开课吧

排序

田船长

进行一系列操作使得序列有序

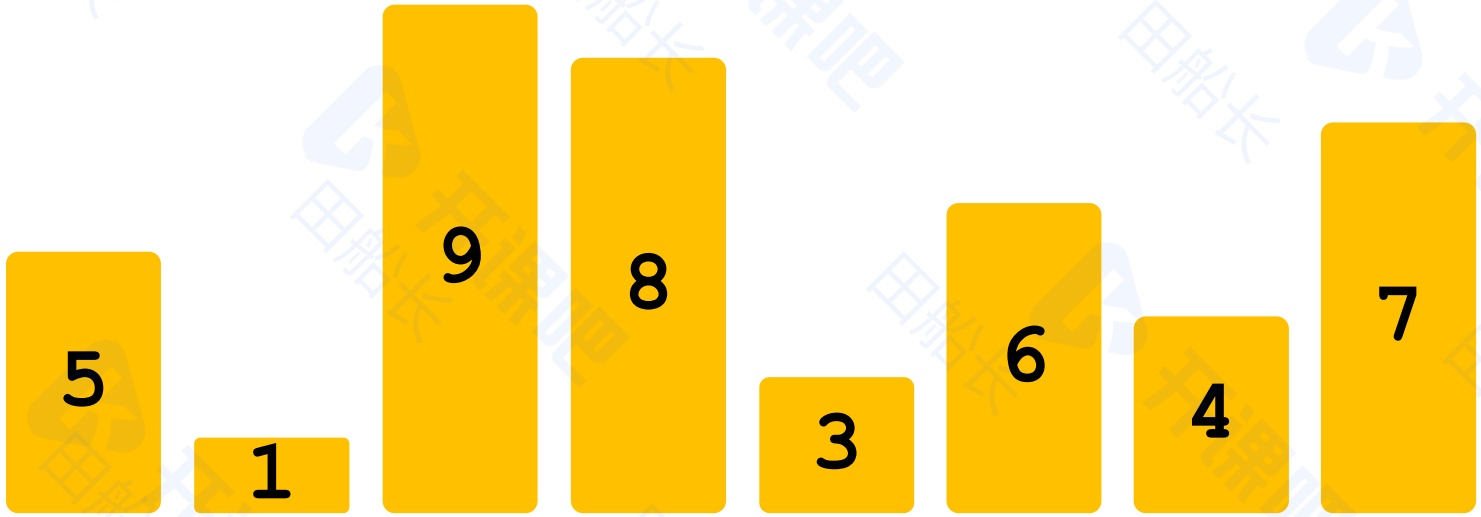
排序算法的性能指标：

时间复杂度（最好、最坏、平均）
（额外）空间复杂度
稳定性

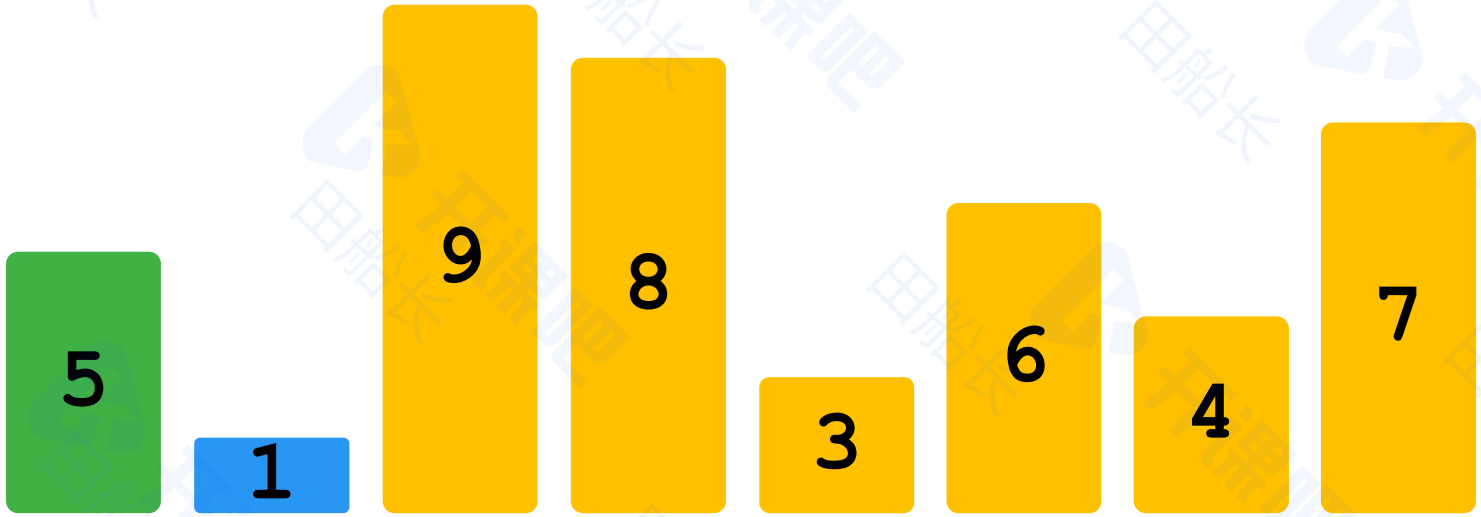
排序方法没有最好的，只有最适用的

插入类排序——直接插入排序

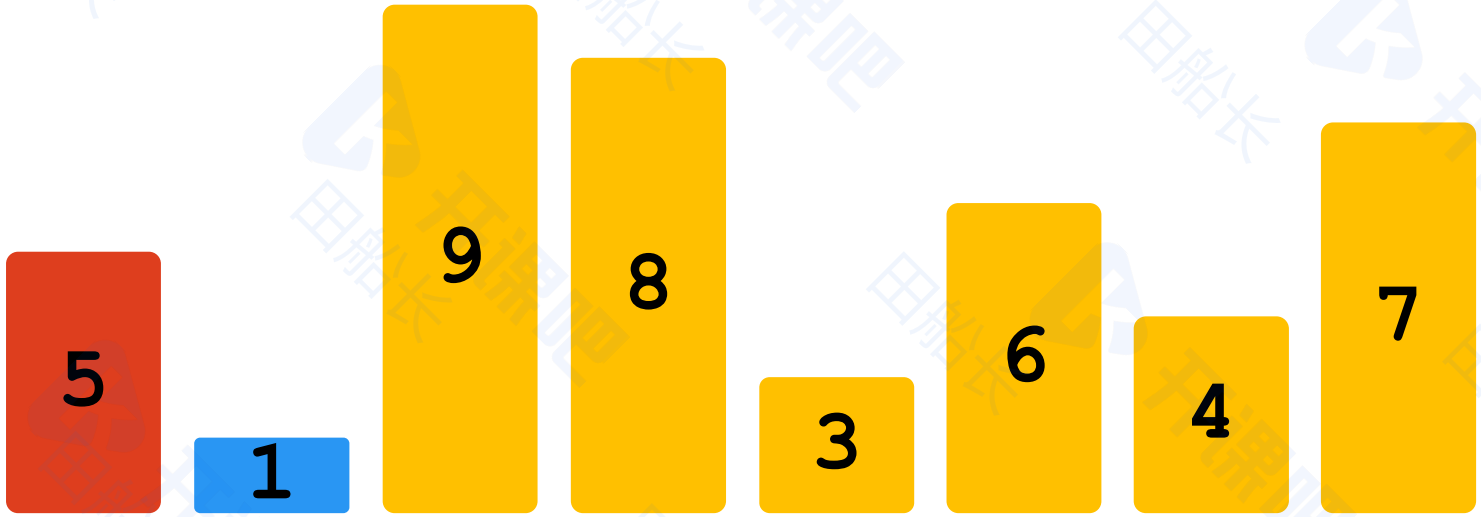
将待排序区的第一个元素通过大小比较
直接插入到已排序区的指定位置
使得已排序元素数量加一
不断重复此过程，直到所有元素均已排序



