Algoritmos e Estruturas de Dados I

Árvores AVL

Prof. Tiago Eugenio de Melo
tmelo@uea.edu.br

www.tiagodemelo.info

Observações

- O conteúdo dessa aula é parcialmente proveniente do Capítulo 14 do livro “*Data Structure and Algorithms Using Python*”.
- As palavras com a fonte `Courier` indicam uma palavra-reservada da linguagem de programação.

Árvores AVL

Introdução

Introdução

- Árvore binária de busca (ABB) é uma estrutura de dados apropriada para armazenamento e pesquisa (busca) de dados.

Introdução

- Árvore binária de busca (ABB) é uma estrutura de dados apropriada para armazenamento e pesquisa (busca) de dados.
- A eficiência das operações em ABB depende bastante da altura da árvore.

Introdução

- Árvore binária de busca (ABB) é uma estrutura de dados apropriada para armazenamento e pesquisa (busca) de dados.
- A eficiência das operações em ABB depende bastante da altura da árvore.
- No melhor caso, uma árvore binária de busca terá altura de $\log n$.

Introdução

- Árvore binária de busca (ABB) é uma estrutura de dados apropriada para armazenamento e pesquisa (busca) de dados.
- A eficiência das operações em ABB depende bastante da altura da árvore.
- No melhor caso, uma árvore binária de busca terá altura de *log n*.
- No pior caso, a altura da árvore será *n*.

Introdução

- Árvore binária de busca (ABB) é uma estrutura de dados apropriada para armazenamento e pesquisa (busca) de dados.
- A eficiência das operações em ABB depende bastante da altura da árvore.
- No melhor caso, uma árvore binária de busca terá altura de *log n*.
- No pior caso, a altura da árvore será *n*.
- Assim, devemos buscar construir uma ABB que tenha altura de *log n*.

Introdução

Introdução

- Como construir uma ABB com altura *log n*?

Introdução

- Como construir uma ABB com altura *log n*?
  - Se conhecessemos a lista completa de chaves.

Introdução

- Como construir uma ABB com altura *log n*?
  - Se conhecessemos a lista completa de chaves.
    - Bastaria deixar a lista ordenada e usar a técnica de busca binária.

Introdução

- Como construir uma ABB com altura *log n*?
  - Se conhecessemos a lista completa de chaves.
    - Bastaria deixar a lista ordenada e usar a técnica de busca binária.
    - O problema é que essa técnica exige conhecer antecipadamente todas as chaves e isso não se aplica a muitos problemas reais.

Introdução

- Como construir uma ABB com altura *log n*?
  - Se conhecessemos a lista completa de chaves.
    - Bastaria deixar a lista ordenada e usar a técnica de busca binária.
    - O problema é que essa técnica exige conhecer antecipadamente todas as chaves e isso não se aplica a muitos problemas reais.
  - Nós poderíamos reconstruir a árvore a cada nó que seja adicionado ou removido da árvore.

Introdução

- Como construir uma ABB com altura *log n*?
  - Se conhecessemos a lista completa de chaves.
    - Bastaria deixar a lista ordenada e usar a técnica de busca binária.
    - O problema é que essa técnica exige conhecer antecipadamente todas as chaves e isso não se aplica a muitos problemas reais.
  - Nós poderíamos reconstruir a árvore a cada nó que seja adicionado ou removido da árvore.
    - Mas o tempo para fazer isso torna inviável essa alternativa.

Introdução

- Como construir uma ABB com altura *log n*?
  - Se conhecessemos a lista completa de chaves.
    - Bastaria deixar a lista ordenada e usar a técnica de busca binária.
    - O problema é que essa técnica exige conhecer antecipadamente todas as chaves e isso não se aplica a muitos problemas reais.
  - Nós poderíamos reconstruir a árvore a cada nó que seja adicionado ou removido da árvore.
    - Mas o tempo para fazer isso torna inviável essa alternativa.
  - O que nós precisamos fazer é manter a árvore com uma altura ideal à medida que novos elementos sejam adicionados ou antigos sejam removidos.

AVL

AVL

- A árvore AVL foi inventada por G. M. **A**del'son-**V**elskii e Y. M. **L**andis em 1962.

AVL

- A árvore AVL foi inventada por G. M. **A**del'son-**V**elskii e Y. M. **L**andis em 1962.
- O objetivo dessa árvore é garantir que a altura da árvore esteja sempre balanceada.

AVL

AVL

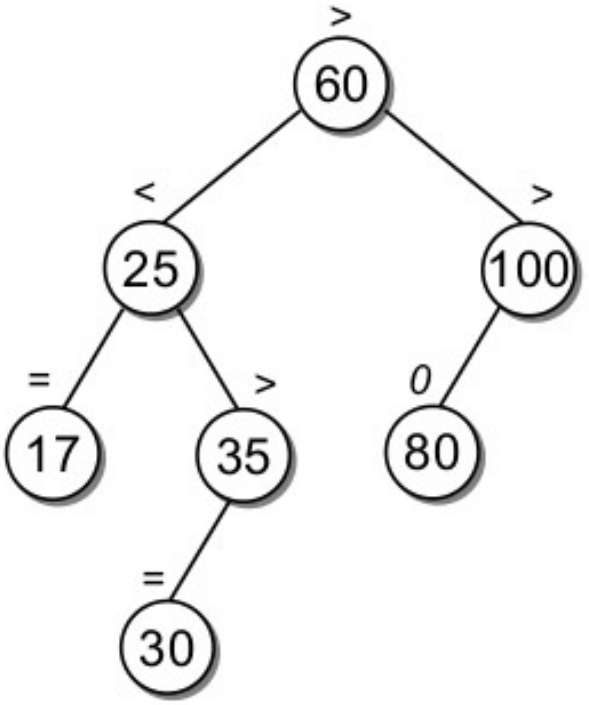
- Uma ABB é AVL se as alturas das subárvores a esquerda e a direita para cada um dos nós variem apenas de 1.

AVL

- Uma ABB é AVL se as alturas das subárvores a esquerda e a direita para cada um dos nós variem apenas de 1.
- Exemplos de árvore AVL:

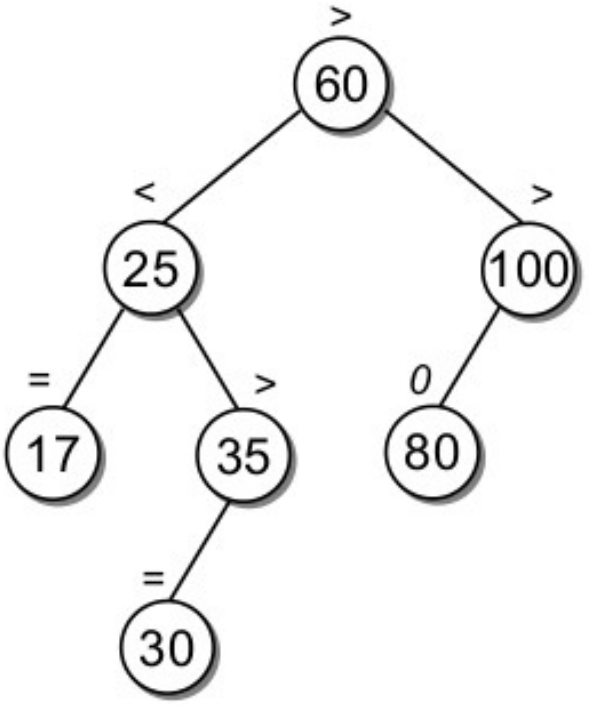
AVL

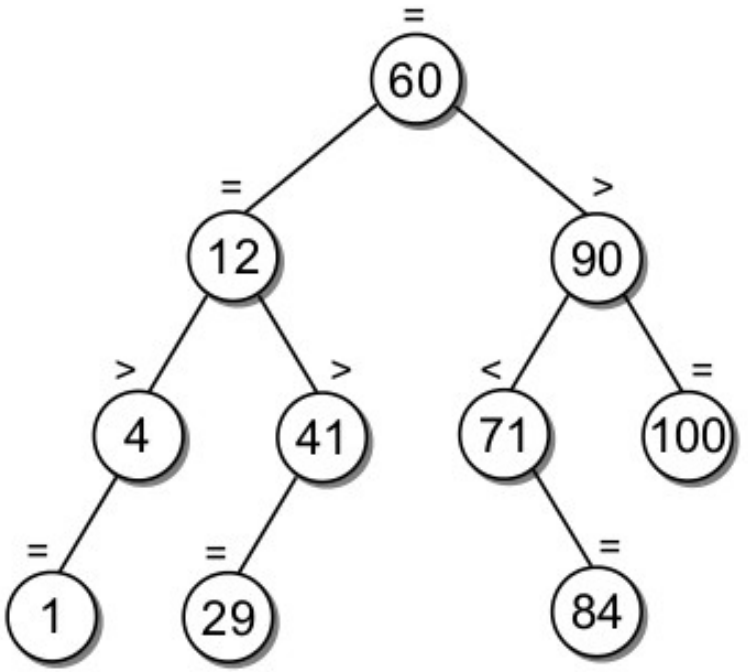
- Uma ABB é AVL se as alturas das subárvores a esquerda e a direita para cada um dos nós variem apenas de 1.
- Exemplos de árvore AVL:



AVL

- Uma ABB é AVL se as alturas das subárvores a esquerda e a direita para cada um dos nós variem apenas de 1.
- Exemplos de árvore AVL:



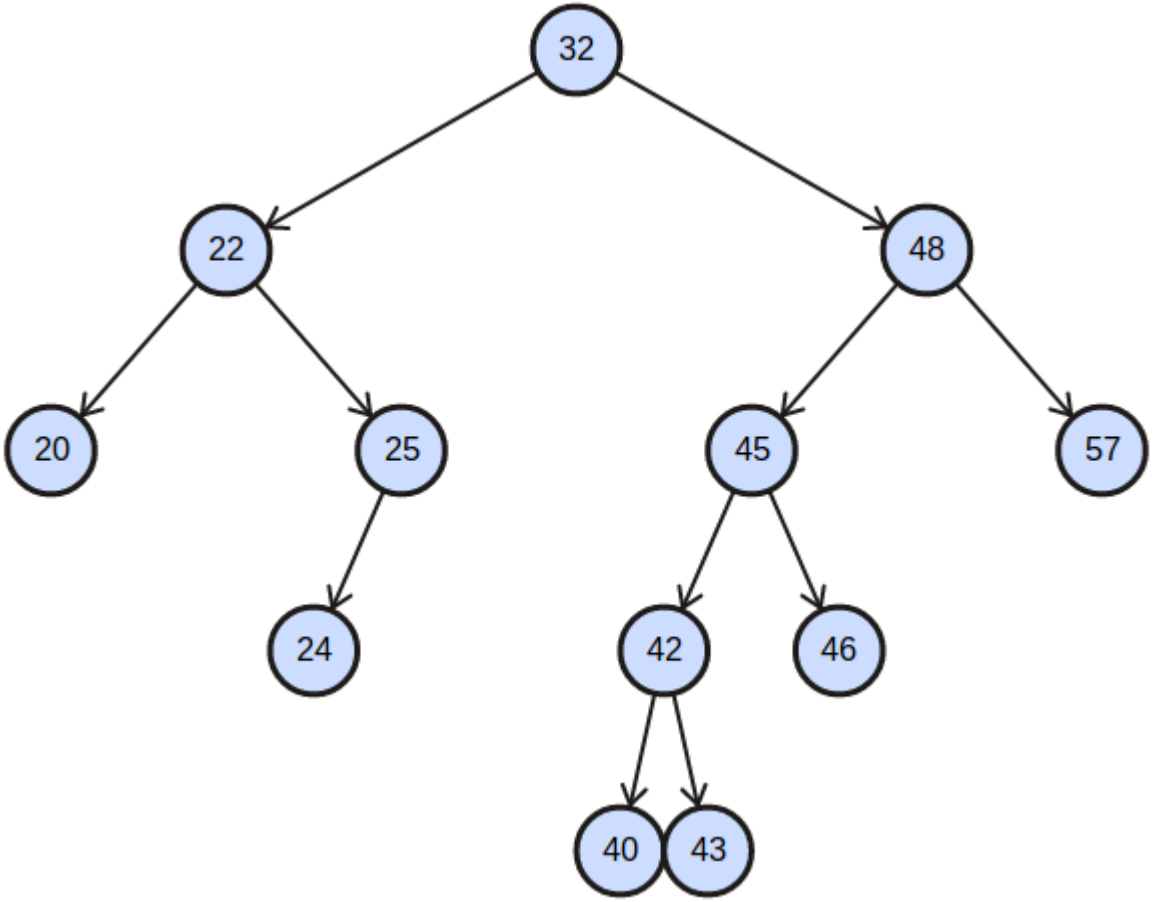


AVL

- Exemplo de árvore que **não** é AVL:

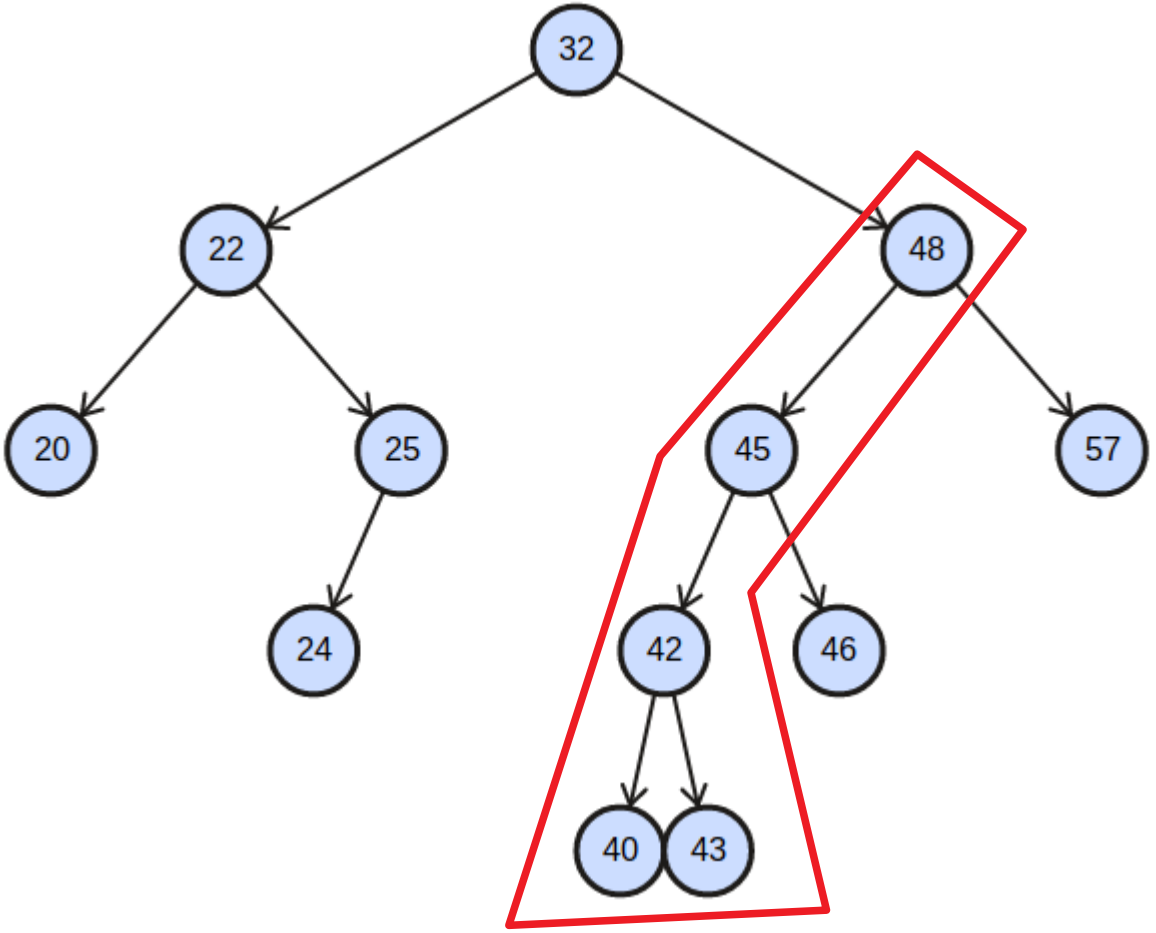
AVL

- Exemplo de árvore que **não** é AVL:



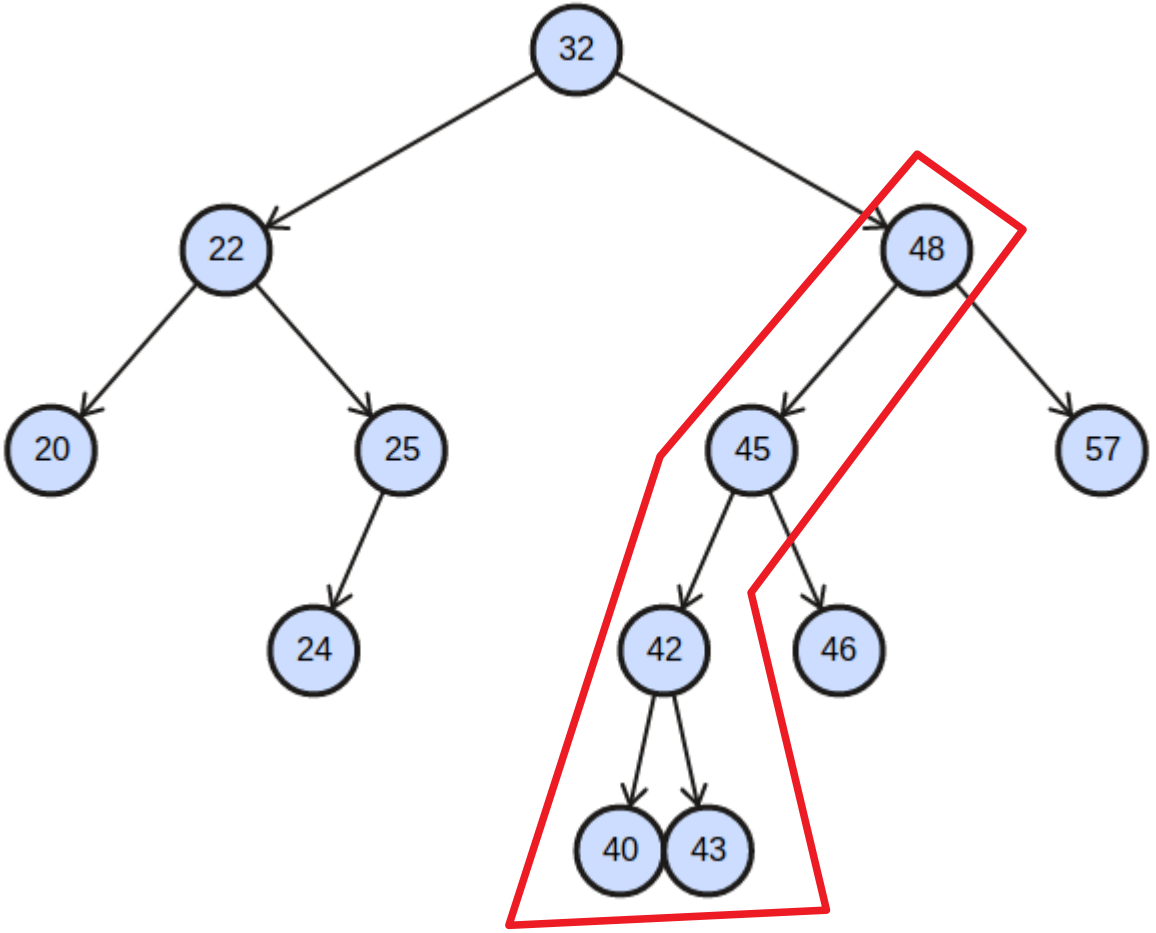
AVL

- Exemplo de árvore que **não** é AVL:



AVL

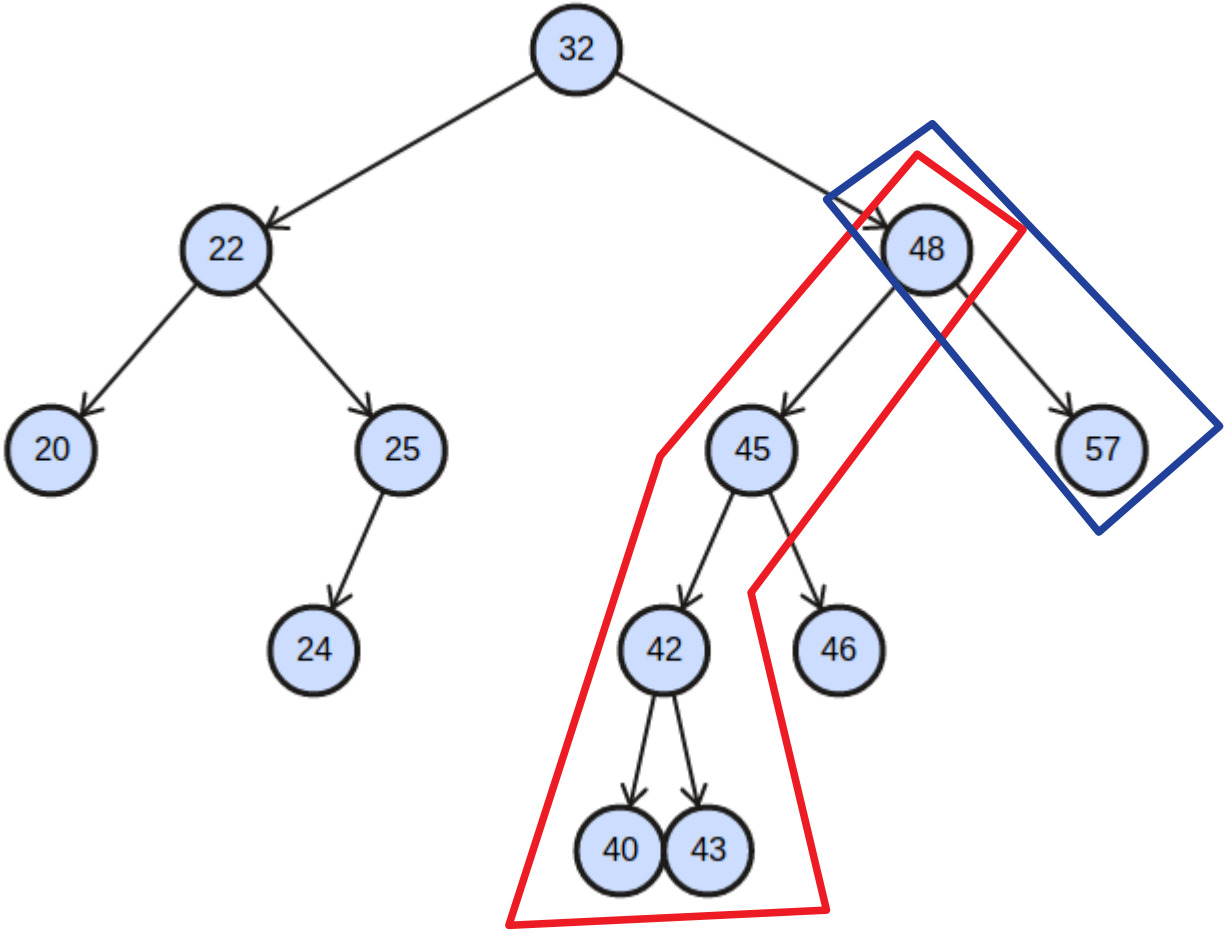
- Exemplo de árvore que **não** é AVL:



ALTURA = 3

AVL

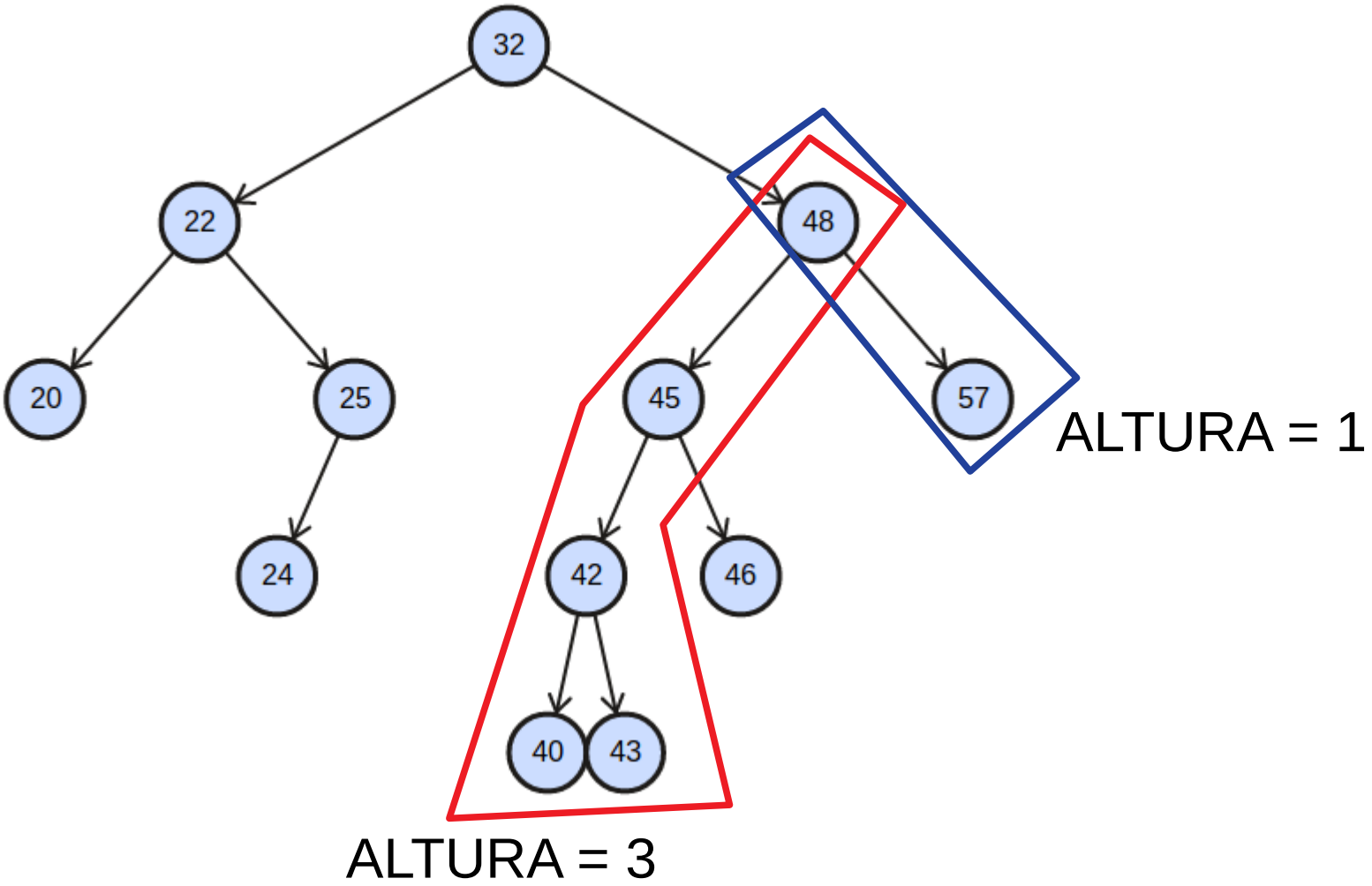
- Exemplo de árvore que **não** é AVL:



ALTURA = 3

AVL

- Exemplo de árvore que **não** é AVL:



AVL

AVL

- As operações de busca e de travessia são iguais nas árvores binárias de busca e nas AVL.

AVL

- As operações de busca e de travessia são iguais nas árvores binárias de busca e nas AVL.
- As operações de inserção e remoção devem ser modificadas para manter a propriedade de balanceamento das AVL.

Inserção

Inserção

- A inserção na AVL inicia com o mesmo procedimento usado nas ABB.

Inserção

- A inserção na AVL inicia com o mesmo procedimento usado nas ABB.
- Nós pesquisamos por uma chave na árvore e adicionamos o novo elemento onde ocorre a “falha” na busca.

Inserção

- A inserção na AVL inicia com o mesmo procedimento usado nas ABB.
- Nós pesquisamos por uma chave na árvore e adicionamos o novo elemento onde ocorre a “falha” na busca.
- Quando um novo elemento é inserido na AVL, a propriedade de balanceamento deve ser mantida.

Inserção

- A inserção na AVL inicia com o mesmo procedimento usado nas ABB.
- Nós pesquisamos por uma chave na árvore e adicionamos o novo elemento onde ocorre a “falha” na busca.
- Quando um novo elemento é inserido na AVL, a propriedade de balanceamento deve ser mantida.
- Se a inserção de um novo elemento causar um desbalanceamento, então a árvore deverá ser rebalanceada.

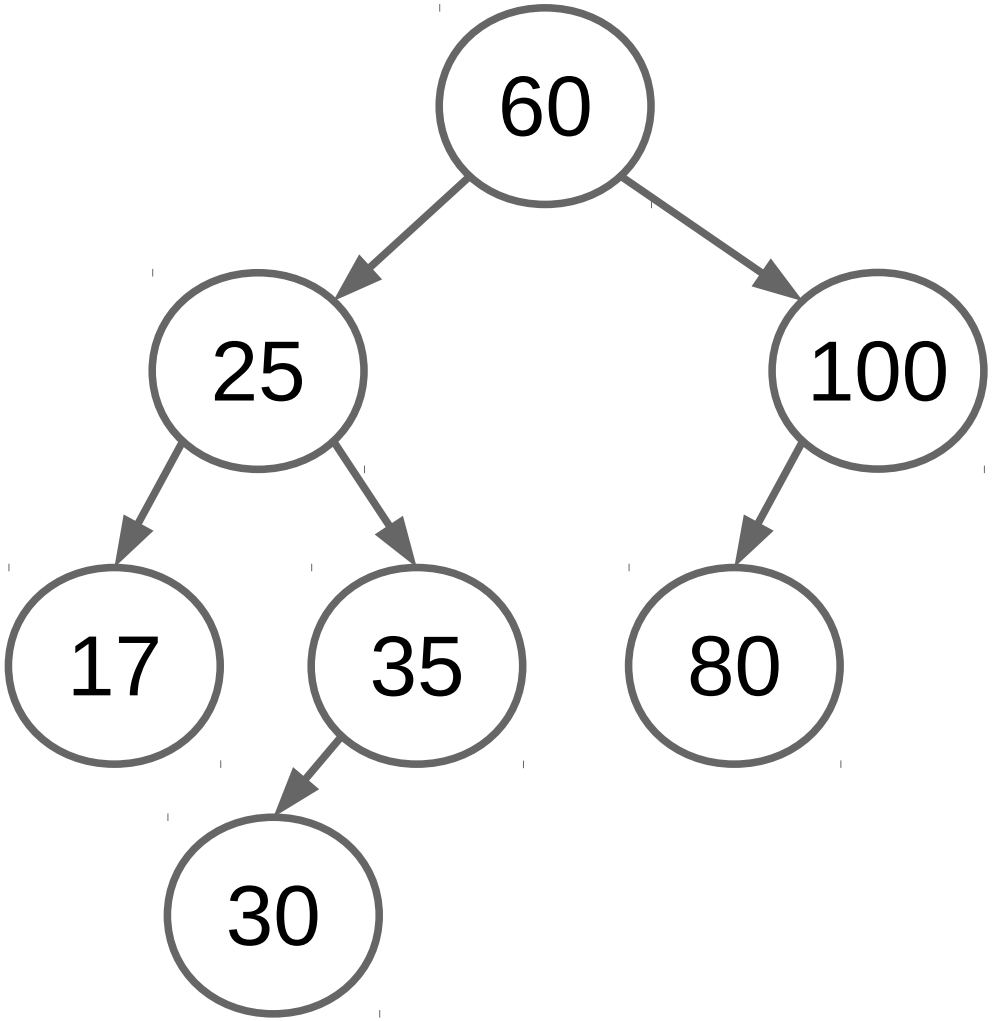
Inserção

Inserção

- Considere que vamos inserir o elemento 120 na árvore abaixo:

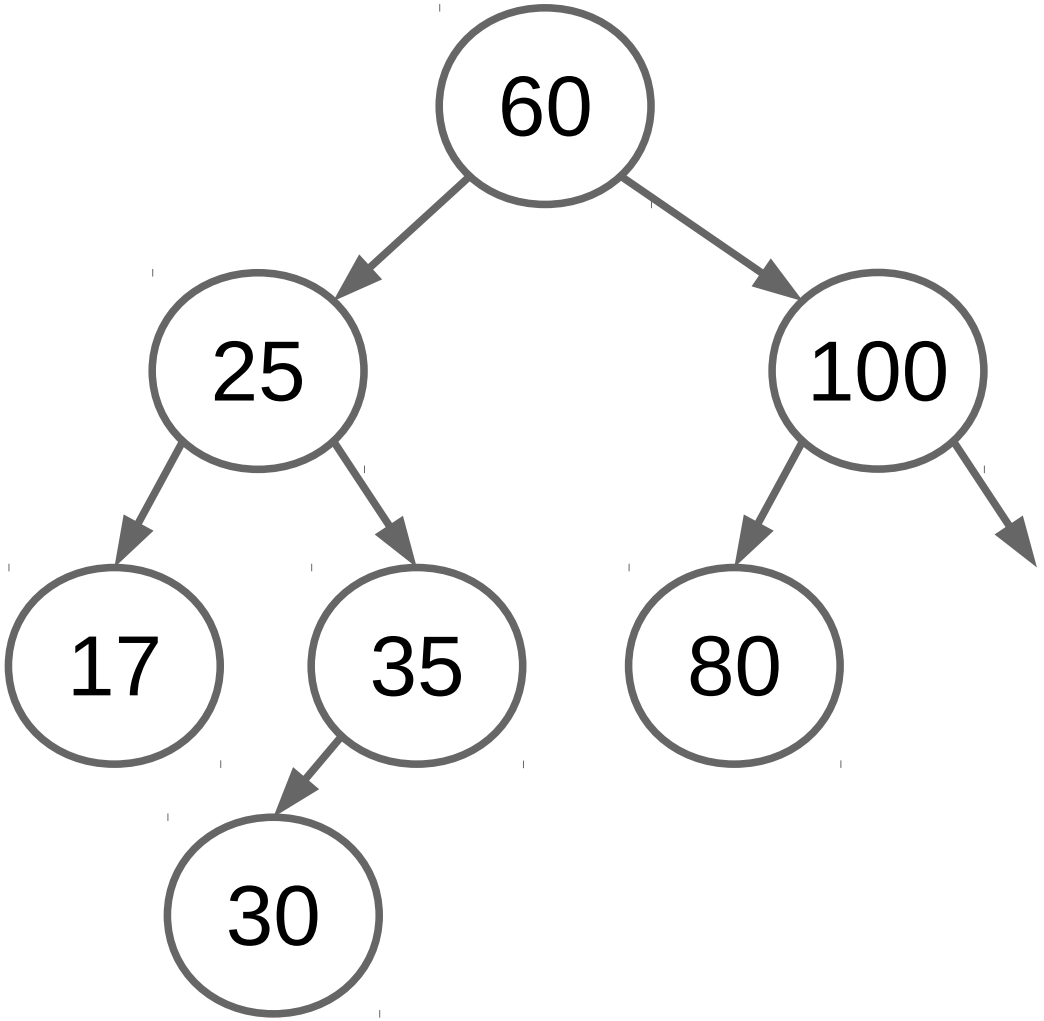
Inserção

- Considere que vamos inserir o elemento 120 na árvore abaixo:



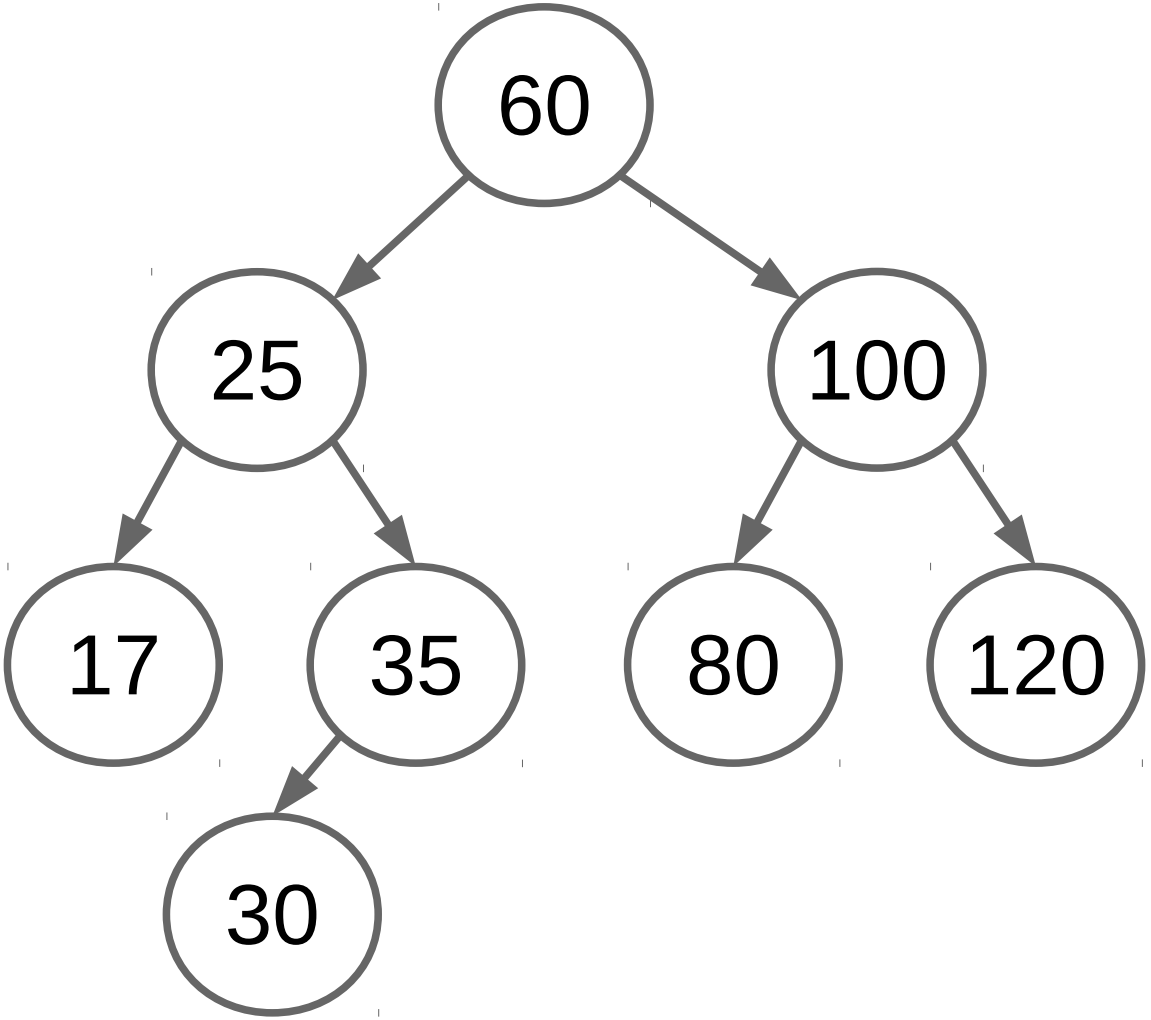
Inserção

- Considere que vamos inserir o elemento 120 na árvore abaixo:



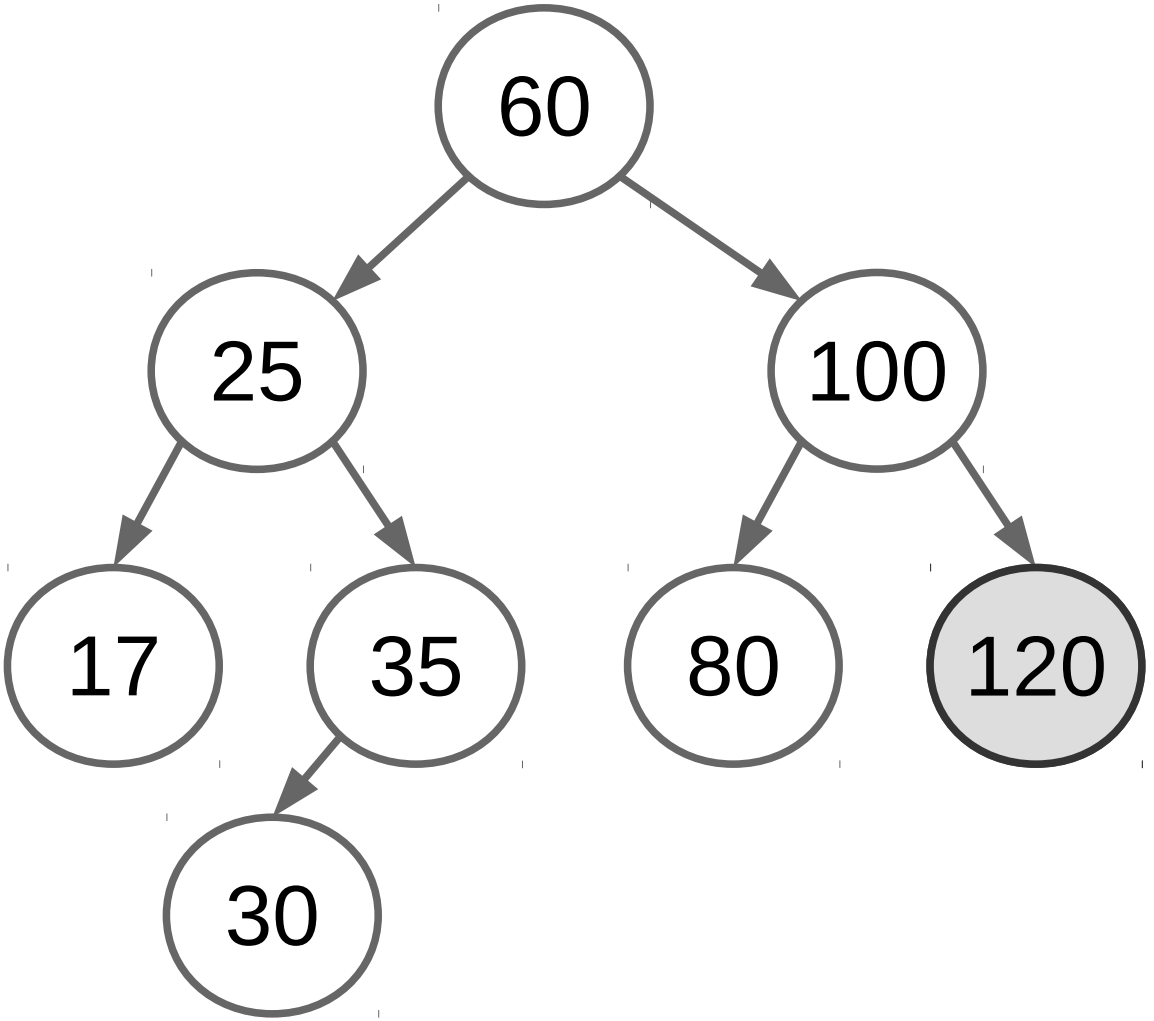
Inserção

- Considere que vamos inserir o elemento 120 na árvore abaixo:



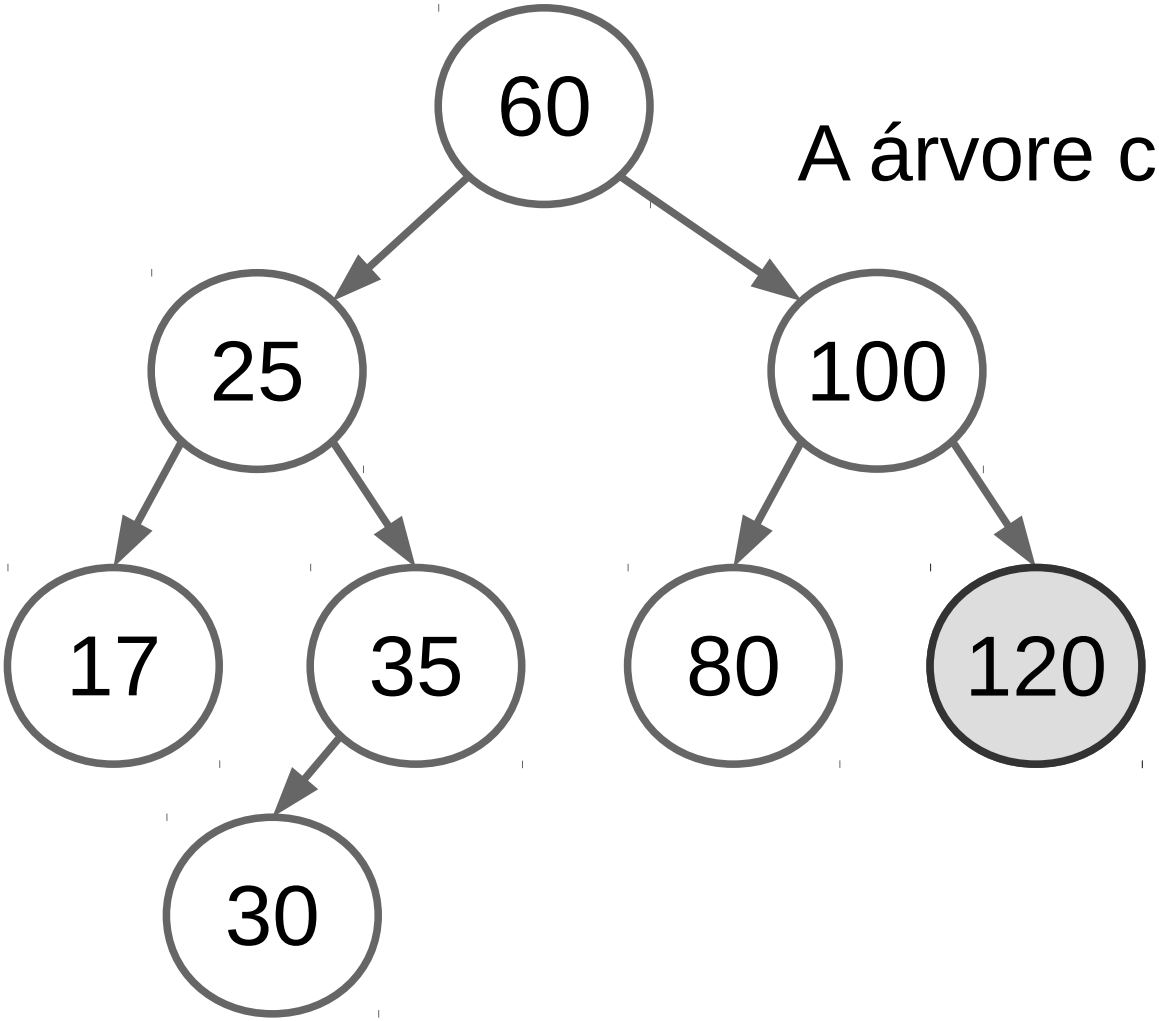
Inserção

- Considere que vamos inserir o elemento 120 na árvore abaixo:



Inserção

- Considere que vamos inserir o elemento 120 na árvore abaixo:



A árvore continua balanceada?

