Федеральное государственное бюджетное образовательное учреждение высшего образования «Санкт-Петербургский государственный университет телекоммуникаций им. проф. М.А. Бонч-Бруевича».

**Факультет Информационных систем и технологий**

**Кафедра Информатики и компьютерного дизайна**

**Лабораторная работа №4**

Исследование и обработка структурированных типов данных в программах

****

**Автор работы:**

Сухоруков А.А.

**Научный руководитель:**

Федоров Д.Ю.

Санкт-Петербург, 2020

**1. Постановка задачи.**

**1.** Сделать свой первый рендер SDL

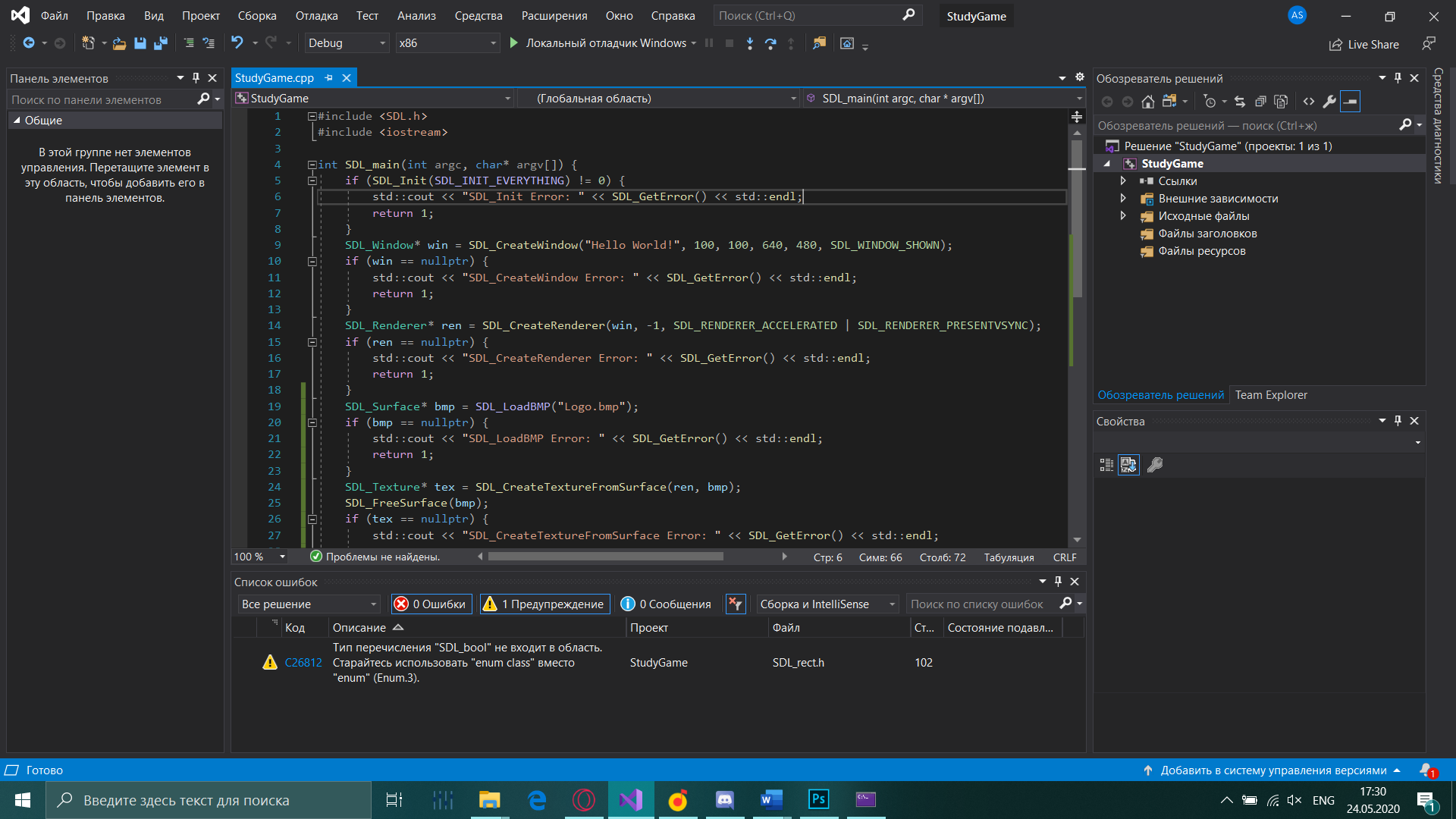
**2. Формализация задачи.**

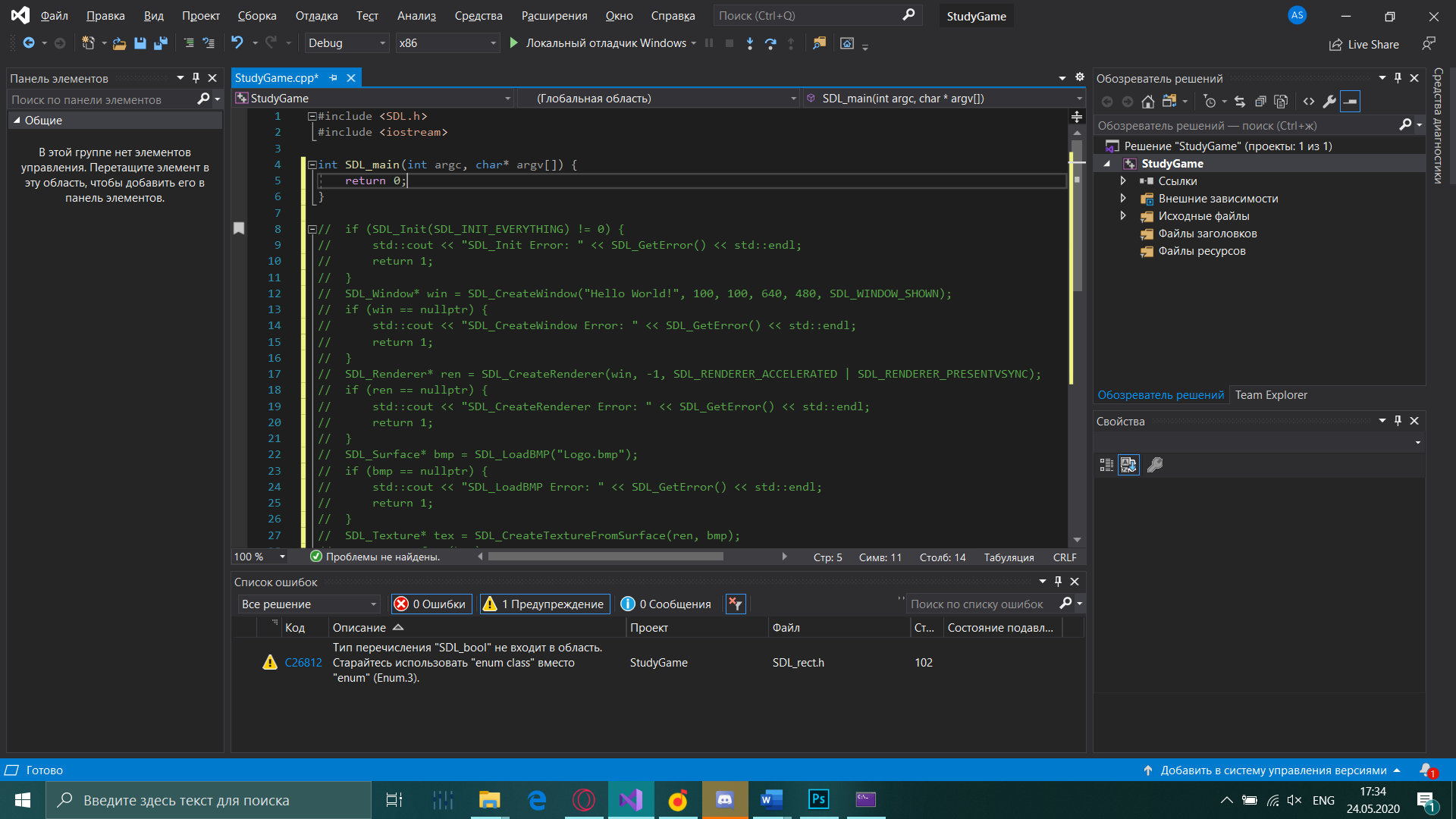
**1.** Сделать свой первый рендер SDL  
Нам будет нужно вывести на экран картинку, что по сути задача не сложная, но она познакомит с основной функцией библиотеки SDL – с рендером. Потому постараюсь подробно описать как превратить нашу картинку в текстуру игры и вывести ее на экран

