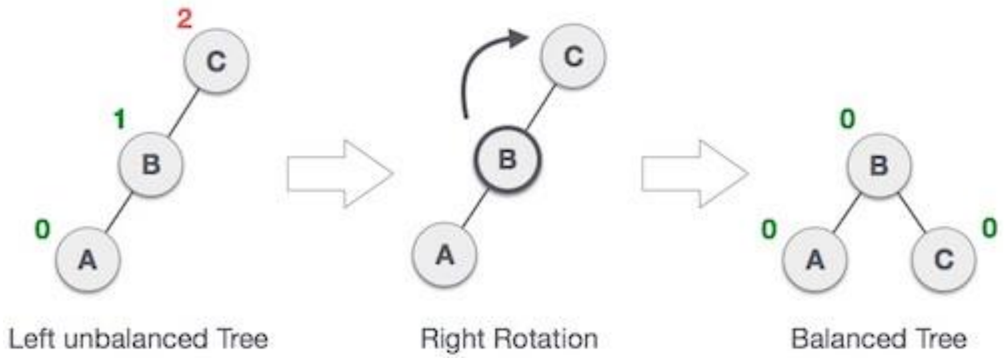
01418231 Data Structure

AVL Tree



https://www.tutorialspoint.com/data_structures_algorithms/avl_tree_algorithm.htm

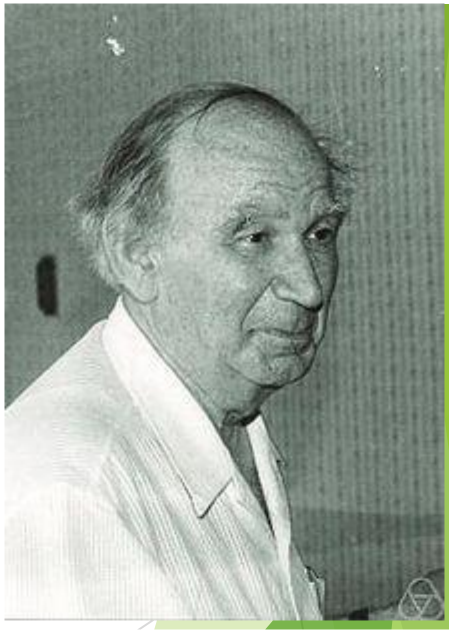
Agenda

- What’s AVL tree?
- What’ s the balance methods?
- How to rotate AVL tree?
- Summary

AVL tree

- The AVL tree is named after its two Soviet inventors, Georgy Adelson-Velsky and Evgenii Landis, who published it in their 1962 paper "An algorithm for the organization of information".

Adelson-Velsky in Moscow 1980



Evgenii Landis at conference on potential theory in Prague, 1987

https://en.wikipedia.org/wiki/AVL_tree

What’s AVL tree?

Binary Search trees

- BST maintain a reasonablee balanced tree all the time.
- *Key idea:* if insertion or deletion get the tree out of balance then fix it immediately
- All operations insert, delete can be done on an AVL tree

What ‘s the balance methods?

AVL TREES (Adelson-Velskii and Landis 1962)

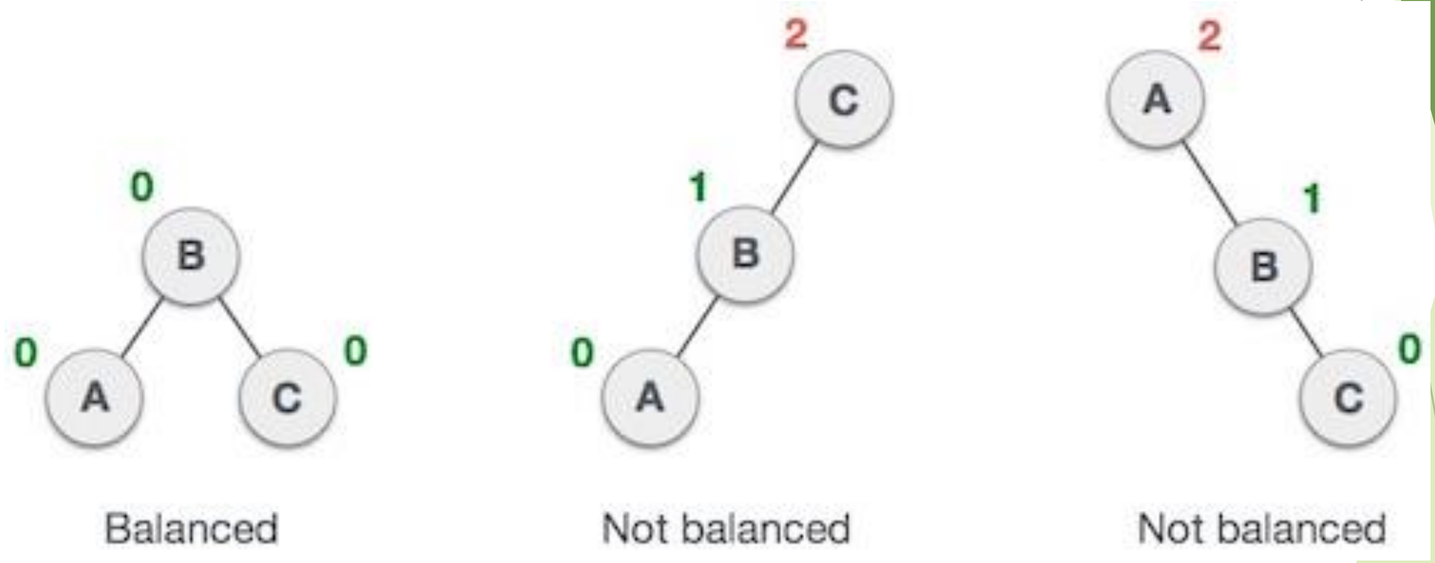
AVL Tree Property: **It is a BST in which the heights of the left and right subtrees of the root differ by at most 1 and in which the right and left subtrees are also AVL trees**

$$|H_{Left} - H_{Right}| \leq 1$$

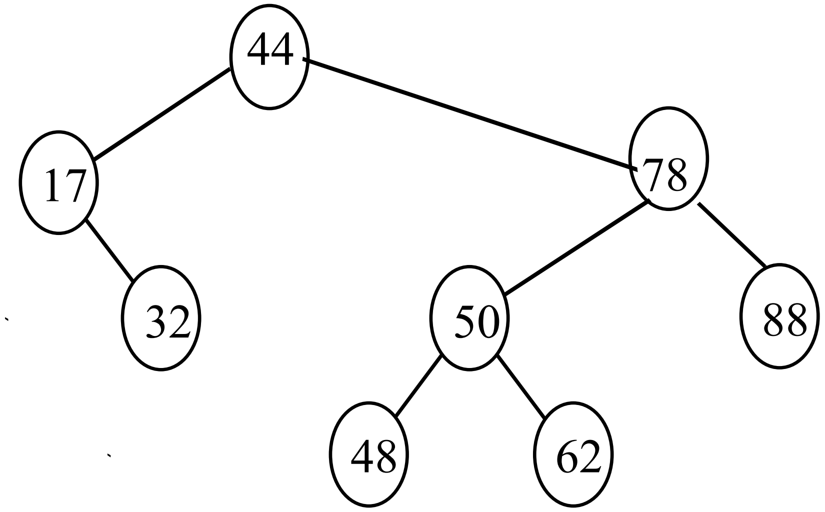
[-1, 0, +1]

Height: **length of the longest path from the root to a leaf**

Balanced of AVL Tree

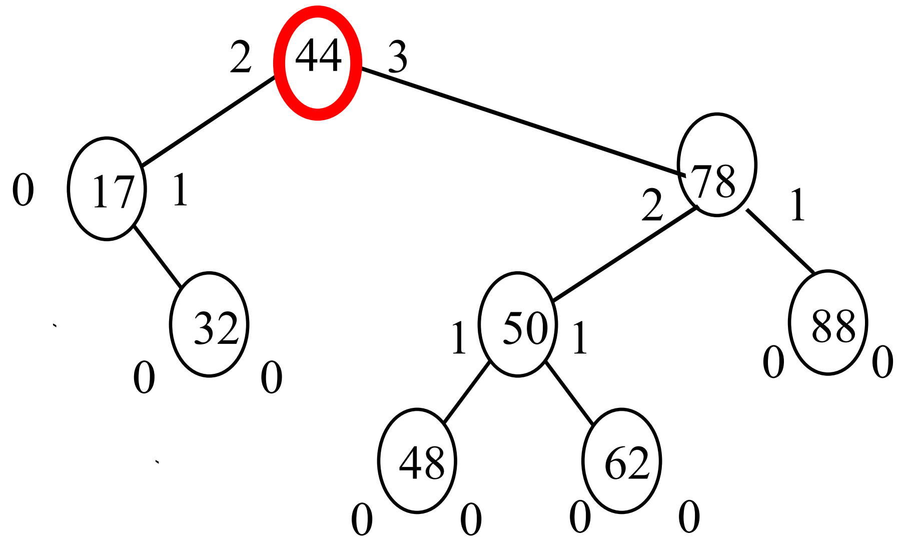


Example: Balanced of AVL Tree



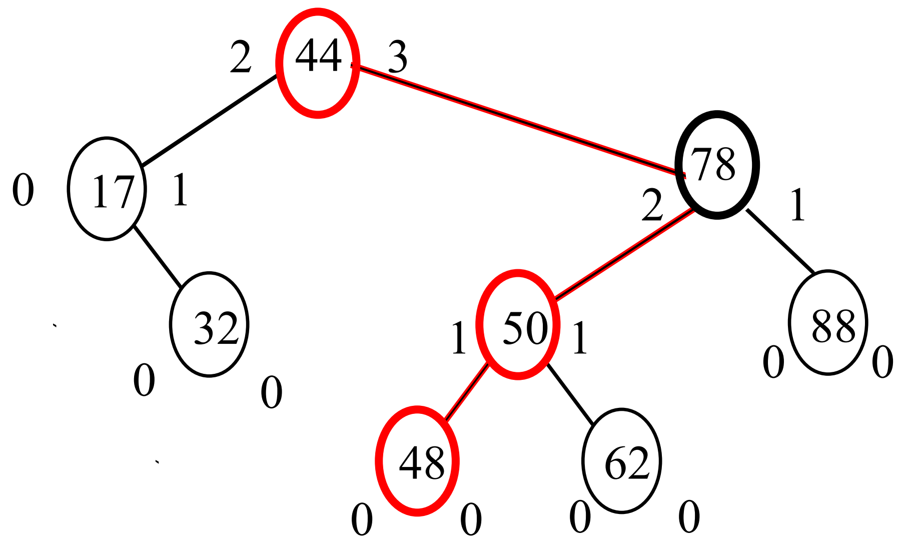
An example of an AVL tree where the heights are shown next to the nodes:

Example: Balanced of AVL Tree



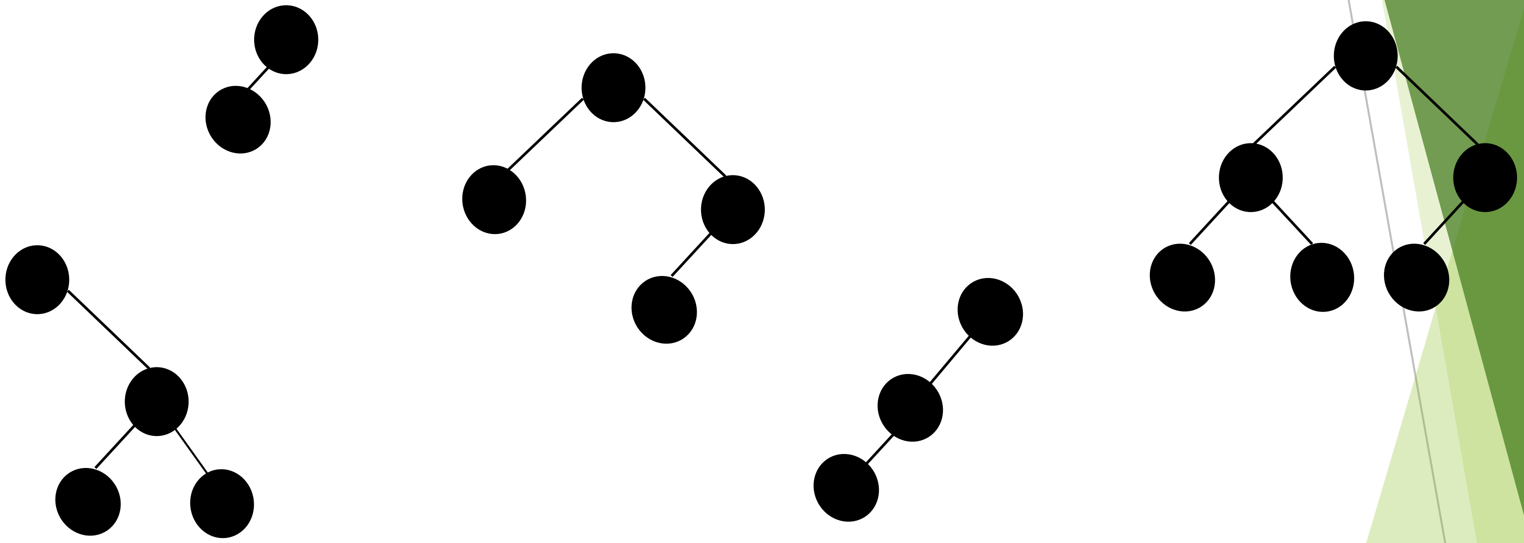
An example of an AVL tree where the heights are shown next to the nodes:

Example: Balanced of AVL Tree

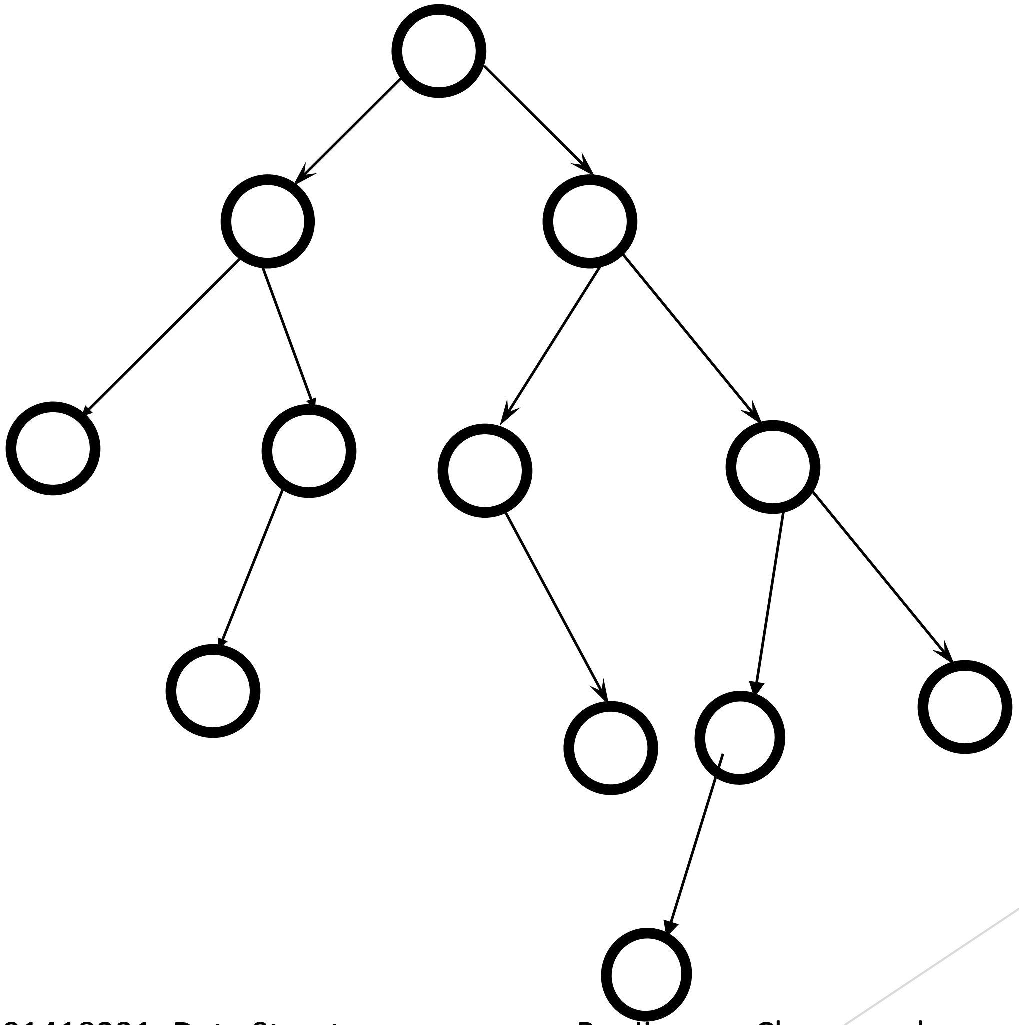


An example of an AVL tree where the heights are shown next to the nodes:

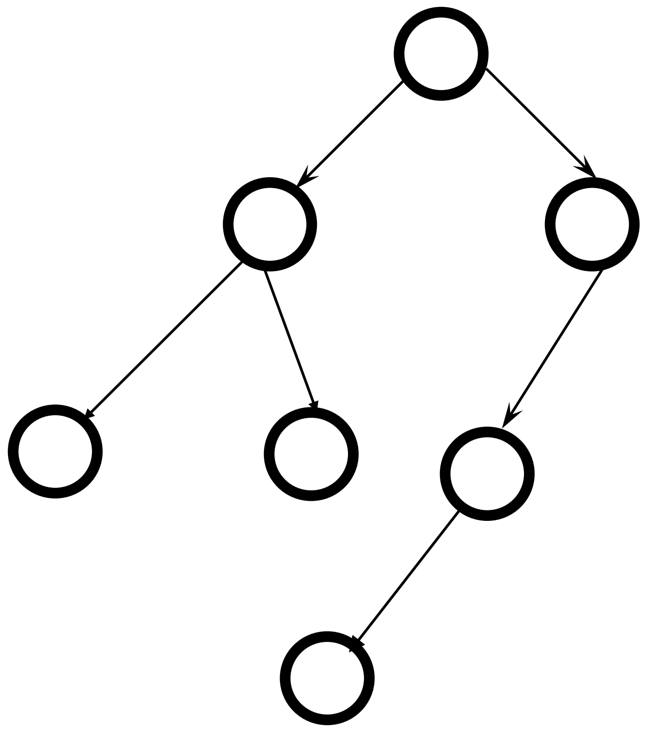
Example: Balanced of AVL Tree



Example: Balanced of AVL Tree



Example: Balanced of AVL Tree



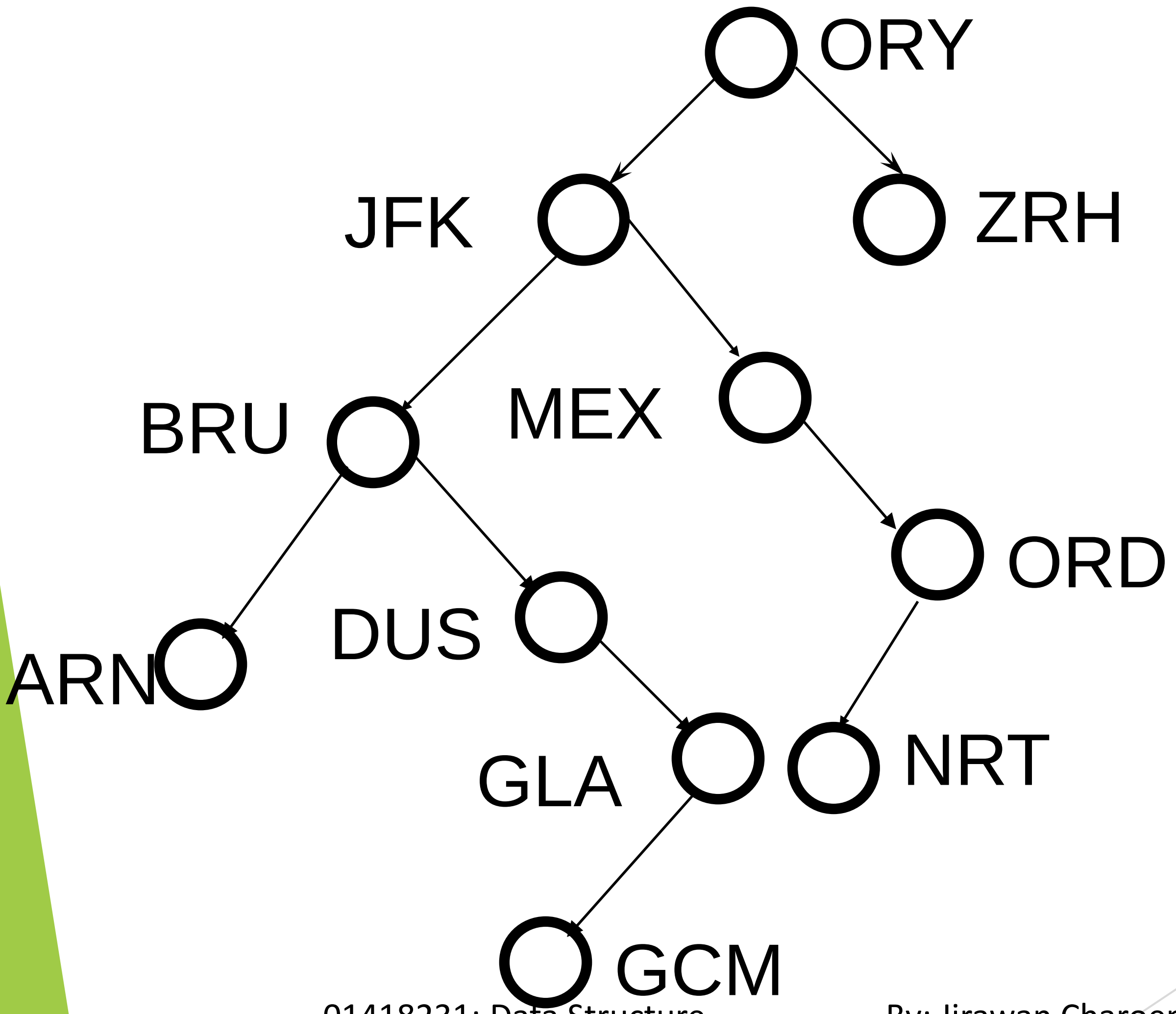
Operator

Insert, Delete

Example: AVL Tree for Airports

- Consider inserting sequentially:
  - ORY, JFK, BRU, DUS, ZRX, MEX, ORD, NRT, ARN, GLA, GCM
- Build a binary-search tree
- Build a AVL tree.

