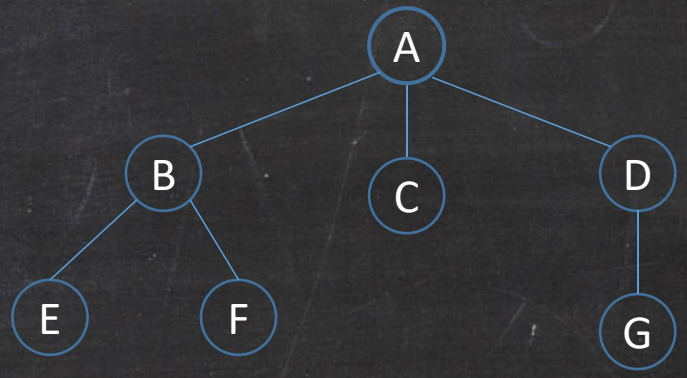
3.3 树和二叉树

主要知识点

- 树及二叉树的定义
- 树的相关概念
- 二叉树的性质
- 满二叉树和完全二叉树
- 二叉树的存储结构
- 最优二叉树
- 二叉树的遍历
- 二叉查找树

树

- 定义：树是n(n≥0)个结点的有限集合。当n=0时称为空树。在任一非空树中，有且仅有一个称为根的结点：其余结点可分为m(m≥0)个互不相交的有限集$T_1, T_2..., T_m$，其中每个集合又都是一棵树，并且称为根结点的子树。

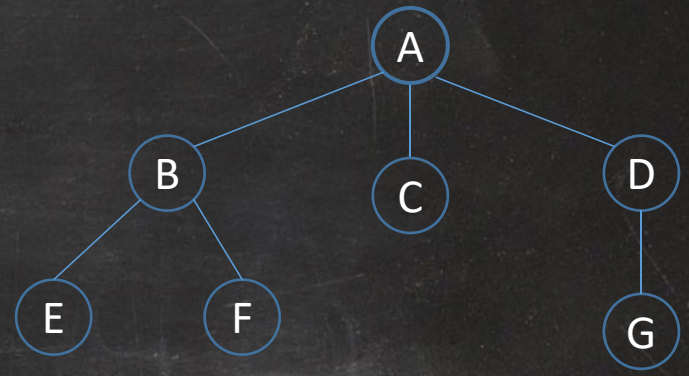


树的相关概念

1、双亲、孩子和兄弟：

2、结点的度：一个结点的子树的个数记为该结点的度。

3、叶子结点：也称为终端结点，指度为零的结点。



4、内部结点：度不为零的结点称为分支结点或非终端结点。除根结点之外，分支结点也称为内部结点。

5、结点的层次：根为第一层，根的孩子在第二层，依此类推。

6、树的高度：一棵树的最大层次数记为树的高度（或深度）。

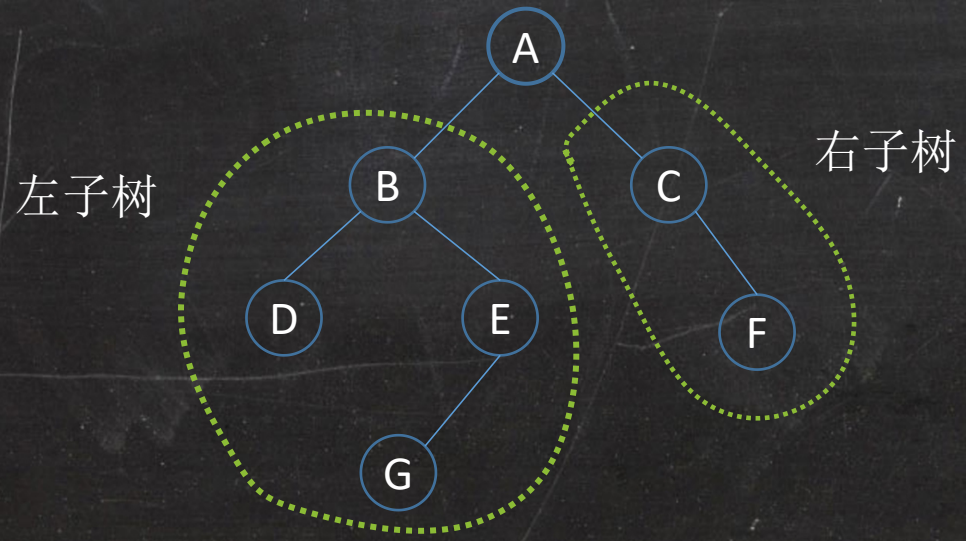
7、有序（无序）树：若将树中结点的各子树看成是从左到右具有次序的，即不能交换，则称该树为有序树，否则称为无序树。

