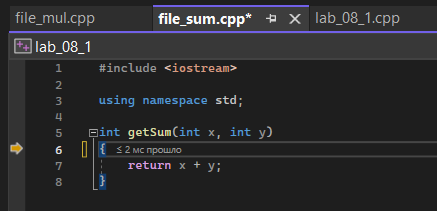
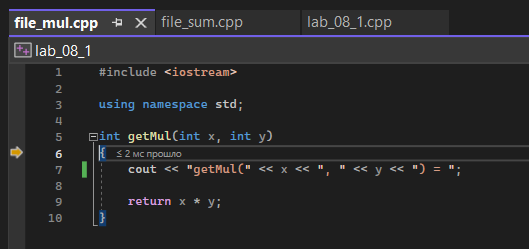
БГТУ ФИТ ПИ Жамойдо Артём Игоревич 1-10-2 ОПИ

ЛАБОРАТОРНАЯ РАБОТА 8

№ 2

Отладка примера из лекции 9 в пошаговом режиме с заходом в функции getSum и getMul:

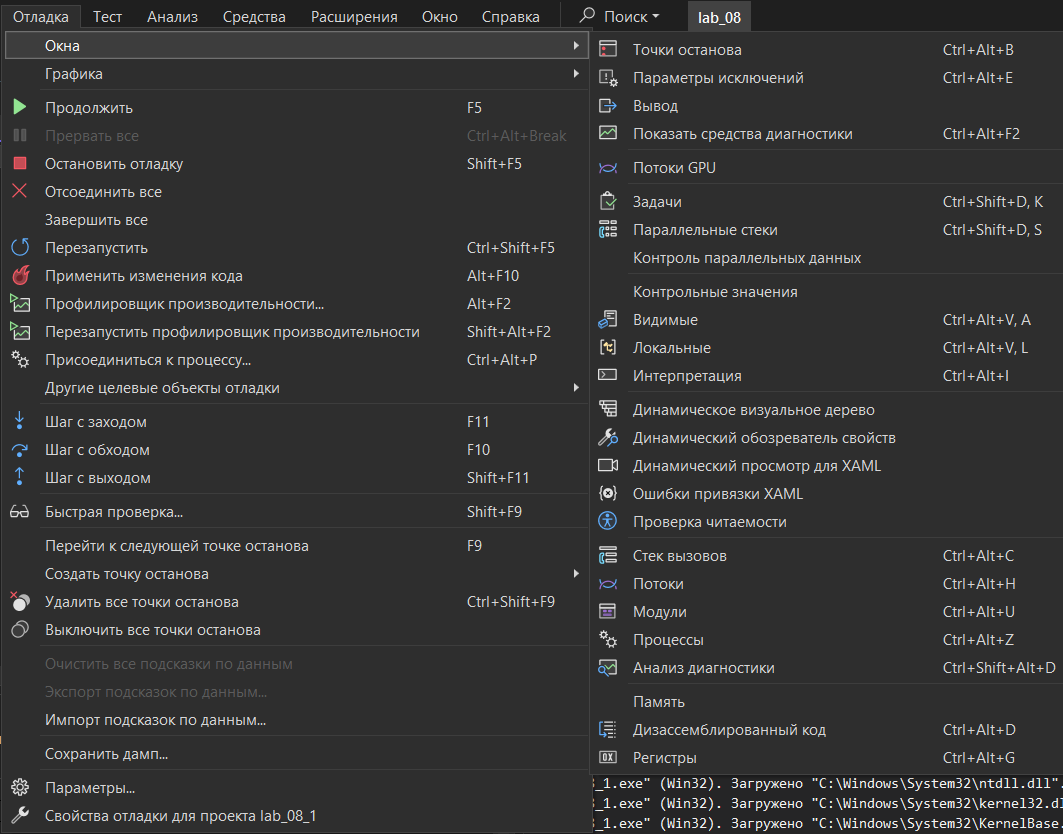


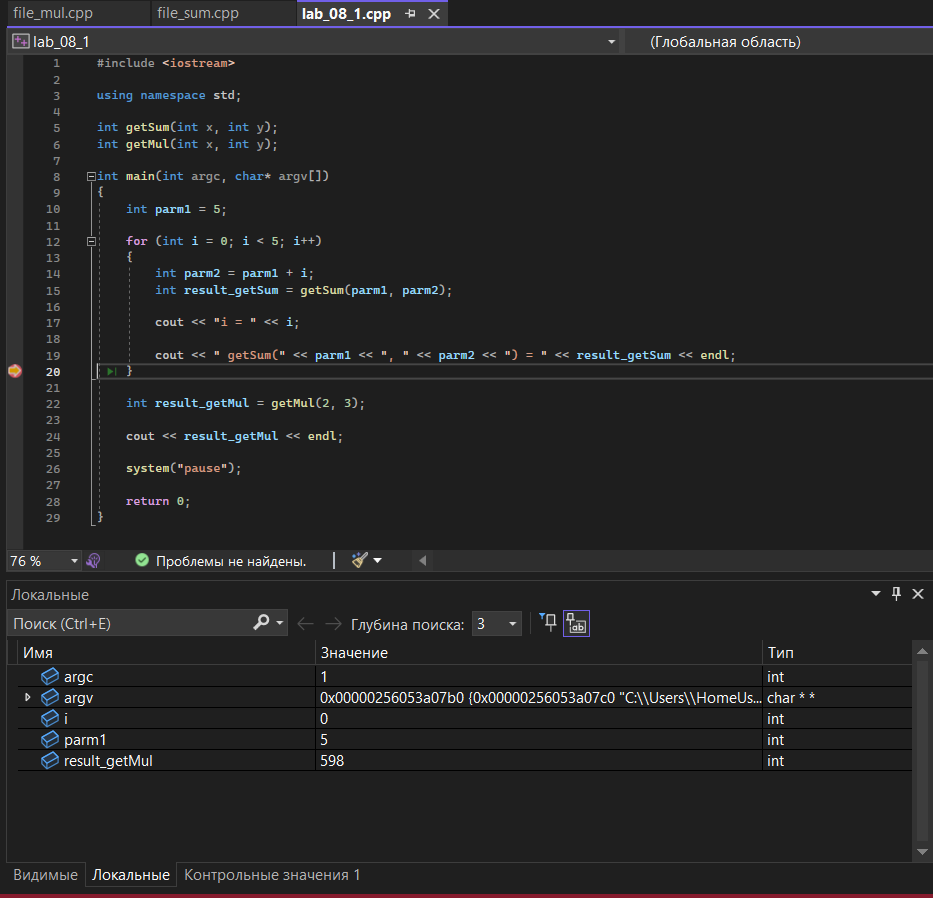


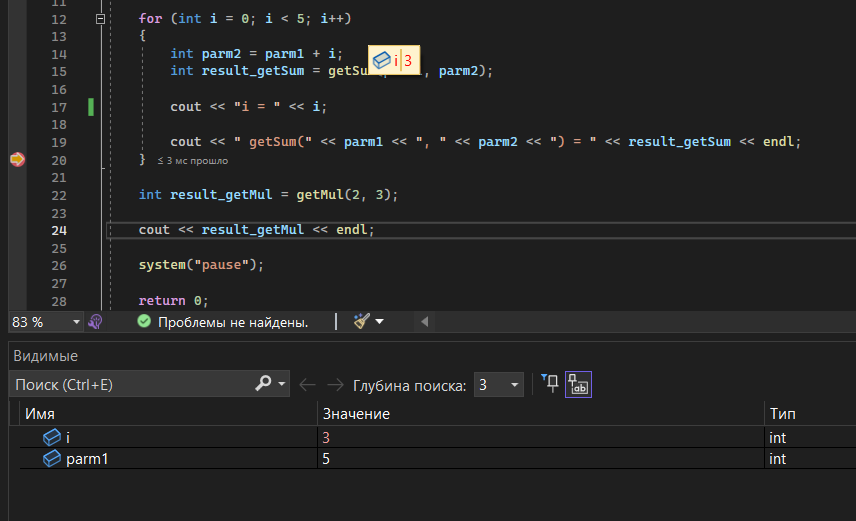
№ 3

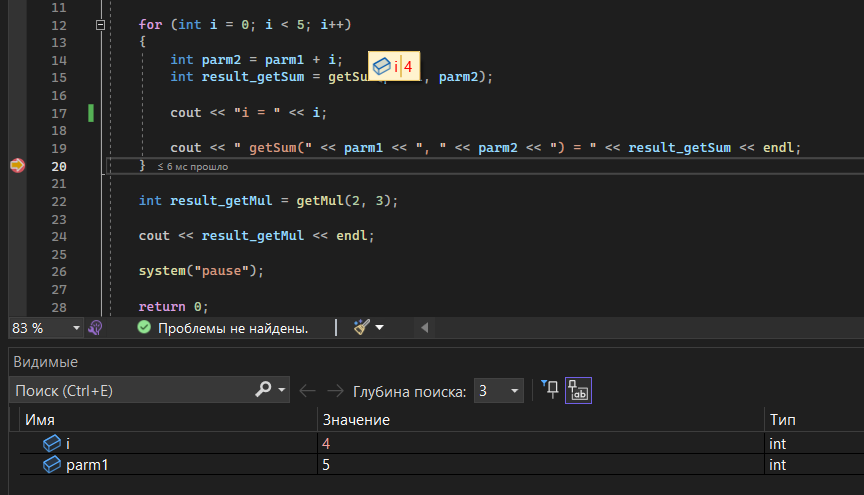
Выполнение всех примеров, приведенных в лекции:

