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 8

Выполнил: ст. гр. ТКИ - 142

Лазарев Андрей Александрович

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

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

Москва 2023

Оглавление

[1. Задание 2.1 3](#_Toc153118816)

[1.1. Формулировка задания 3](#_Toc153118817)

[1.2. Блок-схема алгоритма 4](#_Toc153118818)

[1.3. Код задания 2.1 8](#_Toc153118819)

[1.4. Решение тестовых примеров 12](#_Toc153118820)

[1.5. Зачёт задания в GitHub 14](#_Toc153118821)

[2. Задание 2.2 15](#_Toc153118822)

[2.1. Формулировка задания 15](#_Toc153118823)

[2.2. Блок-схема алгоритма 16](#_Toc153118824)

[2.3. Код задания 2.2 18](#_Toc153118825)

[2.4. Решение тестовых примеров 19](#_Toc153118826)

[2.5. Решение тестовых примеров в Excel 19](#_Toc153118827)

[2.6. Зачёт задания в GitHub 20](#_Toc153118828)

[3. Задание 2.1 21](#_Toc153118829)

[3.1. Формулировка задания 21](#_Toc153118830)

[3.2. Блок-схема алгоритма 22](#_Toc153118831)

[3.3. Код задания 2.3 24](#_Toc153118832)

[3.4. Решение тестового примера 27](#_Toc153118833)

[3.5. Зачёт задания в GitHub 28](#_Toc153118834)

# Задание 2.1

* 1. Формулировка задания

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  – Формулировка задания 2.1

|  |  |
| --- | --- |
| Вариант | Формулы |
| 8 | В зависимости от введённого номера выдать:   * название дня недели; * название месяца. |

* 1. Блок-схема алгоритма

Блок-схема алгоритма представлена на рисунках (Рисунок 1, Рисунок *2*, Рисунок *3*, Рисунок *4*)

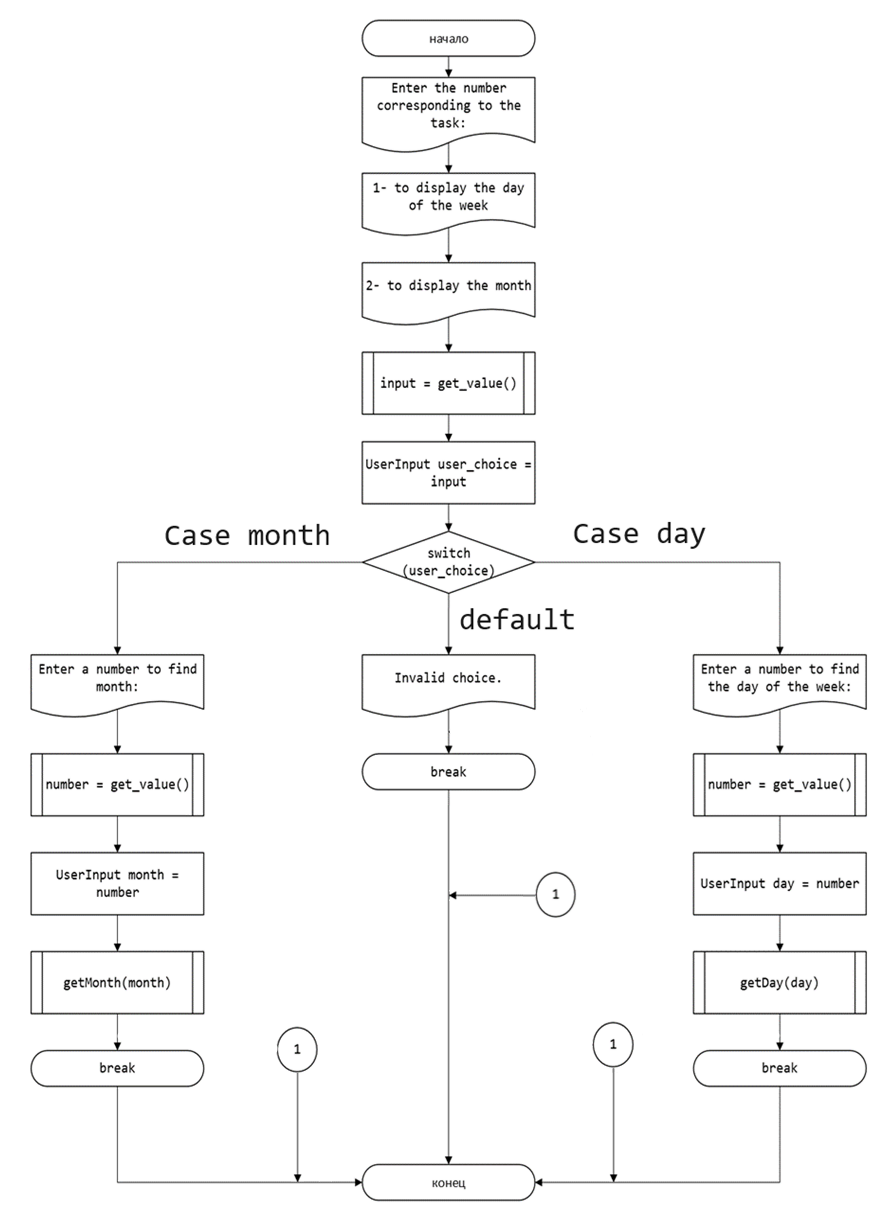


Рисунок 1 – Блок-схема к функции main()