8、森林：m(m≥0)棵互不相交的树的集合。

紫依

二叉树

- 定义：二叉树是n(n≥0)个结点的有限集合，它或者是空树(n=0)，或者是由一个根结点及两棵不相交的、分别称为左子树和右子树的二叉树所组成。
- 树和二叉树的区别：（1）二叉树中结点的子树要区分左子树和右子树，即使只有一棵子树，而树中不用区分。（2）二叉树中结点的最大度为2，而树中无限制。



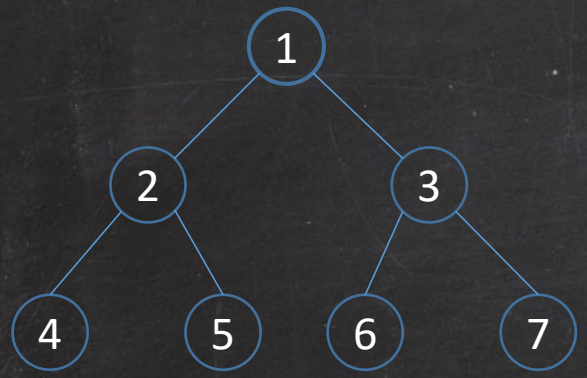
二叉树的性质

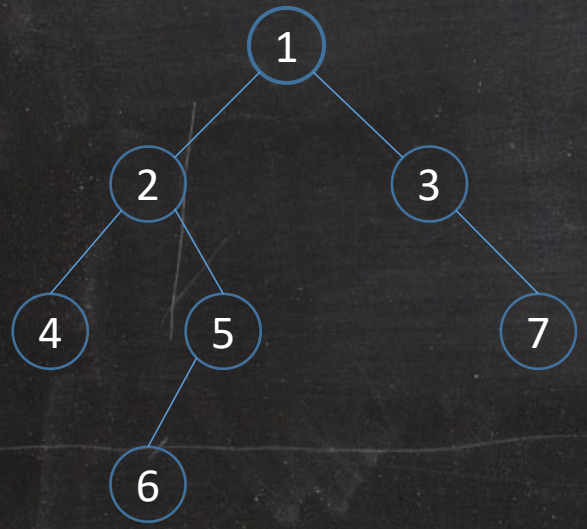
1、二叉树第i（i≥1）层上至多有2^{i-1}个节点。

2、深度为k的二叉树至多有2^k-1个结点（k≥1）。

3、对任何一棵二叉树，若其终端结点数为n_0，度为2的结点数为n_2，则$n_0=n_2+1$。

紫依

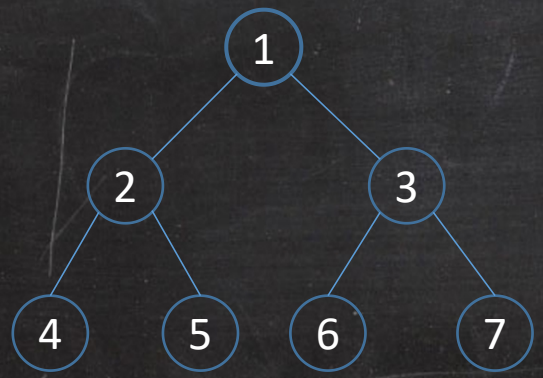




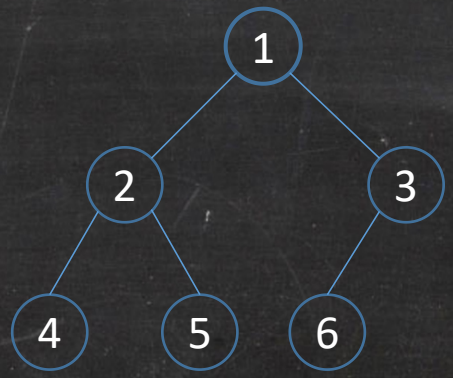
紫依

满二叉树和完全二叉树

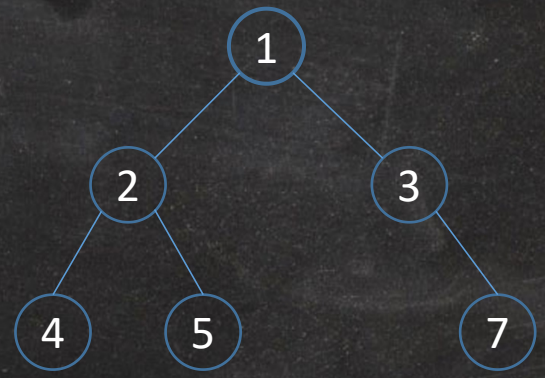
- 满二叉树的定义：若深度为k的二叉树有2^k-1个结点，则称其为满二叉树。
- 完全二叉树：当深度为k、有n个结点的二叉树，当且仅当其每一个结点都与深度为k的满二叉树中编号为1～n的结点一一对应时，称之为完全二叉树。



