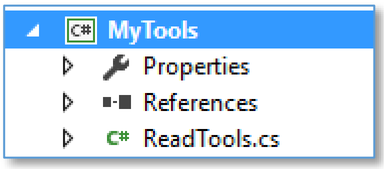
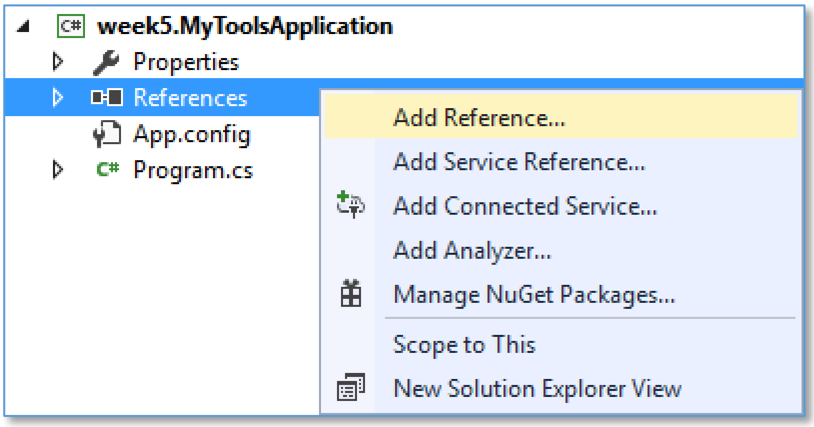
Assignment 1 – A library with some useful methods

We will refactor assigment 0 of the 1st lesson by introducing a Class Library. Later we can use this Class Library in other projects.



a) Create a new project of type 'Class Library' and give it the name ‘MyTools’.
In this project create a new class `ReadTools`.
Now copy the 2 `ReadInt` methods and the `ReadString` method of assignment 0 of the 1st lesson to this class.
Make sure these methods are `public` and `static`.

b) Create a new Project of type 'Console Application' and give it the name ‘MyTools Application’.
Copy the Main method of assignment 0 of the 1st week to this new Console Application.
→ Add a reference in the Console Application to the Class Library ‘MyTools’.



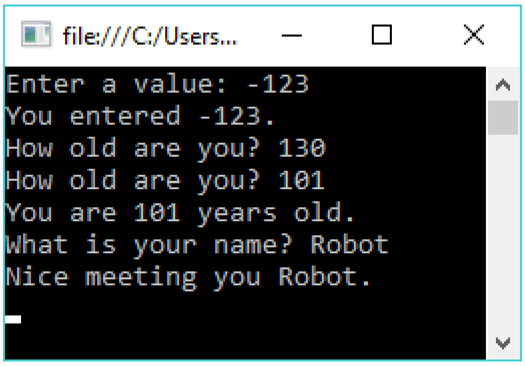
→ Add at the top of your Program the following line:

```
using MyTools;
```

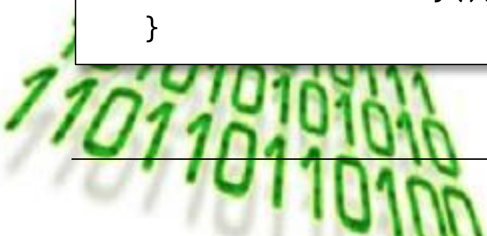
→ Adjust the method calls (in method Start) in order to use the `ReadTools`-methods of Class Library `MyTools` (see code below).
→ Test your programma.

You can add multiple classes to this MyTools Library (for all kinds of useful methods).

```
static void Main()
{
    Program myProgram = new Program();
    myProgram.Start();
}

void Start()
{
    int value = ReadTools.ReadInt("Enter a value: "
    Console.WriteLine("You entered {0}.", value);

    int age = ReadTools.ReadInt("How old are you? ", 0, 120);
    Console.WriteLine("You are {0} years old.", age);

    string name = ReadTools.ReadString("What is your name? ");
    Console.WriteLine("Nice meeting you {0}.", name);

    Console.ReadKey();
}
```



```
Enter a value: -123
You entered -123.
How old are you? 130
How old are you? 101
You are 101 years old.
What is your name? Robot
Nice meeting you Robot.
_
```



