Estrutura Arbórea Autoajustável

Rubro Negra ou Vermelho e Preto

Um novo tipo de árvore?

Deixou de ser binária?

Não! Continuamos trabalhando com uma árvore binária

Árvore Vermelho e Preto

Os conceitos iniciais foram desenvolvidos por Rudolf Bayer baseada na árvore B e Leo Guibas e Robert Sedgewick refinaram o conceito e introduziram a convenção de cor

Ano de divulgação: (meados de) 1972

Mas e AVL?

Sabe-se que as **AVLs** são mais balanceadas, contudo..

A Vermelho-Preto, possui menor quantidade de **rotações** ao **inserir/apagar** nó.

Dessa forma, ela é melhor indicada para situações de muitas operações de inserção/remoção, mas lenta na busca

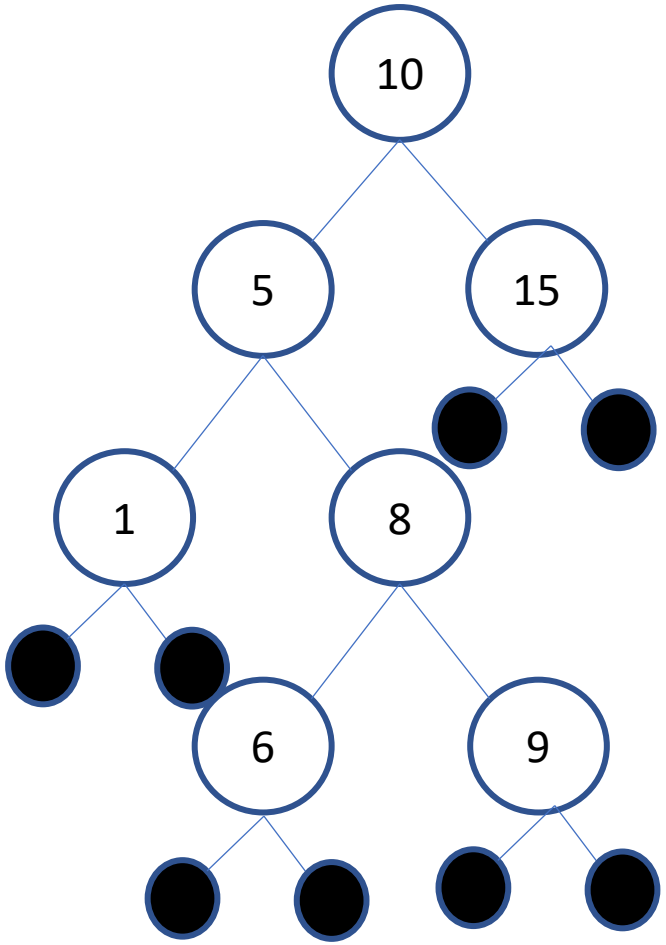
MAS ambas tem O(log n)

Quais as propriedades de uma Vermelho-Preto?

I. Ser uma árvore binária [para essa disciplina]
II. Cada nó deve ter uma cor: **Preta** ou **Vermelha**
III. O nó **raíz** é sempre terá **cor preta**
IV. **Não há** dois nós **vermelhos adjacentes**, ou seja se um nó é vermelho, então ambos os filhos são pretos
V. O número de nós pretos em qualquer caminho da raiz (de toda a árvore ou subárvores) a uma folha é o mesmo.
VI. Todas as **folhas** são **pretas**

Atenção, Atenção! Folhas são os ponteiros dos últimos elementos da árvore

Exemplo, nó folha



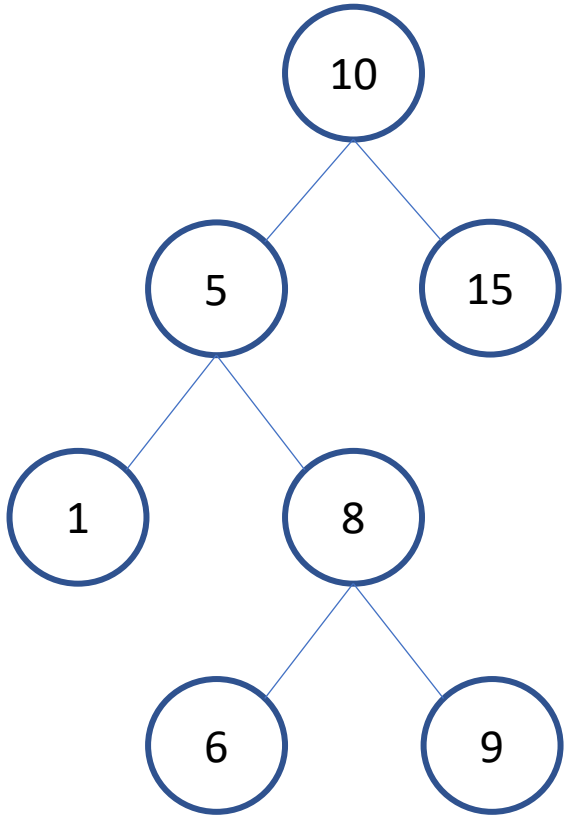
Mas o que essa regra V tem de tão importante?

- Bom, em uma AVL vimos que o balanceamento era definido pelo FB, que considerava a altura da sub-árvores esquerda e direita, não é?
- Show! Na árvore vermelho e preto, temos ainda a análise das alturas, mas consideramos a ‘altura-negra’ da árvore, ou seja, a regra V:

“O número de nós pretos em qualquer caminho da raiz (de toda a árvore ou subárvores) a uma folha é o mesmo”

Com esta regra, podemos definir que a árvore estará balanceada porque as subárvores esquerda e direita do nó raiz devem ser da mesma altura-negra

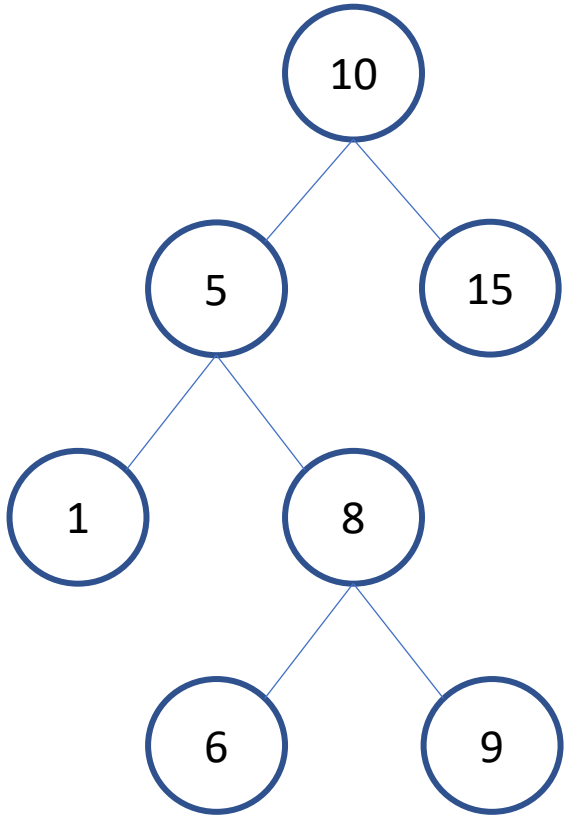
Por vezes, vamos observar uma árvore e pensar “essa tá desbalanceada!”. Olha só:



Aprendemos lá em AVL que ela está desbalanceada, pois:
$FB_{10} = alt_{esq} – alt_{dir} = 3 – 1 = 2$

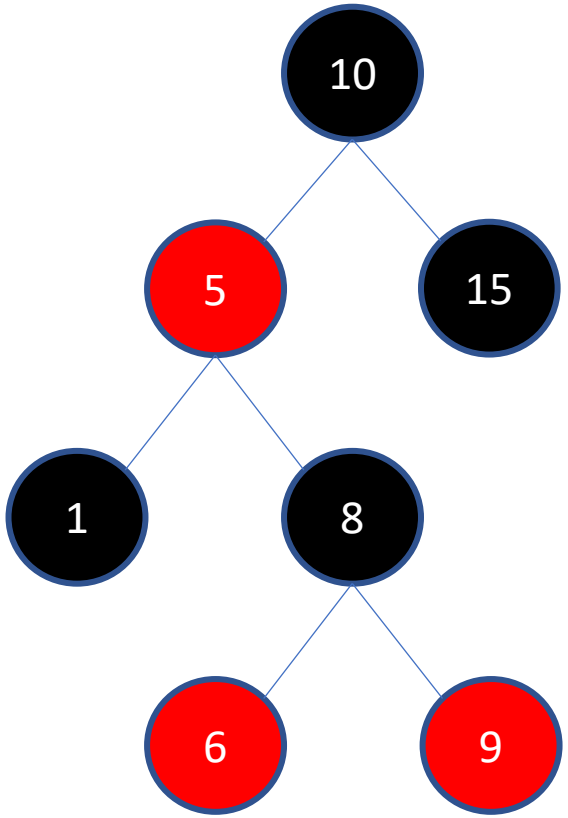
Mas, nos padrões de uma vermelho-preto, não

Olha só:



A altura-negra a esquerda e a direita de qualquer nó é a mesma.

