Algoritmos e Estruturas de Dados I

Árvores

versão 2.3

Fabiano Oliveira
fabiano.oliveira@ime.uerj.br

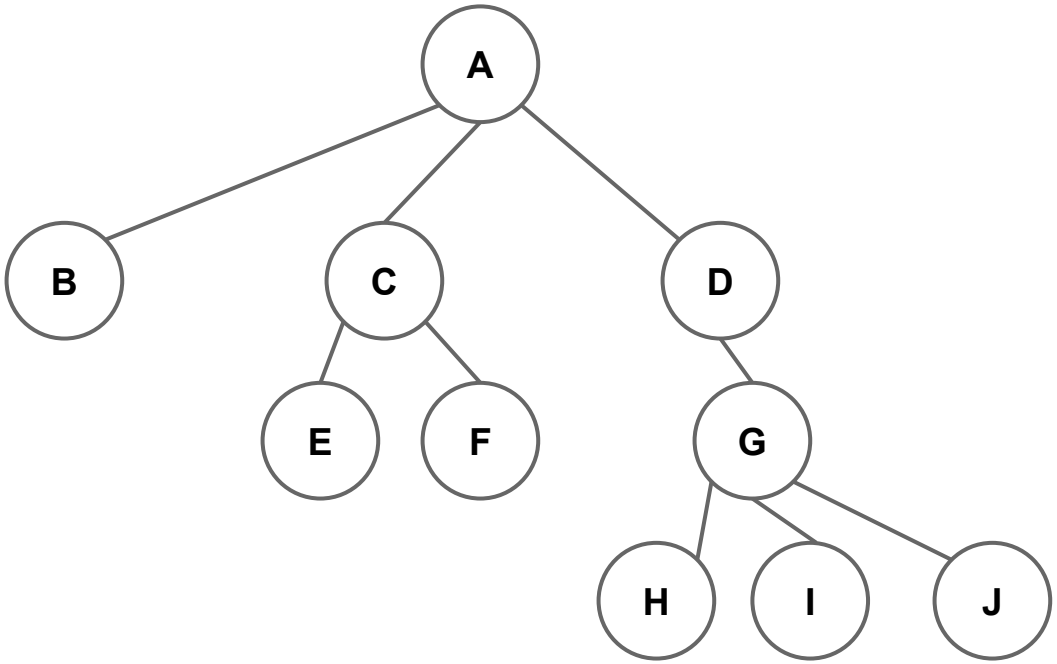
Árvores

- Uma ***árvore*** T é uma estrutura:
  - vazia, denotada por T = ∅, ou
  - composta por:
    - um ***nó*** R chamado de ***nó raiz***
    - 0 ou mais árvores disjuntas T_1, T_2, … associadas a R; tais árvores são chamadas de ***subárvores*** de R

- Um conjunto de árvores é chamado de ***floresta***

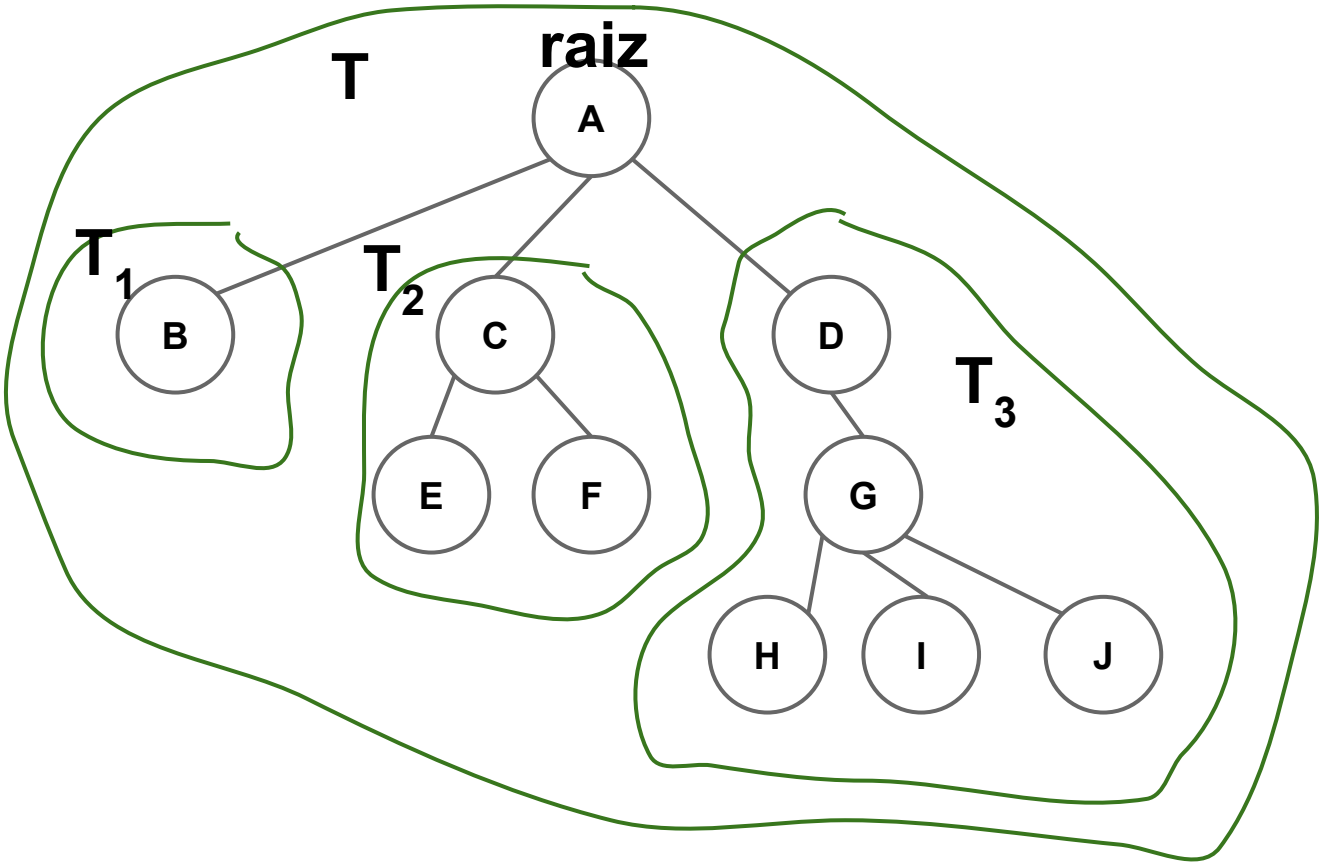
Árvores

- Forma usual de se representar uma árvore:



Árvores

- Forma usual de se representar uma árvore:

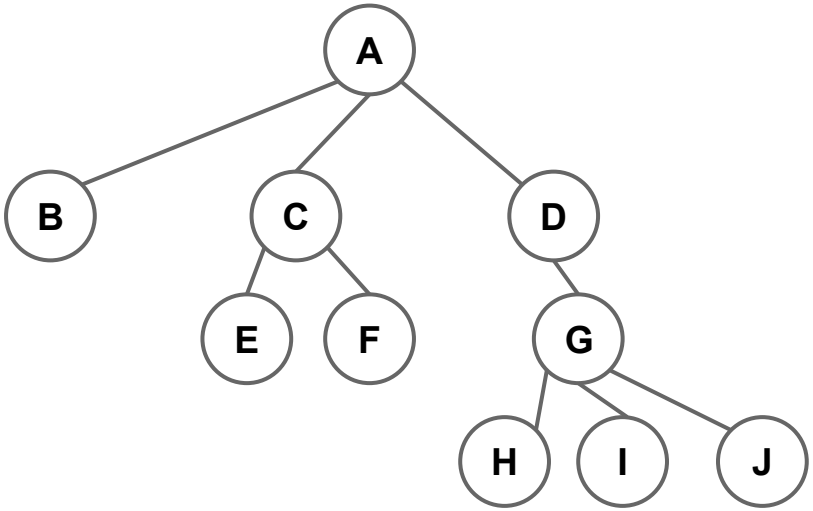


Árvores

- Nomenclatura relacionada a árvores:
  - Se T é uma subárvore de uma raiz R, então:
    - os ***nós de T*** são todas as raízes de subárvores de R (e também as raízes de subárvores de subárvores, etc.), além da raiz de T
    - um nó com 0 subárvores é chamado de ***folha***
    - a raiz de T é um ***nó filho*** de R e R é ***pai*** da raiz de T
    - R é ***ancestral*** a todos os nós de T
    - Todos os nós de T são ***descendentes*** de R

Árvores

- Nomenclatura relacionada a árvores:
  - Uma sequência $v_0,...,v_k$ de nós de uma árvore T tal que dois nós consecutivos pertencem à relação "é filho de" ou "é pai de" é chamada de ***caminho*** de tamanho k de T



Exemplos:

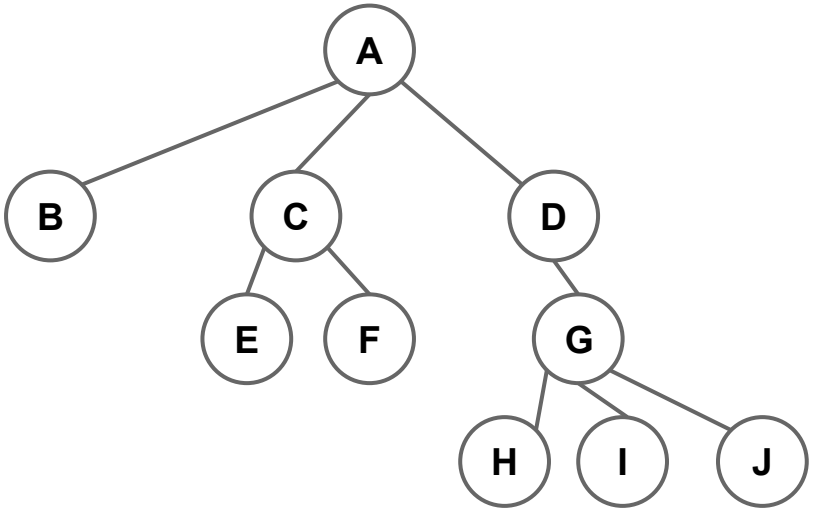
E,C,A

D,G,I

C,A,D,G

Árvores

- Nomenclatura relacionada a árvores:
  - O ***nível*** de um nó P de uma árvore T é o número de vértices do caminho que vai de P até a raiz de T



Exemplos:

nível de A = 1

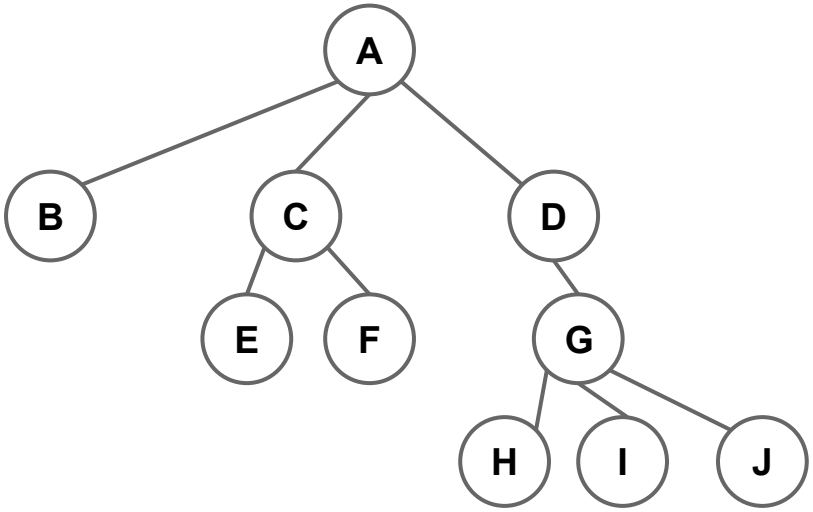
nível de B, C, D = 2

nível de E, F, G = 3

nível de H, I, J = 4

Árvores

- Nomenclatura relacionada a árvores:
  - A ***altura h(P)*** de um nó P é o número de nós do maior caminho que vai de P até uma folha descendente de P
  - A ***altura h(T)*** de uma árvore T é a altura de sua raiz