1

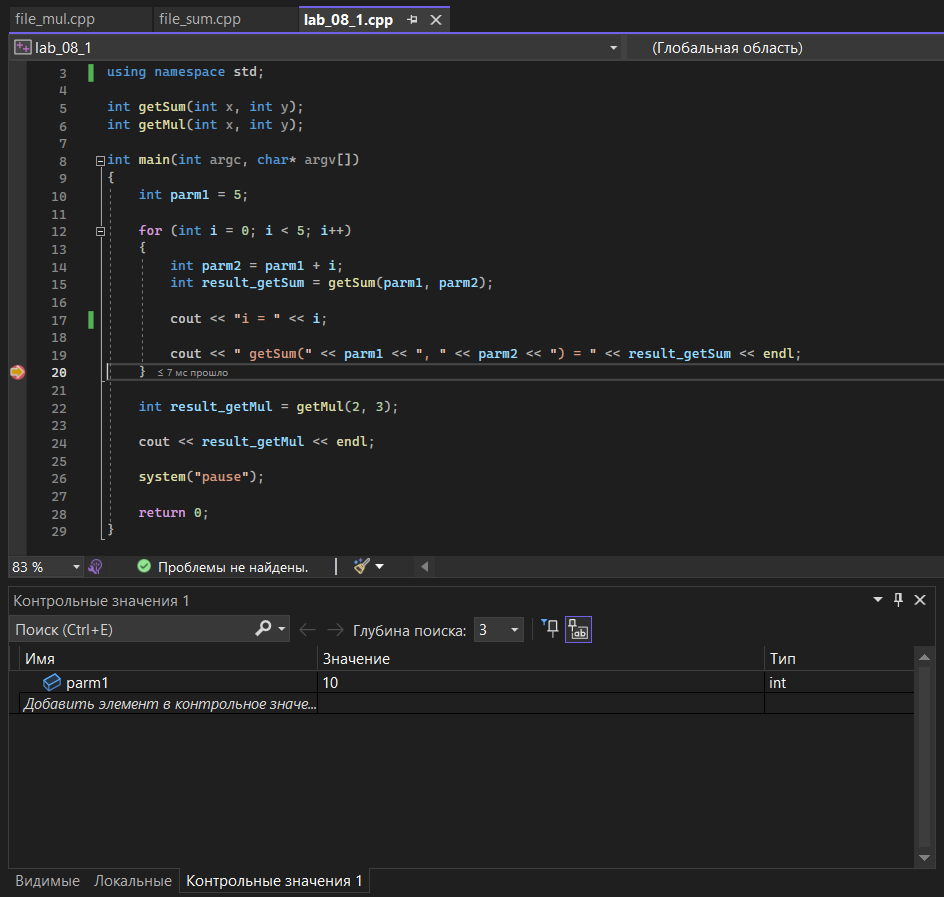


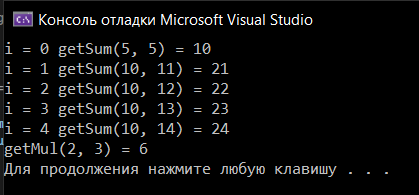




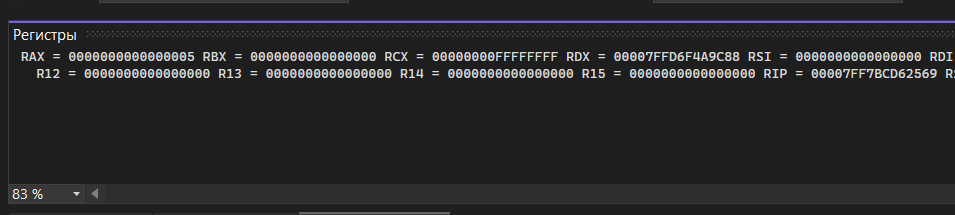


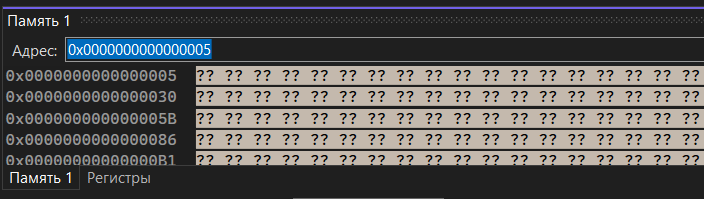
2

