Árvores

ÁRVORES

- Data object muito importante
- Estrutura organizada que representa relacionamento hierárquico
- Exemplos: ascendentes e descendentes
- Relacionamento em dois ramos ou multi-ramos

ÁRVORES

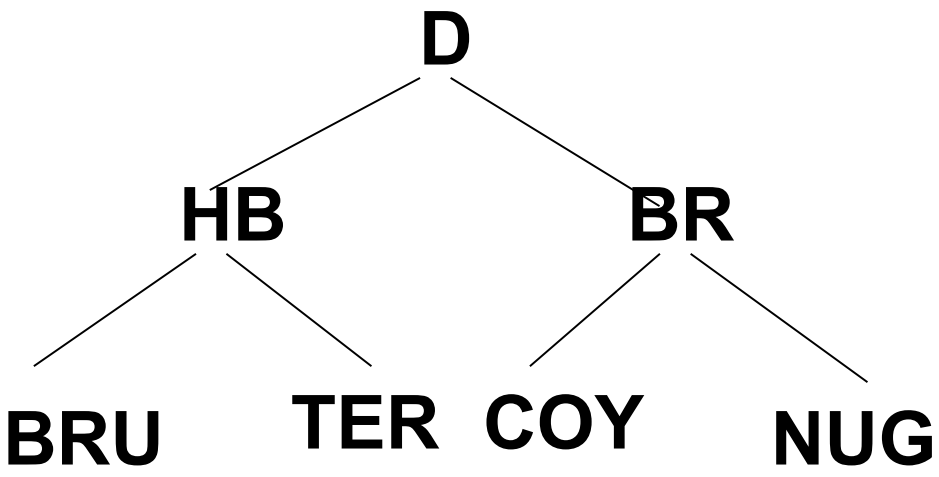
Definição: É um conjunto finito de um ou mais nós tal que:

(a) existe um nó especial chamado de raiz

(b) os nós restantes são particionados em $n \geq 0$ conjuntos disjuntos T_1, ..., T_n onde cada um destes conjuntos são árvores.

T_1, ..., T_n são chamados de sub-árvores

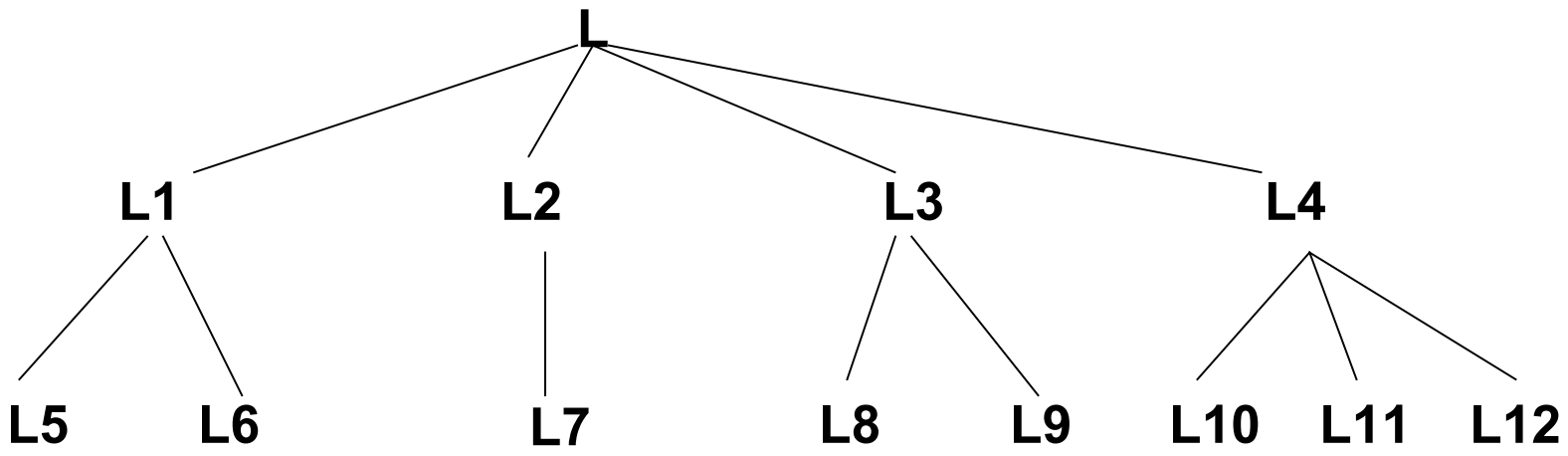
ÁRVORES



Duas subárvores com raízes HB e BR

A condição de conjuntos disjuntos de T_1, ..., T_n não permite a conexão entre sub-árvores