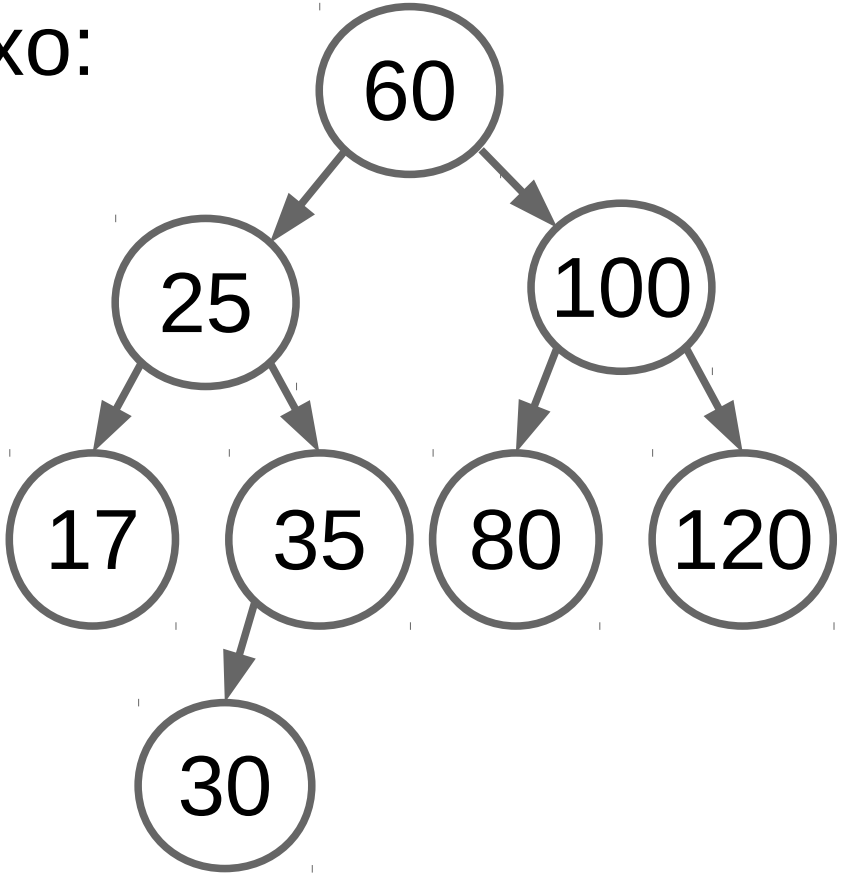
Inserção

Inserção

- Considere que vamos inserir o elemento 28 na árvore abaixo:

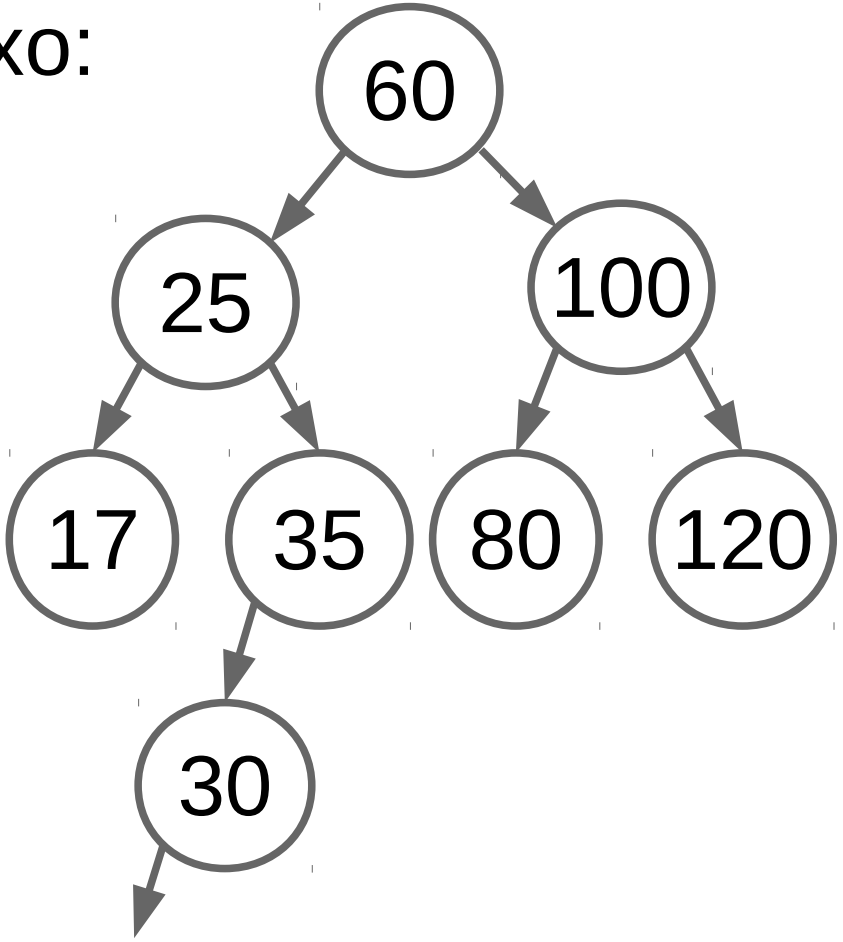
Inserção

- Considere que vamos inserir o elemento 28 na árvore abaixo:



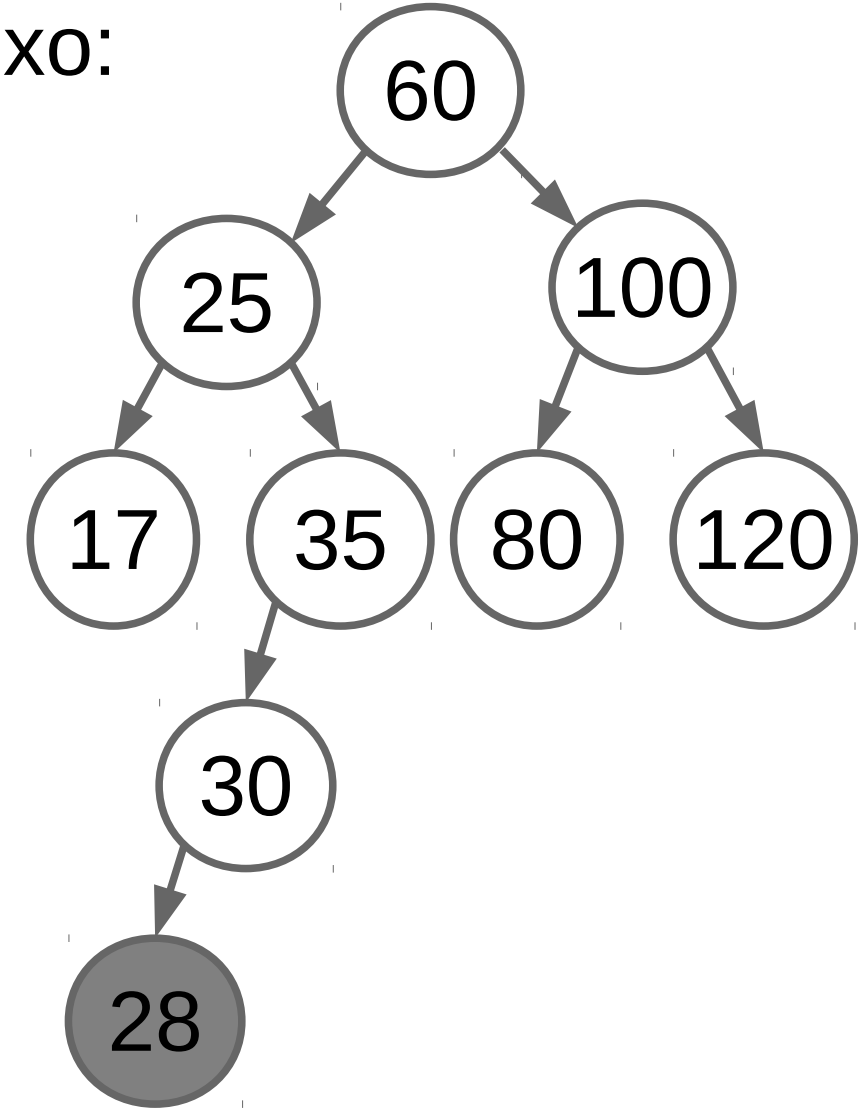
Inserção

- Considere que vamos inserir o elemento 28 na árvore abaixo:



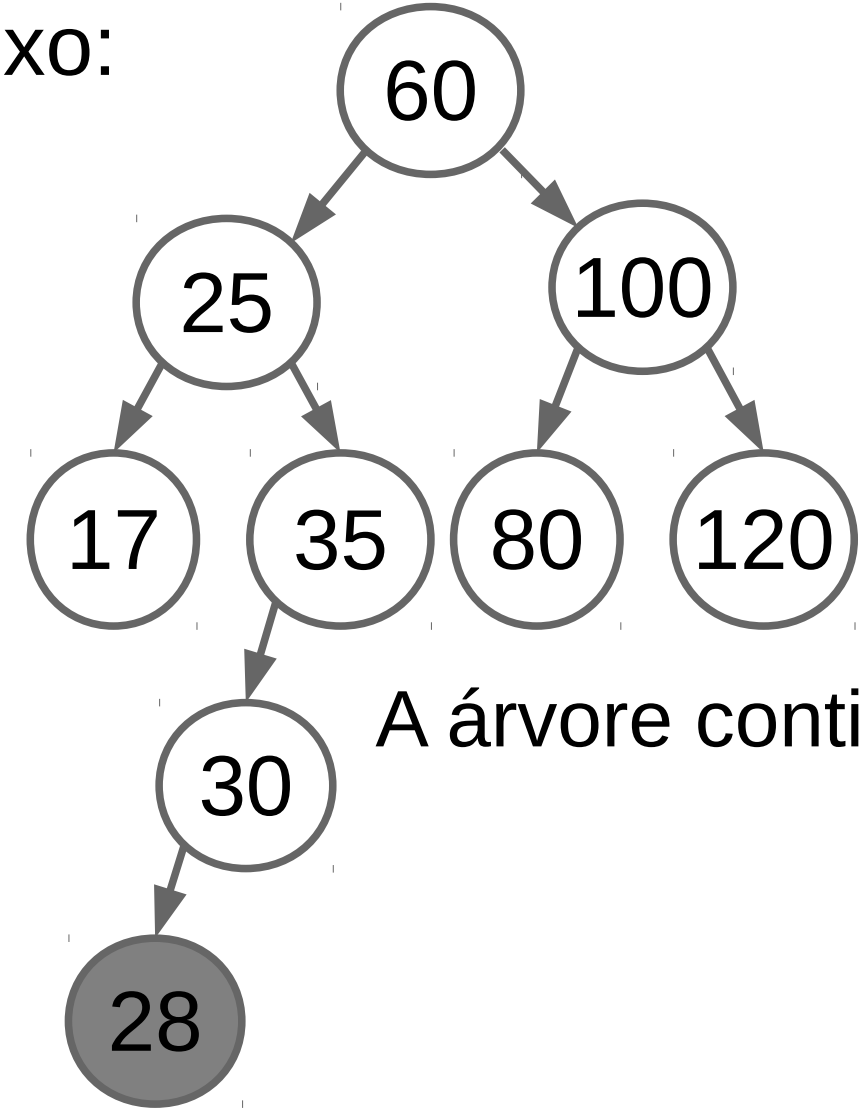
Inserção

- Considere que vamos inserir o elemento 28 na árvore abaixo:



Inserção

- Considere que vamos inserir o elemento 28 na árvore abaixo:



A árvore continua balanceada?

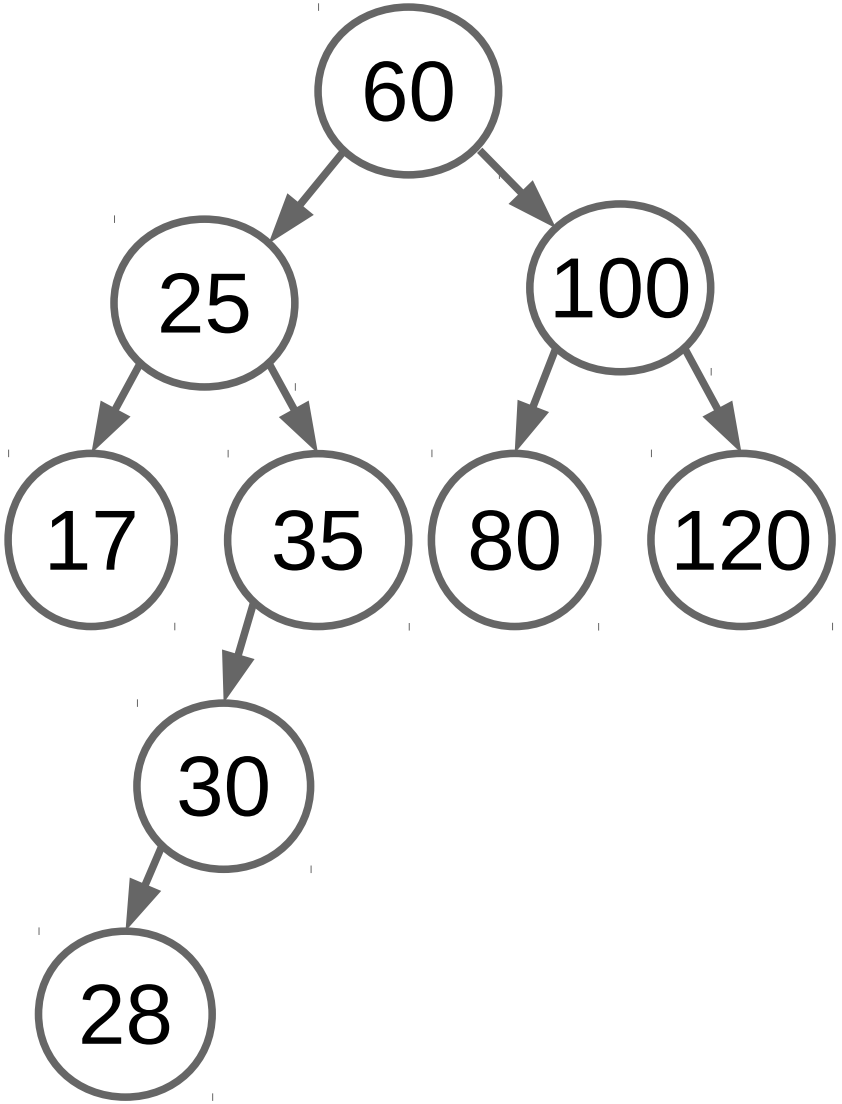
Inserção

Inserção

- Como a árvore ficaria balanceada?

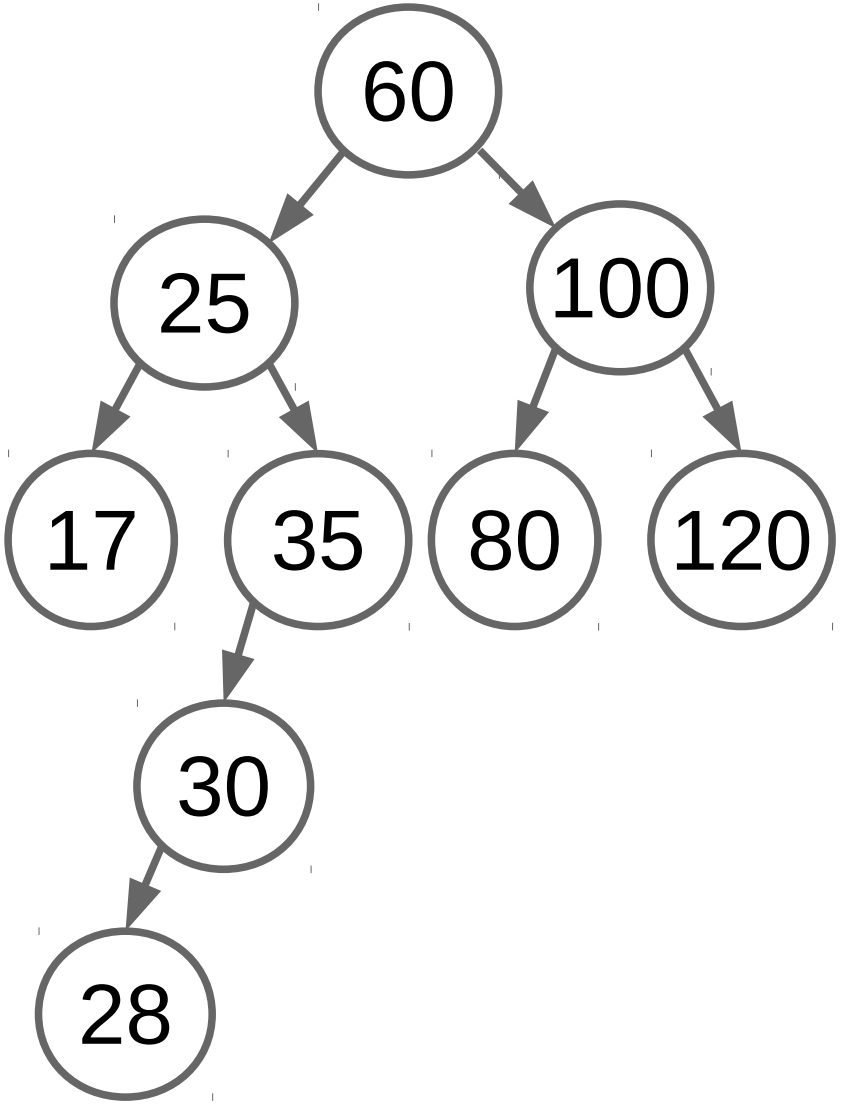
Inserção

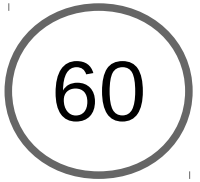
- Como a árvore ficaria balanceada?



Inserção

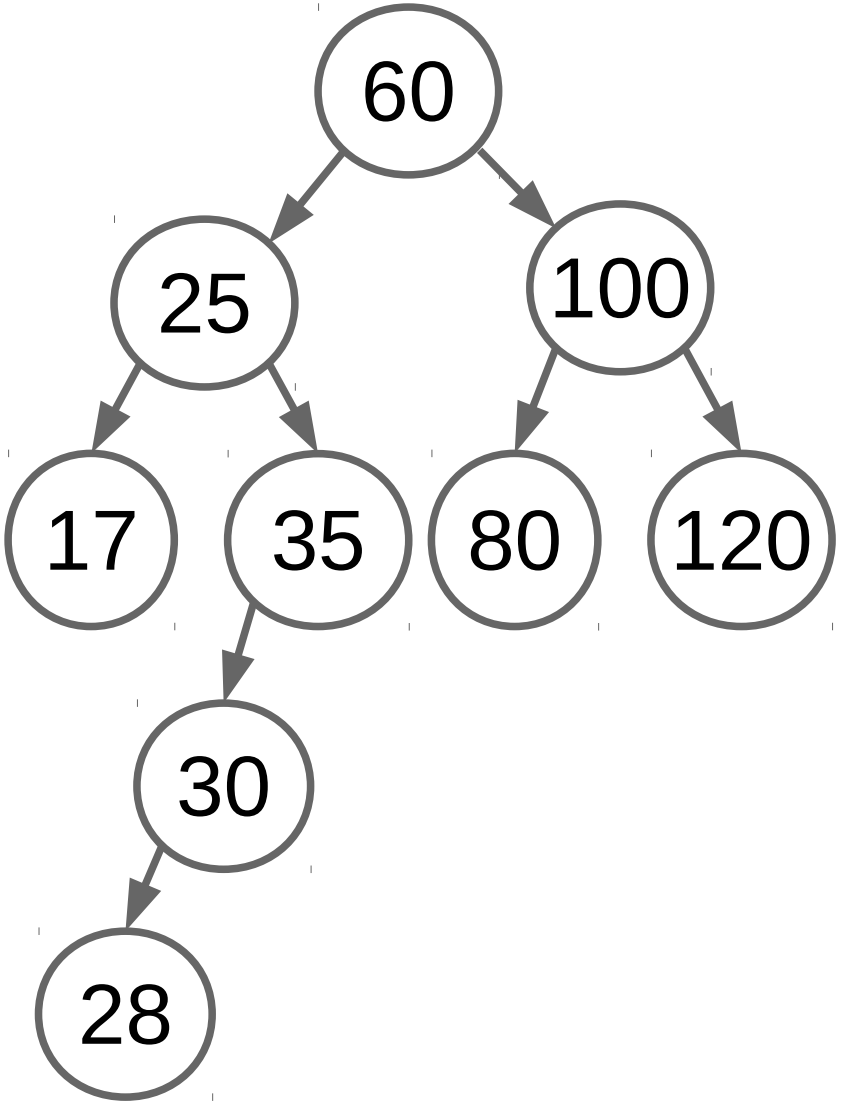
- Como a árvore ficaria balanceada?

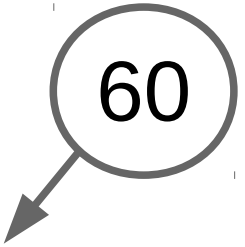




Inserção

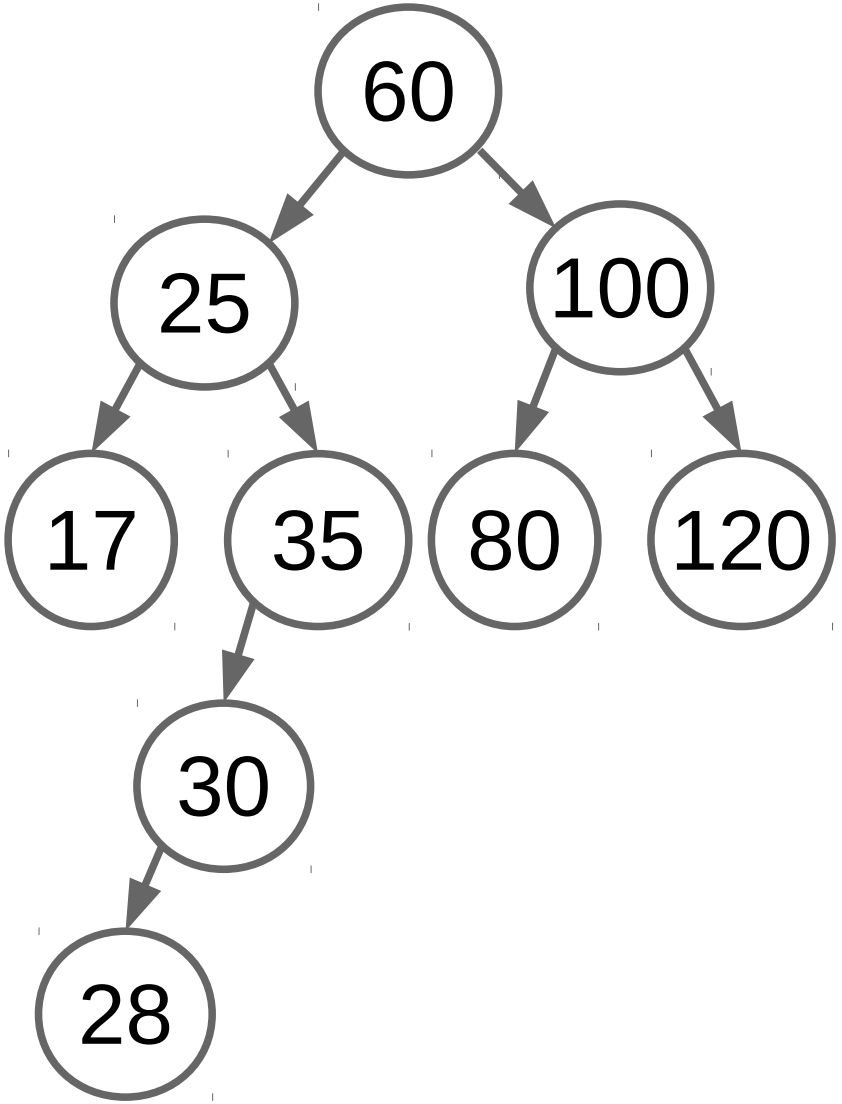
- Como a árvore ficaria balanceada?

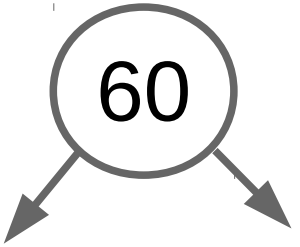




Inserção

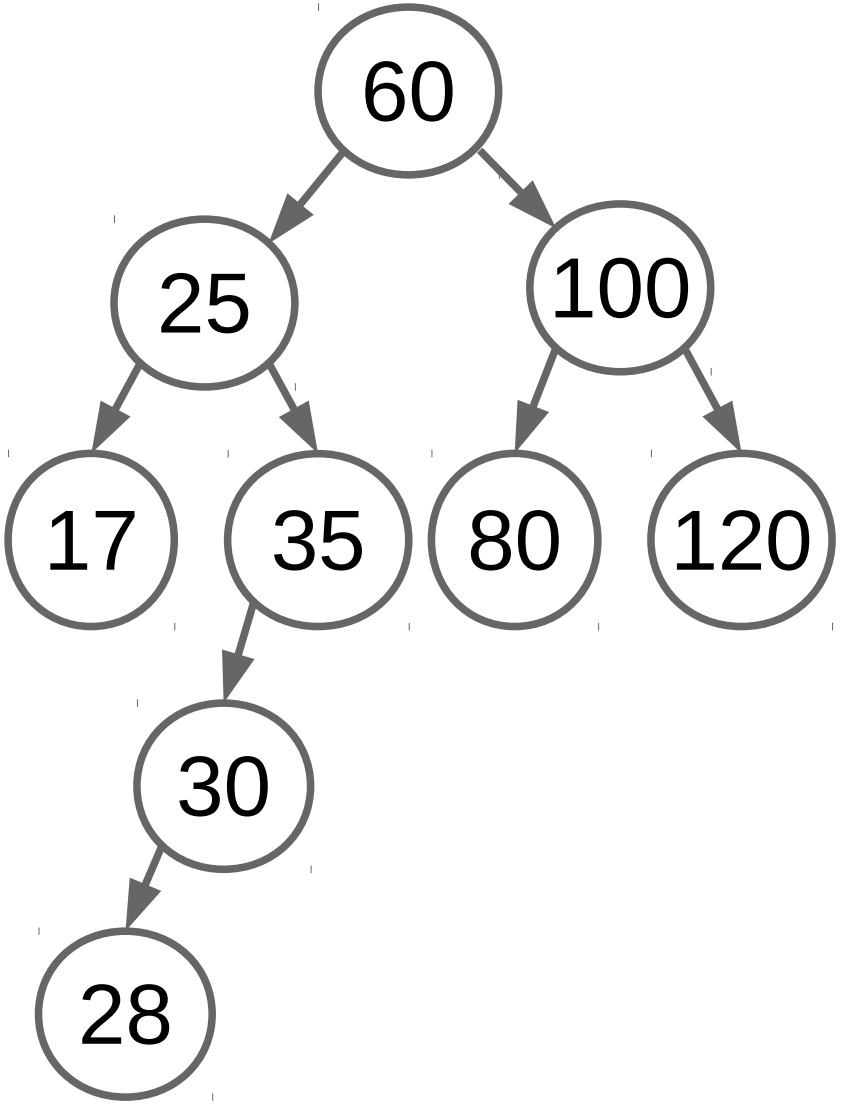
- Como a árvore ficaria balanceada?

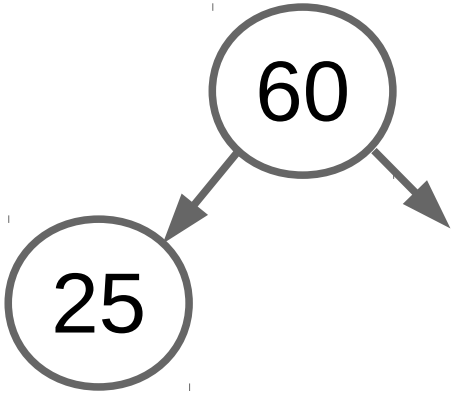




Inserção

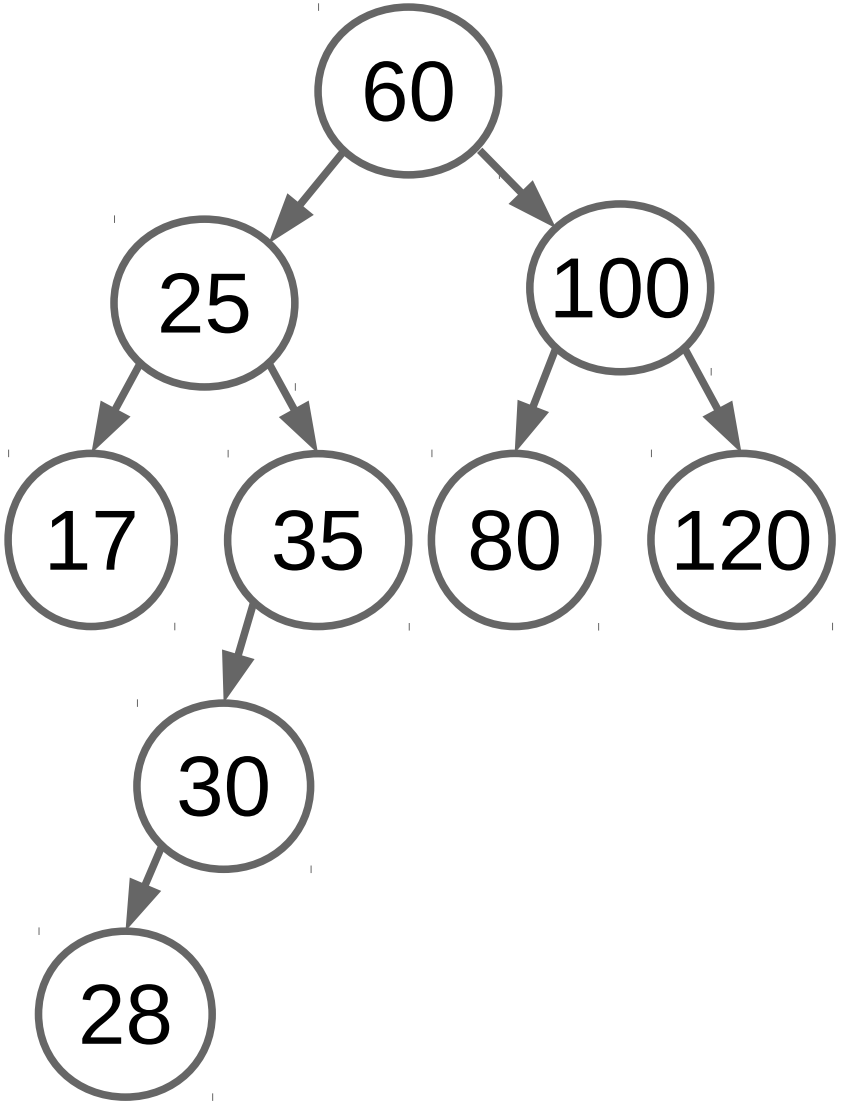
- Como a árvore ficaria balanceada?

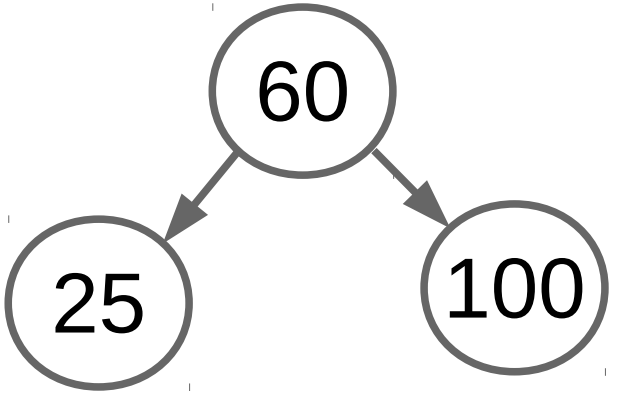




Inserção

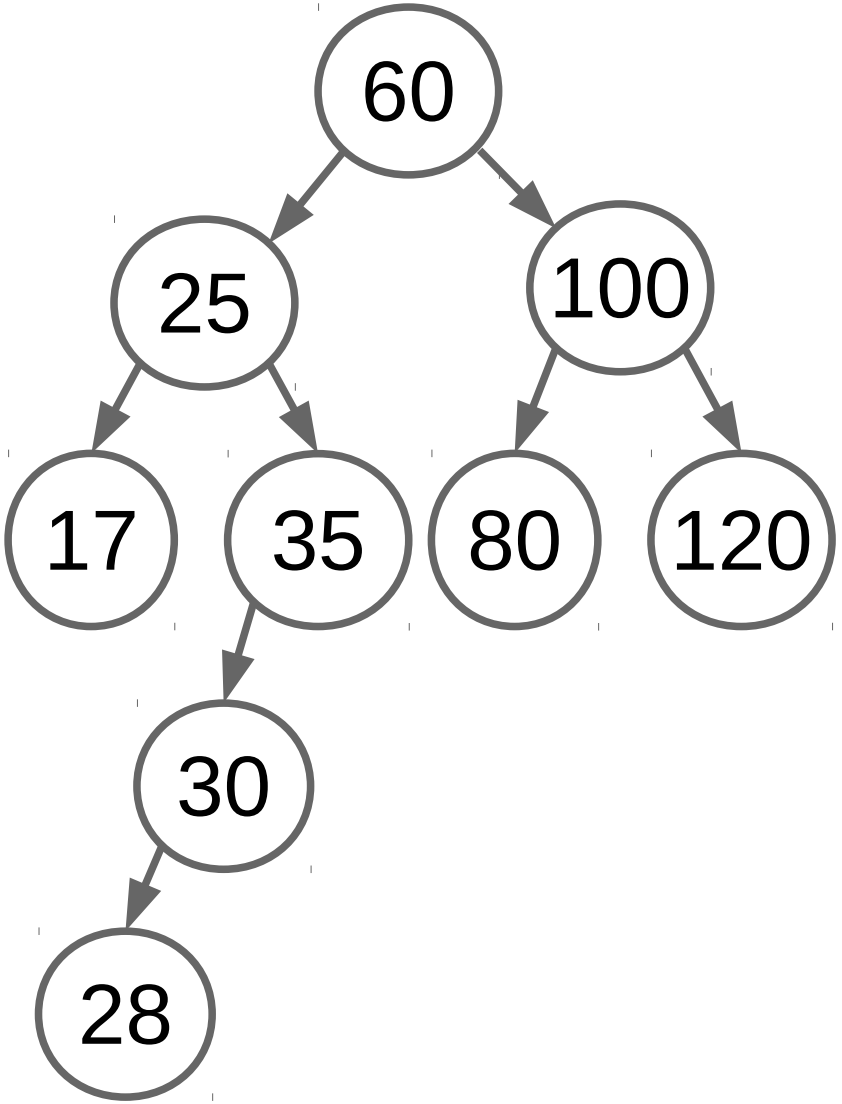
- Como a árvore ficaria balanceada?

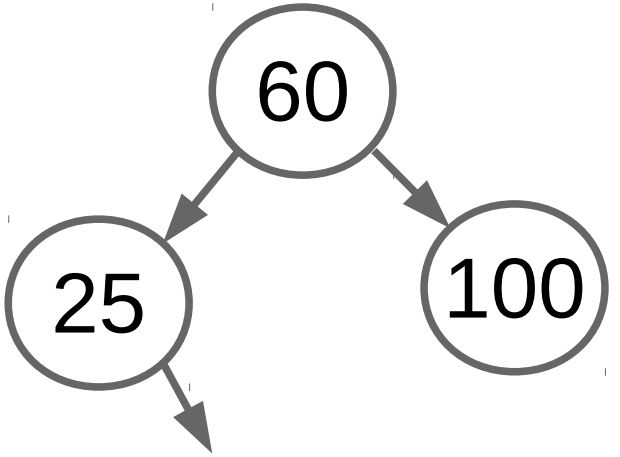




Inserção

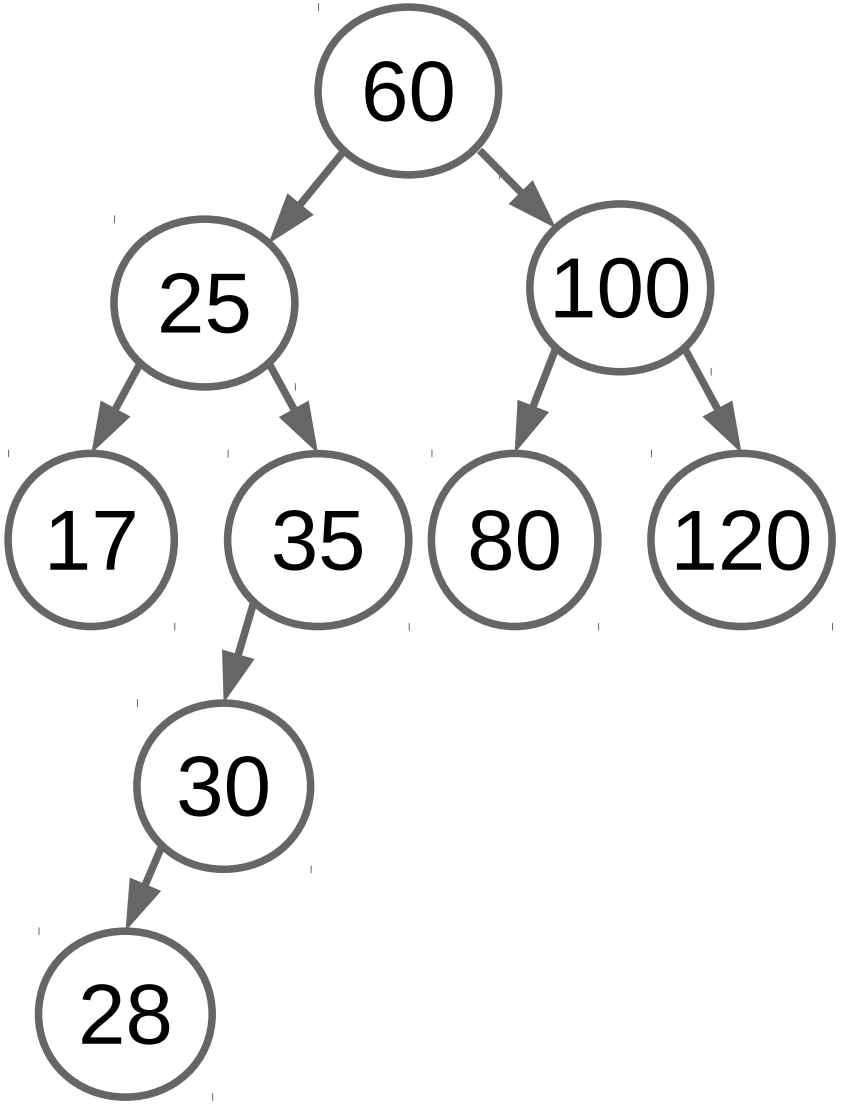
- Como a árvore ficaria balanceada?

