ARVORE BINÁRIA

RUBRO - NEGRO

Alunos: Gustavo, Roniery, José

FUNDAMENTOS DA ÁRVORE

O que é?

- É uma árvore binária de busca balanceada (Binary Search Tree – BST) que mantém o equilíbrio usando cores (vermelho e preto) nos nós.

- O balanceamento garante que a altura da árvore seja proporcional a $\log_2(n)$, permitindo buscas, inserções e remoções rápidas (O(log n)).

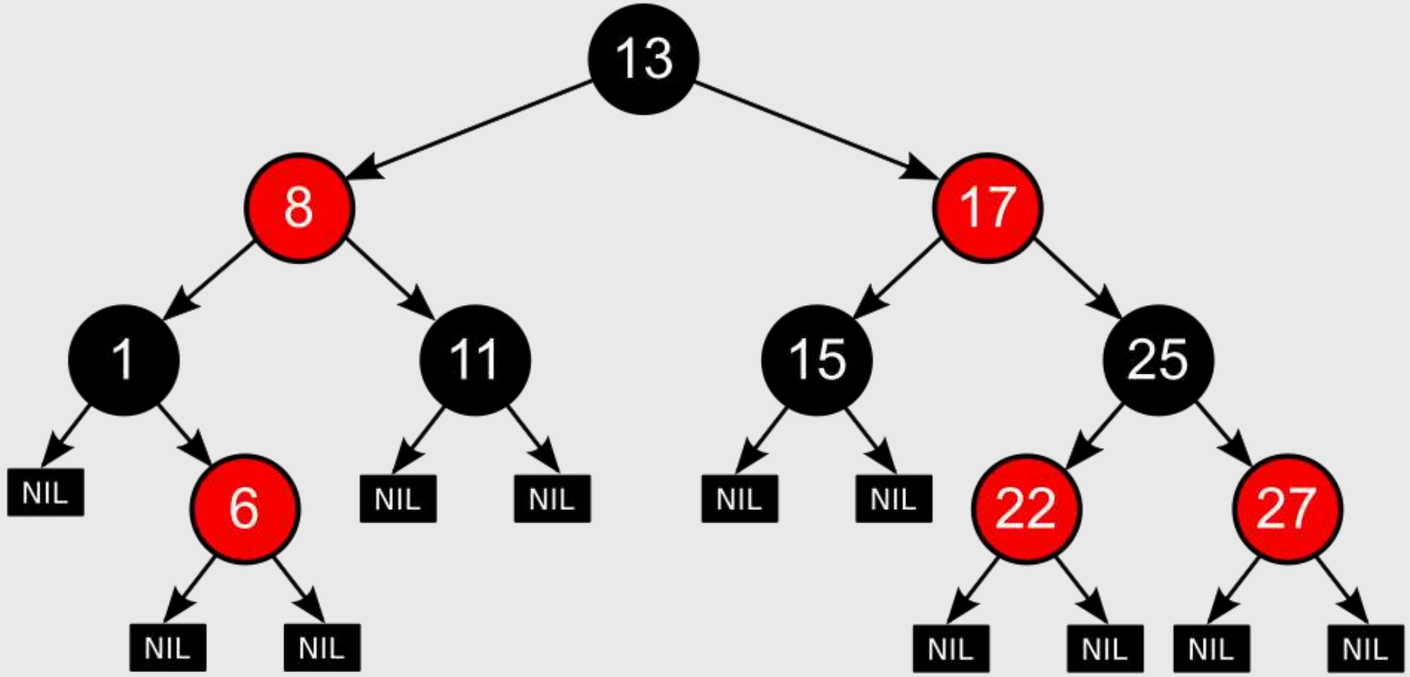
OBJETIVO

- Evitar que a árvore binária de busca fique desbalanceada (muito inclinada para um lado), o que deixaria operações lentas.

- Usar cores e regras específicas para corrigir o formato automaticamente após inserções ou remoções.

