PROG8170

Software Quality Assurance Techniques

Assignment #1: Unit Testing – Due September 25th, 2018 @ 1:00PM – In Class

This assignment must be done **individually** and will be marked out of sixty marks (see the associated Rubric). Do your own work and **do not share** your work with others. Do not use another student's computer for this (or any) assignment. Sharing work is an academic offense and is subject to penalty. Be aware that source code and documents are automatically checked by eConestoga against every other student's work in the course. Academic Offenses will be reported to the College Registrar.

You are to create a C# console application that allows a user to work with and modify a rectangle. You will use Git as source control for this project. Begin by creating a new console application project in Visual Studio, select a location on your local harddrive. **Make note of the folder where you created the Solution**. Once this has been completed, use Git Bash to initialize a git repository within the Solution folder you just created. You know you are within the correct folder if you run an **`ls`** command in Git Bash and see a **`<SolutionName>.sln`** file and the folder **`<ProjectName>`** (there may be a few other files and folders, but these two will definitely be there). Do your first, initial commit of your solution folder in Git. Return to Visual Studio and begin work on your application. **Begin by creating a new Rectangle class as a separate file**. The Rectangle class should have two private integer attributes which hold the length and the width of the rectangle object. The Rectangle class should contain a default constructor which sets the length and width each to the value of 1, and a non-default constructor which sets the length and width to whatever the user inputs.

The Rectangle class must be public, and should contain six methods, plus a default and non-default Constructor. The six methods are:

```
public int GetLength()
public int SetLength(int length)
public int GetWidth()
public int SetWidth(int width)
public int GetPerimeter()
public int GetArea()
```

Each method should do the action as described by its method name. The SetLength method overwrites the old length with the new length. The SetWidth method overwrites the old width with the new width. When you have completed work on the Rectangle class return to Git Bash. Stage and commit your entire solution, and label the commit as containing the completed Rectangle class.

Return to Visual Studio and continue work on your application. You must now create a console application to allow a user to create and work with a single rectangle object. By default when the program loads, it asks the user to please enter the length and width of the rectangle. Each side must be an integer greater than zero. If any incorrect input is provided, the user is informed of their mistake and asked to please enter a value again. This error handling and input validation can be done within the console application, rather than the Rectangle class methods.

The program should then present the user with a menu with seven options:

1. **Get Rectangle Length**
2. **Change Rectangle Length**
3. **Get Rectangle Width**
4. **Change Rectangle Width**
5. **Get Rectangle Perimeter**
6. **Get Rectangle Area**
7. **Exit**

If any incorrect input is given, the menu is shown again. Each of the first six options should work as described. The exit option quits the program, and is the only way to exit the program (other than closing the window).

The length and width of the rectangle cannot be changed to a value less than one.
When you have completed work on the console application return to Git Bash. Stage and commit your entire solution, and label the commit as containing the completed console application.
Return to Visual Studio and begin work on creating unit tests for each of the six regular methods within the Rectangle class. You do not need to test the constructors. This work should be done within its own Project, as described in class. Your solution in Visual Studio should look something like the following:

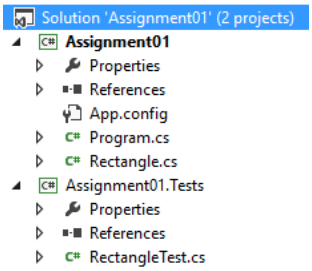

Figure 1. Example Assignment 01 Solution

You have your Main Project, which contains the console app code (Program.cs in Figure 1) and the Rectangle class (Rectangle.cs in Figure 1). You will have a second project in the same solution containing the test cases. **Remember the naming conventions** for the Test Project, the Test Class, and the Test Methods.
When you have completed work on the unit tests return to Git Bash. Stage and commit your entire solution, and label the commit as containing the completed unit tests.
Provide a screenshot of the Unit cases being run (can take a screenshot of the NUnit GUI after a run). All of your unit tests should pass.
Provide a screenshot or output from a `git log` command showing your three commits. Feel free to run more commits as it makes sense for your particular project and your work schedule. But at minimum there should be the three commits as described in the assignment.

I will be coming to each student individually to see a working demo of your program in action as well as seeing the execution of the NUnit tests running.

The format for submitting the assignment is as follows:

1. **Printouts Handed in Class in this order**:
    a. Assignment Cover sheet with your name, student ID, “Assignment #1” in the title and date
    b. Assignment Rubric left blank (found on eConestoga)
    c. Copy of your Program source code (The Program.cs in example Figure 1 screenshot)
    d. Copy of your Rectangle class source code (The Rectangle.cs in example Figure 1 screenshot)
    e. Copy of your Unit Test class source code (The RectangleTest.cs in example Figure 1 screenshot)
    f. Print out showing the screenshot with results of your unit tests being run, and a screenshot/output of your Git repository log.
2. **eConestoga Submission**: A single compressed (.zip format) archive file containing **the entire Solution folder** of your source code (so I can run it) and the doc file with the screenshots.