



Data Structures

Ch5

# 树和二叉树 Trees & Binary Trees

2022 年 10 月 17 日

学而不厌 诲人不倦

- ➡ 5.1 引言
- ➡ 5.2 树的逻辑结构
- ➡ 5.3 树的存储结构
- ➡ 5.4 二叉树的逻辑结构
- ➡ 5.5 二叉树的存储结构
- ➡ 5.6 森林
- ➡ 5.7 最优二叉树
- ➡ 5.8 扩展与提高
- ➡ 5.9 应用实例

## 5.4 二叉树的逻辑结构

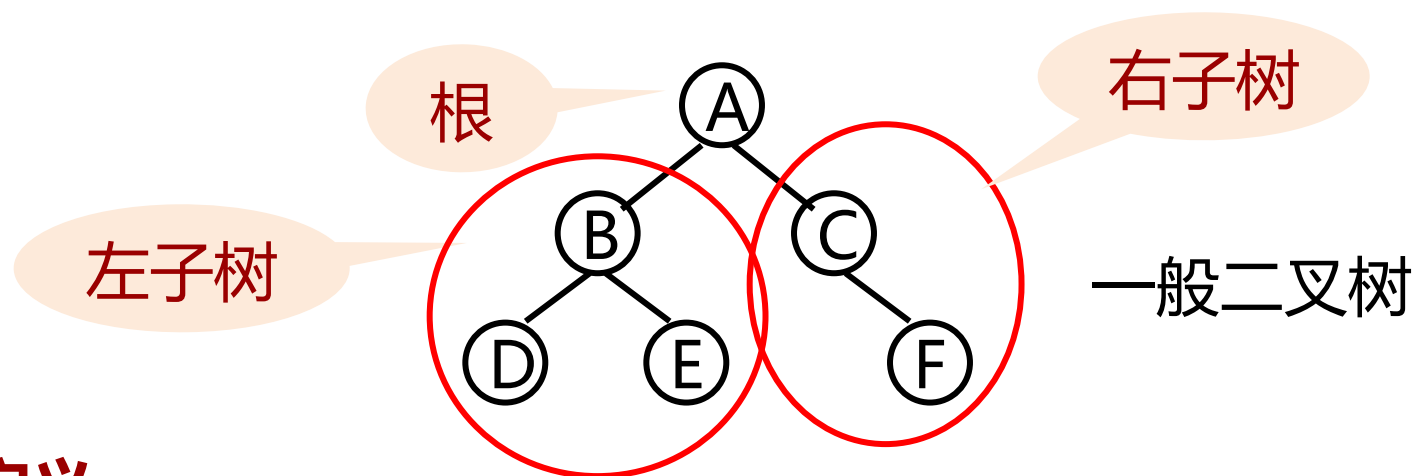
### 5-4-1 二叉树的定义



### 1. 二叉树的定义

**二叉树**是  $n(n \geq 0)$  个结点的有限集，它或者是**空集**，或者是由一个**根**和称为**左、右子树**的两个互不相交的**二叉树**组成。

$\phi$   
空二叉树



二叉树是一个**递归定义**。

树的子树次序不作规定，二叉树的两个子树有**左、右之分**。

树中结点的度没有限制，二叉树中结点的度只能取 **0、1、2**。



## 2. 二叉树的基本形态

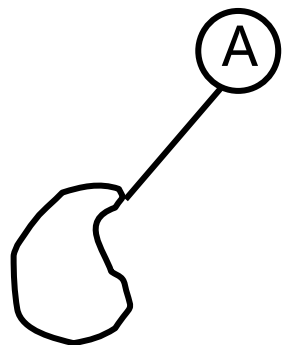
根据定义，二叉树通常具有 5 种基本形态：

$\emptyset$

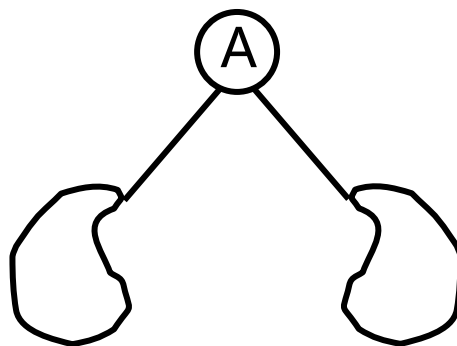
空二叉树

Ⓐ

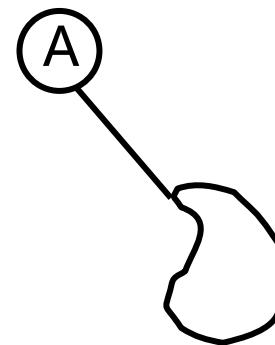
仅有根结点的二叉树



右子树为空的二叉树



左、右子树均非空的二叉树



左子树为空的二叉树

## 5.4 二叉树的逻辑结构

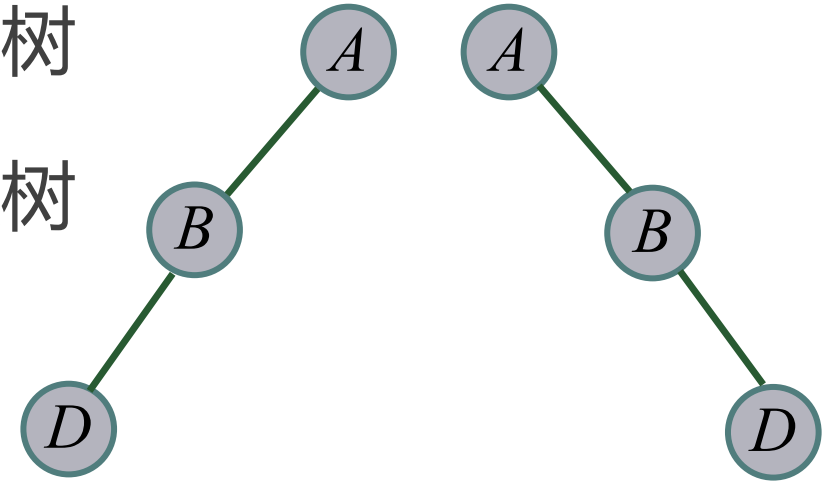