（a）满二叉树



（b）完全二叉树

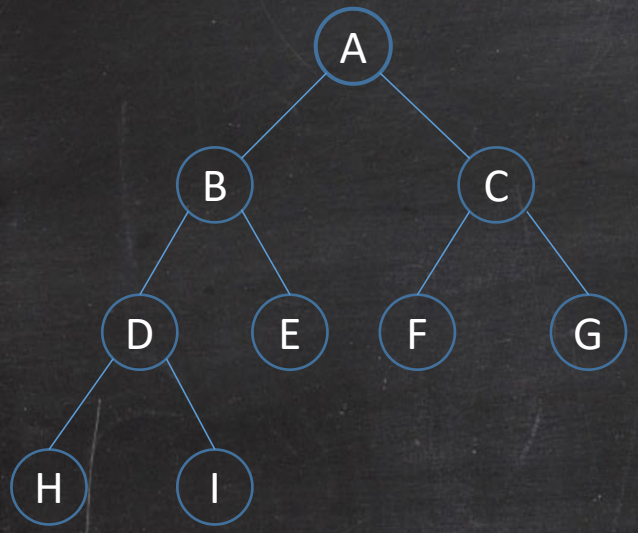


（c）非完全二叉树

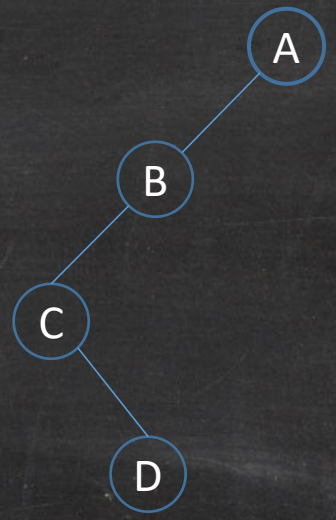
紫依

二叉树的存储结构

（1）顺序存储：



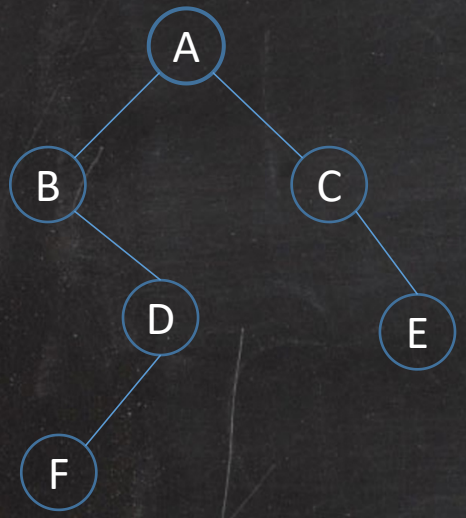
A	B	C	D	E	F	G	H	I	



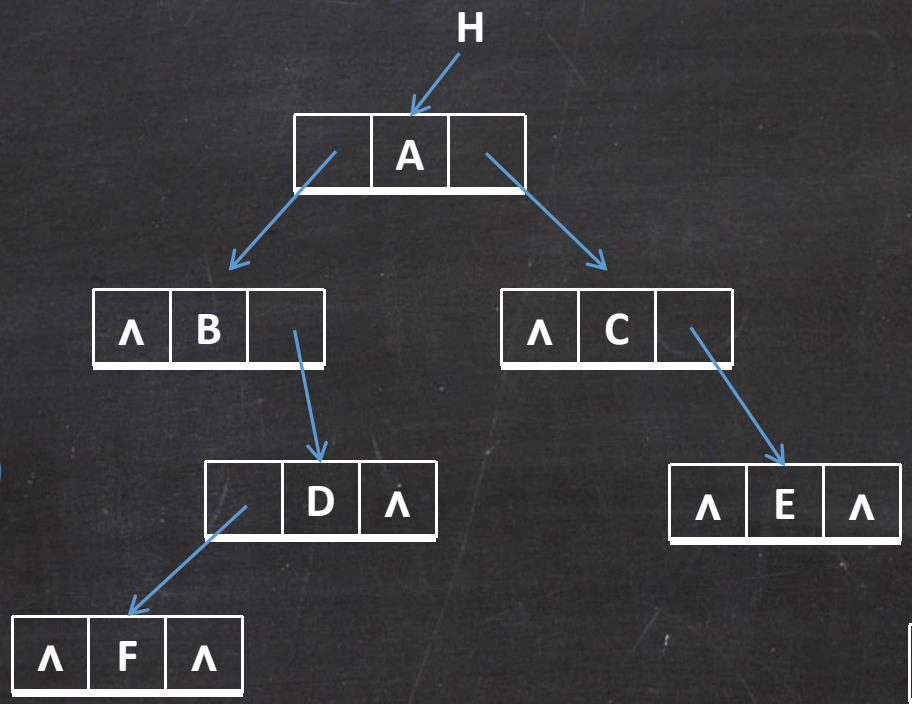
A	B		C					D	

二叉树的存储结构

