数据结构（C语言版）（第2版）

树和二叉树

树和二叉树的定义

主讲教师：汪红松

教学目标

01 OPTION 理解二叉树的定义和术语，二叉树的性质，特殊的二叉树；

02 OPTION 掌握二叉树的存储结构，顺序存储，二叉链表和线索二叉树；

03 OPTION 掌握二叉树的前序、中序、后序、层次遍历方法；

04 OPTION 了解树和森林的定义，树的存储，树、森林与二叉树的转换；

05 OPTION 理解树的应用，哈夫曼树及哈夫曼编码；

06 OPTION 能够写出二叉树的前序、中序、后序、层次遍历，灵活运用遍历算法实现二叉树的操作。

教 学 内 容 Contents

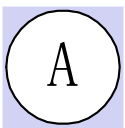
一、树的定义

树（Tree）是n（$n\geq 0$）个结点的有限集，它或为空树（$n=0$）；或为非空树，对于非空树T：

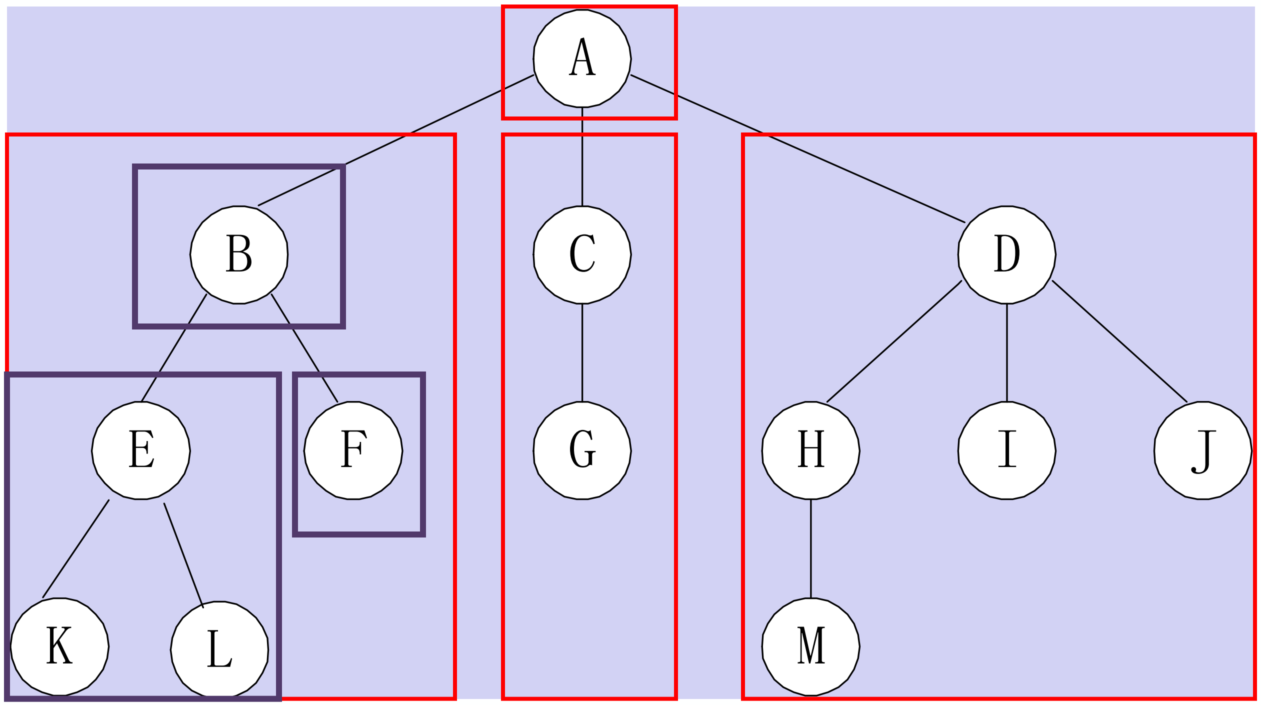
1. 有且仅有一个称之为根的结点；
2. 除根结点以外的其余结点可分为m（$m>0$）个互不相交的有限集$T_1, T_2, \ldots, T_m$，其中每一个集合本身又是一棵树，并且称为根的子树（SubTree）。

一、树的定义

树是n个结点的有限集。



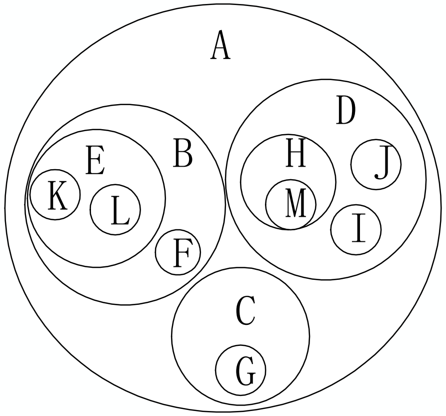
(a)



