Exercises about classes

Solve them in Visual Studio.

Exercise 11.01: Who are you, what's your name?

- The namespace of your project is "LearnCSharp".
- You have a class "Program" that contains the Main().
  - This tests the class "Person".
    - Create 2 persons.
    - Show the full name of the 2 persons.
- Create an extra class with the name "Person".
  - Do this in the same file as the Main().
- The class Person has 2 variables.
  - One for the first name. Choose a good variable name.
  - One for the last name. Choose a good variable name.
  - The 2 parameters must be invisible inside the Main() routine.
    - Tip: Make them private.
- The class Person has a constructor.
  - This must be the same name as the class.
  - 2 input parameters.
    - strFirstName (string).
    - strLastName (string).
- The class has one extra method "ShowFullName".
  - This shows the FirstName and the LastName with a space in between.

Variant

- This is the same exercise.
- But the class Person is a different file.
  - You have an extra file called Person.cs.

Notes

Exercise 11.02: The ark of Noah

- In the slides of Part 1 – C# Class Object you can find a code example with the namespace ArkOfNoah.
- This example allows only 2 animals of the same kind (being a panda) on the ark.
- Change the code, so that only 2 animals of the same kind, that have a different sex, are allowed.

Tip

- When the first animal is male, the second animal that is allowed must be a female.
- But also the other way around, when the first animal is a female, the second animal that is allowed must be a male.
- You will need a variable to know what the sex of the animal is.
- You will need to change the criteria to find out when an animal is accepted to the ark.

Notes

Exercise 11.03: My name is Bond, …, James Bond

Restart from scratch.

You must learn the principle. You are not learning to "Copy Paste".

- The namespace of your project is "LearnCSharp".
- You have a class "Program" that contains the Main().
  - This tests the class "Person".
    - Create 2 persons.
    - Show the full name of the 2 persons.
- Create an extra class with the name "Person".
- The class Person has 2 variables.
  - One for the first name. Choose a good variable name.
  - One for the last name. Choose a good variable name.
  - The 2 parameters must be invisible inside the Main() routine.
    - Tip: Make them private.
- The class Person has a constructor.
  - This must be the same name as the class.
  - 2 input parameters.
    - strFirstName (string).
    - strLastName (string).
- The class Person has 2 public properties.
  - To get and to set the first name.
  - To get and to set the last name.
- The class has one extra method "ShowFullName".
  - This shows the firstname and the lastname with a space in between.
  - Use the properties instead of the variables.

Notes

Exercise 11.04: A name with data checks on correctness

Restart from scratch.

You must learn the principle. You are not learning to "Copy Paste".

When you are confused on the exercise, take a decision or contact the PO.

- The namespace of your project is "LearnCSharp".
- You have a class "Program" that contains the Main().
  - This tests the class "Person".
    - Create 2 persons.
    - Show the full name of the 2 persons.
    - Test with spaces in front.
    - Test with spaces at the back.
    - Test with names that are not filled in.
    - Test with names that are too long.
- Create an extra class with the name "Person".
- The class Person has 2 variables.
  - One for the first name. Choose a good variable name.
  - One for the last name. Choose a good variable name.
  - The 2 parameters must be invisible inside the Main() routine.
    - Tip: Make them private.
- The class Person has 2 public properties.
  - To get and to set the first name.
  - To get and to set the last name.
  - Both properties has 2 corrections for the input.
    - Spaces in front and at the end are removed. Use the method "Trim".
    - The length must be longer than 0, but smaller than 25 for the first name and 50 for the last name.

Notes

- Create for trimming and checking the length a separate private method. Use it in both first and last name.
- When the first name is not filled in, you place “Unknown”.
- When the last name is not filled in, you place “Unknown”.
- When the first name is too long, you only put the first 25 characters in it.
- When the last name is too long, you only put the first 50 characters in it.

- The class Person has a constructor.
  - This must be the same name as the class.
  - 2 input parameters.
    - strFirstName (string), can be an empty string.
    - strLastName (string), can be an empty string.
  - Use the properties to force the correct input.
- The class has one extra method “ShowFullName”.
  - This shows the firstname and the lastname with a space in between.
  - Use the properties instead of the variables.

Is the maximum length for the first name and the last name easy to change?

Meaning, you can change it in one location / line.

Notes

.....

.....

.....

.....

.....

.....

.....

COPY PASTE)

Exercise 11.05: Me Tarzan, You Jane Doe

Make postits for all the actions that must be checked.

