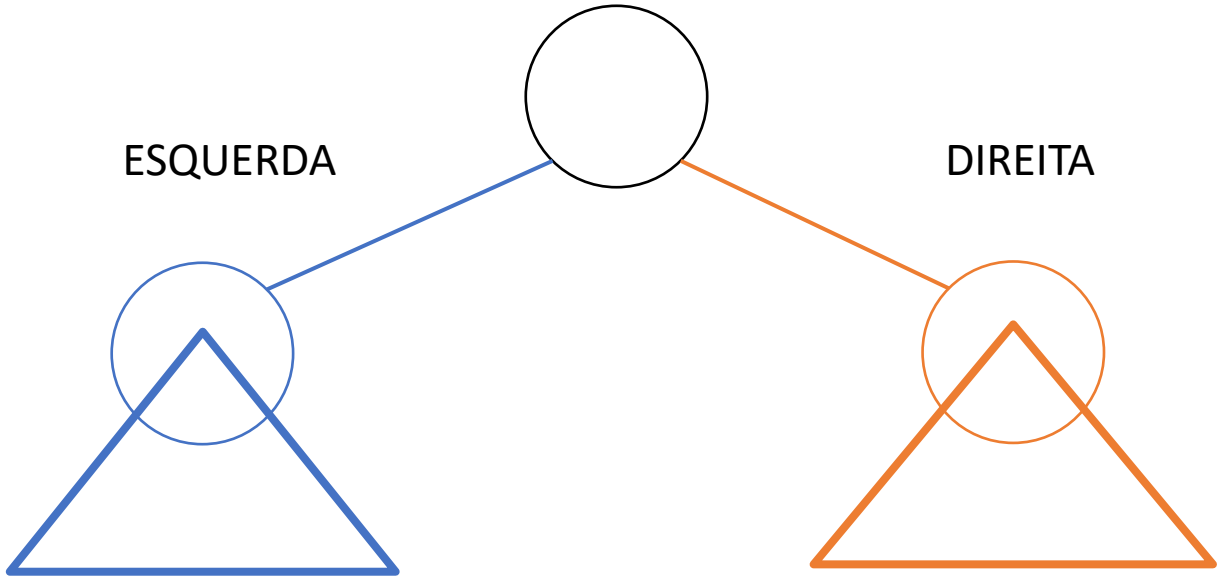
Estruturas de Dados

Árvores Rubro-negras

Universidade Federal do Rio Grande do Norte

Árvores Binárias



Árvores Binárias

- Uma **árvore binária** é um conjunto finito de elementos que está **vazio** ou é **particionado** em **três subconjuntos disjuntos**.

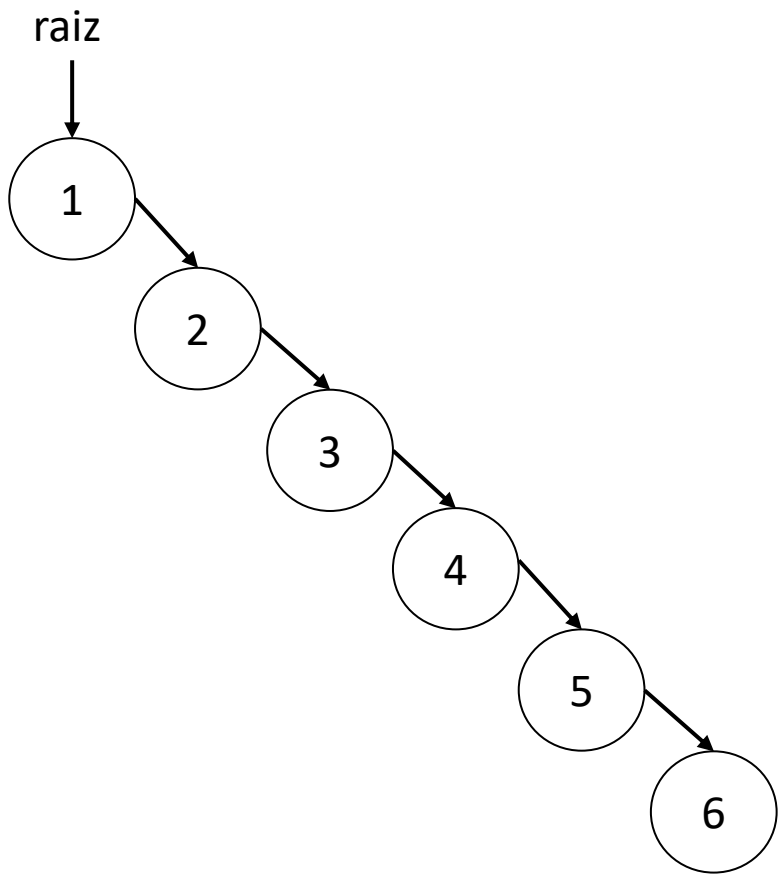
FONTE: https://www.ime.usp.br/~pf/algoritmos/aulas/bint.html

Árvores Binárias

- Altura
- Altura da raiz
- Varredura

Árvores Binárias de Busca

- Uma busca nessa árvore deve realizar uma busca nos N nós.
- Nessa caso N-1 equivale a altura da raiz.

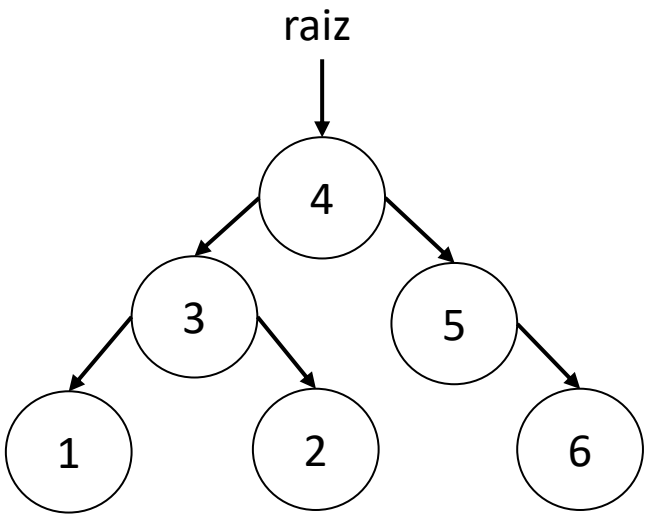


Árvores Binárias de Busca

- Operações de busca, inserção, remoção, etc. levam um tempo proporcional a altura da árvore.
- **Por isso há a necessidade de manter a árvore com a menor altura o possível.**

Árvores Binárias de Busca

- Ideal, altura 2, N = 6



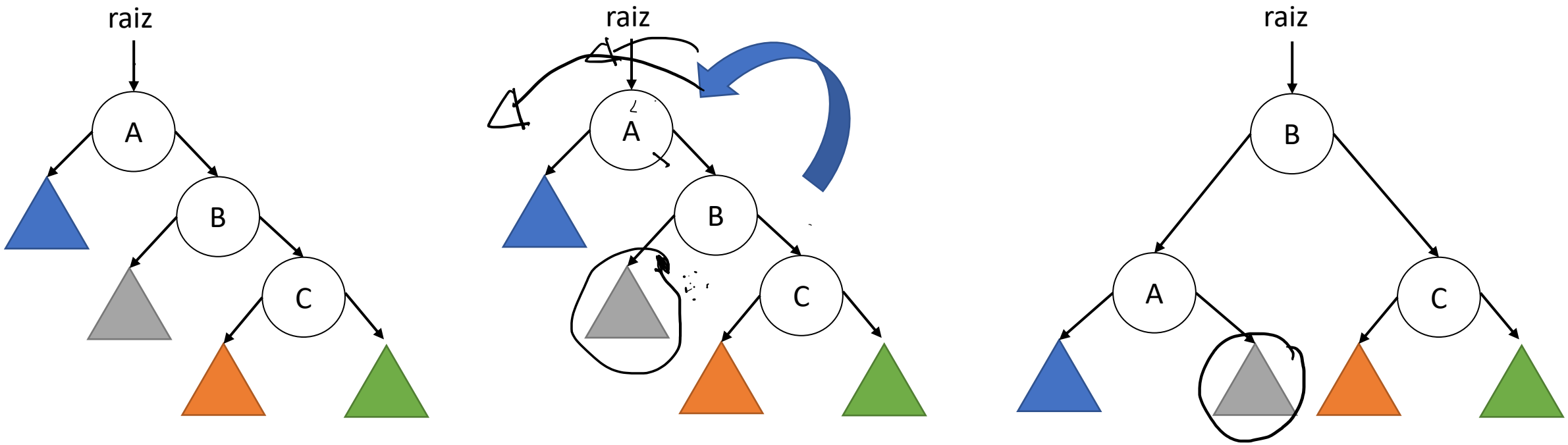
Árvores Binárias Balanceadas

- Uma árvore binária é balanceada se, em cada um de seus nós, as subárvores esquerda e direita tiverem **aproximadamente a mesma altura**.
- Uma árvore binária balanceada com **n** nós tem altura próxima de **lg n.**

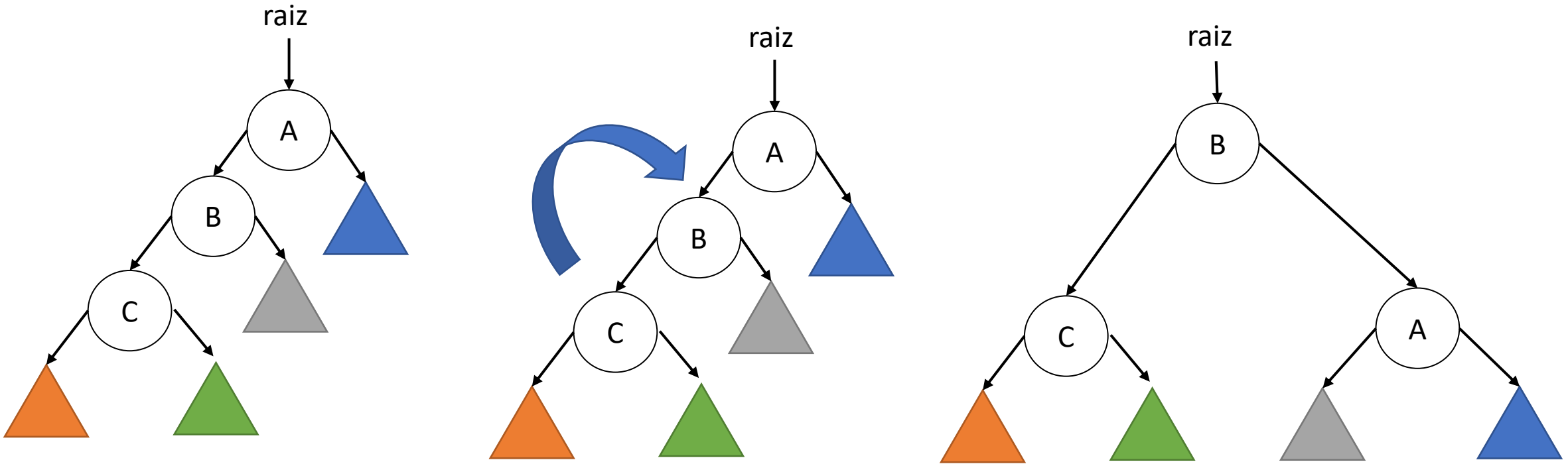
Árvores AVL

- O nome AVL vem dos desenvolvedores **Adelson Velsky e Landis** no artigo "Algoritmos para organização da informação" de 1962.
- Ao realizar operações de inserção ou remoção de nós é preciso garantir o equilíbrio.
- Passo 1: Detectar desequilíbrio.
- Passo 2: Corrigir um desequilíbrio.

Rotação à esquerda



Rotação à direita



Árvores Rubro-negras

- Uma árvore rubro-negra é uma **árvore binária de busca** que insere e remove elementos para assegurar que a árvore permaneça **aproximadamente balanceada.**
- Uma árvore rubro-negra possui um **bit extra** de armazenamento por nó para representar a sua co
  - **vermelho** ou **preto**.

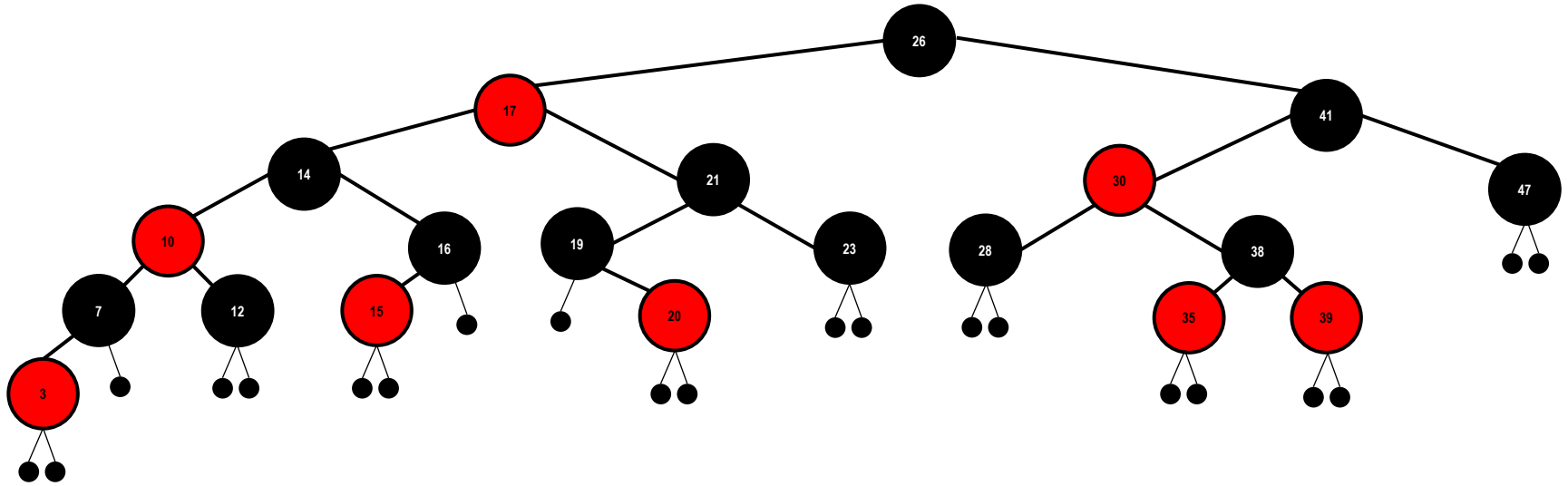
Árvores Rubro-negras

- Através da restrição de como os nós podem ser coloridos em qualquer caminho desde a raiz até uma folha é possível assegurar que **nenhum caminho será duas vezes maior que o comprimento de qualquer outro**.
- Por isso que dizemos que essa é uma árvore **aproximadamente balanceada**.

Propriedades das árvores rubro-negras

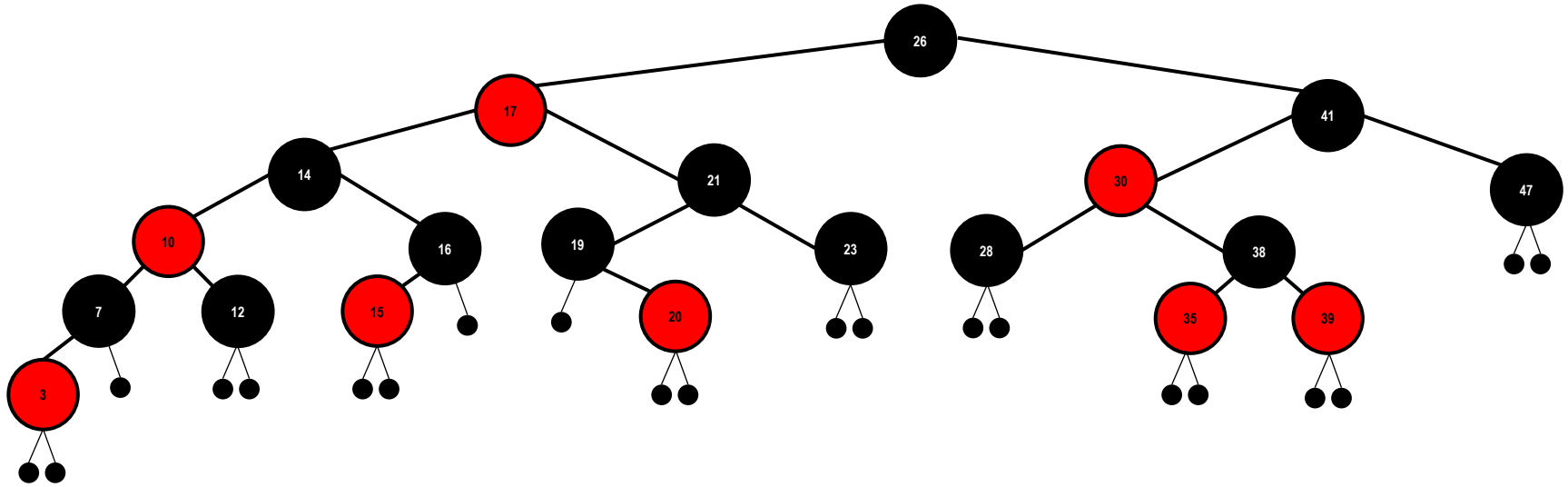
- Todo nó é preto ou vermelho.
- A raiz é preta.
- Todos os nós NULL (folha) são pretos.
- Se um nó é vermelho, ambos os seus filhos são pretos.
- Para cada nó, todos os nós do caminho da raiz até a as folhas descendentes possuem o mesmo número de nós pretos.

