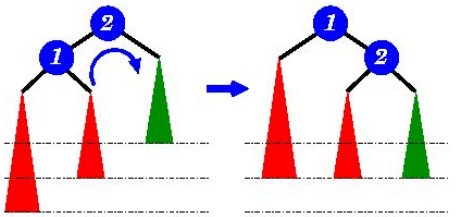
Árvores Balanceadas:
AVL



Árvores Binárias

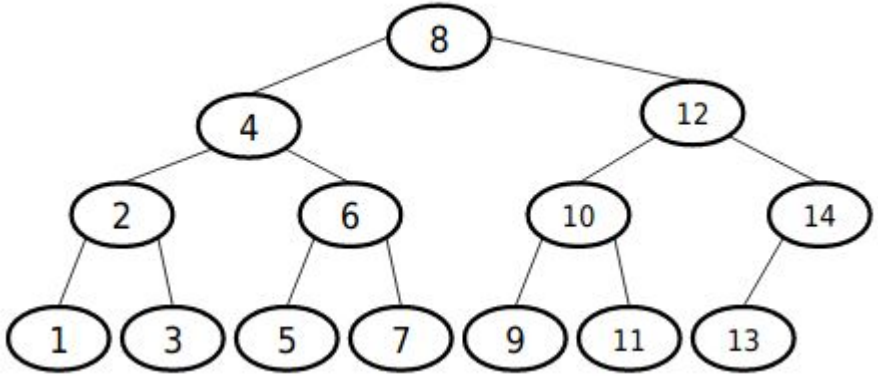
- A eficiência na busca em uma árvore depende de seu balanceamento
  - O(N), caso a árvore seja degenerada (não balanceada)
  - O($\log_b$ N), caso a árvore esteja balanceada (b=2)

Árvores Binárias

- A eficiência na busca em uma árvore depende de seu balanceamento
  - O(N), caso a árvore seja degenerada (não balanceada)
  - $O(\log_b N)$, caso a árvore esteja balanceada (b=2)
- O balanceamento da árvore não é garantido pela inserção dos elementos. A árvore apenas fica balanceada caso a ordem de inserção propicie isso.

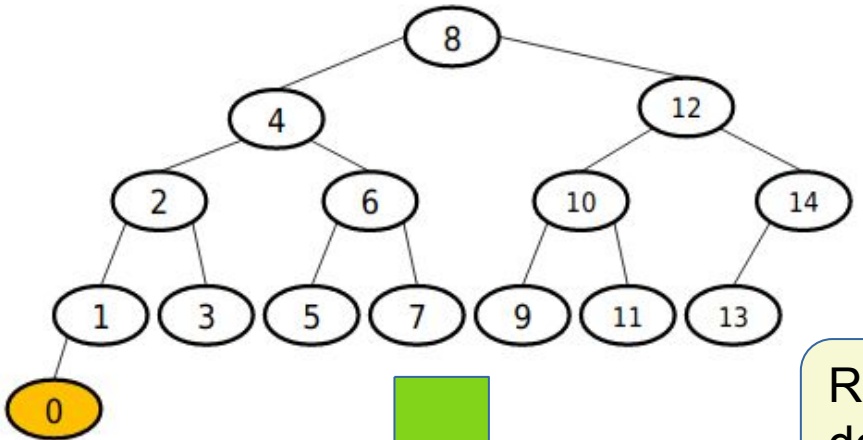
Árvores Binárias

- Uma **árvore completa,** é aquela que minimiza o número de comparações necessárias para o acesso a uma informação
  - Uma árvore deste tipo deve estar perfeitamente balanceada
  - O balanceamento deve ser verificado após cada inserção/remoção; se necessário a árvore deve ser reorganizada para um estado balanceado

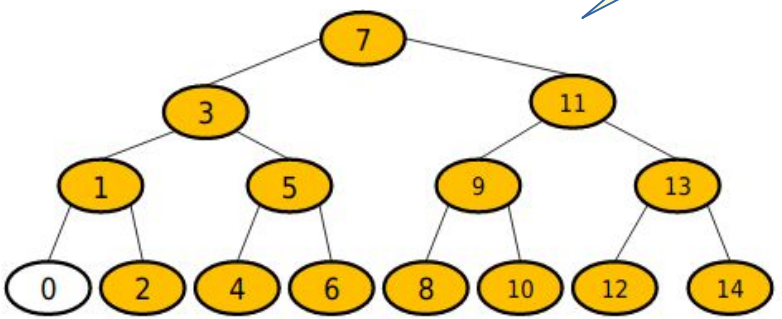


Árvores Binárias

- Uma **árvor** …niza o número de comparaçõe… e uma informação
  - Uma árvor… balanceada
  - O balance… inserção, e caso necessário… do



Reorganizar a árvore desta maneira torna-se muito caro!



Retirado da aula do Prof. Jairo de Souza - UFJF

AVL

- A **AVL** foi proposta por **A**delson-**V**elskii e **L**andis em 1962
  - É uma árvore binária de busca que, após operações de inserção e remoção, realiza rotinas de rebalanceamento
    - rotações simples ou duplas
  - Apenas a parte afetada pela inserção/remoção é rebalanceada

AVL

- Uma AVL é uma árvore na qual as alturas das subárvores da esquerda e da direita de cada nó diferem **no máximo em 1 unidade**

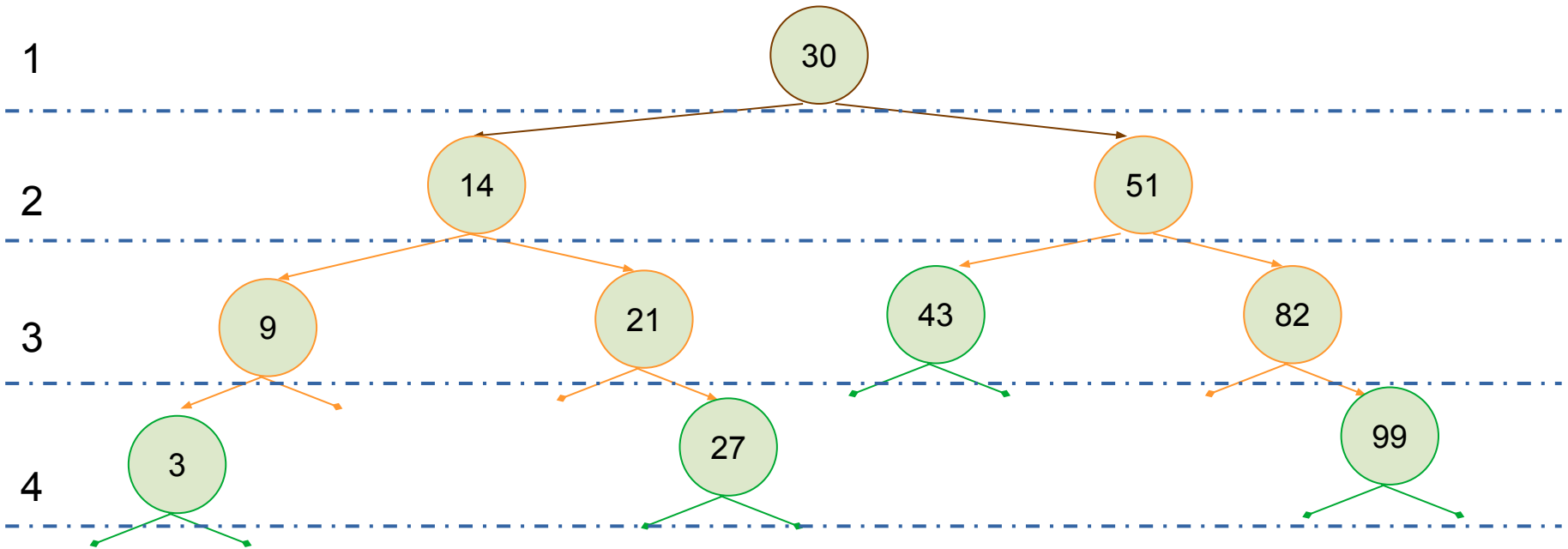
Fator de balanceamento:

(altura da esquerda) – (altura da direita)

deve ser -1, 0 ou 1 em todos os nós

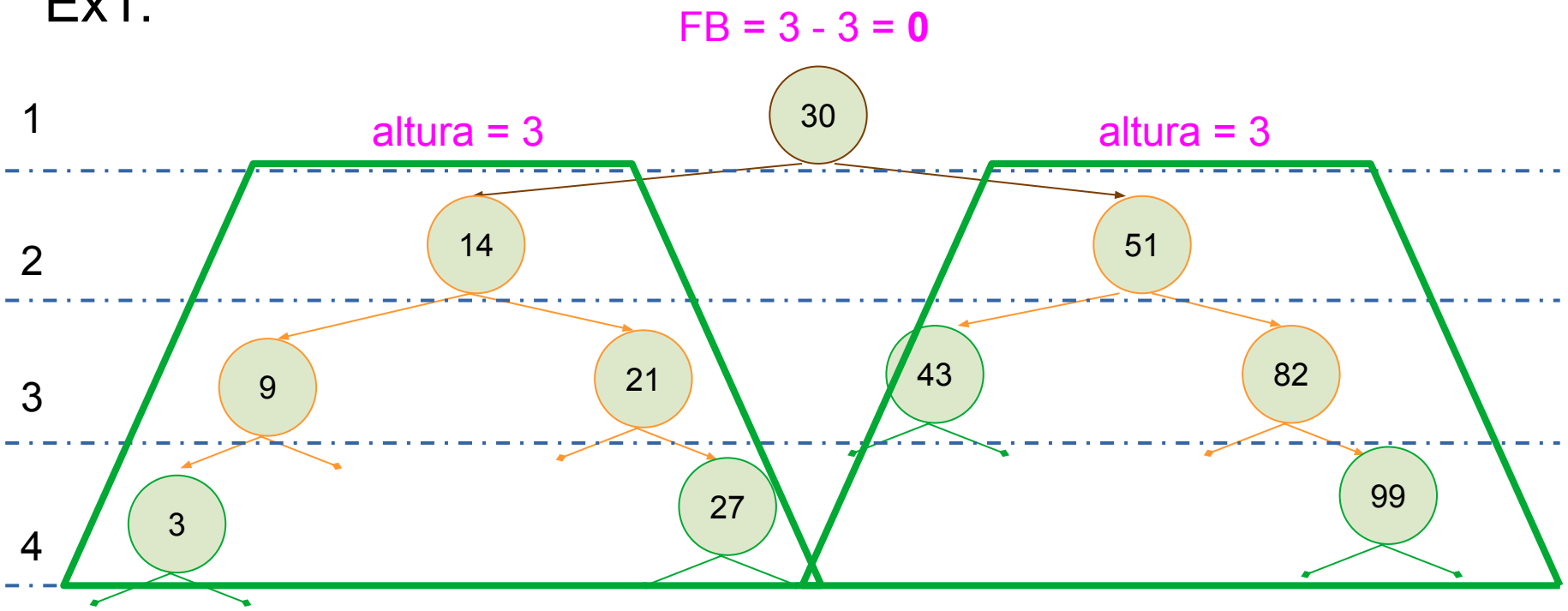
AVL

- Ex1:



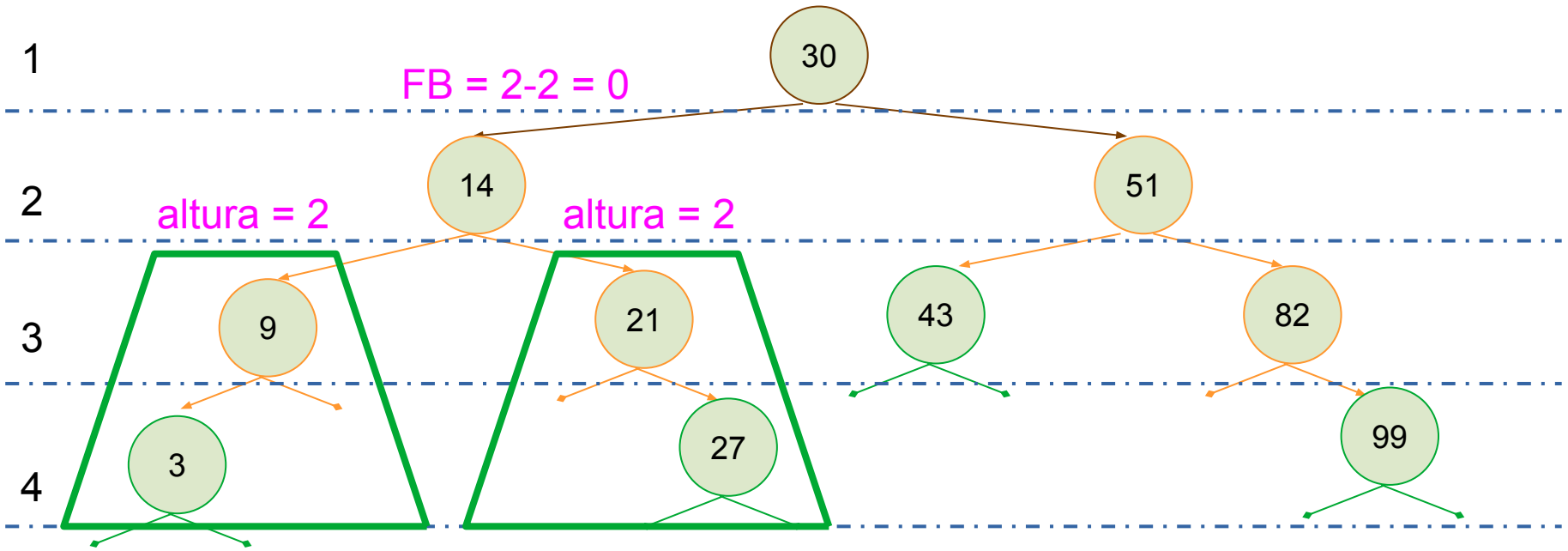
AVL

- Ex1:



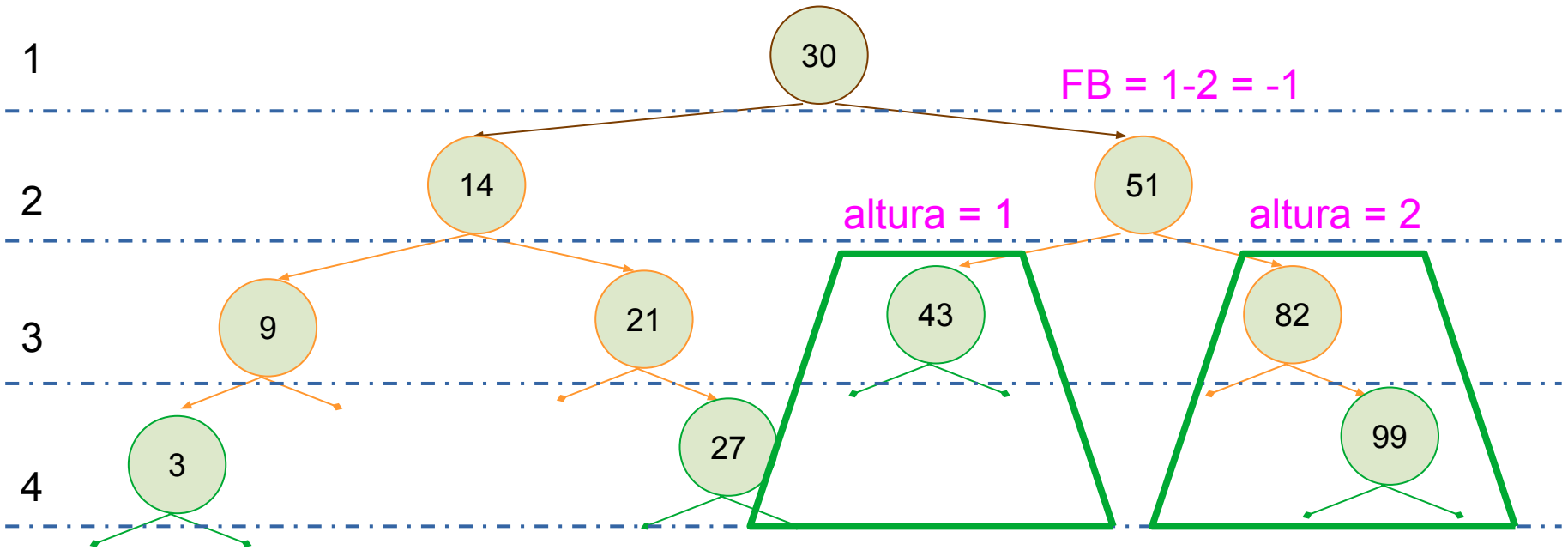
AVL

- Ex1:



