Method assignments

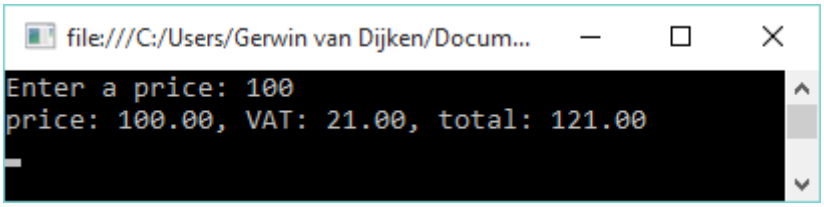
When creating the program code, you must apply the following basic principles:

- create a separate project for each assignment;
- use name ‘assignment1’, ‘assignment2’, etcetera for the projects;
- create one solution for each week containing the projects for that week;
- make sure the output of your programs is the same as the given screenshots;

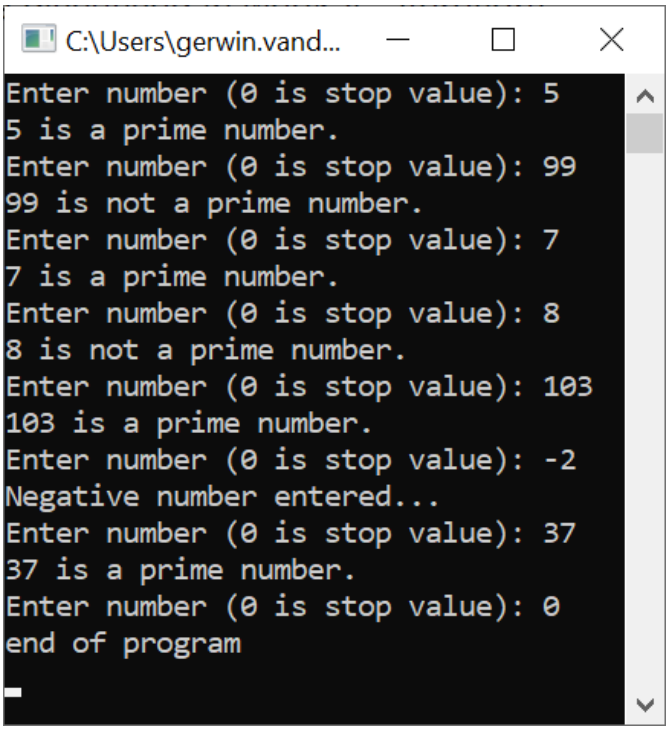
Note: for assignment 1, your output must contain a dot (.) as a decimal separator, and not a comma (,), see screenshots of this assignment. To make sure your program uses a dot, add the following code to your program (2 using-statements and the code at the start of your Main-method):

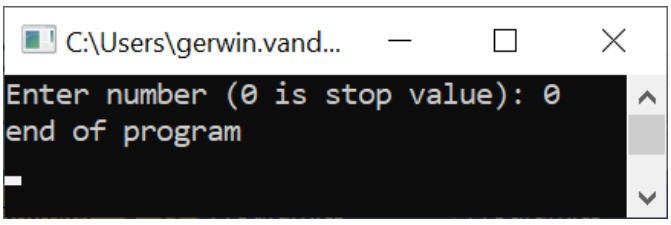
```
using System;
using System.Globalization;
using System.Threading;

static void Main(string[] args)
{
    // set culture of program
    CultureInfo ci = new CultureInfo("en-US");
    Thread.CurrentThread.CurrentUICulture = ci;
    Thread.CurrentThread.CurrentCulture = ci;

    // your code here...
}
```

Assignment 1 (Console App)
Enter a price. Use a separate method to determine the 21% VAT on this price; this method accepts a float parameter and returns a float value (‘return value type’ is float).
Print the price, VAT, and the price + VAT.

