This week we learned about binary search trees. In my opinion, I find this data structure to be very, very powerful. Binary search trees are the fastest type of data structure I have encountered so far. When doing this week’s project, seeing 0 clock ticks was very satisfying! These types of data structures have the ability of only needing to traverse the number of levels it contains rather than traversing each item individually.

A weakness of these trees is that they require some complicated logic to run effectively and efficiently. I have little experience with recursion and because of this I find it very hard to wrap my head around it. BST’s most important aspect is recursion because a tree can be of any size. I believe that it’s worth the effort for the performance, however.

Since each node of a BST holds data, sometimes the data can be very large, and a BST will take up a lot of available memory. In our reading, we went over effective ways to insert items into a BST. If items are inserted in random order, the BST is likely to be smaller than it would be if inserted into sorted order. The memory used when traversing a smaller tree would be less than if it were larger.