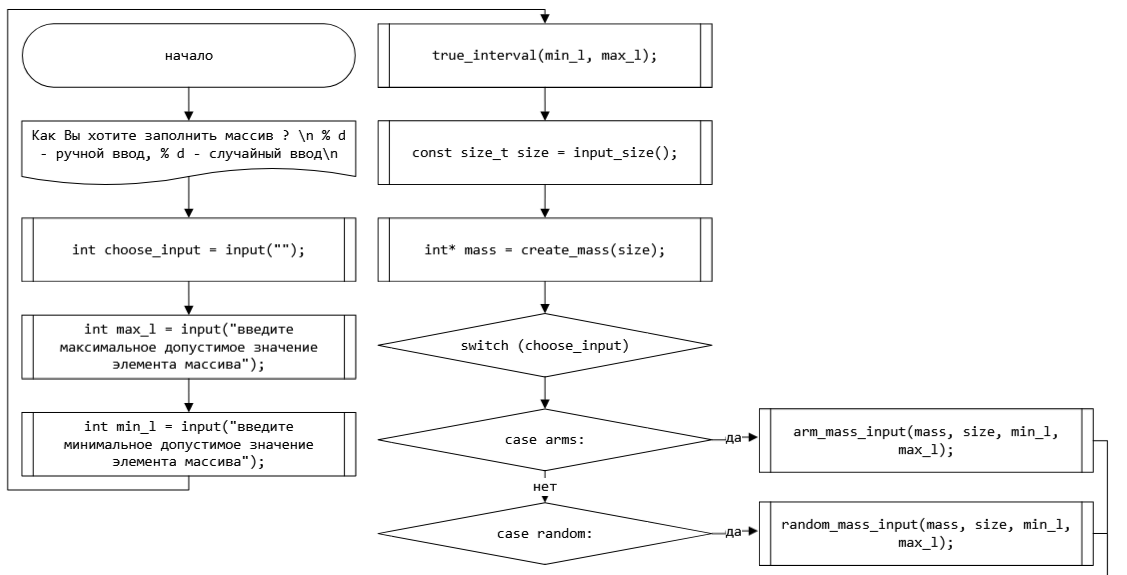
**1.Формулировка задачи**.

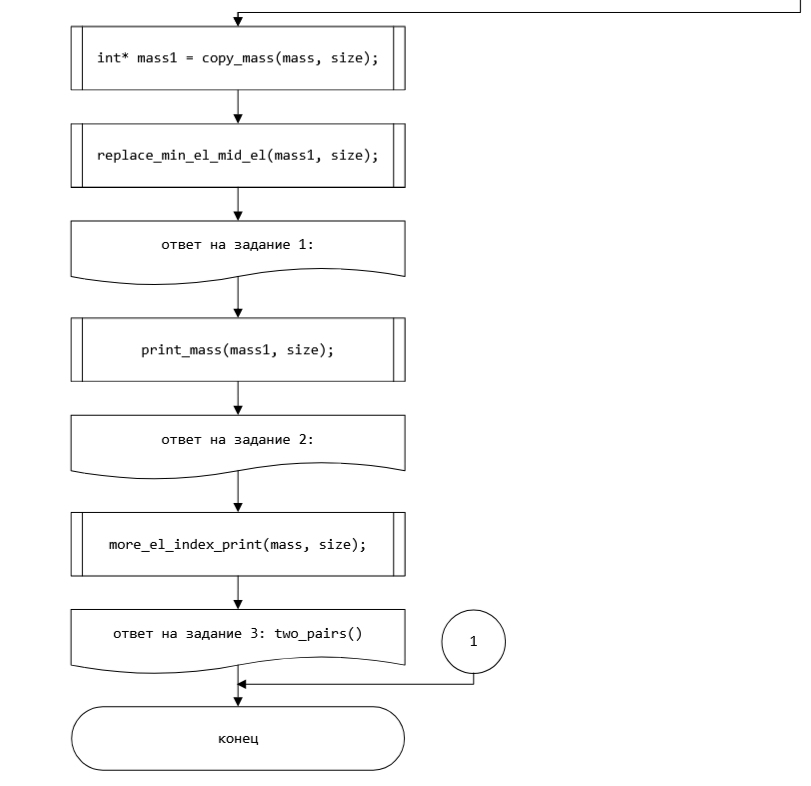
Создать одномерный массив из *n* целых чисел. Реализовать возможность заполнения массива, как случайными числами, так и с помощью клавиатуры по желанию пользователя. Пункты задания организовать в виде функций (методов). Составить блок-схему.