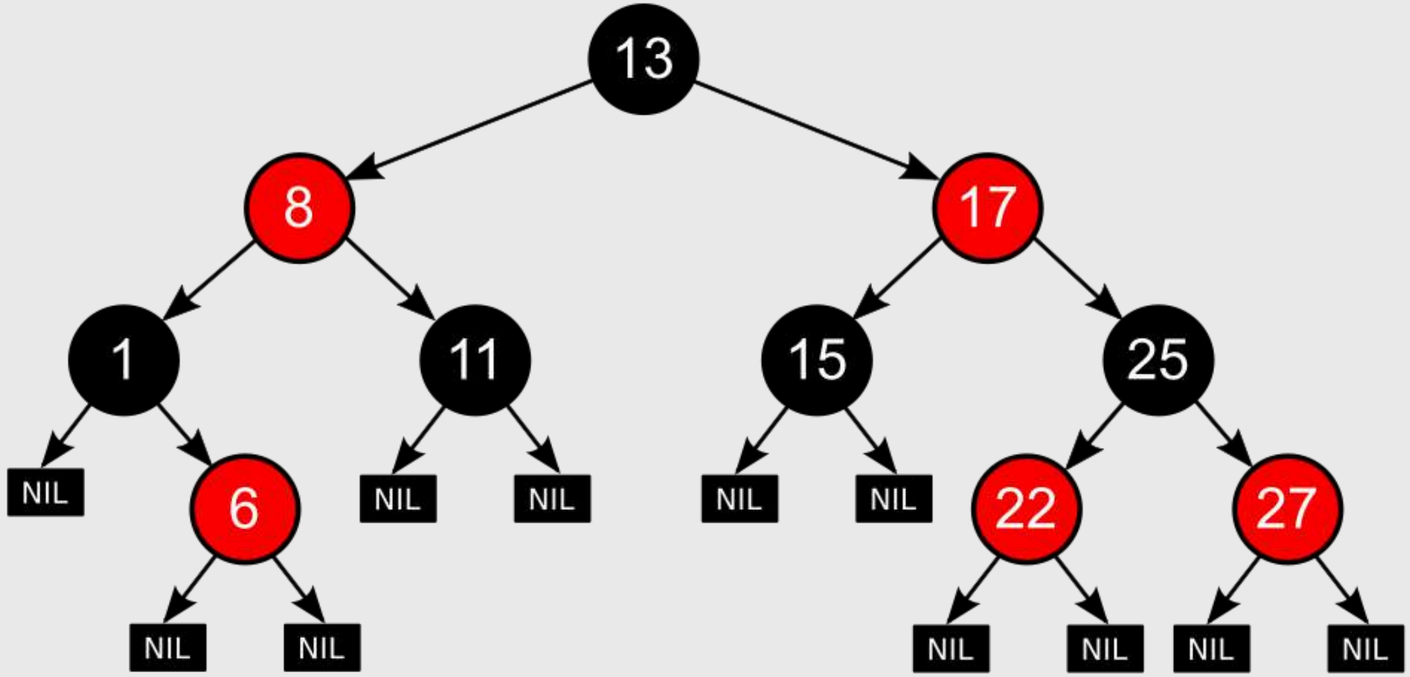
REGRAS DA ÁRVORE

- Todo nó é vermelho ou preto.



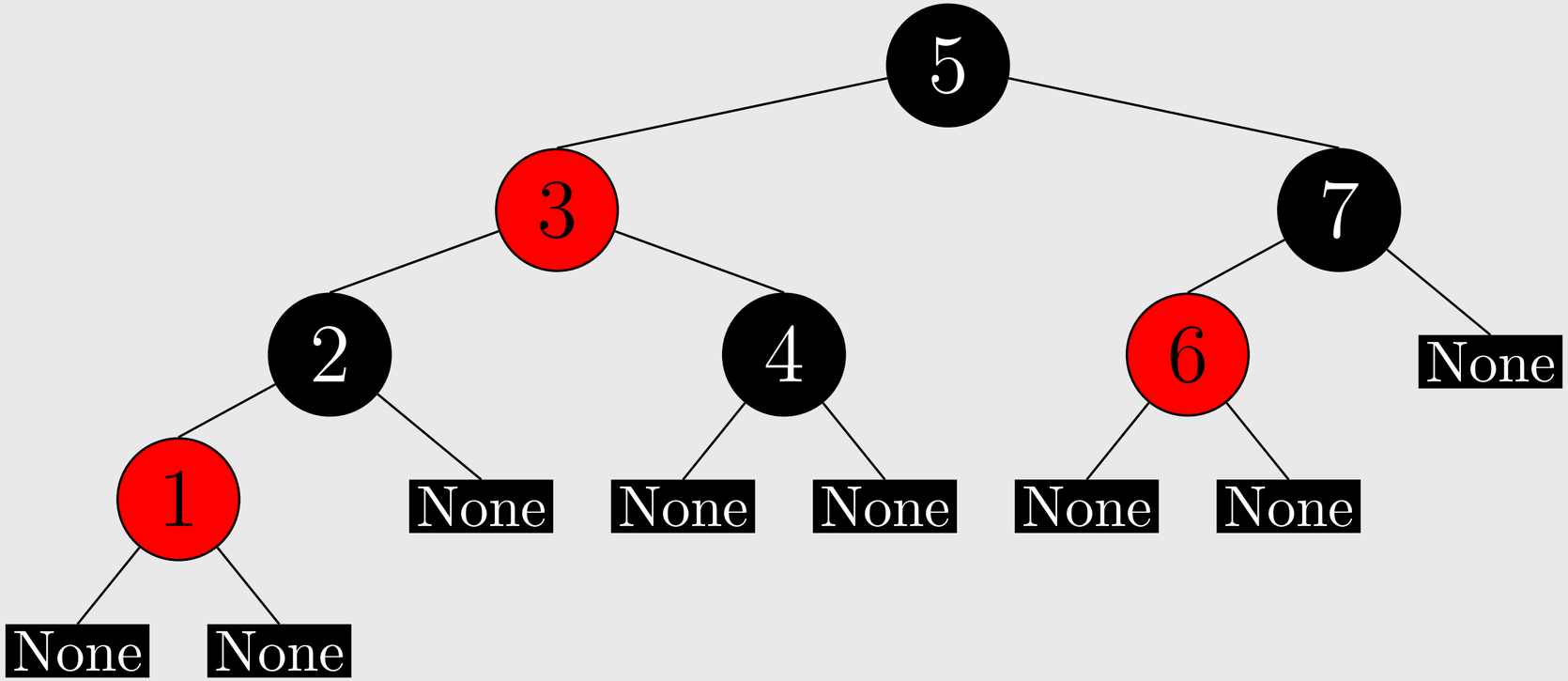
REGRAS DA ÁRVORE

- A raiz é sempre preta.



