Estrutura de Dados
Árvores

Prof. Luciano

Definição

- Até agora estudamos estrutura lineares, uma vez que cada nodo da lista possui apenas um sucessor e um antecessor.
- Estrutura de dados em que um nodo pode ter vários sucessores e apenas um antecessor.
- Exemplos:
  - avô, pai, filho, neto, bisneto,…
  - livro, capítulo, secção, parágrafo, frase

Definição

- Uma árvore A é um conjunto finito de nodos tais que:
  - Se $A = 0$ denomina-se árvore vazia

  ou

  - Se $A \neq 0$ existe um nodo especial chamado Raiz de A; os nodos restantes constituem um único conjunto vazio ou são divididos em $m \geq 1$ conjuntos disjuntos não vazios, denominamos de subárvores de A. Cada um destes conjuntos, por sua vez também é uma árvore.

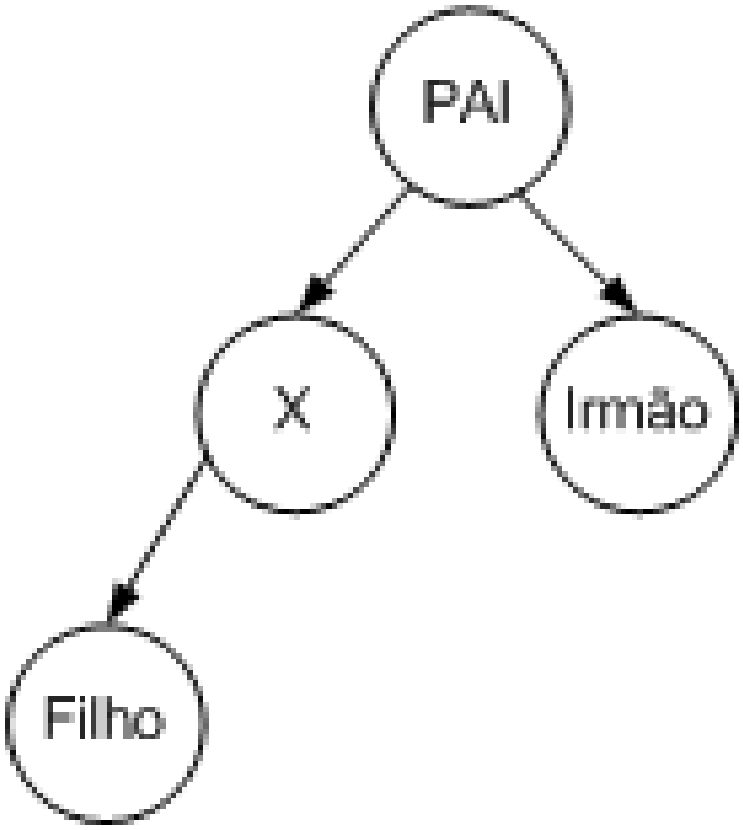
Outras definições

- Grau de um nodo: é o número de subárvores de um nodo;
- Nível de um nodo: é a distância do nodo até o nodo raiz;
- Altura de uma árvore: é o maior nível de uma árvore;
- Raiz de uma árvore: é o nodo que ocupa o nível 0 da árvore;
- Folha de uma árvore: é todo nodo de grau zero, isto é, que não tem filhos;
- Galho de uma árvore: é qualquer nodo que não é nem raiz e nem folha da árvore;

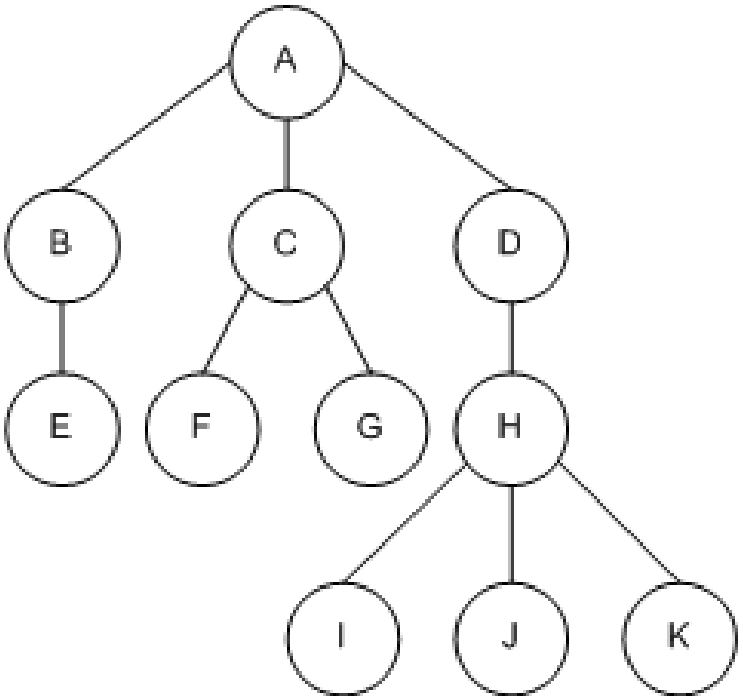
Outras definições

- Aresta de uma árvore: é a linha que liga dois nodos da árvore;
- Caminho entre dois nodos de uma árvore: existe um caminho entre dois nodos A e B de uma árvore, se a partir do nodo A pode-se chegar ao nodo B percorrendo-se as arestas que ligam os nodos intermediários entre A e B;
- Floresta: é um conjunto de árvores;
- Árvore binária: quando cada nodo possui no máximo duas subárvores diz-se que é uma árvore binária;

