CSCI 1301 – Lab 08

September 10, 2018

1 Finishing Previous Lab

Go back to the previous lab and make sure you properly "enriched" `Rectangle.cs`. You can compare your answer with this proposition: Rectangle_Sol.zip.

2 Writing Your Own Class

In this exercise, you will create your own first class instead of using and expanding one that was written for you. The idea is to take inspiration from the class you already know (`Rectangle`) to create a new class, called `PreciseRectangle`, that will manipulate rectangles whose width and length are floating-point values, instead of integers (as in `Rectangle`).

2.1 Conception

Draw the UML diagram of this class: it should have two attributes, of type `double`, and five methods: two setters, two getters (i.e., one for each attribute), and one method to compute the area of a precise rectangle.

2.2 Implementation

To implement your method in VS, I explain two methods below: you can edit the pre-existing project, or start "fresh". I recommend that you pick the one you feel the most comfortable with, and then try the other one.

2.2.1 Edit the Pre-Existing Project

1. Re-download the `Rectangle`[1] project, extract it in a folder, open it with VS.
2. Within VS, re-name the project to "PreciseRectangle", rename the "Rectangle.cs" file to "PreciseRectangle.cs"
3. In the "PreciseRectangle.cs" file, replace `class Rectangle` with `class PreciseRectangle`.
4. Comment out the body of the `Main` method in "Program.cs".
5. Your program should compile as it is, but you have to edit `PreciseRectangle.cs` to now store the `width` and the `length` with `double`, and to propagate this change accordingly. What should be the return type of `GetWidth`, for instance?
6. Declare and manipulate precise rectangles (i.e., with floating-point values for the width and the length) in the `Main` method, and make sure they behave as expected (can you compute the area, for instance?).

[1] Rectangle.zip

2.2.2 Starting From Scratch

1. Create a new project in VS, name it "PreciseRectangle".
2. In the Solution Explorer, right-click on "PreciseRectangle", then on "Add…" and select "Class". Then, select "Class", write "PreciseRectangle.cs" as the name of the file, and click on "Add".
3. You are now suppose to have two ".cs" files opened and displayed in the Solution Explorer: "Program.cs" and "PreciseRectangle.cs".
4. Implement the `PreciseRectangle` class according to your UML diagram.
5. Declare and manipulate rectangles with floating-point values for the width and the length in the `Main` method, and make sure they behave as expected (can you compute the area, for instance?).

3 Pushing Further (Optional)

The following are two independent tasks, to widen your understanding of this class, and to prepare you for the next labs.

1. Class diagram (the one we will be using) are just a special case of UML diagram. Have a look at https://en.wikipedia.org/wiki/Unified_Modeling_Language#Diagrams. In which category are class diagram: behaviour, or structure diagram? Have a look at https://en.wikipedia.org/wiki/Activity_diagram and try to read the example of activity diagram for a guided brainstorming process.
2. Now that you know more about naming convention, have a look at https://docs.microsoft.com/en-us/dotnet/standard/design-guidelines/naming-guidelines, and particularly at

- https://docs.microsoft.com/en-us/dotnet/standard/design-guidelines/general-naming-conventions
- https://docs.microsoft.com/en-us/dotnet/standard/design-guidelines/capitalization-conventions