ÁRVORES

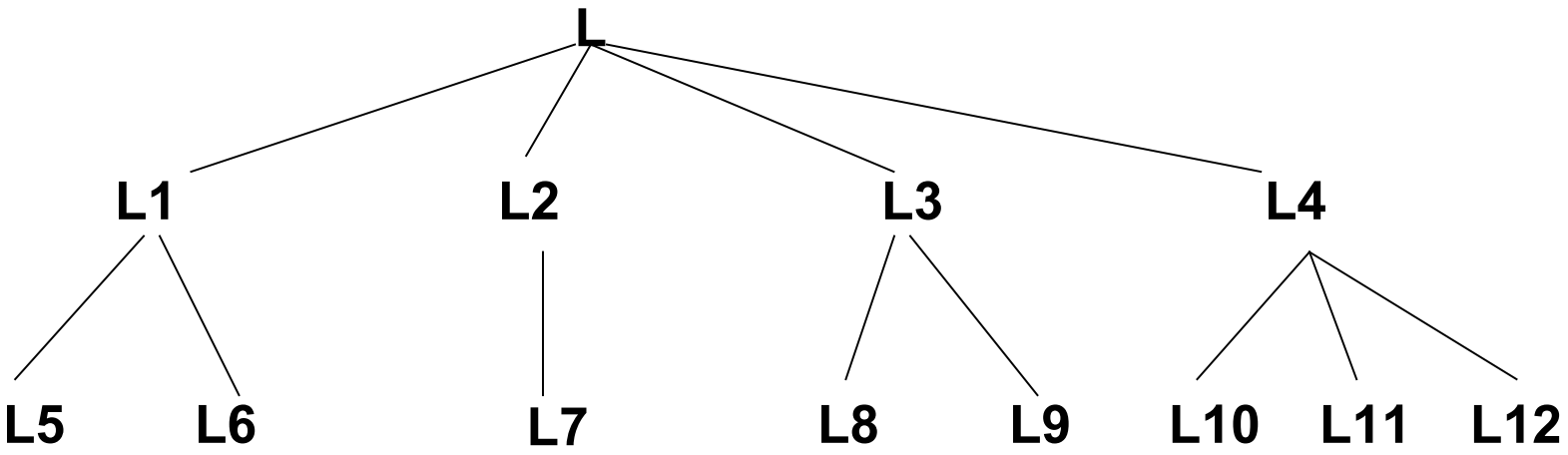


Quatro subárvores com raízes L1, L2, L3 e L4

Cada item é raiz de alguma subárvore. Por exemplo, L12 é uma raiz sem subárvores

ÁRVORES

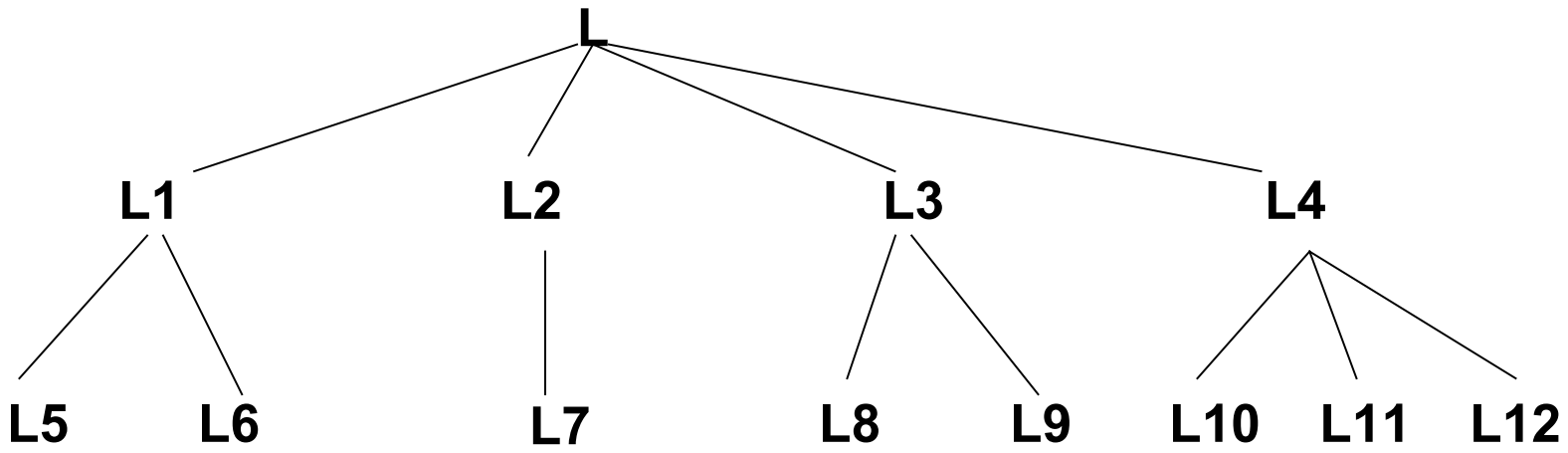
Nó = informação + ligações a outros itens



