



# Algoritmo e Estrutura de Dados II

## COM-112

Aula 15

Vanessa Souza



# Árvores Binárias

---

- ▶ Apesar de termos visto até agora árvores sempre binárias, a definição de árvore não restringe o número de filhos que um nó pode ter.
- ▶ As árvores múltiplas são formadas por nós que possuem mais do que dois filhos e, conseqüentemente, mais do que uma chave associada.





# Árvores Múltiplas

---

- ▶ Um exemplo comum de árvores múltiplas são as árvores 2-3-4.
  - ▶ Árvores multidirecionais, ou seja, cada nó poderá ter um, dois ou três campos de dados e até quatro filhos.
  - ▶ Os números 2,3,4 se referem a quantidade de filhos que um nó poderá ter.
  - ▶ Uma característica importante da árvore 2-3-4 é que um nó deverá ter no mínimo 2 filhos, a não ser que seja uma folha.

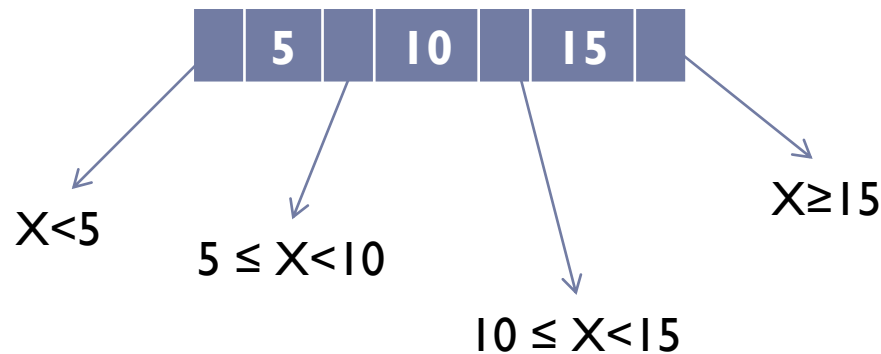




# Árvores Múltiplas

---

- ▶ Sendo assim, o nó da árvore múltipla guarda mais que um valor e mais que um ponteiro para os filhos:



- ▶ As chaves ficam sempre ordenadas no nó.
  - ▶ As chaves associadas a cada filho implicam em relações de ordem da mesma maneira que em árvores binárias.
- 

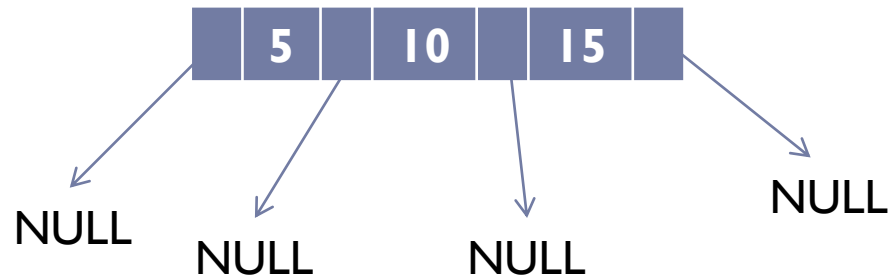




# Árvores Múltiplas

---

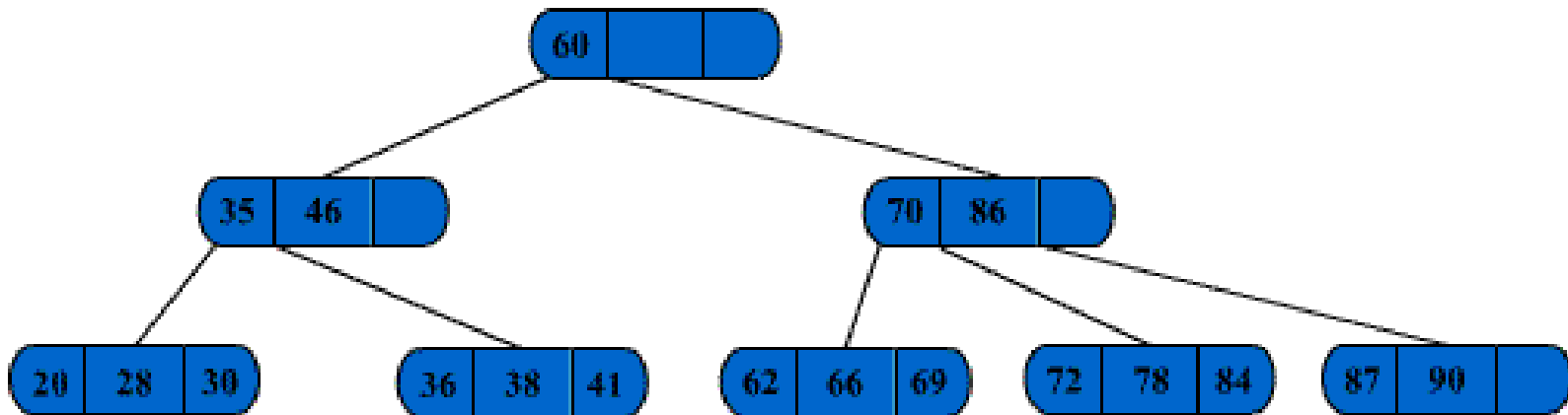
- ▶ Se o nó for folha, ele não tem filhos:





# Árvores Múltiplas

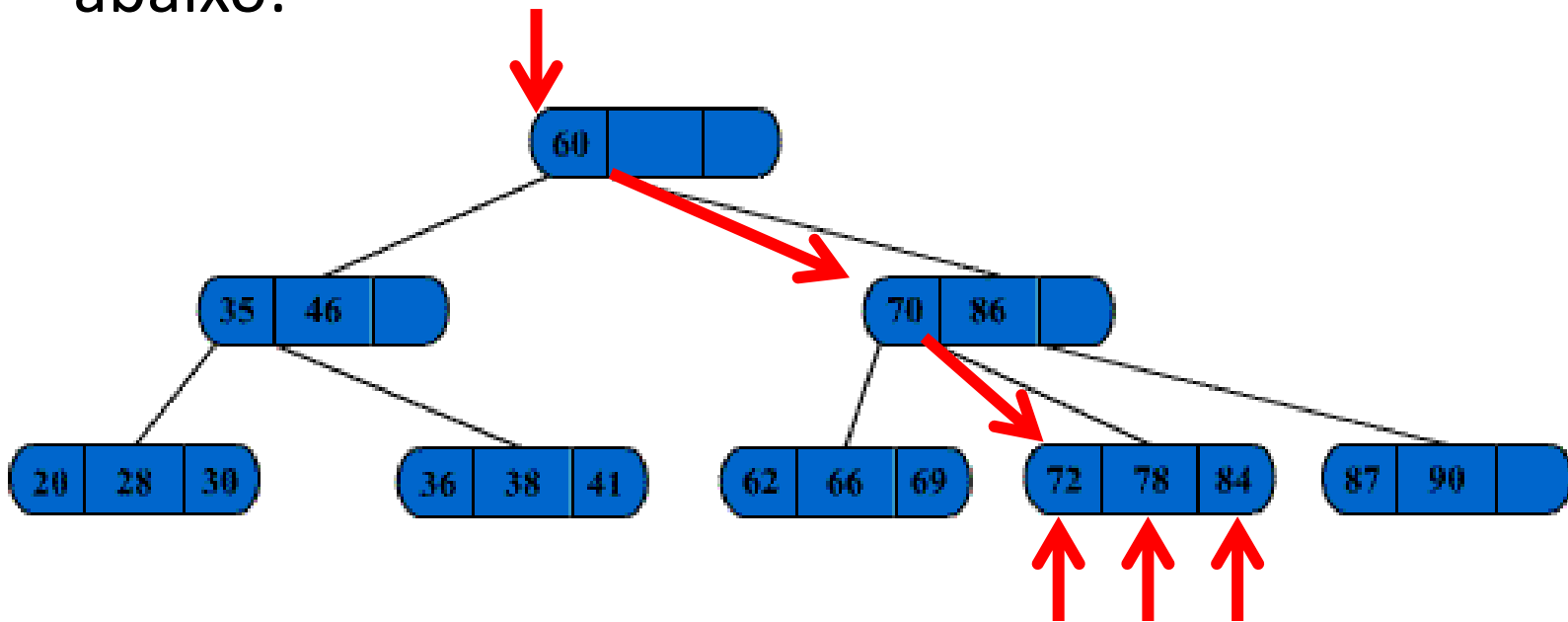
- ▶ Exemplo : Encontrar o nó 84 na árvore múltipla abaixo:





# Árvores Múltiplas

- Exemplo : Encontrar o nó 84 na árvore múltipla abaixo:





# Árvores Múltiplas

---

- ▶ Um exemplo representativo de árvore múltiplas é composto pelas árvores B.
  - ▶ B
  - ▶  $B^+$
  - ▶  $B^*$





Árvore B



## Árvore B

---

- ▶ Inventada por Rudolf Bayer e Edward Meyers McCreightem (1971) enquanto trabalhavam no Boeing *Scientific Research Labs*, a origem do nome (árvore B) não foi definida por estes.
  - ▶ Especula-se que o B venha da palavra balanceamento, do nome de um de seus inventores Bayer ou de Boeing, nome da empresa.
- ▶ São uma generalização das árvores binárias de busca, pois cada nó de uma árvore binária armazena uma única chave de busca, enquanto as **árvores B** armazenam um número maior de chaves em cada nó, ou no termo mais usual para essa árvore, em cada página.
- ▶ *B+Trees are complex disk based trees used to index large amounts of data.*





## Árvore B

---

- ▶ As árvores B são árvores de pesquisa balanceadas projetadas para funcionar bem em discos magnéticos ou outros dispositivos de armazenamento secundário de acesso direto.
- ▶ Em uma aplicação típica de árvore B, a quantidade de dados manipulados é tão grande que não cabem todos na memória principal de uma só vez.
- ▶ São otimizadas para minimizar operações de E/S do disco.
- ▶ Os algoritmos copiam páginas selecionadas do disco para a memória principal conforme necessário e gravam novamente em disco as páginas que foram alteradas.





## Árvore B – Motivação

---

- ▶ Você não verá Árvores B em aplicações normais. Essa estrutura de dados foi inventada com uma motivação para lidar com grande quantidade de dados (que não cabem na memória principal).
- ▶ Sendo assim, é mais aplicada nas indústrias de banco de dados e similares, onde os dados são armazenados no disco e precisam ser acessados, lidos, modificados e reescritos.





## Árvore B

---

- ▶ Uma árvore B é uma forma de construção de árvores de busca em dispositivos de armazenamento secundário (discos), em que o **custo de busca é reduzido** pelo empacotamento de nós da árvore.
- ▶ A ideia é fazer com que cada nó da árvore contenha tanta informação quanto a que cabe num **bloco de disco**.
  - ▶ Um nó de árvore B é normalmente tão grande quanto uma página de disco inteira.
  - ▶ O número de filhos que um nó de árvore B pode ter é então limitado pelo tamanho de uma página do disco.





