# 数据结构与算法面试题

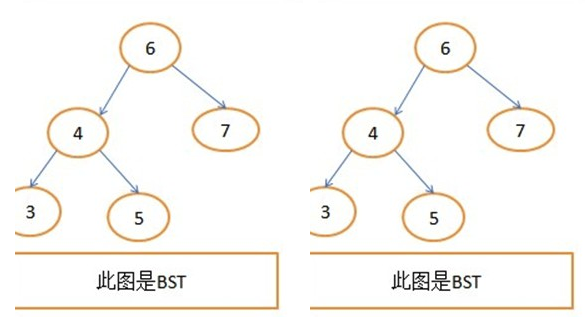
注：红色代表重要信息，绿色代表思考信息。

## 1、二叉排序树

什么是二叉排序树？

二叉排序树(Binary Sort Tree)又称二叉查找树。它或者是一棵空树，或者是有如下性质的二叉树：

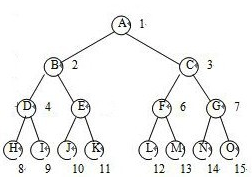
1. 若左子树不空，则左子树上所有节点的值均小于它的根节点的值；
2. 若右子树不空，则右子树上所有节点的值都大于它的节点的值；
3. 左、右子树也都是二叉排序树。



二叉排序树的查找时间复杂度为O(h)，h为树的高度。插入和删除节点都类似于查找操作。所以二叉树的高度越小越好。

二叉排序树的构建

**满二叉树**：一棵深度为k，且有2^k-1个节点的树为满二叉树。



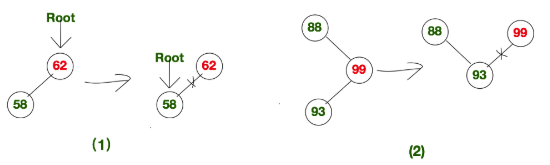
**完全二叉树**：对于深度为k，有n个节点的二叉树，当且仅当其每一个节点都与深度为k的满二叉树中编号从1至n的节点一一对应时称之为完全二叉树。

在构建二叉排序树时，其实就是将每个数插入二叉排序树中，所以这里就转化为了怎样在二叉排序树中插入节点。首先与根节点进行比较，如果小于根节点往左子树插，如果大于根节点往右子树插，重复该操作。所以新给定的节点一定是一个叶子节点。

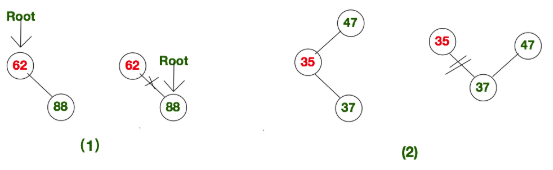
**删除节点**：删除节点要稍复杂一些，因为不单单只将节点删除，而且还要保证删除之后整棵树还是二叉查找树，所以删除节点后还需要做相应的调整。

删除节点要分三种情况：

1. 删除节点只有左子树，直接将该节点的左子树的根节点覆盖删除节点



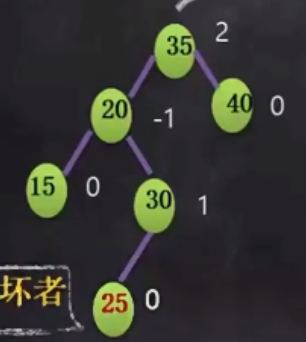
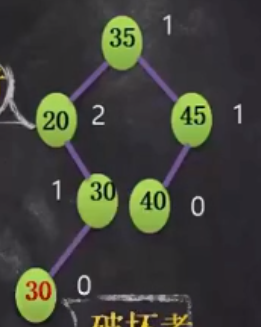
1. 删除节点只有右子树，直接将该节点的右子树的根节点覆盖删除节点