Assignment 2 – CandyCrush with logic layer

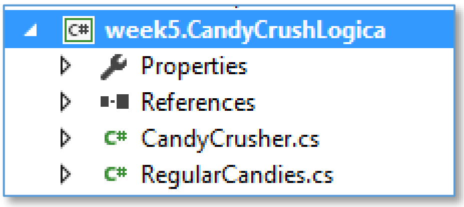
We can distinguish userinteraction and game-logic in the CandyCrushGame. So it makes sence to create a separate layer for the CandyCrush-logic.

a) Create a new CandyCrush-application (Console project) and copy the code of the CandyCrush-application of week 4 into this new application.
→ Check if this new application is working.

b) Add (to your solution) a new ‘Class Library’ project with the name CandyCrushLogic.
Add a `class CandyCrusher` to this Class Library.
Move the 2 ‘Present’ methods to this CandyCrusher class:

- ScoreRowPresent(…)
- ScoreColumnPresent(…)

Also move the file with `enum RegularCandies` to the logic layer (change the namespace, this has to be the same as the namespace of class CandyCrusher).
→ build this Class Library, there should be no compile errors.


week5.CandyCrushLogica
Properties
References
CandyCrusher.cs
RegularCandies.cs

c) Now the CandyCrush application (created in a) has to use the logic layer:

- In the CandyCrush application add a reference to the logic layer.
- In Program.cs add a `using` statement for the namespace of the logic layer.

→ Make sure to use (call) the ‘Present’ methods defined in the logic layer, and test your application again.

Assignment 3 – Translation

In this assignment we will read a file with Dutch words and the corresponding English translation. With this program the user can enter a Dutch word and the program will display the English translation. You can use the file "dictionary.csv" that can be found on Moodle.

a) Create a new Console project with the name 'Translation' and add the following method:

```
Dictionary<string, string> ReadWords(string filename)
```

This method reads all words from a textfile and stores these words in a Dictionary. Each line contains a Dutch word and the English translation, separated with a semicolon (e.g.: "slecht;bad"). Add each line in the dictionary with the Dutch word as key, and the English translation as the value. You can split each read line via `line.Split(';')`, in order to get an array with (in this case 2) separated fields.
Return the dictionary after all lines have been processed.
→ Call method ReadWords from the Start method.

b) Add the following method:

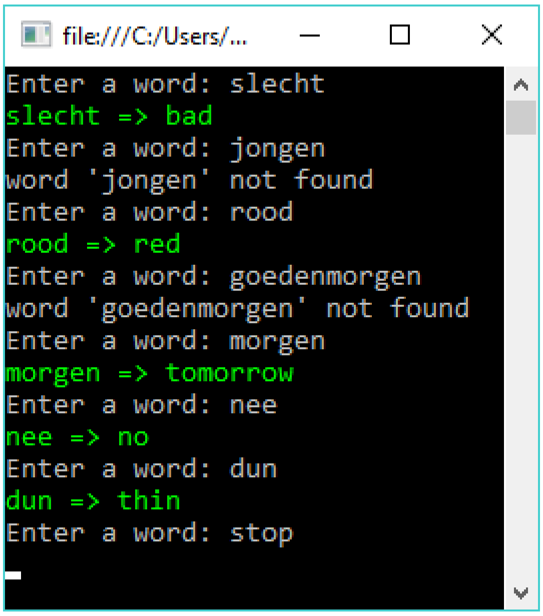
```
void TranslateWords(Dictionary<string, string> words)
```

This method reads continously a Dutch word from the user until the word 'stop' has been entered. For all other words the translation is displayed via the words-dictionary. If the dictionary does not contain the (Dutch) word, then a message is displayed (e.g. "word not found"), otherwise the (English) translation is displayed. A few examples can be seen in the screenshot below, to the left *(the use of different colors is optional)*.
→ call method TranslateWords from the Start method.
→ Test different words to test the program.

