

**Swinburne University of Technology***School of Science, Computing and Engineering Technologies***LABORATORY COVER SHEET**

---

<b>Subject Code:</b>	COS30008
<b>Subject Title:</b>	Data Structures and Patterns
<b>Lab number and title:</b>	1, Using Visual C++
<b>Lecturer:</b>	Dr. Markus Lumpe

---



**Figure 1: Visual Studio 2022 Startup Window.**

## Where to get Visual Studio?

Every student enrolled in a unit of the Faculty of Science, Engineering and Technology can obtain a valid copy of Visual Studio via the faculty's ELMS Campus Portal:

<https://elms.swin.edu.au/>

The screenshot shows a web browser window displaying the ELMS Portal at <https://elms.swin.edu.au/>. The page has a dark header with navigation links: STUDY WITH US, RESEARCH, BUSINESS & PARTNERSHIPS, NEWS, and EVENTS. Below the header is a red Swinburne University of Technology logo and a dark bar with the text "IT Teaching Resources". The main content area is titled "ELMS Portal" and includes a description: "The Electronic License Management System (ELMS) portal provides an interface for eligible STEM students to obtain license codes and download media for select VMware and Microsoft development tools." The login section features fields for "Username" (Your SIMS ID) and "Password" (Your SIMS password), with a red "Sign-in" button. To the right, under "Available Campus:", it states that access is allocated based on enrolment in eligible subjects. Below this, "Azure Dev Tools for Teaching" is listed with three bullet points: Faculty of Science, Engineering and Technology; Centre for Business, Design and ICT; and Department of Information Systems and Logistics. "VMware" is also listed with two bullet points: Centre for Business, Design and ICT; and Faculty of Science, Engineering and Technology. A "More information" section at the bottom provides links for "Forgotten your password?", "What is VMAP?", and "Swinburne network access policies". The footer contains copyright information, provider codes, and various links like "Contact us", "Jobs at Swinburne", "Copyright and disclaimer", "Privacy", "Welcome to Country and Acknowledgement", "Accessibility", "Feedback", and "Index".

STUDY WITH US RESEARCH BUSINESS & PARTNERSHIPS NEWS EVENTS

IT Teaching Resources

FeeNIX > ELMS Portal

### ELMS Portal

The Electronic License Management System (ELMS) portal provides an interface for eligible [STEM](#) students to obtain license codes and download media for select VMware and Microsoft development tools.

**Username**  
Your SIMS ID

**Password**  
Your SIMS password

[Sign-in >](#)

**Available Campus:**  
For member faculties, access is allocated based on enrolment in eligible subjects.

**Azure Dev Tools for Teaching**

- Faculty of Science, Engineering and Technology
- Centre for Business, Design and ICT
- Department of Information Systems and Logistics

**VMware**

- Centre for Business, Design and ICT
- Faculty of Science, Engineering and Technology

**More information**

**Forgotten your password? >**  
Reset your password online. For more help contact the IT Service Desk on (03) 9214 5000 or email [servicedesk@swin.edu.au](mailto:servicedesk@swin.edu.au).

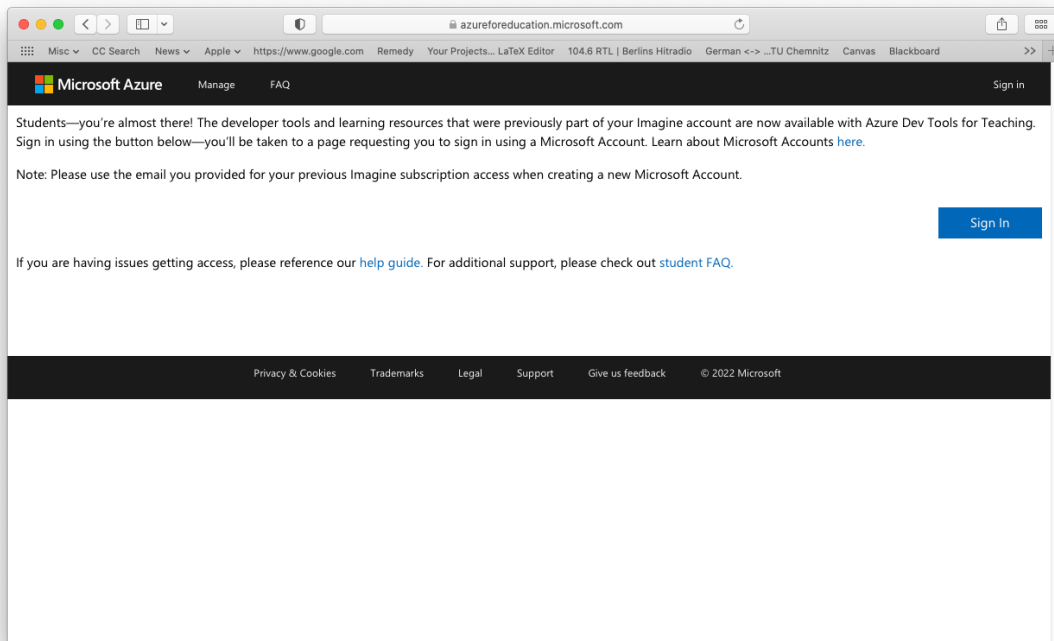
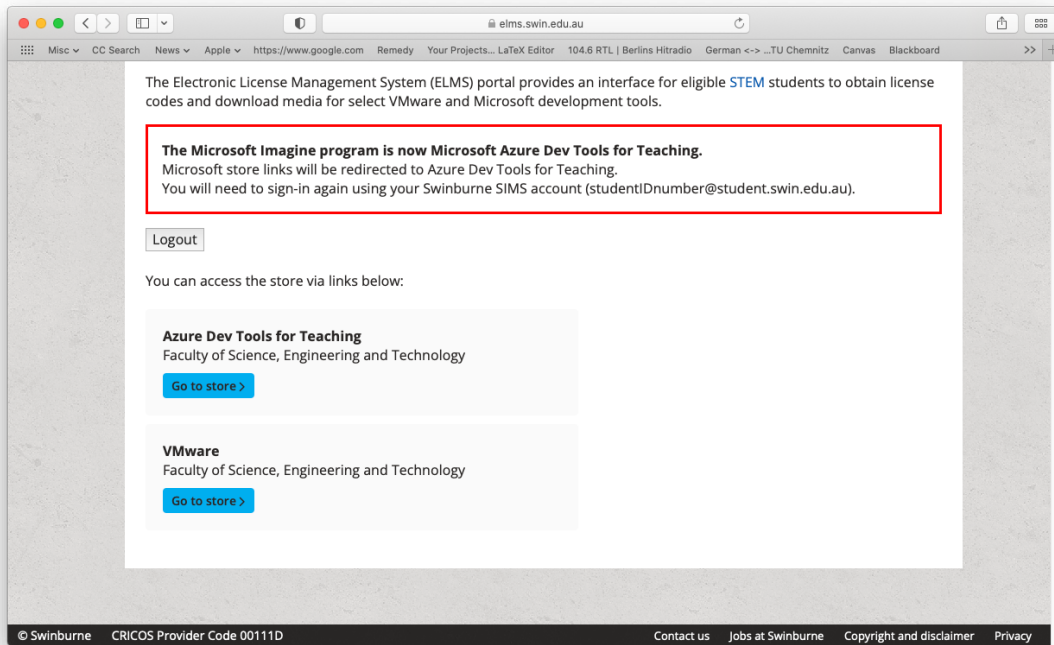
**What is VMAP? >**  
The VMware Academic Program (VMAP) is a comprehensive program designed specifically for the academic community. Click [here](#) for more information.

**Swinburne network access policies >**  
You are required to comply with Swinburne network access policies and the terms of use of licensed content providers.

© Swinburne CRICOS Provider Code 00111D  
TOD Provider Code 3059

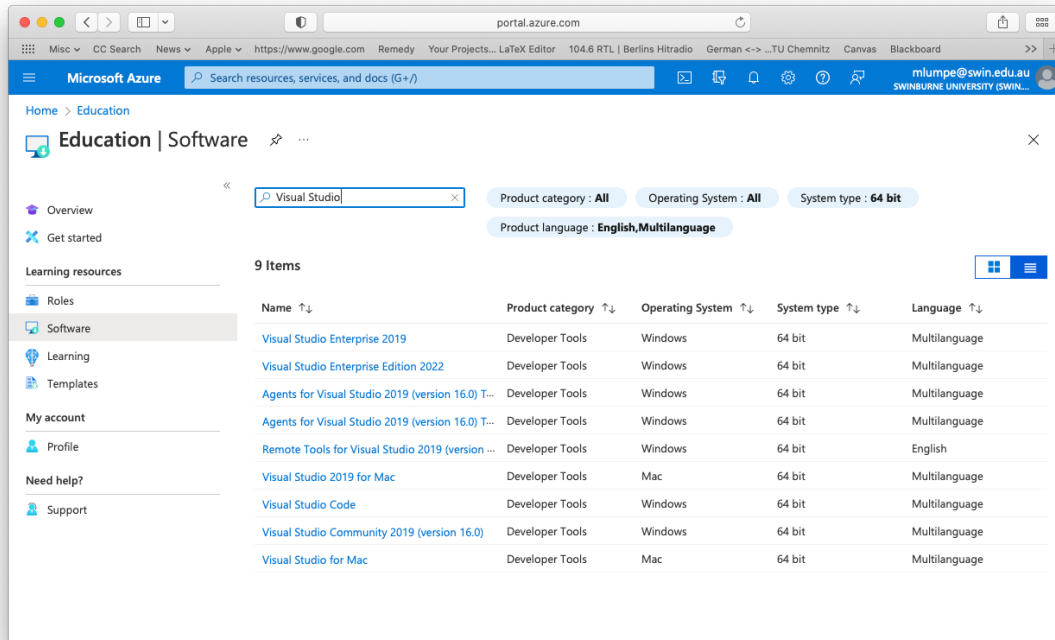
Contact us Jobs at Swinburne Copyright and disclaimer Privacy  
Welcome to Country and Acknowledgement Accessibility Feedback Index

Upon login, you land on the Dashboard. Continue with **Microsoft Imagine** and sign in.



You will be asked to do multi-factor identification. Once you logged in, select the Software tab and search for Visual Studio.

Select Download Software and search for Visual Studio (Enterprise is preferred, if available):



Click on the desired version and view the key and download (if you have not done so previously).

Any edition works for COS30008. The Enterprise Edition provides you with a complete set of tools. However, we will only rely on the basic elements in the unit.