Olha só:



A altura-negra a esquerda e a direita de qualquer nó é a mesma.

Ou seja, a árvore está balanceada

Legal, né?

Mas não pense que a vermelho e preto é só colorir

Por vezes faremos algumas rotações

Em comparação a AVL o número de rotações será menor, mas não anulamos a possibilidade da necessidade de reorganizar os elementos de lugar, ok?

Então bora vê como podemos aplicar as propriedades dessa árvore?

Vamos lá!

Inserção

Primeiro Passo: Agir naturalmente e inserir/remover um elemento da árvore

Após a inclusão do elemento em sua posição natural pelas regras de uma árvore binária, ele será colorido vermelho.

Mas porque vermelho? Vamos voltar as regras?

I. Ser uma árvore binária
II. Cada nó deve ter uma cor: **Preta** ou **Vermelha**
III. O nó **raíz** é sempre terá **cor preta**
IV. **Não há** dois nós **vermelhos adjacentes**, ou seja se um nó é vermelho, então ambos os filhos são pretos
V. O número de nós pretos em qualquer caminho da raiz (de toda a árvore ou subárvores) a uma folha é o mesmo.
VI. Todas as **folhas** são **pretas**

Logo, a **regra V** não será “quebrada”

E se o pai do novo nó estiver colorido de **preto**, tudo estará ok

E nenhuma operação adicional precisará ser feita

Mas, nem tudo tá ok. E se o pai estiver colorido com vermelho?

Ixi. Vamos voltar as regras?

I. Ser uma árvore binária
II. Cada nó deve ter uma cor: **Preta** ou **Vermelha**
III. O nó **raíz** é sempre terá **cor preta**
IV. **Não há** dois nós **vermelhos adjacentes**, ou seja se um nó é vermelho, então ambos os filhos são pretos
V. O número de nós pretos em qualquer caminho da raiz (de toda a árvore ou subárvores) a uma folha é o mesmo.
VI. Todas as **folhas** são **pretas**

E agora?

Bom, agora é observar as cores do "tio" do novo nó

Se ele também for vermelho, podemos alterar a cor do avô para vermelho e a cor do "tio" e "pai" do novo nó para preto

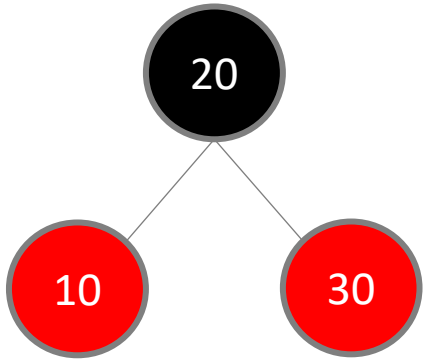
Mas para quê tantas cores?

Para não quebrar a regra V

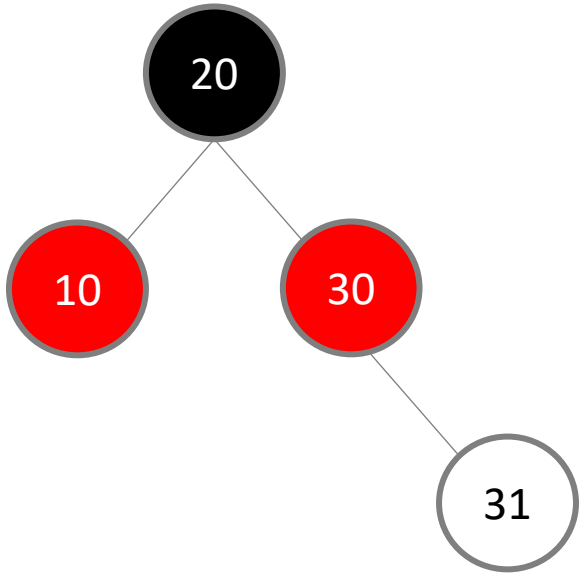
E tem mais! Se nessa mudança a raiz de toda a árvore ficar vermelha, ao final podemos ajusta-la para a cor preta e tudo ficará ok. Nenhuma regra violada

Precisamos de um exemplo, não é?

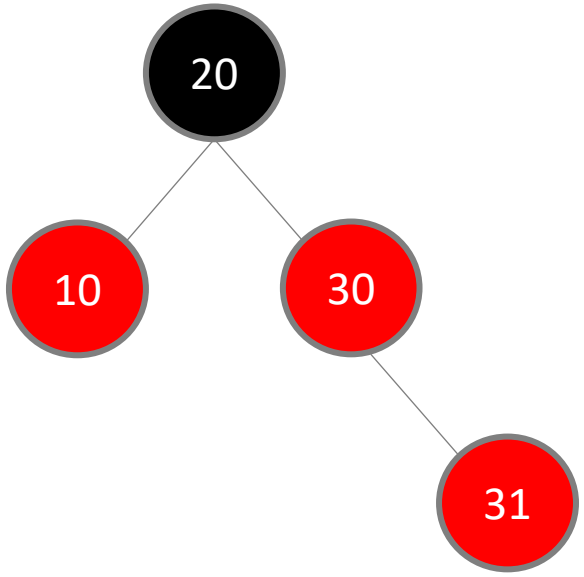
Olha só: Vamos adicionar o elemento: 31



Olha só: Insere o elemento naturalmente na posição correspondente



Olha só: E inicialmente, sua cor é **vermelha**, para não termos a quebra da regra V



Regras (propriedades)

I. Ser uma árvore binária

II. Cada nó deve ter uma cor: **Preta** ou **Vermelha**

III. O nó **raíz** é sempre terá **cor preta**

IV. **Não há** dois nós **vermelhos adjacentes**, ou seja se um nó é vermelho, então ambos os filhos são pretos

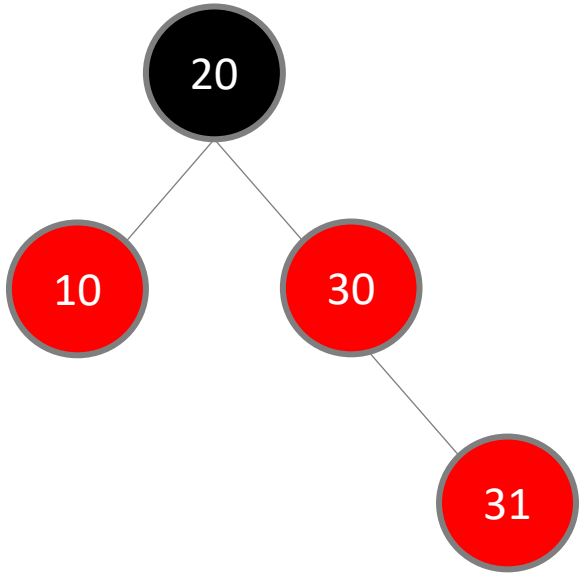
V. O número de nós pretos em qualquer caminho da raiz (de toda a árvore ou subárvores) a uma folha é o mesmo.

