|  |  |
| --- | --- |
| **Project 2** | **AVL Tree** |
| **Due March 28 at 11:30 pm** |  |

Your work should be readable as well as correct.

Always Verify Time!

**Part A:** 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. Source code for a basic AVL tree is provided

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 bookObject; // Title <space> Author’s last name, any other attribute

int height;

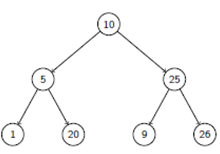
AVLNode leftPtr;

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#   
Source files are attached to the Project link on eLearning

**Part B:** Create a random binary tree (not binary search tree) and verify BST order property and AVL balance condition.

Hint: Should you proceed to validate AVL if the binary tree did not produce BST?

Report the problems. You don’t need to fix anything. Also, do not hard code the tree inside the code.

Potential report:

1. The random binary tree did not create as a binary search tree
2. The random binary tree did create a binary search tree but not an AVL tree
3. The random binary tree did create an AVL tree

Use the following rubric as a guideline; subject to change based on class discussions:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **Project 2 - AVL Tree Implementation** | Points |
| **Part A** | No |  |  |
|  | 1 | Create a text file with at least 25 ISBN numbers and Book titles | 5 |
|  | 2 | Write a function to read the Book records from the text file | 10 |
|  | 3 | For each record read in #2, create a Book object; and use it to create the AVLNode with ISBN number as the key | 10 |
|  | 4 | Modify the AVLTree class file provided with the project to work with AVLNode | 50 |
|  | 5 | Insert all AVLNodes into the AVL tree using the AVLTree class | 25 |
|  | 6 | Output all rotations as depicted in the project document | 25 |
| **Part B** |  |  |  |
|  | 7 | Create at least 25 random ISBN Keys (or use from Part A) | 10 |
|  | 8 | Insert all random ISBN keys into a Binary Tree (class provided) | 15 |
|  | 9 | Write a function to determine if the Binary Tree from #8 is a BST or not a BST | 25 |
|  | 10 | Use the modified AVLTree class to determine if BST is an AVL tree | 25 |
|  |  |  | 200 |

Notes:

1. The word “ISBN” will not be part of the key “ISBN number” in the class AVLNode; and it is a String
2. If Book titles have spaces in the name, then use any delimiter like underscore to replace the spaces
3. Post questions related to this project in the discussion board on eLearning