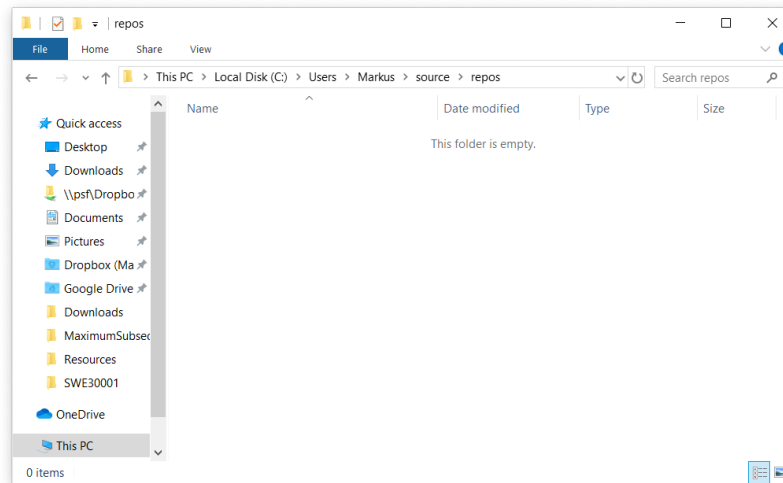
Follow the installation instructions. Make sure, you have at least C++ selected. In addition, you may need to select a Windows 10 SDK. Browse through all options to make an informed decision. You can always restart the installer, if you wish to add more components later.

The installation may take hours. Do not try this in the tutorial.

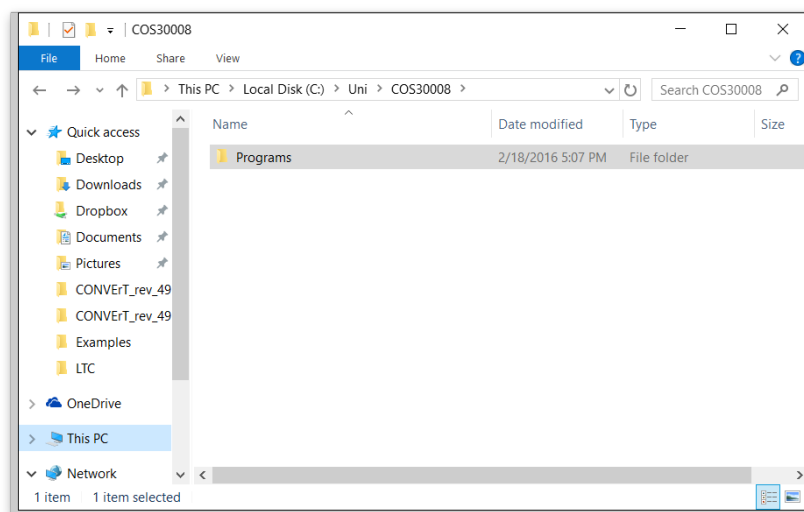
## Lab 1: First Steps in C++ - Visual Studio

### First Step: Select a working directory.

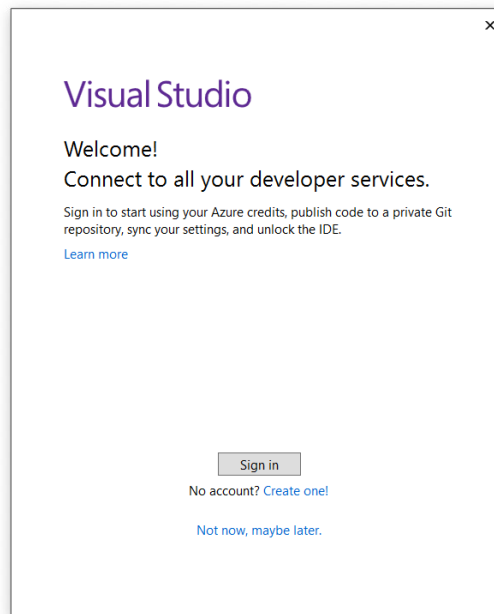
In order to develop a program, you need a [working directory](#). In Visual Studio 2022, this working directory is called [repos](#) and it is located in the current user's folder [source\repos](#):



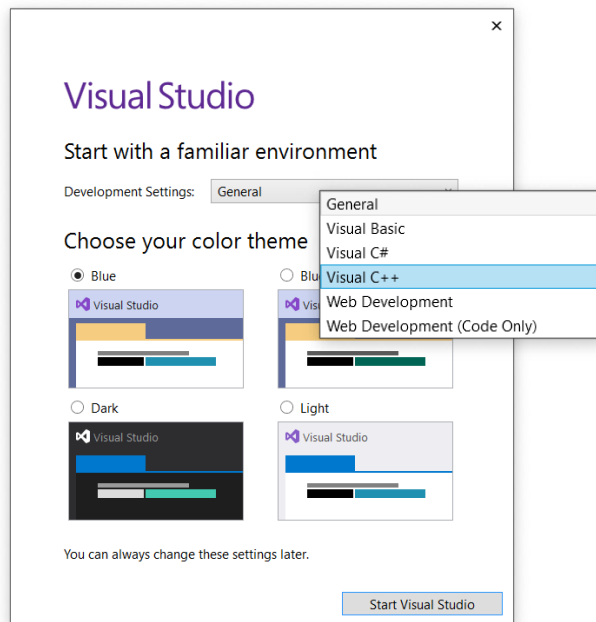
You can use this location, but in a shared environment like the university lab computers there is no guarantee that the contents of this folder will be retained between sessions. Lab machines get routinely reimaged. Also, you may want to separate your projects based on course work. For example, you may want to assemble all your files (i.e., lecture notes, assignments, and projects) in a semester-specific subject directory. For example, you could use [C:\Uni\COS30008\Programs](#) (or another suitable location, if you do not have write permissions for C:) as your working directory (and throughout this tutorial):



## First Start of Visual Studio. You Do Not Need To Sign In!



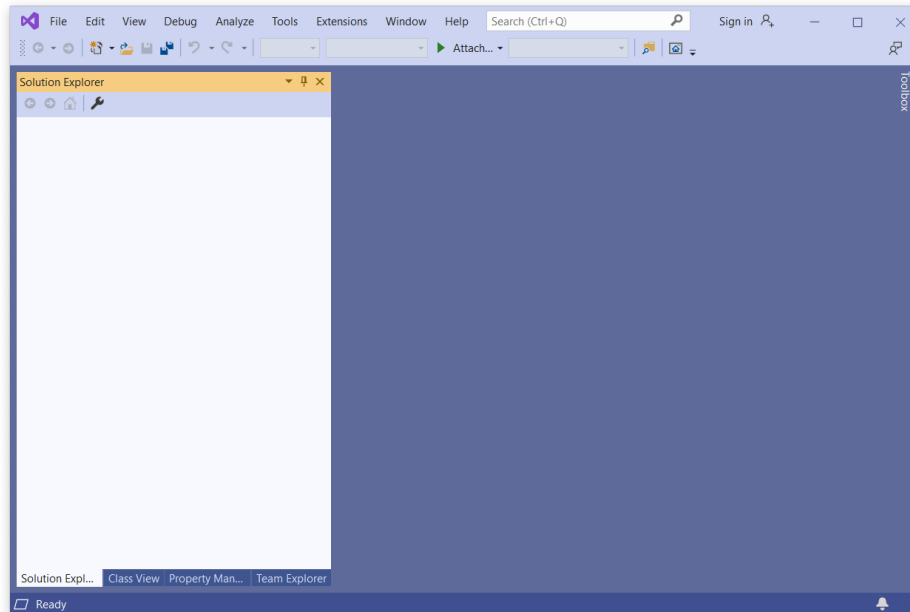
Just select "Not now, maybe later." Next select your preferred theme:



Dark may be good for your eyes. In this document, we use Blue. You can also select the development settings: Visual C++. Then Visual Studio will start. It may take a while.

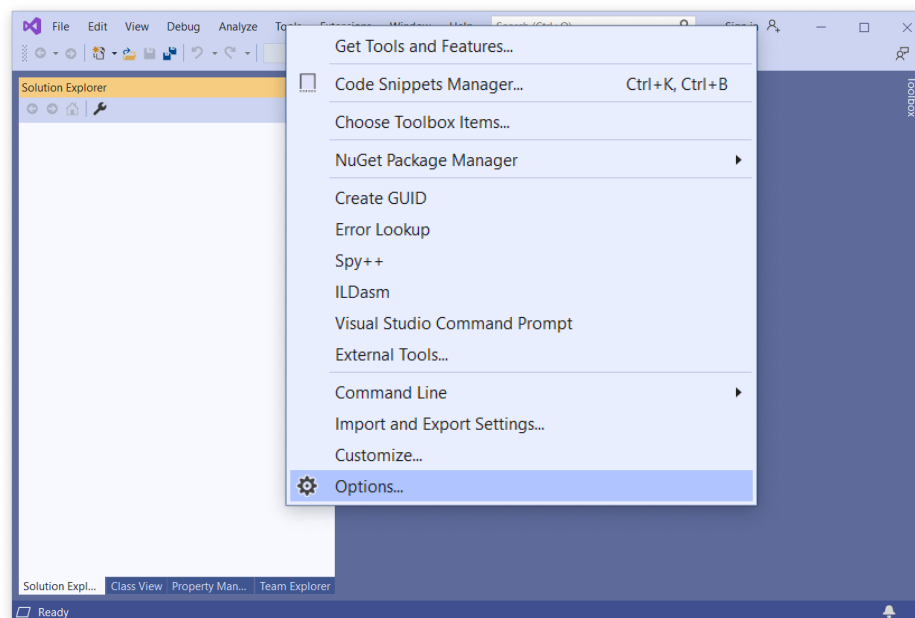
## Second Step: Start Visual Studio and Select C++ Template.