Binary Search Tree for Airport Names

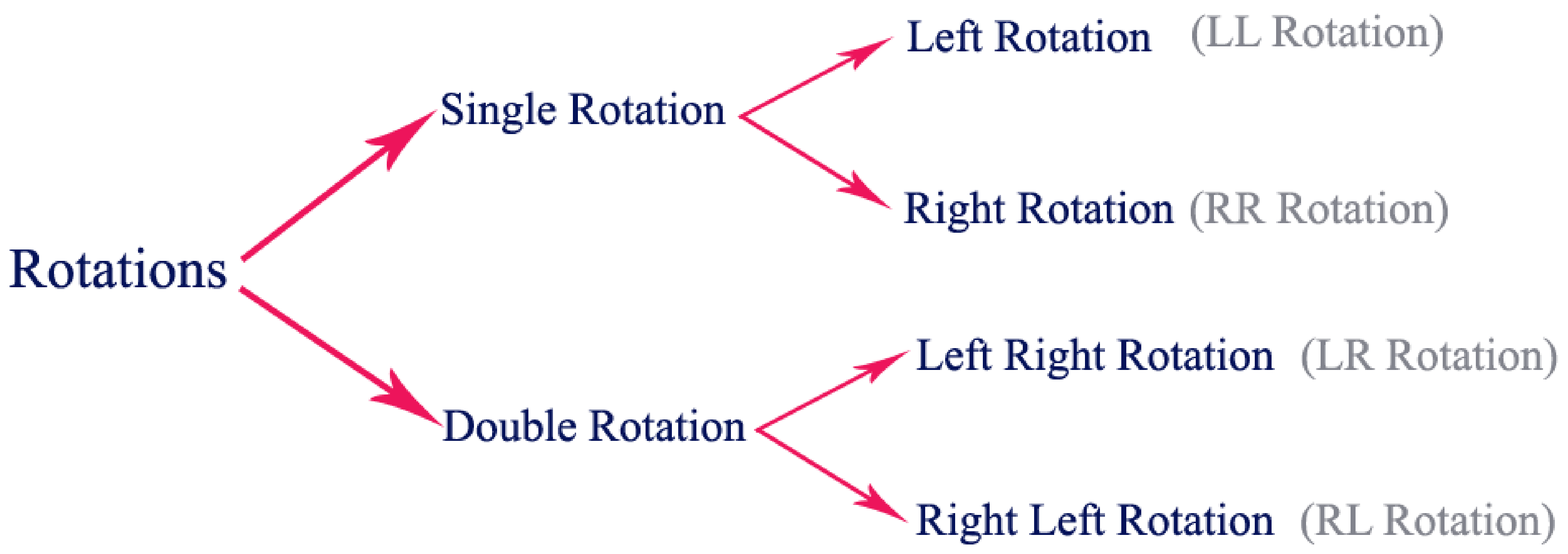


How does the AVL tree work?

- After insertion and deletion we will examine the tree structure and see if any node violates the AVL tree property
  - If the AVL property is violated, it means the heights of left(x) and right(x) differ by exactly 2
- If it does violate the property we can modify the tree structure using "rotations" to restore the AVL tree property

Imbalance of AVL tree

Type of AVL Rotations

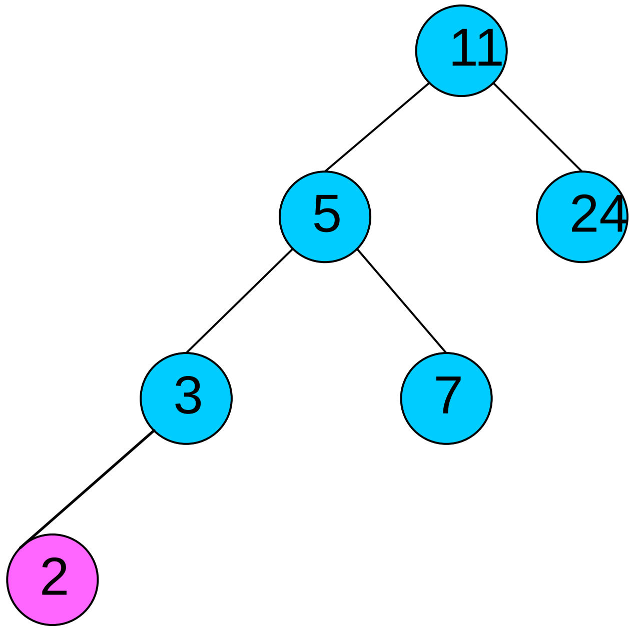

Rotations
Single Rotation
Double Rotation
Left Rotation (LL Rotation)
Right Rotation (RR Rotation)
Left Right Rotation (LR Rotation)
Right Left Rotation (RL Rotation)

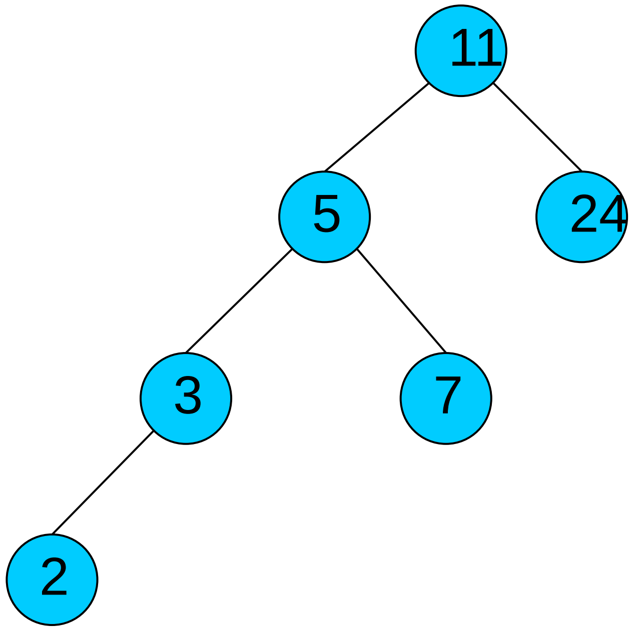
Type of AVL Rotations

- Left rotation
- Right rotation
- Left-Right rotation
- Right-Left rotation
- Remark https://www.tutorialspoint.com/data_structures_algorithms/avl_tree_algorithm.htm
- http://btechsmartclass.com/DS/U5_T2.html