- Create a class for employees.
  - The name of the class must be “Employee”.
- Employees has 7 properties.
  - Every property can be changed using the object of that type “Employee” except “WorkingHoursPerWeek”.
  - An employee number (string).
  - A name (string).
    - The name can’t be empty.
    - If you make it empty the name “John Doe” (when male) or name “Jane Doe” (when female) is used.
  - A gender (boolean).
    - True = Female.
    - False = Male.
  - A start date (DateTime).
  - An end date (DateTime).
    - This date can be empty.
    - When not empty, it must be after of equal to the start date.
    - When it is before the start date, the end date and start date must be equal.
  - PartTime (Boolean).
    - True when PartTime.
    - False when FullTime.
  - WorkingHoursPerWeek.
    - When PartTime this must be 20.

Notes

        - When FullTime this must be 40.
- Create a constructor with an employee number, a name, a gender, a start date and if it is PartTime or not.
    - In the create of the object, the end date is always empty.
- Create a TestProgram (your Main()) to prove that you routine works always correct.
    - This means, all criteria must be matched whatever I try to do in the test program with the class “Employee”.

Exercise 11.06: An employee form

- Change the code of “00101-j Employee.zip”.
- Everywhere you show the full name, you must use a read only property that gives you that value.

Exercise 11.07: How old are you?

- Change the code of “00101-n PersonClass.zip”.
- Add a method that will show the age of the person on 31 December of the current year.
    - Call this method AgeThisYear.

	*Change both the Windows Form and the WPF form.*

Notes

.....

.....

.....

.....

.....

.....

.....

COPY PASTE)

Exercise 11.08: If I were a rich man, …

- A banking account does have a number and a balance.
- The number of a banking account is a number that has exactly 4 digits. Possible numbers are 1000 till 9999 (borders included).
  - When you try to input a wrong number, it will become automatic "1234".
- The balance can't become negative.
- You can add money to your balance.
- You can withdraw money from your balance.
  - When you try to withdraw more money than there is on the balance the withdrawing fails.
    - There is no transaction of a withdrawal. You show an error message.
- Create a method that shows the balance of the account.

You are completely free in how you solve this technically.

Create a little documentation on how this class should be used by any programmer who wants to use this in an application.

- *What are the public properties?*
- *How must the constructor(s) be used?*
- *How to add money.*
- *How to withdraw money?*

Variant

- Keep track of all transactions.
- Create a method that shows all the transactions.
  - From start (the creation) till the actual moment.

Notes

Exercise 11.09: Count me in (Part 01)

Below this paragraph you see a photo of a “counter”. This is an electronic device that can be used to count stuff.

- People entering a building.
- Counting the number of stitches in a crochet project.
- Counting the traffic.

Your task

- Create a class in a namespace that has the same functionality.
  - Namespace should be "CountingTool".
  - Class name should be "Counter"
- There are 2 values / properties:
  - Is it switched on?
  - What is the value?
    - On the photo it is switched on, with value 5.
- There is a method that increments the value.
  - The value becomes 1 higher, if the counter is switched on.
- There is a method that resets the value.
  - The value becomes 0, if the counter is switched on.
- There is a constructor.
  - The value becomes 0.
- The user of the class must be able to get the value.

Notes

COPY PASTE)

  - You show the value, if the counter is switched on.
  - When not switched on, nothing is shown.
- The user of the class must be able to switch on the counter.
  - Value becomes 0.
- The user of the class must be able to switch off the counter.
  - Value becomes nothing. (This is not 0).
- Write a test routine, that test the complete functionality.

Make sure that all needed properties, methods, constructors and classes are public.

You will use them in the next exercise.

Notes

.....

.....

.....

.....

.....

.....

.....

COPY PASTE

Exercise 11.10: Count me in too (Part 02)

In this exercise, you will use the result of exercise 11.09.

- Do not copy the code of the class.
- Create a dependency towards exercise 11.09.

Create a form application that has 2 labels and four buttons.

- Give all components of the form and the form itself a good name.
- The form you have created is a class.
  - Add a field of the data type "Counter".
  - Make sure you are using the namespace "CountingTool".
- The first label has the text "Next Customer", and will never change.
- The second label shows the number of the counter.
  - This can be nothing.
  - This can be a number.
- The first button has the label "Next".
  - Clicking on the button should increment the counter.
  - Make sure you can only click when the counter is on.
  - The value must be shown in the second label.
- The second button has the label "Reset".
  - Clicking on the button should put the counter back to zero.
  - Make sure you can only click when the counter is on.
  - The value must be shown in the second label.
- The third button has the label "On".
  - You switch on the counter.
    - You can't click on this button anymore till you have switched off the counter.
- The fourth button has the label "Off".
  - You can guess what must be done.

- Test what you will deliver to me.

Notes

Class Exercises in framework Karel the Robot

Solve them in Visual Studio.

Example 09999-e Karel the Robot.zip contains all demos given during the courses.

You can adapt, extend the existing code by solving those exercises. Create a new GitHub repository to save your progress. Make me contributor if it, when you want me to evaluate the exercises.

Exercise 11.11:

-

Notes