```
More BSTs & AVL Trees
                       bstDelete
if (key not found)
  return
else if (either subtree is empty)
 delete the node replacing the parents link with the
 ptr to the nonempty subtree or NULL if both
  subtrees are empty
else
 Traverse the left subtree of the node to be deleted
   such that you find the rightmost node (Rnode) in the left
   subtree
 Move the contents of Rnode to the node to be deleted
  Set Rnode's parent pointer to point to the left subtree
   of Rnode
```

Free the unused node CS300 Data Structures (Fall 2014)

bstDelete

• Create a BST from the following keys: 10, 5, 15, 2, 8, 12, 7, 16, 14

• Assume that you always start with the above tree, how would each of the following keys be deleted?

- ▶ 10
- > 15
- > 5

AVL Trees Adelson-Velskii & Landis

- **Defn**: A binary tree is a height-balanced p-tree if for each node in the binary tree, the difference in the height of the left and right subtrees is at most p.
- **Defn**: An AVL (Adelson-Velskii, Landis) tree is a binary search height-balanced 1- tree.
- **Defn**: The balance factor of a node, BF(node), in a binary tree is the difference of the left and right subtrees, hL hR.
- For any node in an AVL tree, the balance factor is either -1, 0, or 1.

AVL Trees

- After inserting a new value into an AVL tree, if any node has a BF other than -1, 0, or 1, the AVL tree must be rebalanced.
- The AVL tree is rebalanced at the closest ancestor, of the inserted node, that has a BF of -2 or +2. We will call the closest ancestor with a BF of +2 or -2 of the inserted node the pivot node, P.
- Four basic rotations are possible where two are single rotations and two are double rotations.

AVL Trees



Green Circle: Node with BF of -1, 0, or 1

Blue border:

Newly added node

Greyed Out Circle: Node not in the rotation trio. (Color based on original parent. Attachments may change)

> Blue/purple circle and arrow: Indicates the nodes being rotated. Arrow indicates direction. Rotation 1 is blue, rotation 2 is purple.

Black line: Tree connections

Blue line: Path from pivot to added nodes.

Blue letters: LR Direction of path to added node.

> Brackets: Depth of subtrees. Color indicates validity of resultant BF.

Calculation: 2-1=1Balance Factor calculation. 3-1=2

The following diagrams were made by Alex Shinsel

AVL Trees LL Rotation





AVL Trees LL Rotation



AVL Trees RR Rotation



AVL Trees RR Rotation



AVL Trees LR Rotation



AVL Trees LR Rotation



AVL Trees RL Rotation





AVL Trees RL Rotation



• Consider

m

n

f

Problems

Q1: Is the tree an AVL tree? Why or why not?

Q2: Does the tree need any kind of rebalancing? If so, rebalance the tree.

P1: Insert z into the tree.

Q3: Does the resulting tree need rebalancing? Why or why not? If so, rebalance the tree.

P2: Insert a into the tree.

Q4: Does the resulting tree need rebalancing? Why or why not? If so, rebalance the tree.

P3: Starting over, insert **j and g** into the tree. Rebalance when necessary.

P4: Starting over, insert **j and a** into the tree. Rebalance when necessary.CS300 Data Structures (Fall 2014)

Problems

• Insert the following months into an AVL tree lexicographically: Mar, May, Nov, Aug, Apr, Jan, Dec, Jul, Feb, Jun, Oct, Sep

•If a rebalance is needed, show the proper rebalance notation for the type of rebalance applied to the AVL tree