Exemplos:

h(B) = 1

h(C) = 2

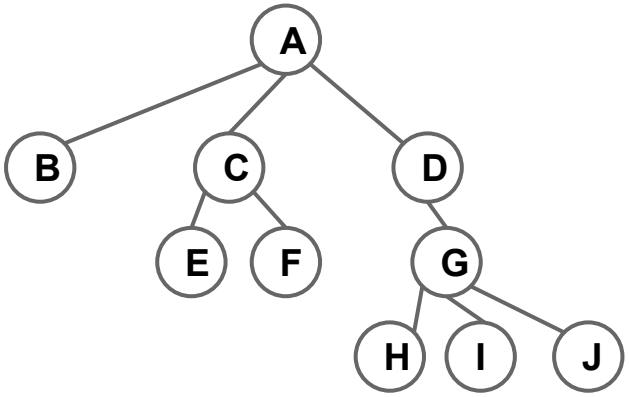
h(D) = 3

h(A) = 4

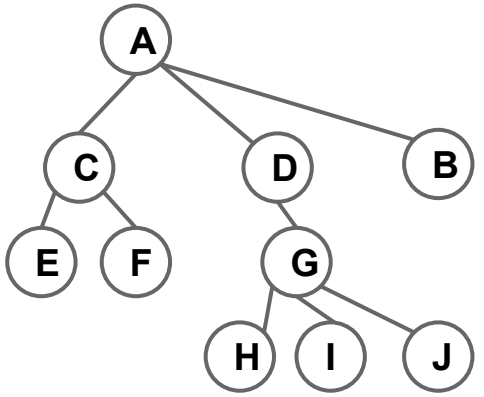
h(T) = h(A) = 4

Árvores

- Nomenclatura relacionada a árvores:
  - Uma árvore T é ***ordenada*** se há uma ordem definida entre as subárvores associadas à raiz de T

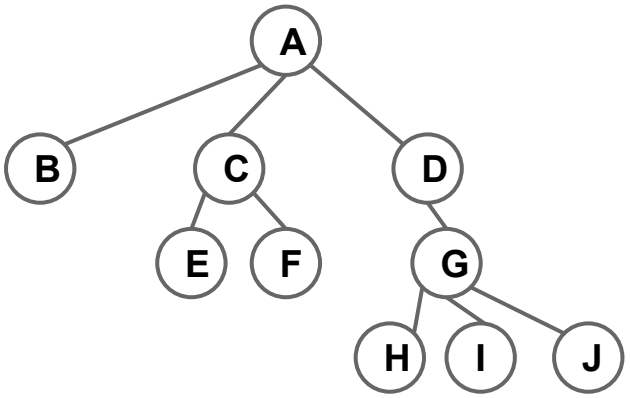


Se as árvores ao lado são consideradas iguais, então são **não ordenadas**. Caso contrário, são **ordenadas**.



Árvores

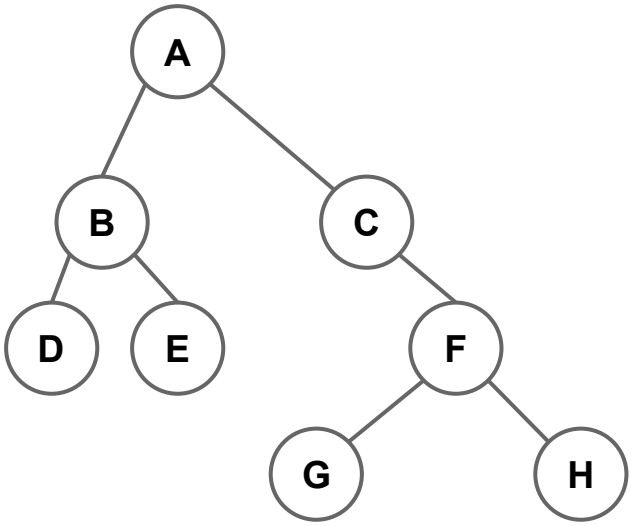
- Uma árvore ordenada é dita ***m-ária*** se cada nó possui m subárvores (binária = 2-ária, ternária = 3-ária, quaternária = 4-ária, etc.), algumas podendo ser vazias



Exemplo de árvore ternária (e também quaternária, 5-ária, 6-ária, etc.)

Árvores

- Numa árvore binária ordenada de raiz R, a primeira subárvore de cada nó é chamada de subárvore à esquerda de R (e sua raiz de filho esquerdo de R) e a segunda subárvore de subárvore à direita de R (e sua raiz de filho direito de R)

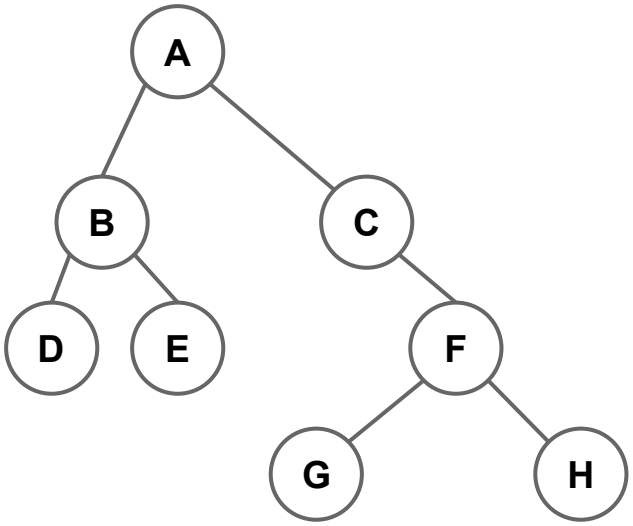


Exemplos:

- B é filho esquerdo e C é filho direito de A
- F é filho direito de C, que não possui filho esquerdo
- G não possui nem filho esquerdo nem filho direito

Árvores

- Uma árvore ***estritamente m-ária*** é aquela na qual cada nó possui 0 ou m filhos

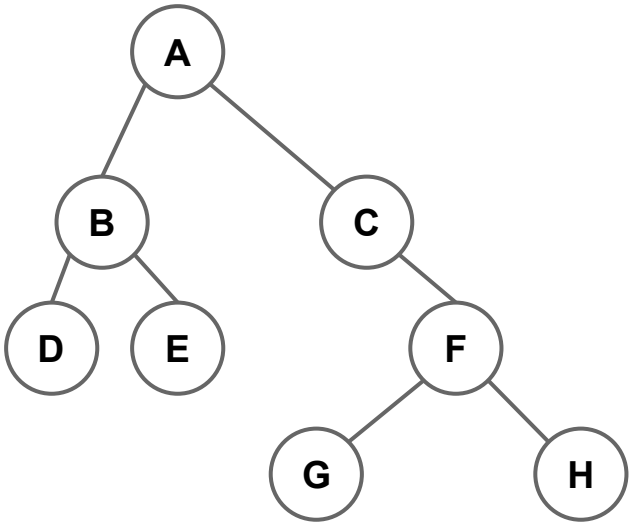


Exemplos:

- A árvore ao lado **não é** estritamente binária
- Se um nó for adicionado como filho de C, a árvore se torna estritamente binária

Árvores

- Uma árvore m-ária ***completa*** é aquela na qual todo nó com alguma subárvore vazia está no último ou penúltimo níveis

