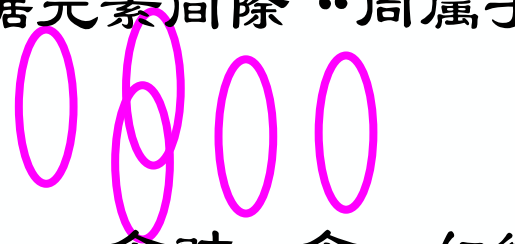
第5章 树和二叉树

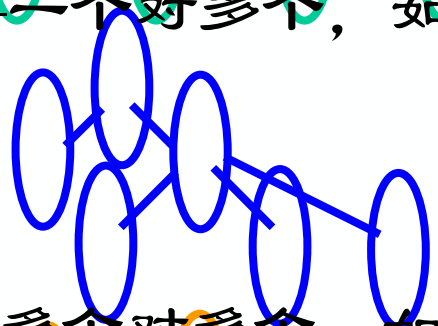
逻辑结构

集合——数据元素间除“同属于一个集合”外，无其它关系

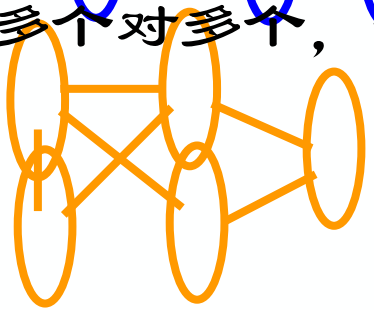


线性结构——一个对一个，如线性表、栈、队列

树形结构——一个对多个，如树



图形结构——多个对多个，如图



第5章　树和二叉树

5.1 树和二叉树的定义

5.2 案例引入

5.3 树和二叉树的抽象数据类型定义

5.4 二叉树的性质和存储结构

5.5 遍历二叉树和线索二叉树

5.6 树和森林

5.7 哈夫曼树及其应用

5.8 案例分析与实现

教学目标

1. 掌握二叉树的基本概念、性质和存储结构
2. 熟练掌握二叉树的前、中、后序遍历方法
3. 了解线索化二叉树的思想
4. 熟练掌握：哈夫曼树的实现方法、构造哈夫曼编码的方法
5. 了解：森林与二叉树的转换，树的遍历方法