Árvores Rubro-negras



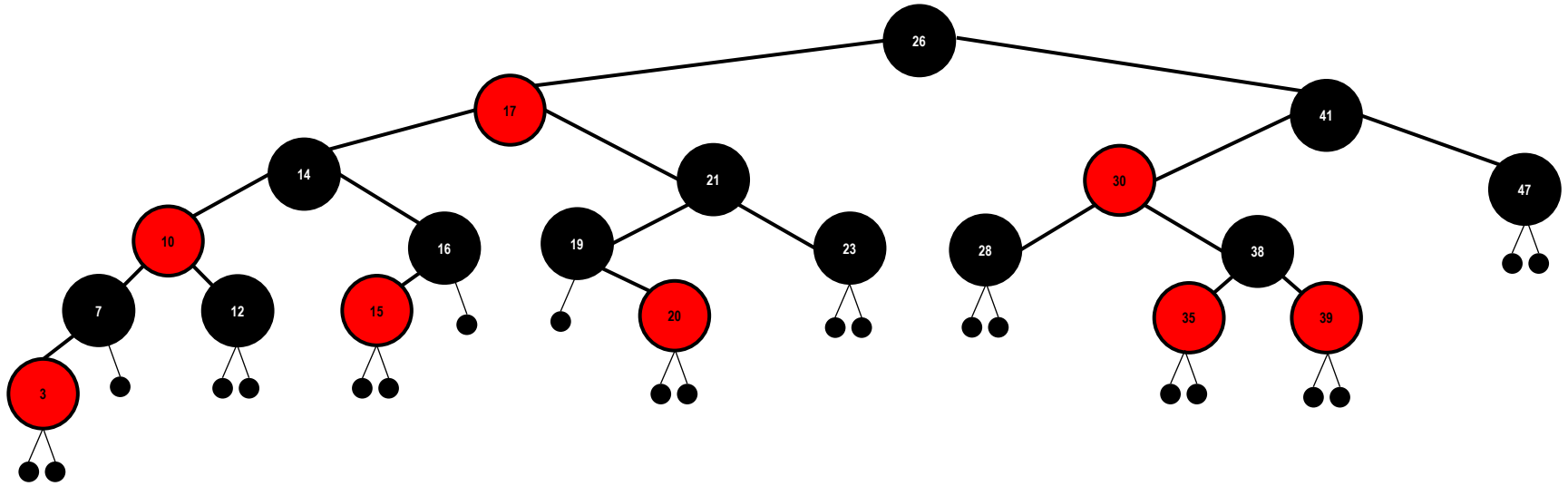
1) Todo nó é preto ou vermelho.

Árvores Rubro-negras



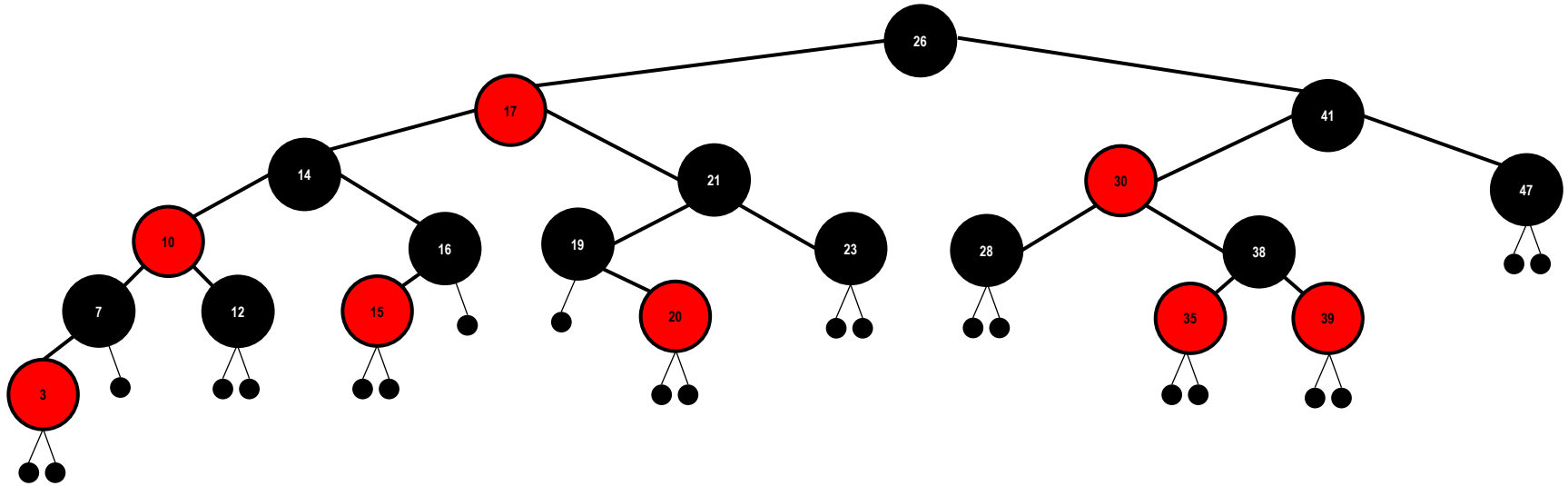
2) A raiz é preta.

Árvores Rubro-negras



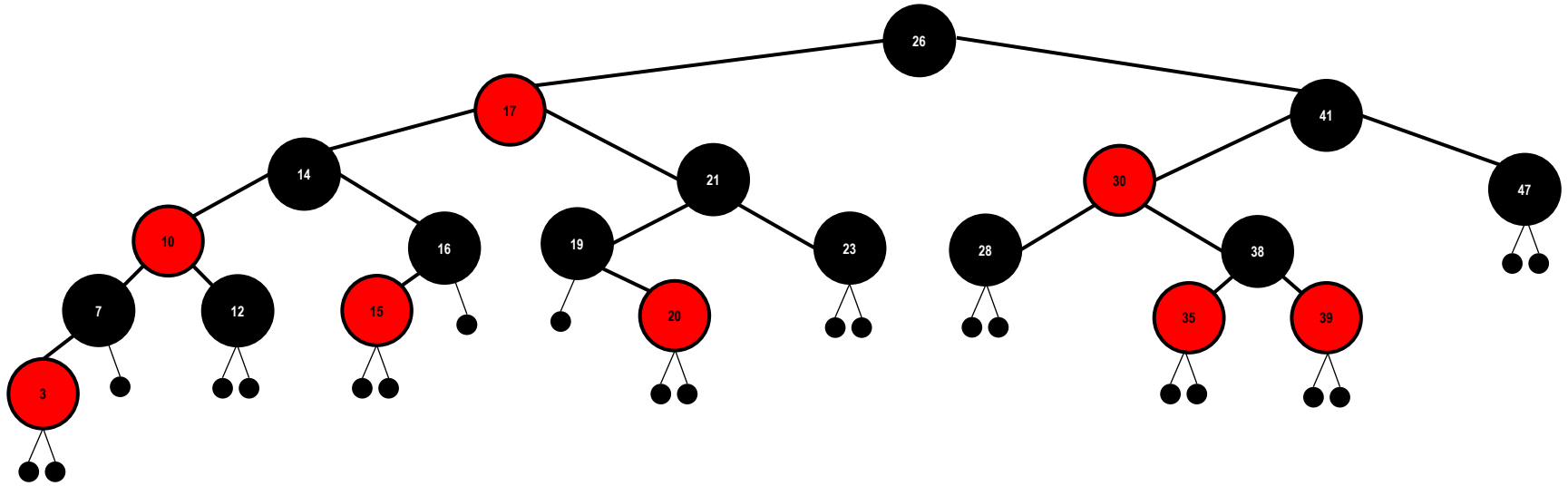
3) Todos os nós NULL (folha) são pretos.

Árvores Rubro-negras



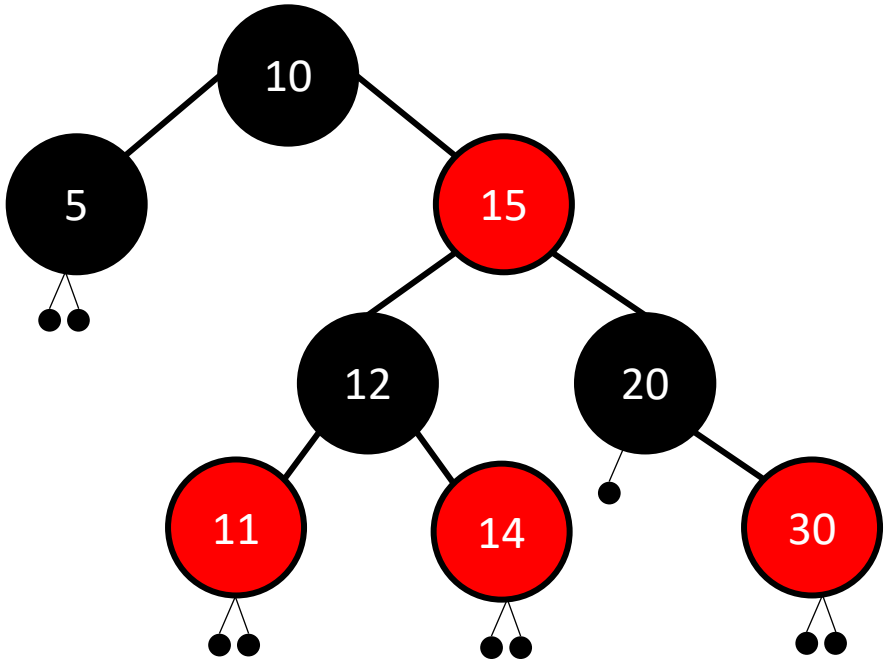
4) Se um nó é vermelho, ambos os seus filhos são pretos.

Árvores Rubro-negras



5) Para cada nó, todos os nós do caminho da raiz até a as folhas descendentes possuem o mesmo número de nós pretos.

Árvores Rubro-negras



Operações em árvores Rubro-negras

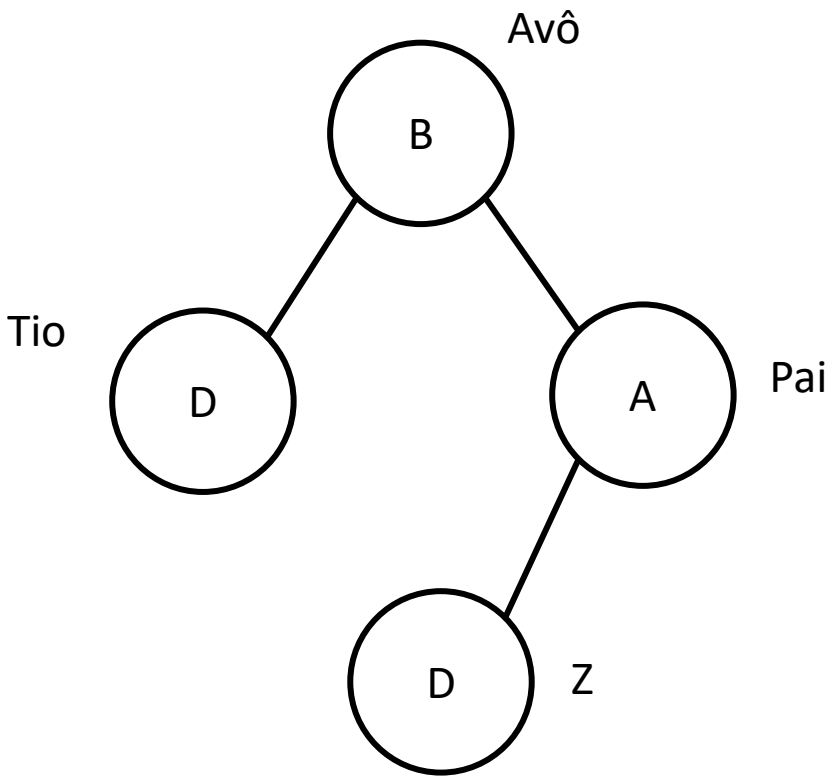
- Inserção O (lg n)
- Remoção O (lg n)
- Busca

- Na inserção e remoção é necessário verificar se as propriedades são respeitadas. Se não forem será necessário **recolorir** nós e **realizar rotações**.

Inserção

- A operação de inserção em uma árvore rubro-negra acontece em dois passos:
  - Inserção como em árvores binárias de busca
  - Coloração do nó inserido como vermelho
  - Correção de cores e balanceamento
- A inserção é realizada como em árvores binárias.
- Vamos aprender a correção de cores e balanceamento.

Relações de Z



Inserção

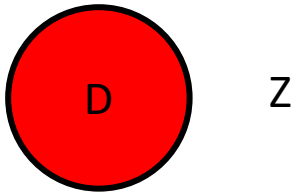
- Quando inserimos um nó vermelho, nós podemos quebrar as propriedades 2 e 4.

1. Todo nó é preto ou vermelho.
2. **A raiz é preta.**
3. Todos os nós NULL (folha) são pretos.
4. **Se um nó é vermelho, ambos os seus filhos são pretos.**
5. Para cada nó, todos os nós do caminho da raiz até a as folhas descendentes possuem o mesmo número de nós pretos.

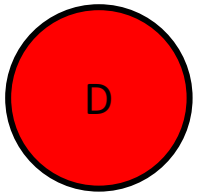
Inserção - Casos

- Z é a raiz
- O tio de Z é vermelho
- O tio de Z é preto (forma triangulo)
- O tio de Z é preto (forma linha)

Caso 1: Z é a raiz



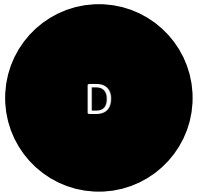
Caso 1: Z é a raiz



Z

Solução: mudar a cor de Z para preto.

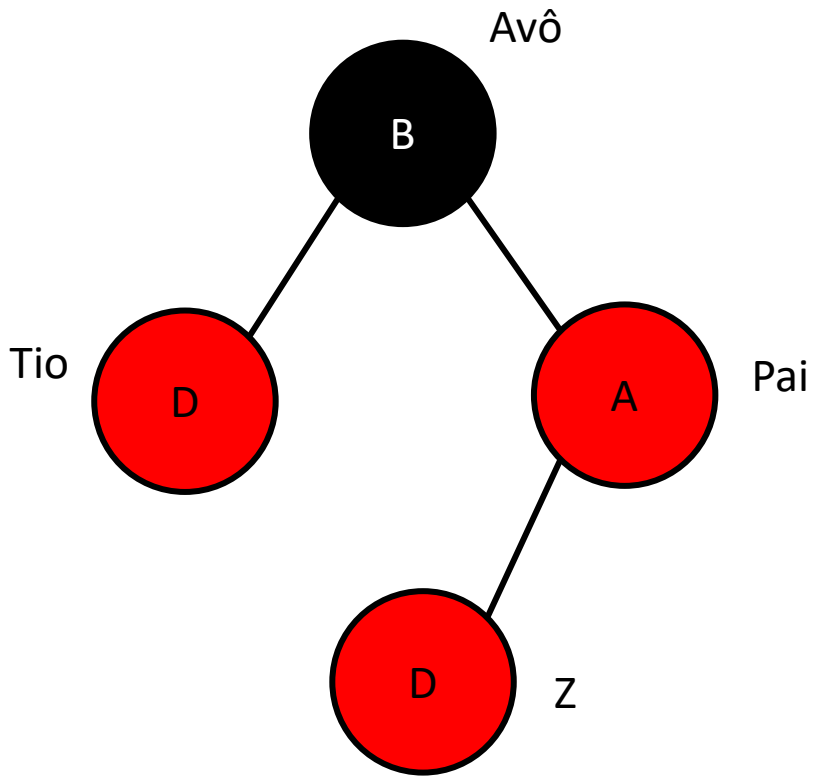
Caso 1: Z é a raiz



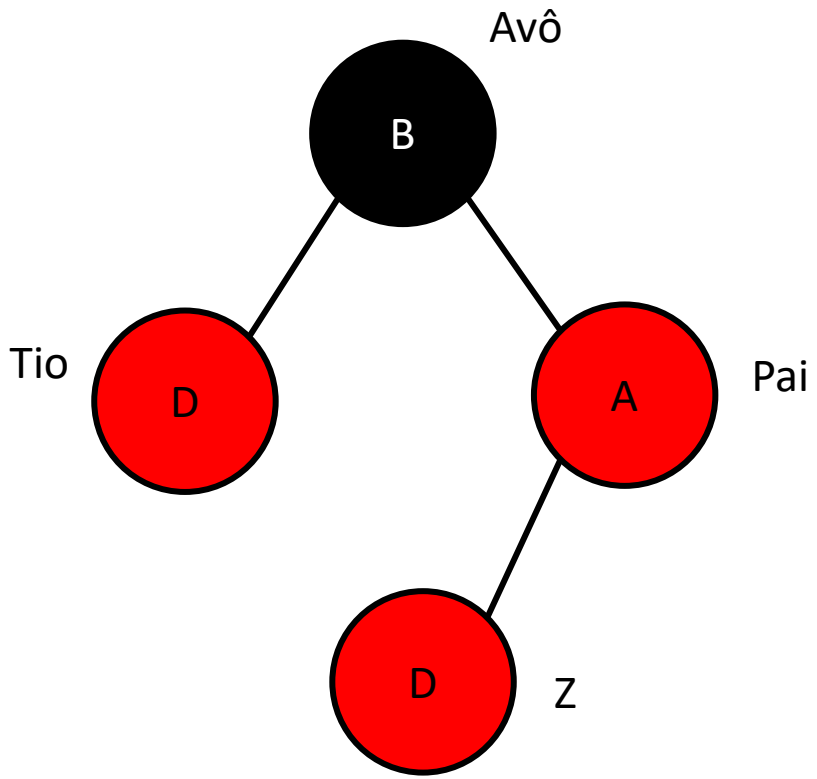
Z

Solução: mudar a cor de Z para preto.