Denominação relativa dos nodos

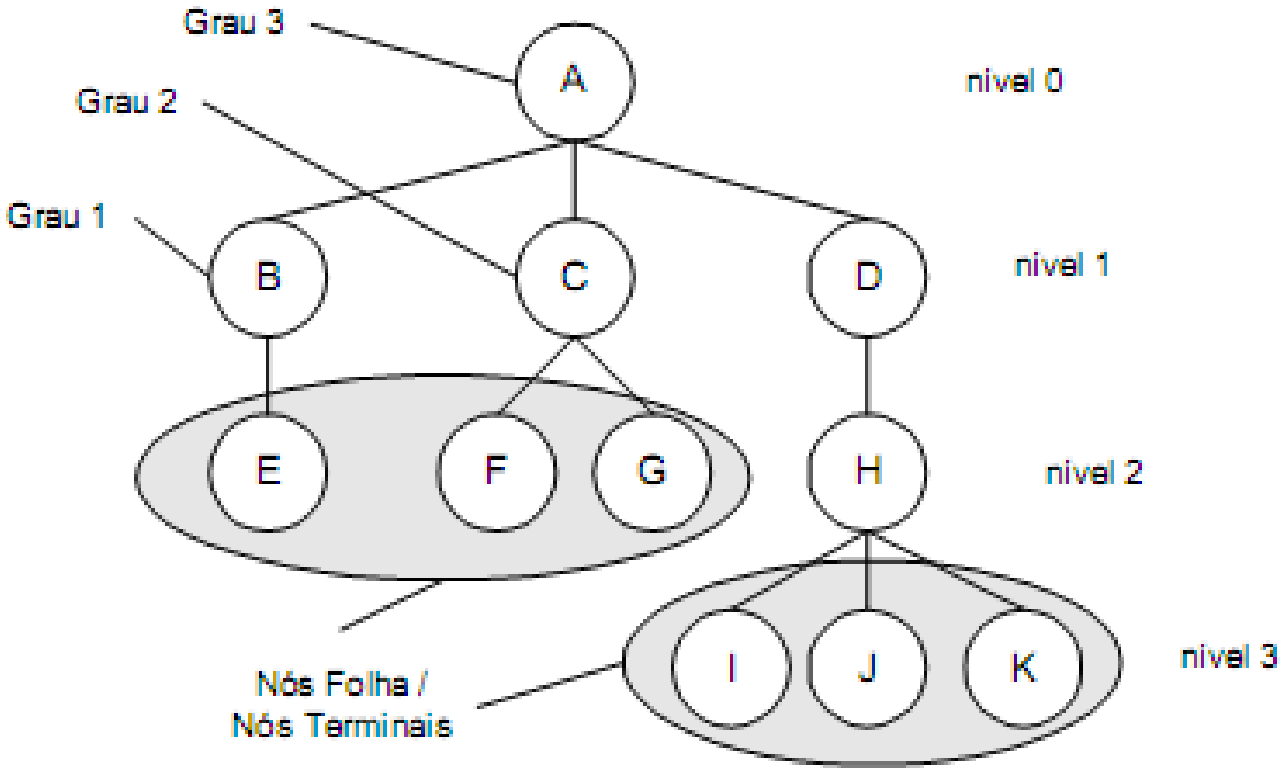


Exemplo: utilizando as definições

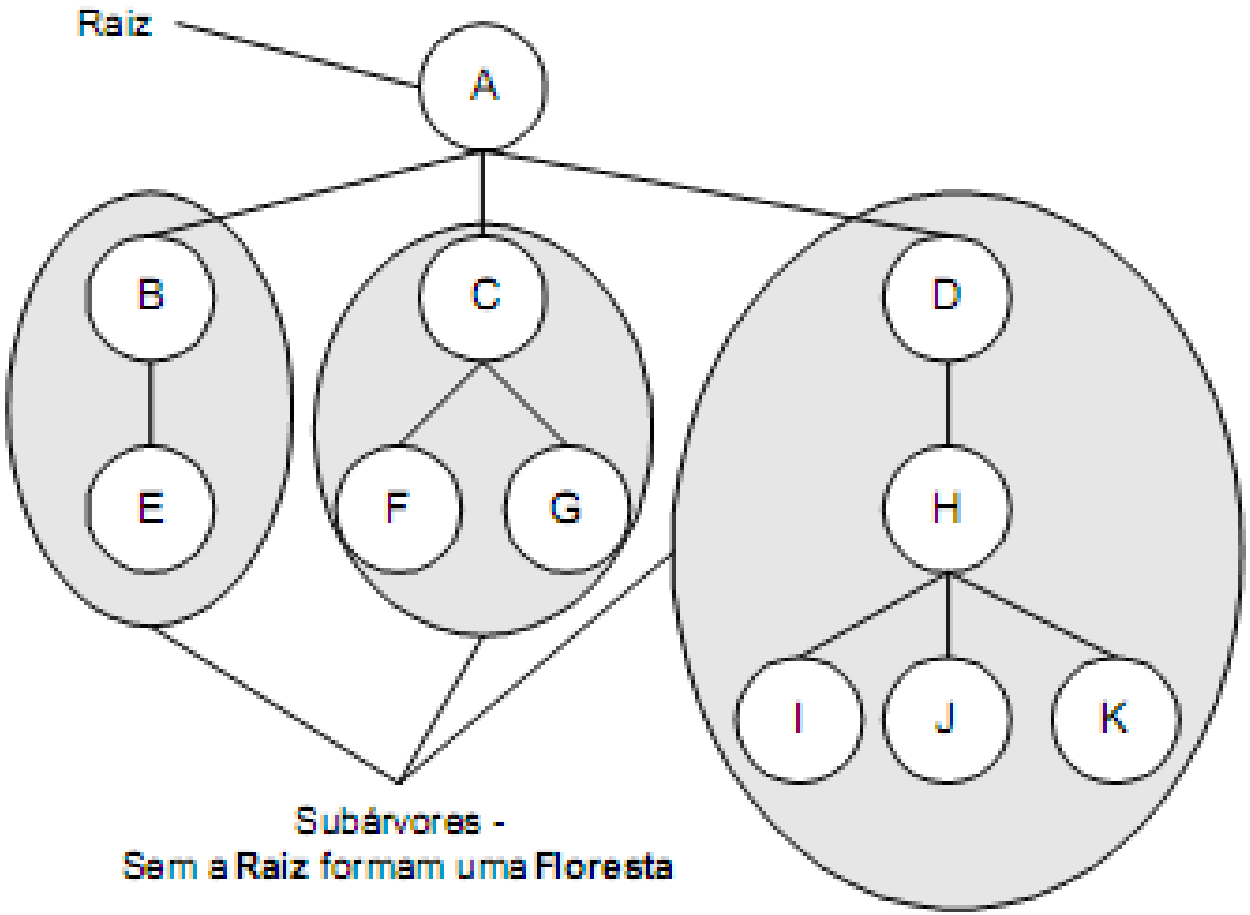


Nodo	Grau	Nível	Observações
A	3	0	Raiz da árvore
B	1	1	
C	2	1	
D	1	1	
E	0	2	Nó terminal (ou folha)
F	0	2	Nó terminal (ou folha)
G	0	2	Nó terminal (ou folha)
H	3	2	
I	0	3	Nó terminal (ou folha)
J	0	3	Nó terminal (ou folha)
K	0	3	Nó terminal (ou folha)

Exemplo: utilizando as definições

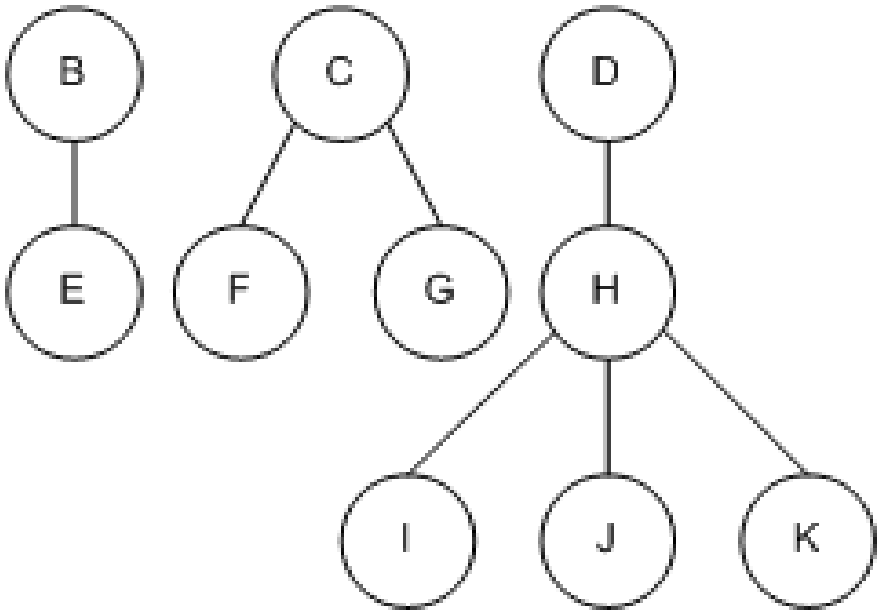


Exemplo: utilizando as definições



Utilizando as definições

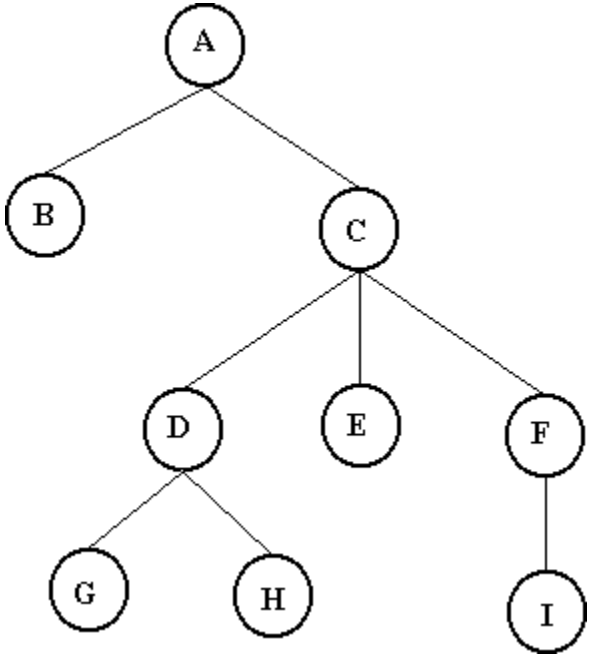
- Floresta elimina-se a raiz de uma árvore, o que sobrar é a floresta.



