本节内容

红黑树

(Red-Black Tree)

RBT

为什么要发明 红黑树？

		BST	AVL Tree	Red-Black Tree
生日		1960	1962	1972
时间复杂度	Search（查）	$O(n)$	$O(log_2 n)$	$O(log_2 n)$
	Insert（插）	$O(n)$	$O(log_2 n)$	$O(log_2 n)$
	Delete（删）	$O(n)$	$O(log_2 n)$	$O(log_2 n)$

平衡二叉树 AVL：插入/删除 很容易破坏“平衡”特性，需要频繁调整树的形态。如：插入操作导致不平衡，则需要先计算平衡因子，找到最小不平衡子树（时间开销大），再进行 LL/RR/LR/RL 调整

红黑树 RBT：插入/删除 很多时候不会破坏“红黑”特性，无需频繁调整树的形态。即便需要调整，一般都可以在常数级时间内完成

平衡二叉树：适用于以查为主、很少插入/删除的场景

红黑树：适用于频繁插入、删除的场景，实用性更强

红黑树大概会怎么考？

红黑树的定义、性质——选择题

红黑树的插入/删除——要能手绘插入过程（不太可能考代码，略复杂），删除操作也比较麻烦，也许不考

2015真题

4．现有一棵无重复关键字的平衡二叉树（AVL 树），对其进行中序遍历可得到一个降序序列。下列关于该平衡二叉树的叙述中，正确的是_____。

A．根结点的度一定为 2　　B．树中最小元素一定是叶结点

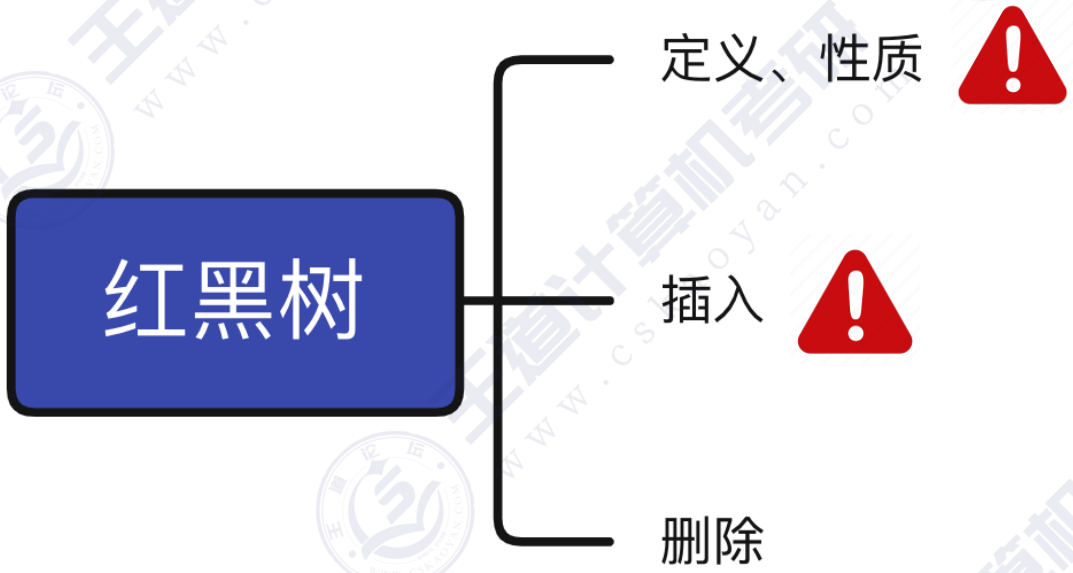
C．最后插入的元素一定是叶结点　　D．树中最大元素一定是无左子树

2013真题

3．若将关键字 1, 2, 3, 4, 5, 6, 7 依次插入到初始为空的平衡二叉树 T 中，则 T 中平衡因子为 0 的分支结点的个数是_____。

A．0　　B．1　　C．2　　D．3

知识总览



红黑树的定义

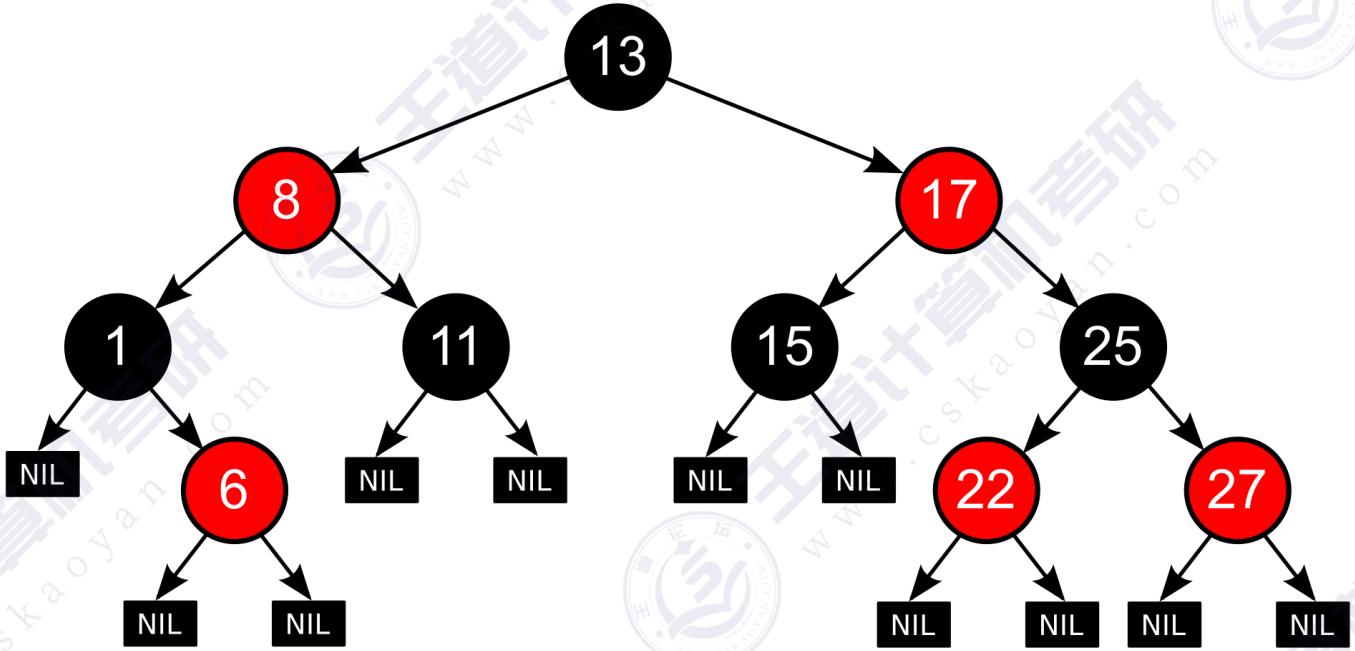
红黑树是二叉排序树 → 左子树结点值 ≤ 根结点值 ≤ 右子树结点值

与普通BST相比，有什么要求 →

①每个结点或是红色，或是黑色的
②根节点是黑色的
③叶结点（外部结点、NULL结点、失败结点）均是黑色的
④不存在两个相邻的红结点（即红结点的父节点和孩子结点均是黑色）
⑤对每个结点，从该节点到任一叶结点的简单路径上，所含黑结点的数目相同