### 5-4-1 二叉树的定义

#### 3. 斜树

📌 左斜树：所有结点都只有左子树的二叉树

📌 右斜树：所有结点都只有右子树的二叉树

📌 斜树：左斜树和右斜树的统称



🕒 斜树有什么特点呢？

(1) 每一层只有一个结点

(2) 结点个数与其深度相同

斜树是树结构的特例，是从树结构退化成了线性结构

## 5.4 二叉树的逻辑结构

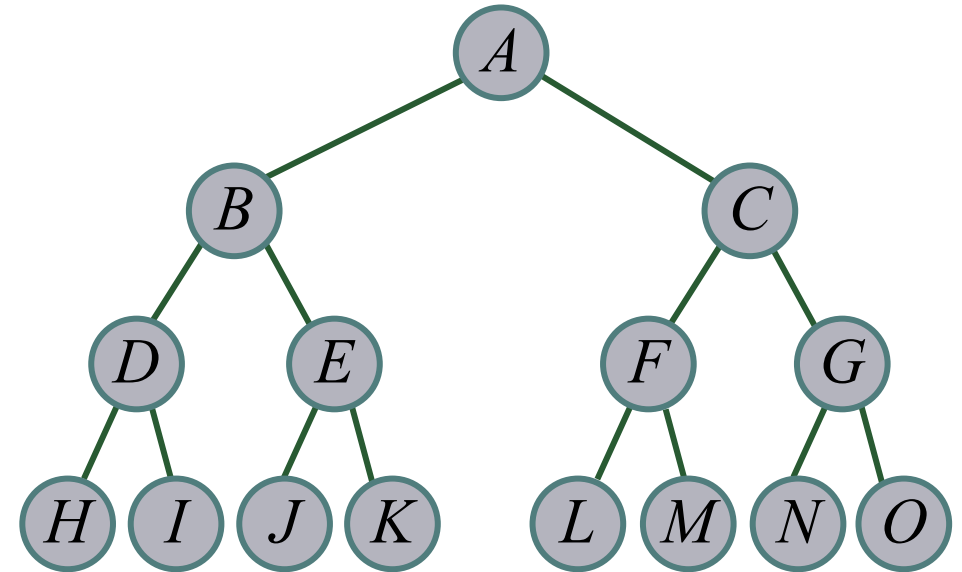
### 5-4-1 二叉树的定义

#### 4. 满二叉树

✚ 满二叉树：所有分支结点都存在左子树和右子树，并且所有叶子都在同一层上的二叉树

🕒 满二叉树有什么特点呢？

- (1) 叶子只能出现在最下一层
- (2) 只有度为 0 和度为 2 的结点
- (3) 在同样深度的二叉树中**结点**个数最多
- (4) 在同样深度的二叉树中**叶子结点**个数最多



满二叉树是树结构的特例，是最**丰满**的二叉树

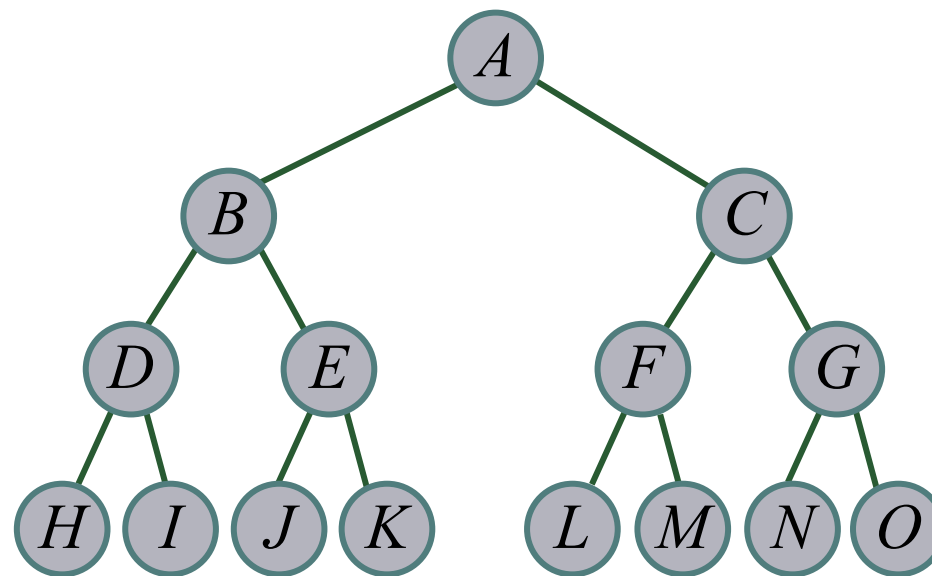
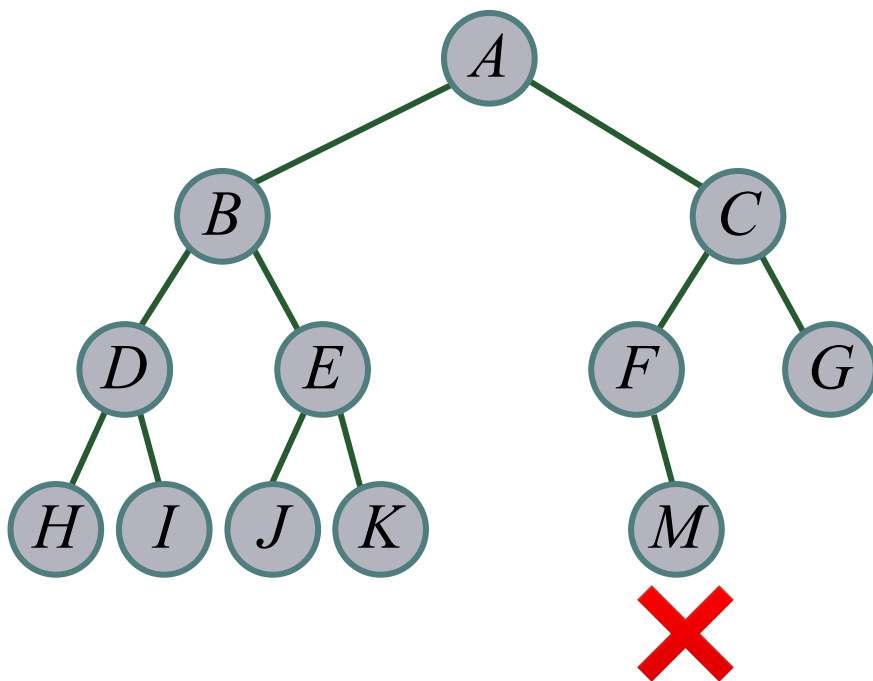
## 5.4 二叉树的逻辑结构

### 5-4-1 二叉树的定义



#### 5. 完全二叉树

✦ 完全二叉树：在满二叉树中，从最后一个结点开始，连续去掉任意个结点得到的二叉树





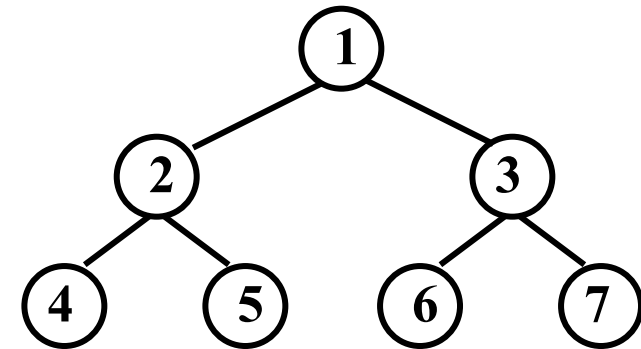
## 5.4 二叉树的逻辑结构

### 5-4-1 二叉树的定义

#### 5. 完全二叉树

✦ **完全二叉树**：深度为  $k$  的，有  $n$  个结点的二叉树，当且仅当其每一个结点都与深度为  $k$  的**满二叉树**中编号从 1 至  $n$  的结点——对应时，称为完全二叉树。