Representação

- Várias formas para representar graficamente uma estrutura de árvore:
  - Tradicional (hierárquica)
  -Diagrama de inclusão
  - Diagrama de barras
  - Parênteses aninhados

Tradicional (hierárquico)



Parênteses aninhados

(A (B) (C (D (G) (H)) (E) (F (I))))

Diagrama de barras

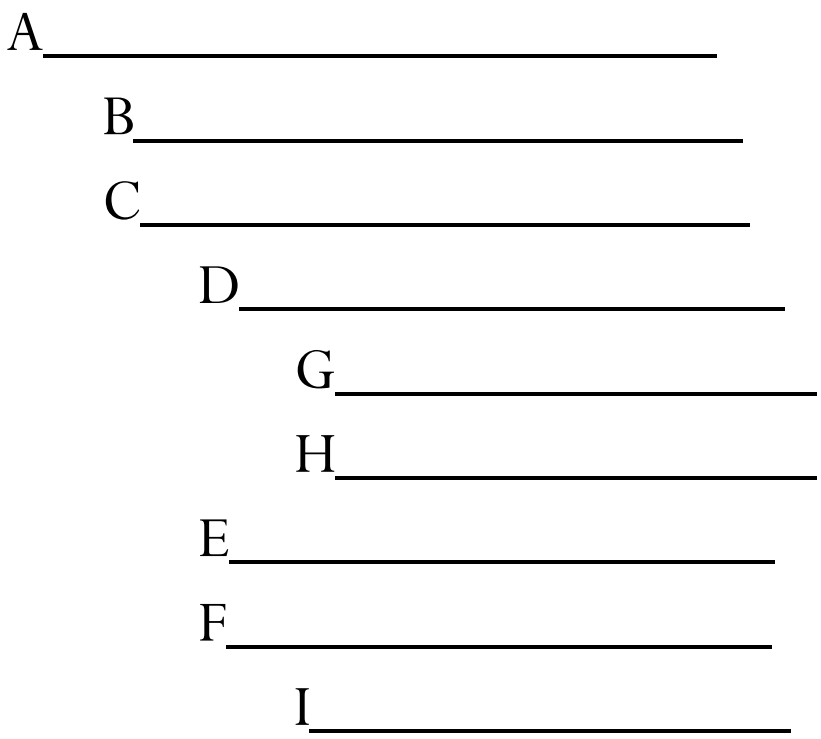
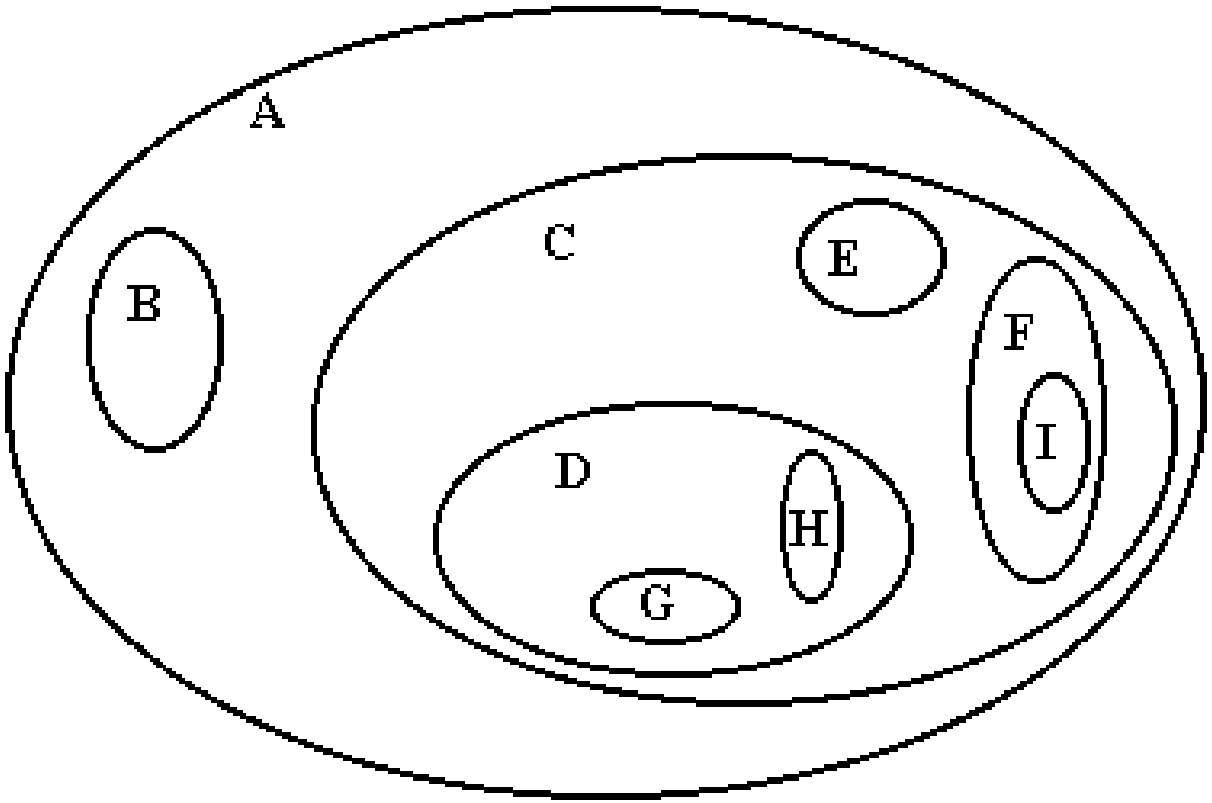
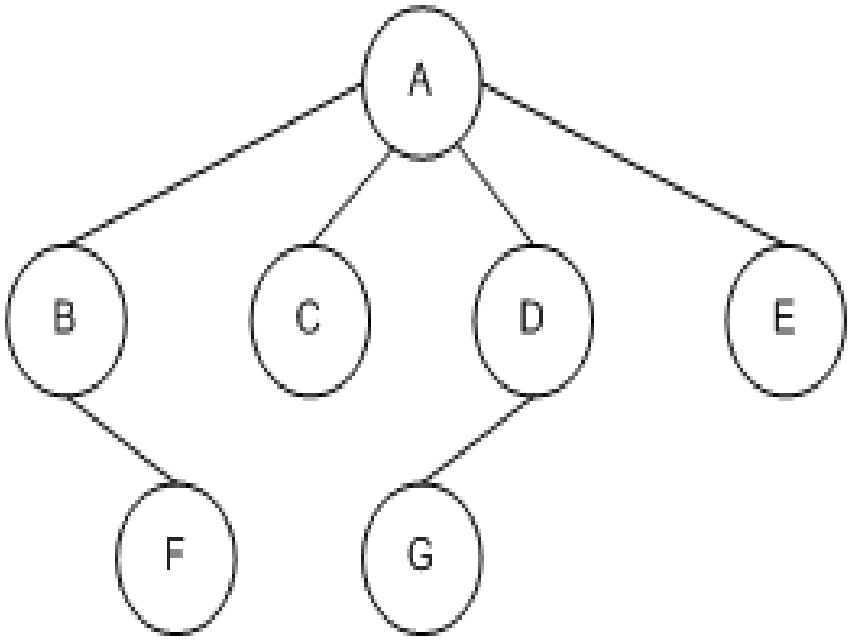


