Árvore-B
Busca e Inserção

Organização e Recuperação de Dados
Profa. Valéria

UEM – CTC – DIN

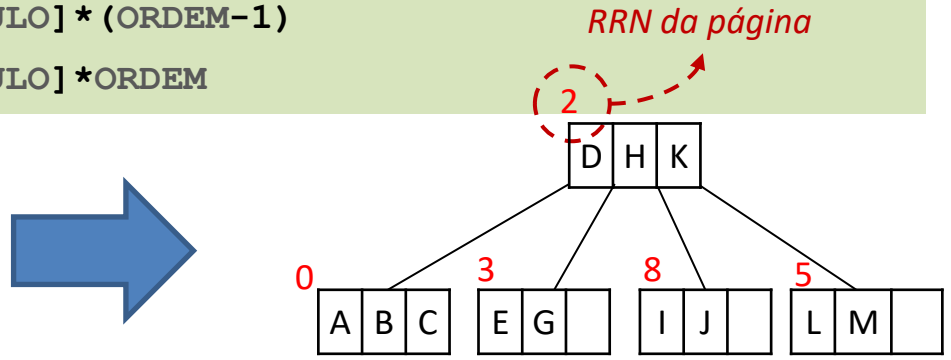
Slides preparados com base no Cap. 8 do livro FOLK, M.J. & ZOELLICK, B. *File Structures*. 2nd Edition, Addison-Wesley Publishing Company, 1992.

Busca e inserção em árvore-B

- Possível estrutura de página de árvore-B em C

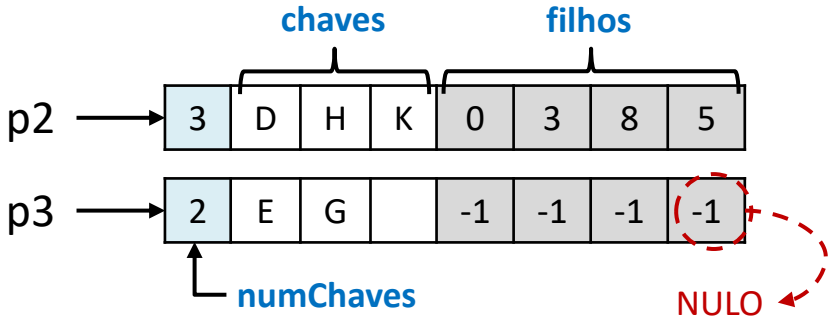
```
class Pagina:
    def __init__(self) -> None:
        self.numChaves: int = 0
        self.chaves: list = [NULO]*(ORDEM-1)
        self.filhos: list = [NULO]*ORDEM
```

- Parte de uma árvore-B de ordem 4



- O arquivo da árvore-B:
  - Registros de tamanho fixo
  - Cada registro armazena uma página da árvore-B

Representação das páginas 2 e 3:



Busca na árvore-B

- Algoritmo de busca na árvore-B
  - Utiliza recursão para “descer” na árvore
- A **busca** sempre inicia na página **raiz**
- Para cada página lida, busca-se internamente pela chave
  - Se a chave for encontrada na página, a função retorna:
    - **Verdadeiro**, o **RRN** da página e a **POS** da chave na lista **chaves**
  - Se a chave não for encontrada, a função **busca** é chamada recursivamente
    - As chamadas ocorrerão até que a chave seja encontrada ou até que se encontre um ponteiro nulo (em uma folha)
    - Se encontrar um ponteiro nulo, a função retorna **Falso**, **NULO**, **NULO**

Busca na árvore-B

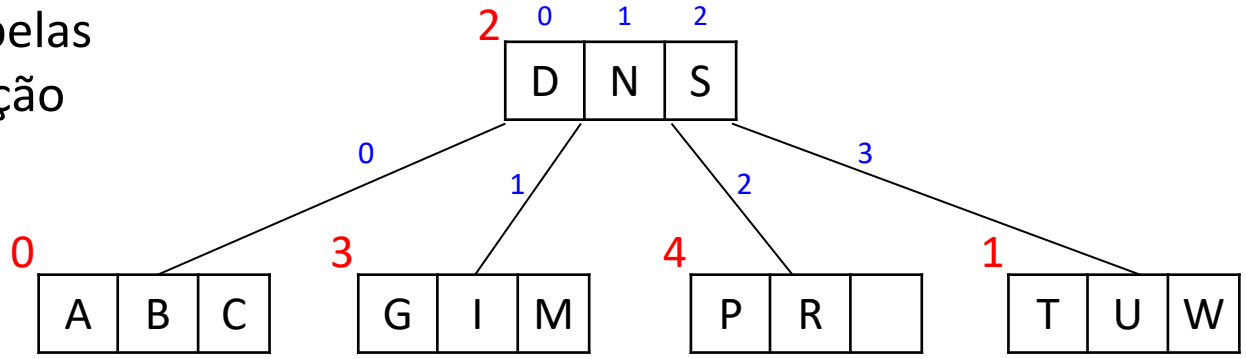
```
FUNÇÃO buscaNaArvore(chave, rrn)
 se rrn == NULO então      # condição de parada da recursão
    retorne Falso, NULO, NULO
 senão
   leia a página armazenada no rrn para pag
   achou, pos = buscaNaPagina(chave, pag)
   # POS recebe a posição em que CHAVE ocorre em PAG
   # ou deveria ocorrer se estivesse em PAG
   se achou então
        retorne Verdadeiro, rrn, pos
   senão
        # busque na página filha
        retorne buscaNaArvore(chave, pag.filhos[pos])
   fim se
 fim se
fim FUNÇÃO
```

Pesquisa em árvore-B

```
FUNÇÃO buscaNaPagina(chave, pag)
    faça pos receber 0
    enquanto pos < pag.numChaves e chave > pag.chaves[pos] faça
        incremente pos
    se pos < pag.numChaves e chave == pag.chaves[pos] então
        retorne Verdadeiro, pos
    senão
        retorne Falso, pos
fim FUNÇÃO
```

Pesquisa em árvore-B

Exercício: Simule a busca pelas chaves ***K*** e ***P*** usando a função ***search***




```
FUNÇÃO buscaNaArvore(chave, rrn)
 se rrn == NULO então      # condição de parada da recursão
    retorne Falso, NULO, NULO
 senão
    leia a página armazenada no rrn para pag
    achou, pos = buscaNaPagina(chave, pag)
    # POS recebe a posição em que CHAVE ocorre em PAG
    # ou deveria ocorrer se estivesse em PAG
    se achou então
          retorne Verdadeiro, rrn, pos
    senão
          # busque na página filha
          retorne buscaNaArvore(chave, pag.filhos[pos])
    fim se
 fim se
fim FUNÇÃO
```

Inserção, divisão e promoção

- Algoritmo de inserção em árvore-B
  - Começa com uma busca e prossegue até alcançar uma folha
    - Se achar a chave, gera um erro e termina
  - Uma vez localizada a posição de inserção (**SEMPRE em uma folha**), pode ser necessário realizar divisão e promoção
  - O algoritmo pode então ser pensado em 3 partes:
    1. Busca pela chave na página atual, antes da chamada recursiva
    2. Chamada recursiva para "descer" na árvore até encontrar um ponteiro nulo, que estará em uma folha
    3. Inserção, divisão e promoção (se necessário) executadas no retorno da chamada recursiva, fazendo com que esses processos ocorram na "subida" da árvore