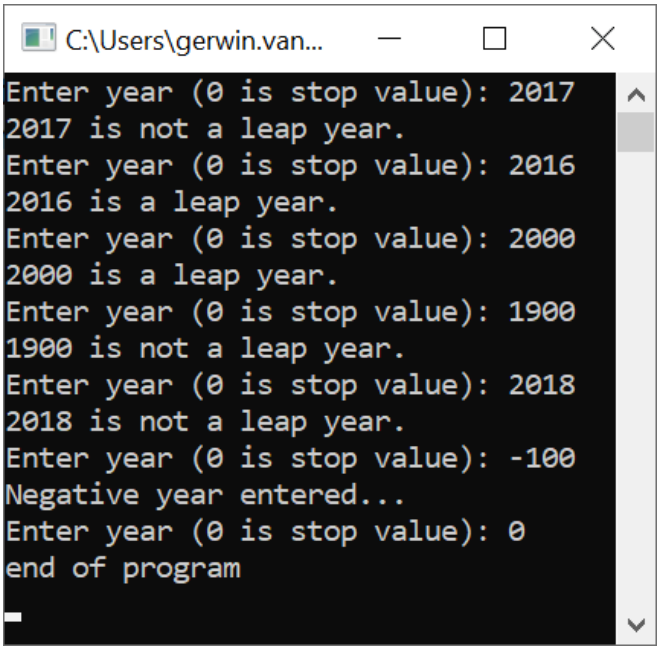


Assignment 2 (Console App)
Several numbers are entered until number 0 is stated. Use a separate method 'IsPrimeNumber' to determine whether the positive numbers (>0) are prime numbers; this method accepts an int parameter and returns a boolean value. *(prime numbers were discussed in week 4 - Iteration)*

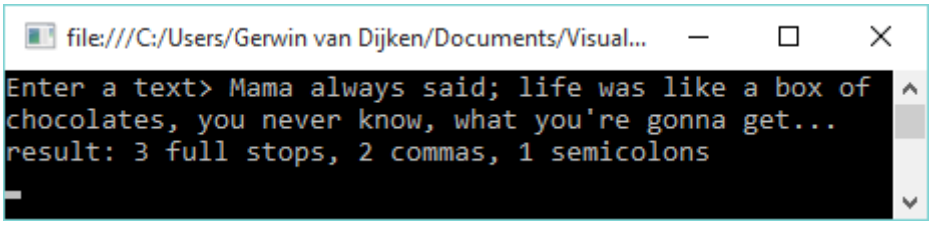




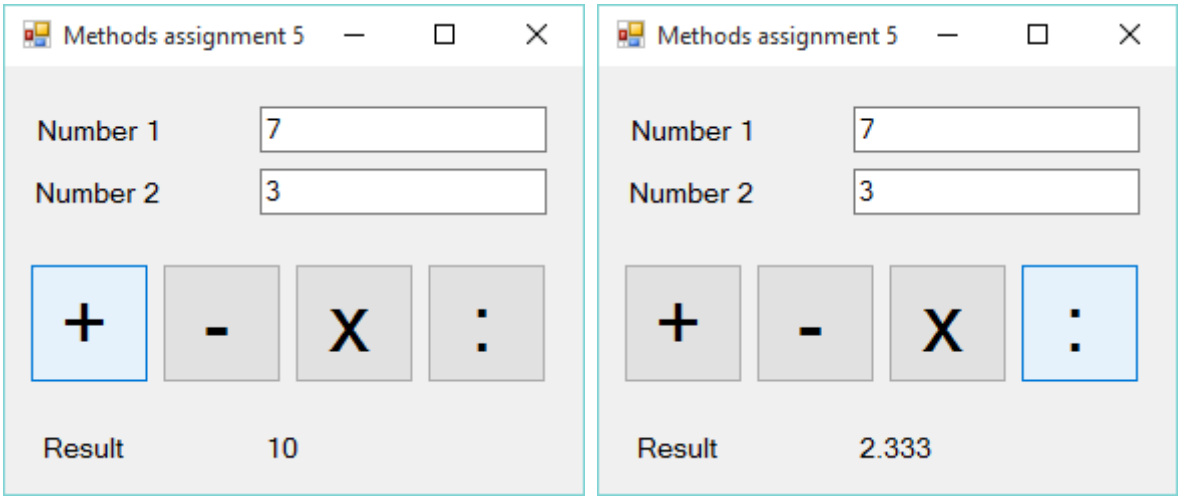
Assignment 3 (Console App)
Several years are entered until number 0 is stated. Use a separate method 'IsLeapYear' to determine whether each year is a leap year; this method accepts an int parameter and returns a boolean value.