Diagrama de inclusão



Exercício

- Utilizando a árvore ao lado responda:
  - o grau de cada nodo da árvore;
  - o nível de cada nodo da árvore;
  - a altura da árvore;
  - a denominação relativa dos nodos diretamente conectados aos nodos de nível 1;
  - o nodo raiz;
  - os nodos galhos;
  - os nodos folhas;

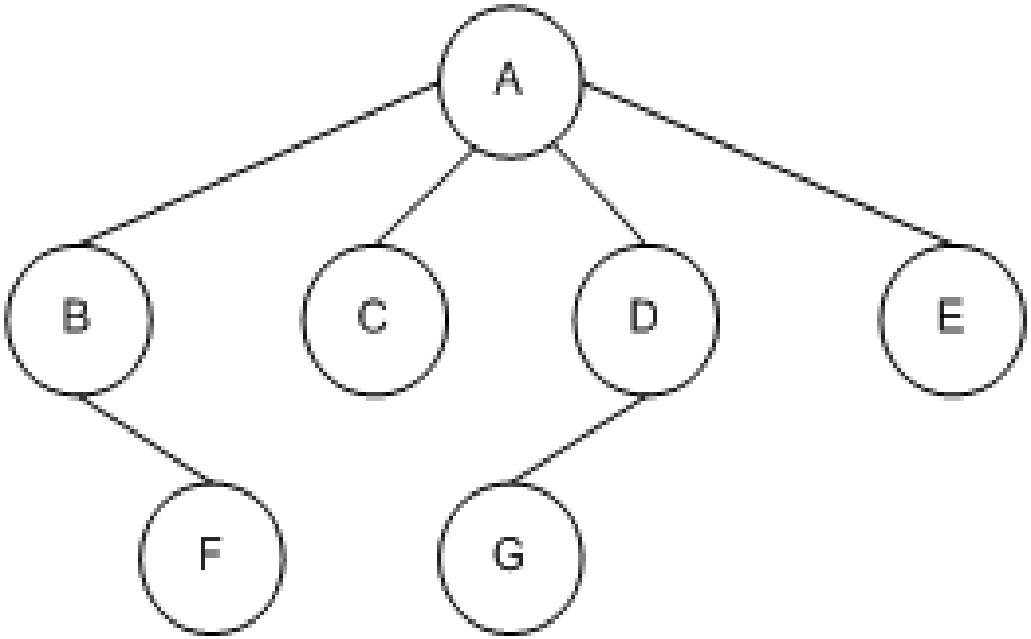


Conversão de árvore genérica em árvore binária

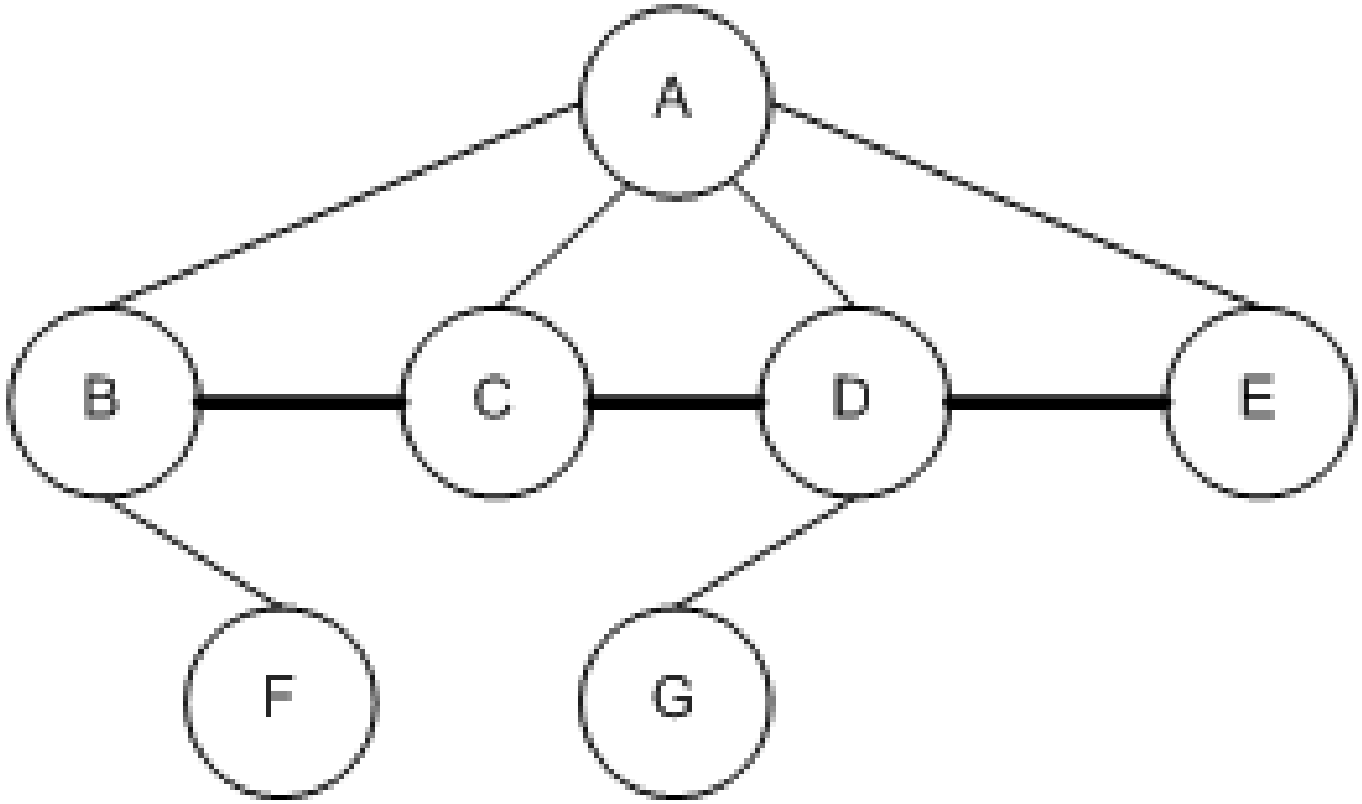
- Em uma árvore genérica o grau dos nodos pode ser qualquer um, o que dificulta a representação em computador deste tipo de árvore.
- Converter a árvore genérica em árvore binária, facilita a representação.
- Regras de conversão:
  1) Ligar todos os nodos irmãos entre si;
  2) Desligar o nodo pai de seus filhos, exceto o primeiro.