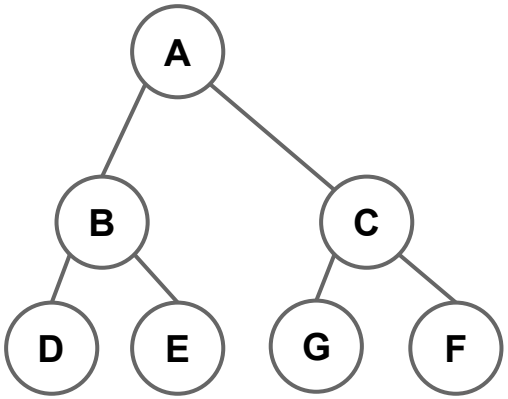


Exemplos:

- A árvore ao lado **não é** completa (C está no antepenúltimo nível)
- Se um nó for adicionado como filho de C, a árvore se torna completa

Árvores

- Uma árvore m-ária ***cheia*** é aquela na qual todo nó com alguma subárvore vazia está no último nível



Exemplo de uma árvore binária cheia

Árvores

- **Para pensar**:
  - Qual a altura **máxima** de uma árvore binária com n nós?
  - Qual a altura **máxima** de uma árvore estritamente binária com n nós?
  - Qual a altura **mínima** de uma árvore binária com n nós?
  - Numa árvore binária cheia com n nós, qual o número de nós no último nível?

Árvores

- **Para pensar**:
  - Qual a altura **máxima** de uma árvore binária com n nós? **n**
  - Qual a altura **máxima** de uma árvore estritamente binária com n nós? **(n+1)/2**
  - Qual a altura **mínima** de uma árvore binária com n nós? **⌈lg(n+1)⌉**
  - Numa árvore binária cheia com n nós, qual o número de nós no último nível? **(n+1)/2**

Árvores

- ***Como*** implementar uma árvore m-ária?

  - Alocação de nós:
    - sequencial
    - encadeada
      - Agrupamento de filhos:
        - sequencial
        - encadeada

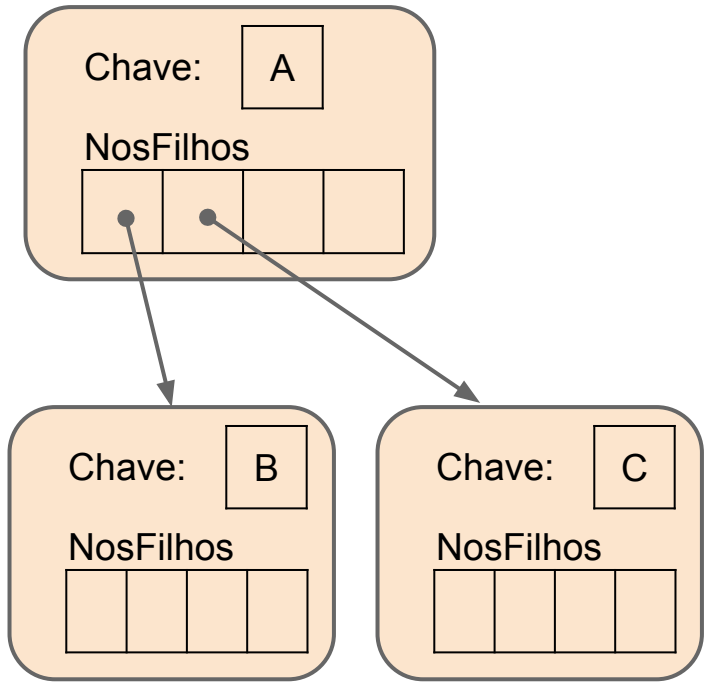
Árvores

- ***Como*** implementar uma árvore m-ária?
  - Alocação encadeada de nós

Espaço:
$\theta(mN)$
$O(N^2)$

```
//opção 1: alocação sequencial dos filhos
var M: Inteiro ← <valor de m>
estrutura No<TChave, TElem>:
    Chave: <TChave>
    Elem: <TElem>
    NosFilhos[1..M]: ^No
estrutura Arvore<TChave, TElem>:
    Raiz: ^No<TChave, TElem>
    m: Inteiro
procedimento Constroi(ref T: Arvore,
                      m: Inteiro)
    T.Raiz, T.m ← NULO, m

var T: Arvore<Caracter, ^Elemento>
Constroi(T, M)
```



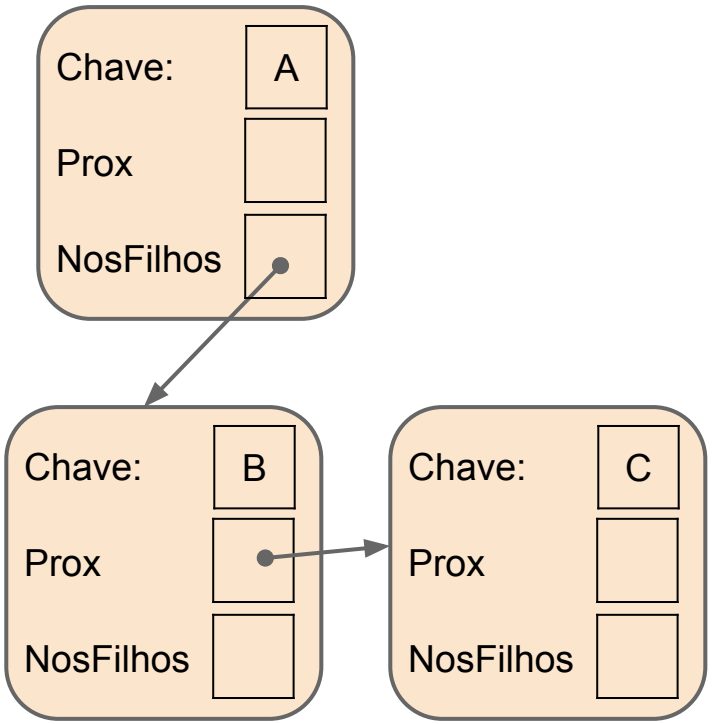
Árvores

- ***Como*** implementar uma árvore m-ária?
  - Alocação encadeada de nós

Espaço:
θ(N)

```
//opção 2:  alocação encadeada dos filhos
var M: Inteiro ← <valor de m>
estrutura No<TChave, TElem>:
    Chave: <TChave>
    Elem: <TElem>
    Prox: ^No, NosFilhos: ^No
estrutura Arvore<TChave, TElem>:
    Raiz: ^No<TChave, TElem>
    m: Inteiro
procedimento Constroi(ref T: Arvore,
                      m: Inteiro)
    T.Raiz, T.m ← NULO, m

var T: Arvore<Caracter, ^Elemento>
Constroi(T, M)
```



