Maayez Imam

report.pdf

bt_succinct_dec.cc:

For the first programming assignment of homework 3, we were given a file with a line of structures and a line of data associated to the preorder sequence of the nodes. My logic when writing my code was this: I first read in the file and store the first line of structures into its own vector and store the second line of data into another vector. For the output, the preorder sequence is already given, so I just print that back out for the preorder. For the inorder sequence, some manipulation is required with a recursive function.

You have to traverse through the tree in an inorder sequence, printing each node as you go along. That way, I find the inorder sequence of the binary tree and I can print the inorder sequence in order.

bst.h:

For the second programming assignment of homework 3, we had to complete the header file bst.h with the implementation of floor(), ceil(), and kth_smal(). My logic when writing my code was this: for the floor() implementation, I first check if the inputted key is the same as the root key. If so, then I just return that value. However, if the root key is greater than the user inputted key, then I recursively call the floor function, but with the left child of the root key because the floor must be in the left subtree. If the root key is less than the user inputted key, the floor is either in the right subtree or the root has the floor value.

For the ceil() implementation, it's the same as the floor, except the left subtree and right subtree are switched around because we want to find the maximum, rather than the minimum.

For the kth_smal() implementation, I push all the nodes onto a stack and move all the way to the left minimum node. Then one by one, I pop them from the stack and check if that node has a right subtree. If it does, then I push the left subtree of that right subtree onto the stack. Then I pop them again and repeat until I find the kth element.

I chose runtime errors for my error handling because I throw those errors if the program runs through all the if and while statements without finding a ceil, floor, or kth element.