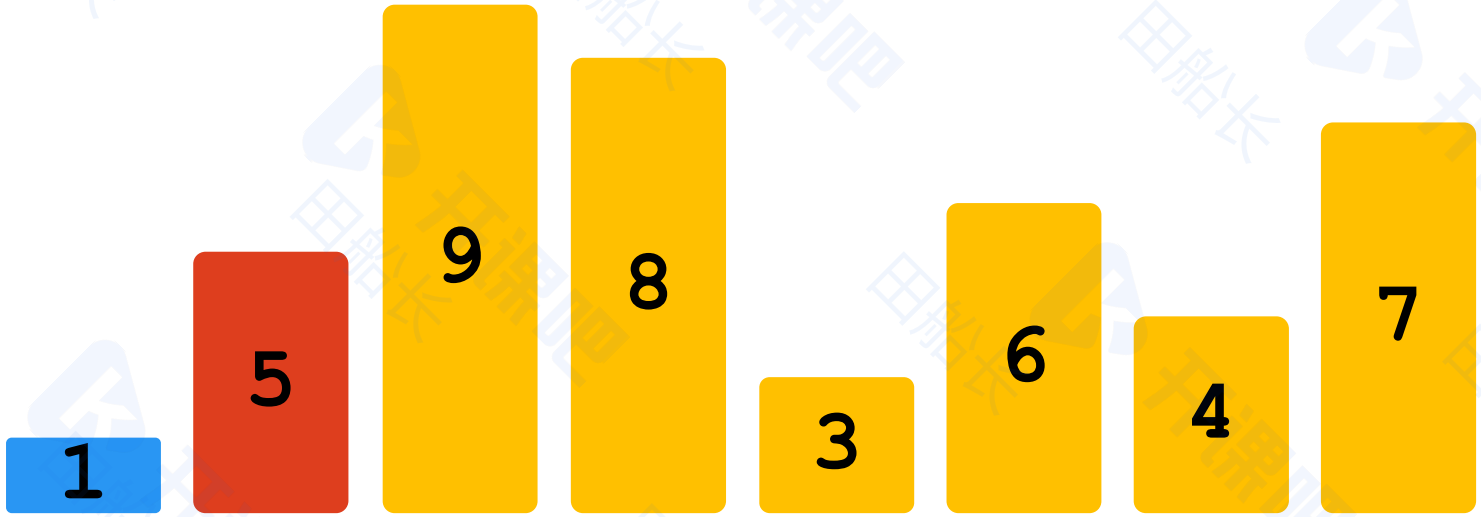
把第1个数字1插入到
由前1个数字组成的
有序数组中



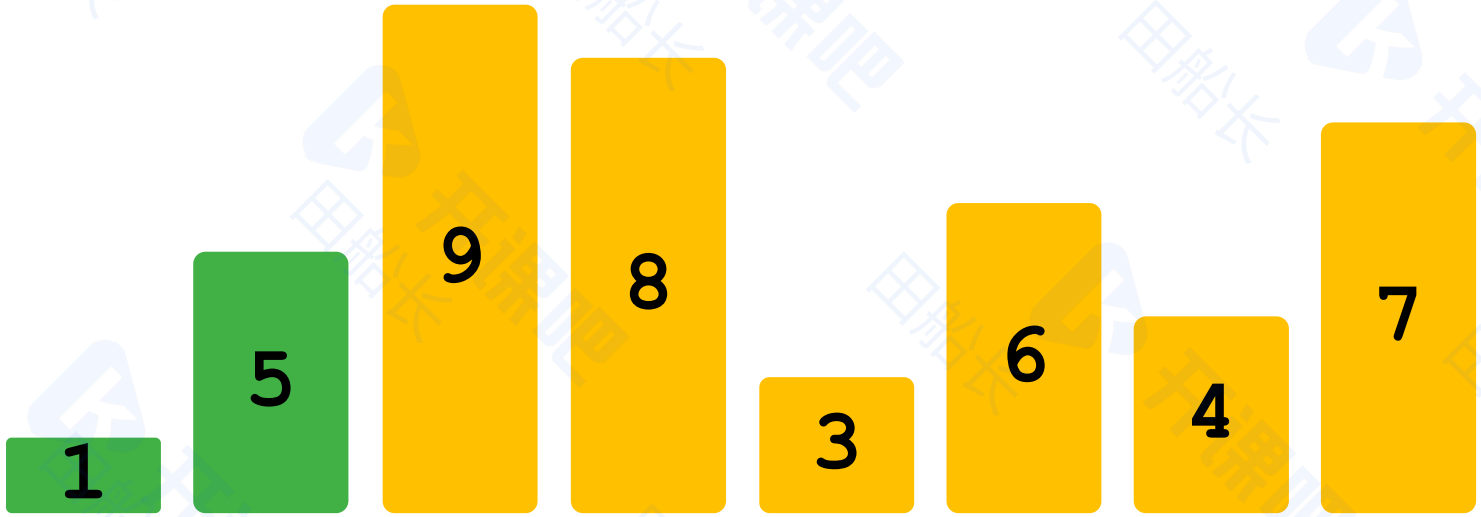
比较1和其前面的5，
看是否为逆序对



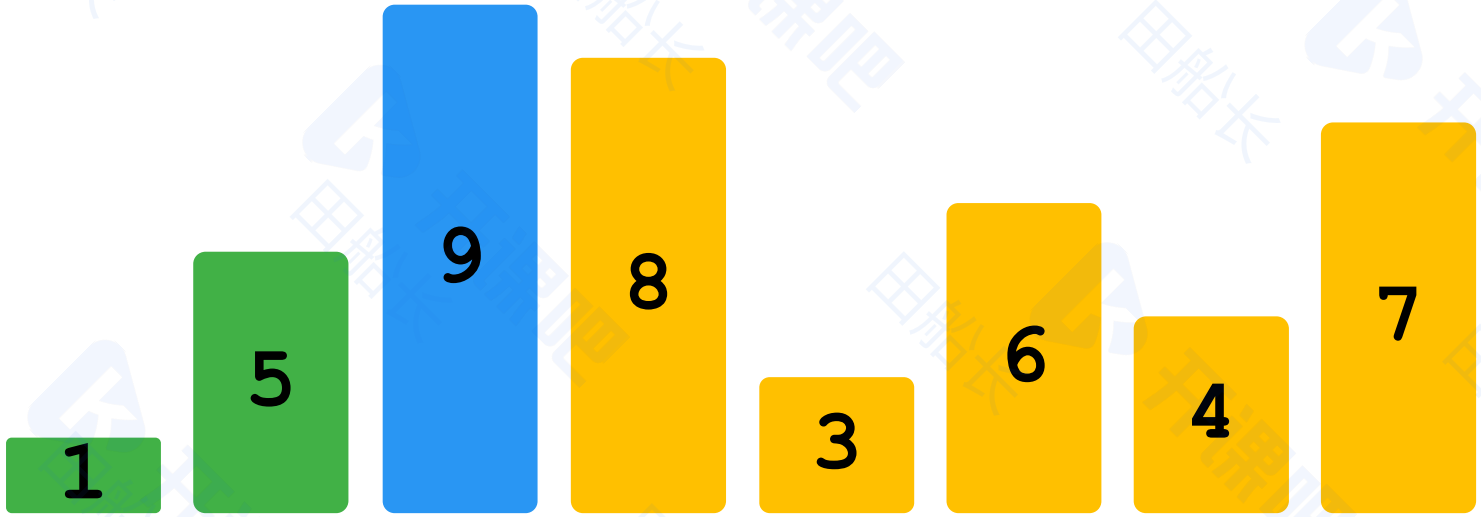
是逆序对，交换5和1的位置



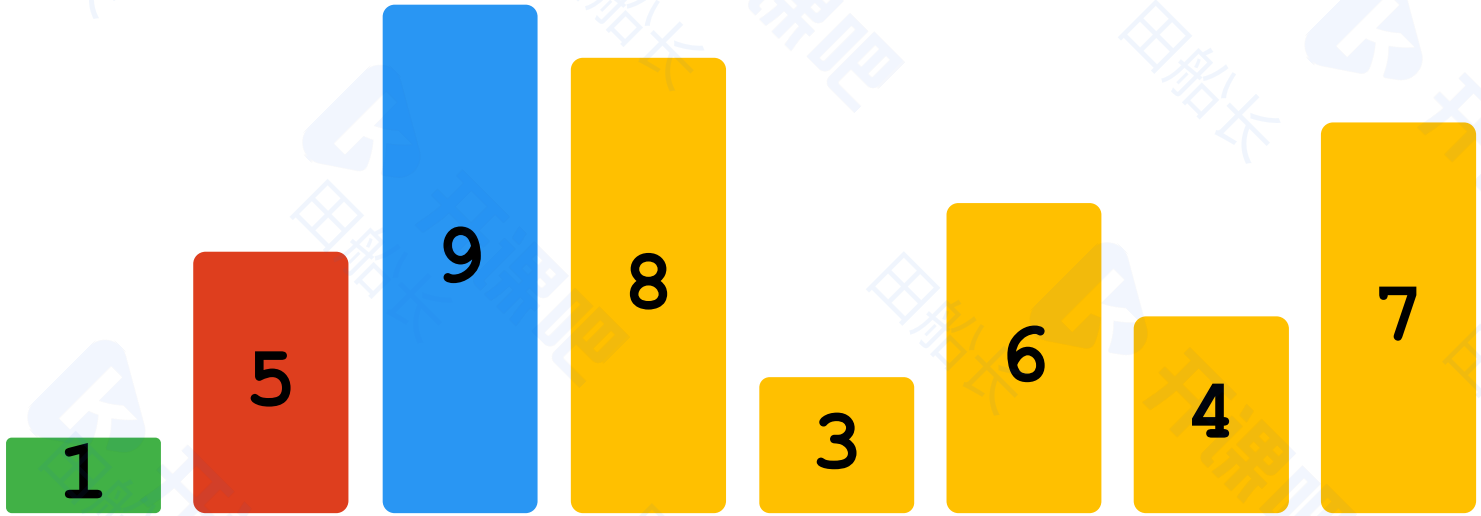
现在从第0个到第1个数
已经有序



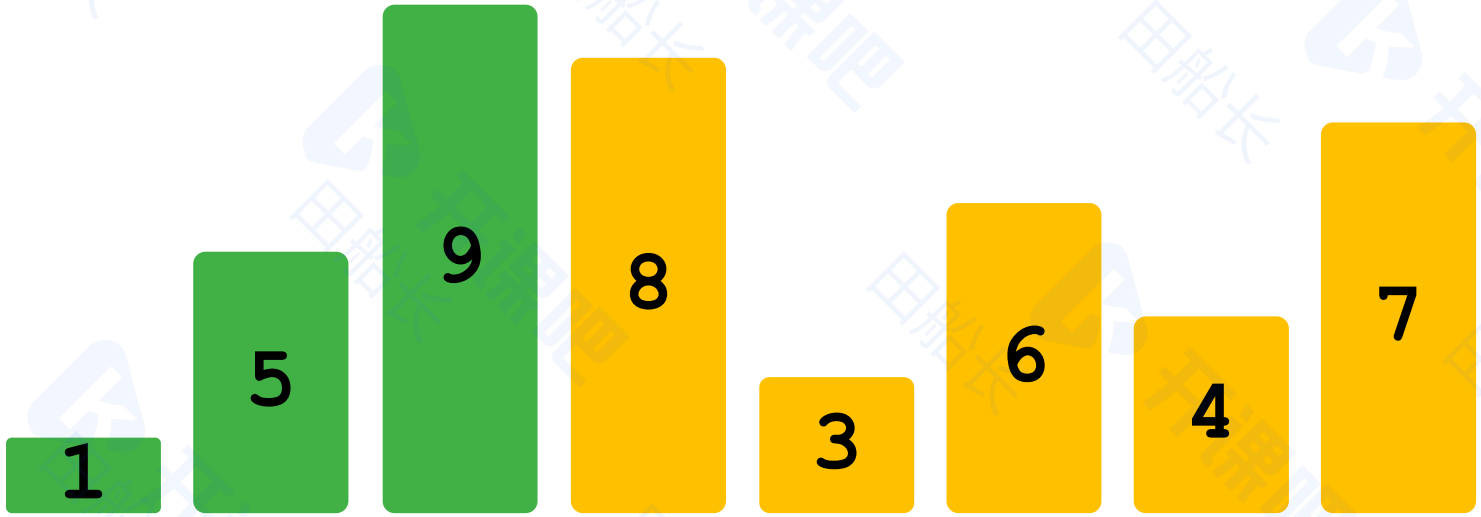
把第2个数字9插入到
由前2个数字组成的
有序数组中



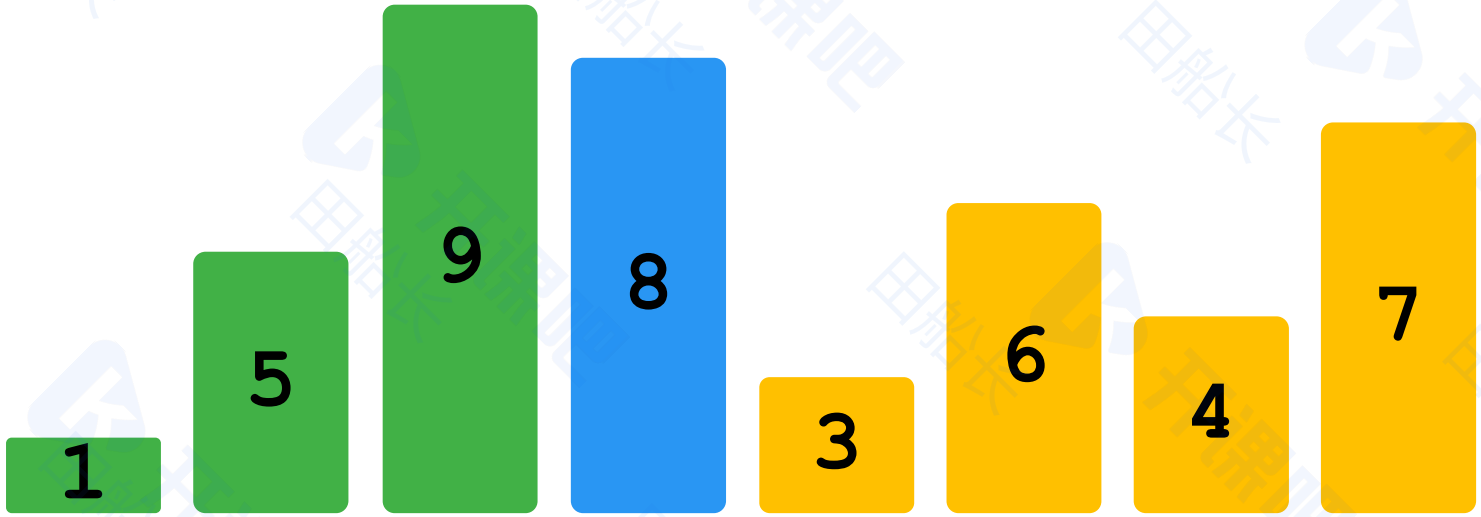
比较9和其前面的5，
看是否为逆序对



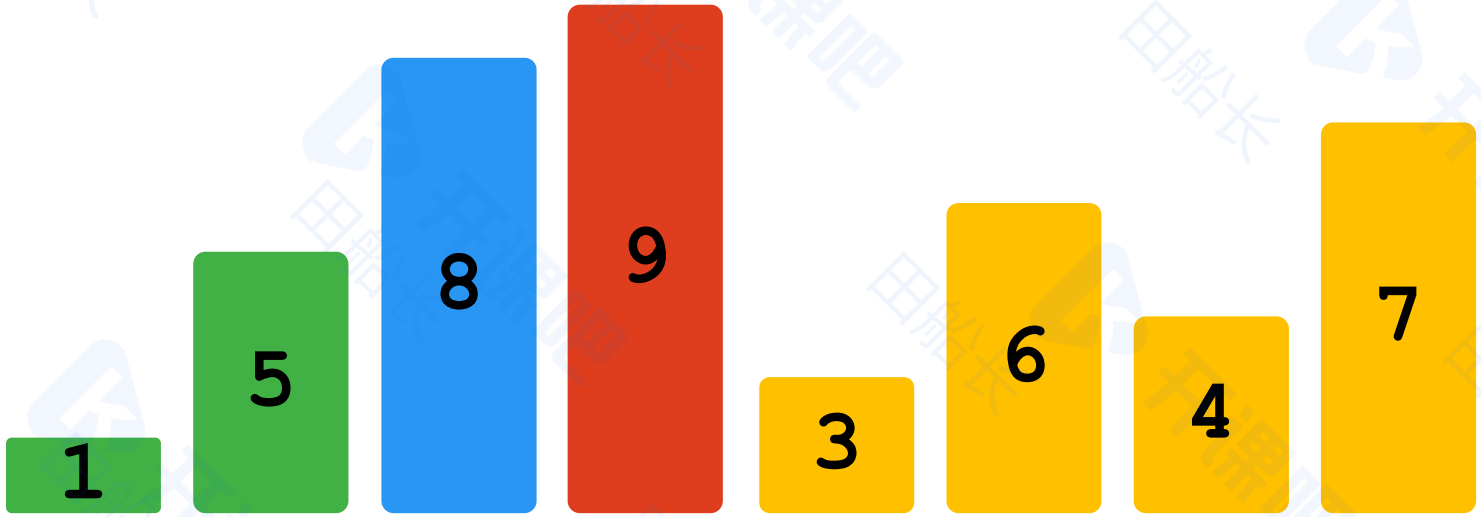
不是逆序对，现在
从第0个到第2个数
已经有序



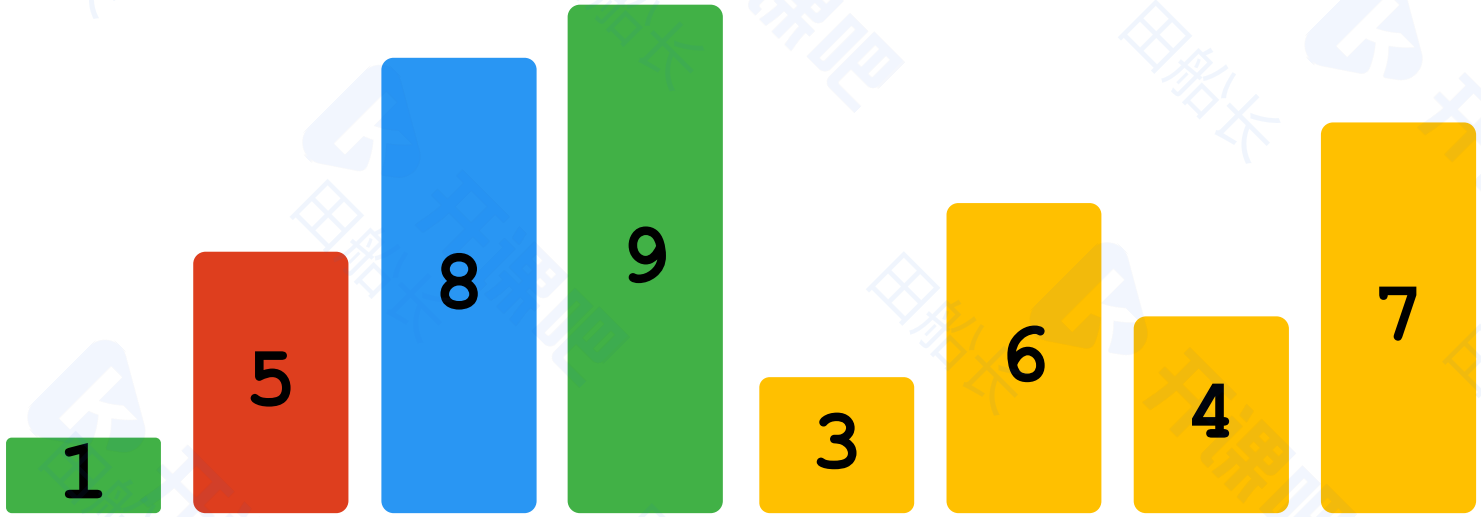
把第3个数字8插入到
由前3个数字组成的
有序数组中



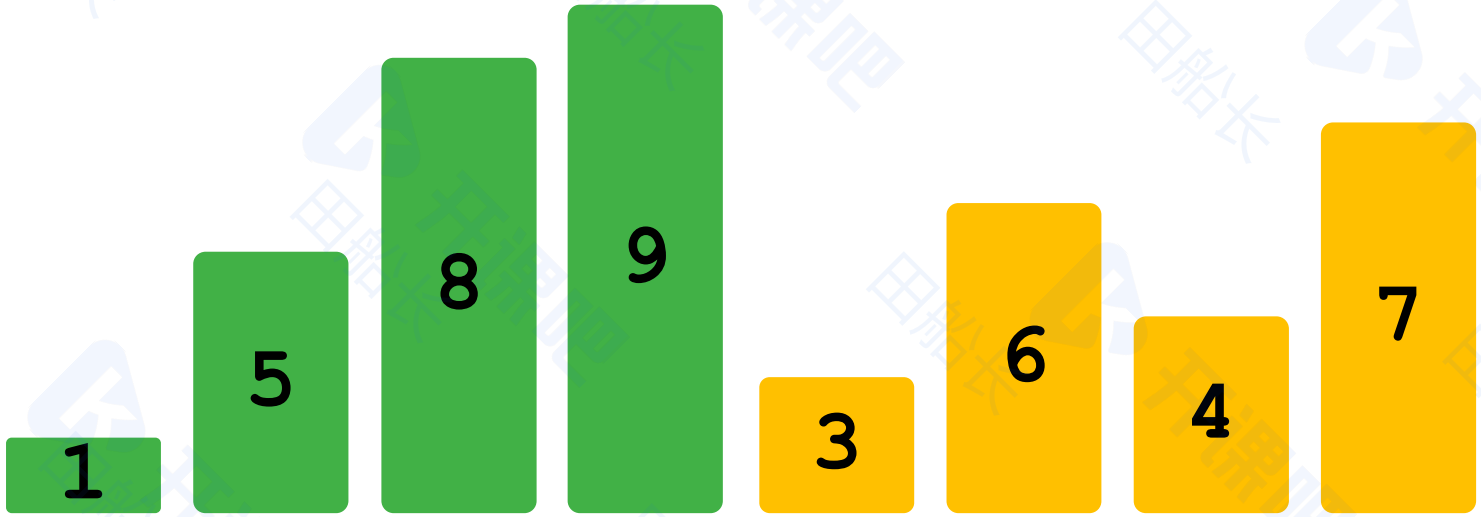
是逆序对，交换9和8的位置，继续向前寻找逆序对



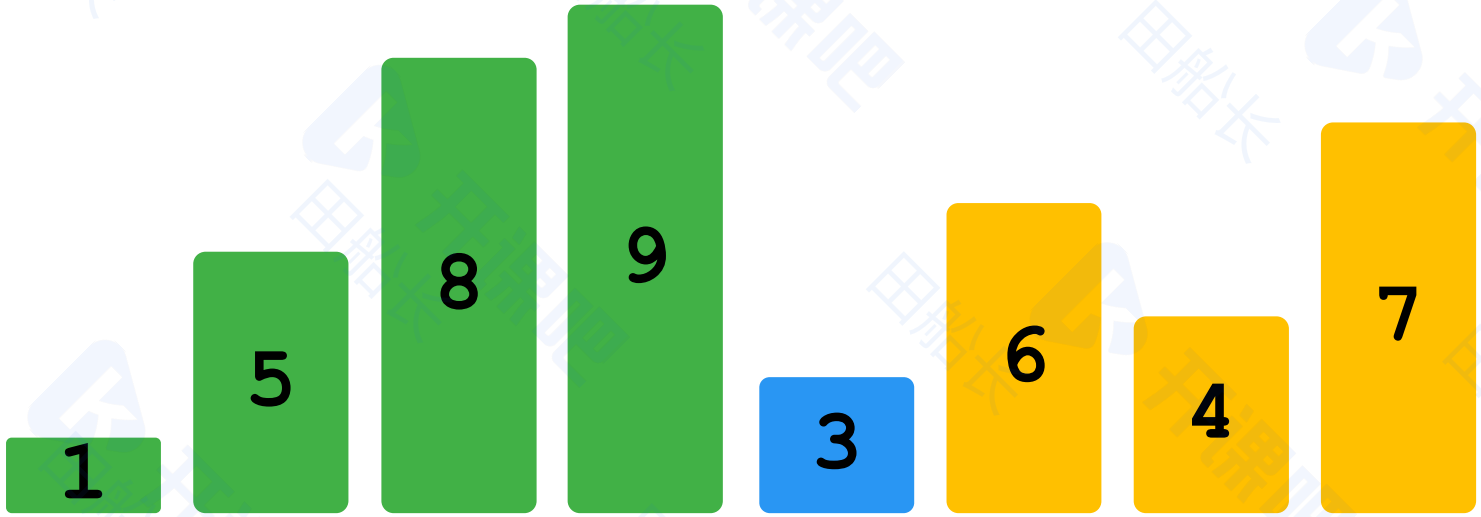
比较8和其前面的5，
看是否为逆序对



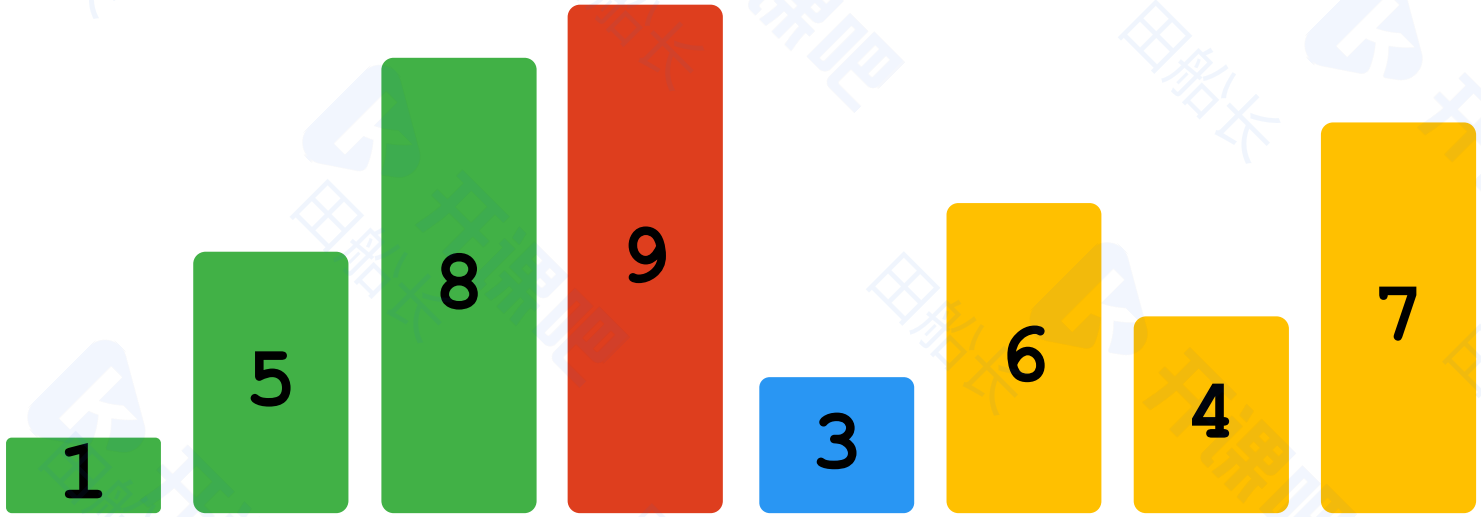
不是逆序对，现在
从第0个到第3个数
已经有序



把第4个数字3插入到由前4个数字组成的有序数组中

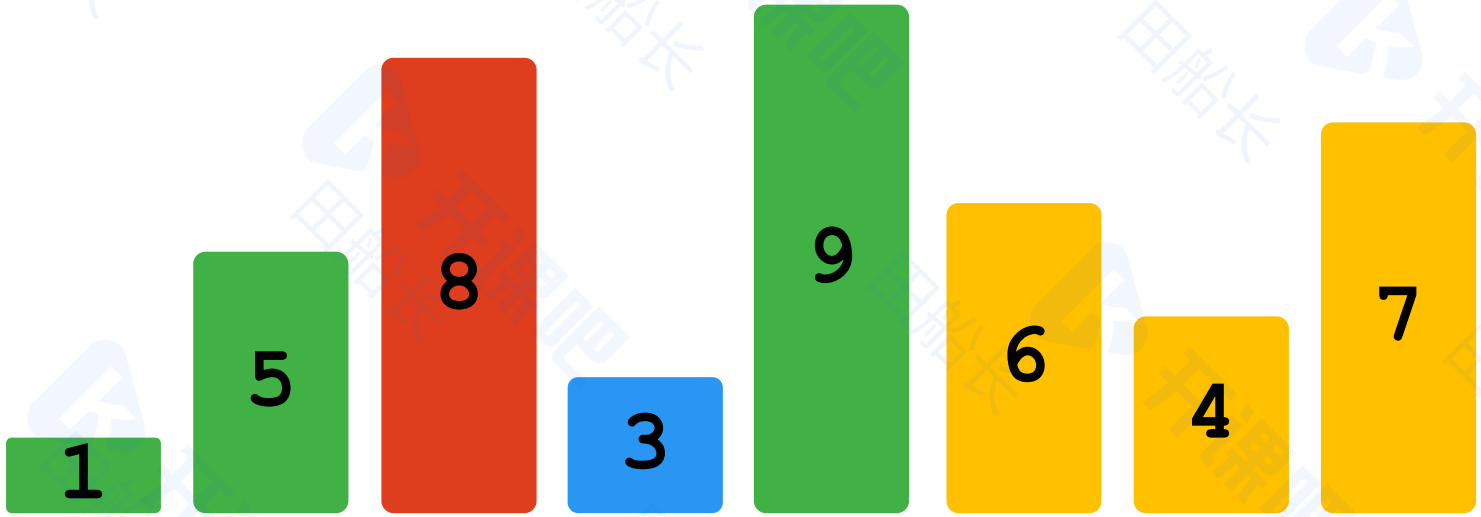


比较3和其前面的9，
看是否为逆序对

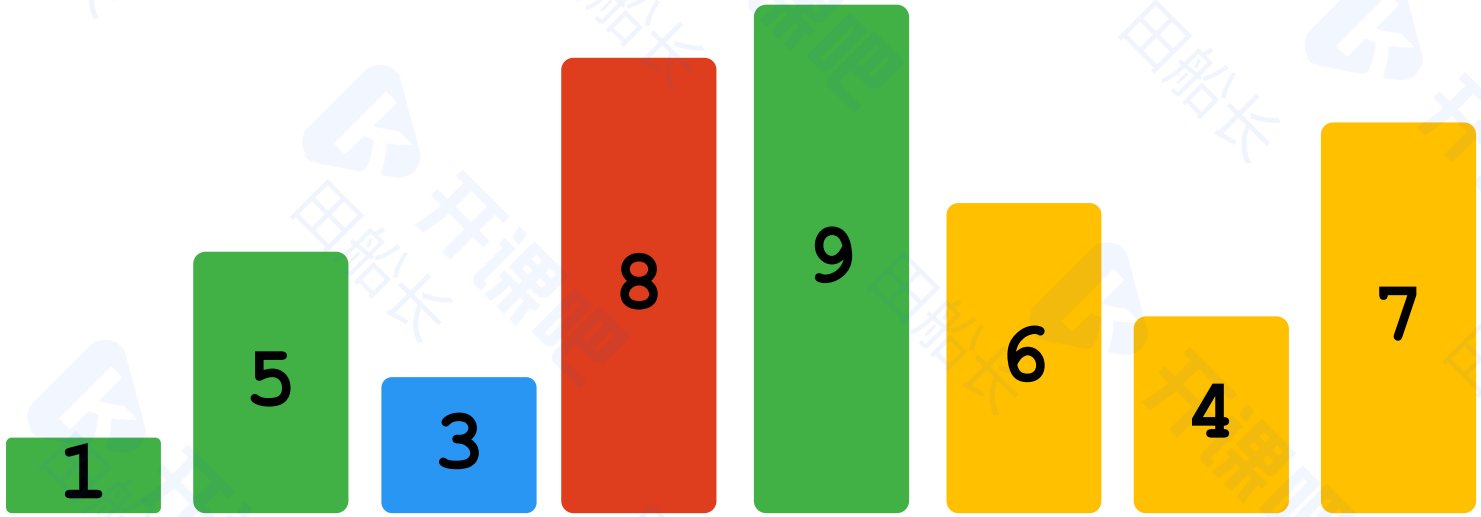


是逆序对，交换9和3的位置，继续向前寻找逆序对

比较3和其前面的8，
看是否为逆序对

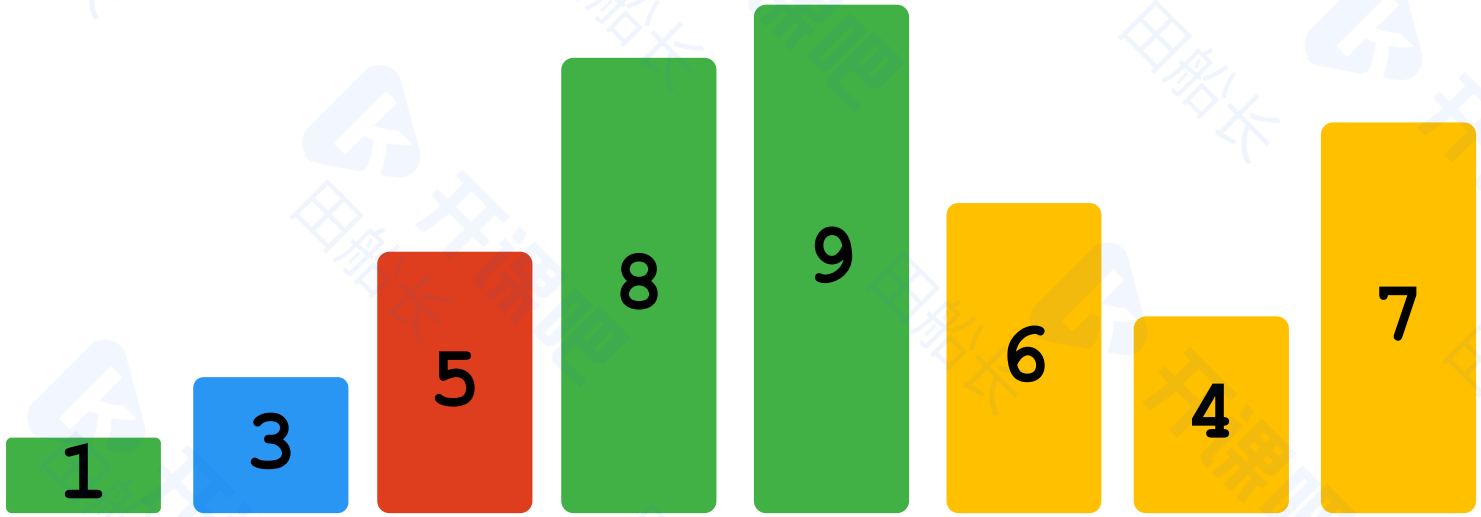


是逆序对，交换8和3的位置，继续向前寻找逆序对

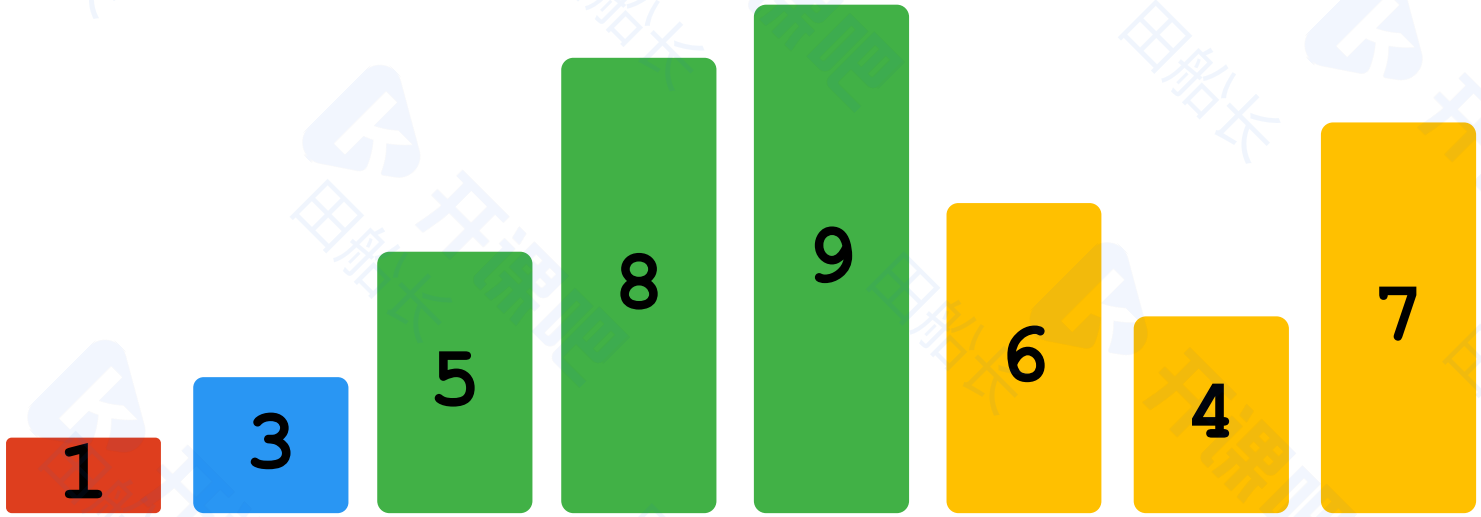


比较3和其前面的5，
看是否为逆序对

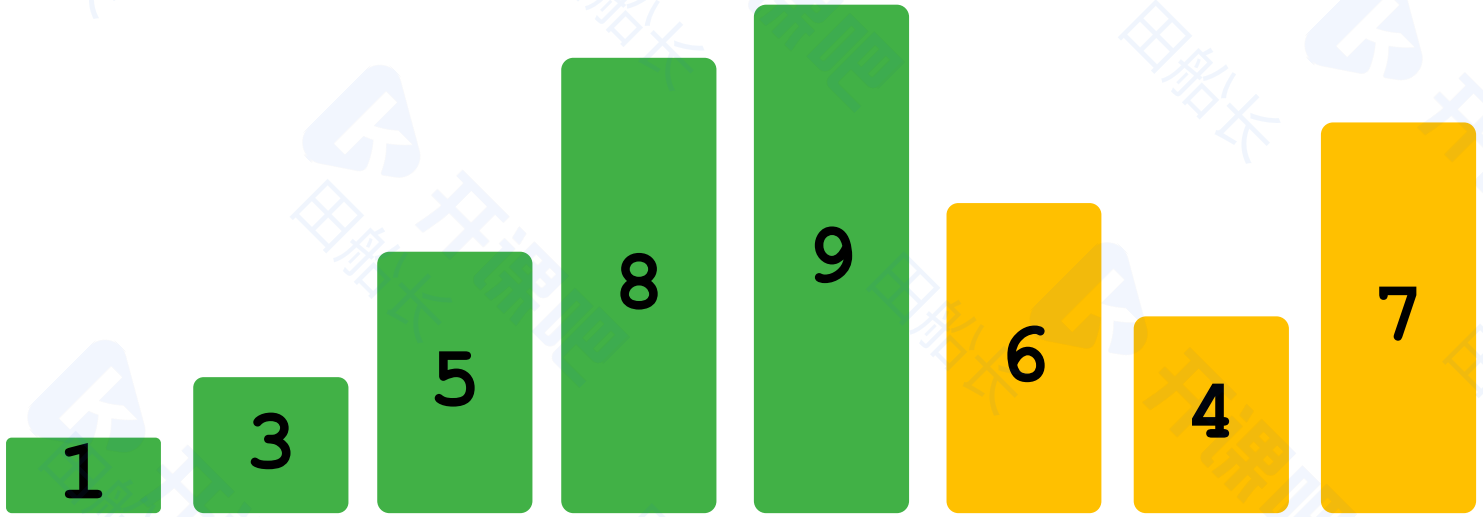
是逆序对，交换5和3的位置，继续向前寻找逆序对



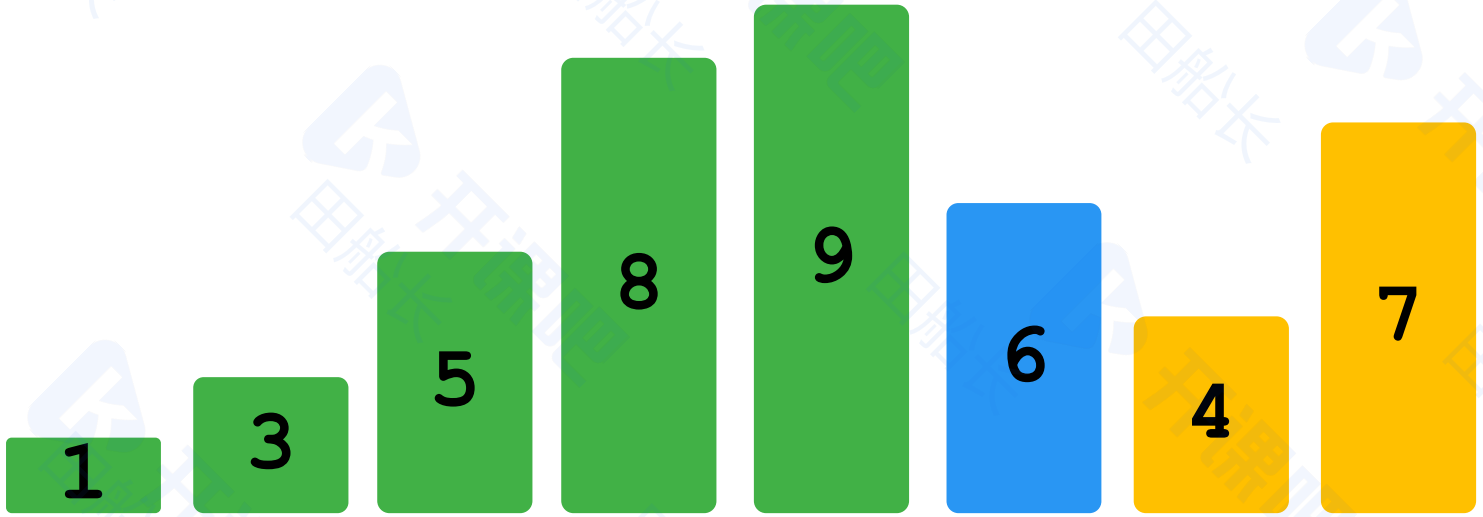
比较3和其前面的1，
看是否为逆序对



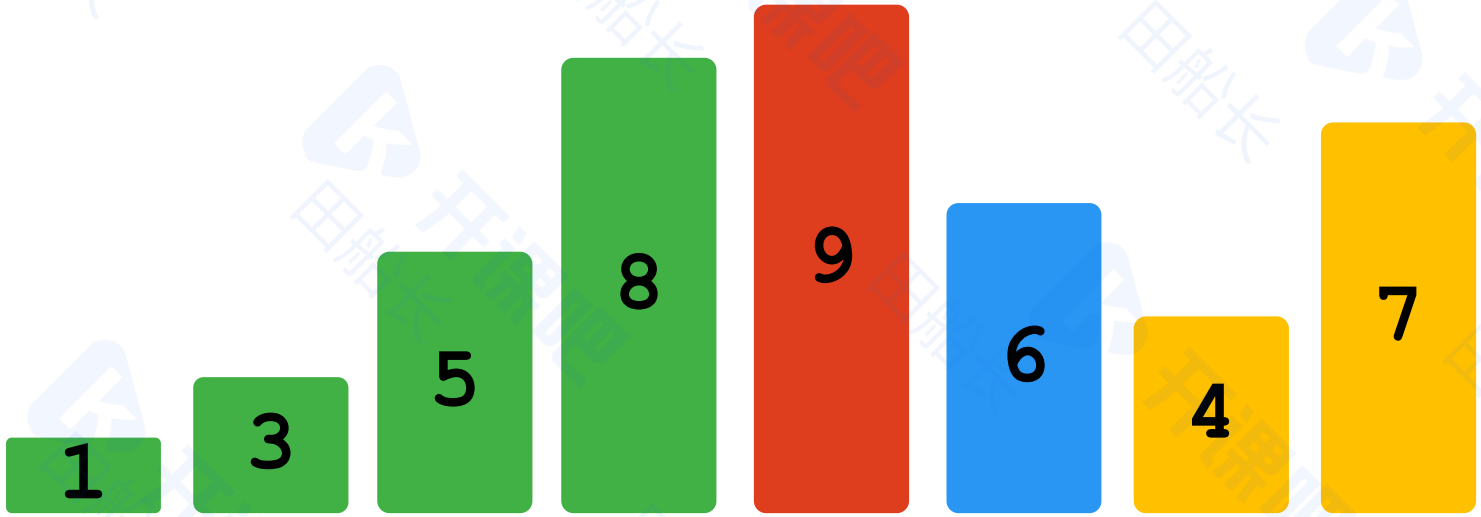
不是逆序对，现在
从第0个到第4个数
已经有序



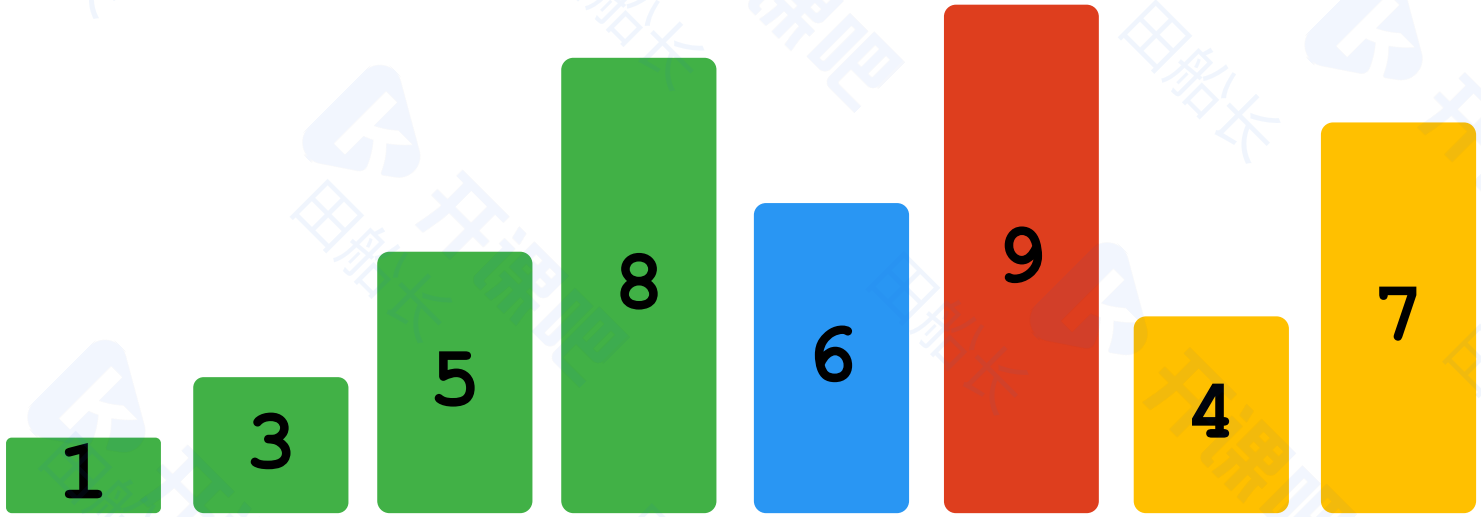
把第5个数字6插入到由前5个数字组成的有序数组中



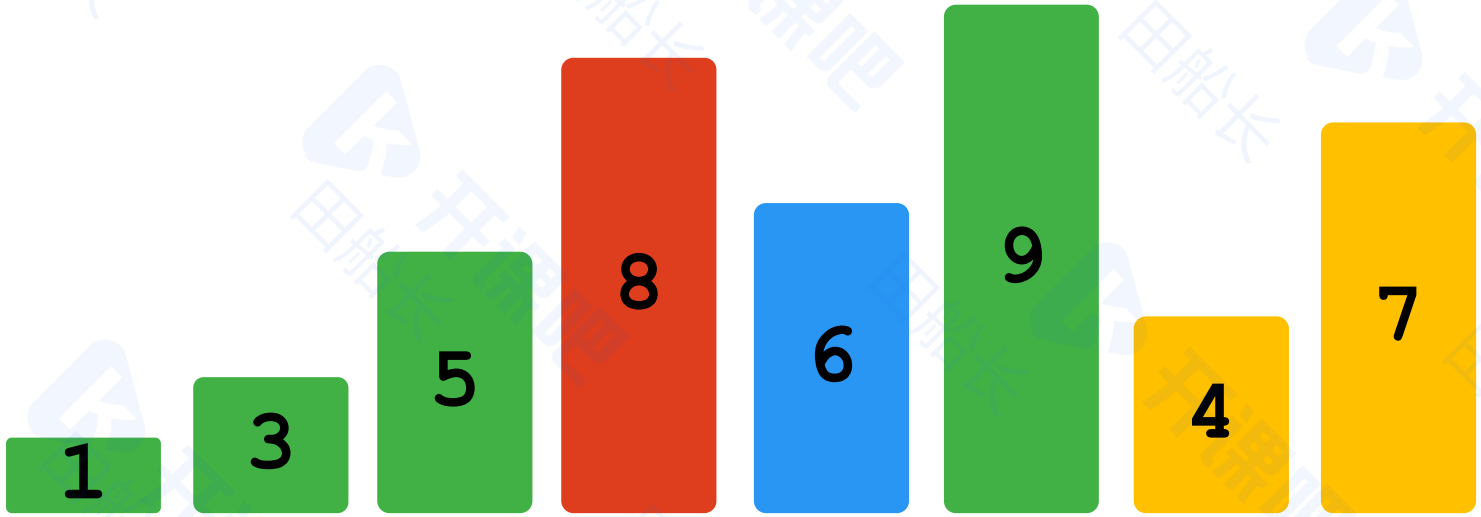
比较6和其前面的9，
看是否为逆序对



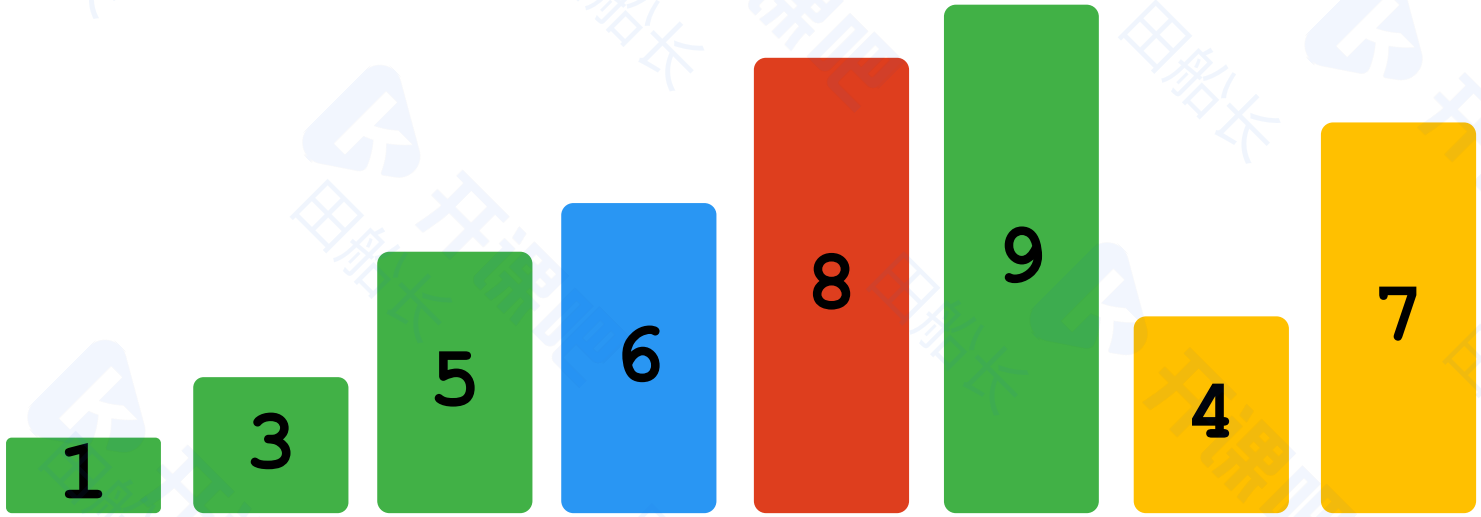
是逆序对，交换9和6的位置，继续向前寻找逆序对



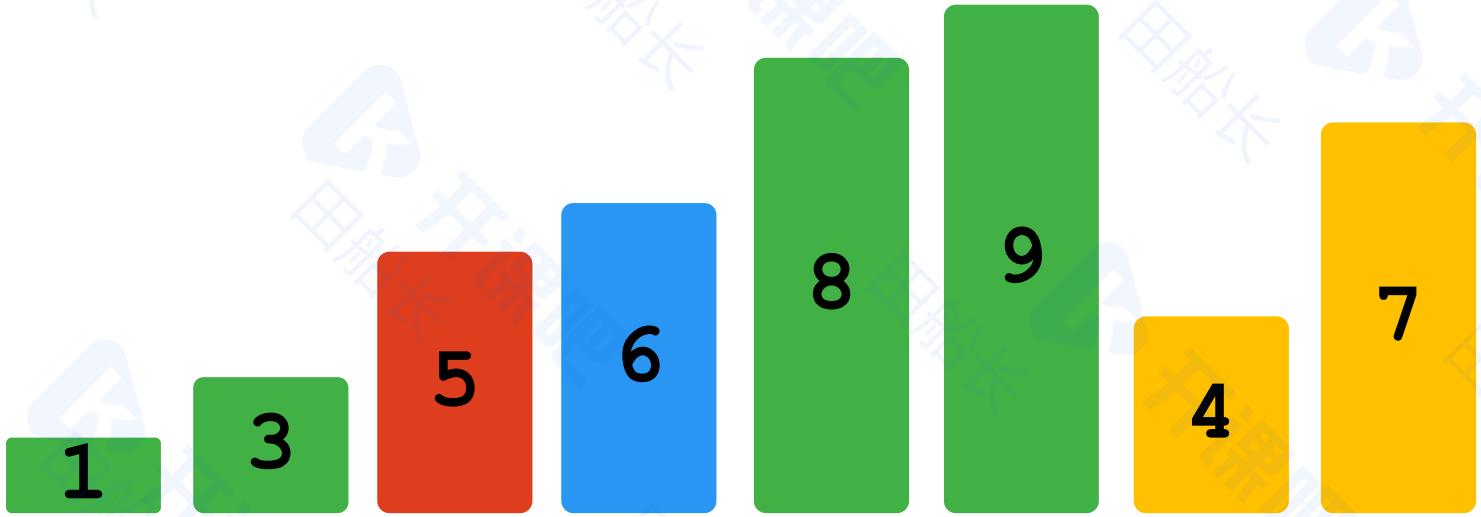
比较6和其前面的8，
看是否为逆序对



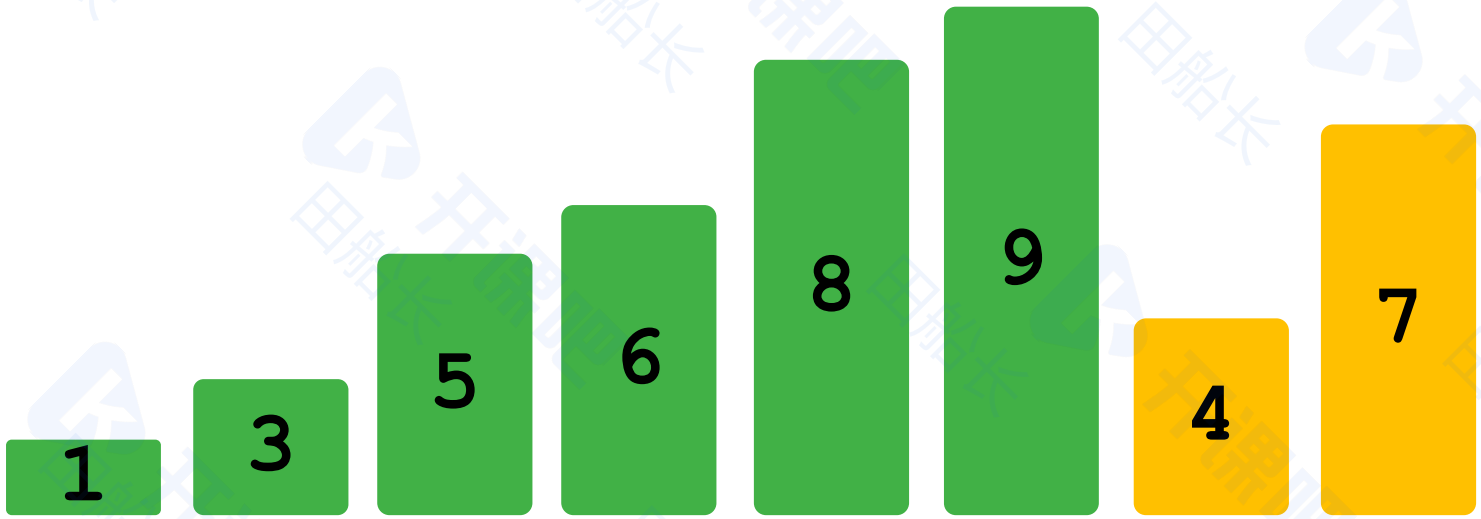
是逆序对，交换8和6的位置，继续向前寻找逆序对



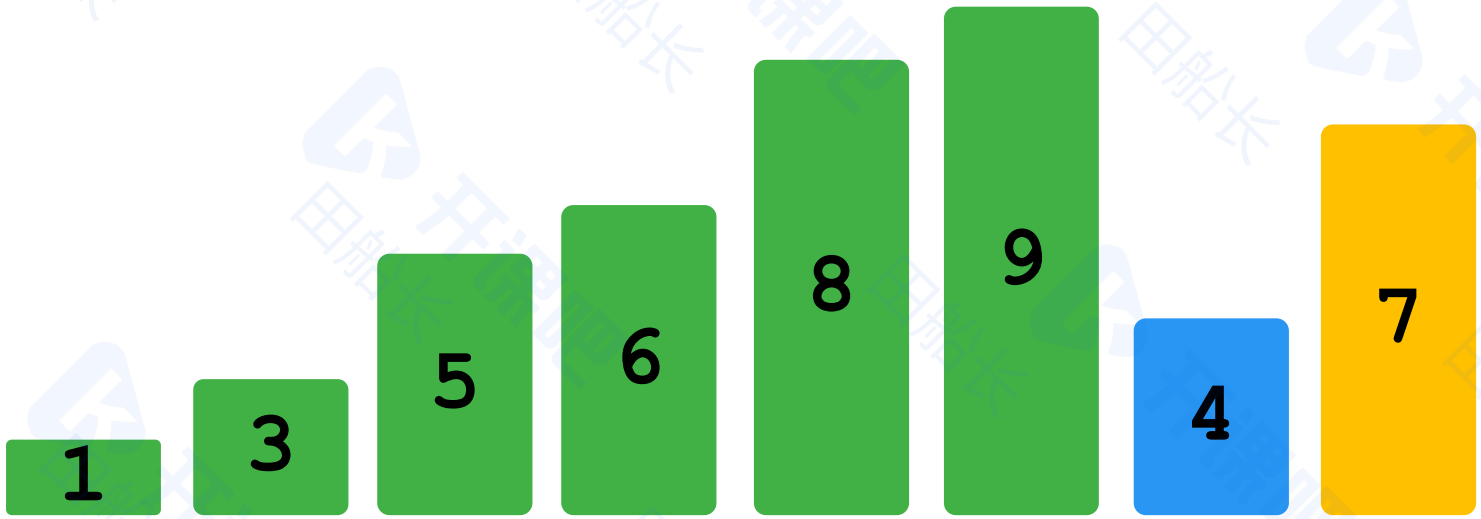
比较6和其前面的5，
看是否为逆序对



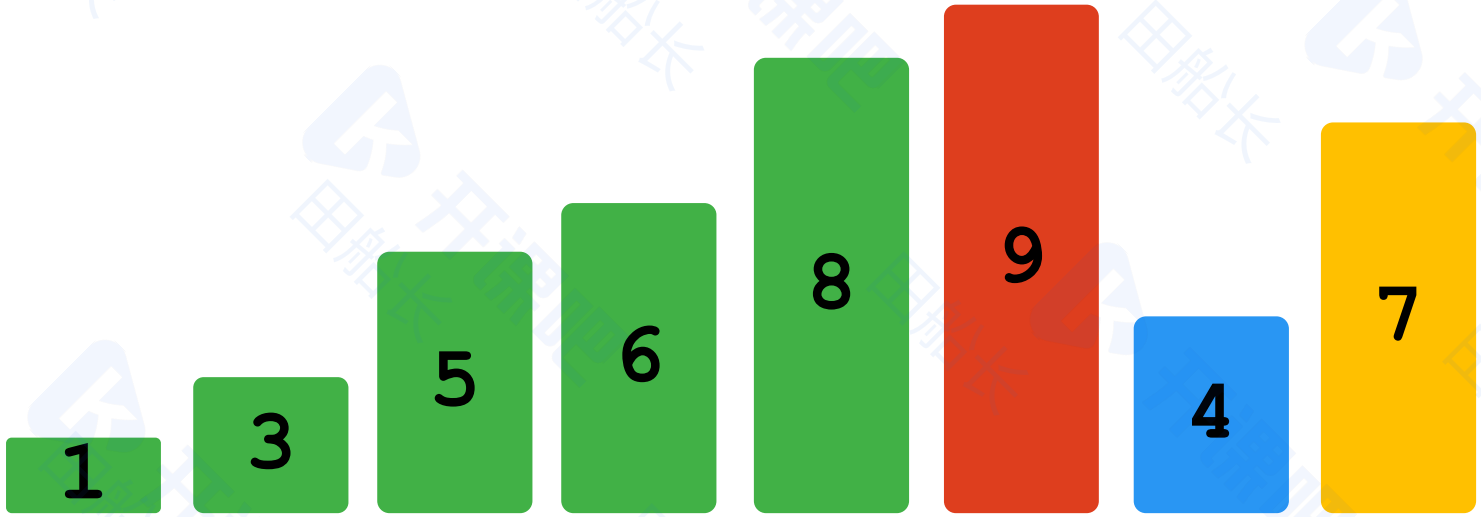
不是逆序对，现在
从第0个到第5个数
已经有序



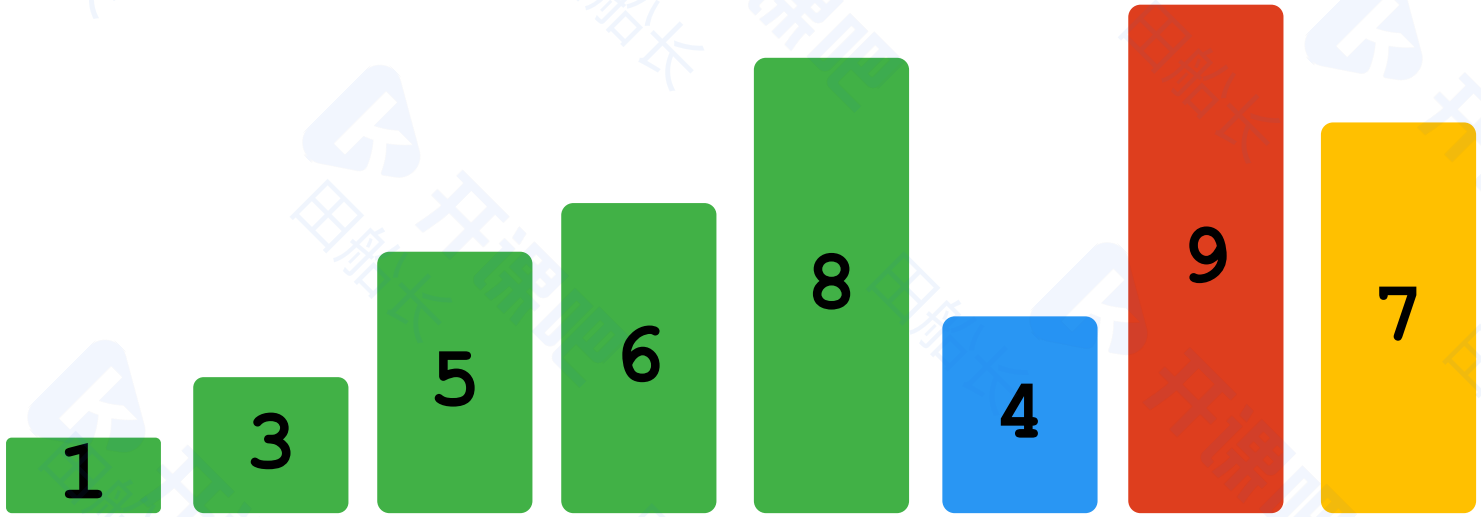
把第6个数字4插入到由前6个数字组成的有序数组中



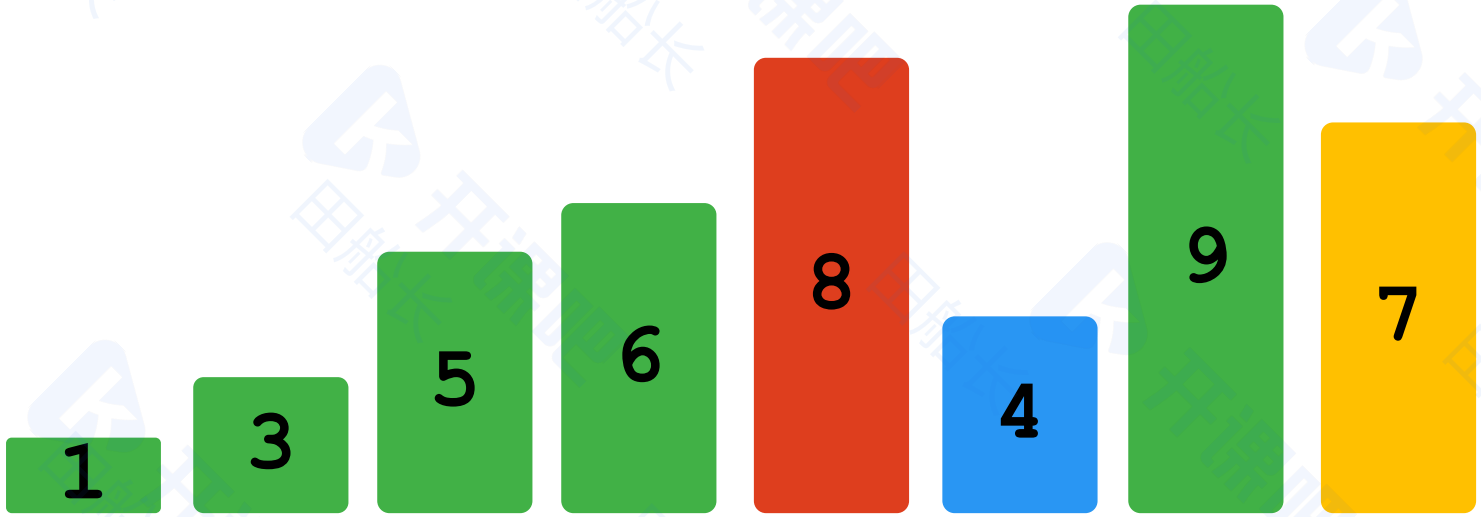
比较4和其前面的9，
看是否为逆序对



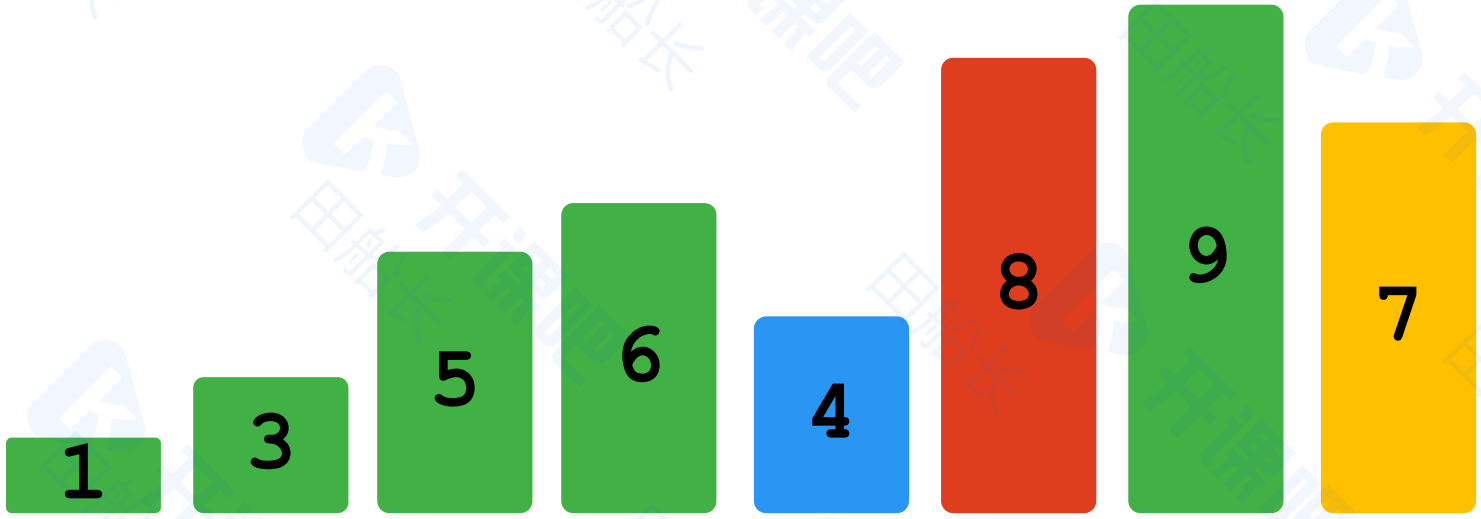
是逆序对，交换9和4的位置，继续向前寻找逆序对



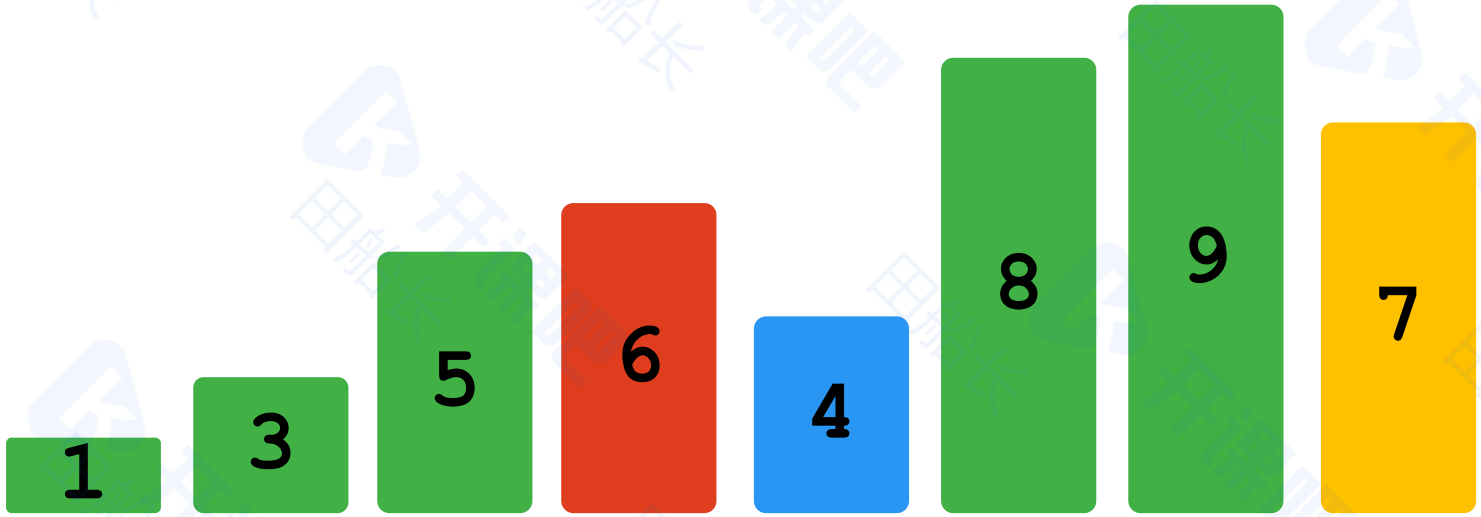
比较4和其前面的8，
看是否为逆序对