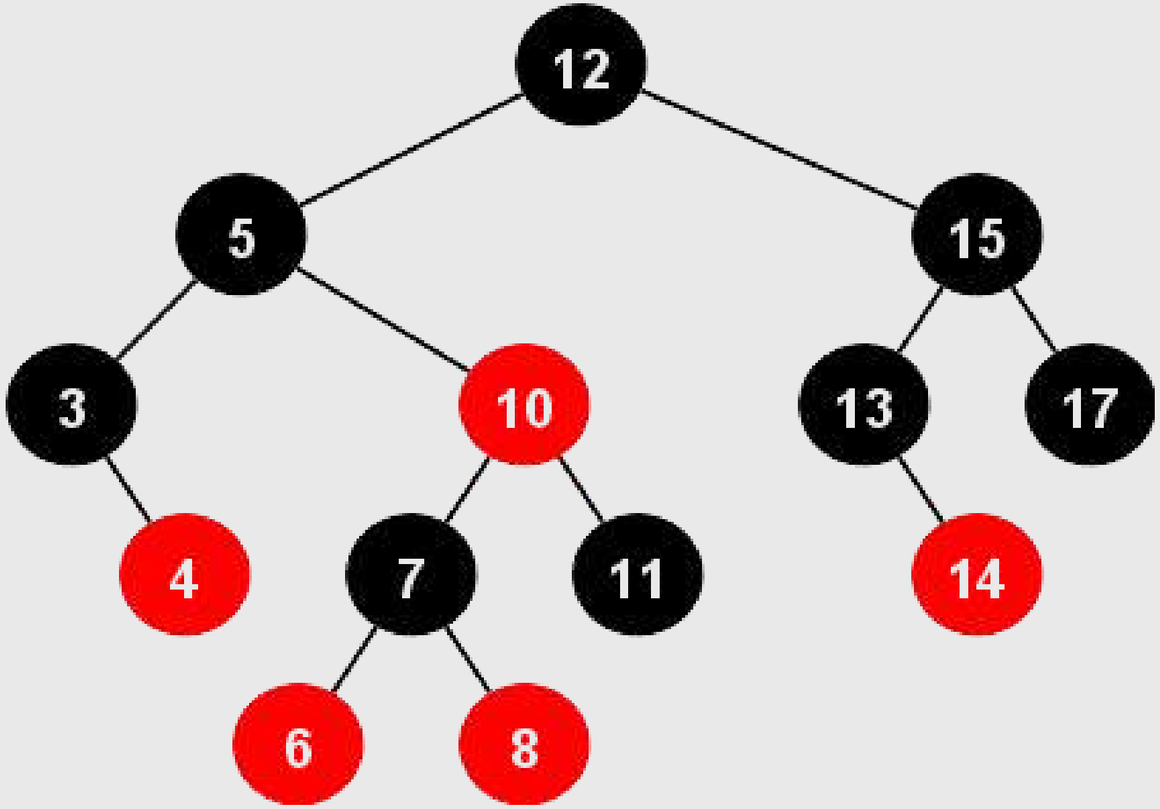
REGRAS DA ÁRVORE

- Todas as folhas nulas (NIL) são pretas.



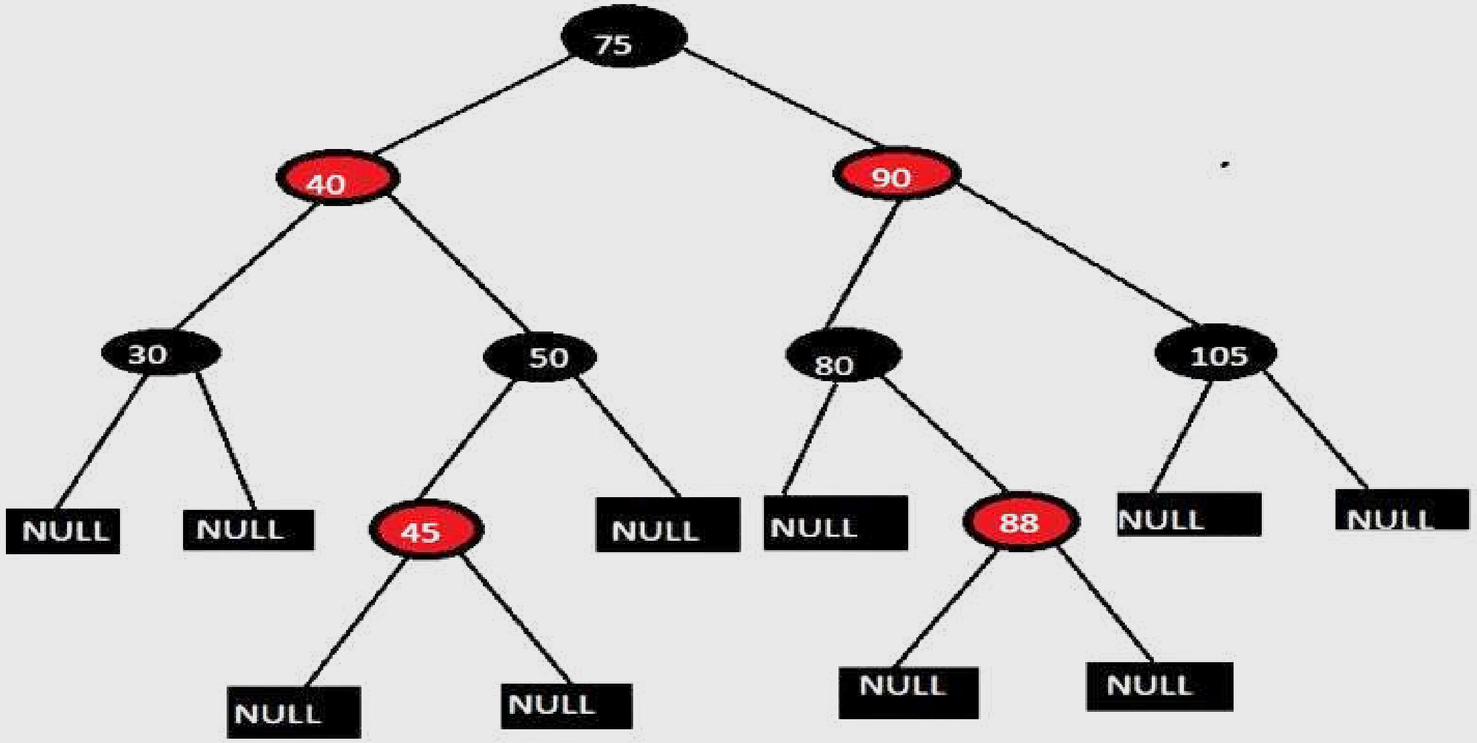
REGRAS DA ÁRVORE

- Se um nó é vermelho, seus dois filhos são pretos.



REGRAS DA ÁRVORE

- Todo caminho da raiz até uma folha nula tem o mesmo número de nós pretos.

