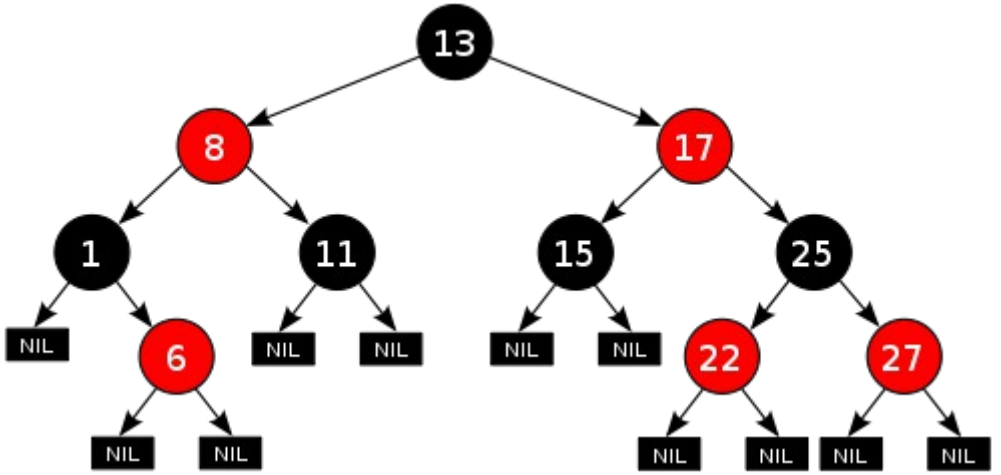
Árvores Binárias



Linguagem de Programação 2
(Estrutura de Dados)

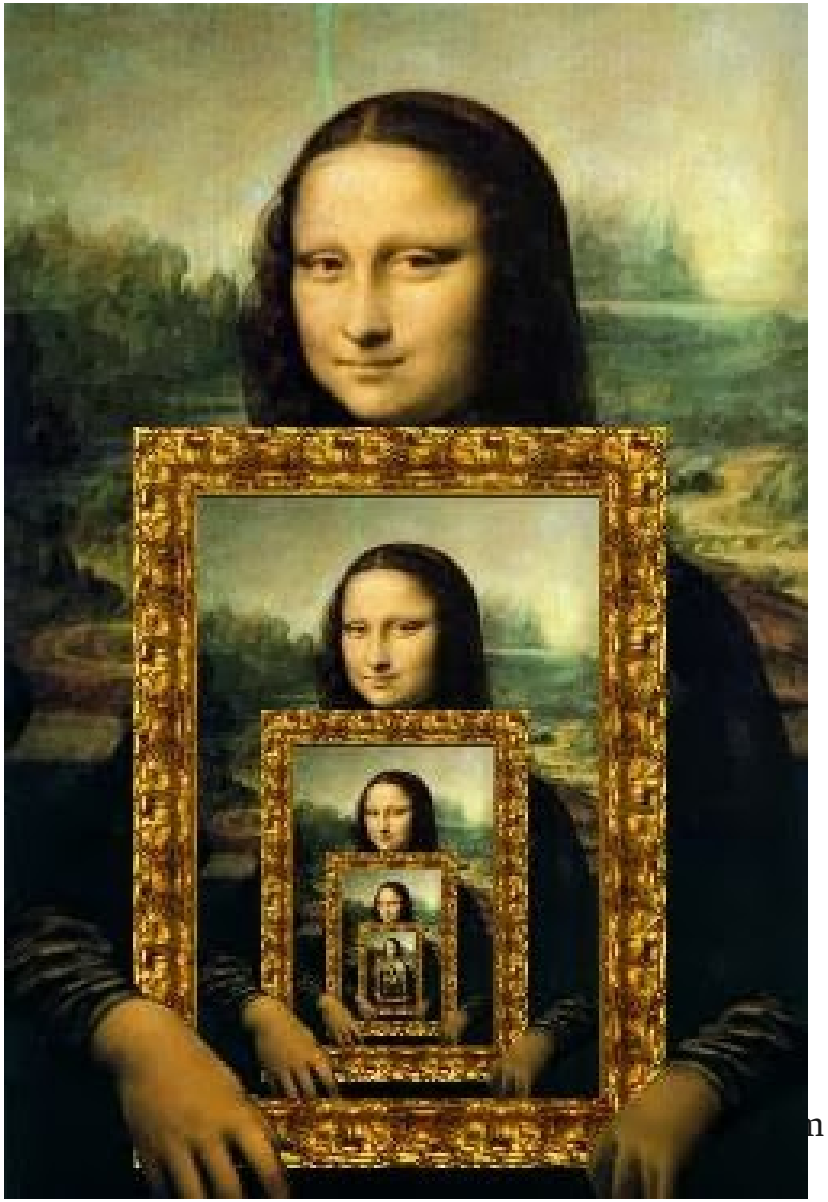
Agenda

- Relembrando Recursividade
- Definições sobre árvore binária
- Entendendo Árvores Binárias de Busca
- Problema da Árvore Binária de Busca Simples!
- Solução: Balancear as Árvores!
- Árvores AVL
- Exercícios de Fixação

Recursividade

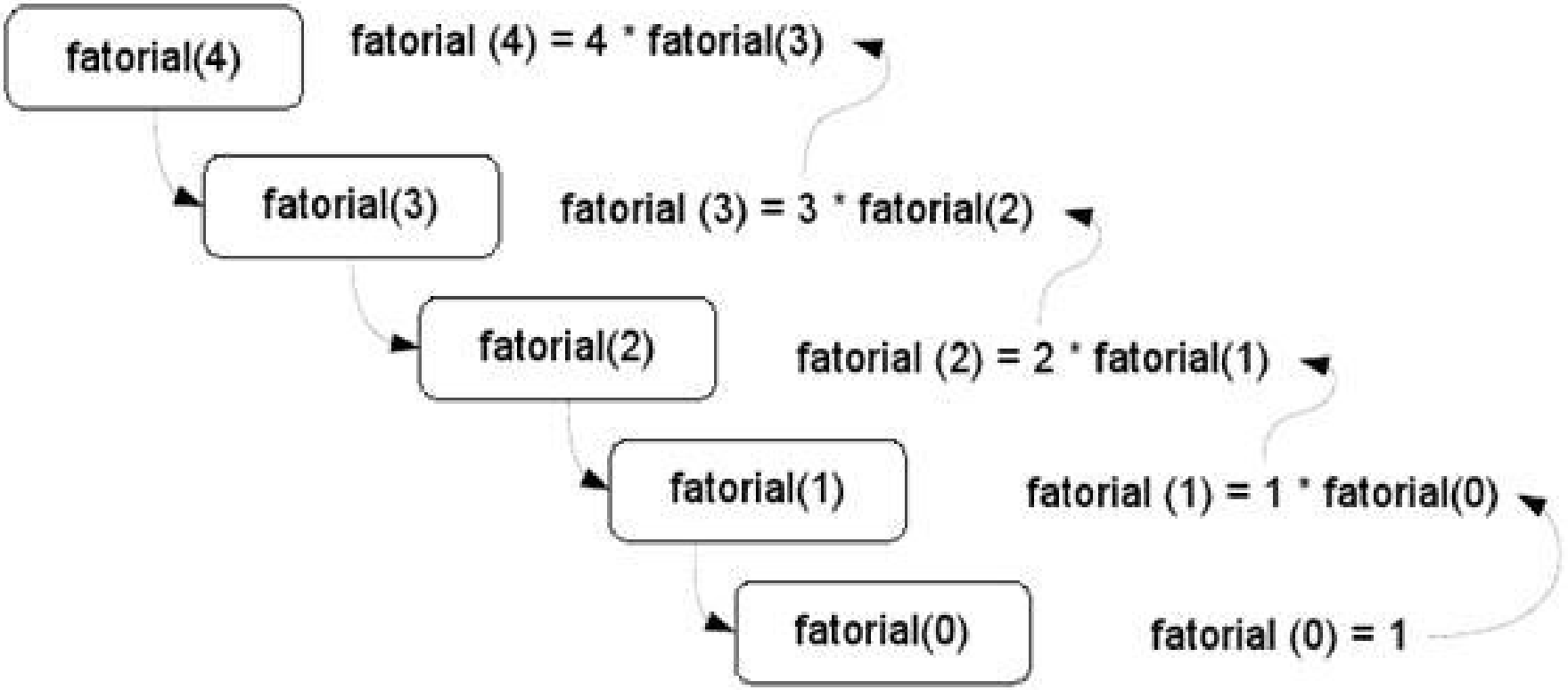
- “Recursão é uma técnica de programação na qual um método chama a si mesmo” pg.221
  - Duas regras devem ser seguidas para uma função recursiva:
    - Ter uma condição de parada;
    - Tornar o problema mais simples;
  - A recursão esta associada a estratégia dividir para conquistar, onde busca-se dividir um problema em partes menores, menores, menores até chegar em algo pequeno o suficiente e depois as soluções vão se completando até ter a solução do problema maior!