Caso 2: O tio de Z é vermelho

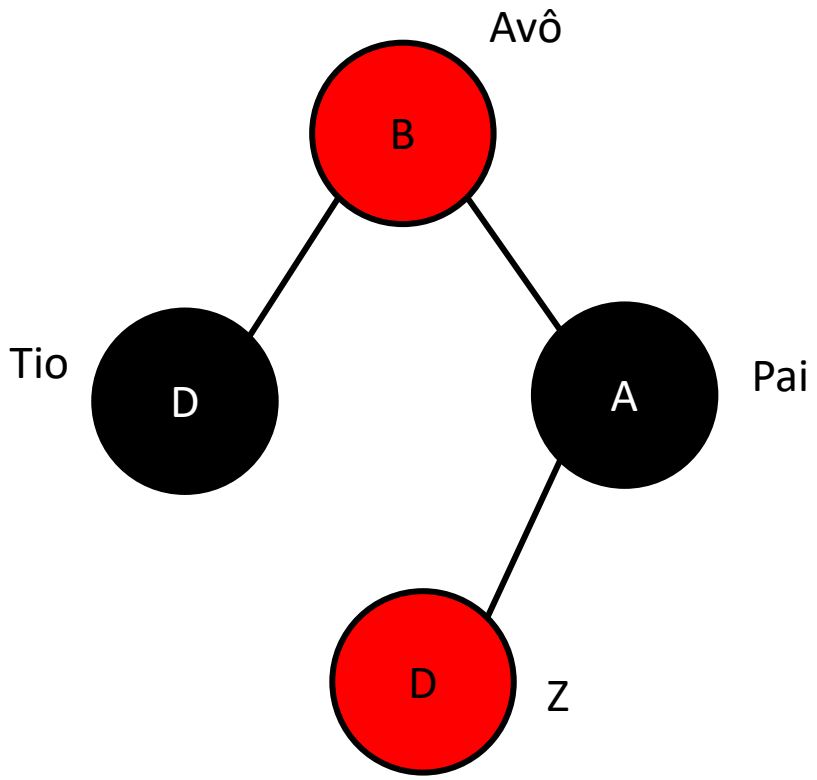


Caso 2: O tio de Z é vermelho



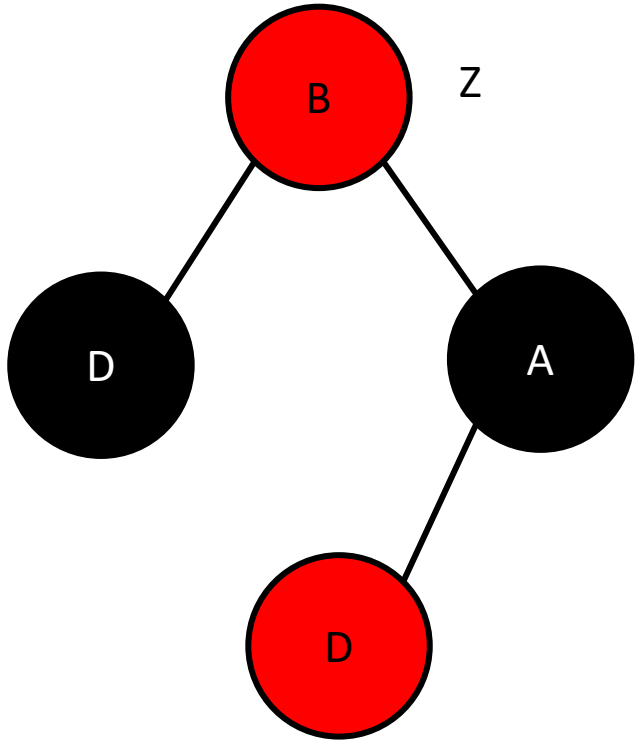
Solução: trocar a cor do pai, do tio e do avô

Caso 2: O tio de Z é vermelho



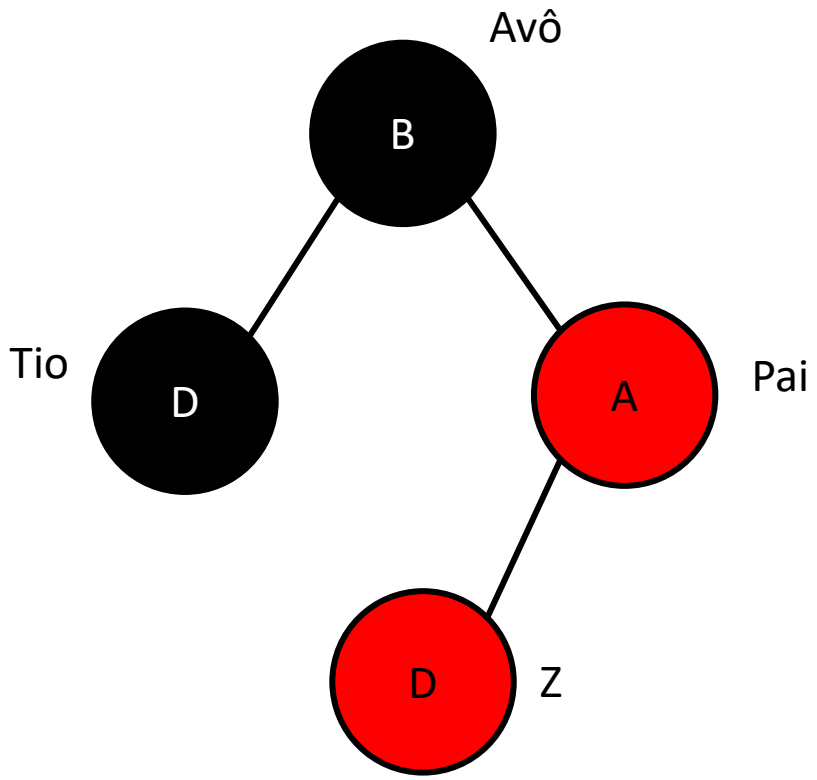
Solução: trocar a cor do pai, do tio e do avô.

Caso 2: O tio de Z é vermelho

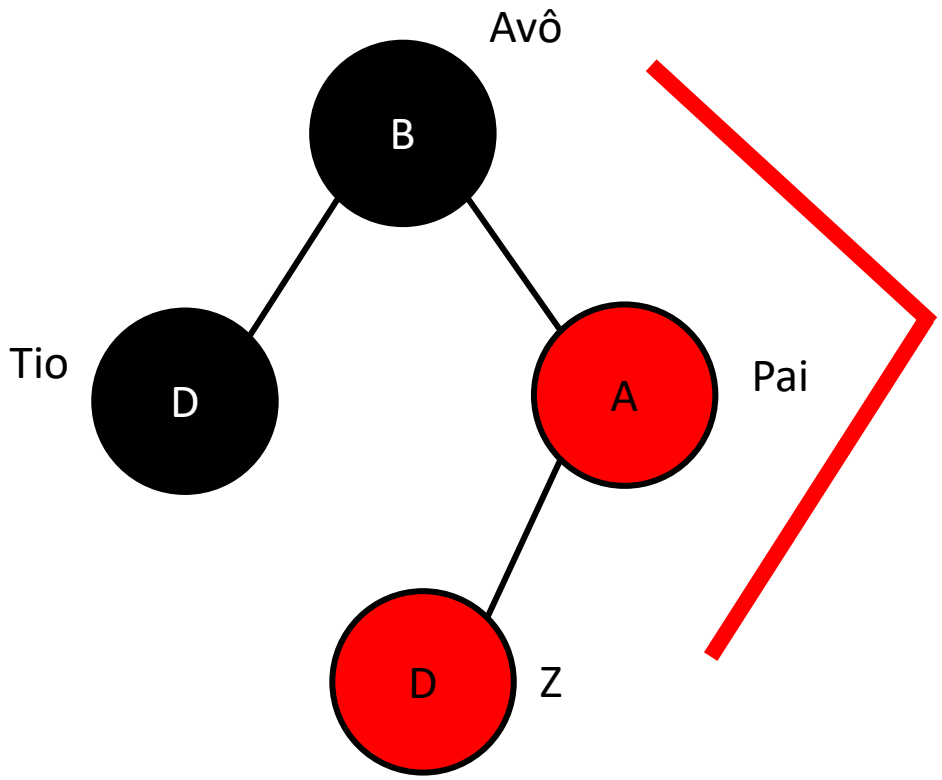


Solução: trocar a cor do pai, do tio e do avô.

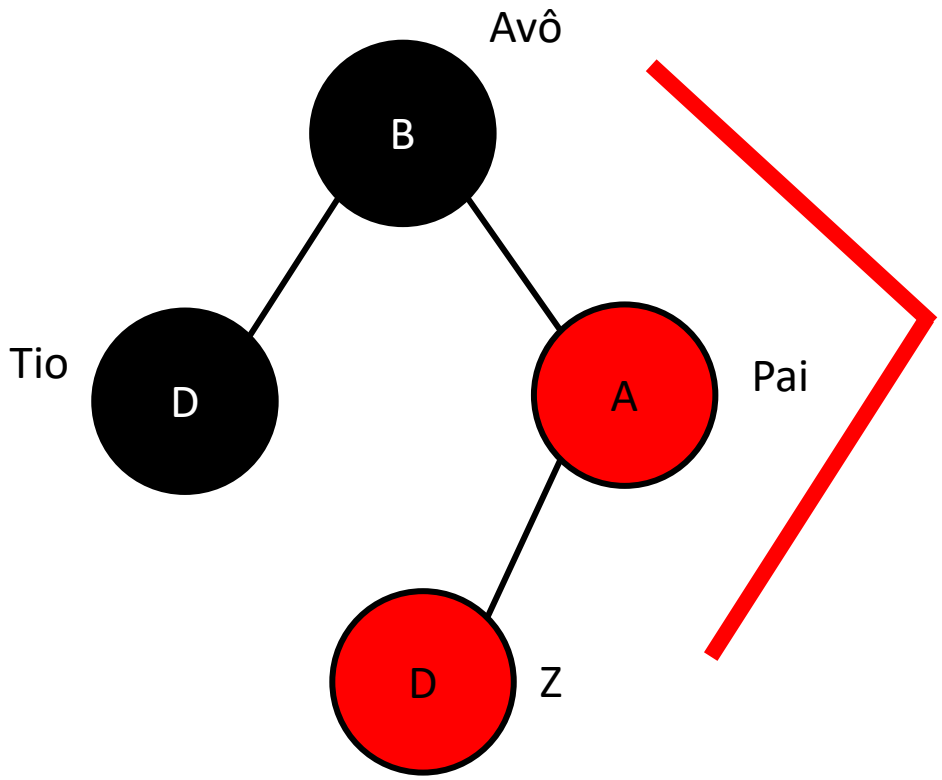
Caso 3: O tio de Z é preto (forma triangulo)



Caso 3: O tio de Z é preto (forma triangulo)

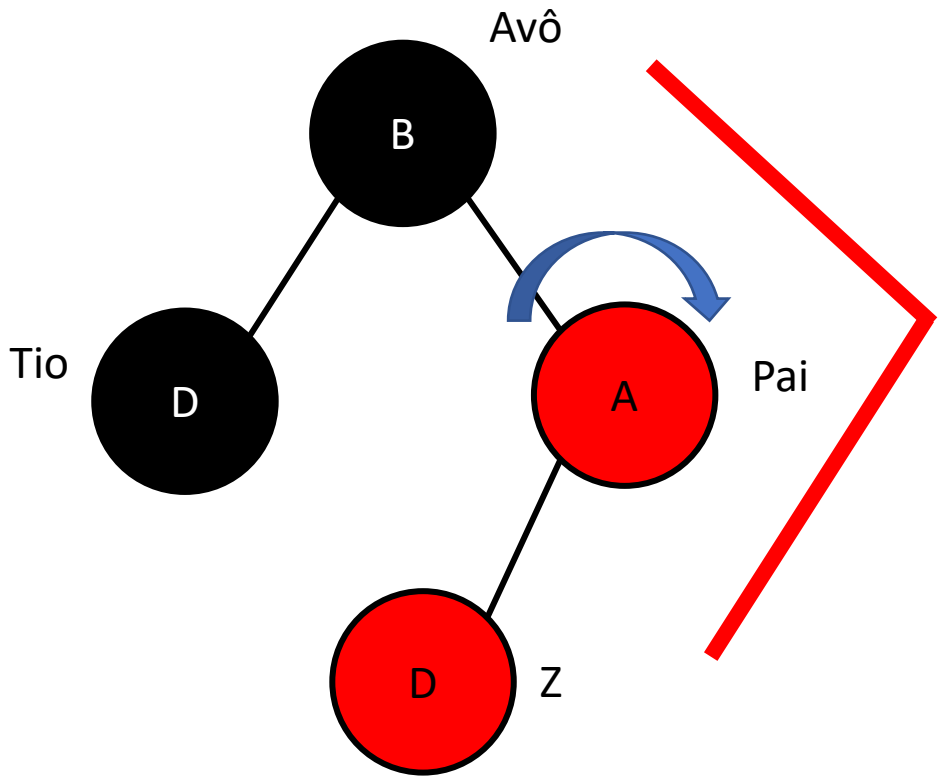


Caso 3: O tio de Z é preto (forma triangulo)



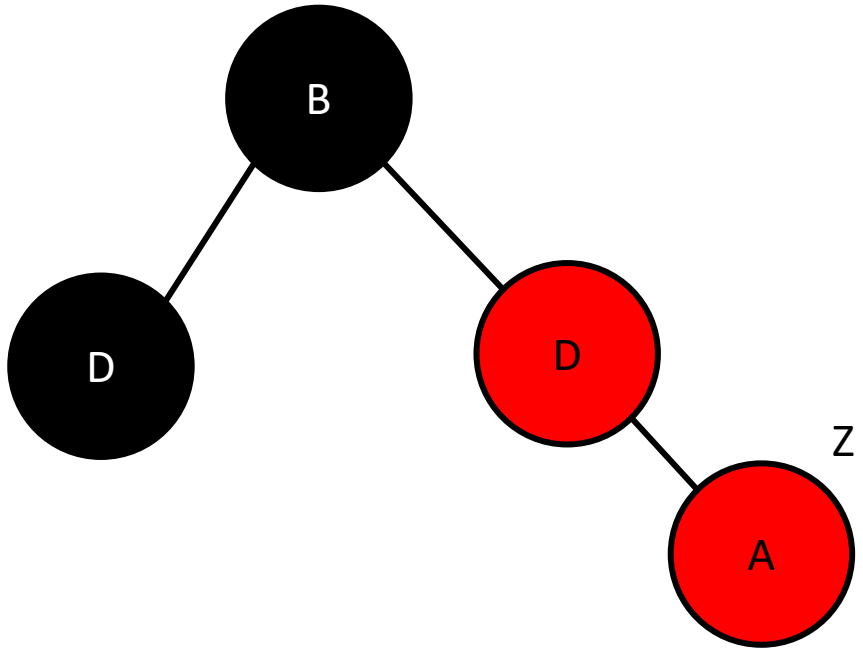
Solução: rotação à direta no pai de Z.