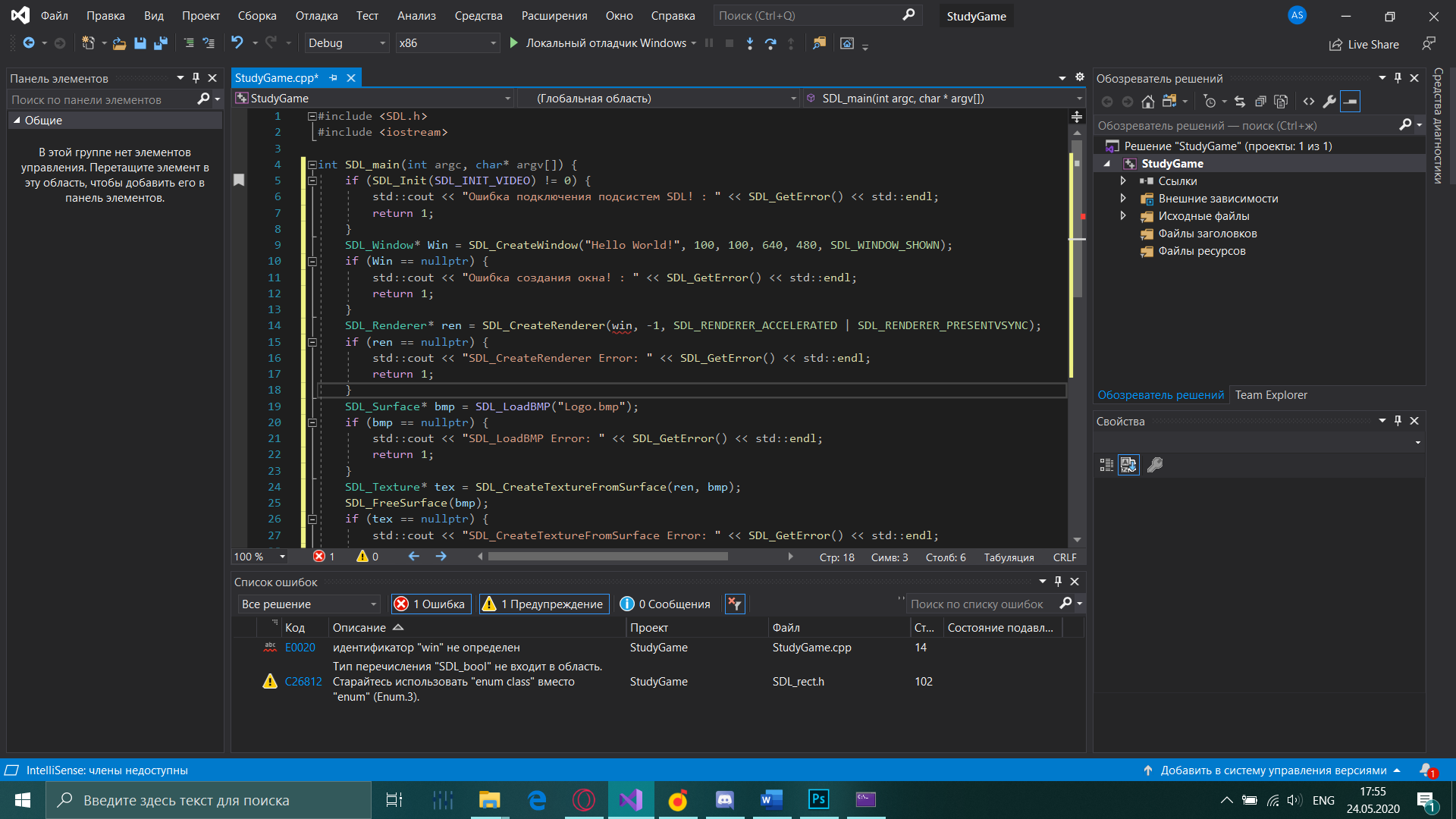
1. **Сделать свой первый рендер SDL**

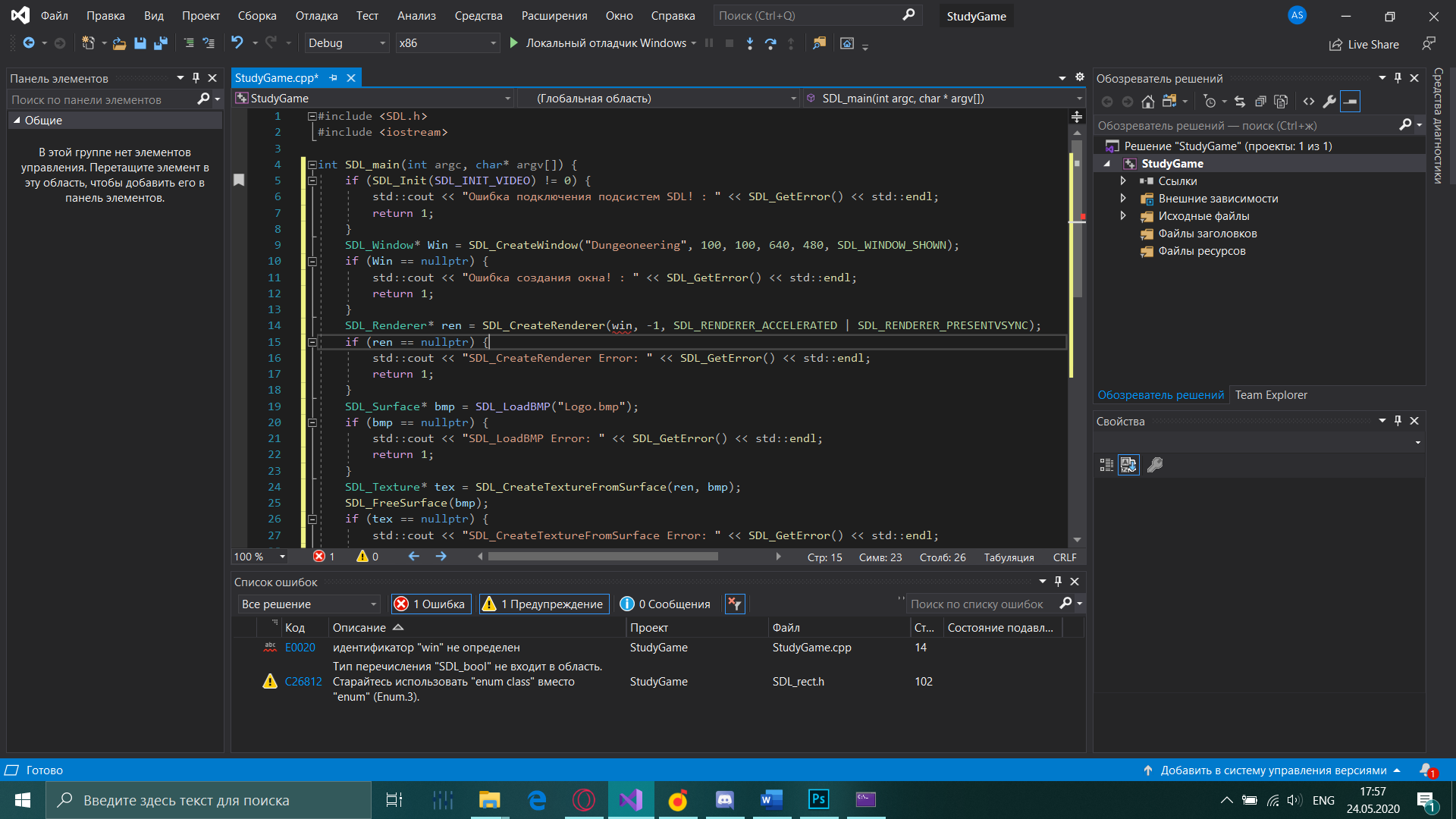
Проделав все шаги, описанные в прошлых лабораторных, я могу приступать уже к своему первому рендеру. И начать, конечно же, стоит с самой простой задачи – **вывести на свой экран картинку**

В прошлых лабораторных я уже рисовал иконку своей игры, потому ее и используем, только в формате **BMP**