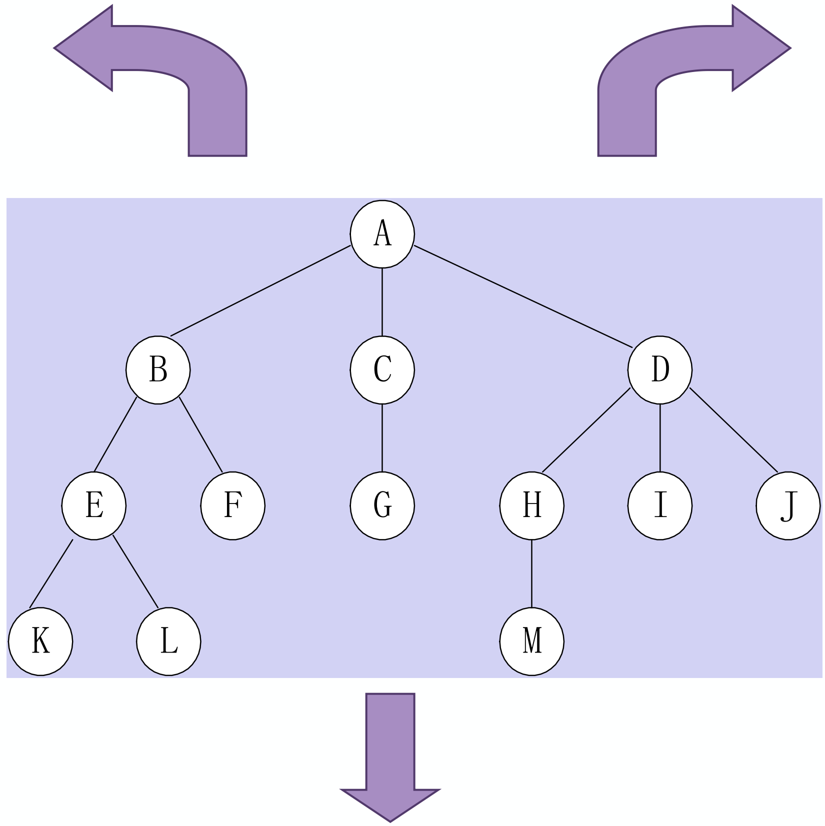
T_1　　T_2　　T_3

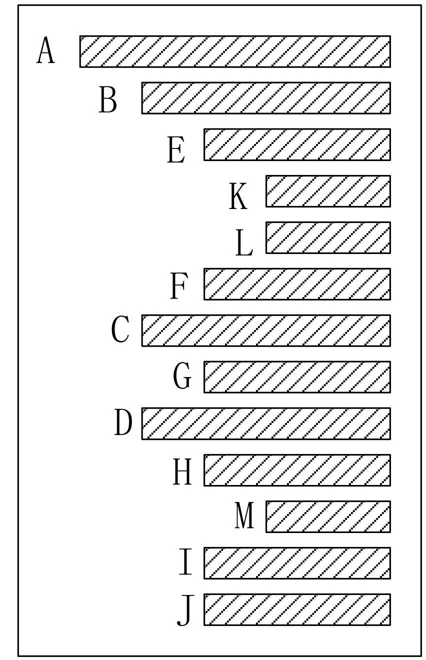
(b)

一、树的定义　　1.树的其它表示方式



(a)嵌套集合





(c)凹入表示

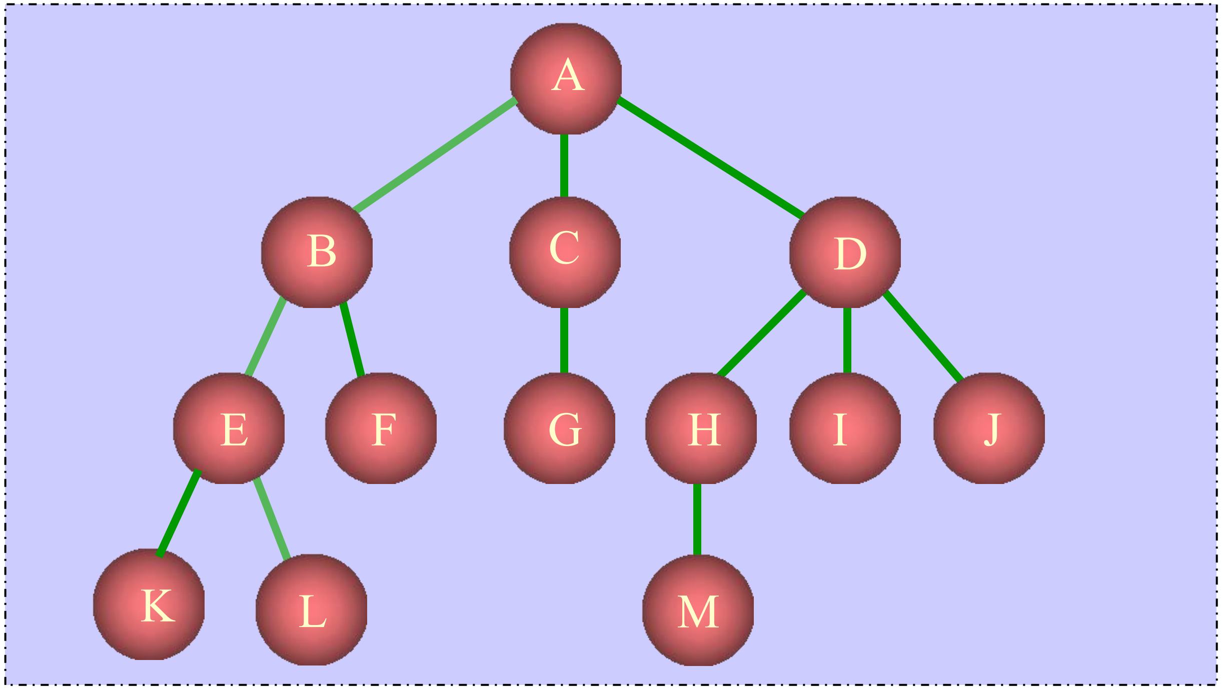
(A(B(E(K,L),F),C(G),D(H(M),I,J)))

