Assignment 1

Intro to C++

Contents

- Introduction
- Getting Started
- Visual Studio Layout
- Adding a File
- Coding
- Code Breakdown
- Running your code
- Running your code AGAIN!
- Errors
- Submitting your code
- Zipping a File

Introduction

The basic idea of the first assignment is to get your development environment setup and ready to go. You'll be able to reference this document for the first 2-3 assignments to remind yourself of the submission process.

In this class, we'll be writing C++ programs. Creating programs is a process that involves an editor, compiler, linker, and more. So, to simplify all of this, we are going to use an Integrated Development Environment **(IDE)**. The great thing about an IDE is that it has everything we need to make programs in one convenient package. In fact, most IDE's have much, much more than what we'll need in this class.

With that said, we'll be using Visual C++. It is recommended you use the most current version of Visual C++ Community Edition, as this is the version we will use to test your programs. You can get it here.

Author: Dr. Jeremy Hatcher
Email: jhatcher@fsu.edu

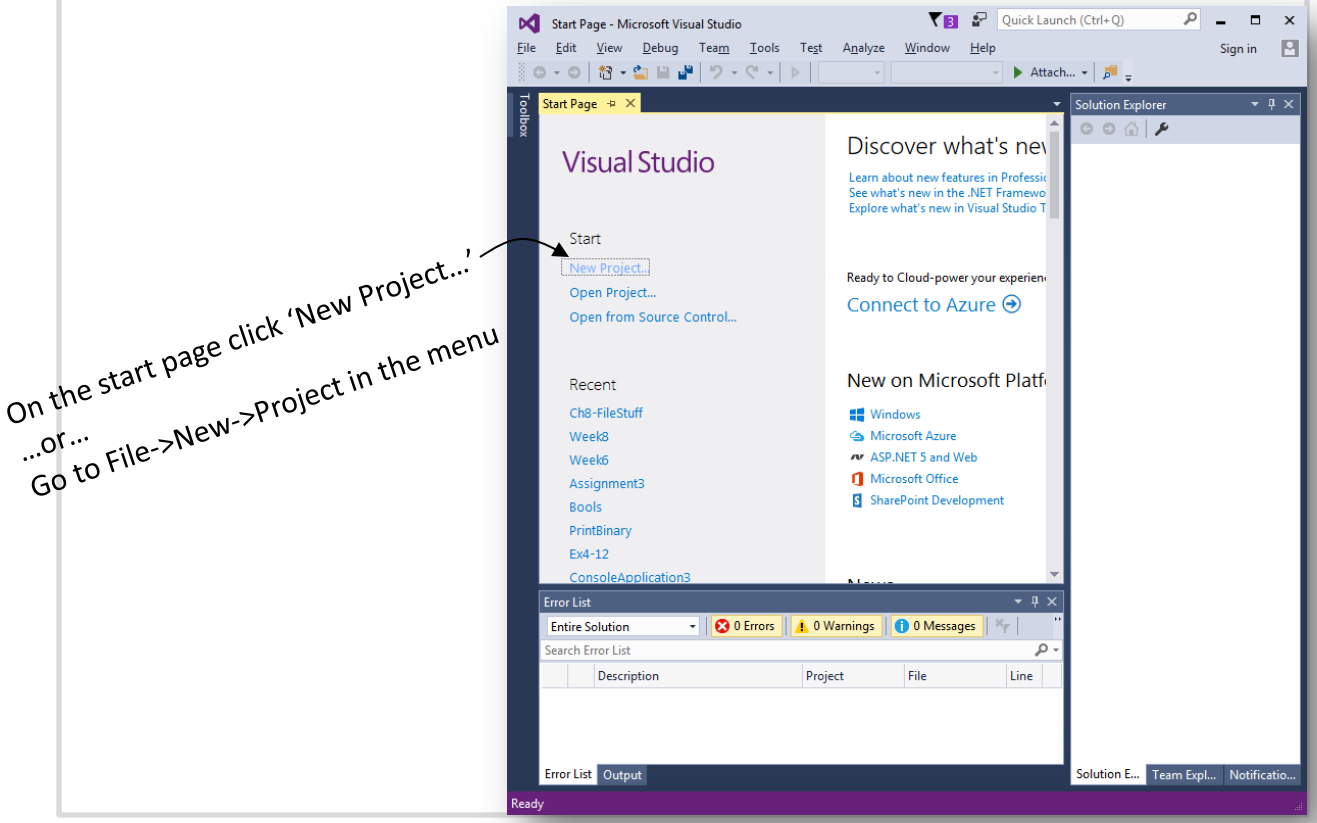
Getting Started