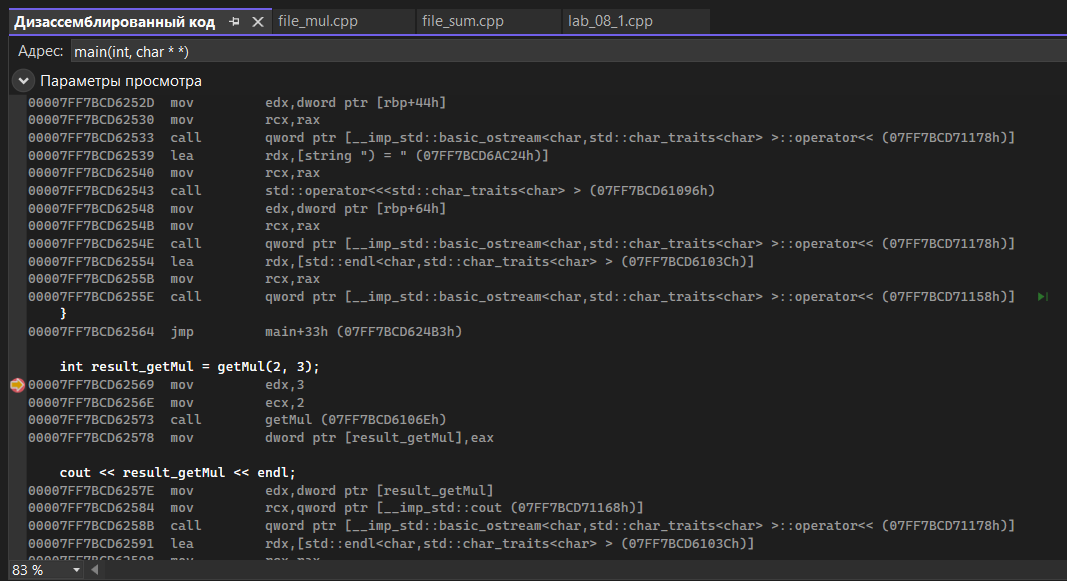


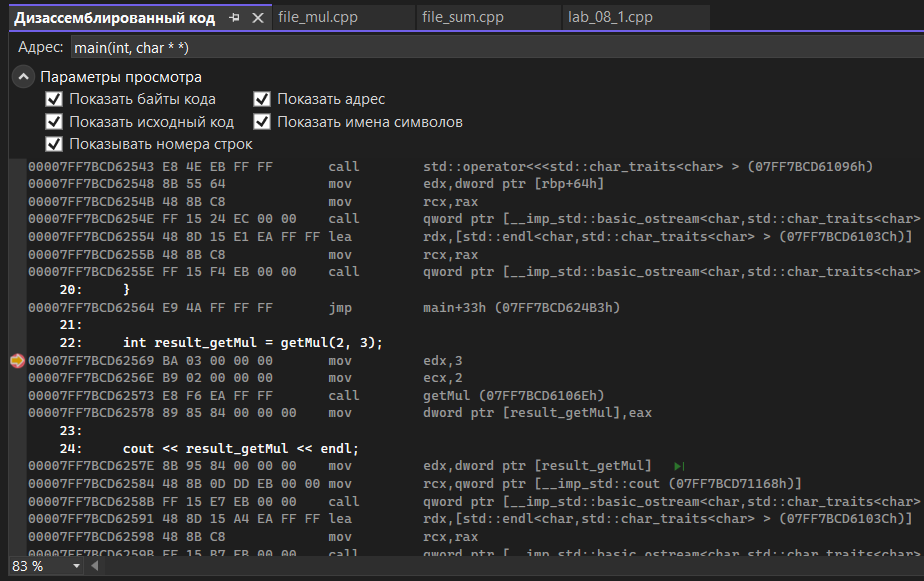


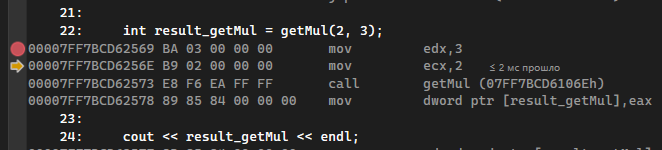
3

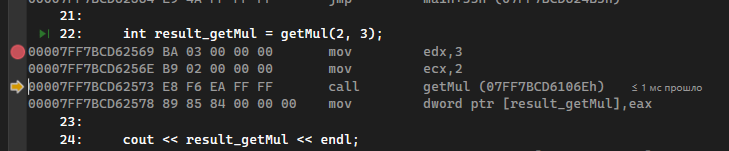


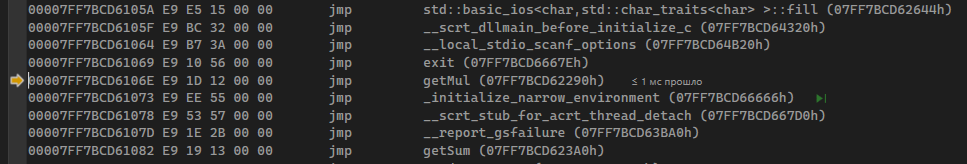




