

# CSCI 1301 – Lab 02

August 21, 2019

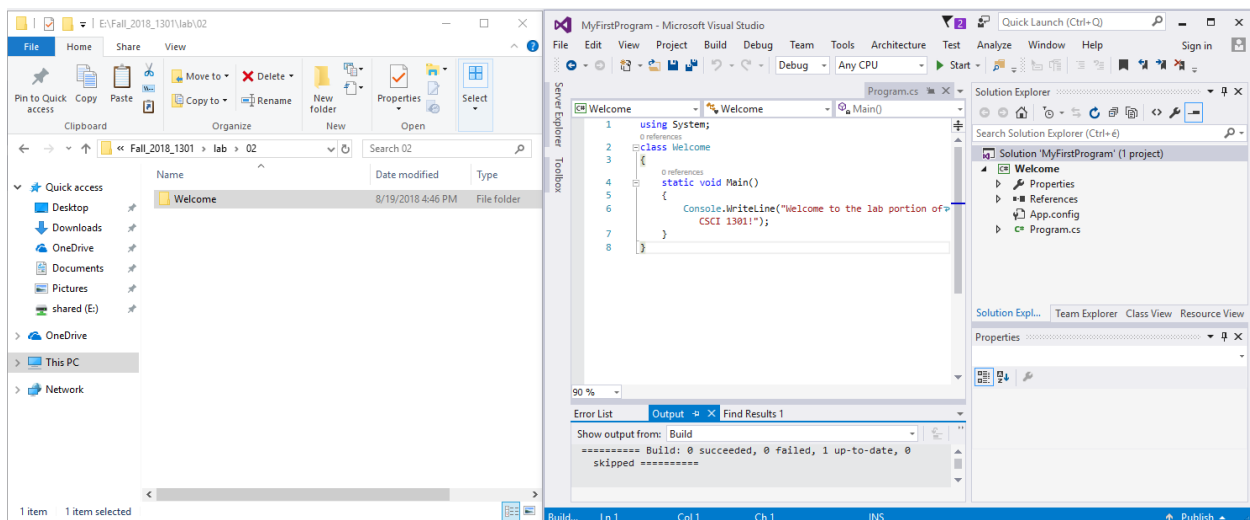
## 1 How Was the Backup?

Re-download or transfer the files you saved last time (the whole `Welcome` folder) on the computer and make sure you can still open the project with Visual Studio (VS). Do you remember how to build the solution and start the program without debugging? Using the shortcuts? If not, go have another look at the previous lab<sup>1</sup>.

If your backup went wrong (you can't open the project, it won't compile, ...), try to understand what happened. Then, re-download the “Welcome” archive<sup>2</sup>, extract it and make sure you can build the solution and start the program without debugging.

## 2 Orientation

Have a look at the following screenshot:



You should have roughly the same on your computer. If not, you can zoom to have a better reading.

Answer the following:

1. Where is the file explorer?
2. How many folders are in the folder opened in the file explorer?
3. How many files are in the folder opened in the file explorer?
4. Where is Visual Studio (VS)?
5. Where is the solution explorer?
6. Where is the window where I can edit the source code?
7. What is the output of the building?

---

<sup>1</sup> ../01/

<sup>2</sup> ../01/Welcome.zip

### 3 Re-using and Editing

Close Visual Studio (VS). With the file explorer, create a new folder (call it for instance 02\_lab) and copy the Welcome folder in it. In this exercise, you will rename and edit this copy.

#### 3.1 Renaming

Open with VS the copy of the Welcome folder that you just created. Rename the solution **within** VS:

1. Right-click on “Solution ‘Welcome’ (1 project)”, select “Rename”, rename it to “MyFirstProgram”.
2. What change(s) do you notice in Visual Studio?
3. Can you still build and debug your program?
4. Look in the file explorer: did the name of the folder changed? Did the name of the .sln file changed?

You should *not* rename a solution with the file explorer: *always* use VS to rename.

#### 3.2 Editing

We will now change (edit) our “MyFirstProgram” solution.

1. In program.cs, replace "Welcome to the lab portion of CSCI 1301!" with "This is my first program.".
2. Build the solution and start the program without debugging. Do you notice any change?
3. Insert a new line after `// This is an in-line comment.` and before the first `}` sign, and paste the following:

```
Console.WriteLine("This is my second message.");
```

4. Build the solution and start the program without debugging. Do you notice any change?
5. Insert another new line after the one you just created, and paste the following:

```
Console.WriteLine("This is my third message.");
```

6. Build the solution and start the program without debugging. Can you notice the difference between `WriteLine` and `Write`?
7. Insert another new line after the one you just created, and paste the following:

```
Console.WriteLine("\t This is my fourth message.");
```

8. Build the solution and start the program without debugging. Can you tell what `\t` is doing?
9. Insert another new line after the one you just created, and paste the following:

```
Console.WriteLine("\n This \n is \n my fifth message.\n");
```

10. Build the solution and start the program without debugging. Can you tell what `\n` is doing?
11. Have a look the documentation page of microsoft on escape sequences<sup>3</sup>, and edit your program by adding a statement that displays the `\` and the `"` characters.

<sup>3</sup><https://docs.microsoft.com/en-us/cpp/c-language/escape-sequences>

12. Add a comment (using `//` or `/*` and `*/`) in your program.

Make a back up of what you just did: upload the `02_lab` folder you created previously on your remote backup, or copy it on your thumb drive. **Do not** use the “Save as...” capacities of VS: close VS and copy / upload the folder “by hand”, using the file explorer or a browser. Re-downloading or re-transferring it and re-opening it is a good way of making sure that your back up was correct.

## 4 Breaking Your Program

If you followed the instructions carefully, you were able to build the solution and start the program without debugging after each step. As you know, `C#` has precise rules, and not respecting them can prevent your solution from being built by VS.

In this exercise, you are asked to do the following:

1. Change the program so that it violates one of the syntax rules of `C#` (refer to your lecture notes, and to the suggestions below).
2. Build the solution, and note that an error is reported. In the build output (cf. the screenshot presented earlier, you may need to click on a tab in VS to see it), you will see a message like

```
===== Build: 0 succeeded, 1 failed, 0 up-to-date, 0 skipped =====
```

3. Click on the “Error list” tab next to the build output, and make sure you understand the error message.
4. Undo your change, using `Ctrl + z`.
5. Make sure you can build the solution without error message anew.

Do it three times, in order to identify three different error messages, and three ways of breaking the rules.

If you have time or lack inspiration, you can try with the following, and see which one(s) make the building impossible (don’t forget to undo your change after):

- Remove the semicolon after using `System`
- Replace `class Welcome` with `class TestOne`
- Remove the brace (or “curly bracket”, i.e., the `}` symbol) at the last line.
- Add three new lines at the end of the file
- Replace `Console.WriteLine` with `CONSOLE.WriteLine`
- Replace `Console.WriteLine` with `Console.WRITELINE`
- Add a new line between `Console.` and `WriteLine`
- Add a new line between `WriteLine` and `(`
- Add a new line between `Write` and `Line`
- Replace `Main()` with `Method()`
- Remove the indentation (i.e., the space between the beginning of the line and the first character of the instruction) on all lines