5.1 树和二叉树的定义

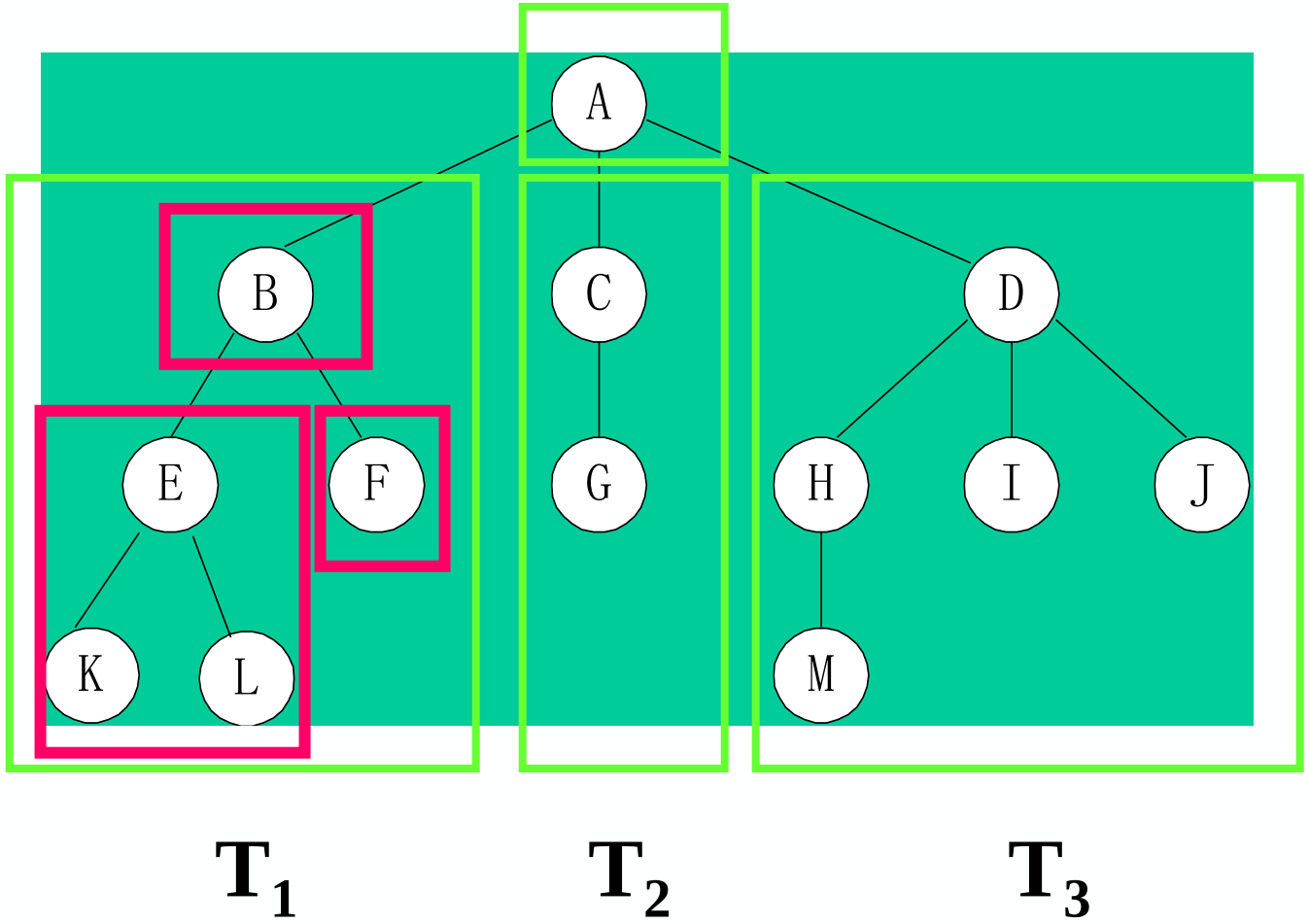
树的定义

树（Tree）是n（$n\geqslant 0$）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:

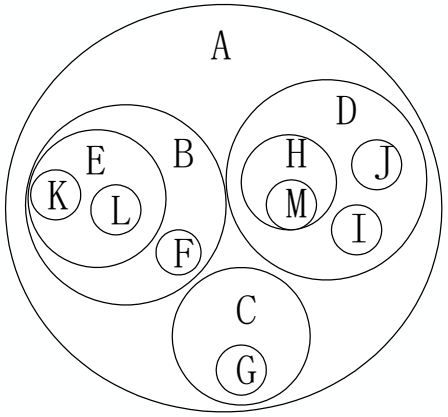
（1）有且仅有一个称之为根的结点；

（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$T_1, T_2, \cdots, T_m$, 其中每一个集合本身又是一棵树，并且称为根的子树（SubTree）。