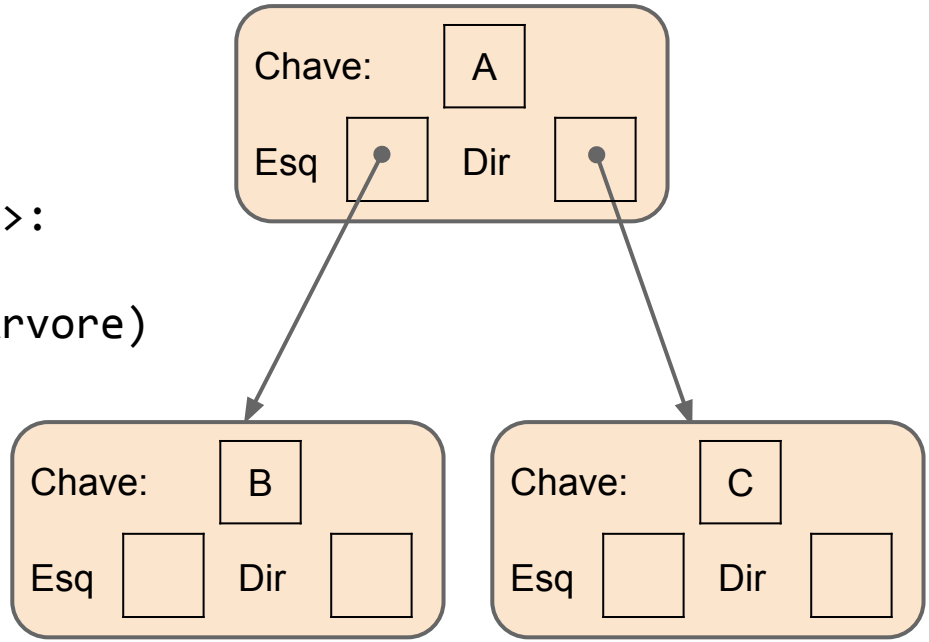
Árvores

- ***Como*** implementar uma árvore m-ária?
  - Alocação encadeada de nós

Espaço:
θ(N)

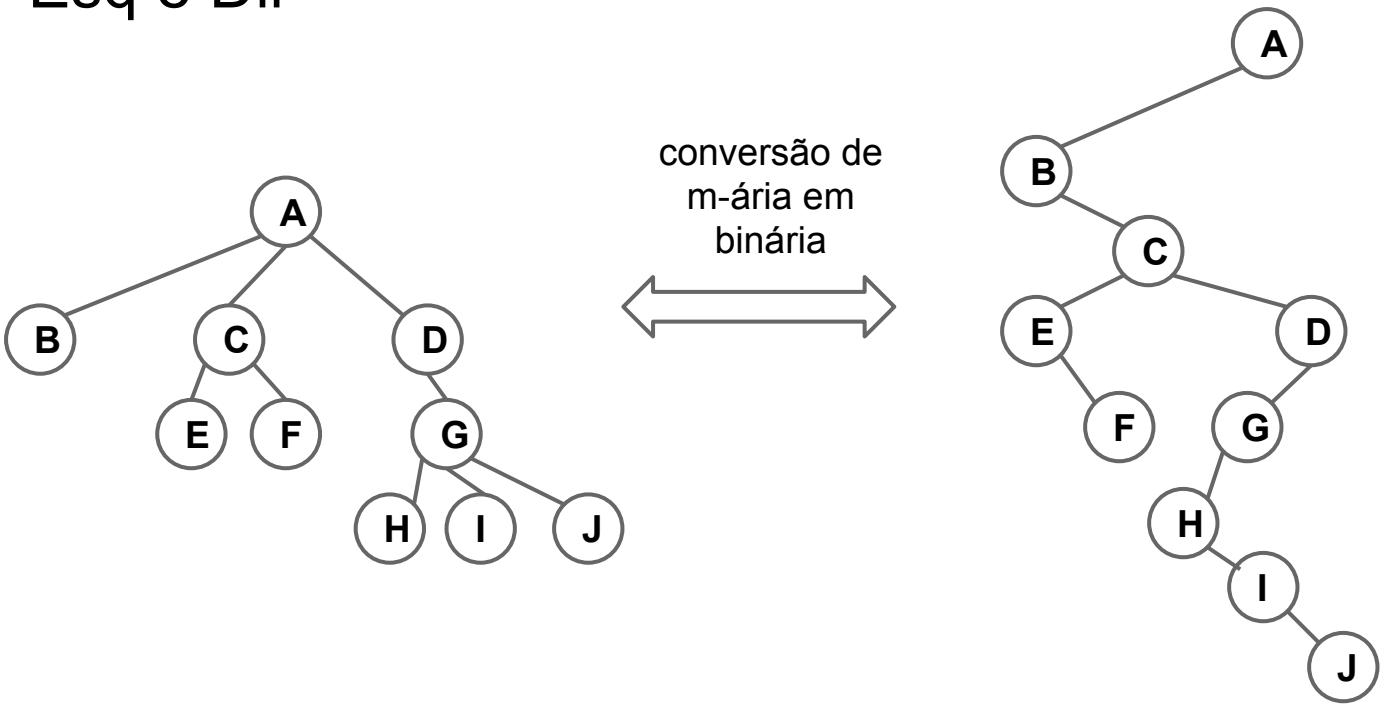
```
//opção usual quando árvore é binária
estrutura No<TChave, TElem>:
    Chave: <TChave>
    Elem: <TElem>
    Esq: ^No, Dir: ^No
estrutura Arvore<TChave, TElem>:
    Raiz: ^No<TChave, TElem>
procedimento Constroi(ref T: Arvore)
    T.Raiz ← NULO

var T: Arvore<Caracter,
                ^Elemento>
Constroi(T)
```



Árvores

- Note que, nas duas implementações anteriores, não há diferenças no estrutura No a menos de uma rerotulação dos campos NosFilhos e Prox para, respectivamente, Esq e Dir



Árvores

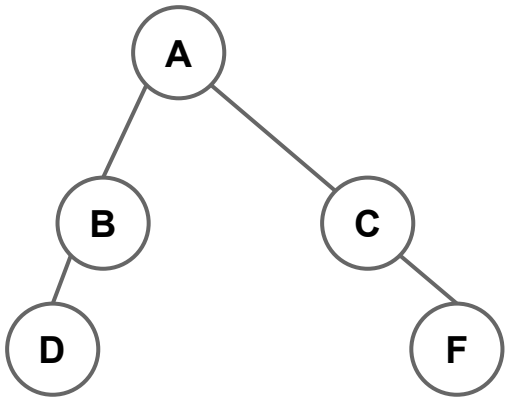
- Em outras palavras, podemos representar qualquer árvore m-ária por uma árvore binária -- aumentado a importância desta última

