**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

Тюнин Илья Андреевич

(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 Васильев С. Ю.

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

Тюнин Илья Андреевич

(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. Васильев С. Ю. (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

Тюнин Илья Андреевич

(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 Васильев С. Ю.

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 6](#_Toc217037405)

[1 ПРАКТИЧЕСКАЯ РАБОТА №1 «НАСТРОЙКА ВНЕШНЕГО ВИДА И ФУНКЦИОНАЛЬНОСТИ СРЕДЫ РАЗРАБОТКИ VISUAL STUDIO» 7](#_Toc217037406)

[1.1 Параметры текстового редактора 8](#_Toc217037407)

[2.1 Изменение шрифтов и цветов 8](#_Toc217037408)

[3.1 Настройка меню и панели инструментов 8](#_Toc217037409)

[4.1 Настройки макетов окон 8](#_Toc217037410)

[5.1 Экспорт настроек 8](#_Toc217037411)

[2 ПРАКТИЧЕСКАЯ РАБОТА №2 «ОТЛАДКА В VISUAL STUDIO» 8](#_Toc217037412)

[1.1 Навигация по коду с помощью отладчика 8](#_Toc217037413)

[2.1 Управление исключениями 8](#_Toc217037414)

[3.1 Профилирование 8](#_Toc217037415)

[ЗАКЛЮЧЕНИЕ 8](#_Toc217037416)

ВВЕДЕНИЕ

Введение пишется ко всему отчету, не к каждой работе отдельно

1. ПРАКТИЧЕСКАЯ РАБОТА №1 «НАСТРОЙКА ВНЕШНЕГО ВИДА И ФУНКЦИОНАЛЬНОСТИ СРЕДЫ РАЗРАБОТКИ VISUAL STUDIO»

Цель:освоить настройку интерфейса и функциональности Visual Studio для комфортной работы.

* 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 C++, открыть общие настройки и выбрать пункт «Форматирование». В последующем установить флаги на соответствующих пунктах. Эти настройки позволят автоматически форматировать код во время ввода указанных символов (Рисунок 1).

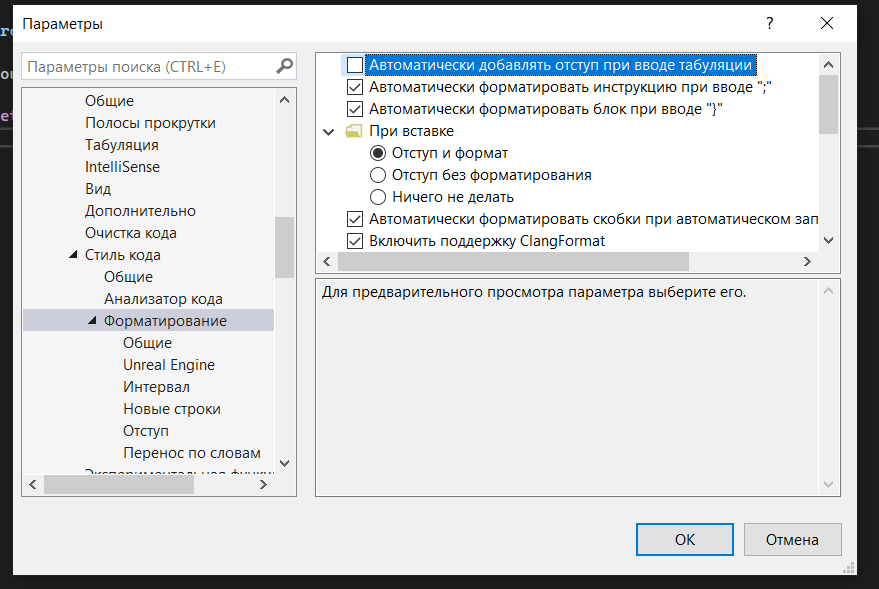


Рисунок 1 – Автоматическое форматирование при вводе } и ;.

Чтобы убрать автоматическое форматирование при вводе} и ;, нам нужно пройти тот же путь и установить флажки, как показано на рисунке (Рисунок 2).

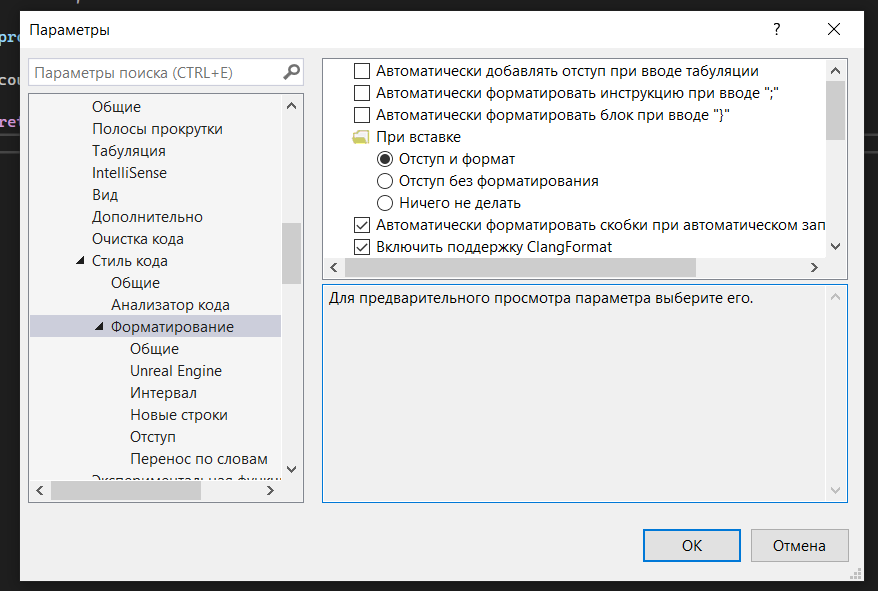


Рисунок 2 – Отключение автоматического форматирования.

Для изменения интервалов в разделе «Средства → Параметры → Текстовый редактор» выбираем раздел «Стиль кода → Форматирование → Интервал», выбираем наш язык программирования (C++) и устанавливаем флаг на пункте «Вставлять пробел между скобками со списком аргументов» (Рисунок 3).

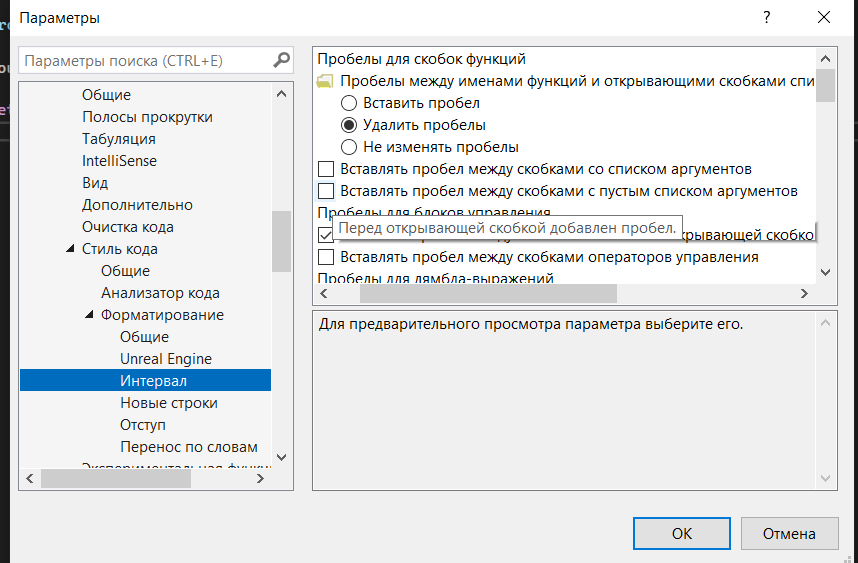


Рисунок 3 – Установка флага «Вставлять пробел между скобками со списком аргументов».

Чтобы включить нумерацию строк, нам необходимо перейти по такому маршруту «Средства → Параметры → Текстовый редактор → C/C++ → Общие» и в разделе «Общие» найти флажок «Номера строк». Кликнув по нему, мы включим нумерацию строк (Рисунок 4).

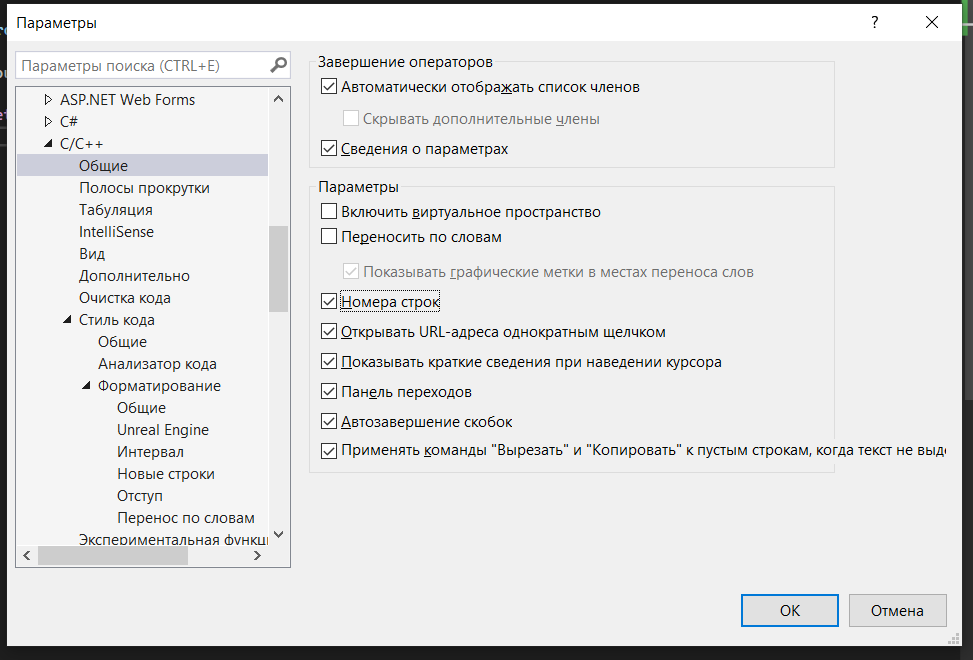


Рисунок 4 – Включение нумерации строк.

* 1. Изменение шрифтов и цветов

Дабы изменить настройки для всплывающих настроек, нам необходимо пройти такой путь «Средства → Параметры → Окружение → Шрифты и цвета», в «Параметры для» необходимо выбрать «Всплывающая подсказка редактора» после чего их можно редактировать (Рисунок 5).



Рисунок 5 – Пример редактирования Всплывающих подсказок.

Для редактирования стиля текста среды проходим такой же путь «Средства → Параметры → Окружение → Шрифты и цвета», но в «Параметры для» мы выбираем «Текстовый редактор», в «Отображаемые элементы» выбираем «Обычный текст» и редактируем (Рисунок 6).

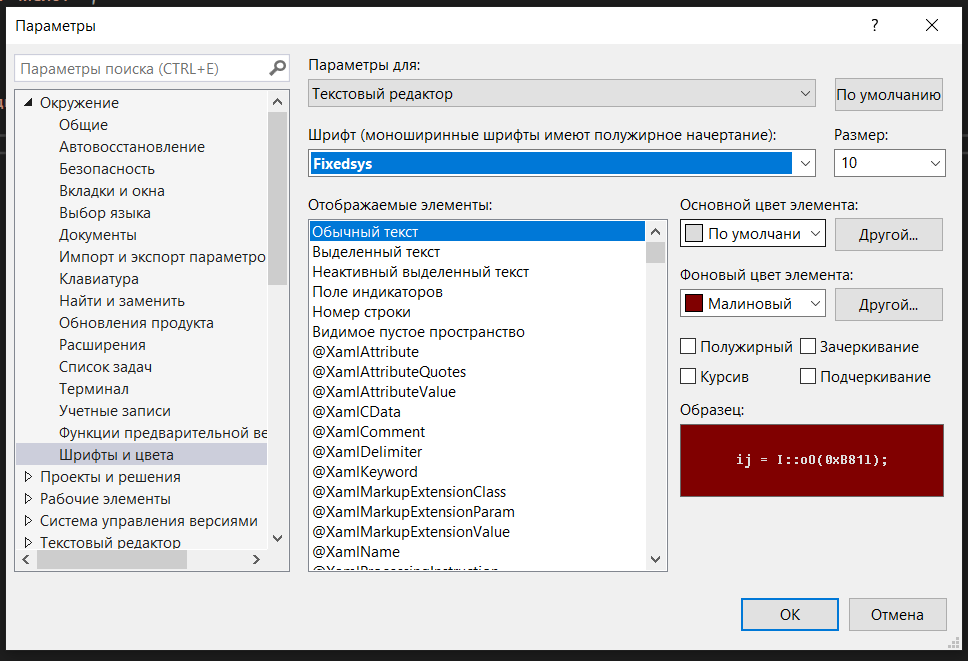


Рисунок 6 – Пример изменения текста.

Для редактирования стиля нумерации строк такой же путь «Средства → Параметры → Окружение → Шрифты и цвета», но в «Параметры для» мы выбираем «Текстовый редактор», в «Отображаемые элементы» выбираем «Номер строки» и редактируем (Рисунок 7).

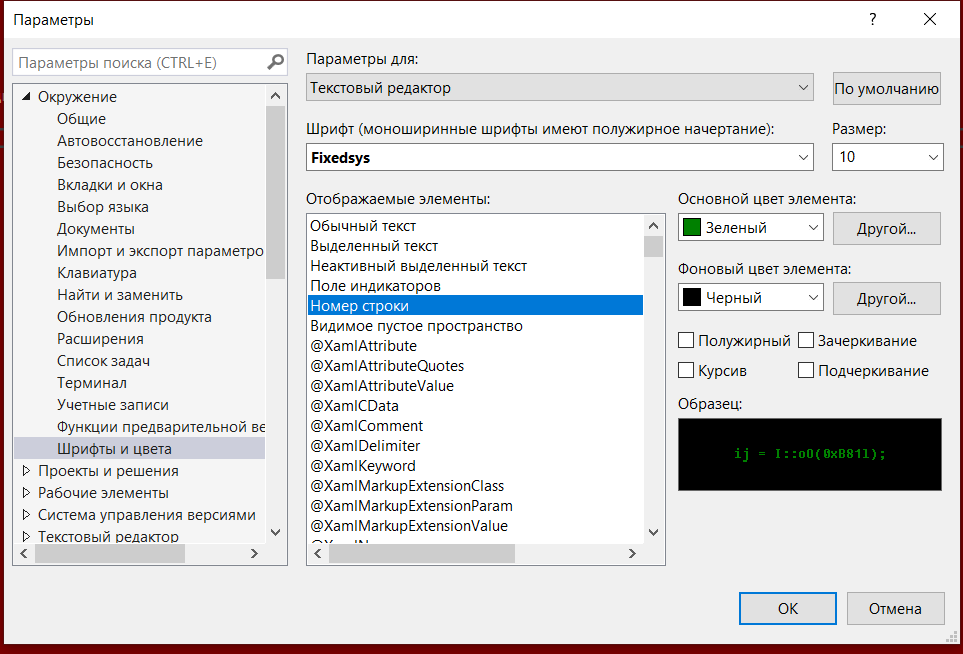


Рисунок 7 – Пример изменения стиля номера строки.

Для редактирования стиля комментариев нужно пройти такой путь «Средства → Параметры → Окружение → Шрифты и цвета», но в «Параметры для» мы выбираем «Текстовый редактор», в «Отображаемые элементы» выбираем «Комментарии» и редактируем (Рисунок 8).

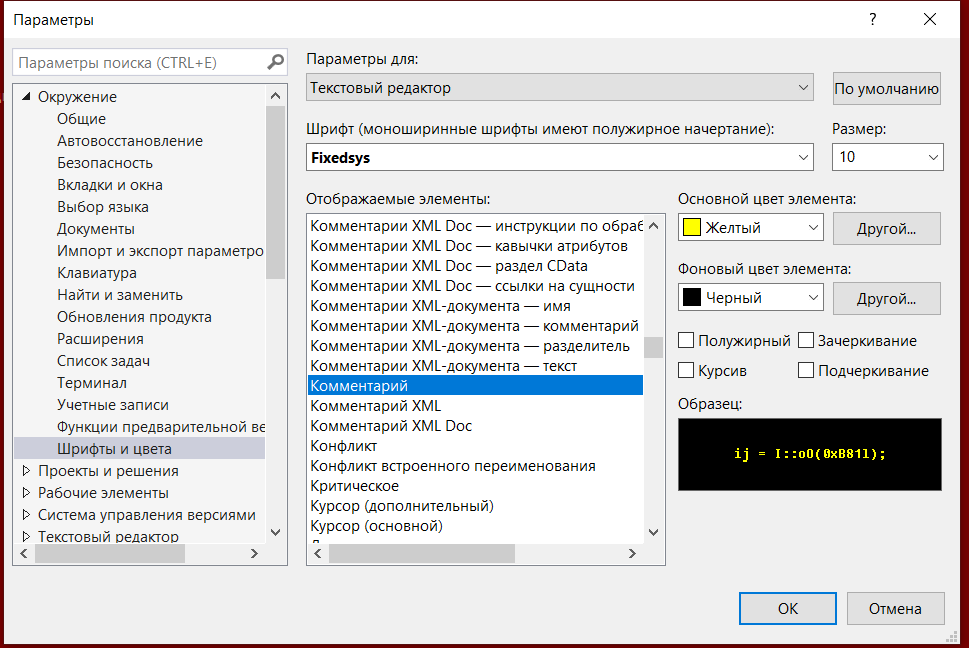


Рисунок 8 – Пример изменения стиля комментариев.

