数据结构（C语言版）（第2版）

树和二叉树

树和森林

主讲教师：汪红松

教学内容 Contents

一、树的存储结构

1.双亲表示法

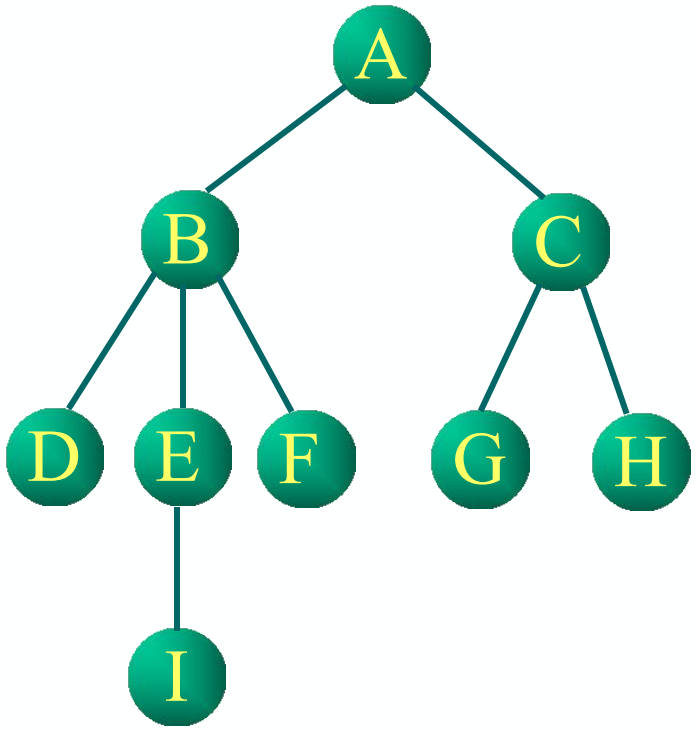
基本思想：用一维数组来存储树的各个结点（一般按层序存储），数组中的一个元素对应树中的一个结点，包括结点的数据信息以及该结点的双亲在数组中的下标。

data	parent

data：存储树中结点的数据信息。
parent：存储该结点的双亲在数组中的下标。

一、树的存储结构

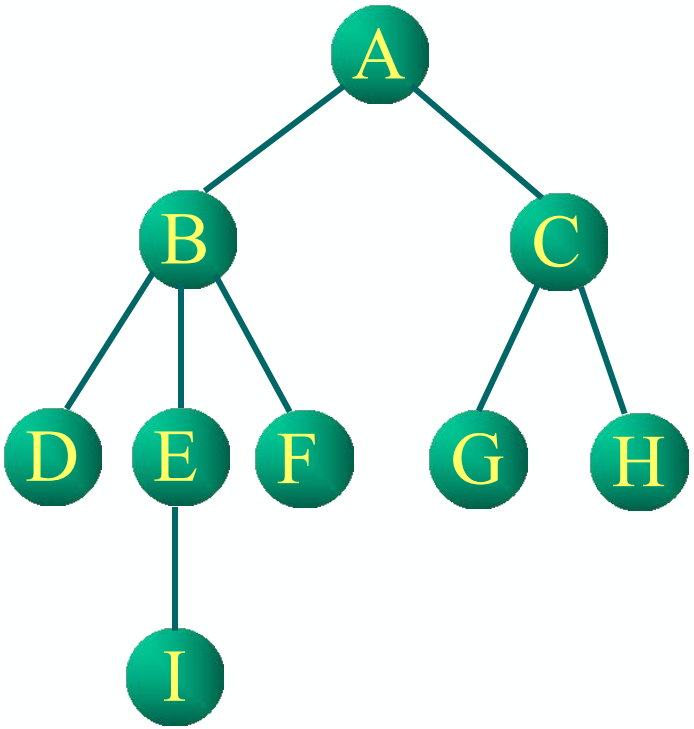
1.双亲表示法



下标	data	parent
0	A	−1
1	B	0
2	C	0
3	D	1
4	E	1
5	F	1
6	G	2
7	H	2
8	I	4

如何查找双亲结点？时间性能？

一、树的存储结构



下标	data	parent	firstchild
0	A	−1	1
1	B	0	3
2	C	0	6
3	D	1	-1
4	E	1	8
5	F	1	-1
6	G	2	-1
7	H	2	-1
8	I	4	-1