Recursividade



Recursividade

- Exemplo 1: O cálculo Fatorial!!!



Recursividade

- Exemplo 1: O cálculo Fatorial!!!
  - Como implementar?

```
//Essa é a função do log ... ela recebe um número n como parametro
public static int log(int n) {
    if (n == 0) // se n for igual a zero, o fatorial é igual a 1
        return 1;

    return n*(log(n-1)); // senão o fatorial é igual ao numero * o fatorial dele menos 1
}                        // essa função irá chamar a si mesma até chegar no
                         // problema mais simples: fatorial de 0 ... e depois ela
                         // vai se "desdobrando
```

Recursividade

- Exemplo 2: Número Triangulares!!
  - Qual o próximo? {1, 3, 6, 10, 15, 21, ?}

Como saber por exemplo, qual o valor do número da posicao 11?

Recursividade

- Exemplo 2: Números Triangulares!!
  - Qual o próximo? {1, 3, 6, 10, 15, 21, ?}

    Um número n é igual a soma de sua **posição** com o valor do número que o antecede.

28

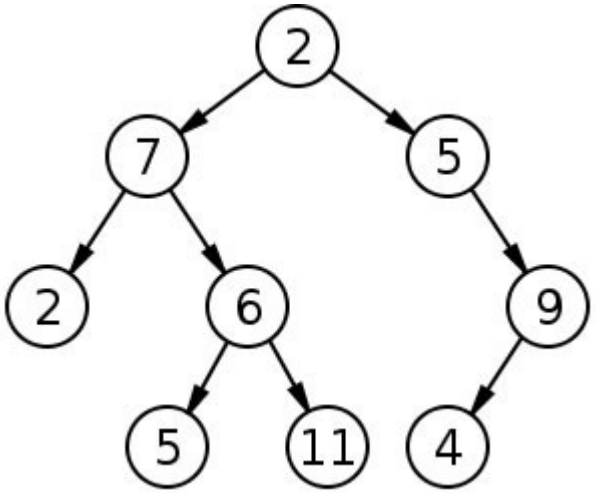
```
public static int tri(int n) {
    if (n == 1)
        return 1;

    int resp = n+(tri(n-1));
    System.out.print(resp + ", ");

    return resp;
}
```

Recursividade

- Para fechar:
  - Recursividade é uma das técnicas mais famosas na programação!
  - Quase sempre quando é necessário quebrar um problema maior em partes menores ela é usada.
    - Por exemplo, pode ser usada em uma **busca binária!!!**



Porque estamos vendo recursividade?

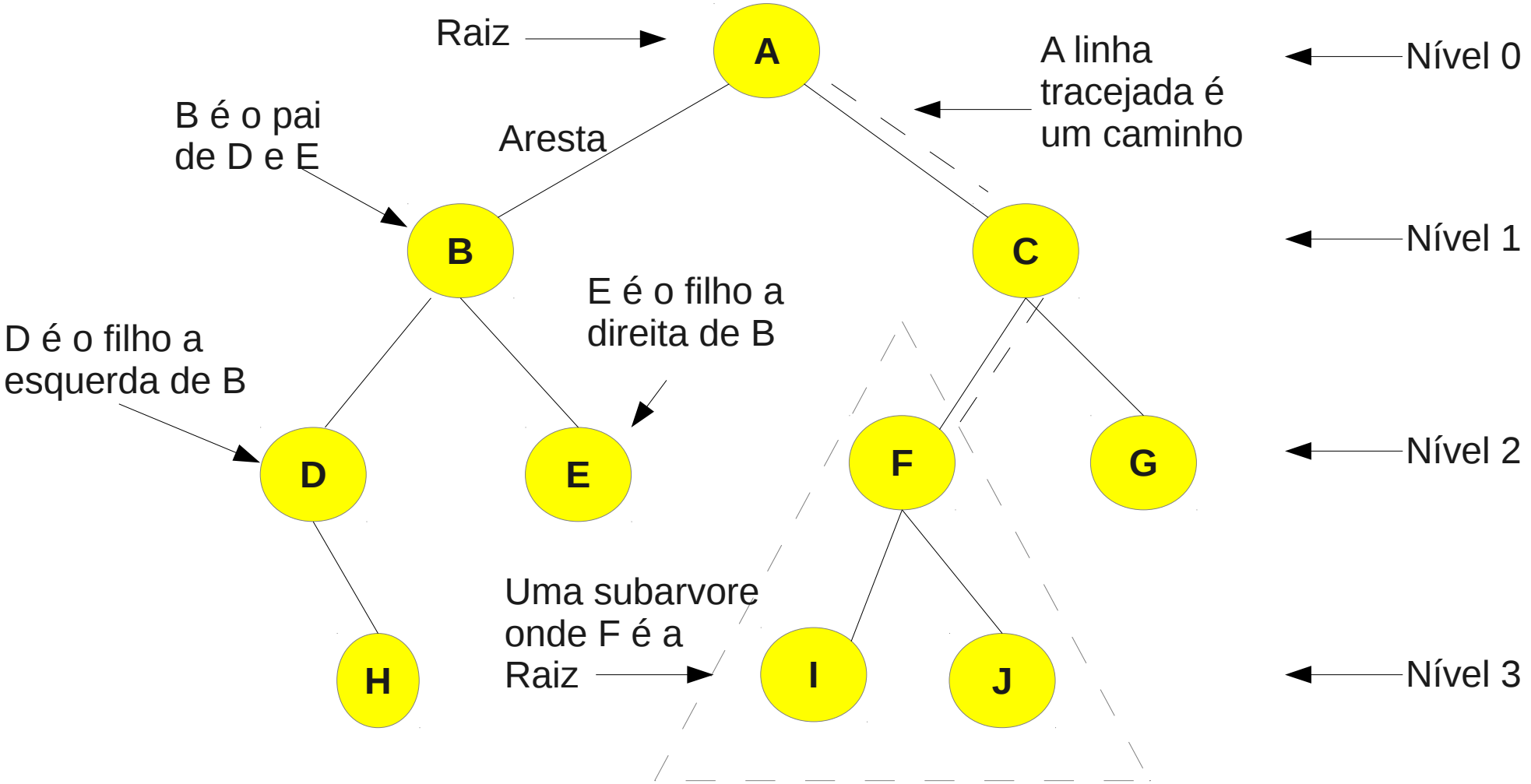
Árvores Binárias

Árvores Binárias

- “Árvores binárias são **estruturas de armazenamento de dados** fundamentais em programação” pg. 327
- Porque usar árvores binárias?
  - Árvores binárias combinam o melhor que existe entre vetor e entre lista encadeada!
    - Buscas rápidas como no vetor ordenado!
    - Inserção e Remoção rápidas como nas listas encadeadas!!!

Árvores Binárias

- Conhecendo uma árvore:

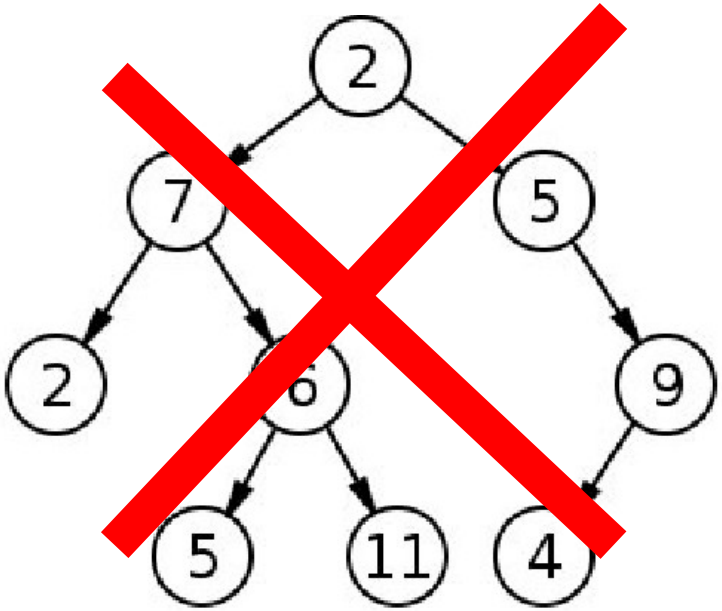


Árvores Binárias

- Terminologia sobre Árvores:
  - **Pai**: o nó acima do nó X é o nó pai de X;
  - **Filho**:os nós imediatamente abaixo de um nó X são chamados filhos (a direita ou esquerda) de X.
  - **Folha**: Um nó que não tem filhos.
  - **Subárvore**: qualquer nó pode ser raiz de uma subárvore, tendo ou não descendentes!
  - **Visitar um nó**: visitar um nó corresponde a ter sobre ele o controle do sistema e visualizar ou manipular seus dados.
  - **Percorrer uma árvore:** visitar todos os seus nós em uma ordem específica
  - **Níveis**: o nível de um nó corresponde a quantas gerações ele esta da raiz.
  - **Chaves**: é o valor utilizado como referência para aquele nó! Lembre de BD!
  - **Árvore Binária**: é a arvore onde todo nó pode ter no máximo 2 filhos: o filho a direita e o filho a esquerda.