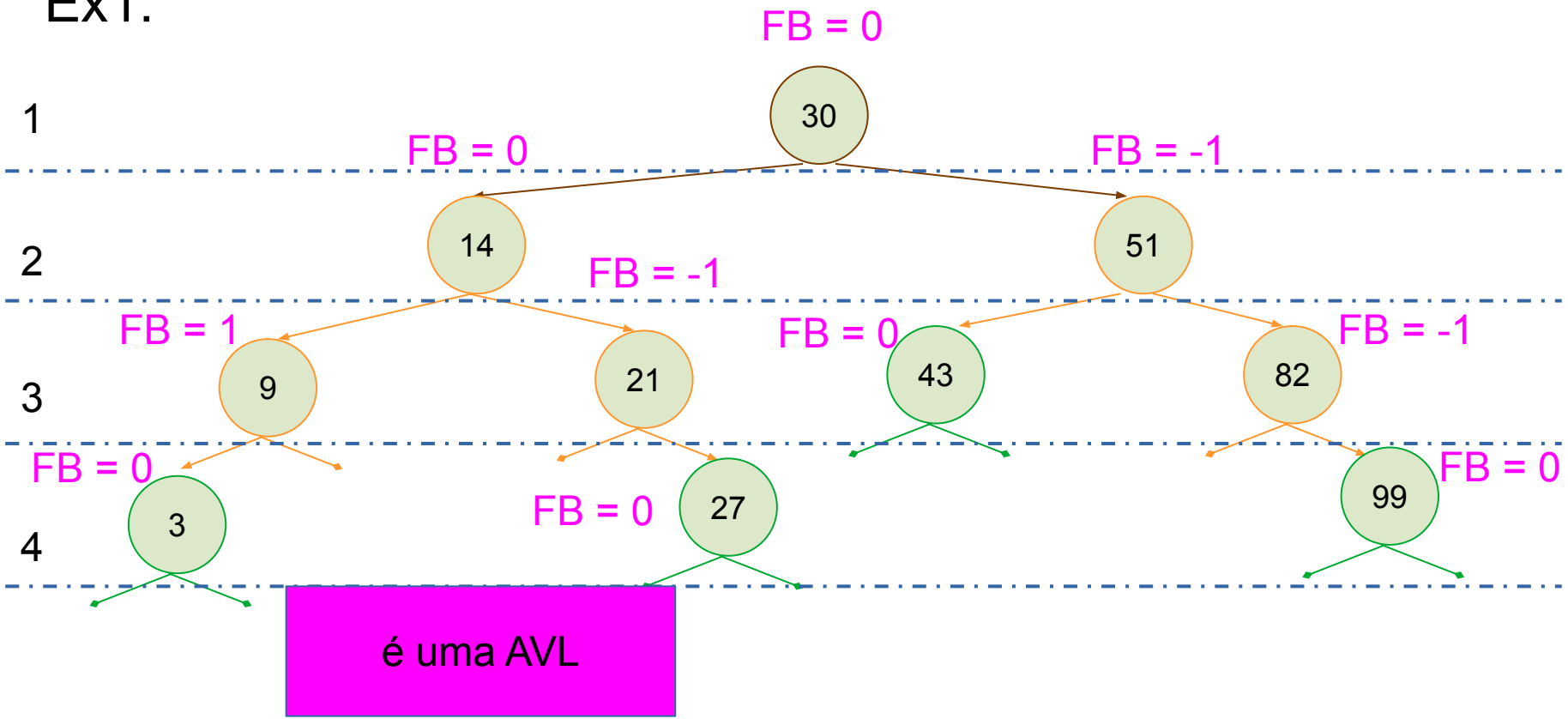
AVL

- Ex1:



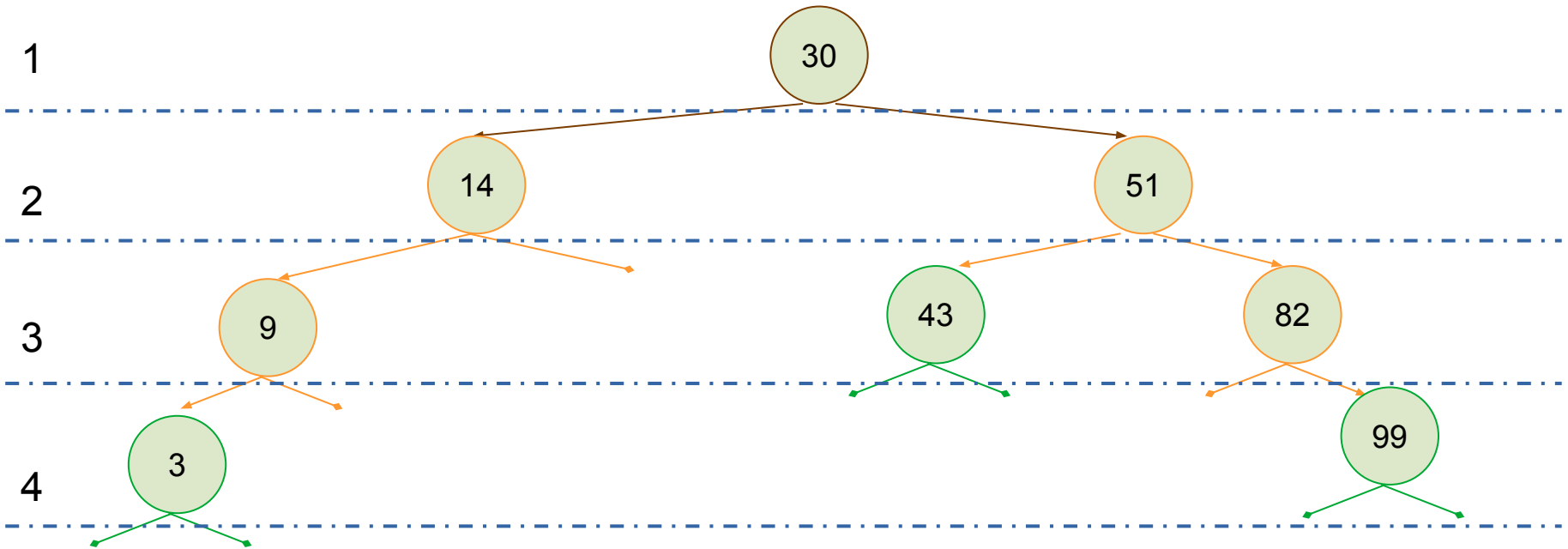
AVL

- Ex1:


FB = 0
30
1
FB = 0
FB = -1
14
51
2
FB = -1
FB = 1
FB = 0
FB = -1
9
21
43
82
3
FB = 0
FB = 0
FB = 0
3
27
99
4
é uma AVL

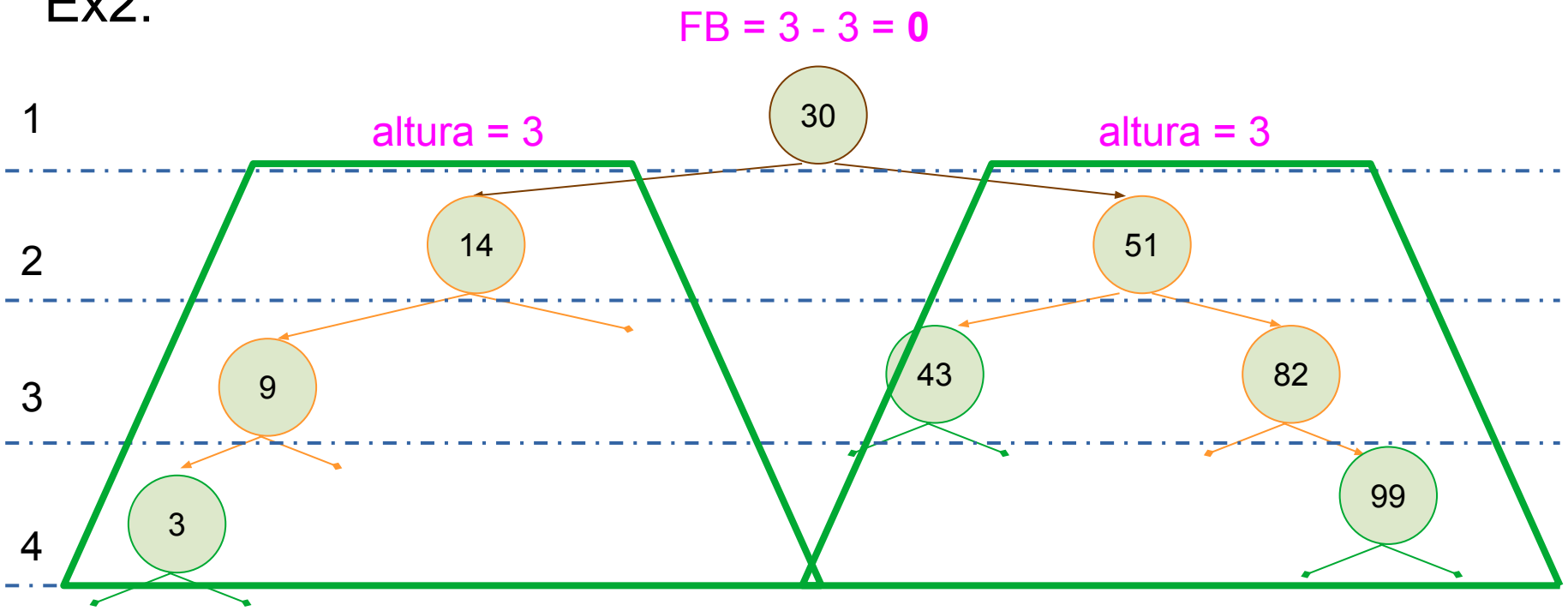
AVL

- Ex2:



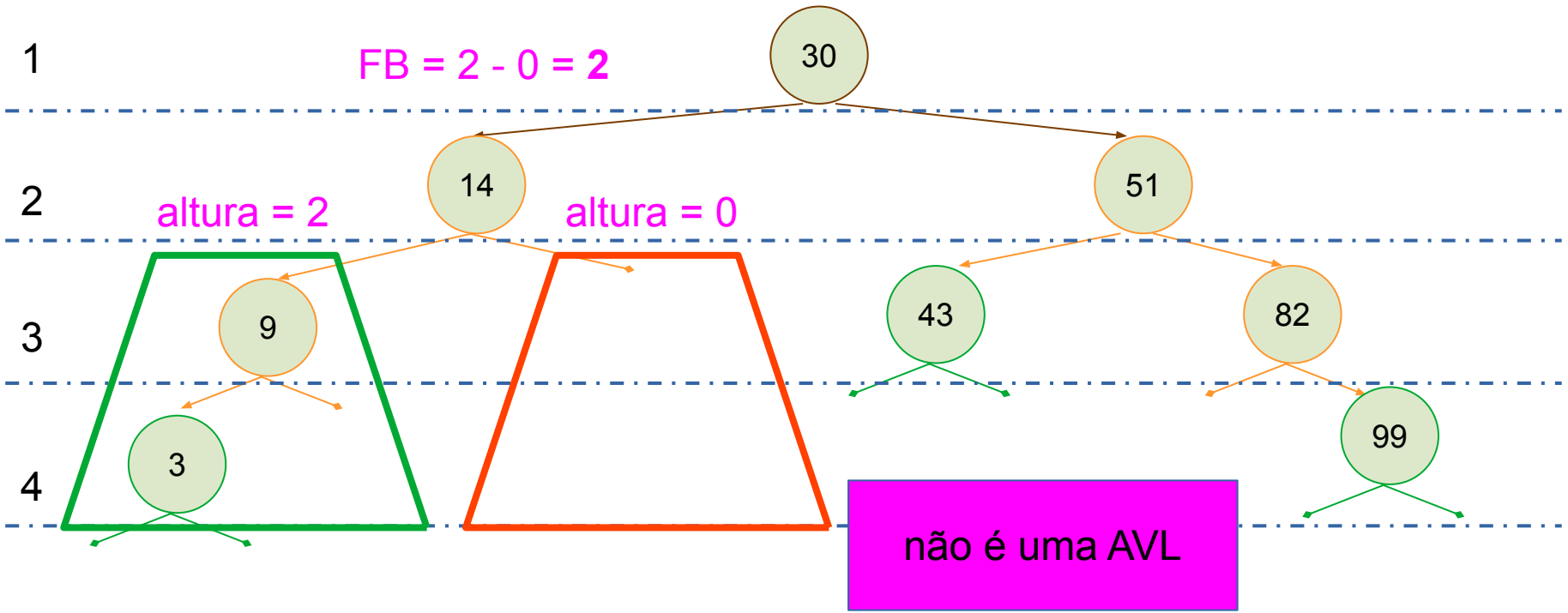
AVL

- Ex2:



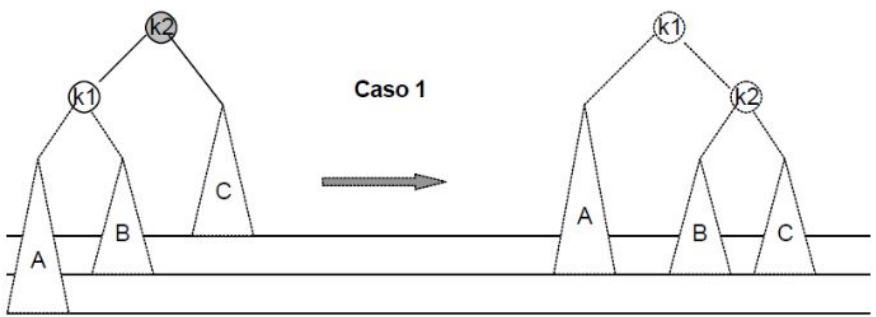
AVL

- Ex2:



AVL

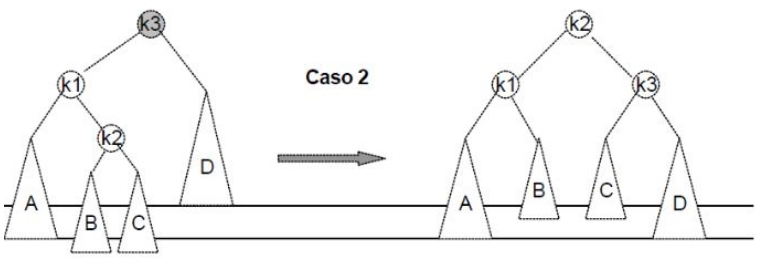
- Caso o desbalanceamento seja detectado, são executadas dois tipos de rotações para balancear a árvore:
  - Rotação simples
    - O nó desbalanceado (pai), seu filho e o seu neto estão todos no mesmo sentido de inclinação



Retirado da aula do Prof. Jairo de Souza - UFJF

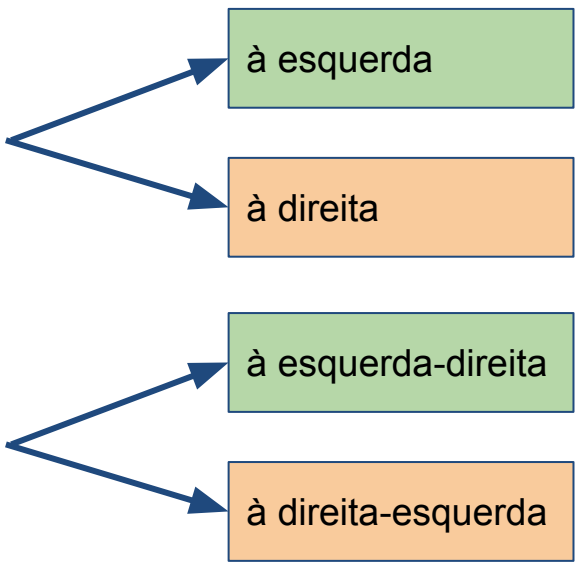
AVL

- Caso o desbalanceamento seja detectado, são executados dois tipos de rotações para balancear a árvore:
  - Rotação simples
    - O nó desbalanceado (pai), seu filho e o seu neto estão todos no mesmo sentido de inclinação
  - Rotação dupla
    - O nó desbalanceado (pai) e seu filho estão inclinados no sentido inverso ao neto
    - Equivale a duas rotações simples



