**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение

высшего образования

«Санкт-Петербургский политехнический университет Петра Великого»

(ФГАОУ ВО «СПбПУ»)

**Институт среднего профессионального образования**

**ОТЧЕТ**

**по учебной практике (по профилю специальности)**

по профессиональному модулю ПМ.04 «Сопровождение и обслуживание программного обеспечения компьютерных систем»

(код и наименование)

Специальность 09.02.07 «Информационные системы и программирование»

(код и наименование специальности)

Студент(ка) II курса 24290907/3091 группы

Поляков Роман Александрович

(ФИО полностью)

Место прохождения практики: ФГАОУ ВО СПбПУ Петра Великого   
Институт СПО, учебно-вычислительный центр, пр. Энгельса д.23

(наименование и адрес организации)

Период прохождения практики

с «08» декабря 2025 г. по «27» декабря 2025 г.

Руководитель практики Курылева А. А.

от учебной организации (подпись) (расшифровка подписи)

Итоговая оценка по практике \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

М.П.

Санкт-Петербург

2025г.

СОГЛАСОВАНО

Председатель ПЦК

\_\_\_\_\_\_\_\_/\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_ 202\_\_ г.

**ЗАДАНИЕ**

**на учебную практику (по профилю специальности)**

по профессиональному модулю ПМ.04 «Сопровождение и обслуживание программного обеспечения компьютерных систем»

(код и наименование)

Специальность 09.02.07 «Информационные системы и программирование»

(код и наименование специальности)

Студент(ка) II курса 24290907/3091 группы

Поляков Роман Александрович

(ФИО полностью)

Место прохождения практики: ФГАОУ ВО СПбПУ Петра Великого  
Институт СПО, учебно-вычислительный центр, пр. Энгельса д.23

(наименование и адрес организации)

Период прохождения практики

с «08» декабря 2025 г. по «27» декабря 2025 г.

**Виды работ, обязательные для выполнения** (переносится из программы, соответствующего ПМ):

1. Персонализация интегрированной среды разработки Visual Studio Community 2022
2. Отладка в IDE Visual Studio Community 2022
3. Обеспечение качества кода
4. Упаковка приложения

**Индивидуальное задание:**

Задание выдал с «8» декабря 2025 г. Курылева А. А. (подпись) (Ф.И.О.)

Задание получил с «8» декабря 2025 г.

(подпись) (Ф.И.О.)

**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение

высшего образования

«Санкт-Петербургский политехнический университет Петра Великого»

(ФГАОУ ВО «СПбПУ»)

**Институт среднего профессионального образования**

**ДНЕВНИК**

**прохождения учебной практики**

**(по профилю специальности)**

по профессиональному модулю ПМ.04 «Сопровождение и обслуживание программного обеспечения компьютерных систем»

(код и наименование)

Специальность 09.02.07 «Информационные системы и программирование»

(код и наименование специальности)

Студент(ка) II курса 24290907/3091 группы

Поляков Роман Александрович

(ФИО полностью)

Место прохождения практики: ФГАОУ ВО СПбПУ Петра Великого  
Институт СПО, учебно-вычислительный центр, пр. Энгельса д.23

(наименование и адрес организации)

Период прохождения практики

с «08» декабря 2025 г. по «27» декабря 2025 г.

Руководитель с места Курылева А. А.

прохождения практики (подпись) (расшифровка подписи)

Санкт-Петербург

2025 г.

**Содержание дневника**

|  |  |  |
| --- | --- | --- |
| Дата | Виды выполненных работ и заданий по программе практики | Подпись руководителя практики |
| **1** | **2** | **3** |
| 8.12.25 | Настройка меню и панели инструментов |  |
| 9.12.25 | Параметры текстового редактора |  |
| 10.12.25 | Создание кода и текстового шаблона |  |
| 11.12.25 | Навигация по коду с помощью отладчика |  |
| 12.12.25 | Использование точек останова |  |
| 13.12.25 | Управление исключениями с помощью отладчика |  |
| 15.12.25 | Использование файлов дампа  Использование средств профилирования |  |
| 16.12.25 | Тестирование в IDE Visual Studio Community 2022 |  |
| 17.12.25 | Документирование кода с помощью XML-комментариев |  |
| 18.12.25 | Изменение кода в соответствии с соглашением о кодировании |  |
| 19.12.25 | Анализ качества кода |  |
| 20.12.25 | Основы системы контроля версий Git |  |
| 22.12.25 | Технологические подходы программирования |  |
| 23.12.25 | Методология программирования |  |
| 24.12.25 | Работа с реестром ОС Windows |  |
| 25.12.25 | Упаковка классического приложения вручную |  |
| 26.12.25 | Упаковка приложения с помощью Visual Studio Package Installer |  |
| 27.12.25 | Создание виртуальной машины |  |

# **СОДЕРЖАНИЕ**

# **ВВЕДЕНИЕ**

Введение пишется ко всему отчету, не к каждой работе отдельно

## **ПРАКТИЧЕСКАЯ РАБОТА №1**

**«Настройка внешнего вида и функциональности среды разработки Visual Studio»**

**Цель:** освоить настройку интерфейса и функциональности visual studio для комфортной работы

2 вариант (четный номер в журнале) – Вам даны два целых числа.

Напишите программу, выводящую сумму целых чисел.

Код задачи

#include <iostream>

using namespace std;

int main()

{

int a, b{};

cin >> a; cin >> b;

cout << a + b; // Выводит сумму элементов

}

Запуски (Рисунок 1).

### **Параметры текстового редактора**

В настройках текстового редактора необходимо настроить автоматическое форматирование при вводе } и ;. Для этого необходимо в разделе «Средства → Параметры → Текстовый редактор» выбрать язык программирования, открыть общие настройки и установить флаги на соответствующих пунктах. Эти настройки позволят автоматически форматировать код во время ввода указанных символов (Рисунок 2, 3).

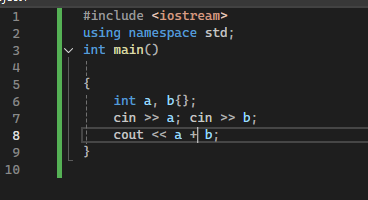


Рисунок 2 – Выключено автоматическое форматирование

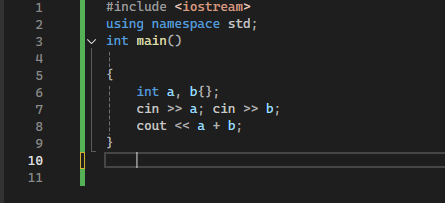


Рисунок 3 – Включено автоматическое форматирование

Для изменения интервалов в разделе «Средства → Параметры → Текстовый редактор» выбираем раздел «Форматирование» устанавливаем флаг на пункт «Вставлять пробел между скобками со списком аргументов» (Рисунок 4, 5).



Рисунок 4 – Выключено изменение интервалов



Рисунок 5 – Включено изменение интервалов

Для включения нумерации строк в разделе «Средства → Параметры → Текстовый редактор» выбираем раздел «C/C++ → Общие» устанавливаем флаг на пункт «Номера строк» (Рисунок 6, 7).

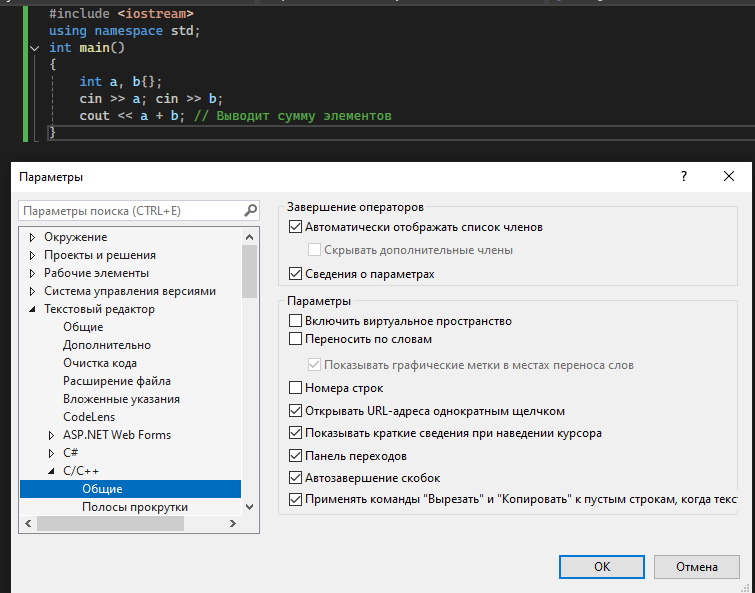


Рисунок 6 – Нумерация строк отсутствует

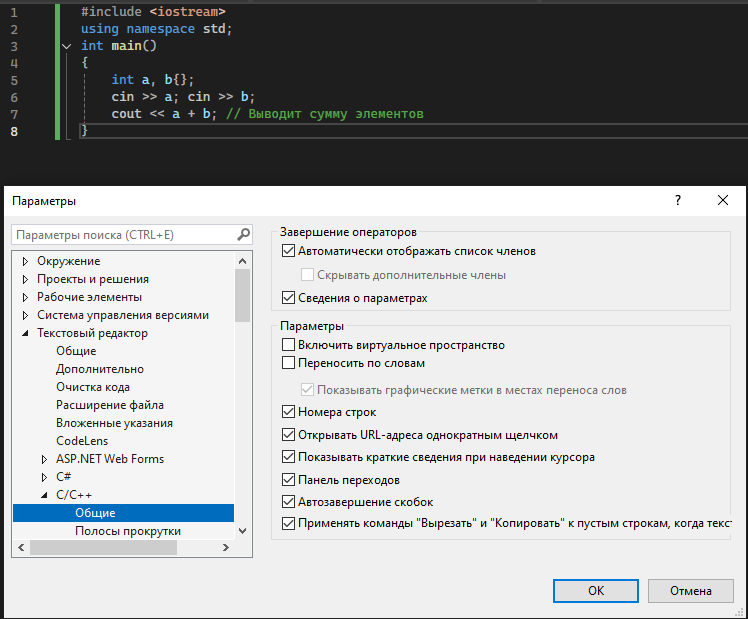


Рисунок 7 – Включение нумерации строк

### **Изменение шрифтов и цветов**

Для изменения шрифта и цвета всплывающих подсказок в разделе «Средства → Параметры → Окружение → Шрифты и цвета» в всплывающем окне выбрать нужный шрифт и цвет (Рисунок 8).

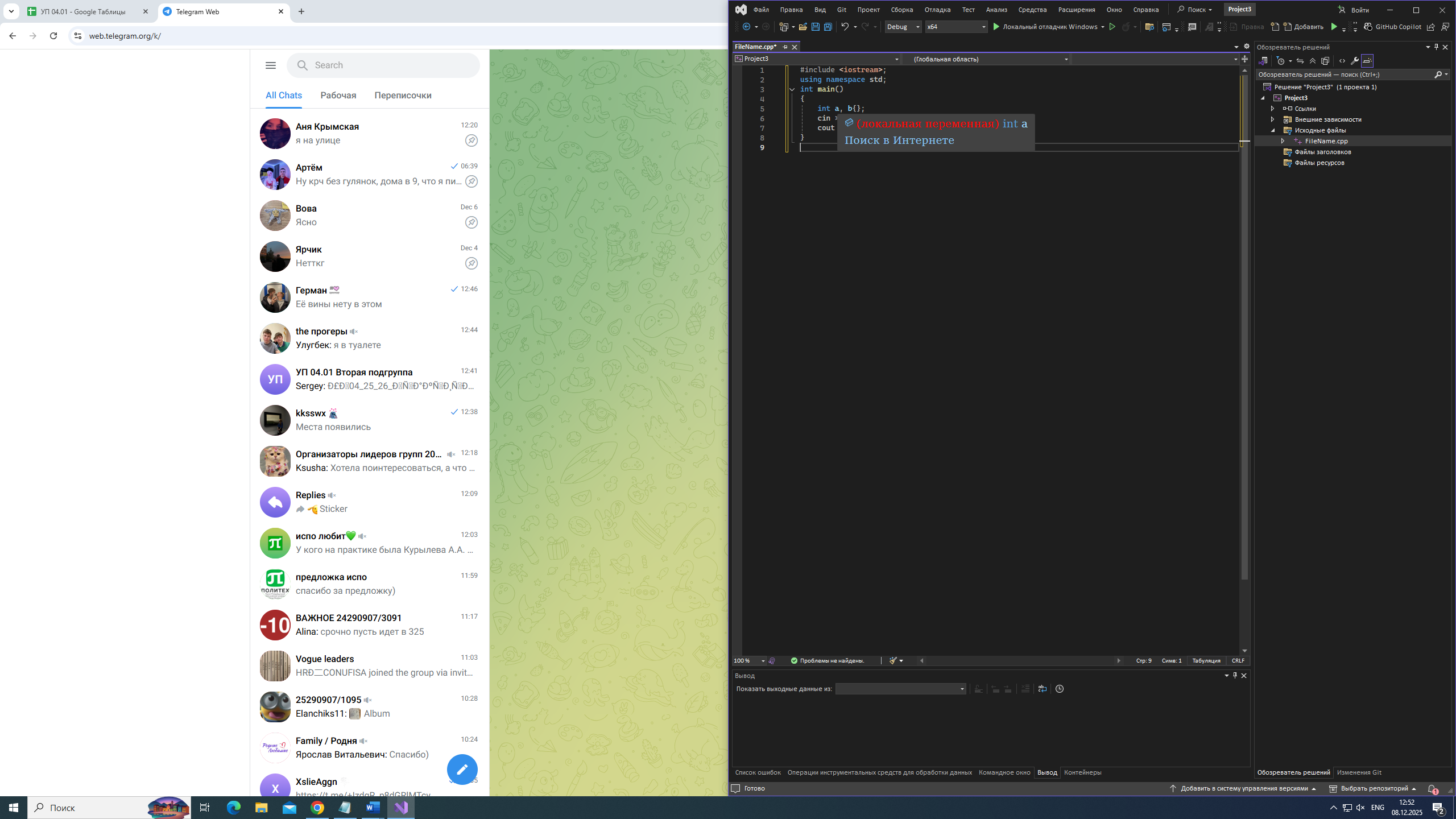


Рисунок 8 – Изменение шрифта и цвета всплывающих подсказок

Для изменения шрифта и цвета текста среды в разделе «Средства → Параметры → Окружение → Шрифты и цвета» в всплывающем окне выбрать нужный шрифт (Рисунок 9).

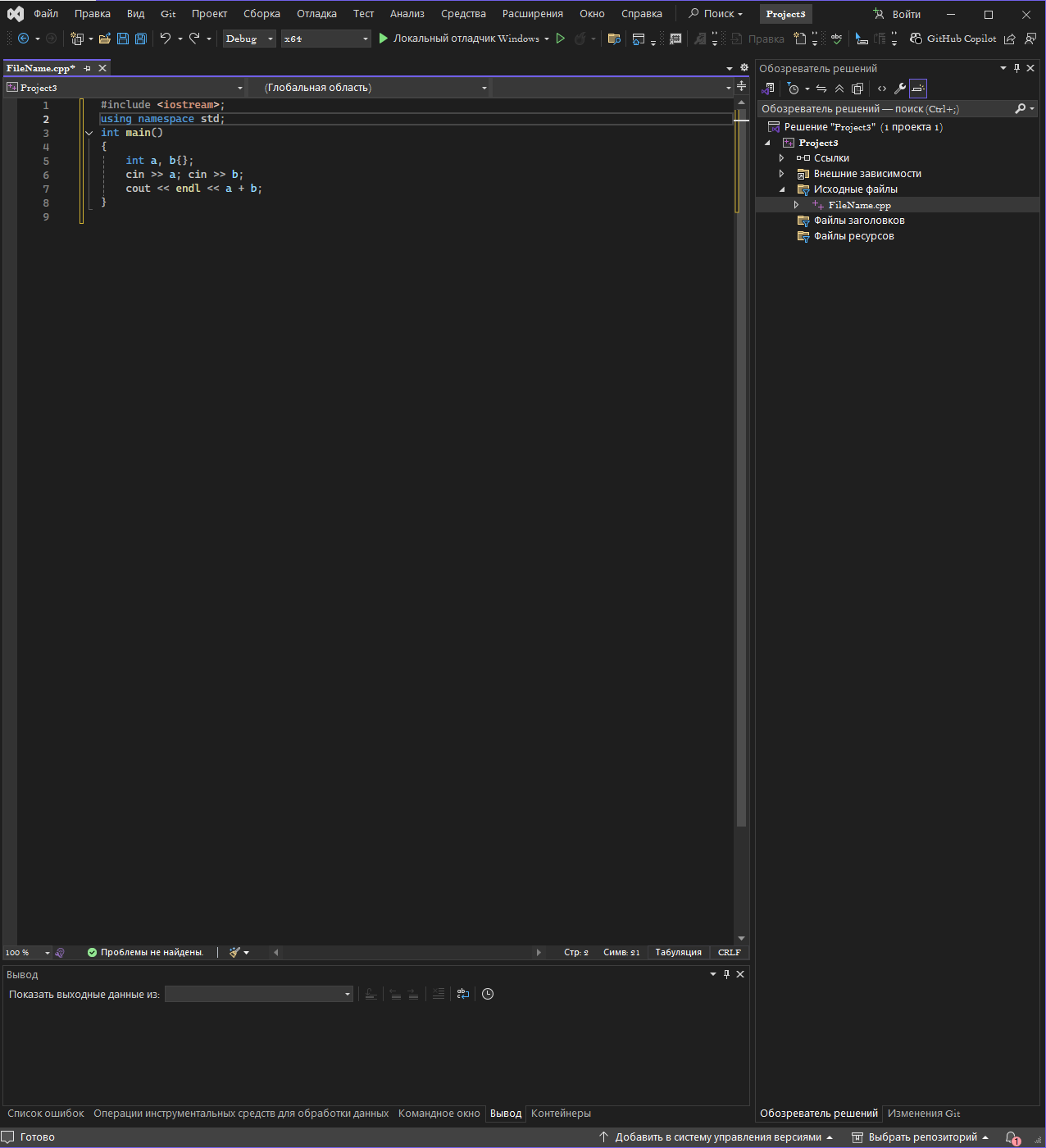


Рисунок 9 – изменение шрифта и цвета текста среды

Для изменения шрифта и цвета номера строки в разделе «Средства → Параметры → Окружение → Шрифты и цвета» в всплывающем окне выбрать нужный шрифт и цвет (Рисунок 10).

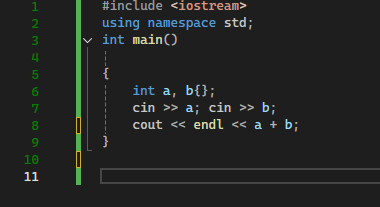


Рисунок 10 – Изменение шрифт и номера строк

Для изменения шрифта и цвета комментариев в разделе «Средства → Параметры → Окружение → Шрифты и цвета» в всплывающем окне выбираем нужный шрифт и цвет (Рисунок 11).

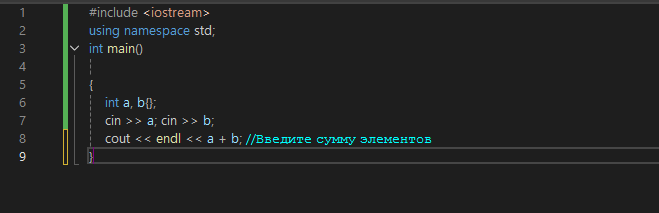


Рисунок 11 – Изменение шрифта и цвета комментария

Для изменения шрифта и цвета строковых литералов в разделе «Средства → Параметры → Окружение → Шрифты и цвета» в всплывающем окне выбираем нужный шрифт и цвет (Рисунок 12).

