开课吧

查找

田船长

线性查找——顺序查找

按照某种顺序依次遍历线性表中的每一个元素进行查找

线性查找——折半查找（二分查找）

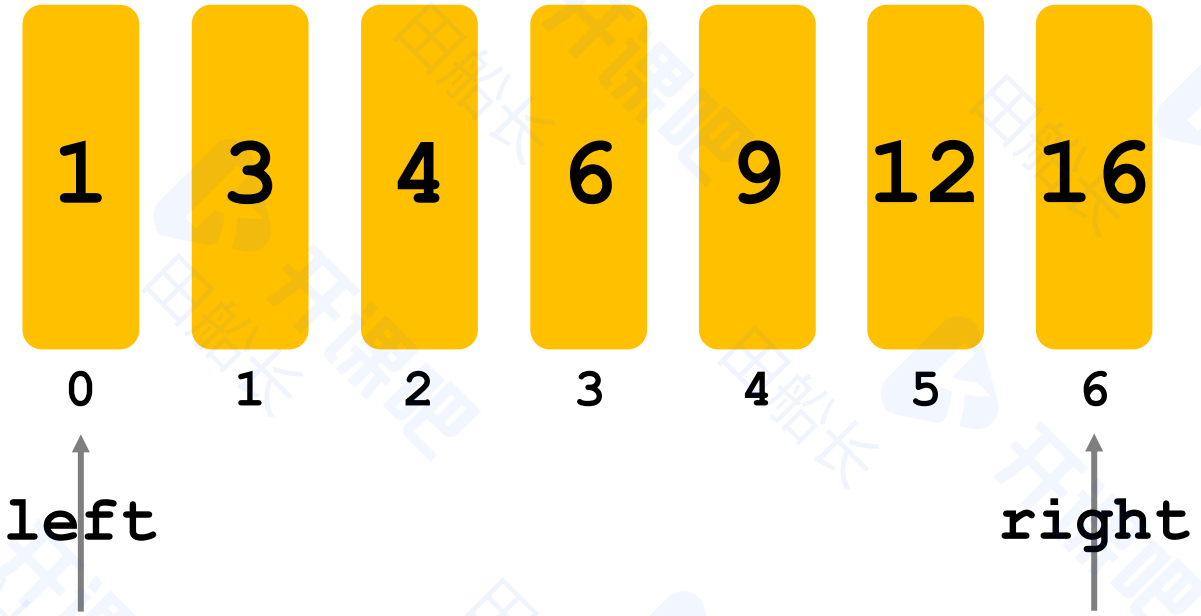
利用待查找序列的有序性
一次找到一半不存在答案的待查找区间


1 3 4 6 9 12 16
0 1 2 3 4 5 6

[任务]折半查找数字 3

