ÁRVORES AVL

Vanessa Braganholo

Estruturas de Dados e Seus Algoritmos

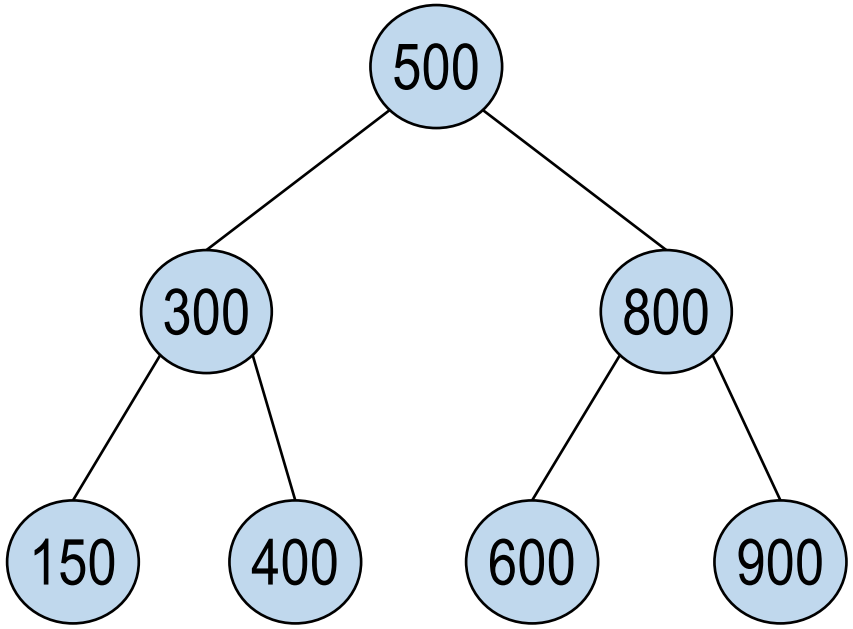
RECAPITULANDO: ÁRVORES BINÁRIAS DE BUSCA

Apresentam uma relação de ordem

A ordem é definida pela chave

Operações:

- inserir
- consultar
- excluir

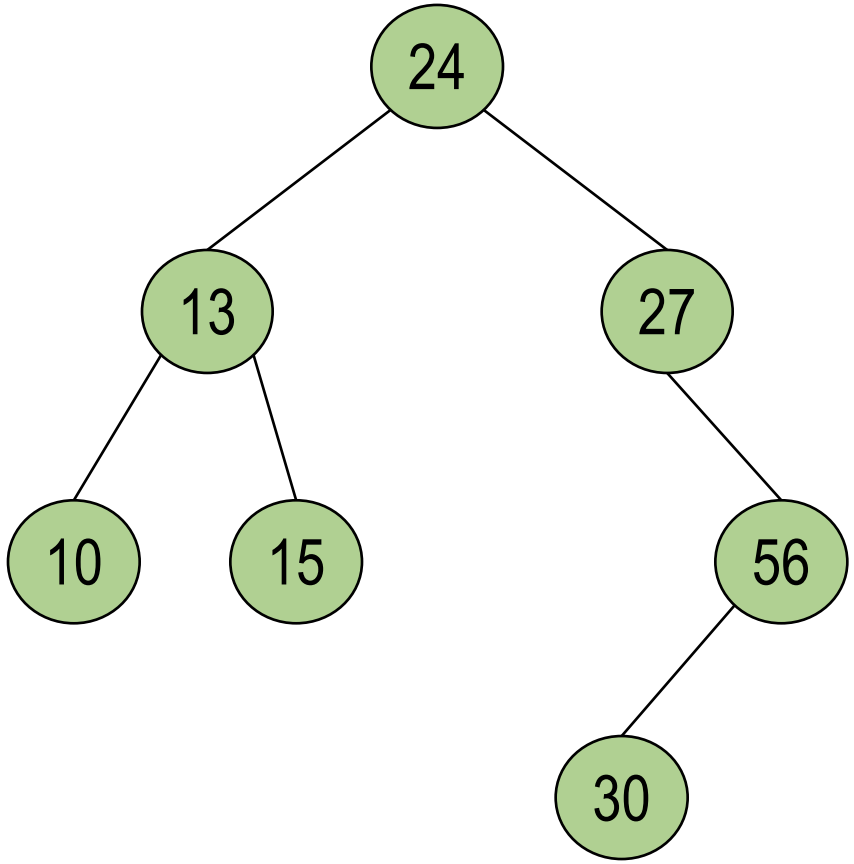


PROBLEMAS COM ÁRVORE BINÁRIA DE BUSCA (ABB)

Desbalanceamento progressivo

Exemplo:

- Inserção: 24, 27, 13, 10, 56, 15, 30

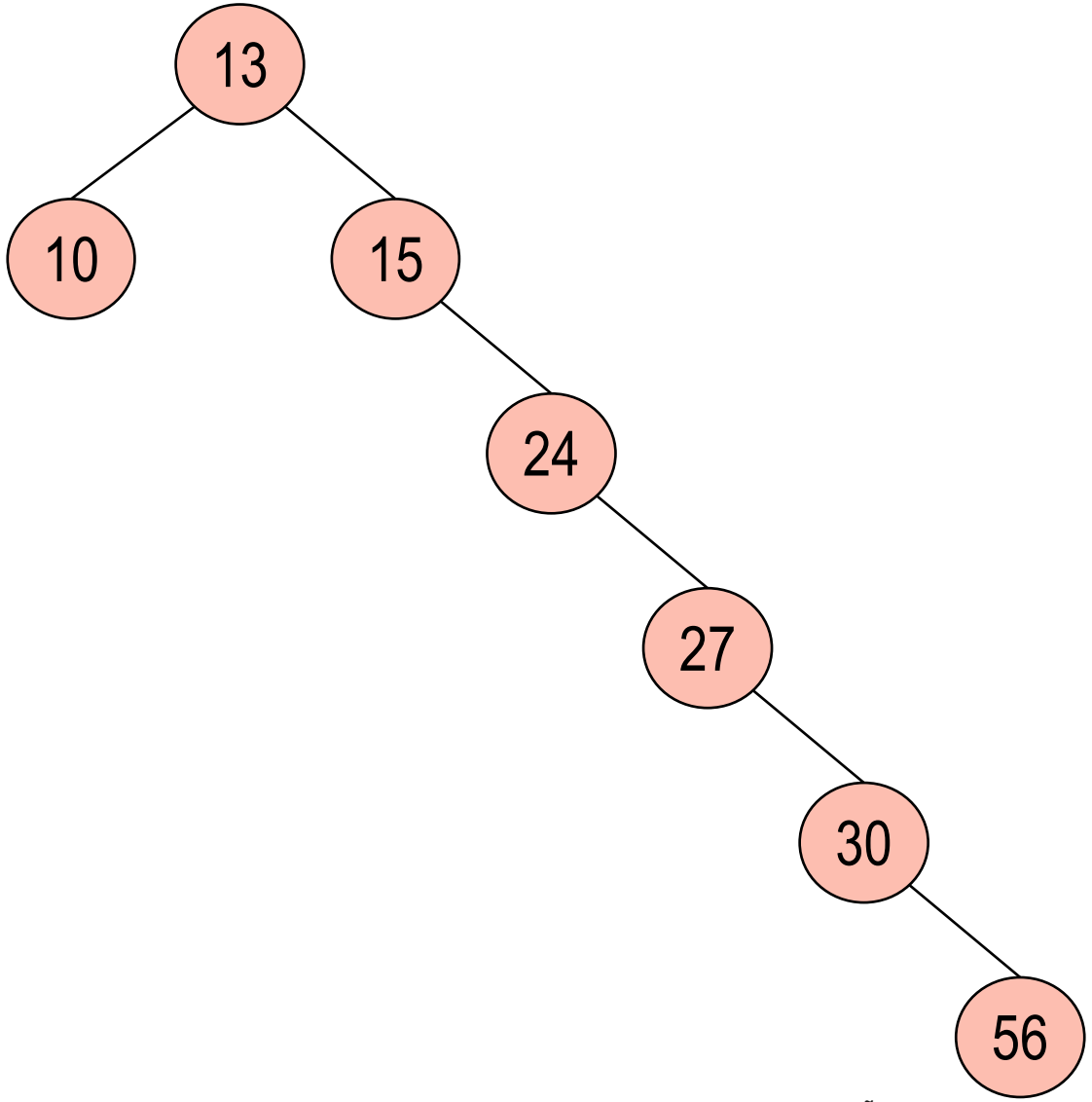


PROBLEMAS COM ABB

Desbalanceamento progressivo

Exemplo:

- Inserção: 13, 10, 15, 24, 27, 30, 56

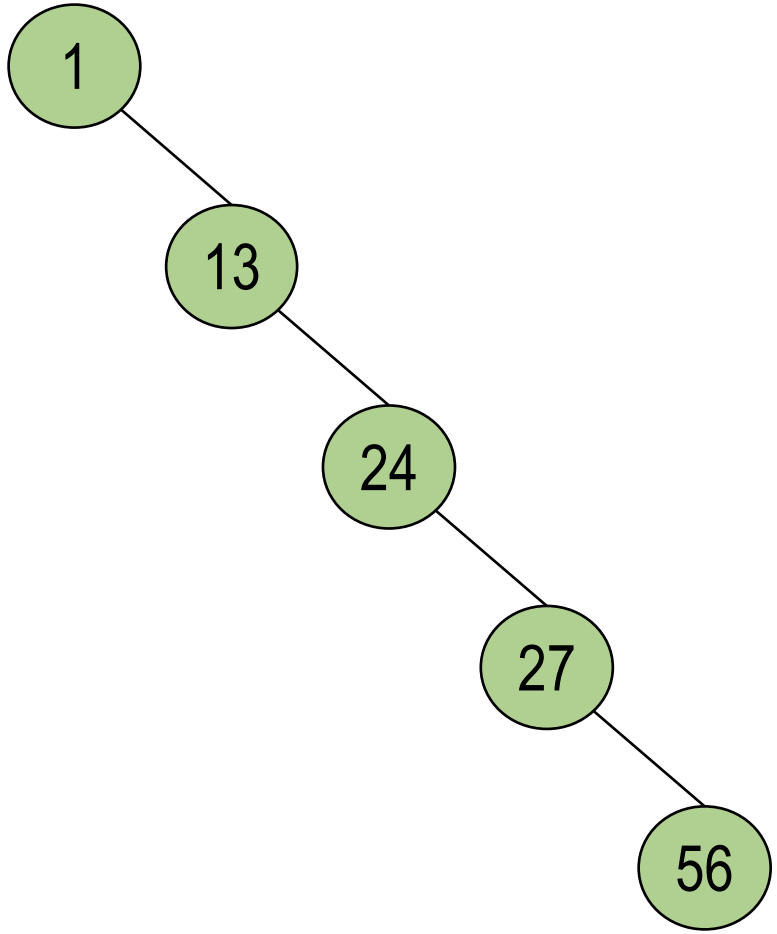


PROBLEMAS COM ABB

Desbalanceamento progressivo

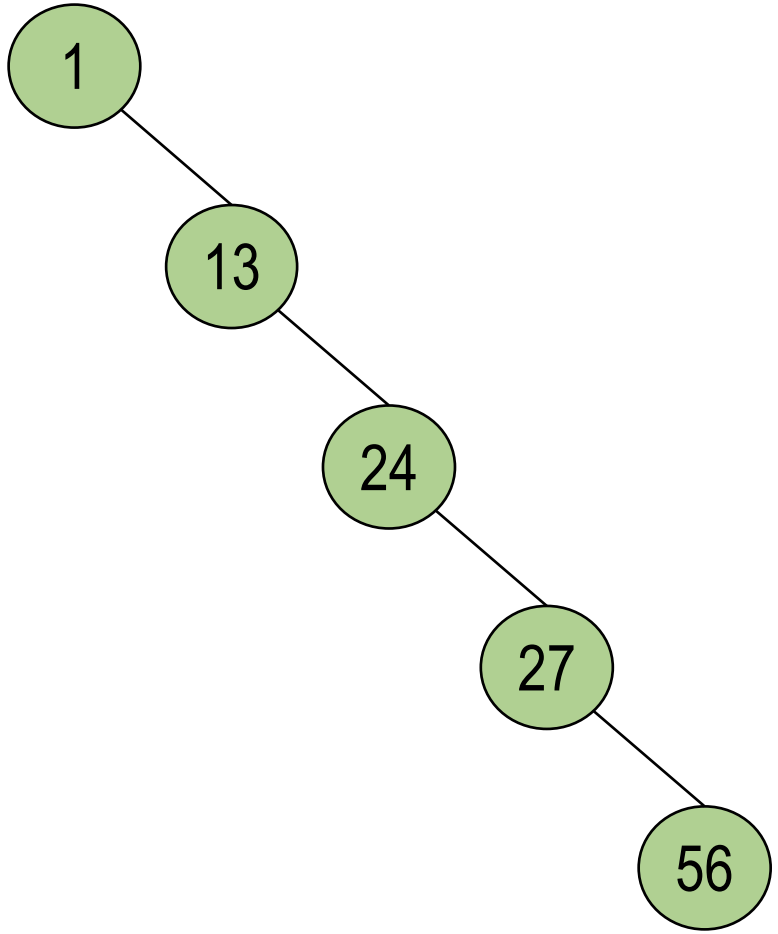
Exemplo:

- inserção: 1, 13, 24, 27, 56



CONSEQUÊNCIA

Buscas ficam mais custosas



BALANCEAMENTO DE ÁRVORES

Distribuição equilibrada dos nós

Objetivo:

- Otimizar as operações de consulta
- Diminuir o número médio de comparações

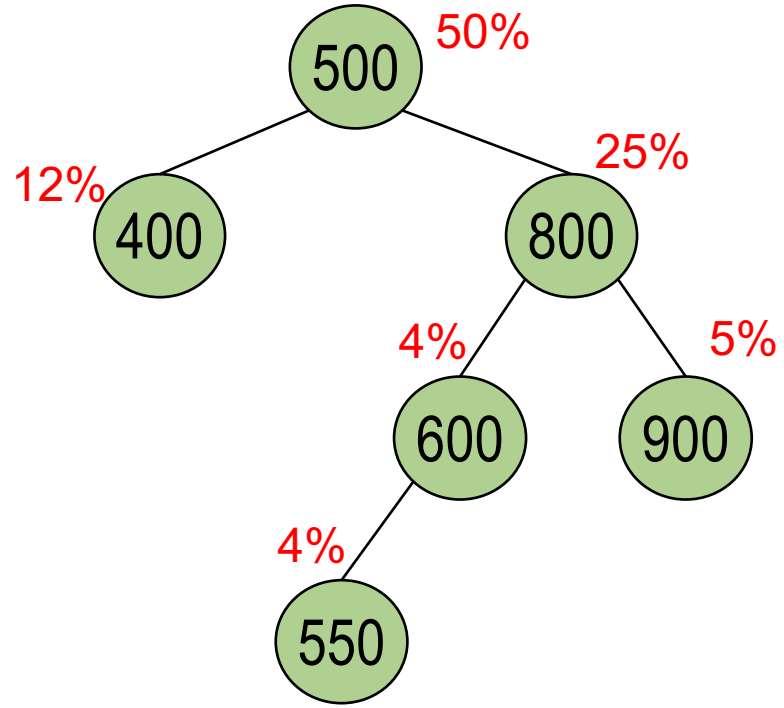
Distribuição

- Uniforme
  - árvore balanceada por altura (distância entre as alturas dos nodos não deve exceder um determinado valor)
- Não uniforme
  - chaves mais solicitadas mais perto da raiz

POR FREQUÊNCIA X POR ALTURA

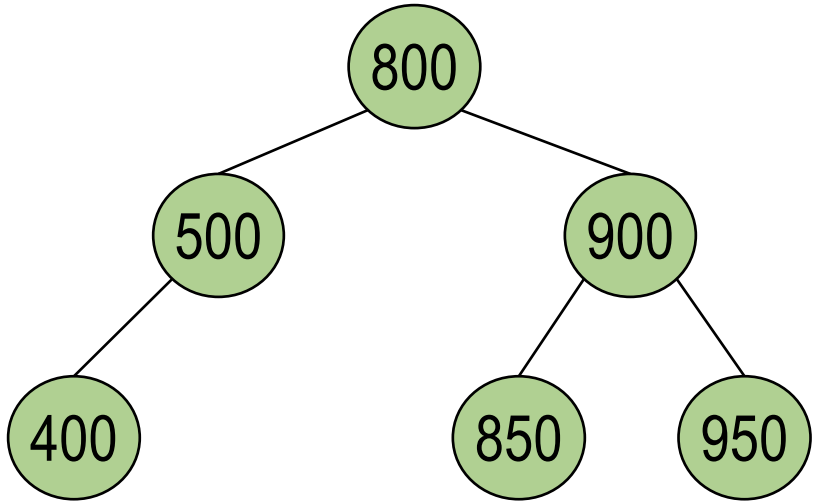
Splay

Nós mais acessados ficam perto da raiz



AVL, Rubro-Negras

Diferença das alturas das subárvores não excedem um determinado valor

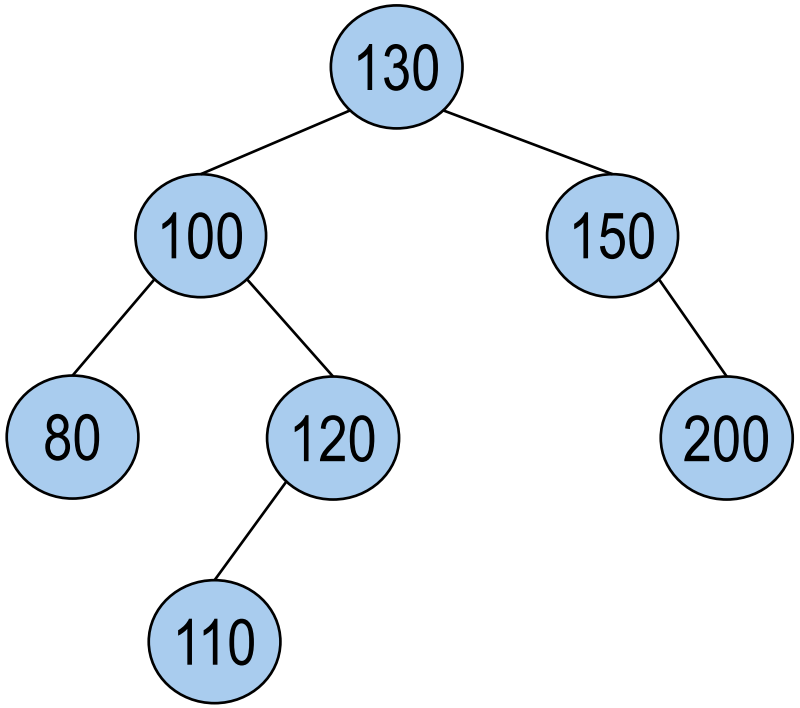


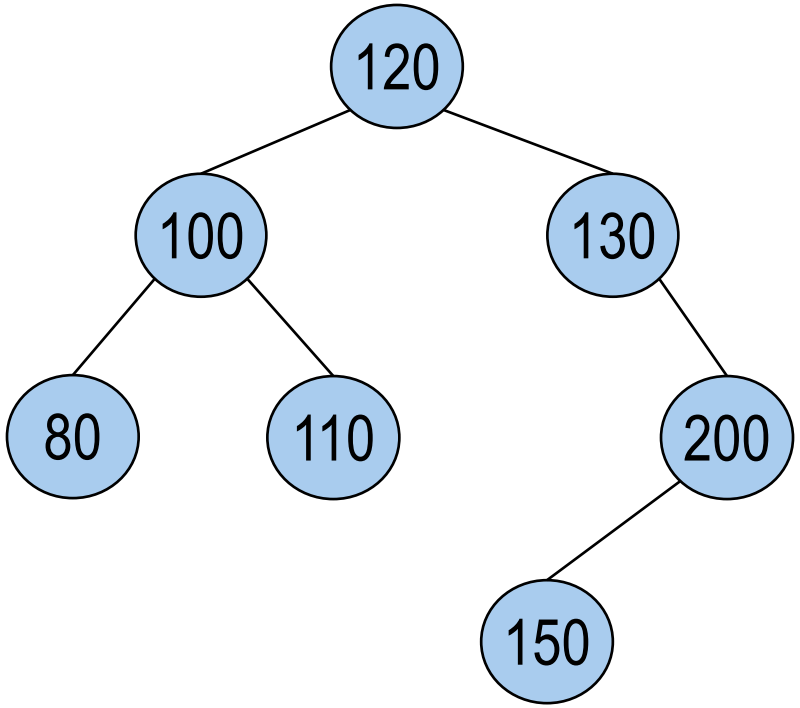
ÁRVORES AVL
ADELSON-VELSKII E LANDIS (1962)

Uma árvore AVL é uma **árvore binária de busca** (ABB) quando, para qualquer um de seus nós, **a diferença** entre as **alturas de suas subárvore direita e esquerda** é no **máximo 1.**

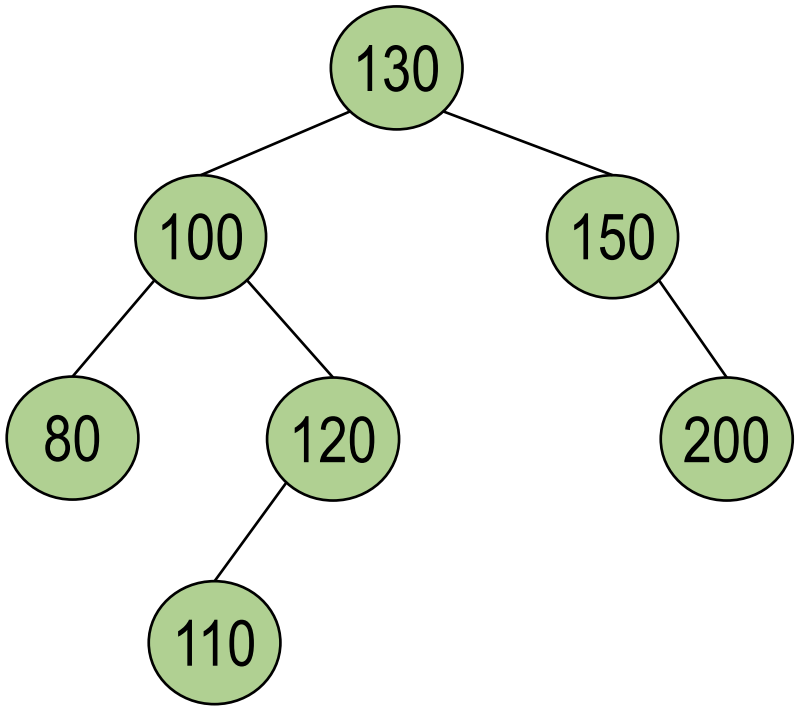
EXERCÍCIO

Verifique quais das ABB são AVL

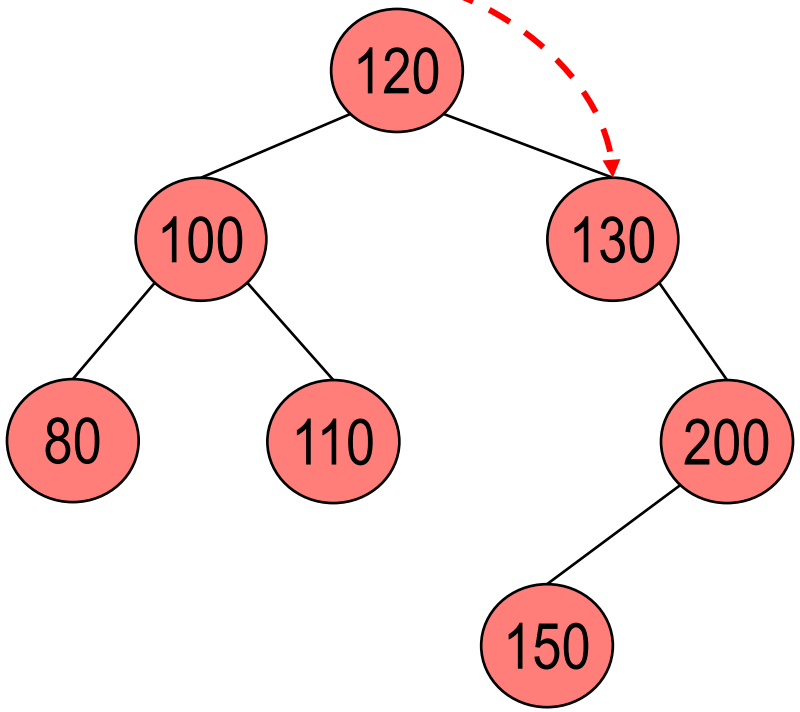




RESPOSTA

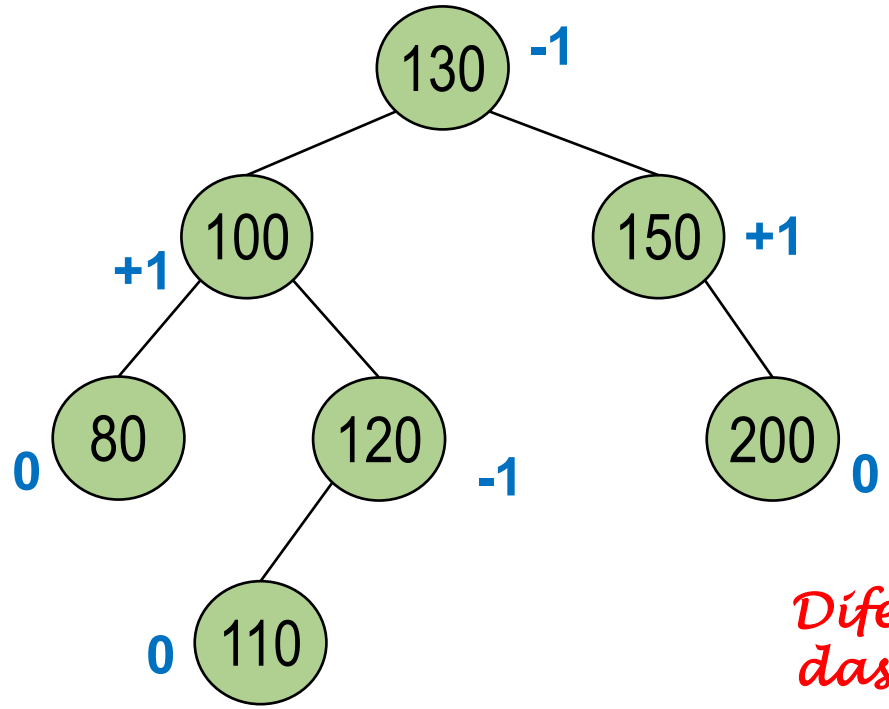


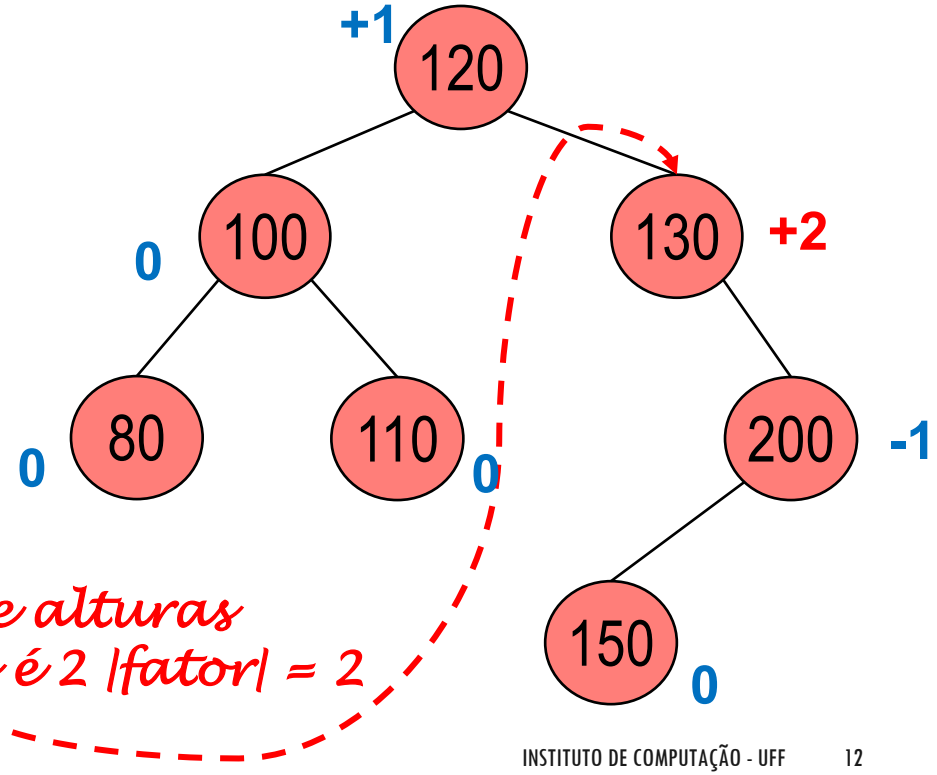
Diferença entre alturas das subárvores é 2



FATOR DE BALANCEAMENTO (FB)

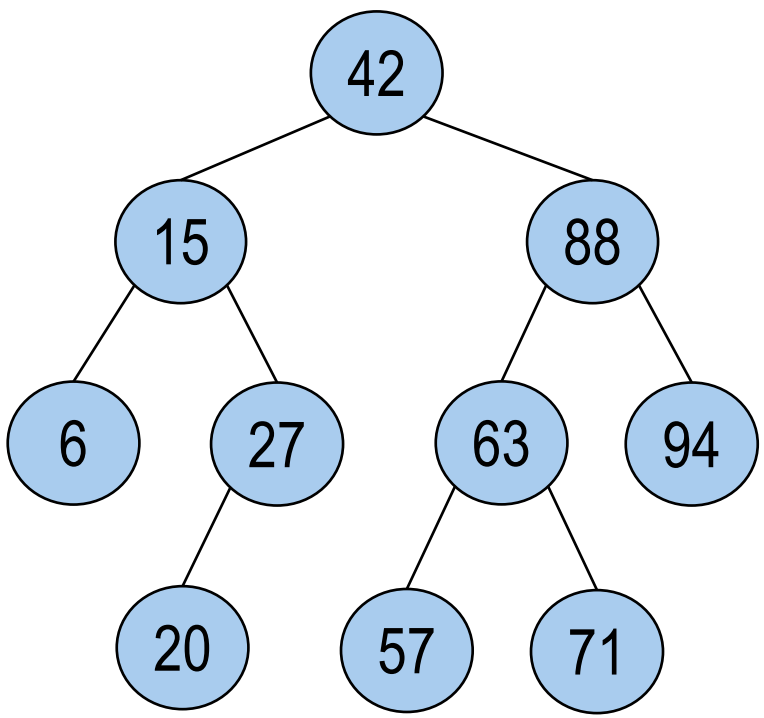
Fator de Balanceamento (diferença entre altura da subárvore direita e esquerda)
Em árvores AVL, FB precisa ser -1, 0 ou +1 para que árvore seja AVL.

