算法与数据结构

liuyubobobo

二叉搜索树 Binary Search Tree

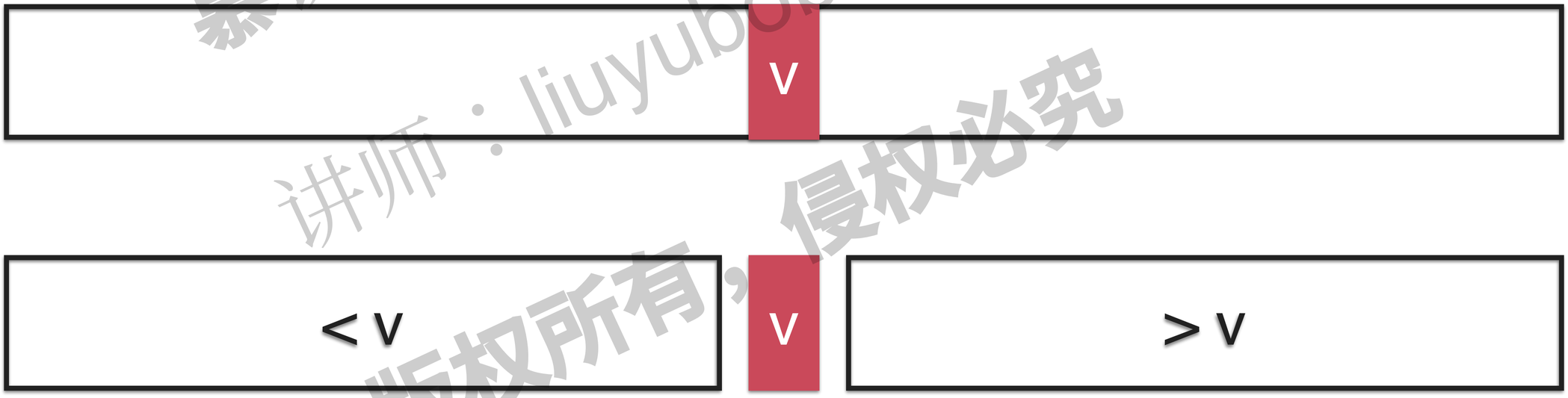
查找问题 Searching Problem

查找问题是计算机中非常重要的基础问题

二分查找法 Binary Search

二分查找法 Binary Search

对于有序数列，才能使用二分查找法 （排序的作用）



二分查找法 Binary Search

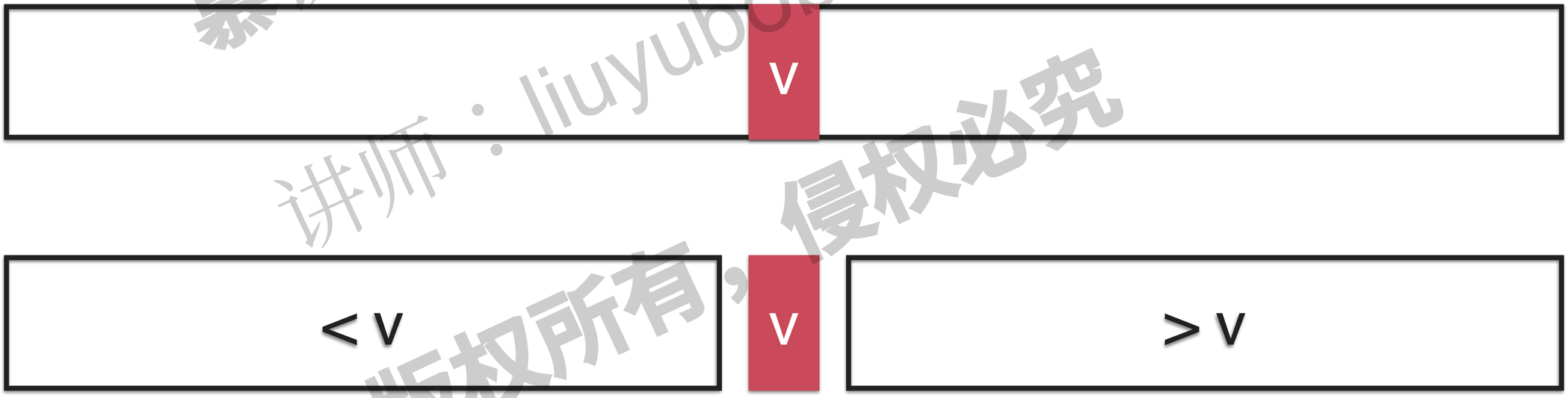
二分查找法的思想在1946年提出。

第一个没有bug的二分查找法在1962年才出现。

操作：实现二分查找法

二分查找法 Binary Search

对于有序数列，才能使用二分查找法 （排序的作用）



使用递归地方式实现二分查找法

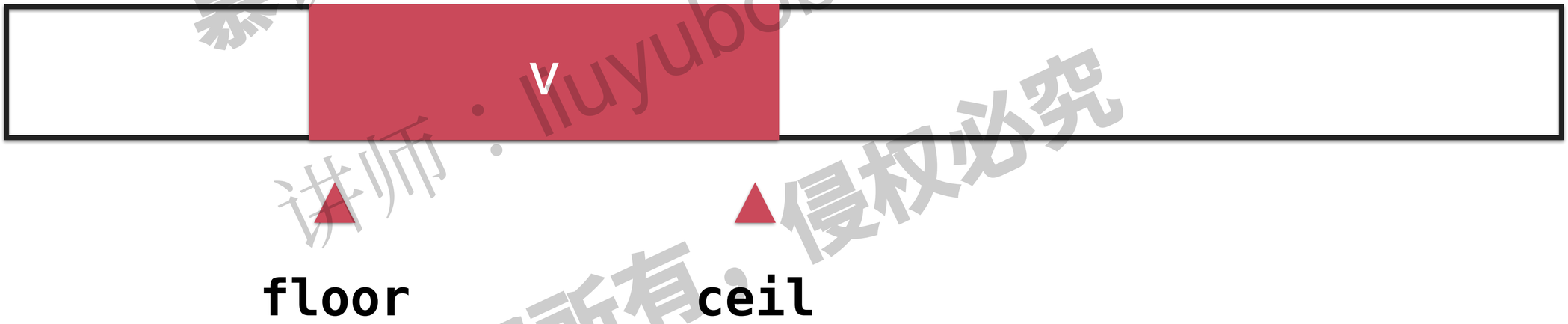
递归实现通常思维起来更容易。

递归在性能上会略差。

练习：实现二分查找法的递归实现

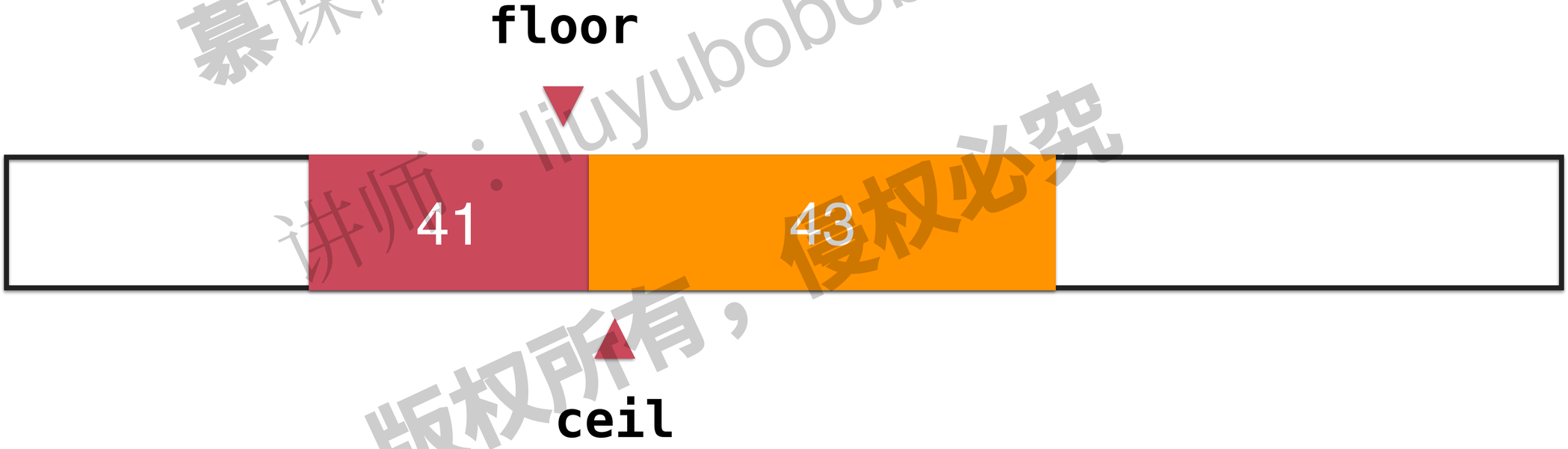
二分查找法的变种

floor 和 ceil



floor 和 ceil

查找42



练习：实现floor和ceil

二分搜索树 Binary Search Tree

二分搜索树的优势

查找表的实现 - 字典数据结构

key1	value1
key2	value2
key3	value3
key4	value4
key5	value5
key6	value6
key7	value7
key8	value8
key9	value9

二分搜索树的优势

	查找元素	插入元素	删除元素
普通数组	O(n)	O(n)	O(n)
顺序数组	O(logn)	O(n)	O(n)
二分搜索树	O(logn)	O(logn)	O(logn)

