CSE 373: Data Structures and Algorithms

More on AVL Trees

Autumn 2018

Shrirang (Shri) Mare
shri@cs.washington.edu

Thanks to Kasey Champion, Ben Jones, Adam Blank, Michael Lee, Evan McCarty, Robbie Weber, Whitaker Brand, Zora Fung, Stuart Reges, Justin Hsia, Ruth Anderson, and many others for sample slides and materials ...

Quick (Anonymous) Feedback

Go to this URL https://tinyurl.com/373-feedback

1. In general, pace of class:
   - 1 too fast
   - 2 kind of fast
   - 3 just right
   - 4 kind of slow
   - 5 Too slow
2. Please **Keep** doing this
3. Please **Quit** doing this
4. Please **Start** doing this

Outline

So far

- BSTs are efficient for insert and remove operations, but in worst case they take linear time – O(n).
- If we keep BSTs 'balanced', we can avoid the worst case.
- We can easily maintain a balanced BSTs using the AVL balance condition

Today

- Maintaining AVL balance condition
- (Maybe) Intro to Hash tables

AVL trees: Balanced BSTs

AVL Trees must satisfy the following properties:

- binary trees: every node must have between 0 and 2 children
- binary search tree (BST property): for every node, all keys in the left subtree must be smaller and all keys in the right subtree must be larger than the root node
- Balanced (AVL property): for every node, there can be no more than a difference of 1 in the height of the left subtree from the right. Math.abs(height(left subtree) – height(right subtree)) ≤ 1

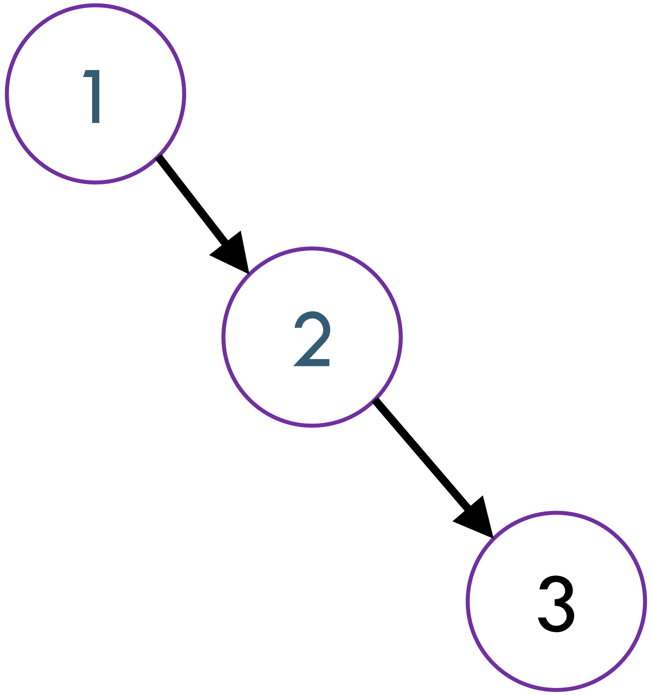
AVL stands for **A**delson-**V**elsky and **L**andis (the inventors of the data structure)

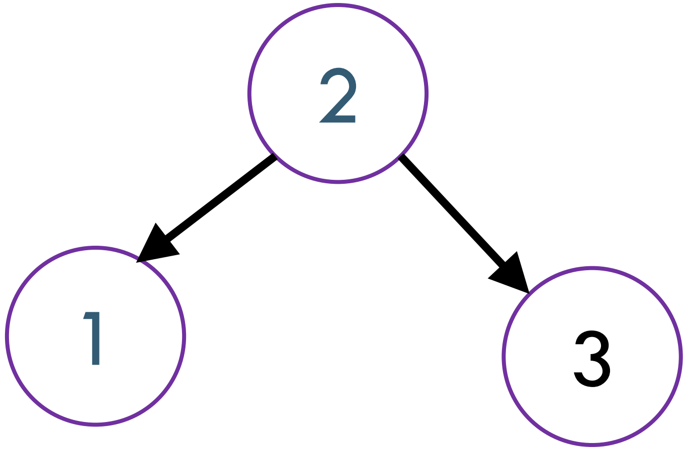
The AVL property:

1. ensures depth is always $O(\log n)$ – Yes!
2. is easy to maintain – Yes! (using single and double rotations)

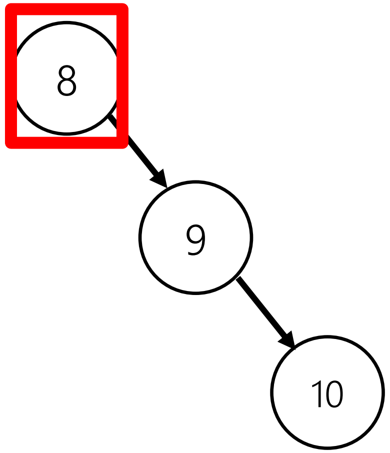
Insertion

What happens if when we do an insert(3), we break the AVL condition?