VI. Todas as **folhas** são **pretas**

Olha só:

Contudo, a cor do "pai" de 31 é vermelha. E quebramos a regra IV

Vamos ter que ajustar! Uma estratégia é colorir o "avô" de vermelho e, "pai" e o "tio" de 31 com a cor preta, para manter as regras IV e V



Regras (propriedades)

I. Ser uma árvore binária

II. Cada nó deve ter uma cor: **Preta** ou **Vermelha**

III. O nó **raíz** é sempre terá **cor preta**

IV. **Não há** dois nós **vermelhos adjacentes**, ou seja se um nó é vermelho, então ambos os filhos são pretos

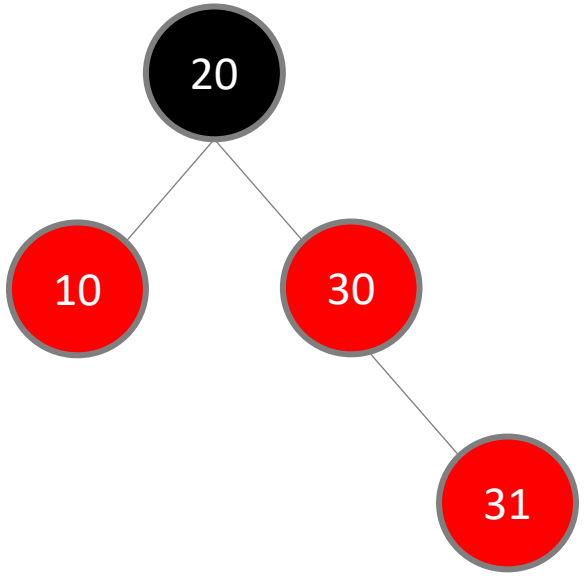
V. O número de nós pretos em qualquer caminho da raiz (de toda a árvore ou subárvores) a uma folha é o mesmo.

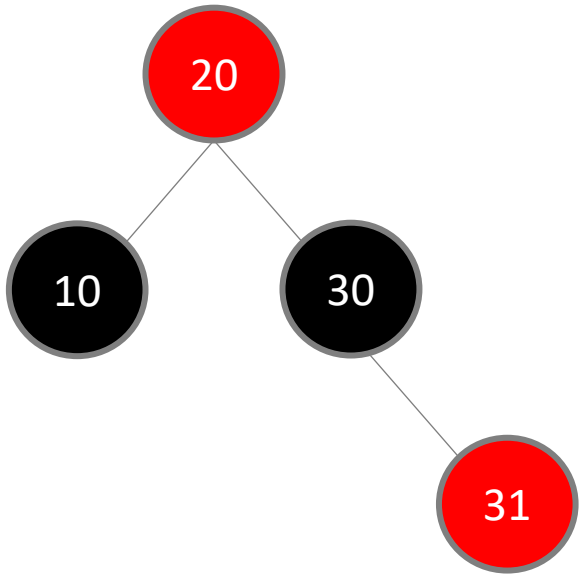
VI. Todas as **folhas** são **pretas**

Olha só:

Contudo, a cor do "pai" de 31 é vermelha. E quebramos a regra IV

Vamos ter que ajustar! Uma estratégia é colorir o "avô" de vermelho e, "pai" e o "tio" de 31 com a cor preta, para manter as regras IV e V





Regras (propriedades)

I. Ser uma árvore binária

II. Cada nó deve ter uma cor: **Preta** ou **Vermelha**

III. O nó **raíz** é sempre terá **cor preta**

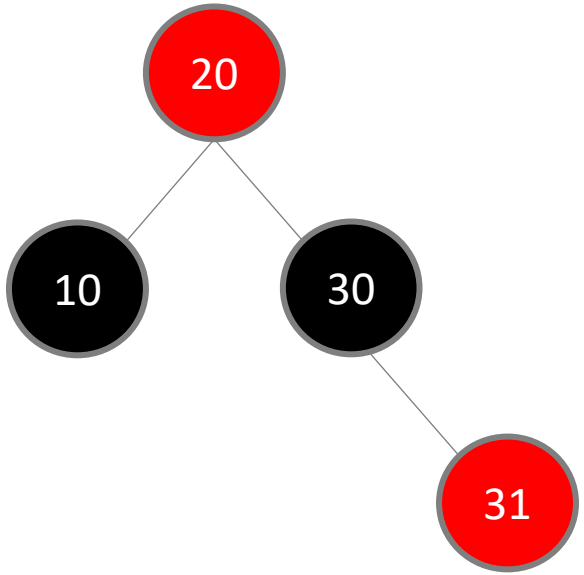
IV. **Não há** dois nós **vermelhos adjacentes**, ou seja se um nó é vermelho, então ambos os filhos são pretos

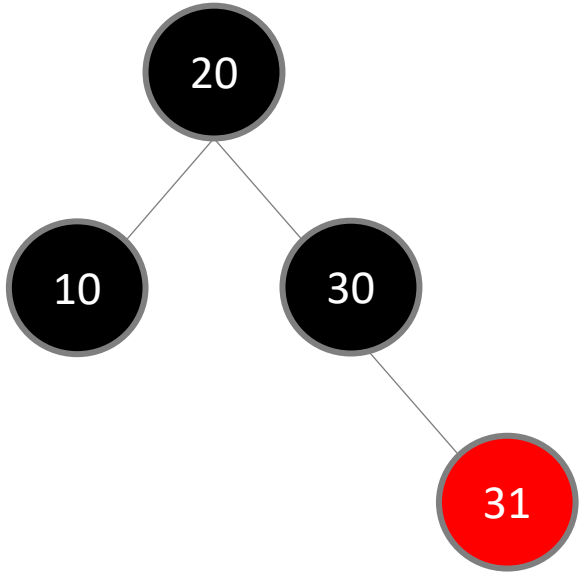
V. O número de nós pretos em qualquer caminho da raiz (de toda a árvore ou subárvores) a uma folha é o mesmo.

VI. Todas as **folhas** são **pretas**

Olha só:

Se nessa mudança a raiz de toda a árvore ficar vermelha, ao final podemos ajusta-la para a cor preta para nenhuma regra ser violada





Regras (propriedades)

I. Ser uma árvore binária

II. Cada nó deve ter uma cor: **Preta** ou **Vermelha**

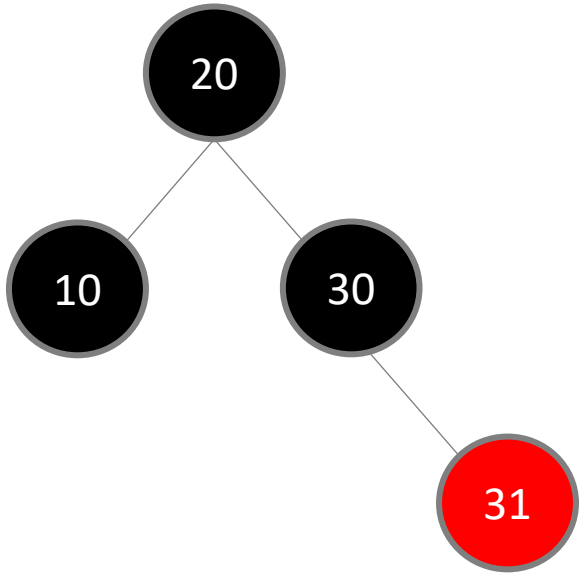
III. O nó **raíz** é sempre terá **cor preta**

IV. **Não há** dois nós **vermelhos adjacentes**, ou seja se um nó é vermelho, então ambos os filhos são pretos

V. O número de nós pretos em qualquer caminho da raiz (de toda a árvore ou subárvores) a uma folha é o mesmo.

VI. Todas as **folhas** são **pretas**

Olha só: Resultado Final



Regras (propriedades)