Для редактирования стиля комментариев проходим такой путь «Средства → Параметры → Окружение → Шрифты и цвета», но в «Параметры для» мы выбираем «Текстовый редактор», в «Отображаемые элементы» выбираем «Строка» и редактируем (Рисунок 9).

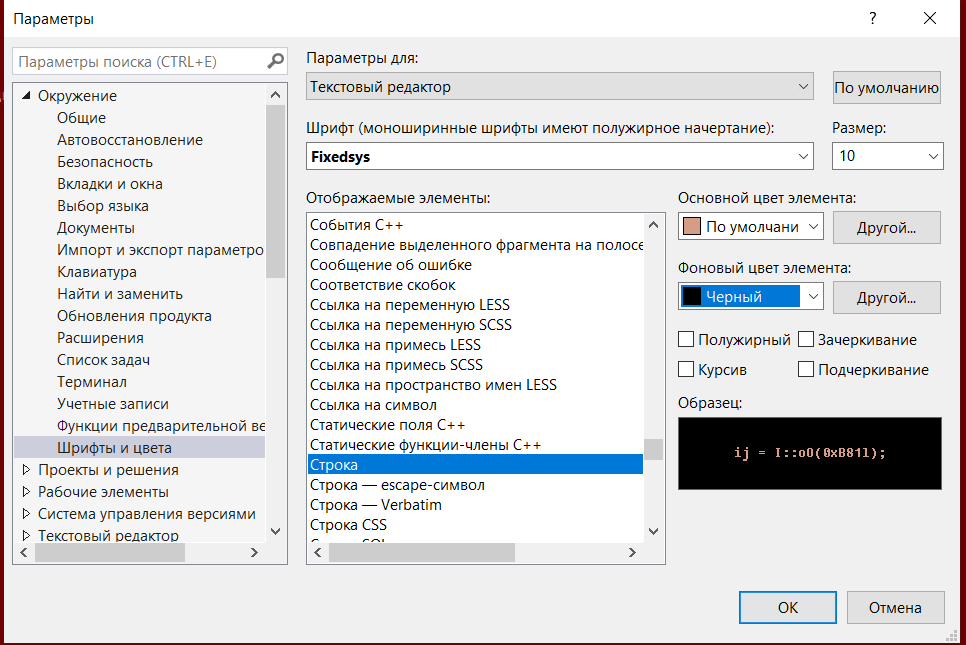


Рисунок 9 – Пример изменения стиля строковых литералов.

Для редактирования стиля чисел проходим такой путь «Средства → Параметры → Окружение → Шрифты и цвета», но в «Параметры для» мы выбираем «Текстовый редактор», в «Отображаемые элементы» выбираем «Число» и редактируем (Рисунок 10).

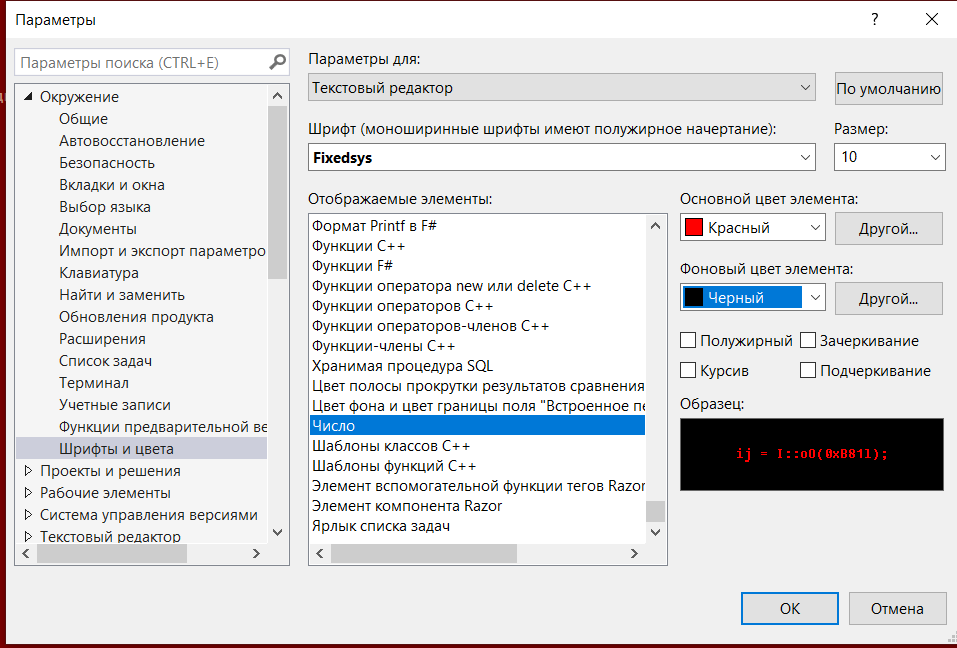


Рисунок 10 – Пример изменения стиля чисел.

* 1. Настройка меню и панели инструментов

Чтобы создать свою панель инструментов, необходимо проделать следующий путь «Средства → Настройка» и нажать на кнопку «Создать», должен получится такой же результат, как показано на рисунке (Рисунок 11).

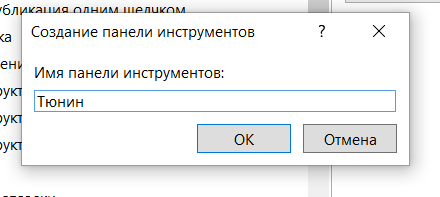


Рисунок 11 – Создание панели инструментов.

Чтобы добавить команды, перейдите в раздел «Команды», выбрать «Панель инструментов», выбрать имя вашей панели инструментов и нажать на кнопку «Добавить команду» (Рисунок 12).

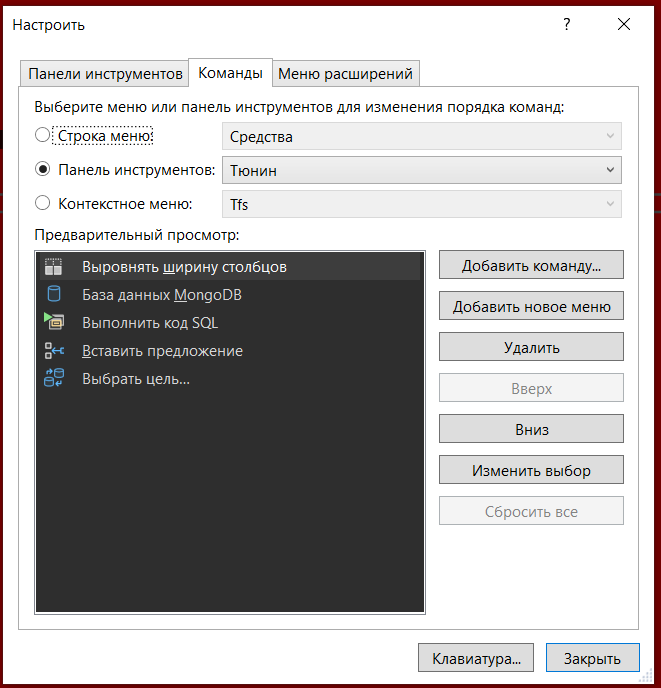


Рисунок 12 – Добавление команд.

Чтобы назначить горячие клавиши перейдите по данному пути «Средства → Параметры → Окружение → Клавиатура», выбрать в списке необходимые команды в меню «Показать команды, содержащие», ввести сочетание клавиш в «Введите сочетание клавиш» (Рисунок 13) и нажать на кнопку «Назначить» (Рисунок 14).

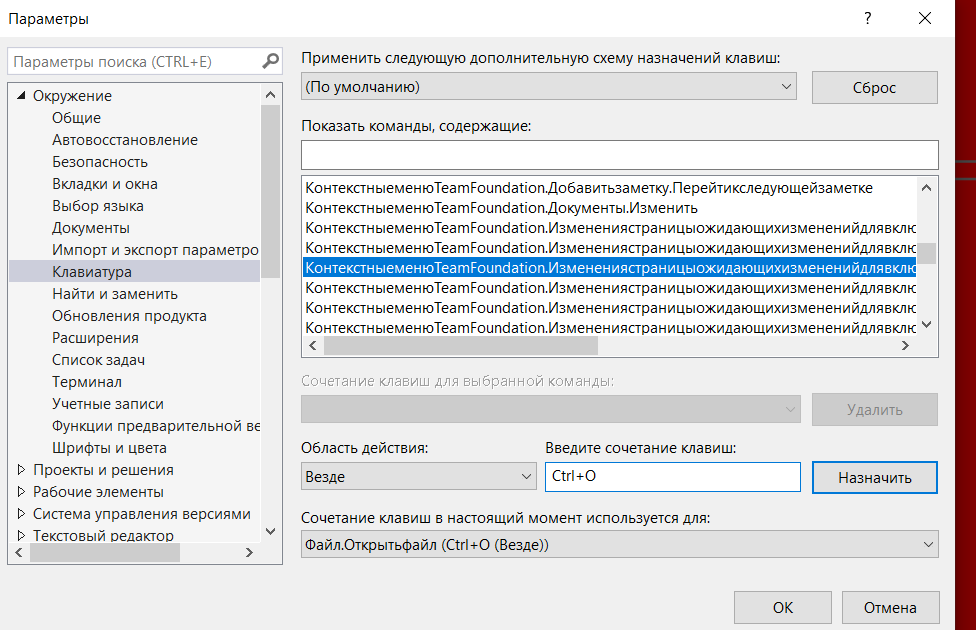


Рисунок 13 – Выбор команды и ввод сочетания клавиш.

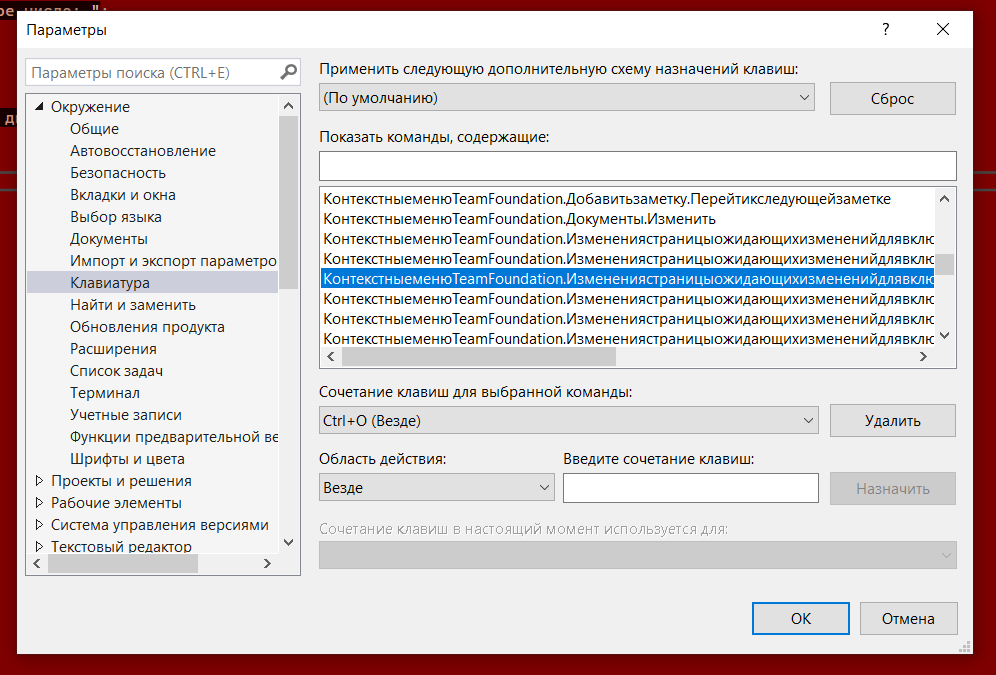


Рисунок 14 – Назначили сочетание клавиш.

* 1. Настройки макетов окон

Чтобы добавить новые окна в Visual Studio 2022 необходимо перейти в «Вид», выбрать «Другие окна», добавить необходимые окна и расположить в удобном Вам формате (Рисунок 15).

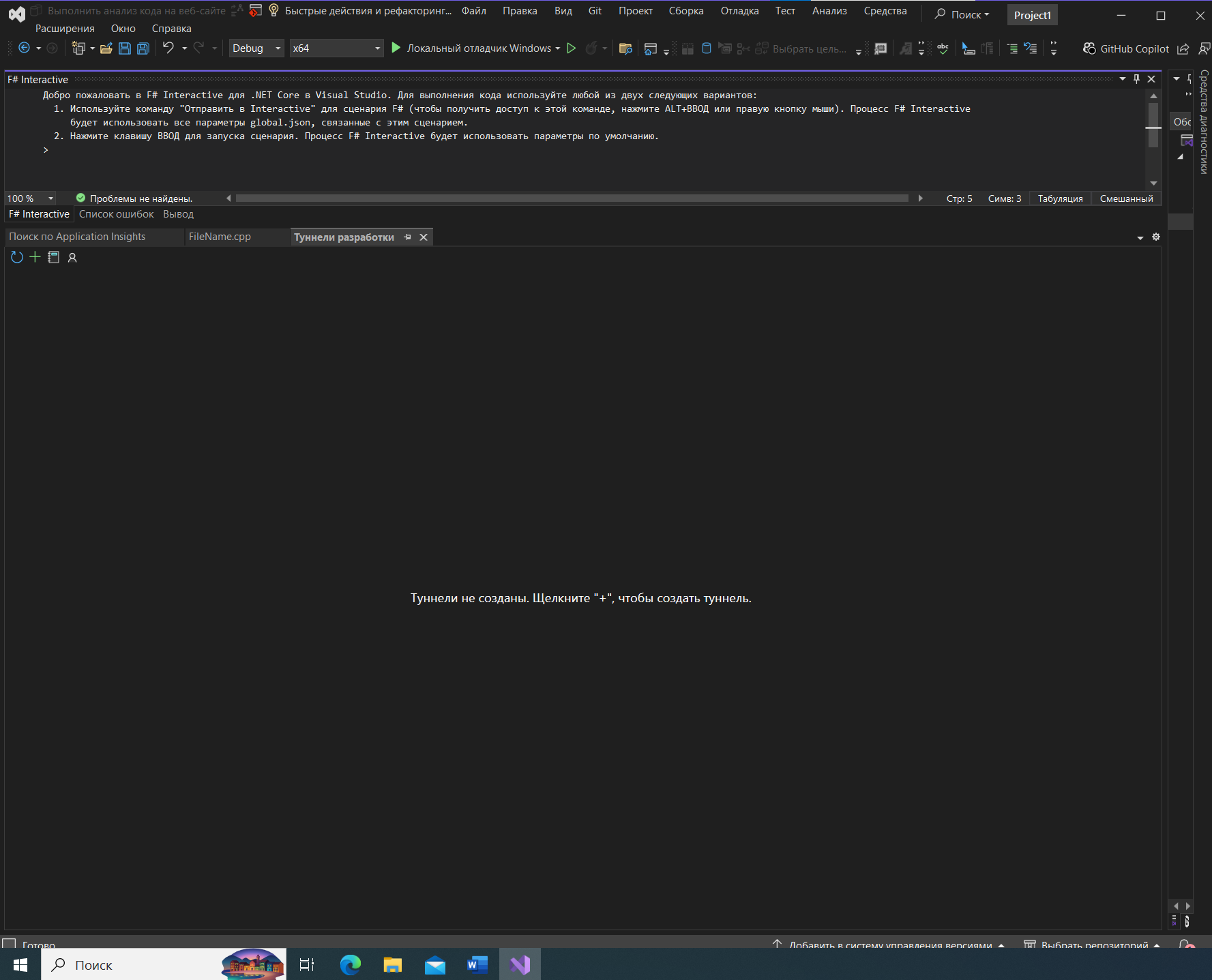


Рисунок 15 – Окна, которые добавил я.

Чтобы сохранить сей макет, нужно открыть вкладку «Окно» и там выбрать «Сохранить макет окна» (Рисунок 16).



Рисунок 16 – Сохранение макета окна.

Чтобы продемонстрировать переключение макета с одного на созданный нами, необходимо перейти в раздел «Окно» и выбрать «Сброс макета окна» (Рисунок 17). После данного действия вновь переходим во вкладку «Окно» и выбираем «Применить макет окна» (Рисунок 18).

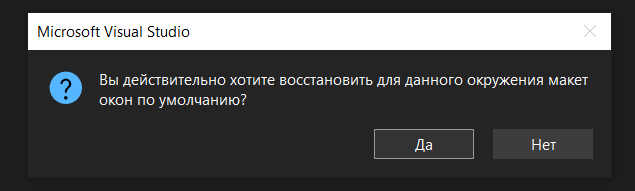


Рисунок 17 – Сброс макета окна.