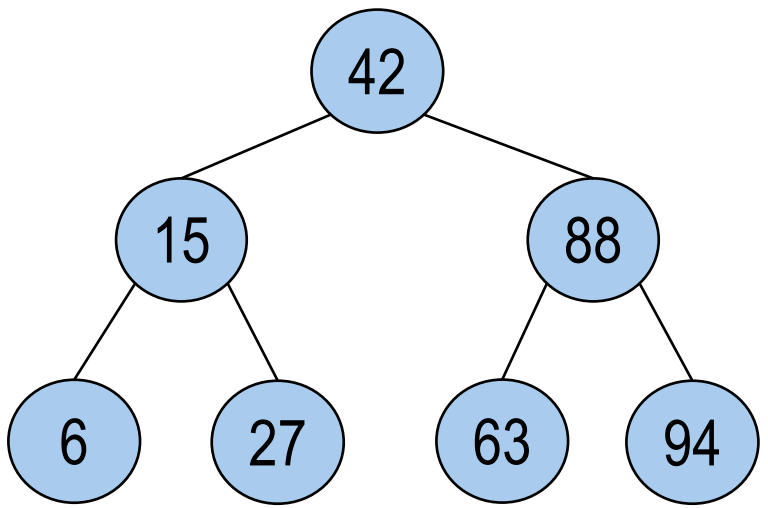


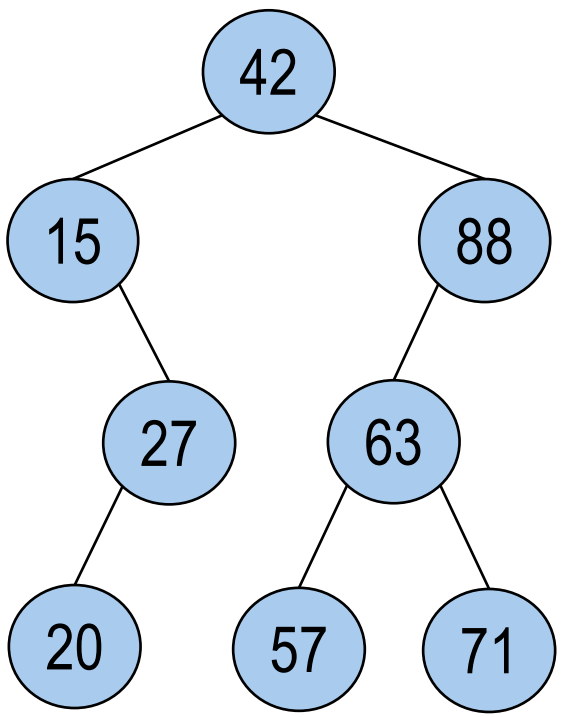


EXERCÍCIO

Verifique quais das ABB são AVL:

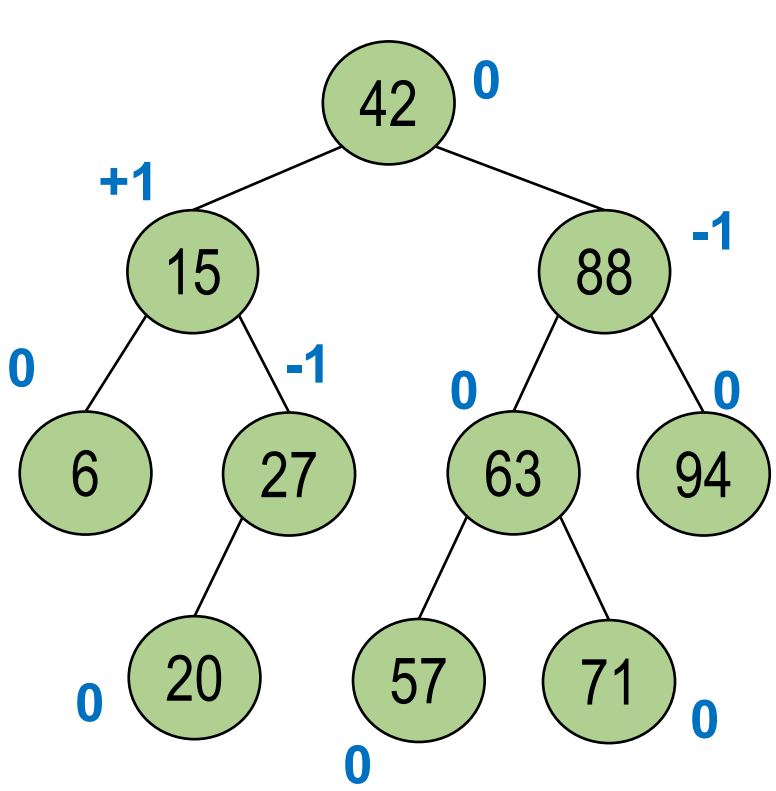


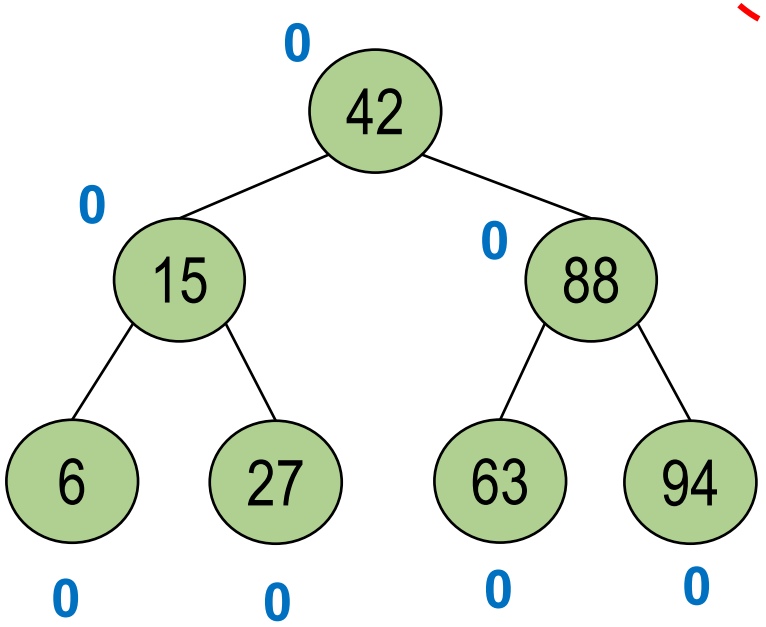


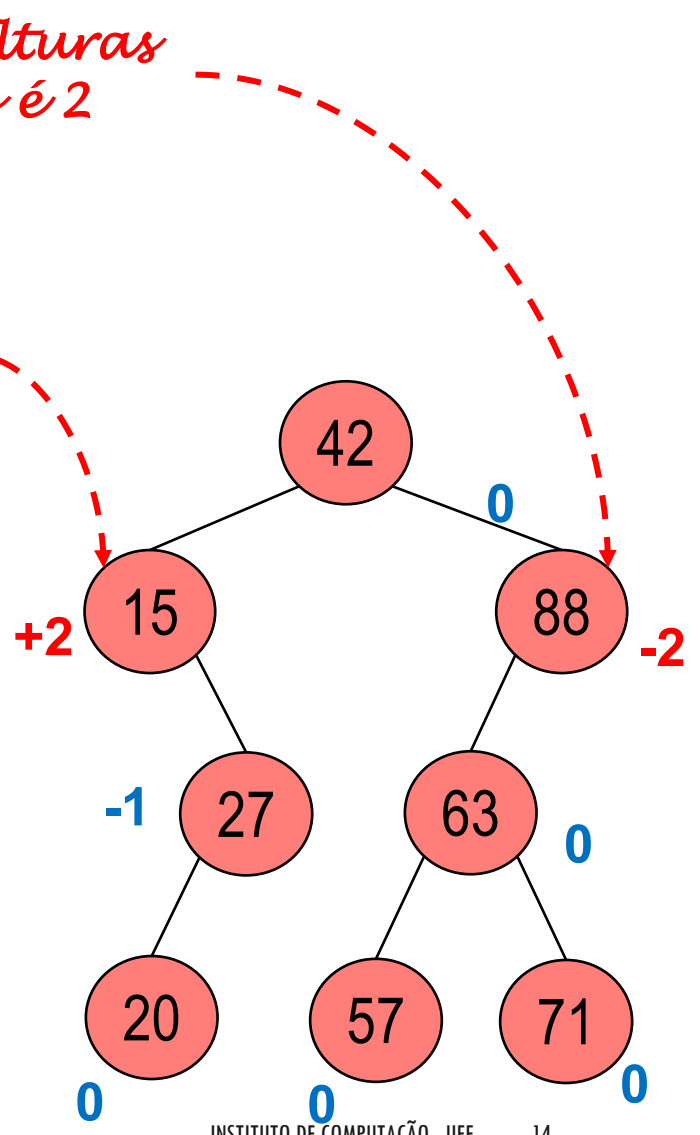


RESPOSTA

Verifique quais das ABB são AVL:

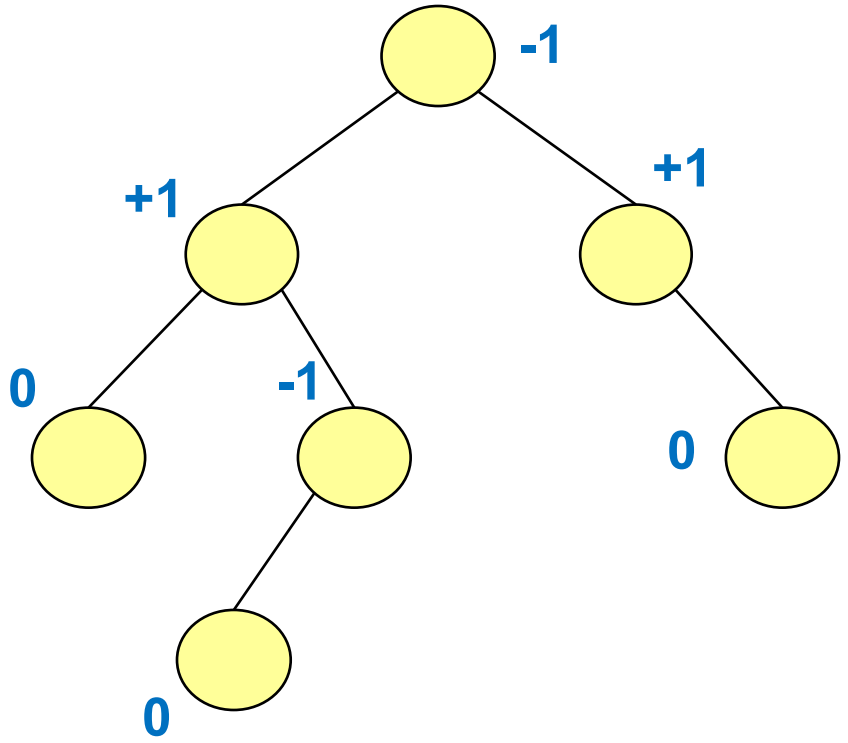




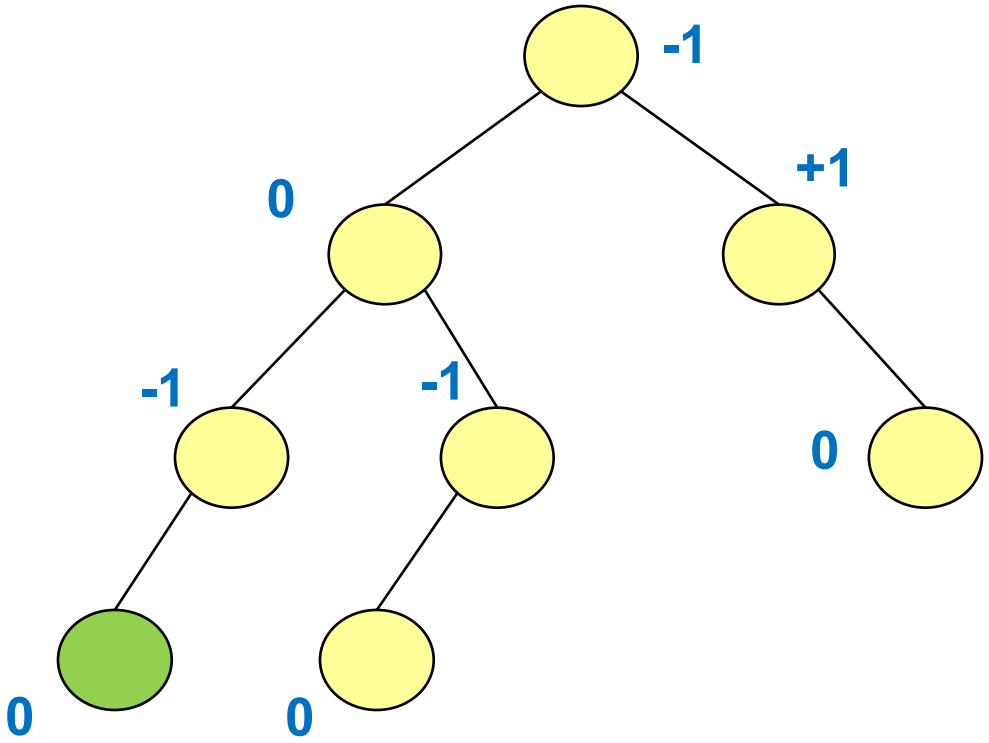


OPERAÇÕES

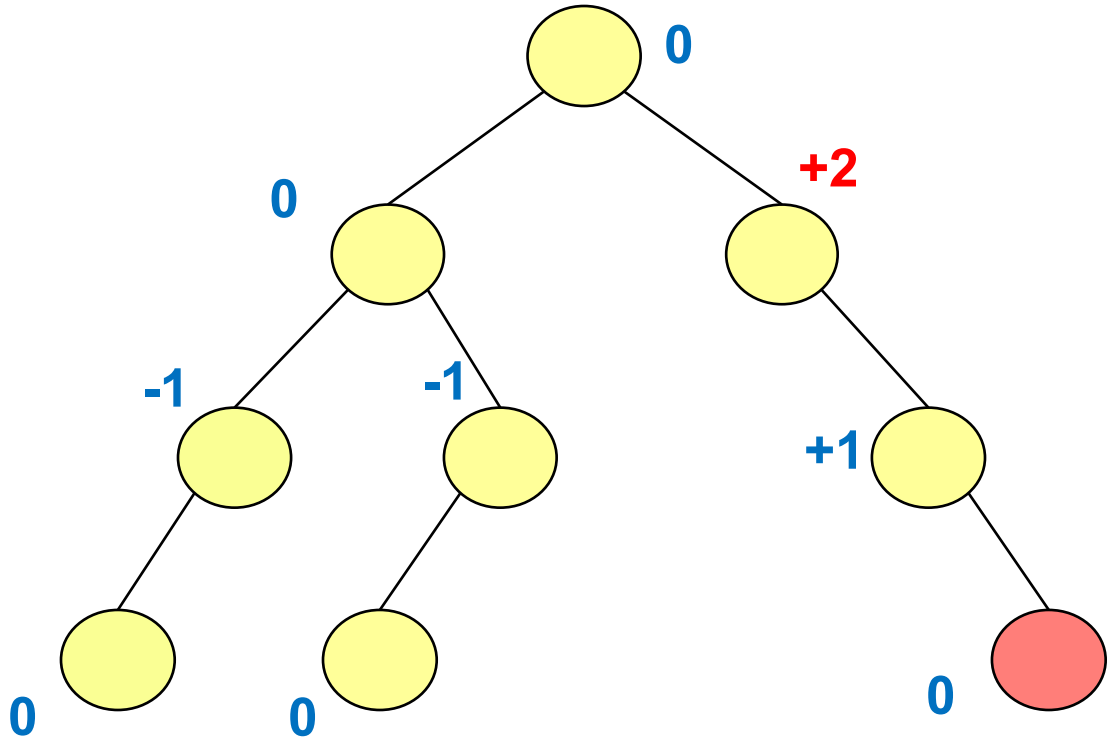
Inserção e Exclusão devem preservar as propriedades da AVL



INSERÇÃO

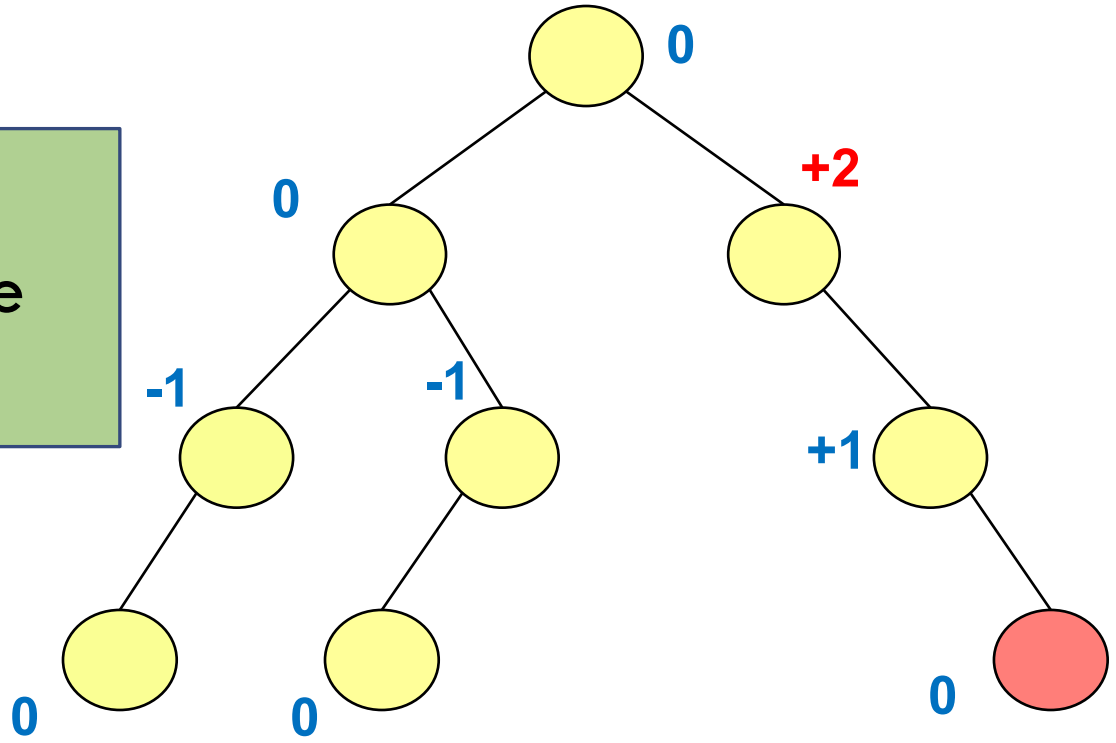


INSERÇÃO



INSERÇÃO

Reestruturar Árvore



OPERAÇÕES

Quando uma inserção ou exclusão faz com que a árvore perca as propriedades de árvore AVL, deve-se realizar uma operação de reestruturação chamada **Rotação**

Rotação preserva a ordem das chaves, de modo que a árvore resultante é uma árvore binária de busca válida e é uma árvore AVL válida

BALANCEAMENTO DE ÁRVORES AVL POR ROTAÇÃO

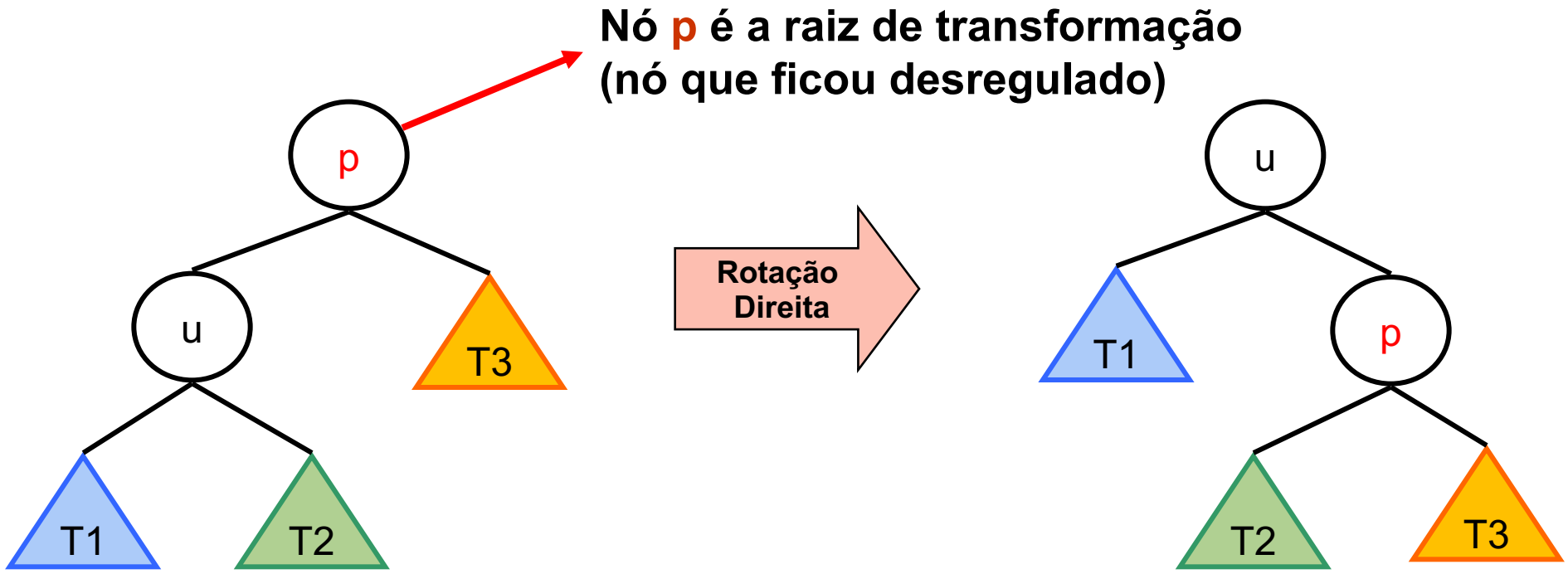
Rotação Simples

- Direita
- Esquerda

Rotação Dupla

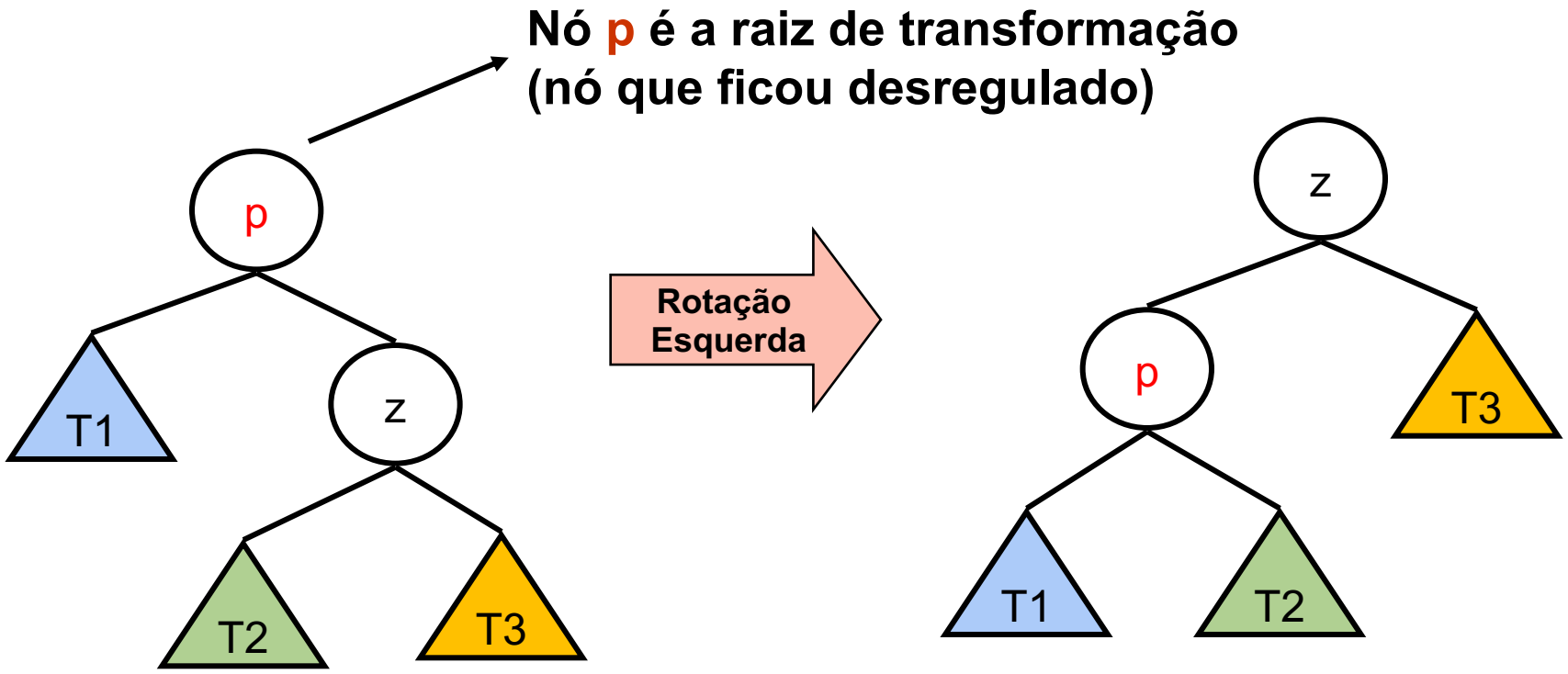
- Direita
- Esquerda

ROTAÇÃO DIREITA



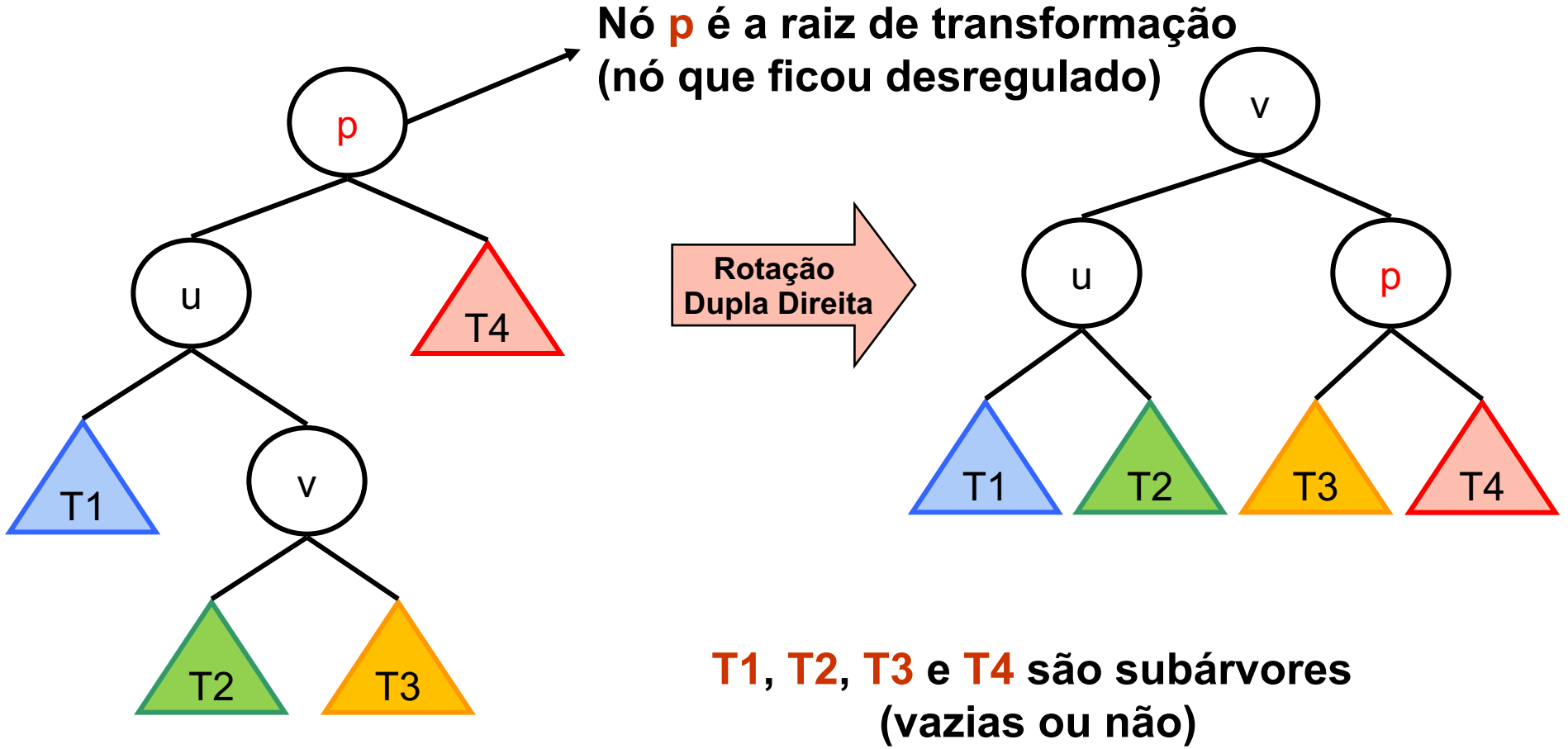
T1, T2, T3 e T4 são subárvores (vazias ou não)