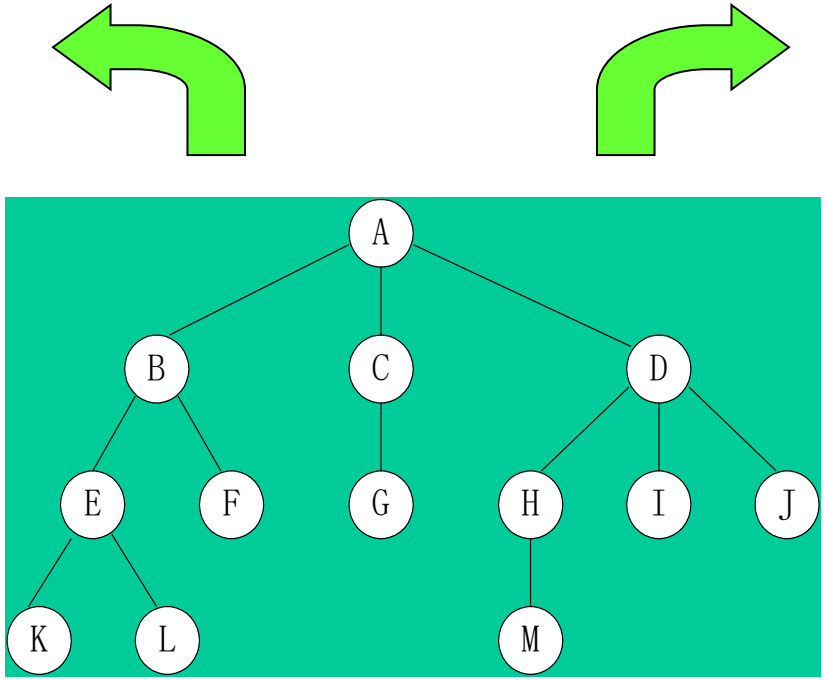
树是n个结点的有限集

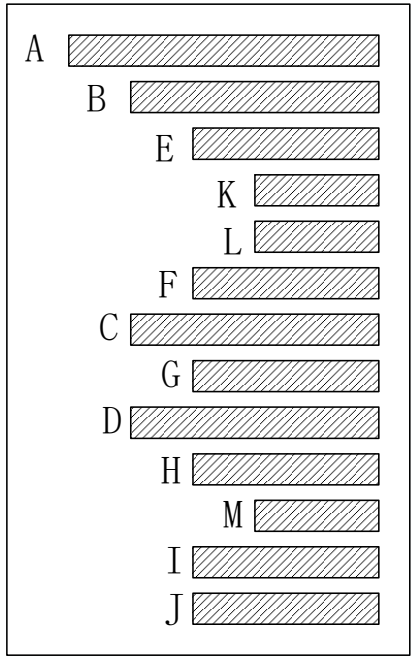


树的其它表示方式



嵌套集合





凹入表示

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

广义表

基本术语

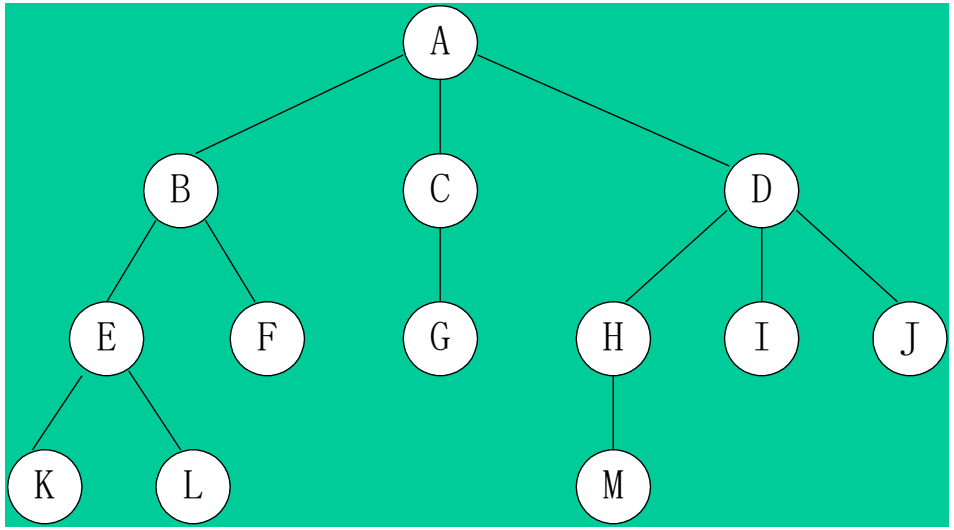
根 ——即根结点(没有前驱)

叶子 ——即终端结点(没有后继)

森林 ——指m棵不相交的树的集合(例如删除A后的子树个数)

