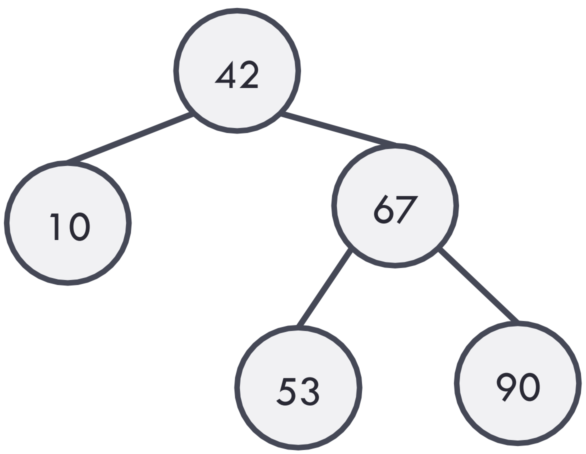
CSC230

Intro to C++ Lecture 21

Outline

- Lab 10 / Project 4 (Required)
- AVL TREE Insertion and Rotation

Height of a node



What is the height of node with key 53?

A. One
B. Two
C. Three
D. Zero

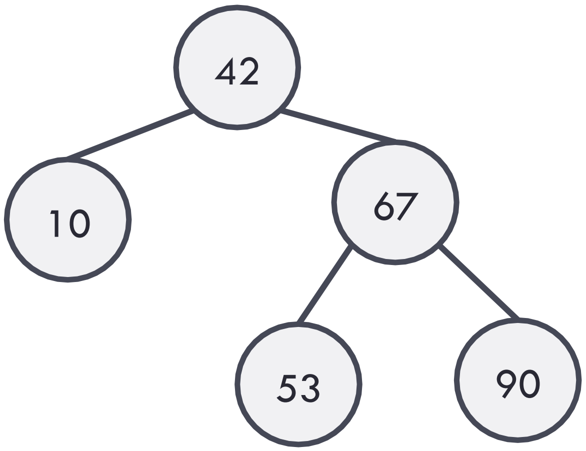
Height of a node =

O(log2N), A leaf node will have a height of 0.

Depth of a node=

O(log2N), A root node will have a depth of 0.

Computation of height of a node



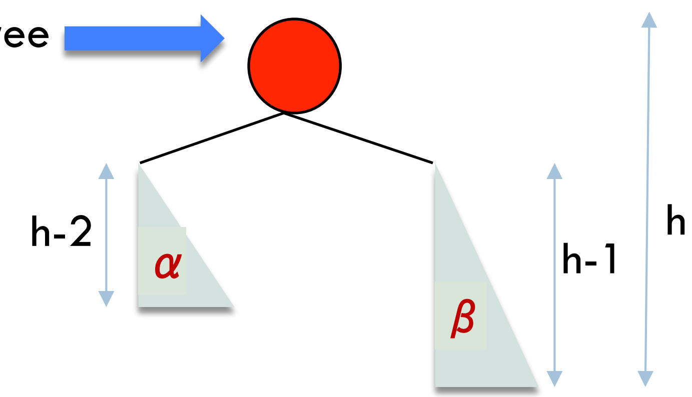
- Goal of AVL trees: For each node, maintain the difference between height of left and right children to within +/-1

- Each node maintains then a balance factor:
- Balance factor= height of the right child – height of left child

AVL tree

- Invented by ***Georgy Adelson-Velsky*** and ***Evgenii Landis*** (**AVL**) in 1962
- It is a self-balancing binary search tree
- Lookup, insertion, and deletion can be done in **O(log n)** under both average and worst cases. n is the number of nodes in the tree
- The heights of two child subtrees of ***any*** given node diff by at most one

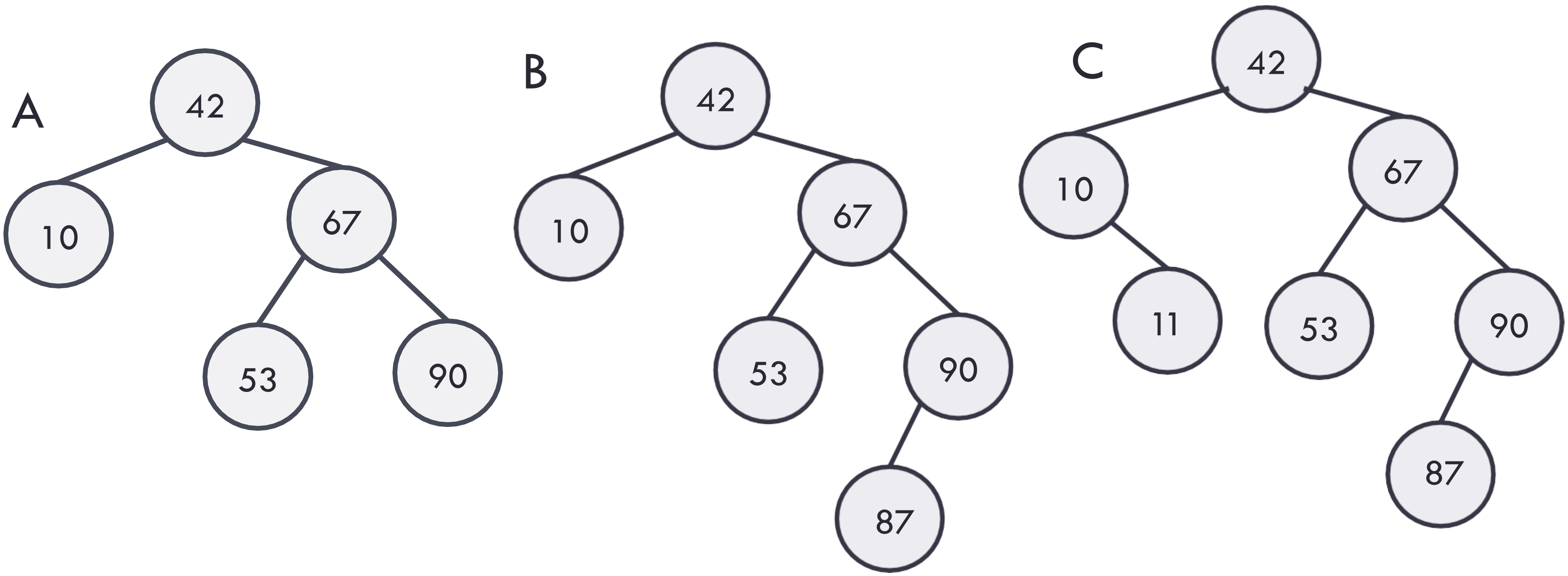
Any node of AVL tree



Insert and Rotation in AVL Trees

- Insert operation may cause balance factor to become 2 or –2 for some node
  - only nodes on the path from insertion point to root node have possibly changed in height
  - So after the Insert, go back up to the root node by node, updating heights
  - If a new balance factor (the difference $h_{left}-h_{right}$) is 2 or –2, adjust tree by *rotation* around the node

Which of the following are balanced

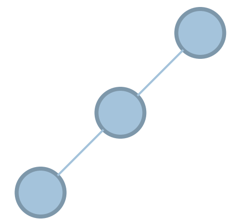


D. A&C
E. A&B&C

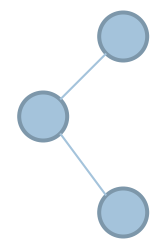
Annotate the trees with balance factors (for those that are balanced)

Insertion in an AVL tree

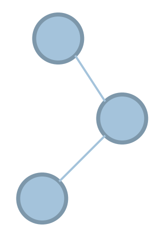
- Let us call the node that must be rebalanced **α**
  - Since any node has at most 2 children, and a height imbalance requires that **α's** 2 subtrees' height differ by 2, there are 4 violation cases:



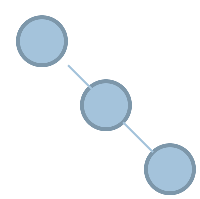
1. An insertion into the left subtree of the left child of **α.**



2. An insertion into the right subtree of the left child of **α**.



3. An insertion into the left subtree of the right child of **α**.



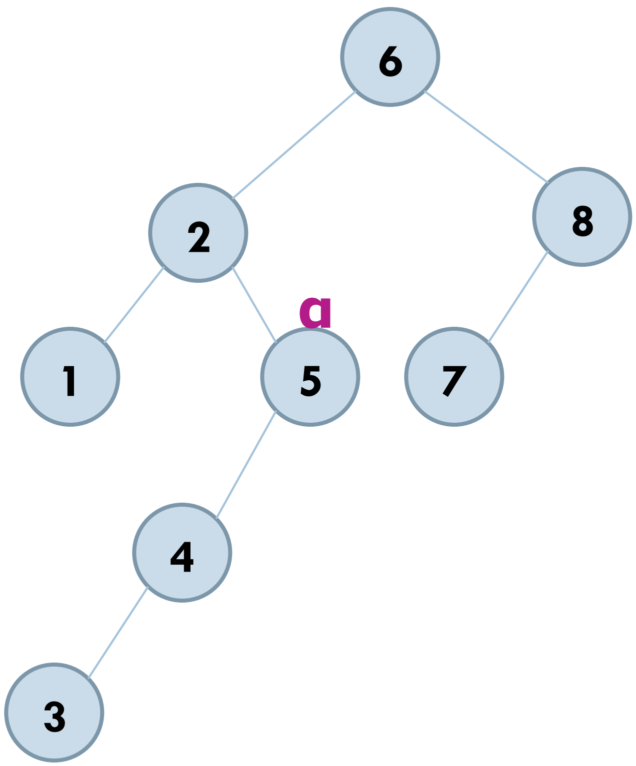
4. An insertion into the right subtree of the right child of **α**.

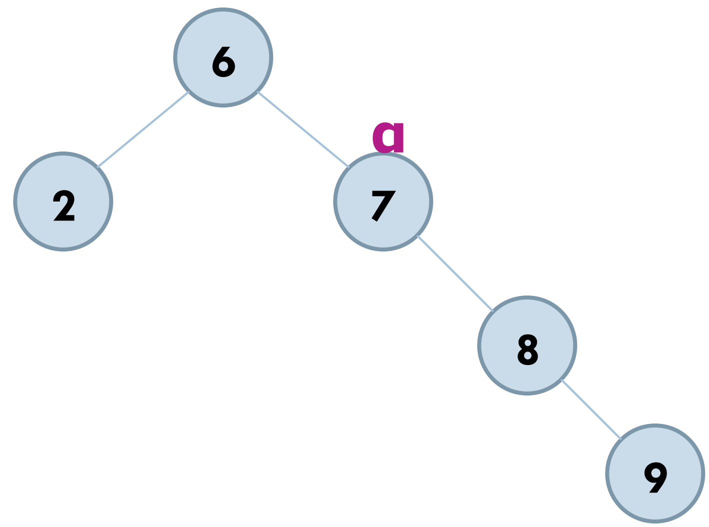
Insertion in an AVL tree

- Outside cases (left-left or right-right), fixed by a **single rotation**:

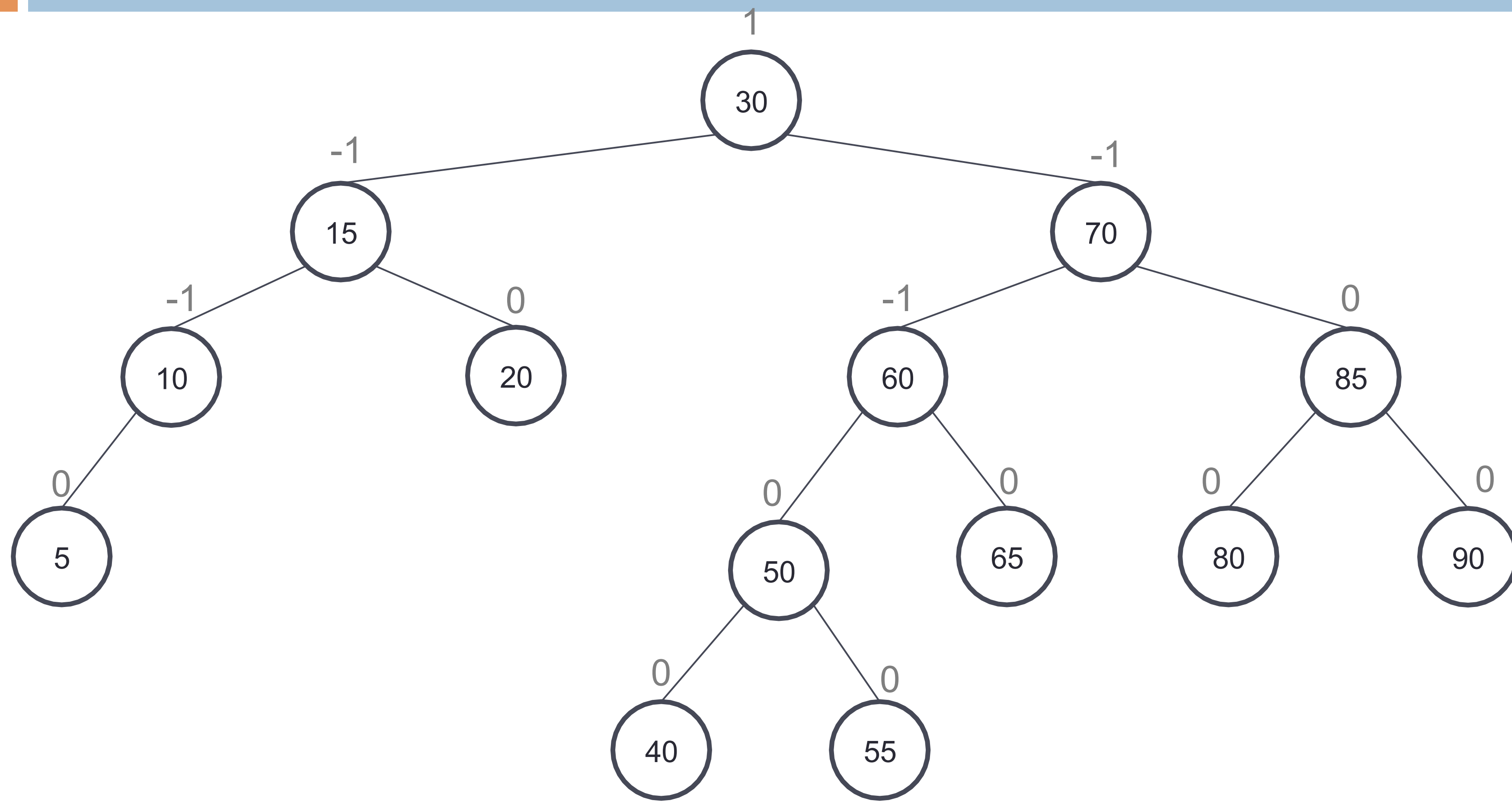
  (1) An insertion into the left subtree of the left child of α.

  (4) An insertion into the right subtree of the right child of α.





