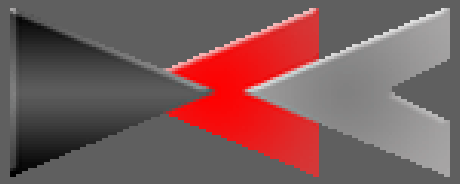
DCC107



Laboratório de Programação II

Tipos Abstratos de Dados Árvores Binárias (TAD Árvore Binária) em C

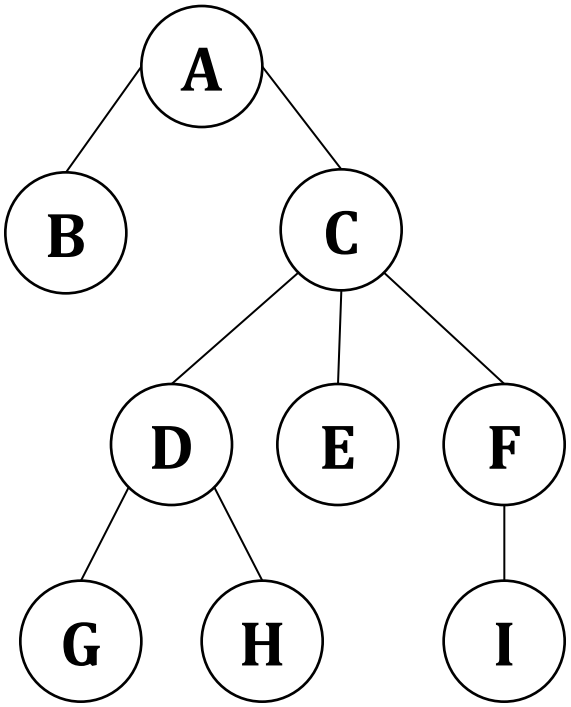
Árvores

- Árvore:
  - Estrutura de dados caracterizada por uma relação de hierarquia entre os elementos que a compõem;
- Árvore Binária:
  - Caso especial de Árvore;
  - Tem como princípio que cada nó da árvore tem no máximo dois descendentes;

Árvores

- Elementos de uma Árvore:
  - **Nó**: elemento que contém a informação;
  - **Arco**: liga dois nós;
  - **Pai**: nó superior de um arco;
  - **Filho**: nó inferior de um arco;
  - **Raiz**: nó topo – não tem um nó pai;
  - **Folhas**: nós das extremidades– não têm nós filhos;
  - **Grau**: representa o número de subávores de um nó;
  - **Grau de uma árvore** (aridade): é definido como sendo igual ao máximo dos graus de seus nós.

Árvores



Grau(T) = 3

- Graus dos nós:
  - G(A)=2
  - G(B)=0
  - G(C)=3
  - G(D)=2
  - G(E)=0
  - G(F)=1
  - G(G)=0
  - G(H)=0
  - G(I)=0

Nós Internos

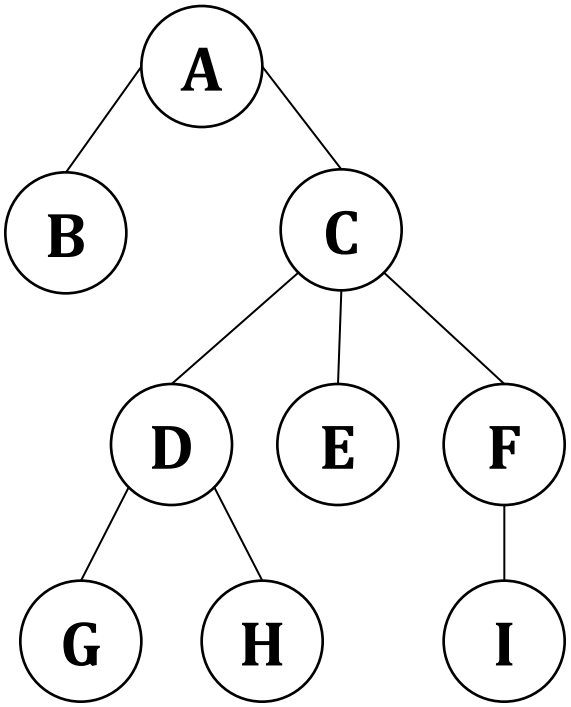
Folhas

Árvores

- Árvore:
  - Cada nó tem que ser atingível a partir da raiz através de uma sequência única de arcos, chamado de **caminho**;
  - **Comprimento** do caminho: o número de arcos do caminho;
  - **Altura** (ou profundidade) é o nível do nó folha que tem o mais longo caminho até a raiz, somando 1.
    - A árvore vazia é uma árvore de altura -1, por definição;
    - Uma árvore com um único nó tem altura 1.