ROTAÇÃO ESQUERDA



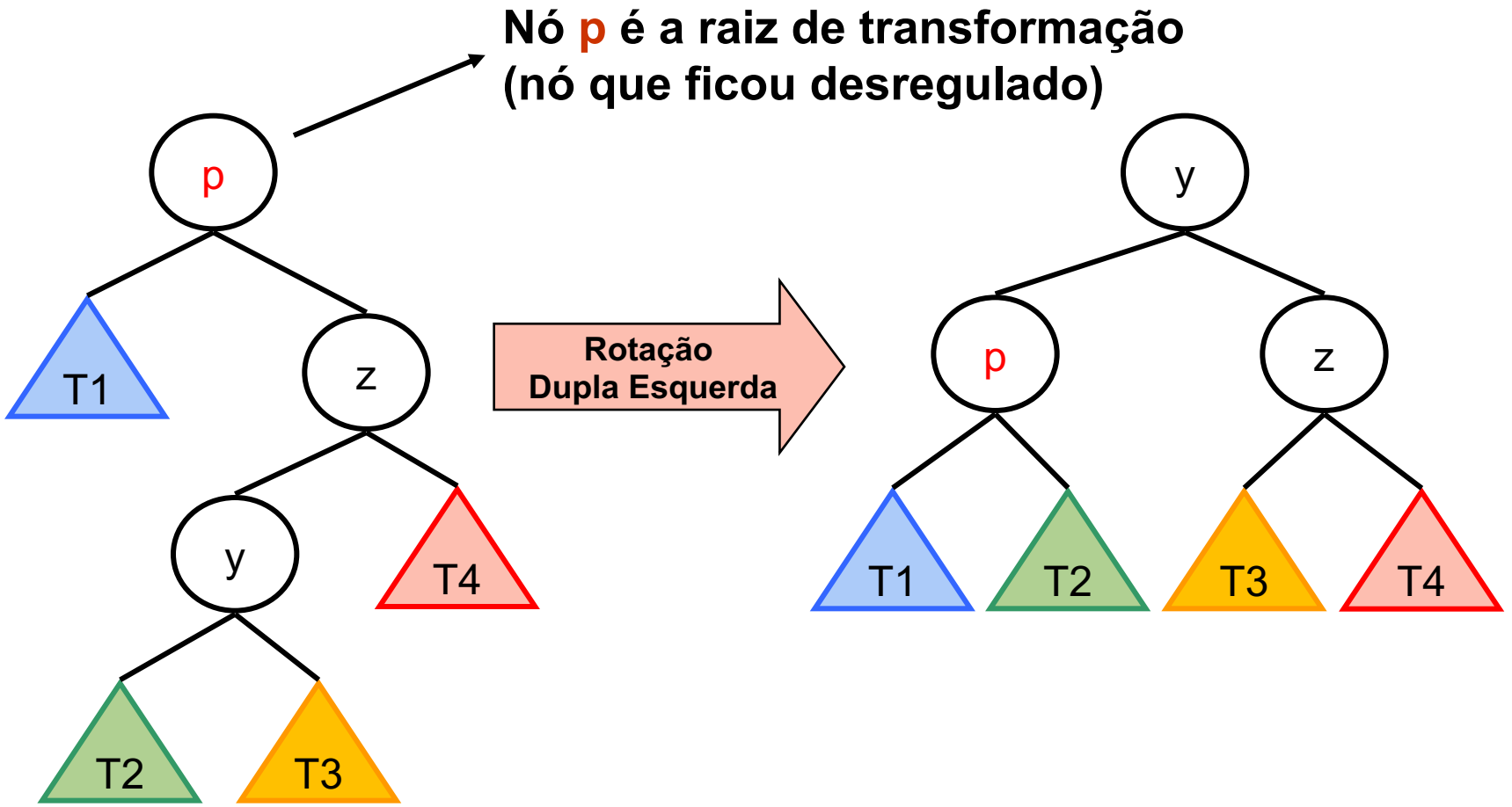
T1, T2, T3 e T4 são subárvores (vazias ou não)

ROTAÇÃO DUPLA DIREITA



T1, T2, T3 e T4 são subárvores (vazias ou não)

ROTAÇÃO DUPLA ESQUERDA



QUANDO APLICAR?

Fator de Balanceamento FB = h(subarv-direita) – h(subarv-esquerda)

Se FB positivo:

- rotações à esquerda

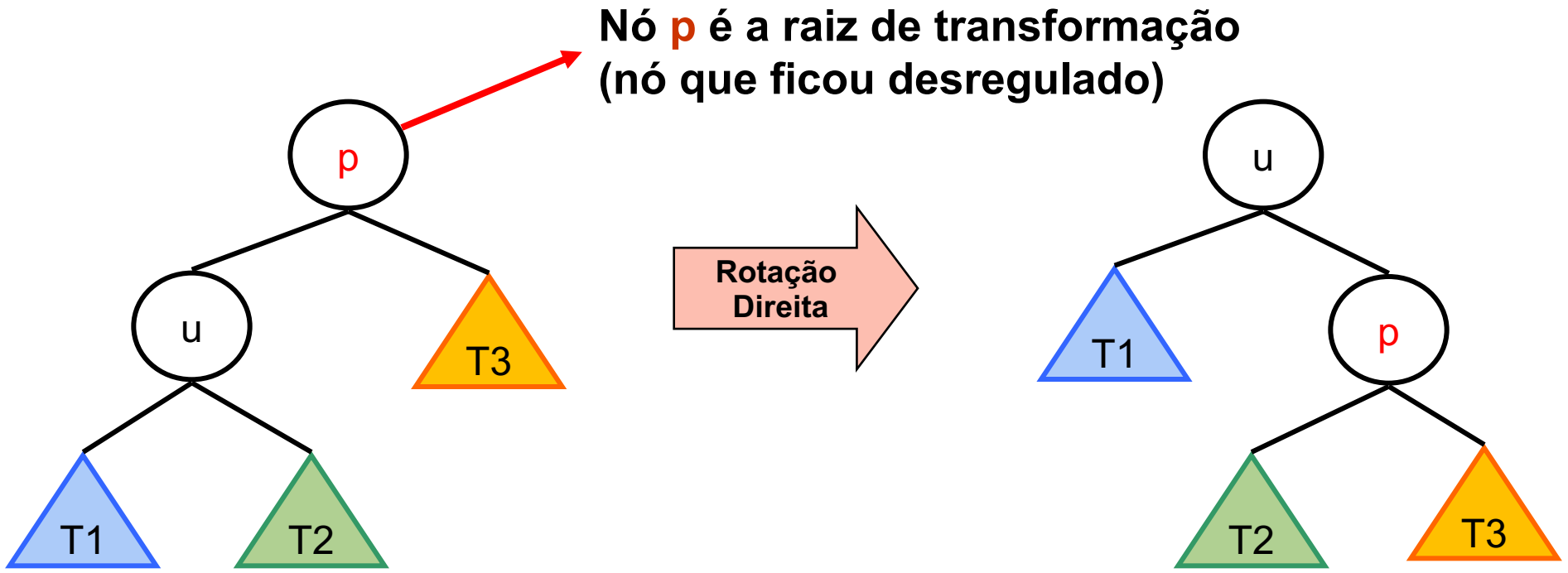
Se FB negativo

- rotações à direita

ROTAÇÃO SIMPLES DIREITA

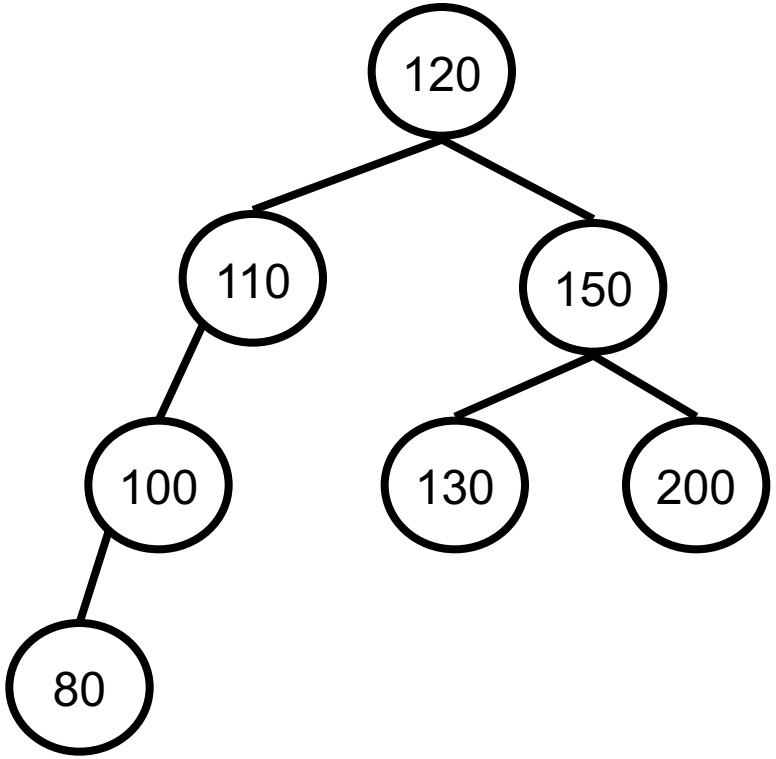
Aplicar toda vez que uma subárvore ficar com um **FB negativo** e sua **subárvore esquerda** também tem com um **FB negativo**

ROTAÇÃO DIREITA

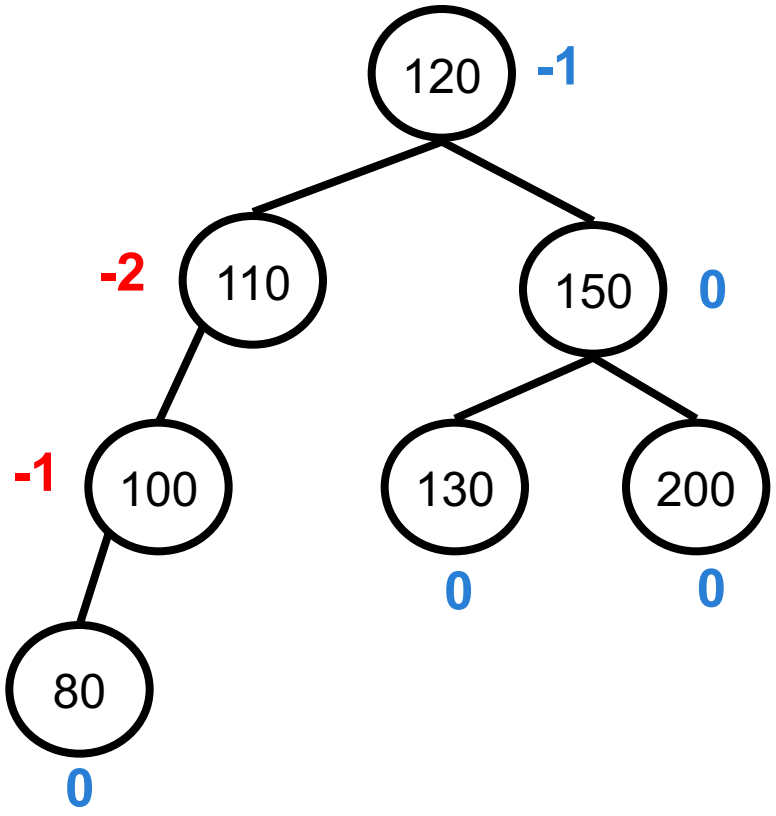


T1, T2, T3 e T4 são subárvores (vazias ou não)

EXEMPLO 1: ROTAÇÃO DIREITA



EXEMPLO 1: ROTAÇÃO DIREITA

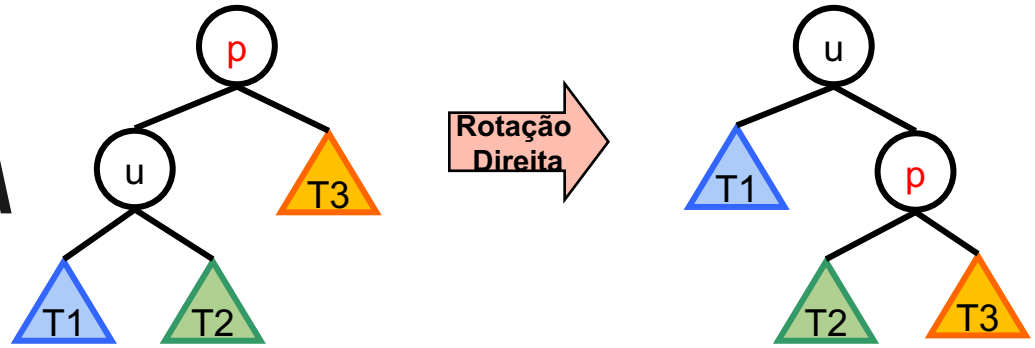


Nó **desregulado** com fator negativo e subárvore esquerda com fator também negativo

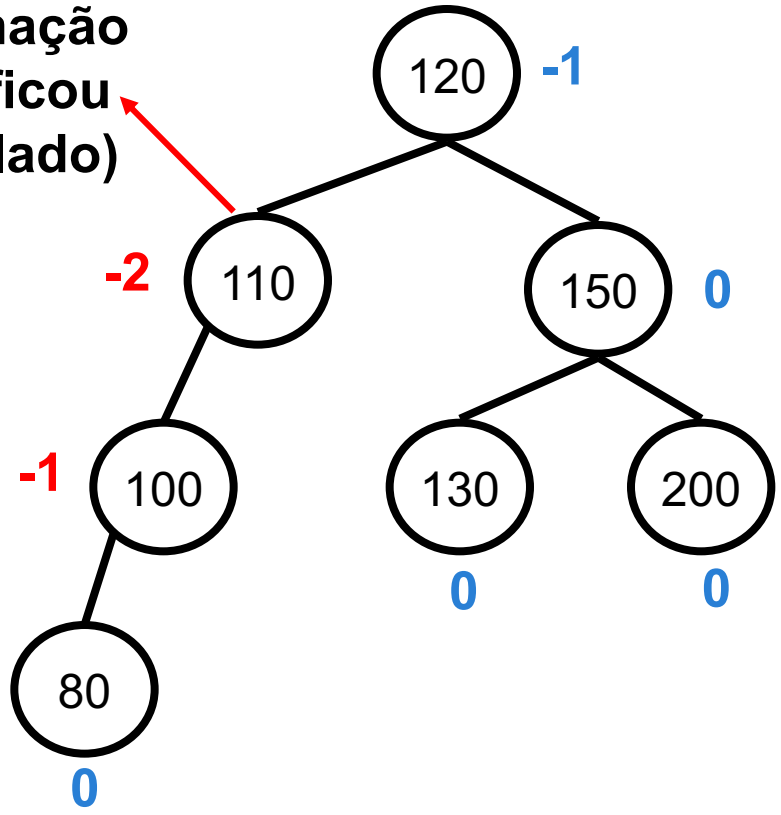
=

ROTAÇÃO DIREITA

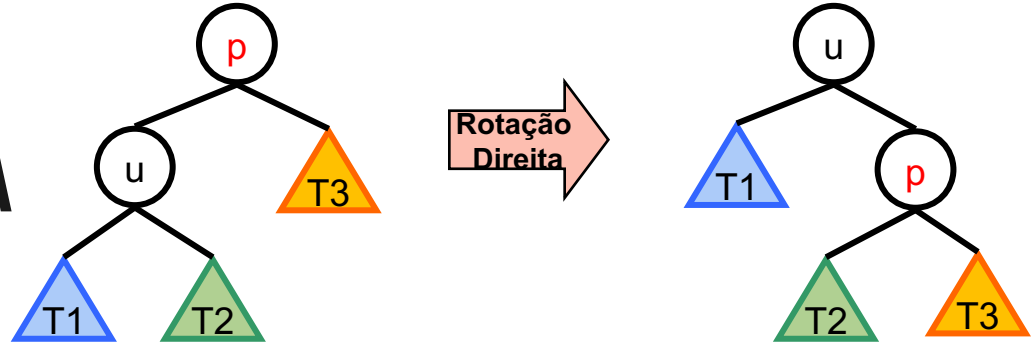
EXEMPLO 1: ROTAÇÃO DIREITA

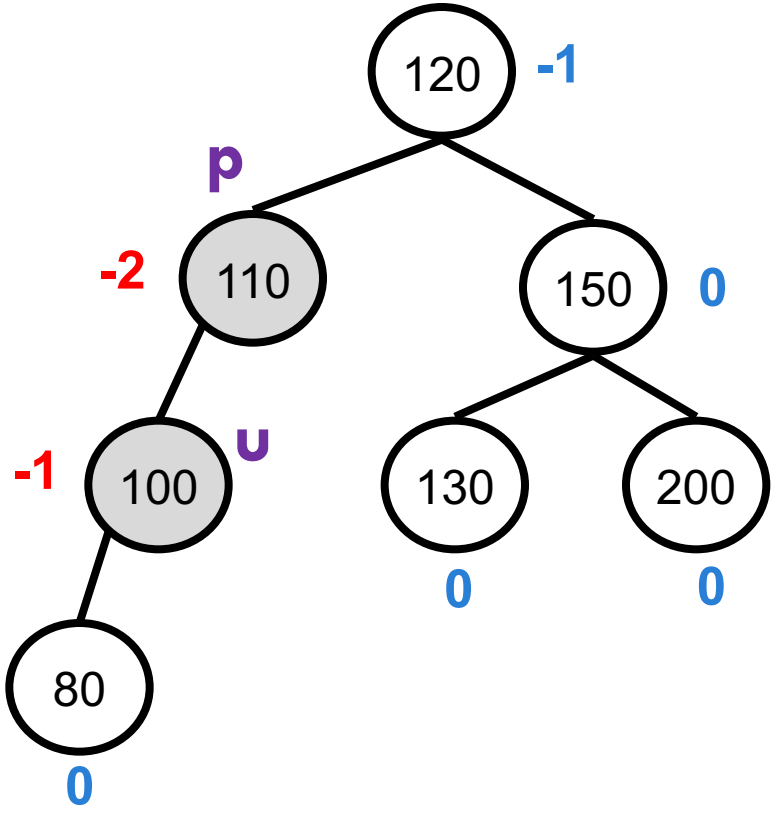


Nó p é a raiz de transformação (nó que ficou desregulado)