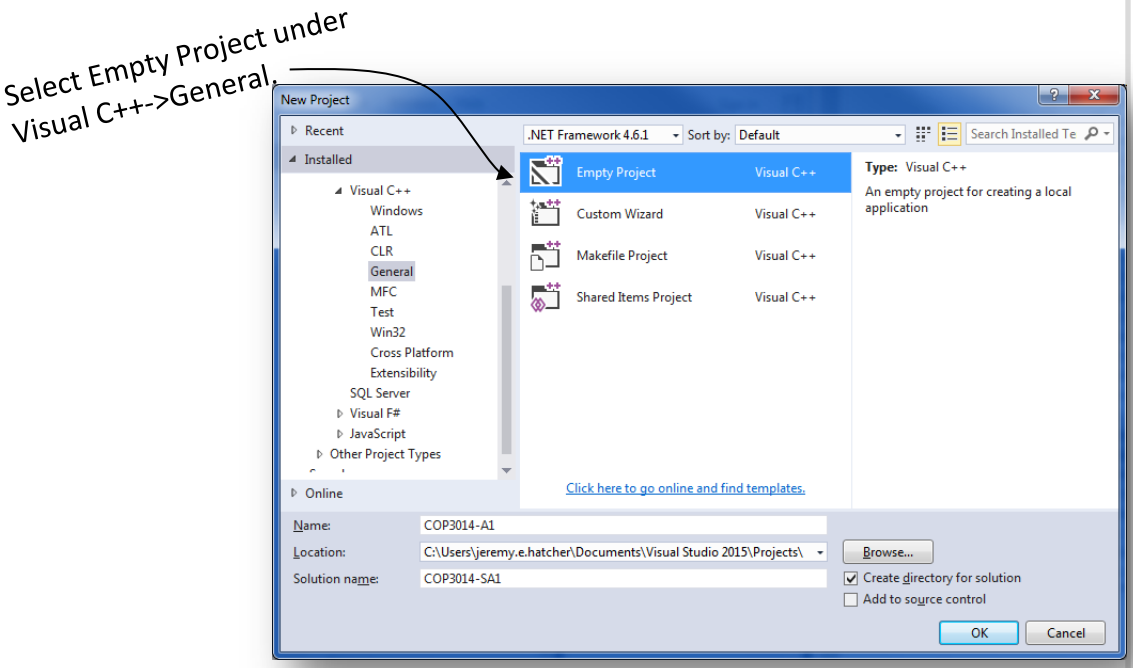
After installing Visual Studio, you are ready to write programs. Here are a couple of suggestions before writing each program:

1. First, make sure you have read the material for the weeks that correspond to the assignment. If you just try to "wing it", you will find yourself having to reference the book too much.
2. Understand the problem at hand. To do so, have an idea of what you want to do before you start to write your program. The book mentions something called pseudocode in chapter 4 that helps to formulate algorithms. Don't get too caught up with all the lingo and specifics of the book's pseudocode. The whole idea is to plan out your code before writing it. That way, you can have an idea where you're going and spot any potential roadblocks sooner rather than later. What I like to do is write the parts of the program that I immediately understand and then plan the details later. Do whatever helps you understand the problem before writing the program.

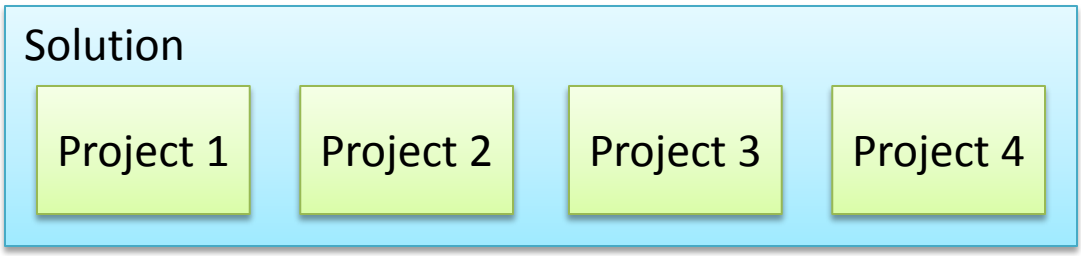
– **Creating a New Project**

- Once you understand the material and have a plan, it's time to write your program. Open Visual C++ and begin every program like this:

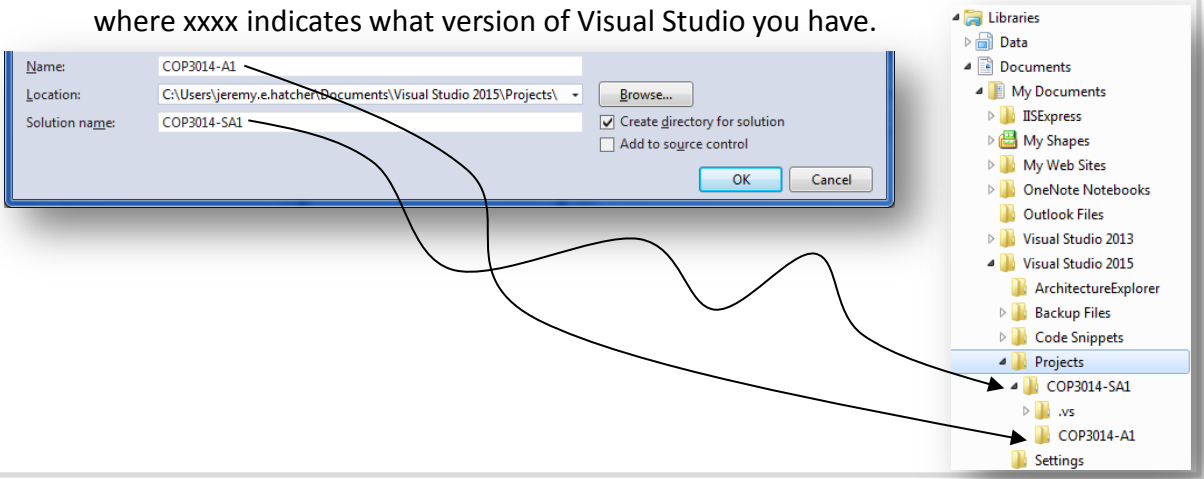




Visual Studio uses '**solutions**' and '**projects**' to organize your files. A solution may contain many projects. In this class, each solution will only have 1 project.

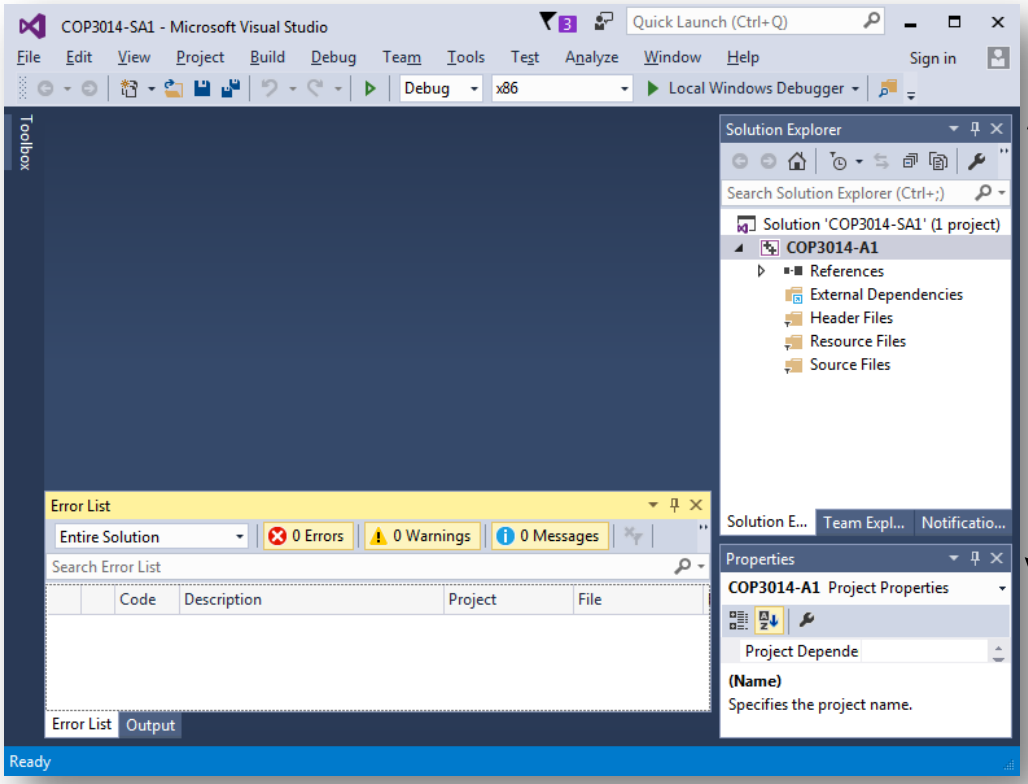


Give the solution and project a name. For this assignment, name the project COP3014-A1. Name the solution COP3014-SA1. The solution and project name correspond to the folder hierarchy that is created for your files. The default location for these files is in your 'Documents/Visual Studio xxxx/Projects' folder, where xxxx indicates what version of Visual Studio you have.



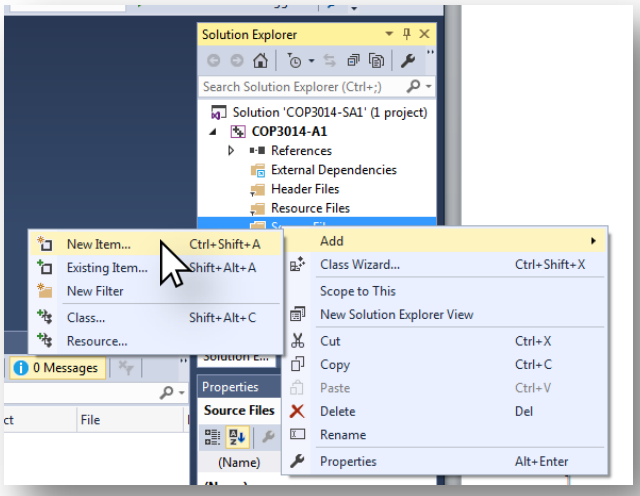
Visual Studio Layout

- Toolbox: We won't use it in this class
- Error List: Look for errors here – and fix them!
- Properties: You might be able to make use of this but you don't have to
- Solution Explorer: This is where you'll see the files you are currently using