Árvores Binárias

- Trabalharemos aqui com a chamada Árvore Binária de Busca que seque a seguinte regra:
  - O Filho da **esquerda** de um nó tem que ter uma chave **menor** que seu pai e o filho à **direita** de um nó tem que ter uma chave **maior** ou igual ao seu pai.



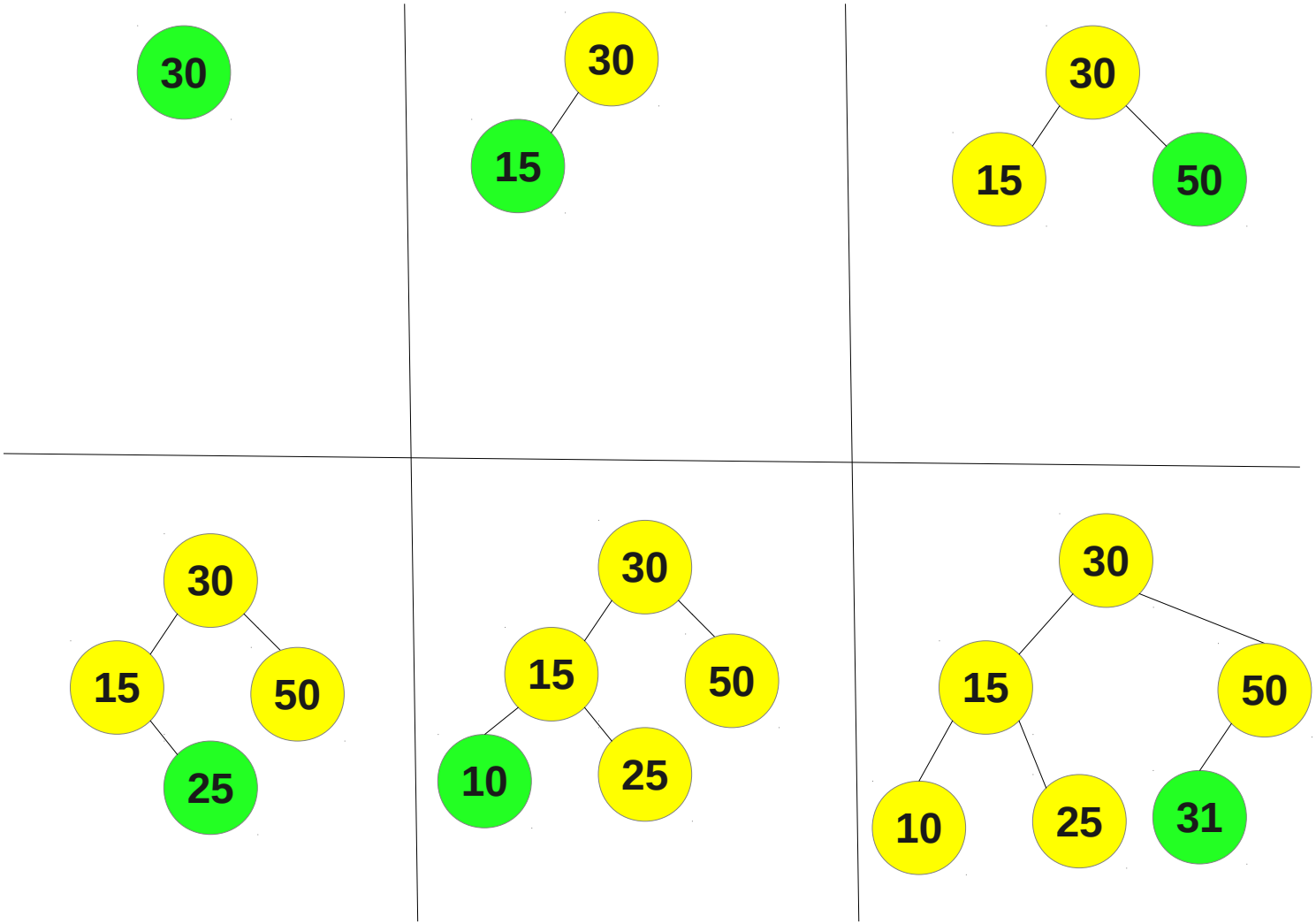
Árvores Binárias - INSERÇÃO

- Regras para inserir em uma árvore binária de busca:
  1. Se ainda não há nó raiz, então o novo elemento será o próprio nó raiz
  2. Se há nó raiz, então compare o novo elemento com o nó raiz:
     1. Caso o novo elemento seja menor que o elemento do nó raiz, então o novo elemento é inserido na sub-árvore da esquerda;
     2. Caso o novo elemento seja maior que o elemento do nó raiz então o novo elemento é inserido na subárvore da direita.

Essa regra é aplicada recursiva!!!

Árvores Binárias - INSERÇÃO

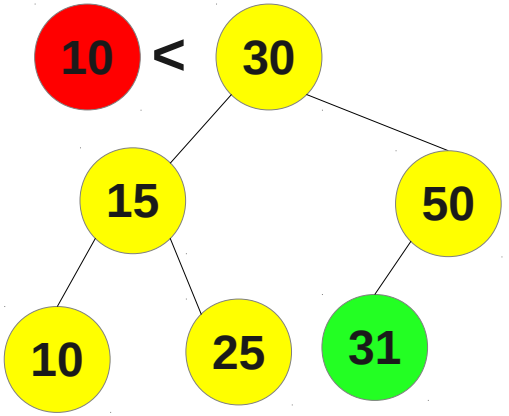
- Inserir(30)
- Inserir(15)
- Inserir(50)
- Inserir(25)
- Inserir(10)
- Inserir(31)

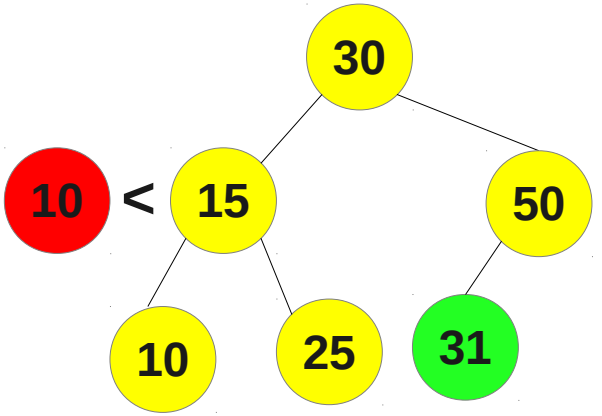


E no JAVA???

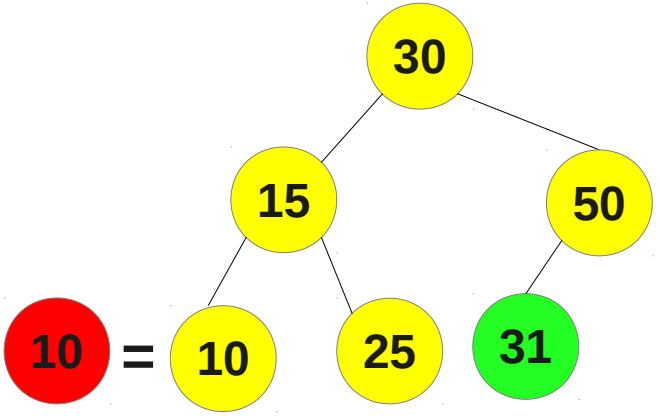
Árvores Binárias - BUSCA

- Buscar(10)





Melhor Caso: 1 Comparação
Pior Caso: Número Níveis!!!