Inserção na árvore-B

Função ***insereNaArvore*** **(chave, rrnAtual)**

- Parâmetros:
  1. **chave**: contém a chave a ser inserida
  2. **rrnAtual**: contém o RRN da página atualmente em uso (inicialmente, a raiz)

- Variável utilizadas para armazenar valores de retorno da função
  1. **chavePro**: contém a chave que está sendo promovida ou **NULO** se não houver promoção
  2. **filhoDpro**: contém o RRN do seu filho direito da **chavePro**, que também pode ser **NULO**
  3. **promo**: contém **Verdadeiro** se houver promoção ou **Falso**, caso contrário

Inserção, divisão e promoção

- Variáveis locais importantes da função:
  - **pag**: página que está sendo examinada
  - **novapag**: nova página que é criada caso ocorra uma divisão
  - **pos**: posição da chave em **pag**, se ela estiver lá; caso contrário, a posição em que ela deve ser inserida (ou a posição do RRN para a próxima página)

Inserção na árvore-B

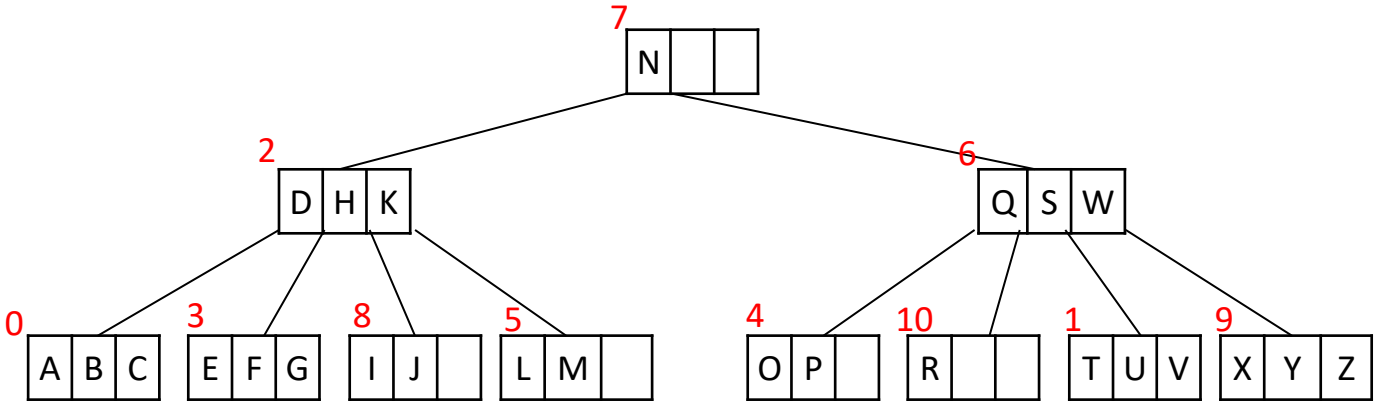
```
FUNÇÃO insereNaArvore(chave, rrnAtual)
    se rrnAtual == NULO então  # condição de parada da recursão
          chavePro = chave
          filhoDpro = NULO
          retorne chavePro, filhoDpro, Verdadeiro
    senão
          leia a página armazenada em rrnAtual para pag
          achou, pos = buscaNaPagina(chave, pag)
    fim se
    se achou então
          gere um erro de valor – "Chave duplicada"
    fim se
    chavePro, filhoDpro, promo = insereNaArvore(chave, pag.filhos[pos])
    …
    # continua no próximo slide
```

Inserção na árvore-B

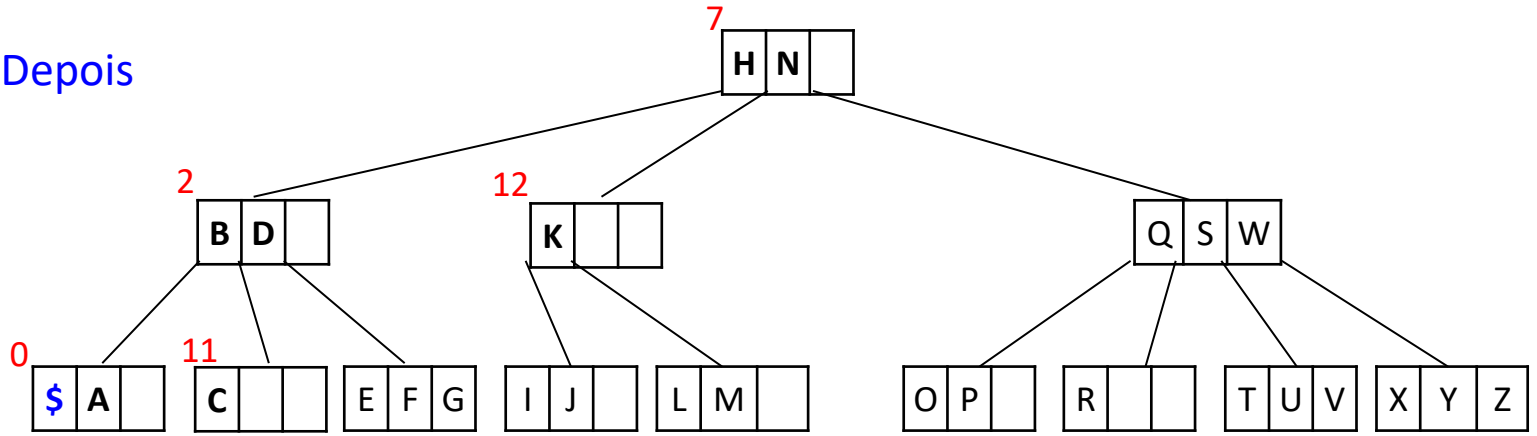
```
   # continuação da função insereNaArvore
   # logo após a chamada recursiva
   se não houve promo então
        retorne NULO, NULO, Falso
   senão
      se existe espaço em pag para inserir chavePro então
        insira chavePro e filhoDpro(chave promovida e filha) em pag
        escreva pag no arquivo em rrnAatual
        retorne NULO, NULO, Falso
      senão
        chavePro,filhoDpro,pag,novapag = divide(chavePro,filhoDpro,pag)
        escreva pag no arquivo em rrnAtual
        escreva novapag no arquivo em filhoDpro
        retorne chavePro, filhoDpro, Verdadeiro
      fim se
   fim se
fim função
```