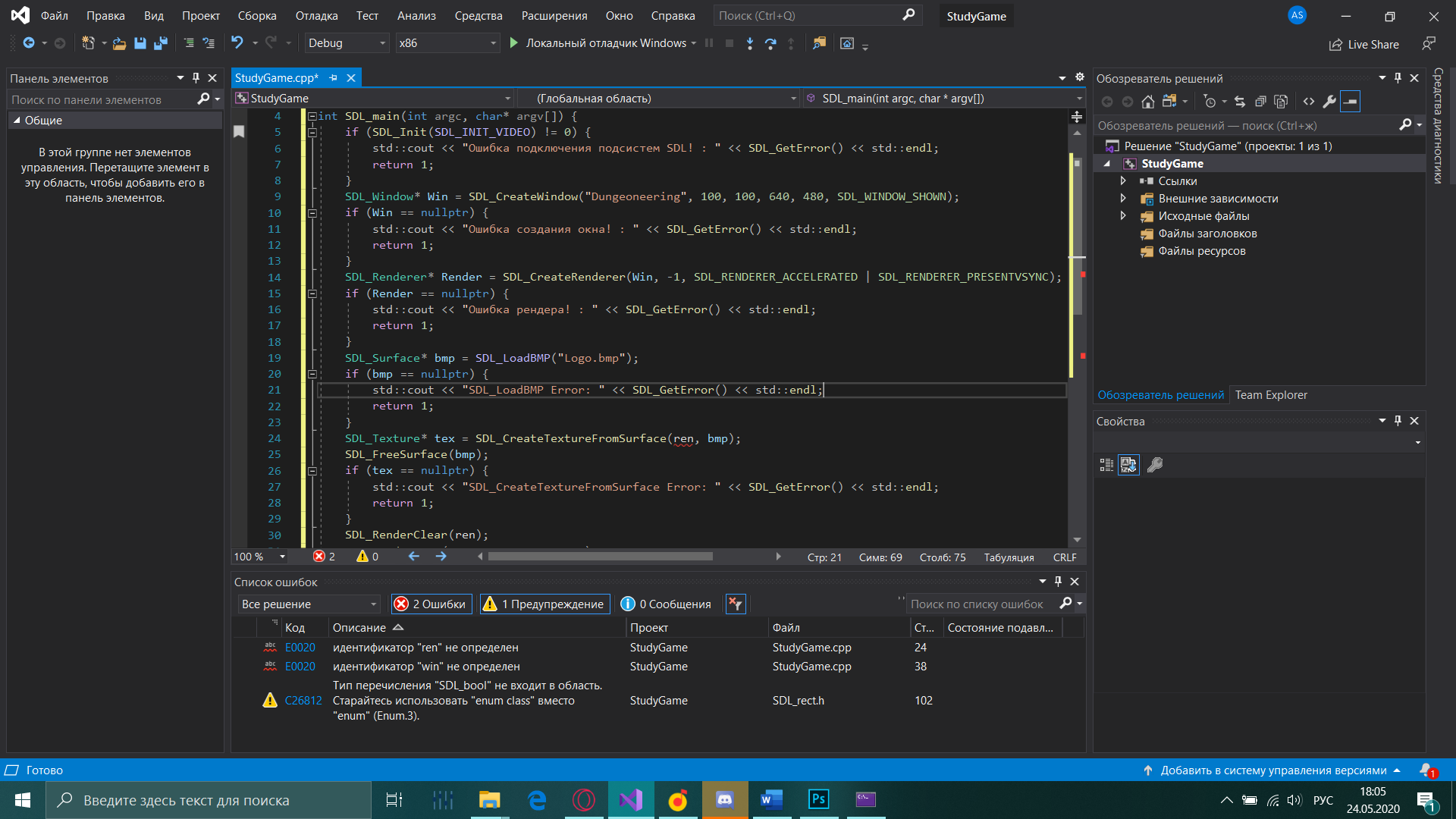
Наш обучающий проект будет называться **StudyGame**, а подключу к нему только SDL и основую библиотеку iostream  