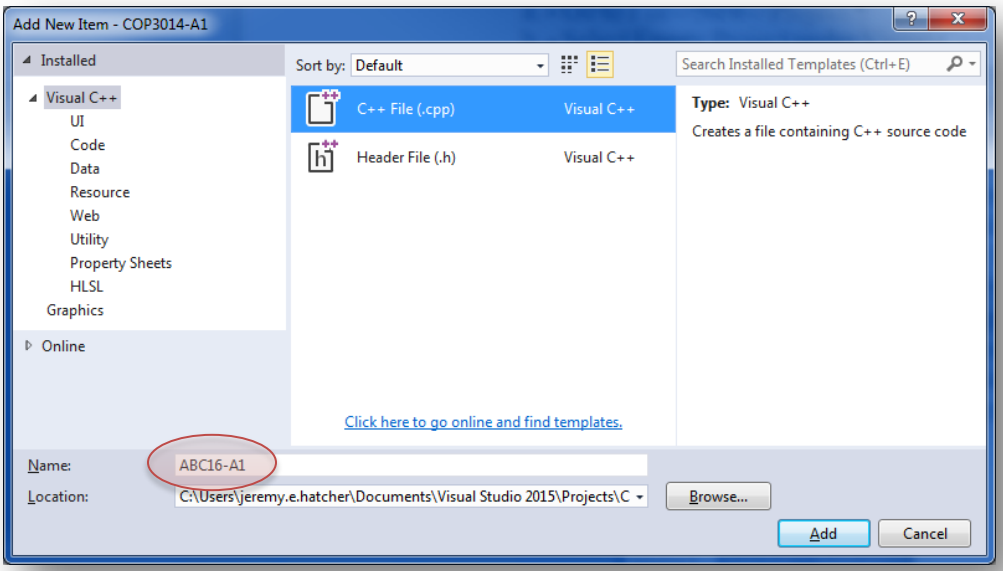


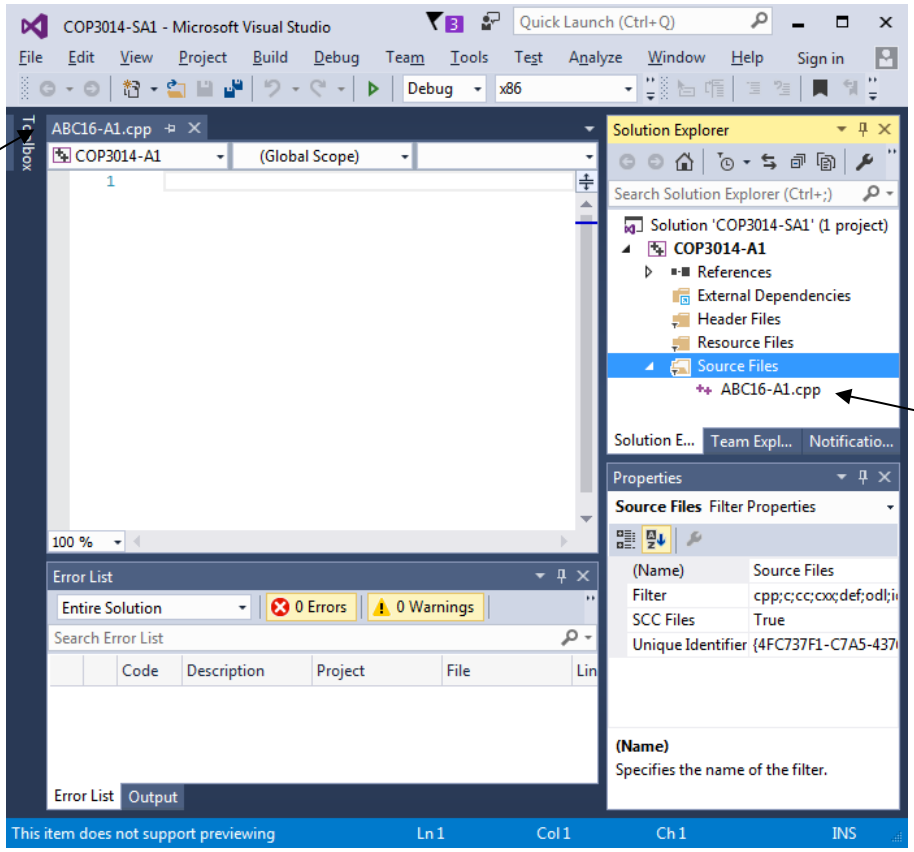
Adding a File

Right-click on the ‘Source Files’ folder and select Add->New Item.



Select C++ File (.cpp). Name the file using your FSUID and the assignment number (A1). For example, if your name is Alpha Bravo Charlie, having enrolled with FSU in 2016, and this is assignment 1, use **ABC16-A1**.



