insert\_value - 10pts

This is the insert\_value function, so if the current node is None, then the new node is added to the current node. Return the same node if the value is already there or recursively call itself to the left child node if the new node less than the current node value or to the right child node if the new node value bigger than the current node value until it finds a place to add the new node.

find\_value - 10pts

The findvalue function will return False if it hits a null node, return True if the current node equals to the value needed. Similar to the insert value, it will recursively call itself till the bottom of the tree or till it finds the target

inorder\_walk - 5pts

inorder recur function return the nodes of the tree by the left node first, then the node and then the right node. Inoder\_recur will recursively call itself till the bottom of the tree. So inorder\_walk can just return the result string.

preorder\_walk - 5pts

Similar to the inorder walk, we also have preorder\_recur function to recursively call itself until the bottom of the tree to return the node, the left child and the right child to the preorder\_walk function

postorder\_walk - 5pts

Postorder\_walk also has a postorder\_recur function to recursively call itself to return the left, right child and the node until the bottom of the tree

height - 5pts

the node\_heights function will also recursively call itself to the bottom of the tree. Each time, it will return the max between the left child’s height and the right child’s height +1. Return -1 if the current node is null

find\_min - 10pts

since the min should be the most left of the BST, findmin function will only recursively call itself with the left child of the current node until it is the most left node.

(5 points) - Show output from running the bst.py file.

(5 points) - Show output from running the pytest file.

(15 point) Briefly show that your avg\_heights.py file does the correct tests. Give a short overview.

For average height, I have a function make tree to insert the nodes to the tree and a random sequence to randomly generate values to a list. Then, I use the height method to find the height of each experiment and append it to the Heights list and calculate the average.

(10 points) Show your results table.

(15 points) Give your analysis.

As n increase, the average height increase as well but at a much slower speed. While n go up to 1024, the average height only go up to 21.6. log2 of 1024 is 10, so it’s approximately about 2 times of log2 of N. So it would be closer to O(log2 of N) rather than O(N)