13 nós

ÁRVORES

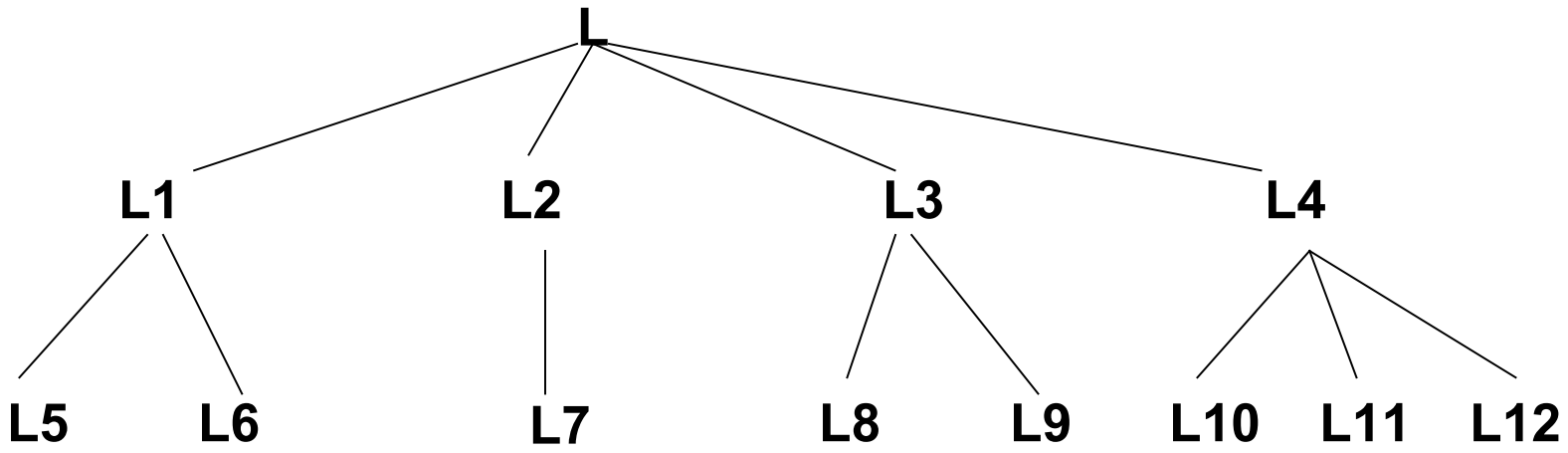
Grau = número de subárvores de um nó



L4 tem grau 3

