

本节内容

平衡二叉树
(AVL)

王道考研/CSKAOYAN.COM

1

公众号： 考研发条 一手课程！

知识总览

平衡二叉树

定义

插入操作

插入新结点后如何调整“不平衡”问题

查找效率分析

王道考研/CSKAOYAN.COM

2

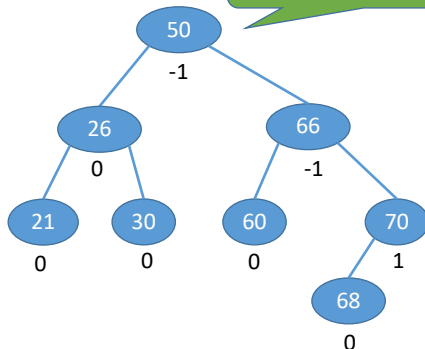
平衡二叉树的定义

G. M. Adelson-Velsky和
E. M. Landis

平衡二叉树（Balanced Binary Tree），简称平衡树（AVL树）——树上任一结点的左子树和右子树的高度之差不超过1。

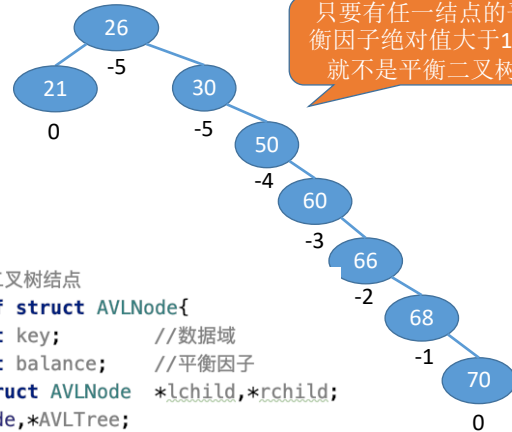
结点的平衡因子=左子树高-右子树高。

平衡二叉树结点的平衡因子的值只可能是-1、0或1。



```
//平衡二叉树结点
typedef struct AVLNode{
    int key;           //数据域
    int balance;       //平衡因子
    struct AVLNode *lchild,*rchild;
}AVLNode,*AVLTree;
```

只要有任一结点的平衡因子绝对值大于1，就不是平衡二叉树



王道考研/CSKAOYAN.COM

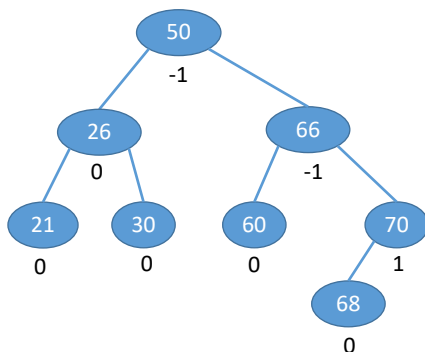
3

公众号： 考研发条

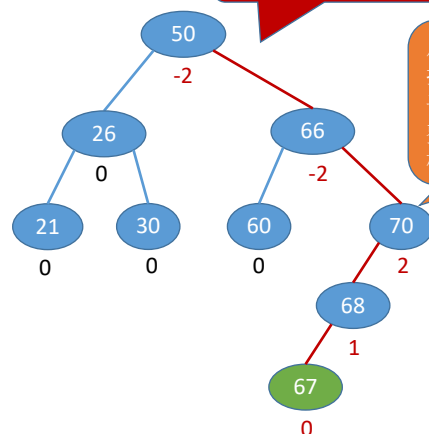
一手课程！

平衡二叉树的插入

在二叉排序树中插入新结点后，如何保持平衡？



插入67



查找路径上的所有结点都有可能受到影响

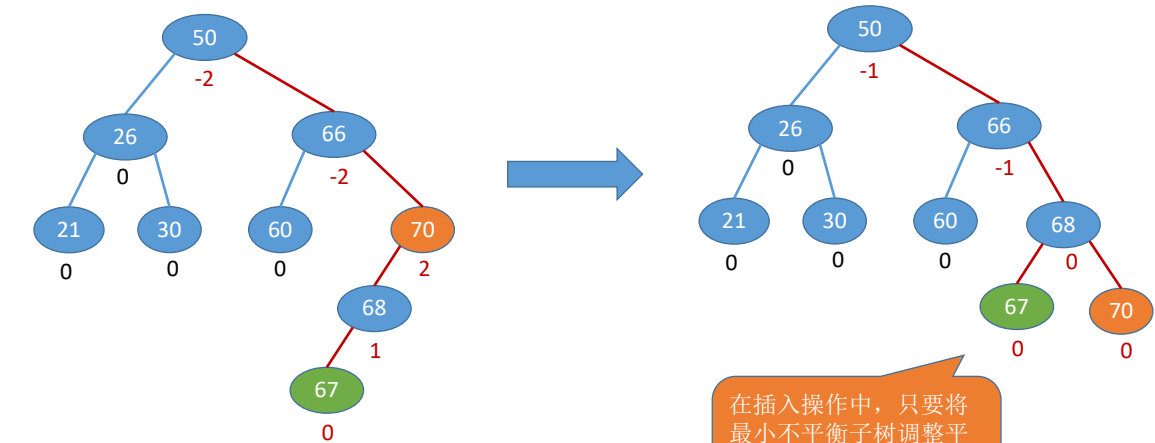
从插入点往回找到第一个不平衡结点，调整以该结点为根的子树

每次调整的对象都是“最小不平衡子树”

王道考研/CSKAOYAN.COM

4

平衡二叉树的插入



王道考研/CSKAOYAN.COM

5

公众号： 考研发条

一手课程！

调整最小不平衡子树

调整最小不平衡子树A

LL ⊖ 在A的左孩子的左子树中插入导致不平衡

RR ⊖ 在A的右孩子的右子树中插入导致不平衡

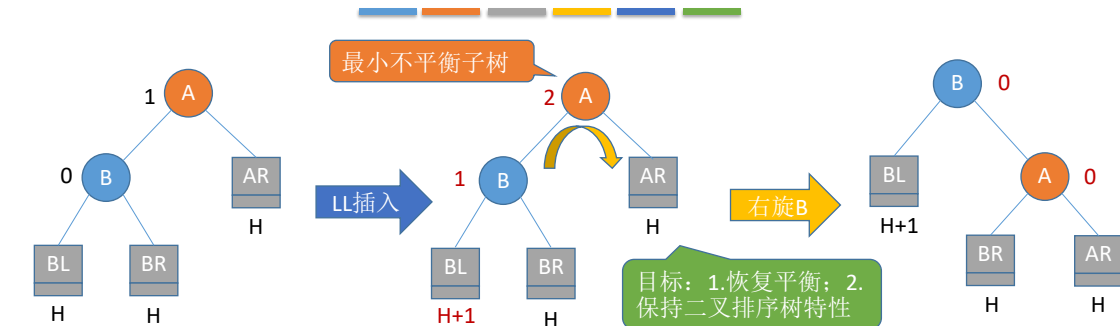
LR ⊖ 在A的左孩子的右子树中插入导致不平衡

RL ⊖ 在A的右孩子的左子树中插入导致不平衡

王道考研/CSKAOYAN.COM

6

调整最小不平衡子树 (LL)



二叉排序树的特性：左子树结点值 < 根结点值 < 右子树结点值

$$BL < B < BR < A < AR$$

1) LL平衡旋转（右单旋转）。由于在结点A的左孩子（L）的左子树（L）上插入了新结点，A的平衡因子由1增至2，导致以A为根的子树失去平衡，需要一次向右的旋转操作。将A的左孩子B向右旋转代替A成为根结点，将A结点向右下旋转成为B的右子树的根结点，而B的原右子树则作为A结点的左子树。

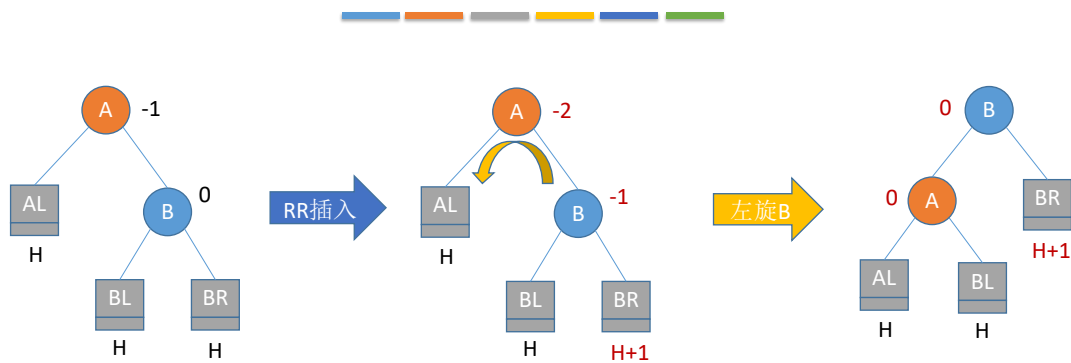
王道考研/CSKAOYAN.COM

7

公众号： 考研发条

一手课程！

调整最小不平衡子树 (RR)