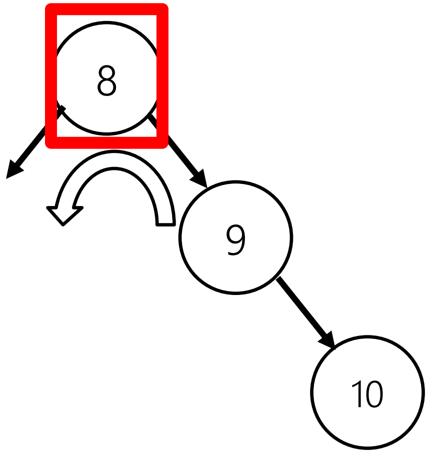




AVL Example: 8,9,10



AVL Example: 8,9,10



Worksheet (Q1)

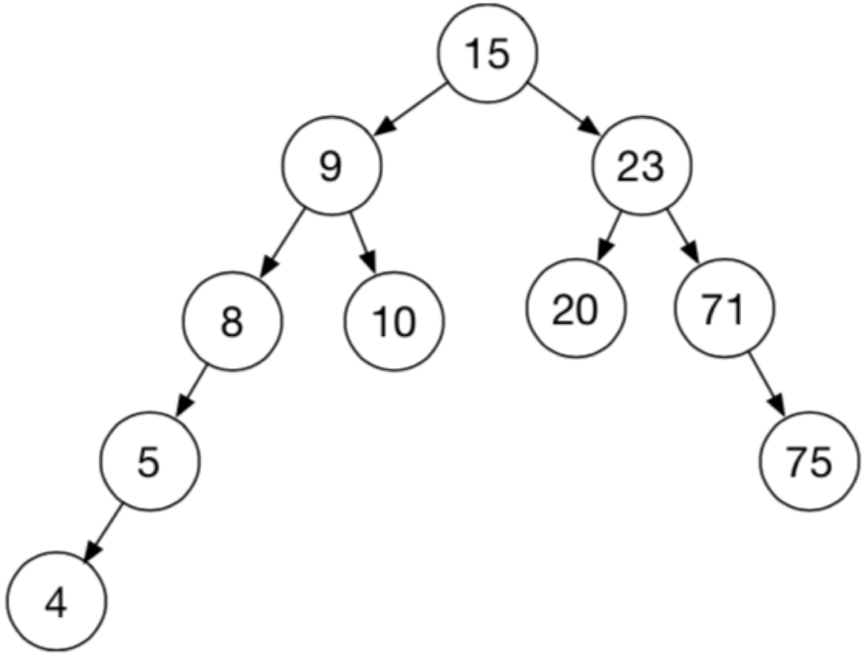
(Q1) Answer the following questions for the corresponding tree (on the right):

A. Is this a BST? (Y/N):

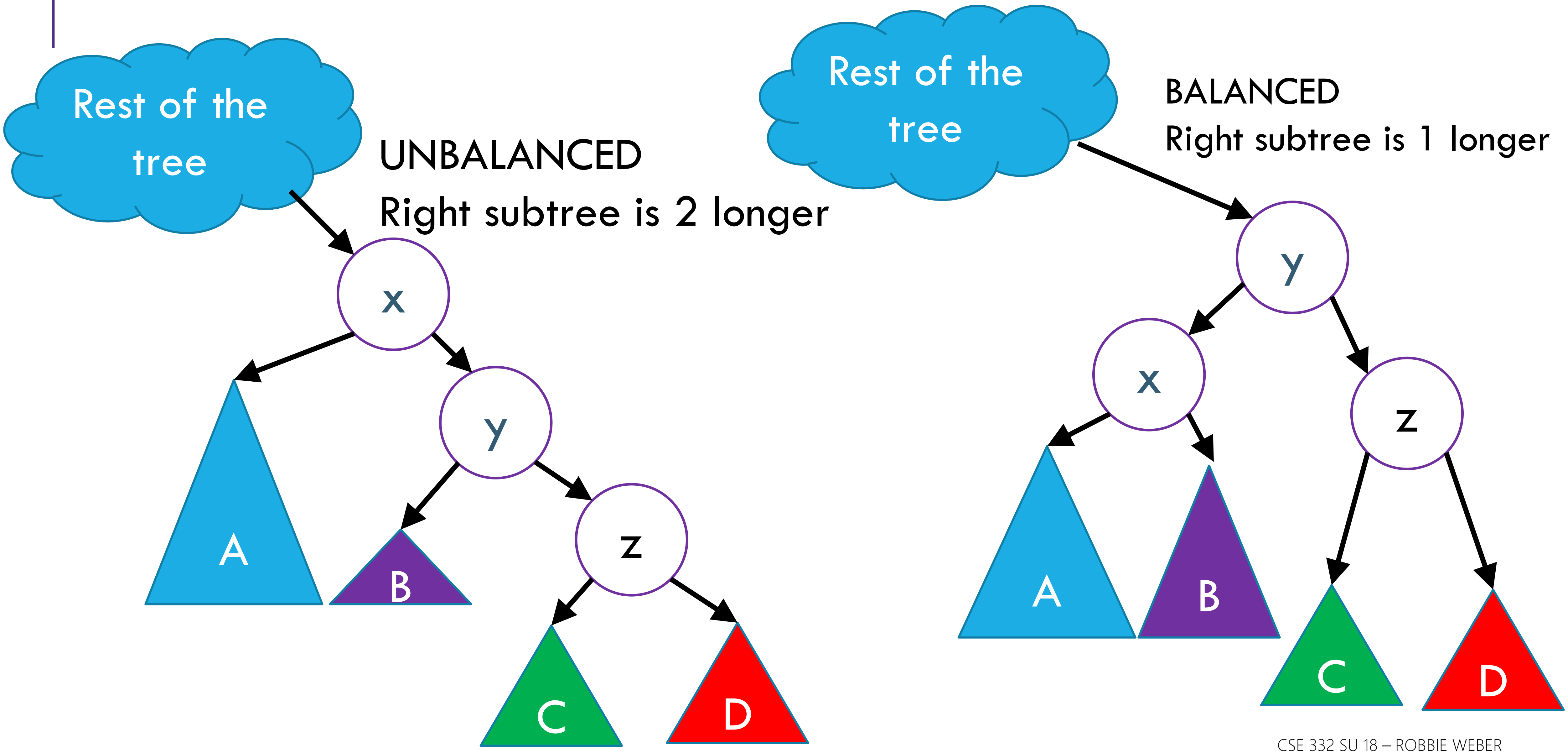
B. Highlight the AVL unbalanced node:

C. Is this a ‘line’ or ‘kink’ case?

D. To make this AVL balanced, how many rotations do you need? (single/double)

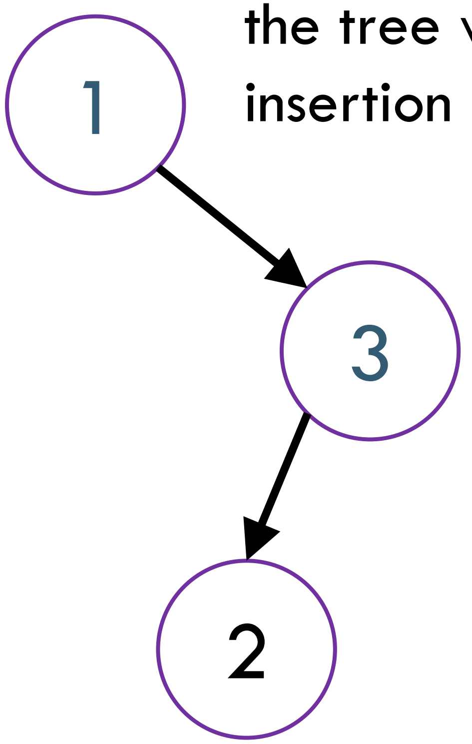


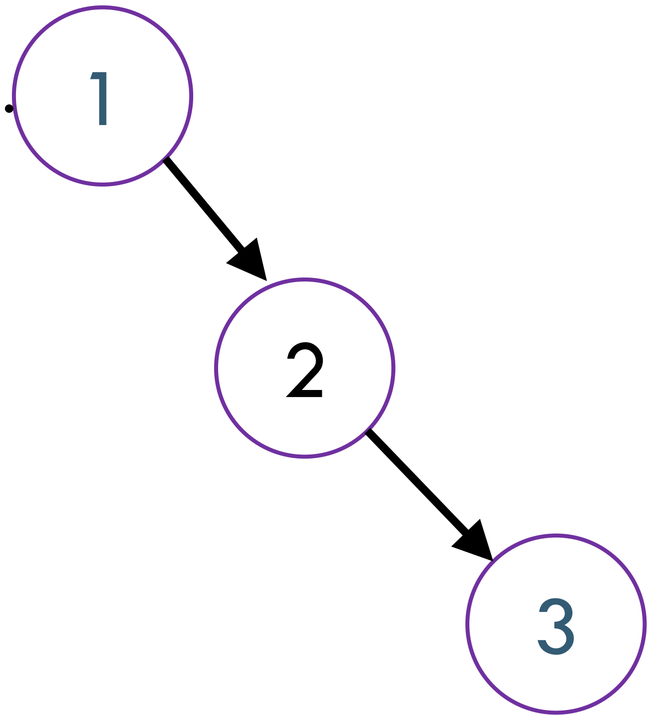
Left Rotation

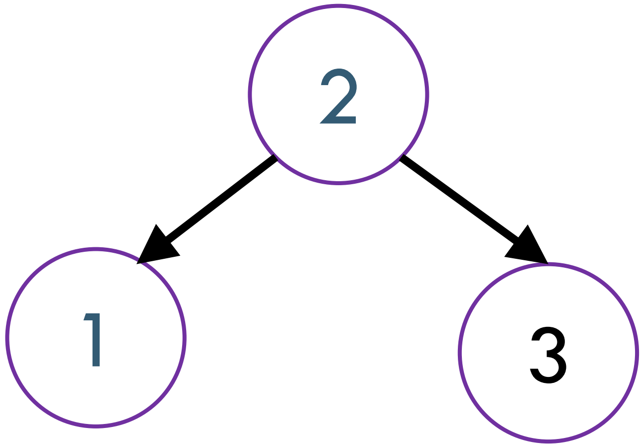


It Gets More Complicated

There's a "kink" in the tree where the insertion happened.