Regras de conversão

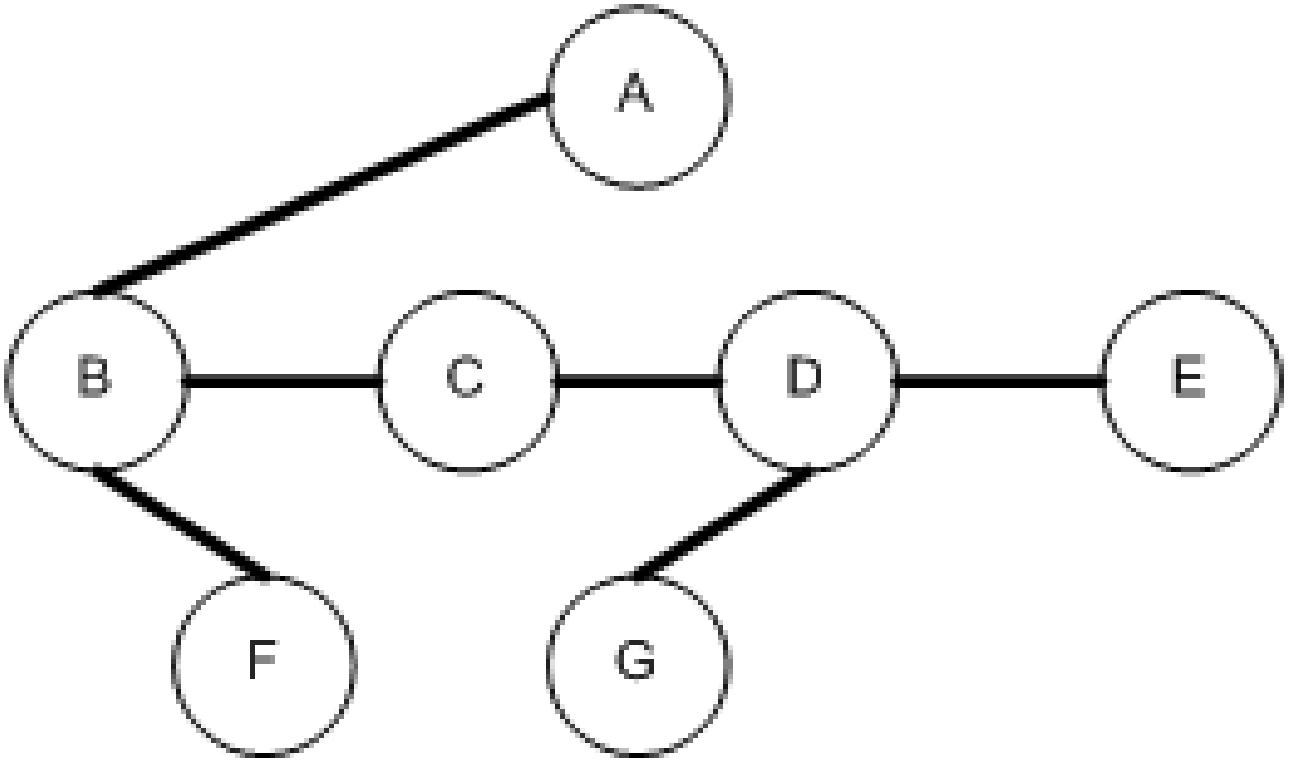
- Ligar todos os nodos irmãos entre si;
- Desligar o nodo pai de seus filhos, exceto o primeiro.



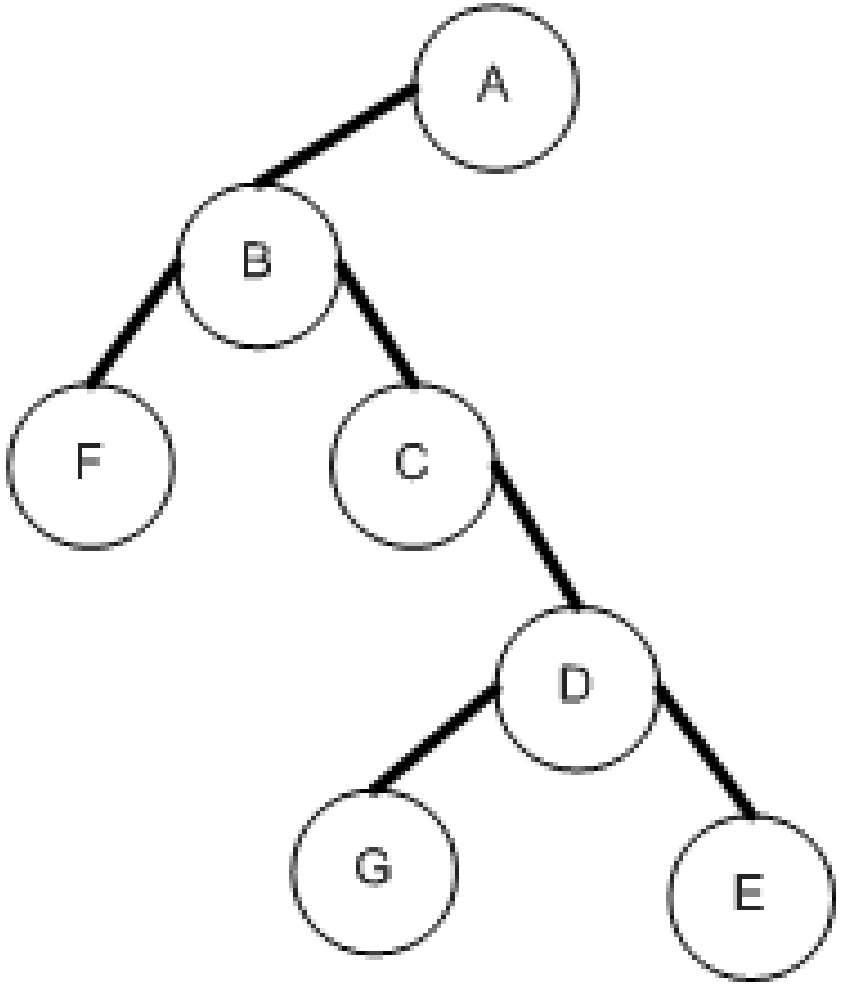
Aplicando a regra 1 temos:



Aplicando a regra 2 resulta:



Redesenhando a árvore

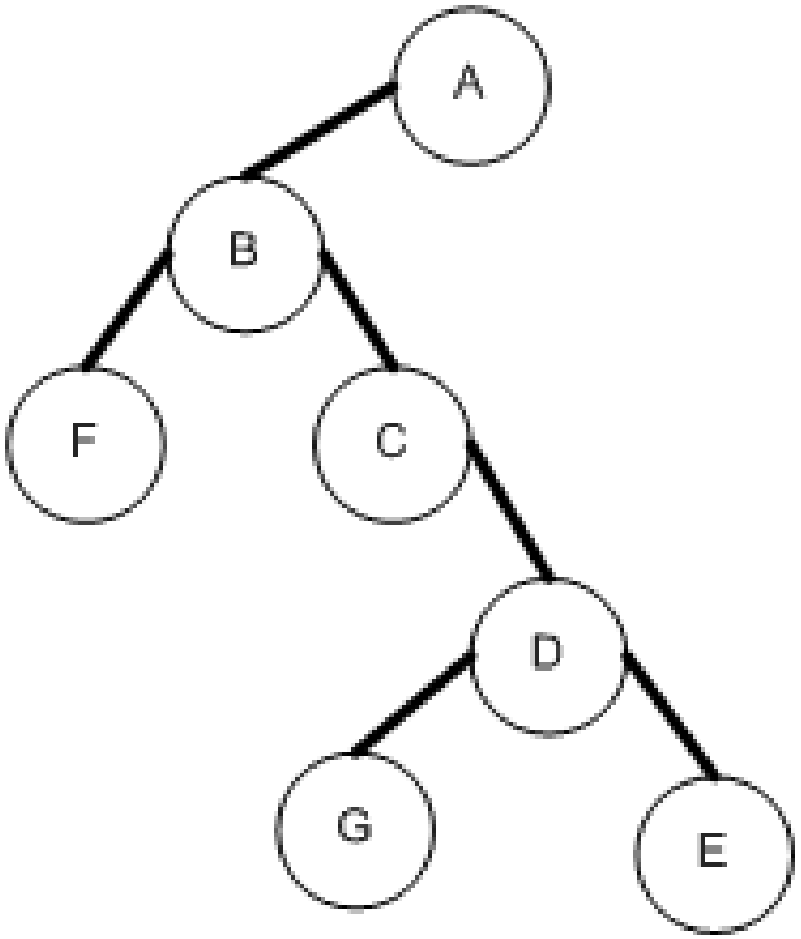


Caminhamento em Árvores

- Forma ordenada de percorrer todos os nodos da árvore;
- Três formas de caminhamento:
  - Pré-fixado: segue a ordem **raiz, esquerda, direita**
  - Central: segue a ordem **esquerda, raiz, direita**
  - Pós-fixado: segue a ordem **esquerda, direita, raiz**