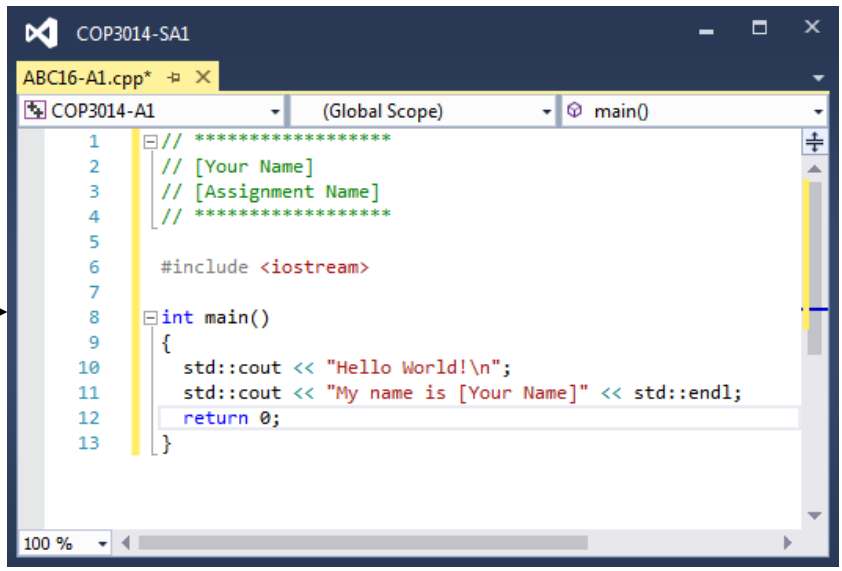


An empty code window to type your code into

You'll find your new code file here

Coding

Now, type this into the code window →



1. Make sure to change [Your Name] to your actual name! Please don't laugh because we've had students fail to do so before.
2. Make sure to change [Assignment Name] to Assignment 1

Code Breakdown

Line #	Code	Description
1-4	//	Any line that starts with two forward slashes is a comment. The compiler ultimately ignores these
6	#include <iostream>	Tells the compiler to include another file called iostream. This other file has code in it that helps make things like 'cout' and 'cin' work properly.
8	int main()	This is the 'function' that runs when you start your code. You'll learn much more about functions later. The 'int' tells the compiler that the function will return an integer at the end (see line 12). The parentheses are the input arguments (there are none right now).
9 / 13	{ }	Curly braces open and close a function.
10	std::cout << "Hello World!\n";	std is a namespace – read about it in the book. cout is a function that tells the compiler that what comes next should be printed 'out' to the screen. << This is the stream insertion operator. You are inserting "Hello World!" into the 'cout' stream. \n tells the stream to start a new line.
11	… std::endl;	Same as above but with different text. The endl is another way to tell the stream to insert a new line.
12	return 0;	You are returning the integer value '0' to whoever called this function. In this case, it was the operating system. Don't worry too much about this for now. Worry about it when we get to the chapter on functions later. For now you just need to know that the code starts operating in the main function

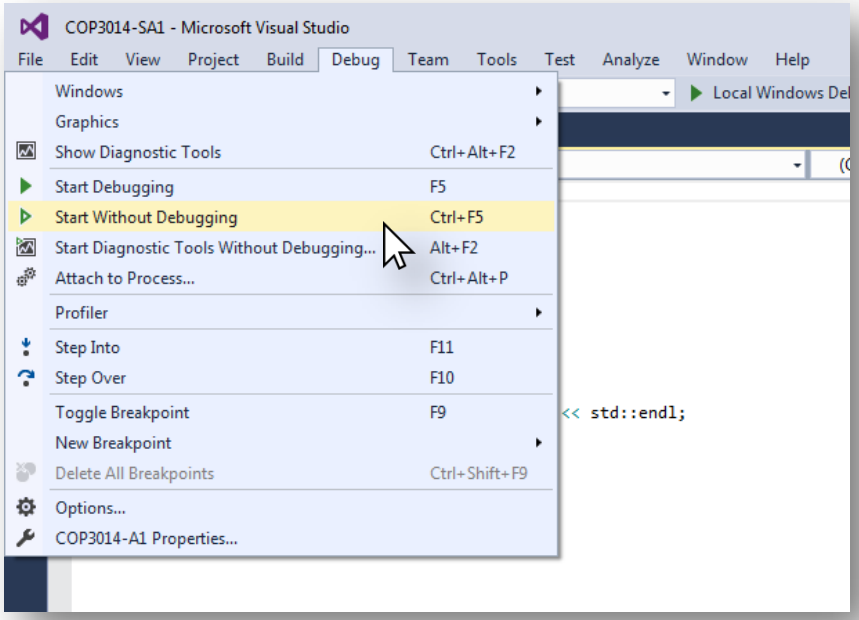
Notice that each line of code ends with a semi-colon. The comments, #include statements, function declaration, and opening-closing curly braces are exceptions to this. There will be more exceptions later. This will be easy to understand as you learn.