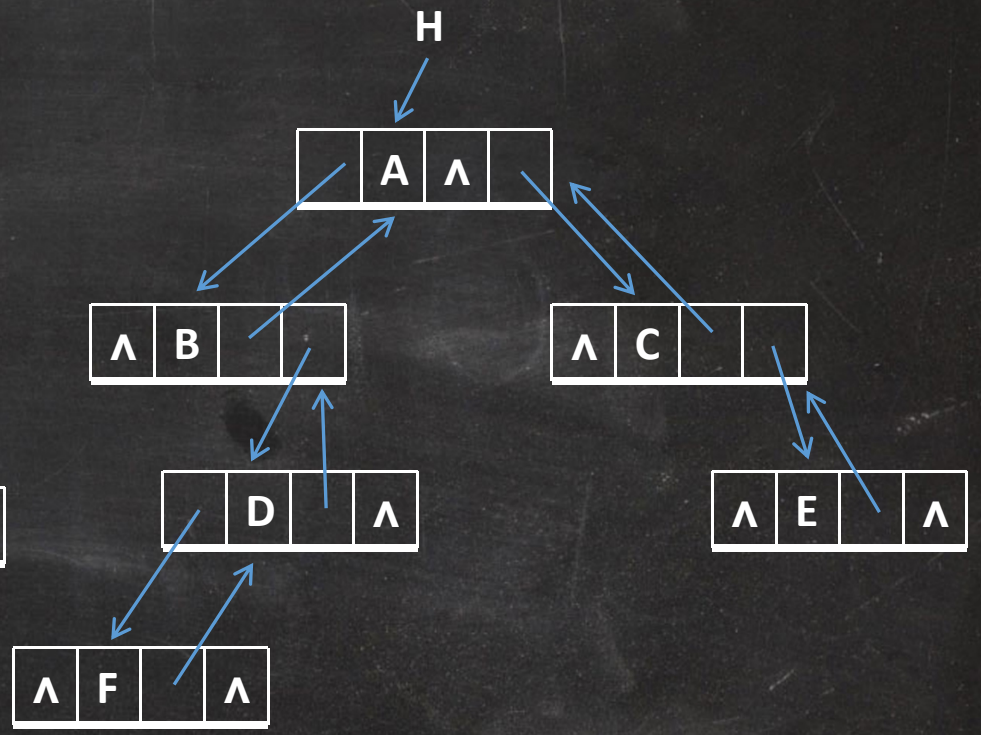
（2）链式存储



（a）二叉树



（b）二叉链表



（a）三叉链表

二叉树的遍历

- 遍历是按某种策略访问树中的每个结点，且仅访问一次。
- 依据访问根结点次序的不同，可分为前序遍历法、中序遍历法、后序遍历法。

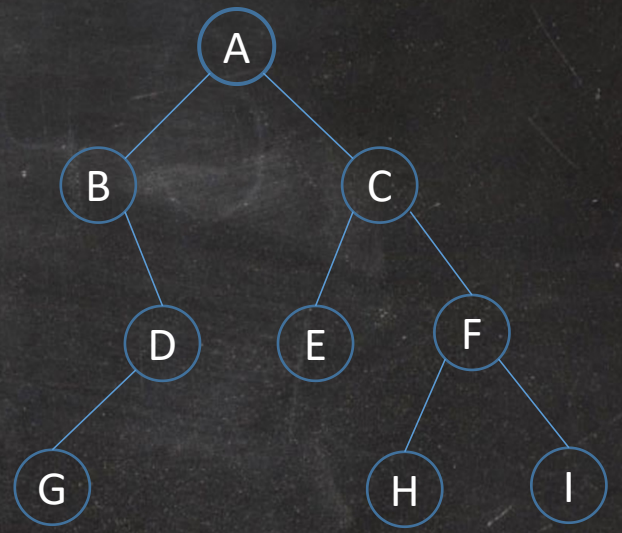
（1）中序遍历法：（左、根、右）

A、中序遍历根的左子树。

B、访问根结点。

C、中序遍历根的右子树。

依此类推。

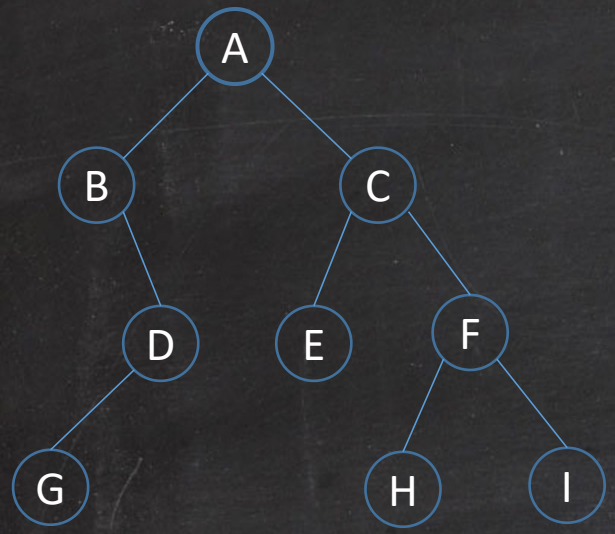