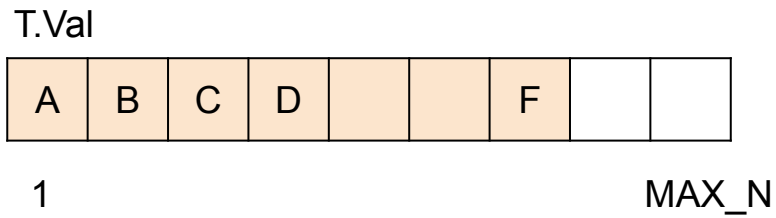
Árvores

- ***Como*** implementar uma árvore binária? (**continuação**)
  - Alocação sequencial de nós

```
var MAX_N: Inteiro ← <NÚMERO MÁXIMO DE ELEMENTOS>
//os nós são armezados em
//níveis, da esquerda para direita
estrutura No <TChave, TElem>:
    Chave: <TElem>
    Elem: <TElem>
estrutura Arvore <TChave, TElem>:
    Val[1..MAX_N]: No<TChave, TElem>
    N: Inteiro  //quantidade de nós
                //da árvore
procedimento Constroi(ref T: Arvore)
    T.N ← 0

var T: Arvore
Constroi(T)
```





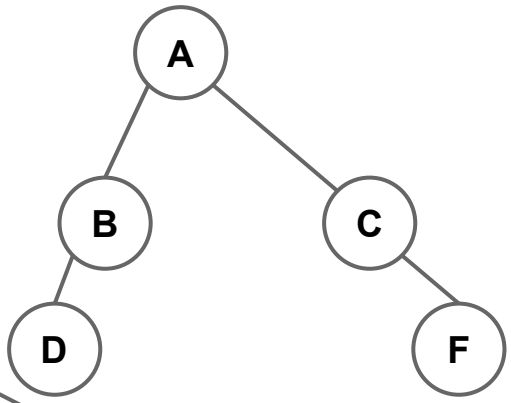
Árvores

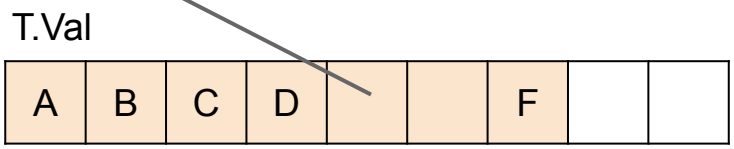
- ***Como*** implementar uma árvore binária? (**continuação**)
  - Alocação sequencial de nós

```
var MAX_N: Inteiro ← <NÚMERO MÁXIMO DE ELEMENTOS>
//os nós
//níveis
estrutur
    Chav
    Elem
estrutur
    Val[1..MAX_N]: No<TChave, TElem>
    N: Inteiro  //quantidade de nós
                //da árvore
procedimento Constroi(ref T: Arvore)
    T.N ← 0

var T: Arvore
Constroi(T)
```

Pode deixar muitos espaços sem uso… por isso, uso adequado para árvores completas





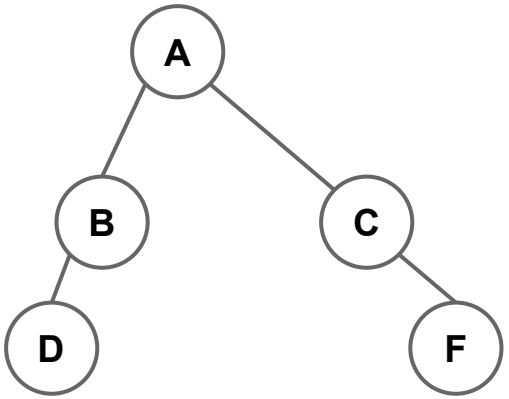
Árvores

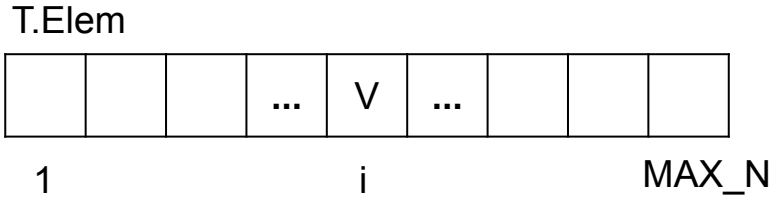
- ***Como*** implementar uma árvore binária?
  - Alocação sequencial de nós

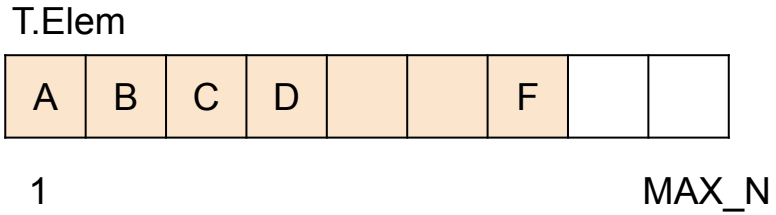
Para pensar:

Dado um nó V na posição i, em que posição estão:

- o pai de V?
- os filhos de V?







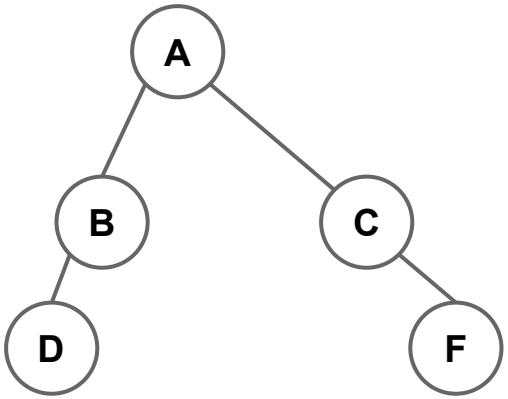
Árvores

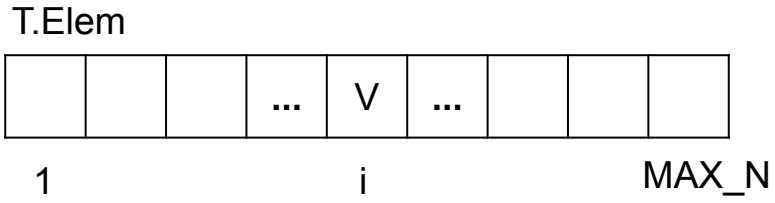
- ***Como*** implementar uma árvore binária?
  - Alocação sequencial de nós