## Árvore B

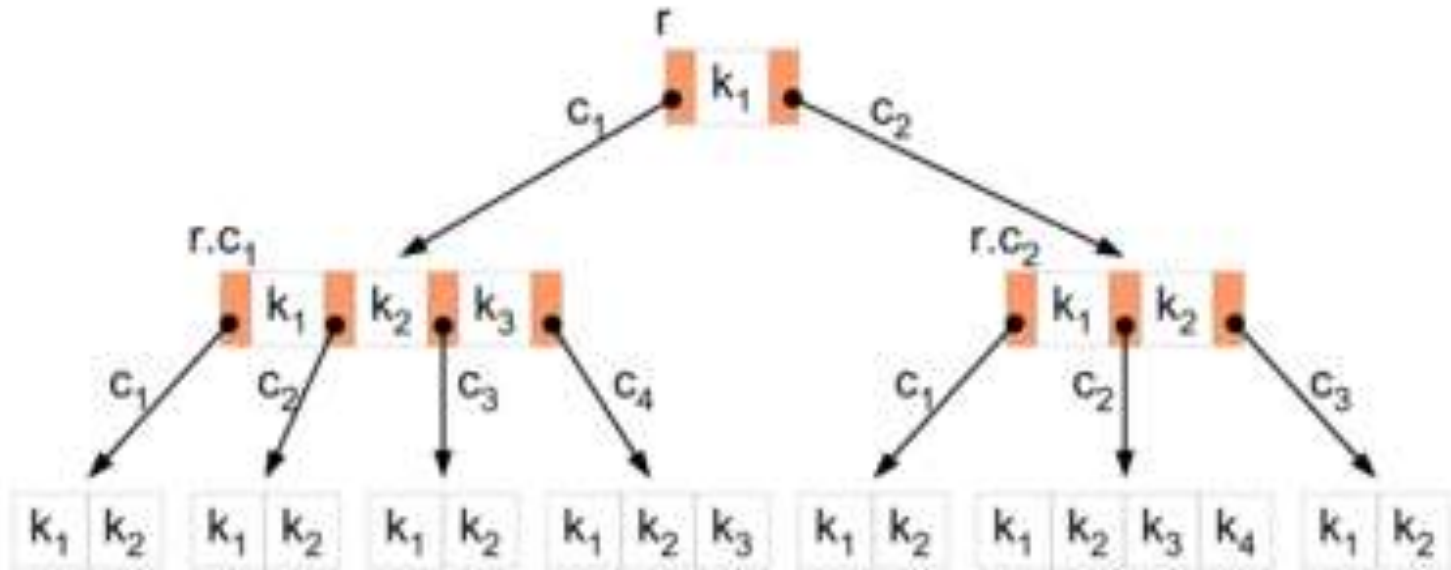
---

- ▶ A consequência disso é que, em geral, a árvore B possui grande largura e pouca profundidade – diminui o número de E/S do disco!!!!
- ▶ Para uma grande árvore B armazenada em um disco, fatores de ramificação entre 50 e 2000 são usados com frequência, dependendo do tamanho de uma chave em relação ao tamanho de uma página.
- ▶ Um grande fator de ramificação reduz drasticamente tanto a altura da árvore quanto o número de acessos ao disco necessário para encontrar qualquer chave.





## Árvore B



- ▶ Se o nó raiz puder ser mantido permanentemente na memória principal, no máximo, apenas dois acessos ao disco são exigidos para encontrar qualquer chave nessa árvore.



## Árvore B – Propriedades

---

- ▶ Uma árvore B de **ordem m** é uma árvore múltipla de procura com as seguintes propriedades:
  - ▶ raiz (a menos que seja folha) tem, pelo menos, 2 filhos;
  - ▶ cada nó interno (não é raiz nem folha) tem k filhos, com  $\lceil m/2 \rceil \leq k \leq m$ ;
  - ▶ cada nó contém k-1 chaves, com  $\lceil m/2 \rceil \leq k \leq m$ ;
  - ▶ todas as folhas da árvore estão no mesmo nível;
  - ▶ as folhas não possuem ponteiros para os filhos





## Árvore B

---

- ▶ Uma árvore B de ordem  $m$  é uma árvore múltipla de procura com as seguintes propriedades:
  - ▶ raiz (a menos que seja folha) tem, pelo menos, 2 filhos;
  - ▶ **cada nó interno (não é raiz nem folha) tem  $k$  filhos, com  $\lceil m/2 \rceil \leq k \leq m$ ;**
  - ▶ **cada nó contém  $k-1$  chaves, com  $\lceil m/2 \rceil \leq k \leq m$ ;**
  - ▶ todas as folhas da árvore estão no mesmo nível;
  - ▶ as folhas não possuem ponteiros para os filhos



# Árvore B

---

## ► Ordem x Grau Mínimo

- Na literatura é comum referir-se a uma árvore B pela sua ordem ou pelo seu grau mínimo
  - A ordem ( $m$ ) é a **quantidade máxima de filhos** que um nó intermediário pode ter.
  - O grau mínimo ( $t$ ) é um inteiro que define os limites inferiores e superiores sobre o **número de chaves** que um nó pode conter.
    - Número máximo de chaves =  $2t - 1$
    - Número mínimo de chaves =  $t - 1$
  - *Uma árvore com grau mínimo  $t$  pode ter, no máximo,  $2t$  filhos em seus nós intermediários.*
    - **$m = 2t$**





## Árvore B

---

### ► Ordem x Grau Mínimo

- Na literatura é comum referir-se a uma árvore B pela sua ordem ou pelo seu grau mínimo
  - A ordem ( $m$ ) é a **quantidade máxima de filhos** que um nó intermediário pode ter.
  - O grau mínimo ( $t$ ) é um inteiro que define os limites inferiores e superiores sobre o **número de chaves** que um nó pode conter.
- Porém a palavra "ordem" é usada de formas diferentes pelos autores, podendo indicar o número máximo de chaves em cada nó, ou até mesmo a ocupação mínima em cada nó.





## Árvore B

---

### ► Ordem x Grau Mínimo

- Na literatura é comum referir-se a uma árvore B pela sua ordem ou pelo seu grau mínimo
  - A ordem ( $m$ ) é a **quantidade máxima de filhos** que um nó intermediário pode ter.
  - O grau mínimo ( $t$ ) é um inteiro que define os limites inferiores e superiores sobre o **número de chaves** que um nó pode conter.

Nesse curso, usaremos a definição acima para a ordem da árvore!