Inserção, divisão e promoção

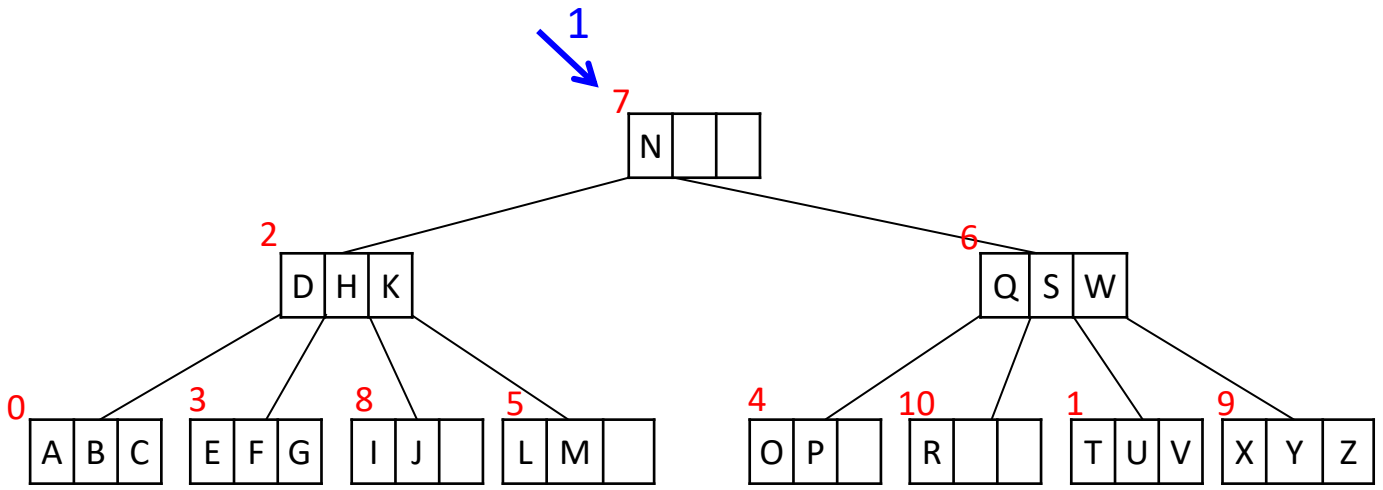
- Exemplo: inserção do caractere **$** na árvore-B abaixo



Depois

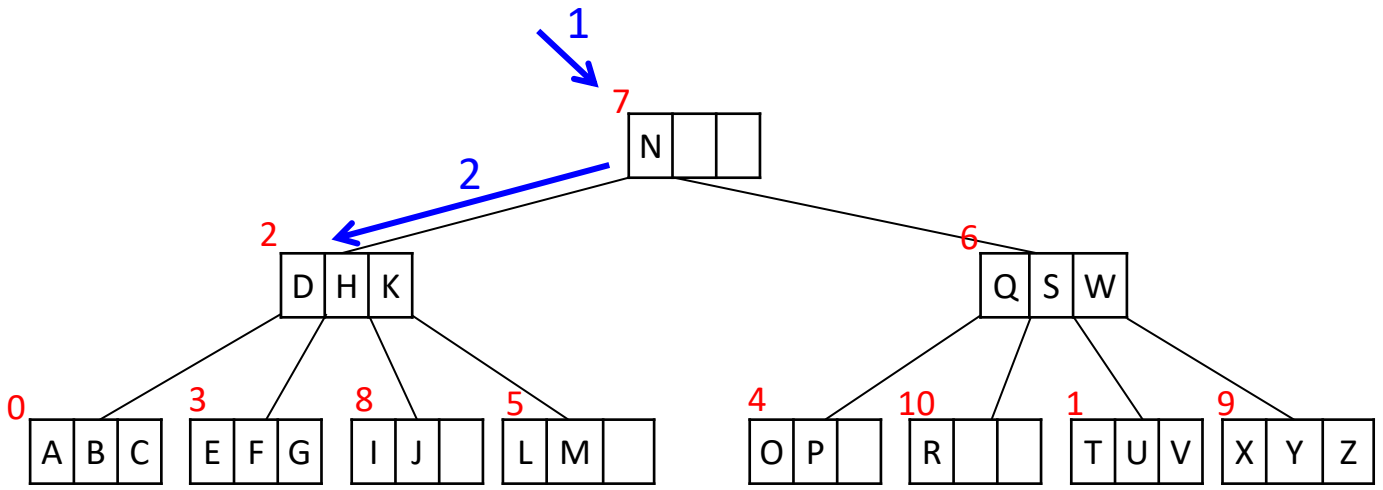


Inserção, divisão e promoção



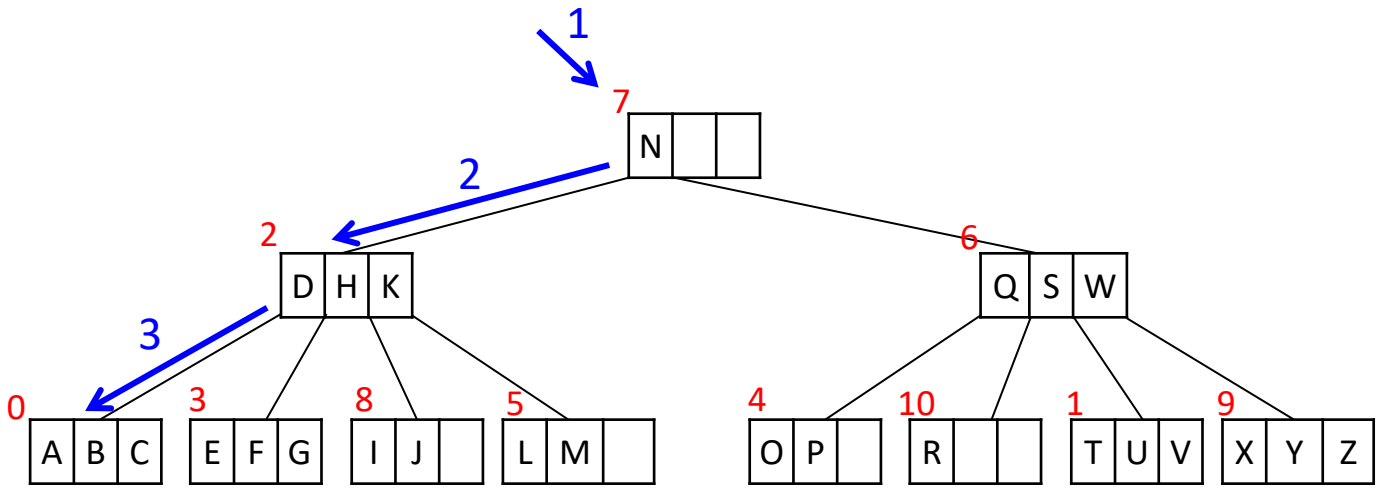

```
1: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=7)
```

