Out of figure 1.3 what is being displayed is the order of execution a program does off a compiler. So, languages such as C, C++, C#, and .NET where when a program in these languages is written and ran, they get compiled first before executing from the Operating System. The program that is initially written in these languages is called the *Source Code* and upon running this code results in a several build steps before execution of the program. Firstly, the Source Code is passed to a Lexical analyzer which is just a fancy term for a program that will go through the source code mapping key syntax keywords to a *Symbol Table.* Which is checked in a *Syntax Analyzer* for any errors the source program possesses. Assuming no errors are found in the syntax then from the Syntax Analyzer there are *Parse Trees* created for the *Intermediate code generator and semantic analyzer.* Parse Trees act as a hierarchal structure of code for the program’s execution. For example, if statements and for loops would have code ran before or after met conditions which is done through code layering. The intermediate code generator is what finally translates the source code to a language the Operating System can send down to the hardware layer of a computer to understand and execute. Intermediate code often is considered the assembly language for the hardware of the computer but not always. Sometimes the intermediate code needs to be further translated to assembly. Once the source code has gone through all of this to be turned to machine language it can then get executed and through the Operating System displayed for the user running the program.

Something that was briefly talked about outside of Figure 1.3 is the Linker. Which is a separate program from the compiler that links the executable code to different runnable addresses in memory which allows the program to be ran seamlessly without the need to in real time jump around different address points in memory for the code to be executed properly.