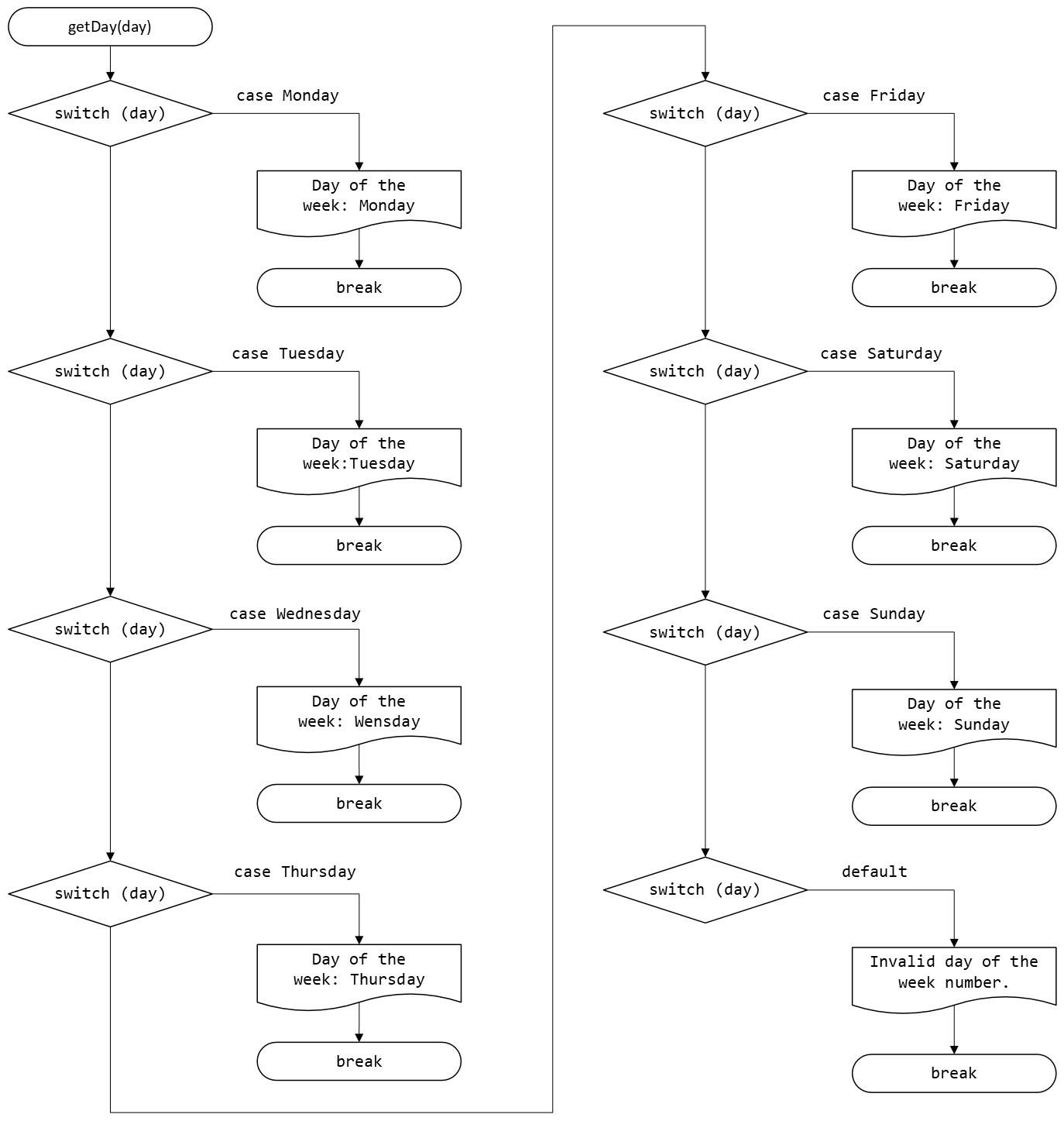


Рисунок 2 – Блок-схема функции getDay(day)

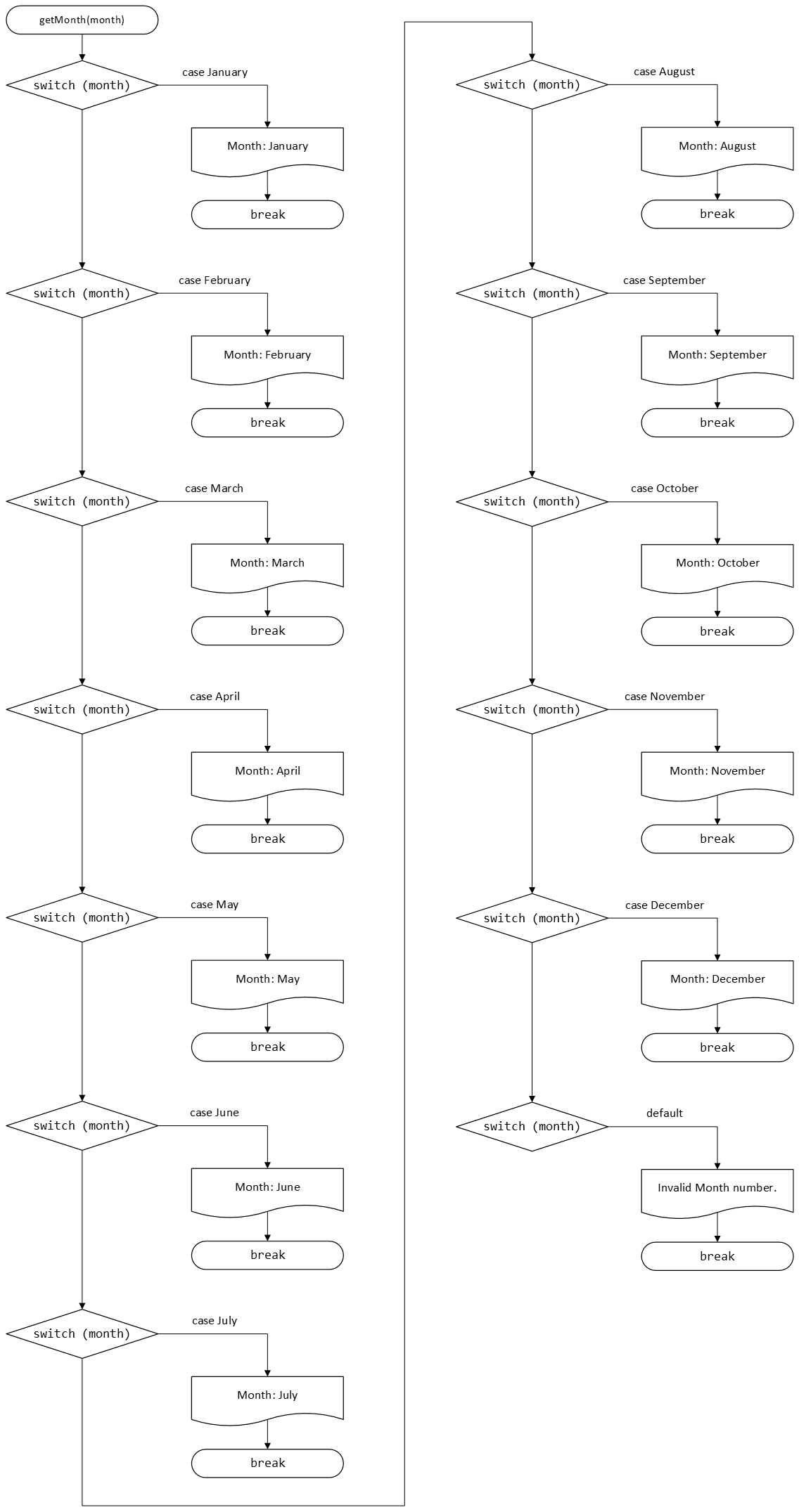


Рисунок 3 – Блок-схема функции getMonth(month)