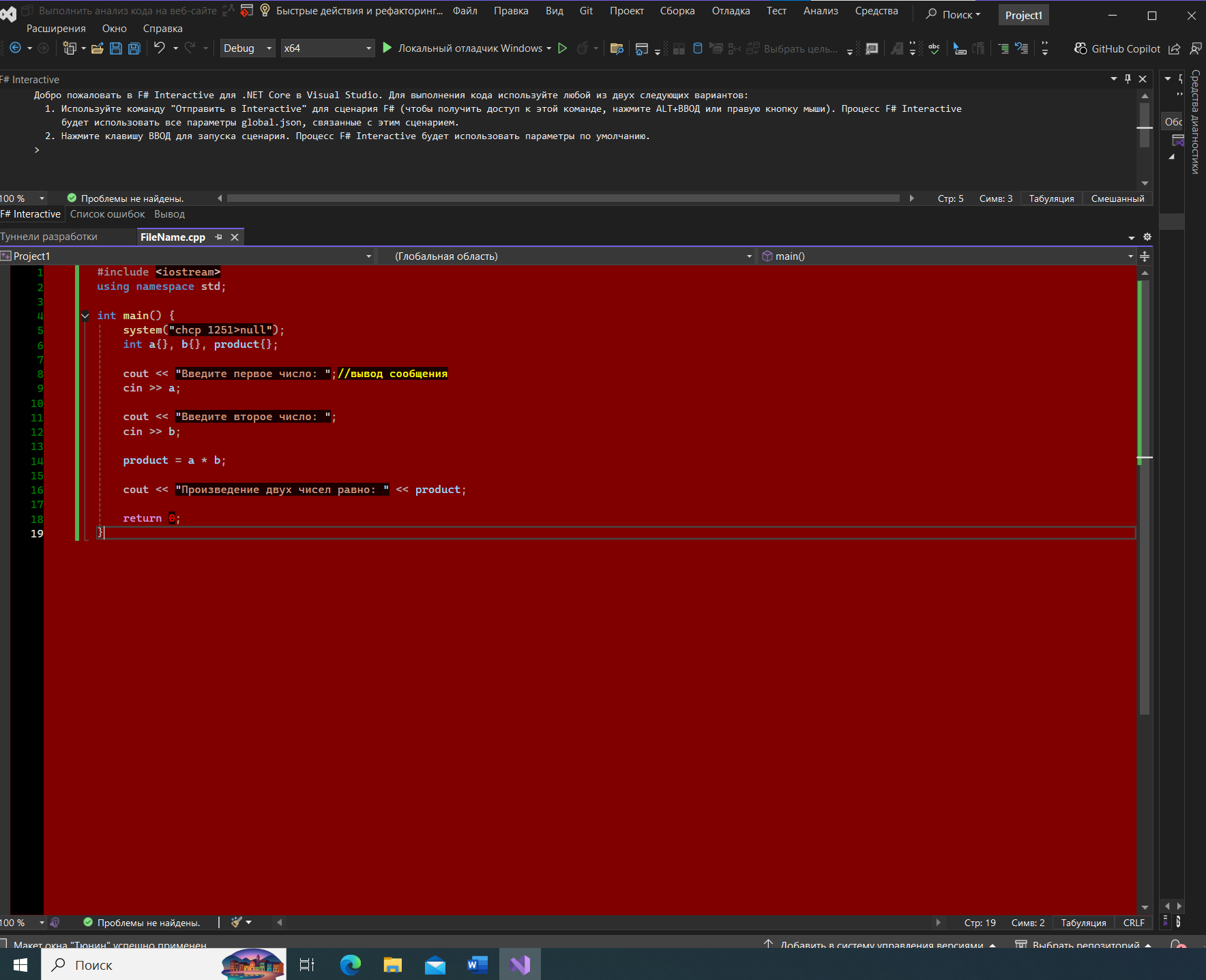


Рисунок 18 – Применили макет окна.

* 1. Экспорт настроек

Дабы экспортировать параметры проекта необходимо перейти по следующему пути «Средства → Импорт и экспорт параметров» (Рисунок 19), выбрать флаг «Экспортировать выбранные параметры окружения» и нажать кнопку «Далее». Выбираем параметры для экспорта и нажимаем кнопку «Далее» (Рисунок 20). Выбираем имя и место сохранения и нажимаем кнопку «Готово» (Рисунок 21).

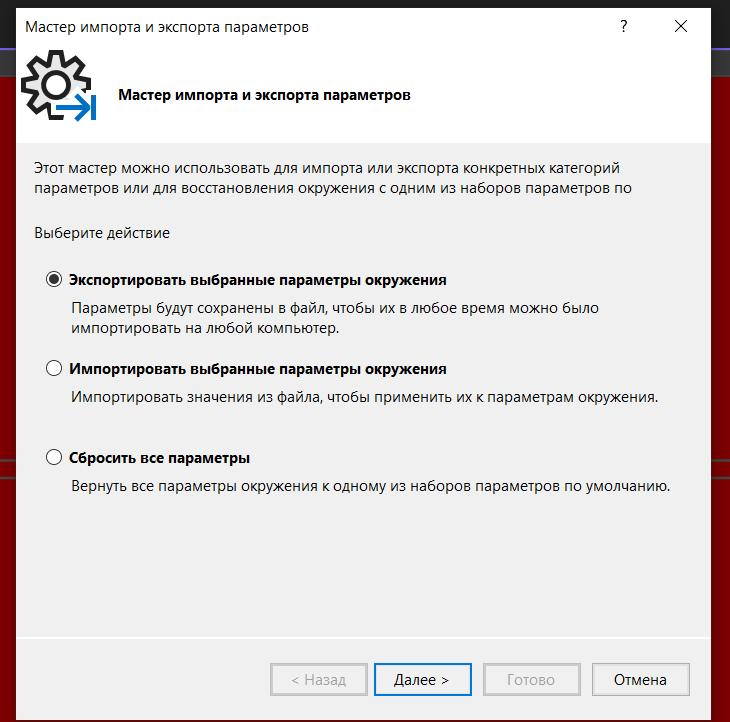


Рисунок 19 – Окно, которое должно появится.

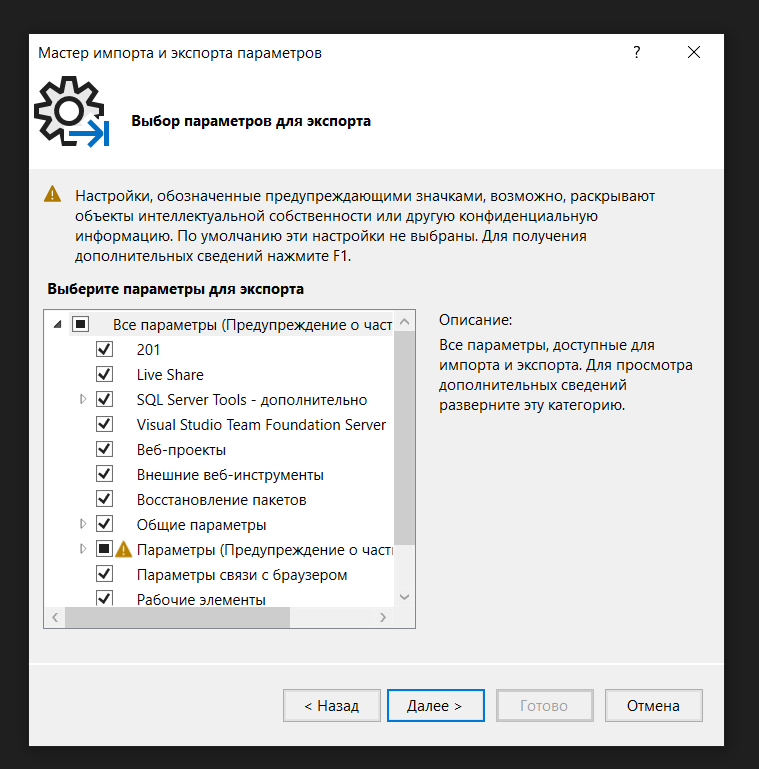


Рисунок 20 – Выбор параметров для экспорта.

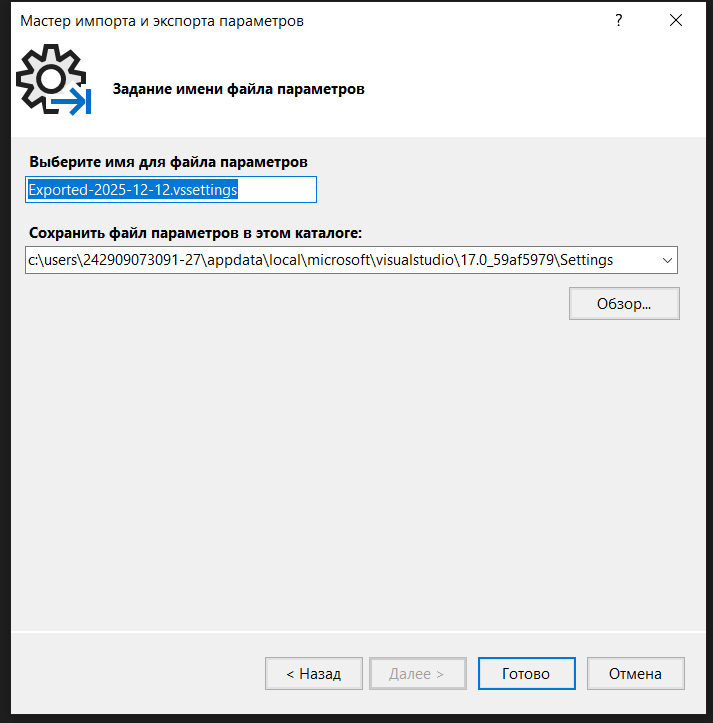


Рисунок 21 – Выбор имени и места сохранения файла.

Чтобы сбросить настройки, выберете флаг «Сбросить все параметры», нажмите кнопку «Далее» (Рисунок 22). Выберете флаг «Сбросить текущие параметры» и нажать «Далее» (Рисунок 23). Нажать ещё раз «Далее».

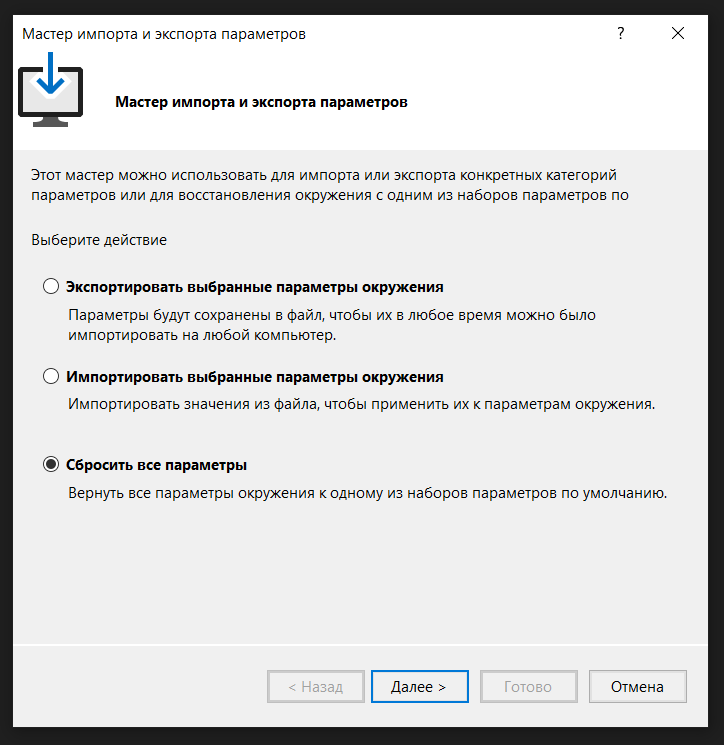


Рисунок 22 – Выбор флага «Сброс параметров».

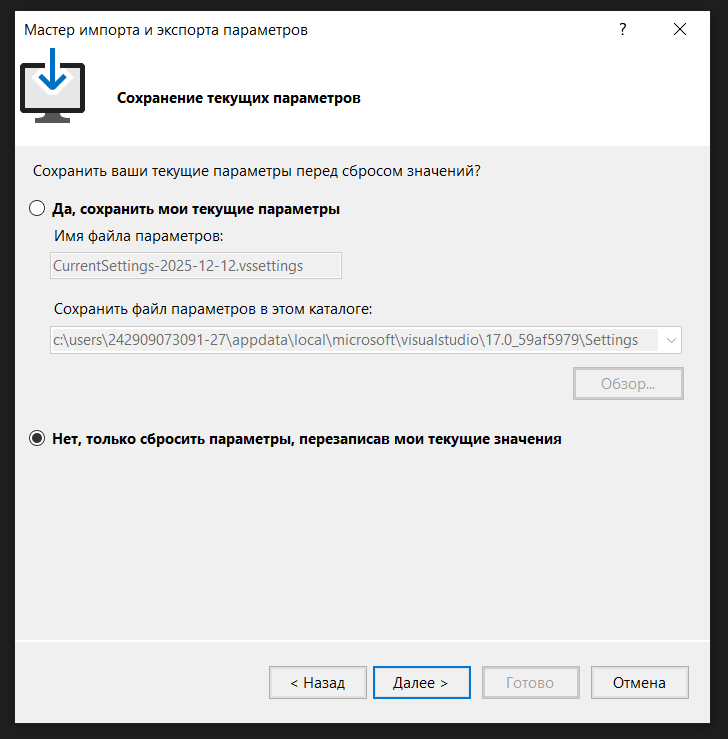


Рисунок 23 – Сброс параметров.

Чтобы импортировать, ранее экспортированные настройки в нашу среду программирования, необходимо перейти по следующему пути «Средства → Импорт и экспорт параметров» (См. Рисунок 19), выбрать флаг «Импортировать выбранные параметры окружения» и нажать кнопку «Далее». Выбираем наш ранее экспортированный файл и нажимаем кнопку «Далее» (Рисунок 24). Выбираем параметры для импорта и нажимаем кнопку «Готово» (Рисунок 25).

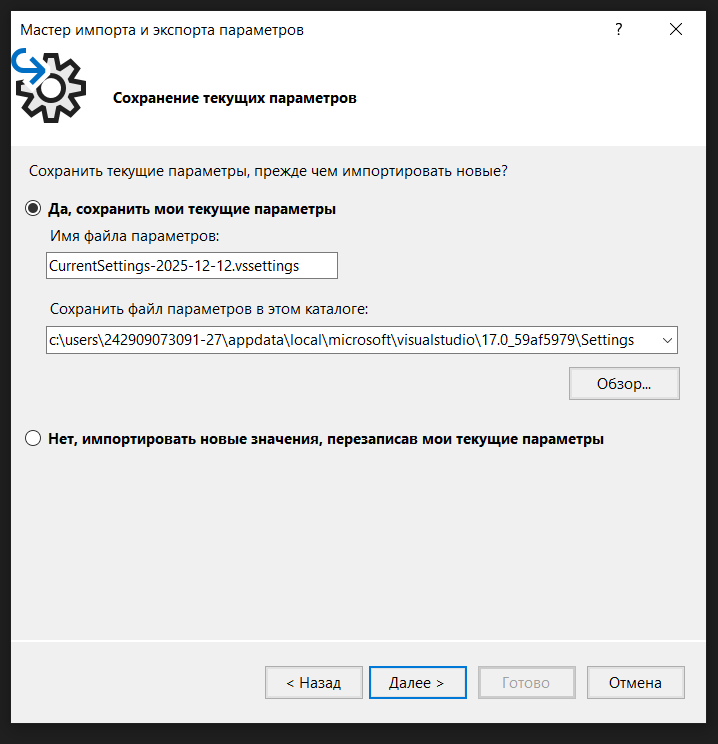


Рисунок 24 – Выбор файла для импорта.

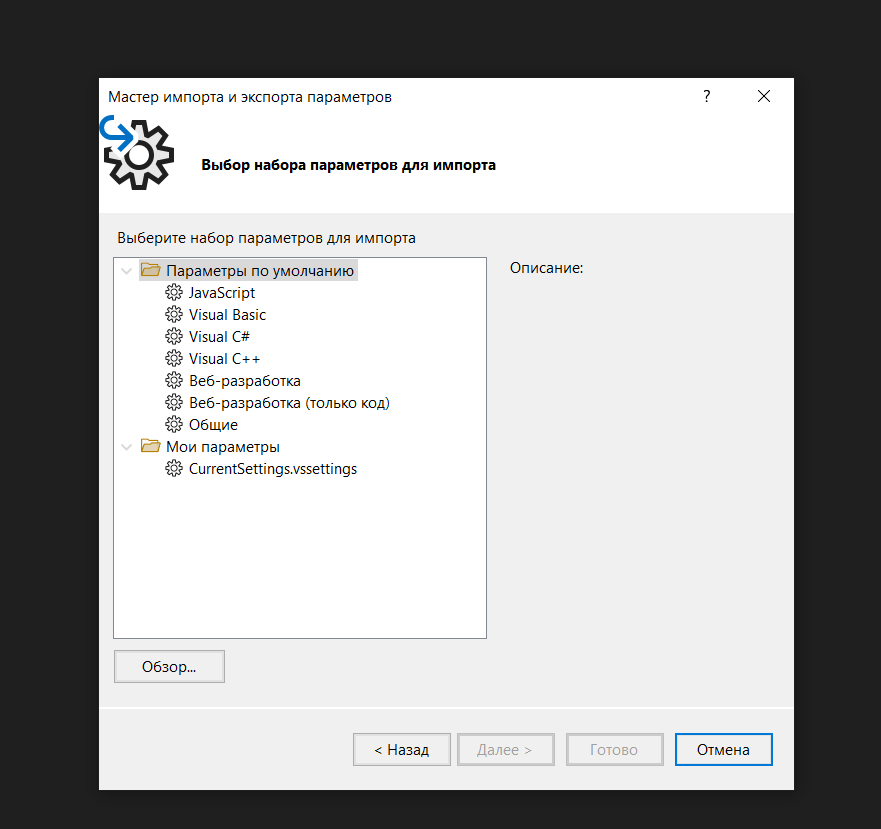
.