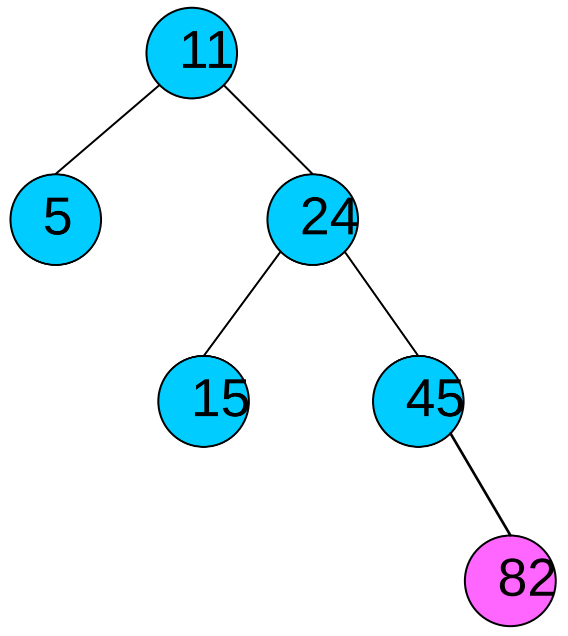
Example 1

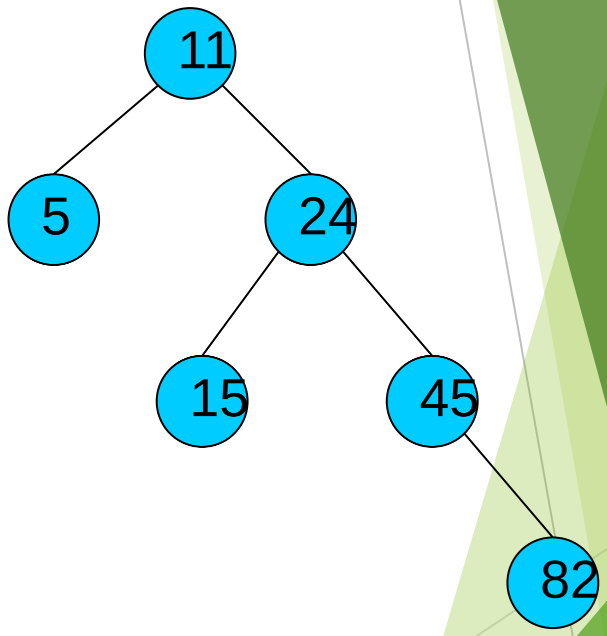
Right rotation



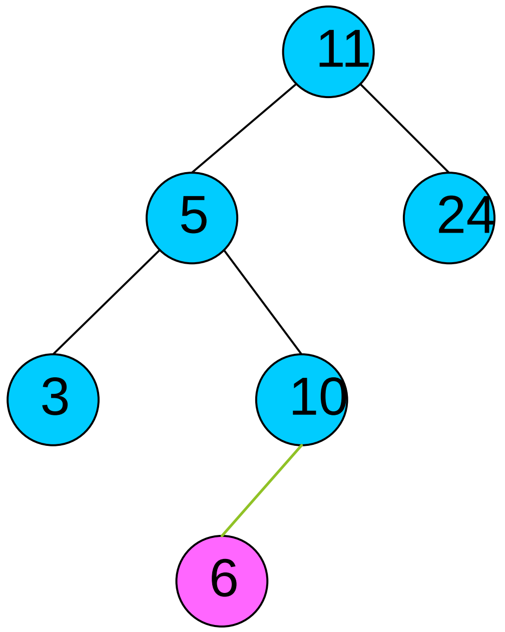


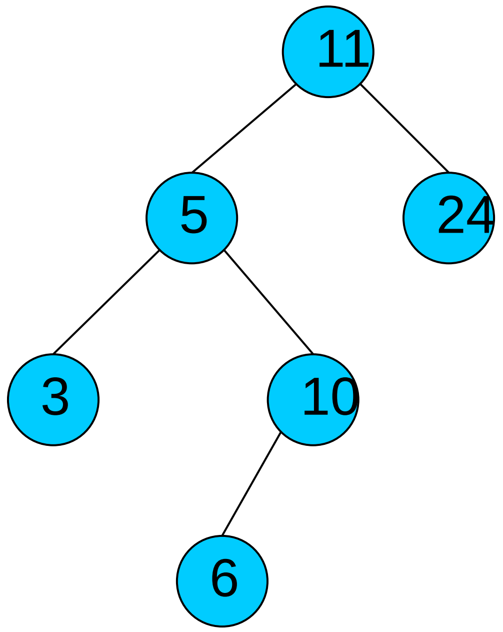
Left rotation



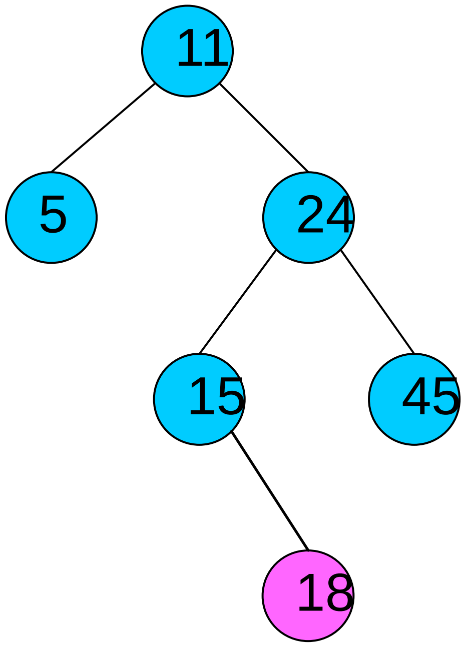


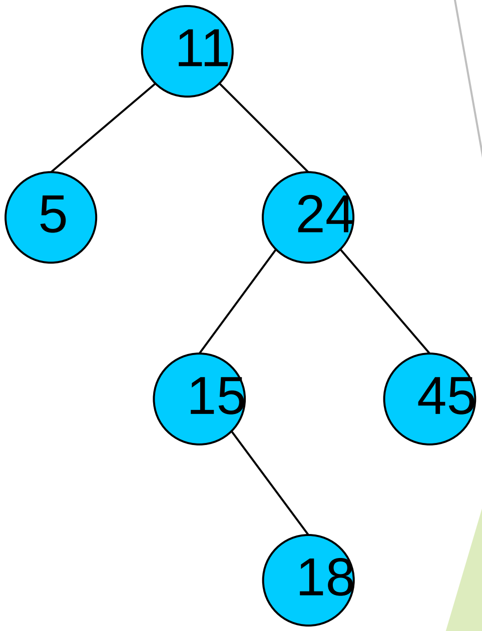
Left-Right rotation





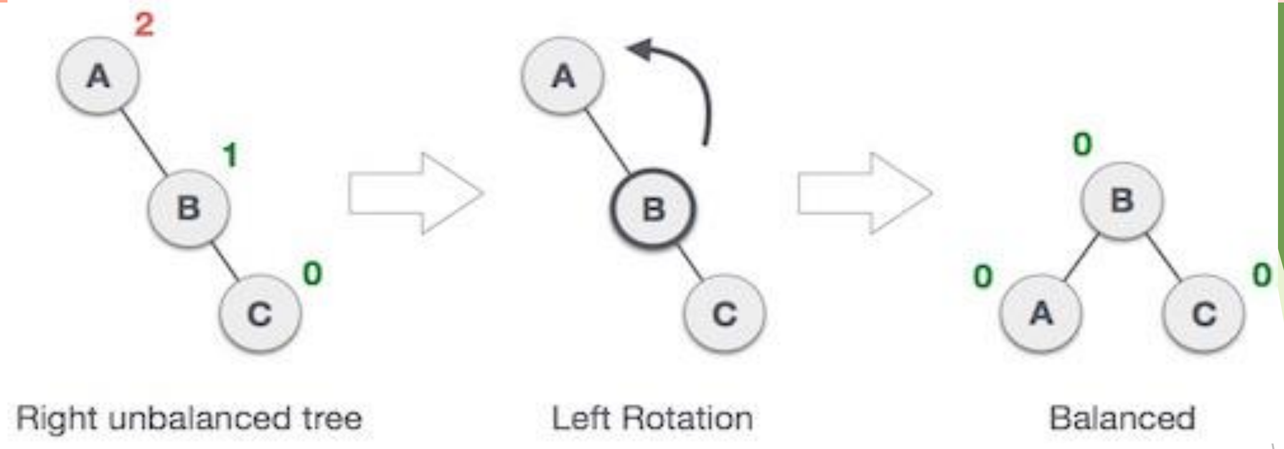
Right-Left rotation

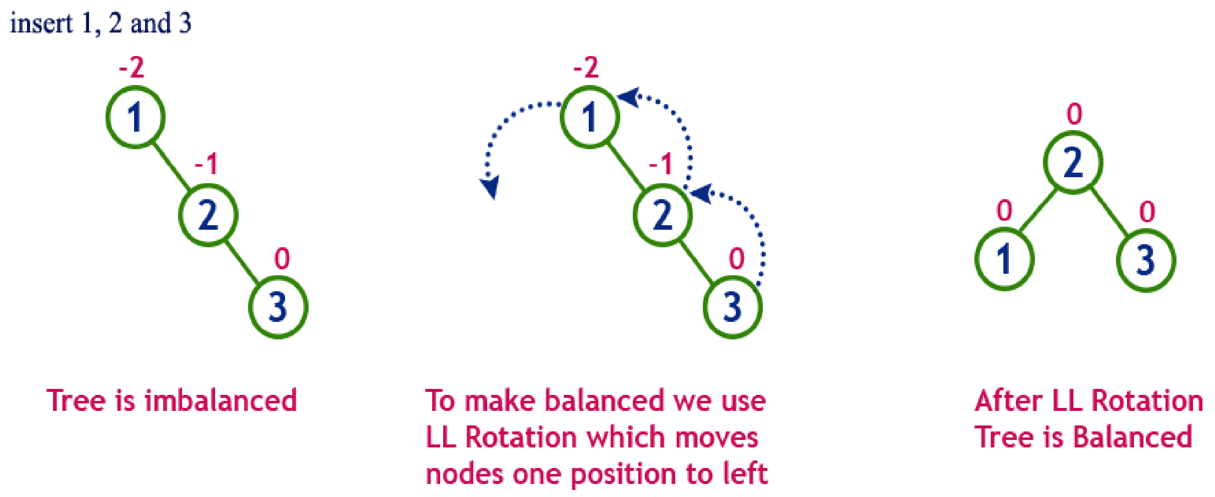




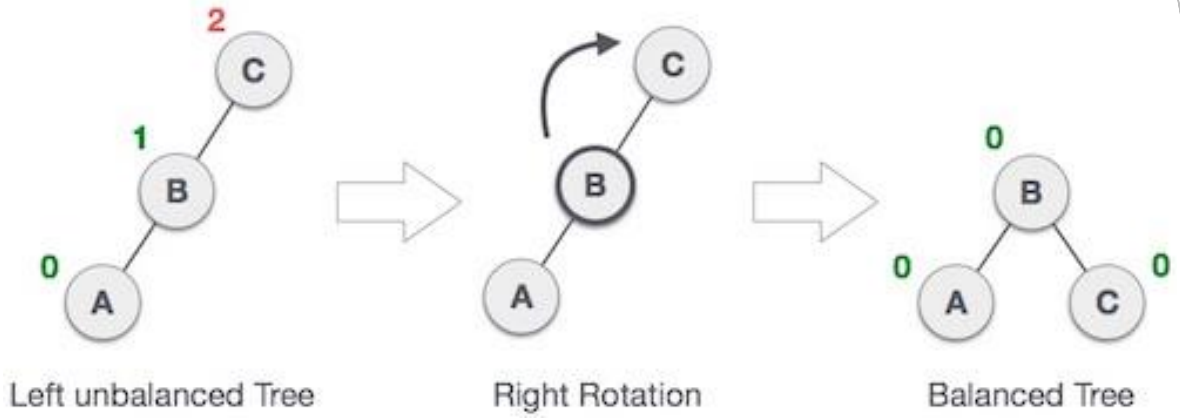
Example 2

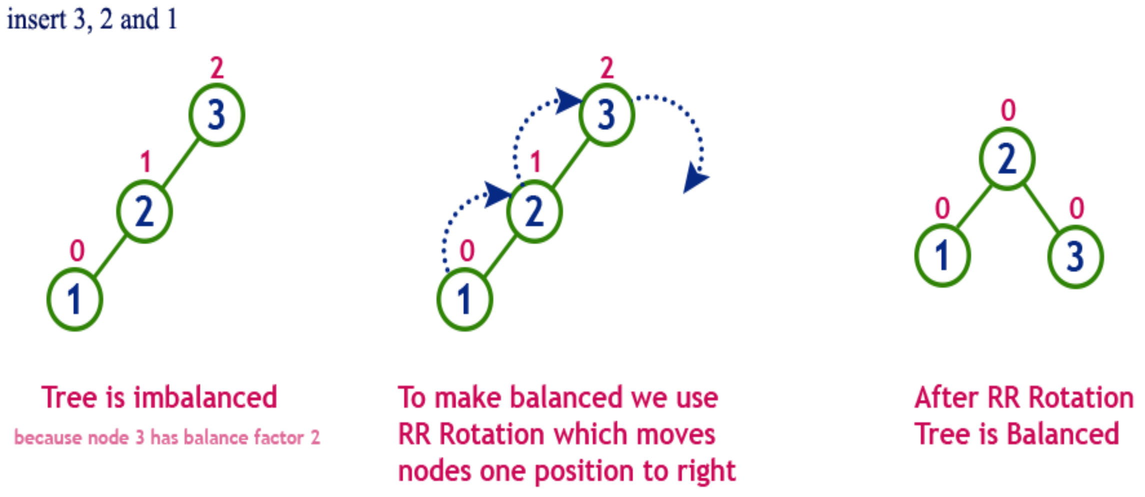
Left rotation



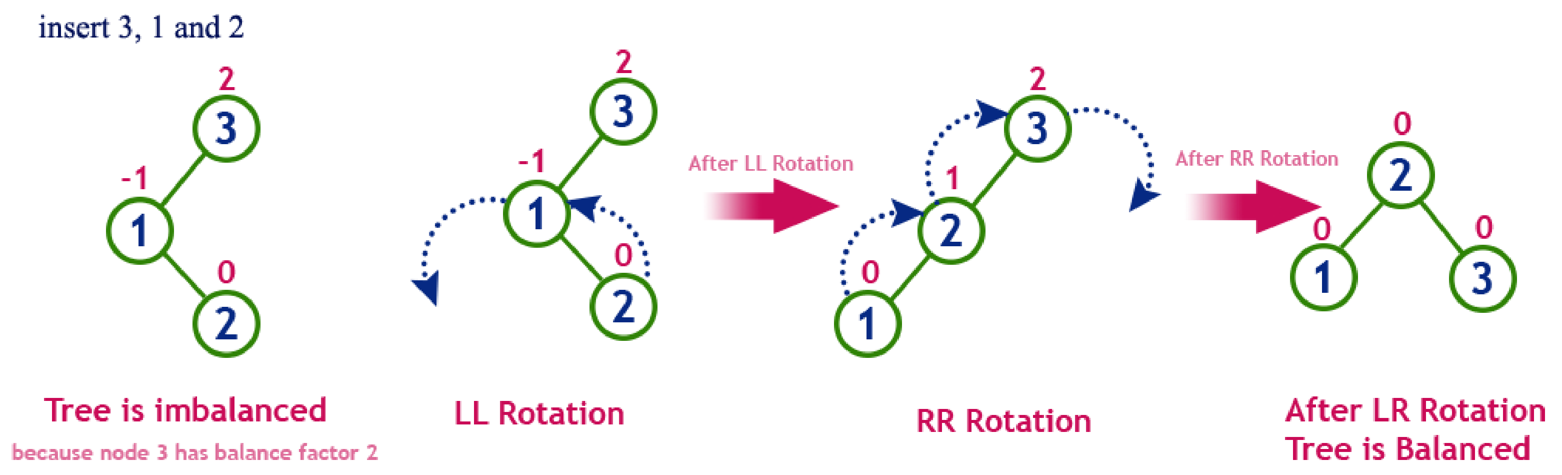


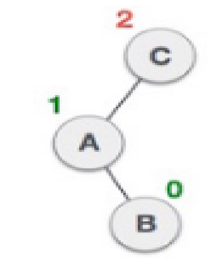
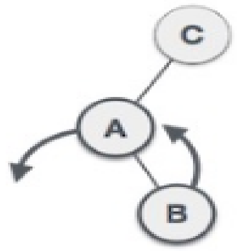
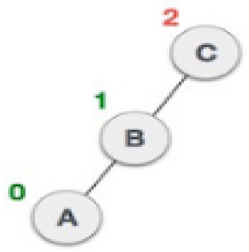
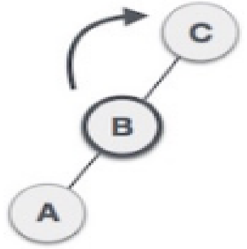
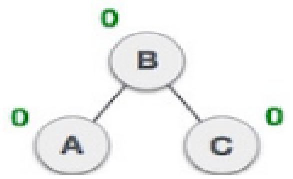
Right rotation