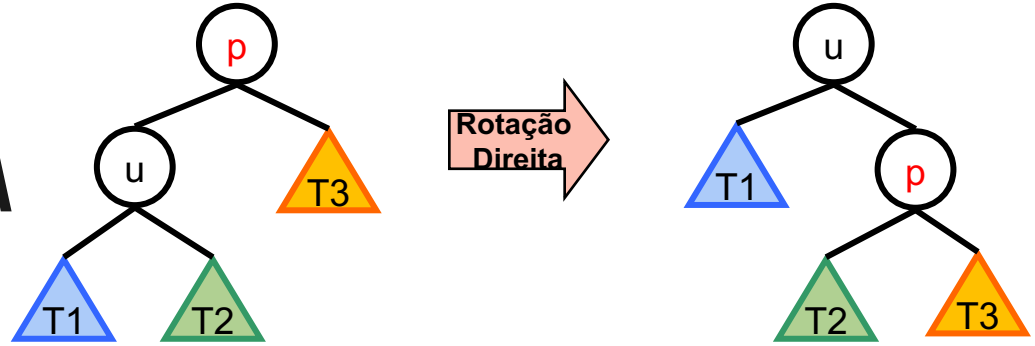


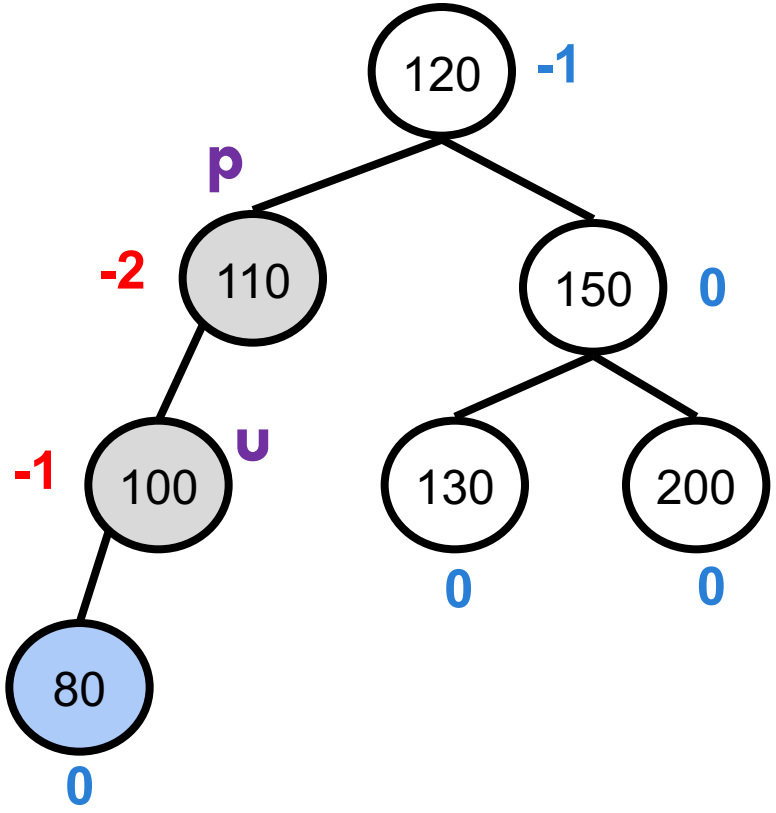
EXEMPLO 1: ROTAÇÃO DIREITA

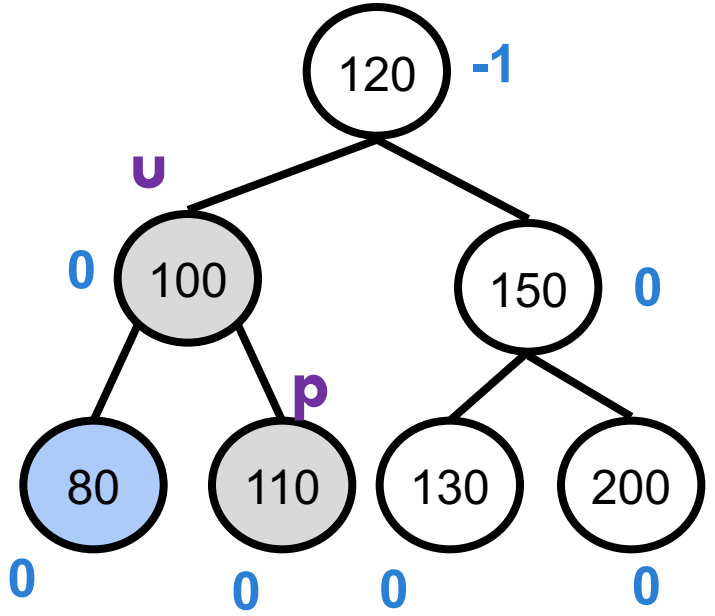




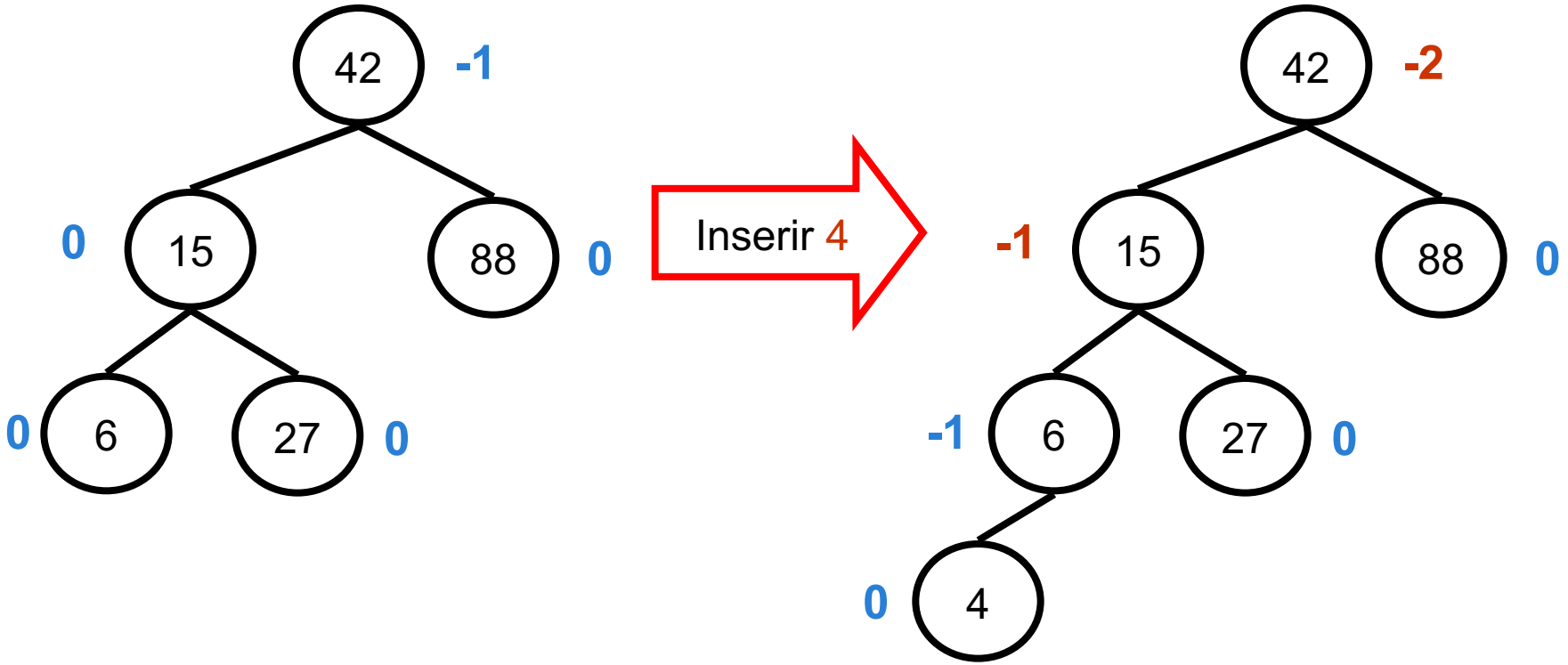
EXEMPLO 1: ROTAÇÃO DIREITA



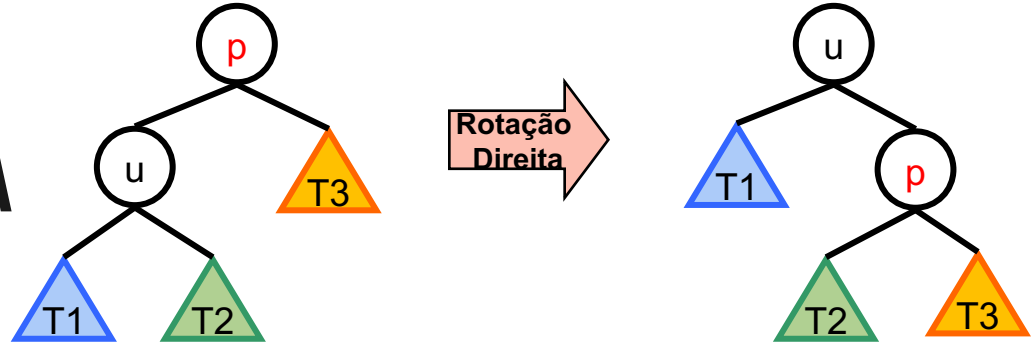


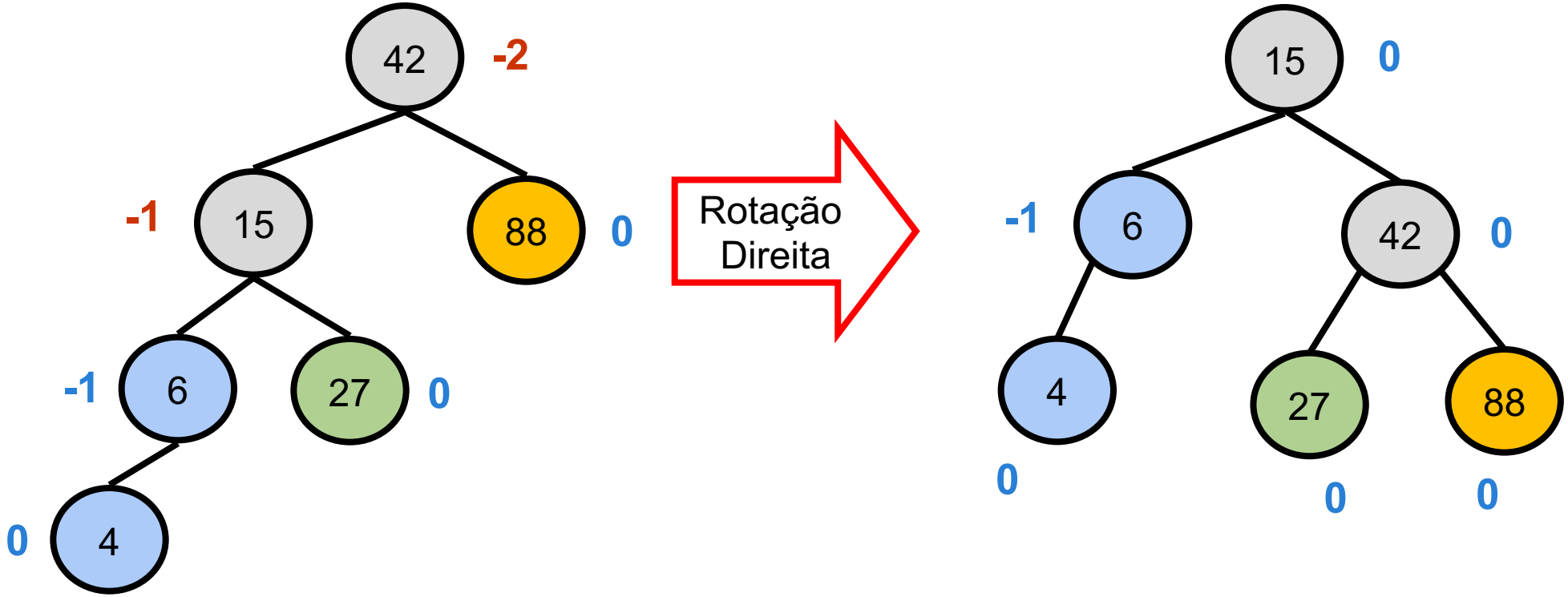


EXEMPLO 2: INSERIR 4

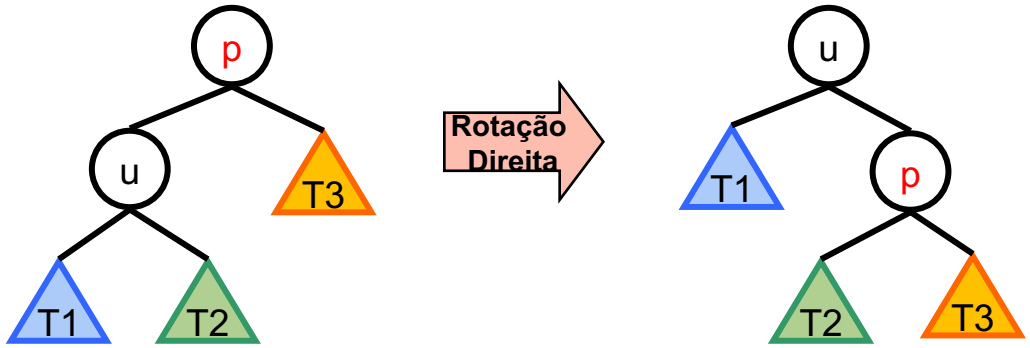


EXEMPLO 2: ROTAÇÃO DIREITA





IMPLEMENTAÇÃO ROTAÇÃO DIREITA



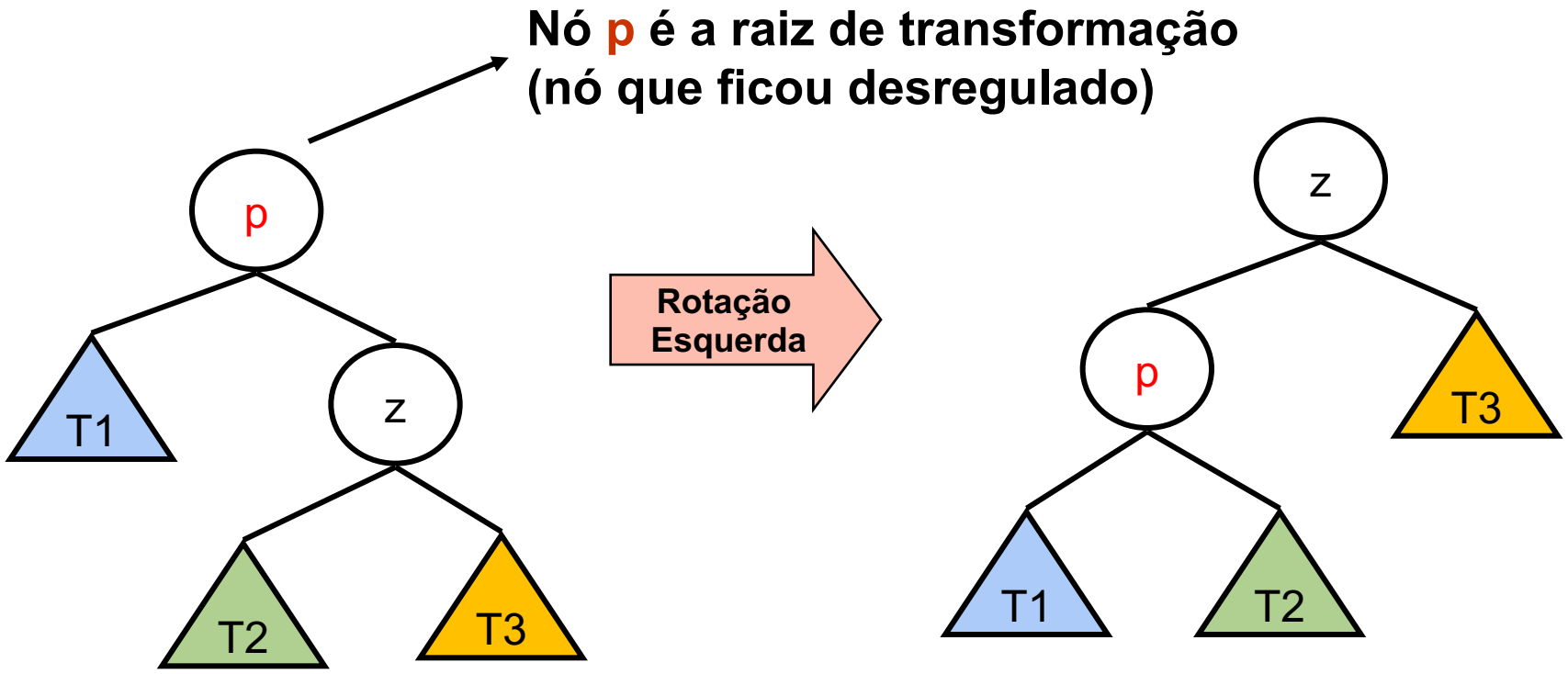

```
/* representação dos nós de Árvore ALV */
typedef struct pNoA {
   TInfo info;
   struct pNoA* esq;
   struct pNoA* dir;
} pNoA;

pNodoA* rotacao_direita(pNoA* pt){
   pNoA *ptu;
   ptu = pt->esq;
   pt->esq = ptu->dir;
   ptu->dir = pt;
   pt = ptu;
   return pt;
}
```

ROTAÇÃO SIMPLES ESQUERDA

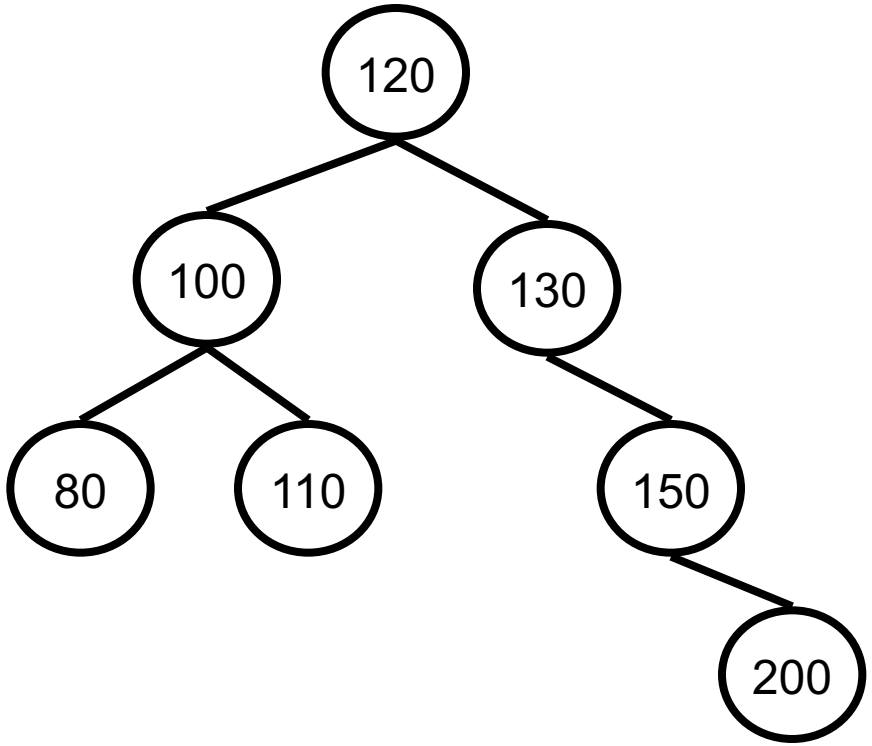
Aplicar toda vez que uma subárvore ficar com um **FB positivo** e sua **subárvore direita** também tem com um **FB positivo**

ROTAÇÃO ESQUERDA

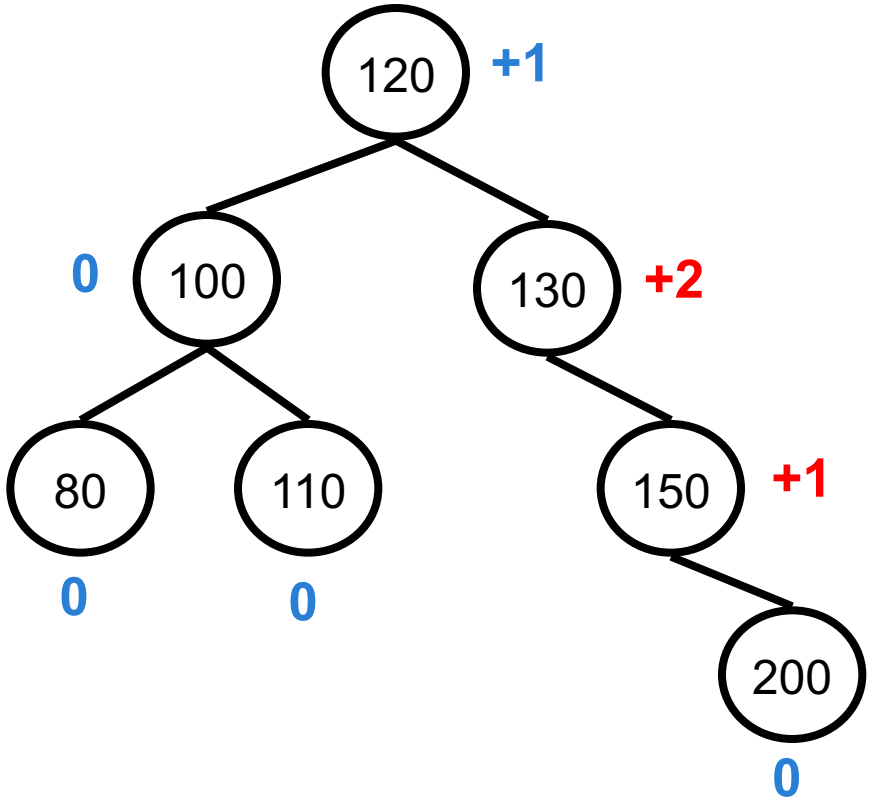


T1, T2, T3 e T4 são subárvores (vazias ou não)

EXEMPLO 1: ROTAÇÃO ESQUERDA



EXEMPLO 1: ROTAÇÃO ESQUERDA

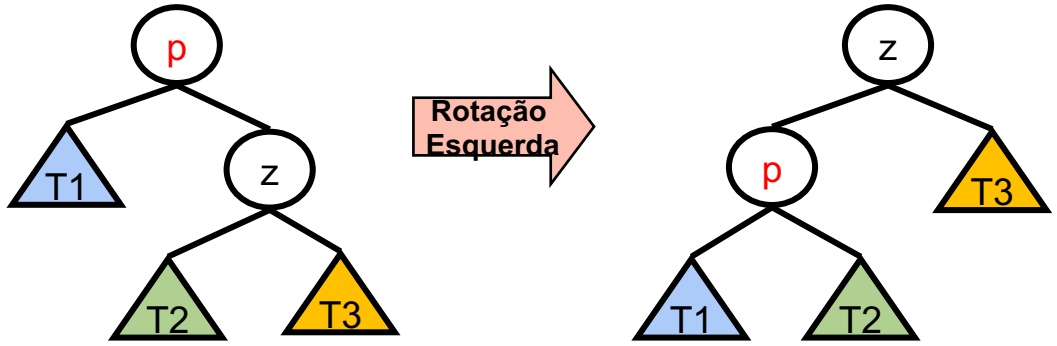


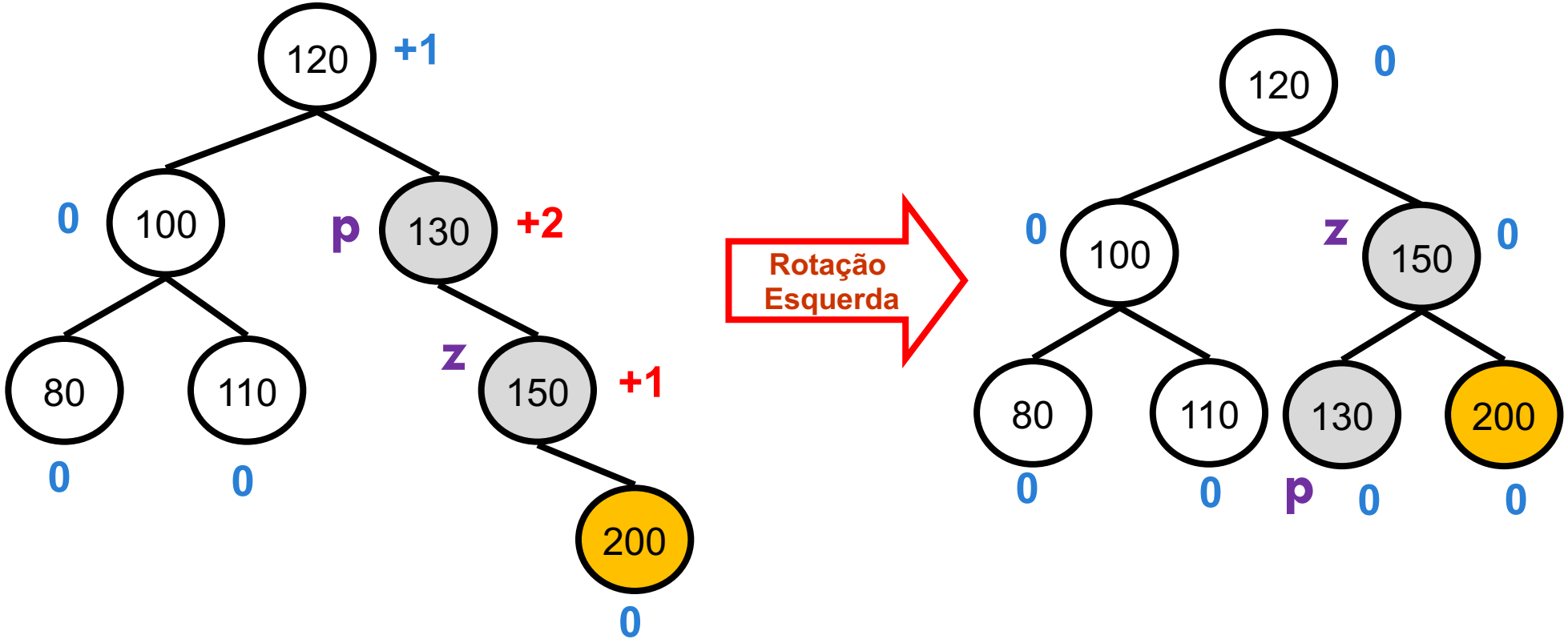
Nó **desregulado** com fator positivo e subárvore direita com fator também positivo

=

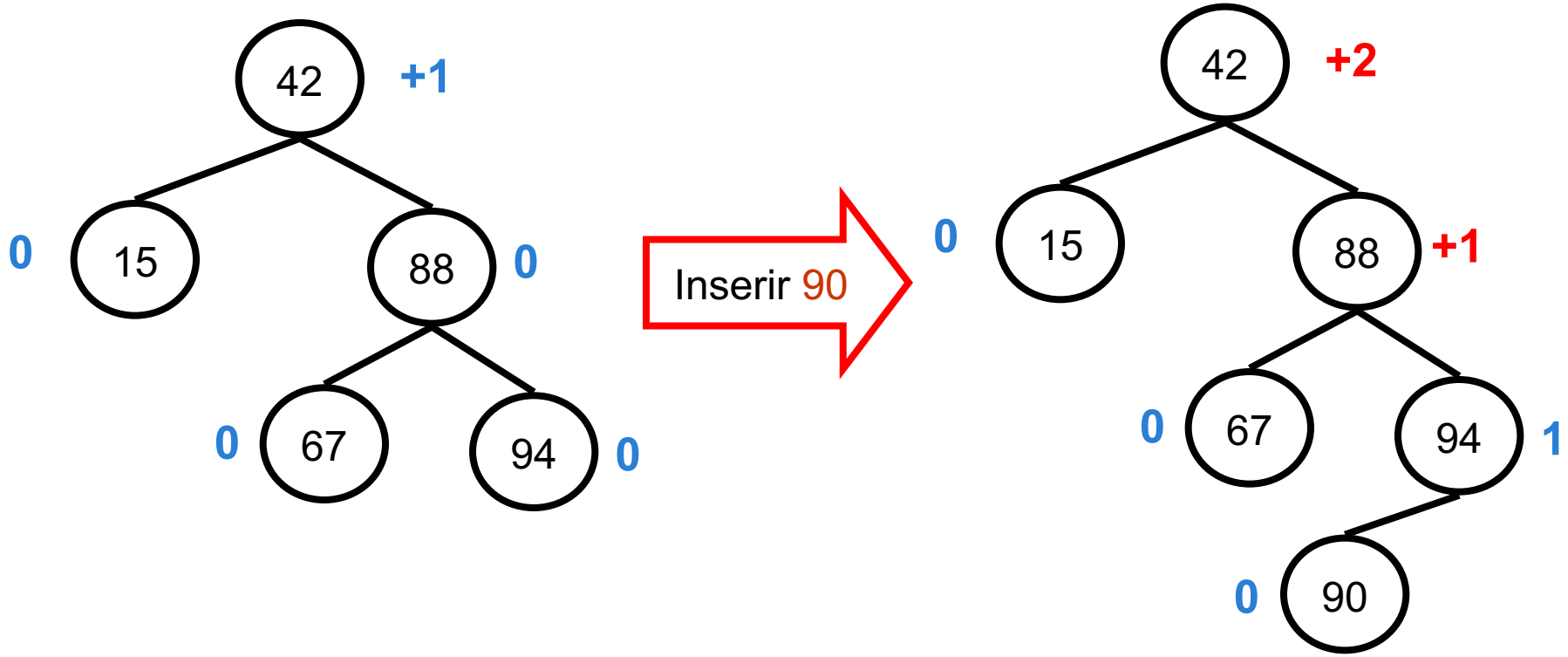
ROTAÇÃO ESQUERDA

EXEMPLO 1: ROTAÇÃO ESQUERDA

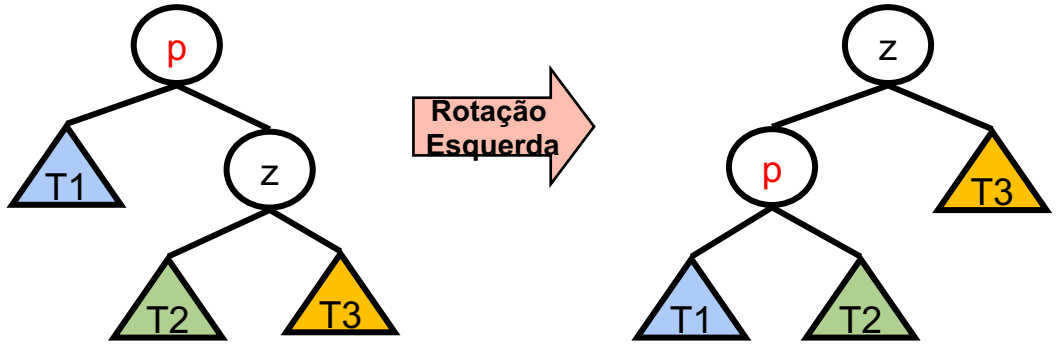


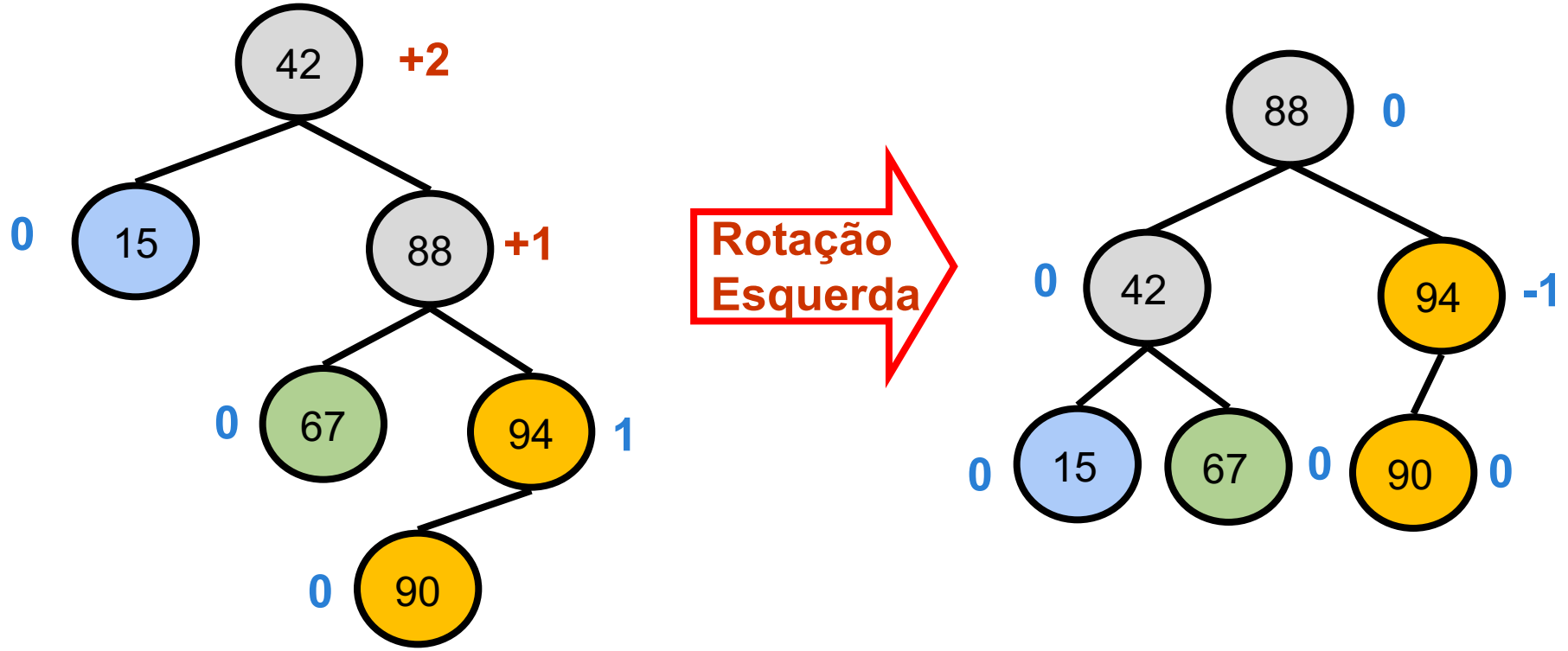


EXEMPLO 2: INSERIR 90

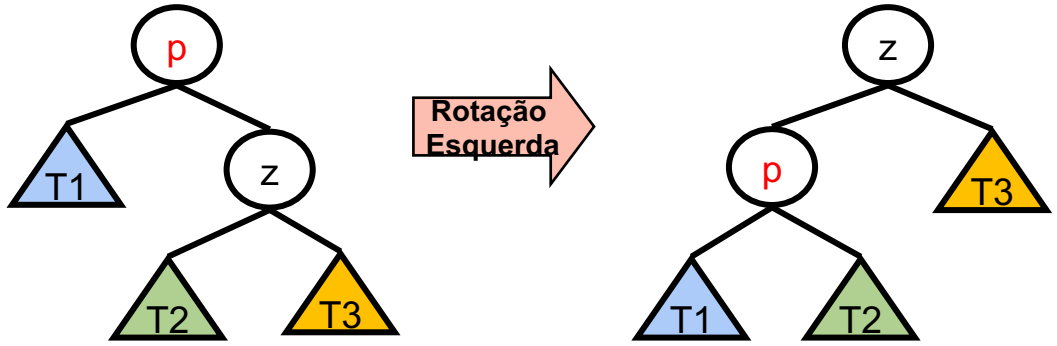


EXEMPLO 2: ROTAÇÃO ESQUERDA





IMPLEMENTAÇÃO ROTAÇÃO ESQUERDA



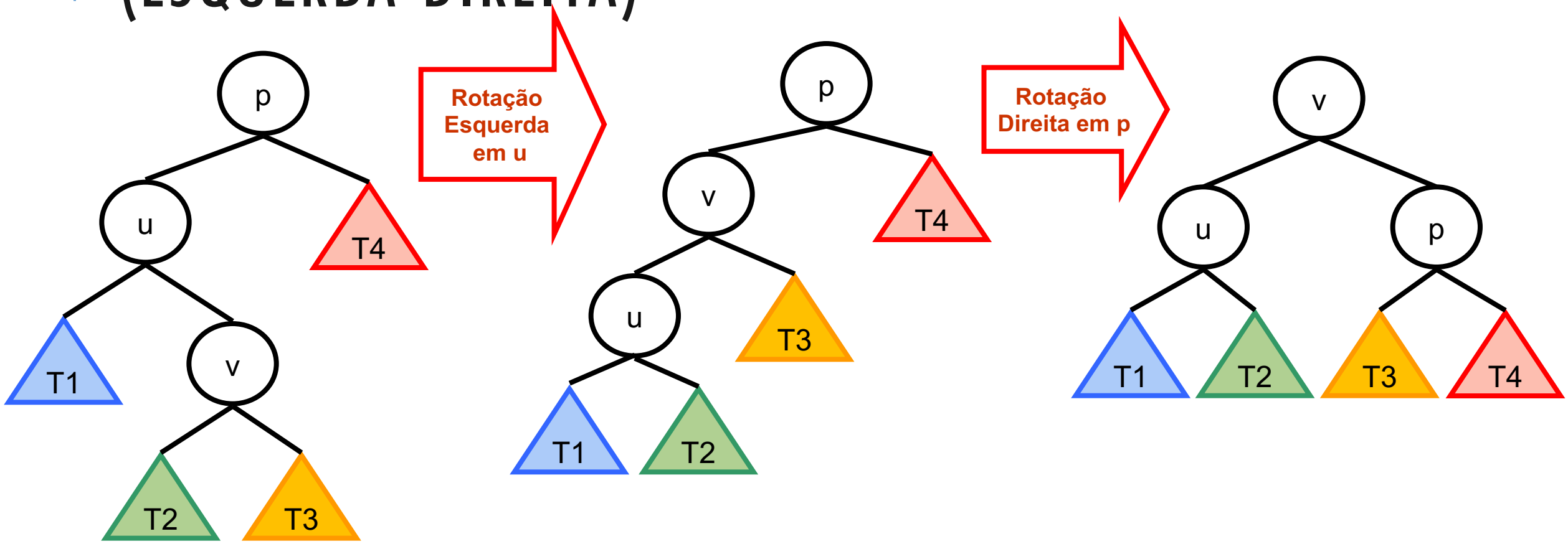

```
/* representação dos nós de Árvore ALV */
typedef struct pNoA {
   TInfo info;
   struct pNoA* esq;
   struct pNoA* dir;
} pNoA;

pNodoA* rotacao_esquerda(pNodoA *pt){
   pNodoA *ptu;
   ptu = pt->dir;
   pt->dir = ptu->esq;
   ptu->esq = pt;
   pt = ptu;
   return pt;
}
```

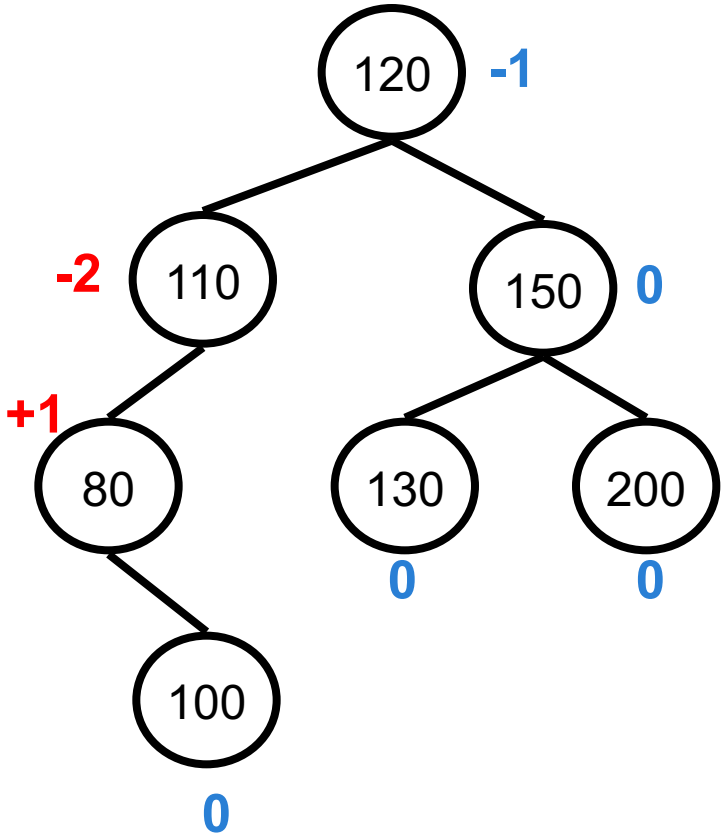
ROTAÇÃO DUPLA DIREITA (ESQUERDA-DIREITA)

Aplicar toda vez que uma subárvore ficar com um **FB negativo** e sua **subárvore esquerda** tem com um **FB positivo**