Caso 3: O tio de Z é preto (forma triangulo)



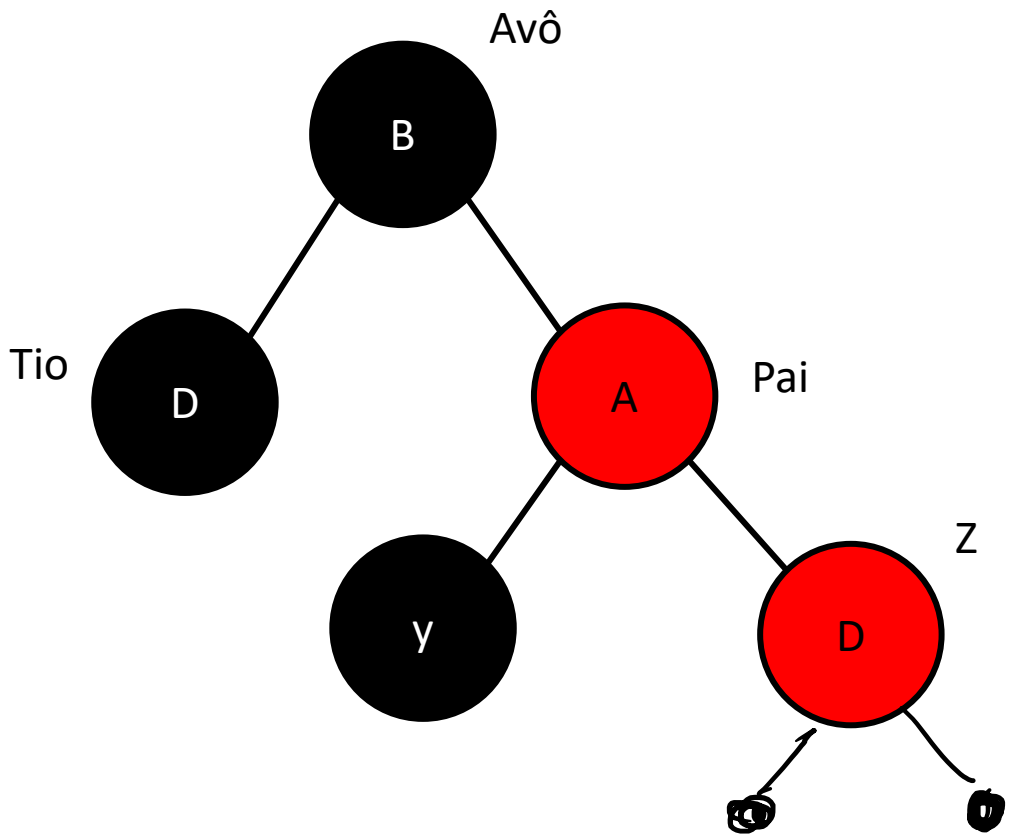
Solução: rotação à direta no pai de Z.

Caso 3: O tio de Z é preto (forma triangulo)

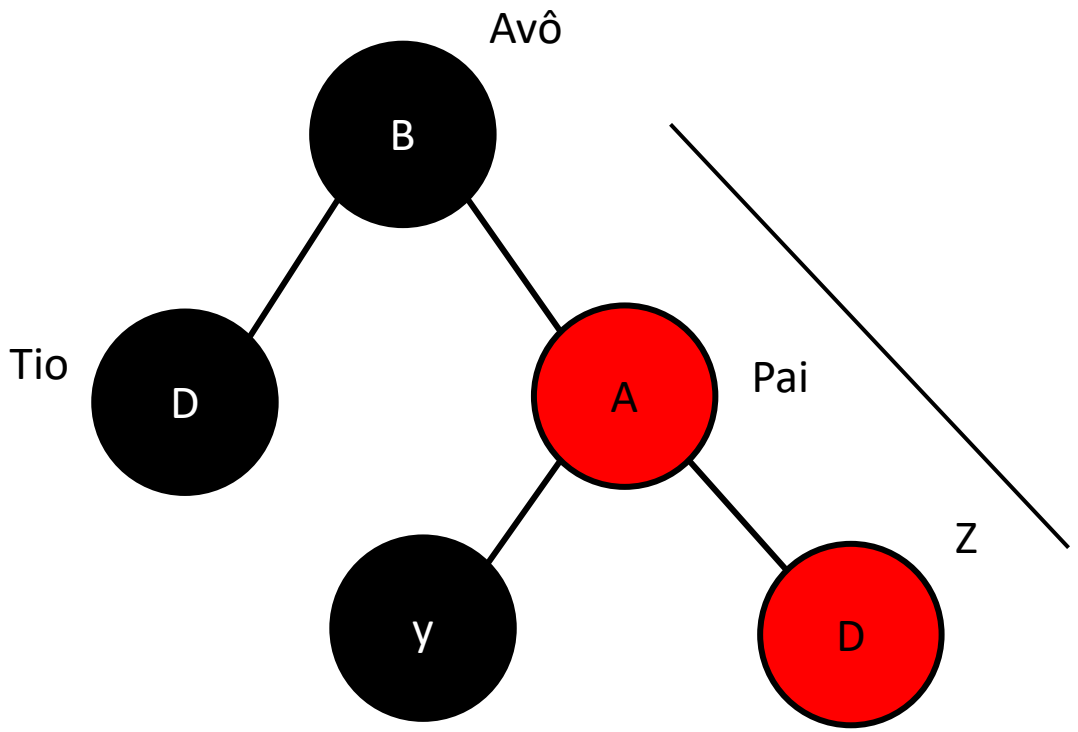


Solução: rotação à direta no pai de Z.

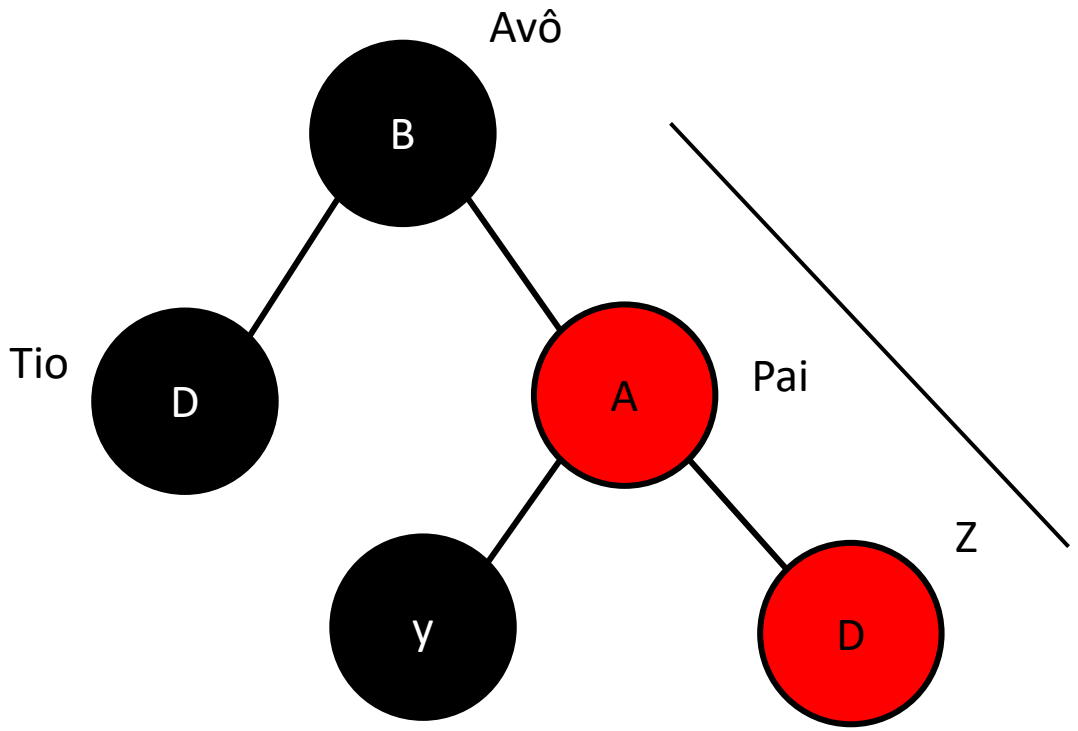
Caso 4: O tio de Z é preto (forma linha)



Caso 4: O tio de Z é preto (forma linha)

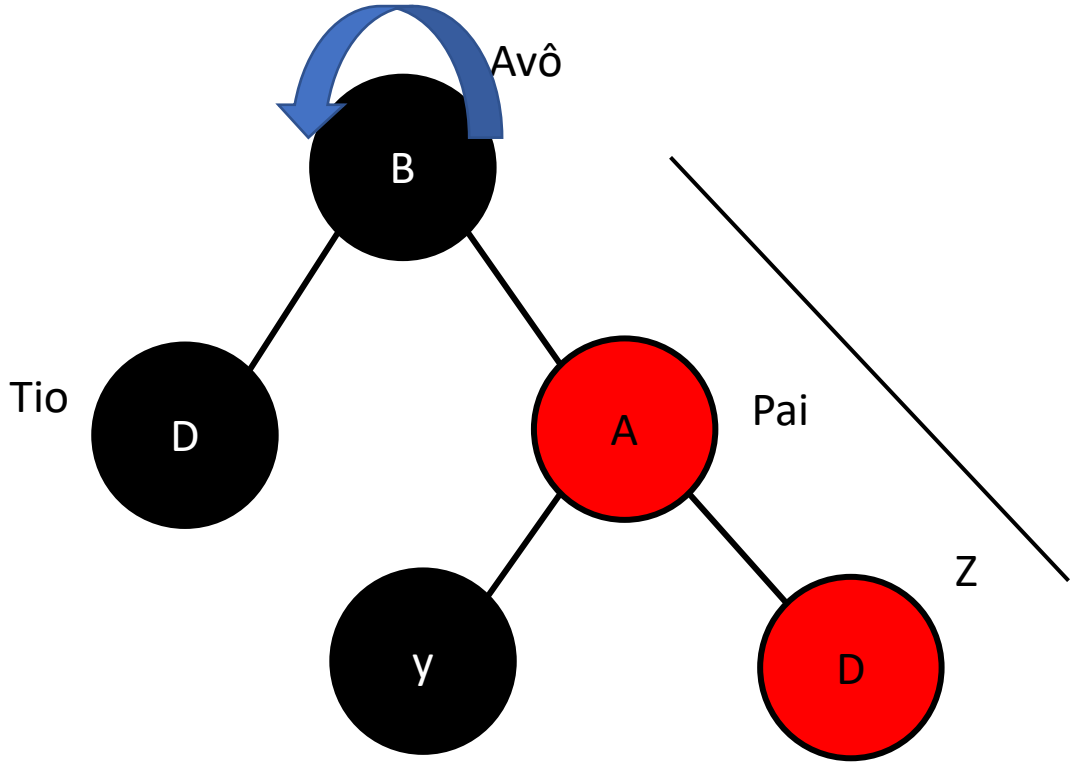


Caso 4: O tio de Z é preto (forma linha)



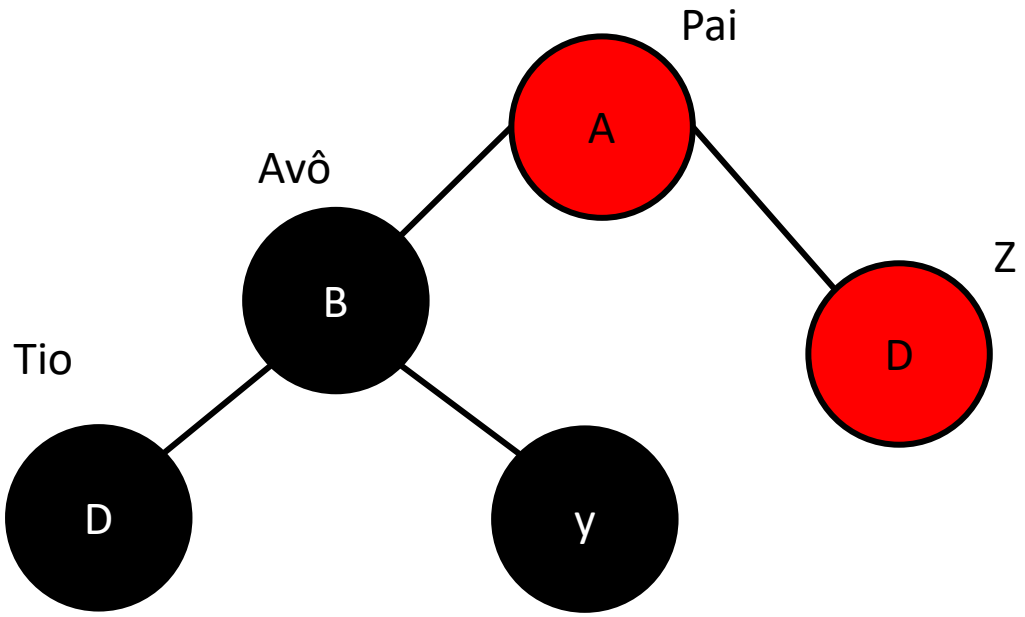
Solução: rotação à esquerda do avô de Z. Recolorir o pai e o avô original.

Caso 4: O tio de Z é preto (forma linha)



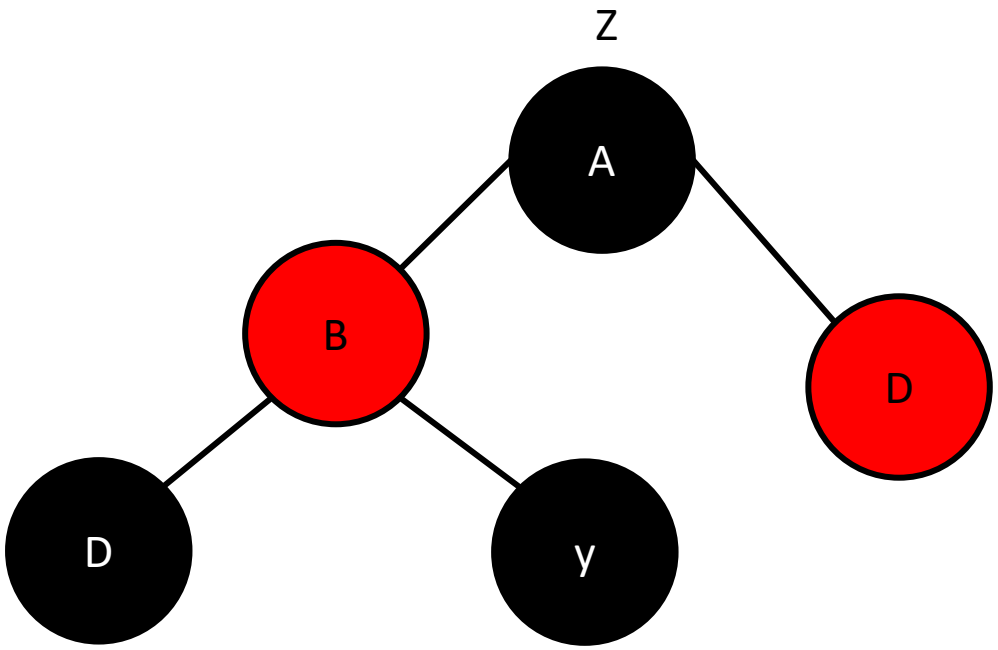
Solução: rotação à esquerda do avô de Z. Recolorir o pai e o avô original.

Caso 4: O tio de Z é preto (forma linha)



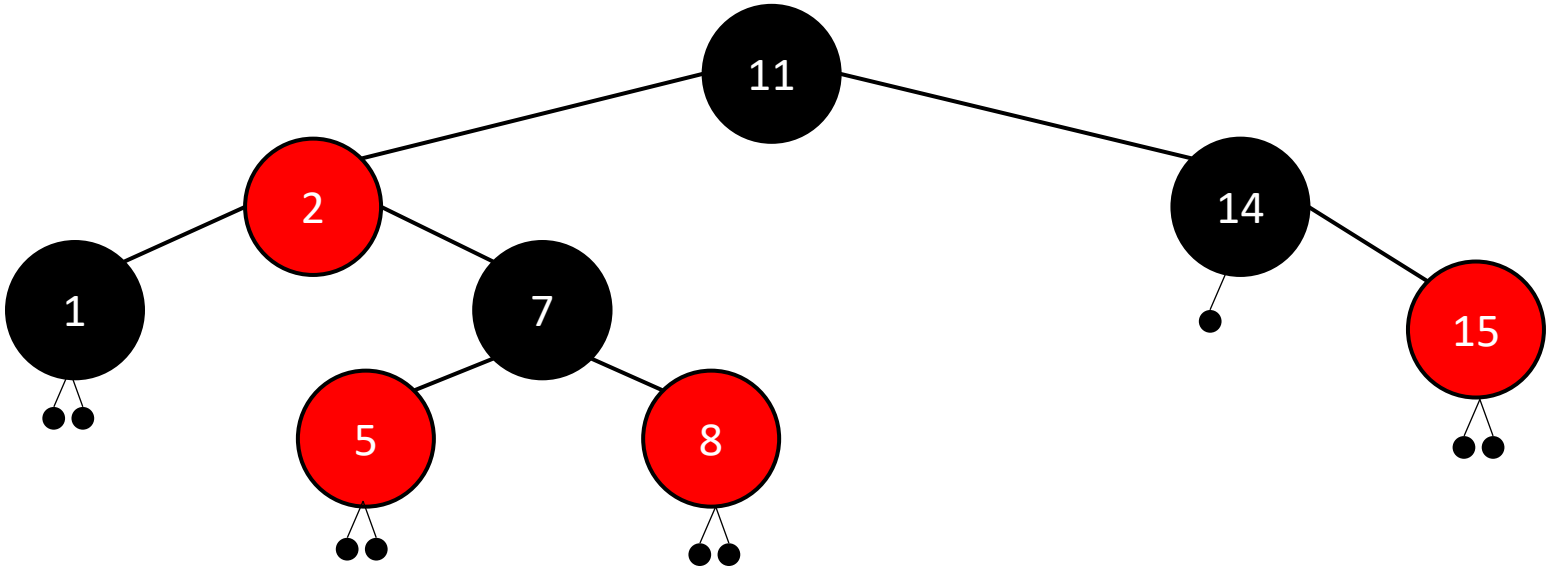
Solução: rotação à esquerda do avô de Z. Recolorir o pai e o avô original.