Рисунок 25 – Выбор параметров для импорта.

**Код задания:**

include <iostream>

using namespace std;

int main() {

system("chcp 1251>null");

int a{}, b{}, product{};

cout << "Введите первое число: ";//вывод сообщения

cin >> a;

cout << "Введите второе число: ";

cin >> b;

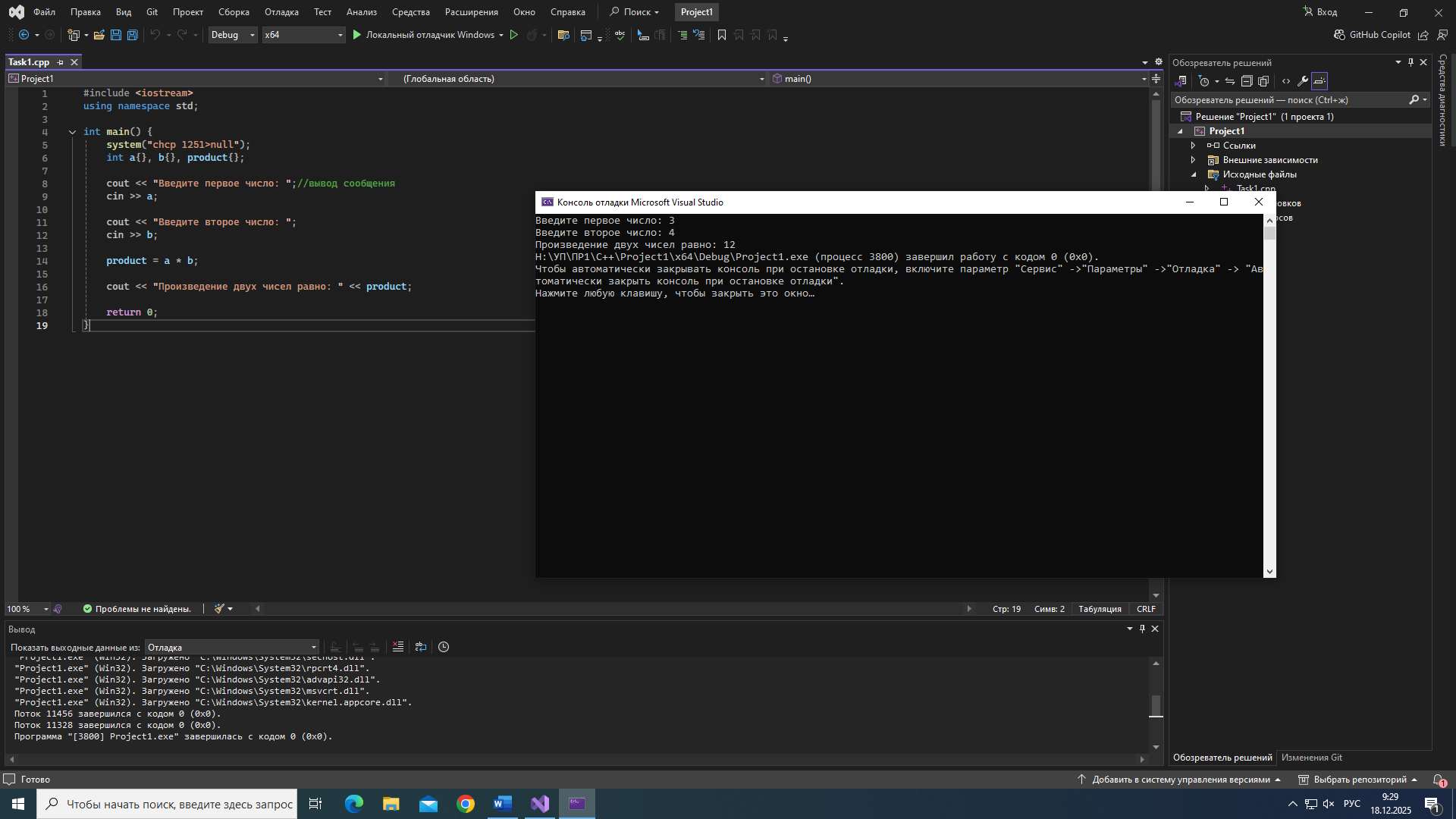
product = a \* b;

cout << "Произведение двух чисел равно: " << product;

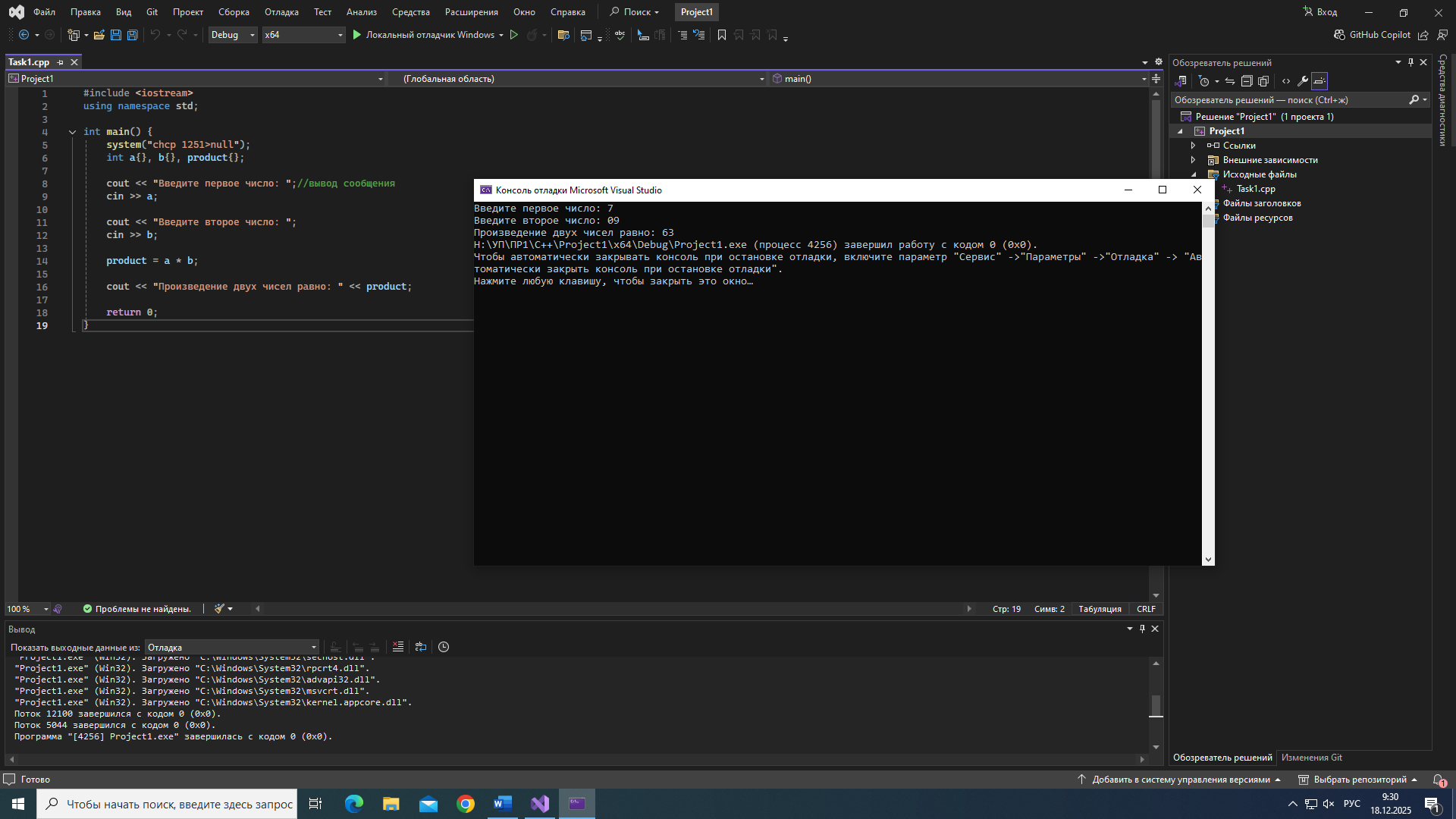
return 0;

}

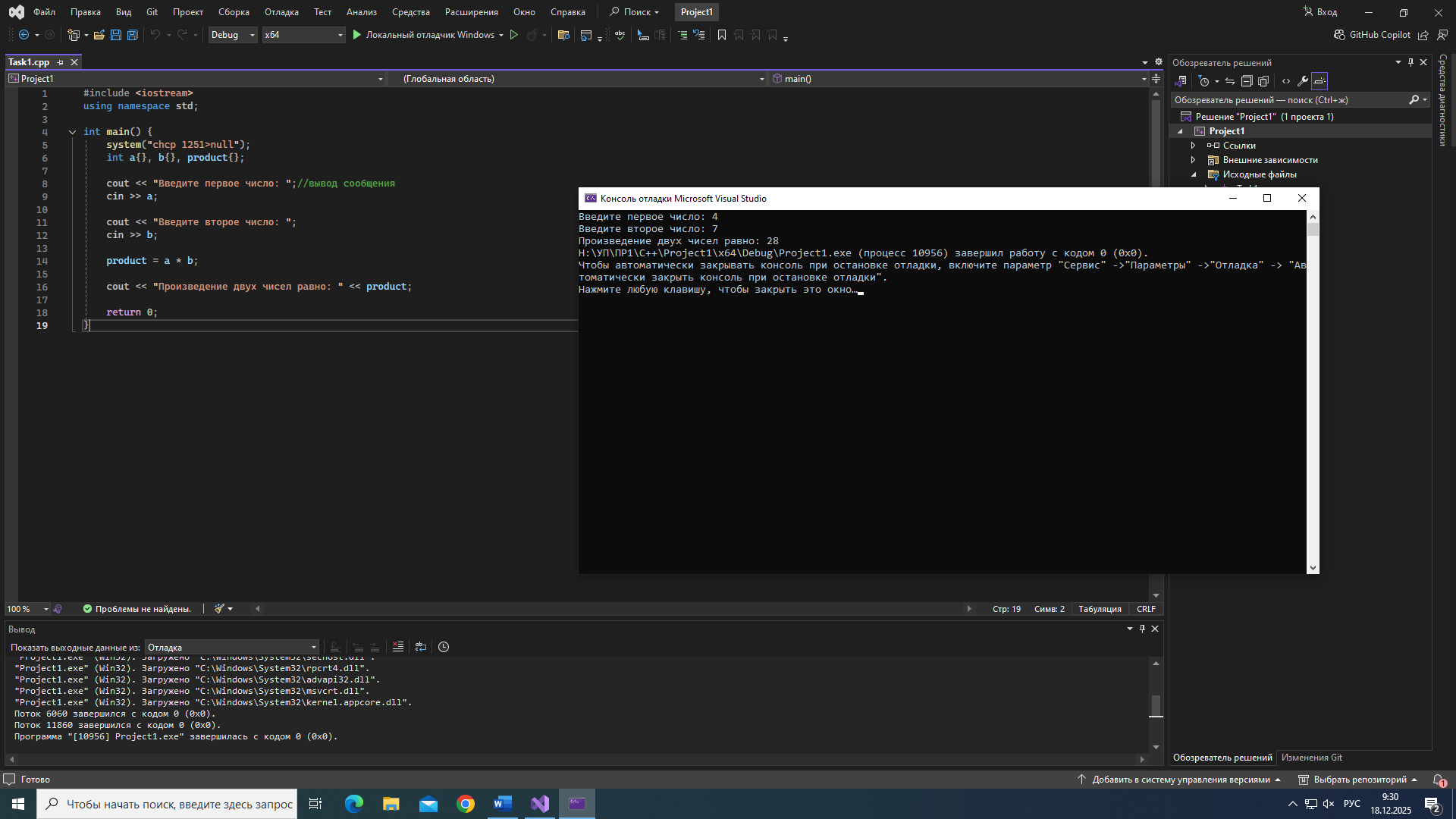
**Скриншоты работы программы:**



Скриншот 1 – Запуск 1.



Скриншот 2 – Запуск 2.



Скриншот 3 – Запуск 3.

1. ПРАКТИЧЕСКАЯ РАБОТА №2 «ОТЛАДКА В VISUAL STUDIO»

Цель: освоить базовые и продвинутые техники отладки.

* 1. Навигация по коду с помощью отладчика

Дабы установить точку остановы на строке с циклом, нам необходимо навести мышь на самую левую панель и там совершить один клик мышью (Рисунок 26).

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

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 26 – Добавление точки остановы.

Чтобы запустить отладку, достаточно просто нажать клавишу F5 и у Вас запустится программа, как показано на рисунке (Рисунок 27).

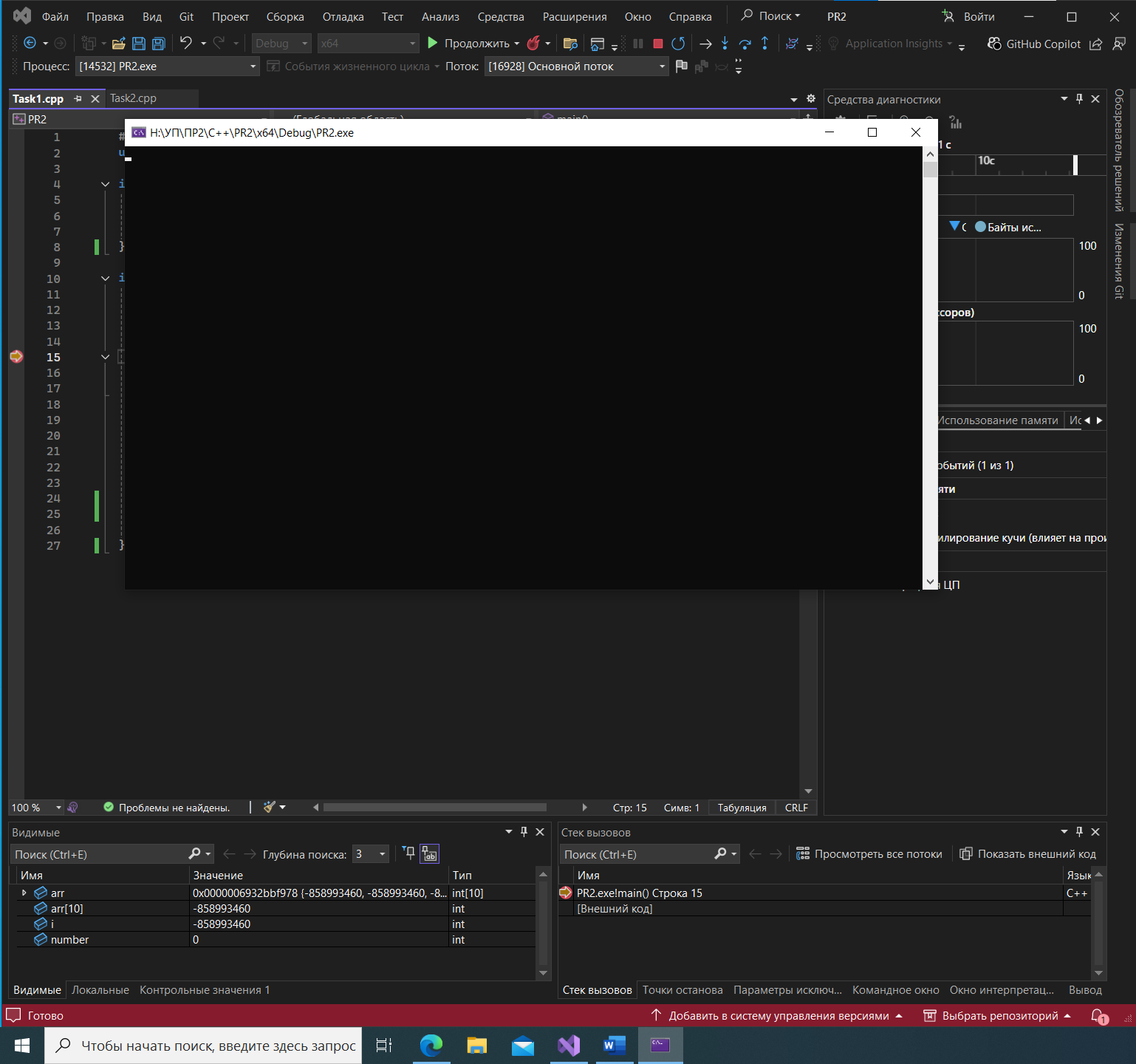


Рисунок 27 – Запуск отладки посредством нажатия на клавишу F5.

Дабы рассмотреть работу шагов отладки, нам необходимо поставить точку остановы в коде самой программы, где Вам это удобнее, чтобы пронаблюдать за их работой (желательно там, где находится вызов функции) (Рисунок 28).

