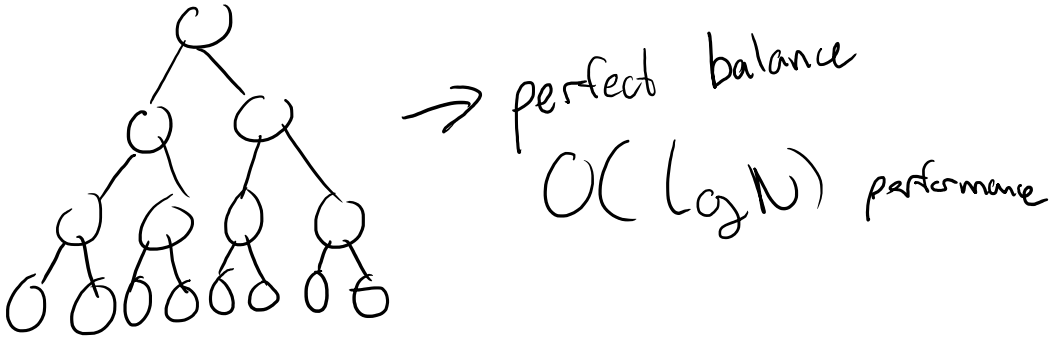
2018-02-22 AVL Trees

Thursday, February 22, 2018 9:05 AM

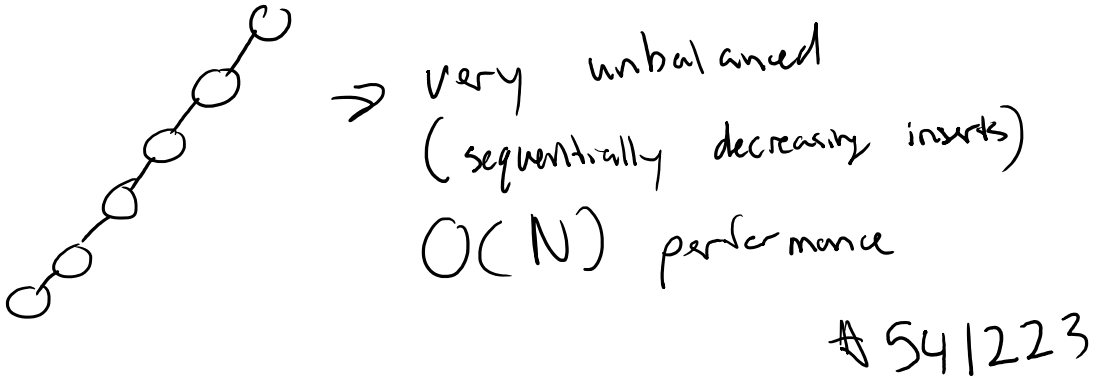
Visualization Resources

- BST Visualization: http://www.cs.usfca.edu/~galles/visualization/BST.html
- AVL Visualization: http://www.cs.usfca.edu/~galles/visualization/AVLtree.html
- Visualization homepage: http://www.cs.usfca.edu/~galles/visualization/Algorithms.html

- In an ideal world, a BST looks like this:



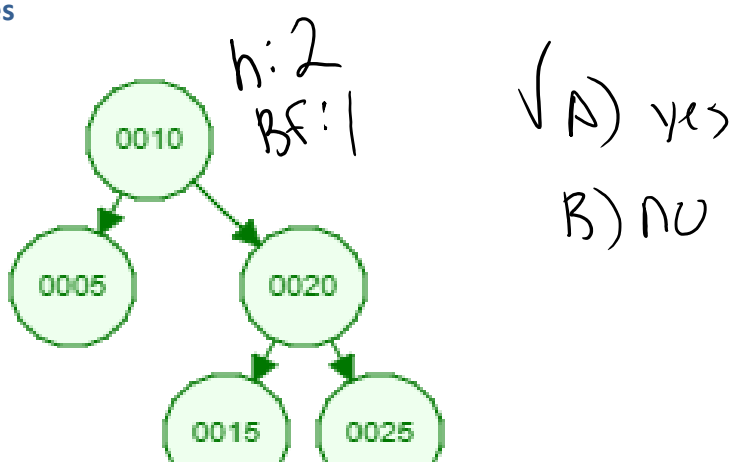
- However, it is very possible to end up with the following tree:

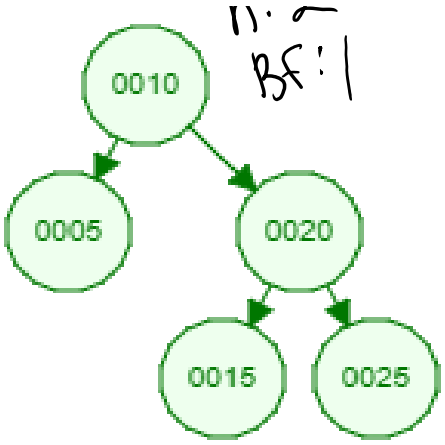


AVL Tree Properties

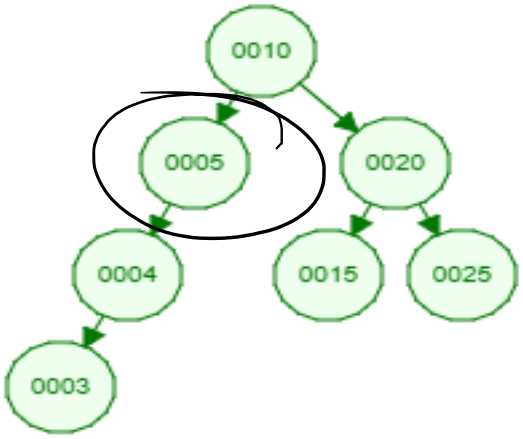
- An AVL tree is a BST that has one additional rule:
  - For each node, the absolute value of the difference between the height of right subtree and left subtree cannot be greater than one

Examples

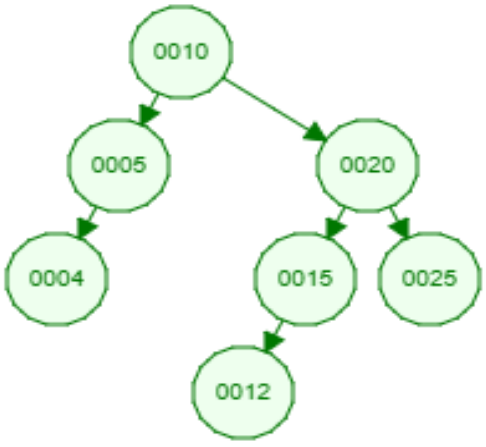




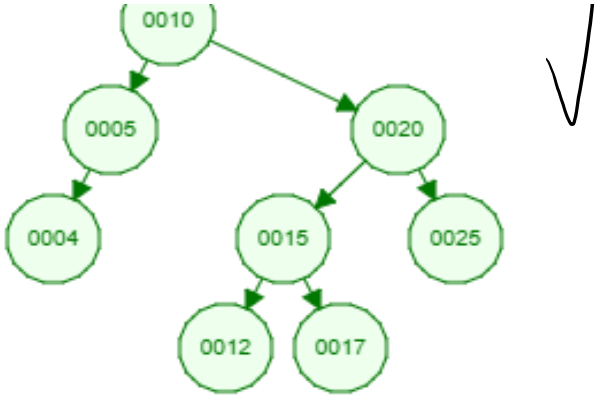
- This is an AVL tree?