# Árvore B

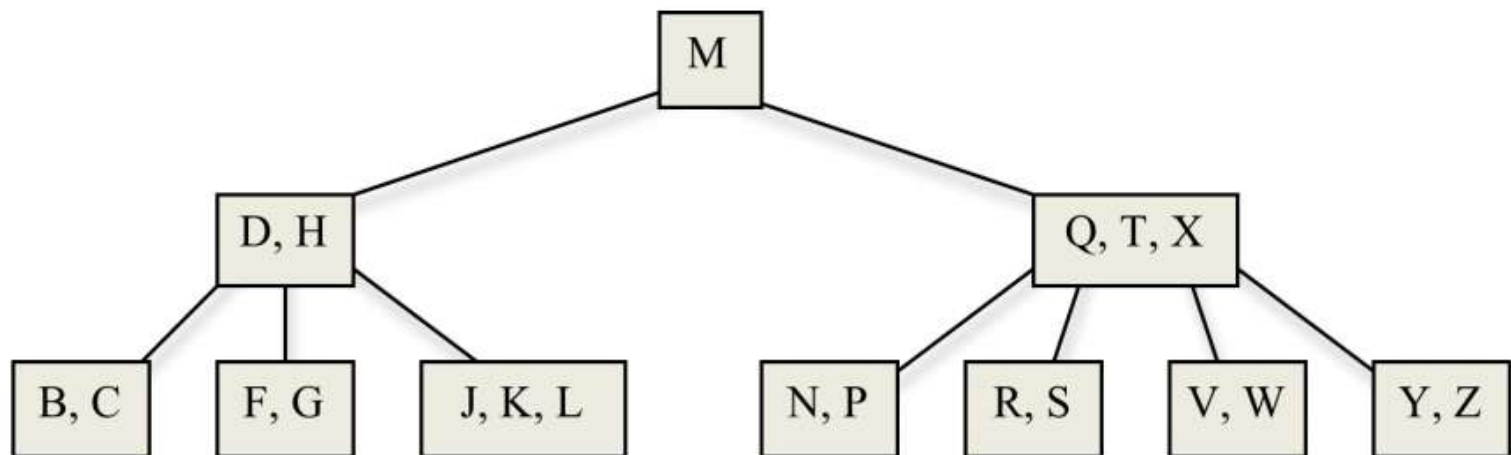
## ► Ordem x Grau Mínimo

### ► Exemplo

$$m = 2t$$

### ► A árvore abaixo tem ordem 4

- Número máximo de chaves nos nós intermediários  $= m - 1 = 3$
- Número mínimo de chaves nos nós intermediários  $= \left\lceil \frac{m}{2} \right\rceil - 1 = 1$





## Árvore B

### ► Ordem x Grau Mínimo

#### ► Exemplo

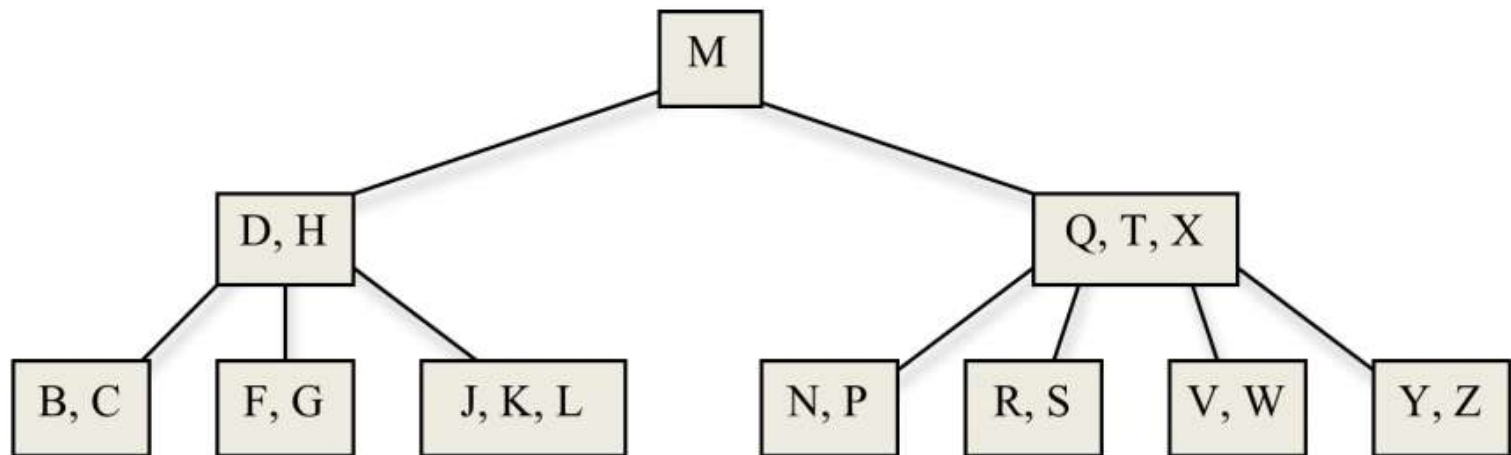
► A árvore abaixo tem ordem 4

► O grau mínimo é 3

**EXCETO A RAIZ**

□ Número máximo de chaves nos nós intermediários =  $2t-1$

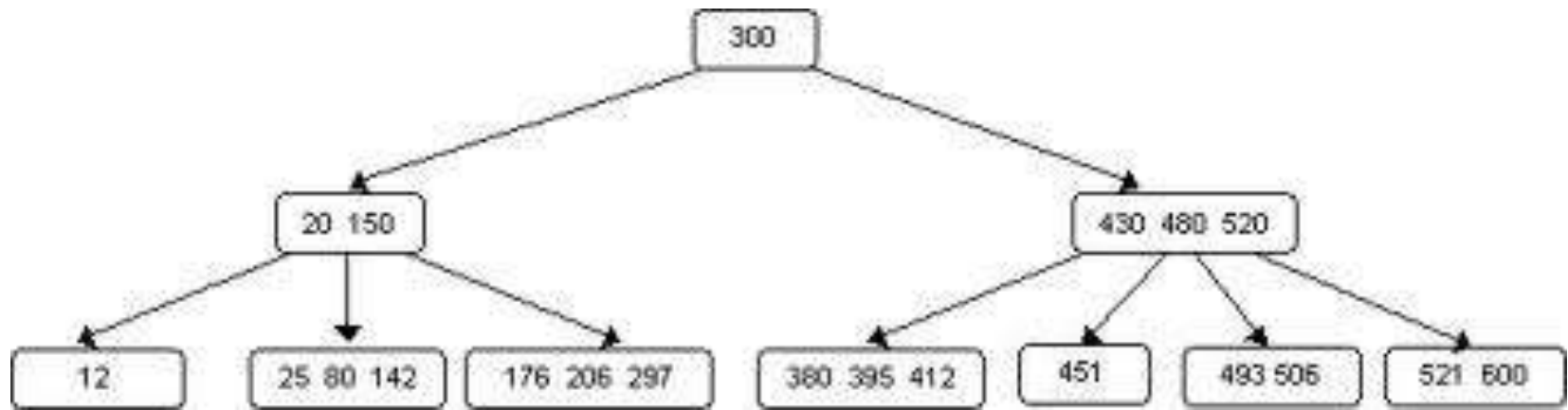
□ Número mínimo de chaves nos nós intermediários =  $t-1$





## Exemplos de árvore B

---



Qual a ordem?