I. Ser uma árvore binária

II. Cada nó deve ter uma cor: **Preta** ou **Vermelha**

III. O nó **raíz** é sempre terá **cor preta**

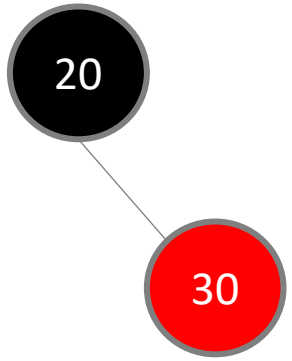
IV. **Não há** dois nós **vermelhos adjacentes**, ou seja se um nó é vermelho, então ambos os filhos são pretos

V. O número de nós pretos em qualquer caminho da raiz (de toda a árvore ou subárvores) a uma folha é o mesmo.

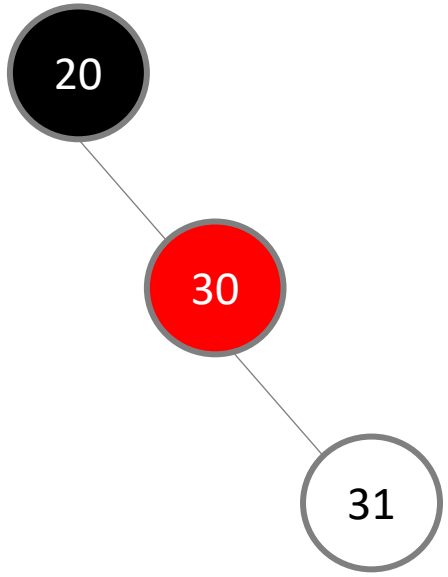
VI. Todas as **folhas** são **pretas**

Vamos pensar em outra situação?

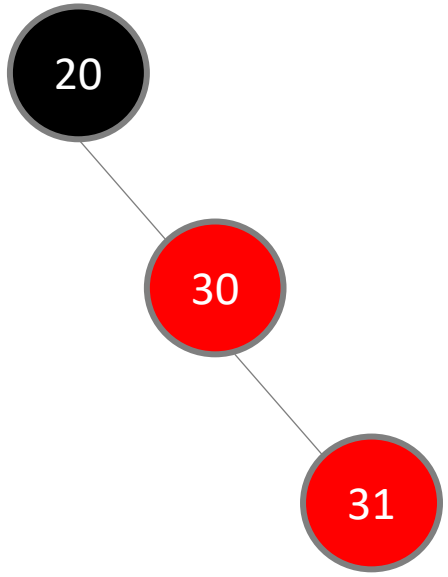
Olha só: Vamos adicionar o elemento: 31



Olha só: Insere o elemento naturalmente na posição correspondente



Olha só: E inicialmente, sua cor é **vermelha**, para não termos a quebra da regra V



Regras (propriedades)

I. Ser uma árvore binária

II. Cada nó deve ter uma cor: **Preta** ou **Vermelha**

III. O nó **raíz** é sempre terá **cor preta**

IV. **Não há** dois nós **vermelhos adjacentes**, ou seja se um nó é vermelho, então ambos os filhos são pretos

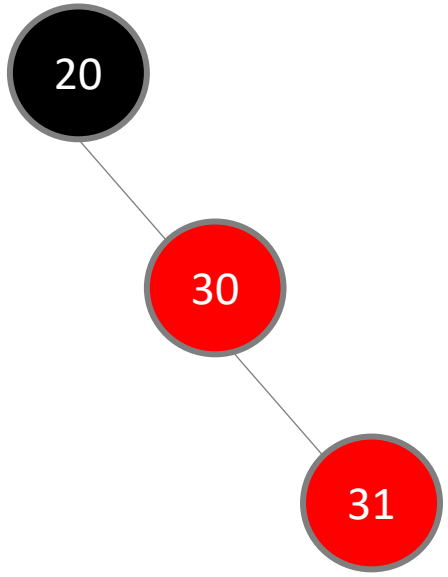
V. O número de nós pretos em qualquer caminho da raiz (de toda a árvore ou subárvores) a uma folha é o mesmo.

VI. Todas as **folhas** são **pretas**

Olha só:

Contudo, a cor do "pai" de 31 é vermelha. E quebramos a regra IV

E esse "pai" não tem um irmão vermelho, dessa forma estratégia é diferente: Vamos trocar a cor do "avô" para vermelho e do "pai" para preto



Regras (propriedades)

I. Ser uma árvore binária

II. Cada nó deve ter uma cor: **Preta** ou **Vermelha**

III. O nó **raíz** é sempre terá **cor preta**

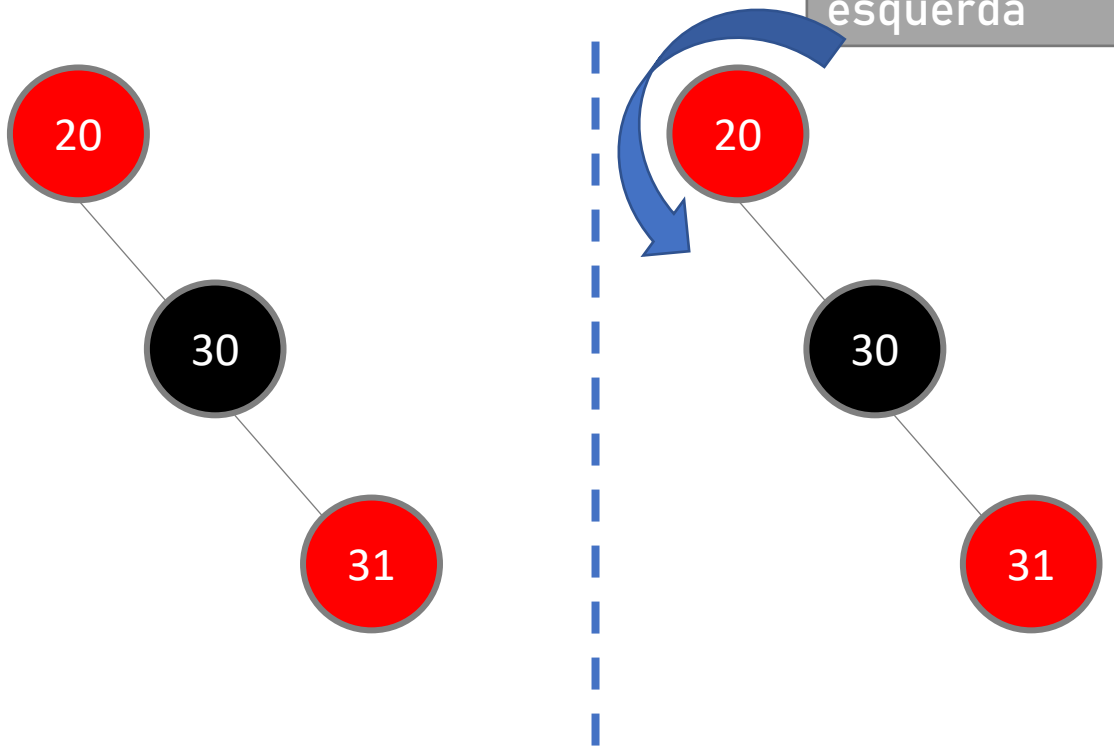
IV. **Não há** dois nós **vermelhos adjacentes**, ou seja se um nó é vermelho, então ambos os filhos são pretos

V. O número de nós pretos em qualquer caminho da raiz (de toda a árvore ou subárvores) a uma folha é o mesmo.

VI. Todas as **folhas** são **pretas**

Olha só: Trocamos a cor do "avô" e do "pai", porém violamos a regra III e V

A alternativa é promovermos uma rotação simples do avô a esquerda



Regras (propriedades)

I. Ser uma árvore binária

II. Cada nó deve ter uma cor: **Preta** ou **Vermelha**

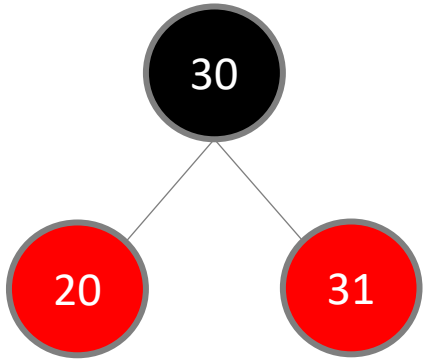
III. O nó **raíz** é sempre terá **cor preta**