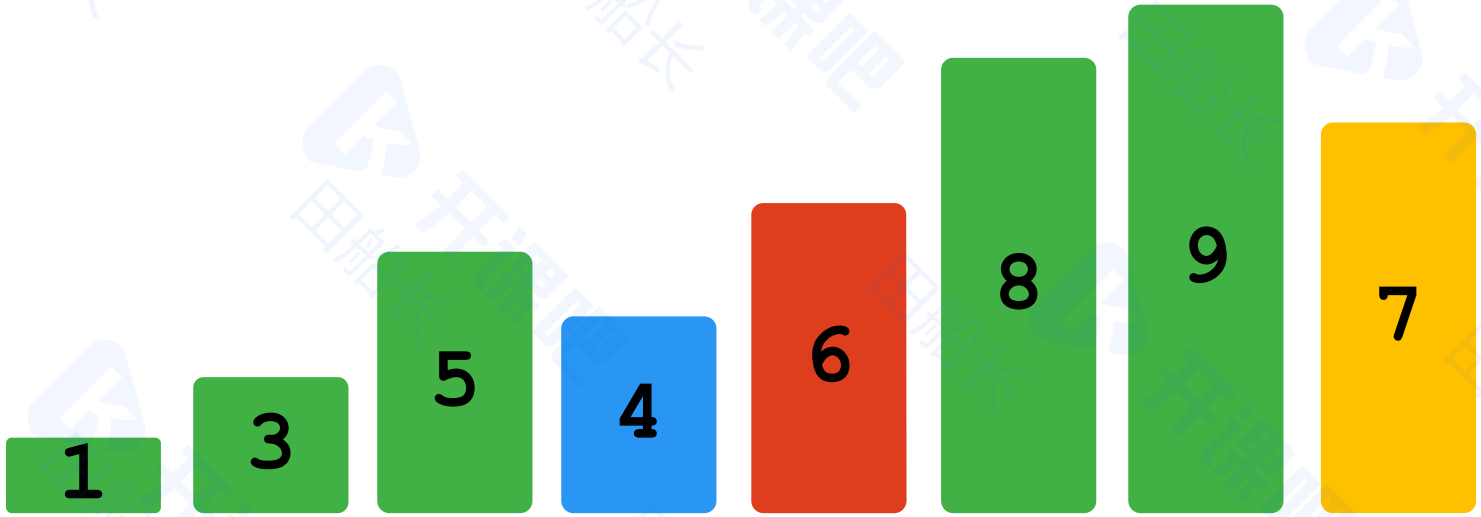
是逆序对，交换8和4的位置，继续向前寻找逆序对



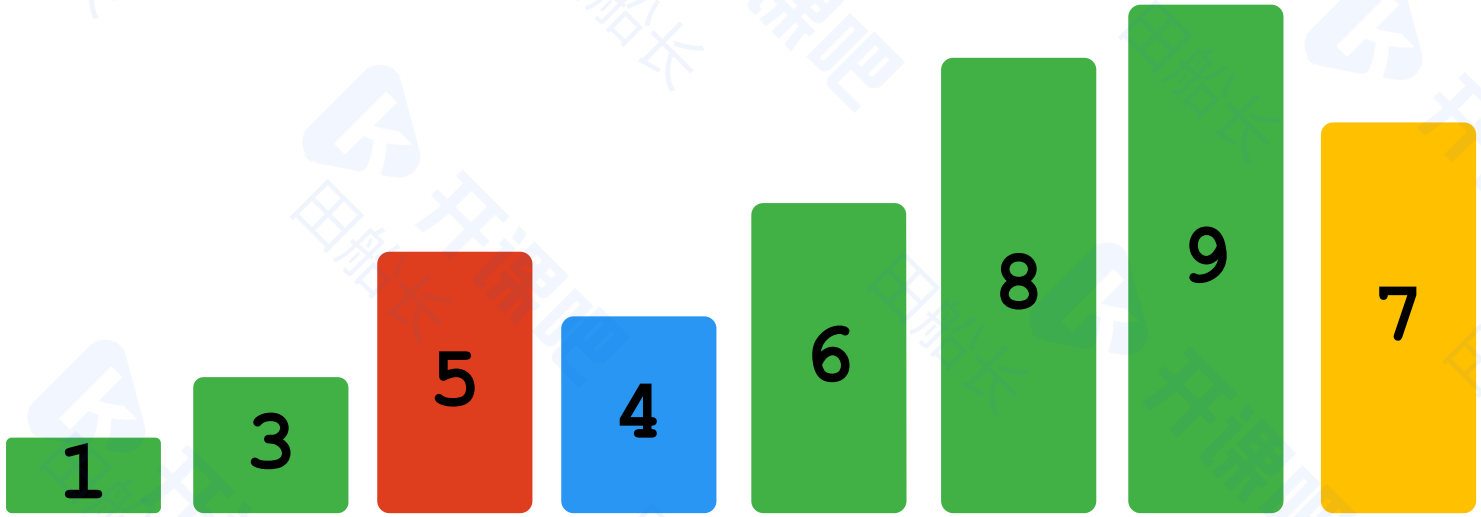
比较4和其前面的6，
看是否为逆序对



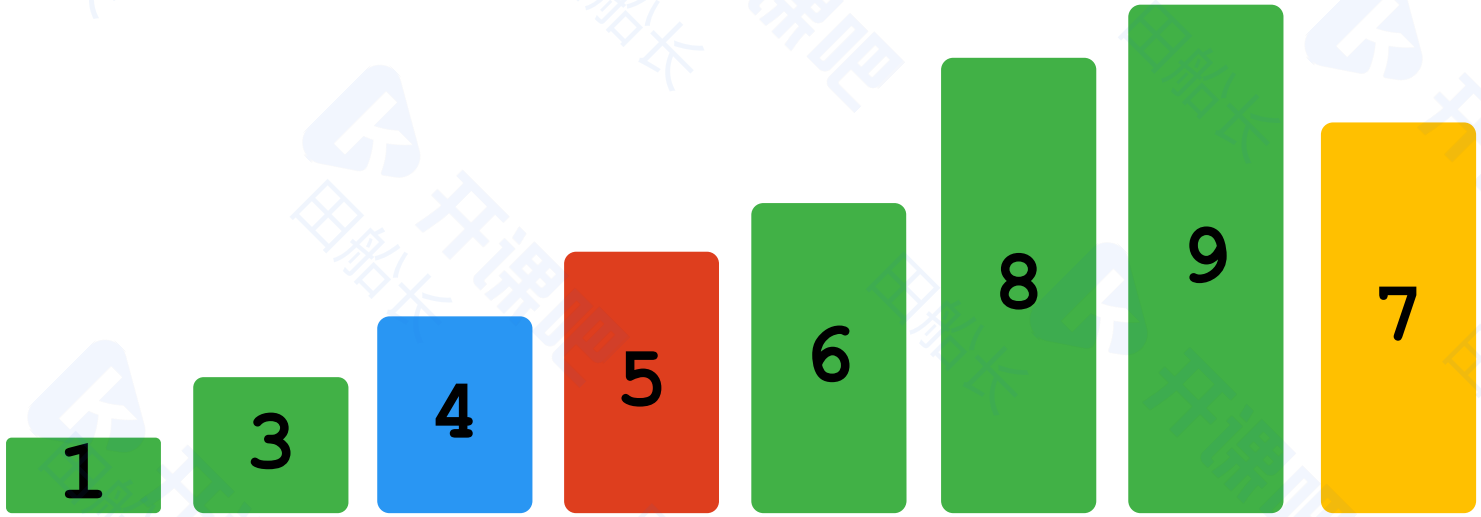
是逆序对，交换6和4的位置，继续向前寻找逆序对



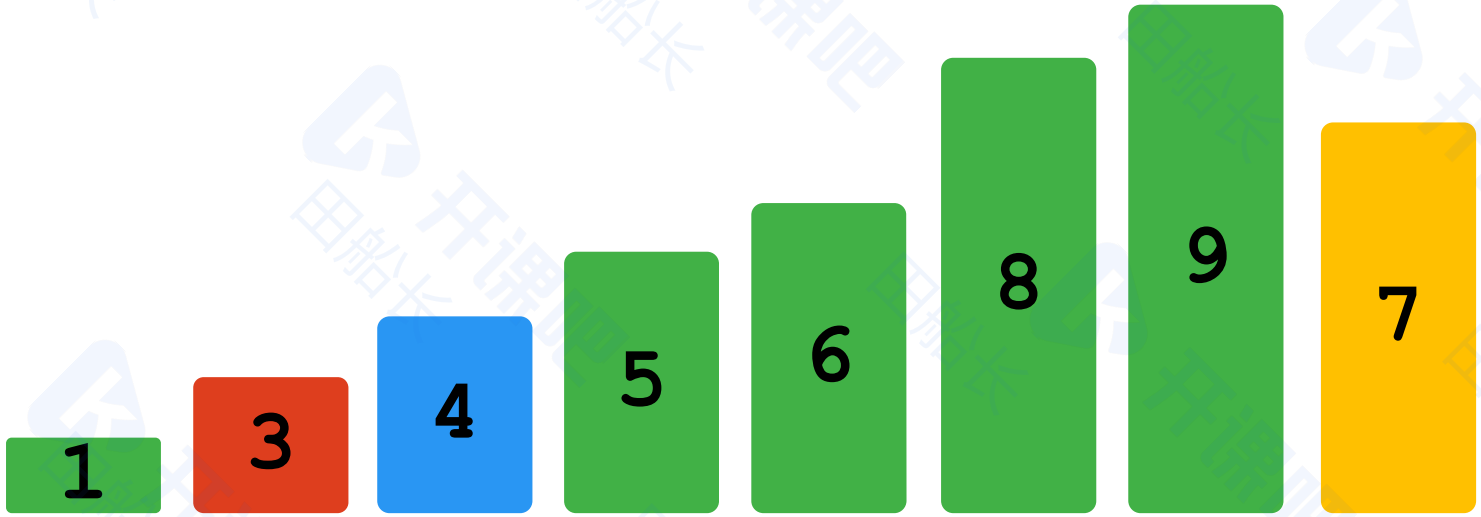
比较4和其前面的5，
看是否为逆序对



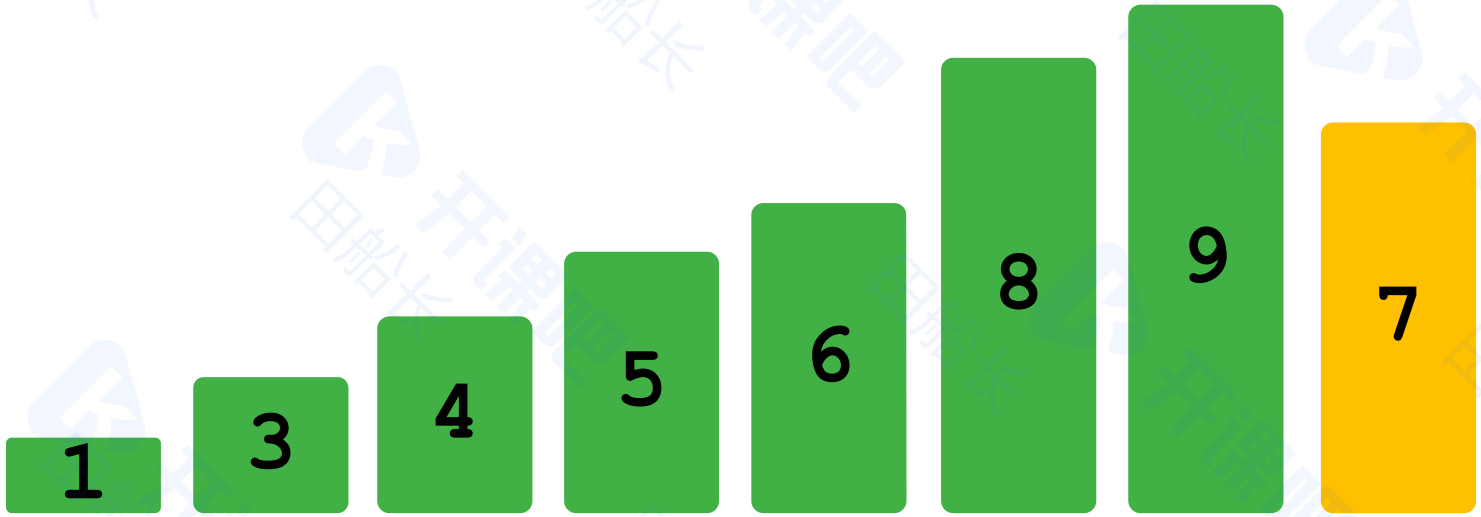
是逆序对，交换5和4的位置，继续向前寻找逆序对



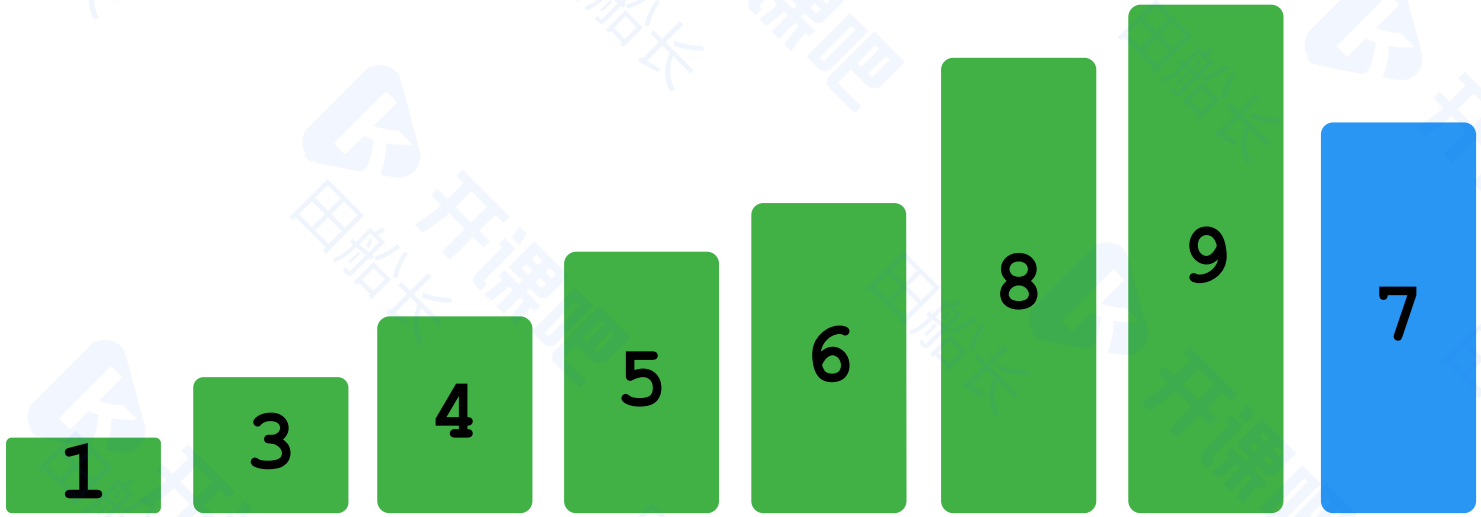
比较4和其前面的3，
看是否为逆序对



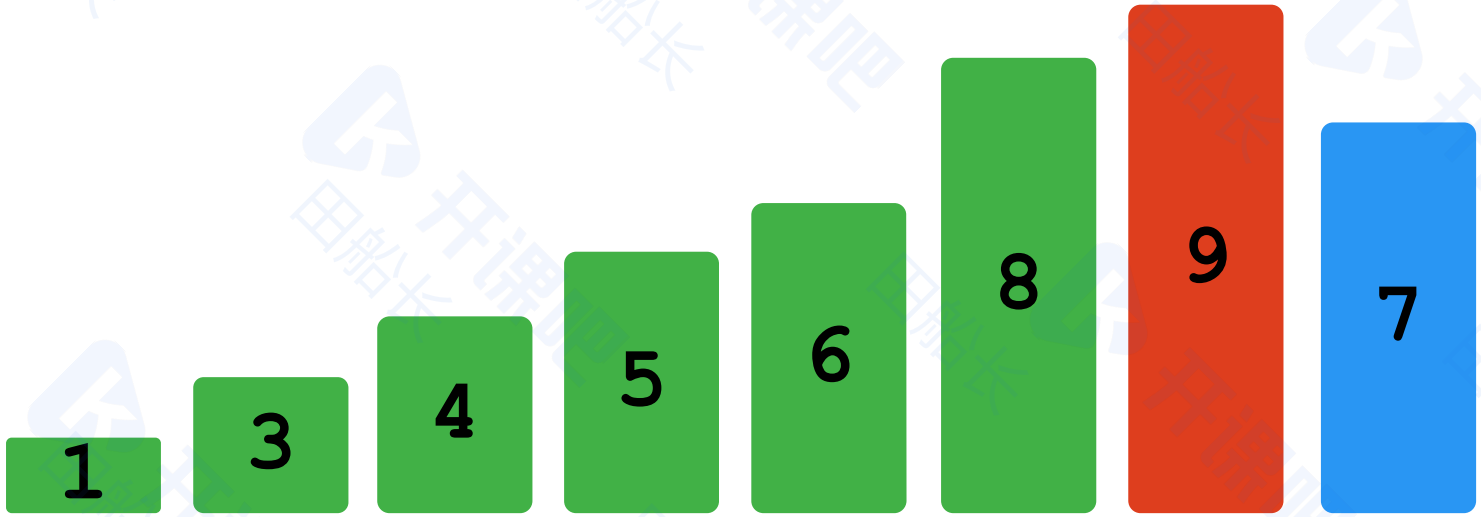
不是逆序对，现在
从第0个到第6个数
已经有序



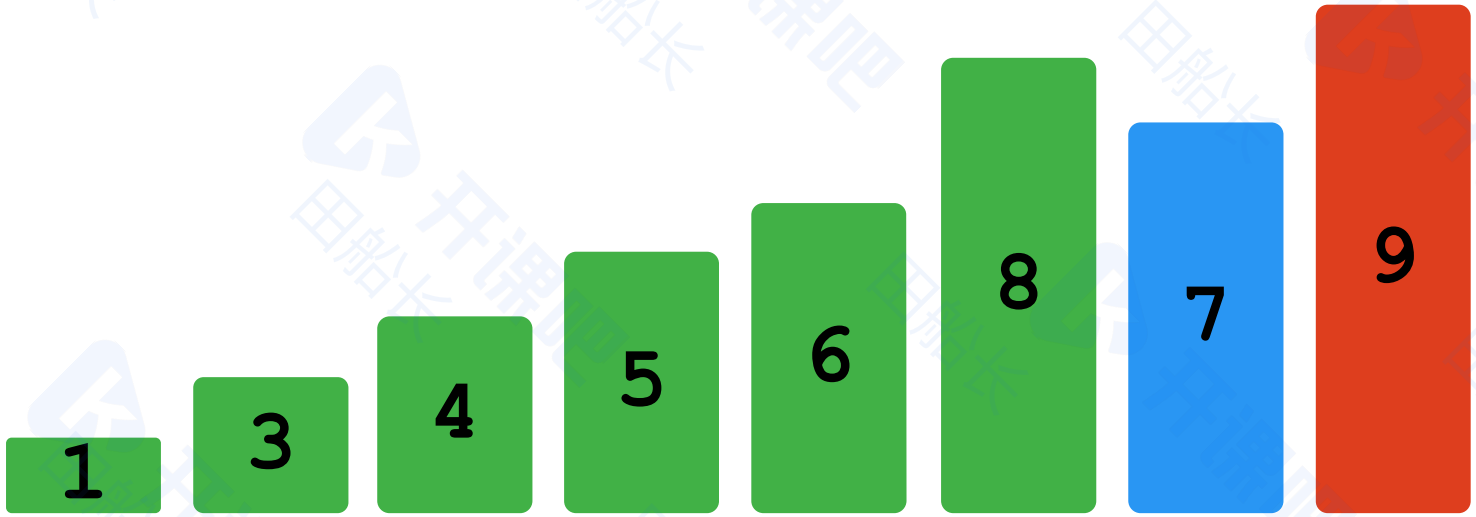
把第7个数字7插入到由前7个数字组成的有序数组中



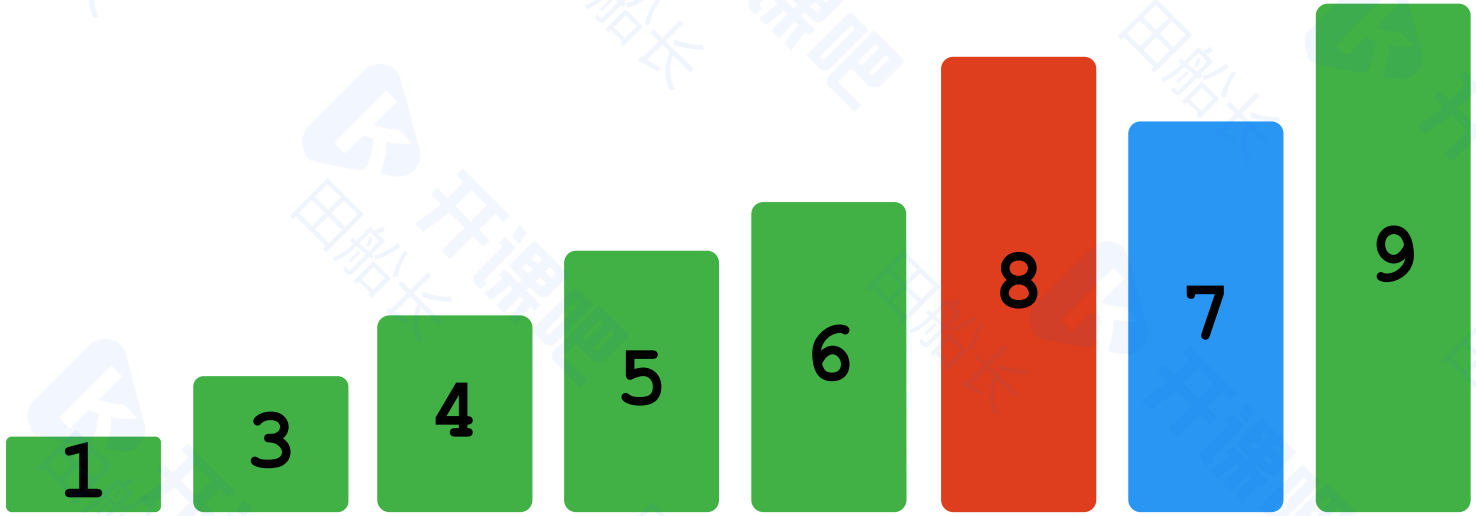
比较7和其前面的9，
看是否为逆序对



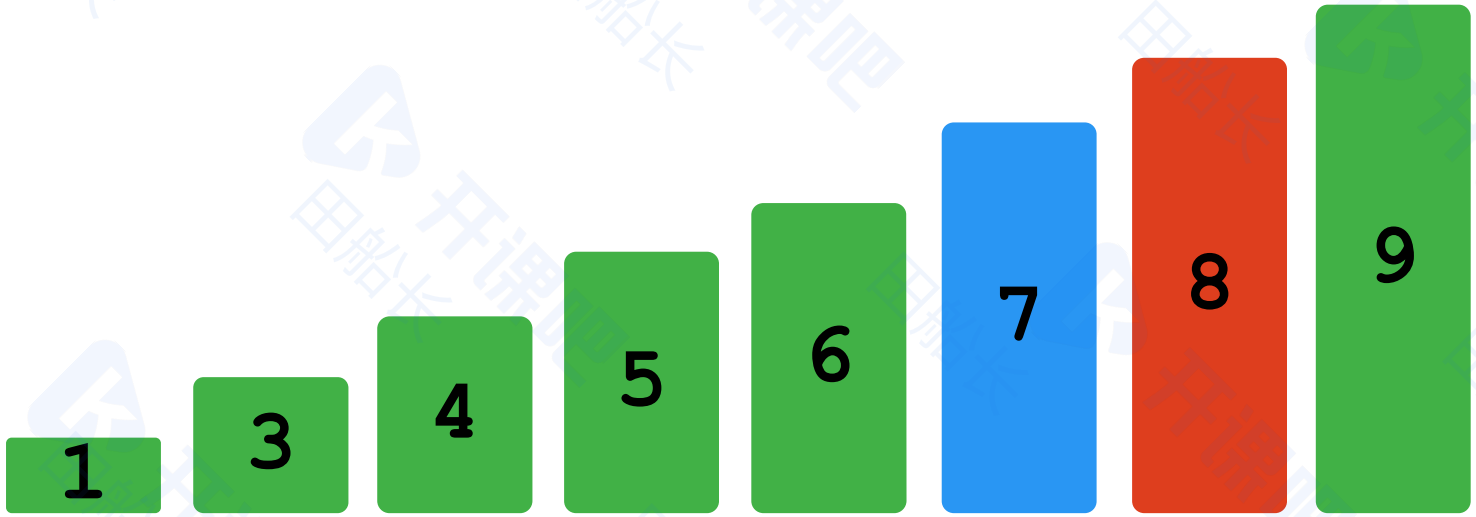
是逆序对，交换9和7的位置，继续向前寻找逆序对



比较7和其前面的8，
看是否为逆序对

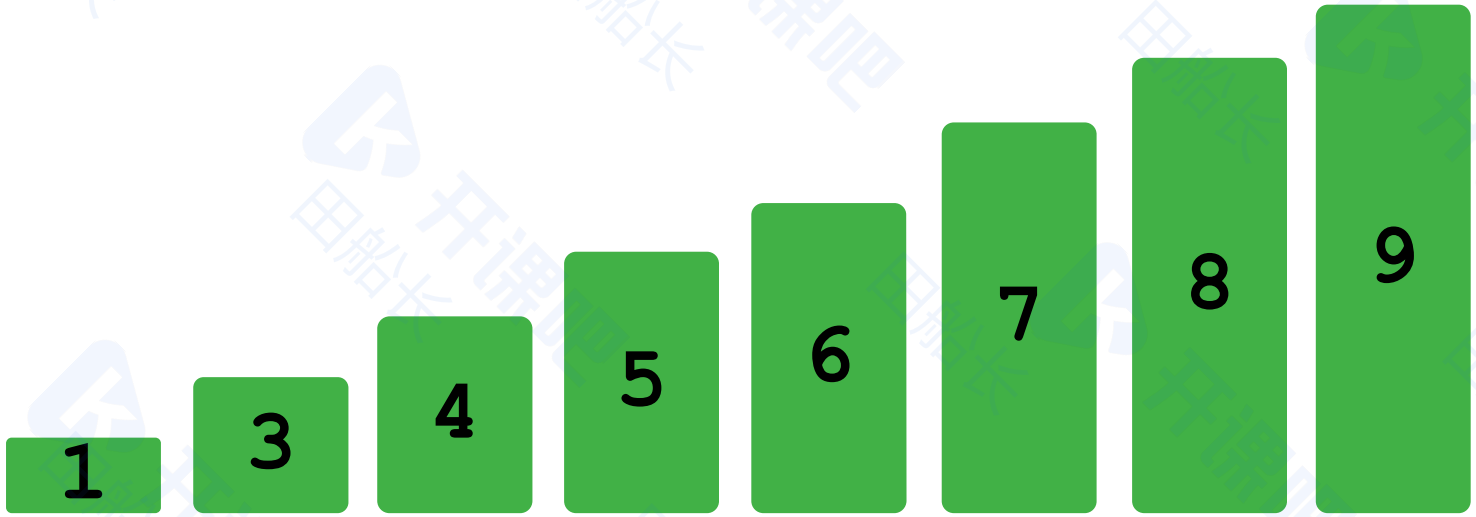


是逆序对，交换8和7的位置，继续向前寻找逆序对



比较7和其前面的6，
看是否为逆序对

不是逆序对，现在
从第0个到第7个数
已经有序



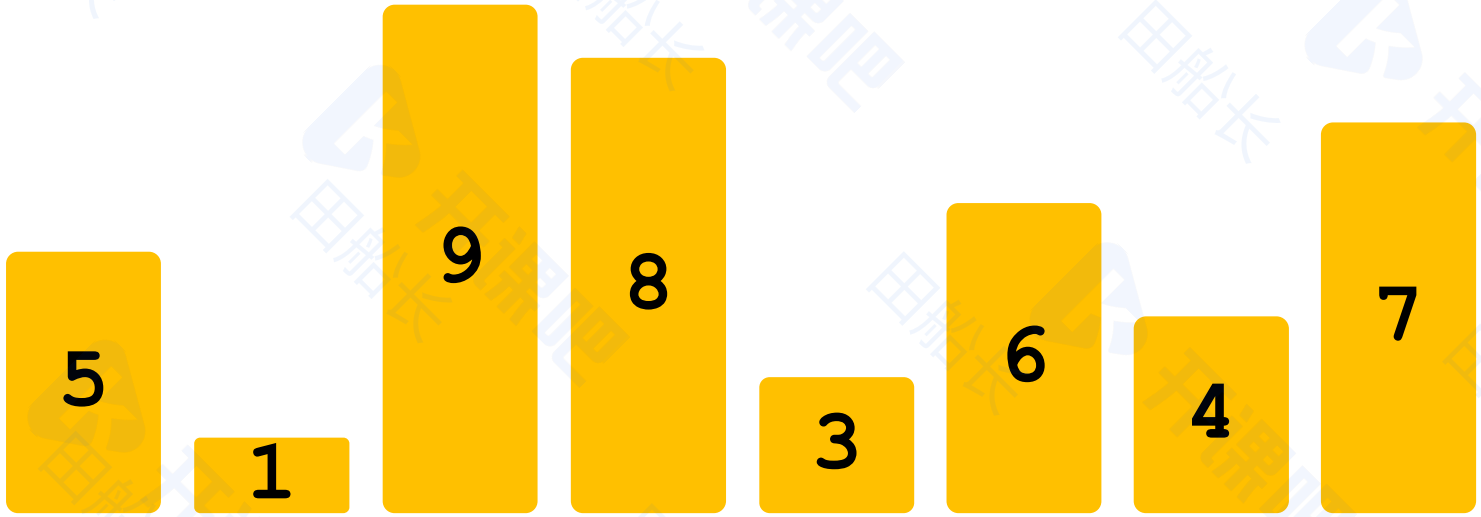
插入排序结束

插入类排序——折半插入排序

与直接插入排序类似
只是在寻找元素插入位置时
改用折半查找法进行查找

插入类排序——希尔排序

根据不同的增量，将数据分组
对于每组数据，分别进行直接插入排序
然后缩小增量，重复这个过程
最终当增量为1时，相当于进行了直接插入排序
这轮排序后，数据整体有序

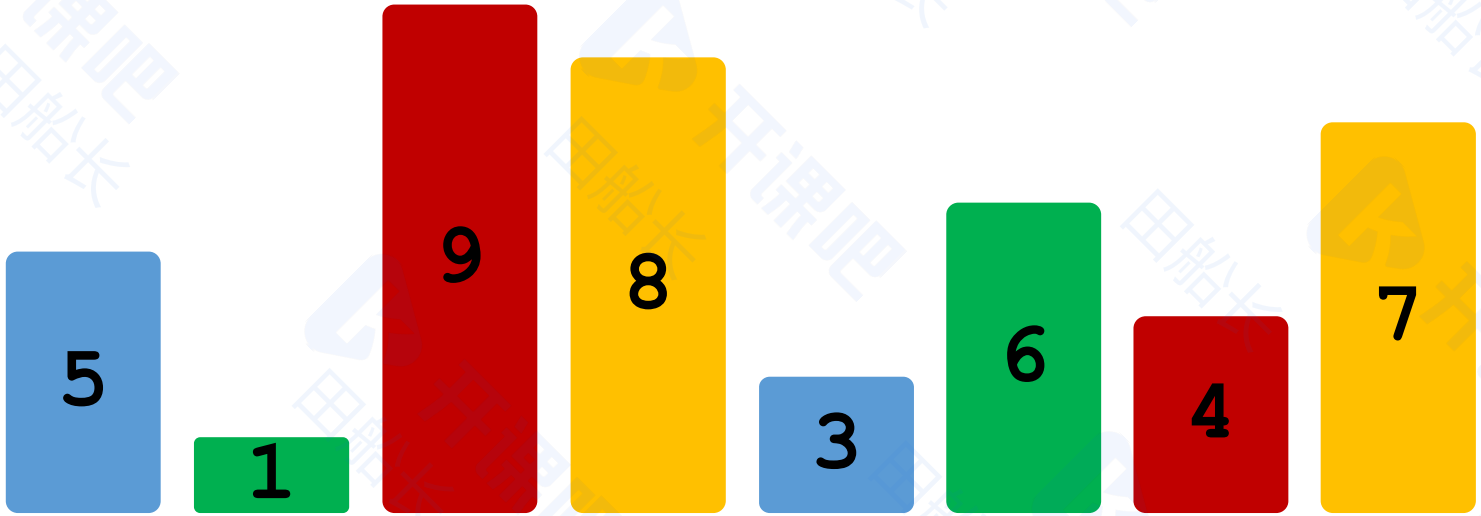


第一趟排序

确定第一组增量

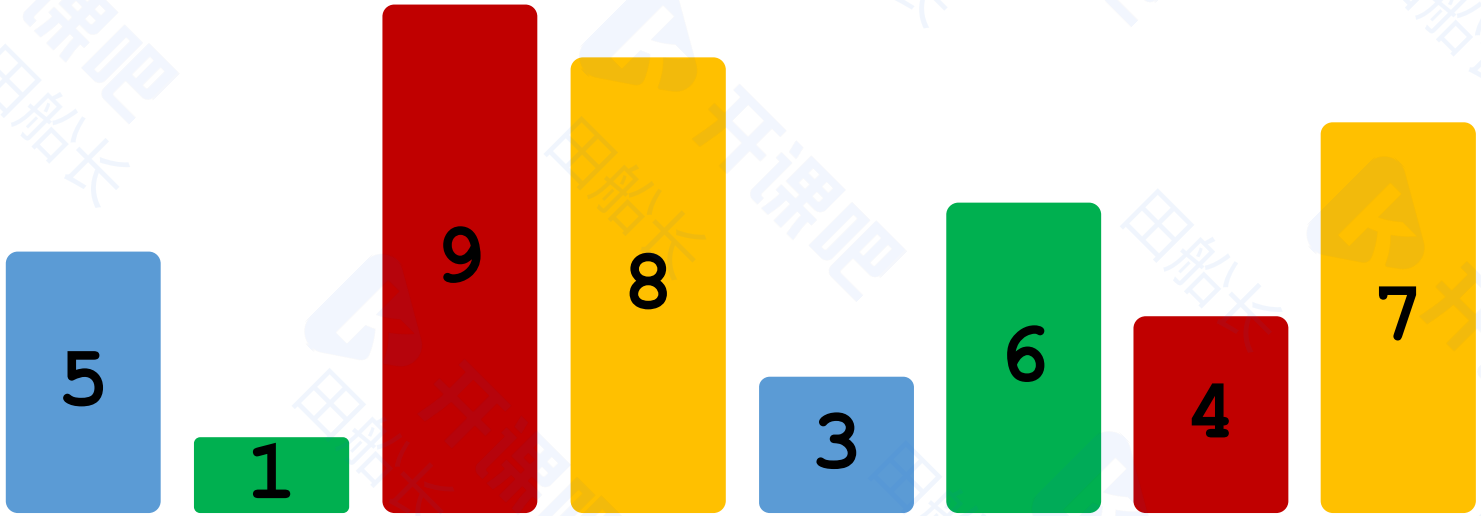
D1 = n / 2 = 4

以间隔D1进行分组



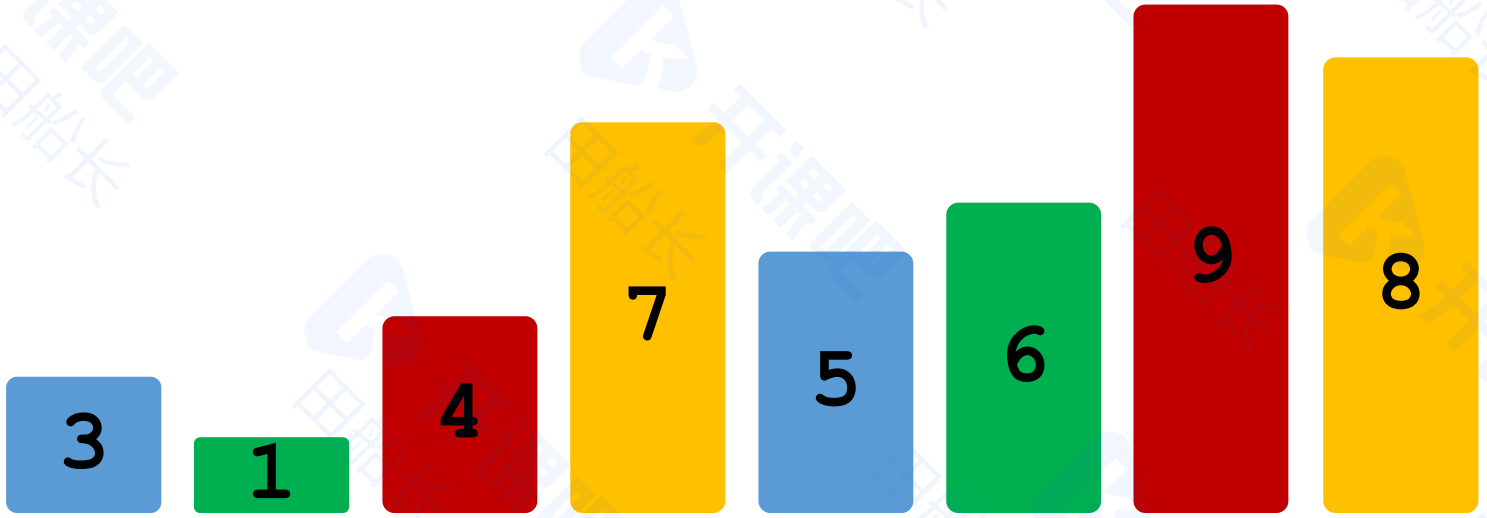
对每组分别进行插入排序

第一趟排序



对每组分别进行插入排序

第一趟排序



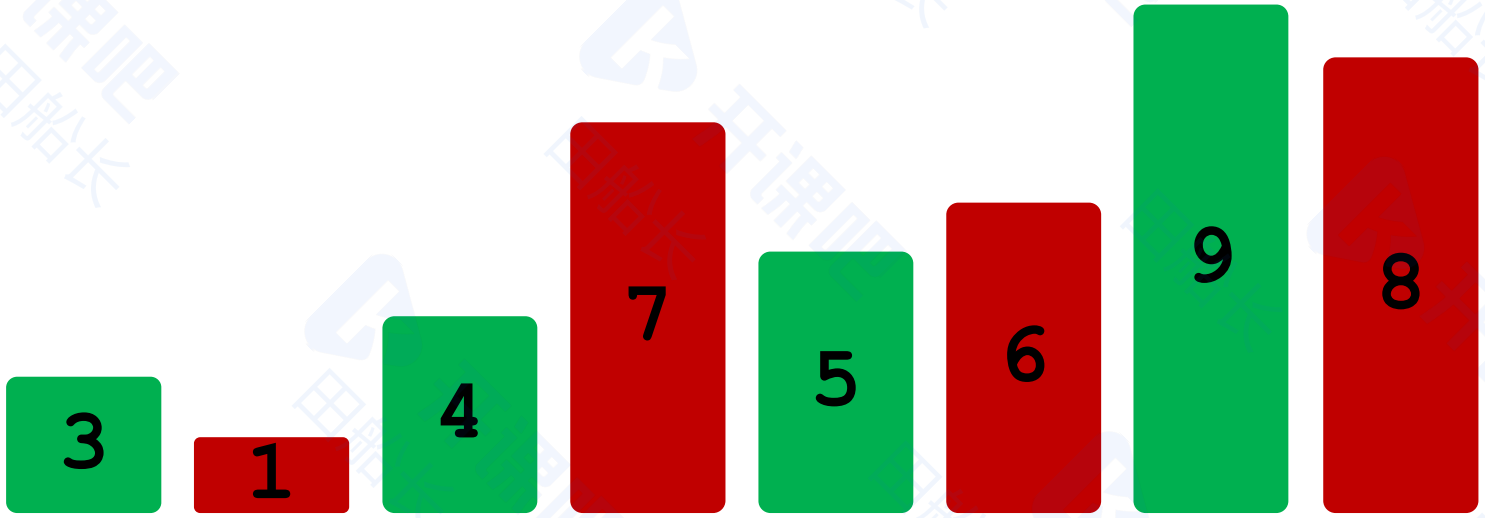
第一趟排序结束

第二趟排序

确定第二组增量

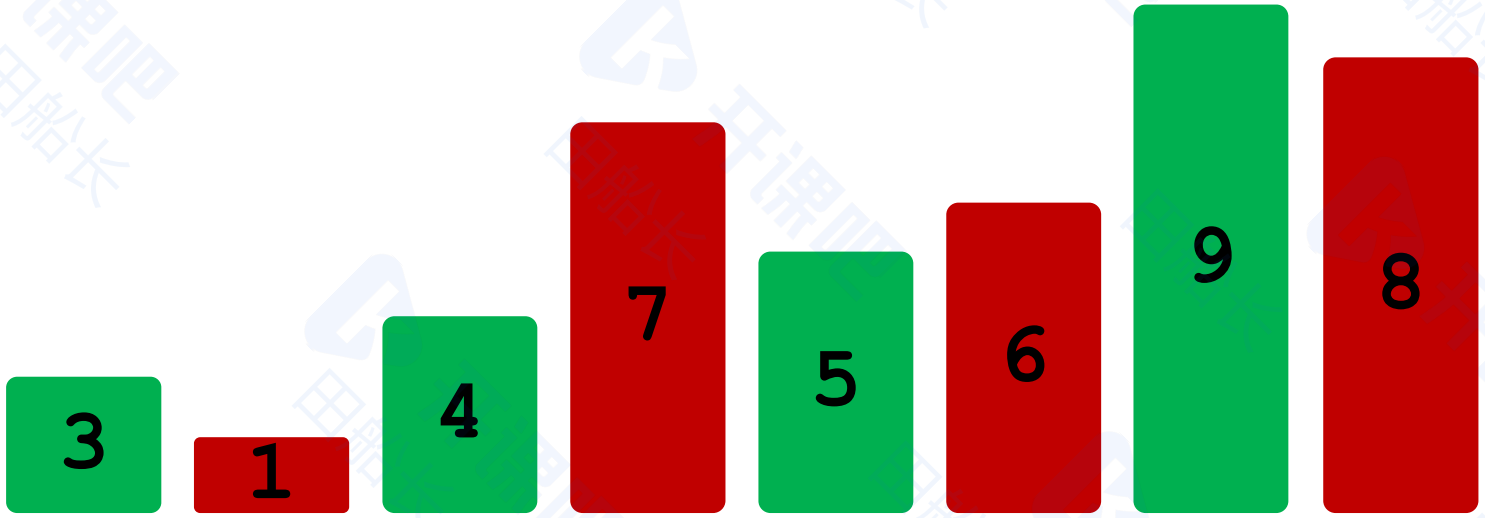
D2 = D1 / 2 = 2

以间隔D2进行分组



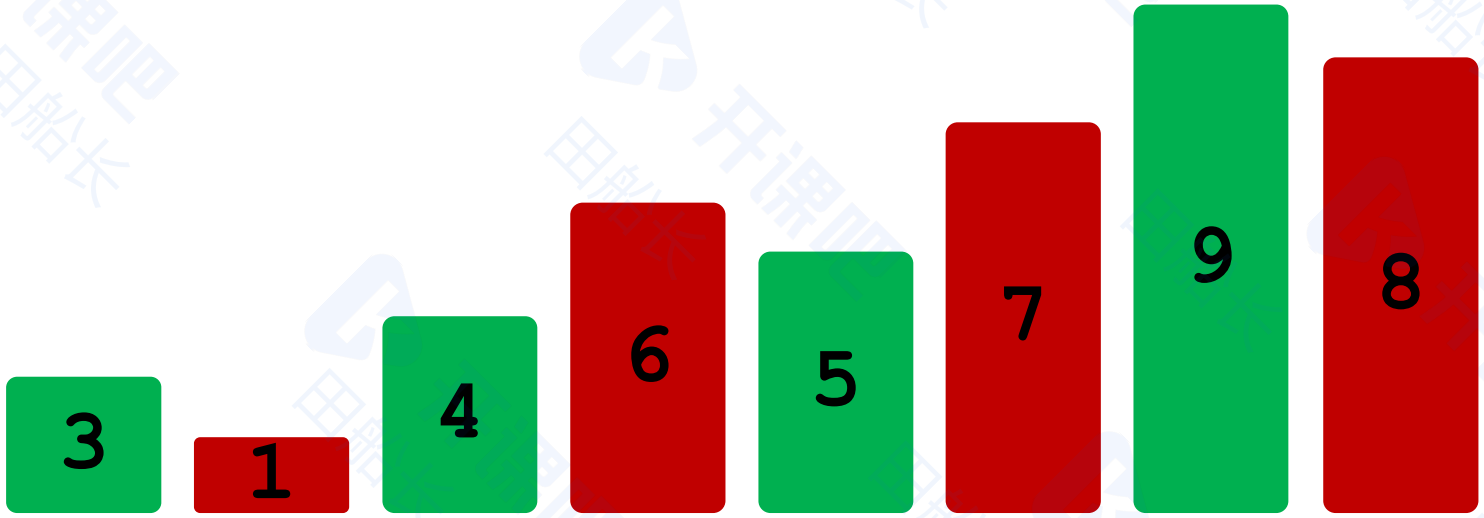
对每组分别进行插入排序

第二趟排序



对每组分别进行插入排序

第二趟排序



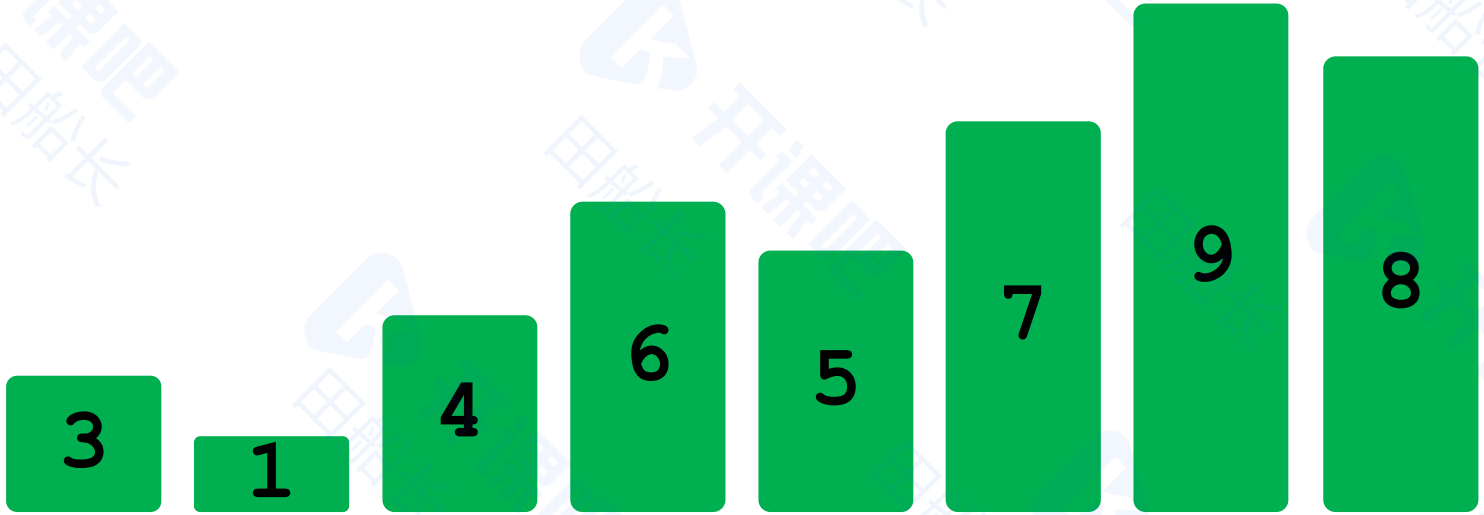
第二趟排序结束

第三趟排序

确定第三组增量

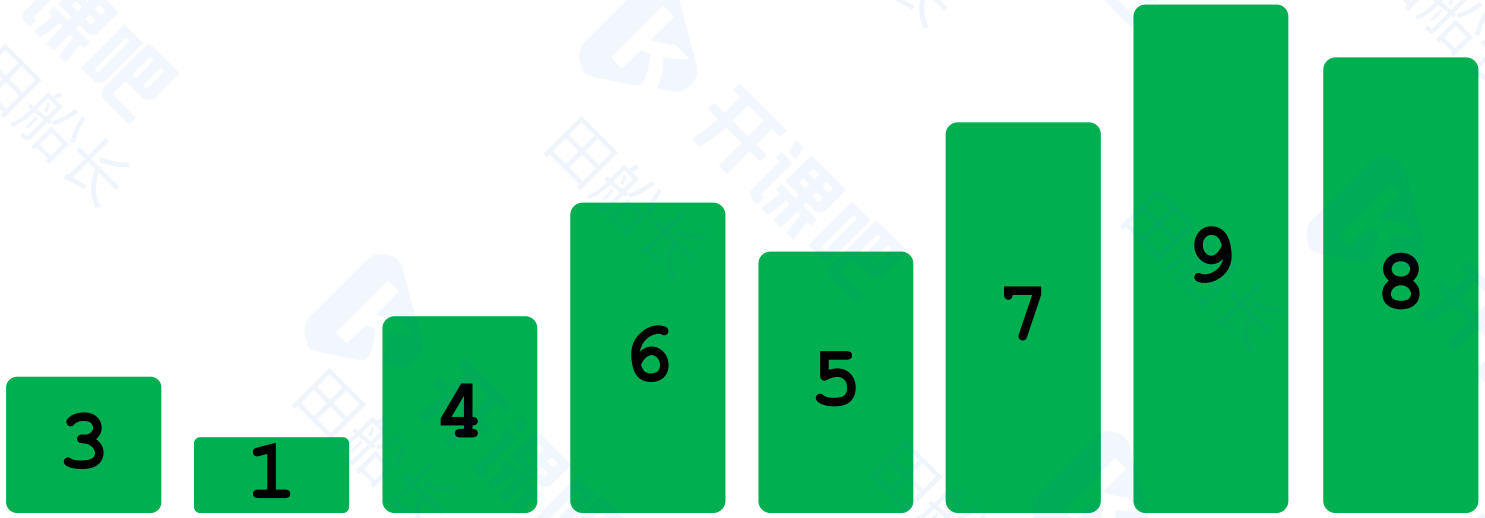
D3 = D2 / 2 = 1

以间隔D3进行分组



对每组分别进行插入排序

第三趟排序



对每组分别进行插入排序

第三趟排序

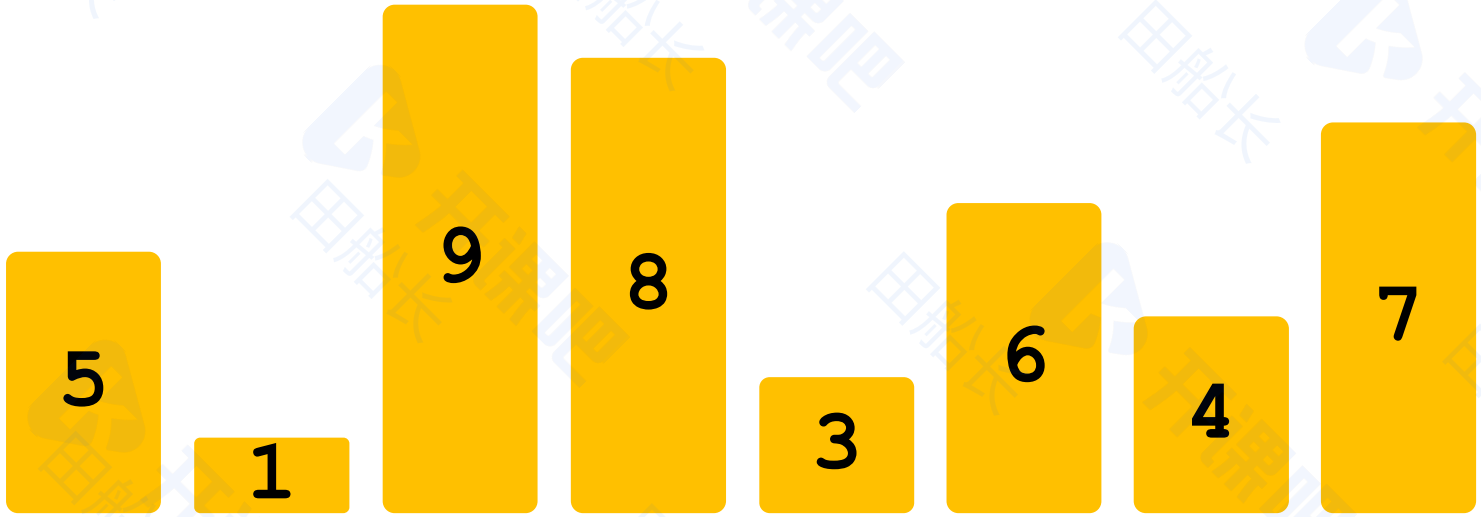
第三趟排序结束

希尔排序结束

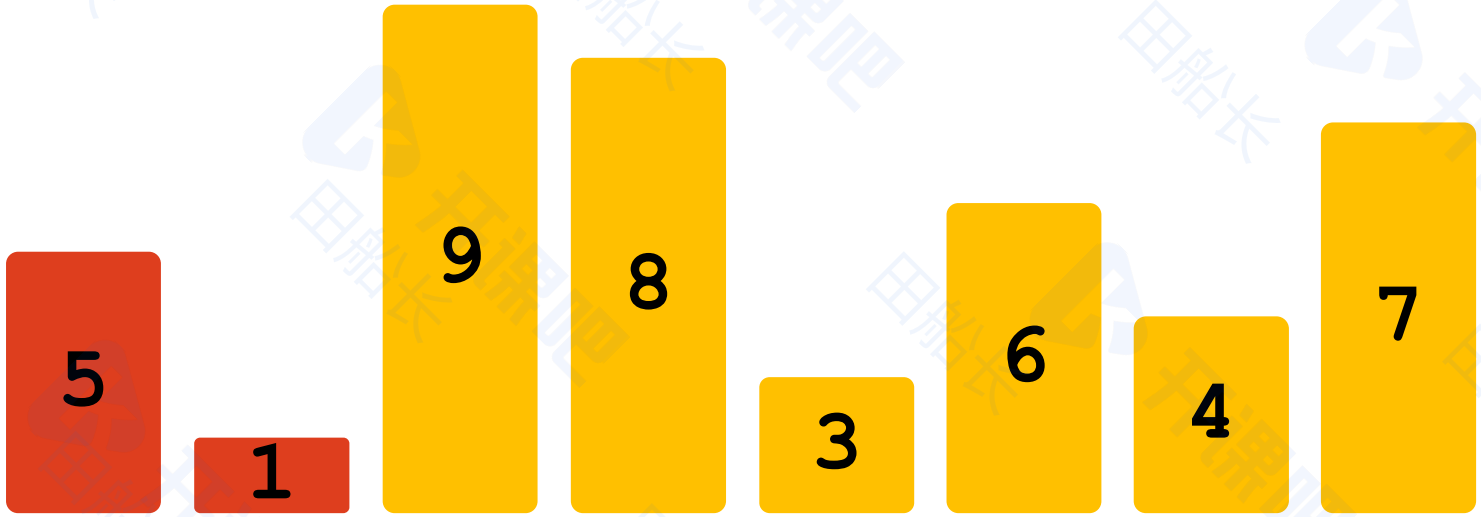
交换类排序——冒泡排序（起泡排序、泡排序）

两两比较相邻的元素
如果不满足大小关系，就交换它们
遍历一轮序列后，最值元素就会被交换到最后
重复上述过程，直到所有元素均有序

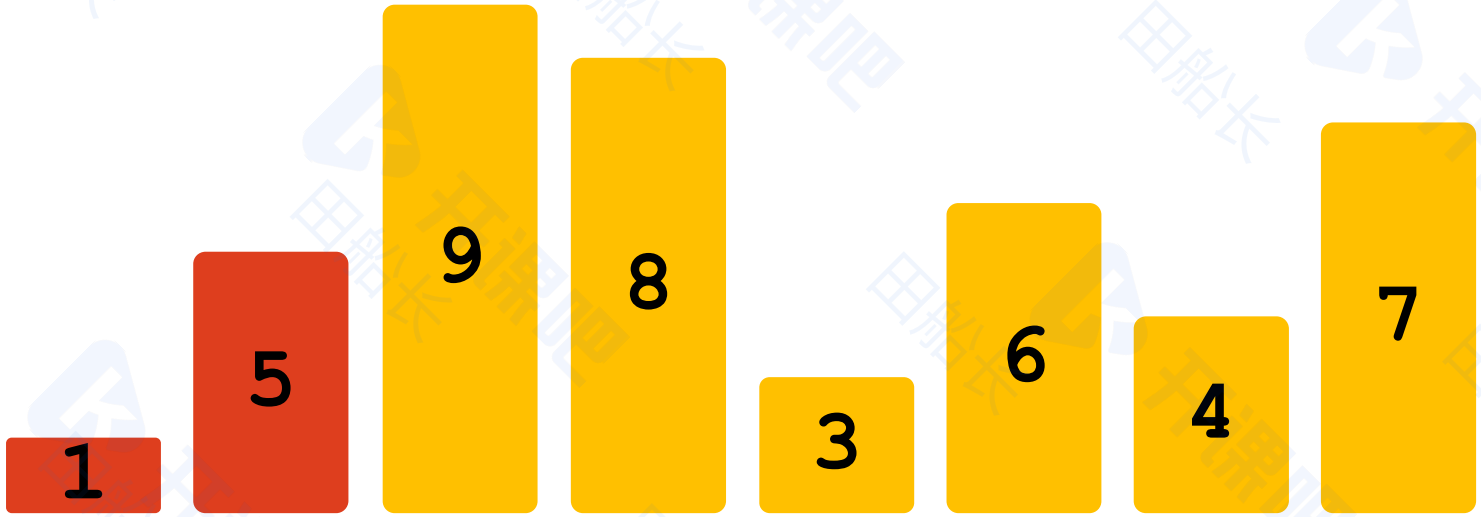
若某一轮遍历没有触发任何交换
则此时序列已经有序，可以提前结束排序



检查是否满足
交换条件即
5 > 1

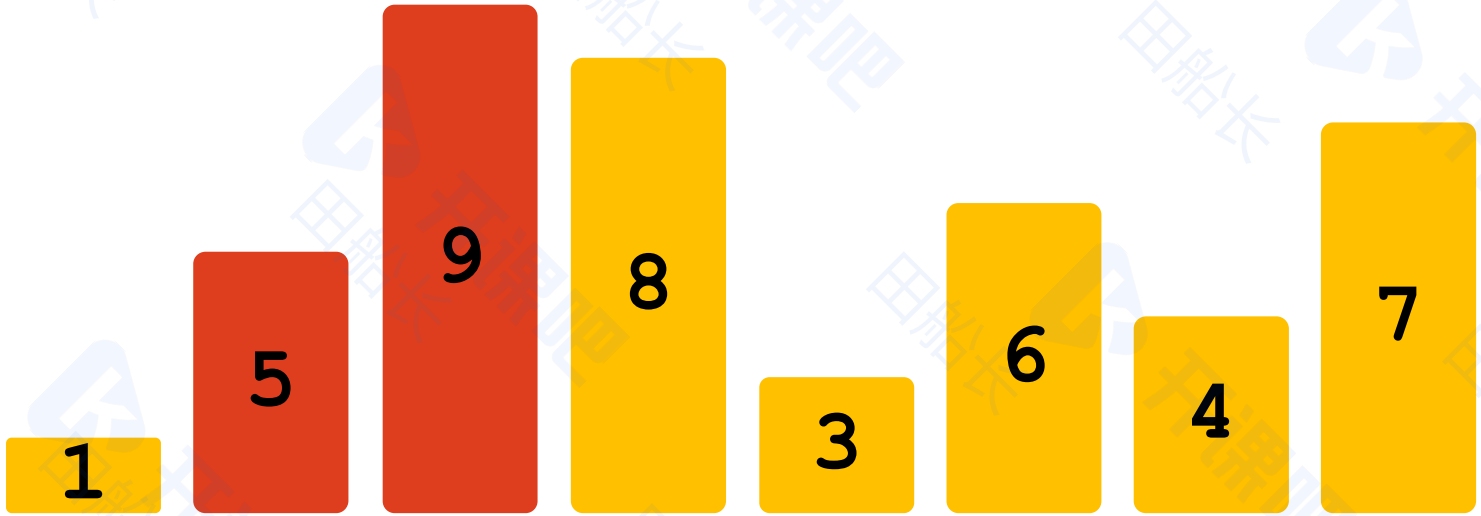


满足交换条件，
交换 5 和 1

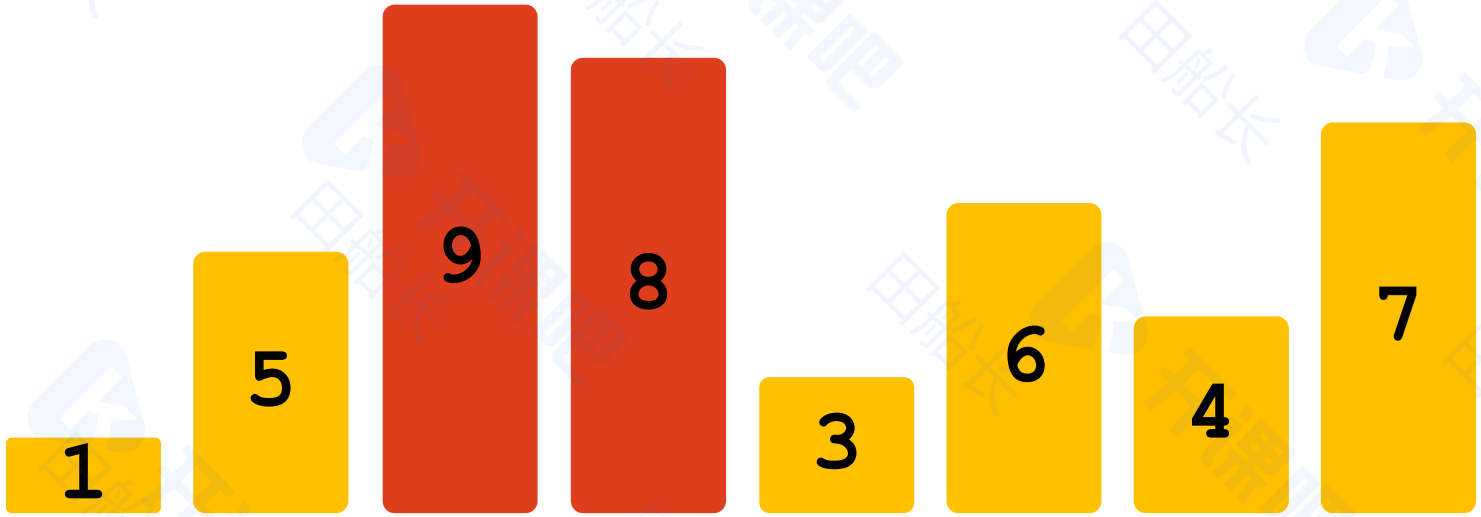