Inserção, divisão e promoção



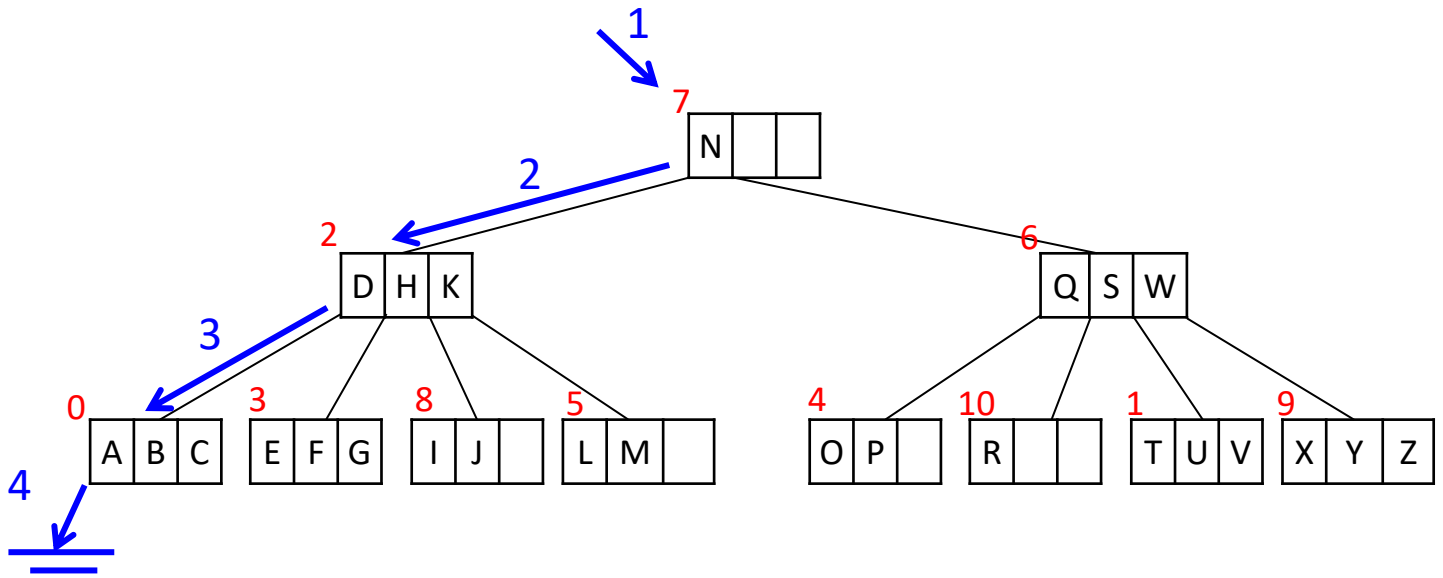

```
1: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=7)
                                                    ↓
2: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=2)
```

Inserção, divisão e promoção



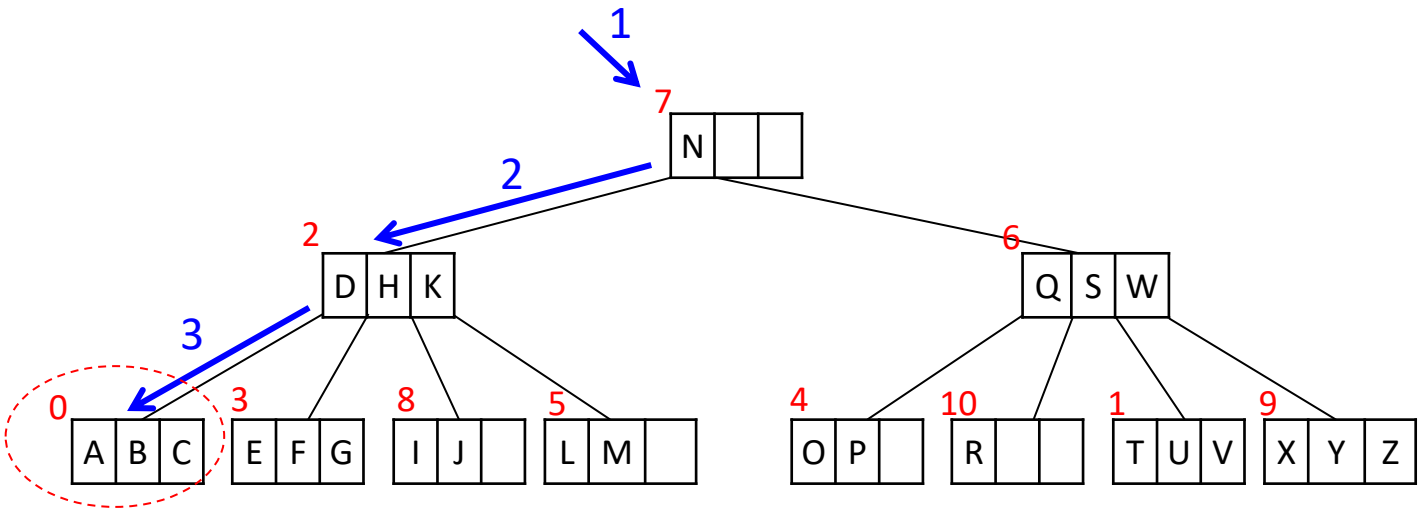

```
1: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=7)
                                                    ↓
2: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=2)
                                                    ↓
3: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=0)
```

Inserção, divisão e promoção



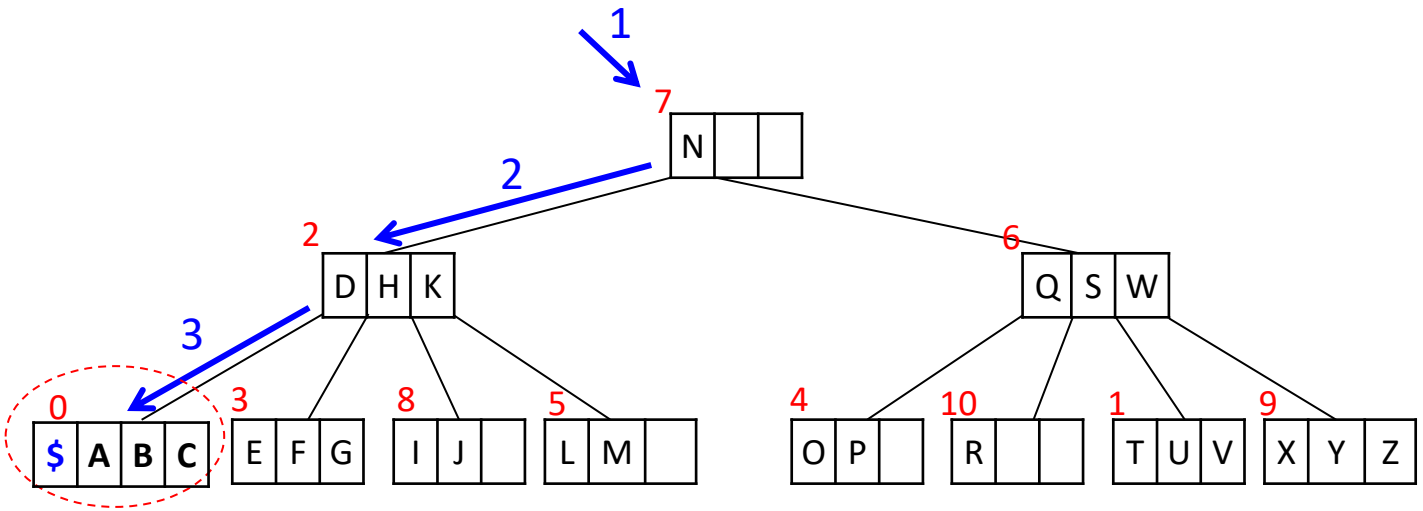

```
1: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=7)
                                                    ↓
2: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=2)
                                                    ↓
3: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=0)
                                                    ↓
4: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=NULO)
```

