# AVL树总结

**AVL树是二叉查找树的一种优化，能将链状的二叉查找树几乎平均地储存下来，从而减少搜索使用的时间。**

**AVL树是空树，或满足以下定义的树：**

**1、左右子树都是AVL树；（递归定义）**

**2、左右子树高度之差不超过1；**

**定义平衡因子：**

**左子树高度-右子树高度，当平衡因子大于等于2时，我们就称这棵树不平衡，需要通过旋转让它重新平衡。**

**获取节点高度：**

**int h(int rt){**

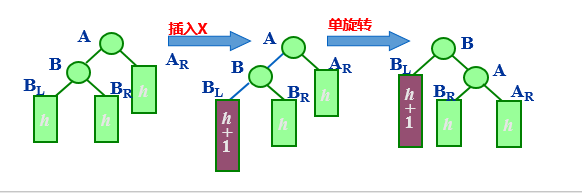
**if(rt==0) return -1;//这里return-1的原因后面阐释**

**return no[rt].height;**

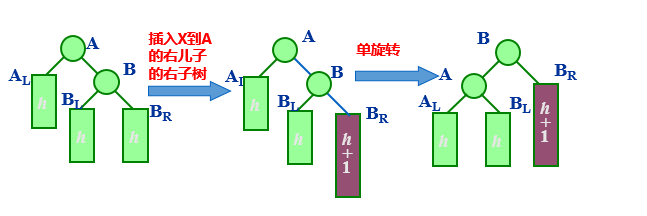
**}**

**单旋转**

**“左左”当根节点的左子树的左儿子与根节点的右儿子不平衡时**



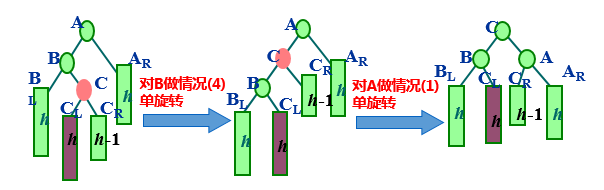
**我们通过单旋转使平衡树符合该树的性质**



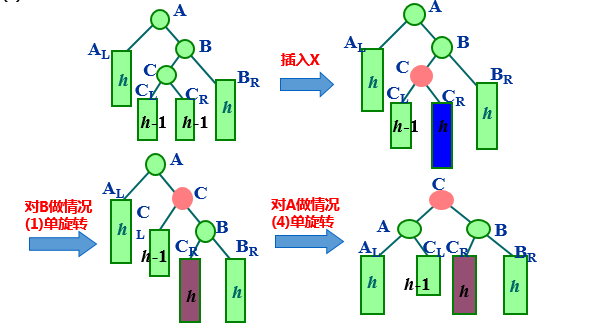
**“右右”方法类似，与左左对称**

**双旋转**

**“左右”当根节点的左子树的右儿子与根节点的右儿子不平衡时**



**旋转两次即可使这棵树平衡**



**“右左”同理。**

**插入操作：先正常插入指定结点，再判断原树是否平衡，不平衡要根据具体情况旋转使原树平衡**