Qual o número máximo de chaves nos nós?

Qual o número mínimo de chaves nos nós?

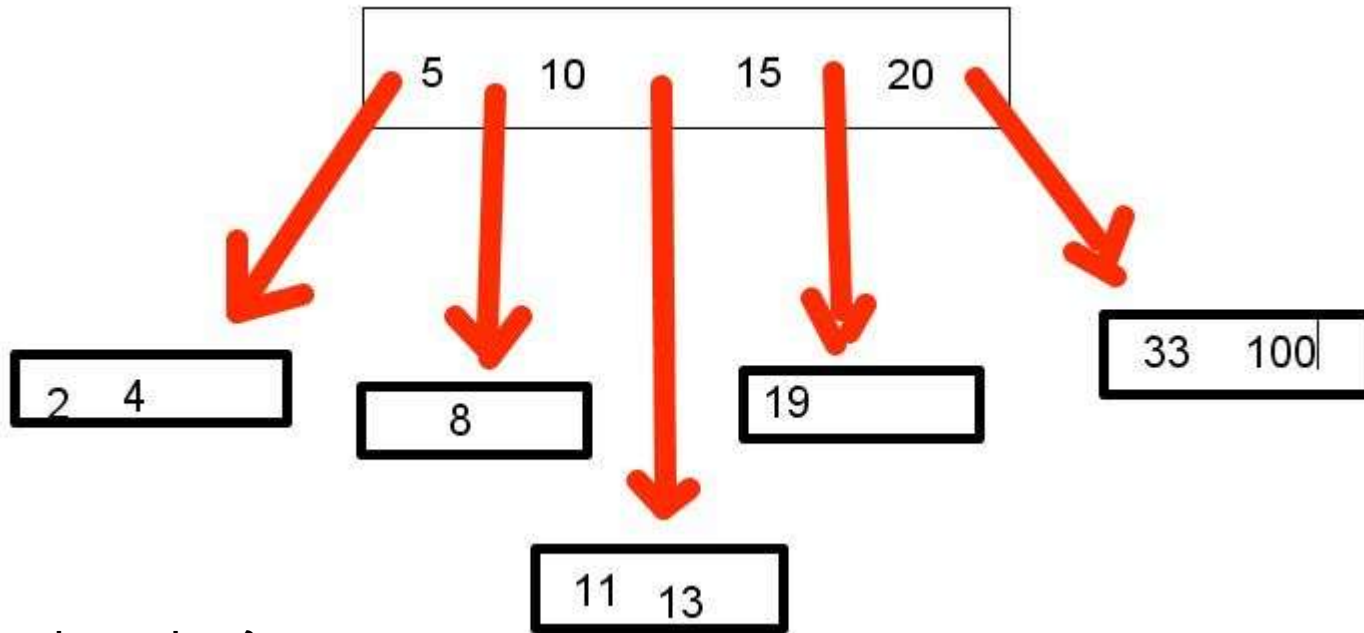
---





## Exemplos de árvore B

---



Qual a ordem?

Qual o número máximo de chaves nos nós?

Qual o número mínimo de chaves nos nós?





## Árvore B – Estrutura da Árvore

---

- ▶ O número de acessos ao disco exigidos para a maioria das operações em uma árvore B é proporcional à altura da árvore.
- ▶ É provado que a altura de uma árvore B, **no pior caso** é:

$$h \leq \log_t \left( \frac{n+1}{2} \right)$$

Onde:

t = grau mínimo

n = número de nós

Raiz contém uma chave e todos os outros nós contém  $t-1$  chaves.

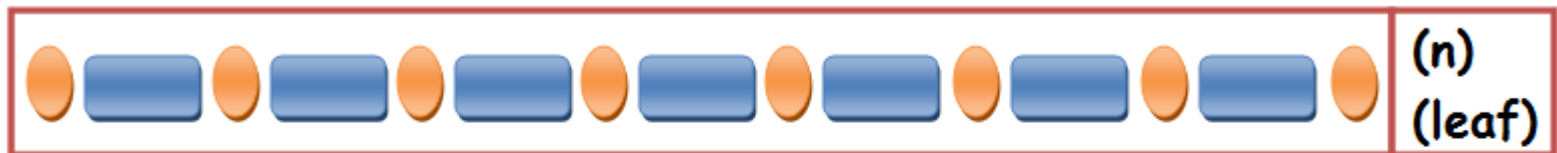




## Árvore B – Estrutura do Nó

---

- ▶ O nó de uma árvore B deve conter as seguintes informações:
  - ▶ Número de chaves armazenadas no nó
  - ▶ Chaves
  - ▶ Ponteiros para os filhos
  - ▶ E pode conter uma indicação se ele é folha ou não



 - child pointer

 - key

(n) - number of keys currently stored

(leaf) - is this a leaf node, boolean





## Árvore B – Estrutura do Nó

---

- ▶ Outra alternativa é implementar nós folhas com estrutura diferente, já que ele não necessita dos ponteiros para os filhos
- ▶ Se o nó for folha, ele terá:
  - ▶ Número de chaves no nó
  - ▶ Chaves
- ▶ Essa é a forma mais correta de implementar, porém também é mais complexa porque será necessário lidar com duas estruturas de nós diferentes.