Exemplo

- Pré-fixado: A,B,F,C,D,G,E
- Central: F,B,G,D,E,C,A
- Pós-fixado: F,G,E,D,C,B,A



Expressões aritméticas utilizando árvores

- Considere a seguinte expressão:

$A + B^2 - (C-5)/2*D$

Representação na forma de árvore

- Pré-fixado (red)

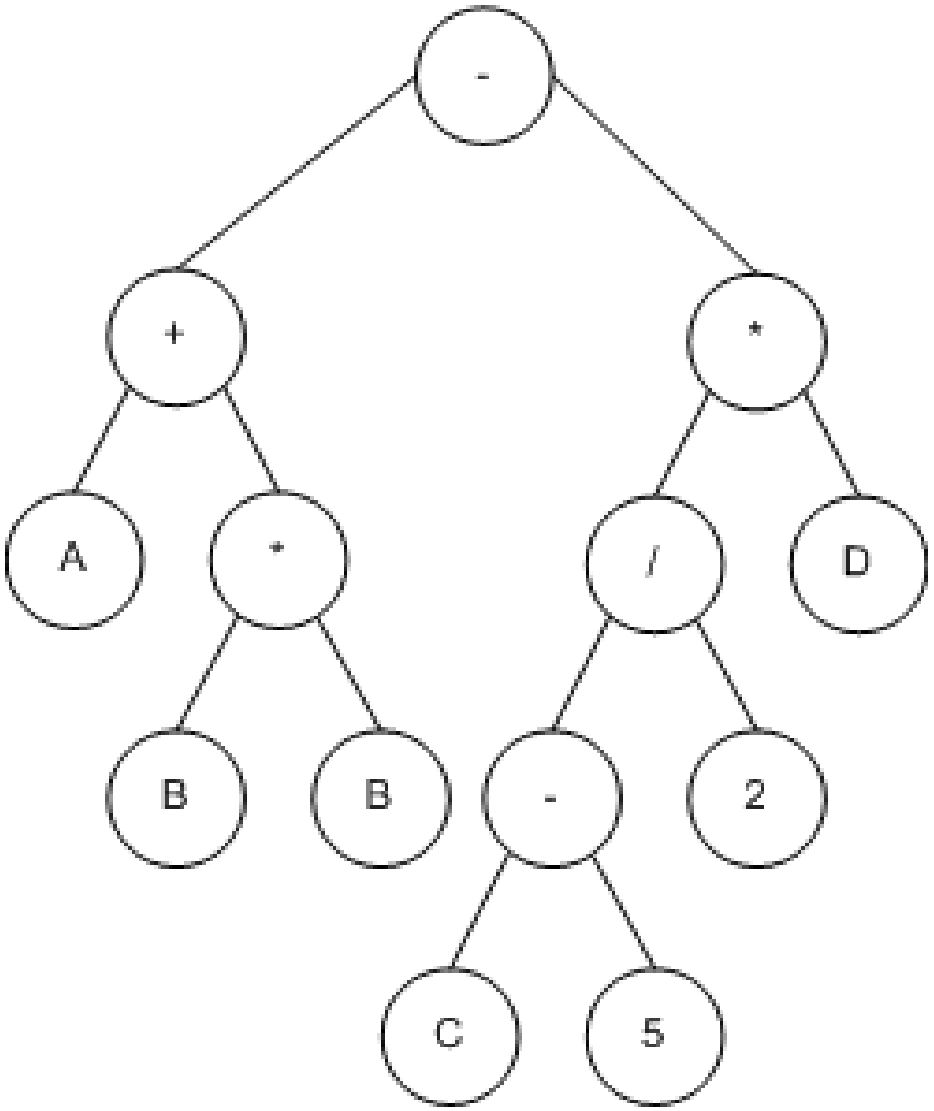
  `-+A*BB*/-C52D`

- Central (erd)

  `A+B*B-C-5/2*D`

- Pós-fixado (edr)

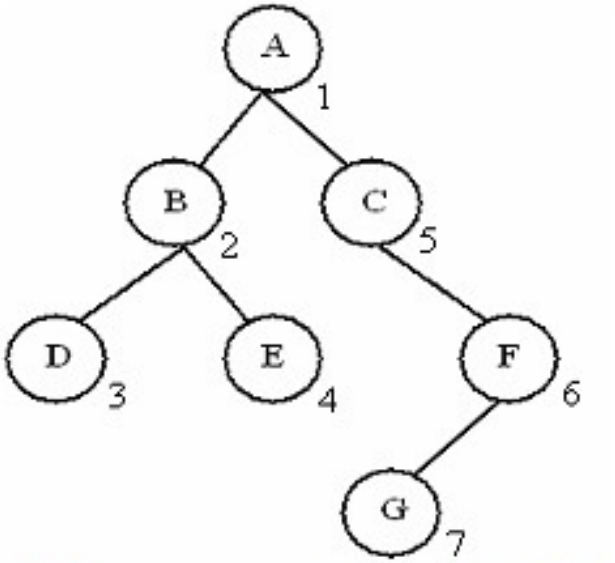
  `ABB*+C5-2/D*-`



Pré-fixado

- Visitar a raiz;
- Caminhar na sub-árvore da esquerda;
- Caminhar na sub-árvore da direita.

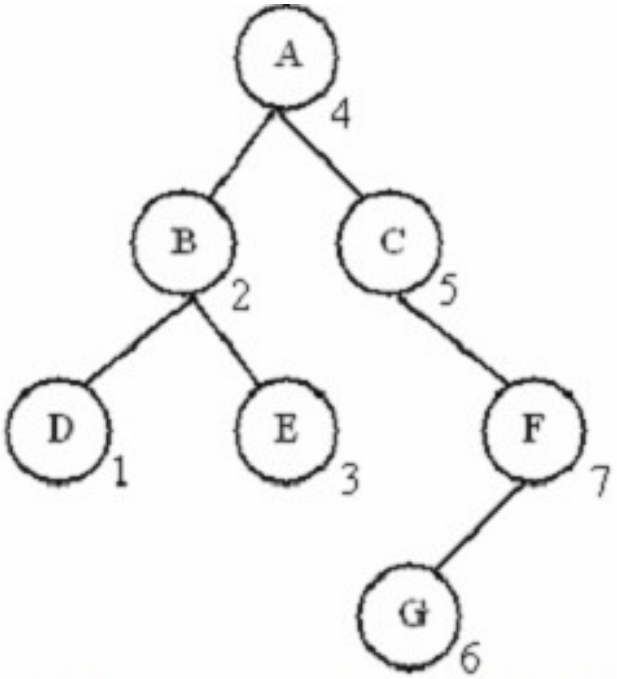
OBS.: “visitar” significa qualquer operação em relação à informação (dado) do nodo.



