**Programming Languages Explanation  
Module Seven Assignment  
Ian Heritier**

*1. Explain the benefits and drawbacks of using C++ in a coding project.*

C++ is highly versatile and is a good place to write your main user interface code. Having access to the standard I/O stream is especially nice when doing things like printing a menu and responding to user input. I’m personally a fan of the C family of languages in general for writing code that is neat, efficient, and effective, and there’s not much that it can’t do if you put a little extra work into it. Working within Visual Studio makes it that much better because it’s a fantastic IDE. C++ can easily take care of output formatting as well, which is important when considering a clean user experience.

One downside of using C++ is that some of the tasks that one has to use for a project like this – particularly File work – involve a lot of extra setup and are not efficient at all. It also isn’t easy for a non-coder to read a complicated program in C++, no matter how much documentation you put into it. Things like chained dot notation and some other syntax-heavy operations are a lot to expect a beginner to wade through.

*2. Explain the benefits and drawbacks of using Python in a coding project.*

I think Python is sort of in the opposite situation compared to C++. It’s made to be easy to read and work with for beginners, and it’s very good at certain things while not being as versatile. File work in particular is very easy in Python compared to C++, and that was extremely beneficial in this particular project, where I would have had to write a significant amount of extra code if I’d attempted to do the whole thing in C++. The Python code I wrote here was also very modular, increasing the possibility of reuse for different applications. I have always thought of Python as “coding for non-coders”, which is sometimes pejorative but when it comes to working with a team I can see why it has such a huge appeal.

A drawback to using Python is that it isn’t designed to handle Object-Oriented Programming; it can do the job, but it’s cumbersome code to write. Much of the easy “WYSIWYG” effect that’s present in C++ output formatting is absent here as well; combining strings can sometimes be a frustrating process in ways that are hard to foresee if you’re used to working in C languages.

*3. Discuss when two or more coding languages can effectively be combined in a project.*

It seems from this project that combining separate languages as modules under Visual Studio or some other IDE is best for when you want to optimize a task to a particular language. I think you want to have your “main” code only in one place (so far, I haven’t seen evidence that there’s a better place than a CPP file), but then attach it to separate code files for whatever works best. One example that I think would be especially useful is connected a console app to something like JavaScript or PHP and creating a de facto web app that I could run from Visual Studio and get online. I haven’t done anything like that yet but it sounds especially interesting. This particular program could easily be put online with a better parser (that works with something beyond the simple text and .dat files) and expanded to create some really useful and interesting applications.