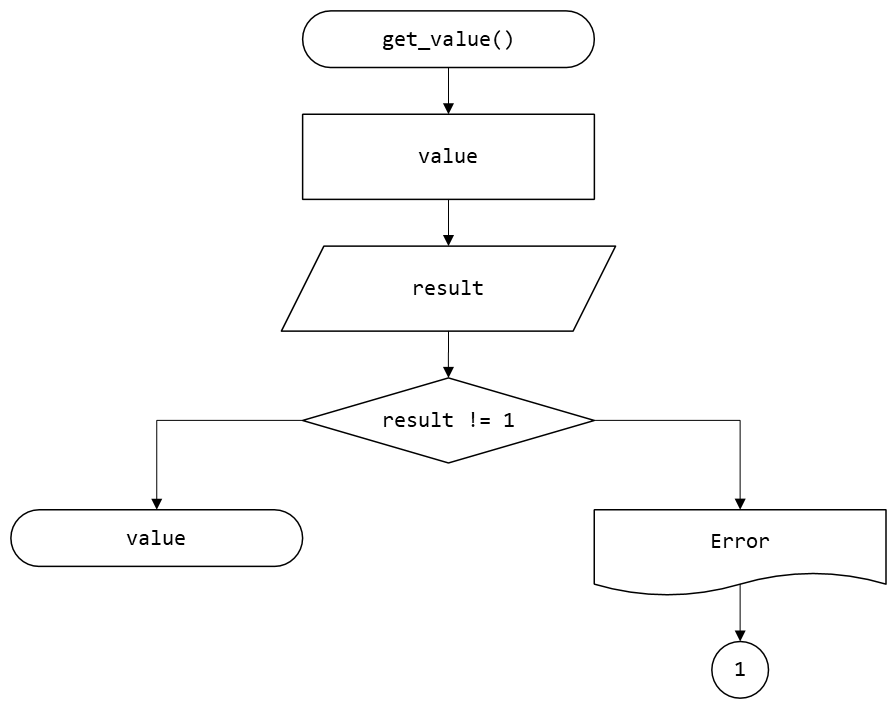


Рисунок 4 – Блок-схема функции get\_value()

* 1. Код задания 2.1

#include <stdio.h>

#include <errno.h>

#include <stdlib.h>

/\*\*

\* @brief Функция вывода дня недели и проверки ввода.

\* @param number Аргумент функции.

\* @return Возвращает название дня недели в случае успеха.

\*/

int getDay(enum day day);

/\*\*

\* @brief Функция вывода месяца и проверки ввода.

\* @param number Аргумент функции.

\* @return Возвращает название месяца в случае успеха.

\*/

int getMonth(enum month month);

/\*\*

\* @brief Пользовательский ввод.

\*/

enum choice

{

/\*\*

\* @brief День.

\*/

Day = 1,

/\*\*

\* @brief Месяц.

\*/

Month = 2

};

/\*\*

\* @brief Дни недели.

\*/

enum day

{

Monday = 1,

Tuesday,

Wednesday,

Thursday,

Friday,

Saturday,

Sunday

};

/\*\*

\* @brief Месяцы.

\*/

enum month

{

January = 1,

February,

March,

April,

May,

June,

July,

August,

September,

October,

November,

December

};

/\*\*

\* @brief Точка входа в программу.

\* @return Возвращает 0 в случае успеха.

\*/

int main()

{

puts("Enter the number corresponding to the task:\n");

printf\_s("%d - to display the day of the week\n", Day);

printf\_s("%d - to display the month\n", Month);

int input = get\_value();

enum UserInput user\_choice = (enum choice)input;

switch (user\_choice)

{

case Day:

{

printf\_s("%s", "Enter a number to find the day of the week: ");

int number = get\_value();

enum UserInput day = (enum choice)number;

getDay(day);

break;

}

case Month:

{

puts("Enter a number to find month: ");

int number = get\_value();

enum UserInput month = (enum choice)number;

getMonth(month);

break;

}

default:

{

puts("Invalid choice.");

break;

}

}

return 0;

}

int getDay(enum day day)

{

switch (day)

{

case Monday:

{

printf\_s("%s", "Day of the week: ");

printf("%s\n", "Monday");

break;

}

case Tuesday:

{

printf\_s("%s", "Day of the week: ");

printf("%s\n", "Tuesday");

break;

}

case Wednesday:

{

printf\_s("%s", "Day of the week: ");

printf("%s\n", "Wensday");

break;

}

case Thursday:

{

printf\_s("%s", "Day of the week: ");

printf("%s\n", "Thursday");

break;

}

case Friday:

{

printf\_s("%s", "Day of the week: ");

printf("%s\n", "Friday");

break;

}

case Saturday:

{

printf\_s("%s", "Day of the week: ");

printf("%s\n", "Saturday");

break;

}

case Sunday:

{

printf\_s("%s", "Day of the week: ");

printf("%s\n", "Sunday");

break;

}

default:

{

printf\_s("%s", "Invalid day of the week number.");

break;

}

}

}

int getMonth(enum month month)

{

switch (month)

{

case January:

{

printf\_s("%s", "Month: ");

printf("%s\n", "January");

break;

}

case February:

{

printf\_s("%s", "Month: ");

printf("%s\n", "February");

break;

}

case March:

{

printf\_s("%s", "Month: ");

printf("%s\n", "March");

break;

}

case April:

{

printf\_s("%s", "Month: ");

printf("%s\n", "April");

break;

}

case May:

{

printf\_s("%s", "Month: ");

printf("%s\n", "May");

break;

}

case June:

{

printf\_s("%s", "Month: ");

printf("%s\n", "June");

break;

}

case July:

{

printf\_s("%s", "Month: ");

printf("%s\n", "July");

break;

}

case August:

{

printf\_s("%s", "Month: ");

printf("%s\n", "August");

break;

}

case September:

{

printf\_s("%s", "Month: ");

printf("%s\n", "September");

break;

}

case October:

{

printf\_s("%s", "Month: ");

printf("%s\n", "October");

break;

}

case November:

{

printf\_s("%s", "Month: ");

printf("%s\n", "November");

break;