Select Visual Studio 2015 from the Apps Menu and wait for the following screen to appear:

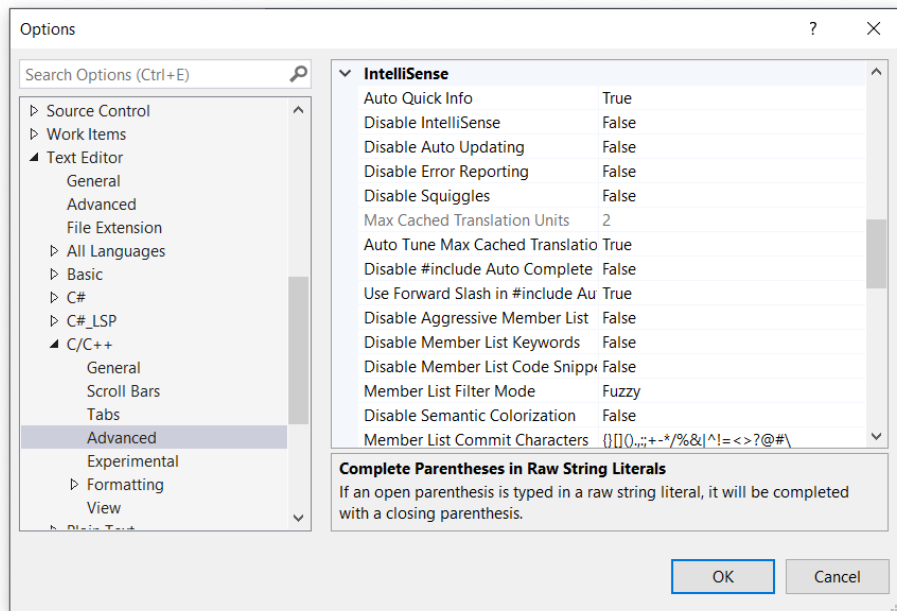


Before we start, let us disable IntelliSense. Even though it is a useful feature in professional software development, when learning or improving our knowledge of C/C++ it is rather a distraction. Moreover, you will not have access to IntelliSense (and code completion) in tests and exams.

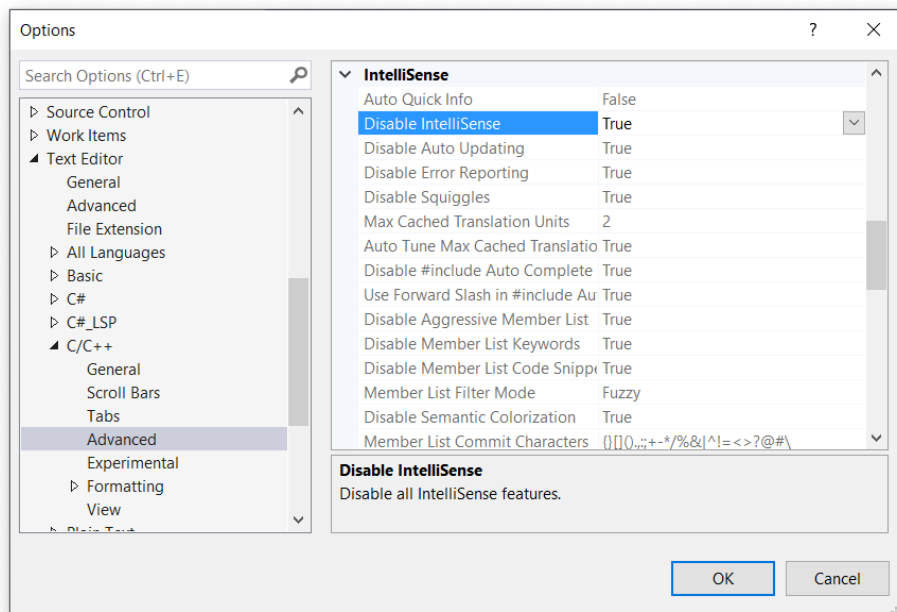
Select [Tools/Options...](#)



...Text Editor/C/C++/Advanced and scroll in the right window to IntelliSense.



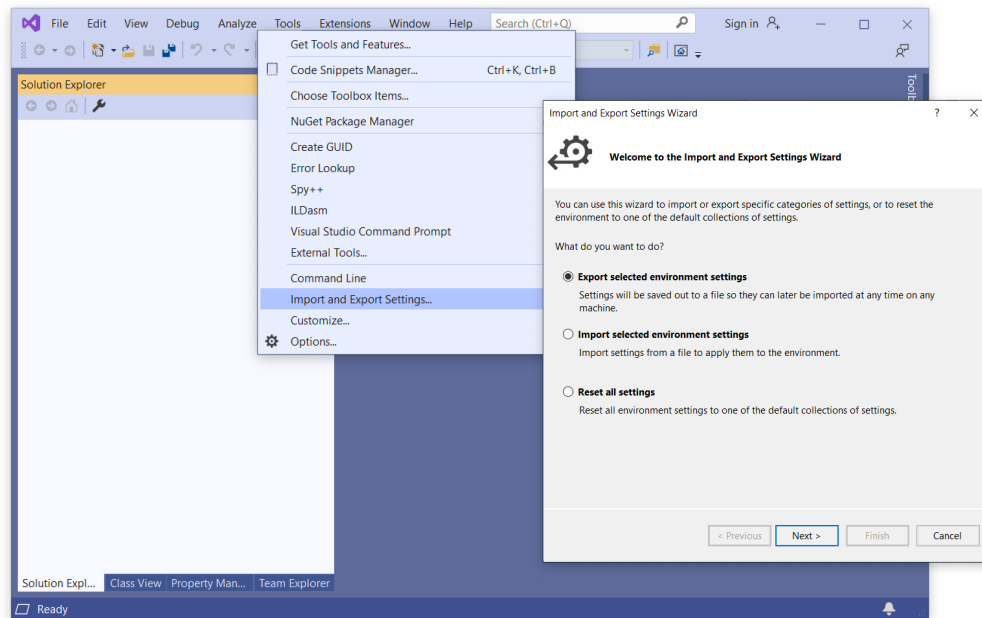
Change **Disable IntelliSense** to True and press **OK**.



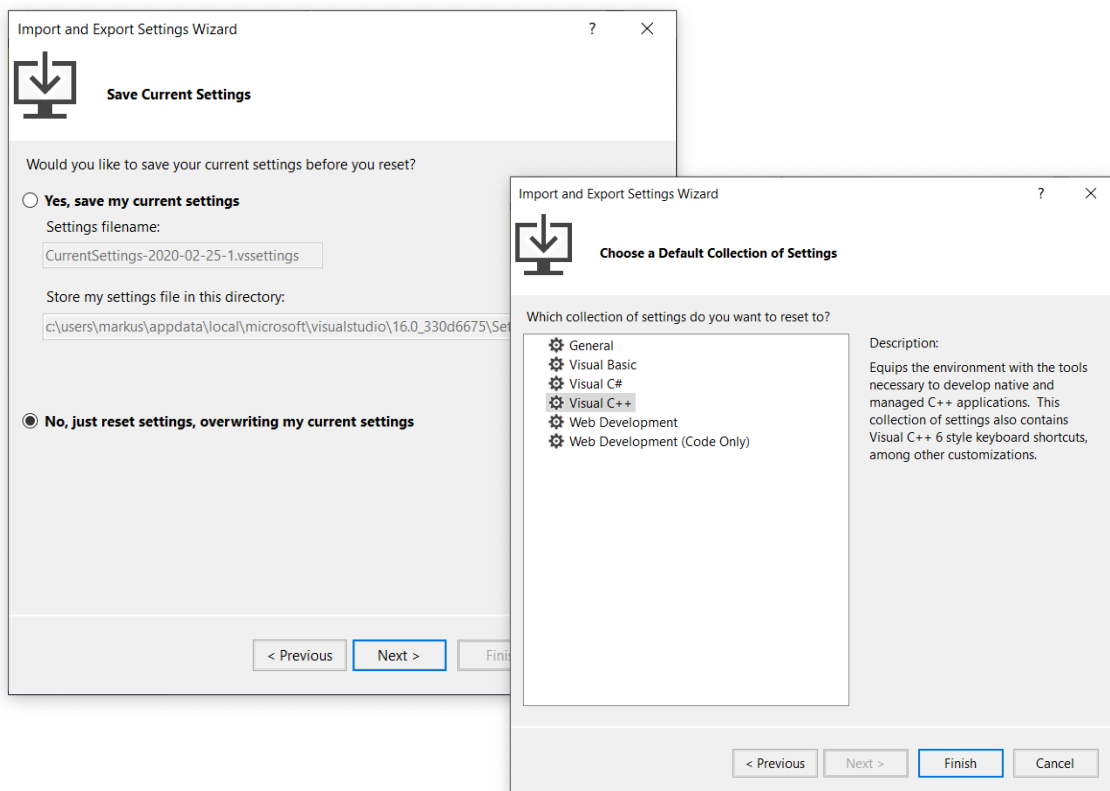
This is not permanent. However, disabling IntelliSense will help you to learn and focus on the programming tasks.



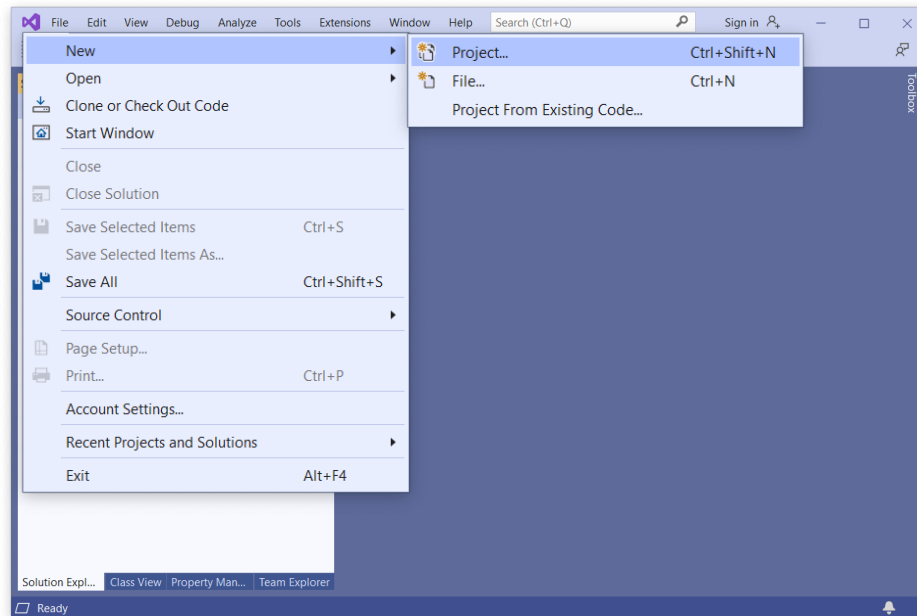
We also may want to set environment to C/C++. We can achieve this by selecting Tools/Import and Export Settings... and reset to current environment to C/C++. This option also allows you to save settings, if necessary.