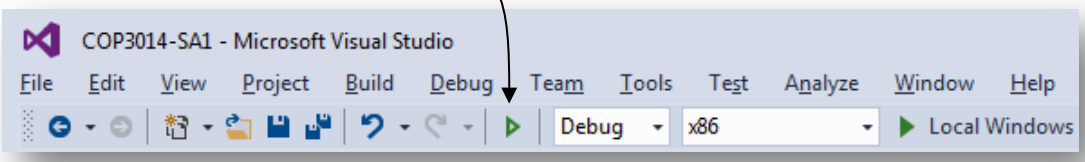
Running Your Code

Click ‘Start Without Debugging’

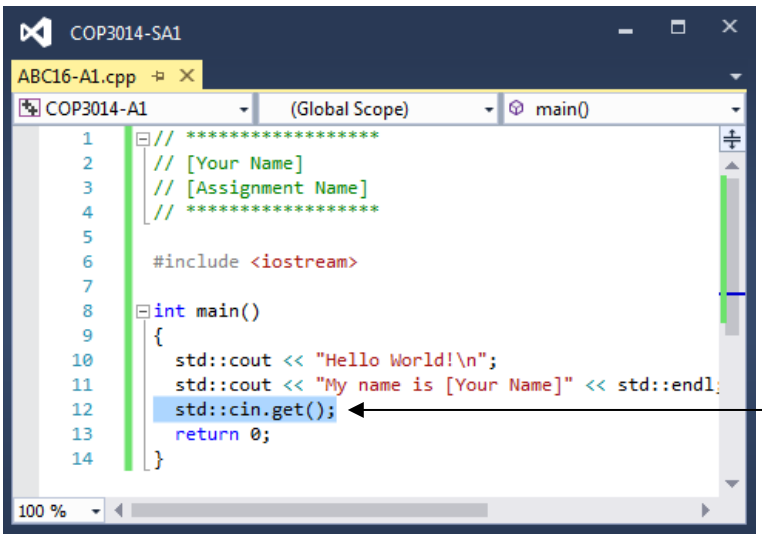
You’ll find it under the ‘Debug’ menu.

You may also have a button for it on your toolbar already.



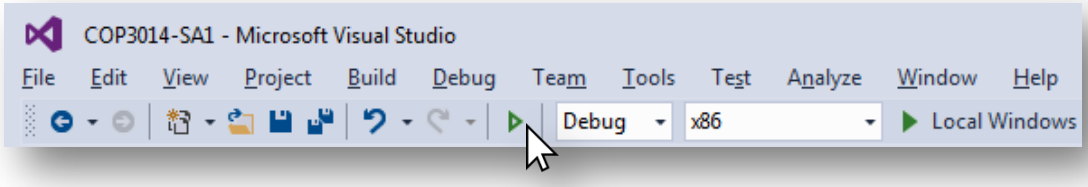


Did you notice a window pop open and then close very quickly? That’s because your code started running, got to the end, and closed. Let’s add a line of code that will make it stop when it gets to the end.

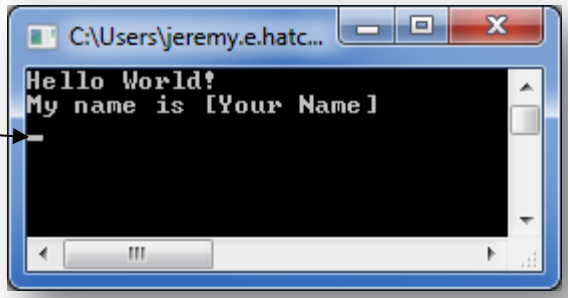


Don’t worry about WHY at this point. Just understand that this line of code tells the computer to wait until the user hits the ENTER key before proceeding.

Running your code AGAIN!

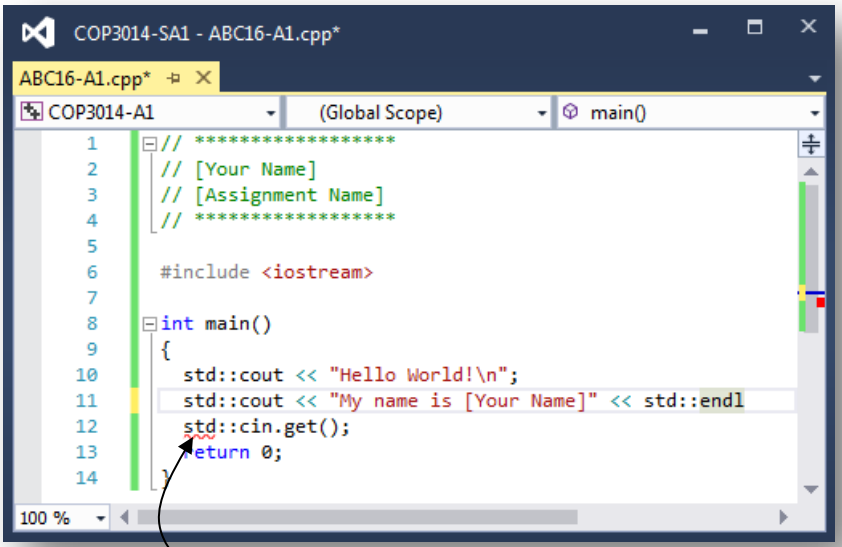


This time when the code runs it stops at the end and waits for you to hit enter. Here is the marker for the cursor