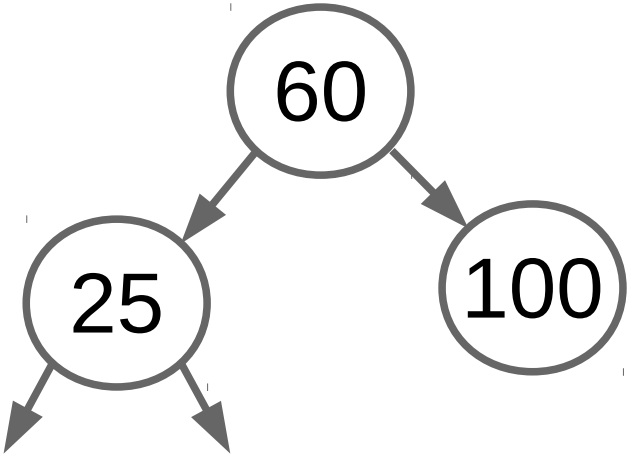




Inserção

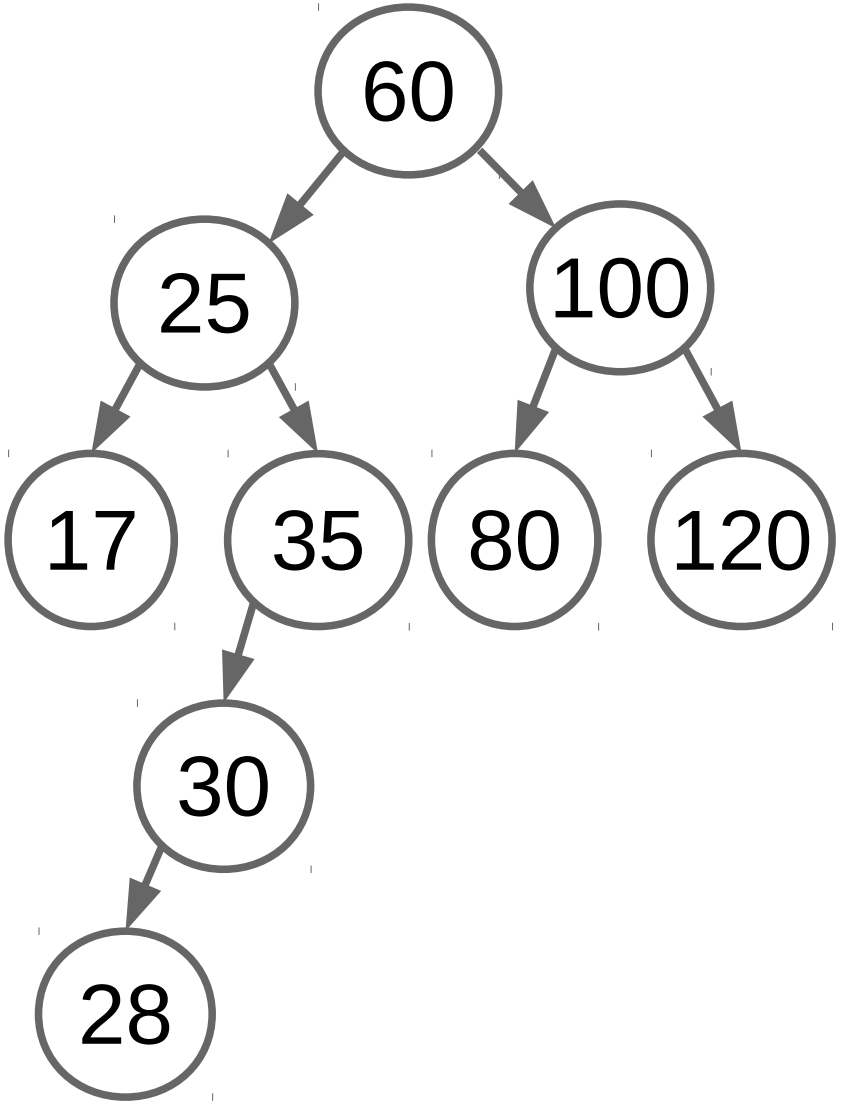
- Como a árvore ficaria balanceada?

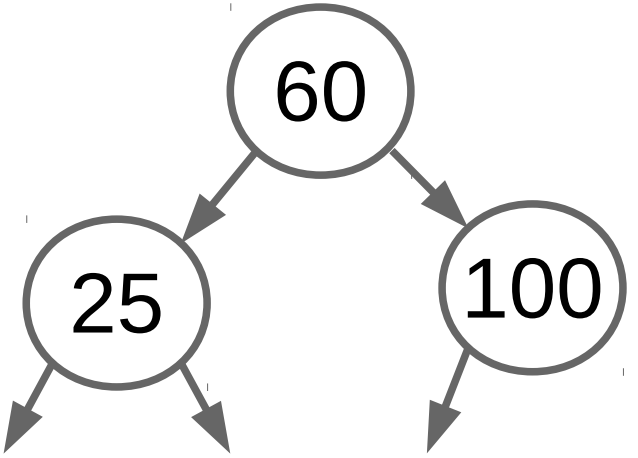




Inserção

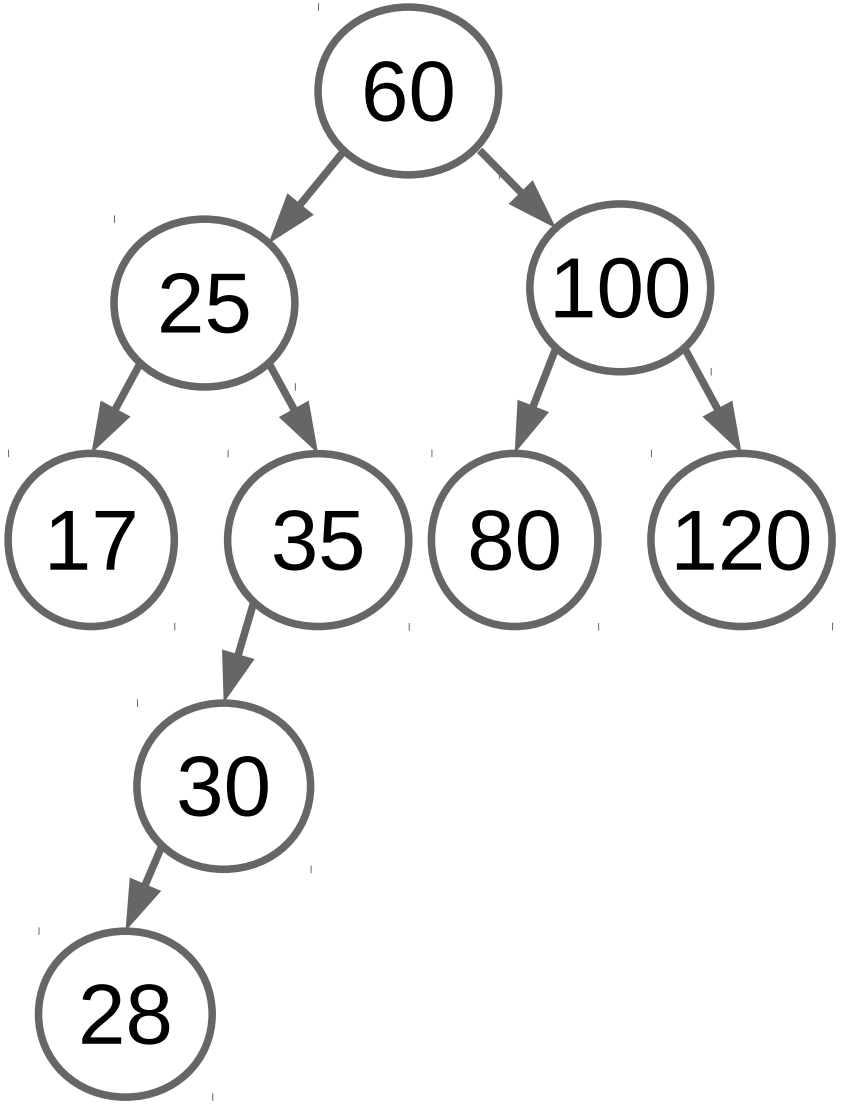
- Como a árvore ficaria balanceada?

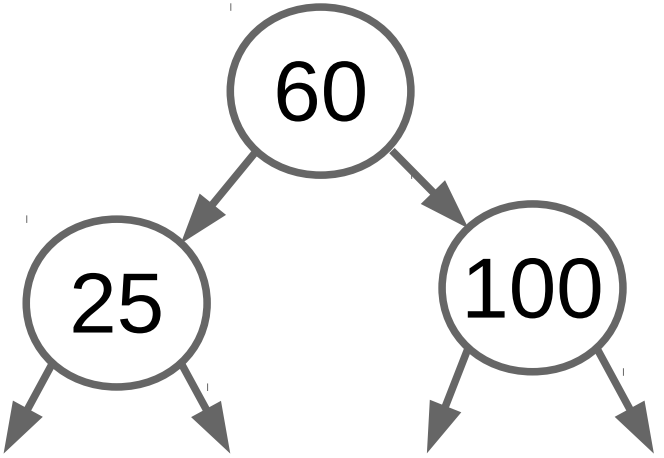




Inserção

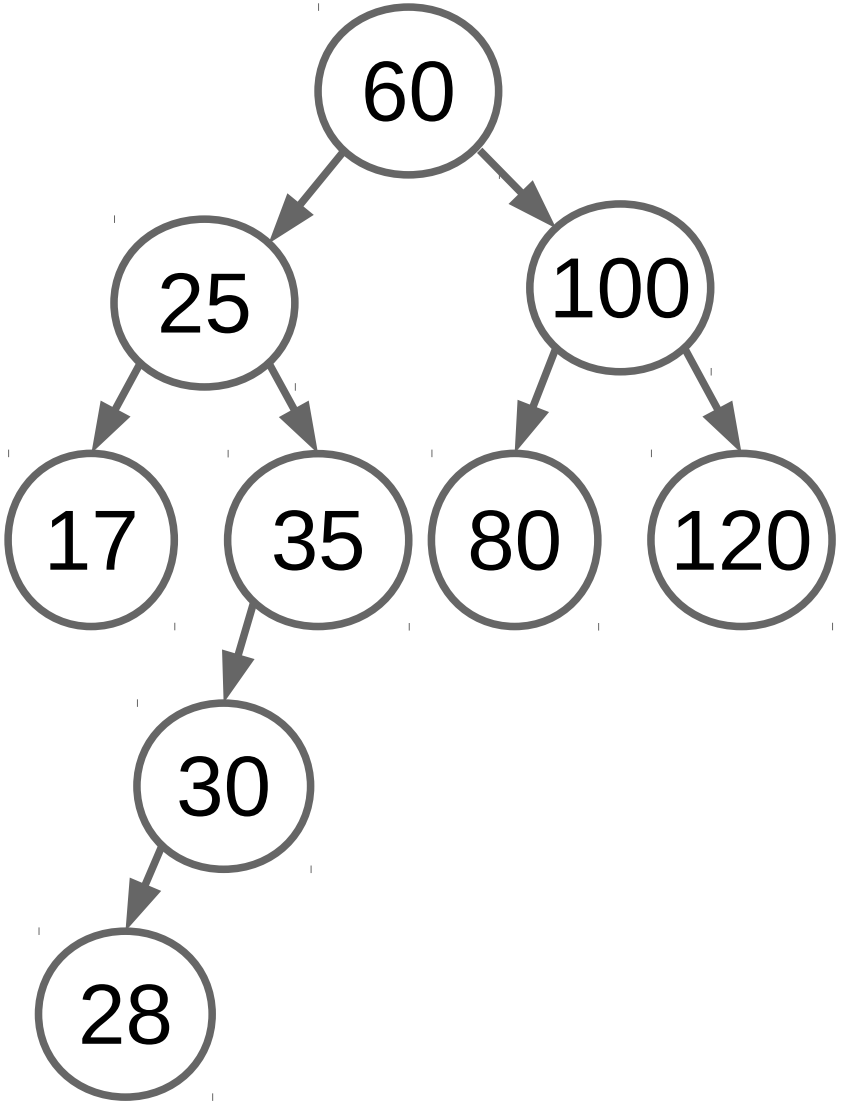
- Como a árvore ficaria balanceada?

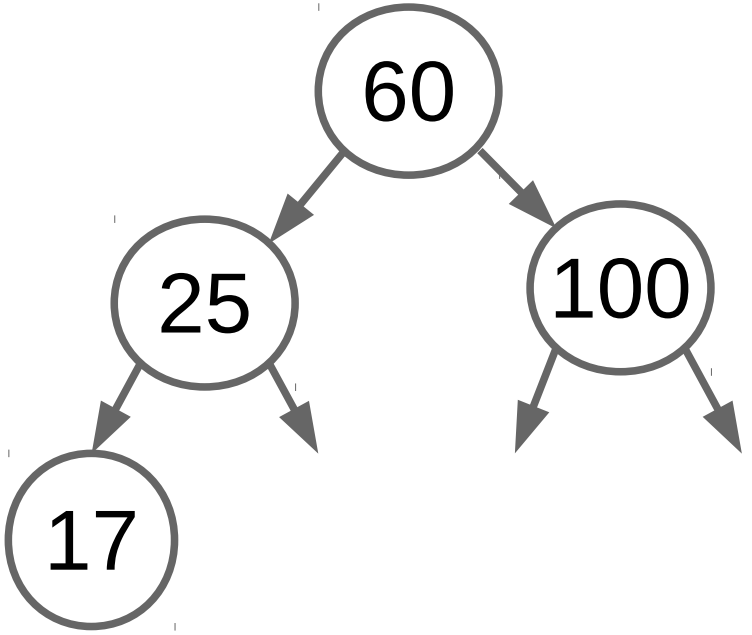




Inserção

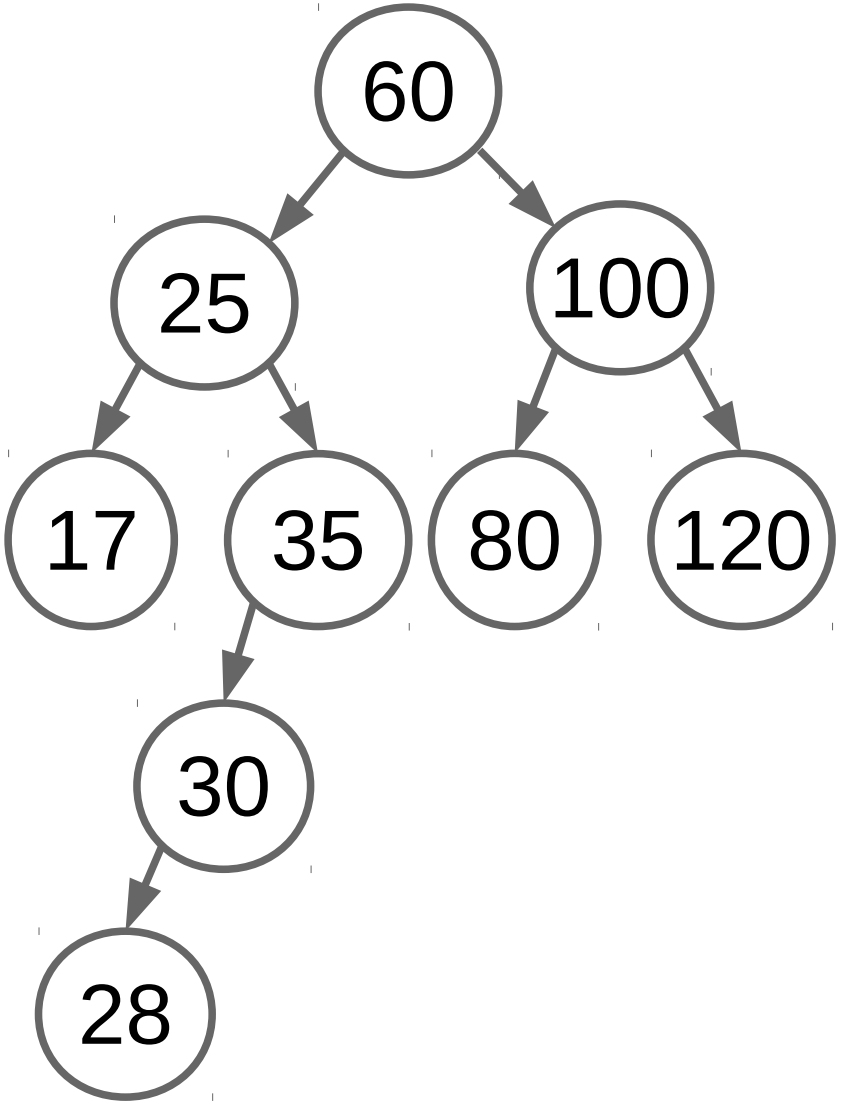
- Como a árvore ficaria balanceada?

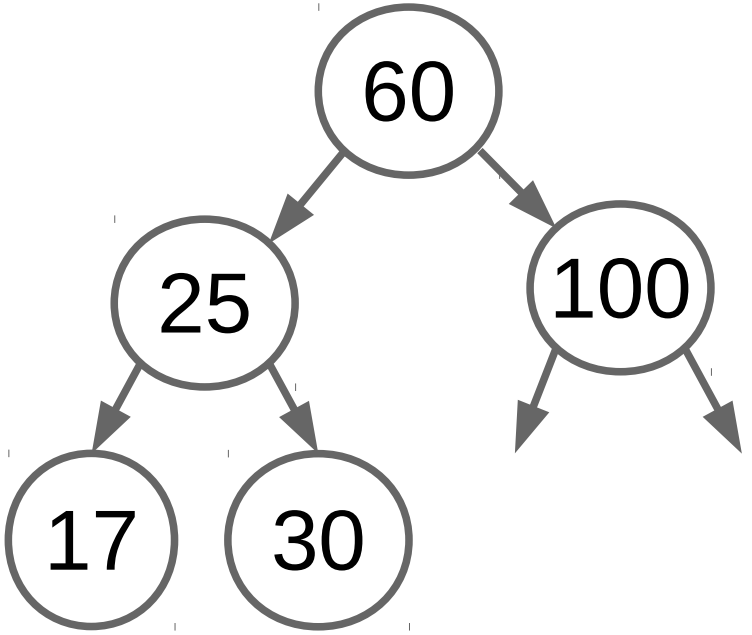




Inserção

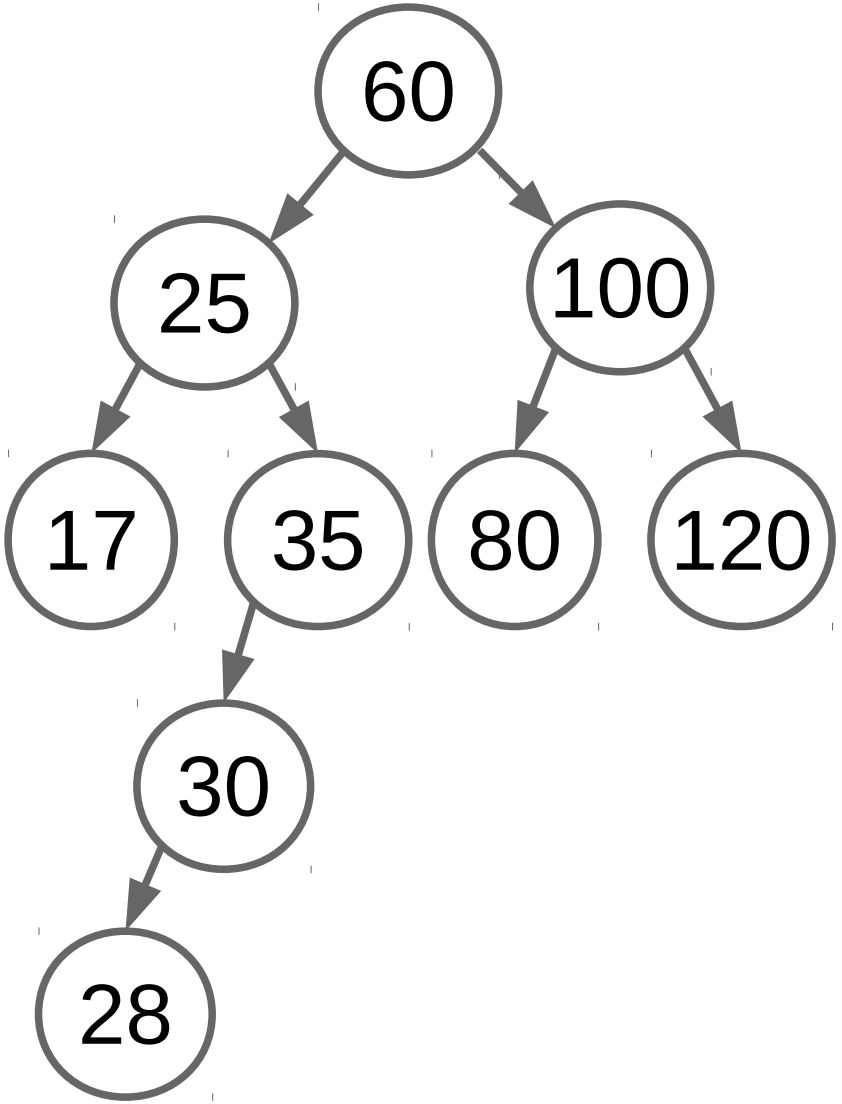
- Como a árvore ficaria balanceada?

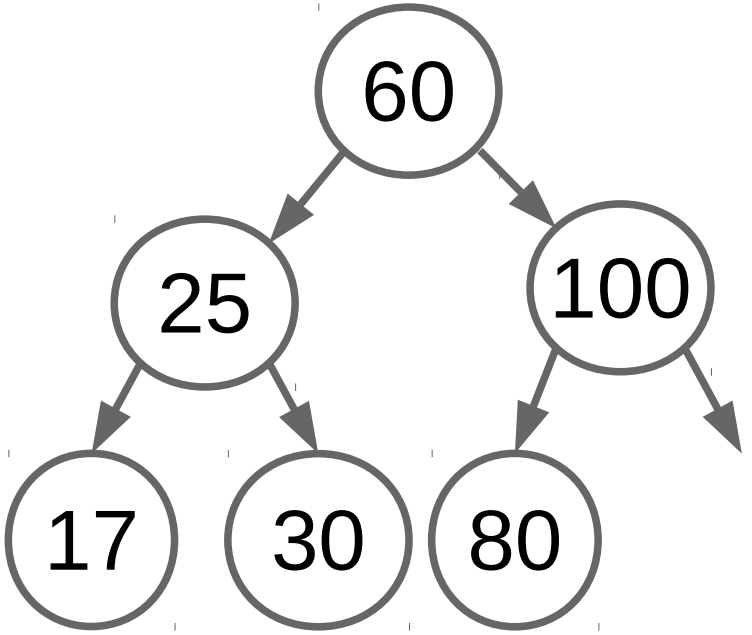




Inserção

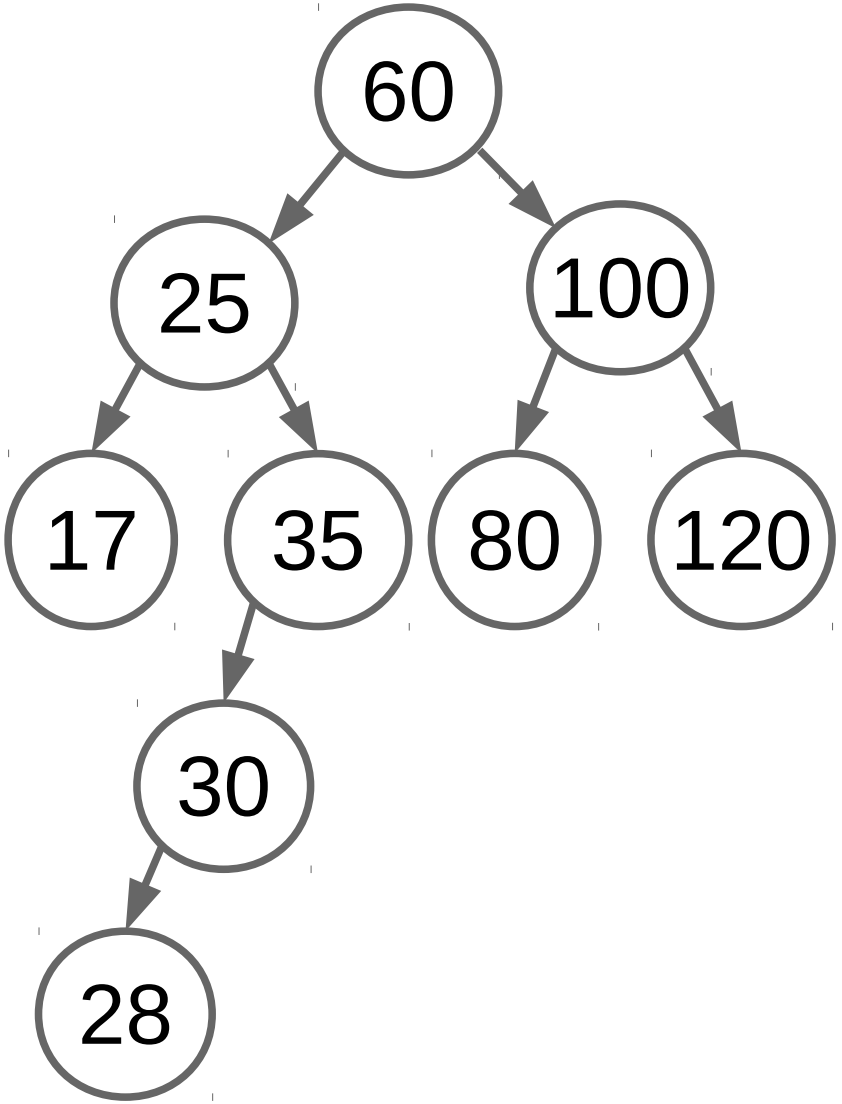
- Como a árvore ficaria balanceada?

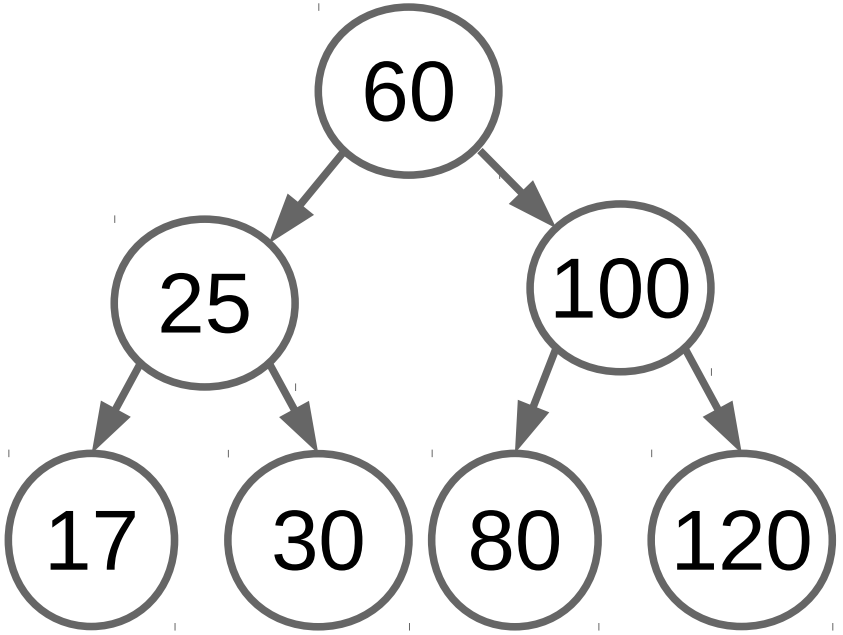




Inserção

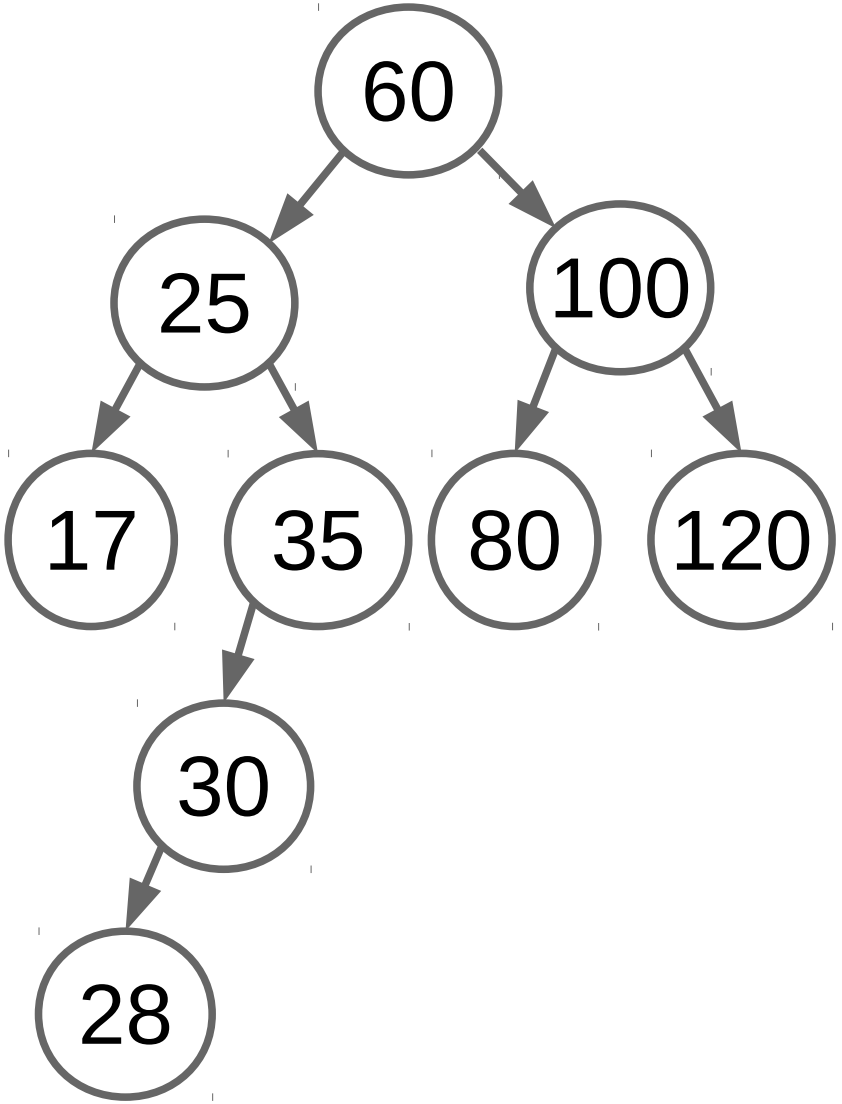
- Como a árvore ficaria balanceada?

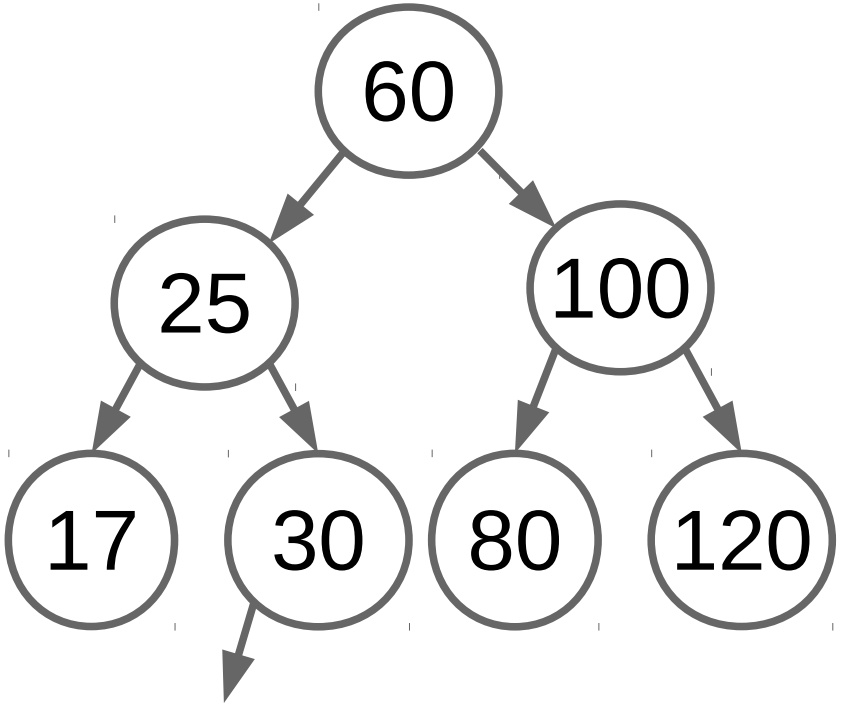




Inserção

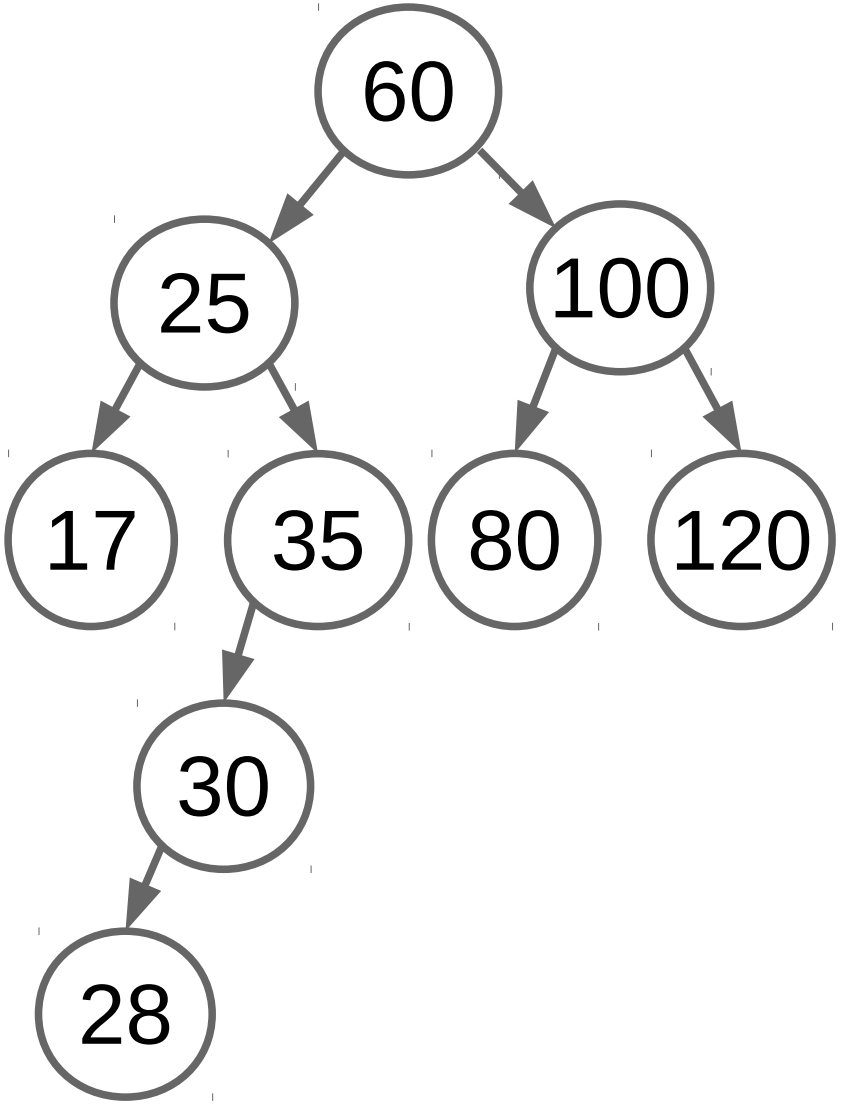
- Como a árvore ficaria balanceada?

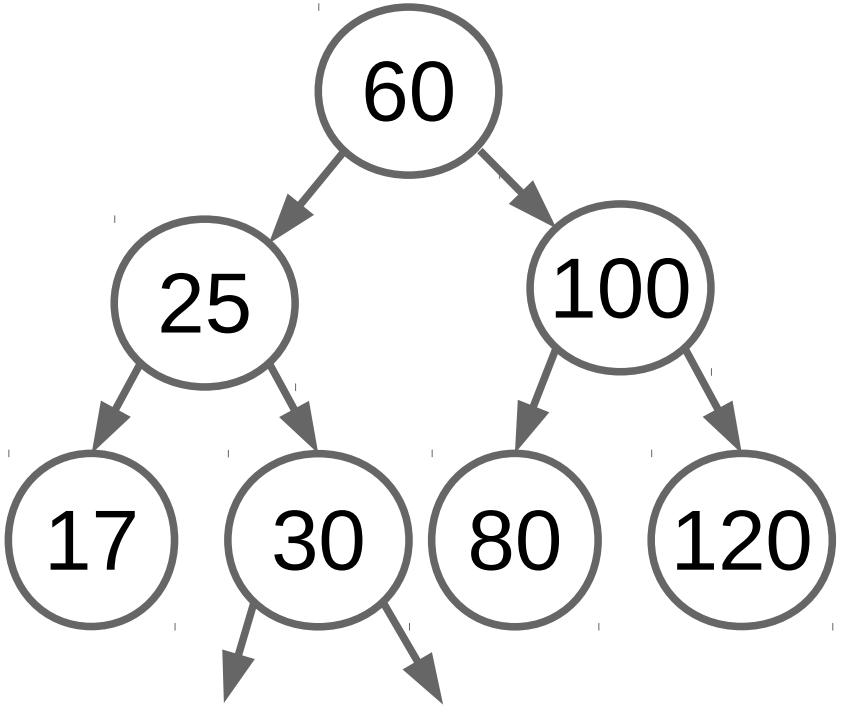




Inserção

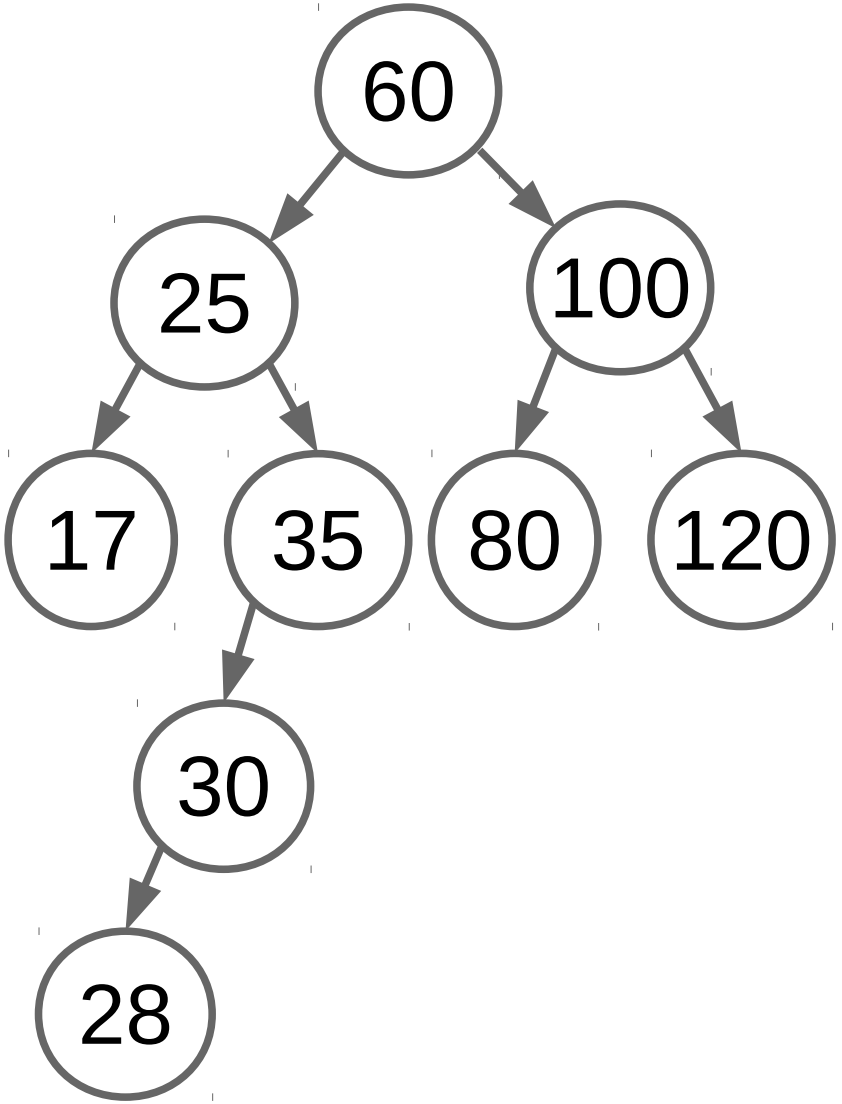
- Como a árvore ficaria balanceada?