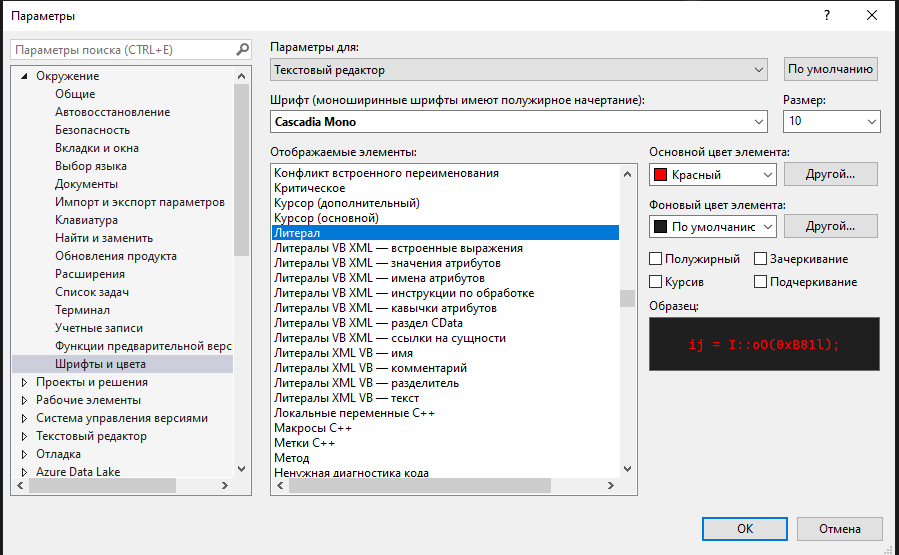


Рисунок 12 – Изменение шрифта и цвета строковых литералов

Для изменения шрифта и цвета чисел в разделе «Средства → Параметры → Окружение → Шрифты и цвета» в всплывающем окне выбираем нужный шрифт и цвет (Рисунок 13).

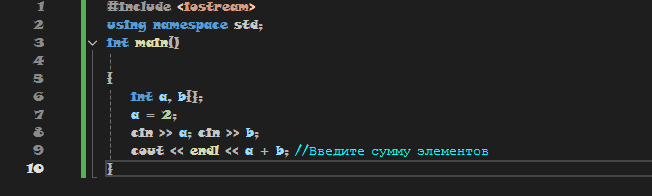


Рисунок 13 – изменение шрифта и цвета числа

Настройка меню и панели инструментов

Для создания своей панели инструментов «Средства → Настройка → Панель инструментов», добавить команды «Средства → Настройка → Команды» (Рисунок 14).

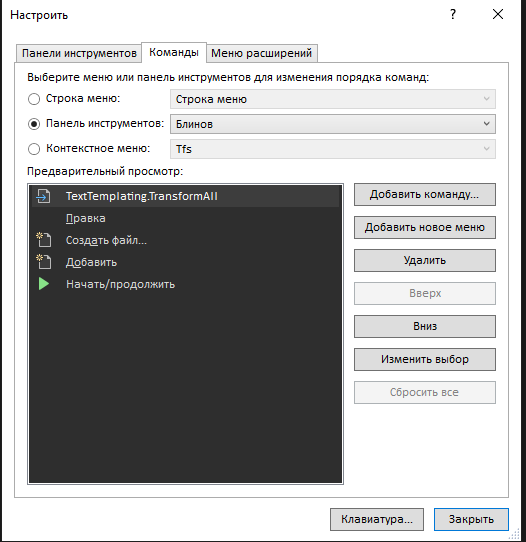


Рисунок 14 – Добавление команд в панель инструментов

Для назначения горячих клавиш через «Средства → Параметры → Окружение → Клавиатура» (Рисунок 15-19)

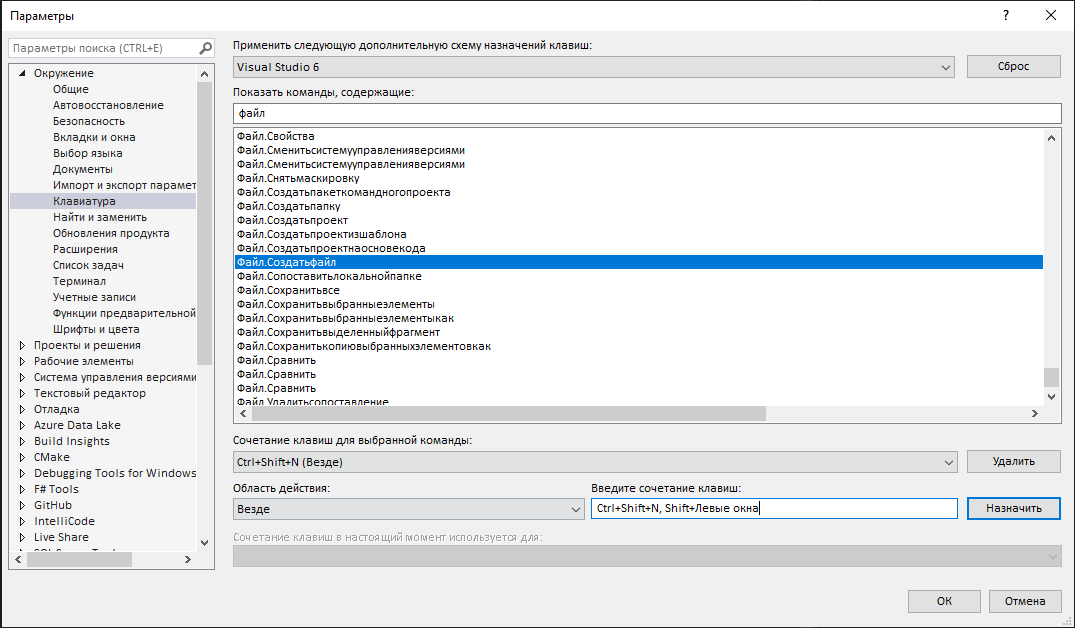


Рисунок 15 – Назначение горячей клавиши

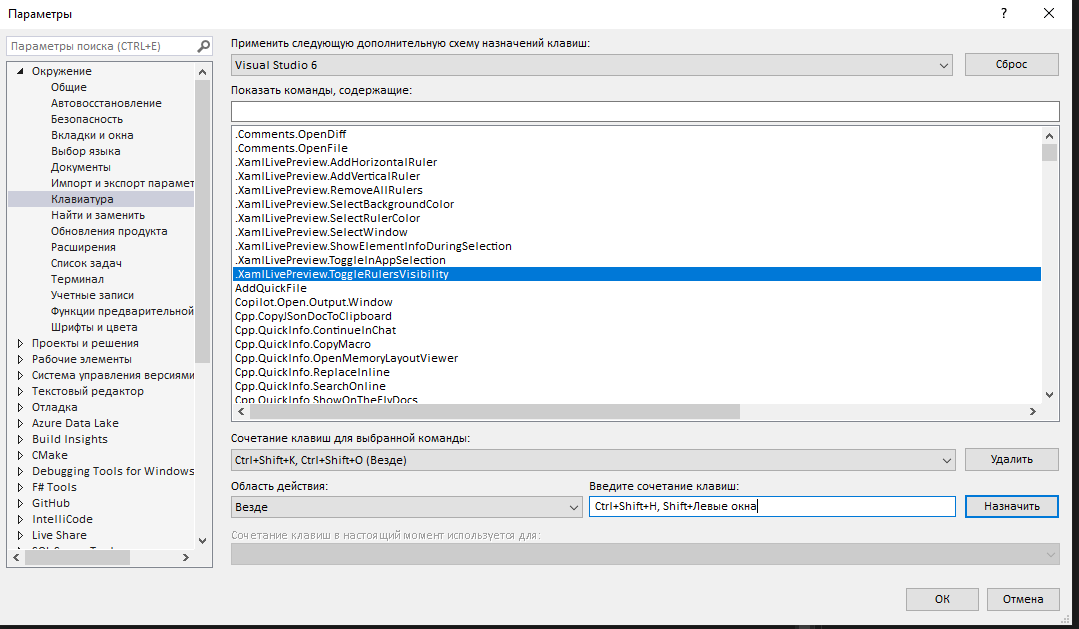


Рисунок 16 – Назначений горячей клавиши

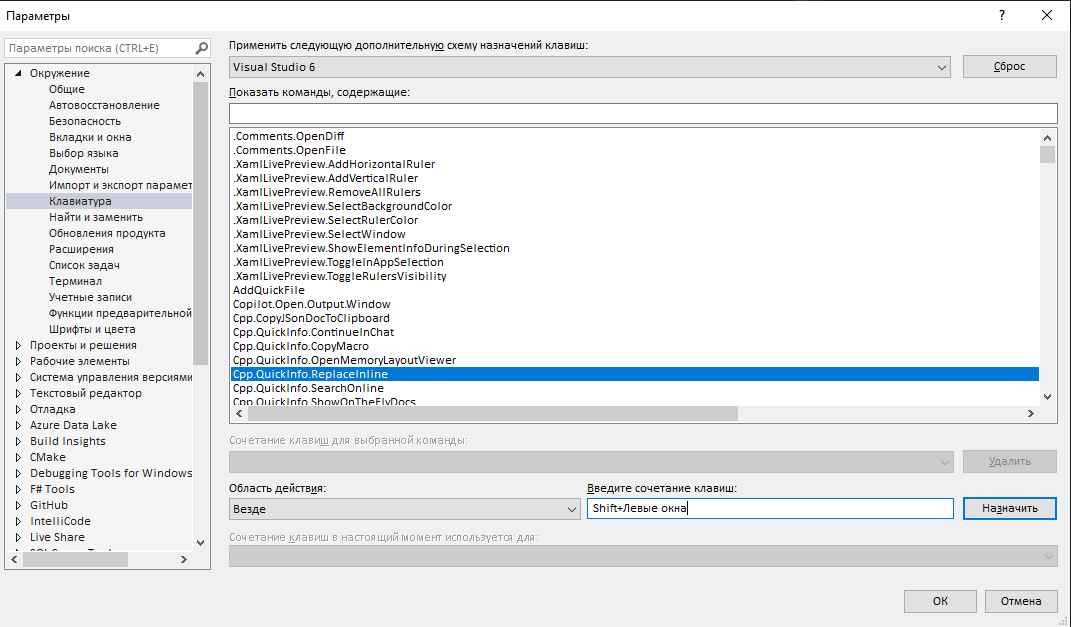


Рисунок 17 – Назначение горячей клавиши



Рисунок 18 – Назначение горячей клавиши

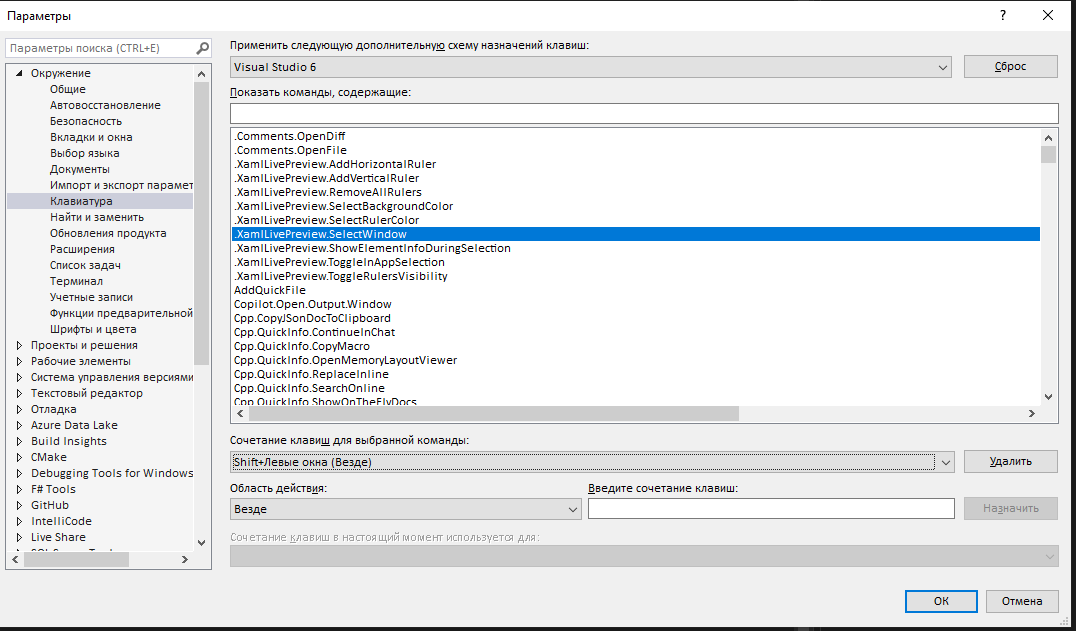


Рисунок 19 – Назначение горячей клавиши

### **Настройка макетов других окон**

Для добавления окон макета «Вид → Другие окна» (Рисунок 20).

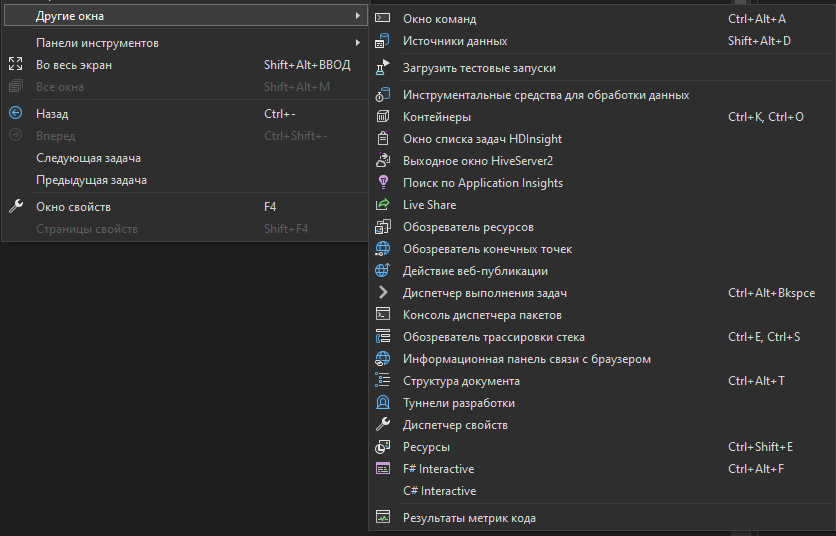


Рисунок 20 – Добавление окон в макет

Для сохранения макета «Окно → Сохранить макет окна» (Рисунок 21).

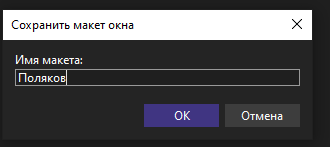


Рисунок 21 – Сохранение текущего макета кон

Переключение окон(Рисунок 22,23,24).

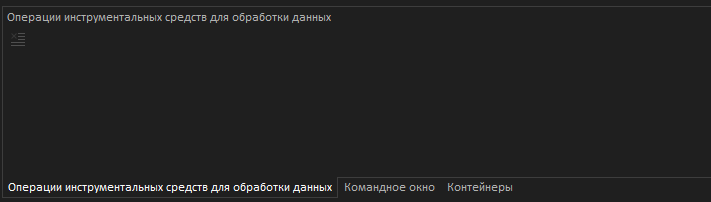


Рисунок 22 – 1 окно

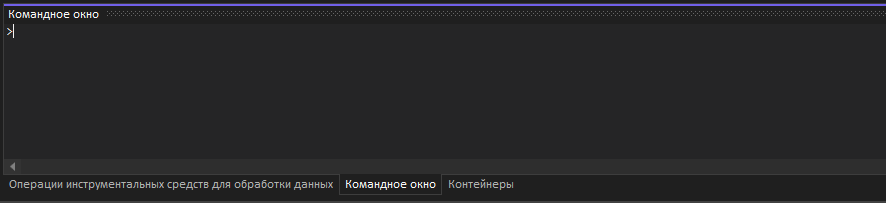


Рисунок 23 – 2 окно

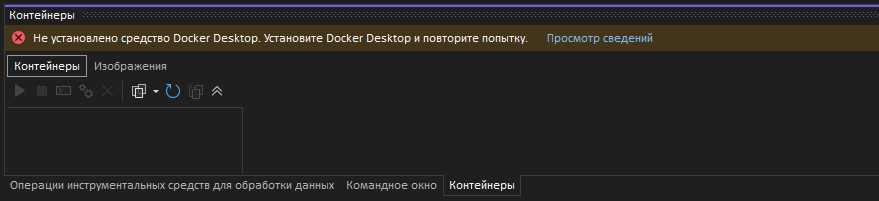


Рисунок 24 – 3 окно

### **Экспорт настроек**

Для сохранения настроек «Средства → Импорт и экспорт параметров → Экспорт» (Рисунок 25).

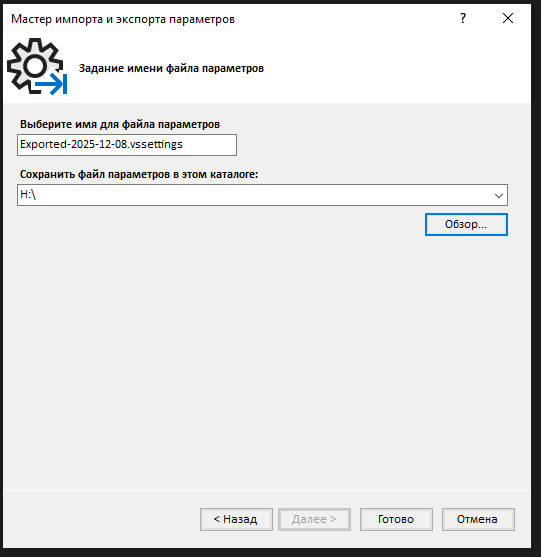


Рисунок 25 – Сохранение текущего макета

Для импорта сохранённого макета «Средства → Импорт и экспорт параметров → Импорт» (Рисунок 26 - 28).