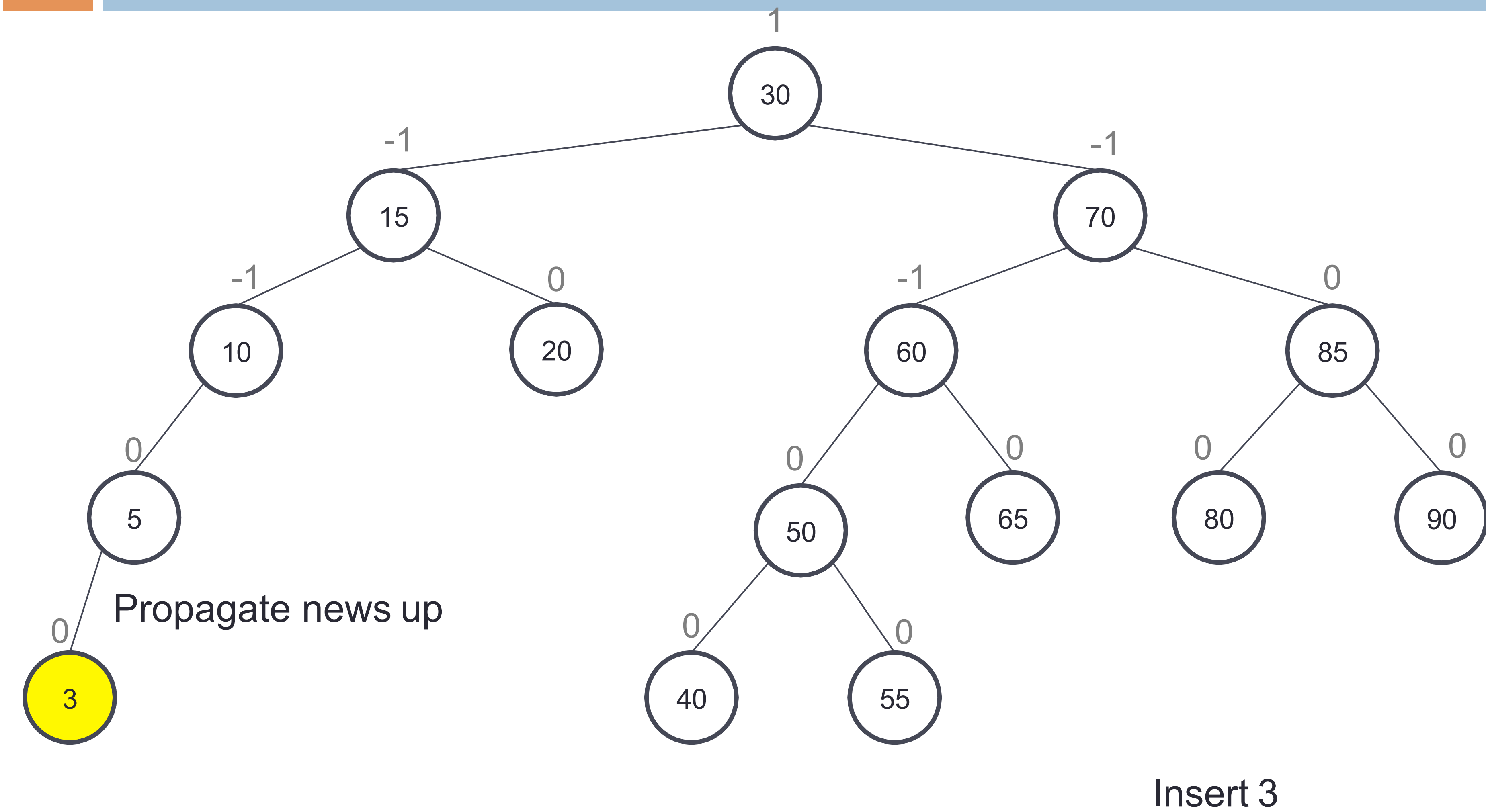
Inserting and rebalancing



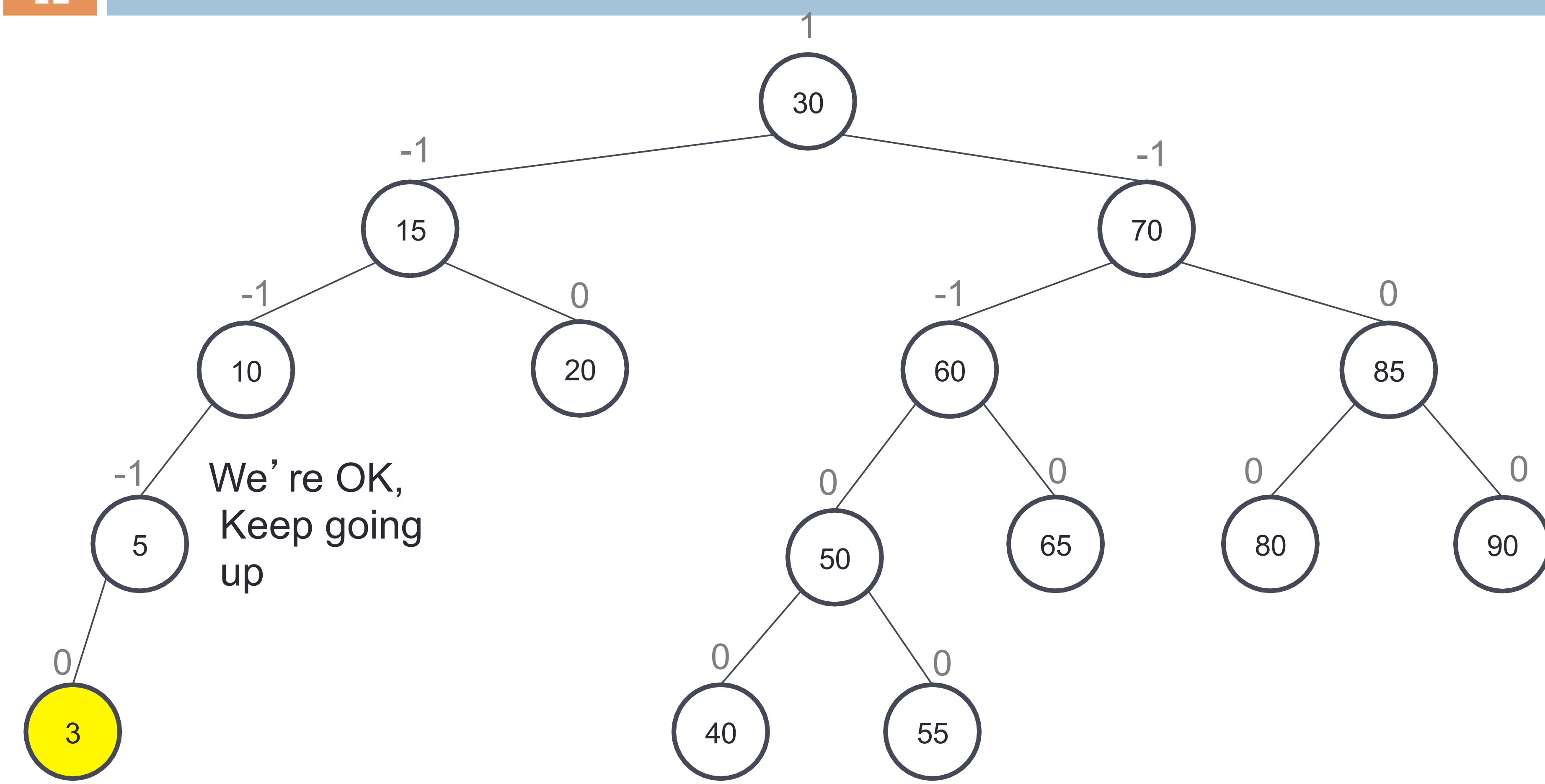
what are the balance factors

Insert 3

Inserting and rebalancing

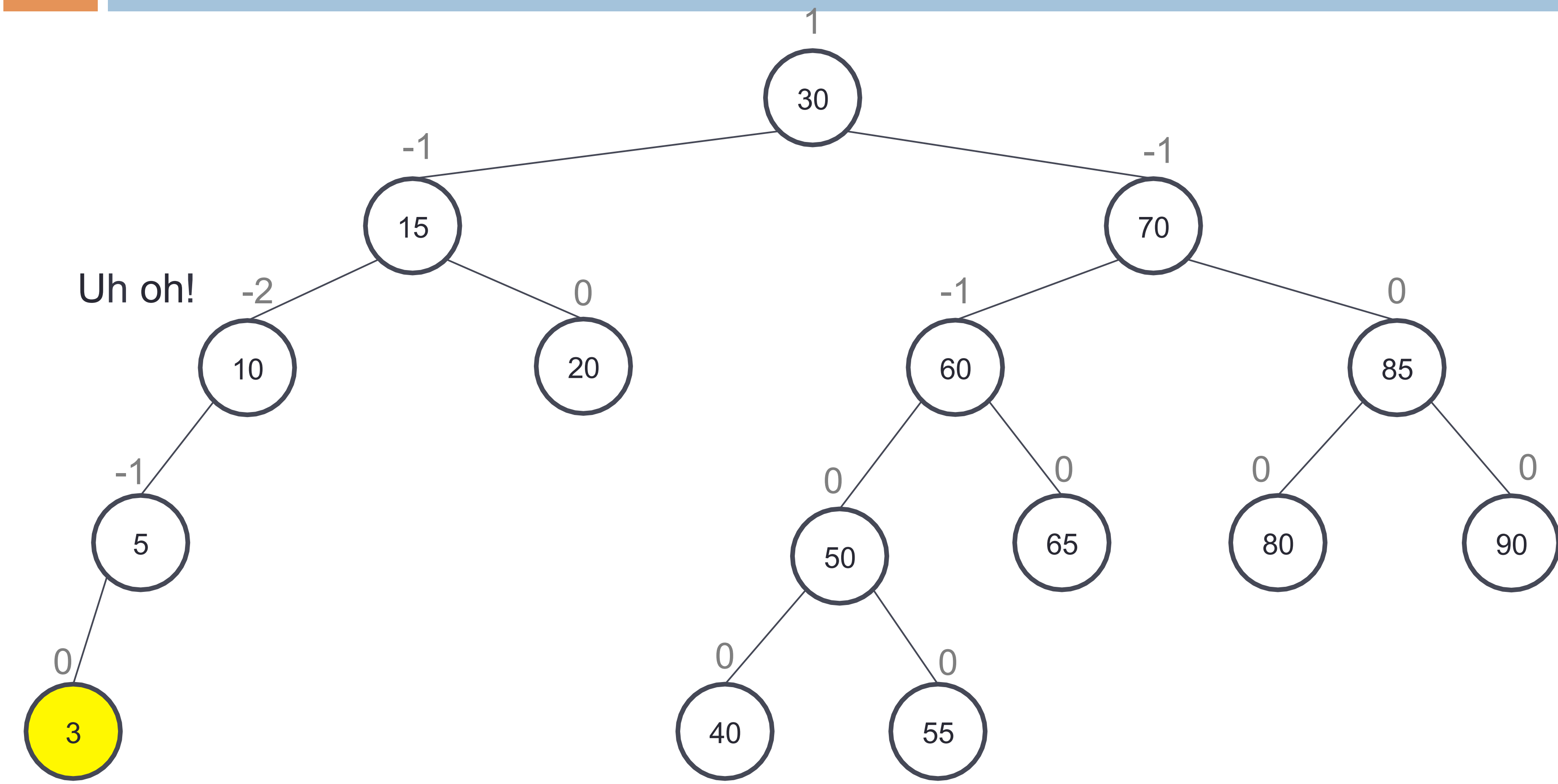


Inserting and rebalancing



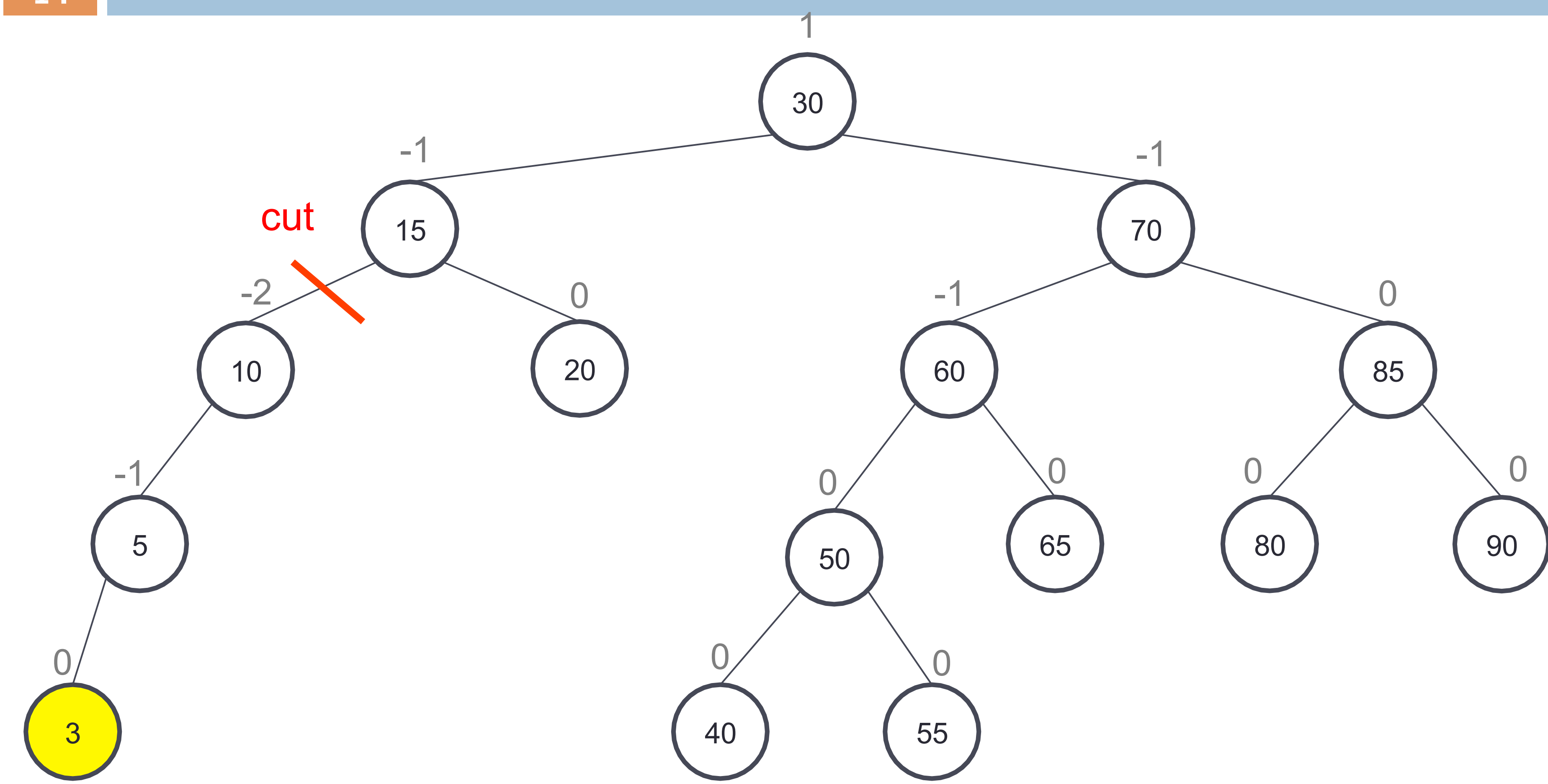
Insert 3

Inserting and rebalancing



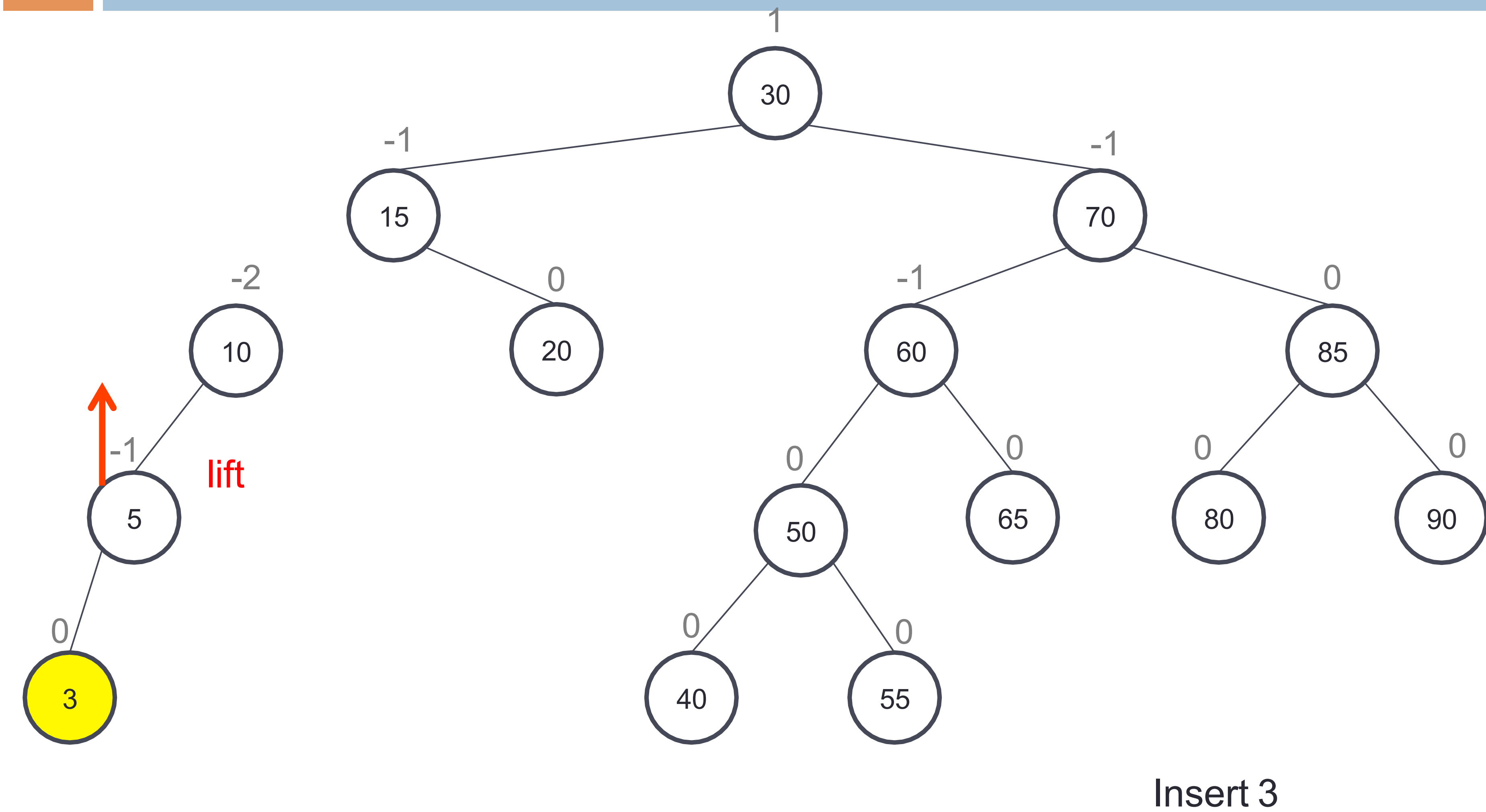
Insert 3

Inserting and rebalancing



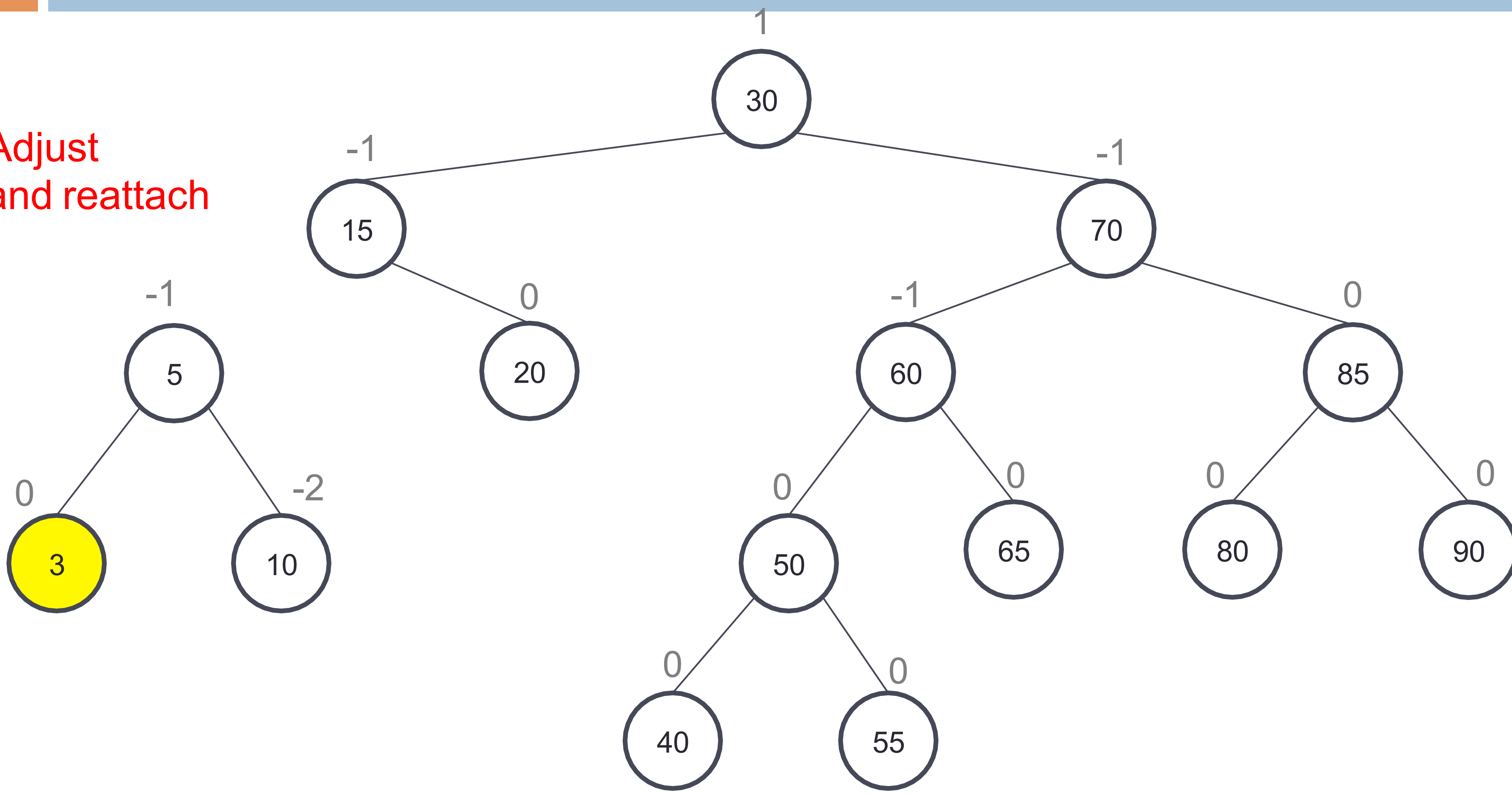
Insert 3

Inserting and rebalancing



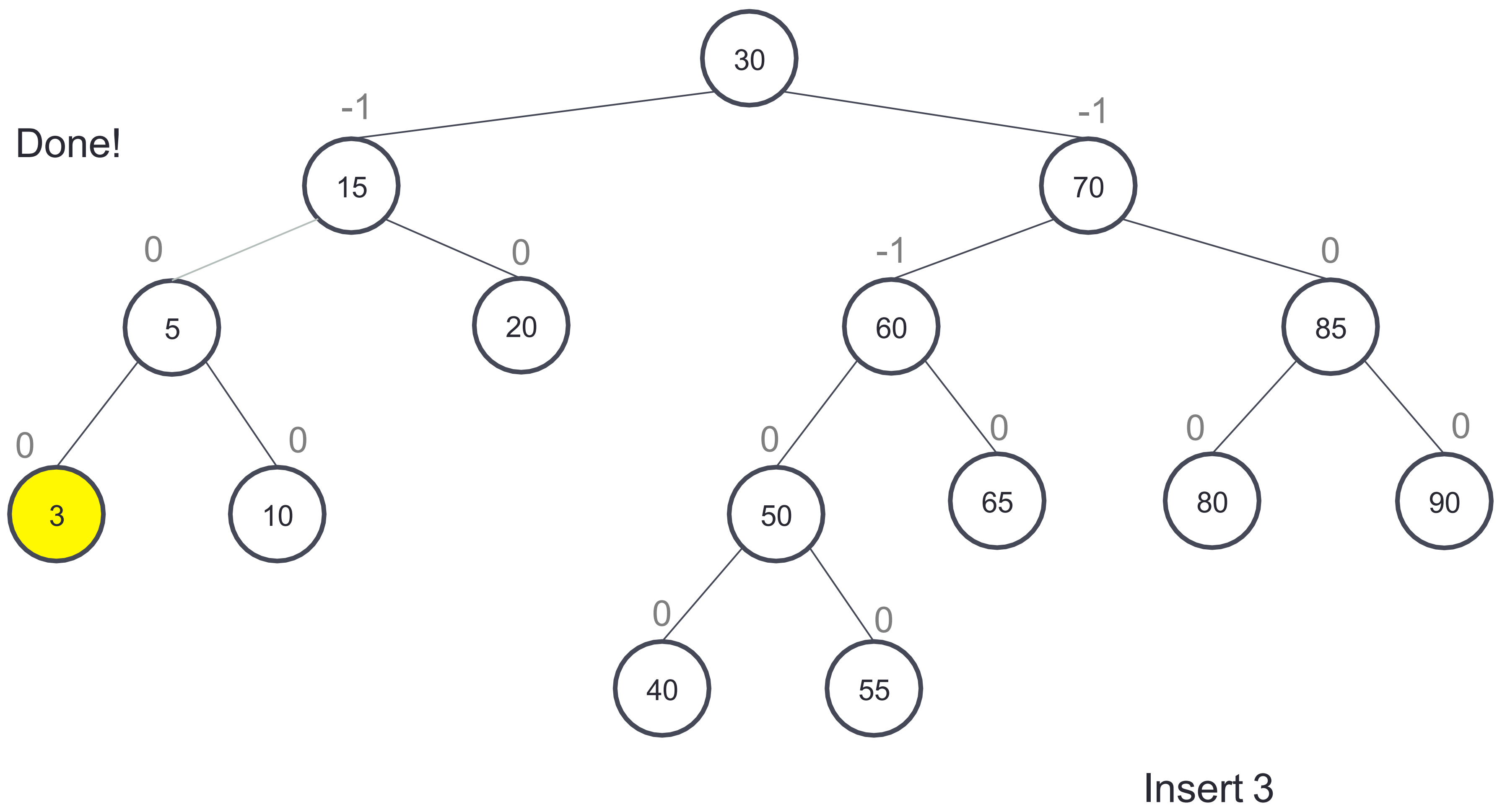
Inserting and rebalancing

Adjust
and reattach

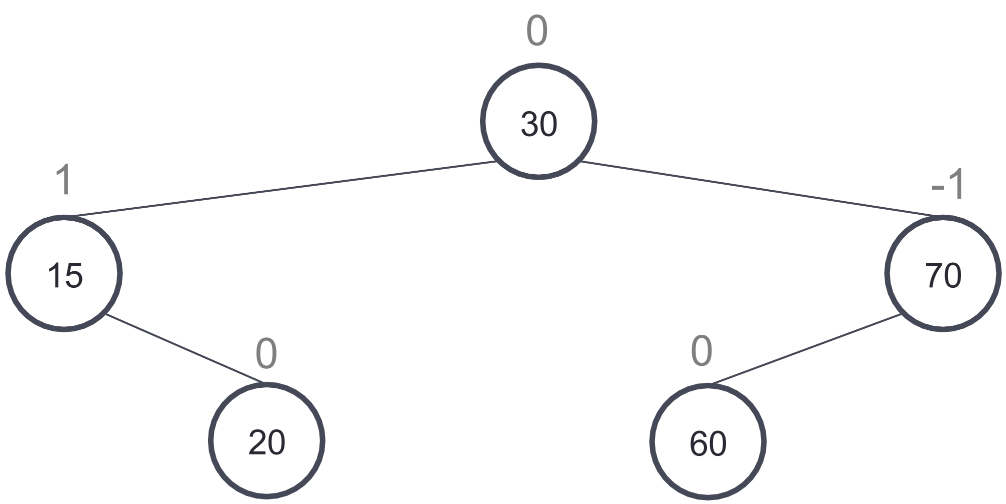


Insert 3

We just did a single rotation of 5 around 10



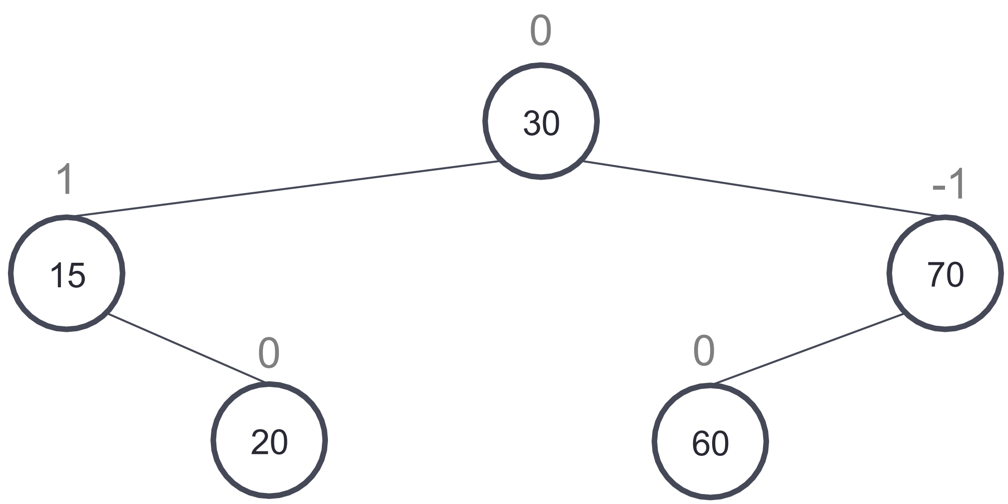
Single rotation practice



What could you insert into this AVL tree that would result in a single right rotation?

