For my testing strategy, I thought about the input and output for each method and then accounted for any side effects that may occur. For example, the AddEdge method takes in the parent node, the child node, and the edge label. The inputs should modify the adjacency matrix and update the label in it. This would lead to side effects such as increasing or decreasing the value representing the number of edges. This method ignores the actual implementation of the method but still retains cause and effect. As for black box testing heuristics, for certain methods/constructors, I used boundary values to establish if the base cases were correct. I also split testing into different tests for different cases for all methods, so I did not test everything in one test but rather split them up into separate tests so the results do not confound each other.

After implementation, my tests are still sufficient. I had already planned all my methods out to the dot, so I knew exactly what would be imputed, outputed, and what side effects may happen. I accounted for all of them and did not need to add any new tests.