Abstraction

The definition of abstraction is, in short, to break something complicated down into more simpler parts. When it comes to working with data and programming, abstraction means that we take what would be one large and complex piece of coding and break it down into many pieces that can be better controlled and managed. A benefit of abstraction is how much more easier it makes problem solving. When we strip away the differing layers from our main code and section them off into their own separate ‘boxes’, we can easily identify problems when they come up, what ‘box’ they’re coming from, and how to go from there. For example, if I were to be making a program that was to make toys, instead of focusing all of my time and effort into one big program file that has hundreds of lines of coding, I can strip all those lines off and leave only the important functions that I need on that main program. I take all the excess lines, sort them by functions or ideas, such as what would the toys be made of, how to make them, shipping them, and make separate files for those that would be linked back to the main code. An example from our classwork would be from our journal program. We made separate files for writing entries, for random prompts for the journal, how the journal itself works, and the main code that brings them all together.