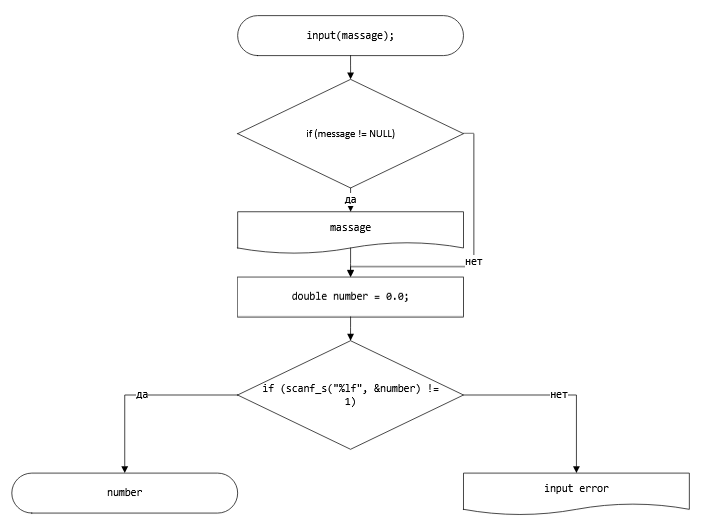
2.Блок-схема алгоритма.

Блок-схема алгоритма представлена на рисунке (***Рисунок 20***, ***Рисунок 21***, ***Рисунок 22***, ***Рисунок 23***, ***Рисунок 24***, ***Рисунок 25***, ***Рисунок 26***, ***Рисунок 27***, ***Рисунок 28***, ***Рисунок 29***, ***Рисунок 30***, ***Рисунок 31***, ***Рисунок 32***)

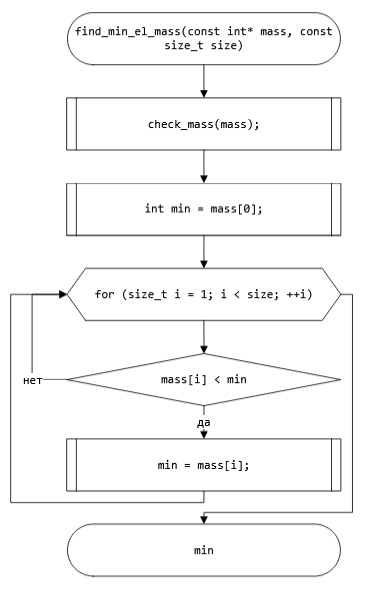




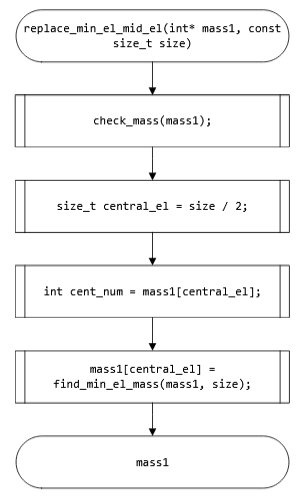
**Рисунок 19.Блок-схема функции main()**