开课吧

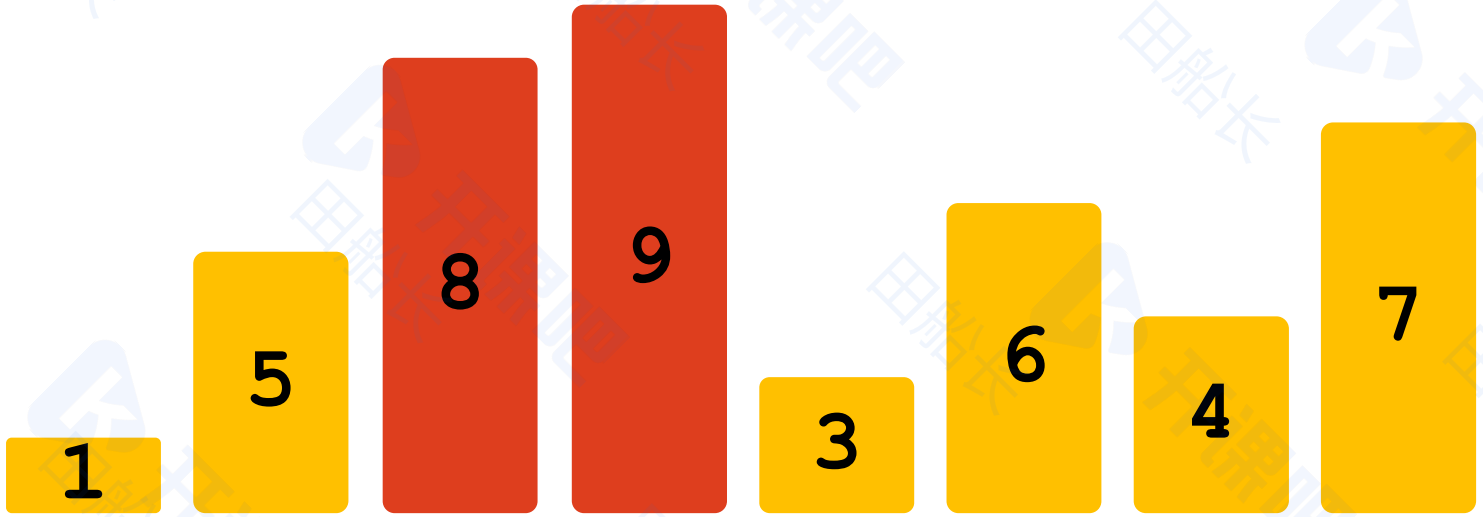
检查是否满足
交换条件即
5 > 9



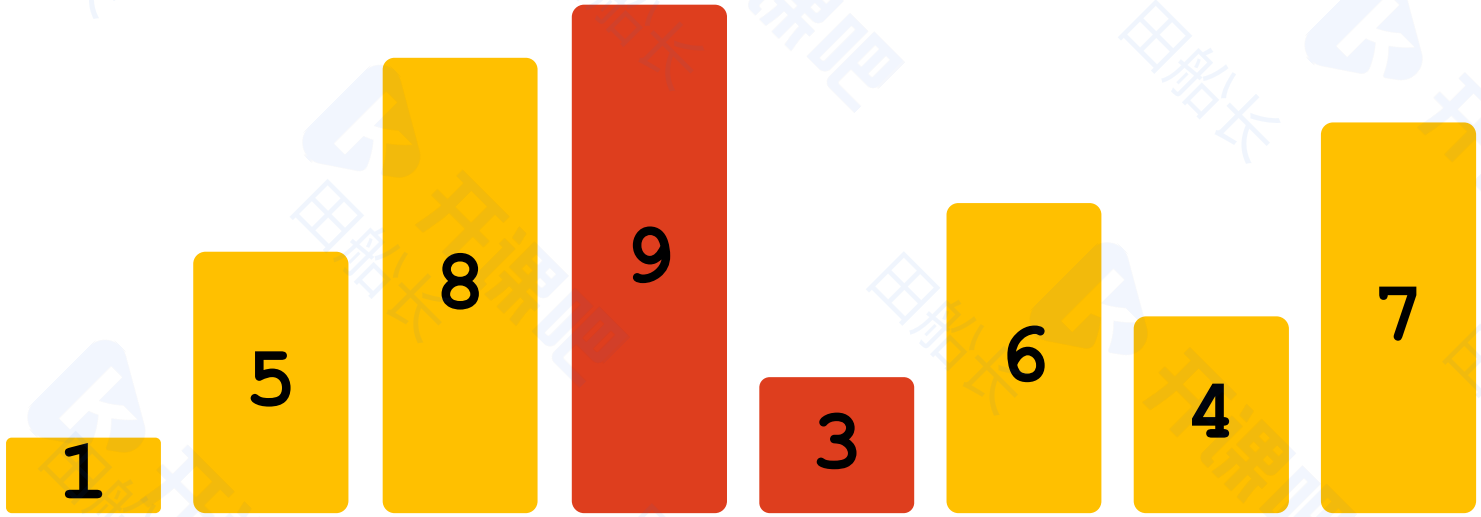
检查是否满足
交换条件即
9 > 8



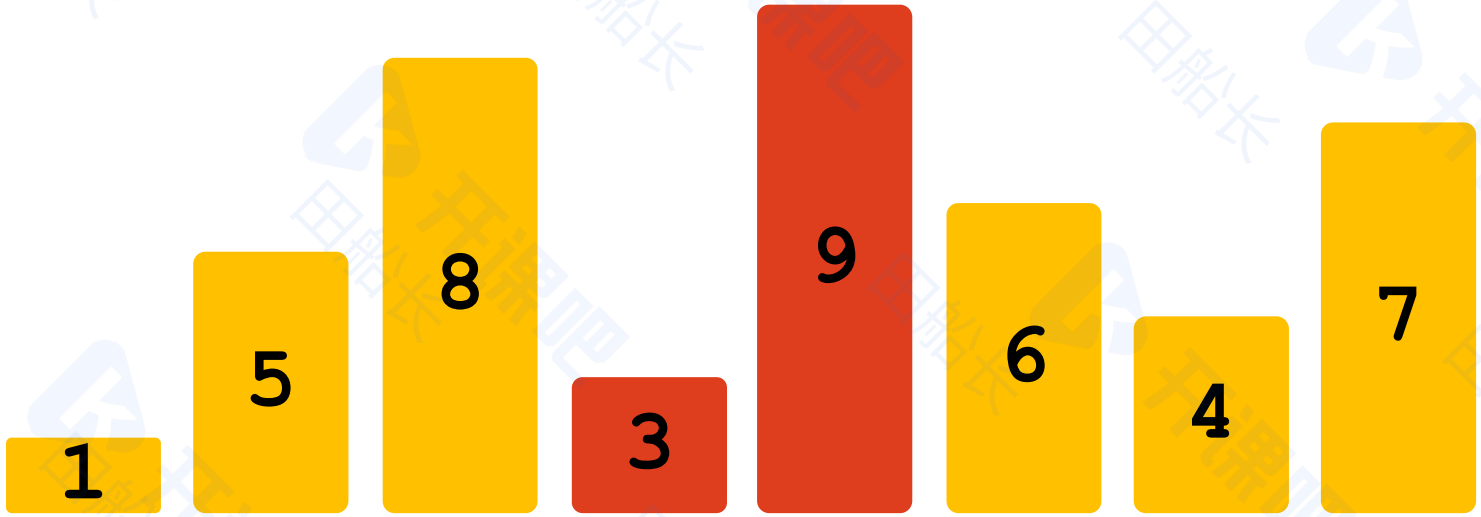
满足交换条件，
交换 9 和 8



检查是否满足
交换条件即
9 > 3

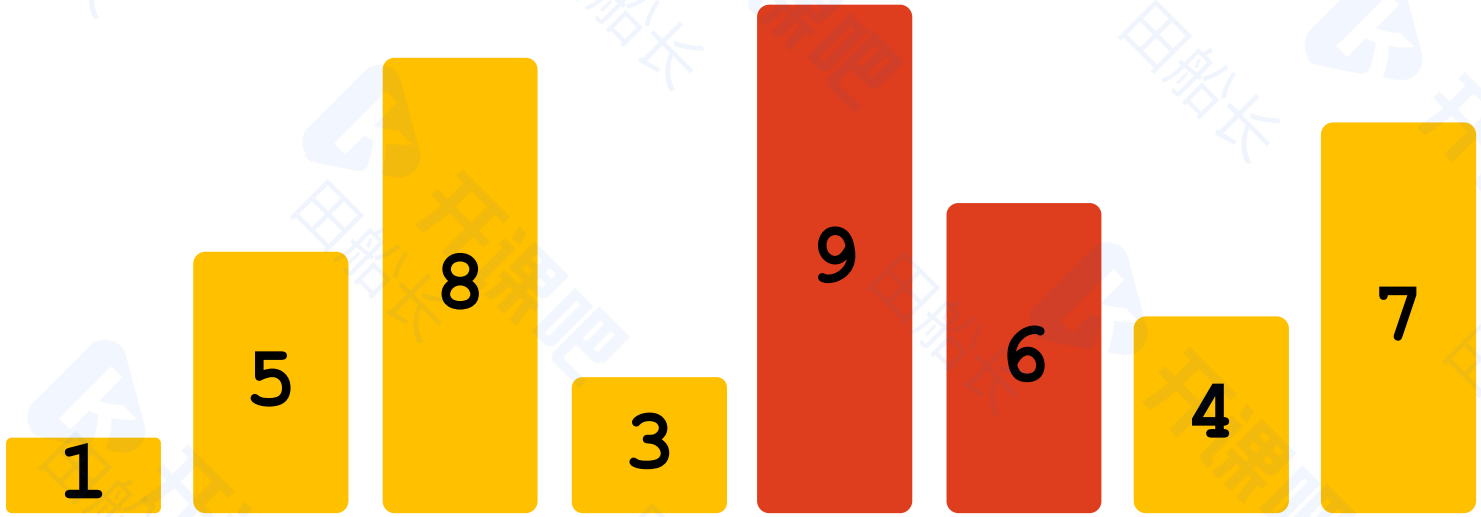


满足交换条件，
交换 9 和 3

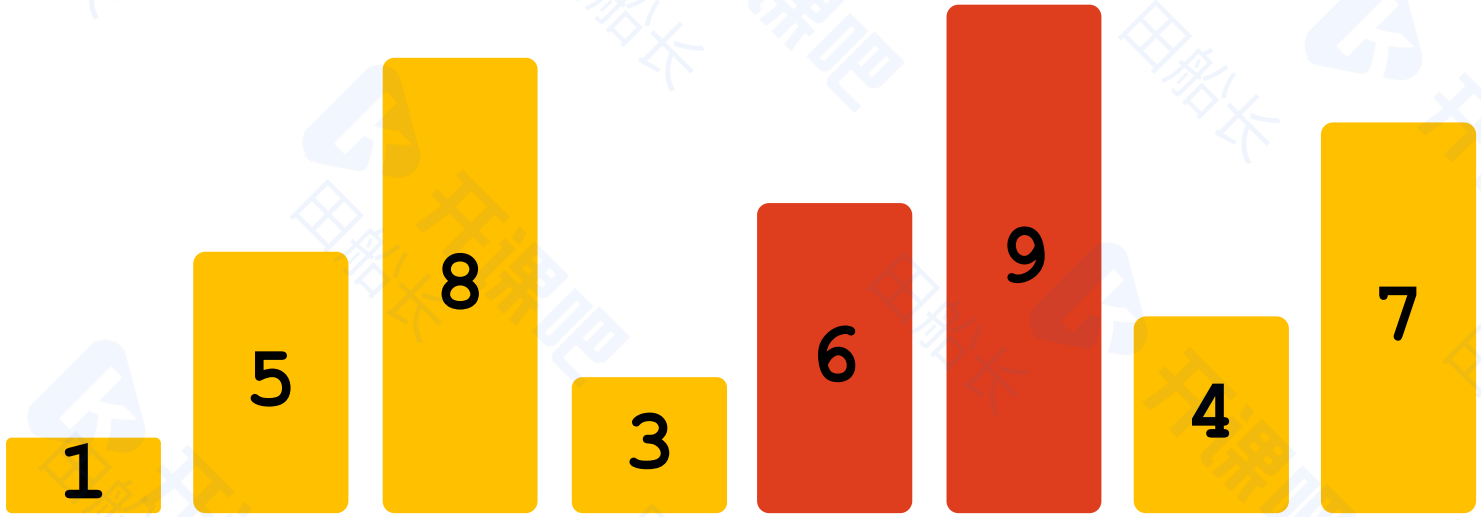


检查是否满足交换条件即

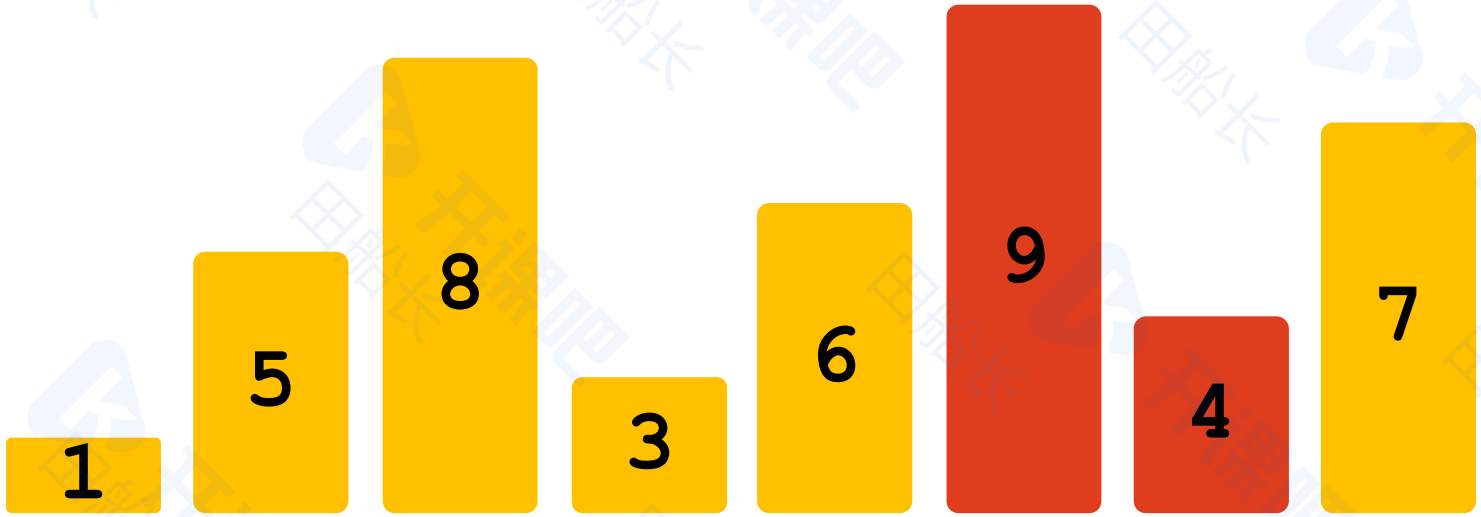
9 > 6



满足交换条件，
交换 9 和 6



检查是否满足
交换条件即
9 > 4



满足交换条件，
交换 9 和 4

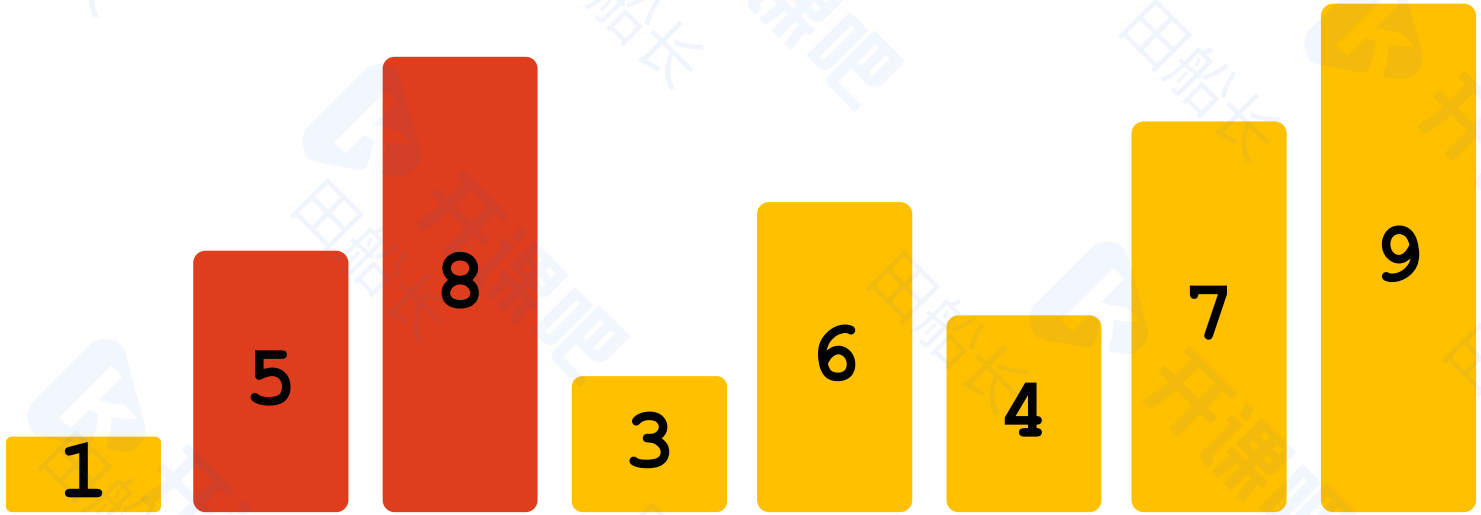
检查是否满足交换条件即

9 > 7

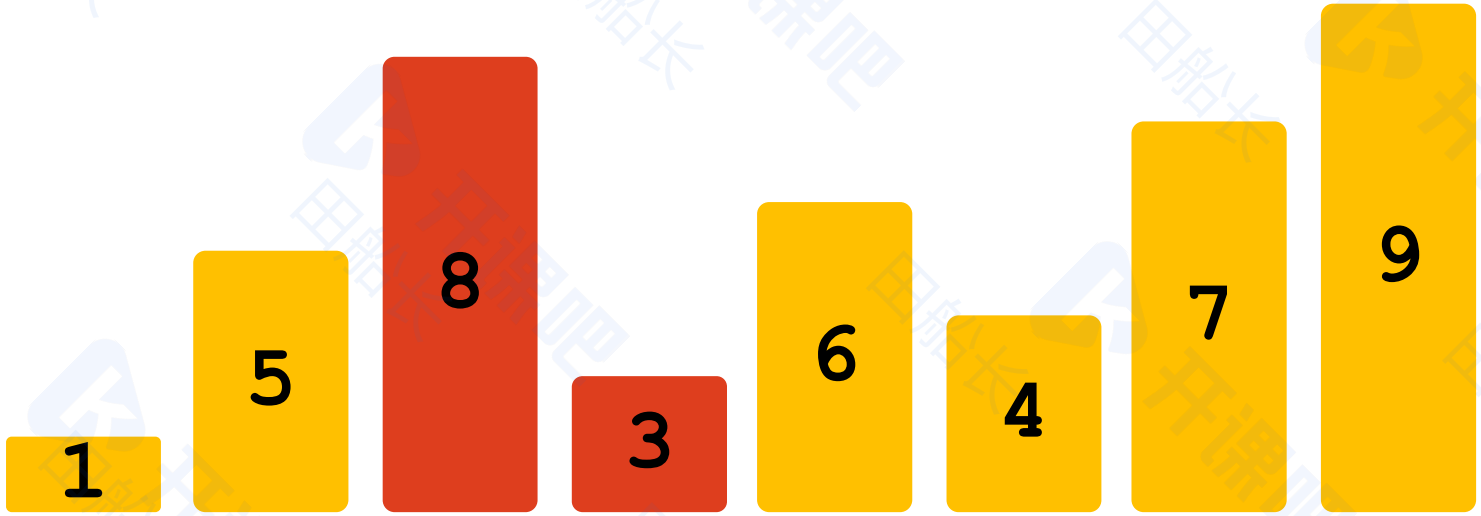
满足交换条件，
交换 9 和 7

检查是否满足
交换条件即
1 > 5

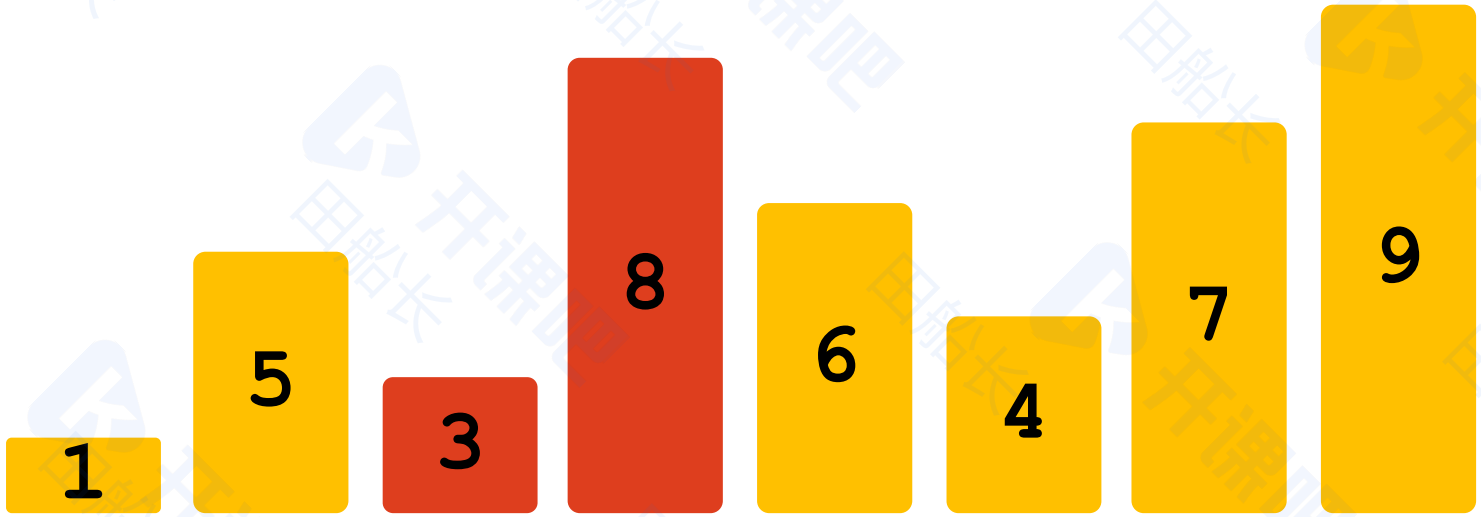
检查是否满足
交换条件即
5 > 8



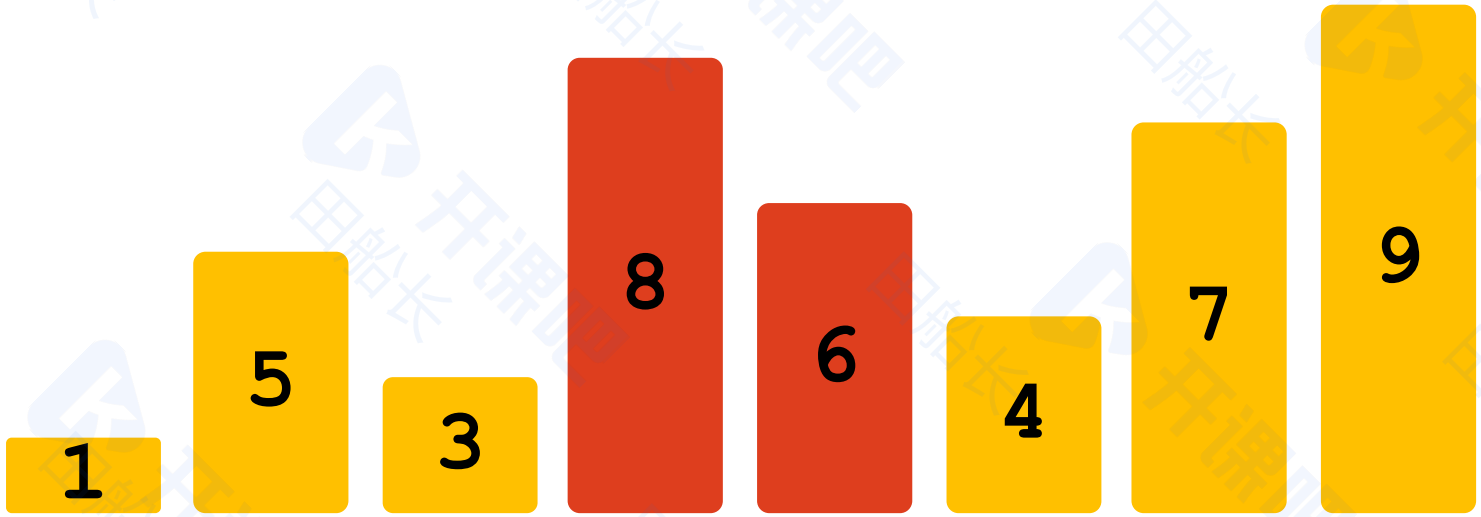
检查是否满足
交换条件即
8 > 3



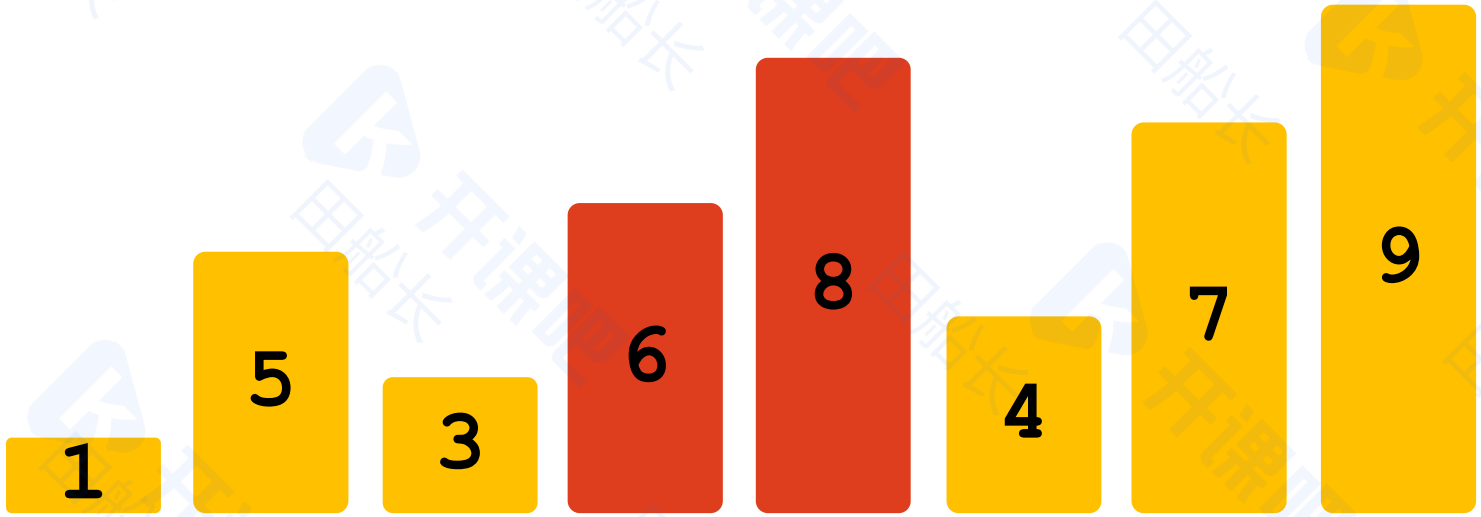
满足交换条件，
交换 8 和 3



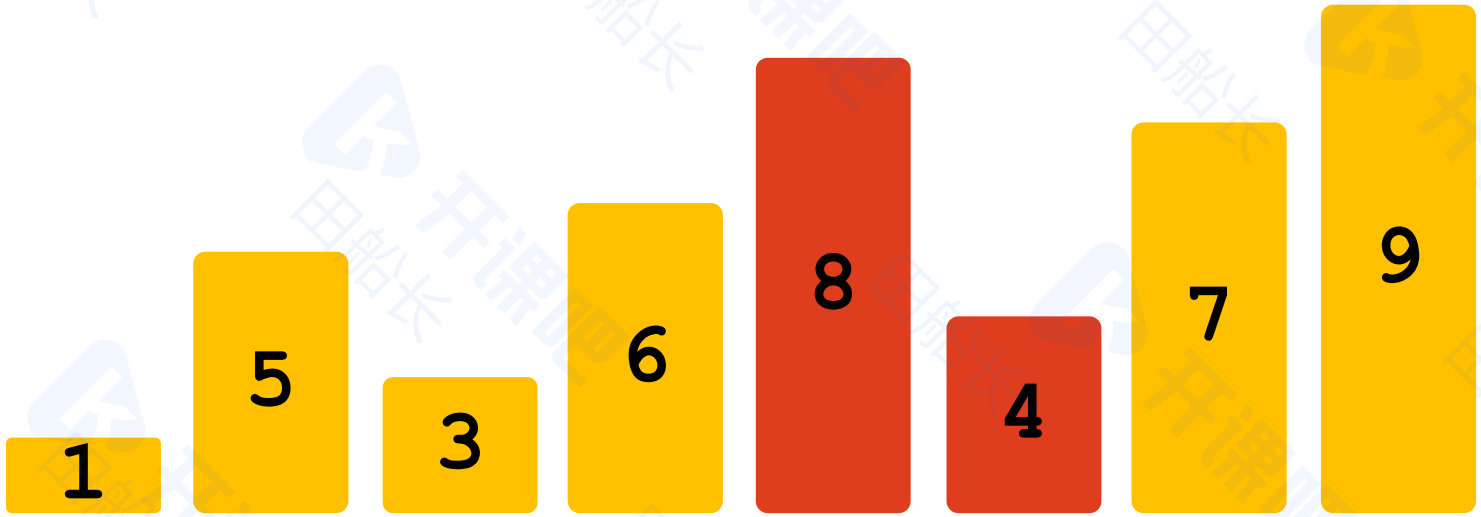
检查是否满足
交换条件即
8 > 6



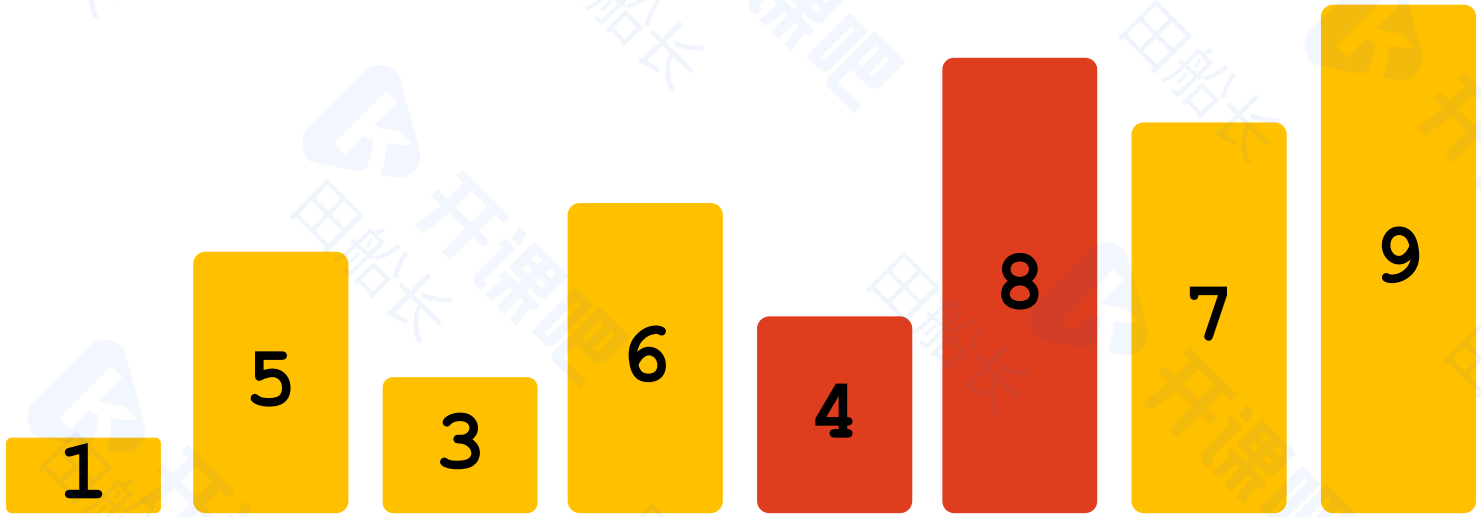
满足交换条件，
交换 8 和 6



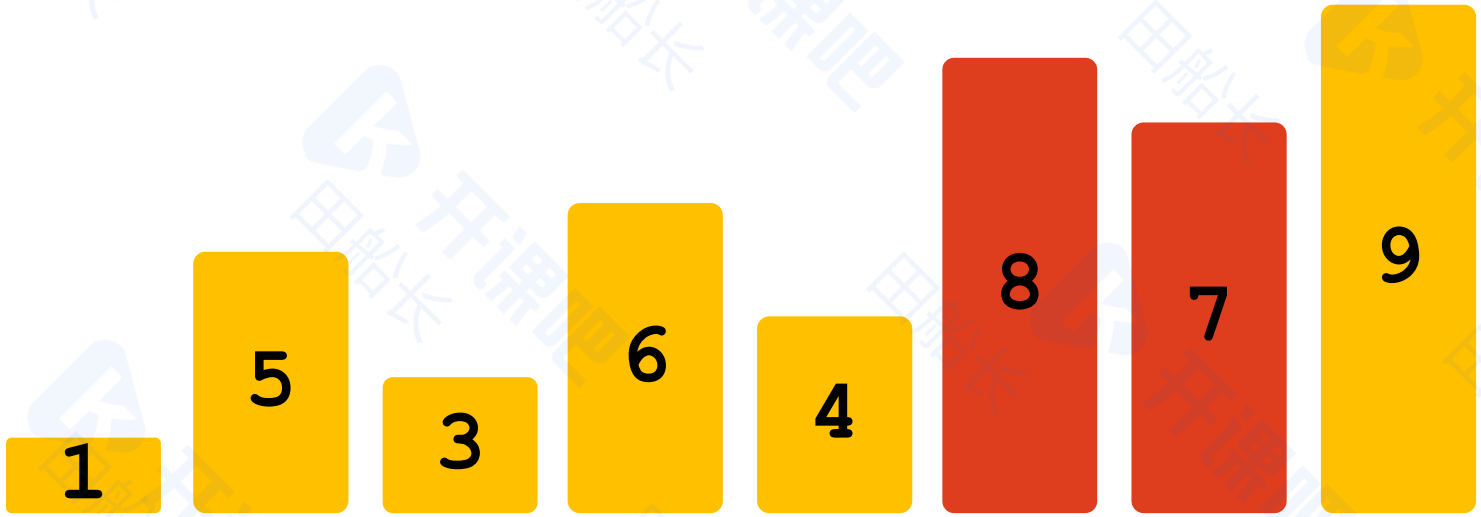
检查是否满足
交换条件即
8 > 4



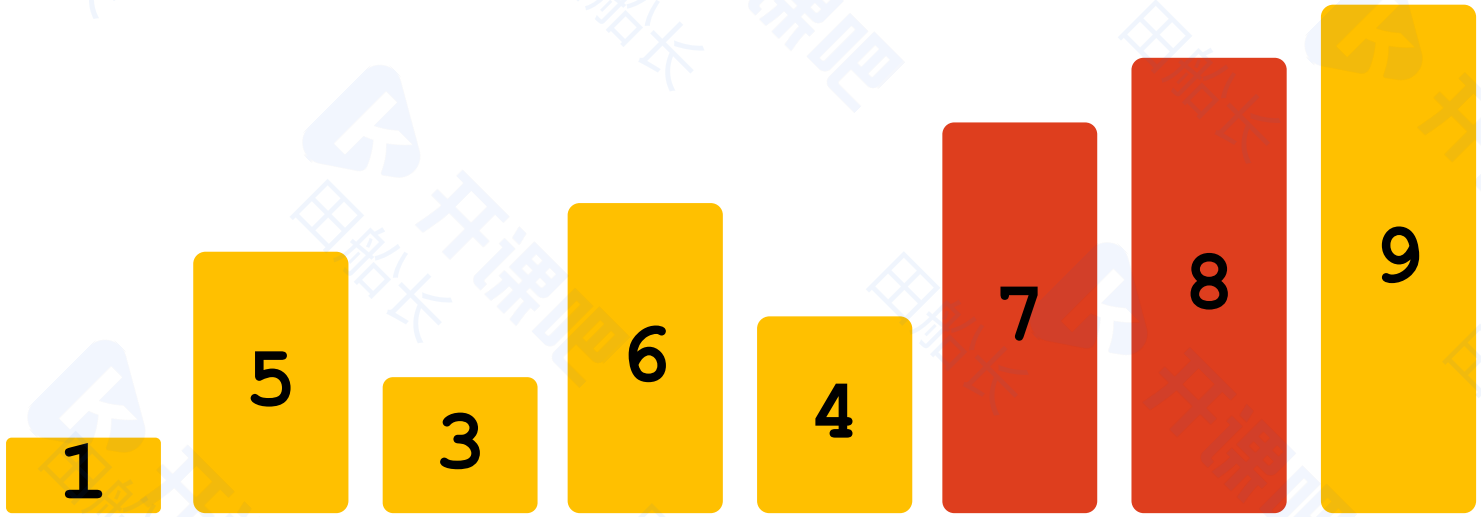
满足交换条件，
交换 8 和 4



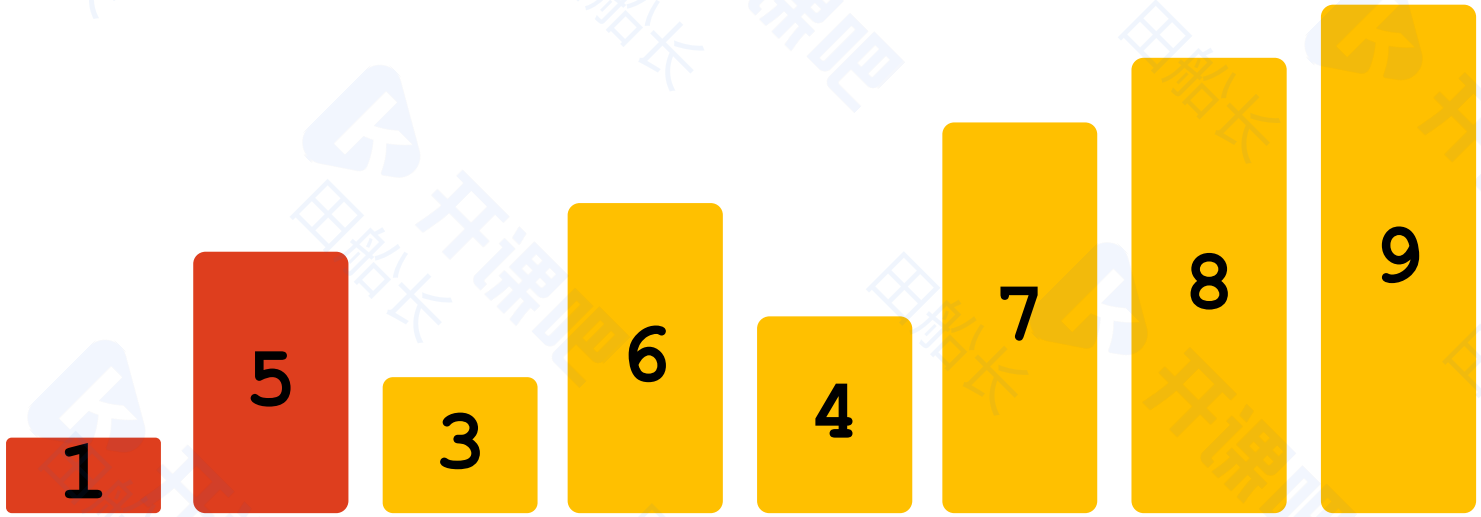
检查是否满足
交换条件即
8 > 7



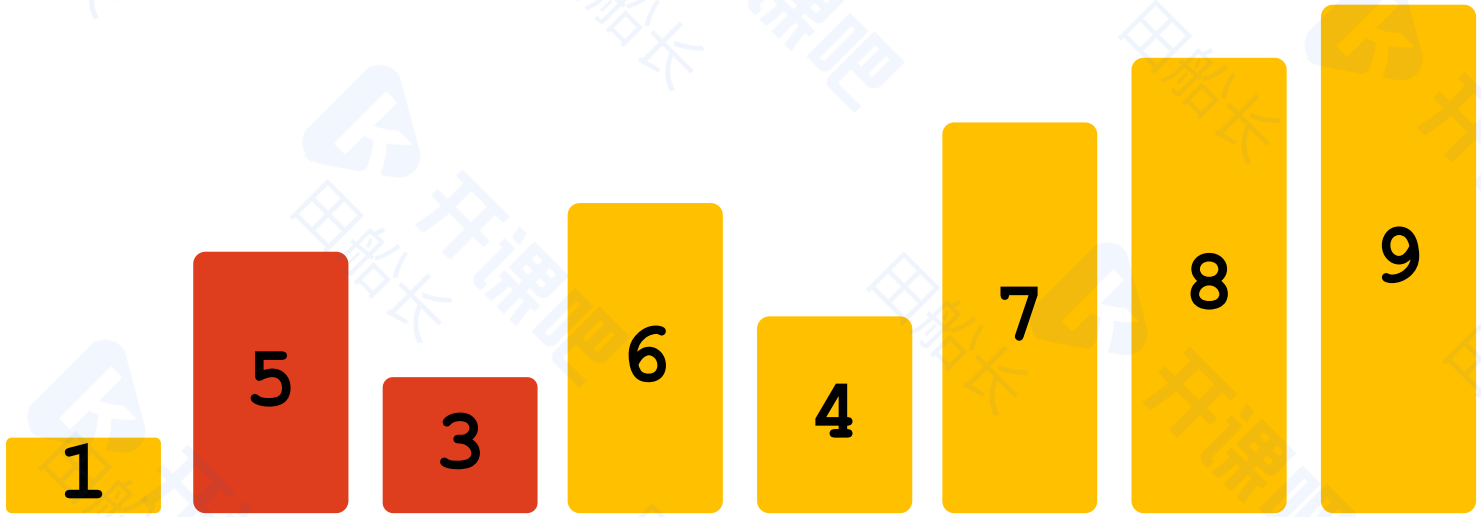
满足交换条件，
交换 8 和 7



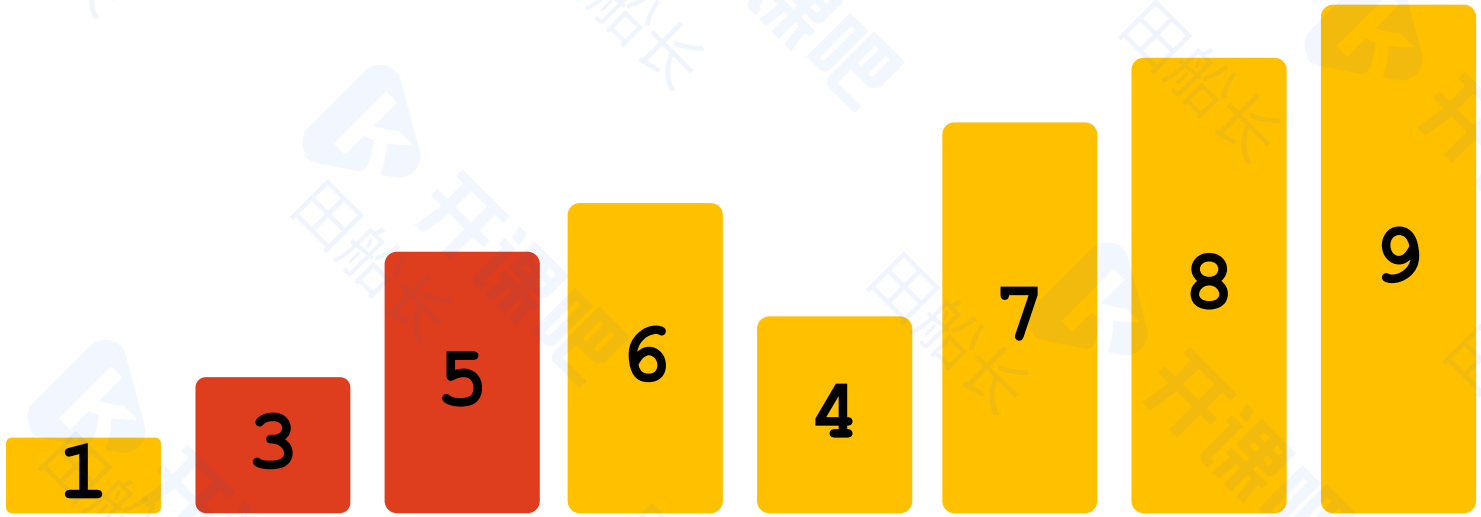
检查是否满足
交换条件即
1 > 5



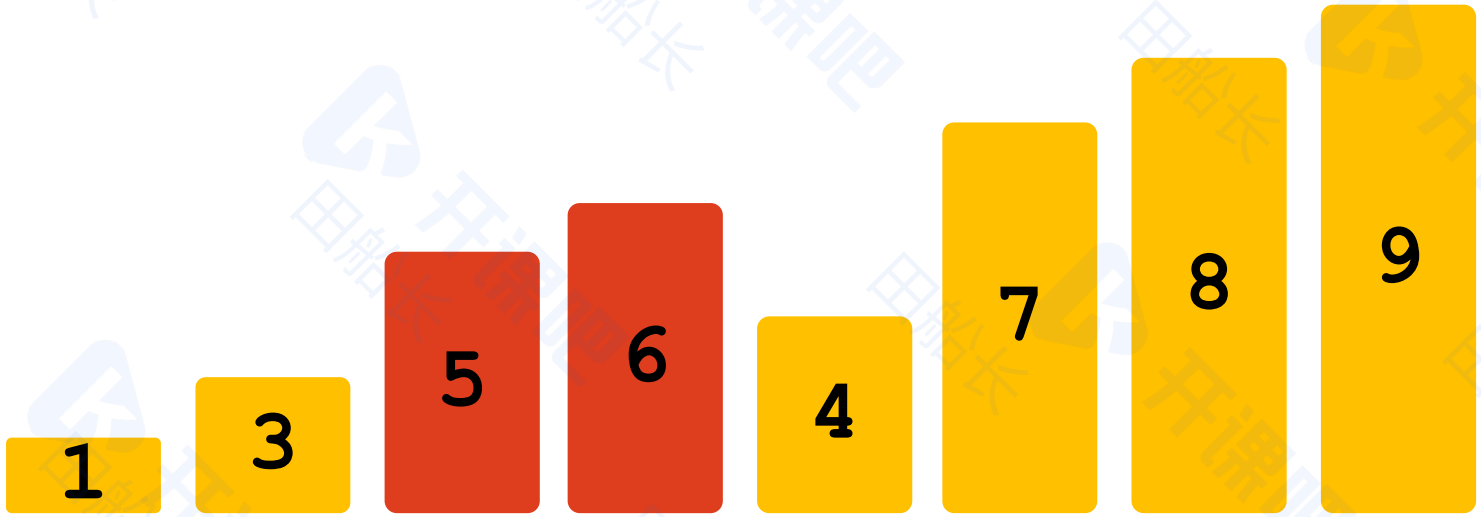
检查是否满足
交换条件即
5 > 3



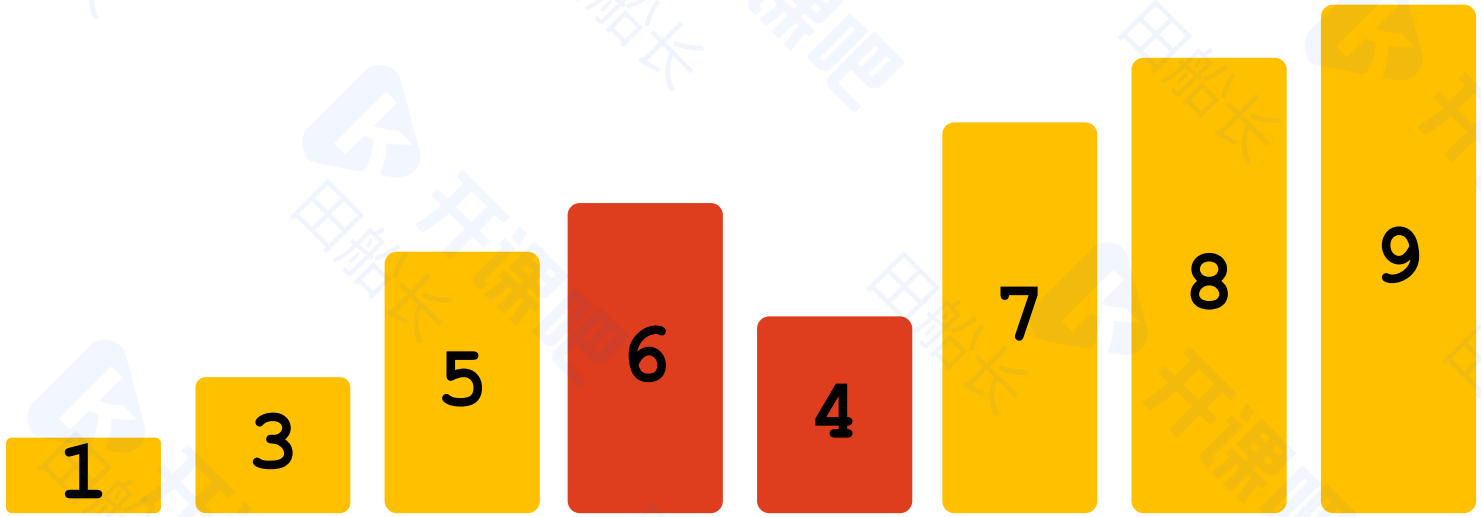
满足交换条件，
交换 5 和 3



检查是否满足
交换条件即
5 > 6

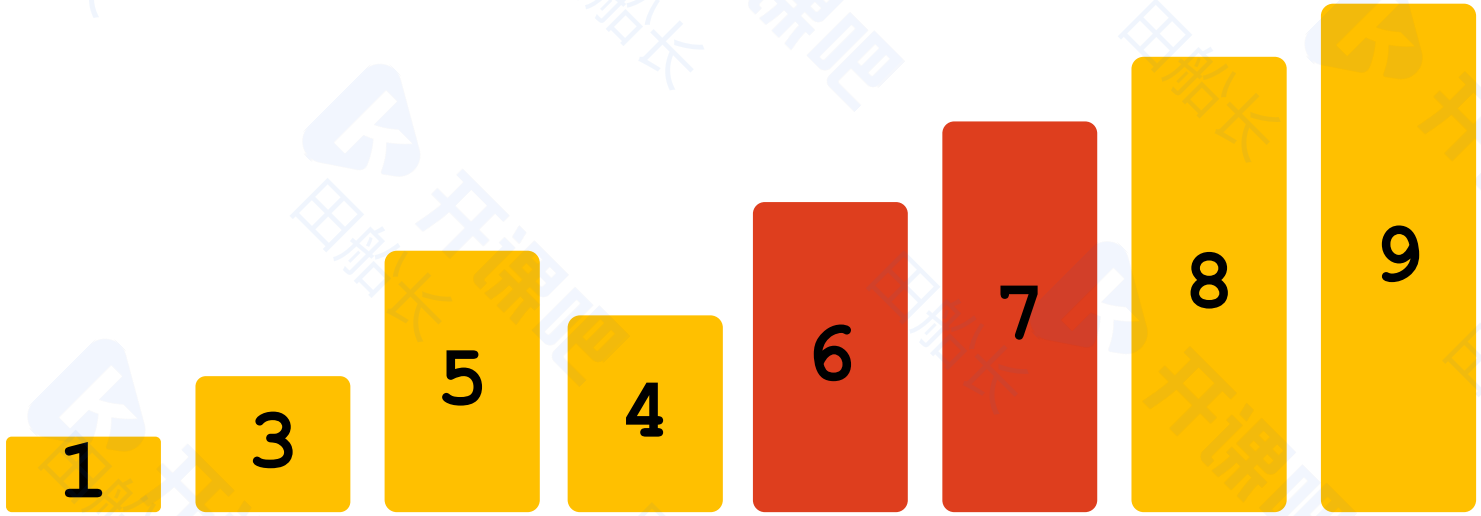


检查是否满足
交换条件即
6 > 4

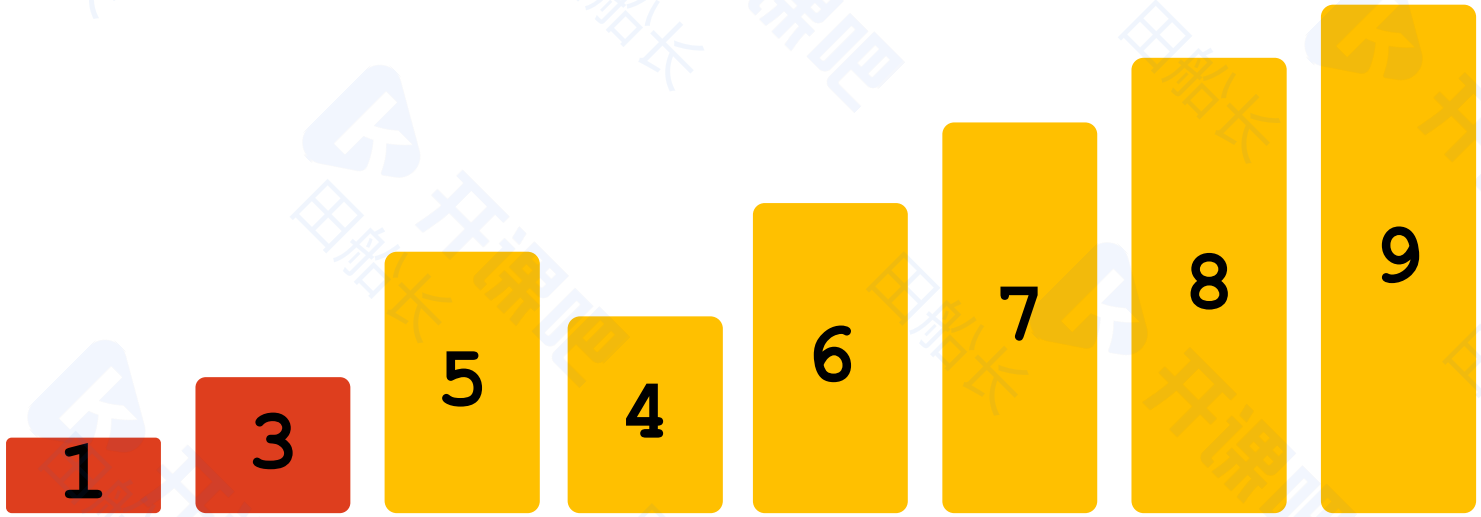


满足交换条件，
交换 6 和 4

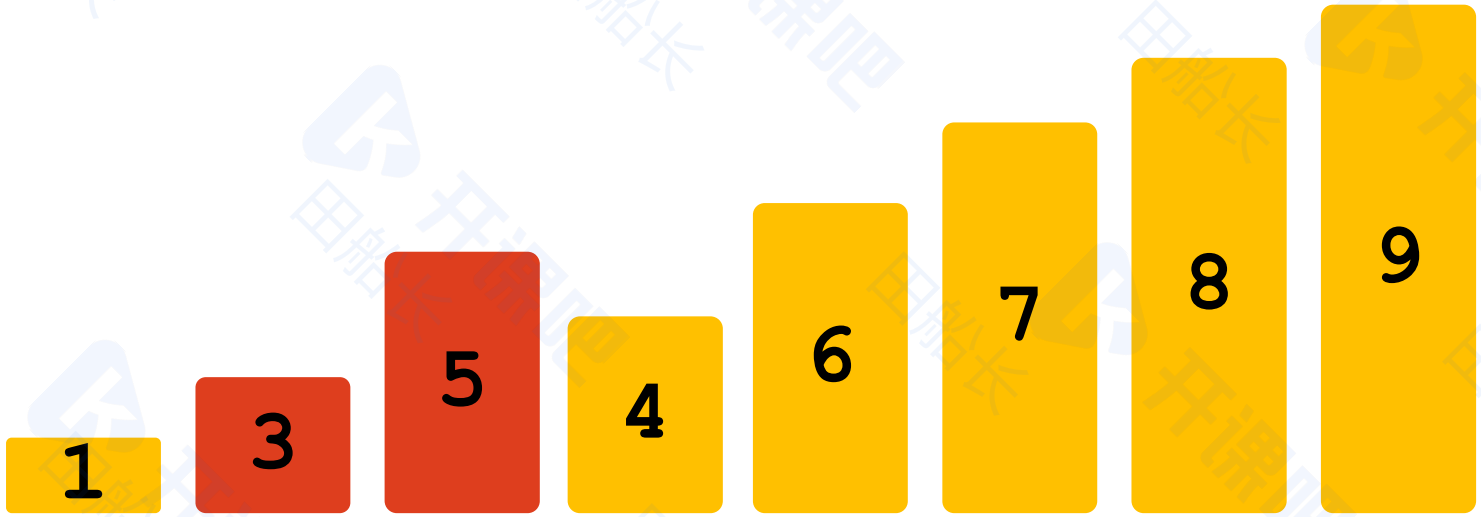
检查是否满足
交换条件即
6 > 7



检查是否满足
交换条件即
1 > 3

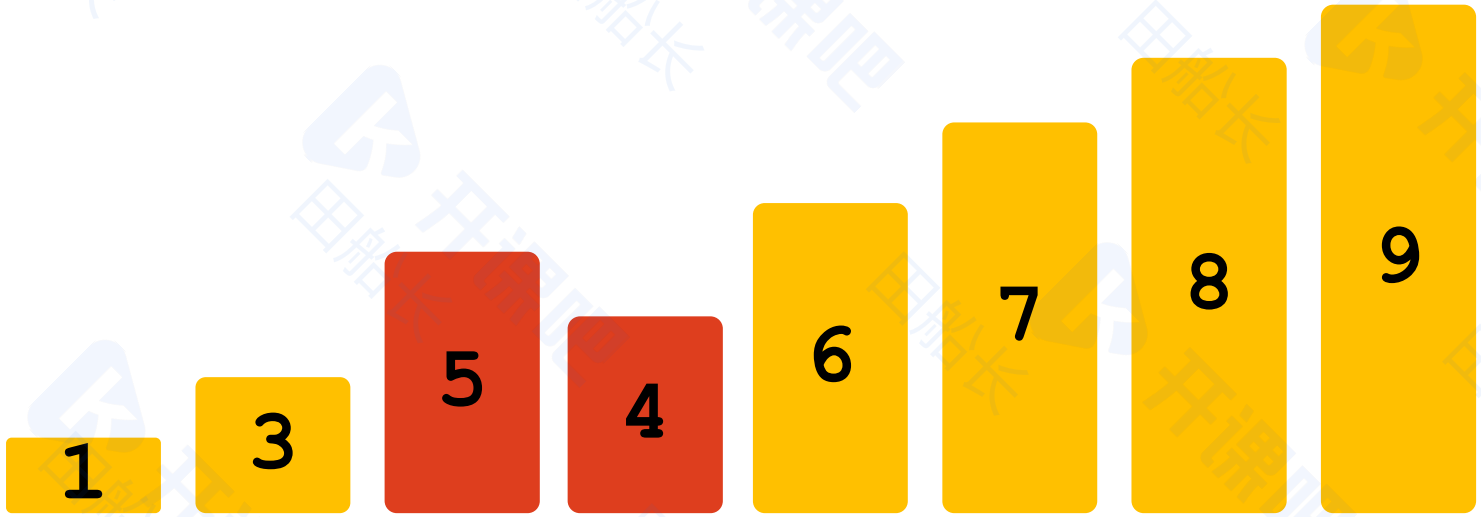


检查是否满足
交换条件即
3 > 5



检查是否满足交换条件即

5 > 4



满足交换条件，
交换 5 和 4

检查是否满足
交换条件即
5 > 6

检查是否满足
交换条件即
1 > 3

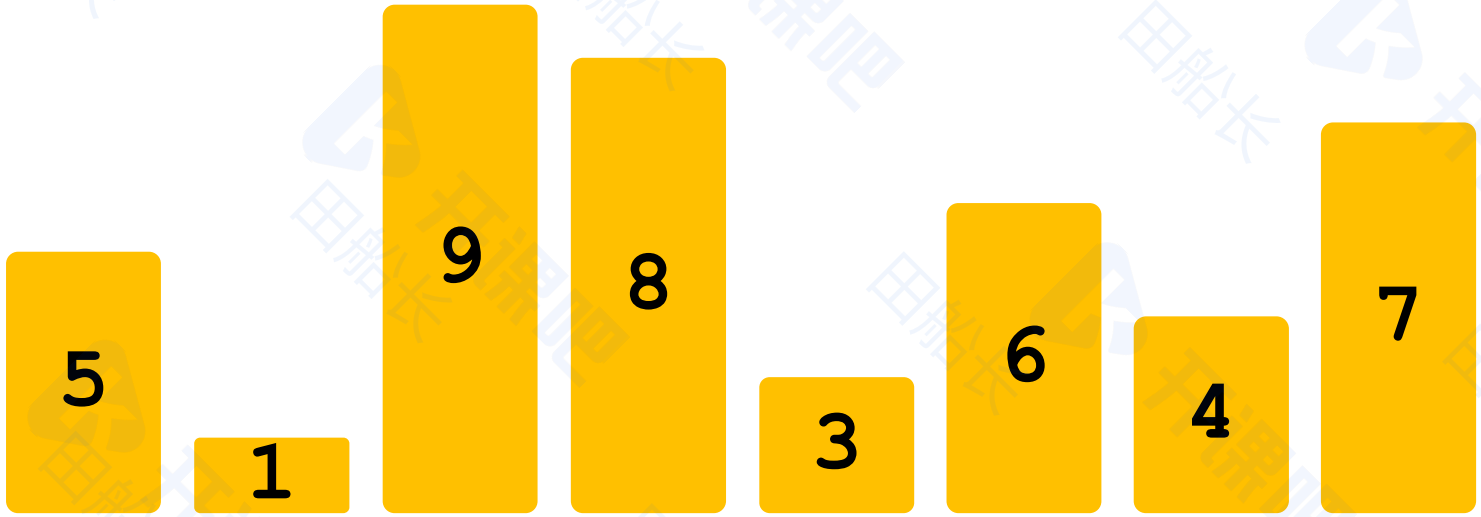
检查是否满足
交换条件即
3 > 4

检查是否满足
交换条件即
4 > 5

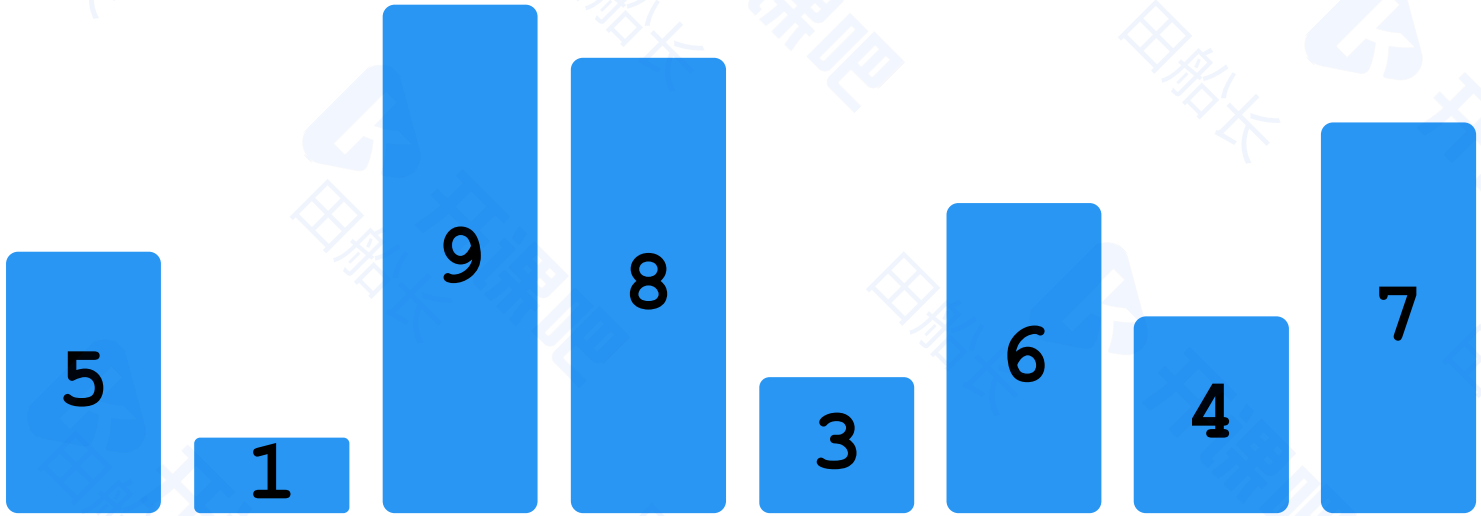
全部有序，
冒泡排序结束

交换类排序——快速排序

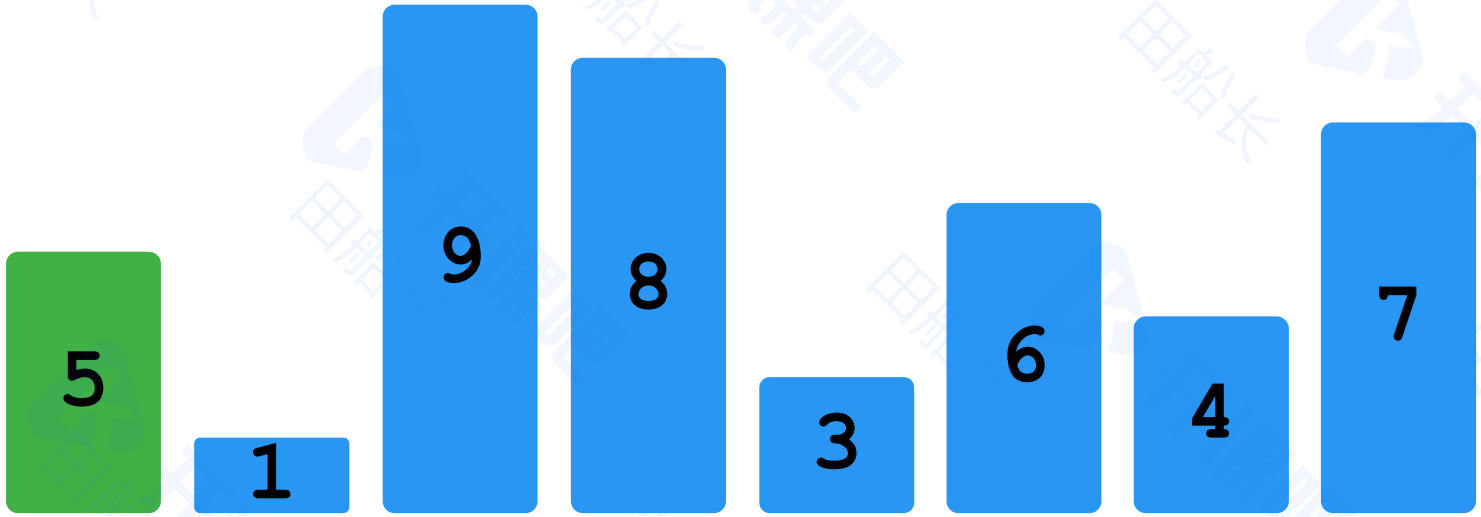
选择一个枢纽元素作为中间值
一轮排序将所有小于中间值的元素放到中间值左面
将所有大于中间值的元素放到中间值右面
接下来对左边和右边重复上述过程
直到序列整体有序