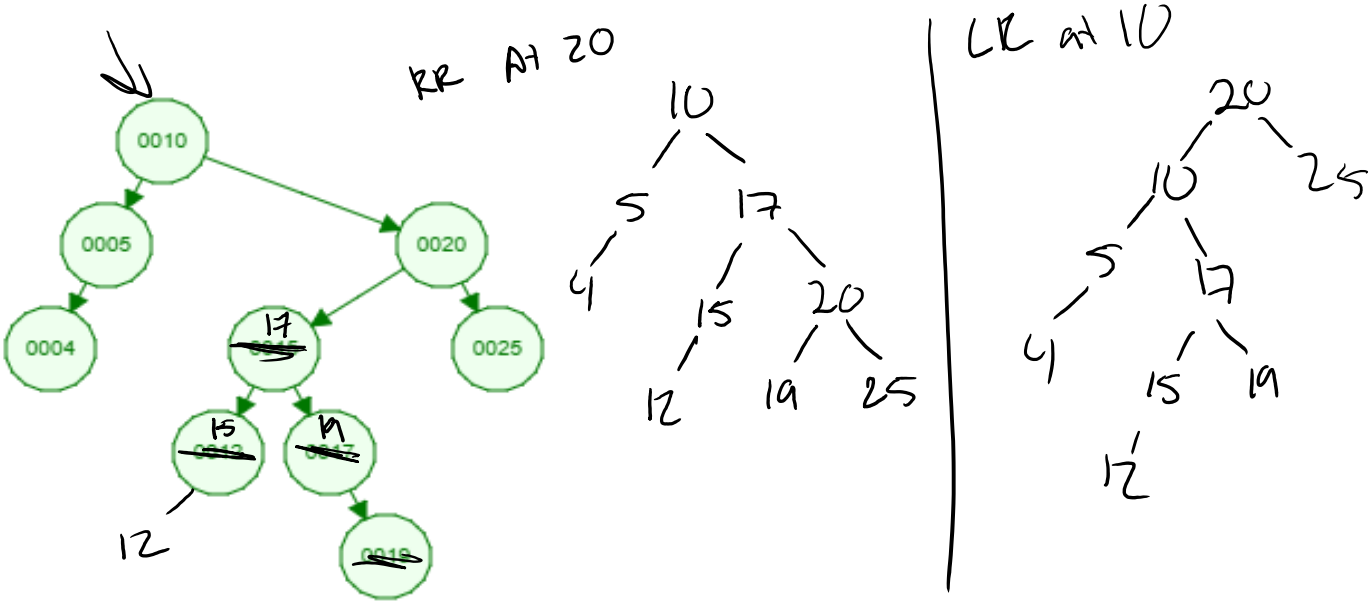
- This is an AVL tree?



- Is this an AVL tree?

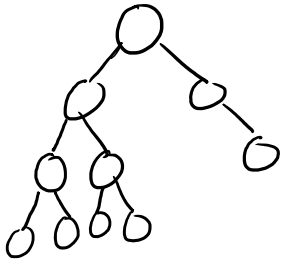


- Is this an AVL tree?

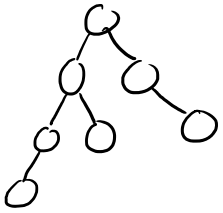


- Is this an AVL tree?

Draw the most unbalanced tree of height 3 that you can that is AVL



Draw the most unbalanced height 3 AVL tree using the fewest nodes possible



- On every insert and removal from a BST, we apply the AVL algorithm whenever

the tree becomes unbalanced (non-AVL compliant)

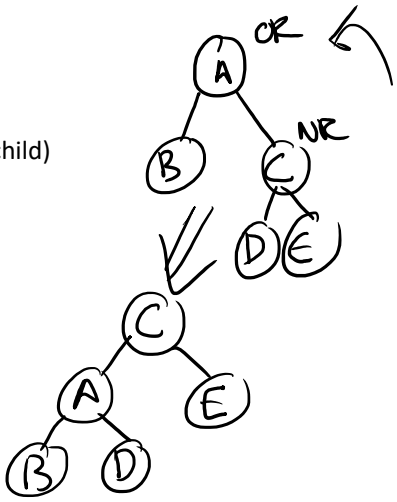
Converting a non-AVL tree into an AVL tree using simple (i.e. single) rotations