（2）前序遍历法：先访问根结点（根、左、右）

（3）后序遍历法：后访问根结点（左、右，根）

（4）层序遍历法：按层从上至下、每层从左至右的顺序遍历。

紫依



（1）根节点A左边有结点，先遍历左边的子树：{ }A{ }

（2）B左边为空，则先遍历B，然后遍历B的右子树： B{ }A{ }

（3）B的右子树中，又有根结点D，D的左边有结点，则先G后D：BGDA{ }

（4）根结点A的左子树遍历完毕，再是A结点，再是A的右子树

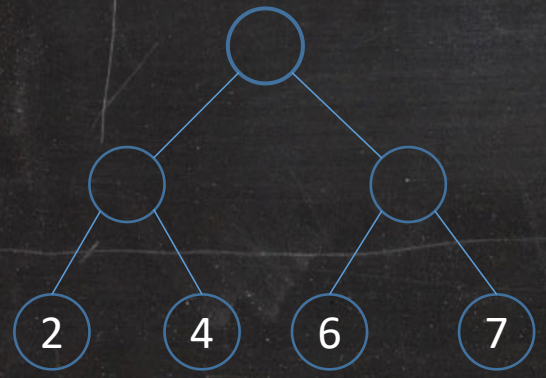
（5）右子树中，C的左边只有一个结点E，则先E后C： BGDAEC{ }