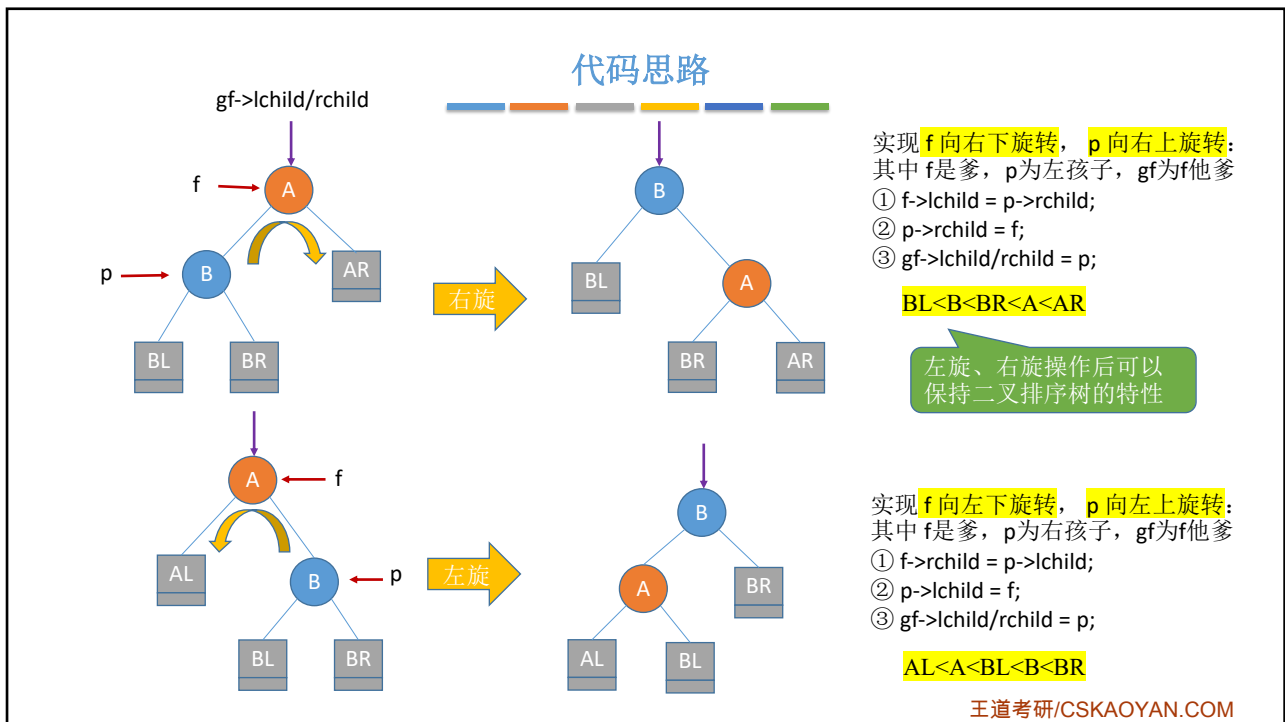
二叉排序树的特性：左子树结点值 < 根结点值 < 右子树结点值

$$AL < A < BL < B < BR$$

2) RR平衡旋转（左单旋转）。由于在结点A的右孩子（R）的右子树（R）上插入了新结点，A的平衡因子由-1减至-2，导致以A为根的子树失去平衡，需要一次向左的旋转操作。将A的右孩子B向左旋转代替A成为根结点，将A结点向左下旋转成为B的左子树的根结点，而B的原左子树则作为A结点的右子树。