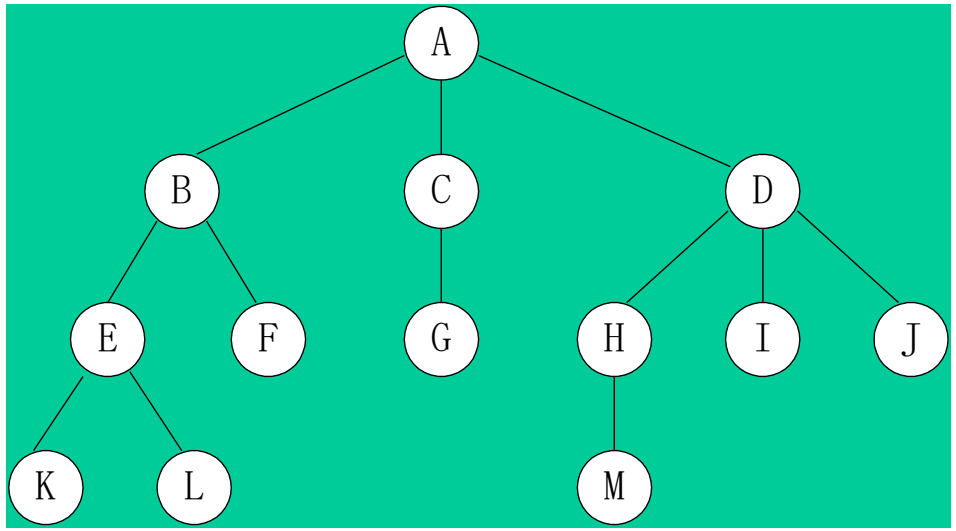
有序树 ——结点各子树从左至右有序，不能互换（左为第一）

无序树 ——结点各子树可互换位置。



基本术语

双亲 ——即上层的那个结点(直接前驱)
孩子 ——即下层结点的子树的根(直接后继)
兄弟 ——同一双亲下的同层结点（孩子之间互称兄弟）
堂兄弟 ——即双亲位于同一层的结点（但并非同一双亲）
祖先 ——即从根到该结点所经分支的所有结点
子孙 ——即该结点下层子树中的任一结点



基本术语

结点 ——即树的数据元素

结点的度——结点挂接的子树数

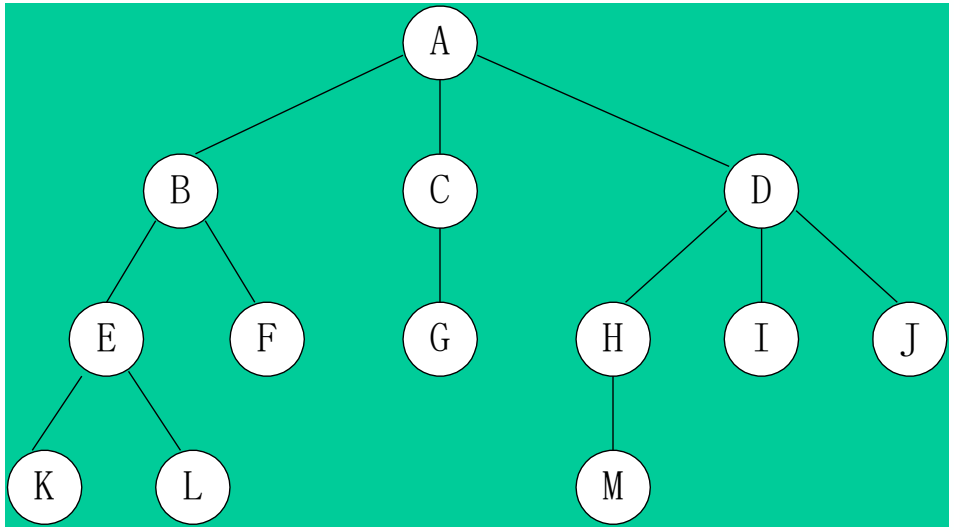
结点的层次——从根到该结点的层数（根结点算第一层）

终端结点 ——即度为0的结点，即叶子

分支结点 ——即度不为0的结点（也称为内部结点）

树的度 ——所有结点度中的最大值

树的深度(或高度)——指所有结点中最大的层数



层次

1

2

3

4

二叉树的定义

二叉树（Binary Tree）是n（$n \geqslant 0$）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。

