红黑树的删除其实和二叉搜索树是很类似的，除了要加一个颜色的调整。

所以这里不做红黑树的删除掌握，但必须掌握二叉搜索的全部操作，有兴趣的同学可以看我这个资料。

**红黑树的性质**

1.每个结点不是红色就是黑色

2.不可能有连在一起的红色结点（黑色的就可以），每个叶子节点都是黑色的空节点（NIL），也就是说，叶子节点不存储数据

3.根结点都是黑色 root

4.每个节点，从该节点到达其可达叶子节点的所有路径，都包含相同数目的黑色节点；

**删除分情况讨论：**

1. 删除节点只有一个子节点：（直接删除此节点，然后将孩子节点代替父亲节点）

（1）如果删除的节点是红色的，直接删除就行，不会影响；

（2）如果删除节点是黑色的，那么就要考虑孩子节点了(只有一个孩子节点，考虑红黑树平衡条件，根据性质4这个孩子节点肯定是红色的，不然左右孩子的黑色路径数就不对了)，所以，直接将孩子节点代替父亲节点后，将这这个节点变成黑色。

2.删除节点没有子节点：（直接删除节点）

（1）如果删除的节点是红色的，那么直接删除

（2）如果删除的节点是黑色的。那这个就要重新来搞平衡了

3.删除节点有两个子节点：（找到这个节点的后继节点，然后将后继节点代替这个节点）这样其实就转换成情况1或者2了。

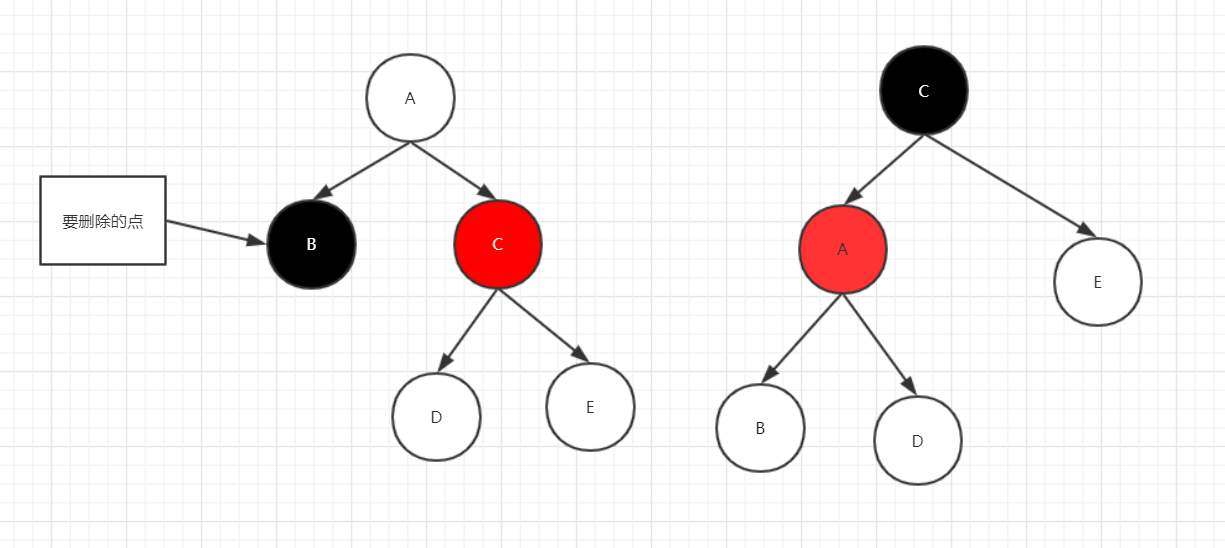
综上所述，红黑树删除主要的就是针对情况2复杂，情况1就是变换下颜色即可。情况3最终是会转化成情况1和2的；我参考的是算法导论

