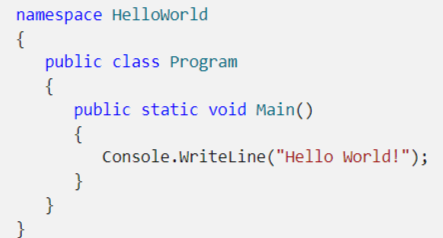
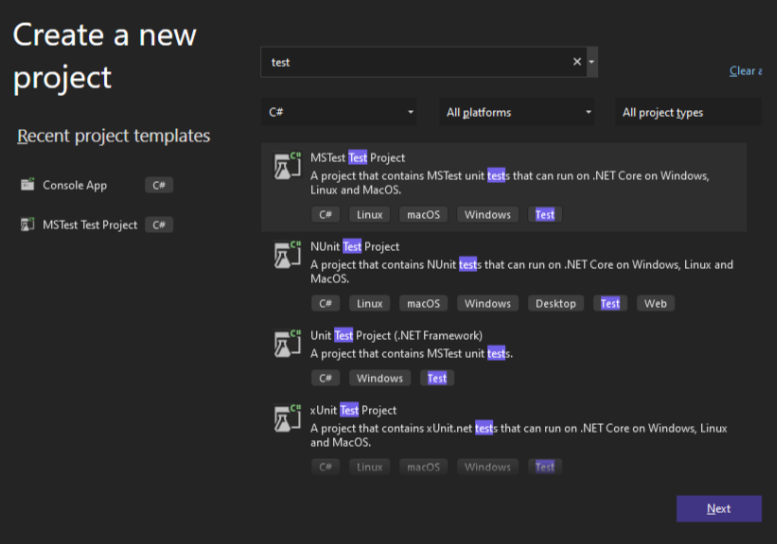
В целях демонстрации примера модульного теста протестируем простой консольный проект C# или "Hello World" с именем **HelloWorld**. Пример кода для такого проекта выглядит следующим образом:

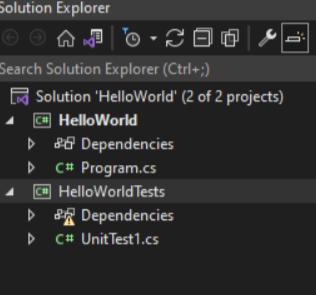


1. Выберите узел решения в обозревателе решений. Затем в верхней строке меню выберите Файл>Добавить>Новый проект.
2. В диалоговом окне нового проекта найдите проект модульного теста, который хотите использовать.

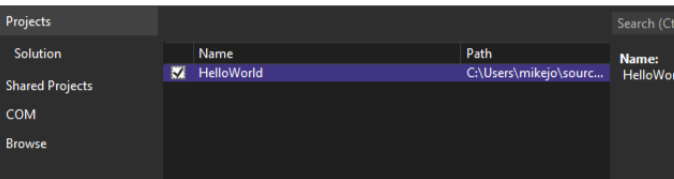
Введите test в поле поиска, чтобы найти шаблон проекта модульного теста для тестовой среды, которую вы хотите использовать, например MSTest (C#) На следующем рисунке показано модульный тест MSTest, поддерживаемый в .NET.



Проект добавляется в решение.

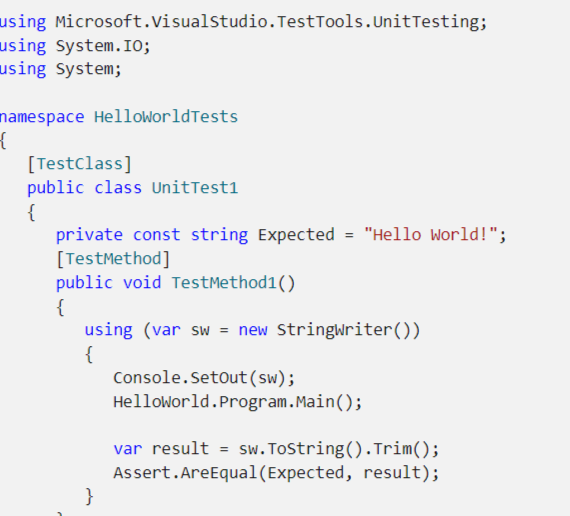


1. В проекте модульного тестирования добавьте ссылку на проект, который вы хотите протестировать, щелкнув правой кнопкой мыши **Ссылки** или **Зависимости**, после чего выбрав **Добавить ссылку** или **Добавить ссылку на проект**.
2. Выберите проект, содержащий код, который будет тестироваться, и нажмите **OK**.



Добавьте код в метод модульных тестов.

