# Exercises about classes

Solve them in Visual Studio.

## Exercise 11.01

* 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.

## Exercise 11.02

* 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.

## Exercise 11.03

|  |  |
| --- | --- |
|  |  |
|  | 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.

## Exercise 11.04

|  |  |
| --- | --- |
|  |  |
|  | 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.
    - 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. |
|  |  |

## Exercise 11.05

|  |  |
| --- | --- |
|  |  |
|  | 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.
    - 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

* 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

* 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. |
|  |  |

## Exercise 11.08

* Een bankrekening nummer heeft een nummer en een saldo.
* Het bankrekening nummer is een getal dat exact uit 4 cijfers bestaat tussen 1000 en 9999 (grenzen inbegrepen).
  + Indien je een fout rekeningnummer probeert in te geven, dan wordt het automatisch 1234.
* Het saldo kan niet negatief zijn.
* Je kunt geld storten op de rekening.
* Je kunt geld afhalen van de rekening.
  + Indien je meer geld probeert af te halen dan dat er opstaat, dan gaat de transactie niet door, door een foutmelding te tonen.
* Maak een methode die toont hoeveel geld er op de rekening staat.

|  |  |
| --- | --- |
|  |  |
|  | Je bent volledig vrij in doen en laten van hoe je dit technisch oplost.  Maak een document met handleiding aan hoe deze klasse kan gebruikt worden.   * Welke properties zijn er? * Hoe werkt de constructor? * Hoe zet er er extra geld op? * Hoe haal je geld eraf? |
|  |  |

### Variant

* Hou een lijst bij van alle transacties die je hebt uitgevoerd.
* Maak een methode die alle transacties toont.