IV. **Não há** dois nós **vermelhos adjacentes**, ou seja se um nó é vermelho, então ambos os filhos são pretos

V. O número de nós pretos em qualquer caminho da raiz (de toda a árvore ou subárvores) a uma folha é o mesmo.

VI. Todas as **folhas** são **pretas**

Olha só: Com esse movimento a árvore está ok



Regras (propriedades)

I. Ser uma árvore binária

II. Cada nó deve ter uma cor: **Preta** ou **Vermelha**

III. O nó **raíz** é sempre terá **cor preta**

IV. **Não há** dois nós **vermelhos adjacentes**, ou seja se um nó é vermelho, então ambos os filhos são pretos

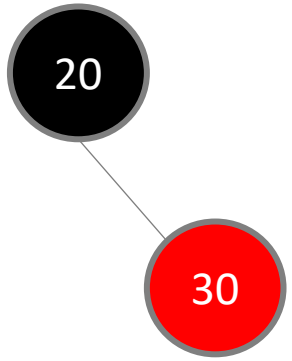
V. O número de nós pretos em qualquer caminho da raiz (de toda a árvore ou subárvores) a uma folha é o mesmo.

VI. Todas as **folhas** são **pretas**

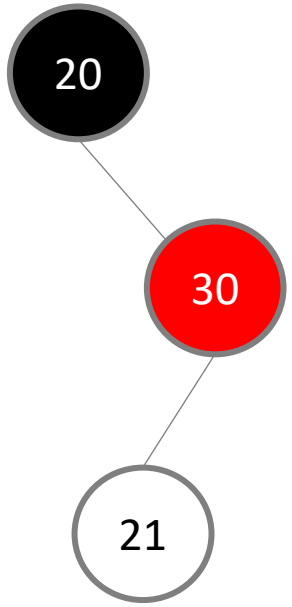
Vamos pensar em outra situação?

Até agora só inserimos o "filho" no mesmo lado que o "pai" está né? E se for no outro lado?

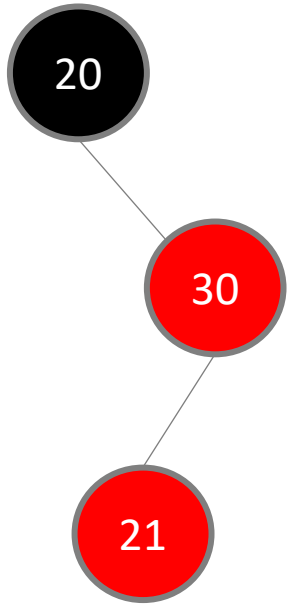
Olha só: Vamos adicionar o elemento: 21



Olha só: Insere o elemento naturalmente na posição correspondente



Olha só: E inicialmente, sua cor é **vermelha**, para não termos a quebra da regra V



Regras (propriedades)

I. Ser uma árvore binária

II. Cada nó deve ter uma cor: **Preta** ou **Vermelha**

III. O nó **raíz** é sempre terá **cor preta**

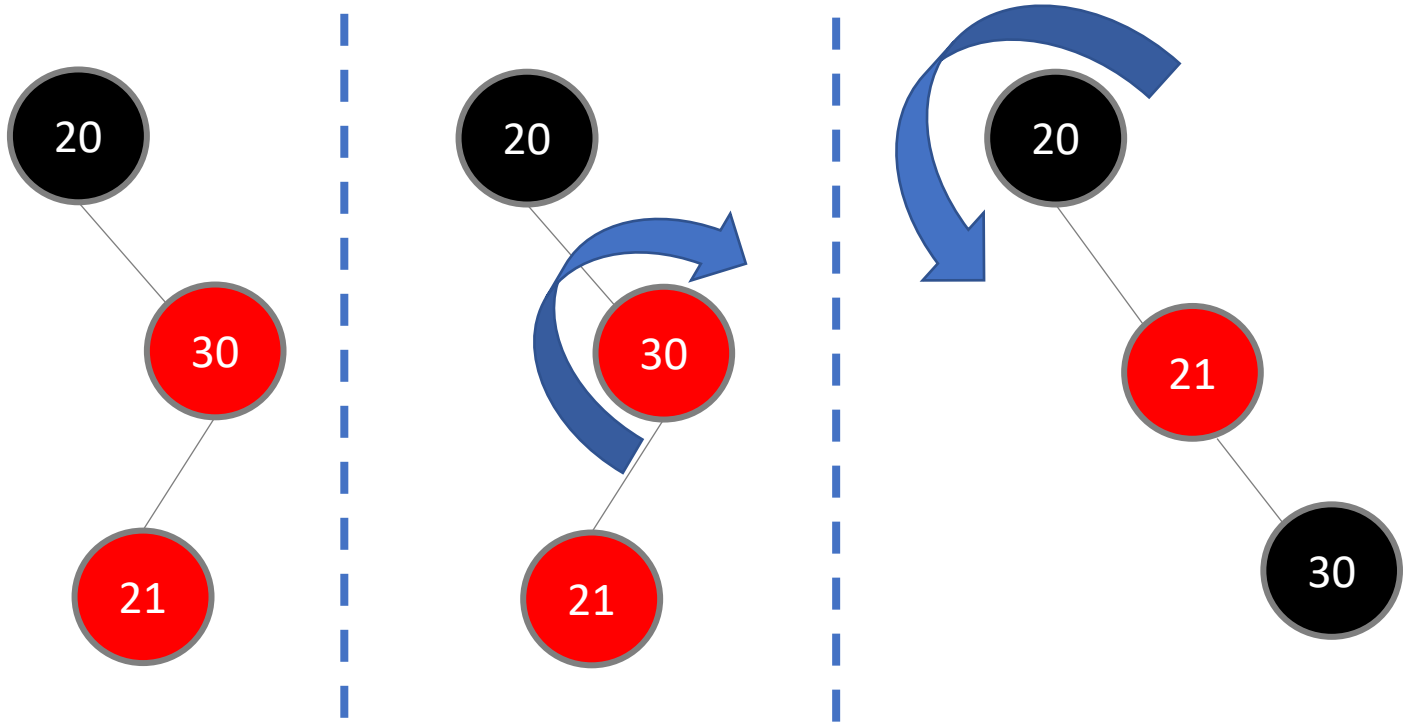
IV. **Não há** dois nós **vermelhos adjacentes**, ou seja se um nó é vermelho, então ambos os filhos são pretos

V. O número de nós pretos em qualquer caminho da raiz (de toda a árvore ou subárvores) a uma folha é o mesmo.

VI. Todas as **folhas** são **pretas**

Olha só:

Só que estamos quebrando a regra III. Contudo, não Trocamos a cor do "avô" e do "pai



A alternativa é promovermos uma rotação dupla. Primeiro no pai, depois no avô e trocamos a cor da nova raiz

Regras (propriedades)

I. Ser uma árvore binária

II. Cada nó deve ter uma cor: **Preta** ou **Vermelha**

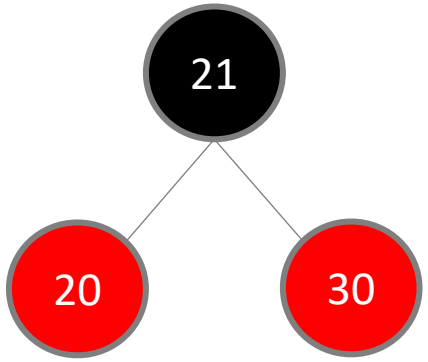
III. O nó **raíz** é sempre terá **cor preta**

IV. **Não há** dois nós **vermelhos adjacentes**, ou seja se um nó é vermelho, então ambos os filhos são pretos

V. O número de nós pretos em qualquer caminho da raiz (de toda a árvore ou subárvores) a uma folha é o mesmo.

VI. Todas as **folhas** são **pretas**

Olha só: Tudo certo. Árvore ok!