При использовании SDL мы меняем имя нашего main на **SDL\_main**

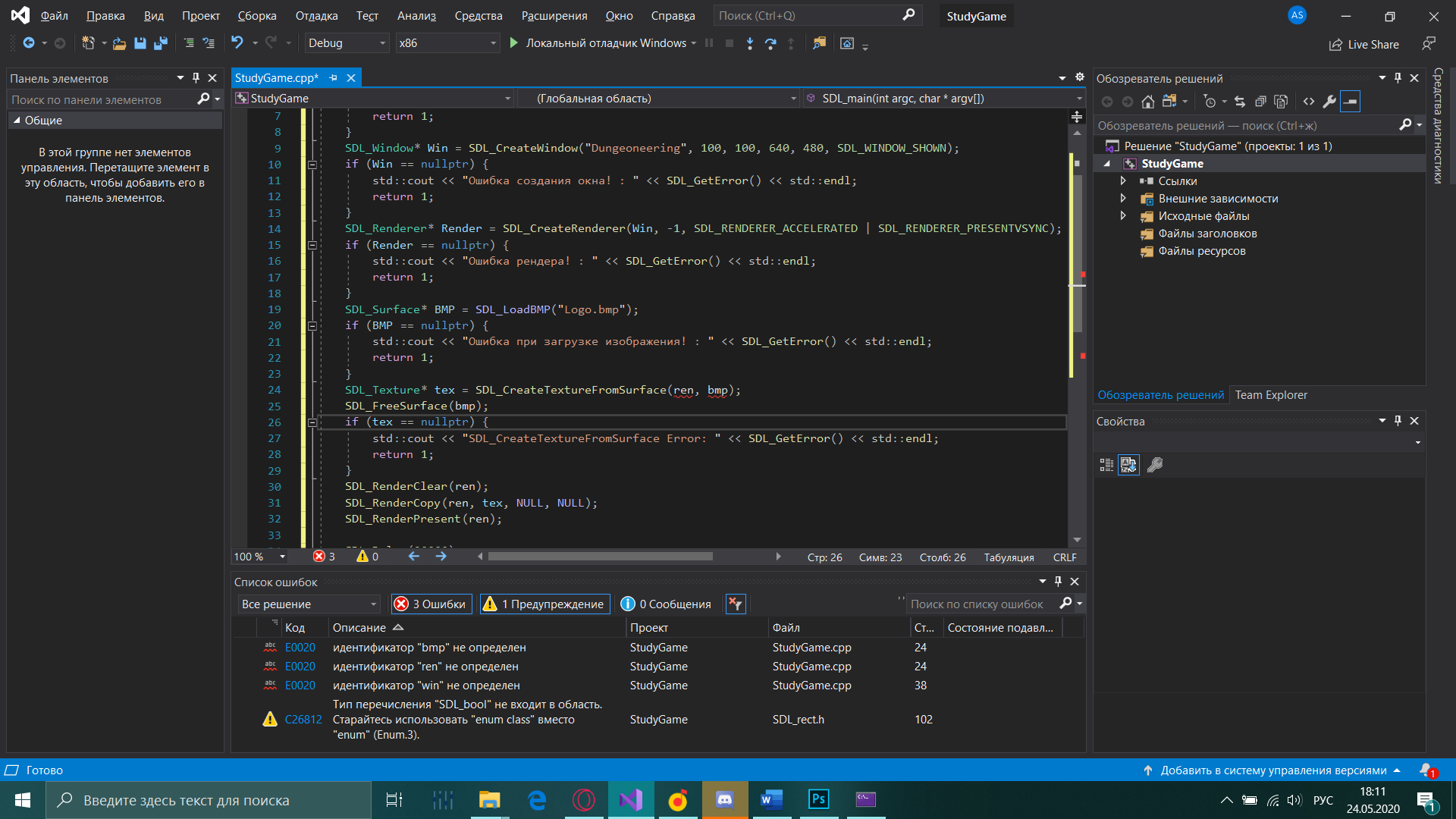
При написании программы на SDL нам нужно инициализировать те подсистемы, что мы будем использовать. Для рендера картинки нам понадобится только подсистема **SDL\_INIT\_VIDEO**. Также пишу в операторе if проверку на успешную инициализацию данной подсистемы. Если произойдет какая ошибка, то SDL\_Init вернет 0 и программа напишет нам об этом благодаря функции **SDL\_GetError()**

Далее мы должны создать окно, в котором и будет отображаться наш будущий рендер. Создадим мы его с помощью **SDL\_CreateWindow**, у которого параметры его название, координаты, высота, ширина и [флаги](http://wiki.libsdl.org/moin.fcg/SDL_WindowFlags). Эта функция вернет **SDL\_Window\***, потому необходимо создать указатель данного типа, это будет **Win**. И далее я сделаю проверку на то, что наш Win не пустой  


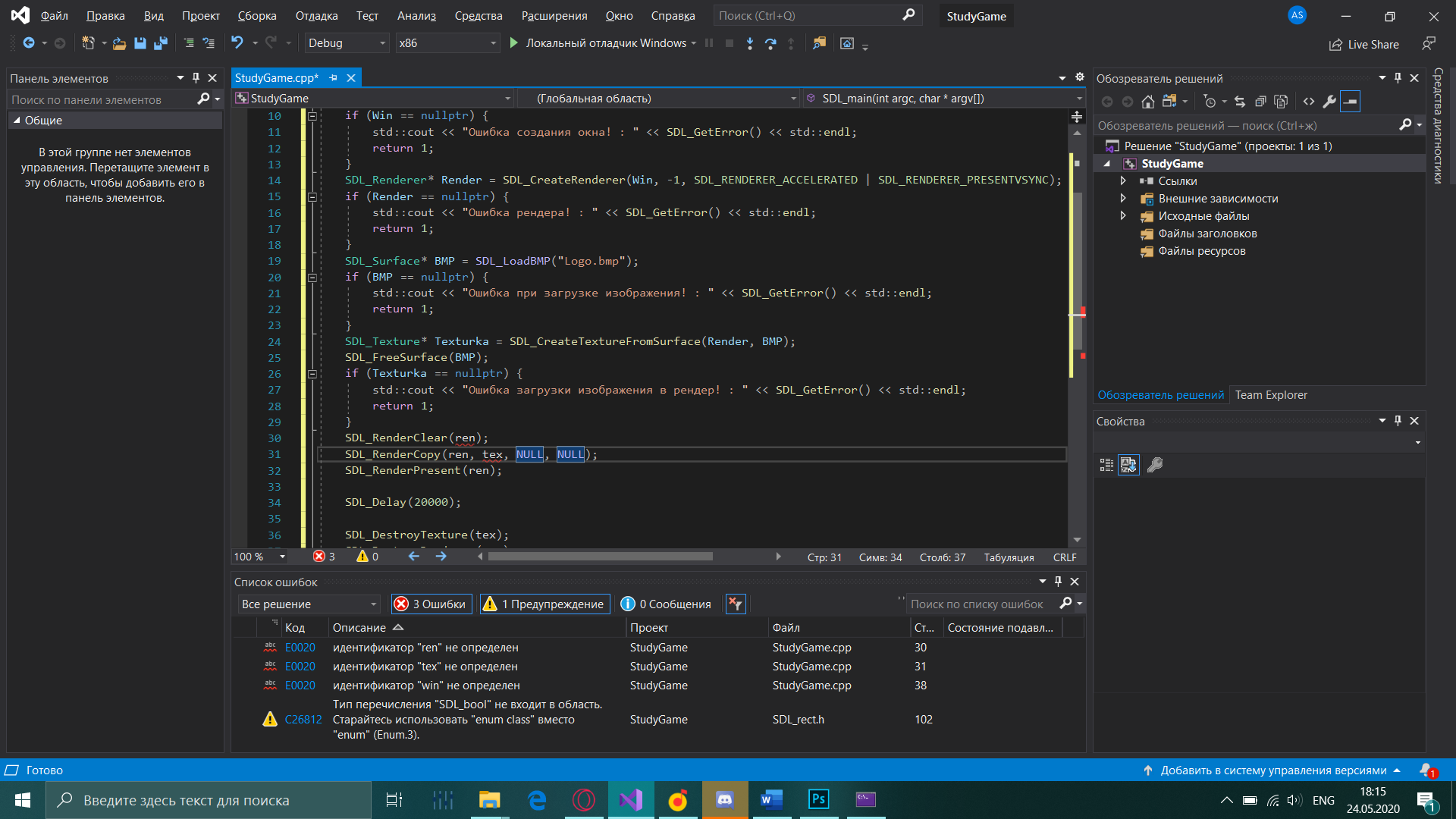
Теперь нам необходимо создать рендер (который и будет рисовать в данном окне) с помощью функции **SDL\_CreateRenderer**. В параметрах он принимает указатель на окно, индекс драйвера (-1, чтобы он выбрал его автоматически) и различные флаги, по типу вертикальной синхронизации и аппаратного ускорения. Данная функция возвращает **SDL\_Renderer\***, потому создаем указатель Render данного типа, чтобы не потерять. Также делаю проверку на то, что наш рендер успешно создался