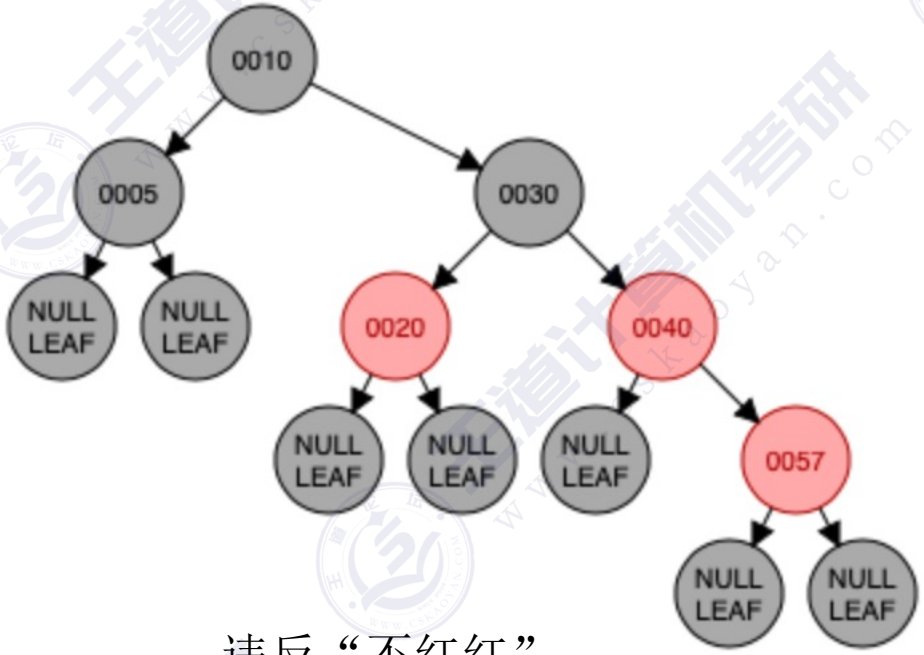

```
struct RBnode {       // 红黑树的结点定义
  int key;            // 关键字的值
  RBnode* parent;     // 父节点指针
  RBnode* lChild;     // 左孩子指针
  RBnode* rChild;     // 右孩子指针
  int color;          // 结点颜色，如：可用 0/1 表示 黑/红，也可使用枚举型enum表示颜色
};
```

实例：一棵红黑树

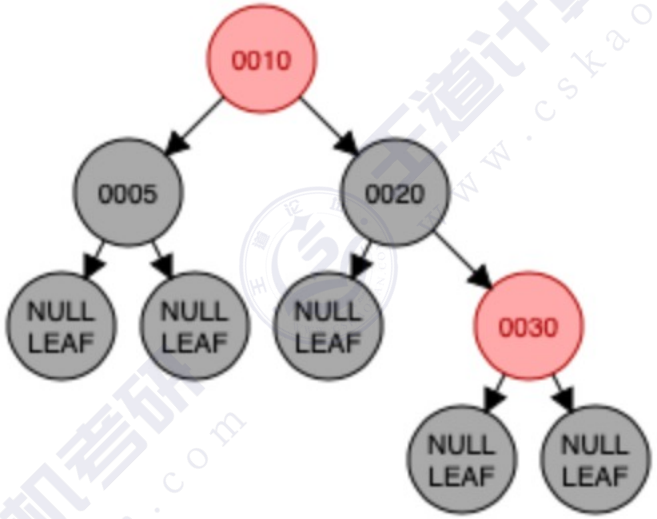

13
8
17
1
11
15
25
NIL
6
NIL
NIL
NIL
NIL
22
27
NIL
NIL
NIL
NIL
NIL
NIL


花里胡哨的肯定

练习：是否符合红黑树要求?

