**红黑树性质：**

**每个节点不是红色就是黑色**

**更结点是黑色**

**红色节点的孩子是黑色的**

**从根到任意null结点的路径上，黑色结点的数量是相同的**

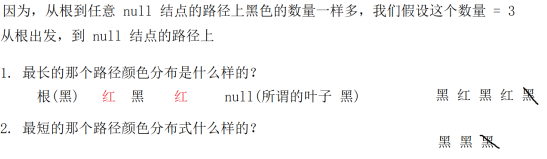
**每个叶子结点是黑色的（叶子结点指的是NULL结点）**

总结：一头一尾是黑色

红色不能挨红色

黑色数量一样多

**这些特征保证了什么？**



保证了红黑树中最长的路径不会超过最短路径的2倍

保证了树的平衡

**红黑树的插入过程：**

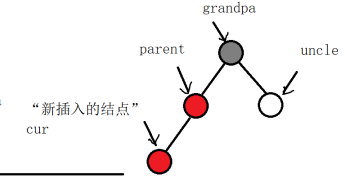
按照普通搜索树进行插入

**新插入的结点默认为红色**，检查是否破坏规则（红色不能挨红色）

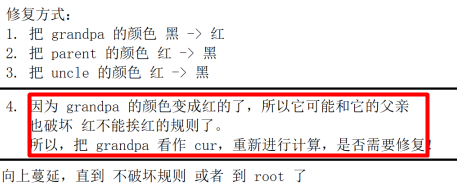
如果没有破坏插入结束

否则进行修复

如果在过程中，**根结点变为红色，则将根结点改为黑色**



**1.uncle存在且为红色**



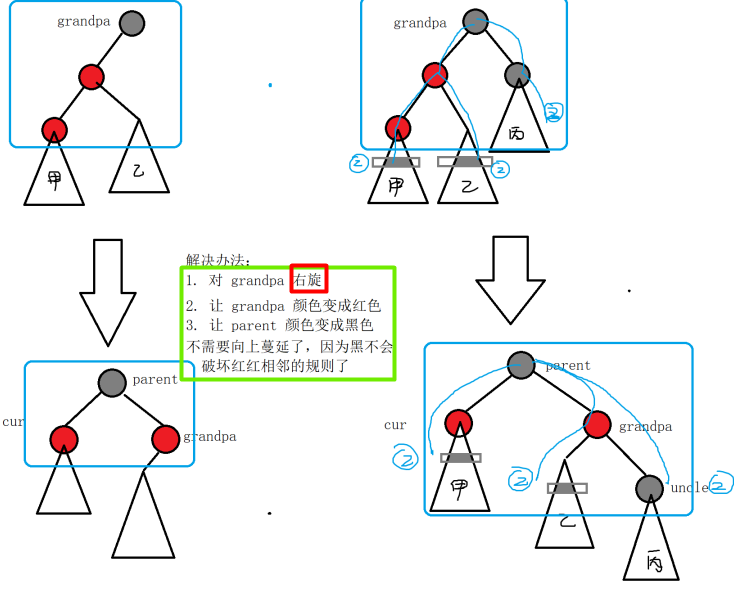
**2.uncle不存在/uncle为黑色**

**parent在grandpa的左边，cur在parent的左边**

对grandpa进行**右旋**

将grandpa改为红色，parent改为黑色

因为此时的grandpa为黑色，没有破坏规则，所以不需要向上蔓延



**cur在parent的右边，parent在grandpa右边**

grandpa进行**左旋**

将grandpa改为红色，parent改为黑色

因为此时的grandpa为黑色，没有破坏规则，所以不需要向上蔓延

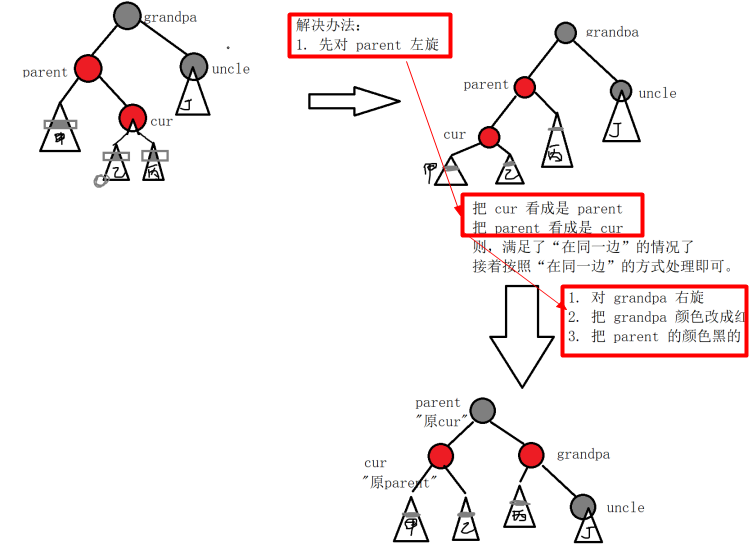
**parent在grandpa左边，cur在parent右边——不在同一边**

先对parent进行**左旋**

原来的cur是现在的parent，所以再对parent进行**右旋**

将grandpa改为红色

将parent改为黑色



**parent在grandpa右边，cur在parent左边**

先对parent进行**右旋**

原来的cur是现在的parent，所以再对parent进行**左旋**

将grandpa改为红色

将parent改为黑色

**AVL与红黑树区别：**