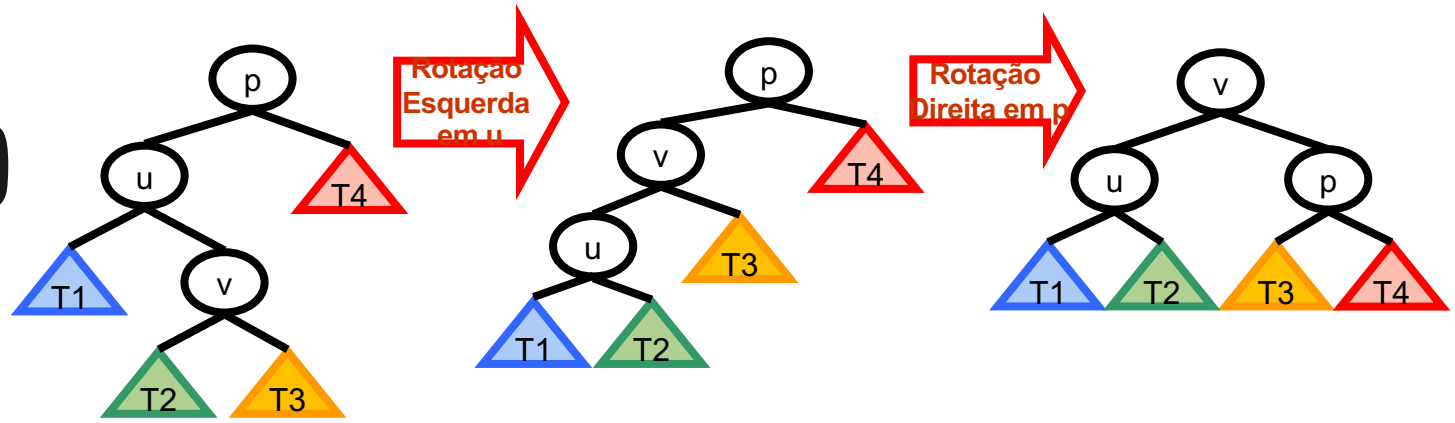
ROTAÇÃO DUPLA DIREITA (ESQUERDA-DIREITA)