二分搜索树的优势

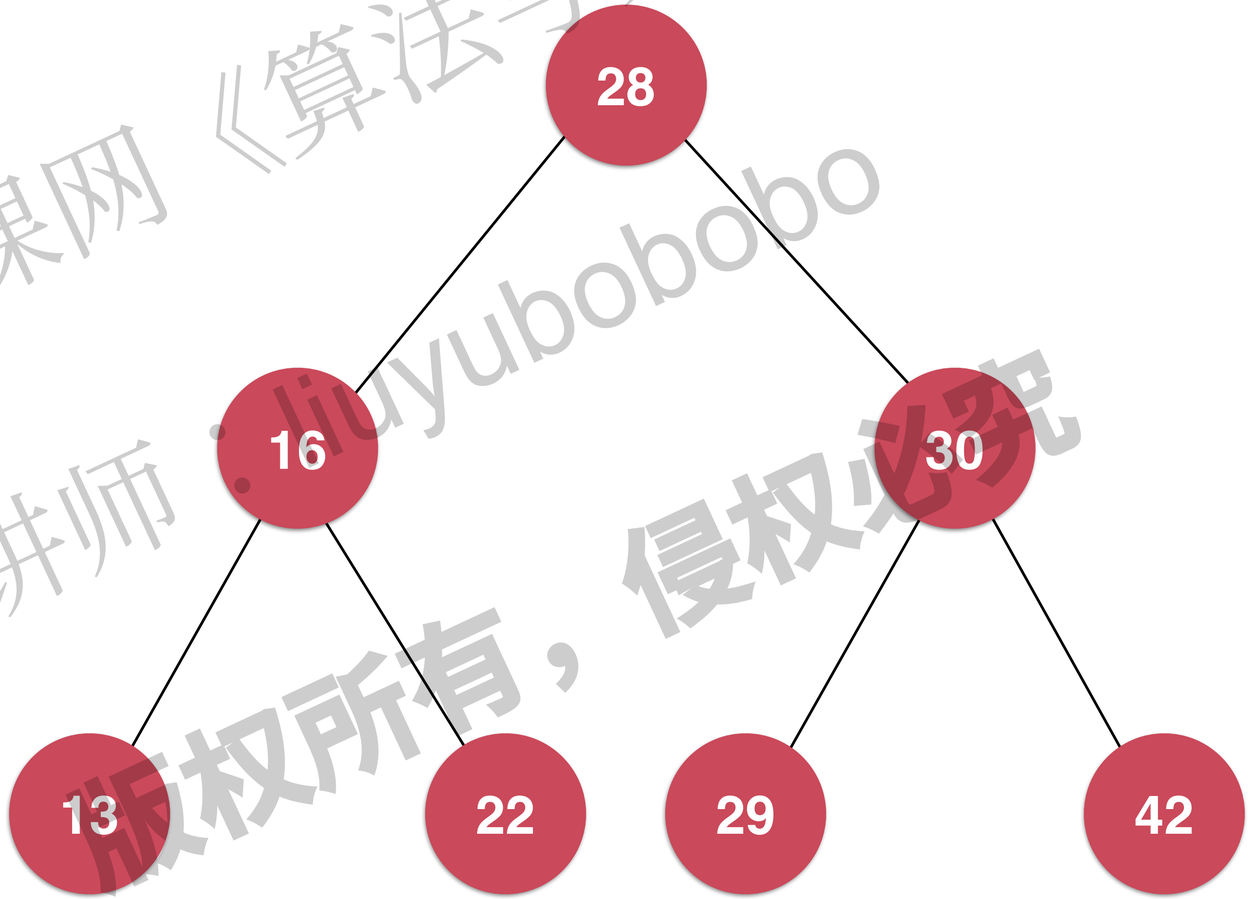
高效

不仅可查找数据；还可以高效地插入，删除数据 - 动态维护数据

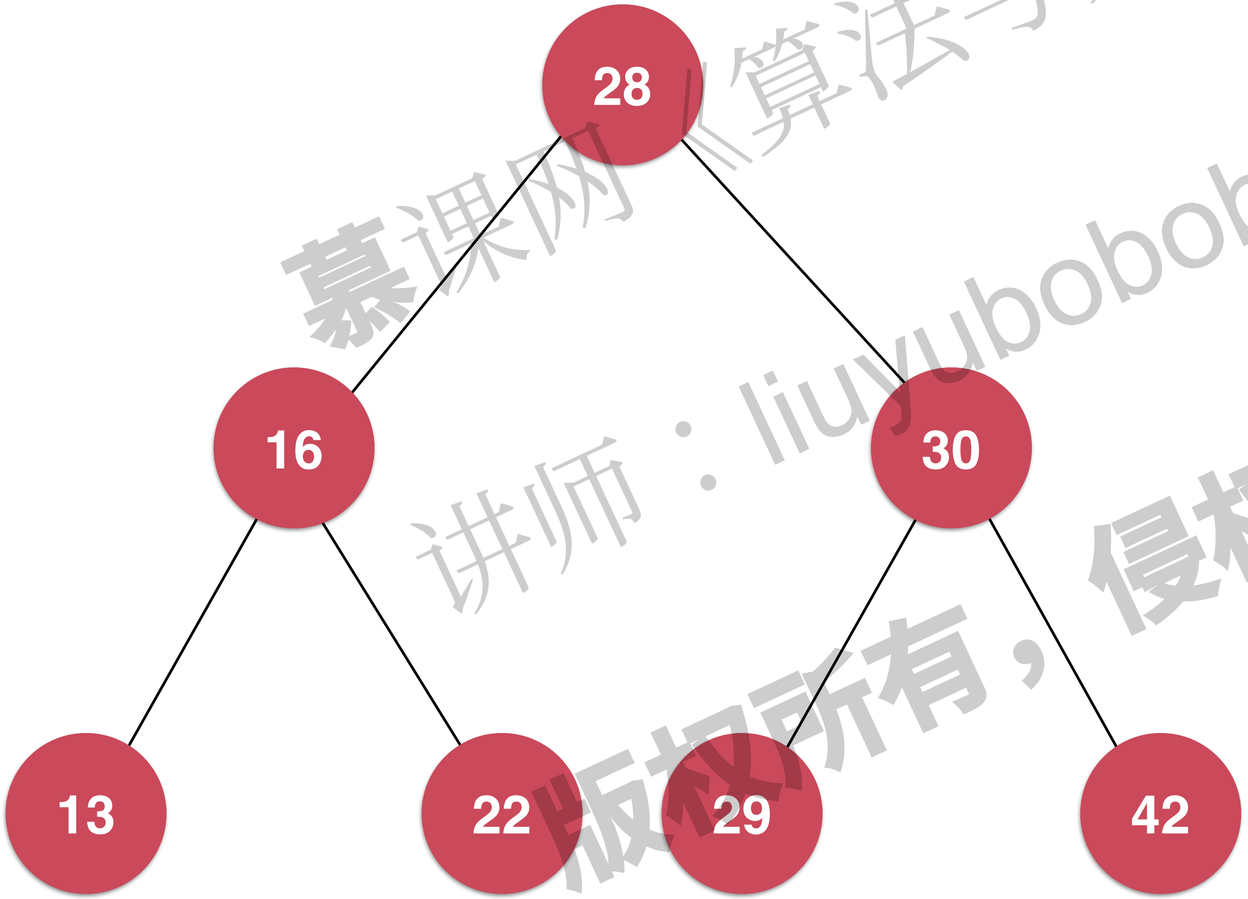
可以方便地回答很多数据之间的关系问题：

- min, max, floor, ceil, rank, select

二分搜索树 Binary Search Tree



二分搜索树 Binary Search Tree



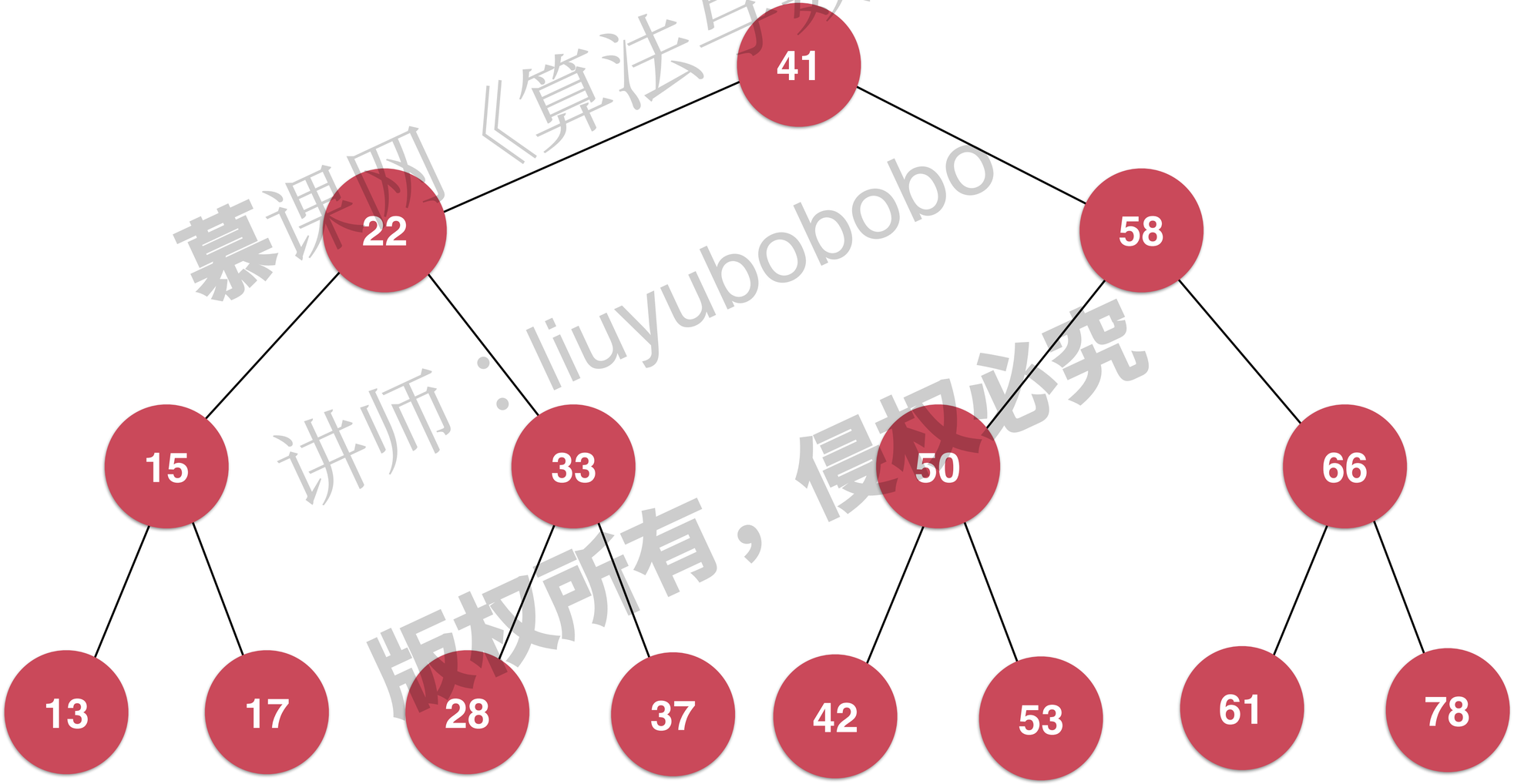
二叉树

每个节点的键值大于左孩子；

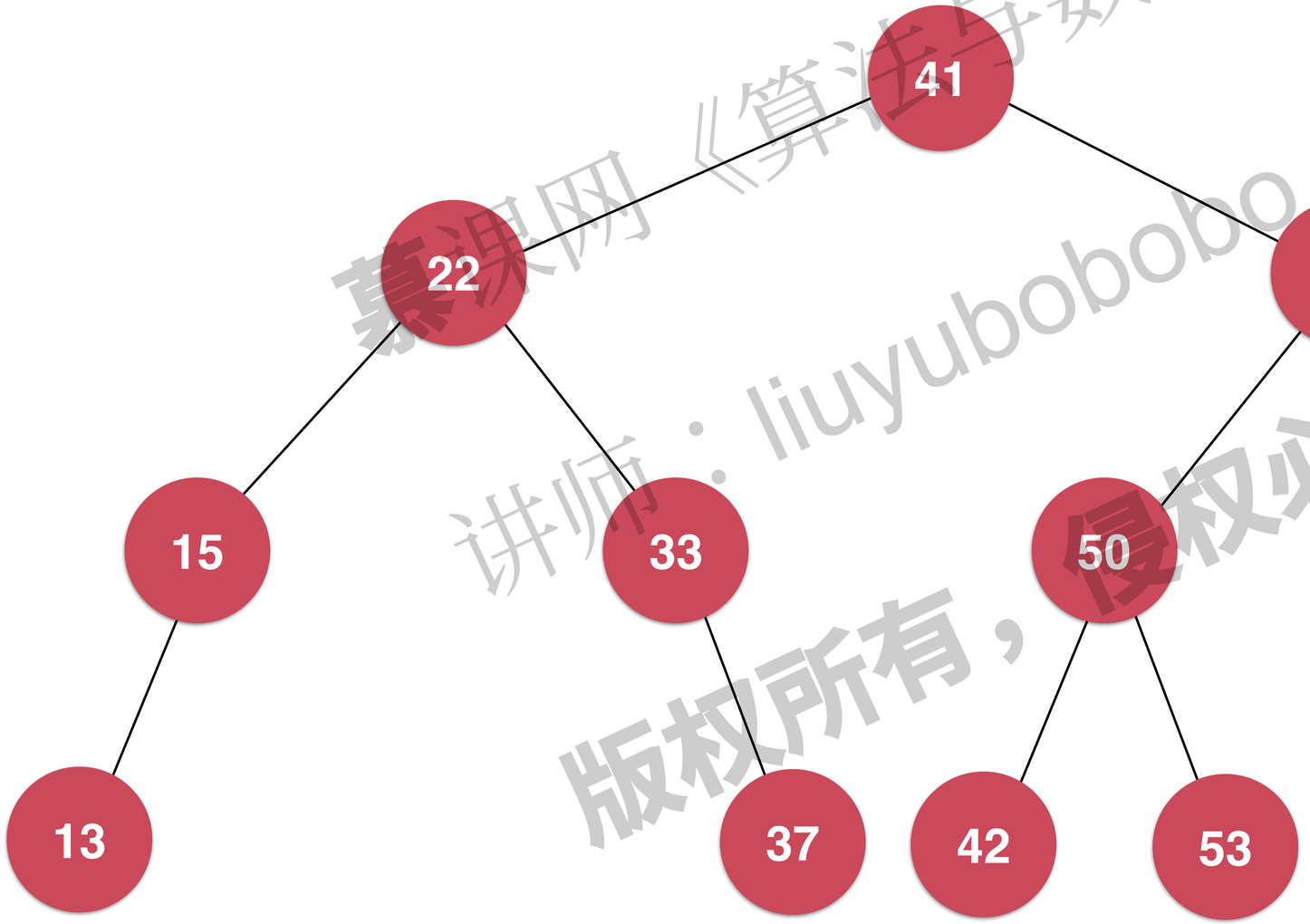
每个节点的键值小于右孩子；

以左右孩子为根的子树仍为二分搜索树

二分搜索树 Binary Search Tree



二分搜索树 Binary Search Tree

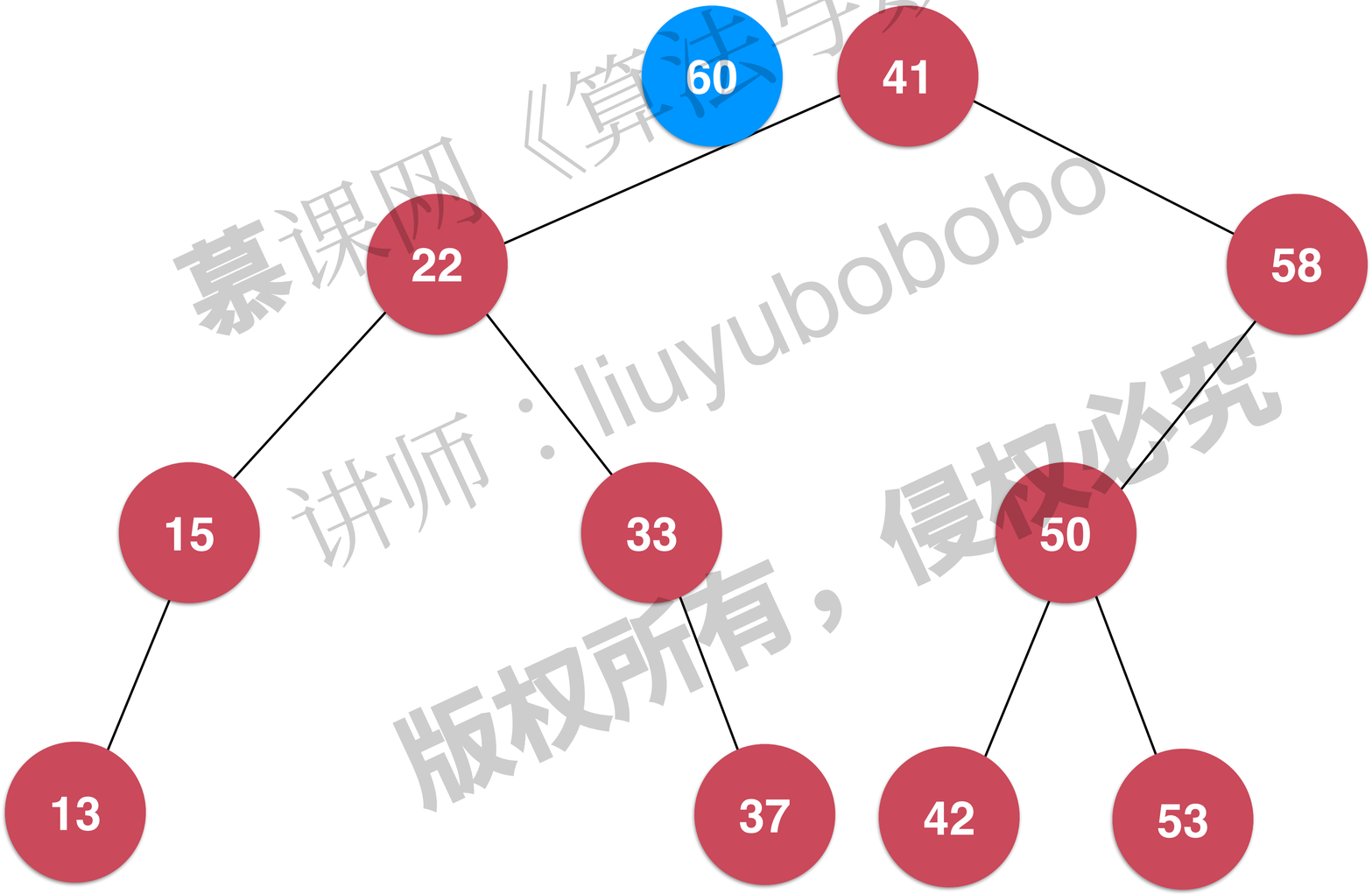


不一定是完全二叉树

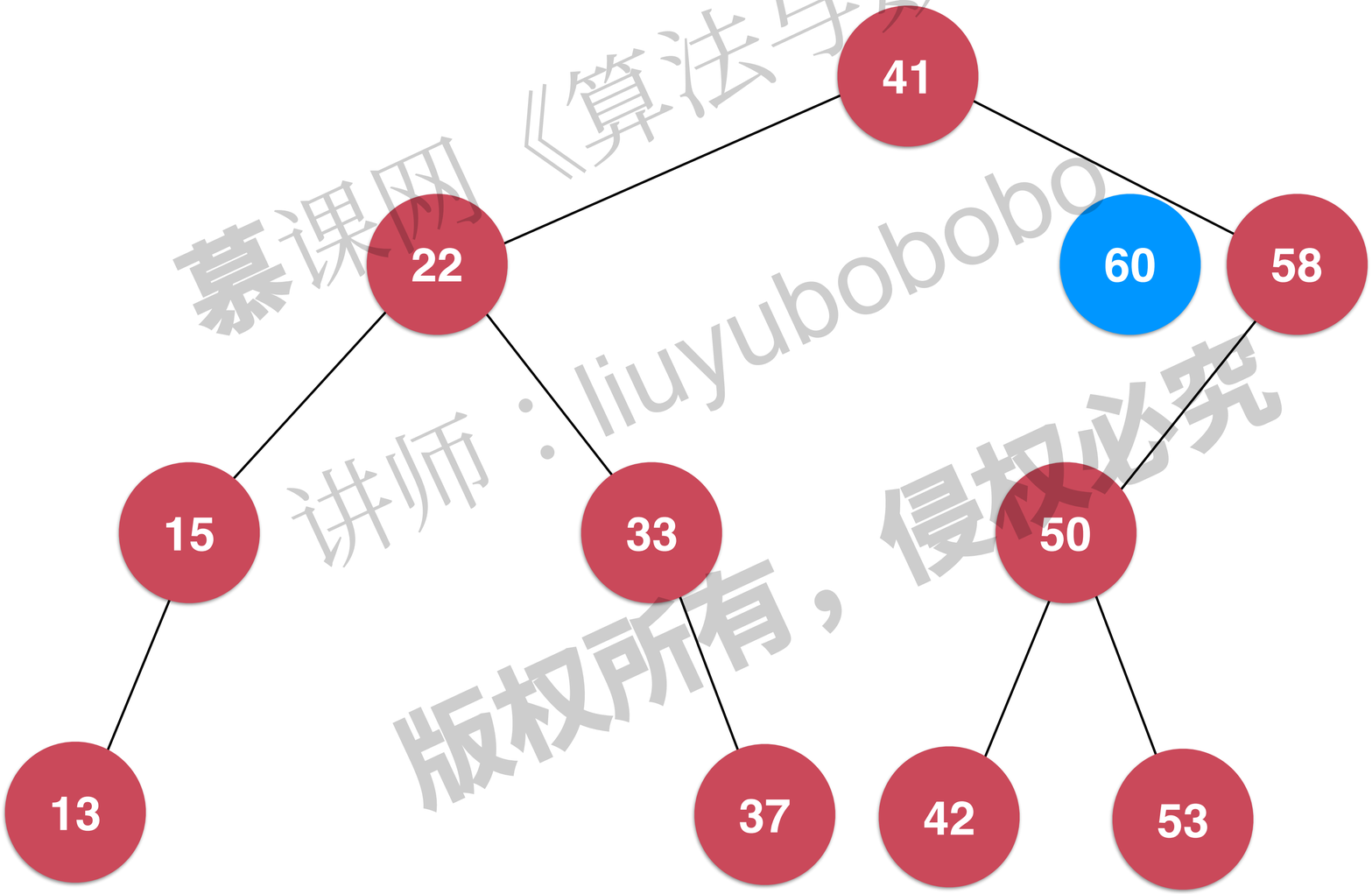
操作：二分搜索树基础结构

插入新的节点

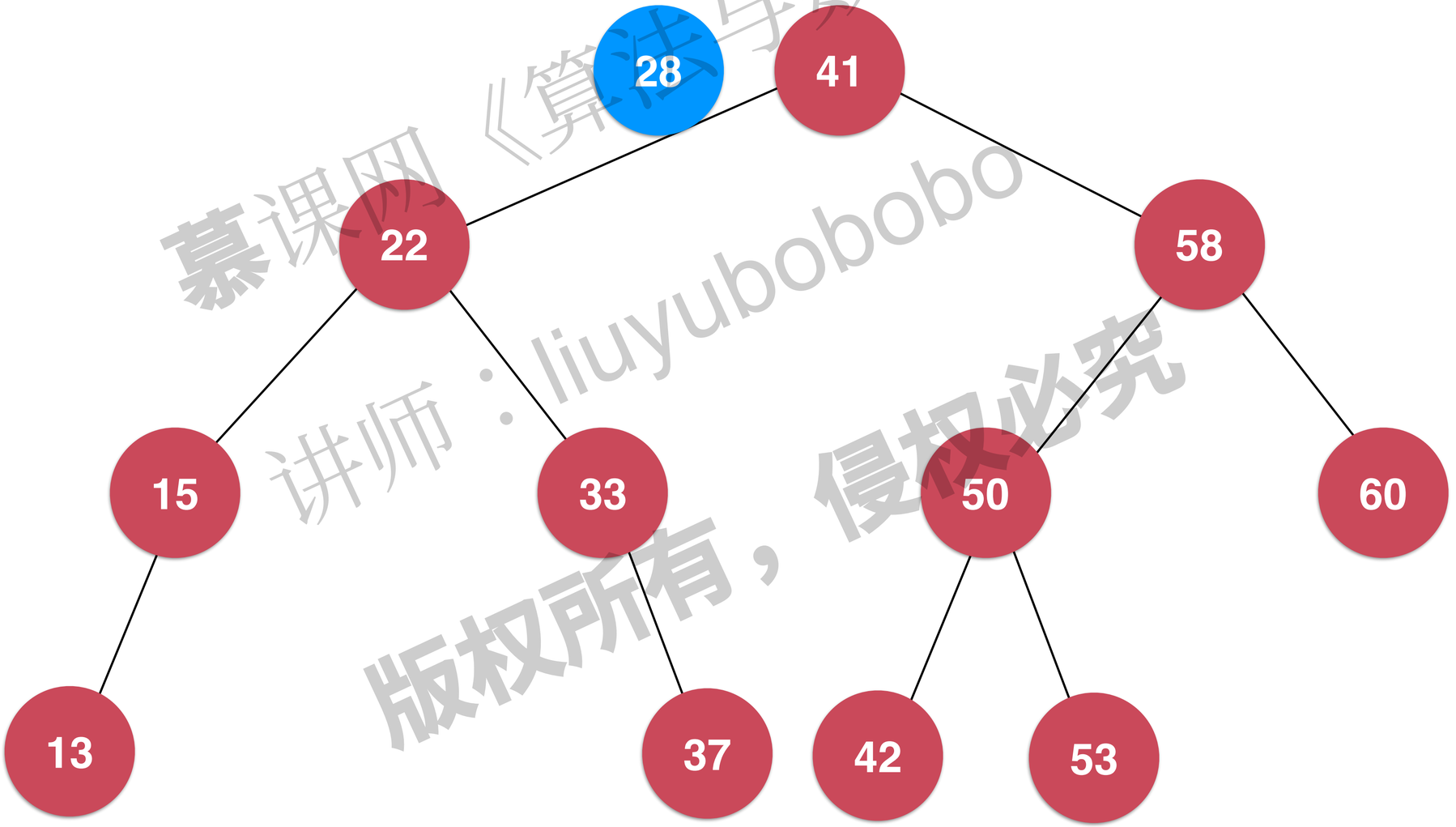
插入新节点 insert



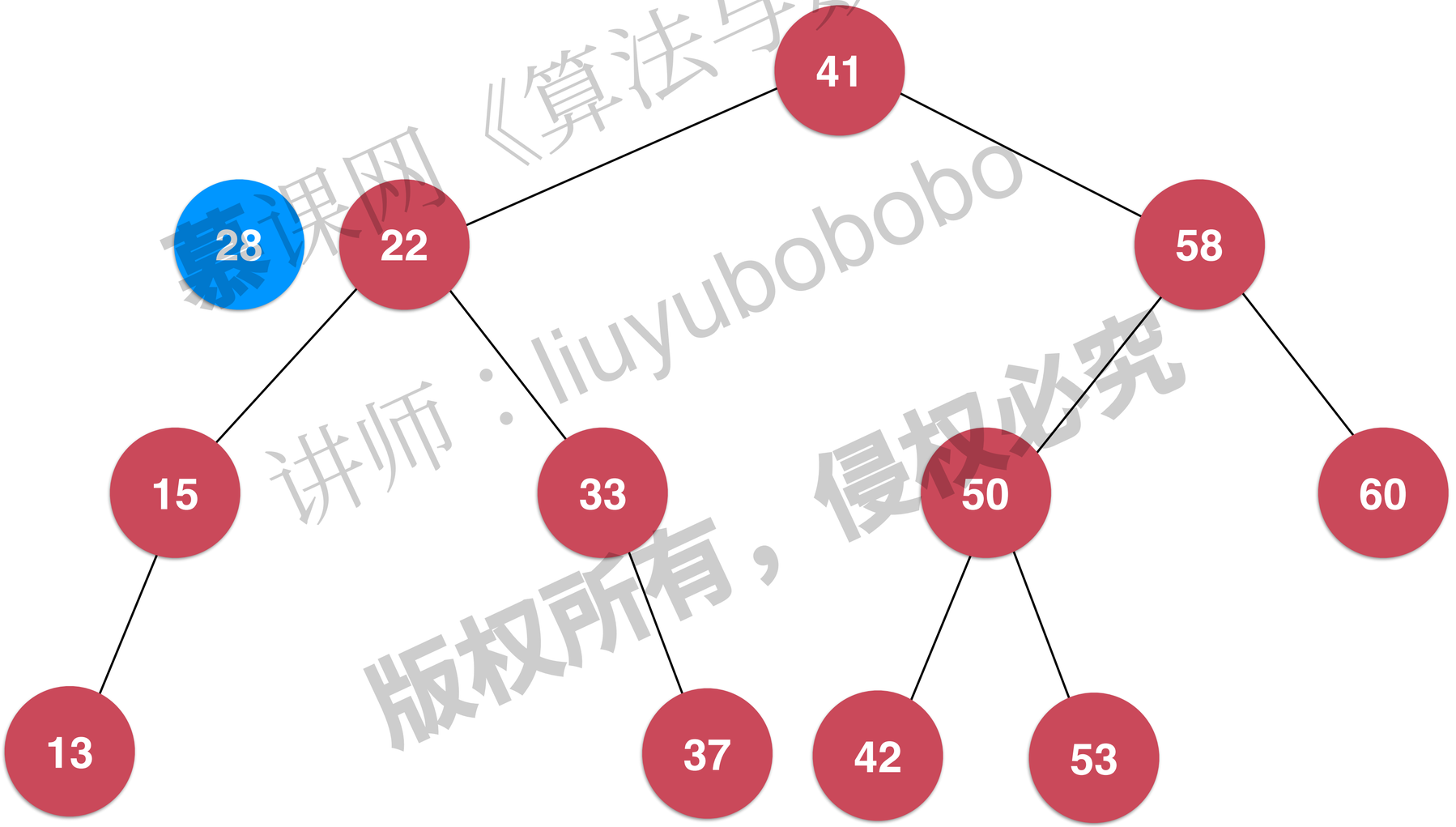
插入新节点 insert



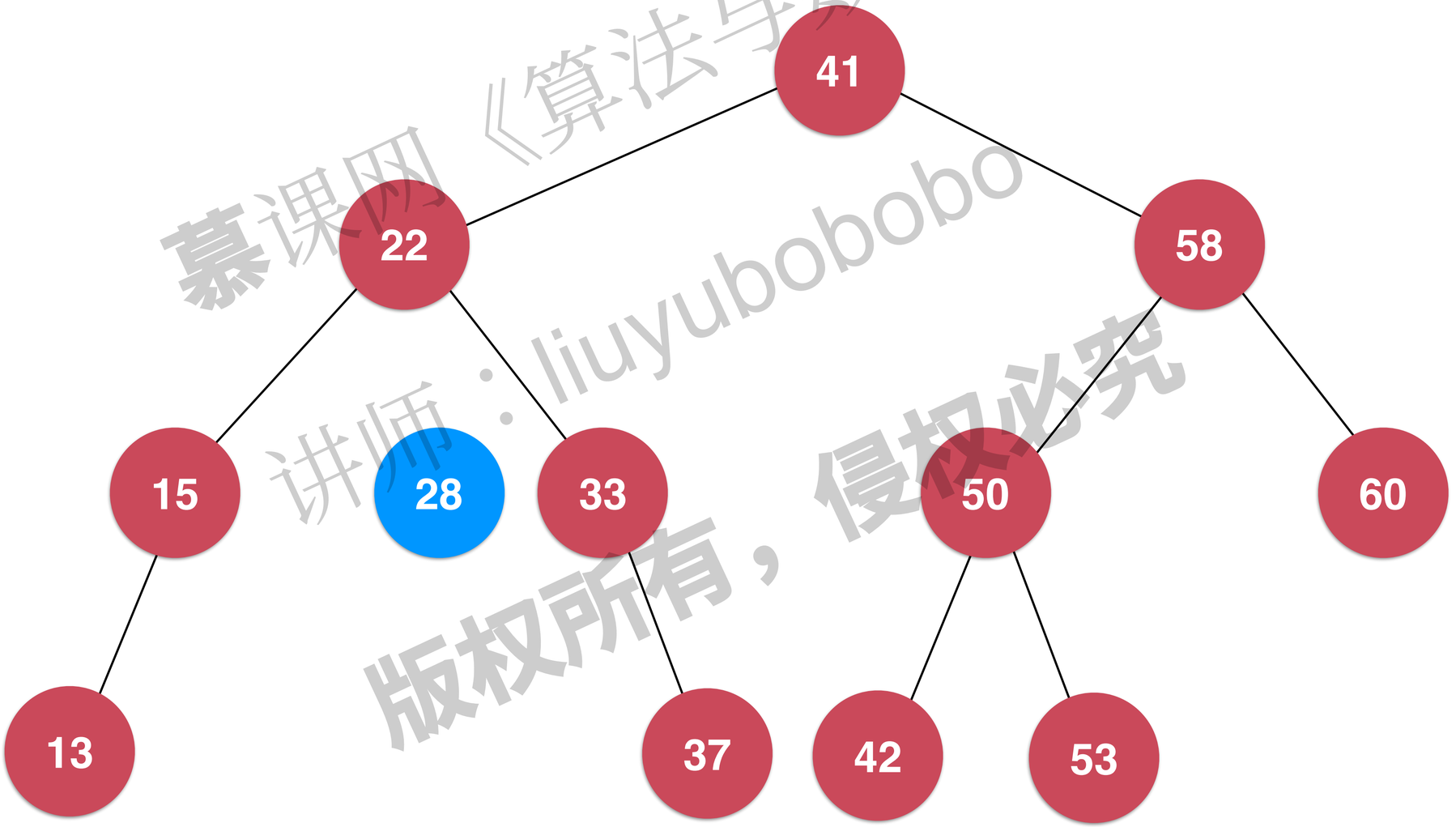
插入新节点 insert



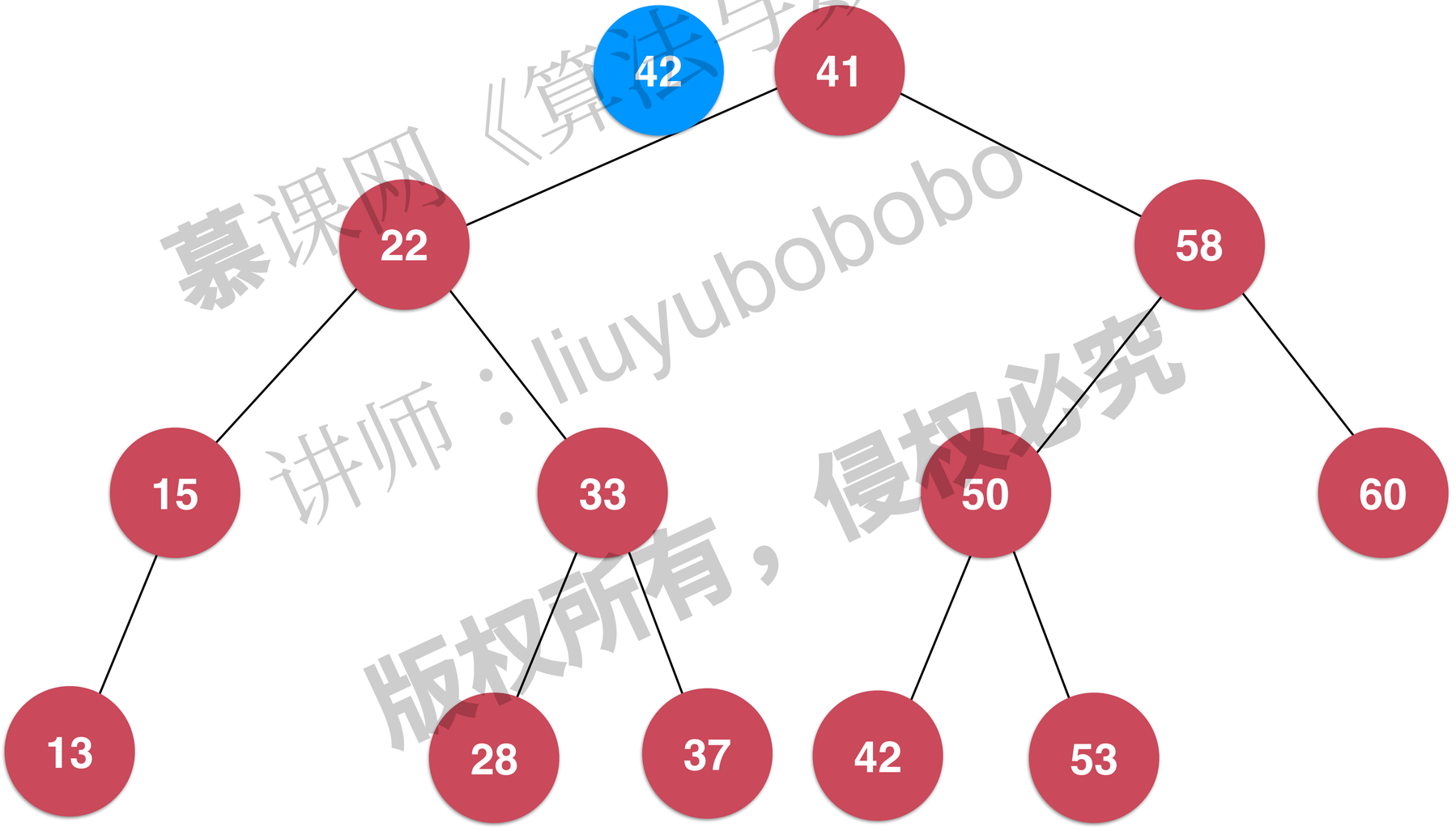
插入新节点 insert



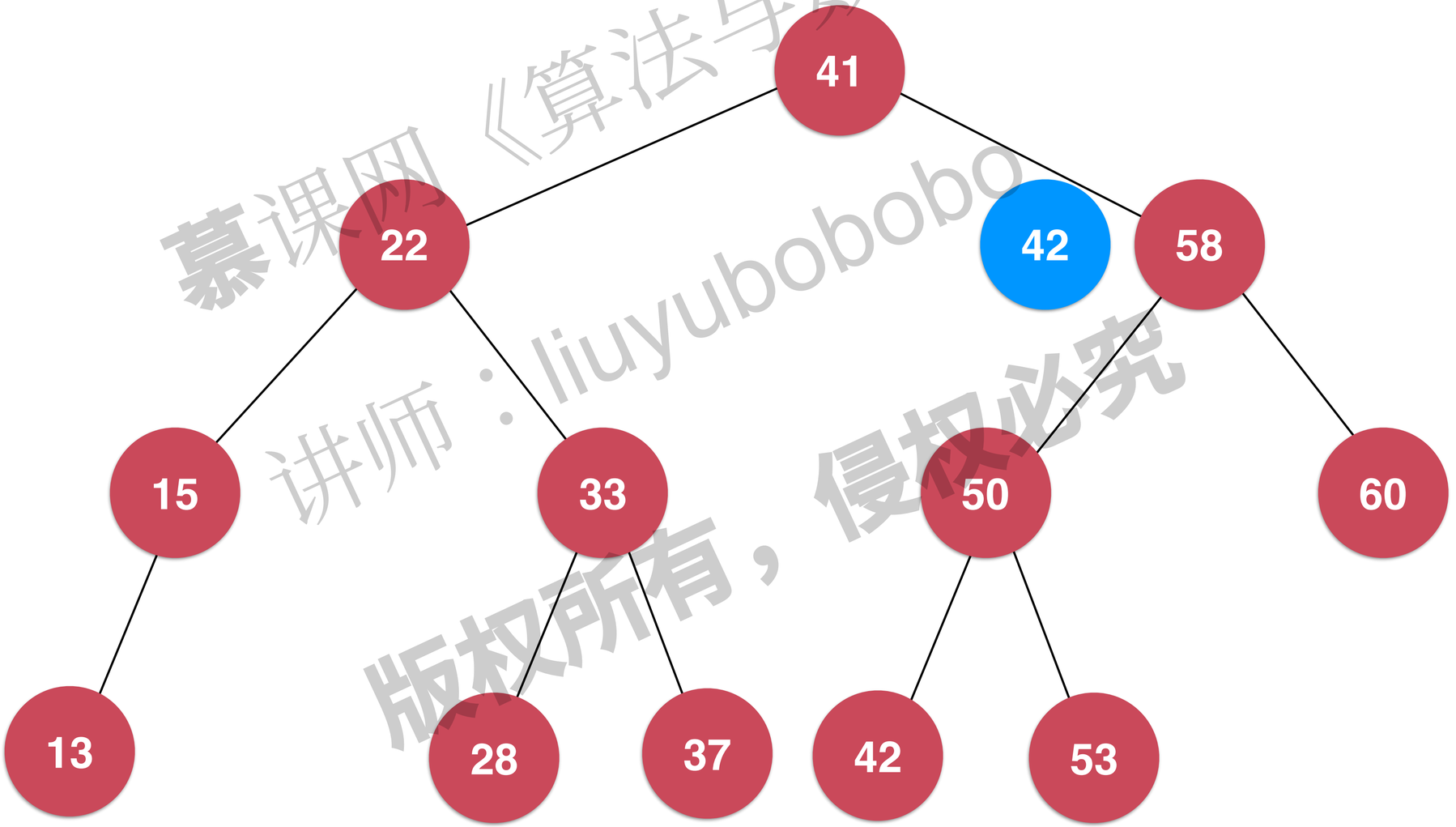
插入新节点 insert



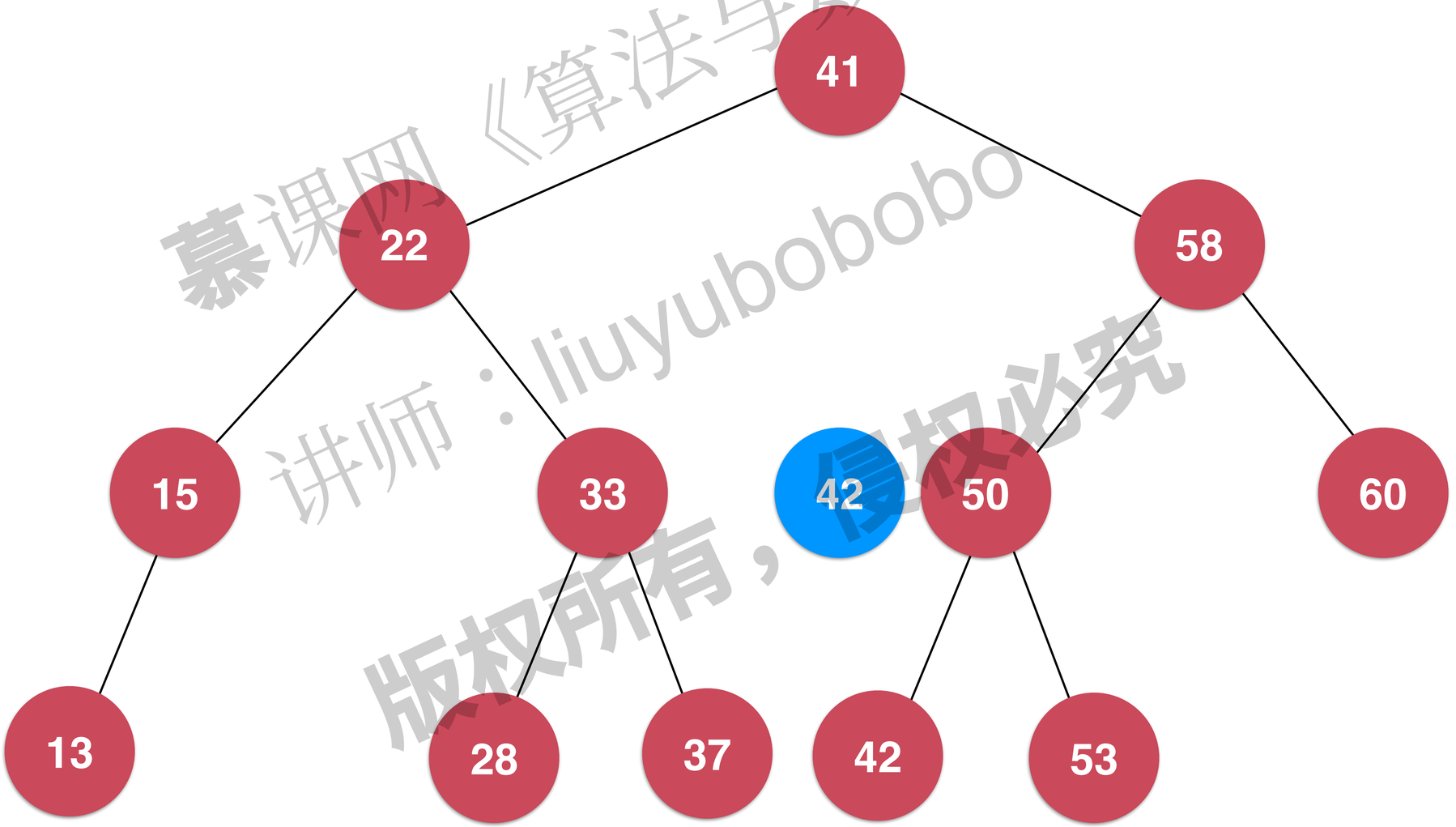
插入新节点 insert



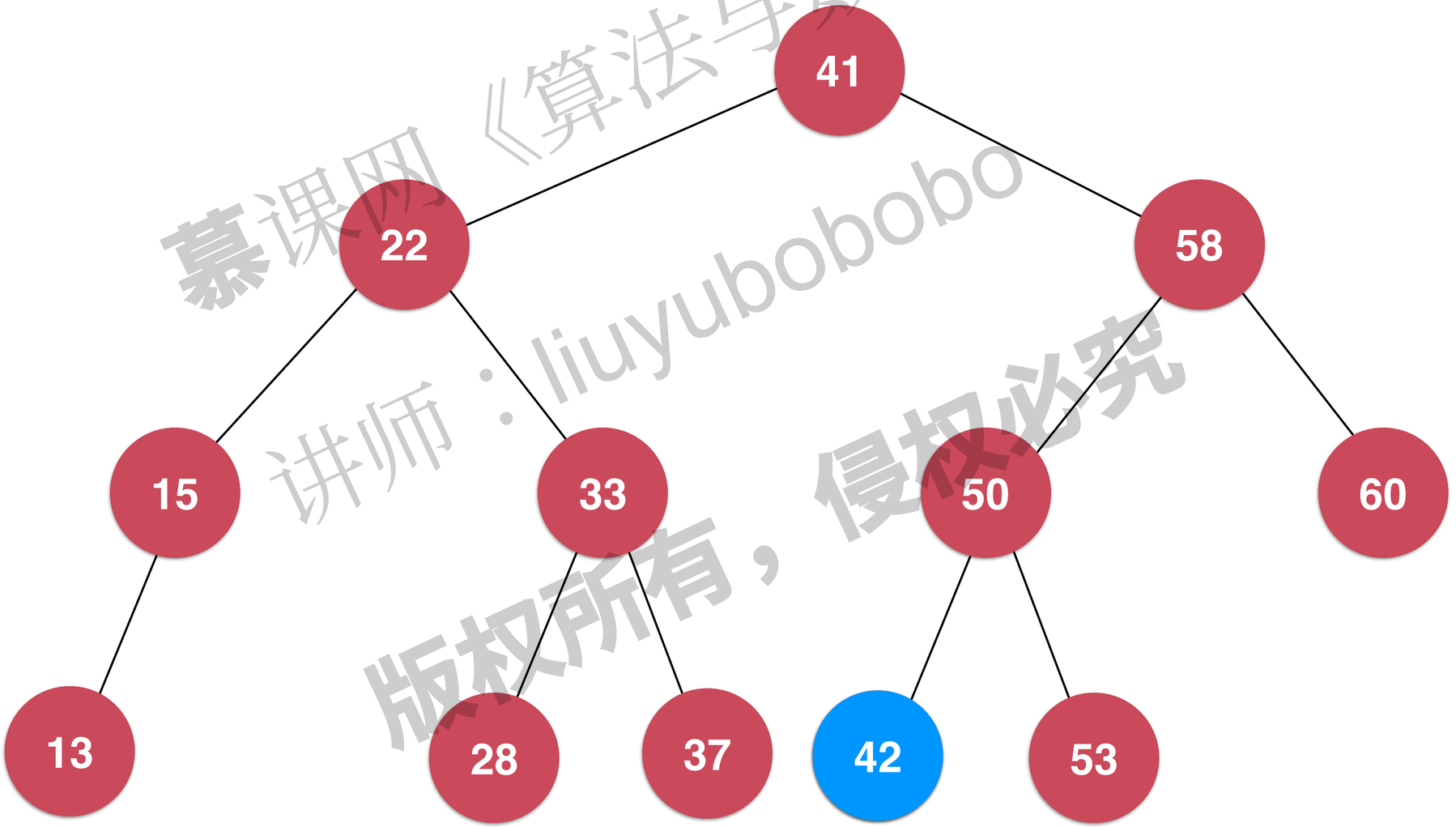
插入新节点 insert



插入新节点 insert



插入新节点 insert

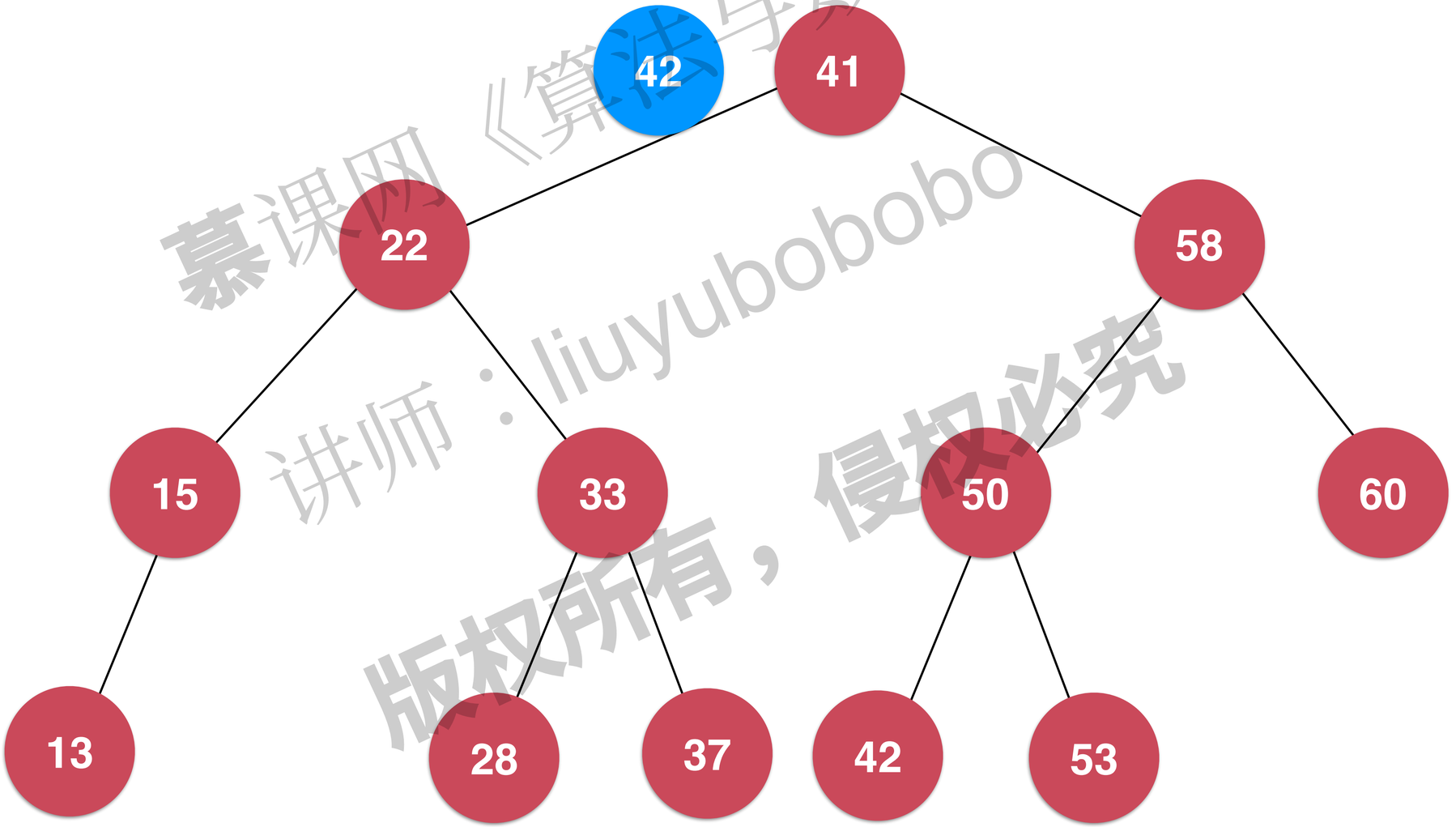


操作：二分查找树插入新节点：insert

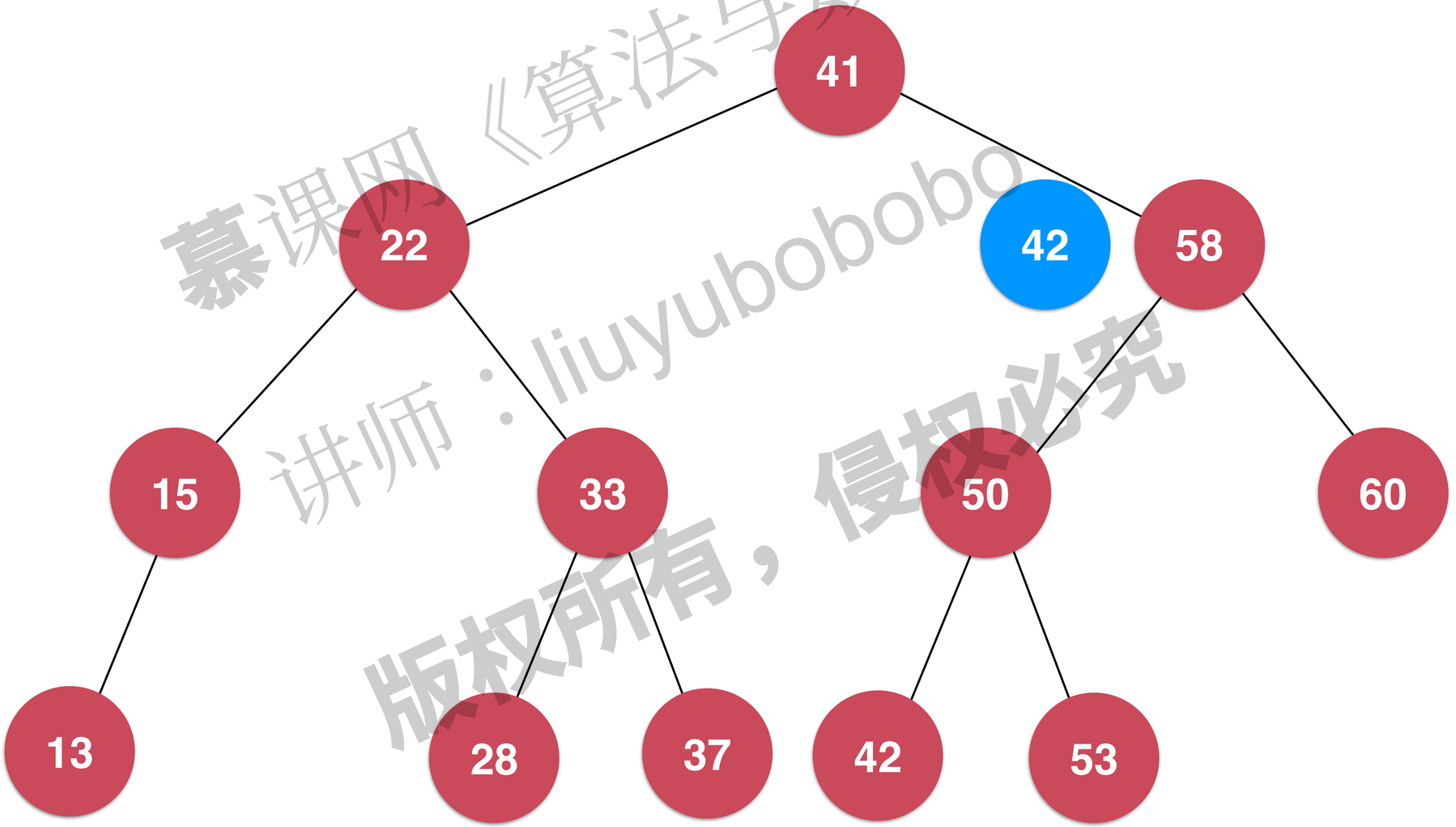
练习：insert的非递归写法

二分查找树的查找

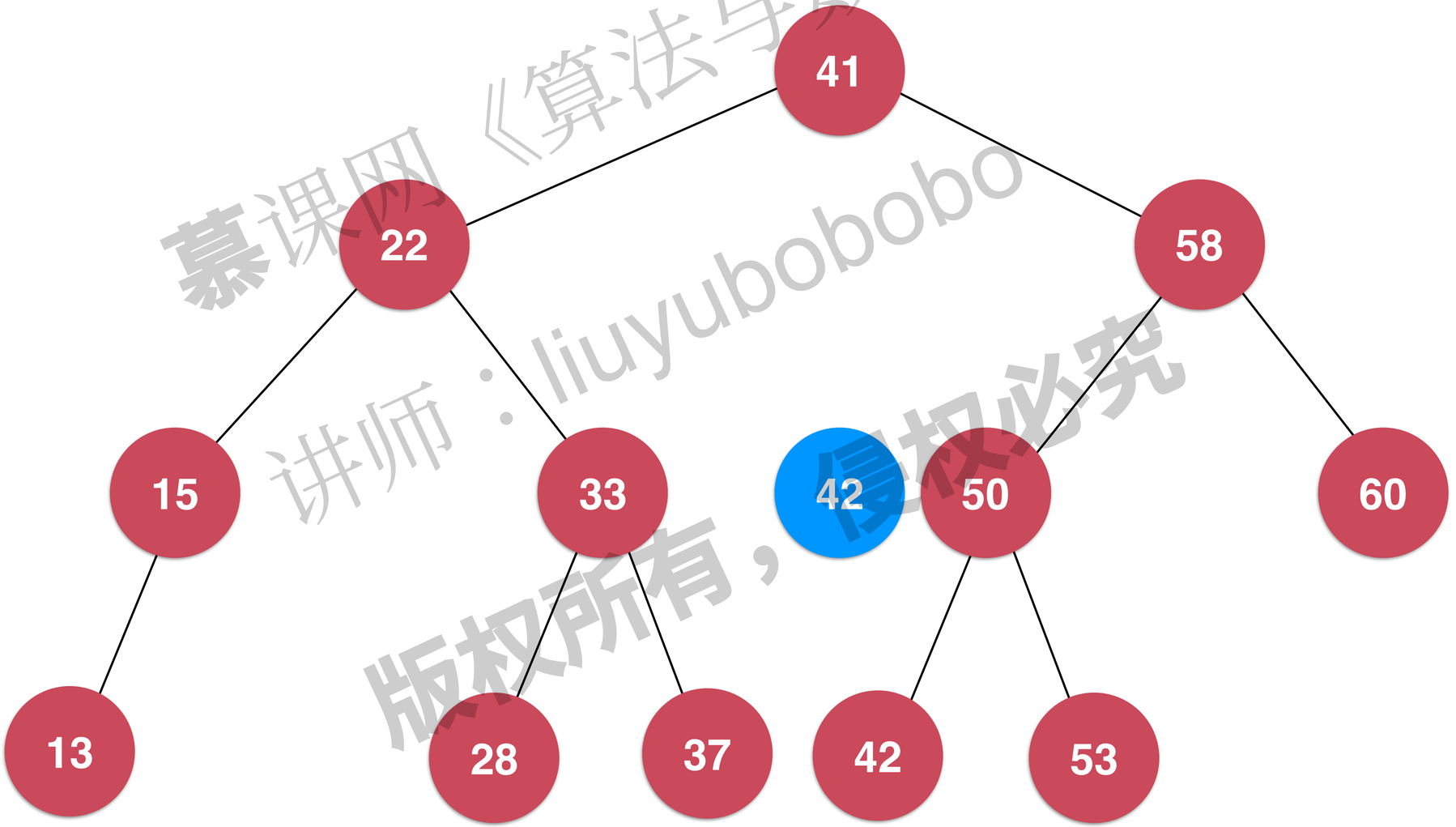
查找 search



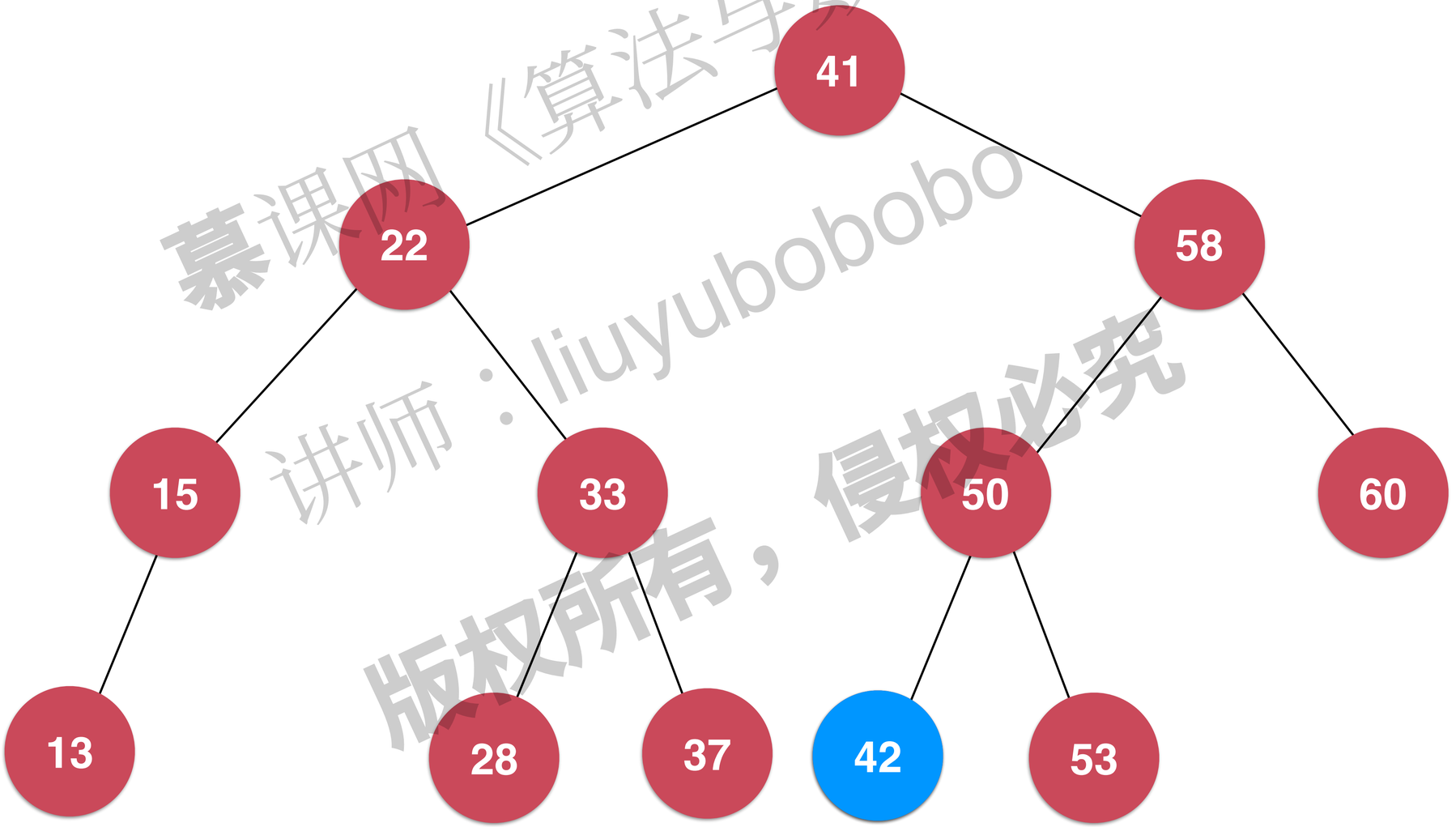
查找 search



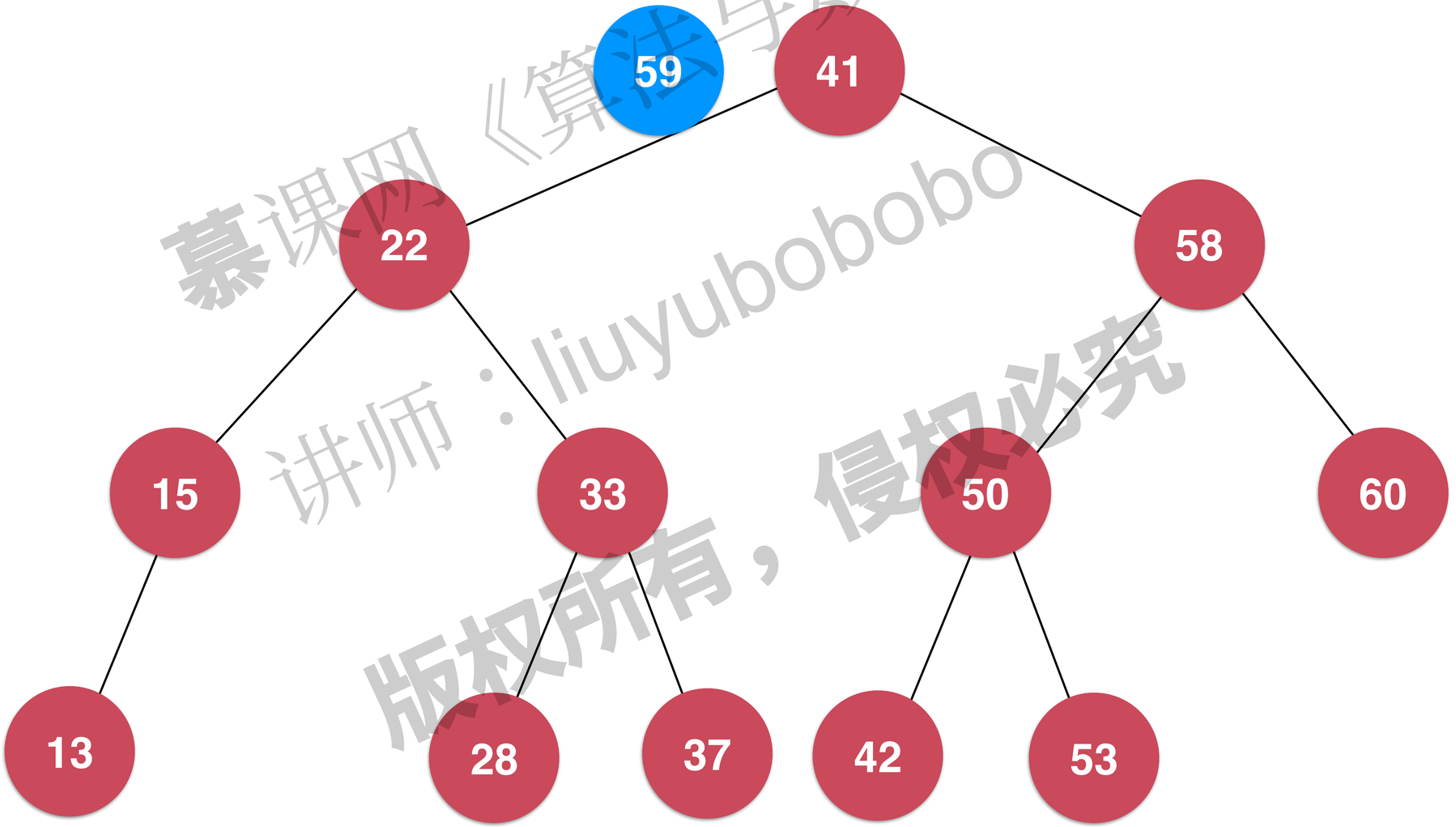
查找 search



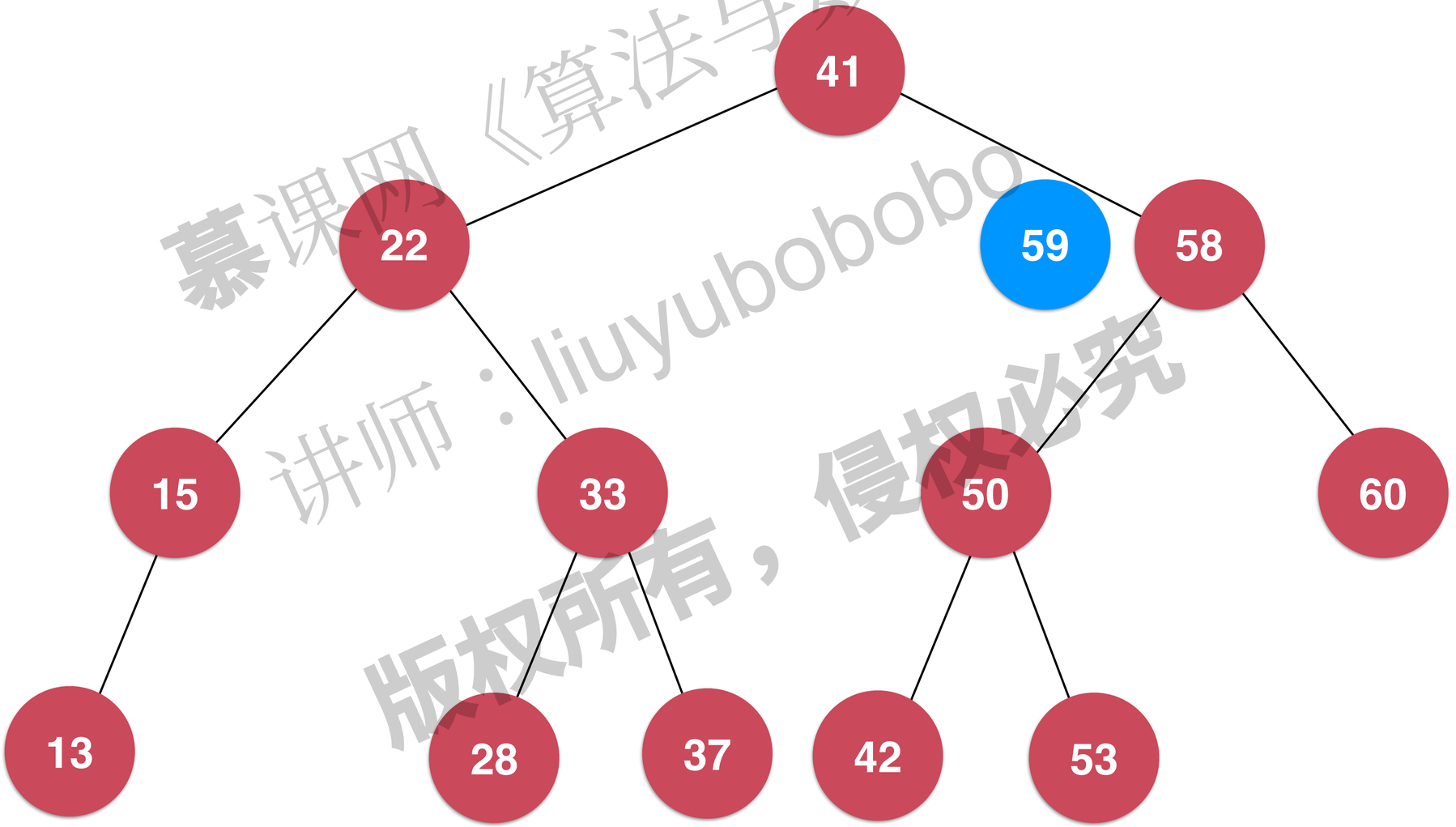
查找 search



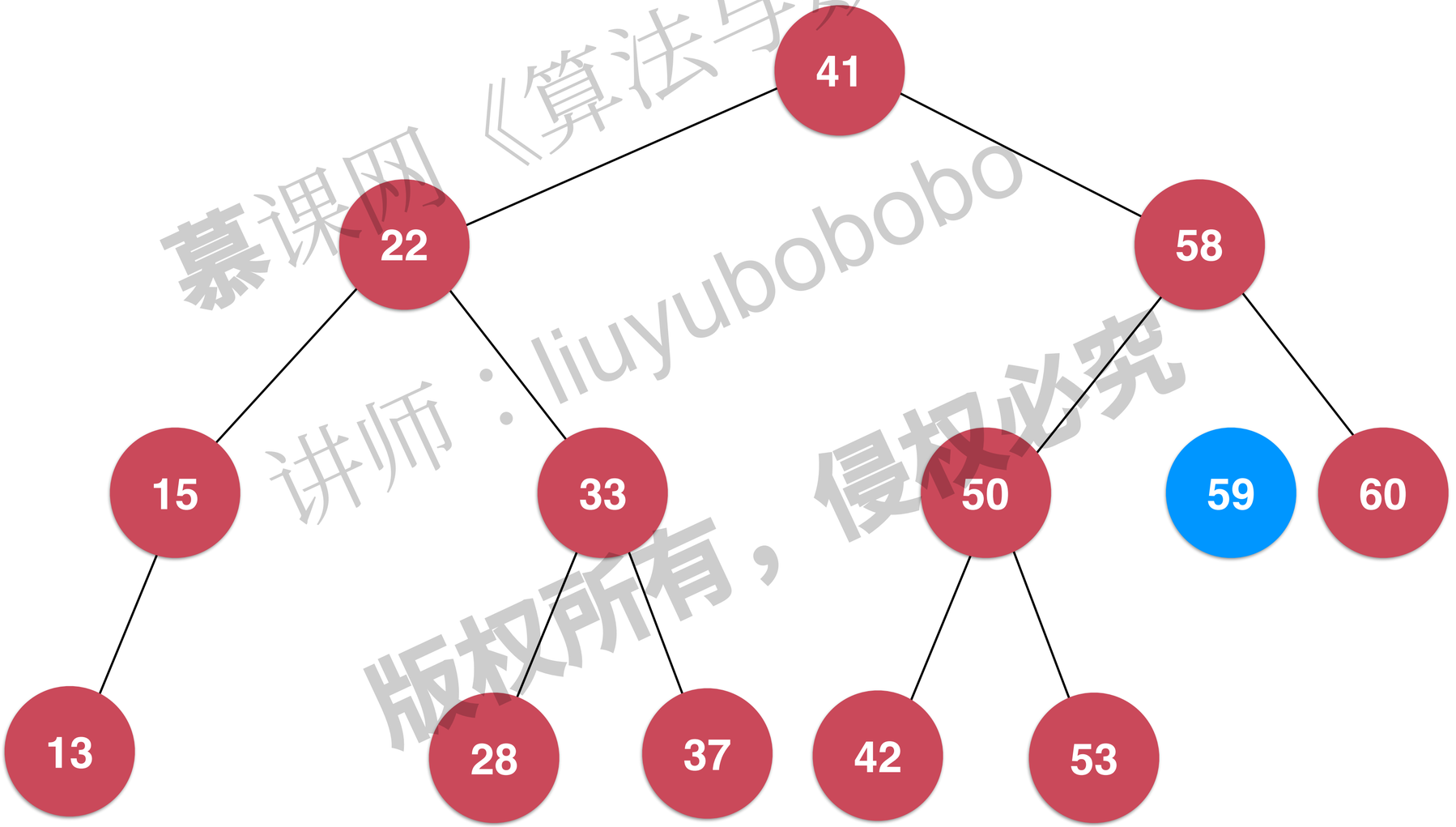
查找 search



查找 search



查找 search



二分搜索树的包含 contain 和查找 search 同质

操作：二分搜索树的包含 contain

操作：二分搜索树的查找 search

练习：search 和 contain 的非递归写法

操作：二分搜索树的速度优势

二分搜索树的遍历

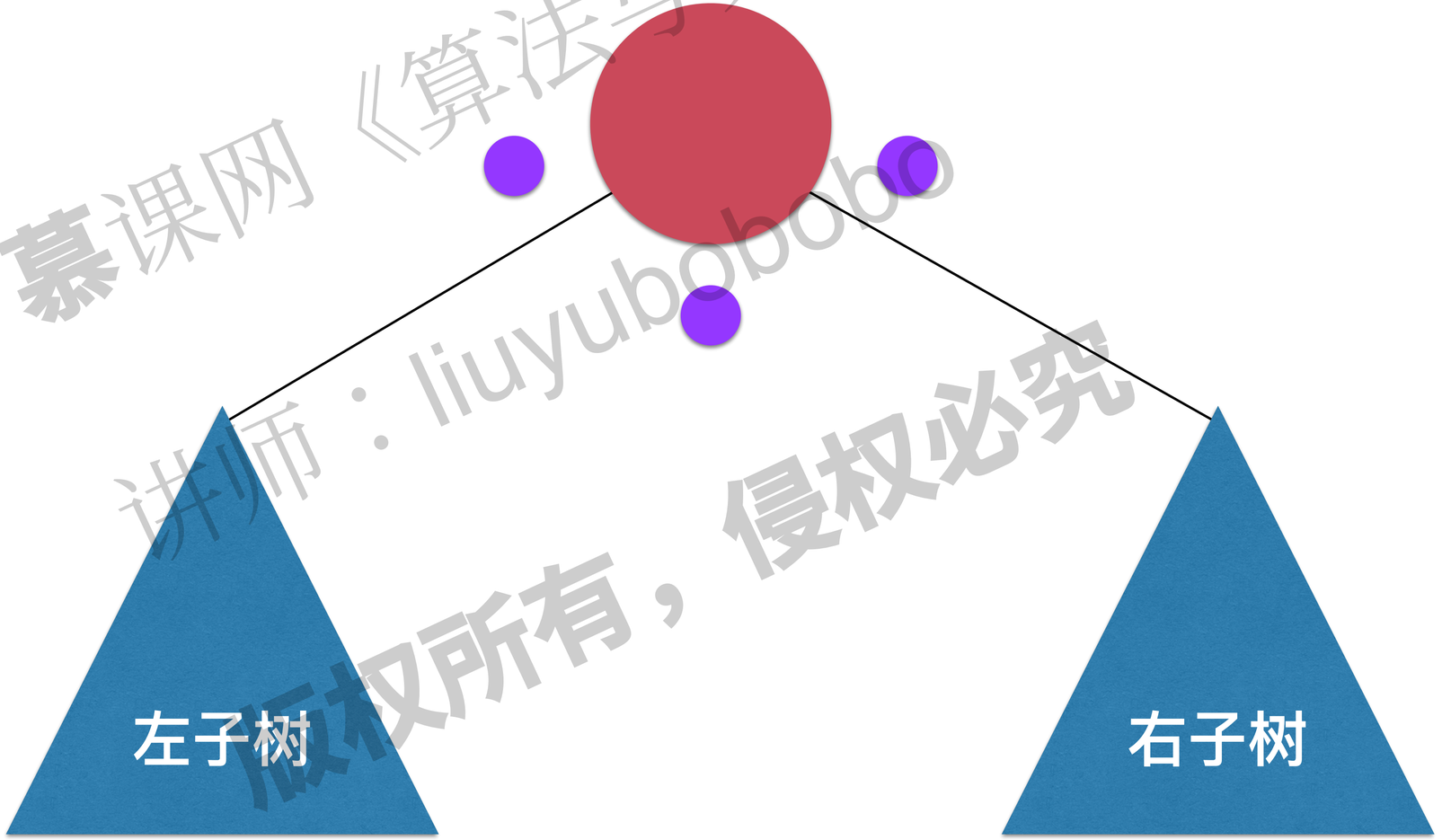