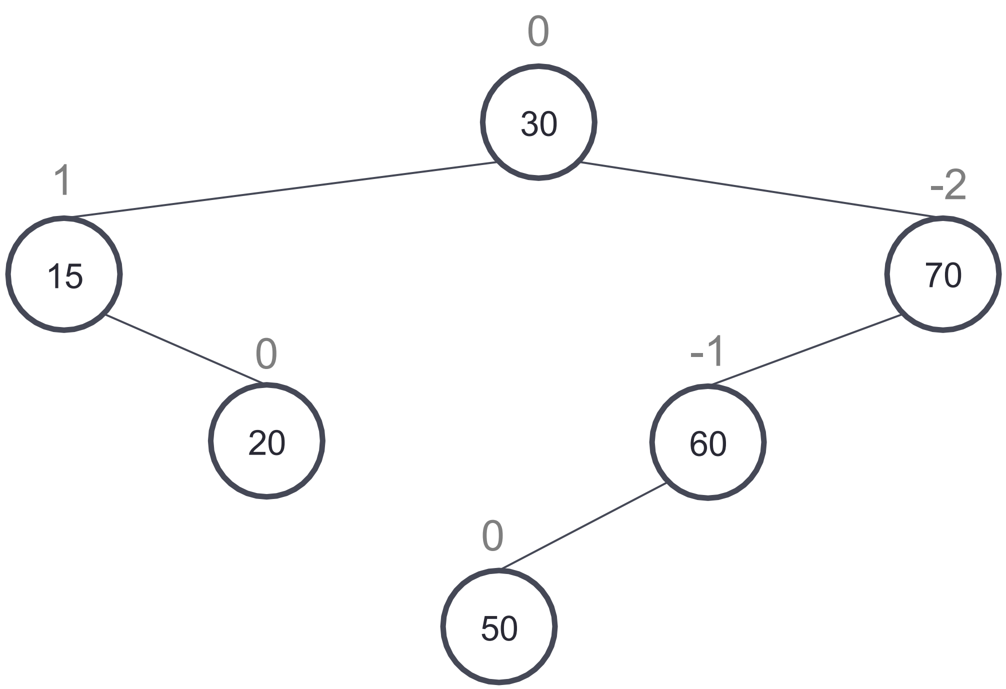
A. 71
B. 10
C. 50
D. 66

Single rotation practice



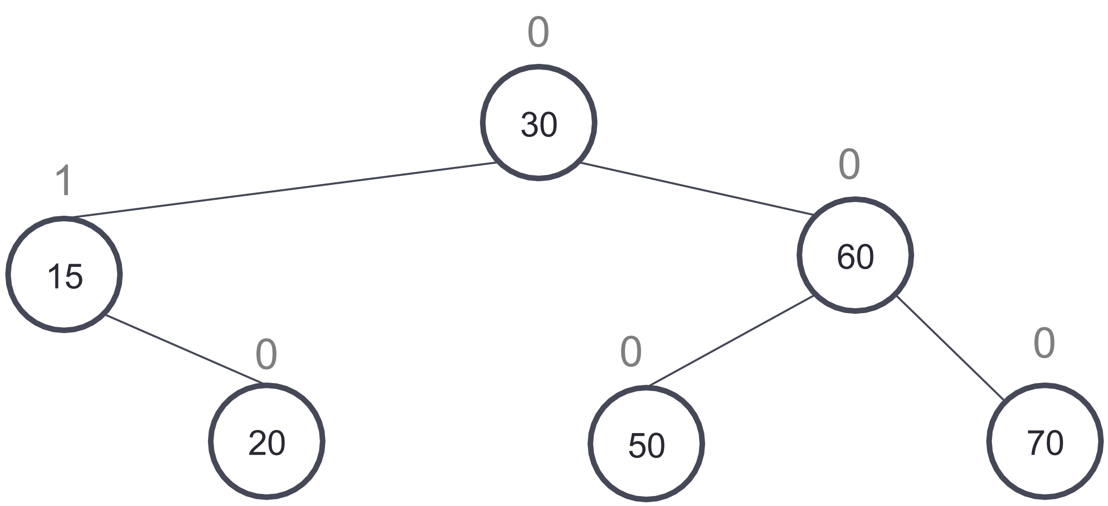
Insert 50. Draw the resulting AVL tree.

Single rotation practice



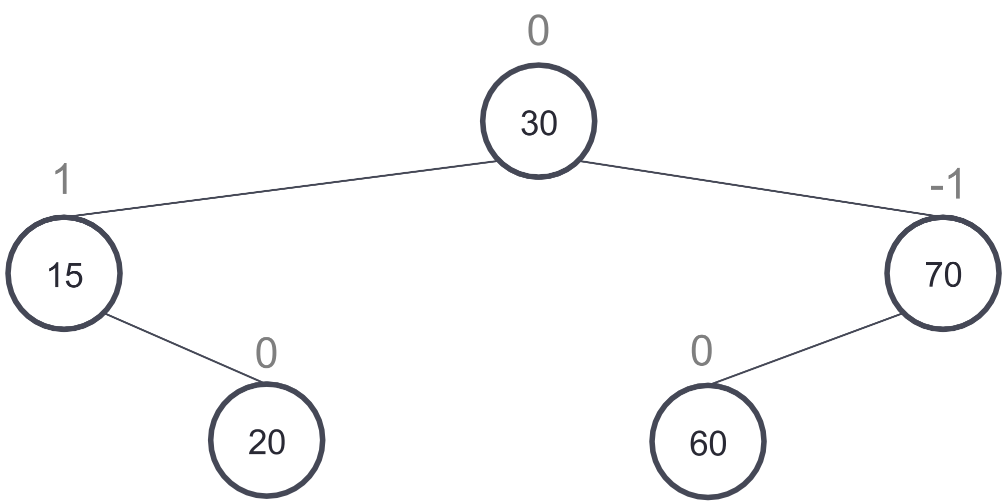
After insertion

Single rotation practice



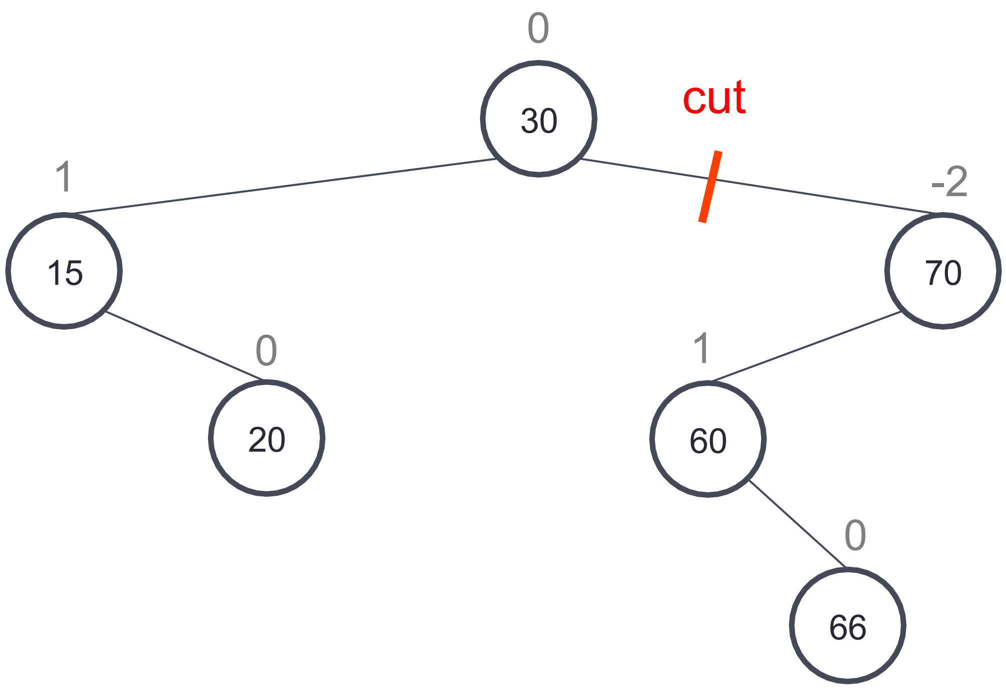
After rotation

Single rotation is not enough

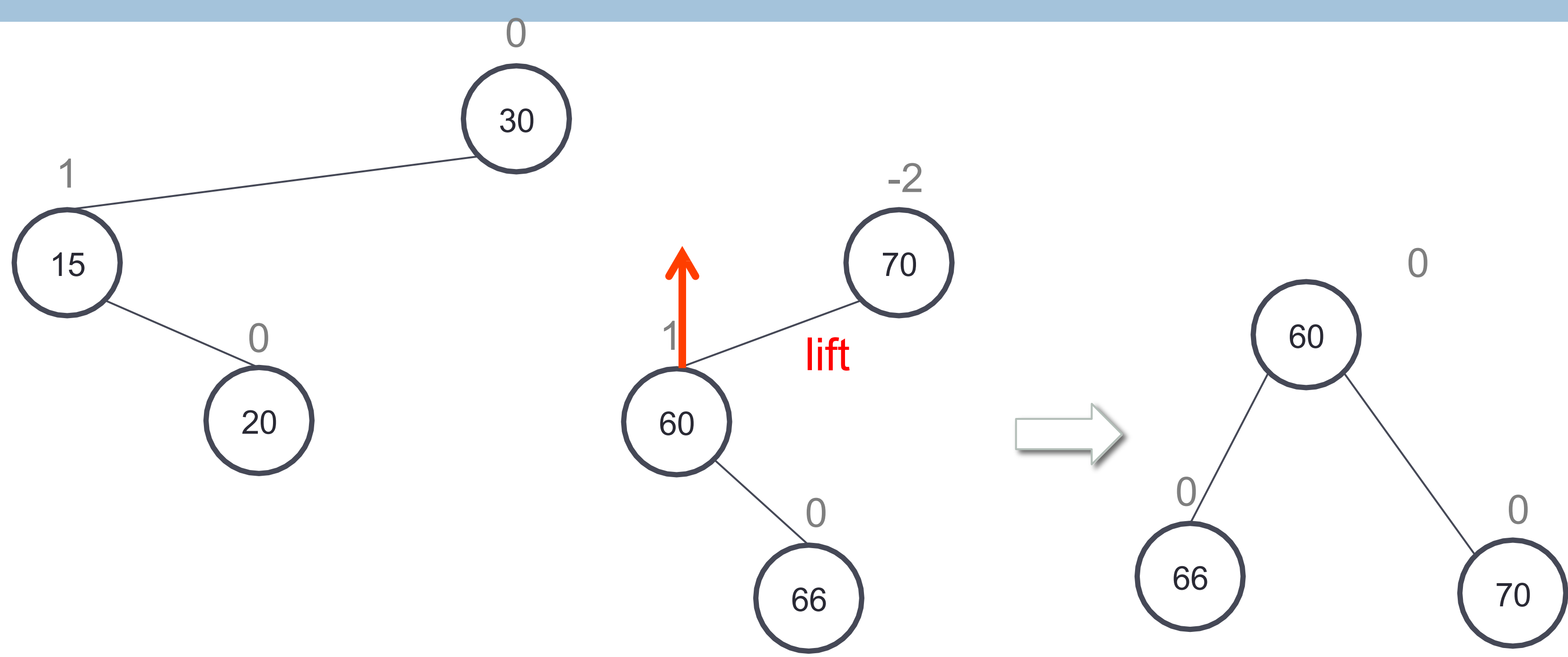


What happens if we insert 66?

Single rotation is not enough



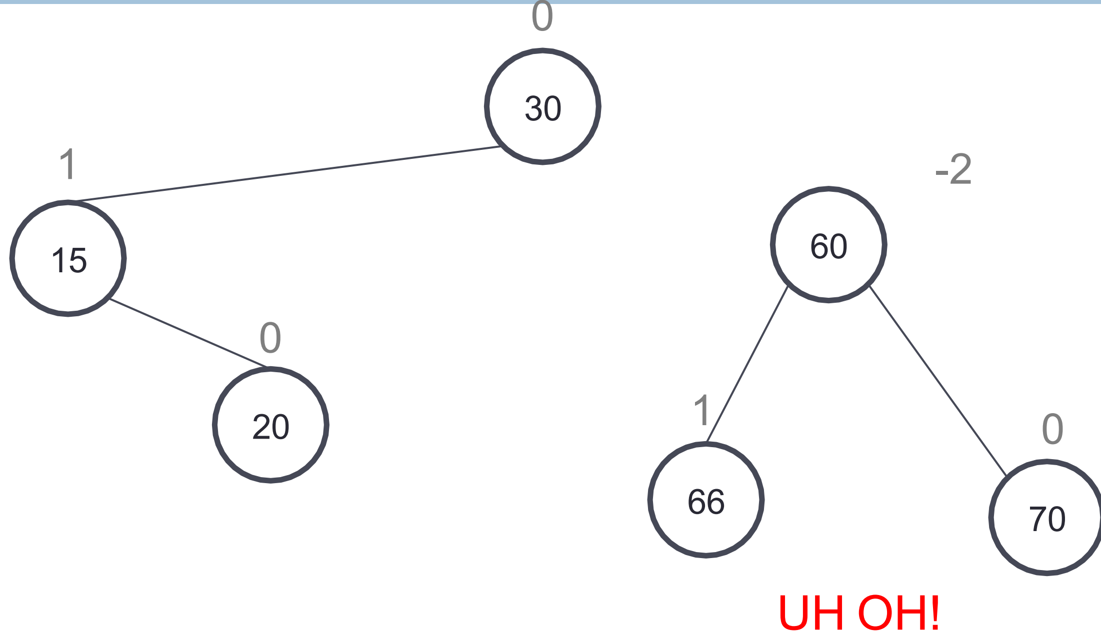
Is this a valid rotation?



A. Yes
B. No

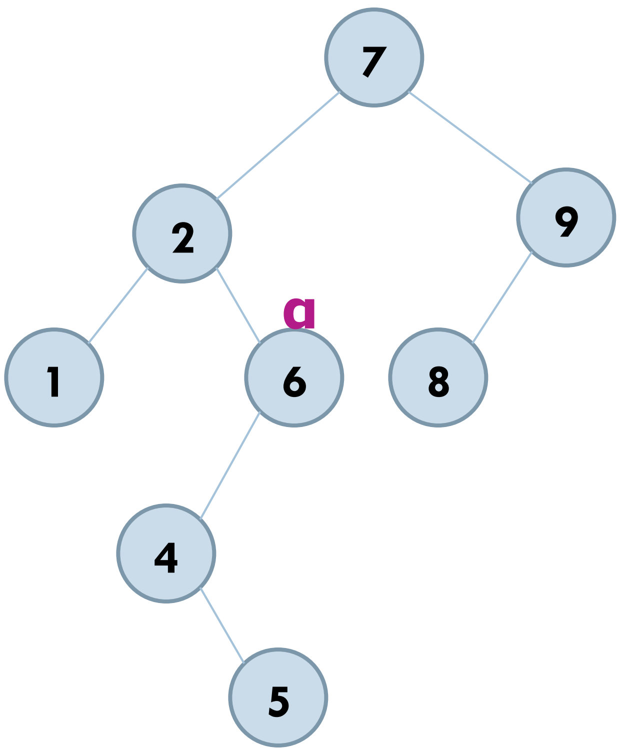
No

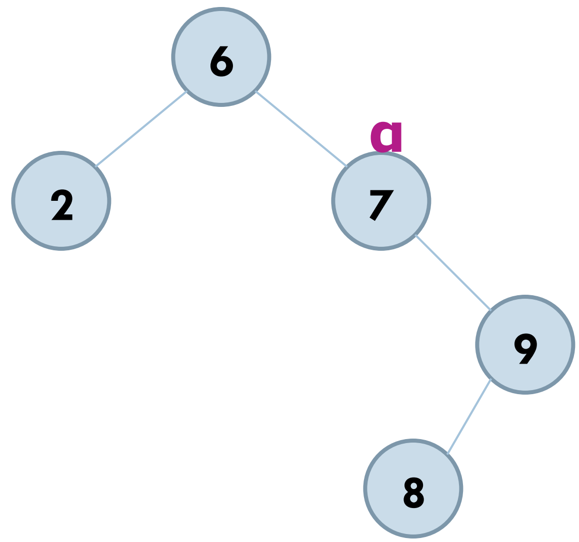
Invalid rotation!



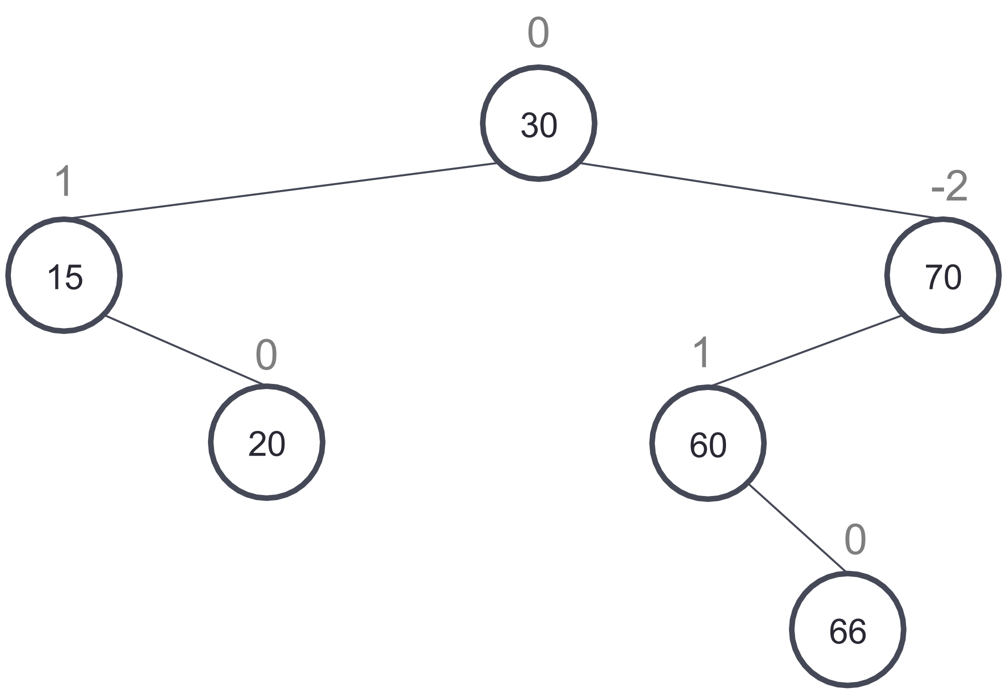
Insertion in an AVL tree

- Inside cases (right-left or left-right), fixed by a **double rotation**:
  (2) An insertion into the right subtree of the left child of α.
  (3) An insertion into the left subtree of the right child of α.



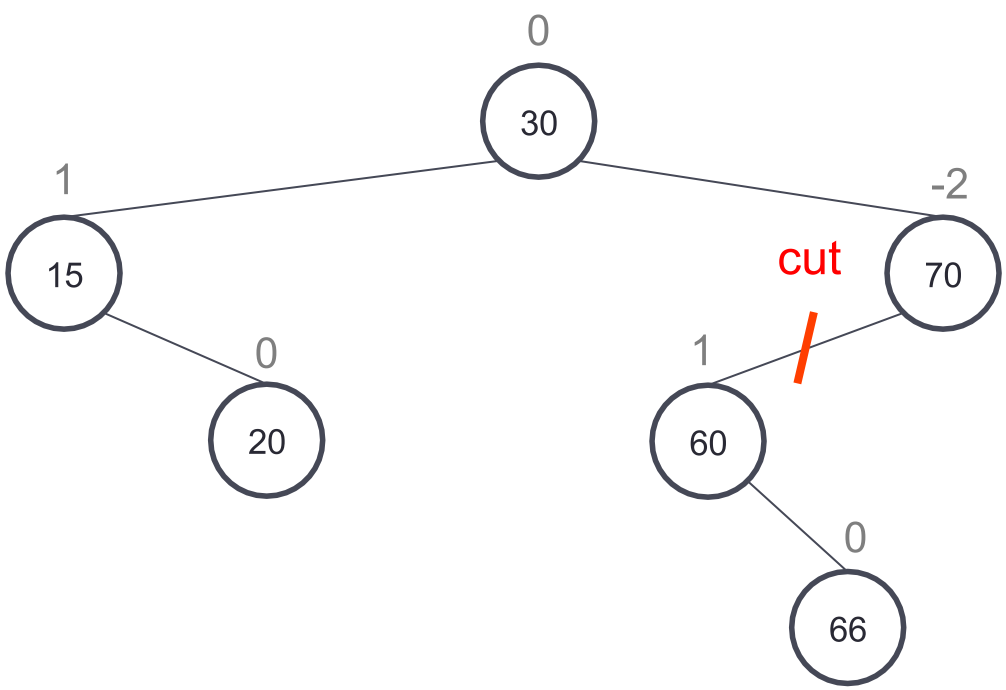


Double rotation to the rescue



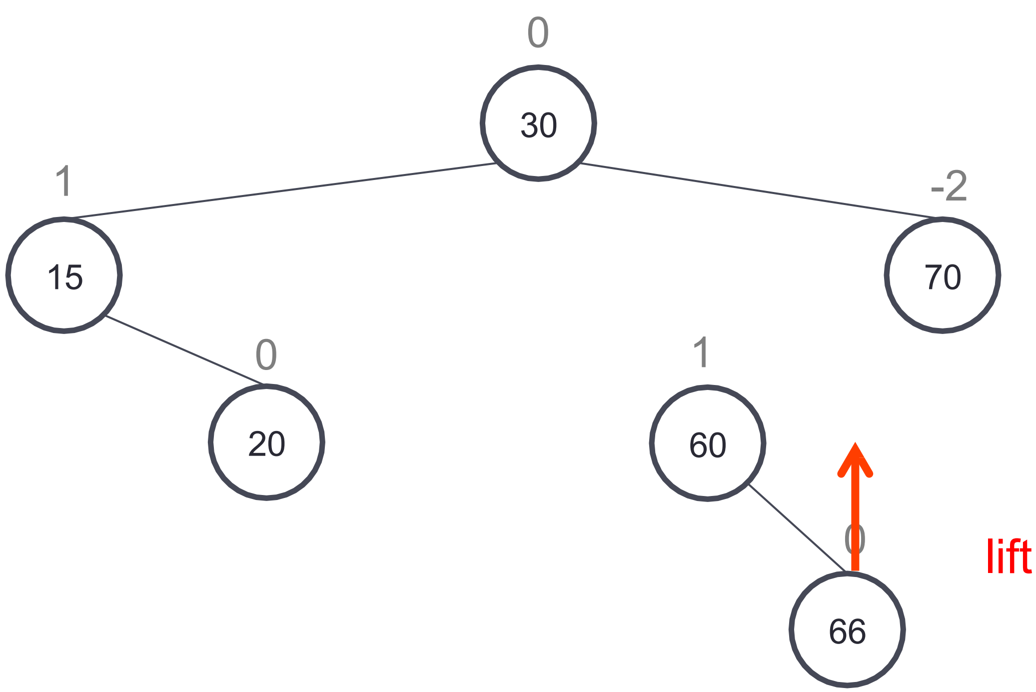
Single rotations only work when involved nodes are "in a line"
So we will first rotate left around 60, then we can rotate right around 70.