对满二叉树的结点进行连续编号，从根结点起，自上而下，自左至右， $1、2、3、……、2^k-1$ 。



满二叉树

## 5.4 二叉树的逻辑结构

### 5-4-1 二叉树的定义

#### 5. 完全二叉树

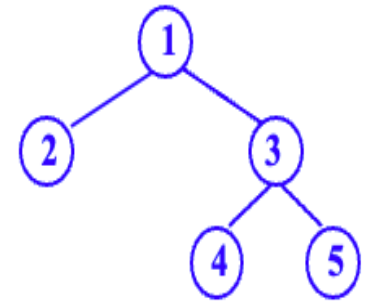
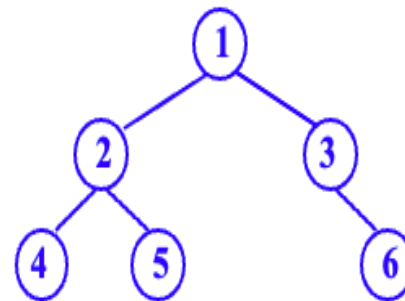
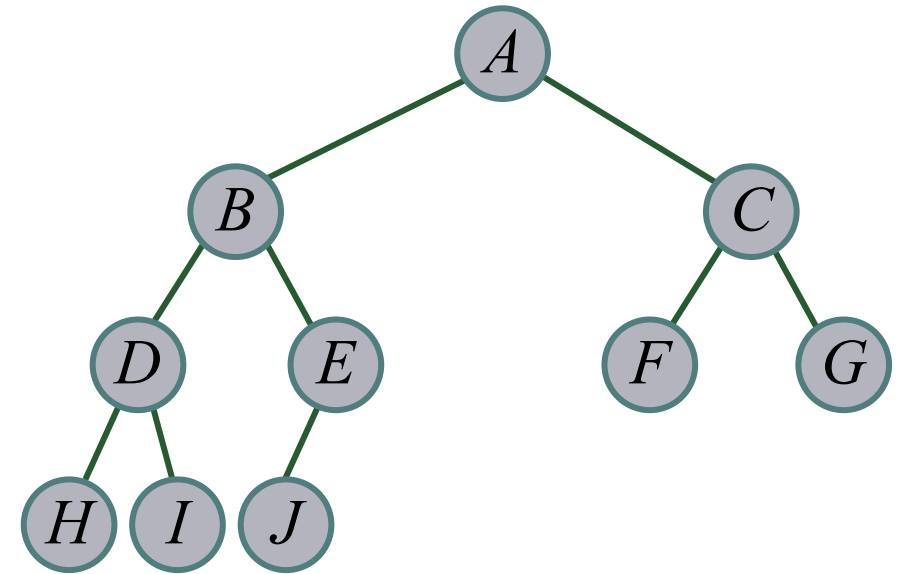
 完全二叉树有什么特点呢？

(1) 叶子结点只能出现在**最下两层**且最下层的叶子结点都集中在二叉树的左面

(2) 完全二叉树中如果有度为 1 的结点，只可能有一个，且该结点只有**左孩子**

(3) 深度为  $k$  的完全二叉树在  $k-1$  层上一定是**满二叉树**

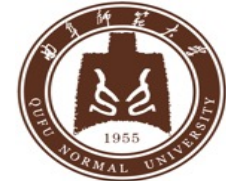
(4) 在同样结点个数的二叉树中，完全二叉树的**深度最小**



非完全二叉树

## 5.4 二叉树的逻辑结构

### 5-4-2 二叉树的基本性质



1.性质 5-1：在一棵二叉树中，如果叶子结点数为  $n_0$ ，度为 2 的结点数为  $n_2$ ，则有： $n_0 = n_2 + 1$

**证明：**(1) 已知，终端结点数为  $n_0$ ，度为 2 的结点数为  $n_2$ ，  
设度为 1 的结点数为  $n_1$ ，  
由于二叉树中的所有结点的度只能为 0、1、2，  
故二叉树的结点总数为  $n = n_0 + n_1 + n_2$ ；  
(2) 除根结点外，其它结点都有一个分支进入，  
设  $B$  为分支总数，故  $n = B + 1$ ，  
由于这些分支均是由度为 1 或 2 的结点引出的，  
所以有  $B = n_1 + 2n_2$ ，故  $n = n_1 + 2n_2 + 1$ ，

由 (1) 和 (2)，可得  $n_0 + n_1 + n_2 = n_1 + 2n_2 + 1$ ，故有  $n_0 = n_2 + 1$ 。



#### 2. 性质 5-2：二叉树的第 $i$ 层上最多有 $2^{i-1}$ 个结点 ( $i \geq 1$ )

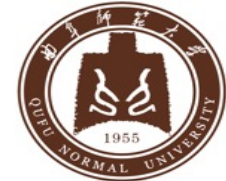
证明：采用归纳法证明。

当  $i = 1$  时，只有一个根结点，而  $2^{i-1} = 2^0 = 1$ ，结论成立。

假设  $i = k$  时结论成立，即第  $k$  层上最多有  $2^{k-1}$  个结点。

考虑  $i = k+1$  时的情形。由于第  $k+1$  层上的结点是第  $k$  层上结点的孩子，而二叉树中每个结点最多有两个孩子，故在第  $k+1$  层上的最大结点个数有  $2 \times 2^{k-1} = 2^k$  个结点，则在  $i = k+1$  时结论也成立。

由此，结论成立。



### 3. 性质 5-3：一棵深度为 $k$ 的二叉树中，最多有 $2^k-1$ 个结点

证明：设深度为  $k$  的二叉树中结点个数最多为  $n$ ，则

$$n = \sum_{i=1}^k (\text{第} i \text{层上结点的最大个数}) = \sum_{i=1}^k 2^{i-1} = 2^k - 1$$