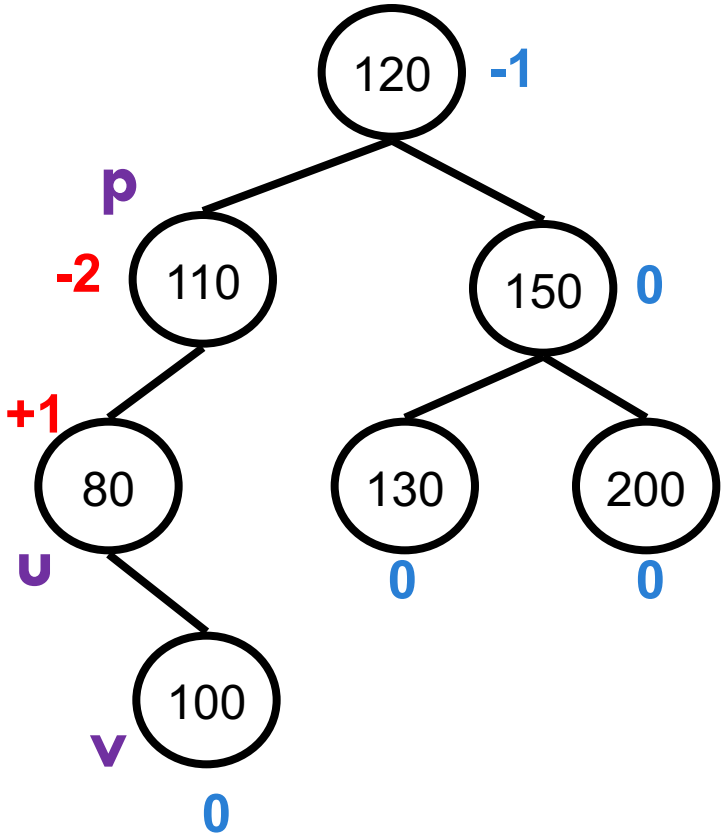


EXEMPLO 1: ROTAÇÃO DUPLA DIREITA

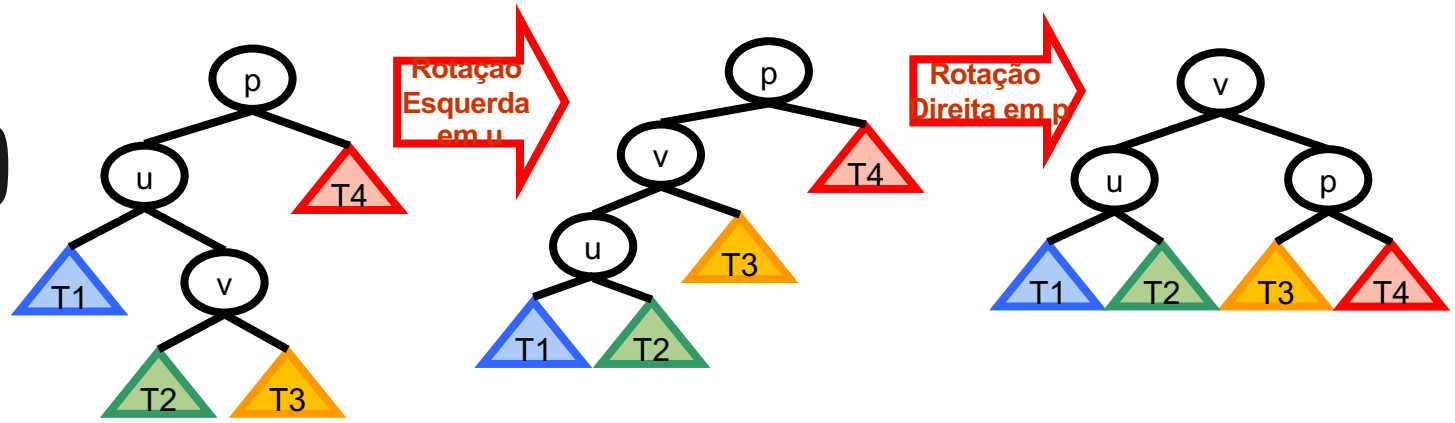


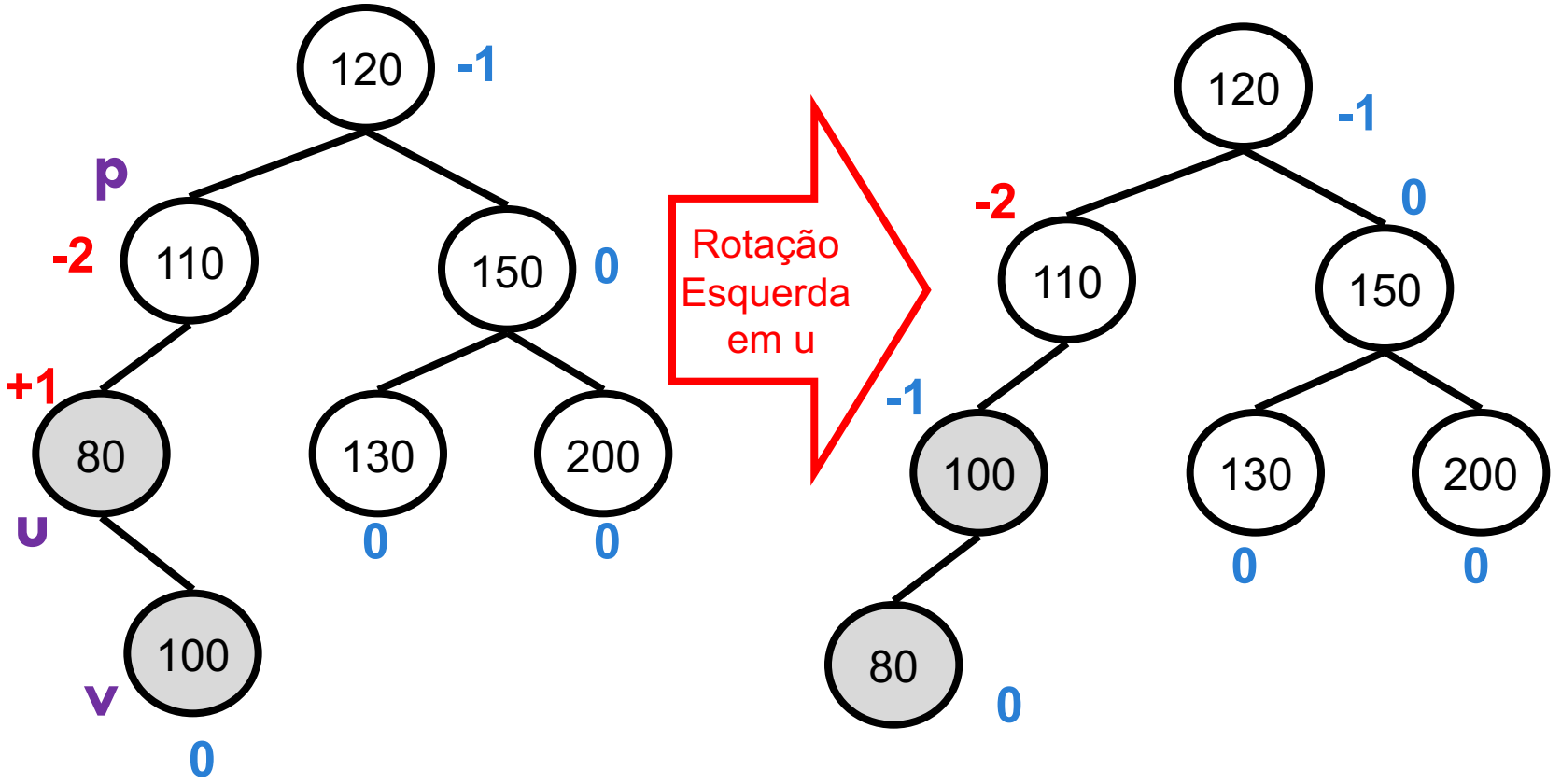
EXEMPLO 1: ROTAÇÃO DUPLA DIREITA



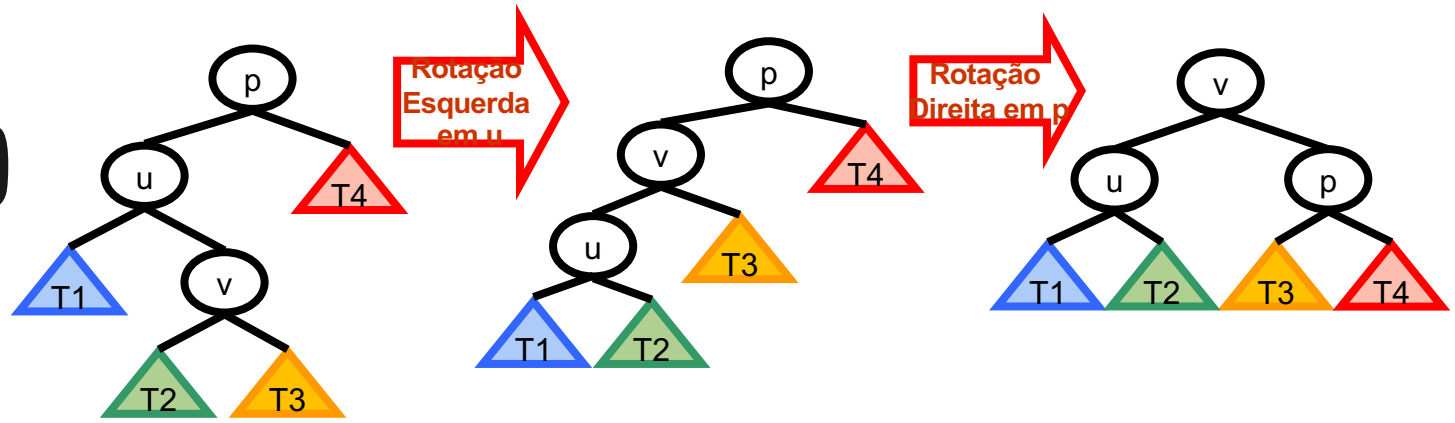


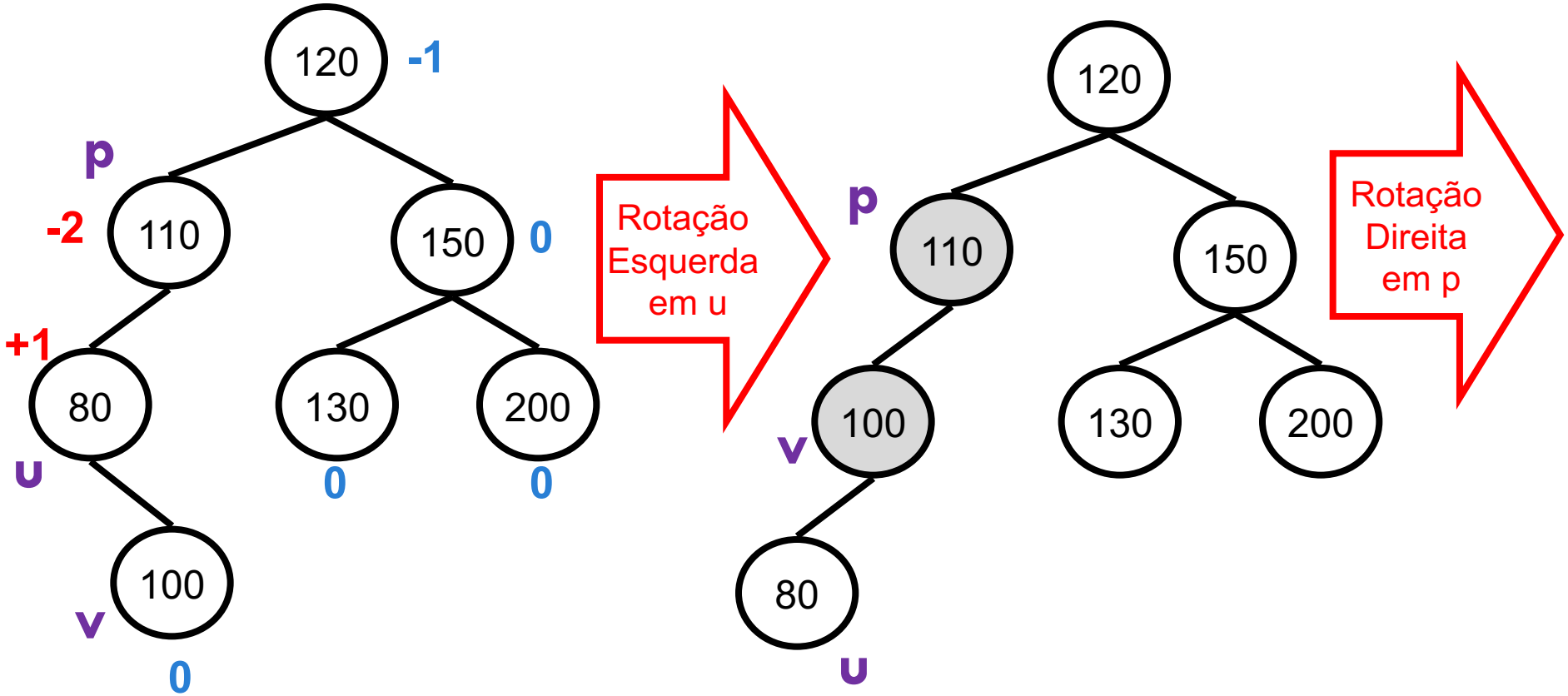
EXEMPLO 1: ROTAÇÃO DUPLA DIREITA



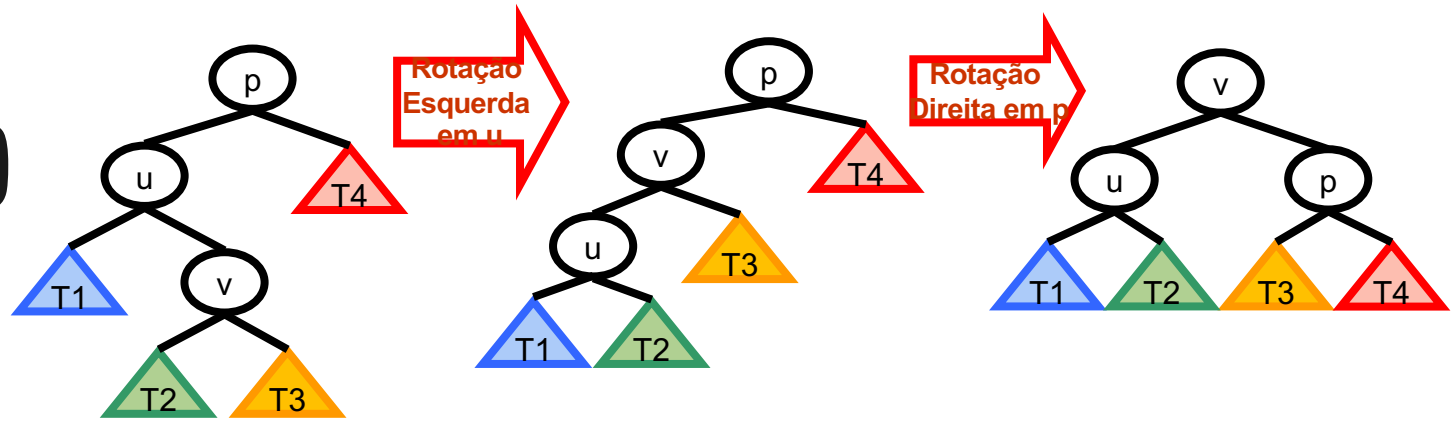


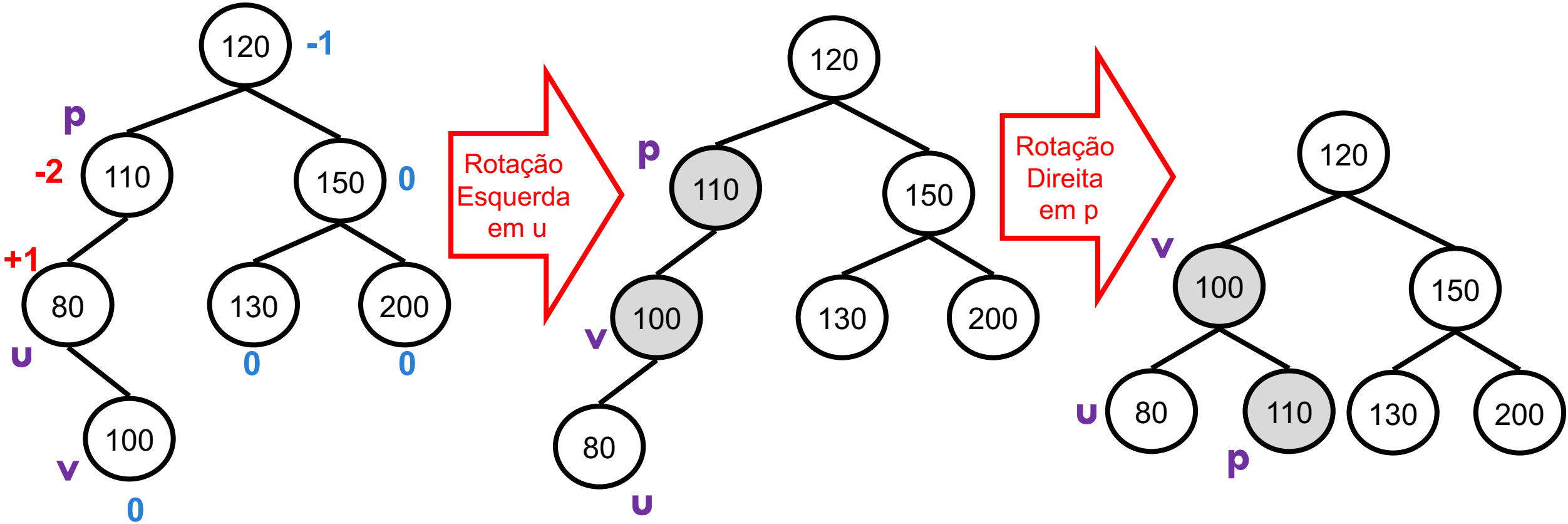
EXEMPLO 1: ROTAÇÃO DUPLA DIREITA



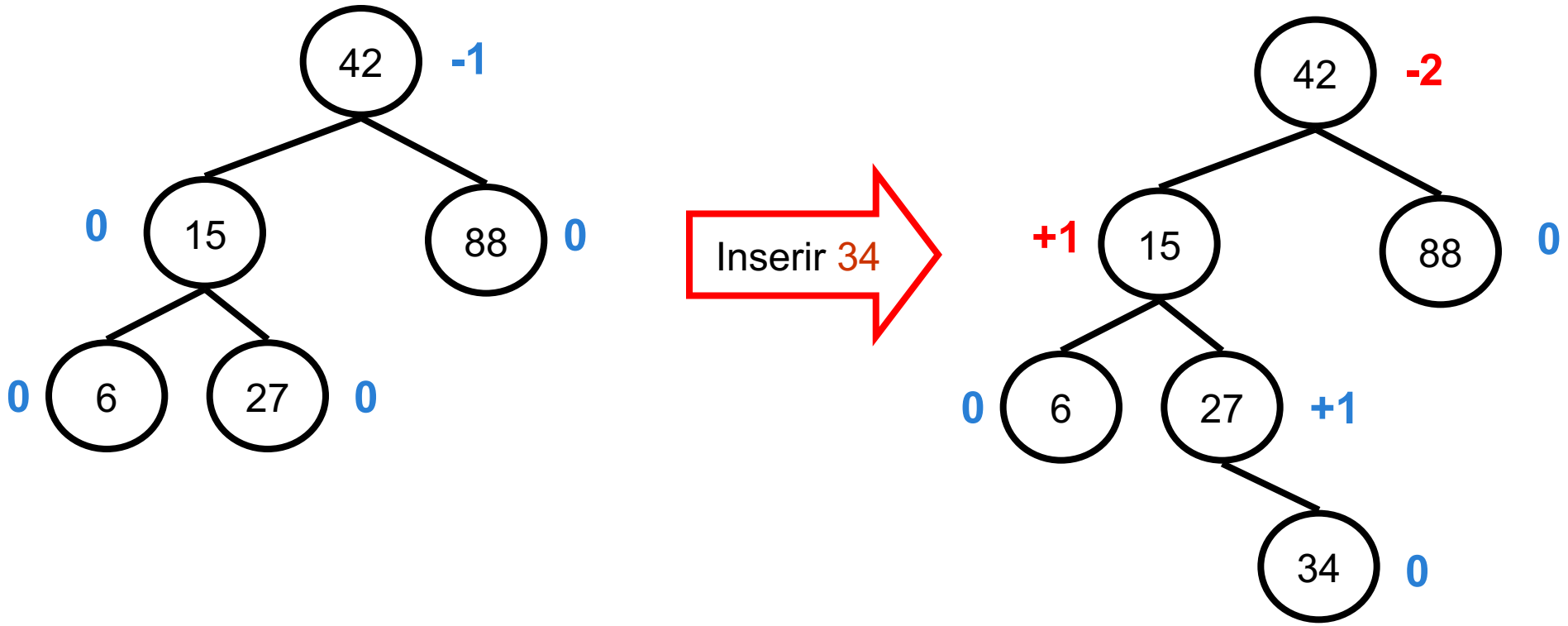


EXEMPLO 1: ROTAÇÃO DUPLA DIREITA

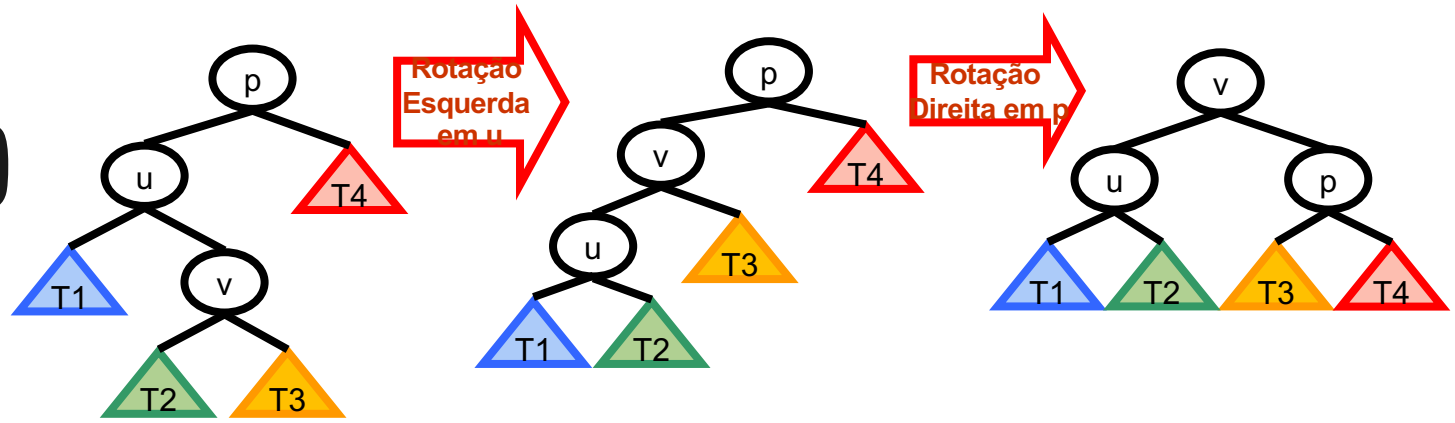


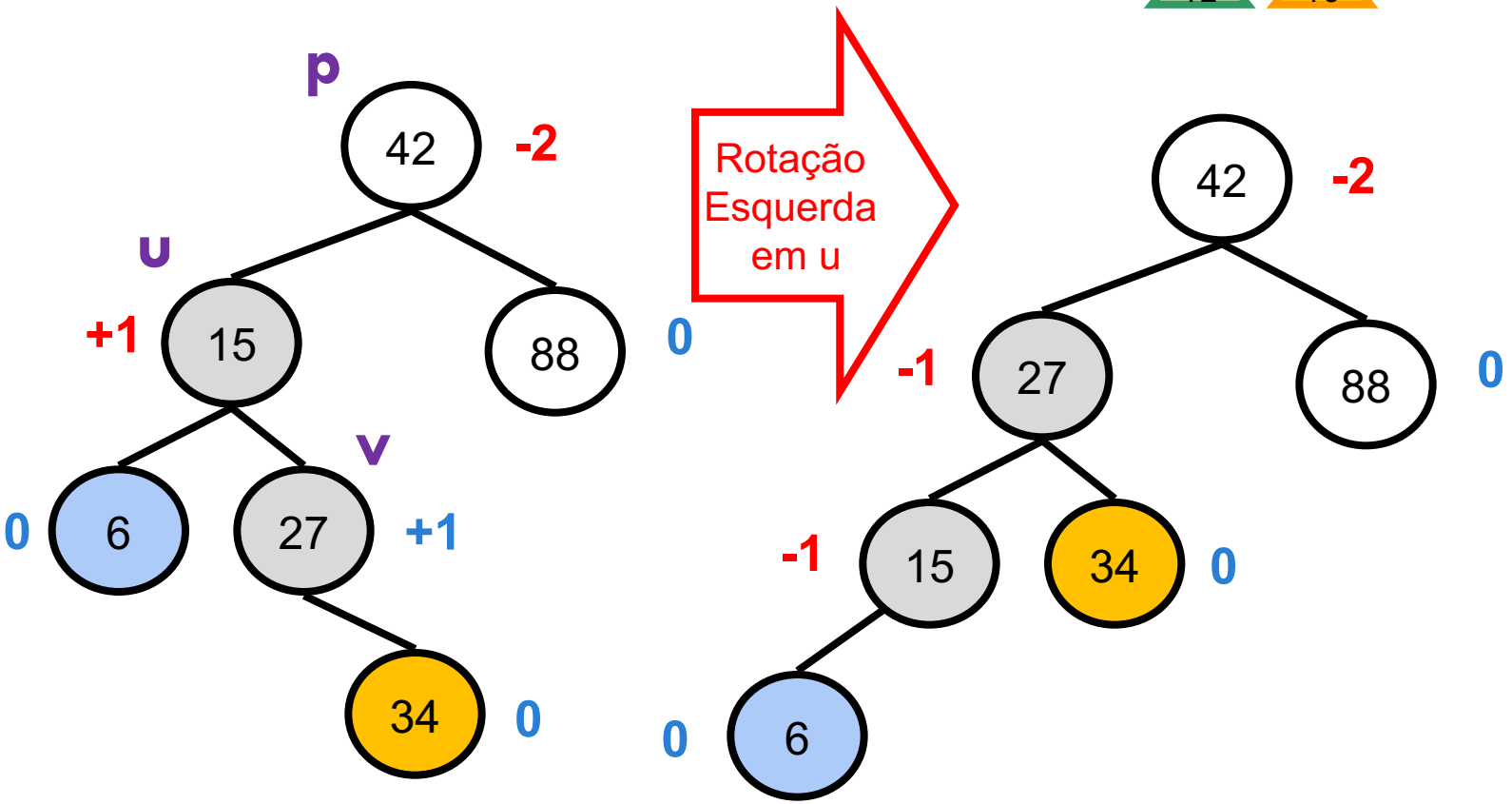


EXEMPLO 2: INSERIR 34

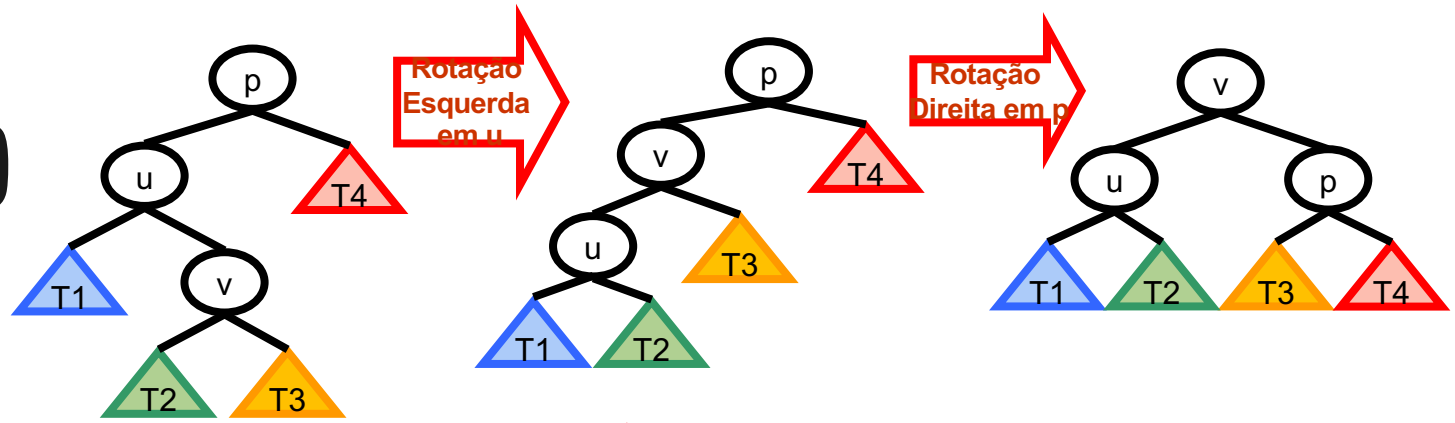


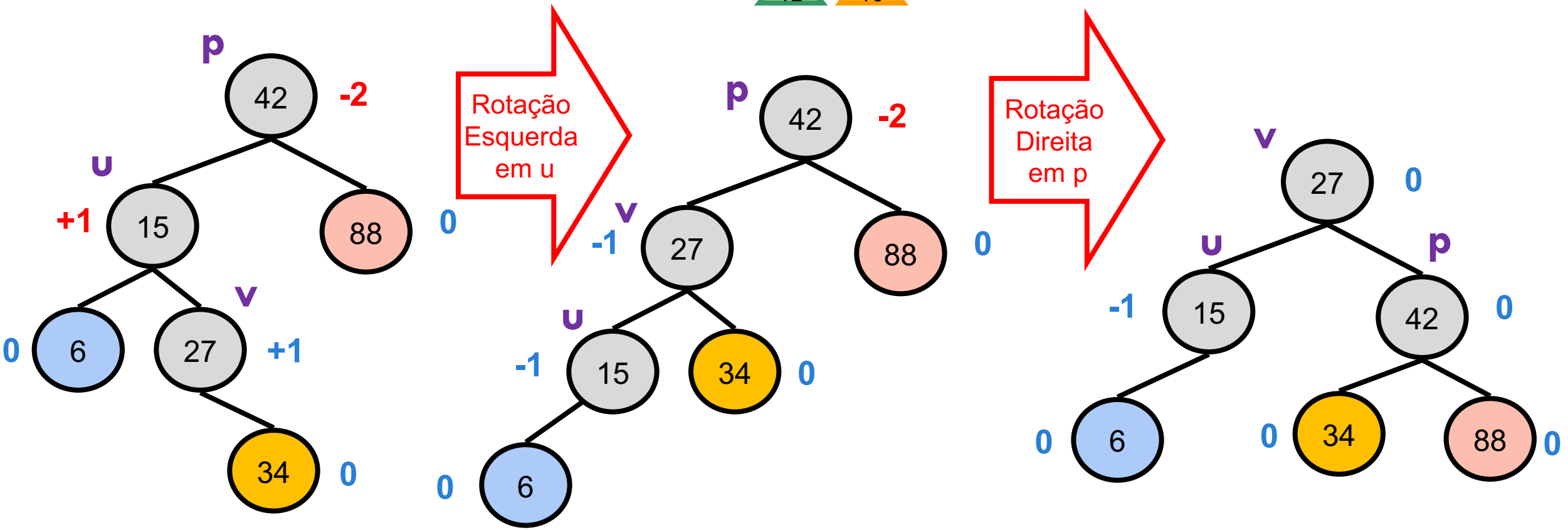
EXEMPLO 2: ROTAÇÃO DUPLA DIREITA



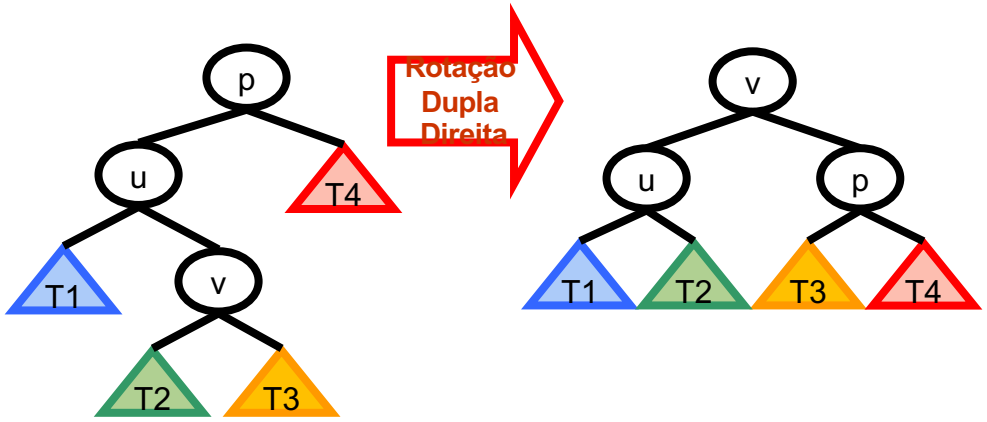


EXEMPLO 2: ROTAÇÃO DUPLA DIREITA





IMPLEMENTAÇÃO ROTAÇÃO DUPLA DIREITA

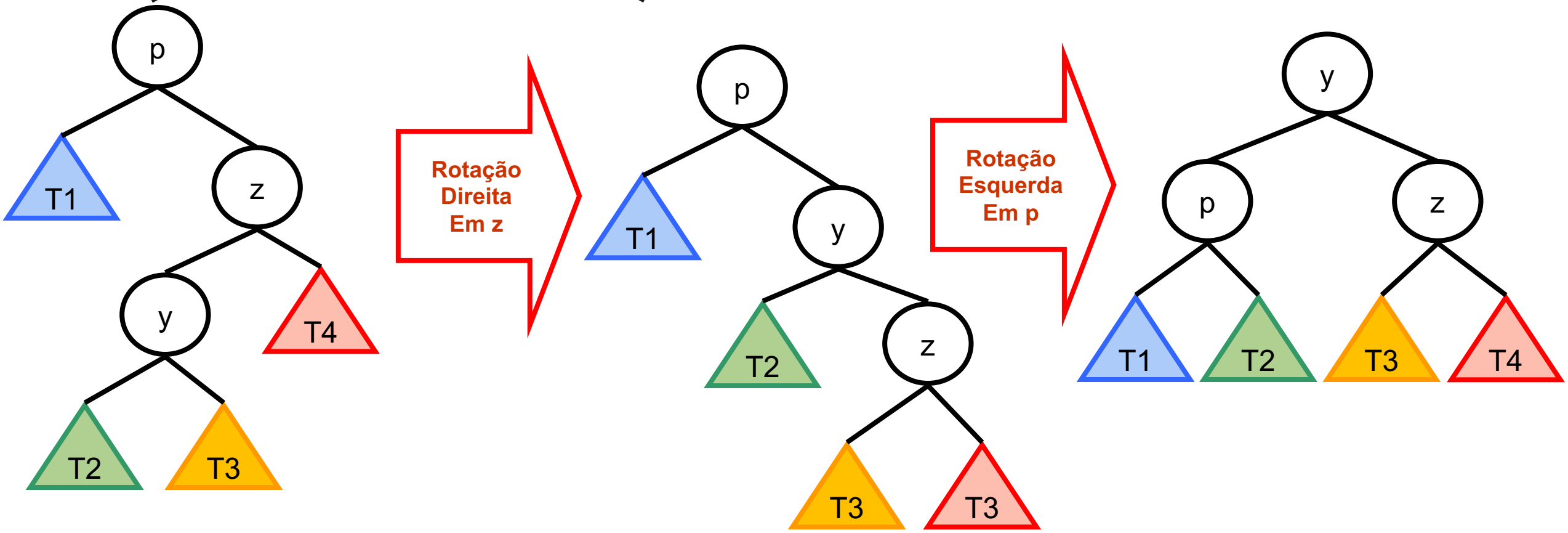



```
pNoA* rotacao_dupla_direita (pNoA* pt){
    rotacao_esquerda(pt->esq);
    rotacao_direita(pt);
    return pt;
}
```

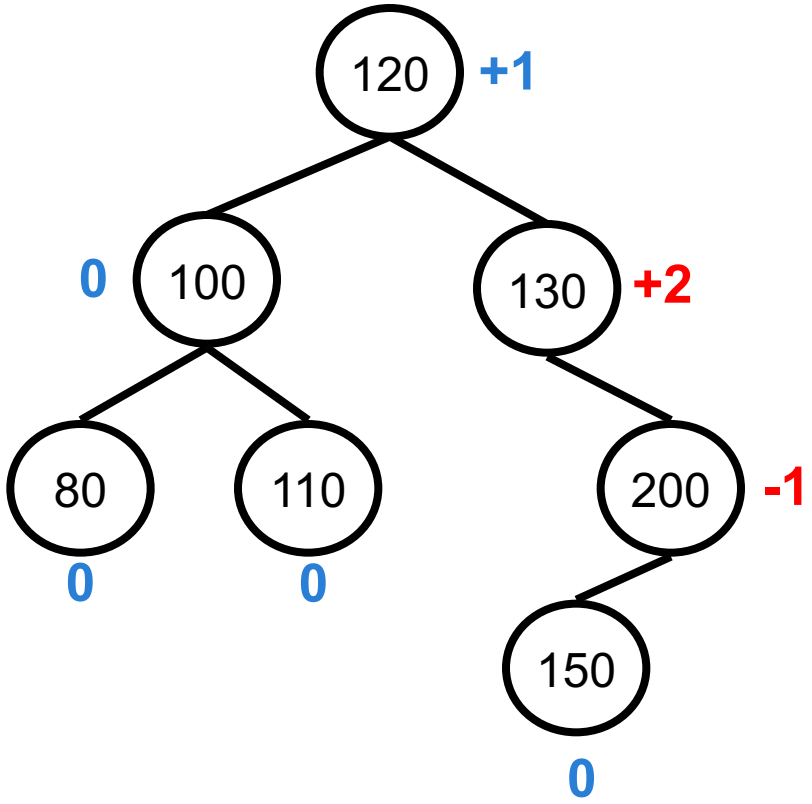
ROTAÇÃO DUPLA ESQUERDA (DIREITA-ESQUERDA)

Aplicar toda vez que uma subárvore ficar com um **FB positivo** e sua **subárvore direita** tem com um **FB negativo**

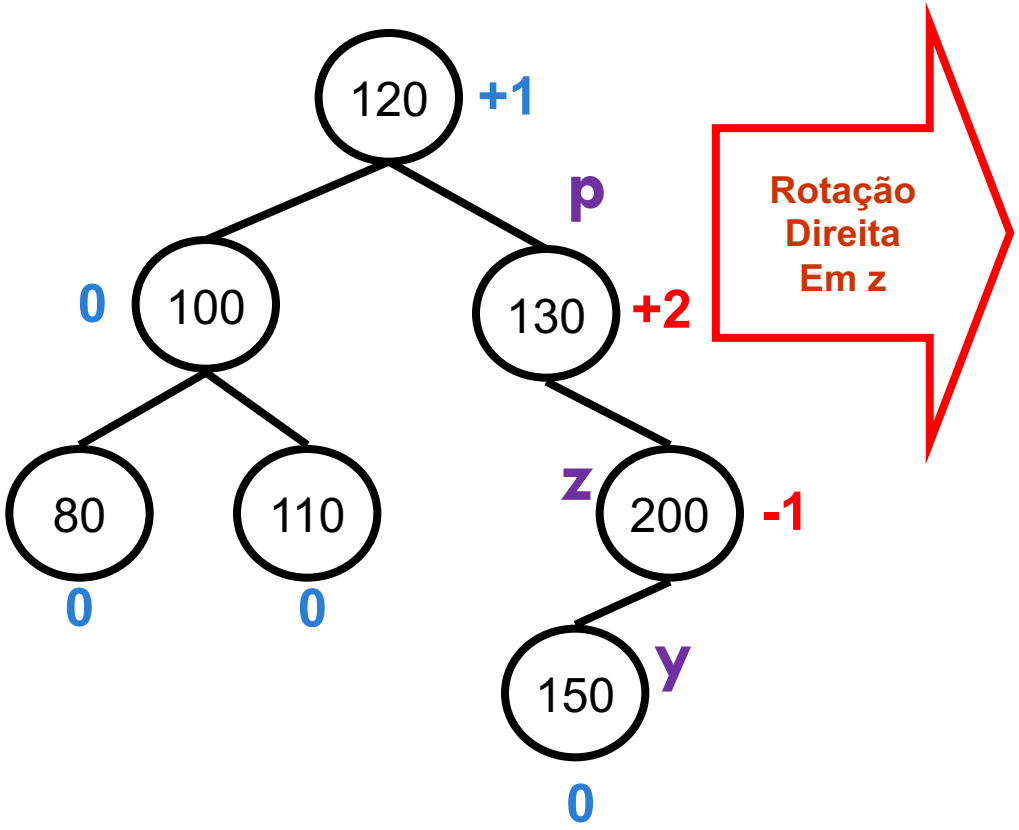
ROTAÇÃO DUPLA ESQUERDA (DIREITA-ESQUERDA)

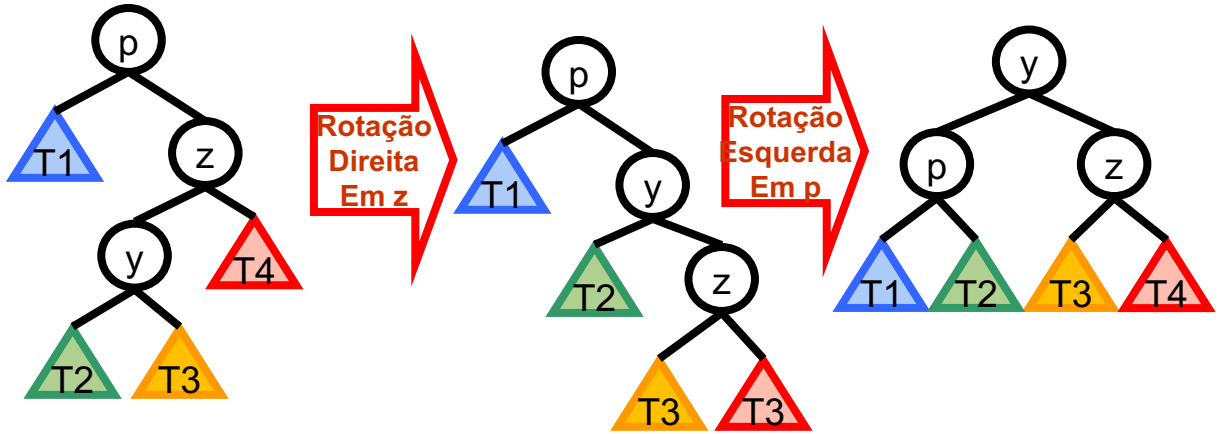


EXEMPLO 1: ROTAÇÃO DUPLA ESQUERDA

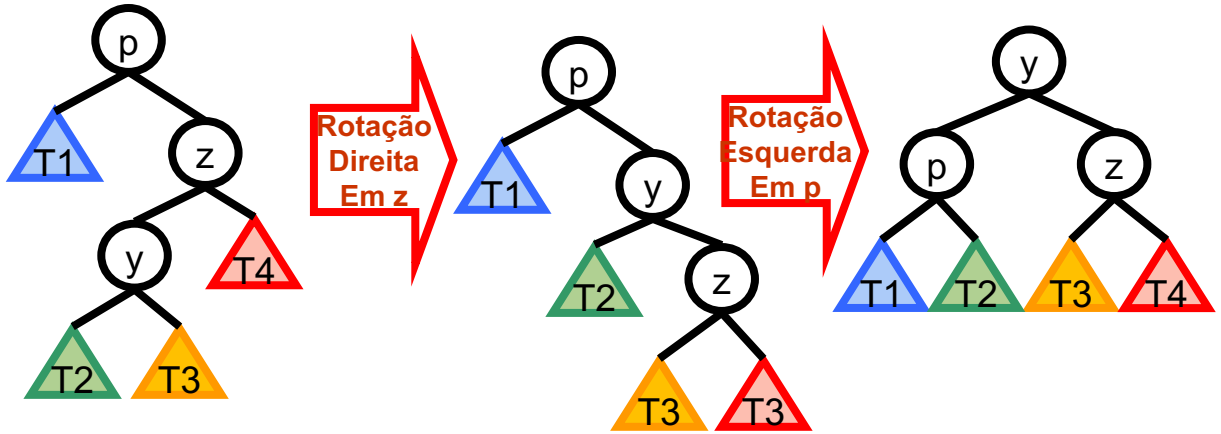


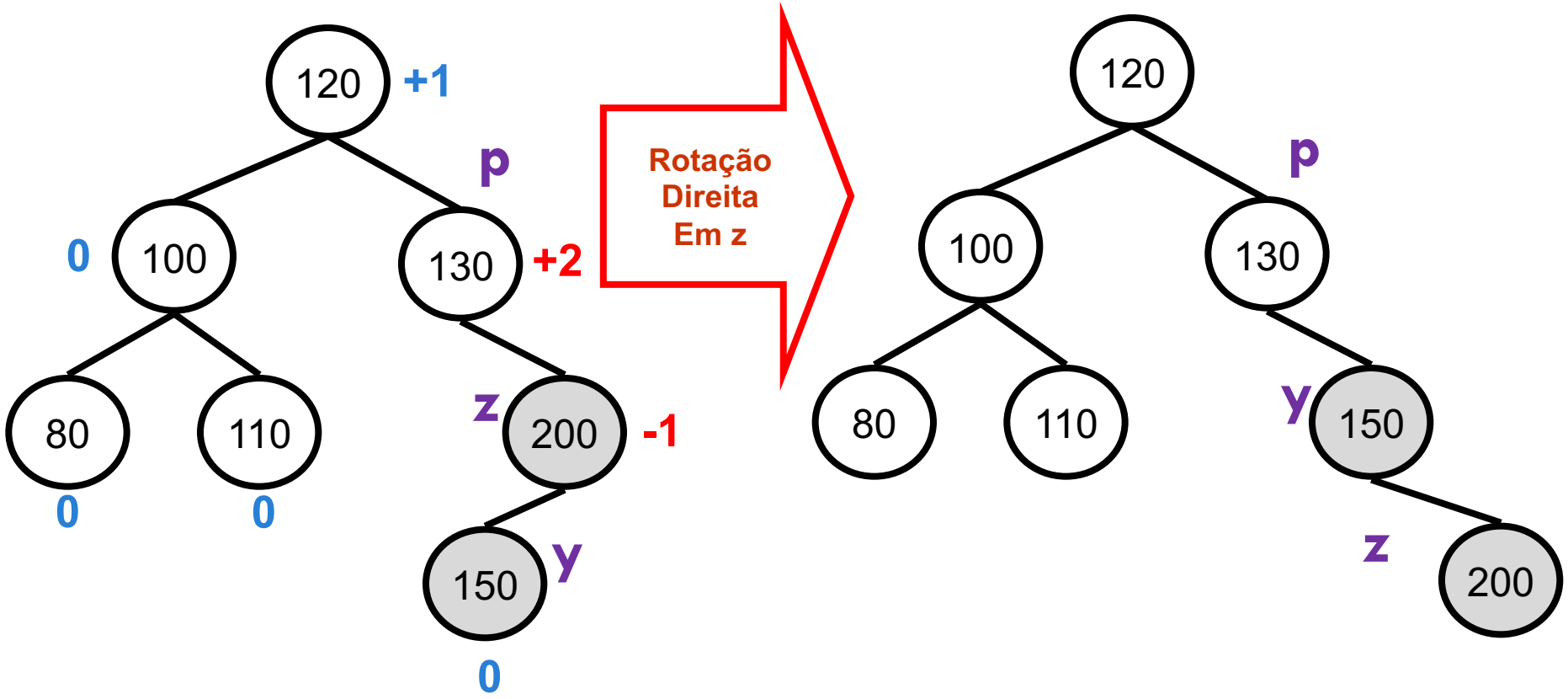
EXEMPLO 1: ROTAÇÃO DUPLA ESQUERDA



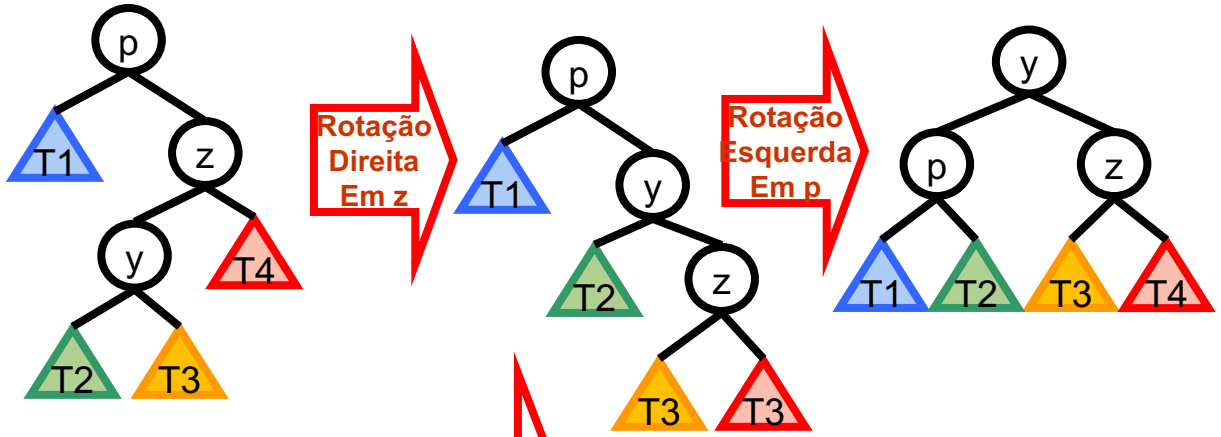


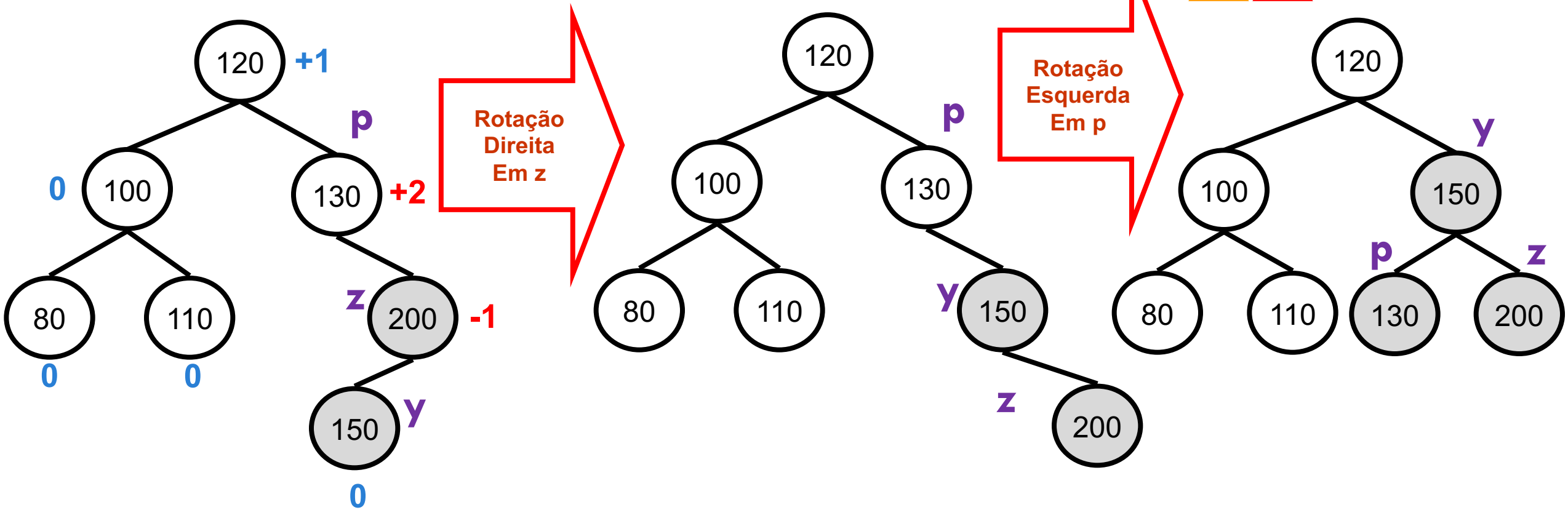
EXEMPLO 1: ROTAÇÃO DUPLA ESQUERDA



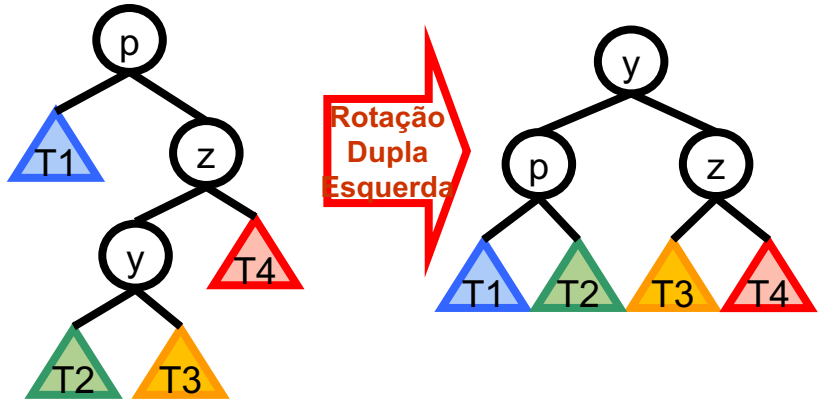


EXEMPLO 1: ROTAÇÃO DUPLA ESQUERDA





IMPLEMENTAÇÃO ROTAÇÃO DUPLA ESQUERDA




```
pNoA rotacao_dupla_esquerda (pNoA *pt){
    rotacao_direita(pt->dir);
    rotacao_esquerda(pt);
    return pt;
}
```

INSERÇÃO DE NÓS EM ÁRVORES AVL

Percorre-se a árvore verificando se a chave já existe ou não

- Em caso positivo, encerra a tentativa de inserção
- Caso contrário, a busca encontra o local correto de inserção do novo nó

Verifica-se se a inclusão tornará a árvore desbalanceada

- Em caso negativo, o processo termina
- Caso contrário, deve-se efetuar o balanceamento da árvore

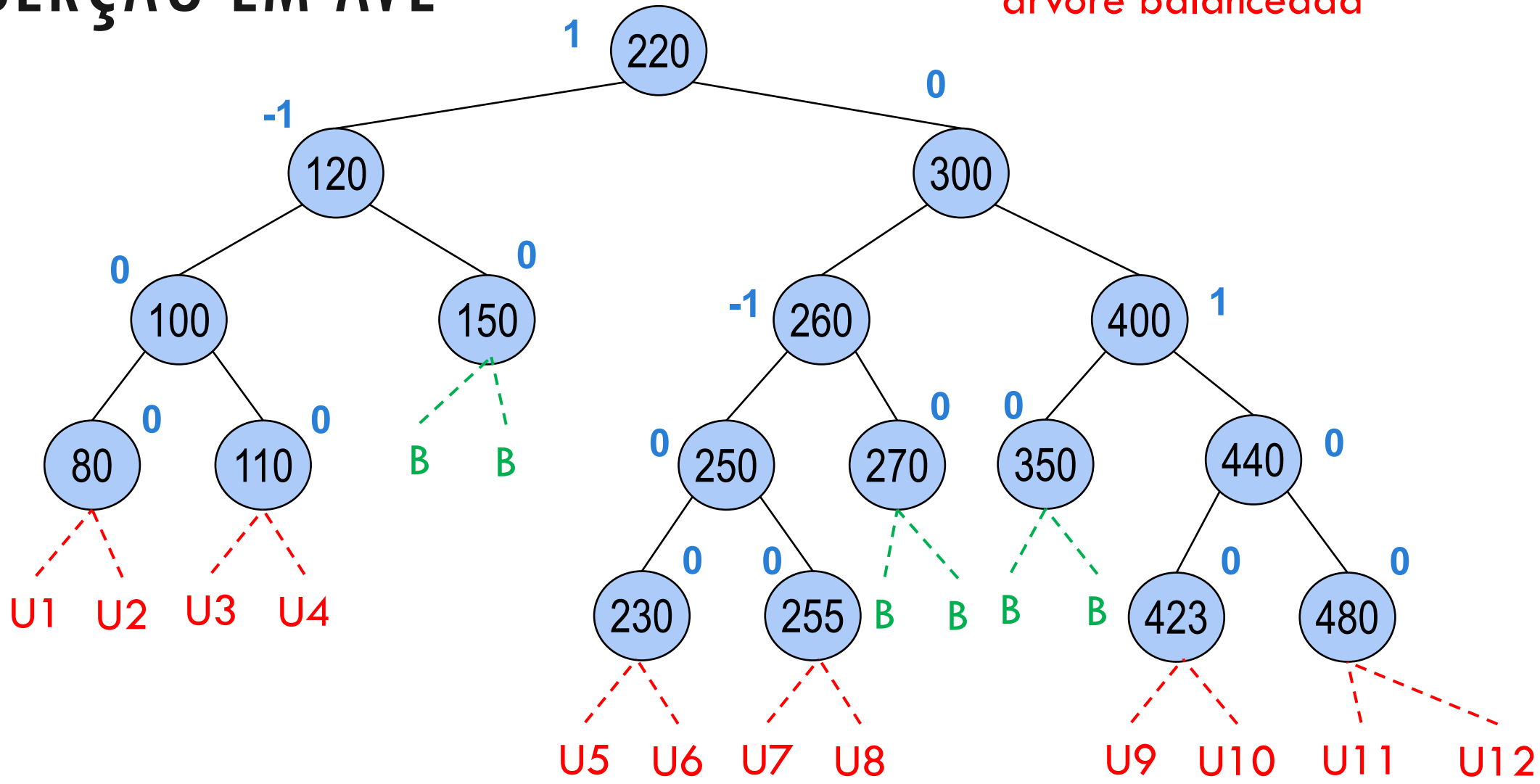
Descobre-se qual a operação de rotação a ser executada