# Operações Básicas sobre a árvore B



# Operações Básicas

---

- ▶ Busca
- ▶ Inserção
- ▶ Remoção





# Operações Básicas

---

## ► Busca

- Pesquisar em uma árvore B é muito semelhante a pesquisar em uma árvore de pesquisa binária, exceto pelo fato de que, em vez de tomar uma decisão de ramificação binária, toma-se a decisão de ramificação de  $(n[x] + 1)$  vias.
- A busca é feita primeiro nas chaves do nó, depois passa-se para os nós intermediários até encontrar a chave procurada.





## Operações Básicas – INSERÇÃO

---

- ▶ A inserção é bem mais complexa em árvores B do que em árvores binárias.
- ▶ Assim como nas árvores binárias, procuramos pela posição **de folha** em que devemos inserir a nova chave.
- ▶ Ao localizar a folha correta, a chave deve ser inserida de forma **ordenada** no nó.
- ▶ O problema ocorre quando a folha está cheia...



# Inserção em Árvore B

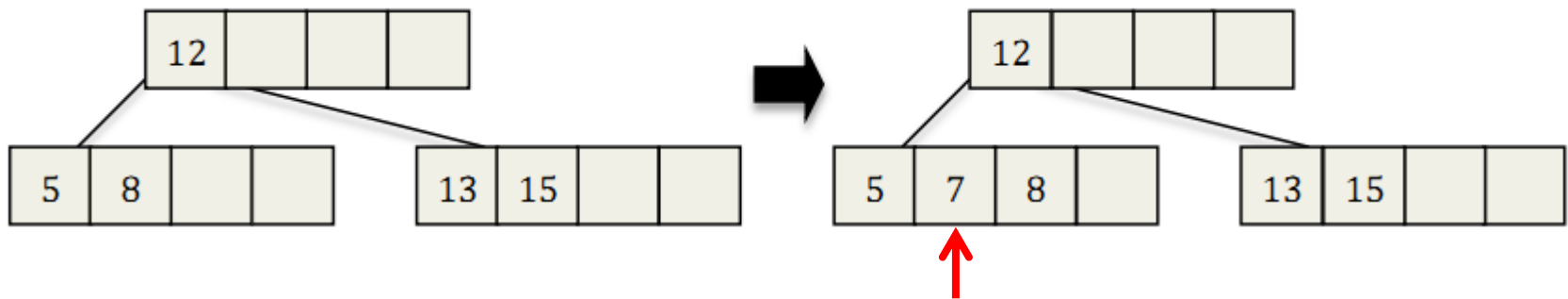




# Inserção

► Três situações podem ocorrer ao inserir uma nova chave:

1. A nova chave é colocada em uma folha que ainda tem espaço. Exemplo: incluir a chave 7.



Busca-se a posição correta da chave dentro do nó folha e insere.



# Inserção

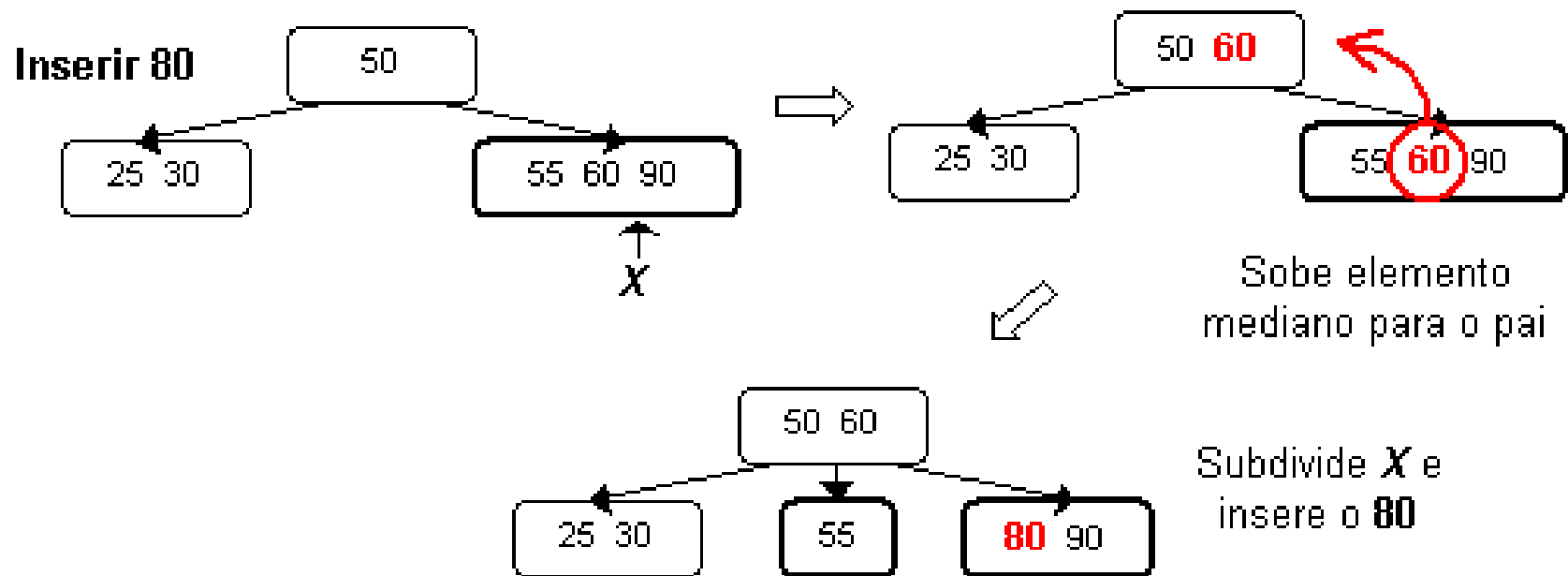
---

- ▶ Três situações podem ocorrer ao inserir uma nova chave:
  - 2. A folha (**X**) onde a chave precisa ser incluída está cheia, mas ainda é possível criar um novo filho para o pai desta folha.
- ▶ Será necessário realizar uma **subdivisão** de nós.
  - 1. Passar o elemento mediano de **X** para seu pai
  - 2. Subdividir **X** em dois novos nós com  **$t - 1$**  elementos
  - 3. Inserir a nova chave
- ▶ Operação de SPLIT





