Compilers for C++

1. Apple’s xCode
2. g++ is a compiler made for unix systems. I didn’t have to do anything but issue the g++ command in my terminal. My Mac already had “g++” once I installed xCode. Because xCode has Clang and g++ is a product that comes from Clang I am able to use it. If you type g++ -v you will see details about the g++ version, you will see the full break down. Essentially, these products both come from the bin folder of xCode, meaning it’s a good idea to install xCode if your on a MacOs.
3. Clang is the default compiler for the MacOs. Clang does come with xCode so you do have to add it but you end up with a nice setup that will allow you to write C++. xCode is actually designed for other languages pertaining to the iOs like swift and object-c but it can also be used for C++.
4. Dev-C++ is an IDE for the windows platform. This product has a compiler built into it and can be used to develop c++ applications. Some of the features that appeal to me are the syntax highlighting (what is a variable, function, static item etc when looking at your code these items are rendered in a different color) and the tab completion of course. It also has standard IDE capabilities like debugging and over all code formatting. I don’t see the point in researching other compilers and IDE’s because they are all very similar.
5. My personal choice is the VScode IDE. I will be using the g++ compiler since it is pretty much baked into my system at this point and VScode which is an IDE by Microsoft and it is mainly used for js applications. I use it for TypeScript and Javascript already so it makes sense to add a plugin that will allow the handling of C++ code. VScode also has the standard bells and whistles of any IDE like syntax highlighting, tab completion, debugging and more. Plus, I already have Android Studio, xCode, VScode, IntelliJ, Eclipse and sublime text installed..I would rather install a simple plugin rather than adding another IDE.

Instructions for using the g++ compiler are simple, first you need to write a program and save it in a file called (for example, Launch.cpp) this will serve as the source file for the compiler. Your next step is to invoke the compiler by issuing “g++ -o startLaunch Launch.cpp”.  Lets break this down, g++ is the actual compiler, “-o” means output and “startLaunch” is the name of the output file which is the actual executable file. “Launch.cpp” is the source file that g++ uses to compile machine language out of.