深度为  $k$  且具有  $2^k-1$  个结点的二叉树一定是**满二叉树**

## 5.4 二叉树的逻辑结构

### 5-4-2 二叉树的基本性质



4. 性质 5-4：具有  $n$  个结点的完全二叉树的深度为  $\lfloor \log_2 n \rfloor + 1$

证明：设具有  $n$  个结点的完全二叉树的深度为  $k$ ，则

因为，

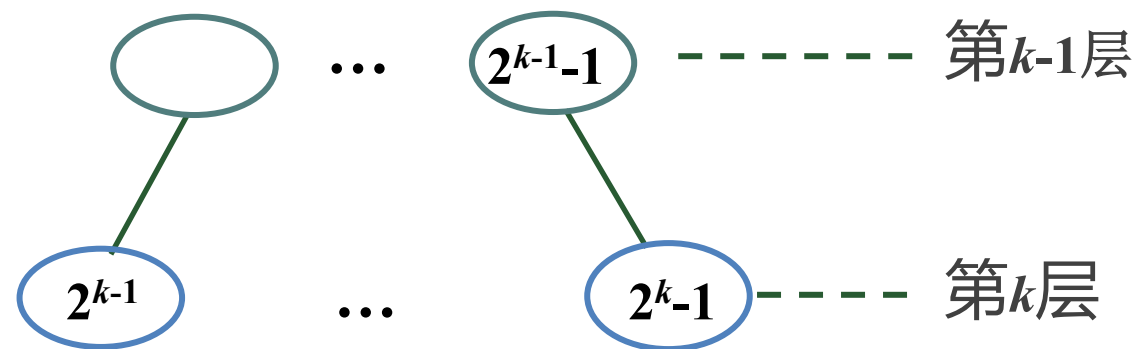
$k-1$  层满二叉树结点数  $<$   $k$  层完全二叉树结点数  $\leq$   $k$  层满二叉树结点数

故  $2^{k-1}-1 < n \leq 2^k-1$

$$2^{k-1} \leq n < 2^k$$

对不等式取对数，有： $k-1 \leq \log_2 n < k$  即： $\log_2 n < k \leq \log_2 n + 1$

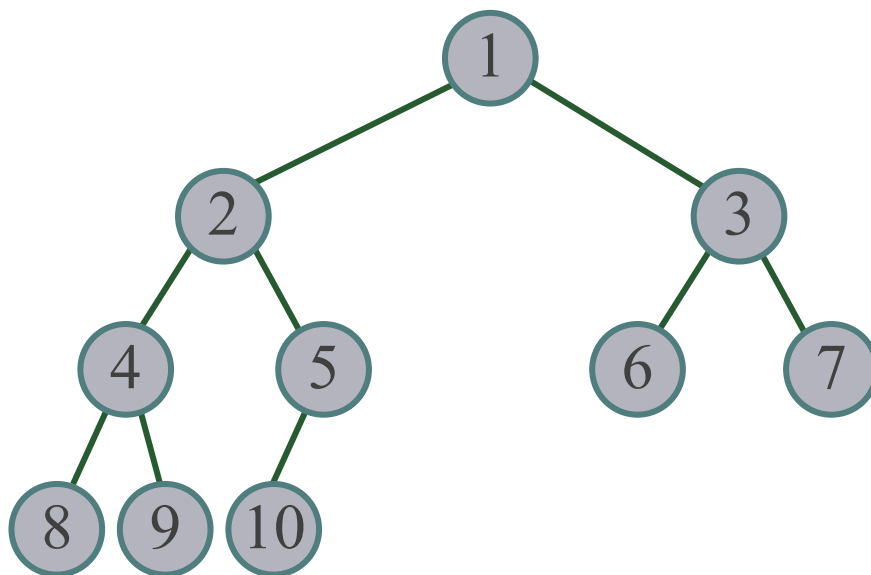
由于  $k$  是整数，故必有  $k = \lfloor \log_2 n \rfloor + 1$



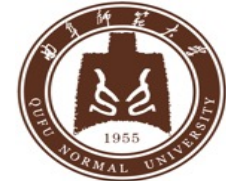


5. 性质 5-5：对一棵具有  $n$  个结点的完全二叉树中从 1 开始按层序编号，对于任意的序号为  $i$  ( $1 \leq i \leq n$ ) 的结点（简称结点  $i$ ），有：

- (1) 如果  $i > 1$ ，则结点  $i$  的**双亲**结点的序号为  $\lfloor i/2 \rfloor$ ，否则结点  $i$  无双亲结点
- (2) 如果  $2i \leq n$ ，则结点  $i$  的**左孩子**的序号为  $2i$ ，否则结点  $i$  无左孩子
- (3) 如果  $2i+1 \leq n$ ，则结点  $i$  的**右孩子**的序号为  $2i+1$ ，否则结点  $i$  无右孩子







## 思考

1. 若一棵完全二叉树的结点总数为500个，请问叶子结点数为多少？
2. 若一棵树的度为4，其中度为1，2，3，4的结点数分别为3，3，1，1个，请问叶子结点数为多少？

$$n = n_0 + n_1 + n_2 + n_3 + n_4 \quad (\text{结点数} = \text{不同度的结点个数之和})$$

$$n = B + 1 \quad (\text{结点数} = \text{分支数} + 1)$$

$$B = 0 \times n_0 + 1 \times n_1 + 2 \times n_2 + 3 \times n_3 + 4 \times n_4$$

## 5.4 二叉树的逻辑结构

### 5-4-3 二叉树的抽象数据类型定义

## 5.4 二叉树的逻辑结构

### 5-4-3 二叉树的抽象数据类型定义



#### 二叉树的抽象数据类型定义

ADT BiTree

DataModel

二叉树由一个根结点和两棵互不相交的左右子树构成，结点具有层次关系

Operation

**InitBiTree**：初始化一棵空的二叉树

**CreatBiTree**：建立一棵二叉树

**DestroyBiTree**：销毁一棵二叉树

**PreOrder**：前序遍历二叉树

**InOrder**：中序遍历二叉树

**PostOrder**：后序遍历二叉树

**LeverOrder**：层序遍历二叉树

简单起见，只讨论二叉树的遍历

endADT



## 二叉树的遍历

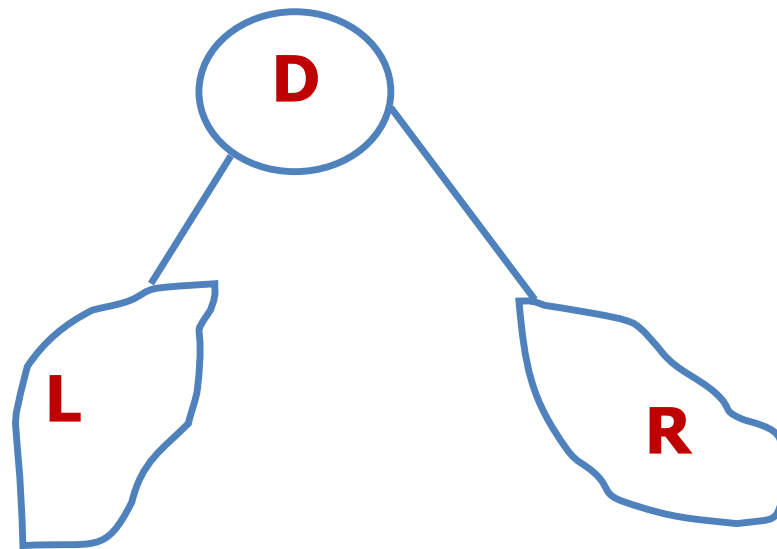
按某条搜索路径巡访树中每个结点，使得**每个结点均被访问一次，而且仅被访问一次**。

