Objectives

After performing this lab, students shall be able to:

✓ Learn and practice AVL trees.

Task 1:

- Implement a binary search tree BST and create a function ‘skewed’ that tells if the BST is left skewed or right skewed.
- Create a function ‘balancedBST’ that converts the skewed BST to a balanced BST (AVL).

A suitable main function to test the above functions.

Check your code for the following inputs:

Input1: 3, 2, 1

Input 2: 4,7,15,30

Input 3: 4,3,5,2,6,1,7

Note: You can make any helping functions to complete the above functions.

Task 2:

Construct an AVL tree that does only one rotation that is left-left rotation.

Input: 20, 15, 10, 5, 3, 25

Output: 15, 5, 3, 10, 20, 25

Task 3:

Implement an AVL tree that can handle duplicate values and implement the following functions:

- Create a function ‘isAVL’ which takes a tree as argument and tells whether the tree is an AVL or not.
- A recursive function ‘findmin’ that finds the minimum element using recursion.
- Recursive functions ‘find2ndmin’ that finds the 2 nd minimum element and ‘find3rdmin’ that finds third minimum element in the tree.
- An ‘itprint’ function that prints the preorder traversal using iterator.