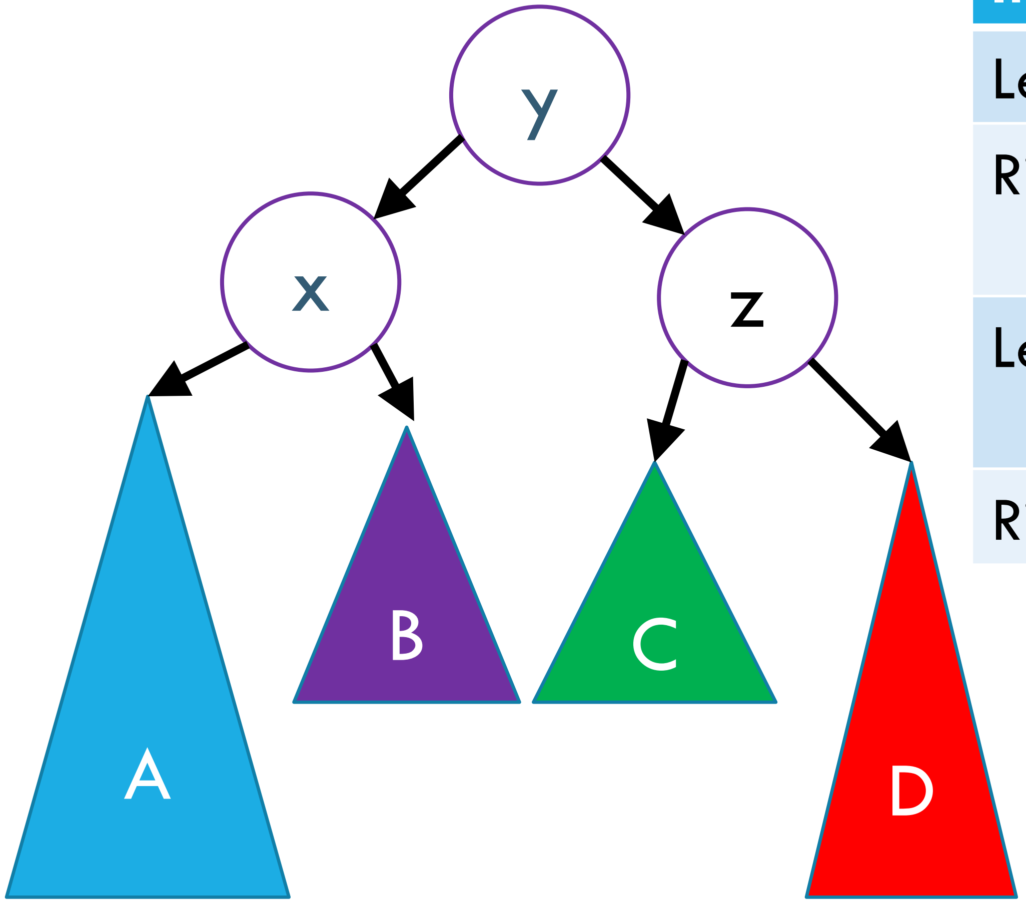


Can't do a left rotation

Do a "right" rotation around 3 first.

