

# Estrutura de Dados 2

## **Árvore Binária de Busca (ABB)**

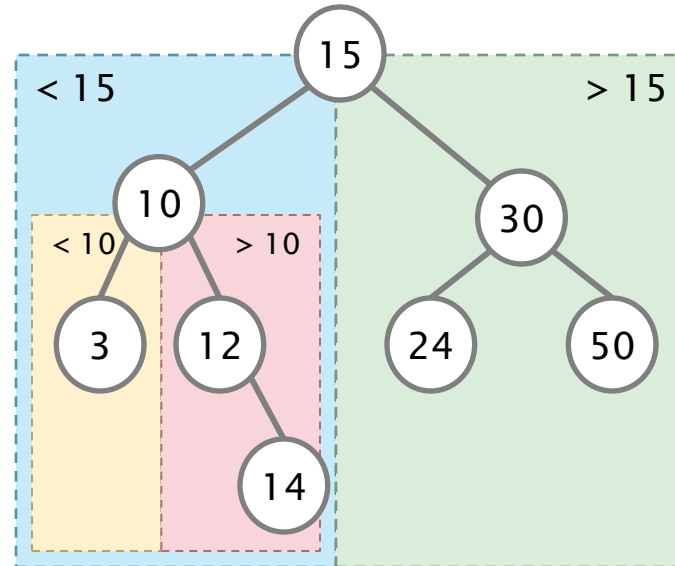
Prof. Rafael Oliveira Vasconcelos

# Definição de ABBs

É uma árvore binária onde, para qualquer nó, as seguintes propriedades são válidas:

- Todas as chaves da subárvore esquerda são menores do que a chave do nó raiz;
- Todas as chaves da subárvore direita são maiores do que a chave da raiz.

# Exemplo de ABB



## Dica!

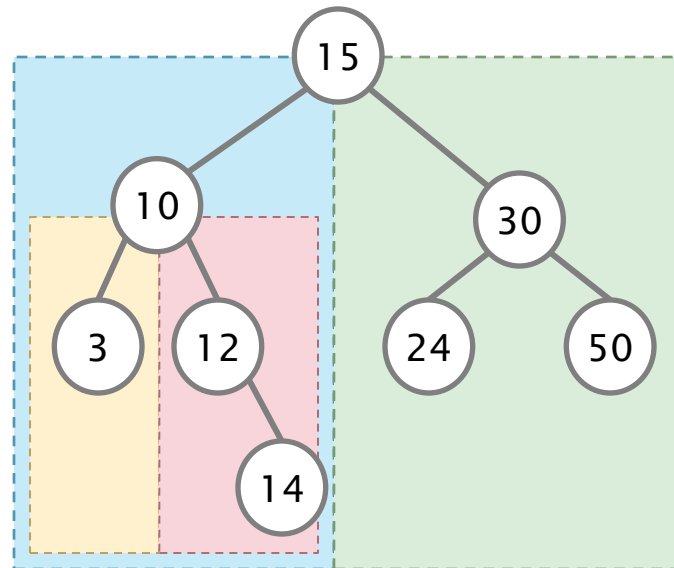
```
void simetrica(NoArvoreBin no) {  
    if (no != null) {  
        simetrica(no.esq);  
        // Visite o no  
        simetrica(no.dir);  
    }  
}
```

Ao se percorrer uma árvore binária de busca em ordem simétrica, o resultado será uma lista ordenada

# Busca em ABB

A busca será feita percorrendo-se o caminho indicado após a comparação feita com a chave da raiz