}

case December:

{

printf\_s("%s", "Month: ");

printf("%s\n", "December");

break;

}

default:

{

printf\_s("%s", "Invalid mounth number.");

break;

}

}

}

int get\_value()

{

int value;

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

if (result != 1)

{

errno = EIO;

perror("Invalid value entered!!!");

abort();

}

return value;

}

* 1. Решение тестовых примеров

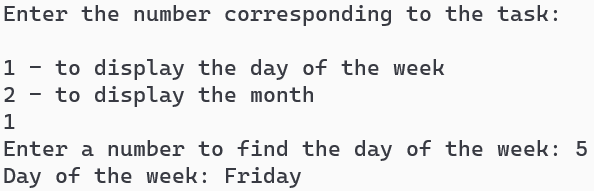


Рисунок 5 – Вывод названия дня недели

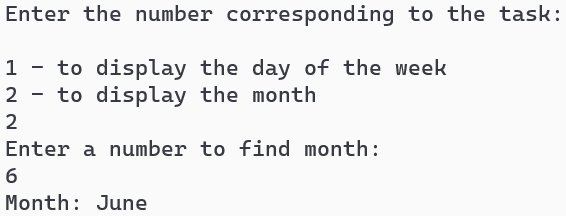


Рисунок 6 – Вывод названия месяца

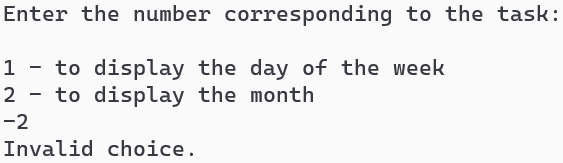


Рисунок 7 – Вывод, когда первый выбор не входит в указанные значения

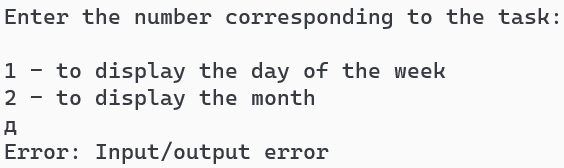


Рисунок 8 – Вывод, когда первый выбор – буква

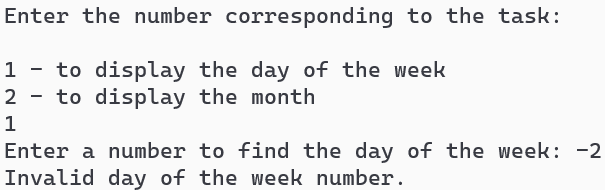


Рисунок 9 – Вывод, когда номер дня недели не входит в промежуток

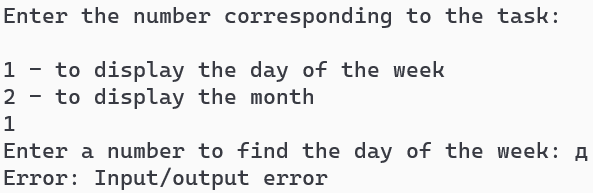


Рисунок 10 – Вывод, когда номер дня недели – буква

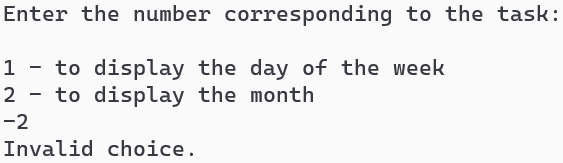


Рисунок 11 – Вывод, когда номер месяца не входит в промежуток

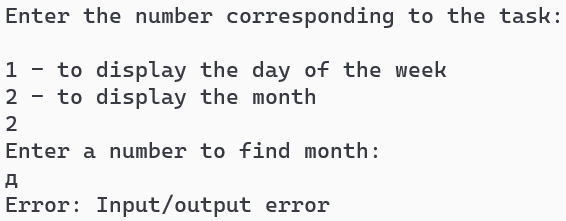


Рисунок 12 – Вывод, когда номер месяца – буква

* 1. Зачёт задания в GitHub

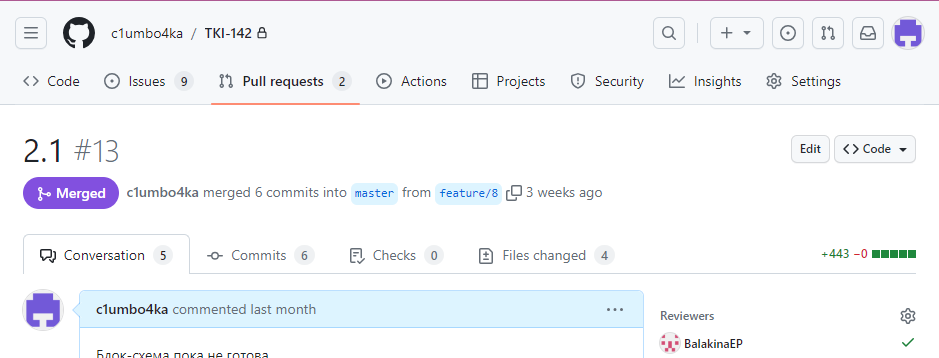


Рисунок 13 – Зачёт задания в GitHub

# Задание 2.2

* 1. Формулировка задания

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.

|  |  |  |
| --- | --- | --- |
| Вариант | Функция | Константы |
| 8 |  |  |

Таблица 2 – Формулировка задания 2.2

* 1. Блок-схема алгоритма

Блок-схема алгоритма представлена на рисунках(Рисунок 14, Рисунок 15, Рисунок 16)

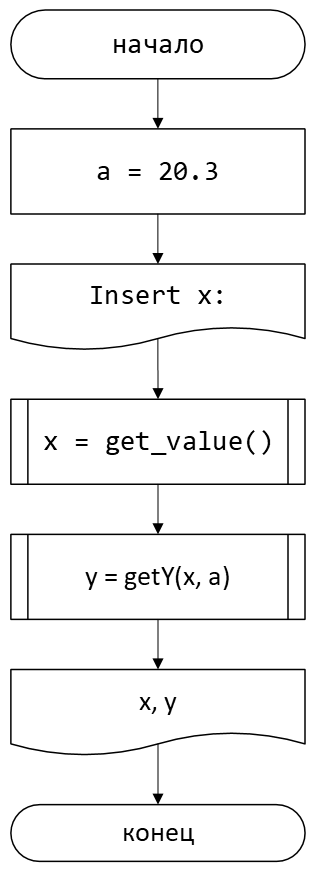


Рисунок 14 – Блок-схема функции main()