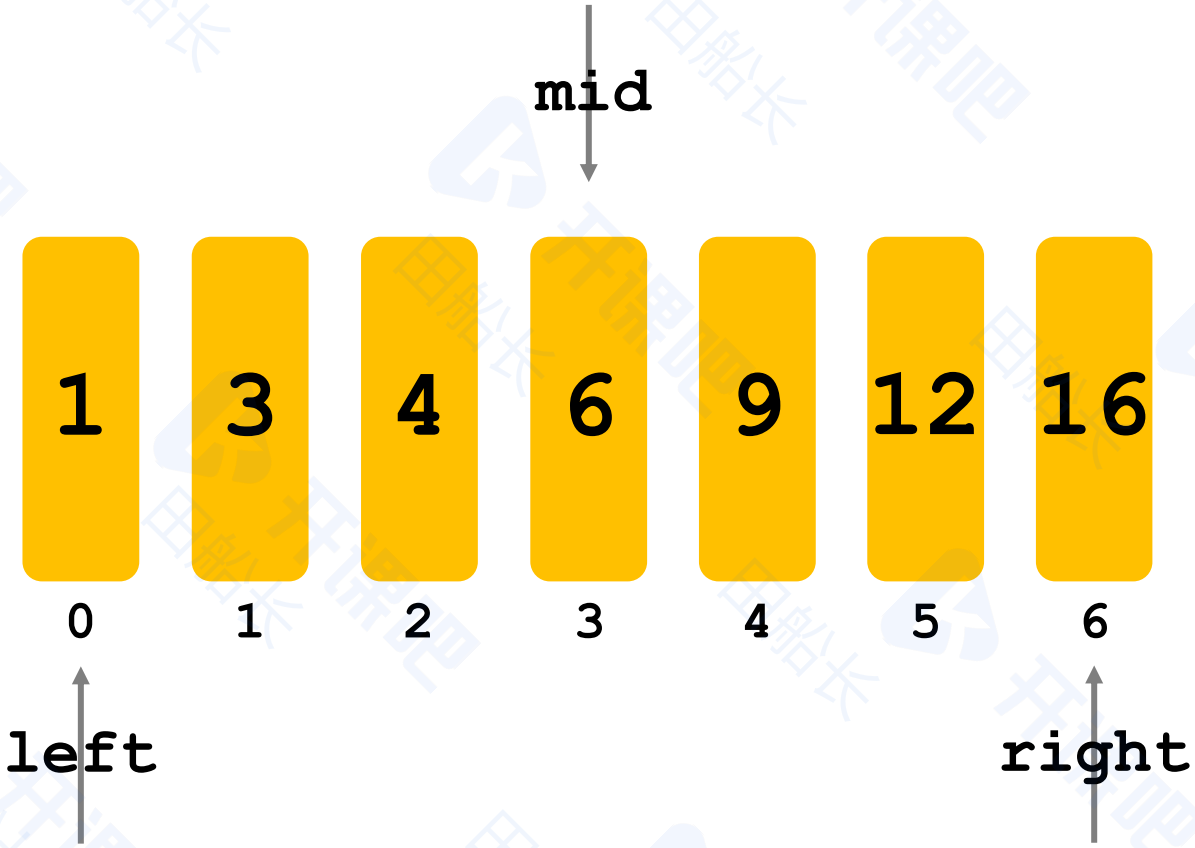
[任务]折半查找数字 3

设置当前查找区间上下界的初值



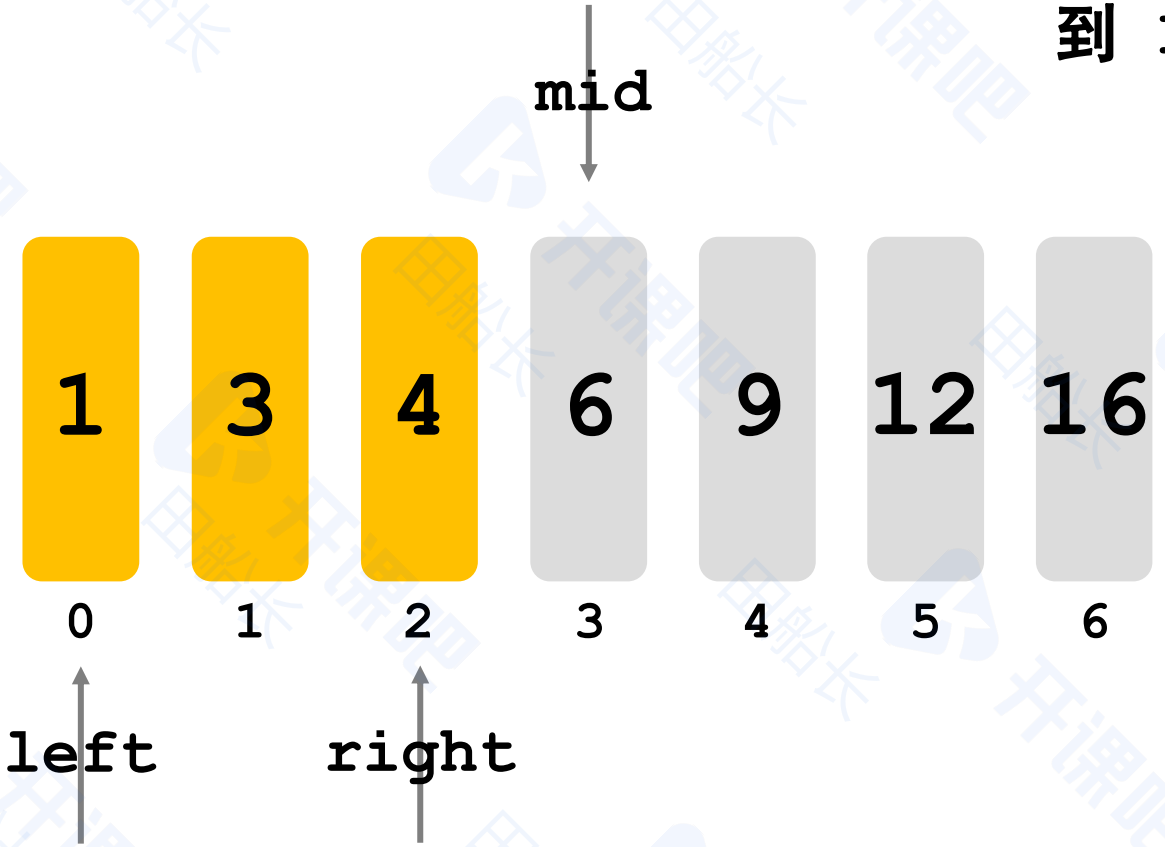
[任务]折半查找数字 3

更新 mid



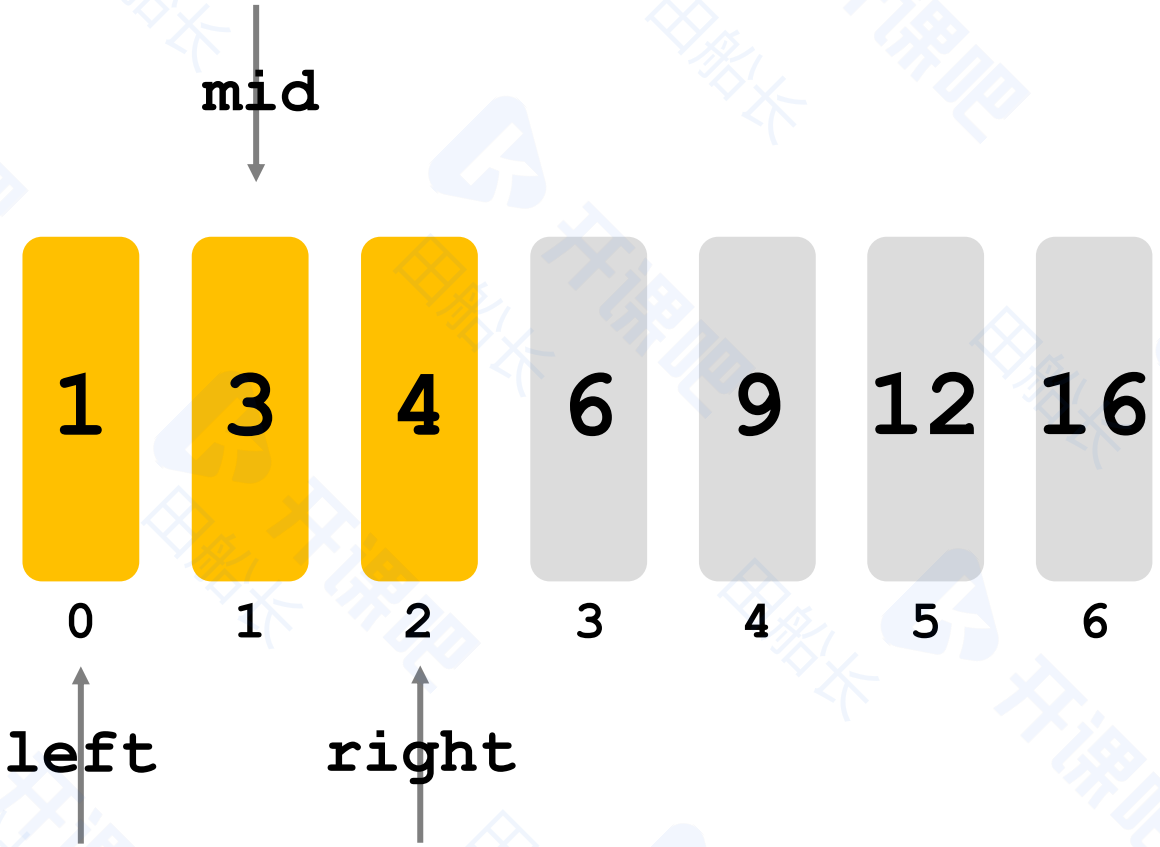
[任务]折半查找数字 3