Inserção, divisão e promoção



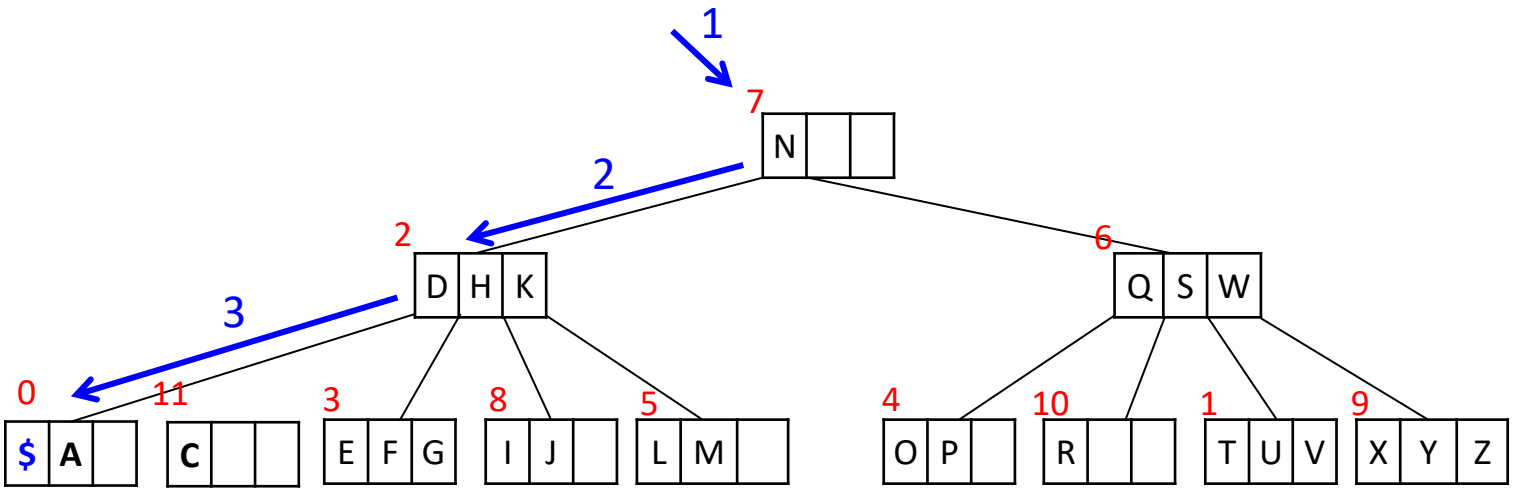

```
1: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=7)
                                      ↓
2: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=2)
                                      ↓
3: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=0)
                                      ↓
4: chavePro=$, filhoDpro=NULO, promo=Verdadeiro = insereNaArvore(chave=$, rrnAtual=NULO)
   ↑_____________________________________________________________________|
```

Inserção, divisão e promoção



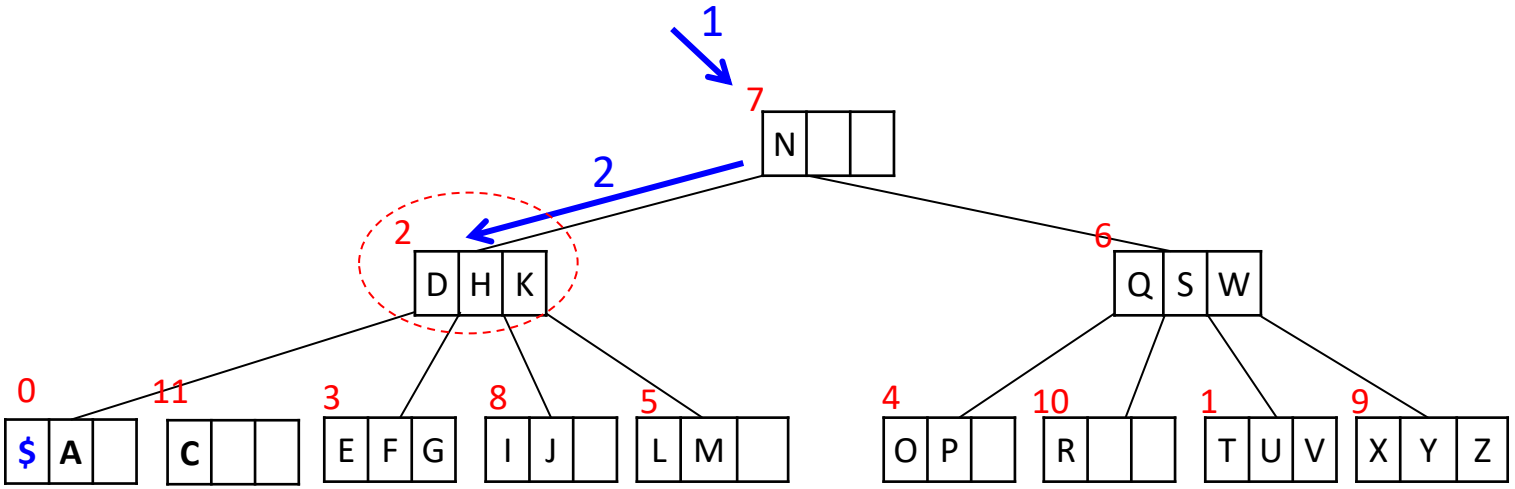

```
1: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=7)
                                      ↓
2: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=2)
                                      ↓
3: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=0)
                                      ↓
4: chavePro=$, filhoDpro=NULO, promo=Verdadeiro = insereNaArvore(chave=$, rrnAtual=NULO)
   Divisão + Promoção
```

Inserção, divisão e promoção



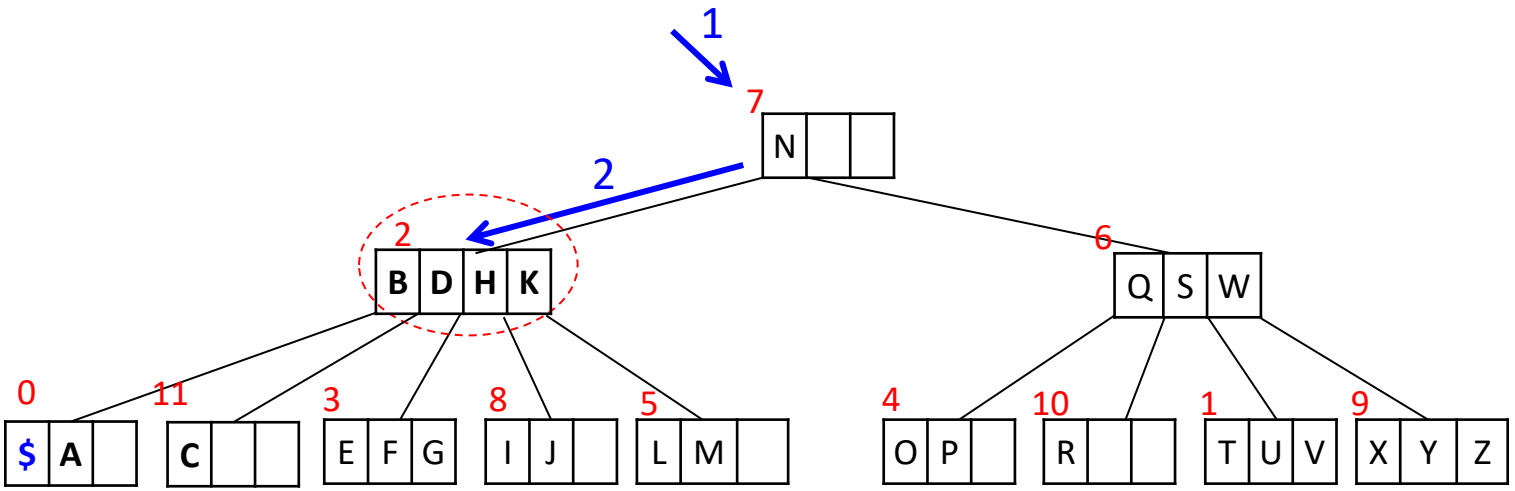

```
1: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=7)
                                      ↓
2: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=2)
                                      ↓
3: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=0)
                                      ↓
4: chavePro=$, filhoDpro=NULO, promo=Verdadeiro = insereNaArvore(chave=$, rrnAtual=NULO)
   Divisão + Promoção
```