(b)广义表

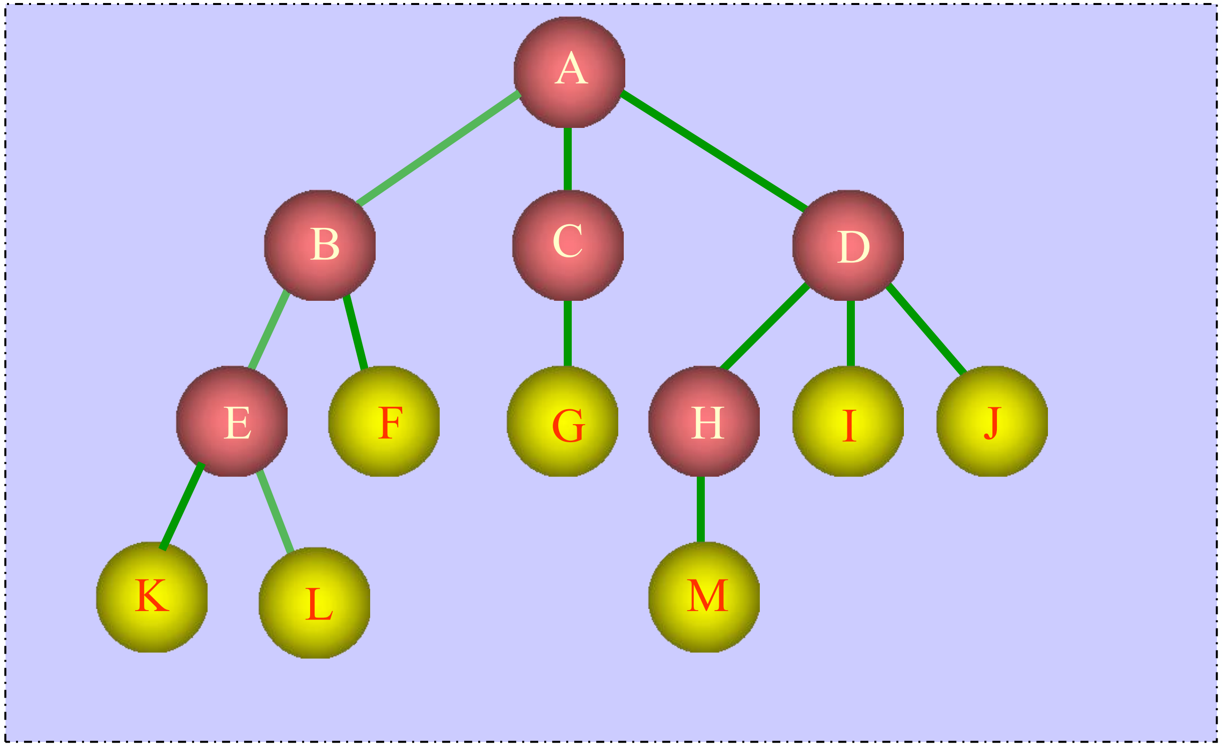
（1）**结点：**树中的一个独立单元。包含一个数据元素及若干指向其子树的分支；