List[mid] > 3，更新 right，继续在 List[0] 到 List[2] 的区间查找



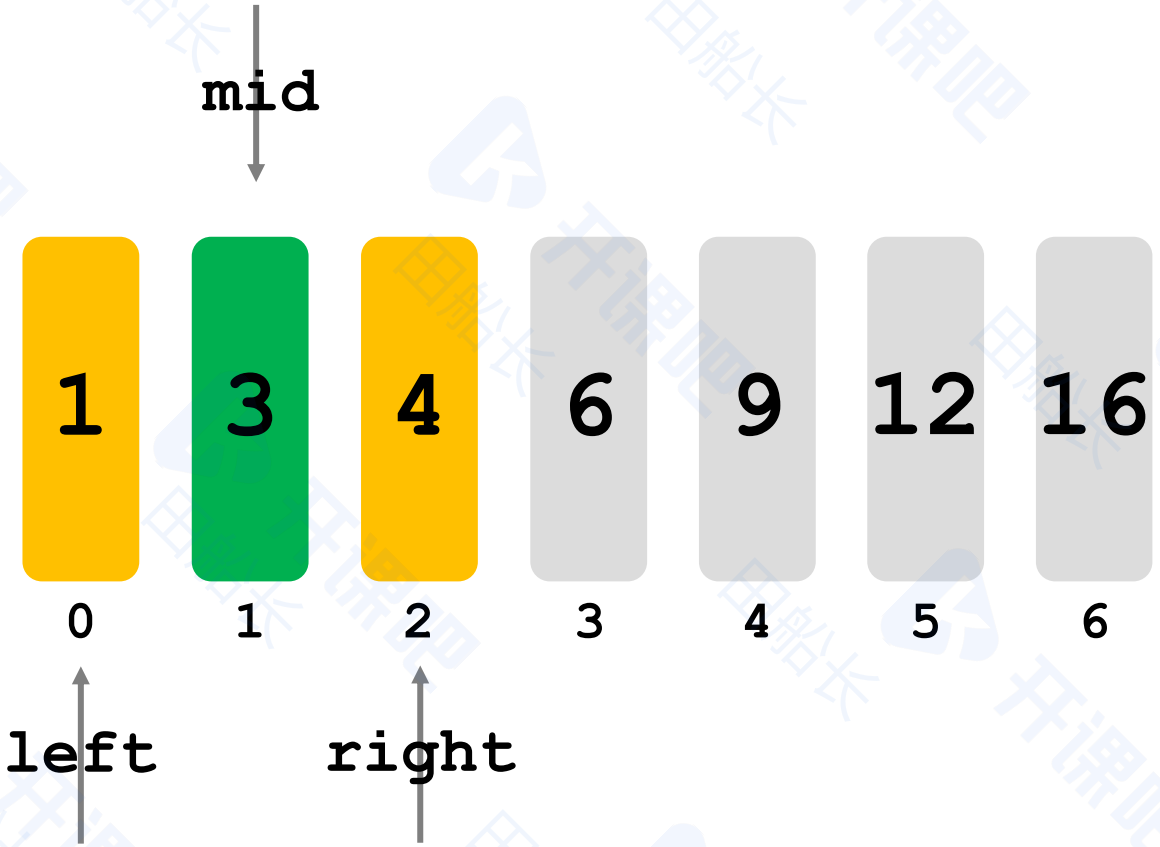
[任务]折半查找数字 3

更新 mid



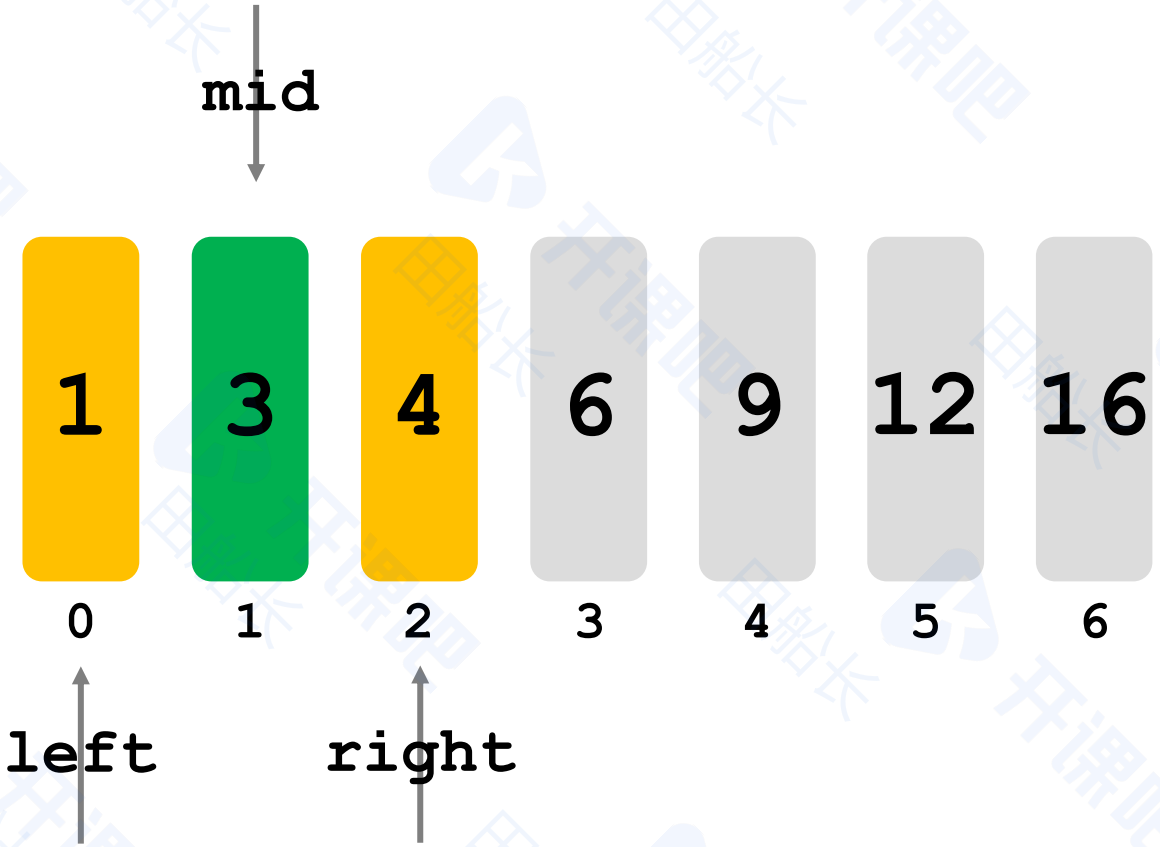
[任务]折半查找数字 3

找到 3，在下标为 1 的位置上



[任务]折半查找数字 3

查找完毕



线性查找——分块查找