Inserção, divisão e promoção



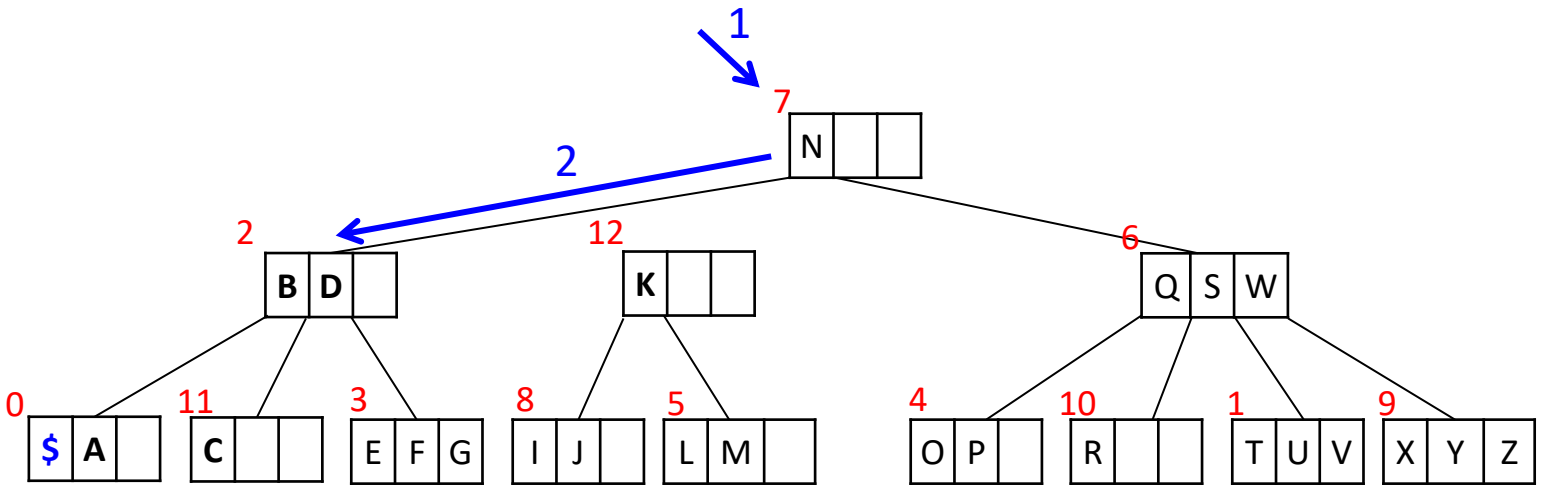

```
1: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=7)
2: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=2)
3: chavePro=B, filhoDpro=11, promo=Verdadeiro = insereNaArvore(chave=$, rrnAtual=0)
4: chavePro=$, filhoDpro=NULO, promo=Verdadeiro = insereNaArvore(chave=$, rrnAtual=NULO)
```

Inserção, divisão e promoção



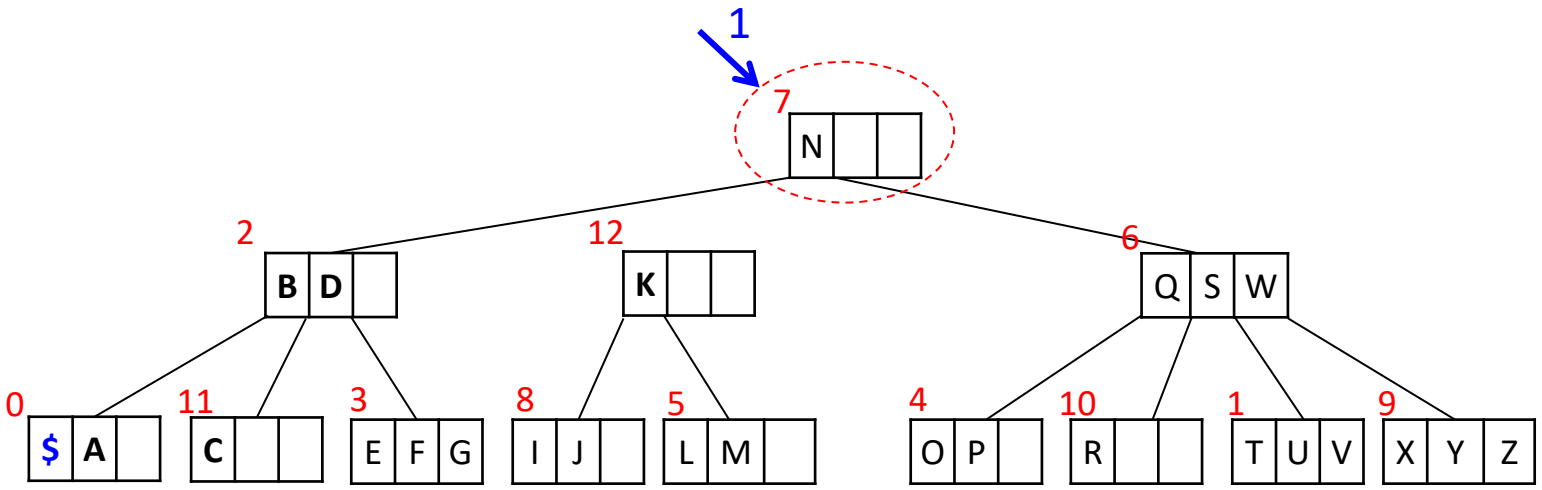

```
1: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=7)
                                      ↓
2: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=2)
                                      ↓
3: chavePro=B, filhoDpro=11, promo=Verdadeiro = insereNaArvore(chave=$, rrnAtual=0)
   Divisão + Promoção                 ↑                    ↓
4: chavePro=$, filhoDpro=NULO, promo=Verdadeiro = insereNaArvore(chave=$, rrnAtual=NULO)
```