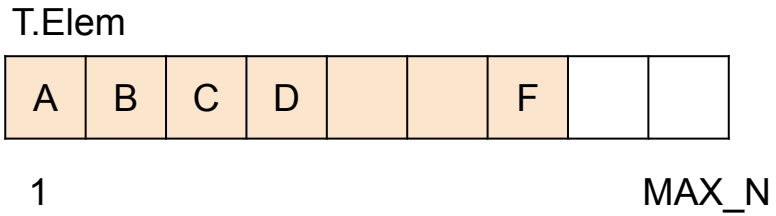
Para pensar:

Dado um nó V na posição i, em que posição estão:

- o pai de V? **⌊i/2⌋**
- os filhos de V? **2i e 2i+1**







Árvores

- Uma floresta pode ser representada computacionalmente criando-se artificialmente um nó raiz que une todas as árvores

Percursos

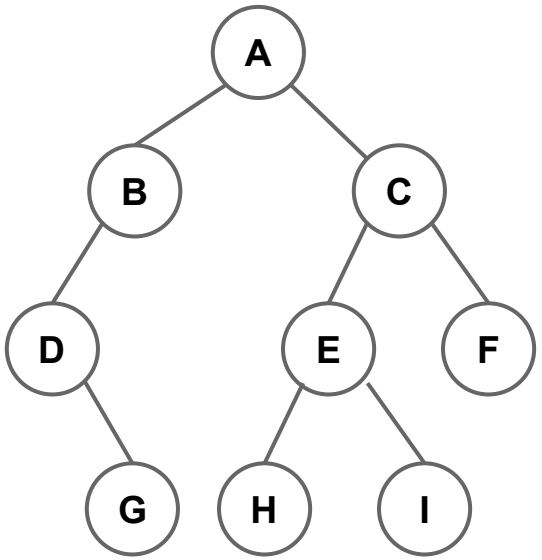
Buscas em Árvores

Árvores

- Um percurso é uma visitação ordenada dos nós
- Como não há uma ordem natural entre os nós (como é o caso das listas lineares), cada percurso pode visitar os nós em ordem distinta. Exemplos de percursos clássicos:
  - em pré-ordem
  - em pós-ordem
  - em in-ordem (ou em ordem simétrica)
  - em nível

Árvores

- Um percurso em pré-ordem numa árvore T é aquele que visita o nó raiz de T e, em seguida, percorre em pré-ordem as subárvores da esquerda e da direita, nesta ordem



Pré-ordem:

Visitas ocorrem na ordem:
A, B, D, G, C, E, H, I, F

Árvores

- Percurso em Pré-Ordem

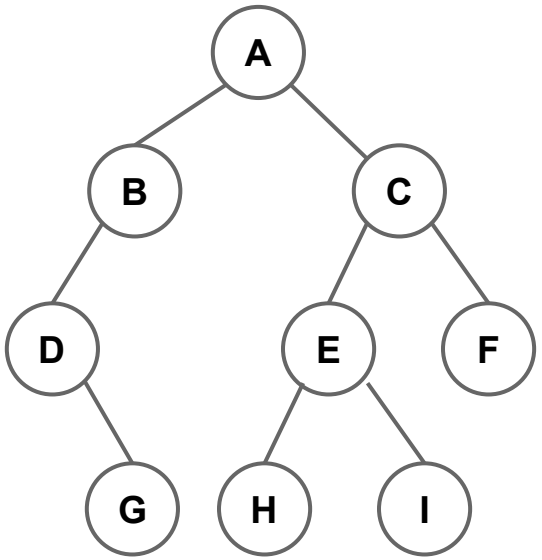
```
procedimento PercursoPreOrdem(T: ^No)
   se T ≠ NULO então
      escrever (T^.Chave)
      PercursoPreOrdem(T^.Esq)
      PercursoPreOrdem(T^.Dir)

procedimento PercursoPreOrdem(ref T: Arvore)
   PercursoPreOrdem(T.Raiz)
```

Tempo:
$\theta(N)$

Árvores

- Um percurso em pós-ordem numa árvore T é aquele que percorre em pós-ordem as subárvores da esquerda e da direita, nesta ordem, e, em seguida, visita o nó raiz de T



Pós-ordem:

Visitas ocorrem na ordem:
G, D, B, H, I, E, F, C, A

Árvores

- Percurso em Pós-Ordem

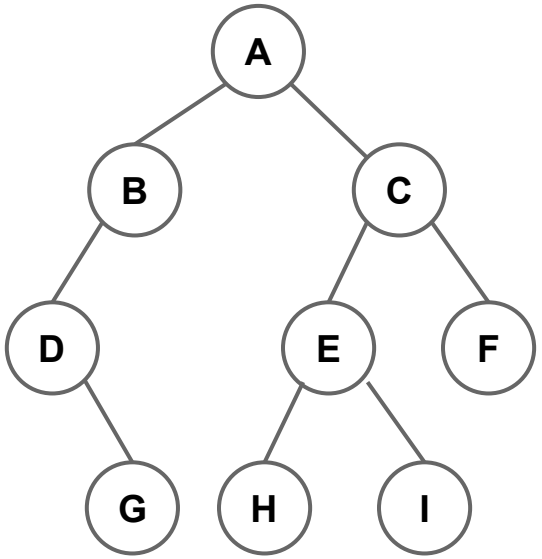
```
procedimento PercursoPosOrdem(T: ^No)
   se T ≠ NULO então
      PercursoPosOrdem(T^.Esq)
      PercursoPosOrdem(T^.Dir)
      escrever (T^.Chave)

procedimento PercursoPosOrdem(ref T: Arvore)
   PercursoPosOrdem(T.Raiz)
```

Tempo:
$\theta(N)$

Árvores

- Um percurso em in-ordem (ou em ordem simétrica) numa árvore T é aquele que percorre em in-ordem a subárvore da esquerda, depois visita o nó raiz de T, e, em seguida, percorre em in-ordem a subárvore da direita



In-ordem:

Visitas ocorrem na ordem:
D, G, B, A, H, E, I, C, F

Árvores

- Percurso em In-Ordem

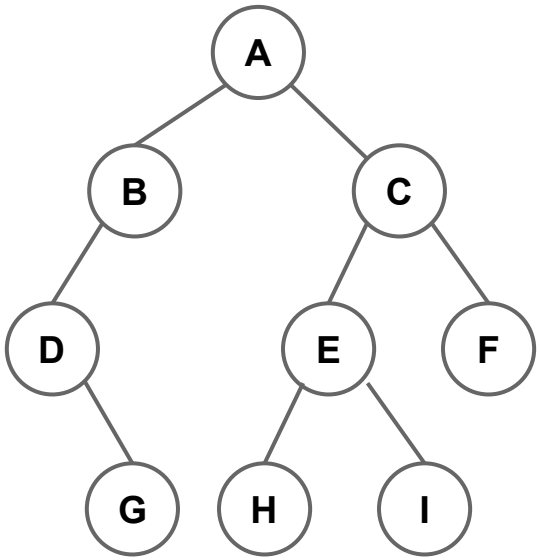
```
procedimento PercursoInOrdem(T: ^No)
   se T ≠ NULO então
      PercursoInOrdem(T^.Esq)
      escrever (T^.Chave)
      PercursoInOrdem(T^.Dir)

procedimento PercursoInOrdem(ref T: Arvore)
   PercursoInOrdem(T.Raiz)
```

Tempo:
$\theta(N)$

Árvores

- Um percurso em níveis numa árvore T é aquele que visita os nós por ordem ascendente de nível e, entre aqueles de mesmo nível, da esquerda para direita



Em nível:

Visitas ocorrem na ordem:
A, B, C, D, E, F, G, H, I

Árvores

```
procedimento PercursoEmNiveis(T: ^No)
   var TProx: ^No, F: Fila<^No>
   Enfileira(F, T)
   enquanto Tamanho(F) > 0 faça
      TProx ← Desenfileira(F)
      se TProx ≠ NULO então
         escrever(TProx^.Chave)
         Enfileira(F, TProx^.Esq)
         Enfileira(F, TProx^.Dir)

procedimento PercursoEmNiveis(ref T: Arvore)
   PercursoEmNiveis(T.Raiz)
```

Tempo:
$\theta(N)$

Exercícios

Exercícios

1. Faça um procedimento que em cada nó P de uma árvore dada como entrada atualize um campo chamado:
   a. Altura com a altura de P
   b. Nível com o nível de P
   c. SomaChaveD, com a soma de todas as chaves de nós descendentes de P
   d. SomaChaveD2, com a soma de todas as chaves de nós descendentes de P (excluindo aquela de P)
   e. SomaChaveA, com a soma de todas as chaves de nós dos quais P é descendente
   f. SomaChaveA2, com a soma de todas as chaves de nós dos quais P é descendente, exceto aquela de P
   g. PNivel com a porcentagem que o valor da chave de P representa em relação à soma de todas as chaves de nós que se encontram no mesmo nível de P

Exercícios

2. Faça um procedimento que execute um percurso em níveis que, para aqueles nós num mesmo nível da árvore, a ordem de visita seja da direita para a esquerda

3. Decidir se duas árvores binárias dadas de entrada são idênticas, isto é, contém os mesmos elementos exatamente nas mesmas posições

4. Decidir se, dadas duas árvores binárias de entrada, uma é subárvore da outra, isto é, um nó de uma árvore é raiz de uma subárvore idêntica à segunda

5. Verificar se uma árvore T m-ária dada de entrada é balanceada, conforme os seguintes critérios de balanceamento:
   a. T é balanceada se para quaisquer folhas u e v, as distâncias de u e v até raiz difere de no máximo 1
   b. T é balanceada se para qualquer nó T' de T, a diferença das alturas das subárvores à esquerda e à direita de T' é no máximo 1

Exercícios

6. Faça algoritmos que, dada uma árvore binária cujos elementos são inteiros, computem:
   a. o número de valores
   b. o produto dos valores
   c. a soma dos valores em folhas da árvore
   d. o maior valor
   e. uma lista ligada com os nós que não são folhas
   f. o menor valor dentre aqueles associados a nós que não possuem subárvores vazias
   g. a maior soma de valores dos nós de uma subárvore (uma subárvore de T é a árvore que se obtém tomando-se um nó X de T e eliminando-se de T todos os nós que não sejam X ou não descendam de X)

Exercícios

7. Dada uma árvore binária T de altura H, criar H listas ligadas de forma que para cada $1 \leq i \leq H$ haja uma lista ligada com os nós de altura i

8. Dada uma árvore binária T com número de níveis J, criar J listas ligadas tal que para cada $1 \leq i \leq J$ haja uma lista ligada com os nós de nível i

9. Dado um ponteiro para um nó de uma árvore que acaba de ser visitado, determinar o próximo nó que será visitado em relação a cada percurso abaixo usando o menor número possível de operações. (Exemplos são dados no formato (nó de entrada → nó de saída) considerando-se a árvore usada de exemplo para os percursos)
   a. in-ordem (ordem simétrica). (Ex: B → A; A → H; I → C)
   b. pré-ordem (Ex: A → B; G → C; H → I)
   c. pós-ordem (Ex: D → B; B → H)
   d. em nível (Ex: B → C; F → G; H → I)

   Suponha que cada nó possui um ponteiro para o nó pai

Exercícios

10. Prove ou refute: Para que duas árvores binárias sejam idênticas, é suficiente possuírem a mesma sequência de visitas aos nós nos percursos:
    a. em níveis
    b. em pré-ordem
    c. em pós-ordem
    d. em in-ordem
    e. em pré-ordem e in-ordem
    f. em pós-ordem e in-ordem
    g. em pré-ordem e pós-ordem
    h. em pré-ordem e em níveis

Exercícios

11. Em uma árvore cheia representada com alocação sequencial, determine a posição i no vetor representando os nós da árvore onde está o nó que atende cada uma das propriedades abaixo:
    a. possui o neto na posição p. Ex: p = 12 ⇒ i = 3.
    b. que é o k-ésimo nó da esquerda para direita no nível ℓ. Ex: k = 3, ℓ = 3 ⇒ i = 10.
    c. que se chega após navegar a partir da raiz r nós à direita seguido de ℓ nós à esquerda. Ex: d = 2, ℓ = 1 ⇒ i = 14.

12. Sobre árvores estritamente m-árias com n nós, responda:
    a. Qual a altura máxima dentre tais árvores?
    b. Qual a altura mínima dentre tais árvores?

Exercícios

13. Elabore um algoritmo com complexidade de tempo θ(N) que atribua ao campo Niv de cada um dos N nós de uma árvore binária com o respectivo nível do nó usando percursos (a) pré-ordem; (b) in-ordem, (c) pós-ordem e (d) em nível

14. Elabore um algoritmo com complexidade de tempo θ(N) que escreva as chaves de uma árvore binária T dada de entrada tal que cada nível de T apareça numa linha de saída.

15. Elabore um algoritmo com complexidade de tempo O(lg N) que insire um novo par chave/elemento numa árvore. A estrutura de um nó de árvore pode ser modificado para conter uma espaço extra de θ(1).