王道考研/CSKAOYAN.COM

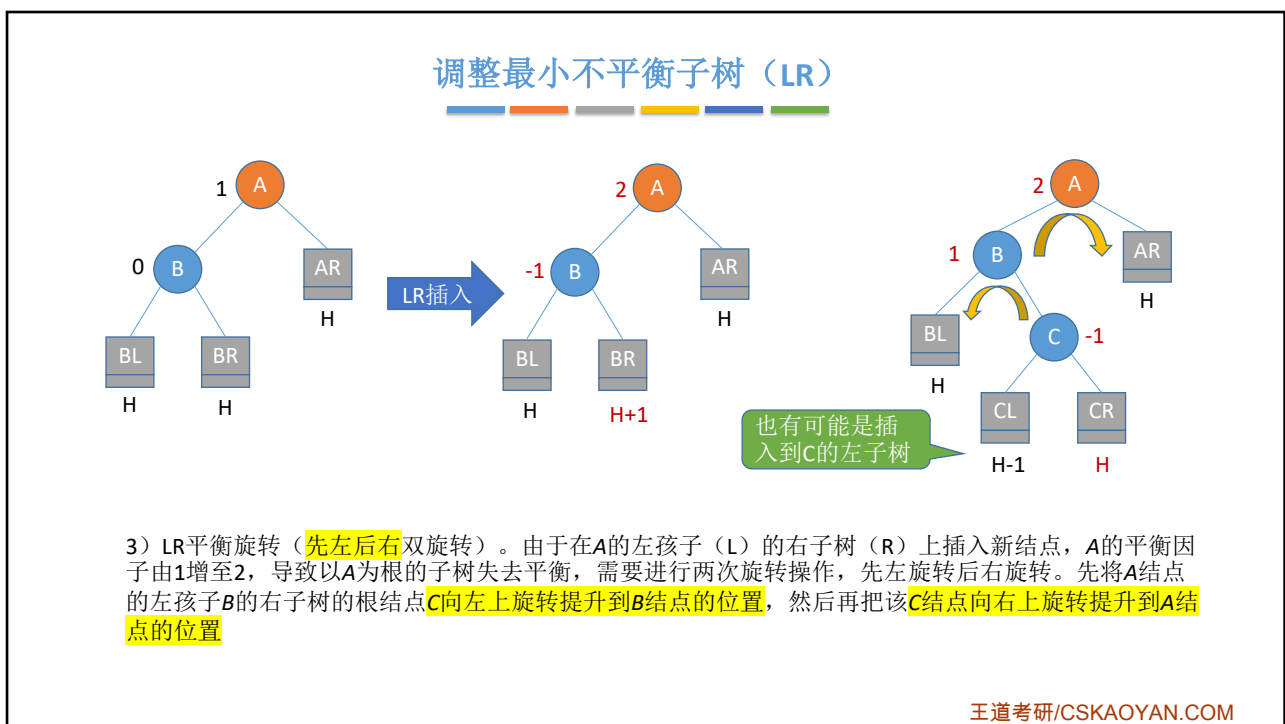
8



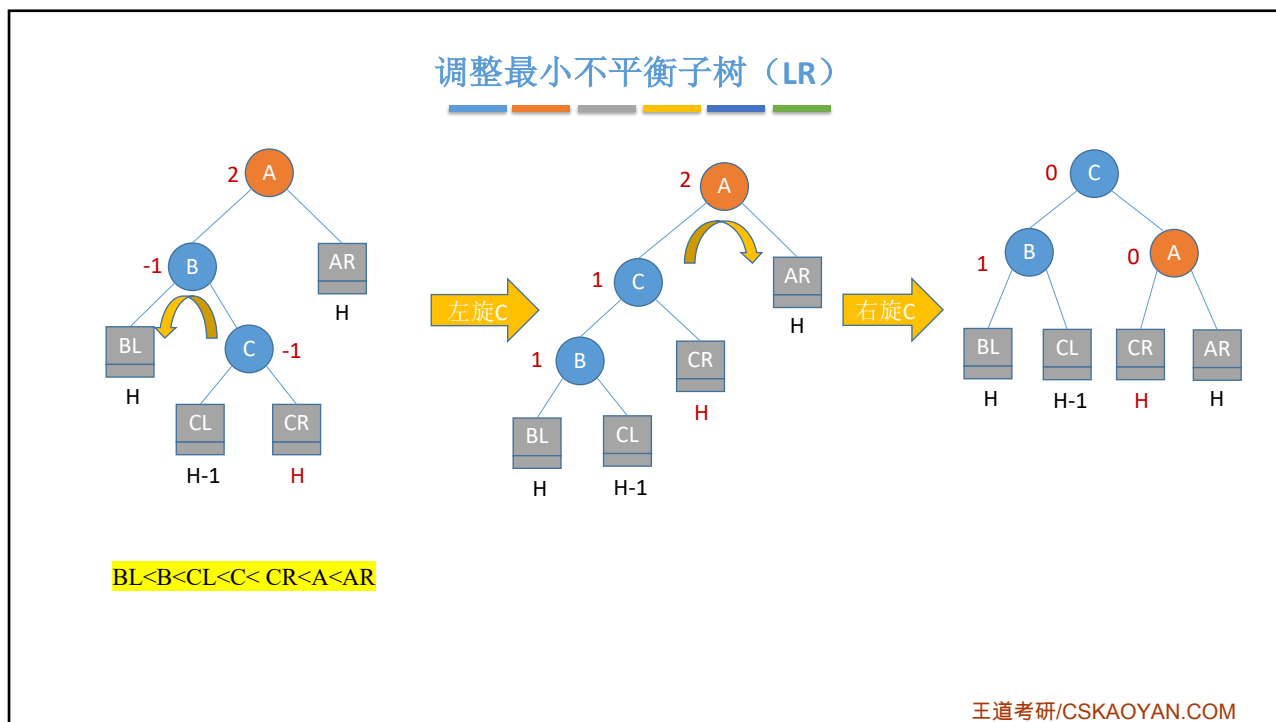
9

公众号： 考研发条

一手课程！



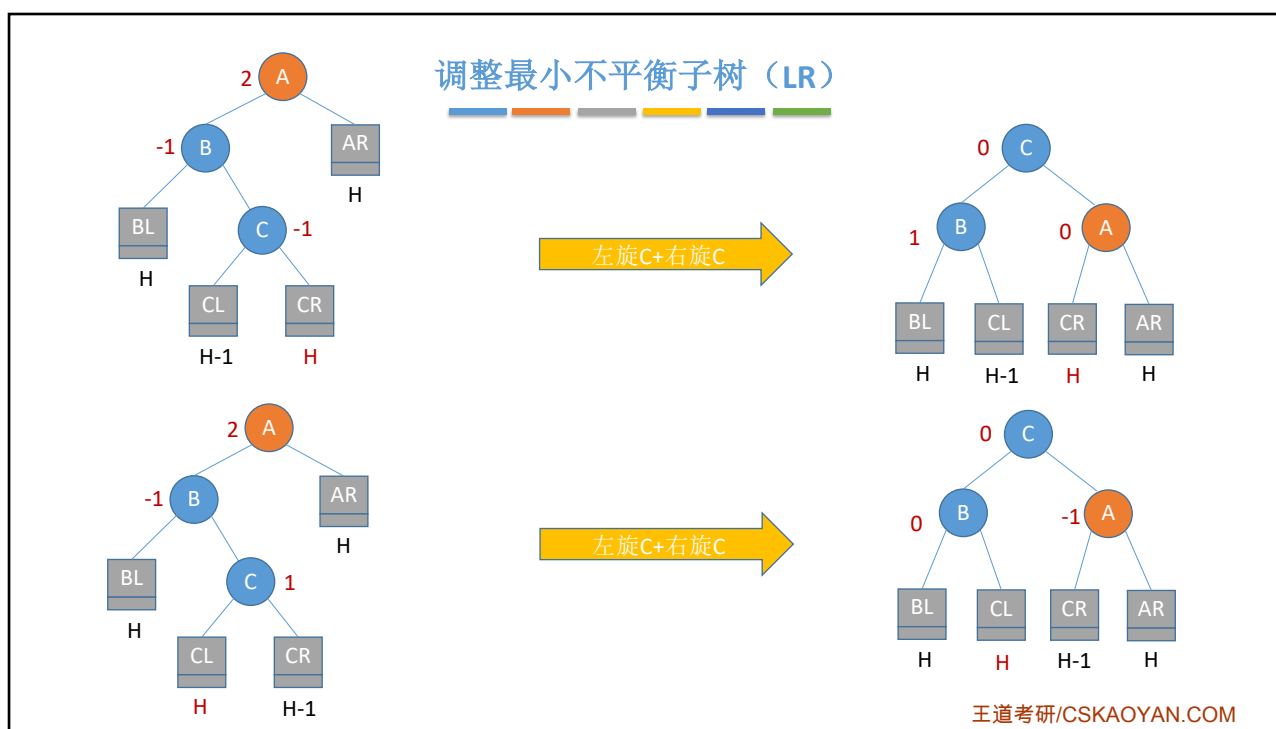
10



11

公众号：考研发条

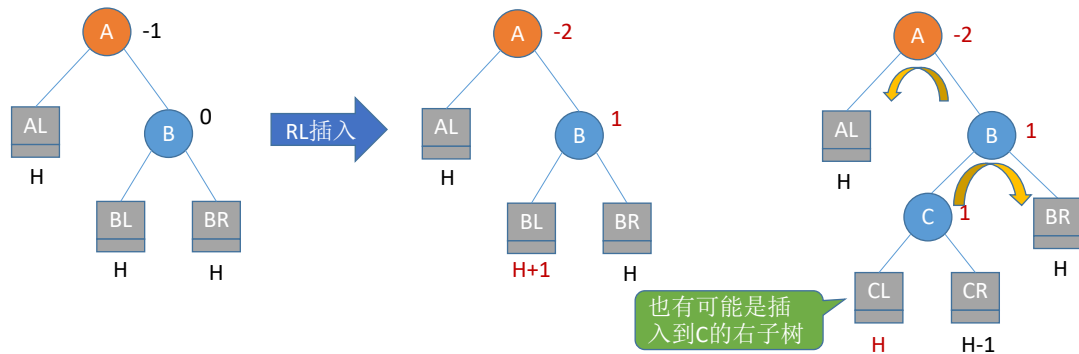
一手课程！



12

调整最小不平衡子树 (RL)

AL < A < CL < C < CR < B < BR



4) RL平衡旋转（先右后左双旋转）。由于在A的右孩子（R）的左子树（L）上插入新结点，A的平衡因子由-1减至-2，导致以A为根的子树失去平衡，需要进行两次旋转操作，先右旋转后左旋转。先将A结点的右孩子B的左子树的根结点C向右上旋转提升到B结点的位置，然后再把该C结点向左上旋转提升到A结点的位置