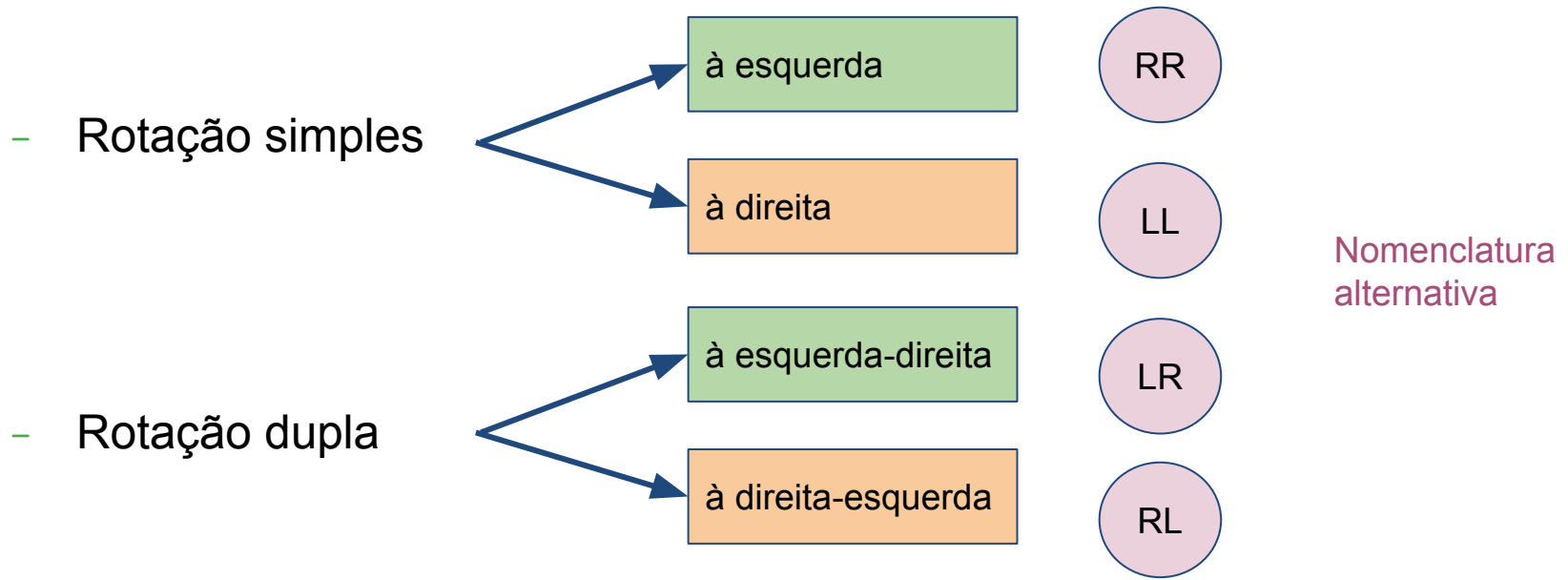
AVL

- As rotações podem ocorrer em ambos os sentidos, dando origem a 4 possíveis movimentos:



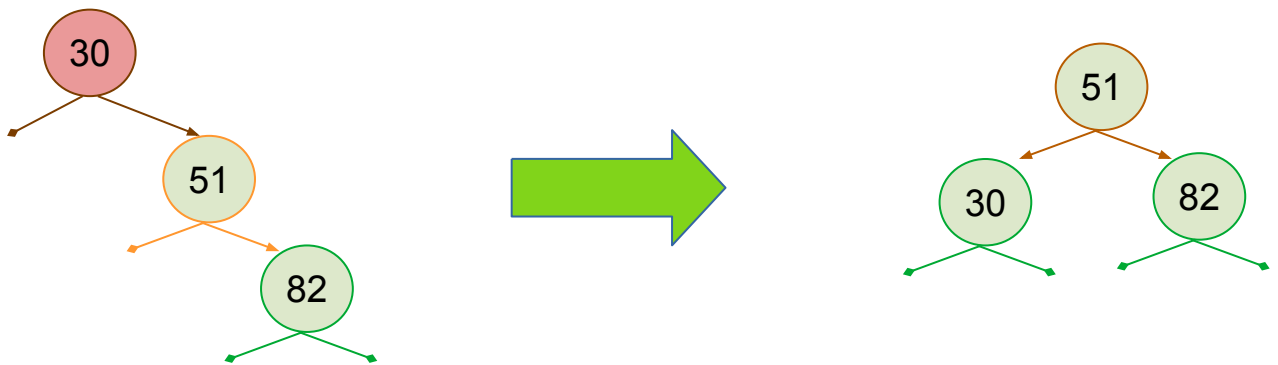
AVL

- As rotações podem ocorrer em ambos os sentidos, dando origem a 4 possíveis movimentos:



AVL

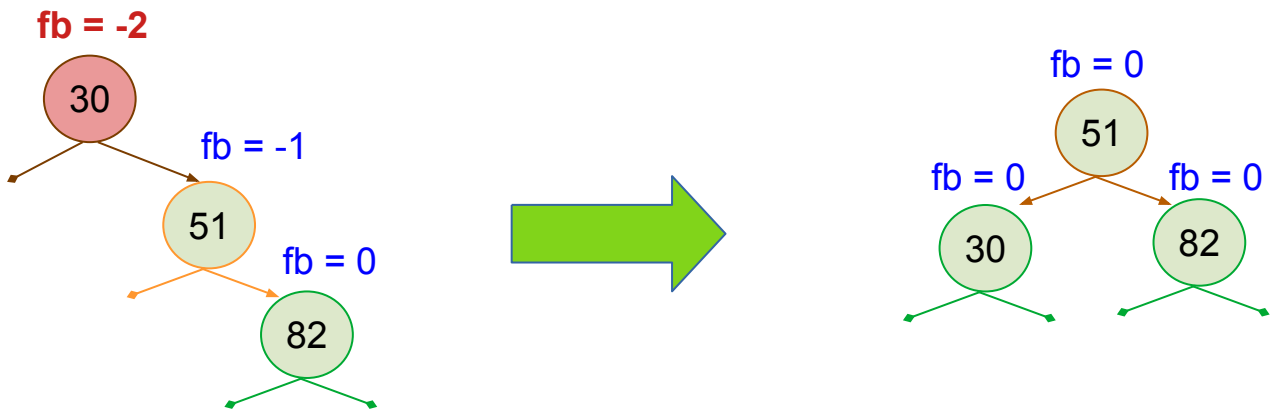
- **Rotação simples à esquerda** ou **Rotação RR**
  - o desbalanceamento é no sentido direita-direita (*Right-Right*)
  - deste modo, a correção envolve uma rotação para a esquerda



AVL

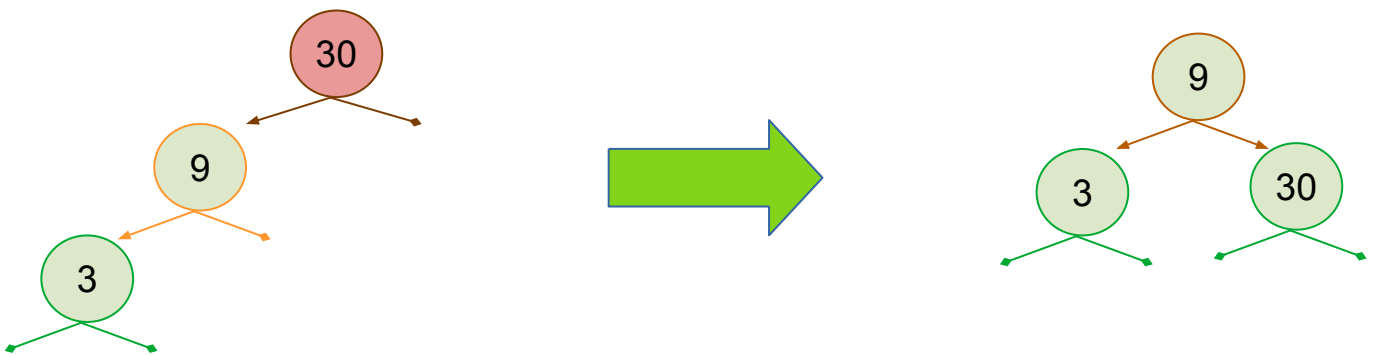
- **Rotação simples à esquerda** ou **Rotação RR**

  - O filho da direita vira a nova raiz
  - A subárvore que antes ficava à esquerda da nova raiz passa para a direita da antiga raiz
  - A antiga raiz vira o filho da esquerda da nova raiz



AVL

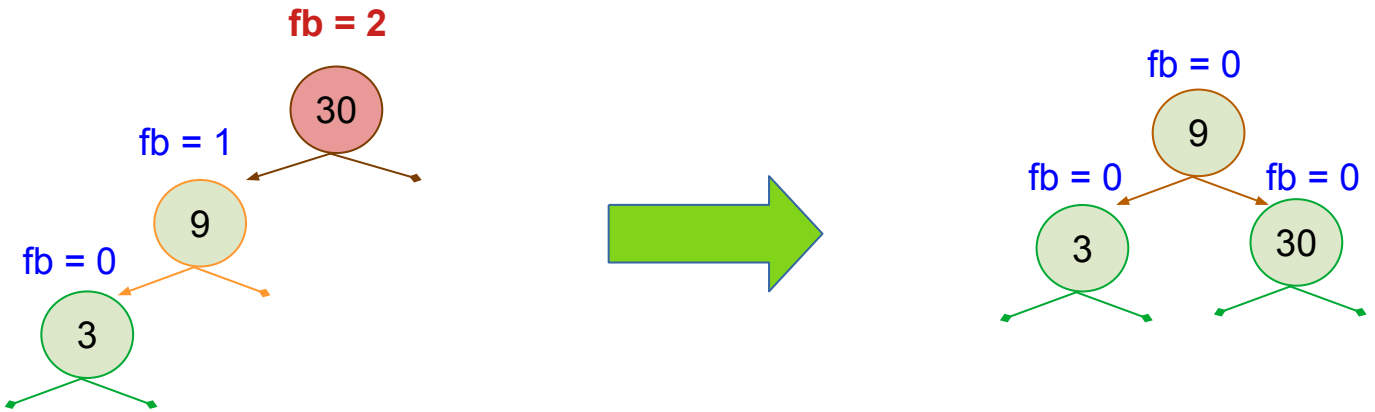
- **Rotação simples à direita** ou **Rotação LL**
  - o desbalanceamento é no sentido esquerda-esquerda (*Left-Left*)
  - deste modo, a correção envolve uma rotação para a direita



AVL

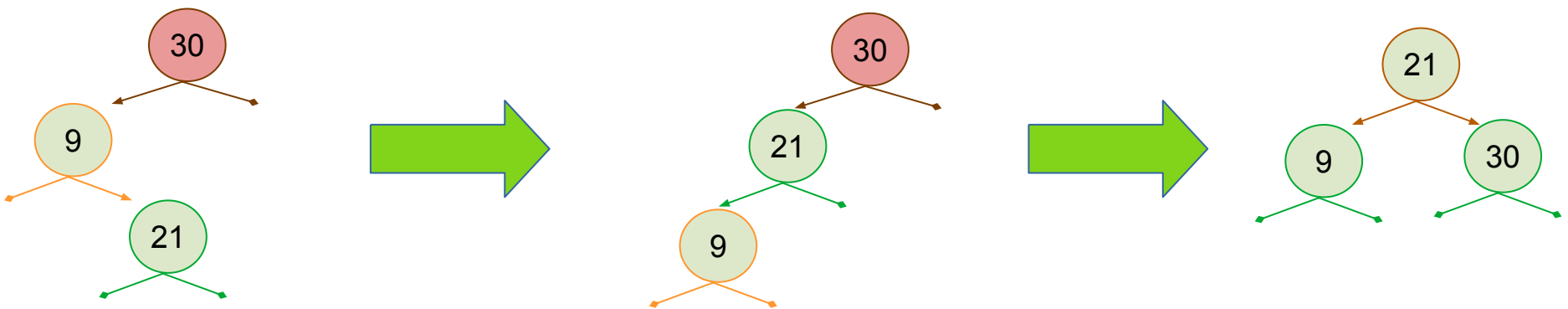
- **Rotação simples à direita** ou **Rotação LL**

  - O filho da esquerda vira a nova raiz
  - A subárvore que antes ficava à direita da nova raiz passa para a esquerda da antiga raiz
  - A antiga raiz vira o filho da direita da nova raiz



AVL

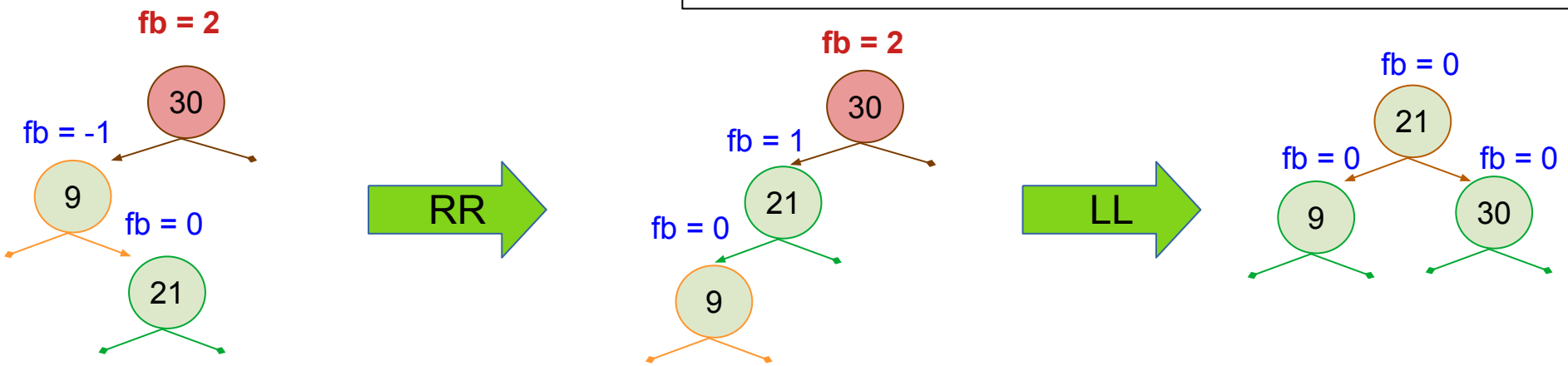
- **Rotação dupla esquerda-direita** ou **Rotação LR**
  - o desbalanceamento é no sentido esquerda-direita (*Left-Right*)
  - deste modo, a correção envolve uma rotação para a esquerda no nó filho, seguida de uma rotação para a direita no nó desbalanceado