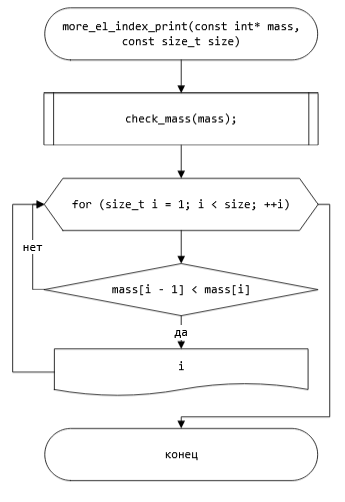
**Рисунок 20. Блок-схема функции input(massage)**



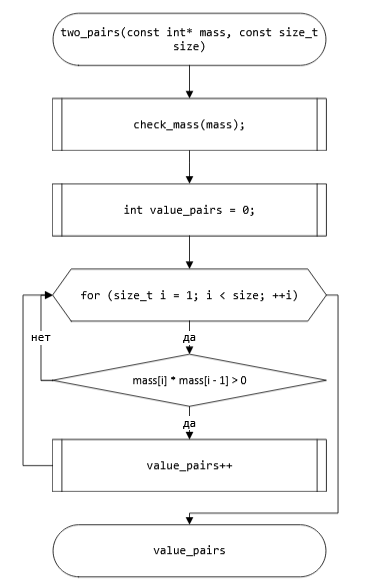
**Рисунок 21. Блок-схема функции find\_min\_el\_mass(mass, size)**



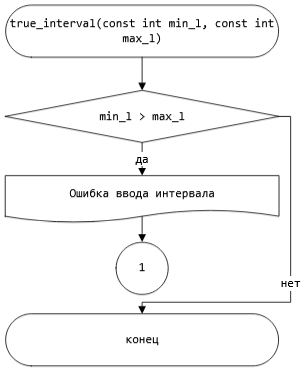
**Рисунок 22. Блок-схема функции replace\_min\_el\_mid\_el(mass1, size)**



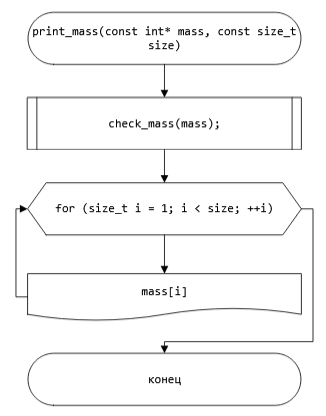
**Рисунок 23. Блок-схема функции more\_el\_index\_print(mass, size)**



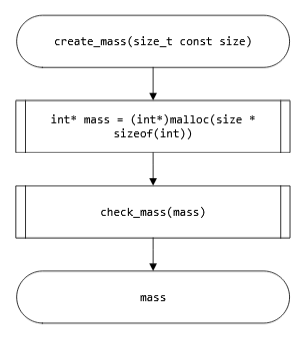
**Рисунок 24. Блок-схема функции two\_pairs(mass, size)**



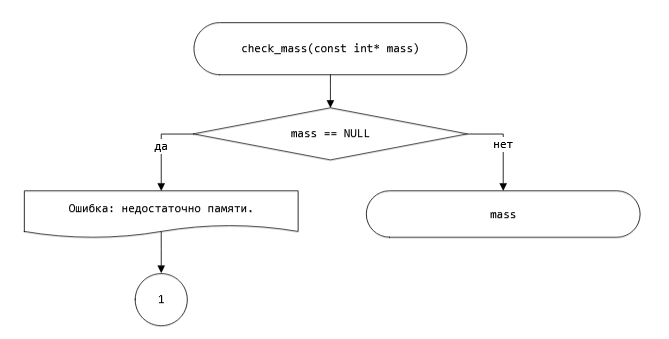
**Рисунок 25. Блок-схема функции true\_interval(min\_l, max\_l)**