Assignment 4 (Console App)
Enter a text (string). Determine the number of times that the characters . (full stop) , (comma) and ; (semi-colon) occur in this string. Use a separate method 'SearchText' for this purpose, with a string as an input parameter, and with 3 out parameters: nrOfFullStops, nrOfCommas, nrOfSemiColons. This method has no return value.

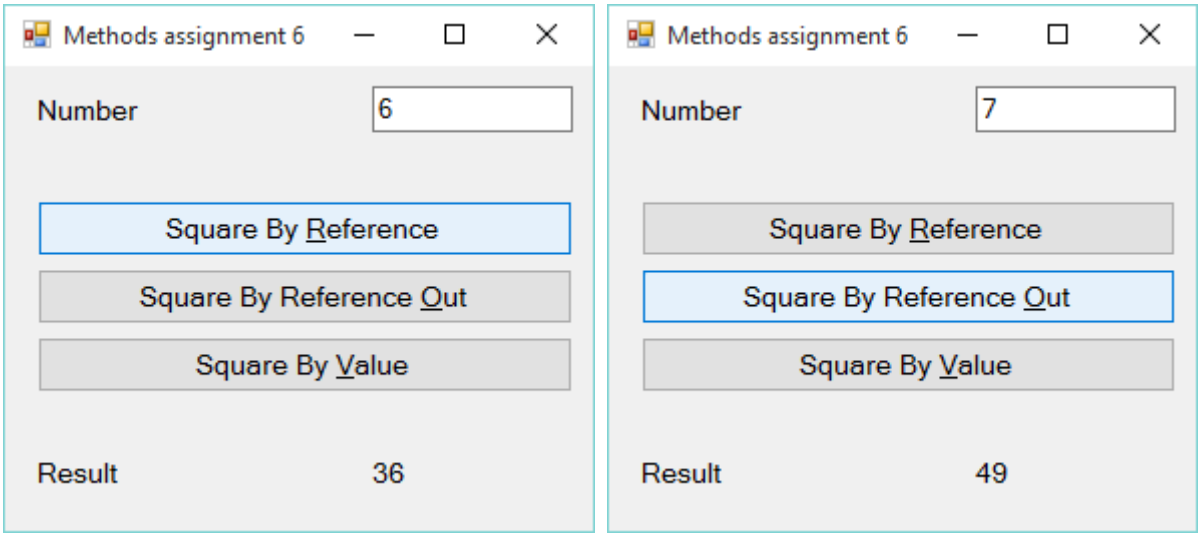


Assignment 5 (Windows App)
Depending on the user's choice, the numbers number1 and number2 are added up, subtracted, multiplied or divided. Write a method for each calculation, clearly name the method to reflect its task. Pass to each method the (int) value of number1 and number2 (as parameters). After the calculation, each method will return an int (or double) as the calculation result.



Assignment 6 (Windows Application)
This application is calculating (and displaying) the square of a number entered. Three methods can be used to calculate squares. The first method is based on the principle of 'call by reference', the second on 'call by reference out' and the third on 'call by value'. When the user presses a command button, the corresponding method is being called. Use the following /signatures for the three methods:

```
void SquareByReference(ref int number)
void SquareByReferenceOut(int number, out int square),
int SquareByValue(int number)
```



Assignment 7 (Windows App)
A Windows Forms application must be designed and implemented to convert temperatures. If, for example, a temperature is entered in degrees Celsius, it can be converted to Kelvin or degrees Fahrenheit or vice versa by selecting a radiobutton and clicking the Calculate button.

Implement a method for each conversion. These 3 methods must be called with an argument, i.e. the temperature that should be converted. Each method returns the converted temperature value. Use the following method names: Celsius2Kelvin(c), Celsius2Fahrenheit(c) and Fahrenheit2Celsius(f).

Conversion formulas (C: Celsius, K: Kelvin and F: Fahrenheit):

- c °C → (c + 273) K
- c °C → (c × 9/5 + 32) °F
- f °F → ((f - 32) × 5/9) °C