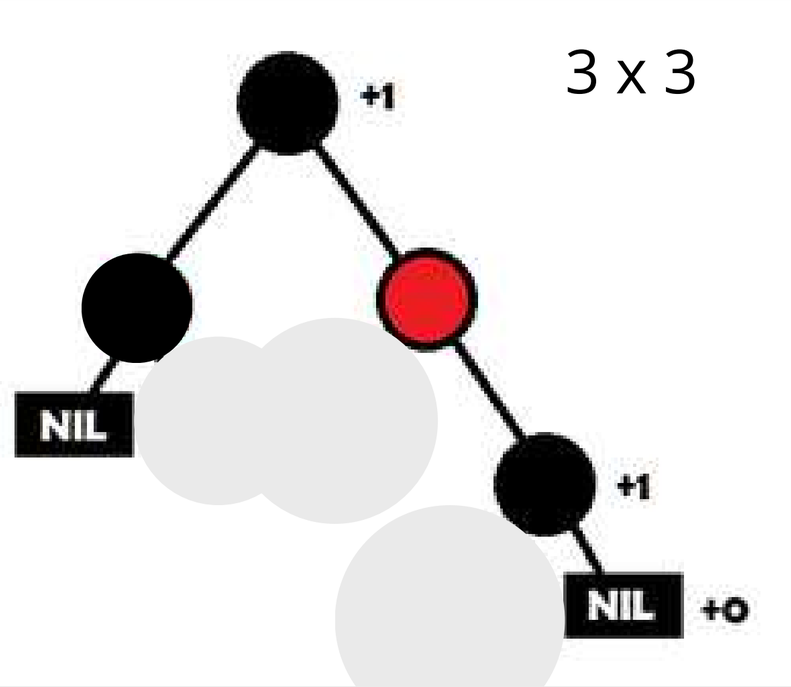


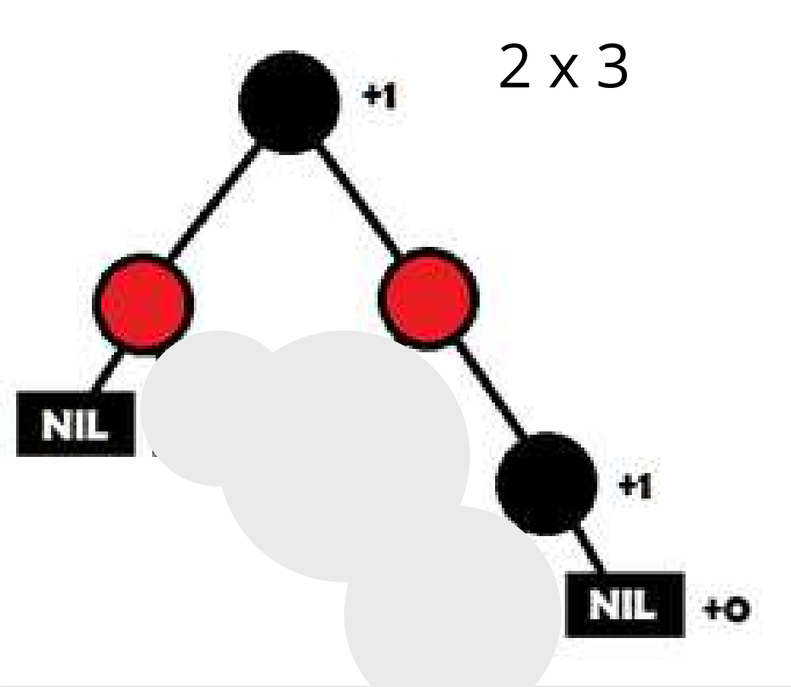
VANTAGENS

- Mantém altura próxima de $\log_2(n)$.
- Operações de busca, inserção e remoção sempre em O(log n).
- Muito usada em sistemas como:
  - Compiladores (mapas e conjuntos em C++).
  - Bancos de dados.
  - Implementação de TreeMap e TreeSet no Java.

Operações e rotação

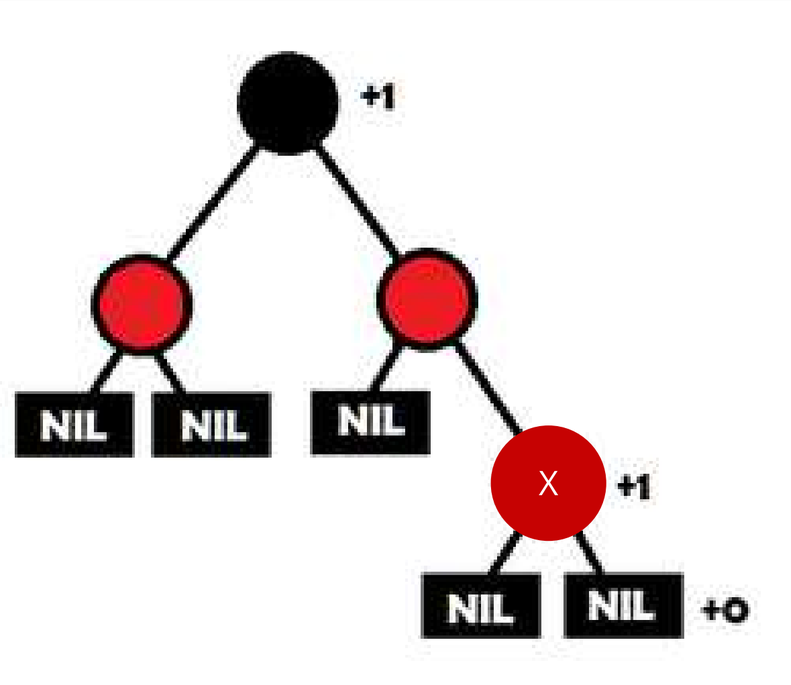
- Corrigir violações imediatamente
- Mesmo numero de nós negros