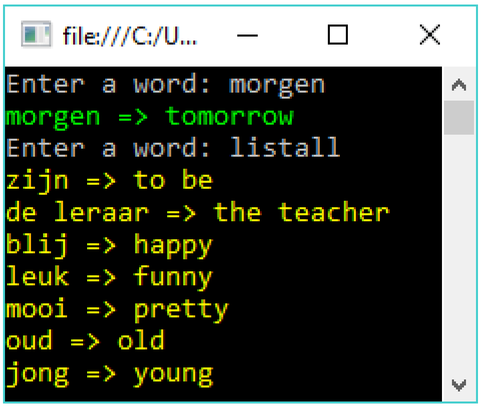
c) Add the following method:

```
void ListAllWords(Dictionary<string, string> words)
```

This method displays all Dutch words in the dictionary, together with the corresponding translations. Adjust method TranslateWords in order to display all words (by calling ListAllWords) when the user enters "listall". An example can be seen below, to the right *(again, the use of different colors is optional)*.



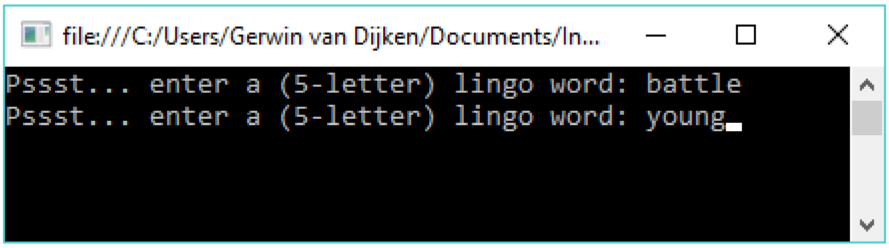
```
Enter a word: slecht
slecht => bad
Enter a word: jongen
word 'jongen' not found
Enter a word: rood
rood => red
Enter a word: goedenmorgen
word 'goedenmorgen' not found
Enter a word: morgen
morgen => tomorrow
Enter a word: nee
nee => no
Enter a word: dun
dun => thin
Enter a word: stop
```



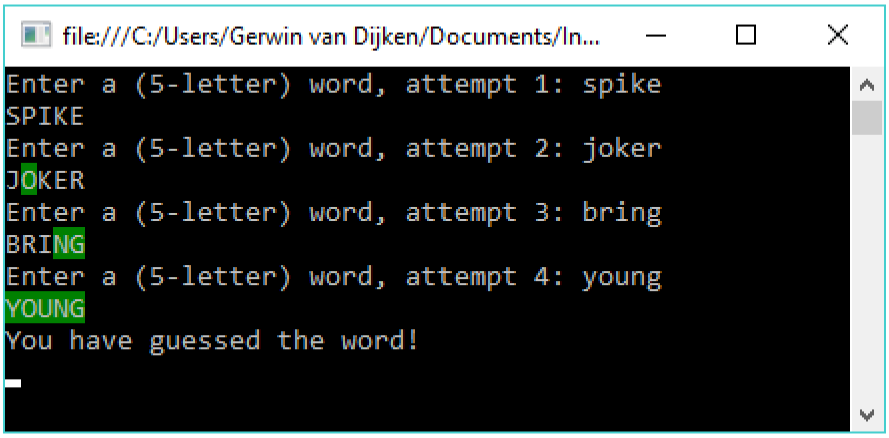
```
Enter a word: morgen
morgen => tomorrow
Enter a word: listall
zijn => to be
de leraar => the teacher
blij => happy
leuk => funny
mooi => pretty
oud => old
jong => young
```

Assignment 4 - Lingo

In the Programming 2 theory class the analysis of the Lingo game has been discussed: the main task and the subtasks of the game. Create this application by implementing the mentioned tasks (methods). Of course you can use the pseudocode from the slides.



```
Pssst... enter a (5-letter) lingo word: battle
Pssst... enter a (5-letter) lingo word: young
```



```
Enter a (5-letter) word, attempt 1: spike
SPIKE
Enter a (5-letter) word, attempt 2: joker
JOKER
Enter a (5-letter) word, attempt 3: bring
BRING
Enter a (5-letter) word, attempt 4: young
YOUNG
You have guessed the word!
```