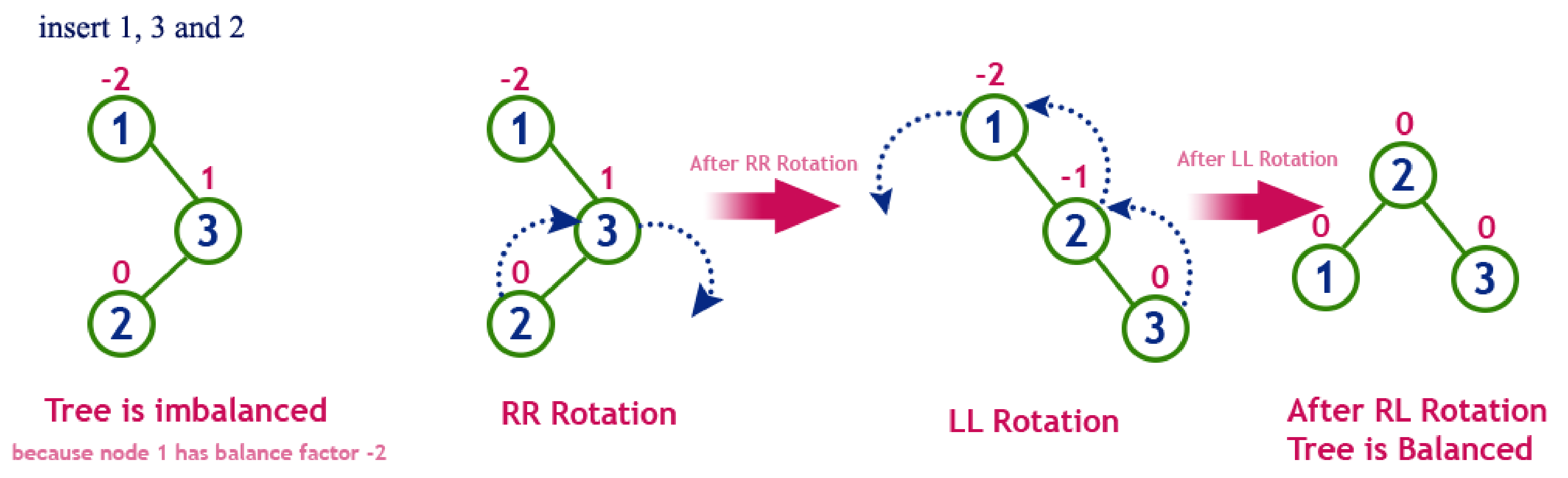

insert 3, 2 and 1
2
3
1
2
0
1
Tree is imbalanced
because node 3 has balance factor 2
2
3
1
2
0
1
To make balanced we use RR Rotation which moves nodes one position to right
0
2
0
1
0
3
After RR Rotation Tree is Balanced

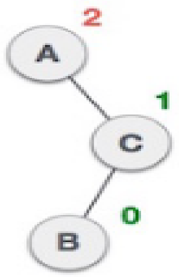
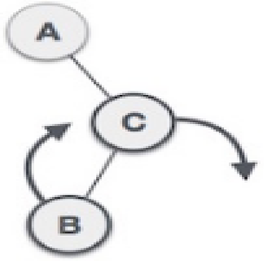
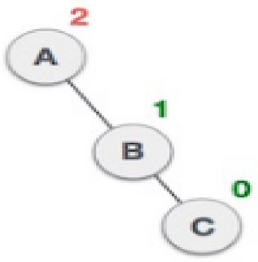
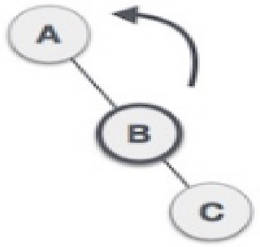
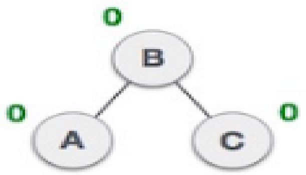
Left-Right rotation



State	Action
	A node has been inserted into the right subtree of the left subtree. This makes **C** an unbalanced node. These scenarios cause AVL tree to perform left-right rotation.
	We first perform the left rotation on the left subtree of **C**. This makes **A**, the left subtree of **B**.
	Node **C** is still unbalanced, however now, it is because of the left-subtree of the left-subtree.
	We shall now right-rotate the tree, making **B** the new root node of this subtree. **C** now becomes the right subtree of its own left subtree.
	The tree is now balanced.

Right-Left rotation

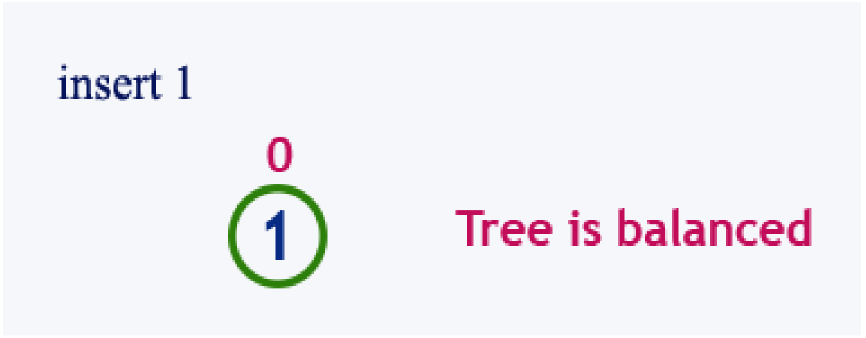


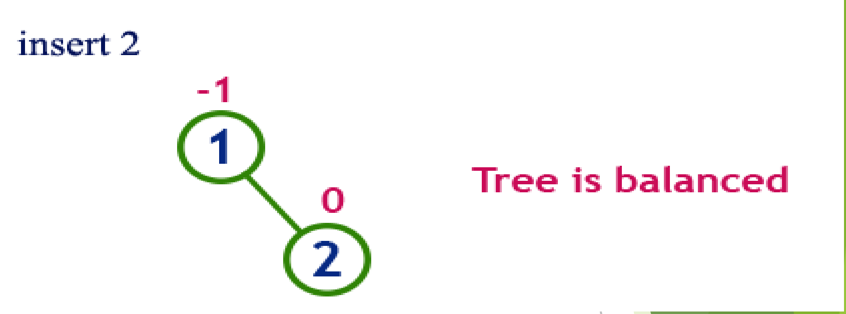
State	Action
	A node has been inserted into the left subtree of the right subtree. This makes **A**, an unbalanced node with balance factor 2.
	First, we perform the right rotation along **C** node, making **C** the right subtree of its own left subtree **B**. Now, **B** becomes the right subtree of **A**.
	Node **A** is still unbalanced because of the right subtree of its right subtree and requires a left rotation.
	A left rotation is performed by making **B** the new root node of the subtree. **A** becomes the left subtree of its right subtree **B**.
	The tree is now balanced.

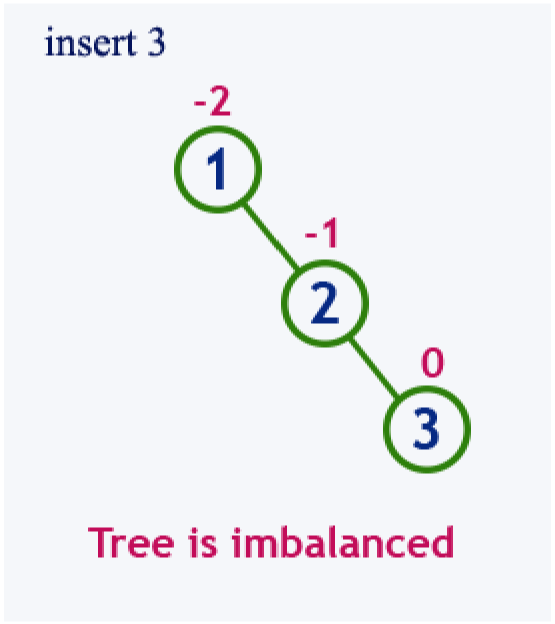
Example 3

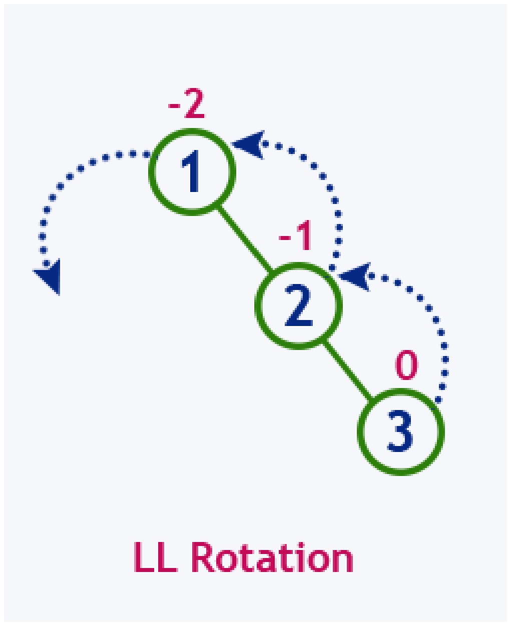
Construct an AVL Tree by inserting numbers from 1 to 8.