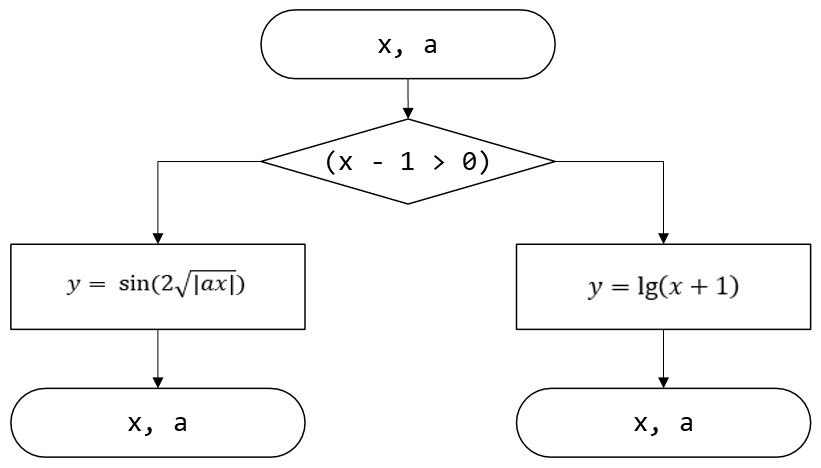


Рисунок 15 – Блок-схема функции getY(x, a)

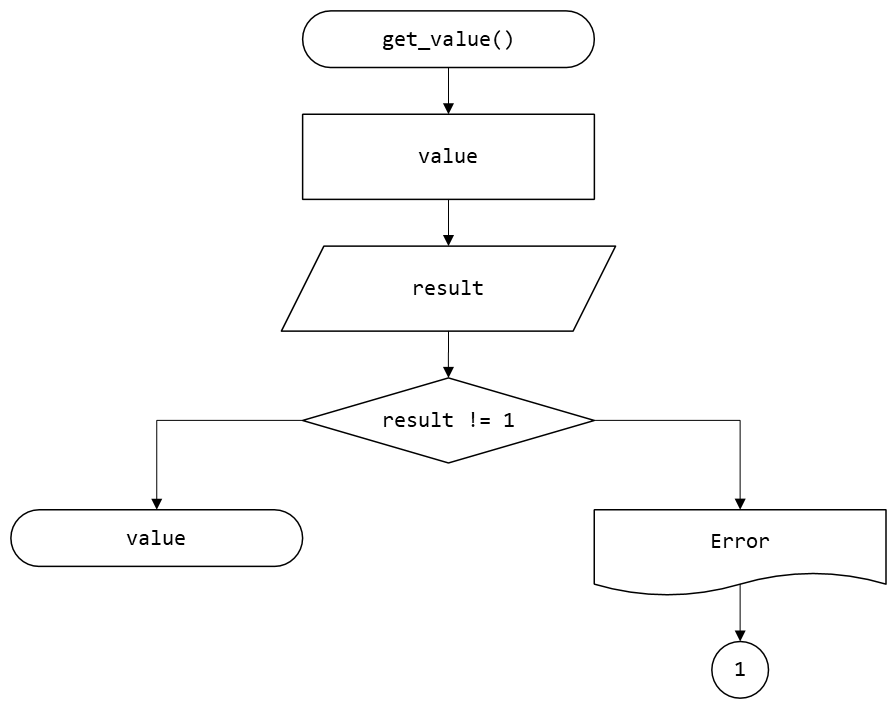


Рисунок 16 – Блок-схема функции get\_value()

* 1. Код задания 2.2

#include <stdio.h>

#include <math.h>

#include <float.h>

#include <errno.h>

#include <stdlib.h>

/\*\*

\* @brief Функция расчёта по заданной формуле.

\* @param x Аргумент функции.

\* @return Возвращает значение функции.

\*/

double getY(double x);

/\*\*

\* @brief Функция проверки ввода на правильность.

\* @return Возвращает значение в случае успеха.

\*/

double get\_value();

/\*\*

\* @brief Точка входа в программу.

\* @return Возвращает 0 в случае успеха.

\*/

int main()

{

double x = 0;

printf("%s", "Insert x: ");

double x = get\_value();

printf("x = %lf\n", x);

printf("y = %lf", getY(x));

return 0;

}

double getY(double x)

{

const double a = 20.3;

if (x - 1 > - DBL\_EPSILON)

{

return log10(x + 1);

}

return sin(2 \* sqrt(abs(a \* x)));

}

double get\_value()

{

double value;

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

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

{

errno = EIO;

perror("Invalid value entered!!!");

abort();

}

return value;

}

* 1. Решение тестовых примеров

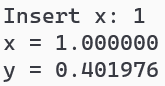


Рисунок 17 – Вывод, когда переменная x меньше либо равна 1

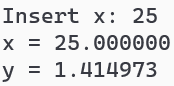


Рисунок 18 – Вывод, когда переменная x больше 1



Рисунок 19 – Вывод, когда переменная x – буква

* 1. Решение тестовых примеров в Excel

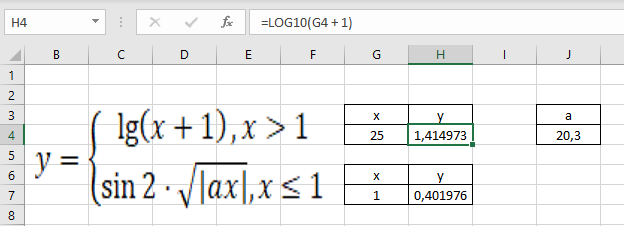


Рисунок 20 – Расчёт y, когда x, больше 1

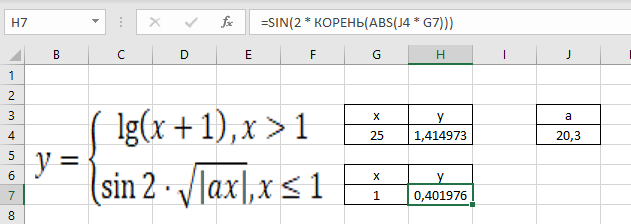


Рисунок 21 – Расчёт y, когда x меньше либо равен 1

* 1. Зачёт задания в GitHub

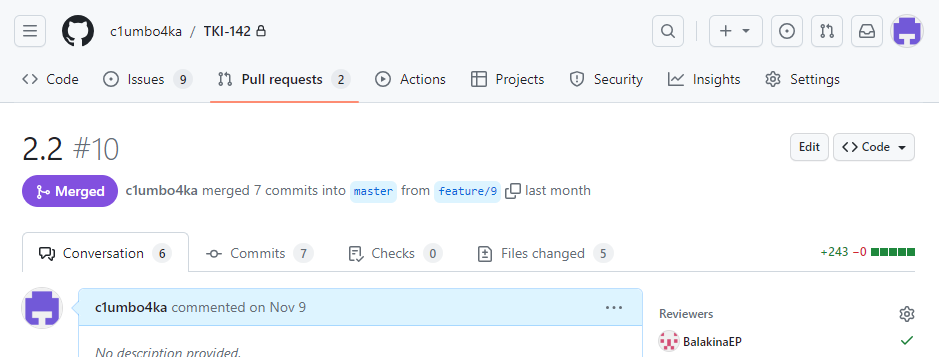


Рисунок 22 – Зачёт задания в GitHub

# Задание 2.1

* 1. Формулировка задания

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 3 –Формулировка задания 2.3