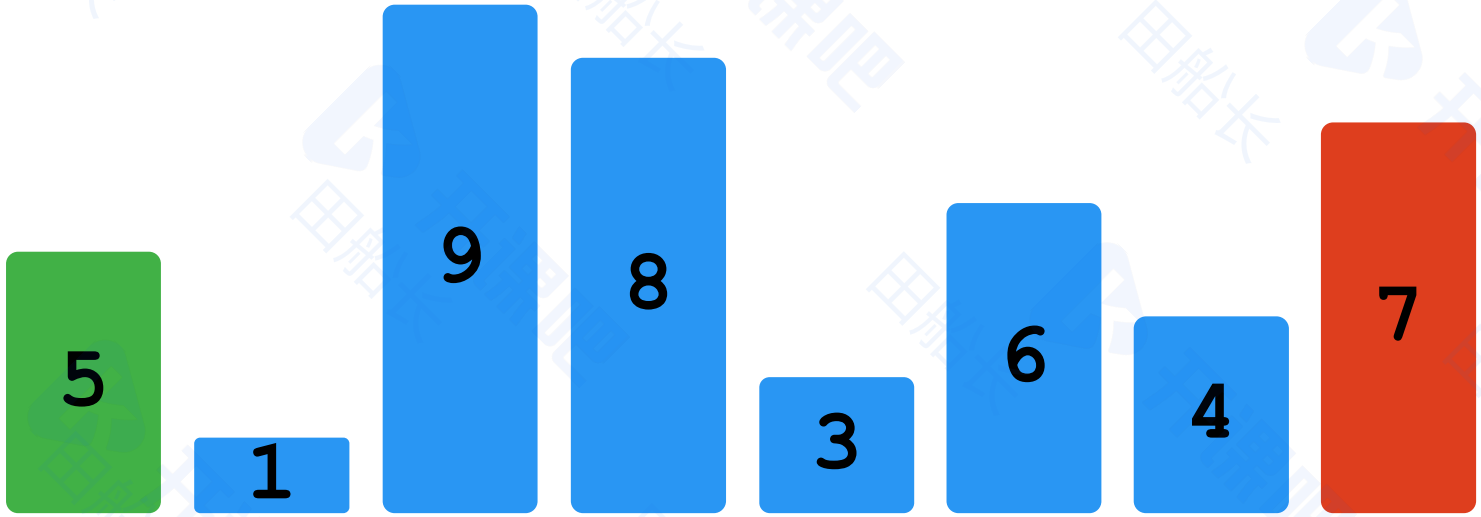
对区间[0,8)内的数进行递归的快速排序



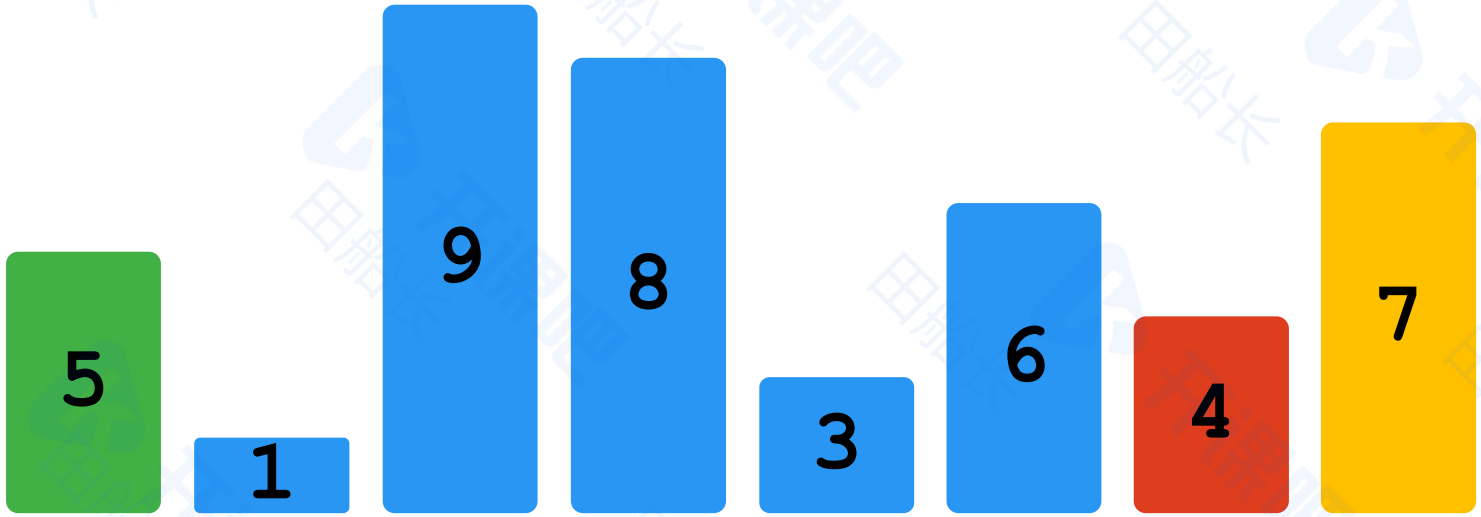
选择5作为本轮的
PIVOT



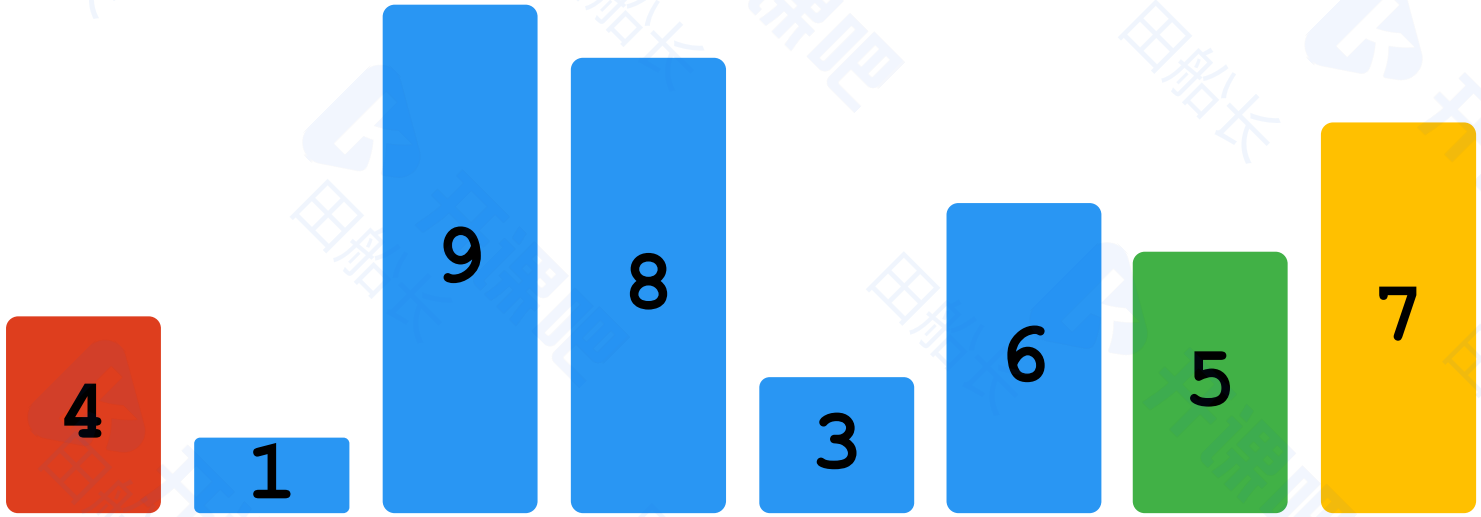
比较最右边的数7和PIVOT即5的大小



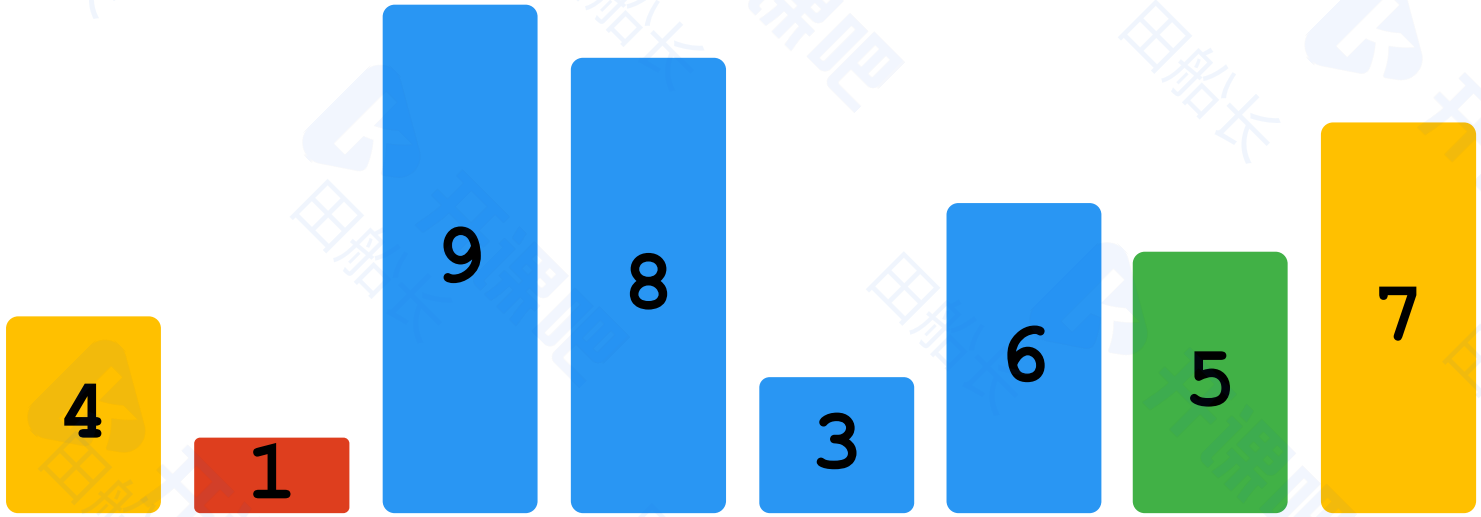
比较最右边的数4和PIVOT即5的大小



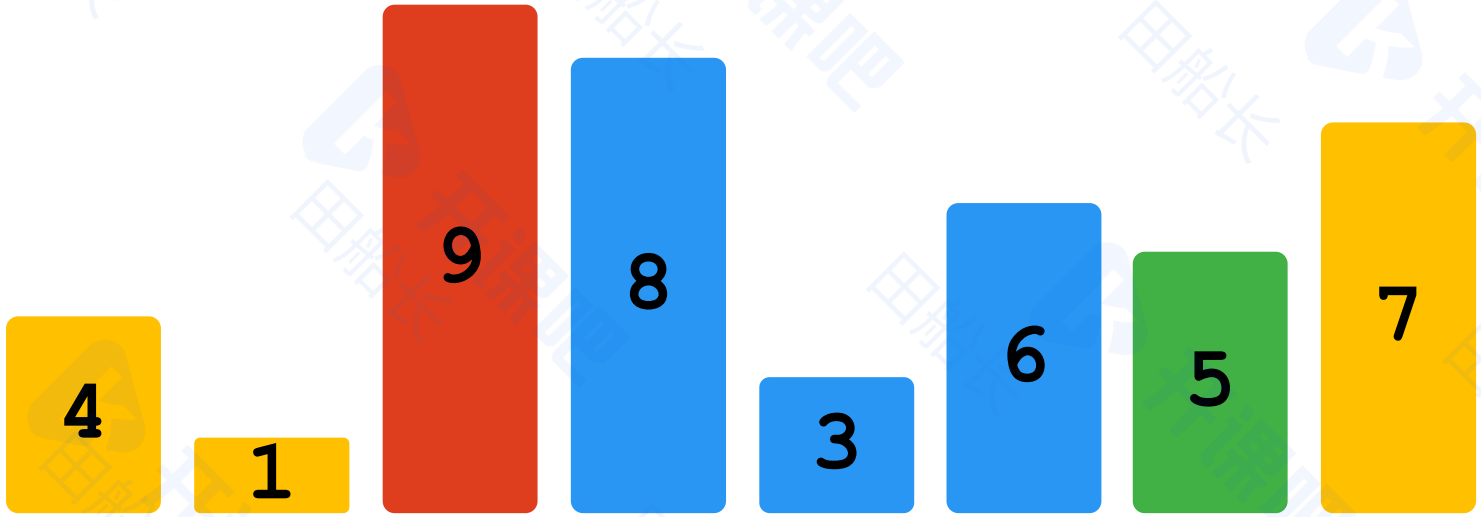
顺序不满足，交换4和PIVOT即5，换边扫描



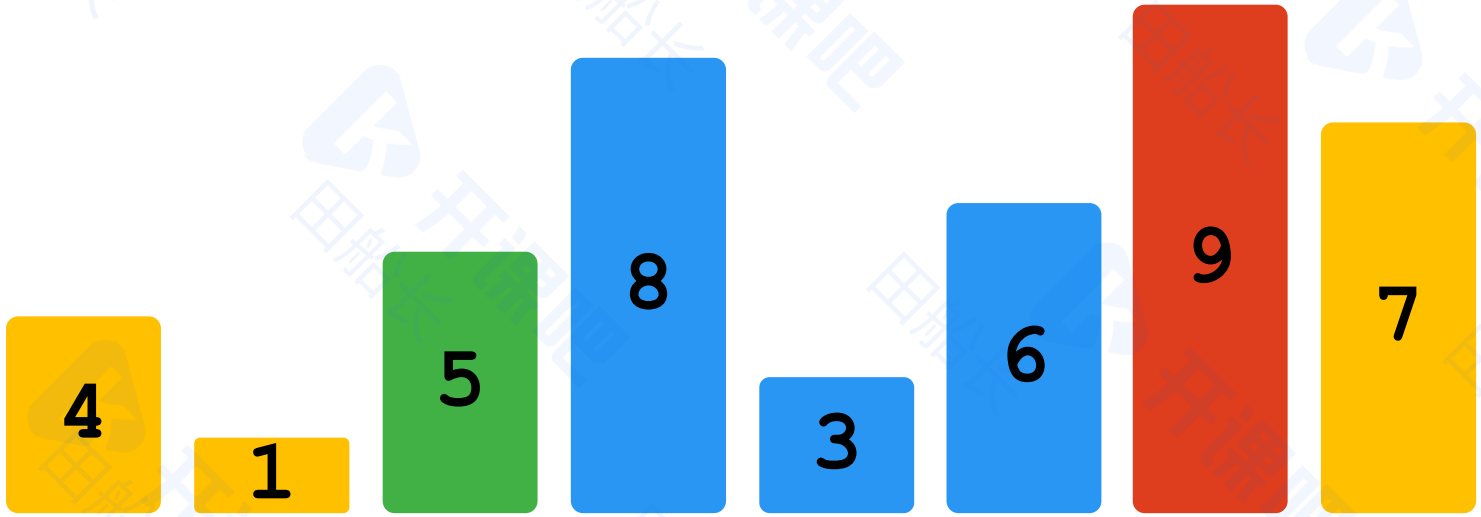
比较最左边的数1和PIVOT即5的大小



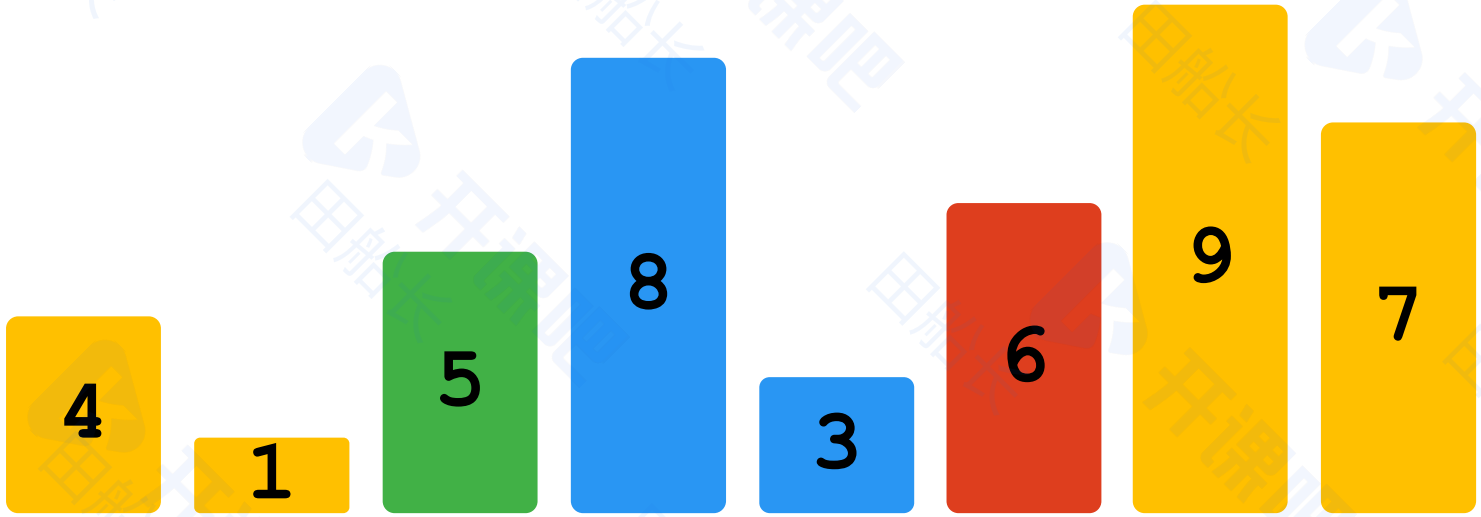
比较最左边的数9和PIVOT即5的大小



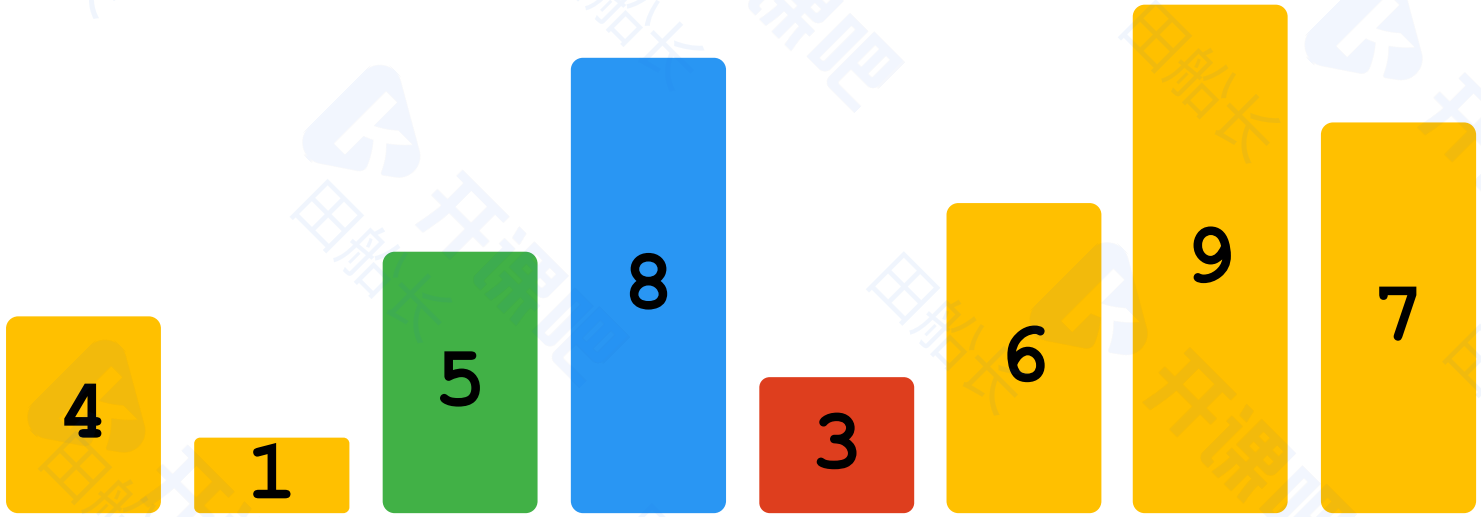
顺序不满足，交换9和PIVOT即5，换边扫描



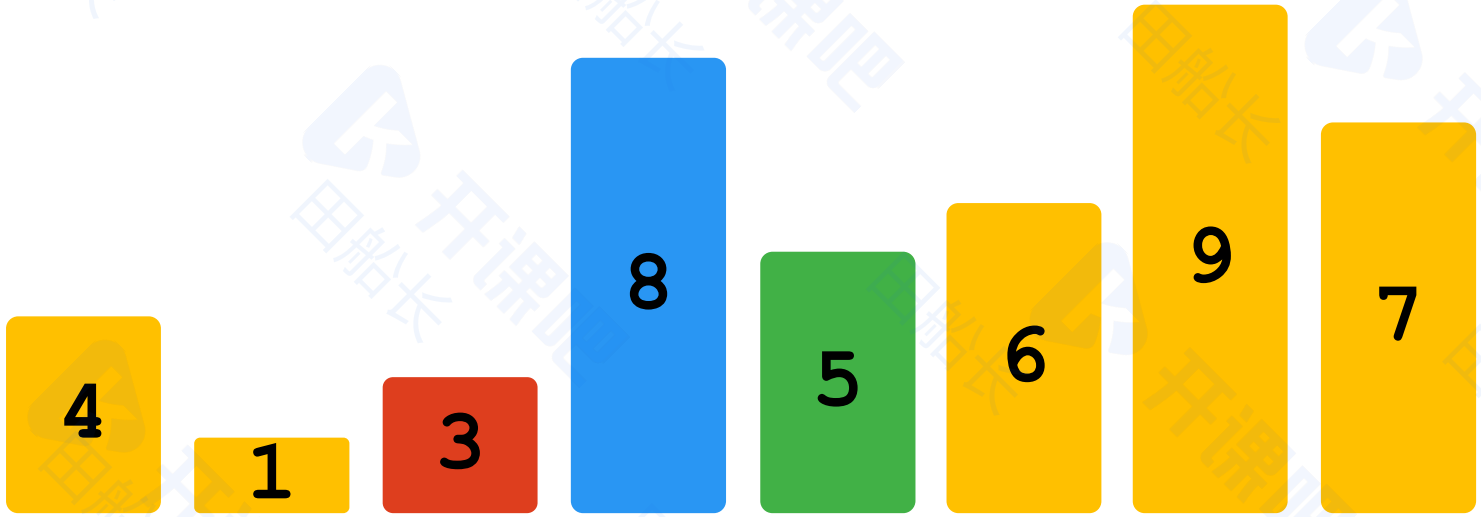
比较最右边的数6和PIVOT即5的大小



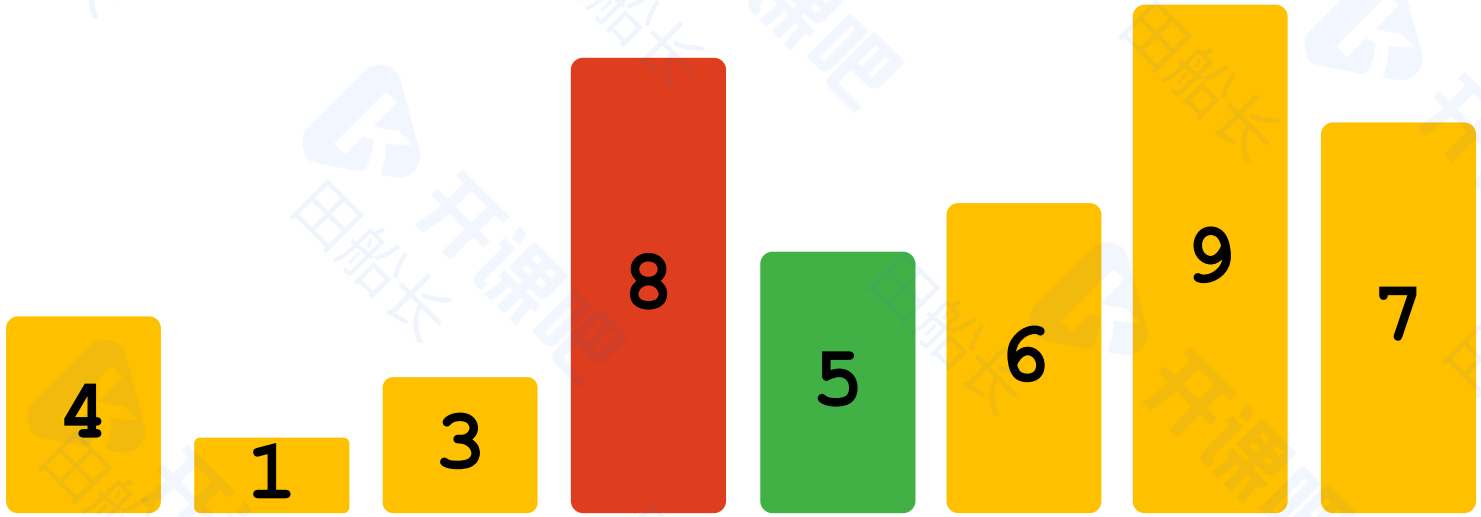
比较最右边的数3和PIVOT即5的大小



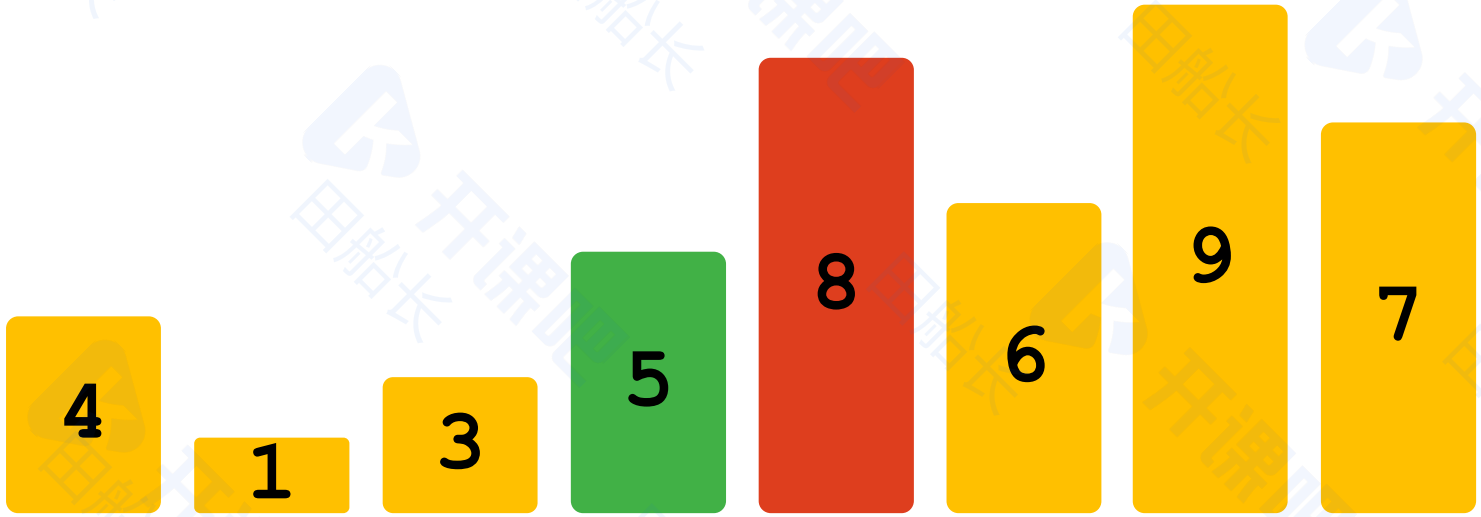
顺序不满足，交换3和PIVOT即5，换边扫描



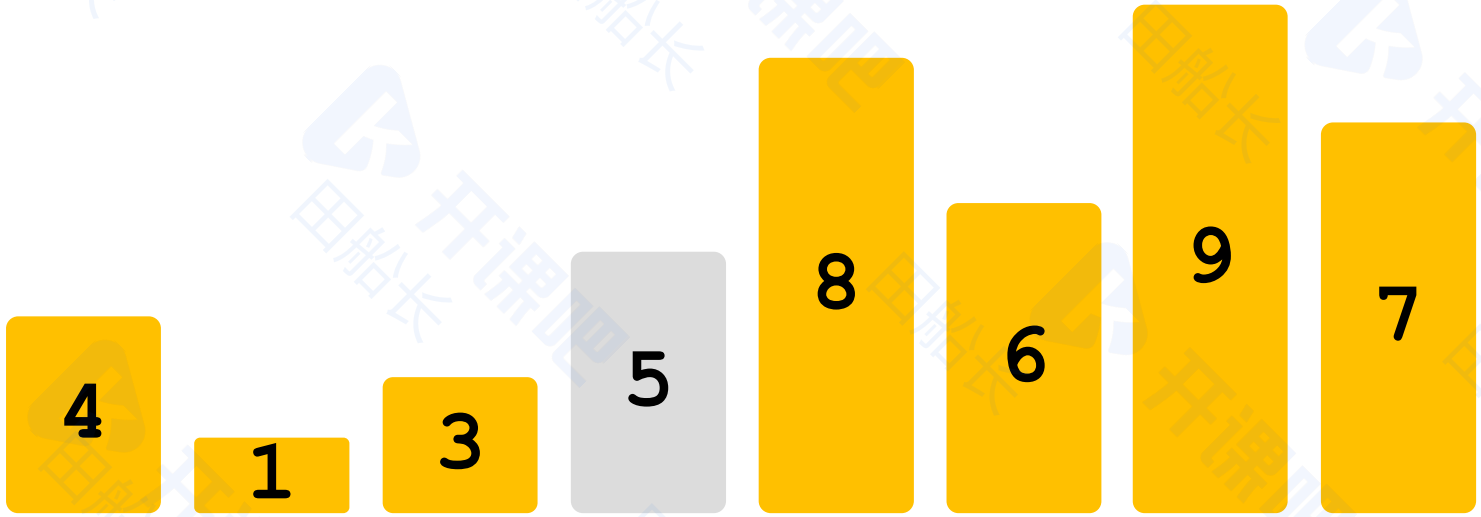
比较最左边的数8和PIVOT即5的大小



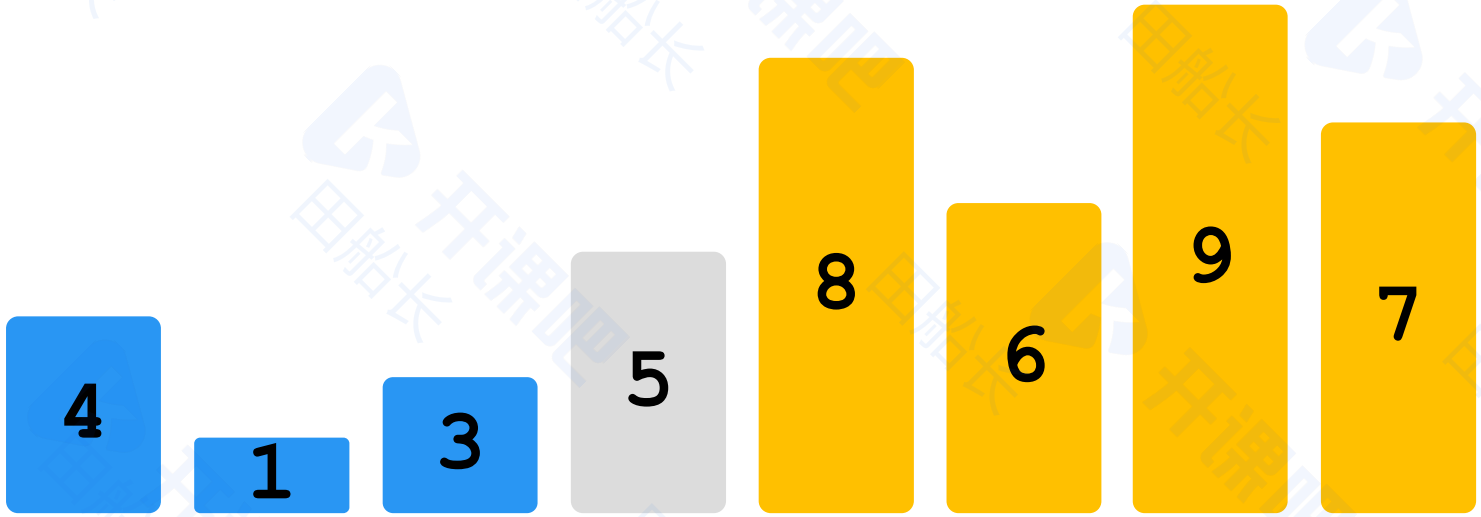
顺序不满足，交换8和PIVOT即5，换边扫描



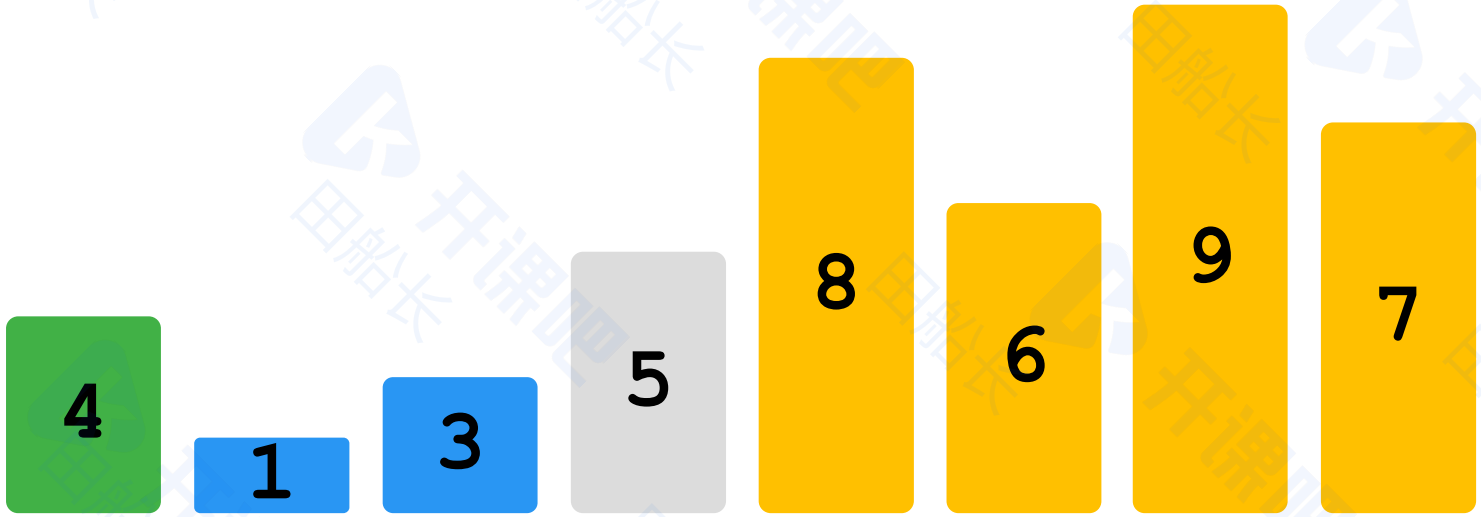
现在PIVOT即5已经在正确的位置上了，下面考虑它的左边和右边



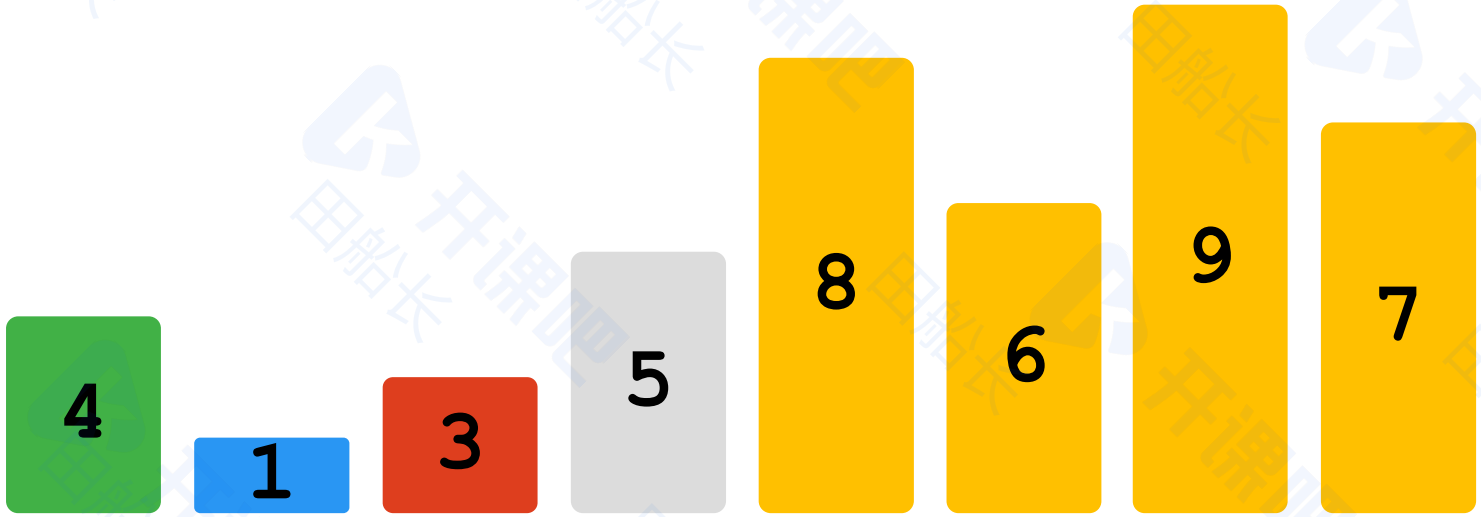
对区间[0,3)内的数进行递归的快速排序



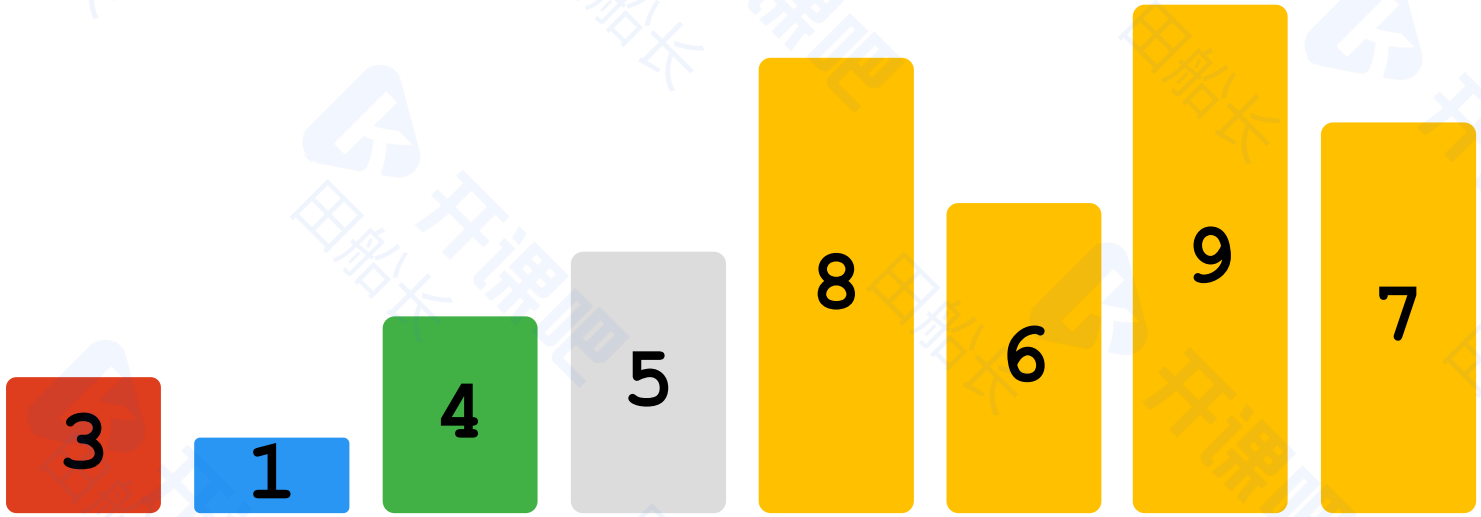
选择4作为本轮的
PIVOT



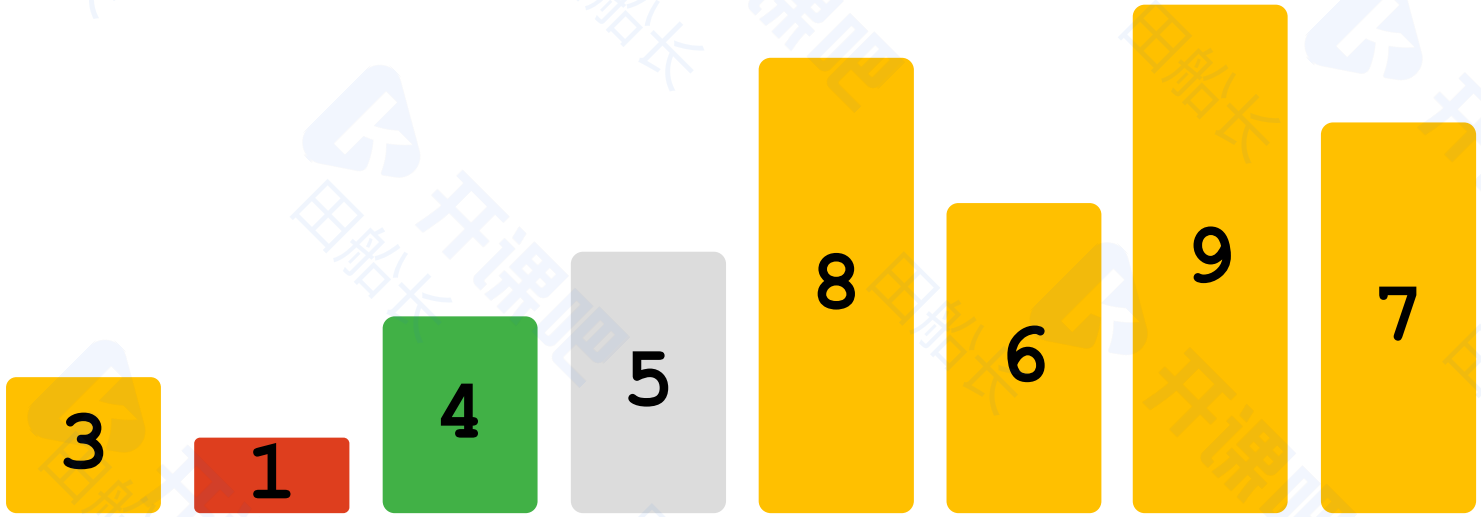
比较最右边的数3和PIVOT即4的大小



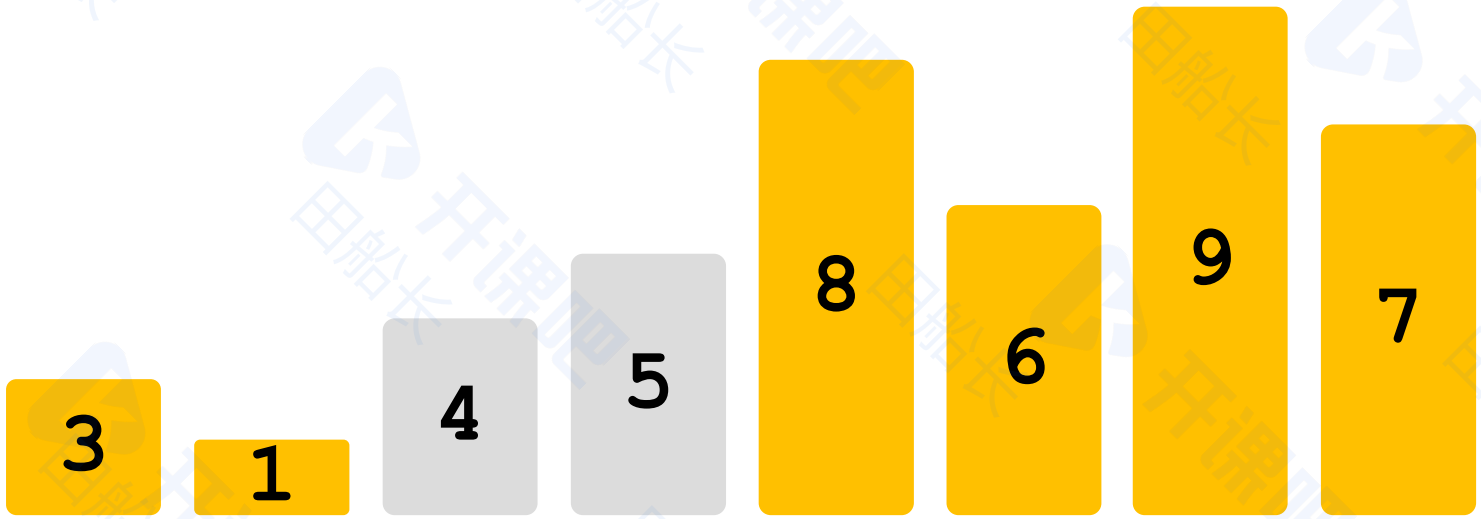
顺序不满足，交换3和PIVOT即4，换边扫描



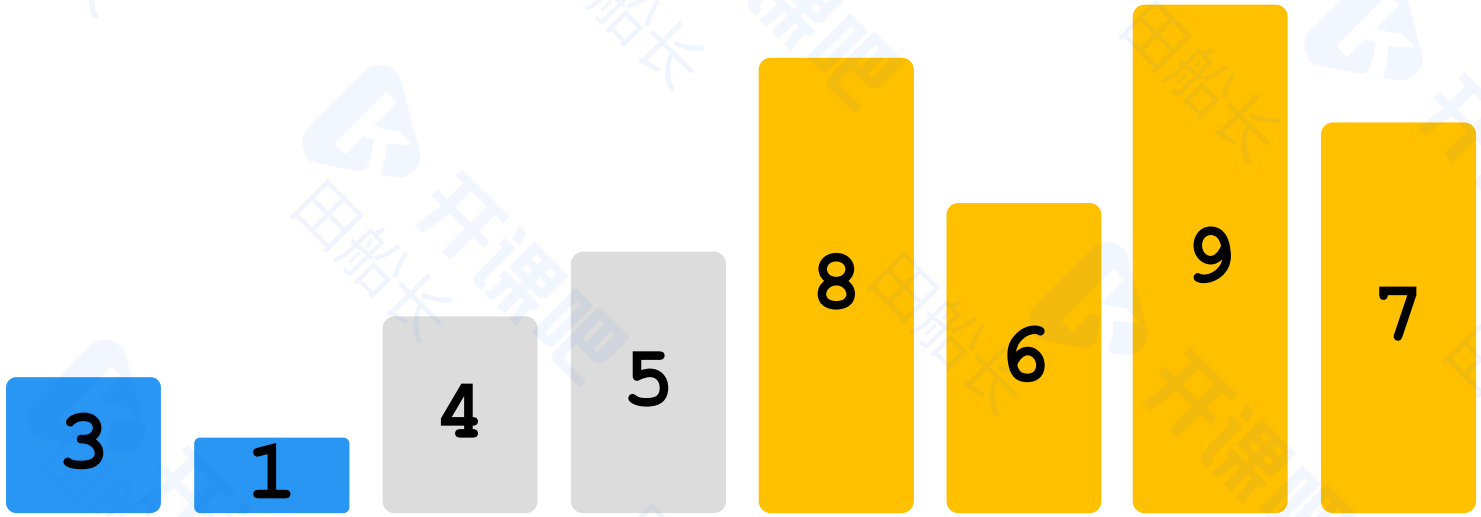
比较最左边的数1和PIVOT即4的大小



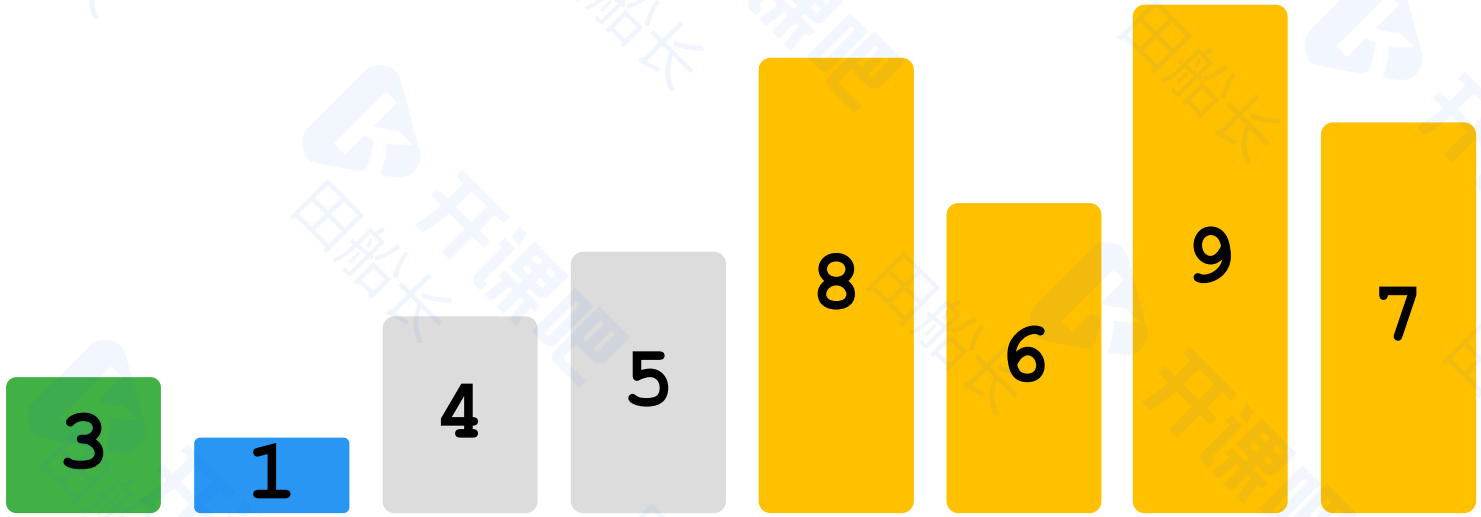
现在PIVOT即4已经在正确的位置上了，下面考虑它的左边和右边



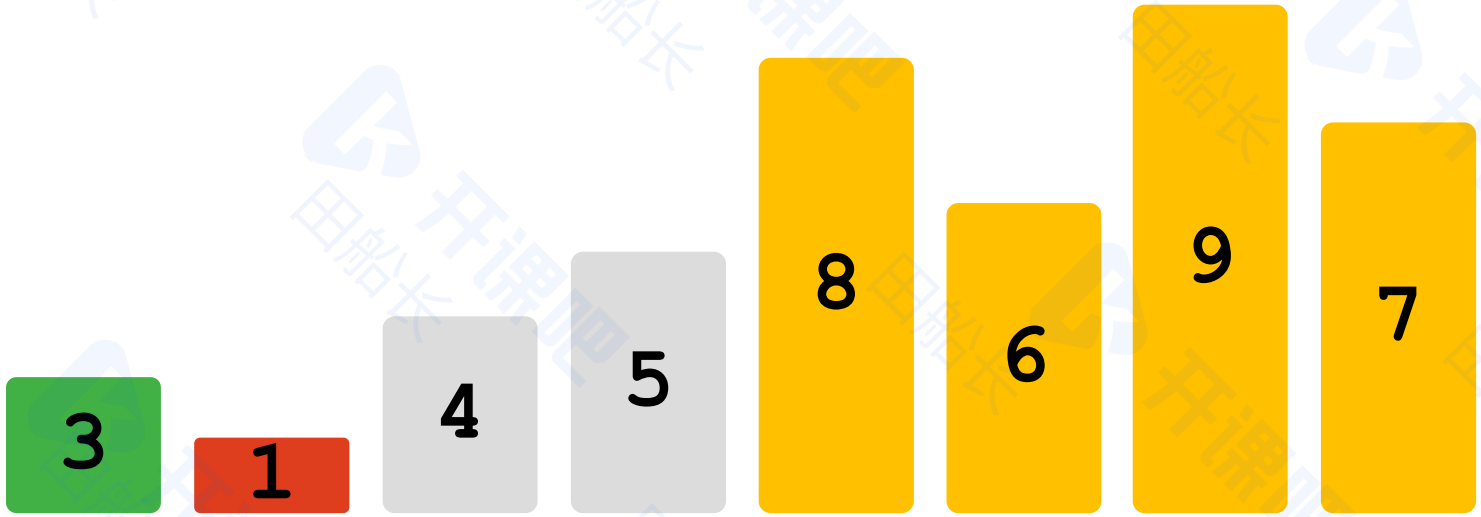
对区间[0,2)内的数进行递归的快速排序



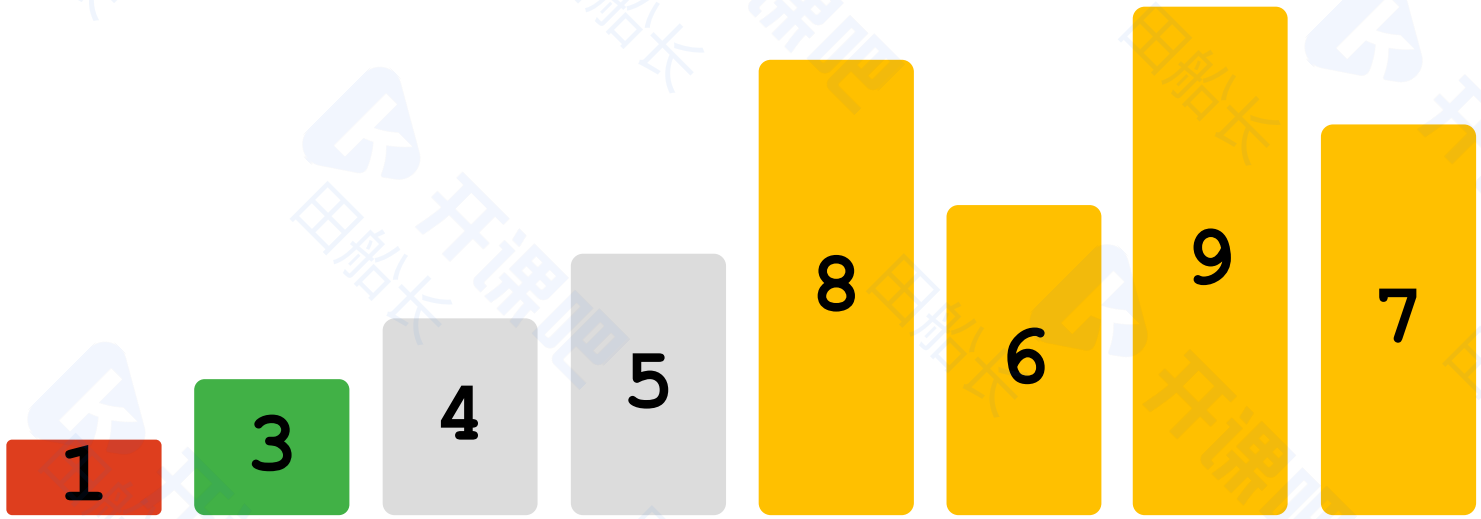
选择3作为本轮的
PIVOT



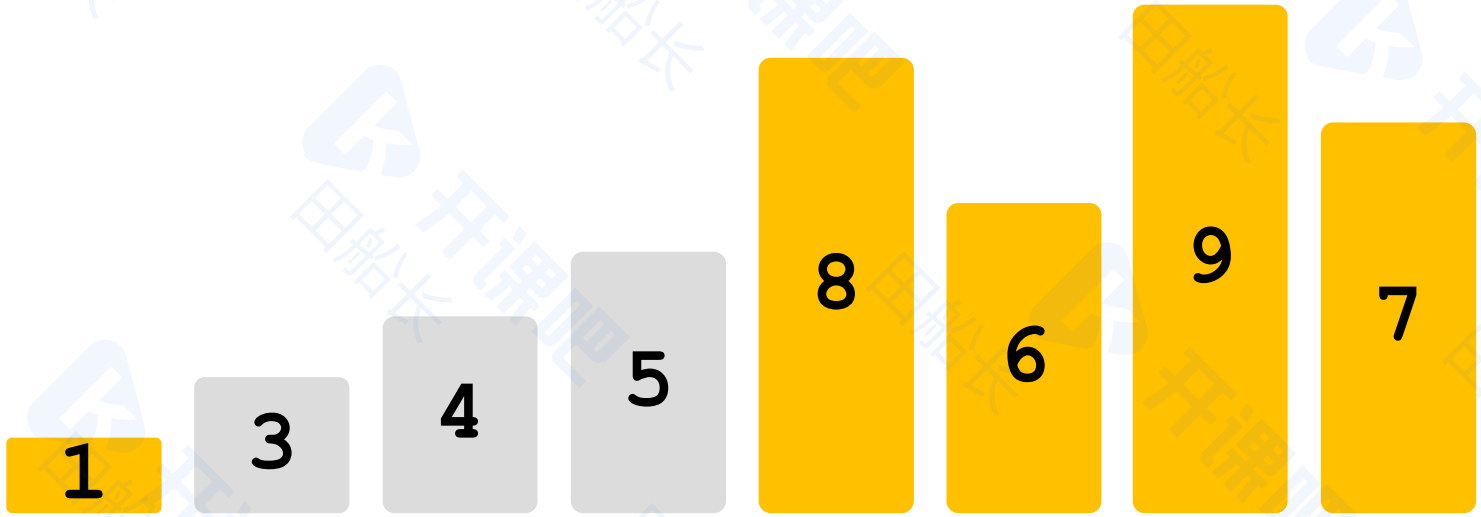
比较最右边的数1和PIVOT即3的大小



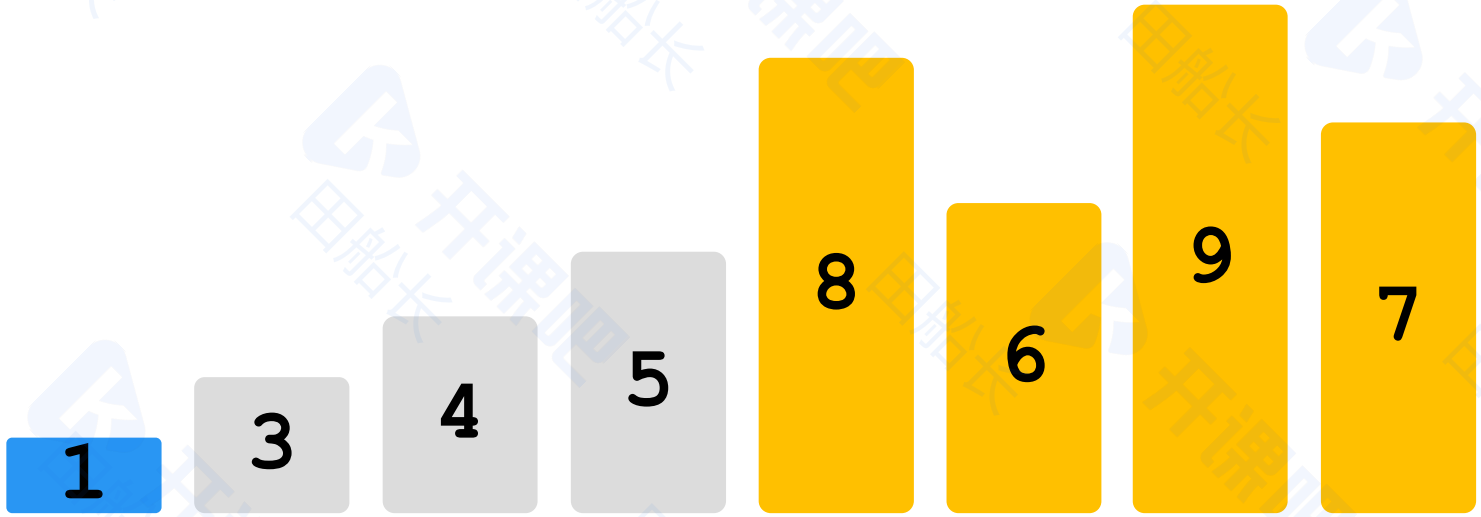
顺序不满足，交换1和PIVOT即3，换边扫描



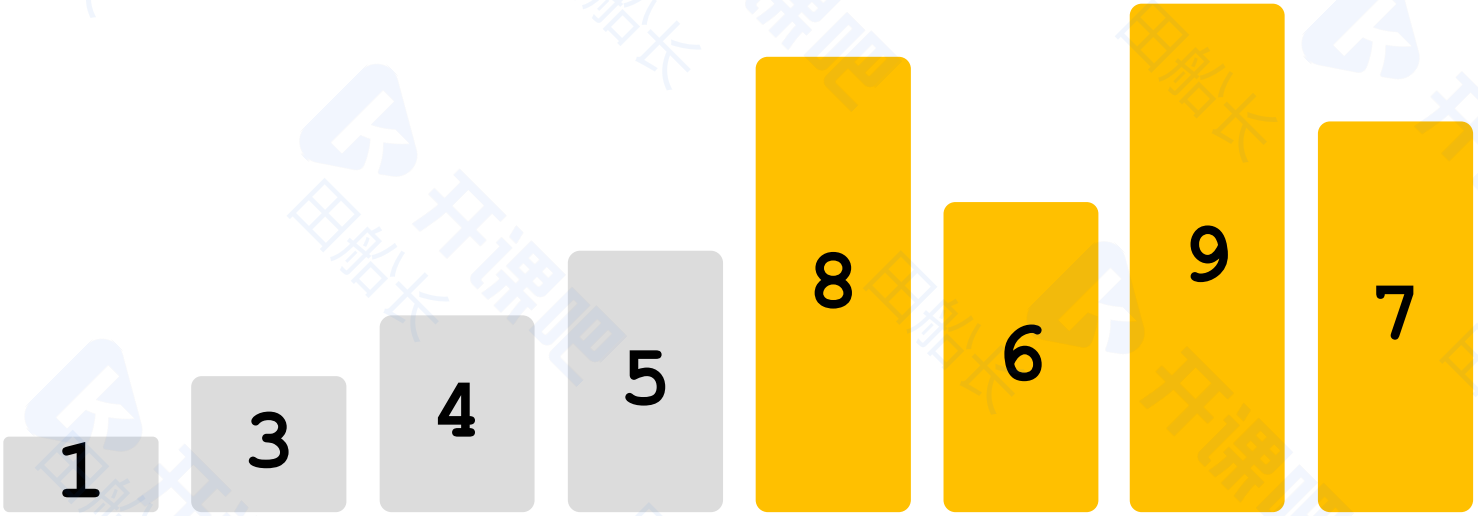
现在PIVOT即3已经在正确的位置上了，下面考虑它的左边和右边



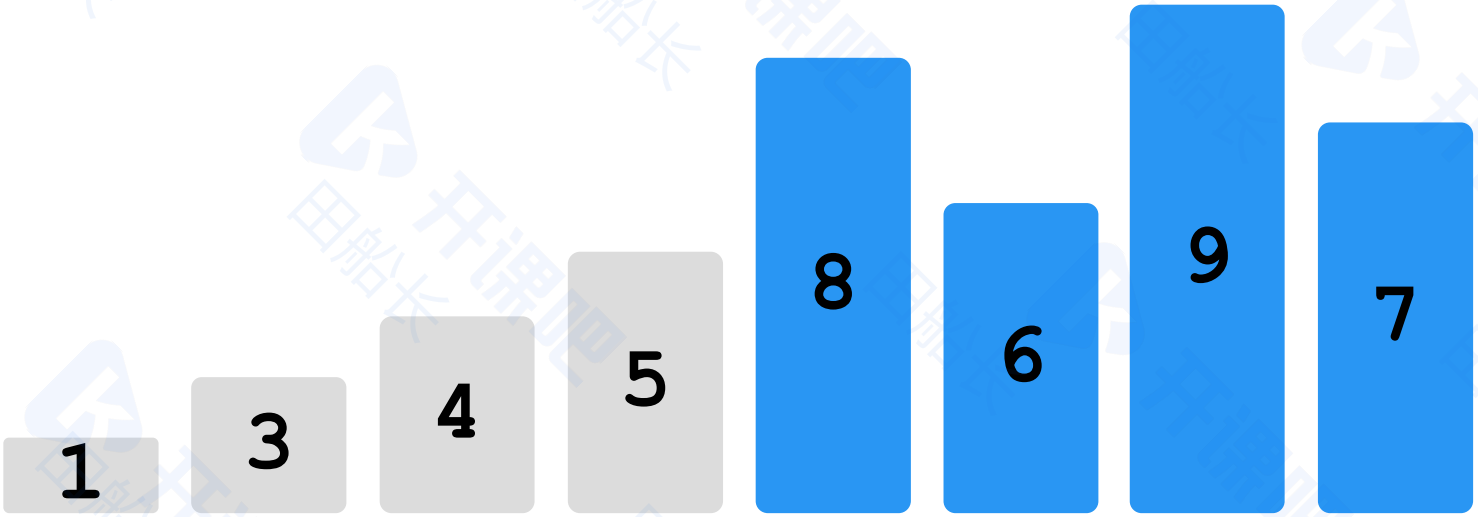
对区间[0,1)内的数进行递归的快速排序



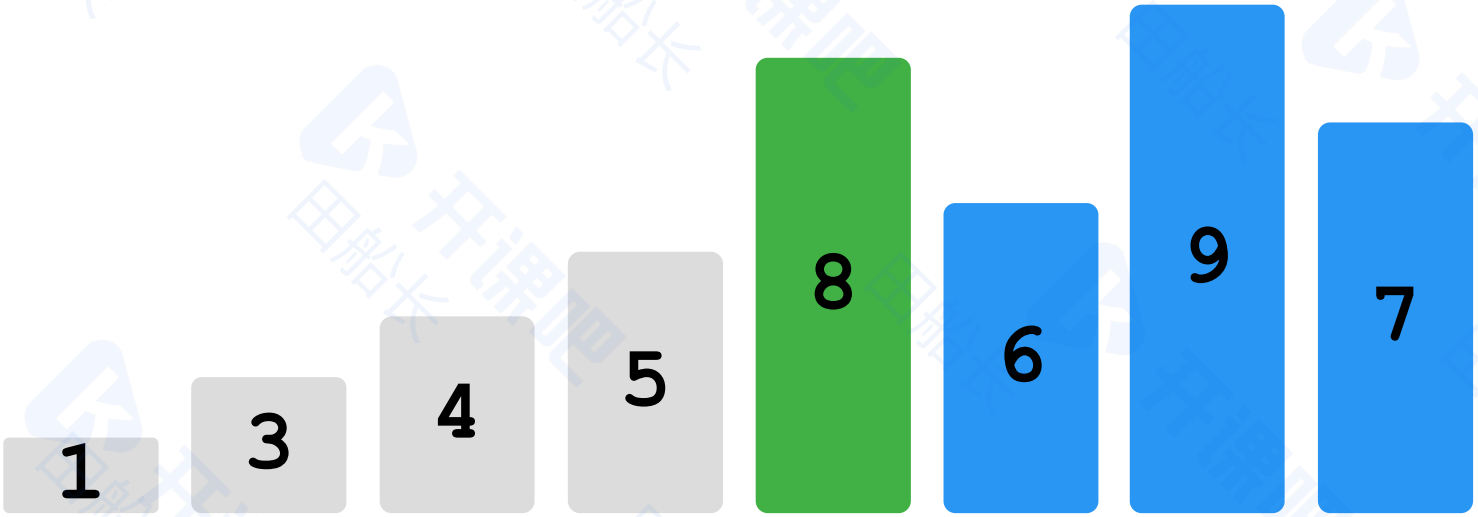
数组长度为1，1已经在正确的位置上了



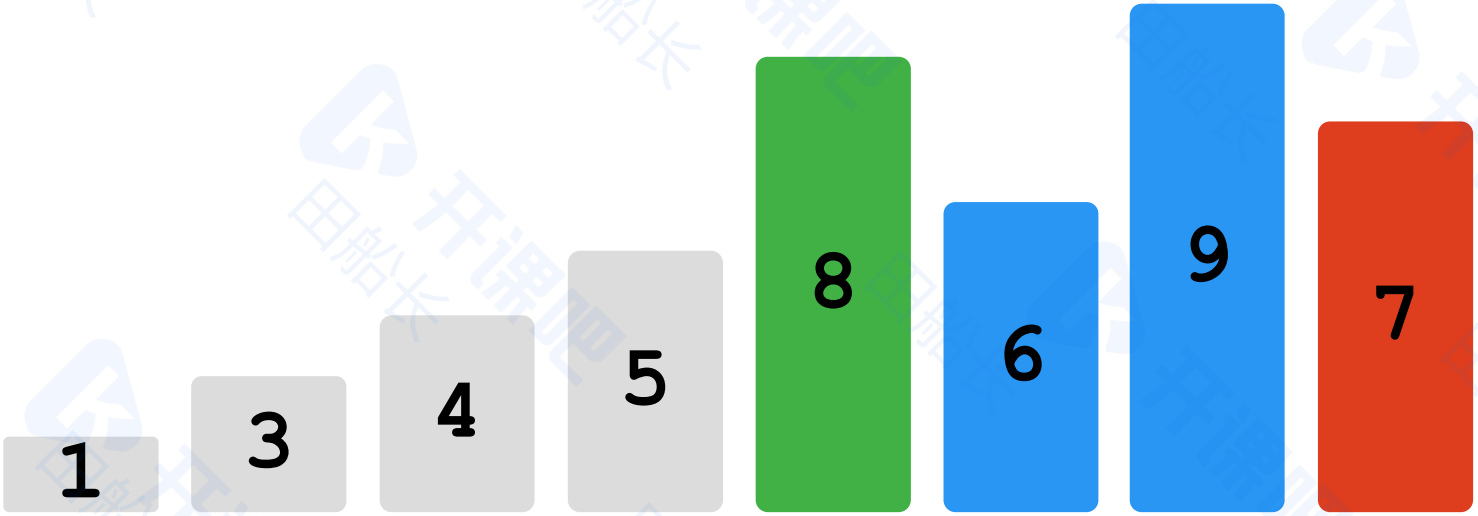
对区间[4,8)内的数进行递归的快速排序



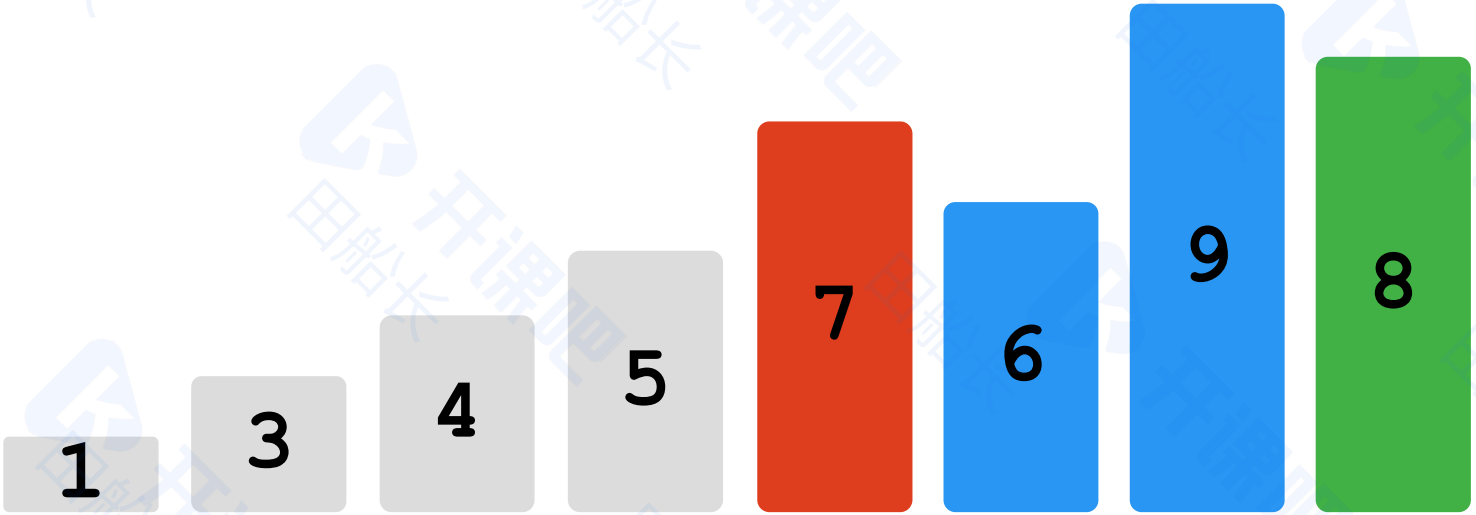
选择8作为本轮的
PIVOT



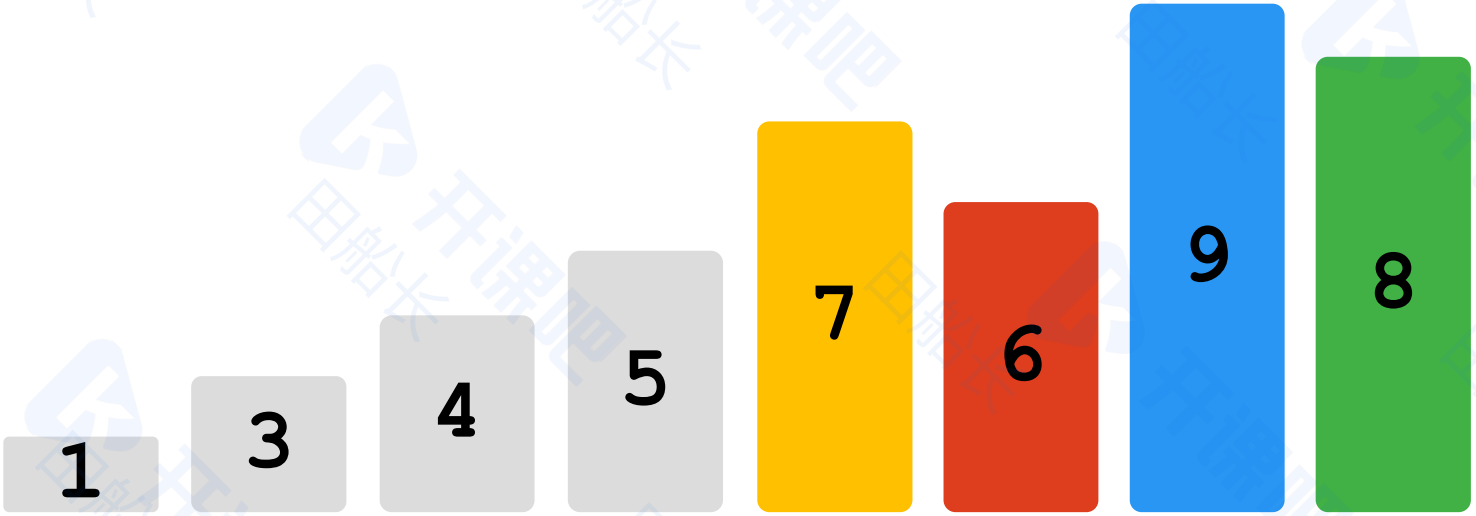
比较最右边的数7和PIVOT即8的大小



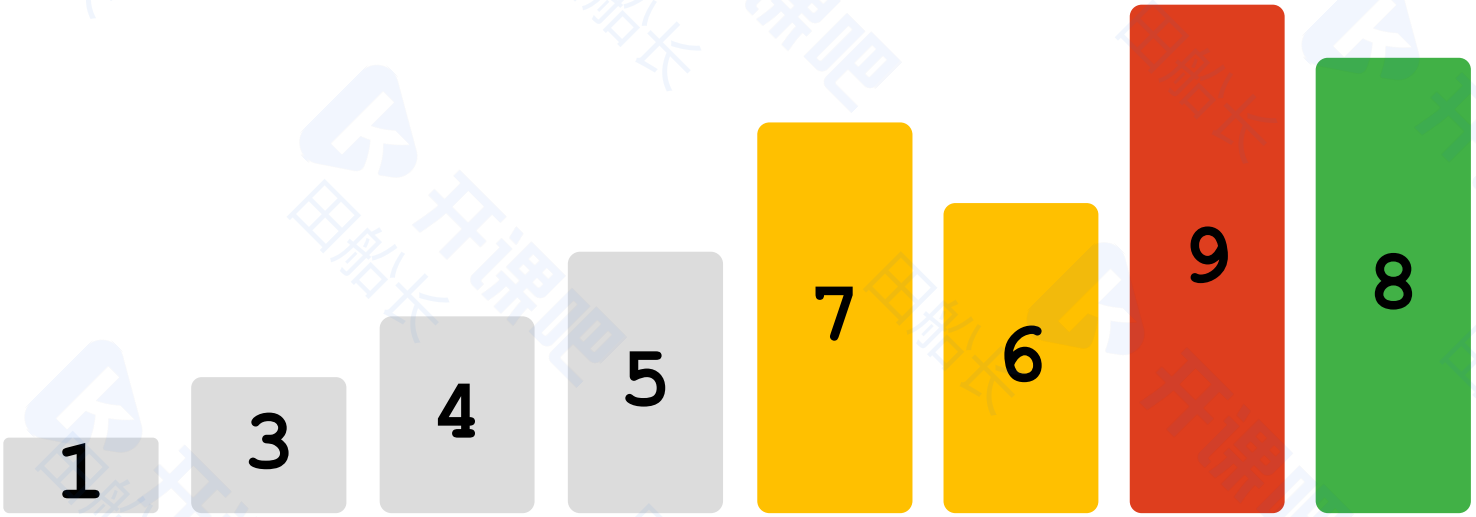
顺序不满足，交换7和PIVOT即8，换边扫描



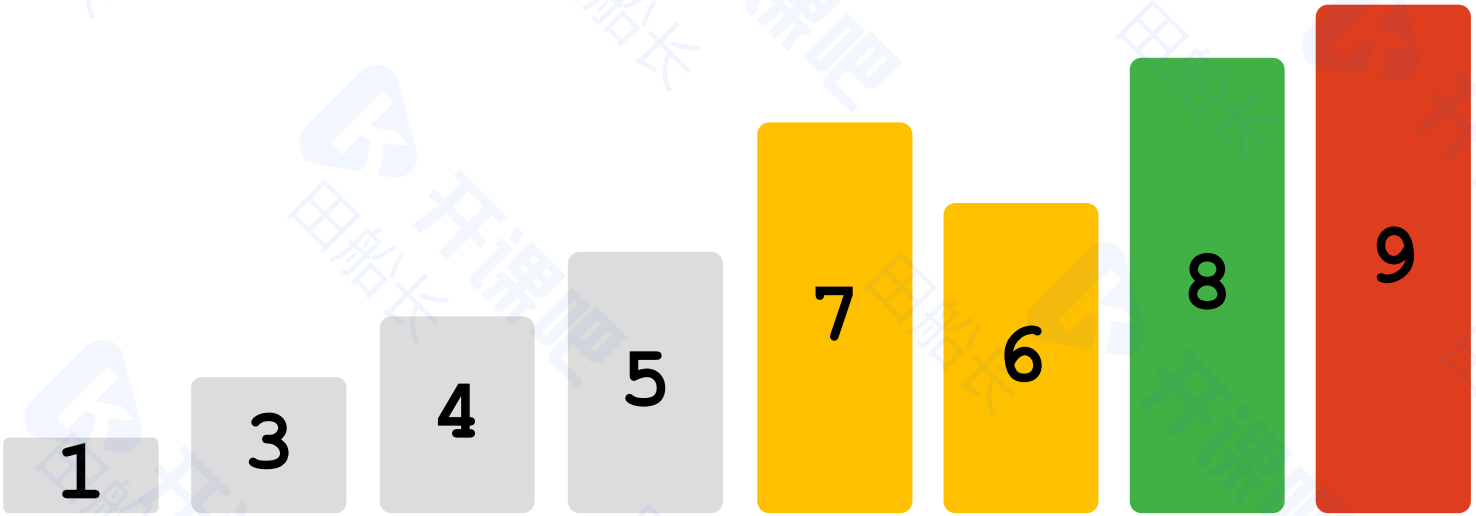
比较最左边的数6和PIVOT即8的大小



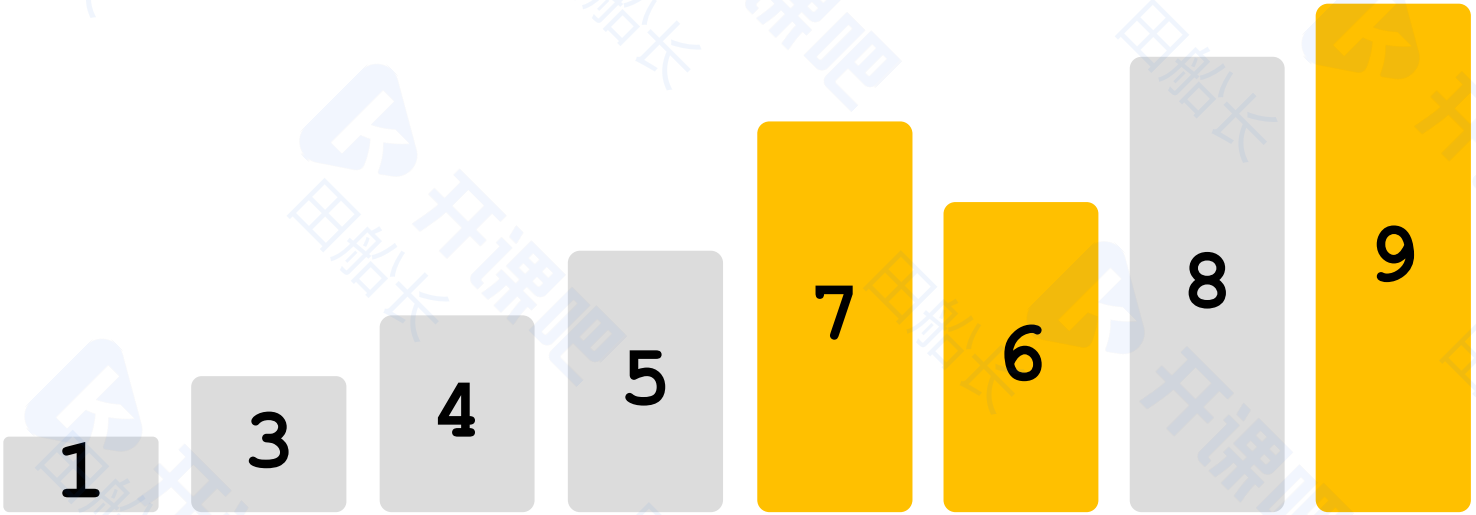
比较最左边的数9和PIVOT即8的大小



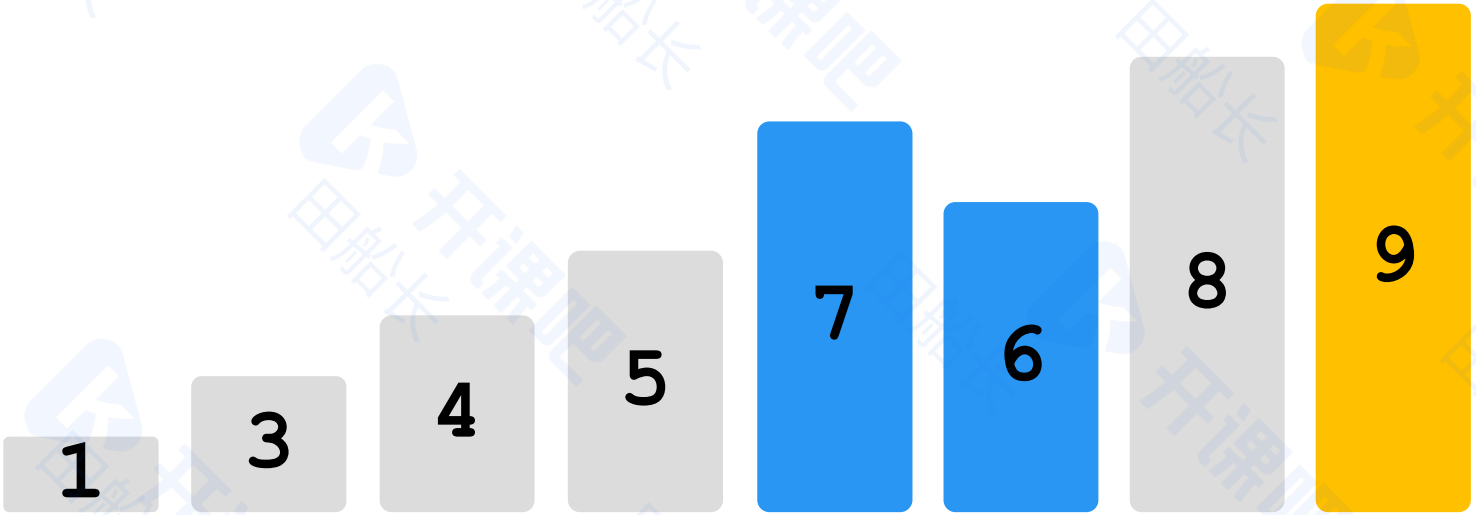
顺序不满足，交换9和PIVOT即8，换边扫描



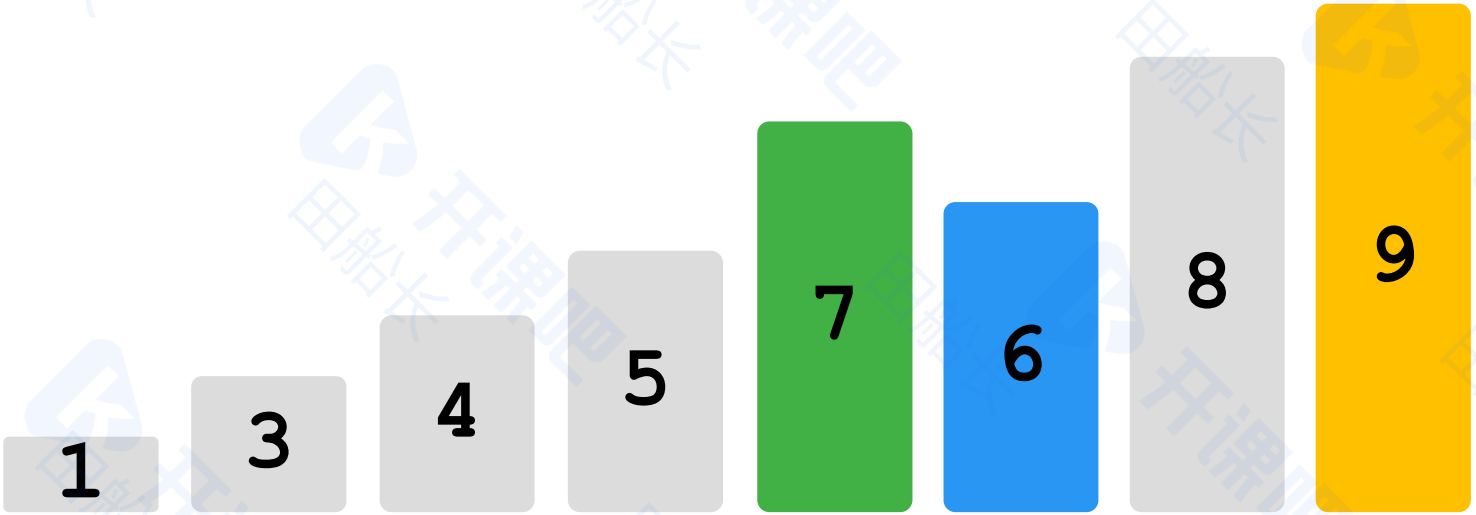
现在PIVOT即8已经在正确的位置上了，下面考虑它的左边和右边



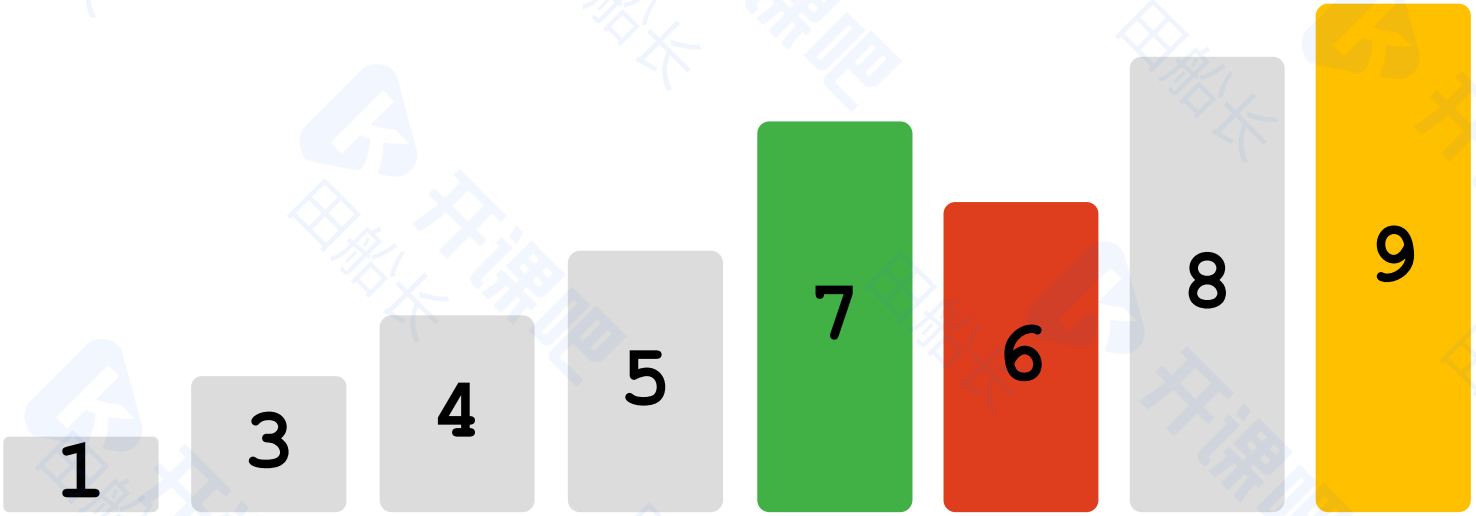
对区间[4,6)内的数进行递归的快速排序



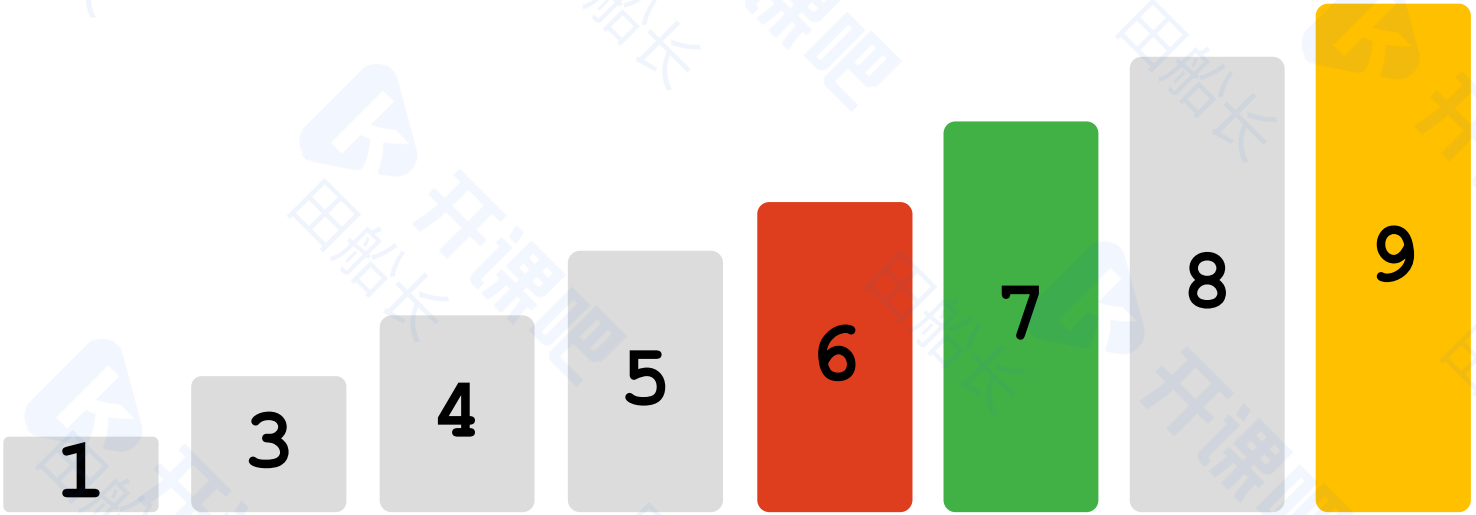
选择7作为本轮的PIVOT



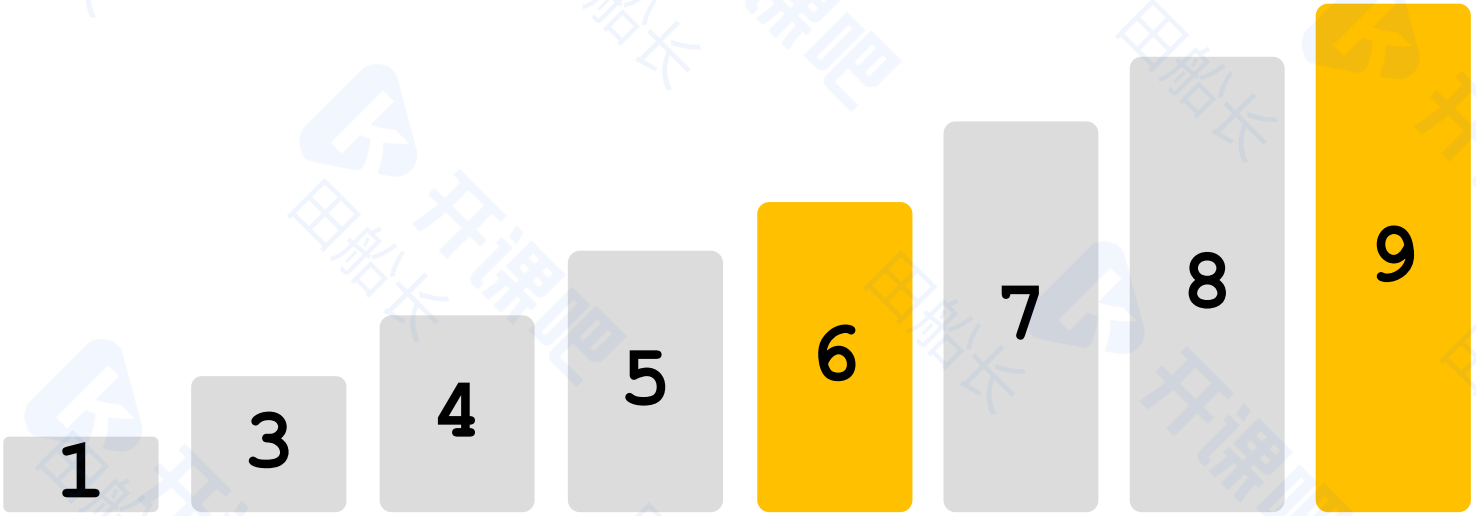
比较最右边的数6和PIVOT即7的大小



顺序不满足，交换6和PIVOT即7，换边扫描



现在PIVOT即7已经在正确的位置上了，下面考虑它的左边和右边



对区间[4,5)内的数进行递归的快速排序

数组长度为1，6已经在正确的位置上了

对区间[7,8)内的数进行递归的快速排序

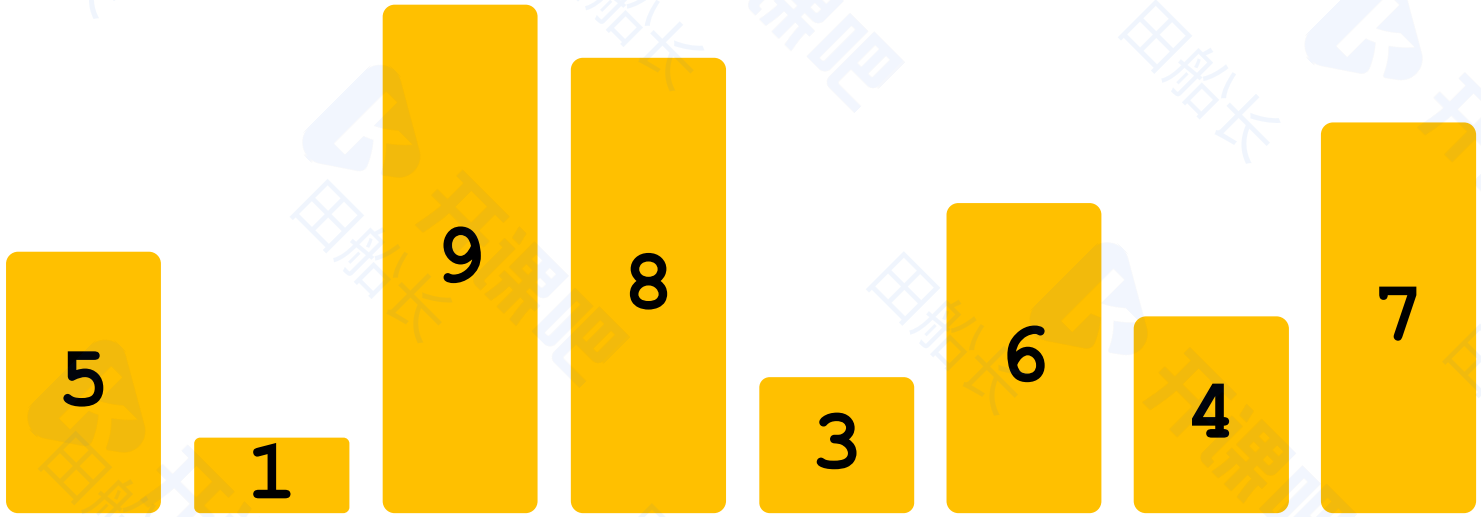
开课吧

数组长度为1，9已
经在正确的位置上了

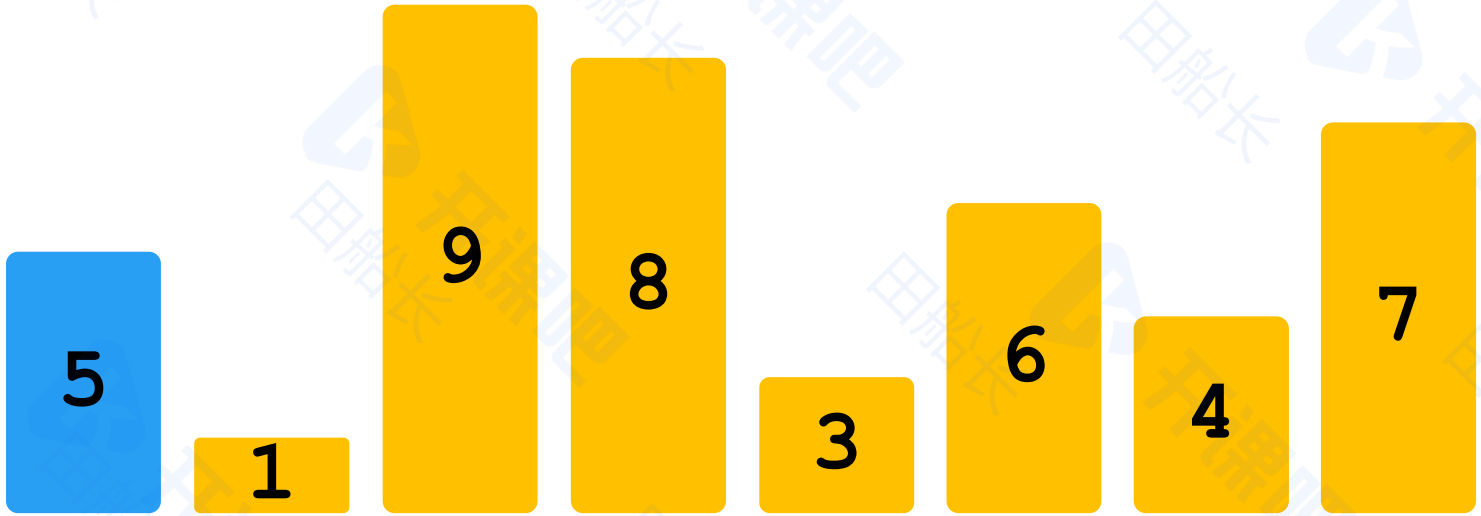
快速排序结束

选择类排序——选择排序

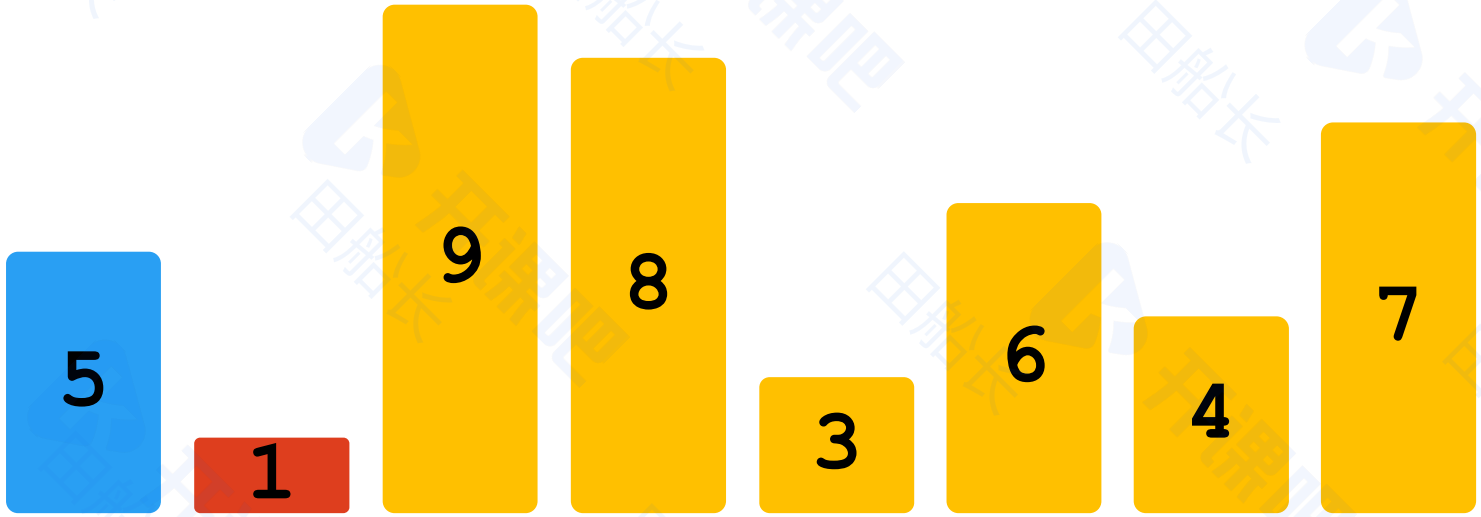
在待排序区间内找到一个最值元素
与待排序区间的第一个元素进行交换
使得已排序区间加一
重复上述过程，最终整个序列有序