（2）**结点的度：**结点所拥有的子树的个数；

（3）**树的度：**树中各结点度的最大值；

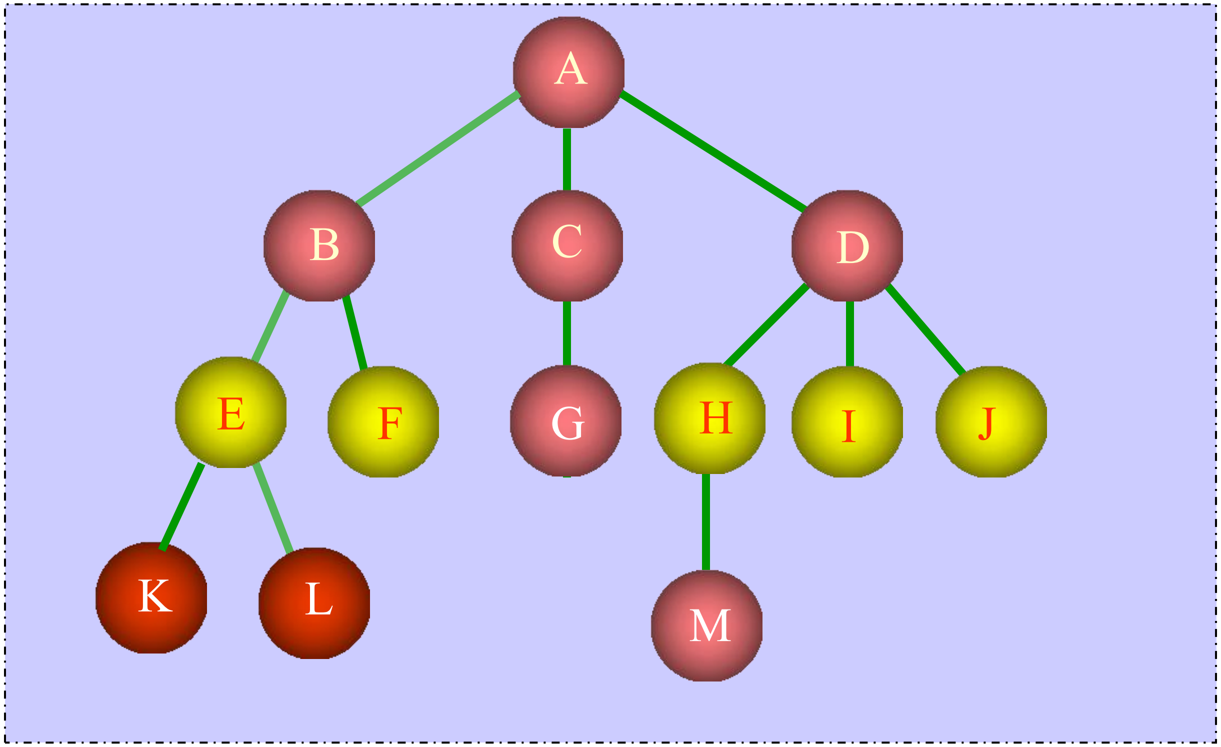


（4）**叶子结点：**度为0的结点，也称为终端结点；
（5）**非终端结点：**度不为0的结点，也称为分支结点；



（6）孩子、双亲：树中某结点子树的根结点称为这个结点的孩子结点，这个结点称为它孩子结点的双亲结点；

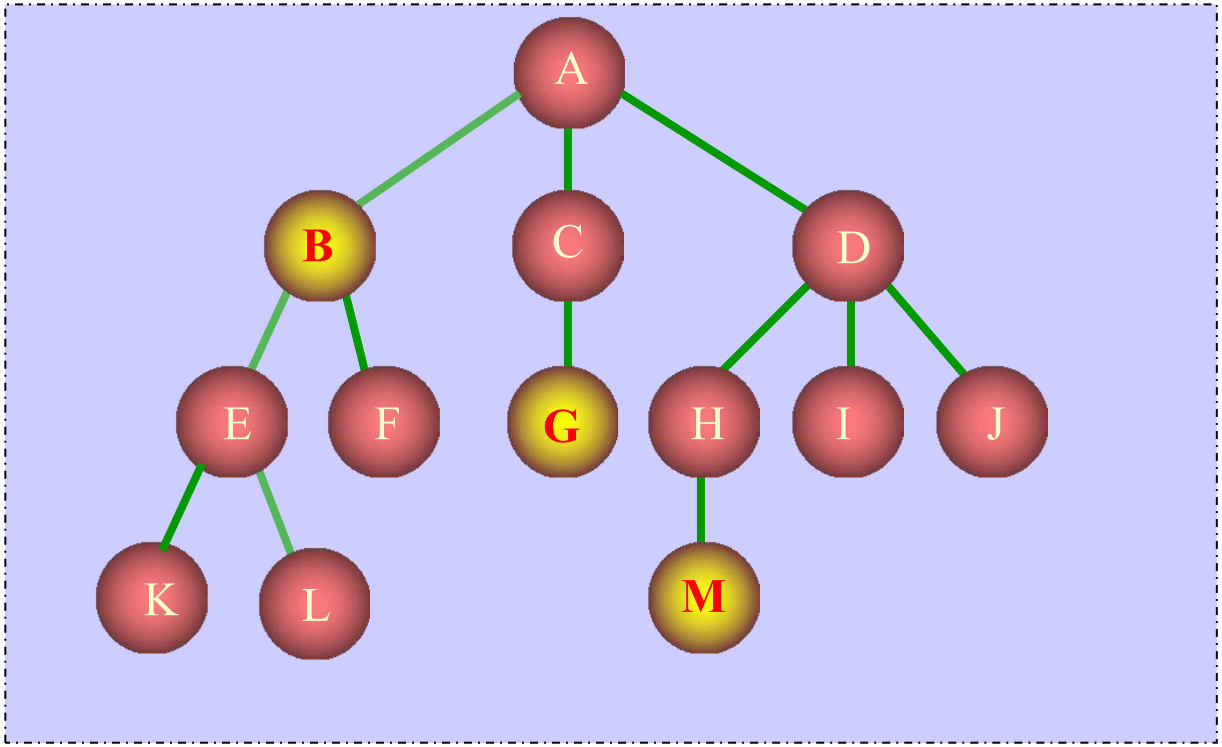
（7）兄弟：具有同一个双亲的孩子结点互称为兄弟；



（8）祖先：从根到该结点所经分支上的所有结点；

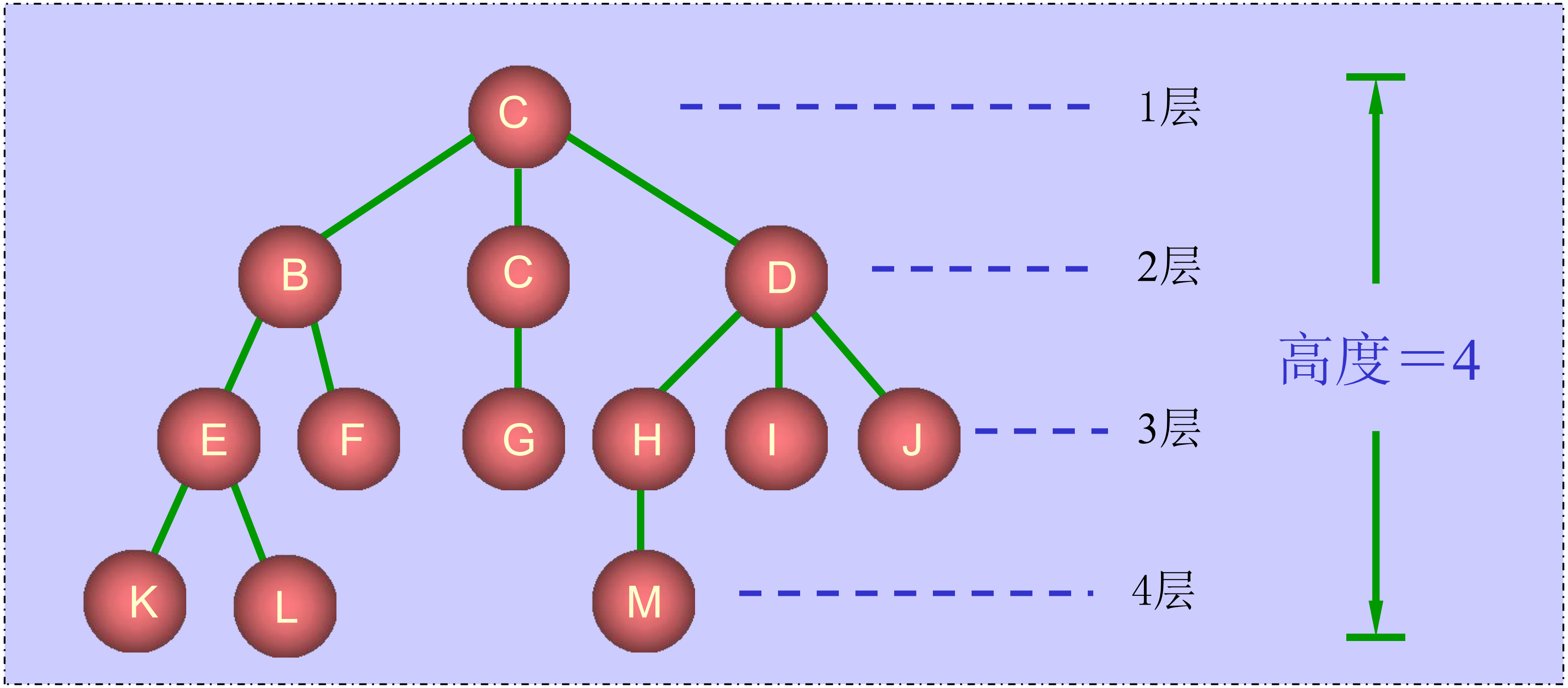