二分搜索树的前中后序遍历

前序遍历：先访问当前节点，再依次递归访问左右子树

中序遍历：先递归访问左子树，再访问自身，再递归访问右子树

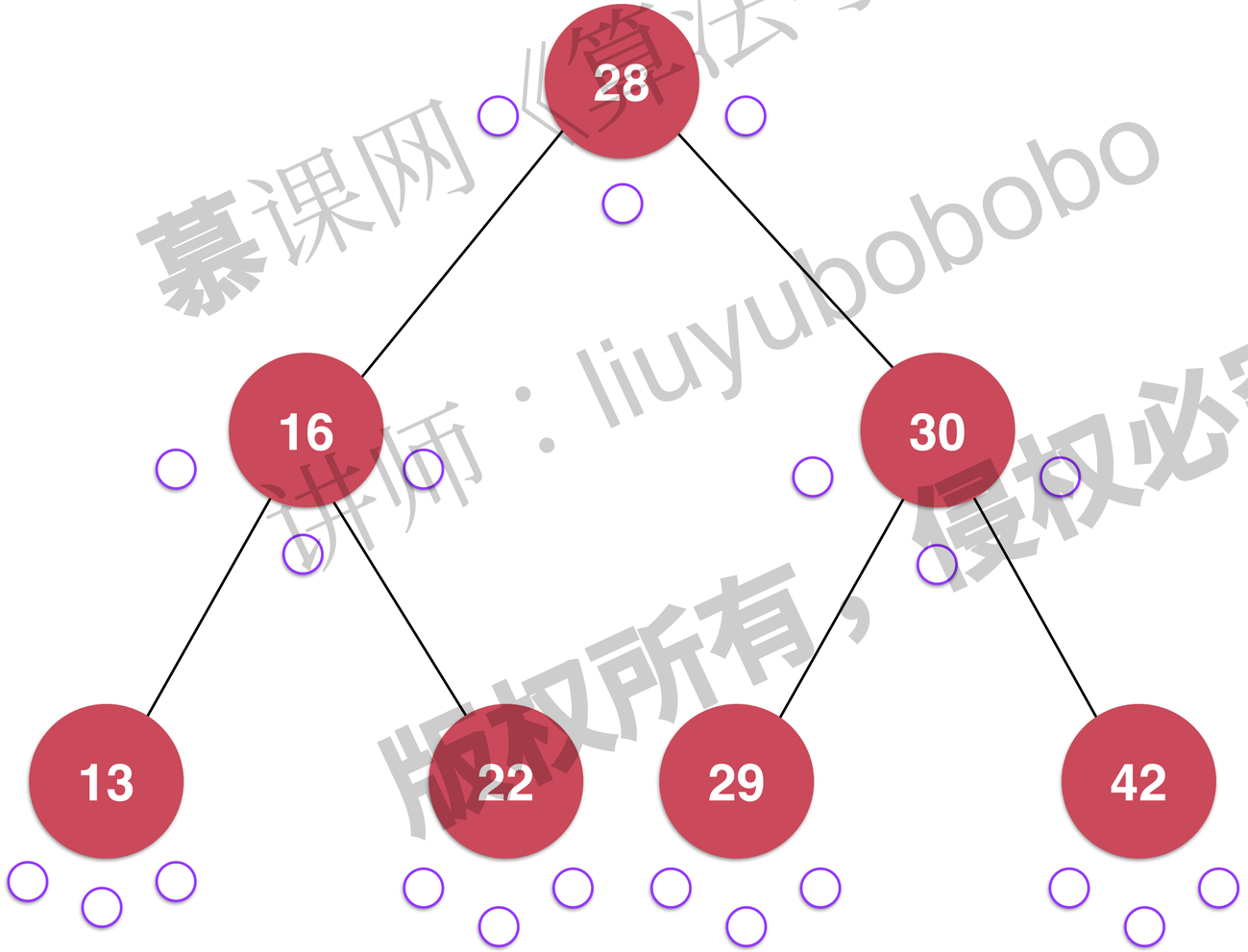
后续遍历：先递归访问左右子树，再访问自身节点

二分搜索树的遍历

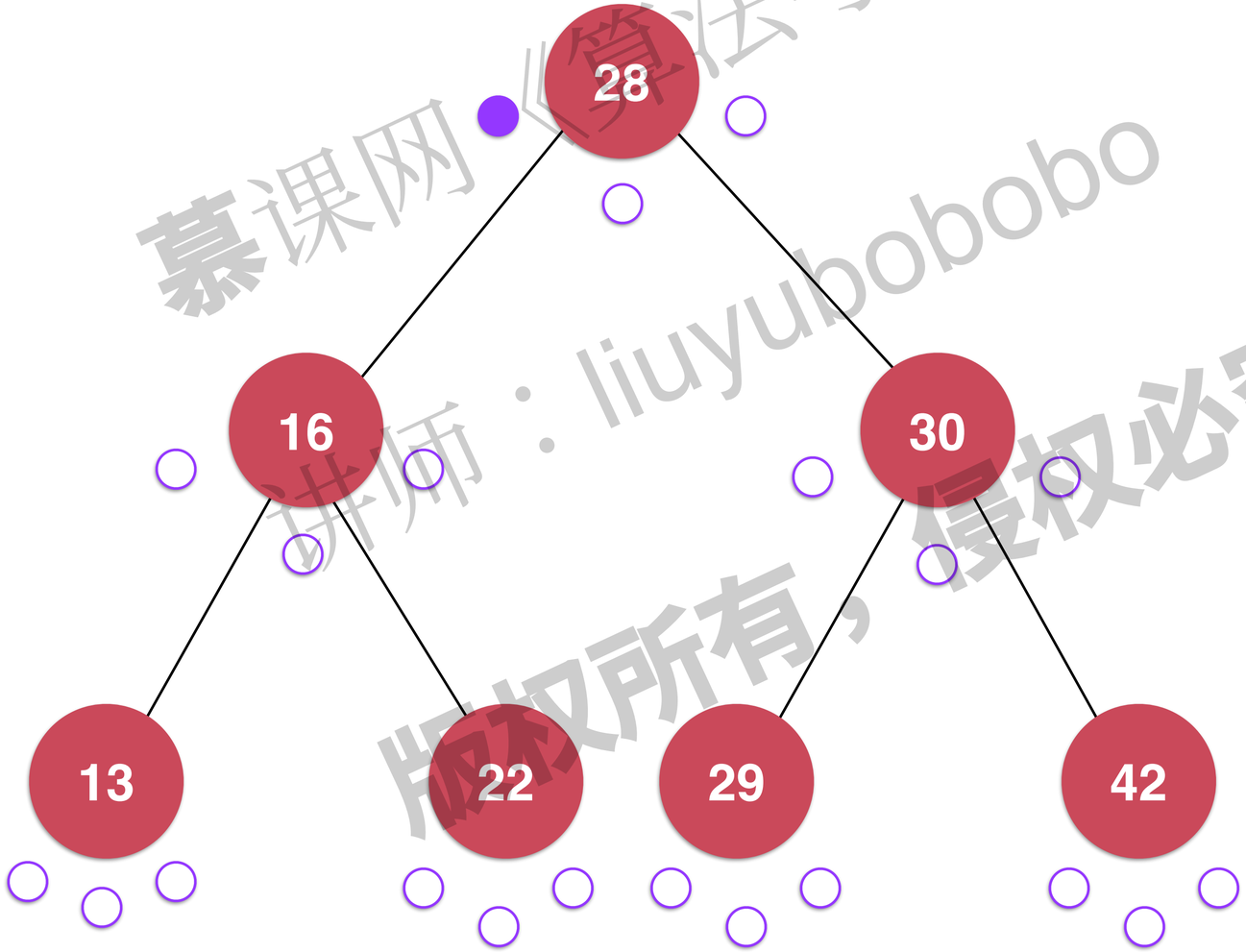


前序遍历

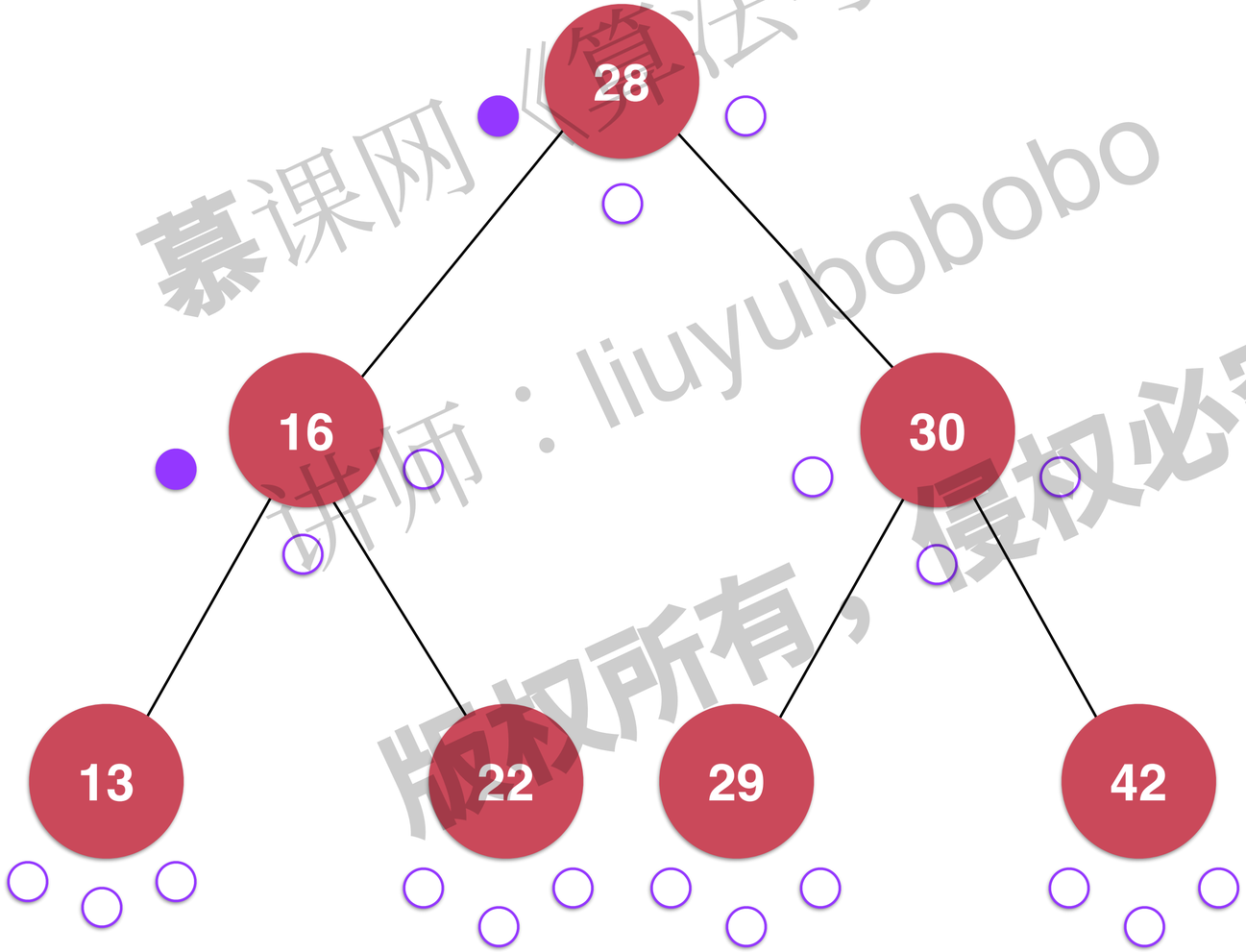
二分搜索树的前序遍历



二分搜索树的前序遍历



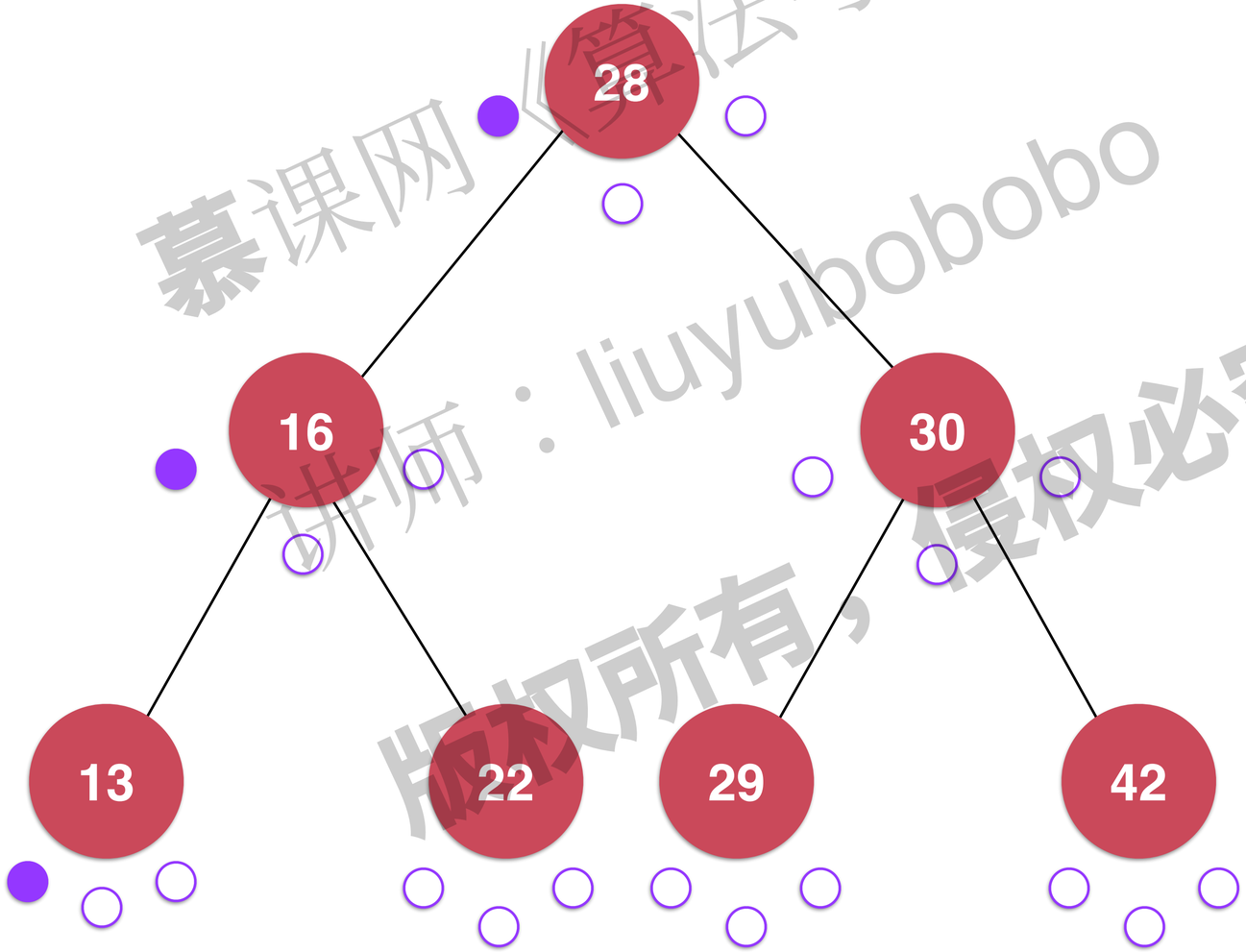
二分搜索树的前序遍历



2

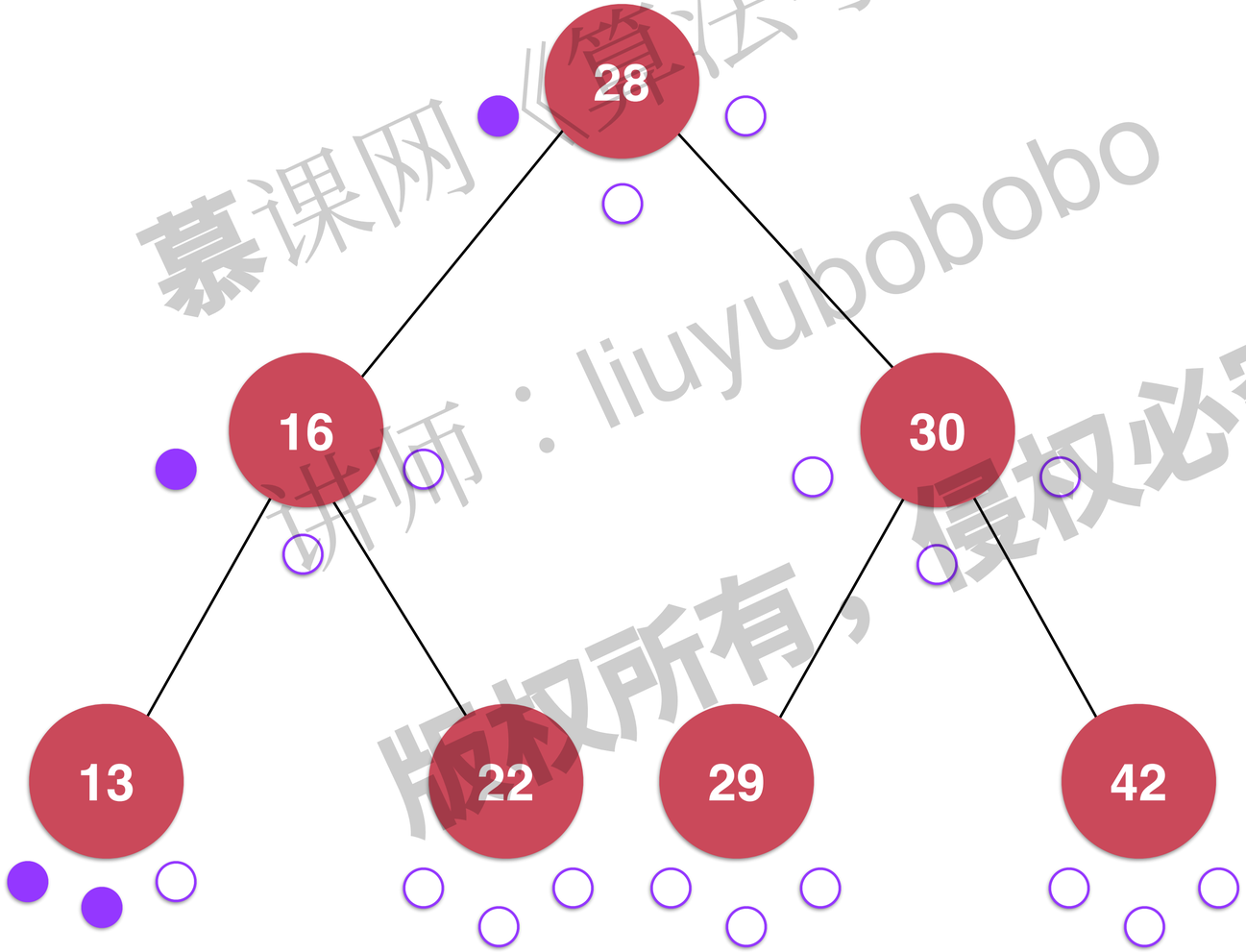
1

二分搜索树的前序遍历



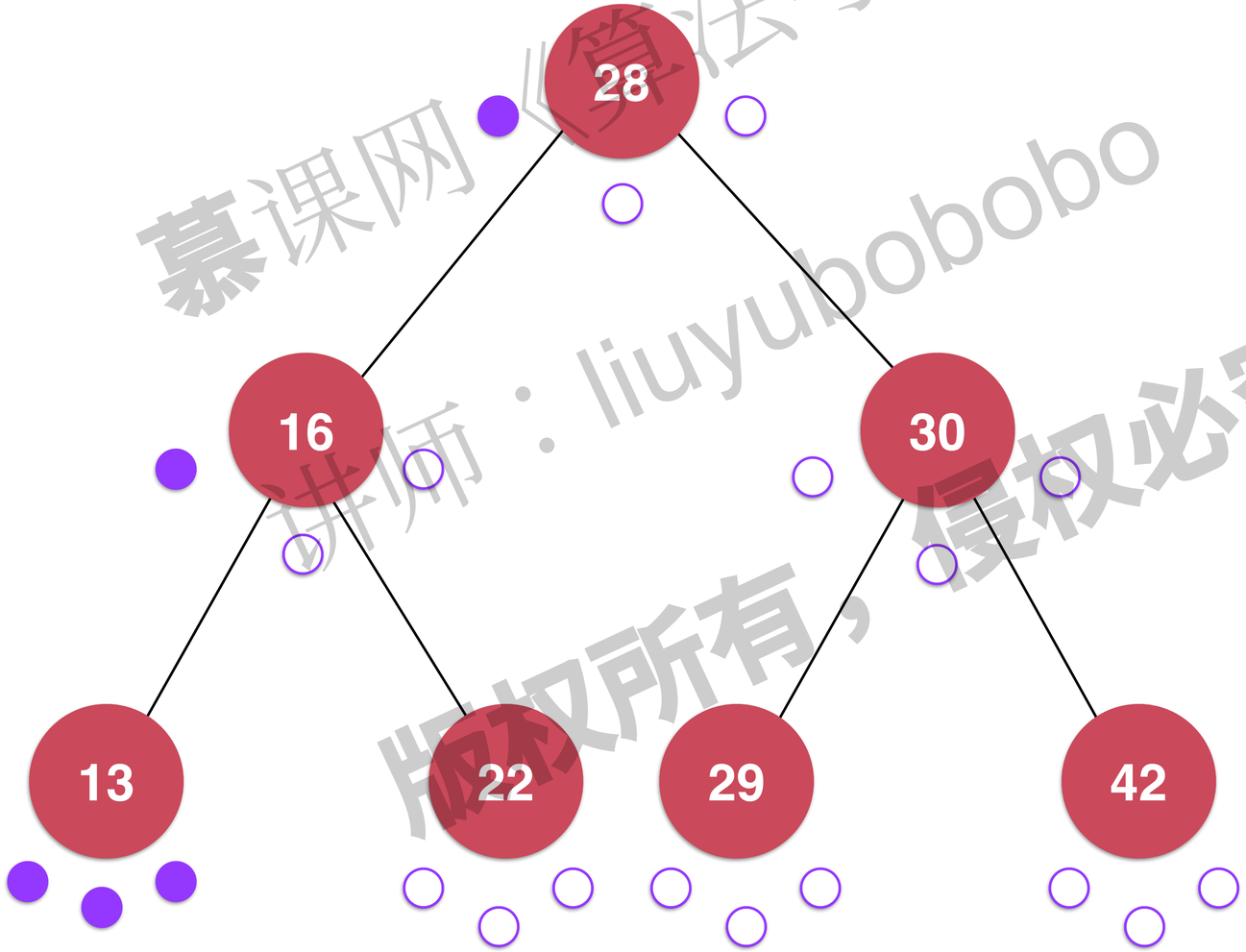
2
1
1

二分搜索树的前序遍历



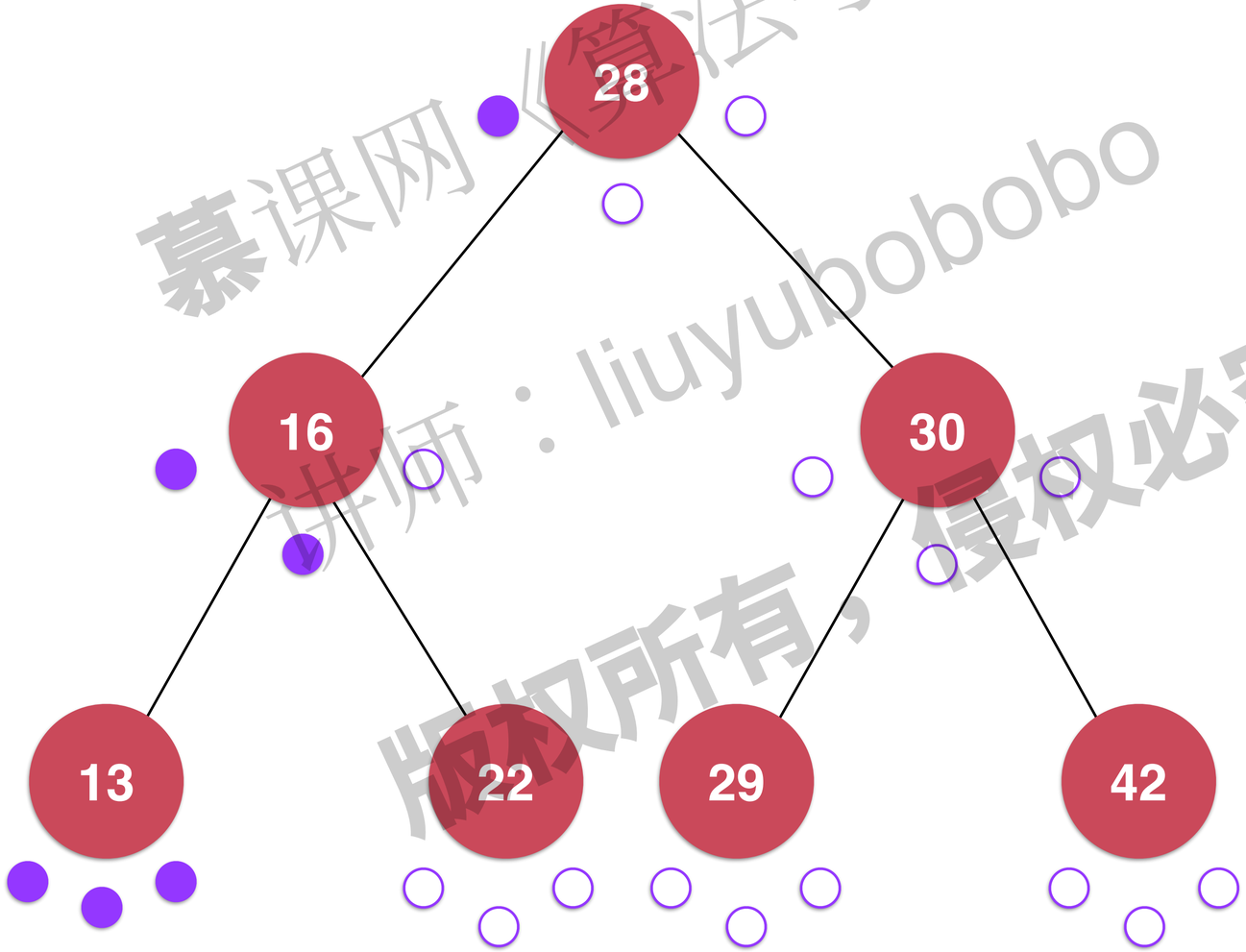
2
1
1

二分搜索树的前序遍历



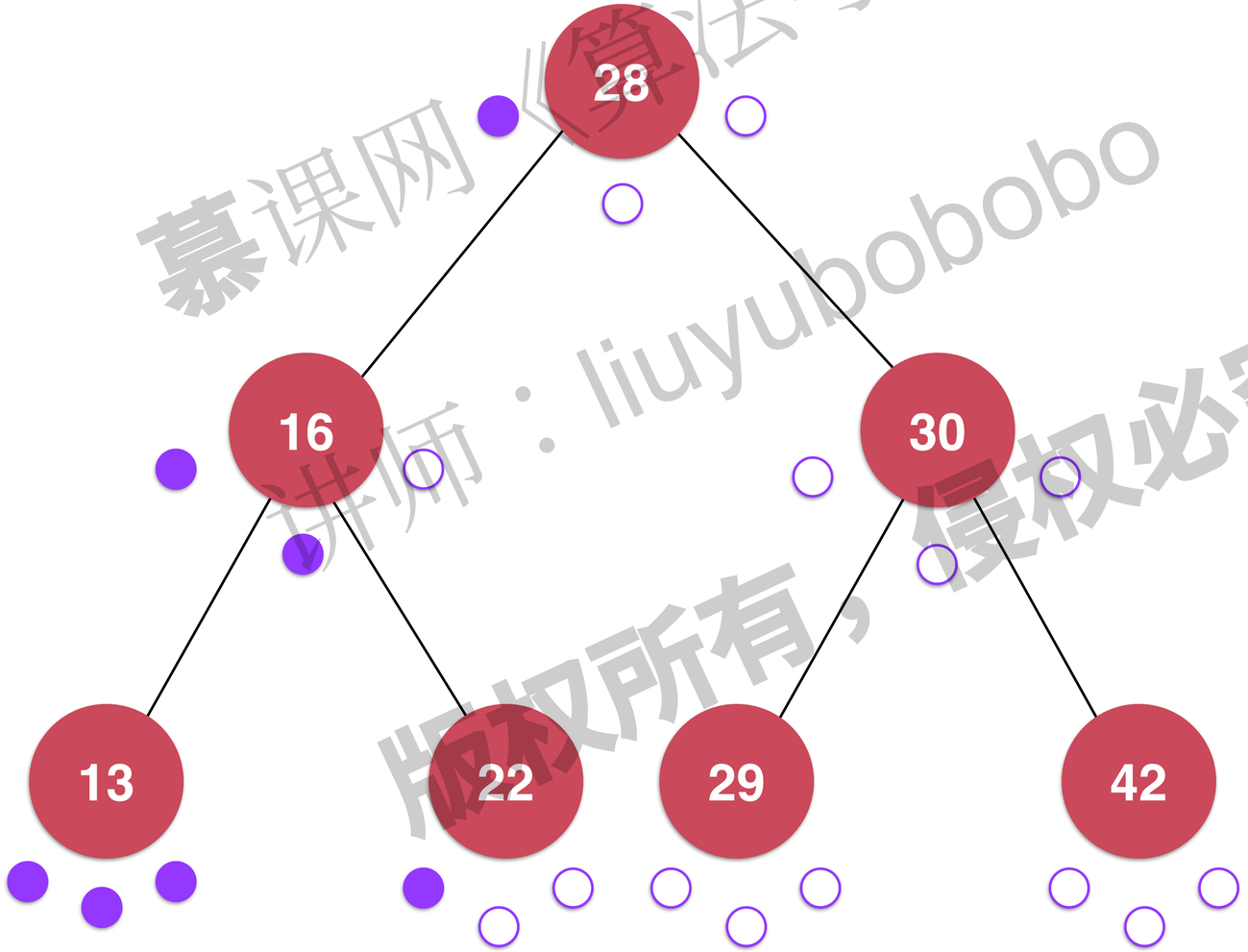
2
1
1

二分搜索树的前序遍历



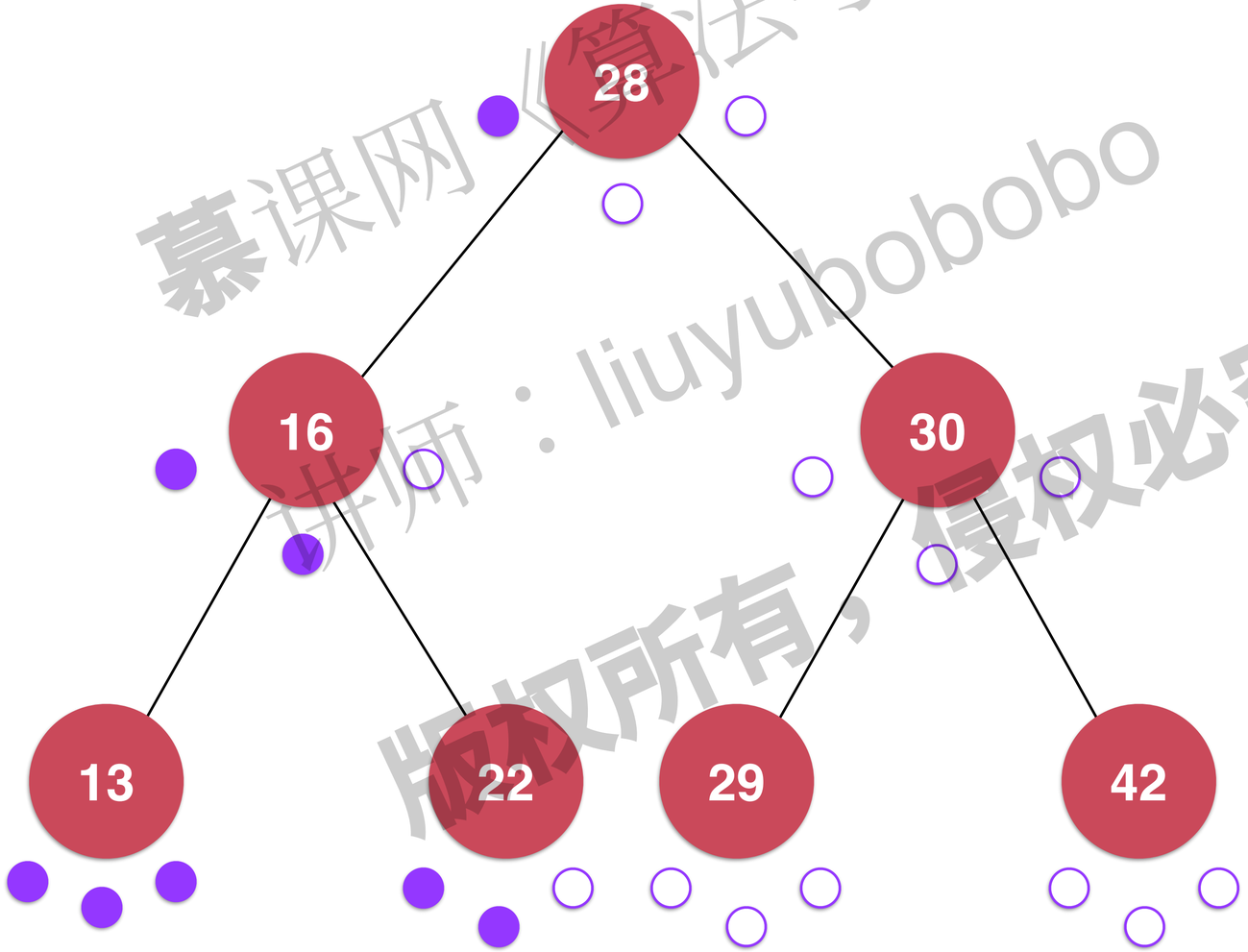
2
1
1

二分搜索树的前序遍历



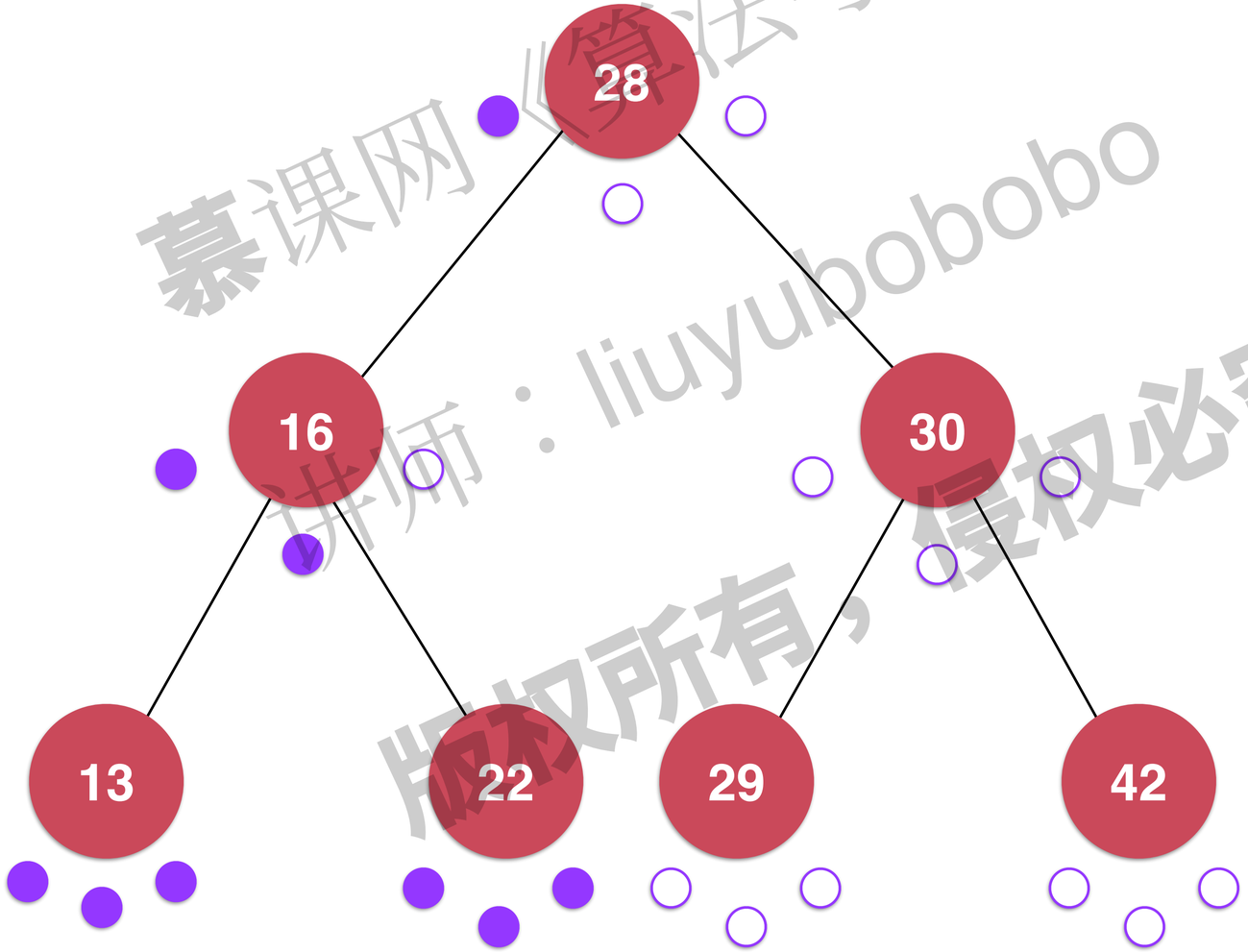
2
1
1
2

二分搜索树的前序遍历



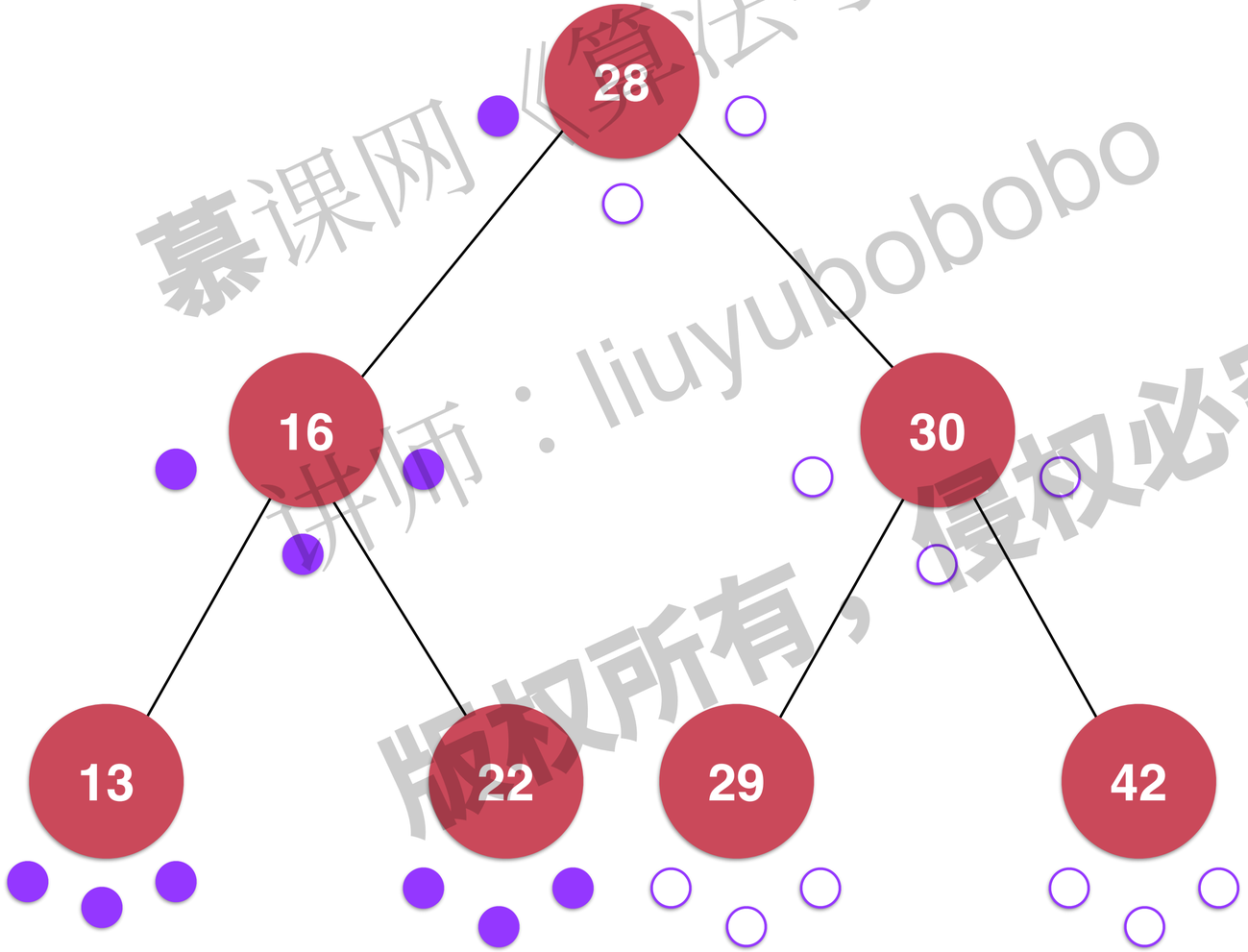
2
1
1
2

二分搜索树的前序遍历



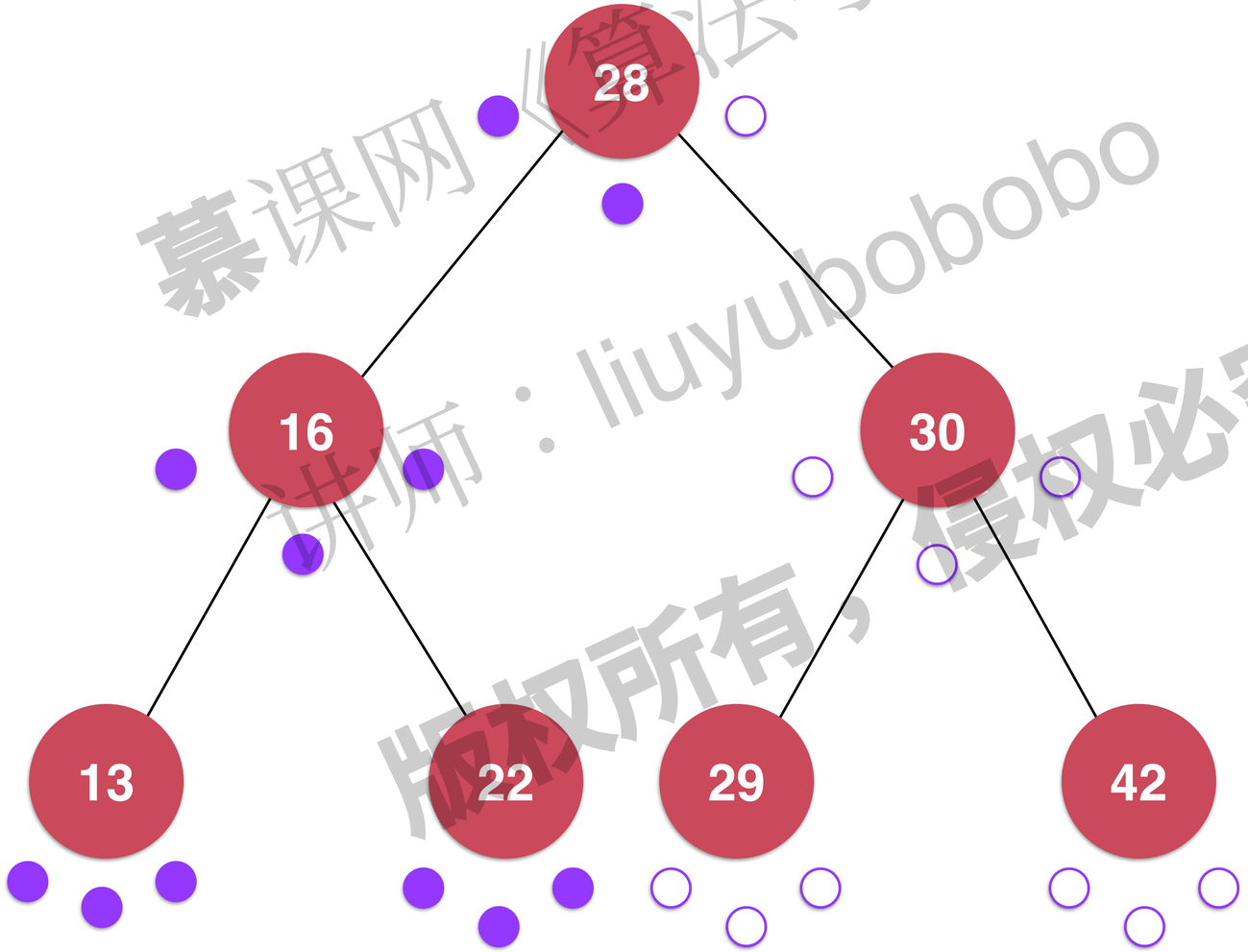
2
1
1
2

二分搜索树的前序遍历



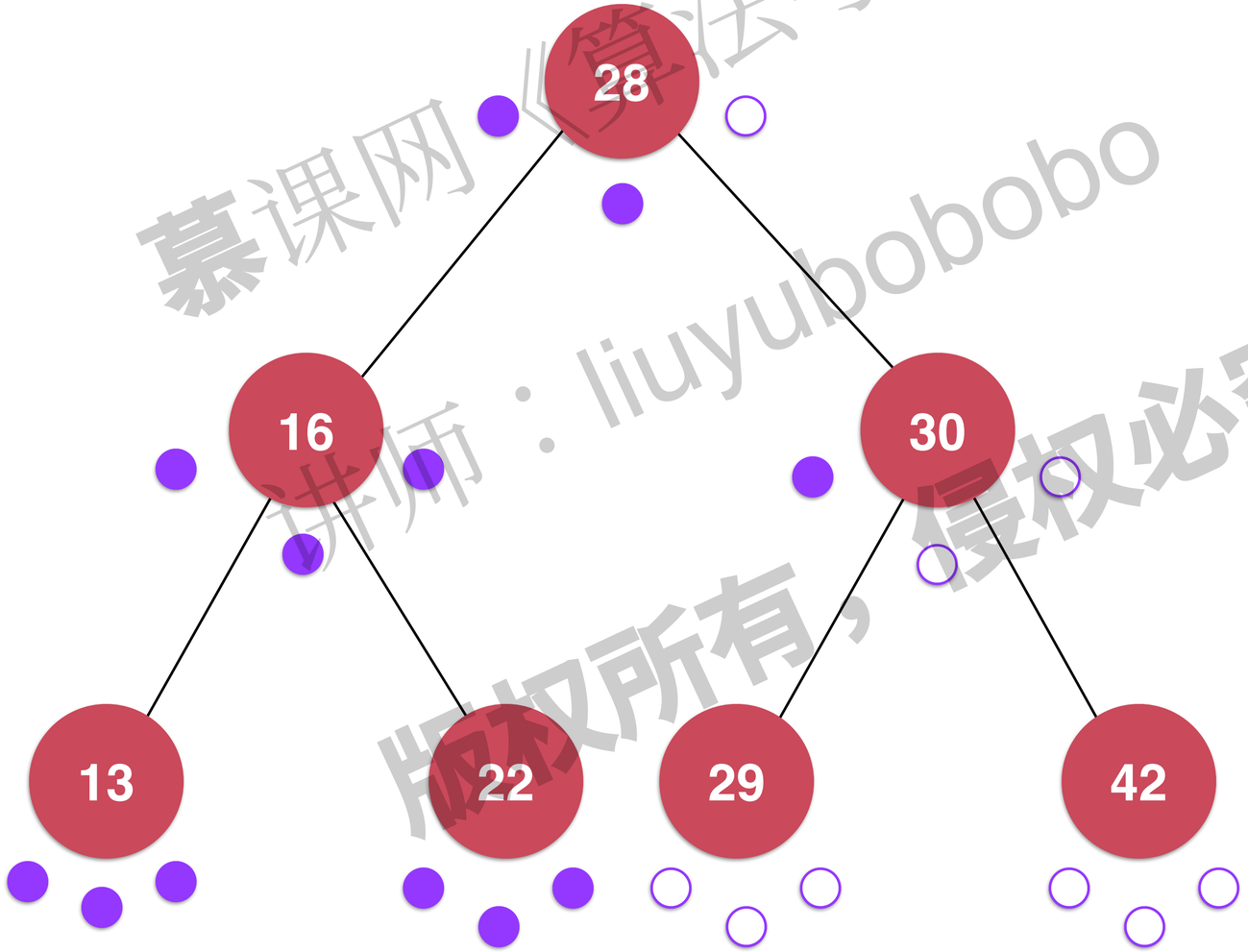
2
1
1
2

二分搜索树的前序遍历



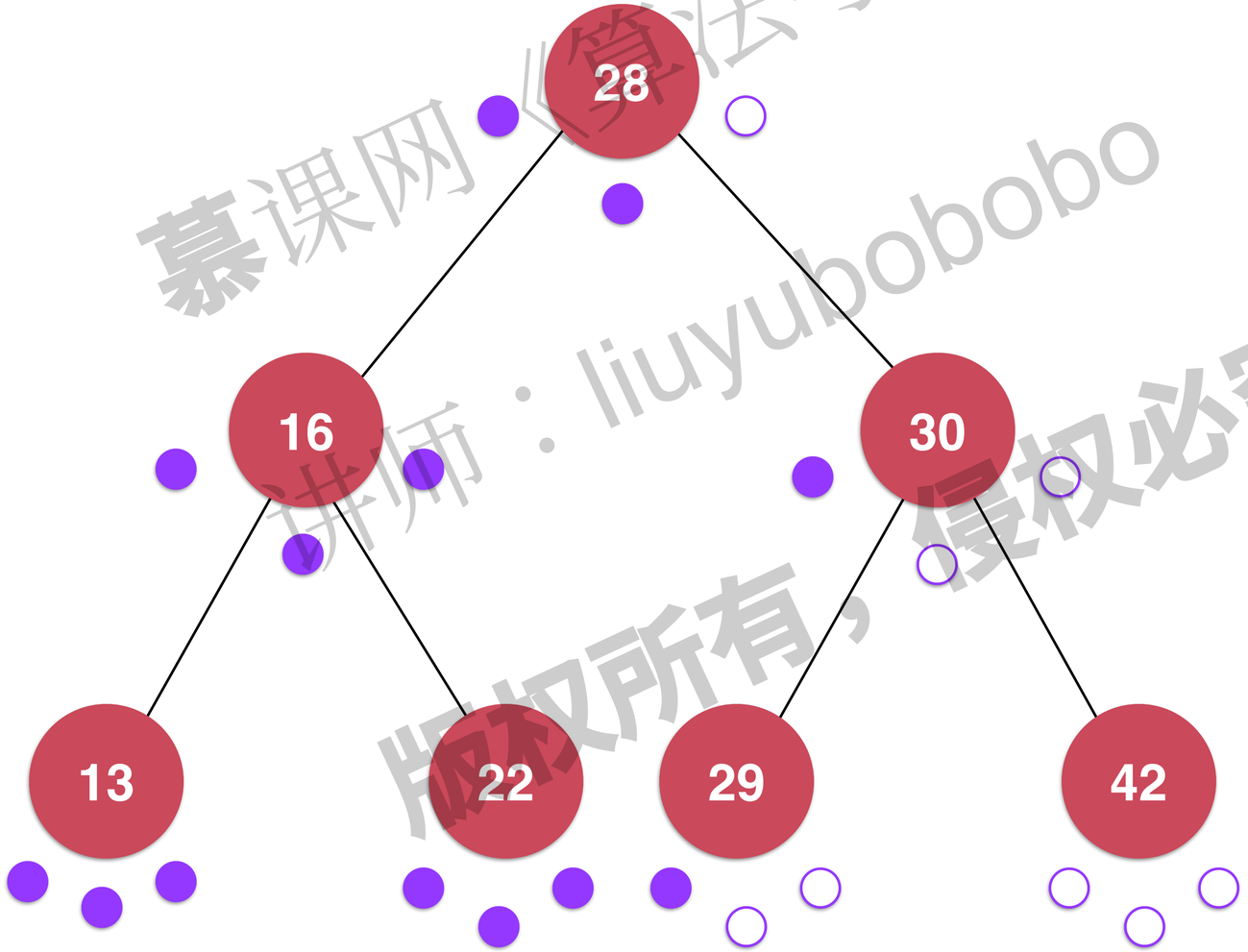
2
1
1
2

二分搜索树的前序遍历



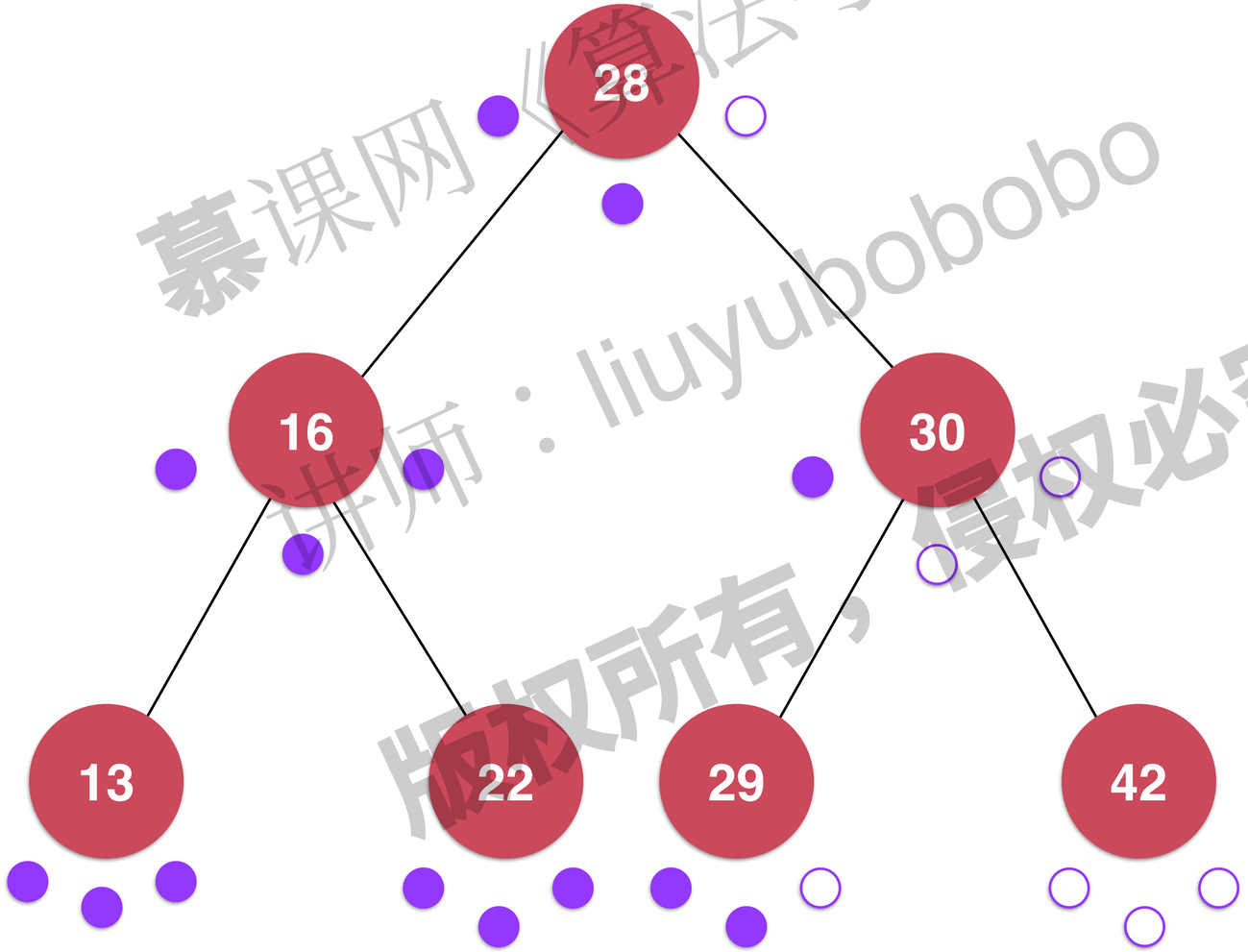
2
1
1
2
3

二分搜索树的前序遍历



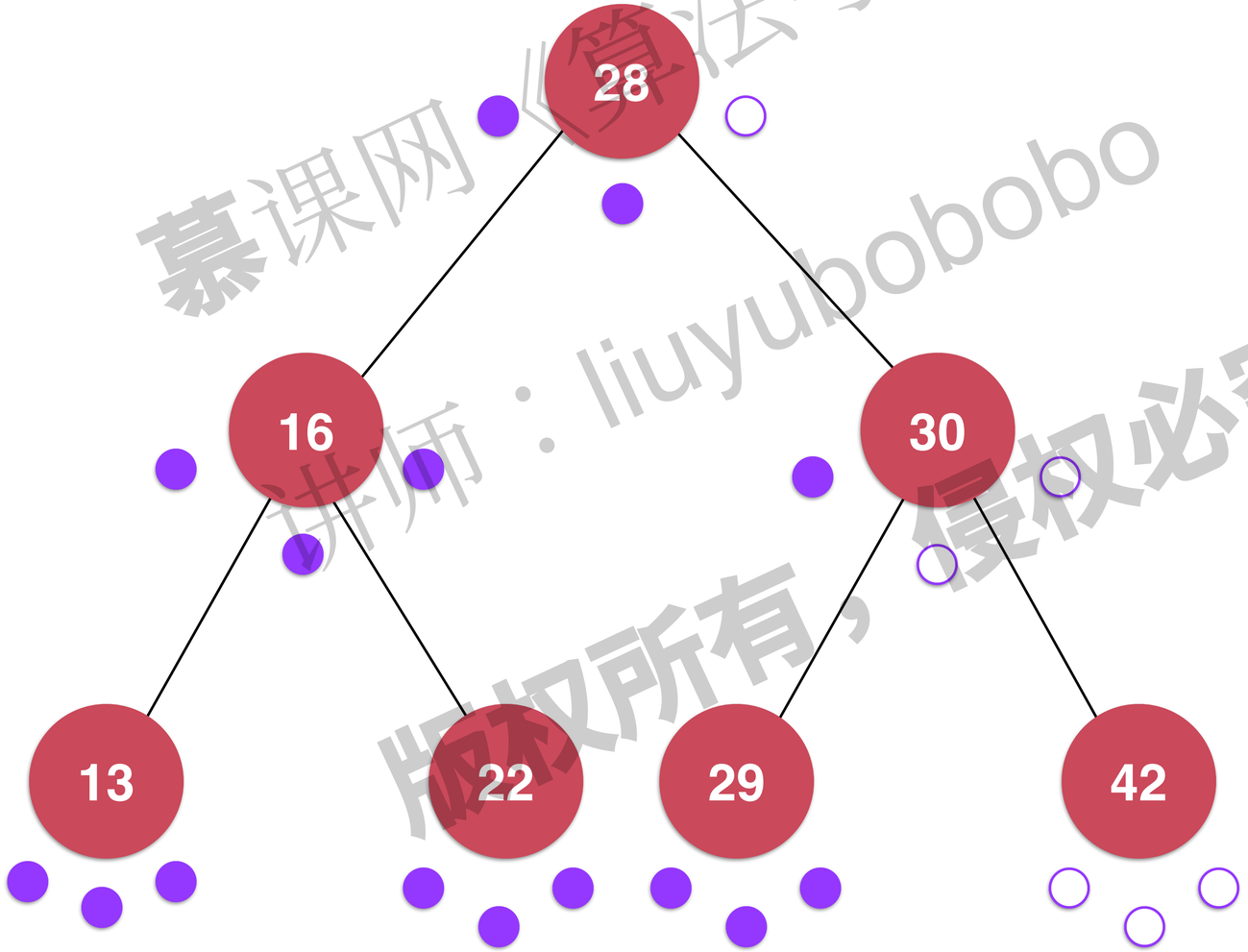
2
1
1
2
3
2

二分搜索树的前序遍历



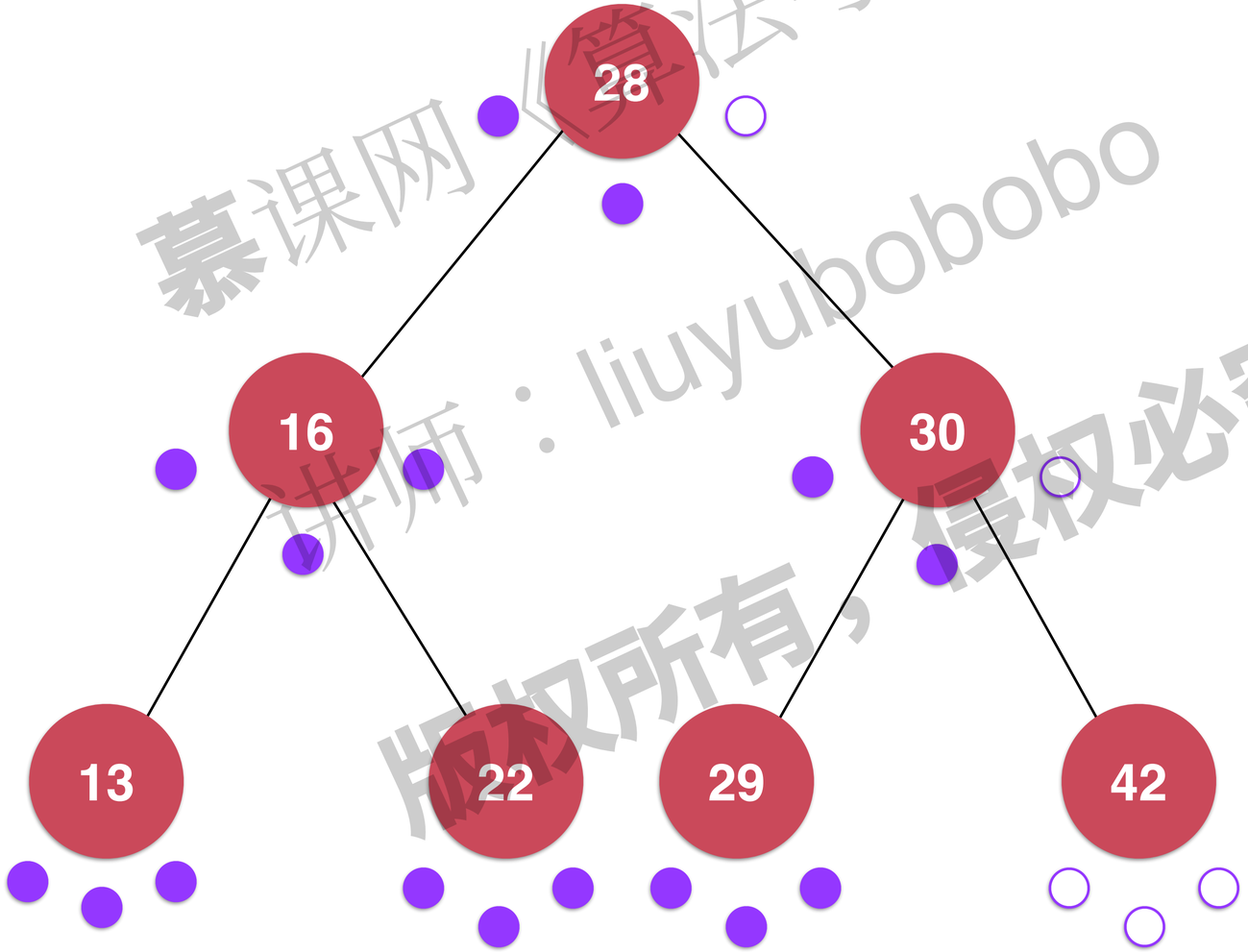
2
1
1
2
3
2

二分搜索树的前序遍历



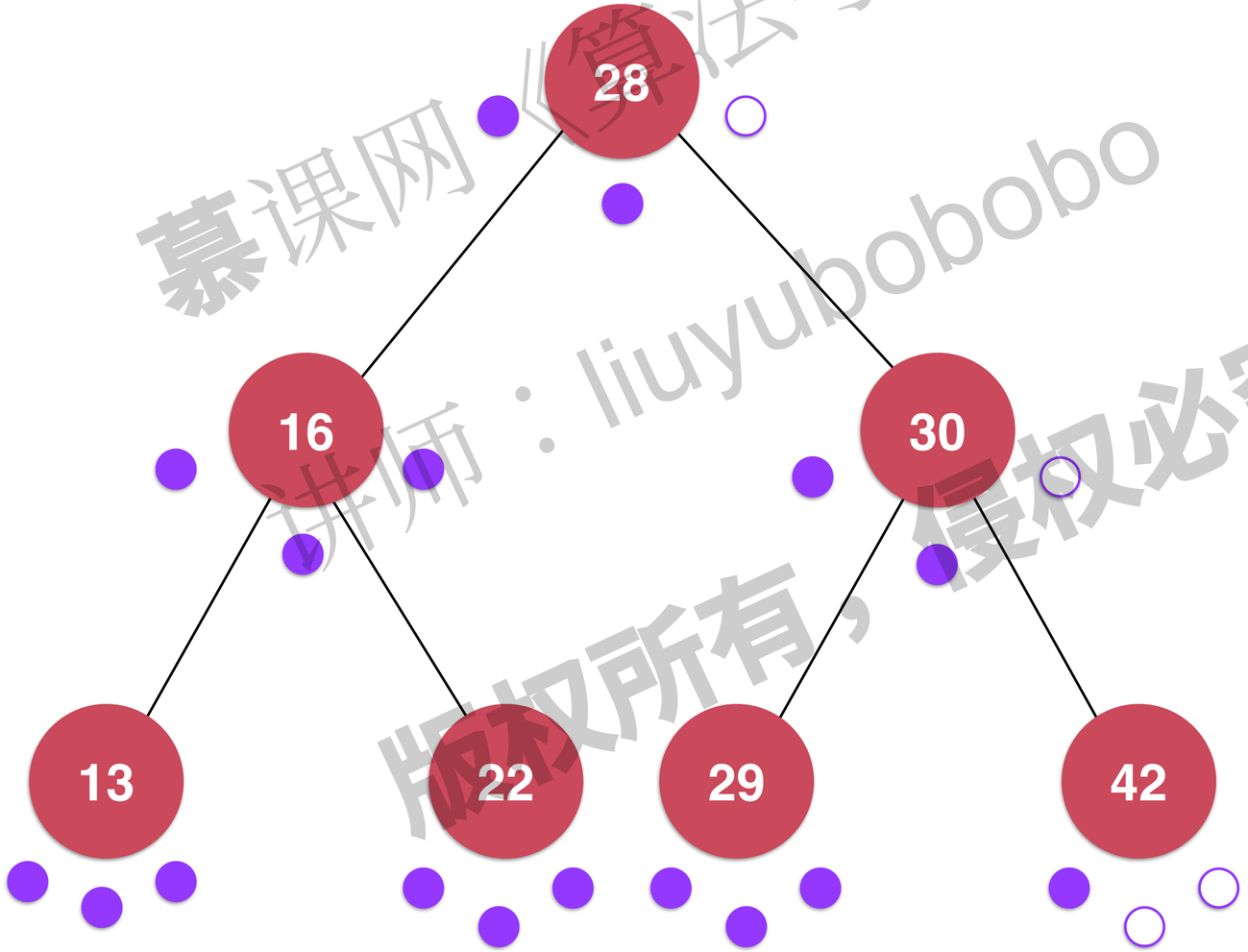
2
1
1
2
3
2

二分搜索树的前序遍历



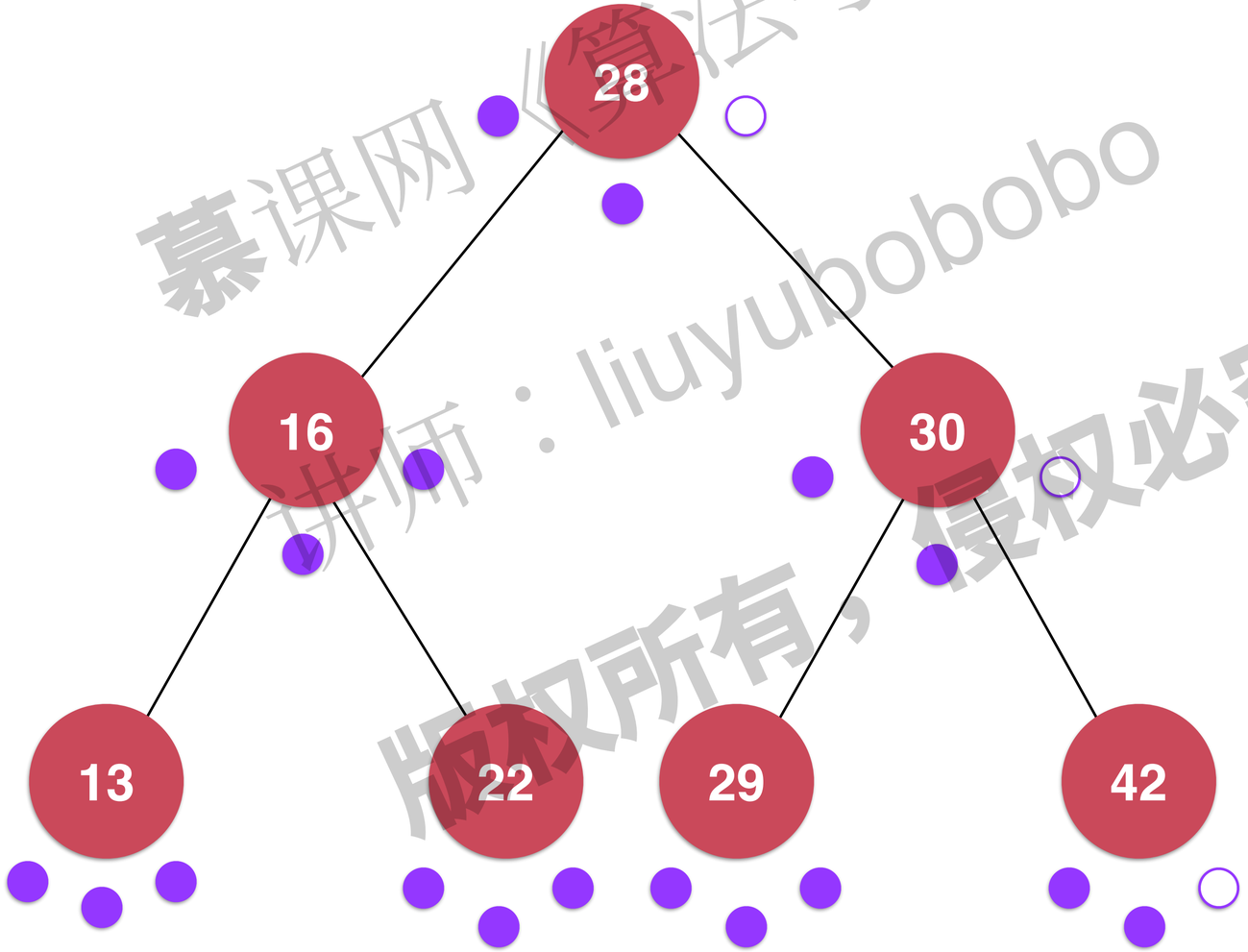
2
1
1
2
3
2

二分搜索树的前序遍历



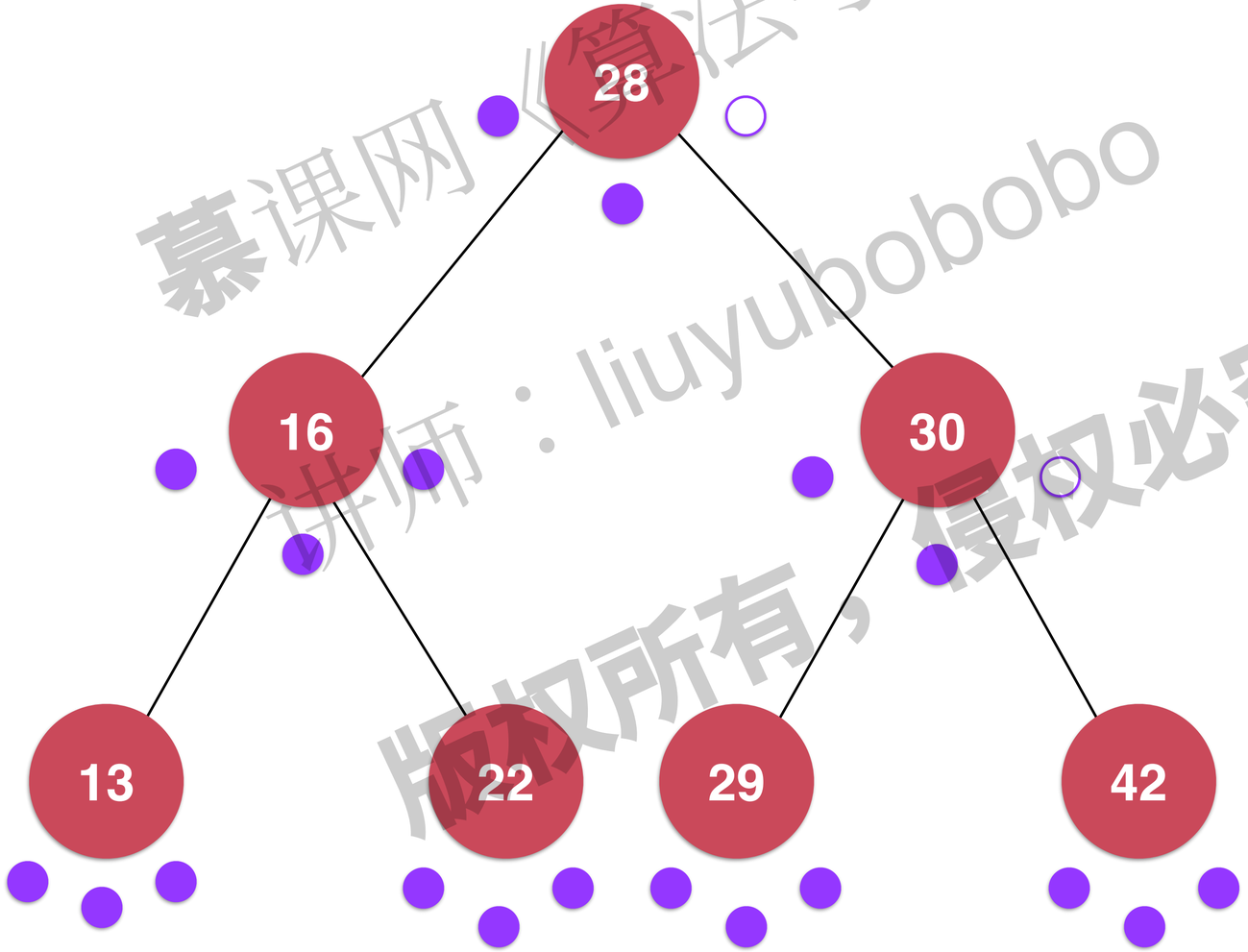
2
1
1
2
3
2
4

二分搜索树的前序遍历



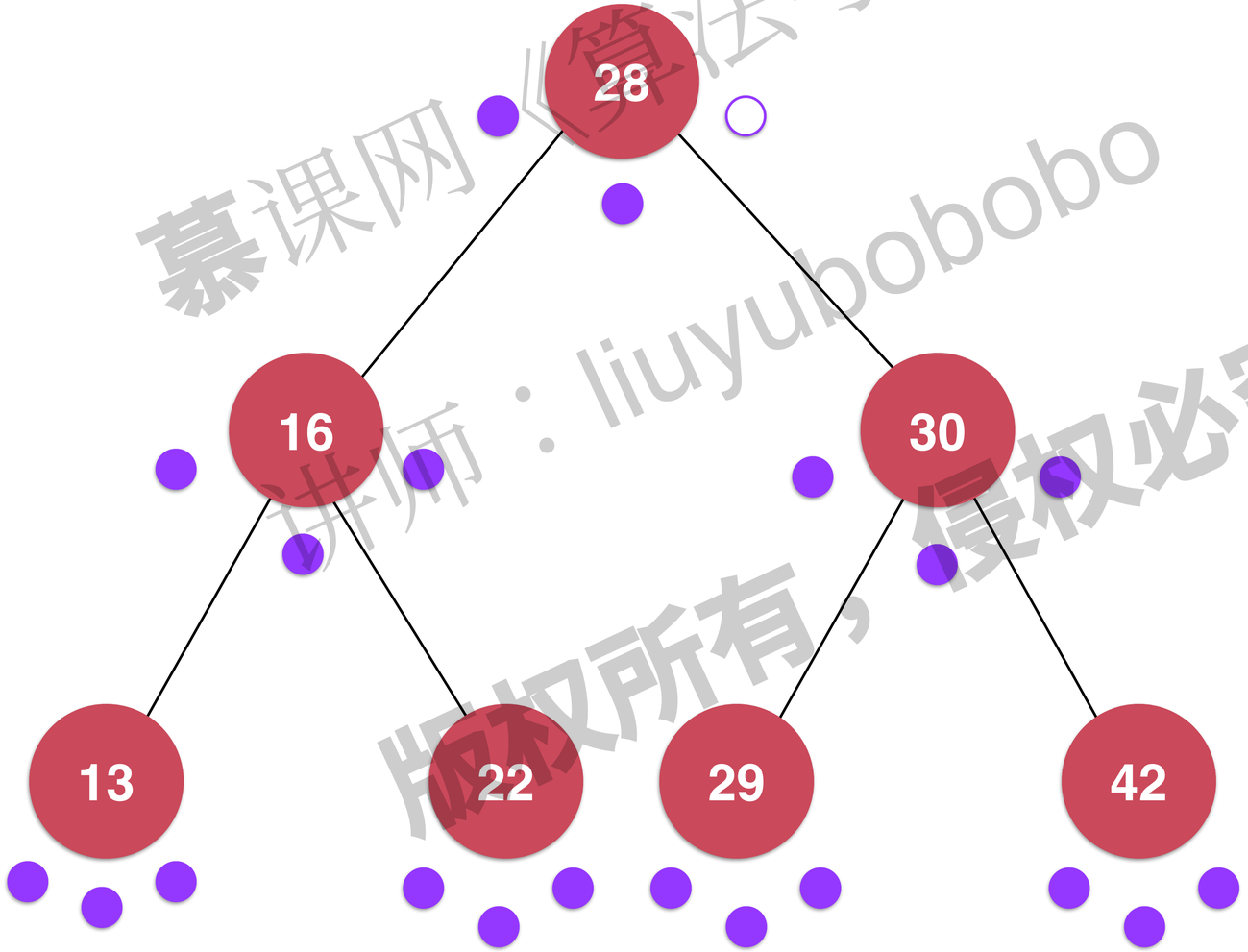
2 1 1 2 3 2 4

二分搜索树的前序遍历



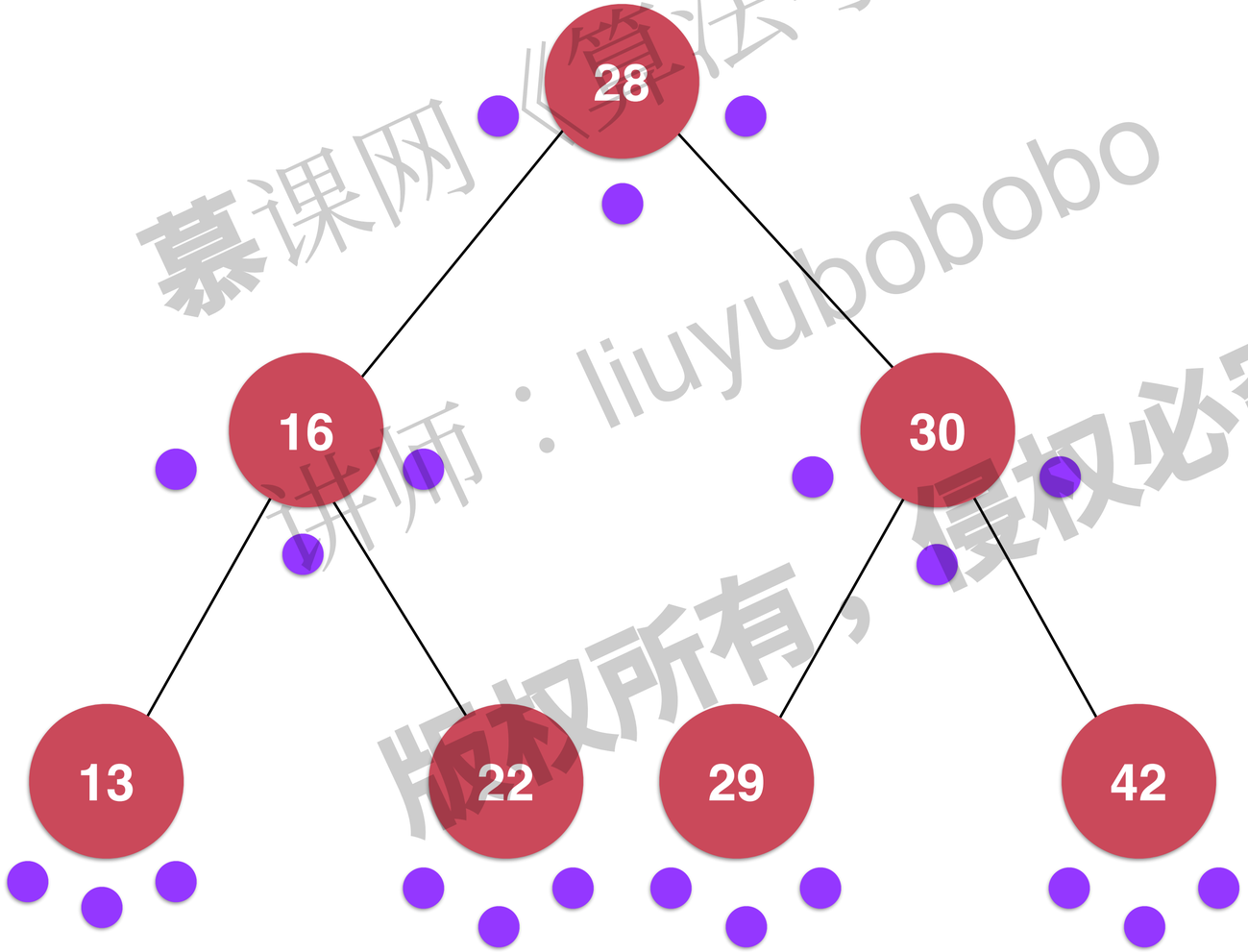
2 1 1 2 3 2 4

二分搜索树的前序遍历



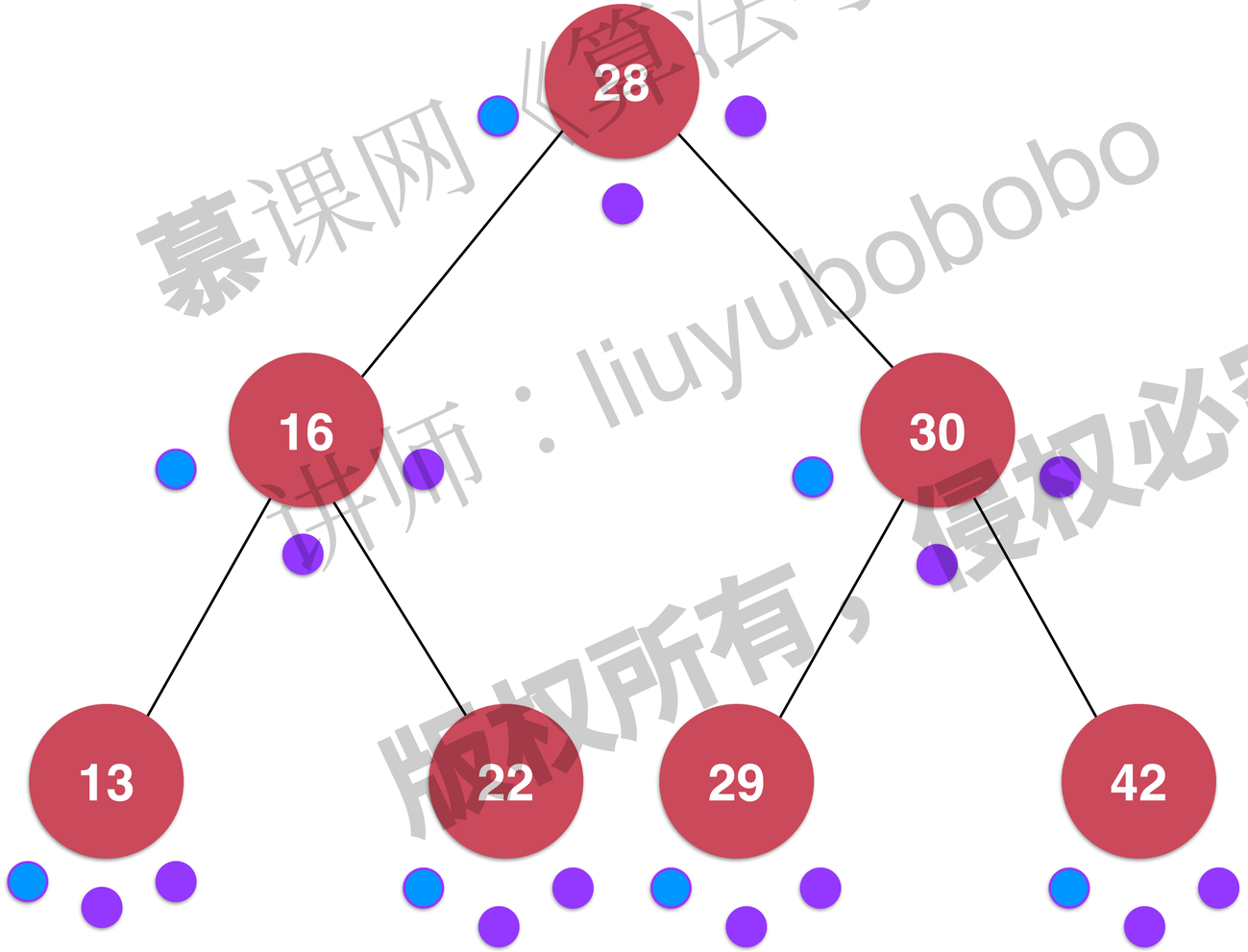
2
1
1
2
3
