**Warning – this practice exercise is bigger than a usual exercise, and gives you a chance to practice the reflection, XML, LINQ/EF materials. The materials not required here, but described in the presentation can also be part of the midterm-tests!**

In our program, we want to handle instances of possibly multiple classes. Those classes can have various properties with various validation rules, for example:

-- The property value cannot be empty string (for string properties)

-- The property must not be null (any reference-types)

For the properties, we have to use attributes to mark the applied validation rules.

For example, the following code means that the name must not be null or empty:

class Person

{

[NotNull]

[NotEmpty]

public string Name { get; set; }

}

* Create the appropriate attributes.
* Create the Person class that is capable of storing data from the following XML file: http://users.nik.uni-obuda.hu/hp/people.xml. Instead of the room number as a string, let’s store the floor number only, as an int.
* Apply meaningful validation rules to the properties of the Person class.
* Create the ValidationMethods class, where static methods with the same name (e.g.: NotNull) realise the validation rules. Every rule has the property value as a parameter, plus additional meaningful parameters for the given rule. Every rule has a bool as the output: true, if the value is OK, and false if the value is bad.
* Create the Validator class, that has a single Validate method that receives an object instance as a parameter. This method uses reflection to check which property requires which validation rules, and then checks the rules by calling the appropriate methods of the ValidationMethods class.
* Read the XML contents, create the Person objects with the new data, and then write out those people, where a validation error occurs.
* Add a Service-Based Database into the Solution, and create a table to store people:

CREATE TABLE people (

personId int primary key,

personName varchar(200),

personPhone varchar(200),

personFloor int

);

* Modify the earlier created Person class: create an attribute (*DbPropertyNameAttribute*), that helps us to map the properties to names that exist in the database, thus to names in the appropriate entity object. For example:

[DbPropertyName(”PersonId”)]

public int ID { get; set;}

* Create a converter class, and a converter method:

public static void ConvertProperties(object source, object destination)

This method must copy the values of the properties marked with DbPropertyName attributes from the source instance into the destination instance’s appropriate properties.

* Using the converter method, copy those XML data that have no validation errors into the database!