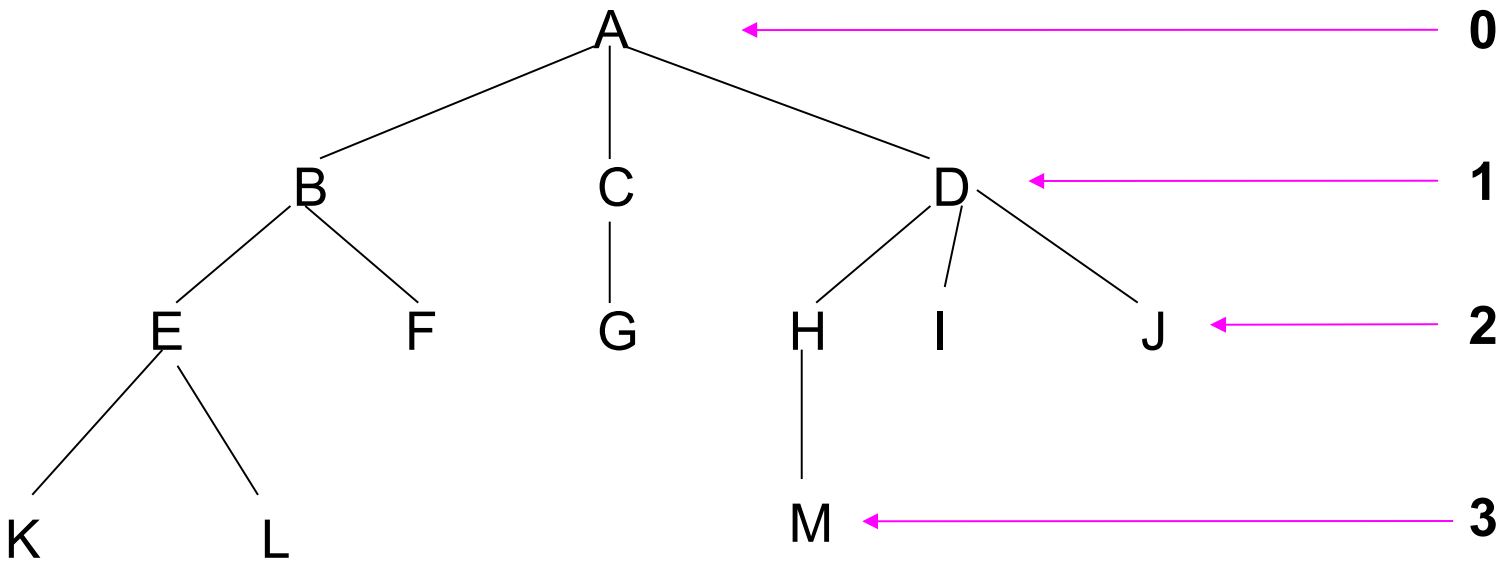
ÁRVORES

Folha = nó com grau 0



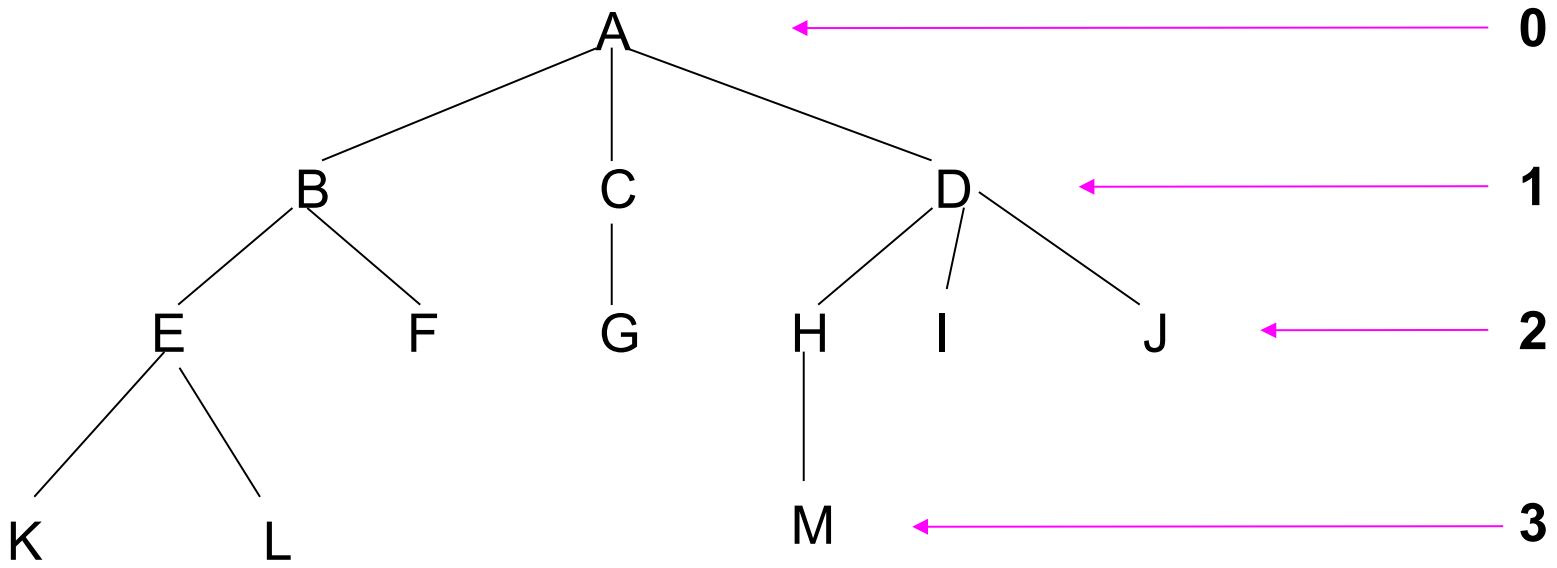
L5 a L12 (também *nós terminais*)