2
4

二分搜索树的前序遍历



2
1
1
2
3
2
4

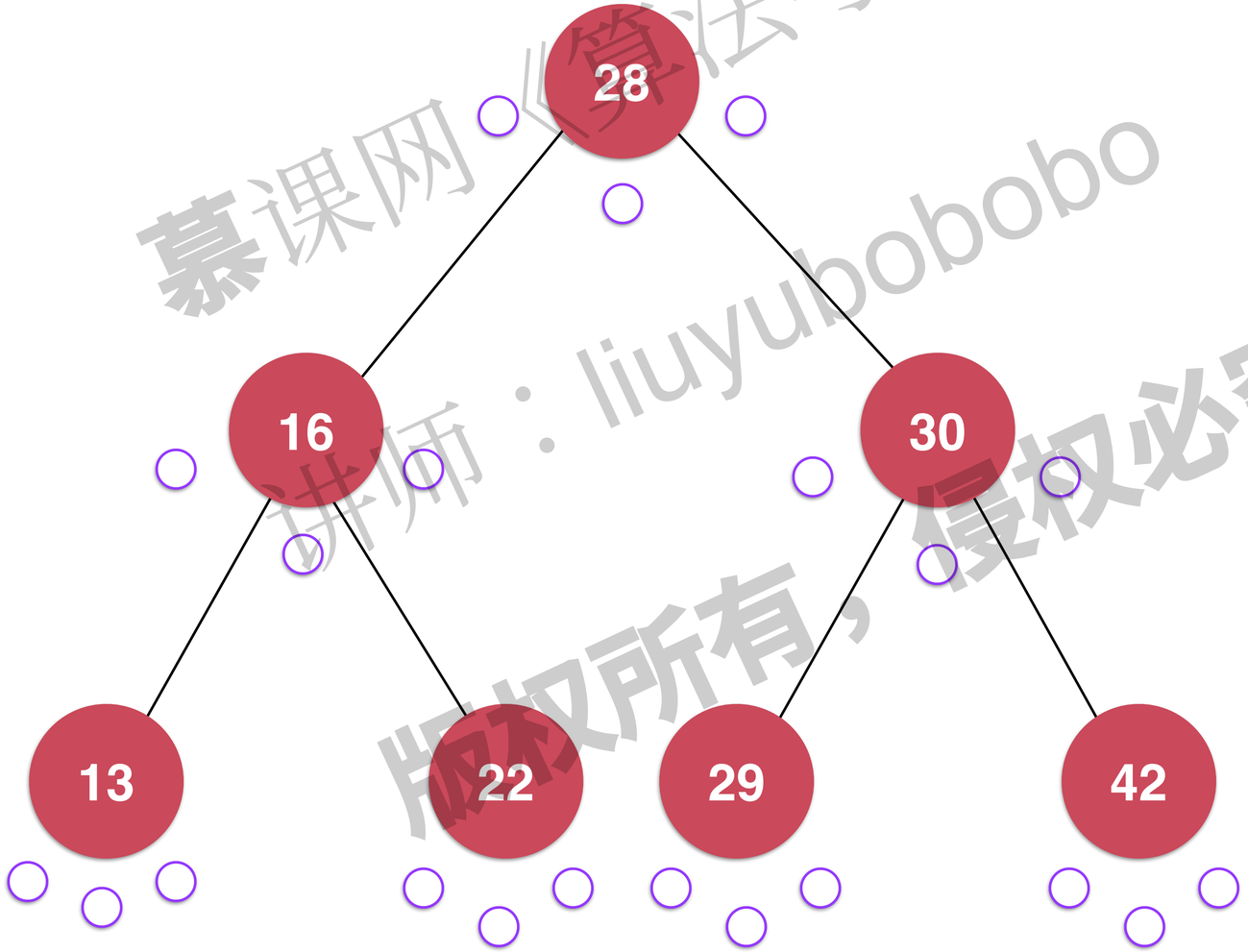
二分搜索树的前序遍历



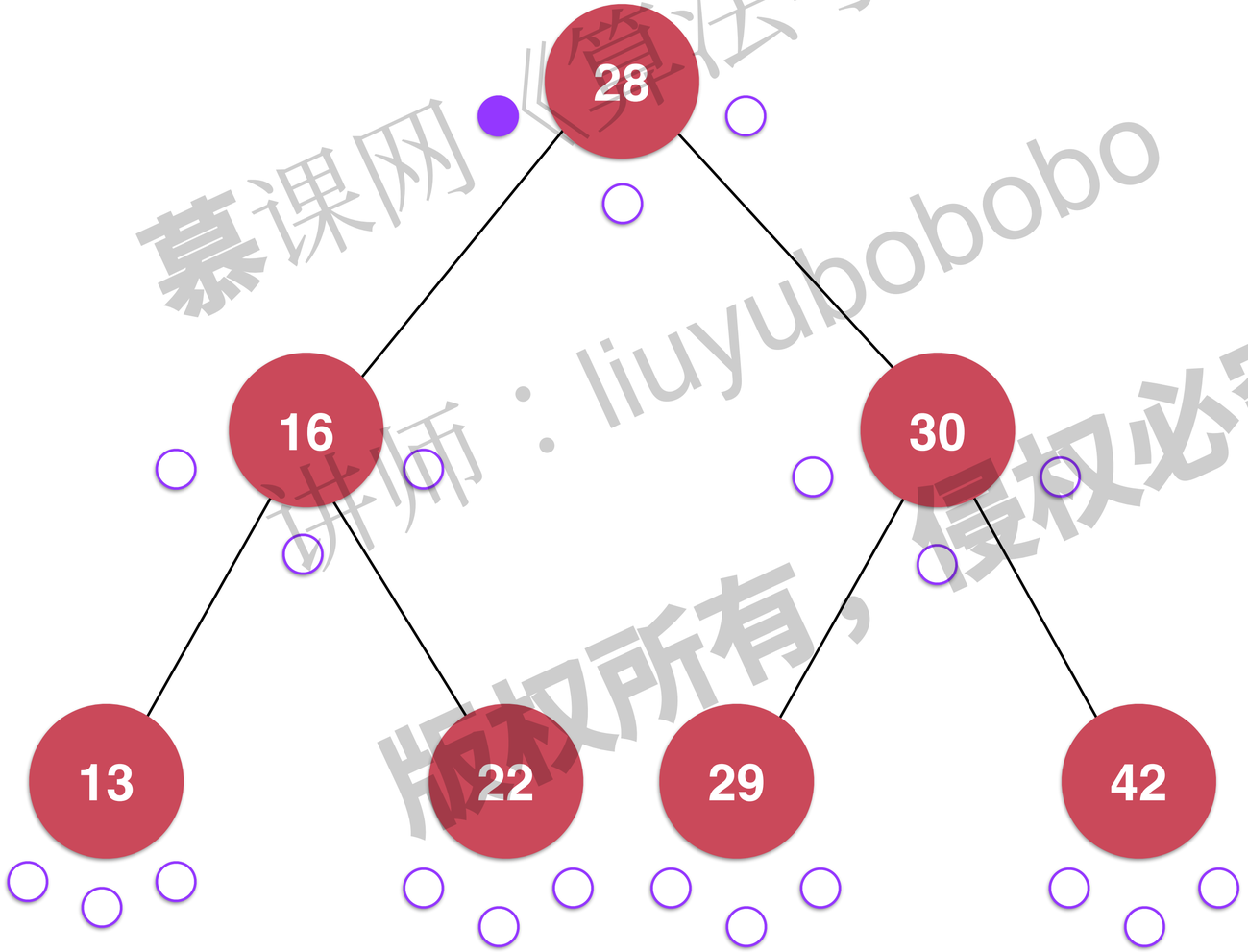
2 1 1 2 3 2 4

中序遍历

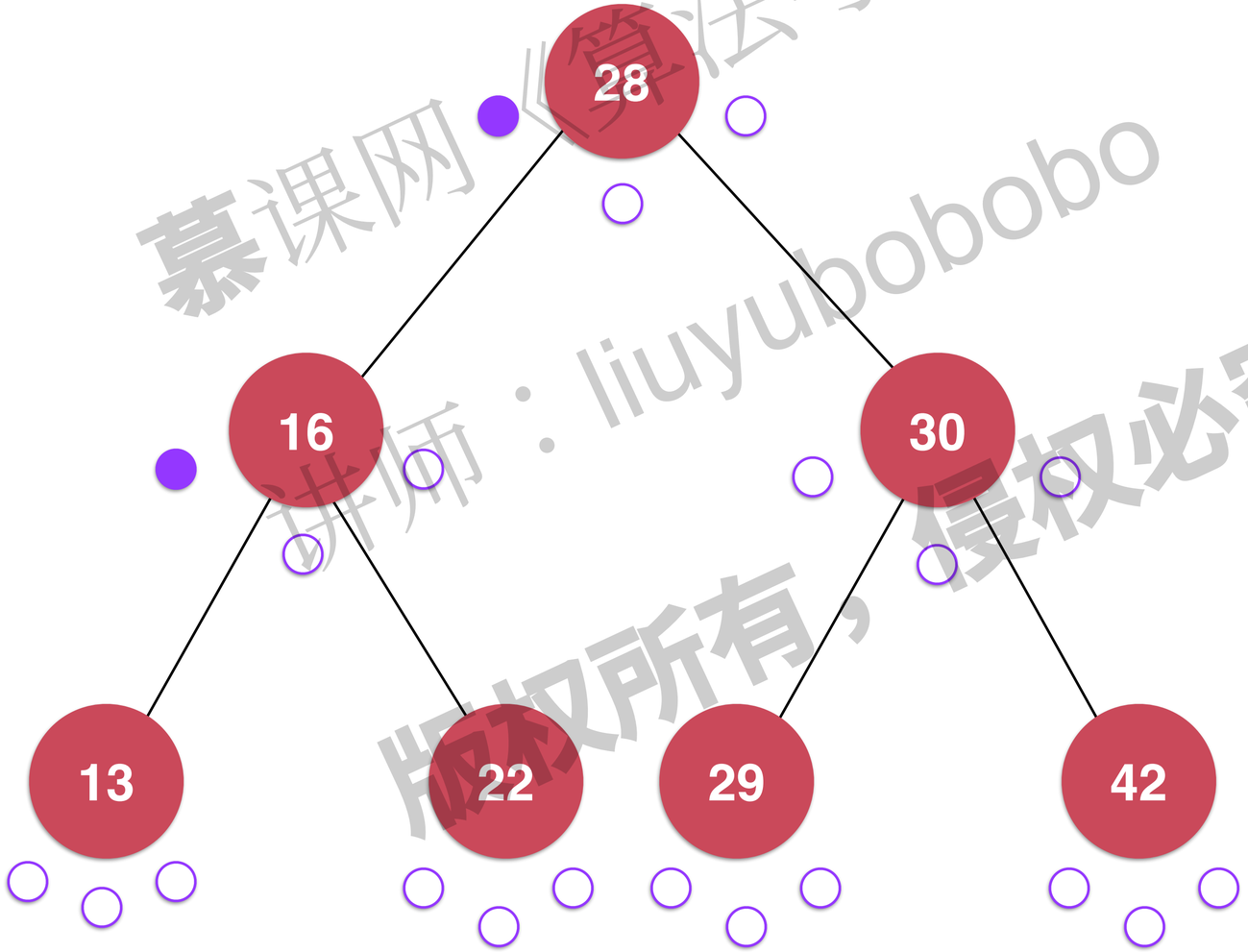
二分搜索树的中序遍历



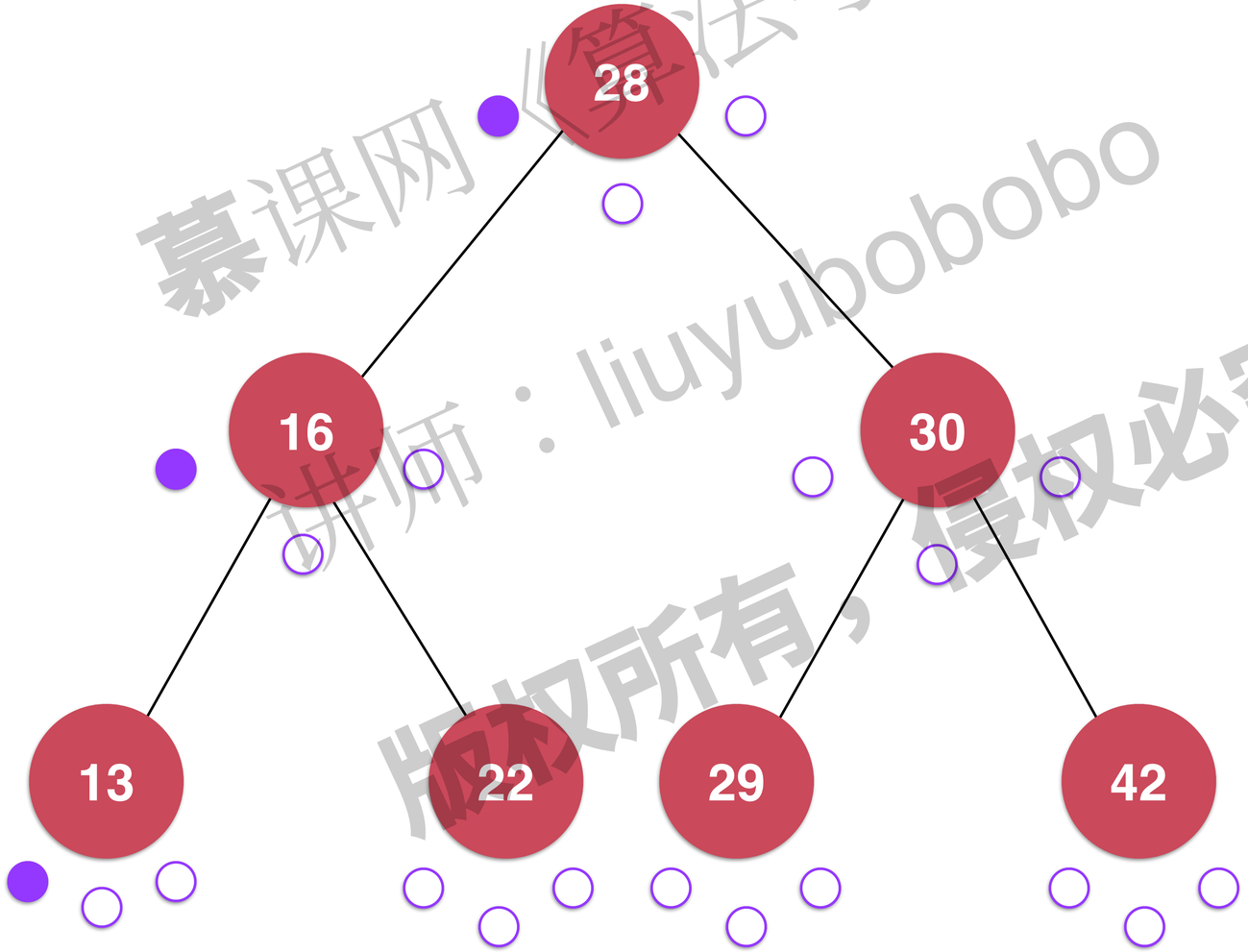
二分搜索树的中序遍历



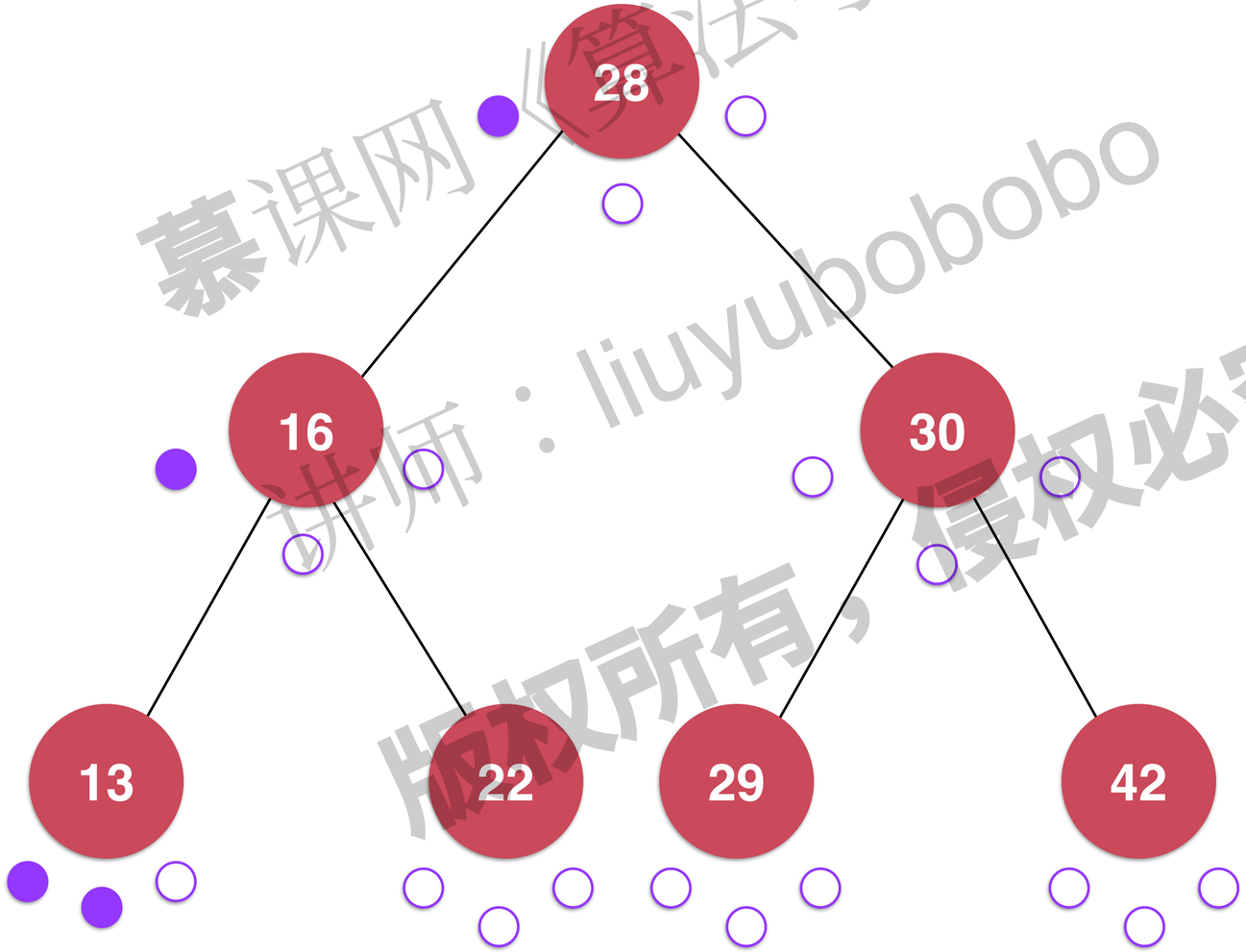
二分搜索树的中序遍历



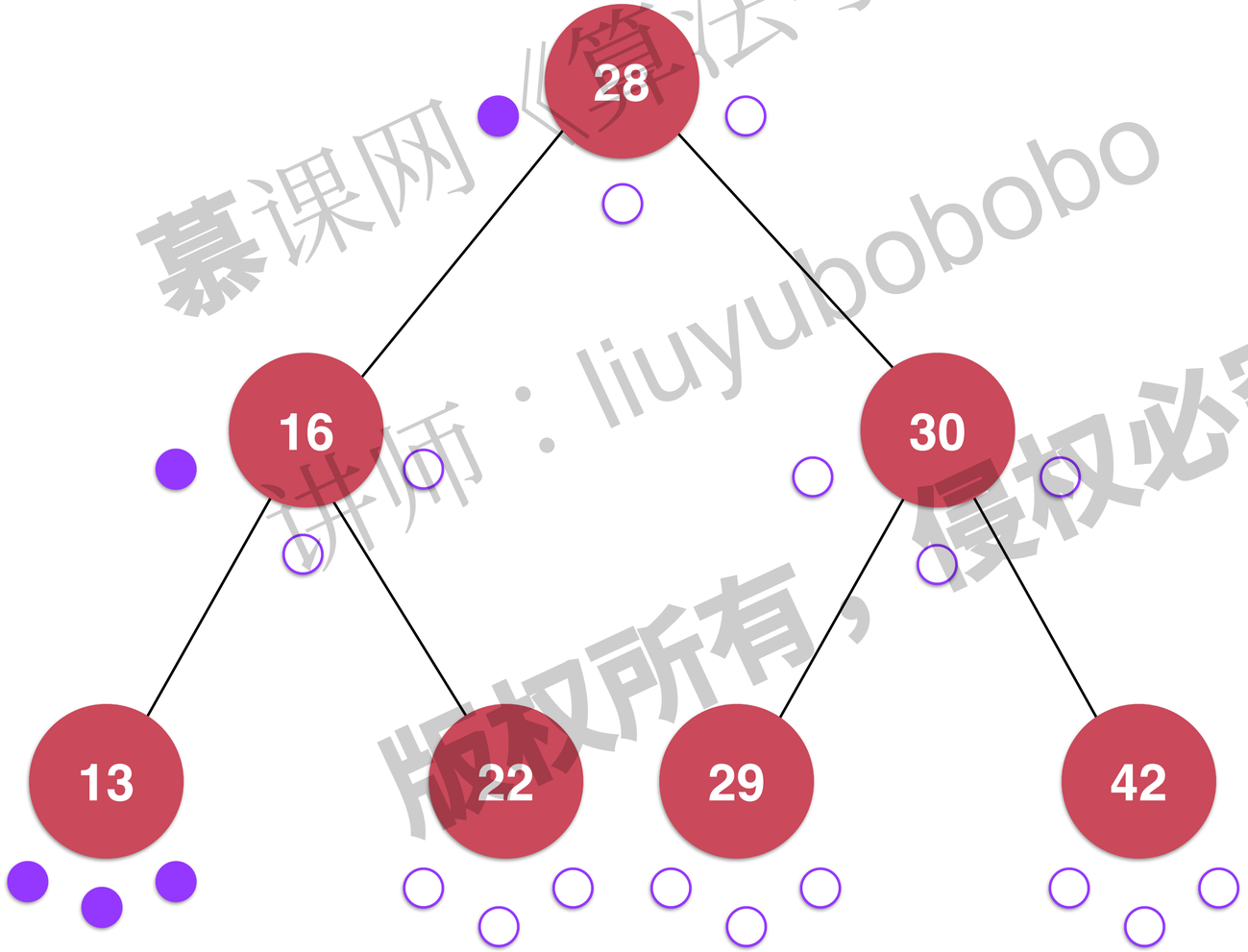
二分搜索树的中序遍历



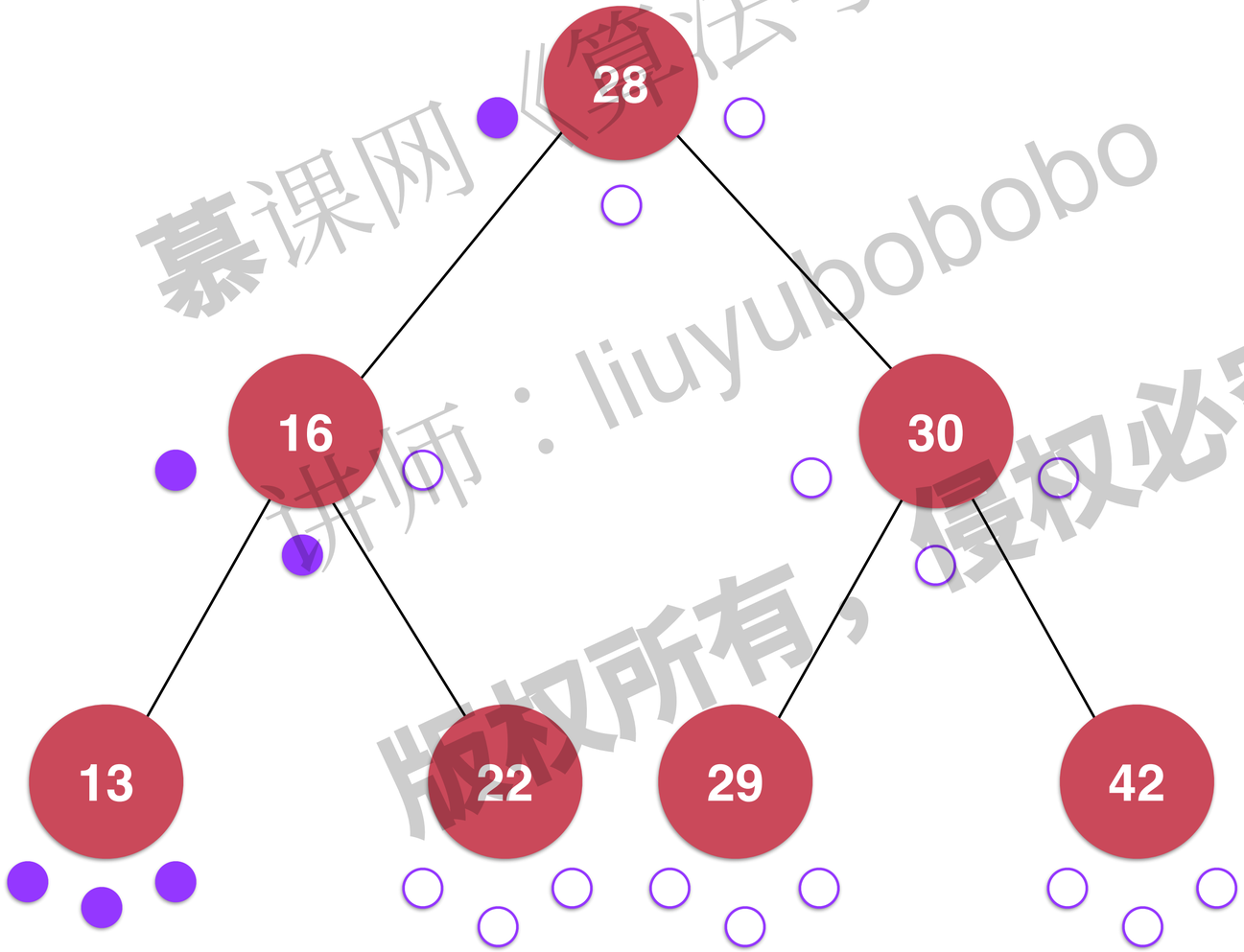
二分搜索树的中序遍历



二分搜索树的中序遍历



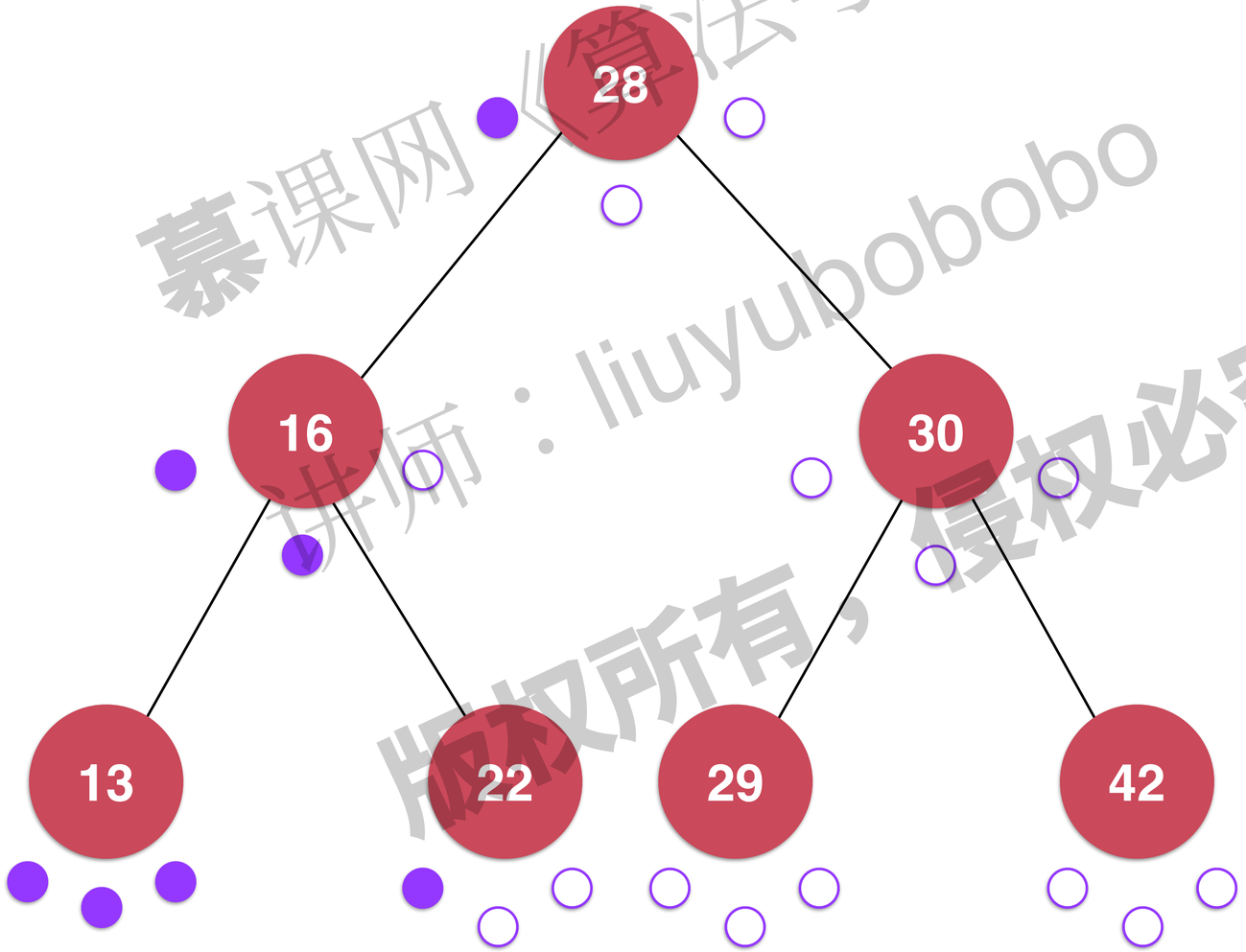
二分搜索树的中序遍历



1

1

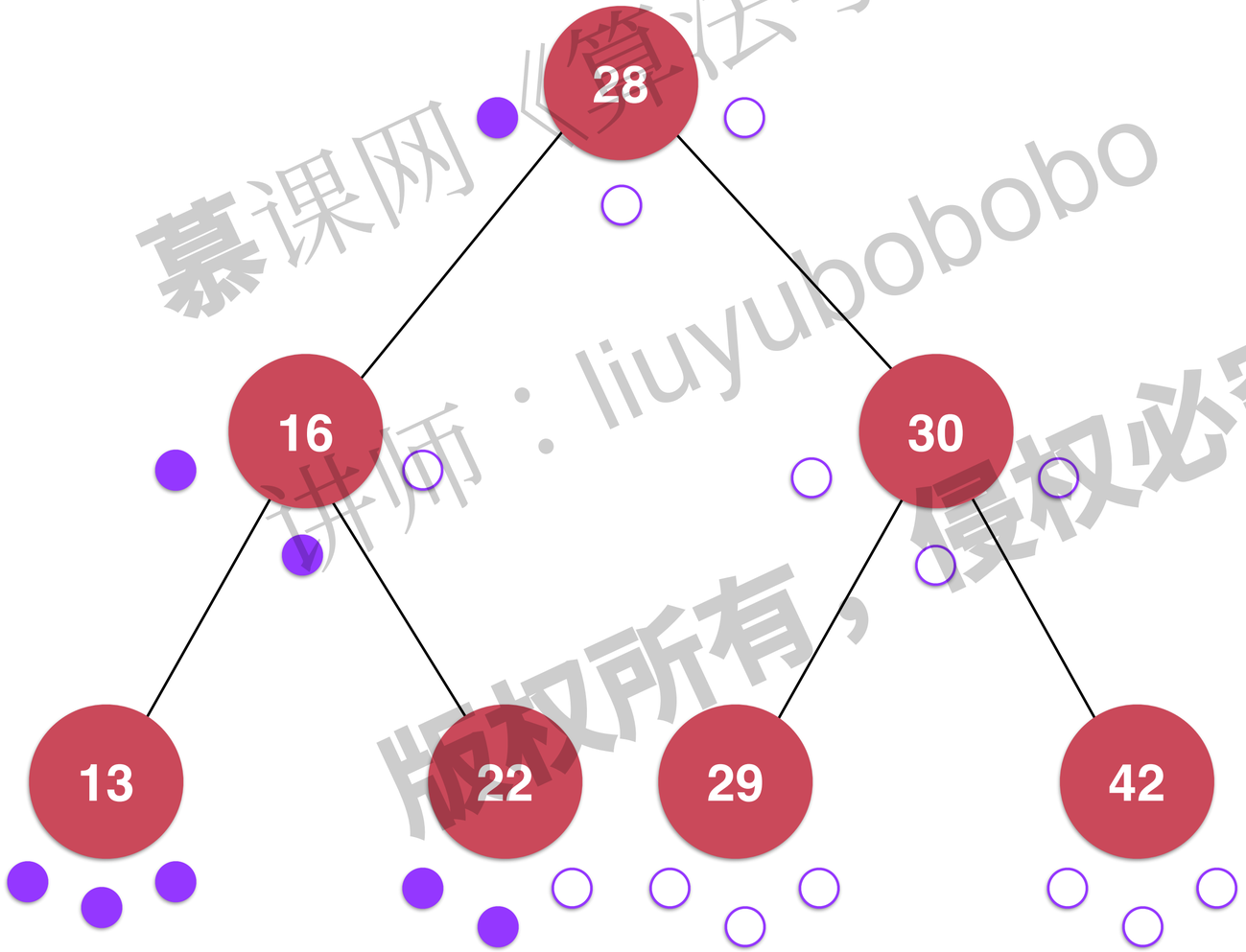
二分搜索树的中序遍历



1

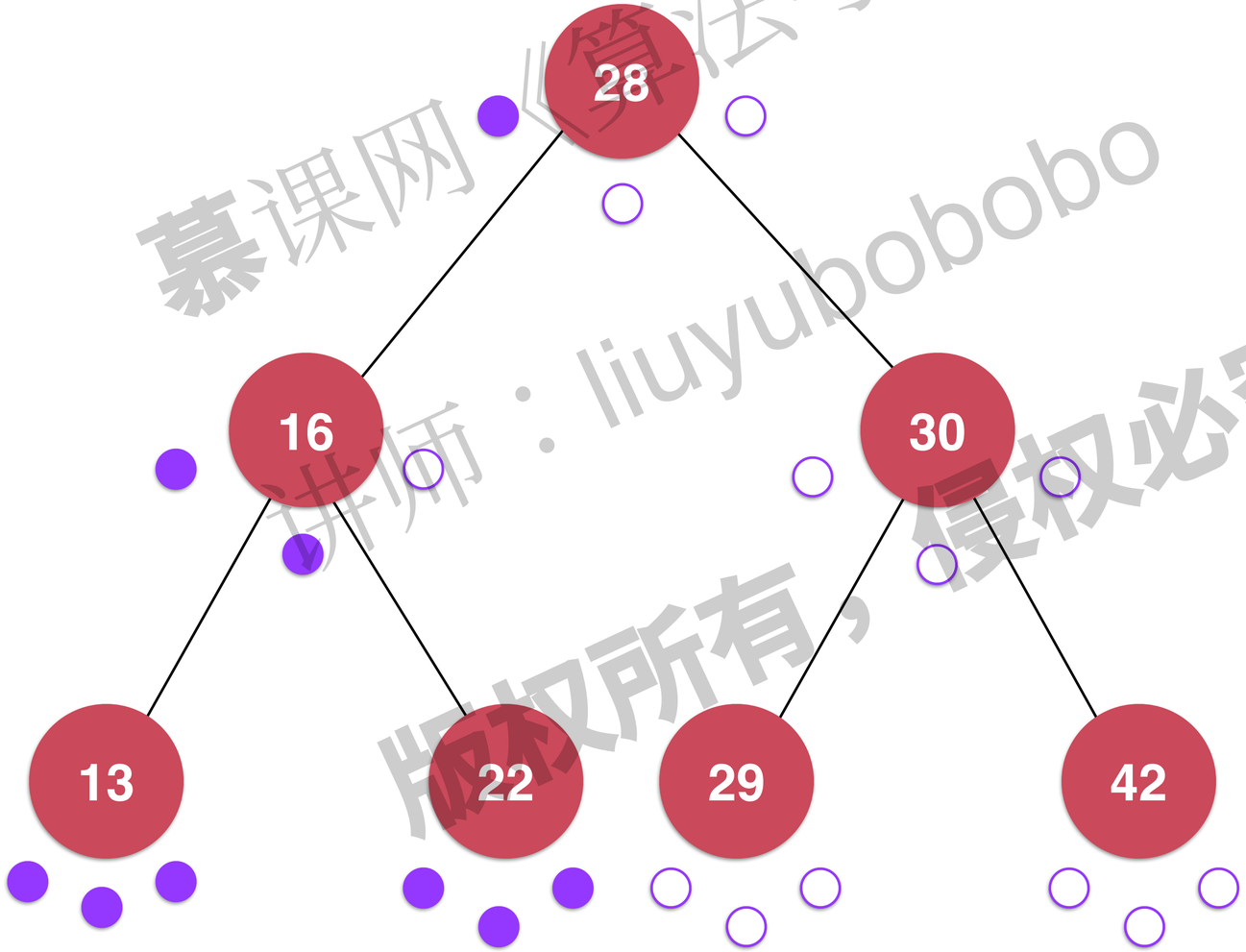
1

二分搜索树的中序遍历



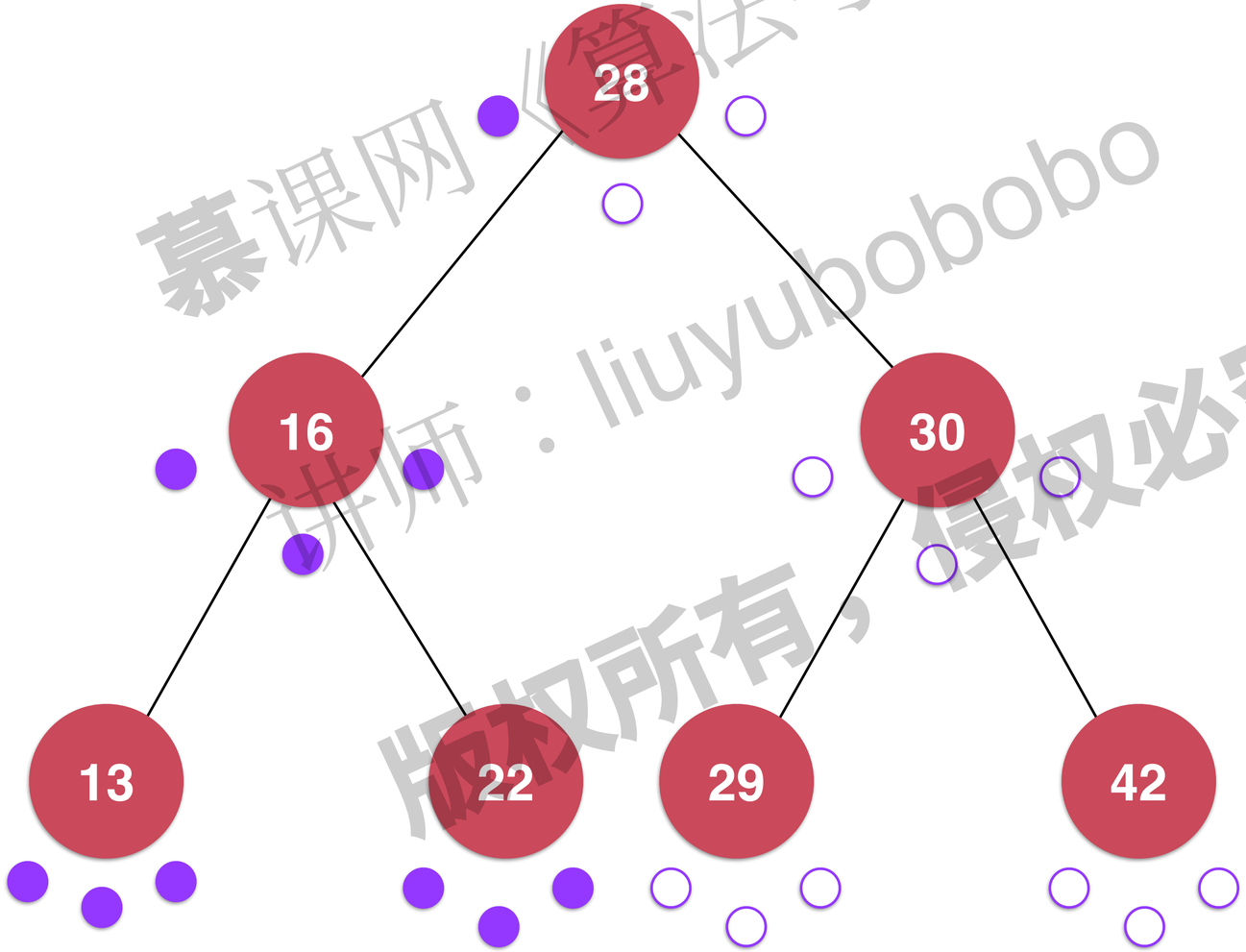
1
1
2

二分搜索树的中序遍历



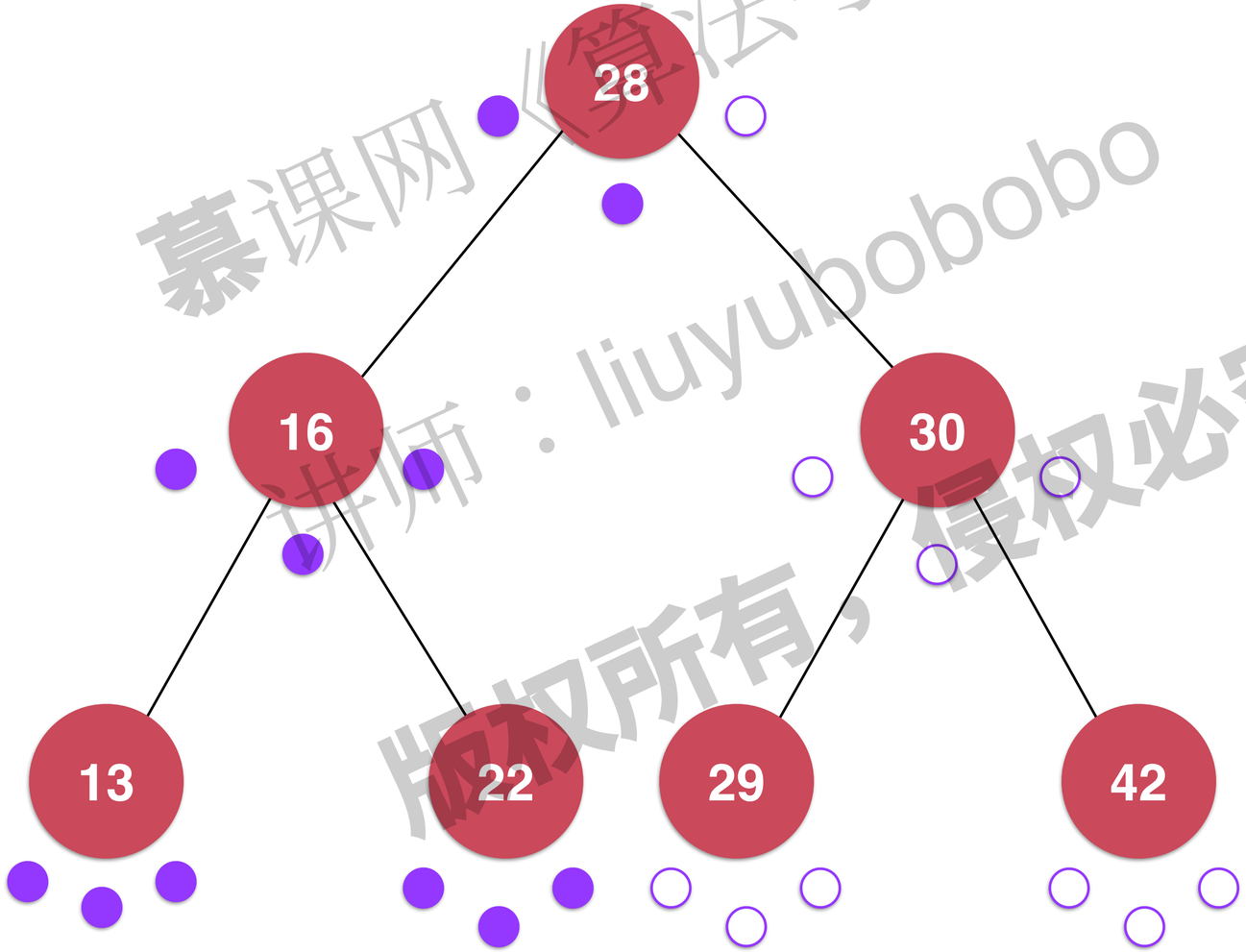
1
1
2

二分搜索树的中序遍历



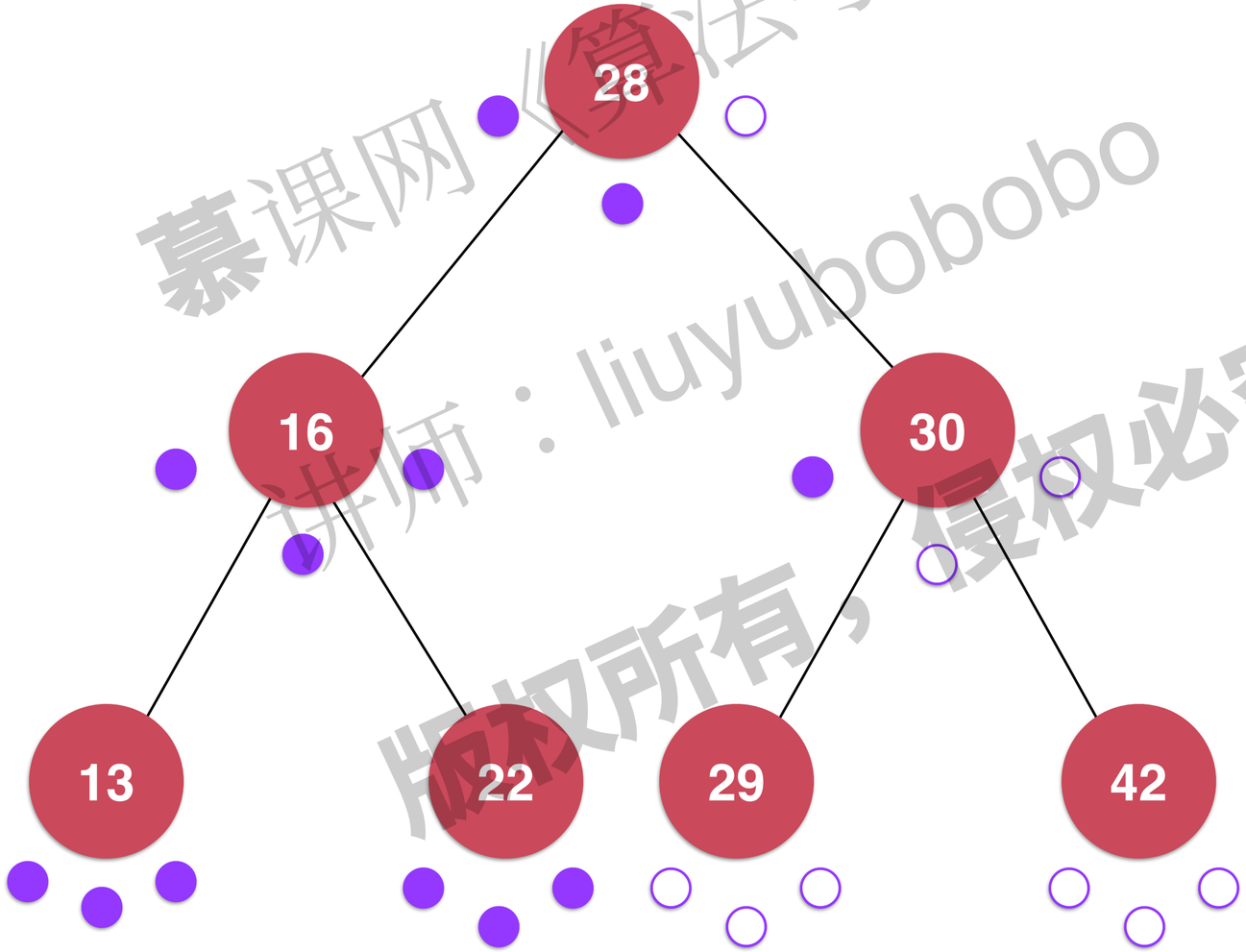
1
1
2

二分搜索树的中序遍历



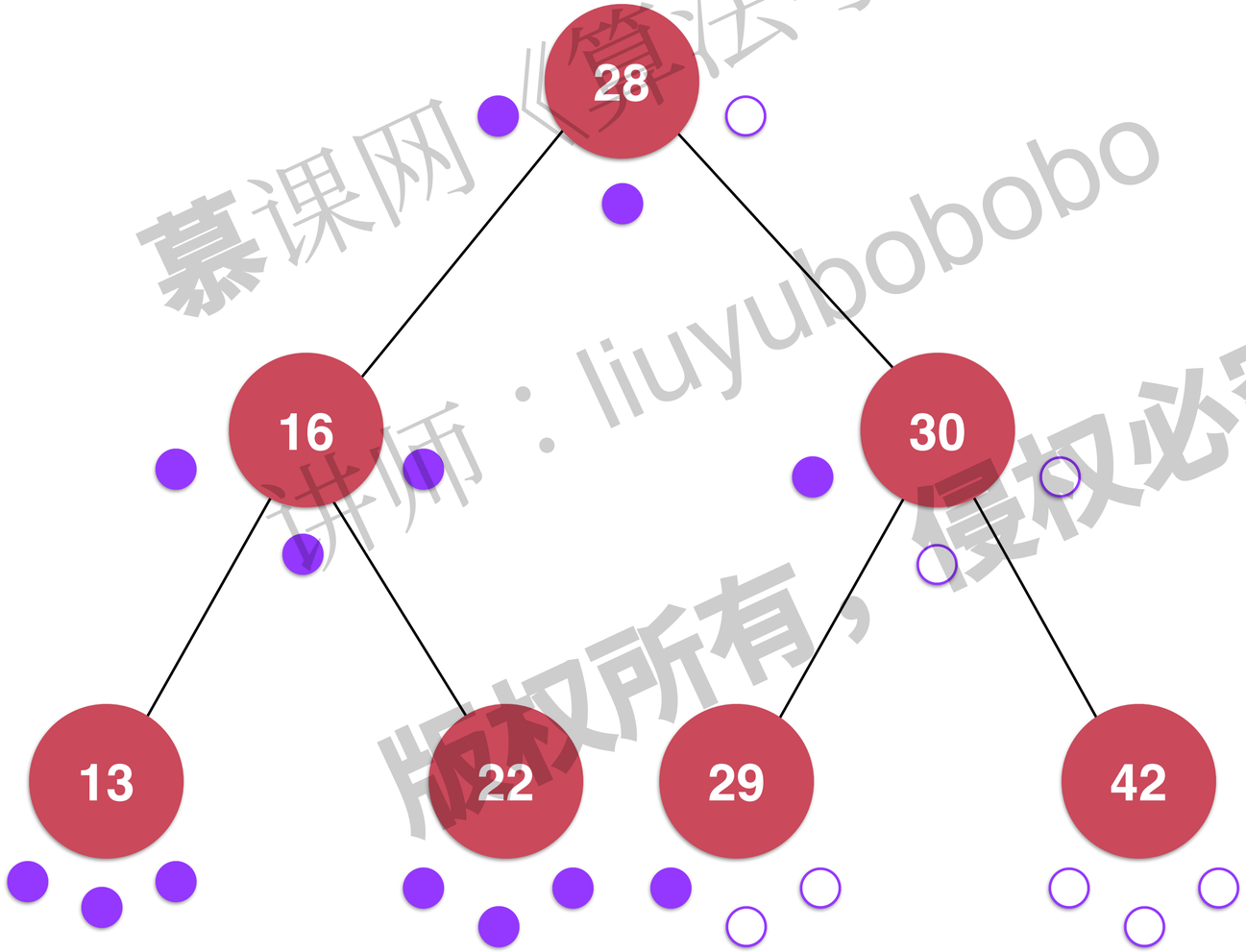
1
1
2
2

二分搜索树的中序遍历



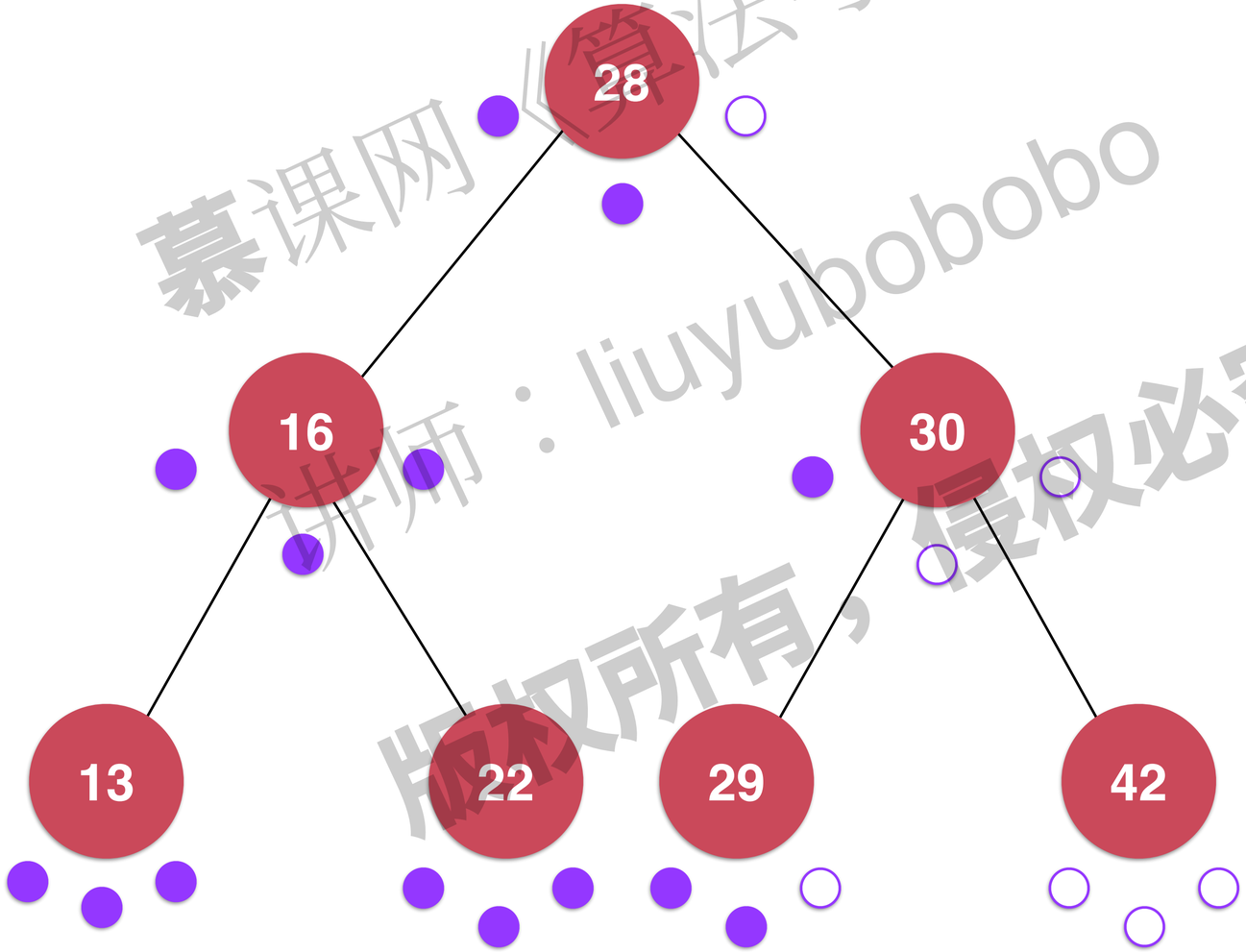
1
1
2
2

二分搜索树的中序遍历



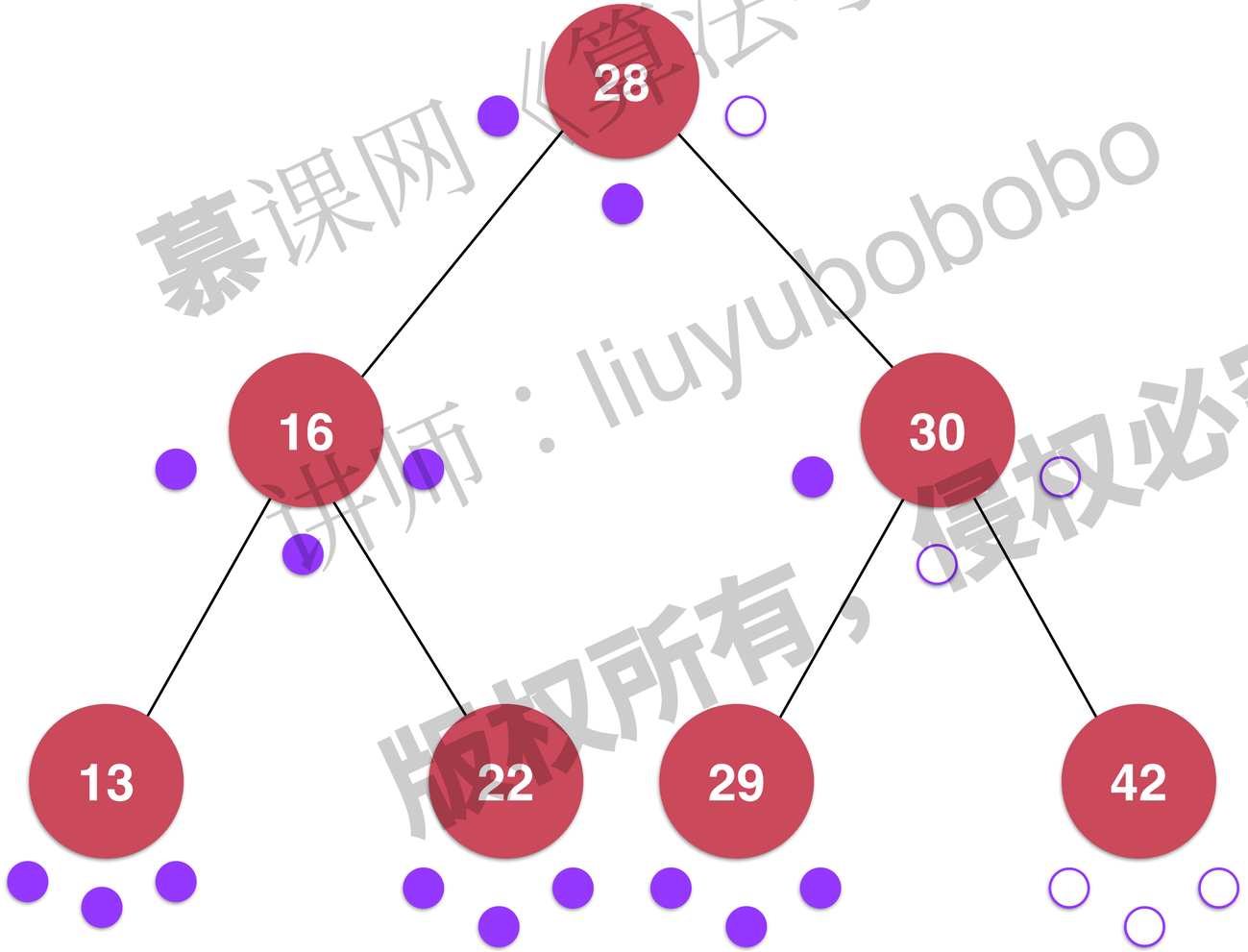
1
1
2
2

二分搜索树的中序遍历



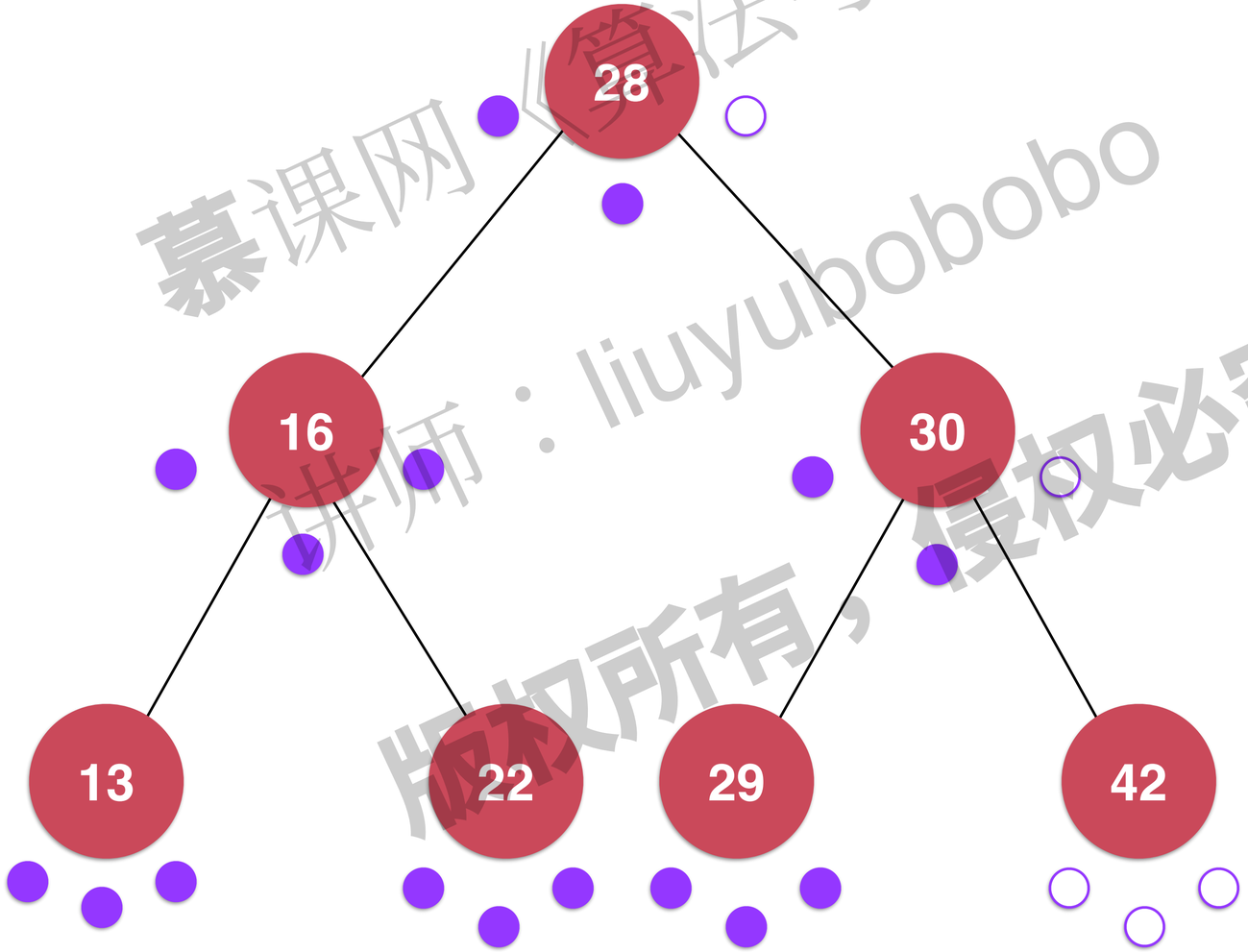
1
1
2
2
2

二分搜索树的中序遍历



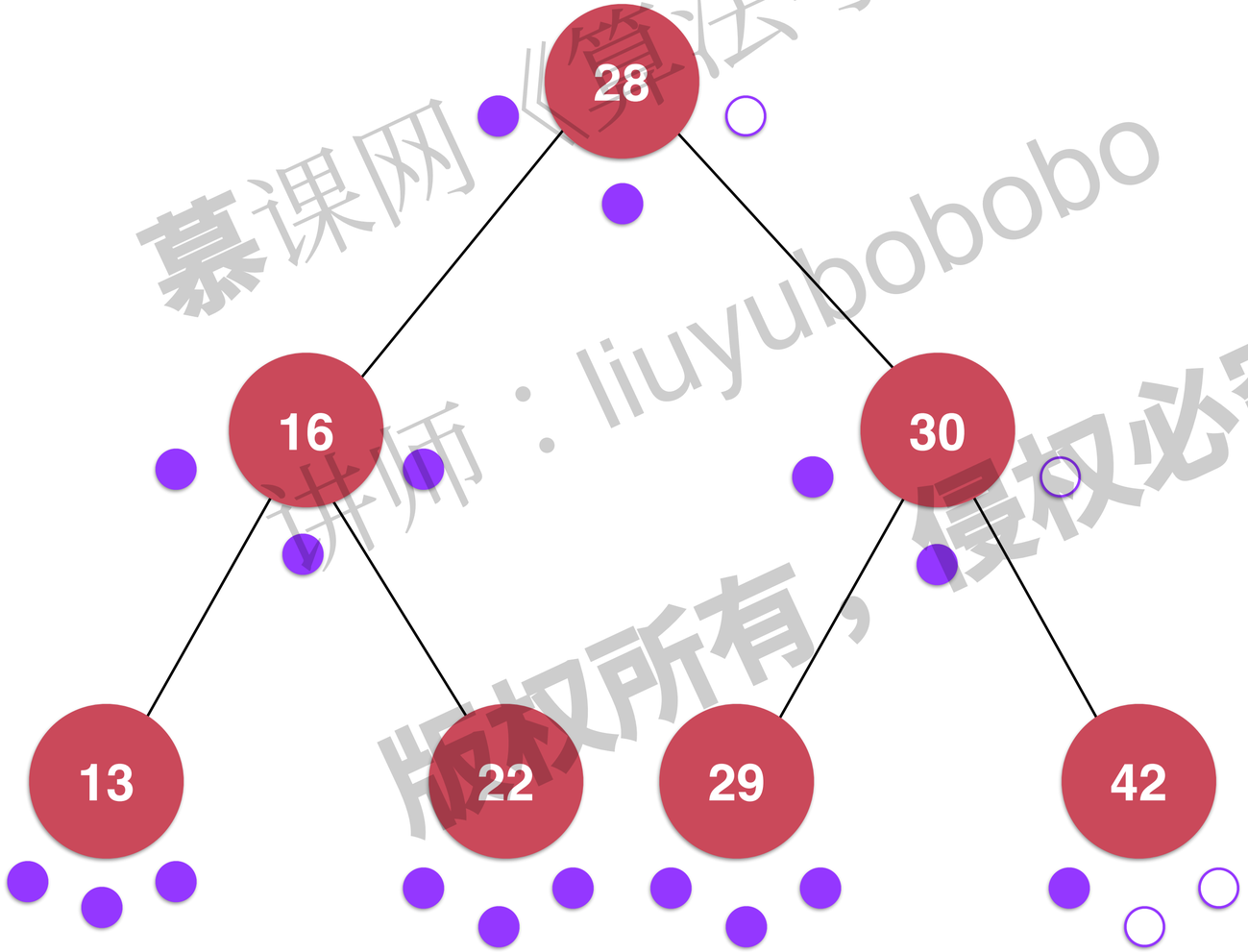
1
1
2
2
2

二分搜索树的中序遍历



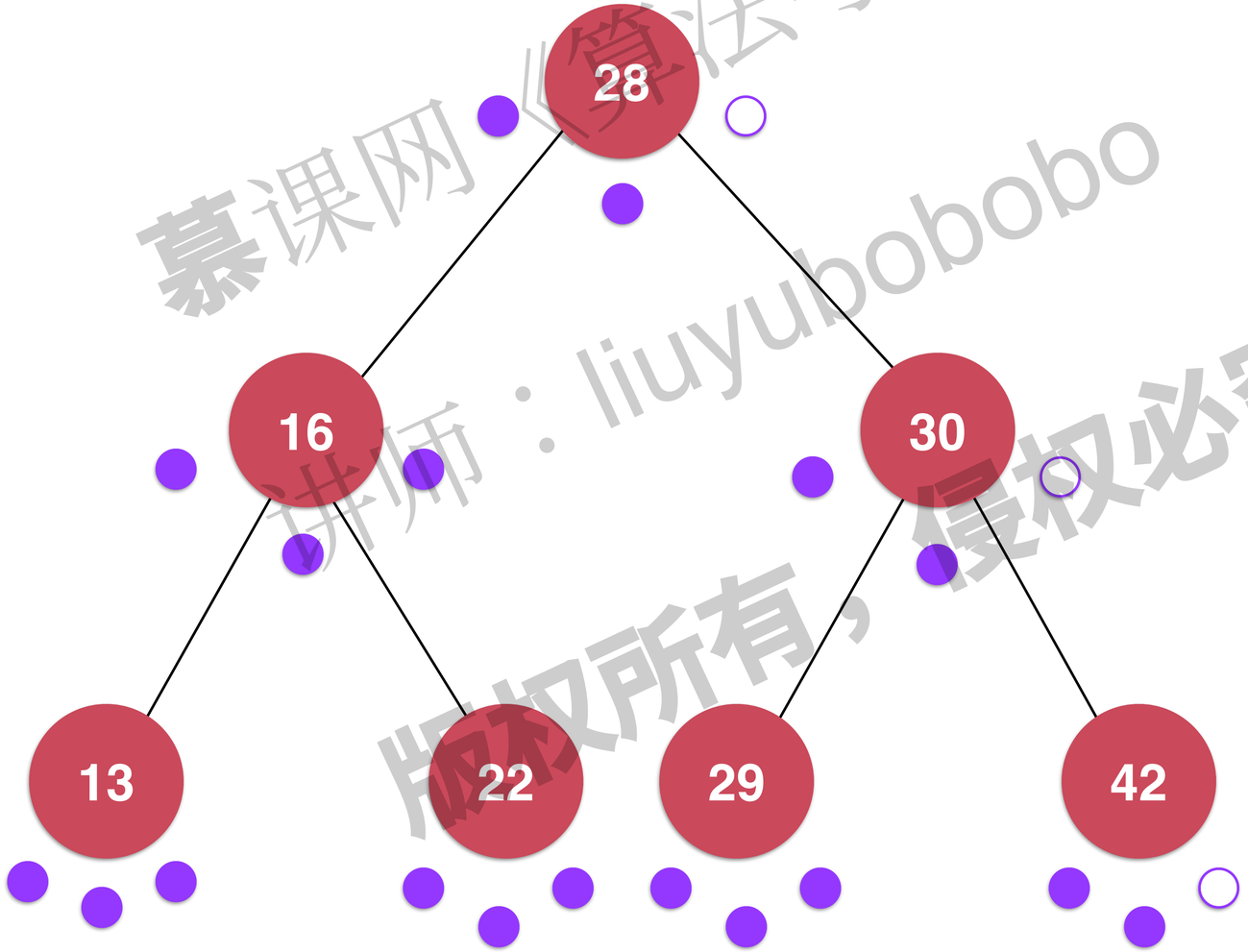
1
1
2
2
2
3

二分搜索树的中序遍历



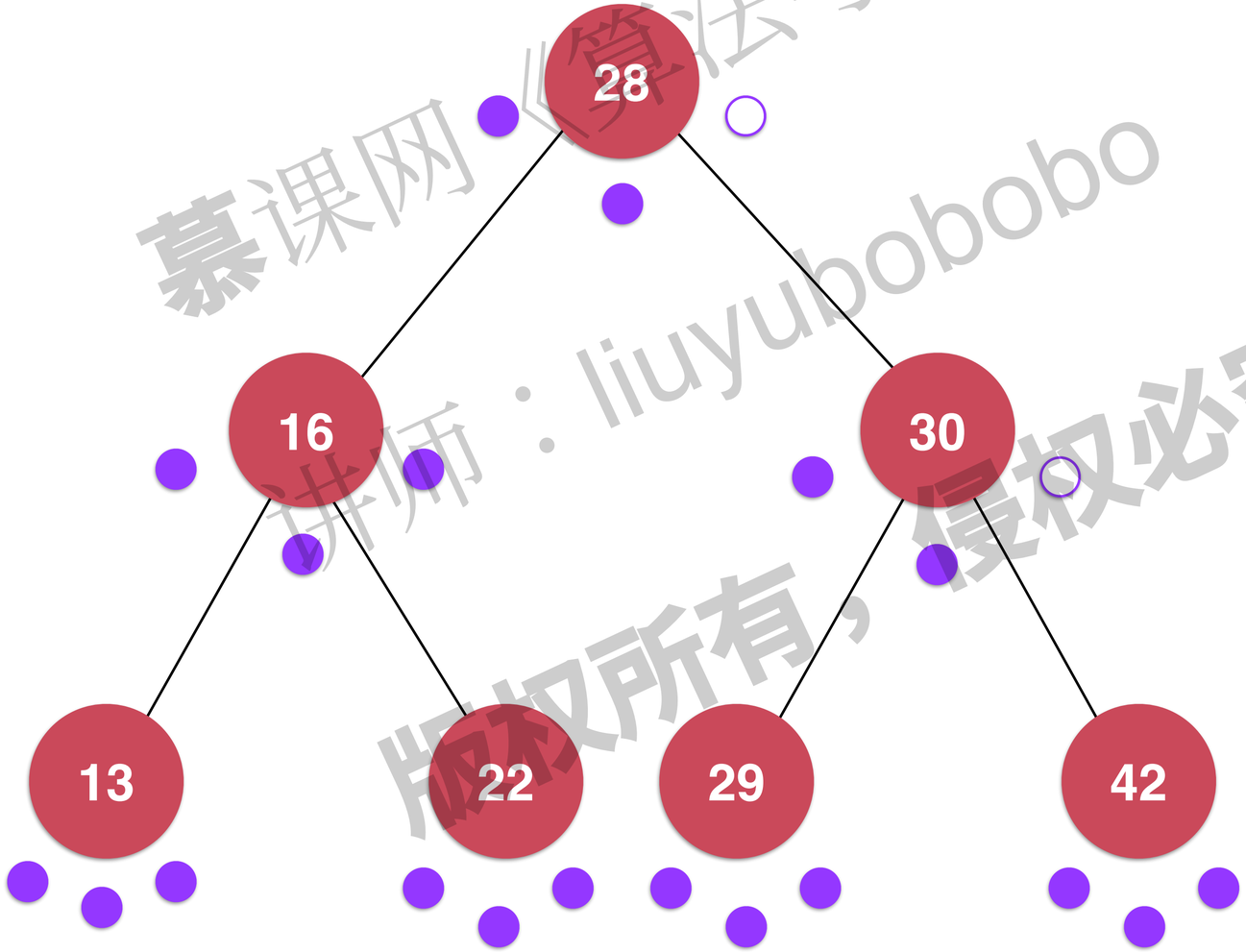
1
1
2
2
2
3

二分搜索树的中序遍历



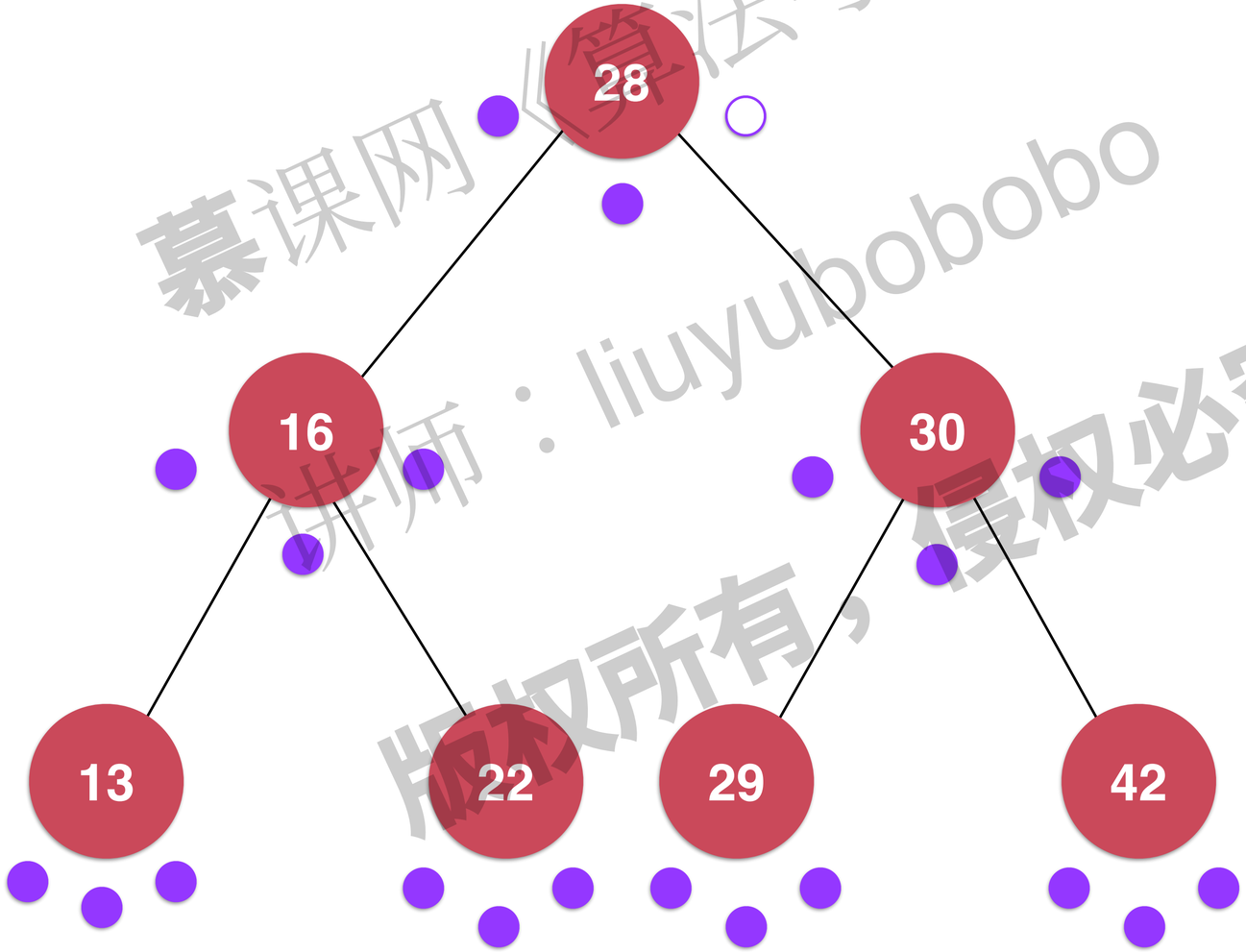
1
1
2
2
2
3
4

二分搜索树的中序遍历



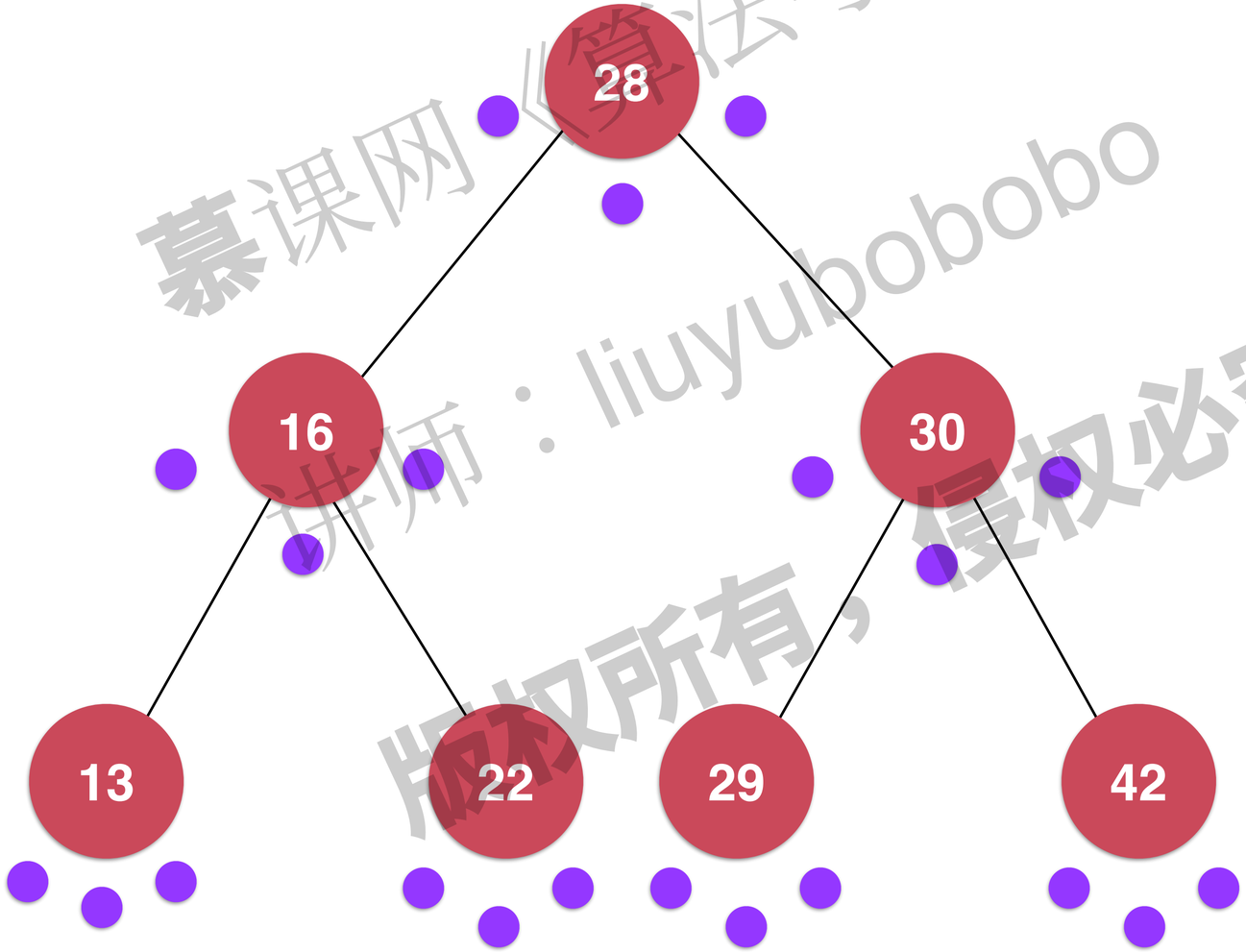
1
1
2
2
2
3
4

二分搜索树的中序遍历

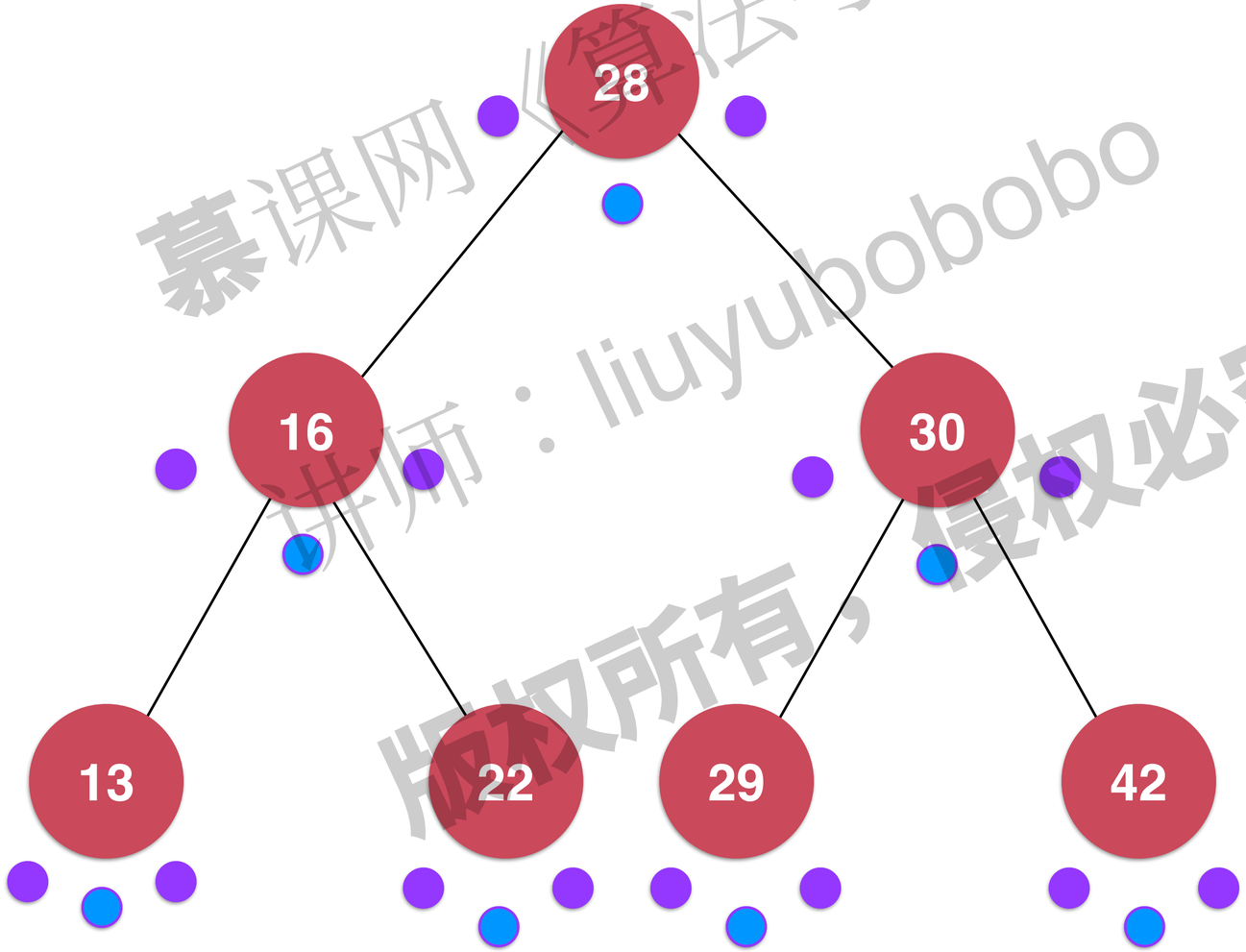


1
1
2
2
2
3
4

二分搜索树的中序遍历



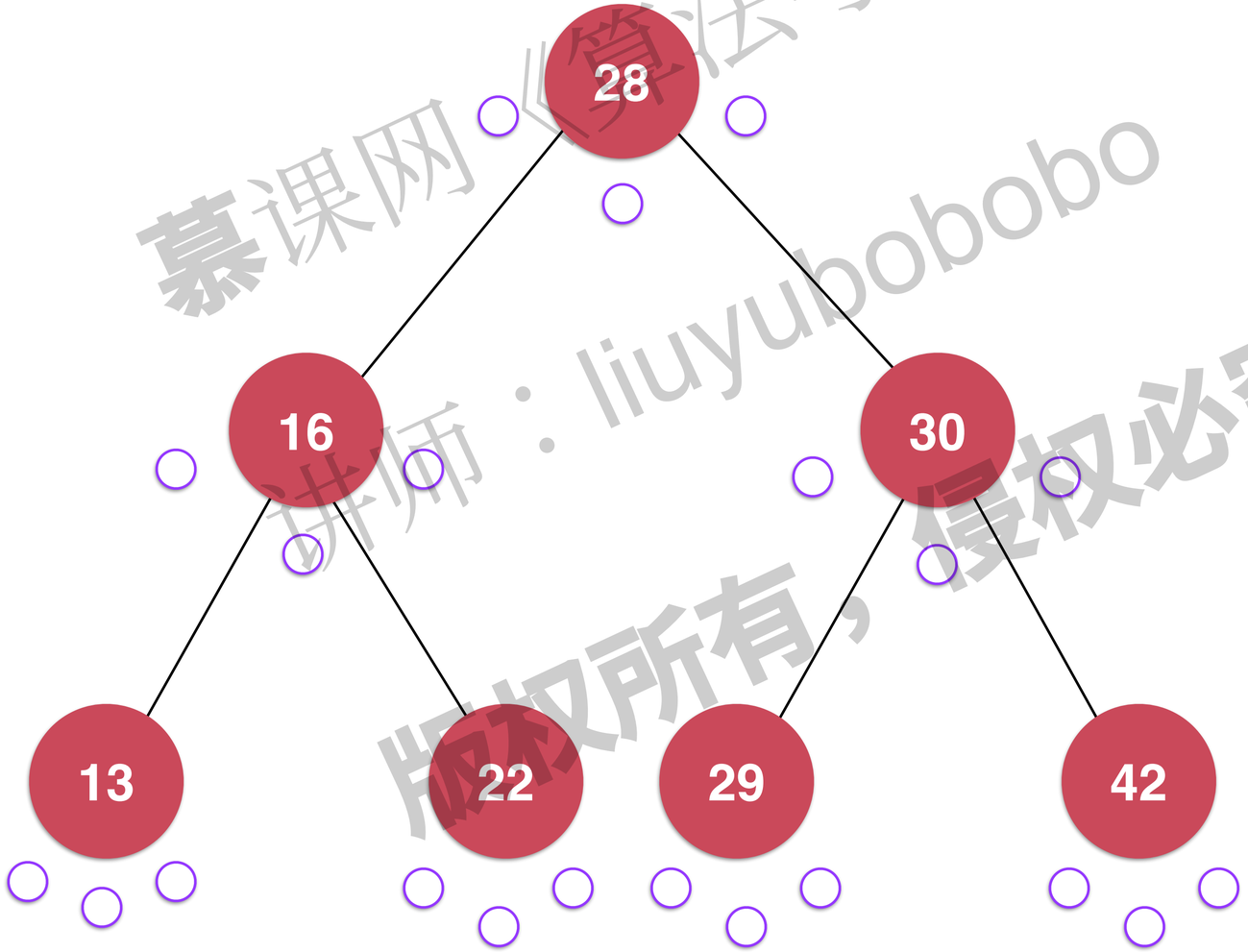
二分搜索树的中序遍历



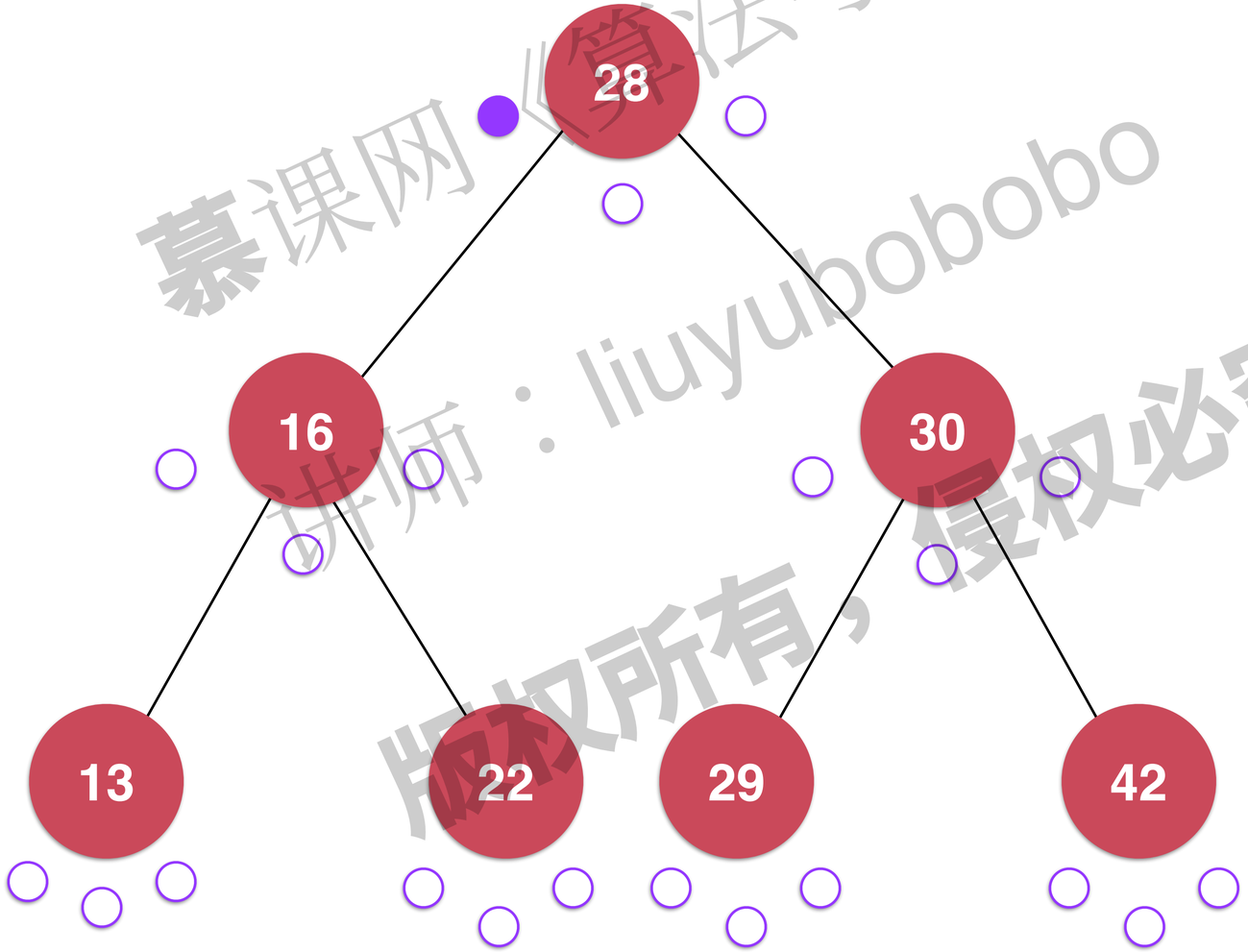