如何查找孩子结点？时间性能？

2.孩子链表表示法

如何表示孩子？

链表中的每个结点包括一个数据域和多个指针域，每个指针域指向该结点的一个孩子结点。

方案一：指针域的个数等于树的度。

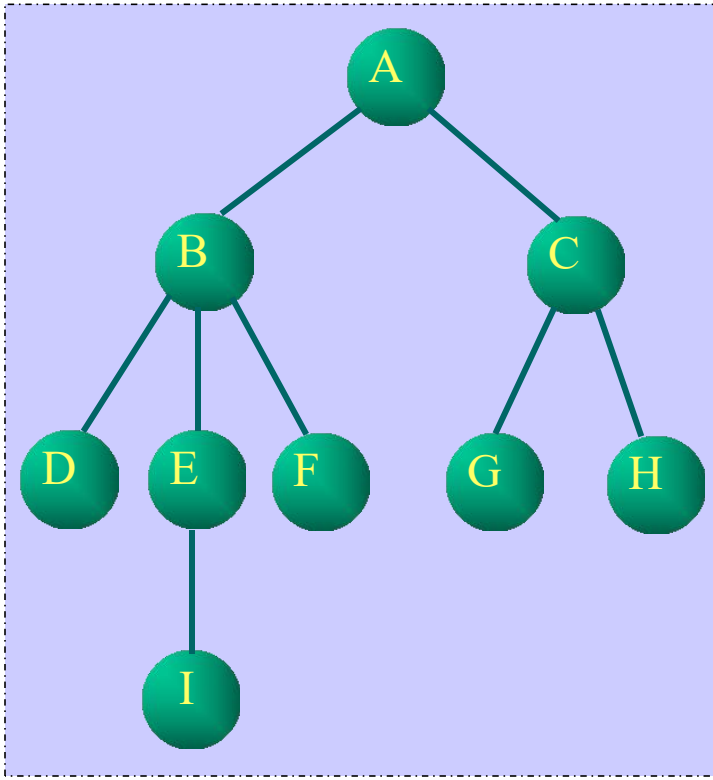
data	child1	child2	……	childd

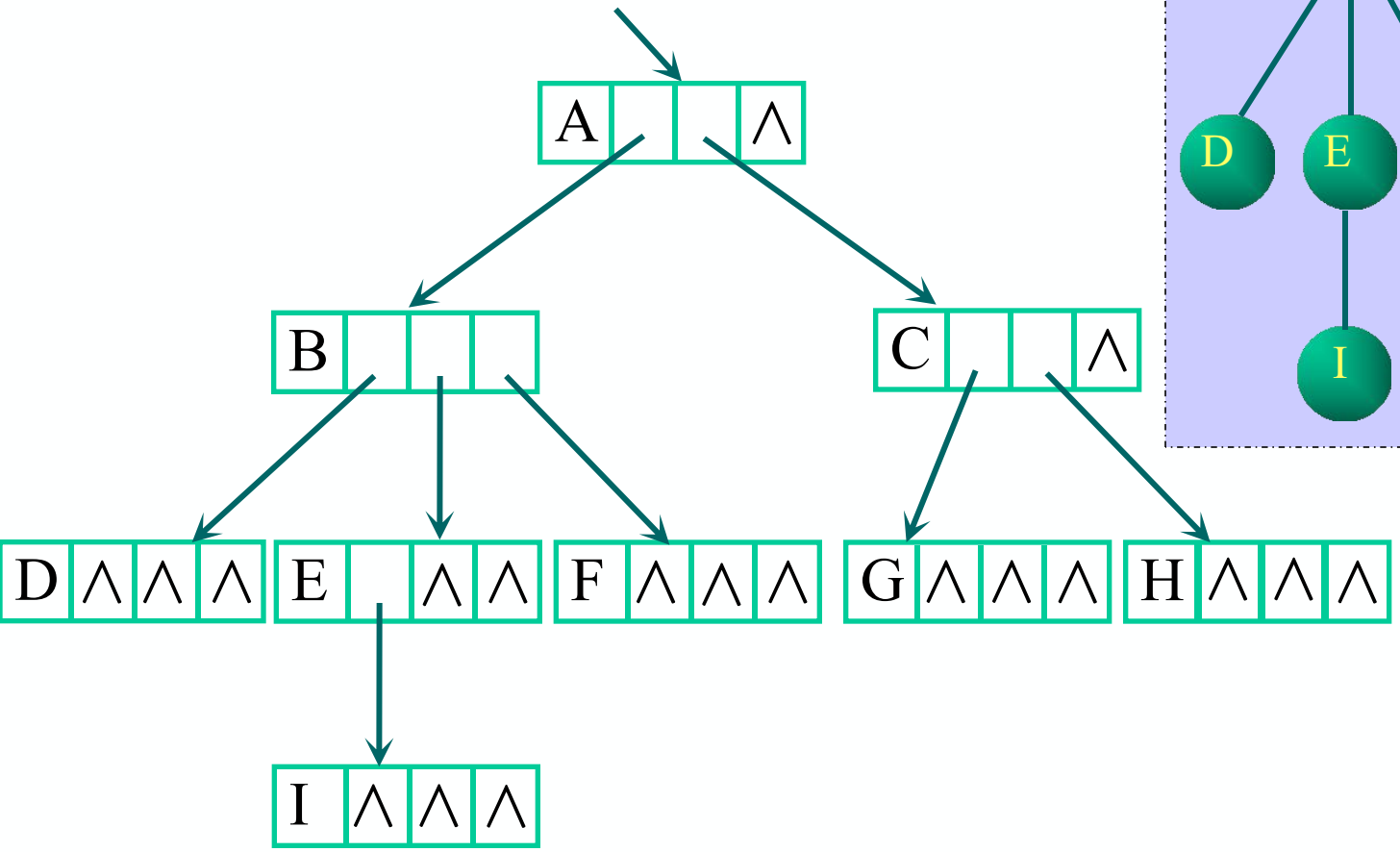
其中：data：数据域，存放该结点的数据信息；

child1~childd：指针域，指向该结点的孩子。

一、树的存储结构

缺点：浪费空间





2.孩子链表表示法

如何表示孩子？

链表中的每个结点包括一个数据域和多个指针域，每个指针域指向该结点的一个孩子结点。

方案二： 指针域的个数等于该结点的度。

data	degree	child1	child2	……	childd

其中：data：数据域，存放该结点的数据信息；
degree：度域，存放该结点的度；
child1~childd：指针域，指向该结点的孩子。

2.孩子链表表示法

如何表示孩子？

将结点的所有孩子放在一起，构成线性表。

孩子链表的基本思想：

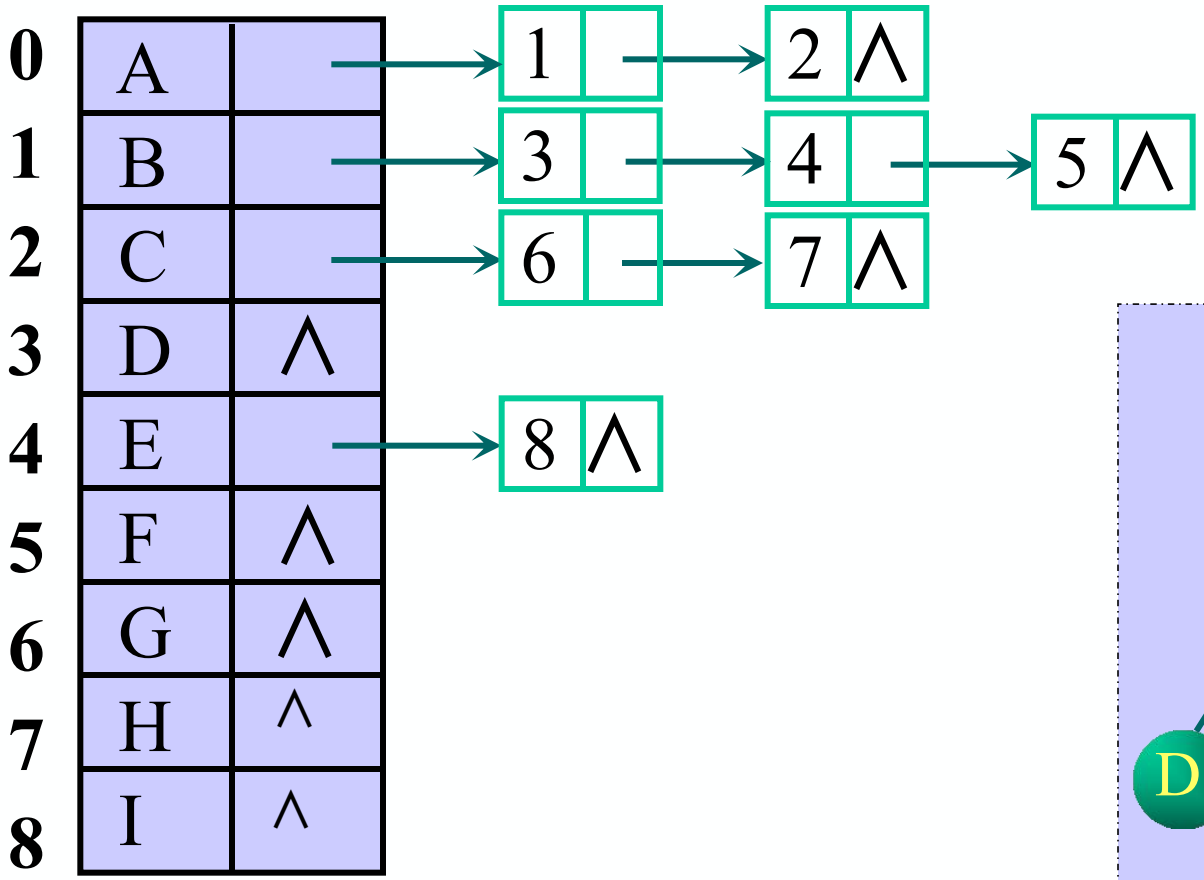
（1）把每个结点的孩子排列起来，看成是一个线性表，且以单链表存储，则*n*个结点共有 *n* 个孩子链表。