遍历对于线性结构而言，比较简单，而二叉树是一种**非线性结构**，因而需要找一种规律**以使二叉树的结点能排列到一个线性队列上来**。

**对二叉树而言，可以有三条搜索路径：**

- 1、先上后下的按层次遍历
- 2、先**左**（子树）后**右**（子树）的遍历
- 3、先**右**（子树）后**左**（子树）的遍历

**DLR, LDR, LRD**, DRL, RDL, RLD



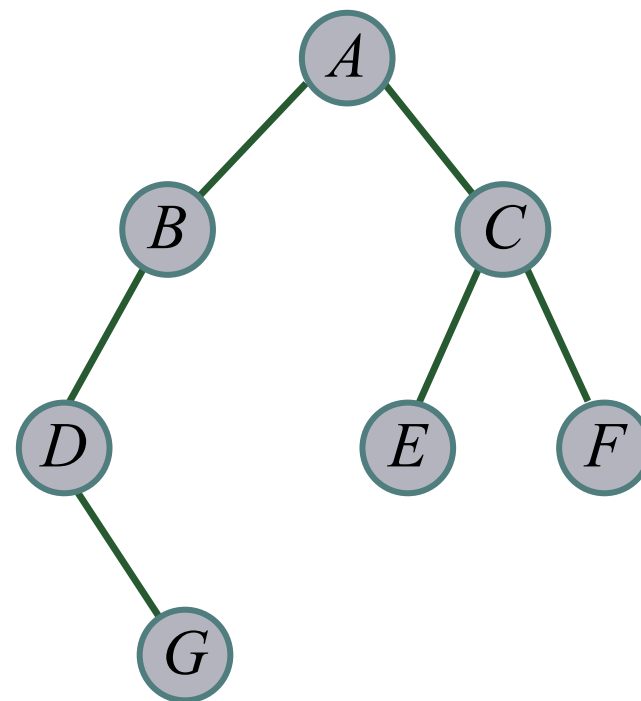


### 二叉树的前序（先根）遍历

若二叉树为空，则空操作返回；否则：

- (1) 访问根结点
- (2) 前序遍历根结点的左子树
- (3) 前序遍历根结点的右子树

前序遍历序列：***A** B D G C E F*



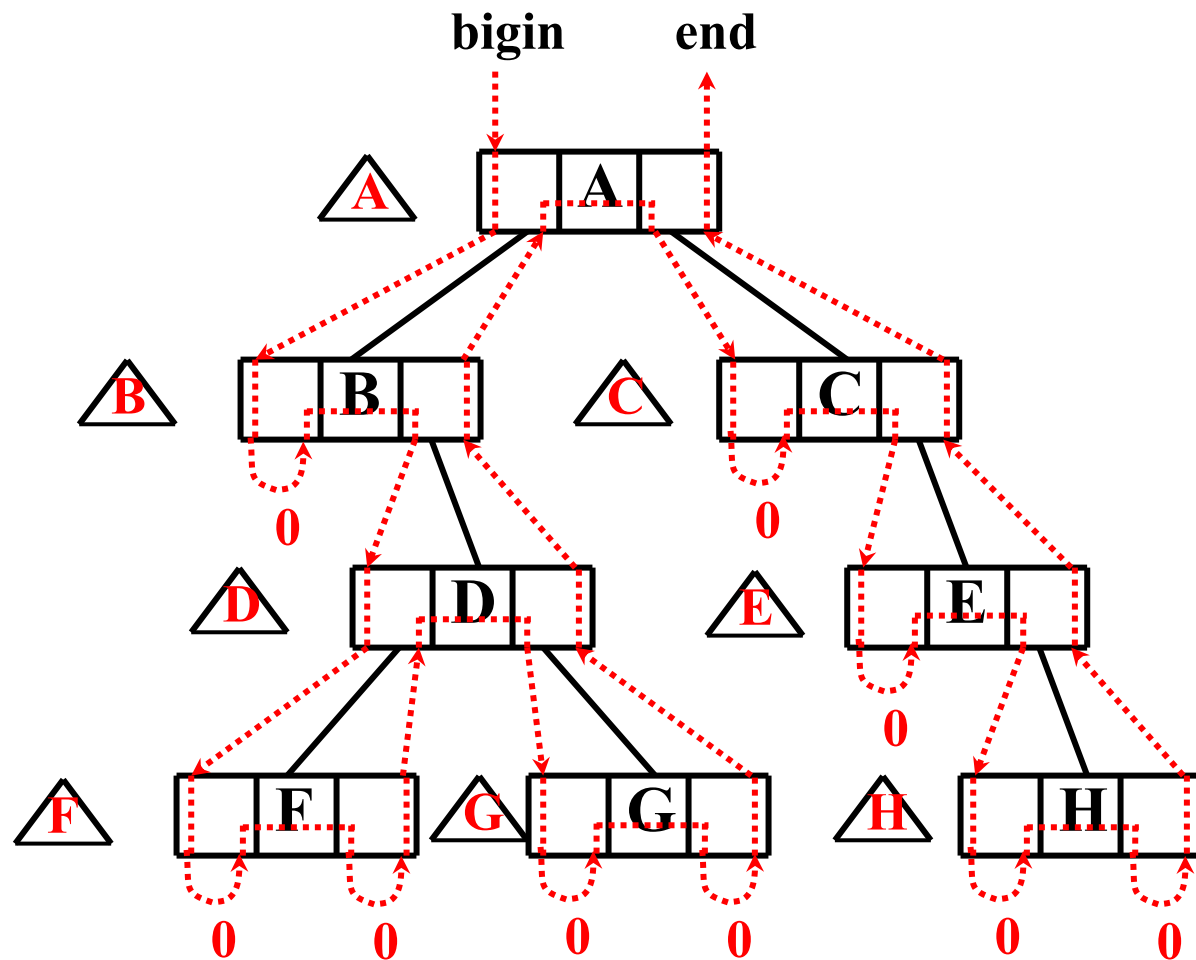
## 5.4 二叉树的逻辑结构

### 5-4-3 二叉树的抽象数据类型定义



#### 二叉树的前序（先根）遍历

先序遍历顺序: **A B D F G C E H**



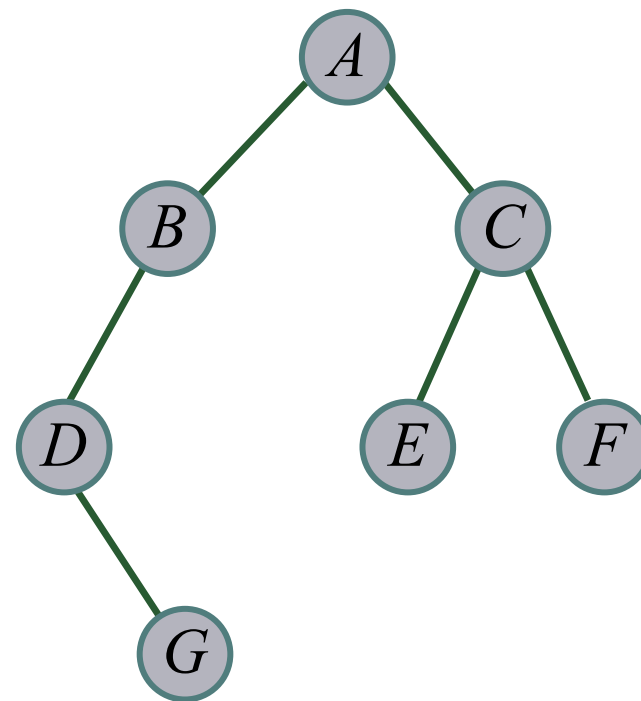