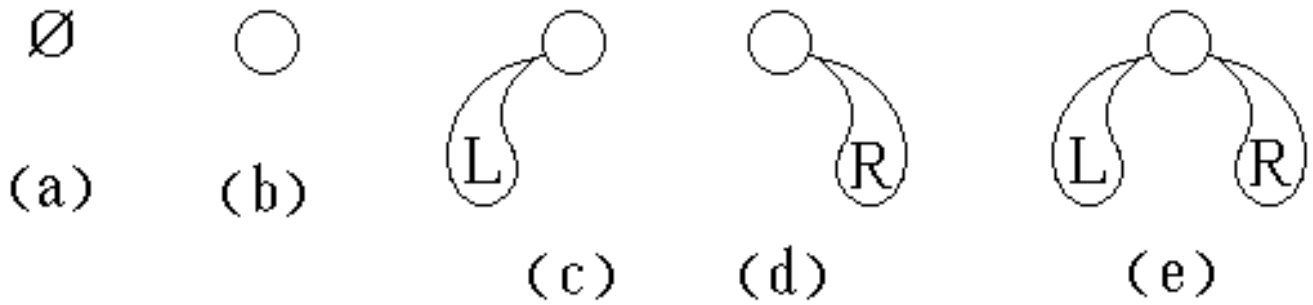
普通树（多叉树）若不转化为二叉树，则运算很难实现

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。

二叉树基本特点：

- 结点的度小于等于2
- 有序树（子树有序，不能颠倒）

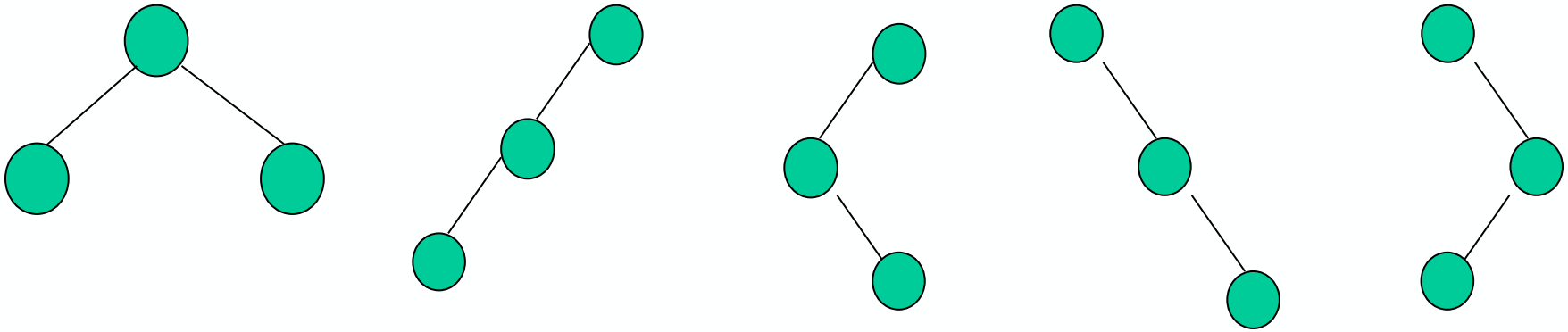


二叉树的五种不同形态

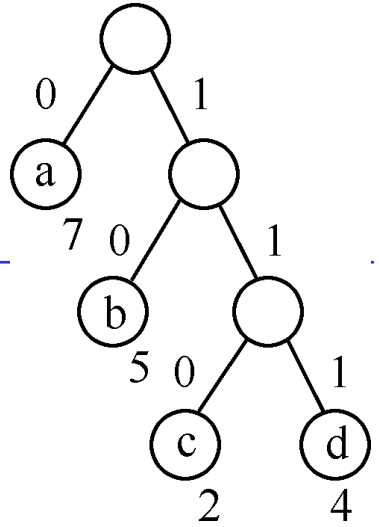
练习

具有3个结点的二叉树可能有几种不同形态？普通树呢？

5种/2种



5.2 案例引入



案例5.1： 数据压缩问题

将数据文件转换成由0、1组成的二进制串，称之为编码。

(a) 等长编码方案

字符	编码
a	00
b	01
c	10
d	11

(b) 不等长编码方案1

字符	编码
a	0
b	10
c	110
d	111

(c) 不等长编码方案2

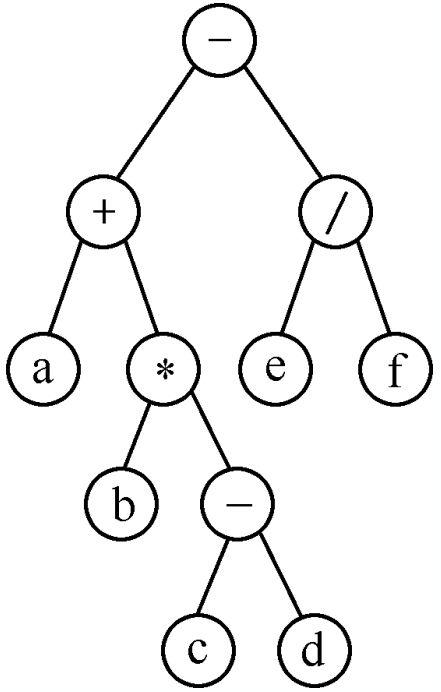
字符	编码
a	0
b	01
c	010
d	111

案例5.2：利用二叉树求解表达式的值

