AVL Trees
AVL Trees
Insert 50, 100, 150, 200 into a Binary Search Tree
AVL Trees

• A “balanced” binary search tree
• AVL trees are balanced in the sense that for any node the height of the left subtree and the height of the right subtree differ by at most 1.
• To keep track of the balance a balance value (tall left, equal, tall right) can be attached to each node or each node can maintain its height.
AVL Trees

• The find functions for AVL trees work exactly like the find functions for binary search trees
• Insert and remove follow the same procedure as that used by binary search trees except after a node is inserted or deleted the tree might have to be rebalanced.
• Rebalancing takes place along the search path
• Inserts require rebalancing at most one node in the search path
• Removes could require rebalancing at every node in the search path
AVL Trees
Insert 50, 100, 150, 200 into a AVL Tree
AVL Trees

Insert 50, 100, 150, 200 into a AVL Tree
AVL Trees

Insert 50, 100, 150, 200 into a AVL Tree
AVL Trees
Insert 50, 100, 150, 200 into a AVL Tree
AVL Trees
Insert 250

Single Rotation
AVL Trees
Insert 125

Double Rotation

Before:

100
  /   
50   200
  /      /   
150  250 125

After:

150
  /   
100  200
  /      /   
50  125  250
AVL Single Rotation
Insertion Done to the Left of S
\[ \text{height}(X) = \text{height}(C) + 2 \]
\[ \text{height}(A) = \text{height}(B) + 1 \]
AVL Single Rotation
Insertion Done to the Right of S
height(C) + 2 == height(X)
height(B) + 1 == height(A)
AVL Tree Double Rotation
Insertion Done to the Left of S
height(X) == height(D) + 2
height(A) + 1 == height(Y)
AVL Double Rotation
Insertion Done to the Right of S
height(D) + 2 == height(X)
height(Y) == height(A) + 1
AVL Trees
Review Question

• Insert the following values into an AVL tree

  100, 50, 5, 75, 2, 80, 12, 90, 200, 150, 300, 125, 110, 95, 130, 3, 25, 97