Executa-se a rotação

INTUIÇÃO: INSERÇÃO EM AVL

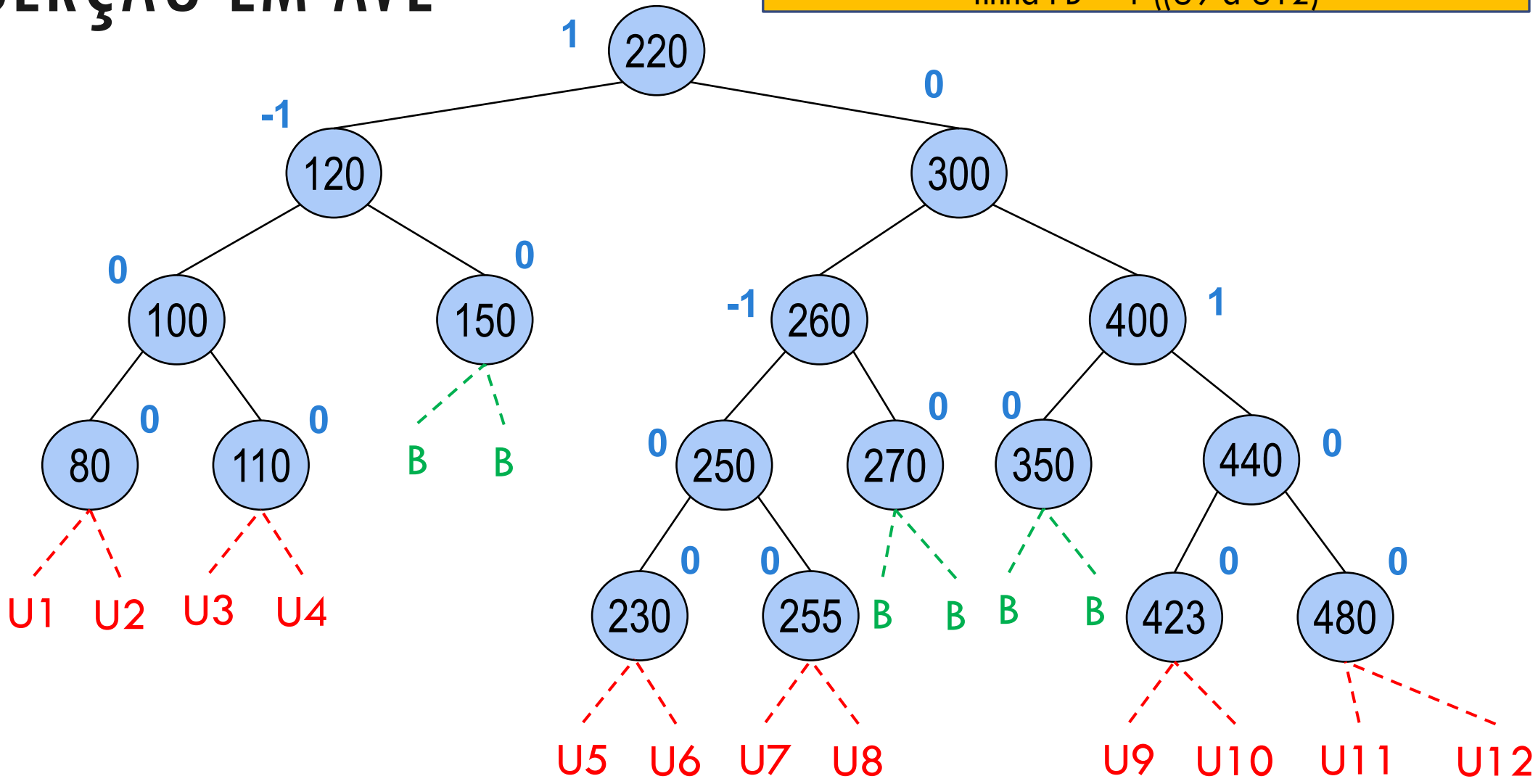
B – Inserções mantêm a árvore balanceada

Un – Inserções **não** mantêm a árvore balanceada



INTUIÇÃO: INSERÇÃO EM AVL

Desbalanceamento:
Se nó inserido é **descendente esquerdo** de um nó que tinha FB = -1 (U1 a U8)
OU
Se nó inserido é **descendente direito** de um nó que tinha FB = 1 ((U9 a U12)



REBALANCEAMENTO: DOIS CASOS

Caso 1

- A raiz de uma subárvore tem FB = 2 (ou -2) e tem um filho com FB = -1 (ou 1), isto é, FB com **sinal oposto ao do pai**

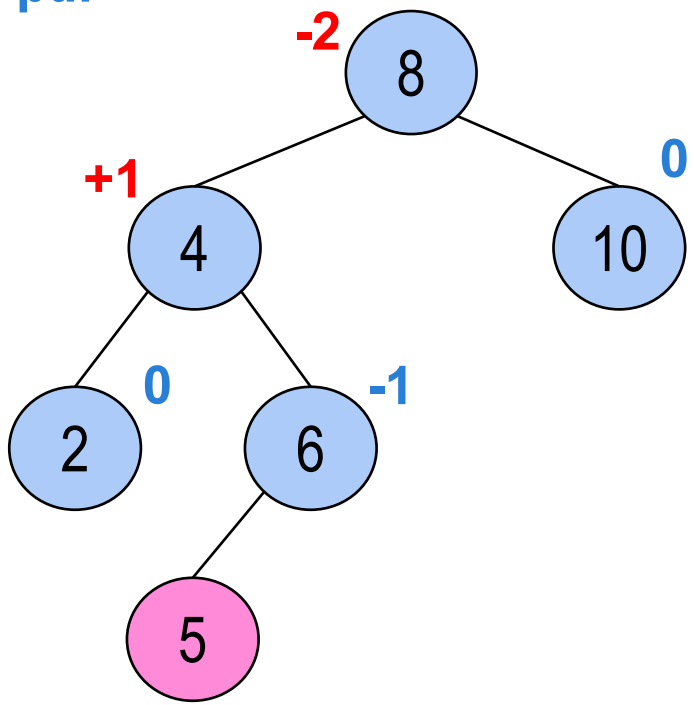
Caso 2

- A raiz de uma subárvore tem FB= -2 (ou 2) e tem filho com FB = -1 (ou 1), i.e., **com mesmo sinal do pai**.

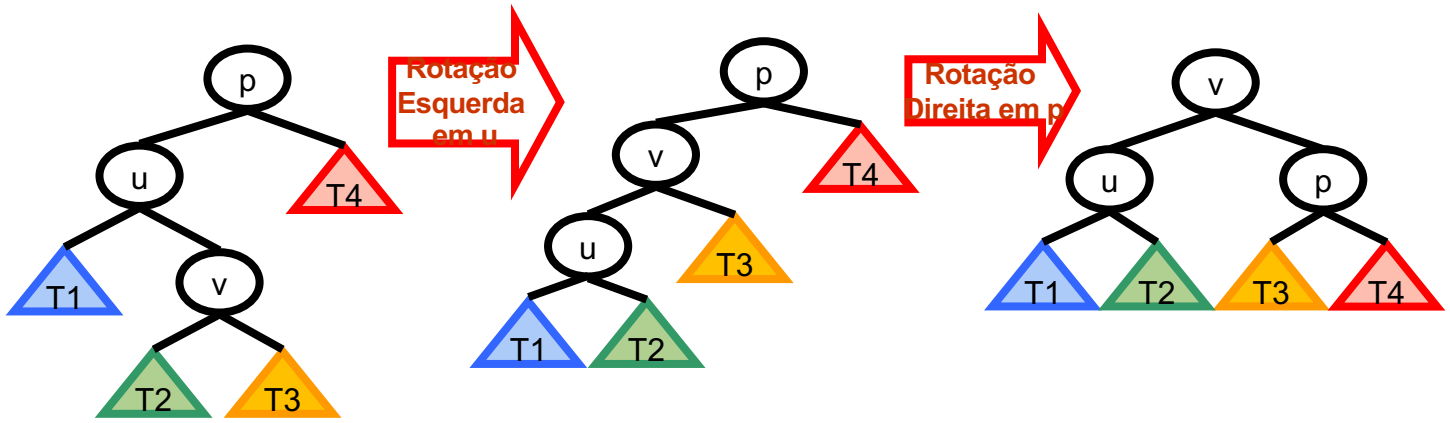
CASO 1

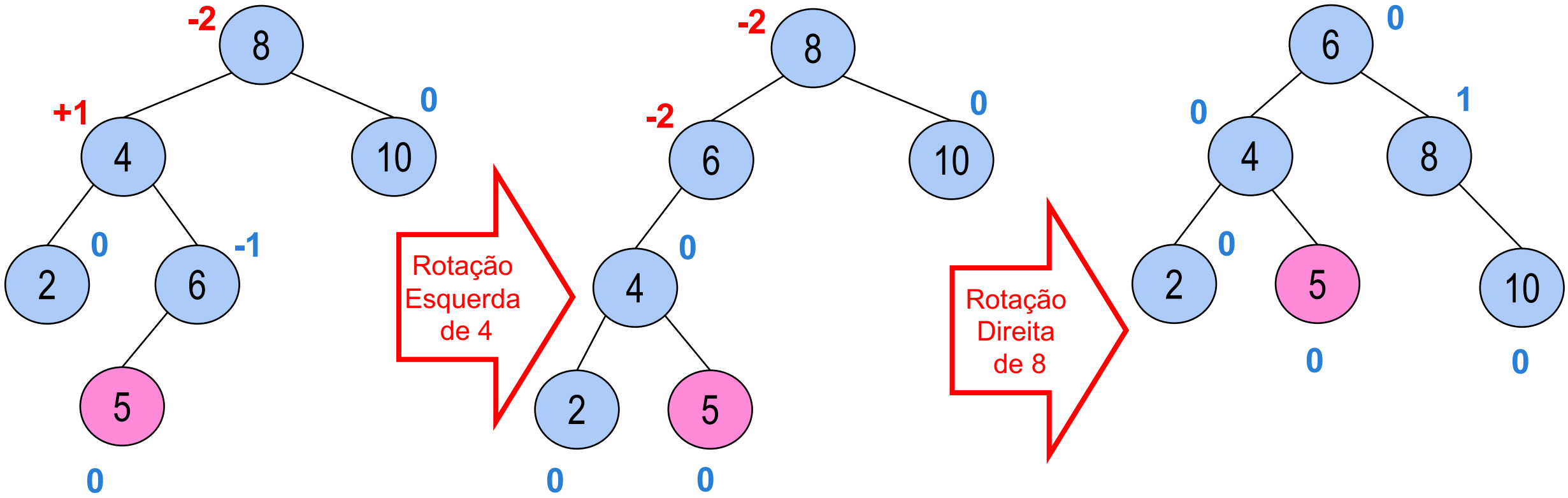
A raiz de uma subárvore tem FB = 2 (ou -2) e tem um filho com FB = -1 (ou 1), isto é, FB com **sinal oposto ao do pai**

Exemplo: Inserção de 5



CASO 1: EXEMPLO





CASO 1: RESUMO

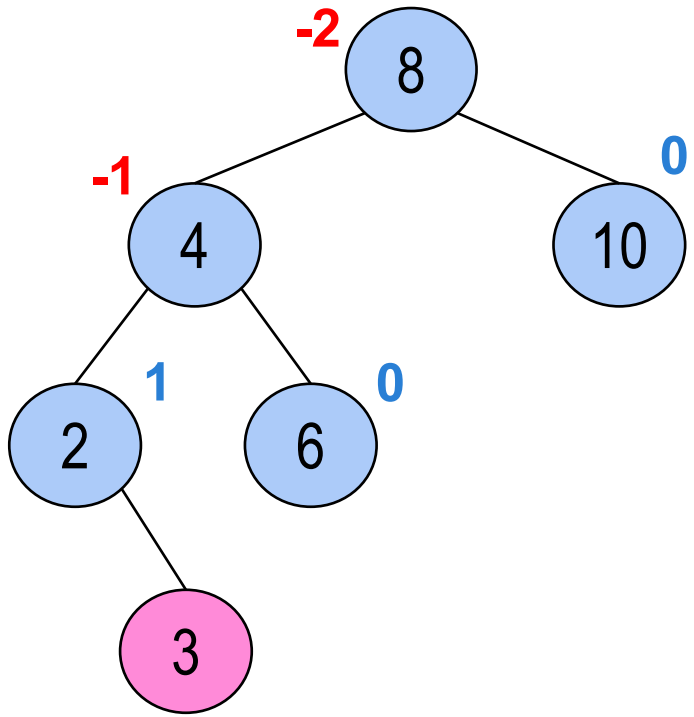
Requer uma rotação dupla: ESQUERDA-DIREITA ou DIREITA-ESQUERDA

1. Rotacionar o nó com FB = -1 (ou 1) na direção apropriada
   a. se FB negativo, rotação para a direita;
   b. se FB positivo, rotação para a esquerda.
2. Rotacionar o nó com FB = -2 (ou 2) na direção oposta

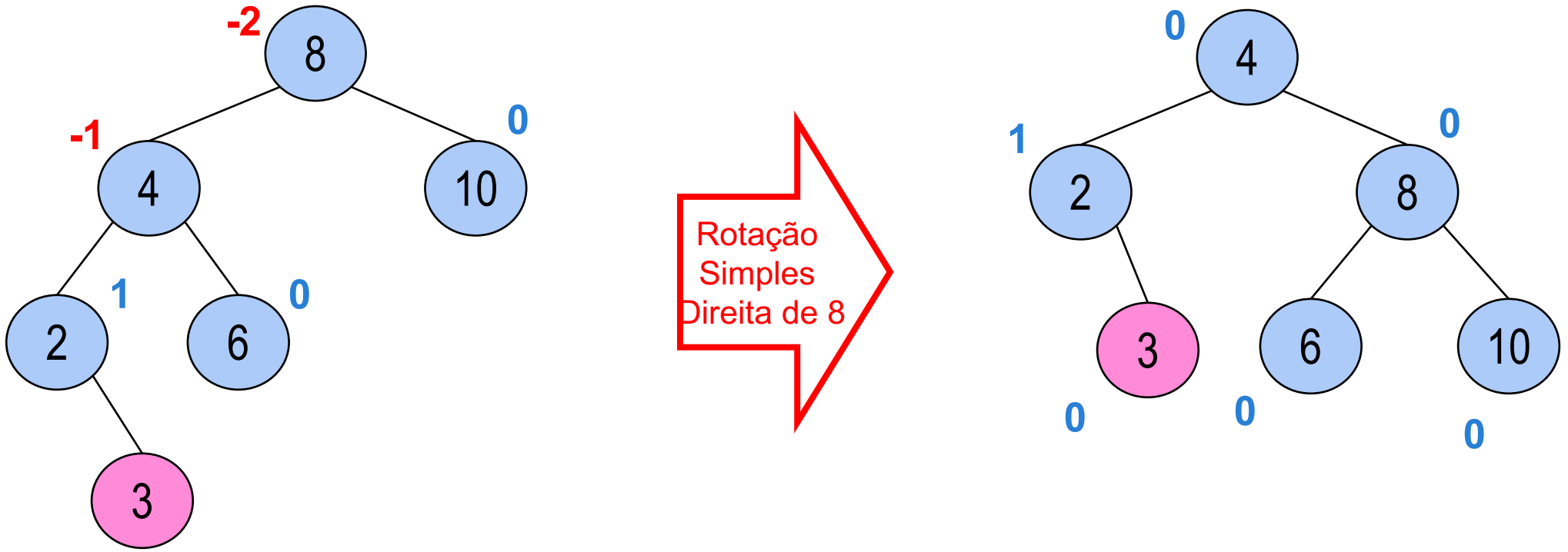
CASO 2

A raiz de uma subárvore tem FB= -2 (ou 2) e tem filho com FB = -1 (ou 1), i.e., **com mesmo sinal do pai**.

Exemplo: Inserção de 3



CASO 2: EXEMPLO



CASO 2: RESUMO

Rotacionar uma única vez o nó de FB = -2 ou 2

- se negativo: rotacionar à direita
- se positivo: rotacionar à esquerda

EXERCÍCIOS

Inserir nós com as seguintes chaves em uma árvore AVL, refazendo a árvore quando houver rotação e anotando as rotações realizadas:

- 50, 40, 30, 45, 47, 55, 56, 1, 2, 3, 49

Na inserção de quais elementos será necessário fazer rotação?

INSERÇÃO DE NÓS EM ÁRVORES AVL: ALGUNS PROBLEMAS

Como saber se a árvore está balanceada?

- Verificando se existe um nó "desregulado"

Como saber se um nó está desregulado?

- Determina-se as alturas de suas sub-árvores e subtrai-se uma da outra

Procedimento muito lento!

Como ser mais eficiente?

- Para cada nó v de uma árvore, armazena-se uma variável **fb** que registra essa diferença (o livro usa **bal** como nome dessa variável)

CÁLCULO EFICIENTE DE FB

Que valores são possíveis para **fb**?

- -1, 0, 1

De novo, como ser eficiente no cálculo do fb?

- Seja **q** o nó inserido.
- Se **q** pertencer à sub-árvore esquerda de v e essa inclusão resultar em aumento na altura da sub-árvore, então fb(v) := fb(v) - 1
  - Se fb(v) = -2, então v está desregulado
- Se **q** pertencer à sub-árvore direita de v e essa inclusão resultar em aumento na altura da sub-árvore, então fb(v) := fb(v) + 1
  - Se fb(v) = +2, então v está desregulado

MANUTENÇÃO DE FB

Mas, quando é que a inclusão de q causa aumento na altura da sub-árvore v?

Suponha que **q** seja incluído na sub-árvore à direita de v

Para **q** incluído na sub-árvore à esquerda, considera-se o caso simétrico.

MANUTENÇÃO DE FB: INSERÇÃO À DIREITA

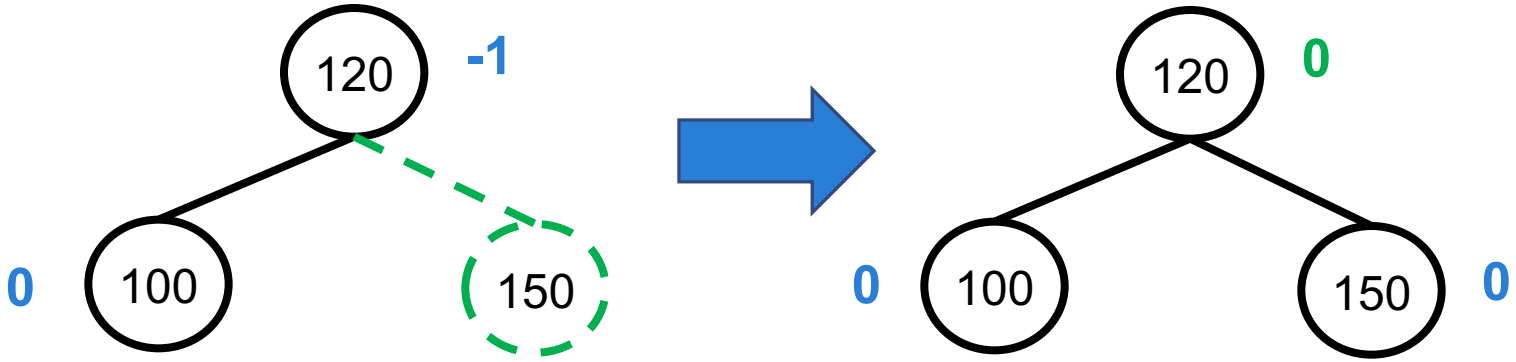
Se, antes da inclusão:

- **fb(v) = -1**, então fb(v) se tornará 0
  - Altura da árvore não foi alterada
  - Por consequência, altura dos outros nós no caminho até a raiz, não se altera também.
- **fb(v) = 0**, então fb(v) se tornará +1
  - Altura da árvore foi modificada
  - Por consequência, altura dos outros nós no caminho até a raiz, pode ter sido alterada também.
  - Repetir o processo (recursivamente), com v substituído por seu pai.
- **fb(v) = +1**, então fb(v) se tornará +2
  - Altura da árvore foi modificada e o nó está desregulado
  - Rotação correta deve ser empregada.
  - Como a árvore será redesenhada, não é necessário verificar os outros nós.

MANUTENÇÃO DE FB: INSERÇÃO À DIREITA

Se, antes da inclusão, fb(v) = -1, então fb(v) se tornará 0

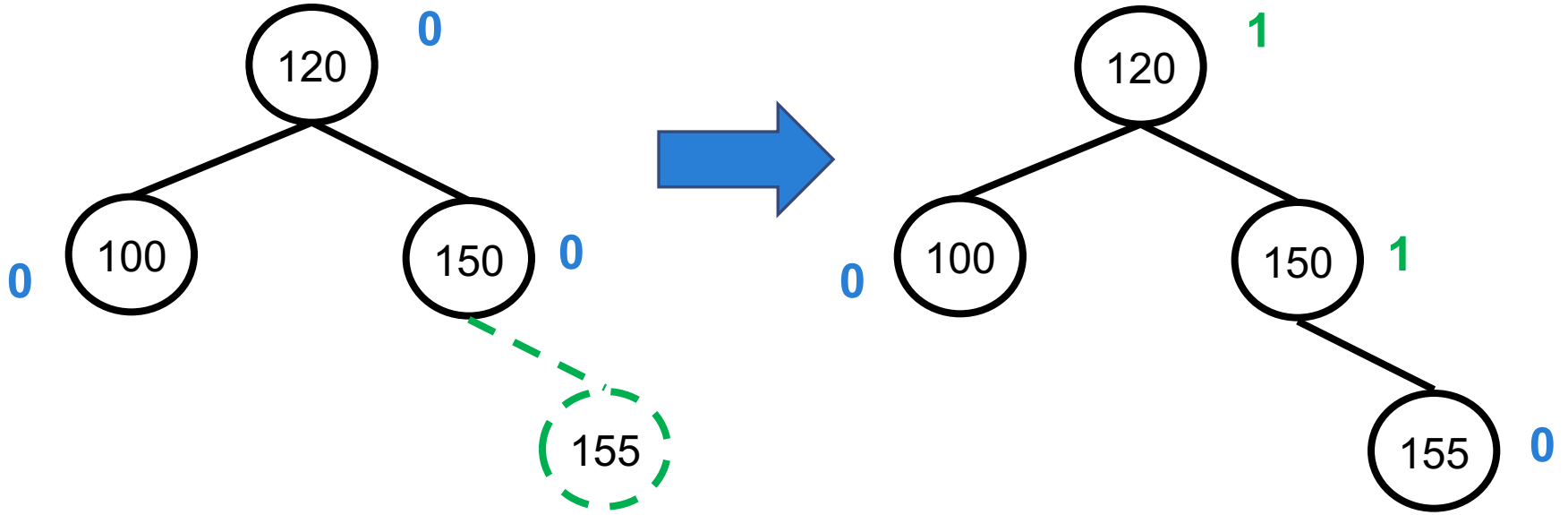
- Altura da árvore não foi alterada
- Por consequência, altura dos outros nós no caminho até a raiz, não se altera também



MANUTENÇÃO DE FB: INSERÇÃO À DIREITA

Se, antes da inclusão, fb(v) = 0, então fb(v) se tornará +1

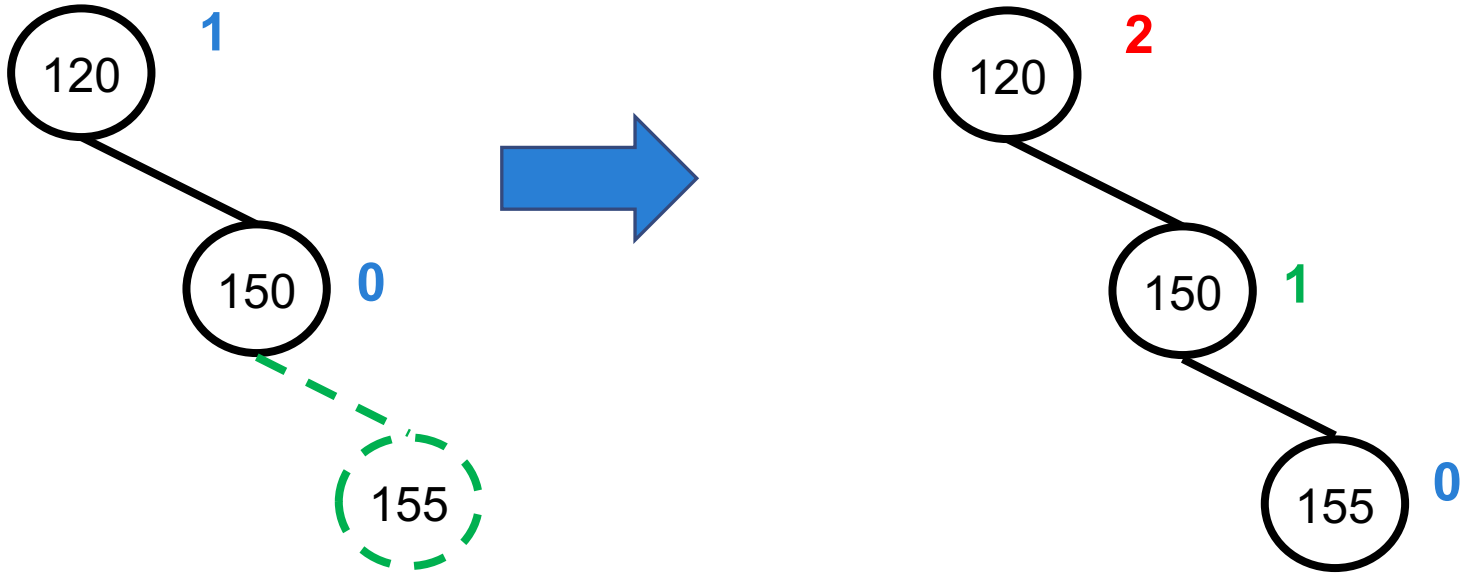
- Altura da árvore foi modificada
- Por consequência, altura dos outros nós no caminho até a raiz, pode ter sido alterada também.
- Repetir o processo (recursivamente), com v substituído por seu pai.



MANUTENÇÃO DE FB: INSERÇÃO À DIREITA

Se, antes da inserção, fb(v) = +1, então fb(v) se tornará +2

- Esse caso só ocorre por propagação de inserção em nó com fb = 0
- Altura da árvore foi modificada e o nó está desregulado
- Rotação correta deve ser empregada.
- Como a árvore será redesenhada, não é necessário verificar os outros nós.