Так вы можете использовать следующий код, выбрав правильную вкладку документации, которая соответствует вашей тестовой среде: MSTest, NUnit или xUnit (поддерживается только в .NET)

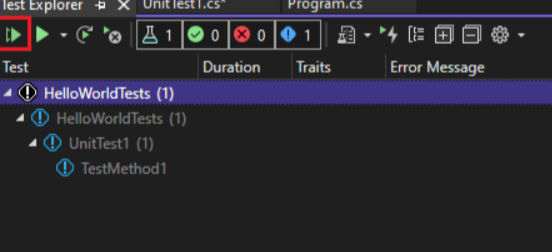


Выполнение модульных тестов

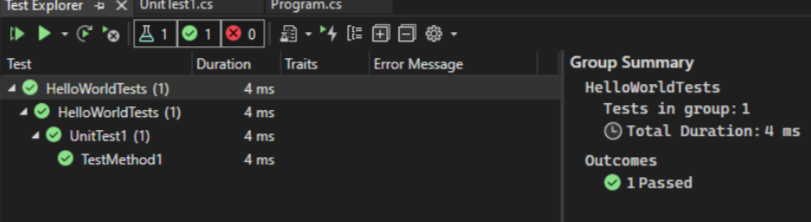
1. Откройте [обозреватель тестов](https://learn.microsoft.com/ru-ru/visualstudio/test/run-unit-tests-with-test-explorer?view=vs-2022).

Откройте обозреватель тестов, выбрав Тест>Обозреватель тестов в верхней строке меню (или нажмите клавиши CTRL + E, T).

1. Запустите модульные тесты, нажав Запустить все (или нажмите клавиши CTRL + R, V).



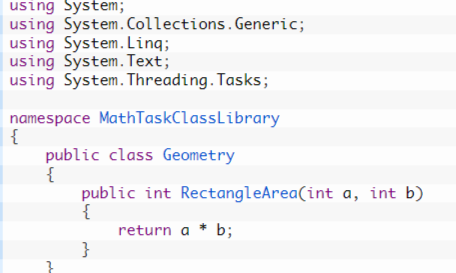
После завершения зеленый флажок указывает, что тест пройден. Красный значок "x" указывает на сбой теста.



Теперь разработаем проект содержащий класс, который вычисляет площадь прямоугольника по длине двух его сторон.

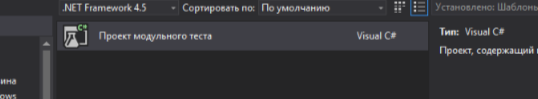
Создадим в Visual Studio новый проект Visual C# -> Библиотека классов. Назовём его MathTaskClassLibrary. Class1 переименуем в Geometry.

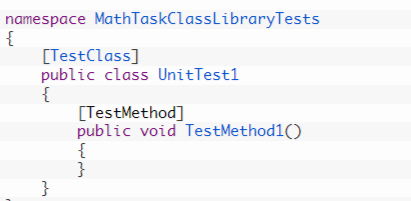
В классе реализуем метод, вычисляющий площадь прямоугольника. Для демонстрации остановимся на работе с целыми числами. Код программы приведён ниже.



Правой кнопкой щёлкните по решению, выберите «Добавить» и затем «Создать проект…».

В открывшемся окне в группе Visual C# щёлкните «Тест», а затем выберите «Проект модульного теста». Введите имя проекта **MathTaskClassLibraryTests**и нажмите «ОК». Таким образом проект будет создан.



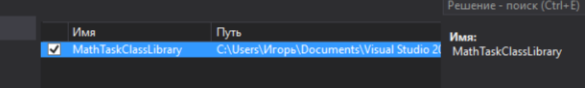


Директива [TestMethod] обозначает, что далее идёт метод, содержащий модульный (unit) тест. А [TestClass] в свою очередь говорит о том, что далее идёт класс, содержащий методы, в которых присутствуют unit-тесты.

В соответствии с принятыми соглашениями переименуем класс UnitTest1 в GeometryTests.

Затем в References проекта необходимо добавить ссылку на проект, код которого будем тестировать. Правой кнопкой щёлкаем на References, а затем выбираем «Добавить ссылку…».

В появившемся окне раскрываем группу «Решение», выбираем «Проекты» и ставим галочку напротив проекта MathTaskClassLibrary. Затем жмём «ОК».



Также в коде необходимо подключить с помощью директивы using следующее пространство имён: using MathTaskClassLibrary;

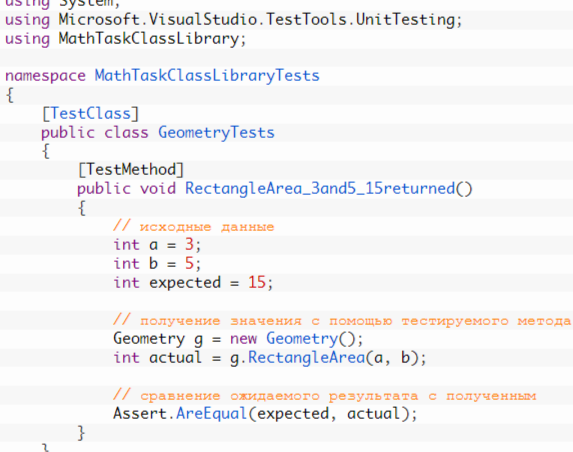
Займёмся написание теста. Проверим правильно ли вычисляет программа площадь прямоугольника со сторонами 3 и 5. Ожидаемый результат (правильное решение) в данном случае это число 15.