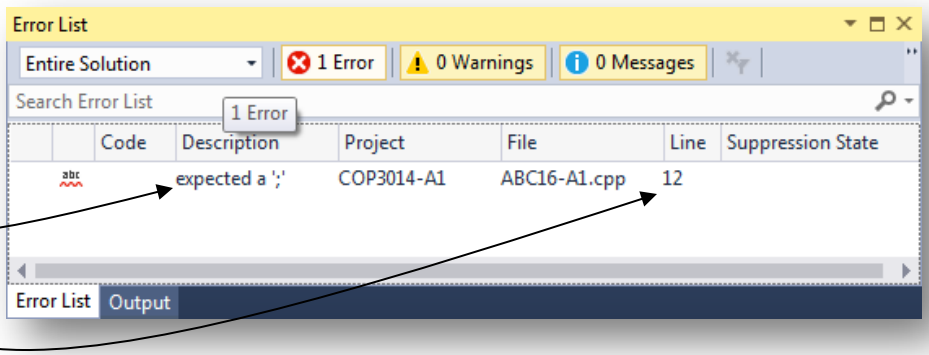
Errors

Congratulations. You just finished your first program. Let's take a quick look at some common errors. Remove a semi-colon from line 11:

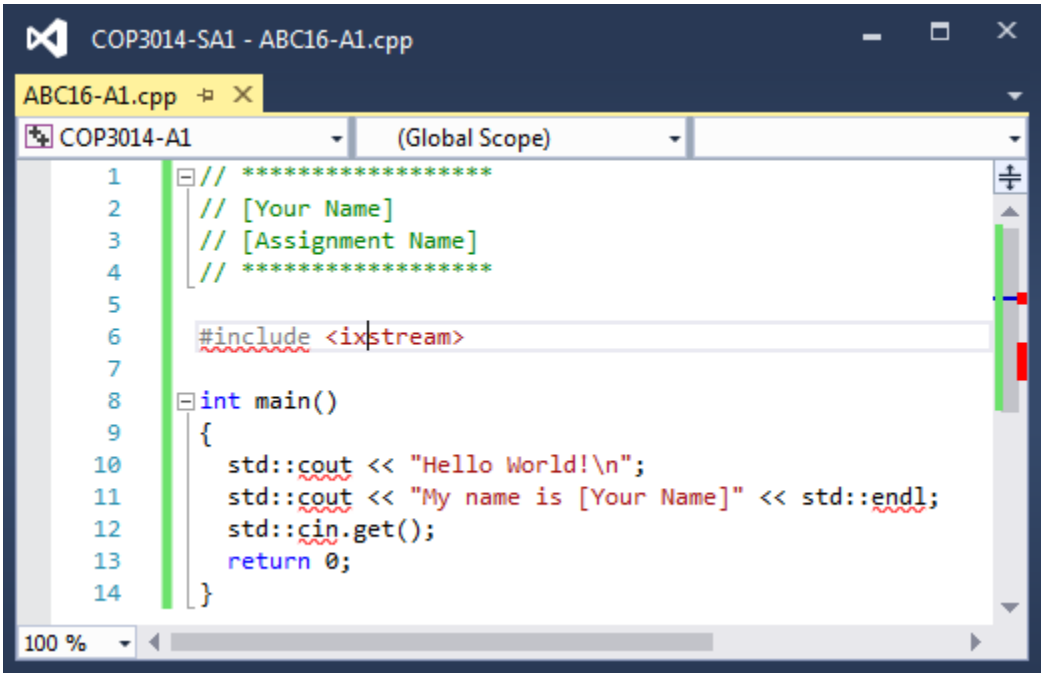


The IDE underlines in red where it has a problem.

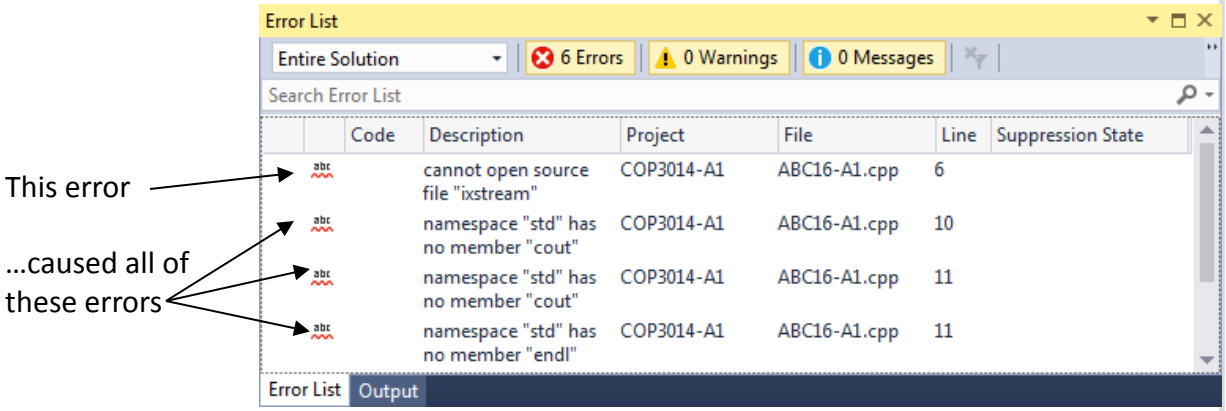
The error list tells you what the problem is and where it detected it



