Project 1

Binary Search Tree

CS241-01

Junda Lou

[lou@cpp.edu](mailto:lou@cpp.edu), [loudakkk@hotmail.com](mailto:loudakkk@hotmail.com)

1.30.2017

Project Description

In this project, we are asked to implement a binary search tree, and some operations on the tree such as Insert and Delete. We need to take in a series of numbers from the user, and construct a binary search tree with these values. As long as the tree is constructed, we need to traverse the tree using in, pre and post orders. Users can do insertion and deletion on the tree by simply give the commands that are prompted on the screen. They can also find predecessor and successor of a certain node in the tree.

Project Specification

The most difficult parts in the project are Insert and Delete. They should be consistently functional along with the Binary Search Tree. The data of the tree need to be updated immediately after any change to make the Insert and Delete work properly. User input is another challenging part. User input should never be trusted. Sometimes they only provide the command without the parameter, sometimes they provide something other than integers. The program should be very careful with the input.

Testing Methodology

I test my program step by step when I code the program. If I check it at the end, errors will be gathered together and I have no idea which one is related to which. The first thing I tested was the user input. Since I use the nextLine().split(“ “) method. I checked the String array that was created after that to make sure it takes everything I input, and it became my data array. After recursively constructing the tree, I checked if the traverses I did by hand matches the output. Then I did the Predecessor and Successor parts first. I took in the data array and built a new array in the order of in-order traverse so that I can easily find the result by looking at the previous or the next index in the array. Finally, I implemented Insert and Delete methods. I found that because the data changed in the tree. I need to update the data array as well because it will affect my Predecessor and Successor methods. I thought arrays are passed by reference in Java at first, but I found my testing results got IndexOutOfBounds every time, so I started to doubt that did I really modified the data array. When I test the array thing on another simple test program, I found that arrays are passed by values. I then added functions called updateArray, which takes in my old array as a parameter, and returns a new updated array back to where I need. I keep this pace in the Insert and Delete methods as well. I coded Insert method first and tested it. After it was perfect, I did the Delete method. I did not use the debugger for this program. Everything was debugged by myself.

Lessons Learned

Writing a perfect program is never an easy job. I still have a lot of details to optimize in my code, especially on the user input part, and I know that there should be a better way to implement it to reduce run time complexity. I could have looked at the implementation that Java provided by itself, but I know that implementing it by myself will surely benefit myself in the future. I spend over 12 hours on this project. When I finished testing and saw the correct output. I felt like all this time is worthy.