The array data structure was consistently faster than the ArrayList. I’d like to do some statistical analysis to see if it was significant. For a project like this, where perception of speed from a user standpoint is pretty much invisible, the speed isn’t significant enough to choose a data structure based on speed rather than design. If you are in a different design context then it might significant.

The data pretty much acted as I expected. There was one surprising finding: The ArrayList filled with Floats incremented faster than the ArrayList filled with Integers. I haven’t been able to figure out why this is the case. For now, I don’t have anything more specific to say than the class differences between Integer and Float are causing this. The classes themselves don’t seem very different in the documentation, they both have the same inheritance hierarchy and implement the same interfaces.

The design process was straightforward. There were some duplicate tasks that, given more flexibility in specification, I would have used a different program design for. I didn’t have to make any smaller programs because I was familiar with the data structures and the resources for time tools in java that were available were very good. Interpreting the specifications did take a little more looking over than I expected.