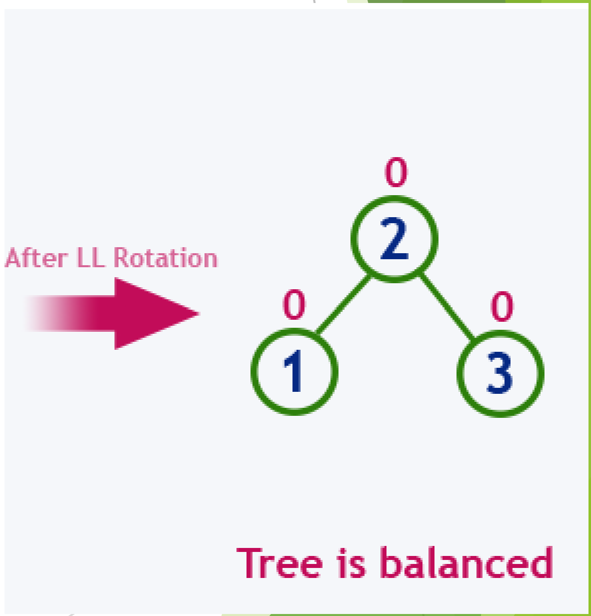
Inserting numbers from 1 to 8



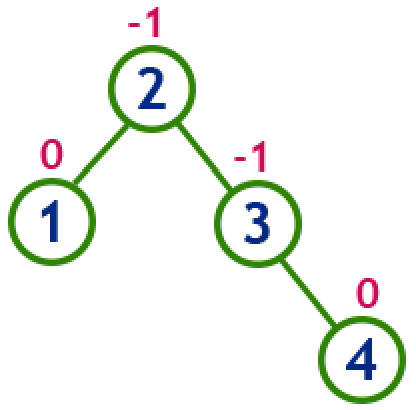




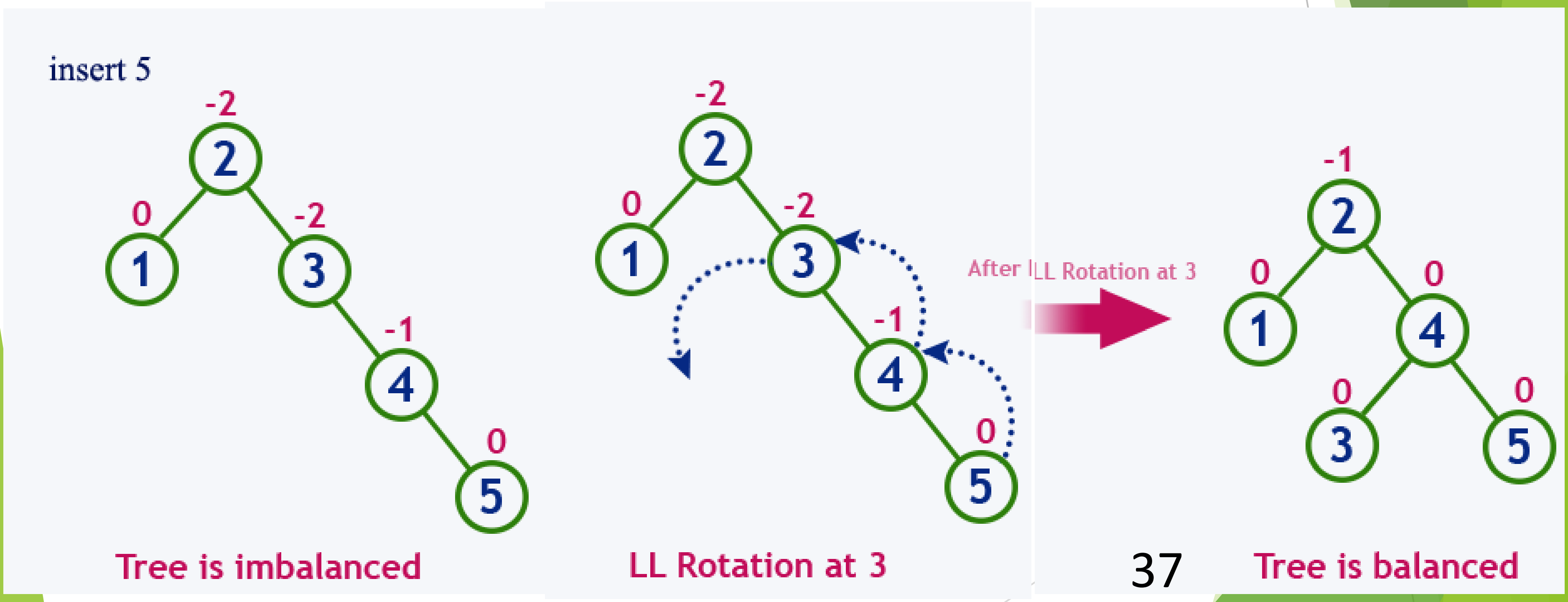




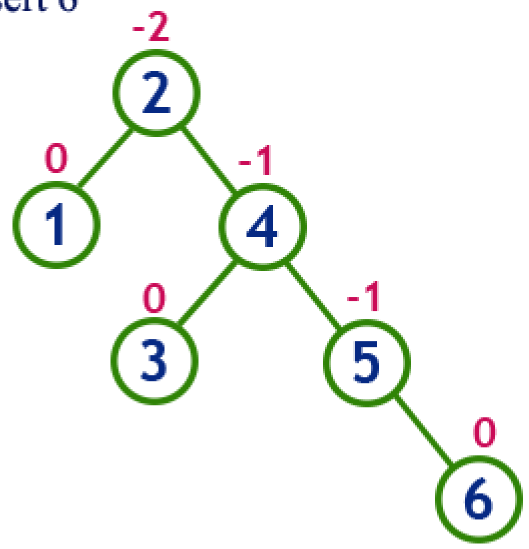
insert 4



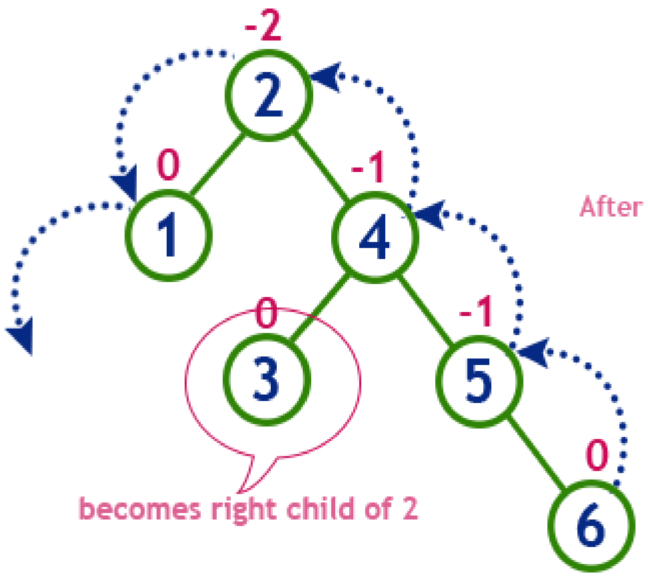
Tree is balanced



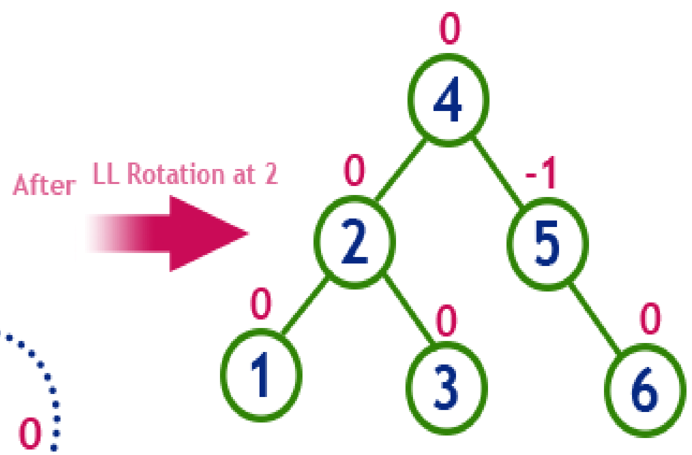
insert 6



Tree is imbalanced

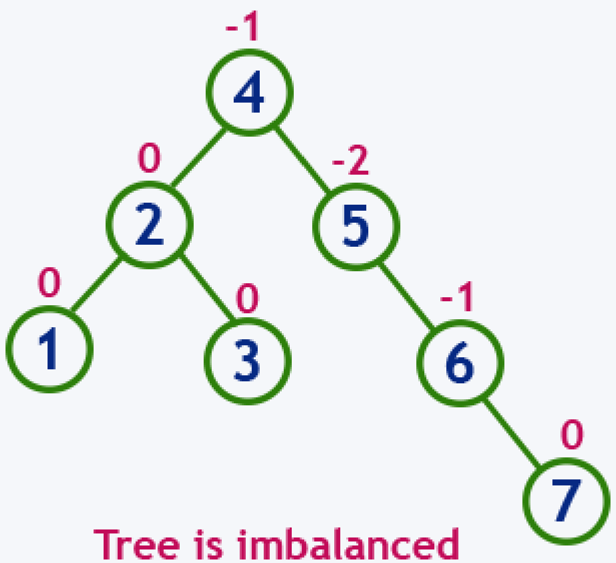


LL Rotation at 2

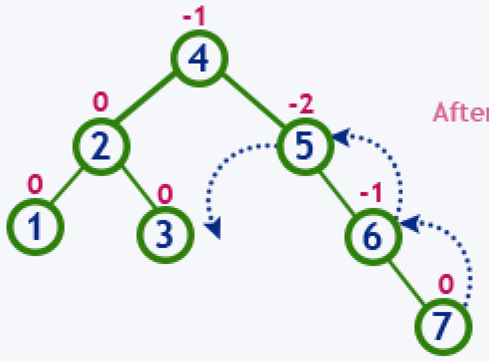


Tree is balanced

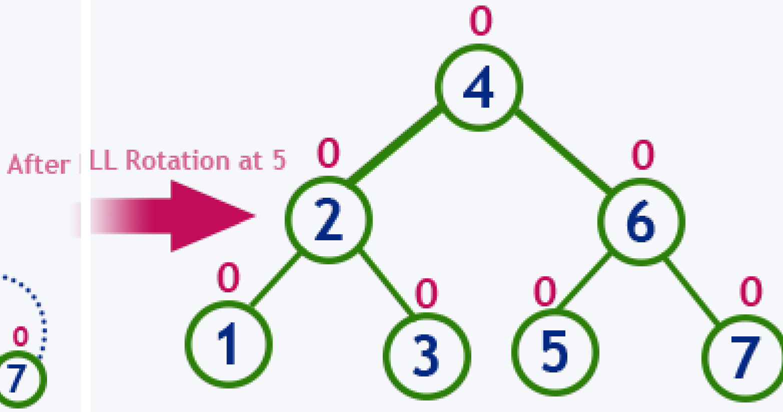
insert 7



Tree is imbalanced

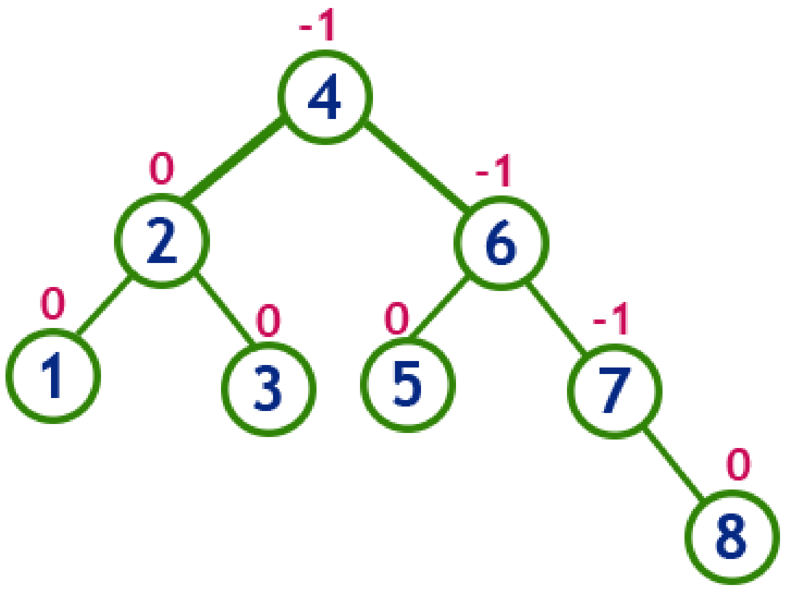