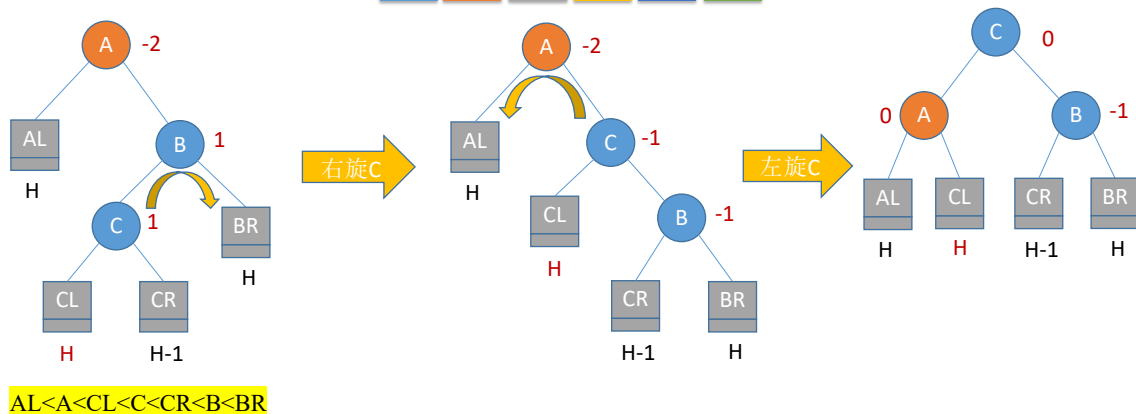
王道考研/CSKAOYAN.COM

13

公众号： 考研发条

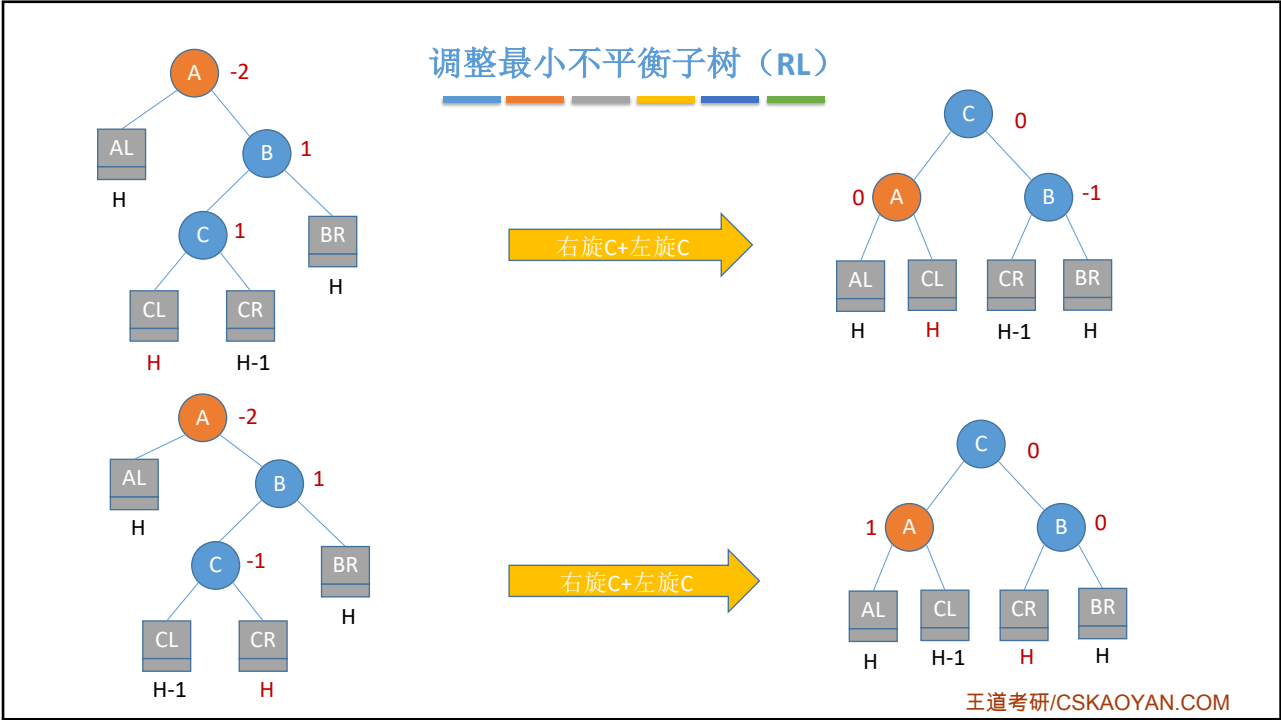
一手课程！

调整最小不平衡子树 (RL)