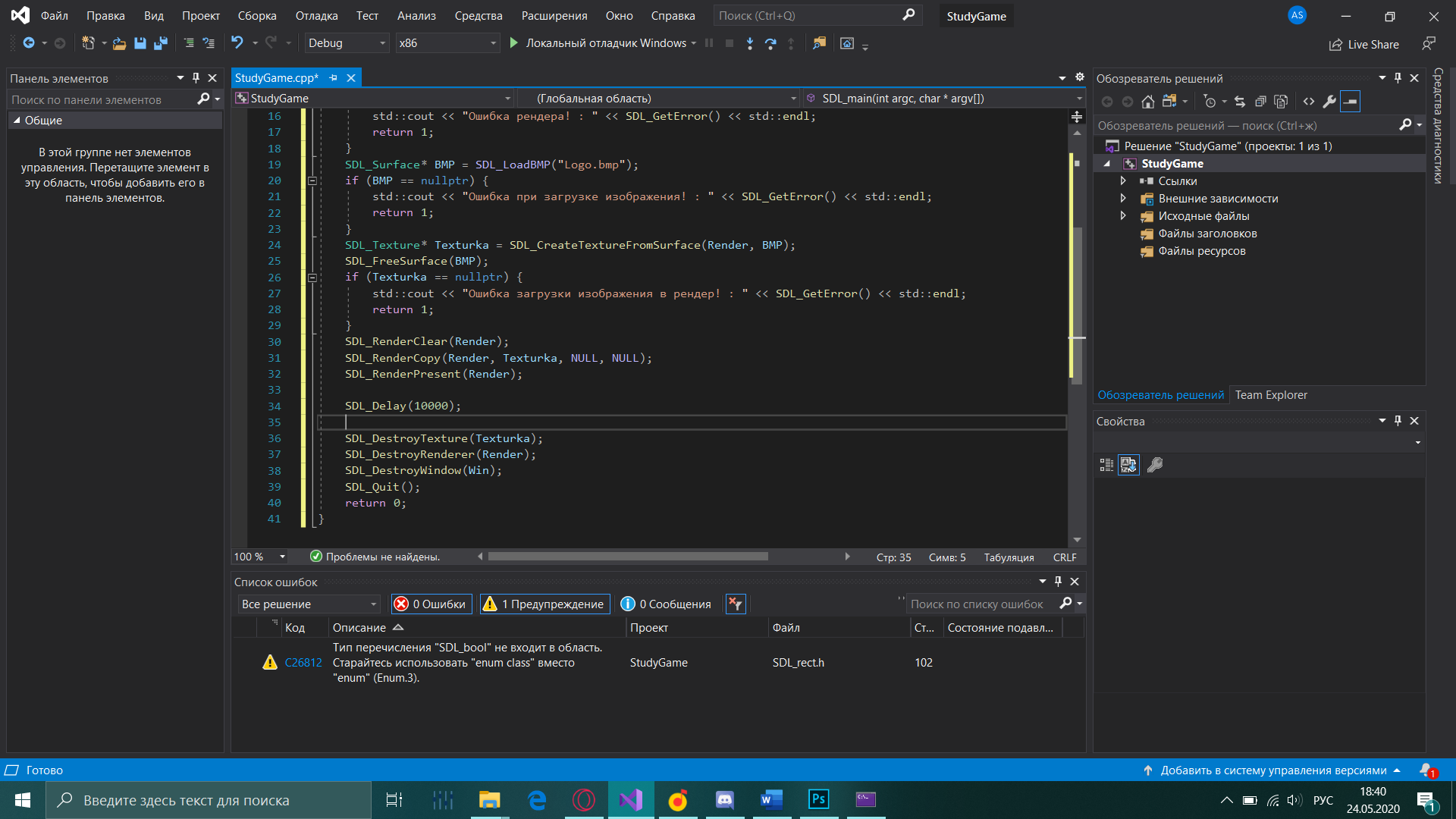
Прежде чем мы вставим нашу картинку формата BMP в рендер надо загрузить ее в память. С этим нам поможет функция [**SDL\_LoadBMP**](http://wiki.libsdl.org/moin.fcg/SDL_LoadBMP) (параметром должен быть путь к нужной картинке или просто название, если она уже в папке проекта), она возвращает  **[SDL\_Surface\*](http://wiki.libsdl.org/moin.fcg/SDL_Surface)**, потому вновь создаем указатель данного типа. И делаем проверку на то, не пуст ли наш указатель