Árvores

- Níveis e altura:

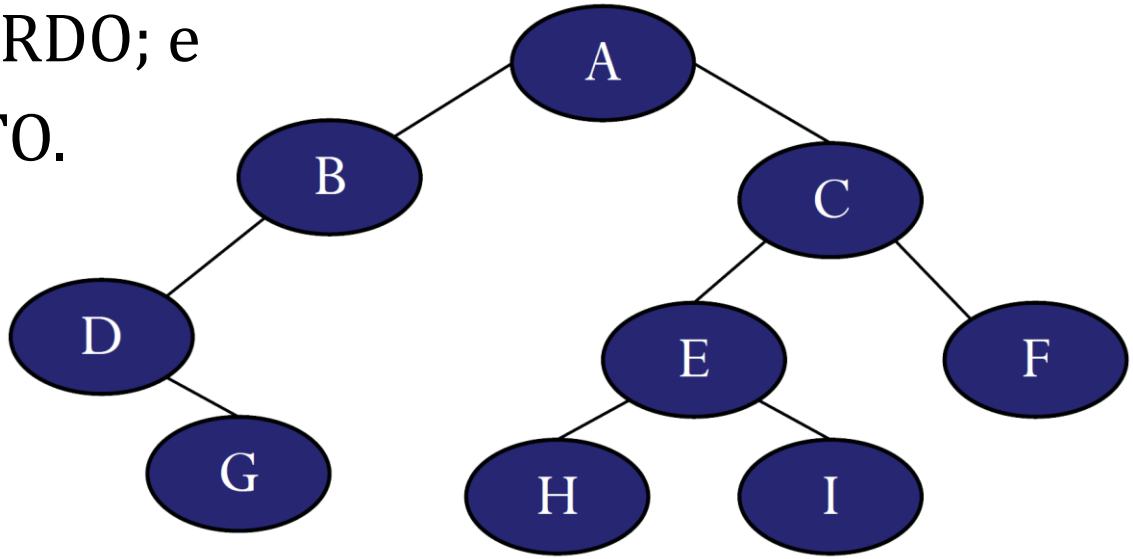


Nível	
A	0
B, C	1
D, E, F	2
G, H, I	3

h(árvore) = 4

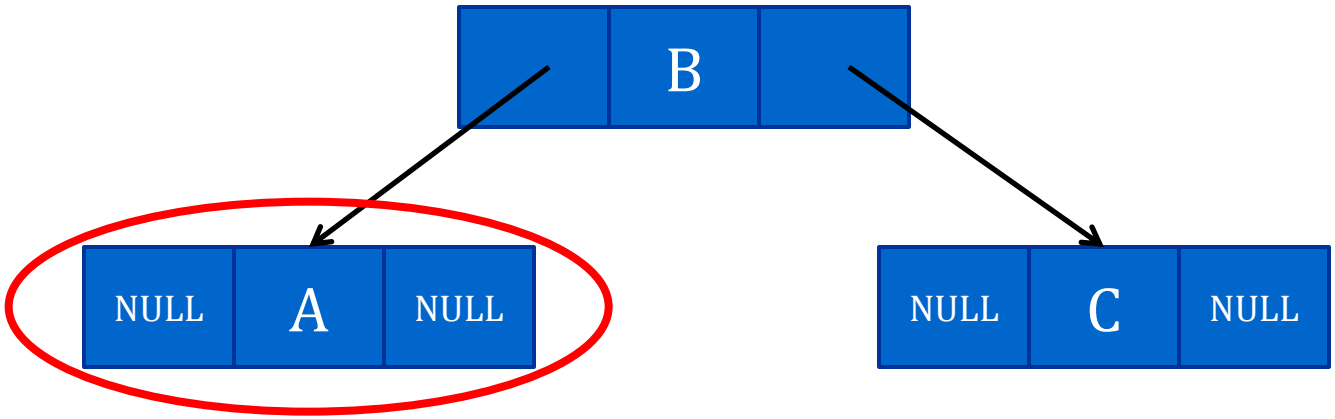
Árvores Binárias

- Uma árvore binária é uma árvore cujos nós têm dois filhos (alguns vazios) e cada filho é designado como filho à esquerda ou filho à direita. Portanto, a árvore binária tem grau máximo 2:
  - Nó filho ESQUERDO; e
  - Nó filho DIREITO.



Árvores Binárias

- Representação Encadeada:
  - Um nó será formado por um registro (struct) composto de:
    - Campo de informação;
    - Ponteiro para nó esquerdo;
    - Ponteiro para nó direito.



Nó da árvore

Árvores Binárias

- Estrutura:

```
struct arv {
   int info;
   struct arv *esq;
   struct arv *dir;
};
typedef struct arv Arv;
```

TAD Árvore Binária

- Principais operações:

//cria uma árvore vazia

```
Arv* inicializa();
```

//cria um nó raiz dadas a informação e as duas subárvores

```
Arv* cria(int c, Arv *sae, Arv *sad);
```

//verifica se a arvore está vazia

```
int vazia(Arv *a);
```

//imprime a informação de todos os nós da árvore

```
void imprime(Arv *a);
```

//libera a estrutura da árvore

```
Arv* libera(Arv *a);
```

TAD Árvore Binária

- Cria:

```
Arv* cria(int c, Arv *sae, Arv *sad)
{
    Arv *p = (Arv*) malloc(sizeof(Arv));
    p->info = c;
    p->esq = sae;
    p->dir = sad;
    return p;
}
```

- Inicializa:

```
Arv* inicializa()
{
    return NULL;
}
```

TAD Árvore Binária

- Vazia:

```
int vazia(Arv *a)
{
    return a == NULL;
}
```

- Imprime:

```
void imprime(Arv *a)
{
    if(!vazia(a))
    {
        printf("%d ", a->info); /* mostra raiz */