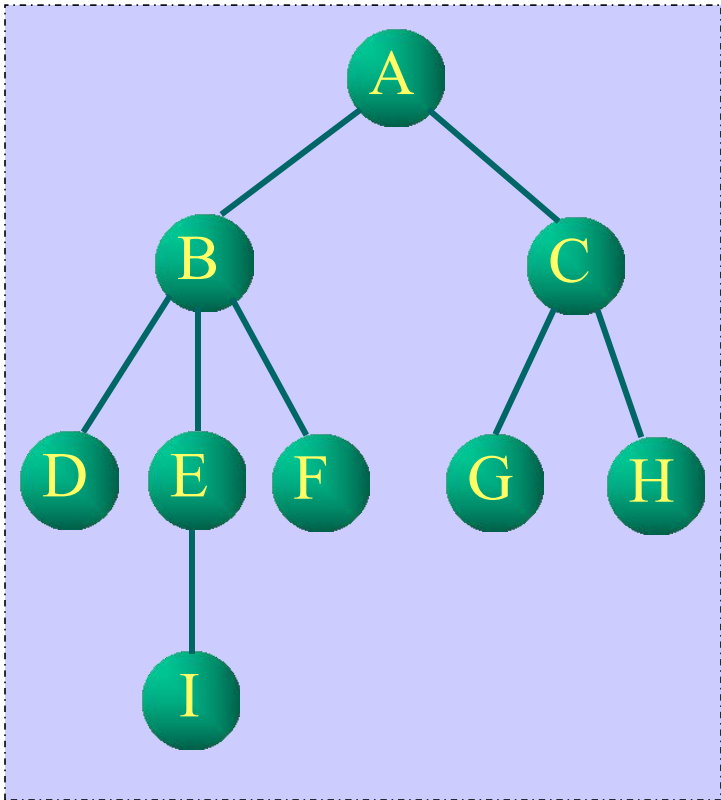
（2） *n* 个单链表共有 *n* 个头指针，这 *n* 个头指针又组成了一个线性表，为了便于进行查找采用顺序存储。

