Notes

Exercises about classes

- Solve them in Visual Studio.

Exercise 12.01

- The namespace of your project is “Animals”.
- You have a class “TestDog” that contains the Main().
  - This tests the class “Dog”.
    - Create 2 or more instances of a dog.
    - Test the full functionality of a dog using those instances.
- Create an extra class with the name “Dog”.
  - You can decide where.
- The class “Dog” has 4 variables with corresponding properties. These properties are no automatic properties and do have gets and sets.
  - Age (of the dog).
  - Breed / Kind (of the dog).
  - Color (of the dog).
  - Name (of the dog).
- The class “Dog” has multiple constructors. You have to use the properties in the constructors. Do not use the variables.
  - One with no parameters.
  - One with only the name of the dog.
  - One with all four parameters / properties / variables.
- The class has extra methods.
  - Bark.
  - Sit.
  - Eat.

Notes

....

....

....

....

....

....

....

        - These methods just show something on the console, so you are able to test the methods.
- There is also 1 property that counts the number of dogs.
    - This is a static property.
    - Make sure that all the constructors call the routine where you add 1 to that property.
    - The property that counts the number of dogs is a property that has a read only behaviour. Only the get is implemented.
- In the test routines, all properties and methods must be tested on get and set.
    - Getting the value and setting the value.
    - There is one exception, the number of dogs.

Variant 1

- The same exercise, but with automatic properties.
    - See slide 12 from Part 05 – C# Class Fields and Properties.

Notes

....

....

....

....

....

....

....

Exercise 12.02

- The namespace of your project is “Classroom”.
- You have a class “Person”.
  - The person has a firstname, lastname and a birthdate.
    - These are all properties with get and set.
  - The person has 1 constructor, that has 3 parameters.
    - The firstname, the lastname and the birthdate.
  - The person has also dynamic properties “AgeInYears” and “AgeInDays”. In a previous example, you have already calculated the age in days and years.
    - See slide 11 from C# Class Fields and Properties – Part 05.
- This must give back, whatever I try with the objects of type Person, a correct result.
- You also create a test routine that proves that it works.

Exercise 12.03

- Create a class “Chicken”.
- This class has one property (get and set).
  - The number of eggs that chicken can lay in one day.
  - This number is 0, 1 or 2.
  - When a negative number is tried to set to that property, the number becomes 0.
  - When a number higher than 2 is tried to set to that property, the number becomes 2.
- Add a constructor to it, no parameters.
  - By default, a chicken can lay 1 egg.
- Add a method that shows the number of eggs that a chicken can lay.
  - Just information on the console is good enough.

Notes

....

....

....

....

....

....

....

Exercise 12.04

- Create a class “Rectangle”.
  - A rectangle has a width and a height.
  - Both values must be positive (zero or larger).
  - When a negative number is given to that property, you change it to the default 1.
  - I need 2 constructors.
    - One without parameters, both width and height become 1 as default.
    - One with 2 parameters, the width and the height.
  - I need 2 dynamic properties.
    - One that gives the circumference / perimeter.
      - The sum of the length of all sides.
    - One that gives the area.
- Write also a test class that proves that the functionality works.

Exercise 12.05

Part 1

- Create a class that defines a point.
- Put this class in the namespace “Mathematics”.
- The point has 2 coordinates.
    - An x (type double).
    - An y (type double).
- The constructor of a point puts randomly a number to the “x-coordinate” and also randomly a number to the “y-coordinate”.
    - Round the numbers to 6 digits after the decimal symbol.
        - So 12,123456 is a correct coordinate.
- You have also for x and y properties with a get and a set.
- Write a test program that checks if this works fine.

Part 2

- Create a class that defines a line.
    - The code must be in a different file in your project.
- Put this class also in the namespace “Mathematics”.
- A line has 2 points.
    - A startpoint (a property of the class line).
    - An endpoint (a property of the class line).
    - Startpoint and endpoint must be different.
- Create a constructor that makes sure that you have a line with a different startpoint and endpoint.
    - So you have randomly 2 different points.
    - Make sure that they are different, if not, repeat to create randomly a point until it is different than the first one you created.
- Create a dynamic property that returns the length of the line.

Notes

Notes

....

....

....

....

....

....

....

- See below for the calculation.

- Write a test program that checks if this works fine.

How to calculate the length of a line?

- If point 1 has coordinates x1 and y1.
- If point 2 has coordinates x2 and y2.
- The length is the square root of the squared difference of the x-coordinates and the squared difference of the y-coordinates.
  - This is Pythagoras ($a^2 + b^2 = c^2$).

$$Length = \sqrt{(x2 - x1)^2 + (y2 - y1)^2}$$

Example

- If point 1 has coordinates 1 and 4.
- If point 2 has coordinates 5 and 1.

$$5 \;=\; Length = \sqrt{(5 - 1)^2 + (1 - 4)^2}$$