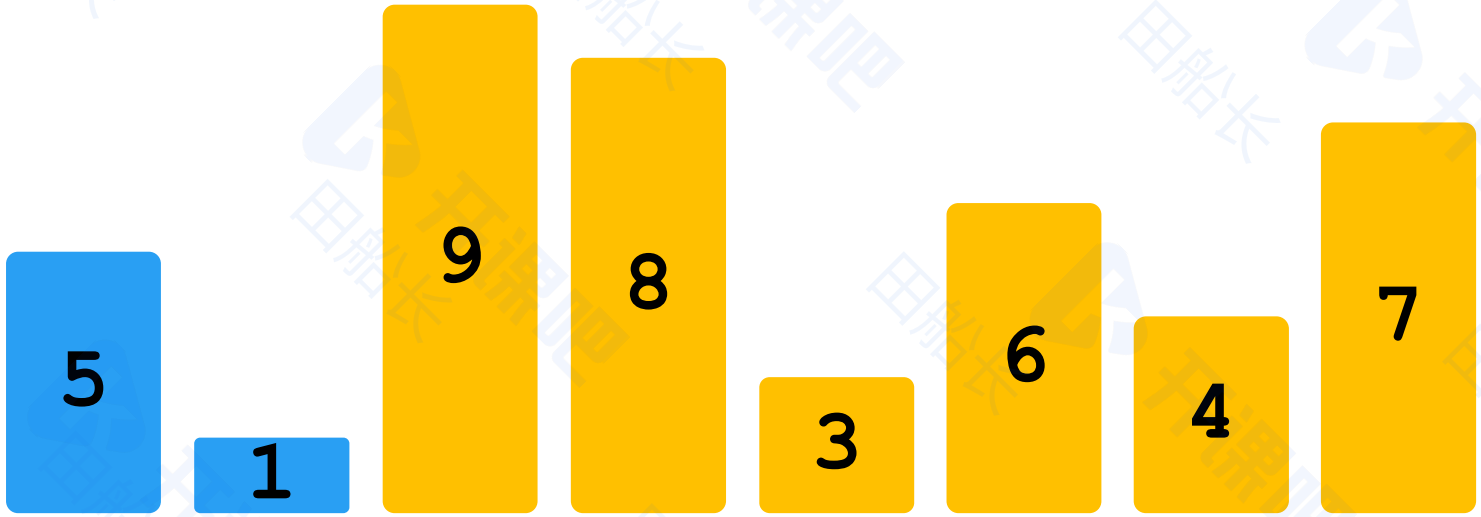
找出第0个及其之后的所有数字中的最小值



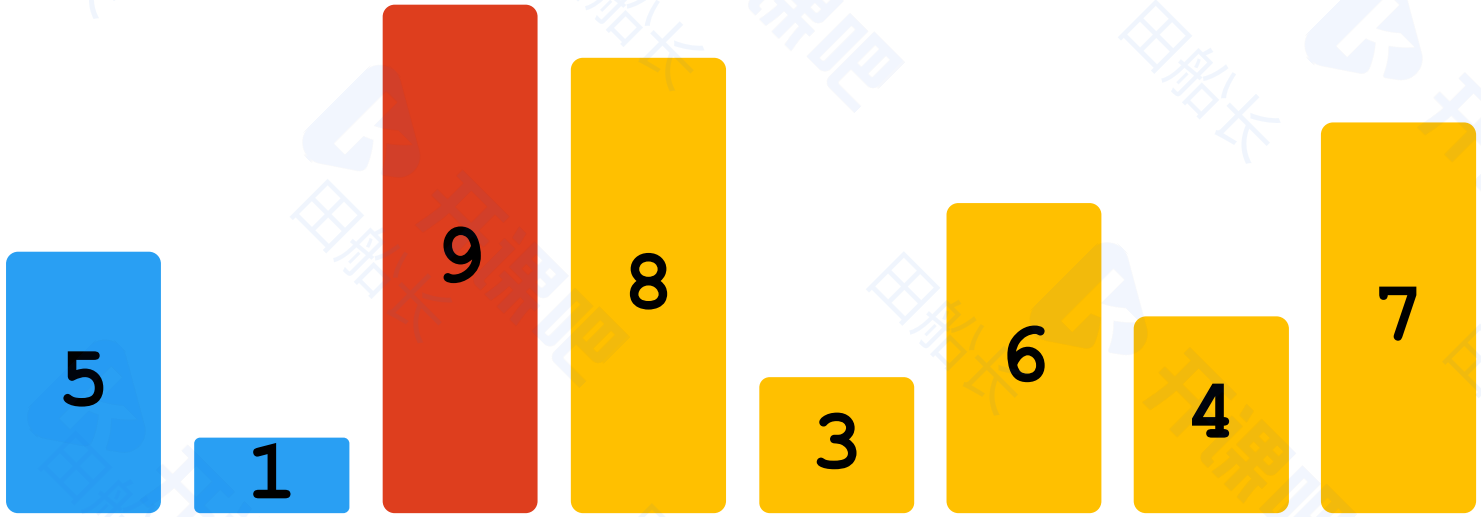
目前的最小值是5，
比较它与1



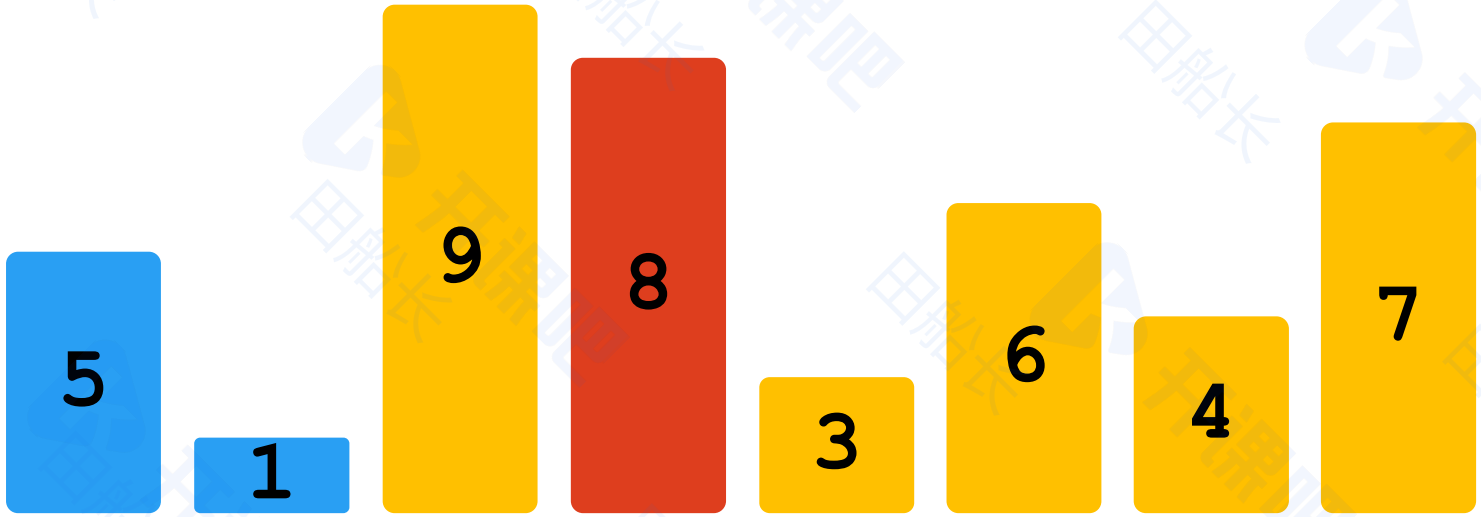
最小值更新为1



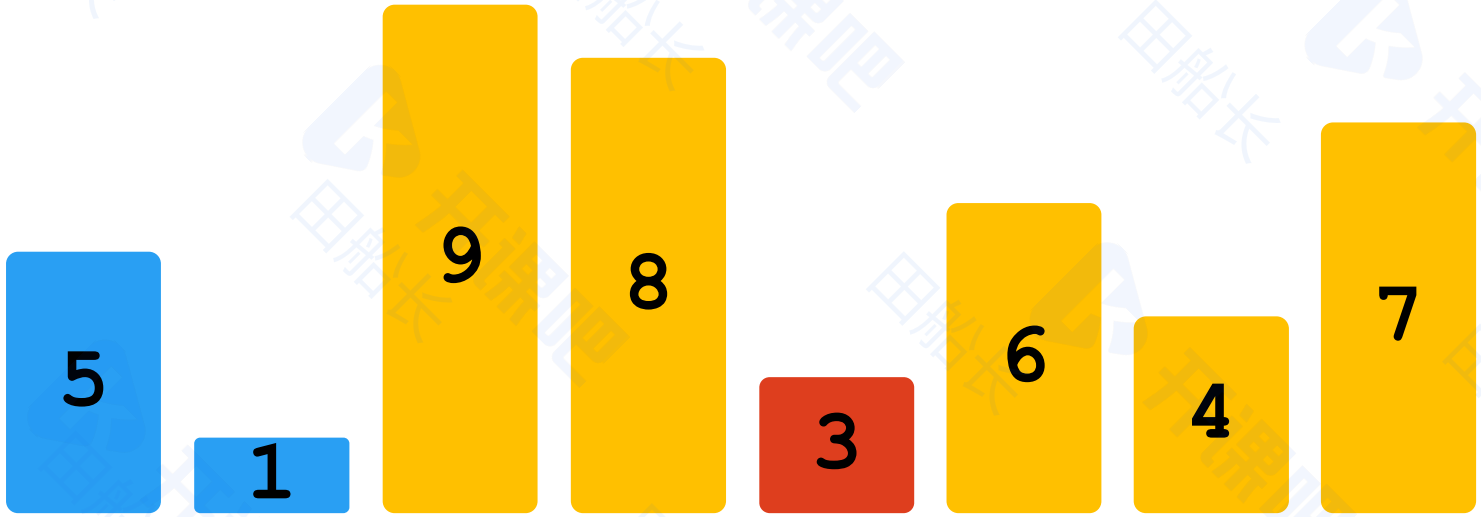
目前的最小值是1，
比较它与9



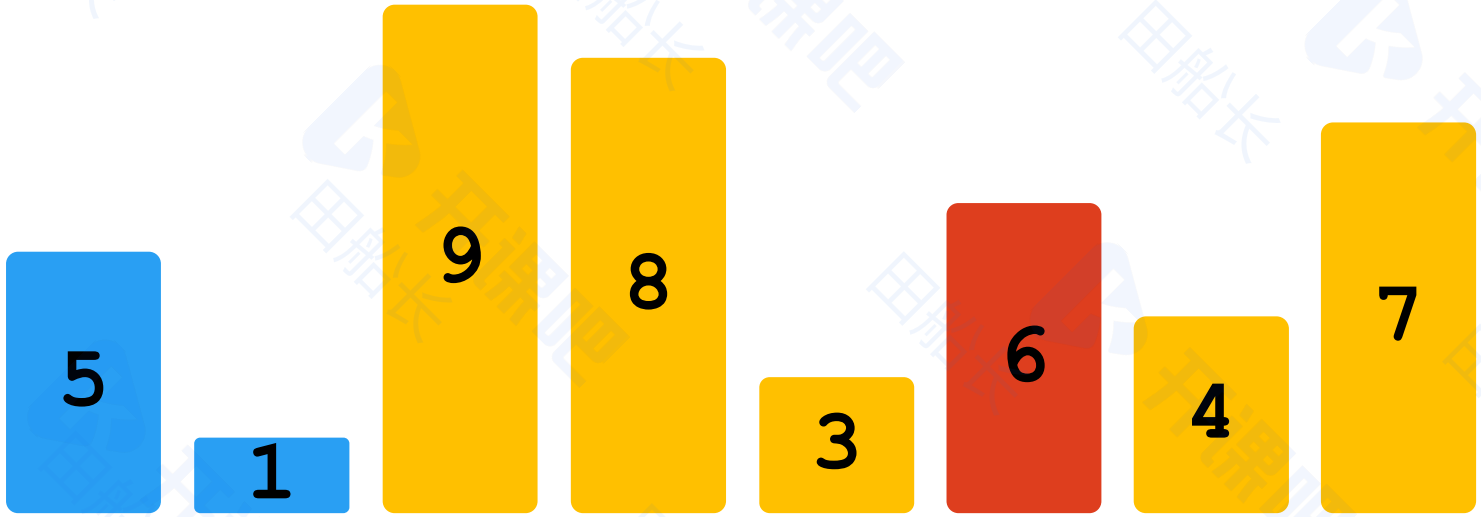
目前的最小值是1，
比较它与8



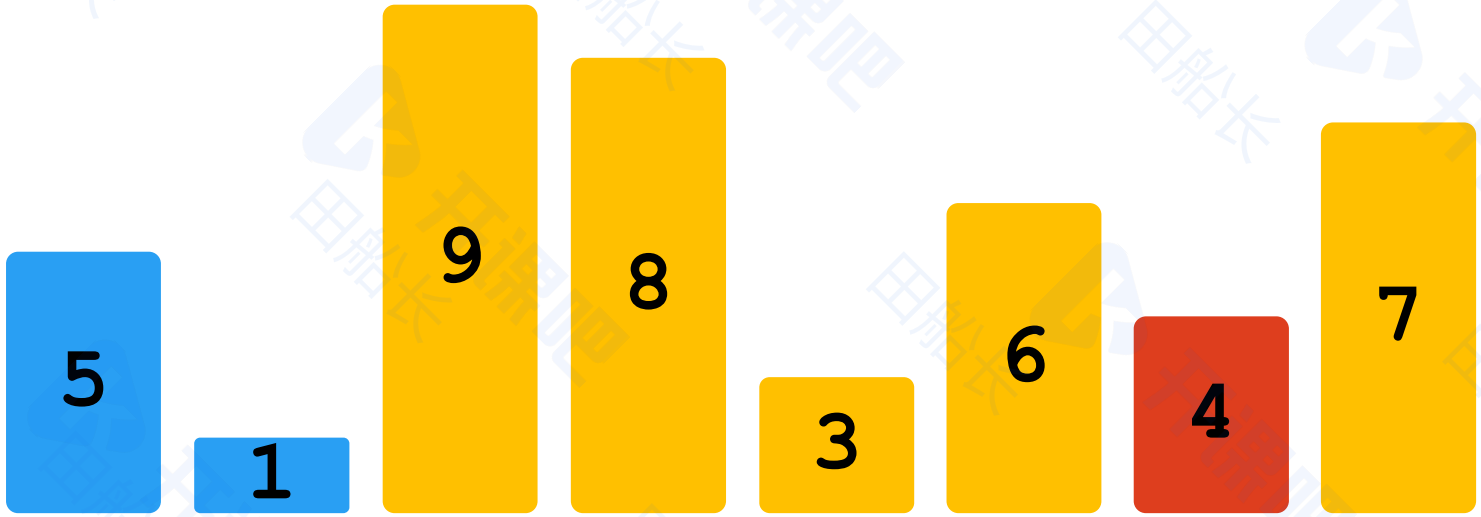
目前的最小值是1，
比较它与3



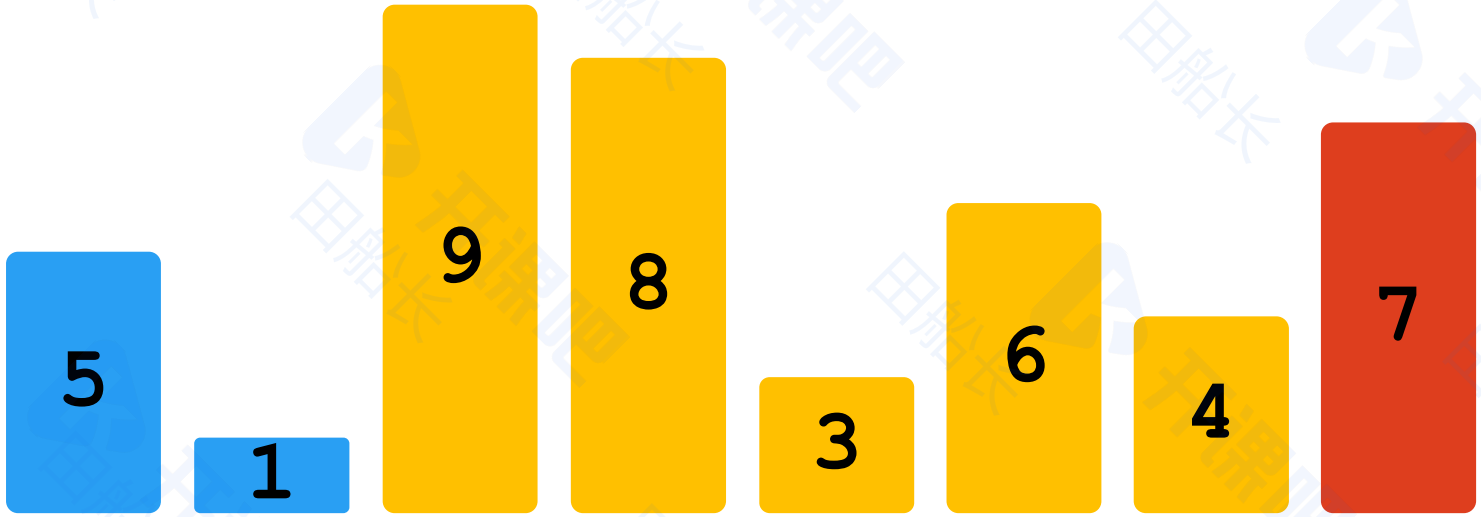
目前的最小值是1，
比较它与6



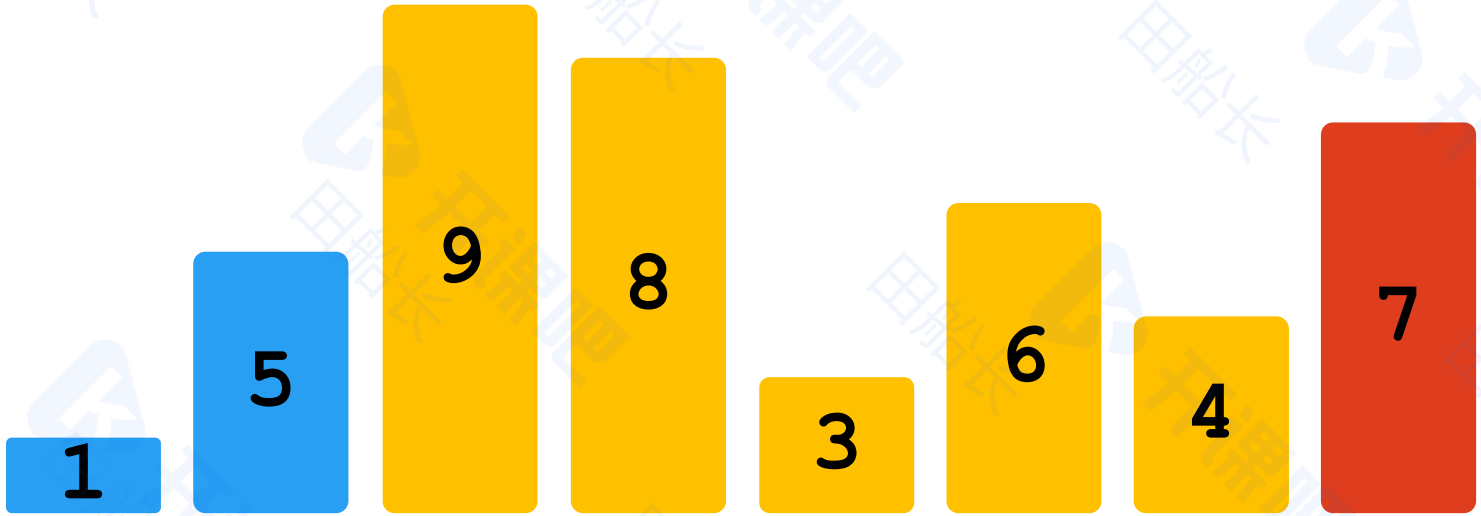
目前的最小值是1，
比较它与4



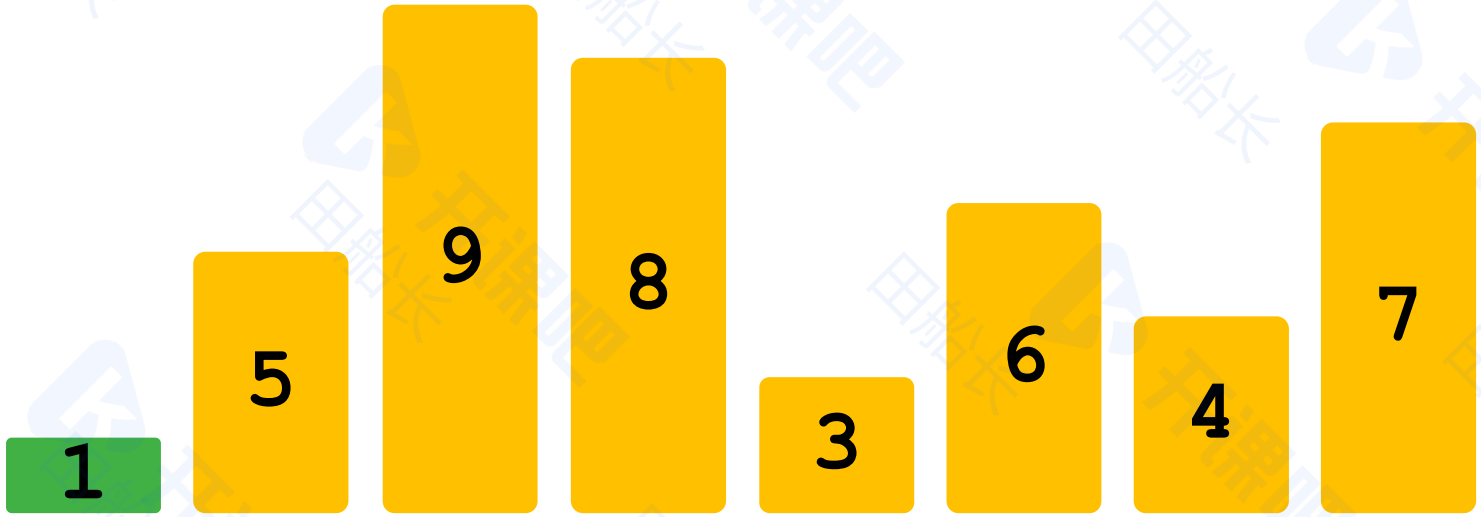
目前的最小值是1，
比较它与7



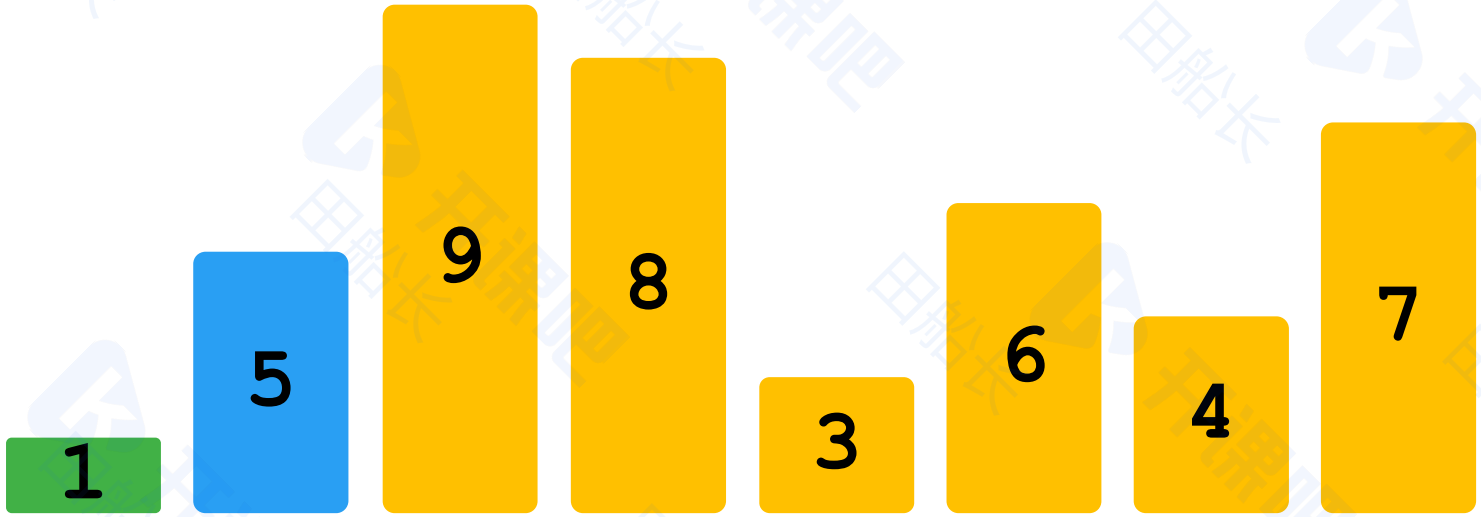
交换第0个数和其后面的最小值



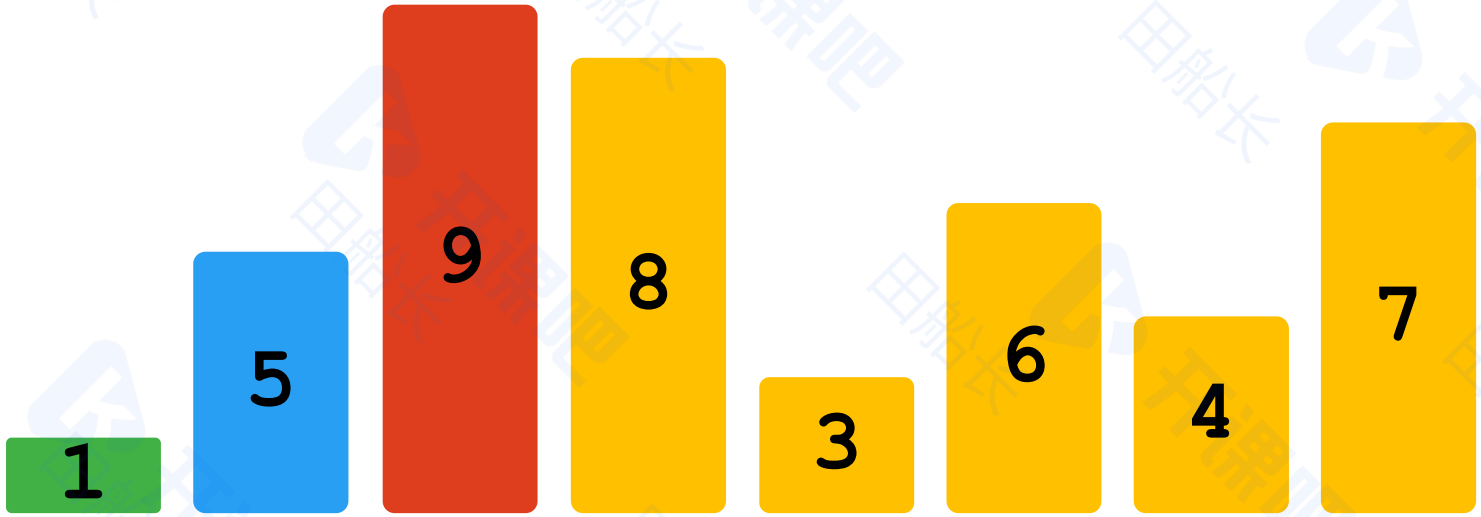
现在从第0个到第0个数已经在正确的位置上



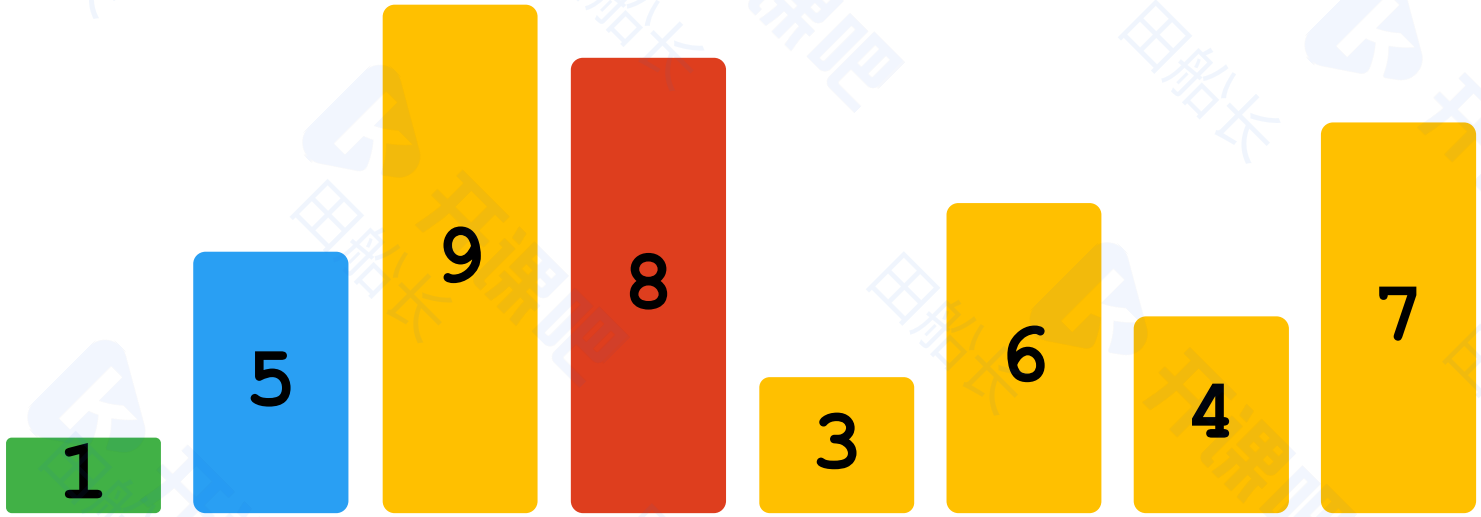
找出第1个及其之后的所有数字中的最小值



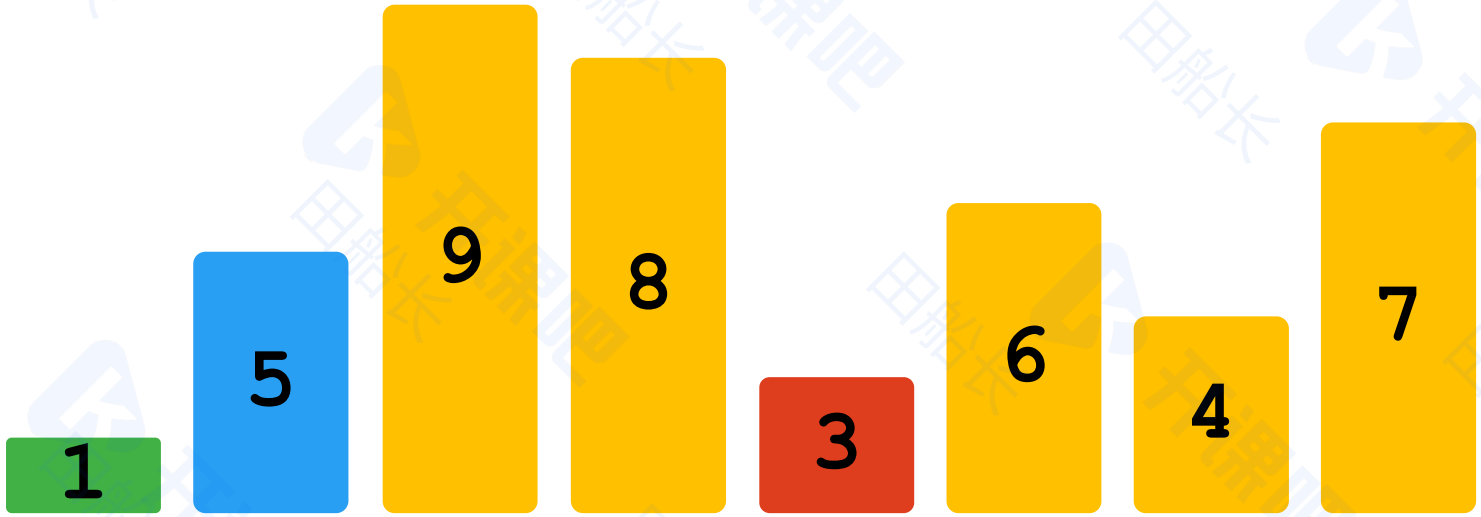
目前的最小值是5，
比较它与9



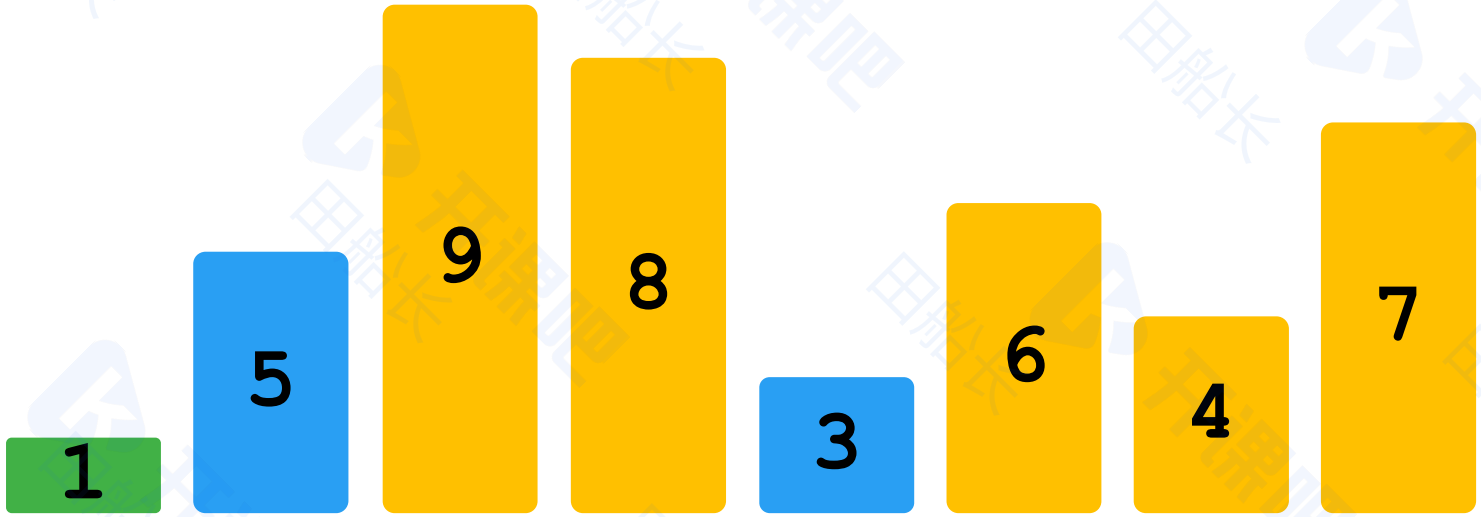
目前的最小值是5，
比较它与8



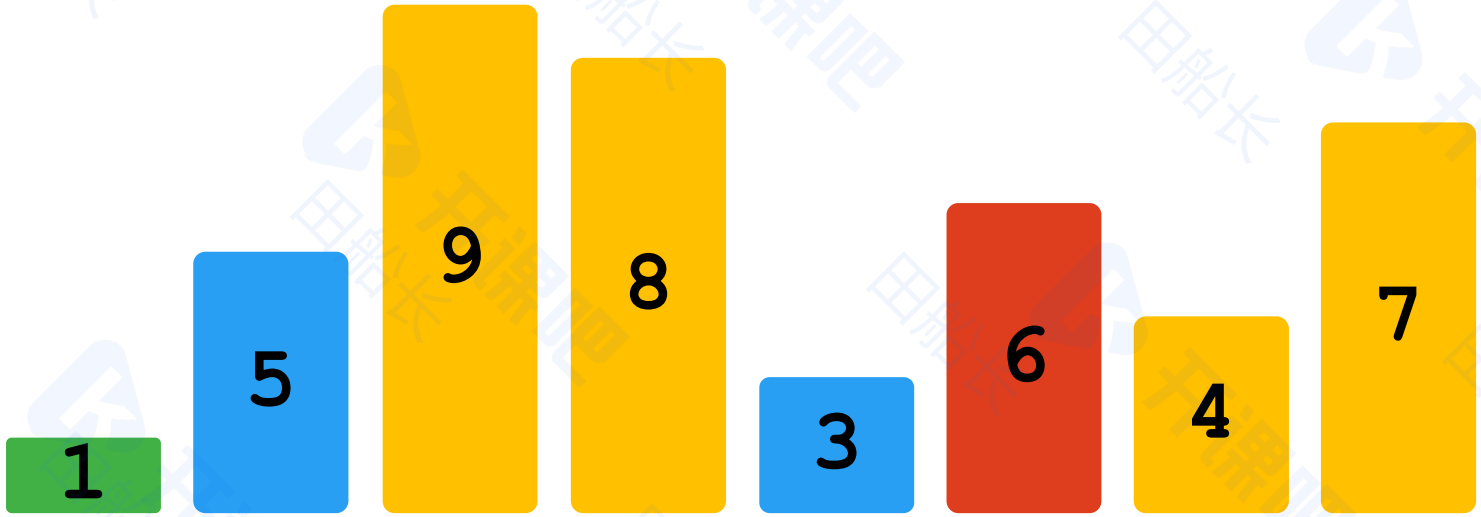
目前的最小值是5，
比较它与3



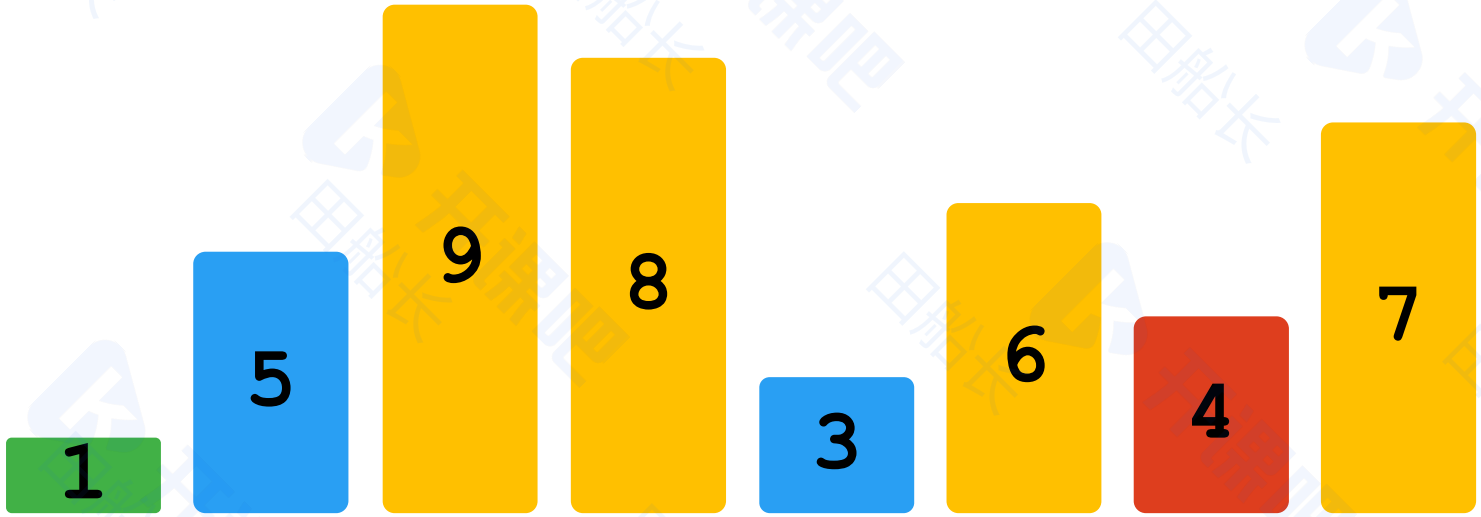
最小值更新为3



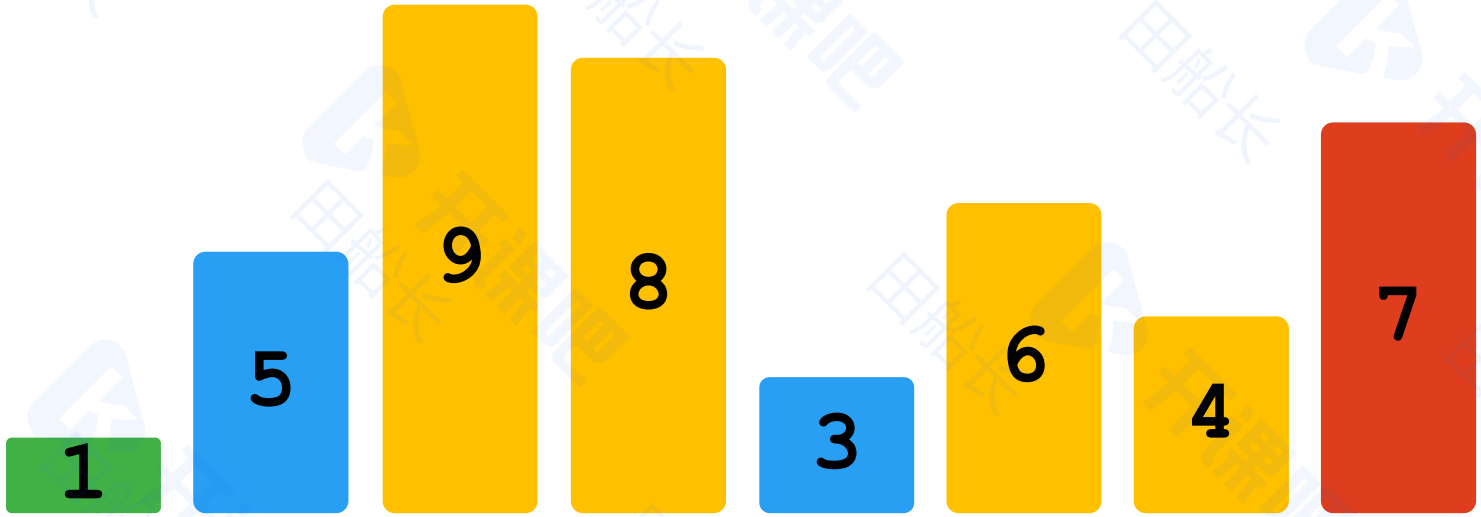
目前的最小值是3，
比较它与6



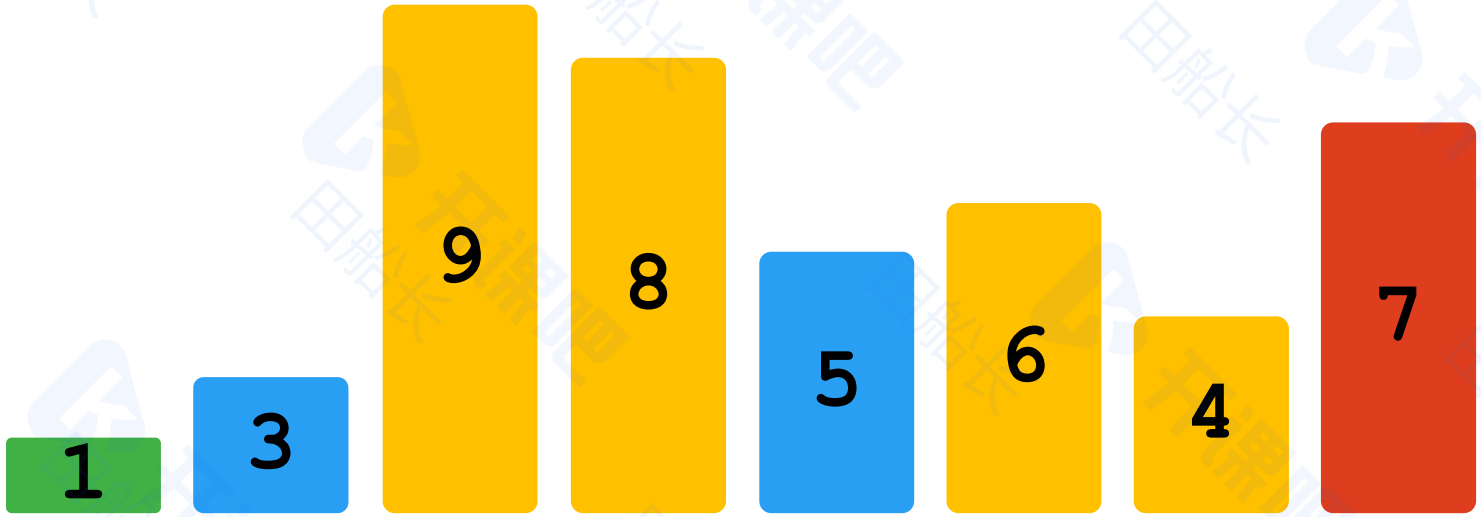
目前的最小值是3，
比较它与4



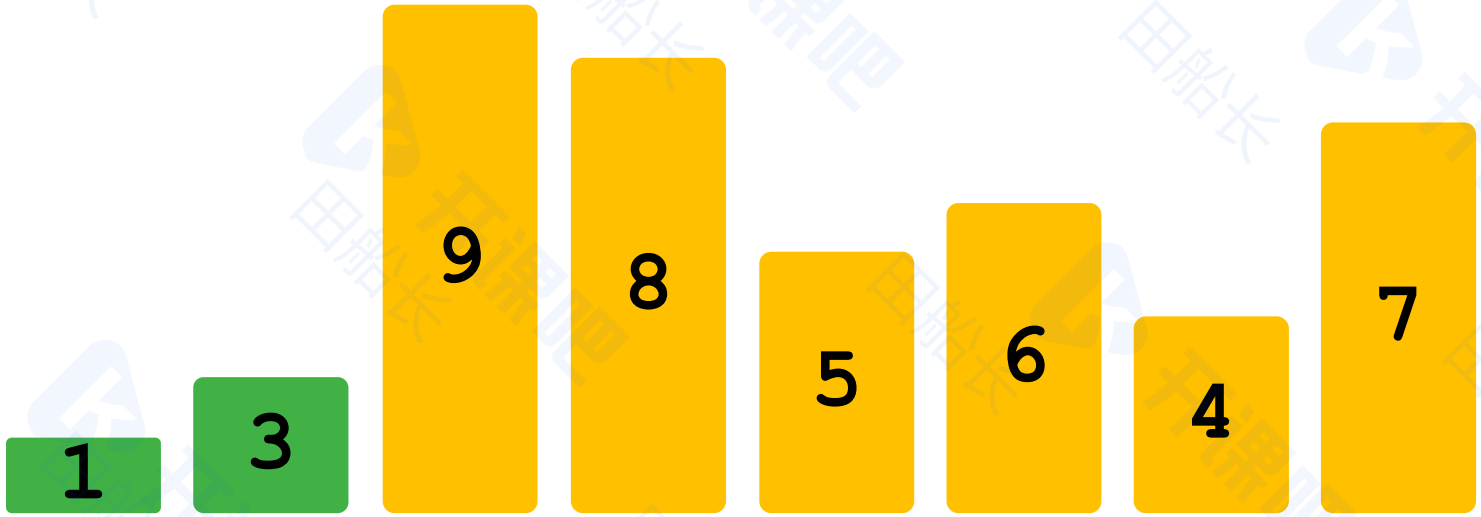
目前的最小值是3，
比较它与7



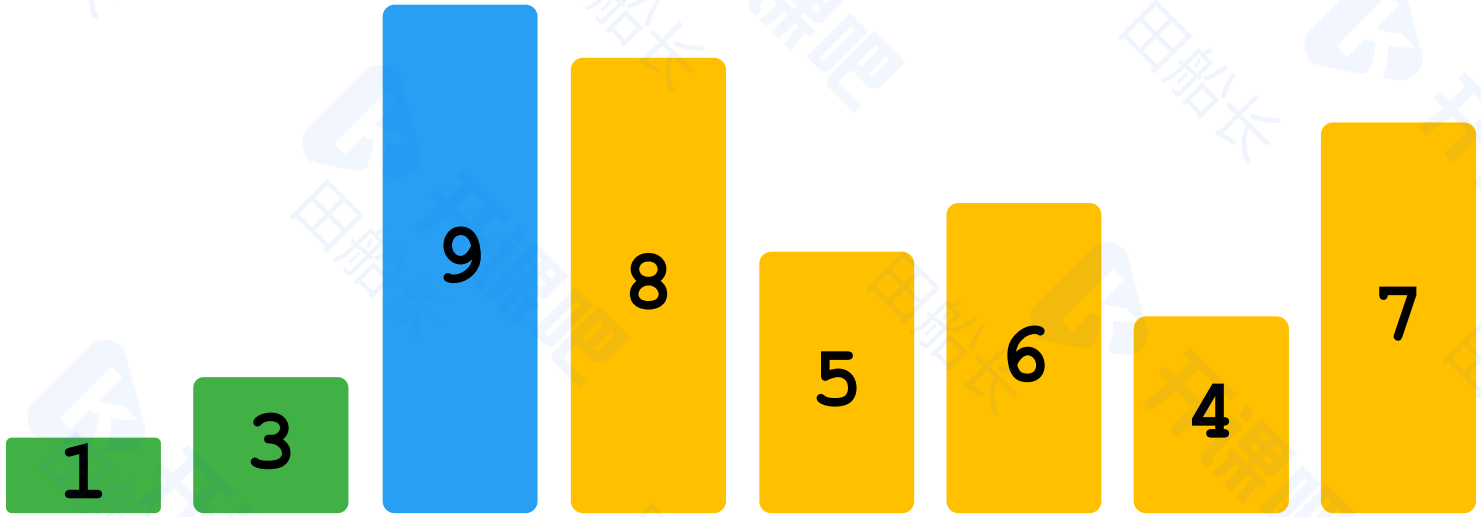
交换第1个数和其后面的最小值



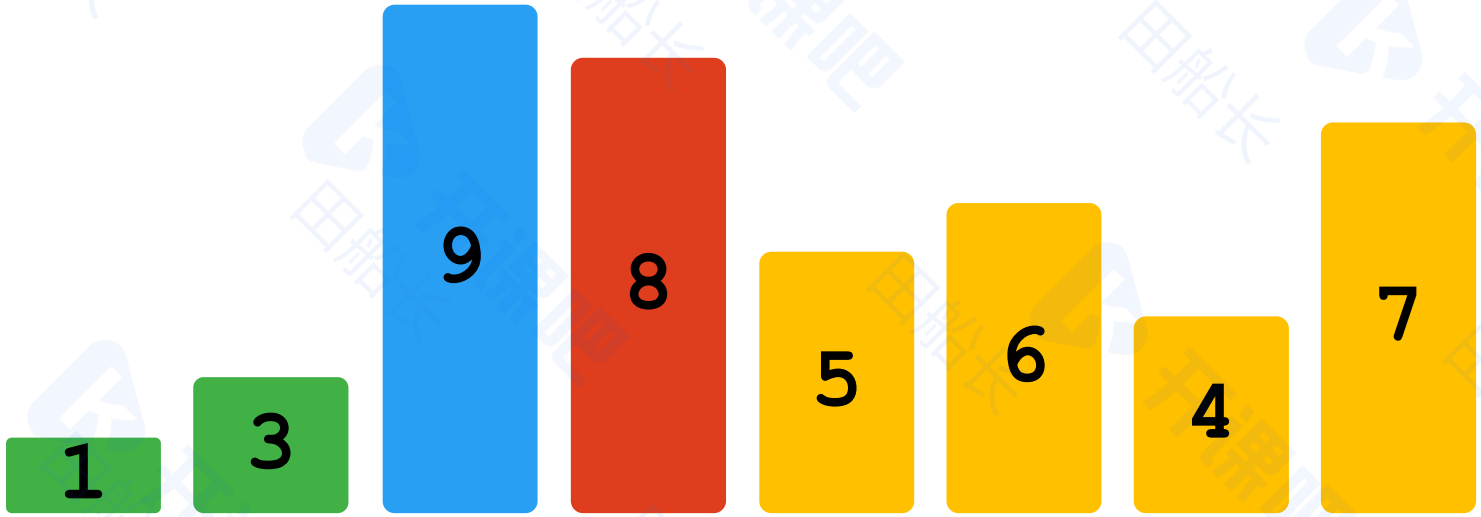
现在从第0个到第1个数已经在正确的位置上



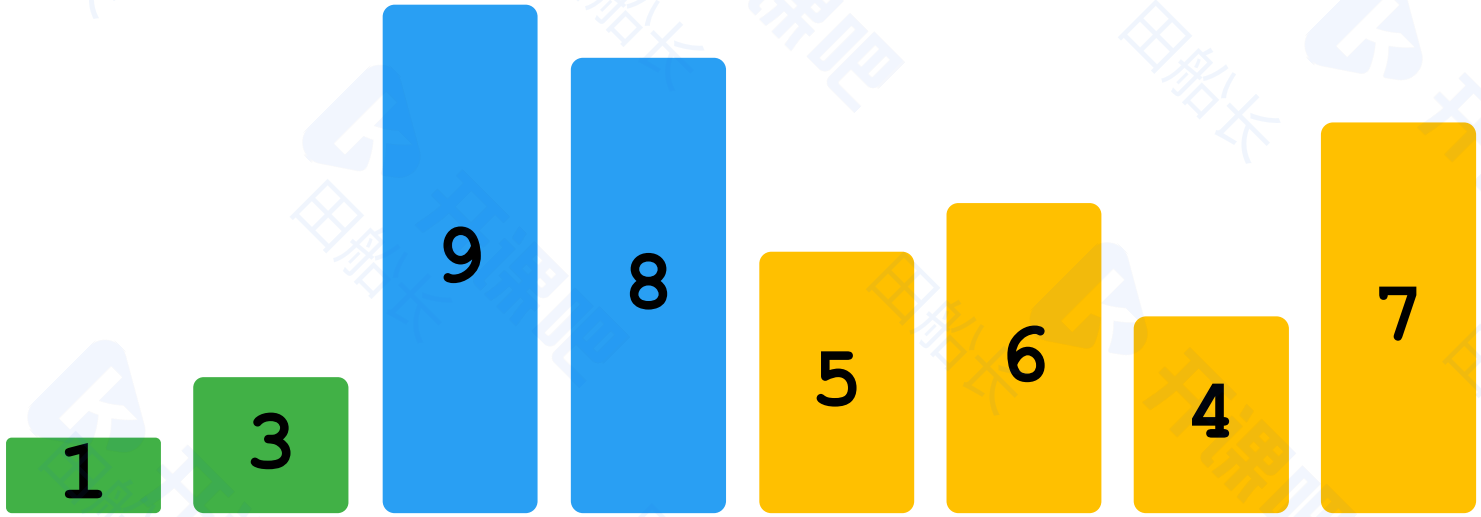
找出第2个及其之后的所有数字中的最小值



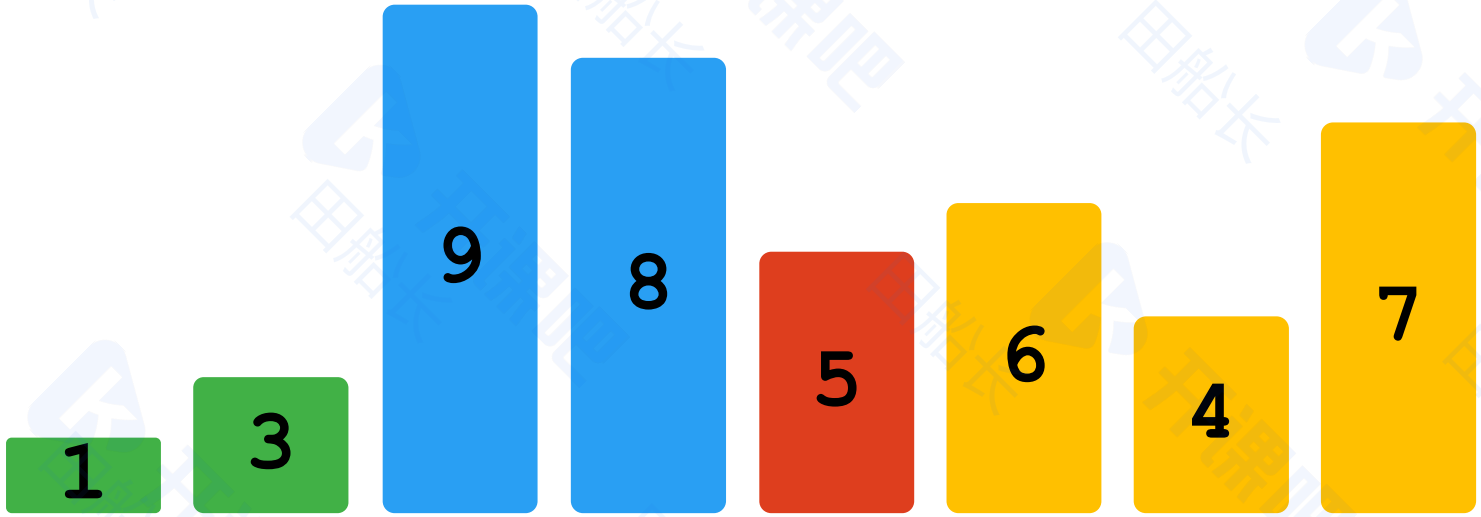
目前的最小值是9，
比较它与8



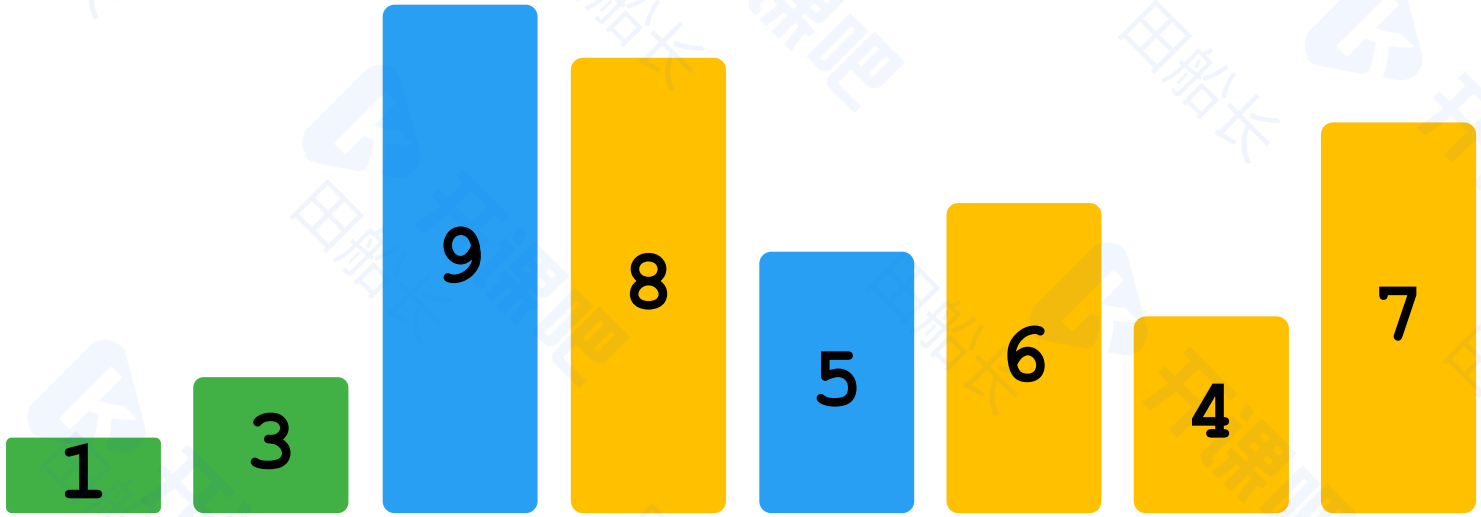
最小值更新为8



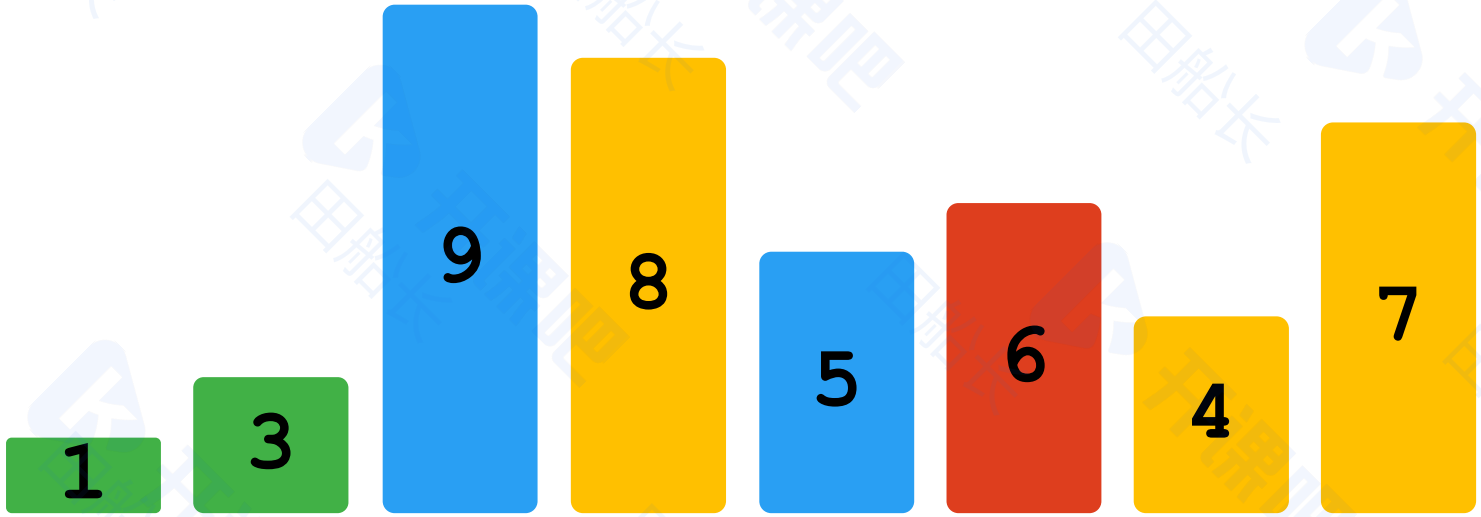
目前的最小值是8，
比较它与5



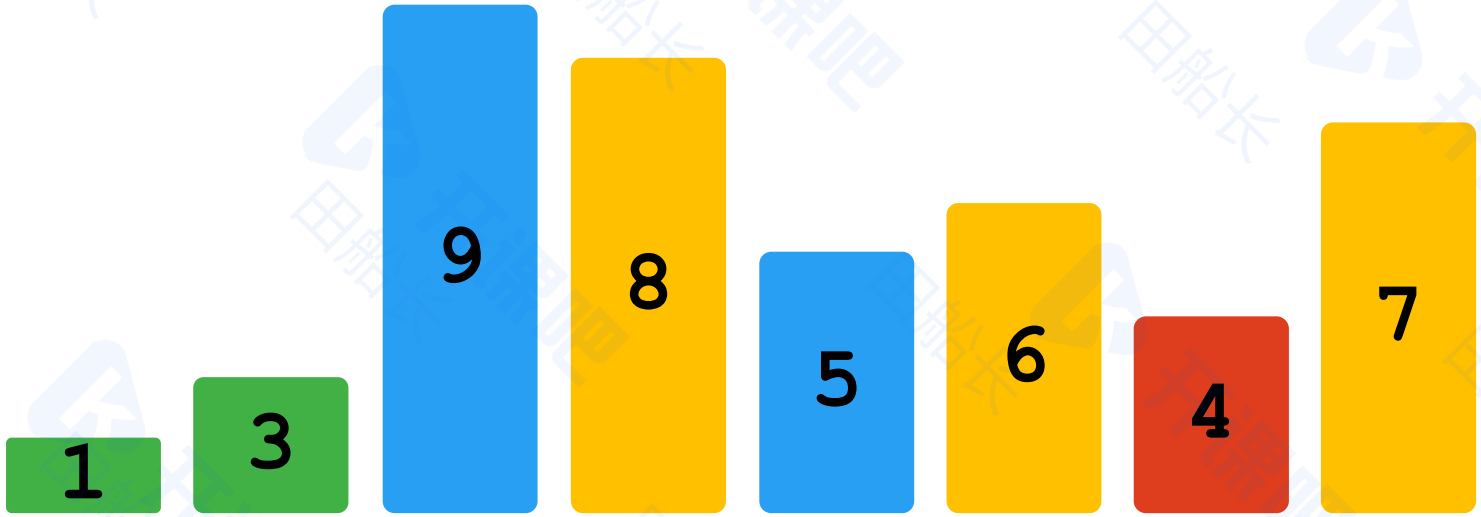
最小值更新为5



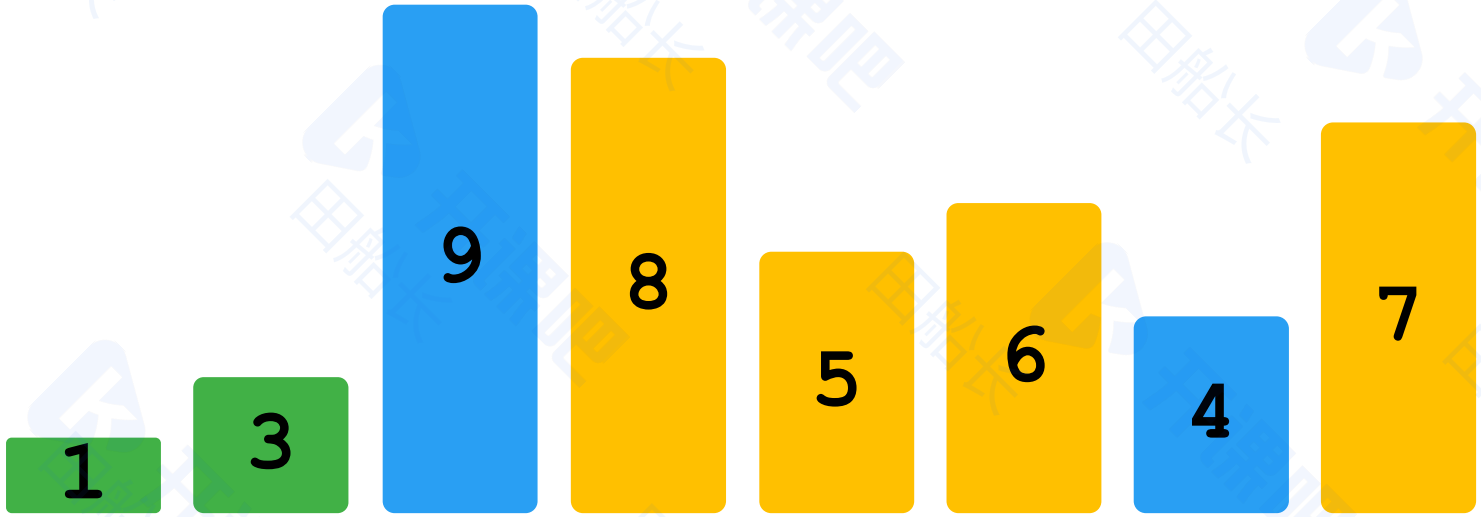
目前的最小值是5，
比较它与6



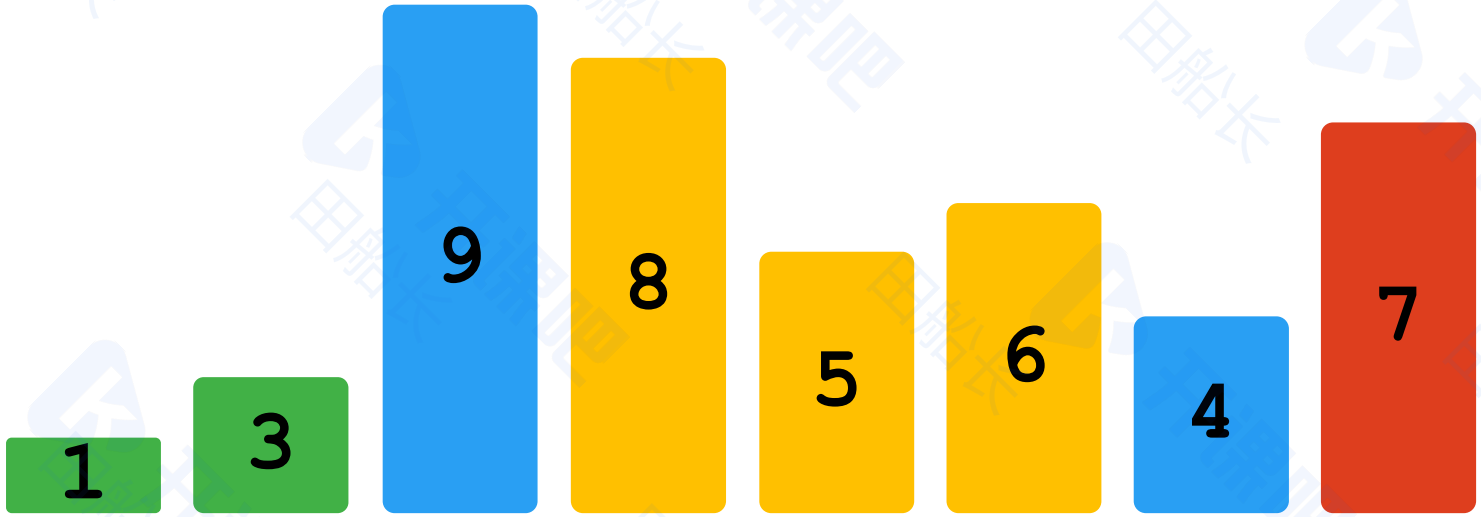
目前的最小值是5，
比较它与4



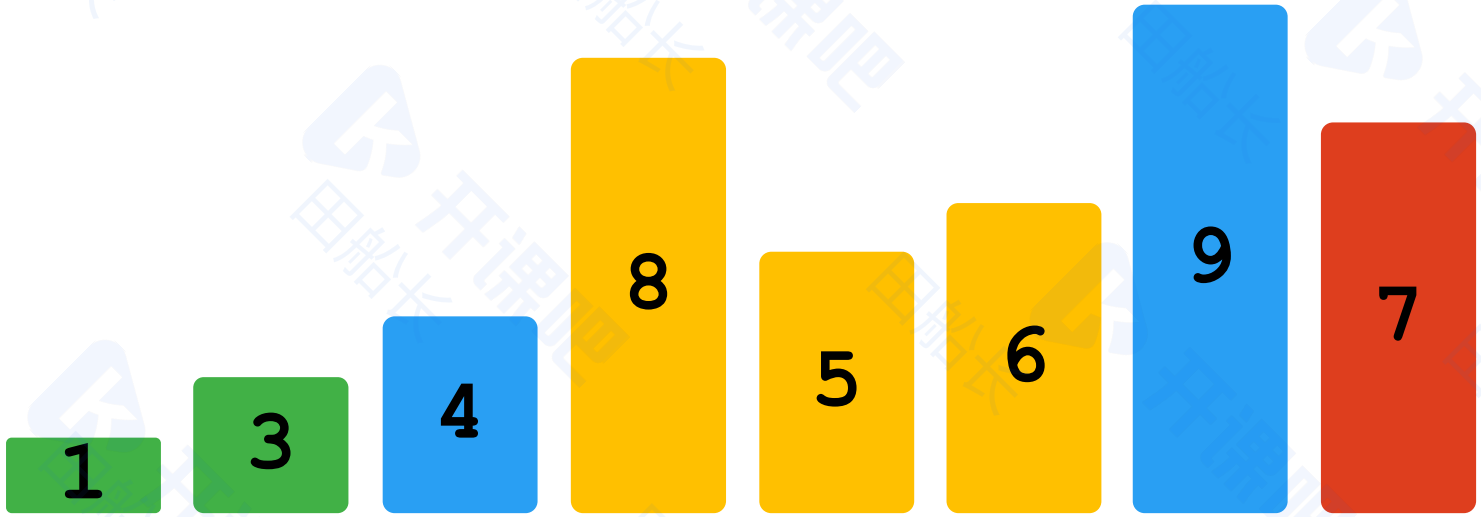
最小值更新为4



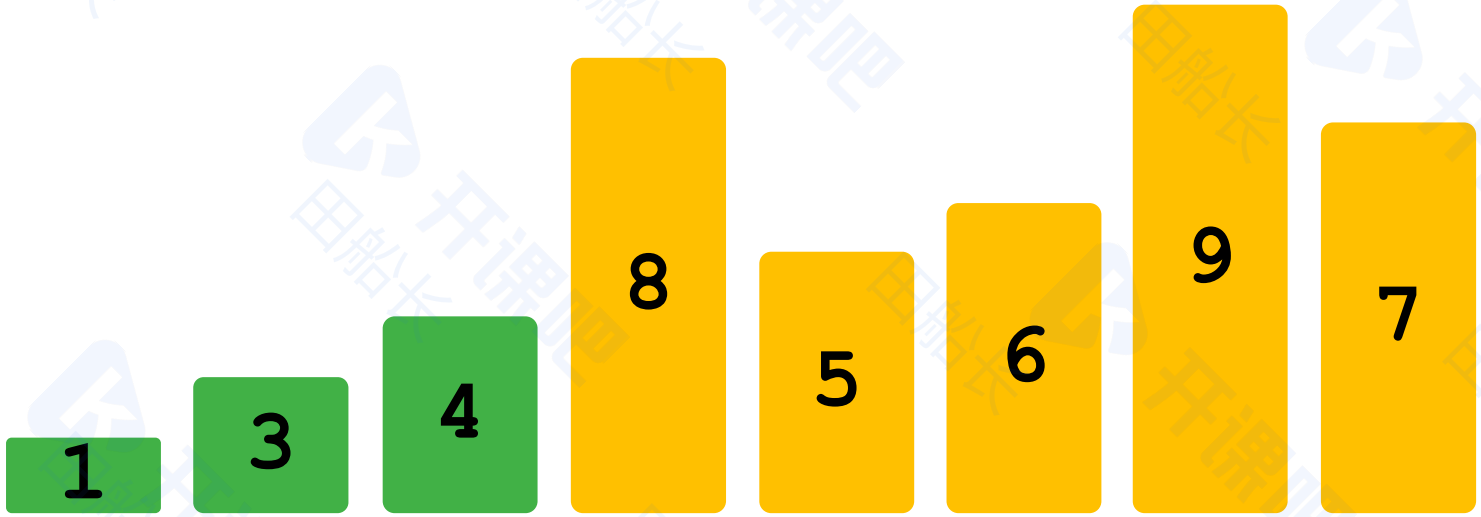
目前的最小值是4，
比较它与7



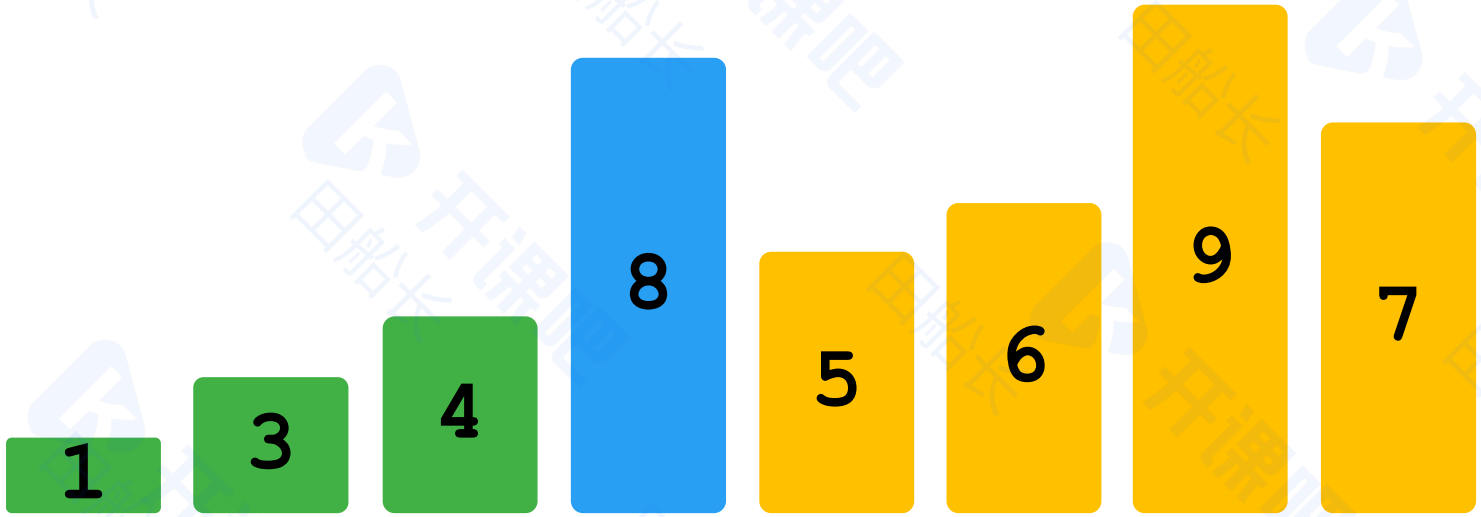
