常见的数据结构

栈

队列

数组

链表

二叉树

二叉查找树

平衡二叉树

红黑树

栈：后进先出，先进后出

数据进入栈模型得过程称为：压/进栈

数据离开栈模型得过程称为：弹/出栈

队列：先进先出，后进后出

数据从后端进入队列模型的过程称为：入队列

数据从前端离开队列模型得过程称为：出队列

数组：

查询快，增删慢的模型

查询速度快：查询数据通过地址值合索引定位，查询任意数据消耗时间相同

删除效率低：要将原始数据删除，同时后面面每个数据前移

添加效率极低：添加位置后得每个数据后移，再添加数据

链表：

查询慢，无论查询哪个数据都要从头开始找，增删相对快

单向链表=>链表中的节点是独立的对象，在内存中是不连续的，每个结点包含数据值和下一个结点的地址

双向链表=>每个结点包含数据和上一个结点的地址和下一个结点的地址

树=>

二叉树的内部结构：父节点地址、值，左子节点地址，右子节点地址

二叉树：每个节点的度≤2

二叉查找树，又称二叉排序树或二叉搜索树：任意节点左子树上的值都小于当前节点，任意节点右子树上的指都大于当前结点的值

平衡二叉树：任意节点左右子树高度差不超过1

平衡二叉树的旋转机制=>

平衡二叉树的触发时机：当添加一个节点之后，该树不再是一棵平衡二叉树。

确定支点：从添加的节点开始，不断地往父节点找不平衡的点

需要旋转的四种情况：

1. 左左=>一次右旋
2. 左右=>先局部左旋在整体右旋
3. 右右=>一次左旋
4. 右左=>先局部右旋在整体左旋

左旋步骤：以不平衡的点作为支点，把支点左旋降级称为左子节点，晋升原来的右子节点

右旋步骤：和左旋相反

二叉树的遍历方式：

1. 前序遍历：当前节点，左子节点，右子节点
2. 中序遍历：左子节点，当前节点，右子节点
3. 后序遍历：左子节点，右子节点，当前节点
4. 层序遍历：一层一层的遍历

红黑树

·红黑树是一种自平衡的二叉查找树，是计算机科学中用到的一种数据结构

·自1972年出现，当时被称为平衡二叉B树，后来1978年被改名为红黑树

·他是一种特殊的二叉查找树，红黑树的每一个节点上都有存储位表示节点的颜色

·每一个节点可以是红或者黑；红黑树不是高度平衡的，他的平衡是通过“红黑规则”进行实现的

红黑树和平衡二叉树的区别：

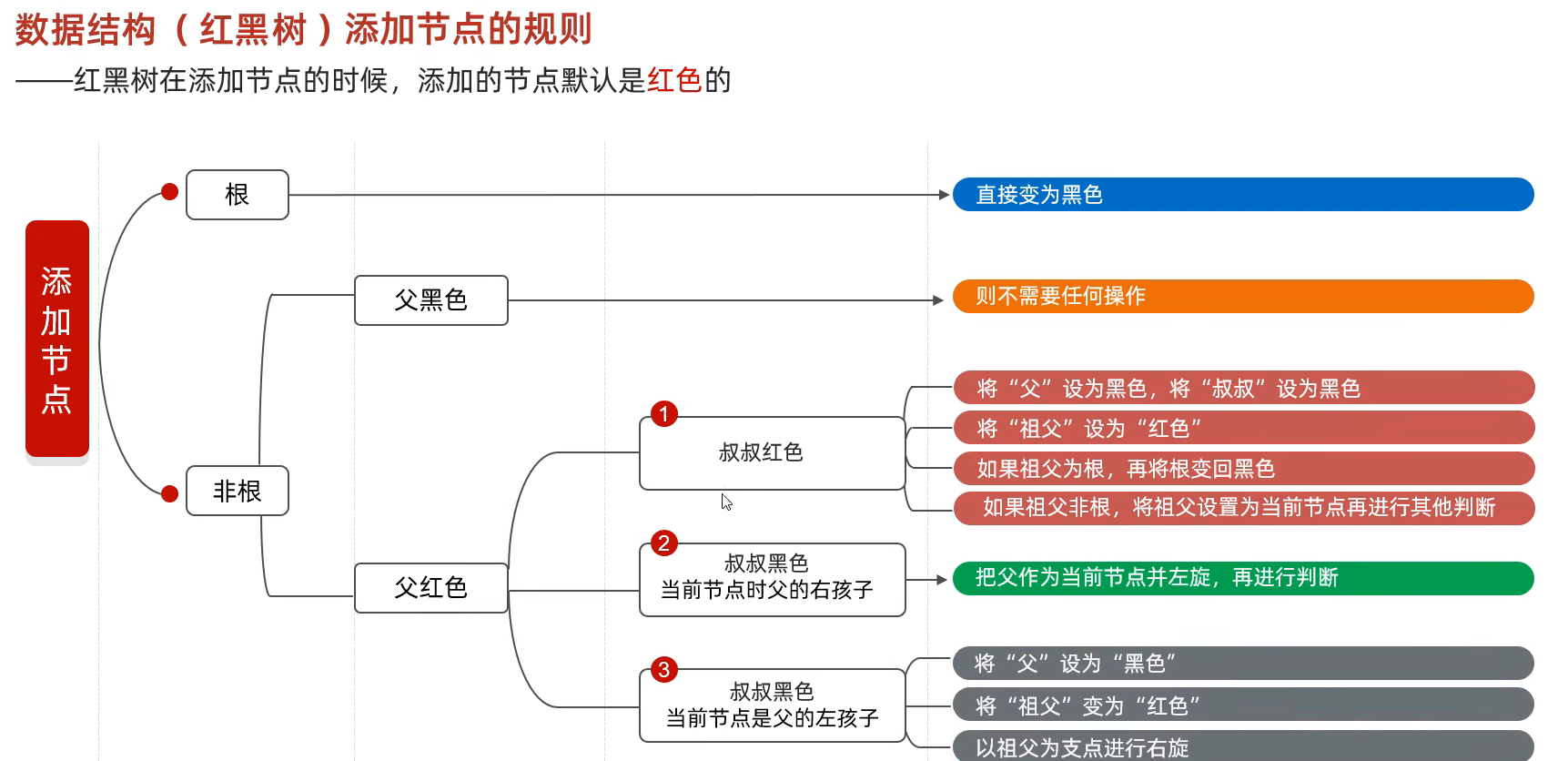
平衡二叉树：高度平衡、当左右子树高度差超过1时，通过旋转保持平衡

红黑树：是一个二叉查找树，但是不是高度平衡，条件=>特有的红黑规则

红黑规则：

1. 每一个节点或是红色的，或者是黑色的
2. 根节点必须是黑色的
3. 如果一个节点没有子节点或者父节点，则该节点相应的指针属性为Nil，这些Nil视为叶节点，每个叶节点是黑色的
4. 如果某一节点是红色，那么他的子节点必须是黑色（不能出现两个红色节点相连的情况）
5. 对每一个节点，从该节点到其所有后代叶节点的简单路径上，均包含相同数目的黑色节点

添加节点默认颜色是红色



**红黑树增删改查性能都很好**