Double rotation to the rescue



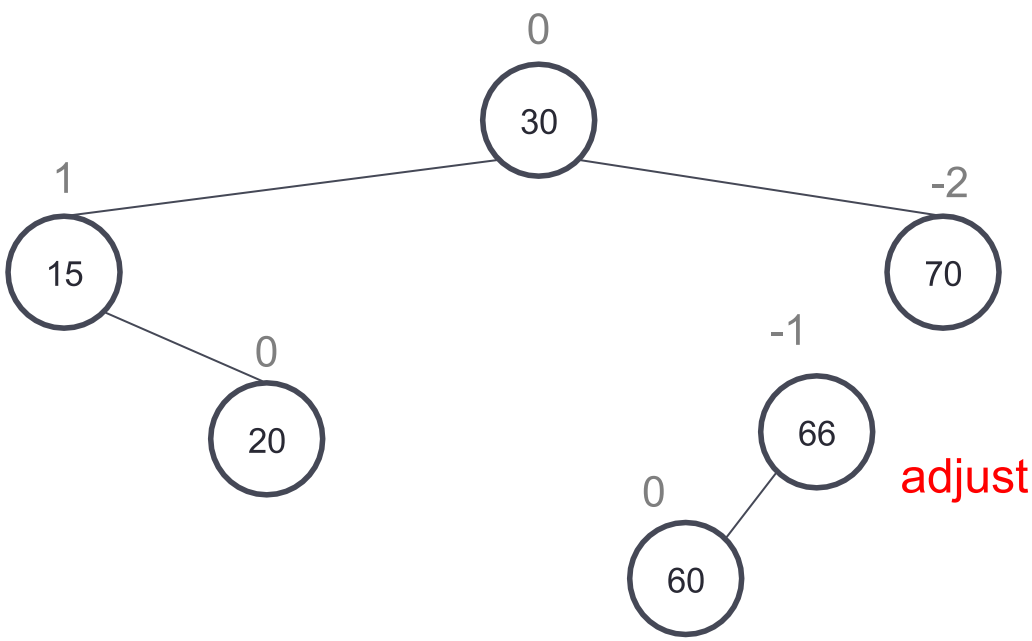
Single rotations only work when involved nodes are "in a line"
So we will first rotate left around 60, then we can rotate right around 70.

Double rotation to the rescue



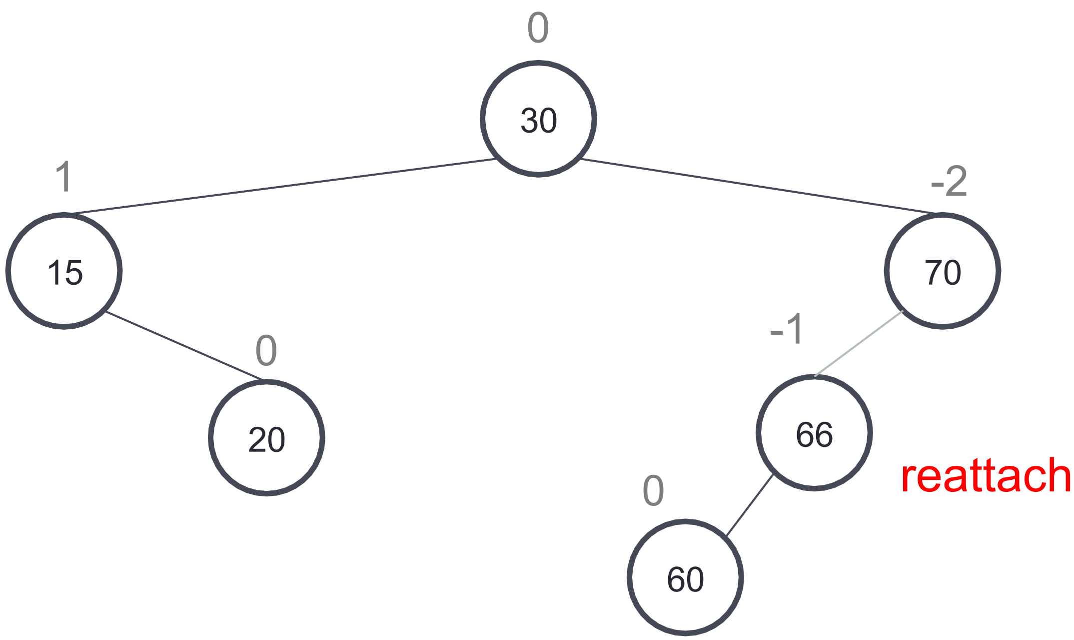
Single rotations only work when involved nodes are “in a line”
So we will first rotate left around 60, then we can rotate right around 70.

Double rotation to the rescue



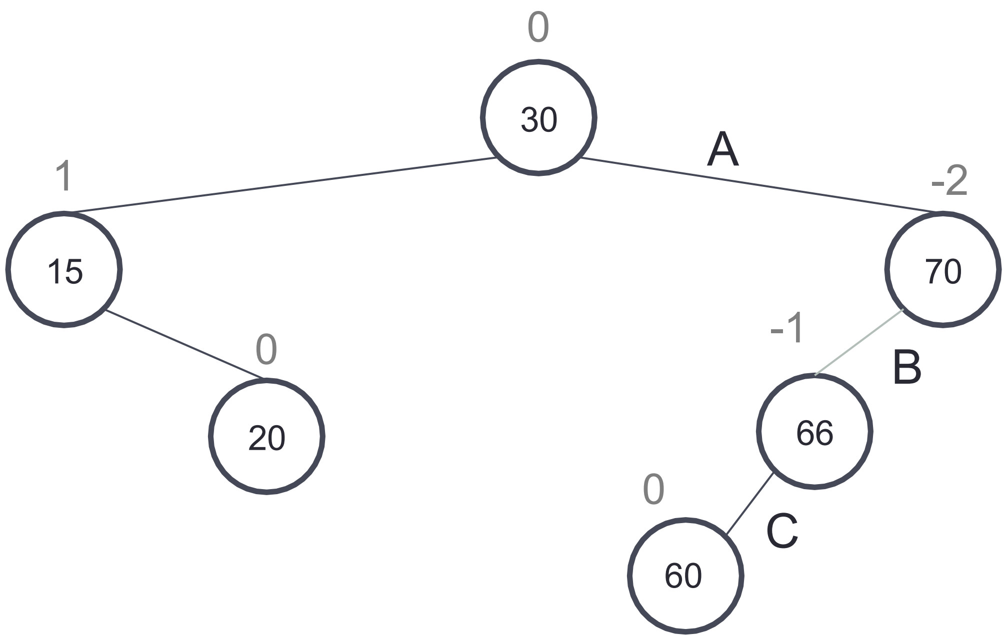
Single rotations only work when involved nodes are “in a line”
So we will first rotate left around 60, then we can rotate right around 70.

Double rotation to the rescue



Single rotations only work when involved nodes are "in a line"
So we will first rotate left around 60, then we can rotate right around 70.

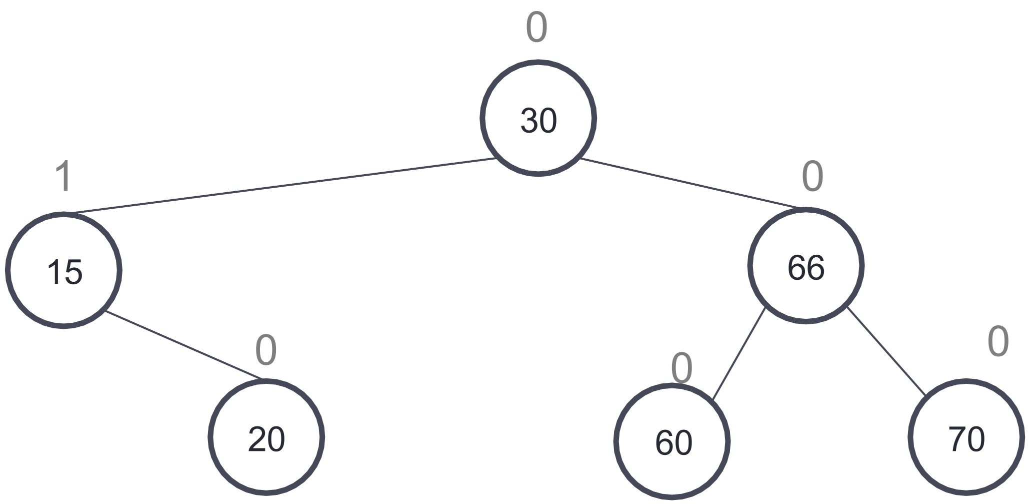
Double rotation to the rescue



Single rotations only work when involved nodes are "in a line" So we will first rotate left around 60, then we can rotate right around 70.

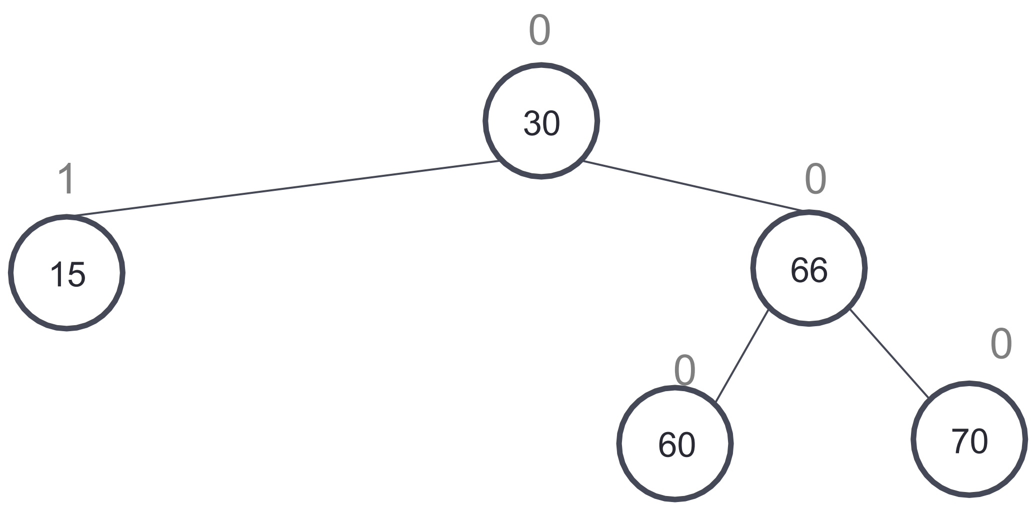
Where in the tree above should I cut to start this rotation?

Double rotation to the rescue



Single rotations only work when involved nodes are “in a line”
So we will first rotate left around 60, then we can rotate right around 70.

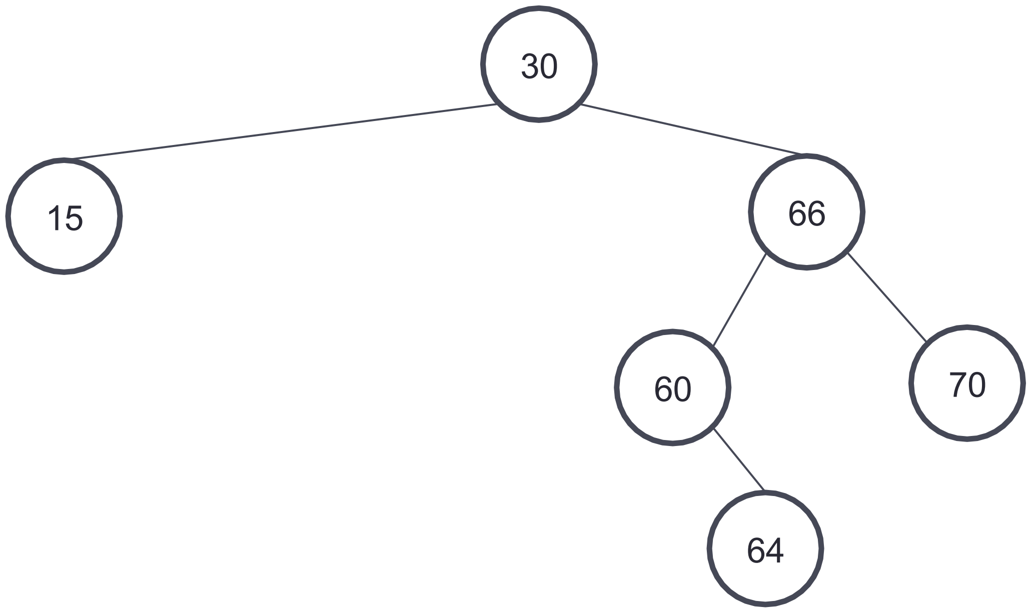
It's sometimes even more complicated



Insert 64… do we need a double or a single rotation?

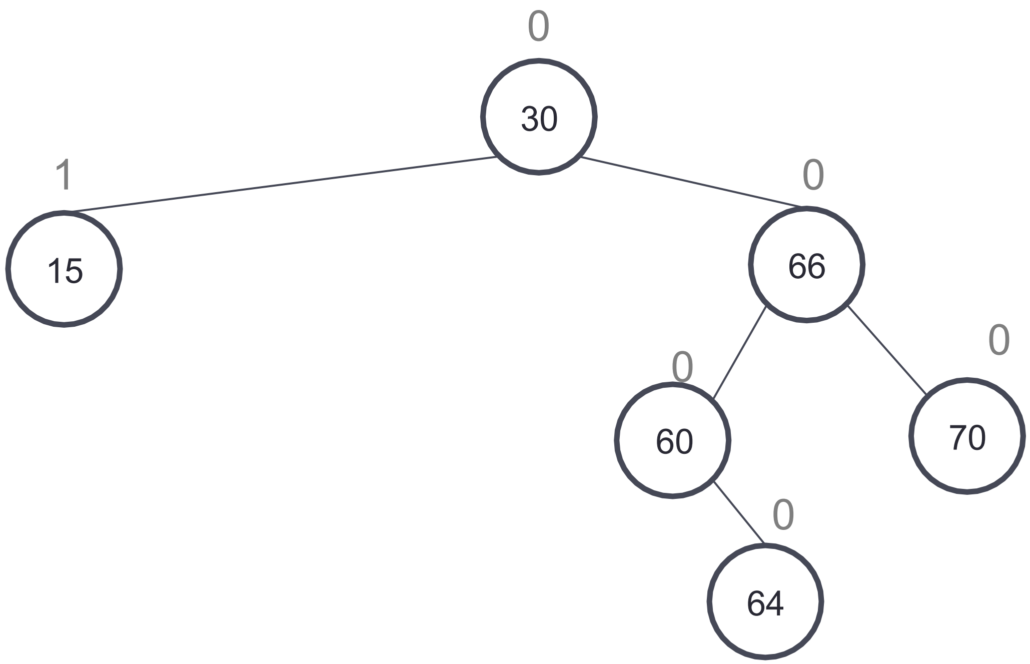
A. Double
B. Single
C. No rotation needed

Where is the tree out of balance?

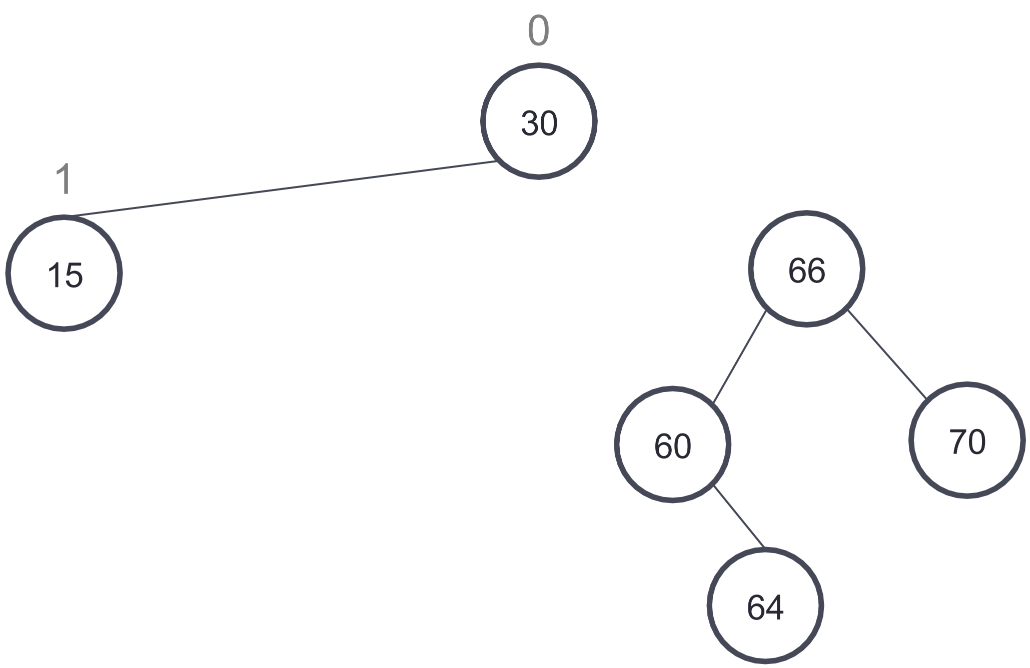


A. Node 30
B. Node 66
C. Node 60
D. Node 64

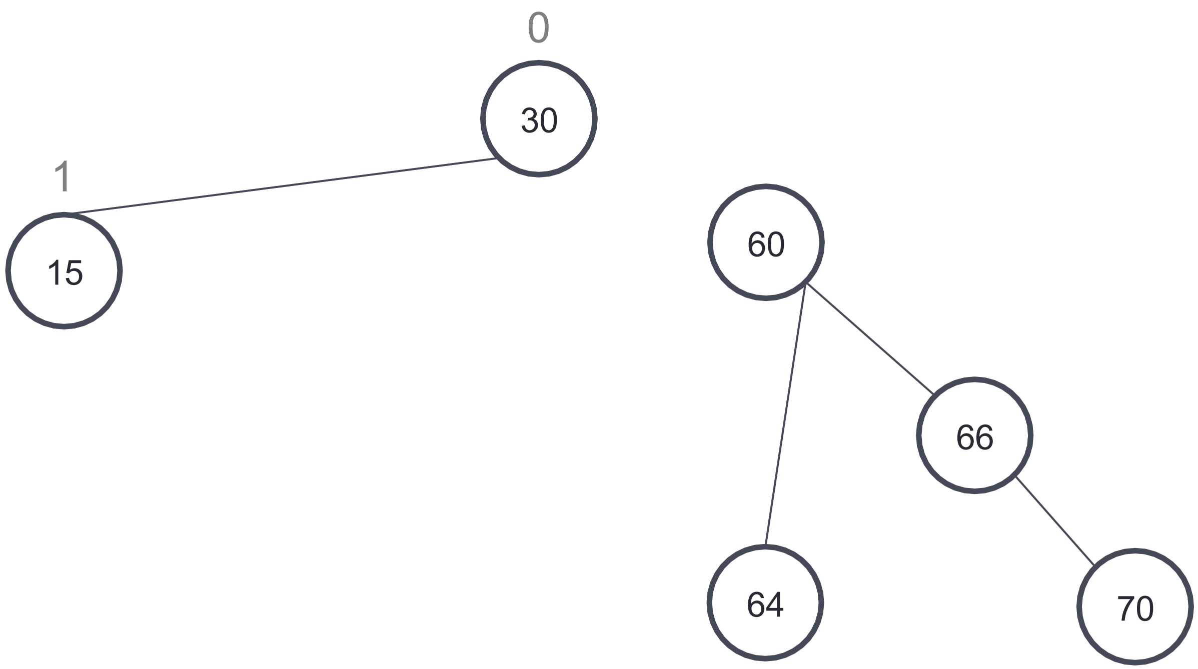
Rotate right around 66 to make a straight line



Rotate right around 66 to make a straight line

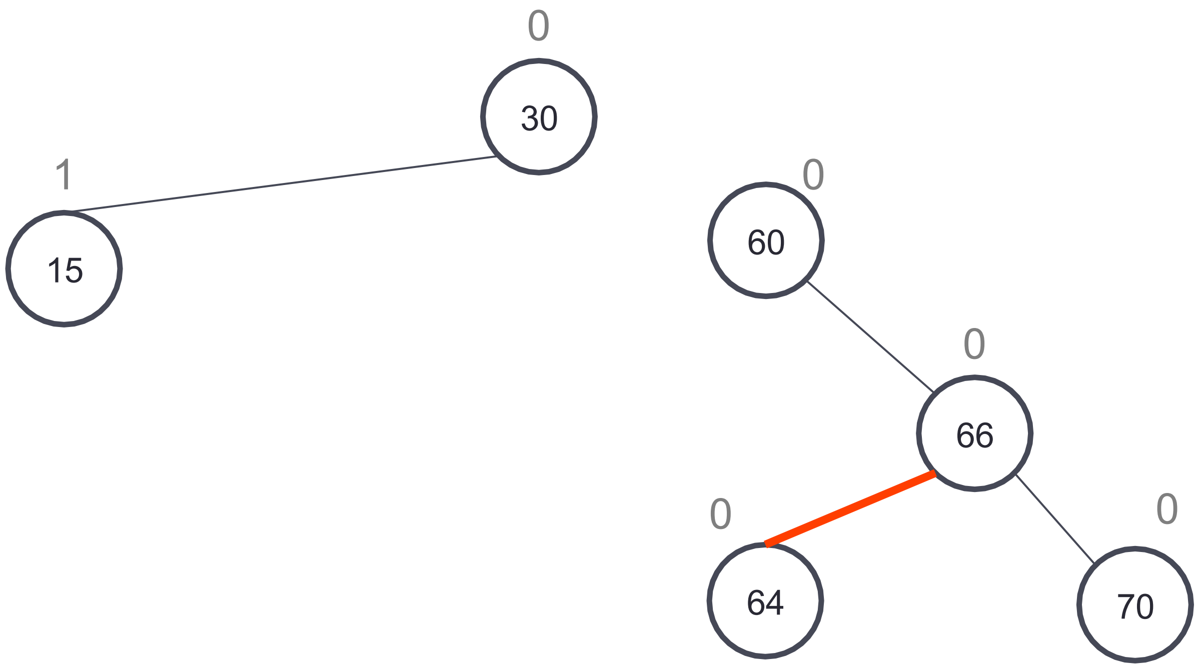


Rotate right around 66 to make a straight line



UH OH! Where do we put 64??
Are we stuck?