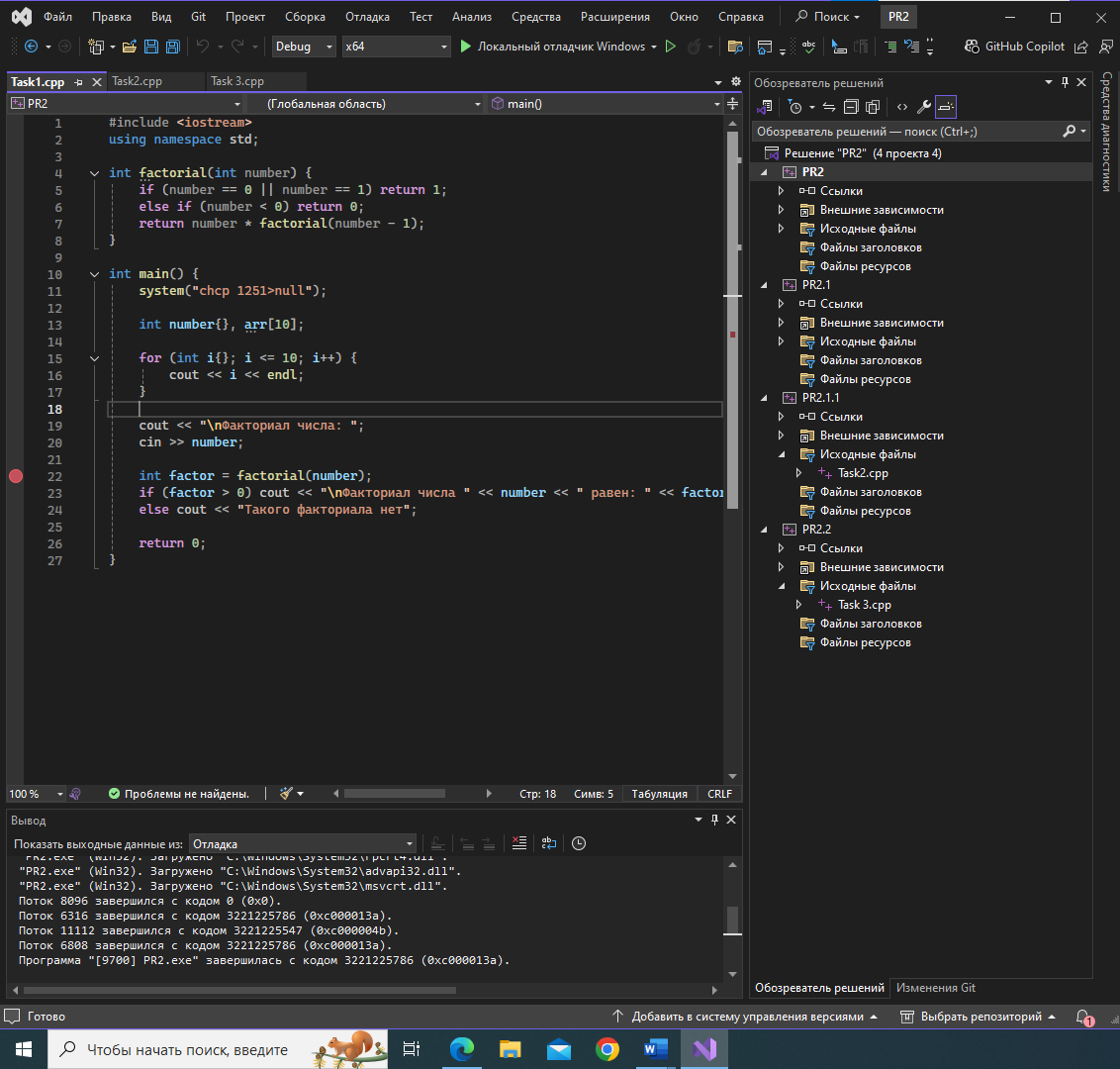


Рисунок 28 – Пример постановки точки остановы.

Чтобы пронаблюдать за первым из них, запустите отладку (F5) и как только ваш код дойдёт до точки остановы, попробуйте нажать F10. Данный ход отладки называется «Шаг с обходом» и предназначен для хода по строкам Вашего кода (построчной работы кода), не вдаваясь в особые подробности, в те же функции, например. Под идеей «не вдаваясь в подробности» я подразумеваю, что ход не будет проходить по коду вызываемой внешней функции, а просто выведет её итоговое значение (Рисунок 29).

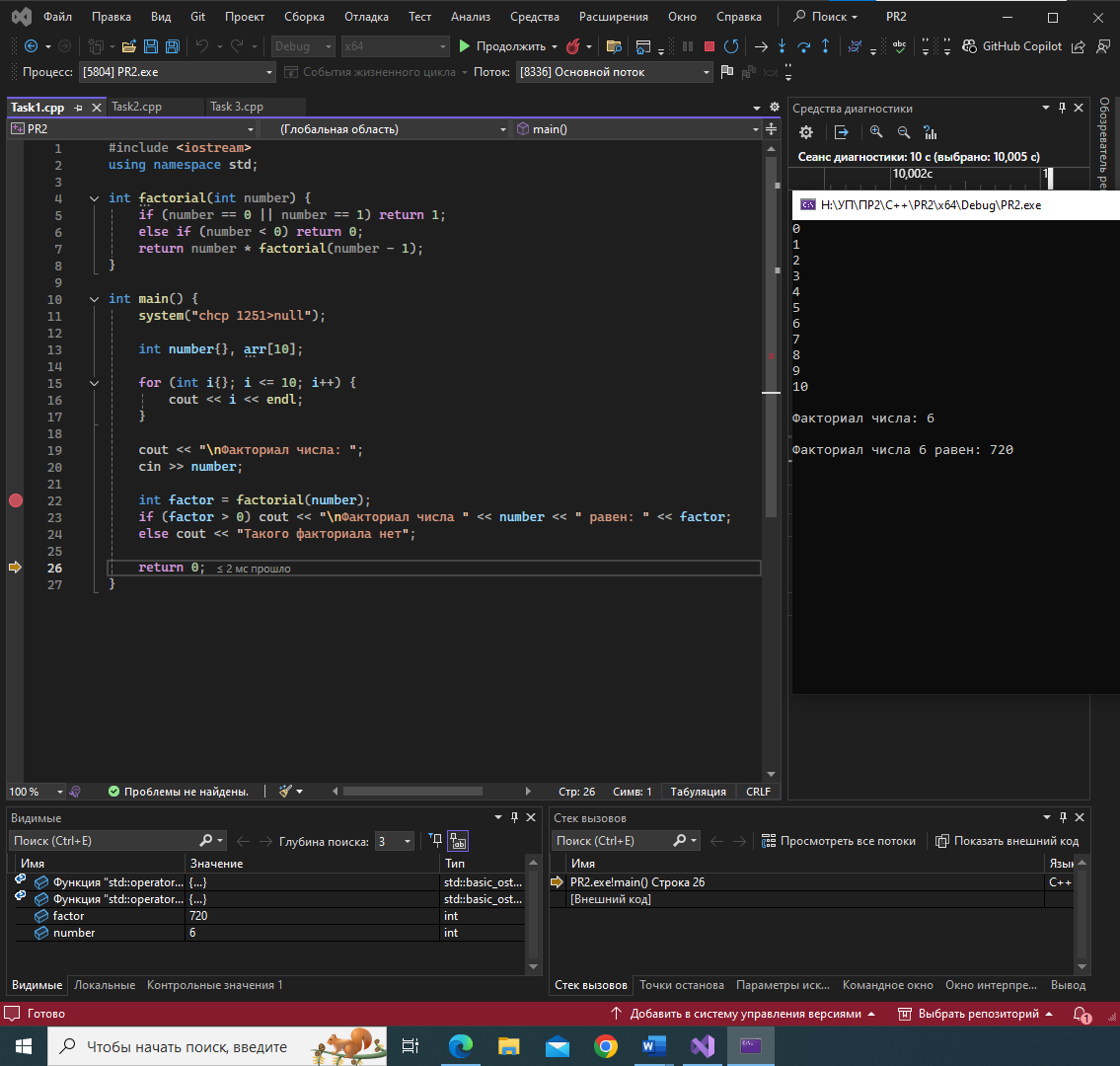


Рисунок 29 – Пример работы шага отладки «Шаг с обходом» (F10).

Чтобы пронаблюдать за вторым из них, запустите отладку (F5) и как только ваш код дойдёт до точки остановы, нажмите F11. Данный ход отладки называется «Шаг с заходом». Он выполняется также построчно, но уже вдаваясь в подробности внешних, вызываемых функций, пристально следя за тем, что делает сейчас программа. (Рисунок 30).

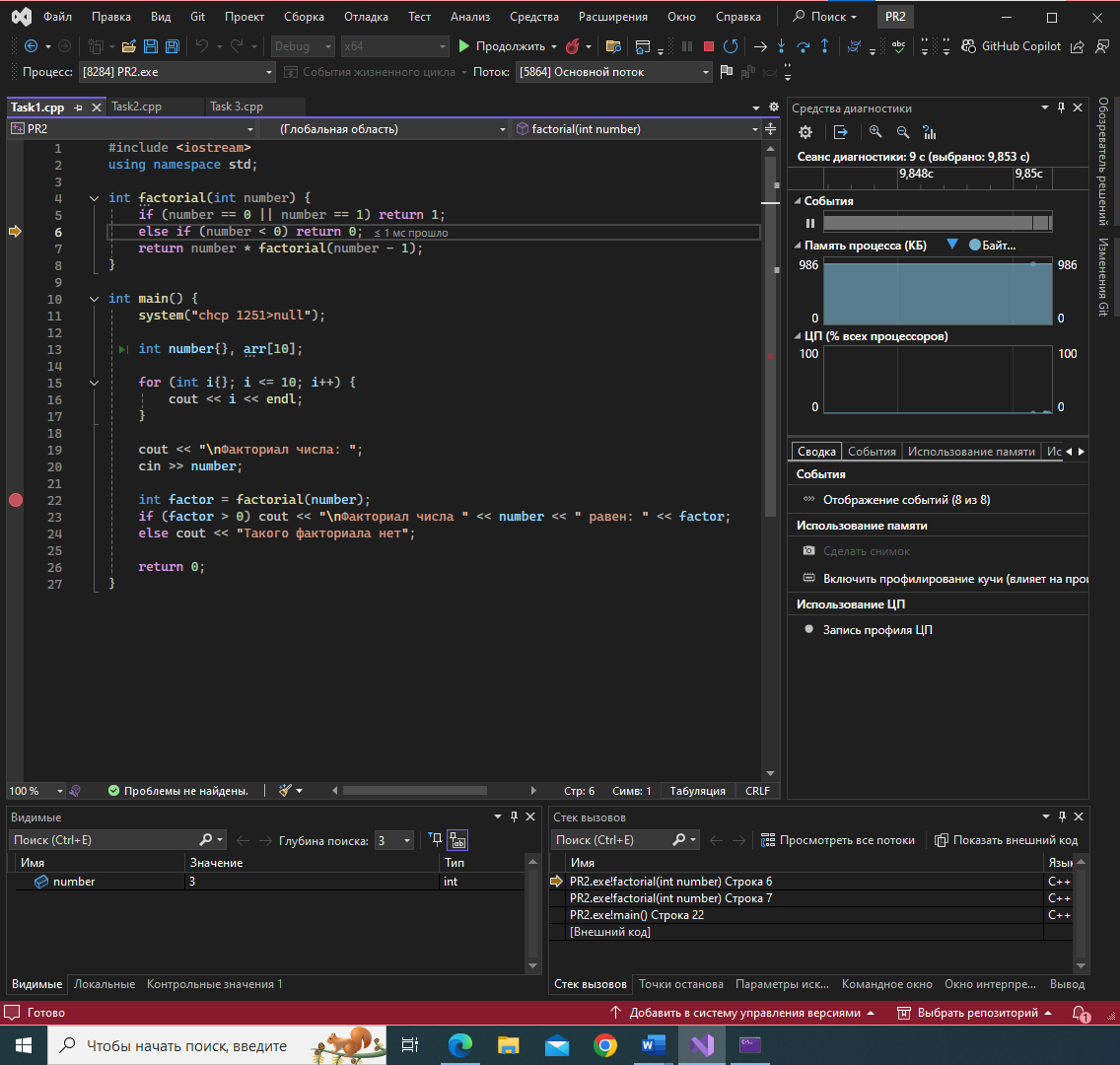


Рисунок 30 – Пример работы шага отладки «Шаг с заходом» (F11).

Чтобы пронаблюдать за вторым из них, запустите отладку (F5) и как только ваш код дойдёт до точки остановы, нажмите сочетание клавиш Shift + F11. Данный ход отладки называется «Шаг с выходом». Он отменяет построчный просмотр кода и возвращает его в обычный вид (Рисунок 31).

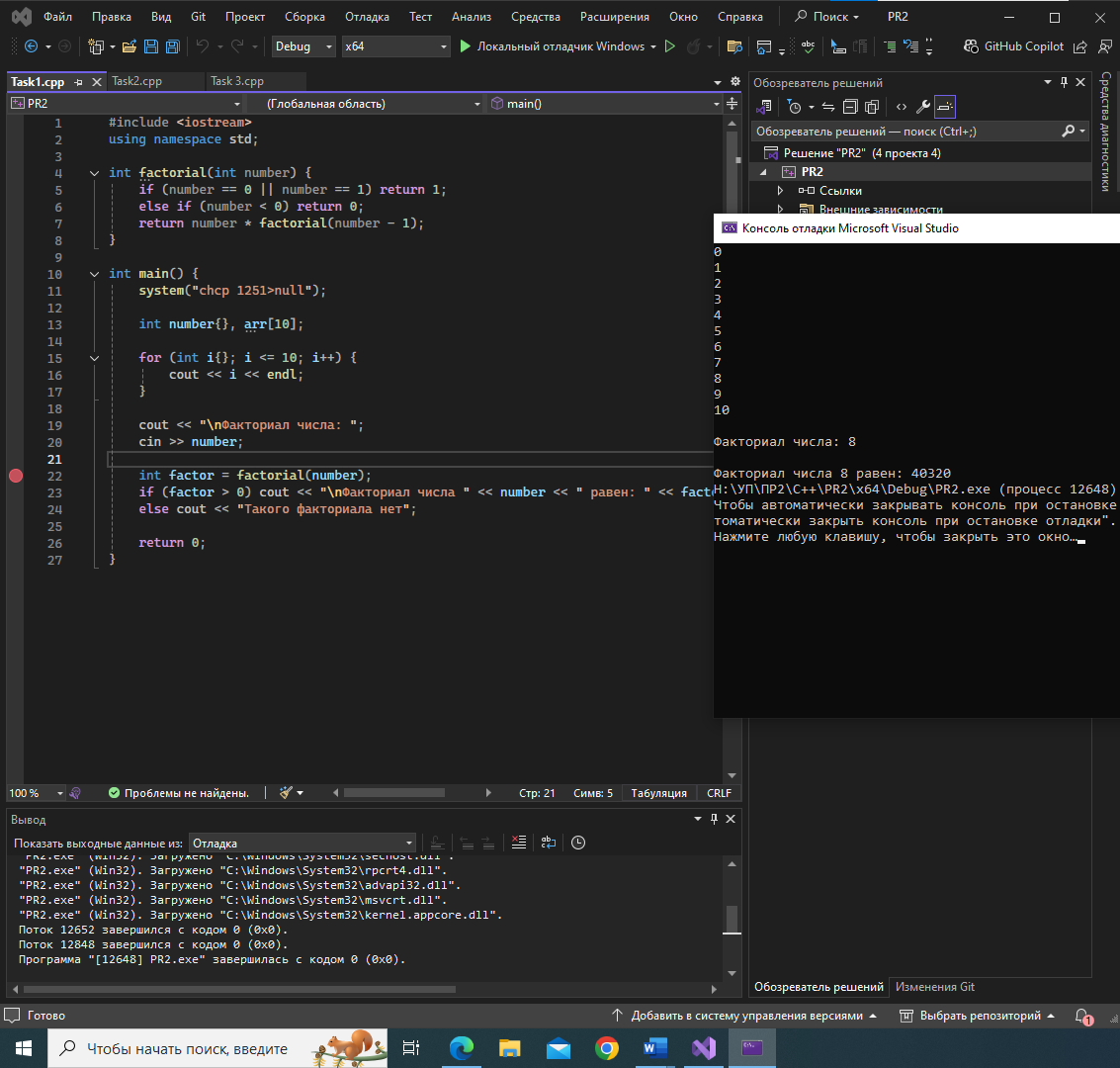


Рисунок 31 – Пример работы шага отладки «Шаг с выходом» (Shift + F11).

Чтобы установить условную точку остановы необходимо кликнуть туда же, где и устанавливаются обычные точки остановы, но правой кнопкой мыши и выбрать «Вставить условную точку остановы». Там для примера можно выбрать условие остановки при i == 5 (Рисунок 32).

