ENGG1340 Final Project- Iqra Abbasi

All the code in this project is written in C++ version 11 in a standard environment. The functions are in-built, and are called when the code is run.

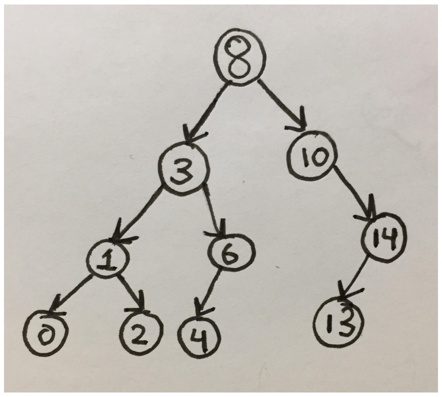
Task 1

I implemented a structure for node and used this to define the functions initialize, search, remove, and insert. The initialize function takes in values and places them appropriately in the search tree before insertion. Insert function uses two reference pointers to navigate the tree for the nodes' correct position. Search() recursively looks for the required key in the tree, and remove() recursively looks for the key to remove the required node, while comparing pointers to rebalance the tree. The search for id 13 is implemented when the code is run.

Task 2

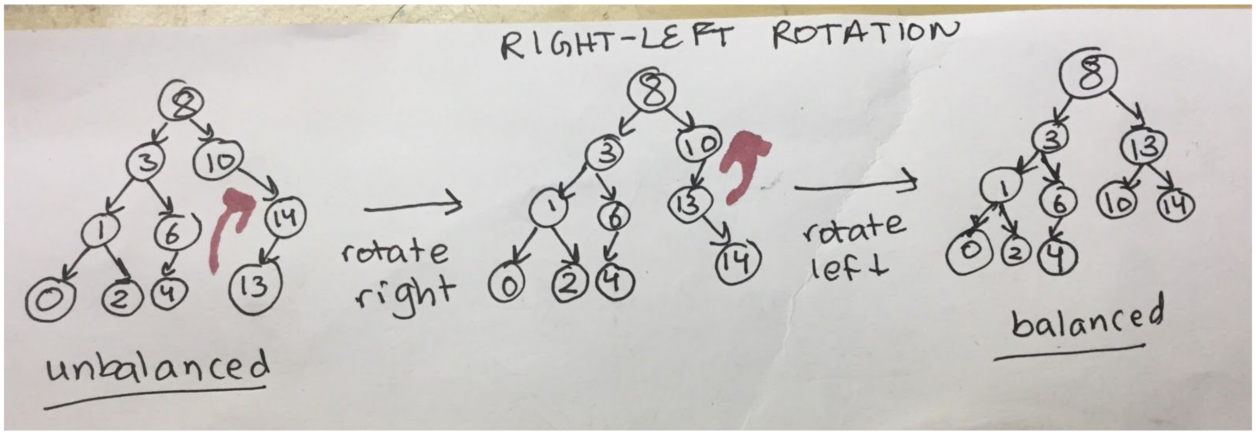
A.1) Since the difference between the heights of the two child subtrees of all the nodes in an AVL tree should not be more than 1, (a) does not follow this rule, as the height of the left subtree is 3 and right subtree is 1. Thus the tree is unbalanced. (b) is balanced because it has nodes with at most a difference of 1 in height, so they are still classified as balanced search trees. And (c) is also unbalanced, as difference of subtrees of node 9 is 2.

A.2)



The binary search tree in task 1 after execution is not balanced, as the difference in heights of subtrees in node 10 is 2.

A.3)



As shown in the diagram above, I implemented the right-left AVL algorithm to balance the binary search tree. The function rebalance() is called at the end of the insert() and remove() functions as these two functions potentially change the structure of the binary search tree.The function also supports single left, single right, double left, and double right rotations for different types of trees. For our instance, the function will call singleRightRotate once, then singleLeftRotate once.

Task 3

I implemented threading for preorder traversal as it non recursively goes through the nodes and converts them to a pre-ordered tree. The program assumes that the previous functions from Task 1 have been executed. Upon initialization, the threading() function is called every time an insert is made, and the thread function calls the Preorder() function inside it. When run, the program displays the age of patient with id 6 through a search function, and the result of the preorder traversal.