（3）最后，将存放 *n* 个头指针的数组和存放*n*个结点的数组结合起来，构成孩子链表的表头数组。

一、树的存储结构

下标 **data firstchild**

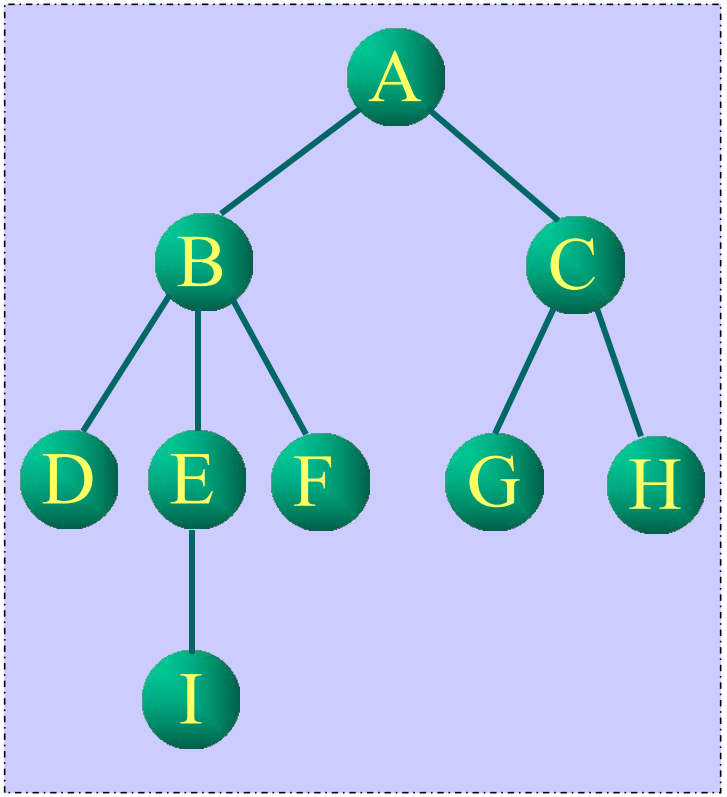




3.孩子兄弟表示法

某结点的第一个孩子是惟一的，
某结点的右兄弟是惟一的。

设置两个分别指向该结点的第一个孩子和右兄弟的指针 。



3.孩子兄弟表示法

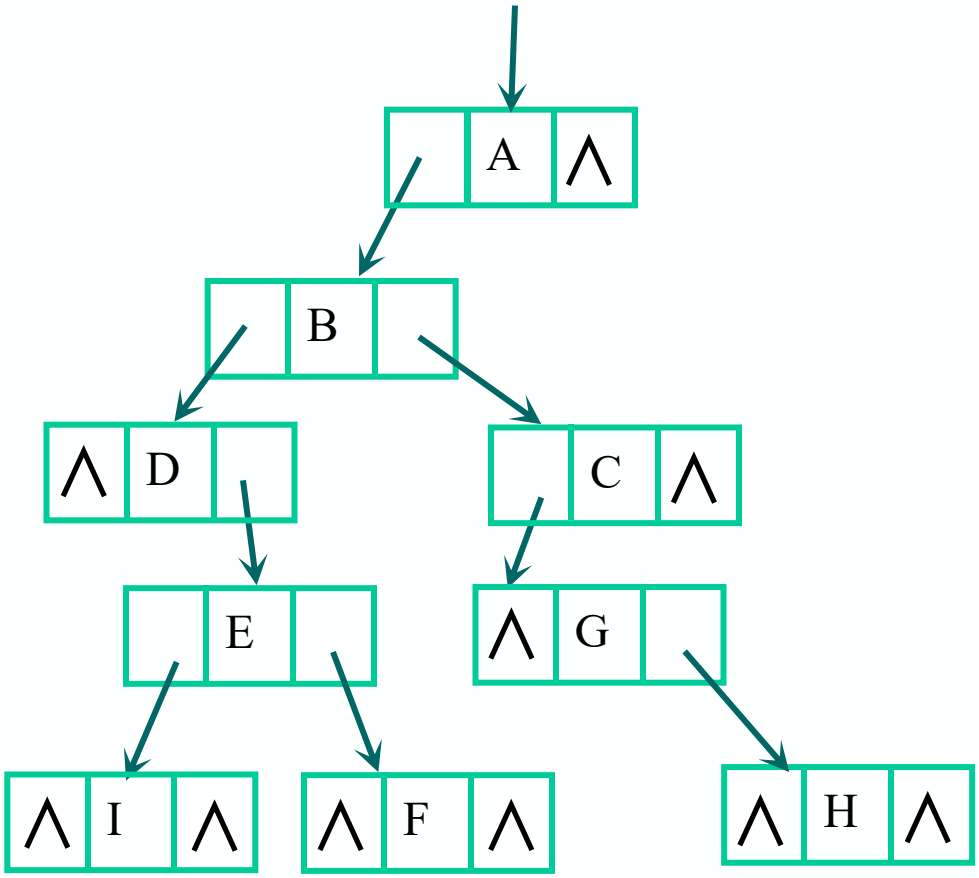
结点结构

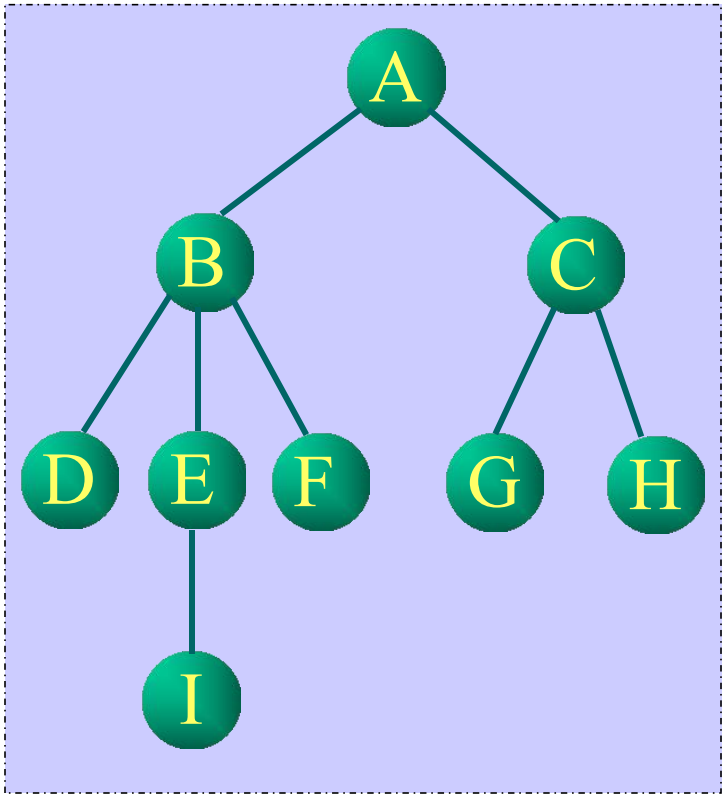
firstchild	data	rightsib

data：数据域，存储该结点的数据信息；
firstchild：指针域，指向该结点第一个孩子；
rightsib：指针域，指向该结点的右兄弟结点。

一、树的存储结构

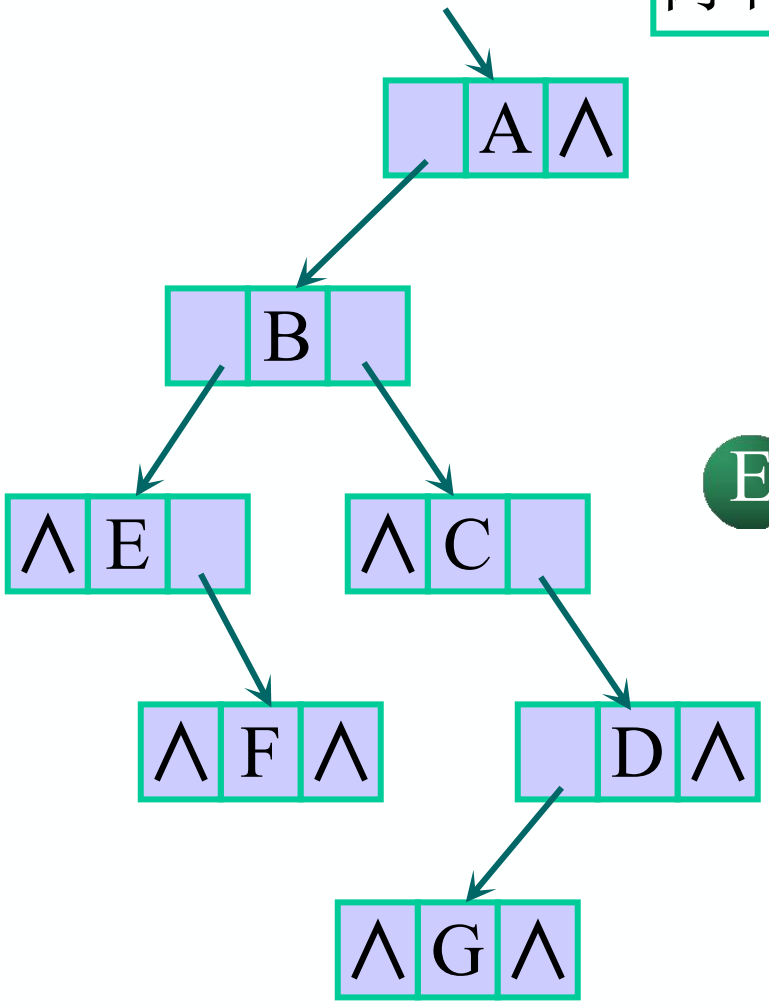
3.孩子兄弟表示法



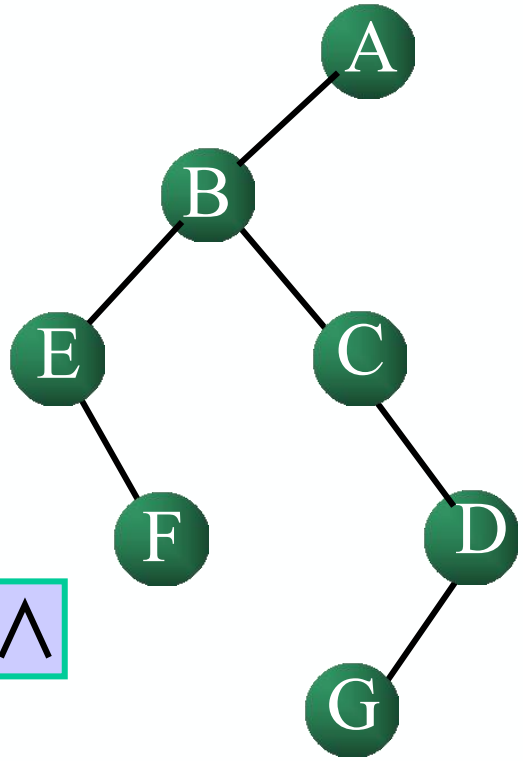


二、树、森林与二叉树的转换

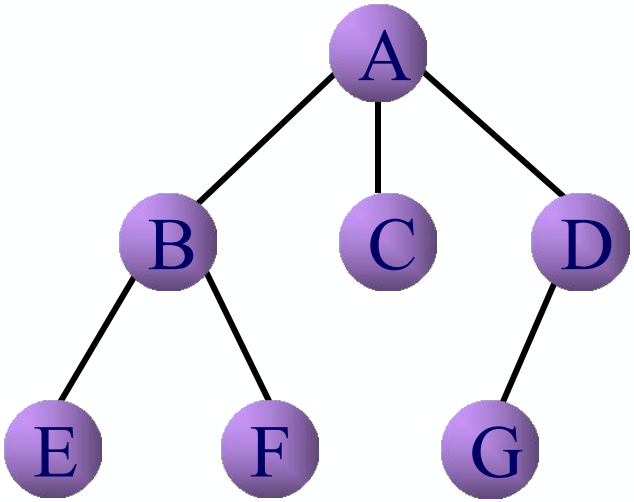
树和二叉树之间具有对应关系



(a)



(b)



(c)