AVL

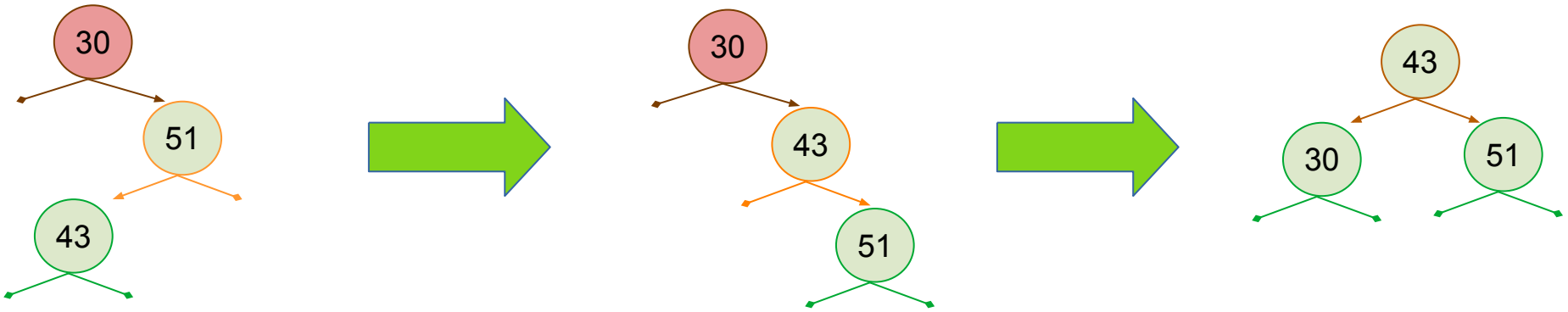
- **Rotação dupla esquerda-direita** ou **Rotação LR**

  - Chama a rotação simples à esquerda (RR) para o filho da esquerda do nó desbalanceado
  - Chama a rotação simples à direita (LL) para o nó desbalanceado



AVL

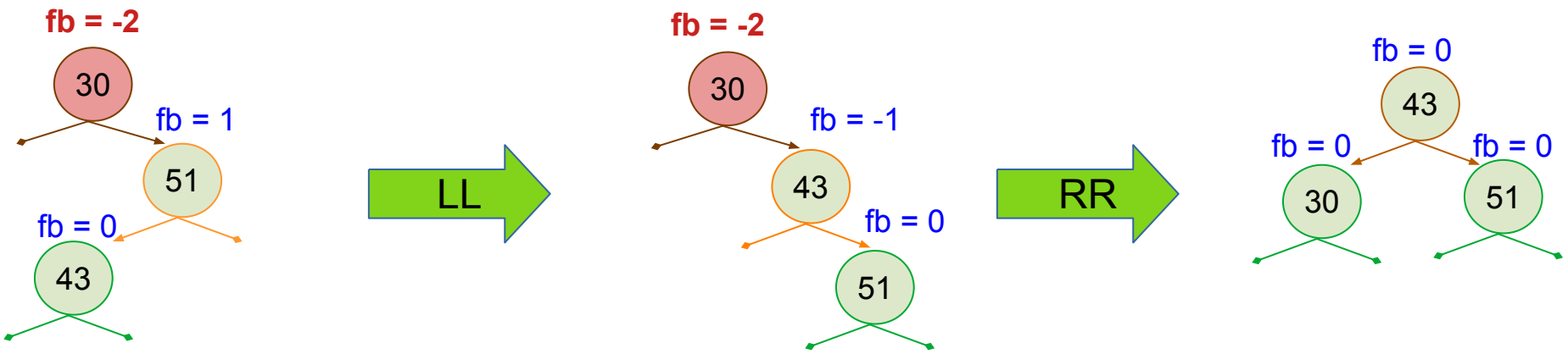
- **Rotação dupla direita-esquerda** ou **Rotação RL**
  - o desbalanceamento é no sentido direita-esquerda (*Right-Left*)
  - deste modo, a correção envolve uma rotação para a direita no nó filho, seguida de uma rotação para a esquerda do nó desbalanceado



AVL

- **Rotação dupla direita-esquerda** ou **Rotação RL**

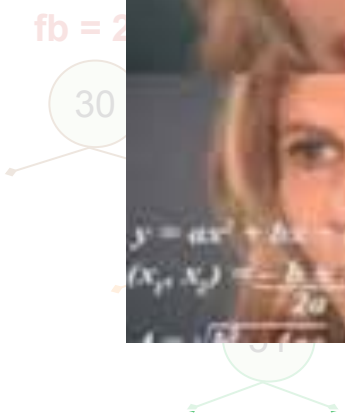
  - Chama a rotação simples à direita (LL) para o filho da direita do nó desbalanceado
  - Chama a rotação simples à esquerda (RR) para o nó desbalanceado



AVL

- **Rotação dupla direita-esquerda** ou **Rotação RL**

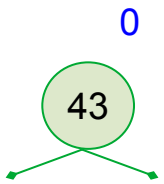
  - Chama a rotação simples à direita (LL) para o filho da direita do nó desbalanceado
  - Ch
    o



Árvore AVL (Ex)

atual

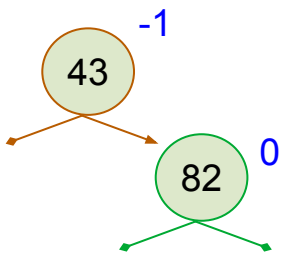
43	82	99	14	21	30	3	51	9	27
0	1	2	3	4	5	6	7	8	9



Árvore AVL (Ex)

atual

43	82	99	14	21	30	3	51	9	27
0	1	2	3	4	5	6	7	8	9

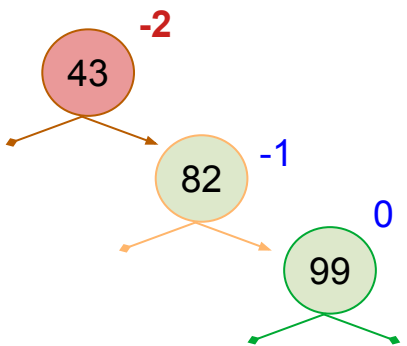


Árvore AVL (Ex)

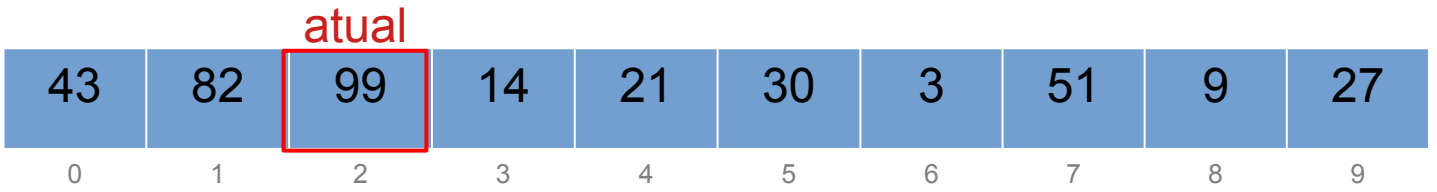
atual

43	82	99	14	21	30	3	51	9	27
0	1	2	3	4	5	6	7	8	9

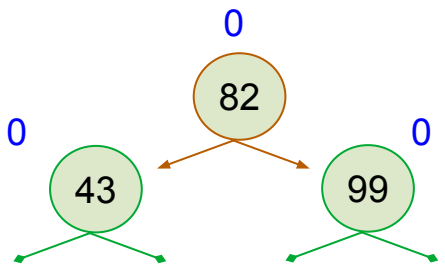
Rotação RR



Árvore AVL (Ex)



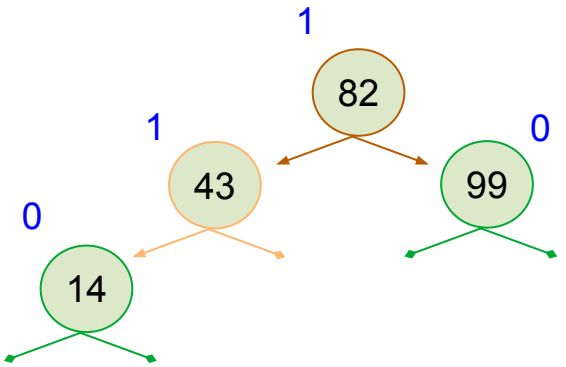
Rotação RR



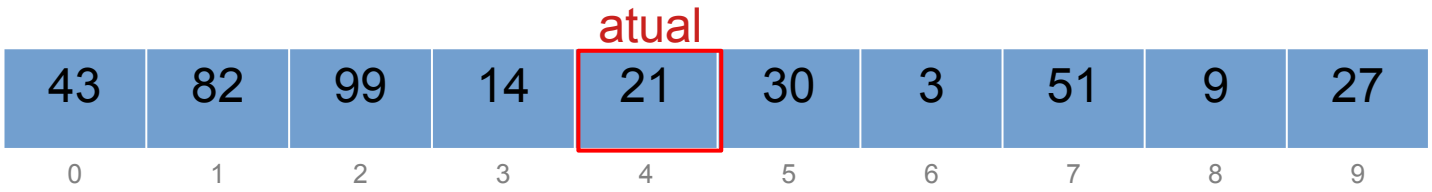
Árvore AVL (Ex)

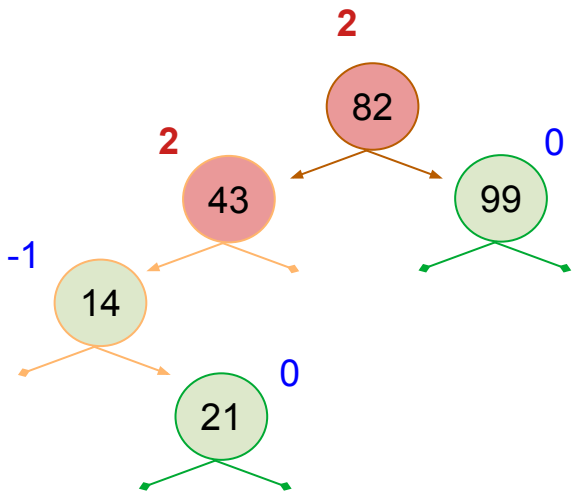
atual

43	82	99	14	21	30	3	51	9	27
0	1	2	3	4	5	6	7	8	9

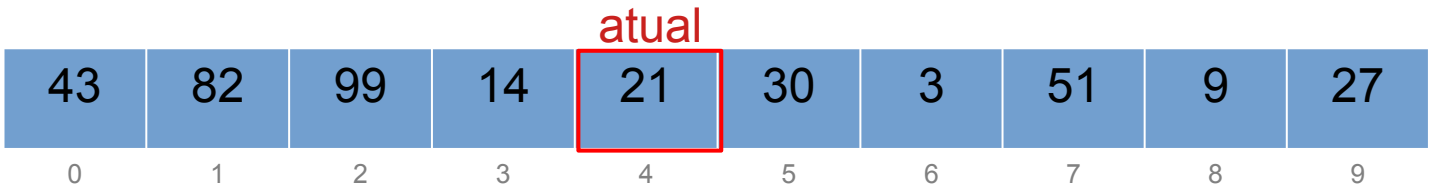


Árvore AVL (Ex)



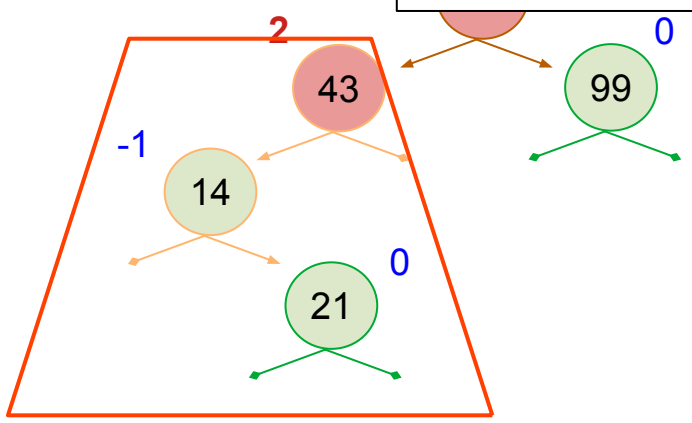


Árvore AVL (Ex)

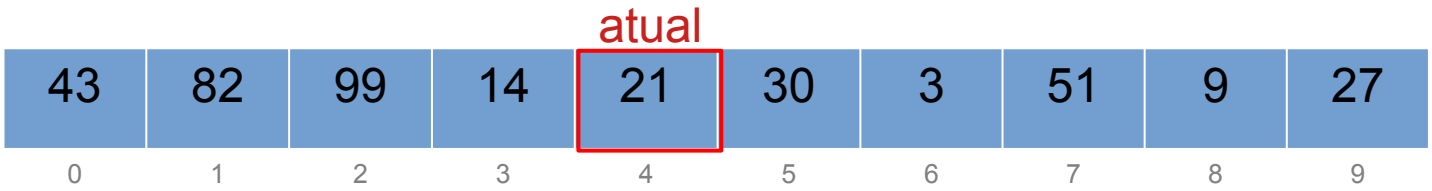


Rotação Left-Right (LR)

- Chama a rotação simples à esquerda (RR) para o filho da esquerda do nó desbalanceado
- Chama a rotação simples à direita (LL) para o nó desbalanceado

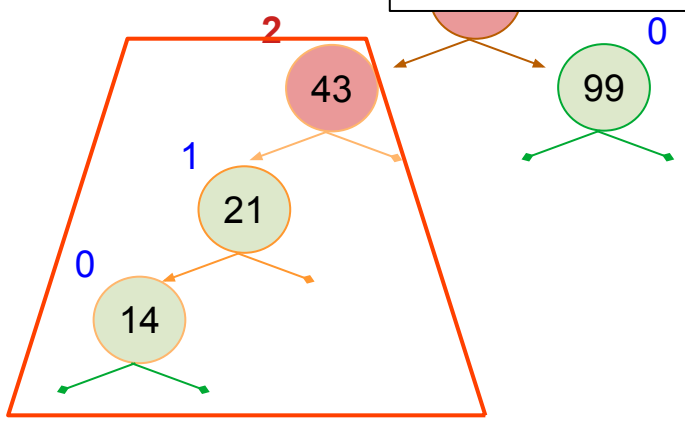


Árvore AVL (Ex)

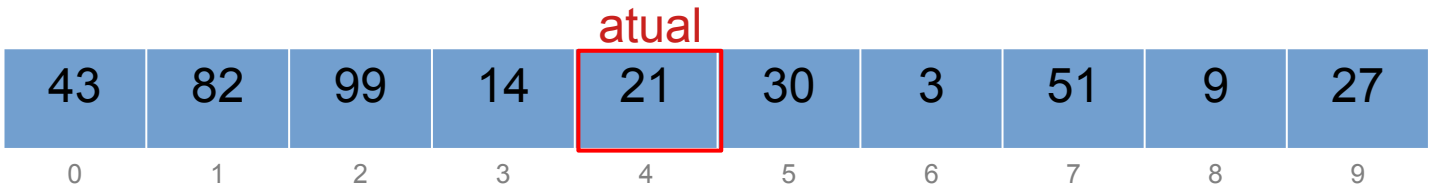


Rotação Left-Right (LR)

- Chama a rotação simples à esquerda (RR) para o filho da esquerda do nó desbalanceado
- Chama a rotação simples à direita (LL) para o nó desbalanceado

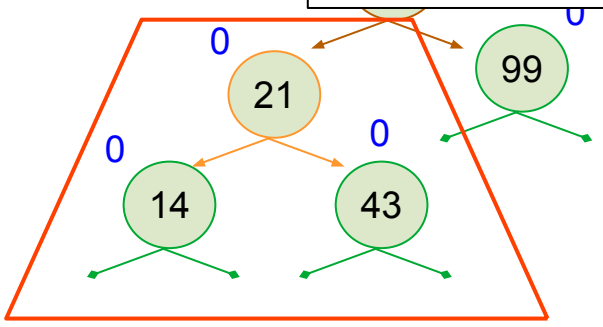


Árvore AVL (Ex)



Rotação Left-Right (LR)

- Chama a rotação simples à esquerda (RR) para o filho da esquerda do nó desbalanceado
- Chama a rotação simples à direita (LL) para o nó desbalanceado

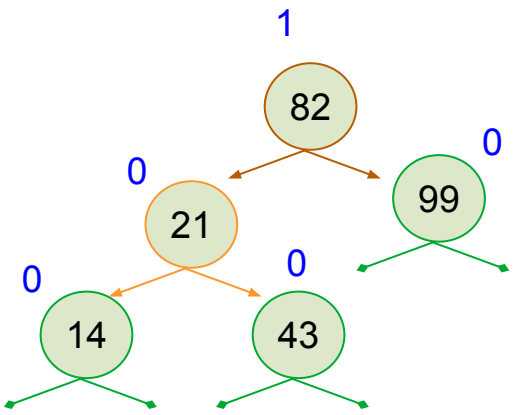


Árvore AVL (Ex)

