|  |  |
| --- | --- |
| **Project 1** | **AVL Tree** |
| **Due: March 3** |  |

Your work should be readable as well as correct.

Always Verify Time!

Part I: Write code to build an AVL tree by inserting Book nodes and detecting imbalance. If imbalance is true, then call the proper rotation function provided in the lecture slides to fix the imbalance condition.

1. Must read AVLNode data from a text file
   * Create a text file containing Book objects
   * ISBN Number <space> Title <space> Author’s last name
2. Create a Book Object; and an AVL node object to be inserted into the AVL tree
3. At each insert, detect imbalance and fix the AVL tree
4. Report each imbalance detection and the node where it occurred; and output the message:

Output:

Imbalance occurred at inserting ISBN 12345; fixed in LeftRight Rotation

Imbalance occurred at inserting ISBN 87654; fixed in Left Rotation

Imbalance occurred at inserting ISBN 974321; fixed in RightLeft Rotation

class AVLNode {

String key; (ISBN number)

Book value; //create a class representing a book with minimum attributes

int height;

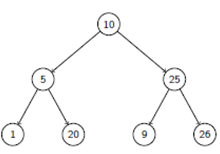
AVLNode leftPtr;

AVLNode rightPtr;

}

You must verify the AVL balance condition at each insert and detect and fix imbalance, output result of each insertion. A null node is considered AVL property of height -1.

Programming Languages: Java, C++ or C#

Part II: Create a random binary tree and verify BST order property and AVL balance condition. Report the problems. You don’t need to fix anything. Also, do not hard code the tree inside the code to force the creation of order.

Submission  
 zip all the documents below  
 - All Source Code (.java, .cpp files)  
 - make file for .cpp   
 - report of your output from part 1 and 2  
 - textfile your used  
 - any other document you wish to submit

Note: ISBN number should be String like: 978-0262033848

TextFile:

978-0262033848 Introduction\_to\_Algorithms Cormen