二、树、森林与二叉树的转换

1.树和二叉树之间的对应关系

树：兄弟关系

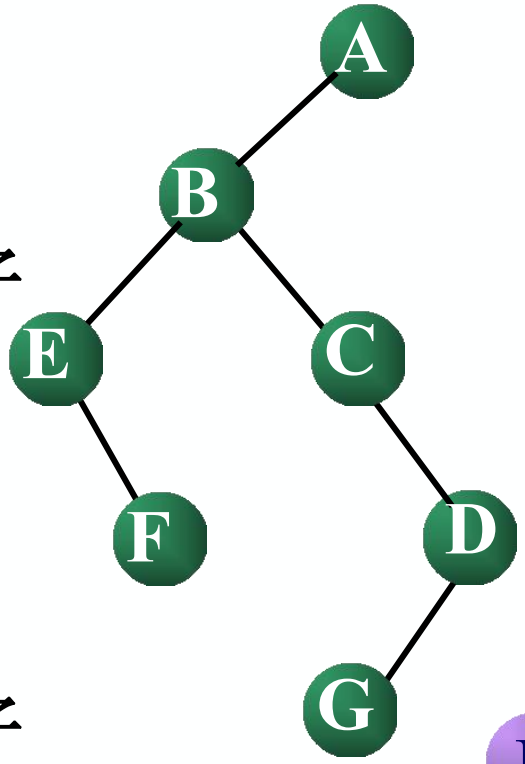
⇕

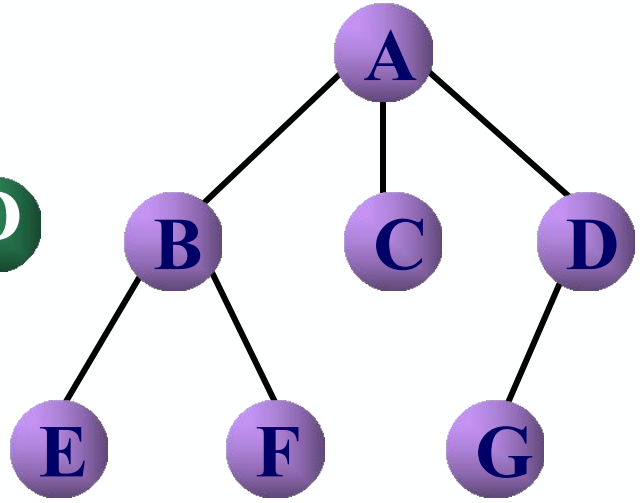
二叉树：双亲和右孩子

树：双亲和长子

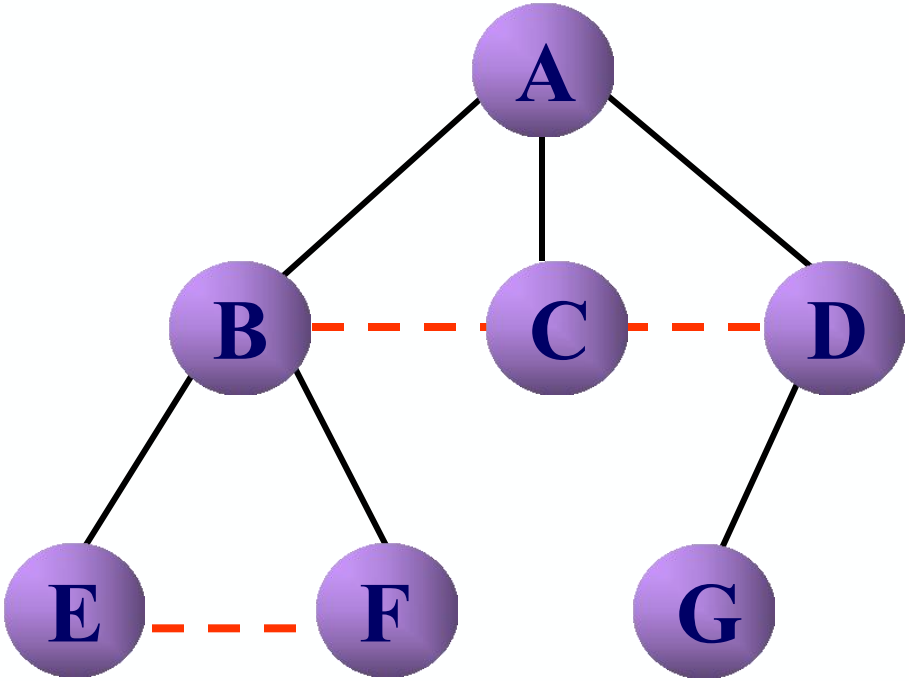
⇕

二叉树：双亲和左孩子



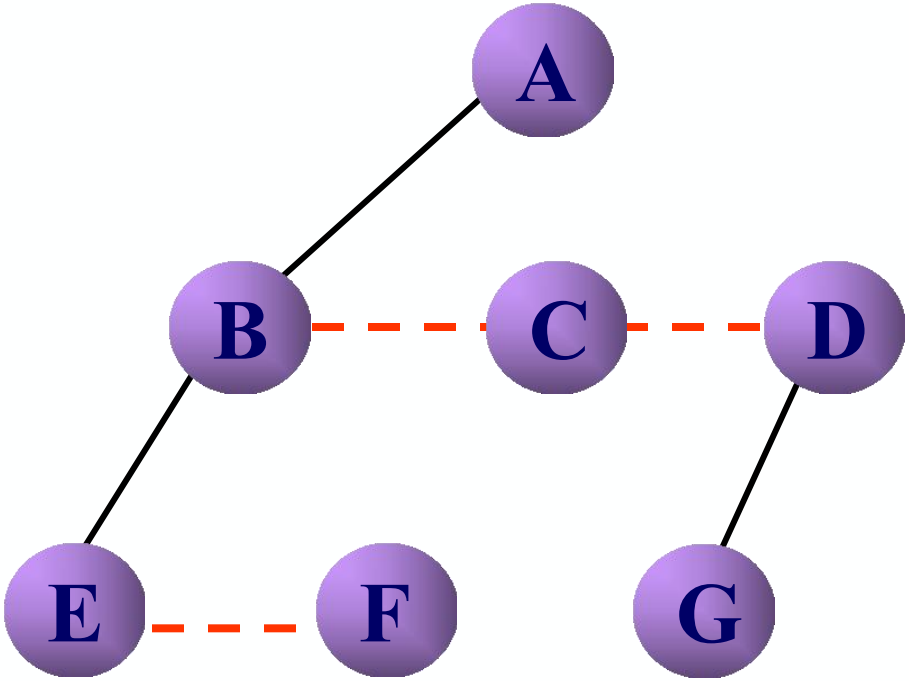


二、树、森林与二叉树的转换



（1）兄弟加线；

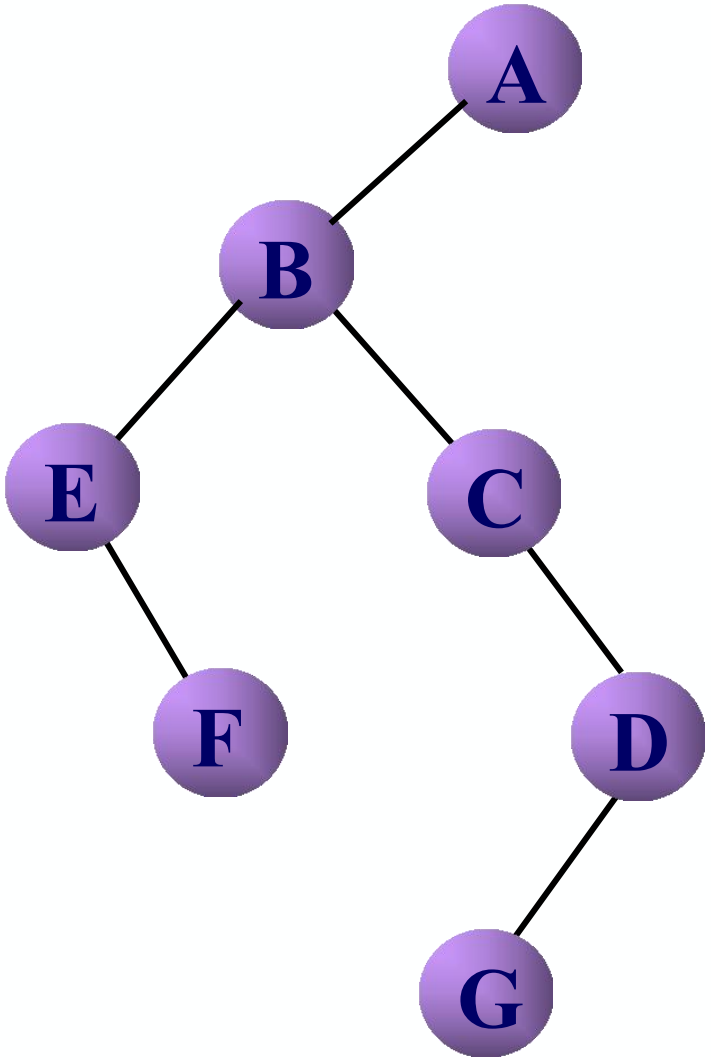
二、树、森林与二叉树的转换



（1）兄弟加线；

（2）保留双亲与第一孩子连线,删去与其他孩子的连线。

二、树、森林与二叉树的转换



（1）兄弟加线；

（2）保留双亲与第一孩子连线，删去与其他孩子的连线；

（3）顺时针转动，使之层次分明。

2. 树转换为二叉树

（1）加线——树中所有相邻兄弟之间加一条连线。