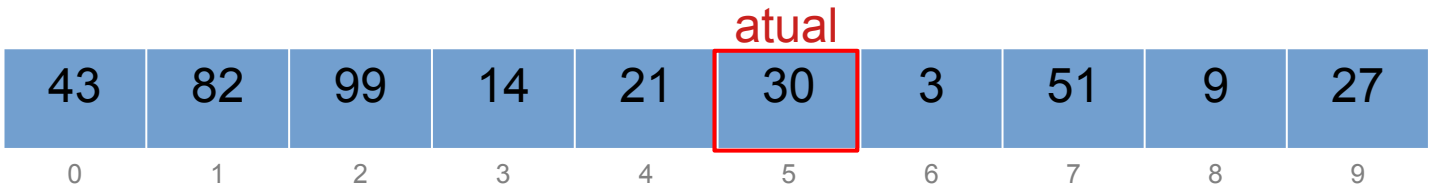
atual

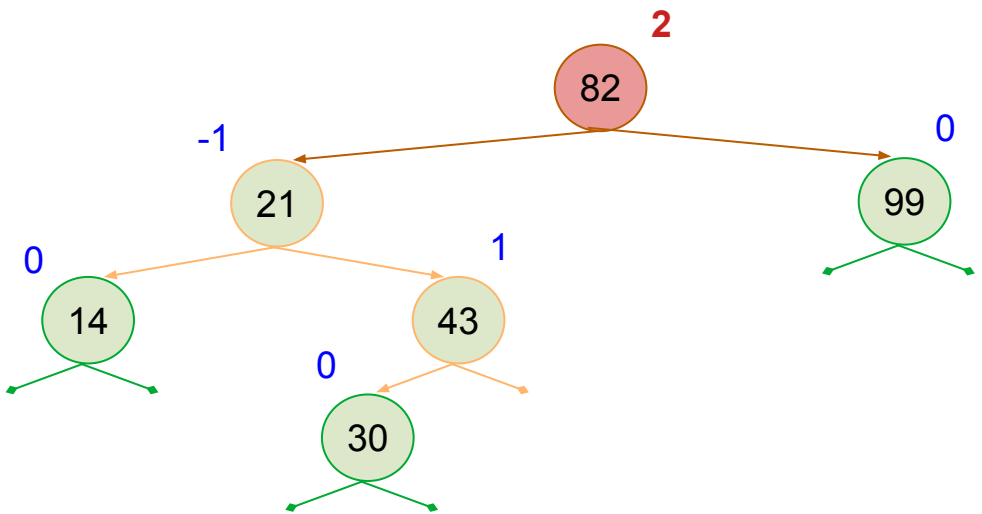
43	82	99	14	21	30	3	51	9	27
0	1	2	3	4	5	6	7	8	9

Rotação Left-Right (LR)

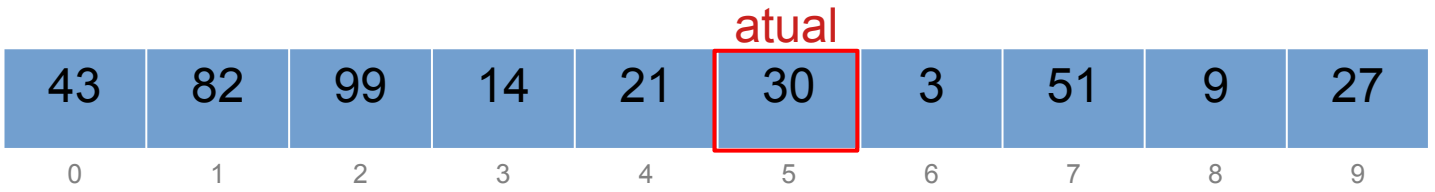


Árvore AVL (Ex)

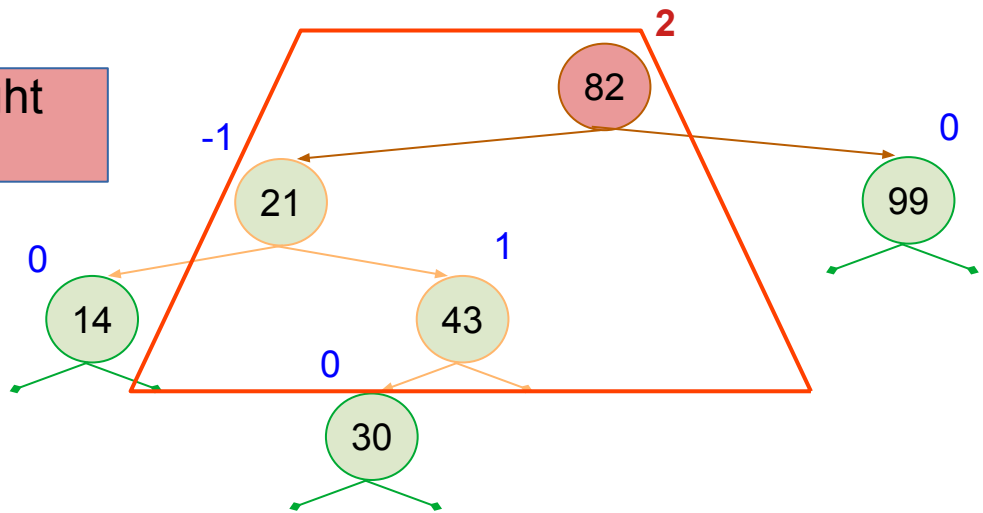




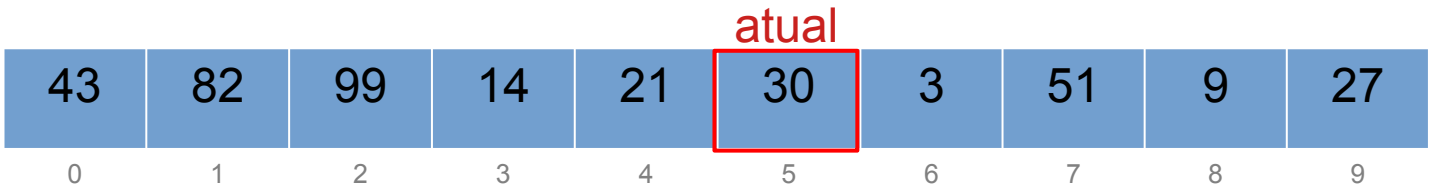
Árvore AVL (Ex)



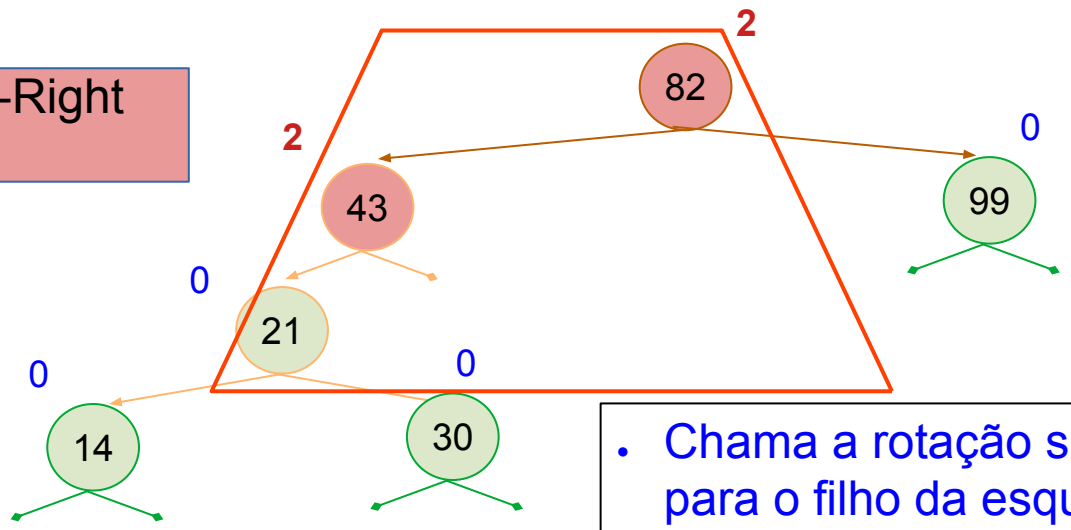
Rotação Left-Right (LR)



Árvore AVL (Ex)

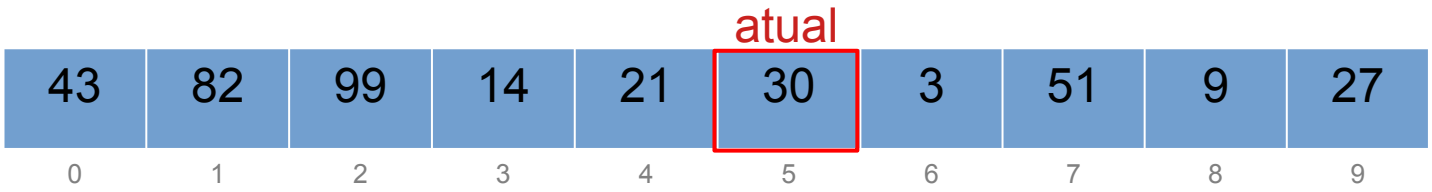


Rotação Left-Right (LR)

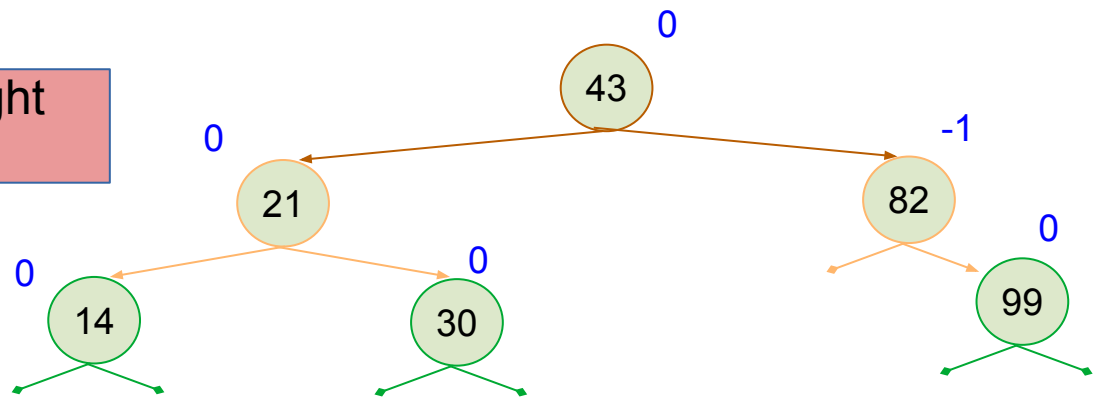


- Chama a rotação simples à esquerda (RR) para o filho da esquerda do nó desbalanceado
- Chama a rotação simples à direita (LL) para o nó desbalanceado

Árvore AVL (Ex)

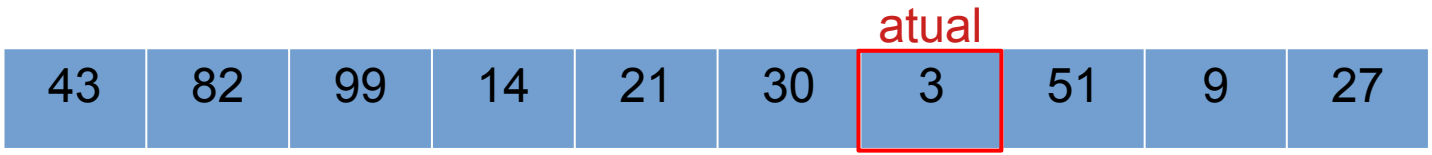


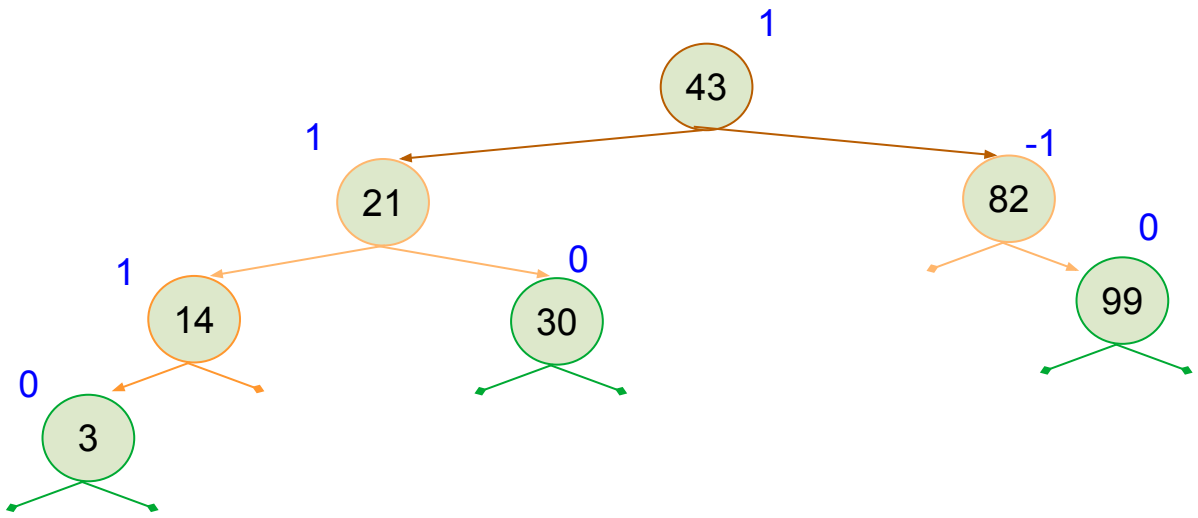
Rotação Left-Right (LR)



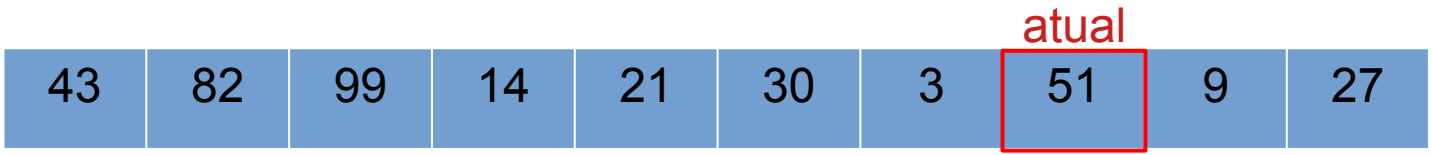
- Chama a rotação simples à esquerda (RR) para o filho da esquerda do nó desbalanceado
- Chama a rotação simples à direita (LL) para o nó desbalanceado

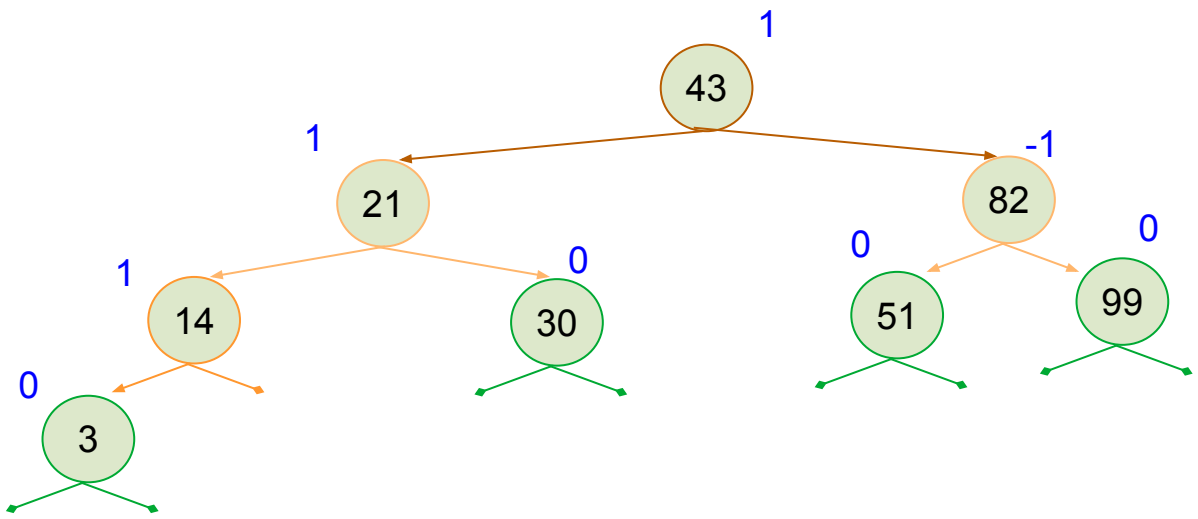
Árvore AVL (Ex)