Regras (propriedades)

I. Ser uma árvore binária

II. Cada nó deve ter uma cor: **Preta** ou **Vermelha**

III. O nó **raíz** é sempre terá **cor preta**

IV. **Não há** dois nós **vermelhos adjacentes**, ou seja se um nó é vermelho, então ambos os filhos são pretos

V. O número de nós pretos em qualquer caminho da raiz (de toda a árvore ou subárvores) a uma folha é o mesmo.

VI. Todas as **folhas** são **pretas**

Vamos pensar em outra situação?

Eita, quantas situações né? Mas vamos fazer melhor, vamos pensarem quais elementos estamos sempre analisando

Quais elementos sempre consultamos?

Regras (propriedades)

I. Ser uma árvore binária

II. Cada nó deve ter uma cor: **Preta** ou **Vermelha**

III. O nó **raíz** é sempre terá **cor preta**

IV. **Não há** dois nós **vermelhos adjacentes**, ou seja se um nó é vermelho, então ambos os filhos são pretos

V. O número de nós pretos em qualquer caminho da raiz (de toda a árvore ou subárvores) a uma folha é o mesmo.

VI. Todas as **folhas** são **pretas**

Pai

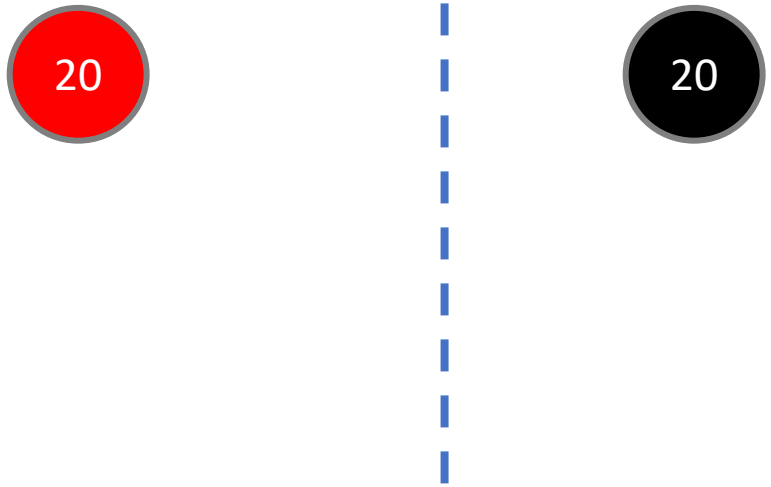
Tio

Avô

Vamos encontrar um padrão?

1 – Árvore vazia, inclusão do 1º elemento: recolorir

Ao inserir ele é vermelho, logo depois será preto, para não quebrar a regra III



Regras (propriedades)

I. Ser uma árvore binária

II. Cada nó deve ter uma cor: **Preta** ou **Vermelha**

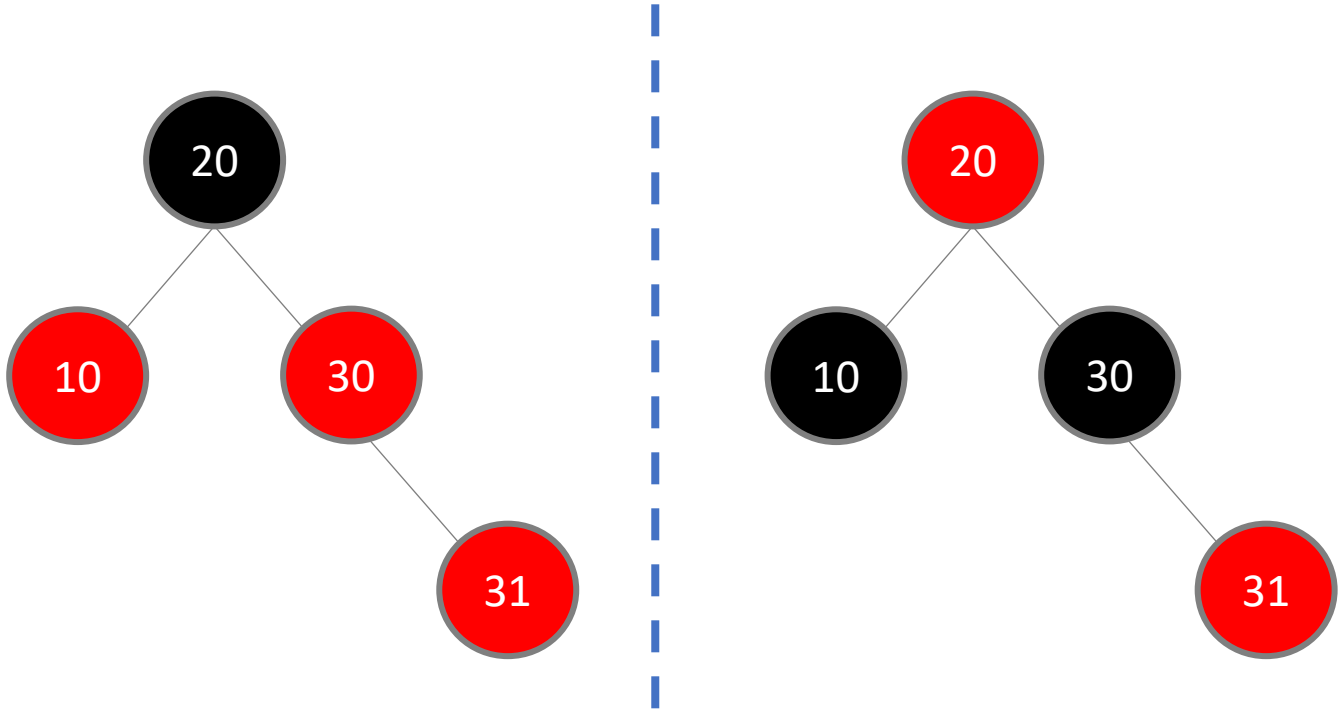
III. O nó **raíz** é sempre terá **cor preta**

IV. **Não há** dois nós **vermelhos adjacentes**, ou seja se um nó é vermelho, então ambos os filhos são pretos

V. O número de nós pretos em qualquer caminho da raiz (de toda a árvore ou subárvores) a uma folha é o mesmo.

VI. Todas as **folhas** são **pretas**

2 – Se o tio for vermelho e o avô for preto: Recolorir



Passo a Passo:

1 – Troca a cor do Pai e do Tio p/ preto

2 – Trocar a cor do avô p/ vermelho

3 – Analise agora o avô (se o avô for a raiz, colorir de preto)

Regras (propriedades)

I. Ser uma árvore binária

II. Cada nó deve ter uma cor: **Preta** ou **Vermelha**

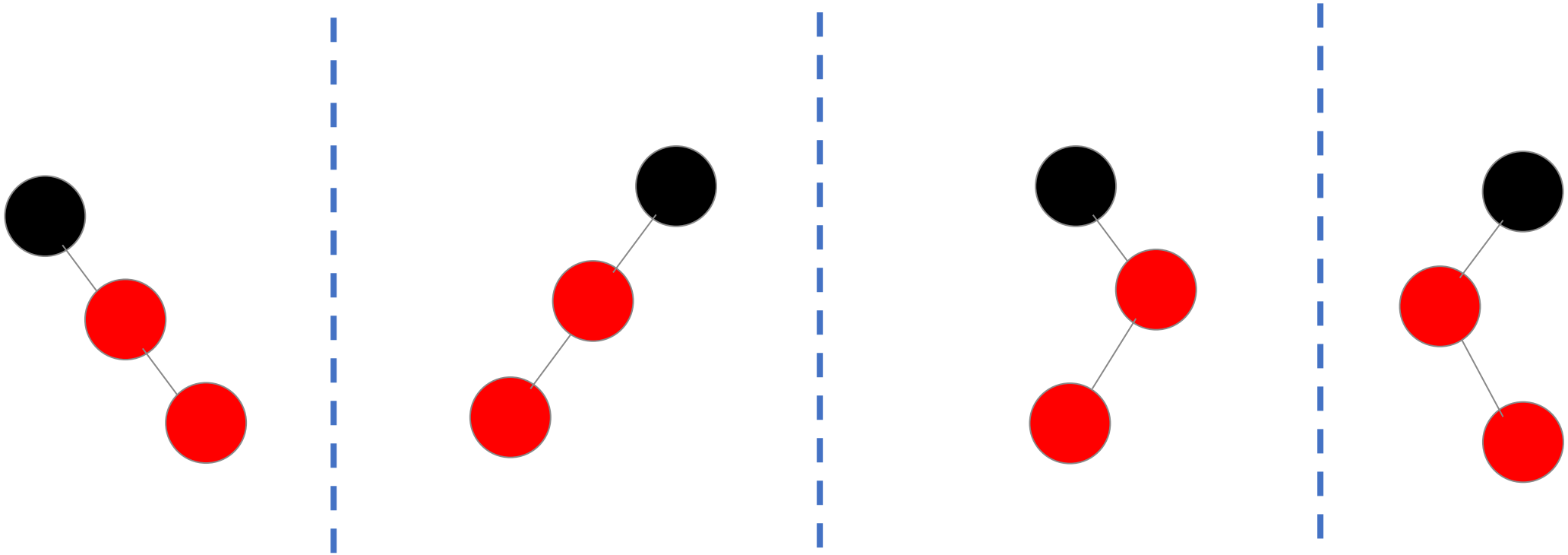
III. O nó **raíz** é sempre terá **cor preta**

