Project 2

T-213-VEFF, Vefforritun, 2015-1

Classes, constructors and overriding ToString()

Your task is to create the classes School and Student, such that the following Main( ) function should compile and execute correctly:

```
namespace SchoolApp
{
    public class Program
    {
        public static void Main( string[] args )
        {
            School sch = new School( "HR" );
            sch.AddStudent( new Student { Name="Nonni", Age=22 } );
            sch.AddStudent( new Student { Name="Sigga", Age=23 } );
            Console.WriteLine( sch );
        }
    }
}
```

Also, the output when this program is executed should look like this:

```
School: HR
Number of students: 2
    Nonni, 22
    Sigga, 23
```

Implementation hints:

- Create the classes School and Student. You will need to write a constructor for School that accepts a single string which is the name of the school.

- Create properties for the class School. The class School needs only one property which stores the name of the school (of type string). It also needs a class member variable which stores a collection of students. The best way to do that is to use a class similar to C++'s std::vector, called List<>. It can be found in the namespace System.Collections.Generic, so you will need to add a using statement at the top of the School.cs file. Then, you should initialize this variable in the class:

  ```
  private List<Student> m_students = new List<Student>( );
  ```

- Add a function to the class School called AddStudent. It should accept a single parameter of type Student. It should simply store the student instance in the List<Student> member variable (use the Add( ) function which works similarly to the push_back( ) function in std::vector).

- Add properties to the class Student. You will need two properties: Name and Age. I'll leave it to you to figure out the type of these properties.

- Finally, override the ToString functions in both classes (see the book for details). The Student version of ToString is quite simple, while the School implementation will need the following:
    - A local string variable that will contain the end result. You can then use the += operator to concatenate to this variable, and use System.Environment.NewLine; to add a newline to the string. Example:

      ```
      string result = "Some string" + System.Environment.NewLine;
      result += "some other string";
      ```

    - A loop that walks through all students in the m_students collection (also, see the book about foreach statements), and adds each student to the string. Note: you can either call the ToString method for each student explicitly, or use a syntax similar to the one seen inside Main above, where the School instance is printed out explicitly.

Multiplication table

Write a single program called MultiplicationTable, that accepts as input a single integer number in the range 1 - 20, and prints out (using Console.WriteLine or Console.Write) a HTML document which displays the multiplication table for the input number.
Example of a command line:

```
MultiplicationTable.exe 5 > Result.html
```

should result in a file called Result.html being created in the same folder as the .exe file (note: the > syntax will take care of creating the file, all you need to do is to print out the results using Console.WriteLine as stated above). This file should be a valid XHTML file, with a single table in the body of the document:

Multiplication table for number 5	
1	5
2	10
3	15
4	20
5	25
6	30
7	35
8	40
9	45
10	50

Note: you can add the command-line parameter to the debug session in Visual Studio by selecting Project - Properties, select the "Debug" tab, and add the number to "Command line arguments" input field. However, using the " > Results.html" syntax is a bit harder, you will have to run the program yourself from the command line to be able to use the > operator on the command line.

Inside Main, you can use args[0] to access the command line argument, and then use Convert.ToInt32( ) to convert it to an integer.

Hand in

The deadline for this assignment is February 18th on midnight. Hand in a single .zip/.rar/.7z file containing the .cs files plus the .exe files, or more specifically:

- School.cs
- Student.cs
- SchoolApp.exe
- MultiplicationTable.cs
- MultiplicationTable.exe

Please note that this is an individual assignment.