- Balance factor = right height - left height

Left (counter-clockwise) Rotation

- Occur when the balance factor is greater than 1

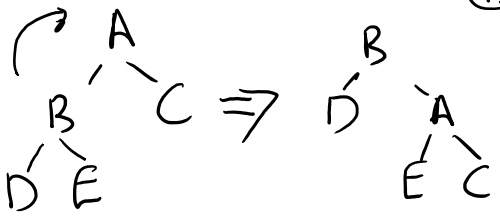
1. At the node whose left and right height differ by more than 1:
2. Let OriginalRoot = node identified in step #1
3. Let NewRoot = OriginalRoot ->rightChild
4. OriginalRoot->rightChild = NewRoot->leftChild (reassign OR's right child)
5. NewRoot->leftChild = OriginalRoot



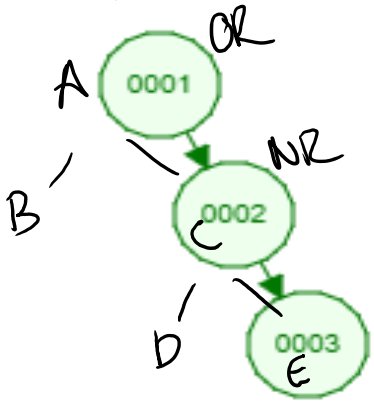
Right (clockwise) Rotation

- Occur when the balance factor is less than -1

1. At the node whose left and right height differ by less than -1:
2. Let OriginalRoot = node identified in step #1
3. Let NewRoot = OriginalRoot->leftChild
4. OriginalRoot->LeftChild = NewRoot->rightChild
5. NewRoot->rightChild = OriginalRoot

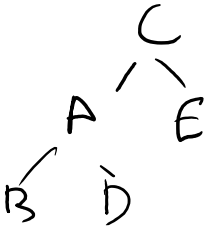


Rotation Examples:

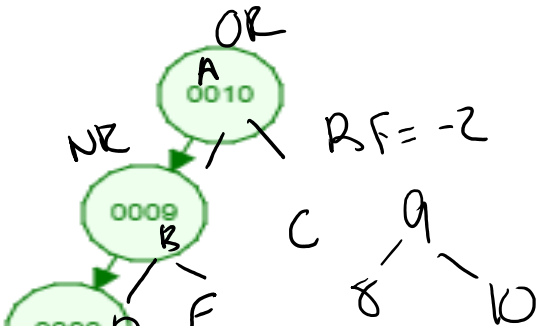


right heavy (BF:2).

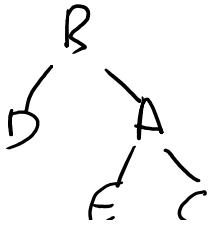
We need to pull nodes from right to left (counterclockwise direction)

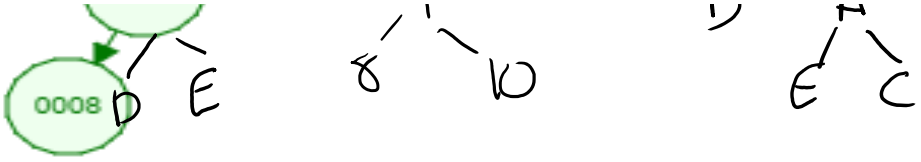


- What is the balance factor?
- What is the type of rotation needed?
- What is the final result?