Árvores Binárias - REMOÇÃO

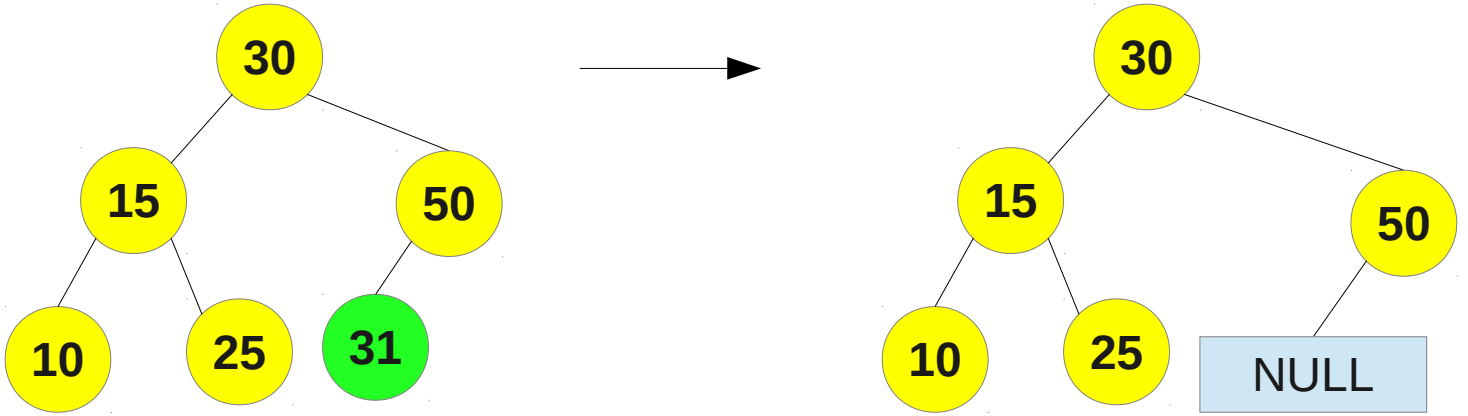
- Regras para remoção:

  Passo 1: Encontrar o nó que se deseja remover …

  – Passo 2: Existem 4 possíveis casos:
    - Caso 1: O nó a ser apagado é uma folha
    - Caso 2: O nó a ser apagado tem um filho
    - Caso 3: O nó a ser apagado tem dois filhos s/ netos
    - Caso 4: O nó a ser apagado tem dois filhos e tem netos!

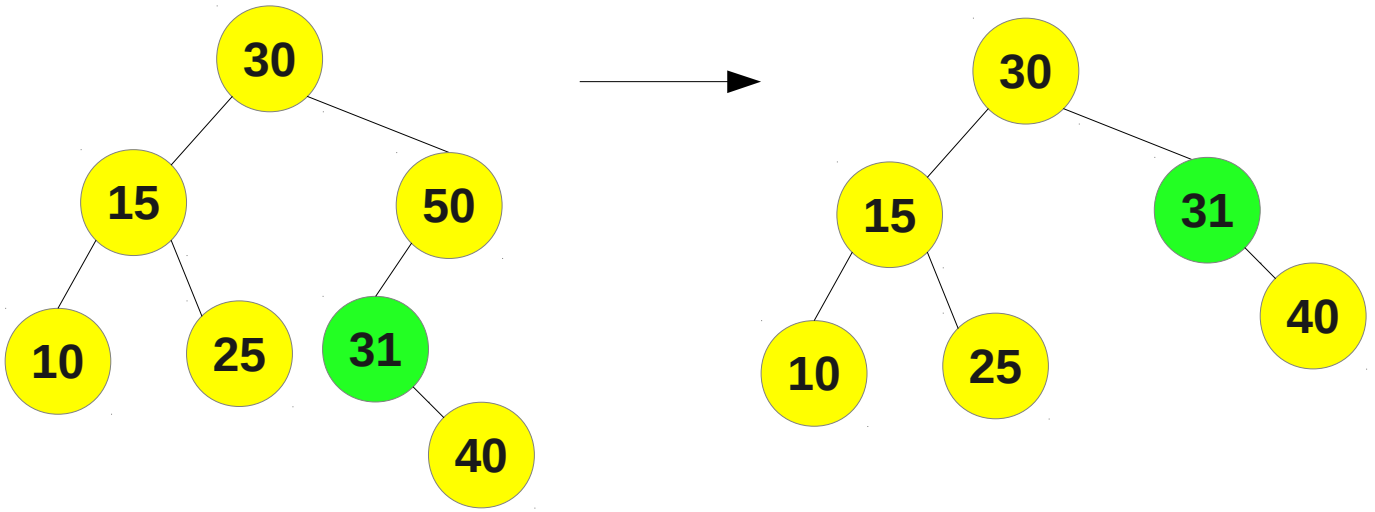
Árvores Binárias - REMOÇÃO

- **Caso 1**: O nó a ser apagado é uma folha
  - Neste caso basta mudar o nó pai do filho que se quer apagar para apontar para NULL!!!
    - Exemplo: Remover o 31



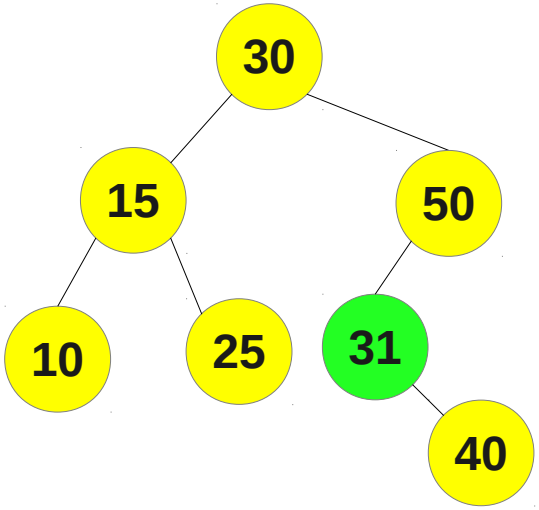
Árvores Binárias - REMOÇÃO

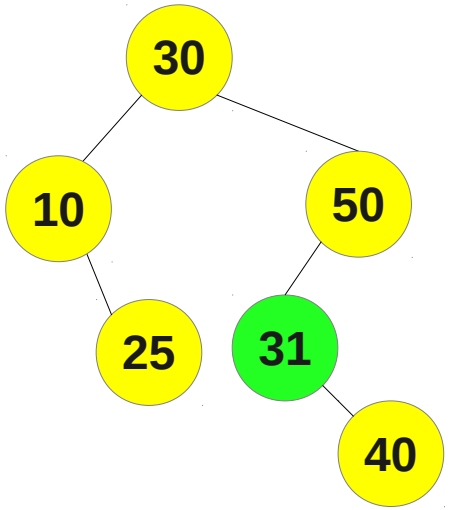
- **Caso 2**: O nó a ser apagado tem somente um filho
  - O filho único será o herdeiro!!!
    - Exemplo: remover o número 50



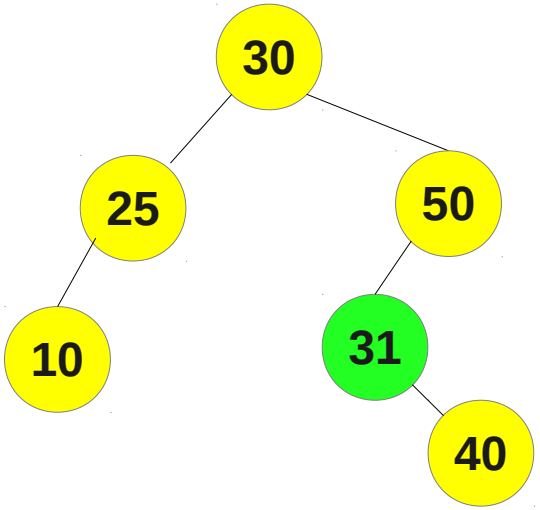
Árvores Binárias - REMOÇÃO

- **Caso 3**: O nó a ser eliminado possui dois filhos s/ netos:
  - Tanto um filho quanto o outro pode assumir o lugar!!!
  - Exemplo: Remover o 15