Caminhamento: ABDECFG

Central ou infixado

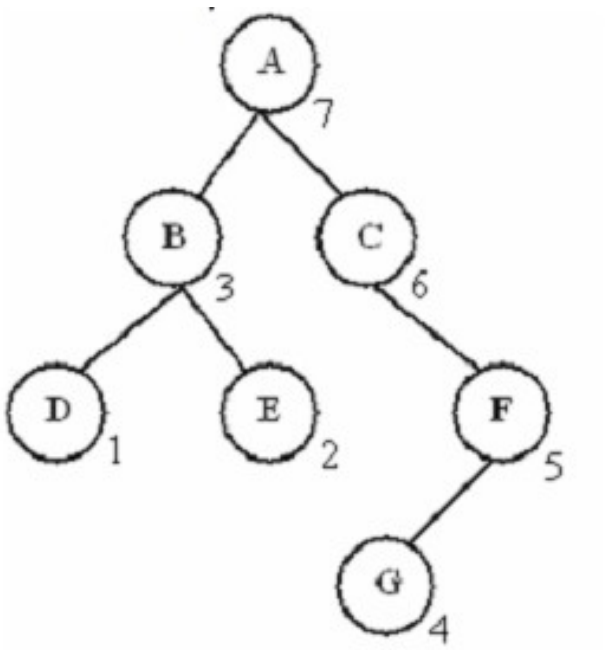
- Caminhar na sub-árvore da esquerda;
- Visitar a raiz;
- Caminhar na sub-árvore da direita.



Caminhamento: DBEACGF

Pós-fixado

- Caminhar na sub-árvore da esquerda;
- Caminhar na sub-árvore da direita;
- Visitar a raiz.



Caminhamento: DEBGFCA

Representação de árvore no computador

- Contiguidade física (vetores)
- Estruturas encadeadas

Contiguidade física (vetores)

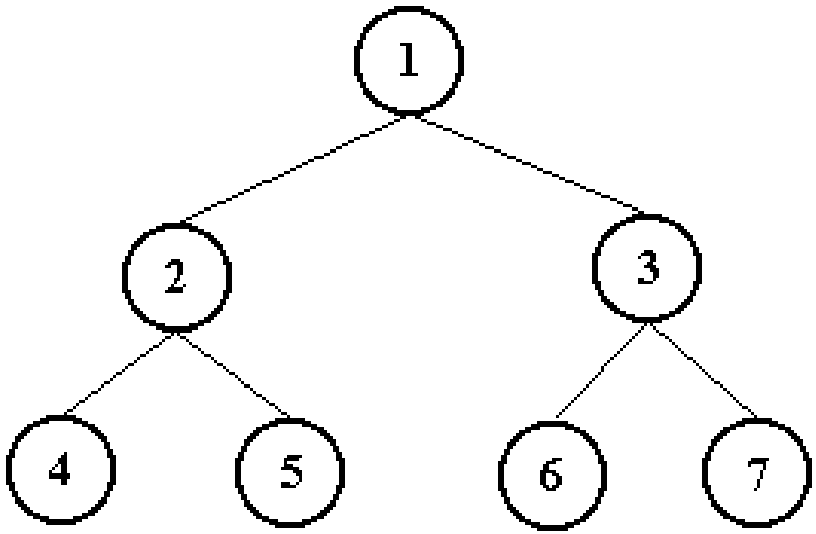
- Vetores podem ser usados para representar tanto árvores binárias quanto árvores genéricas.

Árvores binárias

- Para representar uma árvore binária usas-se a seguinte representação:

  1. A raiz da árvore é colocada na posição 1 do vetor.
  2. O filho esquerdo de um nodo armazenado na posição **n** é colocado na posição **2n**.
  3. O filho direito de um nodo armazenado na posição **n** é colocado na posição **2n+1**.

Exemplo: seja a seguinte árvore binária



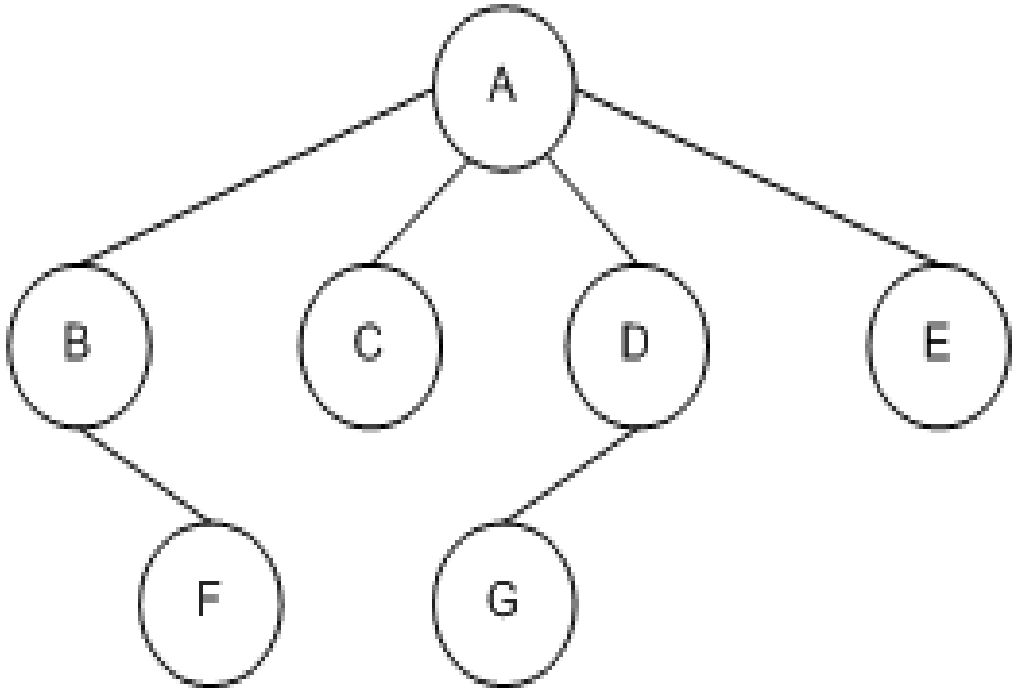
Posição do nó	Posição dos filhos do nó
1	2,3
2	4,5
3	6,7
i	(2n, 2n+1)

1	2	3	4	5	6	7

Árvores genéricas

- Como árvore genérica pode ter um número variável de filhos em cada nodo, deve-se informar o número de filhos esperado como um dado adicional armazenado no vetor. Neste caso, pode-se adotar a seguinte convenção:
  1. o nodo raiz fica armazenado na posição 1 do vetor;
  2. na posição seguinte ao nodo raiz é armazenado seu número de filhos;
  3. o primeiro filho à esquerda é armazenado na posição imediatamente seguinte ao número de filhos do nodo, e então é armazenada toda a sub-árvore desse filho;
  4. os demais irmãos são armazenados em seqüência juntamente com as suas sub-árvores.