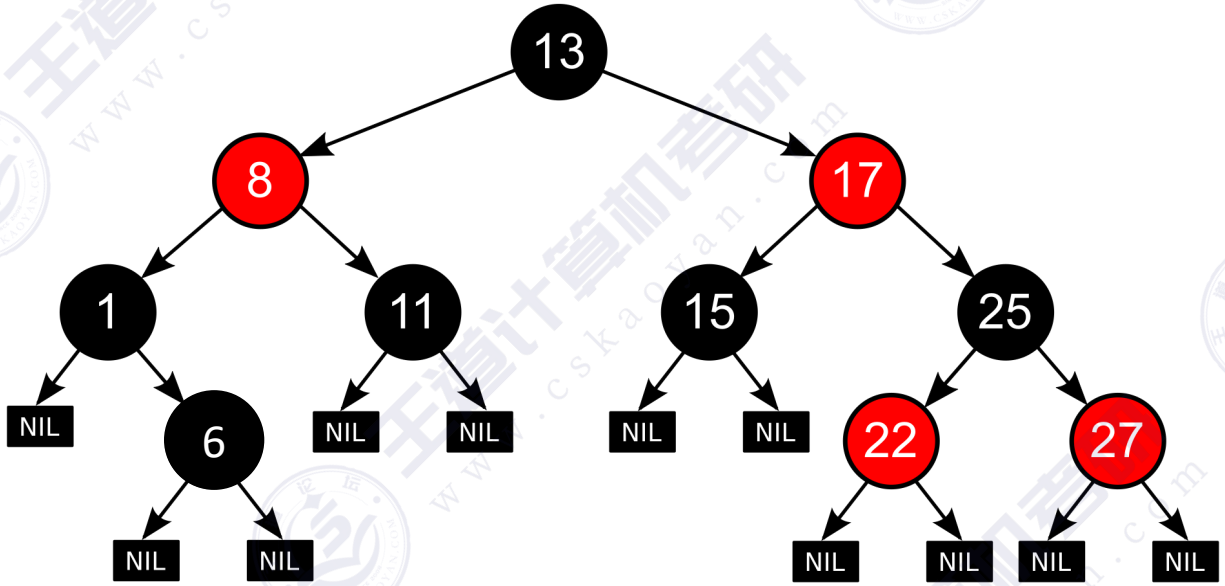


违反“不红红”



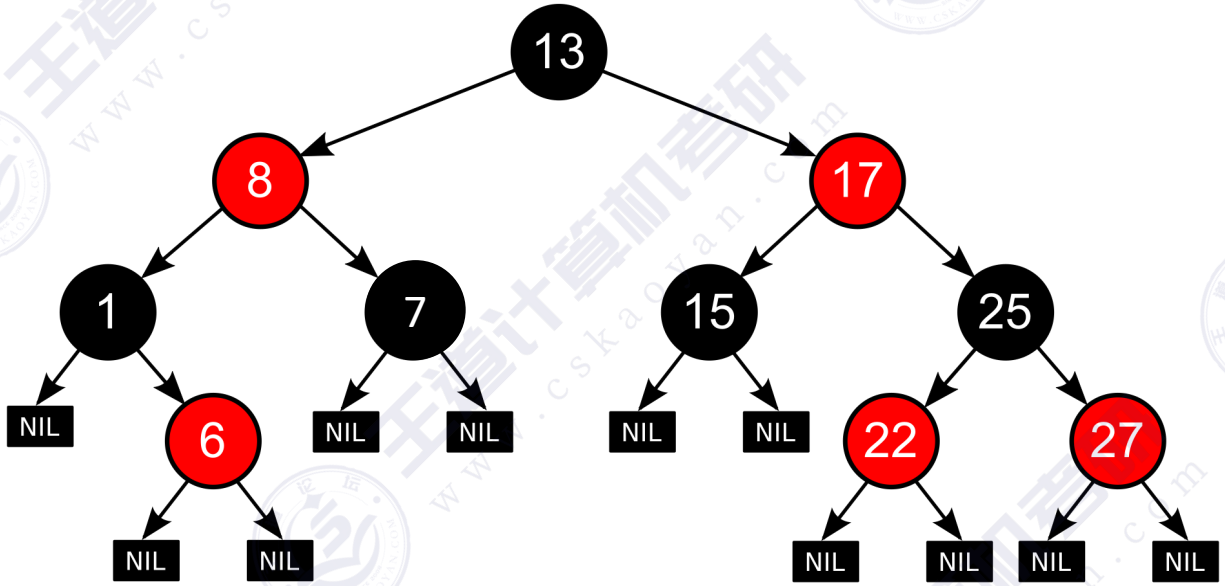
违反“根叶黑”

练习：是否符合红黑树要求?