Possibilidade 1



Possibilidade 2

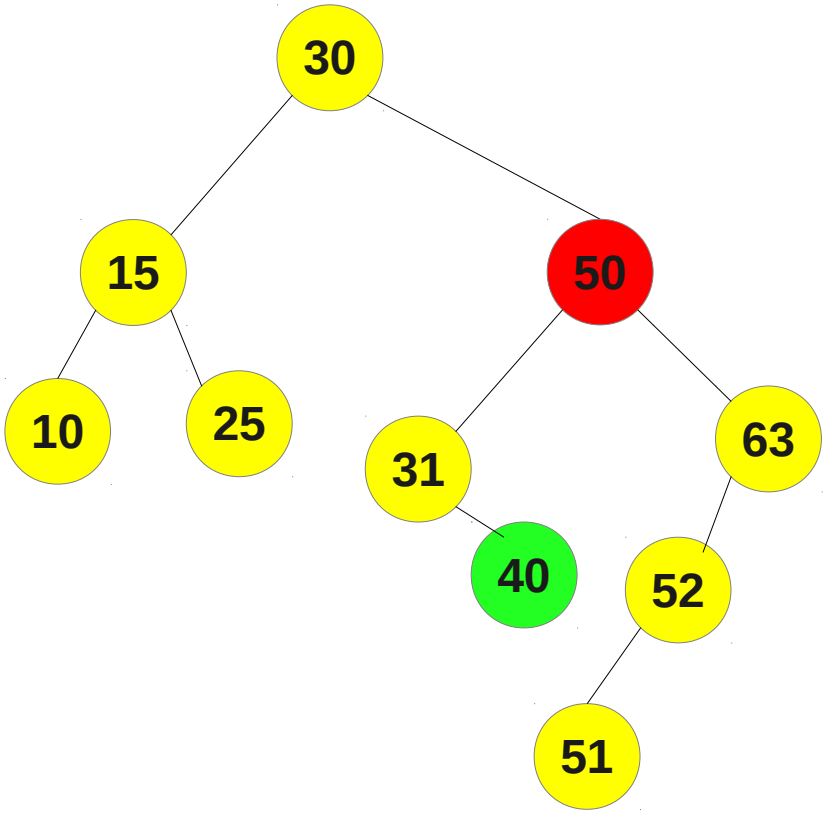
Árvores Binárias - REMOÇÃO

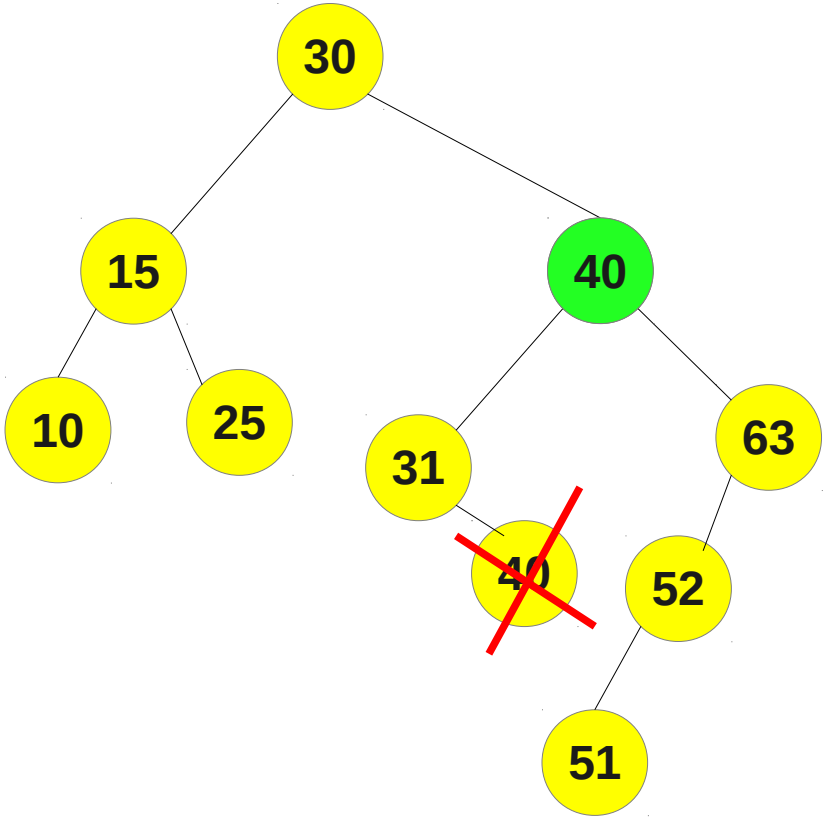
- **Caso 4:** O nó a ser eliminado possui dois filhos e netos:
  - **Dica 1**: Precisamos pensar na recursividade! Todo nó é nó raiz de alguma subárvore!
  - **Solução 1**: Escolhemos para substituir ele o maior elemento da esquerda!
  - **Solução 2**: Escolhemos para substituir ele o menor elemento da direita!
  - **Solução do Lafore:** Substitua pelo sucessor direto!!!

OBS: Ao ir substituir o nó removido, o nó substituto deverá também ser removido da sua subárvore, logo, essa subárvore deverá passar novamente de forma recursiva pela regra!!!

Árvores Binárias - REMOÇÃO

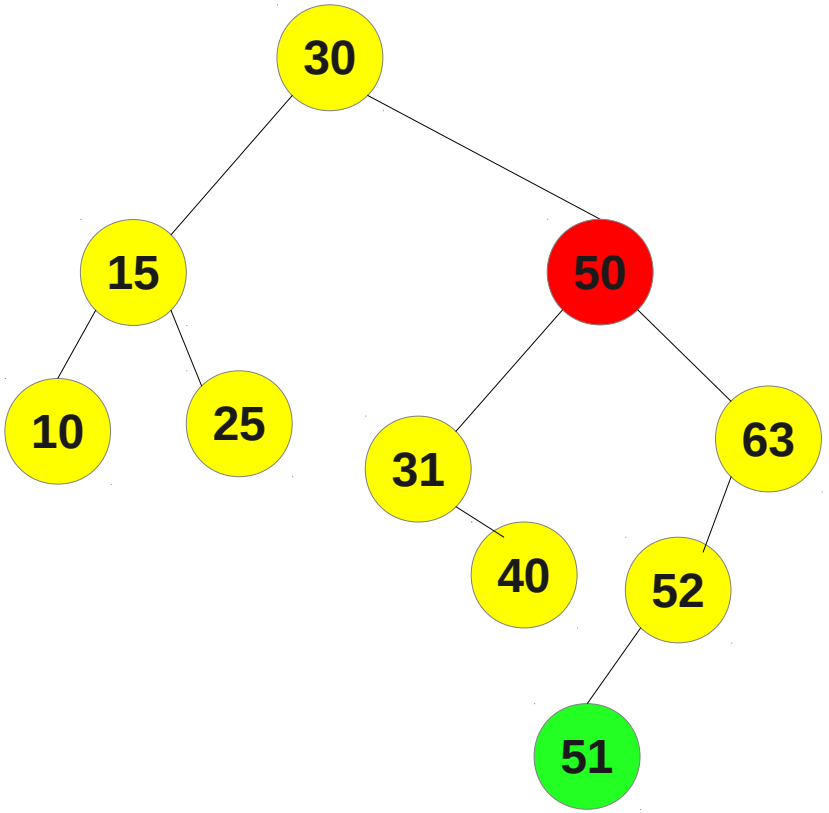
- **Caso 4**: Exemplo: Remover o número 50
  - **Opção A**: Usar o maior da esquerda