Caso 4: O tio de Z é preto (forma linha)



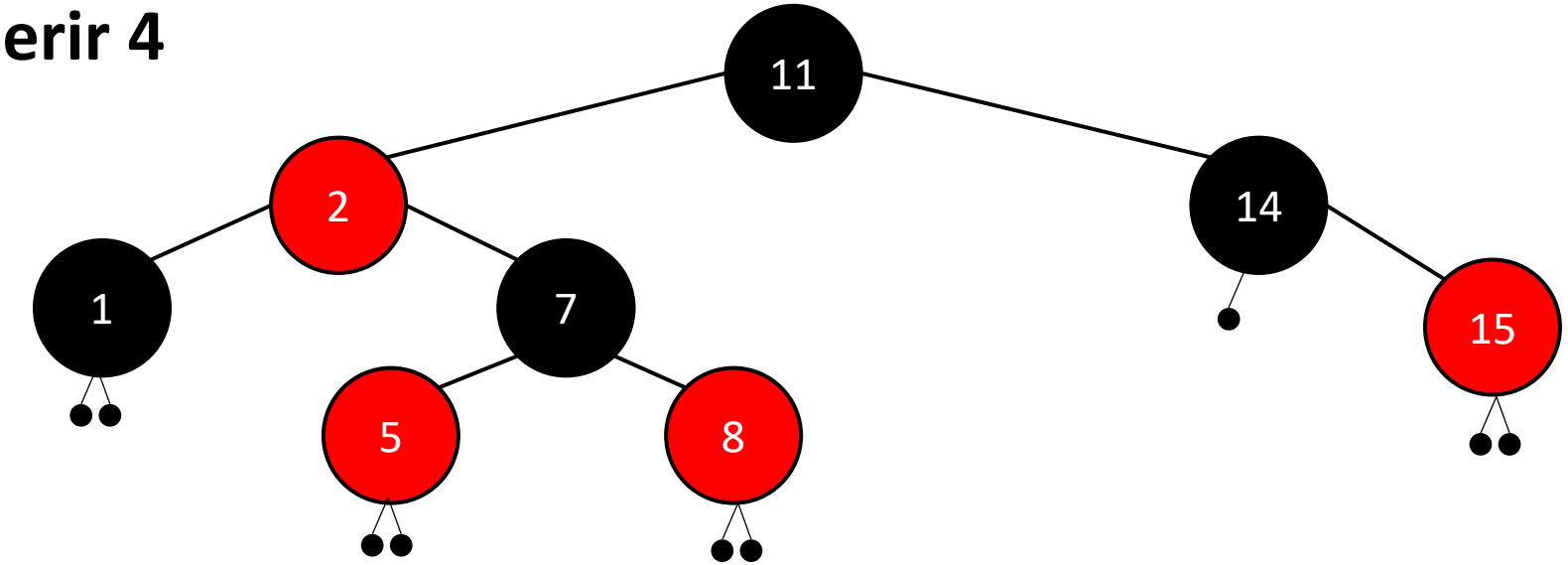
Solução: rotação à esquerda do avô de Z. Recolorir o pai e o avô original.

Exemplo



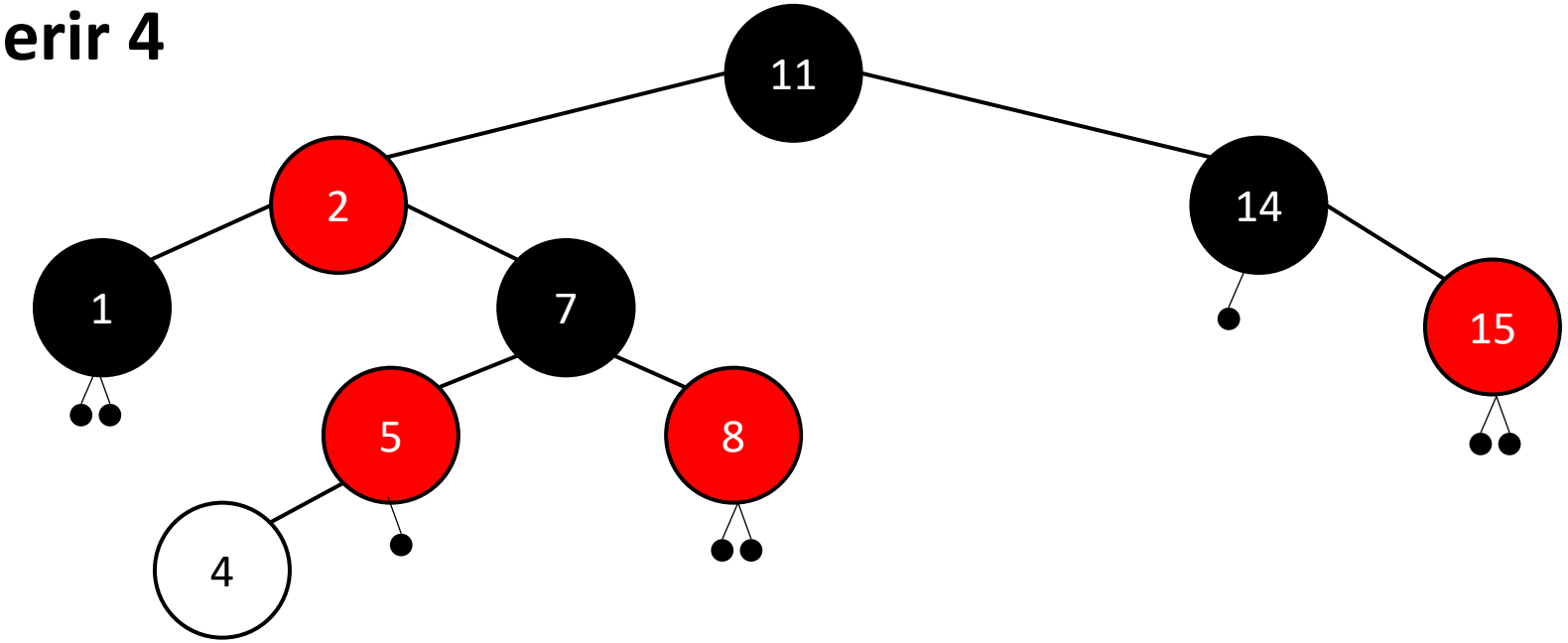
Exemplo

Inserir 4

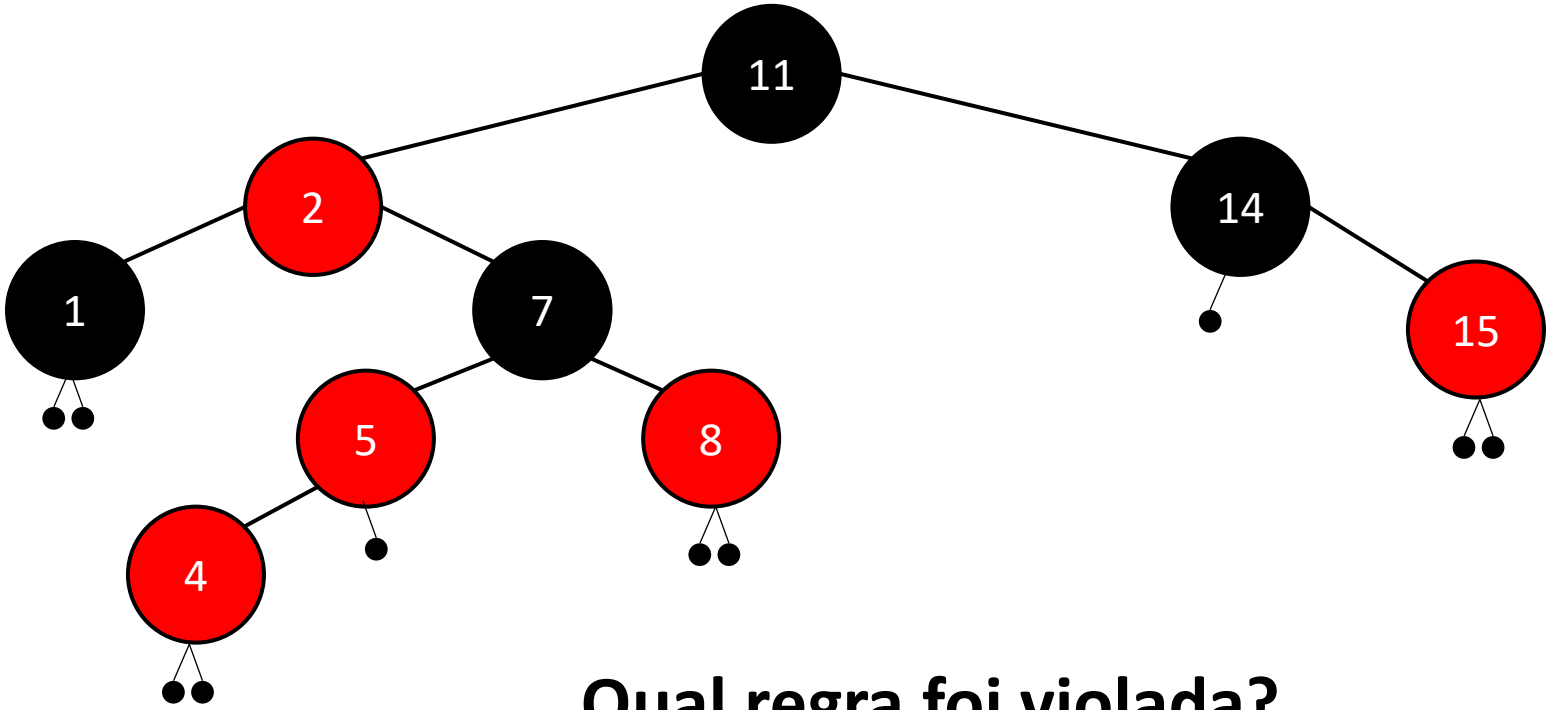


Exemplo

Inserir 4



Exemplo



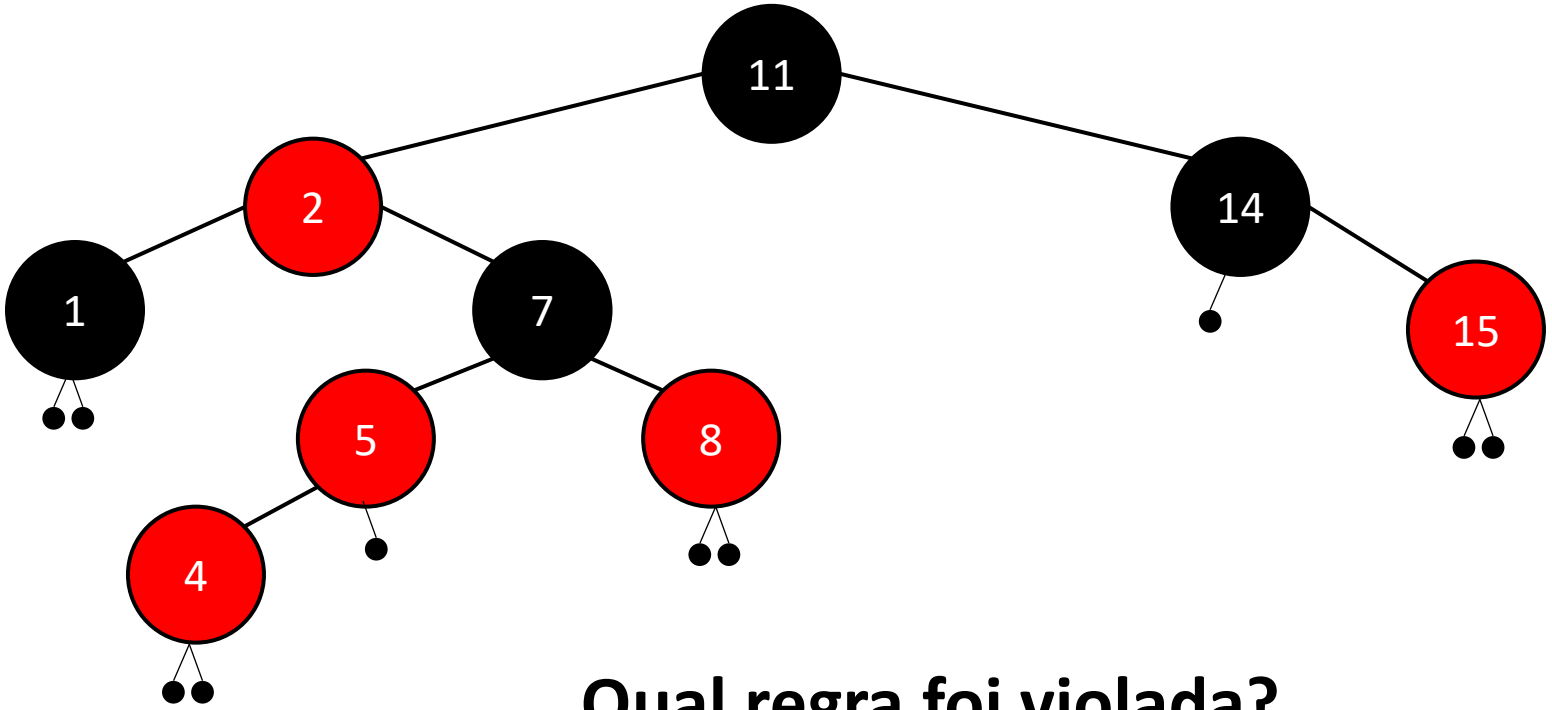
Qual regra foi violada?

Exemplo

- Quando inserimos um nó vermelho, nós podemos quebrar as propriedades 2 e 4.

1. Todo nó é preto ou vermelho.
2. **A raiz é preta.**
3. Todos os nós NULL (folha) são pretos.
4. **Se um nó é vermelho, ambos os seus filhos são pretos.**
5. Para cada nó, todos os nós do caminho da raiz até a as folhas descendentes possuem o mesmo número de nós pretos.

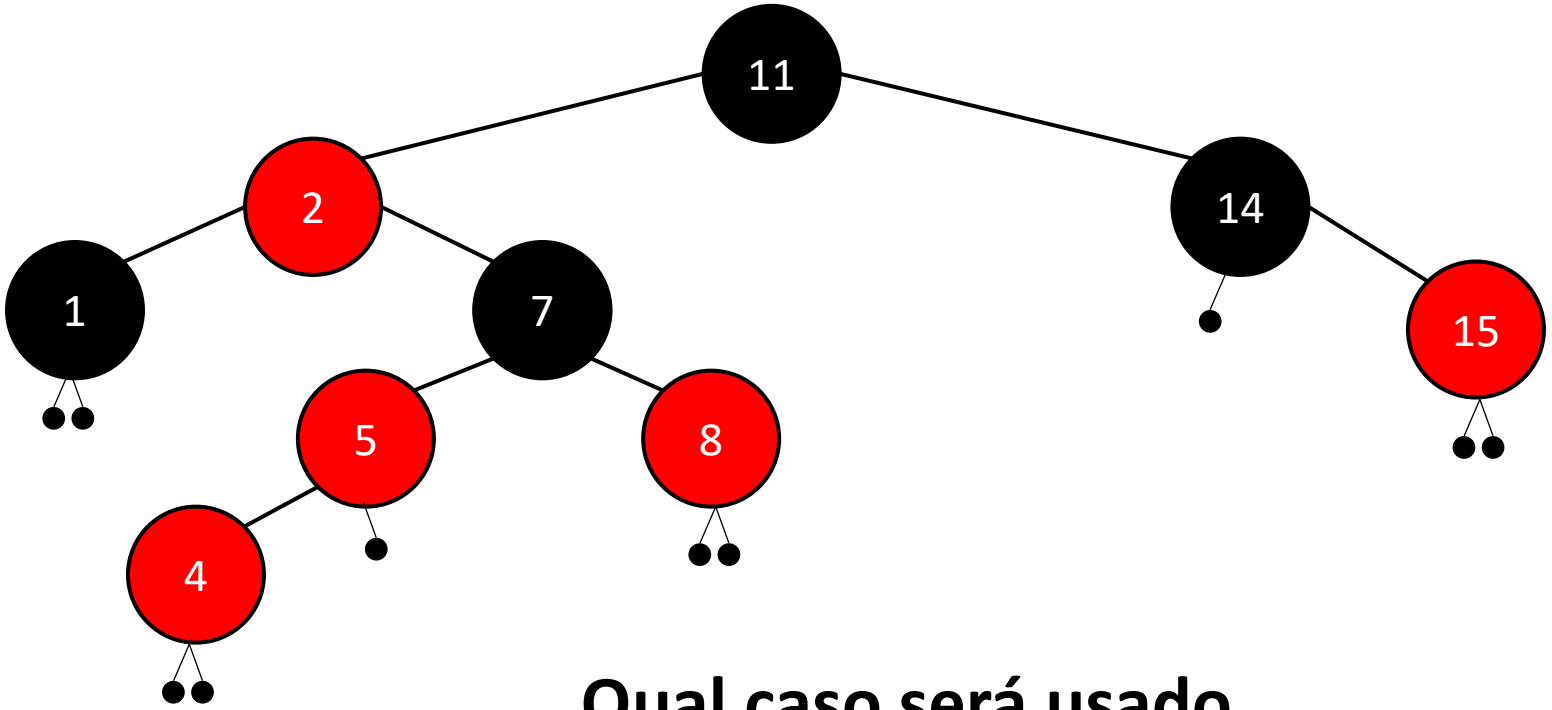
Exemplo



Qual regra foi violada?