交换第2个数和其后面的最小值



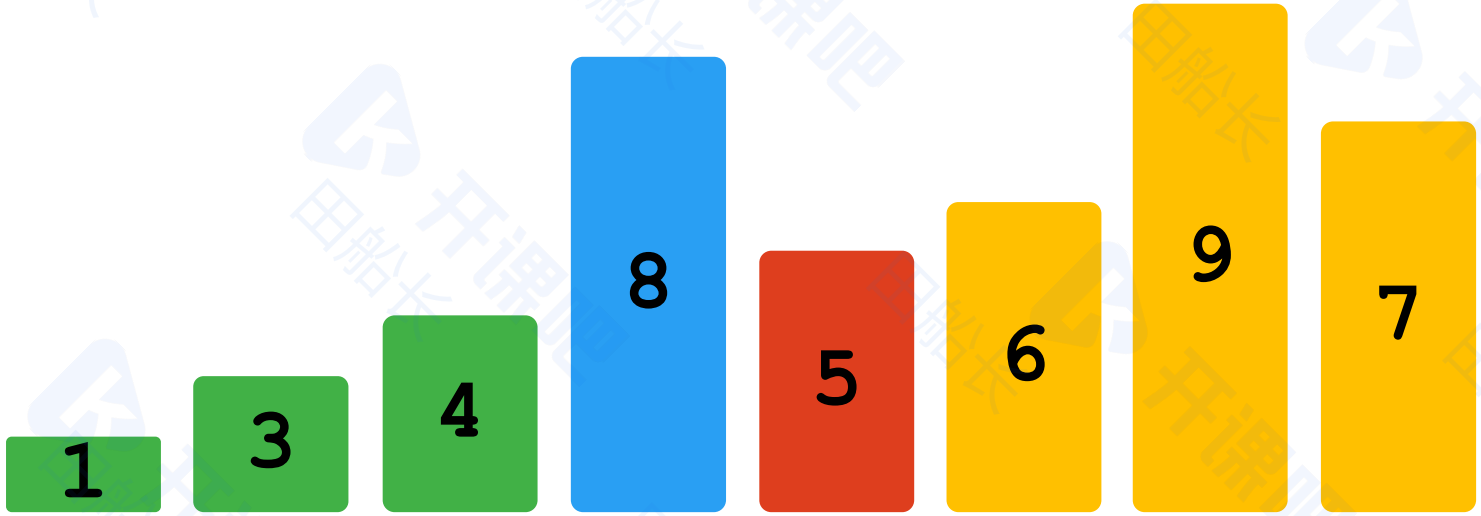
现在从第0个到第2个数已经在正确的位置上



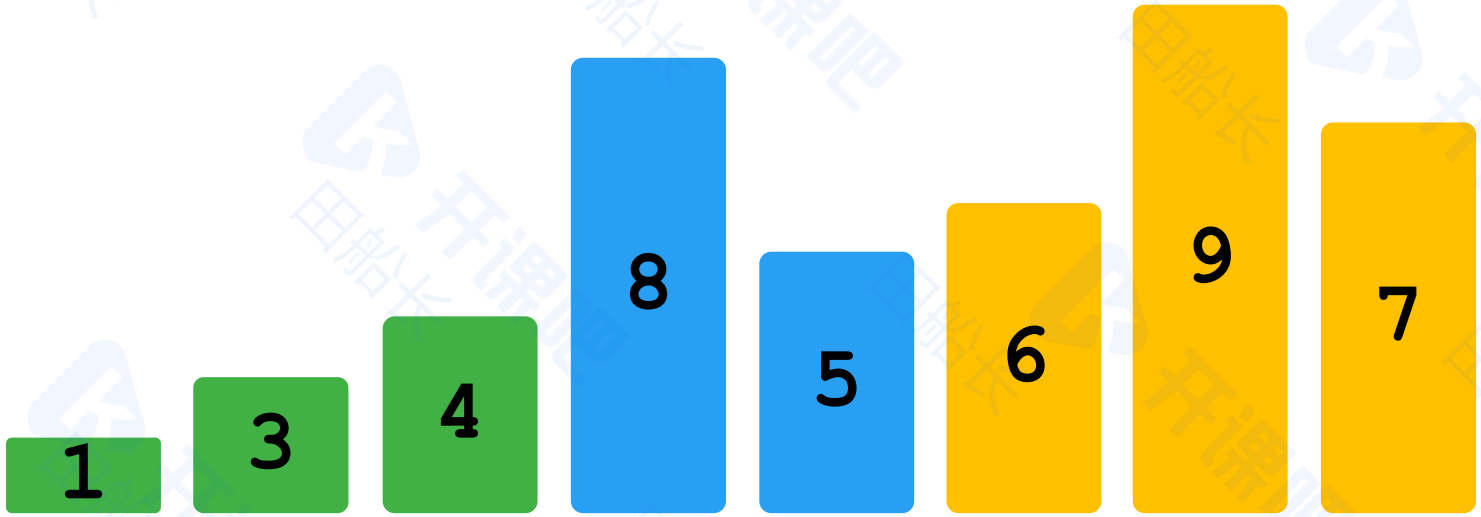
找出第3个及其之后的所有数字中的最小值



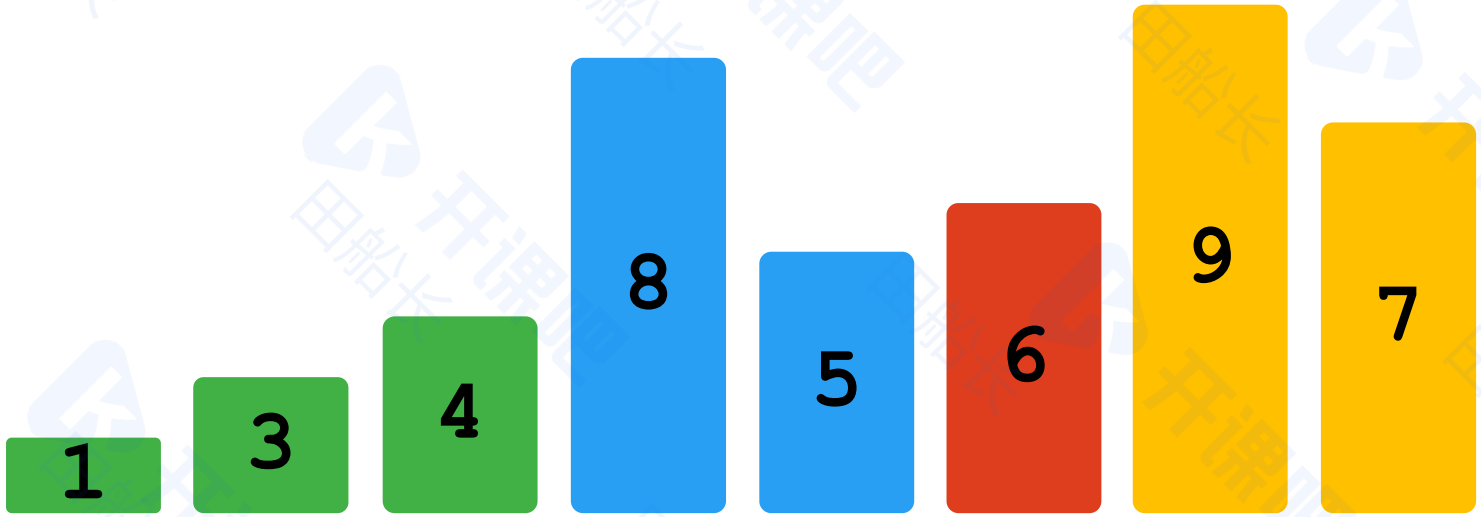
目前的最小值是8，
比较它与5



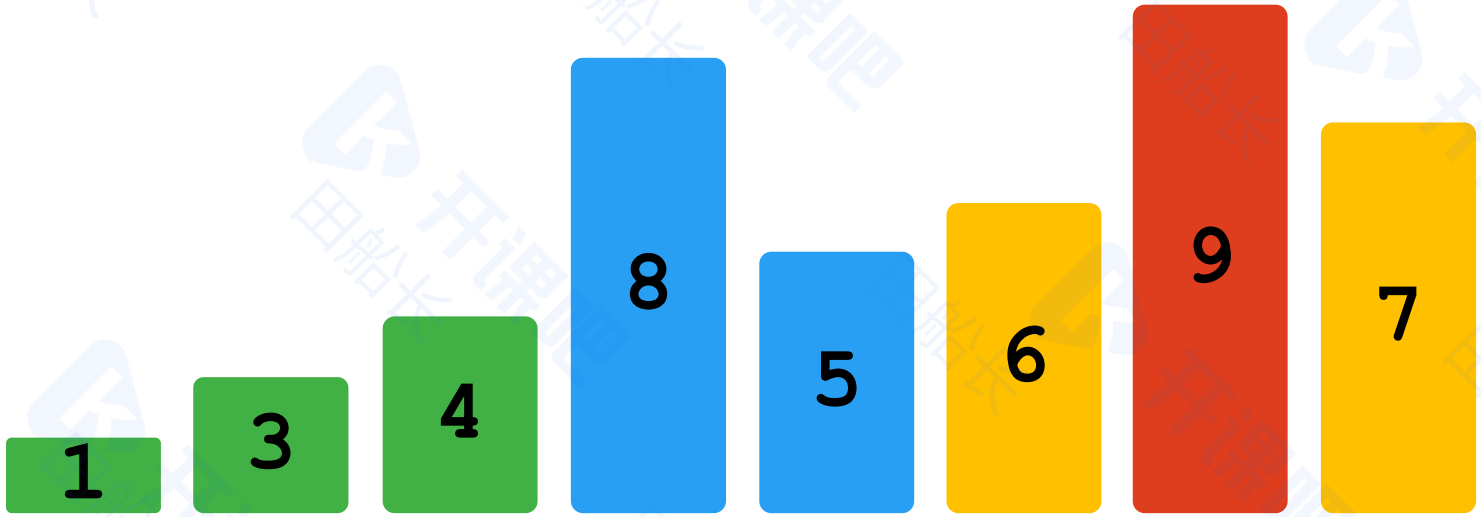
最小值更新为5



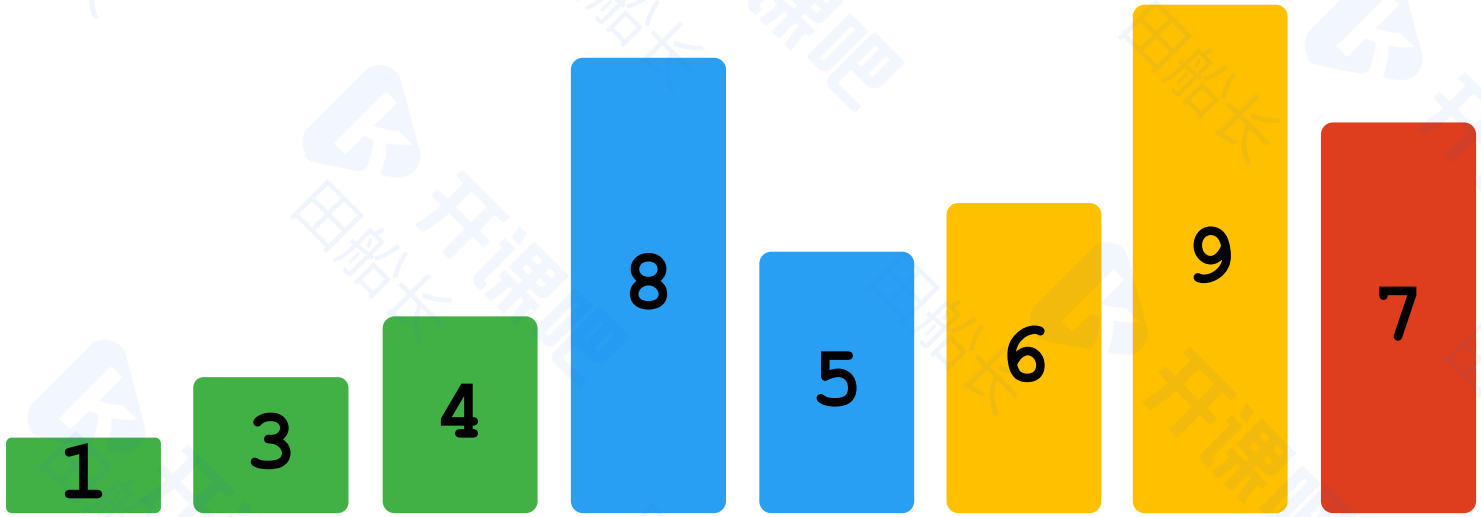
目前的最小值是5，
比较它与6



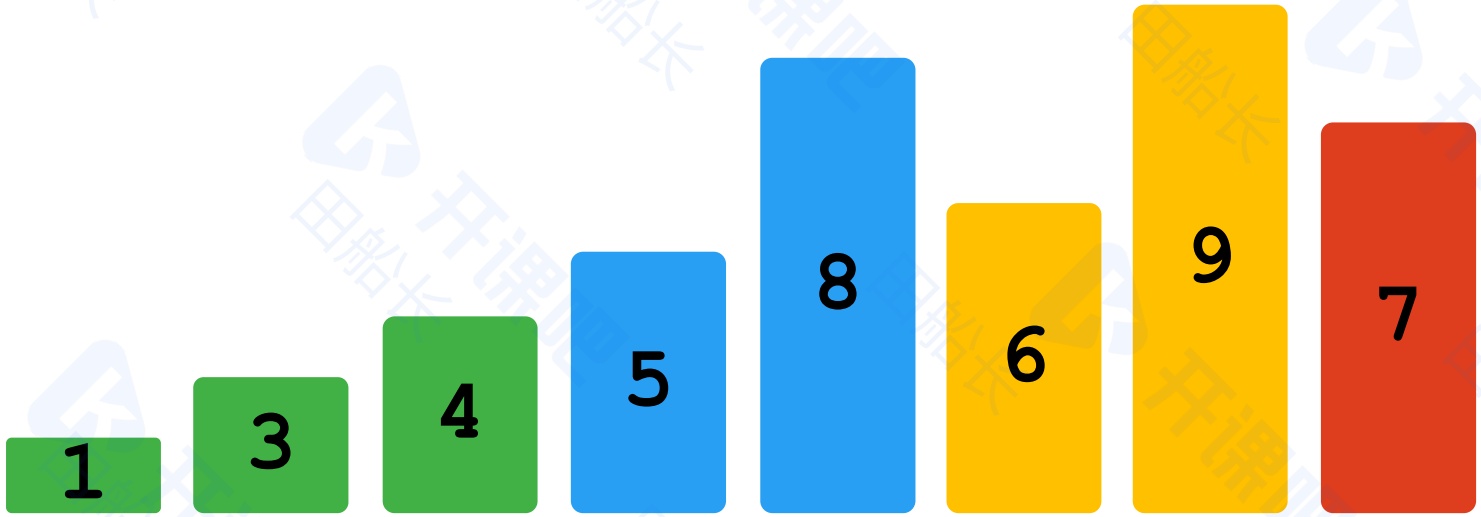
目前的最小值是5，
比较它与9



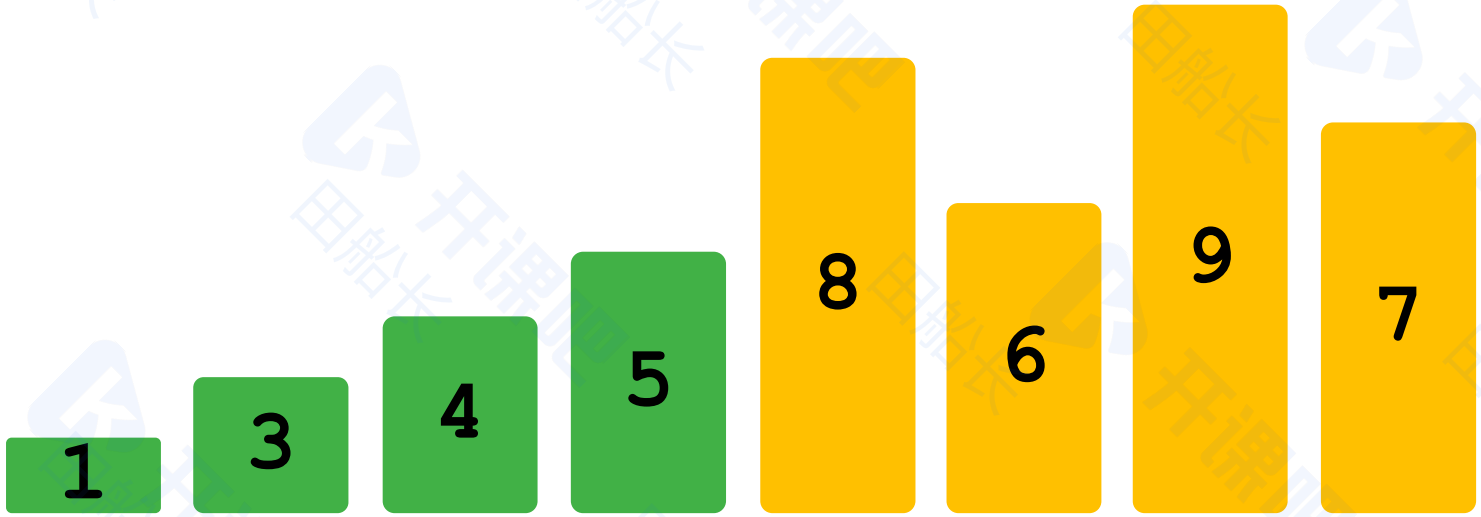
目前的最小值是5，
比较它与7



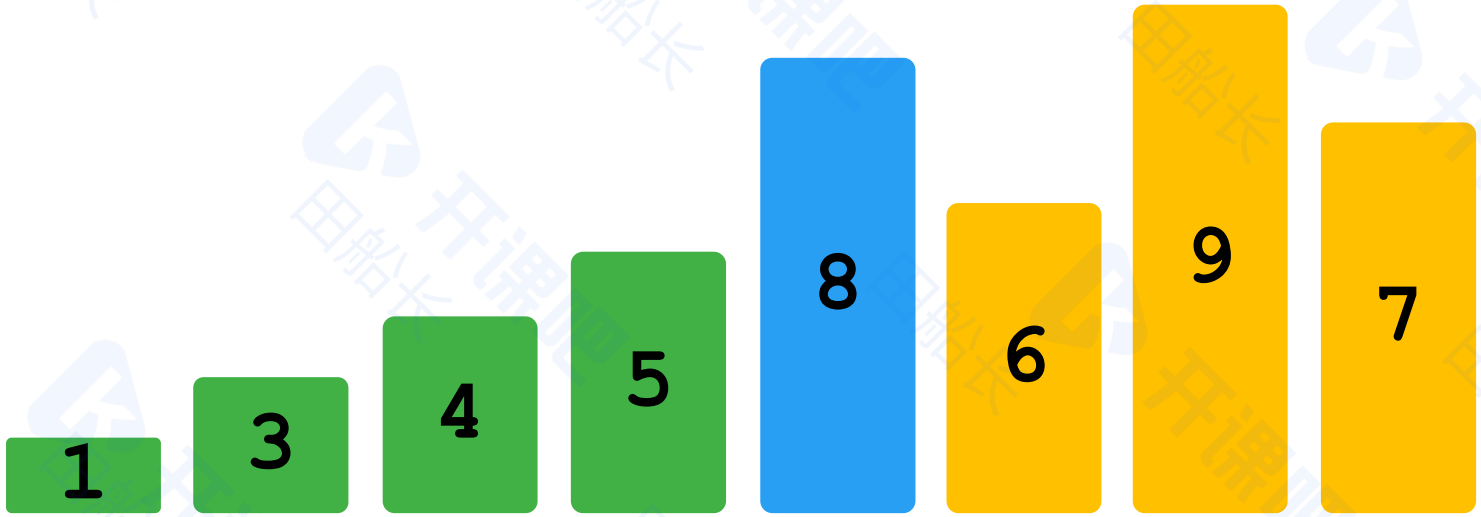
交换第3个数和其后面的最小值



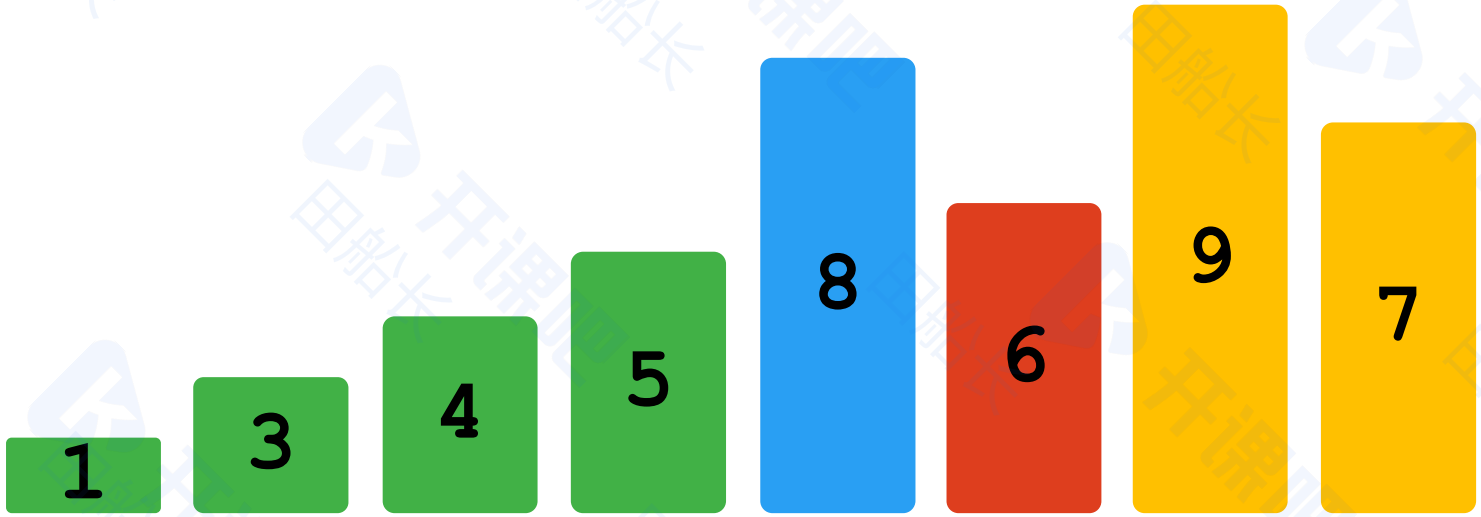
现在从第0个到第3个数已经在正确的位置上



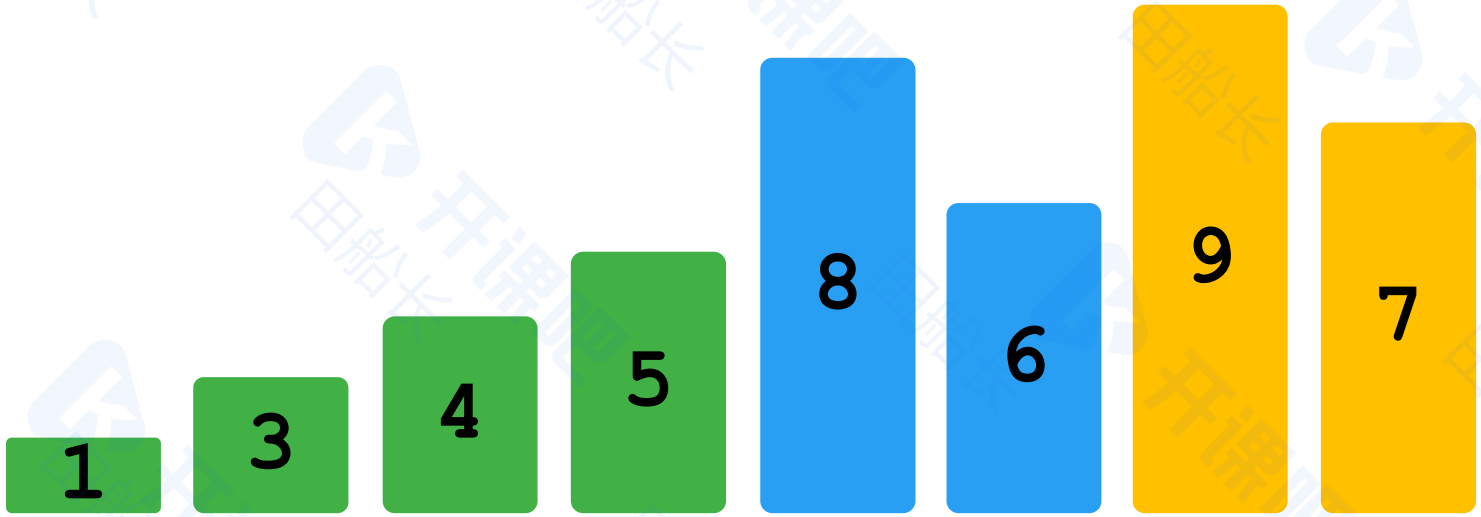
找出第4个及其之后的所有数字中的最小值



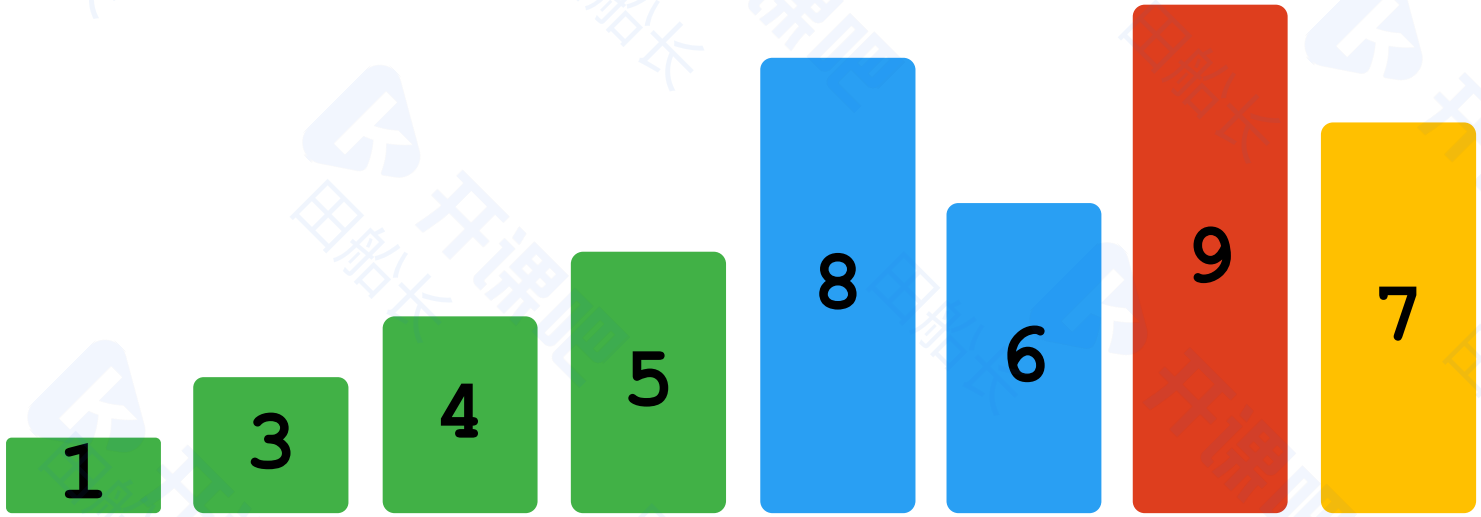
目前的最小值是8，
比较它与6



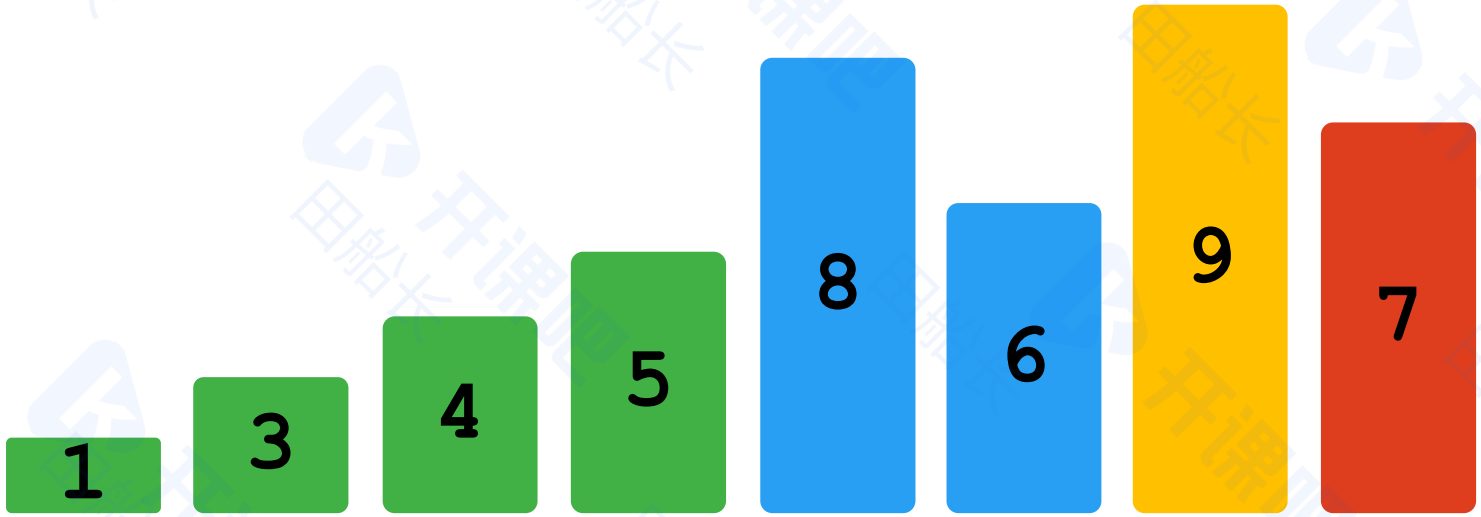
最小值更新为6



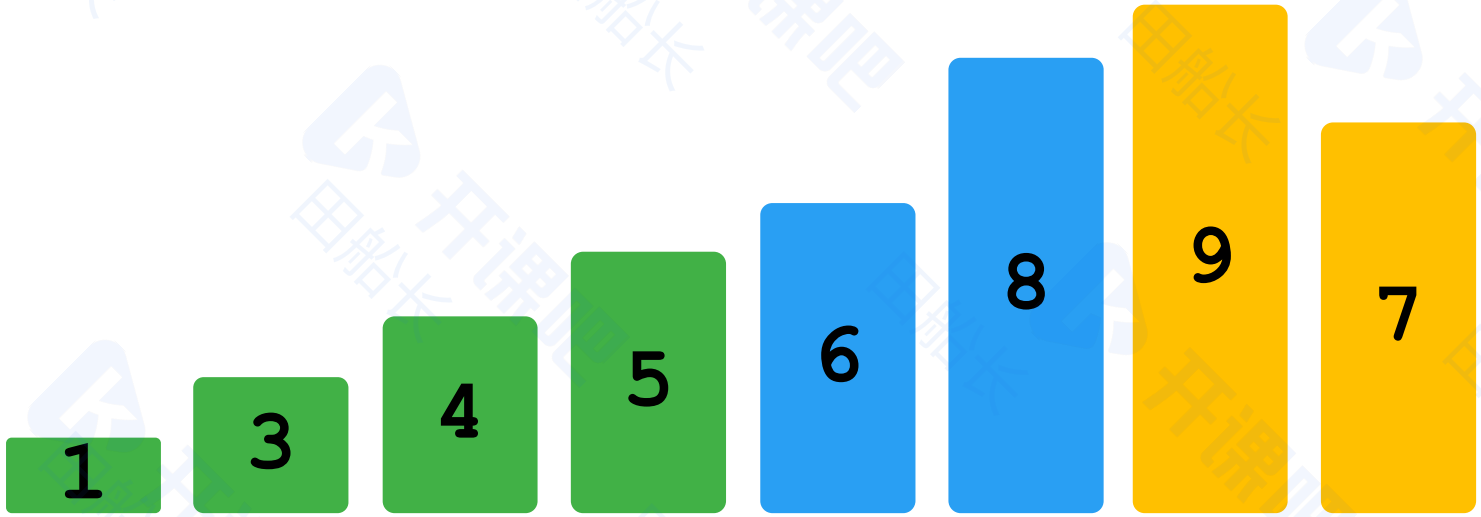
目前的最小值是6，
比较它与9



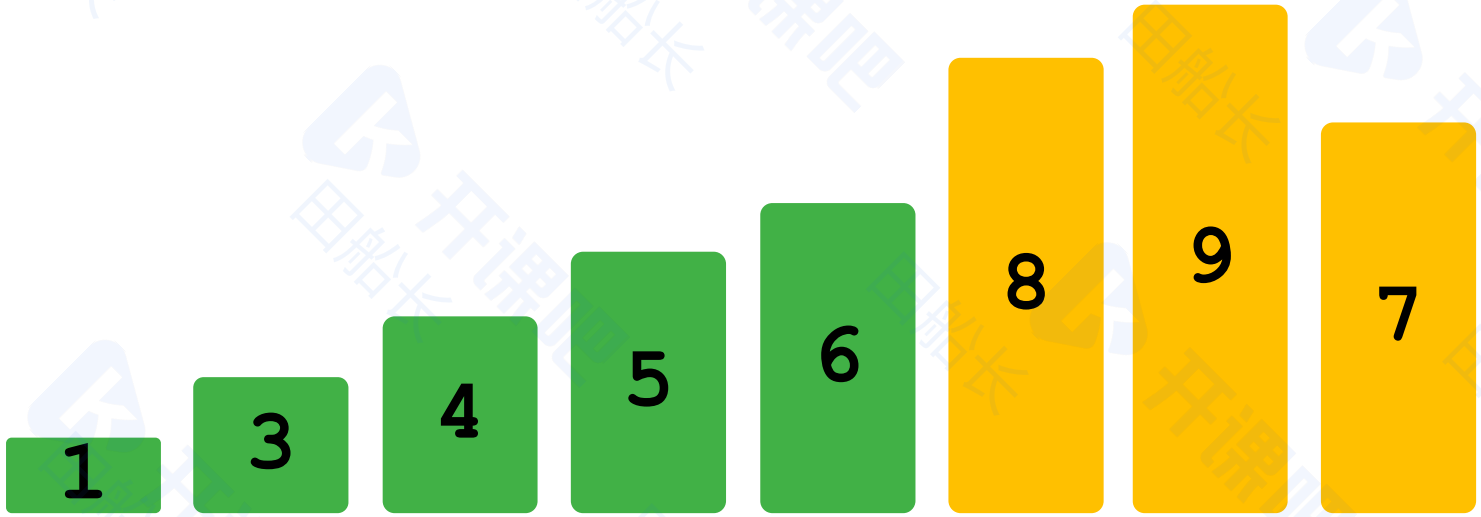
目前的最小值是6，
比较它与7



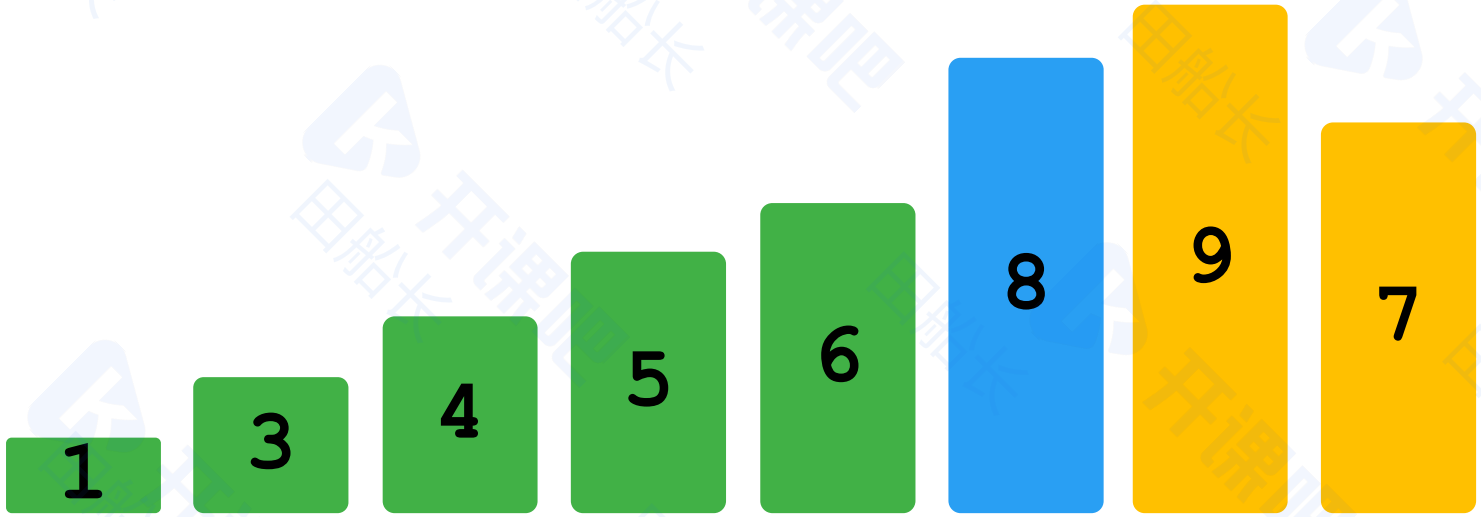
交换第4个数和其后面的最小值



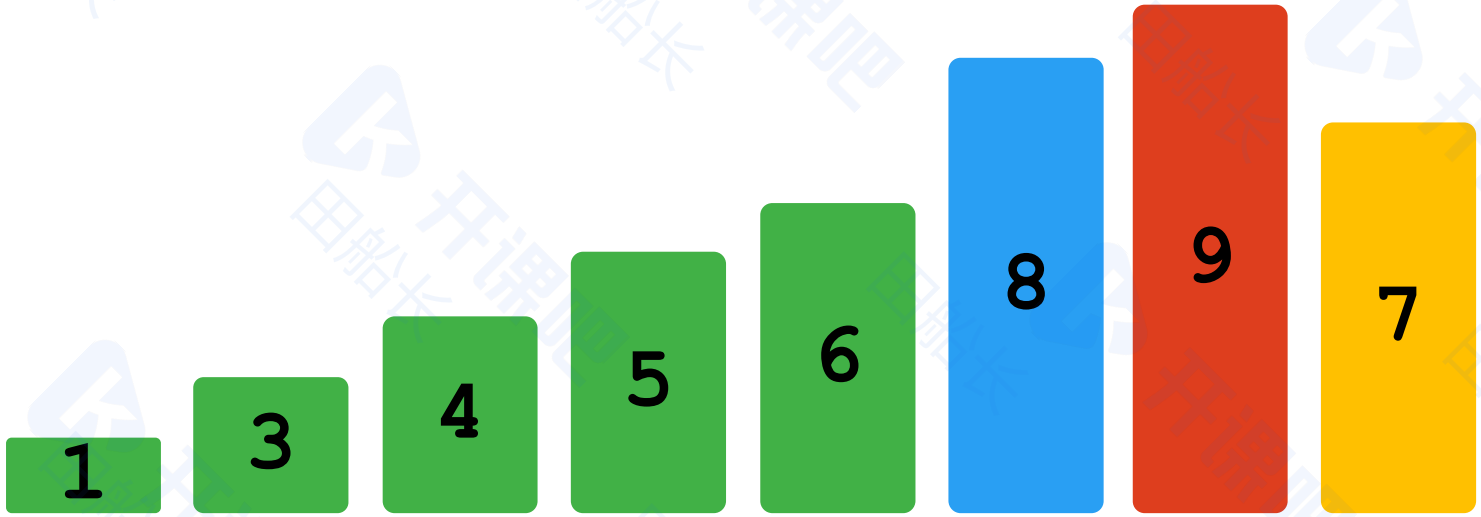
现在从第0个到第4个数已经在正确的位置上



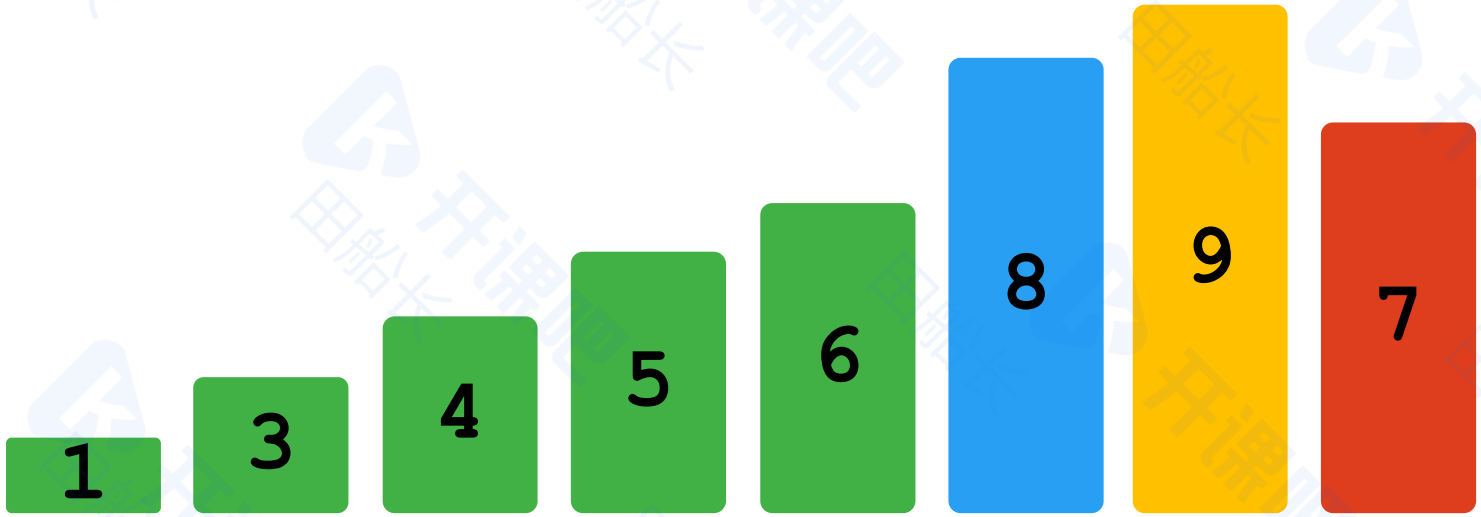
找出第5个及其之后的所有数字中的最小值



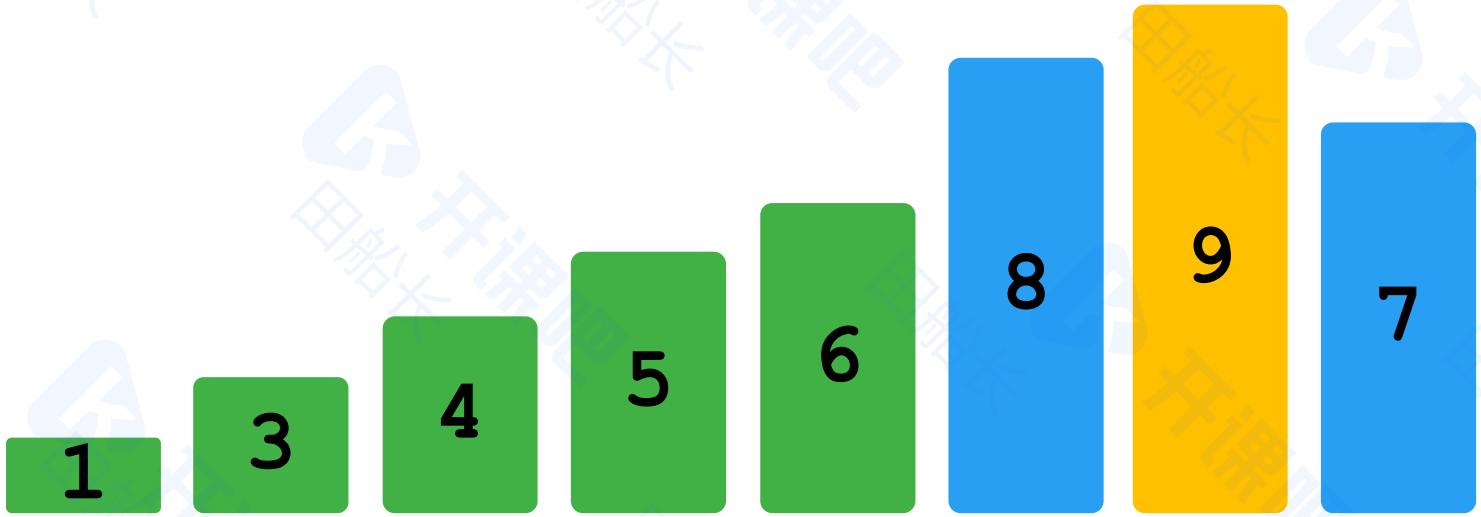
目前的最小值是8，
比较它与9



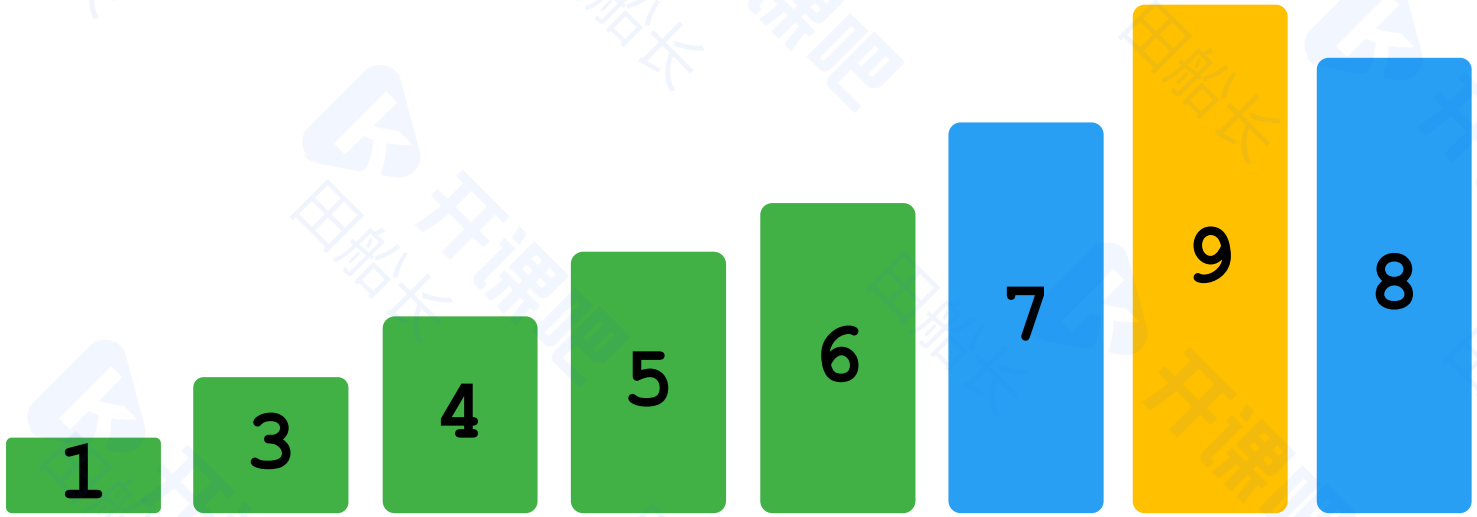
目前的最小值是8，
比较它与7



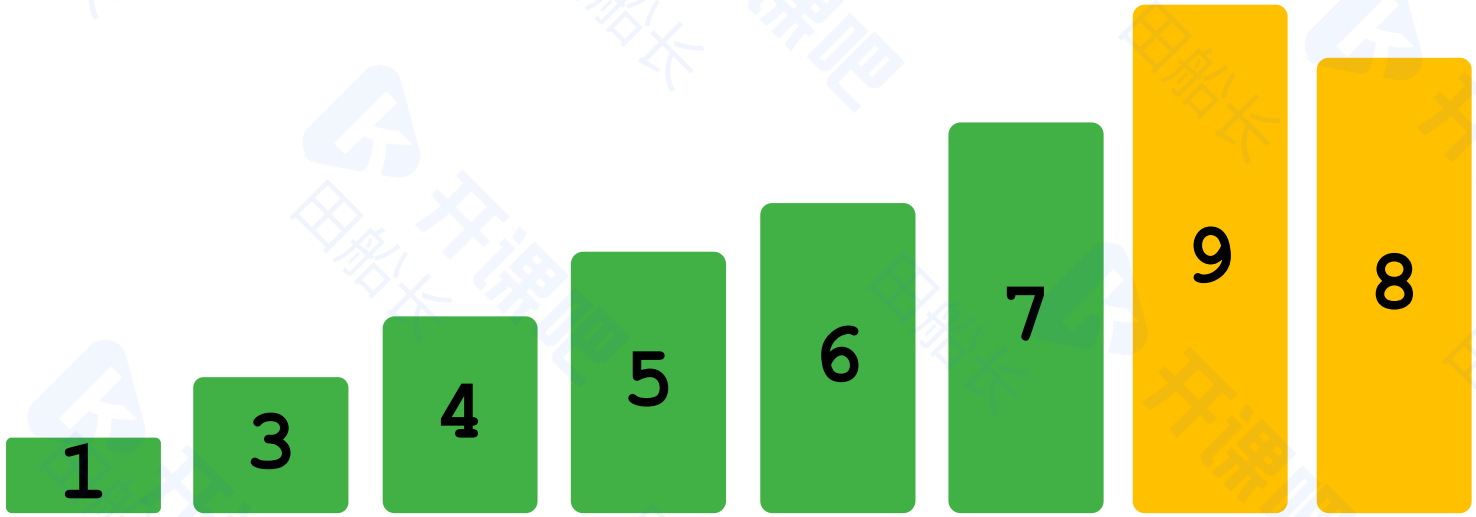
最小值更新为7



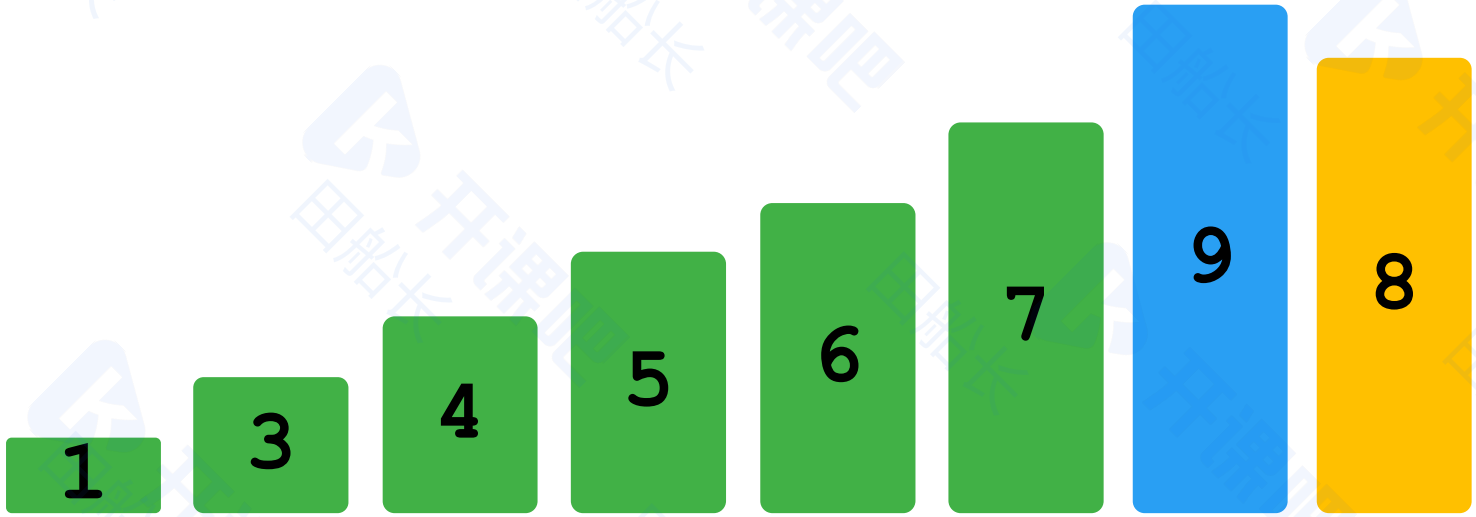
交换第5个数和其后面的最小值



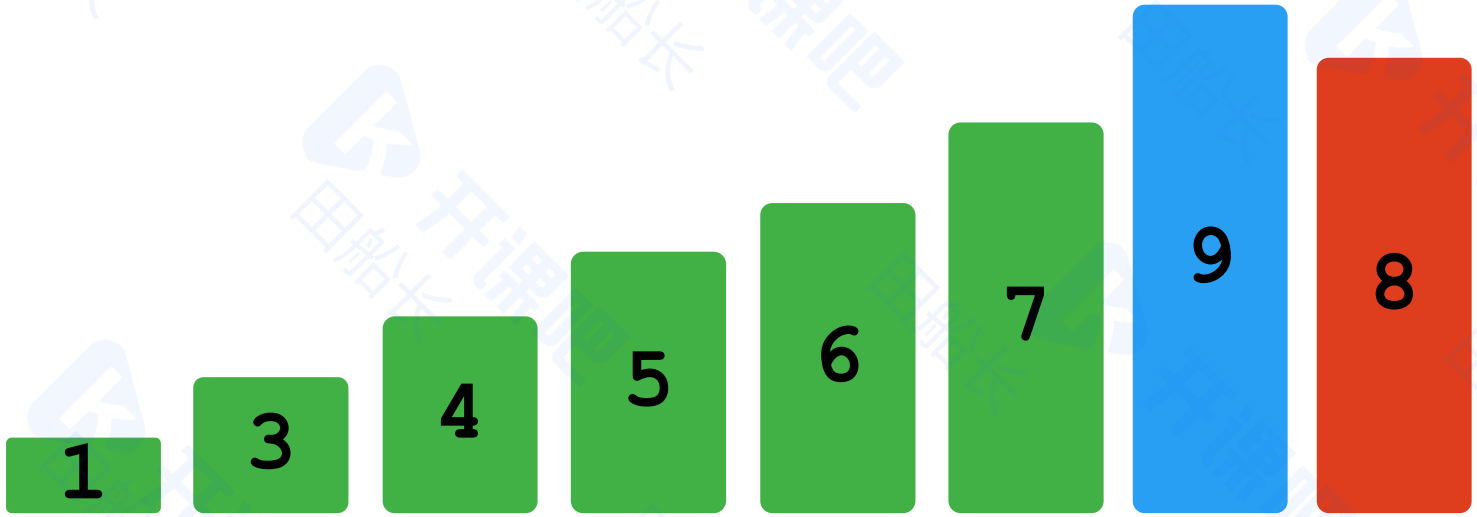
现在从第0个到第5个数已经在正确的位置上



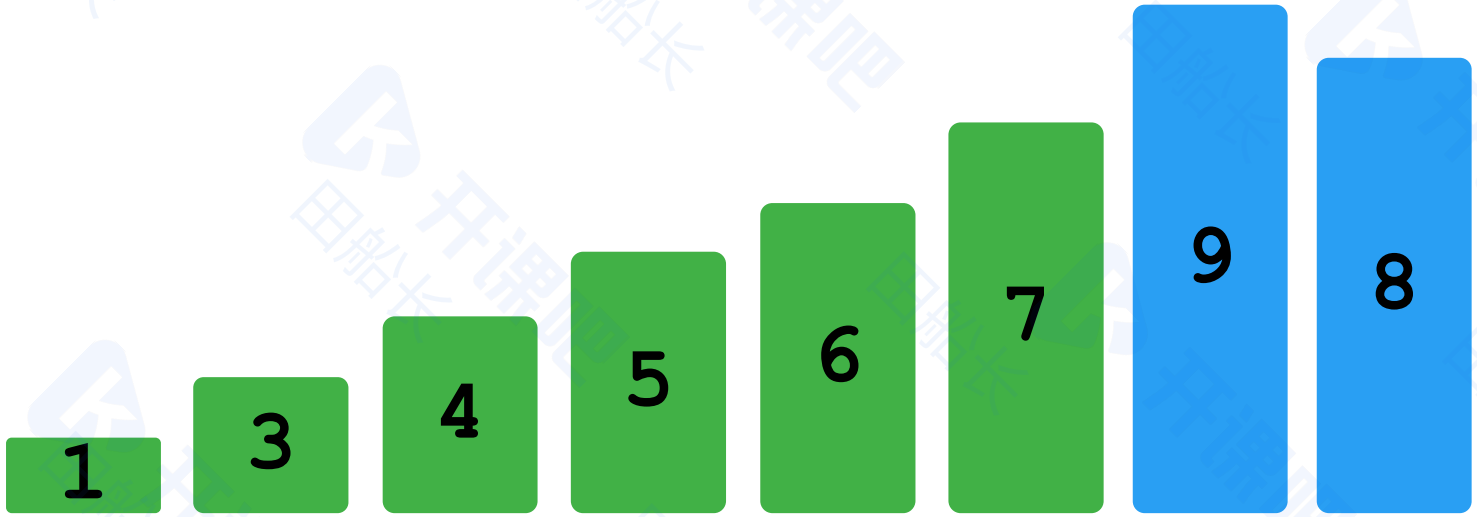
找出第6个及其之后的所有数字中的最小值



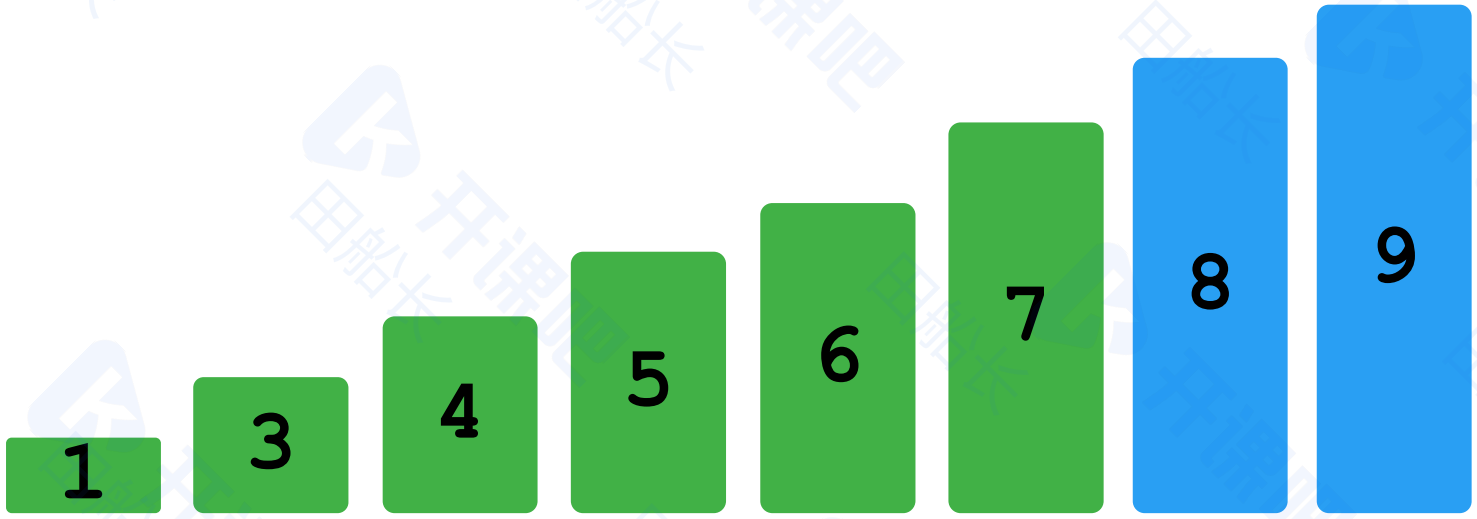
目前的最小值是9，
比较它与8



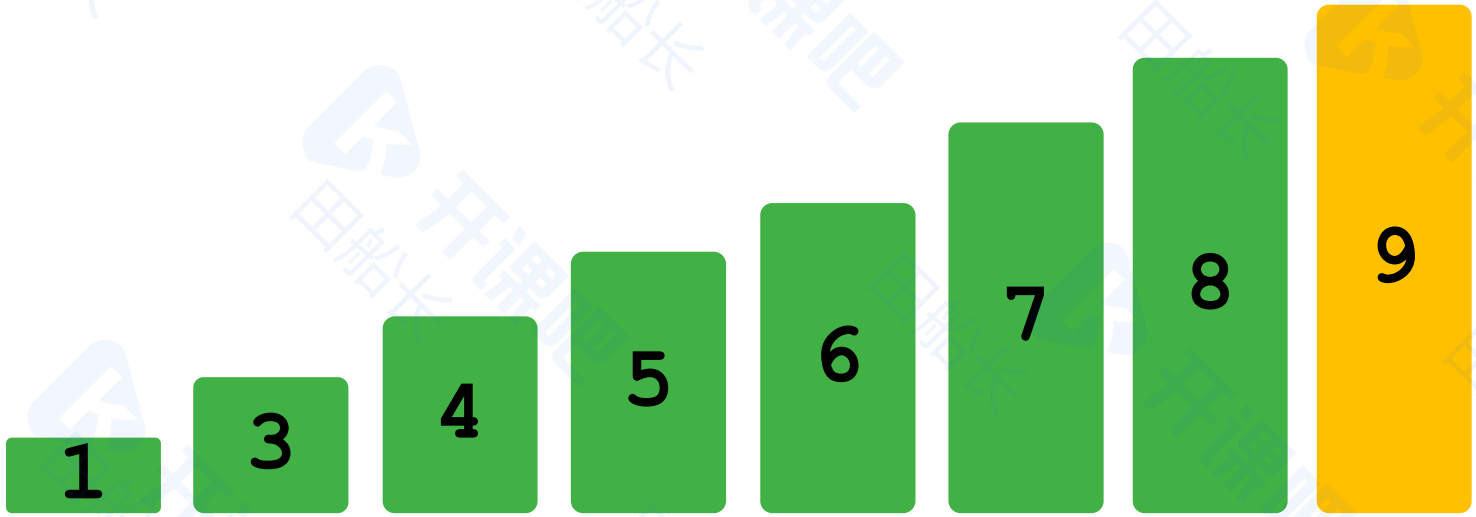
最小值更新为8



交换第6个数和其后面的最小值



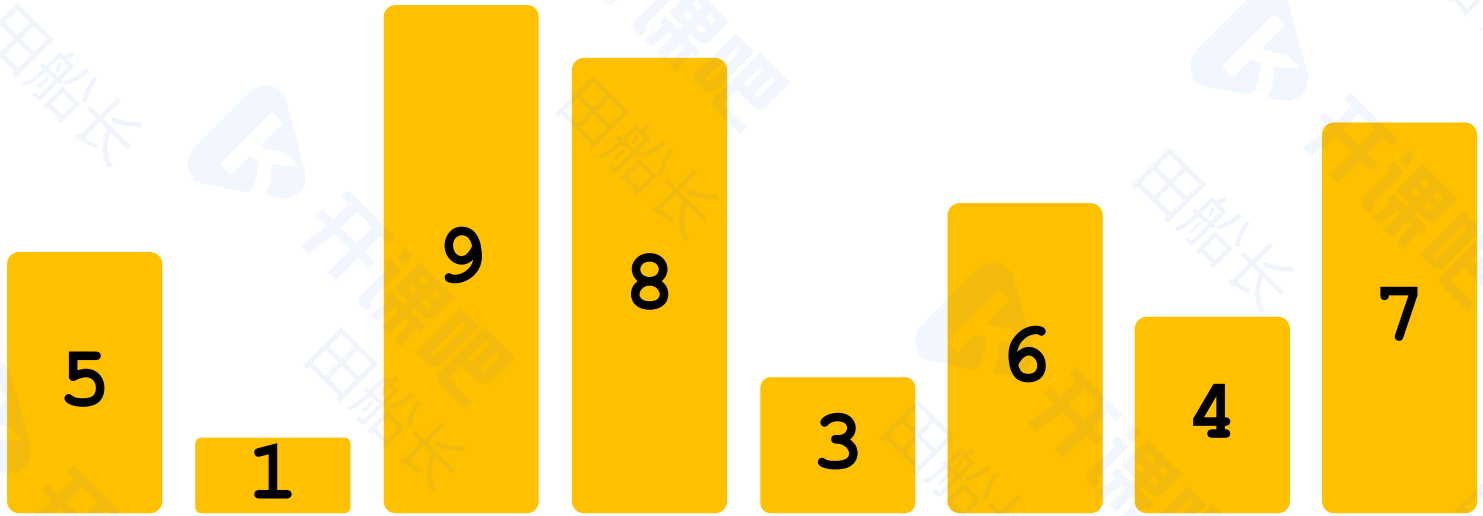
现在从第0个到第6个数已经在正确的位置上

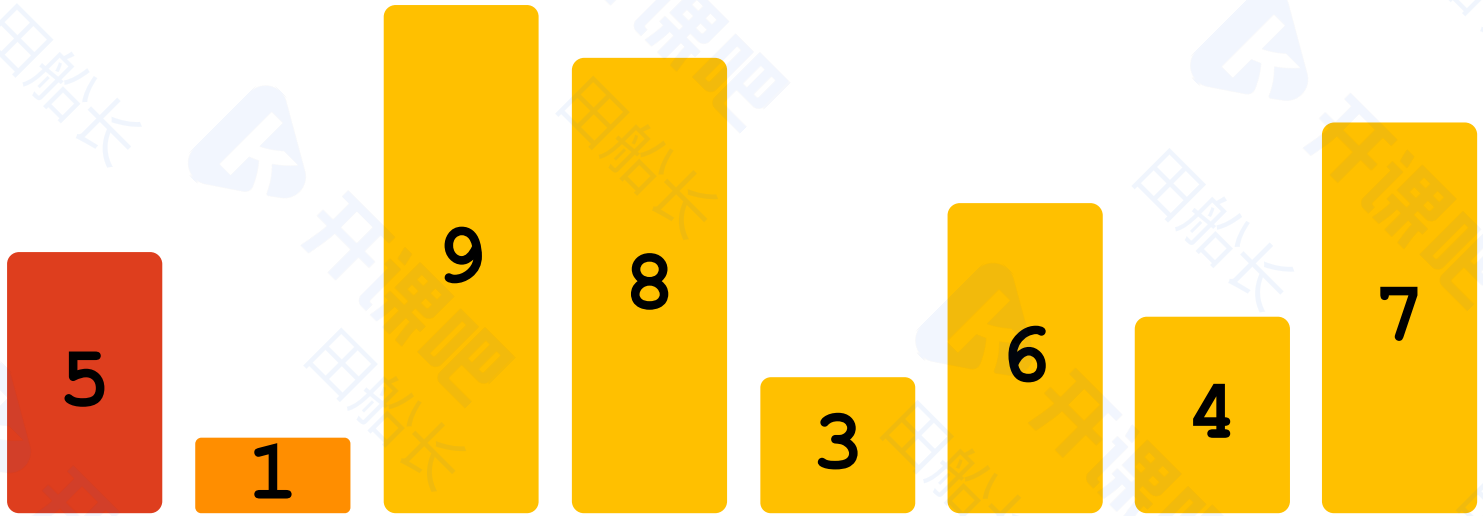


选择排序结束

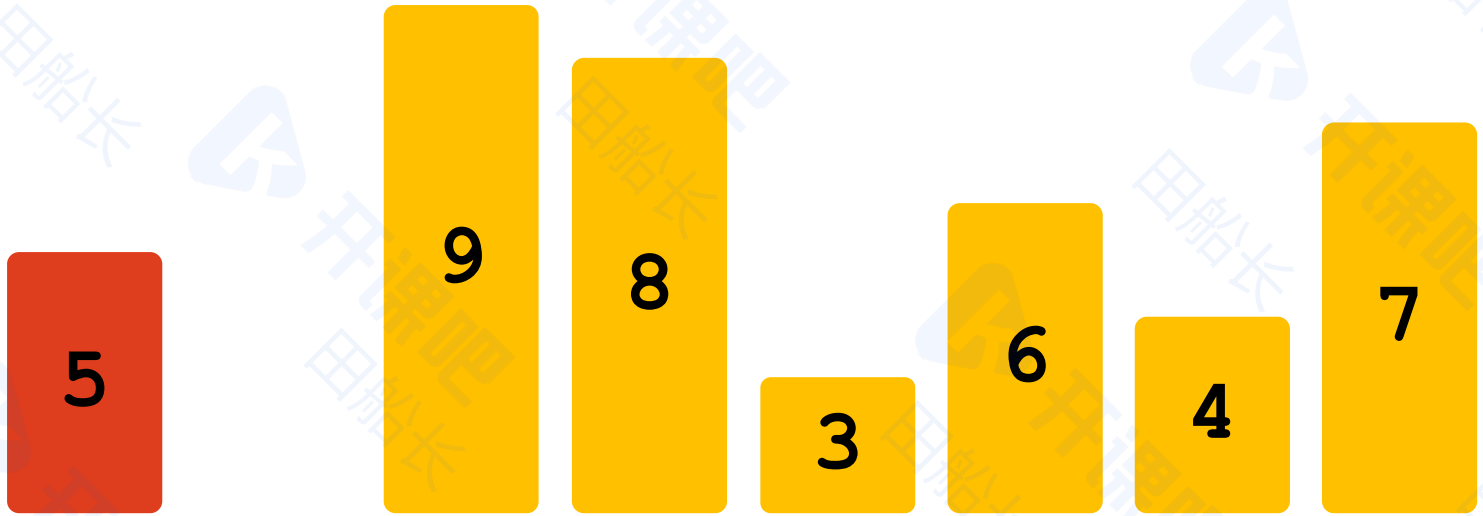
堆（优先队列）是一个用数组模拟的完全二叉树
在堆排序时，分为建堆与排序两个部分
其中自下而上建堆的时间复杂度为 O(N)
排序时相当于不断删除堆顶元素（极值元素）
放到待排序序列最后面
最终使得序列有序