以二叉树表示表达式的递归定义如下：

（1）若表达式为数或简单变量，则相应二叉树中仅有一个根结点，其数据域存放该表达式信息；

（2）若表达式为“第一操作数 运算符 第二操作数”的形式，则相应的二叉树中以左子树表示第一操作数，右子树表示第二操作数，根结点的数据域存放运算符（若为一元运算符，则左子树为空），其中，操作数本身又为表达式。



(a + b *(c-d)-e/f)的二叉树

5.3 树和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 BinaryTree{

数据对象D： D是具有相同特性的数据元素的集合。

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

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。

5.4 二叉树的性质和存储结构

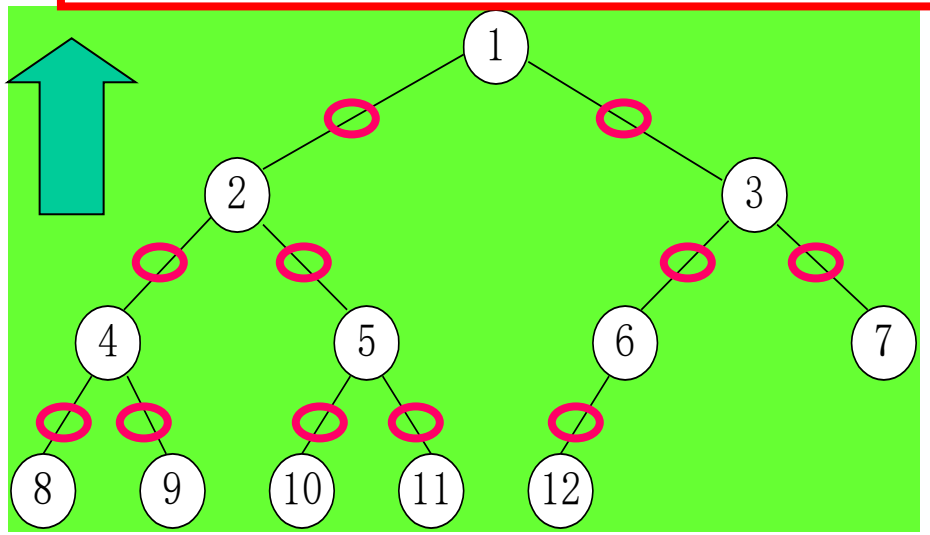
性质1: 在二叉树的第i层上至多有2^{i-1}个结点

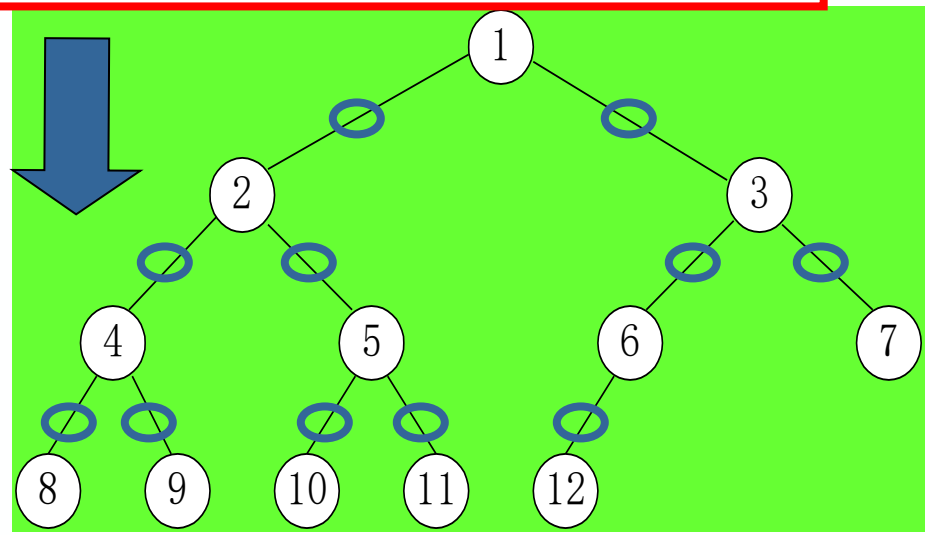
提问：第i层上至少有__1__结点?