# Inserção



Passos para a inserção da chave 80 em uma árvore B com  $t = 2$



## Inserção

---

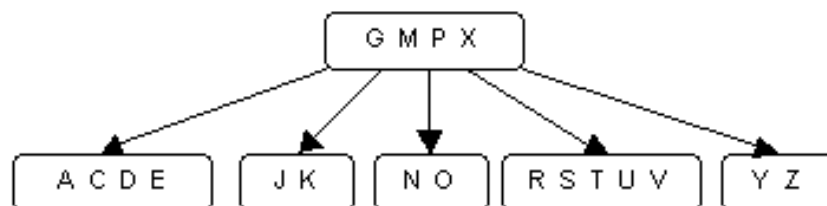
- ▶ Três situações podem ocorrer ao inserir uma nova chave:
  - 3. Não há possibilidade de criar um novo filho para o pai da folha cheia ou o nó ascendente também está cheio.
    - ▶ Se o pai de **X** também estiver cheio, repete-se recursivamente a subdivisão para o pai de **X**.
    - ▶ No pior caso terá que aumentar a altura da árvore B para poder inserir o novo elemento.





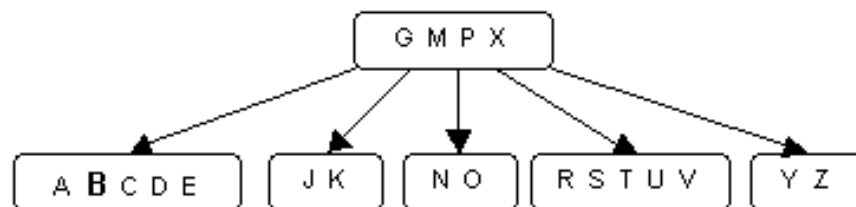
# Inserção

Inicial

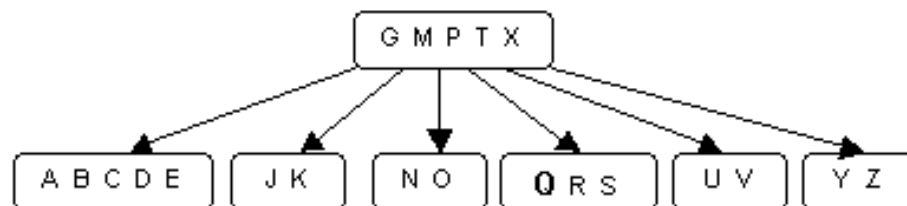


m=6

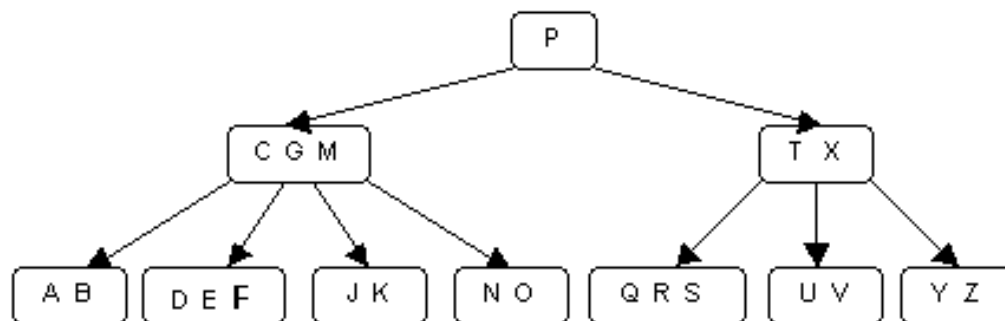
Inserir B



Inserir Q



Inserir F





## Exemplo 1

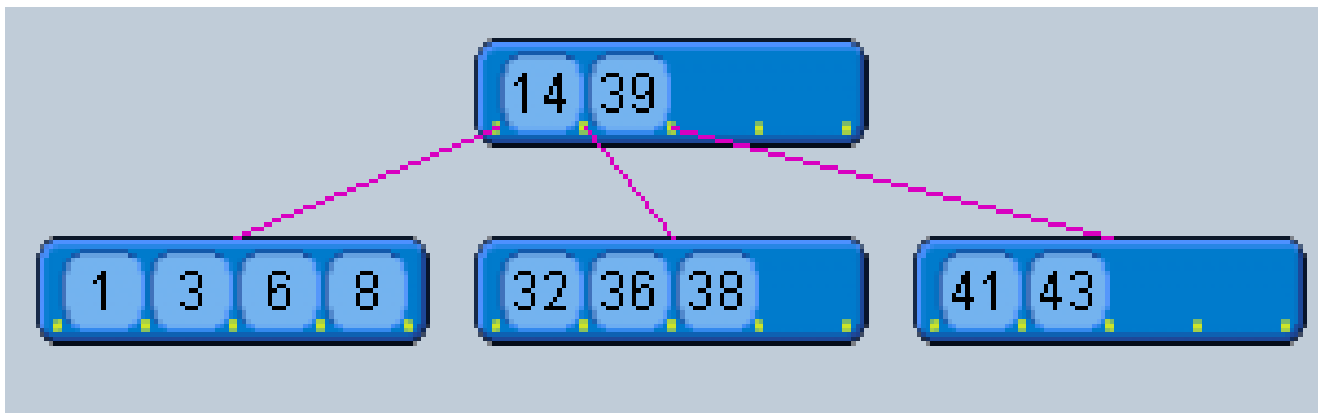
---

- ▶ Mostre os resultados de inserir as chaves a seguir em uma árvore B de **ordem 5** inicialmente vazia:
- ▶ 14, 39, 1, 6, 41, 32, 8, 38, 43, 3, 36.



## Exemplo 1

- ▶ Mostre os resultados de inserir as chaves a seguir em uma árvore B de **ordem** 5 inicialmente vazia:
- ▶ 14, 39, 1, 6, 41, 32, 8, 38, 43, 3, 36.



<http://slady.net/java/bt/view.php>



## Exemplo 2

---

- ▶ Mostre os resultados de inserir as chaves a seguir em uma árvore B de **ordem 4** inicialmente vazia:
- ▶ 14, 39, 1, 6, 41, 32, 8, 38, 43, 3, 36.