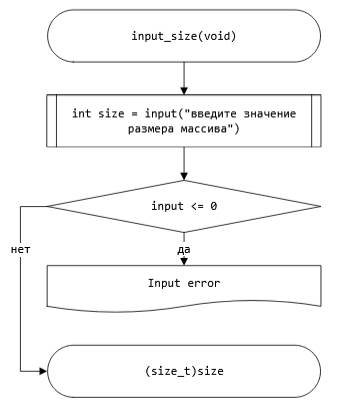
**Рисунок 26. Блок-схема функции print\_mass(mass, size)**



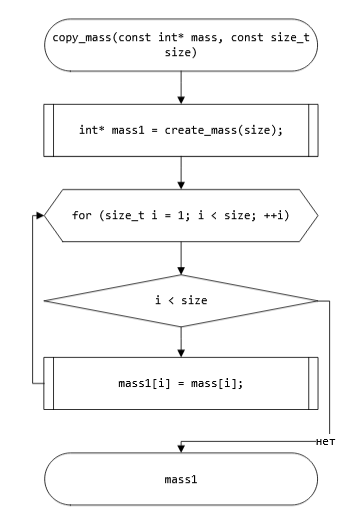
**Рисунок 27. Блок-схема функции create\_mass(size)**



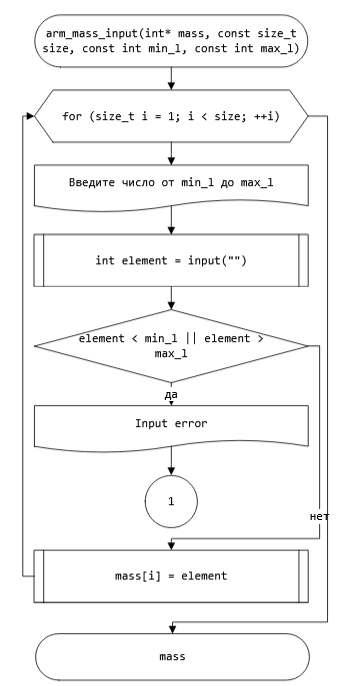
**Рисунок 28. Блок-схема функции check\_mass(mass)**



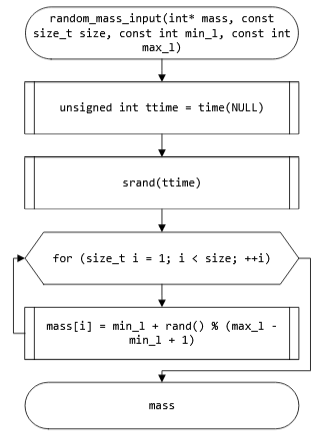
**Рисунок 29. Блок-схема функции input\_size(void)**



**Рисунок 30. Блок-схема функции copy\_mass(mass, size)**



**Рисунок 31. Блок-схема функции arm\_mass\_input(mass, size, min\_l, max\_l)**



**Рисунок 32. Блок-схема функции random\_mass\_input(mass, size, min\_l, max\_l)**

3.Решение задачи на языке программирования C.

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <locale.h>

/\*\*

\* @brief варианты заполнения массива числовыми значениями

\*/

enum mass\_input\_choise {

/\*\*

\* @brief ручным способом

\*/

arms = 1,

/\*\*

\* @brief заполнение случайными числами

\*/

random = 2

};

/\*\*

\* @brief проверяет вводимое значение (является ли числом)

\* @param сообщение которое нужно вывести перед вводом

\* @return возвращает введённое значение, если оно является числом

\*/

int input(const char\* message);

/\*\*

\* @brief Функция для печати массива.

\* @param mass - массив.

\* @param size - длинна массива.

\* @return Массив.

\*/

void print\_mass(const int\* mass, const size\_t size);

/\*\*

\* @brief Функция проверки интервала массива на правильность.

\* @param min\_l минимальный возможный элемент массива.

\* @param max\_l максимальный возможный элемент массива.

\*/

void true\_interval(const int min\_l, const int max\_l);

/\*\*

\* @brief функция для нахождения максимального значения массива

\* @param массив в котором мы ищем максимальный элемент

\* @param size - длинна массива.