1
1
2
2
2
3
4

后序遍历

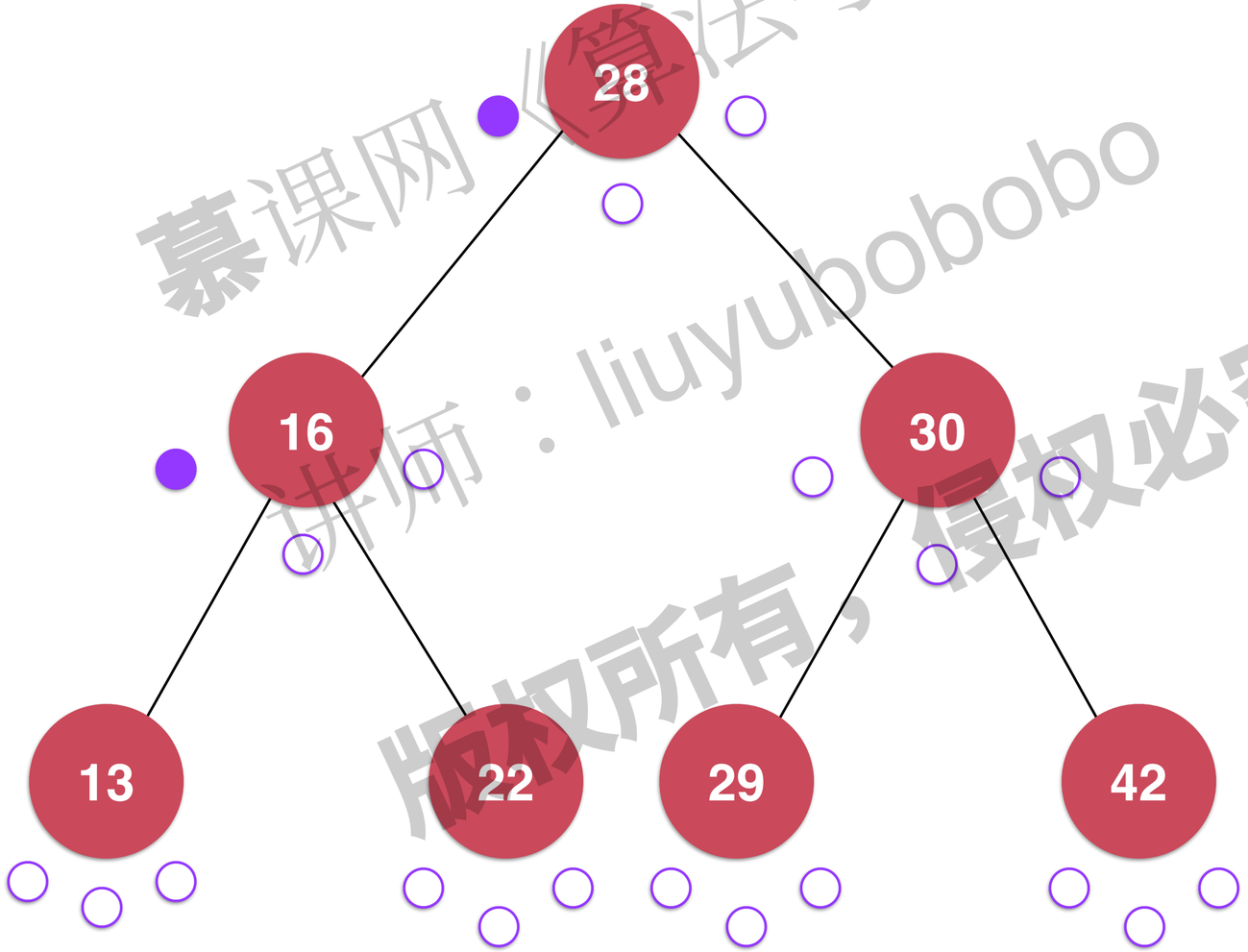
二分搜索树的后序遍历



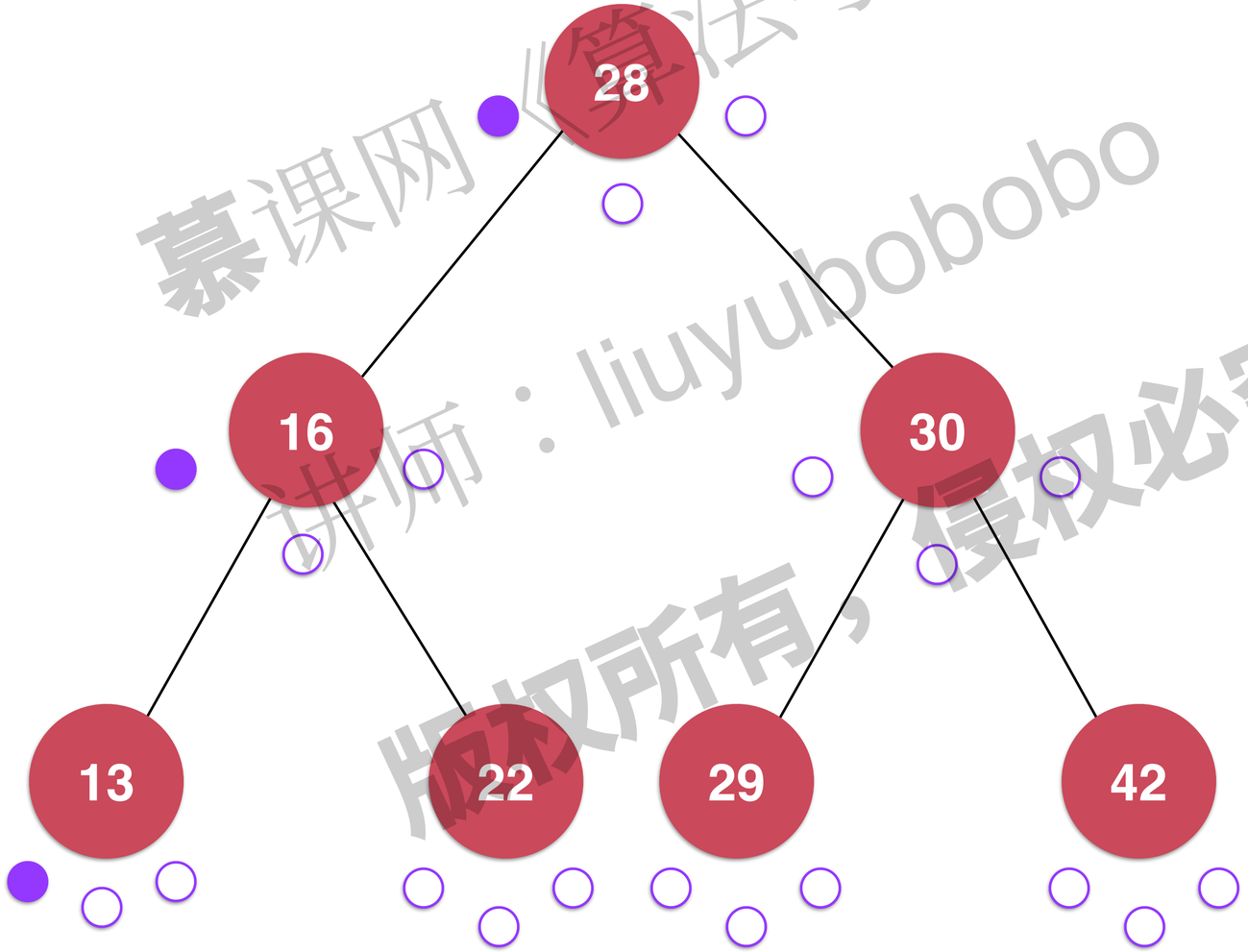
二分搜索树的后序遍历



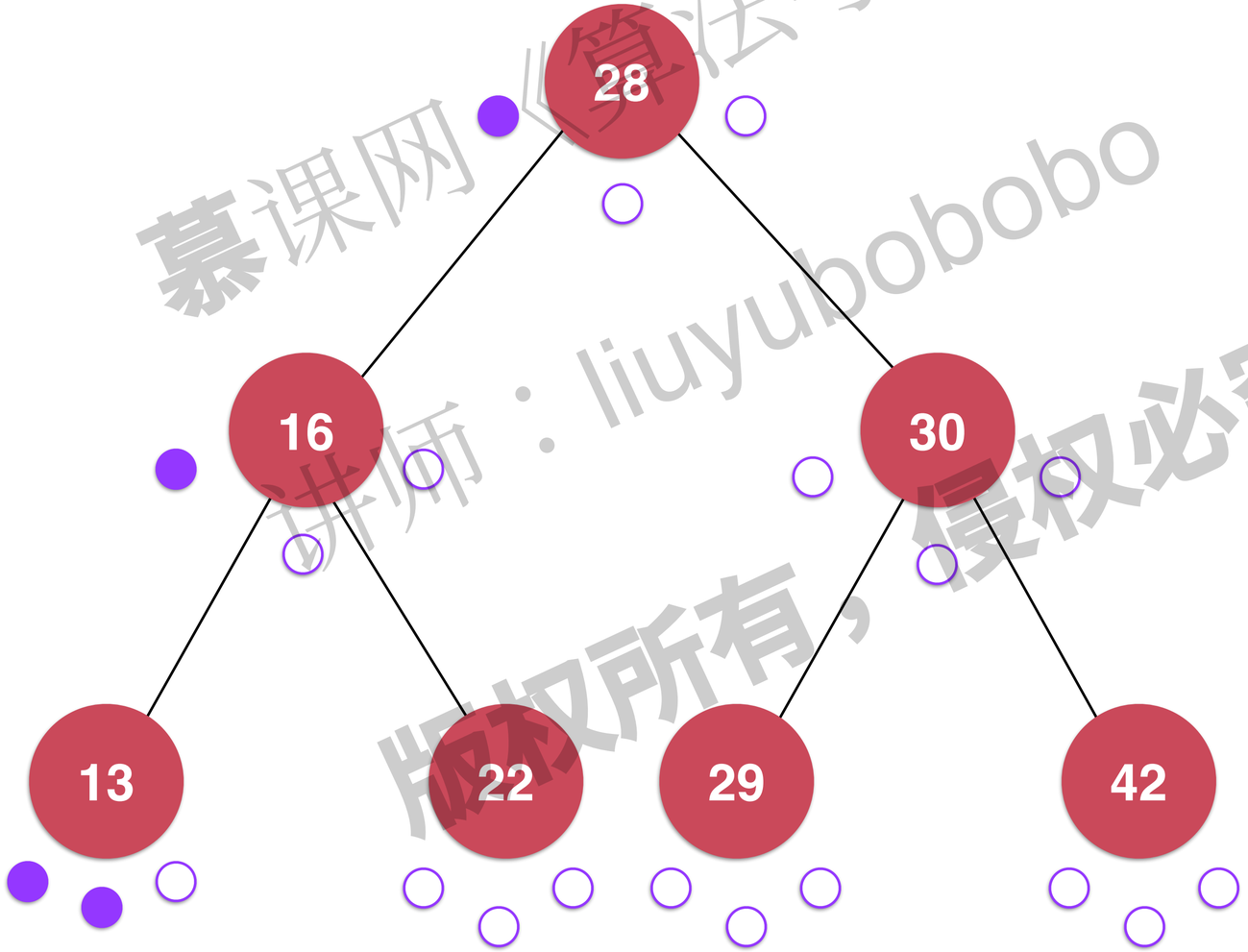
二分搜索树的后序遍历



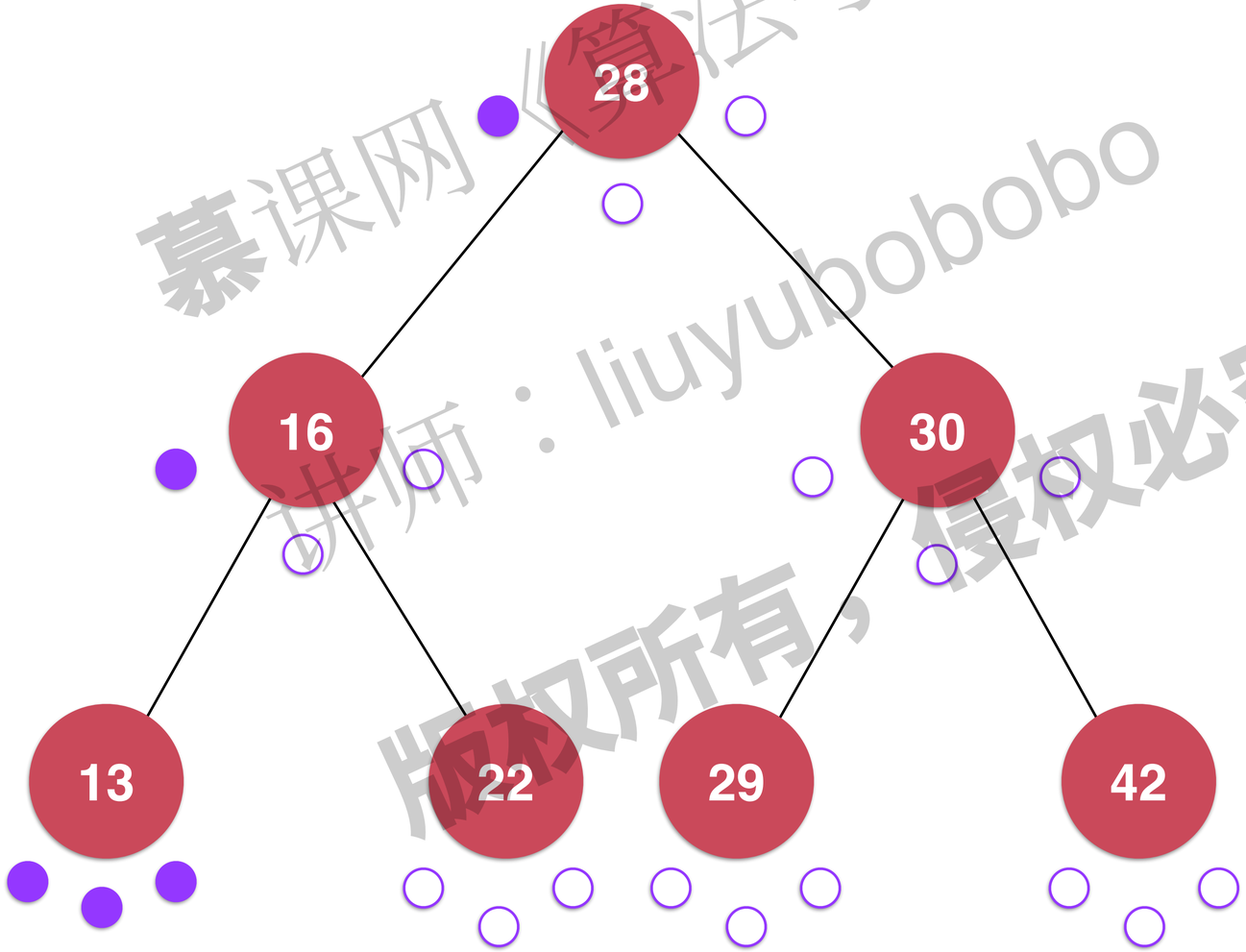
二分搜索树的后序遍历



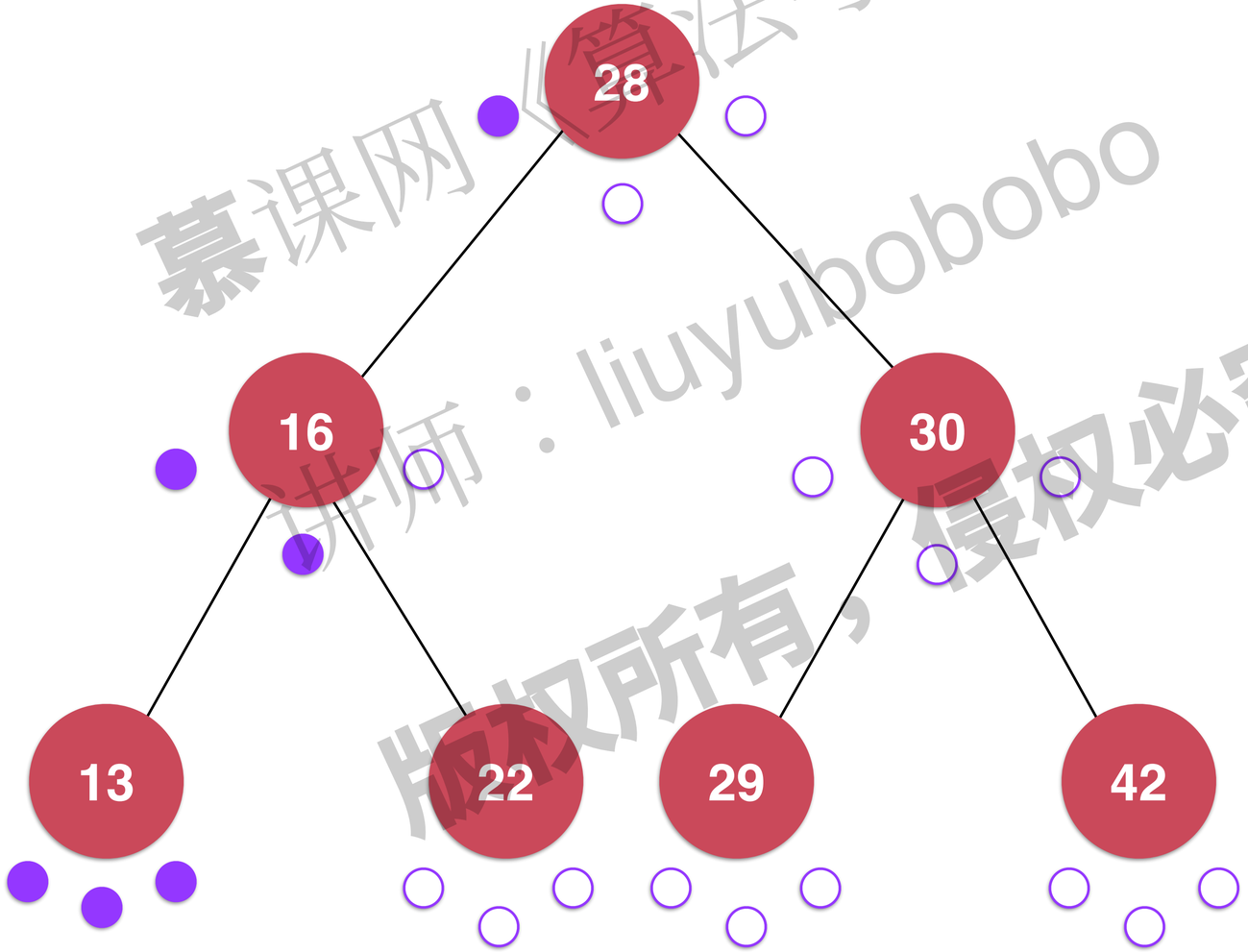
二分搜索树的后序遍历



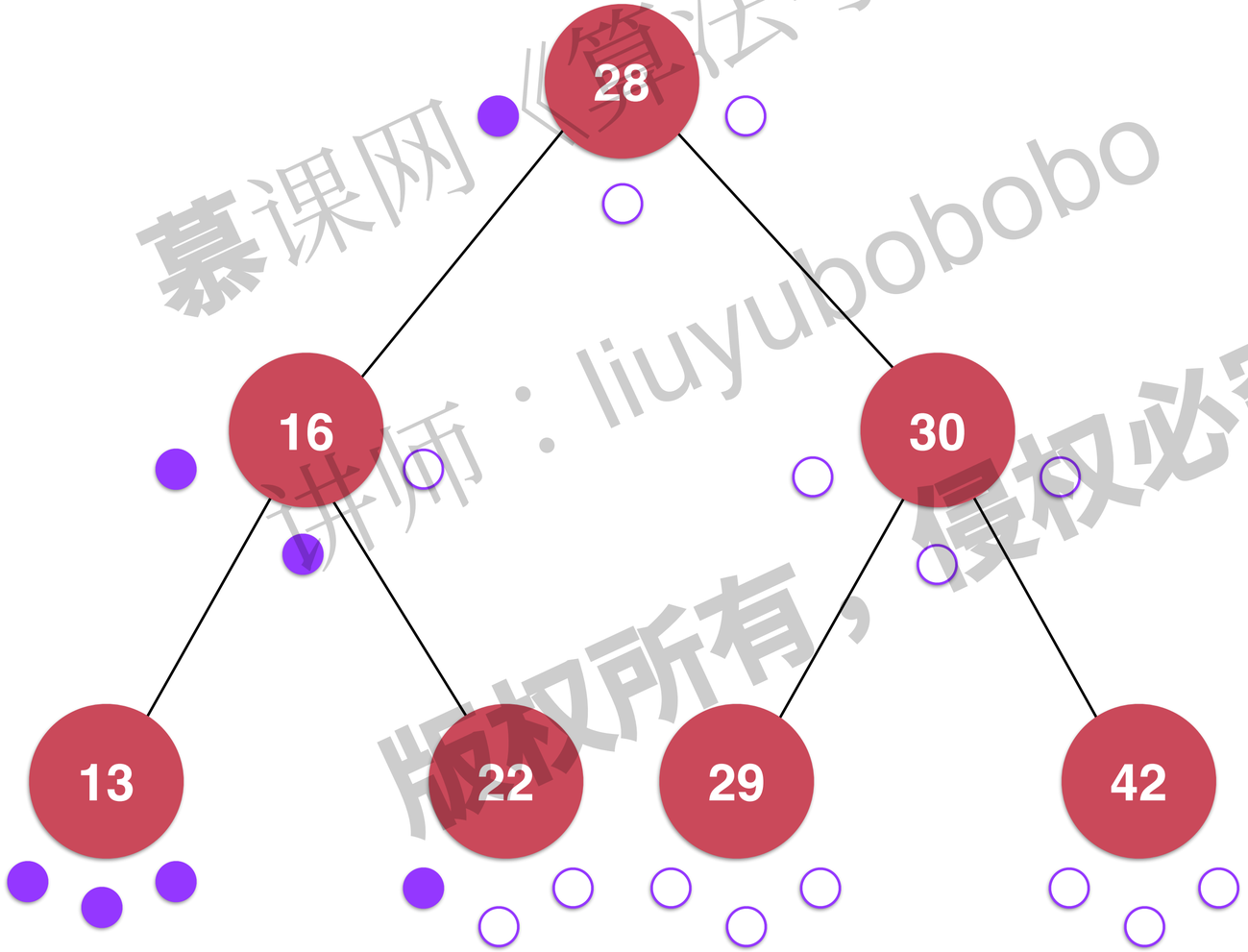
二分搜索树的后序遍历



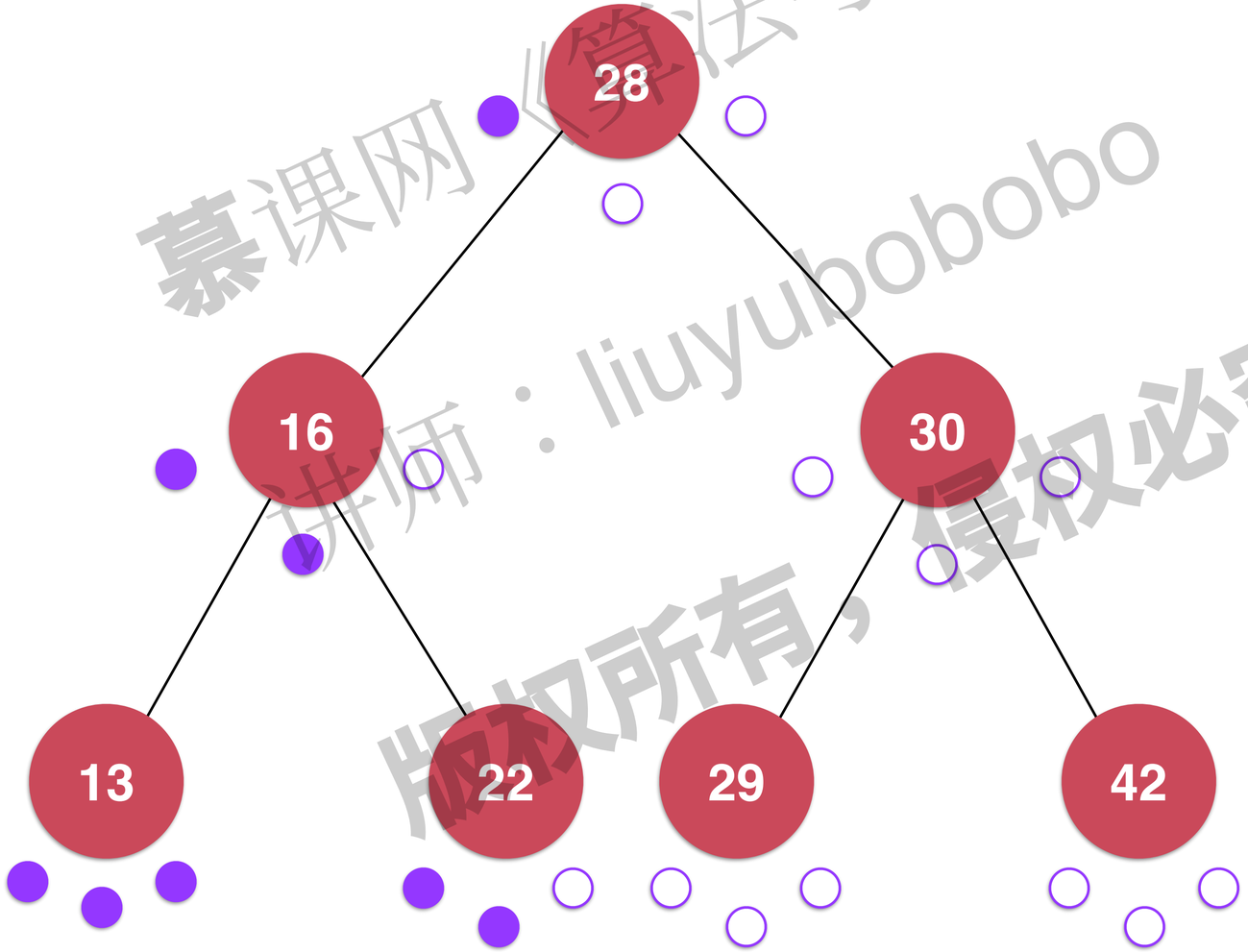
二分搜索树的后序遍历



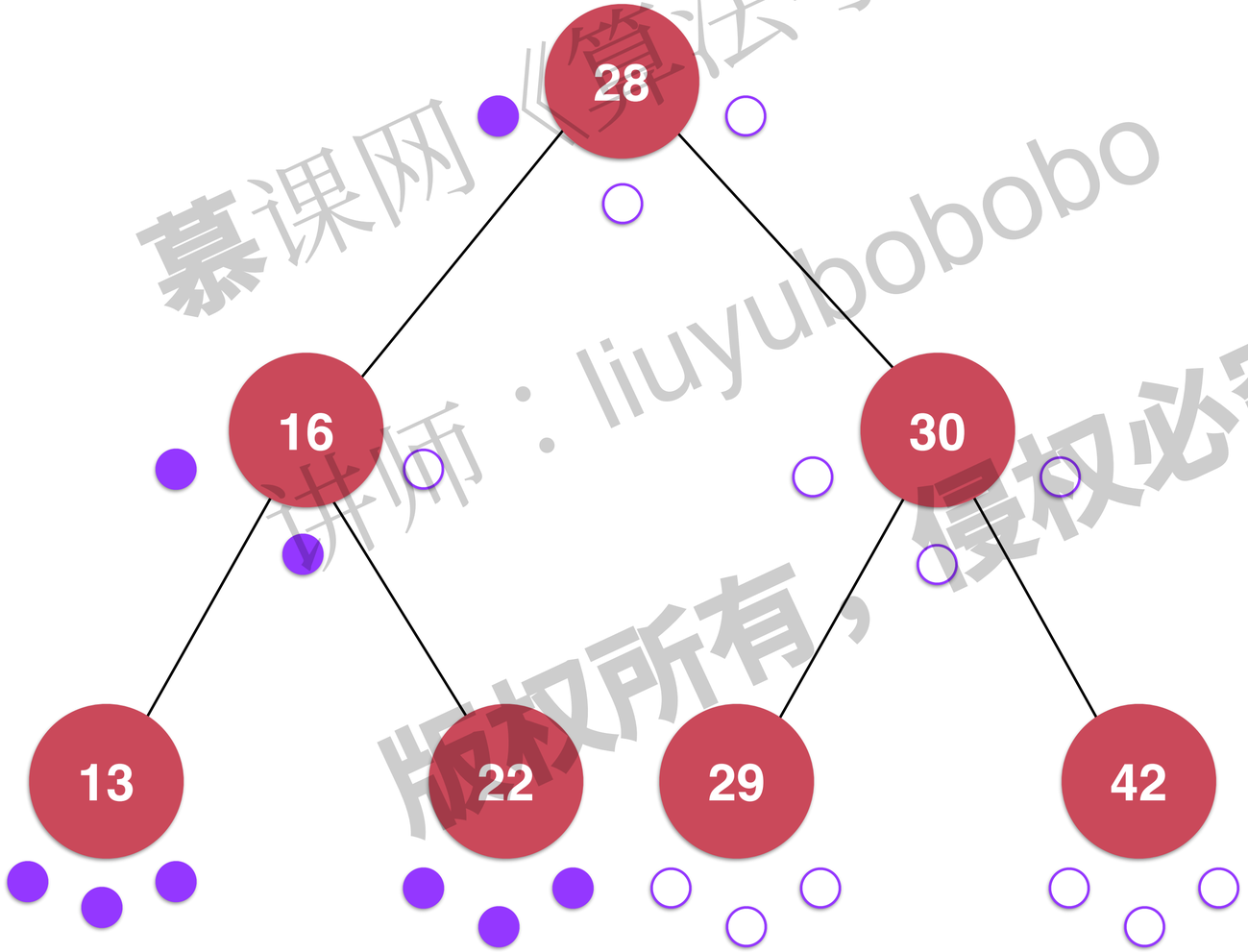
二分搜索树的后序遍历



二分搜索树的后序遍历



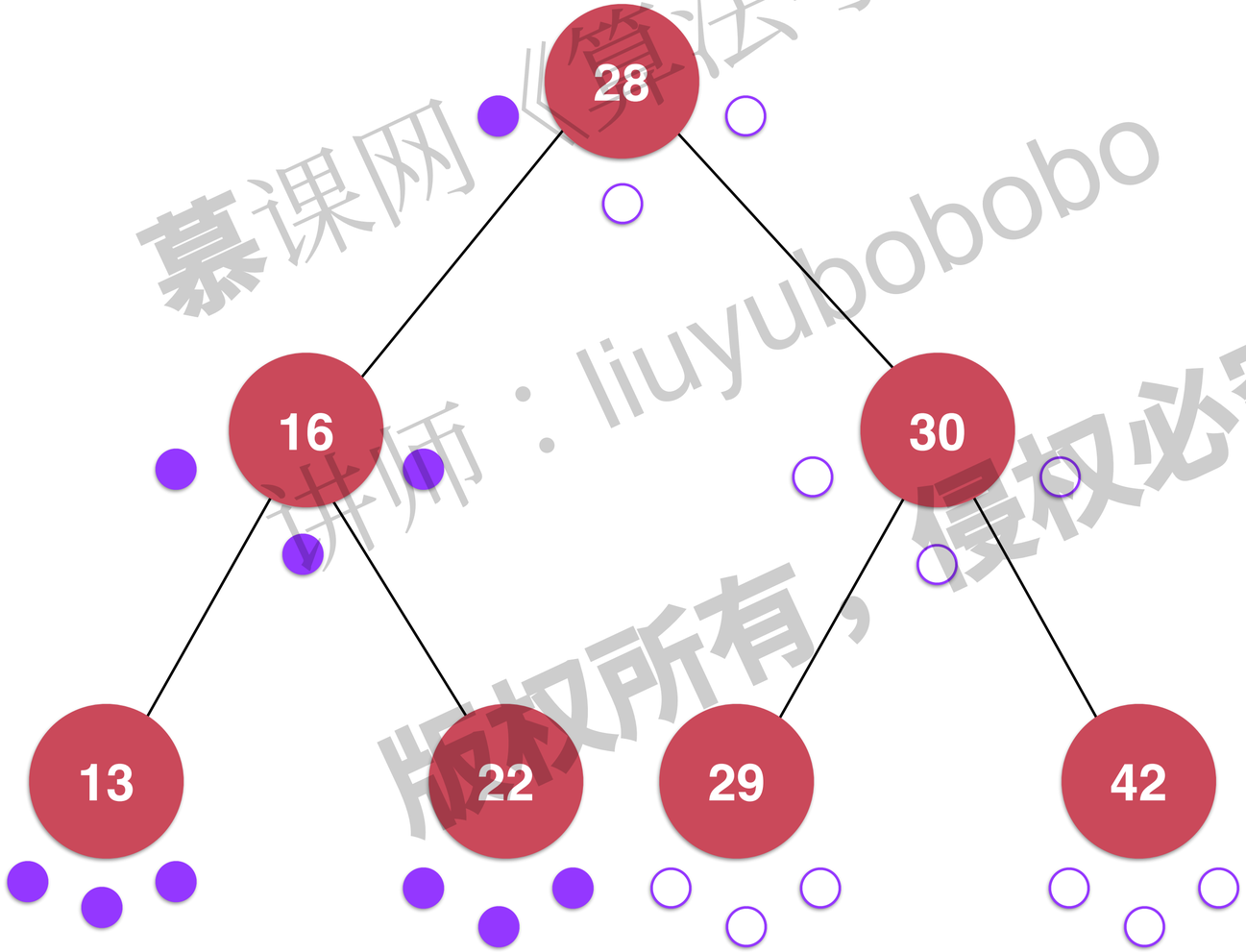
二分搜索树的后序遍历



1

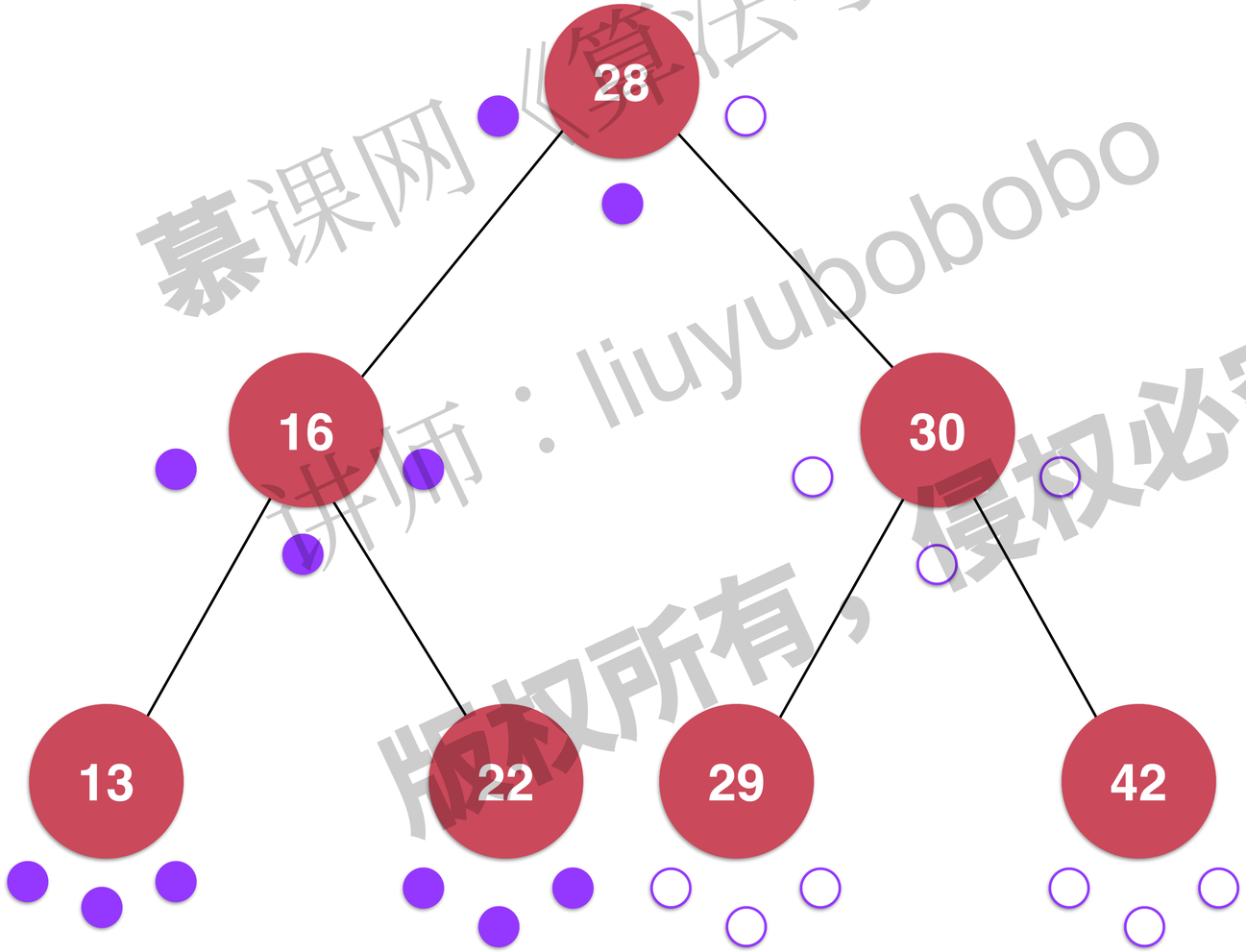
2

二分搜索树的后序遍历



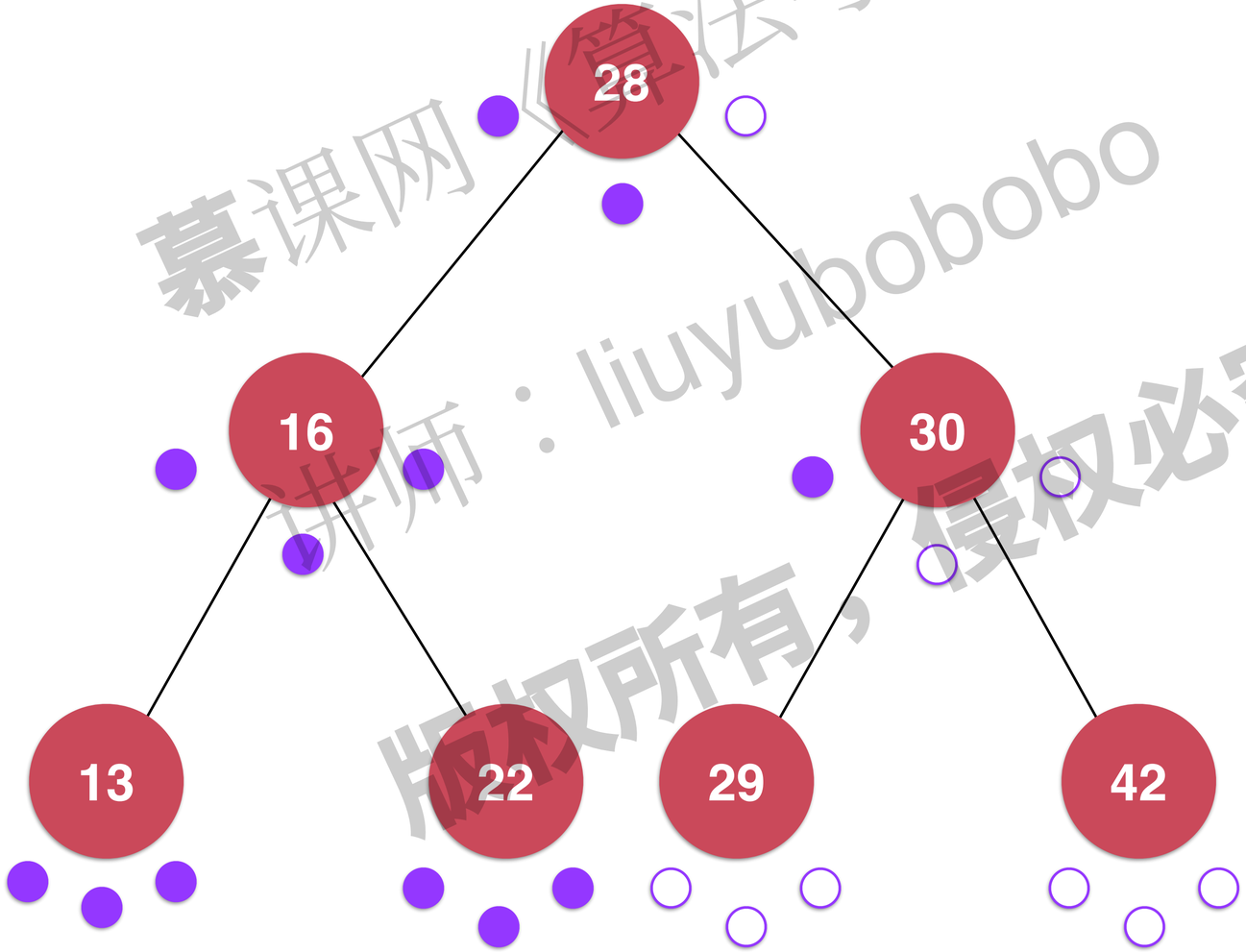
1
2
1

二分搜索树的后序遍历



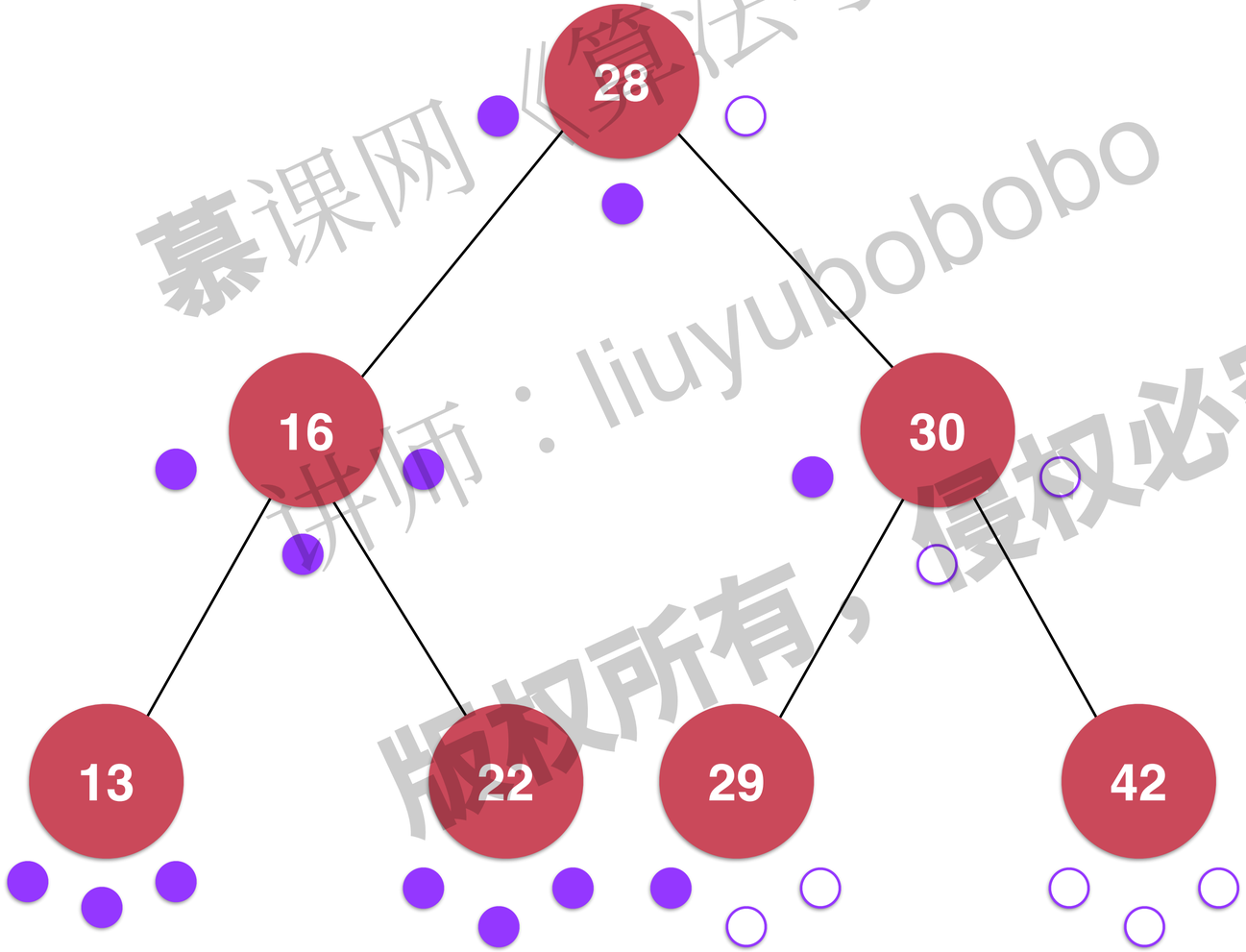
1
2
1

二分搜索树的后序遍历



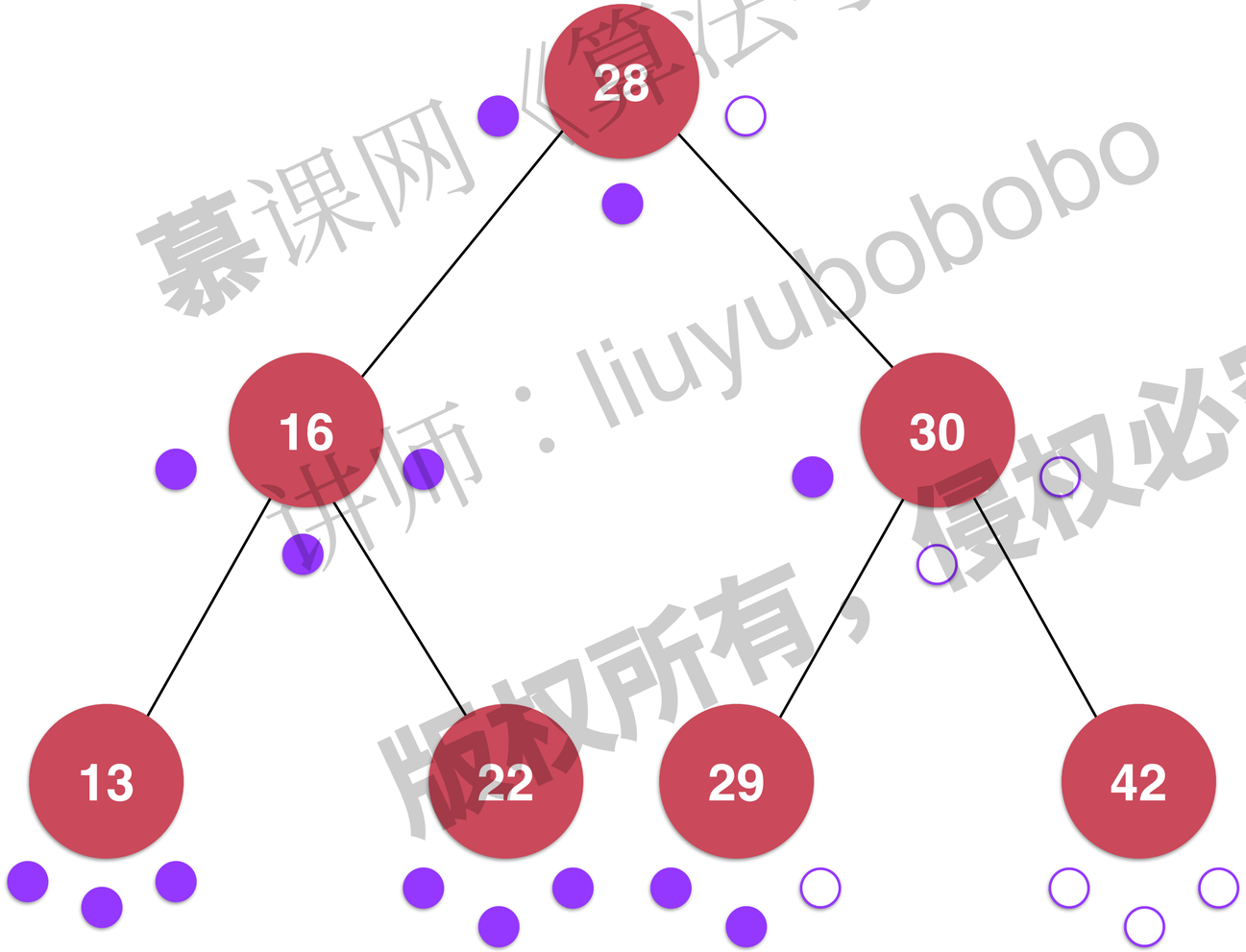
1
2
1

二分搜索树的后序遍历



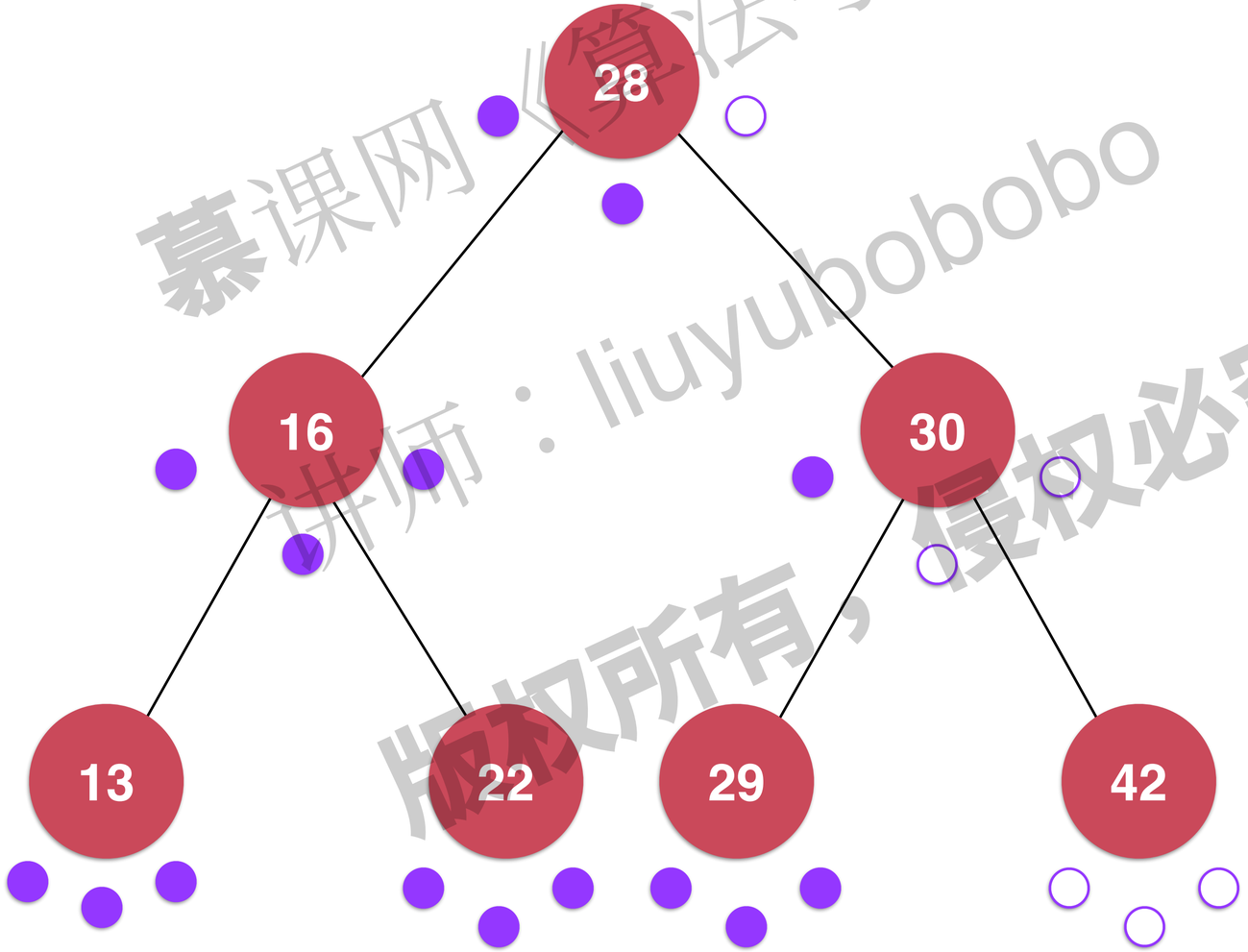
1
2
1

二分搜索树的后序遍历



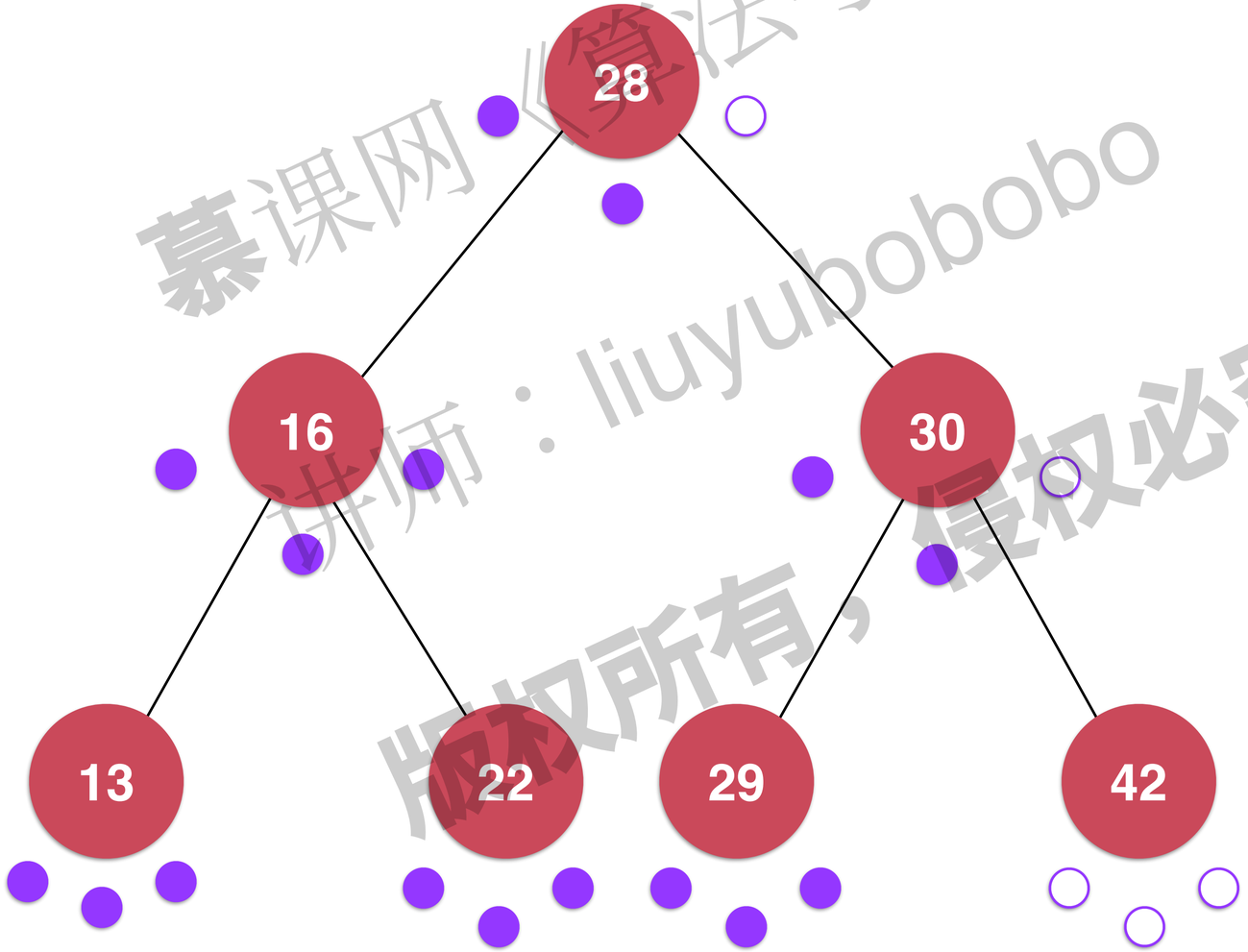
1
2
1

二分搜索树的后序遍历



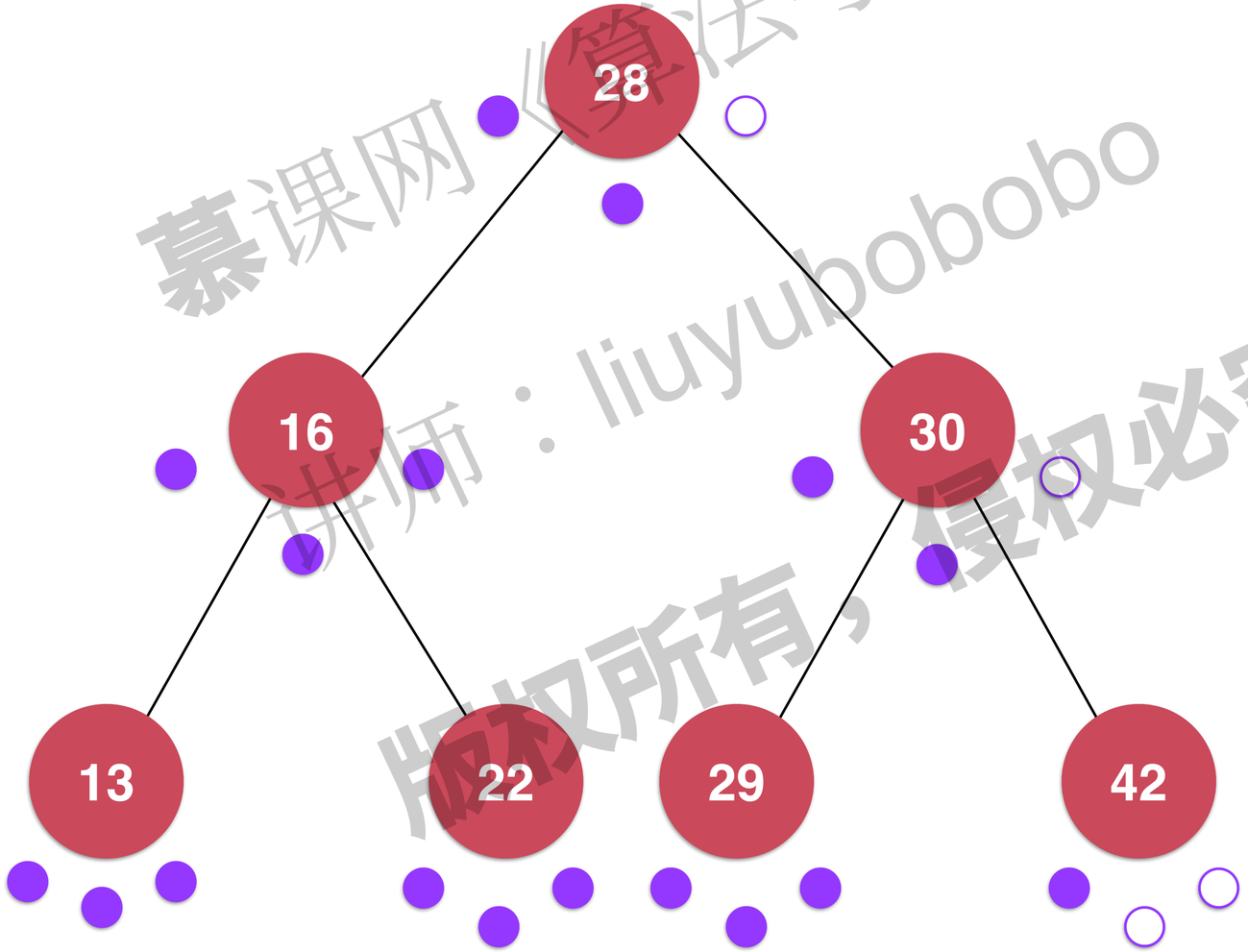
1
2
1
2

二分搜索树的后序遍历



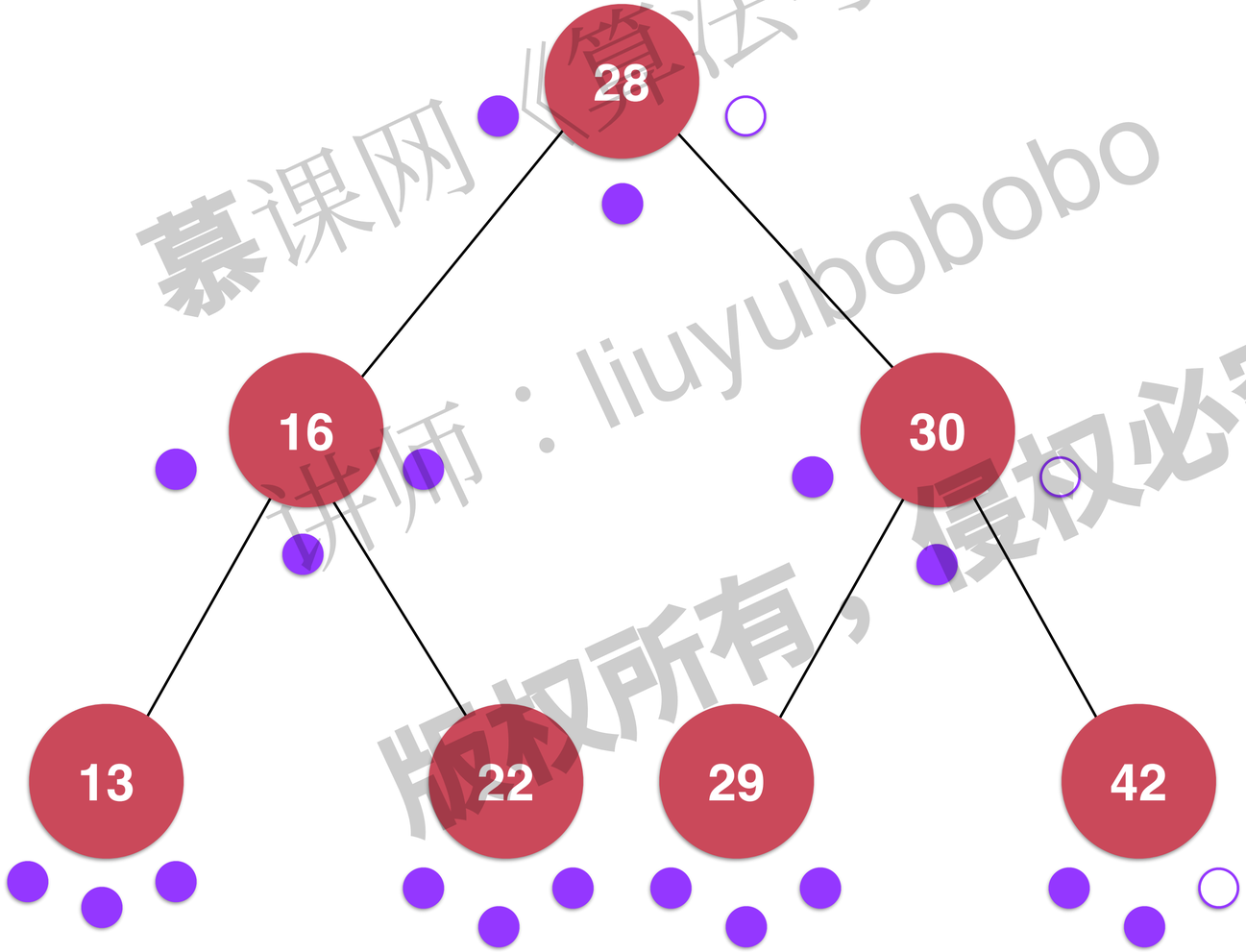
1
2
1
2

二分搜索树的后序遍历



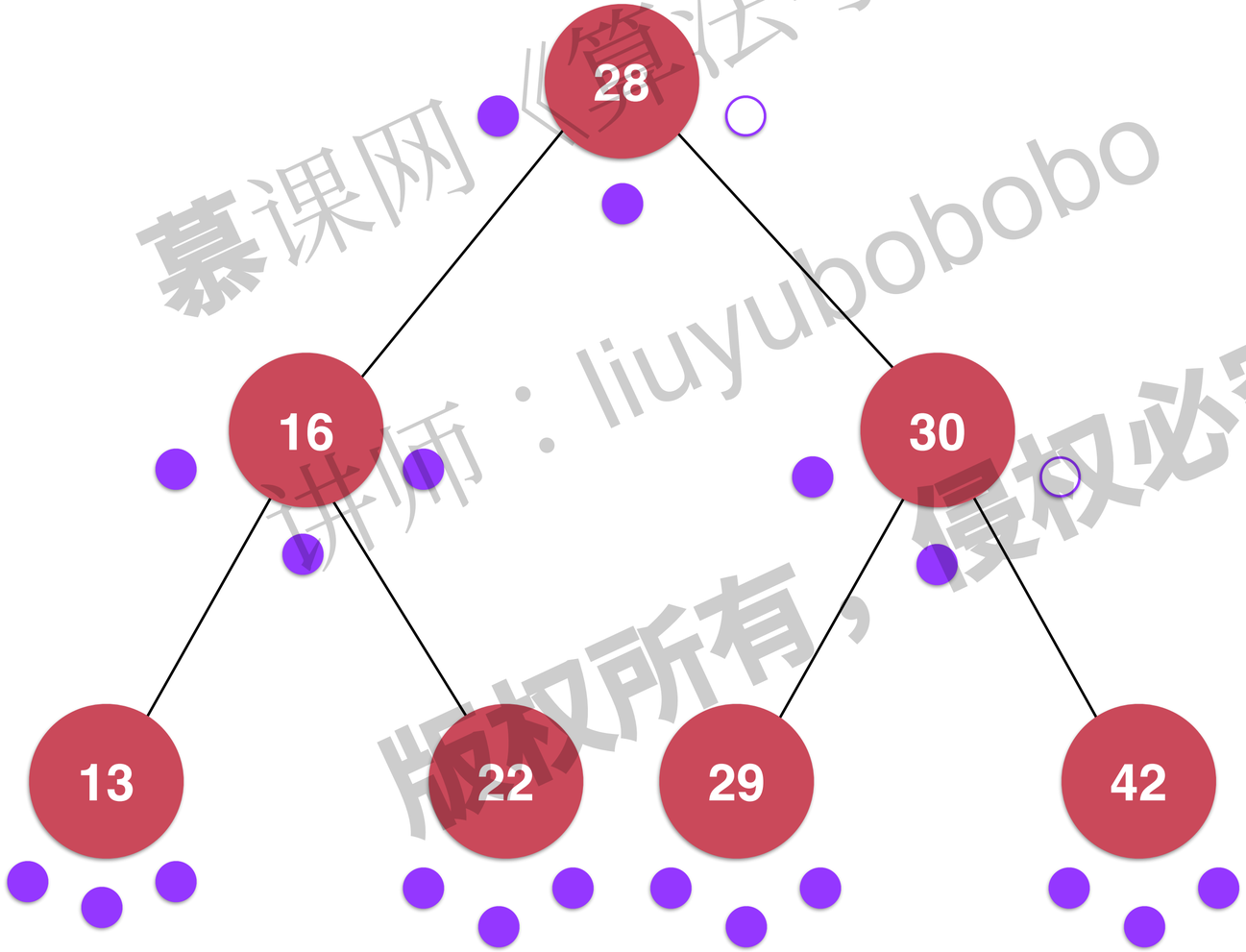
1
2
1
2

二分搜索树的后序遍历

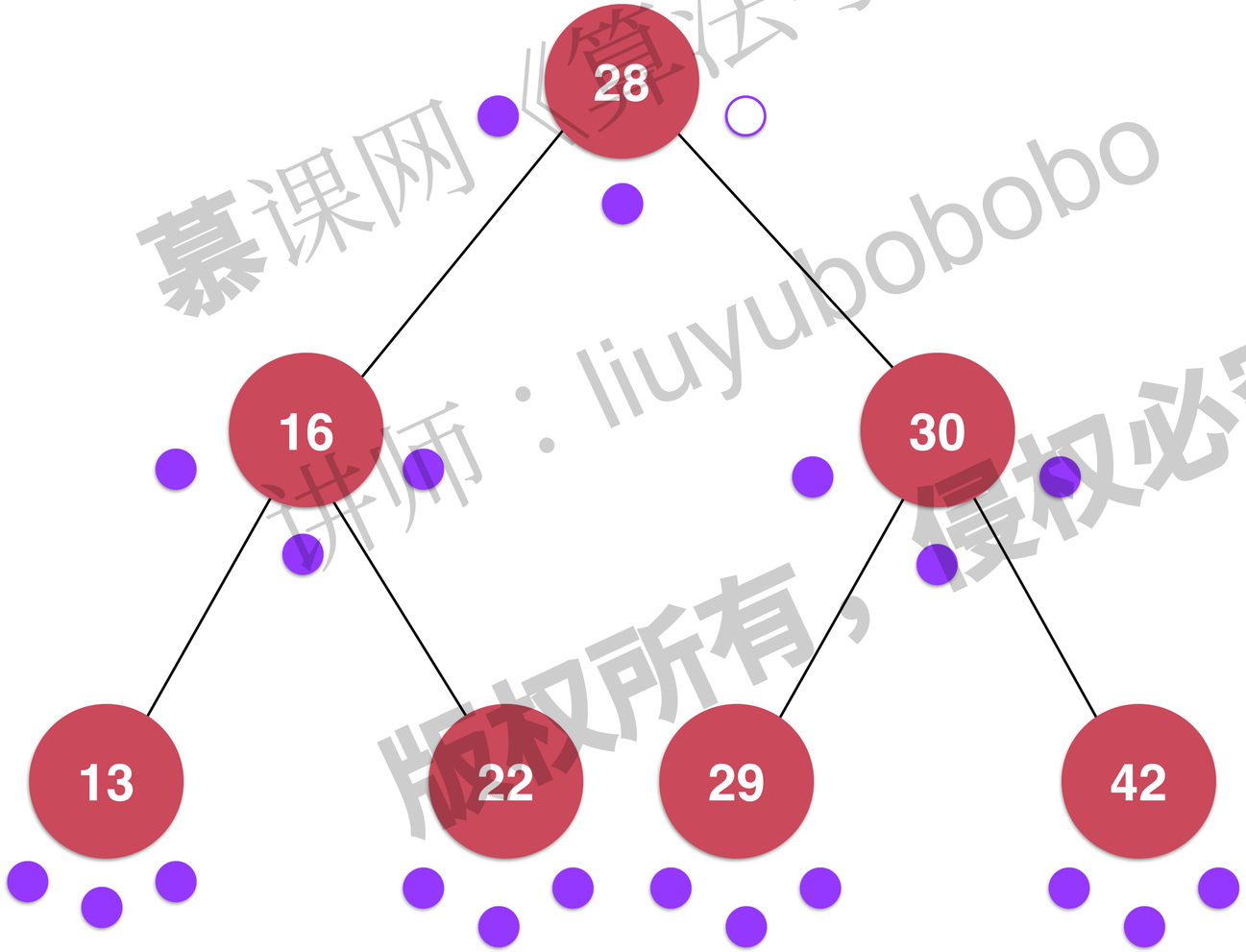


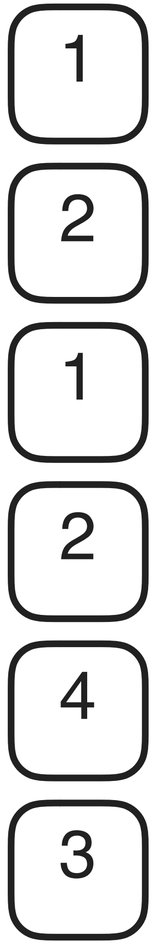
1
2
1
2

二分搜索树的后序遍历

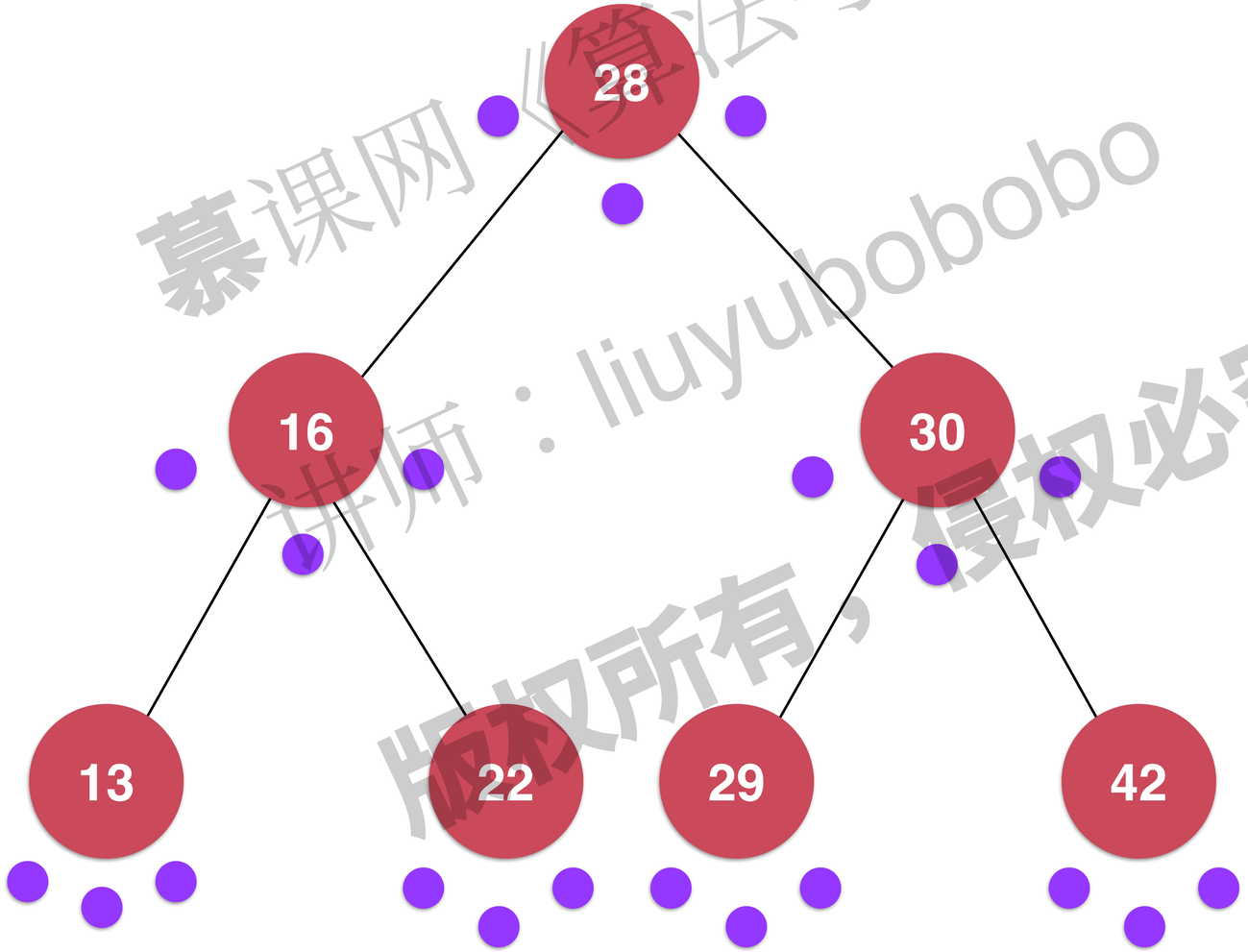


二分搜索树的后序遍历



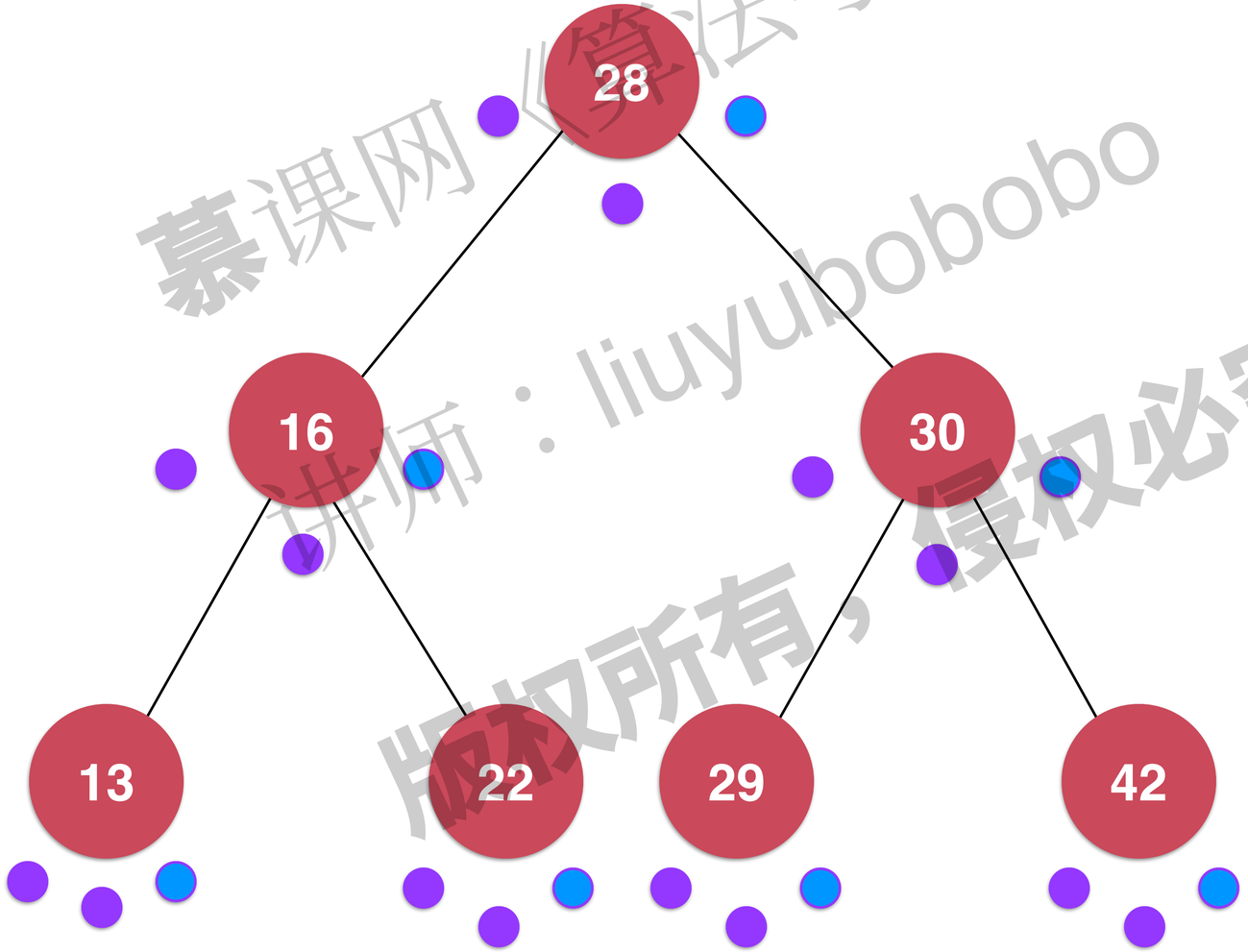


二分搜索树的后序遍历

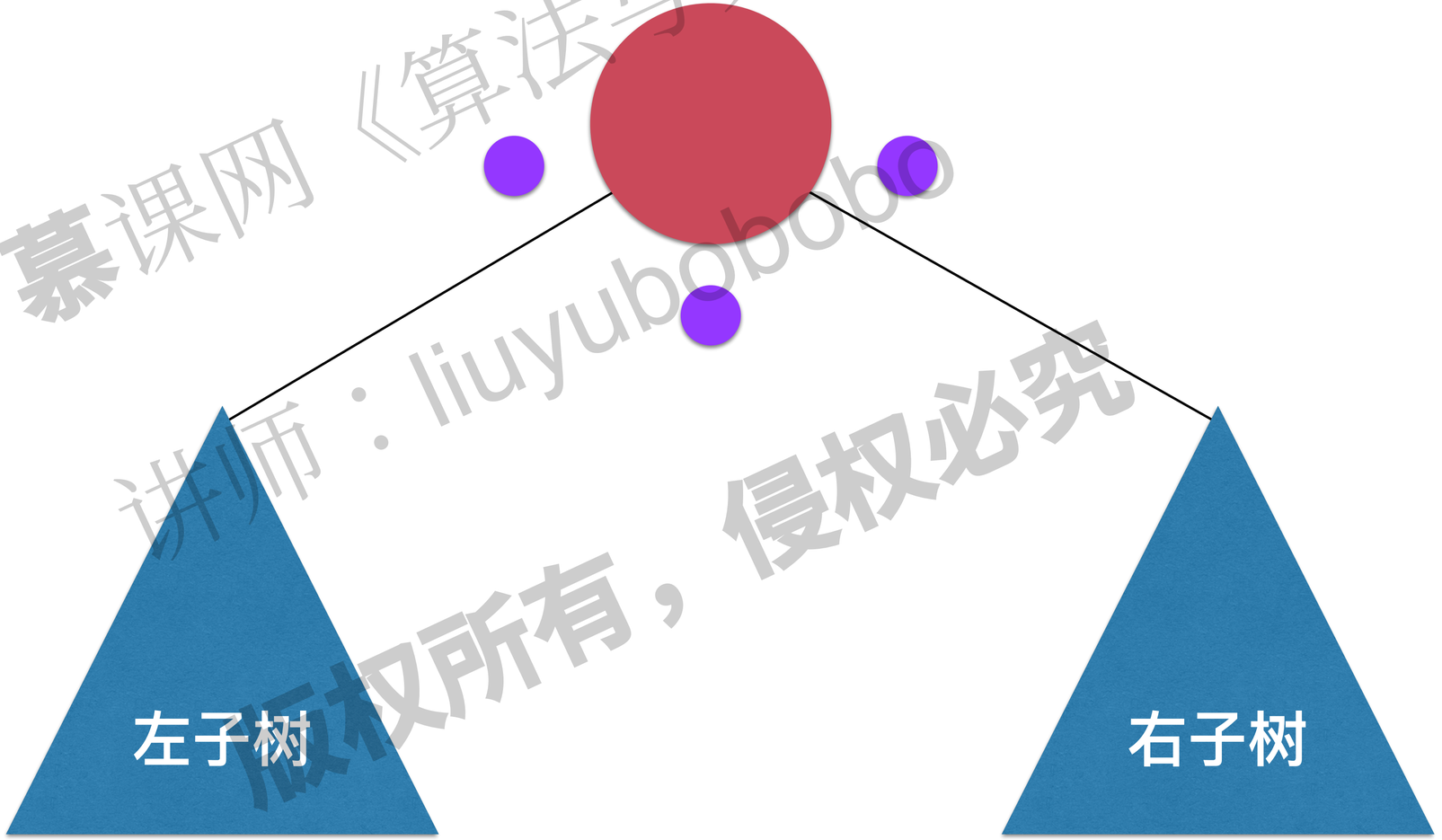


1
2
1
2
4
3
2

二分搜索树的后序遍历



二分搜索树的遍历



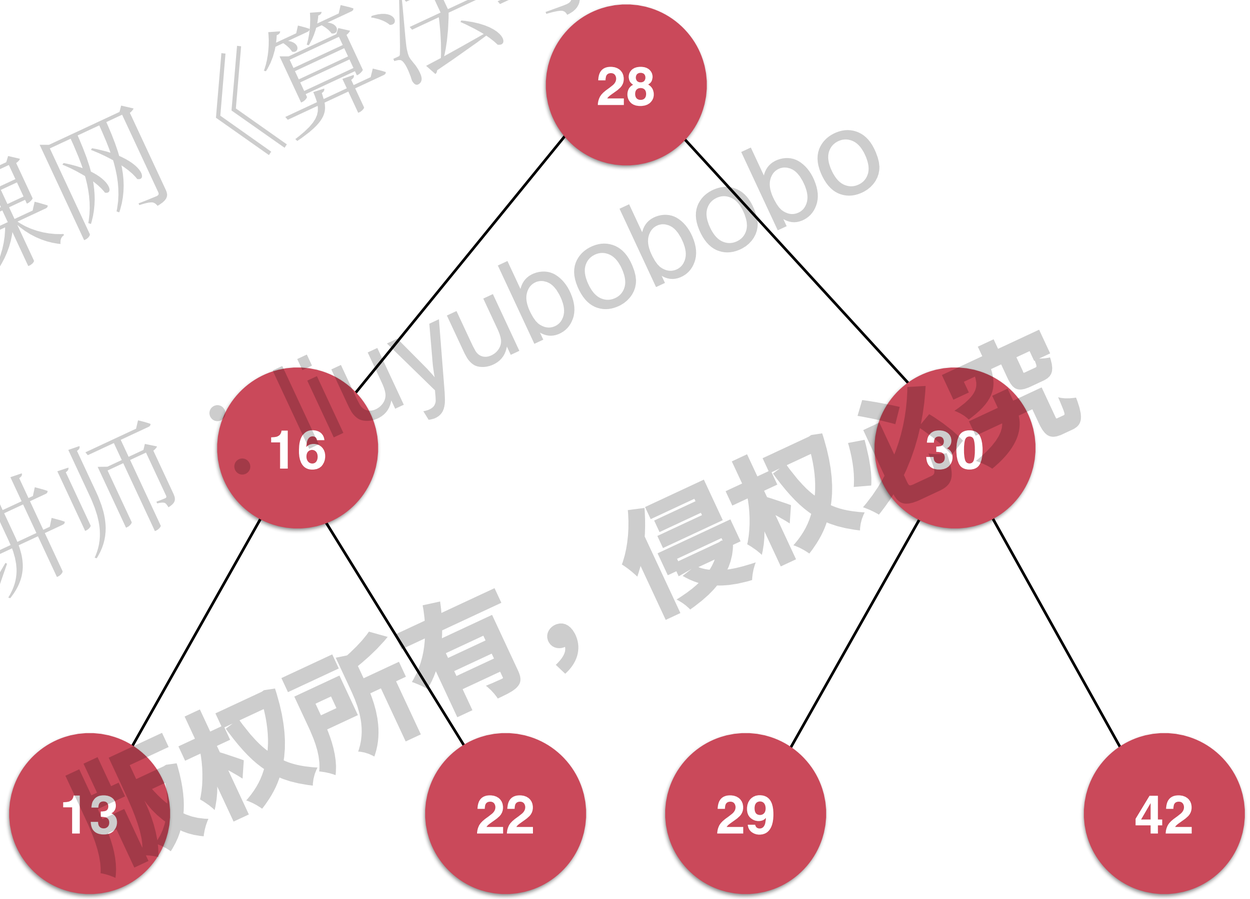
操作：二分搜索树的前中后序遍历

后续遍历的一个应用：二叉树的销毁

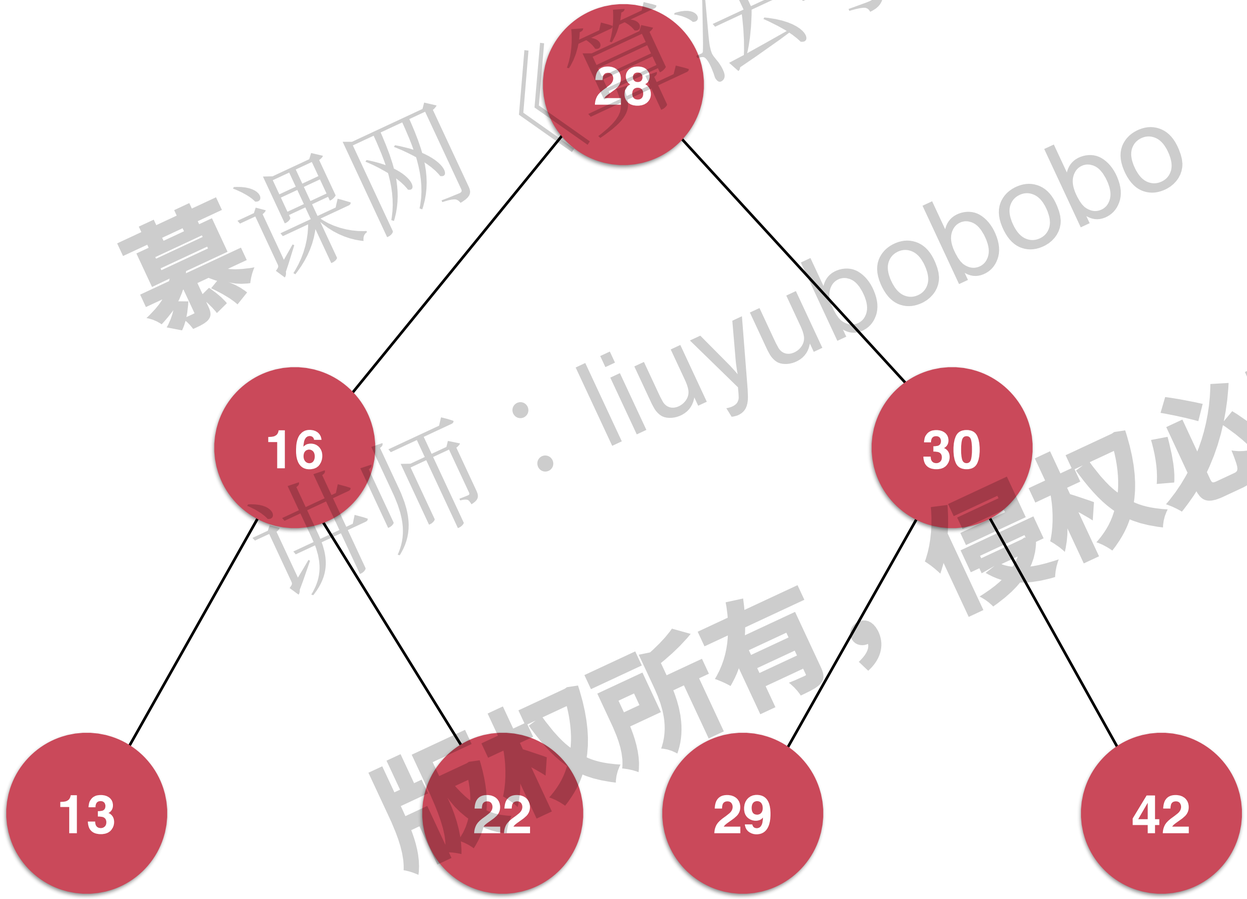
操作：二分搜索树的销毁

层序遍历

二分搜索树的深度优先遍历