性质2: 深度为k的二叉树至多有2^k-1个结点

提问：深度为k时至少有__k__个结点?

性质3: 对于任何一棵二叉树，若2度的结点数有n_2个，则叶子数n_0必定为n_2+1 （即$n_0=n_2+1$）



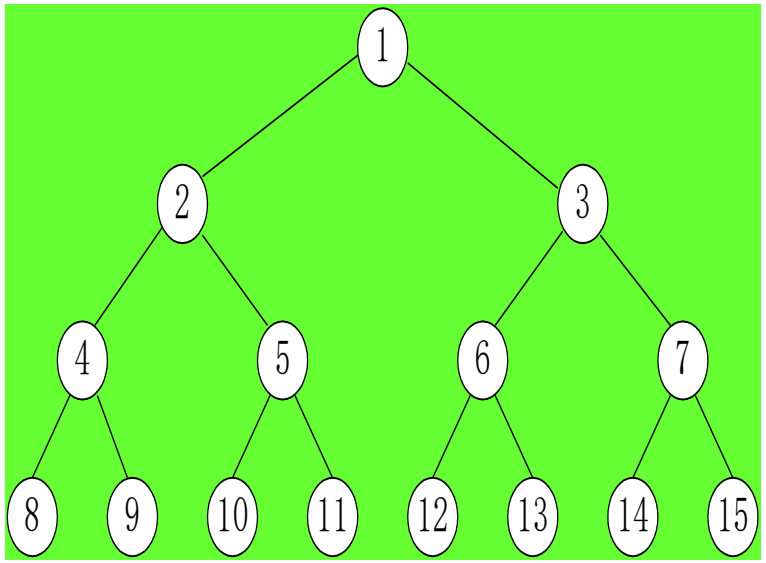


$$B = n - 1$$

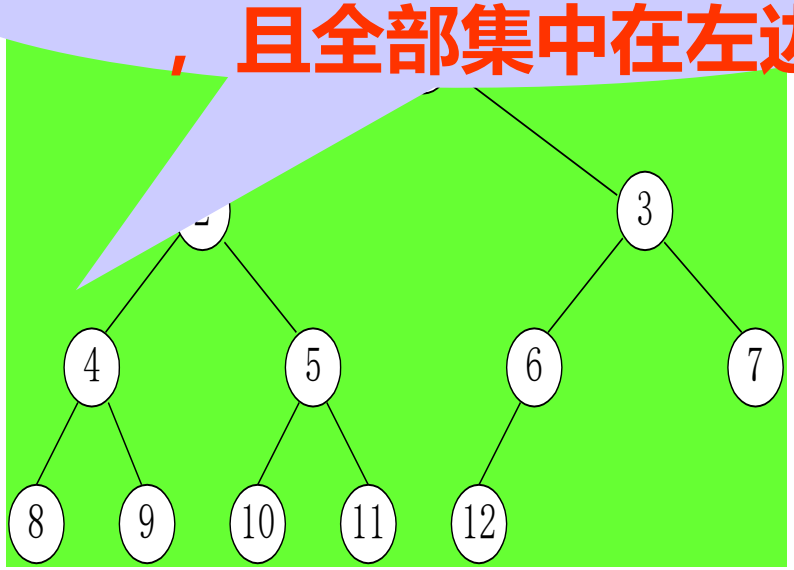
$$B = n_2 \times 2 + n_1 \times 1$$

$$n = n_2 \times 2 + n_1 \times 1 + 1 = n_2 + n_1 + n_0$$

特殊形态的二叉树



只有最后一层叶子不满，且全部集中在左边



满二叉树： 一棵深度为k且有2^k-1个结点的二叉树。（特点：每层都“充满”了结点）

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的结点一一对应