Переименуем метод TestMethod1() в RectangleArea\_3and5\_15returned(). Новое название метода поясняет, что будет проверяться (RectangleArea — площадь прямоугольника) для каких значений (3 и 5) и что ожидается в качестве правильного результата (15 returned).

Тестирующий метод обычно содержит три необходимых компонента:

1. исходные данные: входные значения и ожидаемый результат;
2. код, вычисляющий значение с помощью тестируемого метода;
3. код, сравнивающий ожидаемый результат с полученным.

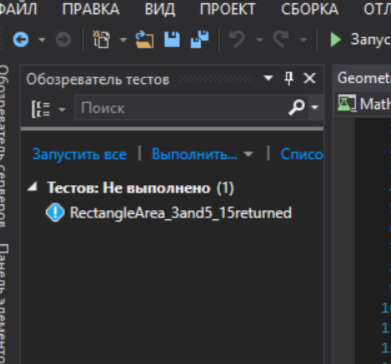
Соответственно тестирующий код будет таким:



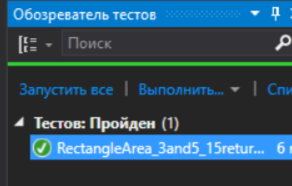
Для сравнения ожидаемого результата с полученным используется метод AreEqual класса Assert (Приложение А). Данный класс всегда используется при написании unit тестов в Visual Studio.

Теперь, чтобы просмотреть все тесты, доступные для выполнения, необходимо открыть окно «Обозреватель тестов». Для этого в меню Visual Studio щёлкните на кнопку «ТЕСТ», выберите «Окна», а затем нажмите на пункт «Обозреватель тестов».

В данный момент список тестов пуст, поскольку решение ещё ни разу не было собрано. Выполним сборку нажатием клавиш Ctrl + Shift + B. После её завершения в «Обозревателе тестов» появится наш тест.



Синяя табличка с восклицательным знаком означает, что указанный тест никогда не выполнялся. Выполним его. Для этого нажмём правой кнопкой мыши на его имени и выберем «Выполнить выбранные тесты».



Зелёный кружок с галочкой означает, что модульный тест успешно пройден: ожидаемый и полученный результаты равны.

Изменим код метода RectangleArea, вычисляющего площадь прямоугольника, чтобы сымитировать провал теста и посмотреть, как поведёт себя Visual Studio. Прибавим к возвращаемому значению 10.

Запустим unit-тест.

Задание: написать 5 программ как показано в примерах выше и протестировать их. Написать test-cases.

Приложение А

Класс Assert является одним из самых часто применяемых, поэтому его важные методы кратко описаны в таблице ниже:

|  |  |
| --- | --- |
| Метод | Описание |
| AreEqual<T>(T, T); AreEqual<T>(T, T, string) | Утверждает, что два объекта типа T имеют одно и то же значение |
| AreNotEqual<T>(T, T); AreNotEqual<T>(T, T, string) | Утверждает, что два объекта типа T не имеют одно и то же значение |
| AreSame<T>(T, T); AreSame<T>(T, T, string) | Утверждает, что две переменные ссылаются на один и тот же объект |
| AreNotSame<T> (T, T); AreNotSame<T>(T, T, string) | Утверждает, что две переменные ссылаются на разные объекты |
| Fail(); Fail(string) | Отрицательный результат утверждения - никакие условия не проверены |
| Inconclusive(); Inconclusive(string) | Показывает, что результат модульного теста не может быть однозначно установлен |
| IsTrue(bool);  IsTrue(bool, string) | Утверждает, что булевское значение равно true - чаще всего используется для оценки выражения, возвращающего булевский результат |
| IsNull(object);  IsNull(object, string) | Утверждает, что переменная не присвоена объектной ссылке |
| IsNotInstanceOfType(object,Type);  IsNotInstanceOfType(object, Type, string) | Утверждает, что объект не относится к указанному типу |

Каждый статический метод в классе Assert позволяет проверить какой-то аспект модульного теста, и если проверка не проходит, эти методы генерируют исключение. Чтобы модульный тест прошел, все утверждения должны завершиться успешно.

Каждый метод из таблицы имеет перегруженную версию, которая принимает параметр string. В случае отрицательного результата утверждения эта строка помещается в элемент сообщения внутри объекта исключения. Методы AreEqual и AreNotEqual имеют несколько перегруженных версий, предназначенных для сравнения специфических типов. Например, существует версия, которая позволяет сравнивать строки без учета регистра символов.

Одним заслуживающим внимания членом пространства имен Microsoft.VisualStudio.TestTools. UnitTesting является атрибут ExpectedException. Это утверждение, которое дает положительный результат, только если модульный тест генерирует исключение с типом, указанным в параметре ExceptionType. Данный атрибут служит надежным способом обеспечения генерации исключений без необходимости в наличии блоков try...catch внутри кода модульного теста.