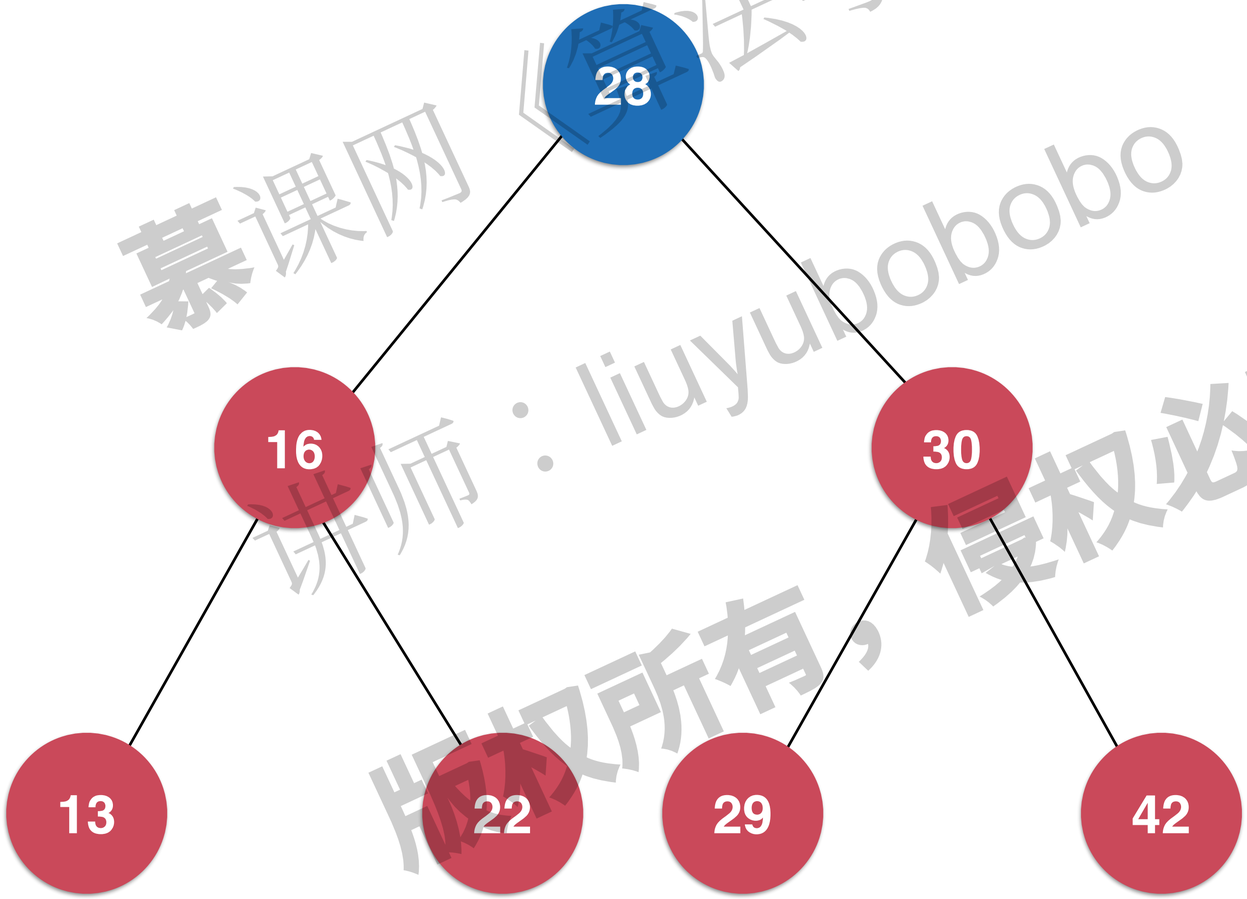
二分搜索树的广度优先遍历（层序）

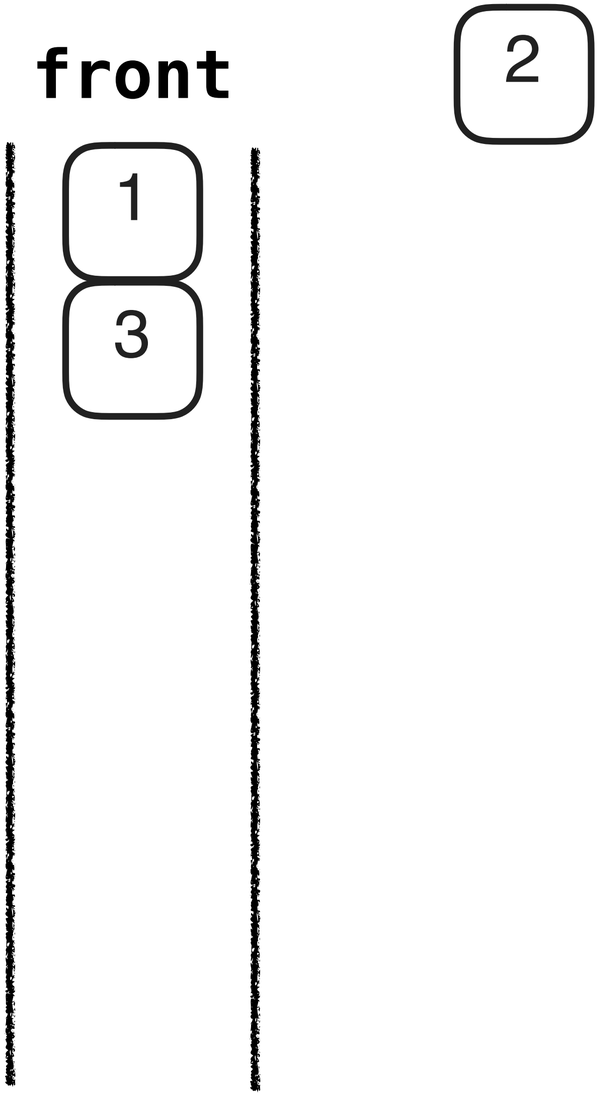


front

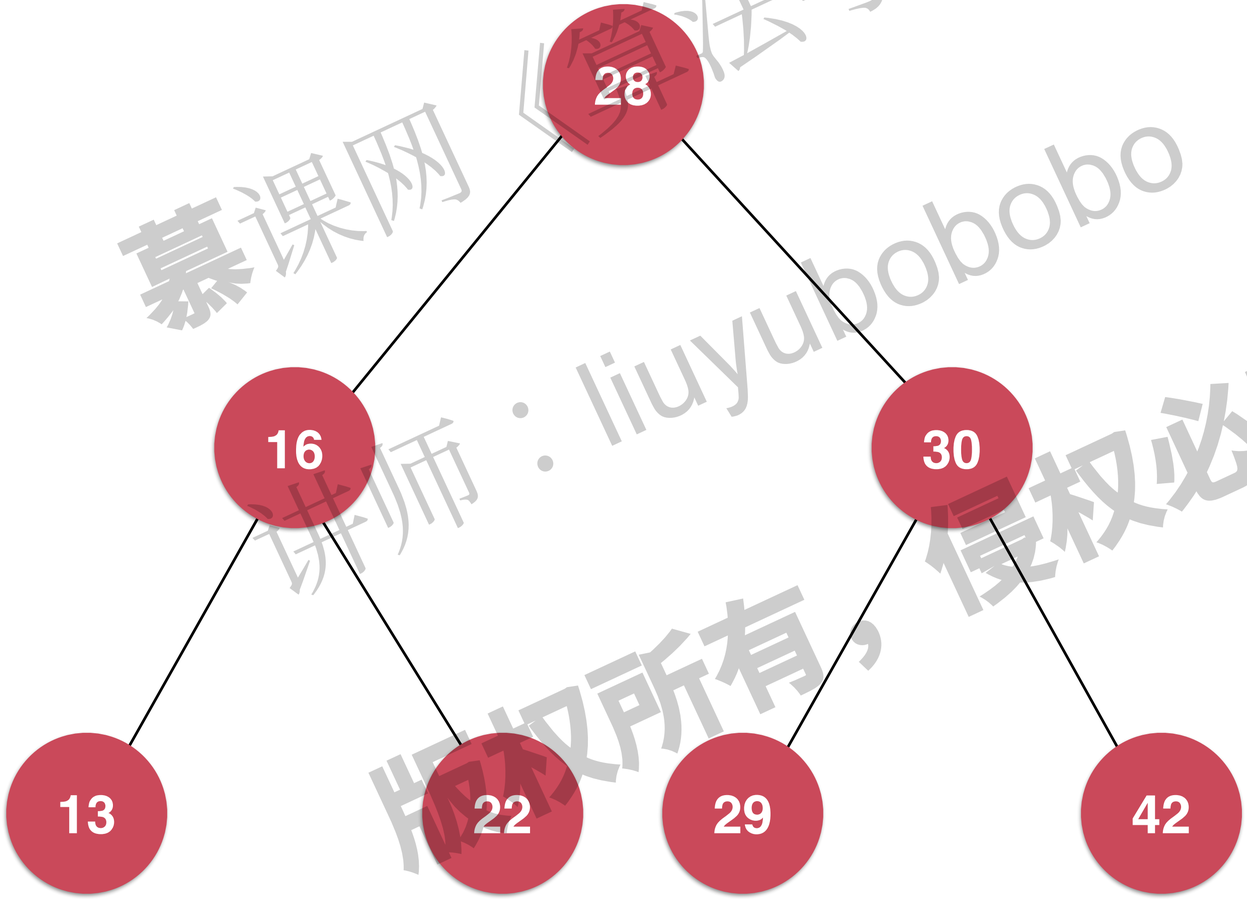
2

二分搜索树的广度优先遍历（层序）

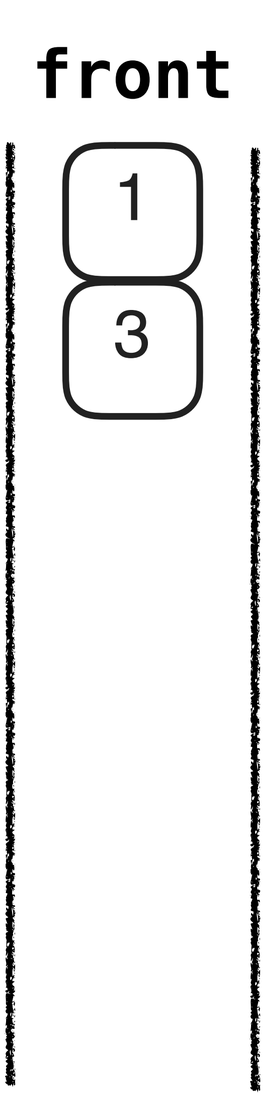




二分搜索树的广度优先遍历（层序）

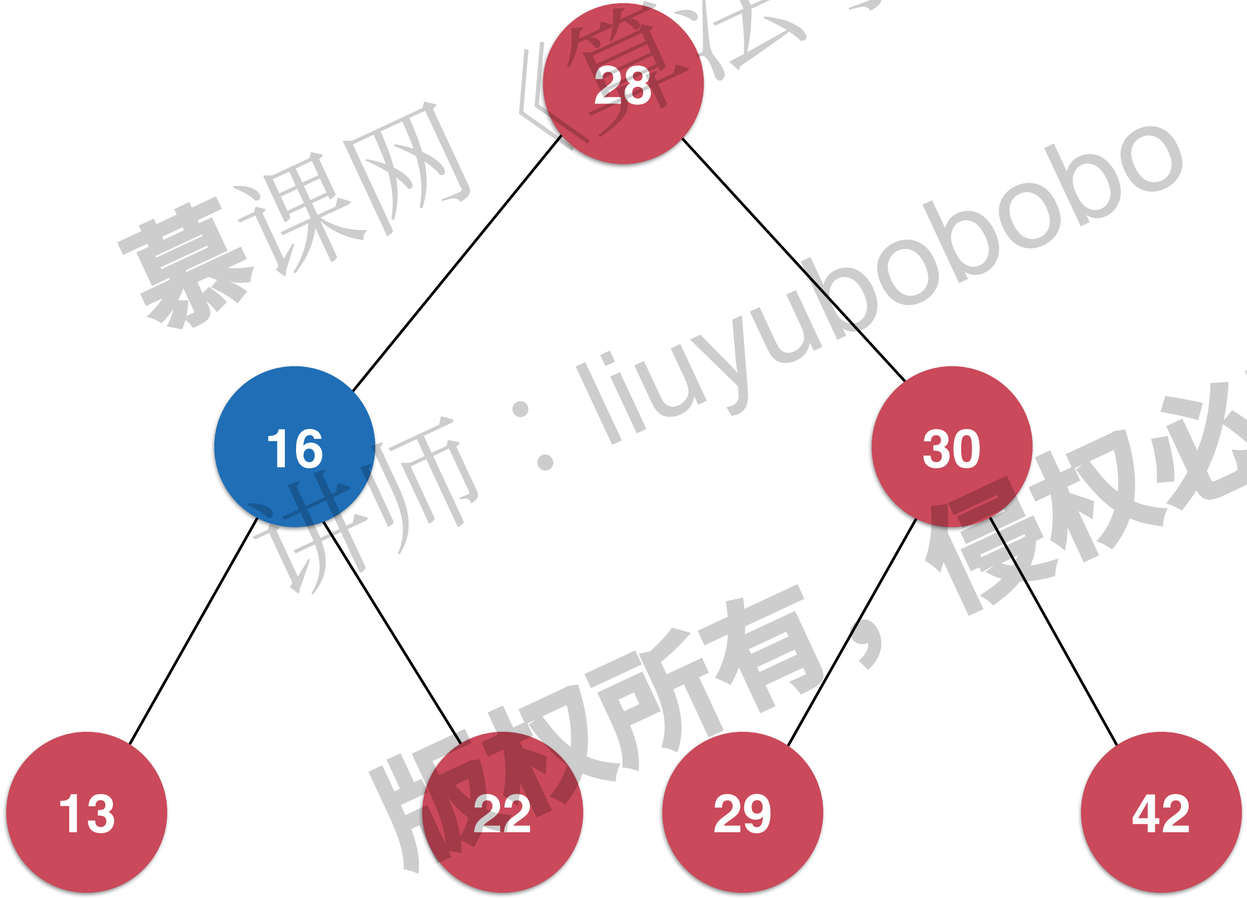


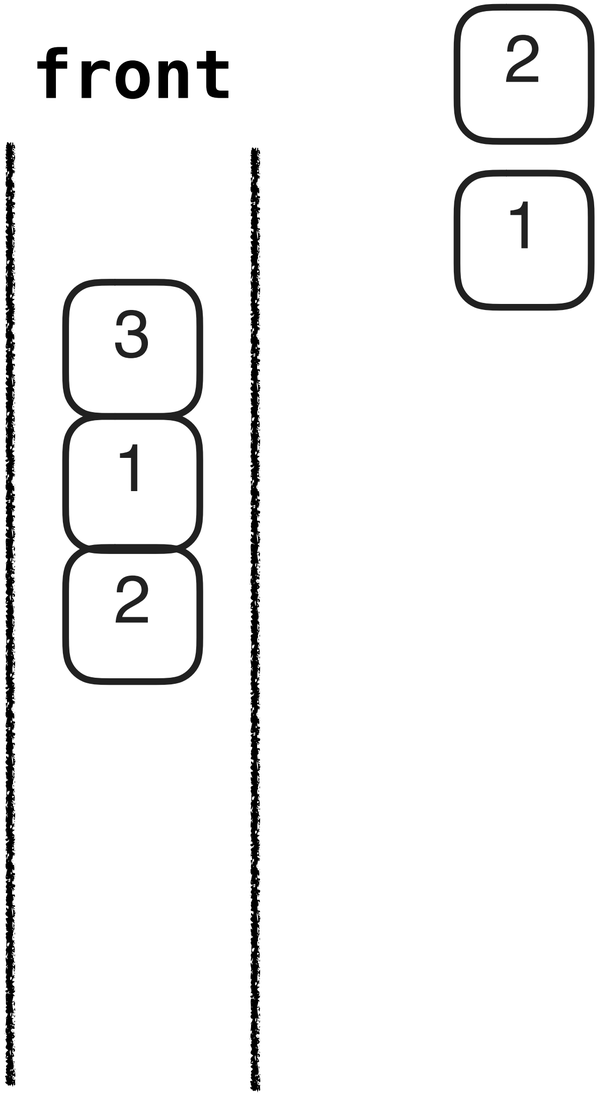
front



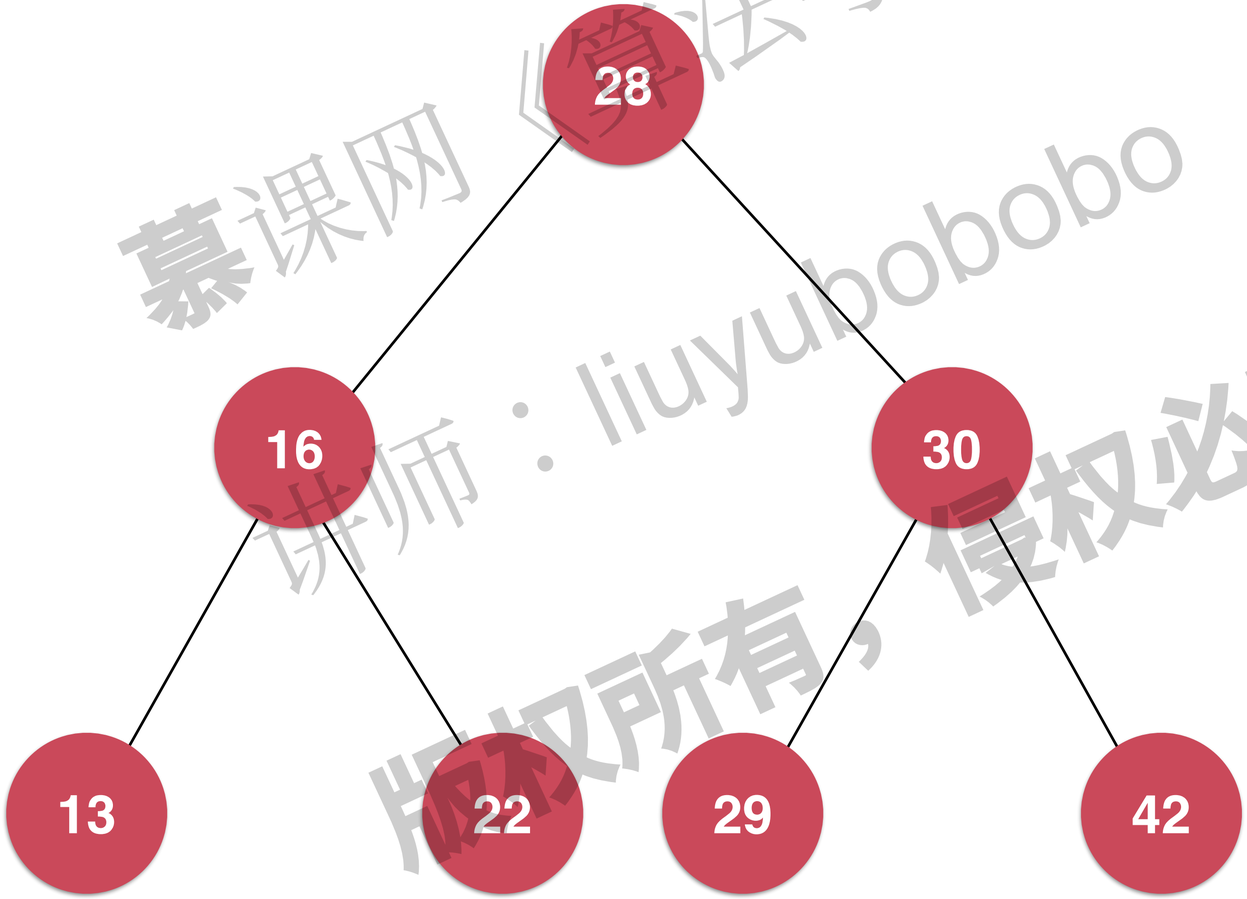
2

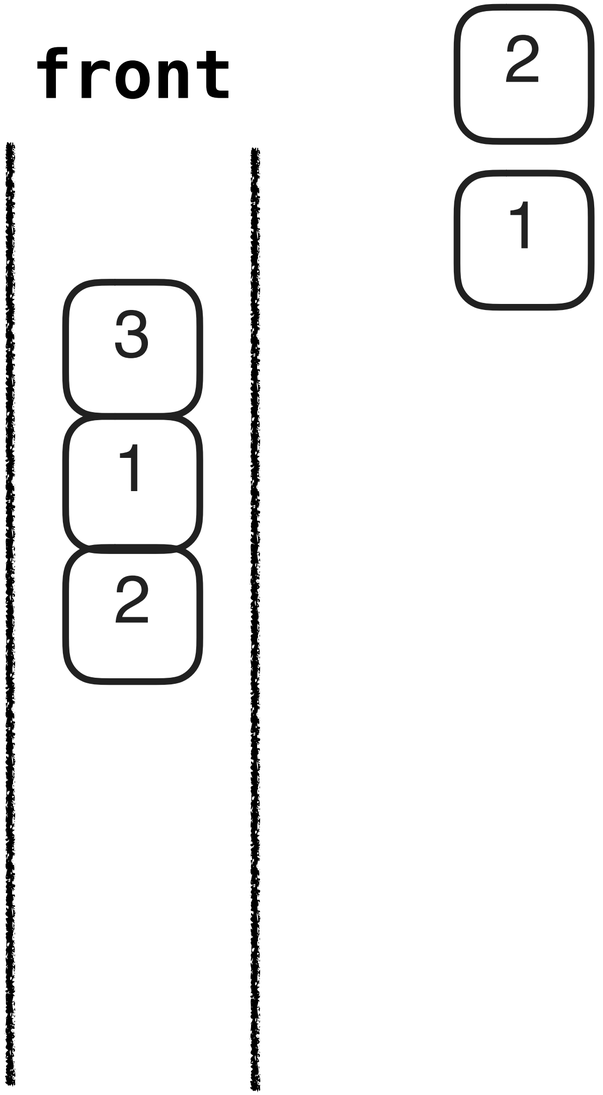
二分搜索树的广度优先遍历（层序）



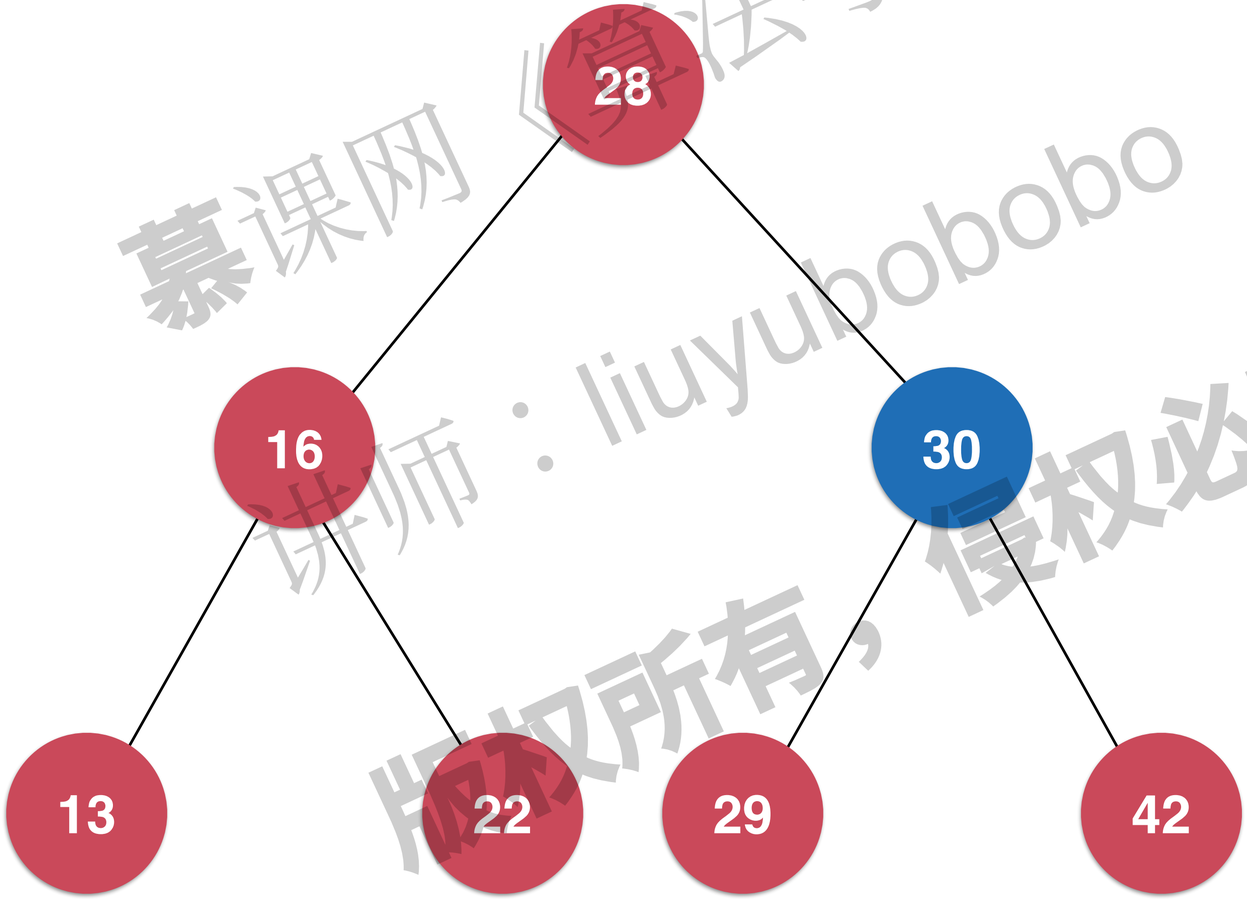


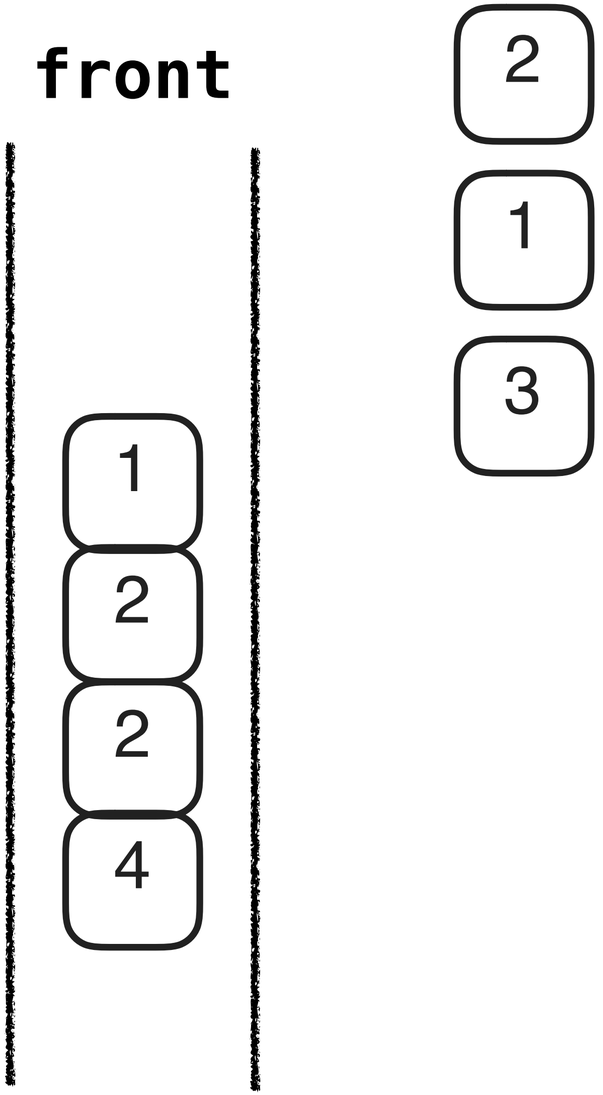
二分搜索树的广度优先遍历（层序）



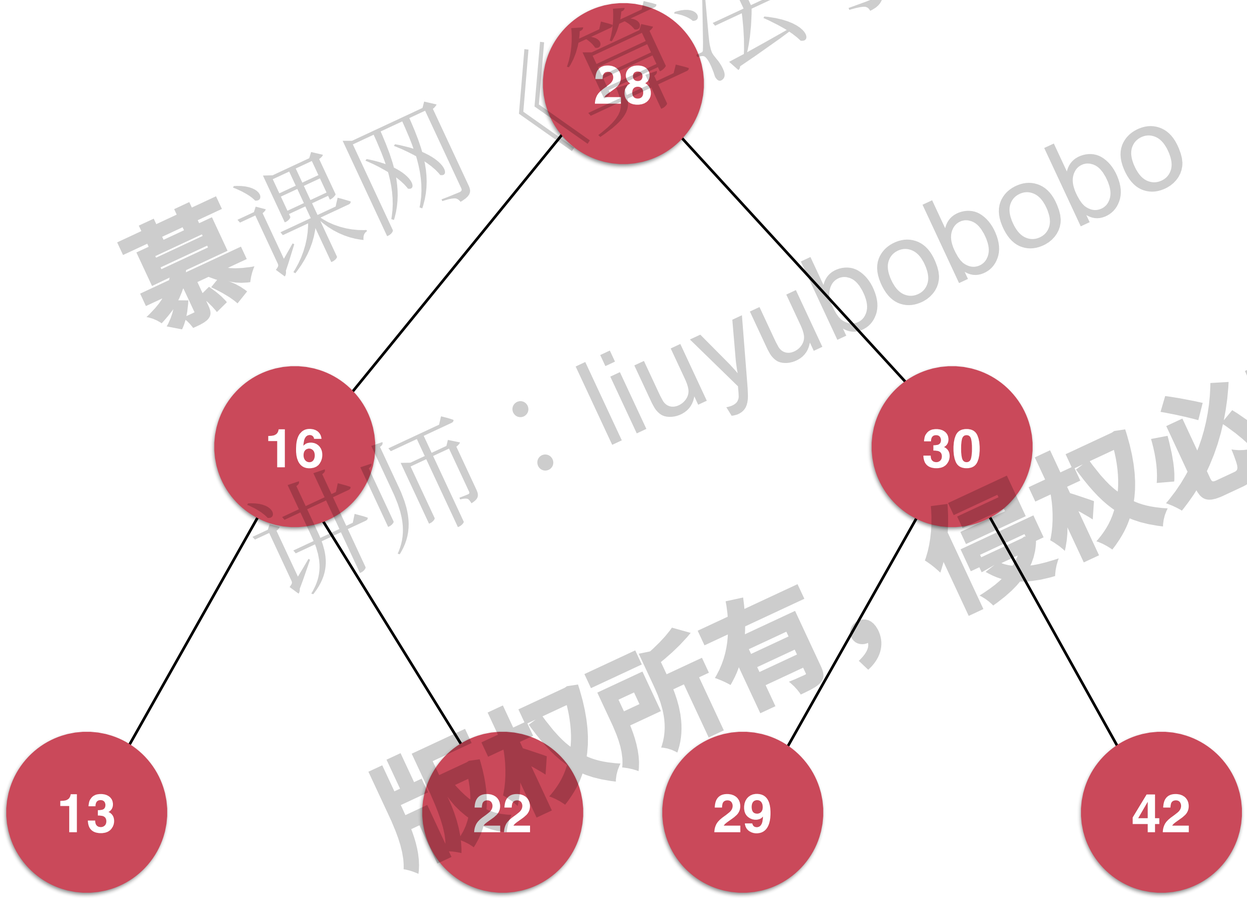


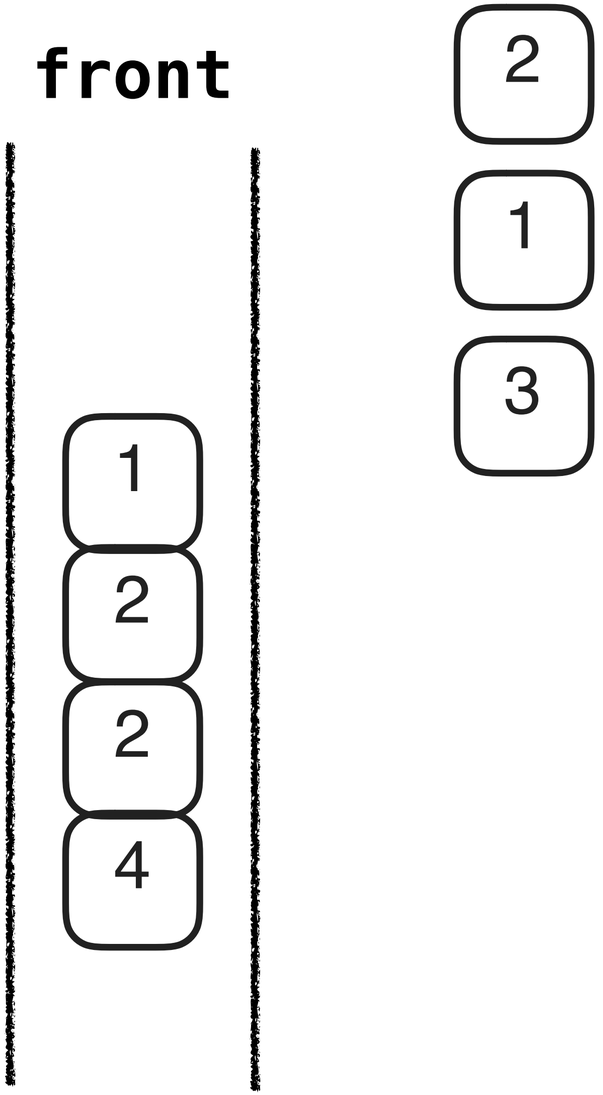
二分搜索树的广度优先遍历（层序）



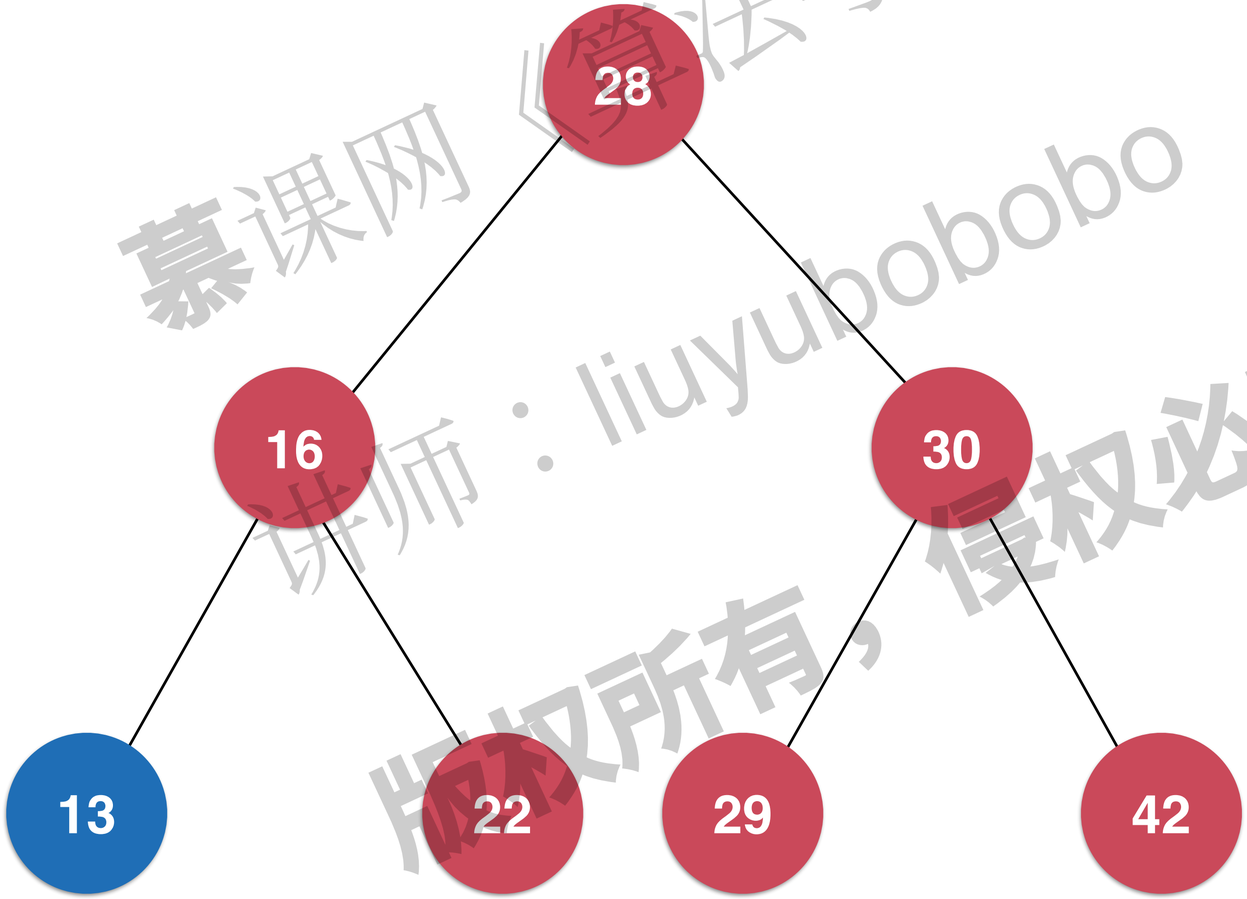


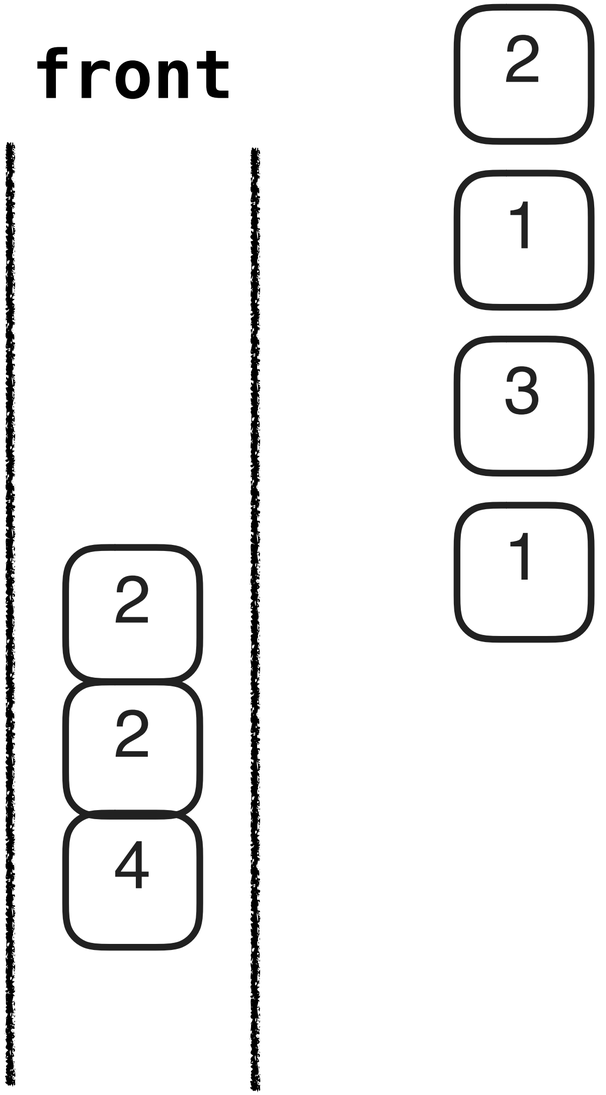
二分搜索树的广度优先遍历（层序）



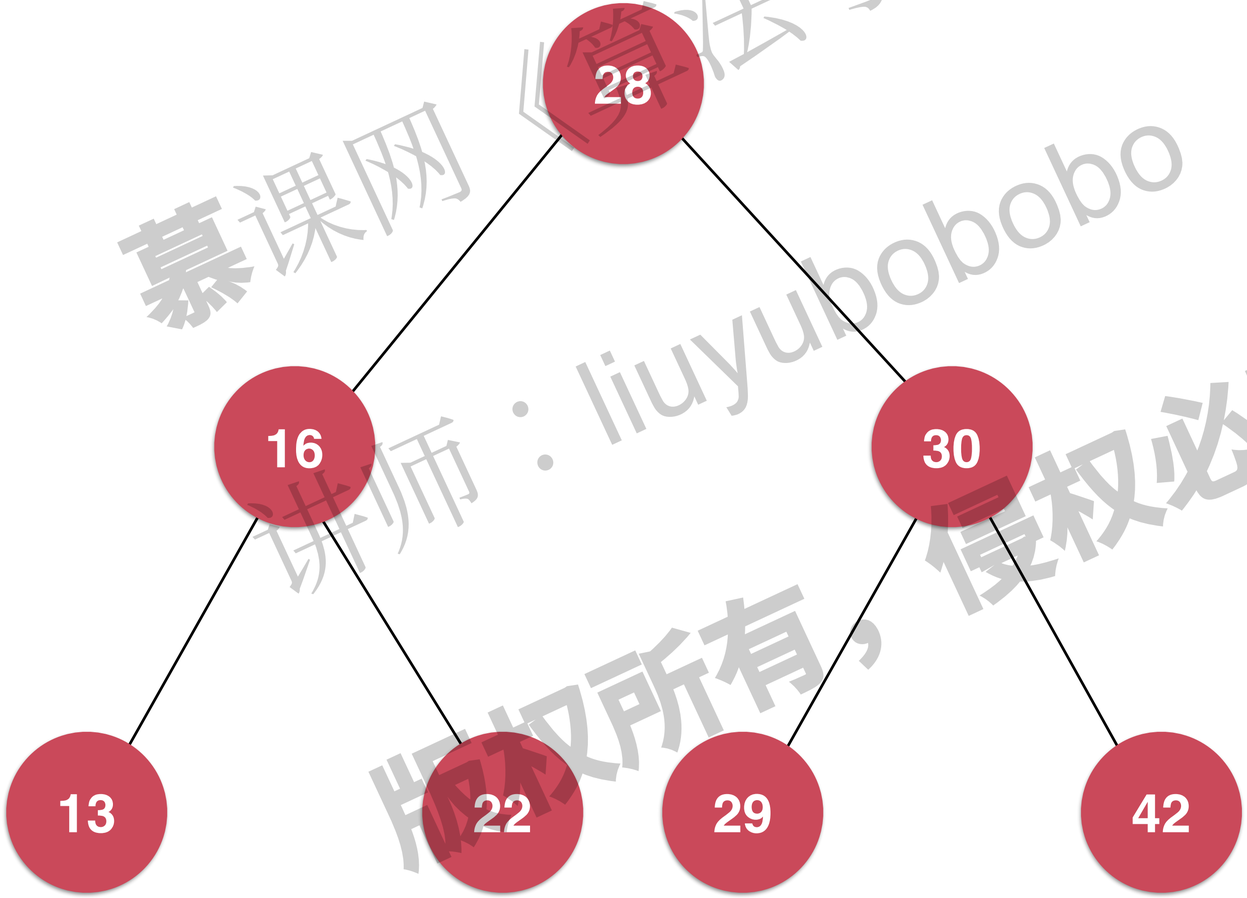


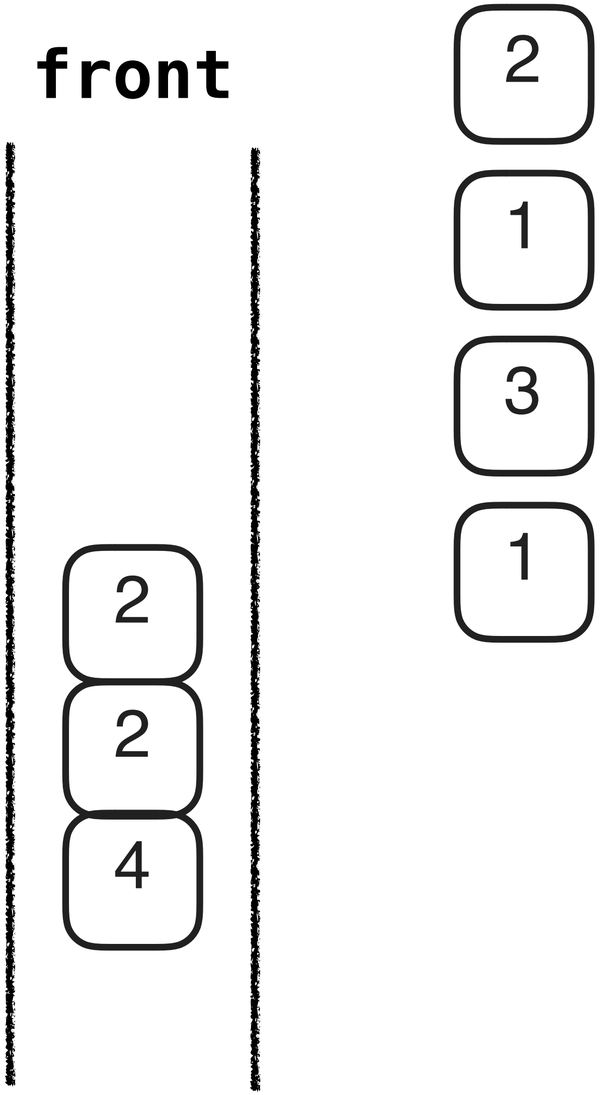
二分搜索树的广度优先遍历（层序）



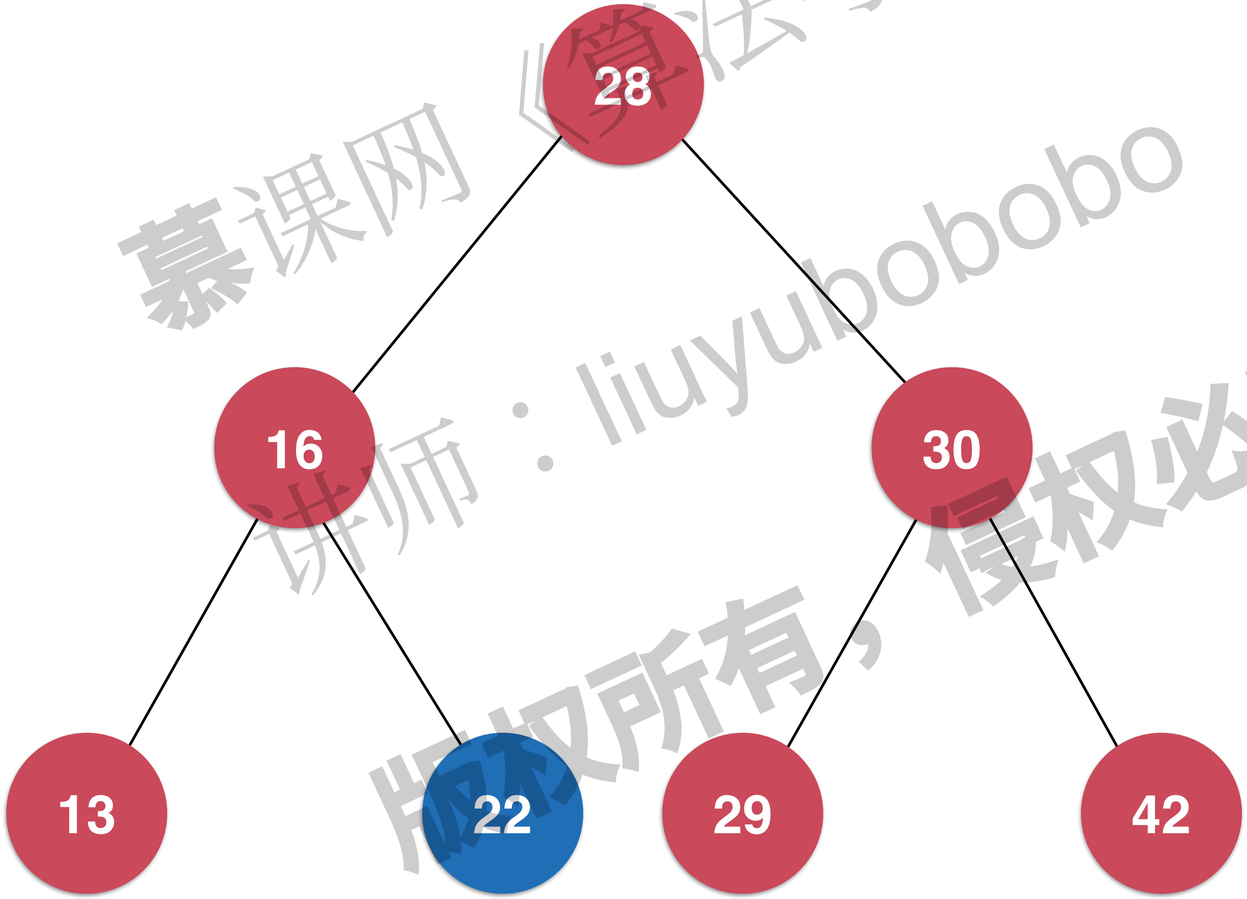


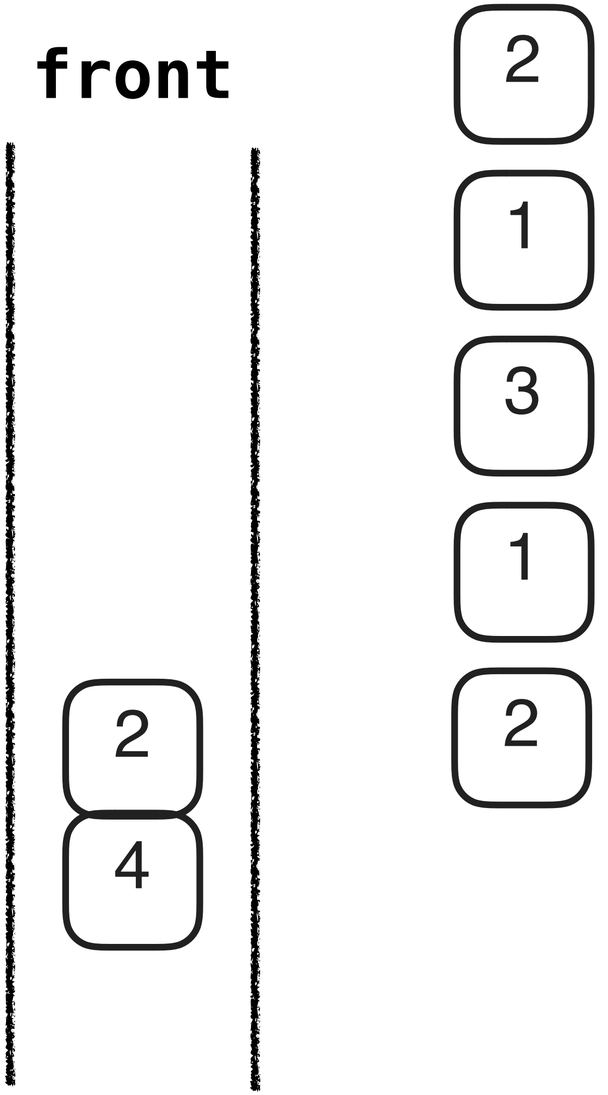
二分搜索树的广度优先遍历（层序）



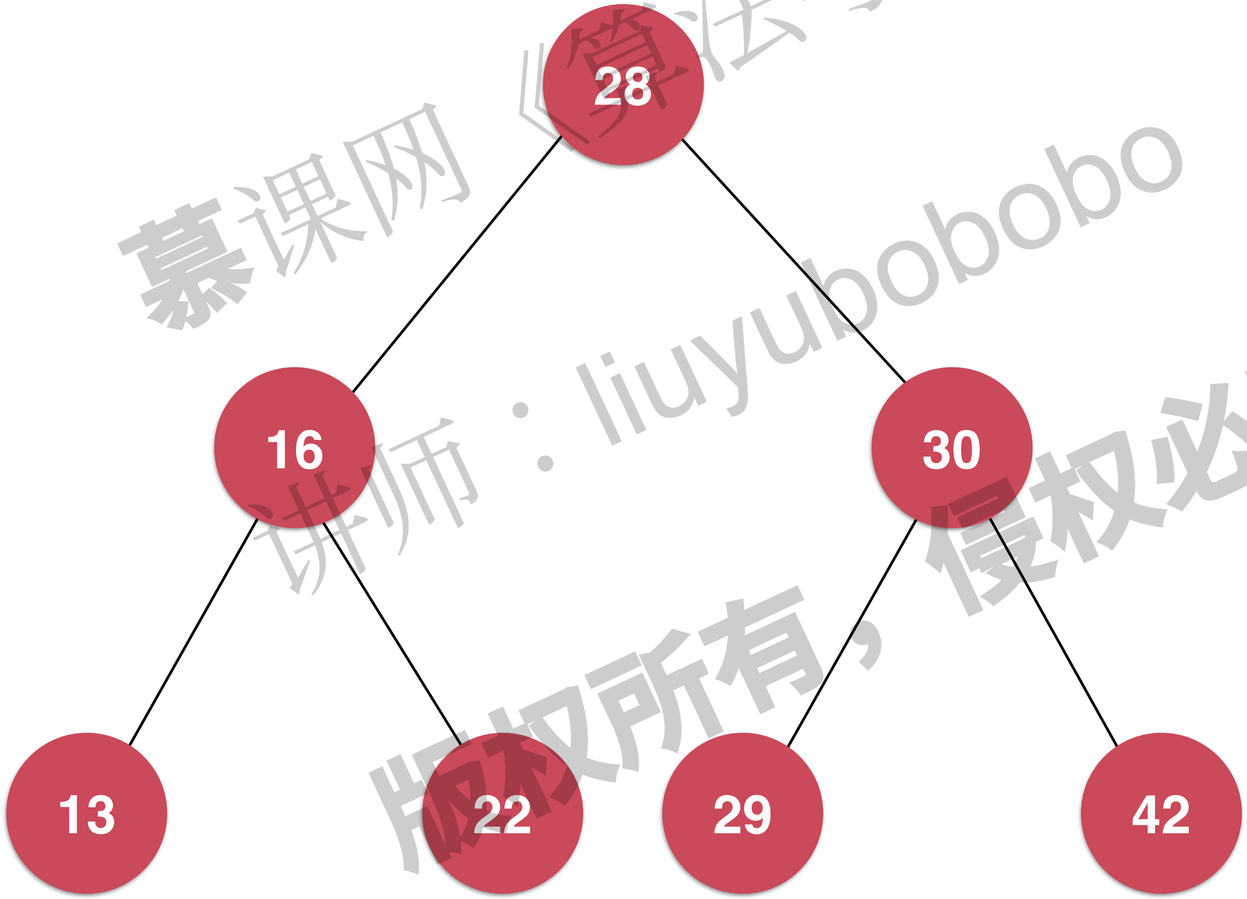


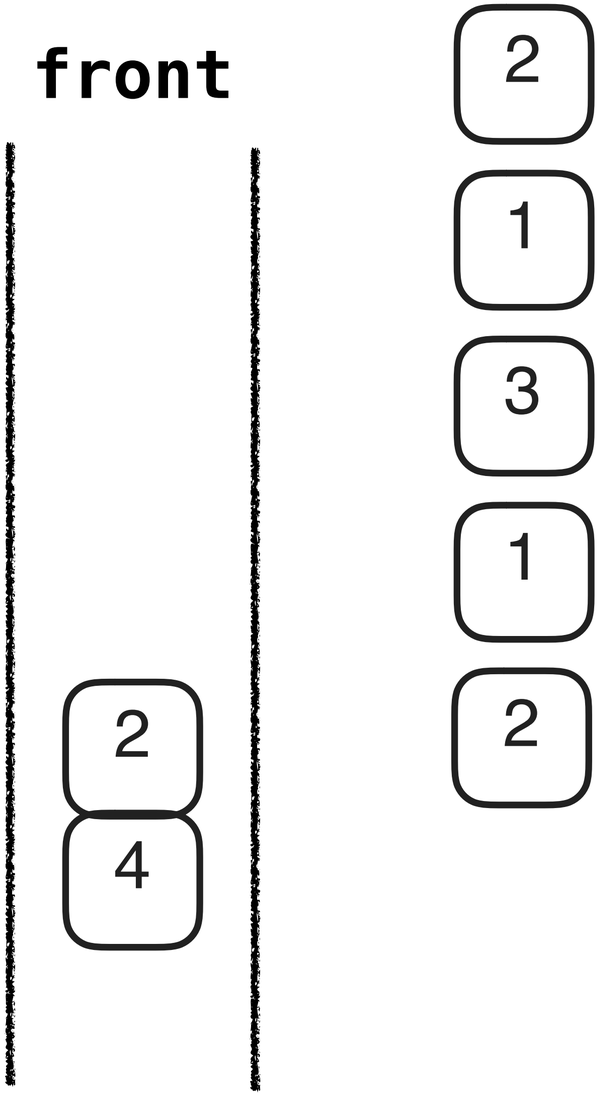
二分搜索树的广度优先遍历（层序）



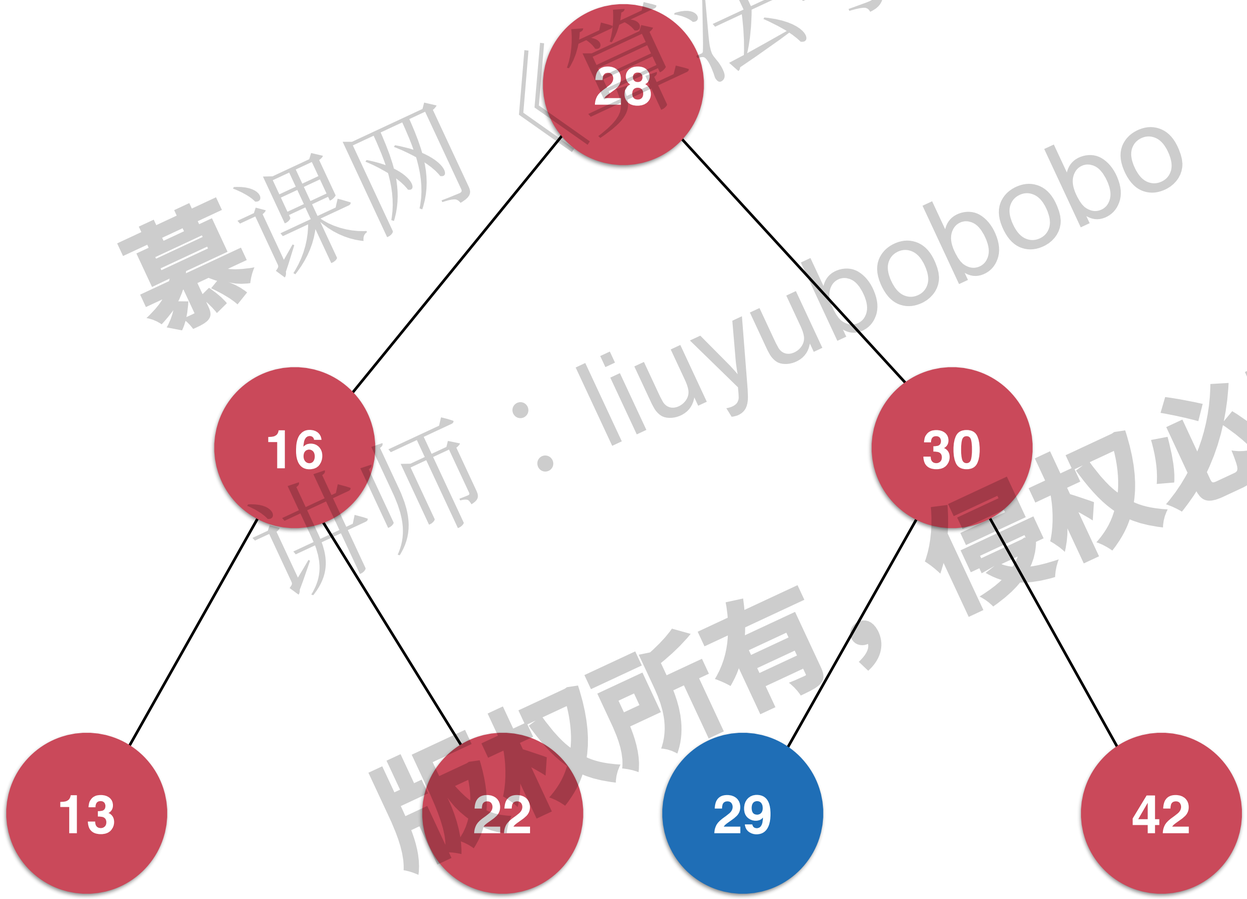


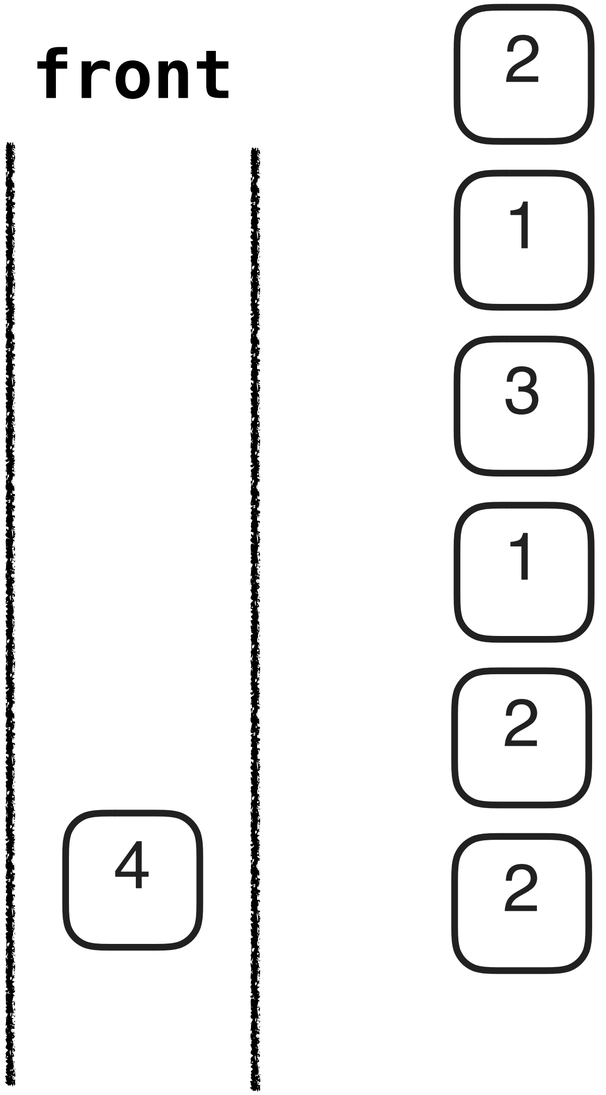
二分搜索树的广度优先遍历（层序）



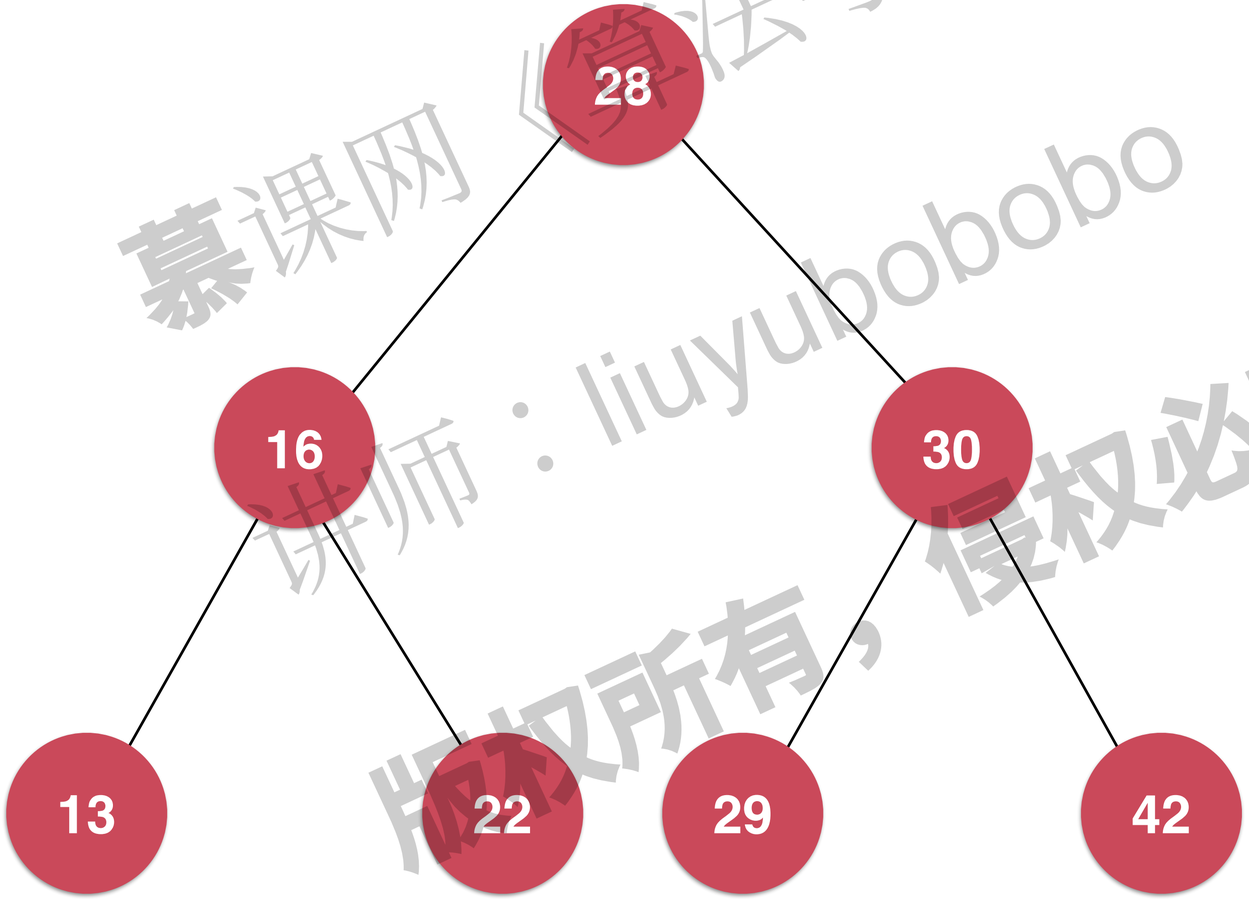


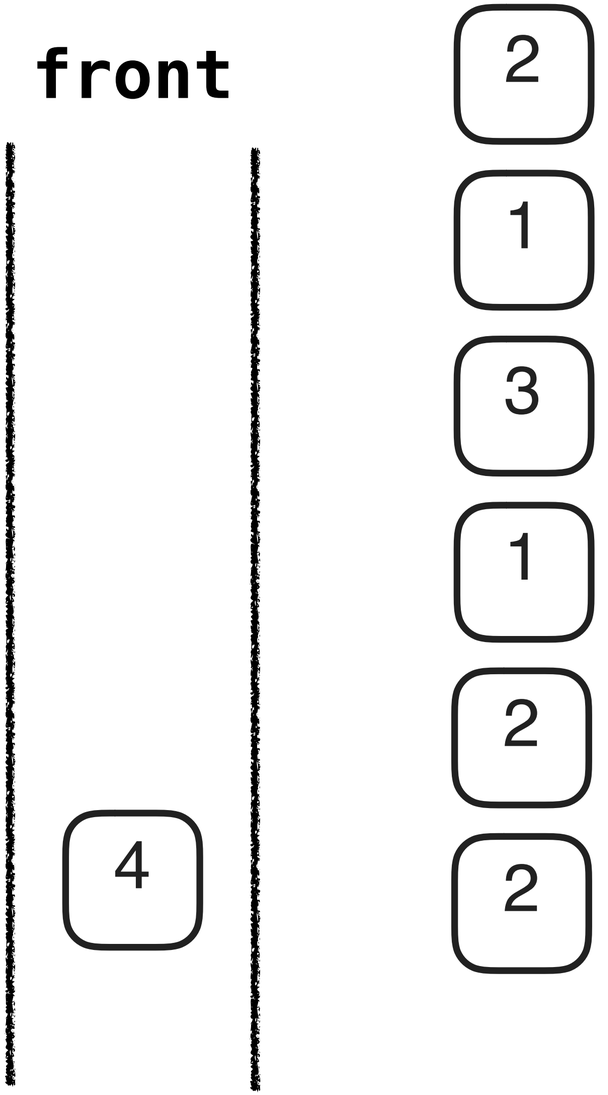
二分搜索树的广度优先遍历（层序）



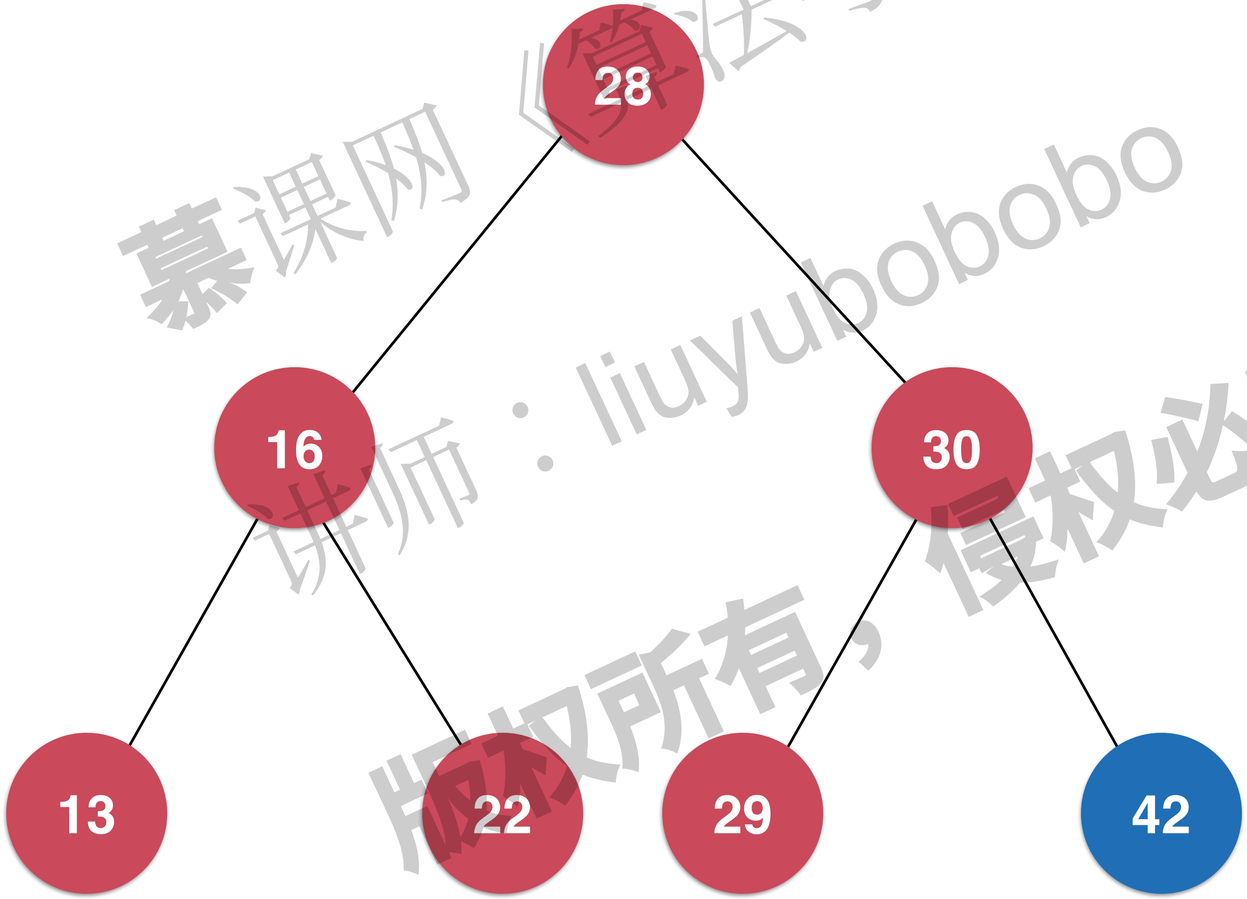


二分搜索树的广度优先遍历（层序）





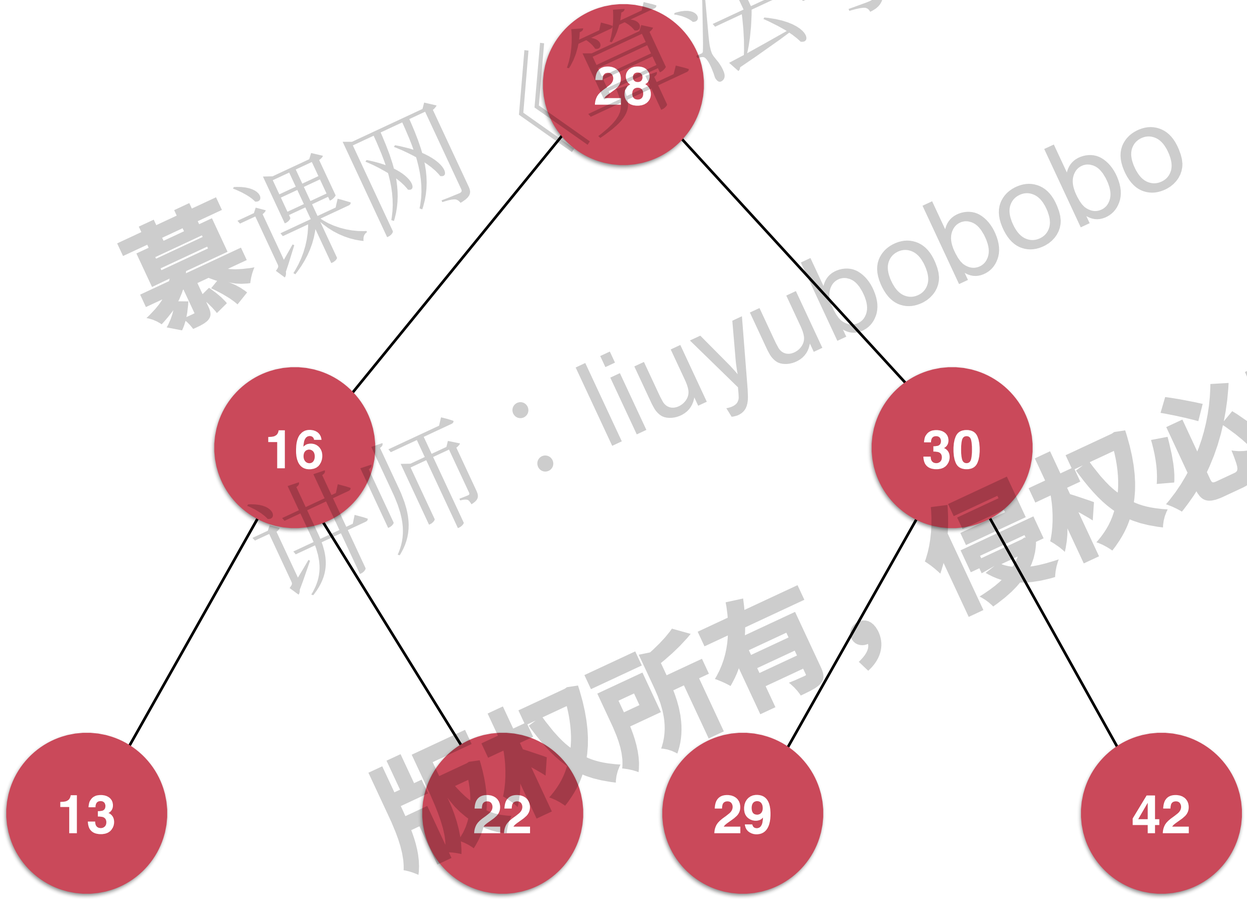
二分搜索树的广度优先遍历（层序）



front

2 1 3 1 2 2 4

二分搜索树的广度优先遍历（层序）

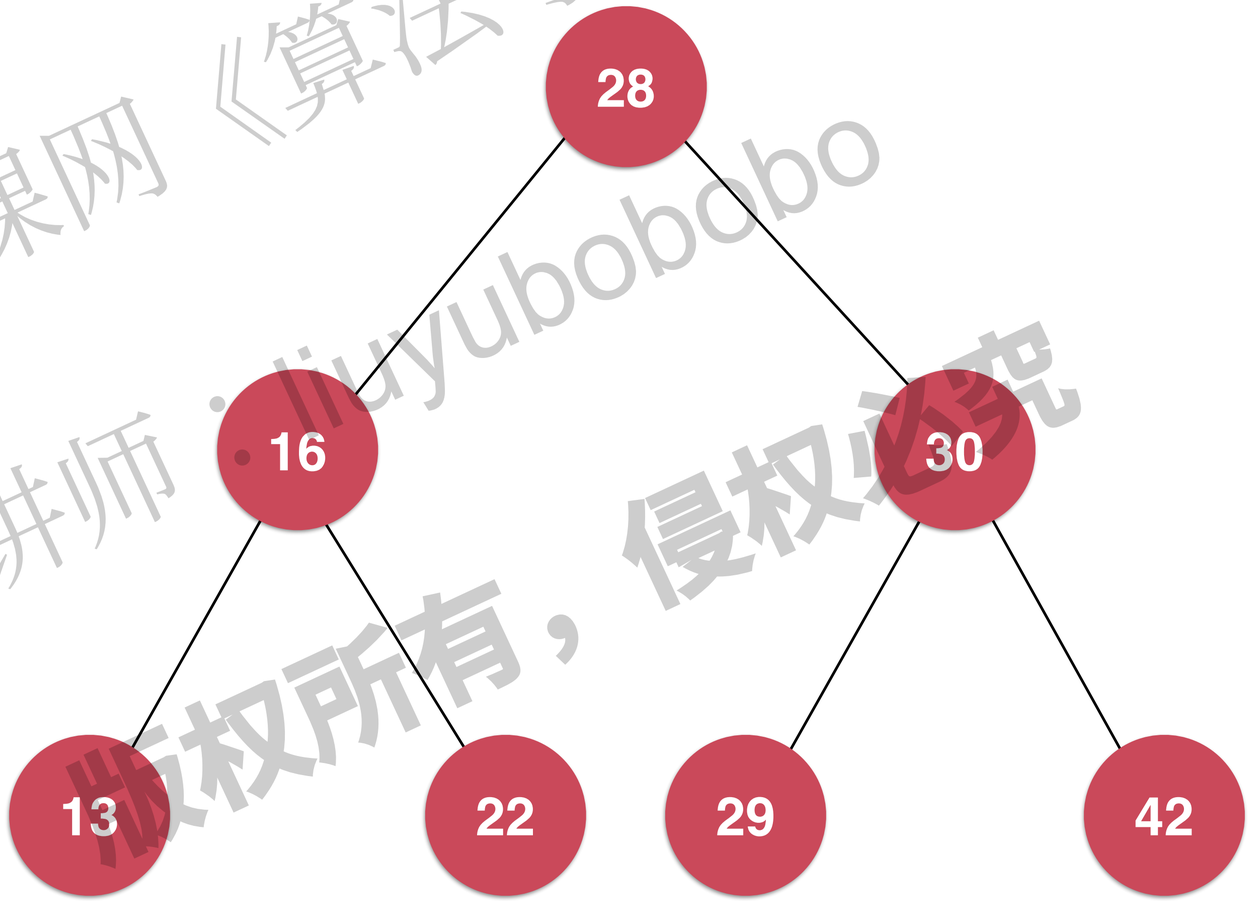


front

2 1 3 1 2 2 4

操作：二分搜索树的层序遍历

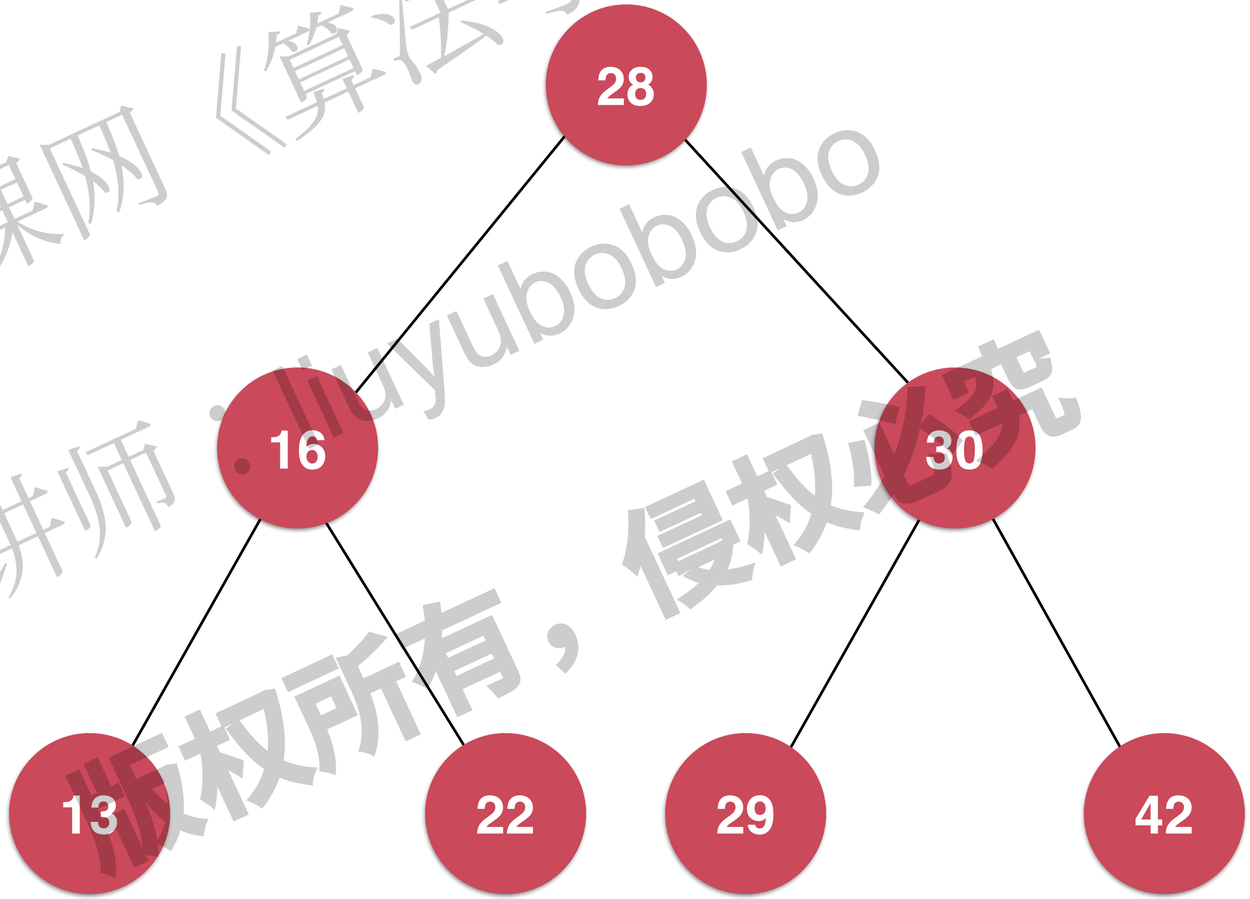
二分搜索树的遍历 - O(n)



