BST树，是二叉搜索树：

二叉查找树（Binary Search Tree），（又：[二叉搜索树](https://baike.baidu.com/item/%E4%BA%8C%E5%8F%89%E6%90%9C%E7%B4%A2%E6%A0%91)，二叉排序树）它或者是一棵空树，或者是具有下列性质的[二叉树](https://baike.baidu.com/item/%E4%BA%8C%E5%8F%89%E6%A0%91)： 若它的左子树不空，则左子树上所有结点的值均小于它的根结点的值； 若它的右子树不空，则右子树上所有结点的值均大于它的根结点的值； 它的左、右子树也分别为[二叉排序树](https://baike.baidu.com/item/%E4%BA%8C%E5%8F%89%E6%8E%92%E5%BA%8F%E6%A0%91)。

* 每个点的值d都不同，因此若插入时候某个点的值在树中存在，则插入失败
* 左边的d一定小于当前点，右边点一定大于当前点
* 中序遍历可得到有序序列

## 查找算法：

查找算法基础，插入删除也会用到，查找值是d0的点如何查？

设当前点now是根节点head,如果：

当前节点小于d0,向右找，

当前节点大于d0,向左找，

当前节点等于d0,说明找到了，

当前节点是空，穷途末路，说明不存在

## 插入结点的算法：

设当前点now是根节点head, 如果：

当前节点小于d0,向右找，

当前节点大于d0,向左找，

当前节点等于d0,根据BST树性质，不能有同样节点，插入错误

当前节点是空，可以插入，那就插入吧

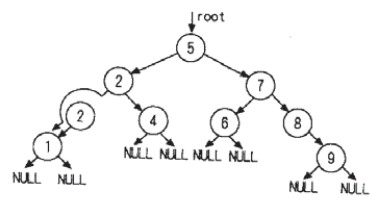
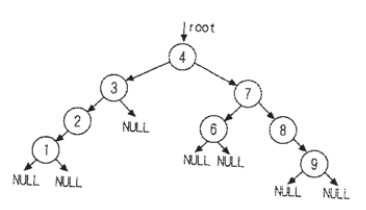
## 删除结点的算法：

对于左边或者右边有一个空的节点(或者左右都空)，只要直接删除再把下面连上即可

删除2，就得到，

如果两边都有节点：

寻找要删除的节点node的所有子节点中比它大一点，或者比它小一点的，和node交换，再删除，如下，删除根节点5

比如找比它小的点，只需要遍历node->left的每个最右的节点走，知道走不了位置。