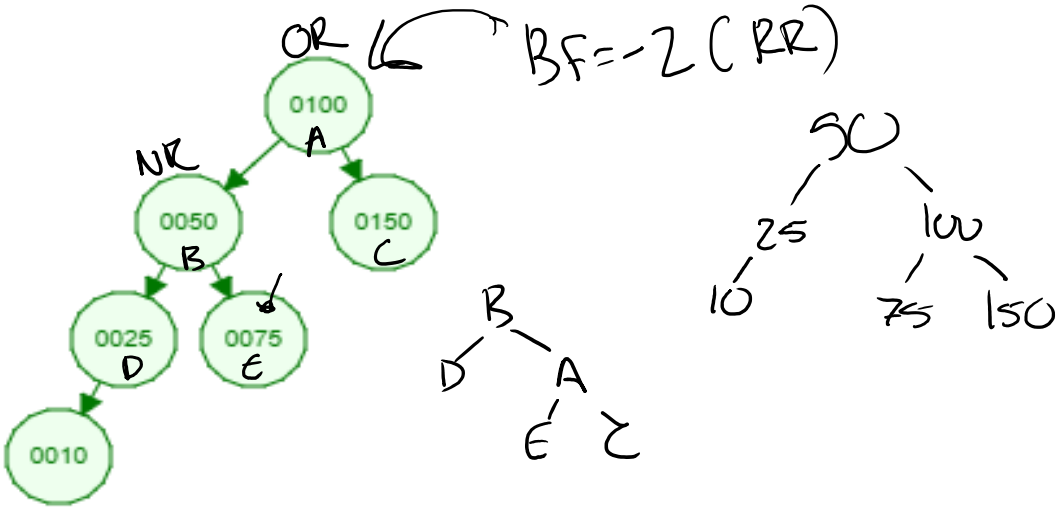


BF = -2

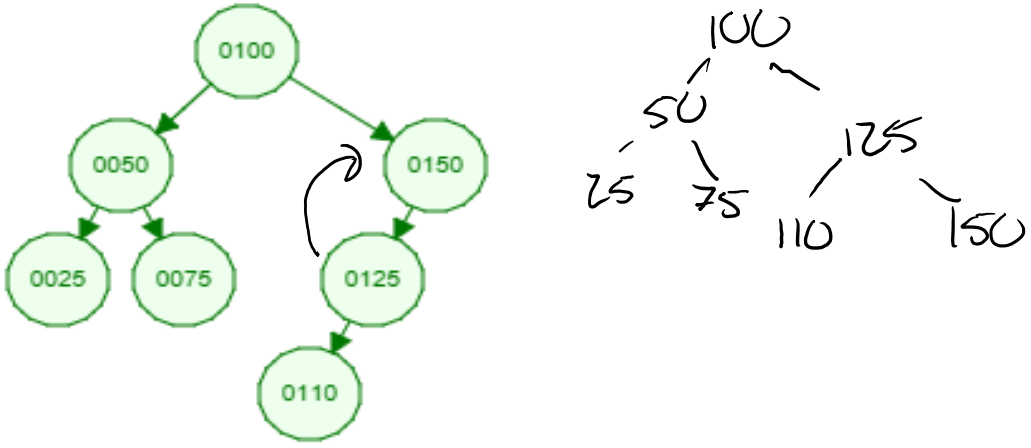




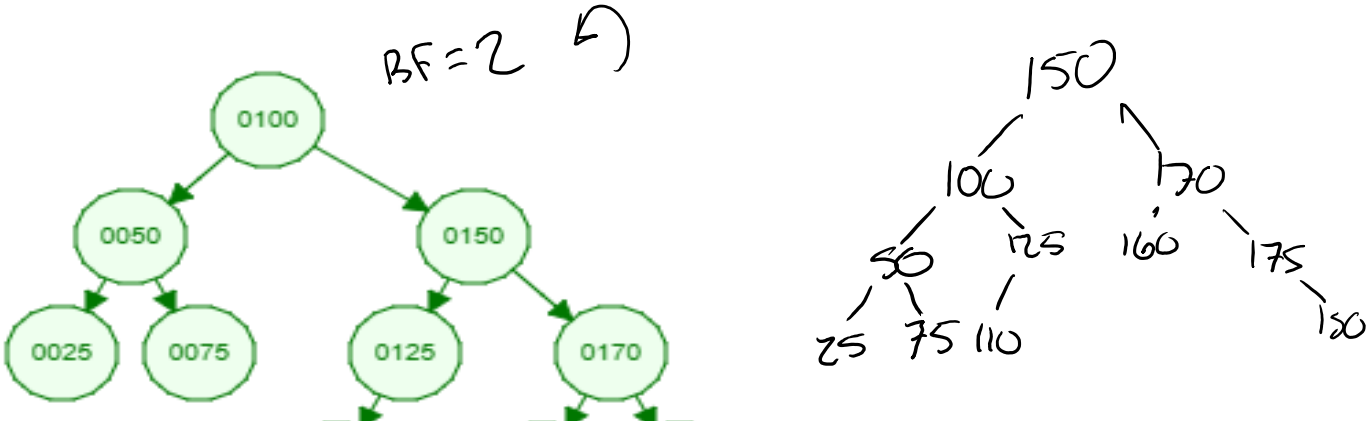
- What is the type of rotation needed?
- What is the final result?