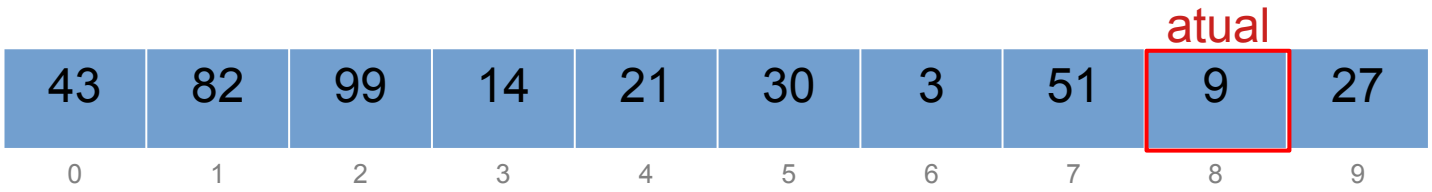


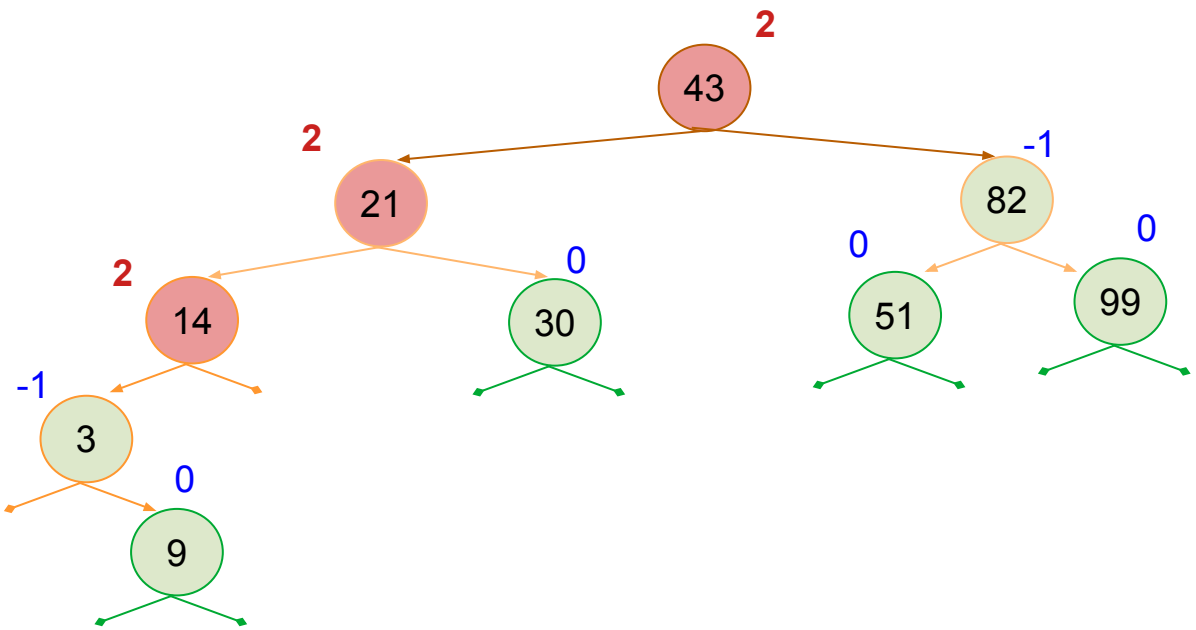
Árvore AVL (Ex)



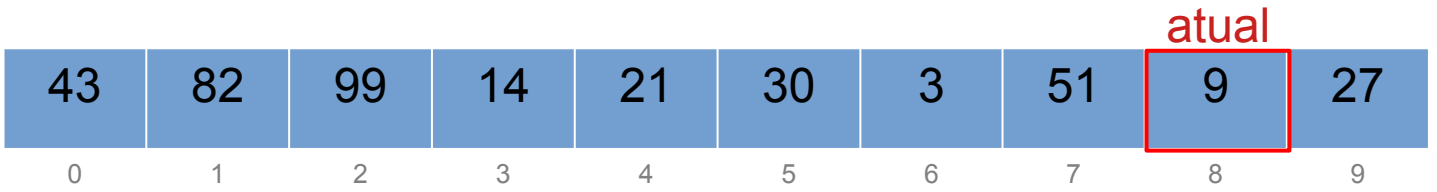


Árvore AVL (Ex)

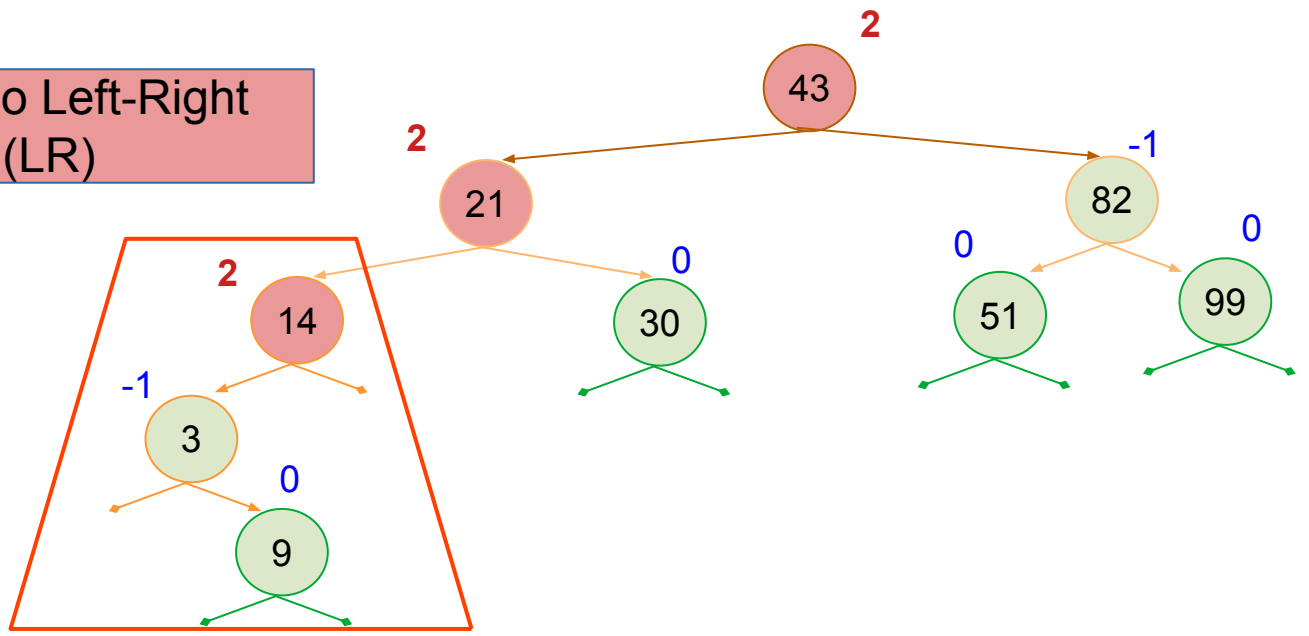




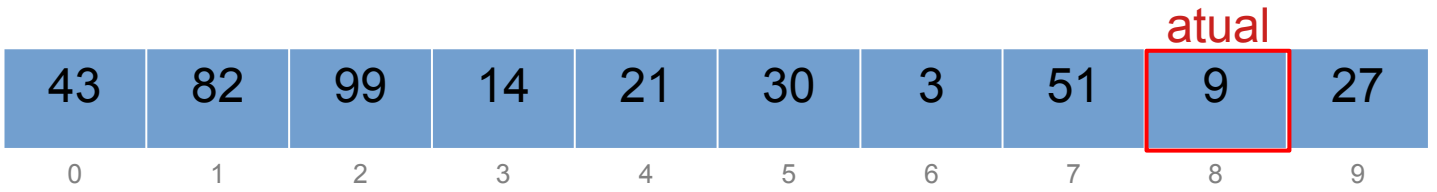
Árvore AVL (Ex)



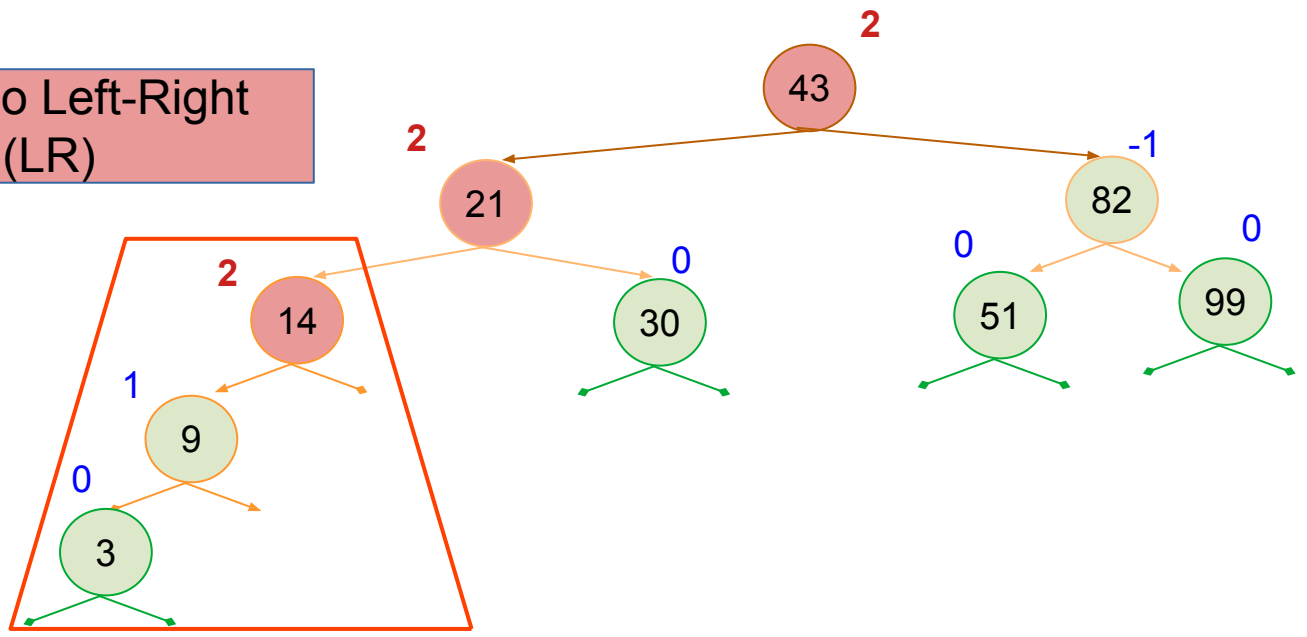
Rotação Left-Right (LR)



Árvore AVL (Ex)



Rotação Left-Right (LR)

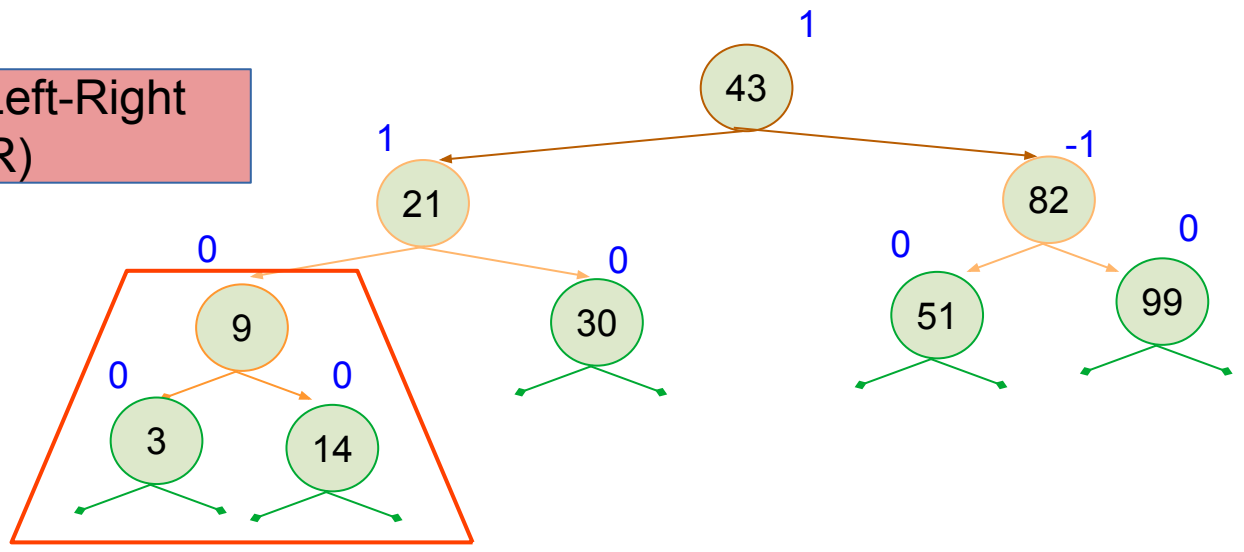


Árvore AVL (Ex)

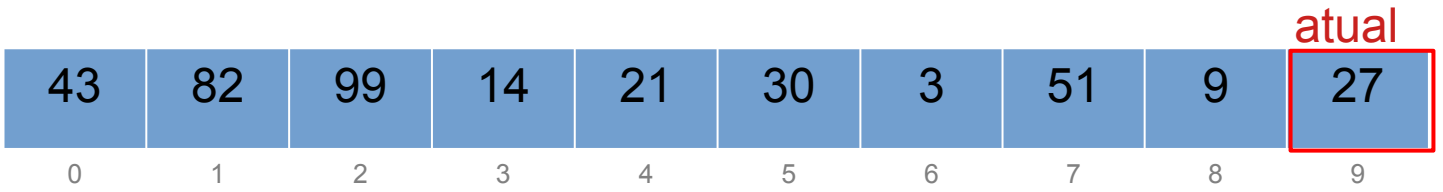
atual

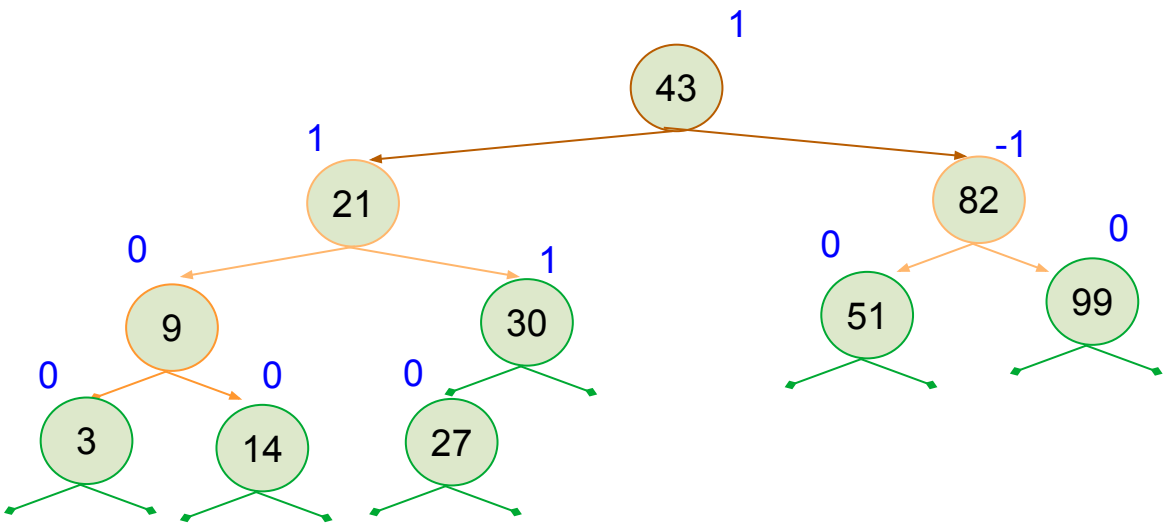
43	82	99	14	21	30	3	51	9	27
0	1	2	3	4	5	6	7	8	9