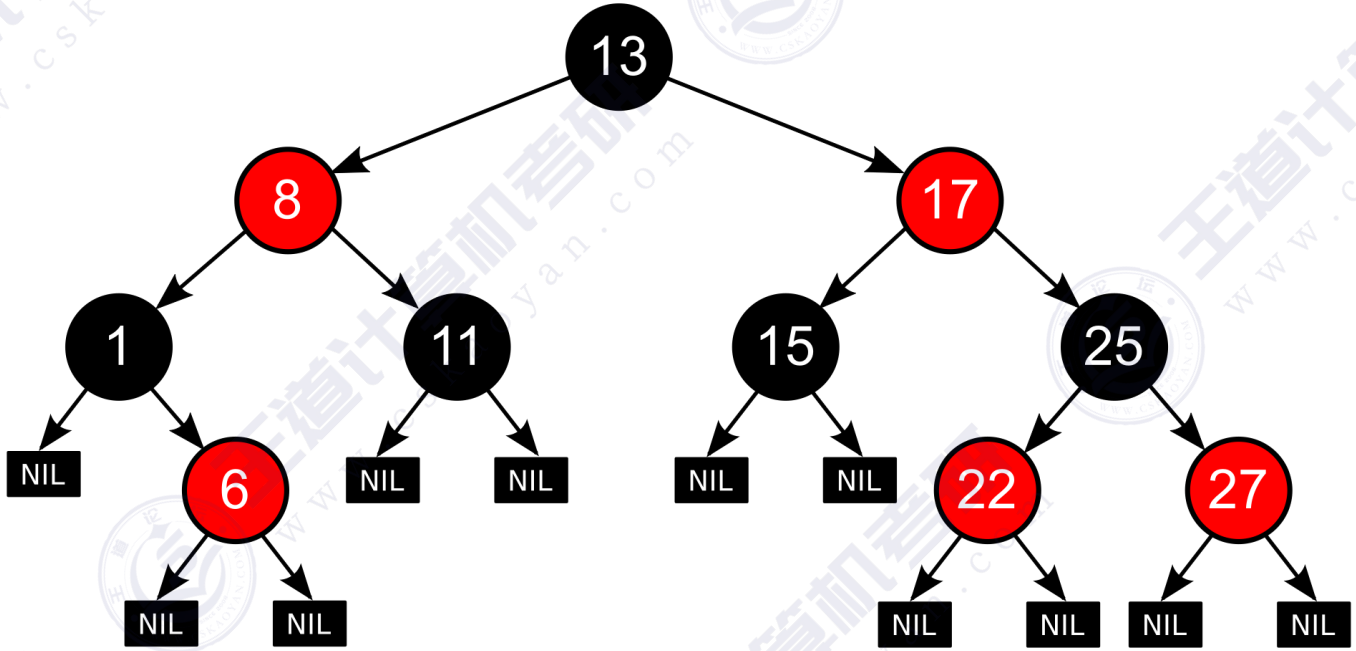
违反“黑路同”

练习：是否符合红黑树要求?



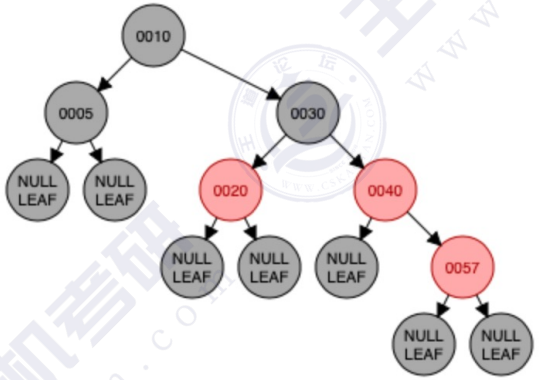
违反“左根右”