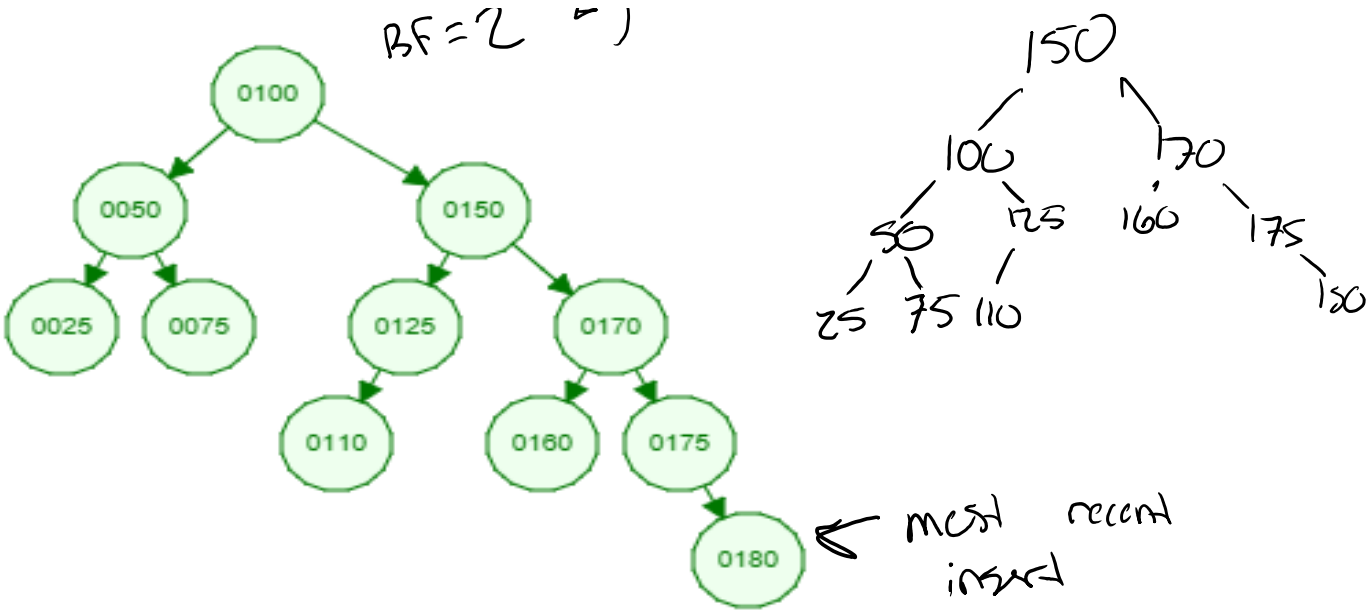


- What is the type of rotation needed?
- What is the final result?



- What is the type of rotation needed?
- What is the final result?





- What is the type of rotation needed?
- What is the final result?

(time permitting) Add values 1-10, remove 10-1