### 二叉树的后序（后根）遍历

若二叉树为空，则空操作返回；否则：

- (1) 后序遍历根结点的左子树
- (2) 后序遍历根结点的右子树
- (3) 访问根结点

后序遍历序列：***G D B E F C A***



## 5.4 二叉树的逻辑结构

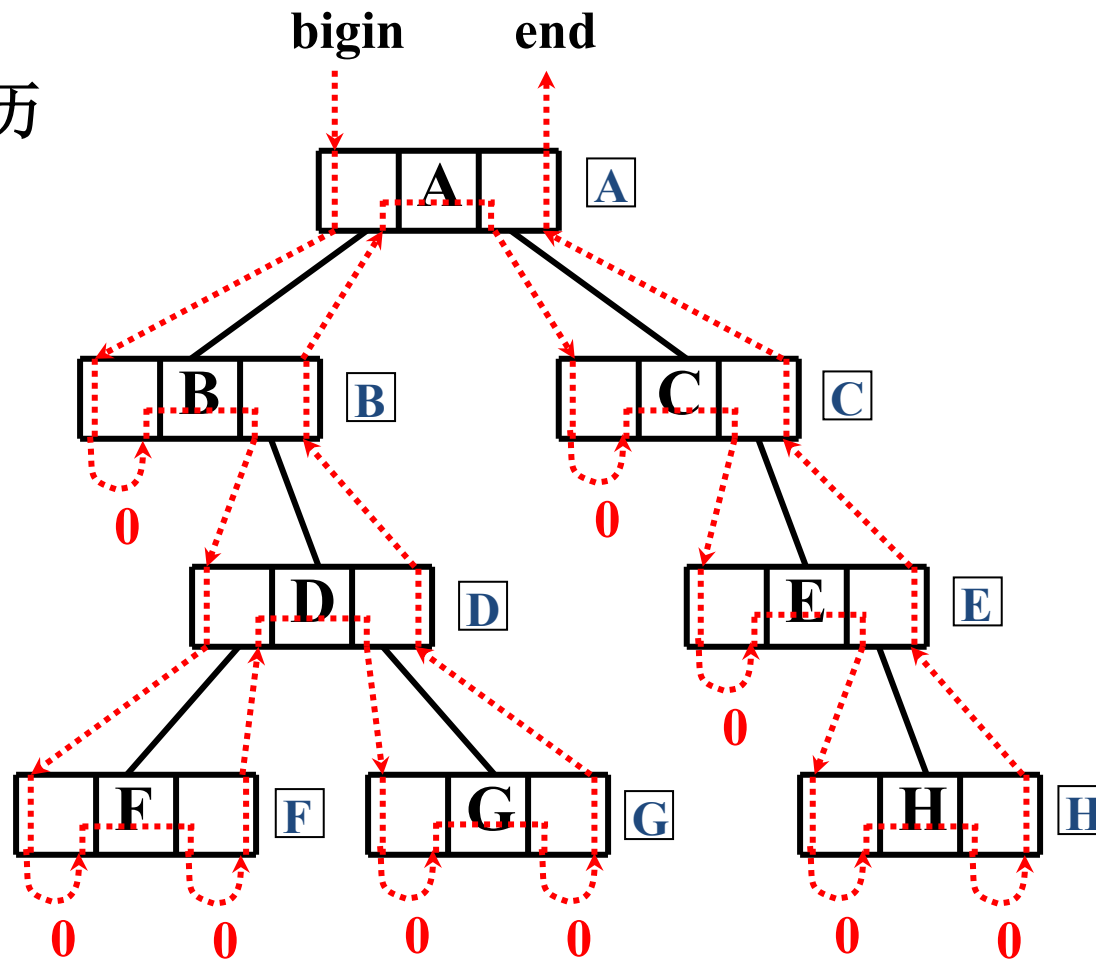
### 5-4-3 二叉树的抽象数据类型定义



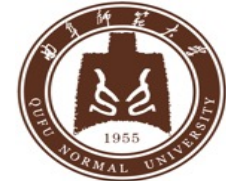
## 二叉树的后序（后根）遍历

后序遍历

后序遍历顺序: **F G D B H E C A**





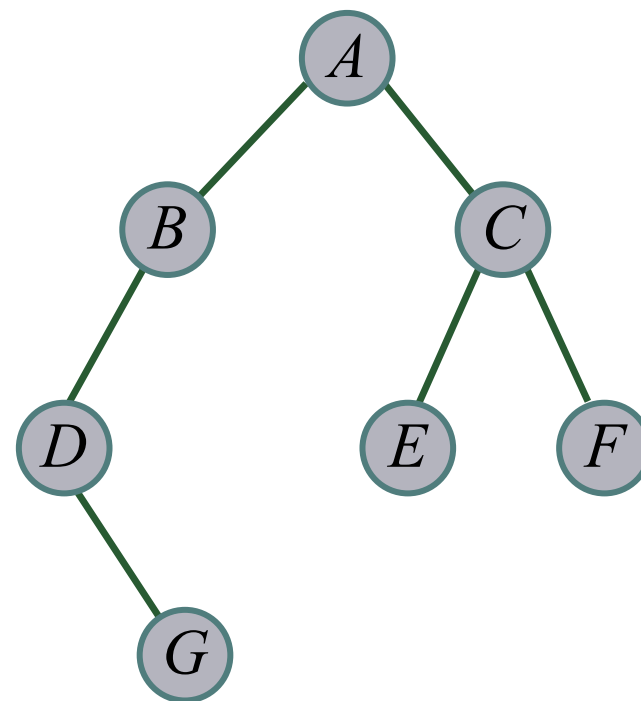


### 二叉树的中序（中根）遍历

若二叉树为空，则空操作返回；否则：

- (1) 中序遍历根结点的左子树
- (2) 访问根结点
- (3) 中序遍历根结点的右子树

中序遍历序列：*D G B A E C F*



## 5.4 二叉树的逻辑结构

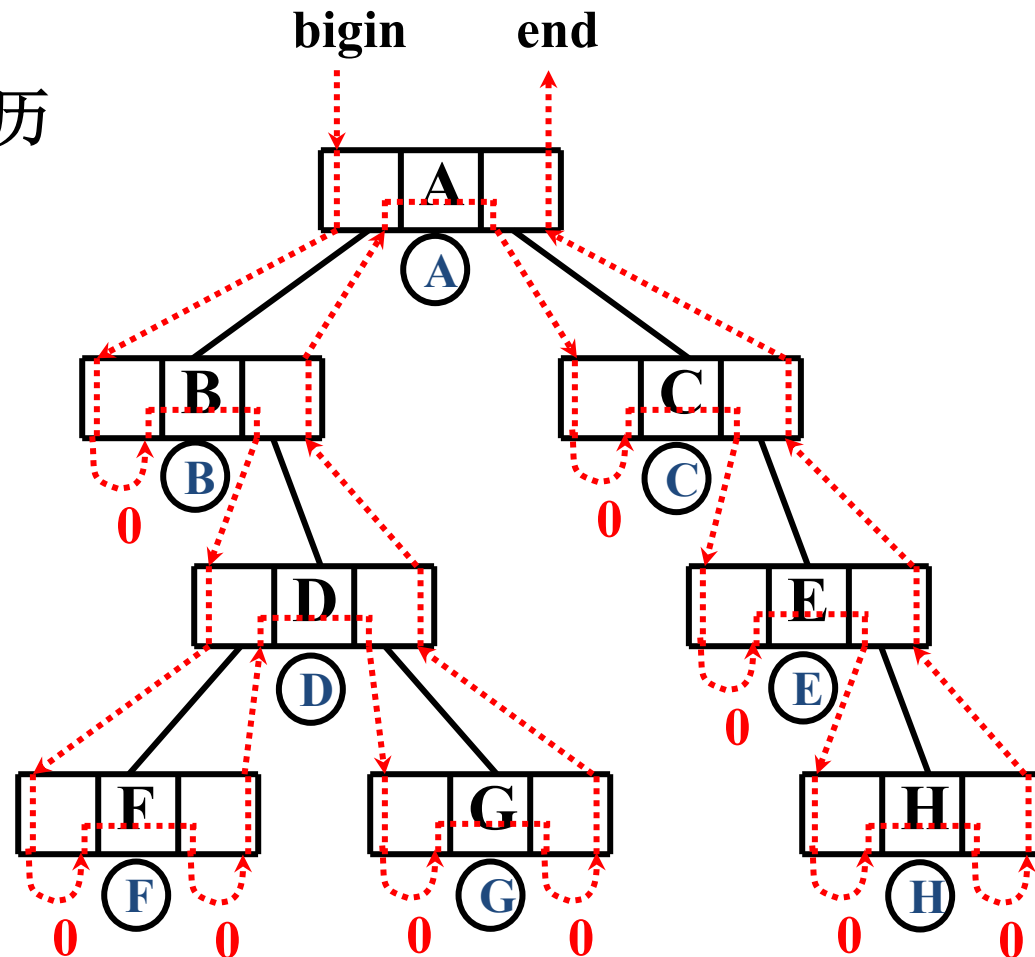
