G) To fill in the gaps for the definitions of generalized AND, generalized OR, neutral true, and neutral false, I used the table given at the end of the notes on Week 5’s Semirings PDF. On there, the generalized AND was noted as addition, but given that we need to handle Cost being either a TheInt or an Inf value, I used the addCosts function. For generalized OR, I used the corresponding minCost function that I just wrote. For gtrue, I set it as TheInt 0 since the neutral element for generalized AND operations is 0. Similarly, I set gfalse as Inf because that’s the neutral element for generalized OR operations.

H) I basically did the same thing as for G in terms of relying on the table given on the Semirings PDF and using the values for output strings. The generalized AND operation was lifted concatenation, and to implement it, I created a helper function in which I iterate through each element in the first set of lists x, create a list of lists, and concatenate the first element in x to each element in the list of lists y. I then concatenate this result to the result of recursively calling the helper function on the remaining elements in x and all of y. For generalized OR, I simply just directly concatenate the two operands together. For gtrue and gfalse, it took a second for me to realize what was going on and to figure out how to distinguish a set with empty string epsilon and a purely empty list {}. However, after reviewing the test cases in the homework and doing some more thinking, I realized gtrue could be represented by [[]] (a list in which there is an actual element, an embedded empty list, comparable to the empty string epsilon) while gfalse could be represented by [] (an empty list with no elements whatsoever).