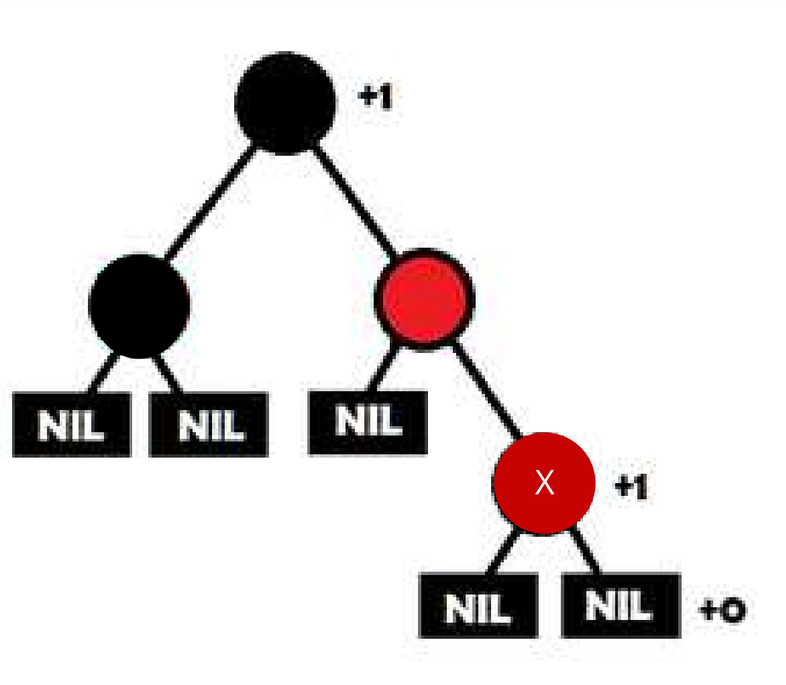


Operação de inserção

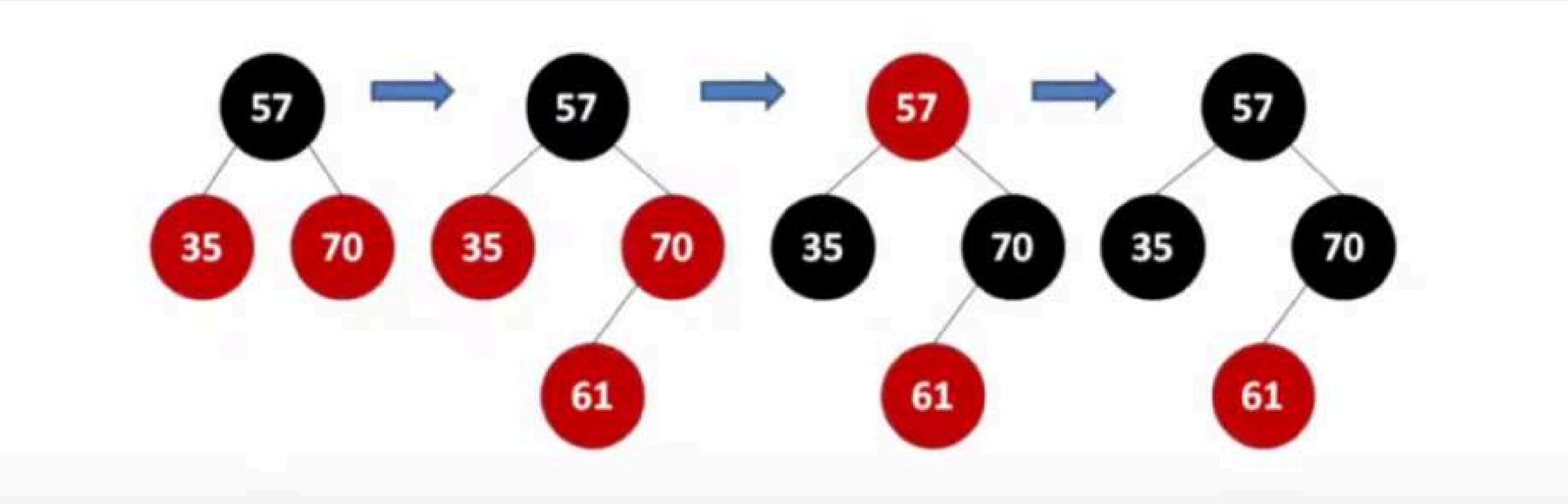
Problema de cor



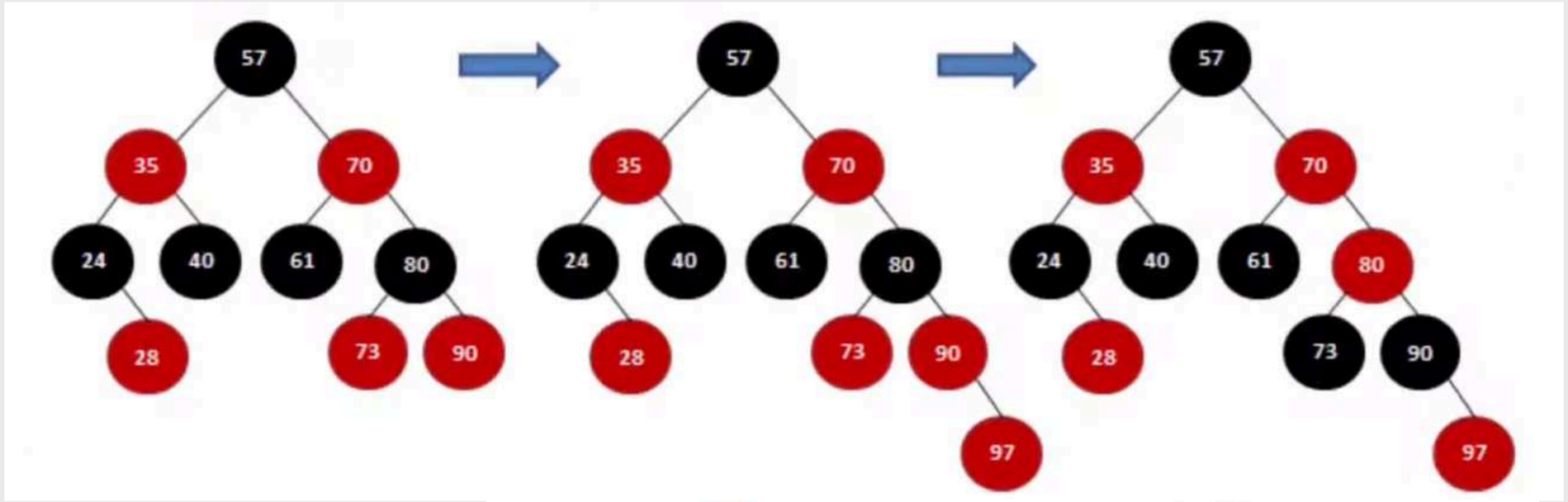