（9）子孙：以某结点为根的子树中的任一结点；

（10）堂兄弟：双亲在同一层的结点互为堂兄弟；

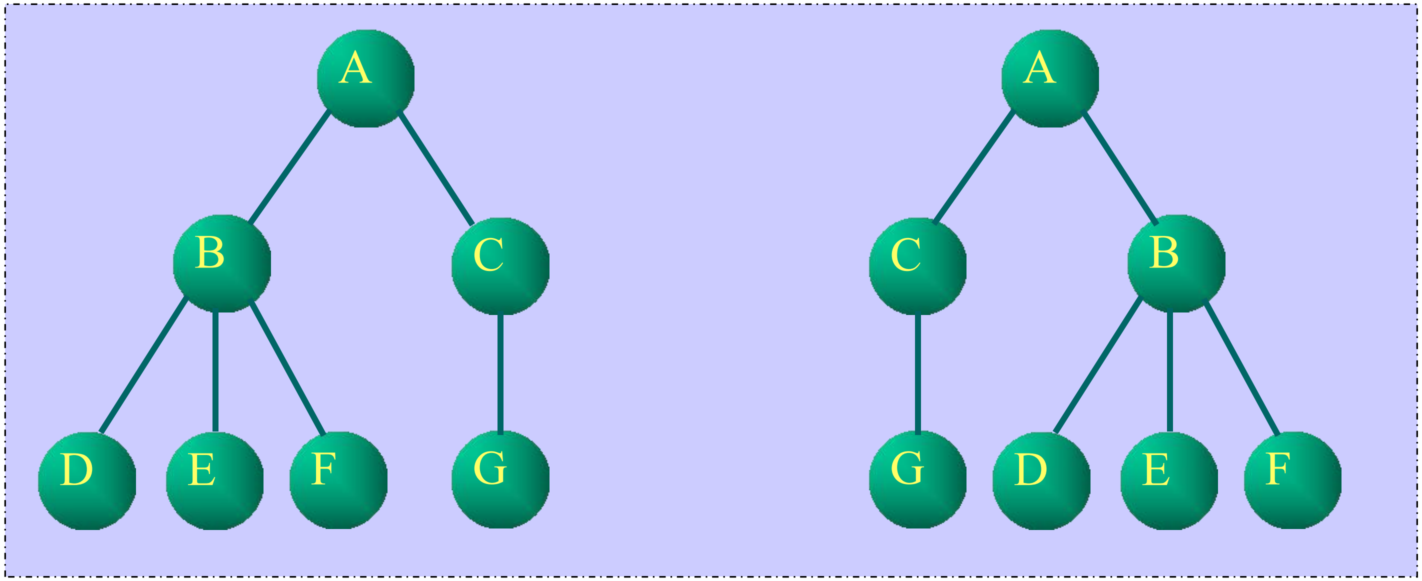


（11）结点所在层次：根结点的层数为1；对其余任何结点，若某结点在第k层，则其孩子结点在第k+1层；

（12）树的深度：树中所有结点的最大层数，也称高度；

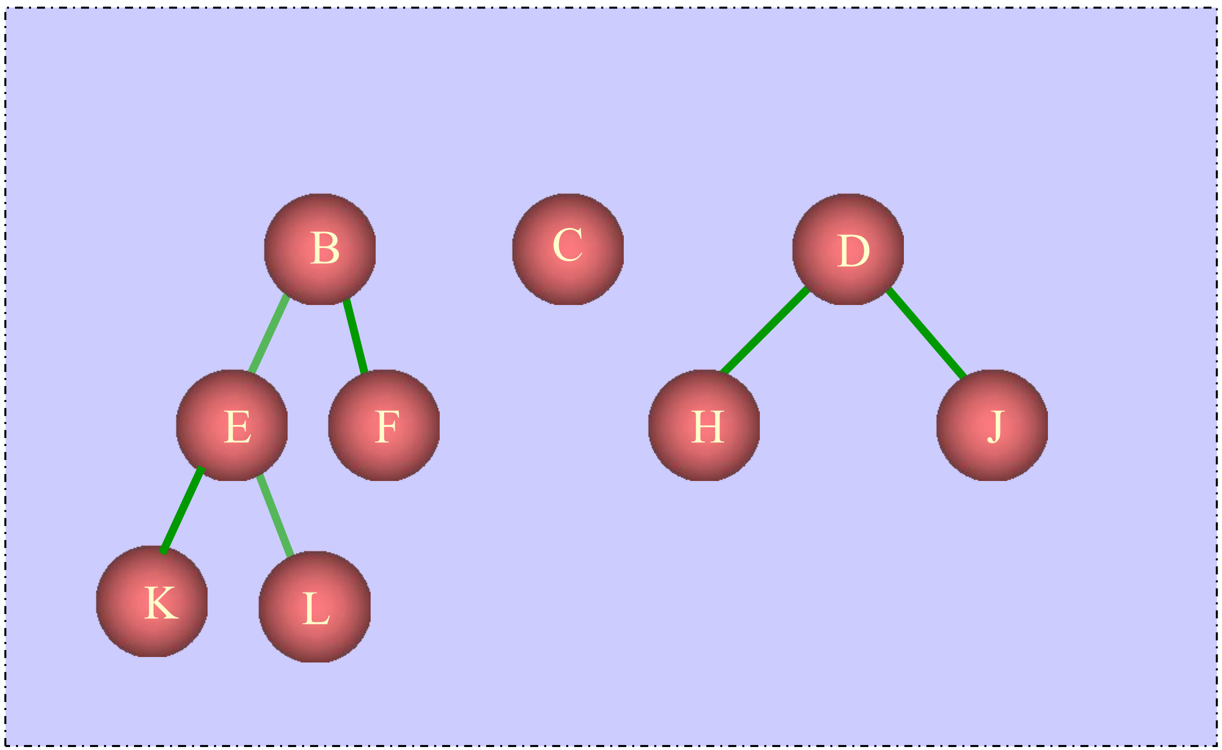


（13）有序树、无序树：如果一棵树中结点的各子树从左到右是有次序的，称这棵树为有序树；反之，称为无序树；

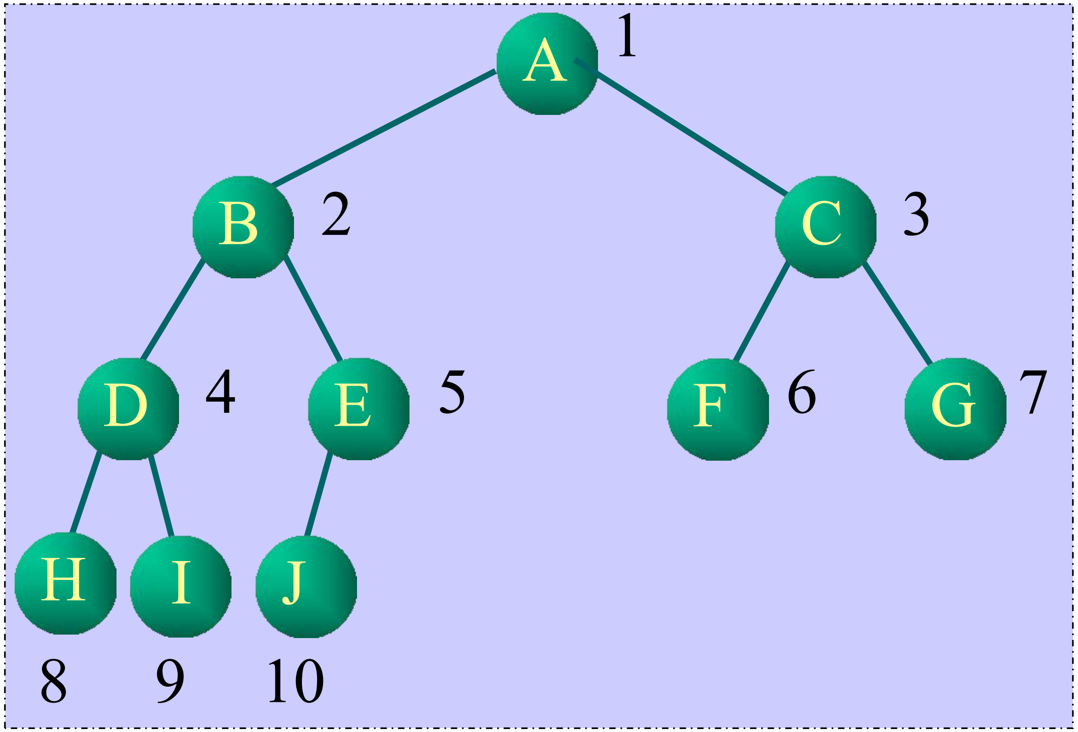


数据结构中讨论的一般都是有序树