|  |  |
| --- | --- |
| Вариант | Задача |
| 8 | Записать программу, которая на название фигуры (треугольник, квадрат, ромб, прямоугольник и т.п.) выводит формулу, по которой вычисляется площадь этой фигуры. В программе использовать оператор CASE. |

* 1. Блок-схема алгоритма

Блок-схема алгоритма представлена на рисунках (Рисунок 23, Рисунок 24, Рисунок 25)

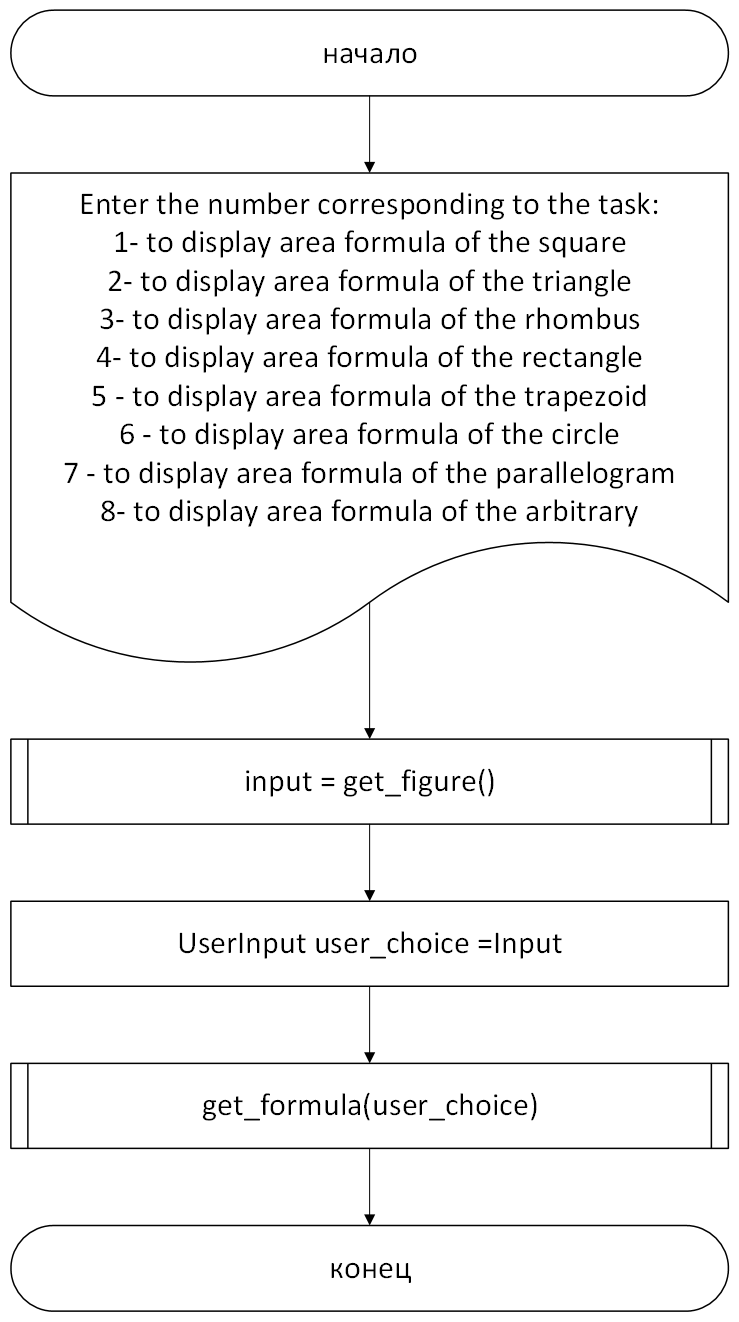


Рисунок 23 – Блок-схема к функции main()