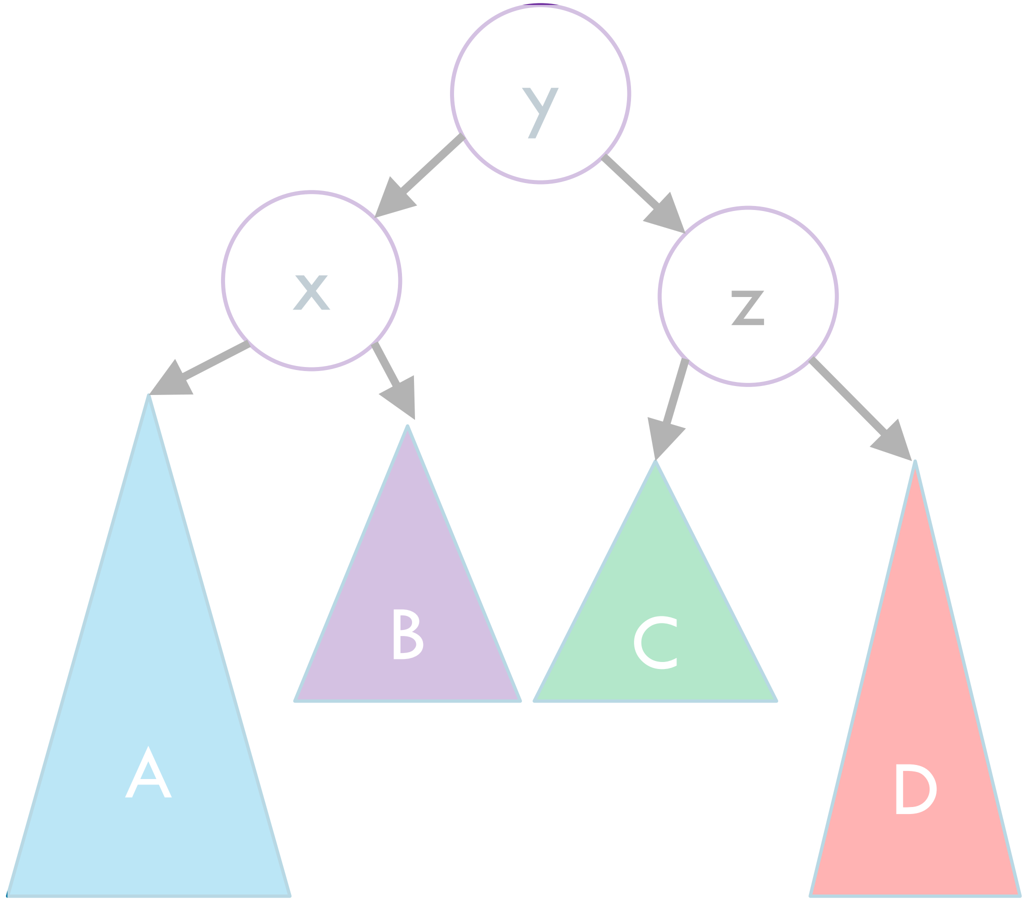
Now do a left rotation.