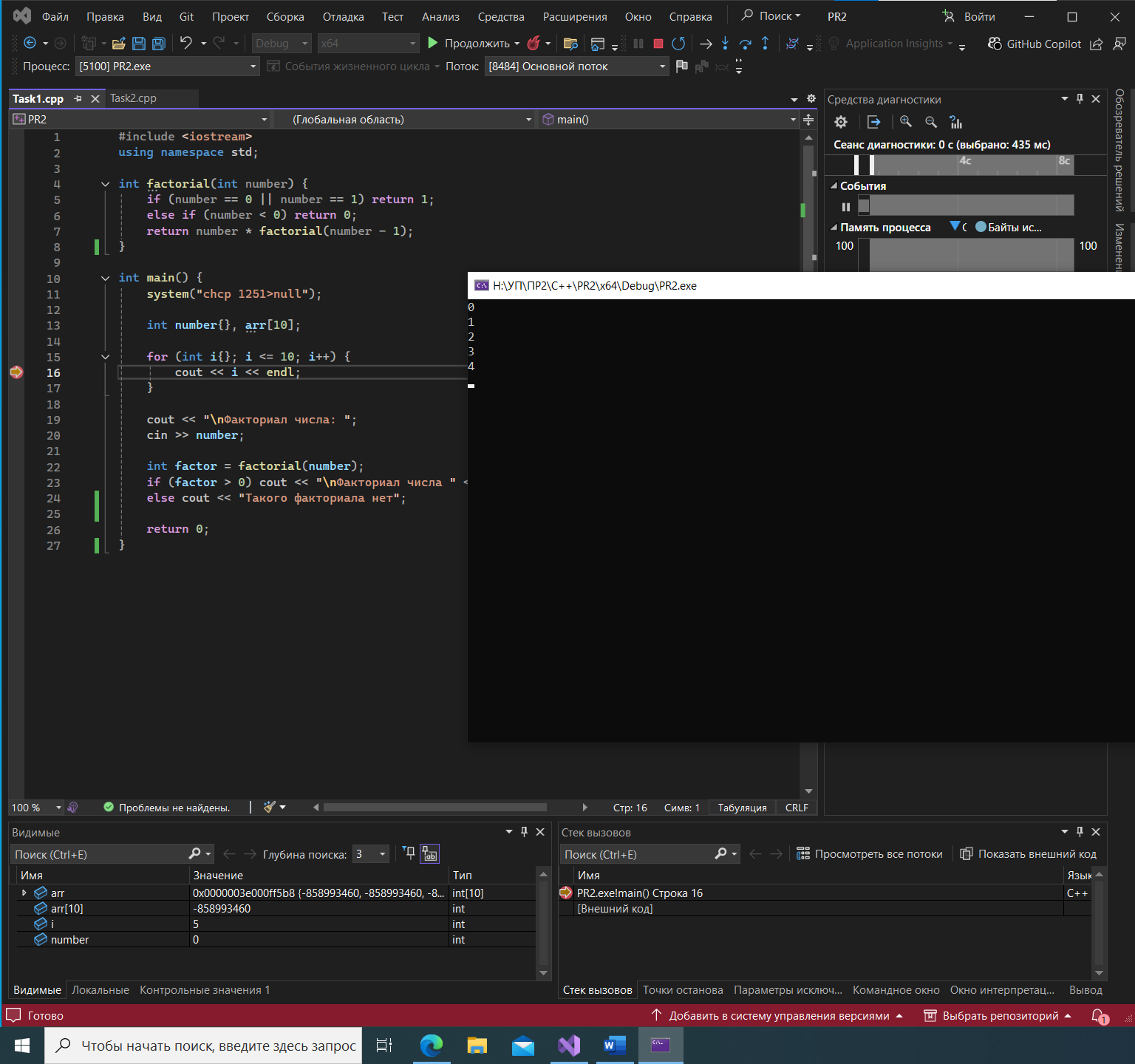


Рисунок 32 – Пример установки условной точки остановы.

Чтобы самостоятельно проследить, как меняется та или иная переменная, мы должны запустить отладку (F5), предварительно установив точку остановы, и открыть Visual Studio. Внизу будут окна, на которых можно следить за значениями переменных. Начнём с «Видимые» это переменные, которые участвуют прямо сейчас и те, что участвовали или были объявлены 2-3 строки назад (Рисунок 33).

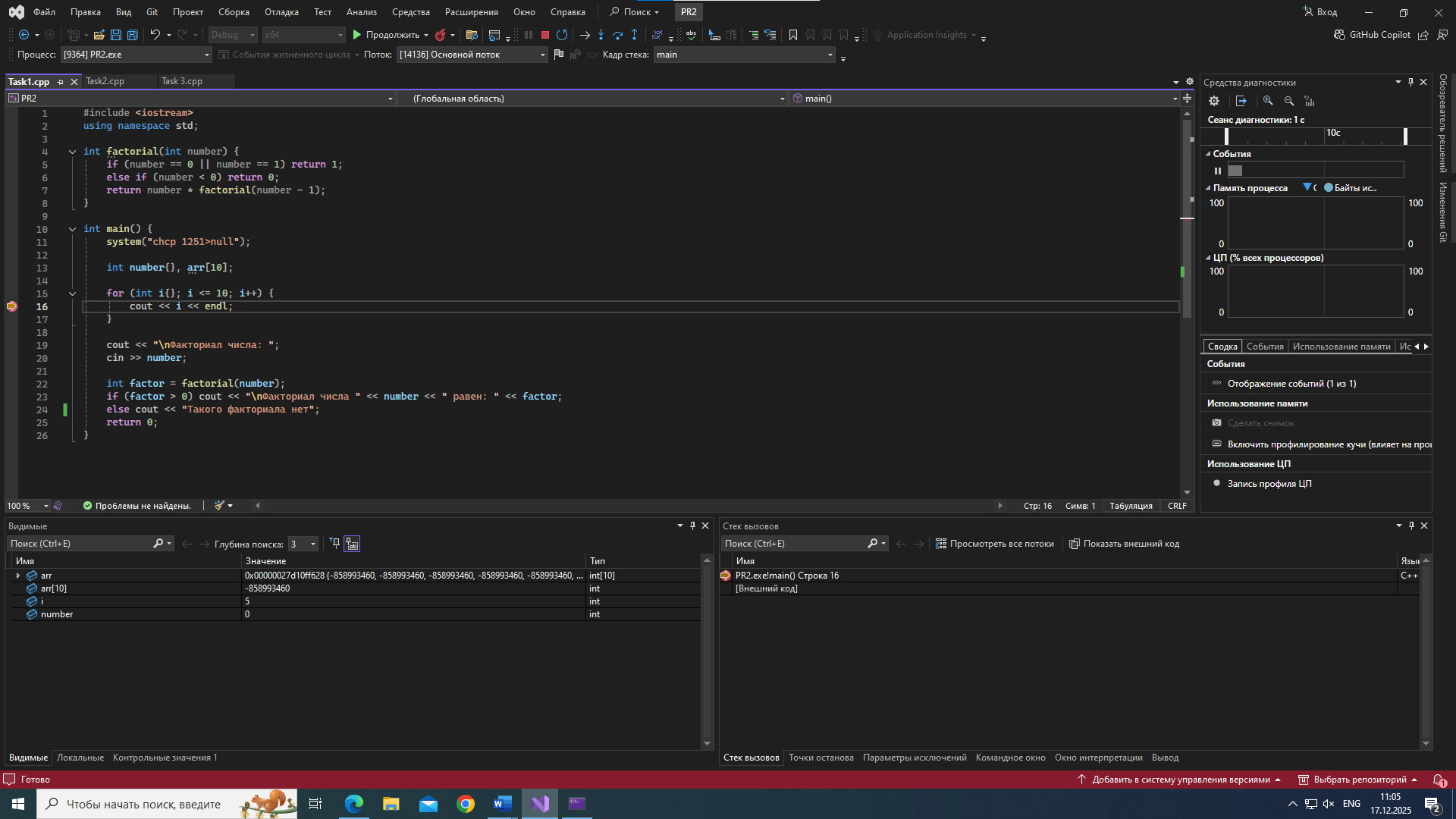


Рисунок 33 – Значения в панели «Видимые».

Во вкладке «Локальные», мы наблюдаем все переменные, которые доступны нам в данной области видимости (Рисунок 34).

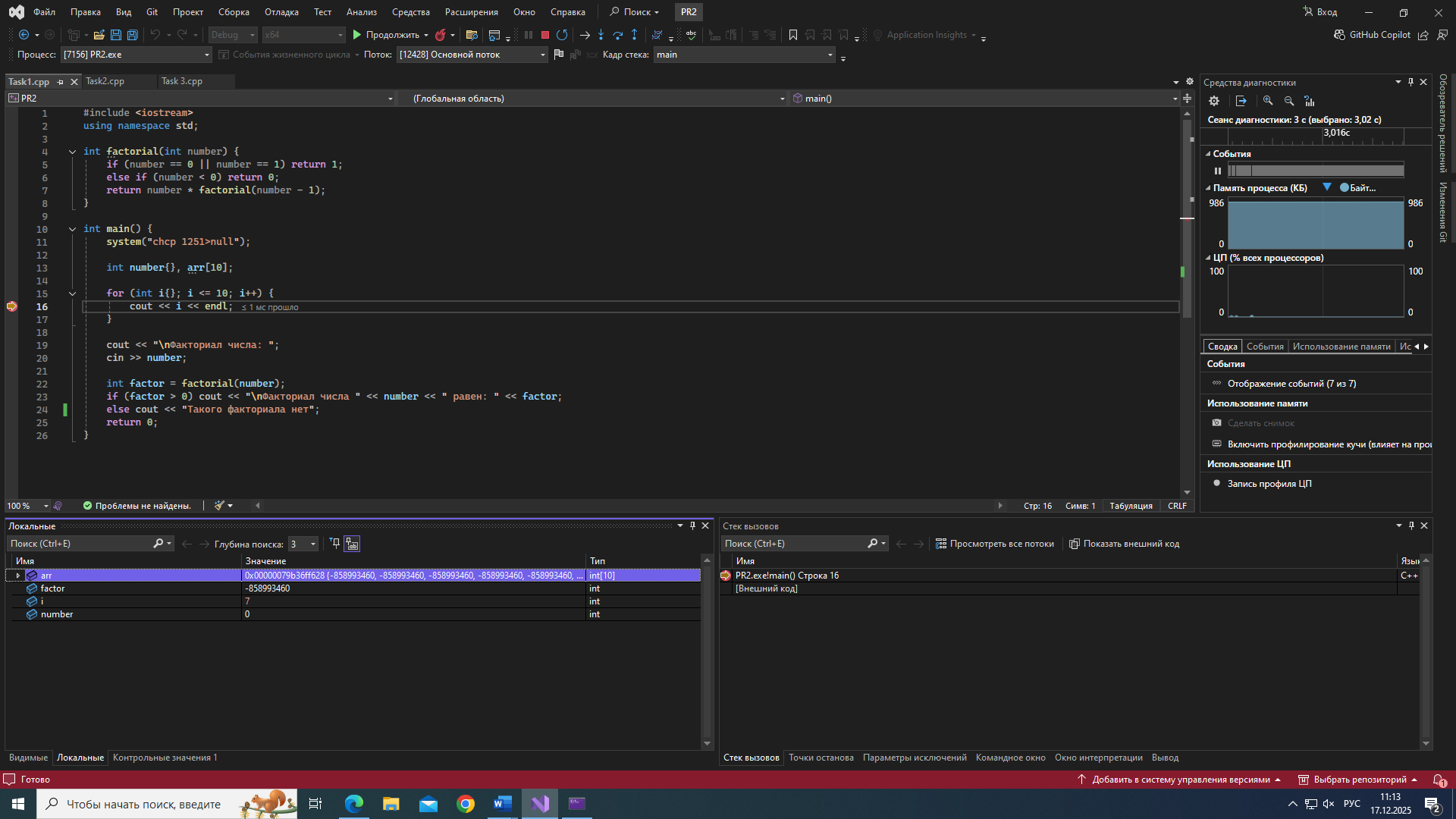


Рисунок 34 – Значения в панели «Локальные».

Во вкладке «Контрольные значения», мы можем наблюдать не за всеми, а только «избранными» переменными, которые нам необходимо отслеживать по заданию (Рисунок 35).

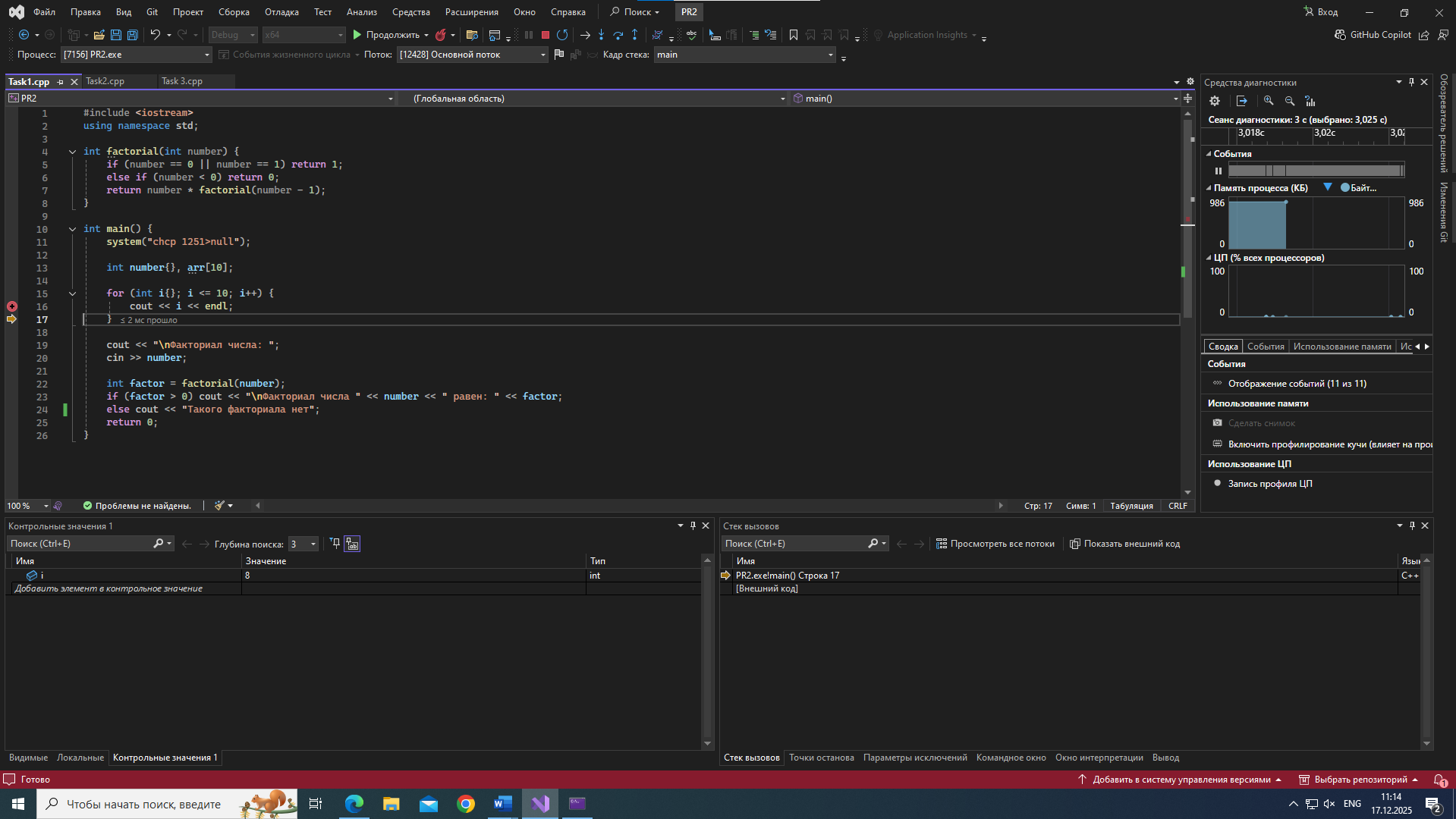


Рисунок 35 – Значения в панели «Контрольные значения».

Чтобы перейти во вкладку «Quick Watch», необходимо выбрать переменную в коде и на ней кликнуть право кнопкой мыши после чего выбрать «Быстрая проверка». В данном меню мы можем добавить не только выбранную переменные, но и выражения в контрольные значения. (Рисунок 36).