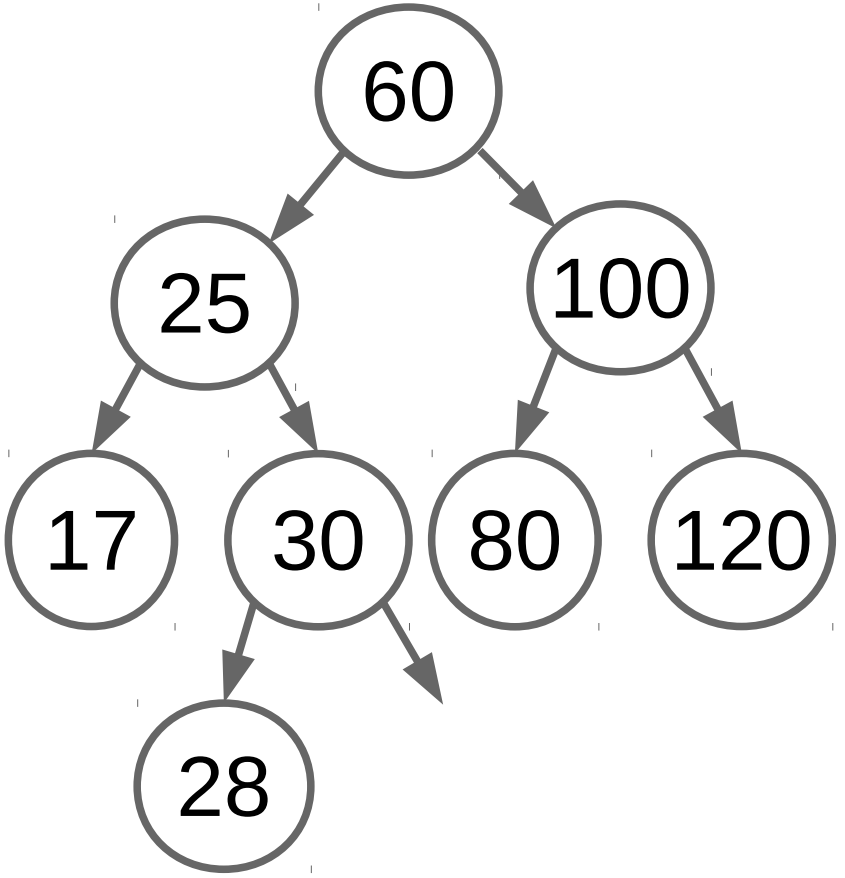




Inserção

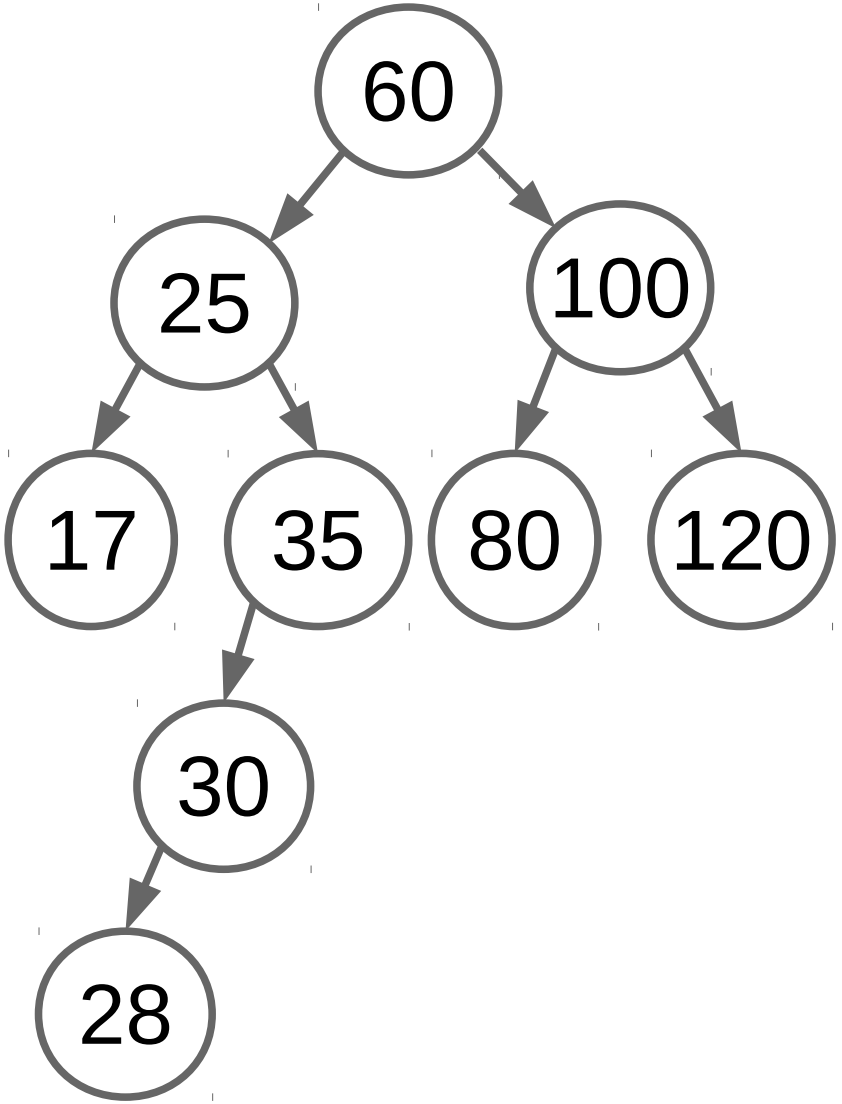
- Como a árvore ficaria balanceada?

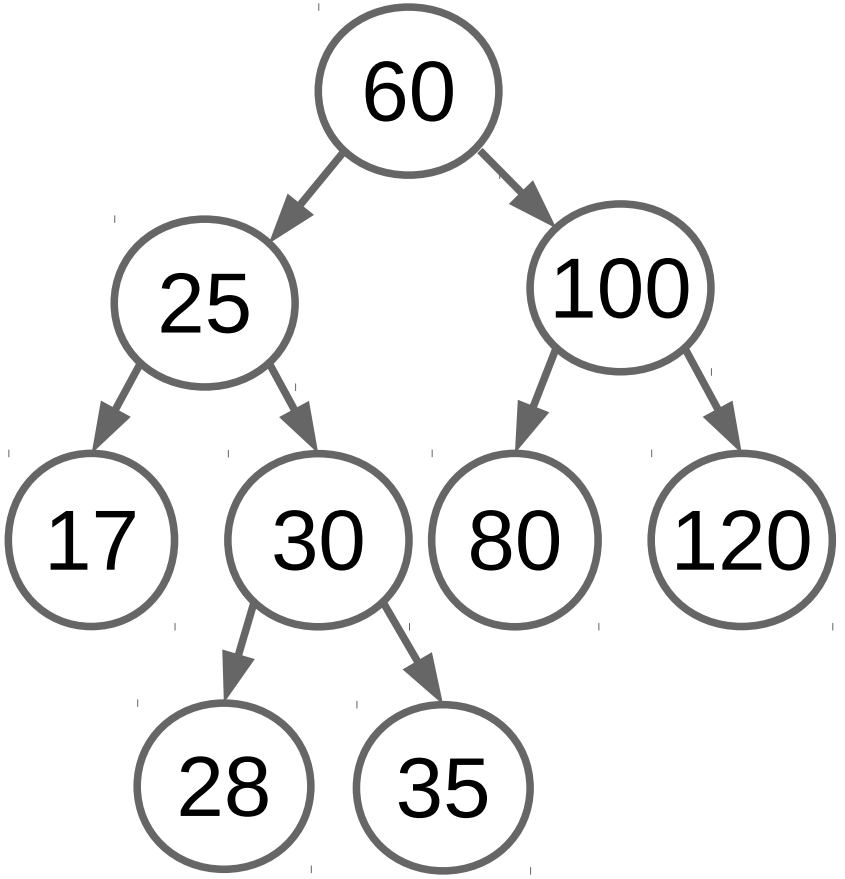




Inserção

- Como a árvore ficaria balanceada?





Inserção

Inserção

- É possível que múltiplas subárvores se tornem desbalanceadas depois da inserção de um novo elemento.

Inserção

- É possível que múltiplas subárvores se tornem desbalanceadas depois da inserção de um novo elemento.
- Porém, somente o **nó raiz mais próximo do novo elemento** terá que ser rotacionado.

Inserção

- É possível que múltiplas subárvores se tornem desbalanceadas depois da inserção de um novo elemento.
- Porém, somente o **nó raiz mais próximo do novo elemento** terá que ser rotacionado.
- Esse elemento que deverá ser rotacionado é chamado de nó pivot.

Inserção

- É possível que múltiplas subárvores se tornem desbalanceadas depois da inserção de um novo elemento.
- Porém, somente o **nó raiz mais próximo do novo elemento** terá que ser rotacionado.
- Esse elemento que deverá ser rotacionado é chamado de nó pivot.
- Portanto, nó pivô é aquele que após a inserção possui fator de balanceamento fora do intervalo.

Inserção

- É possível que múltiplas subárvores se tornem desbalanceadas depois da inserção de um novo elemento.
- Porém, somente o **nó raiz mais próximo do novo elemento** terá que ser rotacionado.
- Esse elemento que deverá ser rotacionado é chamado de nó pivot.
- Portanto, nó pivô é aquele que após a inserção possui fator de balanceamento fora do intervalo.
- Uma subárvore AVL é rebalanceada ao executarmos uma rotação em volta desse *pivot*.
- Existem quatro casos de rotação do *pivot*.

Inserção

Inserção

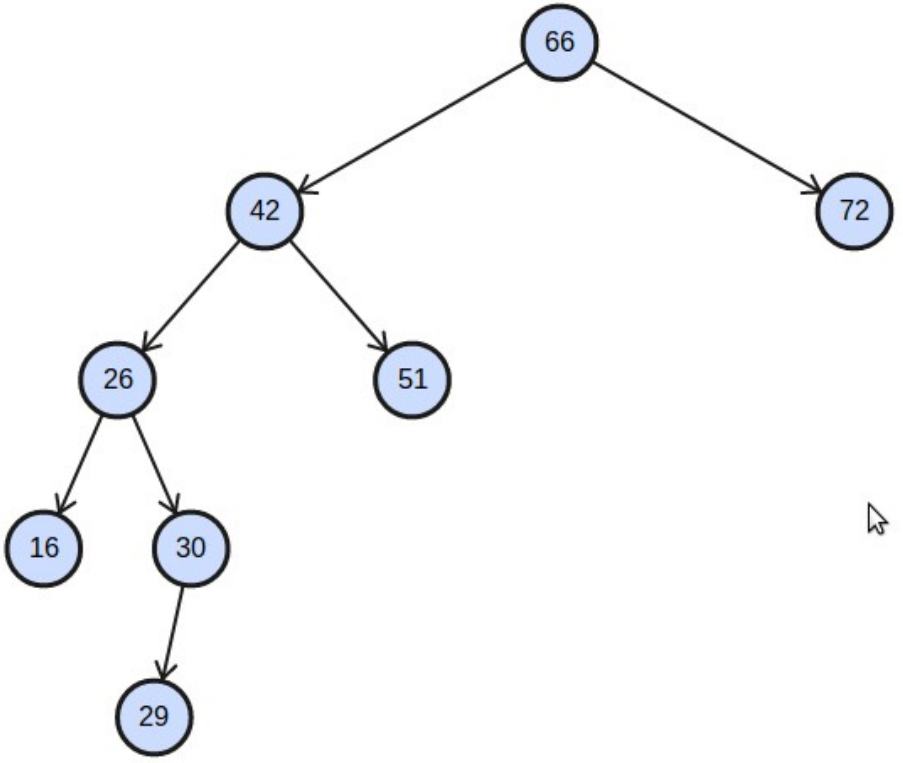
- Caso 1

Inserção

- Caso 1
  - Ocorre quando o fator de balanço do lado esquerdo do nó *pivot* (P) é maior e o novo elemento é inserido no lado do filho esquerdo (C) do *pivot*.

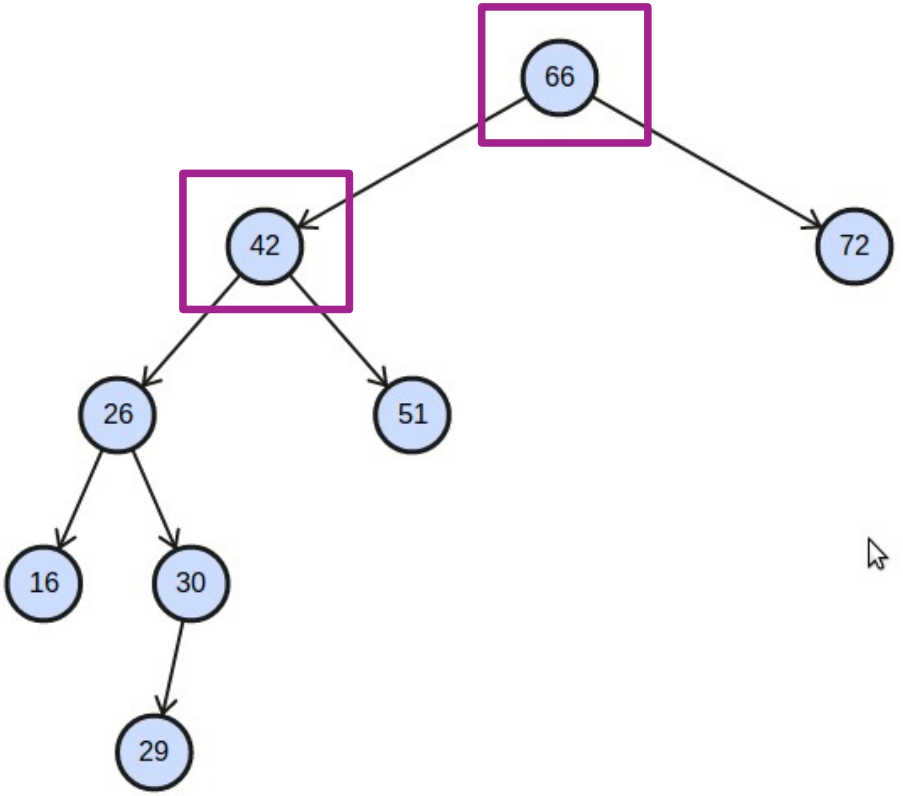
Inserção

- Caso 1
  - Ocorre quando o fator de balanço do lado esquerdo do nó *pivot* (P) é maior e o novo elemento é inserido no lado do filho esquerdo (C) do *pivot*.



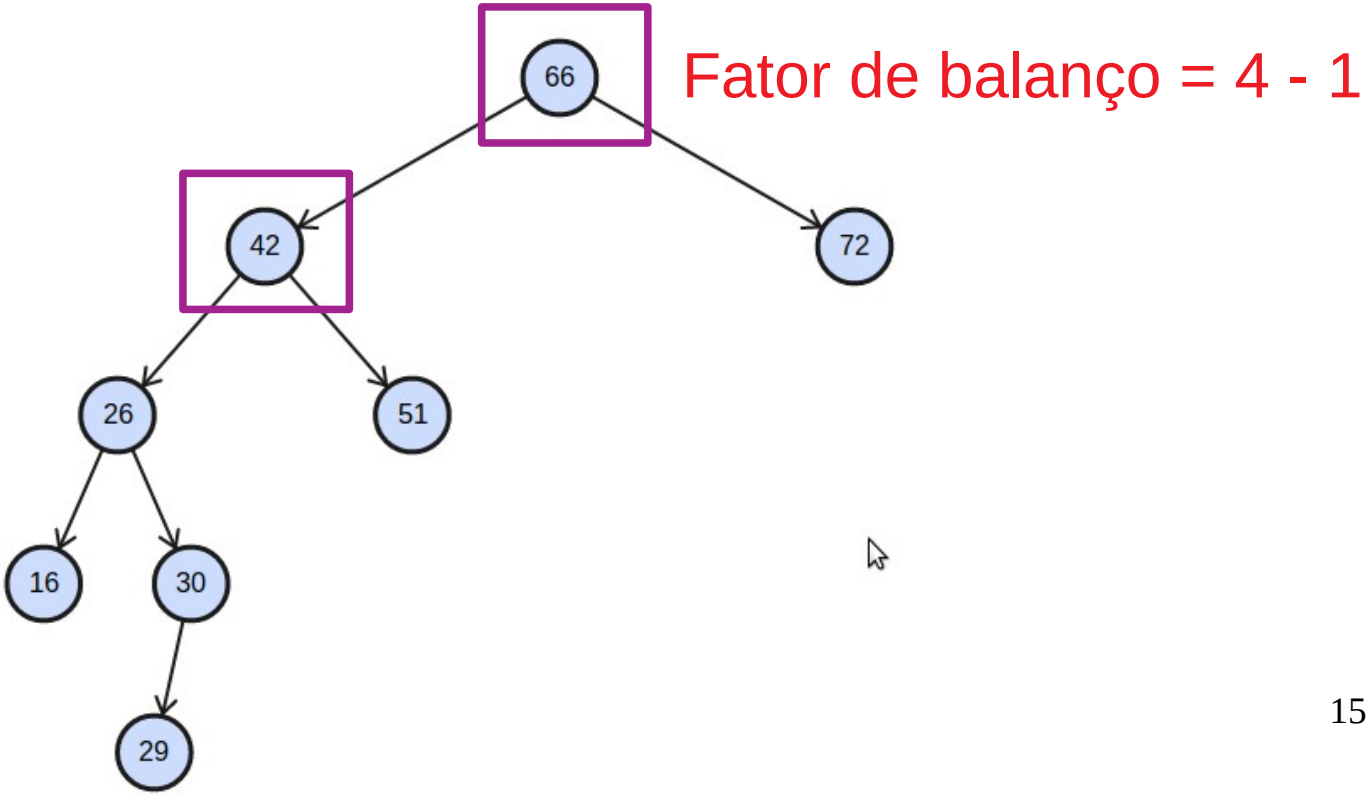
Inserção

- Caso 1
  - Ocorre quando o fator de balanço do lado esquerdo do nó *pivot* (P) é maior e o novo elemento é inserido no lado do filho esquerdo (C) do *pivot*.



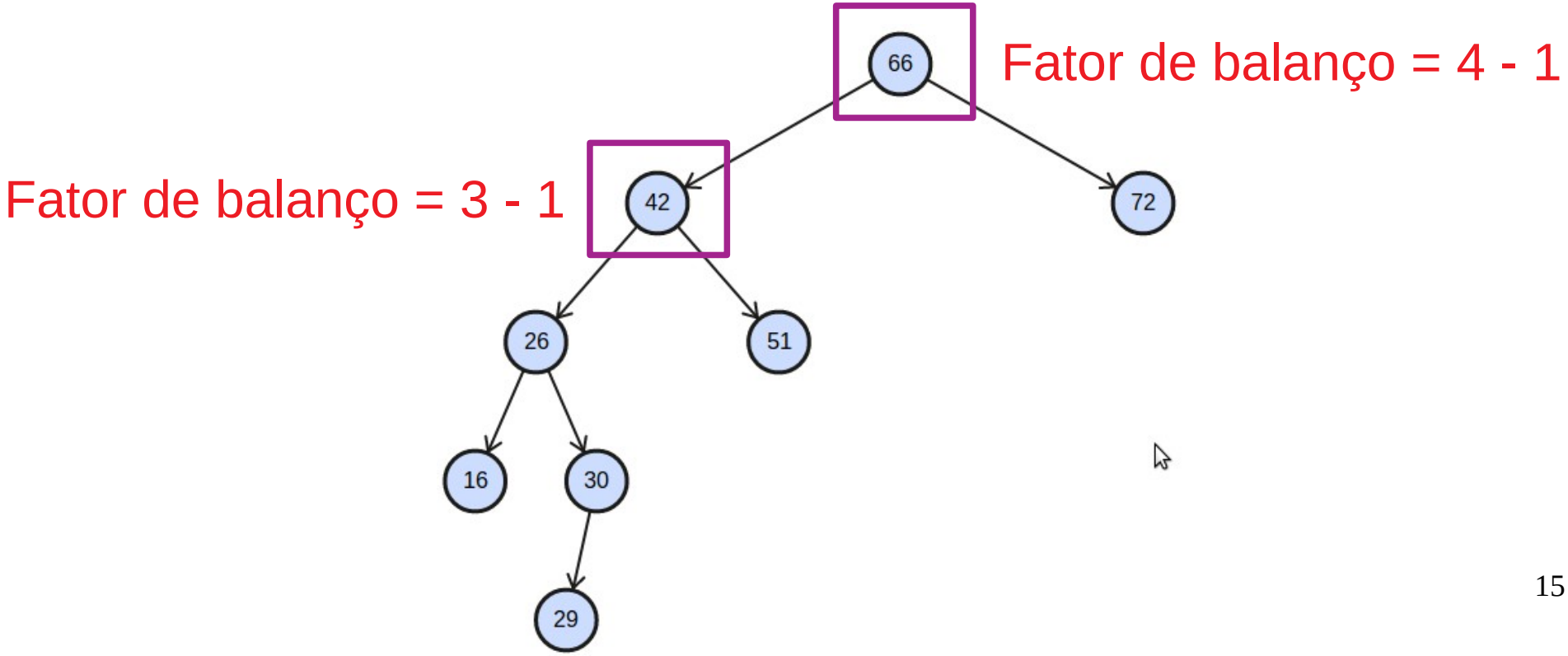
Inserção

- Caso 1
  - Ocorre quando o fator de balanço do lado esquerdo do nó *pivot* (P) é maior e o novo elemento é inserido no lado do filho esquerdo (C) do *pivot*.



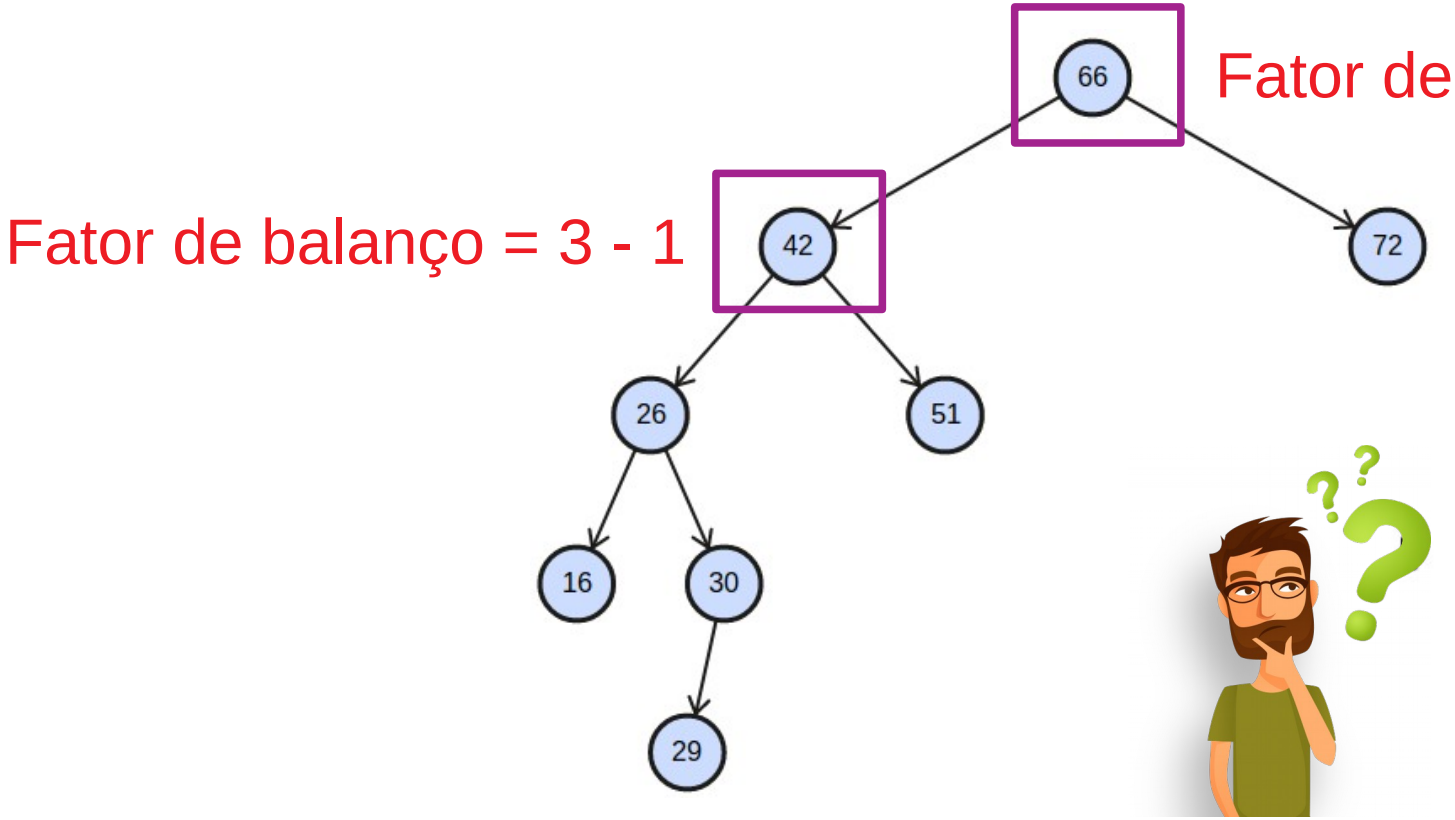
Inserção

- Caso 1
  - Ocorre quando o fator de balanço do lado esquerdo do nó *pivot* (P) é maior e o novo elemento é inserido no lado do filho esquerdo (C) do *pivot*.



Inserção

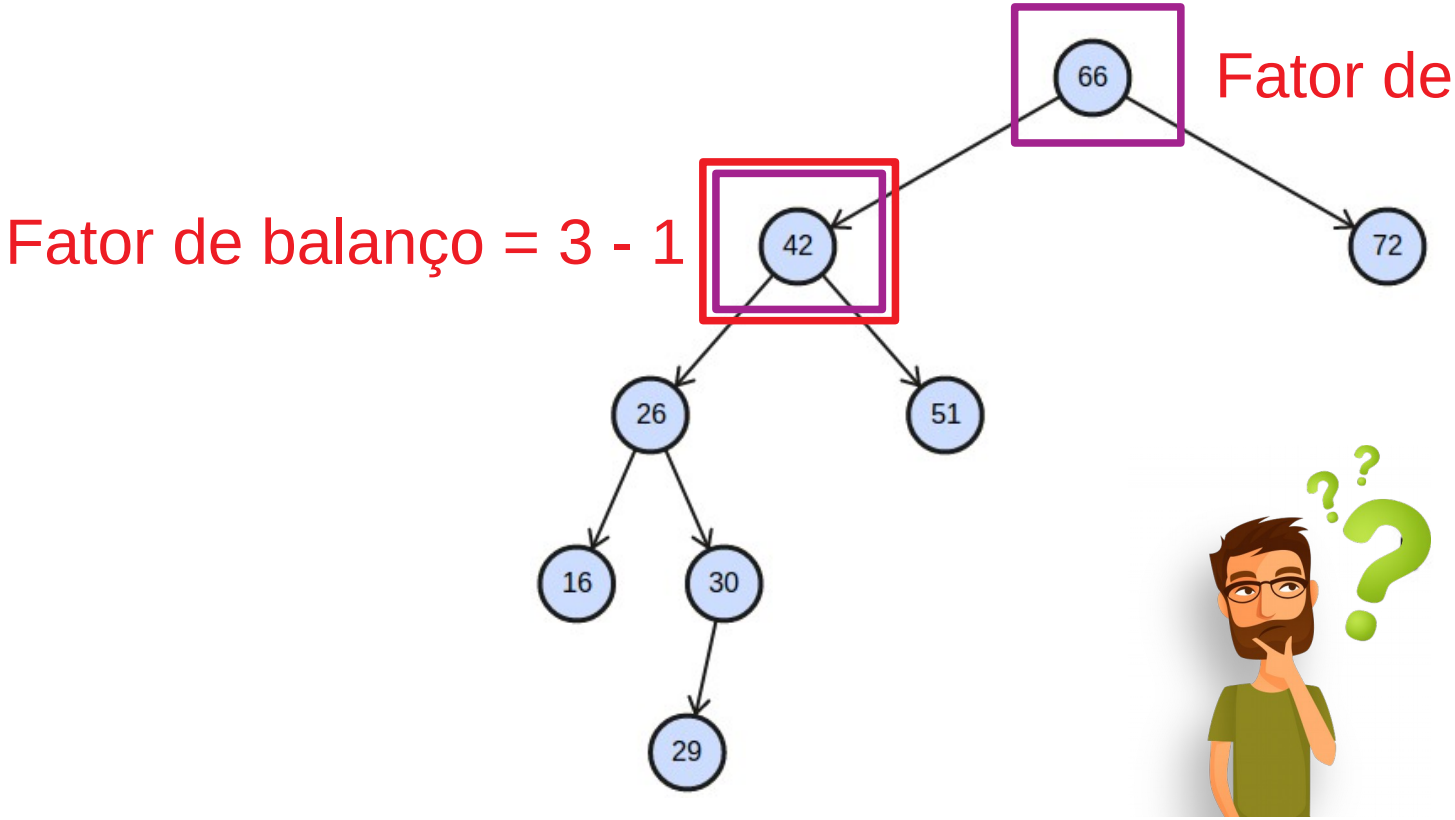
- Caso 1
  - Ocorre quando o fator de balanço do lado esquerdo do nó *pivot* (P) é maior e o novo elemento é inserido no lado do filho esquerdo (C) do *pivot*.



Quem será o *pivot*?

Inserção

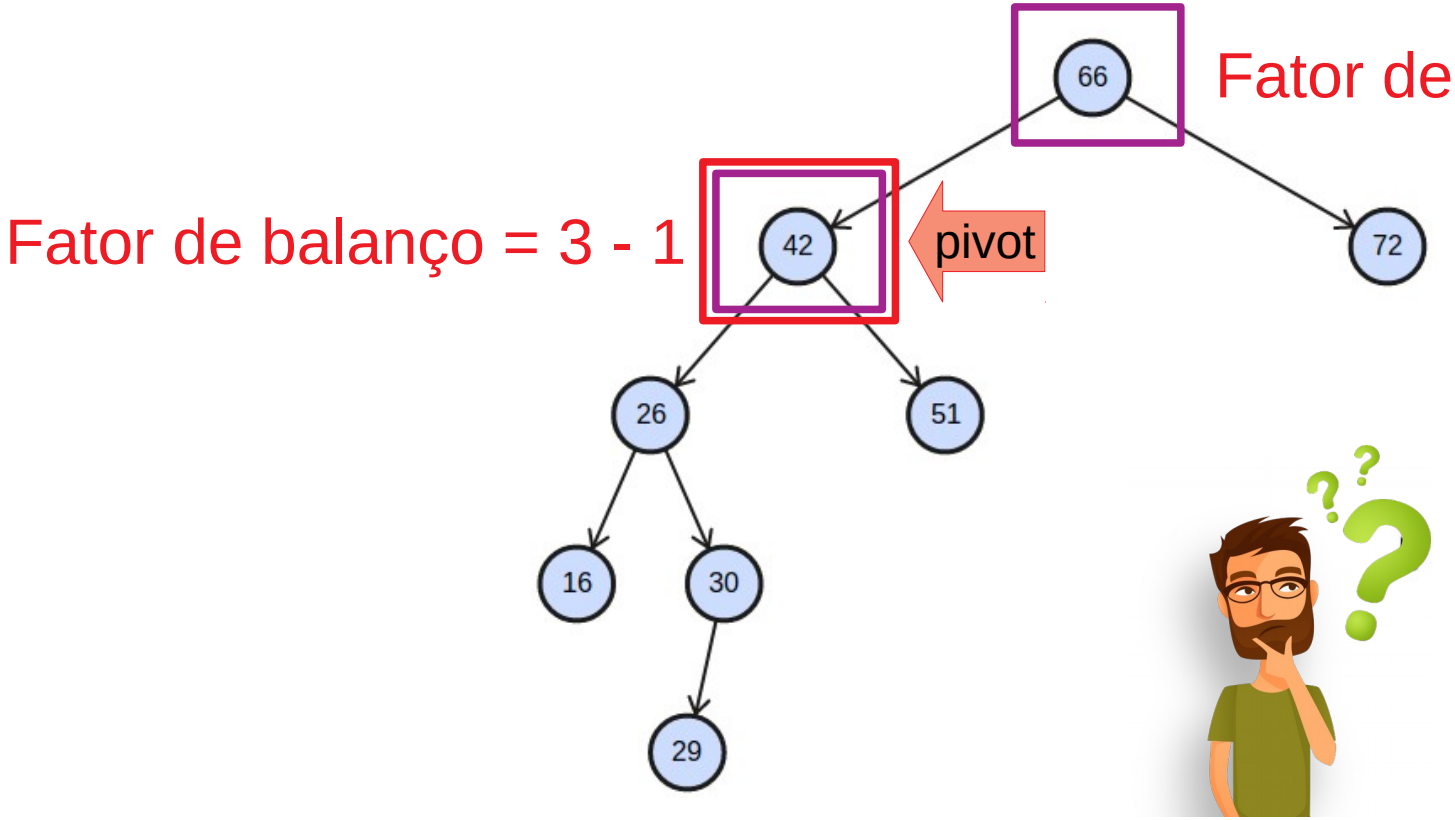
- Caso 1
  - Ocorre quando o fator de balanço do lado esquerdo do nó *pivot* (P) é maior e o novo elemento é inserido no lado do filho esquerdo (C) do *pivot*.



Quem será o *pivot*?

Inserção

- Caso 1
  - Ocorre quando o fator de balanço do lado esquerdo do nó *pivot* (P) é maior e o novo elemento é inserido no lado do filho esquerdo (C) do *pivot*.



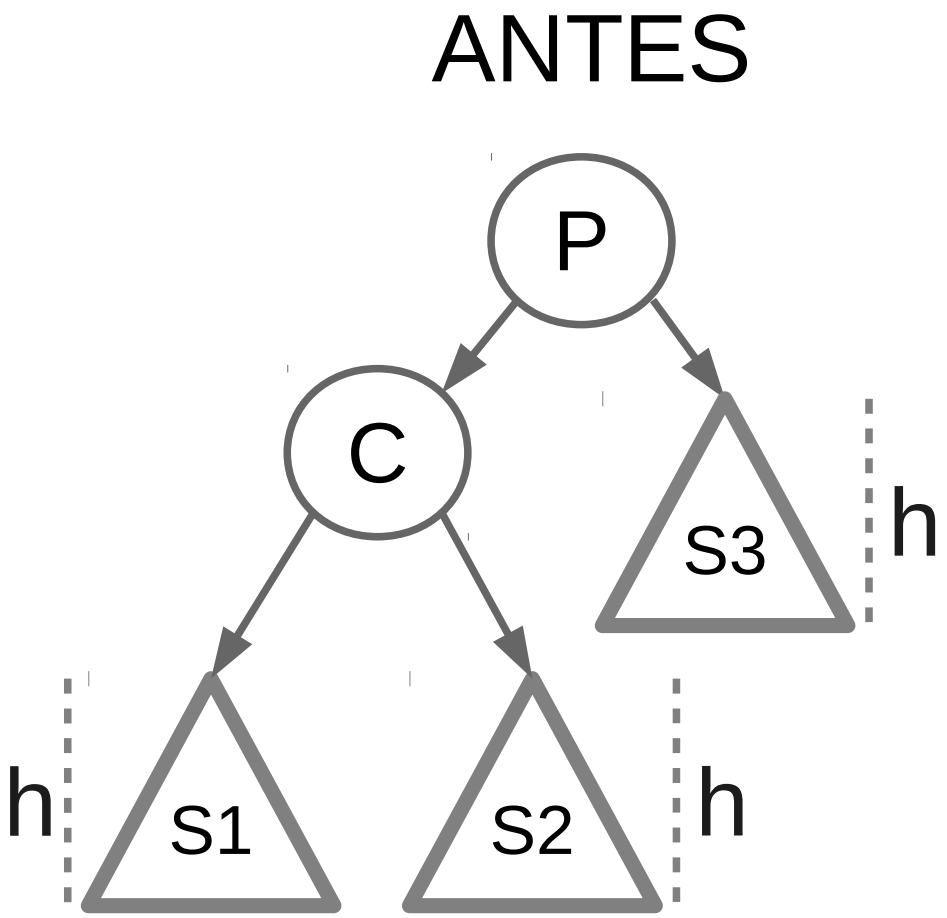
Quem será o *pivot*?

Inserção

Inserção

ANTES

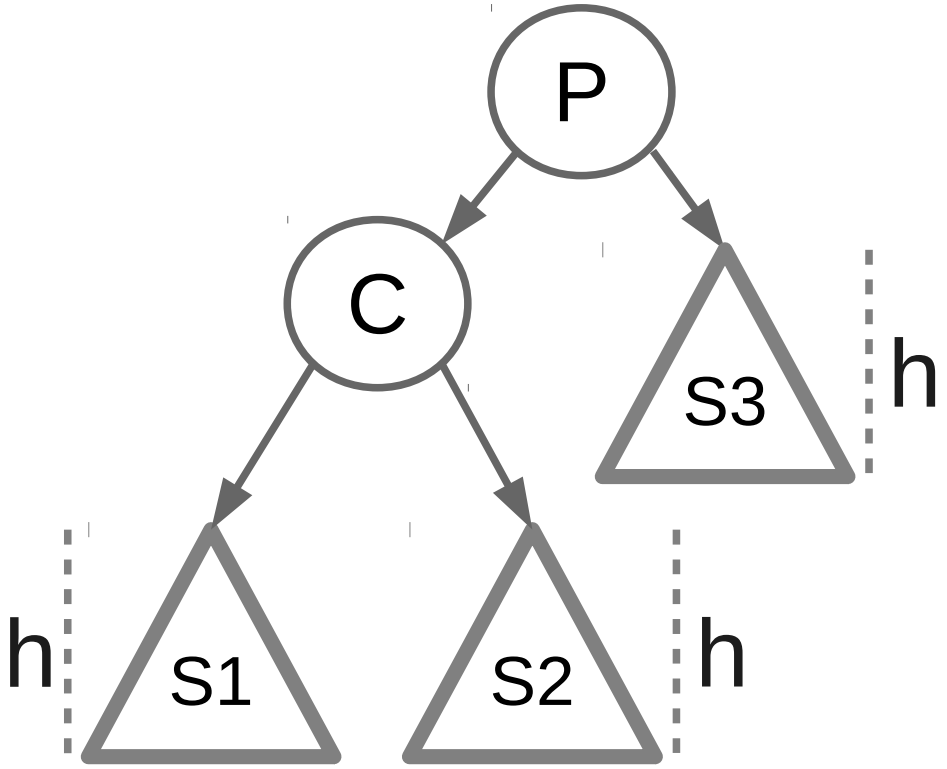
Inserção



Inserção

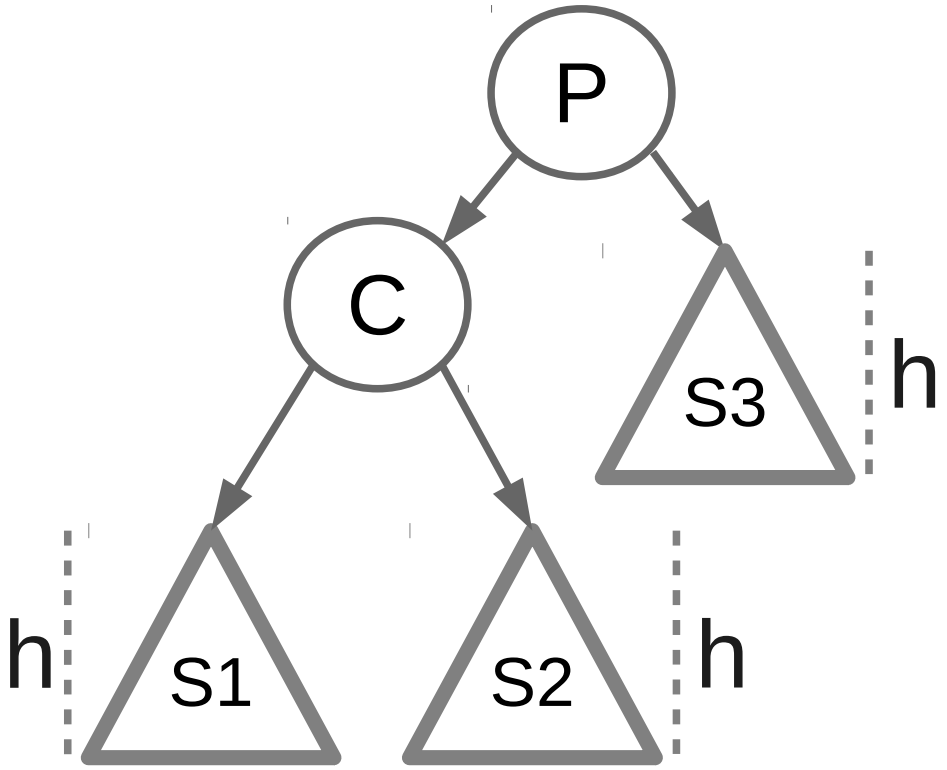
ANTES

DEPOIS

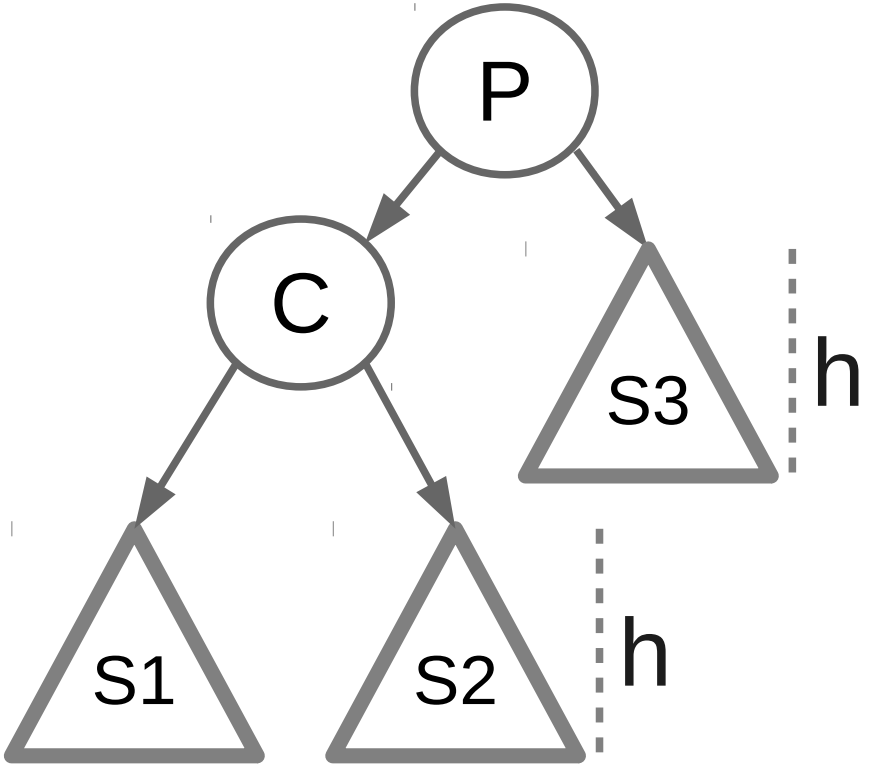


Inserção

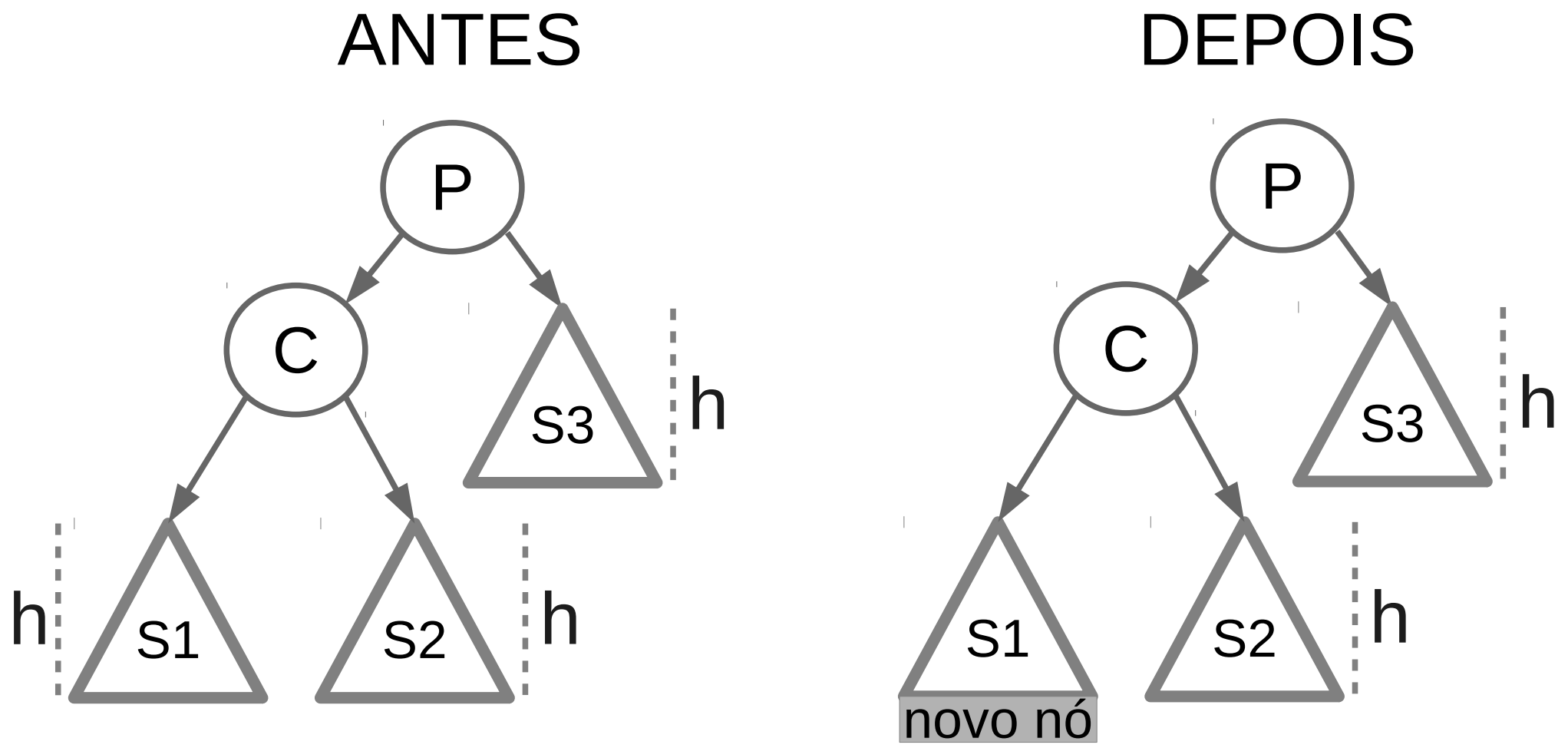
ANTES



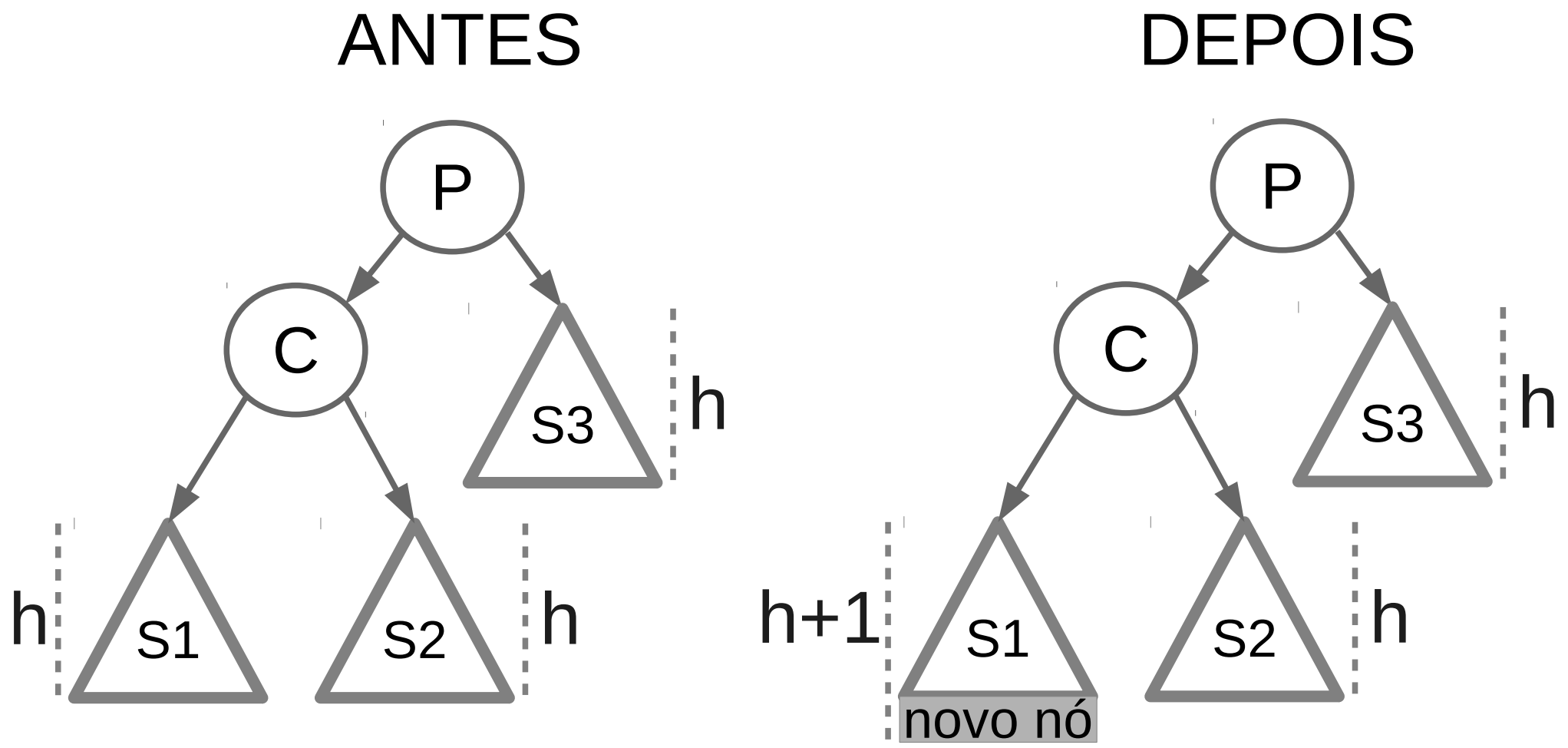
DEPOIS



Inserção



Inserção



Inserção

Inserção

- O nó *pivot* deve ser rotacionado para direita do seu filho à esquerda.