Let's rename 'iostream' to some bogus name that doesn't exist

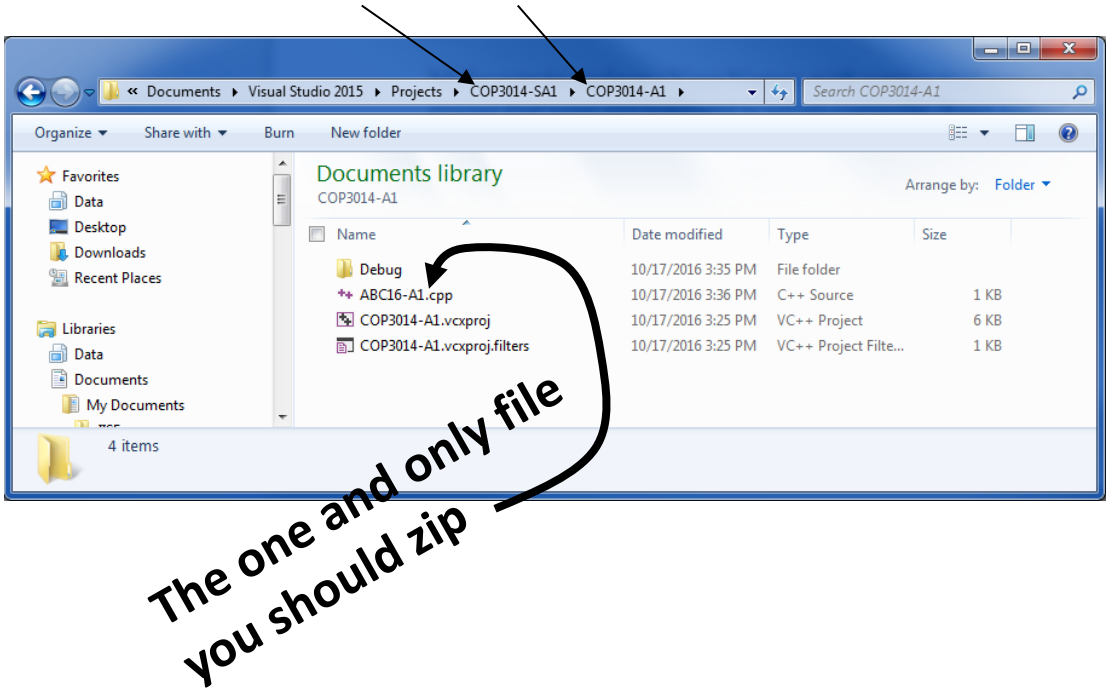


Uh-oh! This caused lots of errors. Usually when you have a lot of errors you can fix more than one by starting at the top and working your way down.



Submitting Your Code

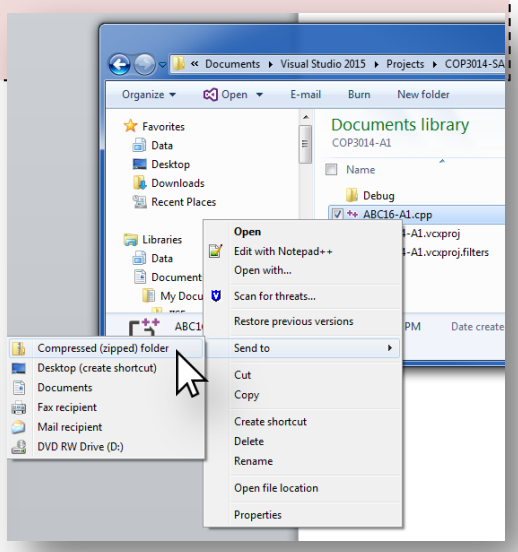
Zip up the file and title the zip with the following format: FSUID, dash, assign1 (e.g. Alpha Bravo Charlie might be ABC-A1.zip). This should be found in the directory where your project is at (see this page). The default is in your "..\Users\[username]\Documents\Visual Studio xxxx\Projects\[Solution Name]\[Project Name]" folder.



Please do not submit any other file than the one-and-only-one cpp file that you create. Make sure that the extension is CPP and **NOT** SLN or something else.

Zipping a File

In Windows, you simply right-click the file, click 'Send to,' and choose Compressed (zipped) folder. The default name will be the same as the file name. This will work fine.



This is the file you should submit on BlackBoard