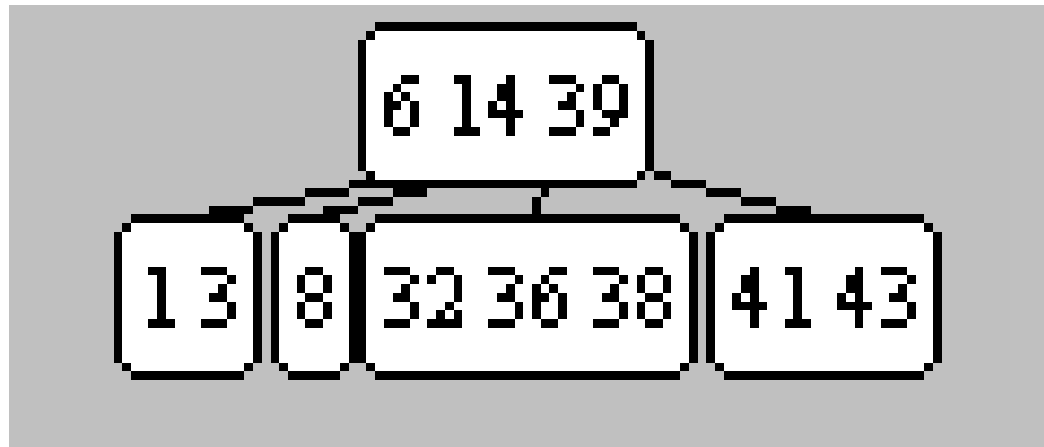




## Exemplo 2

---

- ▶ Mostre os resultados de inserir as chaves a seguir em uma árvore B de ordem 4 inicialmente vazia:
- ▶ 14, 39, 1, 6, 41, 32, 8, 38, 43, 3, 36.

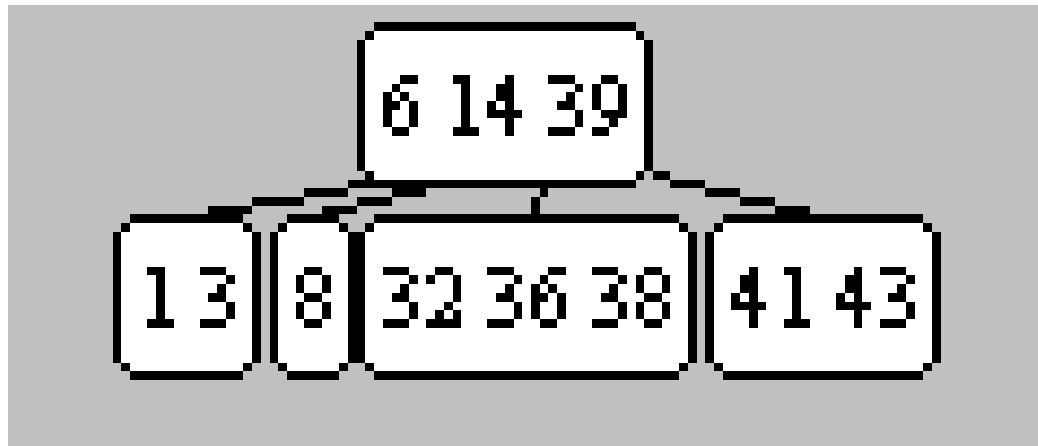




## Exemplo 3

---

- ▶ Inserir a chave 30 na árvore abaixo



<http://www.cs.unm.edu/~rlpm/499/ttft.html>

---

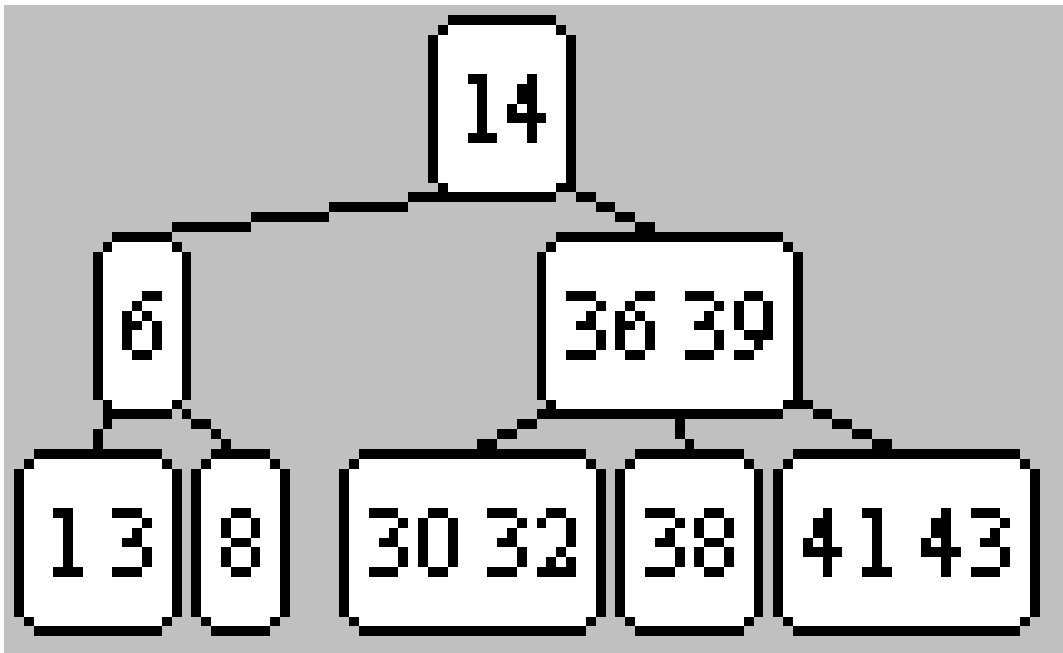




## Exemplo 3

---

- Inserir a chave 30 na árvore abaixo



<http://www.cs.unm.edu/~rlpm/499/ttft.html>

---





## Exercício

---

- ▶ Mostre os resultados de inserir as chaves a seguir em uma árvore B de ordem 5 inicialmente vazia:
- ▶ F, S, Q, K, C, L, H, T, V, W, M, R, N, P, A, B, X, Y, D, Z, E