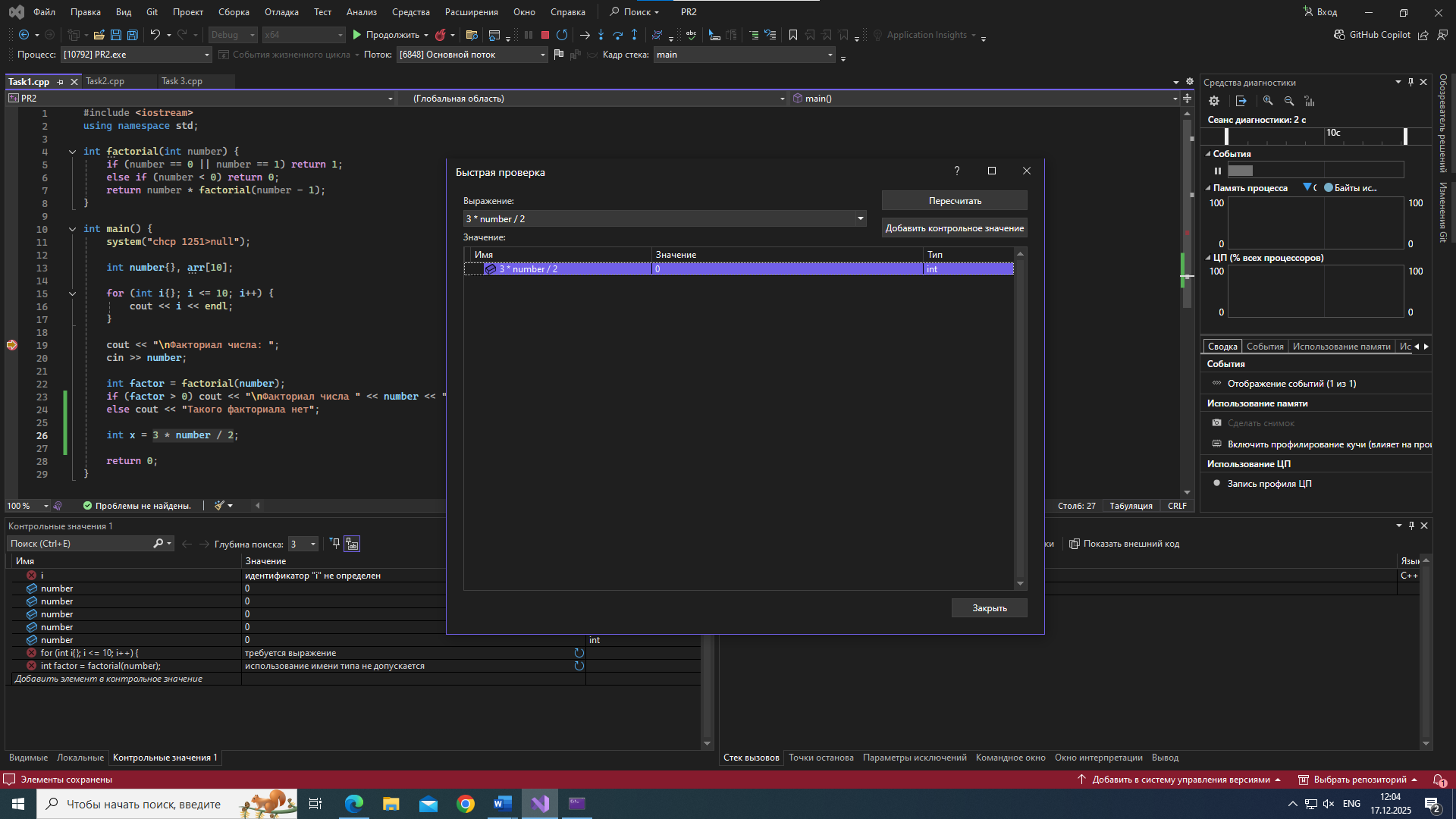


Рисунок 36 – Значения в панели «Quick Watch».

**Код задания:**

#include <iostream>

using namespace std;

int factorial(int number) {

if (number == 0 || number == 1) return 1;

else if (number < 0) return 0;

return number \* factorial(number - 1);

}

int main() {

system("chcp 1251>null");

int number{}, arr[10];

for (int i{}; i <= 10; i++) {

cout << i << endl;

}

cout << "\nФакториал числа: ";

cin >> number;

int factor = factorial(number);

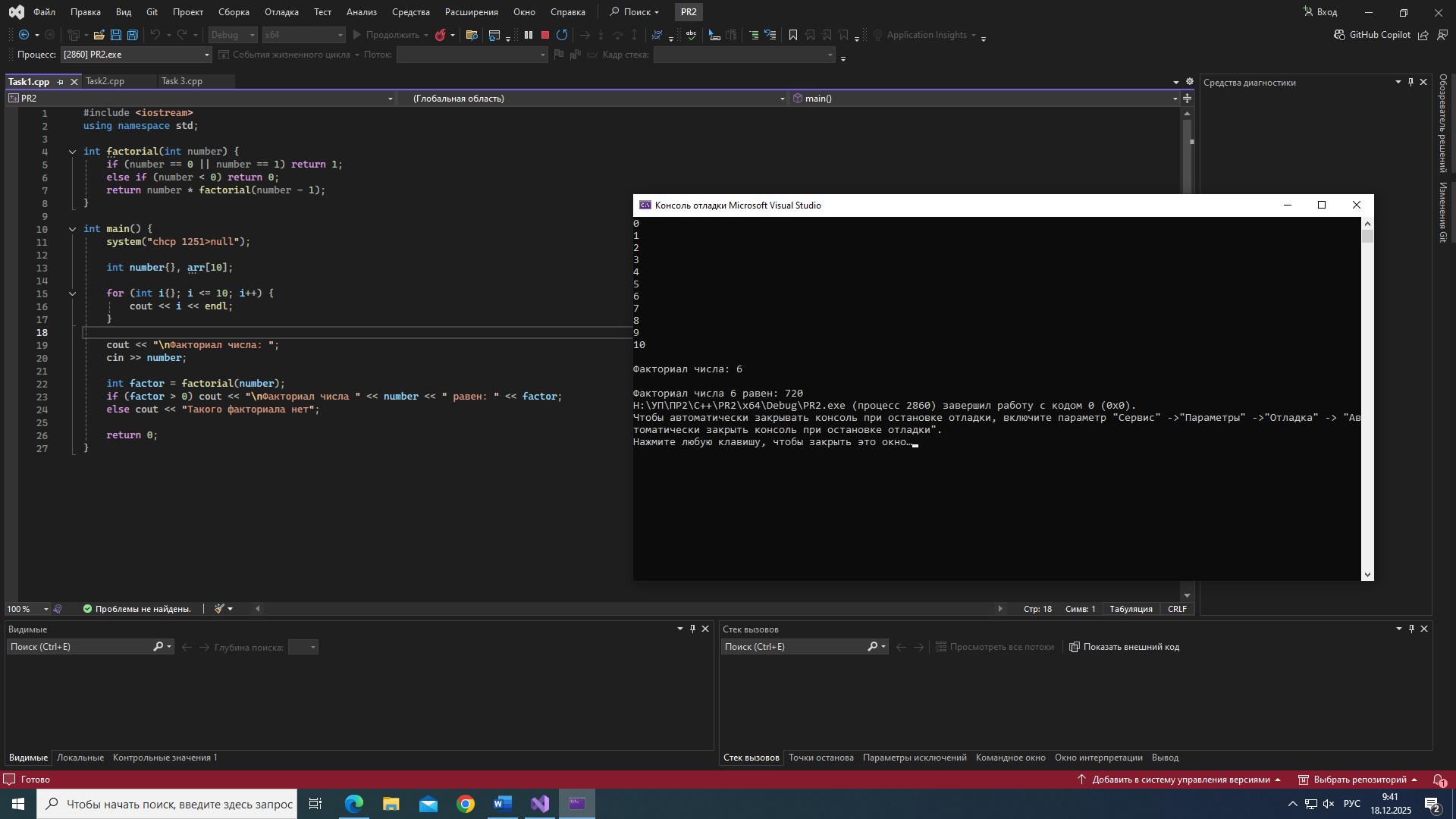
if (factor > 0) cout << "\nФакториал числа " << number << " равен: " << factor;

else cout << "Такого факториала нет";

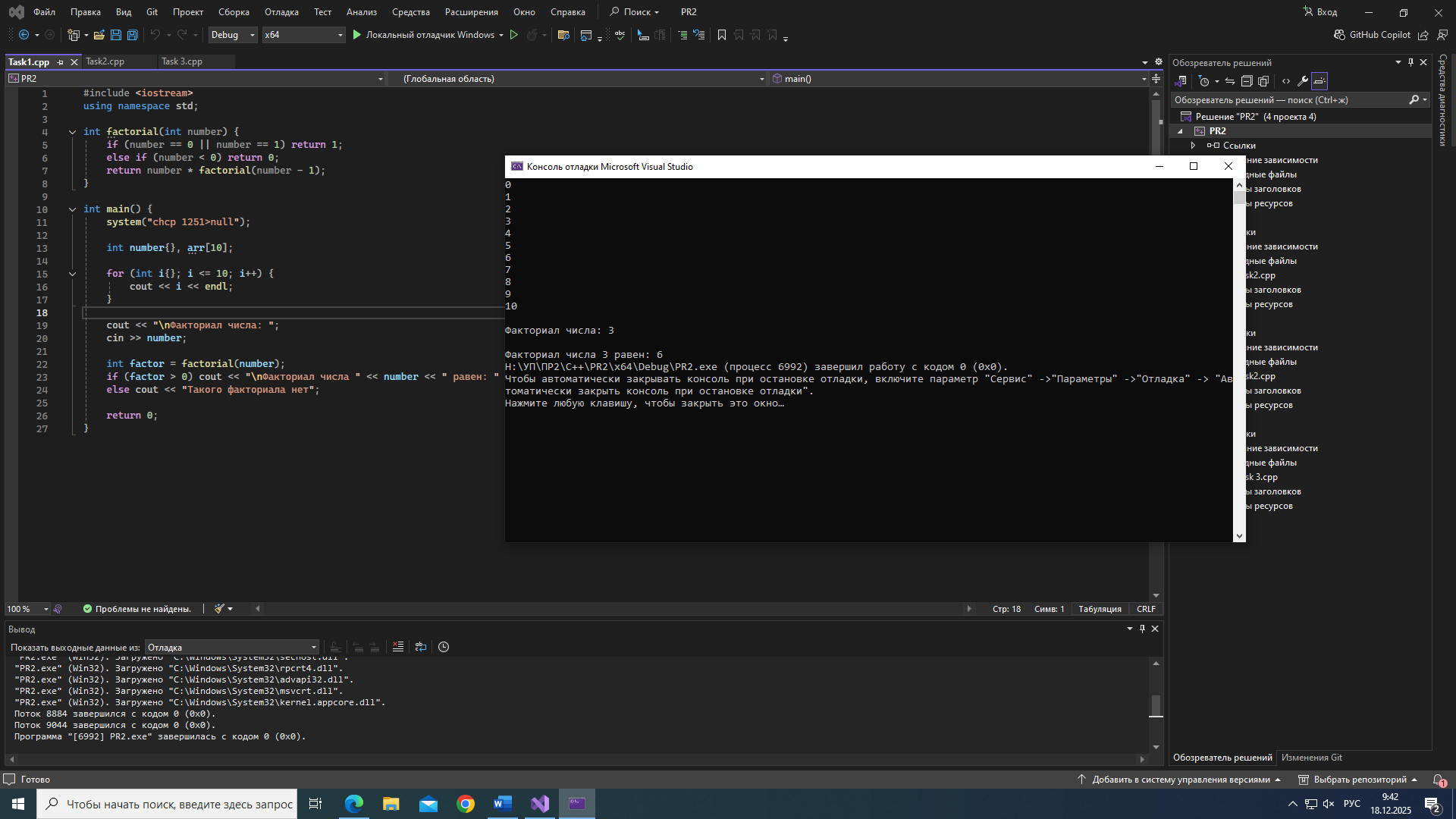
return 0;

}

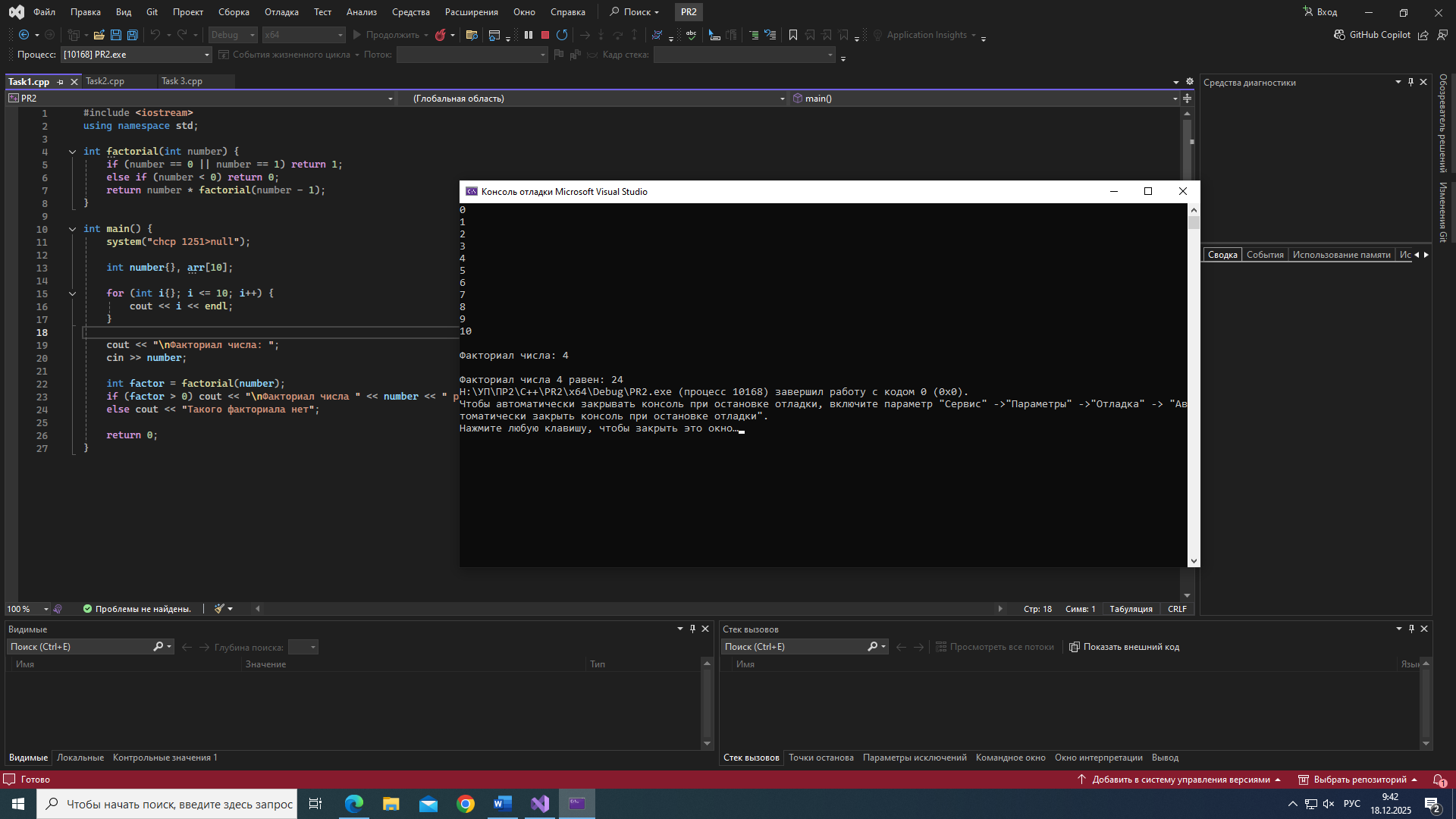
**Скриншоты работы программы:**



Скриншот 4 – Запуск 1



Скриншот 5 – Запуск 2



Скриншот 6 – Запуск 3.

* 1. Управление исключениями

Чтобы настроить отладчик для прерывания при определённых исключениях, нужно запустить отладку (F5) и перейти по следующему пути «Отладка → Окна → Параметры исключения». Появится вот такое меню, как показано на рисунке (Рисунок 37).

