

# CSE 332 Data Abstractions, Winter 2015

## Homework 3

Due: **Wednesday, Jan 28, 2015** at 23:00 (11:00pm) via the catalyst drop box. You should refer to the written homework guidelines on the course website for a reminder about what is acceptable pseudocode. You will notice that this assignment has FOUR Fantastic questions!!

### Submission instructions

Submit an electronic copy to the catalyst dropbox as a PDF file. You can either do the assignment on an electronic word processor (and convert to PDF) or do it on physical paper and scan it (or take a high res photo) and upload a single PDF of the file. It will be much easier to grade if every question starts on a separate page. Don't forget to put your name on the top of the first page. You may want to begin with Problem 3, although it should start on page 3 of your writeup. You can take as many pages as you need for problem 4.

### Problem 1: Verifying AVL Trees (1 page )

Give pseudocode for a linear-time algorithm that verifies that a tree is a proper AVL tree. Assume every node has fields: key, data, height, left, and right and that keys can be compared with  $<$ ,  $=$ , and  $>$ . The algorithm should verify all of the following:

- The tree is a binary search tree.
- The height information stored in each node is correct.
- Every node is balanced.

Your code should throw an exception of your choice if one of these properties is violated. If the tree is a valid AVL tree, no exception should be thrown.

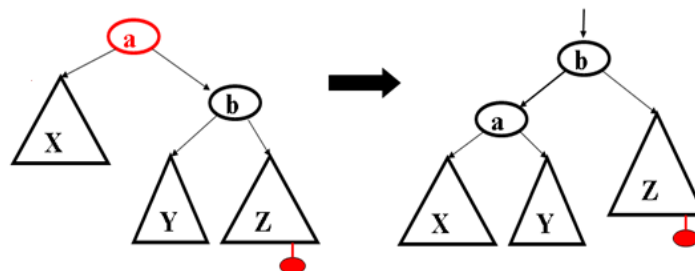
### Problem 2: AVL Rotation Scenarios ( 1 page )

The diagram below shows the general case for performing a case 4 (right-right) rotation; an insertion (the red dot) has taken place in subtree Z and an imbalance has been detected at node 'a'.

- Each triangle is assumed to be a subtree that is AVL balanced.
- After the rotation is applied, the resulting entire tree will be AVL balanced.
- Assume subtree Z has height  $h$  before the insertion (i.e. before the red dot is added), and height  $h+1$  after the insertion.

What you need to do for this problem is argue that subtrees X & Y must have height  $h$ ; that is, show that it is not possible for either to have height  $h-1$  or  $h+1$ .

- State what the problems are with the two specific height scenarios of  $h-1$  and  $h+1$  for both X and Y. While you are describing why X cannot be height  $h-1$  or  $h+1$ , *don't assume anything about the height of Y* and vice versa.
- A sentence or two for each of the four scenarios is fine. A proof is not required.



### Problem 3: AVL Insertion ( 2 pages )

Show the result of inserting 43, 8, 5, 10, 4, 7, 32, 2, 1 and 3 in that order into an initially empty AVL tree. Show the tree after each insertion, clearly labeling which tree is which.

### Problem 4: B-Tree Insertion

Show the result of inserting 28, 12, 17, 4, 31, 34, 8, 14 & 16 in that order into an initially empty B tree with  $M = 3$  and  $L = 2$ . (Recall the text, lecture, and this problem call a B tree what many call a B+ tree.) Show the tree after each insertion, clearly labeling which tree is which. In an actual implementation, there is flexibility in how insertion overflow is handled. However, in this problem, follow these three guidelines:

- Always use splitting (not adoption).
- Split leaf nodes by keeping the smallest 2 elements in the original node and putting the 1 largest element in the new node.
- Split internal nodes by keeping the 2 children with the smaller values attached to the original node and attach the 2 children with the larger values to the new node.

Have fun!