（14）**森林**：***m*** **(*m*≥0)棵互不相交的树的集合；**



（15）层序编号：将树中结点按照从上层到下层、同层从左到右的次序依次给他们编以从1开始的连续自然数。



树结构和线性结构的比较

线性结构	树结构
第一个数据元素	根结点（只有一个）
无前驱	无双亲
最后一个数据元素	叶子结点(可以有多个)
无后继	无孩子
其它数据元素	其它结点
一个前驱,一个后继	一个双亲,多个孩子
一对一	一对多

二、二叉树的定义　　1.定义

二叉树（Binary Tree）是n（$n\geq 0$）个结点所构成的集合，它或为空树（$n = 0$）；或为非空树，对于非空树T：

（1）有且仅有一个称之为根的结点；

（2）除根结点以外的其余结点分为两个互不相交的子集T_1和T_2，分别称为T的左子树和右子树，且T_1和T_2本身又都是二叉树。

二、二叉树的定义

普通树（多叉树）若不转化为二叉树，则运算很难实现。

为何要重点研究每结点最多只有两个 “叉” 的树？

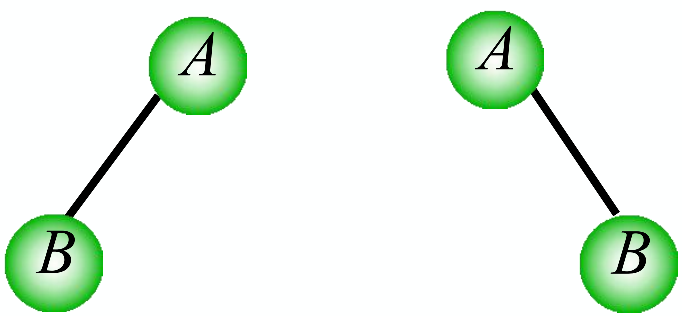
- ✓ 二叉树的结构最简单，规律性最强；
- ✓ 可以证明，所有树都能转为唯一对应的二叉树，不失一般性。

二、二叉树的定义

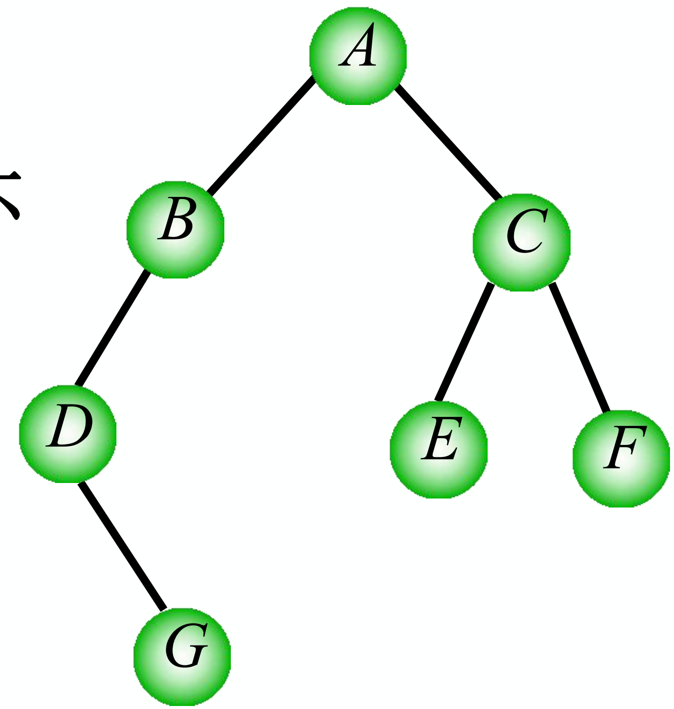
2. 特点

（1）每个结点最多有两棵子树；

（2）二叉树是有序的，其次序不能任意颠倒。



(a)



(b)