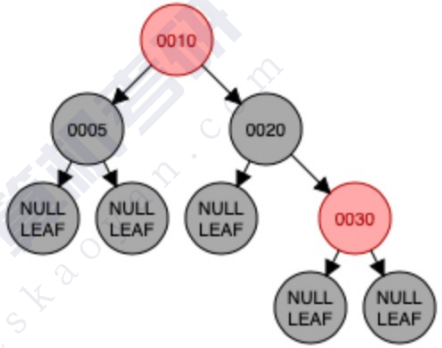
练习：是否符合红黑树要求？

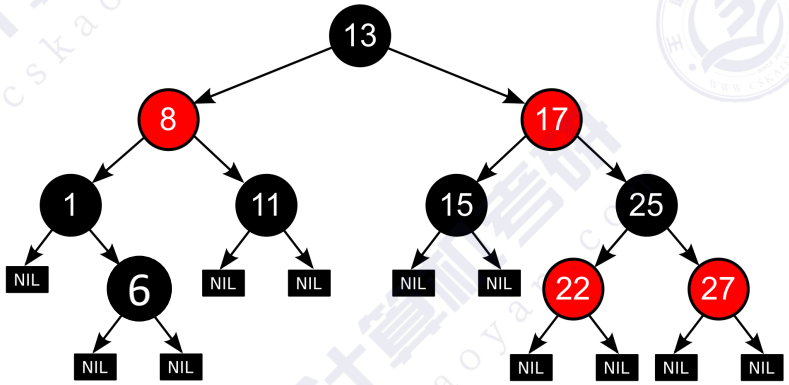


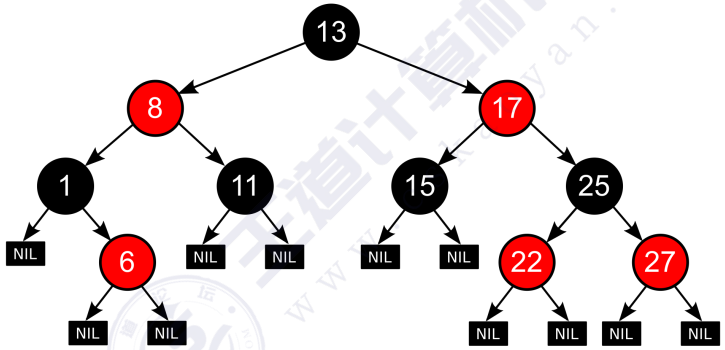
一种可能的出题思路

下面这些选项中，满足“红黑树”定义的是（　）

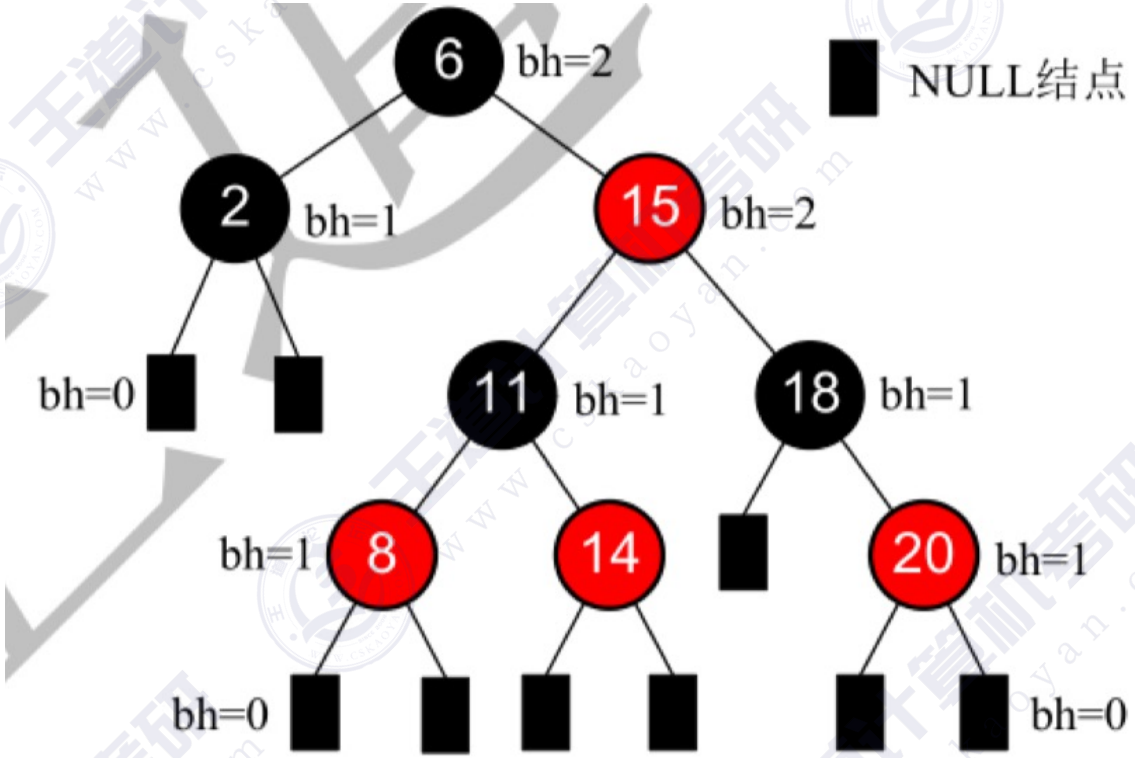








补充概念：结点的“黑高”