MANUTENÇÃO DE FB: INSERÇÃO À ESQUERDA

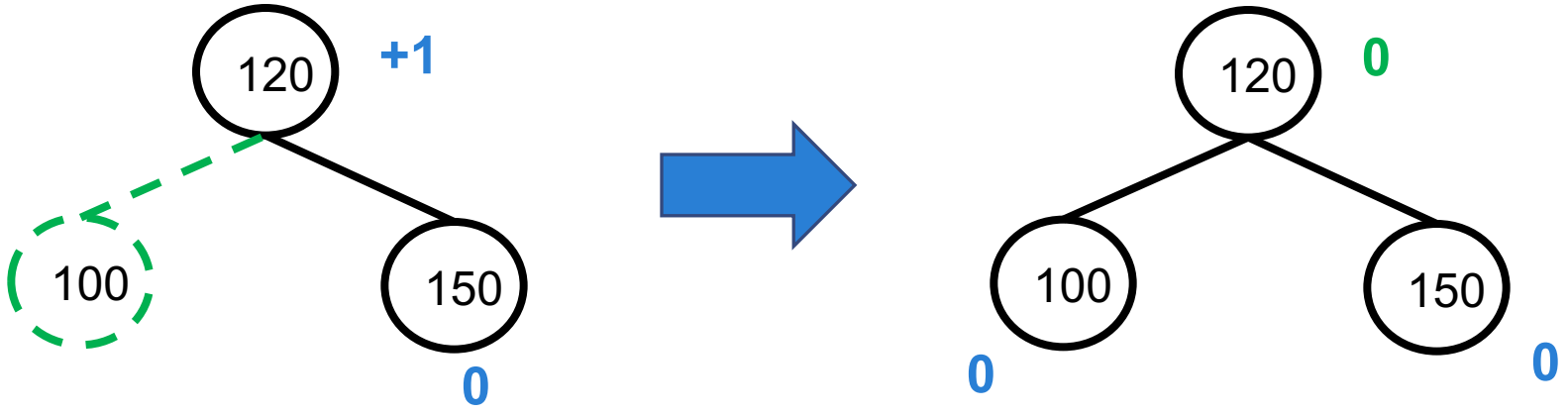
Se, antes da inclusão:

- **fb(v) = +1**, então fb(v) se tornará 0
  - Altura da árvore não foi alterada
  - Por consequência, altura dos outros nós no caminho até a raiz, não se altera também
- **fb(v) = 0**, então fb(v) se tornará -1
  - Altura da árvore foi modificada
  - Por consequência, altura dos outros nós no caminho até a raiz, pode ter sido alterada também
  - Repetir o processo (recursivamente), com v substituído por seu pai
- **fb(v) = -1**, então fb(v) se tornará -2
  - Altura da árvore foi modificada e o nó está desregulado
  - Rotação correta deve ser empregada
  - Como a árvore será redesenhada, não é necessário verificar os outros nós

MANUTENÇÃO DE FB: INSERÇÃO À ESQUERDA

Se, antes da inserção, **fb(v) = +1**, então fb(v) se tornará 0

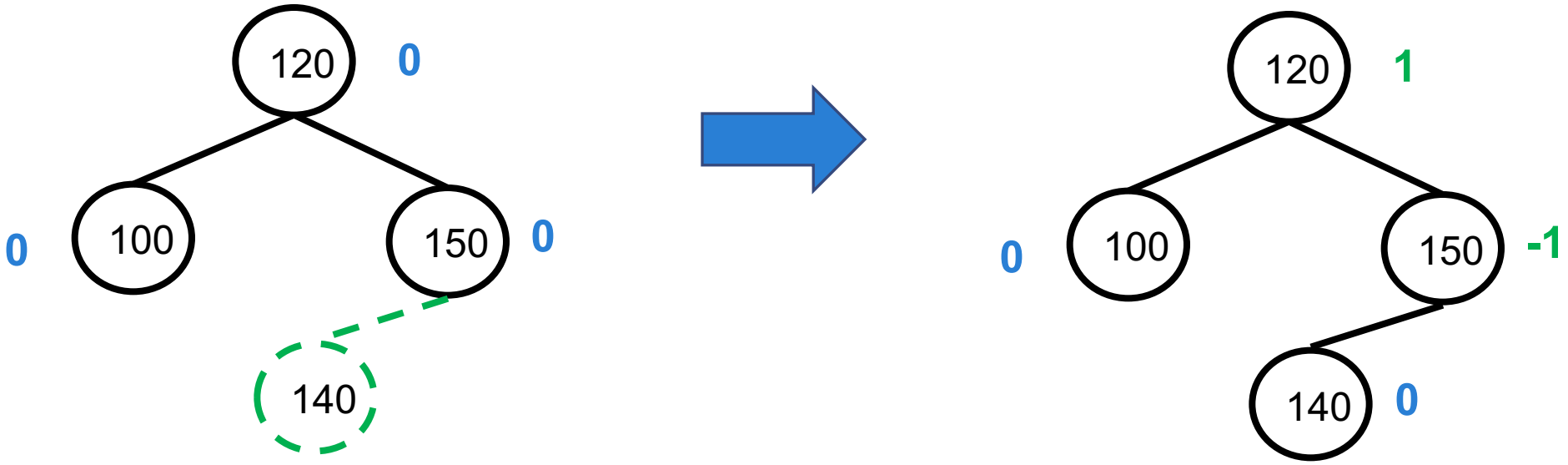
- Altura da árvore não foi alterada
- Por consequência, altura dos outros nós no caminho até a raiz, não se altera também



MANUTENÇÃO DE FB: INSERÇÃO À ESQUERDA

Se, antes da inserção, **fb(v) = 0**, então fb(v) se tornará -1

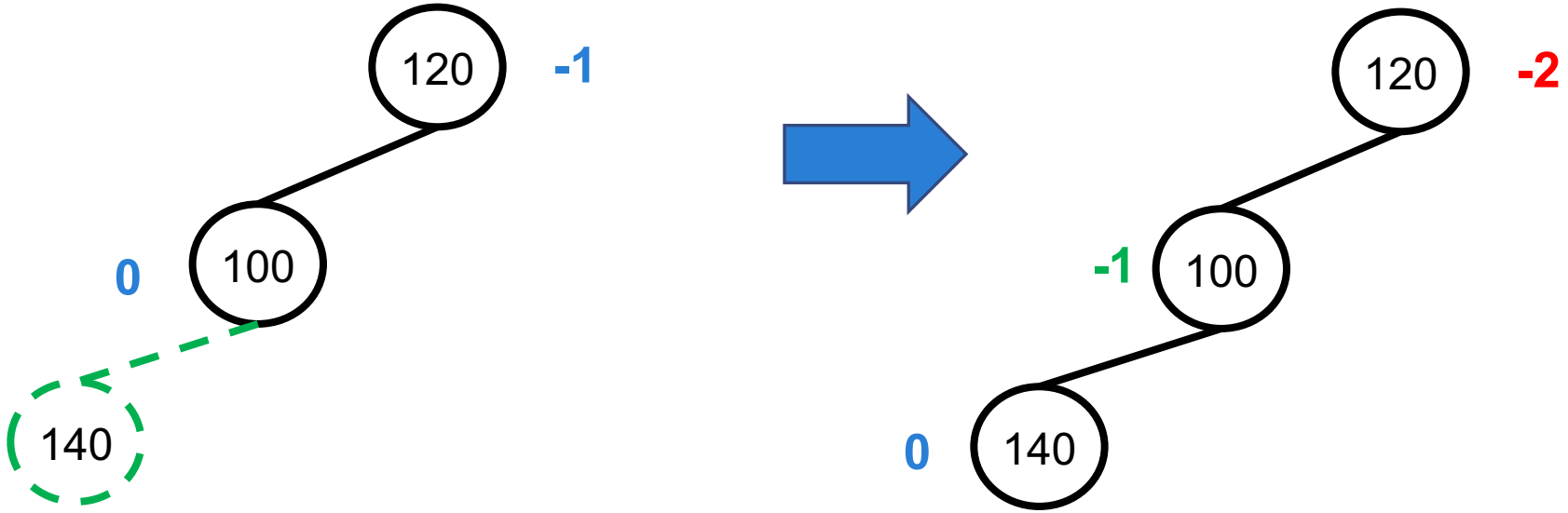
- Altura da árvore foi modificada
- Por consequência, altura dos outros nós no caminho até a raiz, pode ter sido alterada também
- Repetir o processo (recursivamente), com v substituído por seu pai



MANUTENÇÃO DE FB: INSERÇÃO À ESQUERDA

Se, antes da inserção, **fb(v) = -1**, então fb(v) se tornará -2

- Esse caso só ocorre por propagação de inserção em nó com fb = 0
- Altura da árvore foi modificada e o nó está desregulado
- Rotação correta deve ser empregada
- Como a árvore será redesenhada, não é necessário verificar os outros nós



REMOÇÃO DE NÓS EM ÁRVORES AVL

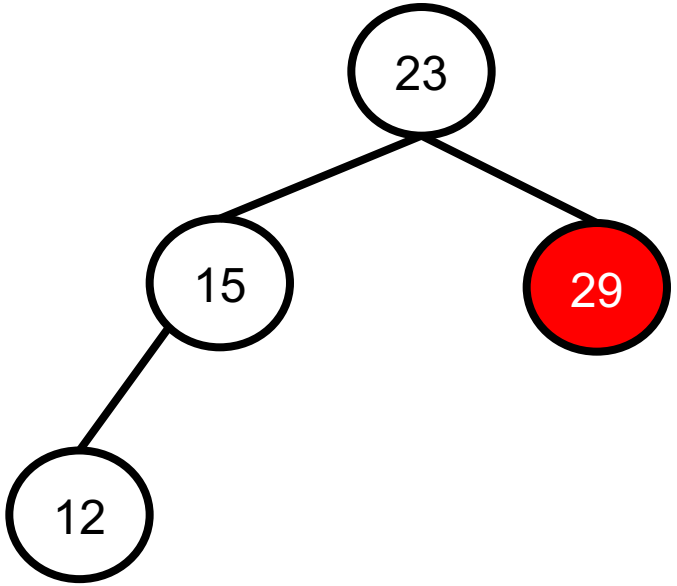
Caso parecido com as inclusões

COMO FAZER REMOÇÃO EM AVL?

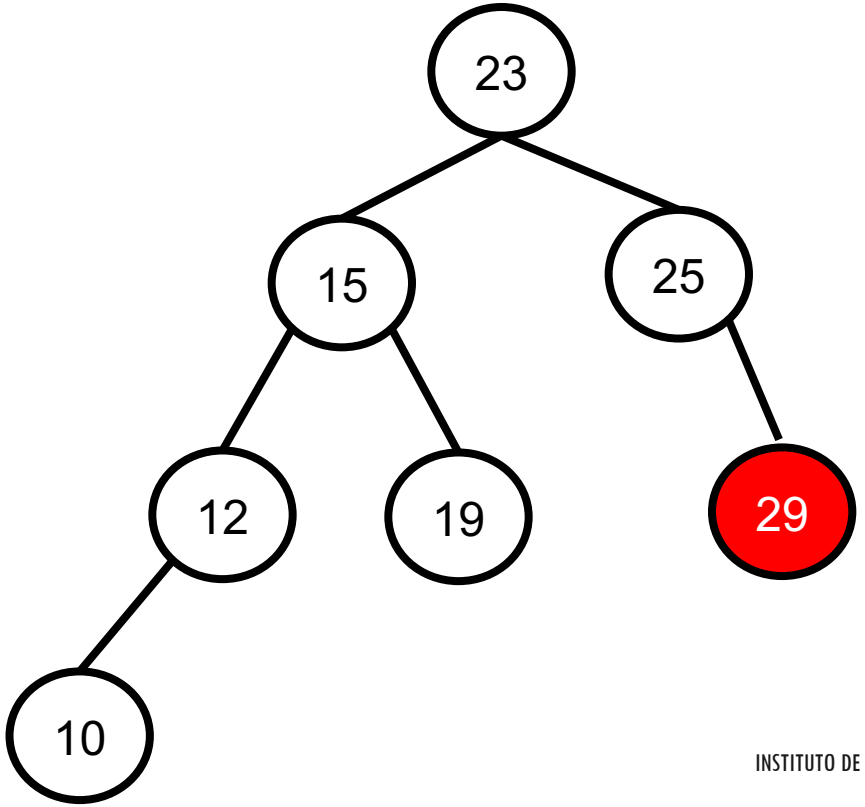
Tentar pensar no caso oposto: quando se remove um elemento, é como se um outro tivesse sido inserido

EXEMPLO

Remoção de 29 = Inserção de 12

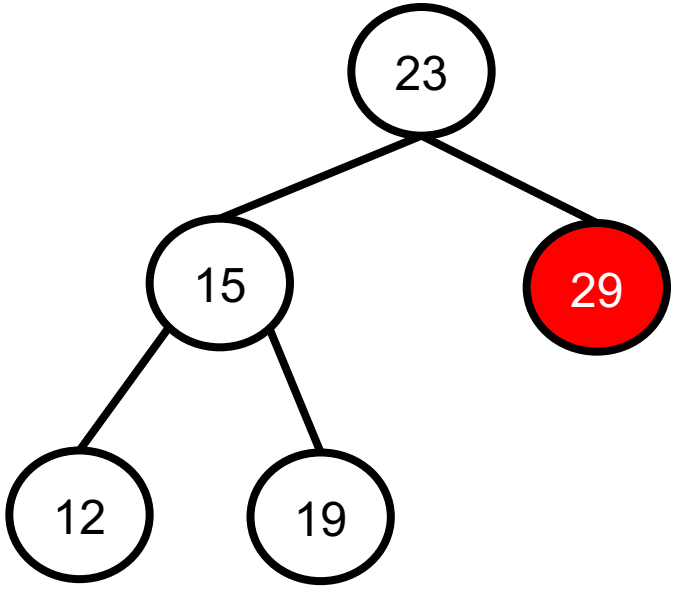


Remoção de 29 = Inserção de 10

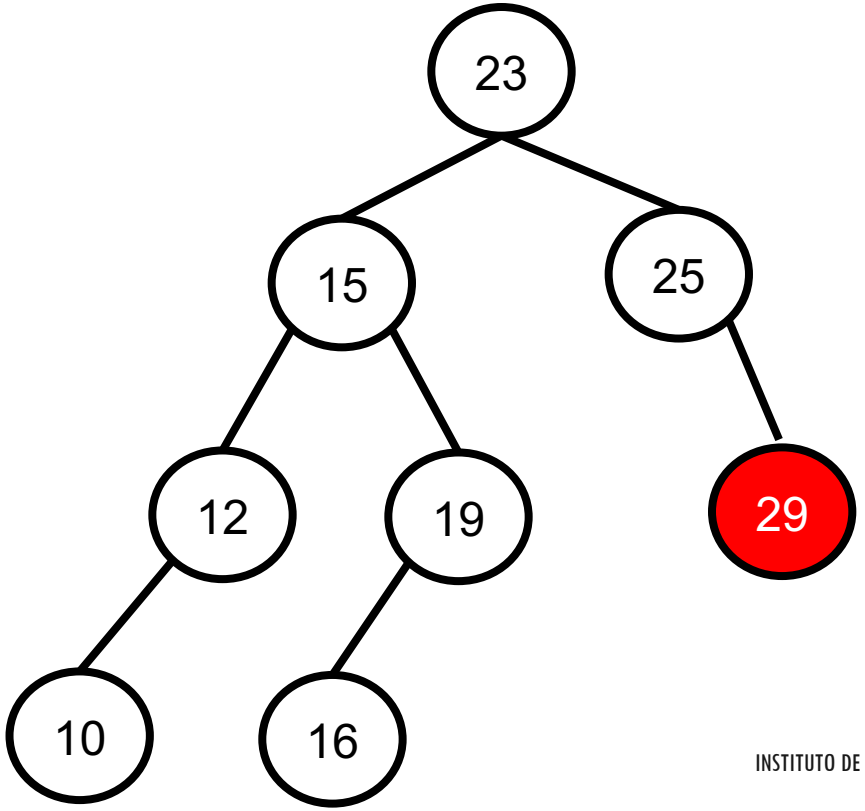


EXEMPLO

Remoção de 29 = Inserção de 12 ou 19



Remoção de 29 = Inserção de 10 ou 16



REBALANCEAMENTO EM REMOÇÃO

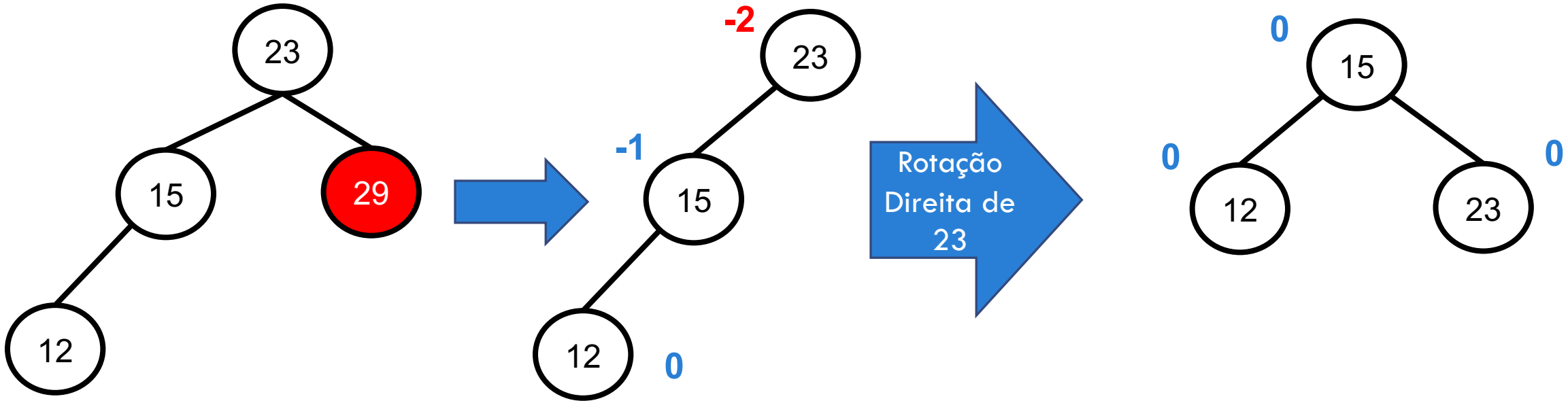
Caso 1

Rotação simples da raiz (FB = 2 ou -2) com filho com FB de mesmo sinal (1 ou -1) ou 0

- Se R tem FB negativo, rotaciona-se para a direita; caso contrário, para a esquerda

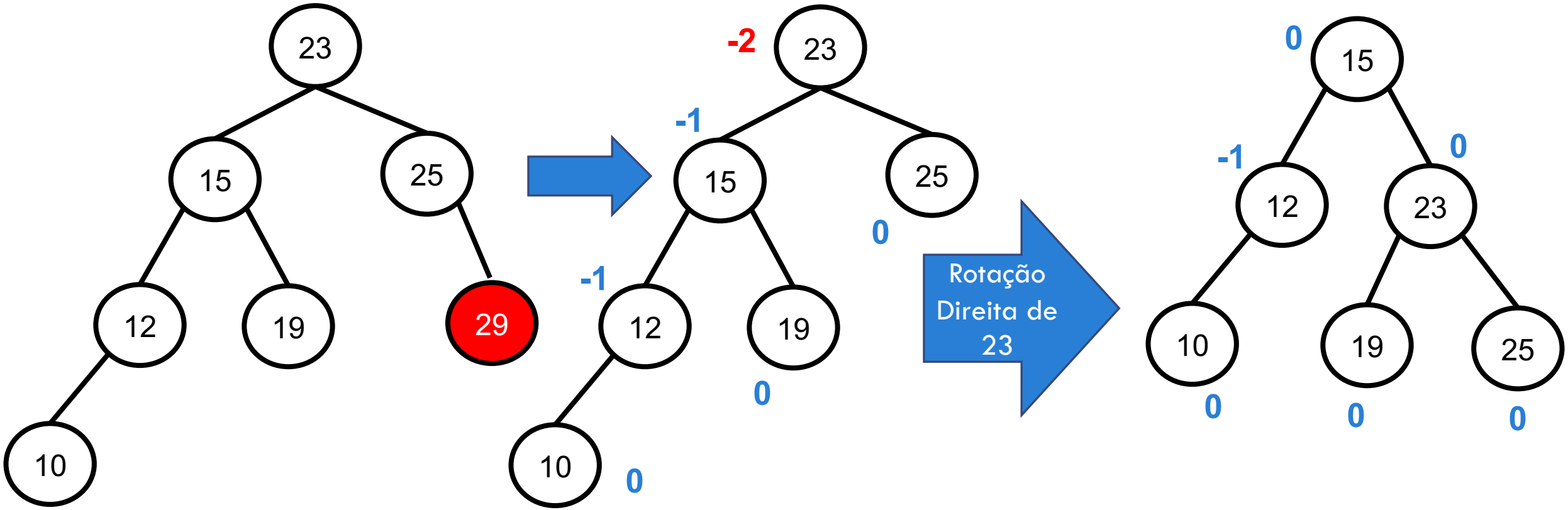
EXEMPLO CASO 1

Mesmo sinal: rotação simples
Rotação de 23 para direita (sinal negativo)



OUTRO EXEMPLO CASO 1

Mesmo sinal: rotação simples
Rotação de 23 para direita (sinal negativo)



REBALANCEAMENTO EM REMOÇÃO

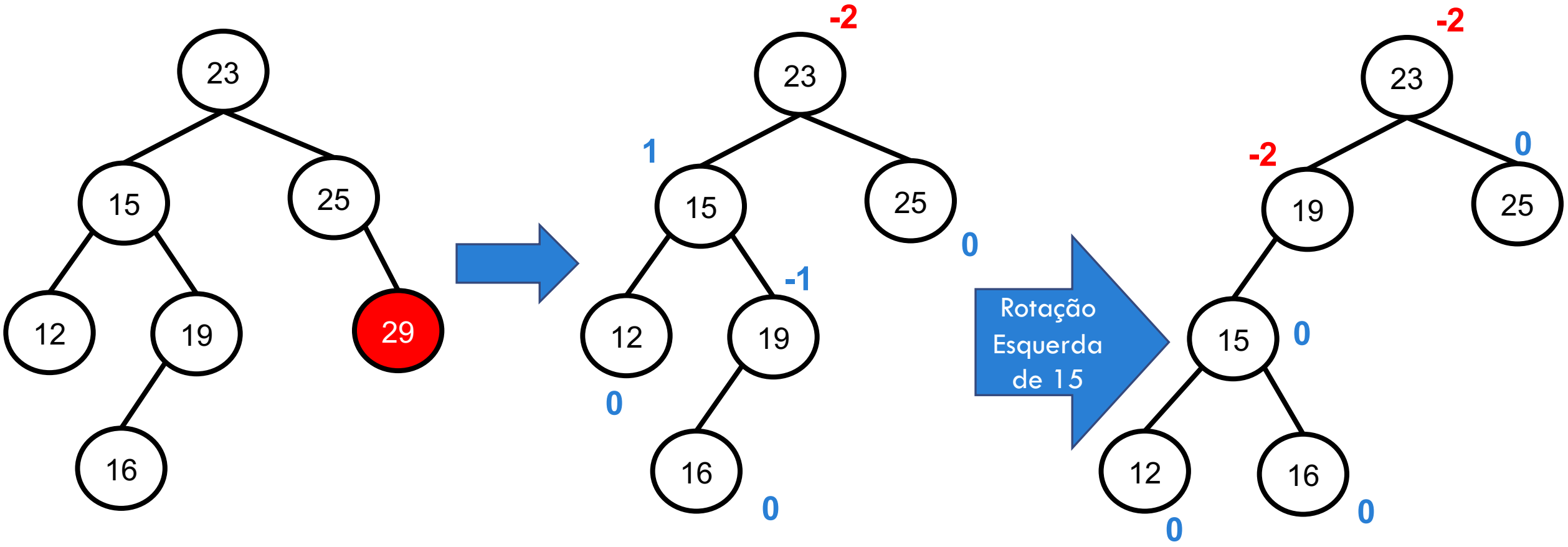
Caso 2

Rotação dupla quando raiz (FB=2 ou -2) e seu filho (1 ou -1) têm FB com sinais opostos

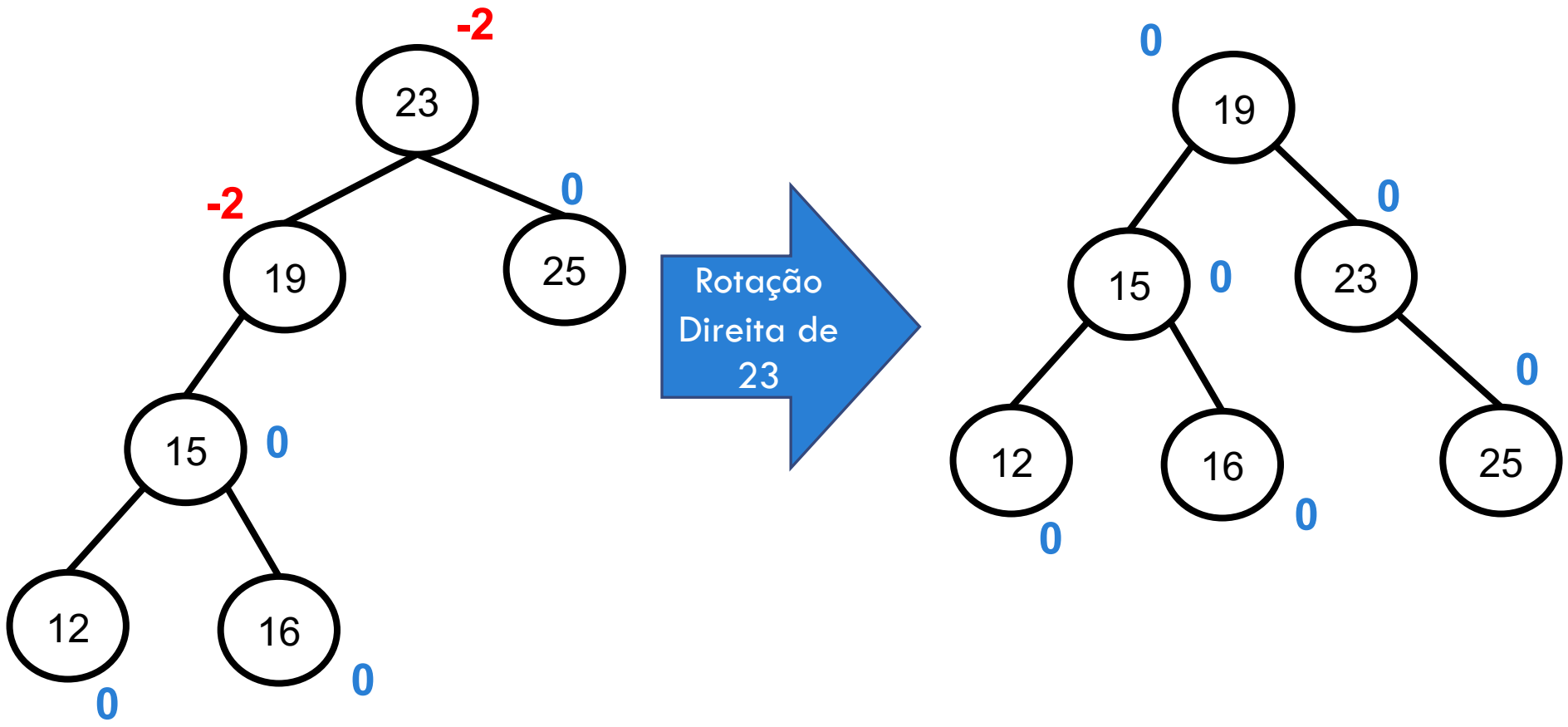
- Rotaciona-se o filho para o lado do desbalanceamento do pai
- Rotaciona-se R para o lado oposto do desbalanceamento

EXEMPLO CASO 2

Sinais opostos: duas rotações
Rotação de 15 para esquerda (lado do desbalanceamento do pai)
Rotação de 23 para direita



EXEMPLO CASO 2 (CONT.)



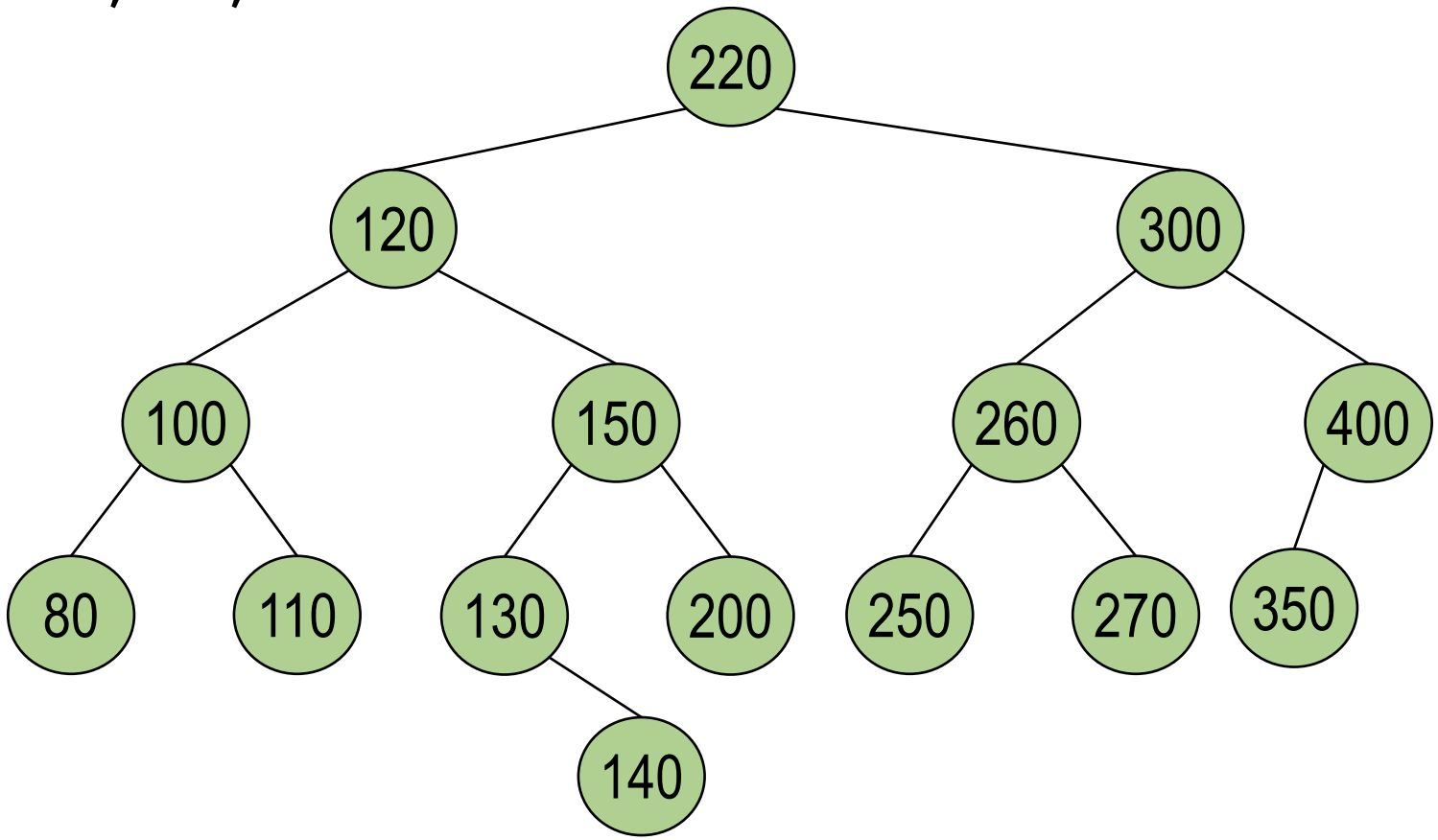
REMOÇÃO DE NÓ INTERMEDIÁRIO

Mesmo raciocínio

Lembrete: se nó excluído tem 2 filhos, substituir pelo nó de maior chave da subárvore esquerda

EXERCÍCIO

Remover os nós de chave 400, 140, 120, 130, 150



REFERÊNCIA

Szwarcfiter, J.; Markezon, L. Estruturas de Dados e seus Algoritmos, 3a. ed. LTC. Cap. 5

AGRADECIMENTOS

Material baseado nos slides de Renata Galante, UFRGS

Material baseado nos slides disponíveis em
http://wiki.icmc.usp.br/images/8/8a/ArvoresAVL.pdf (ICMC-USP)