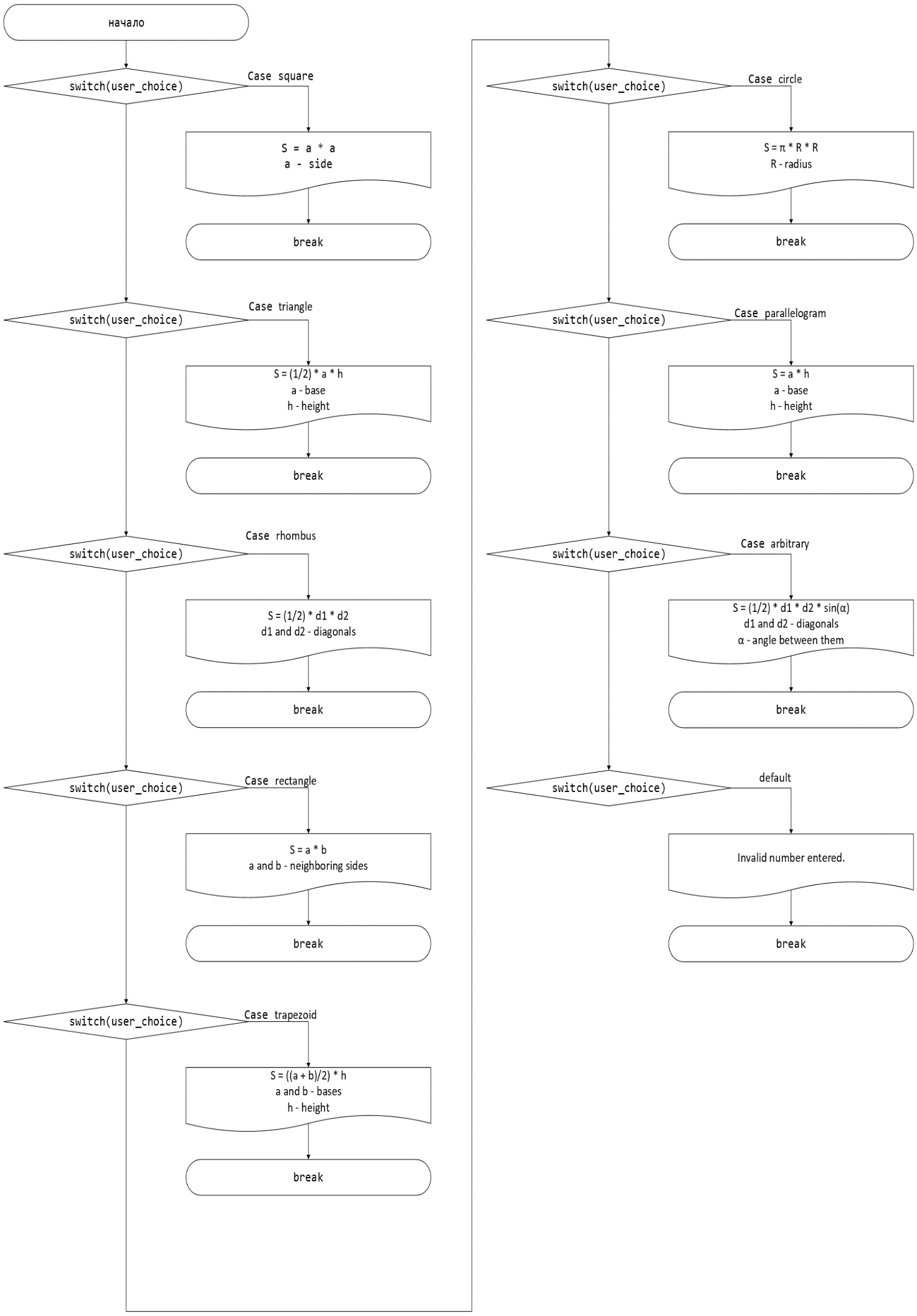


Рисунок 24 – блок-схема функции get\_formula(UserInput user\_choice)

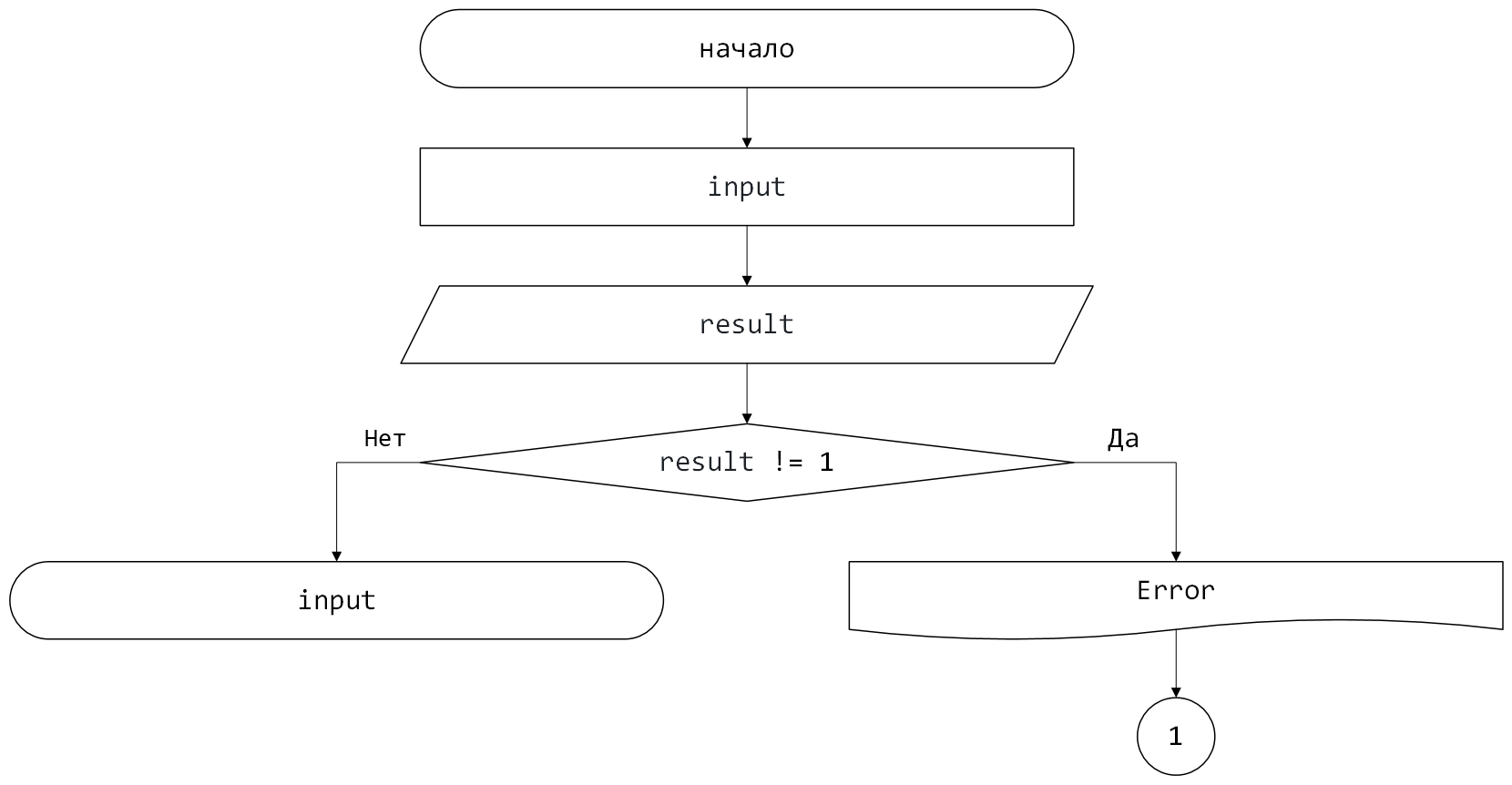


Рисунок 25 – Блок-схема функции get\_figure()

* 1. Код задания 2.3

#include <stdio.h>

#include <errno.h>

#include <stdlib.h>

/\*\*

\* @brief Функция проверки правильности ввода.

\* @return Возвращает значение в случае правильности ввода.

\*/

int get\_figure();

/\*\*

\* @brief Функция вывода формулы площади.

\* @param user\_choice Аргумент функции.