（2）去线——对树中的每个结点，只保留它与第一个孩子结点之间的连线，删去它与其它孩子结点之间的连线。

（3）层次调整——以根结点为轴心，将树顺时针转动一定的角度，使之层次分明。

3. 森林转换成二叉树

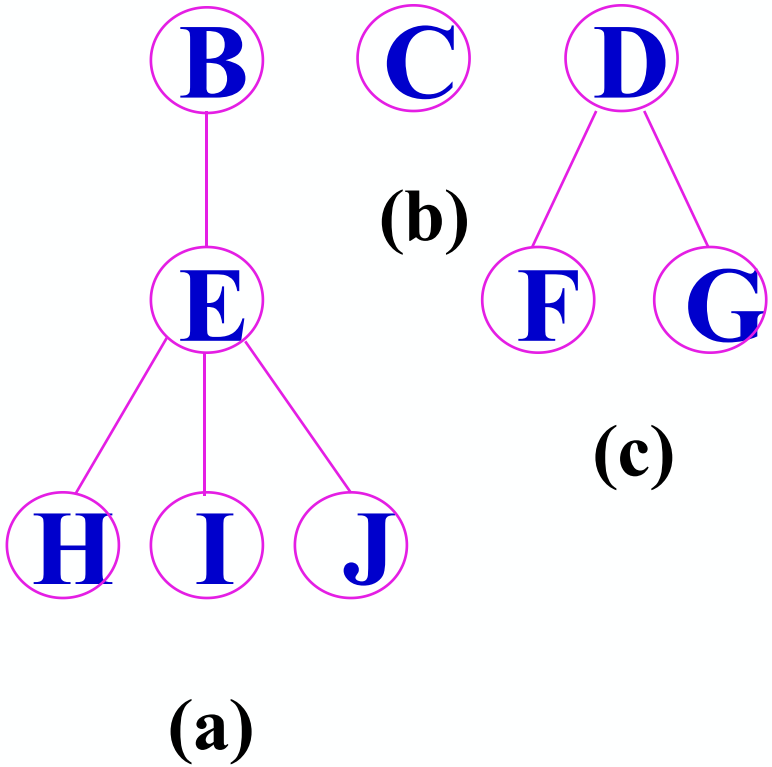
森林转换为二叉树的方法为：

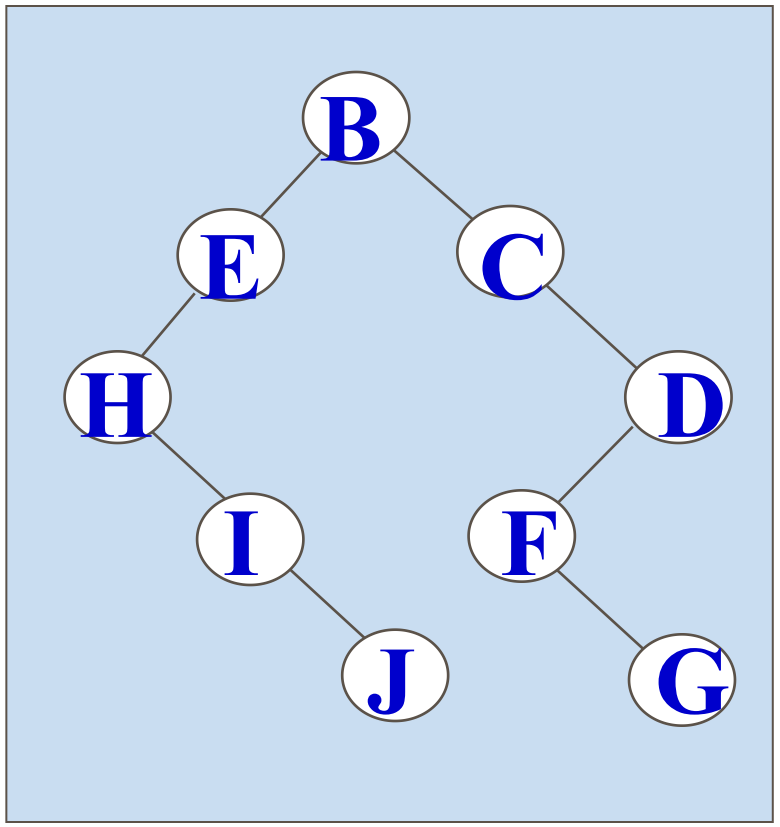
（1）将森林中的每棵树转换成相应的二叉树。

（2）第一棵二叉树不动，从第二棵二叉树开始，依次把后一棵二叉树的根结点作为前一棵二叉树根结点的右孩子，当所有二叉树连在一起后，所得到的二叉树就是由森林转换得到的二叉树。

二、树、森林与二叉树的转换

3. 森林转换成二叉树





4.二叉树转换为树或森林的方法为：

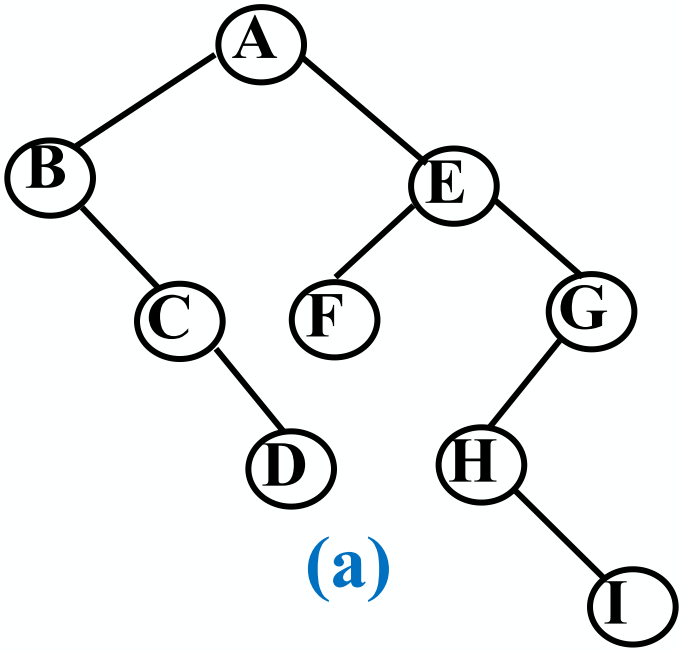
（1）若某结点是其双亲的左孩子，则把该结点的右孩子、右孩子的右孩子、……都与该结点的双亲结点用线连起来；

