

# Árvores

Árvores Binárias

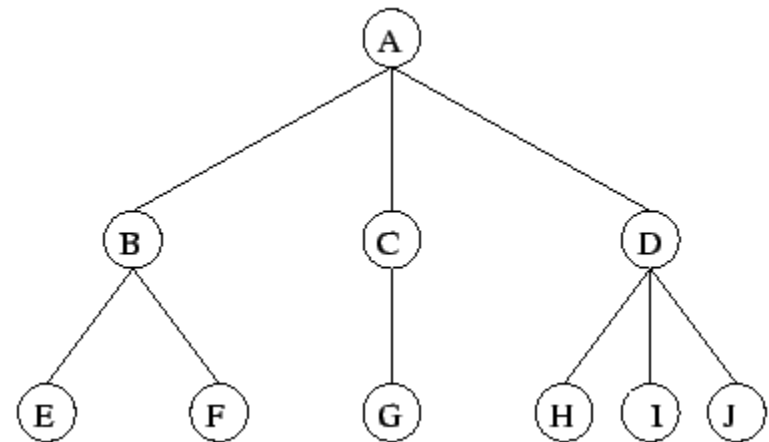
Árvores Binárias de Busca

# Árvores

- A principal característica desta estrutura é a organização das informações, que se faz por meio de ramos.

**Definição:** Uma *árvore* é um conjunto finito de um ou mais nós tal que:

- Existe um nó especial denominado *raiz*
- Os outros nós formam conjuntos disjuntos  $T_1, T_2, \dots, T_n$ , sendo que cada um desses conjuntos é uma árvore.



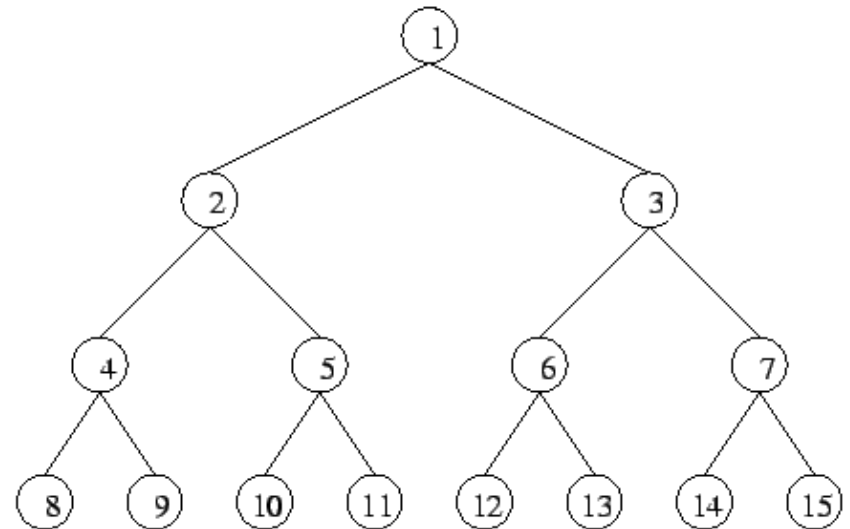
- **Obs:** Todo nó de uma árvore é a raiz de uma sub-árvore.

# Conceitos

- **Grau de um nó:** é o número de sub-árvores do nó.
- **Grau de uma árvore:** é o máximo grau dos nós na árvore.
- **Folhas de uma árvore:** são os nós de grau zero.
- **Filhos de um nó X:** são as raízes das sub-árvores do nó X. Neste caso X é o **Pai** de seus filhos.
- **Nível de um nó:** a raiz da árvore é dita estar no nível zero, se um nó está no nível  $k$ , seus filhos estão no nível  $k+1$ .
- **Altura ou profundidade de uma árvore:** é o nível máximo dos nós na árvore.
- **Ancestrais de um nó:** são todos os nós ao longo do caminho a partir da raiz até o nó.
- **Floresta:** conjunto de árvores disjuntas.