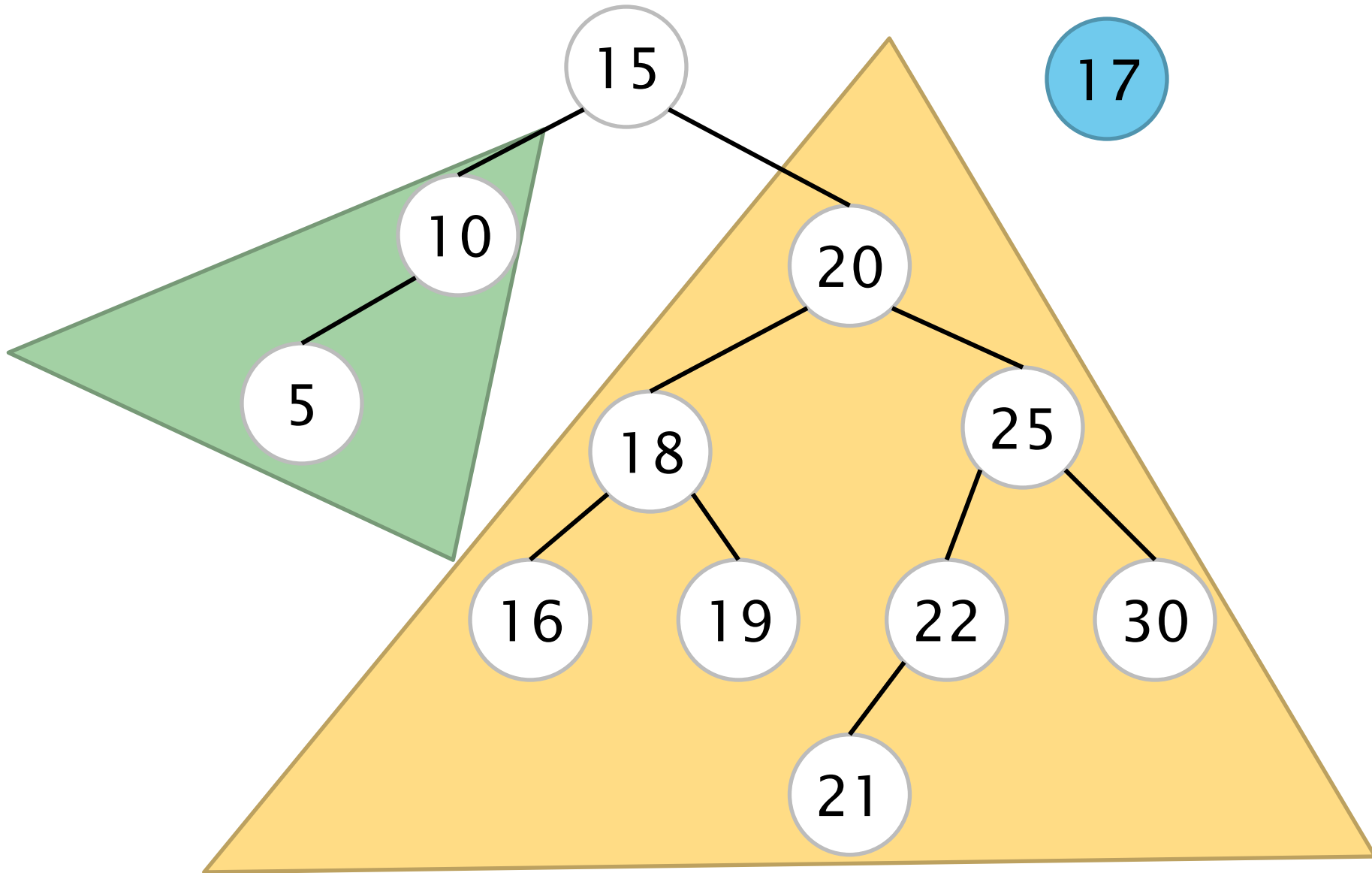
Buscar 14



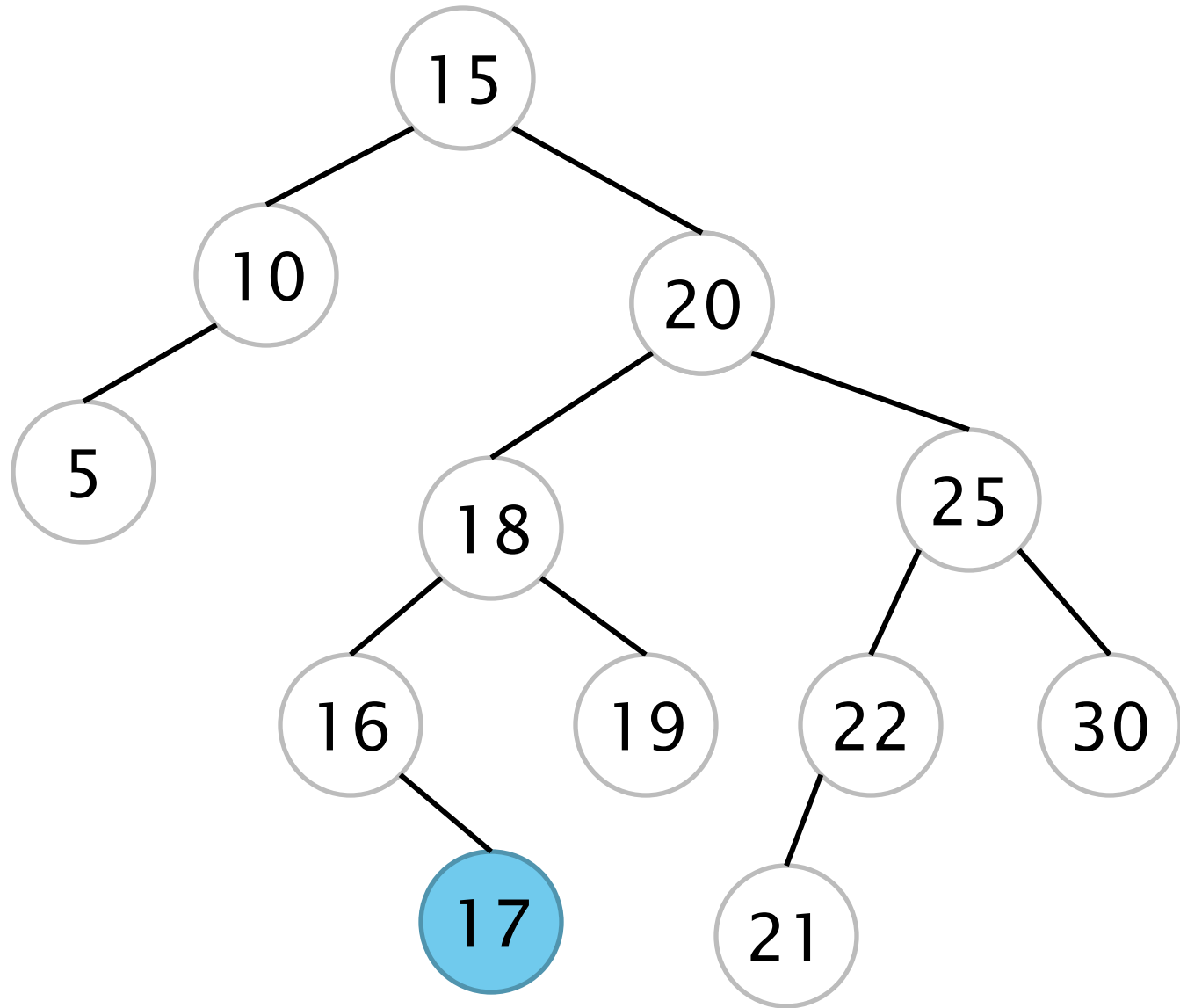
# Busca em ABB - Algoritmo Iterativo

1. Comece a busca pelo nó raiz
2. Enquanto o nó não for **NULL** ou não contiver a chave procurada:
  1. Se a chave do nó for **maior** que a chave procurada vá para o filho à **direita**
  2. Se a chave do nó for **menor** que a chave procurada vá para o filho à **esquerda**
3. Retorne o nó corrente