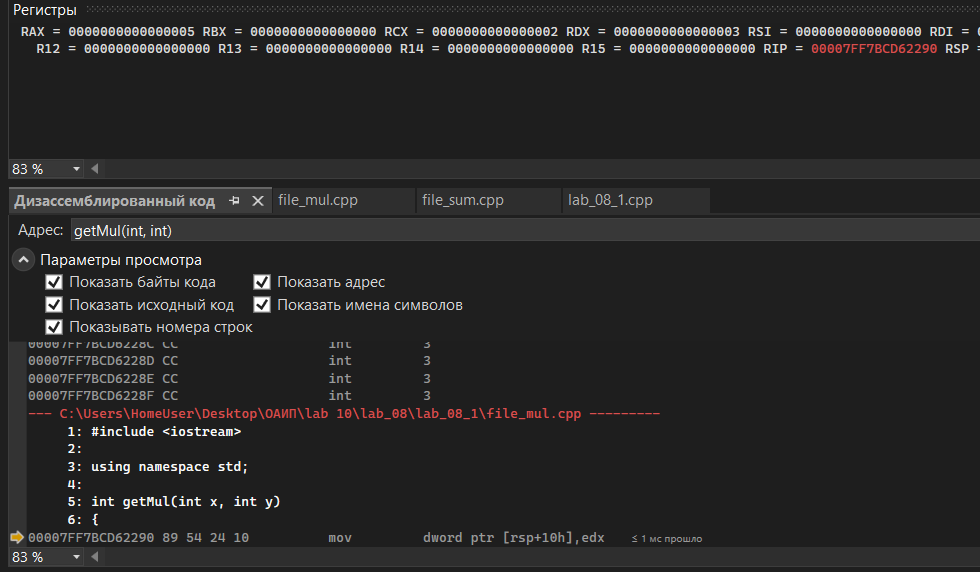


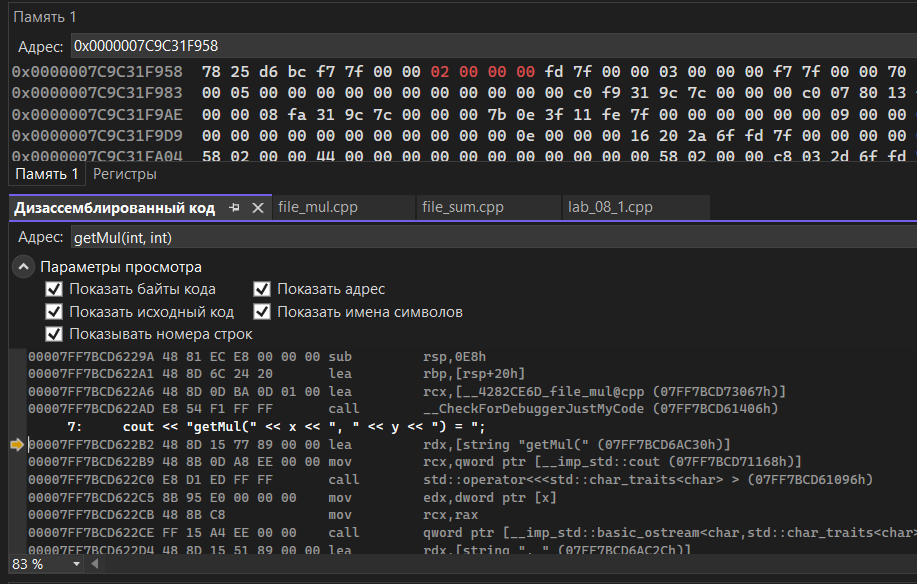






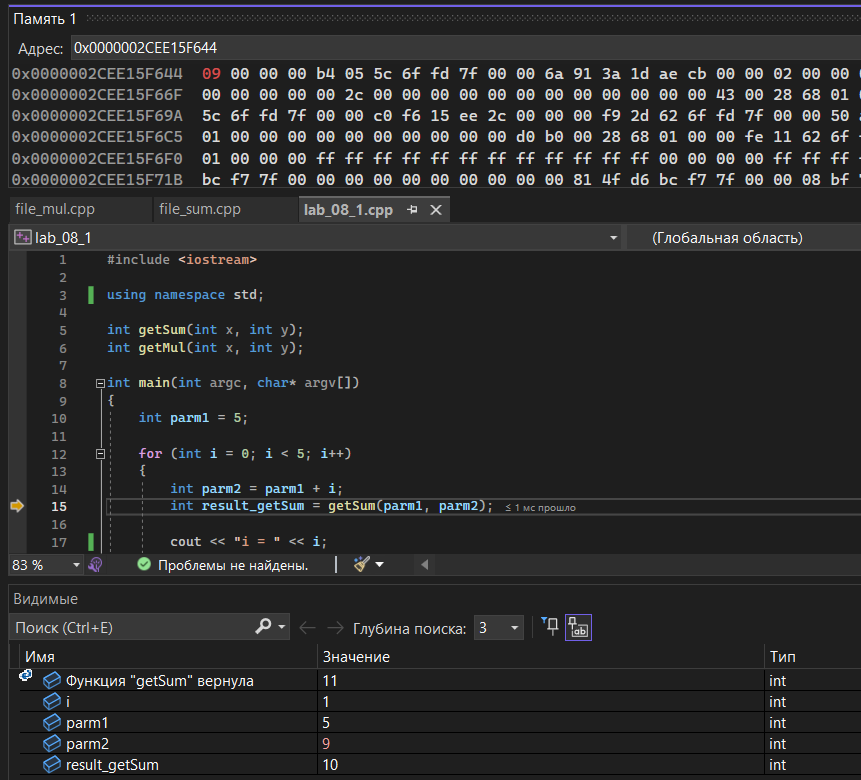






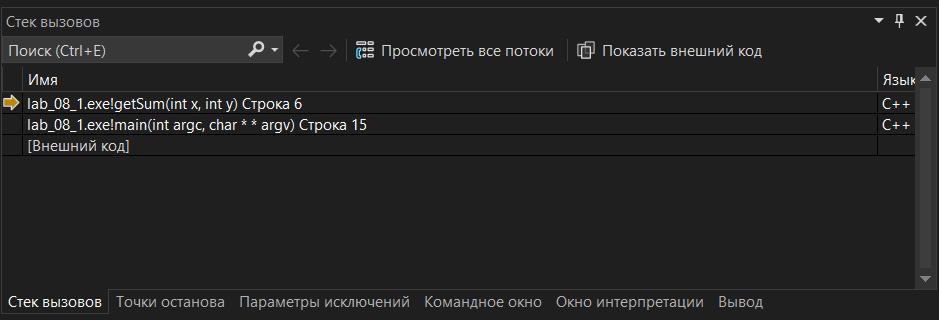
№ 4