Inserção, divisão e promoção



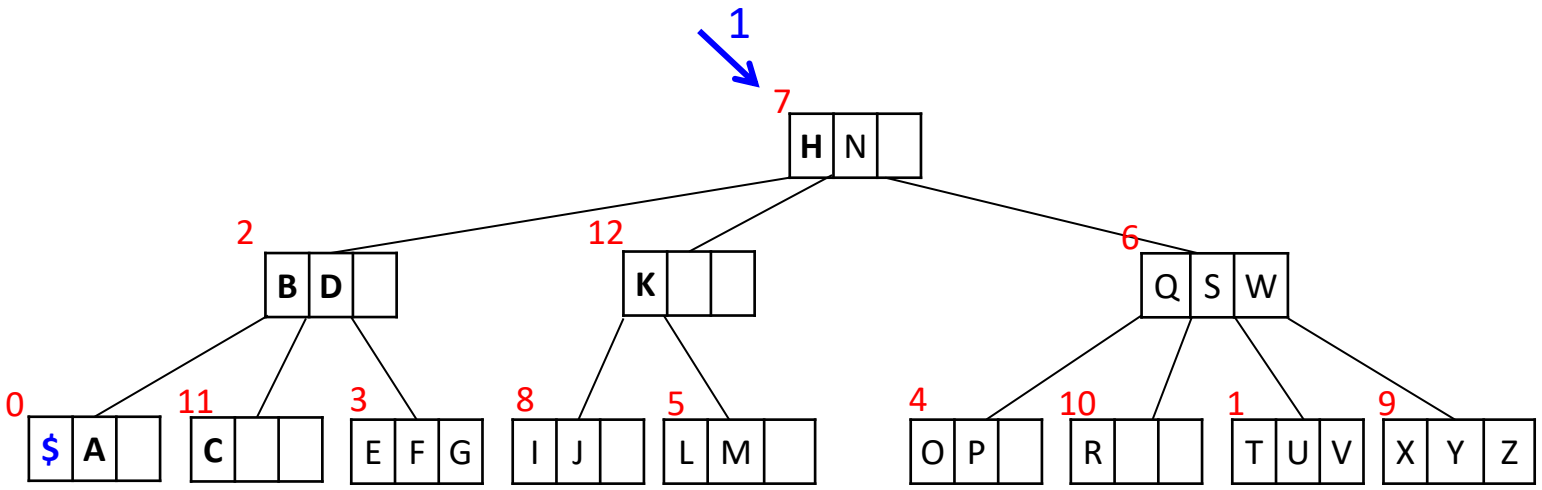

```
1: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=7)
                                      ↓
2: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=2)
                                      ↓
3: chavePro=B, filhoDpro=11, promo=Verdadeiro = insereNaArvore(chave=$, rrnAtual=0)
   Divisão + Promoção                 ↑                     ↓
4: chavePro=$, filhoDpro=NULO, promo=Verdadeiro = insereNaArvore(chave=$, rrnAtual=NULO)
```

Inserção, divisão e promoção




```
1: chavePro=?, filhoDpro=?, promo=? = insereNaArvore(chave=$, rrnAtual=7)
2: chavePro=H, filhoDpro=12, promo=Verdadeiro = insereNaArvore(chave=$, rrnAtual=2)
3: chavePro=B, filhoDpro=11, promo=Verdadeiro = insereNaArvore(chave=$, rrnAtual=0)
4: chavePro=$, filhoDpro=NULO, promo=Verdadeiro = insereNaArvore(chave=$, rrnAtual=NULO)
```

Inserção, divisão e promoção



1: chavePro=?, filhoDpro=?, promo=? = *insereNaArvore*(chave=$, rrnAtual=7)

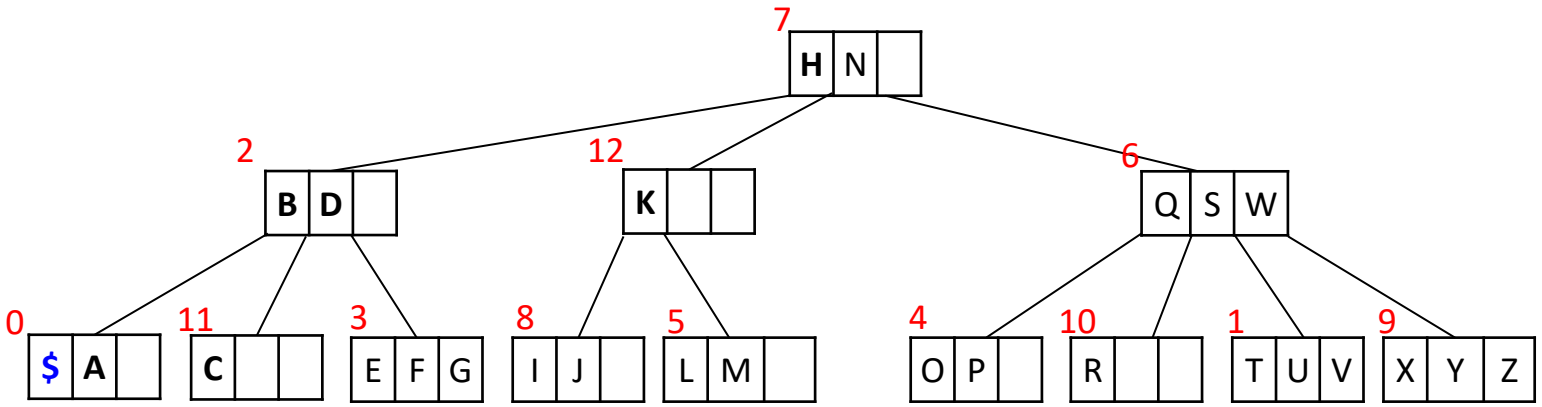
2: chavePro=H, filhoDpro=12, promo=Verdadeiro = *insereNaArvore*(chave=$, rrnAtual=2)

Inserção Simples

3: chavePro=B, filhoDpro=11, promo=Verdadeiro = *insereNaArvore*(chave=$, rrnAtual=0)

4: chavePro=$, filhoDpro=NULO, promo=Verdadeiro = *insereNaArvore*(chave=$, rrnAtual=NULO)

Inserção, divisão e promoção




```
1: chavePro=NULO, filhoDpro=NULO, promo=Falso = insereNaArvore(chave=$, rrnAtual=7)
2: chavePro=H, filhoDpro=12, promo=Verdadeiro = insereNaArvore(chave=$, rrnAtual=2)
3: chavePro=B, filhoDpro=11, promo=Verdadeiro = insereNaArvore(chave=$, rrnAtual=0)
4: chavePro=$, filhoDpro=NULO, promo=Verdadeiro = insereNaArvore(chave=$, rrnAtual=NULO)
```

Inserção, divisão e promoção

- A função *insere* utiliza várias funções auxiliares:
  - `lePagina(rrn)`
  - `escrevePagina(rrn, pag)`
  - `buscaNaPagina(chave, pag)`
  - `insereNaPagina(chave, filhoD, pag)`
  - `divide(chave, filhoD, pag)`

Inserção, divisão e promoção

```
FUNÇÃO lePagina (rrn)
   calcule o byte-offset da página a partir do rrn
   faça seek no arquivo árvore-B para o byte-offset calculado
   leia pag do arquivo árvore-B
   retorne pag
fim FUNÇÃO

FUNÇÃO escrevePagina (rrn, pag)
   calcule o byte-offset da página a partir do rrn
   faça seek no arquivo árvore-B para o byte-offset calculado
   escreva pag no arquivo árvore-B
fim FUNÇÃO
```

Note que a variável **pag** é um objeto da classe **Pagina**. No arquivo, uma página será um **registro de tamanho fixo com campos de tamanho fixo**. Dessa forma, as representações de uma página em memória RAM e em arquivo são diferentes. As funções de *lePagina* e *escrevePagina* devem cuidar para que essa "conversão" ocorra de forma adequada.