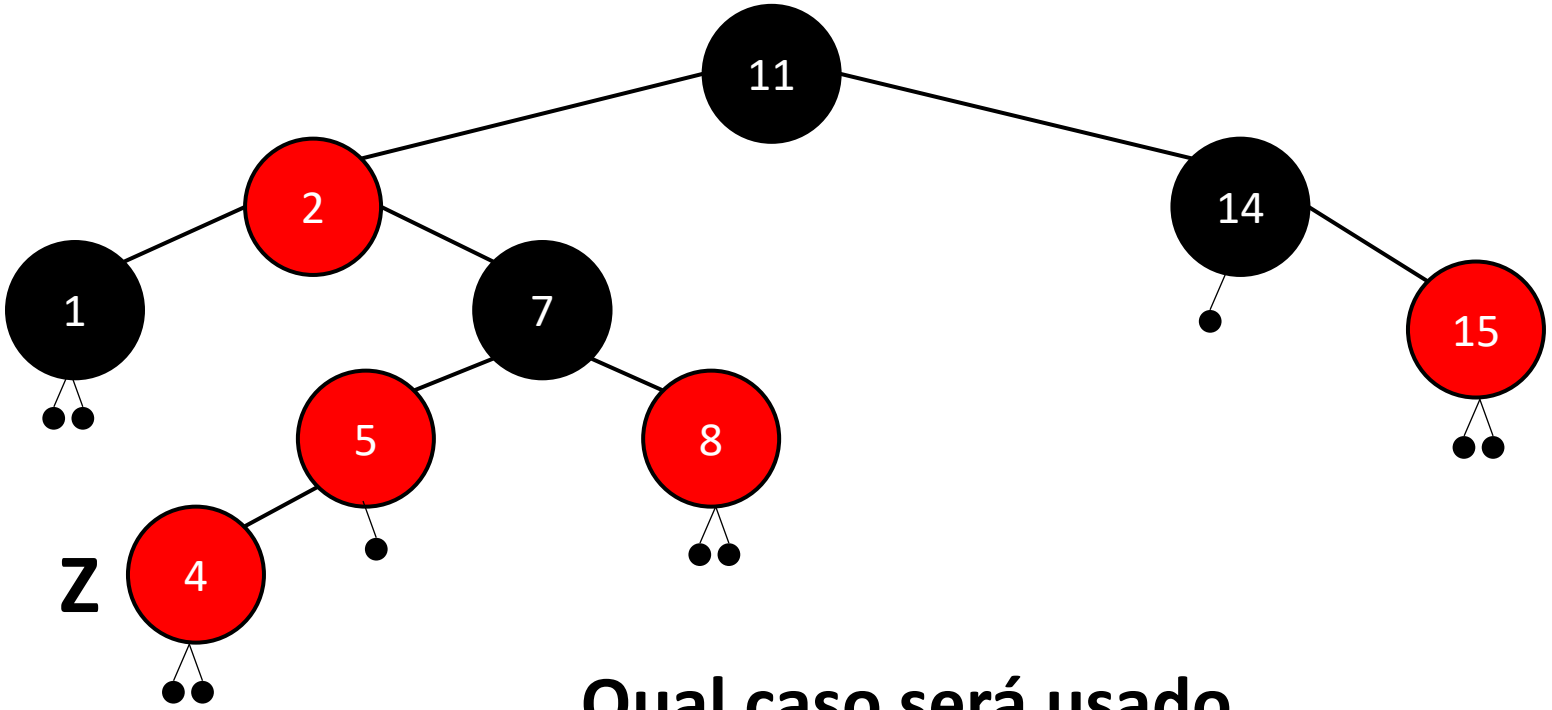
Regra 4.

Exemplo



Qual caso será usado para corrigir?

Exemplo

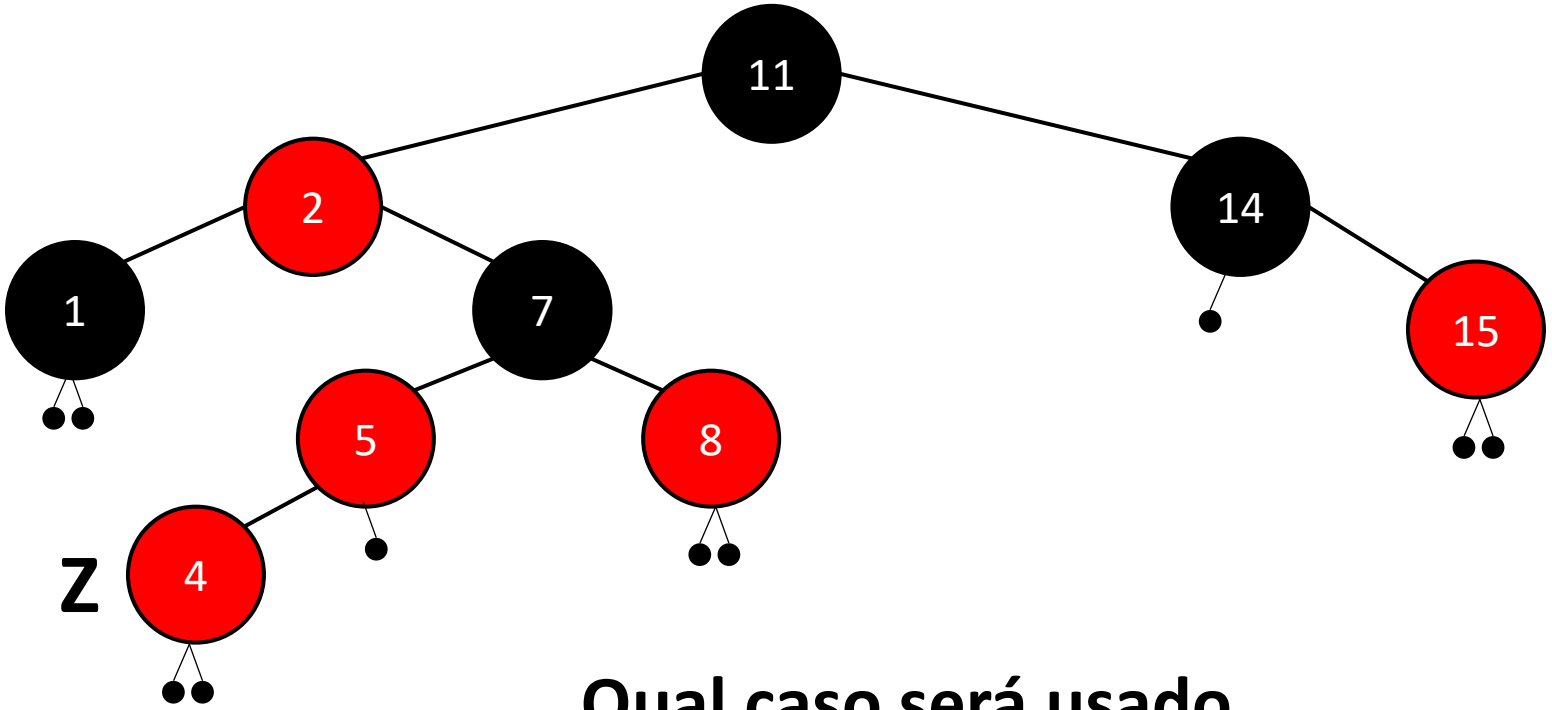


Qual caso será usado para corrigir?

Exemplo

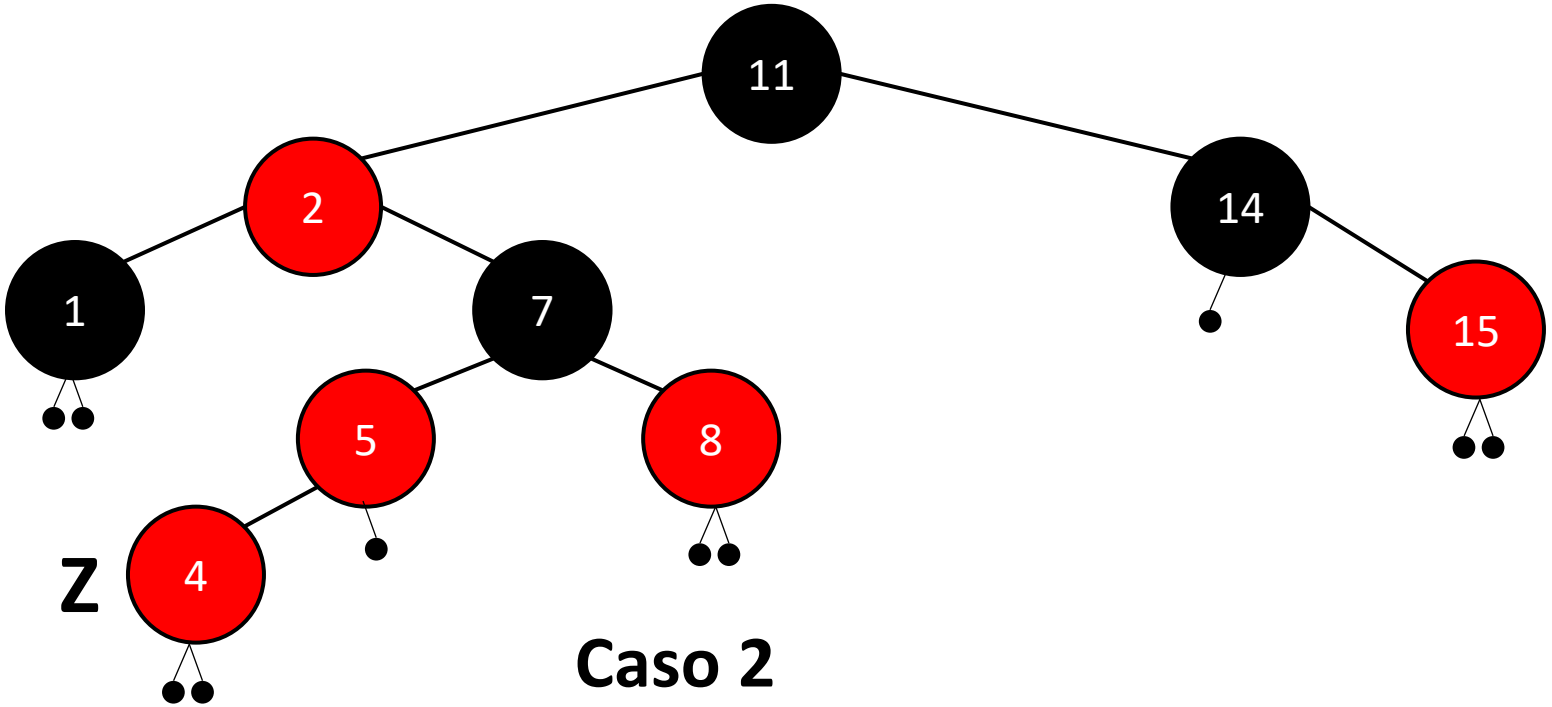
- Z é a raiz
- O tio de Z é vermelho
- O tio de Z é preto (forma triangulo)
- O tio de Z é preto (forma linha)

Exemplo



Qual caso será usado para corrigir?