# Inserção em ABB



# Inserção em ABB



# Inserção em ABB - Algoritmo

1. Começando pelo nó raiz
2. Se o nó for **NULL**, crie um nó com a chave dada
3. Se a chave dada for **maior** que a chave corrente:
  1. Se o nó tiver uma subárvore à **direita**, então insira nela a chave
  2. Senão, crie um nó com a chave dada e este será o filho à direita do nó corrente
4. Se a chave dada for **menor** que a chave corrente:
  1. Se o nó tiver uma subárvore à **esquerda**, então insira nela a chave
  2. Senão, crie um nó com a chave dada e este será o filho à esquerda do nó corrente
5. Se a chave for **igual**, troque a informação associada à chave (não altere a estrutura da árvore)




# Remoção em ABB

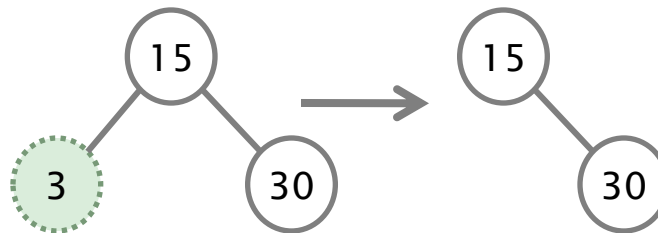
Três casos:

1. Nó folha
2. Nó possui uma subárvore
3. Nó possui duas subárvores

# Remoção em ABB – Passo a Passo

Três casos:

-  1. nó folha  
simplesmente elimina o nó
2. nó possui uma subárvore
3. nó possui duas subárvores



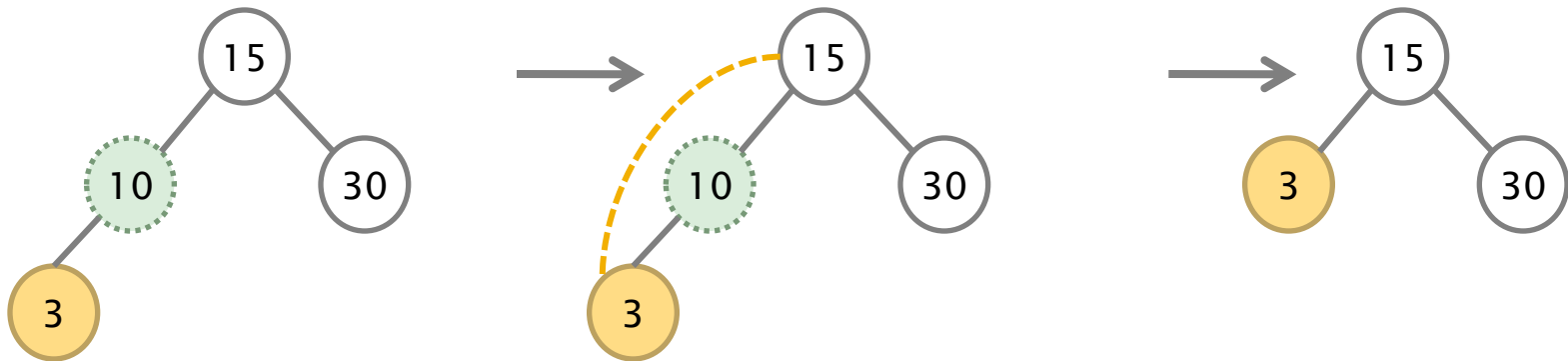
# Remoção em ABB – Passo a Passo

Três casos:

1. nó folha

2. nó possui uma subárvore [dois subcasos: sae, sad]  
pai do nó aponta para o filho do nó (avô aponta para o neto)

3. nó possui duas subárvores



# Remoção em ABB – Passo a Passo

Três casos:

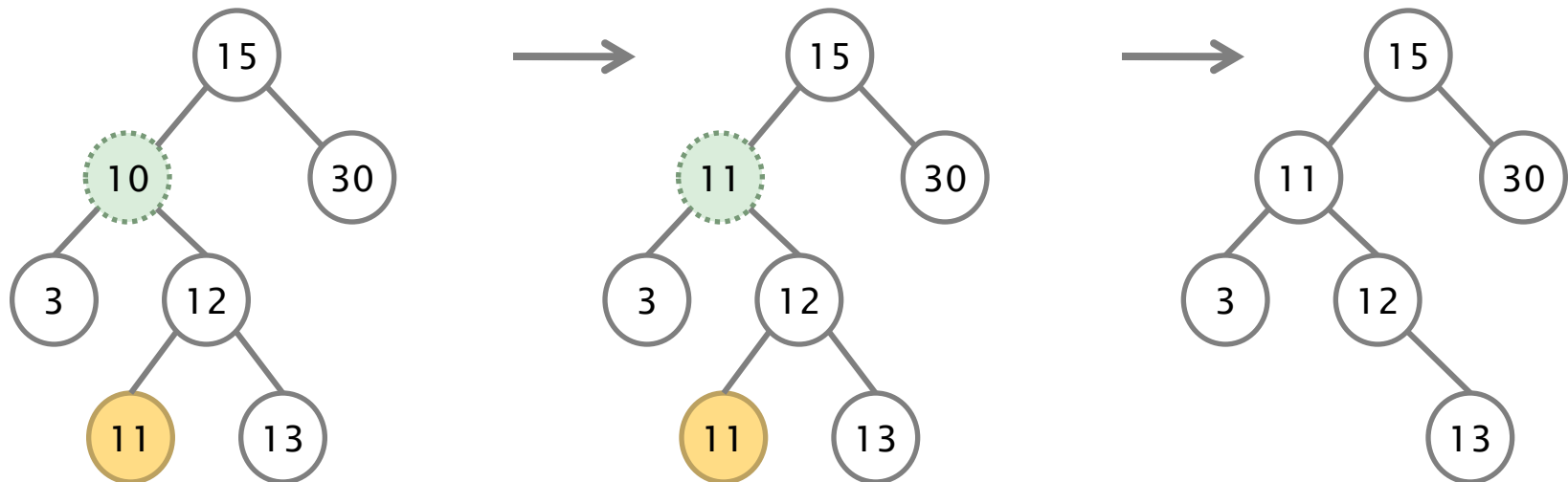
1. nó folha

2. nó possui uma subárvore

3. nó possui duas subárvores

1. coloque a informação do sucessor no nó a ser removido

2. remova o sucessor



# Remoção em ABB – Algoritmo

1. Ache o nó a ser removido
  1. Se ele não tiver filhos, simplesmente remova o nó
  2. Se ele tiver um filho, faça a ligação avô-neto remova o nó
  3. Se ele tiver dois filhos, procure o sucessor, troque o valor do nó pelo do seu sucessor, apague o sucessor

# Exercício

# Exercício ABB

1. Implemente uma árvore binária de busca genérica contendo as principais operações (busca, inserção, remoção e listagem)

Sugestão de implementação da classe Nó:

```
public class No<Chave extends Comparable<Chave>, Valor> {  
    No<Chave, Valor>    pai;  
    No<Chave, Valor>    filhoEsquerdo;  
    No<Chave, Valor>    filhoDireito;  
  
    Chave chave;  
    Valor valor;  
}
```

Interface sugerida para a ABB está no Magister.

# Exercício ABB - Continuação

2. Implemente um programa que utilize a ABB implementada no exercício anterior para armazenar o CPF (inteiro) e nome (string) de uma pessoa (classe Pessoa)
- A chave da classe Pessoa será o CPF
  - O programa deve inserir, consultar e remover pessoas
  - Além disso, deve também listar todas as pessoas armazenadas na ABB em ordem crescente pelo CPF

## Dica!

O tipo <Valor> será substituído pela classe Pessoa

O CPF precisa implementar alguma interface?



# Dúvidas?