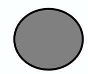
注意：二叉树和树是两种树结构。

二、二叉树的定义

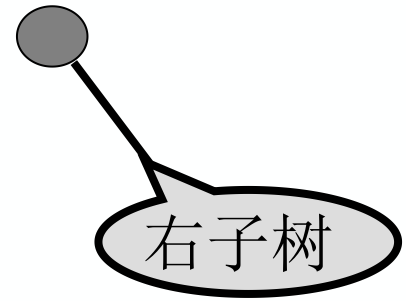
3. 基本形态

Φ

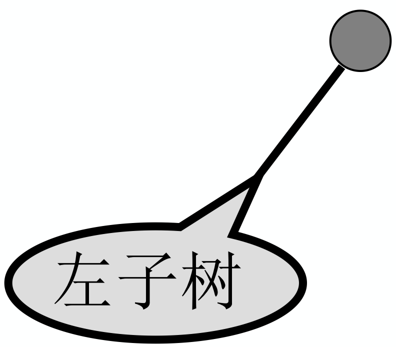
(a)空二叉树



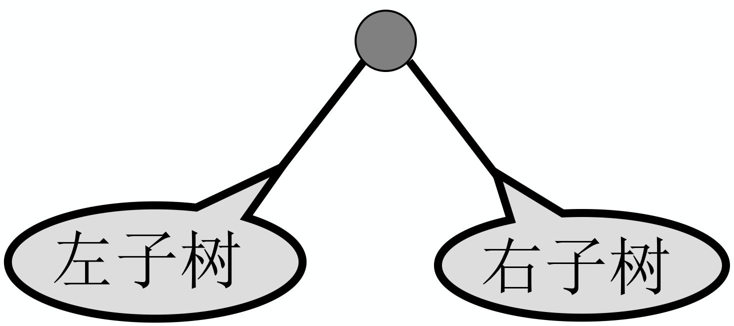
(b)只有一个根结点



(c)根结点只有右子树



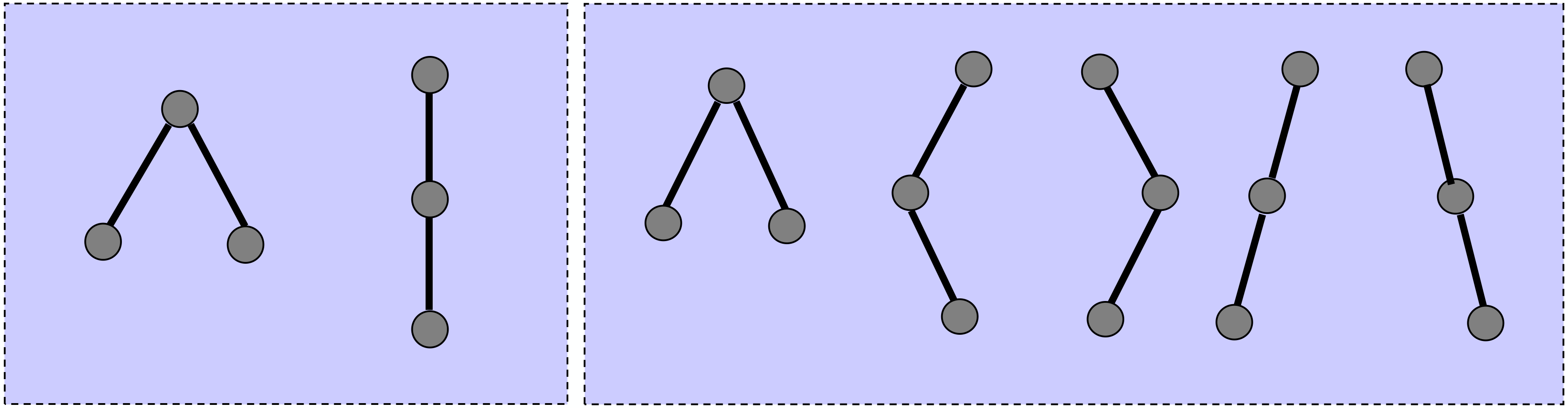
(d)根结点只有左子树



(e)根结点同时有左右子树

练习

具有3个结点的树和具有3个结点的二叉树的形态。



(a) (b)

❖二叉树和树是两种树结构。

三、树的抽象数据类型定义

ADT BinaryTree{

数据对象D: **树是由一个根结点和若干棵子树构成，树中结点具有层次关系及相同数据类型**

数据关系R: 若D=Φ，则R= Φ ；

若D≠Φ，则R= {H}；存在二元关系：

① root 唯一 //关于根的说明

② $D_j \cap D_k = \Phi$ //关于子树不相交的说明

③ …… //关于数据元素的说明

基本操作 P： //至少有15个

}ADT Tree

三、树的抽象数据类型定义

CreateTree(&T,definition)

初始条件；definition给出树T的定义。

操作结果：按definition构造树T。

TreeDepth (T)

初始条件：树T存在。

操作结果：返回T的深度。

InsertChild(&T, p, i, c)

初始条件：树T存在，p指向T中某个结点，非空树c与T不相交。

操作结果：插入c为T中p所指结点的第i棵子树。

TraverseTree(T)

初始条件：树T存在。

操作结果：按某种次序对T的每个结点访问一次。

三、树的抽象数据类型定义 ——二叉树

ADT BinaryTree{

数据对象D: **由一个根结点和两棵互不相交的左右子树构成，结点具有相同数据类型及层次关系**

数据关系R: 若D=Φ，则R= Φ ；

若D≠Φ，则R= {H}；存在二元关系：

① root 唯一 //关于根的说明

② $D_j \cap D_k = \Phi$ //关于子树不相交的说明

③ …… //关于数据元素的说明

④ …… //关于左子树和右子树的说明

基本操作 P： //至少有20个

}ADT BinaryTree

三、树的抽象数据类型定义 ——二叉树

CreateBiTree(&T,definition)

初始条件；definition给出二叉树T的定义。

操作结果：按definition构造二叉树T。

PreOrderTraverse(T)

初始条件：二叉树T存在。

操作结果：先序遍历T，对每个结点访问一次。

InOrderTraverse(T)

初始条件：二叉树T存在。

操作结果：中序遍历T，对每个结点访问一次。

PostOrderTraverse(T)

初始条件：二叉树T存在。

操作结果：后序遍历T，对每个结点访问一次。

小结：

介绍了树的定义及相关的基本术语

介绍了树、二叉树的定义和特点

介绍了树和二叉树的抽象数据类型的定义