Exemplo: seja a seguinte árvore genérica



1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	4	B	1	F	0	C	0	D	1	G	0	E	0

Representação

- Para armazenar os nodos sequencialmente na memória deve-se tomar os seguintes cuidados:
  - Ser armazenado espaço suficiente para armazenas a **estrutura completa**.
  - Os nodos devem ser armazenados em uma lista, onde cada nodo **i** da árvore ocupa o **i-ésimo** nodo da lista.
- Vantagens
  - Adequado para armazenar árvores binárias completas
  - Útil para o armazenamento em disco ou fita (sequencial)
- Desvantagens
  - A estrutura pode ter muitos espaços sem uso
  - Estrutura possui limite finito no número de nodos

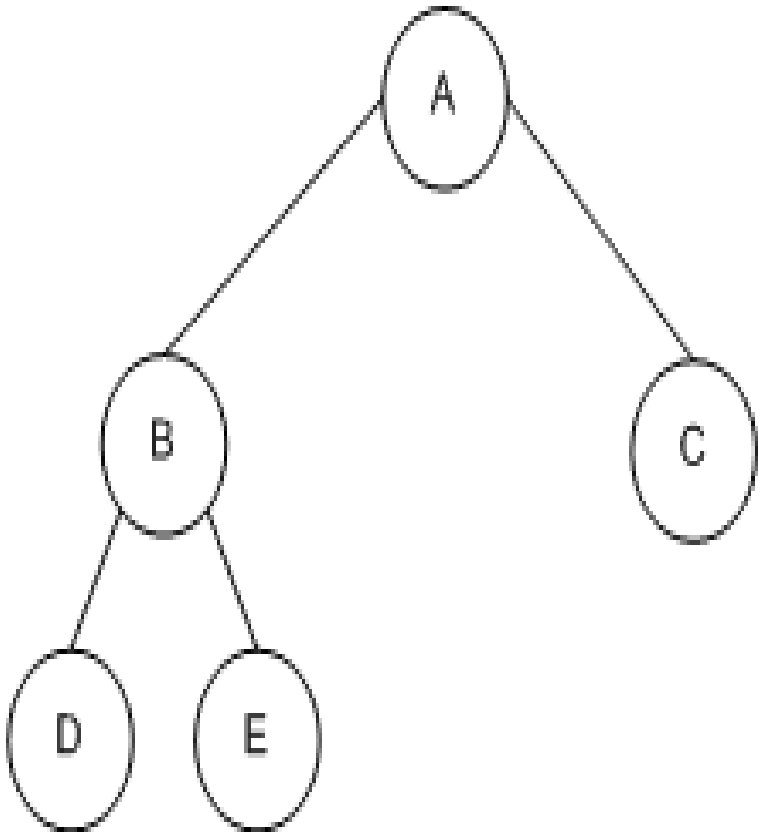
Estruturas encadeadas

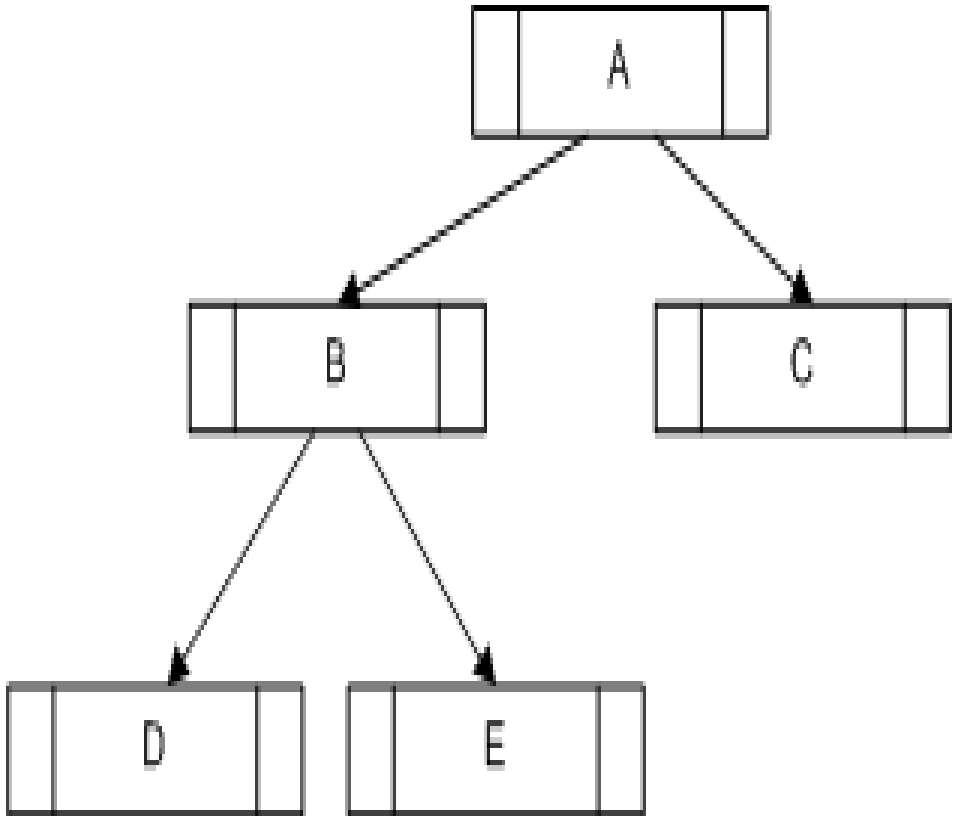
- O encadeamento, associado à alocação dinâmica e à recursividade, oferece a alternativa mais flexível para a representação de árvores em computador.

Representação

- Esq: endereço do nodo filho à esquerda;
- Dado: contém a informação do nodo;
- Dir: endereço do nodo filho à direita.

Representação





Representação

```
typedef int TipoDado;
typedef struct TipoNo *TipoApontador;
typedef struct TipoNo {
  TipoDado Dado;
  TipoApontador Esq, Dir;
} TipoNo;

//Programa principal
TipoNo *Arvore;
```

Operações típicas sobre árvores

- Caminhamentos em árvores
- Pesquisa em árvores
- Remoção de nodos de árvores
- Adição de nodos em árvores
- Conversões diversas

Caminhamento Pré-fixado

```
void Prefix(TipoApontador p)
{ if (p == NULL) return;
  printf("%d ", p->Dado);
  Prefix(p->Esq);
  Prefix(p->Dir);
}
```

Caminhamento Central

```
void Central(TipoApontador p)
{ if (p == NULL) return;
  Central(p->Esq);
  printf("%d ", p->Dado);
  Central(p->Dir);
}
```

Caminhamento Central

```
// Recebe a raiz r de uma árvore binária.
// Imprime os conteúdos dos nós em ordem e-r-d.
// Supõe que a árvore não tem mais que 100 nós.

void erd_i( arvore r) {
    no *x, *p[100];
    int t = 0;
    x = r;
    while (x != NULL || t > 0) {
       // a pilha é p[0..t-1]; o índice do topo é t-1
       if (x != NULL) {
          p[t++] = x;
          x = x->esq;
       }
       else {
          x = p[--t];
          printf( "%d\n", x->conteudo);
          x = x->dir;
       }
    }
}
```

Caminhamento Pós-fixado

```
void Posfix(TipoApontador p)
{ if (p == NULL) return;
  Poafix(p->Esq);
  Posfix(p->Dir);
  printf("%d ", p->Dado);
}
```

Pesquisa árvore binária

Procedimento para Pesquisar na Árvore Uma Chave x.

- Compare-a com a chave que está na raiz.
- Se x é menor, vá para a subárvore esquerda.
- Se x é maior, vá para a subárvore direita.
- Repita o processo recursivamente, até que a chave procurada seja encontrada ou um nó folha é atingido.
- Se a pesquisa tiver sucesso o conteúdo retorna no próprio registro x.

Pesquisa árvore binária

```
void Pesquisa(TipoRegistro *x, TipoApontador *p)
{ if (*p == NULL)
  { printf("Erro: Registro nao esta presente na arvore\n");
    return;
  }
  if (x->Chave < (*p)->Reg.Chave)
  { Pesquisa(x, &(*p)->Esq);
    return;
  }
  if (x->Chave > (*p)->Reg.Chave)
  Pesquisa(x, &(*p)->Dir);
  else *x = (*p)->Reg;
}
```

Vamos a prática