Contents

Warm-up Exercises

Questions

1. Explain the meaning/purpose of a constructor.
   Solution
   A constructor is used to instantiate or "construct" an object from the class that contains it.
2. Consider a fragment of longer C# code, see below.

```
public void SetNumber(int tempNumber)
{ name = tempNumber; } // store the account name
```

1. Which of the following statements is false?

☐ the method returns no value
☐ name is a local instance variable
☒ the parameter tempNumber is of string type
☐ the method can be used outside of the class it is defined in

1. C# is an object oriented language that has roots in:

☐ C
☐ C++
☐ Java
☒ All above.

1. Fill in the blanks: "A class asserts that every object created using it should have _____ (i.e., 'data') and _____ (i.e., 'operations')."

Solution

attributes, methods

1. Give two access modifiers.

Solution

private, public, internal

1. What is the purpose of the keyword **new**?

Solution

The keyword "new" is used for instantiating an object.

1. What does the keyword **`return`** do?

Solution

The keyword "return" ends the execution of a method and, if the method has a non-void return type, returns a value of that type.

1. What does it mean to say that instance variables have a default initial value? How is that different from the variables we have been manipulating in the `Main` method?

Solution

Instance values have a default initial value if they are assigned a value specified within the class definition. Variables in the "Main" method will only default to the value specified by their type.

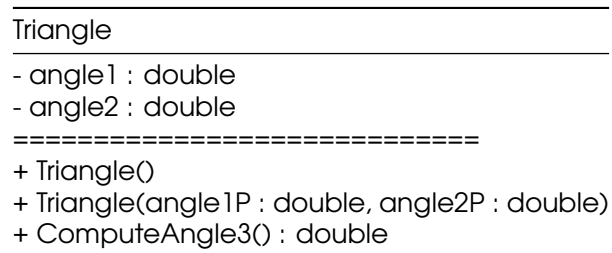
Problems

1. You are going to design a class named `Triangle`. A triangle only has three angles, but knowing the value of only two angles is sufficient to determine the value of the third, since they always add up to 180°. Hence, it is sufficient to have only two **double** attributes, `angle1` and `angle2`. We want to define several methods:

- a no-arg constructor that sets the value of `angle1` to 60.0 and the value of `angle2` to 60.0,
- another constructor that takes two arguments, and assigns to `angle1` the value of the first argument, and to `angle2` the value of the second argument,
- getters for `angle1` and `angle2`,
- a method that computes and returns the value of the third angle, that we name `ComputeAngle3`,
- a method that rotates the triangle: the value of the first angle should become the value of the second angle, and the value of the second angle should become the value of the third angle.

1. Write the UML diagram for the `Triangle` class.
2. Write the full, compilable implementation of the `Triangle` class.

Solution for Part 1



Triangle
+ RotateTriangle() : void

Solution for Part 2

```
class Triangle
{
    private double angle1, angle2;

    public Triangle()
    {
        angle1 = 60.0;
        angle2 = 60.0;
    }

    public Triangle(double angle1P, double angle2P)
    {
        angle1 = angle1P;
        angle2 = angle2P;
    }

    public double GetAngle1()
    {
        return angle1;
    }
    public double GetAngle2()
    {
        return angle2;
    }

    public double ComputeAngle3()
    {
        return 180 - (angle1 + angle2);
    }

    public void RotateTriangle()
    {
        double angle3 = ComputeAngle3();
        angle1 = angle2;
        angle2 = angle3;
    }

}
```