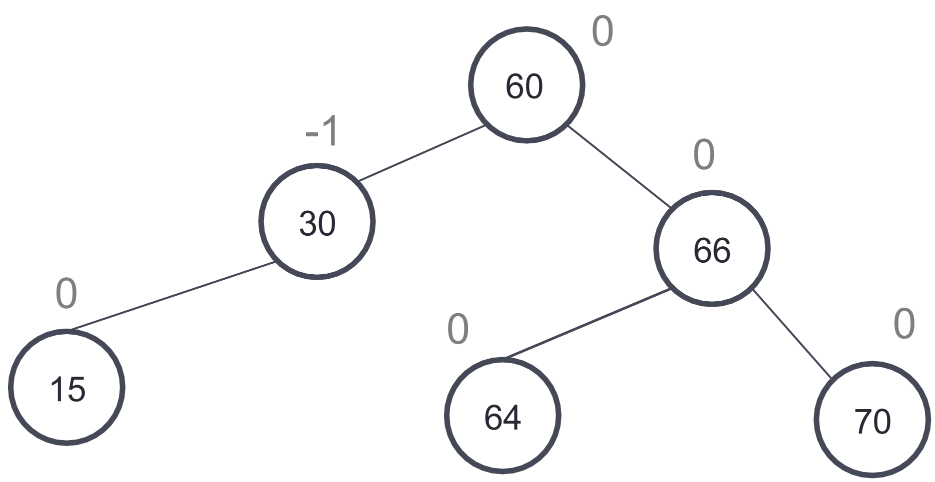
Rotate right around 66 to make a straight line



Will 64 always reattach there?

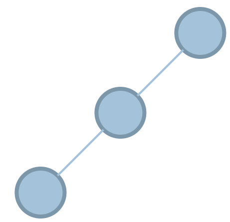
A. No, sometimes this doesn't work
B. Yes, this will always work

Finishing the rotation to balance the tree

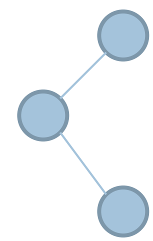


Insertion in an AVL tree

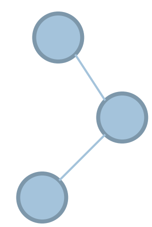
- Let us call the node that must be rebalanced α
  - Since any node has at most 2 children, and a height imbalance requires that α's 2 subtrees' height differ by 2, there are 4 violation cases:



1. An insertion into the left subtree of the left child of α.

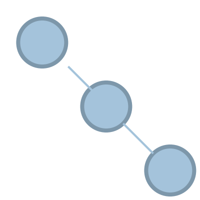


2. An insertion into the right subtree of the left child of α.



3. An insertion into the left subtree of the right child of α.

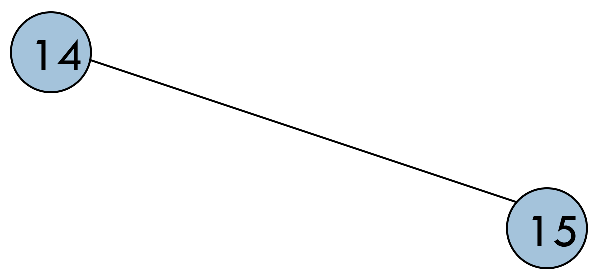
4. An insertion into the right subtree of the right child of α.



AVL Tree Rotations

Single rotations: insert 14, 15, 16, 13, 12, 11, 10

- First insert 14 and 15:

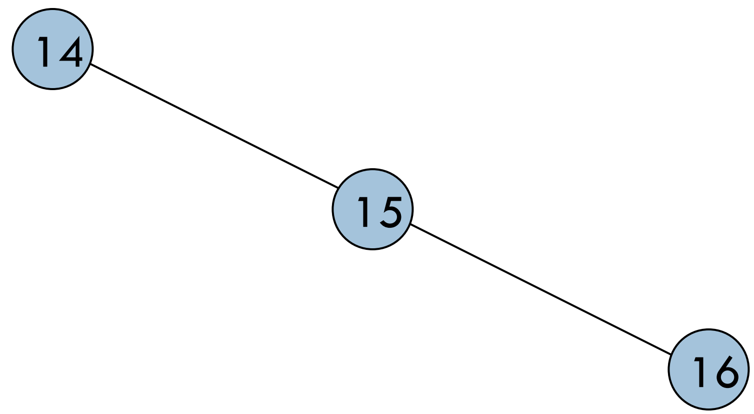


- Now insert 16.

AVL Tree Rotations

Single rotations: insert 14, 15, 16, 13, 12, 11, 10

- Inserting 16 causes AVL violation:

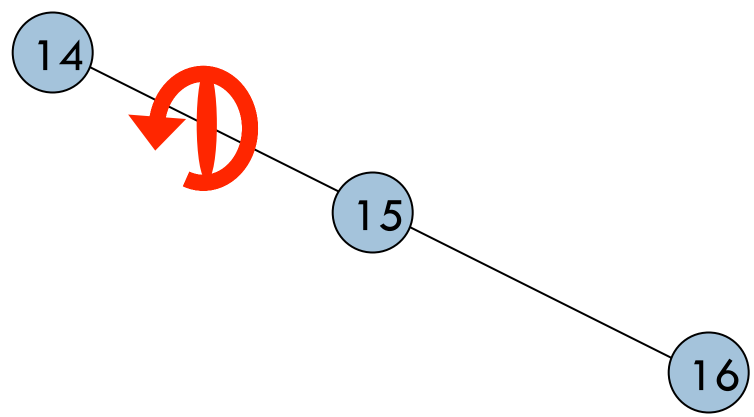


- Need to rotate.

AVL Tree Rotations

Single rotations: insert 14, 15, 16, 13, 12, 11, 10

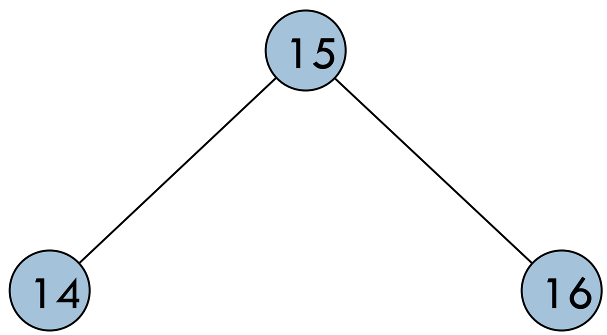
- Rotation type:



AVL Tree Rotations

Single rotations: insert 14, 15, 16, 13, 12, 11, 10

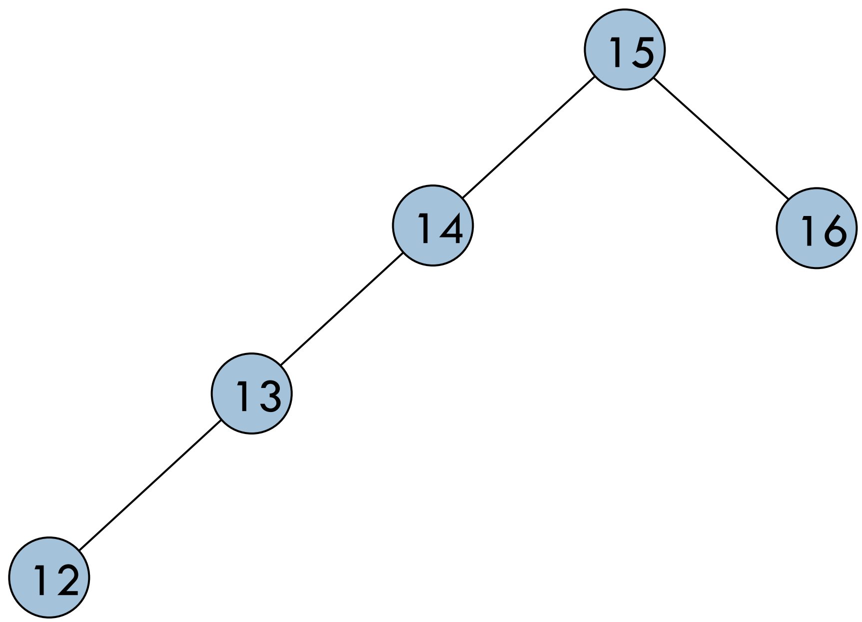
- Rotation restores AVL balance:



AVL Tree Rotations

Single rotations: insert 14, 15, 16, 13, 12, 11, 10

- Now insert 13 and 12:

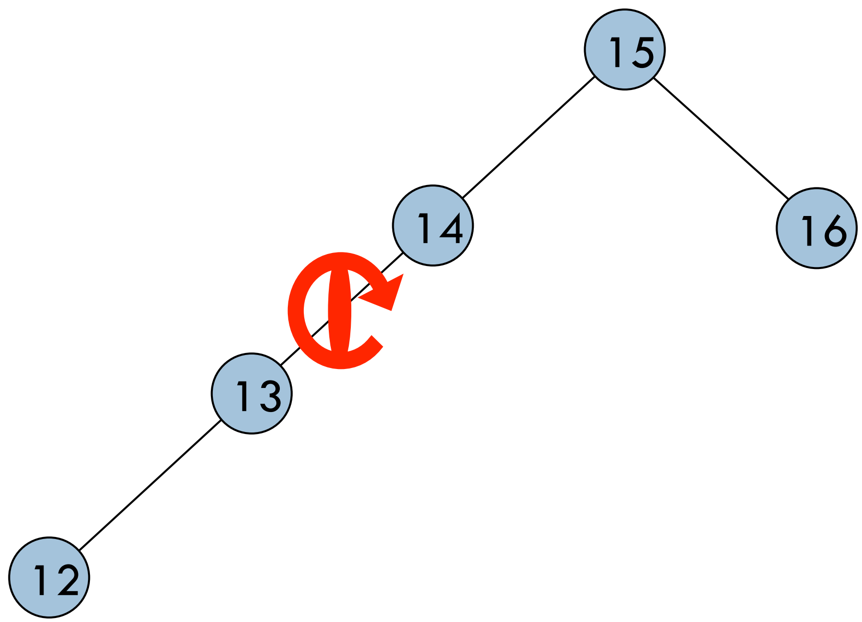


- AVL violation - need to rotate.

AVL Tree Rotations

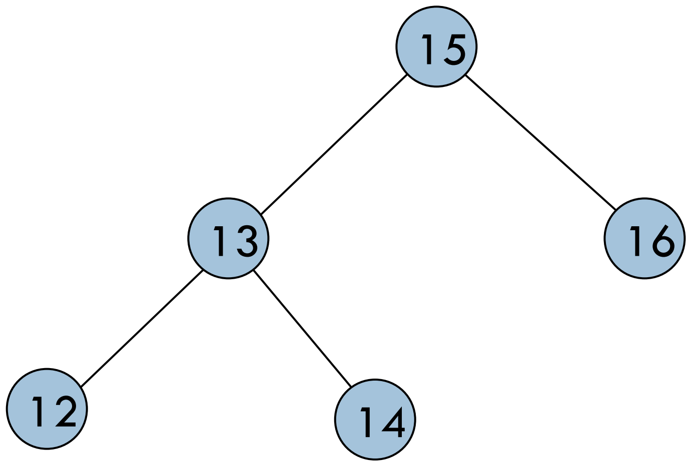
Single rotations: insert 14, 15, 16, 13, 12, 11, 10

- Rotation type:



AVL Tree Rotations

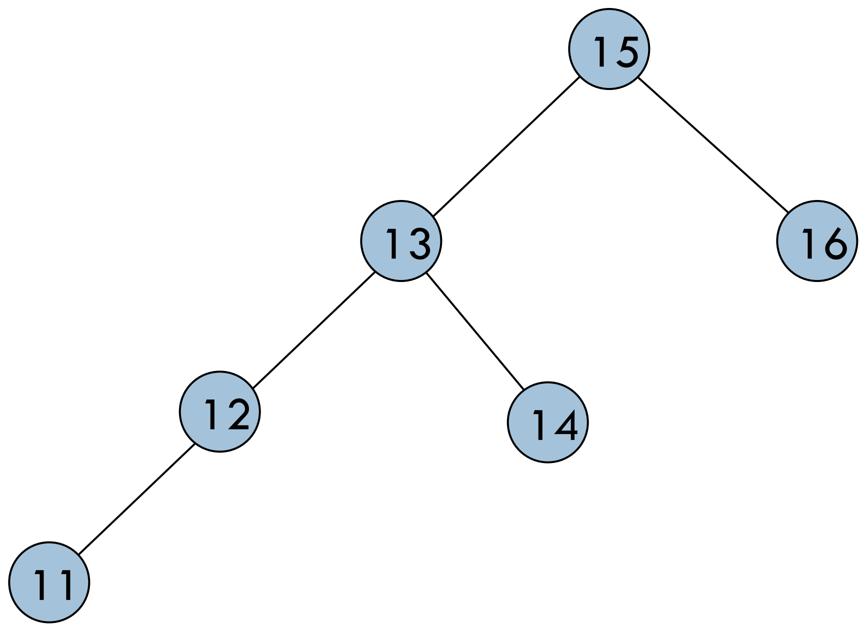
Single rotations: insert 14, 15, 16, 13, 12, 11, 10



- Now insert 11.

AVL Tree Rotations

Single rotations: insert 14, 15, 16, 13, 12, 11, 10

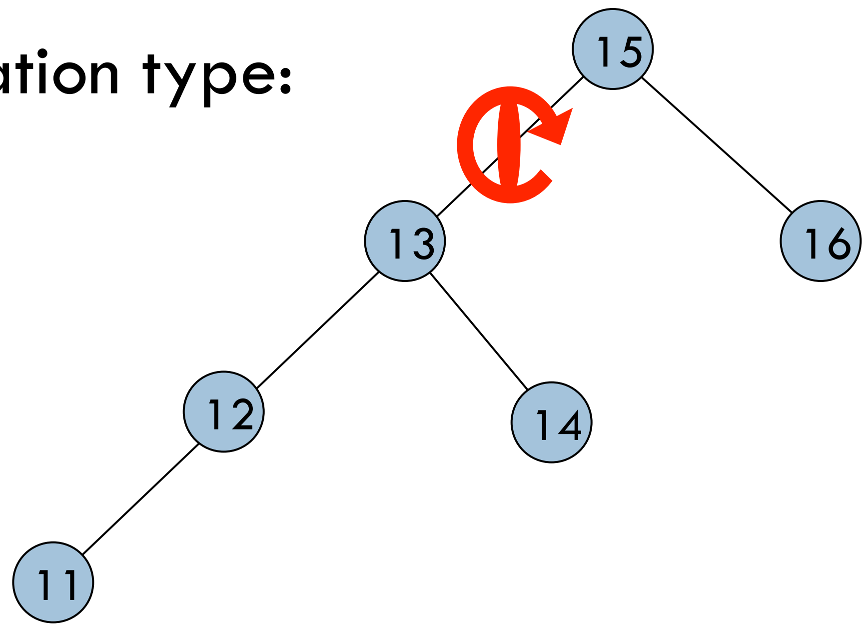


- AVL violation – need to rotate

AVL Tree Rotations

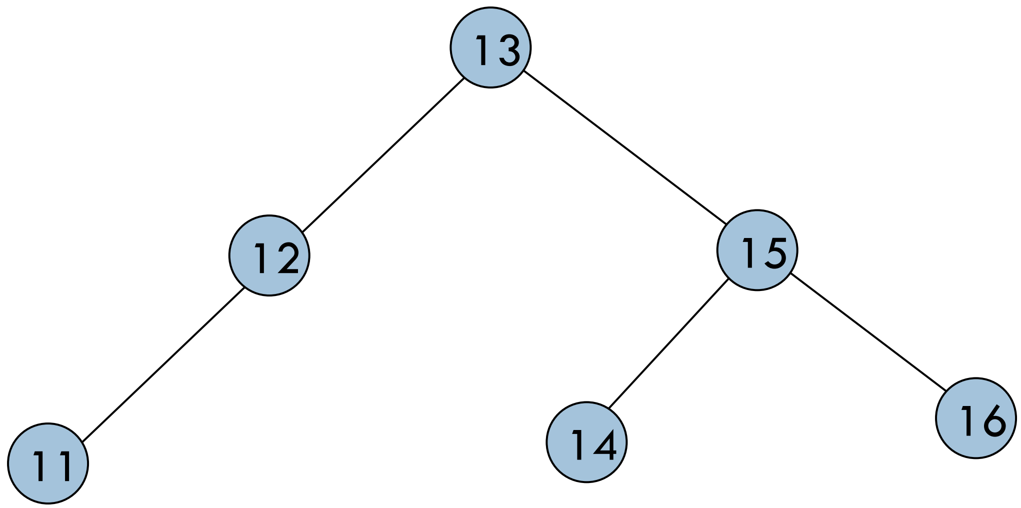
Single rotations: insert 14, 15, 16, 13, 12, 11, 10

- Rotation type:



AVL Tree Rotations

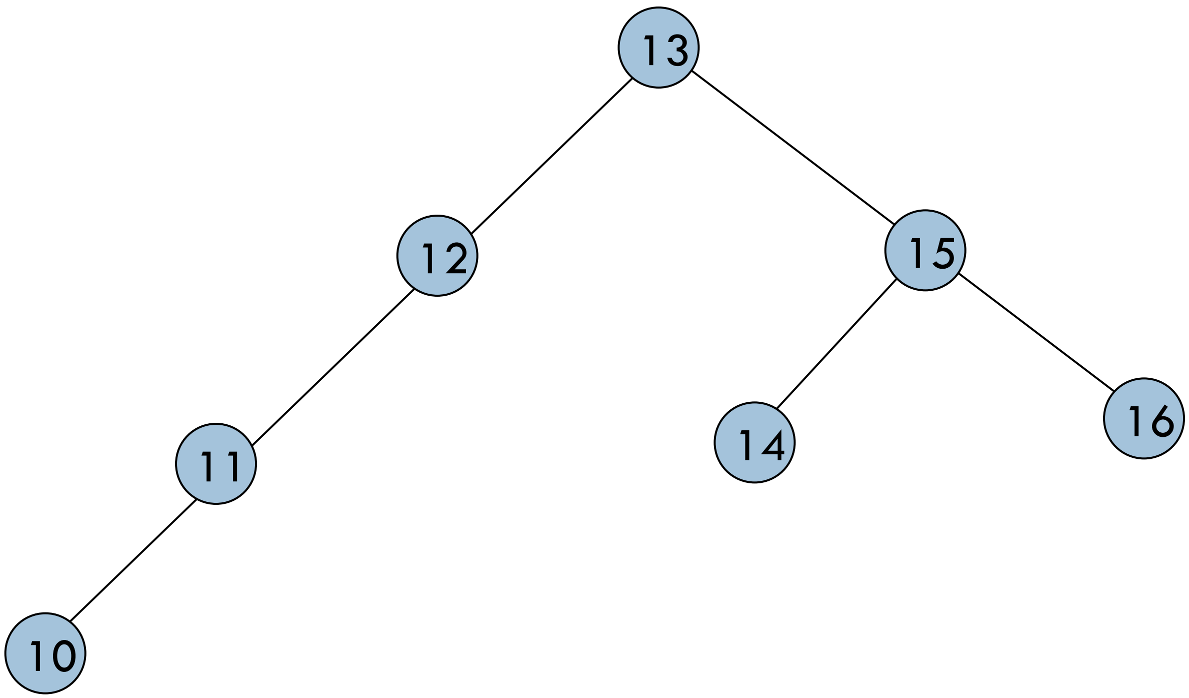
Single rotations: insert 14, 15, 16, 13, 12, 11, 10



- Now insert 10.