那么我们现在就来看情况2：

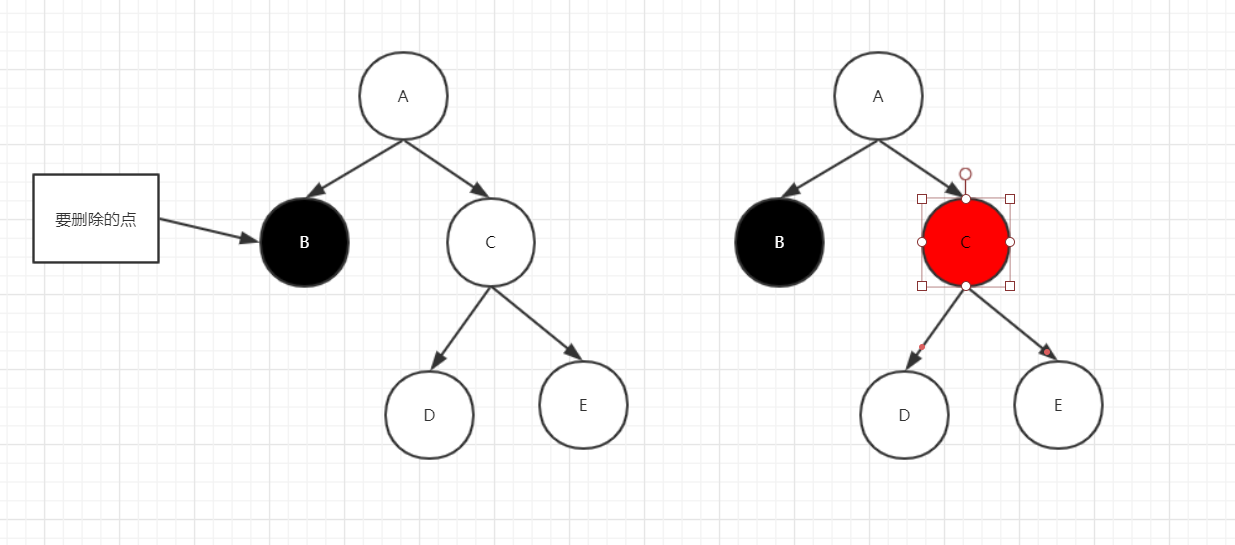
这里我写了半天，也查了一些资料，发现总是没有完全讲清楚。要是有问题的大家可以提出来，或者你有更好的写法也可以放到知识星球：

情况2的修复：

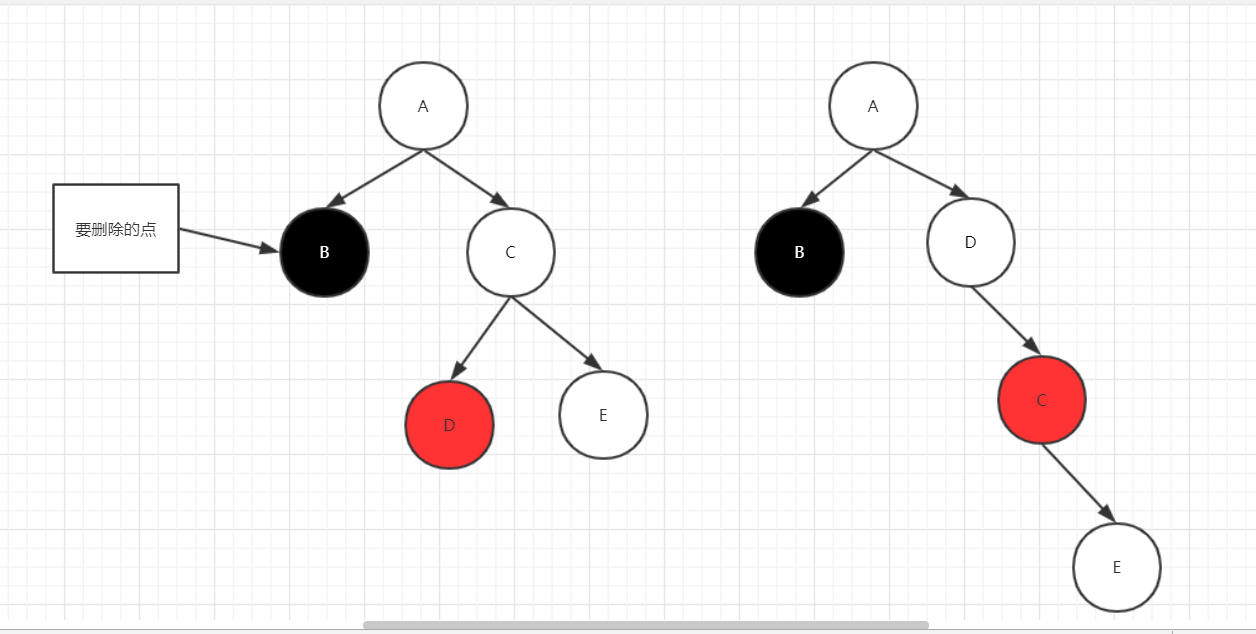
* 1. 待删除节点是黑色的，它的兄弟节点是红色的，根据性质4 父节点必须为黑色，否则有一边的孩子结点就多了一个黑色了。操作：交换兄弟和父亲节点的颜色，以父节点左旋，重新设置兄弟节点。



* 1. 待删除的节点的兄弟节点是黑色的节点，且兄弟节点的子节点都是黑色的。操作：兄弟结点变为红色，指针指向父节点继续处理。



* 1. 待删除的兄弟节点是黑色的节点，且兄弟节点的左子树是红色的，右子树是黑色的。操作：兄弟节点变红，兄弟节点的左节点变黑，以兄弟节点右旋。



（4）待删除的节点的兄弟节点是黑色的节点，兄弟节点的右节点为红色，左节点为任意颜色的。操作：交换父节点和兄弟节点的颜色，兄弟节点的右节点设为黑色，以父节点左旋。