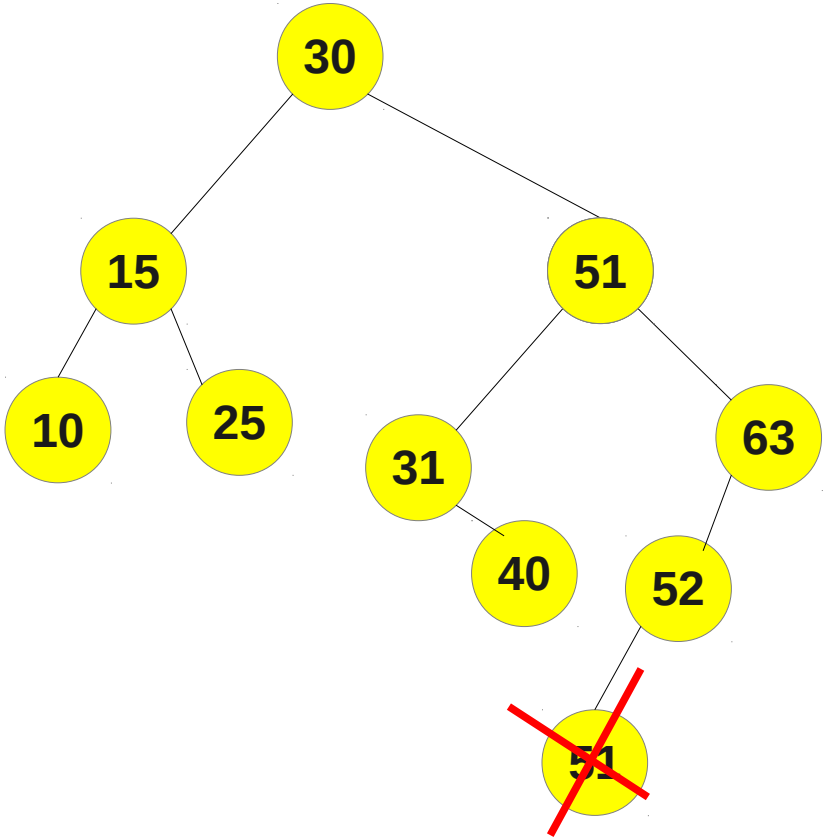




Árvores Binárias - REMOÇÃO

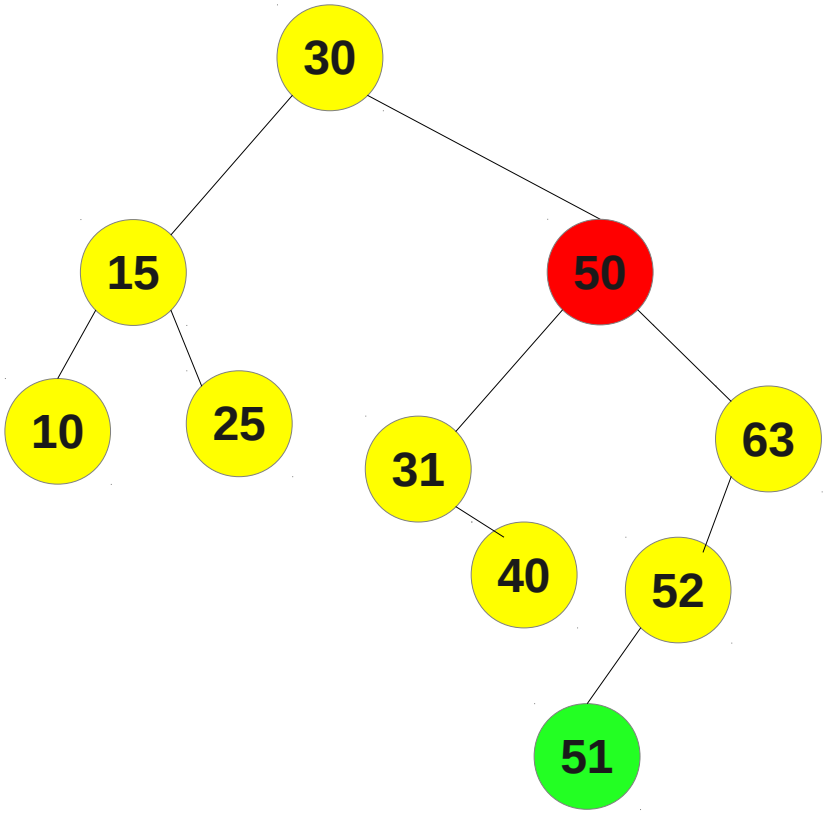
- **Caso 4**: Exemplo: Remover o número 50
  - **Opção B**: Usar o menor da direita

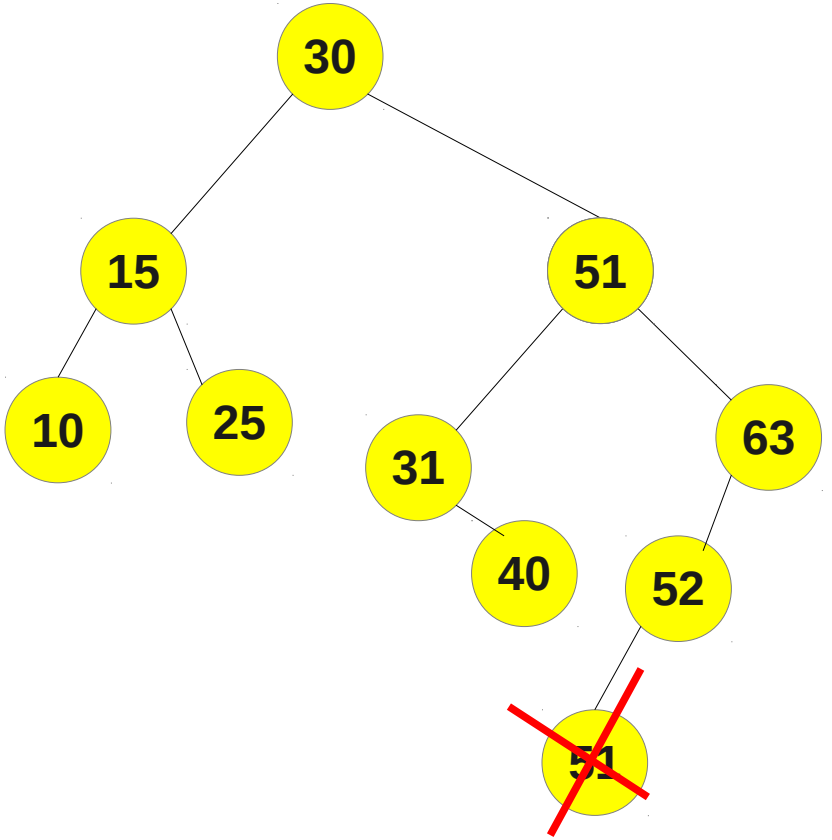




Árvores Binárias - REMOÇÃO

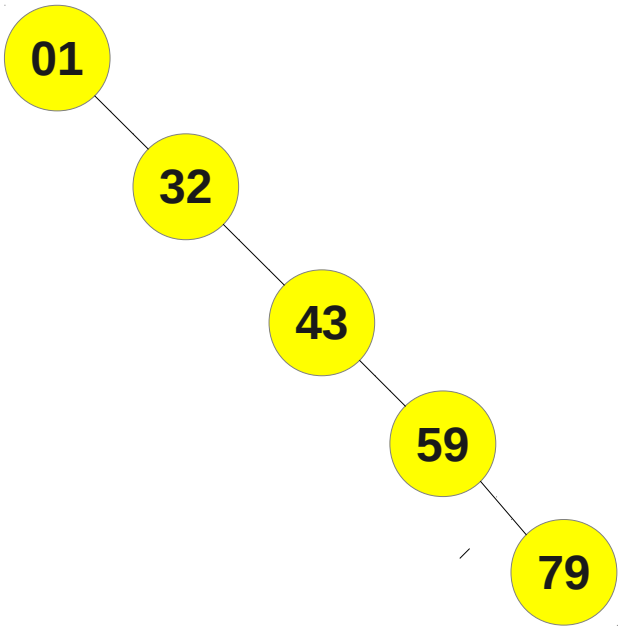
- **Caso 4**: Exemplo: Remover o número 50
  - **Opção Lafore**: Substitua pelo sucessor





PROBLEMA!!!!

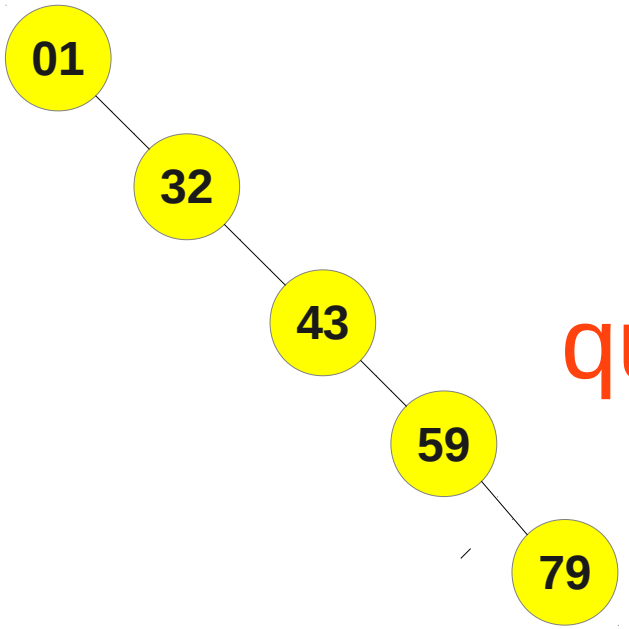
Vamos criar uma árvore com uma sequência de itens ordenados?
{01, 32, 43, 59, 79}



- Problema:
  - Se você inserir n itens de forma ordenada, sua árvore terá n níveis!
    - Tempo de busca alto!
    - Tempo de inserção alto!
    - Tempo de remoção alto!

Lógica!!!!