Inserção

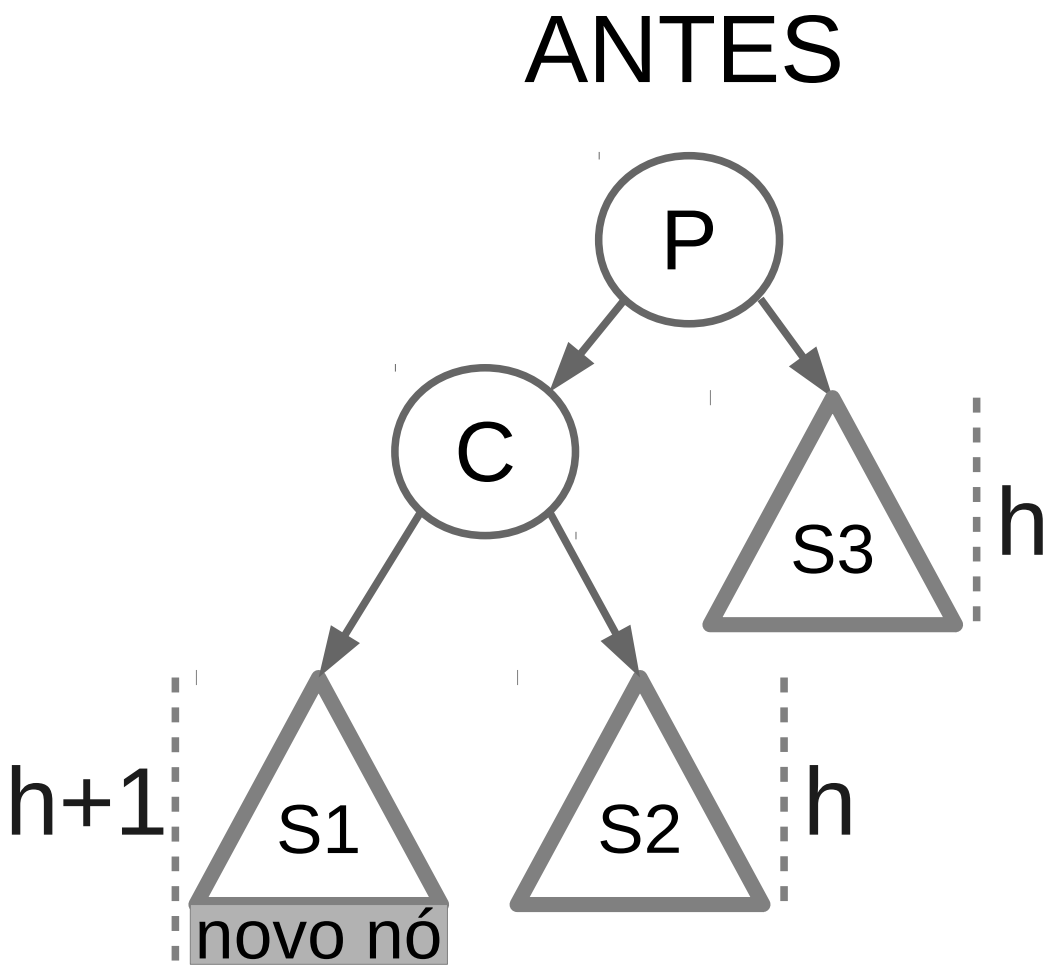
- O nó *pivot* deve ser rotacionado para direita do seu filho à esquerda.
- P se torna o filho à direita de C e o filho à direita de C se torna o filho à esquerda de P.

Inserção

Inserção

ANTES

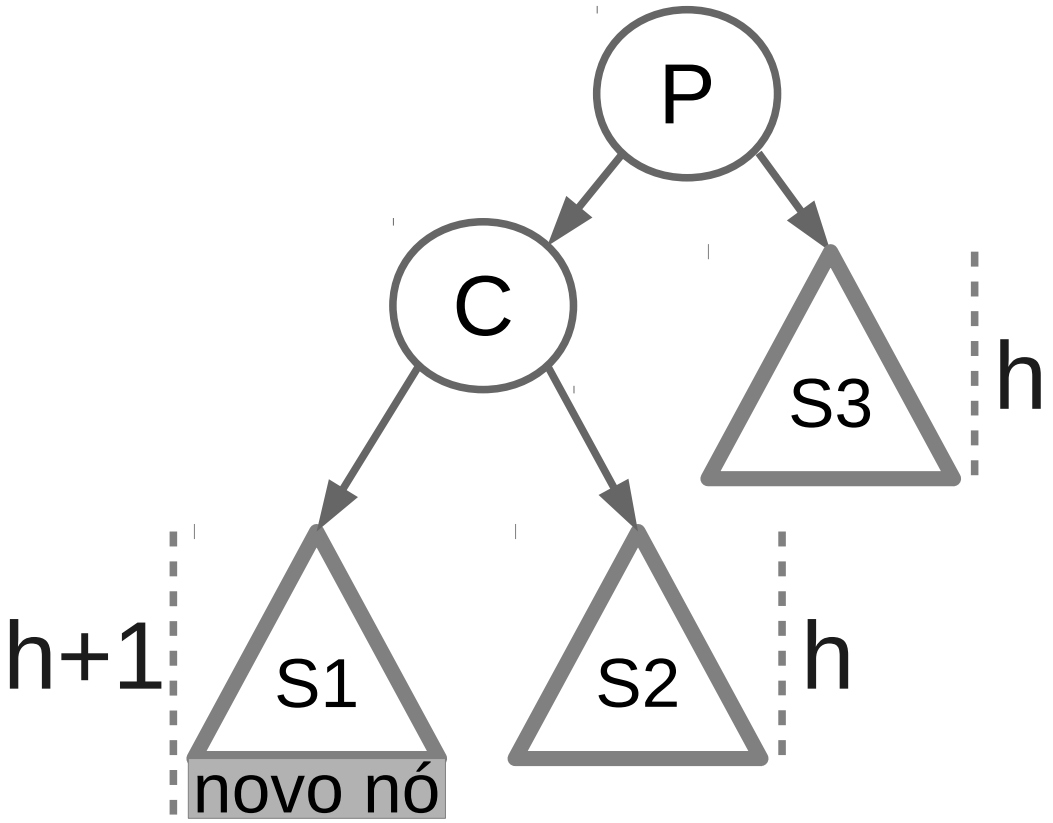
Inserção



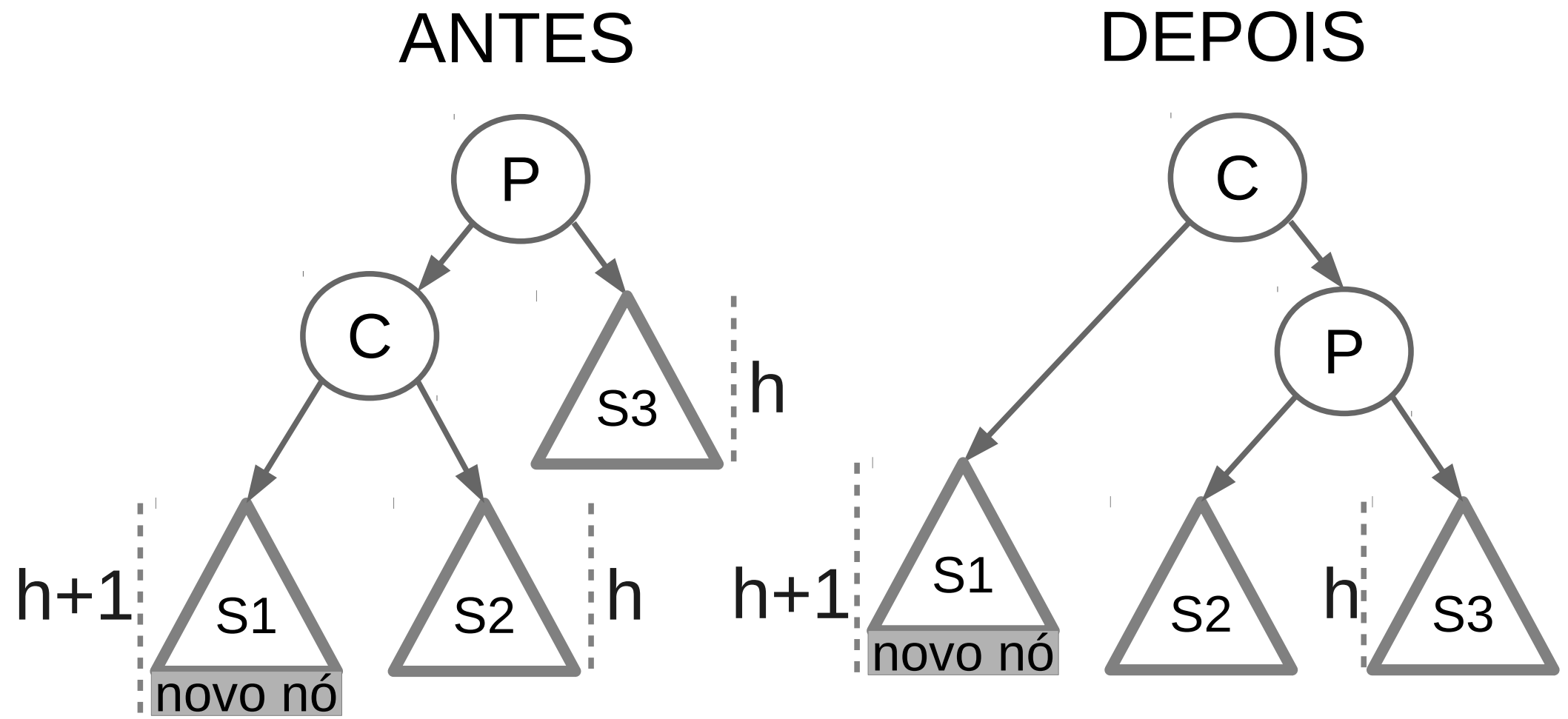
Inserção

ANTES

DEPOIS



Inserção



Inserção

- Exemplo Caso 1

Inserção

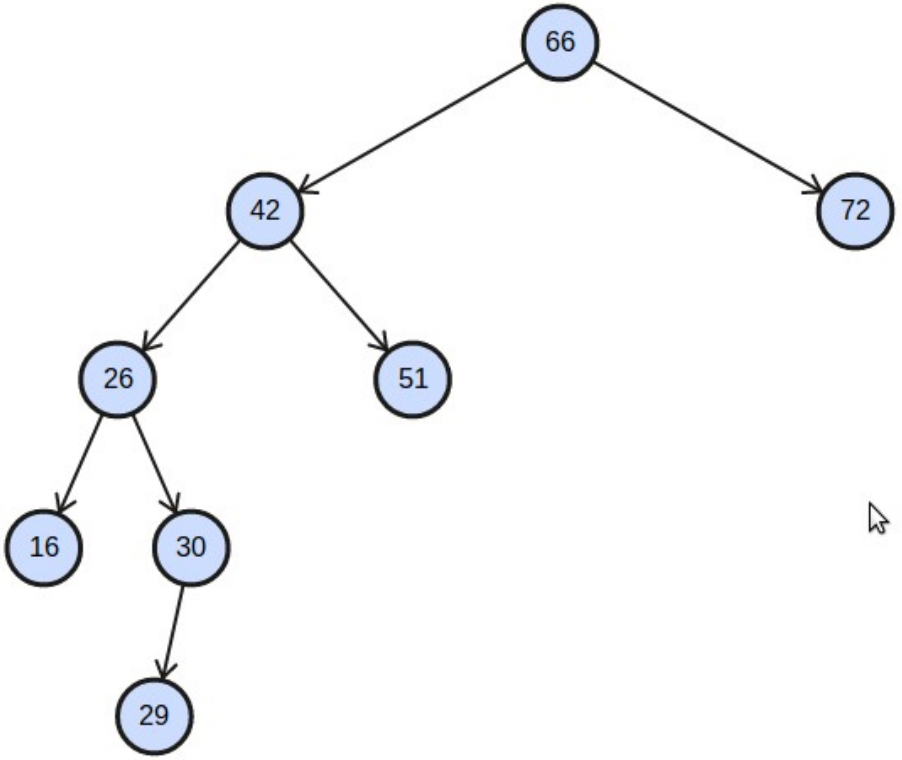
- Exemplo Caso 1

ANTES

Inserção

- Exemplo Caso 1

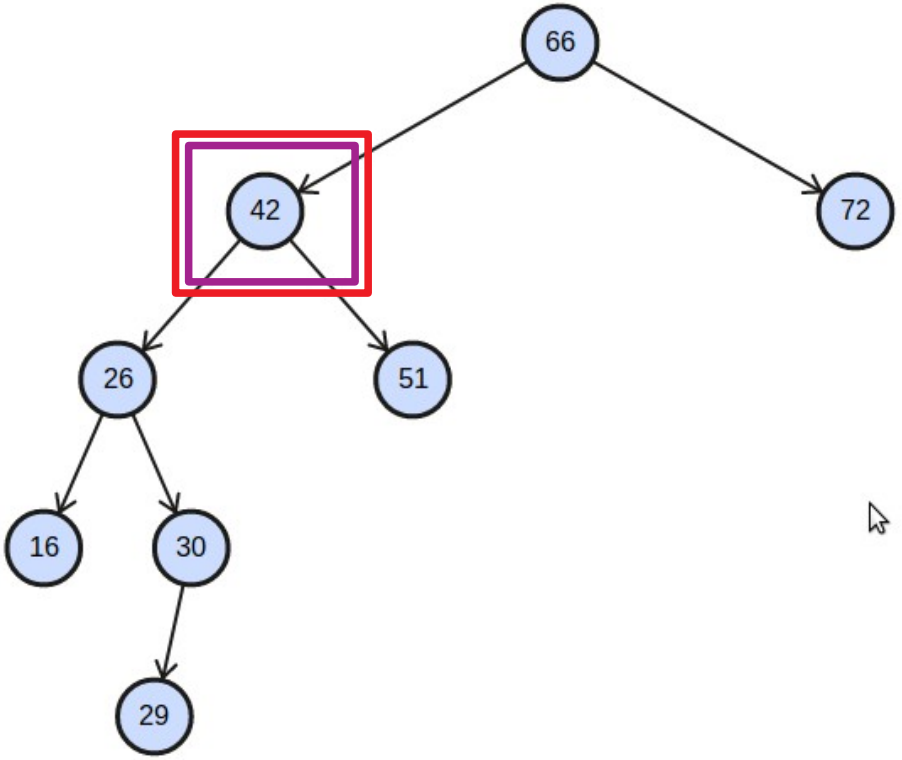
ANTES



Inserção

- Exemplo Caso 1

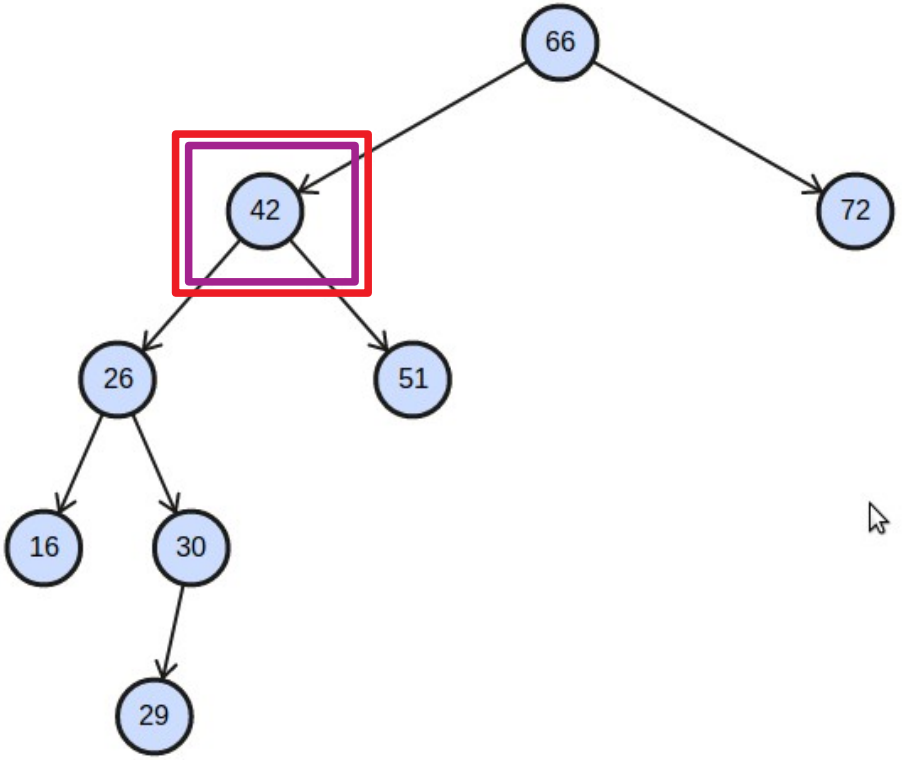
ANTES



Inserção

- Exemplo Caso 1

ANTES

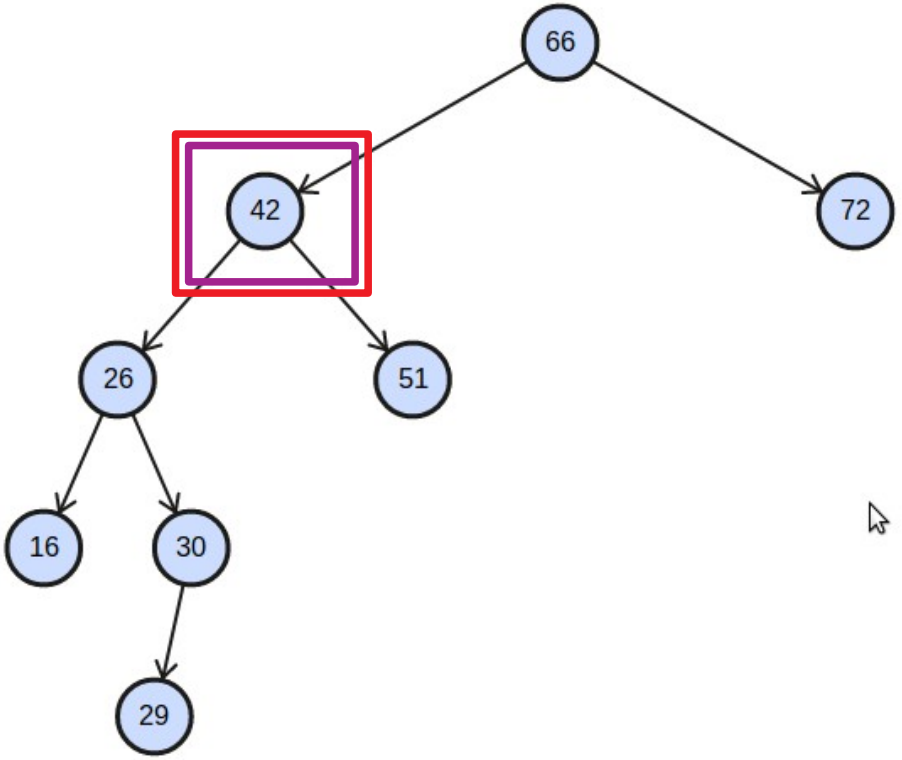


DEPOIS

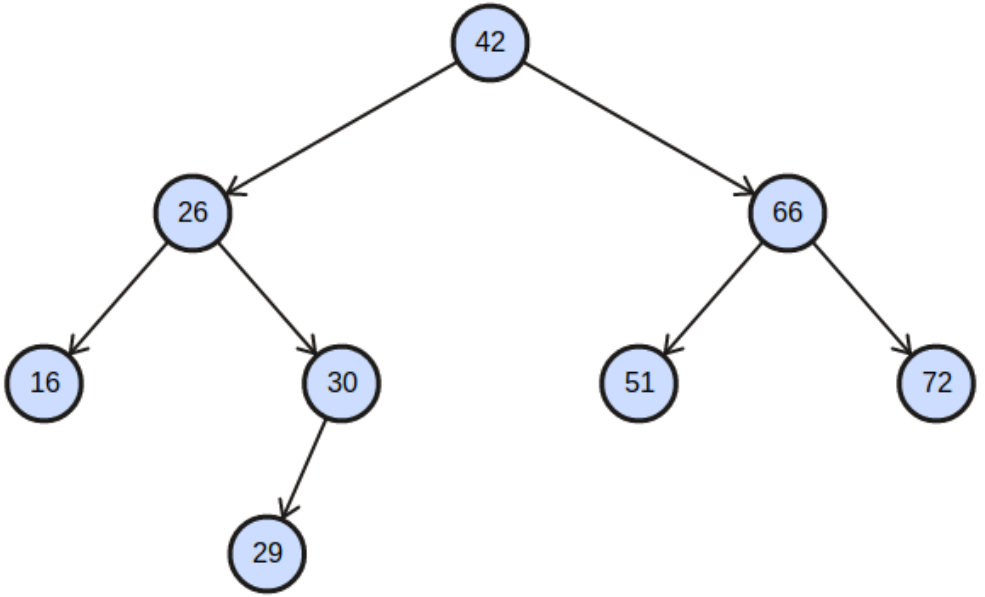
Inserção

- Exemplo Caso 1

ANTES



DEPOIS



Inserção

- Caso 2

Inserção

- Caso 2
  - Este caso envolve três nós: *pivot* (P), o seu filho à esquerda (C) e o filho deste à direita (G).

Inserção

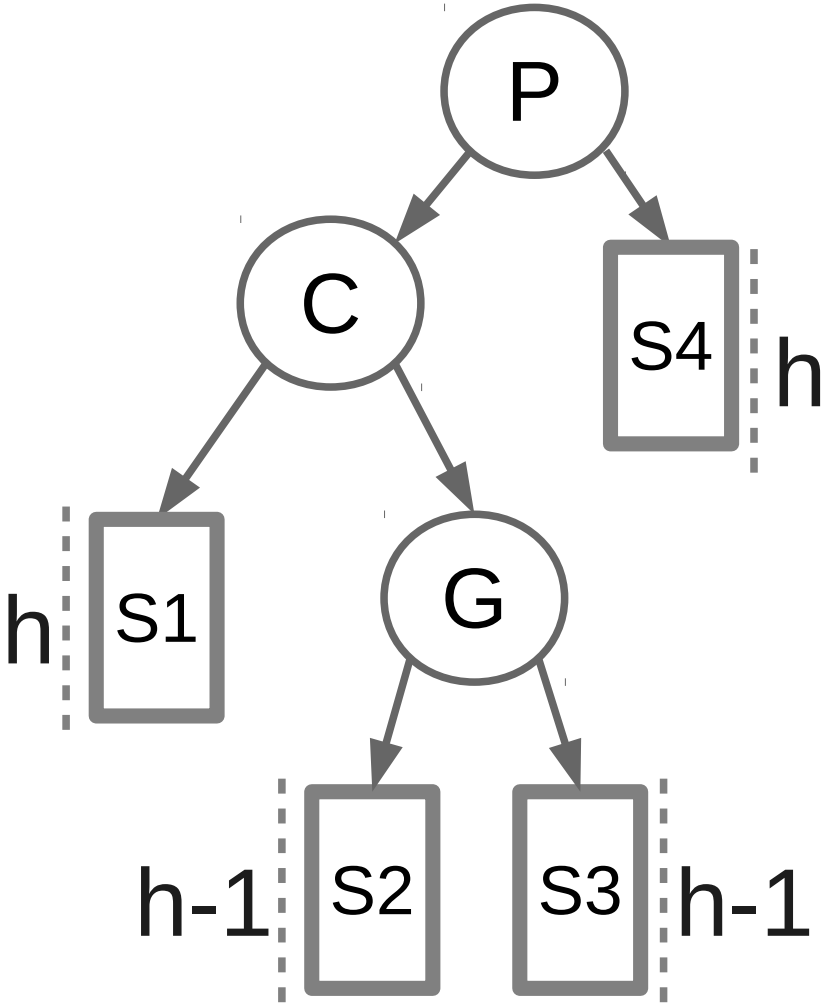
- Caso 2
  - Este caso envolve três nós: *pivot* (P), o seu filho à esquerda (C) e o filho deste à direita (G).
  - Para este caso ocorrer, o fator de balanço à esquerda (C) do nó *pivot* (P) é maior e o novo elemento é inserido do lado direito de C.