AVL Tree Rotations

Single rotations: insert 14, 15, 16, 13, 12, 11, 10

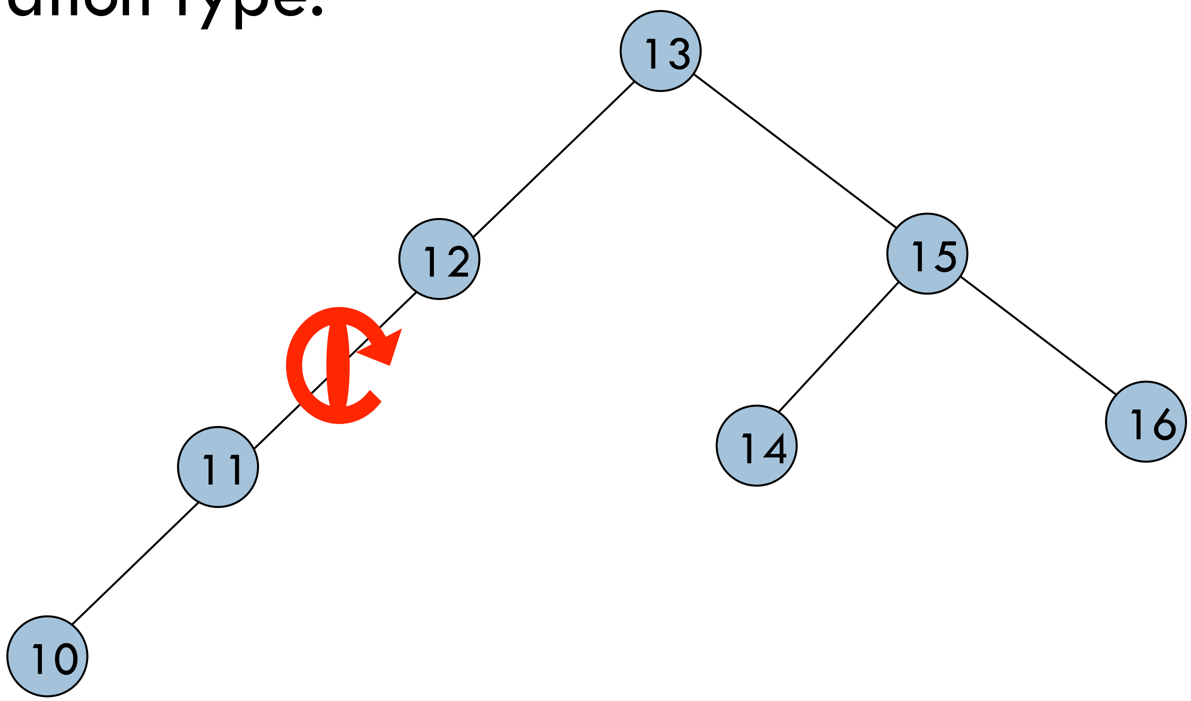


- AVL violation – need to rotate

AVL Tree Rotations

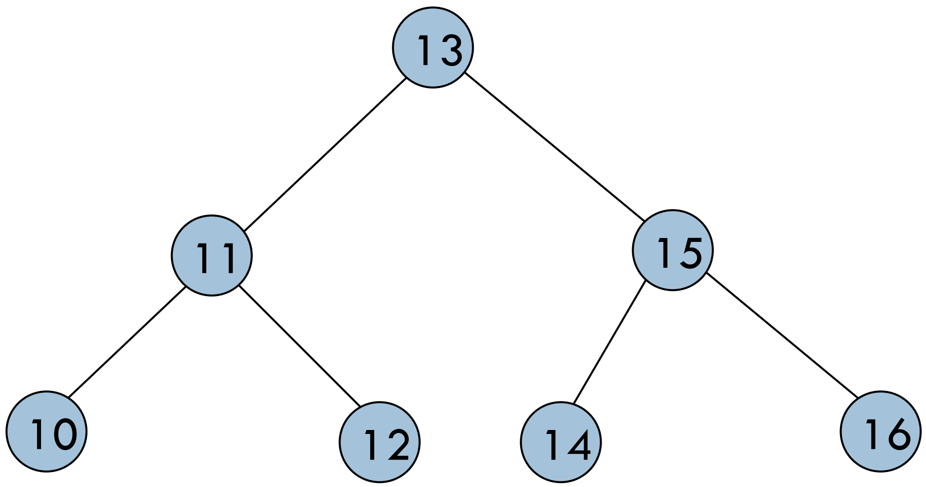
Single rotations: insert 14, 15, 16, 13, 12, 11, 10

- Rotation type:



AVL Tree Rotations

Single rotations: insert 14, 15, 16, 13, 12, 11, 10

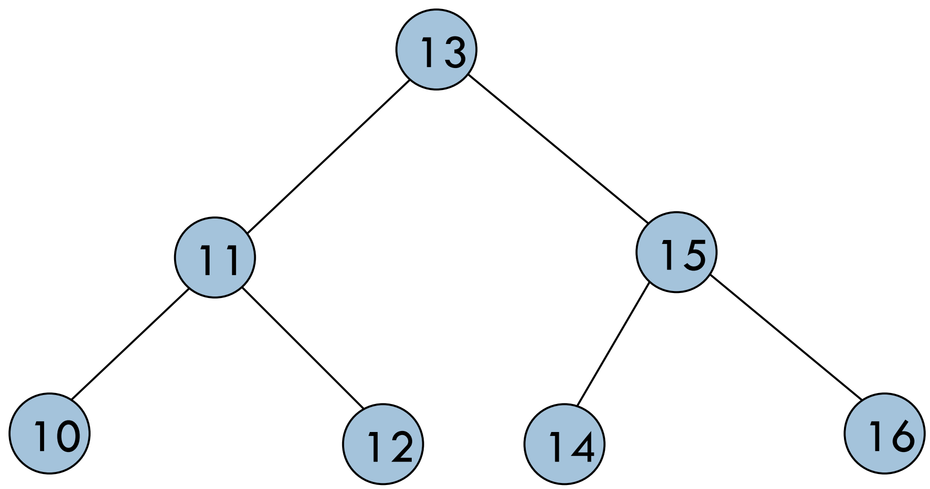


- AVL balance restored.

AVL Tree Rotations

Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8

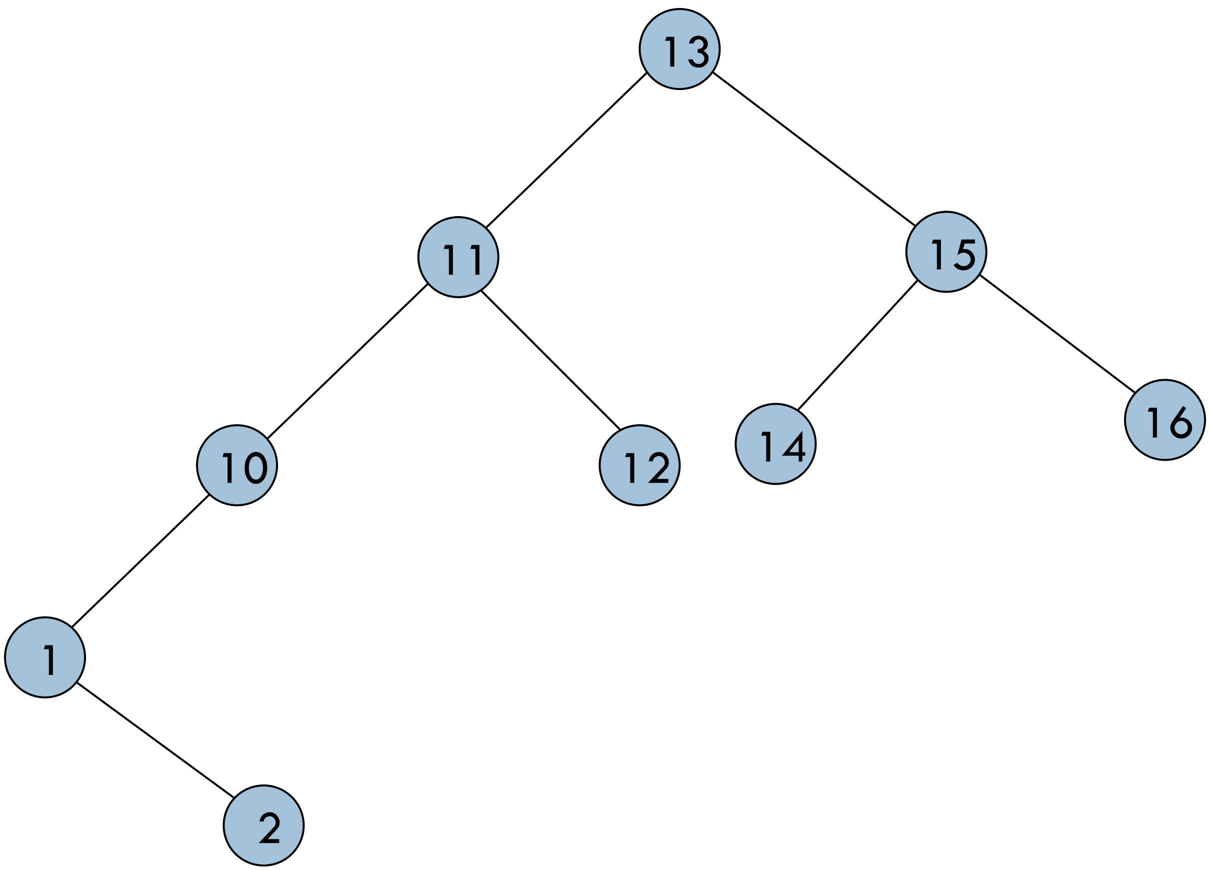
- First insert 1 and 2:



AVL Tree Rotations

Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8

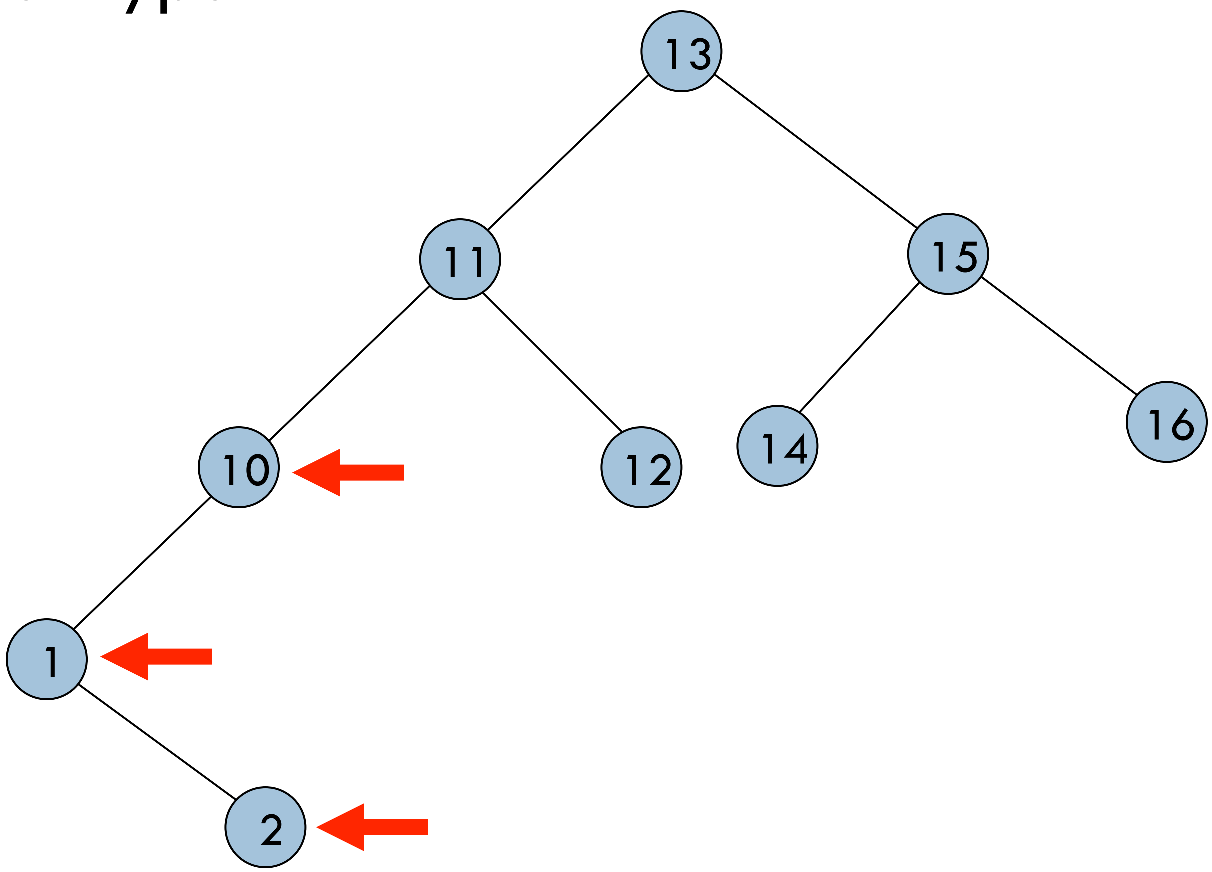
- AVL violation - rotate



AVL Tree Rotations

- Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8

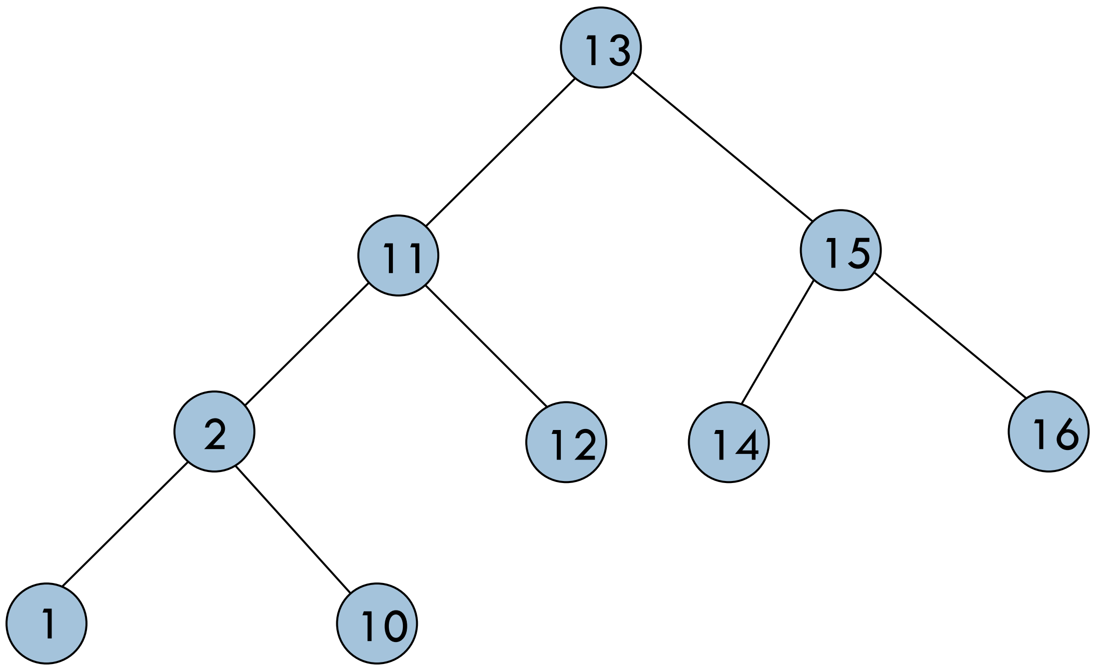
- Rotation type:



AVL Tree Rotations

- Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8

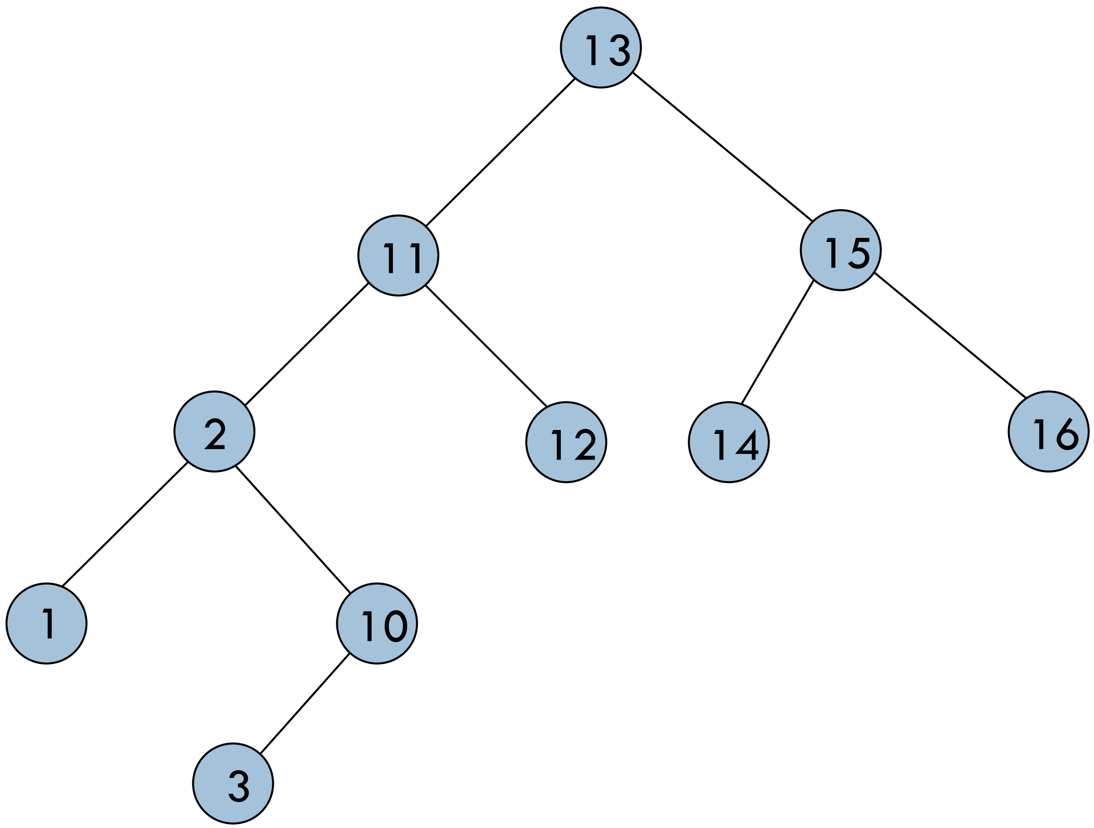
• AVL balance restored:



• Now insert 3.

AVL Tree Rotations

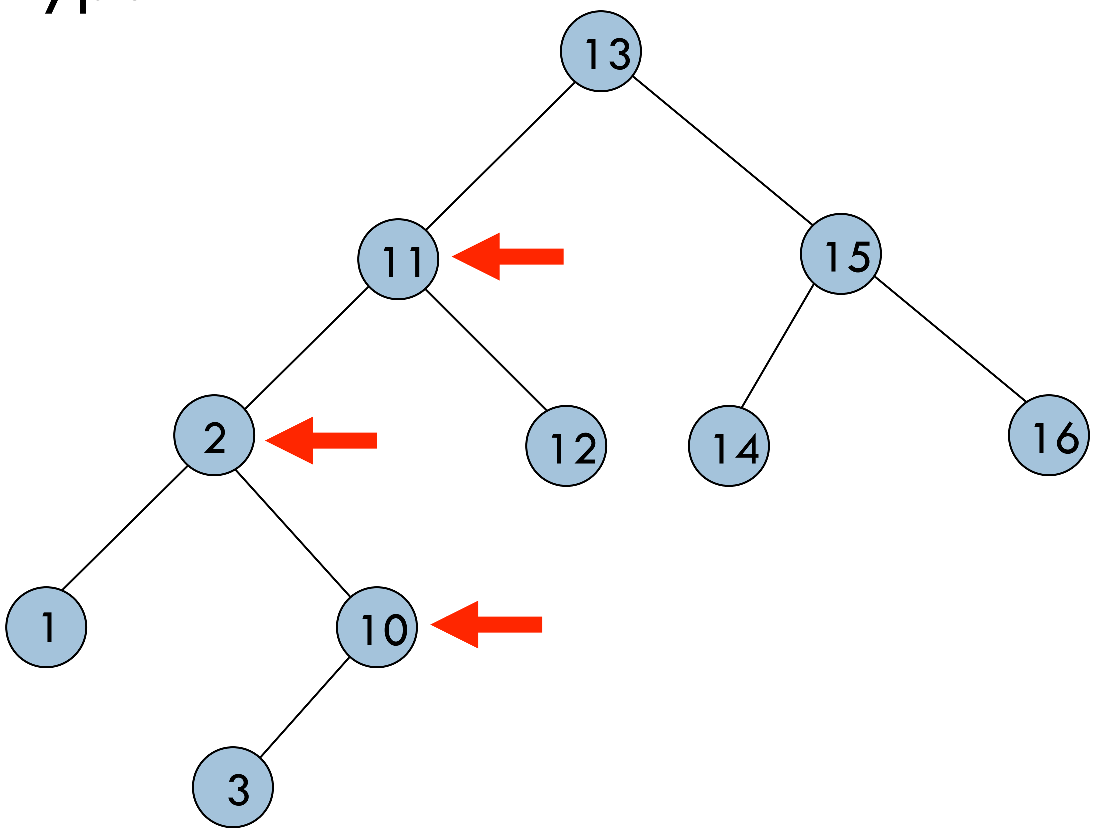
- Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8
  - AVL violation – rotate:



AVL Tree Rotations

- Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8

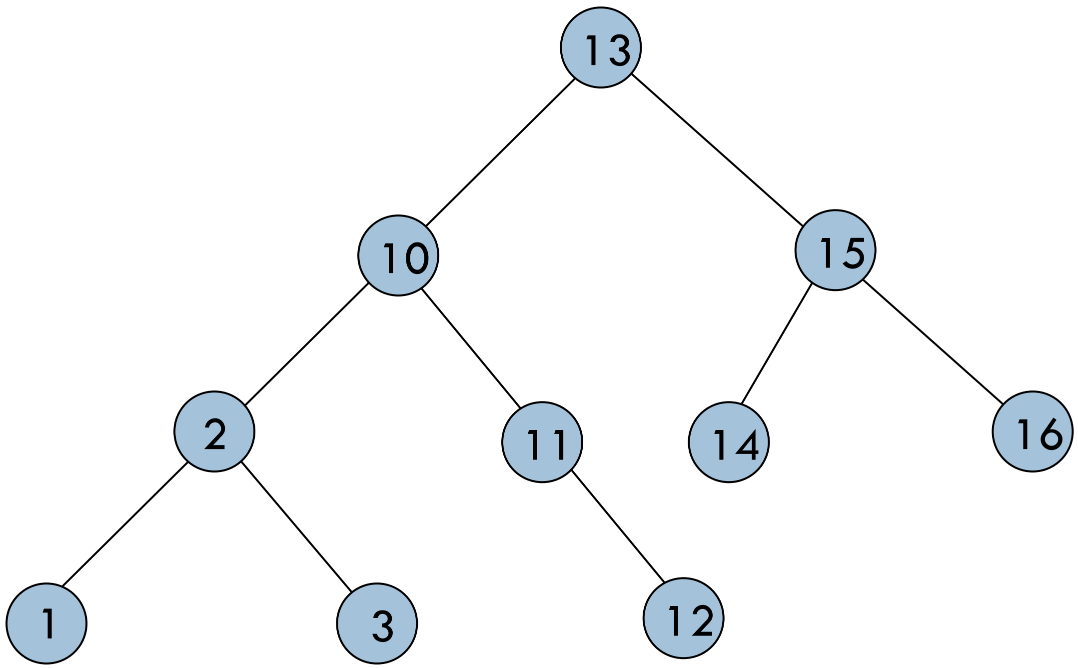
• Rotation type:



AVL Tree Rotations

- Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8

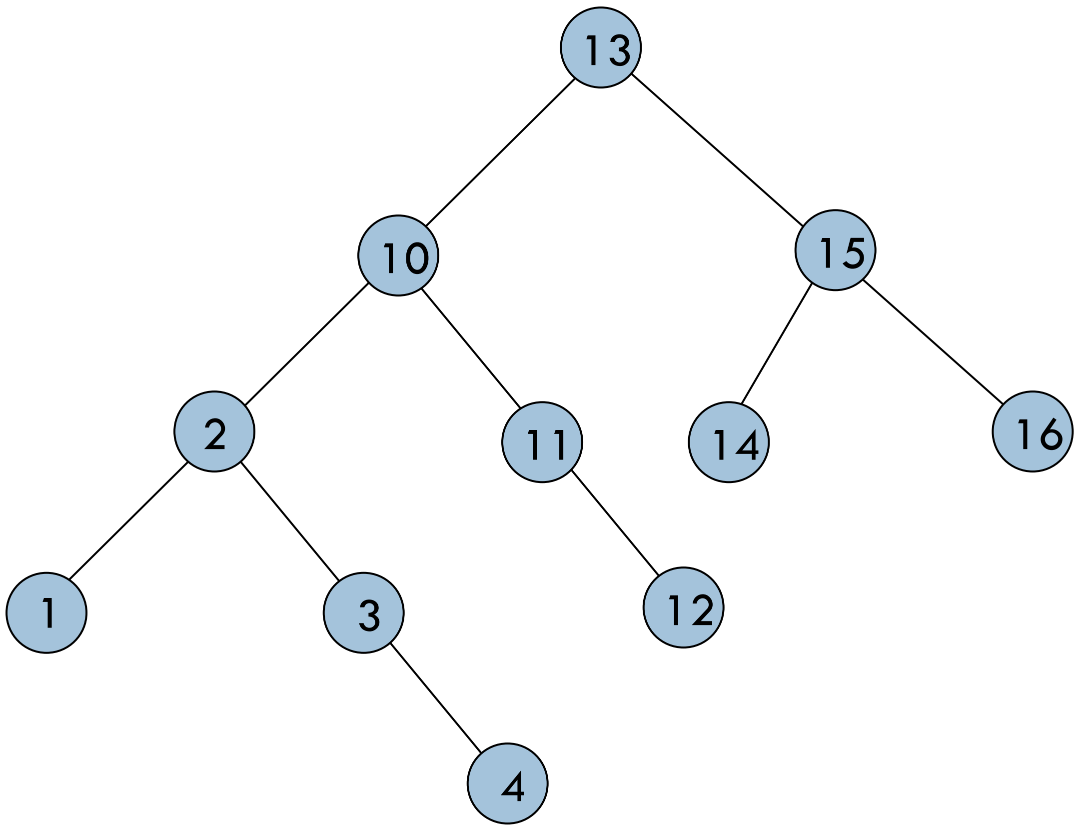
- AVL balance restored:



- Now insert 4.