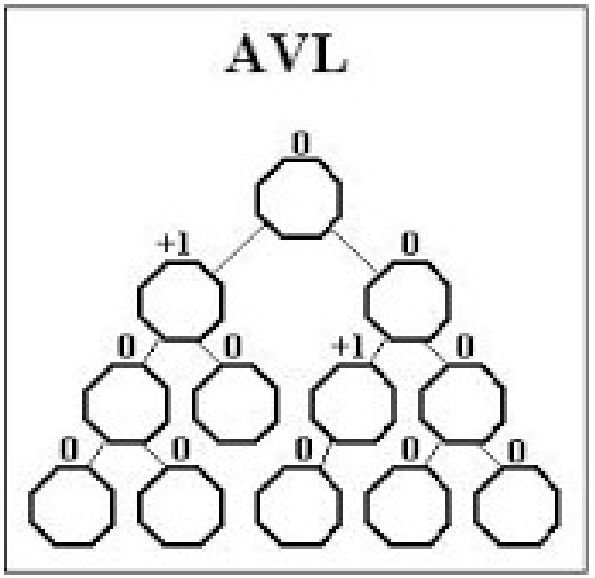
Uma árvore binária simples fica razoavelmente balanceada se inserimos os itens de forma aleatório, porém, se inserimos de forma ordenada, fica horrível!



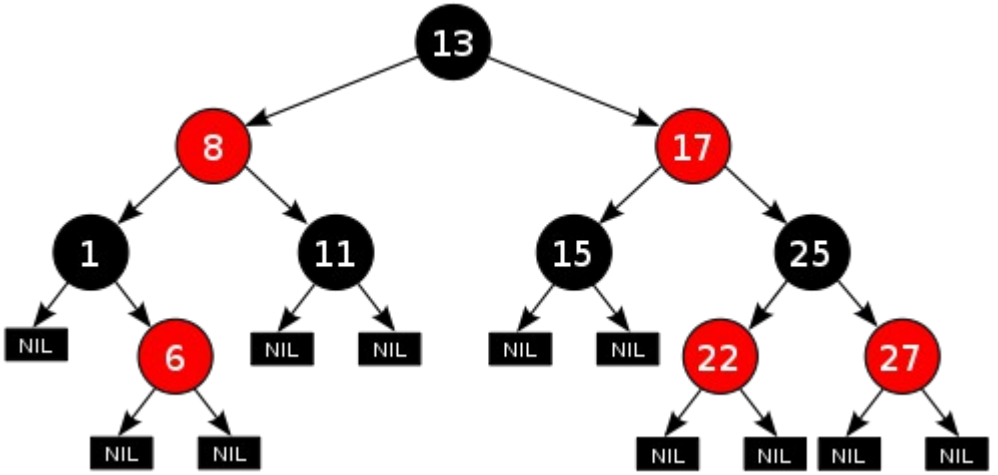
Solução:
Precisamos de mecanismos que permitam balancear essas árvores!

Tipos de Árvores Balanceadas

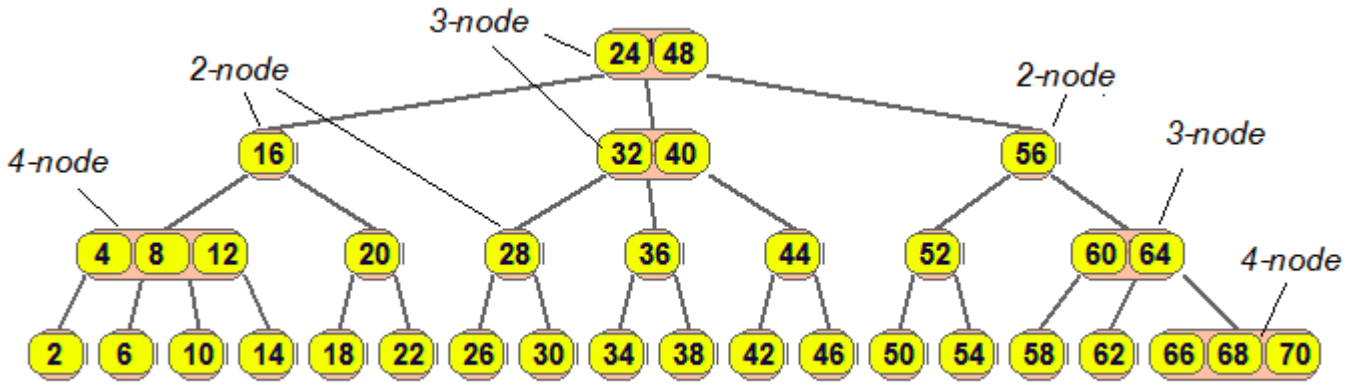
Árvores AVL



Árvores Rubro-Negras



Árvores 2-3-4



Outros ...

Árvores AVL

É o tipo mais antigo de árvore balanceada.
Leva o nome de seus inventores: ***A****delson-****V****elskii e* ***L****andis*

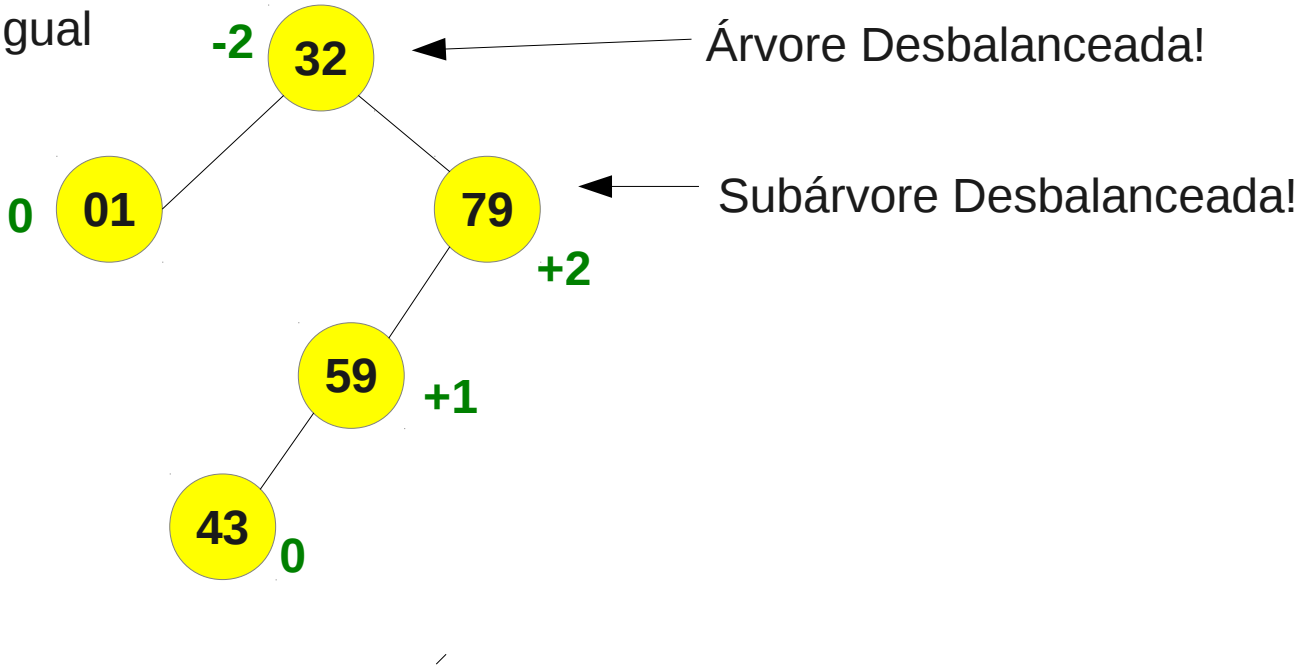
Árvores AVL

- Balancear uma árvore envolve dois passos essenciais:
  1. **Medir o fator de balanceamento de cada nó;**
     - **FB =** Altura Subárvore Esq. - Altura SubÁrvore Dir.
  2. **Ajustar o balanceamento com as devidas rotações;**
     - Existem 4 rotações possíveis:
       - **Rotação Simples a Esquerda;**
       - **Rotação Simples a Direita;**
       - **Rotação Dupla a Direita;**
       - **Rotação Dupla a Esquerda:**

Árvores AVL

- **Medir o fator de balanceamento de cada nó;**
  - **FB =** Altura Subárvore Esq. - Altura SubÁrvore Dir.