### 5-4-3 二叉树的抽象数据类型定义



## 二叉树的中序（中根）遍历

中序遍历

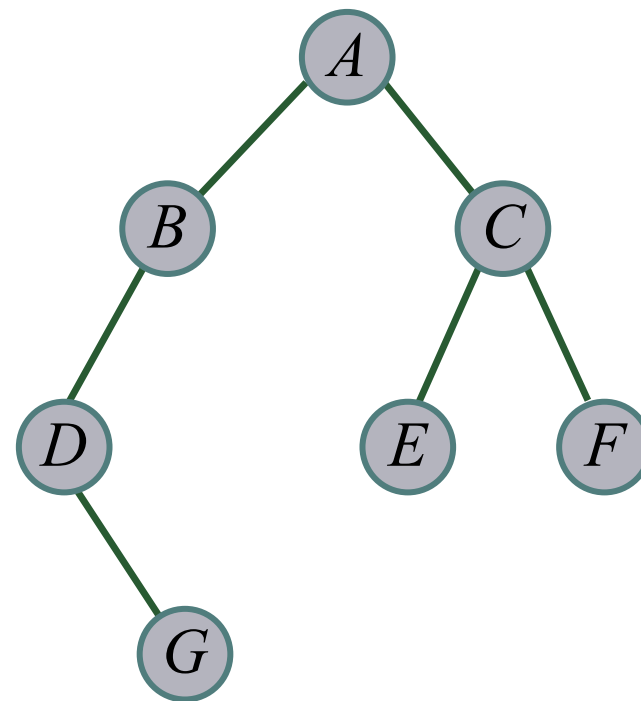
中序遍历顺序: **B F D G A C E H**





### 二叉树的层序遍历

从二叉树的根结点开始，从上至下逐层遍历，在同一层中，则按从左到右的顺序对结点逐个访问

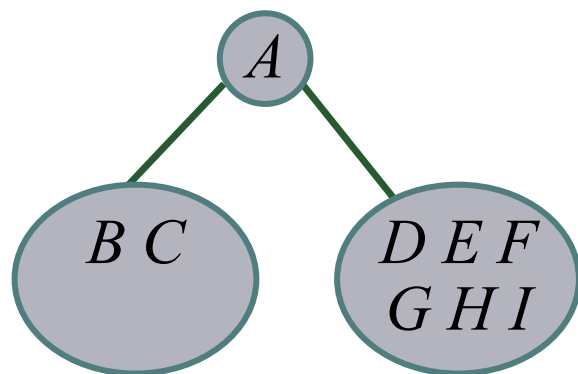


层序遍历序列：*A B C D E F G*



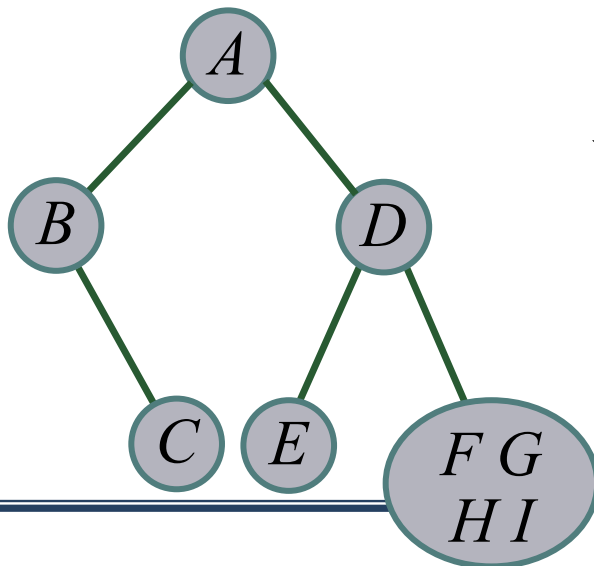
## 二叉树的遍历

🕒 若已知一棵二叉树的前序序列和中序序列，能否唯一确定这棵二叉树呢？



前序: A B C D E F G H I  
中序: B C A E D G H F I

前序: B C  
中序: B C

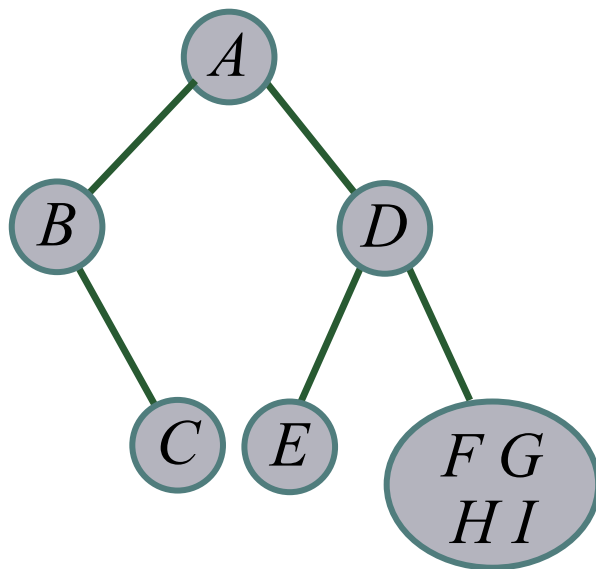


前序: D E F G H I  
中序: E D G H F I

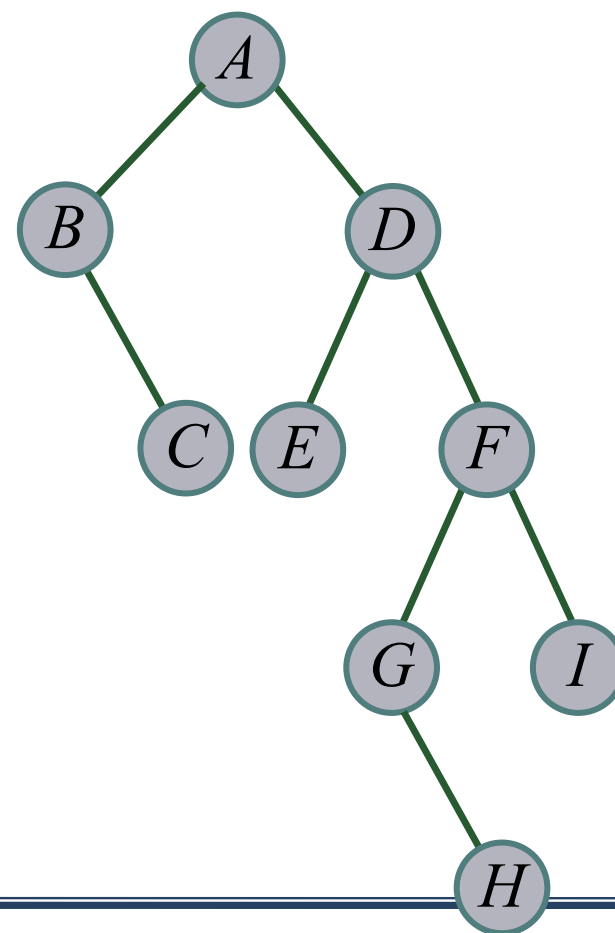