ÁRVORES



Grau de uma árvore = grau máximo dos nós da árvore → 3

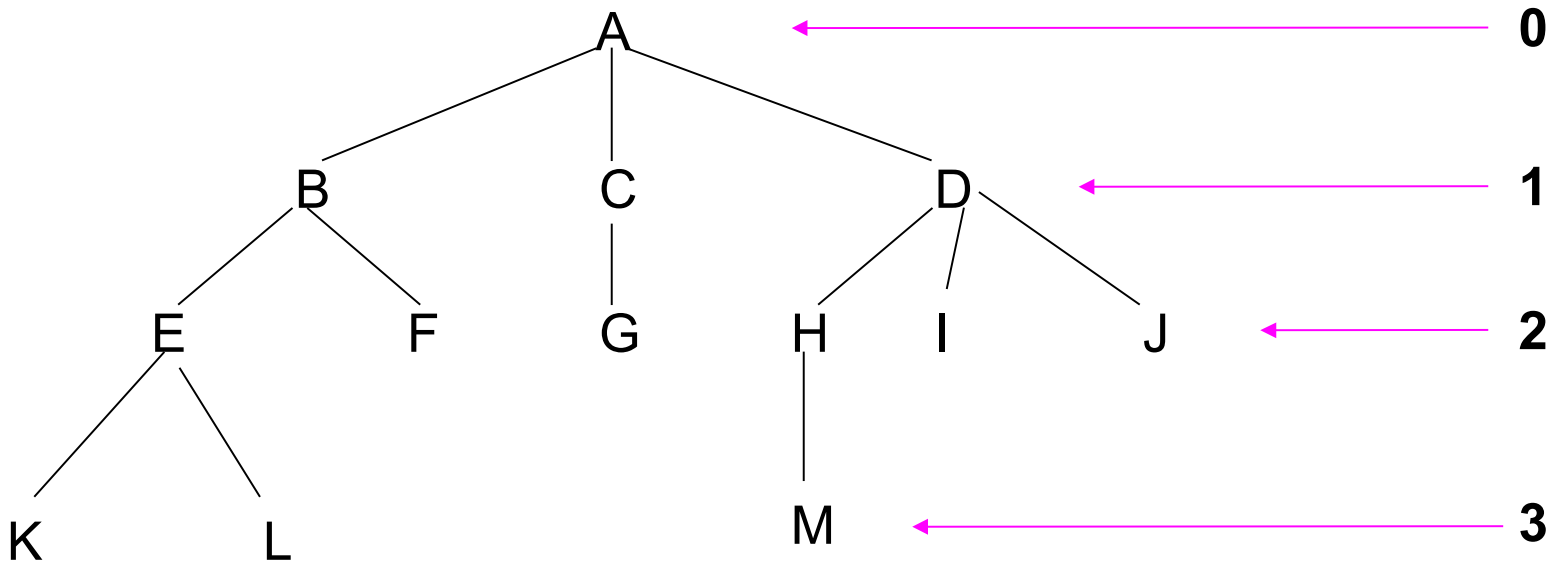
ÁRVORES



Ancestrais de um nó = todos os nós no caminho raiz-nó

→ os ancestrais de M são A, D e H

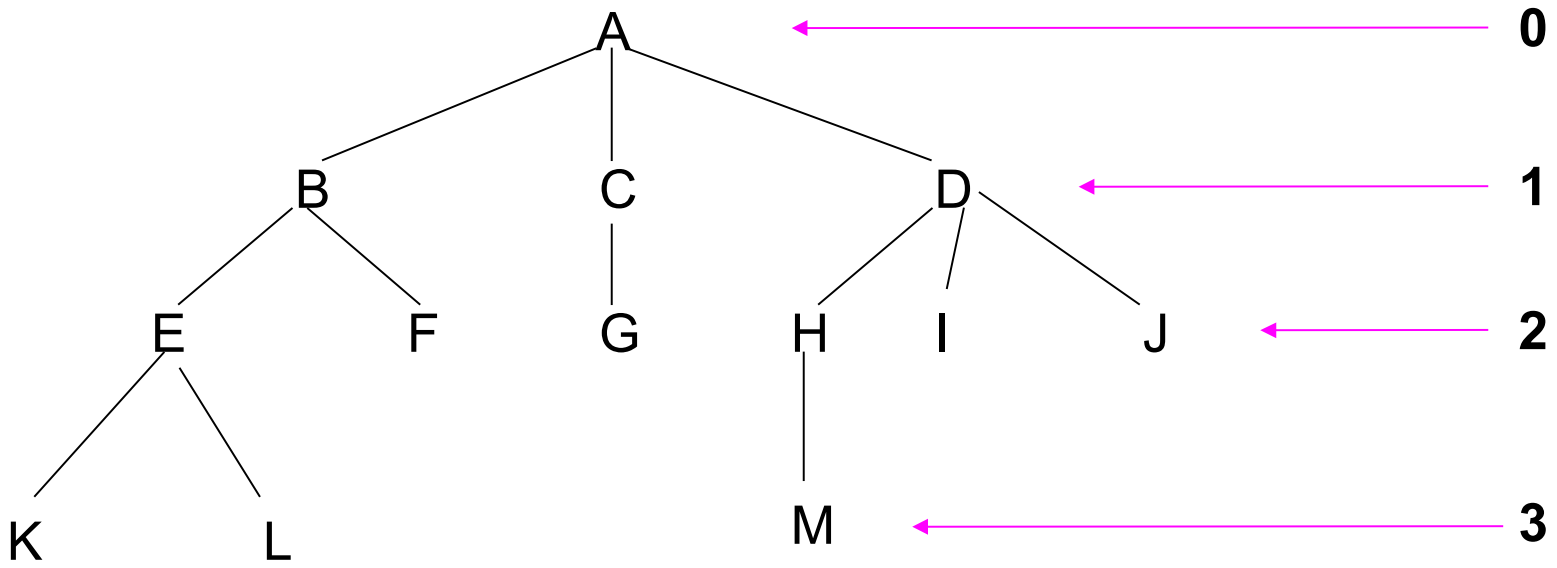
ÁRVORES



Nível = raiz sempre no nível 1.