同一块待查找序列可以无序，所有区块之间有序
（不绝对）


22
22
12
13
3
9
44
33
42
44
38
24
86
60
58
74
49
86

[任务]将1插入到对应的块中

[任务]将1插入到对应的块中

比较索引表第0个索引，1应在第0个索引内

[任务]将1插入到对应的块中

将1插入到列表中，插入操作完成

[任务]查找元素43

[任务]查找元素43

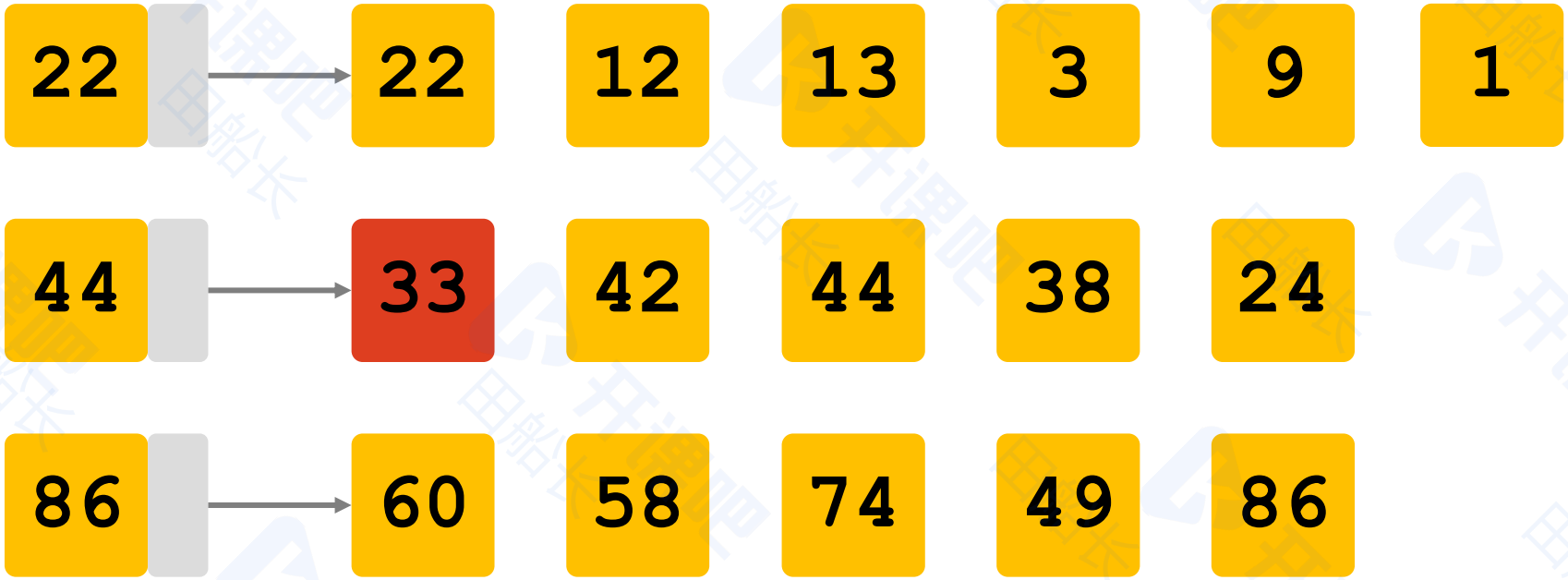
比较索引表第0个索引

[任务]查找元素43

比较索引表第1个索引

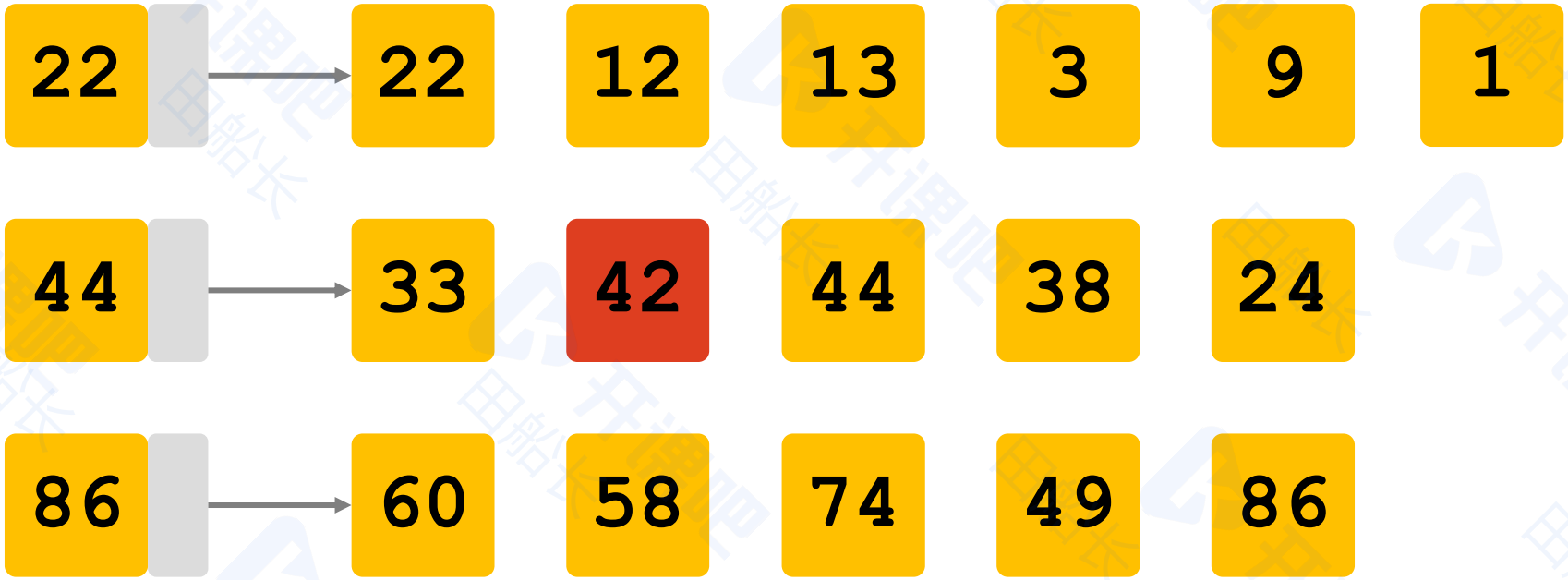
[任务]查找元素43

查找第1个索引中第0个元素



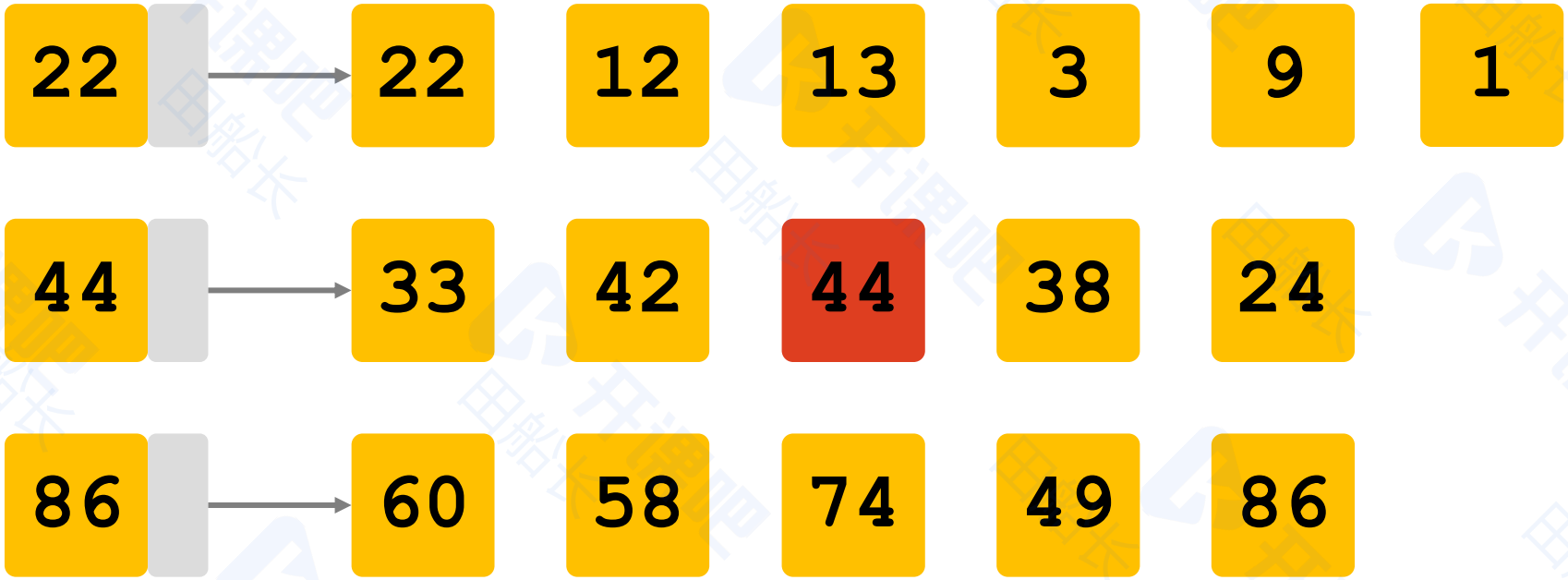
[任务]查找元素43

查找第1个索引中第1个元素