二分搜索树 删除节点

从最简单的，删除二分搜索树的最小值和最大值开始

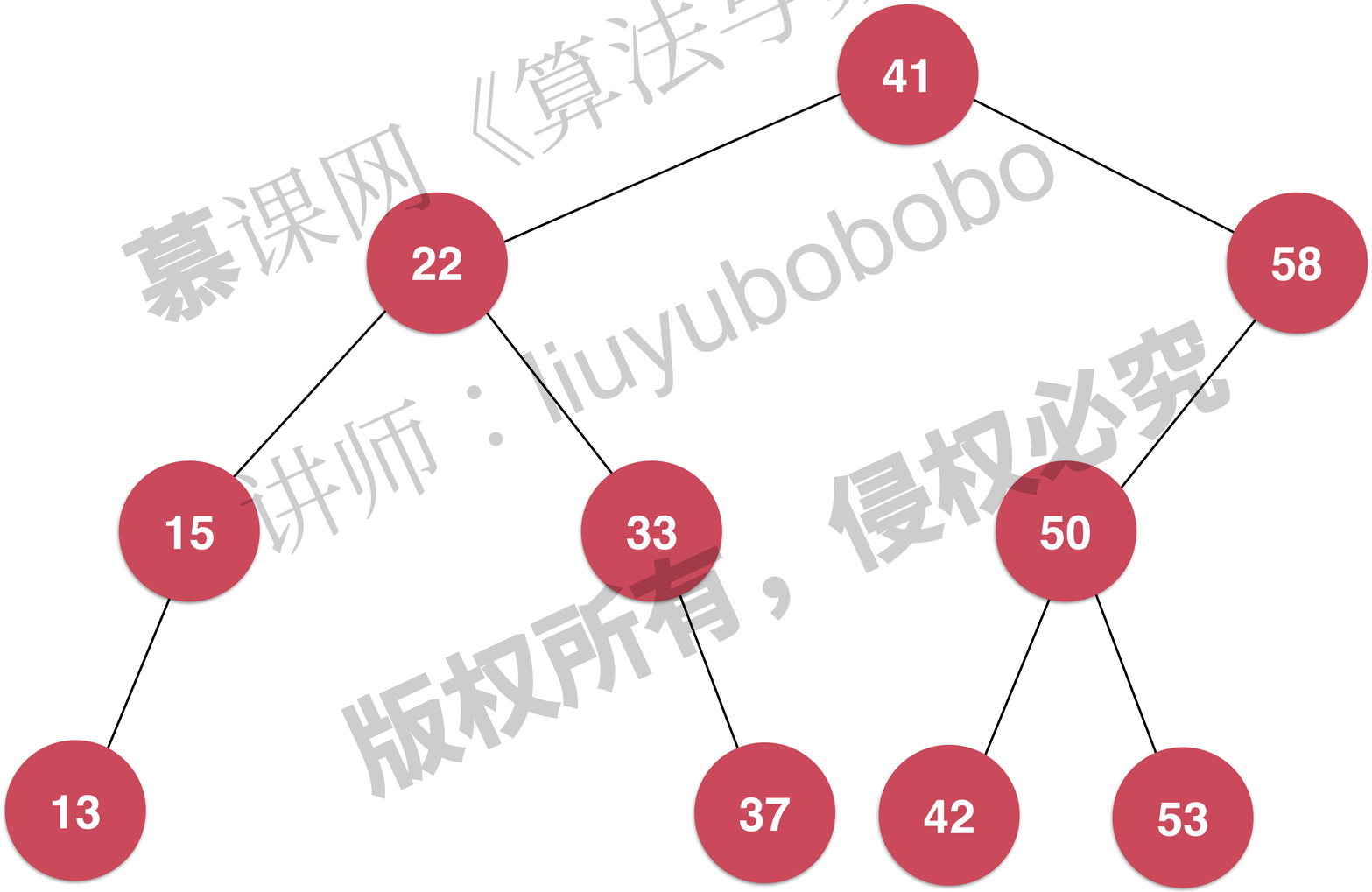
二分搜索树的最小值和最大值



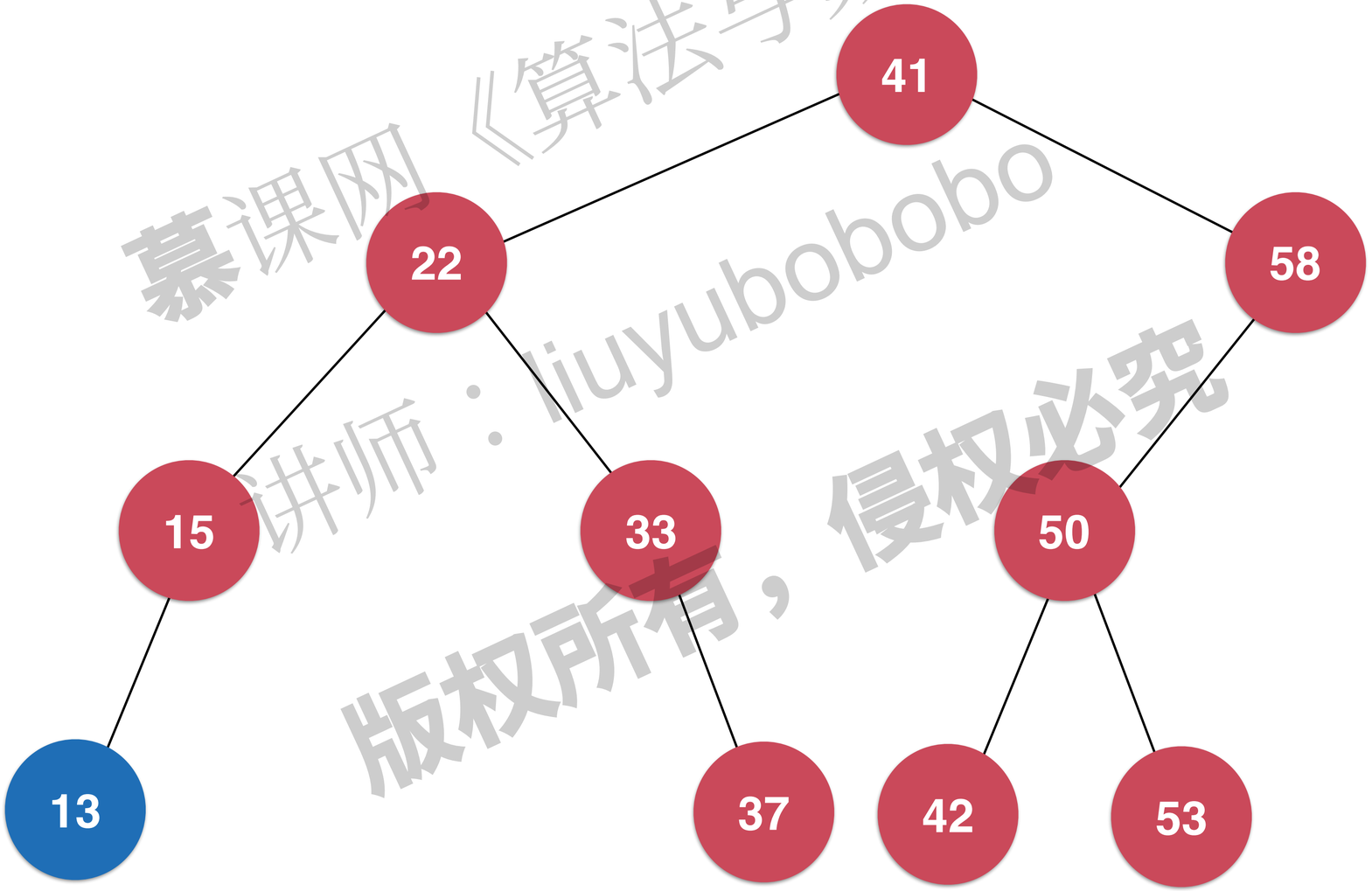
操作：求二分搜索树的最小值和最大值

练习：求二分搜索树的最小值和最大值的非递归实现

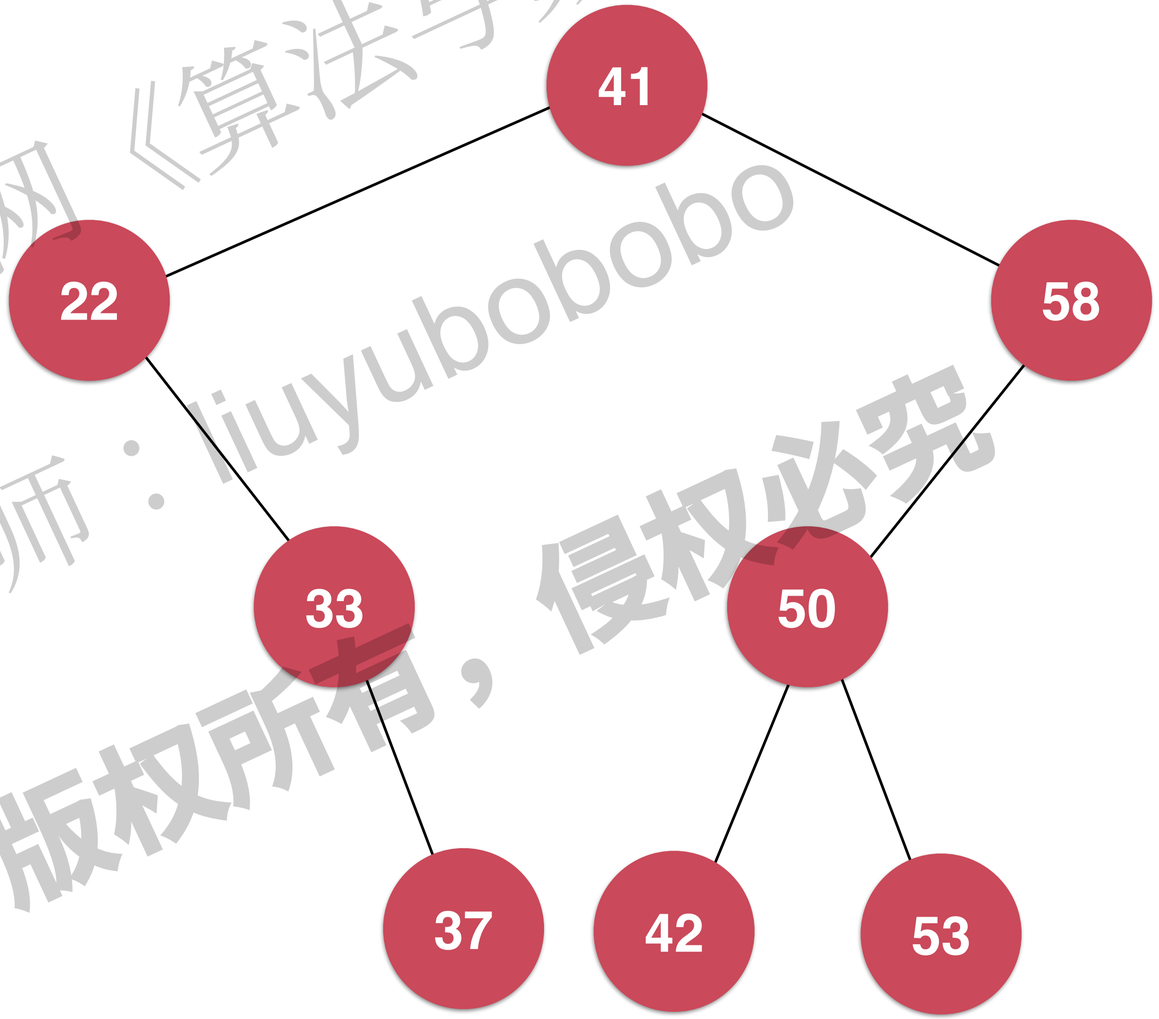
删除二分搜索树的最小值



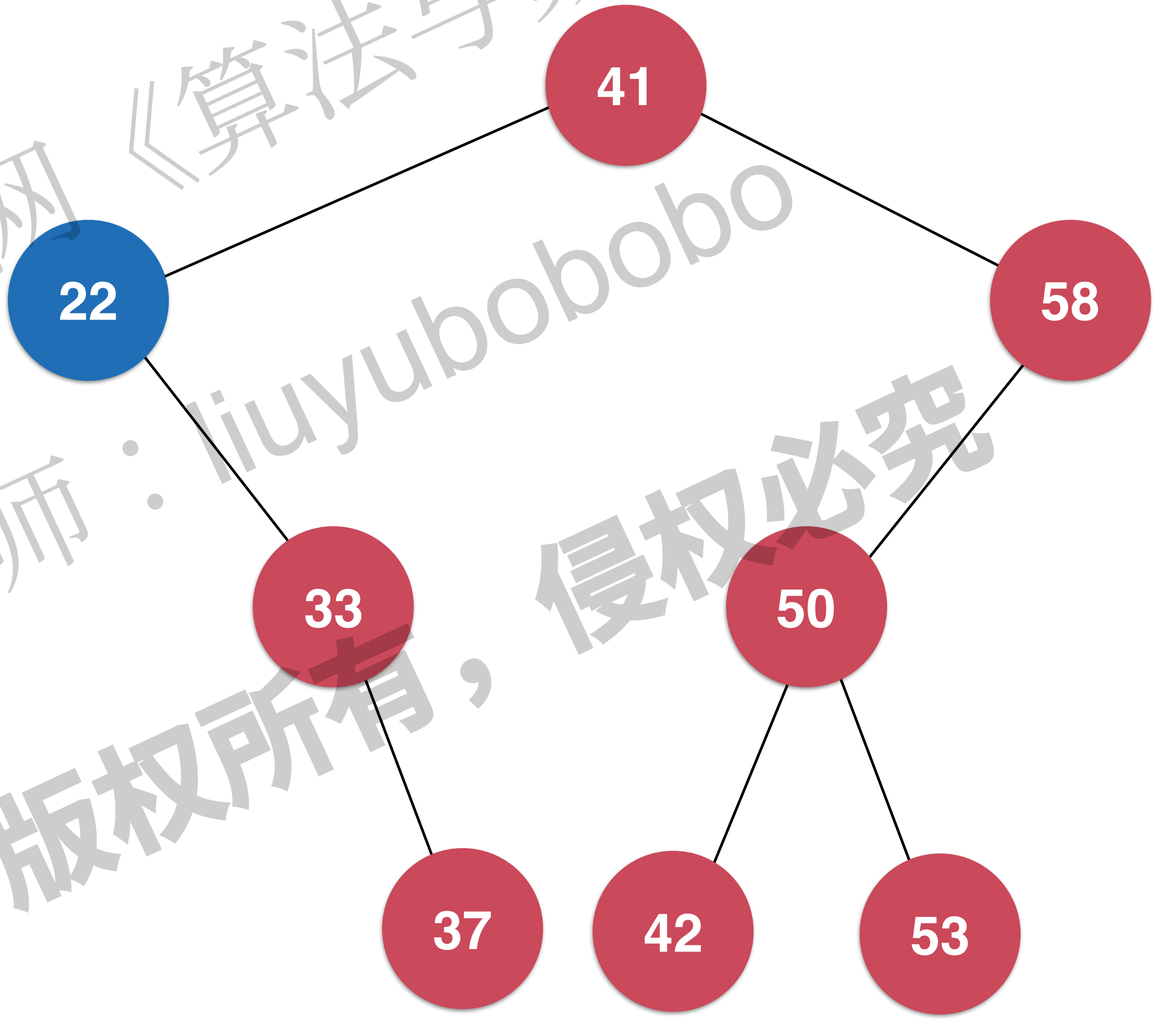
删除二分搜索树的最小值



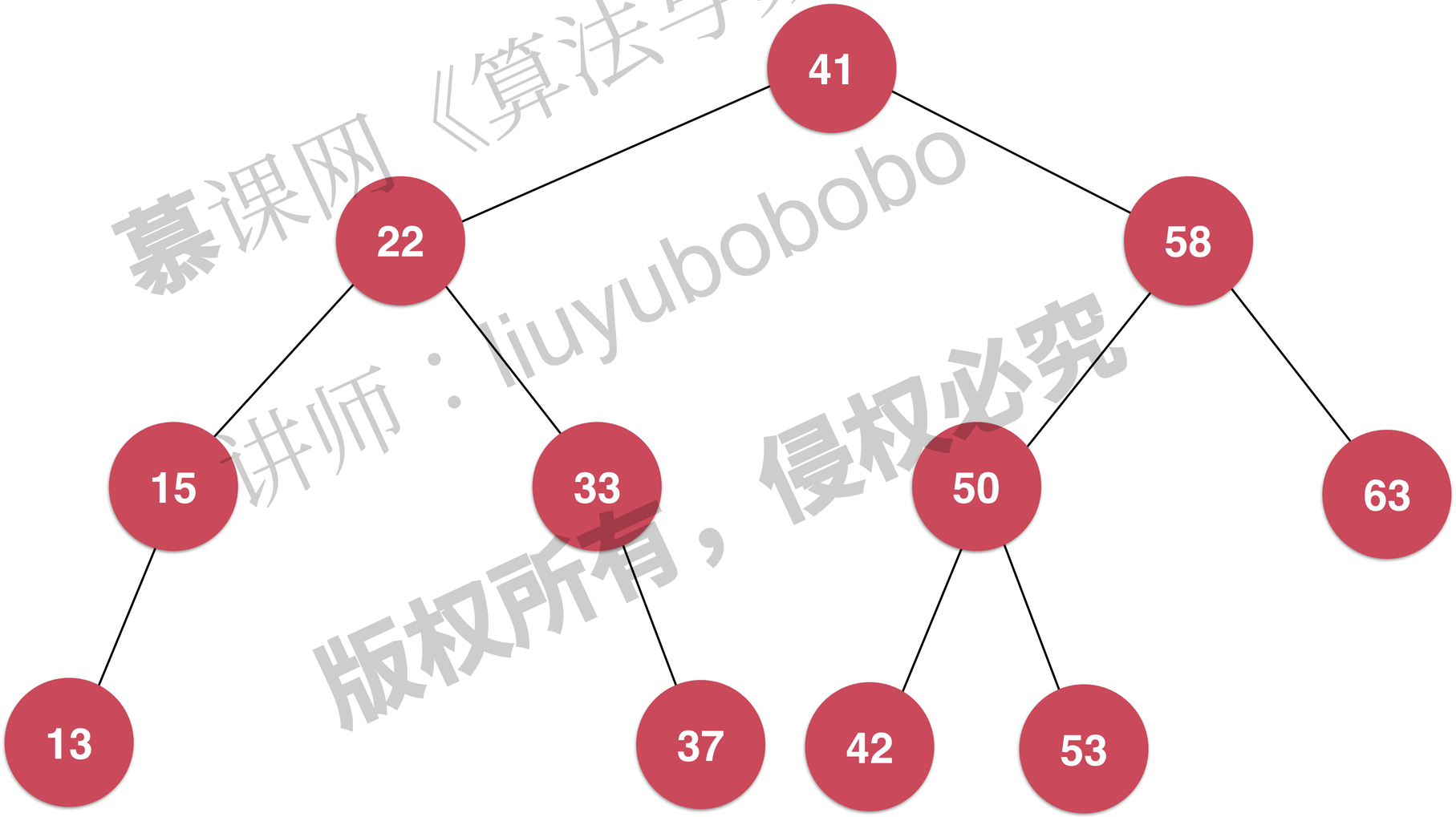
删除二分搜索树的最小值



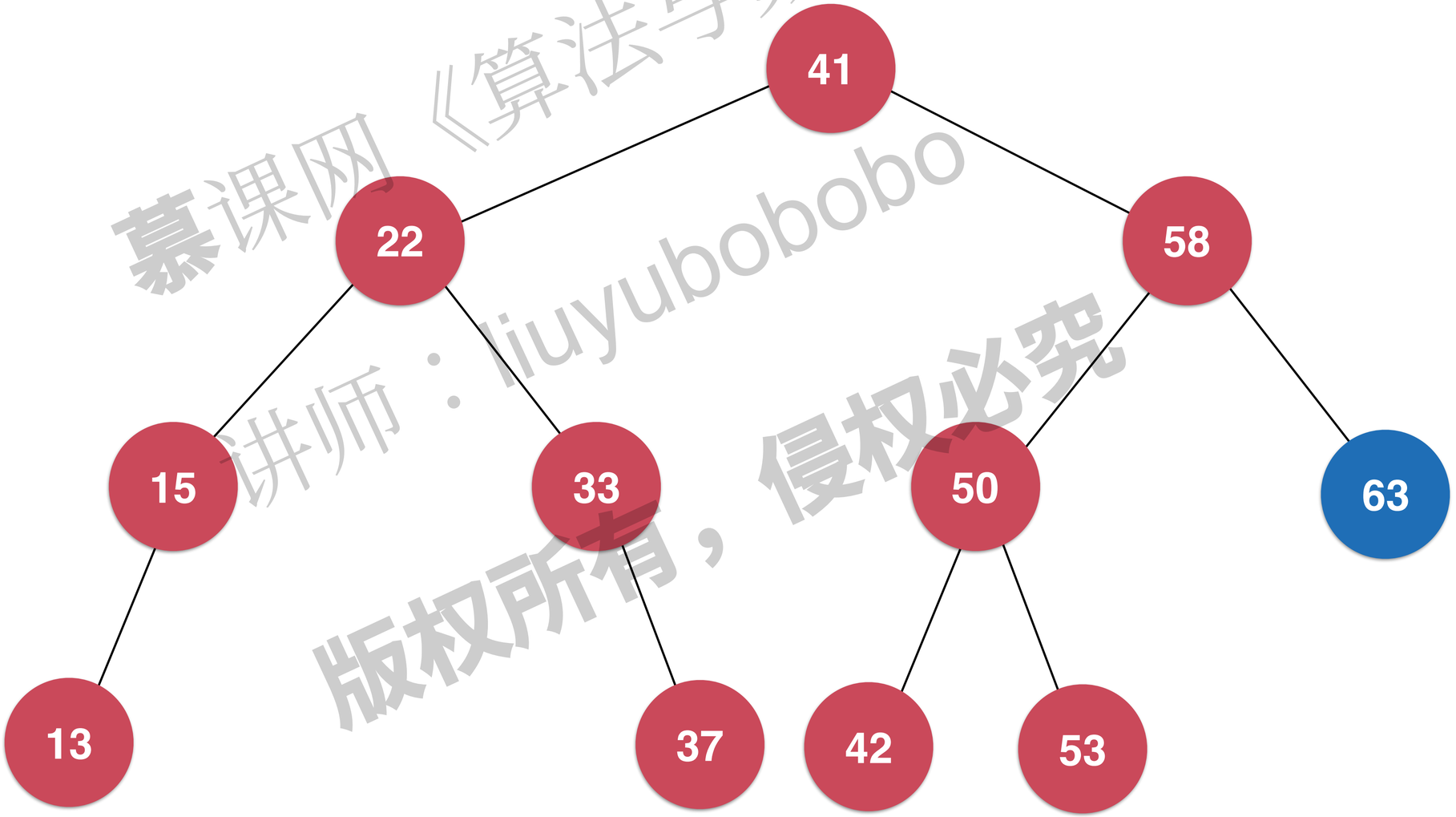
删除二分搜索树的最小值



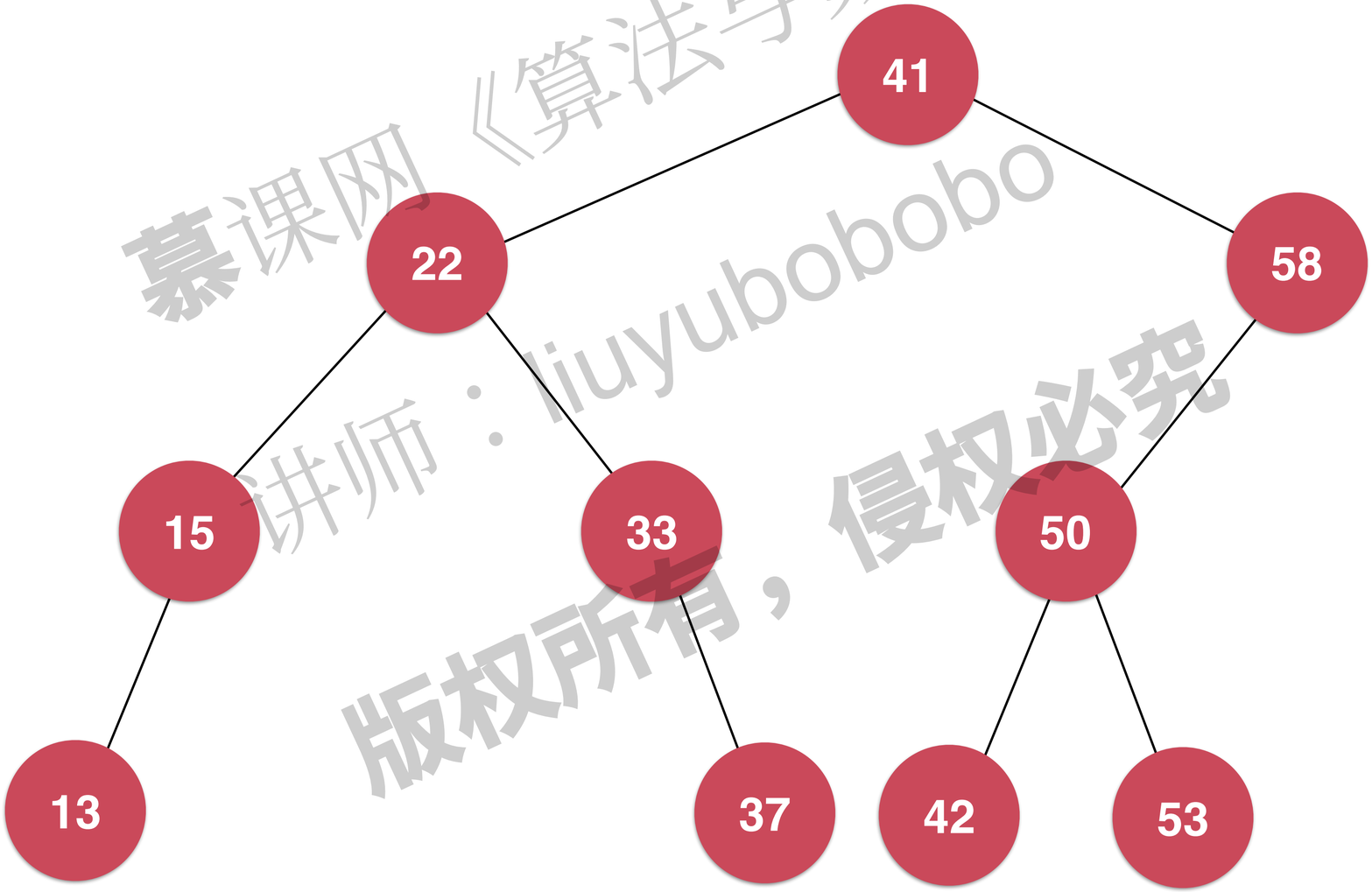
删除二分搜索树的最大值



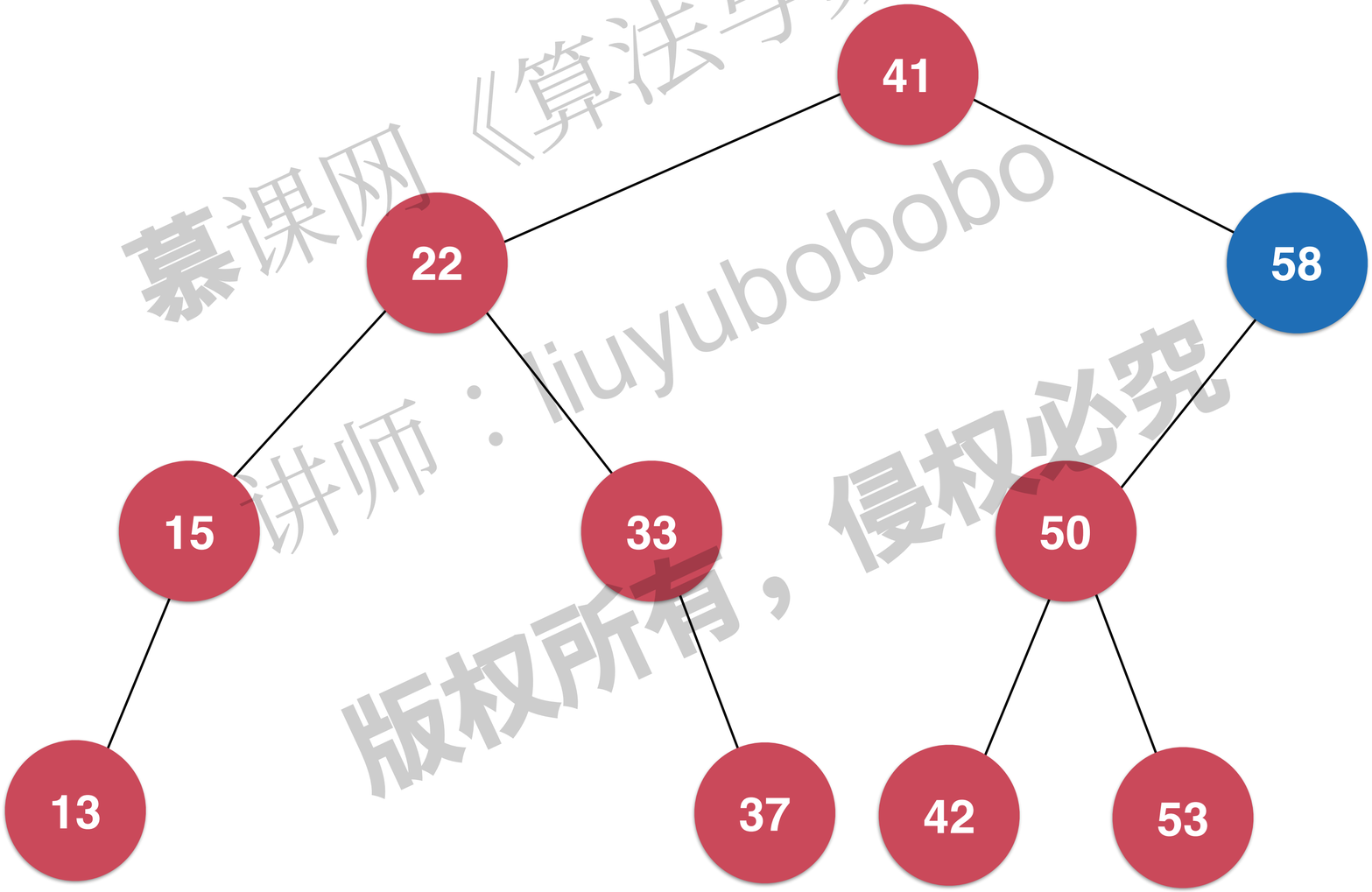
删除二分搜索树的最大值



删除二分搜索树的最大值



删除二分搜索树的最大值

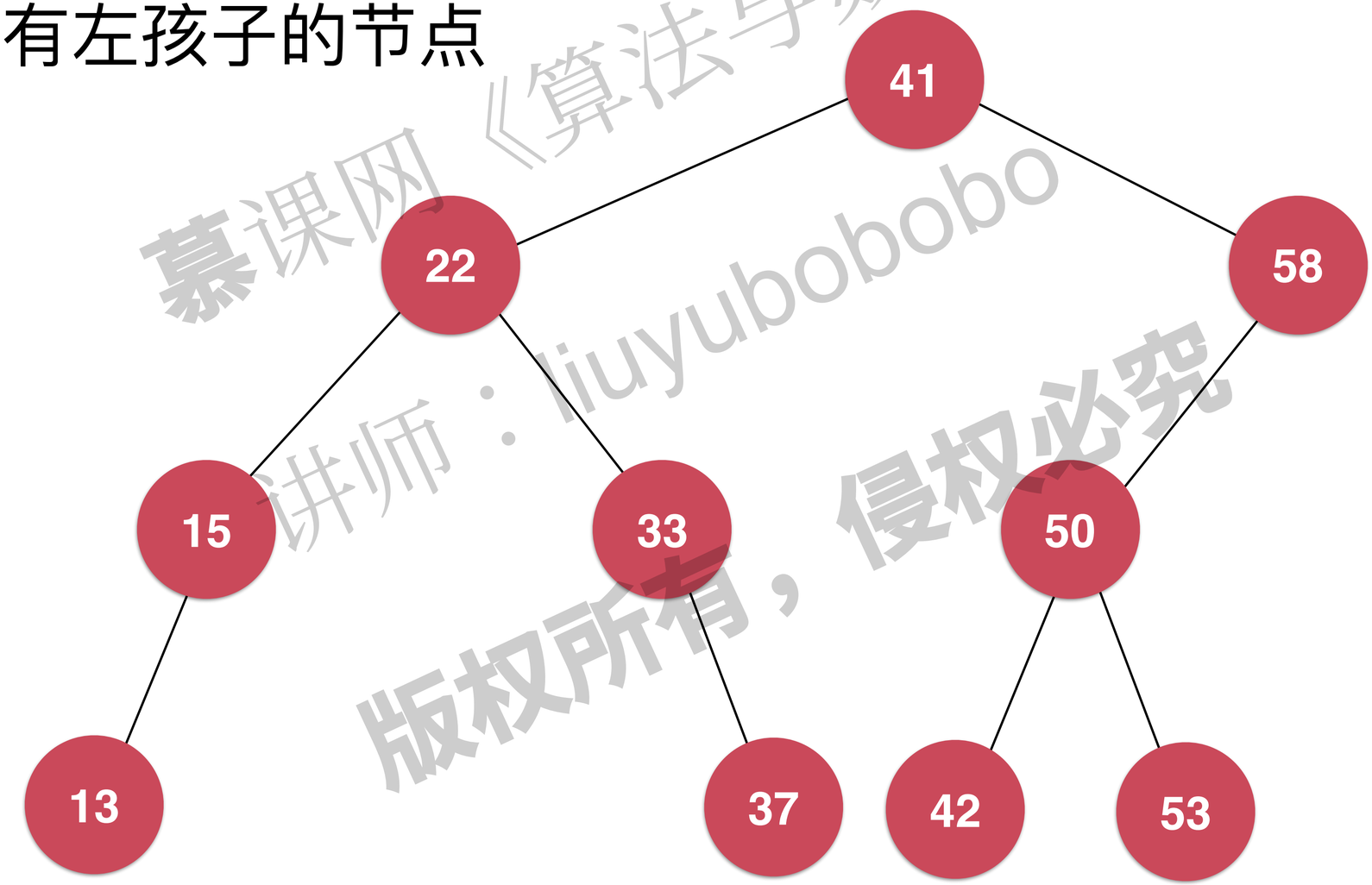


操作：删除二分搜索树的最小值和最大值

练习：删除二分搜索树的最小值和最大值的非递归实现

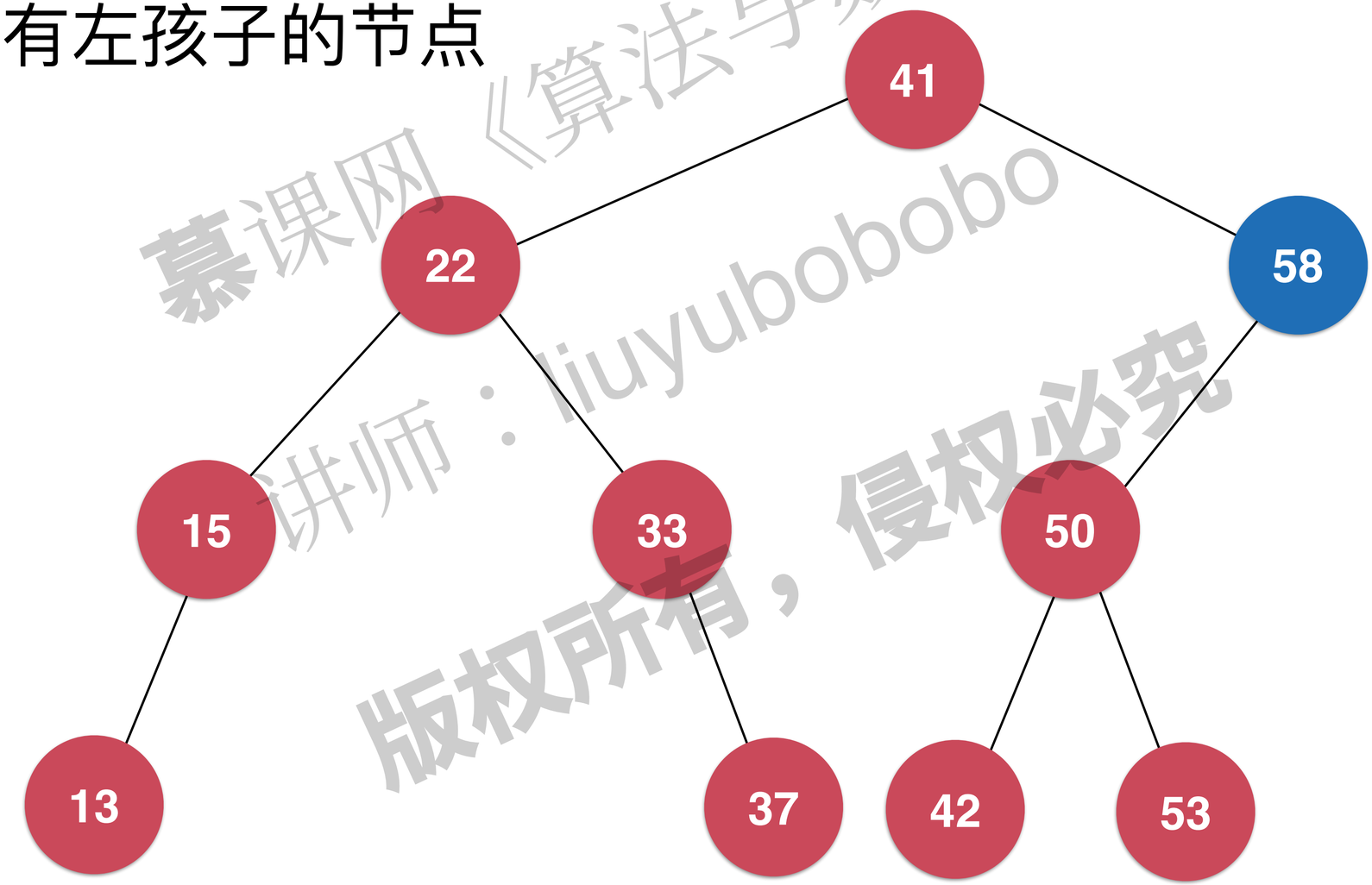
二分搜索树删除节点

删除只有左孩子的节点



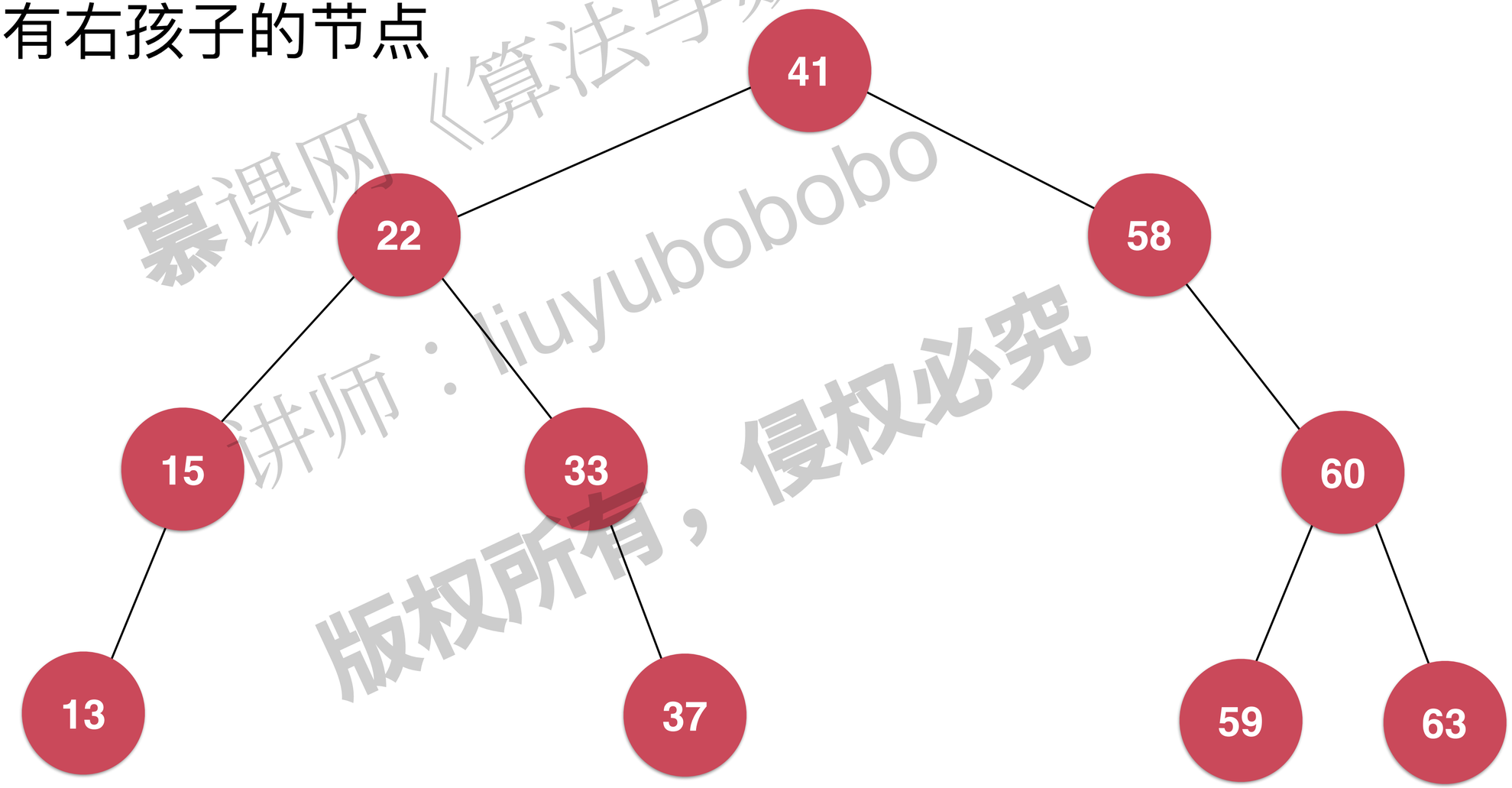
二分搜索树删除节点

删除只有左孩子的节点



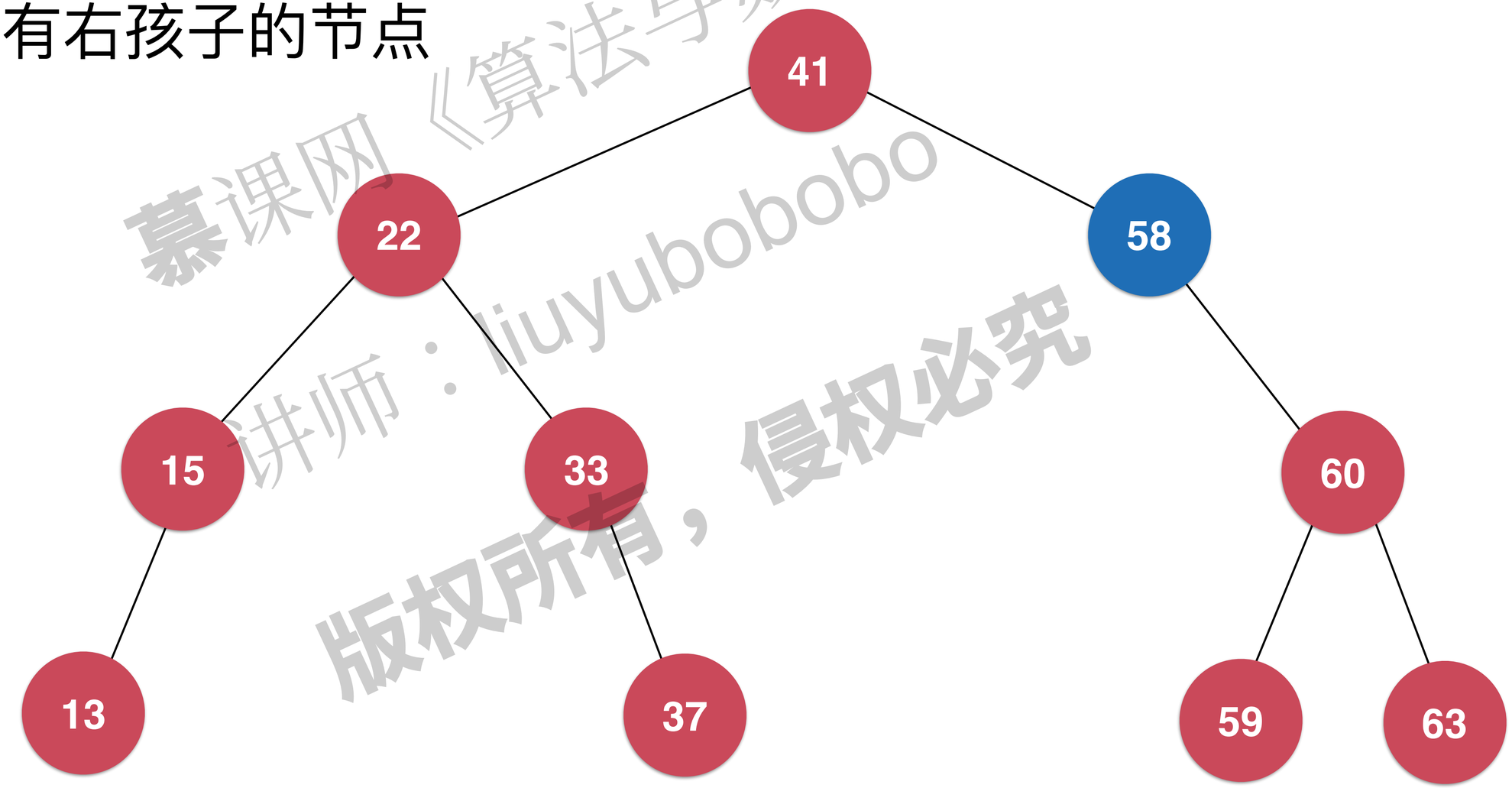
二分搜索树删除节点

删除只有右孩子的节点



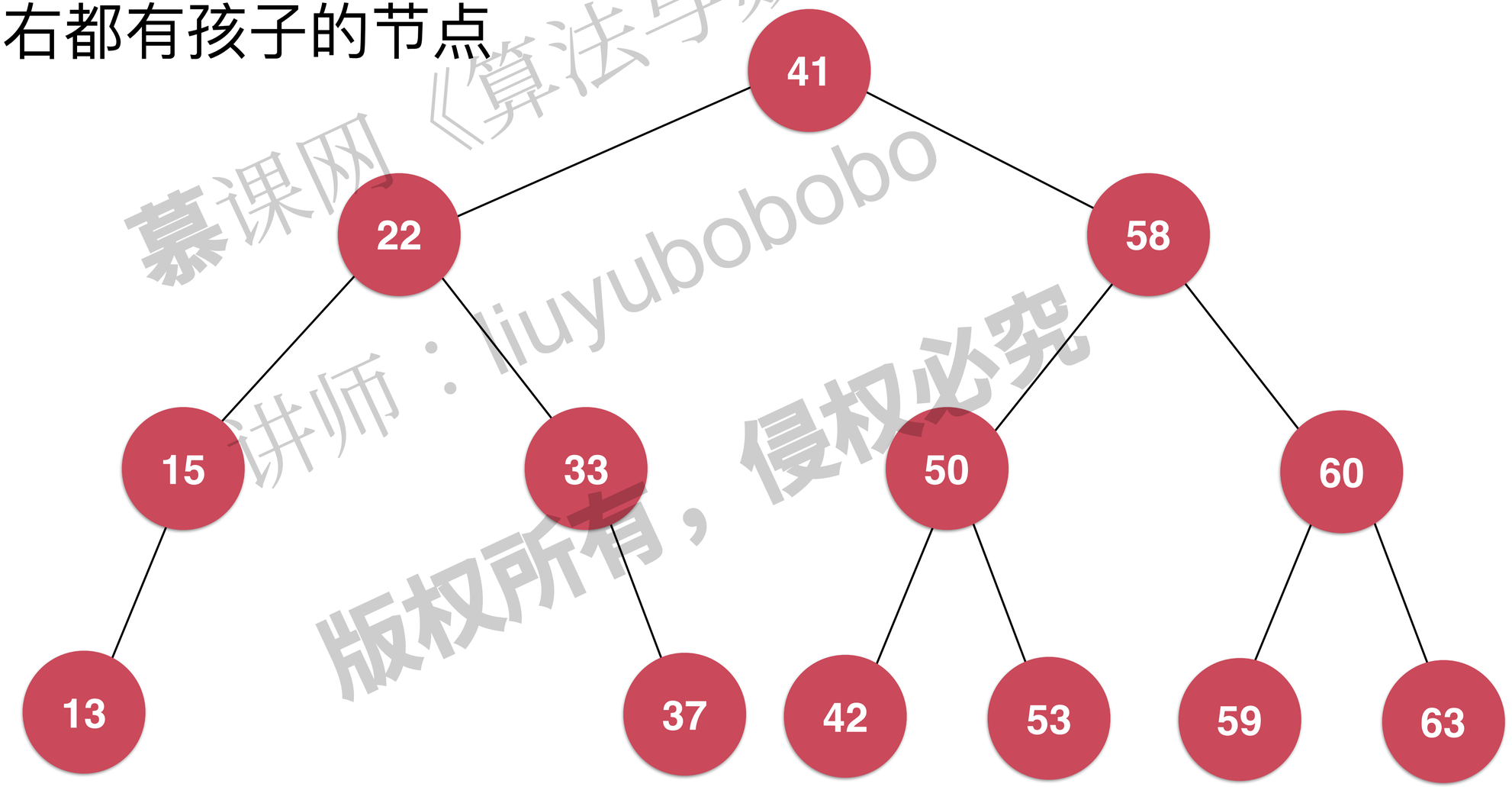
二分搜索树删除节点

删除只有右孩子的节点



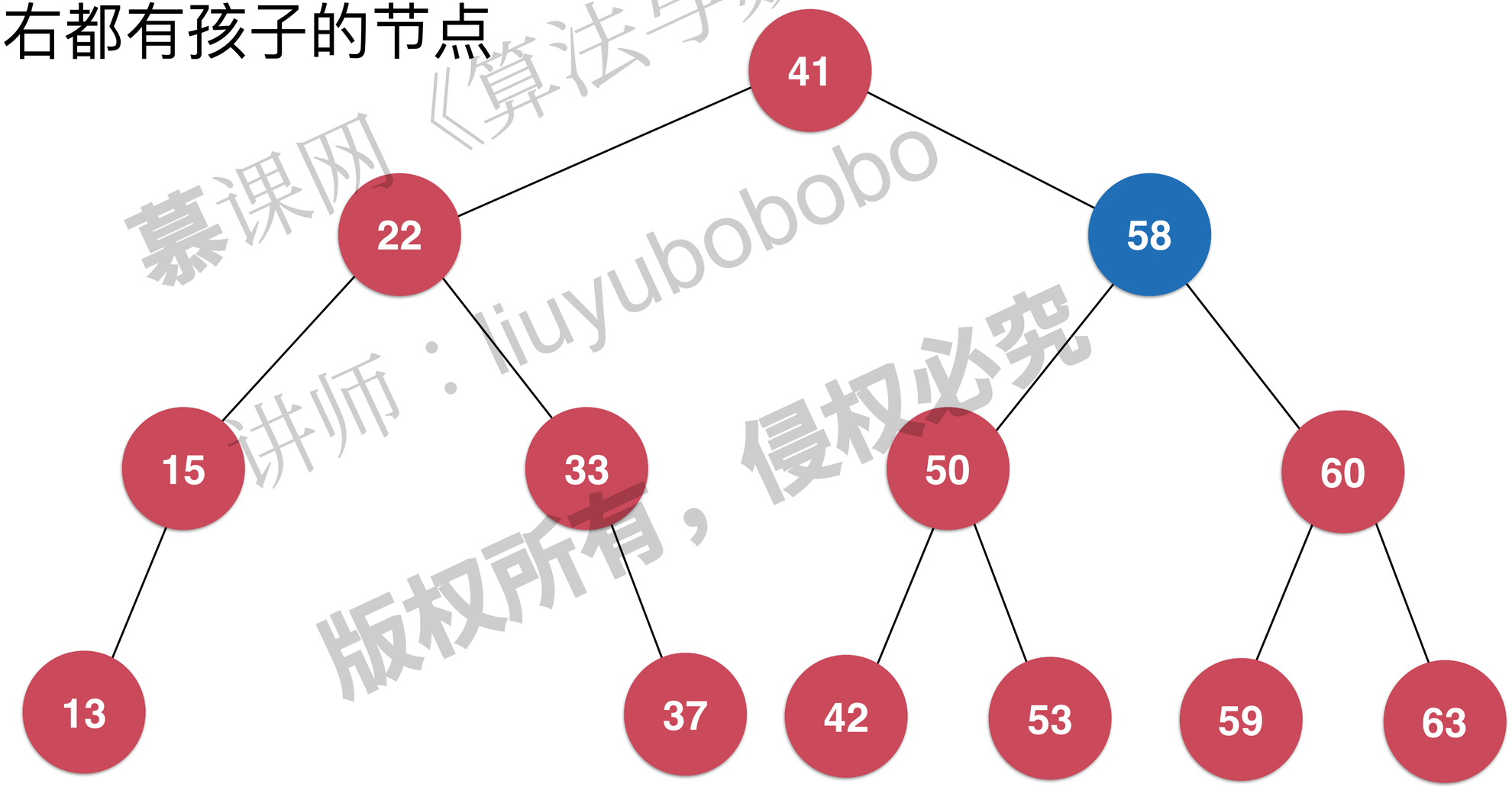
二分搜索树删除节点

删除左右都有孩子的节点



二分搜索树删除节点

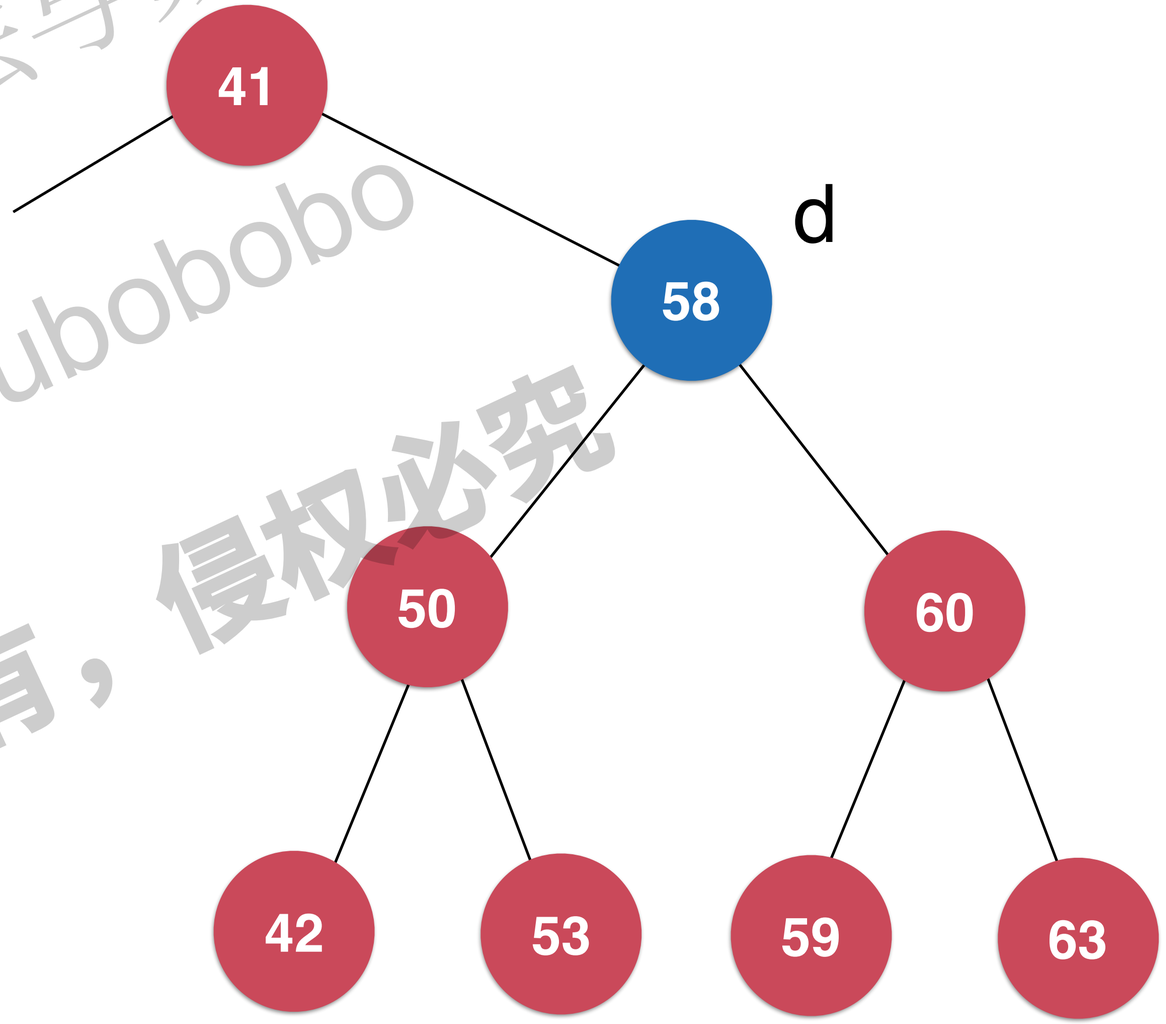
删除左右都有孩子的节点



1962年，Hibbard提出 - Hubbard Deletion

二分搜索树删除节点

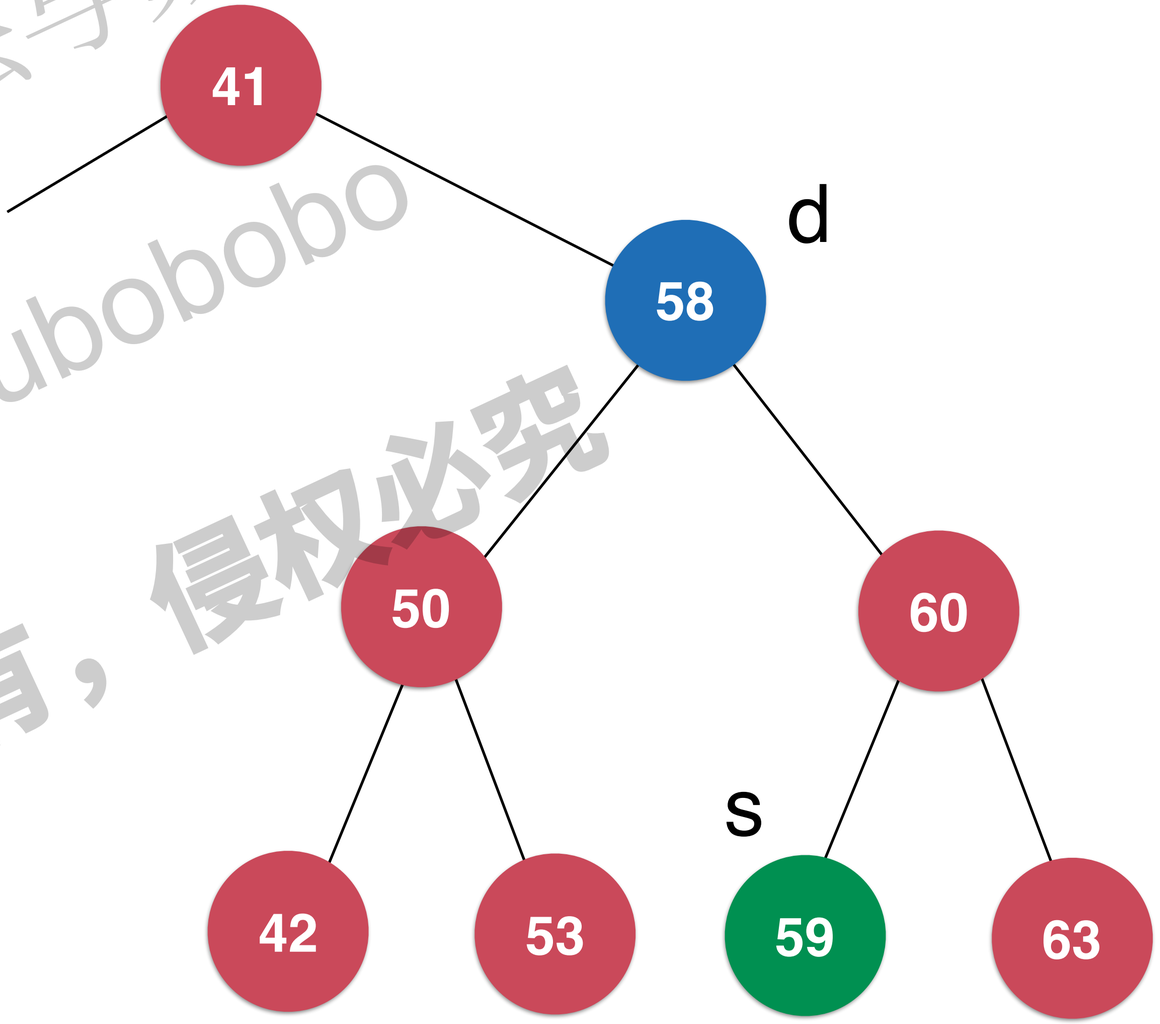
删除左右都有孩子的节点 d



二分搜索树删除节点

删除左右都有孩子的节点 d

找到 s = min(d->right)

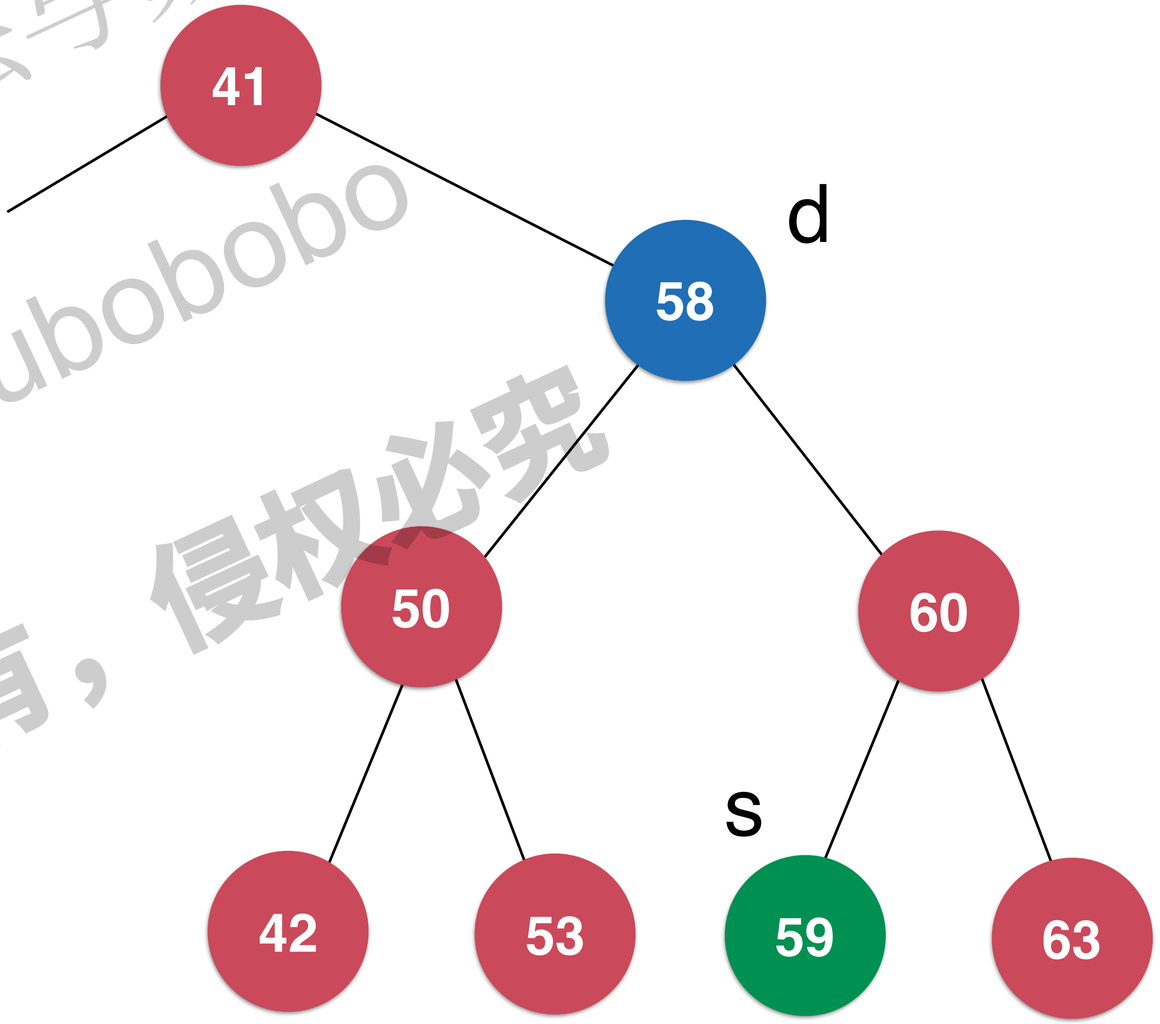


二分搜索树删除节点

删除左右都有孩子的节点 d

找到 s = min(d->right)

s 是 d 的后继



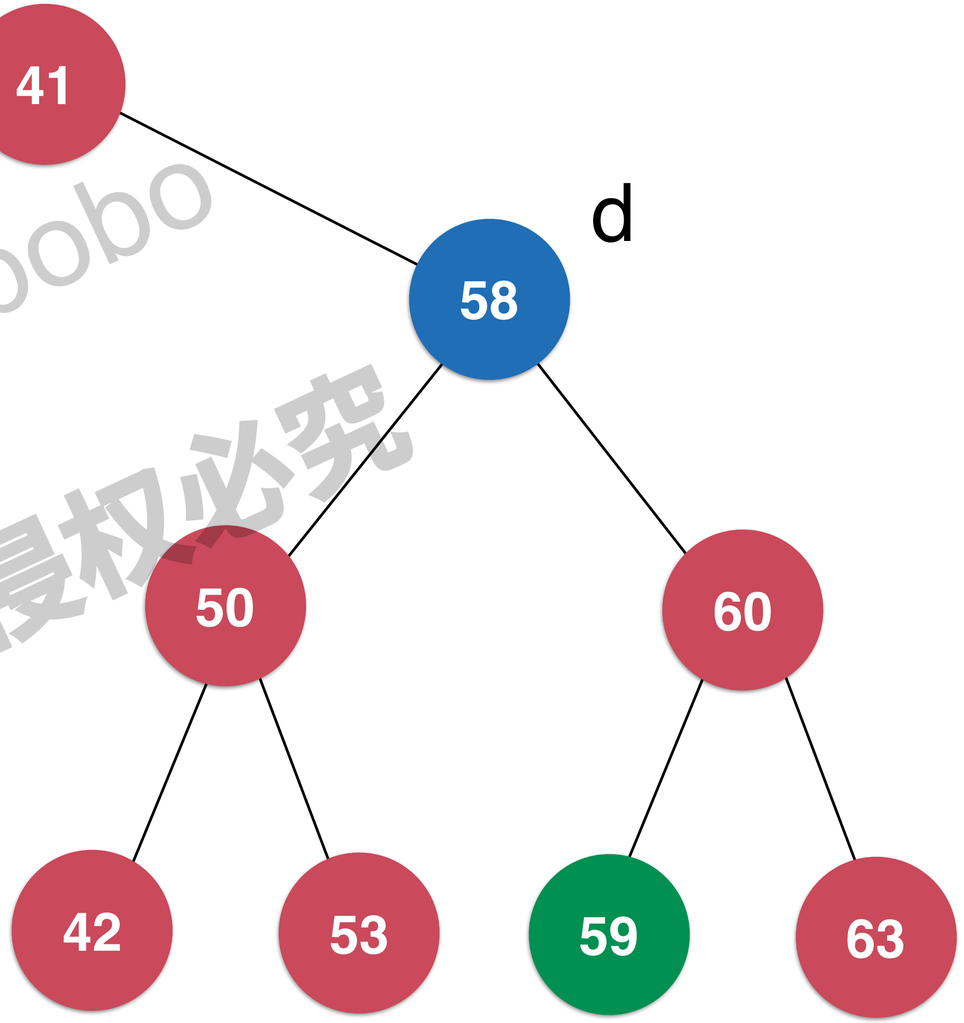
二分搜索树删除节点

删除左右都有孩子的节点 d

找到 s = min(d->right)

s 是 d 的后继

s->right = delMin(d->right)



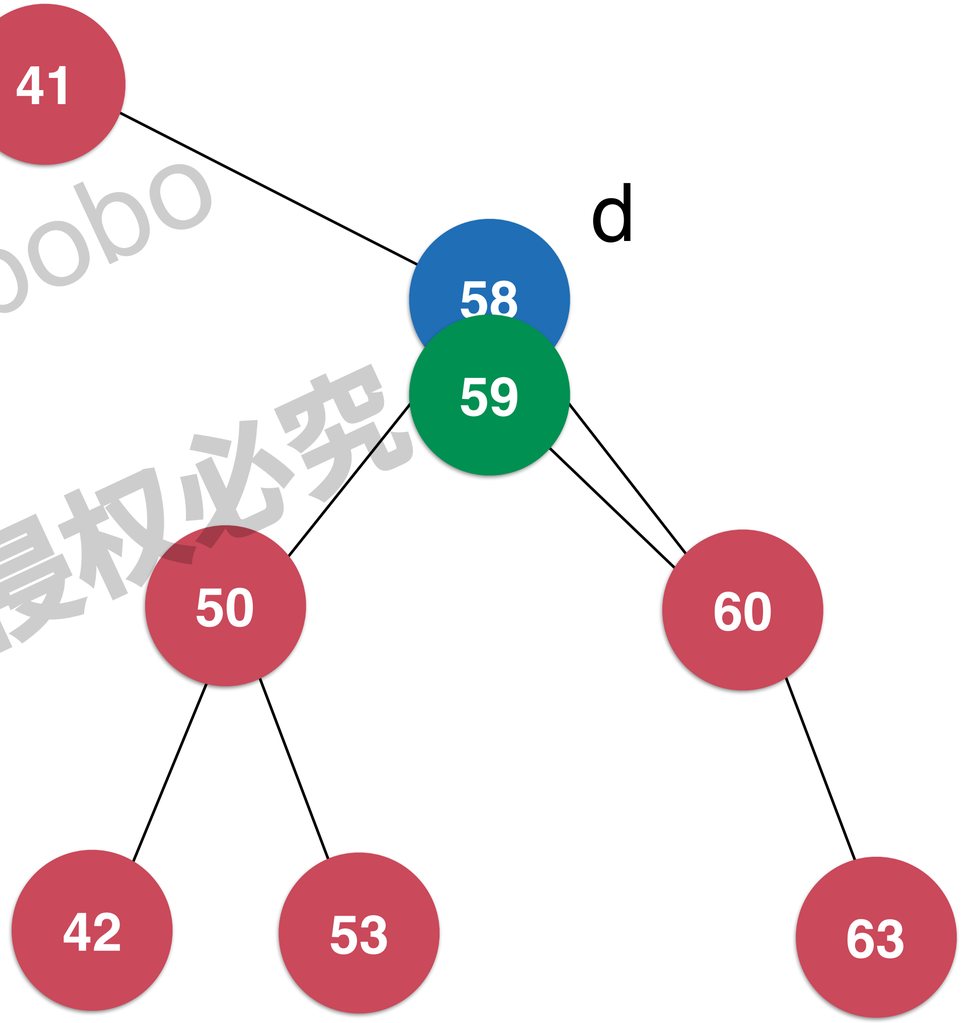
二分搜索树删除节点

删除左右都有孩子的节点 d

找到 s = min(d->right)

s 是 d 的后继

s->right = delMin(d->right)