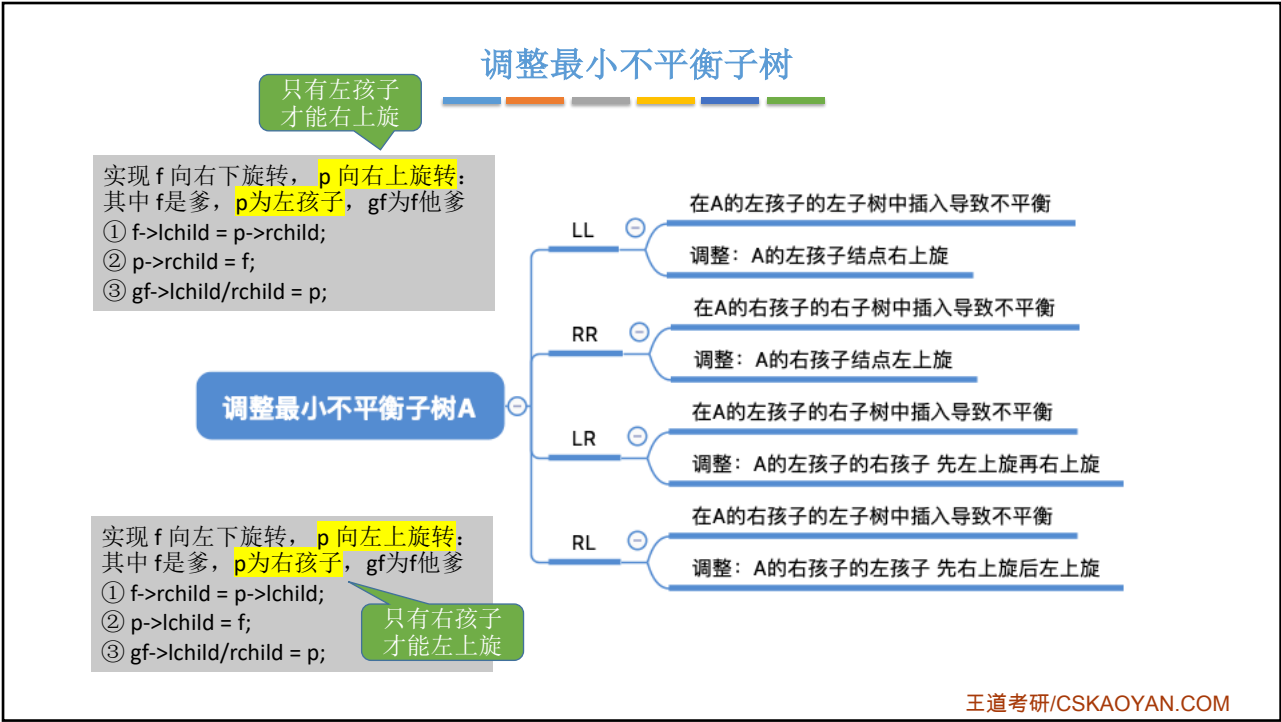
王道考研/CSKAOYAN.COM

14

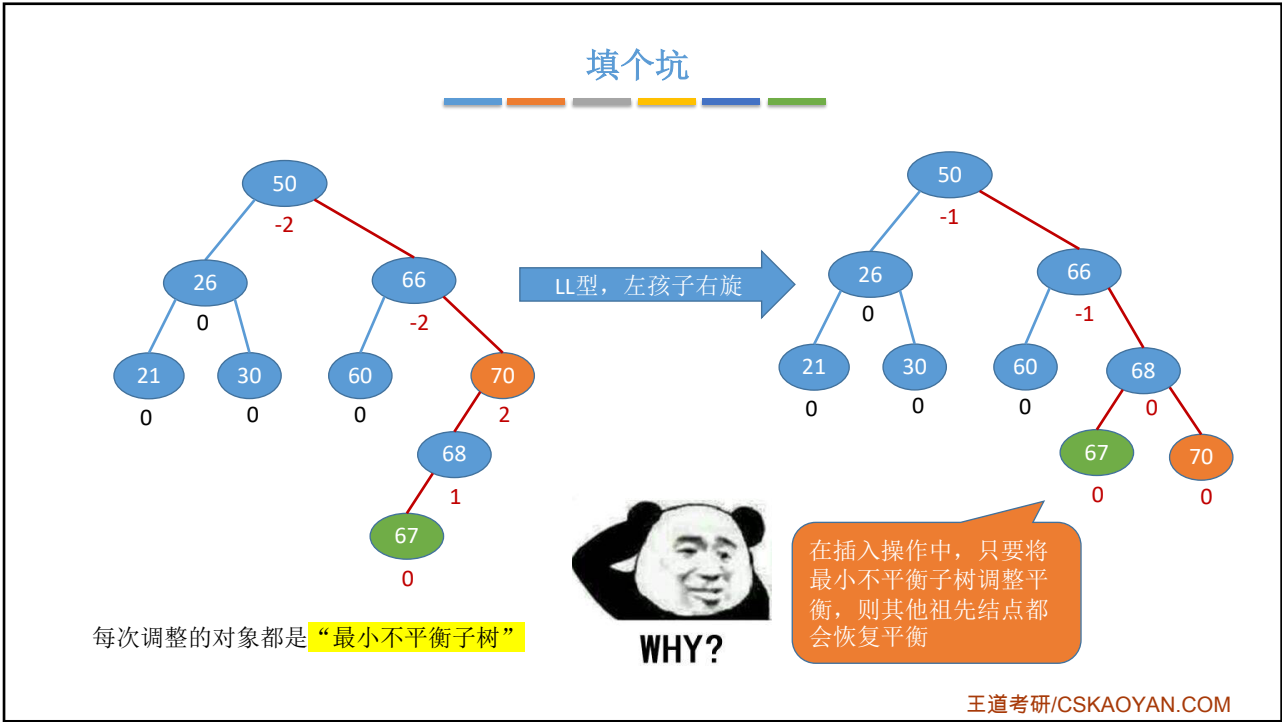


15

公众号： 考研发条 一手课程！

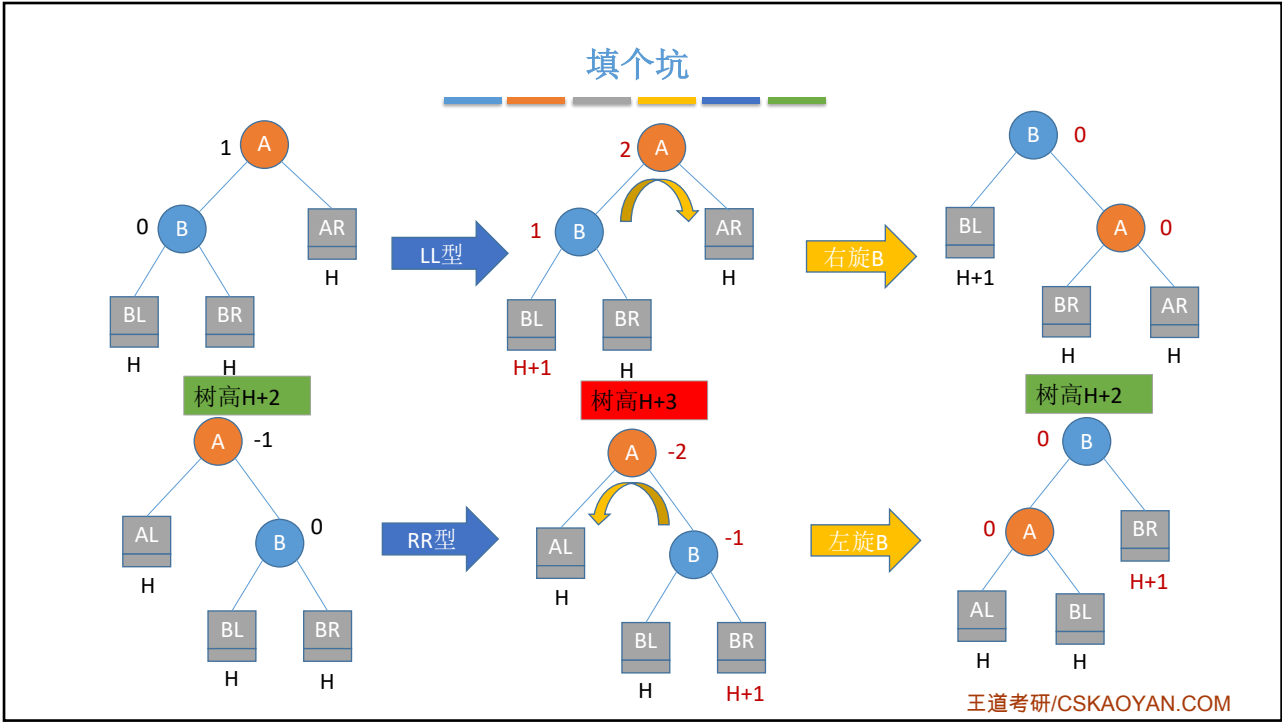


16

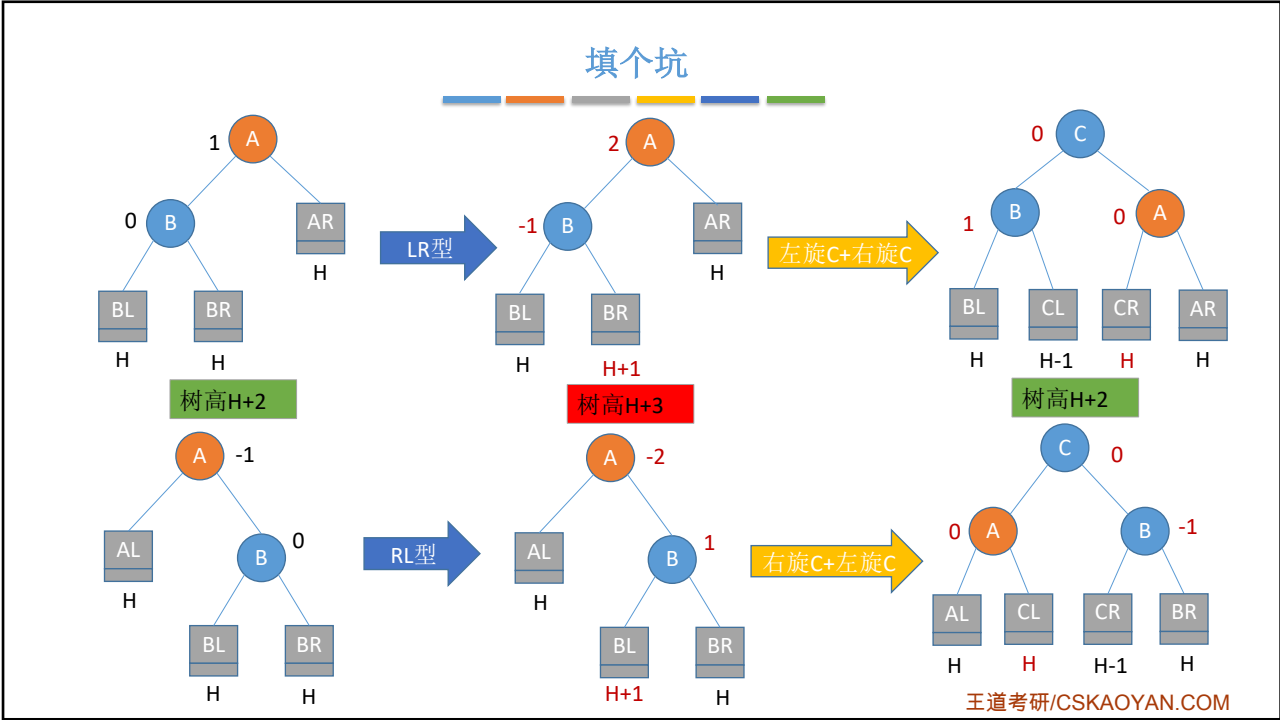


17

公众号： 考研发条 一手课程！

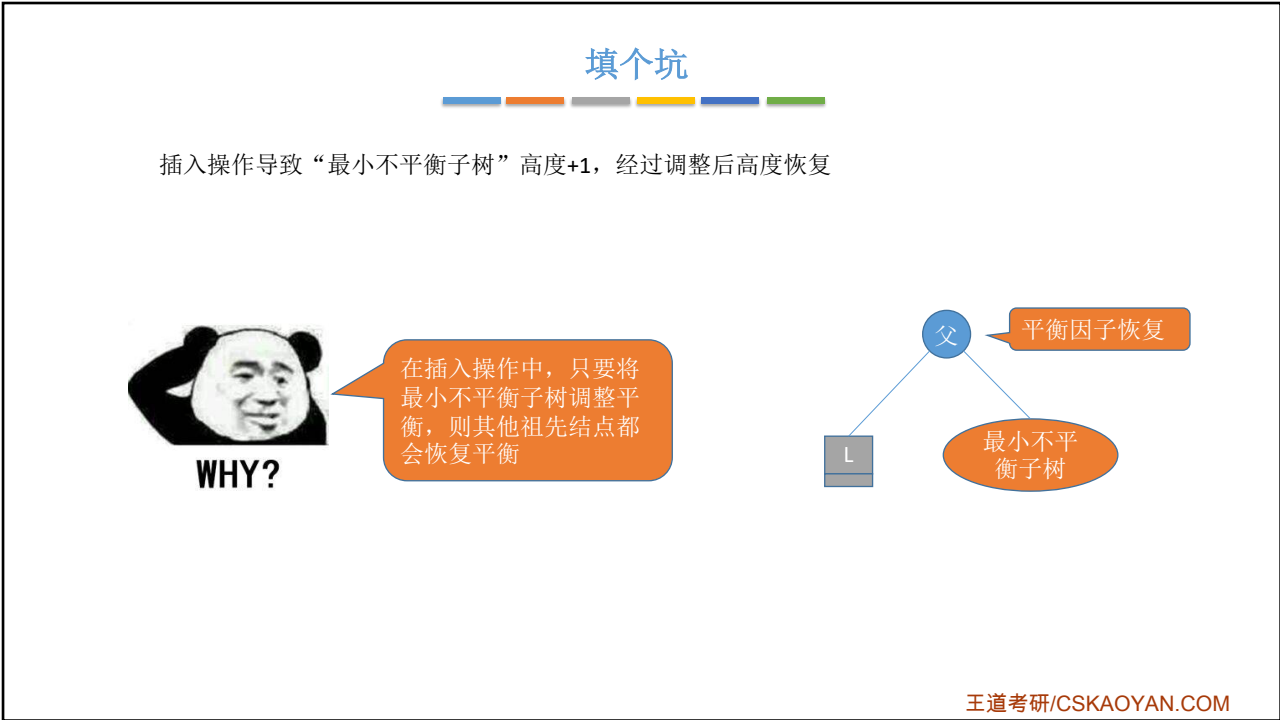


18

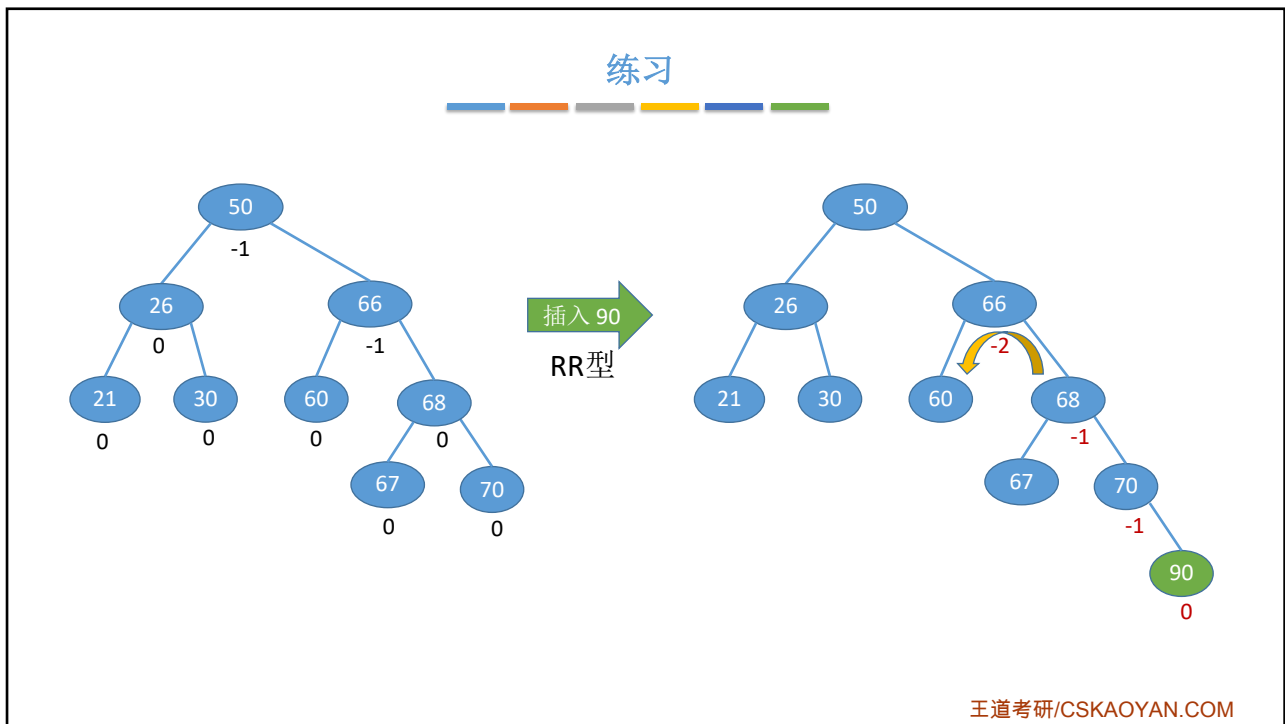


19

公众号： 考研发条 一手课程！



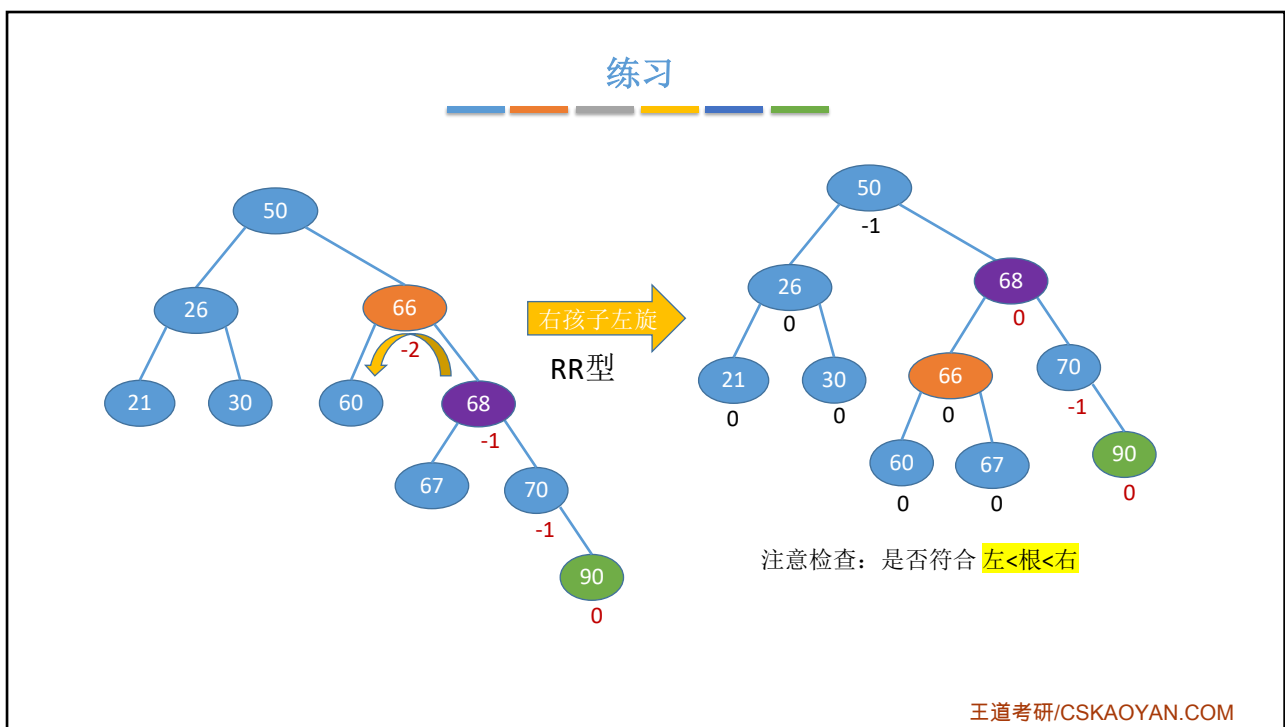
20



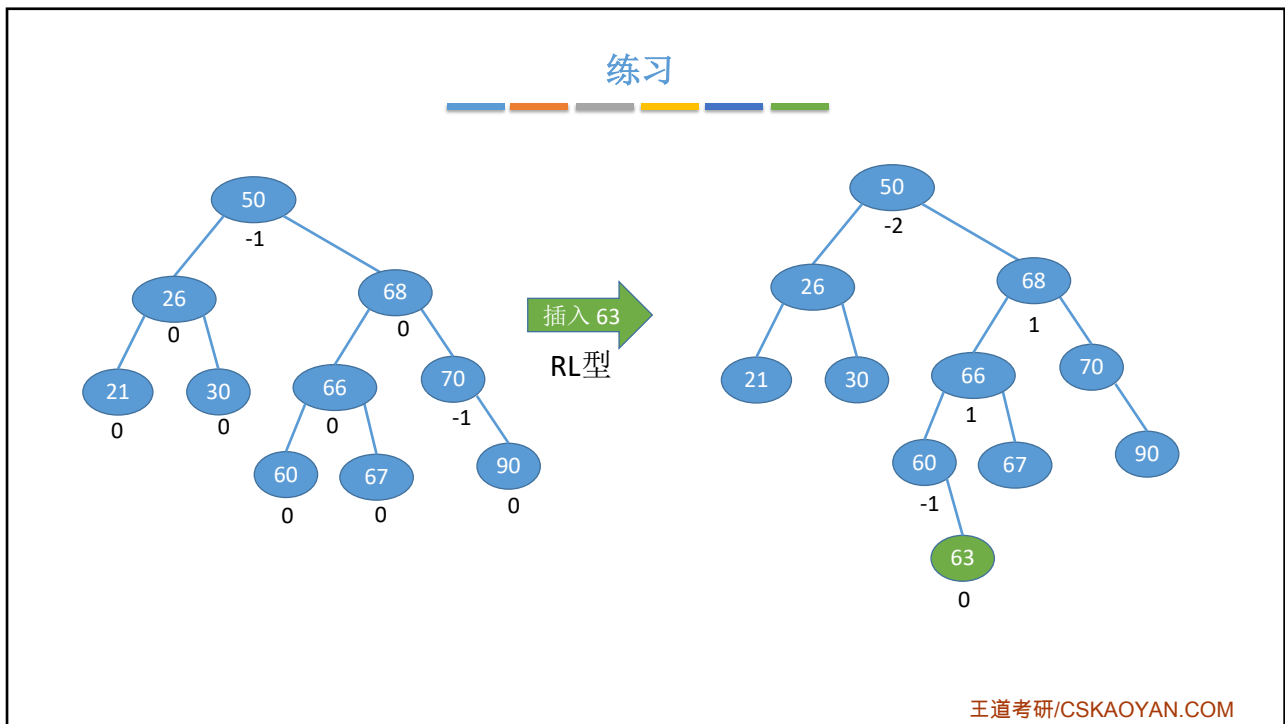
21

公众号： 考研发条

一手课程！



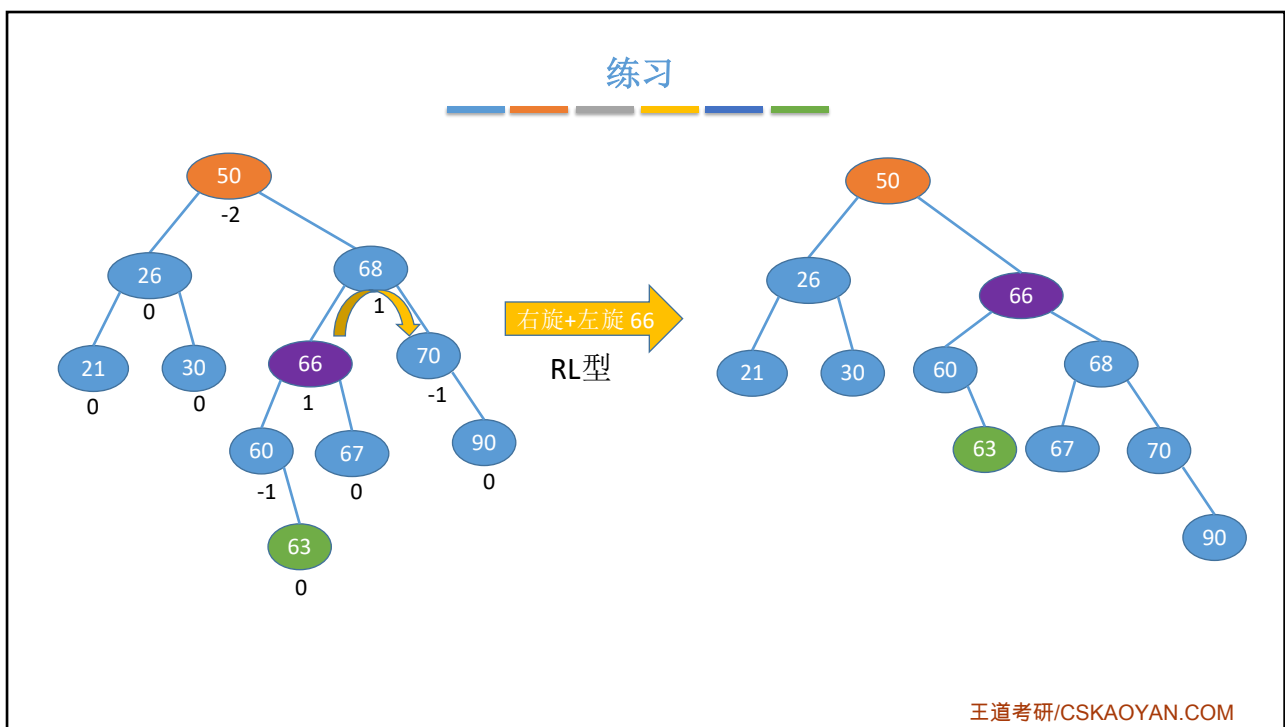