Rotação Left-Right (LR)



Árvore AVL (Ex)

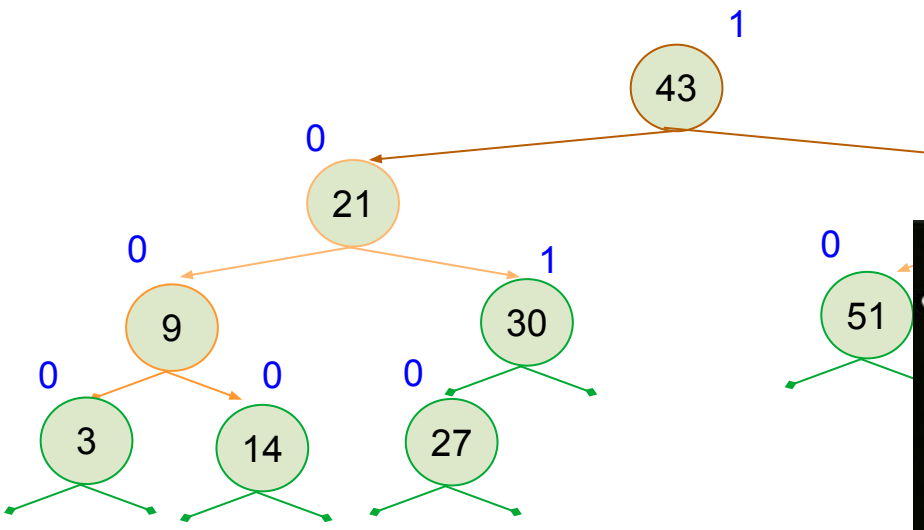




Árvore AVL (Ex)

atual

43	82	99	14	21	30	3	51	9	27
0	1	2	3	4	5	6	7	8	9



AVL

- A cada alteração feita na árvore (inserção ou exclusão) deve ser feita a verificação da regra de balanceamento:
  - Se a partir de um nodo a diferença de altura de suas subárvores for menor que -1 ou maior que 1, o sistema deve balancear a partir do nodo em questão
  - Com base no nodo onde foi detectado o desbalanceamento, deve ser selecionada e aplicada a regra correta de balanceamento

AVL

- Inserção
  - Busca na árvore a posição onde o elemento será adicionado
    - Percorre a árvore em pré-order
  - Após realizar a inserção do novo nó, retorna
  - Calcula o fator de balanceamento para o pai do nó inserido
    - Verifica a altura da subárvore da esquerda e subtraia a altura da direita
    - Se o **fator > 1** ou **fator < -1** o sistema deve ser balanceado
  - Caso o desbalanceamento seja detectado, verifica qual a regra de balanceamento que deve ser usada e a executa
  - Após balanceado, o nó retorna para seu pai, calcula-se o fator de balanceamento...
    - Isso repete-se até que retorne à raiz

AVL

- Detectamos o desbalanceamento, qual regra utilizar?

- Se $fator_{balanceamento}$(pai) é positivo
  - Calcula o fator de balanceamento do filho da esquerda
    - Caso seja negativo, use LR
    - Caso seja positivo, use LL
- Se $fator_{balanceamento}$(pai) é negativo
  - Calcula o fator de balanceamento do filho da direita
    - Caso seja negativo, use RR
    - Caso seja positivo, use RL

AVL

- Exclusão
  - Busca na árvore a posição onde o elemento será removido
    - Percorre a árvore em pré-order
  - Para a exclusão temos 3 situações a serem consideradas
    - Excluindo uma folha
    - Excluindo um nó com um filho
    - Excluindo um nó com dois filhos

AVL

- Exclusão
  - Busca na árvore a posição onde o elemento será removido
    - Percorre a árvore em pré-order
  - Para a exclusão temos 3 situações a serem consideradas
    - Excluindo uma folha
      - Simples: só dar free no nó e retornar NULL
    - Excluindo um nó com um filho
    - Excluindo um nó com dois filhos

AVL

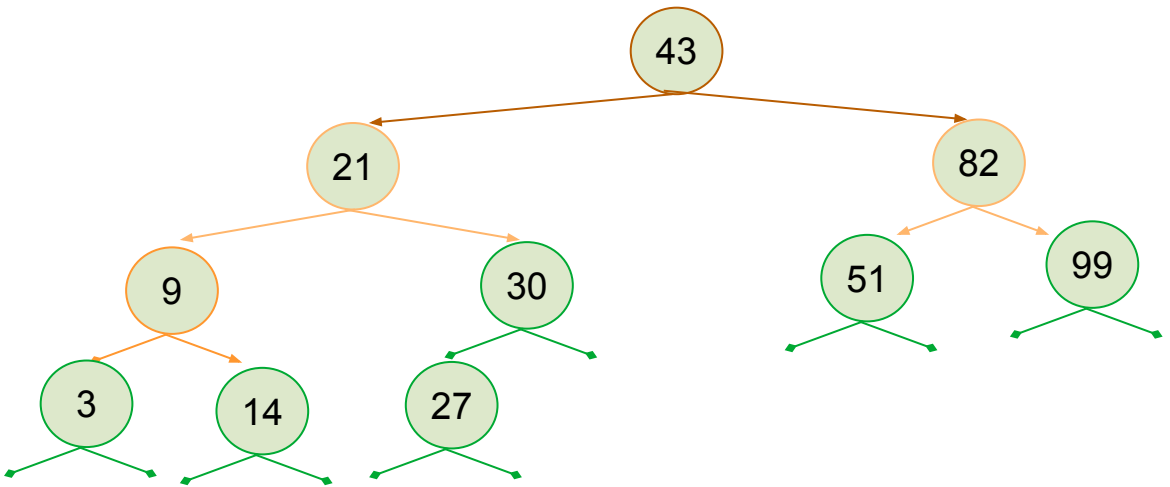
- Exclusão
  - Busca na árvore a posição onde o elemento será removido
    - Percorre a árvore em pré-order
  - Para a exclusão temos 3 situações a serem consideradas
    - Excluindo uma folha
      - Simples só dar free no nó e retornar NULL
    - Excluindo um nó com um filho
      - Remove o elemento com free, e retorna seu filho
    - Excluindo um nó com dois filhos

AVL

- Exclusão
  - Busca na árvore a posição onde o elemento será removido
    - Percorre a árvore em pré-order
  - Para a exclusão temos 3 situações a serem consideradas
    - Excluindo uma folha
      - Simples só dar free no nó e retornar NULL
    - Excluindo um nó com um filho
      - Remove o elemento com free, e retorna seu filho
    - Excluindo um nó com dois filhos
      - Mais complexo, precisa promover um dos filhos
      - Promover o maior elemento da esquerda ou o menor elemento da direita