\* @return возвращает значение максимального элемента массива

\*/

int find\_min\_el\_mass(const int\* mass, const size\_t size);

/\*\*

\* @brief функция для замены минимального элемента массива на центральный элемент того же массива

\* @param массив с которым мы работаем

\* @param размер массива

\* @return возвращает полученный массив

\*/

void replace\_min\_el\_mid\_el(int\* mass, const size\_t size);

/\*\*

\* @brief функция для вывода индексов элементов, которые больше предыдущих

\* @param массив с которым мы работаем

\* @param размер массива

\*/

void more\_el\_index\_print(const int\* mass, const size\_t size);

/\*\*

\* @brief функция для определения присутствия хотя бы двух пар соседних элементов с одным и тем же знаком

\* @param массив с которым мы работаем

\* @param размер массива

\* @return возвращает количество пар соседних элементов с одним знаком

\*/

int two\_pairs(const int\* mass, const size\_t size);

/\*\*

\* @brief Функция для ручного ввода элементов массива

\* @param size длинна массива

\* @param mass массив

\* @param min\_l значение минимально допустимого значения элемента массива

\* @param max\_l значение максимально допустимого значения элемента массива

\*/

void arm\_mass\_input(int\* mass, const size\_t size, const int min\_l, const int max\_l);

/\*\*

\* @brief Функция для рандомного ввода элементов массива

\* @param size длинна массива

\* @param mass массив

\* @param min\_l значение минимально допустимого значения элемента массива

\* @param max\_l значение максимально допустимого значения элемента массива

\*/

void random\_mass\_input(int\* mass, const size\_t size, const int min\_l, const int max\_l);

/\*\*

\* @brief Функция для проверки существования массива.

\* @param mass массив.

\* @remarks Экстренное завершение программы, в случае неправильной длинны массива.

\*/

void check\_mass(const int\* mass);

/\*\*

\* @brief проверяет вводимое значение

\* @remarks Экстренное завершение программы, в случае неправильной длинны массива.

\*/

size\_t input\_size(void);

/\*\*

\* @brief копирует массив

\* @param mass - массив который нужно скопировать

\* @param size - длинна массива

\* @remarks возвращает новый массив идентичный копируемому

\*/

int\* copy\_mass(const int\* mass, const size\_t size);

/\*\*

\* @brief Функция создания массива

\* @param размер создаваемого массива

\* @return возвращает созданный по размеру массив

\*/

int\* create\_mass(size\_t const size);

/\*\*

\* @brief точка входа в программу

\* @return возвращает в случае успеха

\*/

int main(void) {

setlocale(LC\_ALL, "Russian");

printf("Как Вы хотите заполнить массив ? \n % d - ручной ввод, % d - случайный ввод\n", arms, random);

int choose\_input = input("");

int max\_l = input("введите максимальное допустимое значение элемента массива");

int min\_l = input("введите минимальное допустимое значение элемента массива");

true\_interval(min\_l, max\_l);

const size\_t size = input\_size();

int\* mass = create\_mass(size);

switch (choose\_input) {

case arms:

arm\_mass\_input(mass, size, min\_l, max\_l);

case random:

random\_mass\_input(mass, size, min\_l, max\_l);

case random:

printf("Input error");

}

int\* mass1 = copy\_mass(mass, size);

replace\_min\_el\_mid\_el(mass1, size);

printf("ответ на задание 1:\t");

print\_mass(mass1, size);

printf("\nответ на задание 2:\t");

more\_el\_index\_print(mass, size);

printf("\nответ на задание 3:%d\t", two\_pairs(mass, size));

return 0;

}

int input(const char\* message) {

if (message != NULL) {

printf("%s", message);

}

int number = 0;

if (scanf\_s("%i", &number) != 1) {

printf("Input error");

exit(EXIT\_FAILURE);