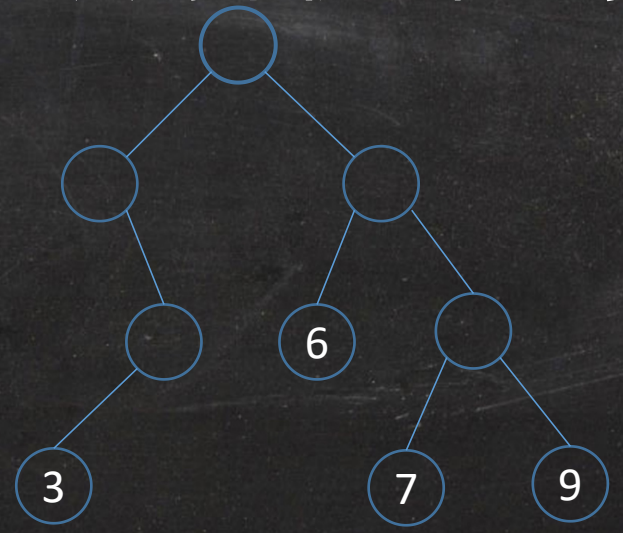
（6）C的右子树中，F的左右都只有一个结点，则是HFI： BGDAECHFI

紫依

最优二叉树

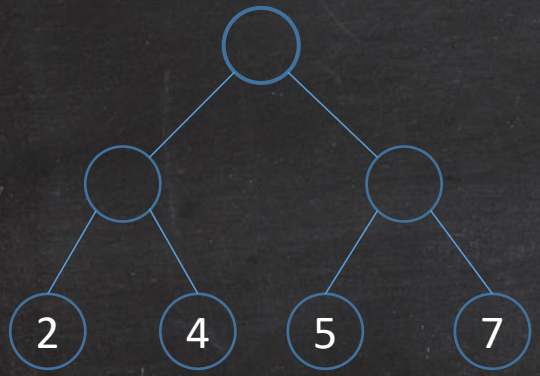
- 最优二叉树又称哈夫曼树，是一类带权路径长度最短的树。
- 权：是一个人为的概念，表示计算机对每个结点的访问频率。
- 路径长度：是每一个结点到根结点的路径的长度。
- 结点的带权路径长度：是指从该结点到根结点之间的路径长度与该结点权的乘积。
- 树的带权路径长度为树中所有叶子结点的带权路径长度之和。