        imprime(a->esq); /* mostra sae */
        imprime(a->dir); /* mostra sad */
    }
}
```

TAD Árvore Binária

- Libera:

```
Arv* libera(Arv *a)
{
    if(!vazia(a))
    {
        libera(a->esq); /* libera sae */
        libera(a->dir); /* libera sad */
        free(a); /* libera raiz */
    }
    return NULL;
}
```

Exercícios

- Determinar se uma informação se encontra ou não na árvore:
  - `void busca(Arv *a, int c);`
- Determinar a altura de uma árvore:
  - `int altura(Arv *a);`
- Contar quantos são os nós de uma árvore:
  - `int nos(Arv *a);`
- Contar quantas são as folhas de uma árvore:
  - `int folhas(Arv *a);`

TAD Árvore Binária

- Busca:

```
void busca(Arv *a, int c)
{
    if(!vazia(a))
    {
        if(a->info == c) /* achou o elemento */
            printf("\nEncontrou o %d.", a->info);
        else
        {
            busca(a->esq, c); /* busca na sae */
            busca(a->dir, c); /* busca na sad */
        }
    }
}
```

TAD Árvore Binária

- Altura:

```
int altura(Arv *a)
{
    if(a == NULL)
        return 0;
    else
    {
        int he = altura(a->esq);
        int hd = altura(a->dir);
        if(he < hd)
            return hd + 1;
        else
            return he + 1;
   }
}
```

TAD Árvore Binária

- Número de Nós:

```
int nos(Arv *a)
{
    if(a == NULL)
        return 0;
    else
        return 1 + nos(a->esq) + nos(a->dir);
}
```

TAD Árvore Binária

- Número de Folhas:

```
int folhas(Arv *a)
{
    if(a == NULL)
        return 0;
    else
        if(a->esq == NULL & a->esq == NULL)
            return 1;
        else
            return folhas(a->esq) + folhas(a->dir);
}
```