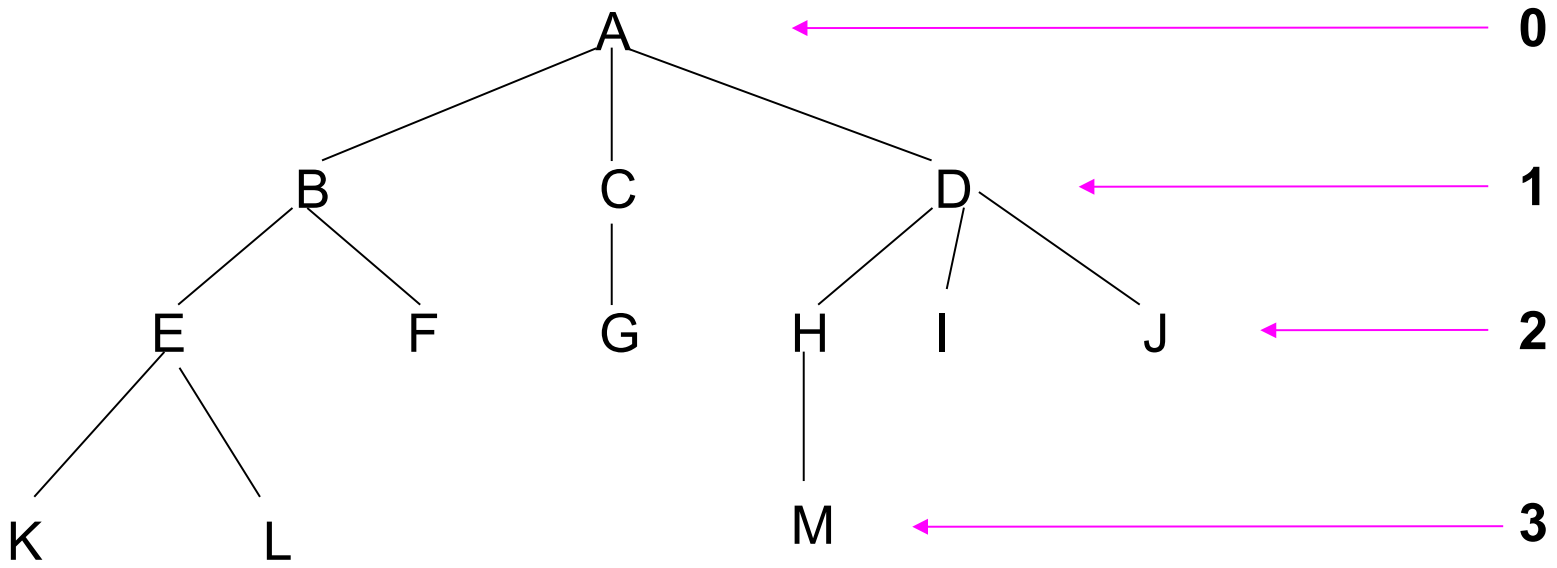
Nó no nível *l*, filhas no nível *l+1*

ÁRVORES



Profundidade = nível máximo de qualquer nó na árvore $\rightarrow$ 4

ÁRVORES



Floresta = n≥0 árvores disjuntas

↓

se eliminar o nó A

ÁRVORES BINÁRIAS

- É uma estrutura muito importante
- Qualquer nó pode ter no MÁXIMO dois ramos (grau ≤ 2)
- Distinção entre sub-árvore à esquerda e sub-árvore à direita
- Pode ter 0 (zero) nós
- É um objeto bem diferente do objeto árvore

ÁRVORES BINÁRIAS

Definição: Uma árvore binária é um conjunto finito de nós que ou é vazia ou consiste de uma raiz e duas árvores binárias disjuntas chamadas de *subárvore esquerda* e *subárvore direita*

Número máximo de nós num nível *l* = 2^{l-1}, para $l \geq 1$
Número máximo de nós com profundidade k = $2^k - 1$

OBS: às vezes alguns consideram nível de raiz como 0 (zero)

ÁRVORES BINÁRIAS

Processamento sistemático e ordenado de cada nó da árvore apenas uma vez para obter uma sequência linear dos nós.