AVL Tree Rotations

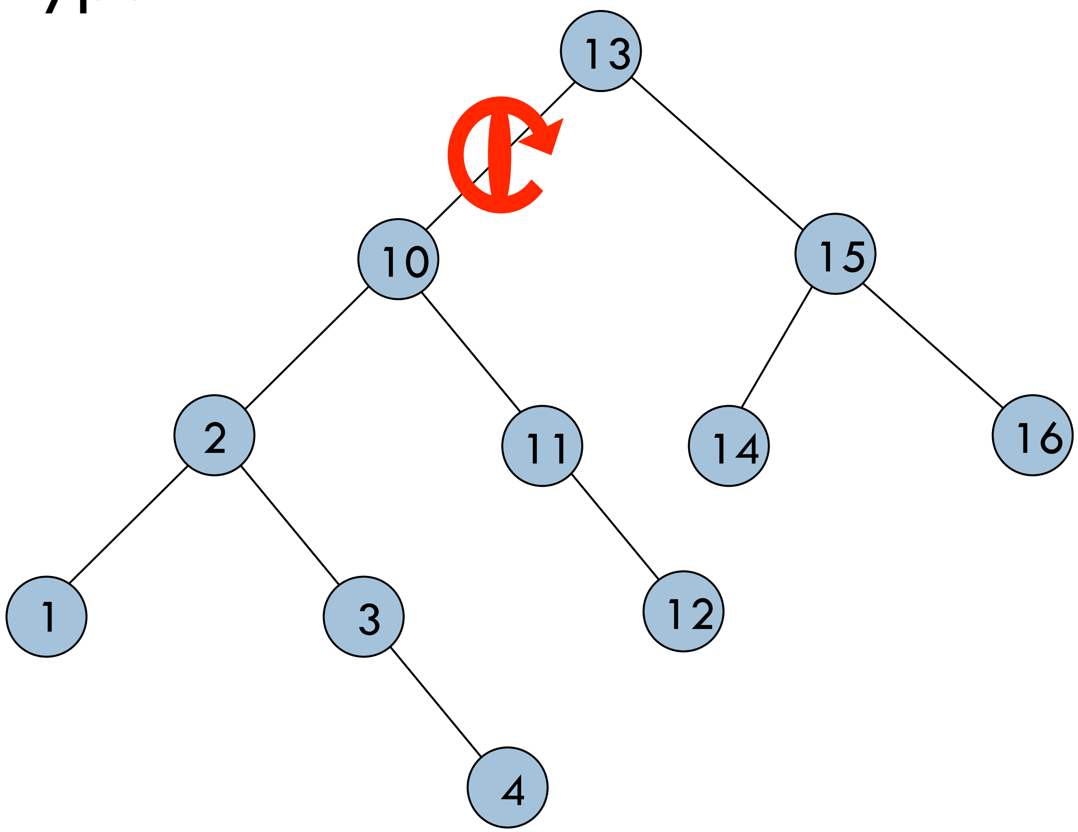
- Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8
  - AVL violation - rotate



AVL Tree Rotations

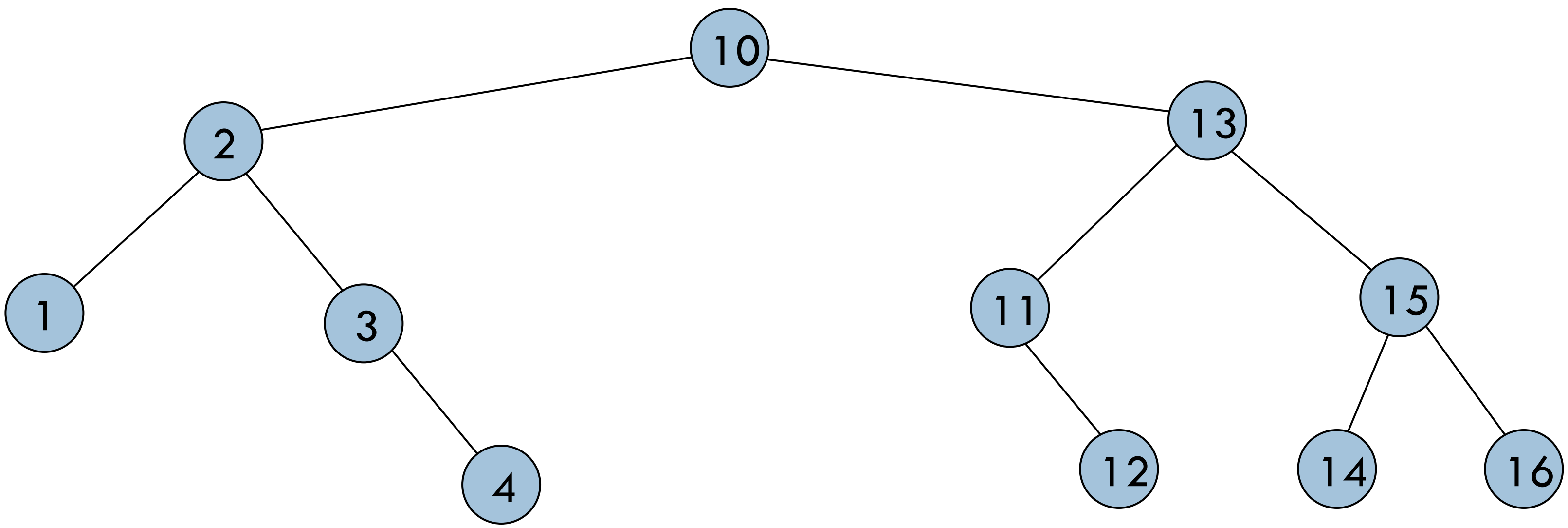
- Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8

• Rotation type:



AVL Tree Rotations

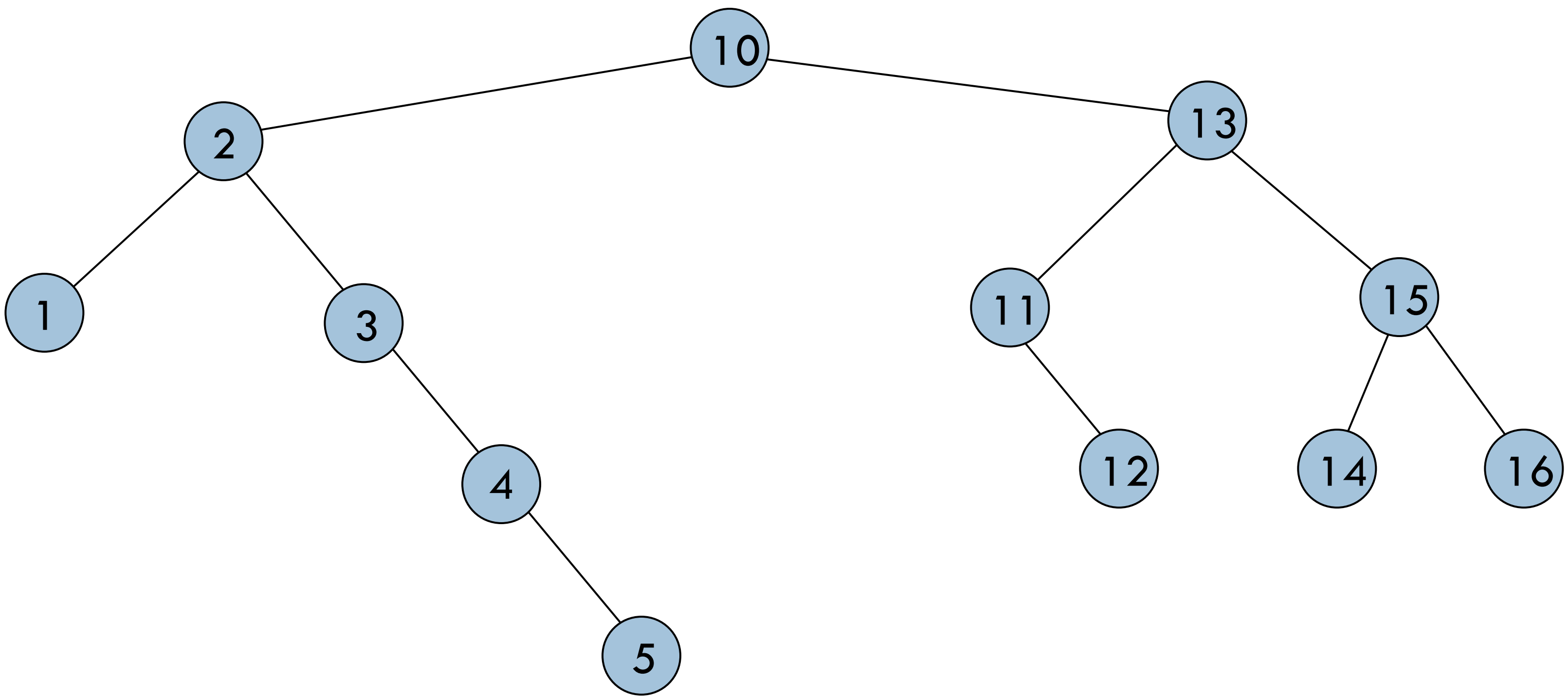
- Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8



- Now insert 5.

AVL Tree Rotations

- Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8

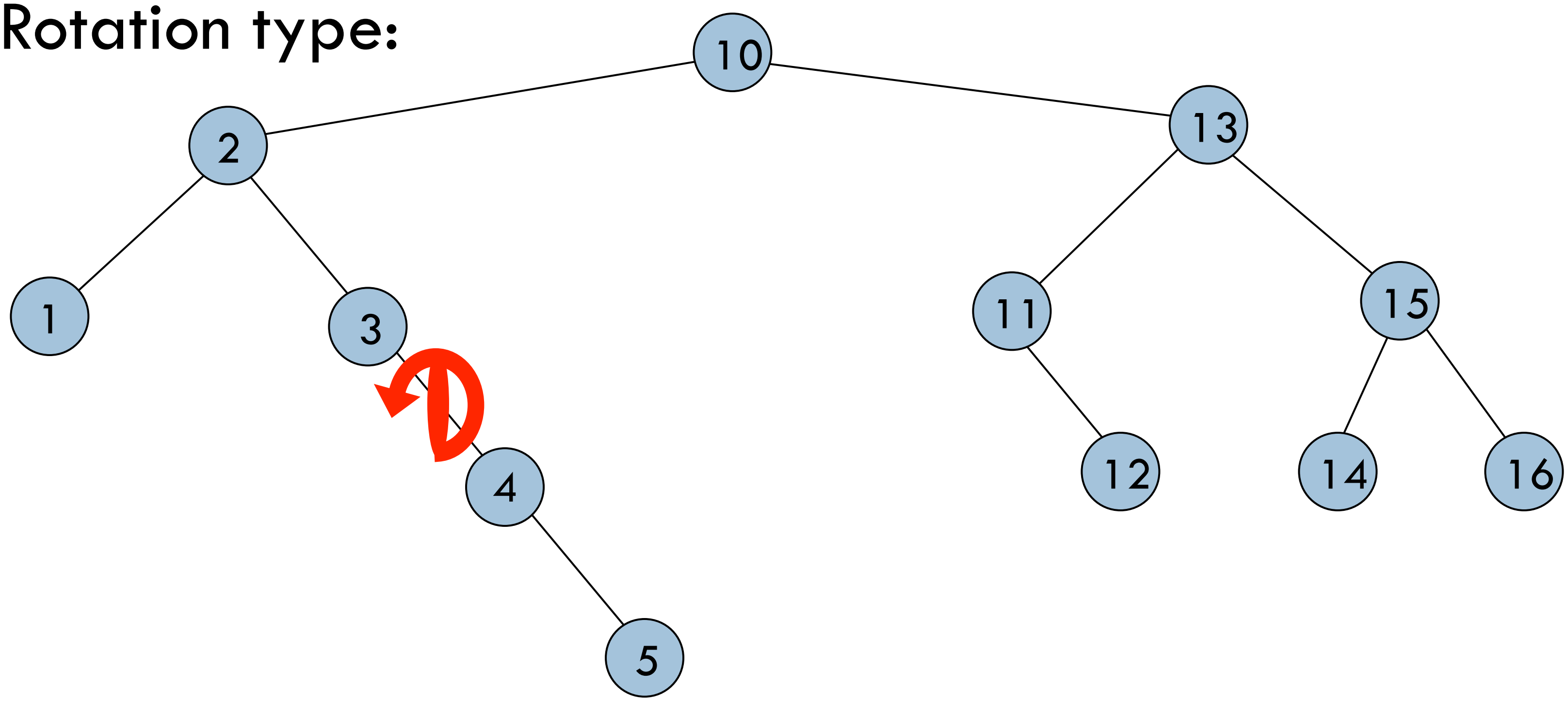


- AVL violation – rotate.

AVL Tree Rotations

- Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8

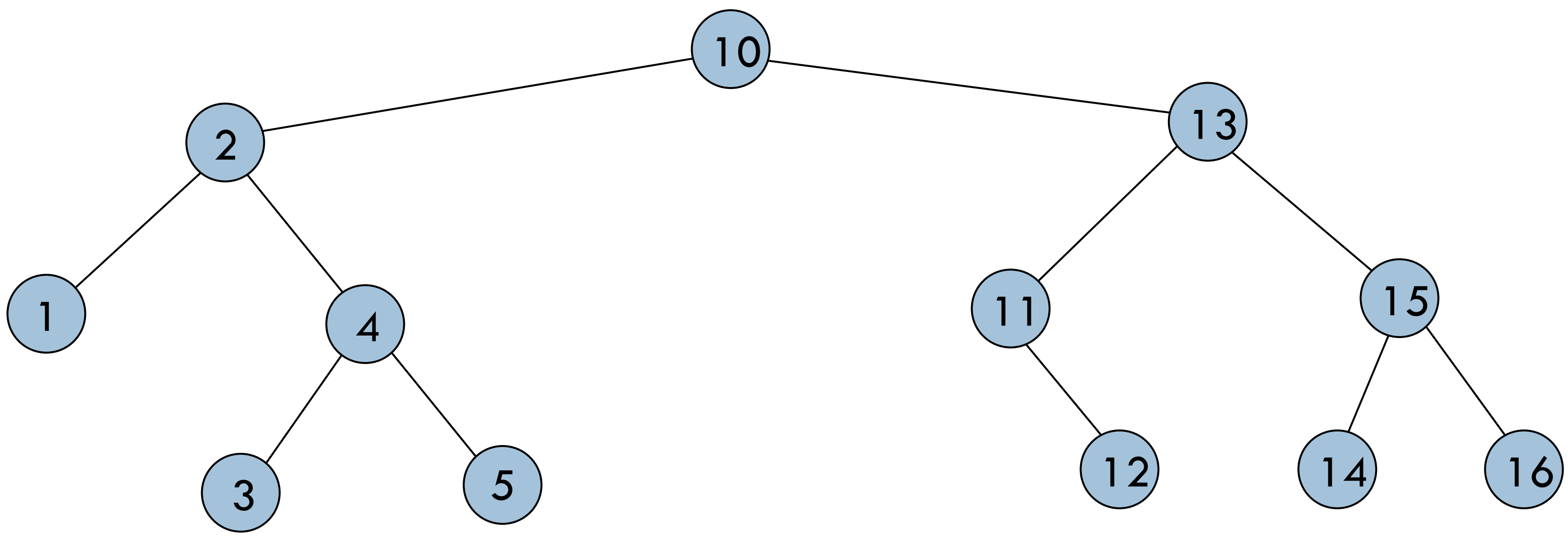
• Rotation type:



AVL Tree Rotations

- Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8

- AVL balance restored:

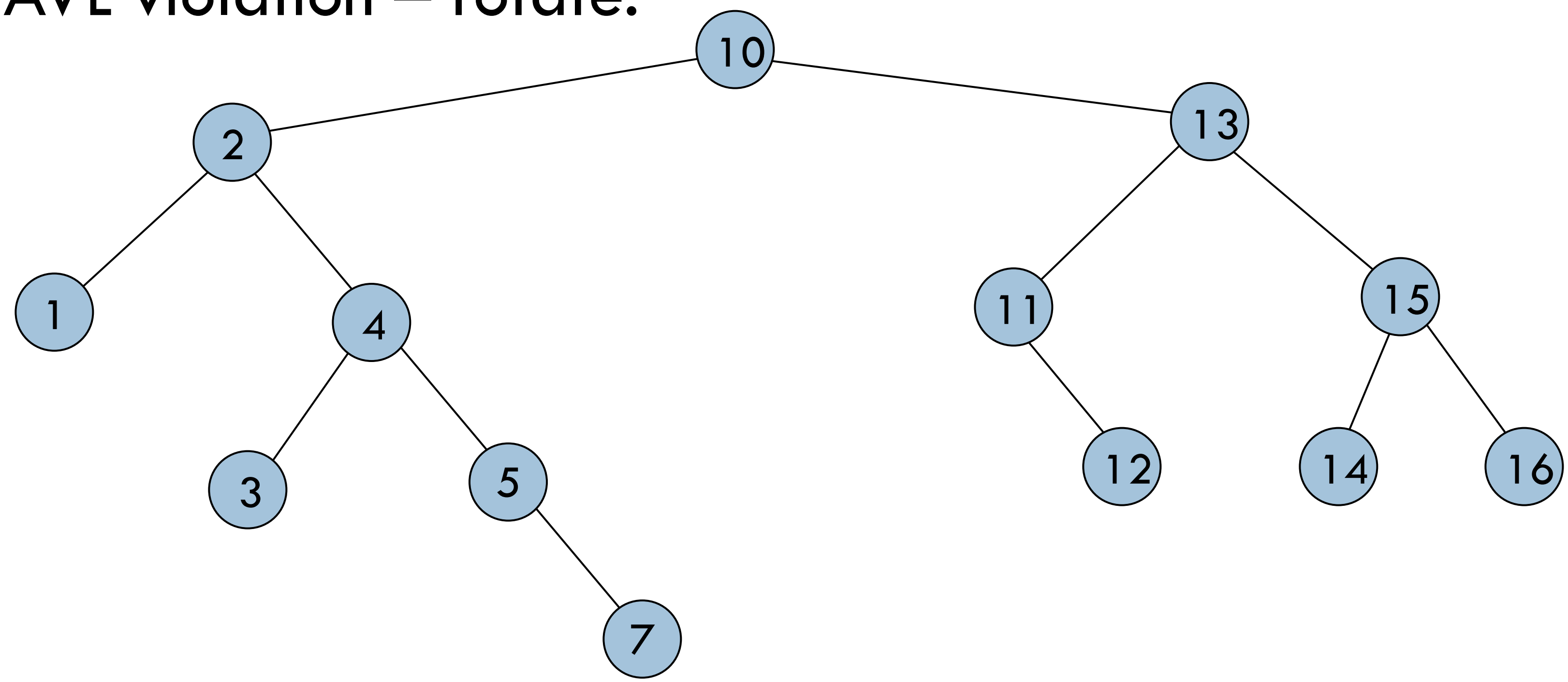


- Now insert 7.