LL Rotation at 5



Tree is balanced

insert 8

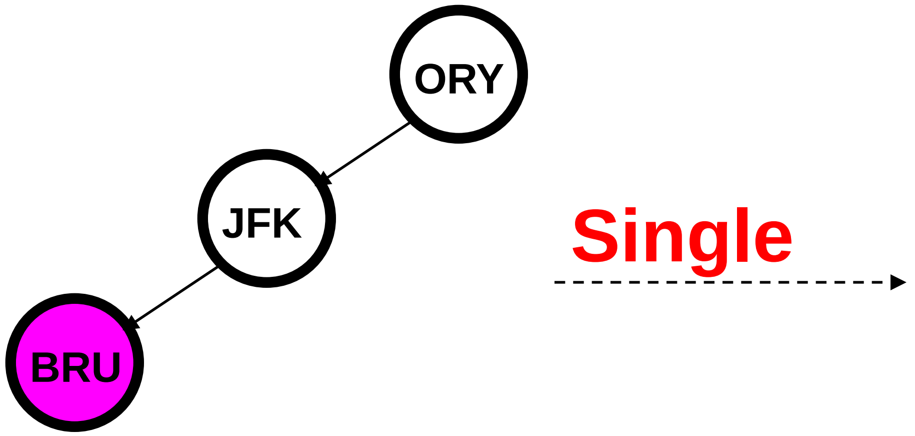


Tree is balanced

Example 4

An AVL Tree for Airport Names

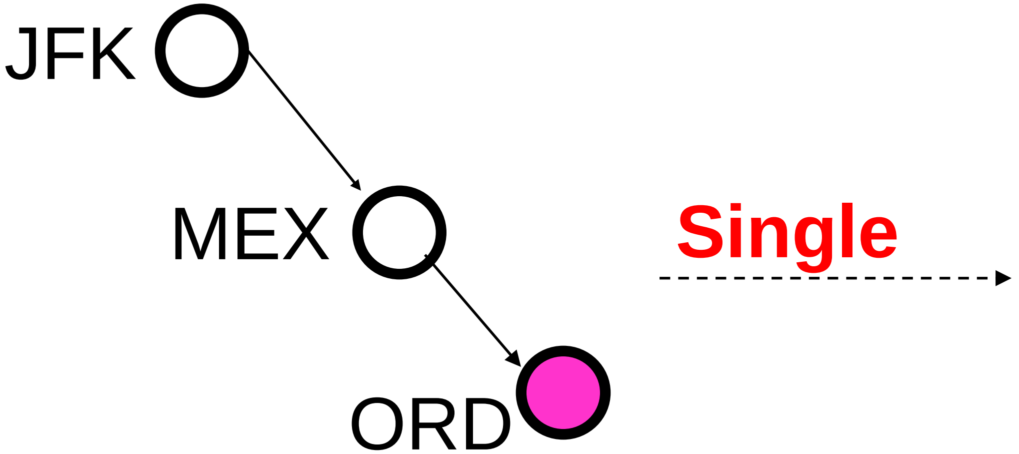
- After insertion of ORY, JFK and BRU



Not AVL balanced

An AVL Tree for Airport Names

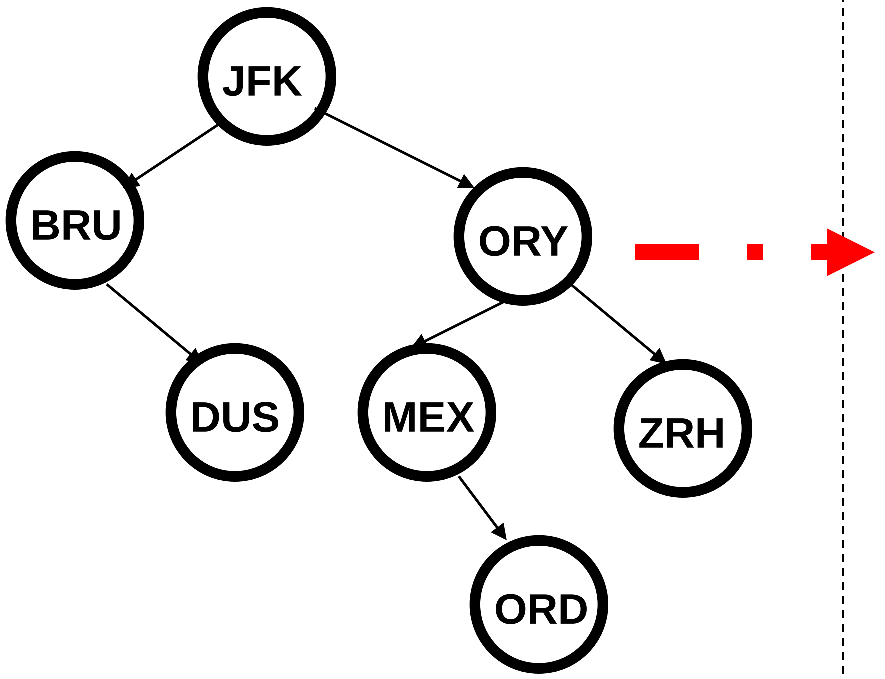
•After insertion of JFK , MEX and ORD :



Not AVL balanced

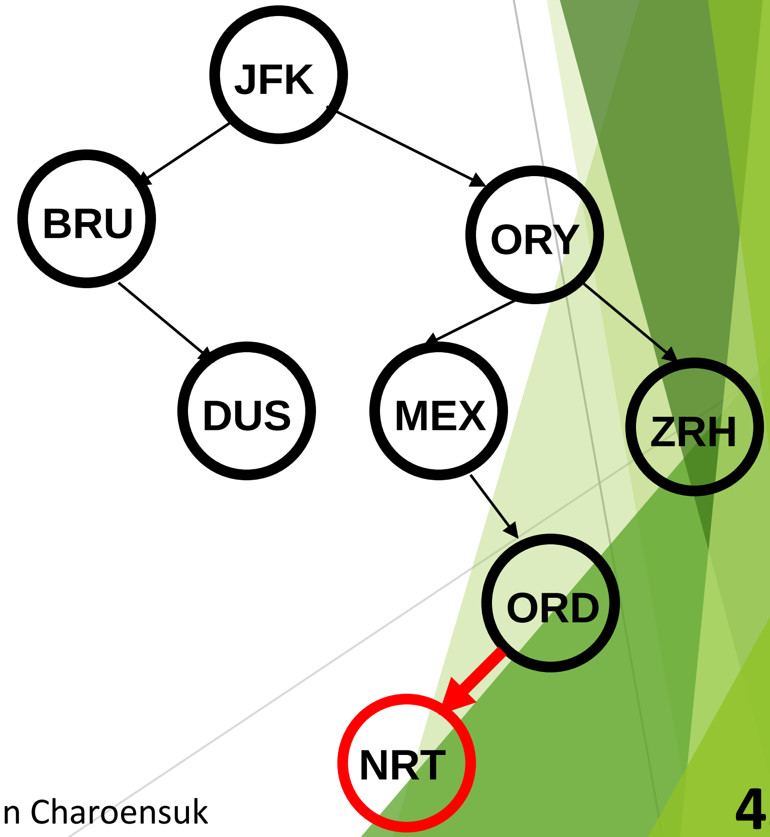
An AVL Tree for Airport Names (contd.)

After insertion of DUS, ZRH, MEX and ORD



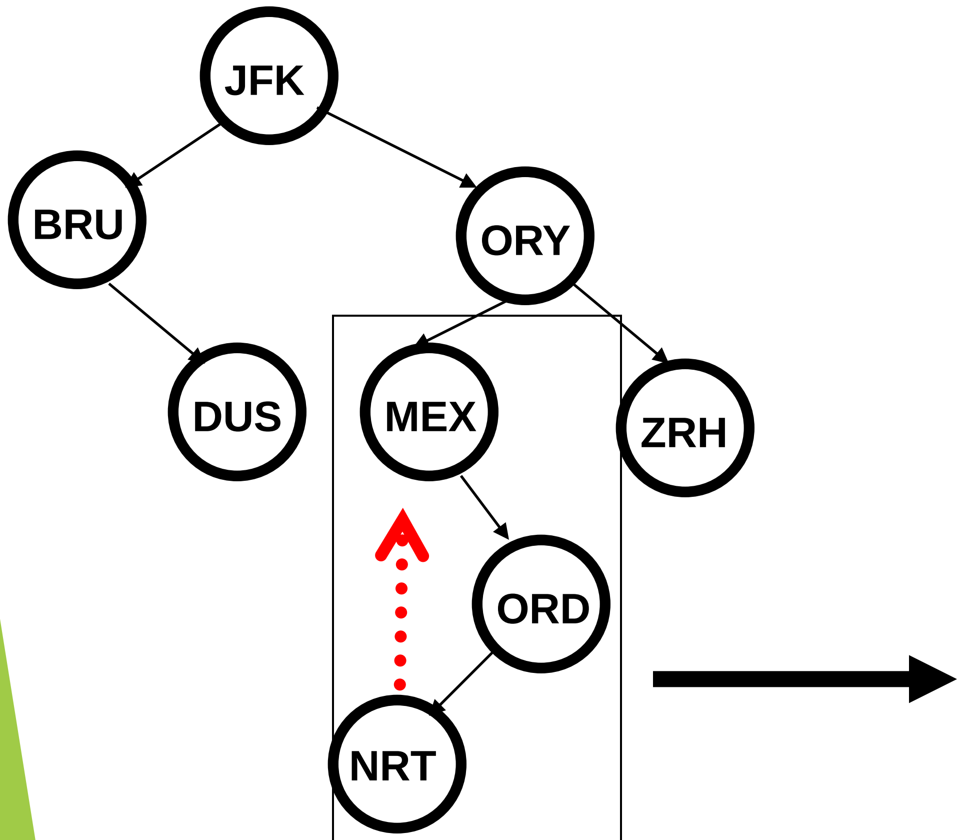
Still AVL Balanced

After insertion of NRT?