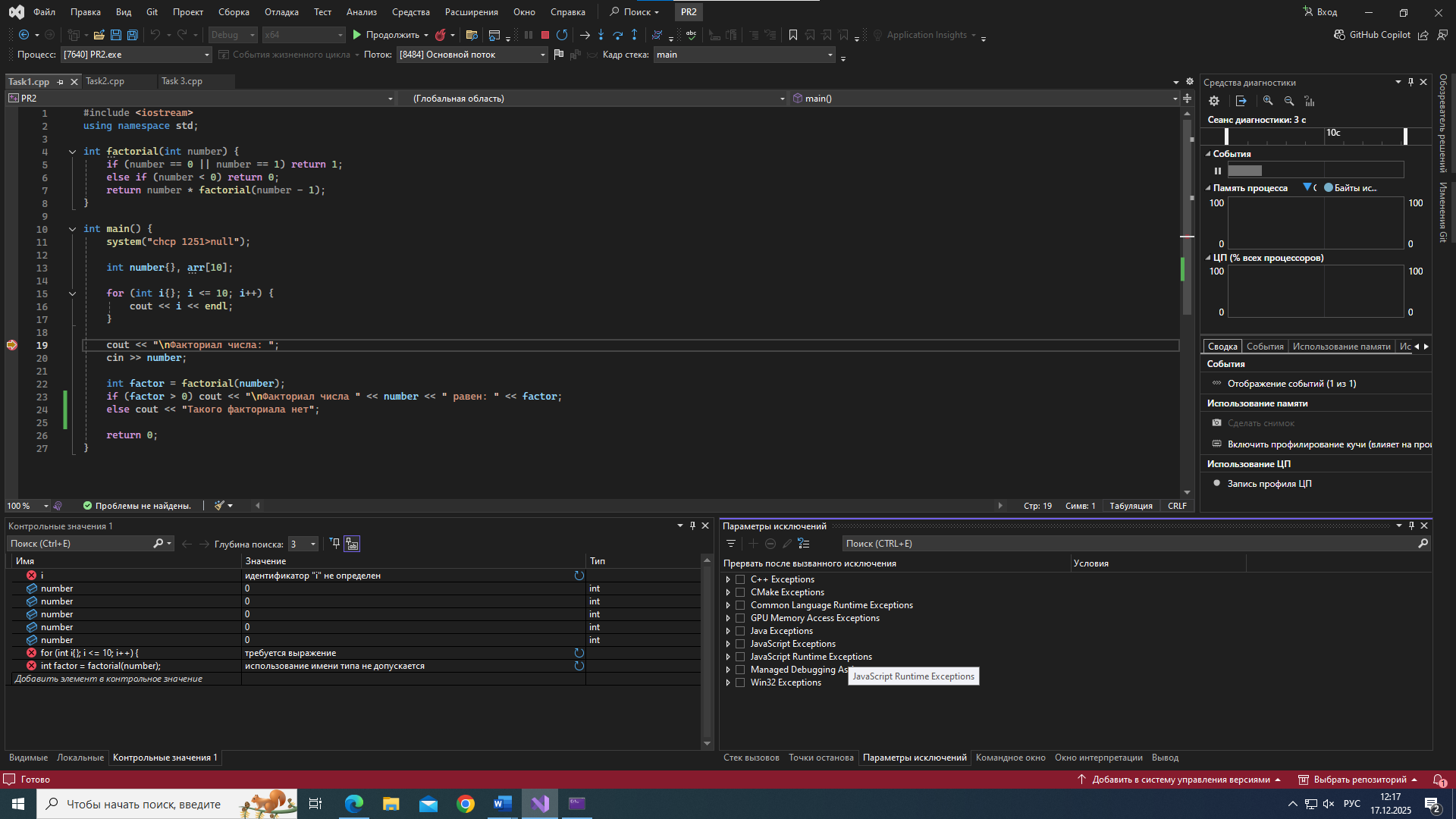


Рисунок 37 – Значения в панели «Quick Watch».

**Код задания:**

#include <iostream>

#include <vector>

using namespace std;

int factorial(int number) {

if (number == 0 || number == 1) return 1;

else if (number < 0) return 0;

return number \* factorial(number - 1);

}

void divideByZero() {

int a{3}, b{};

cout << "Введите число для деления числа " << a << ": ";

cin >> b;

if (b != 0) a / b;

else throw "Деление на ноль!";

}

void accessInvalidMemory() {

vector<int> arr(5);

int b; // value = arr.at(10);

cout << "Введите индекс елемента масиива размерностью 5: ";

cin >> b;

if (b < 5 && b > 0) cout << "Повезло!";

else throw "Выход за границы массива.";

}

int main() {

system("chcp 1251>null");

int number{};

for (int i{}; i <= 10; i++) {

cout << i << endl;

}

cout << "\nФакториал числа: ";

cin >> number;

int factor = factorial(number);

if (factor > 0) cout << "\nФакториал числа " << number << " равен: " << factor;

else cout << "Такого факториала нет";

cout << "\n\nПро деление на ноль\n";

try {

divideByZero();

}

catch (const char\* error\_message) {

cout << "Error: " << error\_message << std::endl;

}

cout << "\nПро массив\n";

try {

accessInvalidMemory();

}

catch (const char\* error\_message) {

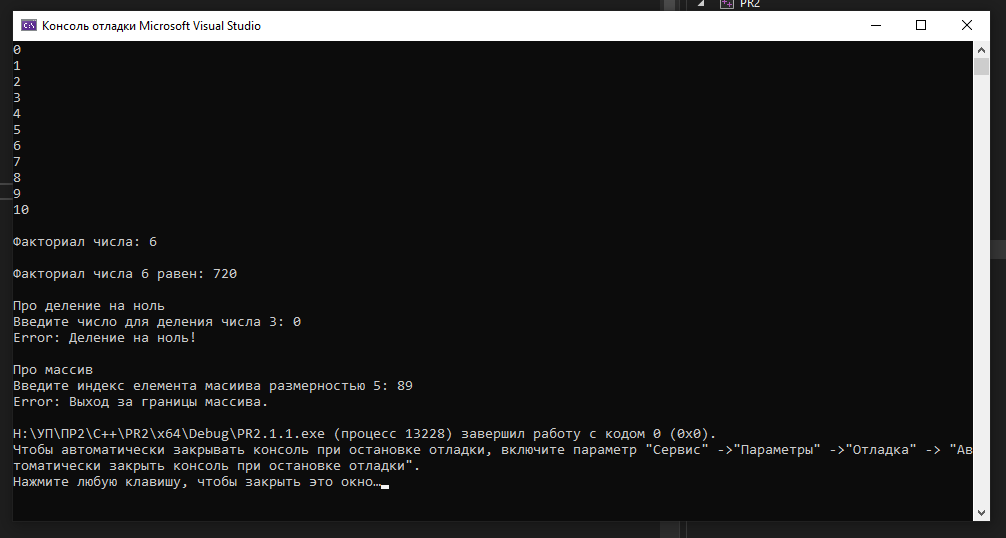
cout << "Error: " << error\_message << std::endl;

}

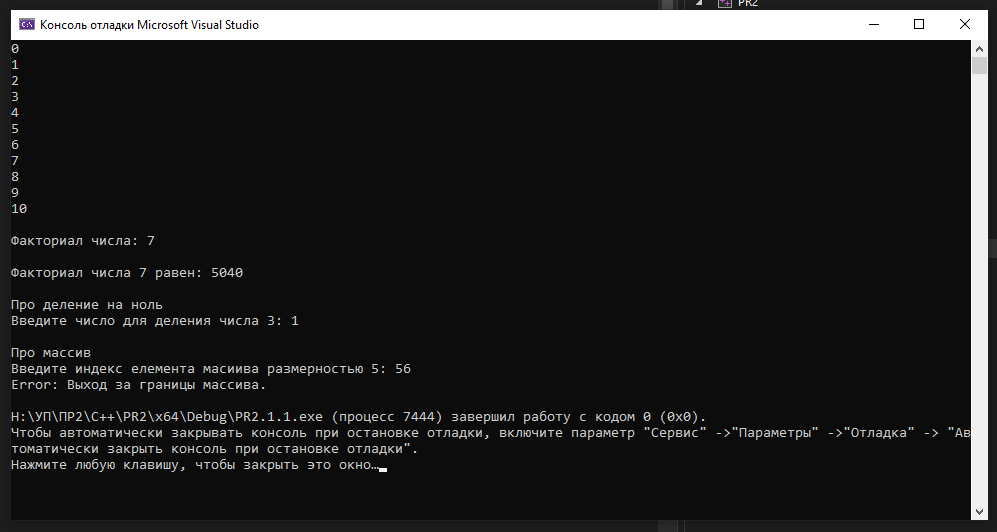
return 0;

}

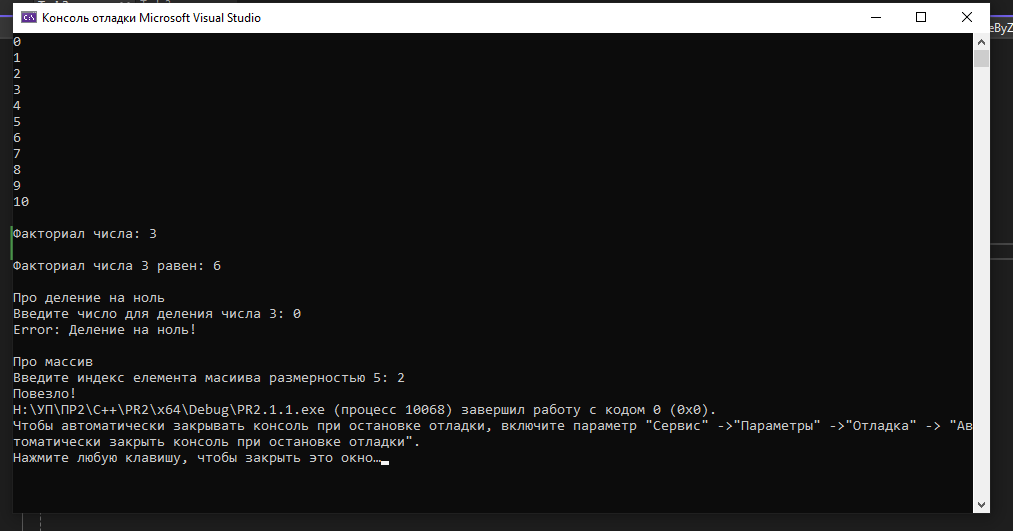
**Скриншоты работы программы:**



Скриншот 7 – Запуск 1.



Скриншот 8 – Запуск 2.



Скриншот 9 – Запуск 3.

* 1. Профилирование

Чтобы узнать использование ЦП и использование памяти компьютера при компиляции кода, нужно пройти по следующему пути «Отладка → Профилировщик» и в самом окне выбрать «Использование ЦП» и «Использование памяти» и нажимаем кнопку «Начать». В конце получаем графики (Рисунок 38).

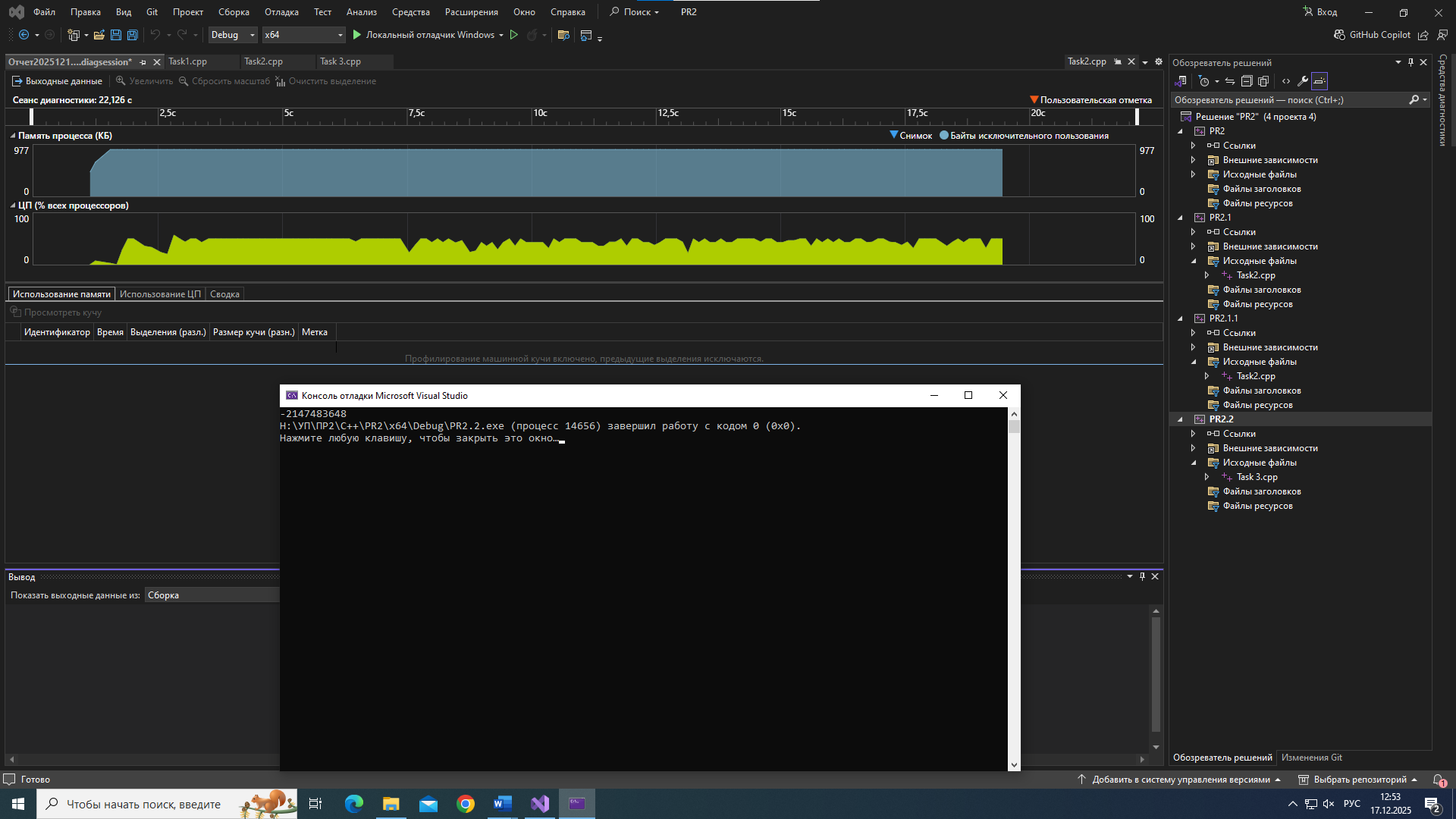


Рисунок 38 – Значения в панели «Quick Watch».

**Код задания:**

#include <iostream>

#include <vector>

using namespace std;

int main() {

system("chcp 1251>null");

int product{1}, A{}, B{};

for (int i{}; i <= 10000; i++) {

for (int j{}; j <= 10000; j++) {

A = i;

B = j;

product = pow(A, B);

}

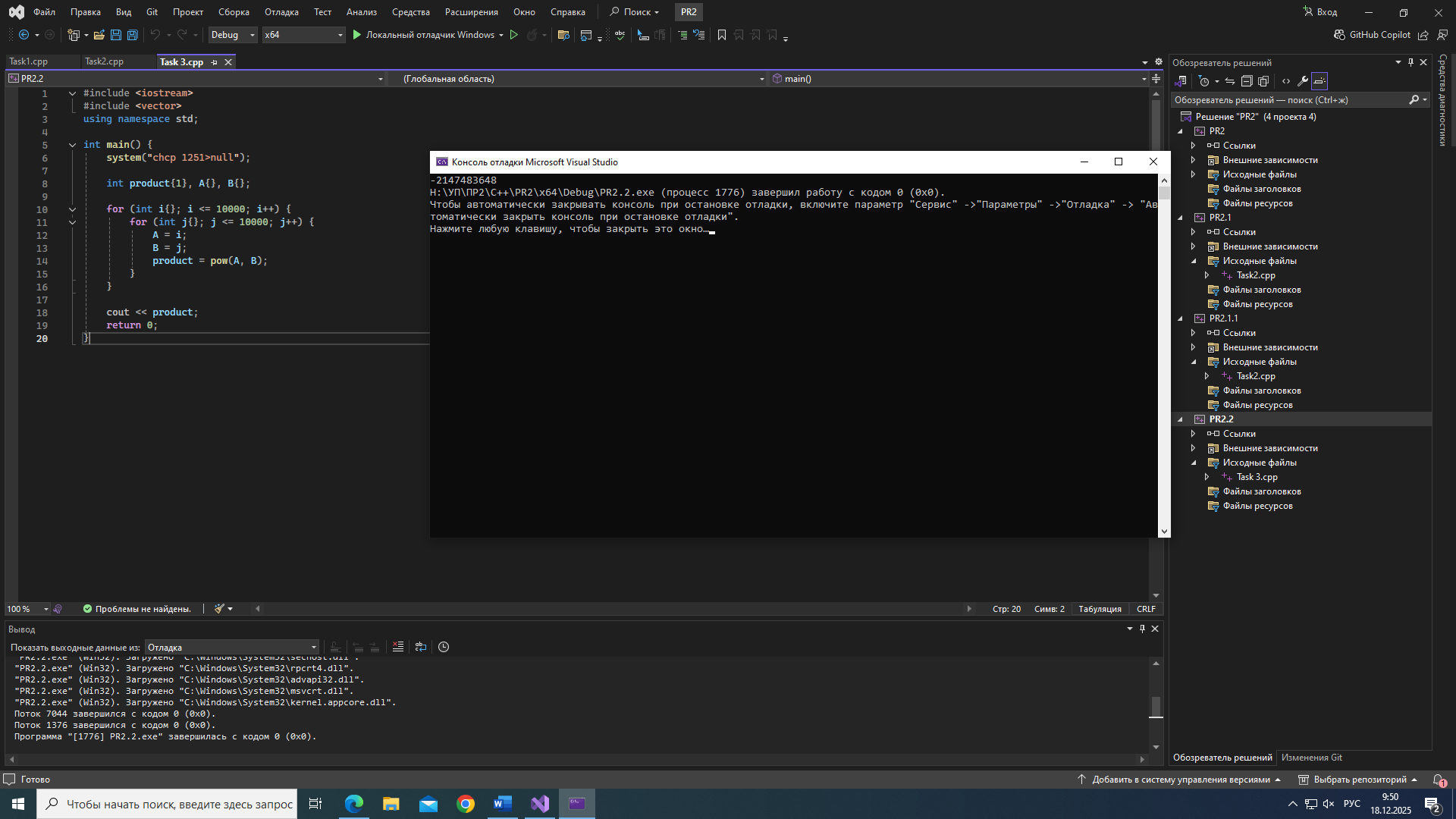
}

cout << product;

return 0;

}

**Скриншоты работы программы:**



Скриншот 10 – Запуск (первый и последний, ибо программа не имеет вариативности).

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

Тюнин Илья Андреевич

(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     Васильев С. Ю. \_\_\_\_\_\_\_\_\_\_\_\_\_

(Ф.И.О.) (подпись)