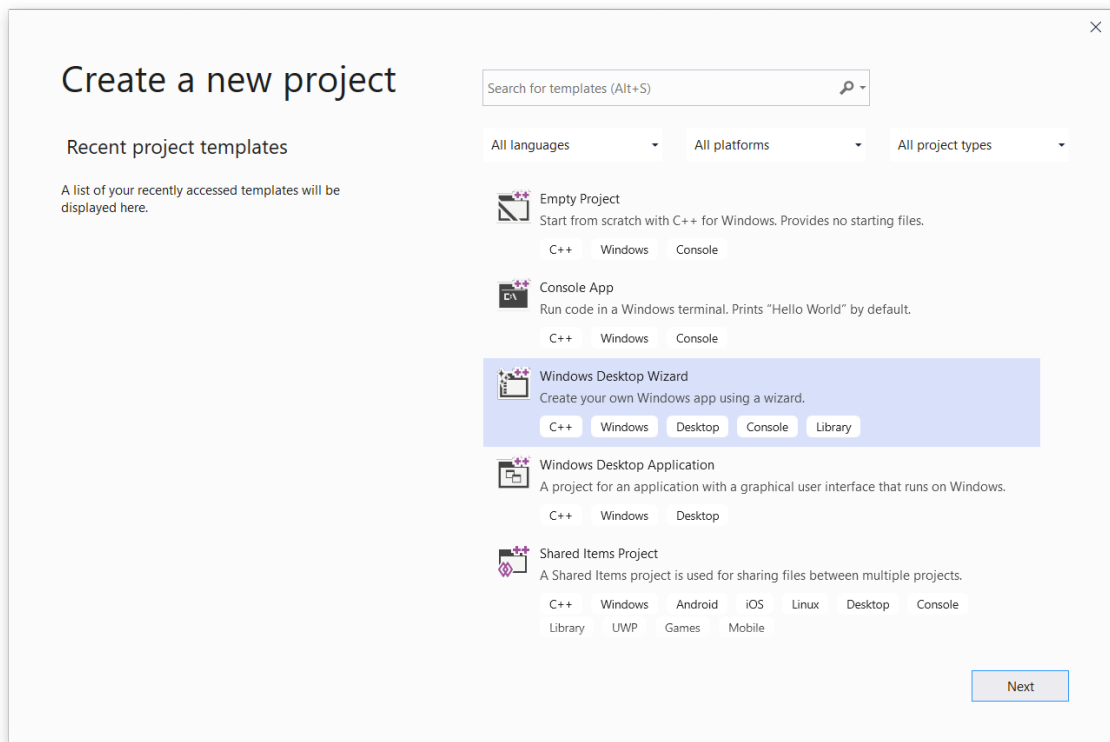
We use [Reset all settings](#) and choose as default [Visual C++](#). Completing this step guarantees that new projects will be automatically populated with C++ templates.



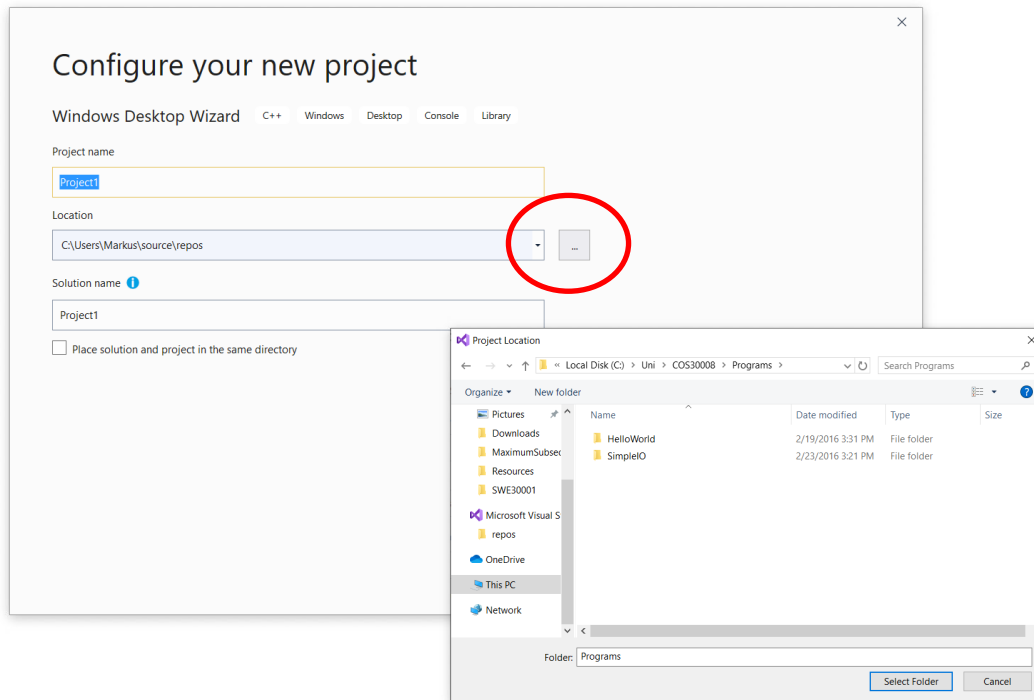
Select **New Project...** now:



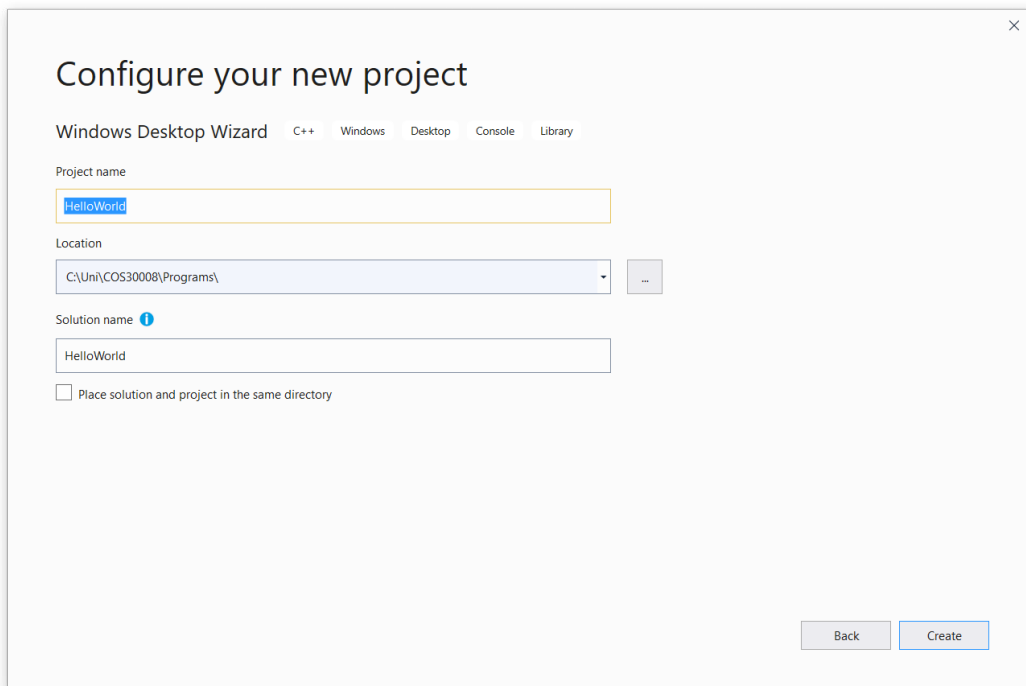
As a result, the New Project dialog window will appear. This is a so-called modal dialog window and it will stay on top of Visual Studio to force you to choose a proper project template. That is, you cannot do anything else before you have selected the type of project that you want to work on and you have completed the associated project-specific configuration steps. When working on projects in COS30008, choose **Windows Desktop Wizard**. It allows you to configure everything.



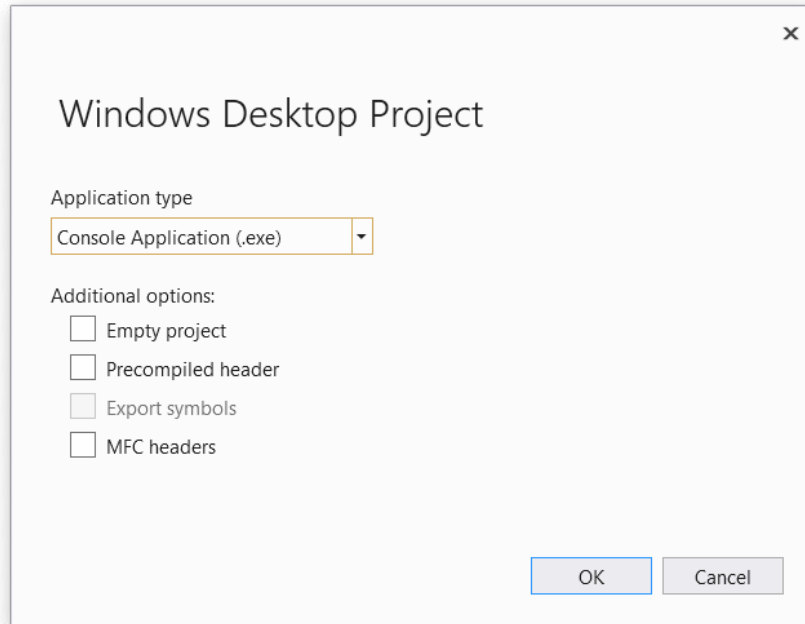
You need to set the project location. That is click on the [Browse...](#) button, which will bring up the [Project Location](#) dialog. Select the desired folder:



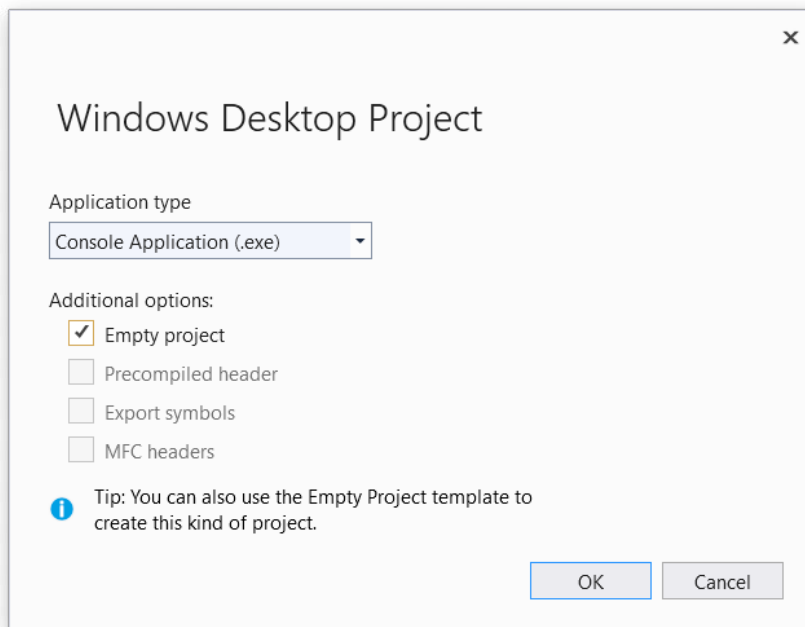
Then select a proper project name. Here we use [HelloWorld](#).