Problema de estrutura

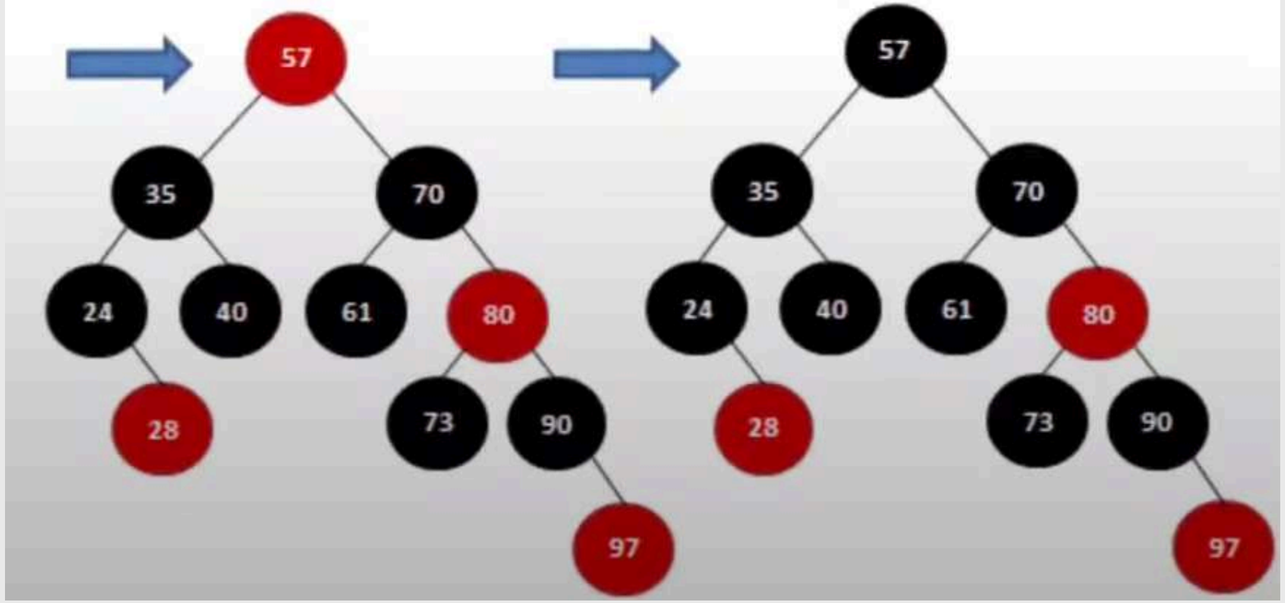


Caso 1 : Inserção

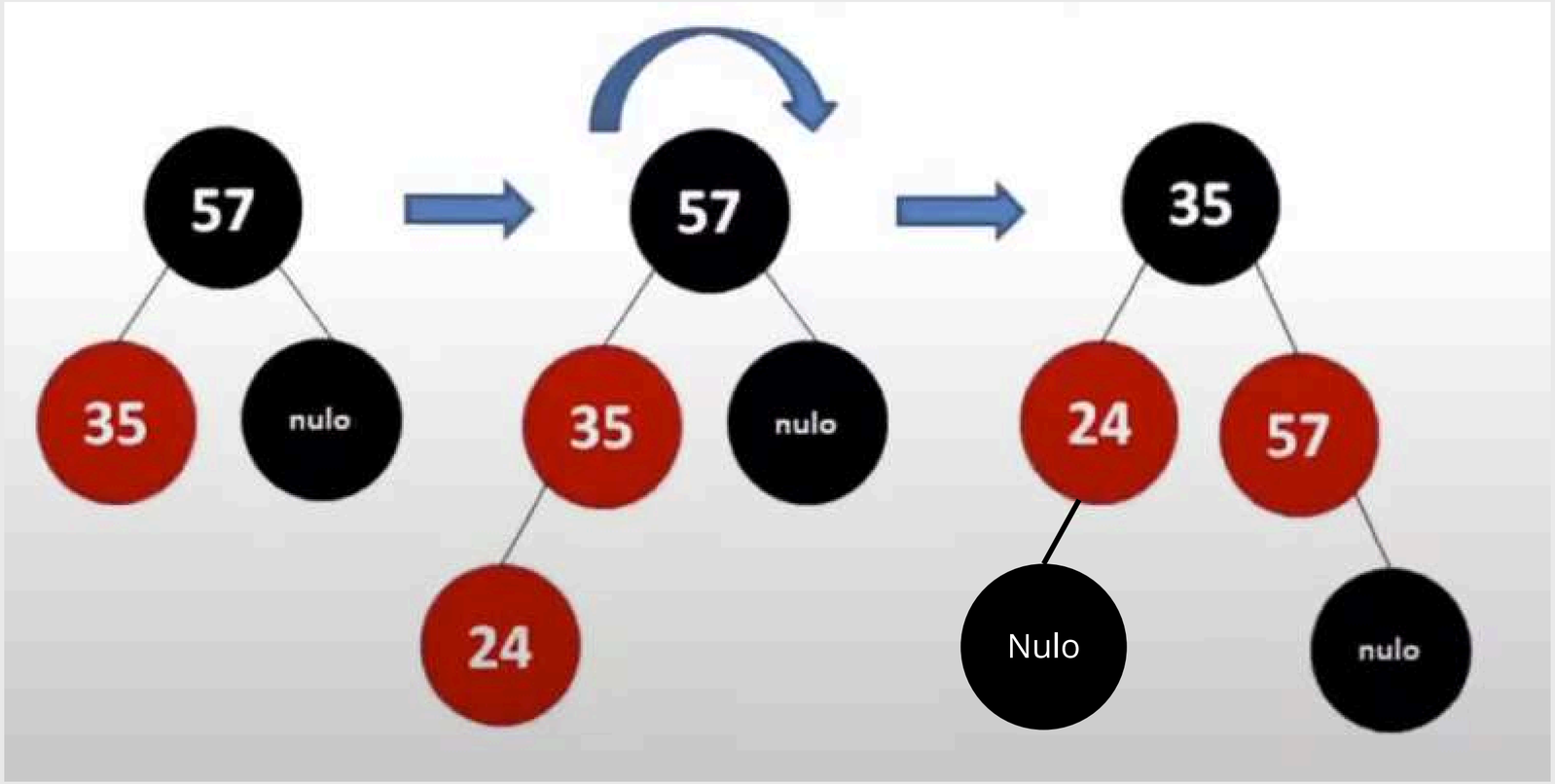


Se a raiz não começar ali?

