Blake Boswell

010728180

Project 2: Part 2

The purpose of this assignment was to make sure the programmer had a good enough understanding of avl trees to implement one while trying to use inheritance from the binary search tree to the best of their ability. In my design, I decided would rewrite the binary search tree class from the previous assignment to separate the Node struct and binary search tree code, instead of just using the binary search tree. Trying to do inheritance with my new binary search tree didn’t yield great results because the implementation used Node data types as parameters for the recursive functions, and I could not call the parent class’s method without doing some sort of templating given my implementation of the binary search tree. All the functions I would originally use inheritance for I just decided to use the same logic just with an AVLNode instead of the standard Node. My insert and remove functions both had to implement RL, RR, LR, and LL rotations to keep the tree balanced. This was done by checking the balance factors of the AVLNodes on the return up the tree to make sure their absolute value was not greater than 2. I struggled the most with the remove function. I could get it working with my original test, but I failed to get it to work properly with the empirical study removing the large number of elements I had inserted. I believe my issue was that I was deleting extra nodes somewhere in the code, but I could not find the problem with my logic, and sadly ran out of time to debug. When ran on the University’s Turing machine the insertion of 25000 random integers took only 11.57 seconds, whereas the standard binary search tree took 0 seconds to insert the same 25000 integers. This shows how much overhead goes into keeping the tree balanced after every insertion.

The project was successful in giving me a check on my knowledge of the AVL tree, and even though I didn’t implement inheritance properly, I still learned a lot about derived classes in C++ from my mistakes trying to implement it. If there was anything I could change about my design is that it doesn’t currently utilize inheritance of the binary search tree class from part 1. Something I would do differently next time would be to test the empirical study before assuming my design worked for the small sized test I created to debug my code.