结点的黑高 bh —— 从某结点出发（不含该结点）到达任一空叶结点的路径上黑结点总数

红黑树的定义→性质

红黑树是二叉排序树 → 左子树结点值 ≤ 根结点值 ≤ 右子树结点值

与普通BST相比，有什么要求 →

①每个结点或是红色，或是黑色的
②根节点是黑色的
③叶结点（外部结点、NULL结点、失败结点）均是黑色的
④不存在两个相邻的红结点（即红结点的父节点和孩子结点均是黑色）
⑤对每个结点，从该节点到任一叶结点的简单路径上，所含黑结点的数目相同

左根右，根叶黑
不红红，黑路同

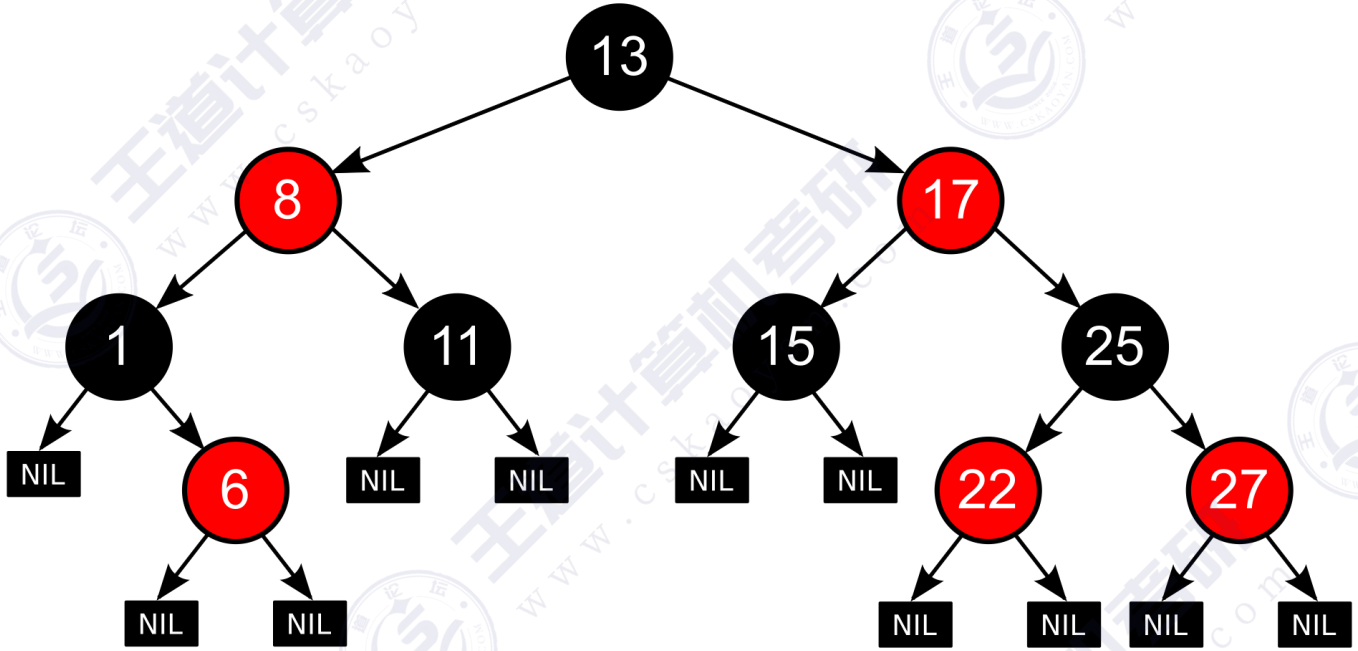
张口就是freestyle

→ 性质1：从根节点到叶结点的最长路径不大于最短路径的2倍
性质2：有n个内部节点的红黑树高度 $h \leq 2log_2(n+1)$

→ 红黑树查找操作时间复杂度 = $O(log_2 n)$

查找效率与AVL树同等数量级

红黑树的查找



与 BST、AVL 相同，从根出发，左小右大，若查找到一个空叶节点，则查找失败

复习：平均查找长度 ASL，查找成功、查找失败时分别怎么计算？