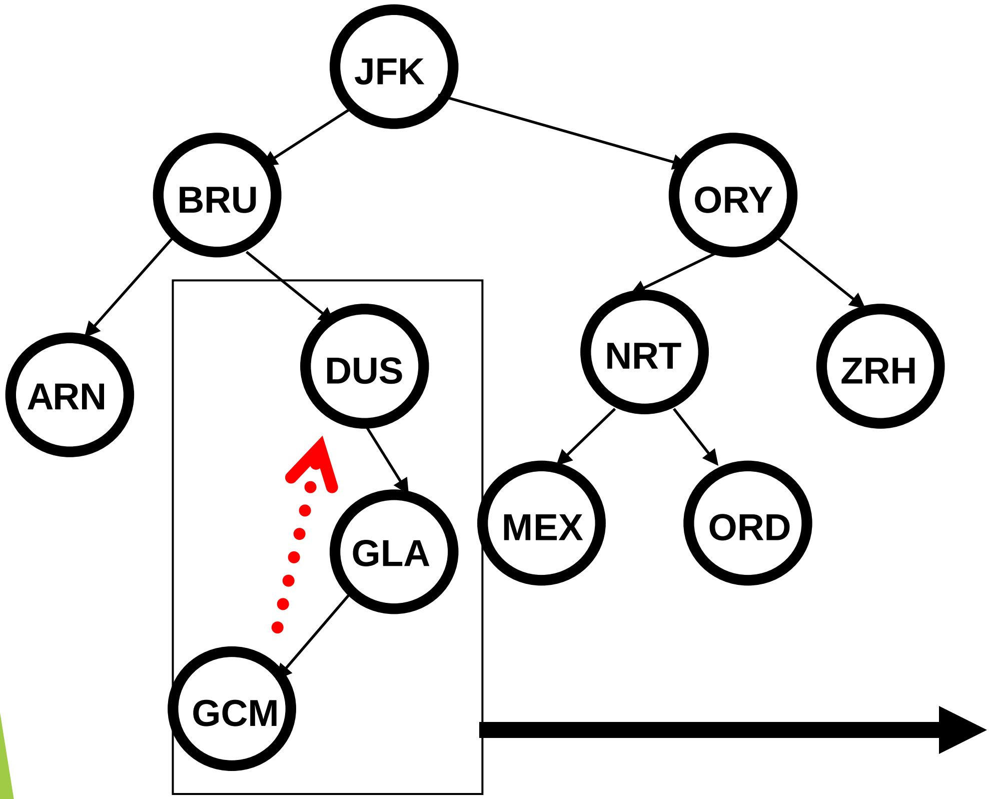


An AVL Tree

Not AVL Balanced

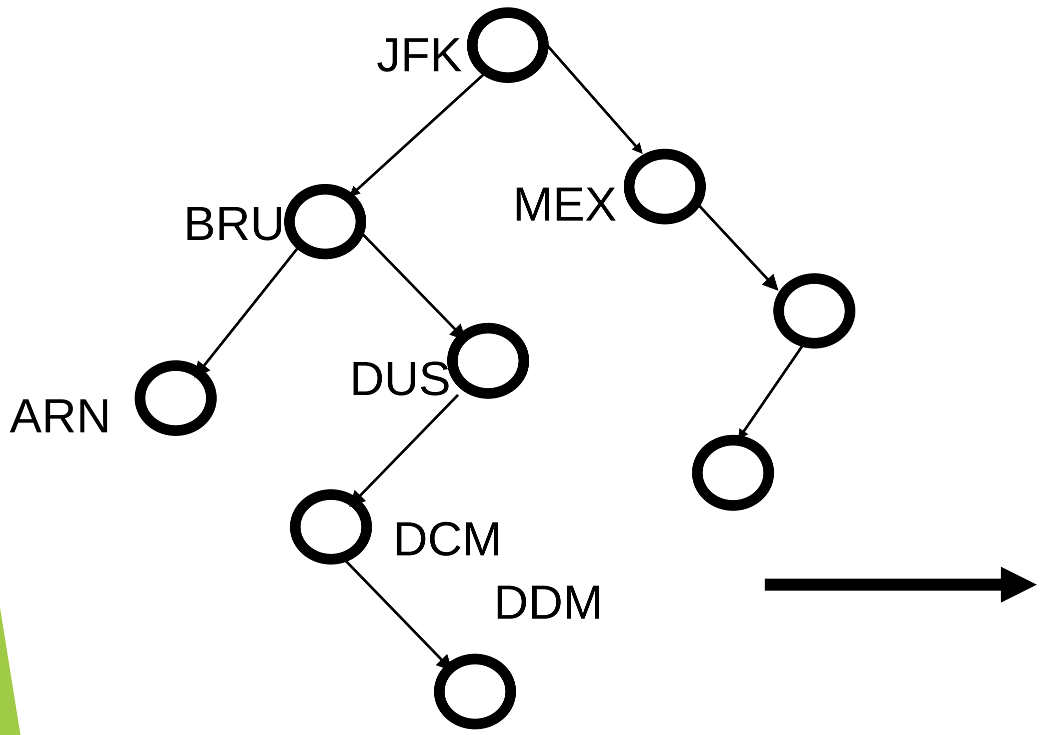


An AVL Tree…



NOT AVL BALANCED

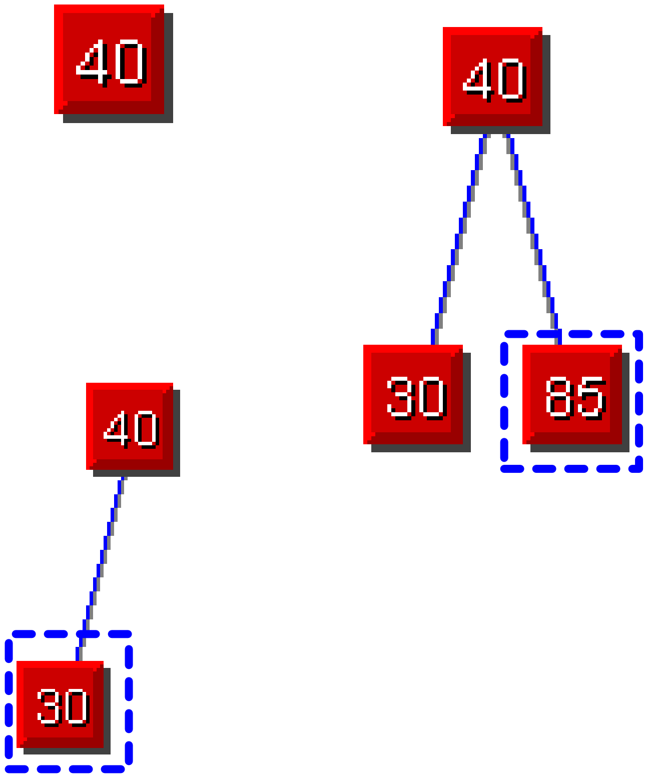
An AVL Tree…



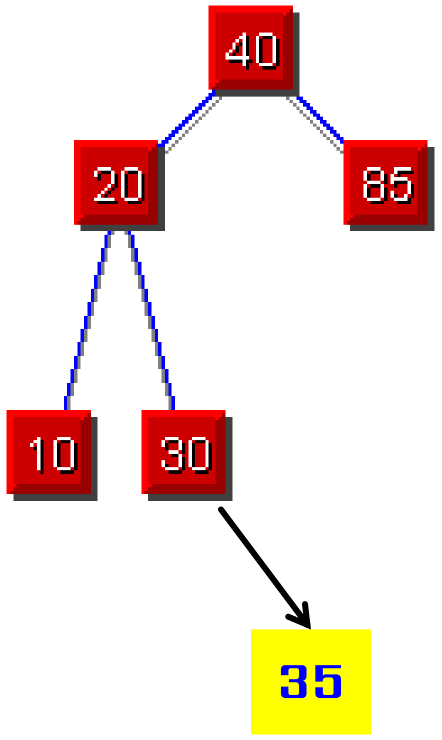
double right

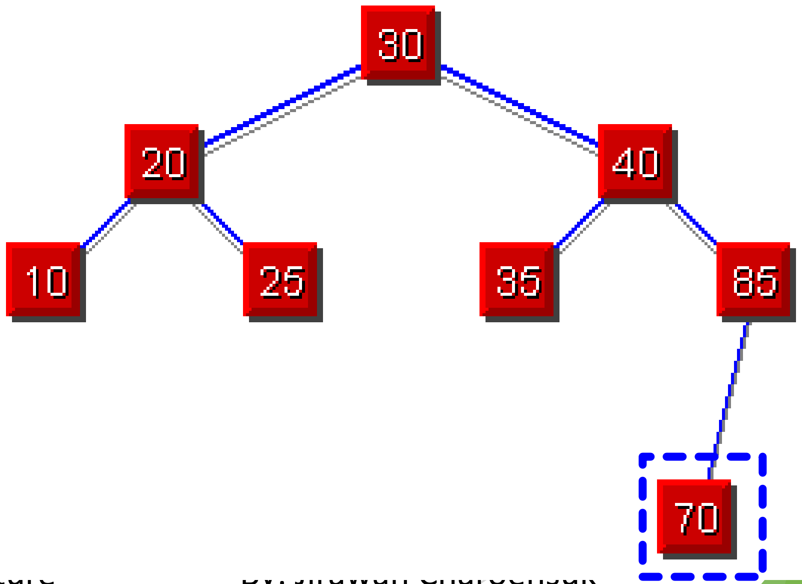
Example 5

40 30 85 20 10 35 25 70 78 60

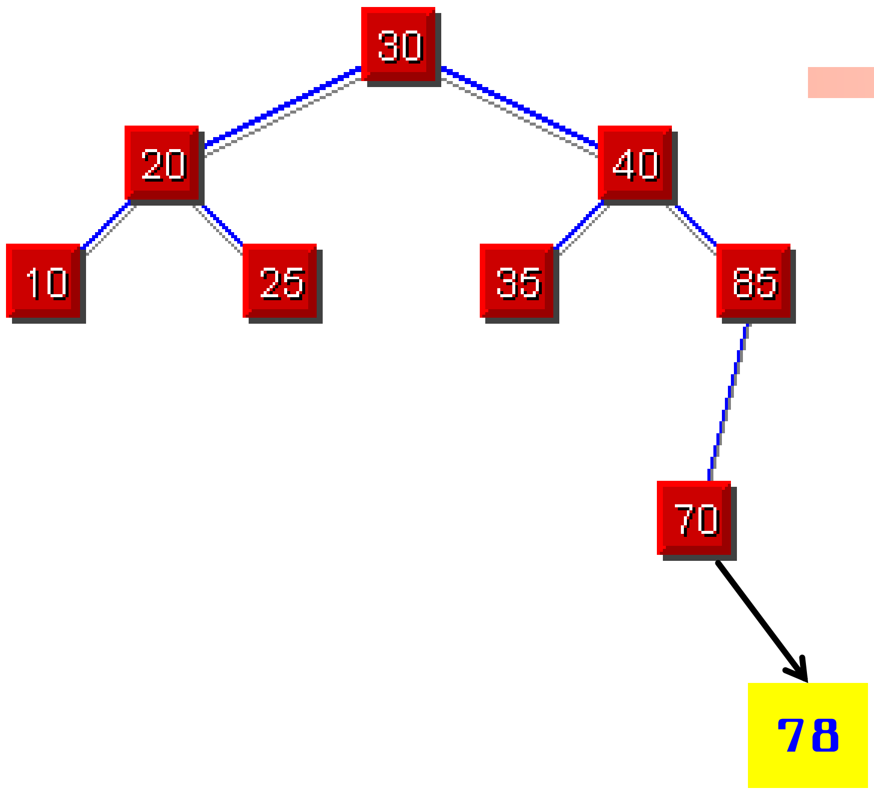


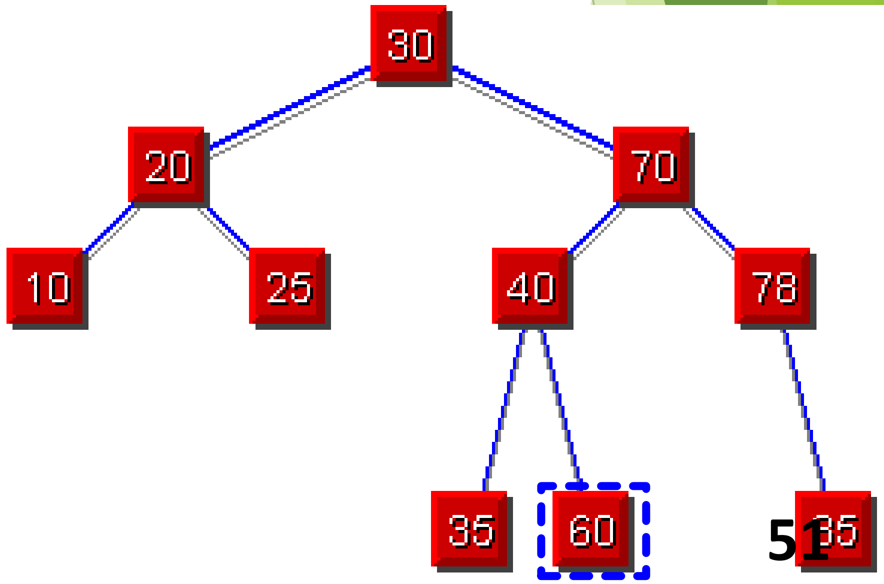
40 30 85 20 10 35 25 70 78 60





40 30 85 20 10 35 25 70 78 60





Question