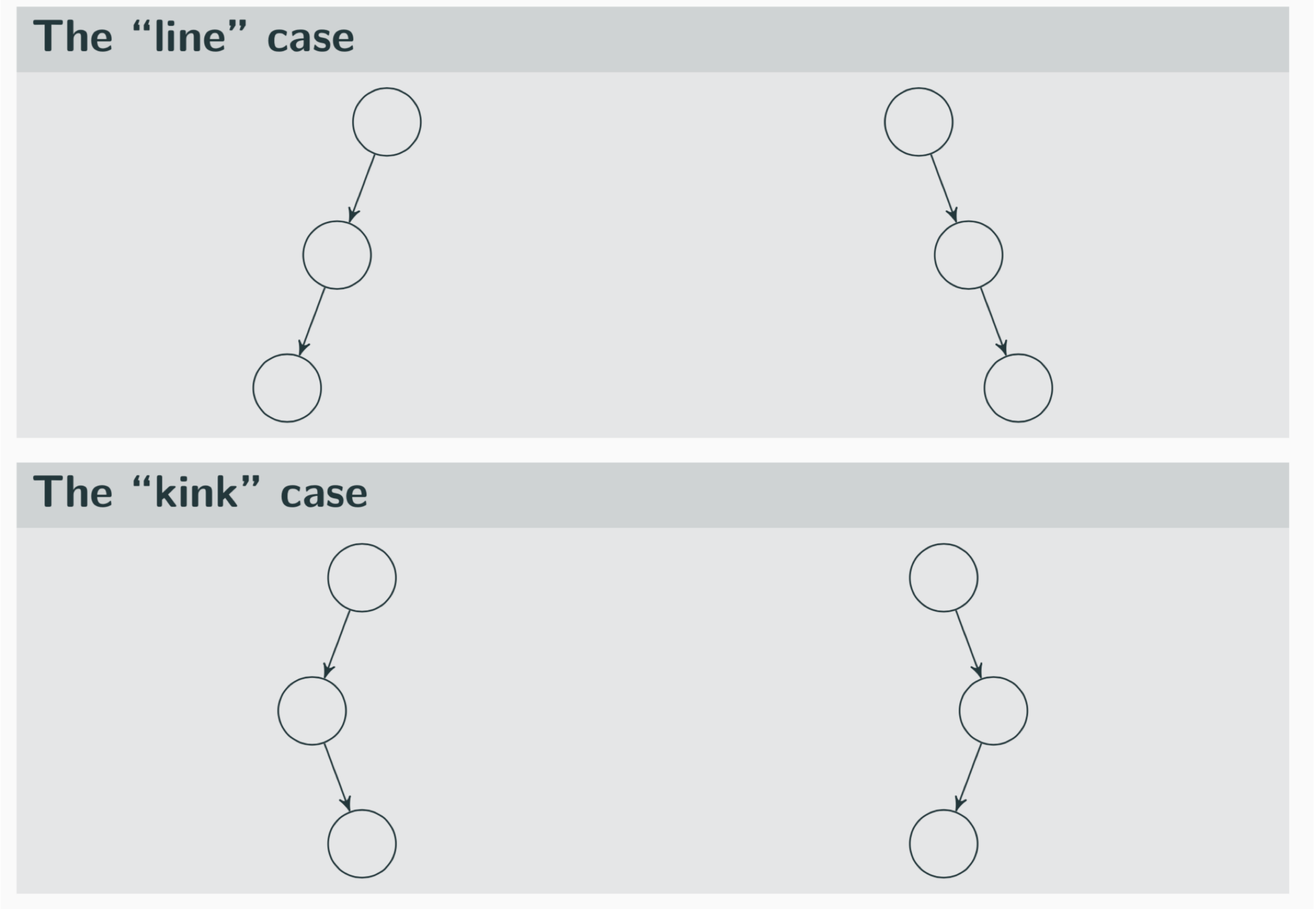
Four cases to consider



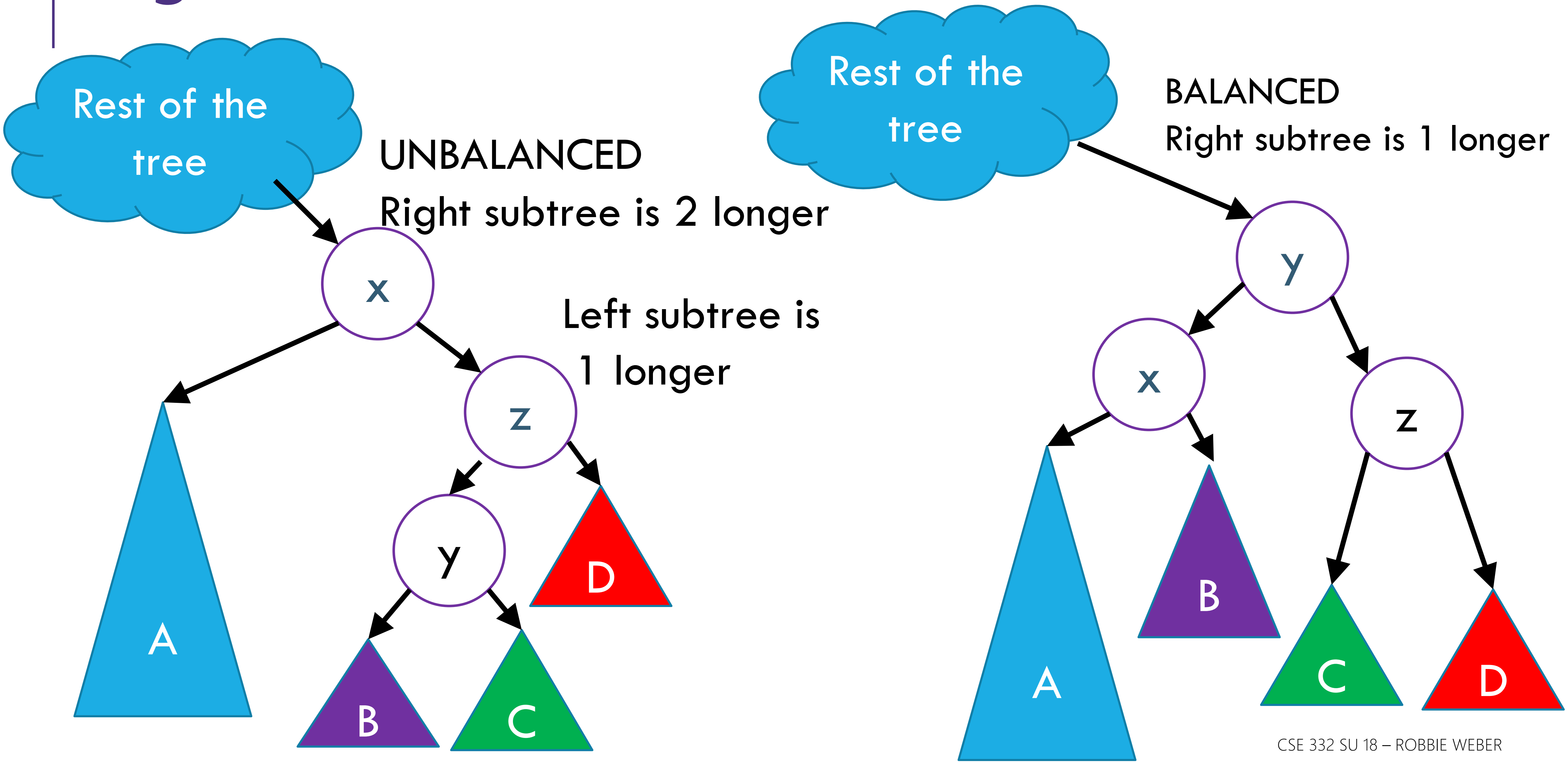
Insert location	Solution
Left subtree of left child of y	Single right rotation
Right subtree of left child of y	Double (left-right) rotation
Left subtree of right child of y	Double (right-left) rotation
Right subtree of right child of y	Single left rotation