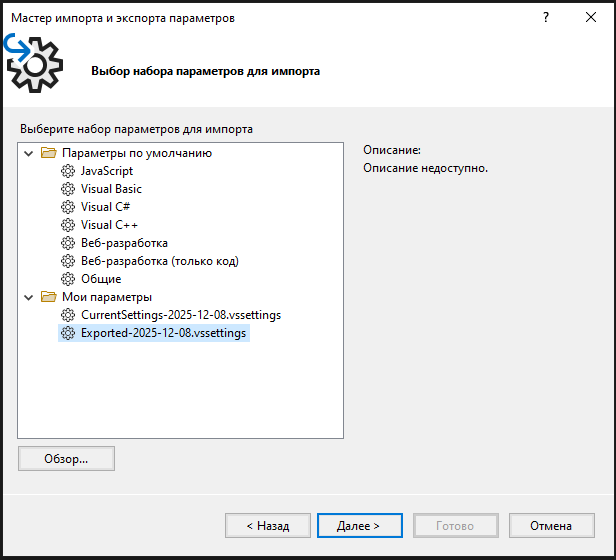


Рисунок 26 – Импорт сохраненного макета

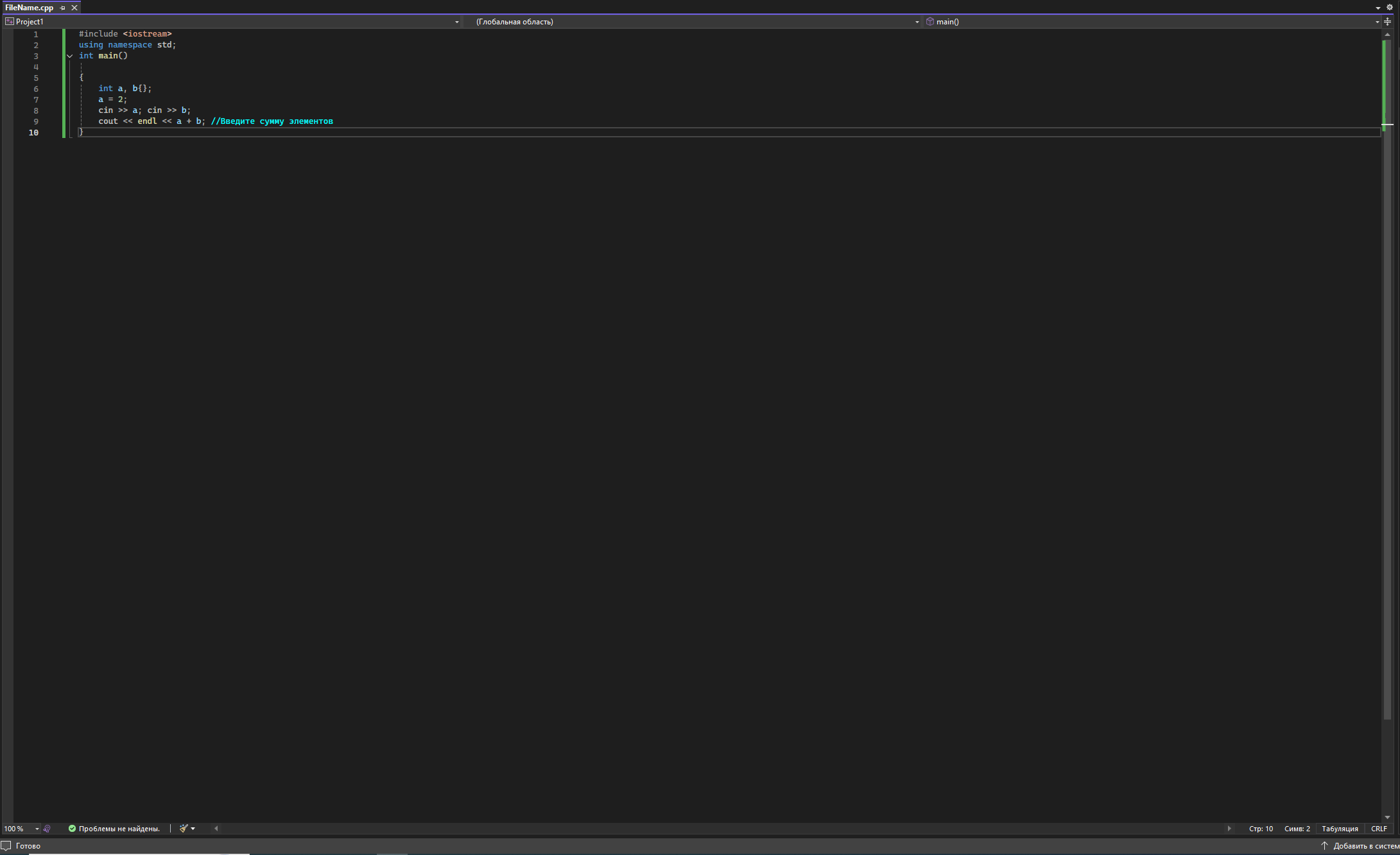


Рисунок 27 – До импорта

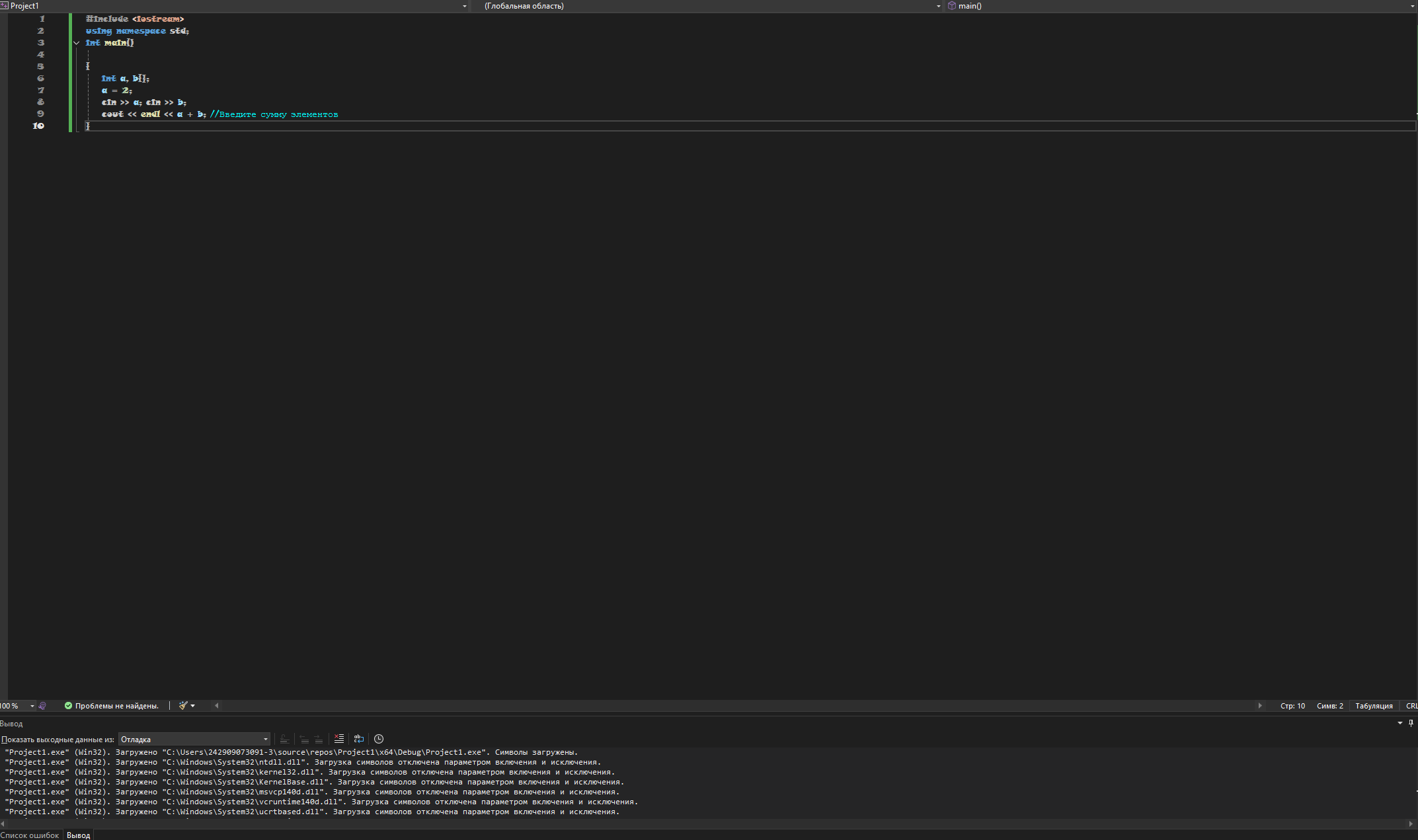


Рисунок 28 – после импорта

## **ПРАКТИЧЕСКАЯ РАБОТА №2 «ОТЛАДКА В VISUAL STUDIO»**

### **Задание 1:**

Создайте консольное приложение на C++, содержащее цикл(например, вывод чисел от 1 до 10) и рекурсивный метод.

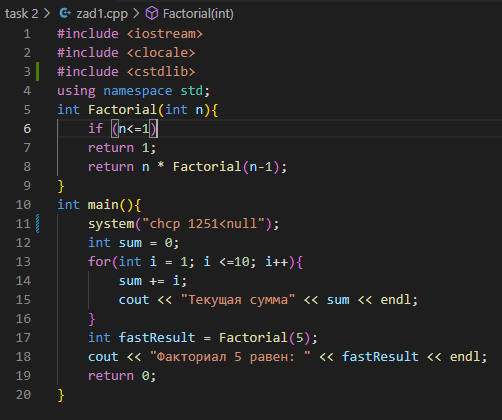


Рисунок 29 – Код

Установите точку останова на строке с циклом



Рисунок 30 – Установка точки на строке с циклом

Запустите откладку(F5):

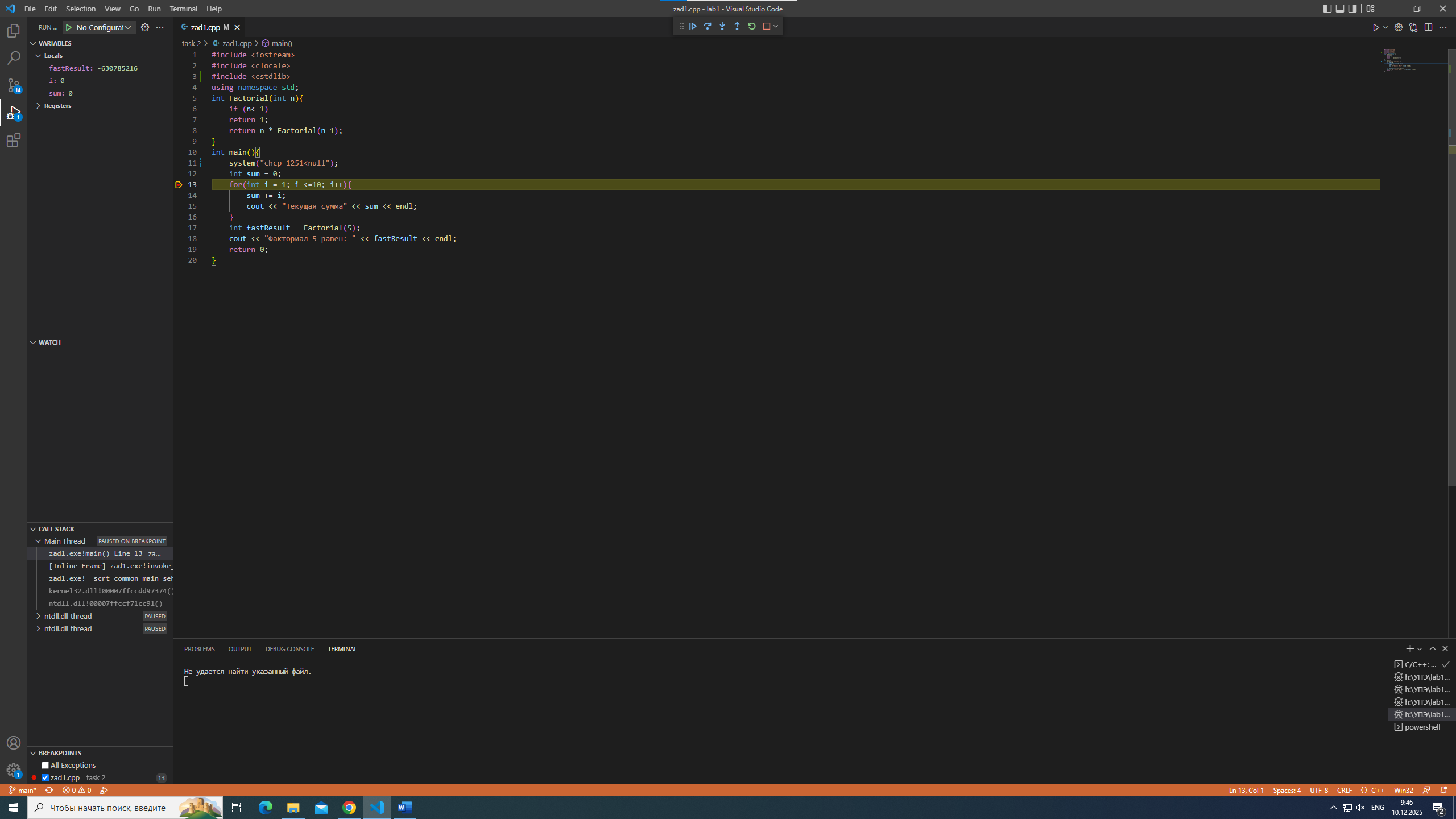


Рисунок 31 – Запуск отладки

Используйте шаги откладки(в отчете распишите про разницу следующих шагов):

Шаг с обходом(F10) – выполнить текущую строку.

Шаг с заходом(F11) – войти в метод.

Шаг с выходом(Shift + F11) – выйти из метода.

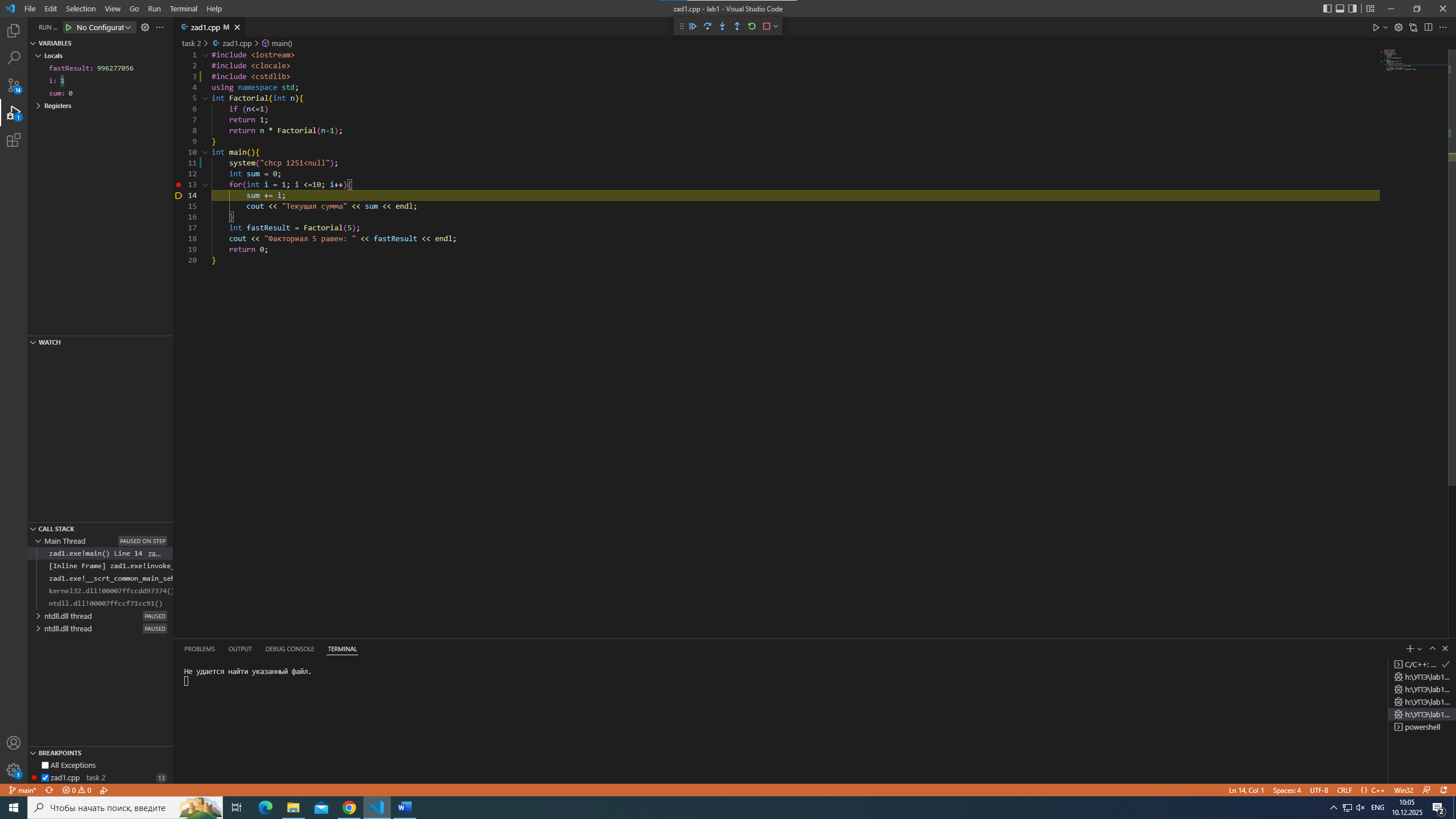


Рисунок 32 – Шаг с обходом (F10)

Шаг с обходом (F10) – выполнение текущей строки и переход на следующую строку, переменная I == 1.

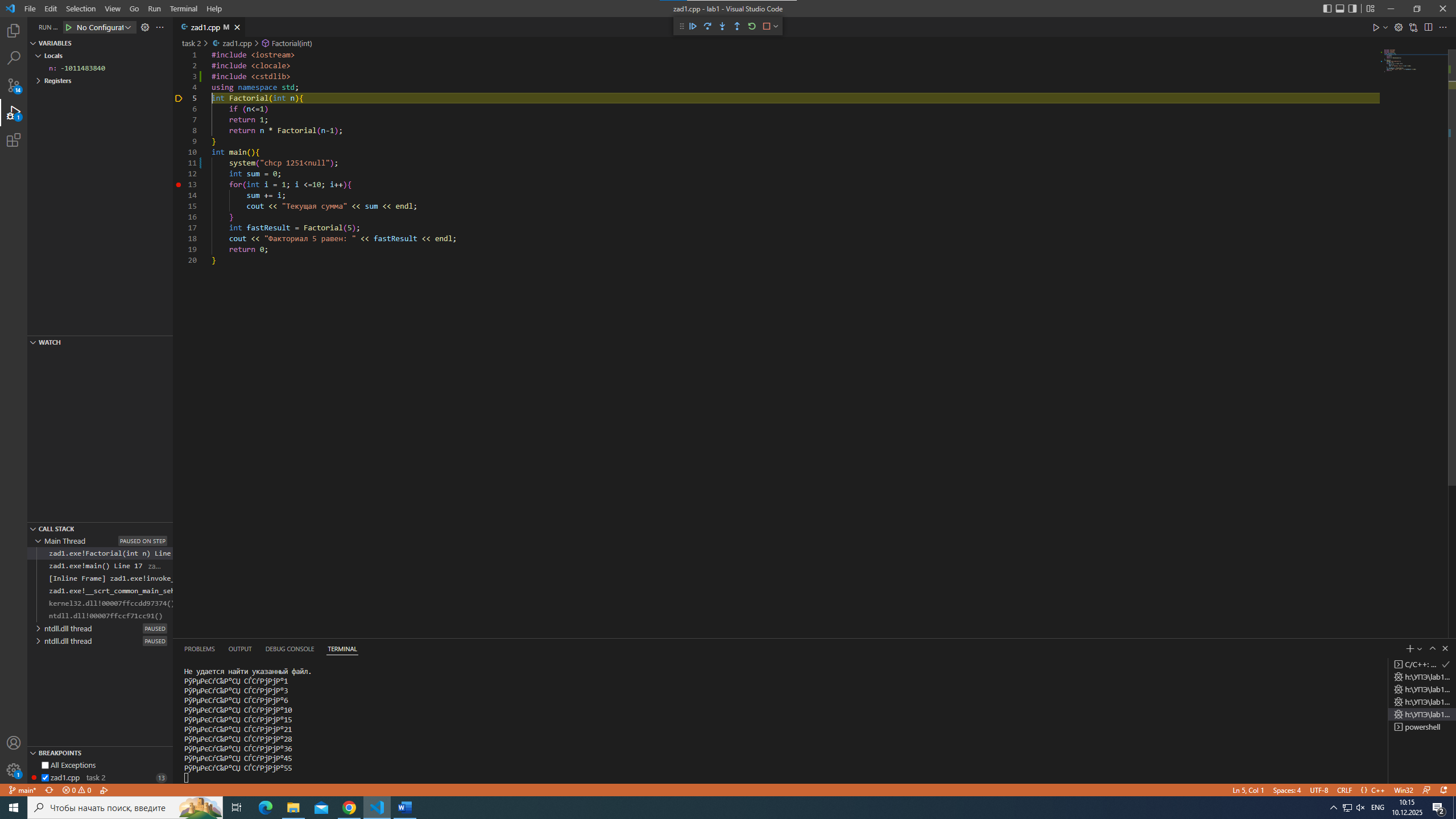


Рисунок 33 – Шаг с заходом (F11)

Шаг с заходом(F11) – при выполнении строки int fastResult = Factorial(5) переход внутрь функции Factorial, для того чтобы найти ошибку либо понять как работает данная функция.