}

return number;

}

int find\_min\_el\_mass(const int\* mass, const size\_t size) {

check\_mass(mass);

int min = mass[0];

for (size\_t i = 1; i < size; ++i) {

if (mass[i] < min) {

min = mass[i];

}

}

return min;

}

void replace\_min\_el\_mid\_el(int\* mass1, const size\_t size) {

check\_mass(mass1);

size\_t central\_el = size / 2;

int cent\_num = mass1[central\_el];

mass1[central\_el] = find\_min\_el\_mass(mass1, size);

}

void more\_el\_index\_print(const int\* mass, const size\_t size) {

check\_mass(mass);

for (size\_t i = 1; i < size; ++i) {

if (mass[i - 1] < mass[i]) {

printf("%Iu\t", i);

}

}

}

int two\_pairs(const int\* mass, const size\_t size) {

check\_mass(mass);

int value\_pairs = 0;

for (size\_t i = 1; i < size; ++i) {

if (mass[i] \* mass[i - 1] > 0) {

value\_pairs++;

}

}

return value\_pairs;

}

void print\_mass(const int\* mass, const size\_t size)

{

check\_mass(mass);

for (size\_t i = 0; i < size; i++)

{

printf("%d,", mass[i]);

}

}

void true\_interval(const int min\_l, const int max\_l)

{

if (min\_l > max\_l)

{

printf("Oшибка ввода интервала");

exit(EXIT\_FAILURE);

}

}

void arm\_mass\_input(int\* mass, const size\_t size, const int min\_l, const int max\_l) {

for (size\_t i = 0; i < size; i++)

{

printf("Введите число от %d до %d: ", min\_l, max\_l);

int element = input("");

if (element < min\_l || element > max\_l)

{

printf("input error");

exit(EXIT\_FAILURE);

}

mass[i] = element;

}

}

void random\_mass\_input(int\* mass, const size\_t size, const int min\_l, const int max\_l)

{

unsigned int ttime = time(NULL);

srand(ttime);

for (size\_t i = 0; i < size; i++)

{

mass[i] = min\_l + rand() % (max\_l - min\_l + 1);

}

}

void check\_mass(const int\* mass) {

if (mass == NULL)

{

printf("Ошибка: недостаточно памяти.\n");

exit("EXIT\_FAILURE");

}

}

size\_t input\_size(void)

{

int size = input("введите значение размера массива");

if (input <= 0)

{

puts("Input error");

exit(EXIT\_FAILURE);

}

return (size\_t)size;

}

int\* copy\_mass(const int\* mass, const size\_t size)

{

int\* mass1 = create\_mass(size);

for (size\_t i = 0; i < size; i++) {

mass1[i] = mass[i];

}

return mass1;

}

int\* create\_mass(size\_t const size)

{

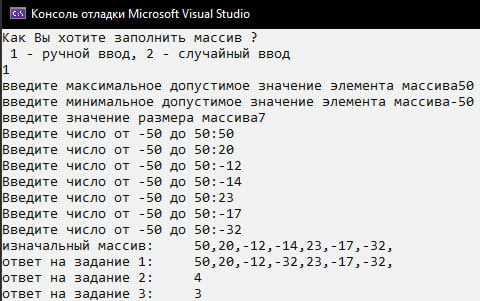
int\* mass = (int\*)malloc(size \* sizeof(int));

check\_mass(mass);

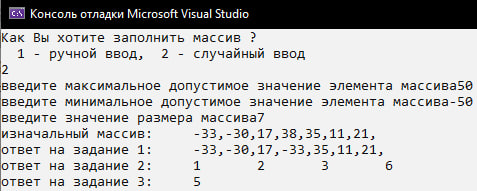
return mass;

}

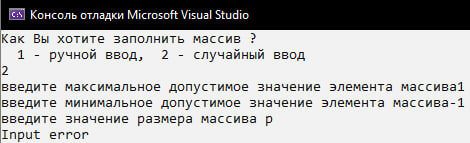
4.Решение тестовых примеров.



