In this exercise you will be using Microsoft Unit Test framework to test a class.

# You must follow the specifications exactly (To demo to instructor at the end of this class)

# Create a library containing the Fraction class

After compiling a console application project, the output will be an executable. A library project on the other hand will produce a dll file(s). This is because a library template compiles to a dynamic linked library. Hence you will not be able to run this project.

This class consist of 6 members.

|  |
| --- |
| **Fraction**  **Class** |
| **Properties** |
| **+ <<C# property>>Top : int**  **+ <<C# property>>Bottom : int** |
| **Methods** |
| **+ <<constructor>> Fraction(int top = 0, int bottom = 1)**  **+$ <<operator>> +( Fraction left, Fraction right): Fraction**  **+$ <<operator>> -( Fraction left, Fraction right): Fraction**  **+ ToString() : string** |

#### Description of class members

##### Fields:

Some common decorators for class members

**+** → public  
**#** → protected  
**-** → private  
**$** → class member (static)

There are no fields

##### Properties:

All the properties have public getters and the setters are absent.

**Top** – this property represents the numerator of this object.

**Bottom** – this property represents the denominator of this object

##### Constructor:

**Fraction(int top = 0, int bottom = 1)** – This constructor takes two optional parameters and assigns them to the appropriate properties.

##### Operators:

There are two overloaded operators: the addition and the subtraction operators

**public static Fraction operator +(Fraction left, Fraction right)** – This will implement the addition operation.

Look at the Complex Number Lab to for hints on operator overloading

**public static Fraction operator -(Fraction left, Fraction right)** – This will implement the subtraction operation. ~~This will throw an exception if the right argument is larger than the left side argument. i.e if the resulting difference will be negative~~.

##### Methods:

**public override string ToString()** – This is a public method overrides the corresponding method in the object class to return a stringify form of the object. You get to decide how the properties will be display to the user.

Try to keep the output on a single line.

### Unit Testing

Remember in testing you will compare the expected value to the actual value. You need to write the testing methods for the following:

If you are having problems accessing your **Fraction** class from your testing project, or running your tests check the following:

* Verify that the Fraction class is public
* Ensure a reference to the library is added to your testing project
* Add the necessary using statement
* Confirm that you are **NOT** running from an unsecured drive such as the network H drive

//1, 2 & 3 – Constructor. You must test the three cases: creating a fraction with two ints, with one int and with no argument

//arrange

//declare and suitably initialise two int: expectedTop and expectedBottom

//act

//create a fraction using the above two arguments

//assert

//compare the expectedTop with the Top property of the above object

//compare the expectedBottom with the Bottom property of the above fraction

Note: To test this constructor exhaustively, you need to check three conditions: constructor with zero, one and two arguments

**[TestMethod]**

**public void Constructor\_WithZeroArgument()**

**{**

**//arrange**

**int expectedTop = 0, expectedBottom = 1;**

**//act**

**Fraction f = new Fraction()**

**//assert**

**Assert.AreEqual(expectedTop, f.Top);**

**Assert.AreEqual(expectedBottom, f.Bottom);**

**}**

//4 – ToString.

//arrange

//declare and initialise two ints

//declare the variable expectedString that will depend on your ToString() method.

//act

//declare and create an object using the above two ints

//assert

//compare the expectedString variable to the actual output of the ToString() method

.

//5 – Addition.

You may use the following sample for your addition

//6 – Subtraction without exception.

You may use the following sample for your subtraction

~~//7 - Subtraction with exception.~~

~~You may use the following sample for your subtraction~~