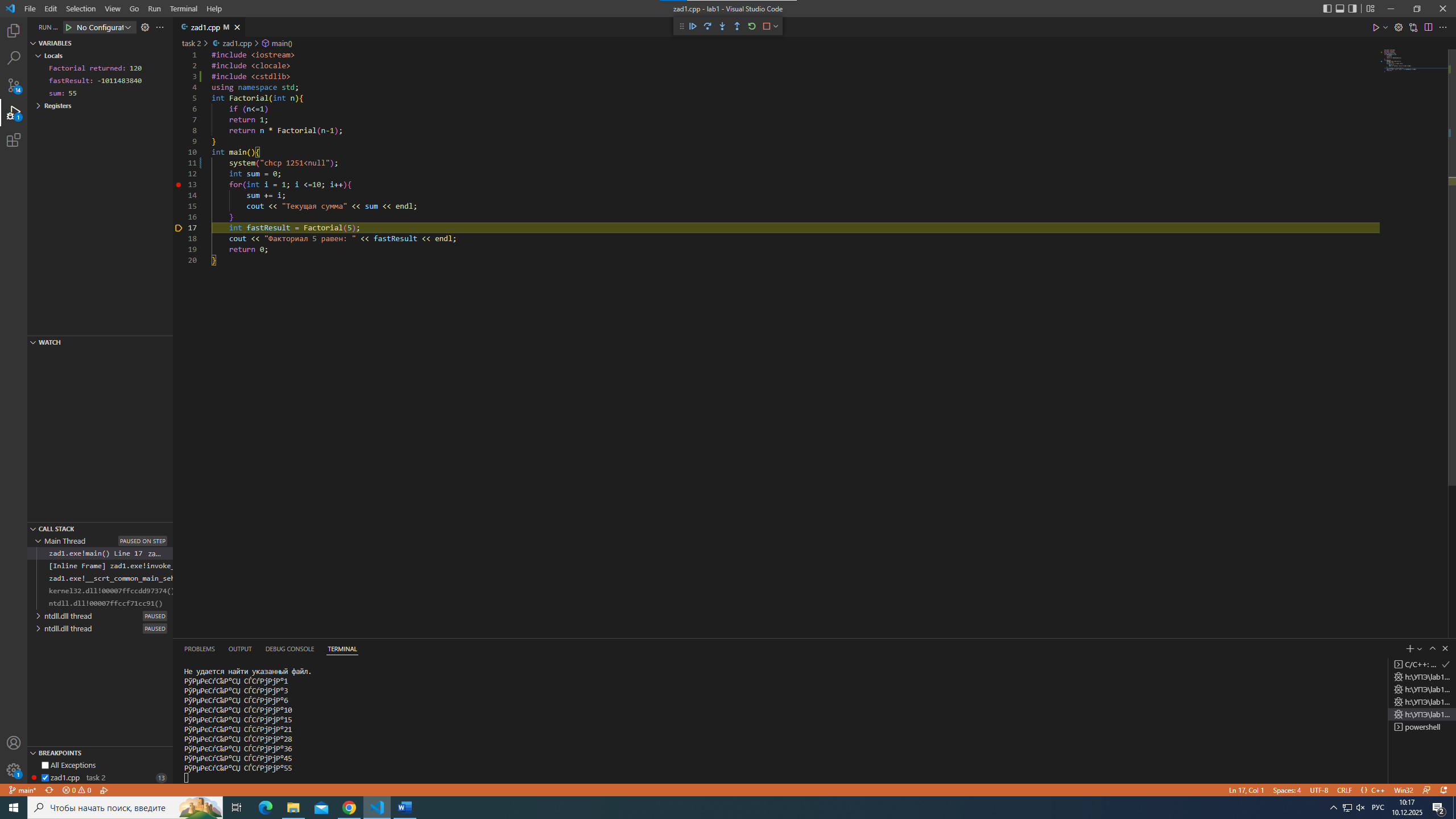


Рисунок 34 – Шаг с выходом(Shift + F11)

Шаг с выходом (Shift + F11) – переход со строки 5 в строку 17, отладчик выполнит остаток функции Factorial и вернет обратно в main.

Разница между шагами:

F10(Обход): Воспринимает функцию как одну простую инструкцию. Он не показывает, что внутри, а просто выполняет ее и переходит к следующей строке текущего кода.

F11(Заход): Воспринимает функцию как портал. Он открывает ее код и ставит курсор на самую первую строку внутри этой функции.

Shift + F11(Выход): Воспринимает текущую функцию как “пройденный этап”. Он выполняет весь оставшийся код этой функции без остановок и возвращает курсор, туда, где эта функция была вызвана.

Установите условную точку останова, например, для остановки при значении переменной I == 5.

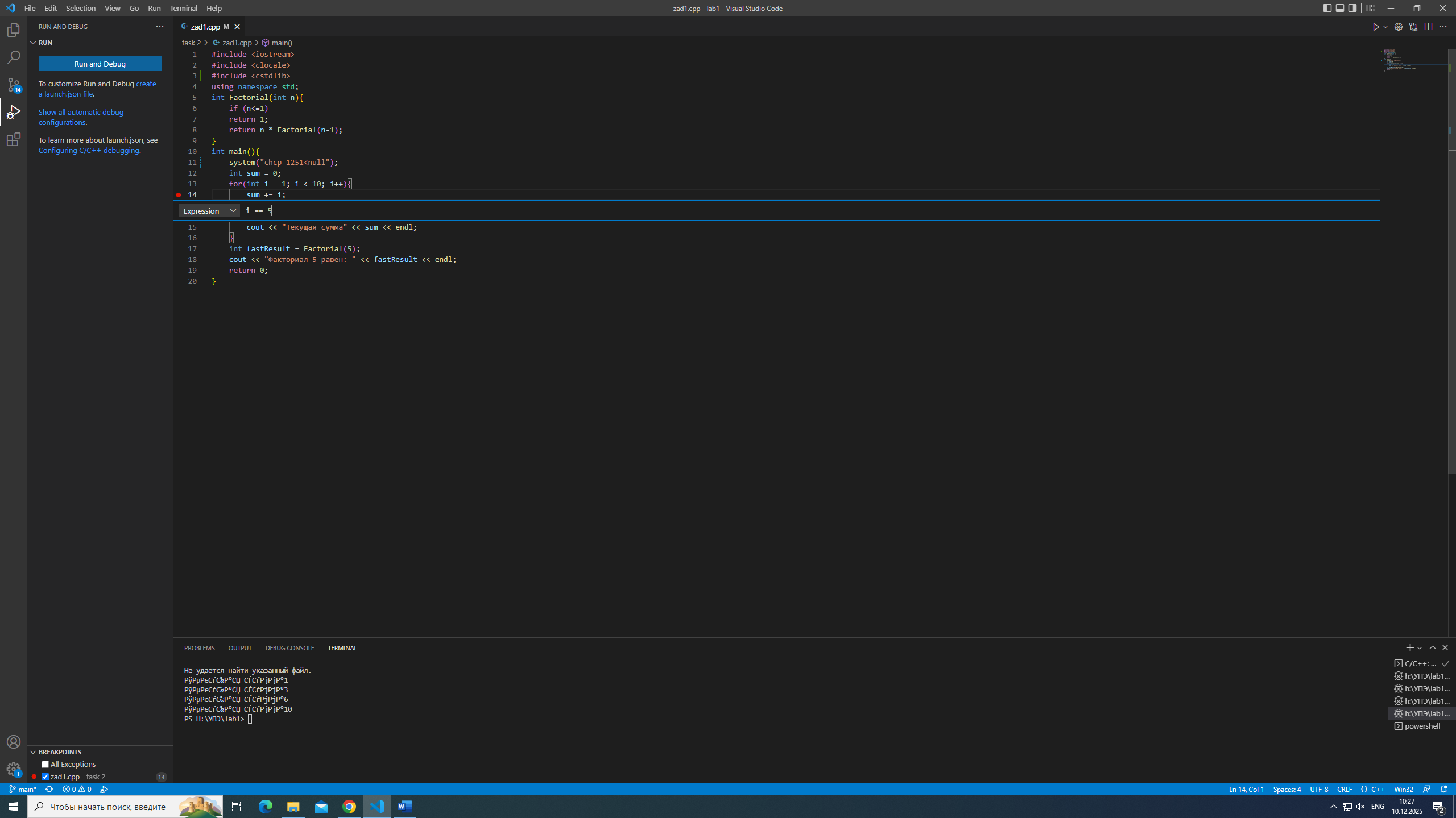


Рисунок 35 – I == 5

Наблюдайте за значениями переменных в окнах(в отчете распишите про отличия следующих окон):

Видимые и локальные окна.

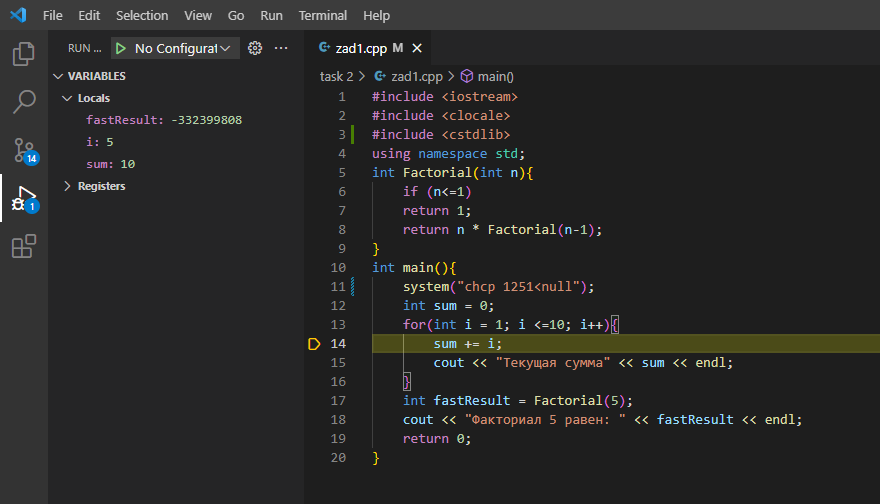


Рисунок 36 – Видимые и локальные окна

Контрольные значения

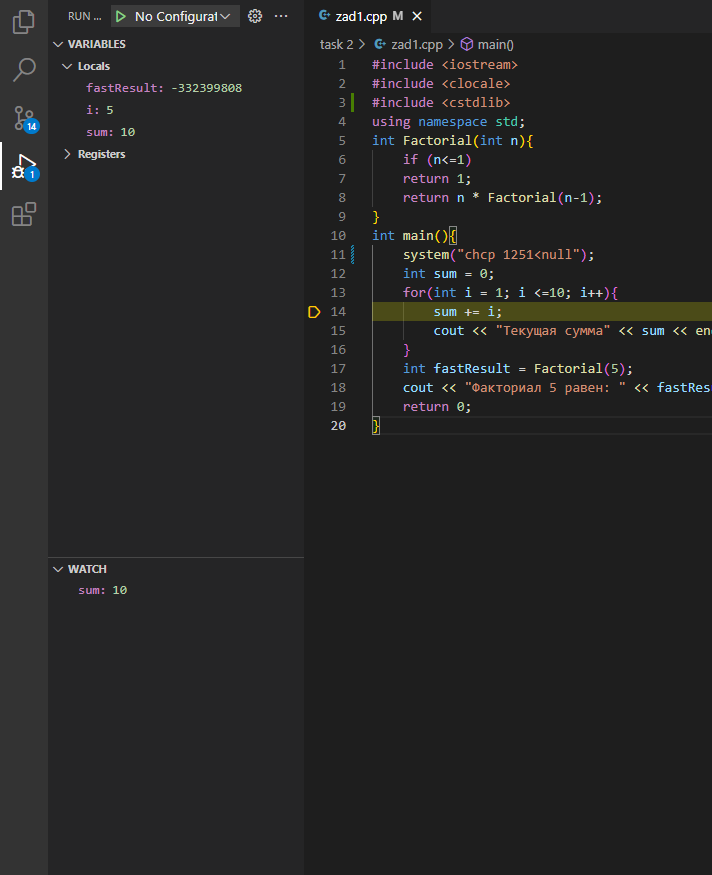


Рисунок 37 – Контрольные значения

QuickWatch.



Рисунок 38 - QuickWatch

Различие окон:

Видимые и локальные

Локальные: Показывают абсолютно все переменные, которые существуют внутри текущей функции

Видимые: только те переменные, которые участвуют в текущей строке кода.

Контрольные значения: показывает только то, что вы сами туда добавили

Быстрая проверка: всплывающее окно на один раз. Открыли посмотрели значение и закрыли.

### **Задание 2**

Добавьте код, генерирующий исключение деление на ноль или выход за границы массива.

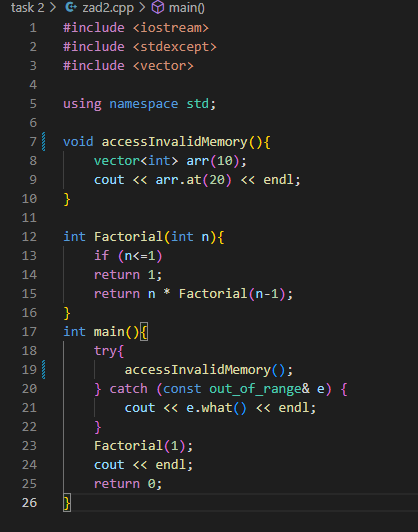


Рисунок 39 - Код

Настройте отладчик для прерывания при определенных исключениях.



Рисунок 40 – отладчик для прерывания

Запустите отладку и продемонстрируйте, как отладчик останавливается при возникновении исключения.

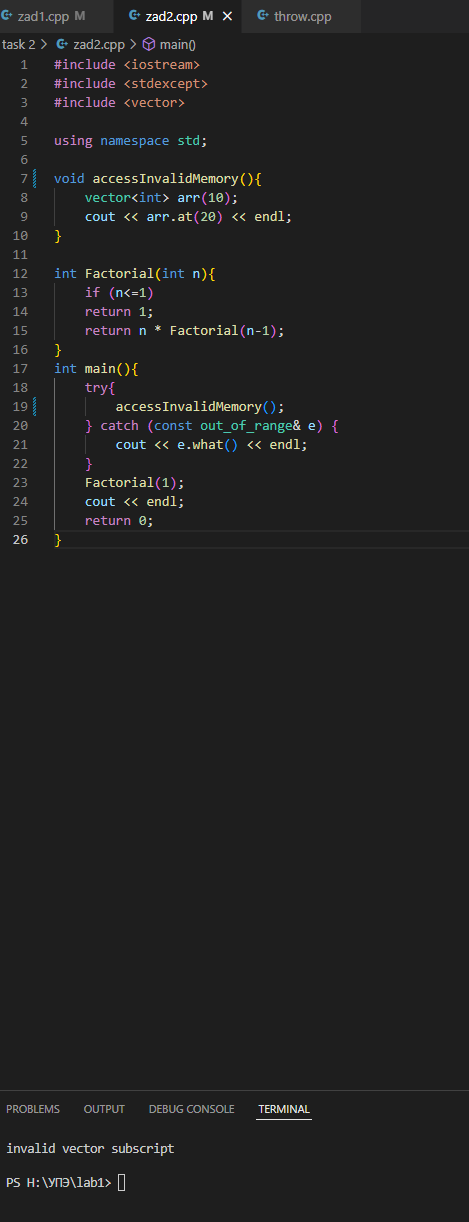


Рисунок 41 – остановка отладчика

### **Задание 3**

Добавьте код с интенсивными вычислениями(например, цикл с большим количеством итераций или работа с коллекциями).



Рисунок 42 - Код

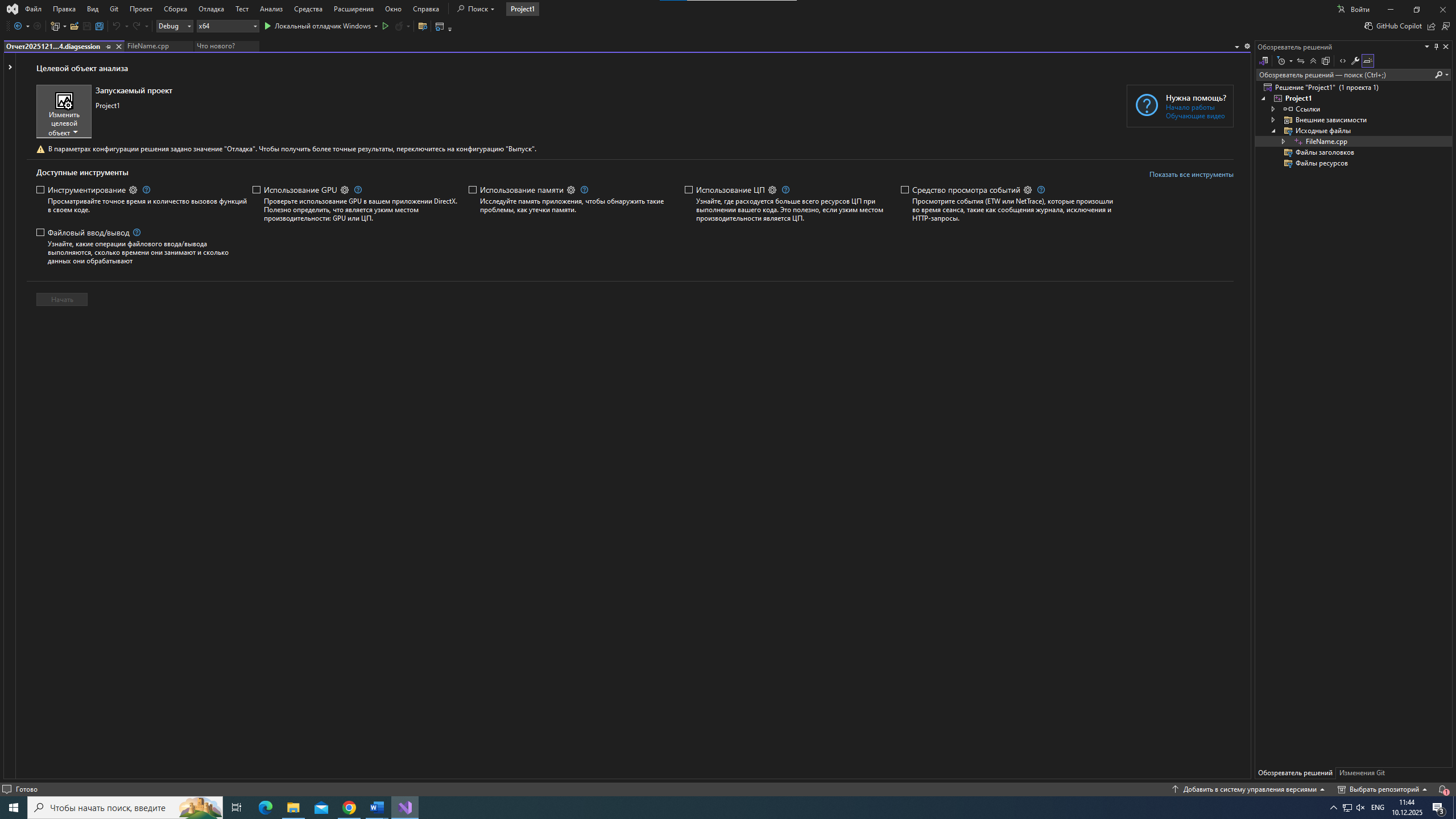


Рисунок 43 - отладка профилировщик производительности

Запустит анализ Использование ЦП для определения нагрузочных функций.

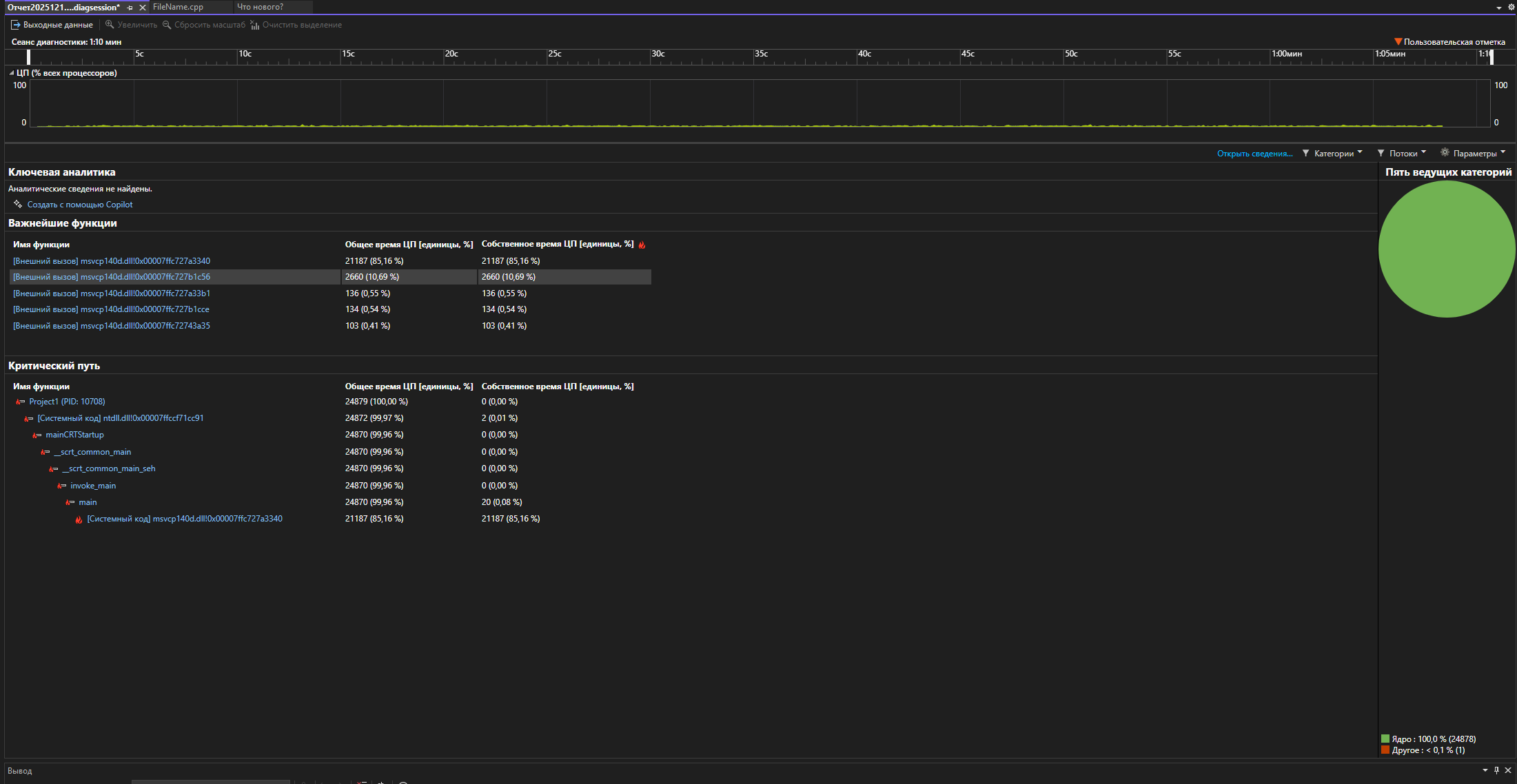


Рисунок 44 – Анализ ЦП

Запустите анализ Использование памяти для отслеживания выделения памяти и утечек.