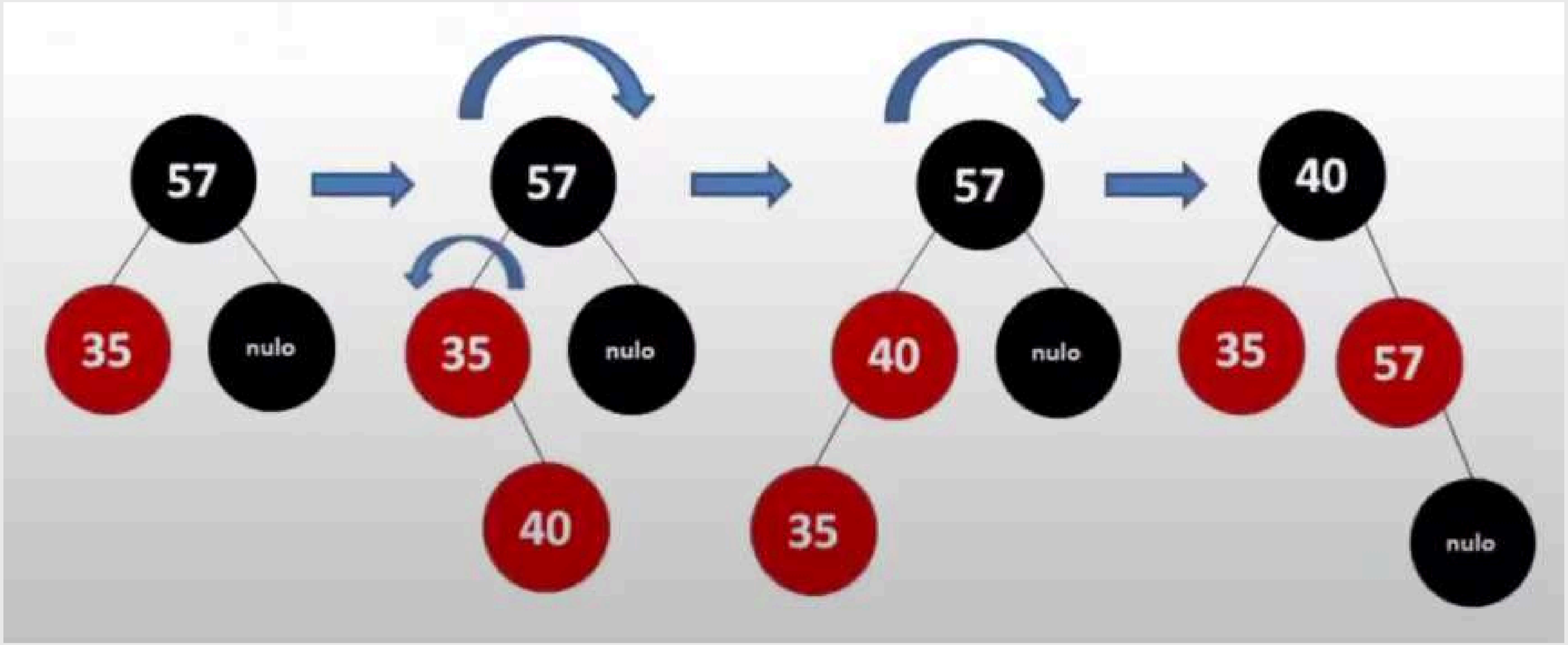




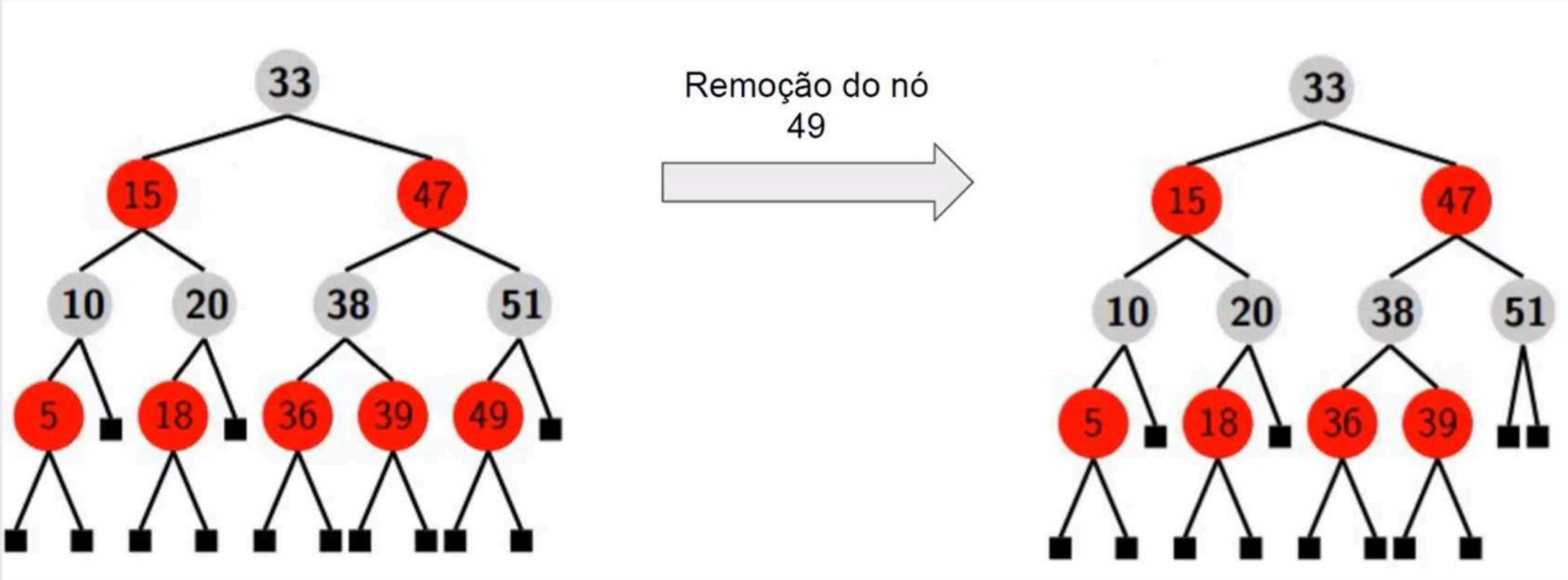
Caso 2: Rotação simples



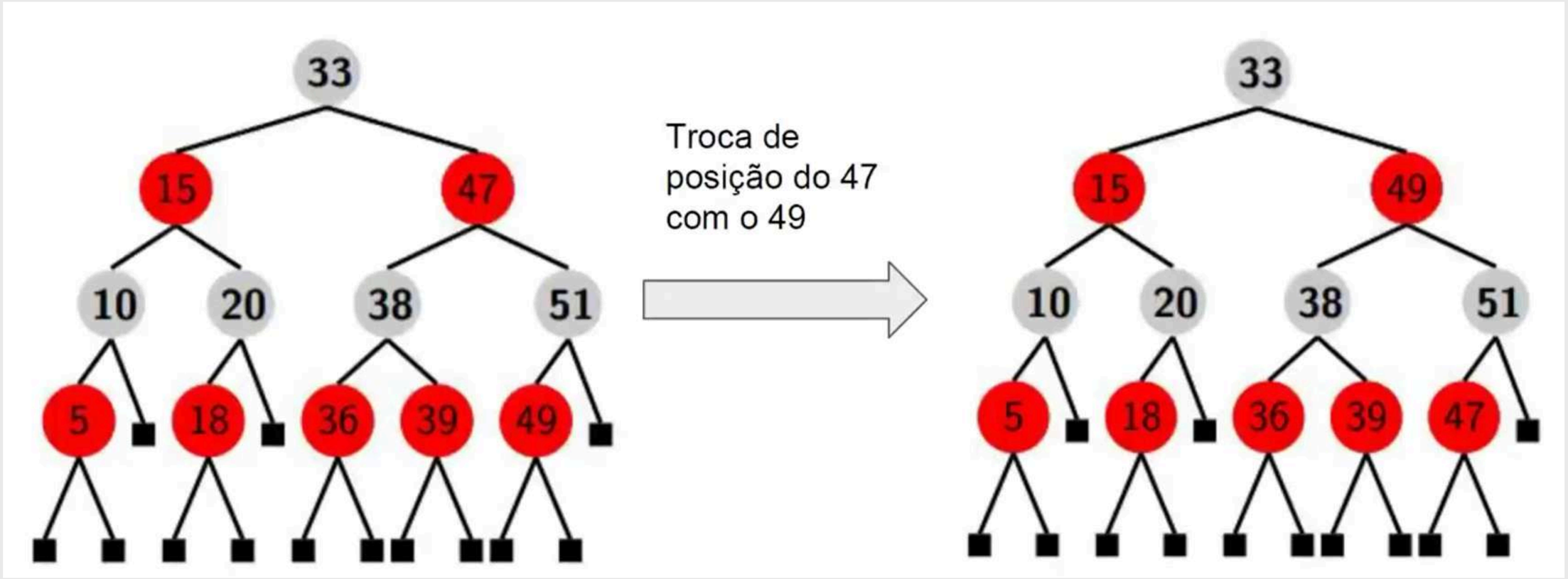
