Kyle Zalewski

CS202

Prog4 New Syntax

The biggest adjustment for me so far when moving from C++ to Java is by far input/output. There are significant differences that make chasing down a proper solution tricky. For one thing, since every operation is a function call, and moreover an object is required for that function call, simple iostream operators are a thing of the past. Though this is great for overall concision and uniformity of code, it’s pretty tough to find exactly which types of objects need created and which functions to call on those objects. In the future I’ll have to spend some more time investigating best practices and researching which specific functions to use and why they are optimal for the specific task I’m applying them to.

Undoubtedly there are numerous acceptable functions for each input/output operation I desire, which both adds and subtracts from the overall difficulty. Though it’s nice to have many options, it’s a bit overwhelming when there are many different ideas for how to tackle what seemed like a simple problem. What I found to be very helpful is some very smart overloading of some of the functions. I was having some trouble getting file output to properly append data instead of overwrite it. As it turns out, I simply needed an additional argument to the function call to tell it to append. Experiences like that definitely expose the power of structuring a language this way. I’m glad to have found that functionality in included class definitions, and will certainly keep things like that in mind for future methods I write myself.

I’m still getting used to the idea of everything being in a class, and this project threw a few curveballs my way to help me get there. Since I’m not able to just define a function any place and then call it (needing to instead create an object first), I think it helps me to program more intelligently and decide on a case-by-case basis what class a particular method should be part of and properly implement it from the beginning. This saves me from the eventual trouble of having to pass data around in endless circles until it reaches the proper class. These principles are certainly a good takeaway for any future projects of mine, including those in other languages that are not so strict from the beginning. Having the freedom in C++ to define and use functions wherever you please can be handy for troubleshooting, but in the end it’s a bit of a crutch.