# Árvores Binárias

- Uma **árvore binária** é um conjunto finito de elementos que está vazio ou é particionado em três subconjuntos disjuntos.
- O primeiro subconjunto contém um único elemento, chamado **raiz da árvore**.
- Os outros dois subconjuntos são em si mesmos árvores binárias, chamadas **subárvores esquerda e direita** da árvore original.
- Uma subárvore esquerda ou direita pode estar vazia.
- Cada elemento de uma árvore binária é chamado **nó da árvore**.
- **Propriedade:** O número máximo de nós no nível  $i$  de uma árvore binária é  $2^i$  (compare com a soma de uma linha  $i$  do Triângulo de Pascal)

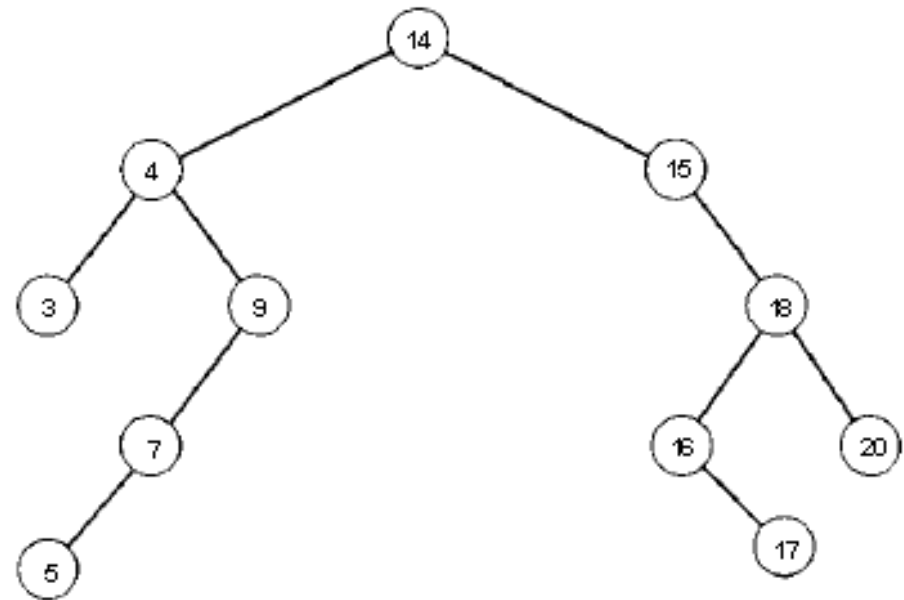


# Árvore Binária de Busca

- é uma árvore binária
- o filho esquerdo de um nó é sempre  $\leq$  ao pai
- o filho direito de um nó é maior que o pai

# Árvore Binária de Busca

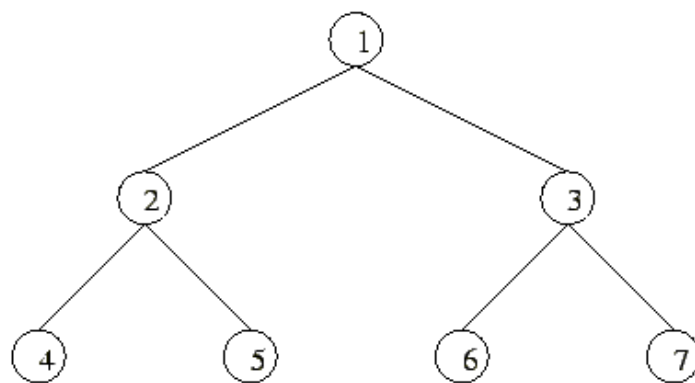
- Uma árvore binária com essa propriedade é chamada **árvore de busca binária**.
- Se uma árvore de busca binária for percorrida em ordem simétrica (esquerda, raiz, direita) e o conteúdo de cada nó for impresso à medida que o nó for visitado, os números serão impressos em ordem ascendente.
- A árvore de busca binária da Figura ao lado ilustra a descrição acima



# Representação implícita

- O esquema de enumeração de nós descrito na seção anterior permite armazenarmos uma árvore binária completa em *array* de dimensão 1:
    - Se uma árvore binária completa com  $n$  nós é armazenada em um array de acordo com o esquema de enumeração acima, então para qualquer nó  $i$  temos: (supondo que o array comece em 1)
1.  $\text{Pai}(i) = \lfloor i/2 \rfloor$ . Se  $i$  é 1 então ele é a raiz da árvore.
  2.  $\text{Filho\_esq}(i) = 2i$ . Se  $2i$  é maior que  $n$  então  $i$  não tem filho da esquerda.
  3.  $\text{Filho\_dir}(i) = 2i+1$ . Se  $2i+1$  é maior que  $n$  então  $i$  não tem filho da direita.