IV. **Não há** dois nós **vermelhos adjacentes**, ou seja se um nó é vermelho, então ambos os filhos são pretos

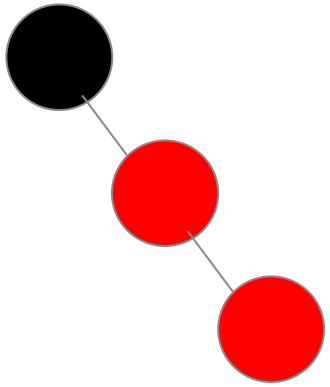
V. O número de nós pretos em qualquer caminho da raiz (de toda a árvore ou subárvores) a uma folha é o mesmo.

VI. Todas as **folhas** são **pretas**

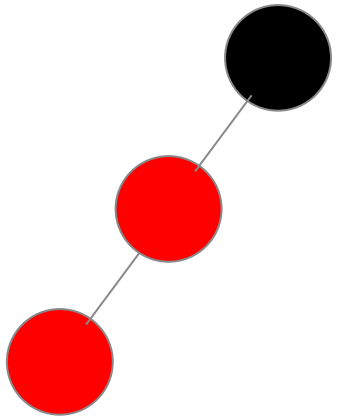
3 – Se o tio for preto (ou não existir), temos 4 situações: ou recolorimos ou rotacionamos



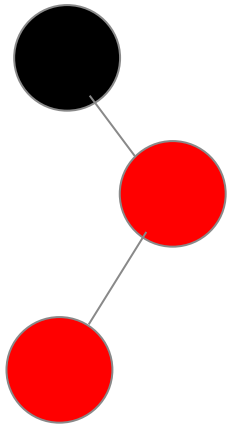
3 – Se o tio for preto (ou não existir), temos 4 situações: ou recolorimos ou rotacionamos



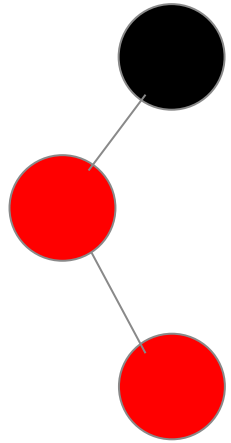
Pai a direita e Filho a direita



Pai a esquerda e Filho a esquerda



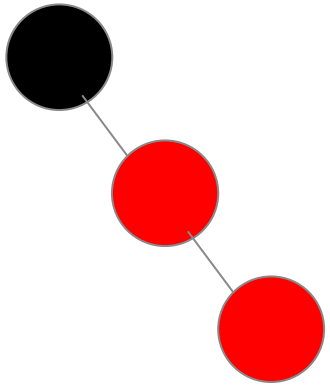
Pai a direita e Filho a esquerda



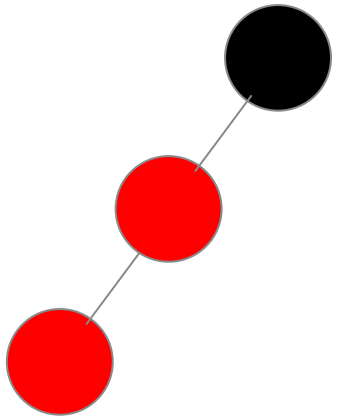
Pai a esquerda e Filho a direita

3 – Se o tio for preto (ou não existir), temos 4 situações: ou recolorimos ou rotacionamos

Inclusão do nó no mesmo lado do Pai

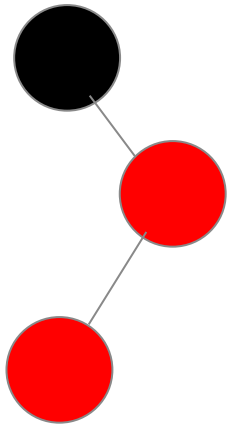


Pai a direita e Filho a direita

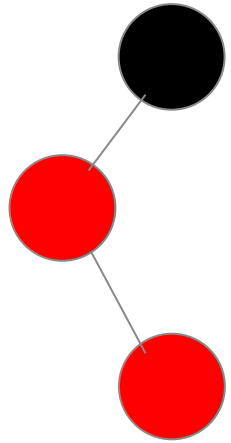


Pai a esquerda e Filho a esquerda

Inclusão do nó do lado inverso ao Pai



Pai a direita e Filho a esquerda

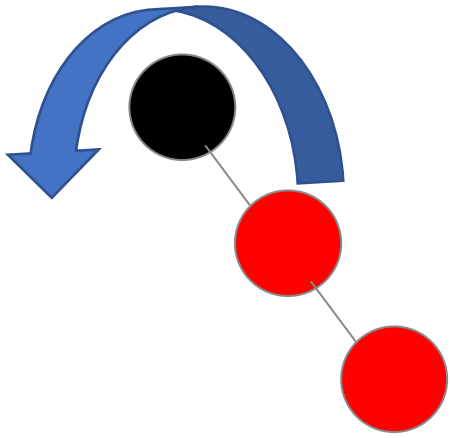


Pai a esquerda e Filho a direita

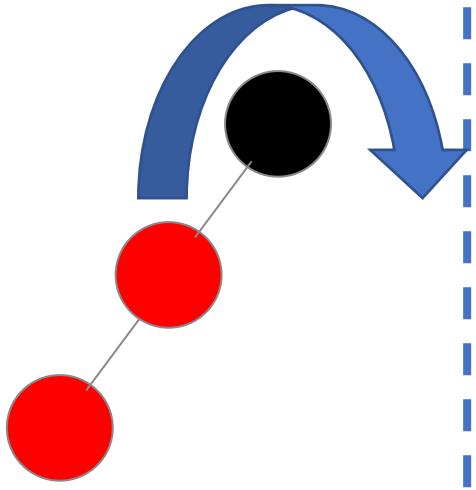
3 – Se o tio for preto (ou não existir), temos 4 situações: ou recolorimos ou rotacionamos