Four cases to consider

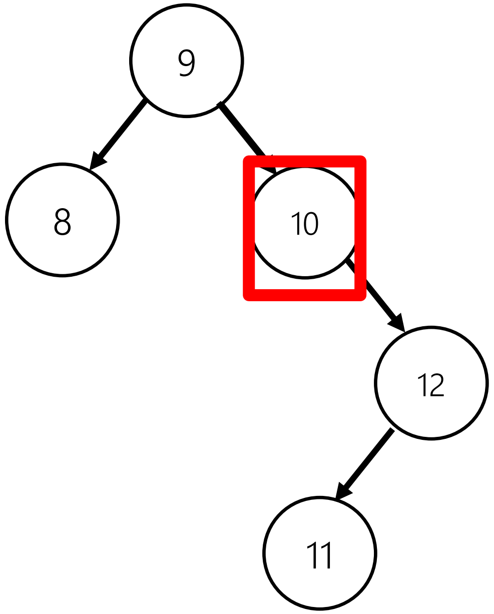




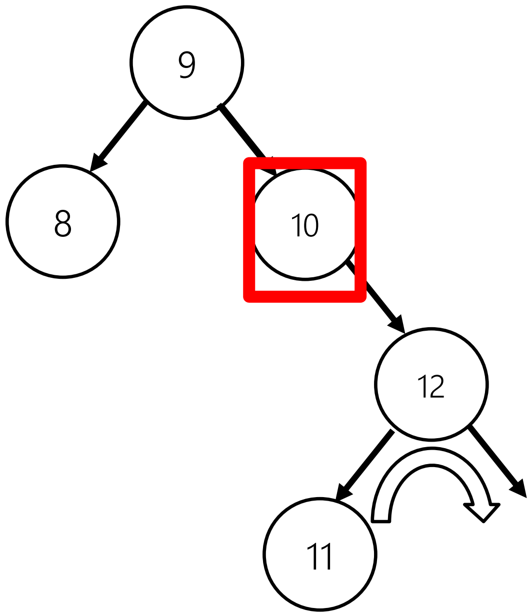
Right Left Rotation



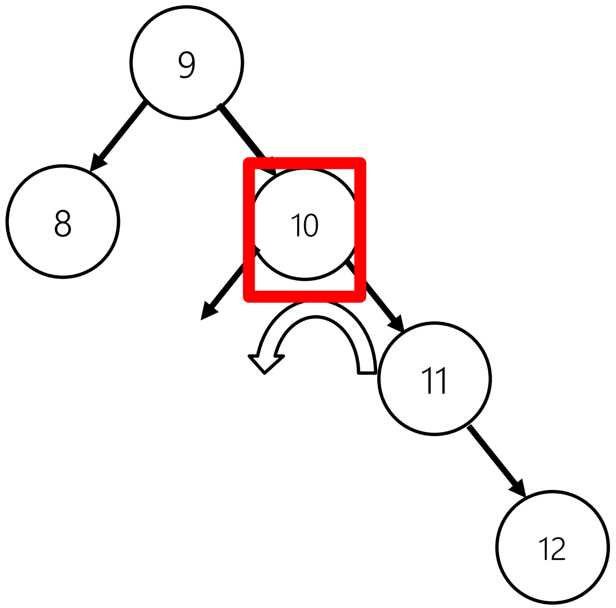
AVL Example: 8,9,10,12,11



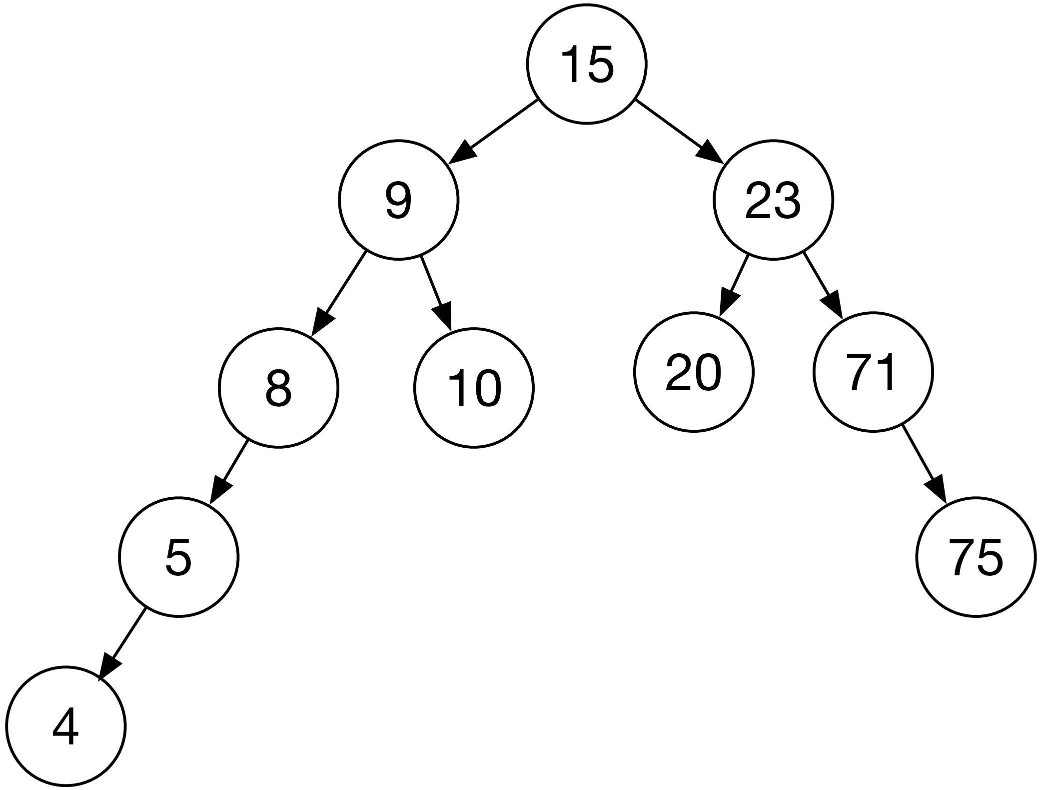
AVL Example: 8,9,10,12,11



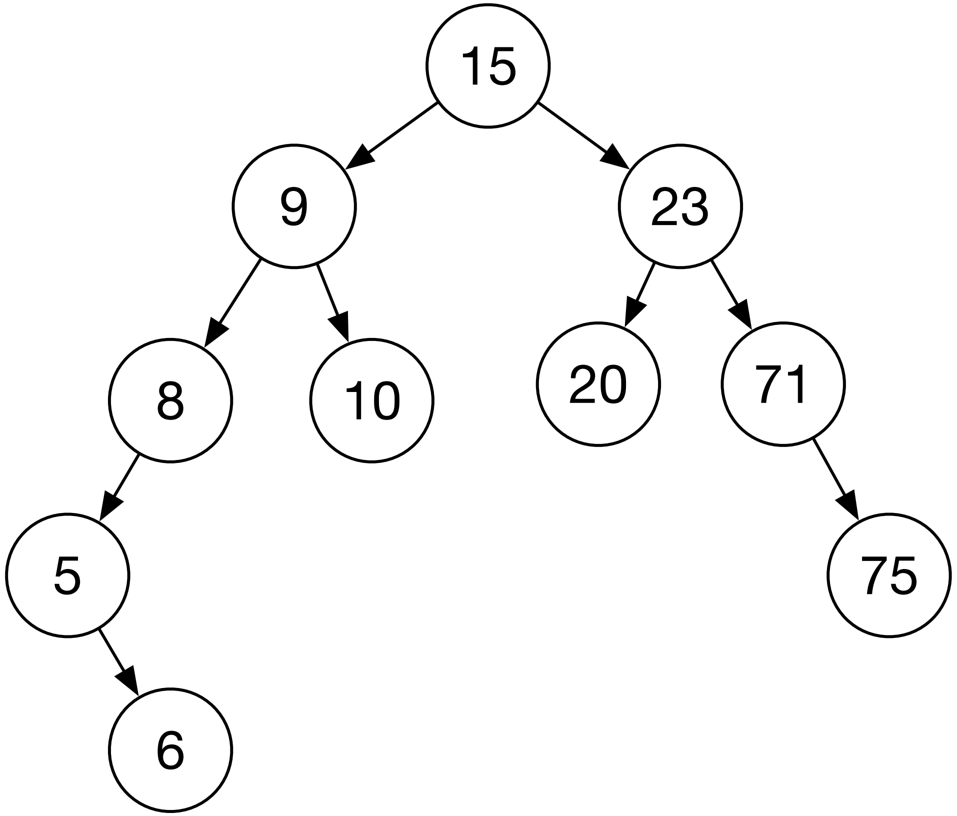
AVL Example: 8,9,10,12,11



Worksheet (Q10A)



Worksheet (Q10B)



How Long Does Rebalancing Take?

Assume we store in each node the height of its subtree.

How do we find an unbalanced node?

How many rotations might we have to do?

How Long Does Rebalancing Take?

Assume we store in each node the height of its subtree.

How do we find an unbalanced node?

- Just go back up the tree from where we inserted.

How many rotations might we have to do?

- Just a single or double rotation on the lowest unbalanced node.
- A rotation will cause the subtree rooted where the rotation happens to have the same height it had before insertion.

Lots of cool Self-Balancing BSTs out there!

Popular self-balancing BSTs include:

AVL tree

Splay tree

2-3 tree

AA tree

Red-black tree

Scapegoat tree

Treap

(Not covered in this class, but several are in the textbook and all of them are online!)

(From https://en.wikipedia.org/wiki/Self-balancing_binary_search_tree#Implementations)