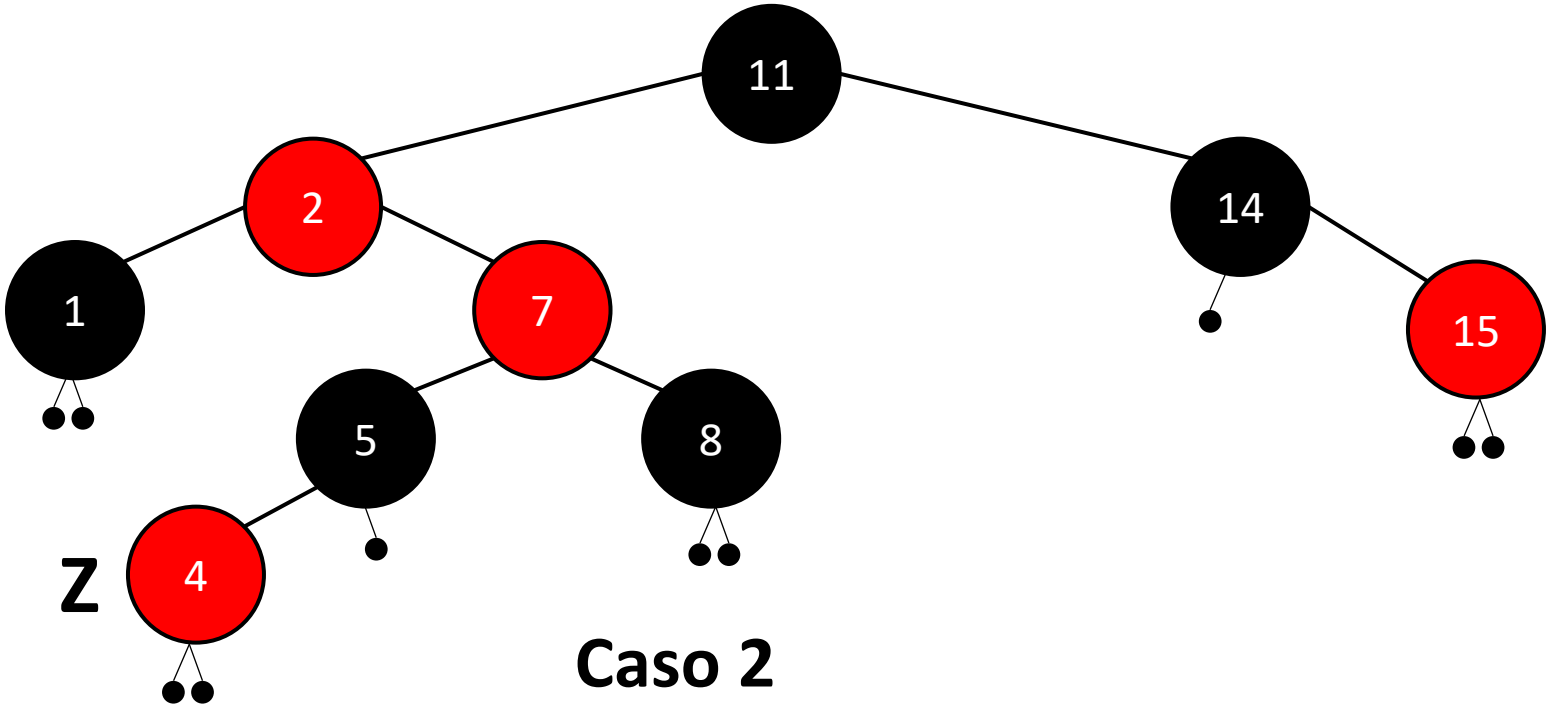
Exemplo



Caso 2

Solução: trocar a cor do pai, do tio e do avô

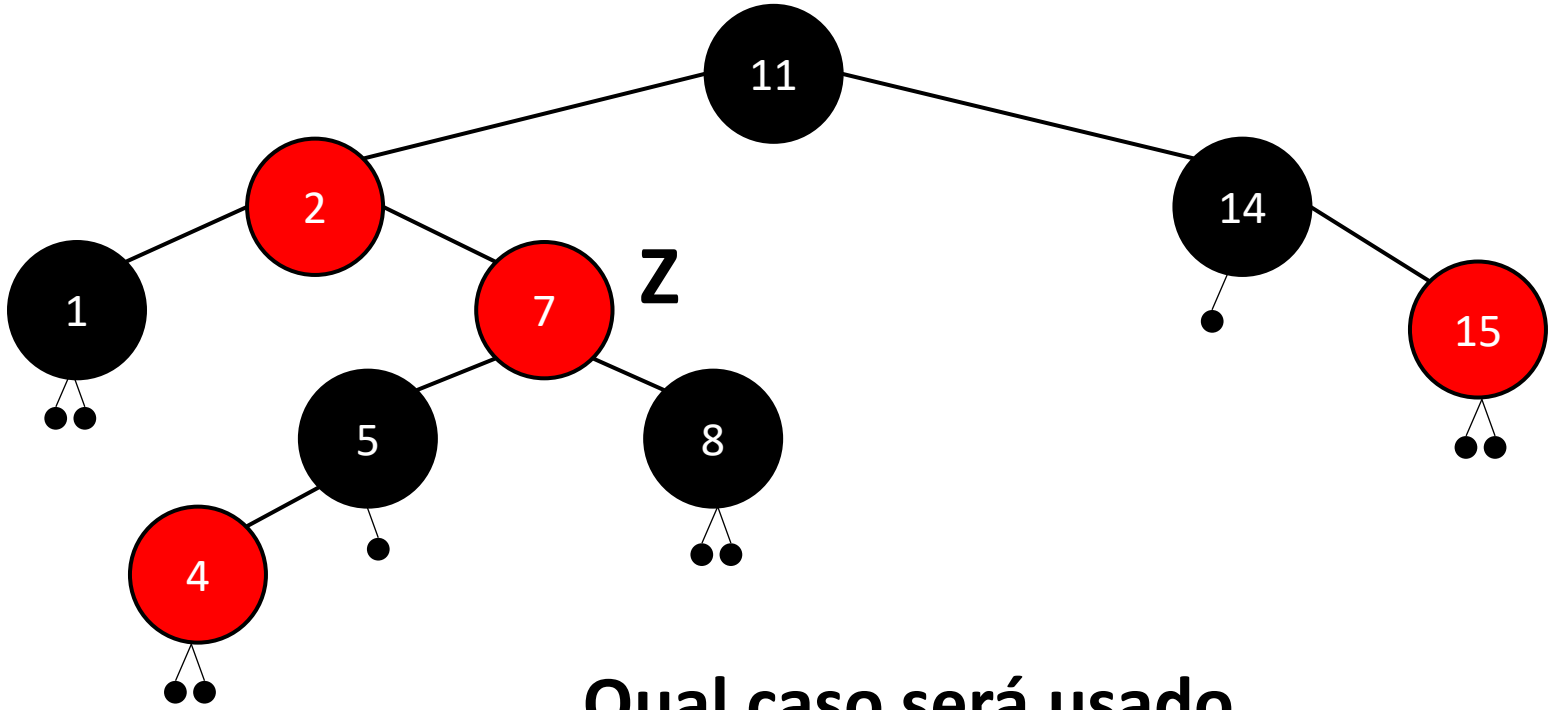
Exemplo



Caso 2

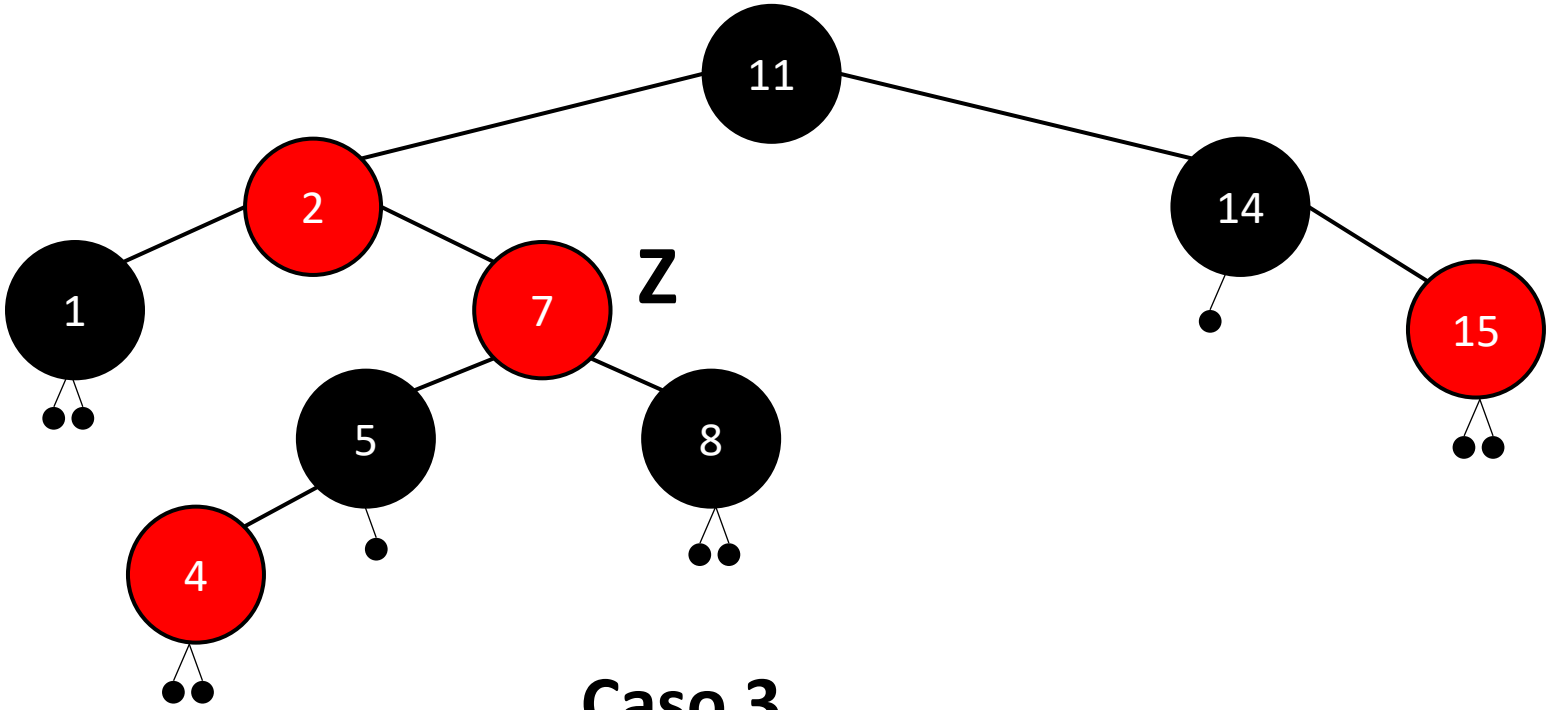
Solução: trocar a cor do pai, do tio e do avô

Exemplo



Qual caso será usado para corrigir?

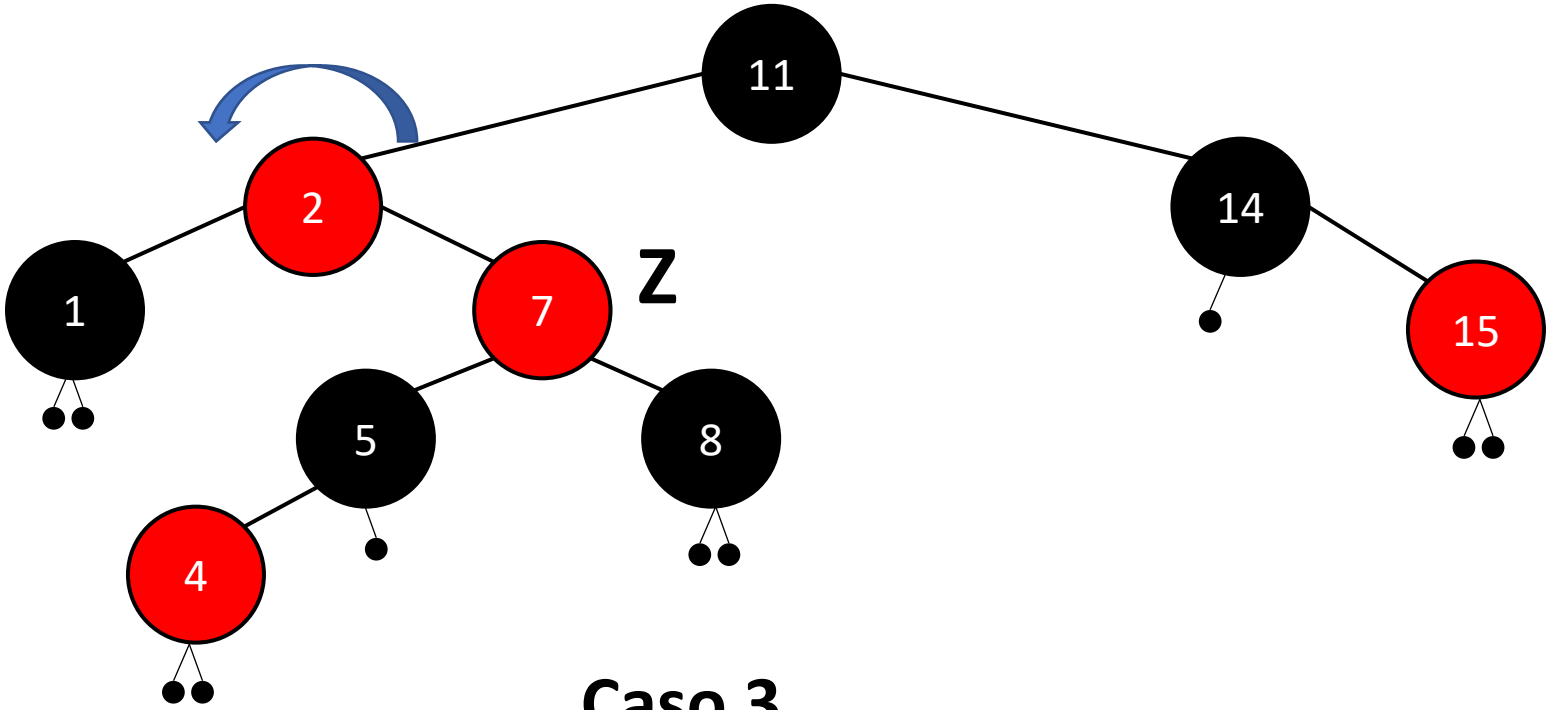
Exemplo



Caso 3

Solução: rotação à esquerda no pai de Z.

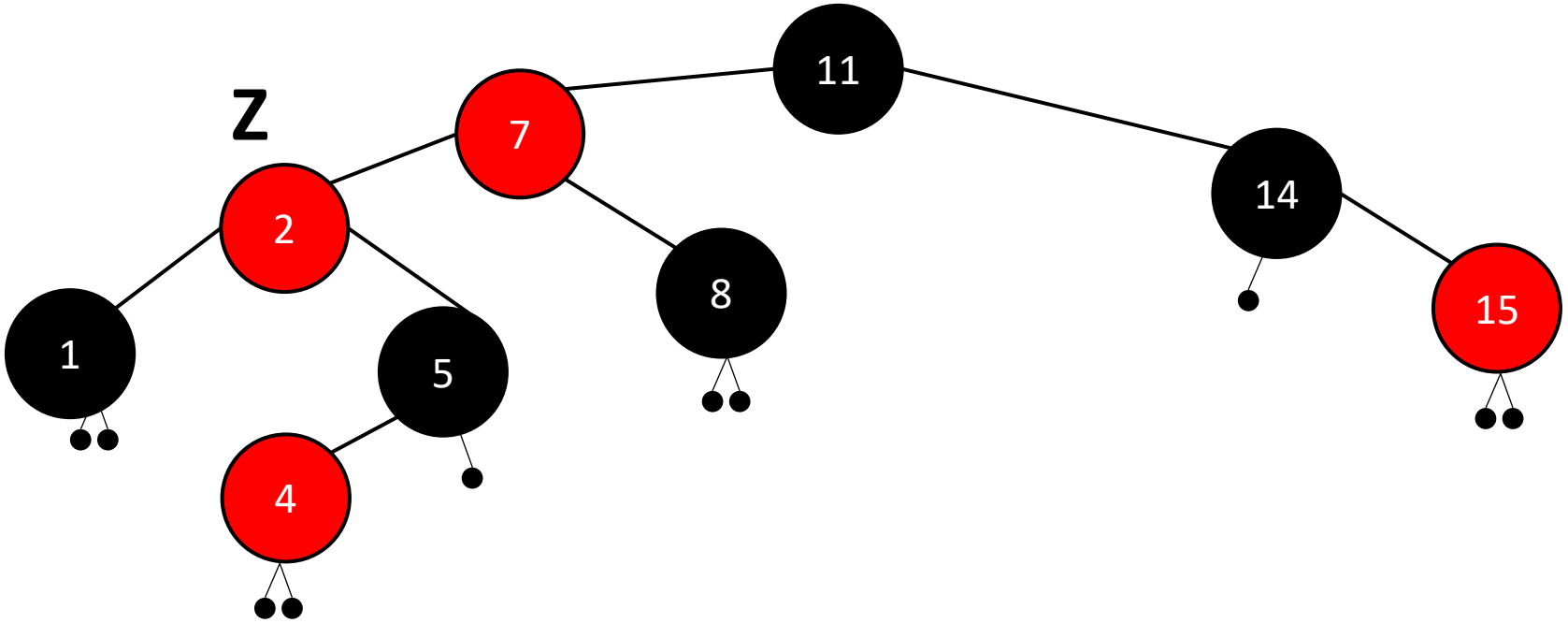
Exemplo



Caso 3

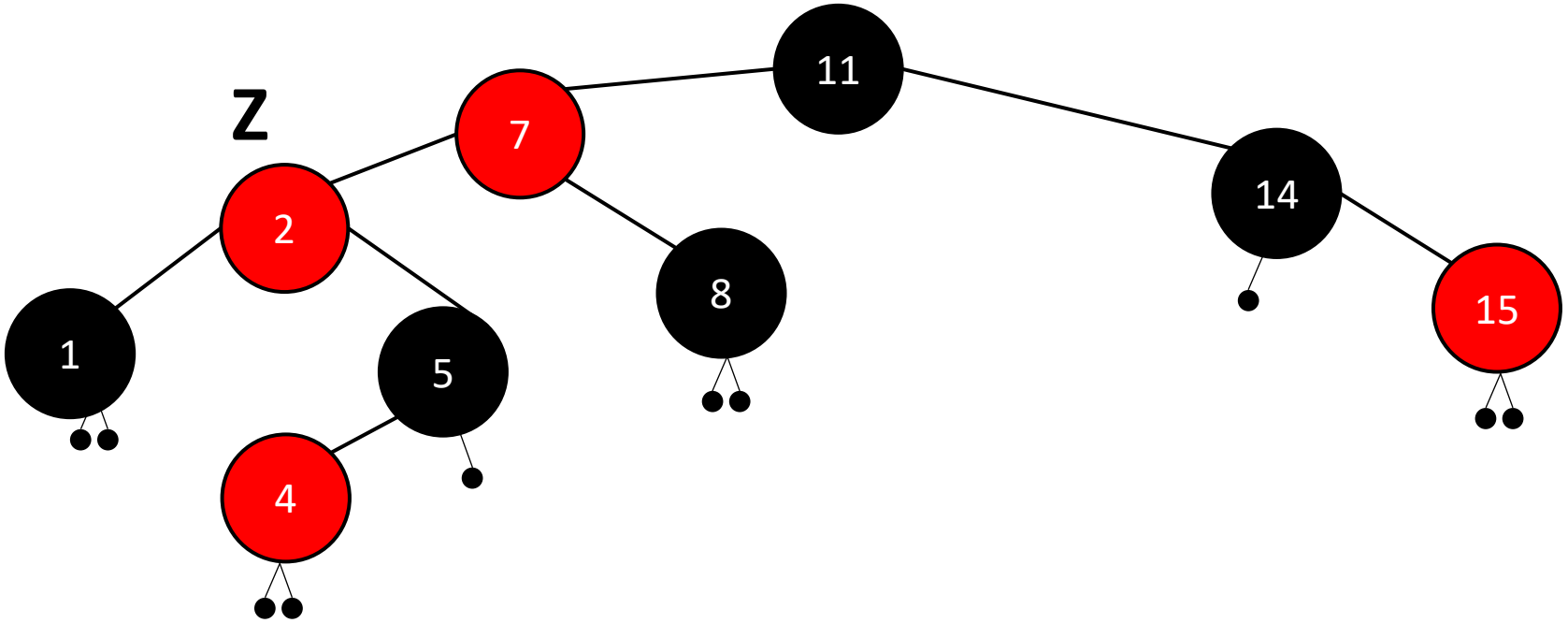
Solução: rotação à esquerda no pai de Z.

Exemplo



Qual caso será usado para corrigir?

