PROG2070 – Quality Assurance: Winter 2017

Assignment 01

[Maximum points: 30]

Due Date: Week of Feb. 6, 2016, in Class

This assignment should be done **individually**. Do your own work and **do not share** your work with others. 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 square. 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 G: (if on a lab computer, if you are using your own computer select a location anywhere). **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 Square class as a separate file. The Square class should have a single private integer attribute which holds the side length of the square object. The Square class should contain a default constructor which sets the side length to 1, and a non-default constructor which sets the side length to whatever the user inputs.

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

```
public int GetSide()

public int ChangeSide(int side)

public int GetPerimeter()

public int GetArea()
```

Each method should do the action as described by its method name. ChangeSide overwrites the old side length with the new side length.

When you have completed work on the Square class return to Git Bash. Stage and commit your entire solution, and label the commit as containing the completed Square 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 square object. By default when the program loads, the program asks the user to please enter the side length of the square. A side must be an integer greater than zero. If any incorrect input is give, the user is informed of their mistake and asked to please enter a

side length again. This error handling and input validation can be done within the console application, rather than the Square class methods.

The program should then present the user with a menu with 5 options:

1. **Get Square Side Length**
2. **Change Square Side Length**
3. **Get Square Perimeter**
4. **Get Square Area**
5. **Exit**

If any incorrect input is given, the menu is shown again. Each of the first four 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 side length of the square 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 four regular methods within the Square 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:

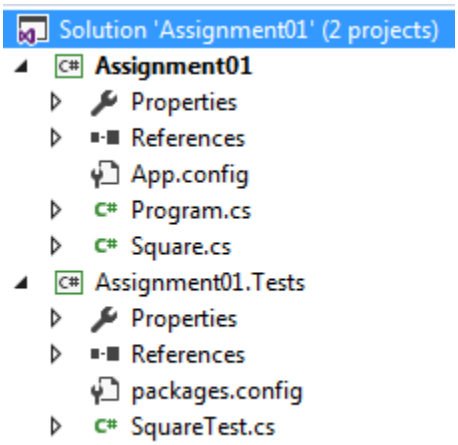

Figure 1. Example Assignment 01 Solution

You have your Main Project, which contains the console app code (Program.cs in Figure 1) and the Square class (Square.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.

The format for submitting the assignment is as follows:

1. **Printouts Handed in Class**:
    a. Assignment Cover sheet properly filled (found on eConestoga)
    b. Assignment Rubric left blank (found on eConestoga)
    c. Copy of Program source code (The Program.cs in example Figure 1 screenshot)
    d. Copy of Square class source code (The Square.cs in example Figure 1 screenshot)
    e. Copy of Unit Test class source code (The SquareTest.cs in example Figure 1 screenshot)
    f. Print out of Doc 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.