Árvore **Desbalanceada** quando o fator de balanceamento é maior ou igual a **+ 2** ou menor ou igual a **-2**

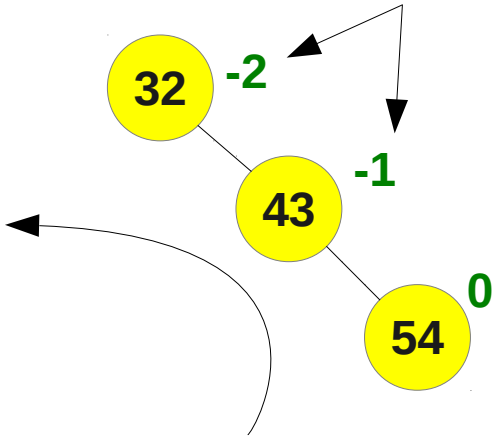


Árvores AVL

- **Ajustar o Balanceamento:**

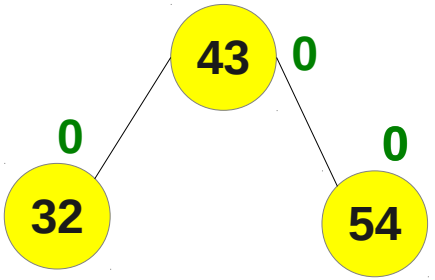
  Caso 1: Desbal. Negativo e Filho Direita Negativo

Desbalanceamento negativo
e filho a direita negativo



RSE(32, 43)

Solução:
Rotação Simples a Esquerda!!!

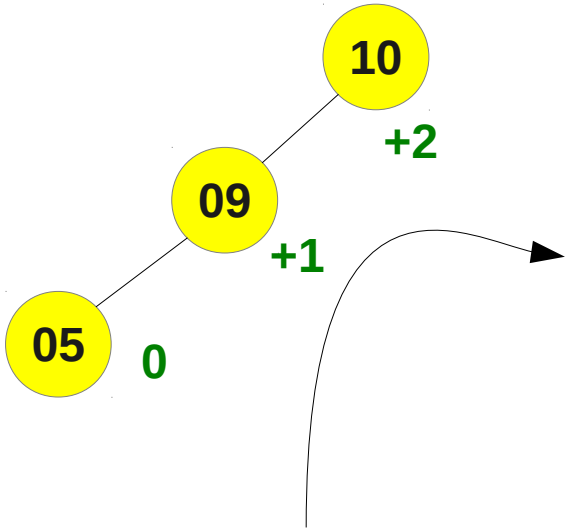


Árvores AVL

- **Ajustar o Balanceamento:**

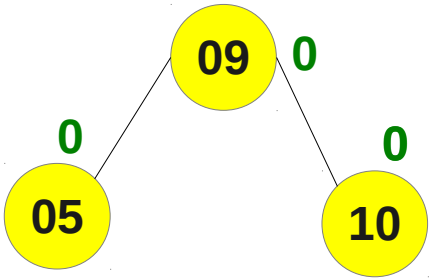
  Caso 2: **Desbal. Positivo e Filho Esquerda Positivo**

Desbalanceamento positivo
e filho a esquerda positivo



RSD(10, 09)

Solução:
Rotação Simples a Direita!!!

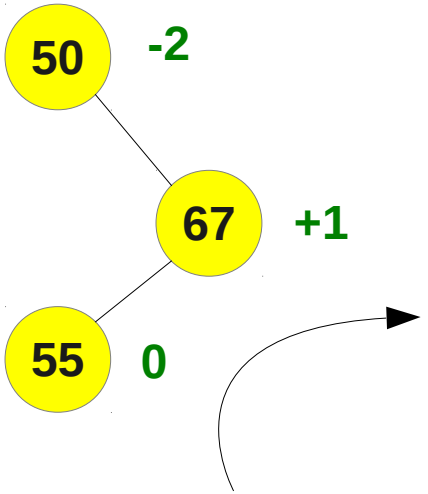


Árvores AVL

- **Ajustar o Balanceamento:**

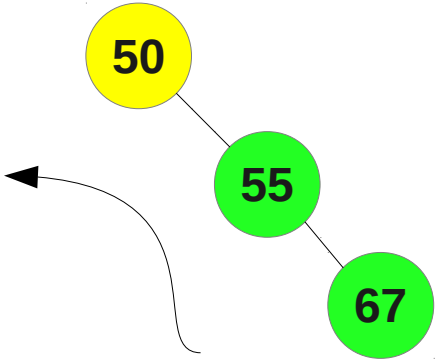
  Caso 3: **Desbal Negativo e filho a direita positivo**

Desbalanceamento negativo
e filho a direita positivo

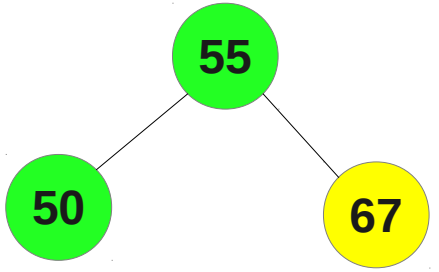


Solução:
Rotação Dupla a Esquerda

Passo 1: Rotação
Simples a Direita
RSD(67,55)



Passo 2: Rotação
Simples a Esquerda
RSE(50,55)

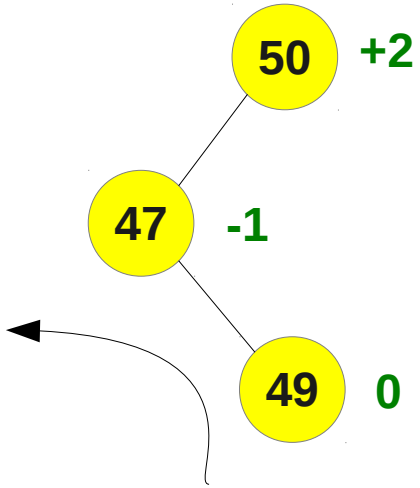


Árvores AVL

- **Ajustar o Balanceamento:**

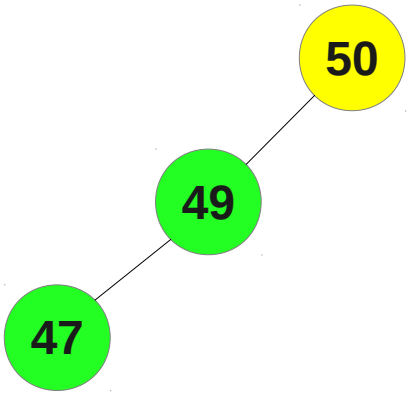
Caso 4: **Desbal Positivo e filho a esquerda negativo**

Desbalanceamento positivo
e filho a esquerda negativo

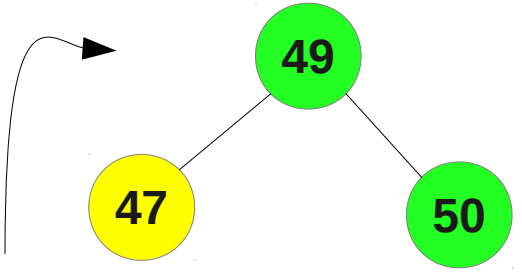


Solução:
Rotação Dupla a Direita

Passo 1: Rotação
Simples a Esquerda
RSD(47,49)



Passo 2: Rotação
Simples a Direita
RSE(50,49)



Árvores AVL

- **Ajustar o Balanceamento:**

  Atenção Especial: Quando nas rotações um nó tem filhos, é necessário que um nó tome o lugar do outro! Acontecendo isso, o nó que teve seu lugar tomado, passa a ser filho do nó que tomou seu lugar!

Vamos ao vídeo exemplo:
http://www.youtube.com/watch?v=JAeQuNsKQWk

Árvores AVL

- **Exemplo de Sala:**
  - **Criar uma árvore AVL com os números:**

    {15, 28, 51, 12, 9, 69, 67, 11, 13, 19, 18}

Árvores AVL

- **Exercícios:**

  1) Crie uma árvore AVL com os números:

  {40,50,73, 10, 8, 6, 30, 9}

  2) Crie uma árvore AVL com os números:

  {16, 29, 52, 13, 10, 70, 68, 12, 14, 20, 19}

O que não veremos sobre Estrutura de Dados?

Árvores Rubro Negras

Cap9 - Lafore

Árvores 2-3-4 e Árvores B

Cap10 - Lafore

Tabelas Hash

Cap11 - Lafore

Grafos

Cap13 - Lafore

Referências

LAFORE, Robert. Estruturas de Dados & algoritmos em Java. Rio de Janeiro. 2004

Introdução sobre Árvores Binárias:
http://www.youtube.com/watch?v=PgZflufXGUU

Arvore Binária de Busca:
http://www.youtube.com/watch?v=XZ0MEDhb4oE

Árvore AVL:
http://www.youtube.com/watch?v=JAeQuNsKQWk

Visão Geral da Estrutura de Dados

Dúvidas? Questionamentos?

ale.garcia.aguado@gmail.com