Create the project. As you used the Windows Desktop Wizard, another dialog appears that allows you to make the final adjustments to your project. This step is crucial.

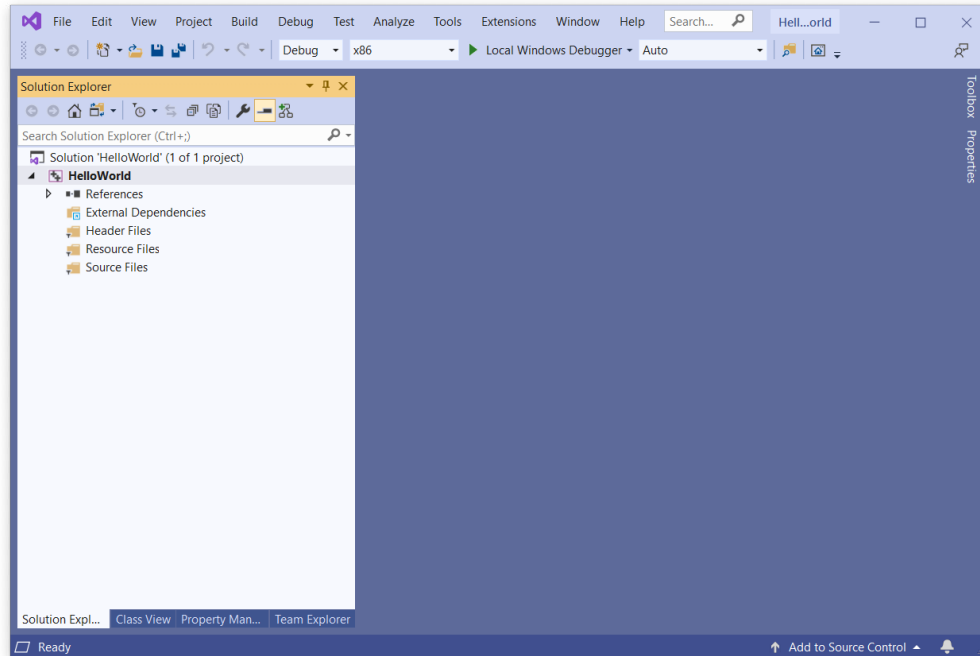


Make sure that **Console Application (.exe)** is the application type. Most importantly, select **Empty project**. This guarantees that Visual Studio does not create any extra project artifacts. You want to be in full control and only add things to your project that you need.



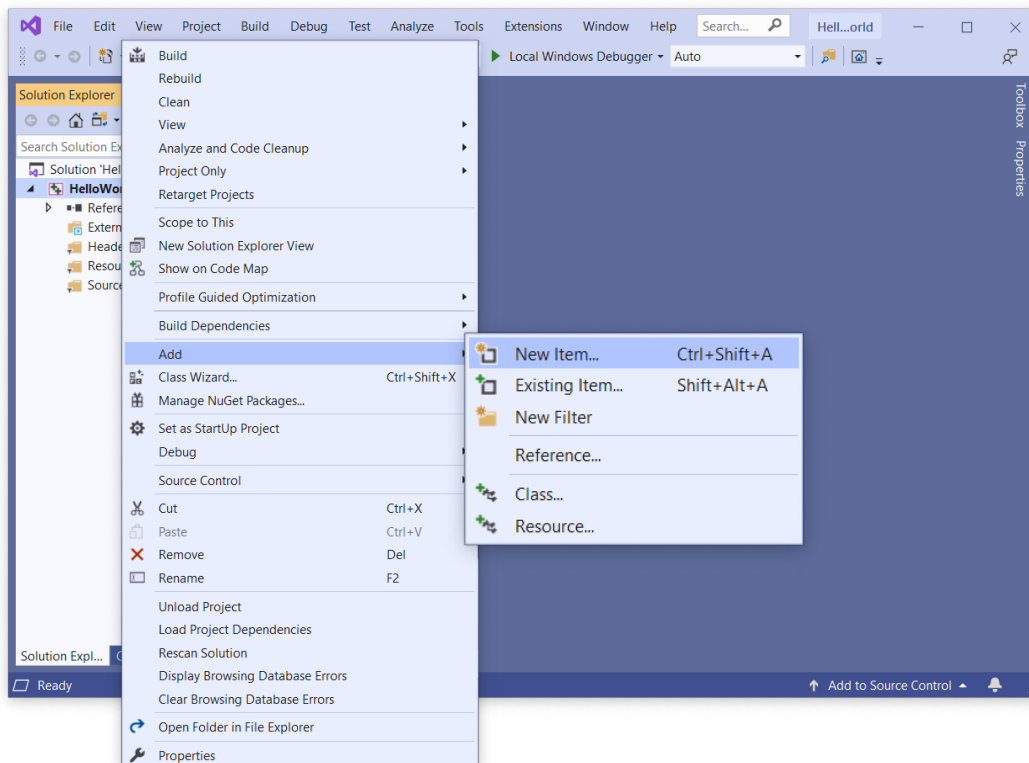
Ignore the tip. It is somewhat misleading. You are not really creating an empty project, you are creating a properly set up project with no source code yet.

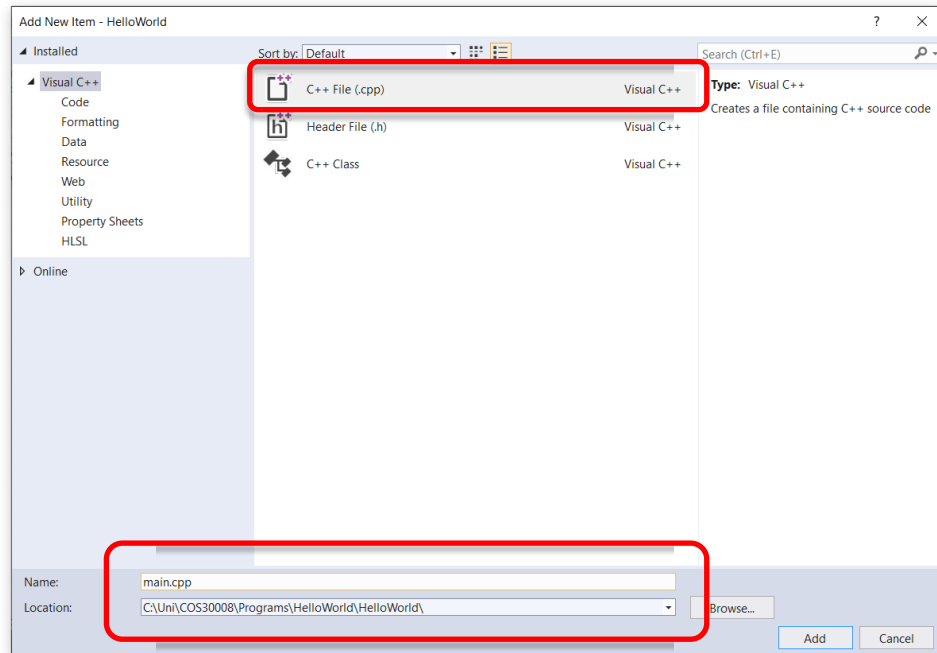
Once you select OK, the project should be loaded into Visual Studio. It is indeed empty.



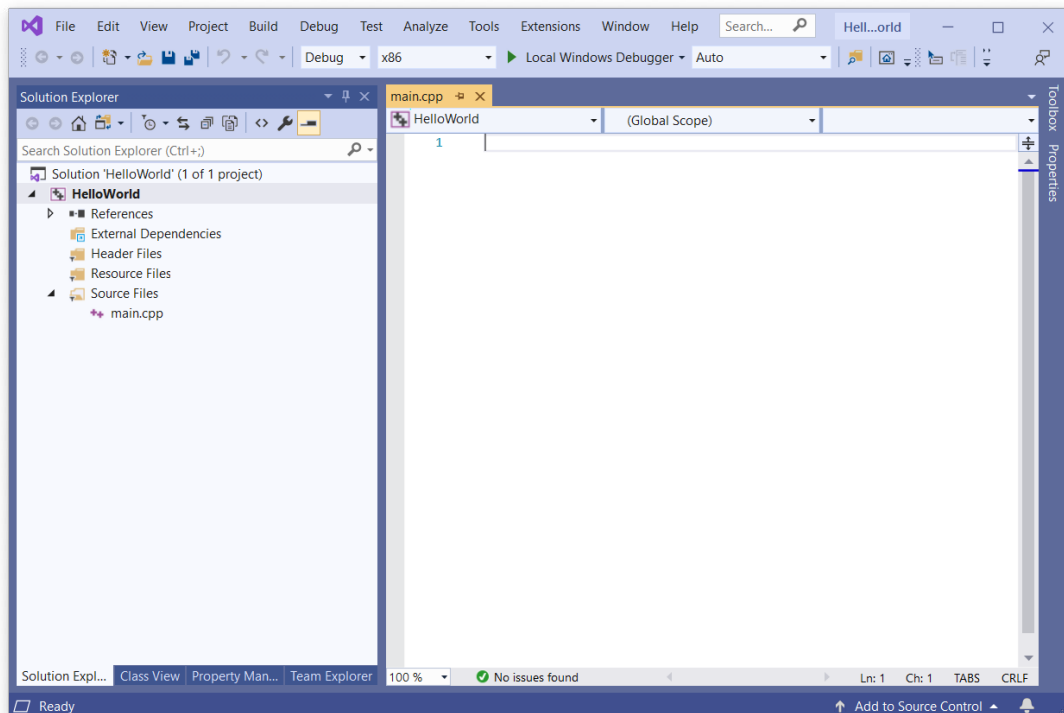
Visual Studio uses a docking system. That is, you can rearrange the location of windows any way you see fit. Try it.