# Representação implícita



$$\text{Filho\_esq}(1) = 2 * 1 = 2$$

$$\text{Filho\_dir}(1) = 2 * 1 + 1 = 3$$





# Representação Encadeada

- Uma árvore binária (= *binary tree*) é um conjunto de estruturas que satisfaz certas condições:
  - Os registros são chamados *nós* ou *células*.
  - Cada nó possui além das informações que devem ser armazenadas, endereços ou referências de memória para os seus nós sucessores.
- A implementação deve considerar que cada nó tem três campos:
  - a informação: uma classe qualquer
  - e as referências para os nós sucessores: o filho da direita e o filho da esquerda, se existirem.

# APLICAÇÕES DE ÁRVORES BINÁRIAS

- Uma árvore binária é uma estrutura de dados útil quando precisam ser tomadas decisões bidirecionais em cada ponto de um processo. Por exemplo, encontrar todas as repetições numa lista de números.

# APLICAÇÕES DE ÁRVORES BINÁRIAS

- Uma maneira de fazer isso é comparar cada número com todos os que o precedem. Entretanto, isso envolve um grande número de comparações. O número de comparações pode ser reduzido usando-se uma árvore binária.
  - O primeiro número na lista é colocado na raiz de uma árvore binária com as subárvores esquerda e direita vazias.
  - Cada número sucessivo na lista é, então, comparado ao número na raiz. Se coincidirem, configura-se repetição.
  - Se for menor, examina-se a subárvore esquerda; se for maior, examina-se a subárvore direita.
  - Se a subárvore estiver vazia, o número não será repetido e será colocado num novo nó nesta posição na árvore.
  - Se a subárvore não estiver vazia, compararemos o número ao conteúdo da raiz da subárvore e o processo inteiro será repetido com a subárvore.

# Percurso em Árvores

- Outra operação comum é percorrer uma árvore binária, ou seja, percorrer a árvore enumerando cada um de seus nós uma vez.
- Pode-se simplesmente querer imprimir o conteúdo de cada nó ao enumerá-lo, ou pode-se processá-lo de alguma maneira.
- Seja qual for o caso, visita-se cada um dos nós à medida que são enumerados.
- Evidentemente, a ordem na qual os nós de uma lista linear são visitados num percurso é do primeiro para o último (ou vice-versa).
- Entretanto, não existe uma ordem "natural" para os nós de uma árvore. Sendo assim, são usados diferentes ordenamentos de percurso em diferentes casos.
- Definem-se três desses métodos de percurso. Em cada um desses métodos, não é preciso fazer nada para percorrer uma árvore binária vazia.
- Todos os métodos são definidos recursivamente, de modo que percorrer uma árvore binária envolve visitar a raiz e percorrer suas subárvores esquerda e direita.
- **A única diferença entre os métodos é a ordem na qual essas três operações são efetuadas.**

# Pré-ordem

- Para percorrer uma árvore binária não-vazia *em pré-ordem (conhecida também como percurso em profundidade)*, efetuam-se as três seguintes operações:
  1. Visita-se a raiz.
  2. Percorre-se a subárvore esquerda em ordem prévia.
  3. Percorre-se a subárvore direita em ordem prévia.

- ***Em ordem (ou ordem simétrica):***

1. Percorre-se a subárvore esquerda em ordem simétrica.
2. Visita-se a raiz.
3. Percorre-se a subárvore direita em ordem simétrica.

- ***Em pós-ordem:***

1. Percorre-se a subárvore esquerda em ordem posterior.
2. Percorre-se a subárvore direita em ordem posterior.
3. Visita-se a raiz.