22



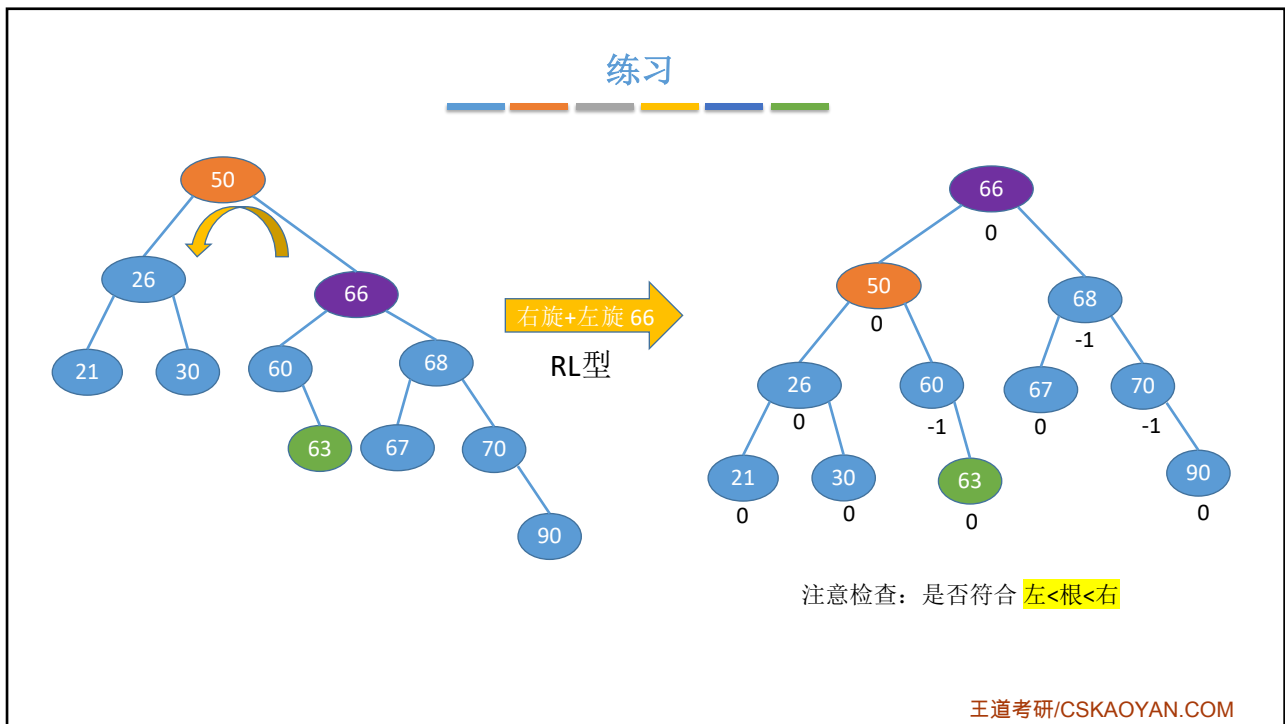
23

公众号： 考研发条

一手课程！



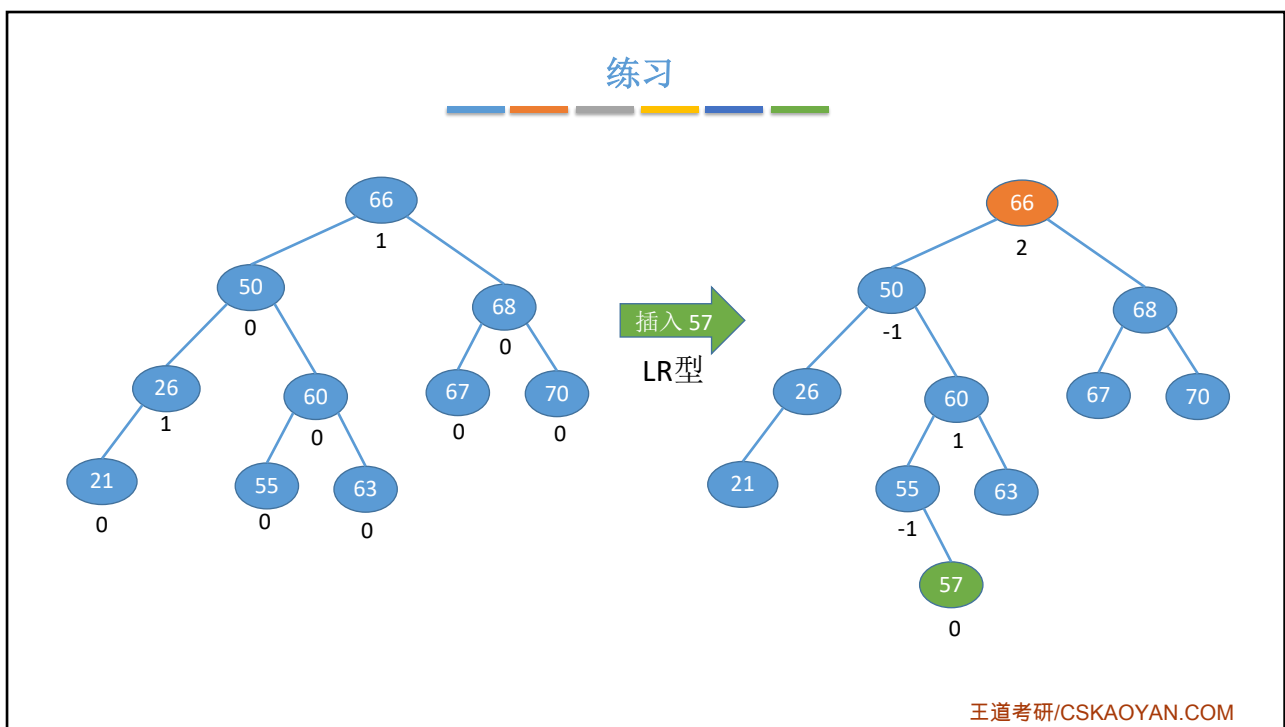
24



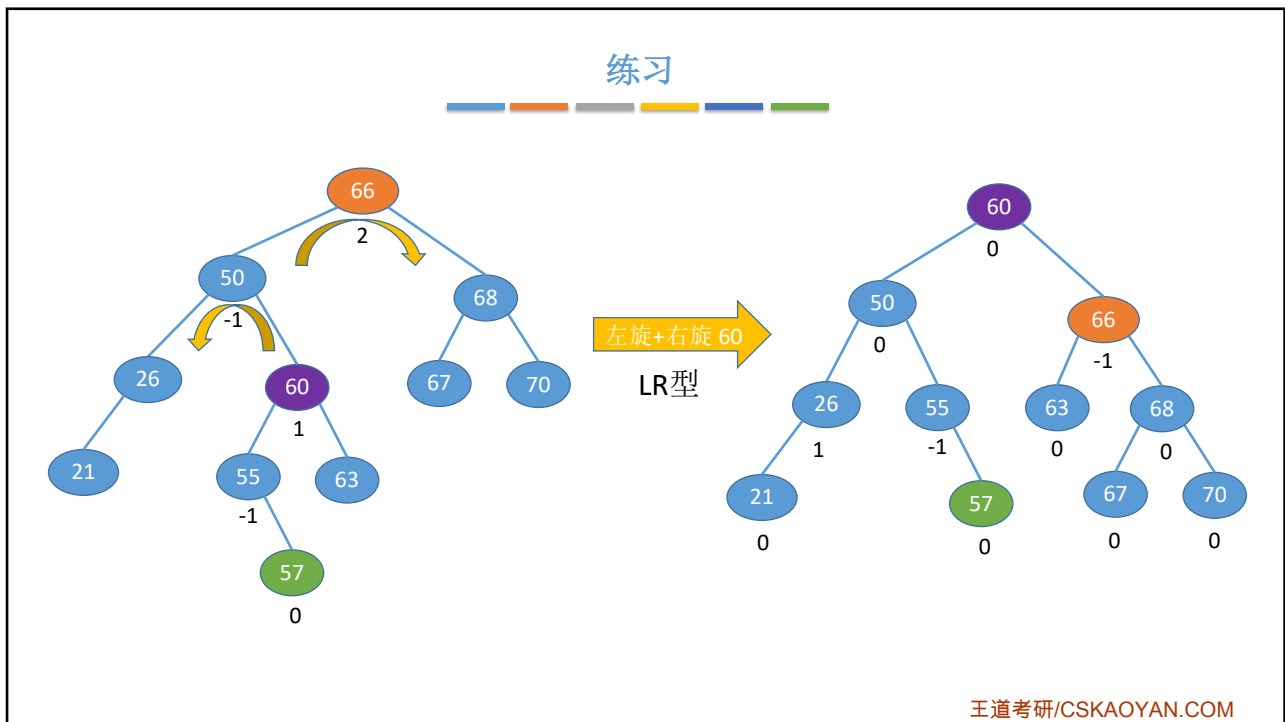
25

公众号： 考研发条

一手课程！



26



27

公众号： 考研发条

一手课程！

查找效率分析

若树高为 h ，则最坏情况下，查找一个关键字最多需要对比 h 次，即查找操作的时间复杂度不可能超过 $O(h)$

平衡二叉树——树上任一结点的左子树和右子树的高度之差不超过1。

假设以 n_h 表示深度为 h 的平衡树中含有的最少结点数。

则有 $n_0 = 0, n_1 = 1, n_2 = 2$ ，并且有 $n_h = n_{h-1} + n_{h-2} + 1$

可以证明含有 n 个结点的平衡二叉树的最大深度为 $O(\log_2 n)$ ，平衡二叉树的平均查找长度为 $O(\log_2 n)$

王道考研/CSKAOYAN.COM

28