Select the project name and right click. Now we add a **New Item**. We select **Code** of **Visual C++**, select **C++ File (.cpp)**, and type **main.cpp** as the name for the file:



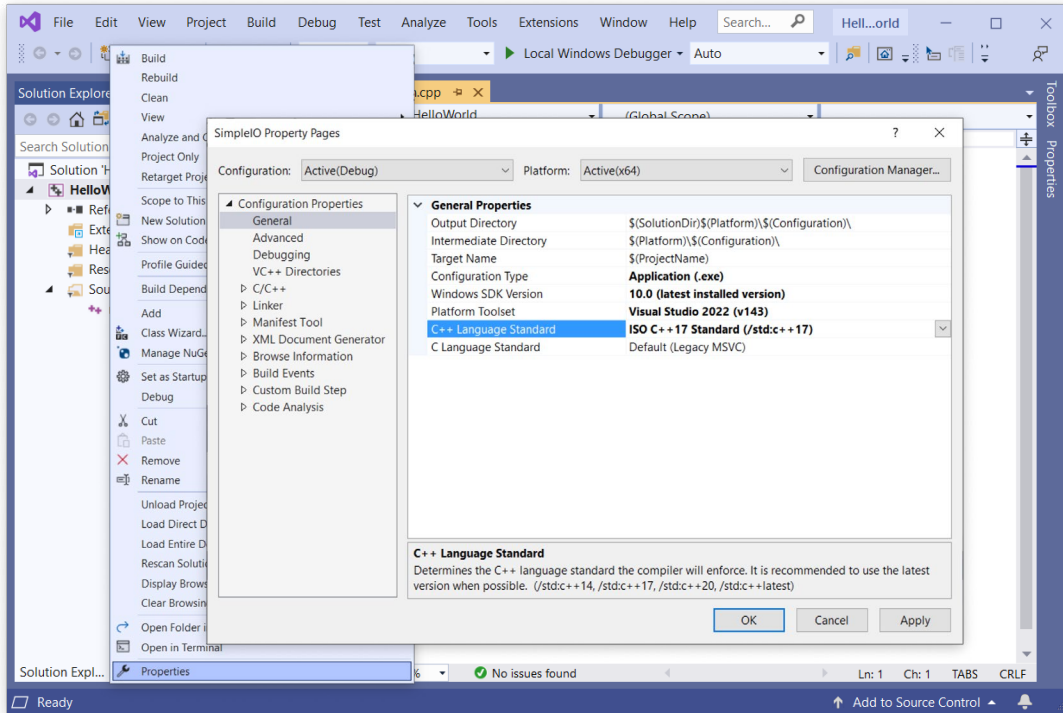


If we press the **Add** button, then an empty **main.cpp** file will be inserted into our project and Visual Studio will start the built-in code editor.

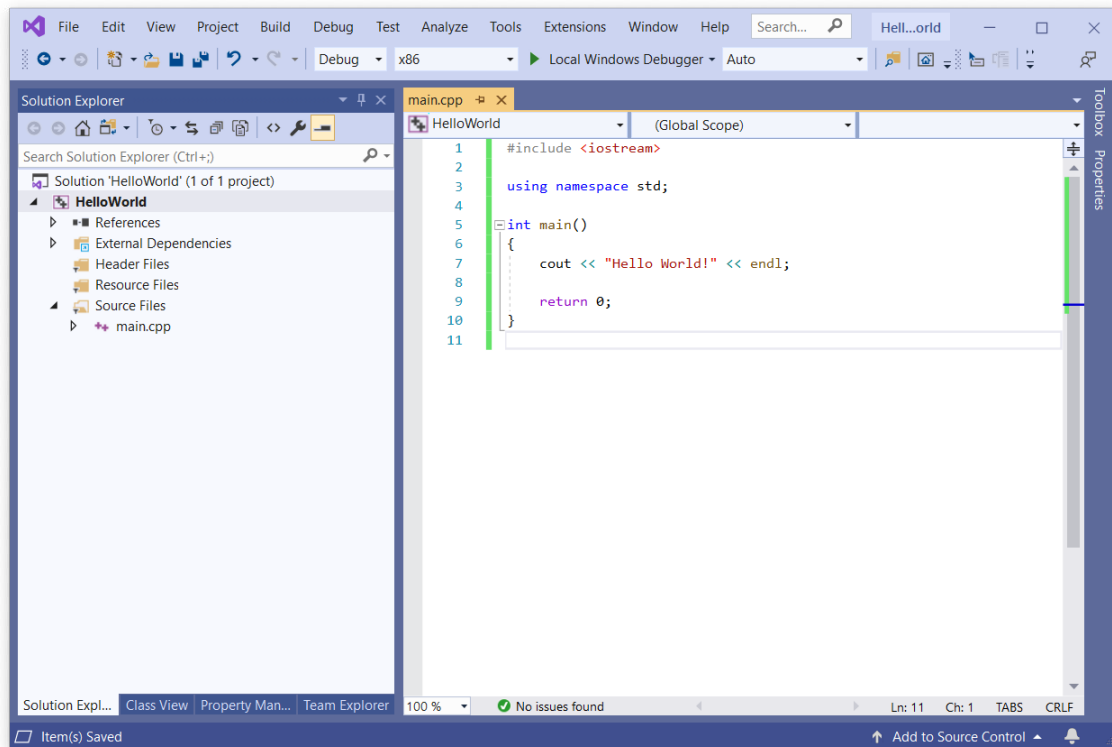


We also have to change the language standard. By default, Visual Studio sets the C++ language standard to C++14. In COS30008, we use C++17.

Select the project name and right click. Now select [Properties](#), go to [Configuration Properties/General](#), and choose [ISO C++17 Standard](#) for C++ Language Standard:



### Third Step: Our first program.



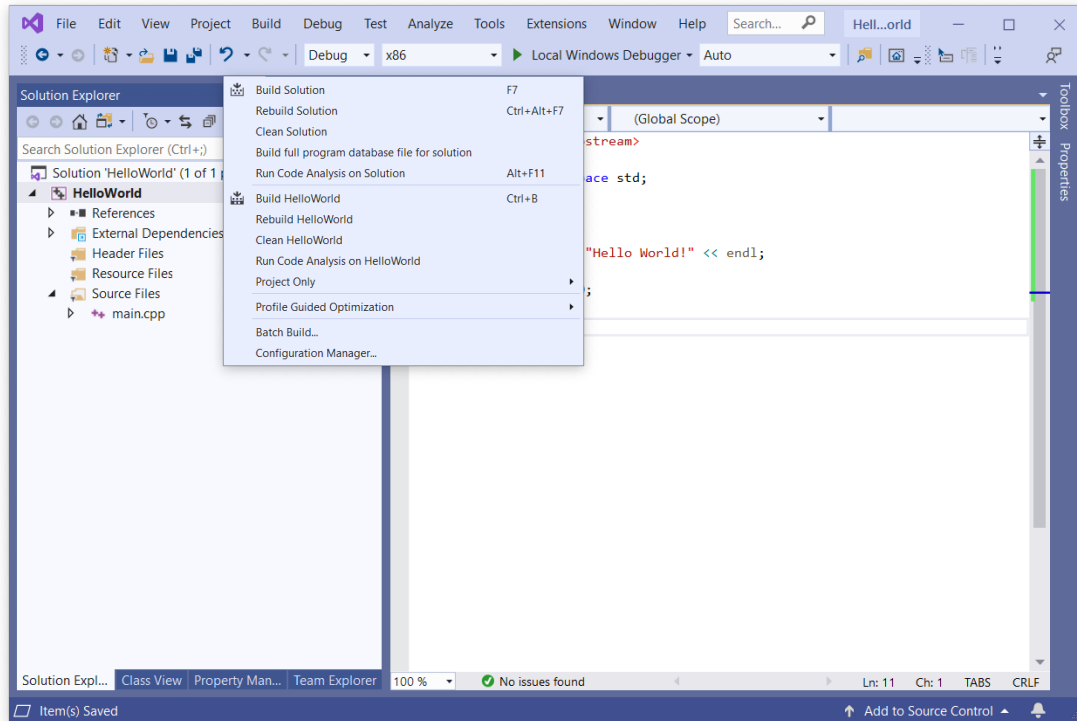
In particular, we write:

<code>#include &lt;iostream&gt;</code>	Include the definitions for console I/O. We are using two I/O values: <code>cout</code> (standard output) and <code>endl</code> (the environment-specific newline).
<code>using namespace std;</code>	Tell C++ to look for all library names in namespace <code>std</code> .
<pre>int main() {     cout &lt;&lt; "Hello World!" &lt;&lt; endl;      return 0; }</pre>	<p>Our program just prints the string "Hello World!" to the screen.</p> <p>At the end of the main function we <code>return</code> the value <code>0</code> – success – to the operating system (i.e., Windows)</p>

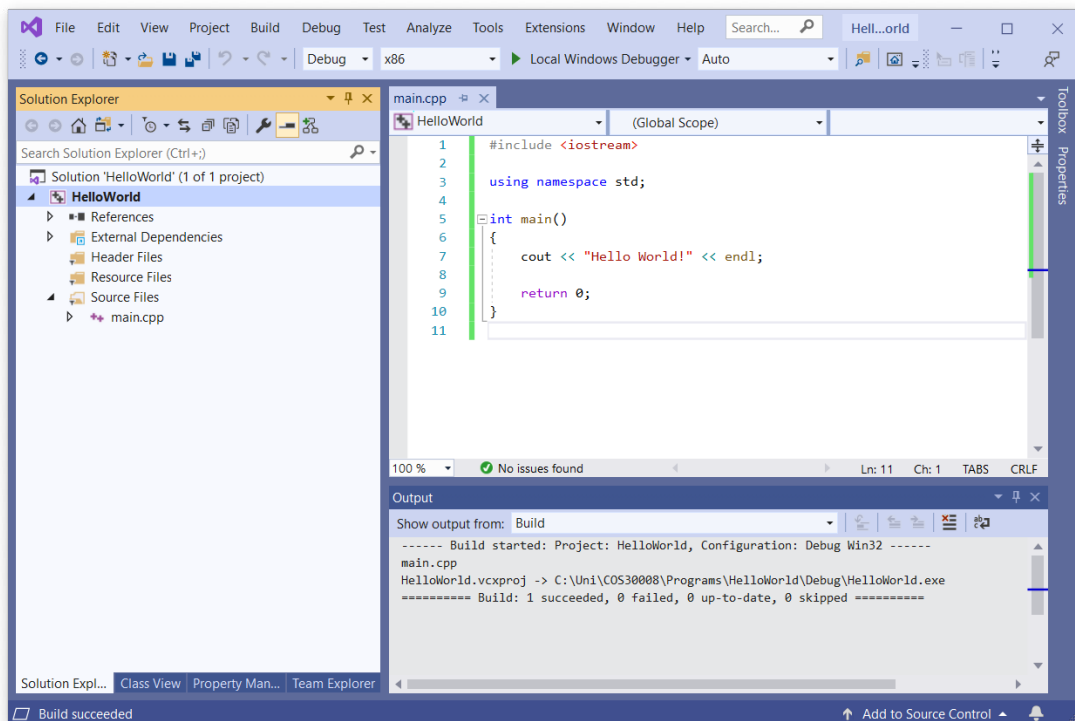
Save your work with **Ctrl-S**.