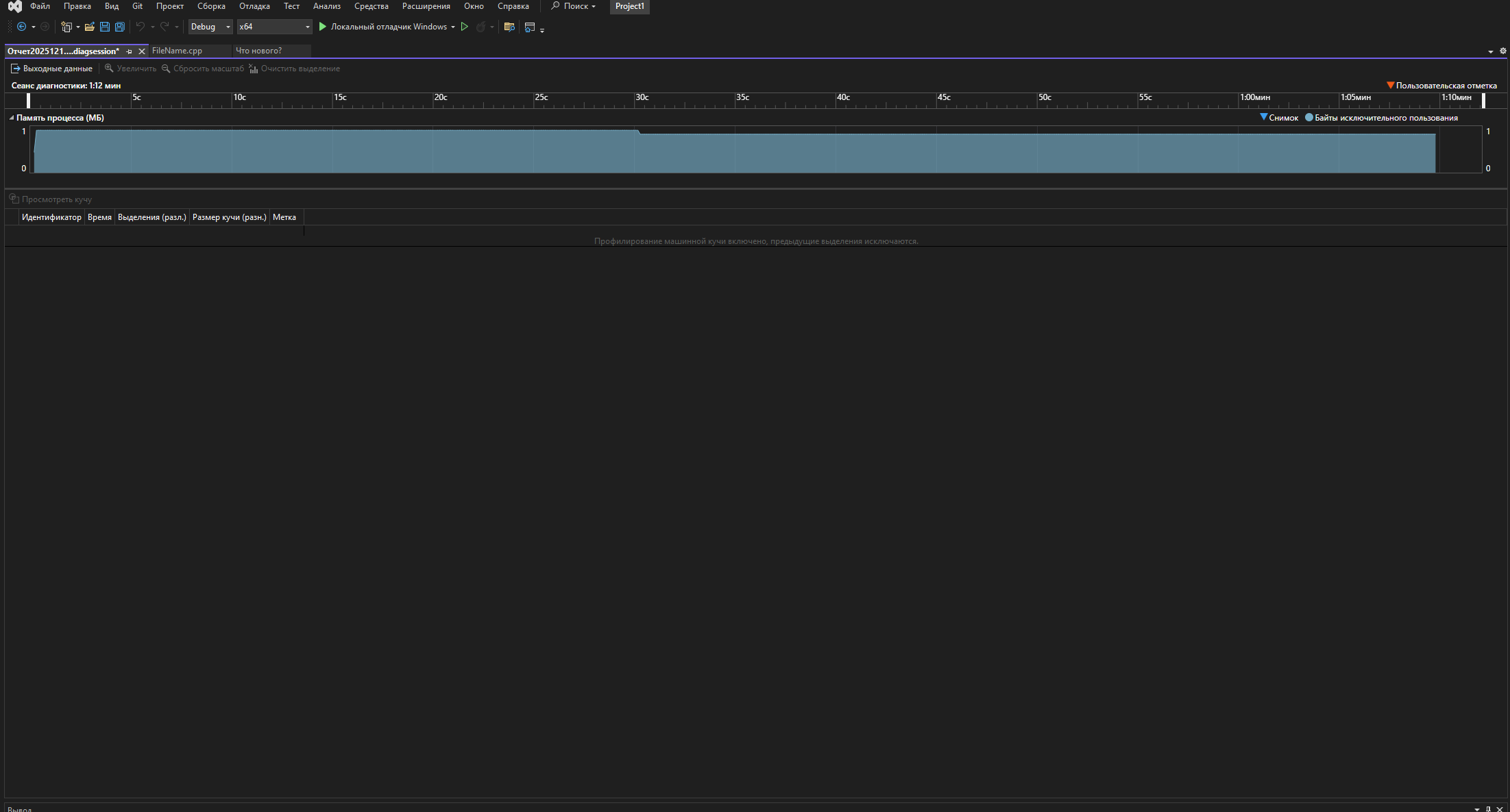


Рисунок 45 – Анализ памяти

## **ПРАКТИЧЕСКАЯ РАБОТА №3 «ОБЕСПЕЧЕНИЕ КАЧЕСТВА КОДА НА C++»**

Цель: научиться писать чистый, понятный и хорошо документированный код.

### **Задание 1**

Создайте новый консольный проект C++

Напишите свою функцию п варианту – «Вариант 1. Напишите функцию расчета площади прямоугольника. Функция принимает длину и ширину, возвращает площадь. Вызовите функцию с разными параметрами в main.»

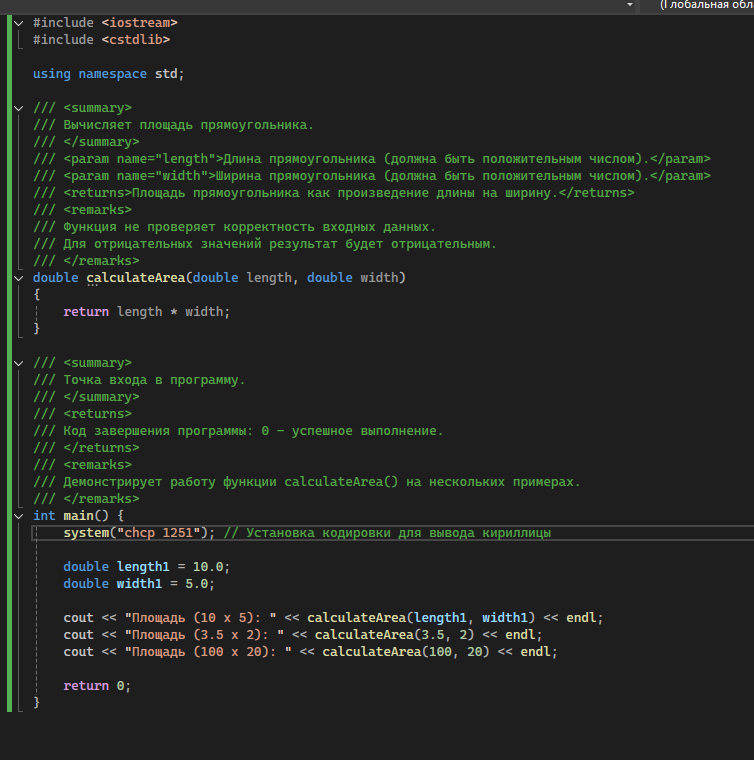


Рисунок 46 - Код

Покажите, как в Visual Studio появляется подсказка с вашим комментарием.

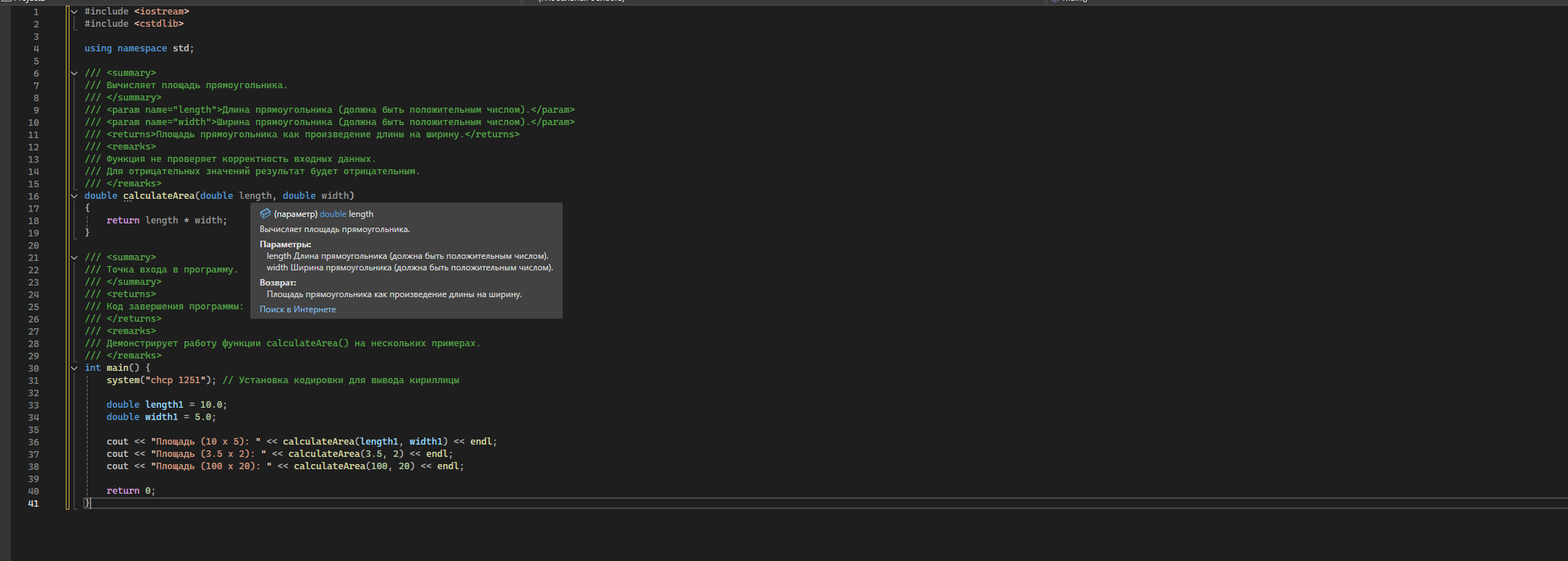


Рисунок 47 – Подсказка с комментарием

### **Задание 2**

Цель: оформить свой код согласно правилам C++

Исправленный код должен:

1. Иметь правильные отступы и пробелы

2. Использовать правильные имена переменных и функций

3. Иметь пробелы вокруг операторов

4. Правильно расставленные фигурные скобки

5. Убрать using namespace std; (опционально, но рекомендуется).

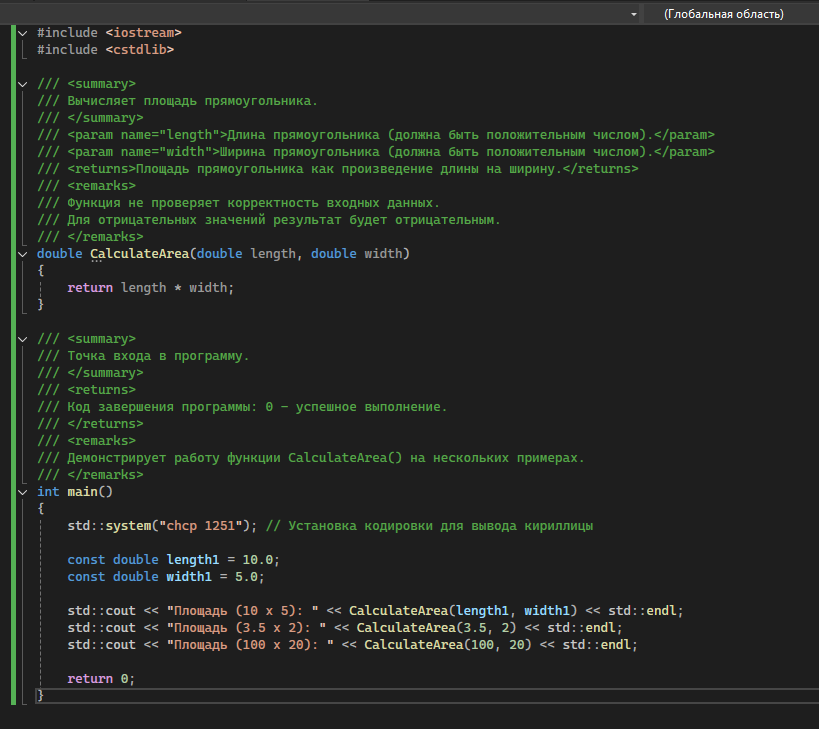


Рисунок 48 – Исправленный код

### **Задание 3**

Метрика Джилба(упрощенный вариант)

Цель: Для вашей функции из Задания 1, рассчитать сложно по метрике Джилба: метрике Джилба:

Посчитайте η1 - количество всех исполняемых строк кода (исключая объявления и комментарии)

Посчитайте η2 - количество логических переходов (if, условия в циклах)

Вычислите Gillb = η1 + η2

Gillb = 8 + 0

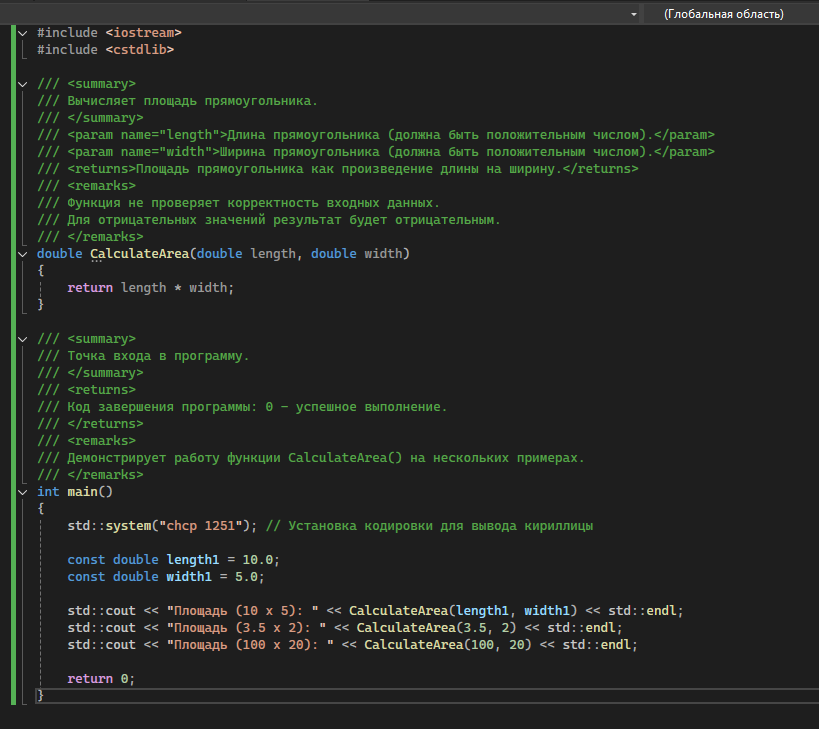


Рисунок 49 – Итоговый код

## **ПРАКТИЧЕСКАЯ РАБОТА №4 «РАБОТА С РЕЕСТРОМ ОС WINDOWS»**

Цель: Понять что такое реестр, для чего нужен. Научиться работать с файлами.

### **Задание**

* + - 1. Создать консольное приложение.
      2. Реализовать функционал для сохранения и восстановления из файла
      3. Обеспечить корректный вывод сохраненных данных при его перезапуске.

Вариант 1

Сохранить и вывести любимое слово.

Сохранить цвет консоли по выбору пользователя (red, blue, green) и восстанавливать его при запуске.

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

// функция для установки цвета по названию

void setConsoleColor(const string& color) {

    if (color == "red") {

        system("color 4F"); // красный фон + белый текст

    }

    else if (color == "blue") {

        system("color 1F"); // синий фон + белый текст

    }

    else if (color == "green") {

        system("color 2F"); // зелёный фон + белый текст

    }

    else if (color == "yellow") {

        system("color 6F"); // жёлтый фон + белый текст

    }

    else if (color == "white") {

        system("color 7F"); // белый фон + чёрный текст

    }

    else {

        system("color 0F"); // по умолчанию: чёрный фон + белый текст

    }

}

int main() {

    setlocale(0, "");

    string color;

    string name;

    setlocale(LC\_ALL, "Russian");

    ifstream fin("settings.txt"); //Создает объект fin для чтения из файла и открывает файл

    if (fin.is\_open()) {// Проверка открытия файла

        // читаем сохраненные данные

        fin >> color >> name;// Читает данные из файла (аналогично cin)

        fin.close(); //Закрывает файл и освобождает ресурсы

        setConsoleColor(color);

        cout << "Сохраненные настройки:" << endl;

        cout << "Любимый цвет: " << color << endl;

        cout << "Ваше имя: " << name << endl;

        char choice;

        cout << "\nХотите изменить настройки? (y/n): ";

        cin >> choice;

        if (choice == 'y' || choice == 'Y') {

            cout << "Введите новый любимый цвет (red, blue, green, yellow, white): ";

            cin >> color;

            cout << "Ваше новое имя: ";

            cin >> name;

            ofstream fout("settings.txt"); //Создает объект fout для записи в файл и открывает файл

            //ofstream fout("settings.txt", ios::app); // Открываем для добавления, если нужно внести еще какие-то параметры в файл ios::app, если не указывать по умолчанию очищает

            fout << color << " " << name; //Записывает данные в файл (аналогично cout)

            fout.close();

            cout << "Настройки обновлены!" << endl;

        }

    }

    else {

        // файла нет – вводим новые настройки

        cout << "Введите любимый цвет (red, blue, green, yellow, white): ";

        cin >> color;

        cout << "Ваше имя: ";

        cin >> name;

        ofstream fout("settings.txt");

        fout << color << " " << name;

        fout.close();

        cout << "Настройки сохранены!" << endl;

    }

    return 0;

}

## **ПРАКТИЧЕСКАЯ РАБОТА №5 «ЗАДАНИЕ ДЛЯ ТЕСТИРОВЩИКА»**

Цель: Провести комплексное тестирование сайта с целью оценки его качества, удобства использования и выявления потенциальных дефектов.

Задача: Вам необходимо проверить сайт как с точки зрения рядового пользователя, так и с технической стороны.(citilink)

### **Функционально тестирование (Functional Testing)**

Все кликабельные элементы ведут куда нужно

Навигационное меню работает на всех видах устройств

Сайт работает по поиску и выдает релевантные результаты

Форма проверяет указанные данные и указывает на ошибку в поле

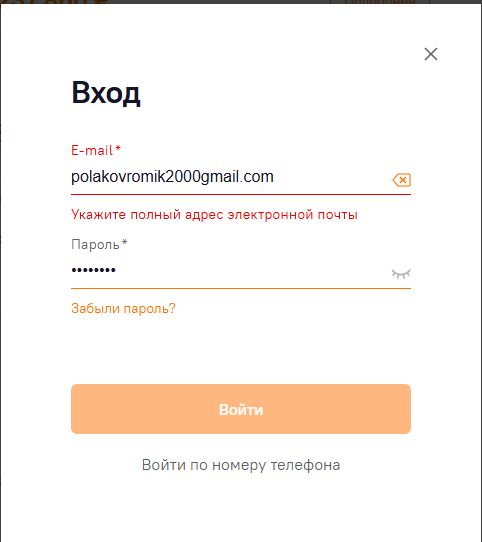


Рисунок 50 – Форма Авторизации

Нельзя отправить форму, не заполнив обязательные поля.