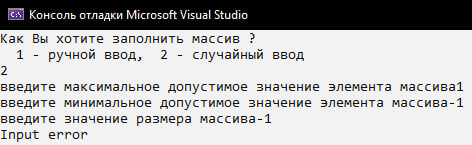
**Рисунок 33.Решение тестового примера с пользовательским вводом массива**



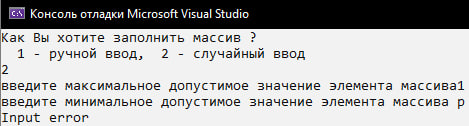
**Рисунок 34. Решение тестового примера с рандомным вводом массива**



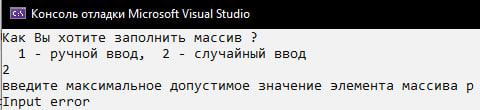
**Рисунок 35. Решение тестового примера если значение размера массива буква**



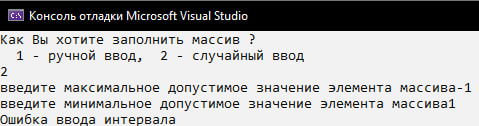
**Рисунок 36. Решение тестового примера когда размер массива отрицательное число**



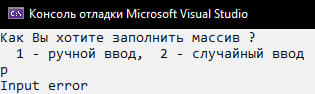
**Рисунок 37. Решение тестового примера когда минимум массива буква**



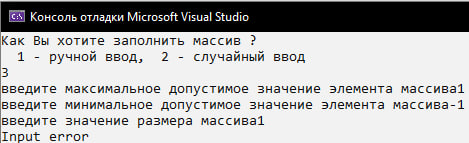
**Рисунок 38. Решение тестового примера когда максимум массива буква**



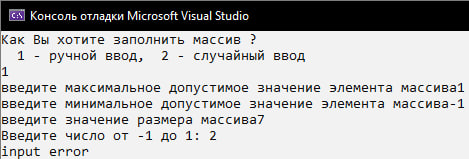
**Рисунок 39. Решение тестового примера когда минимум массива больше максимума**



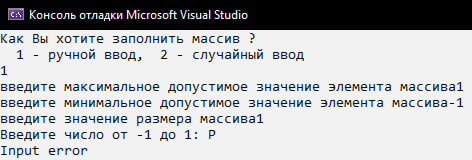
**Рисунок 40. Решение тестового примера когда выбор способа заполнения массива буква**



**Рисунок 41. Решение тестового примера когда выбор способа заполнения массива не входит в указанные значения**

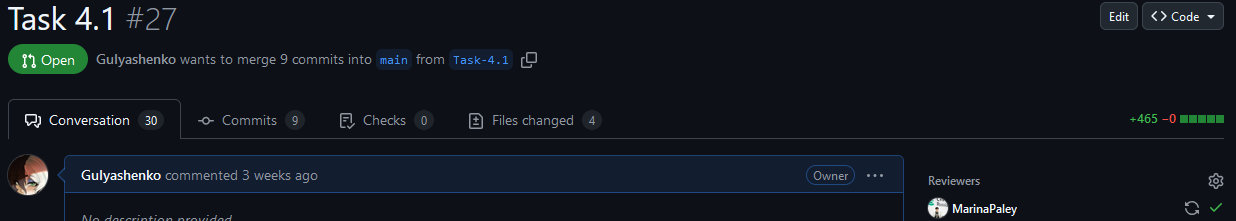


**Рисунок 42. Решение тестового примера когда пользовательский ввод элемента массива не входит в указанный промежуток**



**Рисунок 43. Решение тестового примера когда пользовательский ввод элемента массива буква**

5. Отметка о выполнении задания в веб-хостинге системы контроля версий



**Рисунок 45.Отметка о выполнении задания**