AVL

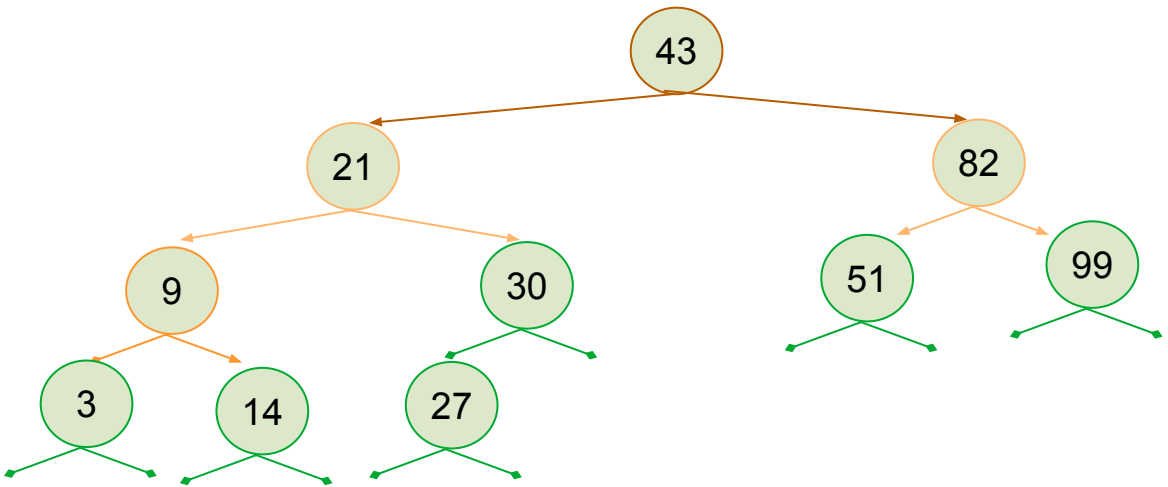
- Exclusão



AVL

- Exclusão

Delete **3**

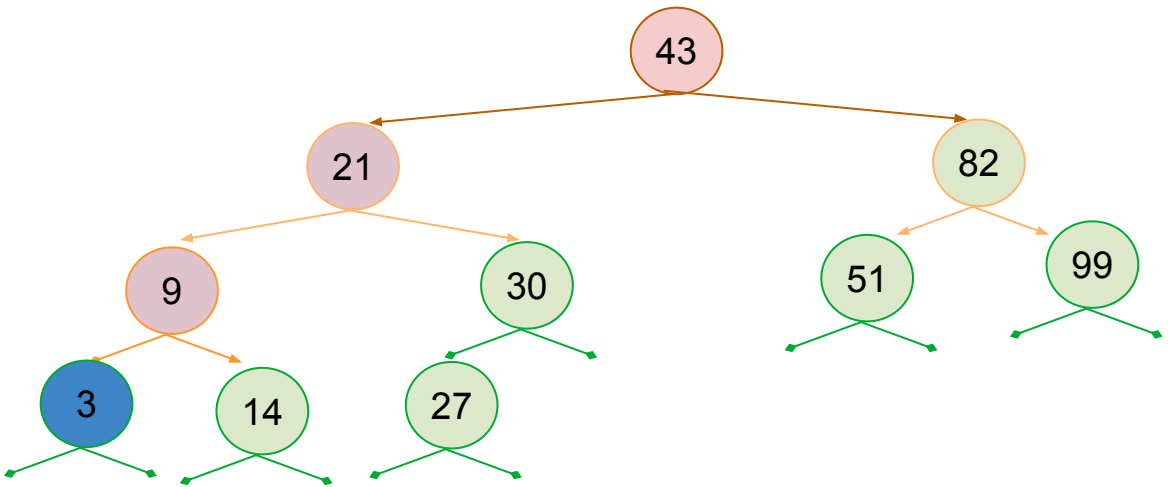


AVL

- Exclusão

- Free(no);
- Retorna NULL

Delete **3**

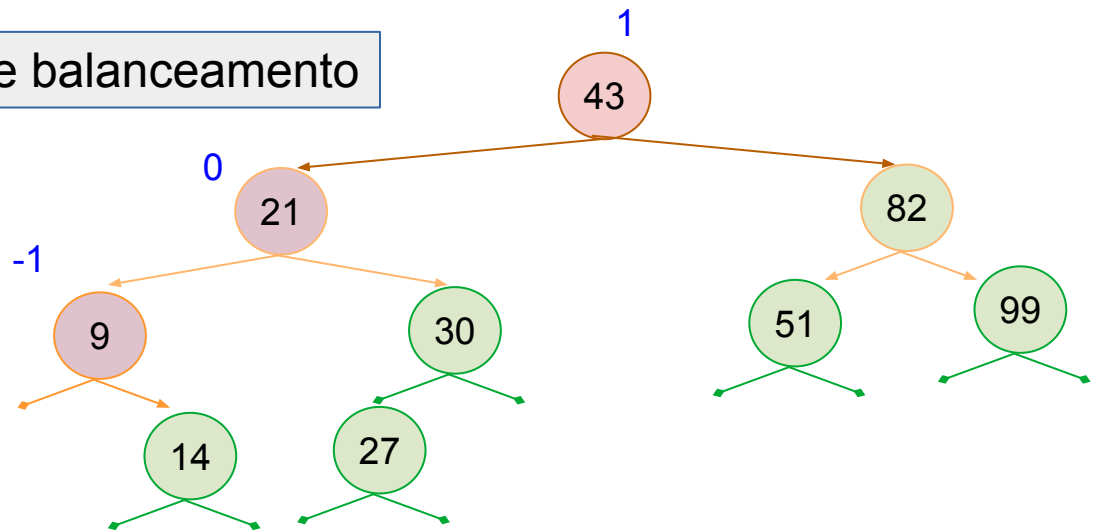


AVL

- Exclusão

Delete **3**

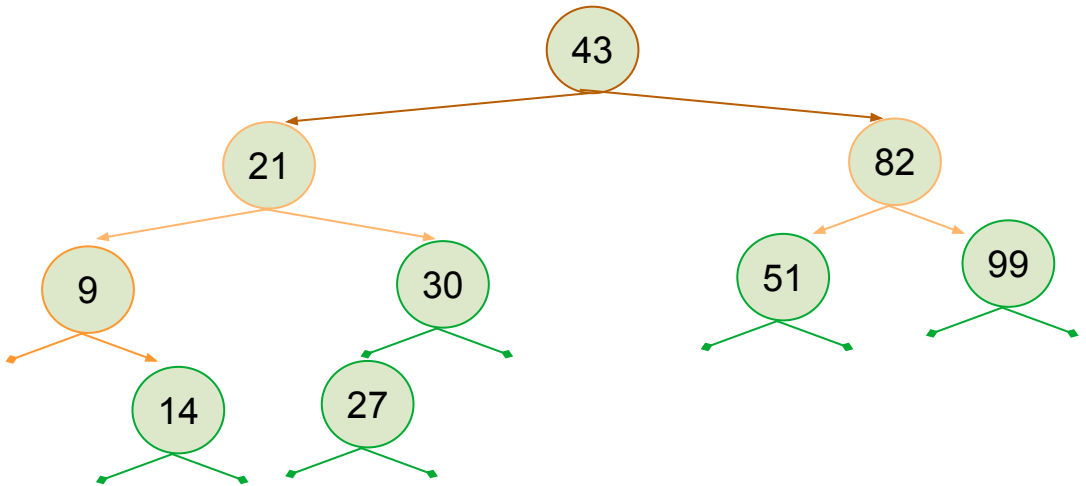
Calcula fator de balanceamento



AVL

- Exclusão

Delete **9**

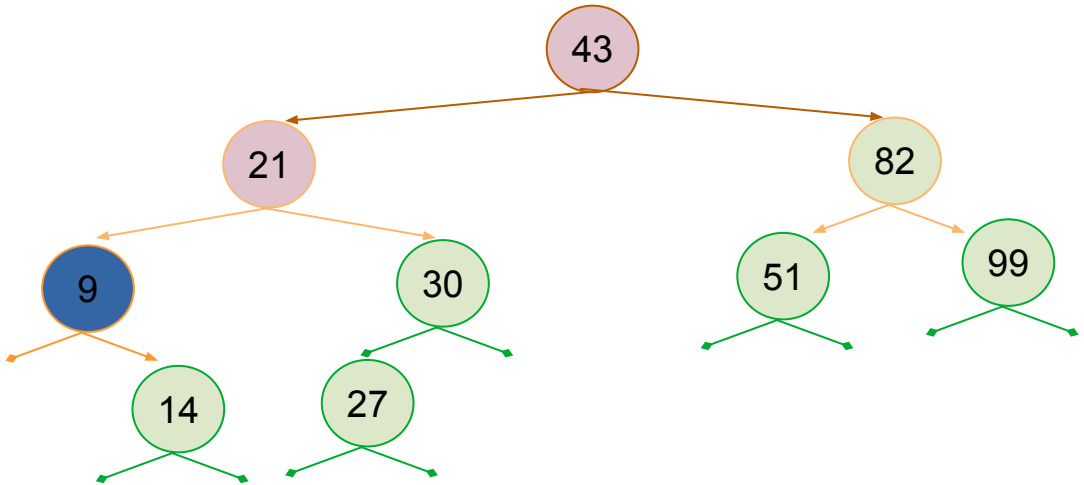


AVL

- Exclusão

Delete **9**

- Free(no);
- Retorna o filho que tiver
  - Direita ou esquerda

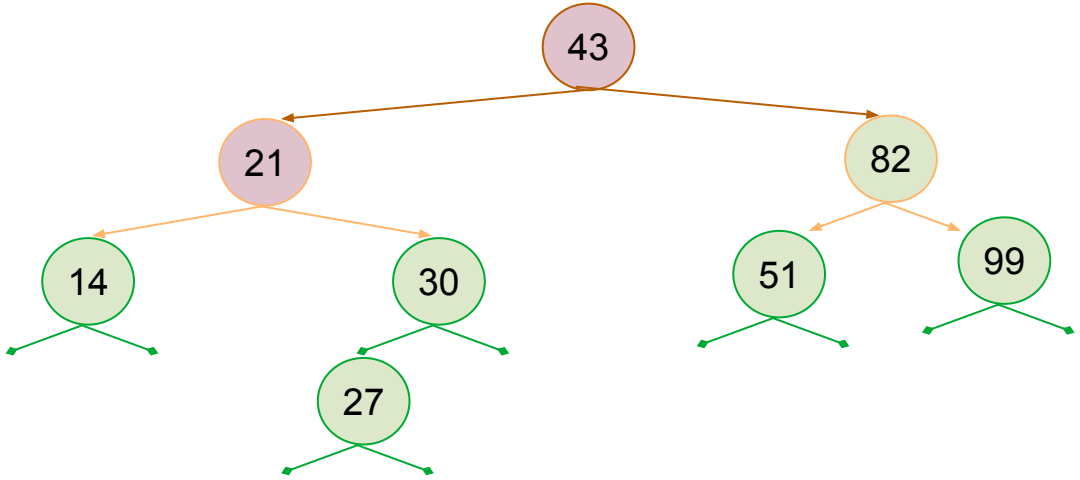


AVL

- Exclusão

Delete **9**

- Free(no);
- Retorna o filho que tiver
  - Direita ou esquerda

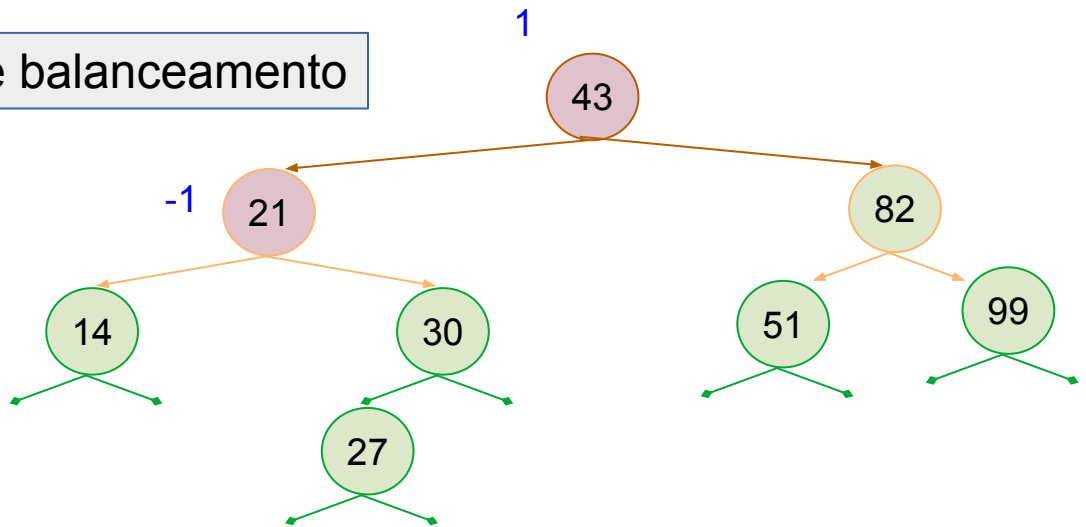


AVL

- Exclusão

Delete **9**

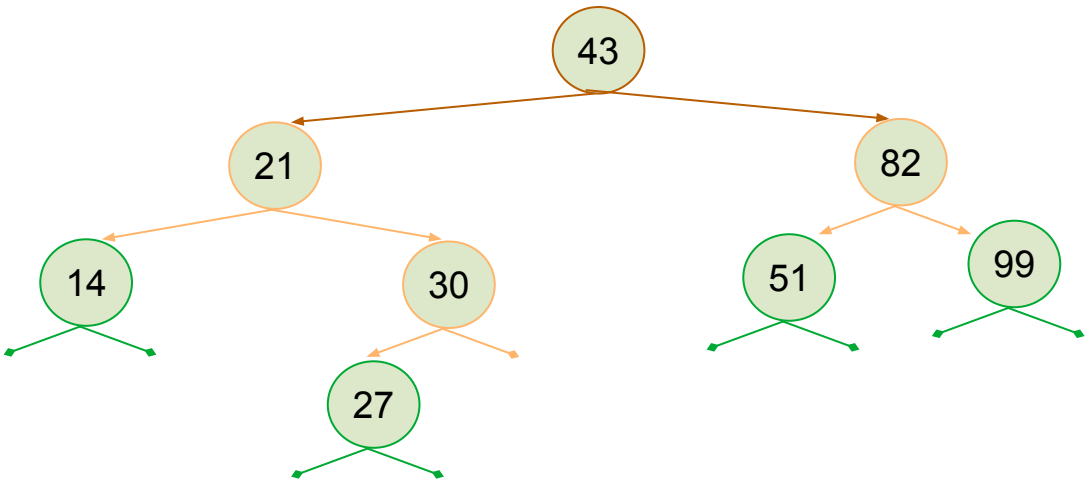
Calcula fator de balanceamento



AVL

- Exclusão

Delete **43**

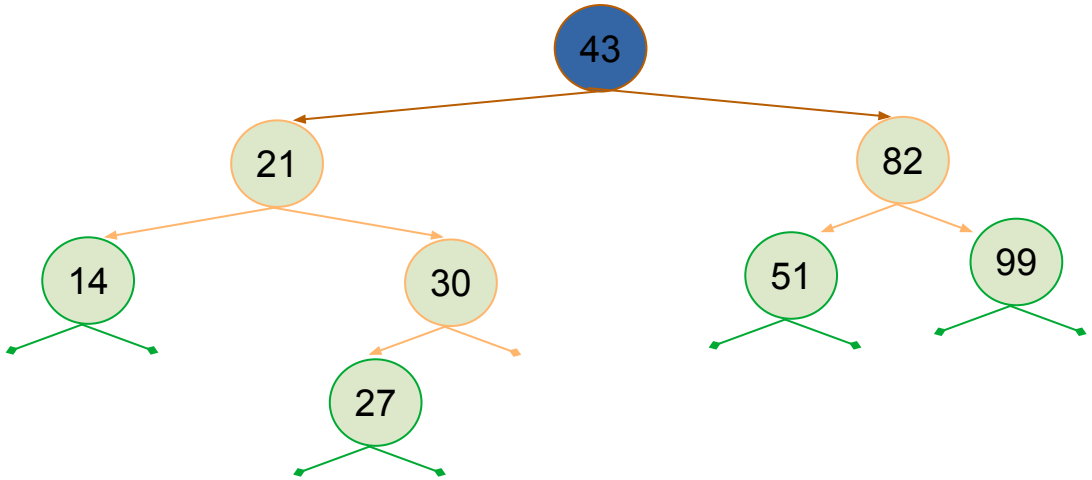


AVL

- Exclusão

Delete **43**

- Busca o maior elemento à sua esquerda
- Promove esse elemento
- free(no)
- Retorna a nova raiz.

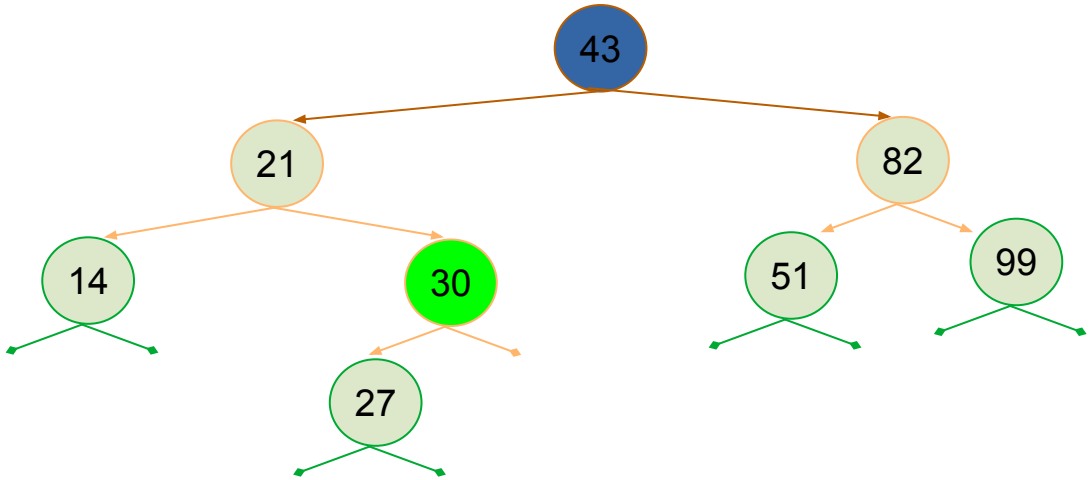


AVL

- Exclusão

Delete **43**

- Busca o maior elemento à sua esquerda
- Promove esse elemento
- free(no)
- Retorna a nova raiz.

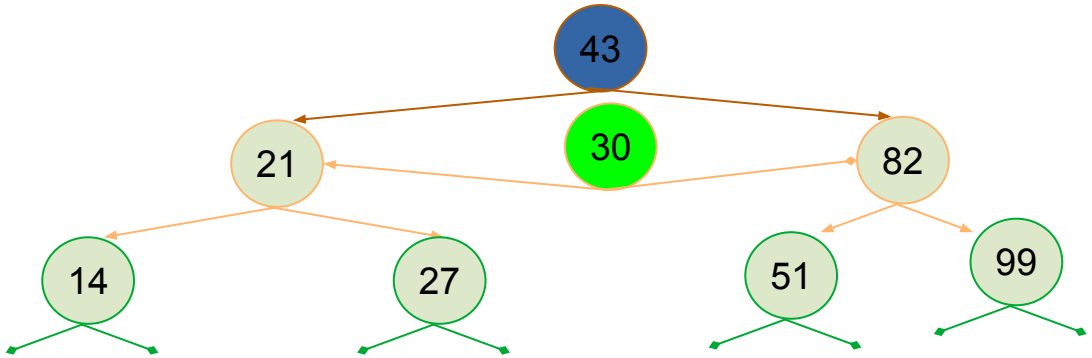


AVL

- Exclusão

Delete **43**

- Busca o maior elemento à sua esquerda
- Promove esse elemento
- free(no)
- Retorna a nova raiz.

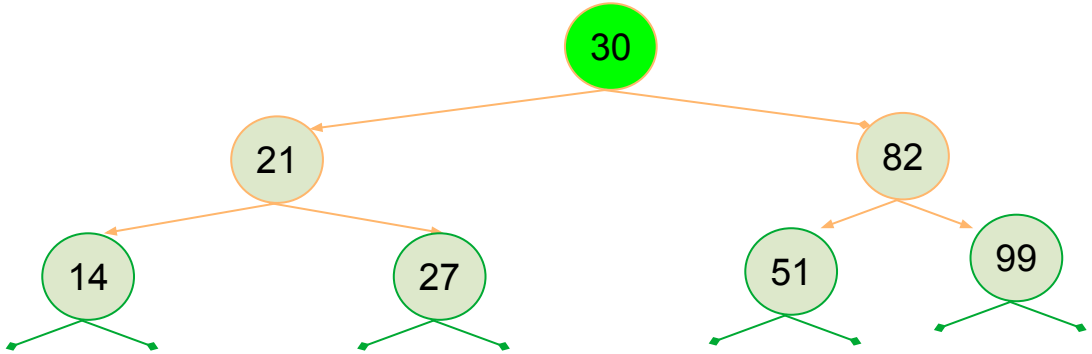


AVL

- Exclusão

Delete **43**

- Busca o maior elemento à sua esquerda
- Promove esse elemento
- free(no)
- Retorna a nova raiz.

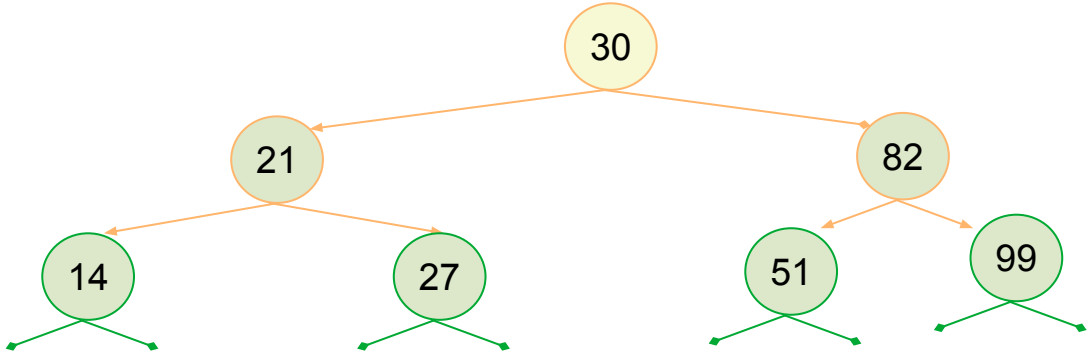


AVL

- Exclusão

Delete **43**

- Busca o maior elemento à sua esquerda
- Promove esse elemento
- free(no)
- Retorna a nova raiz.

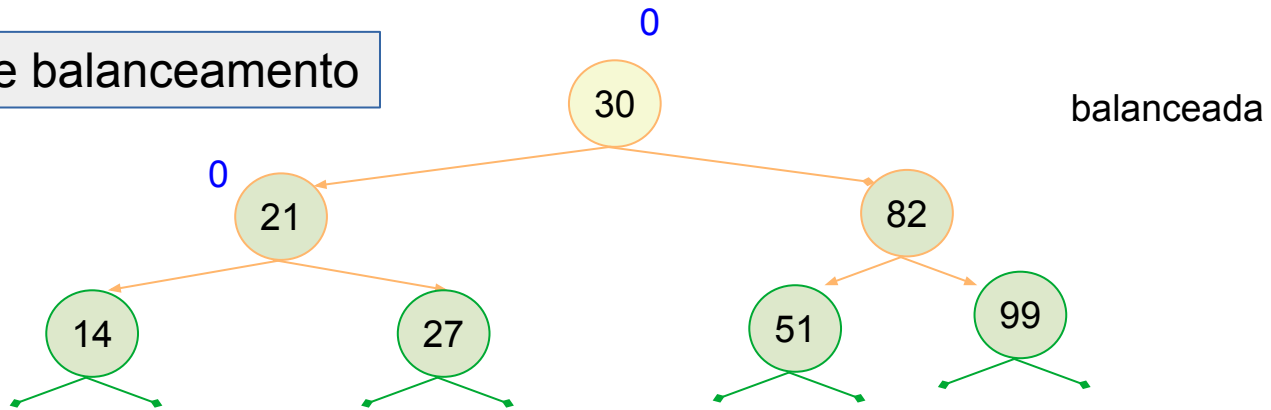


AVL

- Exclusão

Delete **43**

Calcula fator de balanceamento



balanceada

Atividade

https://codeshare.io/8pxrRZ

Código Andressa: https://codeshare.io/PdRBdM