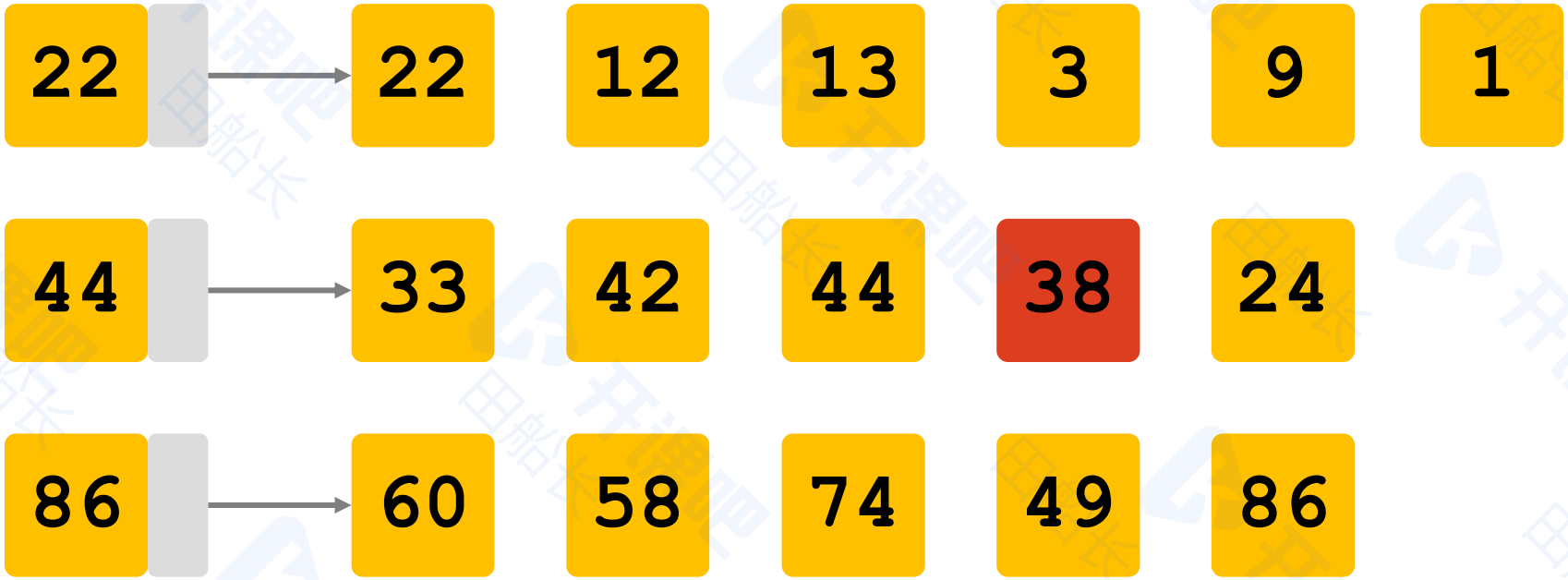
[任务]查找元素43

查找第1个索引中第2个元素



[任务]查找元素43

查找第1个索引中第3个元素



[任务]查找元素43

查找第1个索引中第4个元素

[任务]查找元素43

43不在表中

[任务]查找元素43

查找完毕

树形查找——平衡二叉树（AVL树）

因为二叉排序树的性质过于简单
在最坏情况下会退化成链表
使得效率极差
所以引入了平衡二叉树的概念
对于平衡二叉树来说
每个结点的平衡因子只能是-1,0,1三数之一

平衡因子=左子树的高度-右子树的高度

树形查找——平衡二叉树（AVL树）

由于有平衡因子的存在
所以在插入元素和删除元素时
需要进行平衡性的调整

树形查找——B树（多路平衡查找树）

m阶B树的性质：（⌈ ⌉为上取整）

1.除根结点外，每个结点的关键字个数最少为⌈m/2⌉−1

2.每个结点的关键字个数最多为m−1

3.除根结点外，每个结点的子树最少为⌈m/2⌉棵

4.每个结点的子树最多为m棵

5.最下一层的结点为叶子结点，不存储任何信息

散列查找——散列表（哈希表，杂凑）

根据哈希函数直接计算出某元素对应的哈希值
并将元素放入对应位置的桶当中
关于哈希需要注意常见的哈希函数与常见的冲突处理方法

装填因子=表中元素数/表的容量上限（表长）