Inserção

- Caso 2

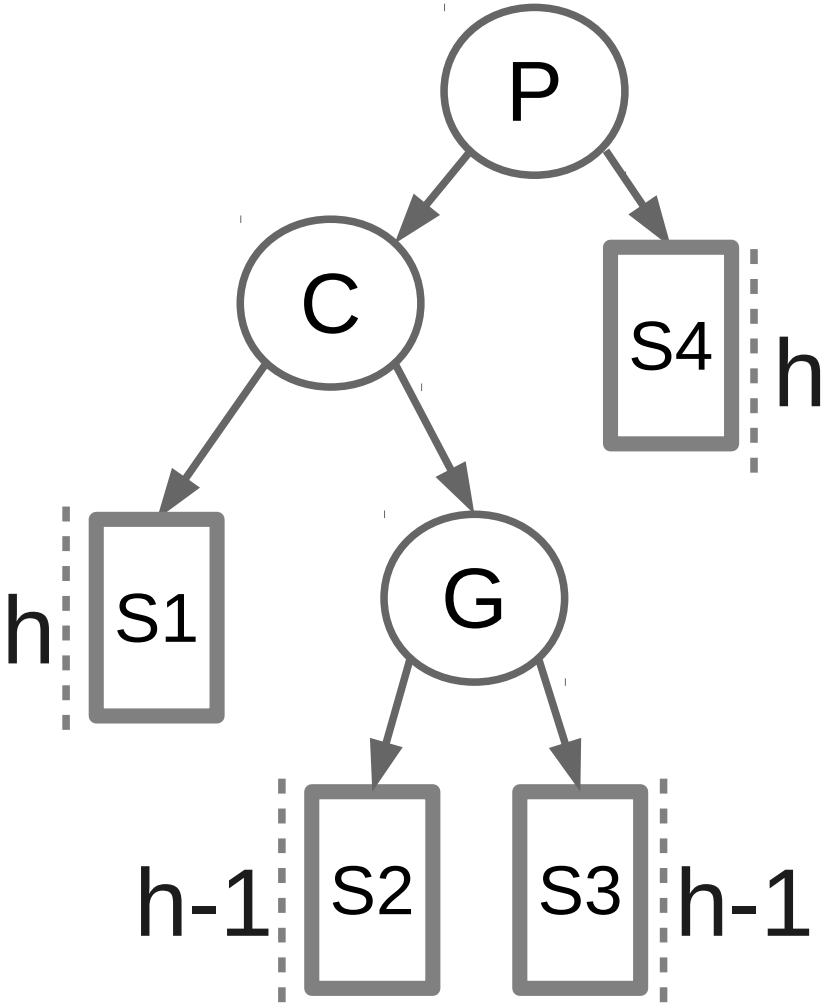
Inserção

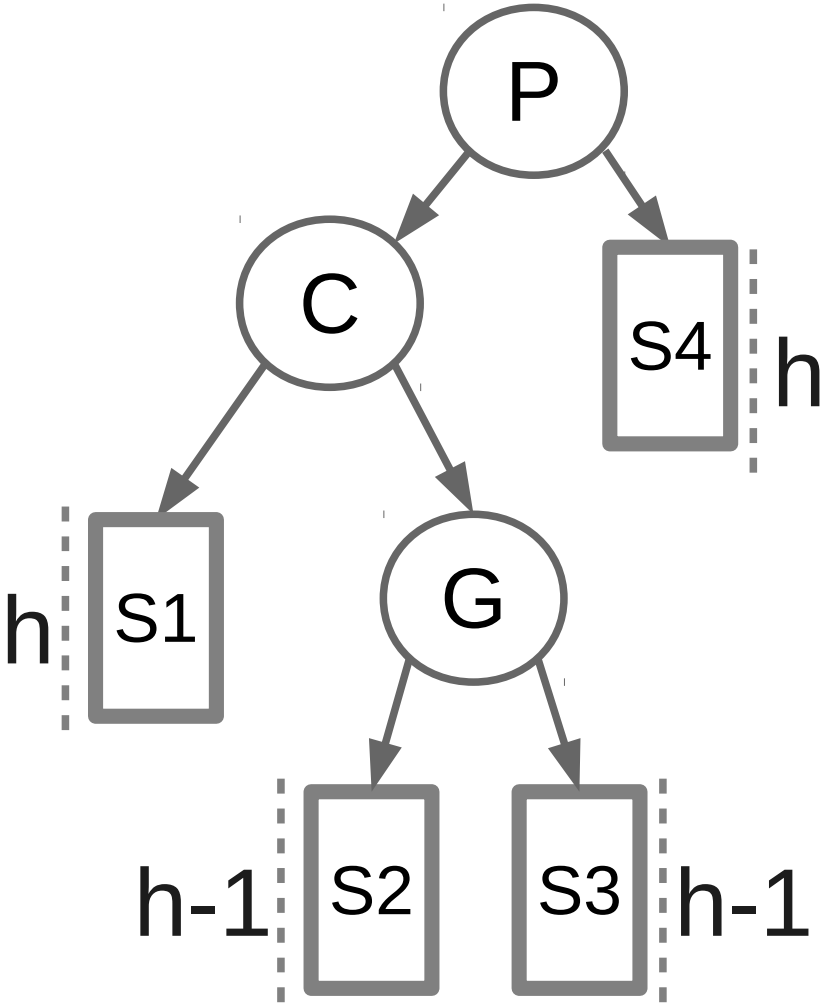
- Caso 2



Inserção

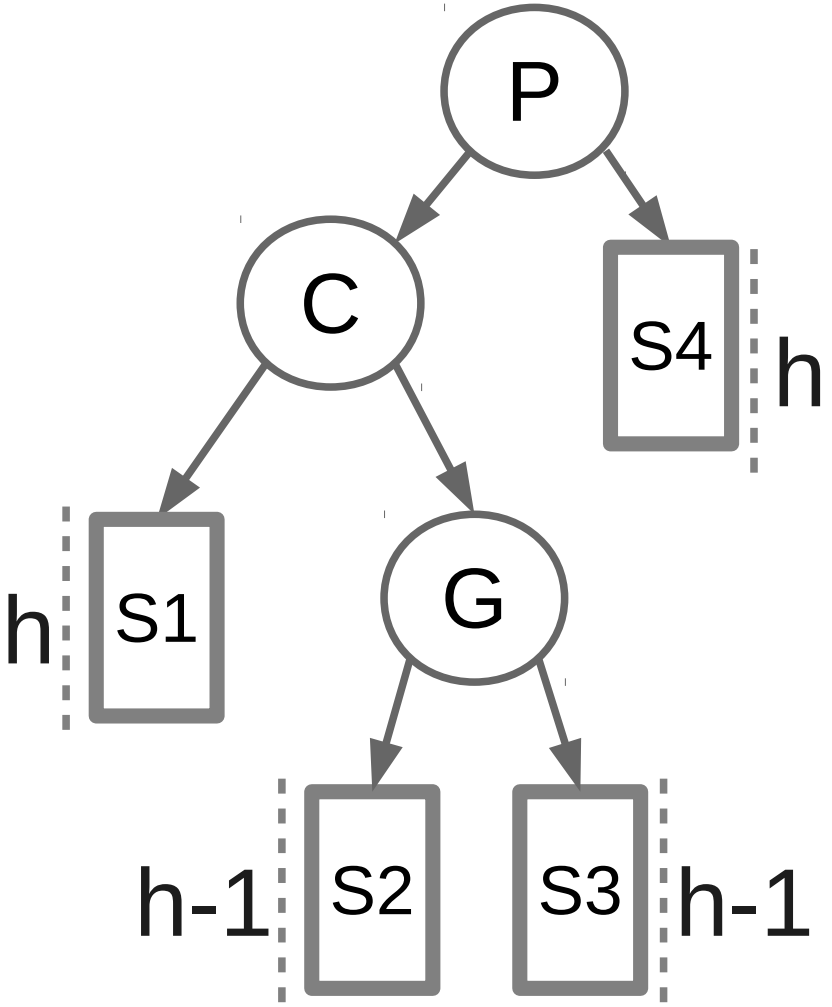
- Caso 2

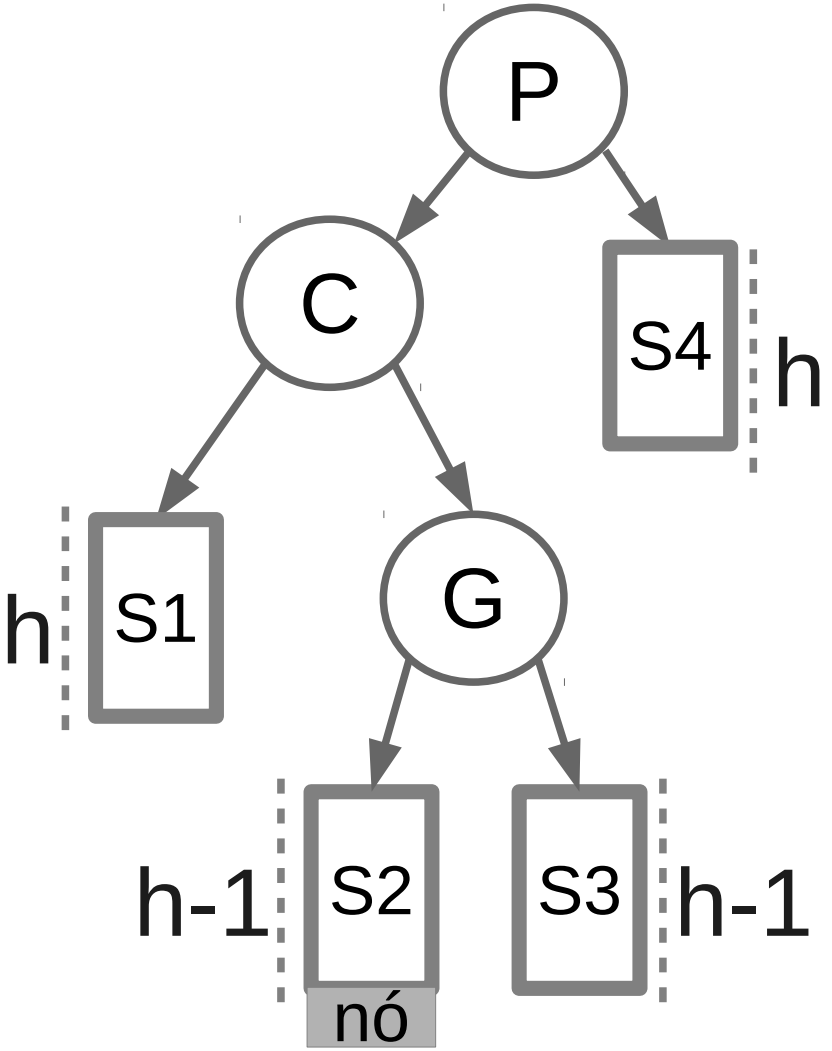




Inserção

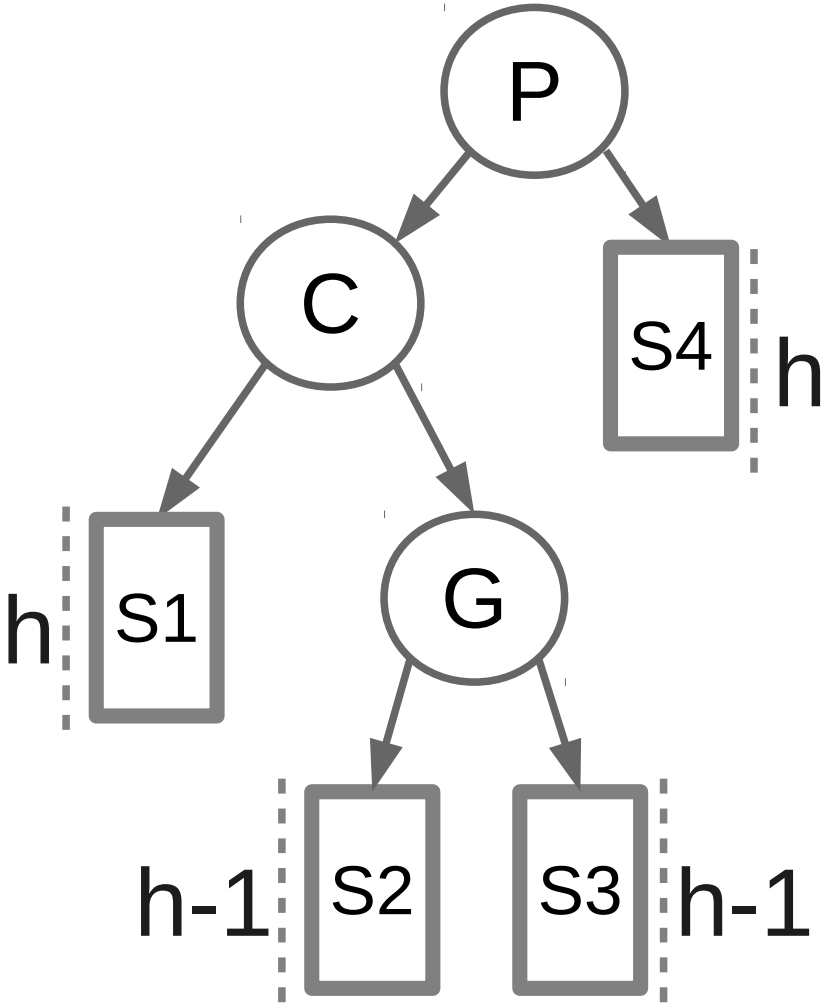
- Caso 2

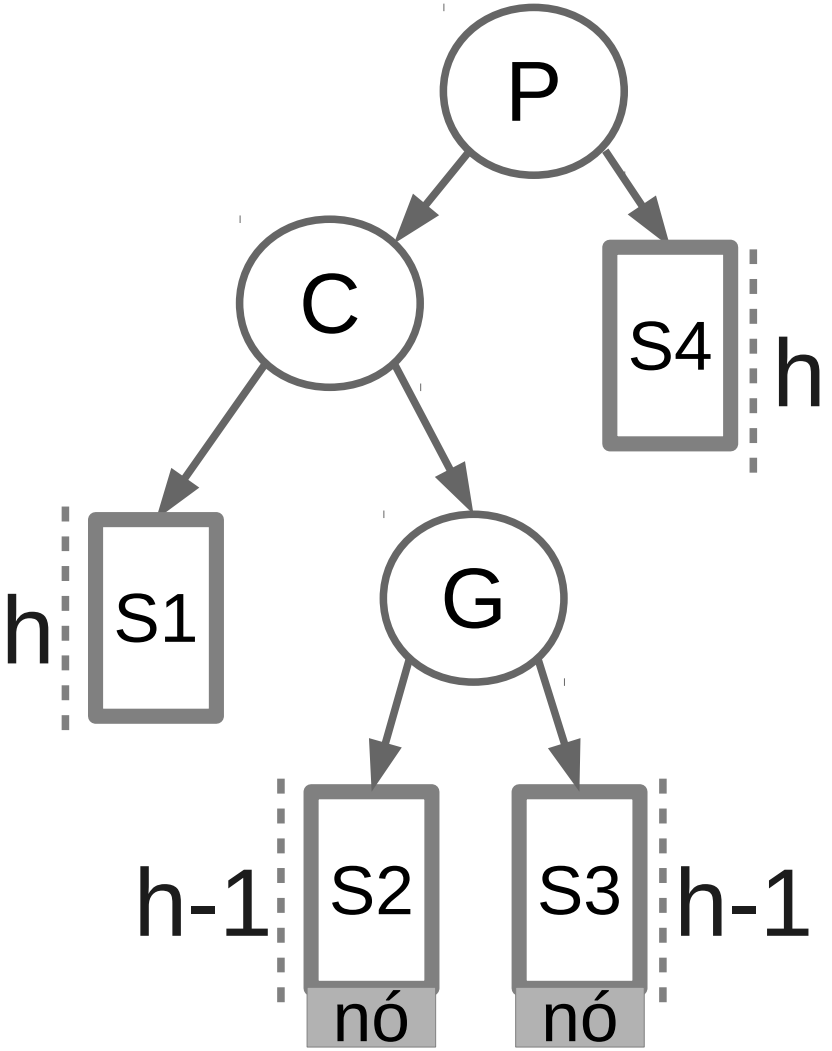




Inserção

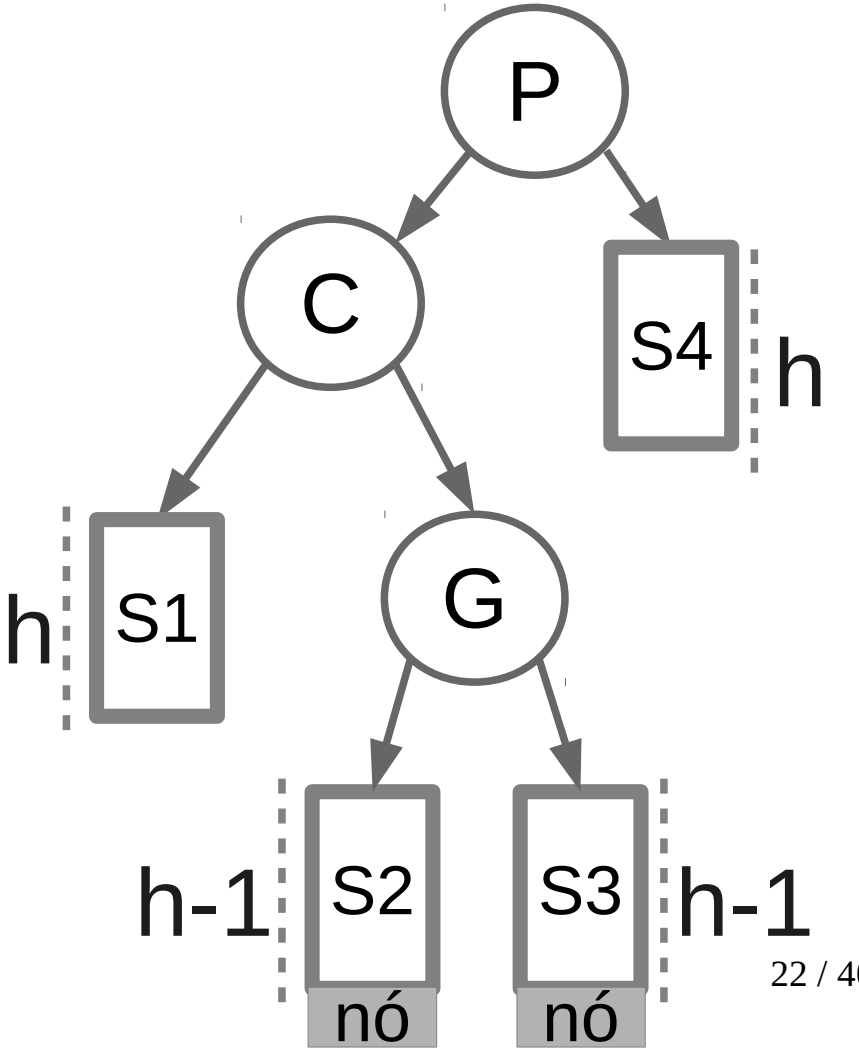
- Caso 2





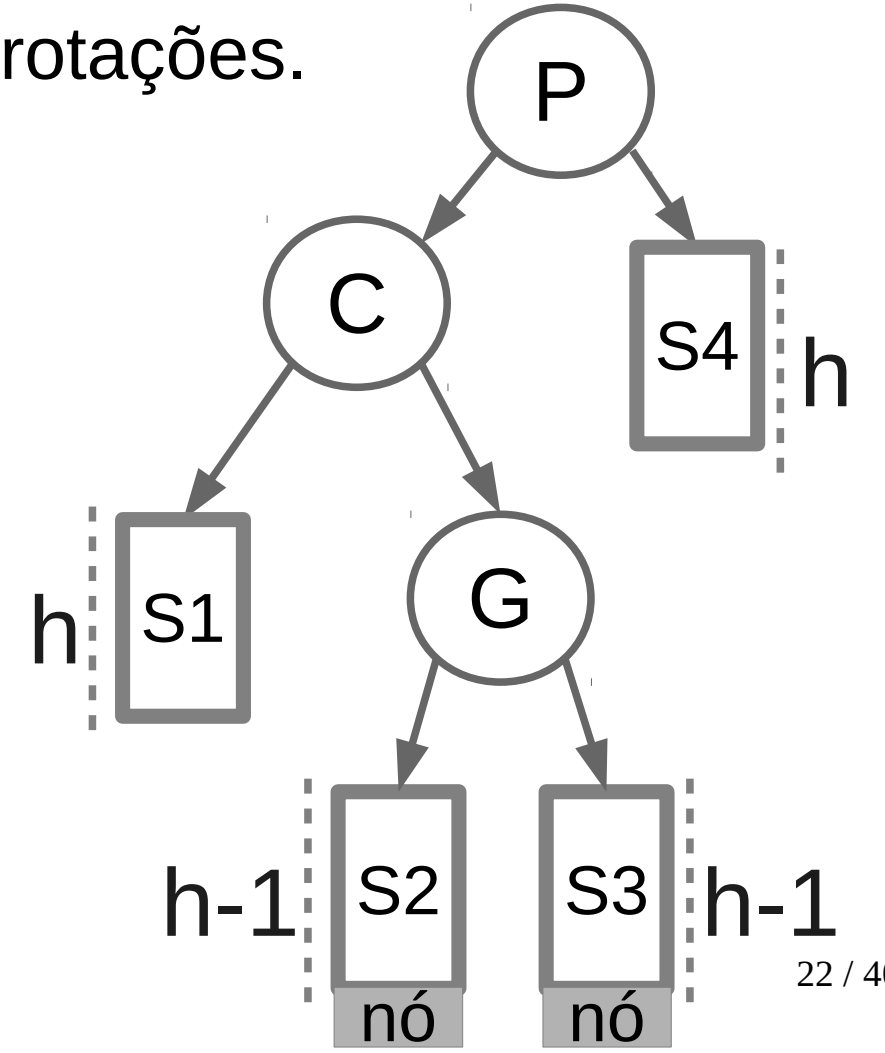
Inserção

- Caso 2



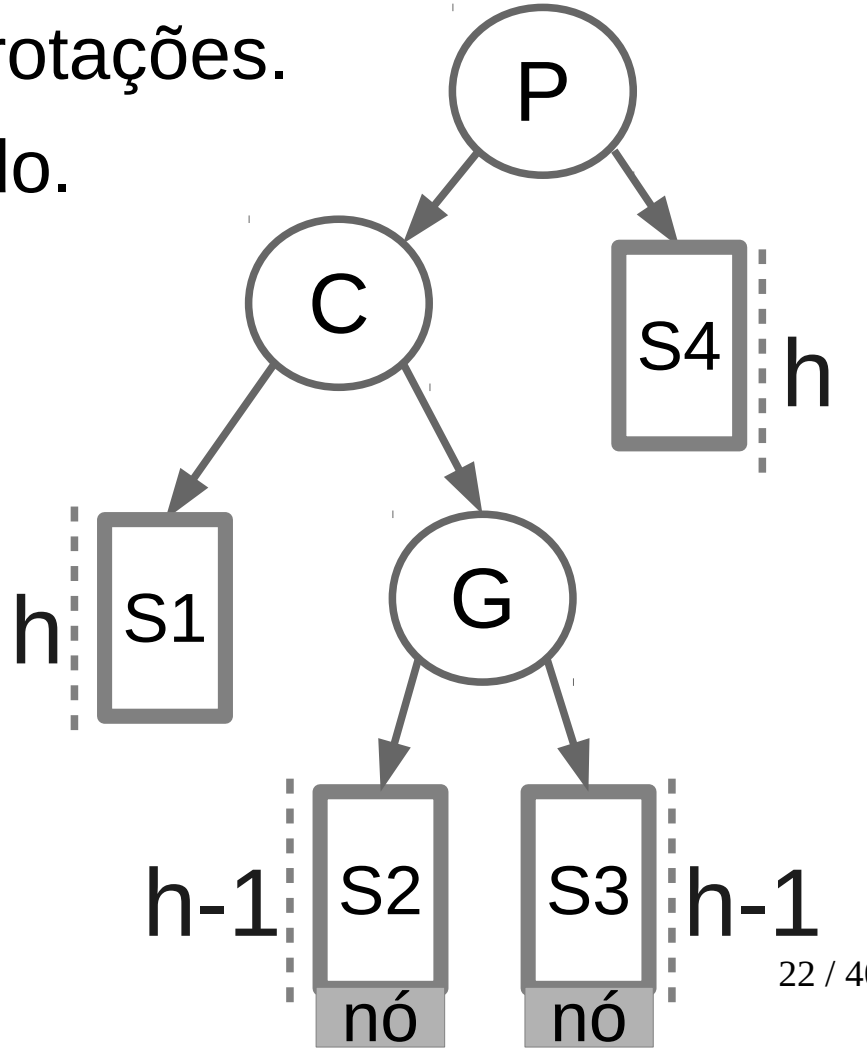
Inserção

- Caso 2
  - Serão necessárias duas rotações.



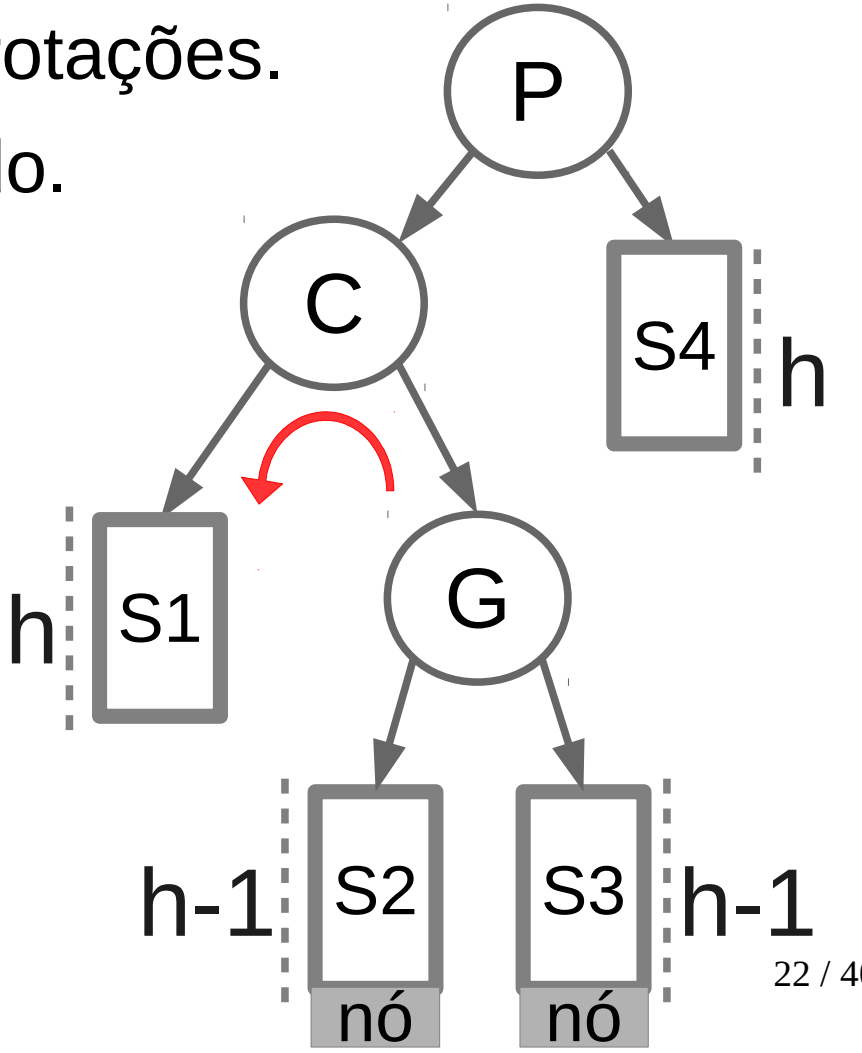
Inserção

- Caso 2
  - Serão necessárias duas rotações.
  - Nó G deve ser rotacionado.



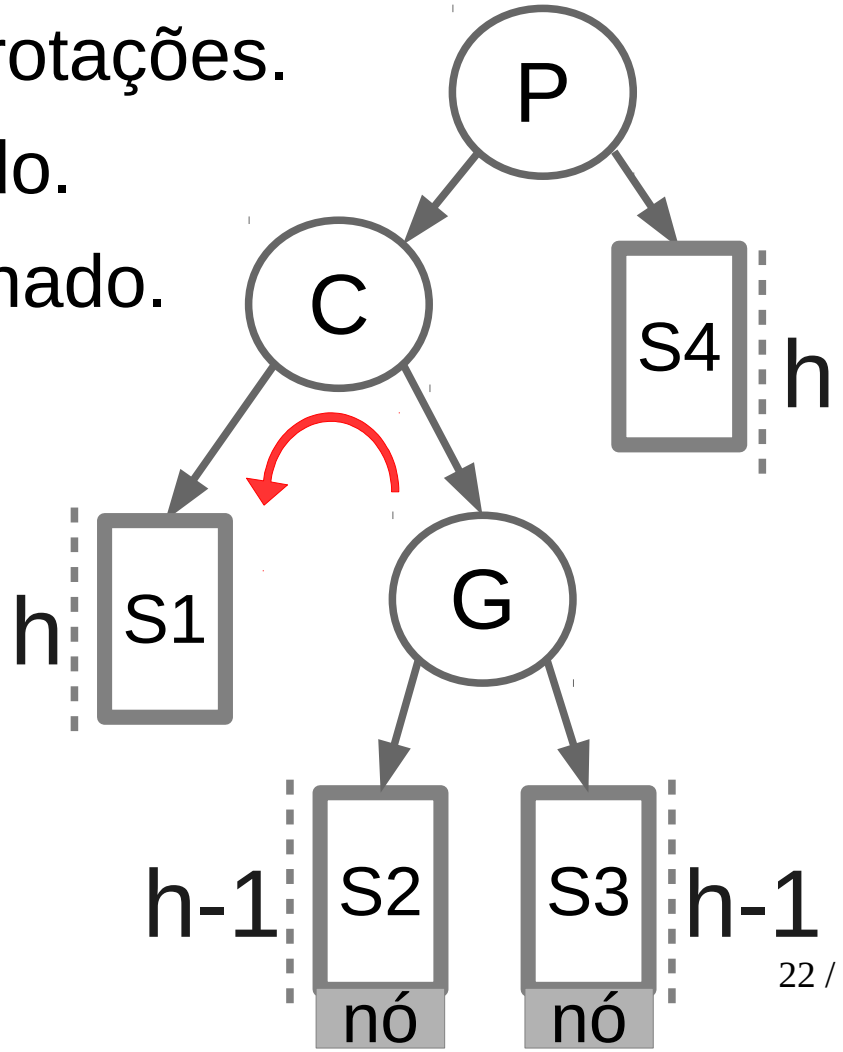
Inserção

- Caso 2
  - Serão necessárias duas rotações.
  - Nó G deve ser rotacionado.



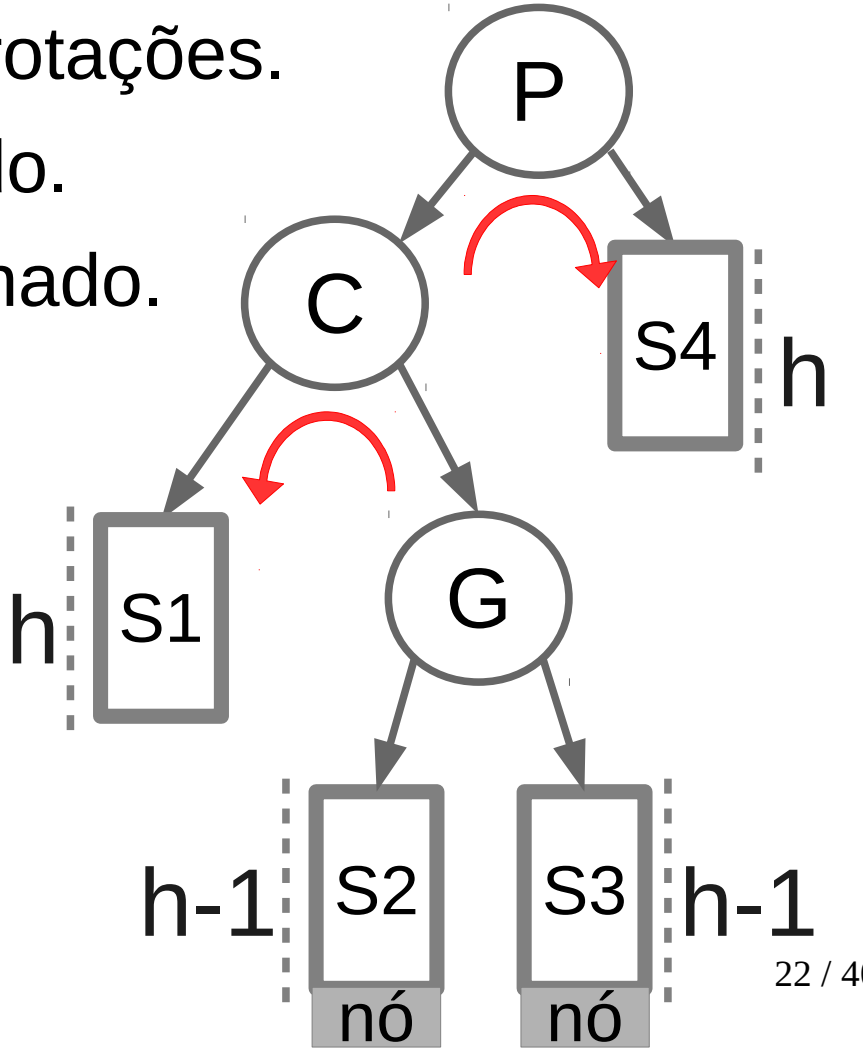
Inserção

- Caso 2
  - Serão necessárias duas rotações.
  - Nó G deve ser rotacionado.
  - O nó *pivot* P será rotacionado.



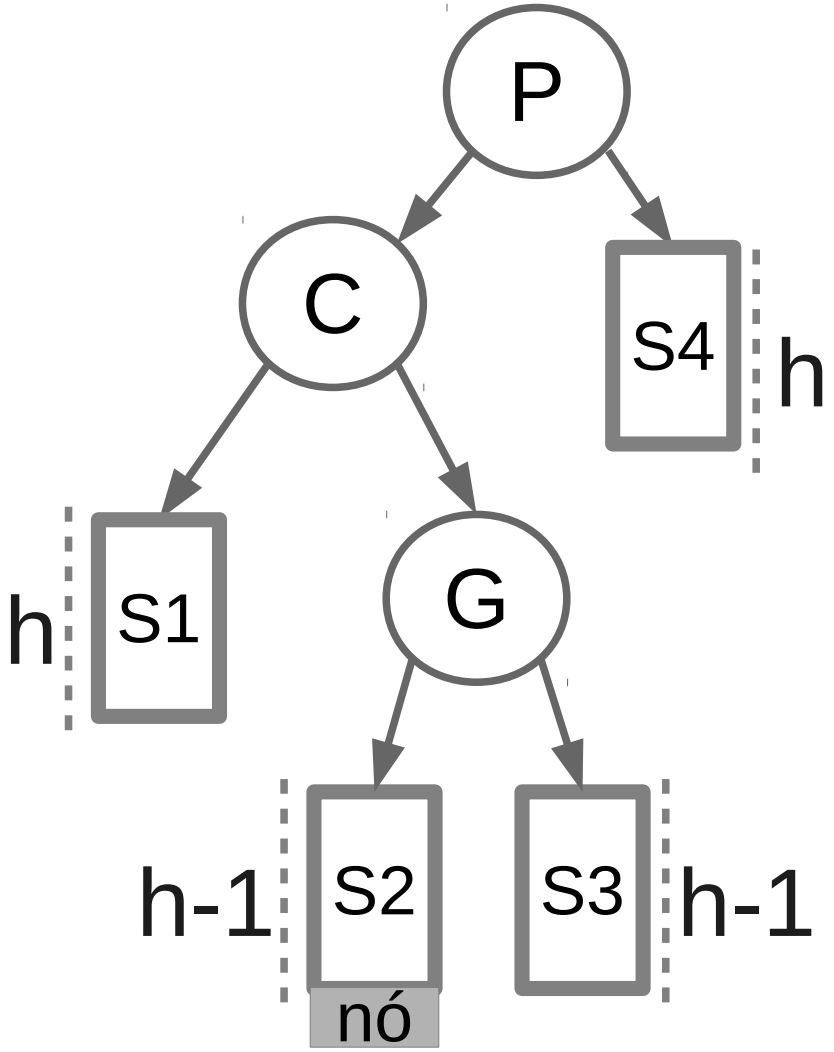
Inserção

- Caso 2
  - Serão necessárias duas rotações.
  - Nó G deve ser rotacionado.
  - O nó *pivot* P será rotacionado.