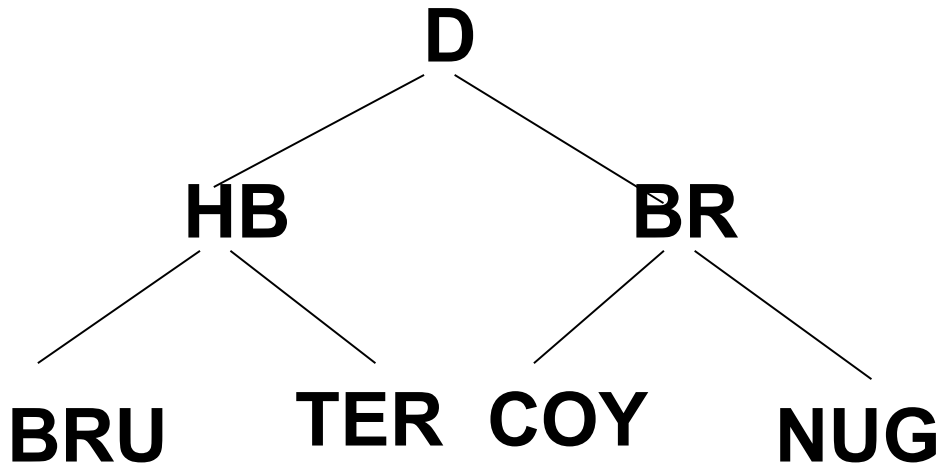
A sequência consiste de: *antecessor* e *sucessor*

Existem três maneiras de percorrer uma árvore binária

- Pre-Order (pré-ordem)
- In-Order (ordem simétrica)
- Post-Order (ordem final)

ÁRVORES BINÁRIAS

Pre-Order:



- Visita a raiz
- Percorre a sub-árvore à esquerda
- Percorre a sub-árvore à direita

D - HB - BRU - TER - BR - COY - NUG

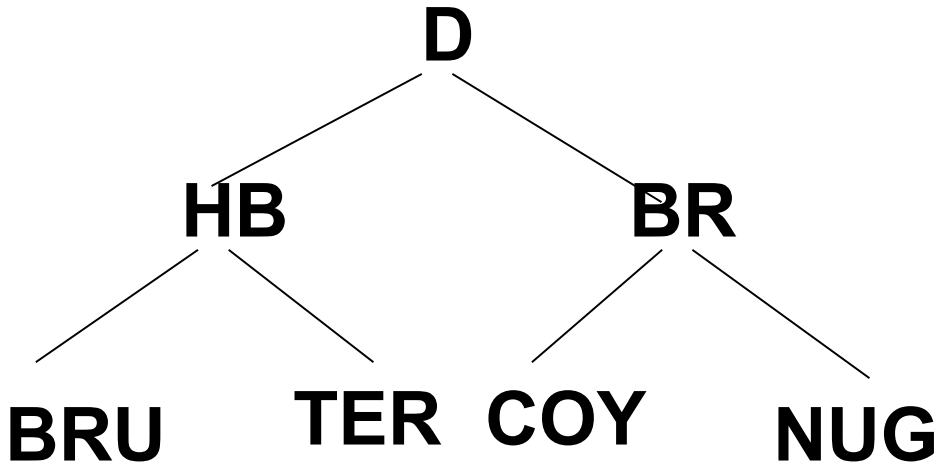
ÁRVORES BINÁRIAS

```
Pre-Order ( T )

// T é uma árvore binária com três campos:
// Lchild, Data, Rchild
{
    if T ≠ 0
    {
        print Data (T)
        Pre-Order ( Lchild (T) )
        Pre-Order ( Rchild (T) )
    }
}
```

ÁRVORES BINÁRIAS

In-Order:



- Percorre a sub-árvore à esquerda
- Visita a raiz
- Percorre a sub-árvore à direita

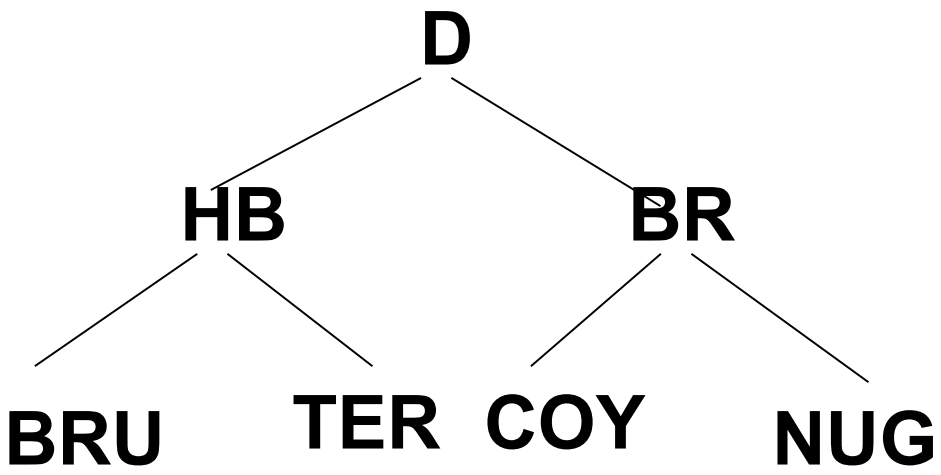
BRU - HB - TER - D - COY - BR - NUG

ÁRVORES BINÁRIAS

```
In-Order ( T )

// T é uma árvore binária com três campos:
// Lchild, Data, Rchild
{
    if T ≠ 0
    {
        In-Order ( Lchild (T) )
        print Data (T)
        In-Order ( Rchild (T) )
    }
}
```