### 二叉树的遍历

🕒 若已知一棵二叉树的前序序列和中序序列，能否唯一确定这棵二叉树呢？



前序: **F** G H I  
中序: G H **F** I





## 思考

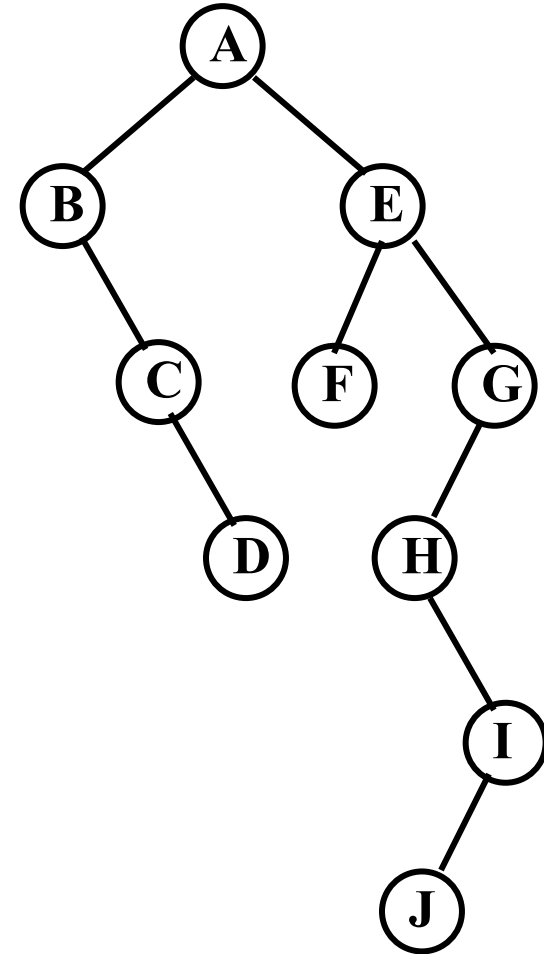
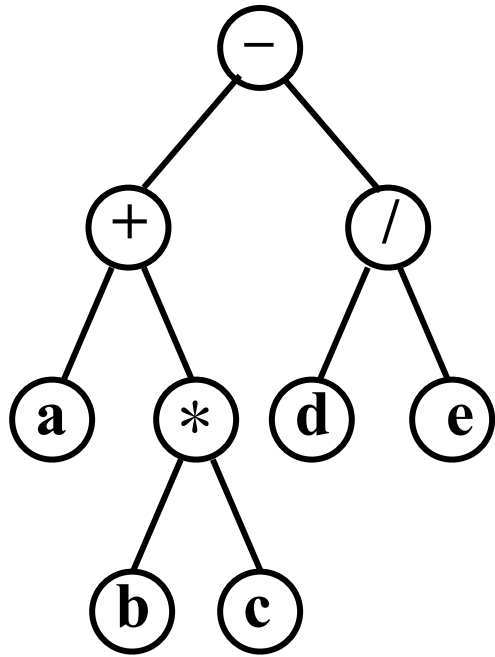
1. 若已知一棵二叉树先序遍历序列为：**BEFCGDH**，中序遍历序列为：**FEBGCHD**，则后序遍历序列为\_\_\_\_\_。
2. 若已知一棵二叉树先序遍历序列为：**CBDFAE**，中序遍历序列为：**BFDCEA**，则后序遍历序列为\_\_\_\_\_。
3. 若已知一棵二叉树后序遍历序列为：**TRVSWU**，中序遍历序列为：**VRTUSW**，则先序遍历序列为\_\_\_\_\_。

## 小结

1. 熟练掌握二叉树的递归定义和基本性质
2. 理解二叉树的抽象数据类型定义
3. 熟练掌握二叉树的遍历方法（前序、后序、中序、层次）

# 作业

4. 写出如下图中二叉树的先序遍历、后序遍历、中序遍历序列。







*Thank You !*

*Q & A*