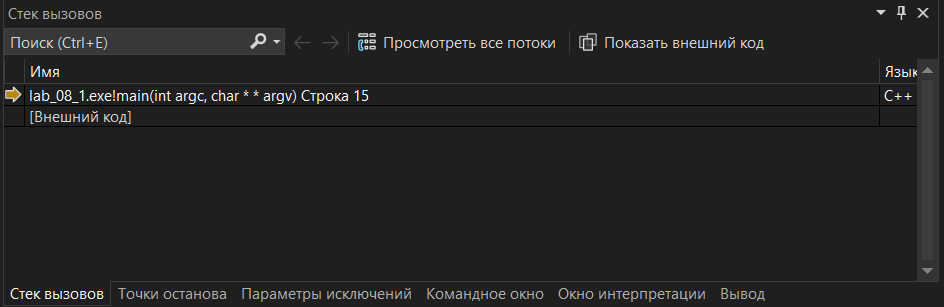
Просмотр/изменение значения некоторой переменной в памяти:



№ 5

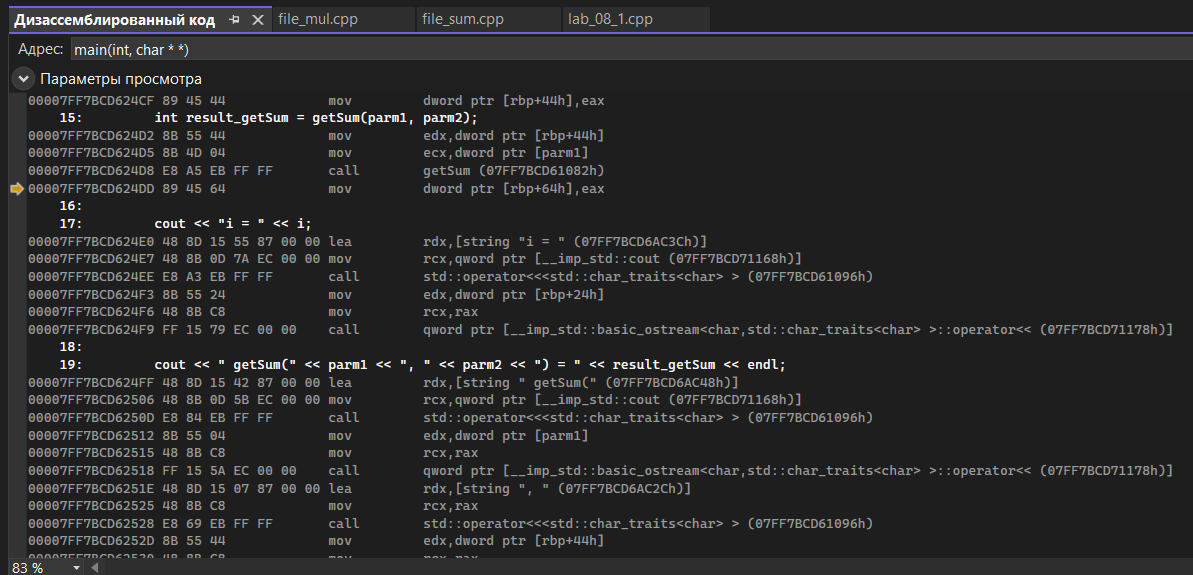
Просмотр стека вызовов на нескольких итерациях цикла:





№ 6

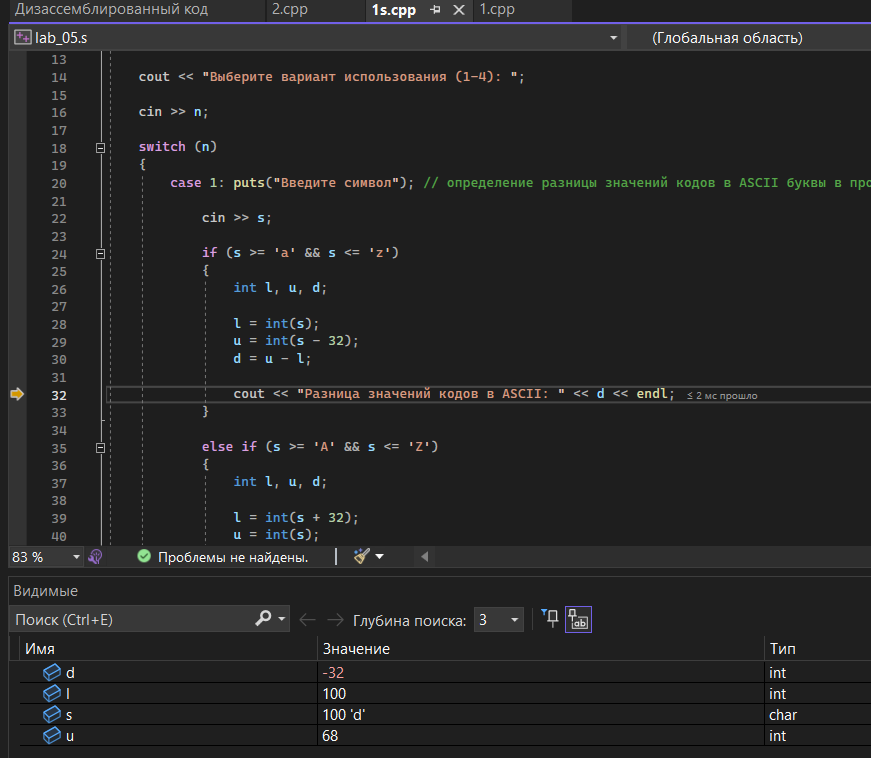
Окно «Дизассемблированный код»:



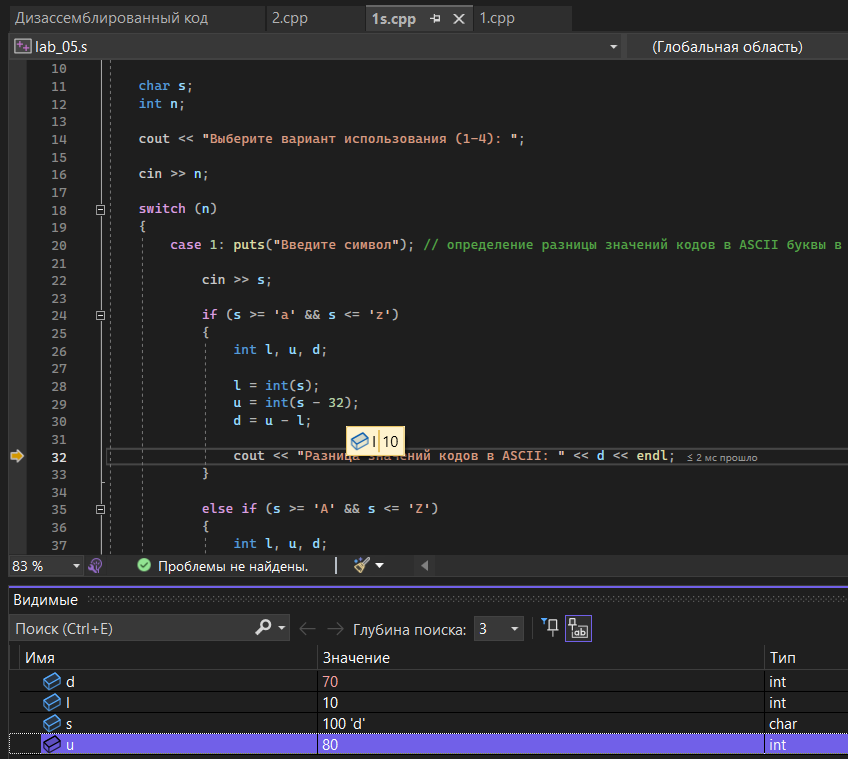
№ 7

Тестирование программы из п.5 лабораторной работы № 5

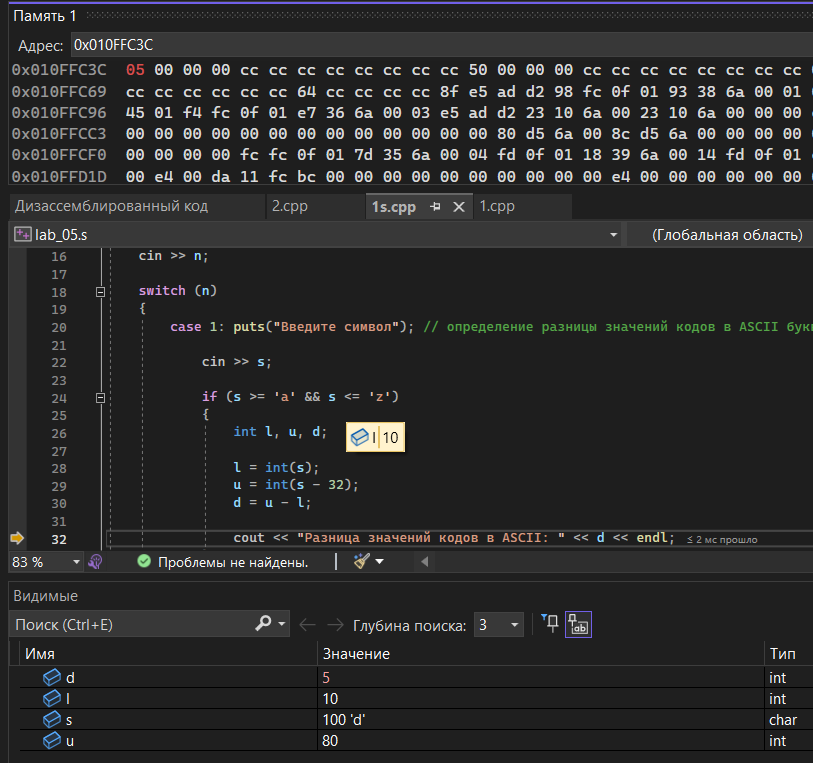
Отладка программы в пошаговом режиме:



Проверка и изменение значения переменных всеми возможными способами (используя окна отладчика, подсказки):



Значения переменных в памяти:



Ответы на вопросы

1. Устранение ошибок в коде программы.

2. Основные возможности отладчика VC++ включают установку точек останова, выполнение программы по шагам, просмотр значений переменных и стека вызовов, анализ кода и т.д.

3. Точка останова - это место в программном коде, где выполнение программы будет приостановлено, чтобы можно было произвести отладку.

4. Способы установки точек останова включают клик мышью на строке кода, использование команды breakpoint в отладчике, а также установку условной точки останова, которая будет срабатывать только при выполнении определенного условия.

5. Способы запуска отладчика включают запуск программы из самого отладчика, а также подключение к уже запущенной программе.

6. Основные возможности управления режимами отладки включают установку точек останова, выполнение программы по шагам, изменение значений переменных во время выполнения программы и т.д.

7. Способы для отображения значений переменных включают использование окон локальные, видимые и контрольные значения .

8. Отладочные окна Debuggerа включают окна локальные, видимые, контрольные значения , памяти и т.д.

9. Окно памяти используется для просмотра содержимого определенной области памяти во время выполнения программы.

10. Окно регистров используется для просмотра значений регистров процессора во время выполнения программы.

11. Окно дизассемблированного кода используется для просмотра машинного кода программы и анализа ее работы.

12. В окне дизассемблированного кода отображается машинный код программы, разбитый на отдельные инструкции, а также соответствующий им исходный код на языке C++.