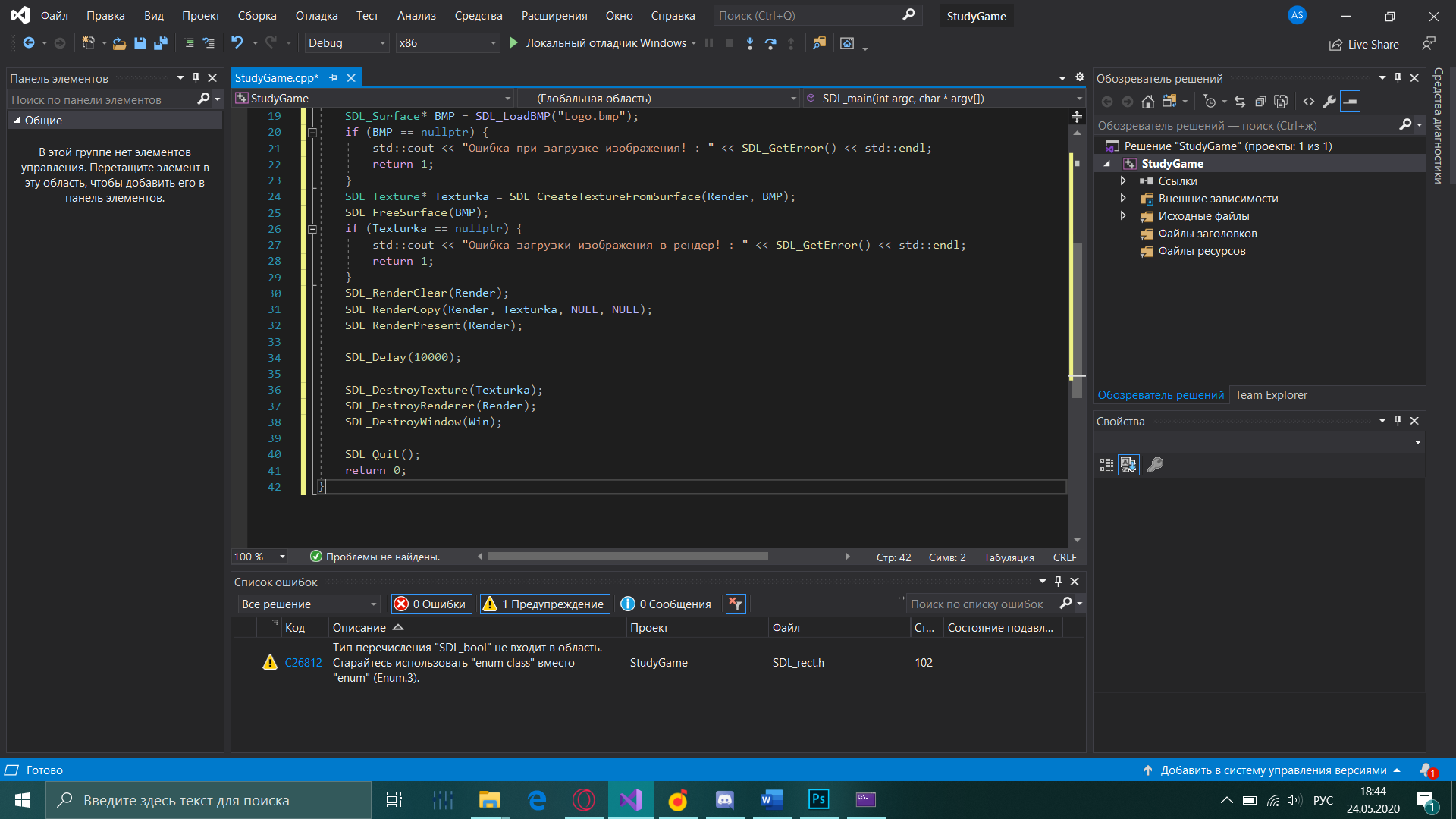
Теперь мы загружаем наше изображение, которое находится в оперативной памяти, в рендер. Для этого нам нужна команда [**SDL\_CreateTextureFromSurface**](http://wiki.libsdl.org/moin.fcg/SDL_CreateTextureFromSurface), которая первым параметром принимает ранее созданный нами рендер (указатель на него) и указатель на ту часть памяти, где хранится наша картинка. Возвращает эта функция **SDL\_Texture\***, потому создаем указатель **Texturka** данного типа. Также делаю проверку на то, что все успешно записалось в наш указатель, ведь иначе он будет пустым (nullptr).   
Далее нам нужно освободить память, которую мы ранее выделяли под хранения изображения (не путаем с памятью под текстуру). Для освобождения памяти нам понадобится функция **SDL\_FreeSurface**



Последним шагом осталось вывести изображение на экран. Вначале мы чистим наш рендер с помощью функции **SDL\_RenderClear**, в которую передаем наш указатель на рендер. Это необходимо, чтобы он не нарисовал нам мусор, образованный внутри него в прошлых этапах.   
Теперь мы используем функцию **SDL\_RenderCopy**, чтобы отрендерить нашу текстуру. В качестве параметров мы указываем указатель на наш рендер, указатель на текстуру и нужный размер (пишем Null, Null чтобы он растянул нашу картинку на весь экран)   
Далее выводим то, что мы сделали в рендере, на экран с помощью **SDL\_RenderPresent**, где параметр – это указатель на наш рендер.   
Теперь пишем функцию **SDL\_Delay**, чтобы наше окно было открыто еще некоторое время перед тем, как мы устроим чистку. 1000 параметром – 1 секунда, потому ставлю на 10 секунд