Rotação Simples

Inclusão do nó no mesmo lado do Pai



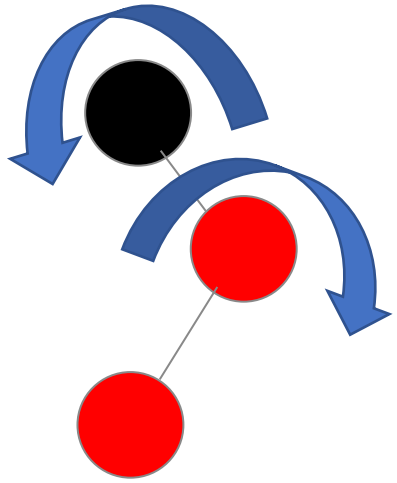
Pai a direita e Filho a direita



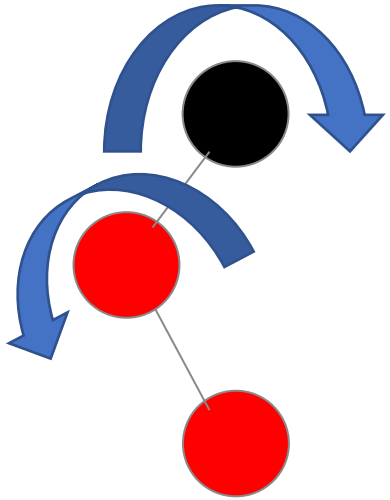
Pai a esquerda e Filho a esquerda

Rotação Dupla

Inclusão do nó do lado inverso ao Pai



Pai a direita e Filho a esquerda



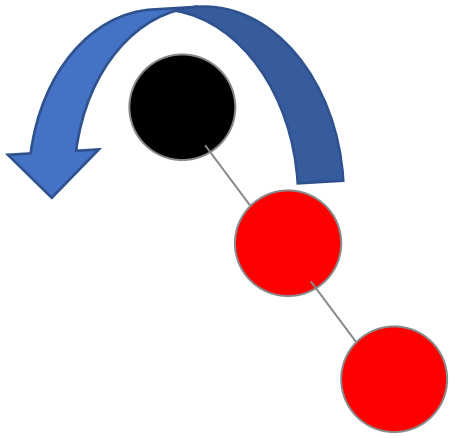
Pai a esquerda e Filho a direita

3 – Se o tio for preto (ou não existir), temos 4 situações: ou recolorimos ou rotacionamos

Rotação Simples

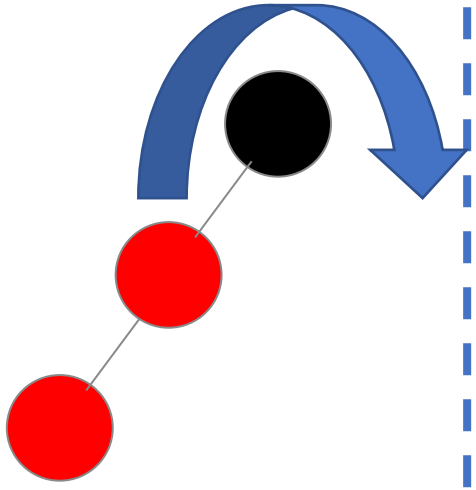
Inclusão do nó no mesmo lado do Pai

Rotação Esquerda



Pai a direita e Filho a direita

Rotação Direita

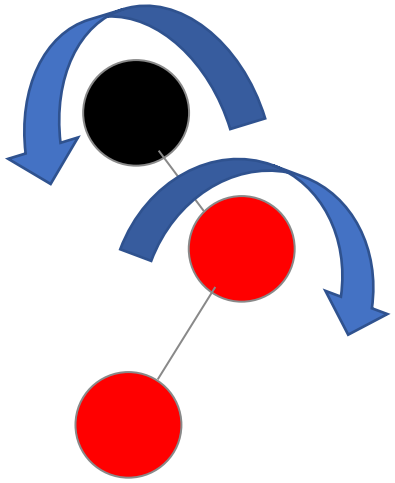


Pai a esquerda e Filho a esquerda

Rotação Dupla

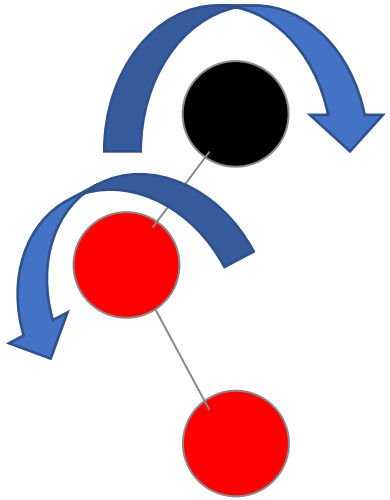
Inclusão do nó do lado inverso ao Pai

Rotação Direita - Esquerda



Pai a direita e Filho a esquerda

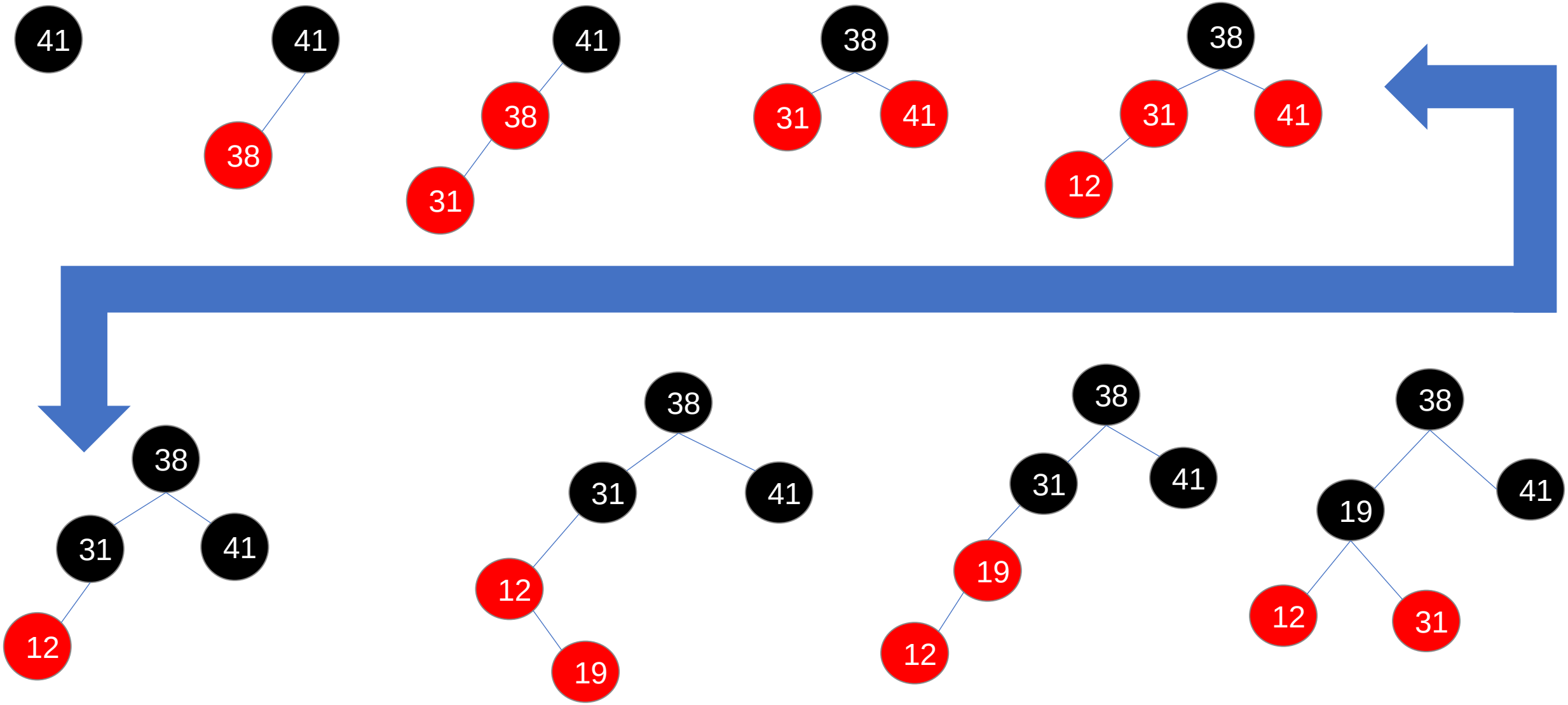
Rotação Esquerda - Direita



Pai a esquerda e Filho a direita

Vamos para um exemplo?

Inserindo os elementos 41 38 31 12 19



Fonte: Material Prof. Jairo Souza

Fontes de Referência

- KOFFMAN, Elliot B.; WOLFGANG, Paul AT. Objetos, Abstração, Estruturas de Dados E Projeto Usando C+. Grupo Gen-LTC, 2000.