二分搜索树删除节点

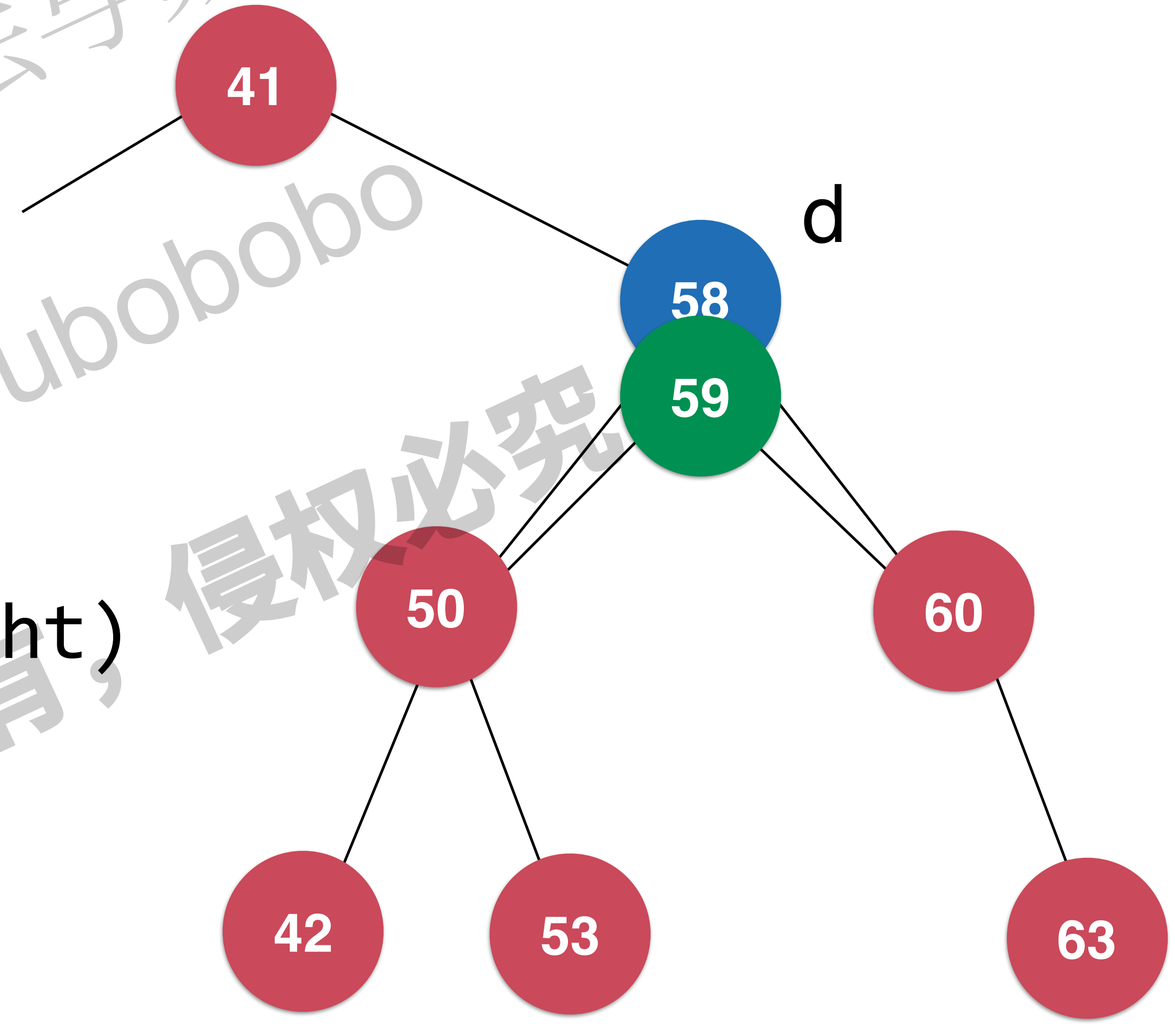
删除左右都有孩子的节点 d

找到 s = min(d->right)

s 是 d 的后继

s->right = delMin(d->right)

s->left = d->left



二分搜索树删除节点

删除左右都有孩子的节点 d

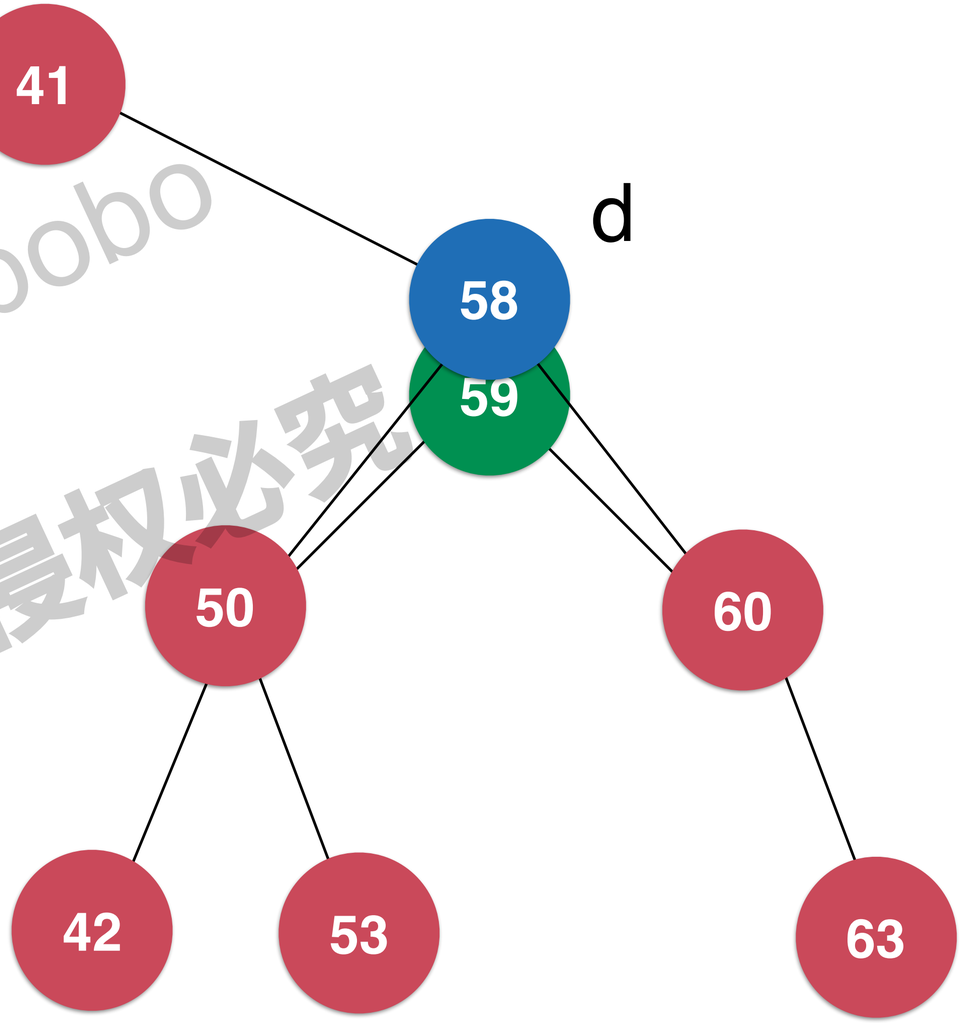
找到 s = min(d->right)

s 是 d 的后继

s->right = delMin(d->right)

s->left = d->left

删除d，s是新的子树的根



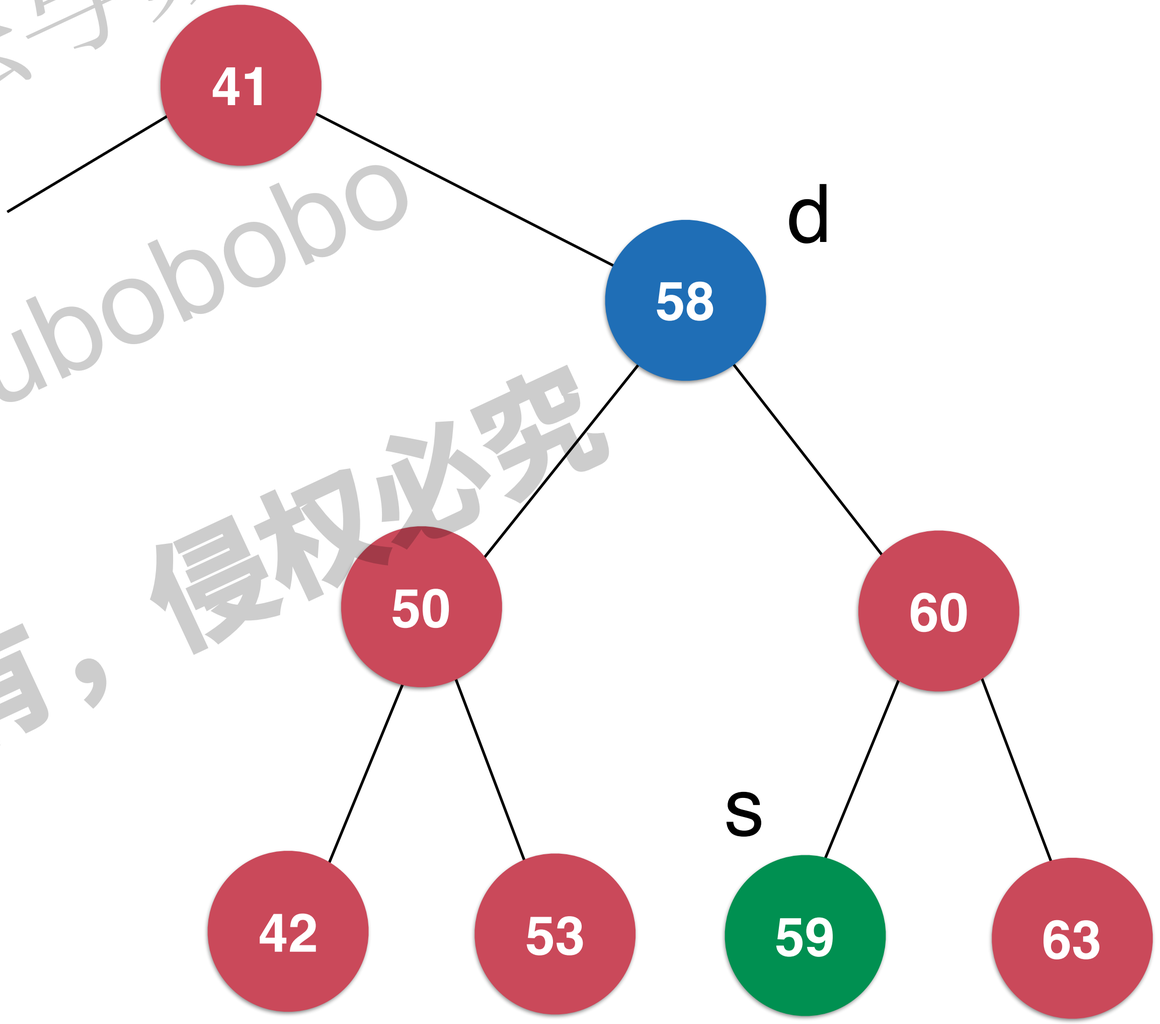
操作：删除二分搜索树的任意一个节点

二分搜索树删除节点

删除左右都有孩子的节点 d

找到 s = min(d->right)

s 是 d 的后继

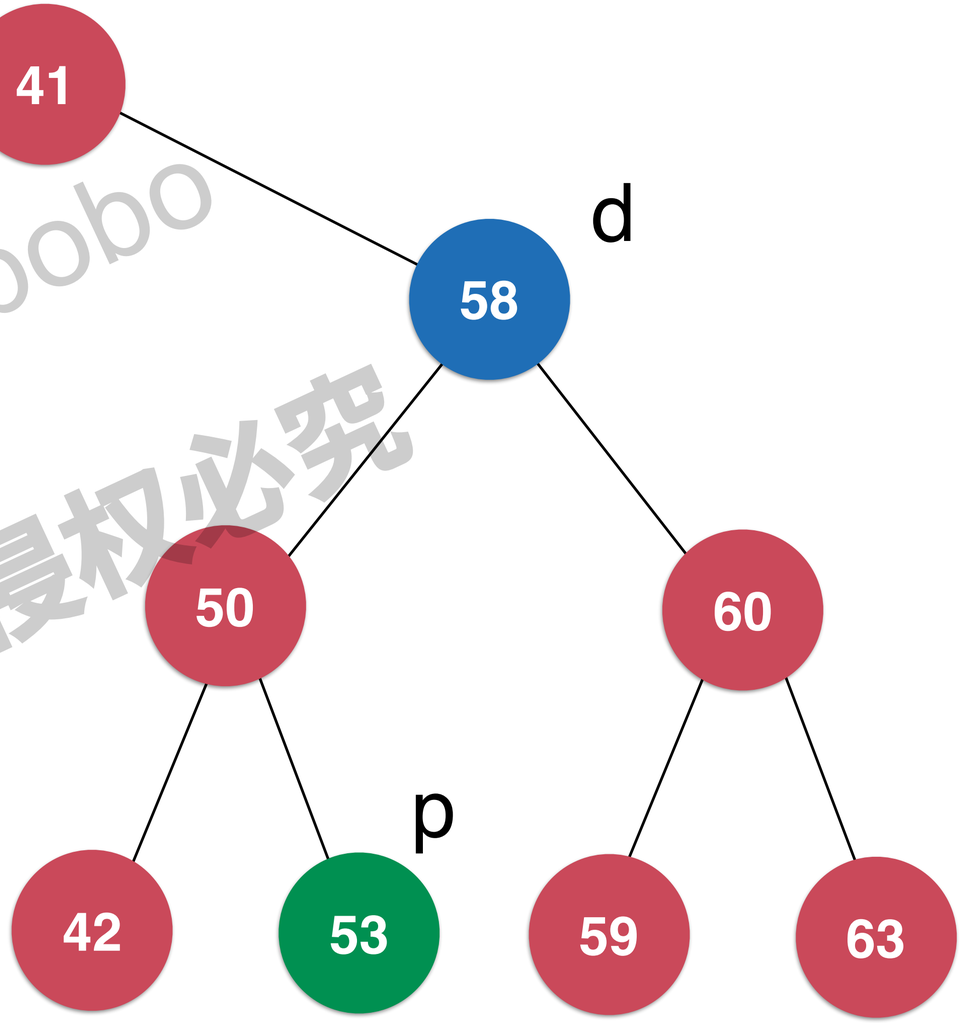


二分搜索树删除节点

删除左右都有孩子的节点 d

找到 p = max(d->left)

p 是 d 的前驱



练习：使用d的前驱p代替d节点的hibbard deletion

删除二分搜索树的任意一个节点 时间复杂度 O(logn)

二分搜索树的顺序性

二分搜索树的顺序性

minimum，maximum

二分搜索树的顺序性

successor，predecessor

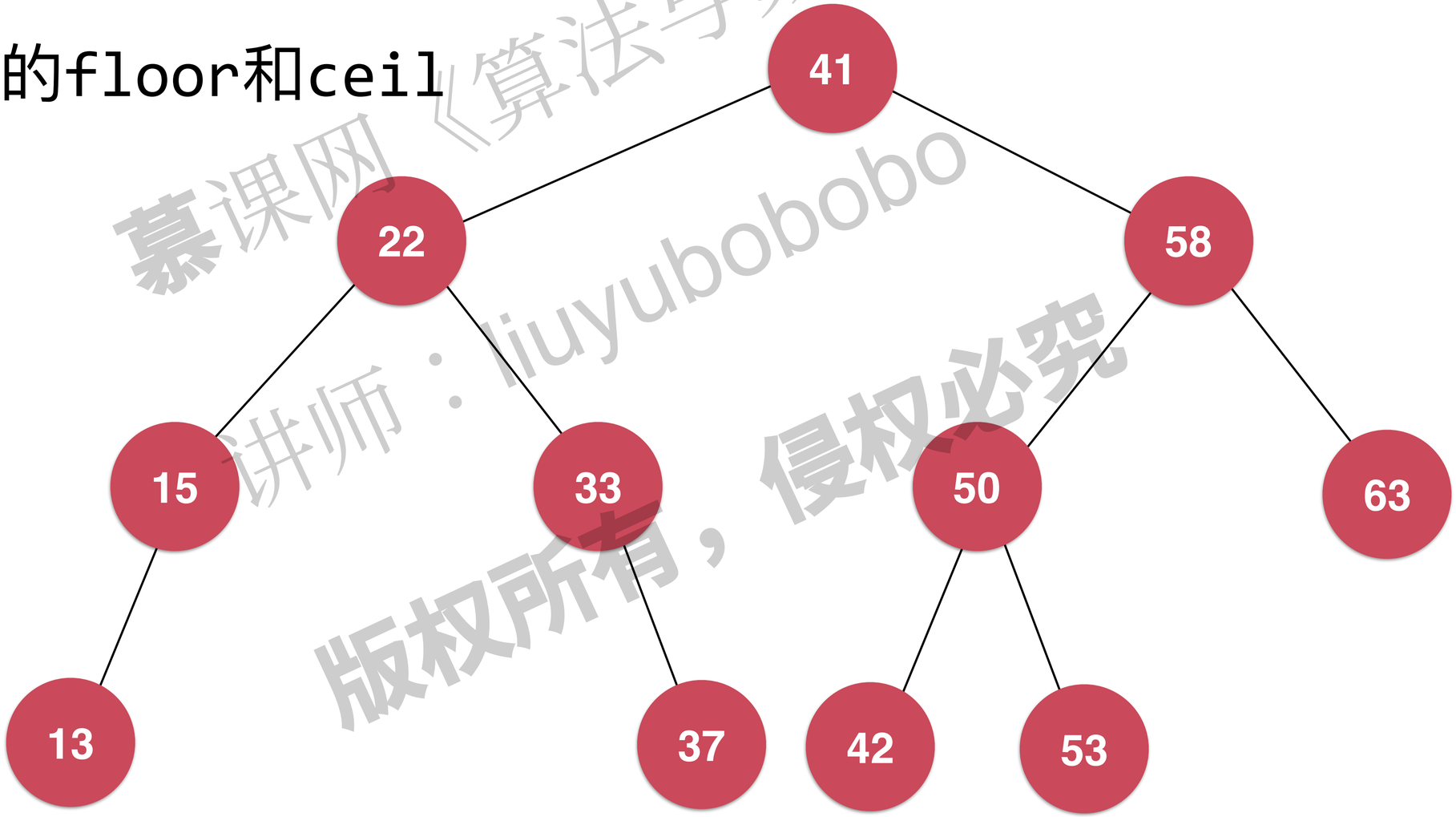
练习：实现 successor，predecessor

二分搜索树的顺序性

floor，ceil

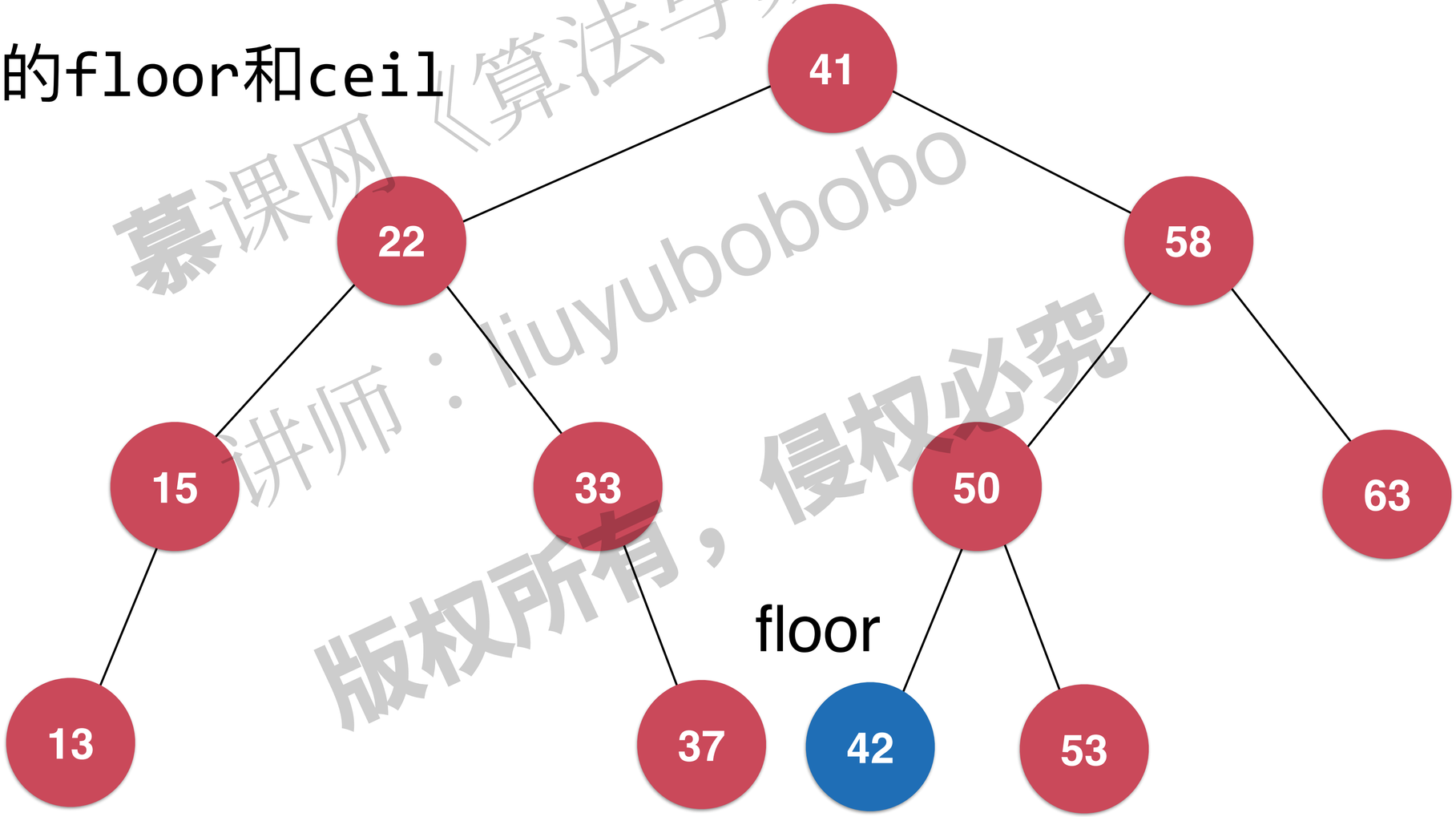
二分搜索树的floor和ceil

寻找45的floor和ceil



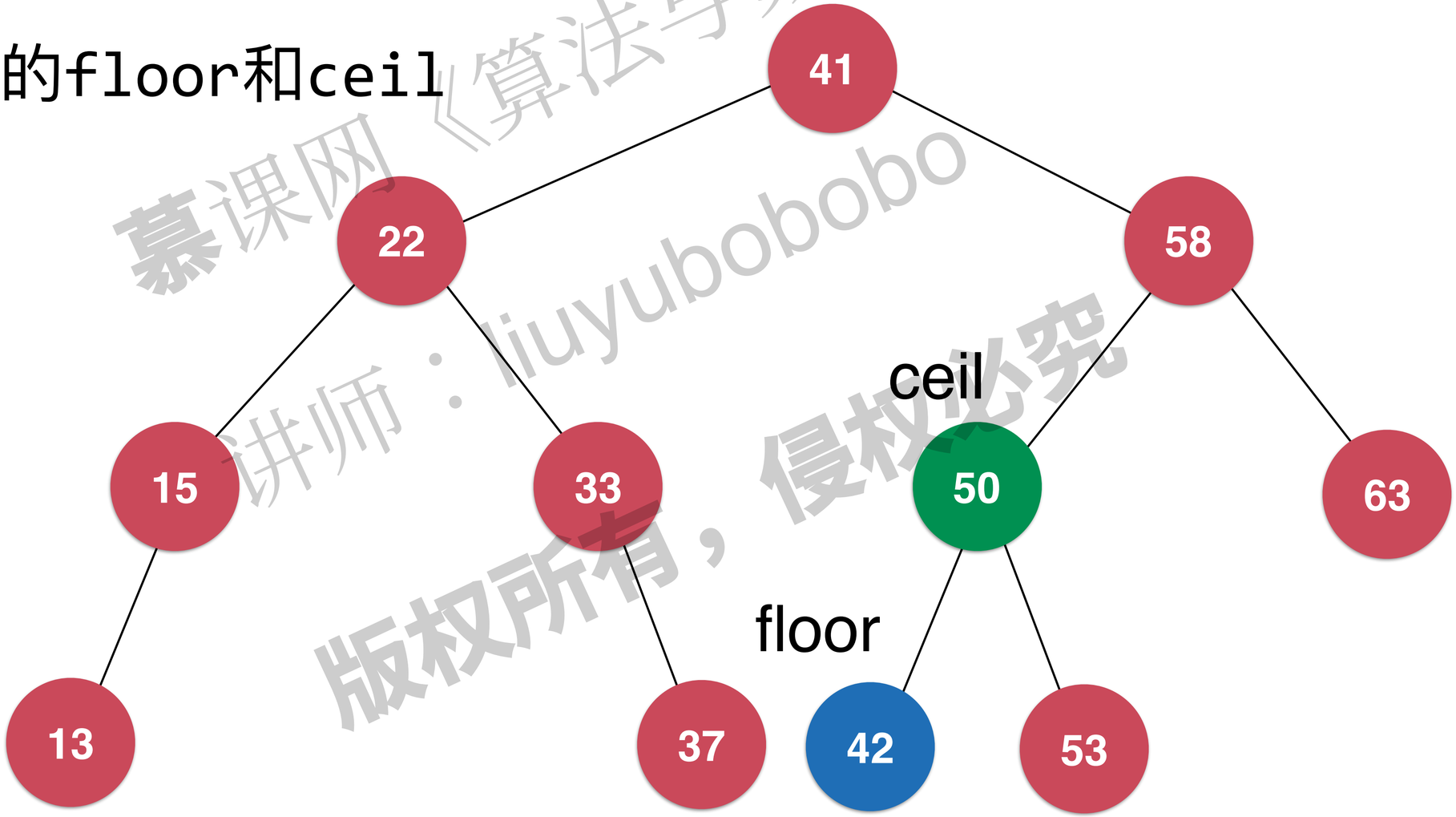
二分搜索树的floor和ceil

寻找45的floor和ceil



二分搜索树的floor和ceil

寻找45的floor和ceil



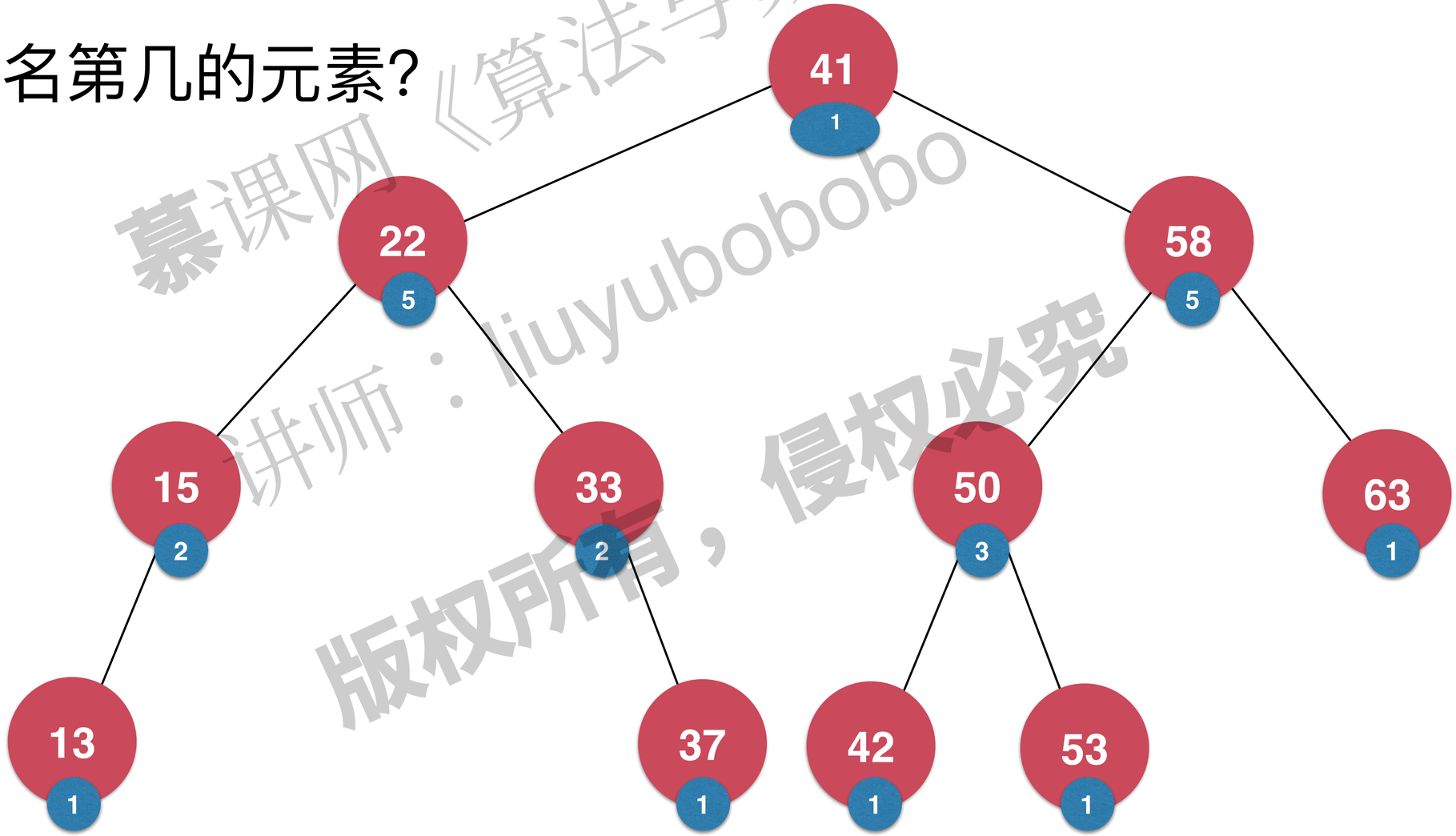
练习：实现 floor, ceil

二分搜索树的顺序性

rank，select

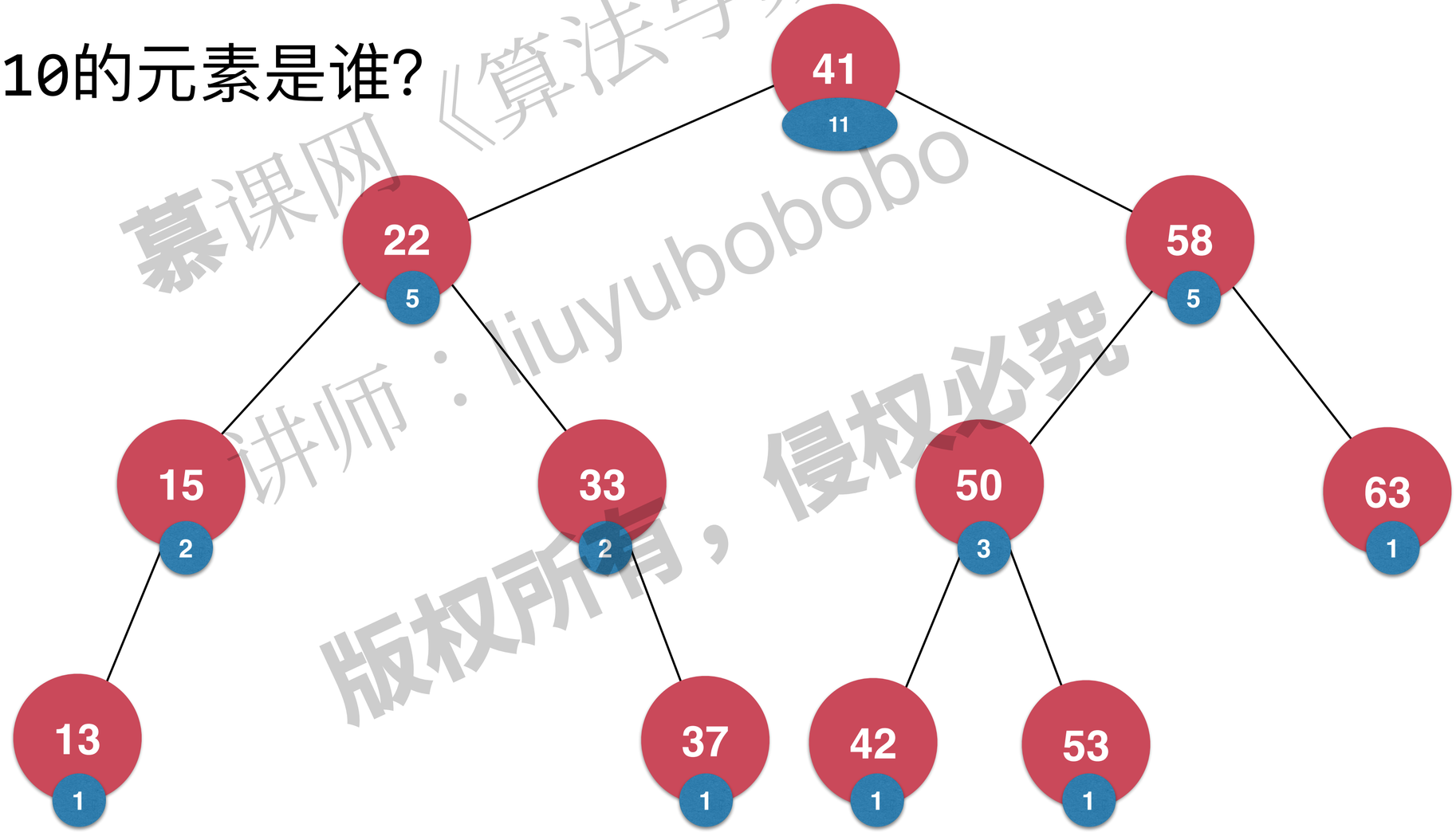
二分搜索树的rank

58是排名第几的元素?



二分搜索树的select

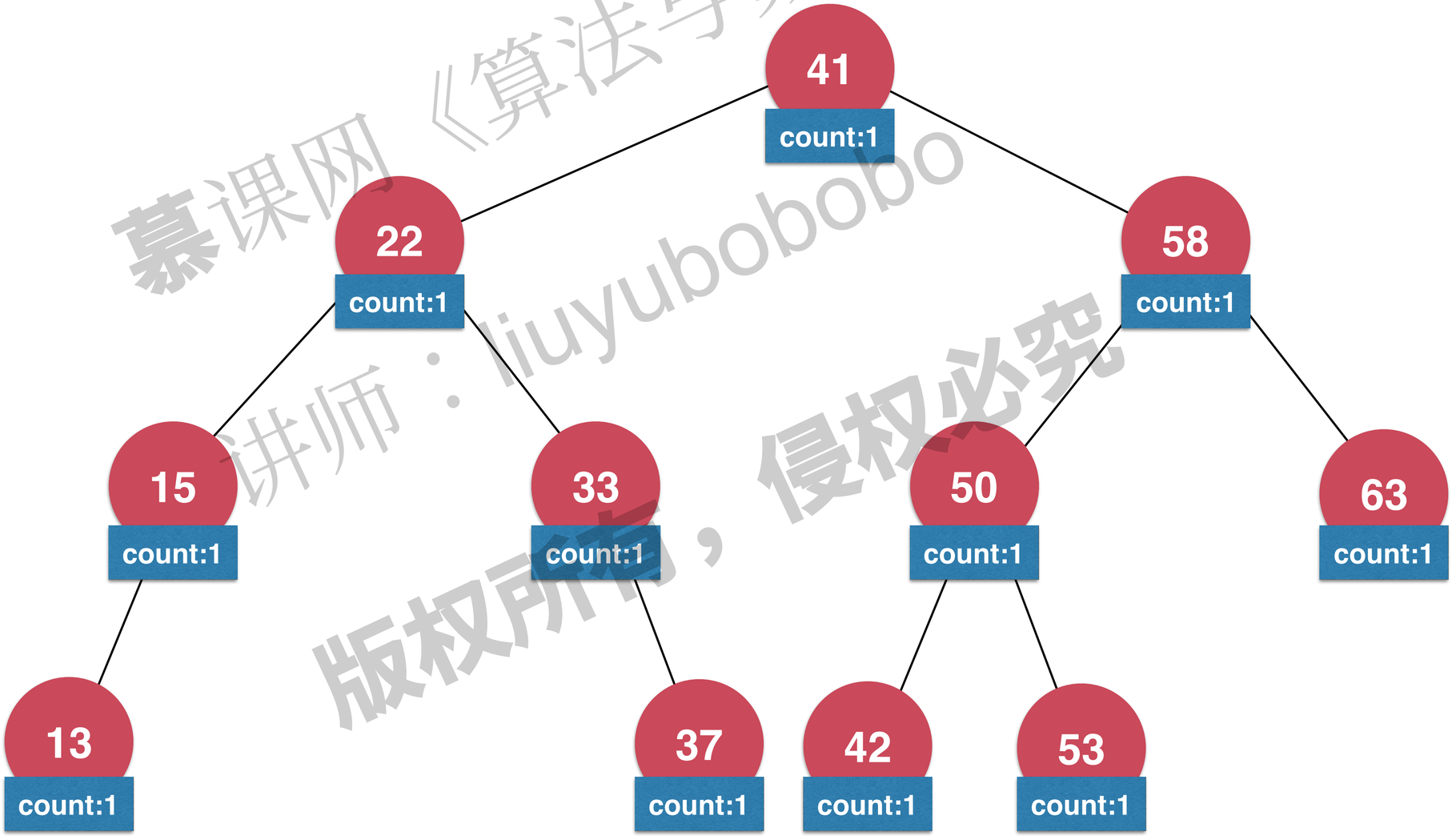
排名第10的元素是谁?



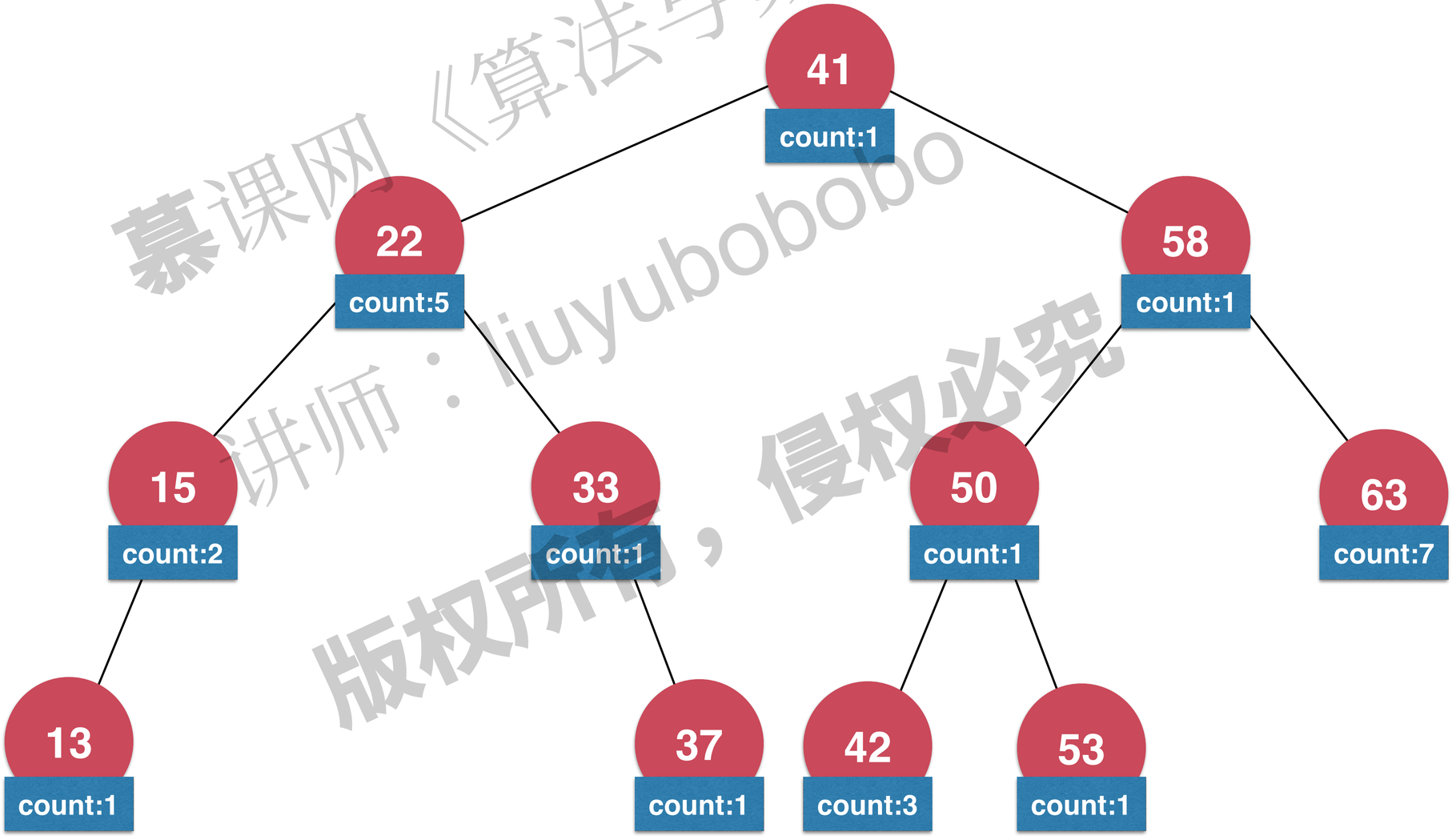
练习：实现 rank，select 并维护size属性

支持重复元素的二分搜索树

支持重复元素的二分搜索树



支持重复元素的二分搜索树

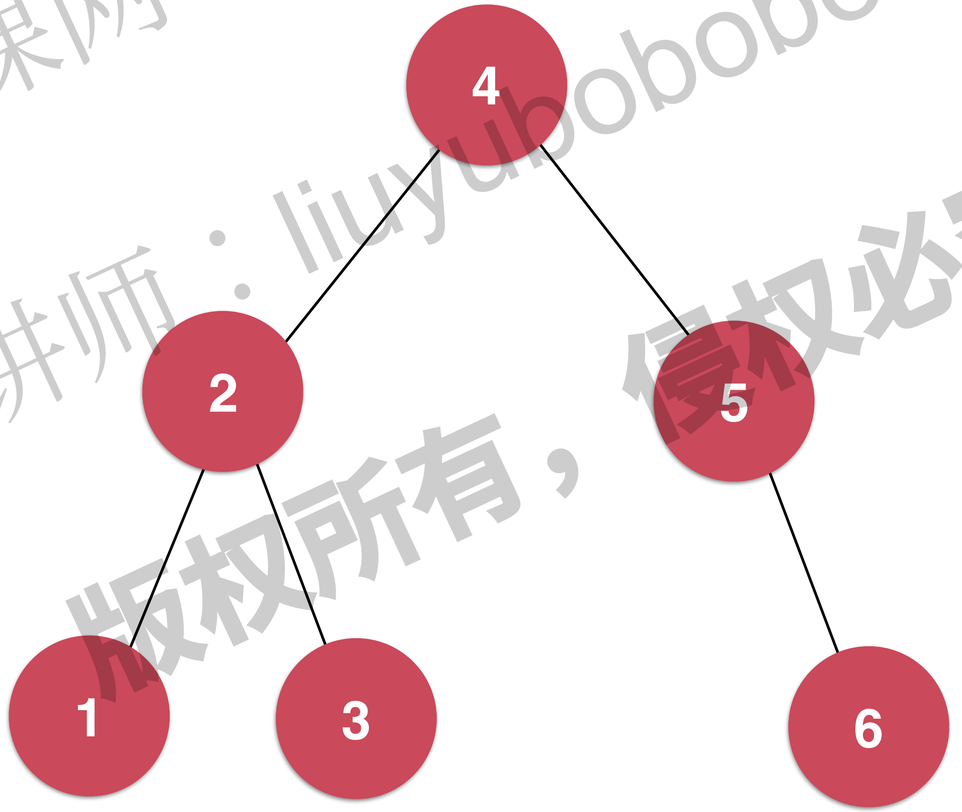


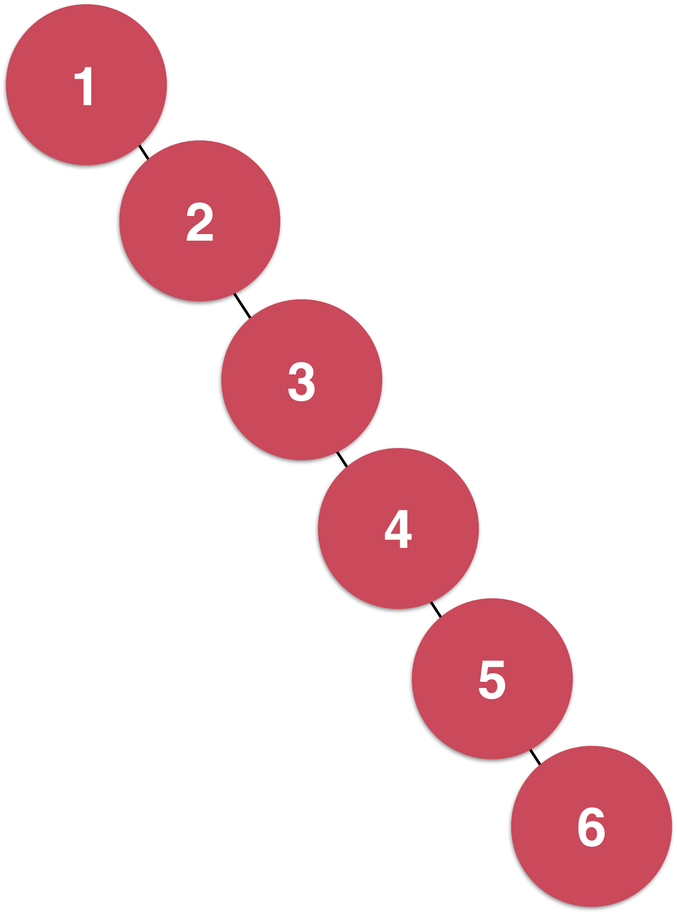
练习：实现一棵支持重复元素，同时支持之前介绍所有操作的二分搜索树

二分搜索树的局限性

同样的数据，可以对应不同的二分搜索树

1，2，3，4，5，6

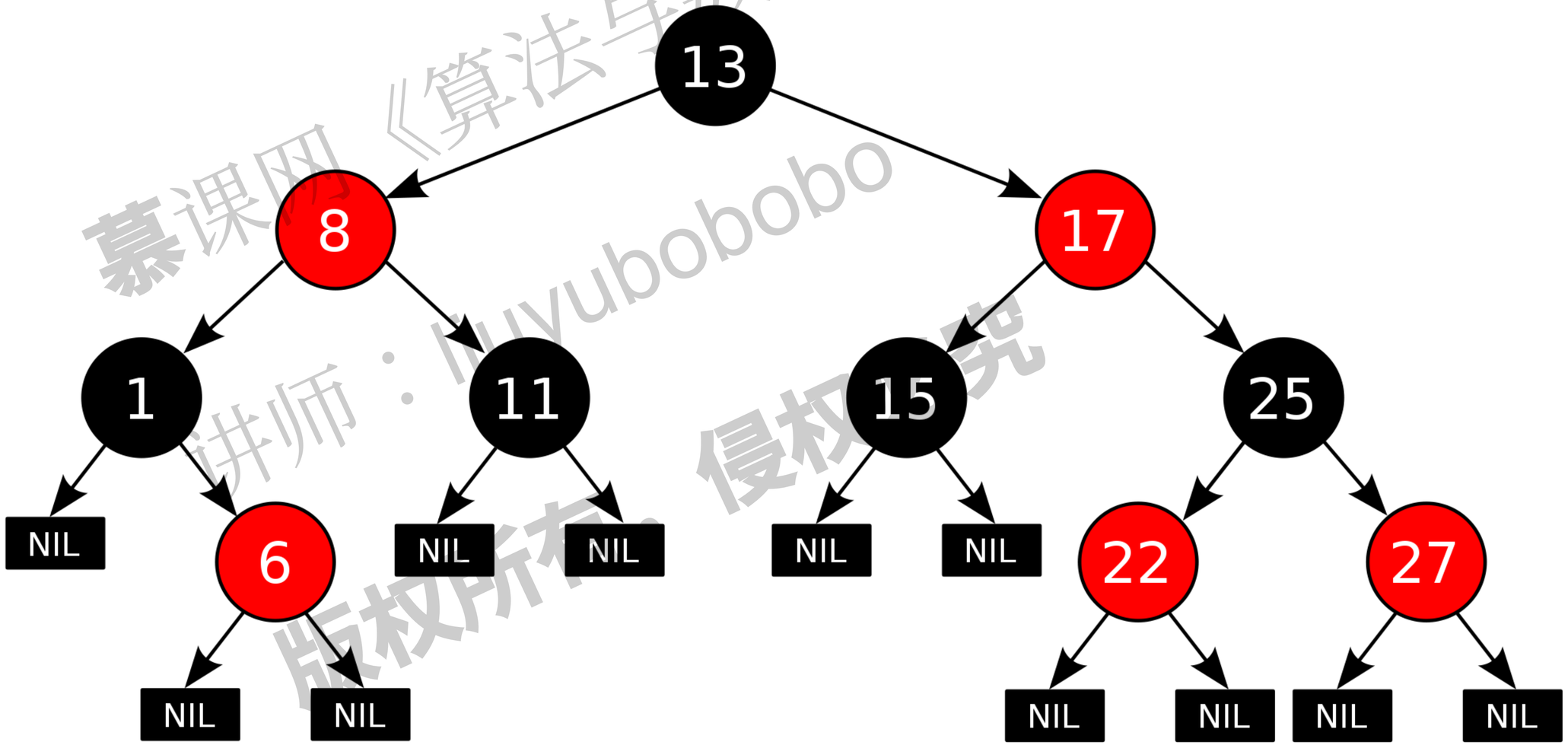




二分搜索树可能退化成链表

平衡二叉树：红黑树

红黑树



其他平衡二叉树的实现

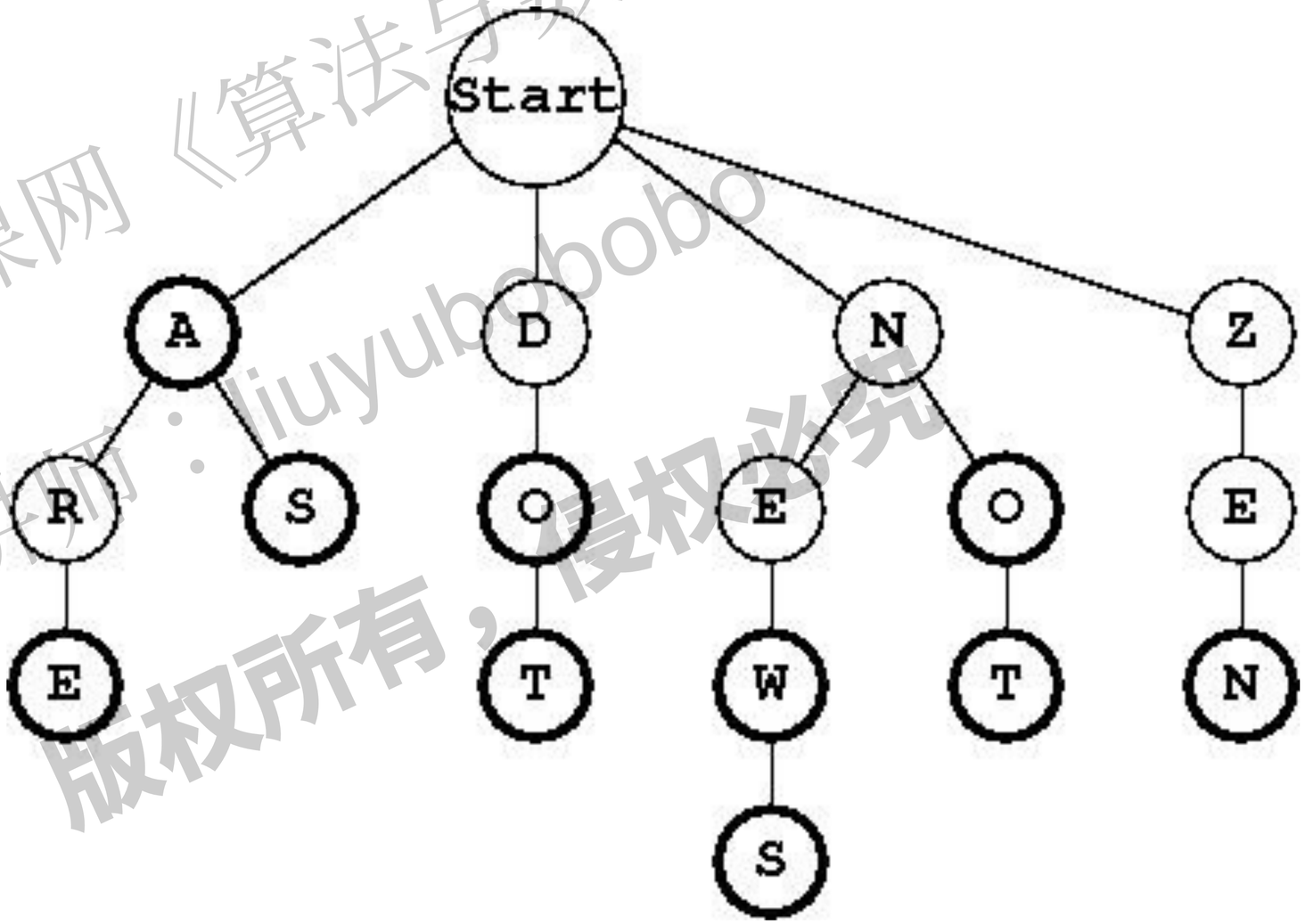
2-3 tree

AVL tree

Splay tree

平衡二叉树和堆的结合：Treap

trie

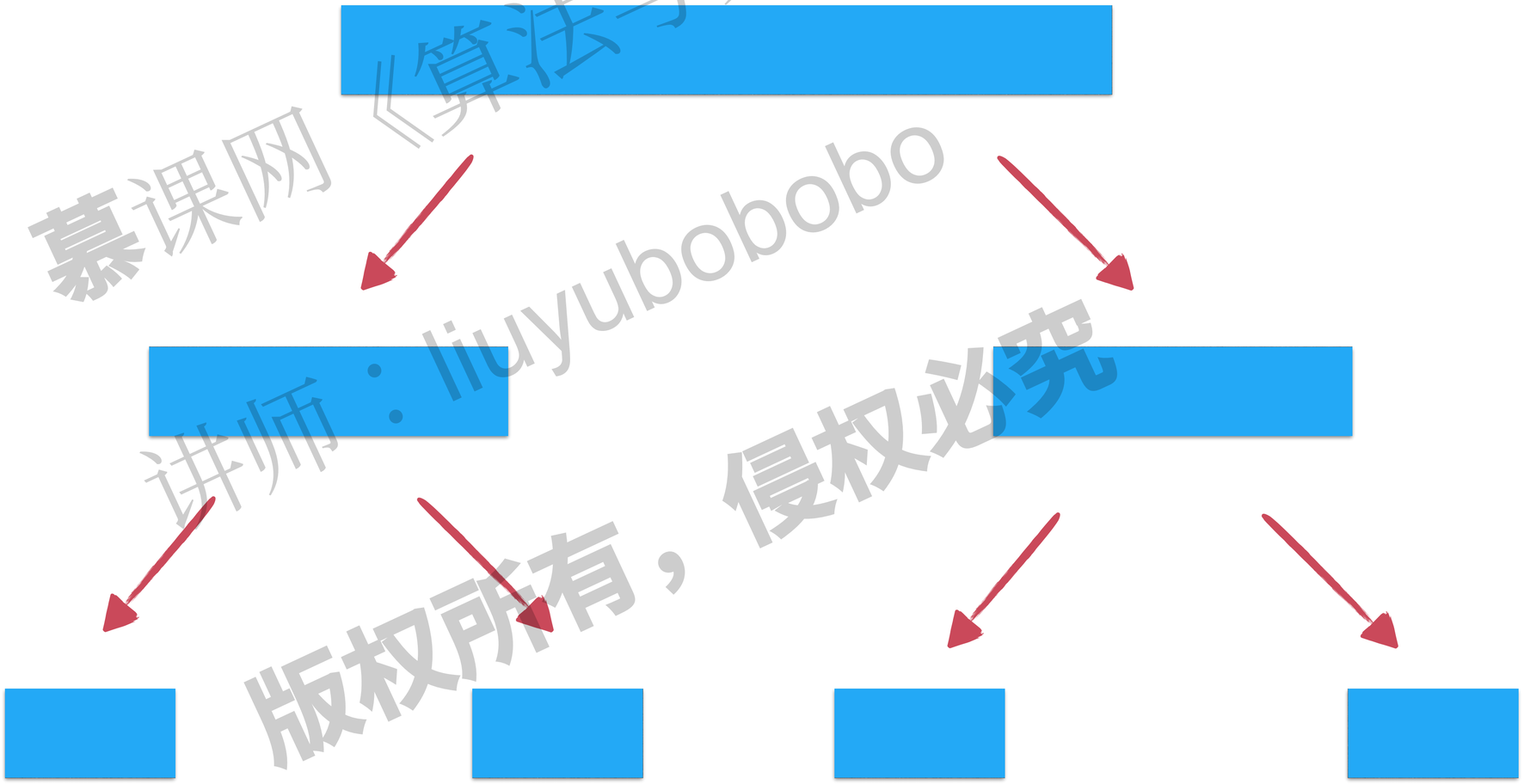


练习：实现一个trie

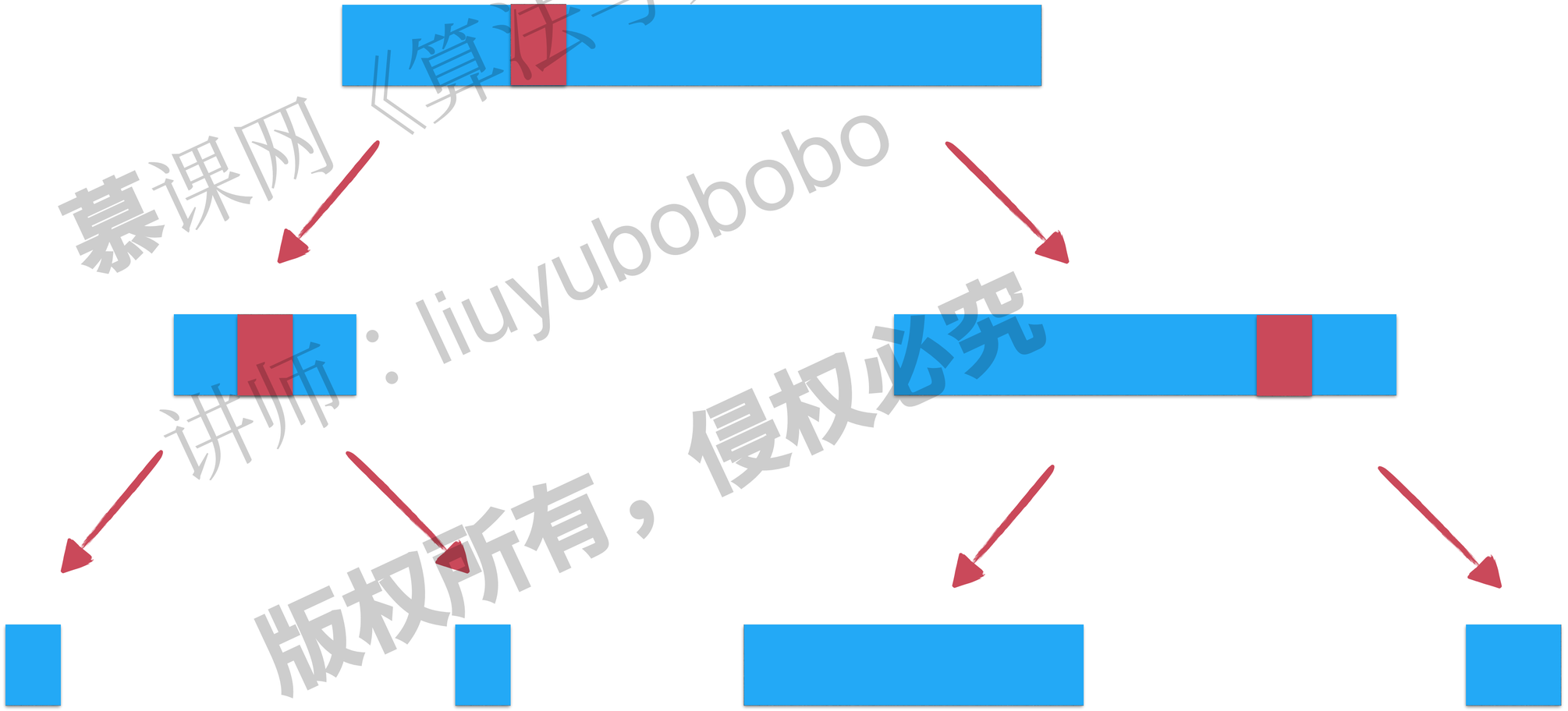
树形问题

递归法天然的树形性质

归并排序

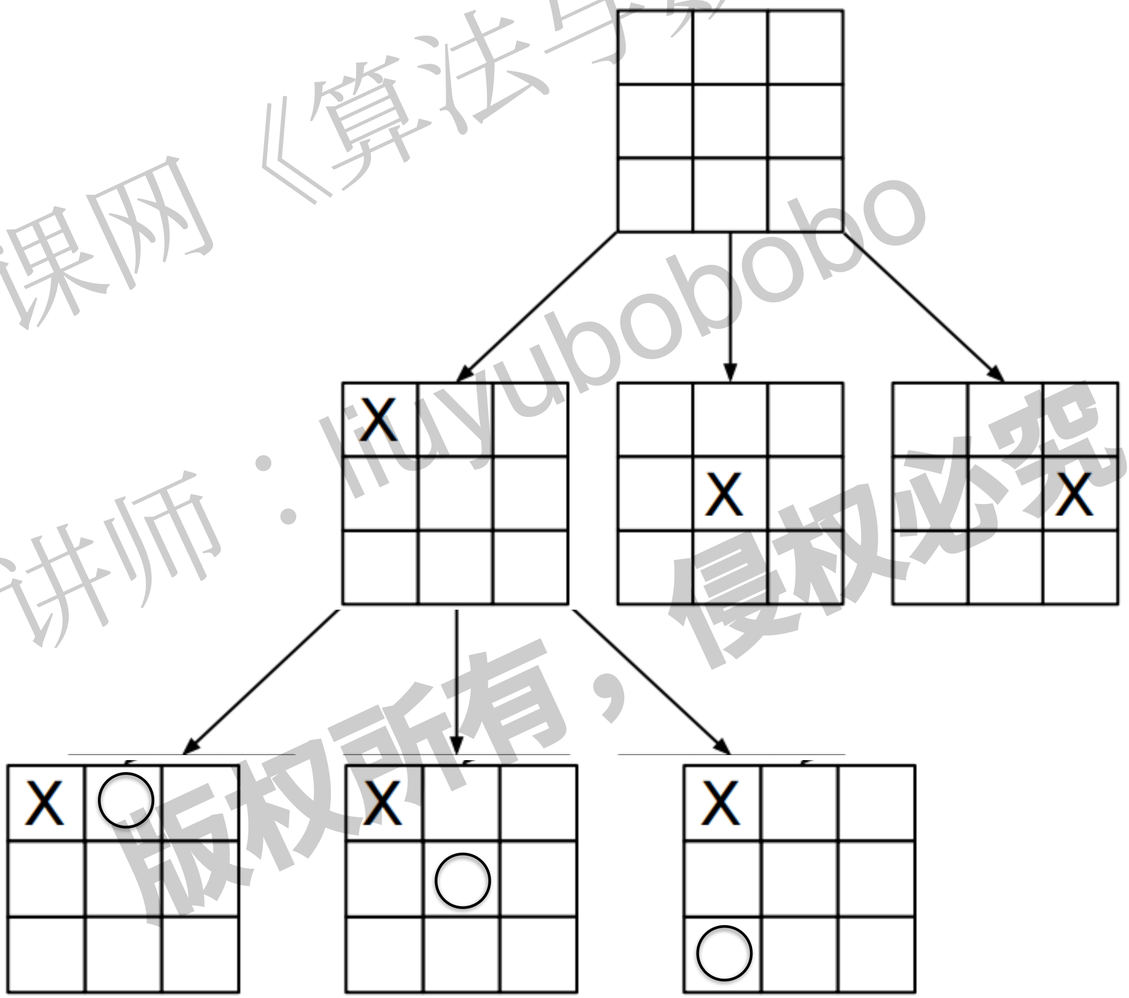


快速排序

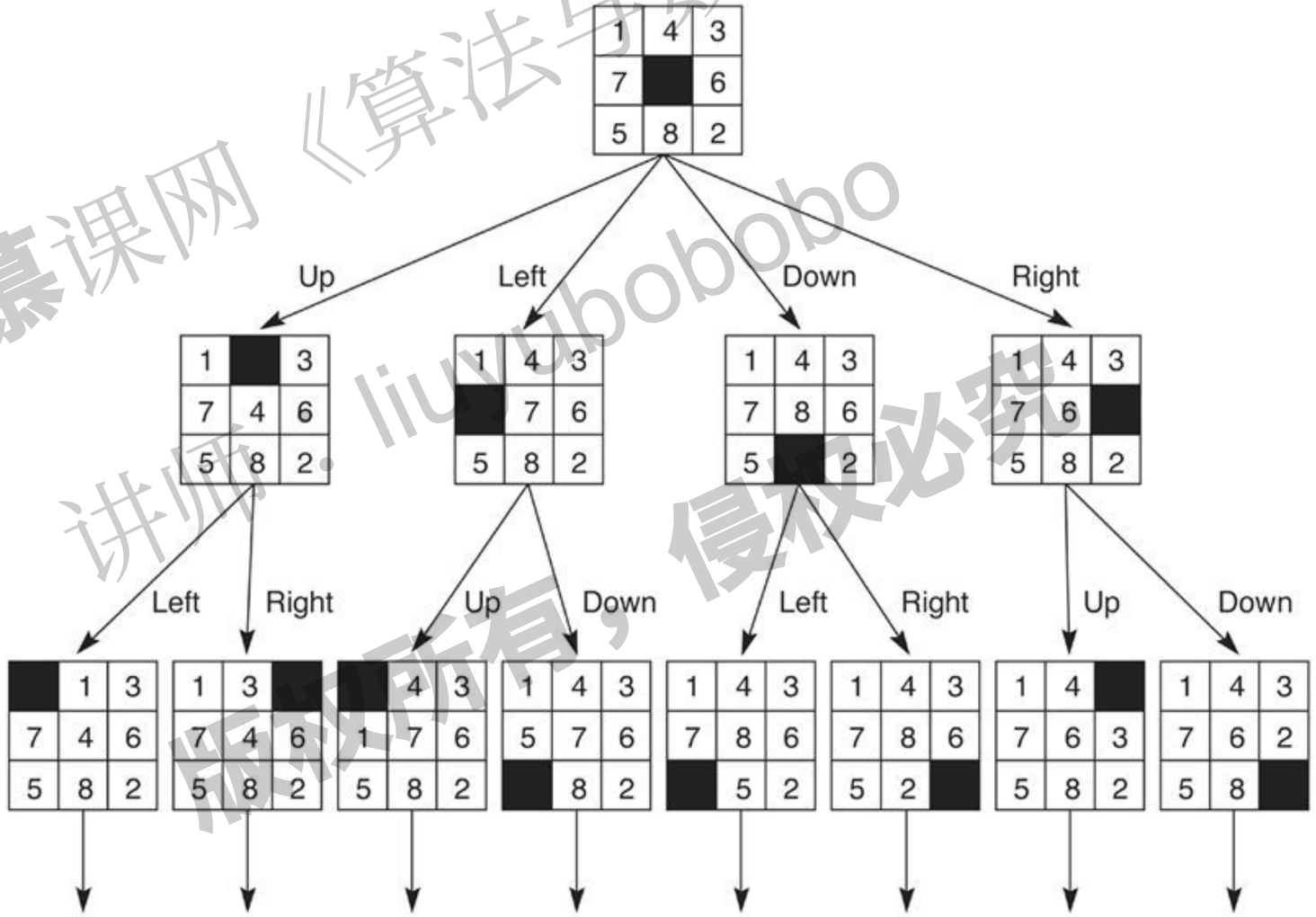


搜索问题

一条龙游戏



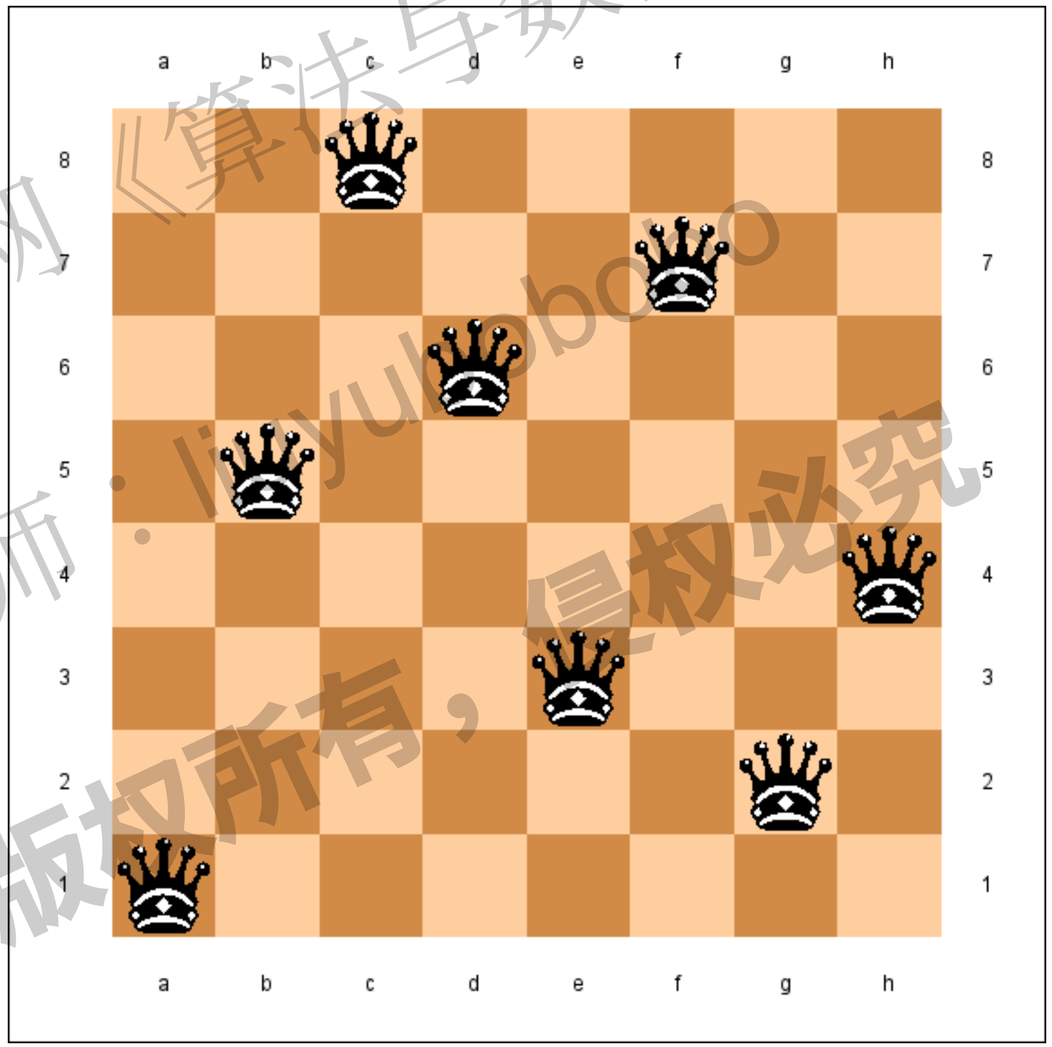
8数码



练习：书写一个八数码求解程序

递归法天然的树形性质

8皇后



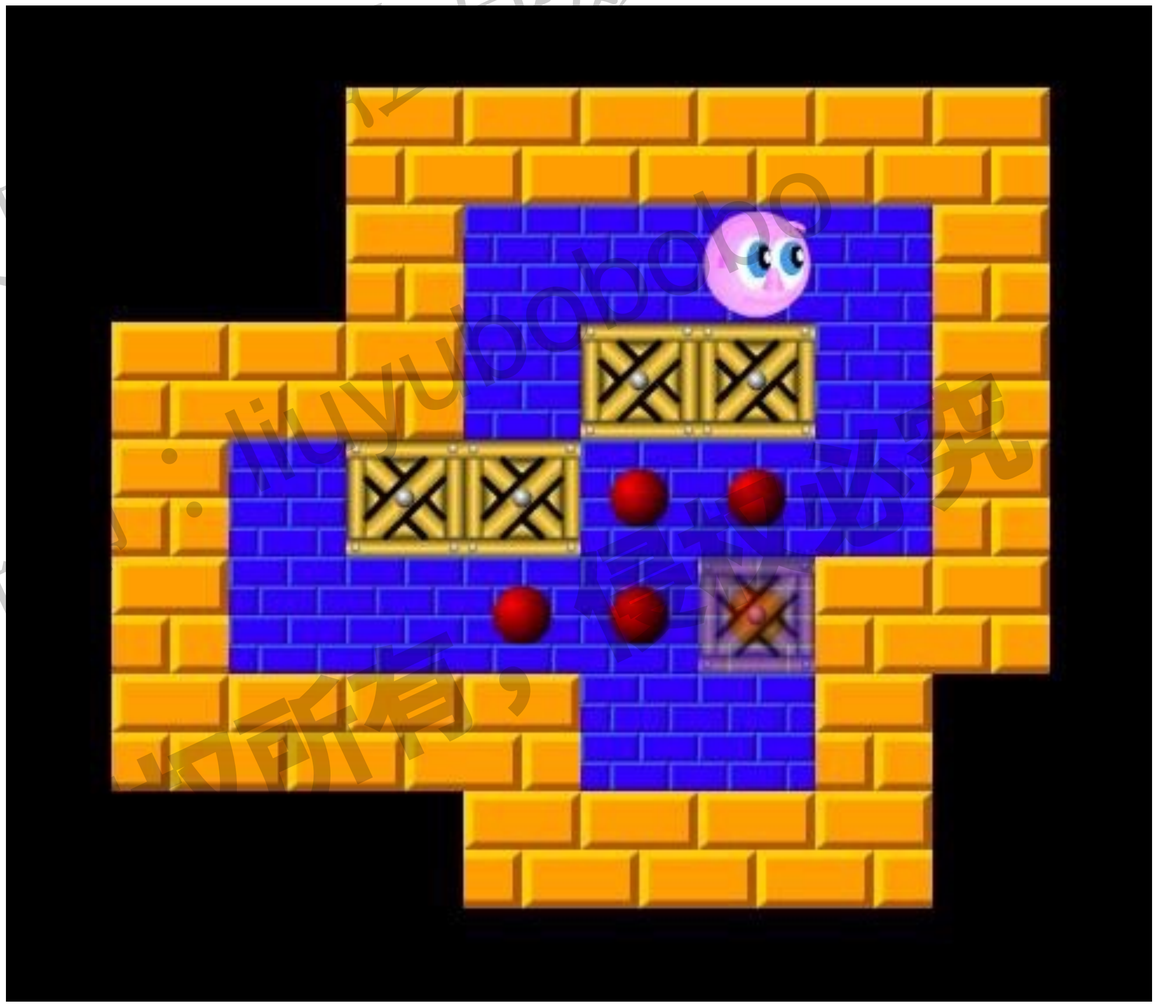
练习：求解八皇后问题的所有解

数独

	6		5	9	3			
9		1				5		
	3		4				9	
1		8		2				4
4			3		9			1
2				1		6		9
	8			6			2	
		4				8		7
			7	8	5		1	

练习：尝试求解数度问题

搬运工



练习：尝试求解搬运工问题

更多树

各种各样的树

KD 树

区间树

哈夫曼树

其他

欢迎大家关注我的个人公众号：是不是很酷

算法与数据结构

liuyubobobo