AVL Tree Rotations

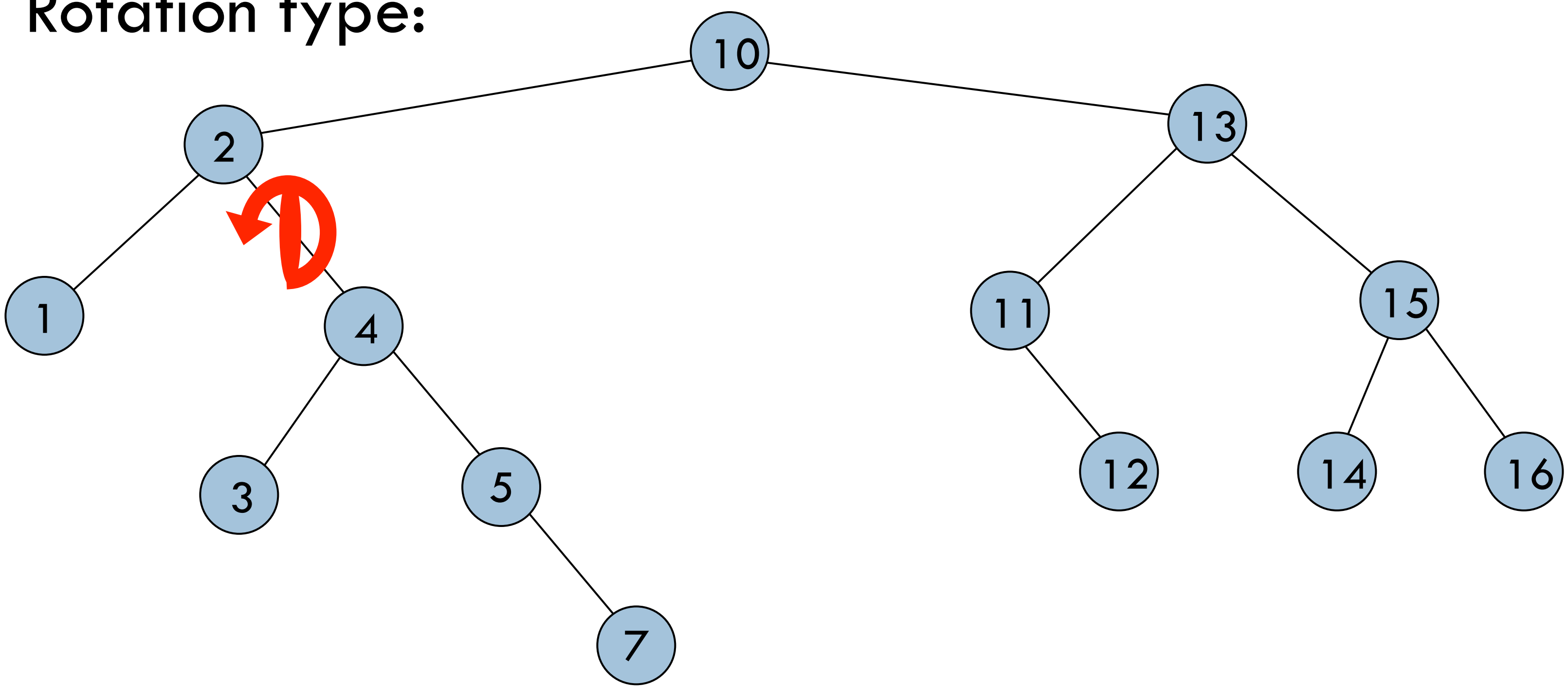
- Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8

• AVL violation – rotate.



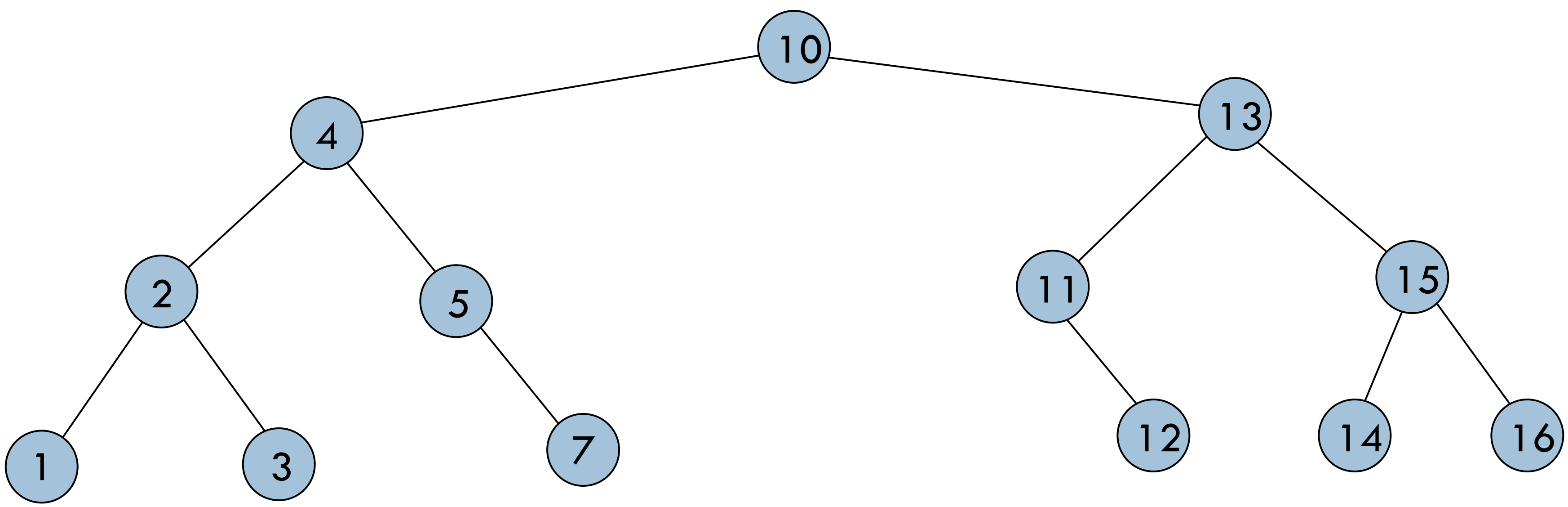
AVL Tree Rotations

- Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8
- Rotation type:



AVL Tree Rotations

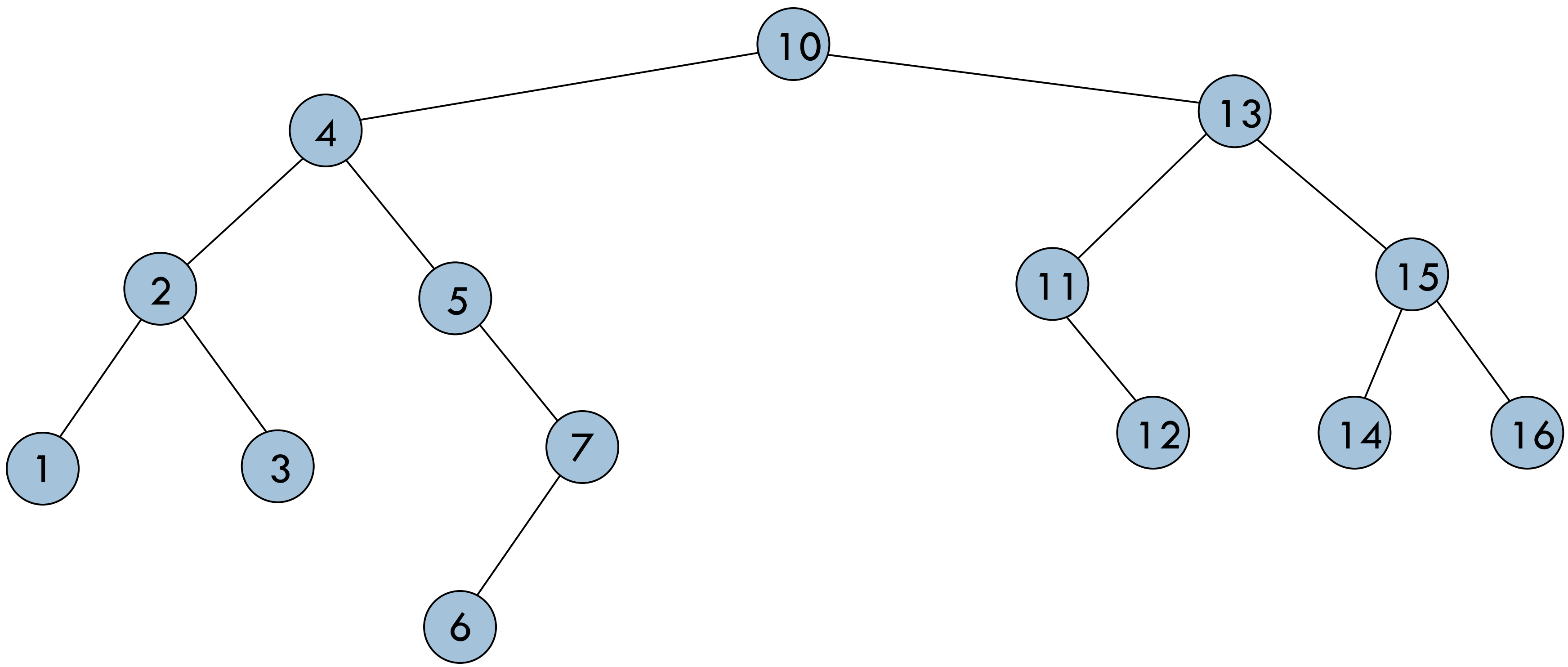
- Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8
  - AVL balance restored.



- Now insert 6.

AVL Tree Rotations

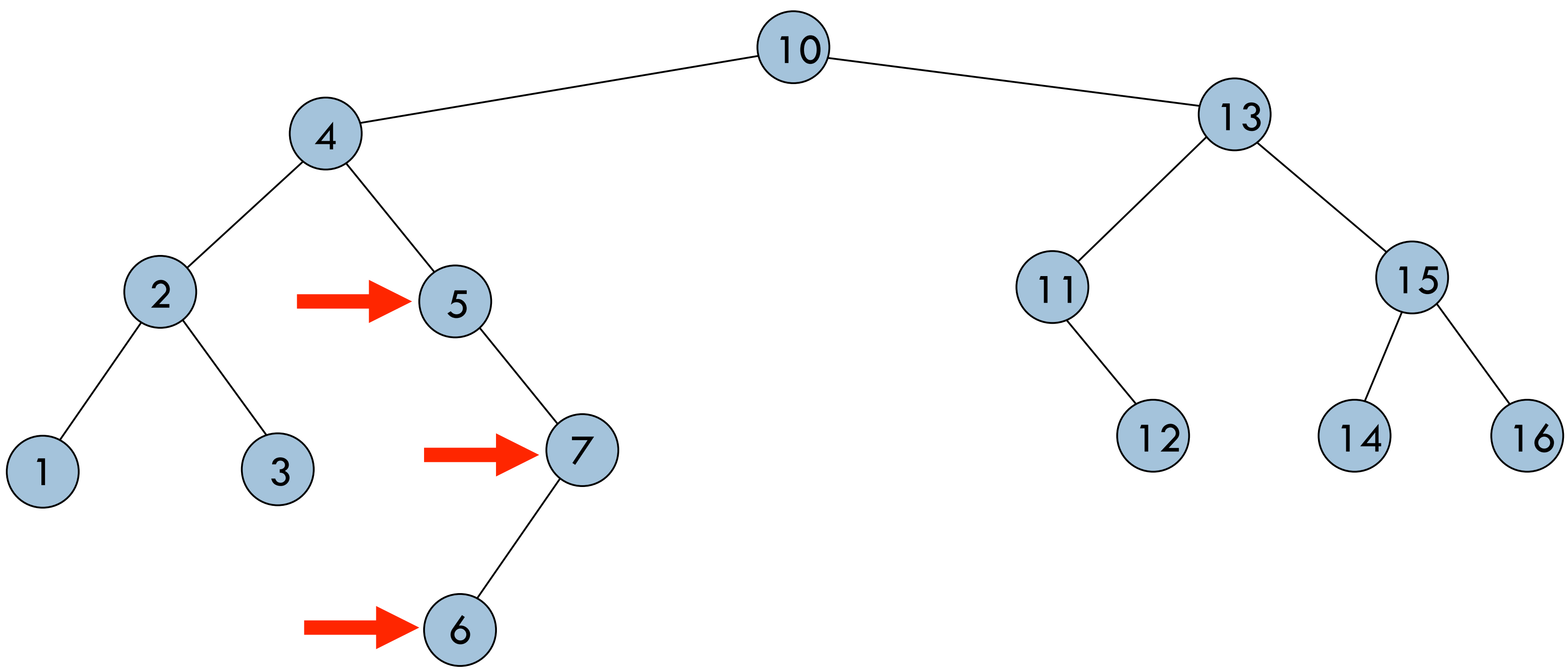
- Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8
  - AVL violation - rotate.



AVL Tree Rotations

- Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8

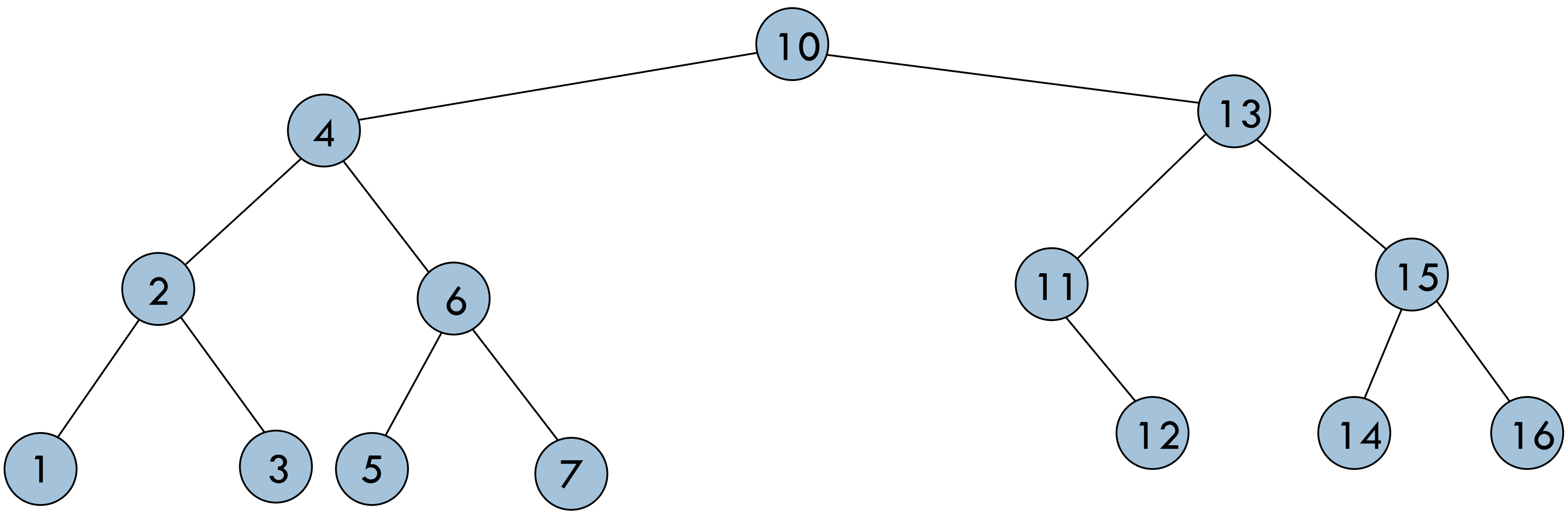
- Rotation type:



AVL Tree Rotations

- Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8

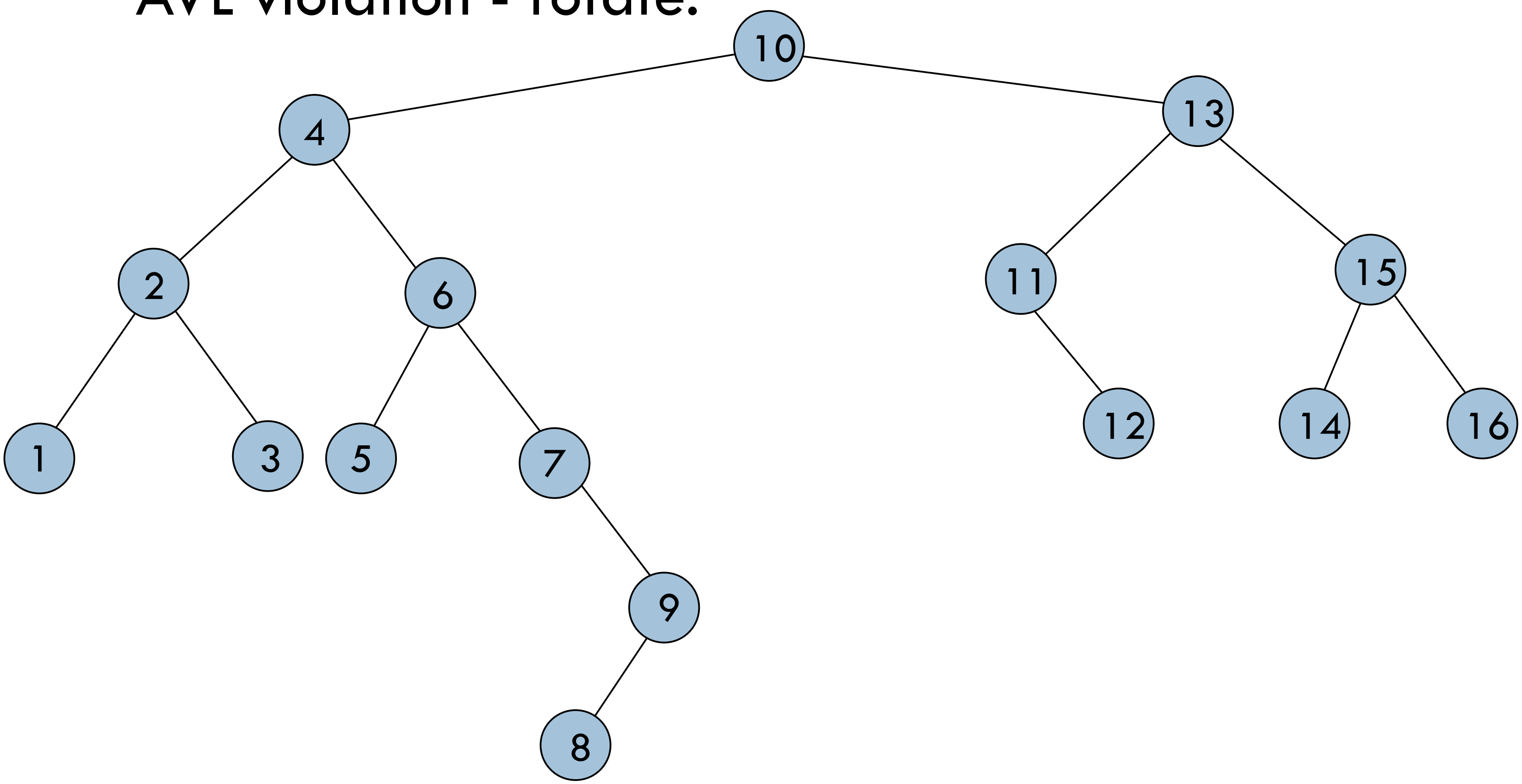
- AVL balance restored.



- Now insert 9 and 8.

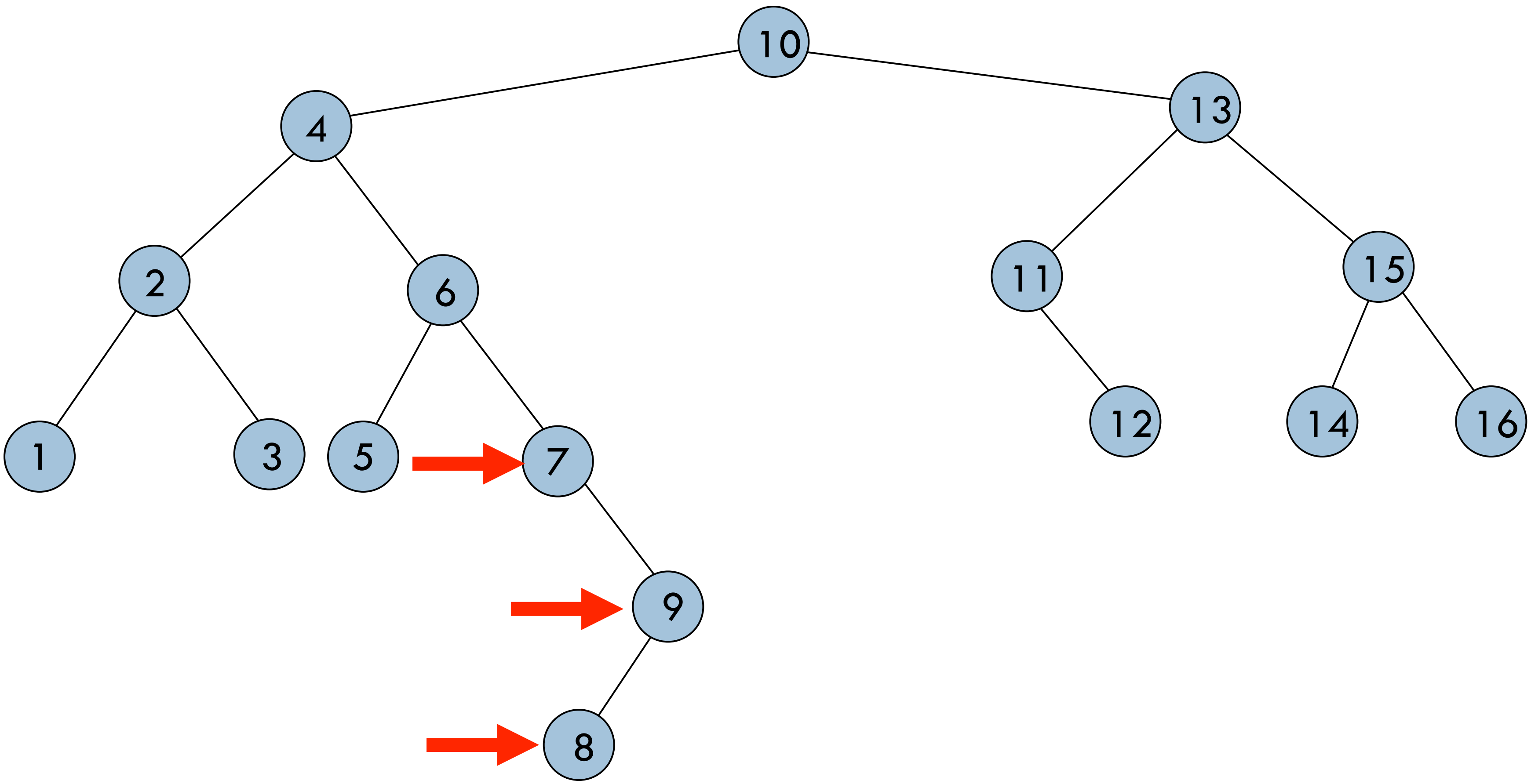
AVL Tree Rotations

- Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8
  - AVL violation - rotate.



AVL Tree Rotations

- Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8
- Rotation type:



AVL Tree Rotations

- Double rotations: insert 1, 2, 3, 4, 5, 7, 6, 9, 8

- Tree is almost perfectly balanced