Форма отправляется корректно, приходит текст об успешной отправке

Поля формы очищаются после успешной отправки

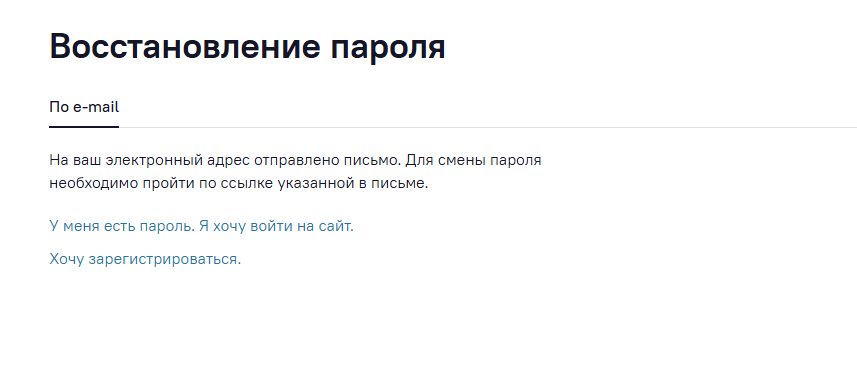


Рисунок 51 – Форма восстановления пароля

Слайдеры, карусели и аккордеоны работают корректно

Добавление, изменение и удаление товар в корзине работает корректно

Фильтры корректно применяются в сортировках и каталогах

### **Юзабилити(Usability) и Пользовательский опыт(UX)**

Первое впечатление: Цель сайта понятно с первых секунд т.к главной страницей является каталог товаров, нужную информацию найти легко

Текст читабелен при разных разрешениях экрана

Грамматических и орфографических ошибок не замечено

Структура сайта логична

Сообщения об успехе/ошибке нету, изменение цвета кнопки при наведении присутствует.

### **Тестирование совместимости(Compatibility Testing)**

Сайт проверялся на браузерах Google Chrome и Microsoft Edge.

Мобильные устройства:

Сайт адаптирован под смартфоны(IOS/Android)

Элементы не наезжают друг на друга. На кнопки нажимать удобно

Верстка не плывет

Скорость загрузки сайта: примерно 1 секунда



Рисунок 52 – скорость загрузки сайта

Загрузка контента: около 2 секунд



Рисунок 53 – загрузка контента

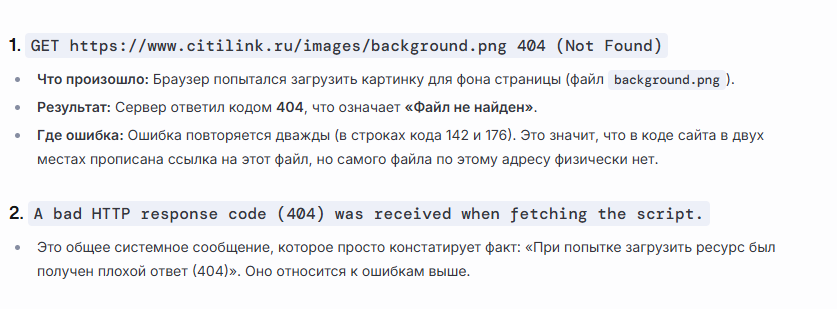
### **Тестирование безопасности(Security Testing) – (Базовая проверка)**

HTTPS: Сайт использует защищенное соединение и замок в адресной строке.

Формы: Данные из форм не передаются в виде (http)

Попытка обхода: Нельзя получить доступ к закрытым разделам(/wp-admin/,/admin/)

Ошибки при запуске сайта:



## **ПРАКТИЧЕСКАЯ РАБОТА №6 «СОЗДАНИЕ ИНСТАЛЛЯТОРОВ»**

Цель работы: Создать консольное приложение с классом и сформировать итоговый пакет.

### **Задание 1**

Создание консольного приложение с классом.

Класс «СумматорМассива» (ArraySum)

Задание: Создать класс для работы с массивом чисел.

setArray(arr,size) – установить массив

calculateSum()-вычислить сумму всех элементов

findMax()- найти максимальный элемент

findMin()-найти минимальный элемент

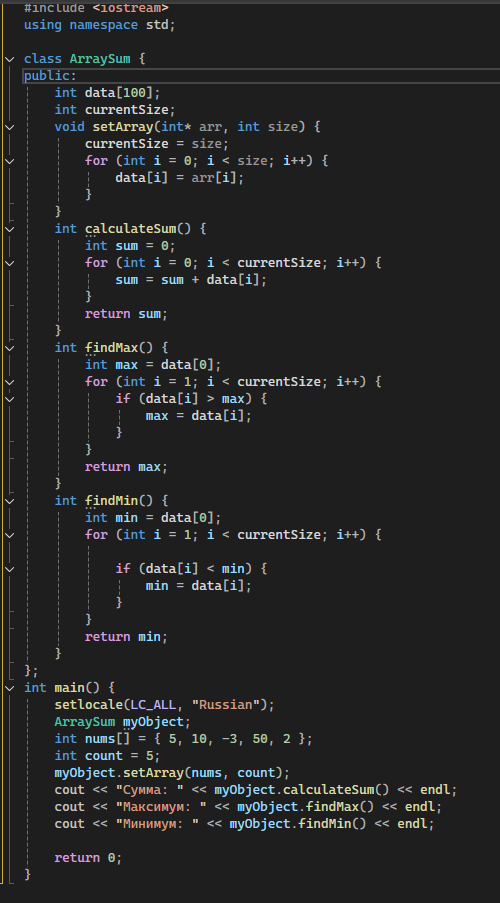


Рисунок 54 – код программы

## **ПРАКТИЧЕСКАЯ РАБОТА №7 «ТЕСТИРОВАНИЕ»**

Цель: Научиться работать со структурами, понять разницу между классом и структурой, научиться тестировать программу с помощью unit-тестов.

Задания:

Создать свою структуру по варианту.

Реализовать функции (методы) структуры по варианту.

Изучить и расписать отличия каждого вида тестирования

Написать два позитивных и 2 негативных теста

Написать 1 тест для функции напарника(обменяться проектами)

Задание:

Вариант 1

Структура для операций с дробями Написать структуру для операций с дробями, реализовать методы:

static void addFractions() - сложение двух дробей

static void multiplyFractions() - умножение двух дробей

static void simplifyFraction() - упрощение дроби

static double toDecimal() - преобразование дроби в десятичное число

Код:

#include <iostream>

using namespace std;

struct Fraction {

int n;

int d;

};

struct MathFractionOperations {

static Fraction simplify(Fraction f) {

if (f.d == 0) {

cout << "Ошибка: знаменатель равен 0" << endl;

return f;

int a = f.n;

int b = f.d;

if (a < 0) a = -a;

if (b < 0) b = -b;

while (b > 0) {

int temp = b;

b = a % b;

a = temp;

}

f.n = f.n / a;

f.d = f.d / a;

return f;

}

static Fraction add(Fraction f1, Fraction f2) {

Fraction res;

res.n = f1.n \* f2.d + f2.n \* f1.d;

res.d = f1.d \* f2.d;

return simplify(res);

}

static Fraction multiply(Fraction f1, Fraction f2) {

Fraction res;

res.n = f1.n \* f2.n;

res.d = f1.d \* f2.d;

return simplify(res);

}

static double toDecimal(Fraction f) {

if (f.d == 0) return 0.0;

return (double)f.n / f.d;

}

};

int main() {

setlocale(LC\_ALL, "");

Fraction a = { 1, 2 };

Fraction b = { 2, 4 };

cout << "Дробь 1: " << a.n << "/" << a.d << endl;

cout << "Дробь 2: " << b.n << "/" << b.d << endl;

cout << "----------------" << endl;

Fraction sumRes = MathFractionOperations::add(a, b);

cout << "Сумма: " << sumRes.n << "/" << sumRes.d << endl;

Fraction multRes = MathFractionOperations::multiply(a, b);

cout << "Умножение: " << multRes.n << "/" << multRes.d << endl;

double decimal = MathFractionOperations::toDecimal(a);

cout << "Десятичное число: " << decimal << endl;

return 0;

}

## **ПРАКТИЧЕСКАЯ РАБОТА №8 «РАЗРАБОТКА ТЗ И МАКЕТОВ ДЛЯ САЙТА»**

Цель работы: Научиться создавать четкое техническое задание и разрабатывать к нему визуальные макеты(как черно-белые, так и цветные) для статического сайта.

### **Техническое задание**

На разработку сайта-визитки для кофейни «У Кирилла»

Цели и описание проекта

Ключевая цель: Создать уютное онлайн-пространство, которое привлечет новых посетителей, проинформирует гостей о текущих акциях и позволит ознакомиться с ассортиментом меню и интерьером заведения.

Целевая аудитория:

Студенты и фрилансеры (поиск места для работы/учебы).

Жители района (утренний кофе, встречи с друзьями).

Ценители качественного зерна и альтернативных способов заваривания.

Общее описание:

Разработка многостраничного статического сайта. Проект не требует подключения базы данных (БД) или сложной админ-панели. Акцент делается на визуальную атмосферу («уют»), интуитивную навигацию и демонстрацию спецпредложений.

Функциональные требования

Главная страница

Слайдер акций (JS): В верхней части страницы («Первый экран») расположен автоматический слайдер.

Контент: Баннеры с текущими предложениями (например, «Утренний кофе -20%», «Сезонное меню»).

Функционал: Переключение слайдов по таймеру и вручную (стрелки/точки).

Приветственный блок: Краткий текст, передающий атмосферу кофейни, и кнопка (CTA) «Посмотреть меню», ведущая на соответствующую страницу.

Страница «Меню»

Структура: Простой и понятный текстовый список позиций без перегрузки фотографиями (согласно требованию «простой список»).

Категории: Меню разделено на логические блоки: «Кофе», «Авторские чаи», «Десерты», «Завтраки».

Содержание позиции: Название, состав/объем, цена.

Страница «О нас»

Философия: Текстовый блок, описывающий подход к выбору зерна, обжарке и обслуживанию гостей.

Команда: Блок с кратким описанием бариста или истории создания кофейни.

Страница «Галерея»

Сетка (Grid): Фотографии интерьера, напитков и гостей, расположенные в плитку.

Лайтбокс (Lightbox JS): При клике на миниатюру изображение открывается в полноэкранном режиме с затемнением фона. Должна быть возможность закрыть фото (крестик) или перелистнуть на следующее.

Страница «Контакты»

Информация: Адрес, телефон, режим работы, ссылки на социальные сети.

Форма обратной связи/Брони:

Поля: «Имя», «Телефон», «Дата», «Комментарий».

Отправка данных осуществляется через внешний сервис (Formspree) на email администратора.

Сквозные элементы (Шапка и Подвал)

Шапка: Логотип «У Кирилла», навигационное меню (Главная, Меню, О нас, Галерея, Контакты).

Подвал: Копирайт, адрес, иконки соцсетей (Instagram, Telegram).

Техстек и интеграции

Фронтенд:

HTML5: Семантическая верстка.

CSS3: Flexbox и Grid для макетирования, медиа-запросы для адаптивности.

JavaScript (Vanilla): Реализация слайдера на главной и лайтбокса в галерее без использования тяжелых библиотек (типа jQuery).

Хостинг: GitHub Pages (статический хостинг).

Интеграции:

Форма: Сервис Formspree или EmailJS для обработки заявок.

Карты: Встроенный iframe Яндекс.Карт или Google Maps на странице контактов.

Иконки: Библиотека FontAwesome или SVG.

Нефункциональные требования

a) Производительность

Оптимизация изображений (формат WebP, сжатие) для быстрой загрузки галереи.

Время полной загрузки главной страницы < 3 секунд (3G/4G).

Плавная анимация слайдера и модальных окон (без рывков).

b) Совместимость

Адаптивность: Корректное отображение на смартфонах (вертикальная верстка, бургер-меню), планшетах и десктопах (от 320px до 1920px).

Кроссбраузерность: Chrome, Safari, Firefox, Edge (последние актуальные версии).

c) Удобство использования (Usability) и Стиль

Дизайн: «Уютный». Цветовая палитра: теплые кофейные оттенки (коричневый, бежевый, молочный), мягкий оранжевый или зеленый как акцент.

Типографика: Читабельные шрифты. Для заголовков возможно использование шрифта с засечками (serif) или рукописного стиля, для основного текста — гротеск (sans-serif).

Интуитивность: Пользователь должен находить цену кофе или адрес не более чем за 2 клика.

d) Масштабируемость и сопровождение

Понятная структура кода с комментариями (особенно в JS-скриптах слайдера и галереи).

Использование методологии БЭМ для CSS-классов.

Критерии тестирования

Слайдер: Слайды переключаются автоматически каждые 5 секунд и реагируют на клики по стрелкам.

Галерея: При клике фото открывается на весь экран, не перекрывается меню сайта.

Адаптив: На мобильных устройствах меню сворачивается, таблица/список меню не выходит за границы экрана.

Форма: При отправке заявки приходит письмо на тестовый email.

Сроки поэтапной сдачи

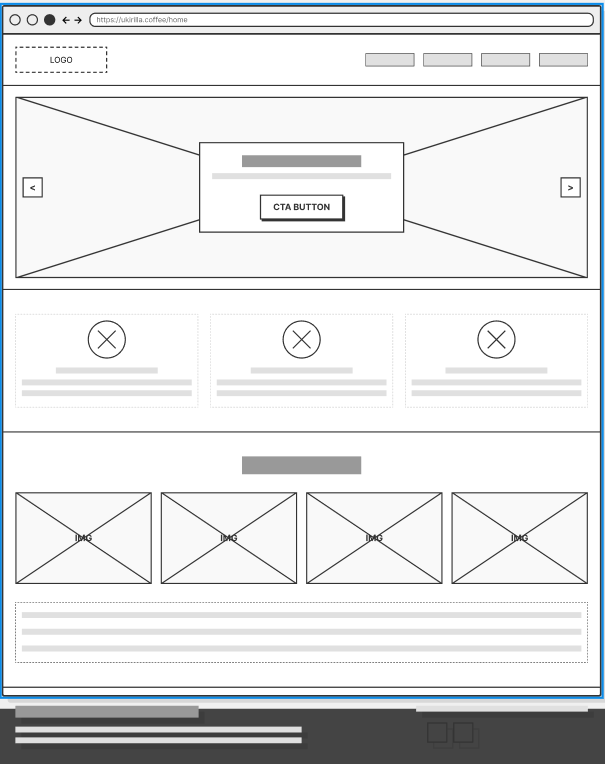
Этап 1 (Неделя 1): Согласование черно-белых макетов (Wireframes) Главной страницы и Галереи.

Этап 2 (Неделя 2): Согласование цветных макетов, утверждение шрифтов и цветовой гаммы.

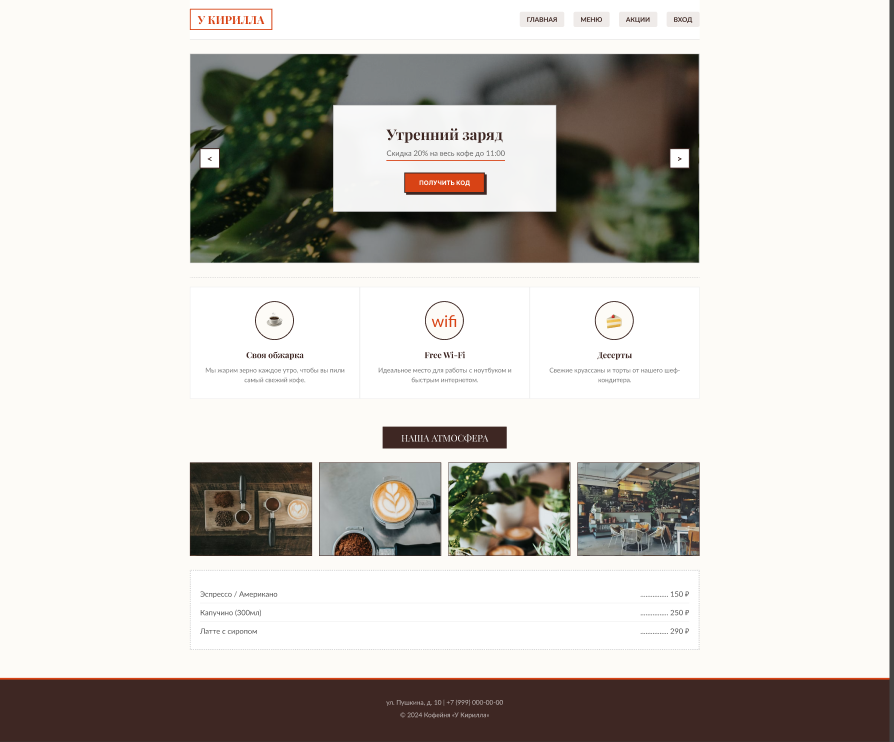
Этап 3 (Неделя 3): Верстка Главной страницы (со слайдером) и страницы Меню.

Этап 4 (Неделя 4): Верстка остальных страниц, подключение скриптов галереи и форм, финальное тестирование.

### **Макеты**







## **ПРАКТИЧЕСКАЯ №9 «РАБОТА С СИСТЕМОЙ КОНТРОЛЯ ВЕРСИЙ GIT»**

### **Задание 1**

«Введение»

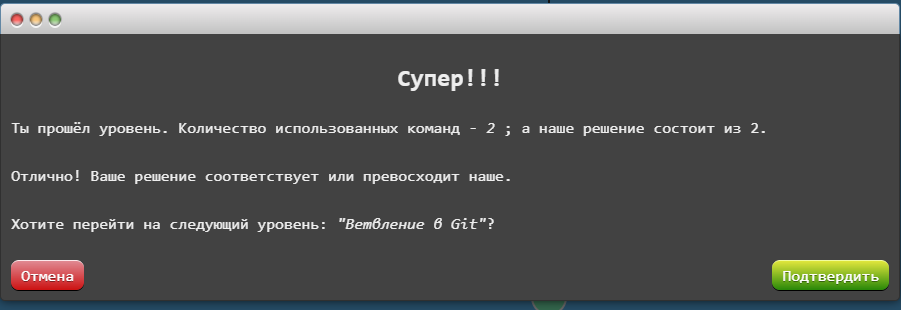


Рисунок – 1 лвл



Рисунок – прохождение 1 лвл

Использовались команды:

Git commit

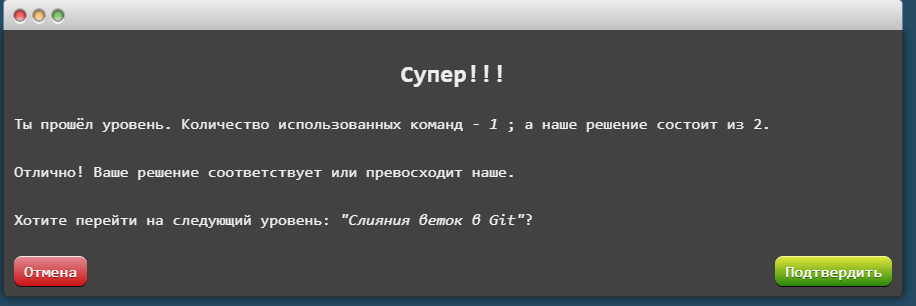


Рисунок – 2 лвл

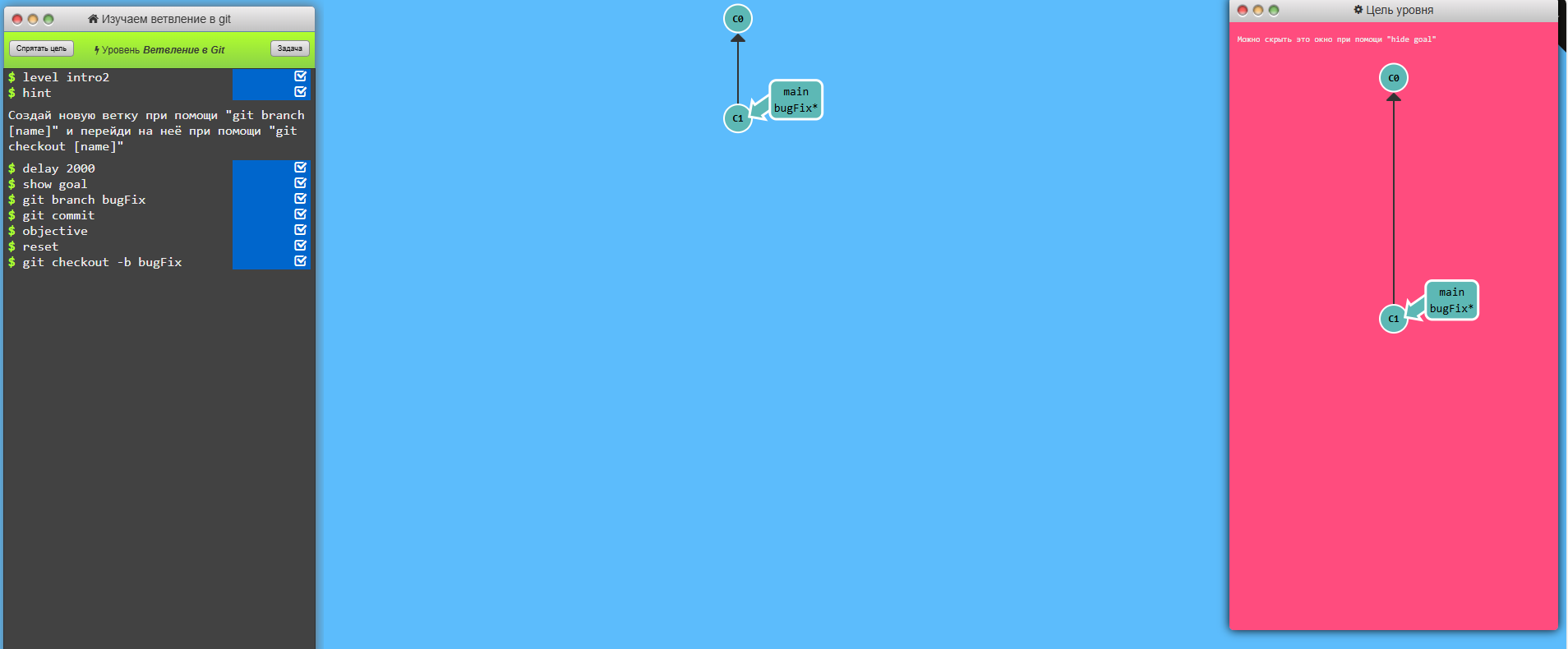


Рисунок –прохождение 2 лвл

Использовались команды:

Git checkout -b(branch)



Рисунок –3 лвл

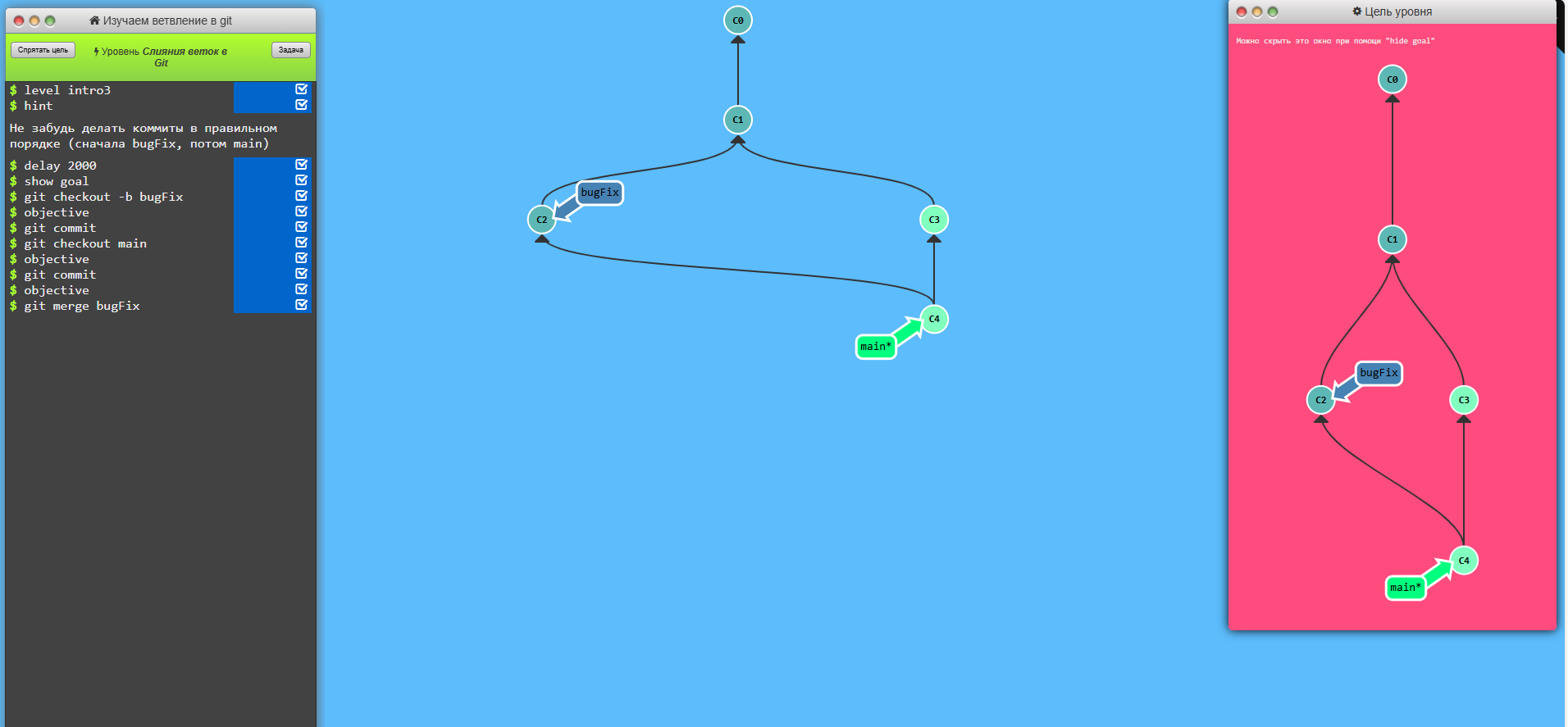


Рисунок – прохождение 3 лвл

Использовались команды:

Git checkout -b

Git commit

Git checkout

Git merge

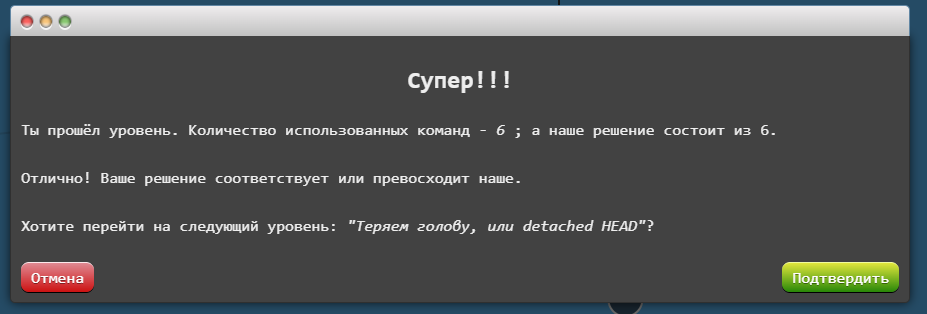


Рисунок – 4 лвл



Рисунок –прохождение 4 лвл

Использовались команды:

Git checkout -b

Git commit

Git checkout

Git rebase

«Едем дальше»

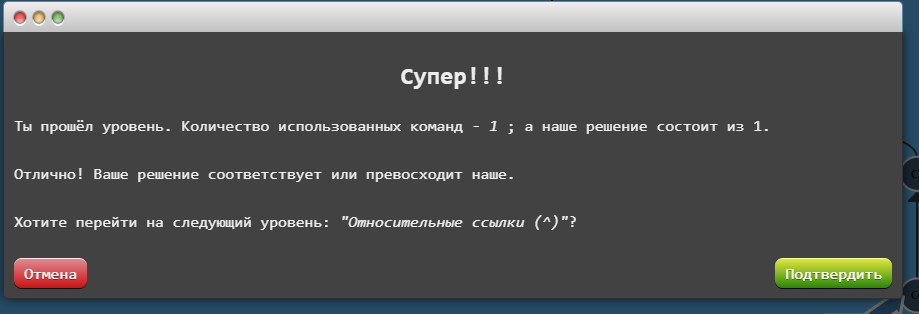


Рисунок – 1 лвл

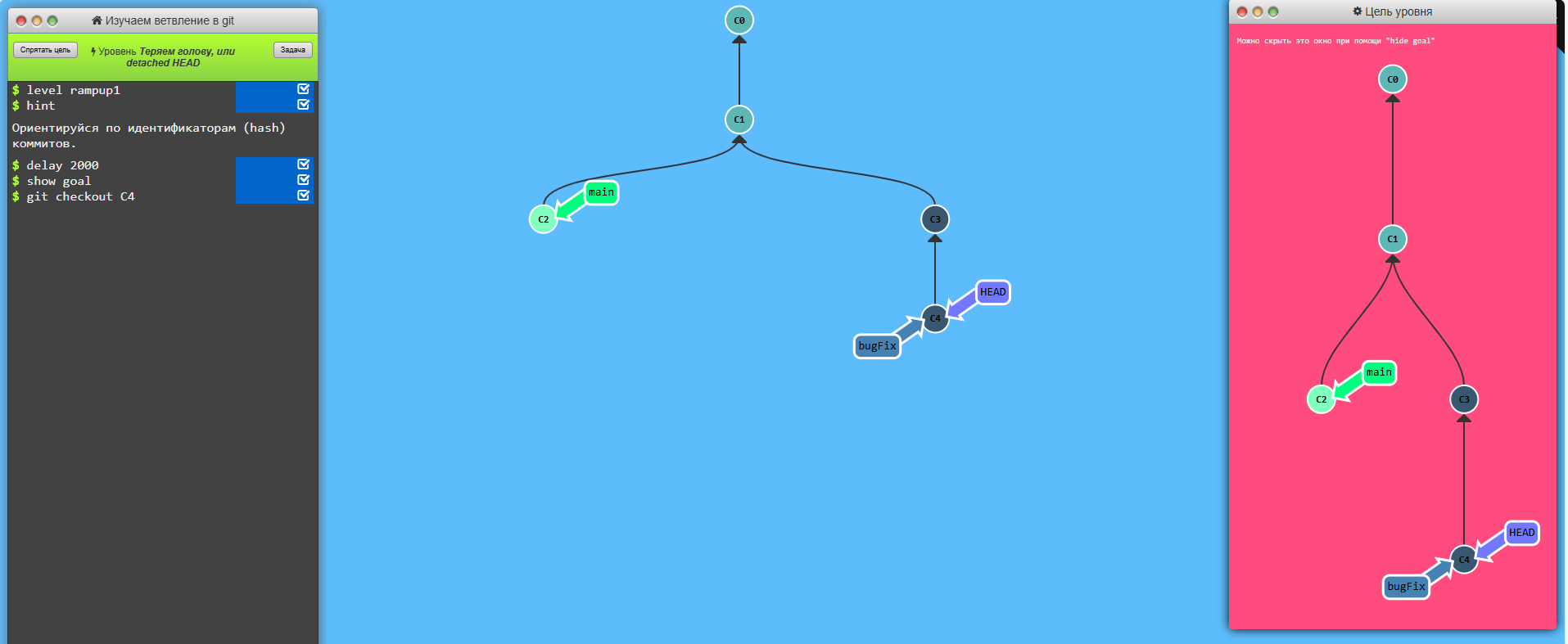


Рисунок – прохождение 1 лвл

Использовались команды:

Git checkout

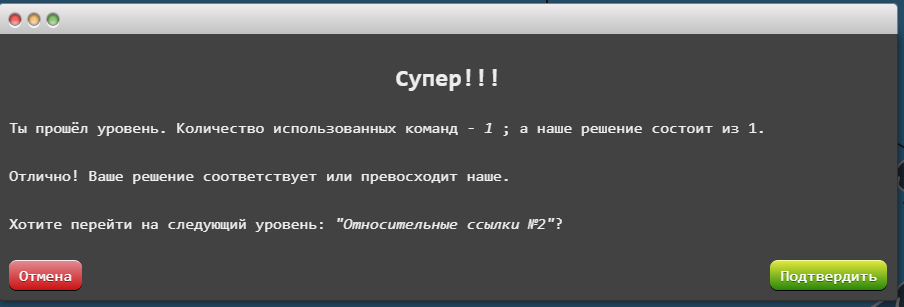


Рисунок –2 лвл

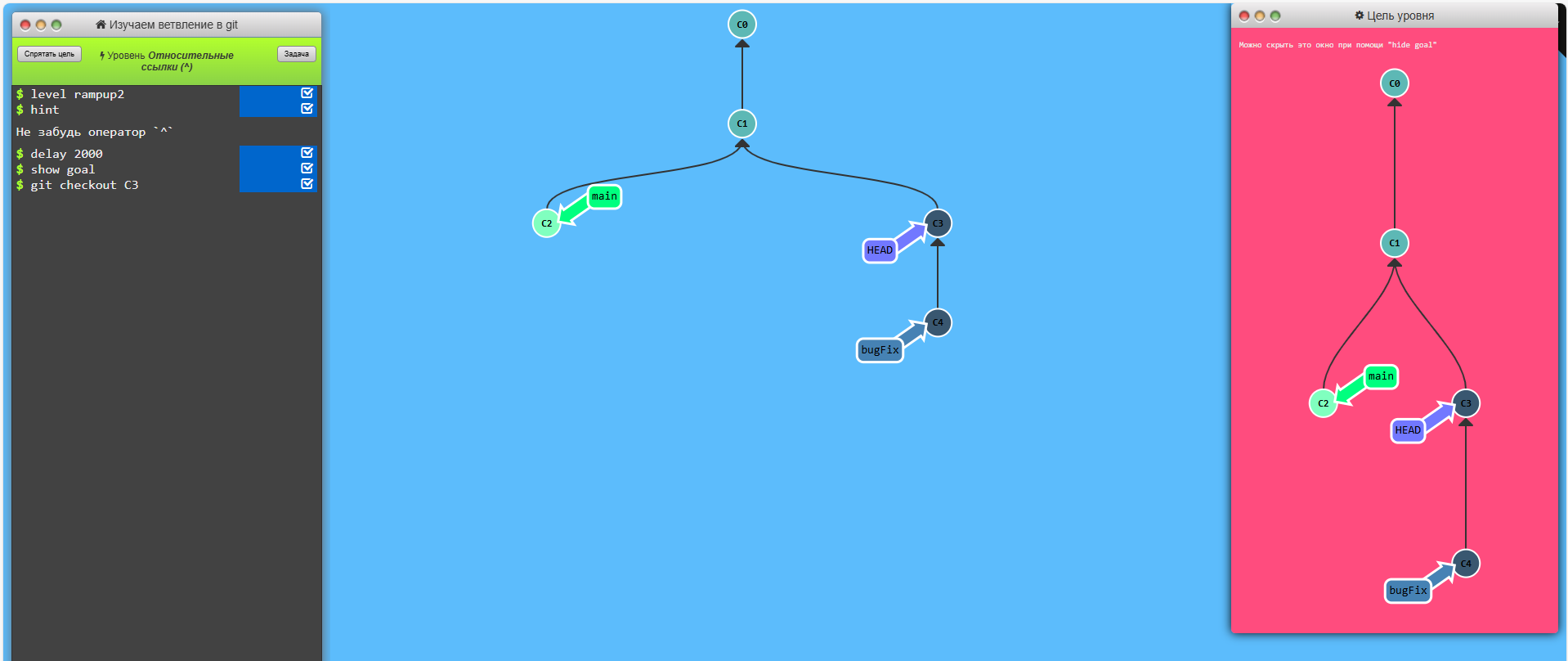


Рисунок – прохождение 2 лвл

Использовались команды:

Git checkout

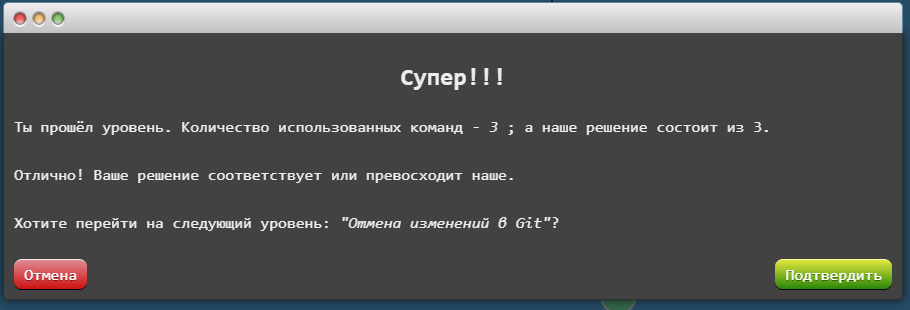


Рисунок – 3 лвл

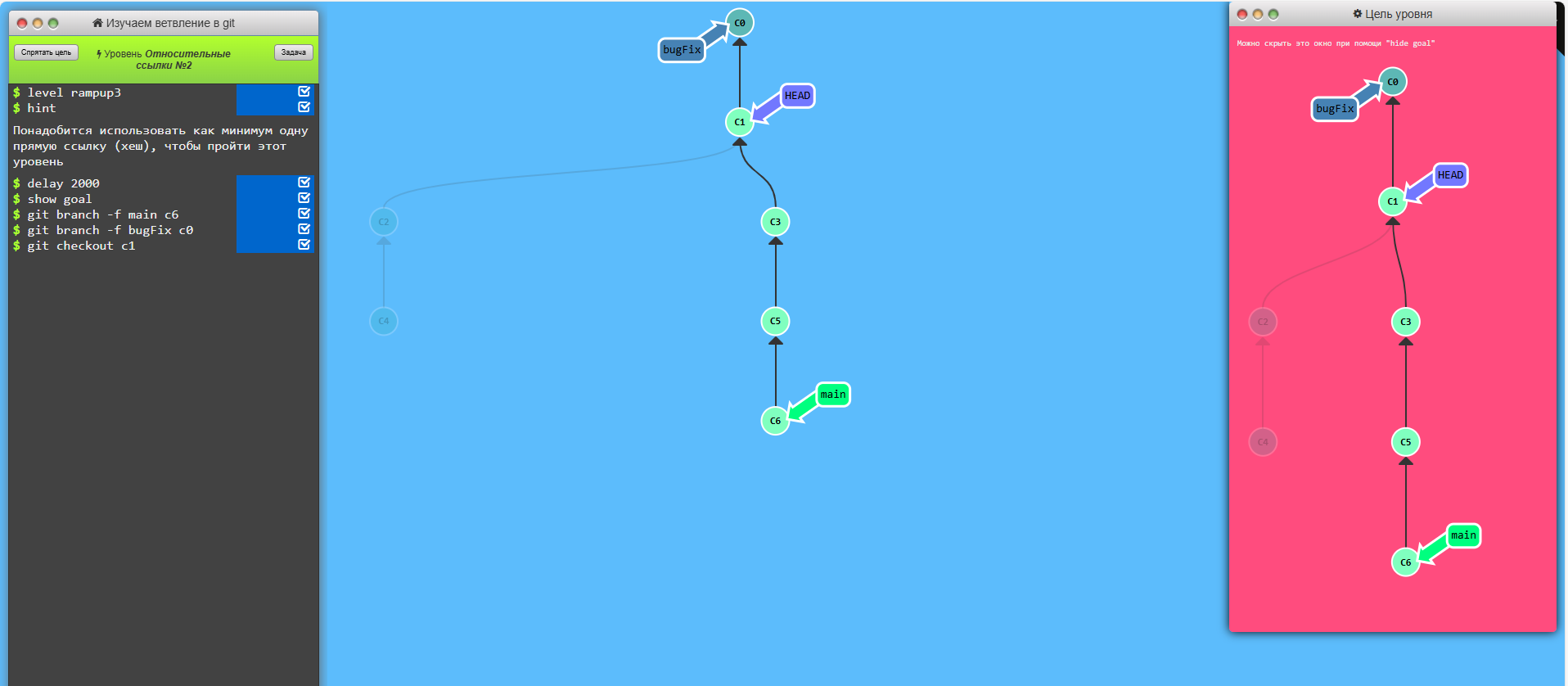


Рисунок – прохождение 3 лвл

Использовались команды:

Git branch -f

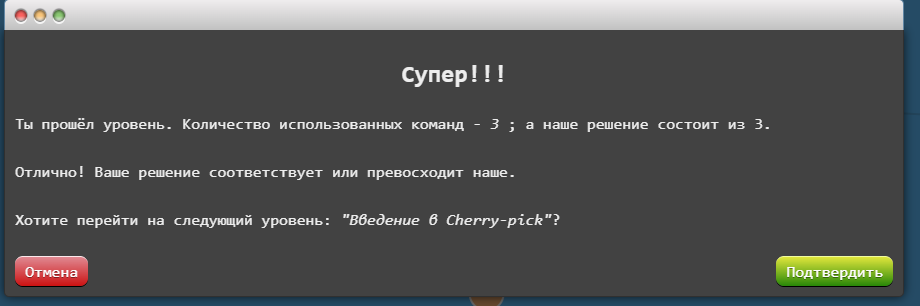


Рисунок – 4 лвл

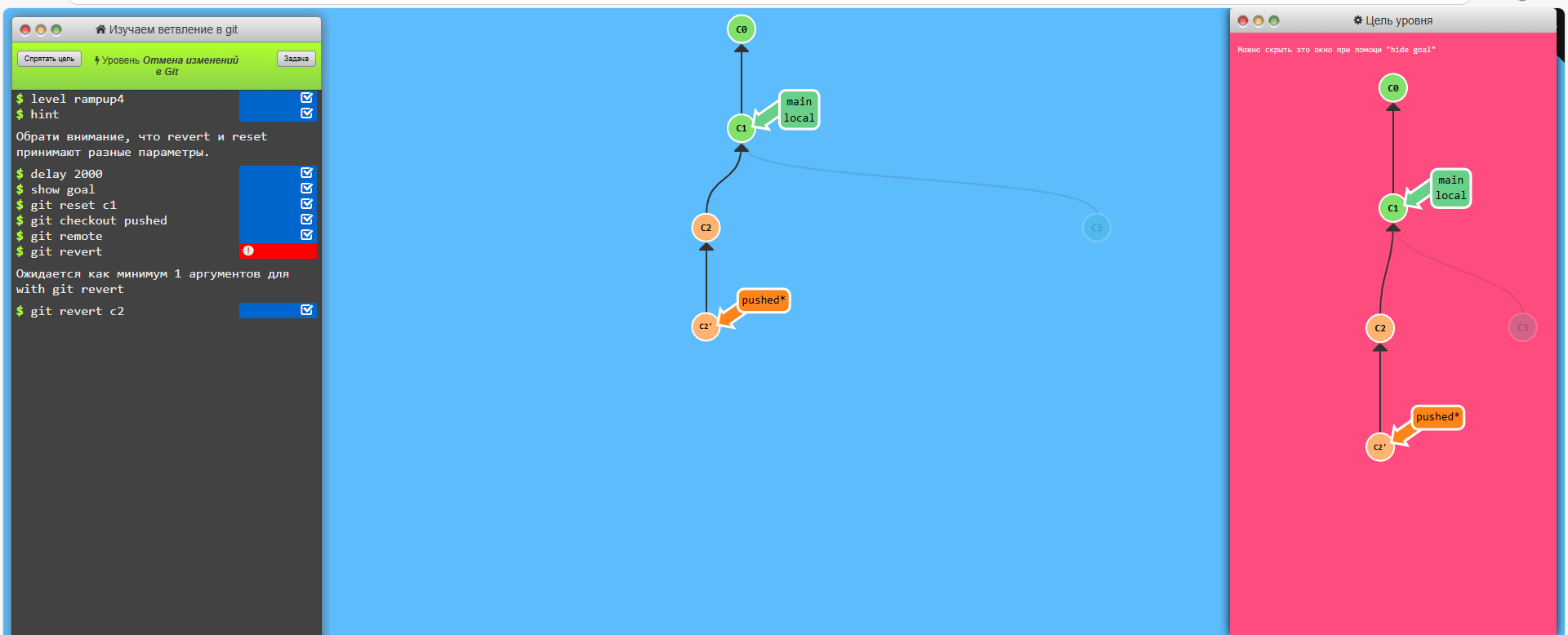


Рисунок – прохождение 4 лвл

Использовались команды:

1. Git reset
2. Git checkout
3. Git remote
4. Git revert

«Push & Pull удаленные репозитории в git»

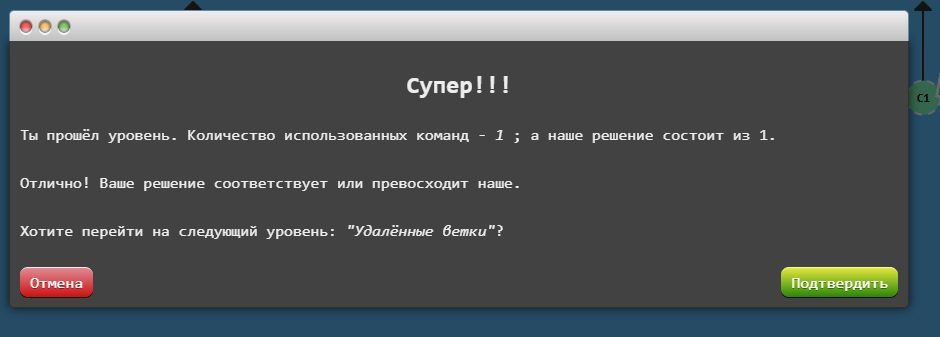


Рисунок – 1 лвл

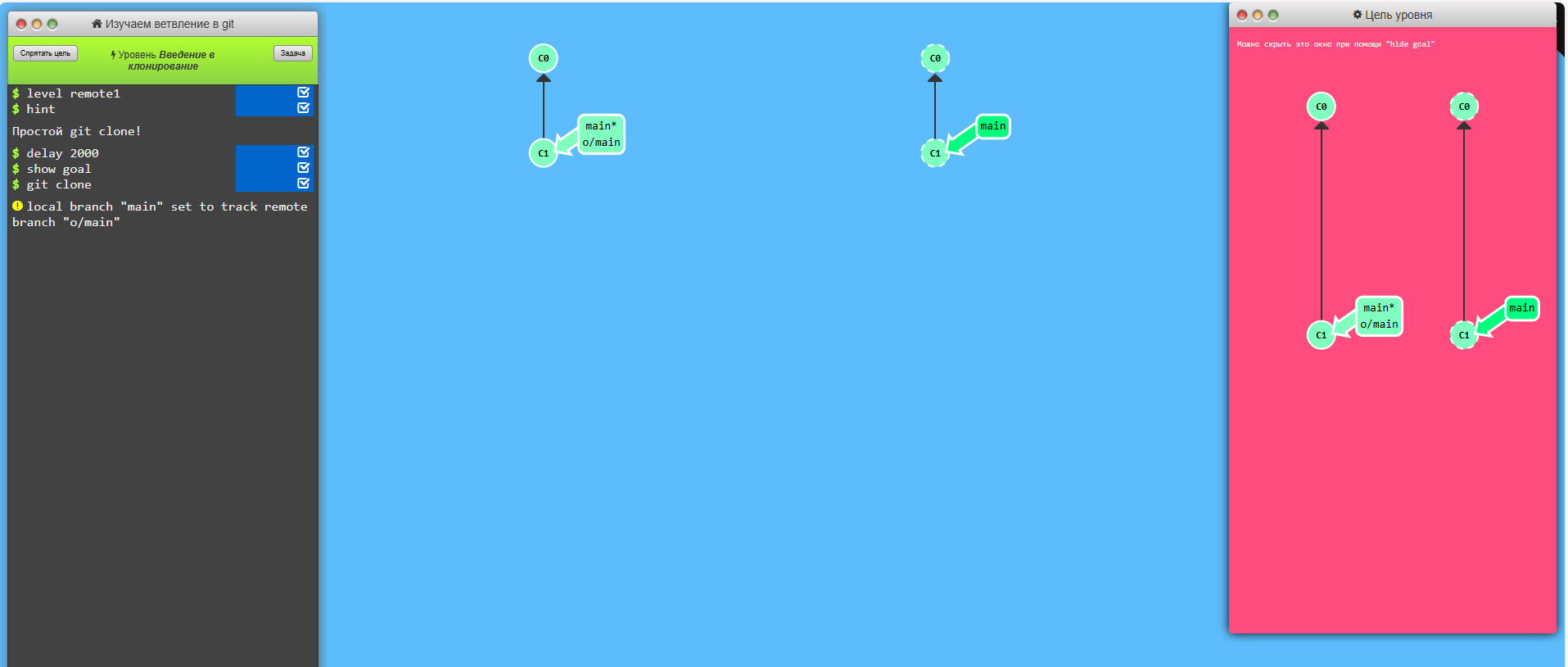


Рисунок –прохождение 1 лвл

Использовались команды:

Git clone

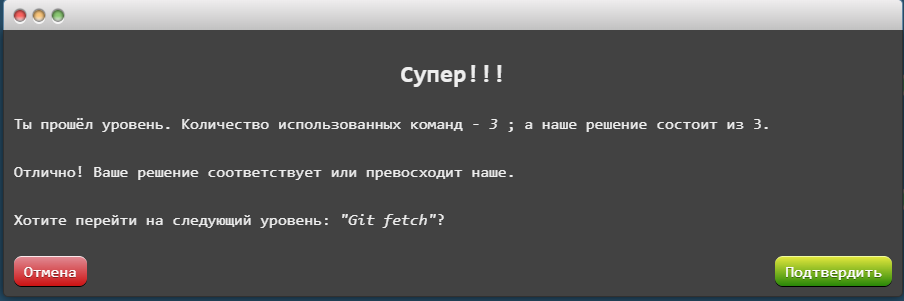
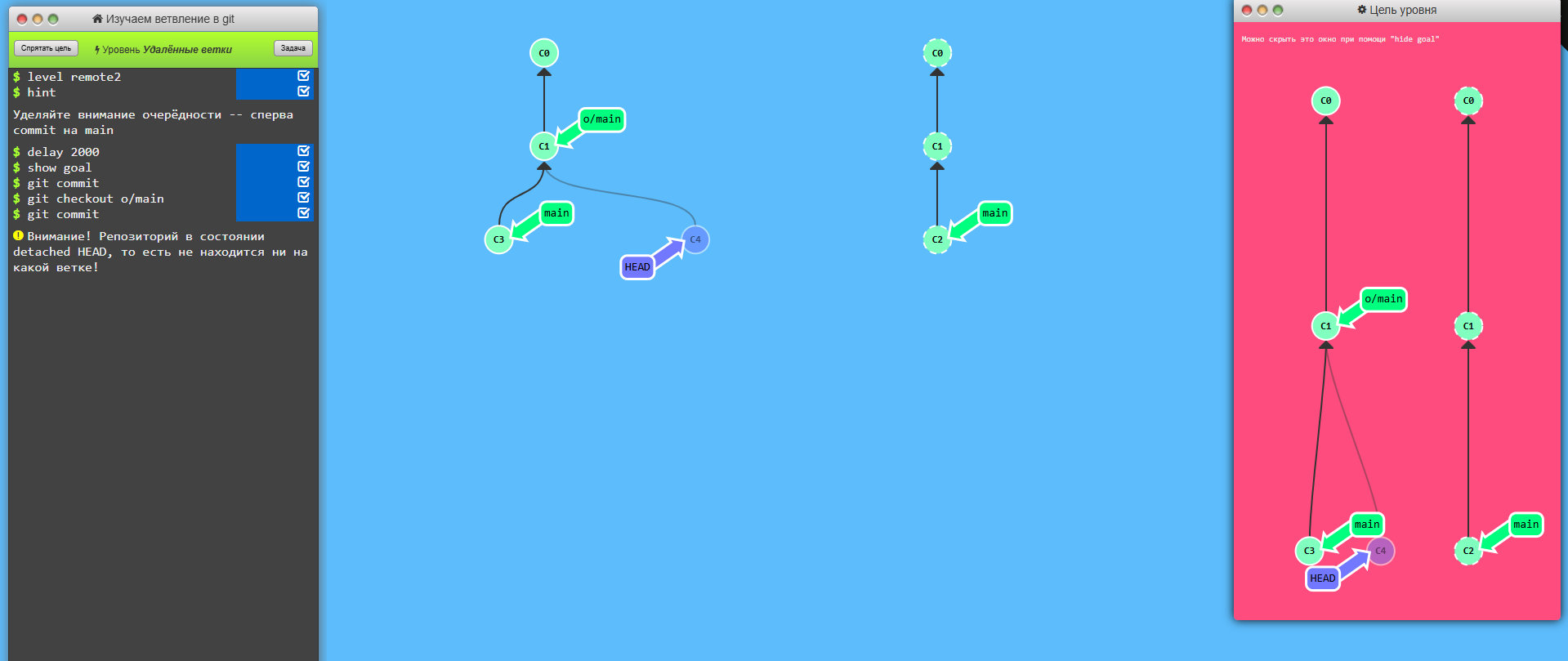


Рисунок – 2 лвл

Рисунок –прохождение 2 лвл

Использовались команды:

Git commit

Git checkout

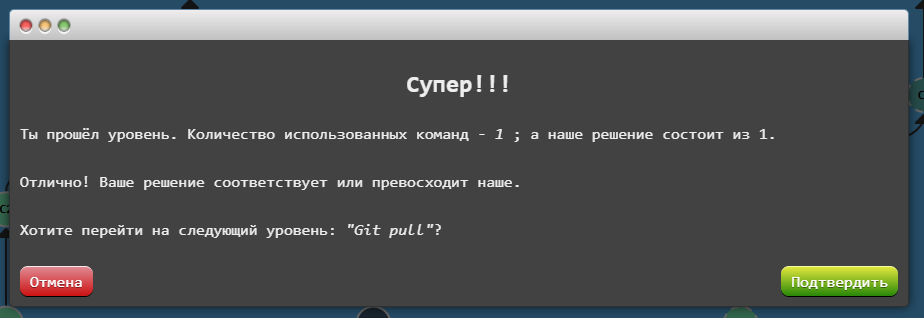


Рисунок –3 лвл

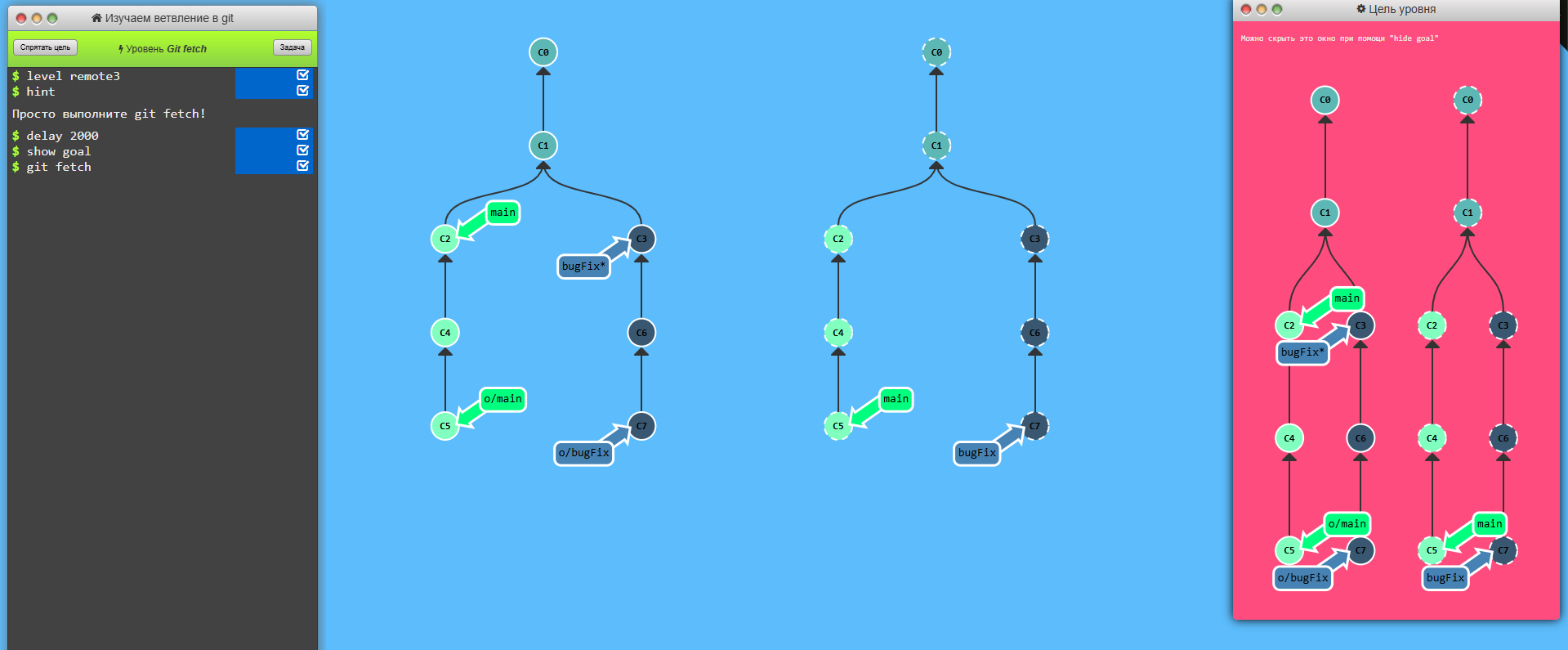


Рисунок –прохождение 3 лвл

Использовались команды:

### **Задание 2**

Git – программа для контроля версий, устанавливается на компьютер, отслеживает изменения в коде и работает полностью офлайн

GitHub – веб-сервис (сайт) для хранения Git – репозиториев в облаке, позволяет делиться кодом и работать в команде.

GitBash – терминал (командная строка) для Windows с Unix – окружением, через который вводятся команды Git.

Шаги регистрации на GitHub:

1. Перейти на сайт github.com и нажать “Sign up”
2. Ввести email адрес и нажать “Continue”
3. Придумать и ввести пароль(минимум 15 символов или 8 символов с цифрой и буквой, нажать “Continue”
4. Придумать имя пользователя – оно будет отображаться в ссылках на профиль и репозитории, нажать “Continue”
5. Выбрать, получать ли рассылку
6. Пройти проверку «Я не робот» и нажать “Create account”
7. Открыть почту, найти письмо от GitHub и ввести код подтверждения из письма
8. Ответить на вопросы о себе
9. Выбрать тип аккаунта

Добавление проектов:

1. Зайти на GitHub и авторизоваться
2. Нажать «+» в правом верхнем углу, “New repository”
3. Ввести название репозитория
4. Выбрать видимость
5. Нажать “create repository”

GitHub

https://github.com/koluvashka/Pr04.01.git

ЗАКЛЮЧЕНИЕ

Заключение пишем ко всему отчету сразу, не к отдельным работам

**АТТЕСТАЦИОННЫЙ ЛИСТ**

**ПО УЧЕБНОЙ ПРАКТИКЕ (ПО ПРОФИЛЮ СПЕЦИАЛЬНОСТИ)**

по профессиональному модулю ПМ.04 «Сопровождение и обслуживание программного обеспечения компьютерных систем»

(код и наименование)

Специальность 09.02.07 «Информационные системы и программирование»

(код и наименование специальности)

Студент(ка) II курса 24290907/3091 группы

Поляков Роман Александрович

(ФИО полностью)

Место прохождения практики: ФГАОУ ВО СПбПУ Петра Великого  
Институт СПО, учебно-вычислительный центр, пр. Энгельса д.23

(наименование и адрес организации)

Период прохождения практики

с «8» декабря 2025 г. по «27» декабря 2025 г.

**Виды и качество выполнения работ**

|  |  |  |
| --- | --- | --- |
| **Виды выполненных работ обучающимся**  **во время практики** | **Объем работ, час.** | **Качество выполнения работ (оценка по пятибалльной системе)** |
| **Тема 1** Персонализация интегрированной среды разработки Visual Studio Community 2022 | 18 |  |
| **Тема 2** Отладка в IDE Visual Studio Community 2022 | 30 |  |
| **Тема 3** Обеспечение качества кода | 36 |  |
| **Тема 4** Упаковка приложения | 24 |  |

Характеристика учебной/профессиональной деятельности обучающегося во время учебной практики (по профилю специальности):

Общие и профессиональные компетенции, предусмотренные программой практики, *освоены* ***/*** *не освоены*.

(нужное подчеркнуть)

Итоговая оценка по практике \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель практики     Курылева А. А. \_\_\_\_\_\_\_\_\_\_\_\_\_

(Ф.И.О.) (подпись)