\*/

void get\_formula(enum UserInput user\_choice);

/\*\*

\* @brief Пользовательский ввод.

\*/

enum figure

{

/\*\*

@brief Формула площади квадрата.

\*/

square = 1,

/\*\*

@brief Формула площади треугольника.

\*/

triangle,

/\*\*

@brief Формула площади ромба.

\*/

rhombus,

/\*\*

@brief Формула площади прямоугольника.

\*/

rectangle,

/\*\*

@brief Формула площади трапеции

\*/

trapezoid,

/\*\*

@brief Формула площади круга.

\*/

circle,

/\*\*

@brief Формула площади параллелограмма.

\*/

parallelogram,

/\*\*

@brief Формула площади произвольного выпуклого многоугольника.

\*/

arbitrary

};

/\*\*

\* @brief Точка входа в программу.

\* @return Возвращает 0 в случае успеха.

\*/

int main()

{

puts("Enter the number corresponding to the task:\n");

printf\_s("%d - to display area formula of the square\n", square);

printf\_s("%d - to display area formula of the triangle\n", triangle);

printf\_s("%d - to display area formula of the rhombus\n", rhombus);

printf\_s("%d - to display area formula of the rectangle\n", rectangle);

printf\_s("%d - to display area formula of the trapezoid\n", trapezoid);

printf\_s("%d - to display area formula of the circle\n", circle);

printf\_s("%d - to display area formula of the parallelogram\n", parallelogram);

printf\_s("%d - to display area formula of the arbitrary\n", arbitrary);

char input = get\_figure();

enum UserInput user\_choice = (enum figure)input;

get\_formula(user\_choice);

return 0;

}

void get\_formula(enum UserInput user\_choice)

{

switch (user\_choice)

{

case square:

{

puts("S = a \* a");

puts("a - side");

break;

}

case triangle:

{

puts("S = (1/2) \* a \* h");

puts("a - base");

puts("h - height");

break;

}

case rhombus:

{

puts("S = (1/2) \* d1 \* d2");

puts("d1 and d2 - diagonals");

break;

}

case rectangle:

{

puts("S = a \* b");

puts("a and b - neighboring sides");

break;

}

case trapezoid:

{

puts("S = ((a + b)/2) \* h");

puts("a and b - bases");

puts("h - height");

break;

}

case circle:

{

puts("S = π \* R \* R");

puts("R - radius");

break;

}

case parallelogram:

{

puts("S = a \* h");

puts("a - base");

puts("h - height");

break;

}

case arbitrary:

{

puts("S = (1/2) \* d1 \* d2 \* sin(α)");

puts("d1 and d2 - diagonals");

puts("α - angle between them");

break;

}

default:

{

puts("Invalid number entered.");

break;

}

}

}

int get\_figure()

{

int input;

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

if (result != 1)

{

errno = EIO;

perror("Invalid figure entered!!!");

abort();

}

return input;

}

* 1. Решение тестового примера

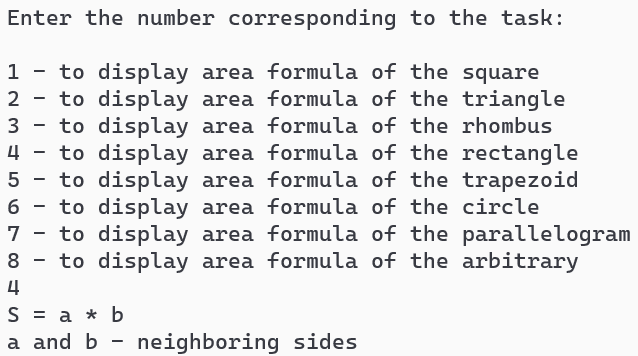


Рисунок 26 – Вывод формулы площади фигуры

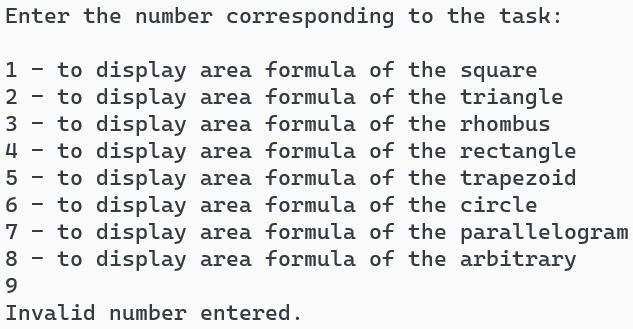


Рисунок 27 – Вывод программы, когда введённое значение не совпадает ни с одной из перечисленных

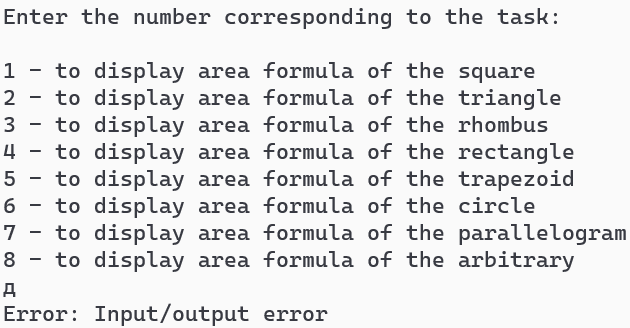


Рисунок 28 – Вывод программы, когда значение выбора – буква

* 1. Зачёт задания в GitHub

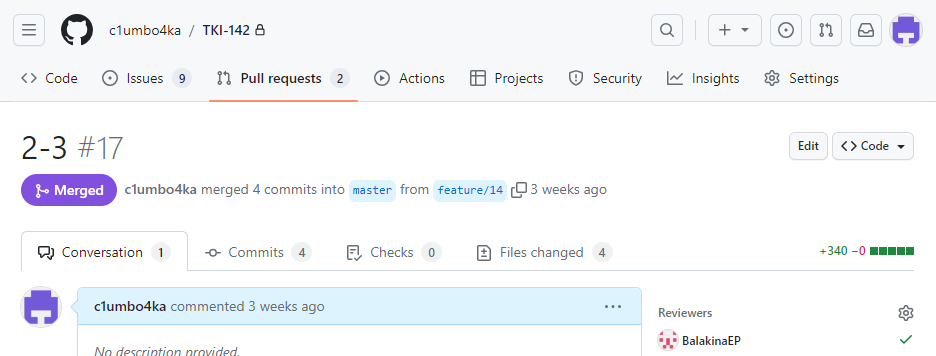


Рисунок 29 – Зачёт задания в GitHub