（2）删掉原二叉树中所有双亲结点与右孩子结点的连线；

（3）整理由（1）、（2）两步所得到的树或森林，使之结构层次分明。

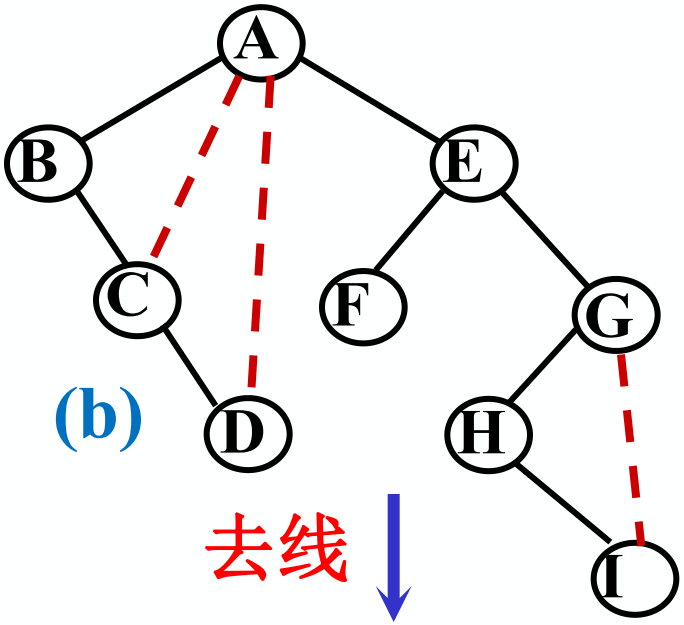
二、树、森林与二叉树的转换

二叉树转换为树或森林



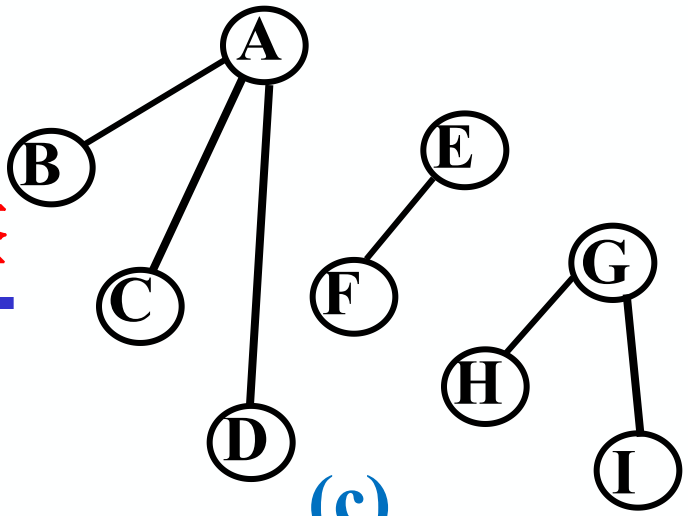
(a)

加线



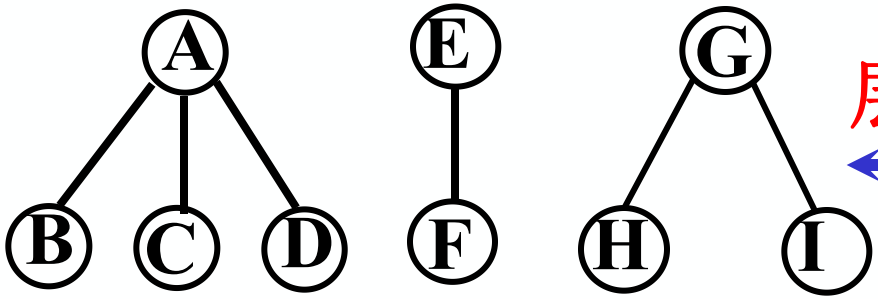
(b)

去线



(c)

层次调整

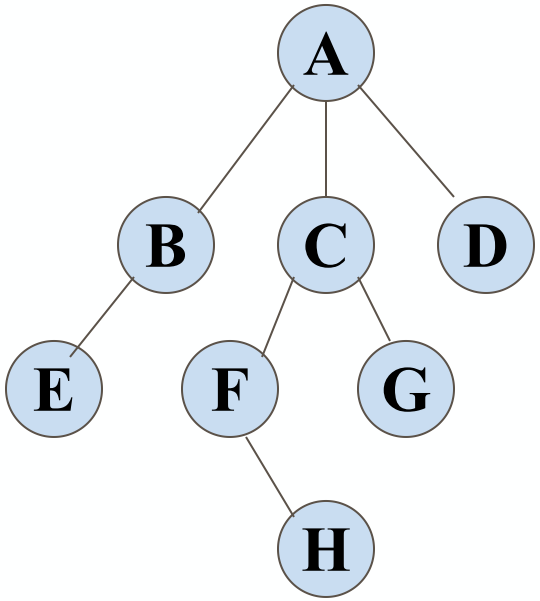


(d)

