**IMPLEMENTATION OF ADVANCED DATA STRUCTURES AND ALGORITHMS**

**Assignment: SP4**

**Group: G19**

**Group Members:**

Jayakarthigayan Sridharan

Sagarikha Srinivasan

SathyaNarayanan Srinivasan

Sahithireddy Andem

1-3 : Create BST, Find its level order traversal, instantiate it using a sorted list – balanced BST, modifying the remove in BST class

a. Level order traversal:

Use a queue and find the level and store the enqueue root's children and dequeue them and store them in a comparable array.

b. Instantiate using a sorted ArrayList:

The input obtained from stream for normal BST is also added to a arraylist, once terminating condition (0/-ve number) is obtained, list is sorted and sent to constructor of BST. Binary search is used for adding elements to BST.

c. Modifying the remove:

A random boolean value is set and if true, it does the remove with maximum element in left subtree and false does the remove with minimum element in right subtree.

Compiling: javac BST.java

RunTime : java BST

<Provide the input elements>

4. Correcting Poorly written code in TreeNode class:

Running time obtained with answers:

|  |
| --- |
| java Tree  Answer: 4613 Time: 1 msec.  Memory: 0 MB / 124 MB. |
| java Tree 1000  Answer: 24745 Time: 3 msec.  Memory: 0 MB / 124 MB. |
| java -Xss256m Tree 10000  Answer: 13783 Time: 8 msec.  Memory: 3 MB / 124 MB. |
| java -Xss256m Tree 100000  Answer: 8811 Time: 69 msec.  Memory: 21 MB / 124 MB. |

Reasons for code being slow:

Recursively and redundantly computing the depth and height of every node.

Changes made in the code:

Made depth and height as fields in TreeNode class.

Made the depth of the tree non-recursive

Made the height of the tree to be run once when the tree is created using recursion by passing the root node, all the subtree's height is computed in between recursion and stored as a field in the TreeNode class.

5. RBT Insertion:

A RBT class is created and elements are added to it using addRBT function. It tries to solves the three cases of Repair.

Note: When parent is red and child is red but parent does not have a sibling, it goes to case2Repair or case3Repair depending upon where child is located.

Compile: javac RBT.java

RunTime: java RBT

<Provide the inputs>

6. AVL Tree Verification:

AVL Tree is created and then using level order traversal, height of every node is obtained,the elements in AVL tree are also obtained. If element is null or if node.height>1 or if monotonocity is not maintained in elements obtained in Inorder Traversal, AVL Verification fails.

Compile: javac AVL.java

RunTime: java AVL