ÁRVORES BINÁRIAS



Post-Order:

- Percorre a sub-árvore à esquerda
- Percorre a sub-árvore à direita
- Visita a raiz

BRU - TER - HB - COY - NUG - BR - D

ÁRVORES BINÁRIAS

```
Post-Order ( T )

// T é uma árvore binária com três campos:
// Lchild, Data, Rchild
{
    if T ≠ 0
    {
        Post-Order ( Lchild (T) )
        Post-Order ( Rchild (T) )
        print Data (T)
    }
}
```

ÁRVORES BINÁRIAS

Numa árvore binária com n nós, tem-se n+1 links para NULO

Seria interessante aproveitar esses links

Perlis/Thornton - alteração da representação original
→ COSTURA

- LinkEsq - endereço do nó antecessor
- LinkDir - endereço do nó sucessor

OBS: DEPENDE DA SEQÜÊNCIA LINEAR A SER UTILIZADA

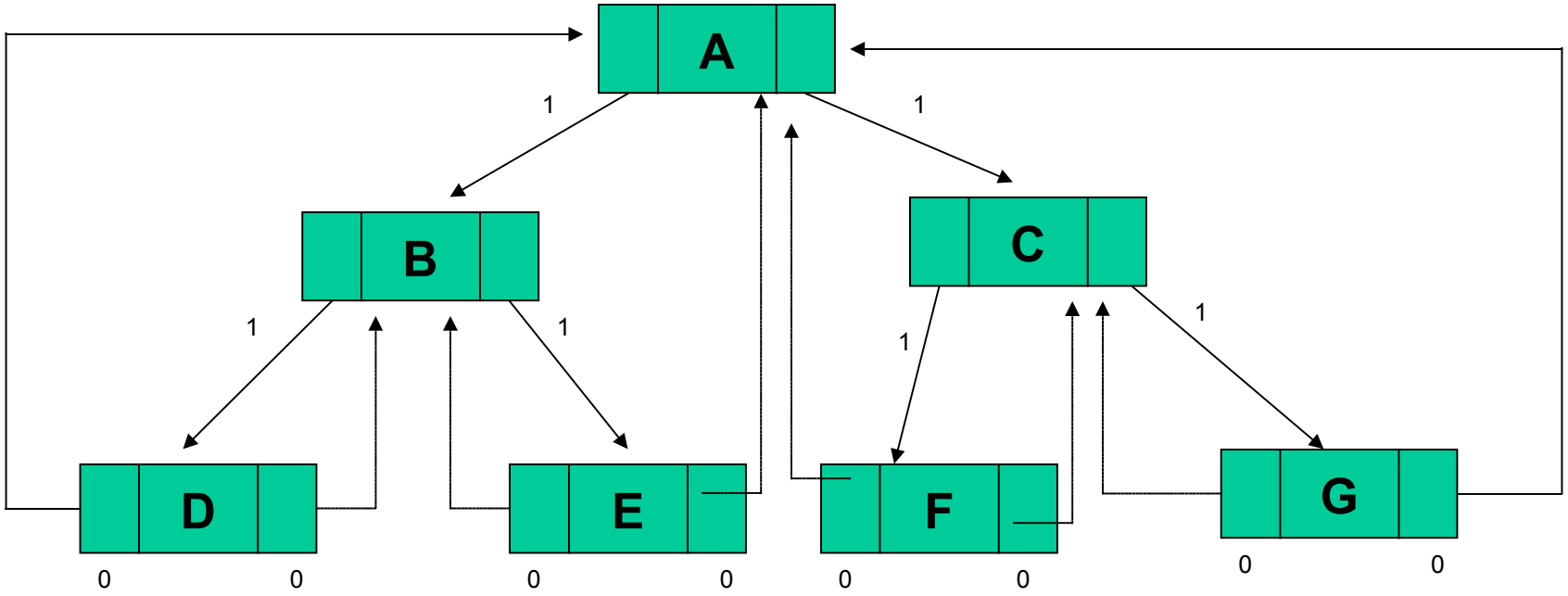
ÁRVORES BINÁRIAS

Determinar se o Link é VERDADEIRO ou é devido a COSTURA

Informação correspondente tem que ser embutida

Se LinkEsq ≠ NULO, marcar VERDADEIRO
Se LinkDir ≠ NULO, marcar VERDADEIRO
Se LinkEsq == NULO
{
    marcar FALSO
    aponta para antecessor
}
Se LinkDir == NULO
{
    marcar FALSO
    aponta para sucessor
}

ÁRVORES BINÁRIAS



In-Order: *D - B - E - A - F - C - G*

ÁRVORES BINÁRIAS

Tratamento especial para

PRIMEIRO NÓ	SEGUNDO NÓ
Sem antecessor	Sem sucessor

APONTA PRA RAIZ OU NÓ CABEÇA