To build the program, either press **F7** or select the menu item **Build/Build Solution**:





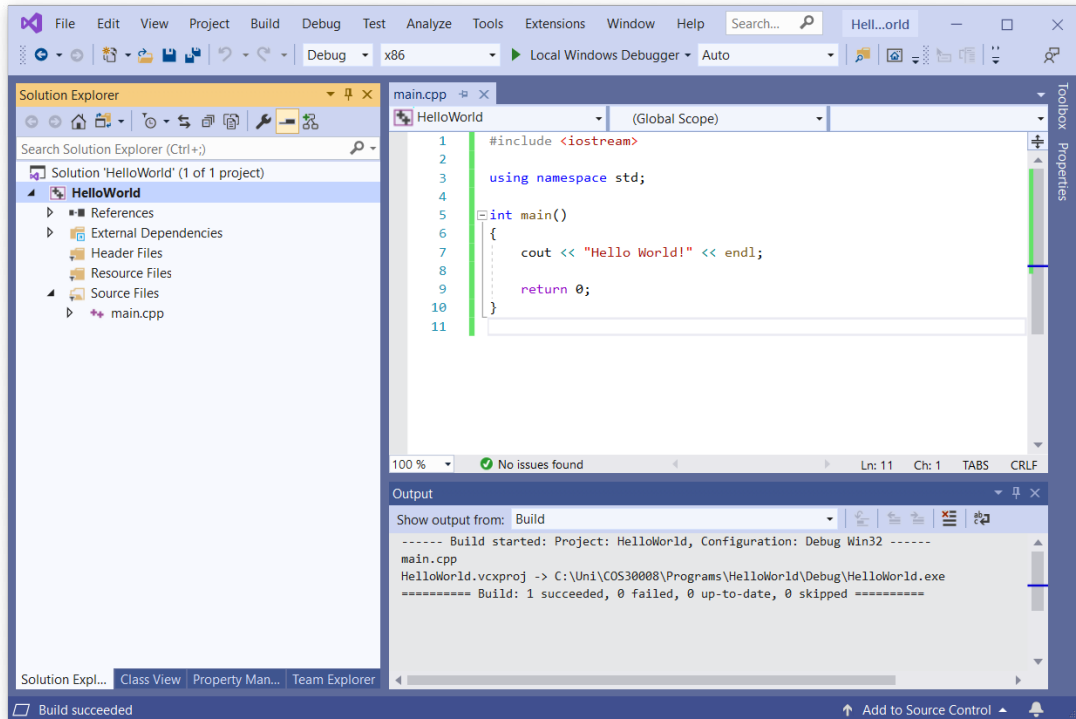
If everything goes well, the output should look like this:



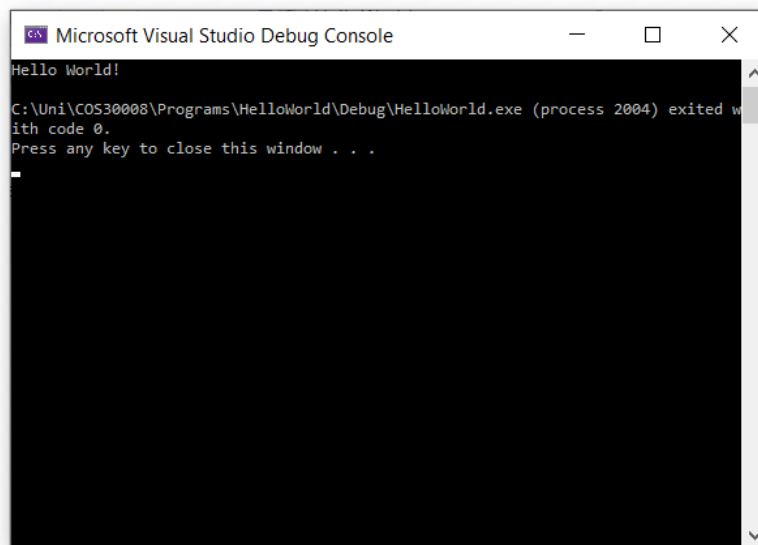
If there are any errors, check the source code, try to correct them, and rebuild your project until the build process succeeds. Check with your tutor.

### Fourth Step: Running the program.

To run the program press **CTRL+F5** or select **Debug/Start Without Debugging** from the menu:



We should see the following command window (press any key to close it).



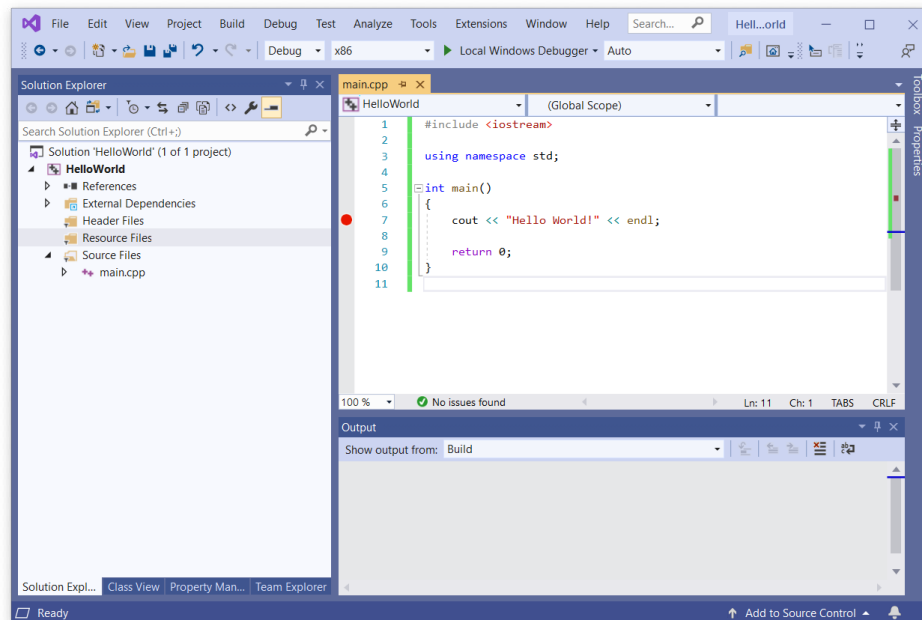
You have successfully built your first C++ program with Visual Studio. This should really work!

## Fifth Step: Debugging.

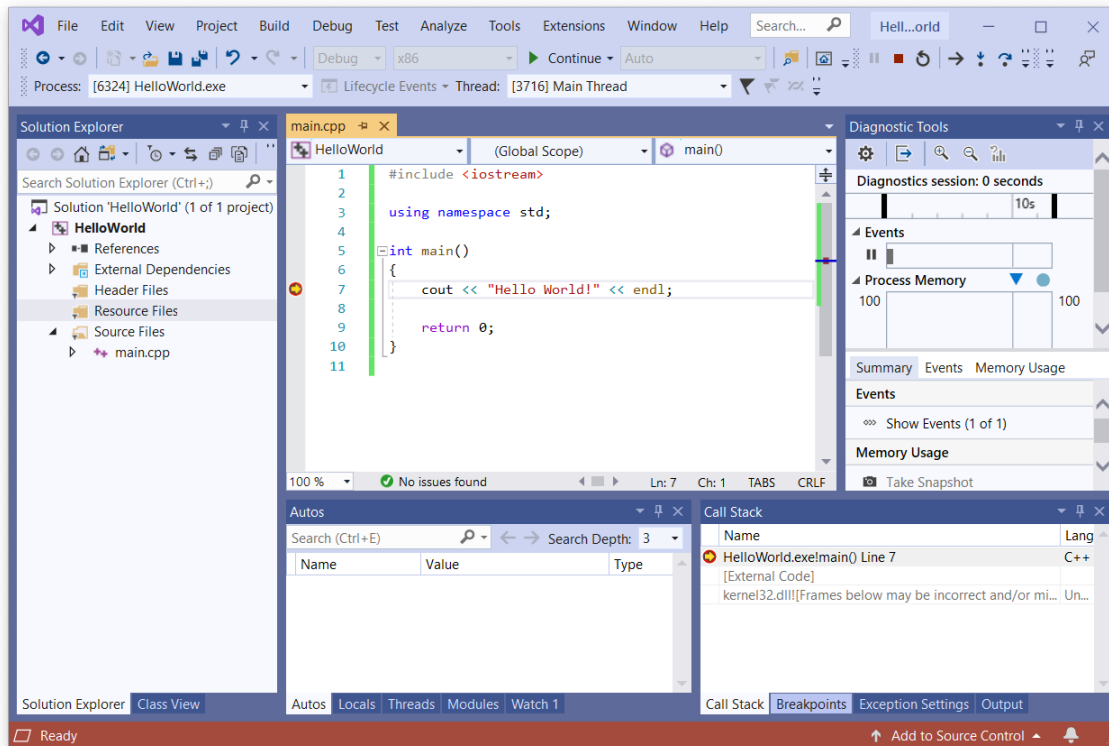
Unfortunately, it happens often that our program contains bugs. To locate them we need to test our program. The simplest form of testing is debugging, a step-wise procedure to monitor how the program evolves. Visual Studio is equipped with a built-in debugger. You can activate a debugging session with **F5** or [Debug/Start Debugging](#).

Try it. You will notice that the program starts, a command window appears briefly, and then disappears immediately without giving you a chance to see the output. This is normal. The debugger executes the program normally and stops only at enabled breakpoints. Since we have not specified one, the debugger will not stop anywhere and just run the program as usual.