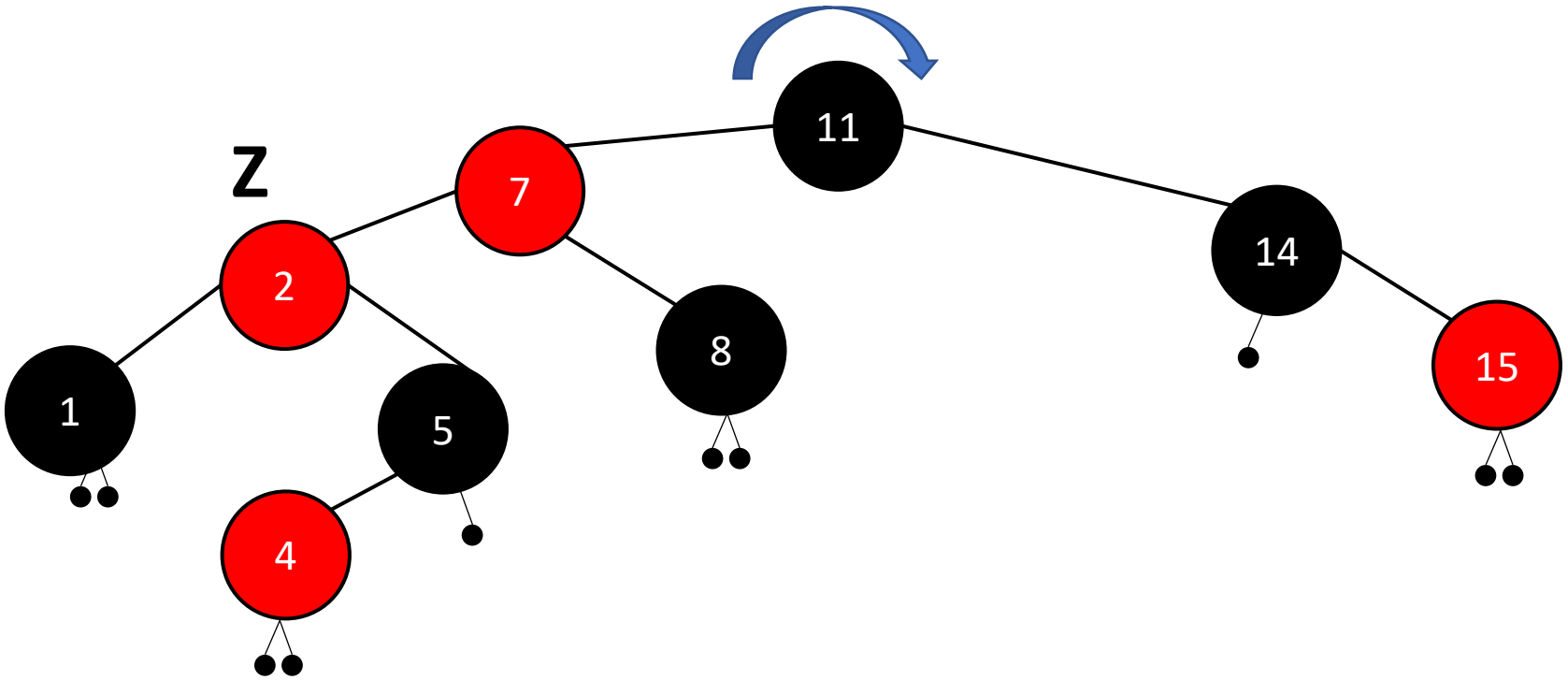
Exemplo



Caso 4

Solução: rotação à direita do avô de Z. Recolorir o pai e o avô original.

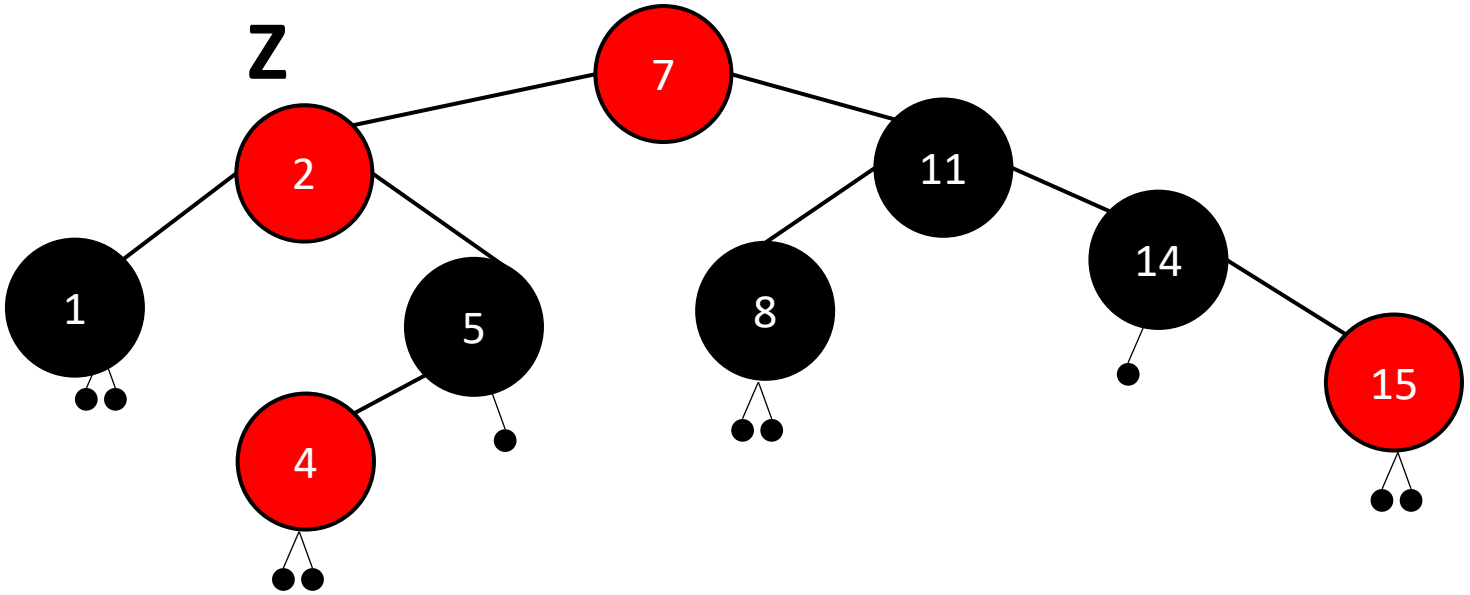
Exemplo



Caso 4

Solução: rotação à direita do avô de Z. Recolorir o pai e o avô original.

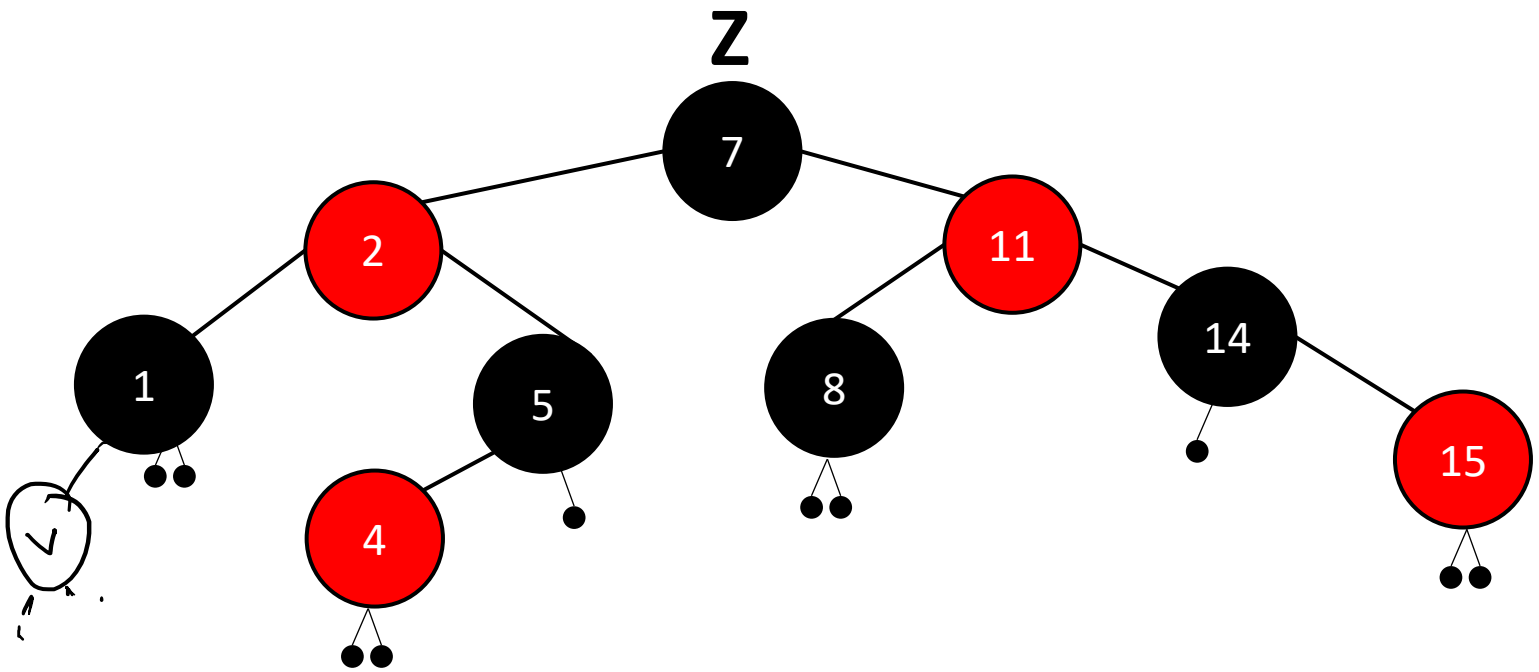
Exemplo



Caso 4

Solução: rotação à direita do avô de Z. Recolorir o pai e o avô original.

Exemplo



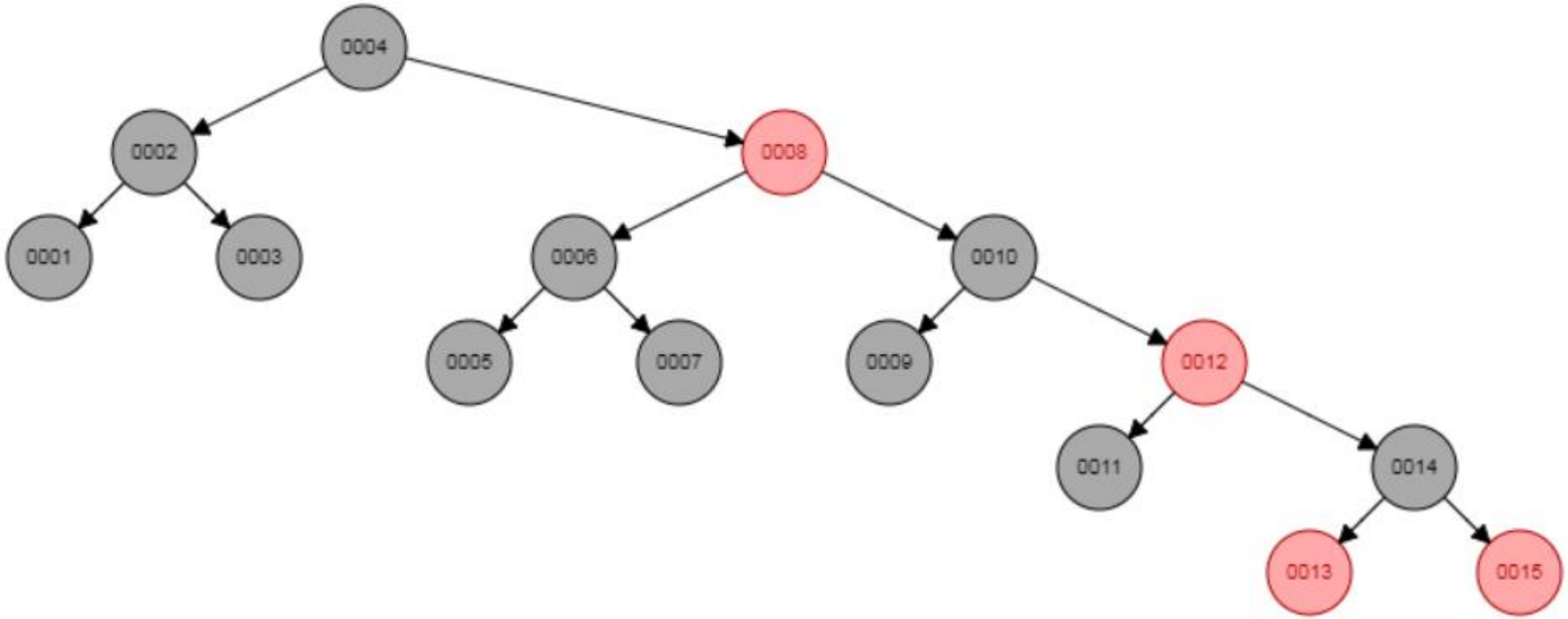
Caso 4

Solução: rotação à direita do avô de Z. Recolorir o pai e o avô original.

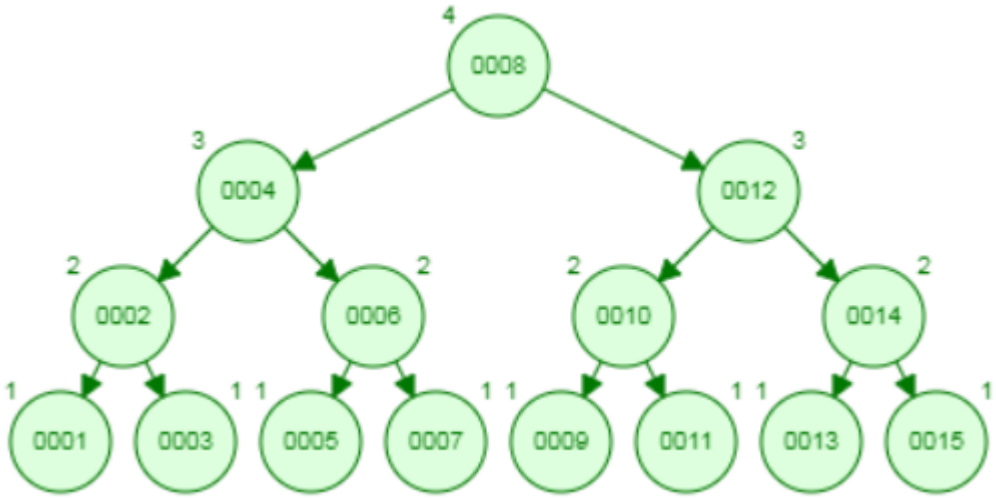
Atividade

- Inserir os nós de 1 até 15 numa árvore rubro-negra.

Atividade


0004
0002
0008
0001
0003
0006
0010
0005
0007
0009
0012
0011
0014
0013
0015

Exemplo


4
0008
3
0004
3
0012
2
0002
2
0006
2
0010
2
0014
1
0001
1
0003
1
0005
1
0007
1
0009
1
0011
1
0013
1
0015

Referências

- CORMEN, Thomas H et al. **Algoritmos: teoria e prática.** Rio de Janeiro: Elsevier, 2012. 926 p. ISBN: 9788535236996.
- ASCENCIO, Ana Fernanda Gomes. **Estruturas de dados: algoritmos, análise da complexidade e implementações em Java e C/C++.** São Paulo: Pearson, c2010. 432 p. ISBN: 9788576052216, 978857605816.
- PIVA JÚNIOR, Dilermando (et al). **Estrutura de dados e técnicas de programação**. 1. ed. Rio de Janeiro, RJ: Campus, 2014. 399 p. ISBN: 9788535274370.
- SKIENA, S. S. **The Algorithm Design Manual**. Second Edition. Springer London. 2008.
- Reinaldo Faria. Notas de aula. Disponível em: http://www.decom.ufop.br/reinaldo/site_media/uploads/2013-02-bcc202/aula_04_-_analise_de_algoritmos_(parte_1)_(v2).pdf

Referências

- FERRARI, Roberto et al. **Estruturas de dados com jogos**. 1. ed. Rio de Janeiro: Elsevier, 2014. 259p. ISBN: 9788535278040.
- GRONER, Loiane. **Estruturas de dados e algoritmos em Javascript**: aperfeiçoe suas habilidades conhecendo estruturas de dados e algoritmos clássicos em JavaScript. São Paulo: Novatec, 2017. 302 p. ISBN: 9788575225530.
- SZWARCFITER, Jayme Luiz; MARKENZON, Lilian. **Estruturas de dados e seus algoritmos**. 3. ed. Rio de Janeiro: LTC, 2010. xv, 302 p. ISBN: 9788521617501.
- GOODRICH, Michael T; TAMASSIA, Roberto. **Estruturas de dados e algoritmos em Java**. 5. ed. Porto Alegre: Bookman, 2013. xxii, 713 p. ISBN: 9788582600184.
- GUIMARÃES, Ângelo M. **Algoritmos e estruturas de dados**. LTC, 1994.