1. 删除节点既有左子树又有右子树，首先查找右子树中最小的那个节点，显然就是最左下角的那个值，然后用该值覆盖要删除的节点，然后删除右子树中最小节点。

## 2、平衡二叉树

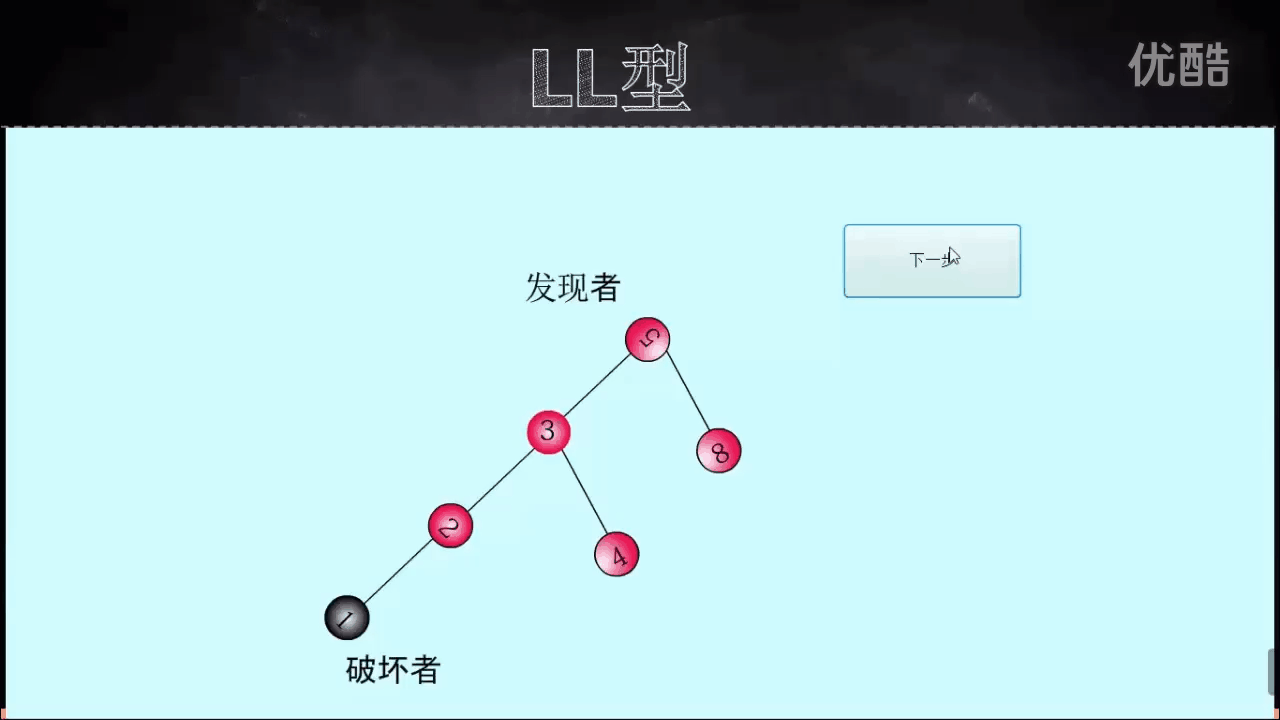
**平衡因子**：某节点的左子树与右子树的高度差即为平衡因子。

平衡二叉树是一棵空树或者它的左右两个子树的高度差的绝对值不超过1，并且左右两棵子树都是一棵平衡二叉树。

平衡二叉树的旋转分为LL型、LR型、RL型、RR型。旋转的过程：首先如下图，最小不平衡树的根节点为5，插入的节点的位置为5节点的左孩子的左子树，所以为LL型。这时候又发现整棵树左边较重右边较轻，所以将右边向右旋转(顺时针旋转)，如下图。旋转之后发现4节点发生冲突，重新调整4节点，作为5节点的左孩子，这时旋转完毕。

LL型的旋转

其他旋转类型类型类似，只不过有的类型旋转一次之后发现仍然不平衡，还需要继续旋转。详情见视频。