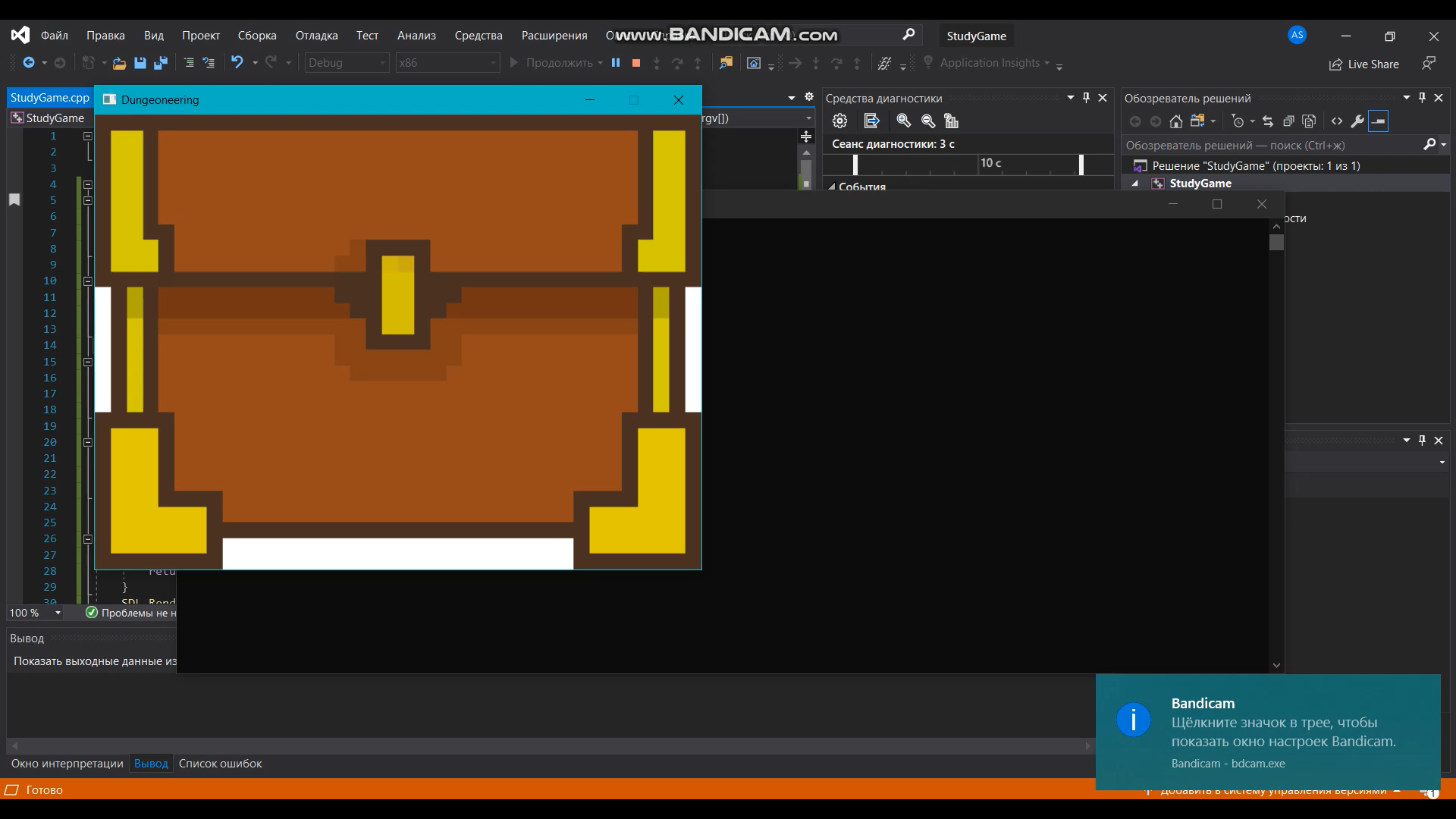


А теперь устраиваем чистку для освобождения памяти. Память под Surface картинки мы уже чистили, остались лишь Render, Texturka и окно Win. **SDL\_DestroyRenderer, SDL\_DestroyTexture, SDL\_DestroyWindow** соответственно.

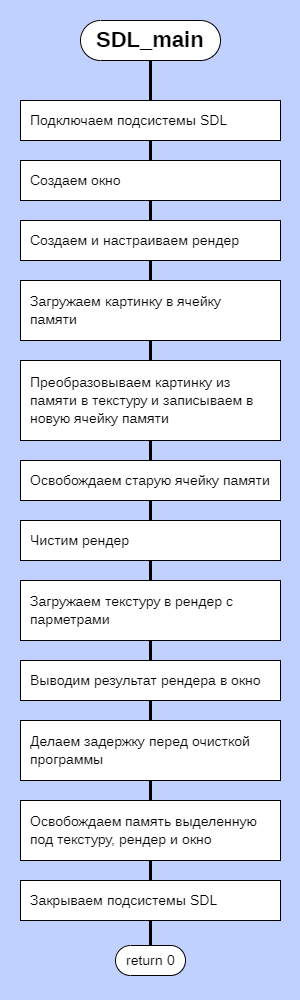


И теперь закрываем подсистемы SDL и возвращаем 0 нашему main.

Если мы все сделали правильно, то программа на 10 секунд выведет нашу картинку на окно



**Готово!**

****