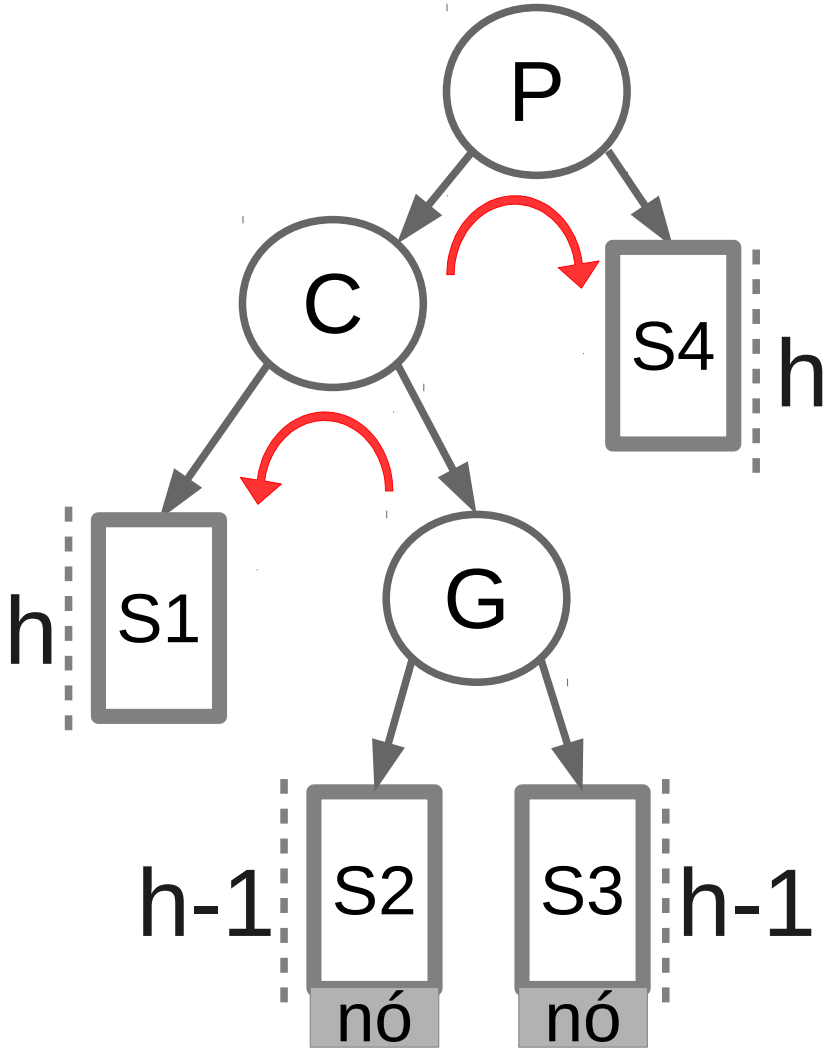
Inserção

- Caso 2



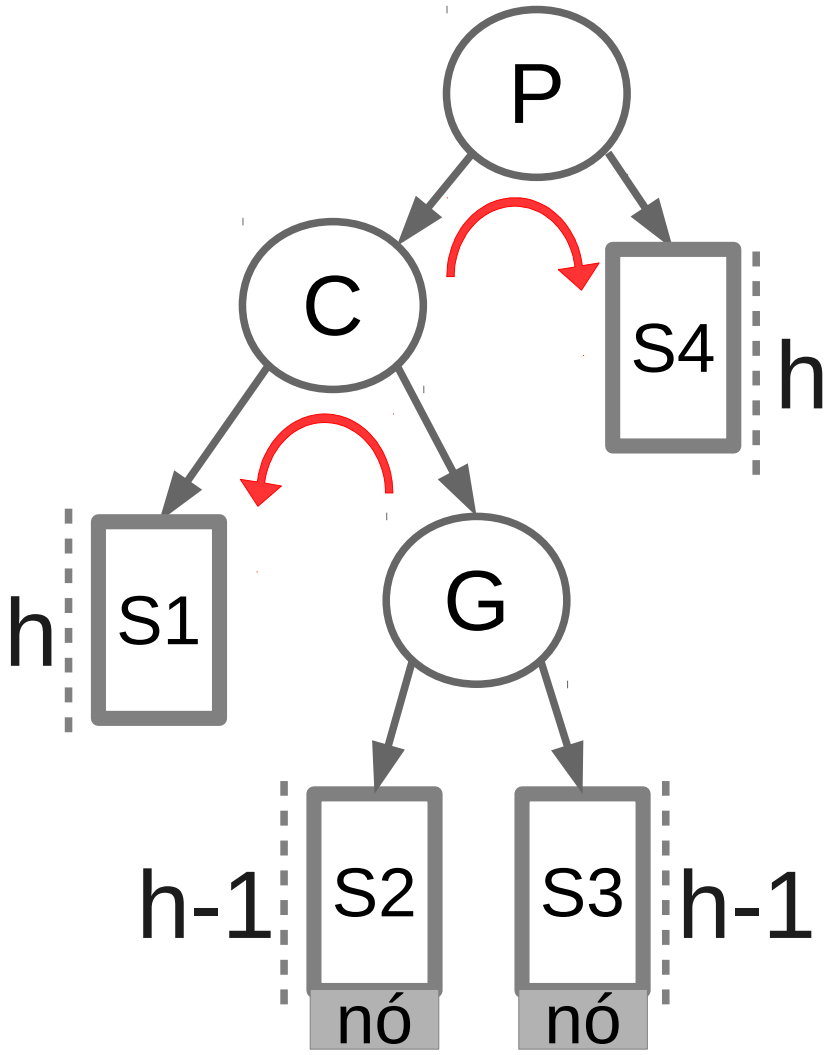
Inserção

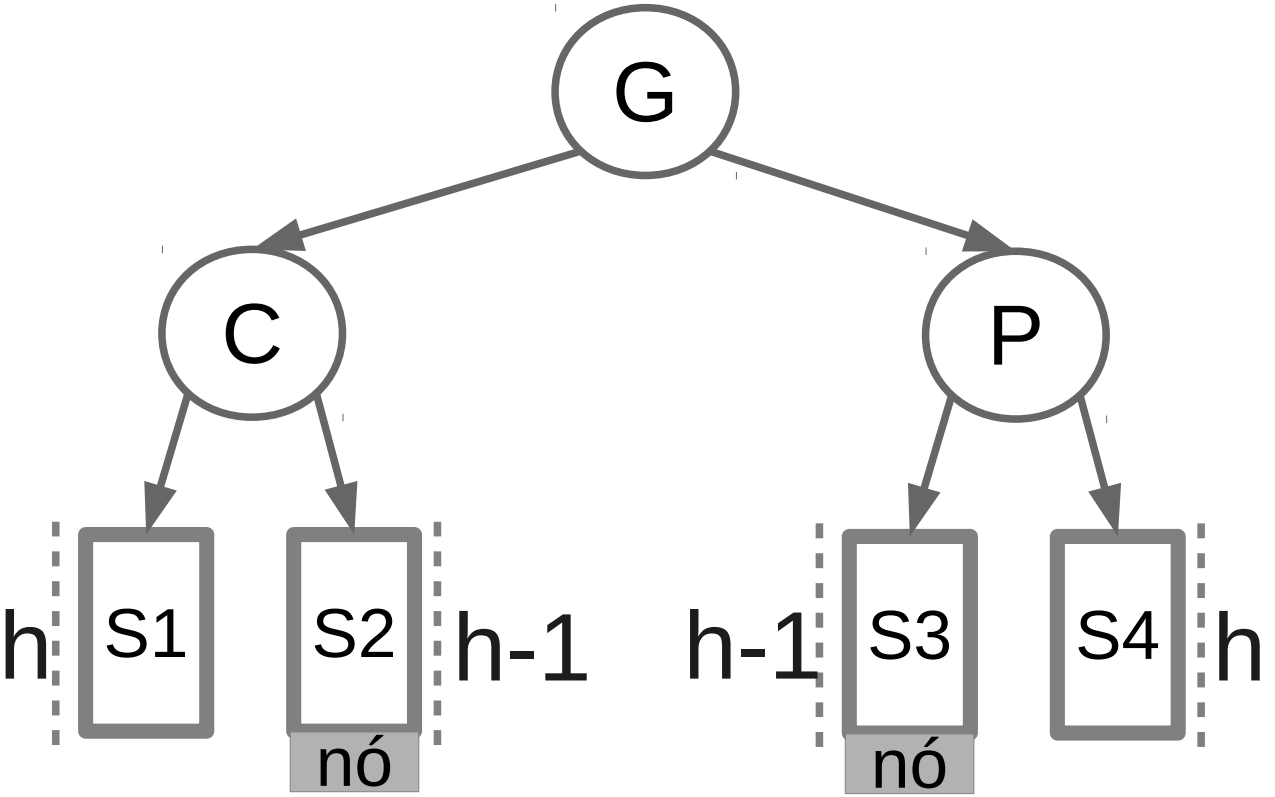
- Caso 2



Inserção

- Caso 2



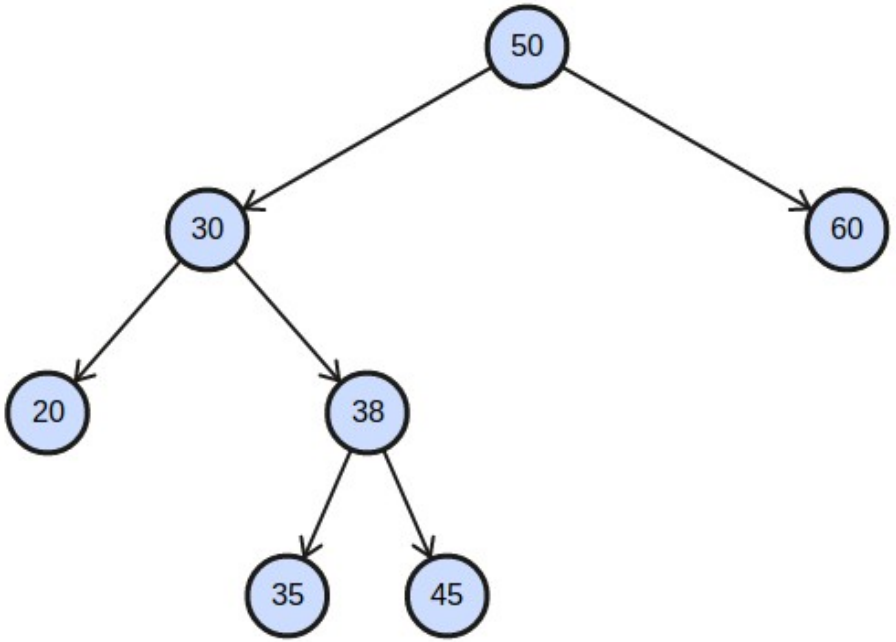


Inserção

- Caso 2

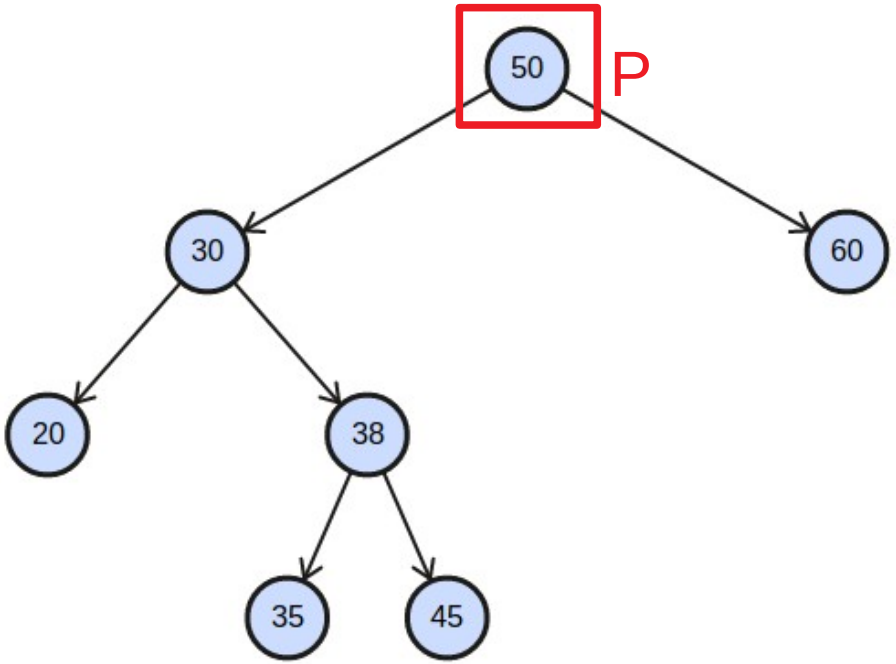
Inserção

- Caso 2



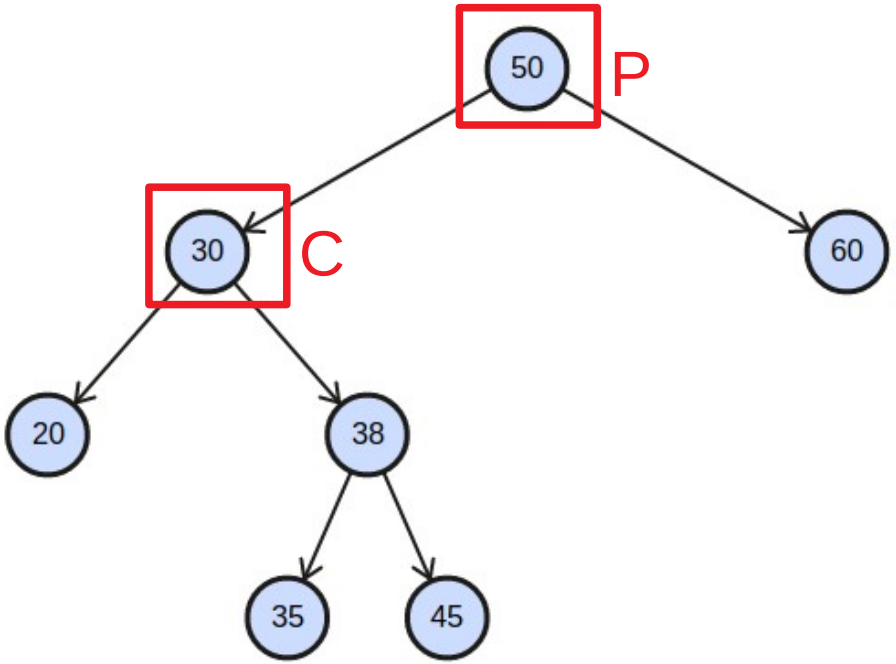
Inserção

- Caso 2



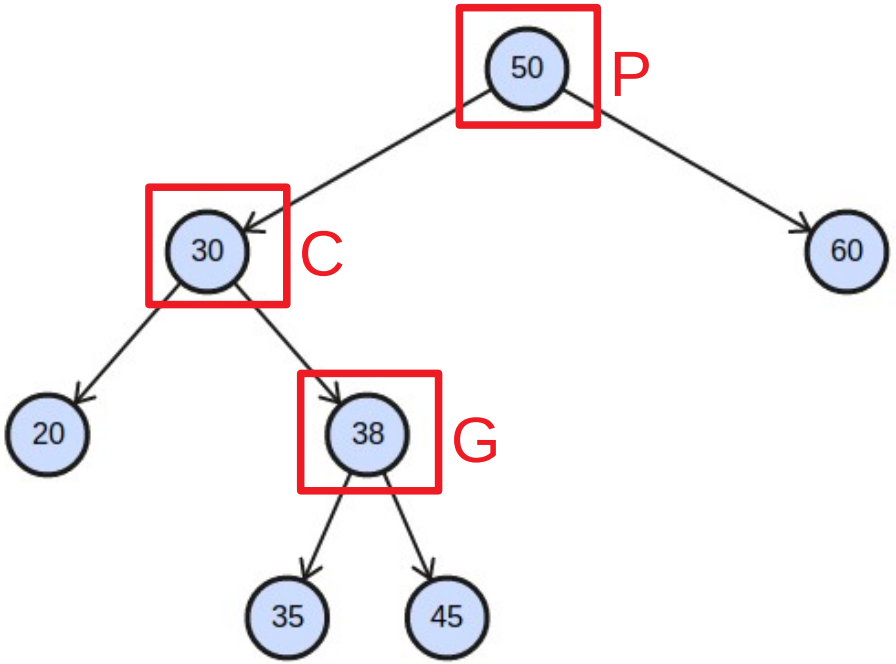
Inserção

- Caso 2



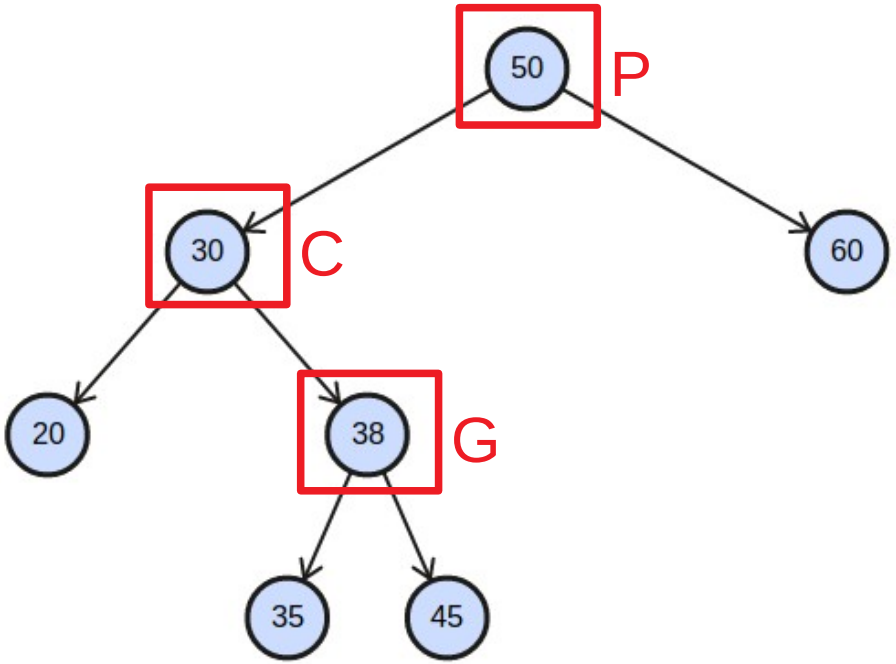
Inserção

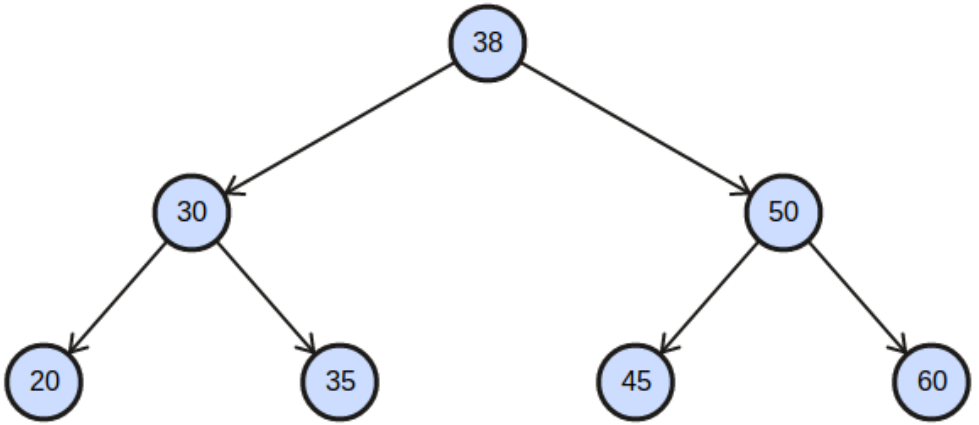
- Caso 2



Inserção

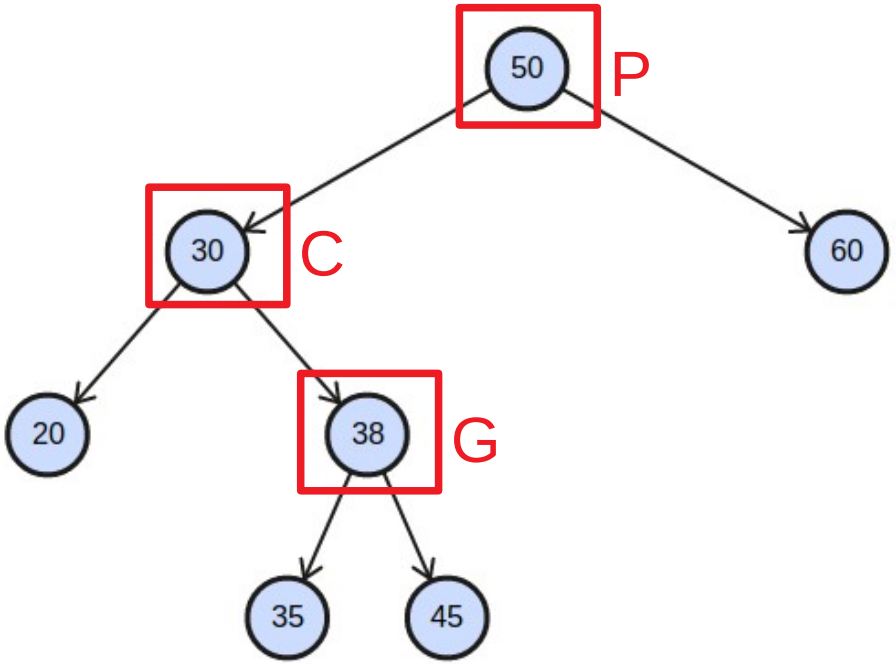
- Caso 2

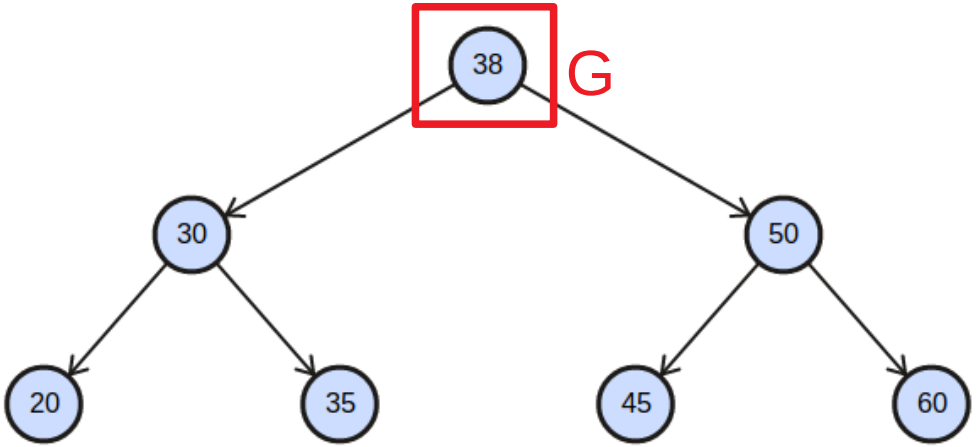




Inserção

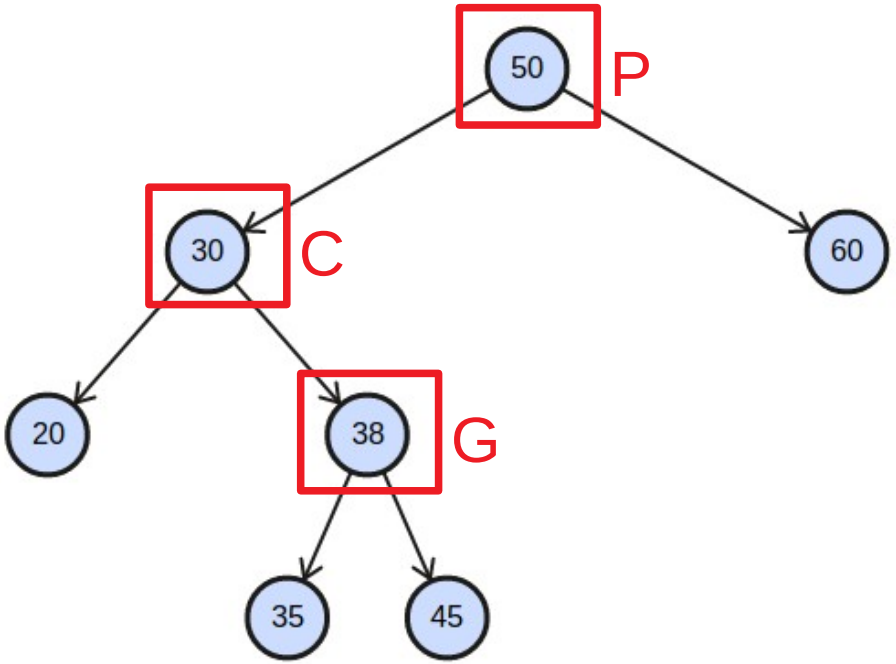
- Caso 2

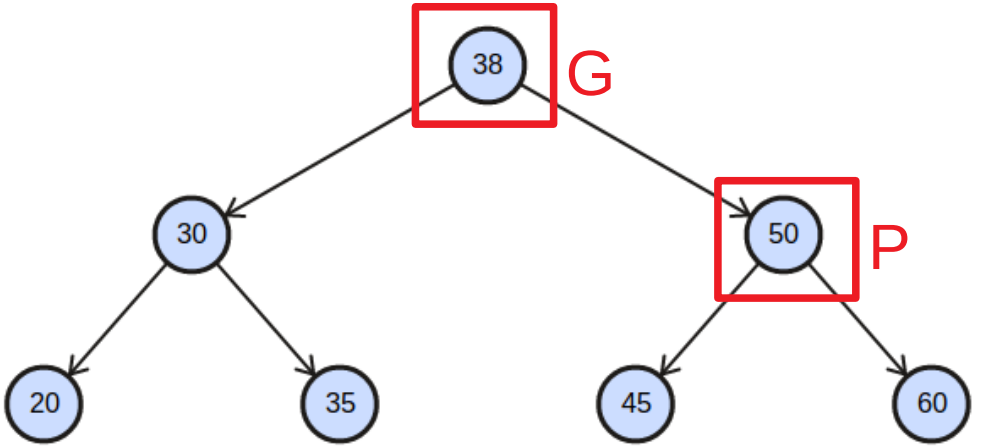




Inserção

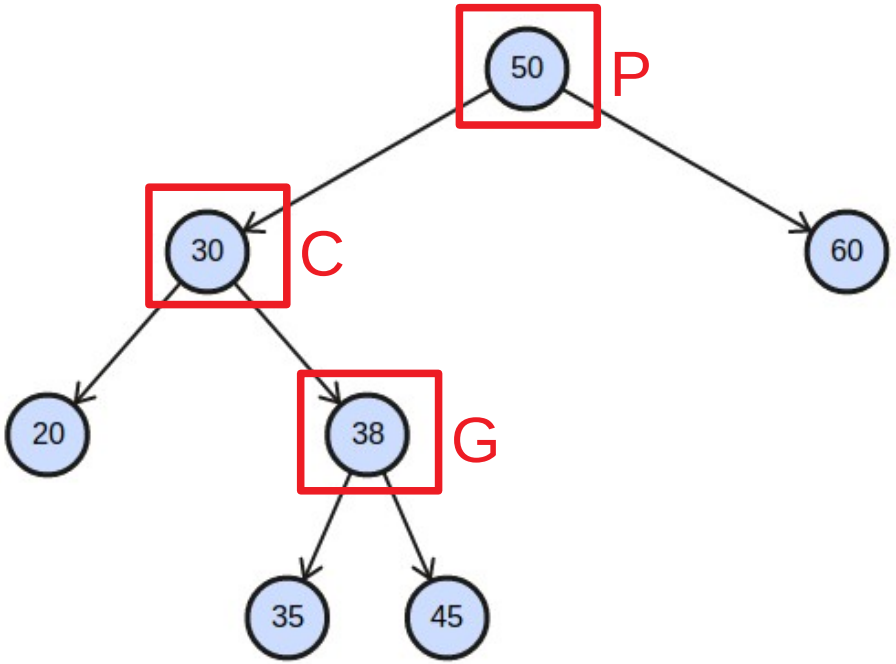
- Caso 2

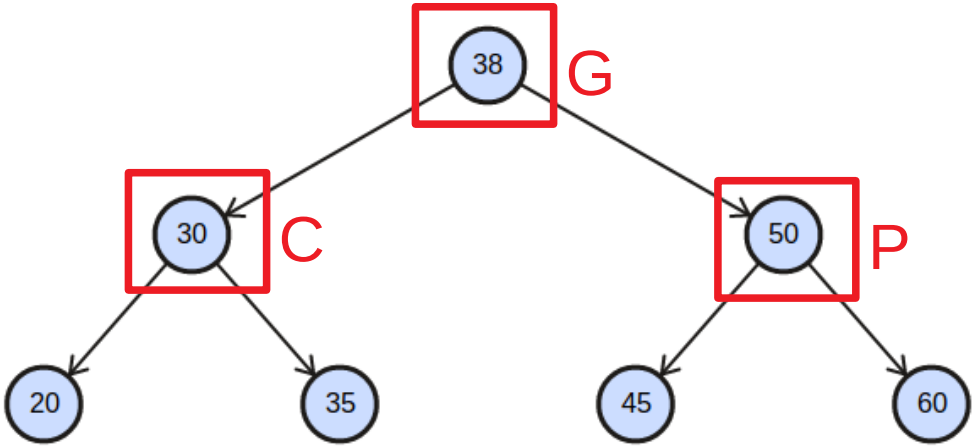




Inserção

- Caso 2





Inserção

- Casos 3 e 4

Inserção

- Casos 3 e 4
  - O caso 3 é um espelho do caso 1.

Inserção

- Casos 3 e 4
  - O caso 3 é um espelho do caso 1.
  - O caso 4 é um espelho do caso 2.

Inserção

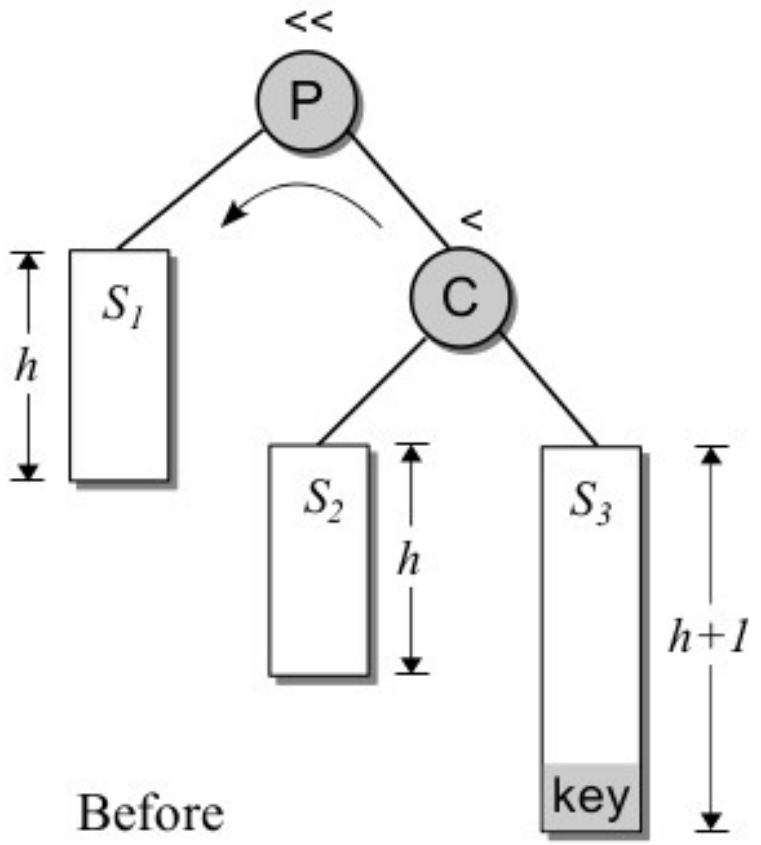
- Casos 3 e 4
  - O caso 3 é um espelho do caso 1.
  - O caso 4 é um espelho do caso 2.
  - A diferença é que o novo elemento é inserido na subárvore à direita do nó *pivot* ou como um descendente da subárvore à direita.

Inserção

- Caso 3

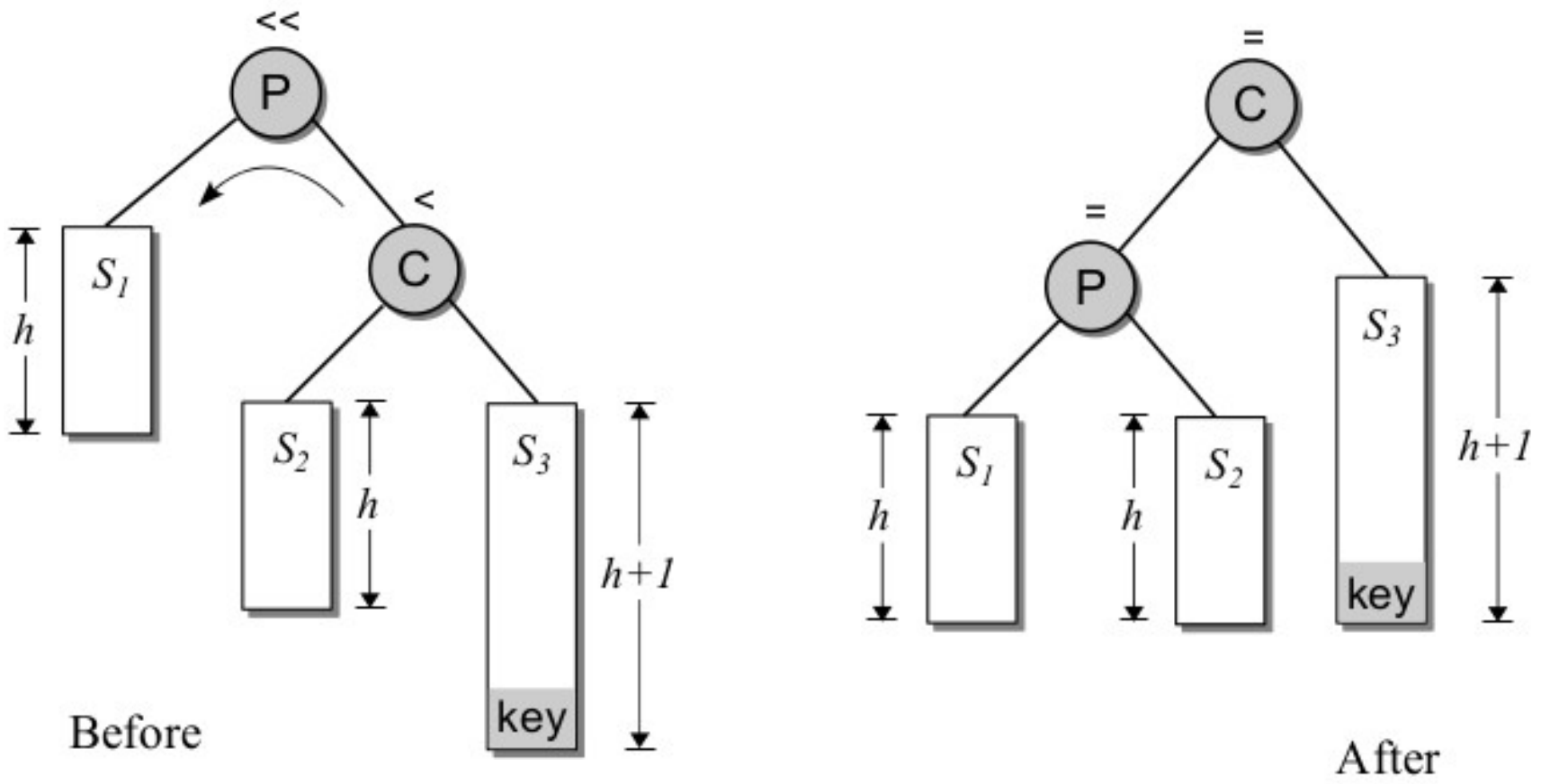
Inserção

- Caso 3



Inserção

- Caso 3

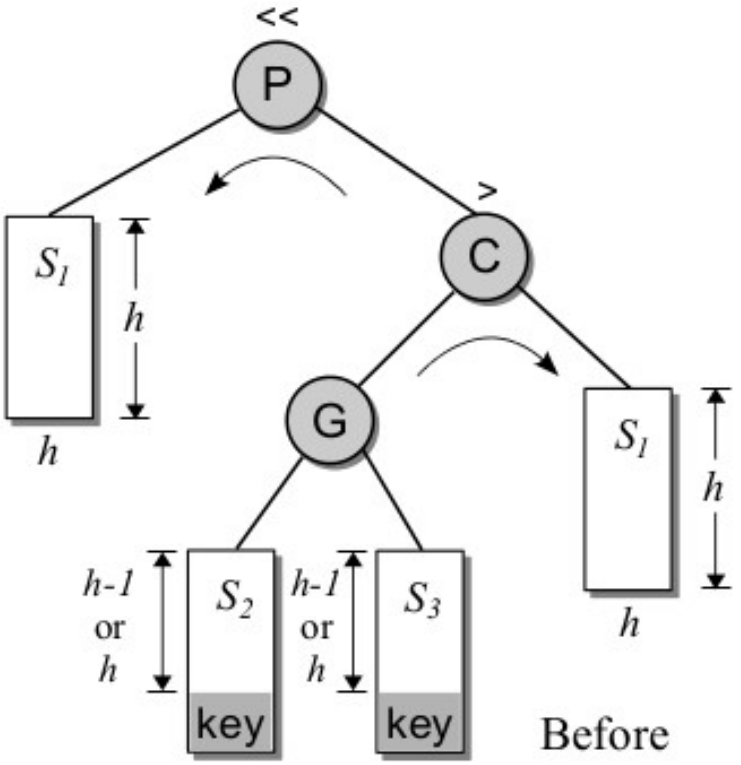


Inserção

- Caso 4

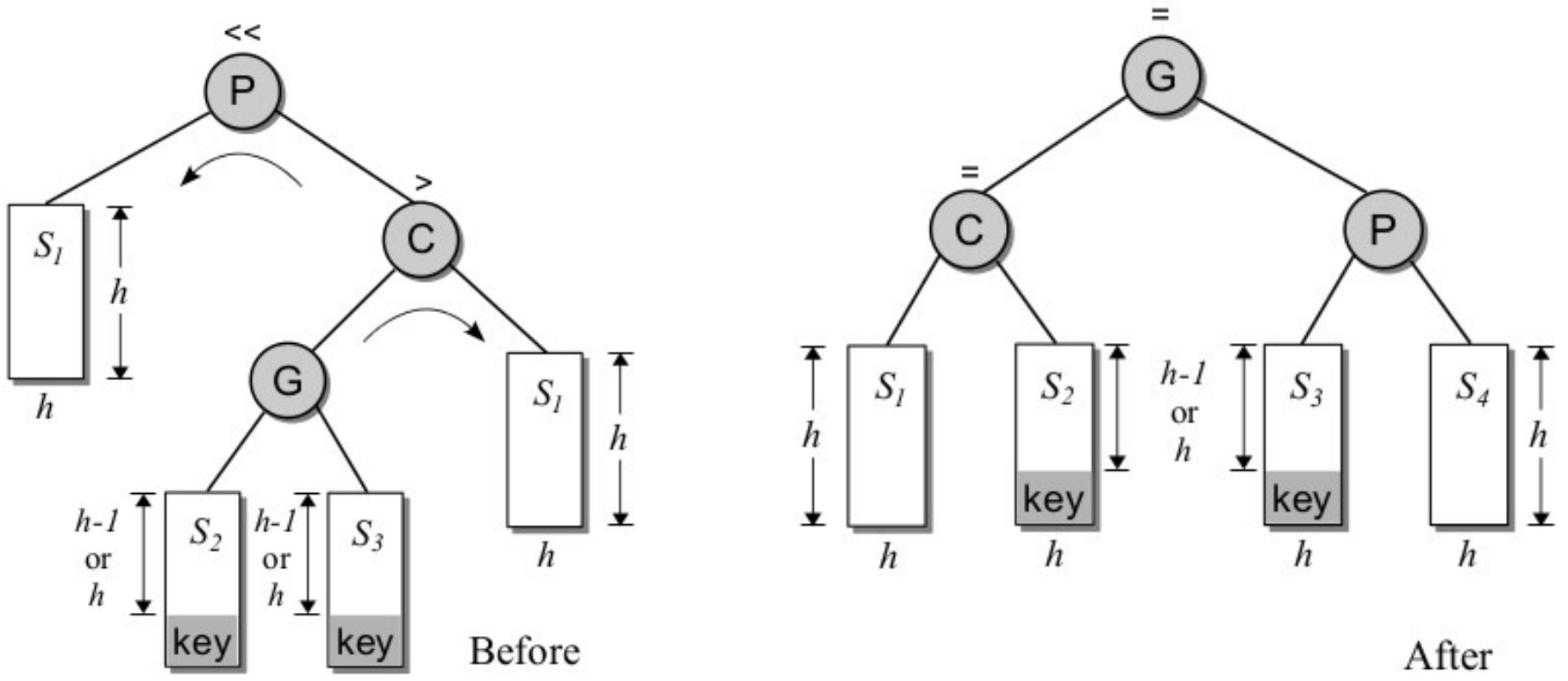
Inserção

- Caso 4



Inserção

- Caso 4

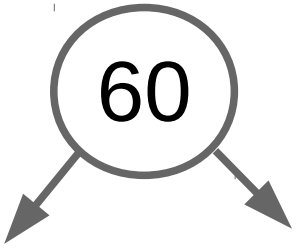


Inserção

- Vamos construir uma AVL com os seguintes elementos: [60, 25, 35, 100, 17, 80]

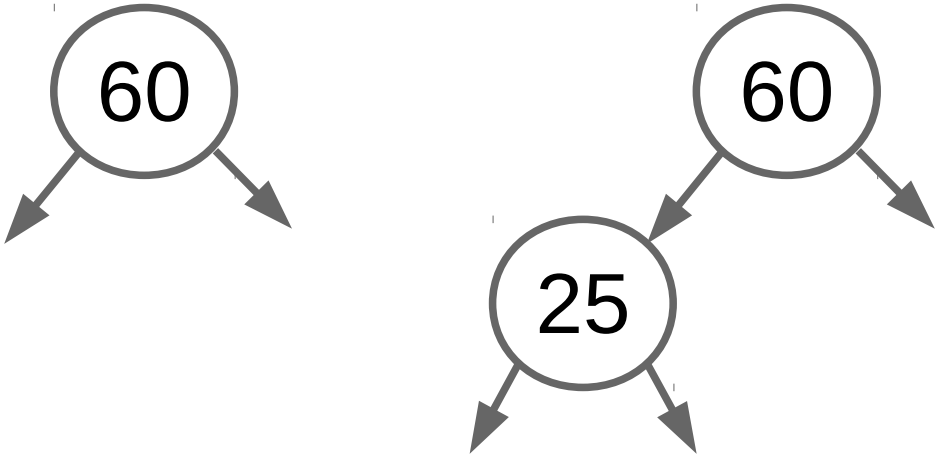
Inserção

- Vamos construir uma AVL com os seguintes elementos: [60, 25, 35, 100, 17, 80]



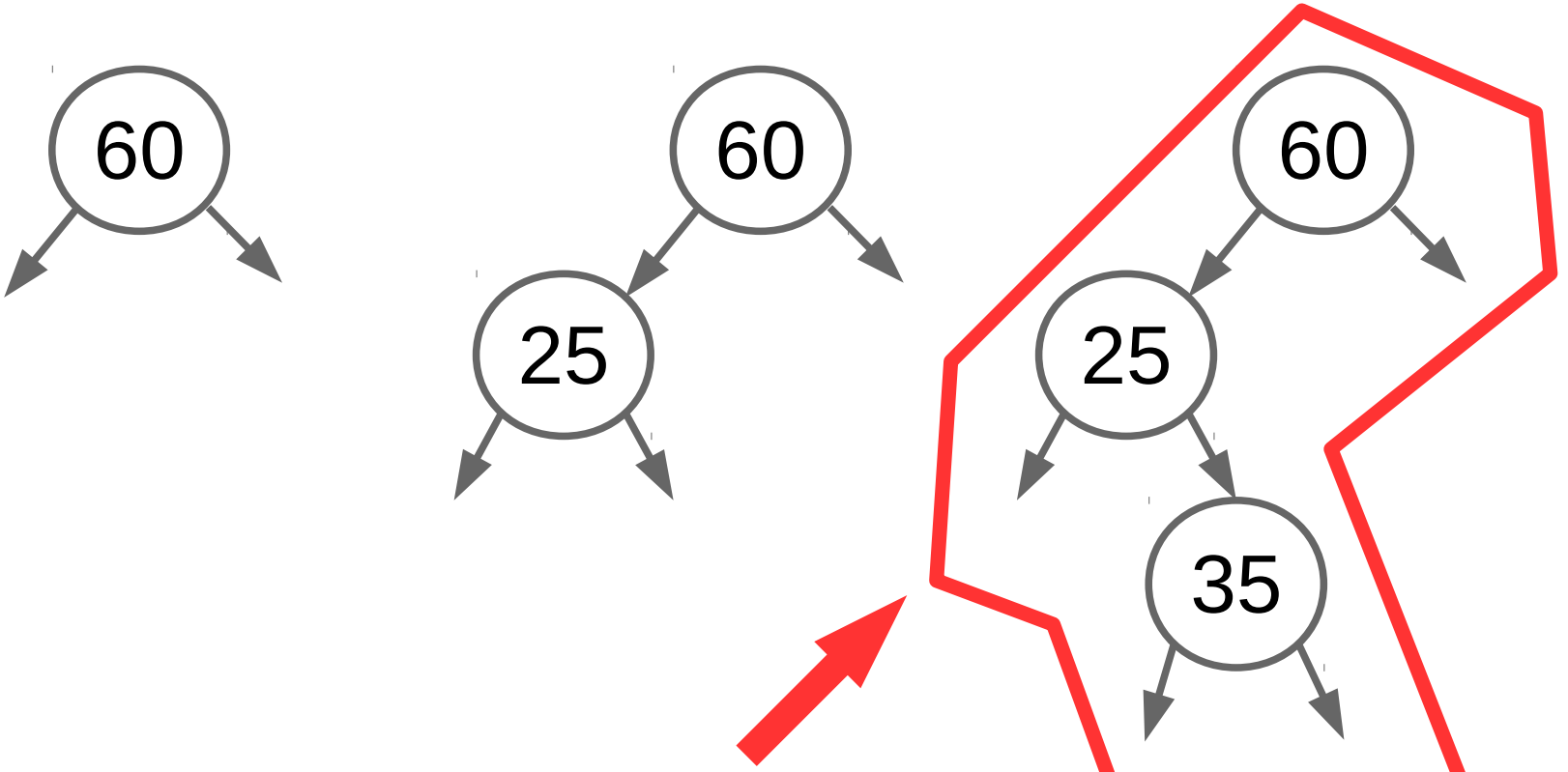
Inserção

- Vamos construir uma AVL com os seguintes elementos: [60, 25, 35, 100, 17, 80]



Inserção

- Vamos construir uma AVL com os seguintes elementos: [60, 25, 35, 100, 17, 80]

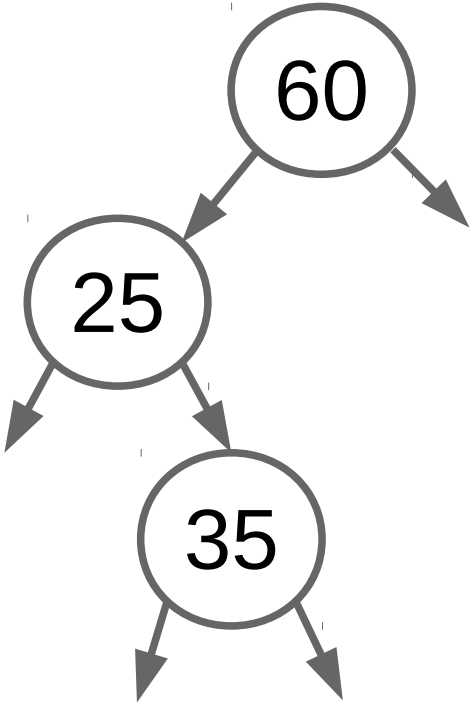


Inserção

- Rotacionamento

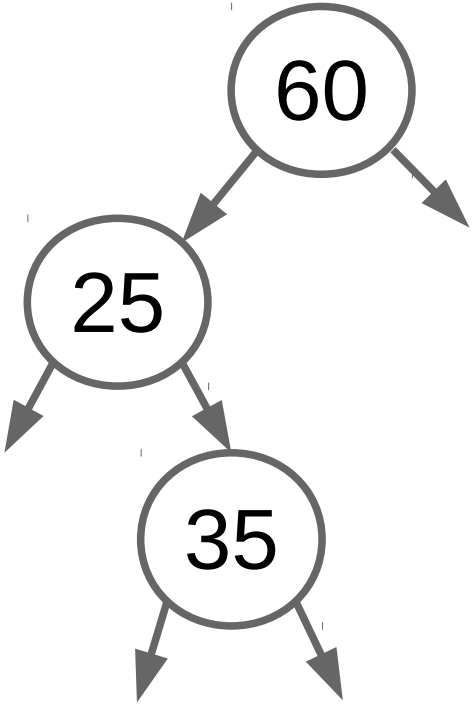
Inserção

- Rotacionamento



Inserção

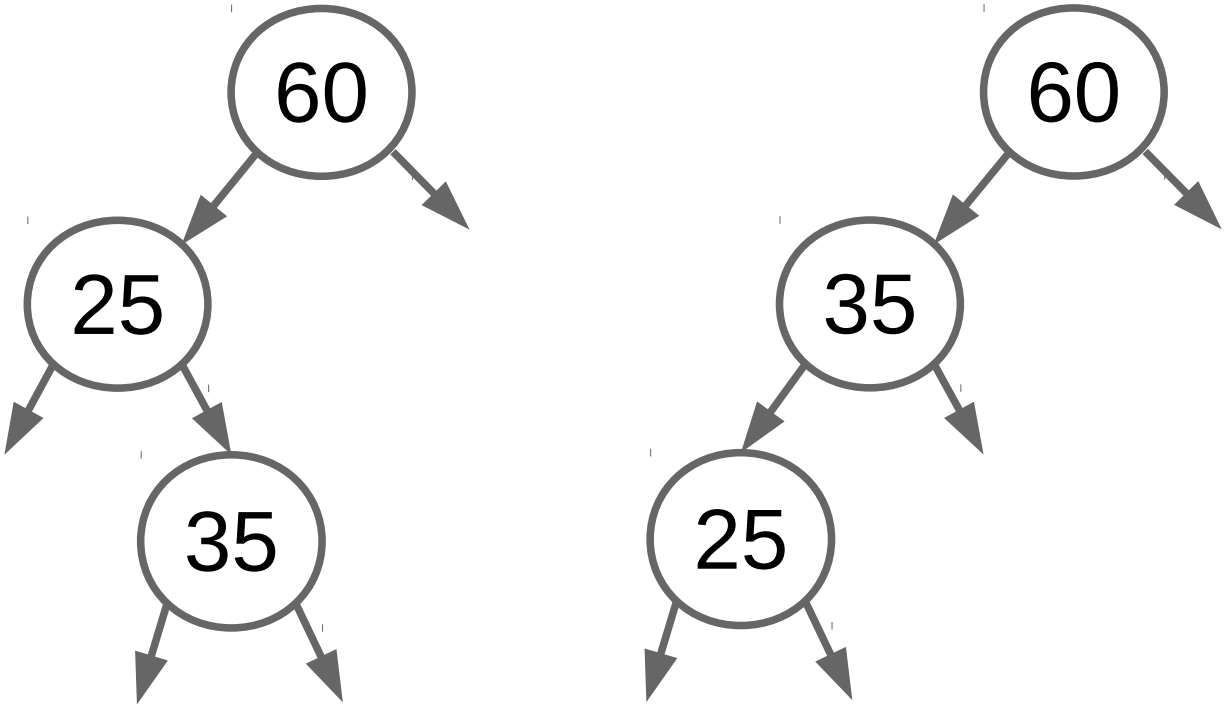
- Rotacionamento PASSO 1



Inserção

- Rotacionamento

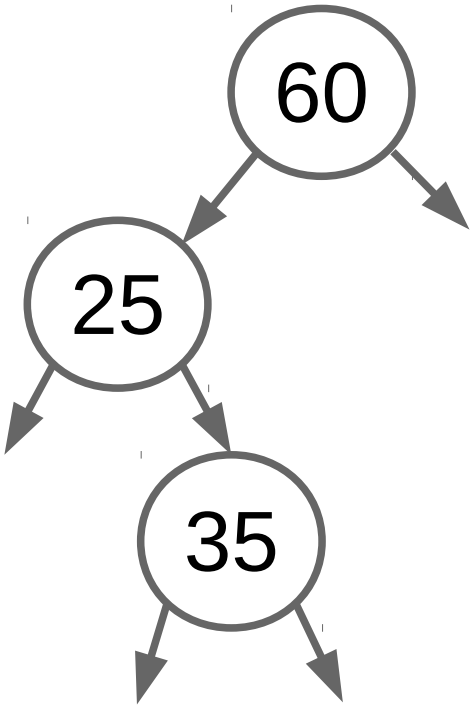
PASSO 1



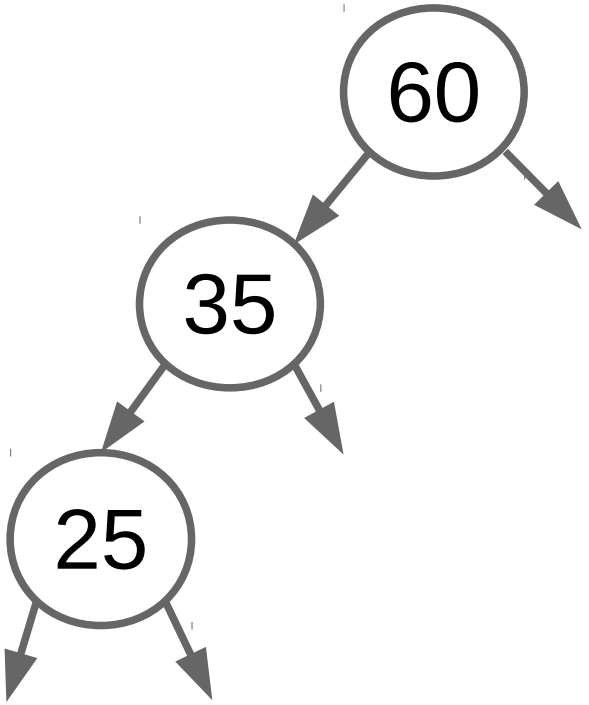
Inserção

- Rotacionamento

PASSO 1

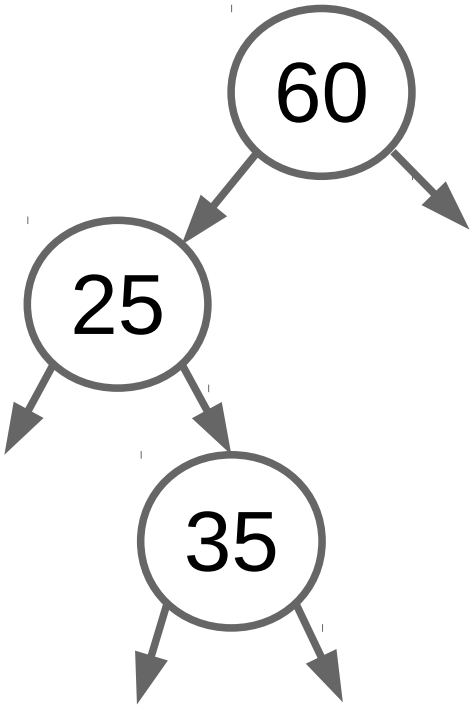


PASSO 2

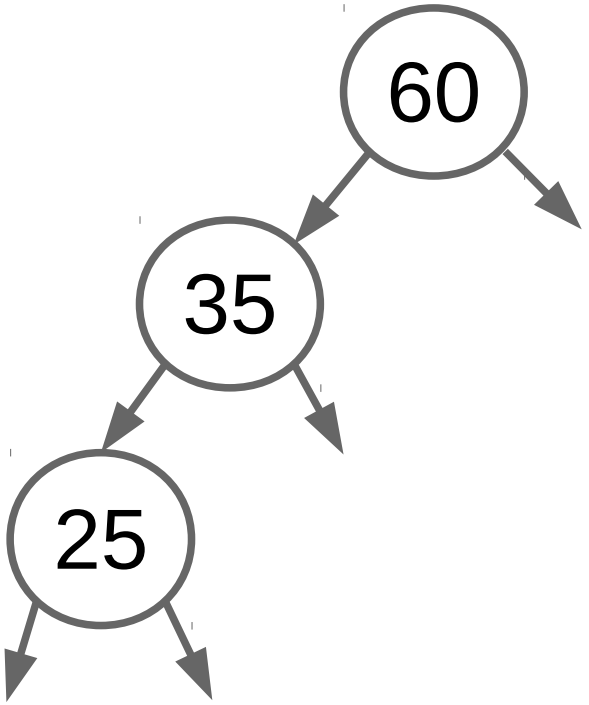


Inserção

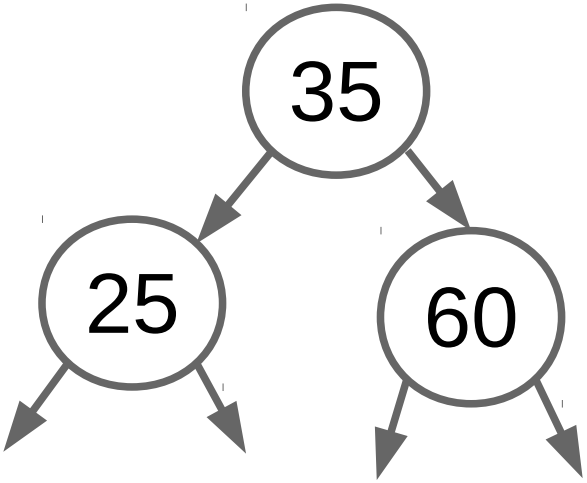
- Rotacionamento



PASSO 1



PASSO 2

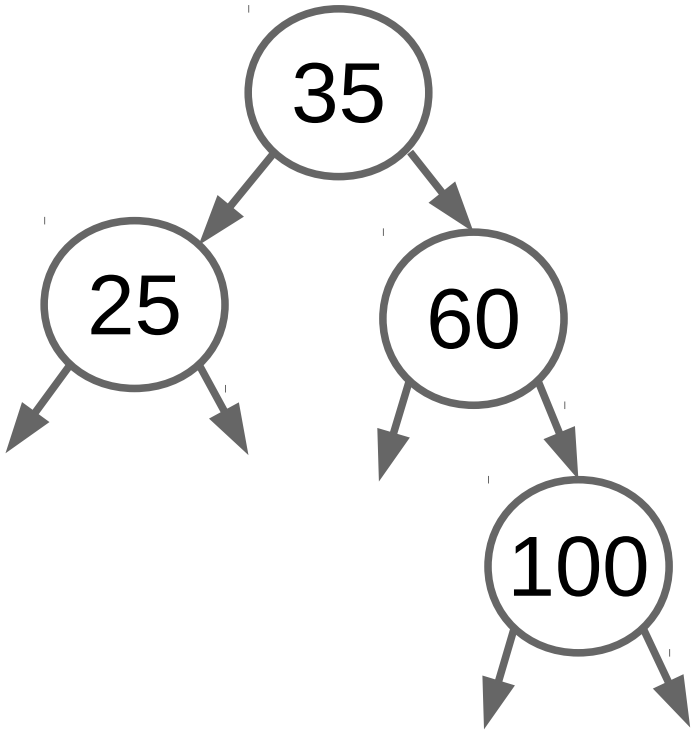


Inserção

- Continuando a inserçao [..., 100, 17, 80]