Inserção, divisão e promoção

```
FUNÇÃO insereNaPagina(chave, filhoD, pag)
    se pag estiver cheia, aumente a sua capacidade
    faça i receber pag.numChaves
    enquanto i > 0 e chave < pag.chaves[i-1] faça
        pag.chaves[i] = pag.chaves[i-1]
        pag.filhos[i+1] =  pag.filhos[i]
        decremente i
    faça pag.chaves[i] receber chave
    faça pag.filhos[i+1] receber filhoD
    incremente pag.numChaves
fim FUNÇÃO
```

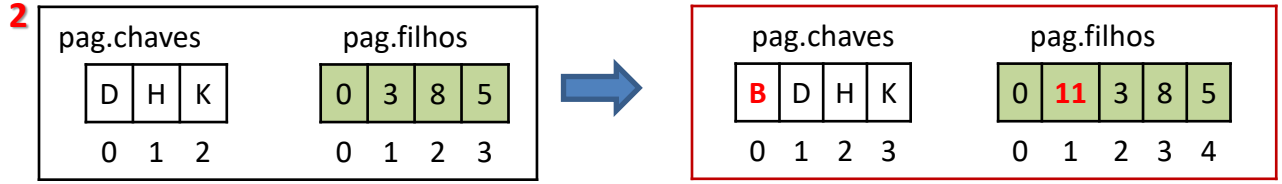
Quando a página estiver cheia, adicione um elemento **NULO** à lista **chaves** e à lista **filhos**

Inserção, divisão e promoção

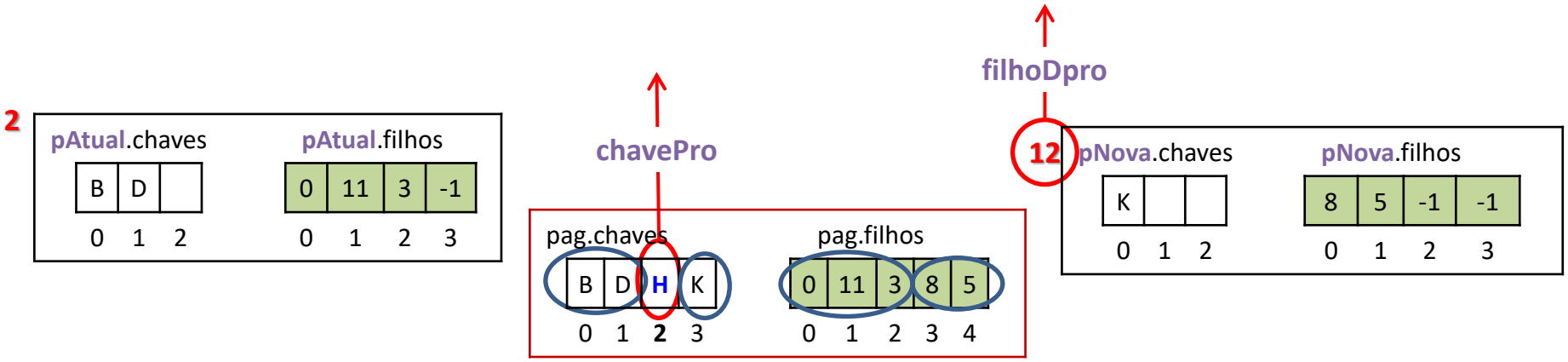
- Função ***divide***
  - Cria uma nova página (**pNova**)
  - Distribui as chaves entre a página atual (**pAtual**) e a nova página (**pNova**)
  - Determina qual chave (**chavePro**) e qual RRN (**filhoDpro**) promover
    - **chavePro** (chave promovida) → sempre é a chave mediana da página
      - Como a página tem um tamanho fixo, a chave mediana sempre estará na mesma posição da lista de chaves
    - **filhoDpro** (referência do filho direito) → sempre é o RRN da nova página
      - A nova página sempre é gravada no fim do arquivo → cálculo do RRN da nova página pode ser feito por uma função ***novo_rrn()***
  - Retorna **chavePro**, **filhoDpro**, **pAtual**, **pNova**

Inserção, divisão e promoção

- Exemplo: ***divide*** **(chave = 'B', filhoD = 11, pag = 2)**
  - Insira **chave** (**B**) e **filhoD** (**11**) na posição apropriada em **pag**



  - Divida o conteúdo de **pag** entre **pAtual** e **pNova**, exceto pela chave mediana (**H**), que será promovida juntamente com o RRN da **pNova** (**12**)



Inserção, divisão e promoção

```
FUNÇÃO divide (chave, filhoD, pag)
    insira chave e filhoD em pag  # usando a função insereNaPagina
    faça meio receber ORDEM // 2
    faça chavePro receber pag.chaves[meio]
    faça filhoDpro receber o RRN que a pNova terá no arquivo árvore-b
    faça pAtual receber o conteúdo de pag até meio
    faça pNova receber o conteúdo de pag a partir de meio+1
    retorne chavePro, filhoDpro, pAtual, pNova
```

Inserção, divisão e promoção

- Como saber qual RRN a **pNova** terá no arquivo árvore-b?
  - **Sempre que uma página é criada, ela é gravada no fim do arquivo**
  - As páginas têm tamanho fixo e conhecido → *TAM_PAG*

```
FUNÇÃO novo_rrn()
    faça seek para o fim do arquivo
    faça offset receber o byte-offset do fim do arquivo
    retorne (offset - TAM_CAB)//TAM_PAG
fim FUNÇÃO
```

Árvore-B

- As inserções devem ser feitas por uma função gerenciadora
- Tal função executa os seguintes processos:
  - Lê as chaves a serem armazenadas na árvore-B e chama a função *insereNaArvore()*
  - **Cria uma nova raiz quando houver divisão da raiz atual**
    - I.e., quando a função *insereNaArvore()* retornar **Verdadeiro** para a promoção
    - Cria a página que será a nova raiz, inserindo a chave promovida e atualizando seus filhos
    - Atualiza o RRN da raiz

Árvore-B

```
FUNÇÃO gerenciadorDeInsercao(raiz)
    leia uma chave e armazene-a em chave
    enquanto existirem chaves a serem inseridas faça
       chavePro, filhoDpro, promoção = insereNaArvore(raiz, chave)
       se promoção então
          inicialize pNova
          pNova.chaves[0] = chavePro          # nova chave raiz
          pNova.filhos[0] = raiz              # filho esquerdo
          pNova.filhos[1] = filhoDpro         # filho direito
          incremente pNova.numChaves
          escreva pNova no arquivo da árvore-b
          faça raiz receber o RRN de pNova    # RRN da nova página raiz
       fim se
       leia a próxima chave e armazene-a em chave
    fim enquanto
    retorne raiz
fim FUNÇÃO
```

Árvore-B

- A função principal fica responsável por abrir (ou criar) o arquivo da árvore-B e chamar o gerenciador

```
FUNÇÃO principal()
    se o arquivo árvore-B existe então
        abra-o para leitura e escrita como arqArvb
        leia o cabeçalho e armazene-o em raiz
    senão
        abra-o um novo arquivo para leitura e escrita como arqArvb
        faça raiz receber 0 e a escreva no cabeçalho de arqArvb
        inicialize pag e a escreva no arquivo arqArvb
    fim se
    raiz = gerenciadorDeInsercao(raiz)
    escreva raiz no cabeçalho do arquivo arqArvb
    feche o arquivo arqArvb
fim FUNÇÃO
```