查找效率分析

《An algorithm for the organization of information》——G.M. Adelson-Velsky 和 E.M. Landis ,1962



Figure 1



Figure 2

The recording algorithm is such that at each moment, the reference board is an admissible tree.
Lemma 1. Let the number of cells of the admissible tree be equal to N . Then the maximum length of the branch is not greater than $(3/2) \log_2 (N + 1)$.

Proof. Let us denote by N_n the minimum number of cells in the admissible tree when the given maximum length of the branch is n . Then it can be easily proven (see Figure 2) that $N_n = N_{n-1} + N_{n-2} + 1$.

When we solve this equation in finite remainders, we get

$$N_n = \left(1 + \frac{2}{\sqrt{5}}\right) \left(\frac{1 + \sqrt{5}}{2}\right)^n + \left(1 - \frac{2}{\sqrt{5}}\right) \left(\frac{1 - \sqrt{5}}{2}\right)^n - 1.$$

Whence

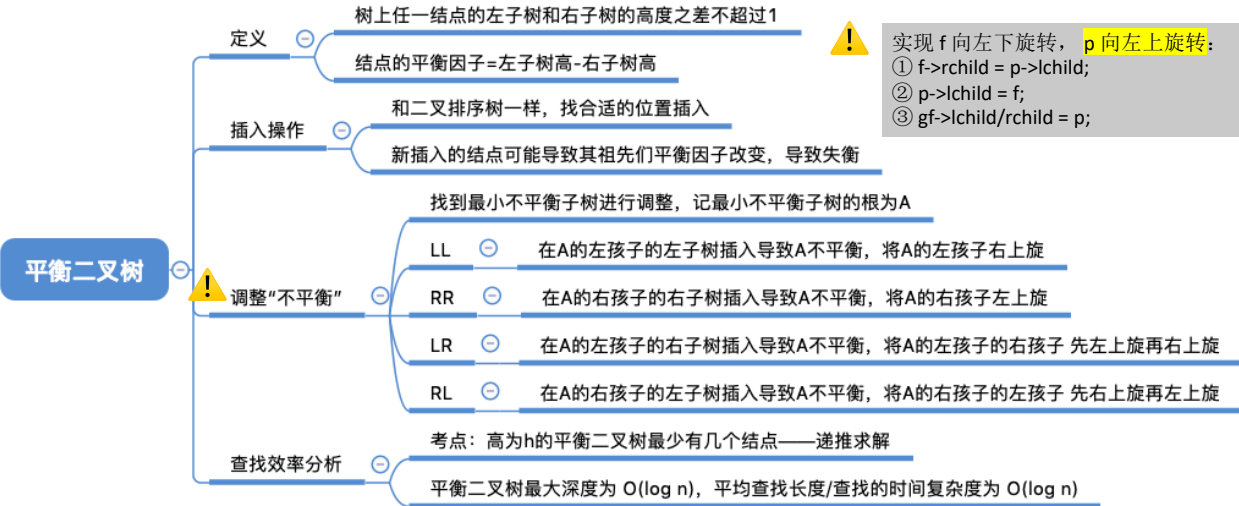
$$n < \log_{\frac{1+\sqrt{5}}{2}} (N + 1) < \frac{3}{2} \log_2 (N + 1),$$

王道考研/CSKAOYAN.COM

29

公众号： 考研发条 一手课程！

知识回顾与重要考点



王道考研/CSKAOYAN.COM

30