To specify a breakpoint, we need to position the mouse pointer on the gray area left of the editor window. To activate a breakpoint at the start of our program, click (left mouse button) on the area next to the first line in the main program:



Start a new debug session and you will see that the debugger stops at this line:



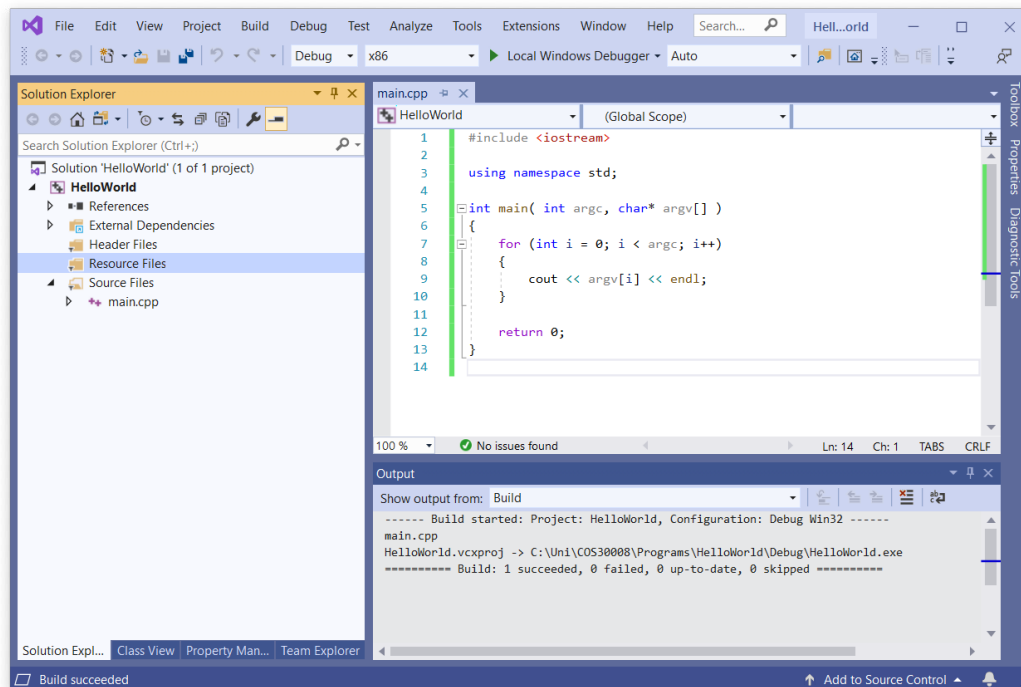
You have four options now:

- Continue (F5) – the debugger will resume the program until it reaches the next active breakpoint or terminates the program normally.
- Step Into (F11) – the debugger steps into the next function to be called and stops again. You can use this facility to inspect functions that you have written.
- Step Over (F10) – the debugger will execute the current line and stop again. The facility is useful, when you want to monitor the progress of your program in a step-wise fashion.
- Step Out (Shift+F11) – the debugger will resume the current function and stop at the line where the function has been called. This facility allows you to return to the so-called surrounding scope of a function. The surrounding scope of the function main is the C++ runtime environment.

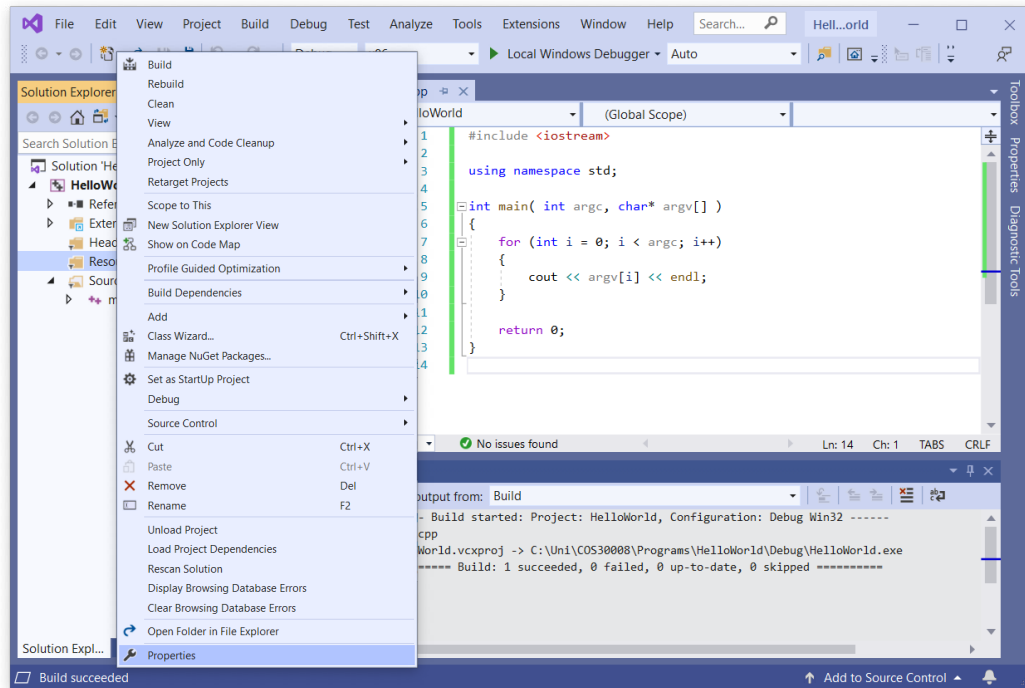
Play with all options. Currently, there may be no difference. This may change when you work with more complex programs.

## Sixth Step: Working with command line arguments.

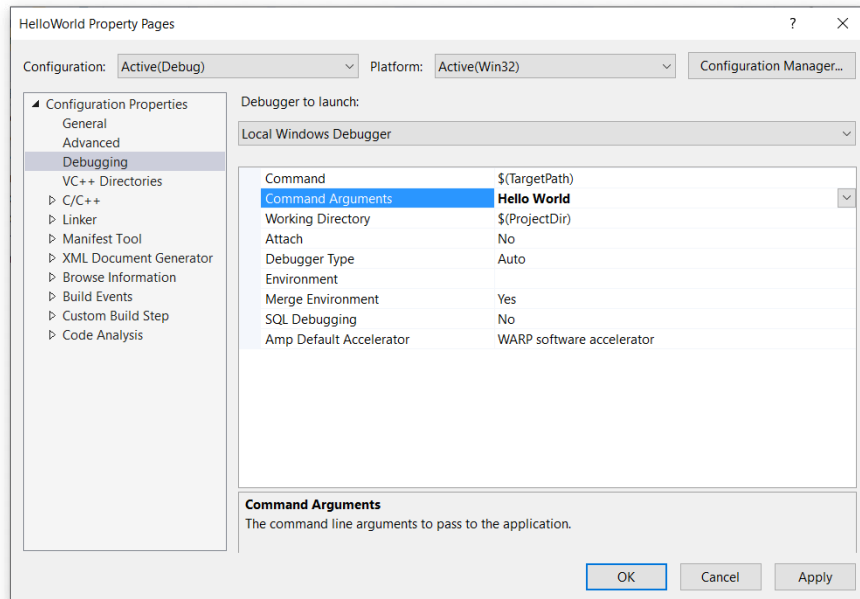
We change our main function to print the command line arguments to the screen. For this reason, we add `int argc` and `char * argv[]` as arguments to `main`. By convention, `argc` contains the number of arguments, that is, the number of elements in the array `argv`. The parameter `argv`, on the other hand, contains an array of C-Strings (i.e., char arrays terminated with `'\0'`). The element at index 0 is always the name of the command. We use a simple `for-loop` to iterate over the `argv`-array and print each element in turn.



We need to tell Visual Studio that we would like to attach some [Command Arguments](#). In our example we write [Hello World](#), that is, we define two arguments [Hello](#) and [World](#). Application-specific setting can be applied in the [Property Pages](#) of the current project. You need to right-click on your project [Hello World](#) in the [Solution Explorer](#) and select [Properties](#):

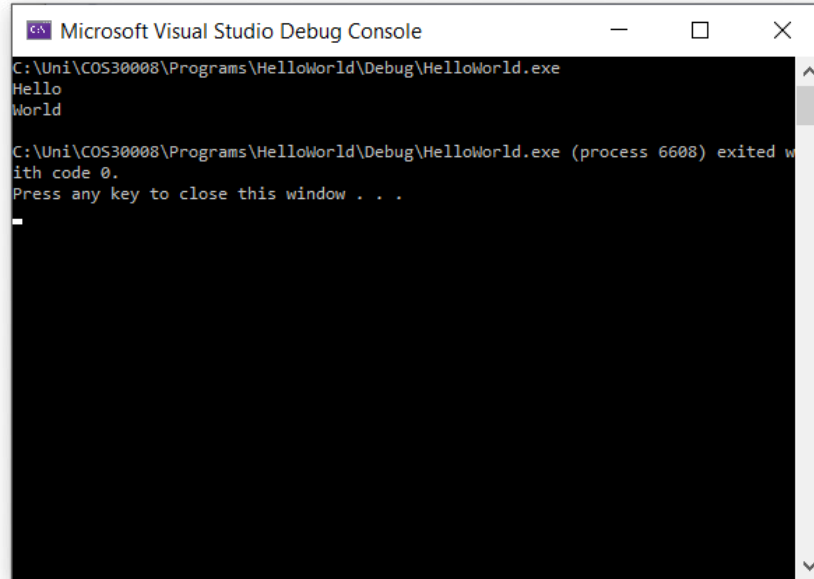


This brings up the following dialog window:



You need to select [Configuration Properties/Debugging](#) in the left pane. In the right pane, you can now specify the desired [Command Arguments](#) (edit value).

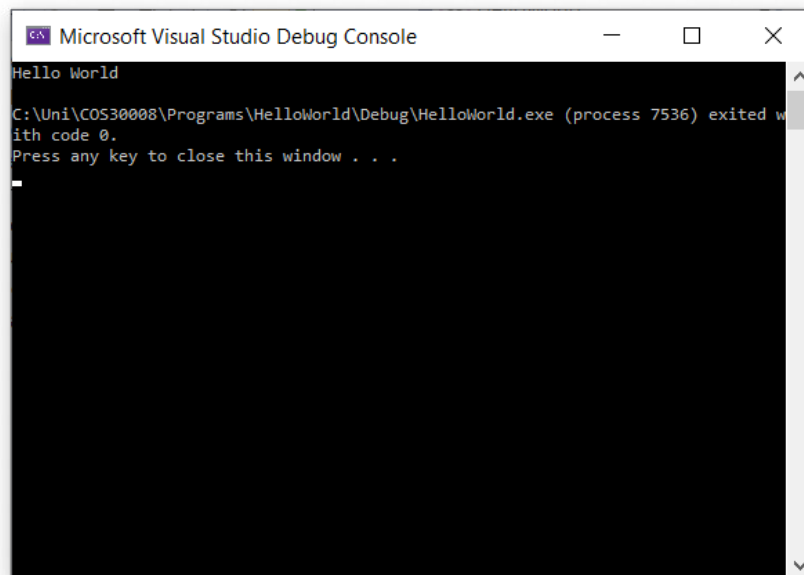
Press [OK](#) to accept the settings and test the program using [Start Without Debugging](#). You may need to build it again.



```
Microsoft Visual Studio Debug Console
C:\Uni\CO530008\Programs\HelloWorld\Debug\HelloWorld.exe
Hello
World
C:\Uni\CO530008\Programs\HelloWorld\Debug\HelloWorld.exe (process 6608) exited w
ith code 0.
Press any key to close this window . . .
```

The output should consist of three lines: the full path of the program, the line "Hello", and the line "World".

Can you find a way to print only "Hello World" in one line?



```
Microsoft Visual Studio Debug Console
Hello World
C:\Uni\CO530008\Programs\HelloWorld\Debug\HelloWorld.exe (process 7536) exited w
ith code 0.
Press any key to close this window . . .
```

## Step Seven: Build and Run the Lecture Program