Caso 2: Rotação dupla



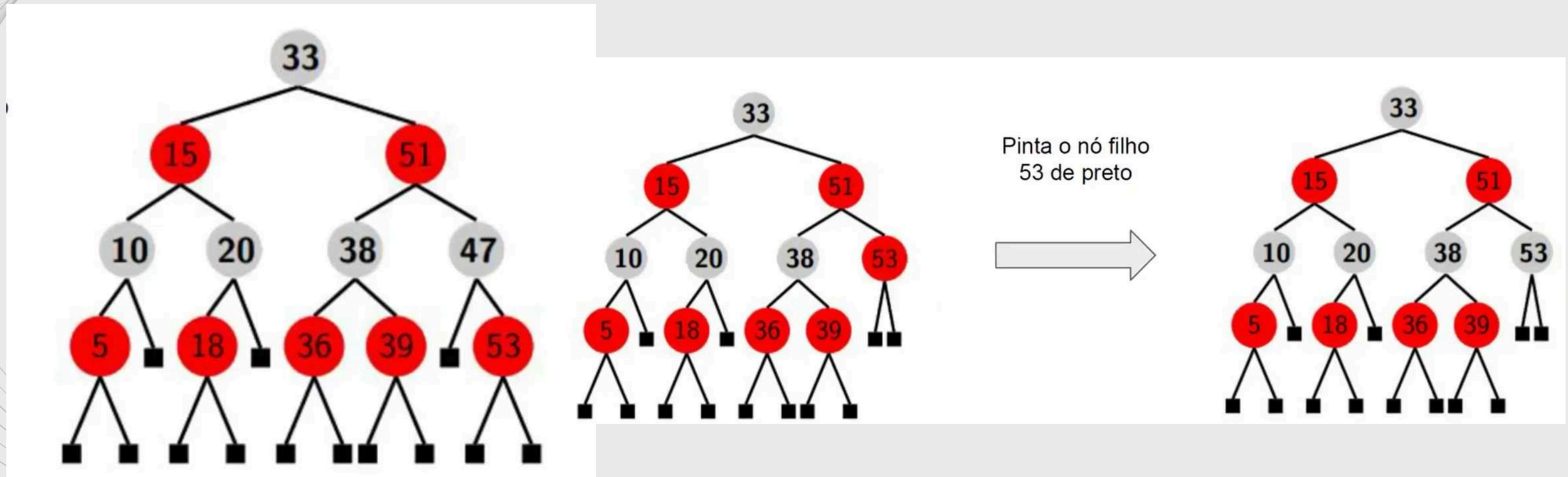
Caso 1: Remoção fácil



Caso 2: Remoção nó rubro



Caso 3: Remoção nó negro



Obrigado!