Part 2 was much simpler, and easy enough to build on the work on part 1. First, the assumptions I’ve made so far is that when it comes to preferences, for example if you say you’d like a water ride, it excludes non-water rides from the choices. However, if you say N to that, that you do not have a preference, water rides are not excluded. That is to say, only stating an explicit preference for something means we look for those rides or exclude the ones that don’t match.

For this part, I didn’t change much from part 1 for getting single rides. Basically, I was already constructing a Ride object so I extracted that to a method which now generates and returns a collection of all 20 rides. From there, with appropriate validation we choose the correct ride object and pass that into our question tree traversal method (I had written it in part 1 to work with any ride).

For generating the recommendations for the whole park, I got clever here. Starting with the whole park in a list of rides, as we traverse the tree we filter the list of rides based on the user recommendation. To do this, we use java lambdas .filter method to create a filtered ride list from the original as we progress. At the end, we are left with a ride list which contains only the recommended rides left.