三、树、森林的遍历

1. 树的遍历主要有以下两种：

先根遍历　　后根遍历



（1）先根遍历

若树非空，则遍历过程为：

① 访问根结点。

② 从左到右，依次先根遍历根结点的每一棵子树。

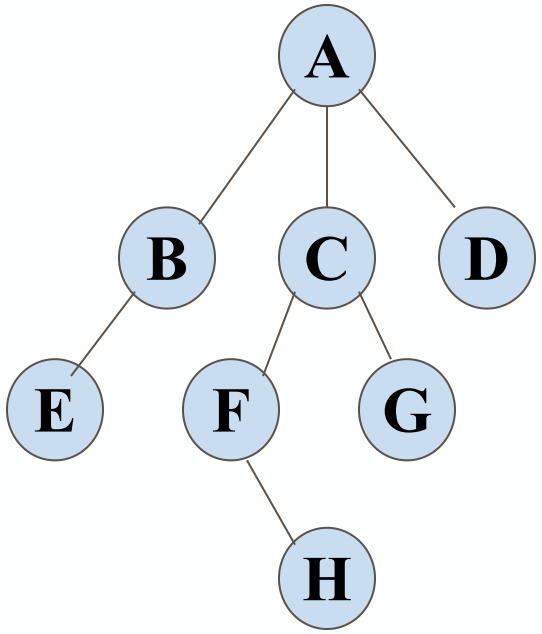
如图中树的先根遍历序列为：ABECFHGD。

（2）后根遍历

若树非空，则遍历过程为：

① 从左到右，依次后根遍历根结点的每一棵子树。

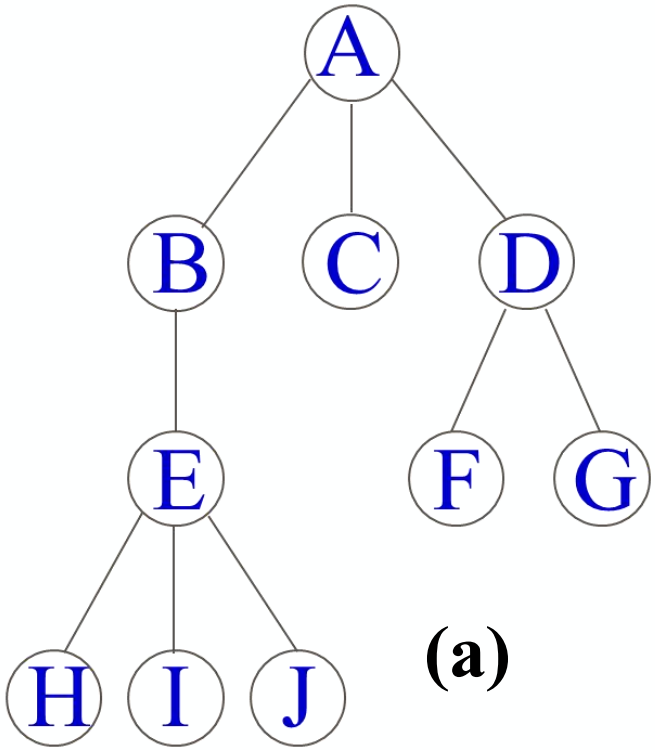
② 访问根结点。



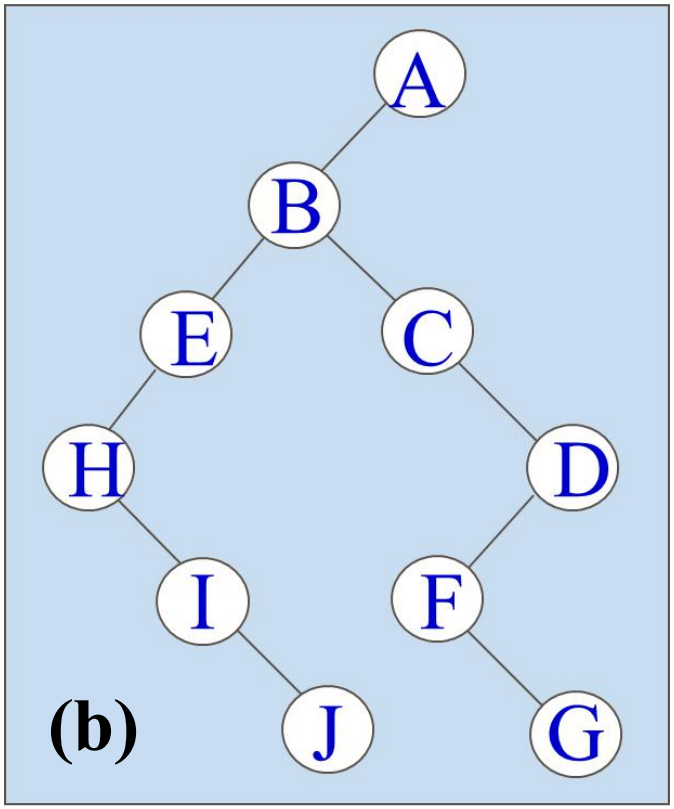
如图树的后根遍历序列为：

EBHFGCDA。

三、树、森林的遍历



(a)



(b)

树的先根遍历：A B E H I J C D F G

对应二叉树的先序遍历

树的后根遍历：H I J E B C F G D A

对应二叉树的中序遍历

三、树、森林的遍历

2. 森林的遍历方法主要有以下两种：
先序遍历　中序遍历

（1）先序遍历

若森林非空，则遍历方法为：

① 访问森林中第一棵树的根结点。

② 先序遍历第一棵树的根结点的子树森林。

③ 先序遍历除去第一棵树之后剩余的树构成的森林。

即：依次从左至右对森林中的每一棵树进行先根遍历。

（2）中序遍历

若森林非空，则遍历方法为：

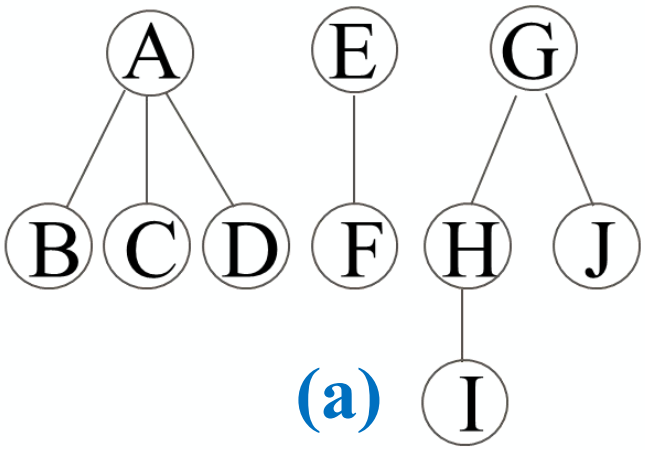
①中序遍历森林中第一棵树的根结点的子树森林。

②访问第一棵树的根结点。

③中序遍历除去第一棵树之后剩余的树构成的森林。

即：依次从左至右对森林中的每一棵树进行后根遍历。

三、树、森林的遍历

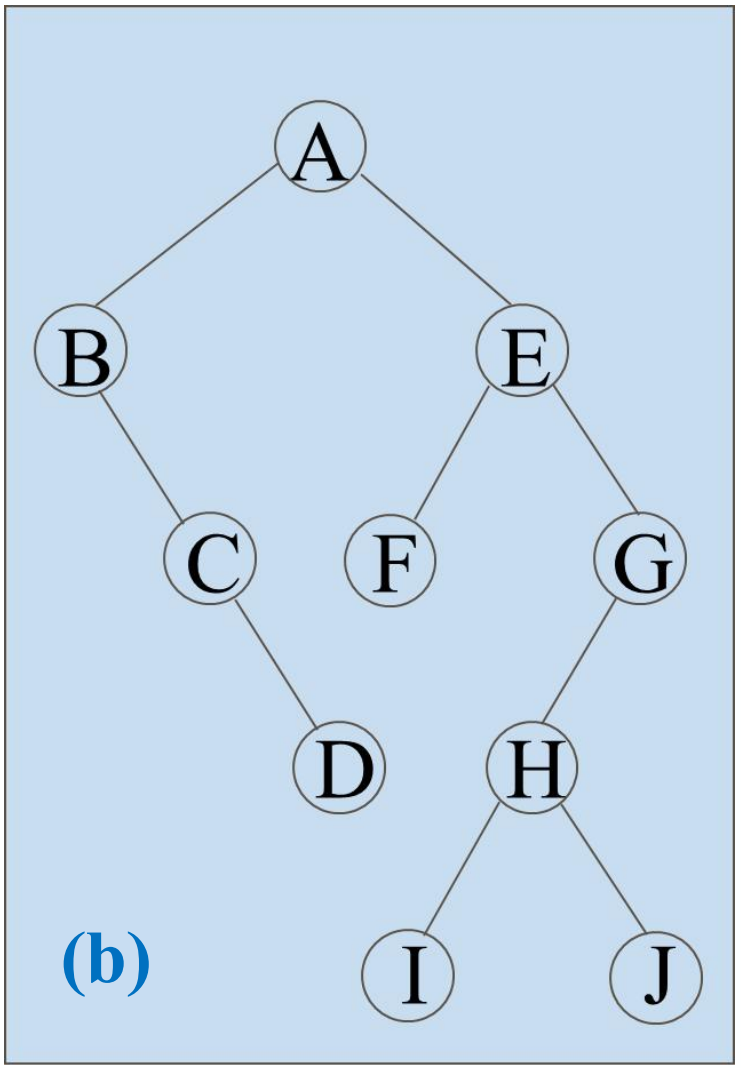


先序遍历森林时顶点的访问次序：

A B C D E F G H I J

中序遍历森林时顶点的访问次序：

B C D A F E I H J G



小结

1. 树的三种存储结构
2. 树与二叉树、森林与二叉树的转换方法
3. 树和森林的遍历方法