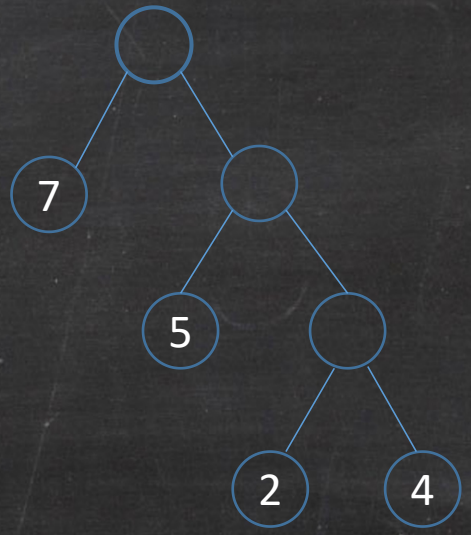


紫依

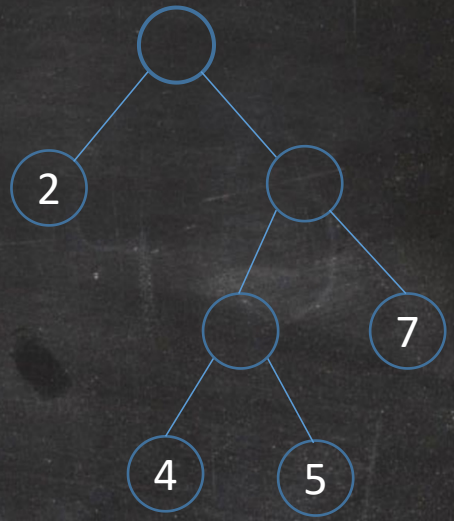
最优二叉树



（a）WPL=(2+4+5+7)×2=36



（b）WPL=(2+4)×3+5×2+7×1=35



（c）WPL=(4+5)×3+7×2+2×1=43

紫依

最优二叉树

- 构造最优二叉树的哈夫曼方法：

（1）根据给定的n个权值{w_1，w_2，...，w_n}构成n棵二叉树的集合F={T_1, T_2，...，T_n}，其中每棵树T_i只有一个带权为w_i的根结点，其左右子树均空。

（2）在F中选取两棵根结点的权值最小的树作为左右子树，构造一棵新的二叉树，置新构造二叉树的根结点的权值为其左、右子树根结点的权值之和。

（3）从F中删除这两棵树，同时将新得到的二叉树加入到F中。

重复（2）（3）步，直到F中只含一棵树时为止，这棵树便是最优二叉树（哈夫曼树）。

- 最优二叉树的一个应用是对字符集中的字符进行编码和译码。

例：现在有一组权值{30，25，15，22，8}，下面我们来演示利用这组权值构造最优二叉树（哈夫曼树）的过程。

二叉查找树

- 二叉查找树又称为二叉排序树。它或者是一棵空树，或者是具有如下性质的二叉树：

（1）若它的左子树非空，则左子树上所有结点的关键码值均小于根结点的关键码值。

（2）若它的右子树非空，则右子树上所有结点的关键码值均大于根结点的关键码值。

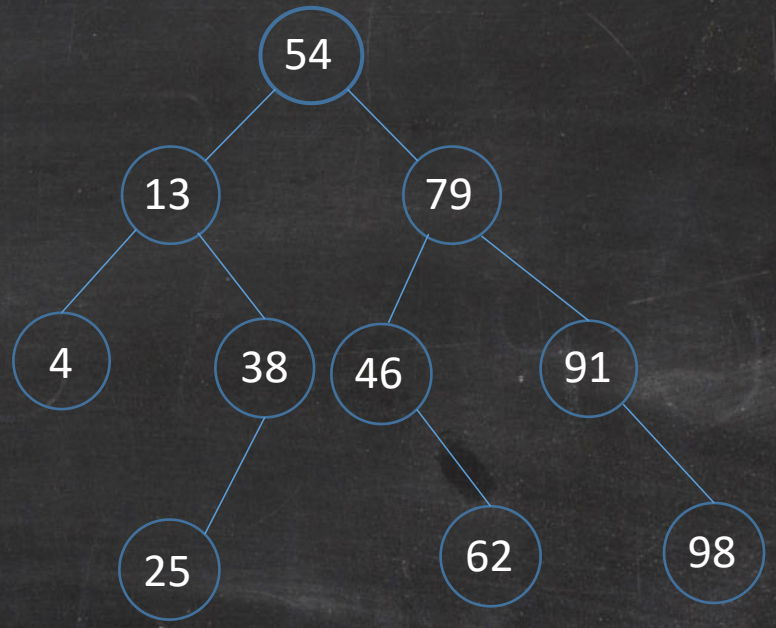
（3）左、右子树本身就是两棵二叉查找树。

- 对二叉查找树进行中序遍历，可得到一个关键码递增有序的结点序列。
- 二叉查找树的作用：使用二叉查找树来查找树中的数值比普通的二叉树更为方便。

二叉查找树

（a）二叉查找树

中序遍历：4,13,25,38,46,54,62,91,98



（b）非二叉查找树

中序遍历：4,13,25,38,54,46,62,79,91,98

紫依