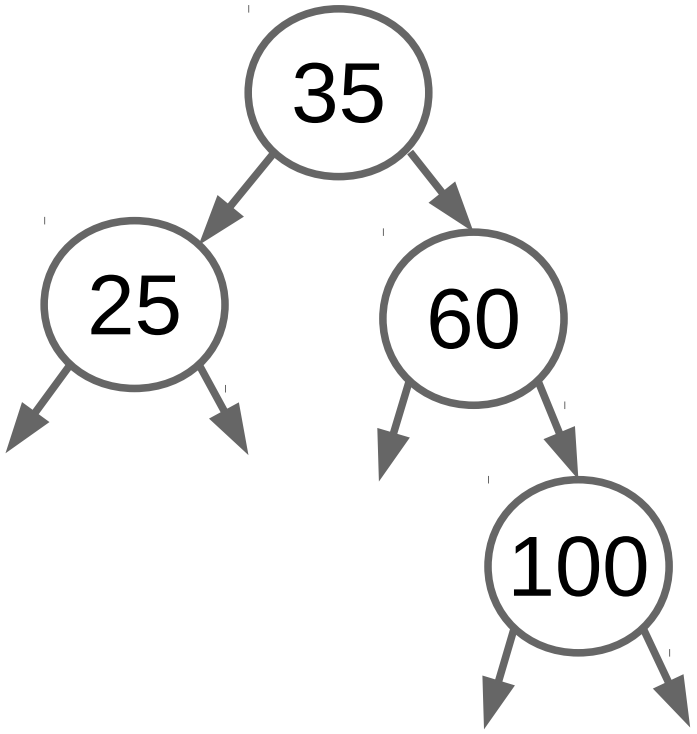
Inserção

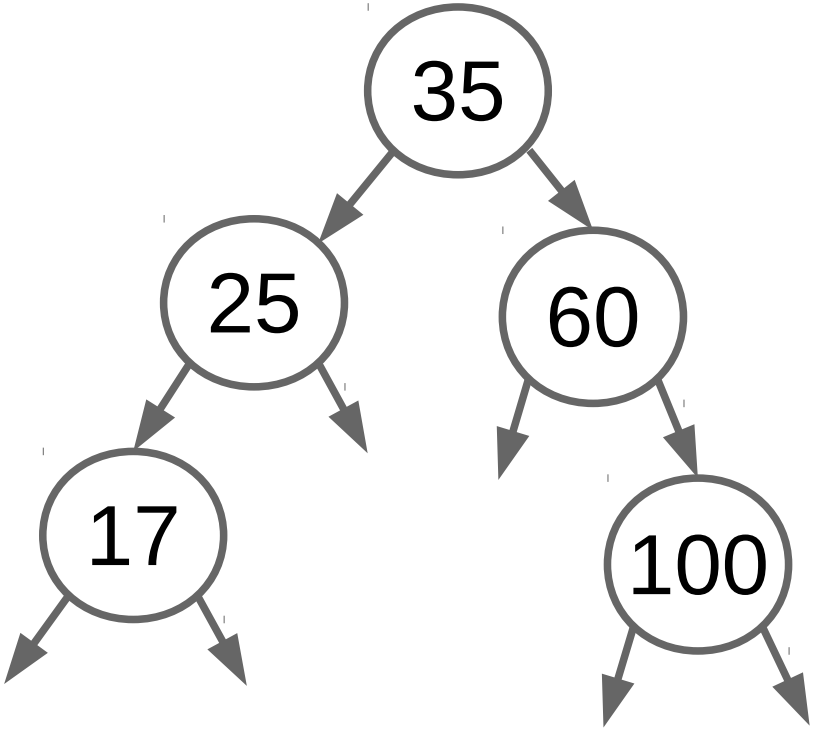
- Continuando a inserçao [..., 100, 17, 80]



Inserção

- Continuando a inserçao [..., 100, 17, 80]



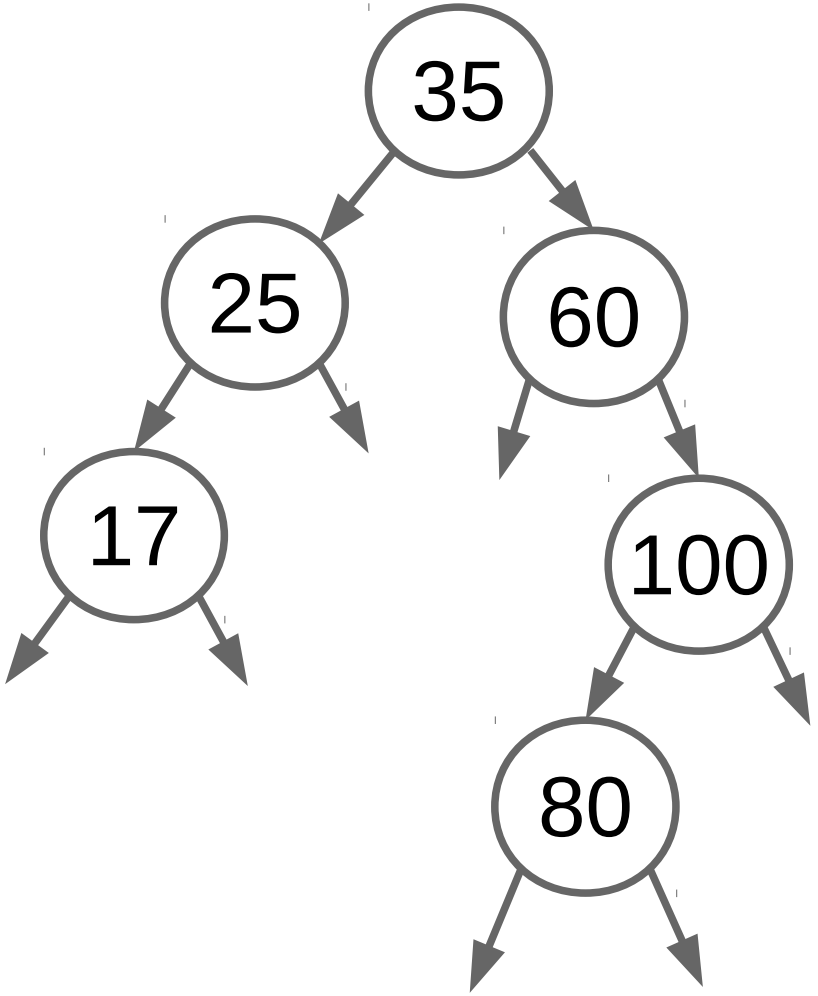


Inserção

- Continuando a inserção [..., 80]

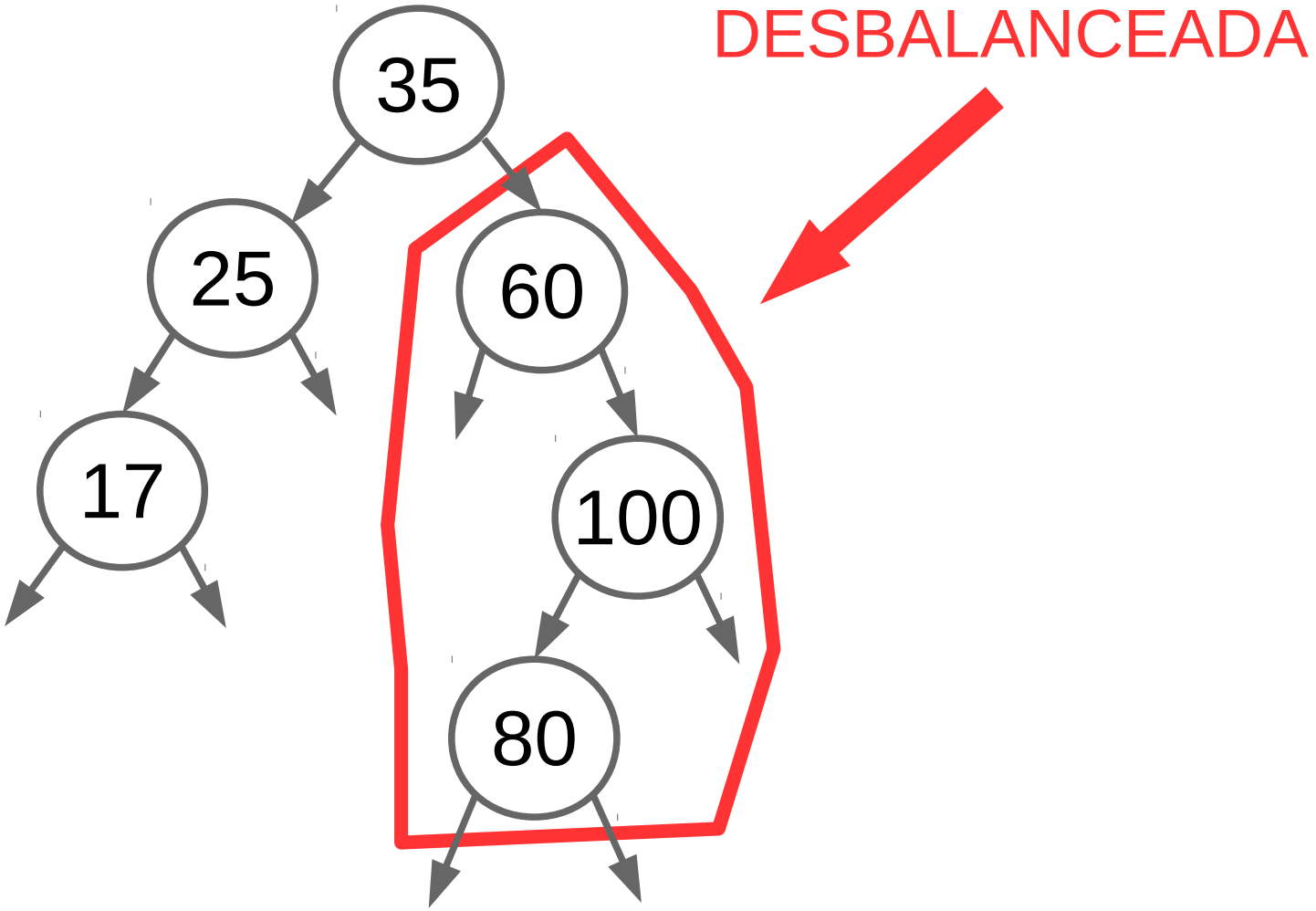
Inserção

- Continuando a inserção [..., 80]



Inserção

- Continuando a inserção [..., 80]

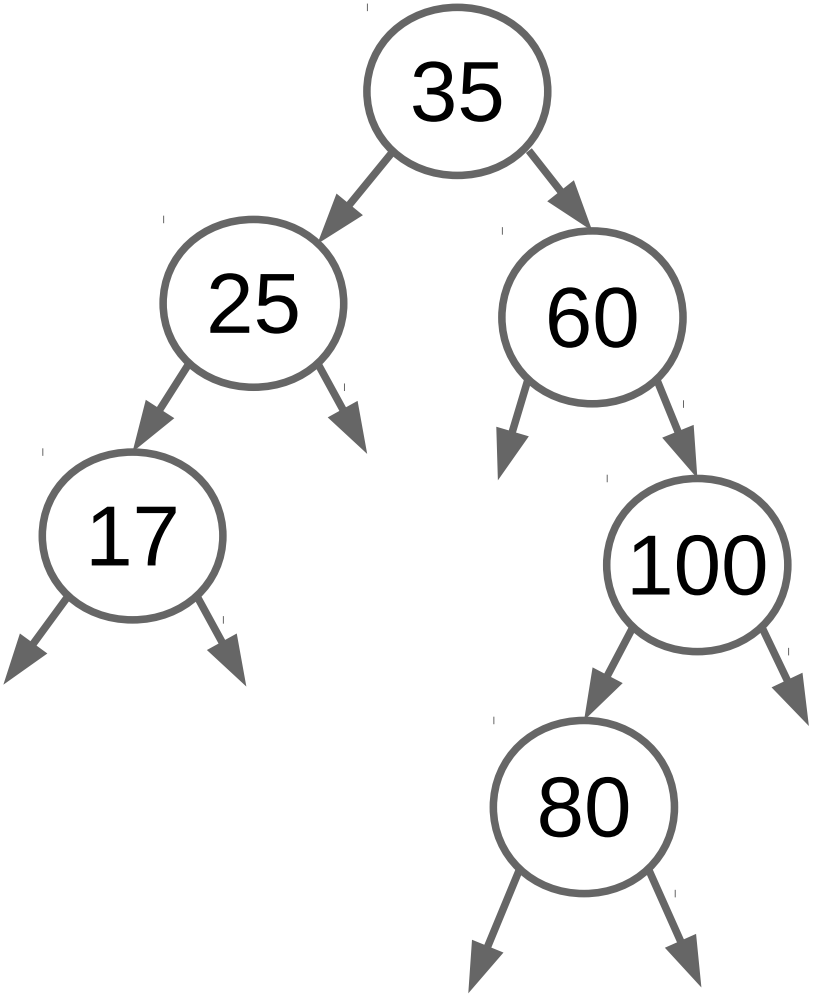


Inserção

- Rotacionamento

Inserção

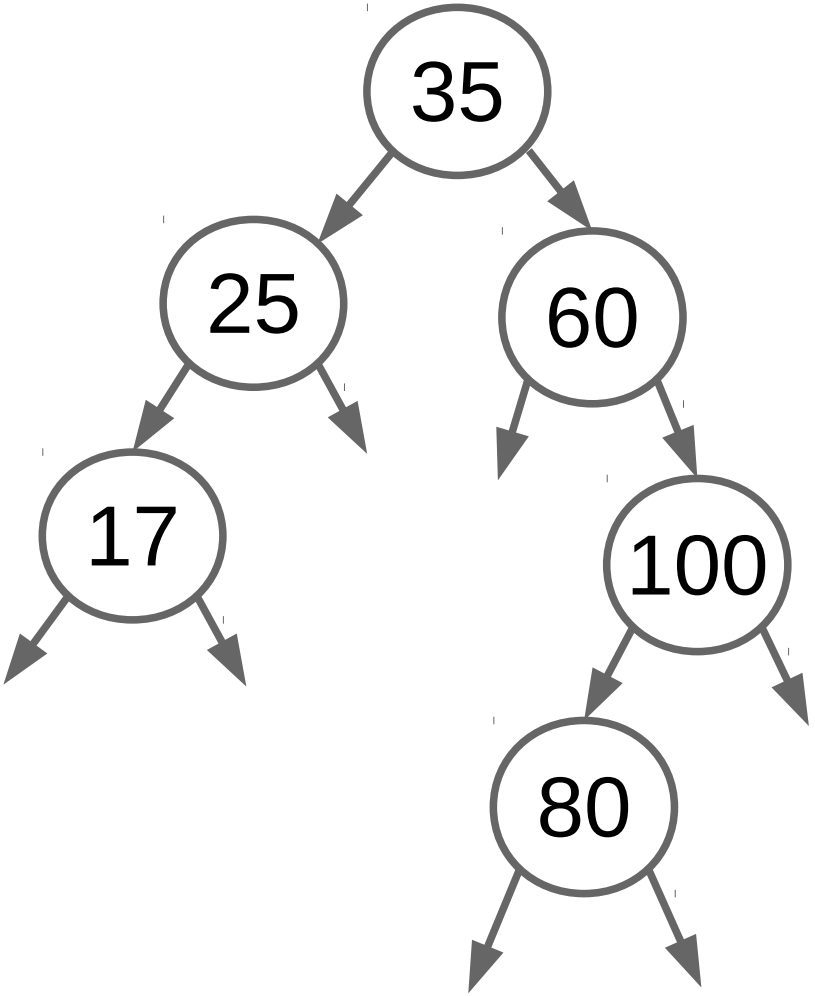
- Rotacionamento



Inserção

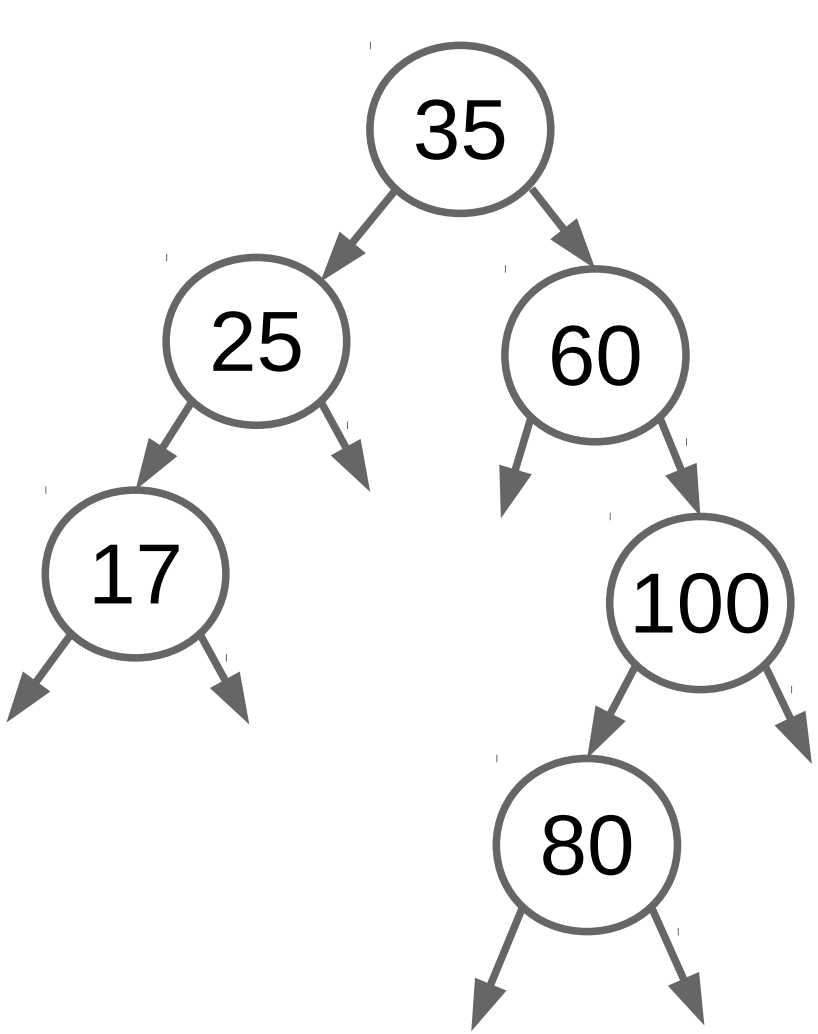
- Rotacionamento

PASSO 1

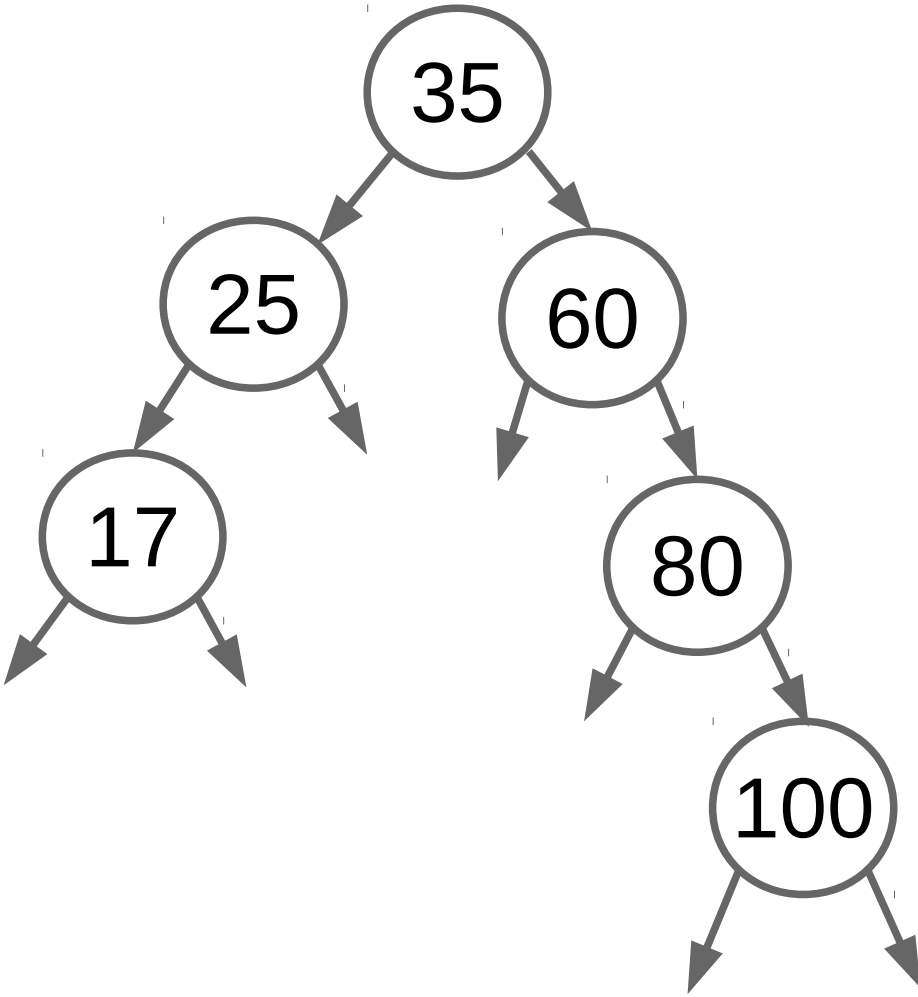


Inserção

- Rotacionamento



PASSO 1

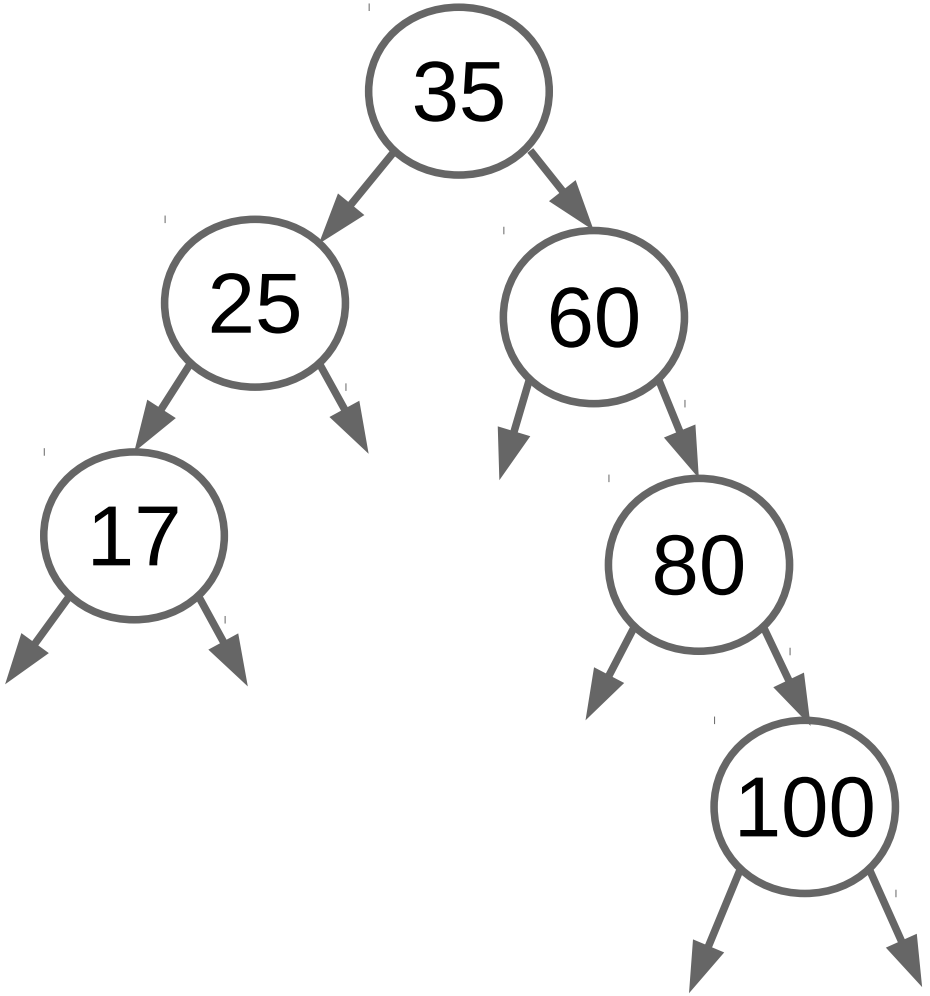


Inserção

- Rotacionamento

Inserção

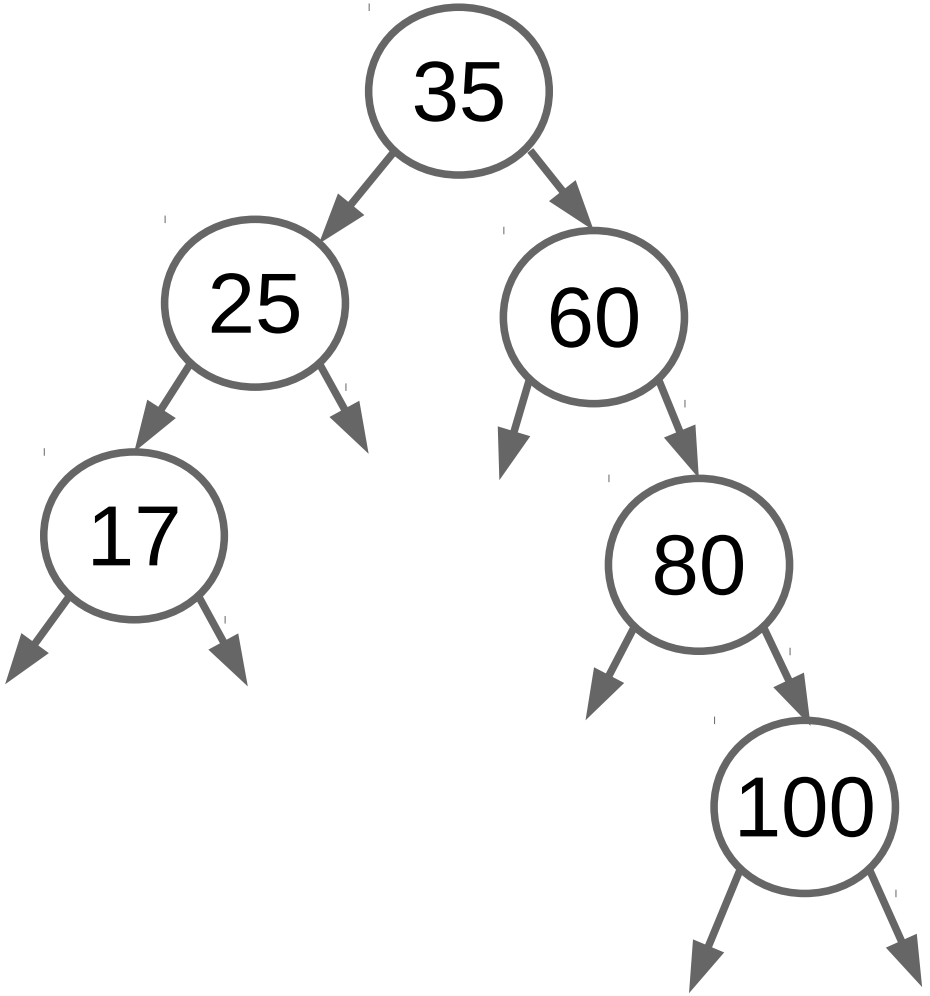
- Rotacionamento



Inserção

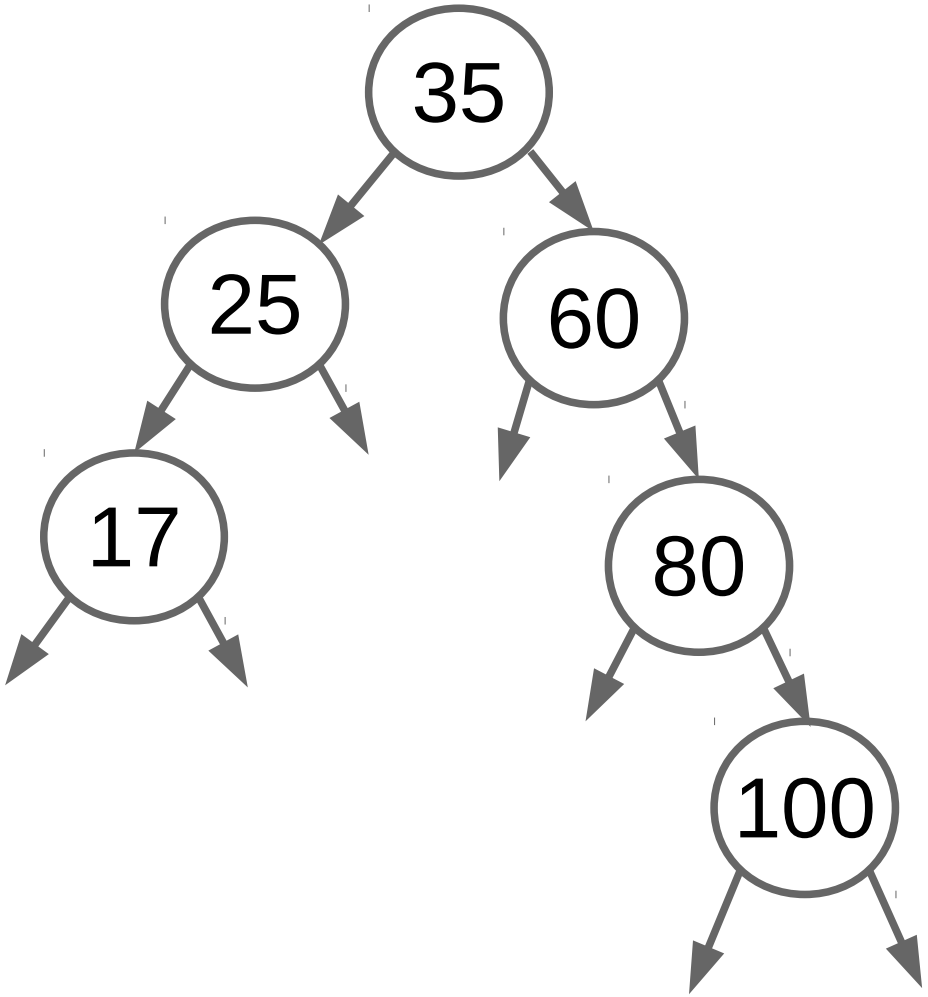
- Rotacionamento

PASSO 2

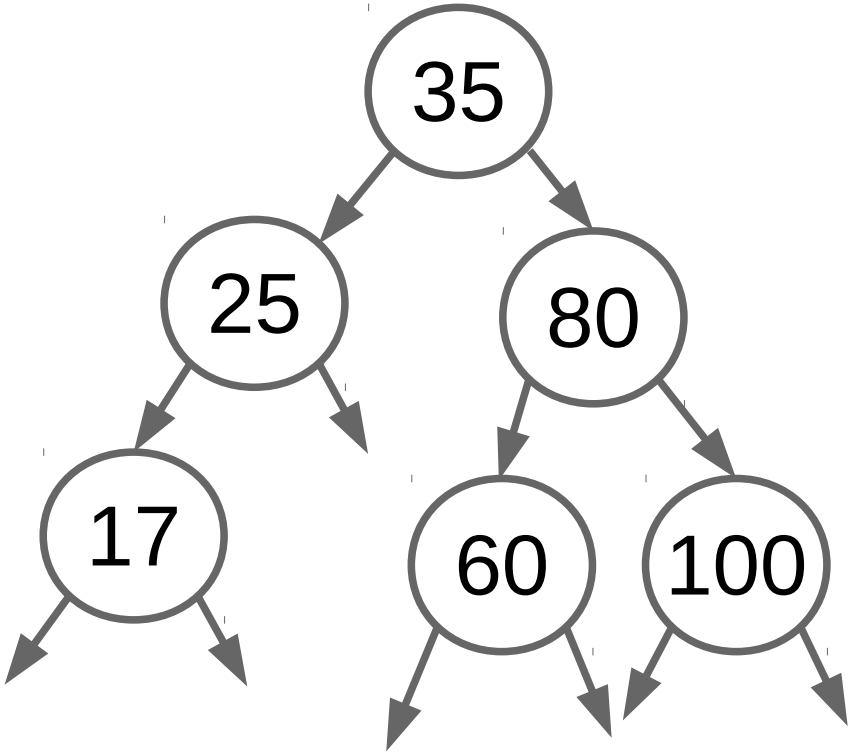


Inserção

- Rotacionamento



PASSO 2



Remoção

Remoção

- Quando um elemento é removido de uma AVL, nós devemos garantir que a propriedade de balanceamento seja mantida.

Remoção

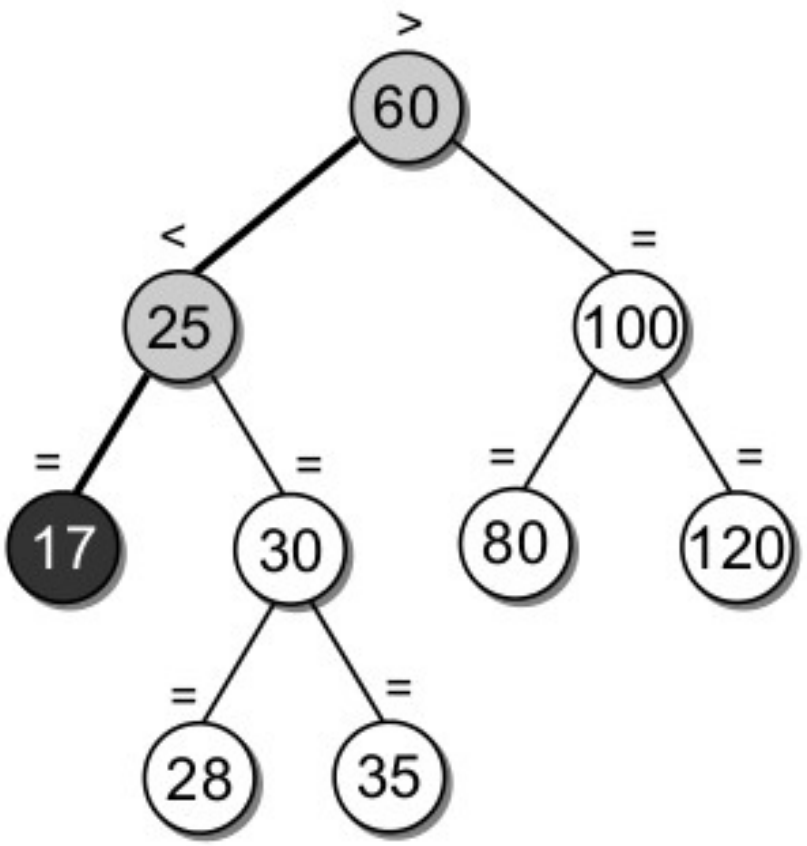
- Quando um elemento é removido de uma AVL, nós devemos garantir que a propriedade de balanceamento seja mantida.
- Depois de remover o elemento desejado, as subárvores devem ser rebalanceadas.

Remoção

- Suponha que vamos remover o nó 17 da AVL abaixo:

Remoção

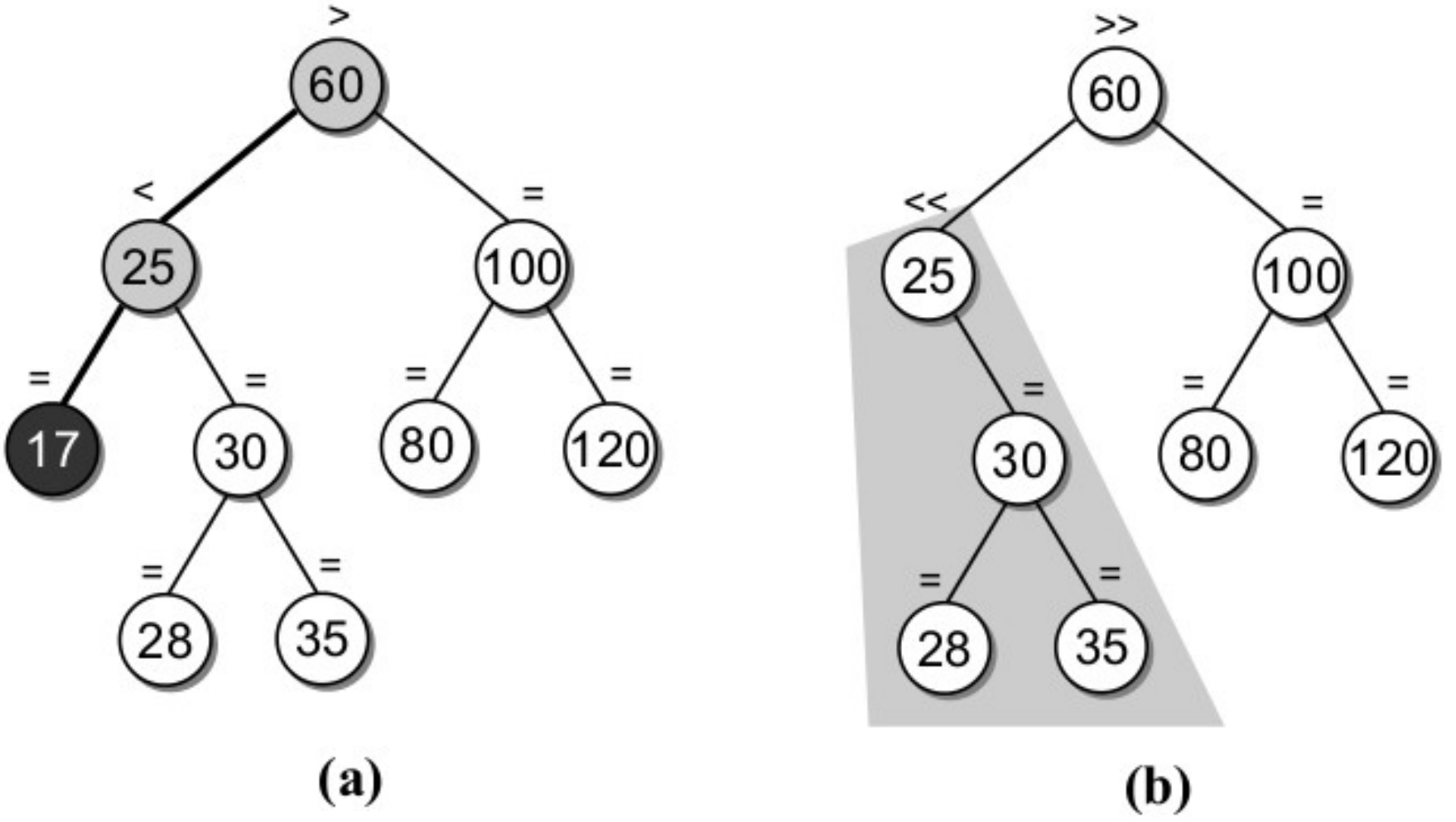
- Suponha que vamos remover o nó 17 da AVL abaixo:



(a)

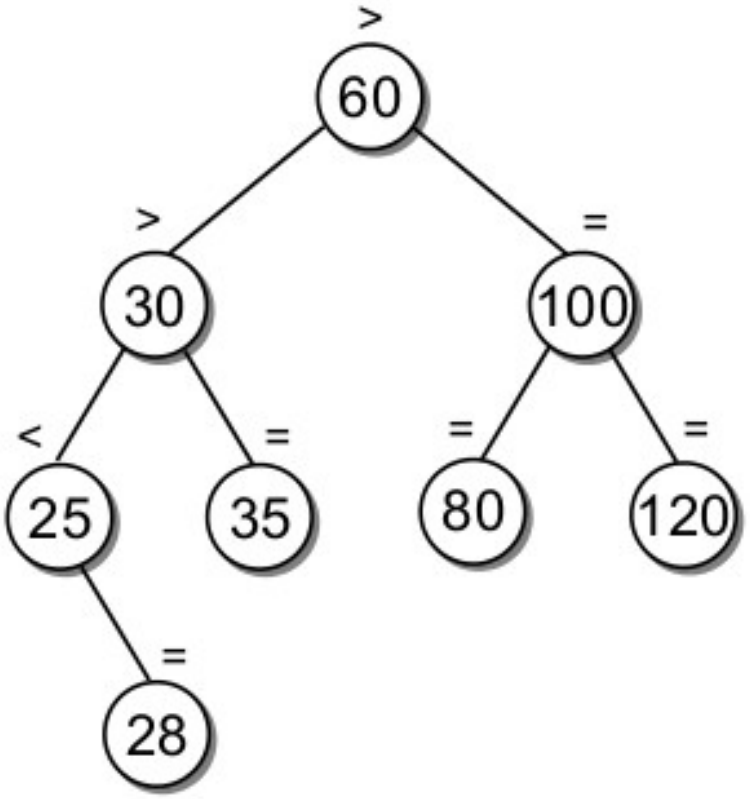
Remoção

- Suponha que vamos remover o nó 17 da AVL abaixo:



Remoção

- Necessária uma rotação no nó pivot 25.



(c)

Remoção

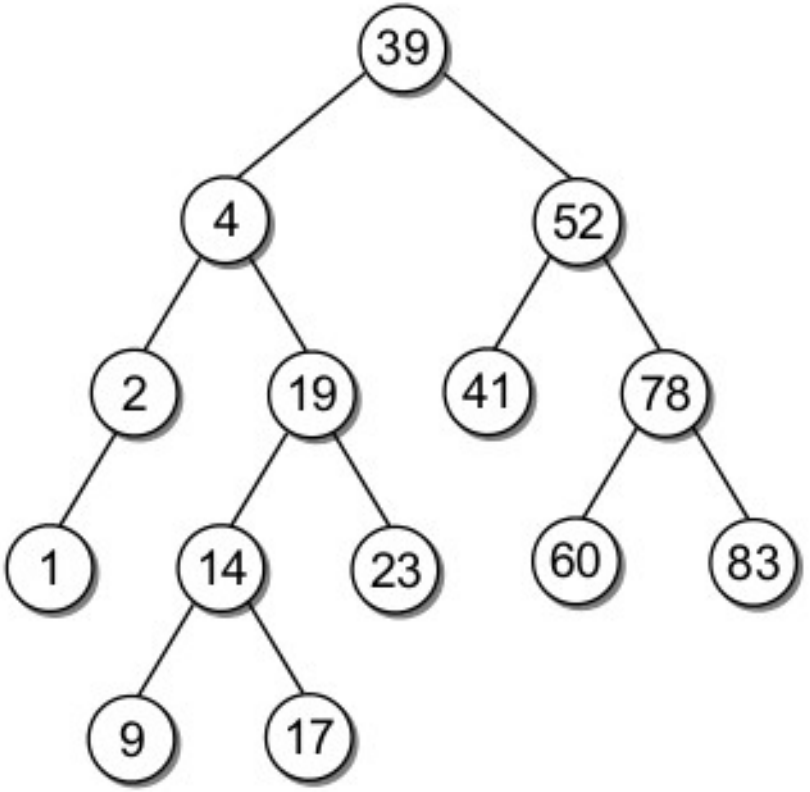
- Quando uma subárvore é rebalanceada devido a uma remoção dos seus nós, pode ser que os pais desse nó (pivot) se tornem desbalanceados.
- Esse efeito colateral pode ir subindo na árvore.
- Se isso ocorrer, esses nós deverão ser avaliados e rebalanceados, se necessário.

Exercícios

- Considere o conjunto de valores abaixo e mostre a construção, de cada elemento, em uma árvore AVL.
  - Valores: [30, 63, 2, 89, 16, 24, 19, 52, 27, 9, 4, 45]
- A ordem em que os elementos são inseridos numa árvore AVL não importa, pois independente da ordem, sempre resulta na mesma árvore. O que você acha?

Exercícios

- Considere a árvore AVL abaixo e mostre o resultado dessa árvore após remover as chaves 1, 78 e 41.



Exercícios

- Qual é a diferença entre ABB e AVL?
- Dada uma árvore binária T, proponha um programa em Python que determine se T é uma árvore AVL (lembrando-se das restrições de altura e de árvore de busca).
- Quantas árvores AVL existem com altura 1, 2, 3 e 4? Faça o mesmo considerando um número mínimo de nós (dica: construa todas as possíveis árvores binárias que também sejam AVL).