将待排序区间分为左区间和右区间
对两个区间分别进行归并排序
在两个区间有序后
对两个区间进行合并
最终使得整体有序



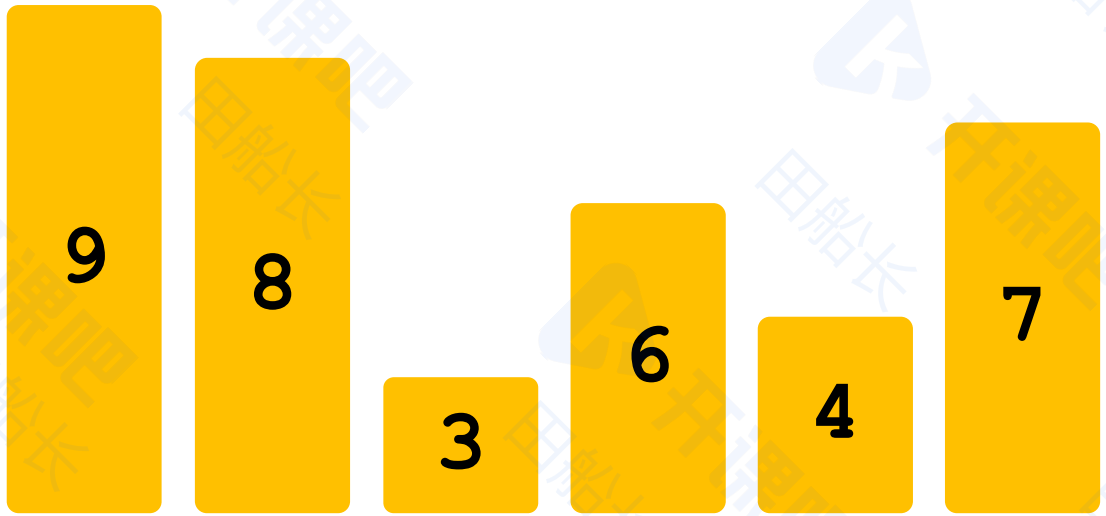


对两个区间[0，1)和[1，2)内的数进行归并

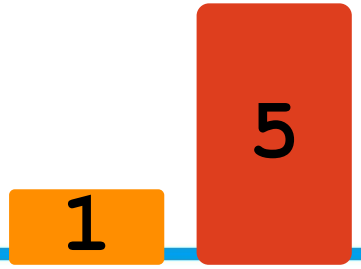


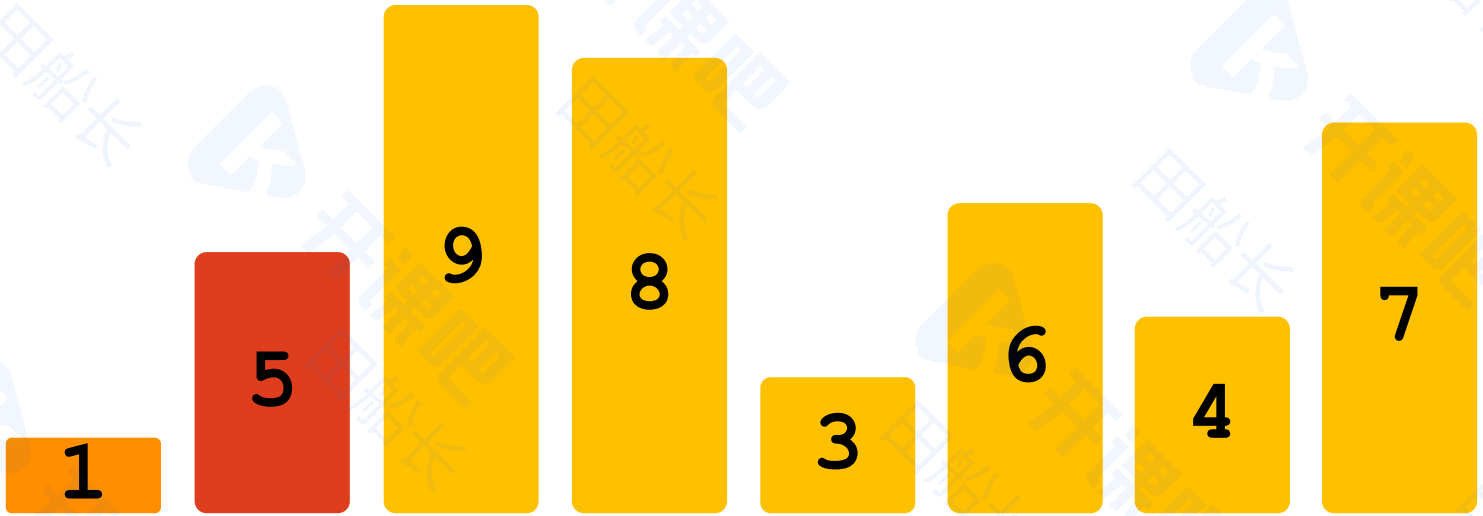
从待归并的数中选出最小值即1加入归并数组

1

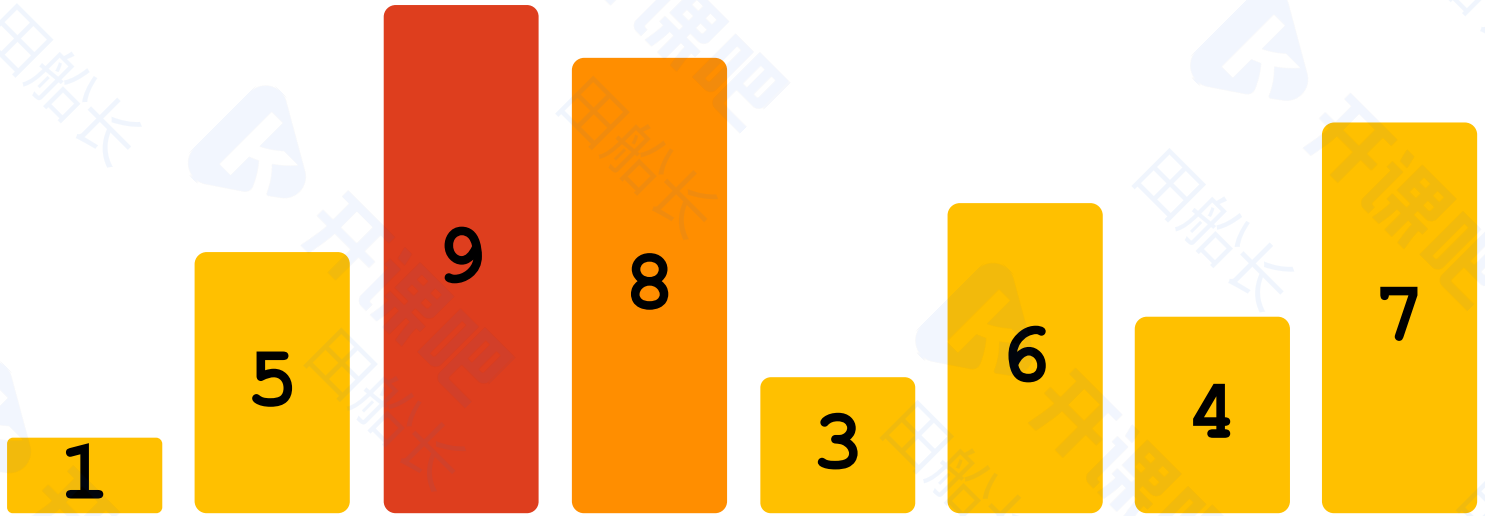


从待归并的数中选出最小值即5加入归并数组

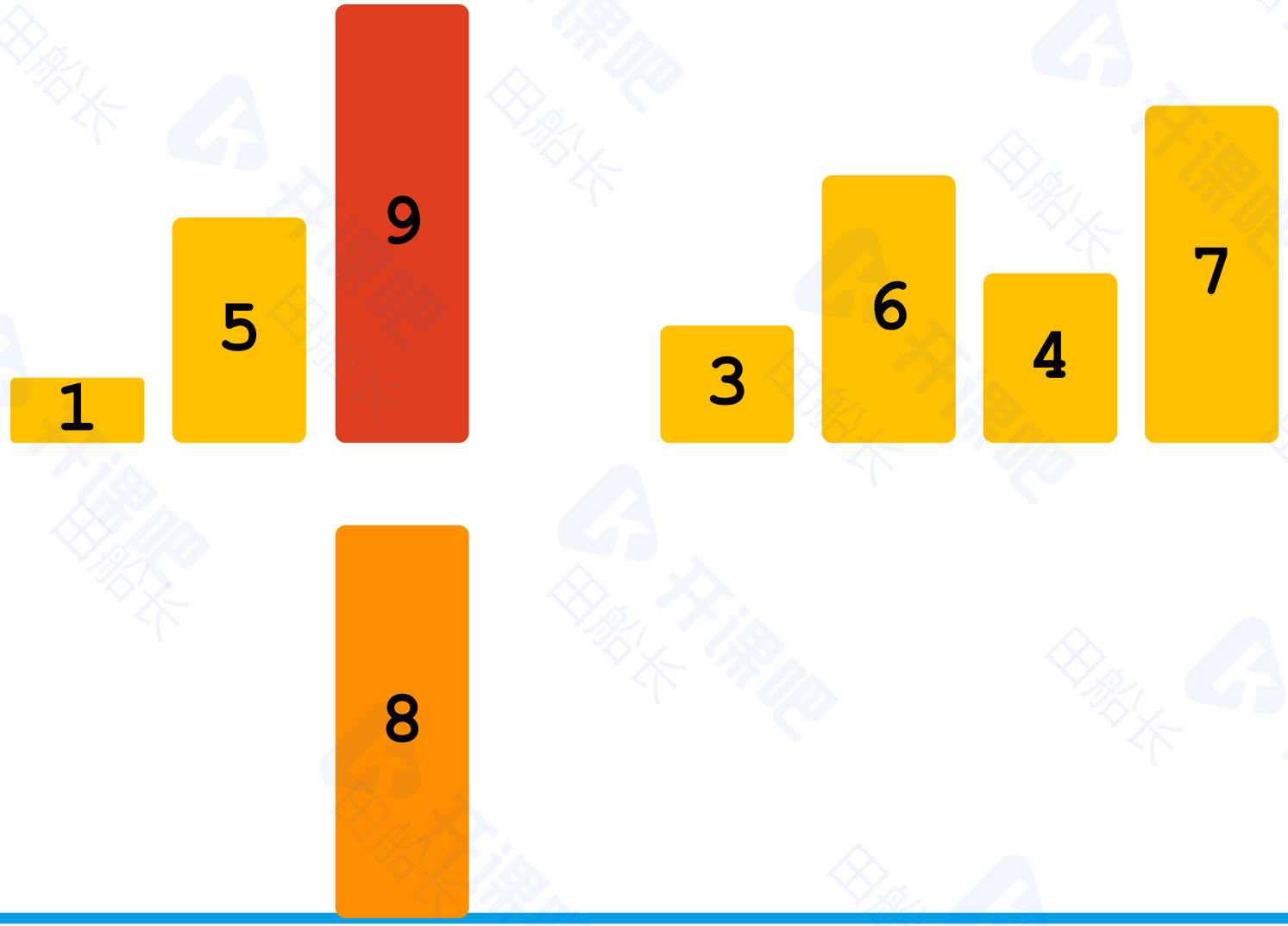




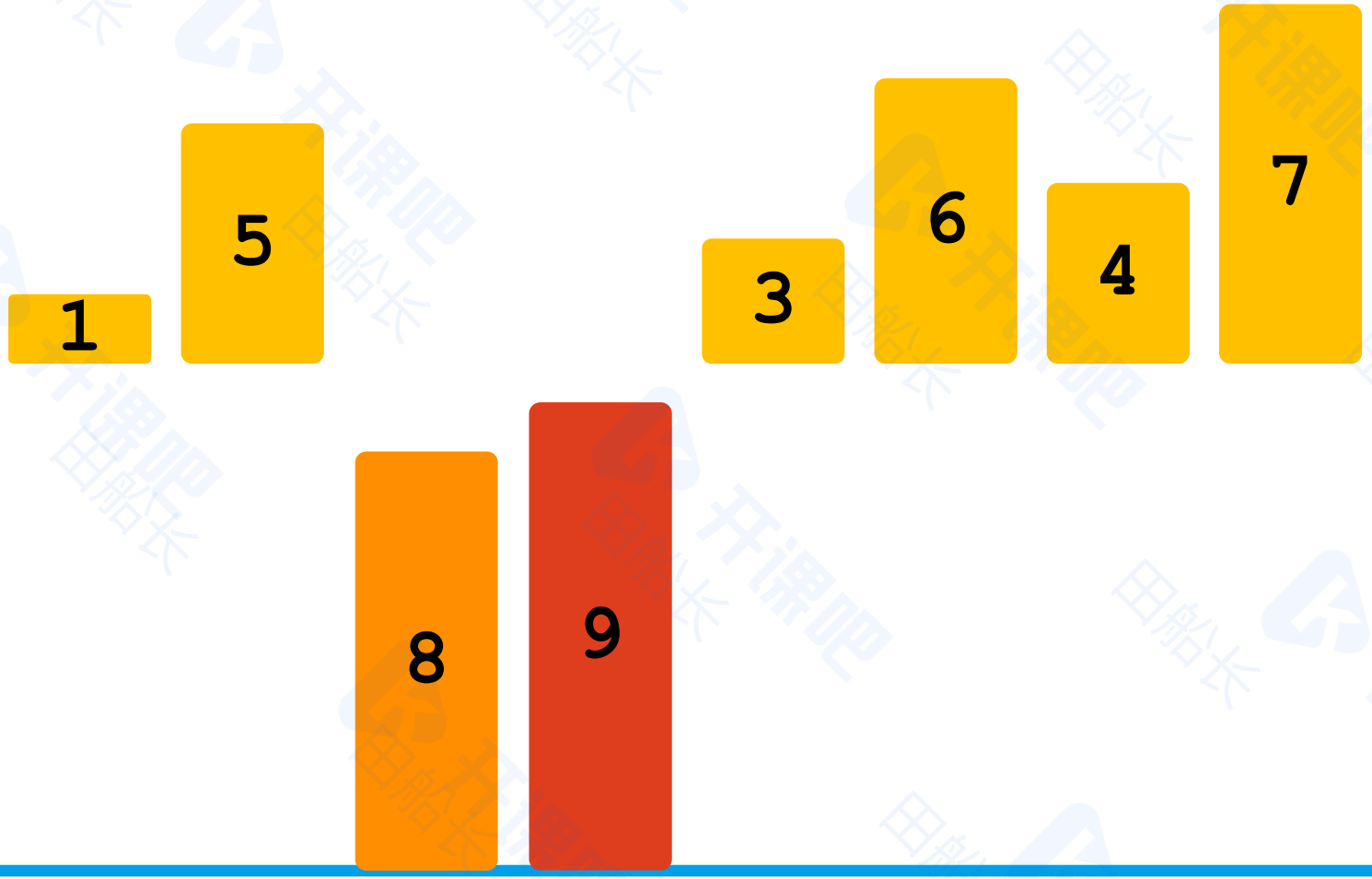
现在区间[0，2)内的数已经有序



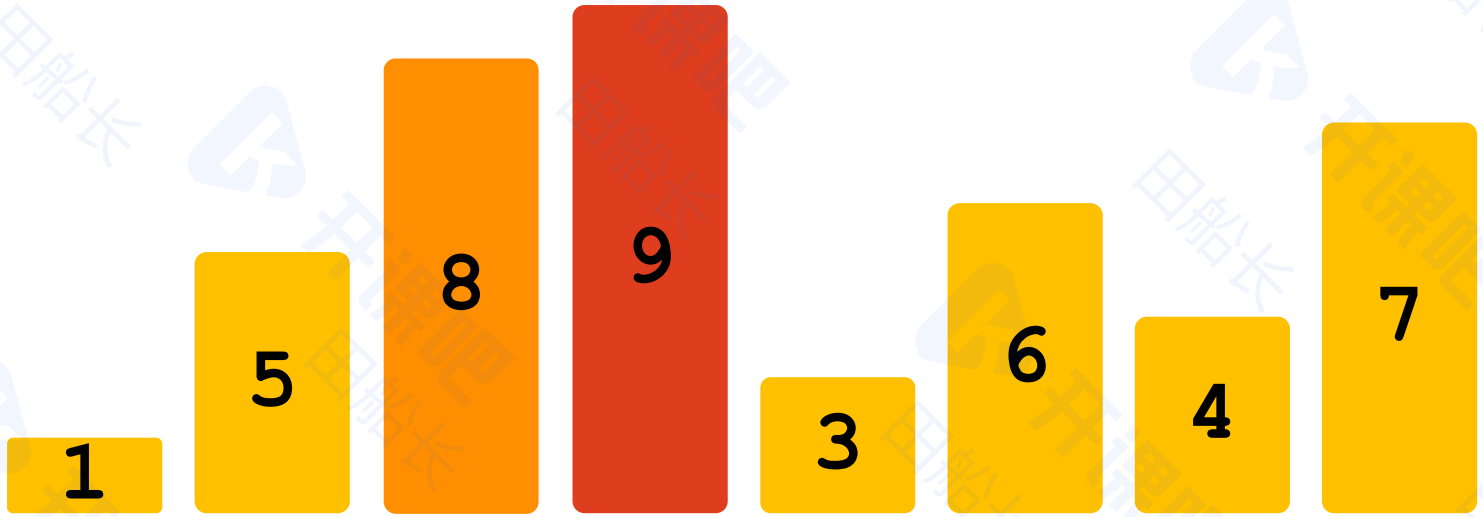
对两个区间[2，3)和[3，4)内的数进行归并



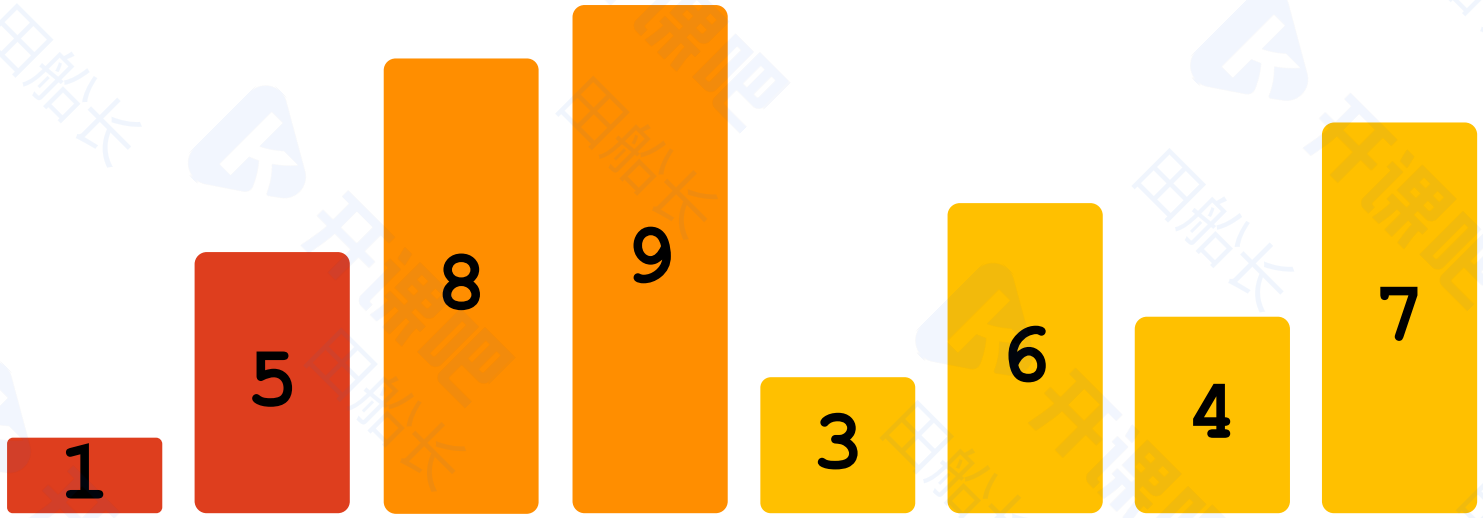
从待归并的数中选出最小值即8加入归并数组



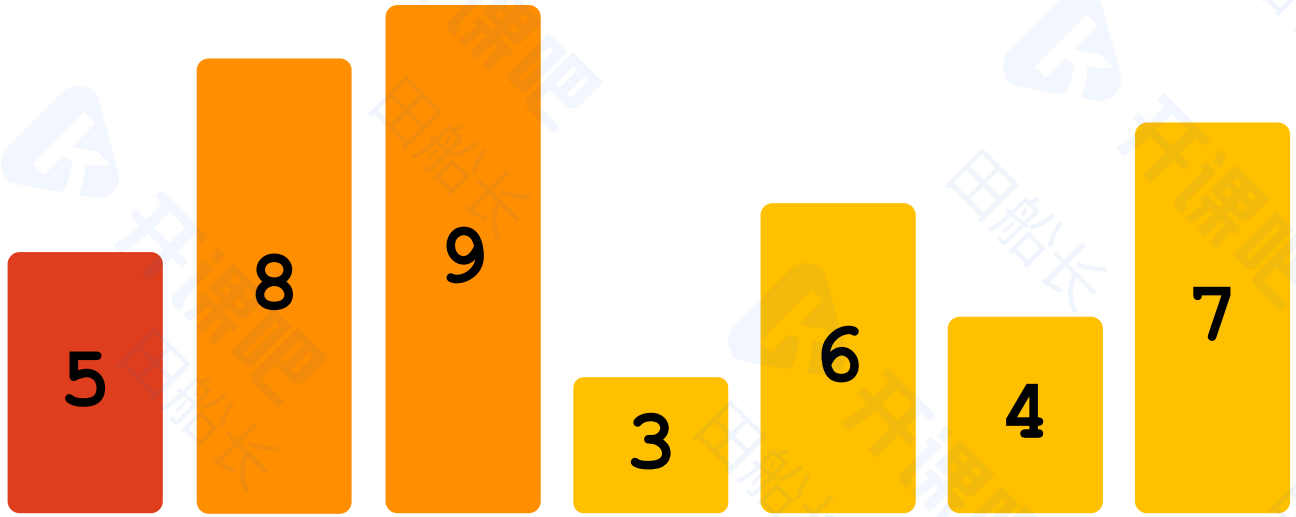
从待归并的数中选出最小值即9加入归并数组



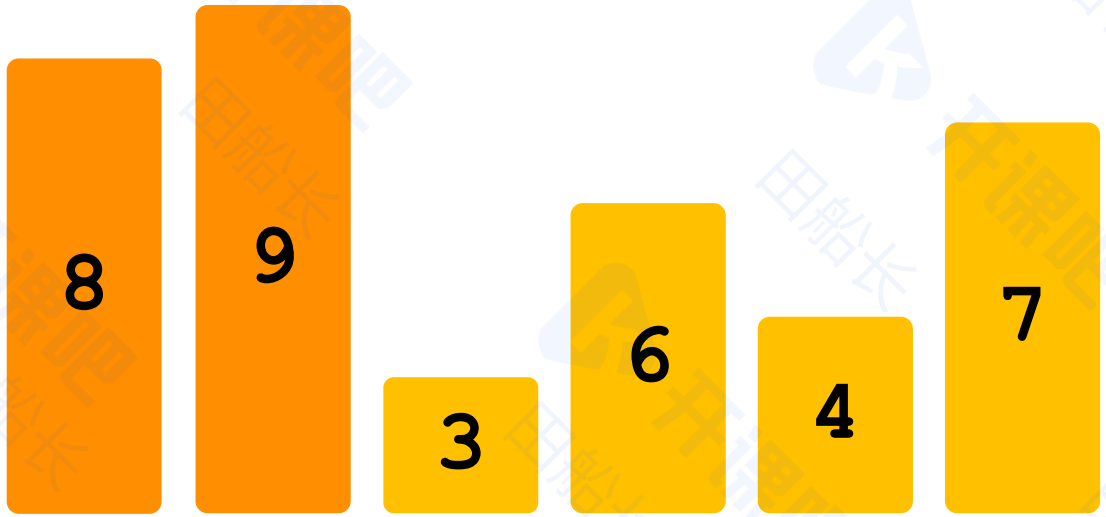
现在区间[2，4)内的数已经有序



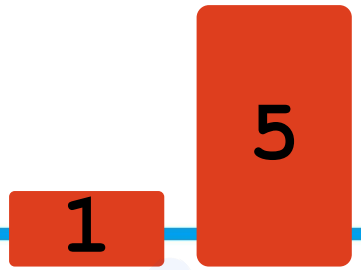
对两个区间[0，2)和[2，4)内的数进行归并

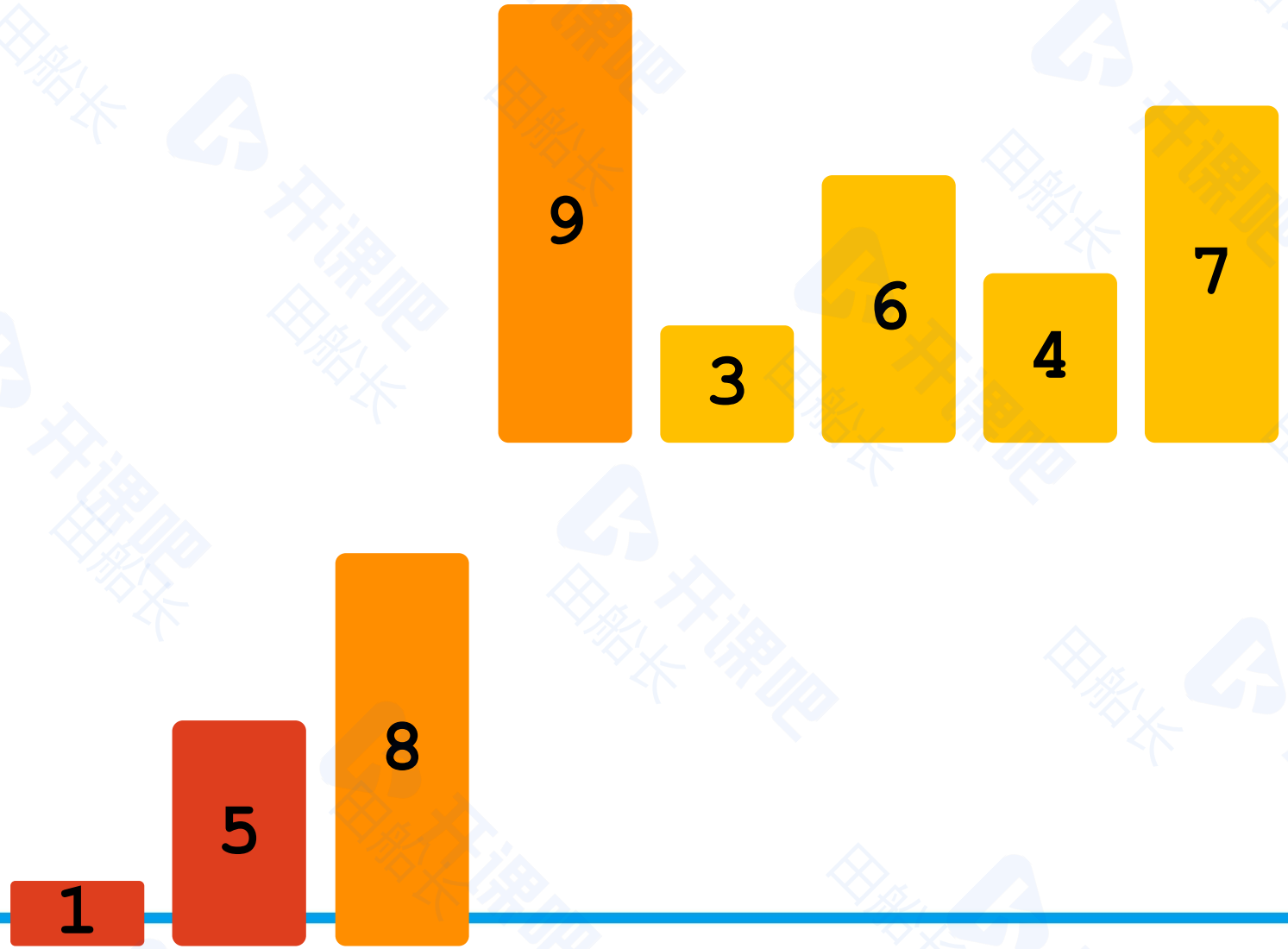


从待归并的数中选出最小值即1加入归并数组

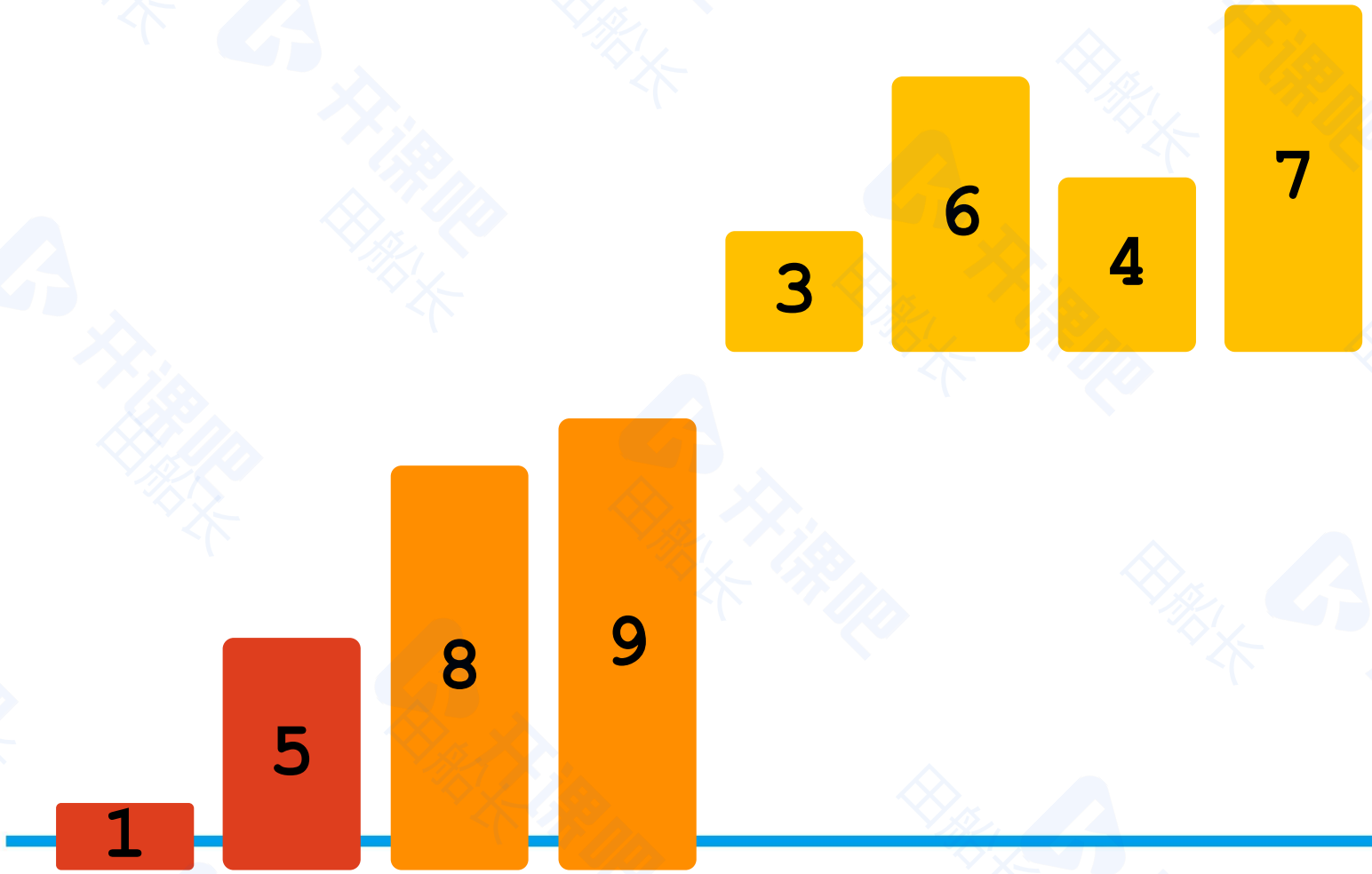


从待归并的数中选出最小值即5加入归并数组

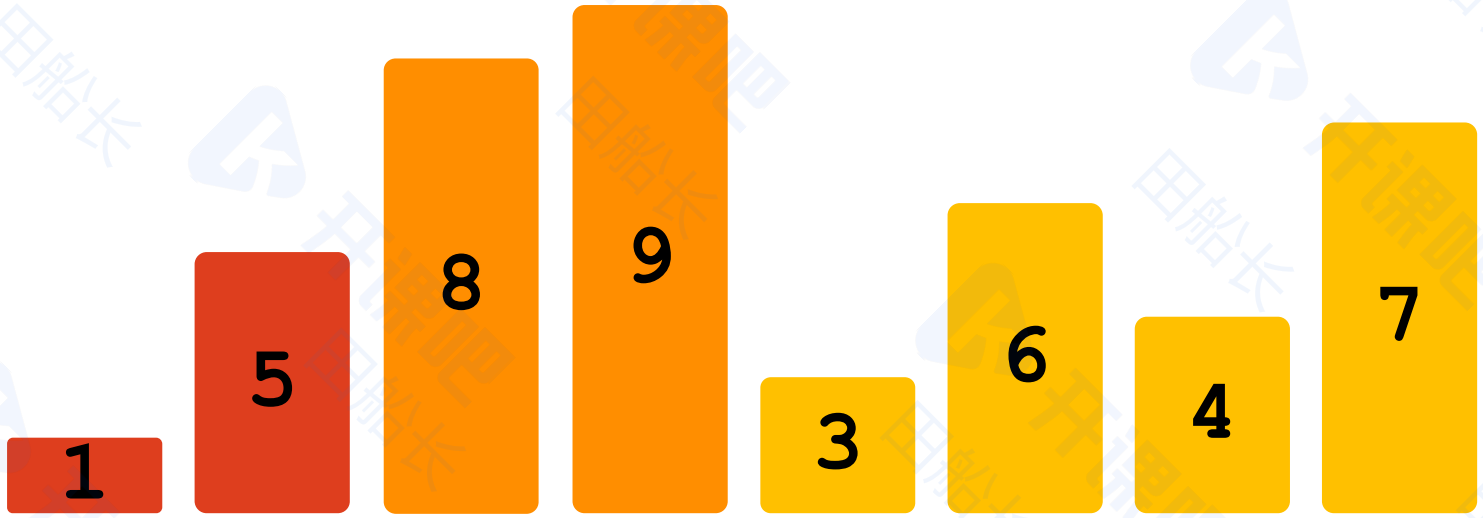




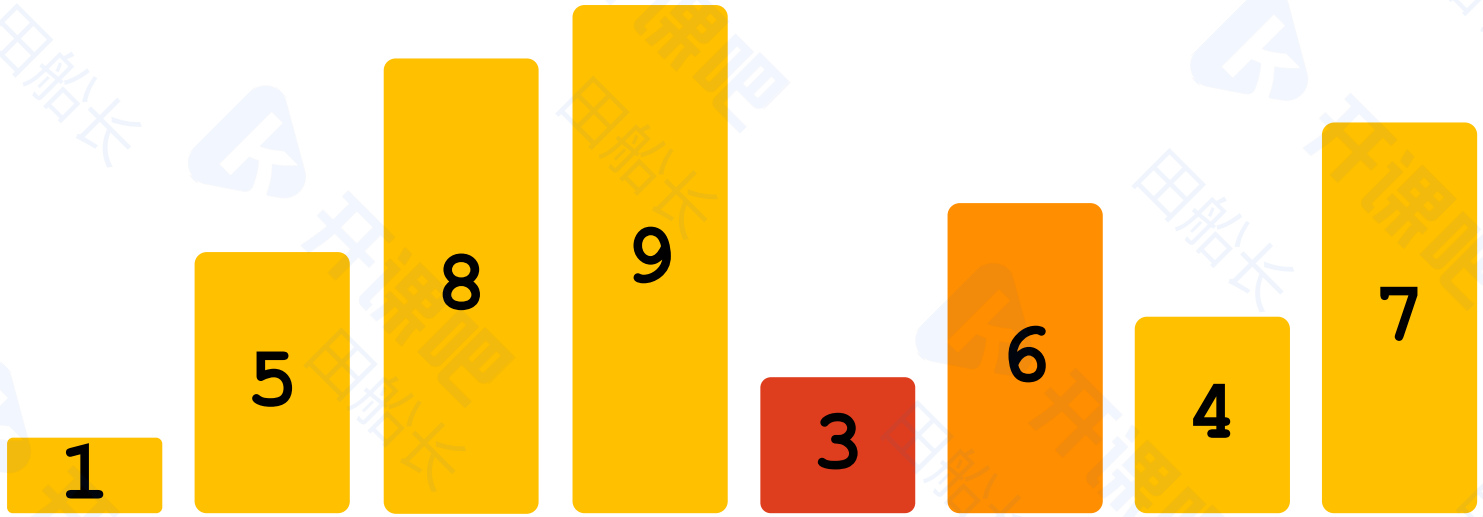
从待归并的数中选出最小值即8加入归并数组



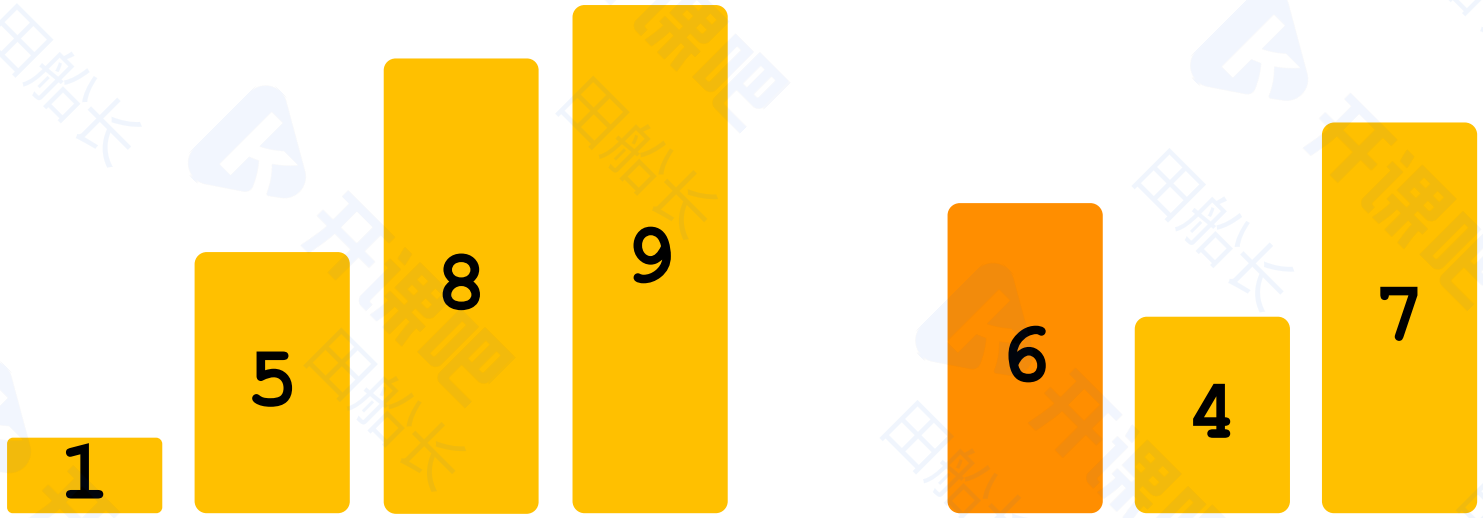
从待归并的数中选出最小值即9加入归并数组



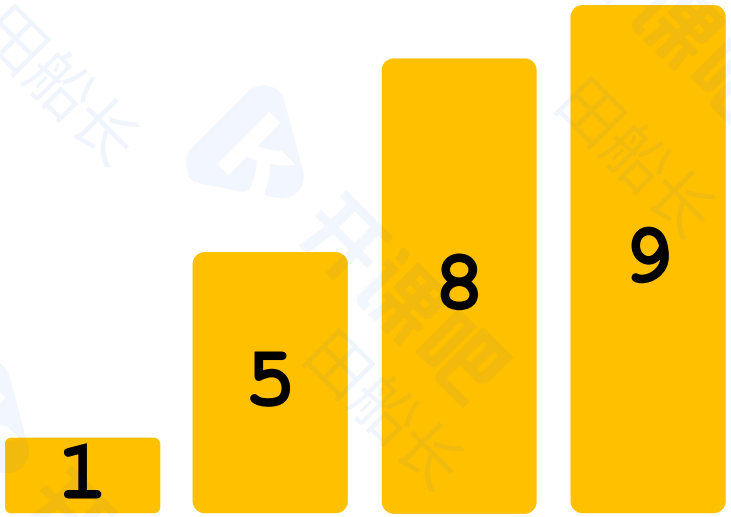
现在区间[0, 4)内的数已经有序

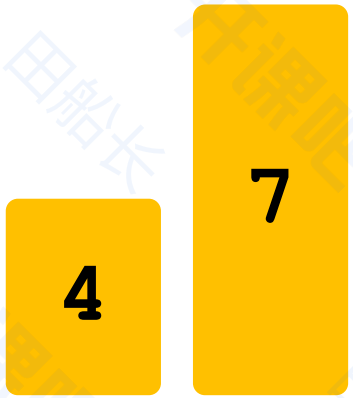


对两个区间[4，5)和[5，6)内的数进行归并

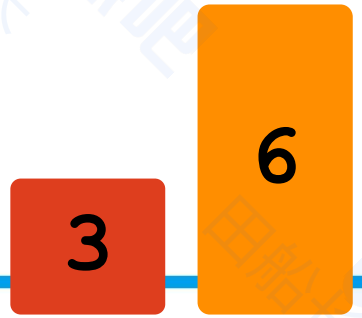


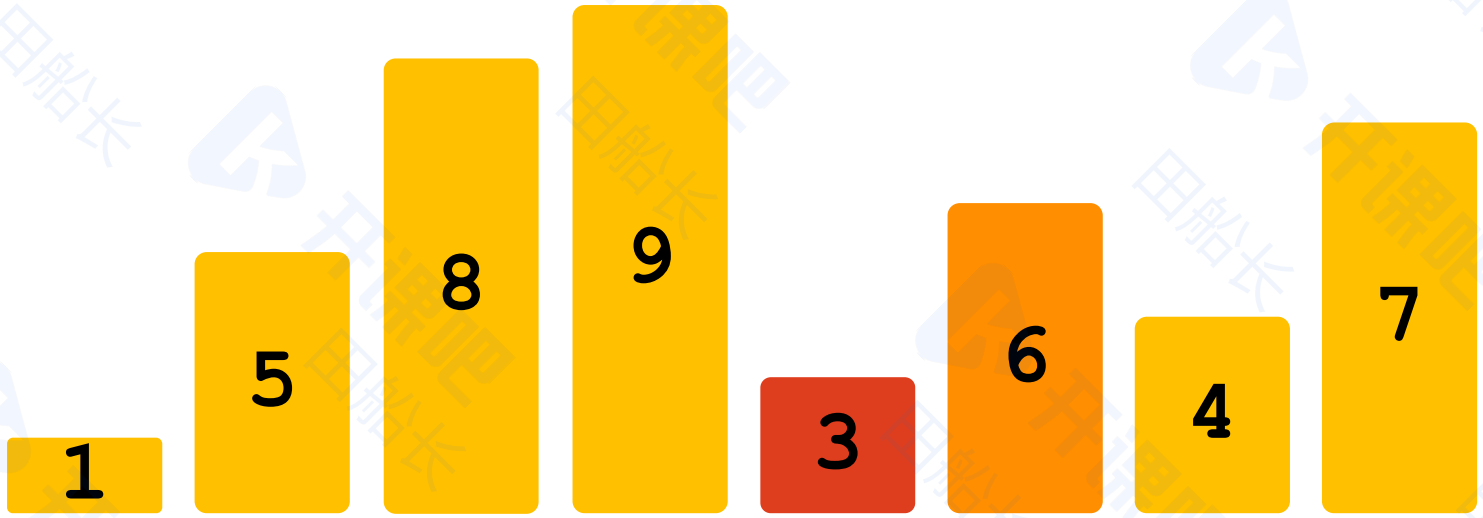
从待归并的数中选出最小值即3加入归并数组



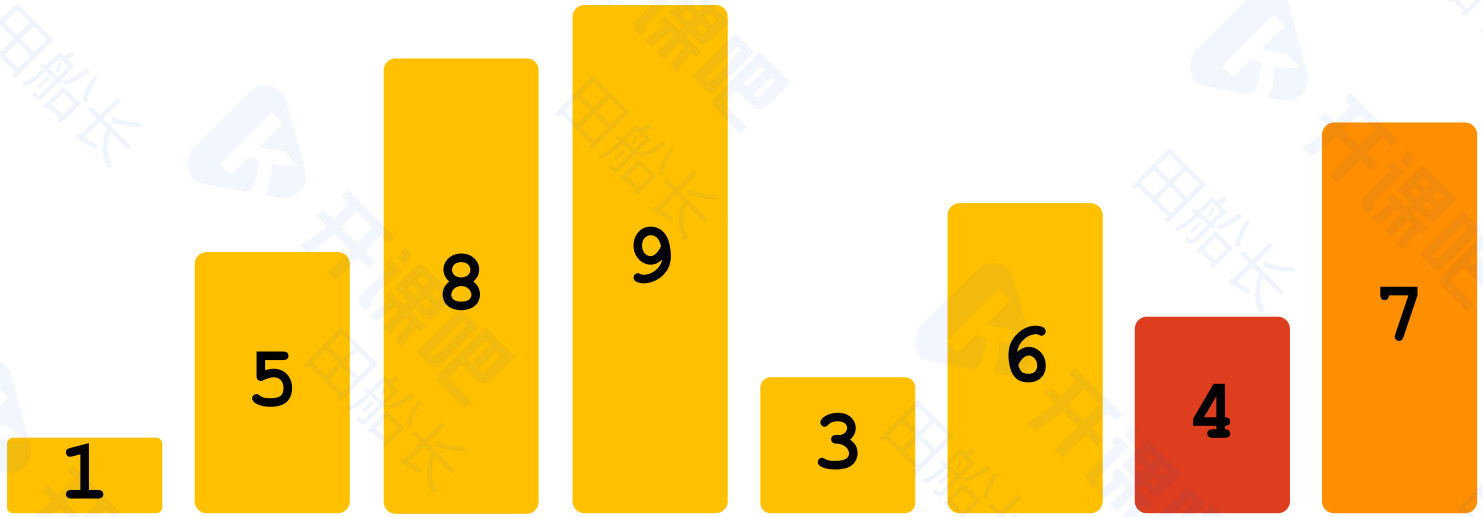


从待归并的数中选出最小值即6加入归并数组

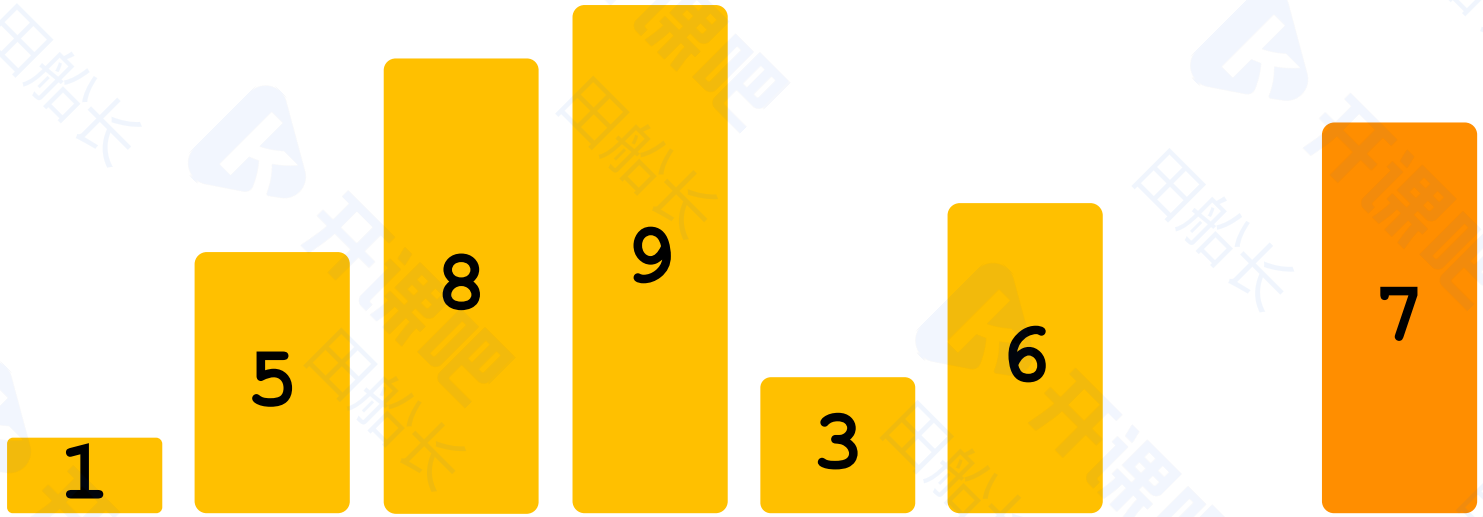




现在区间[4，6)内的数已经有序

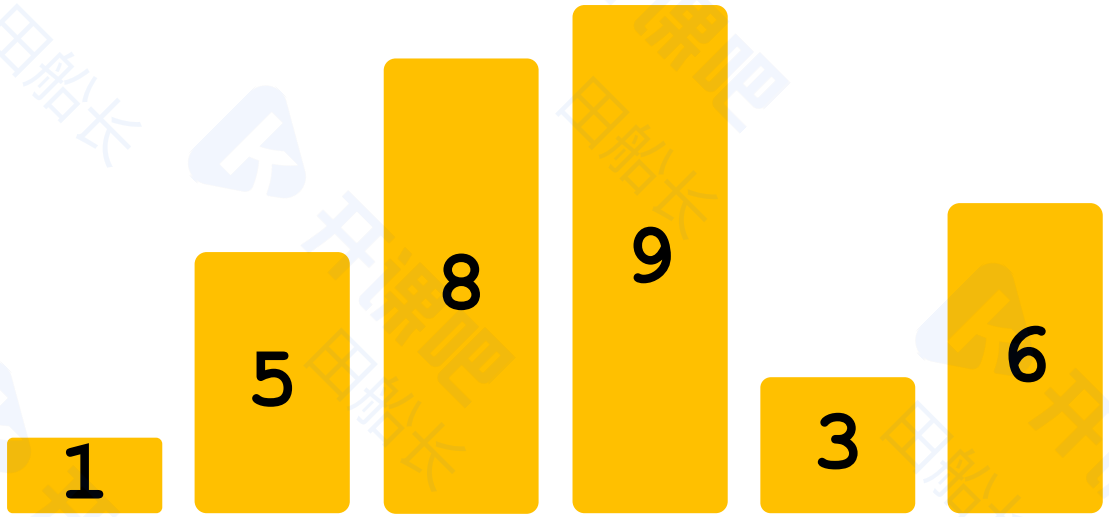


对两个区间[6，7)和[7，8)内的数进行归并

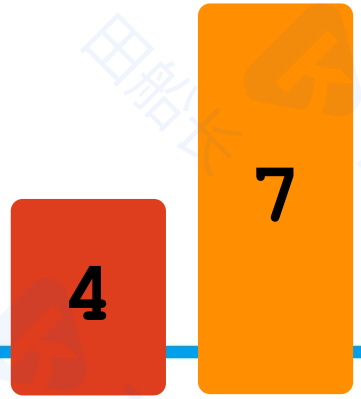


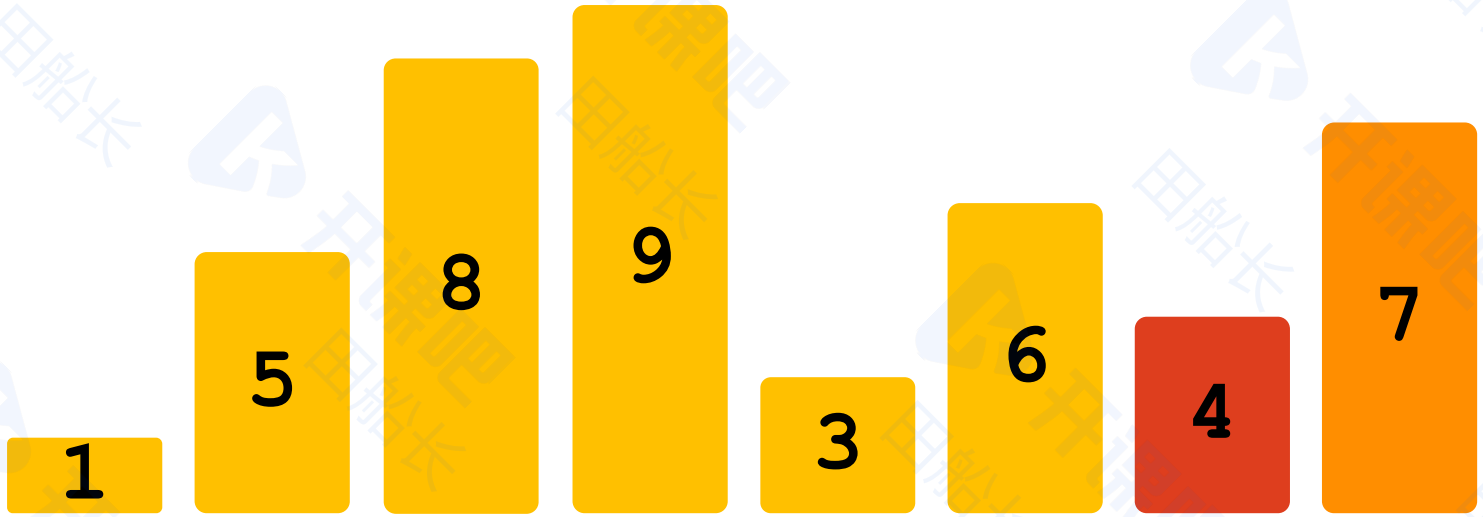
从待归并的数中选出最小值即4加入归并数组

4

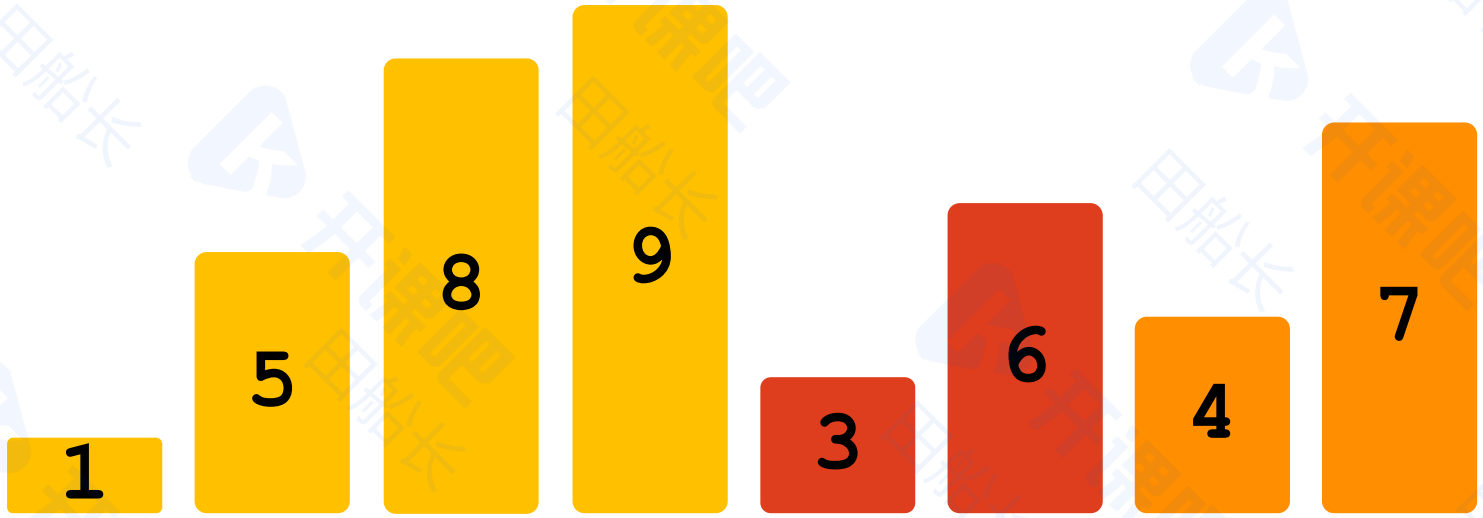


从待归并的数中选出最小值即7加入归并数组

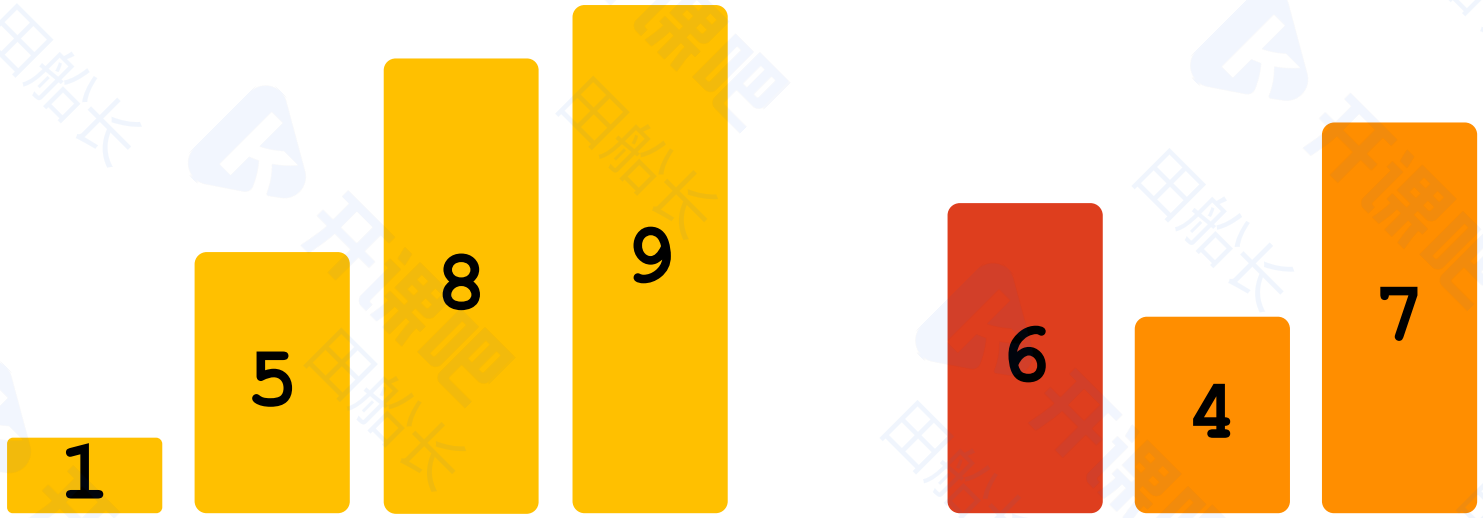




现在区间[6，8)内的数已经有序

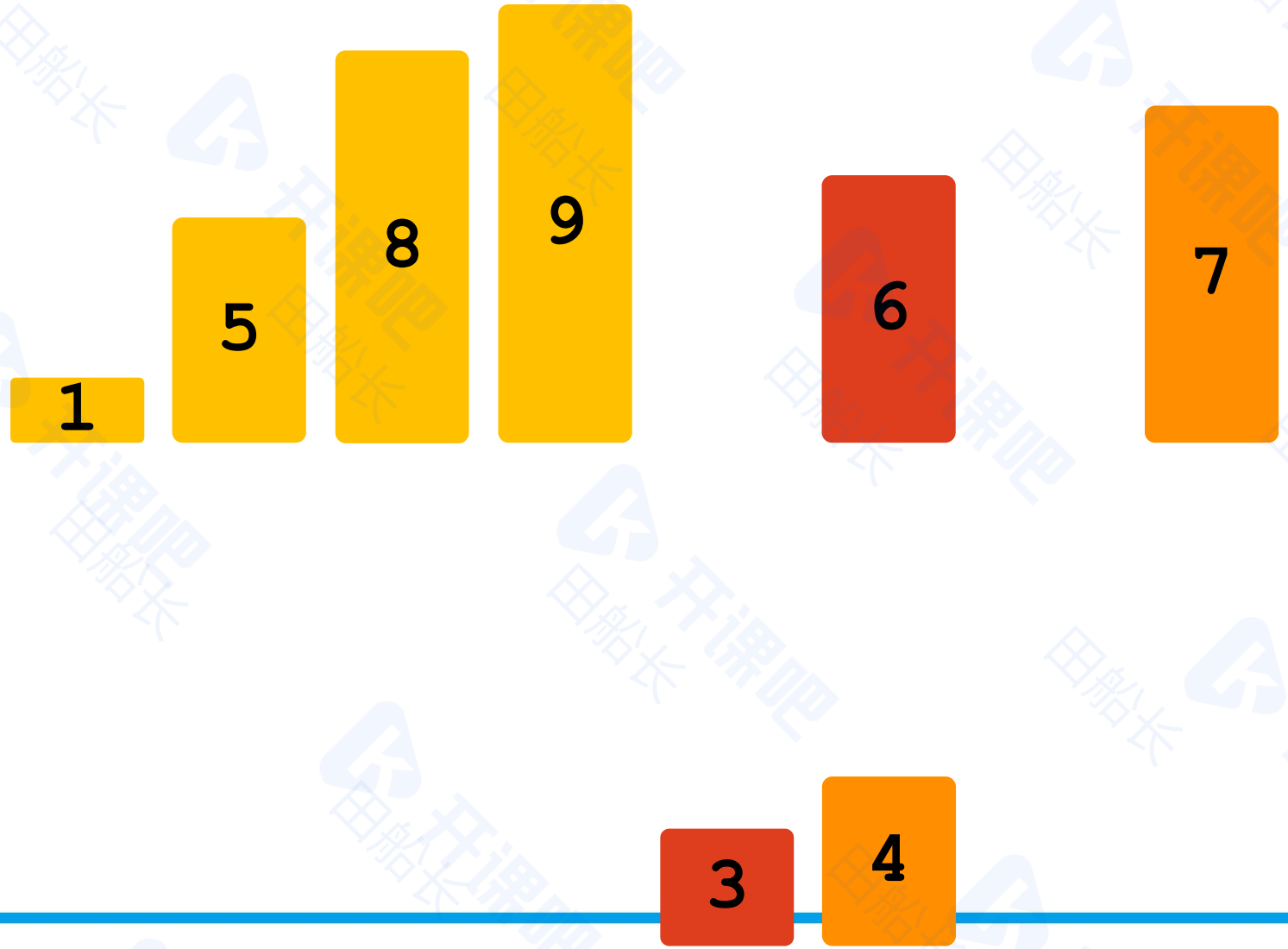


对两个区间[4，6)和[6，8)内的数进行归并

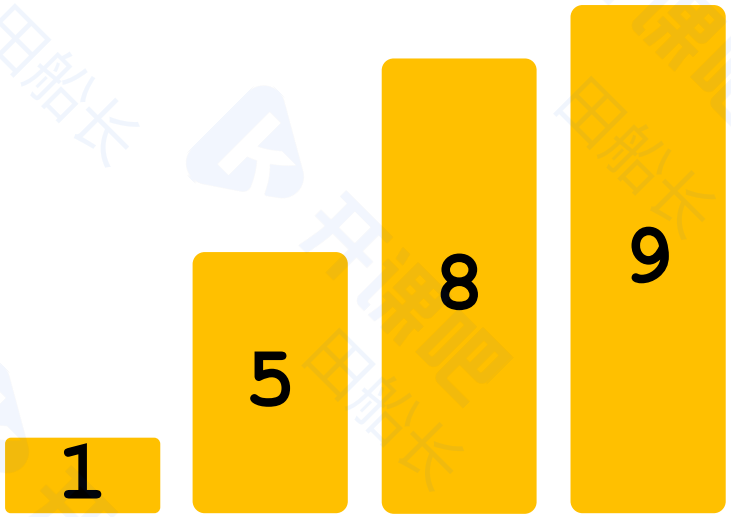


从待归并的数中选出最小值即3加入归并数组

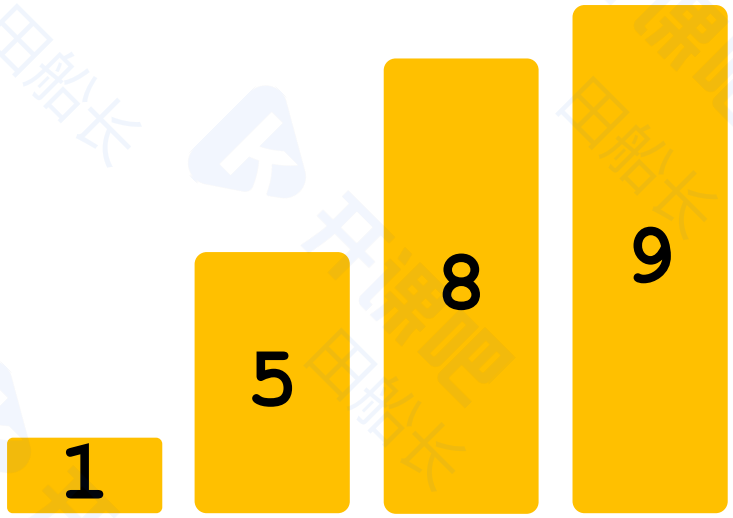
3



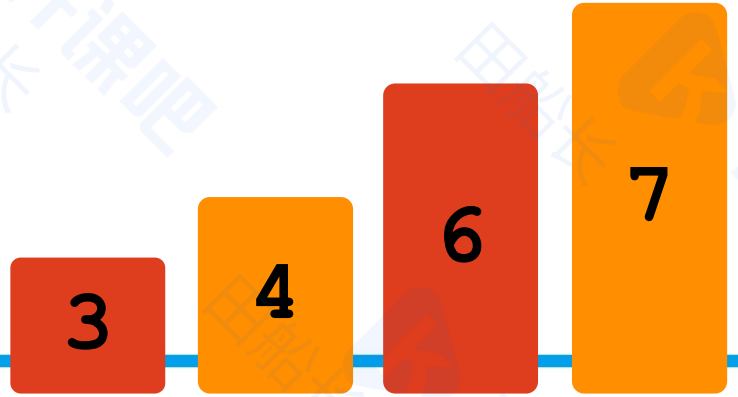
从待归并的数中选出最小值即4加入归并数组

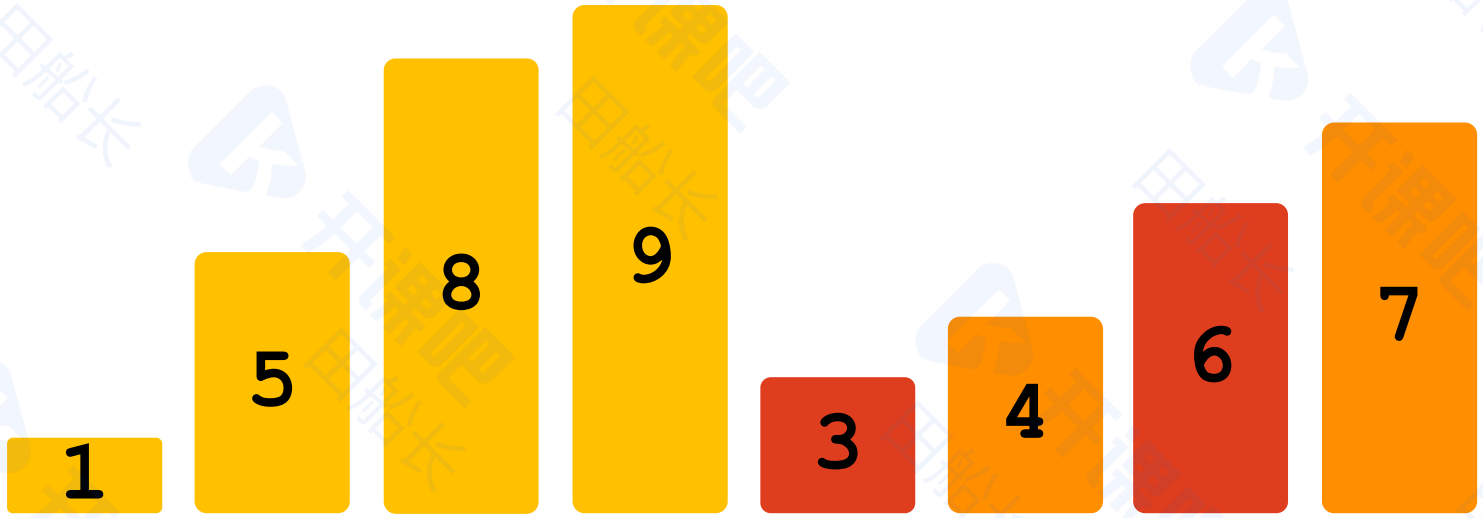


从待归并的数中选出最小值即6加入归并数组

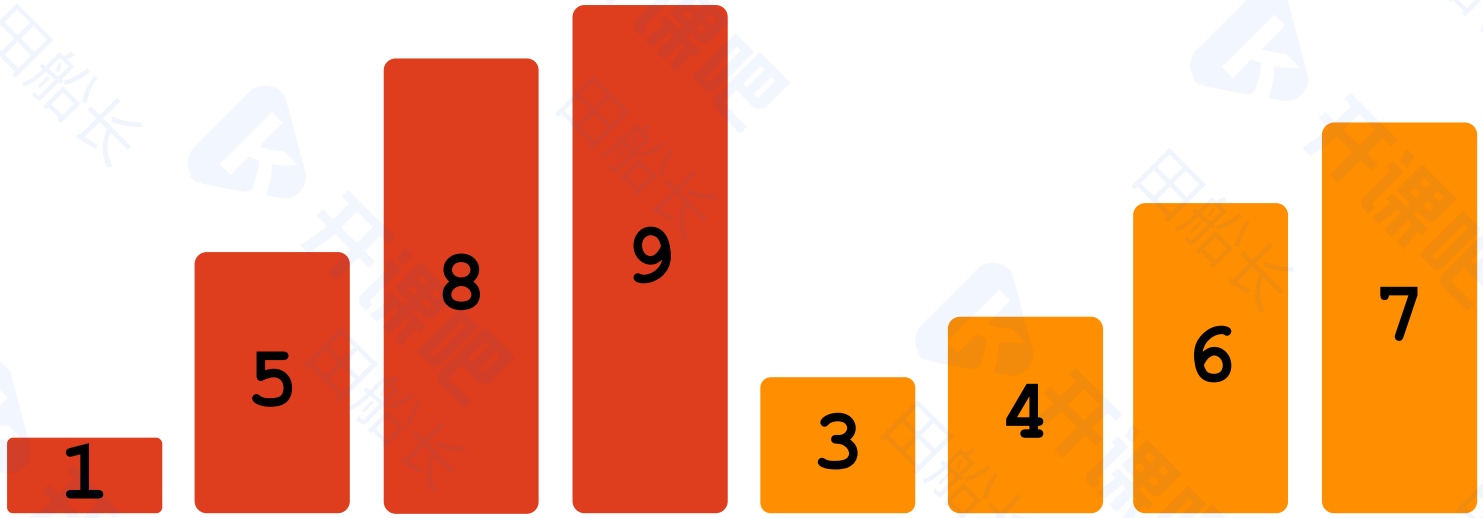


从待归并的数中选出最小值即7加入归并数组

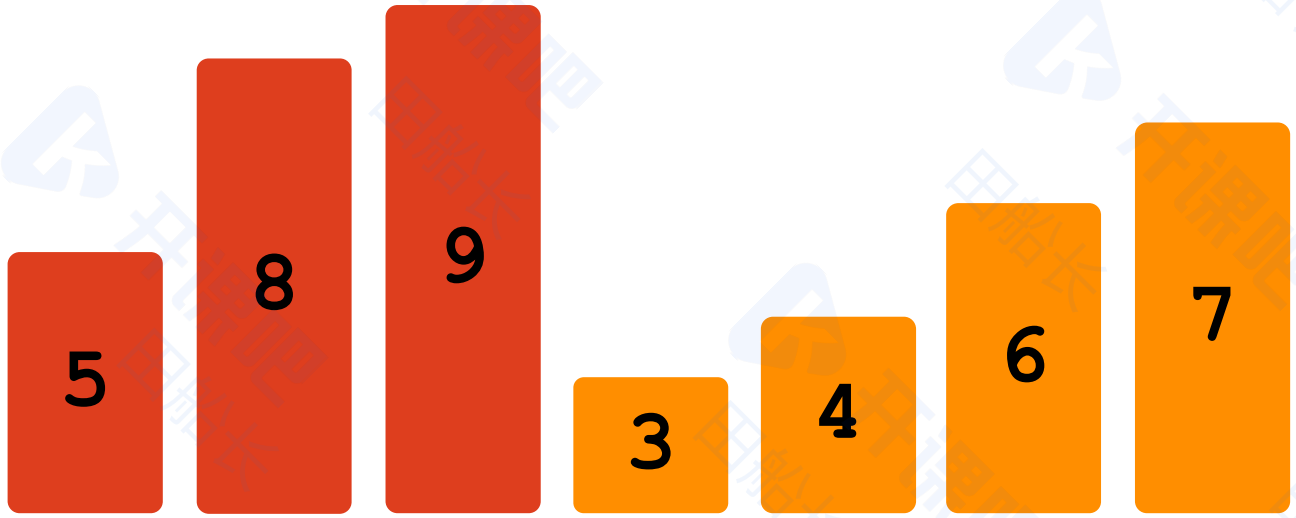




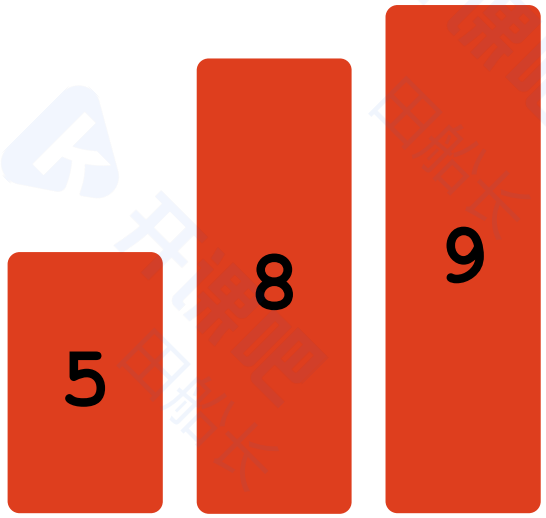
现在区间[4，8)内的数已经有序

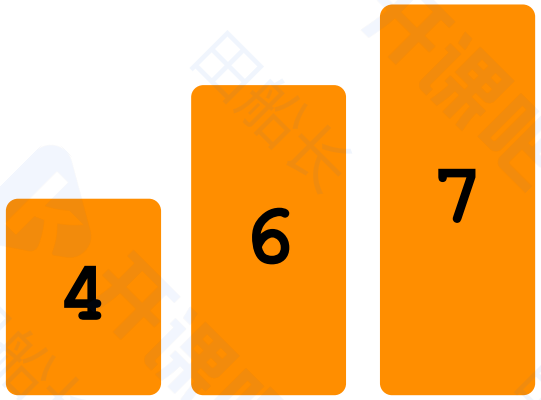


对两个区间[0，4)和[4，8)内的数进行归并

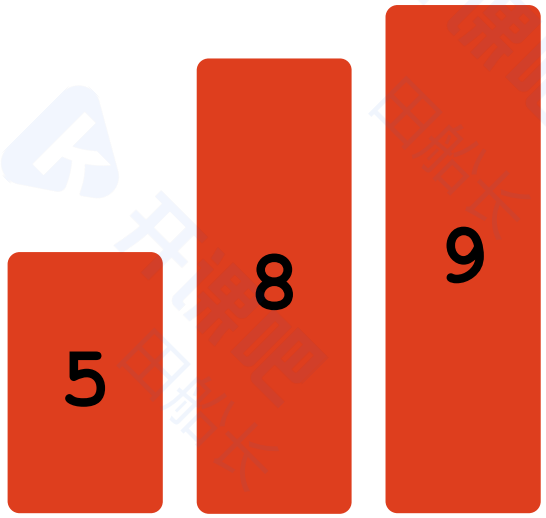


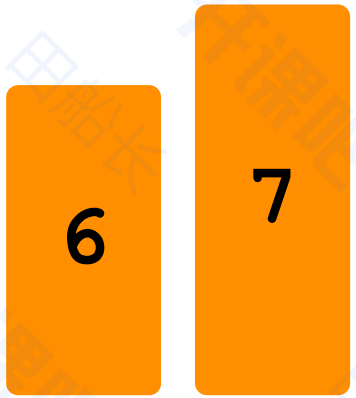
从待归并的数中选出最小值即1加入归并数组



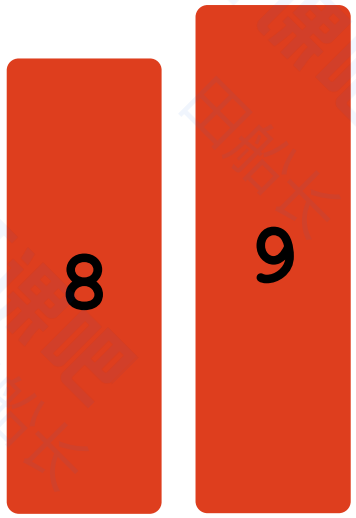


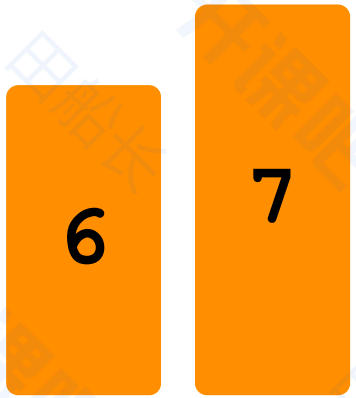
从待归并的数中选出最小值即3加入归并数组



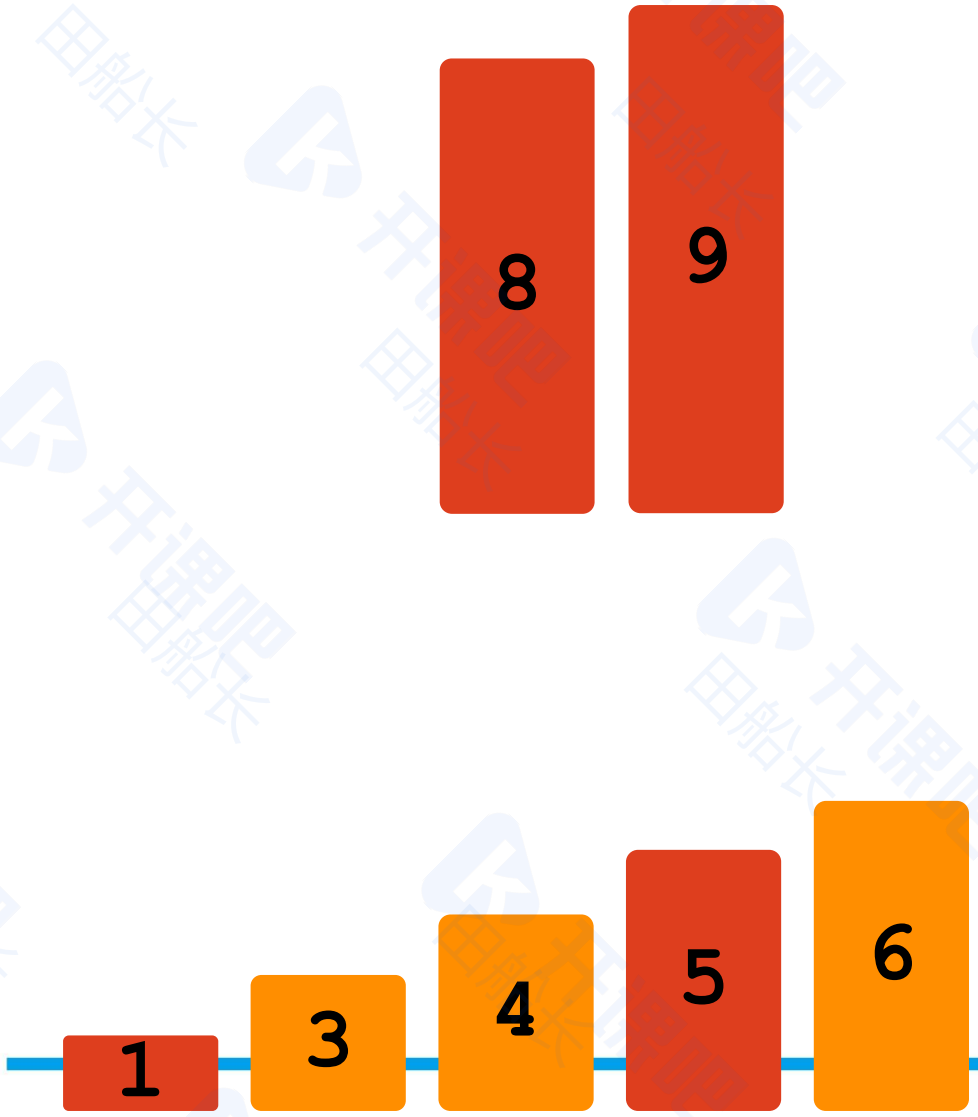


从待归并的数中选出最小值即4加入归并数组

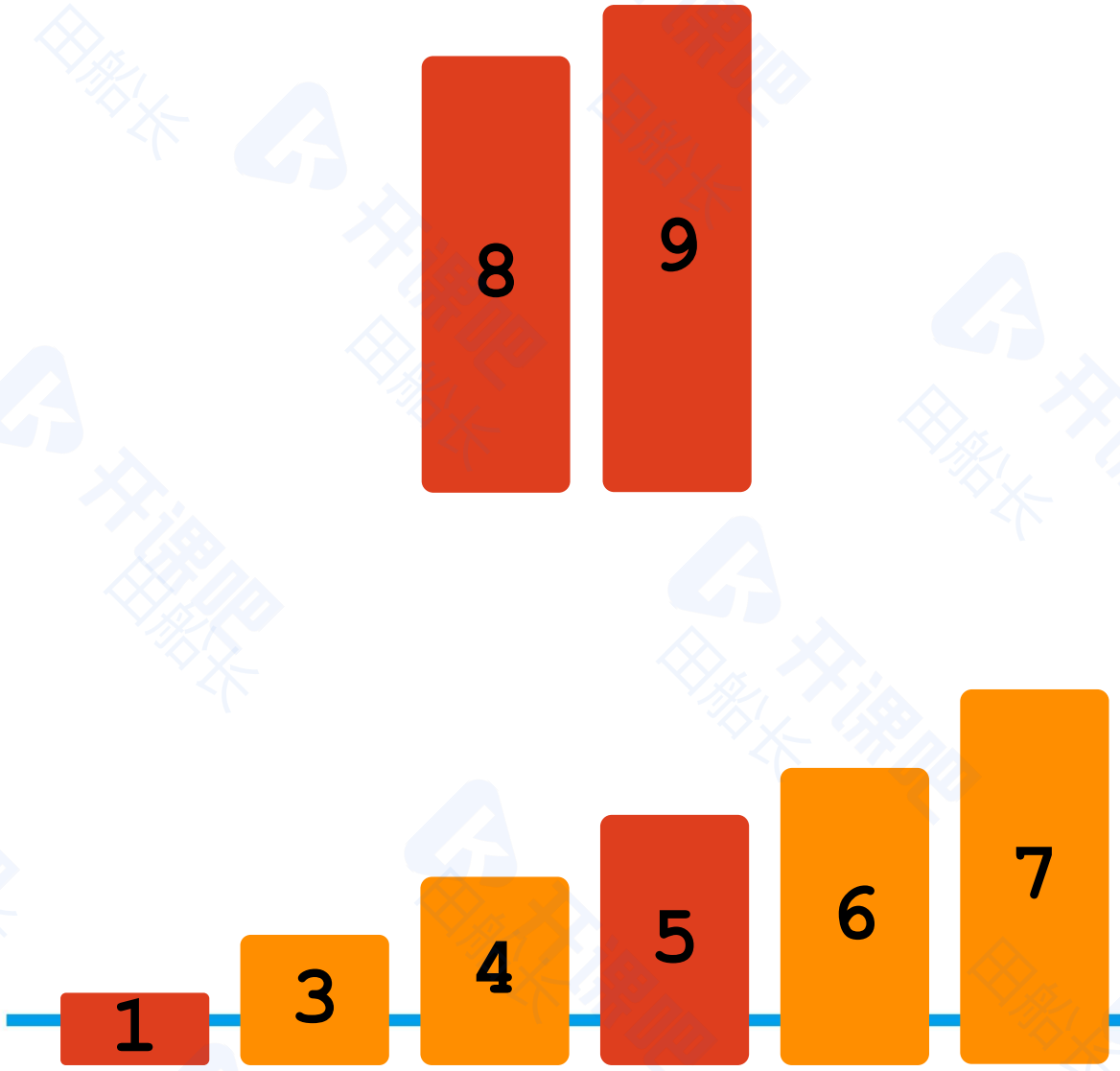




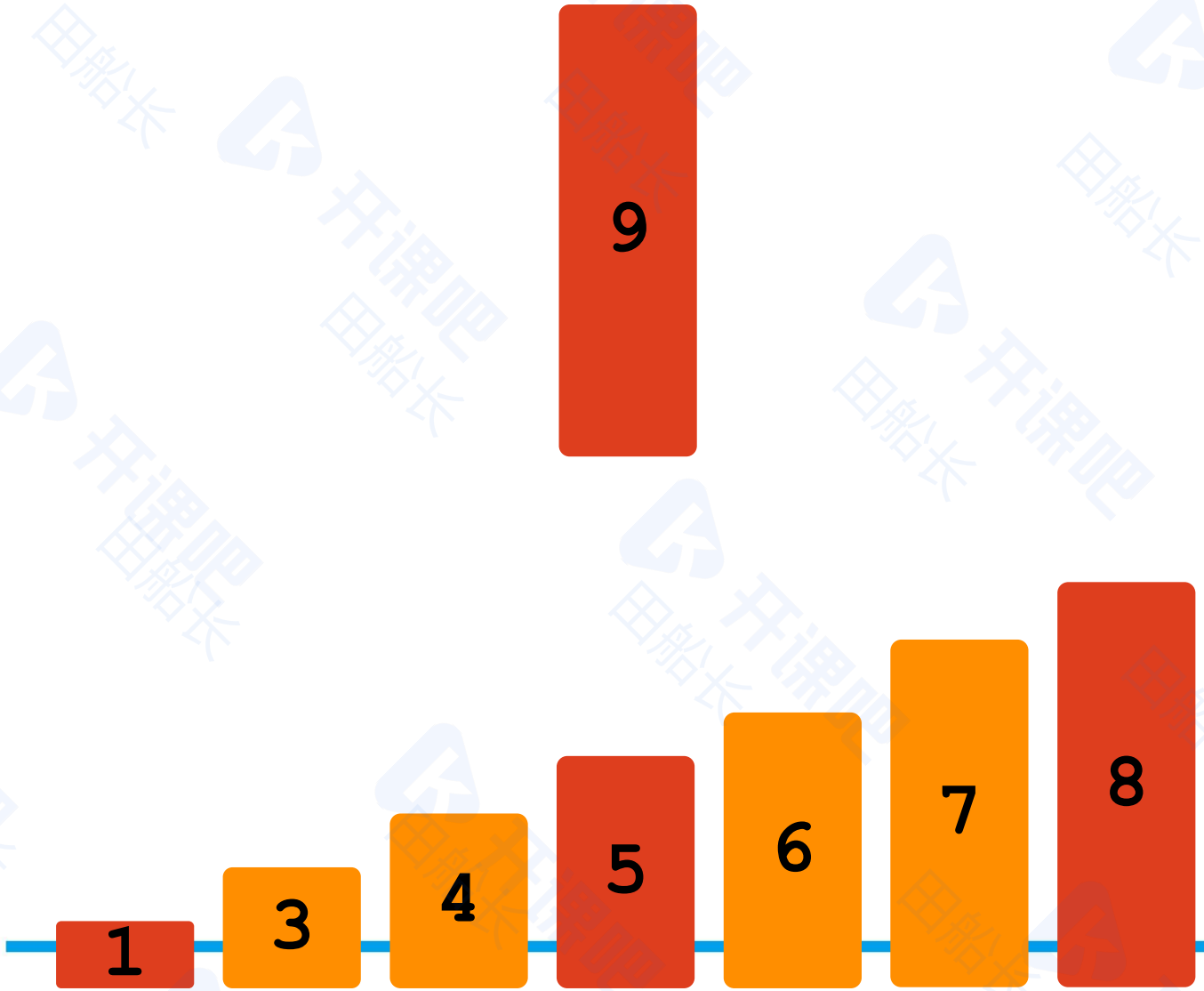
从待归并的数中选出最小值即5加入归并数组



从待归并的数中选出最小值即6加入归并数组

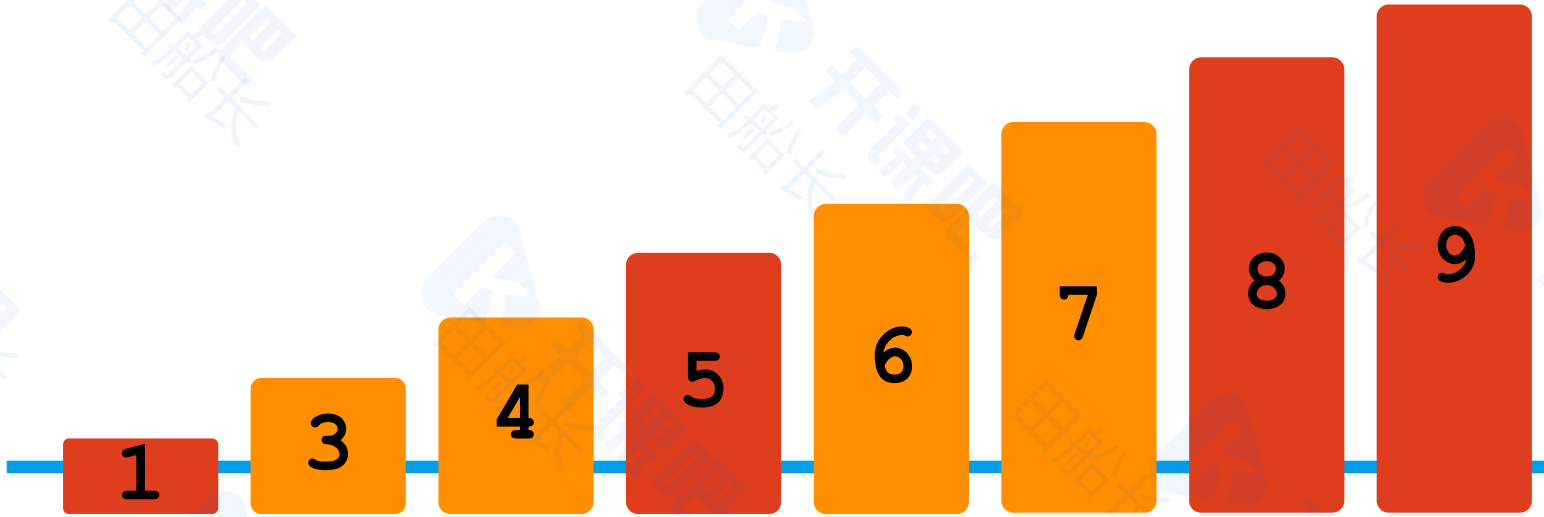


从待归并的数中选出最小值即7加入归并数组



从待归并的数中选出最小值即8加入归并数组

从待归并的数中选出最小值即9加入归并数组



现在区间[0, 8)内的数已经有序

归并排序结束

统计每个元素的出现次数，进行排序

将待排序序列组织进不同的桶当中
再按照桶的顺序将元素拿出
经过k轮操作后
序列整体有序

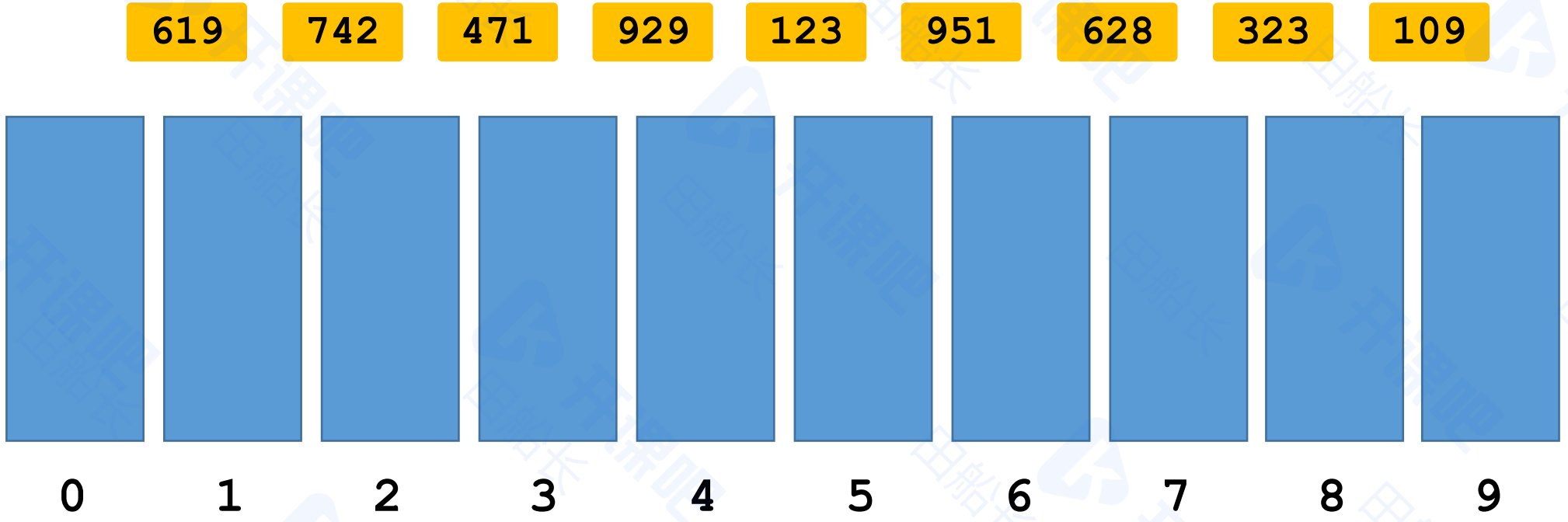
注：还有一种排序叫计数排序，与基数排序不同

619 742 471 929 123 951 628 323 109

开课吧

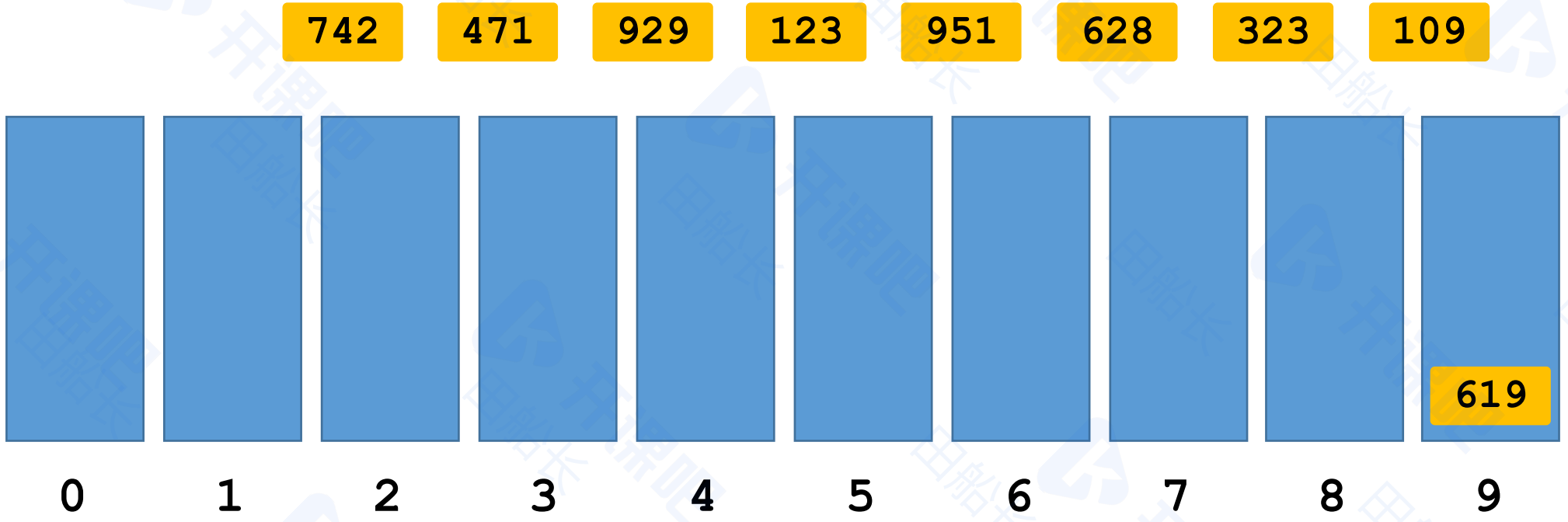
第一趟排序

以个位组织进桶中



第一趟排序

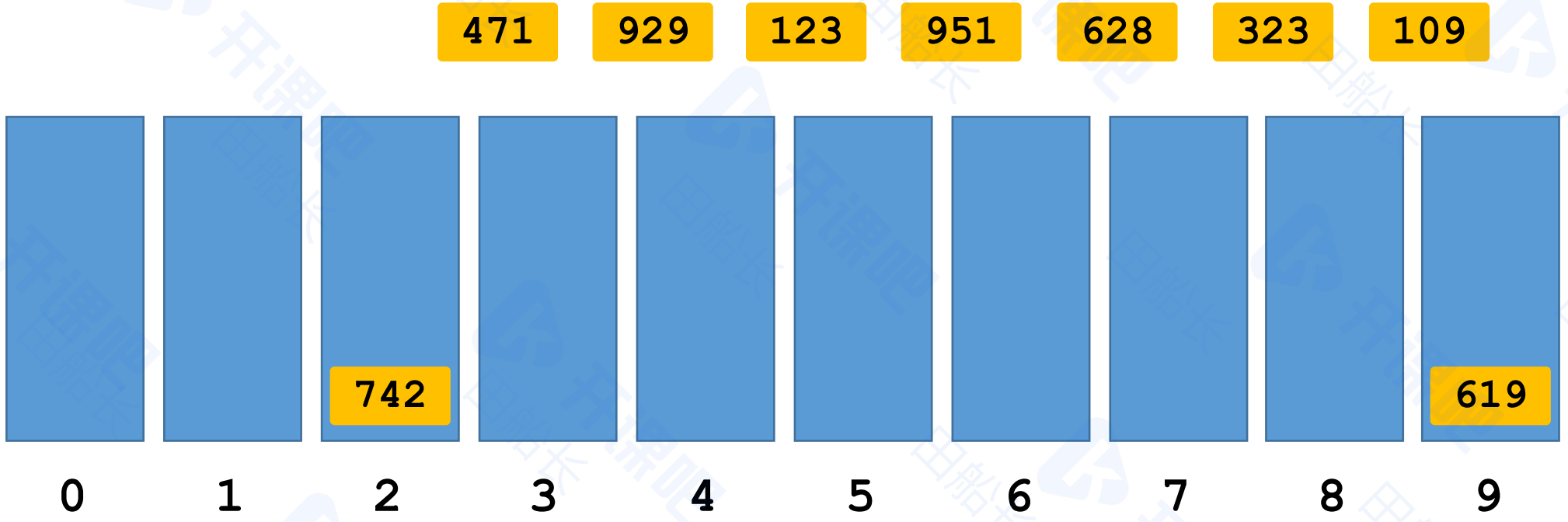
以个位组织进桶中



开课吧

第一趟排序

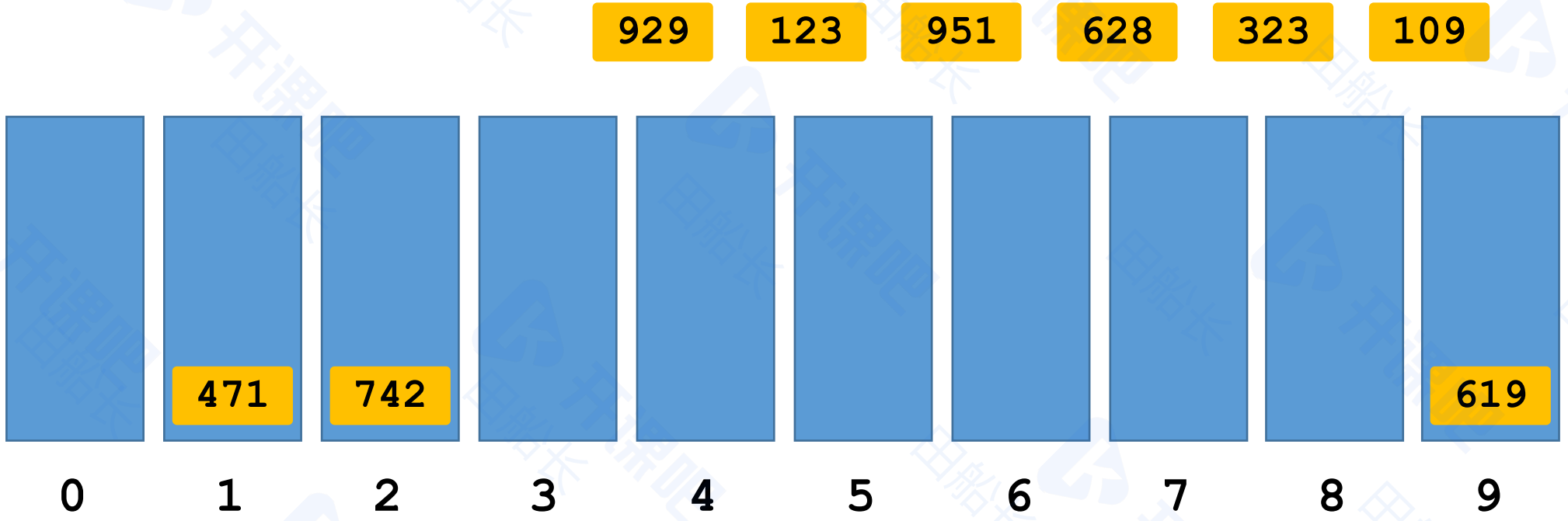
以个位组织进桶中



开课吧

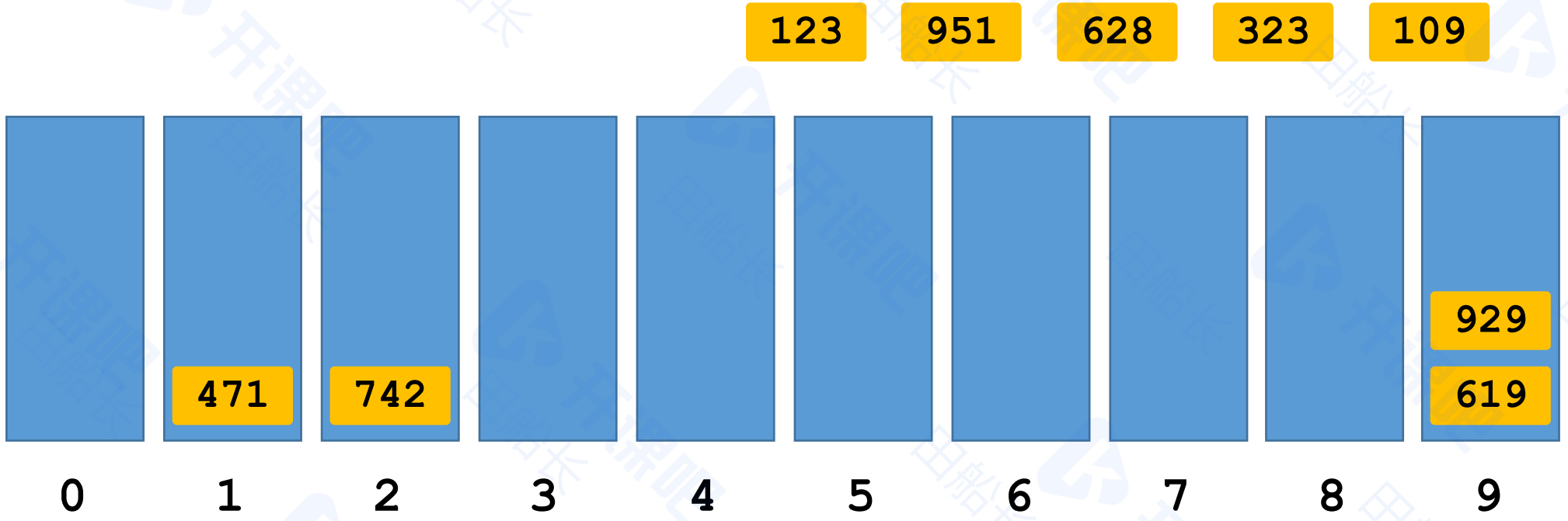
第一趟排序

以个位组织进桶中



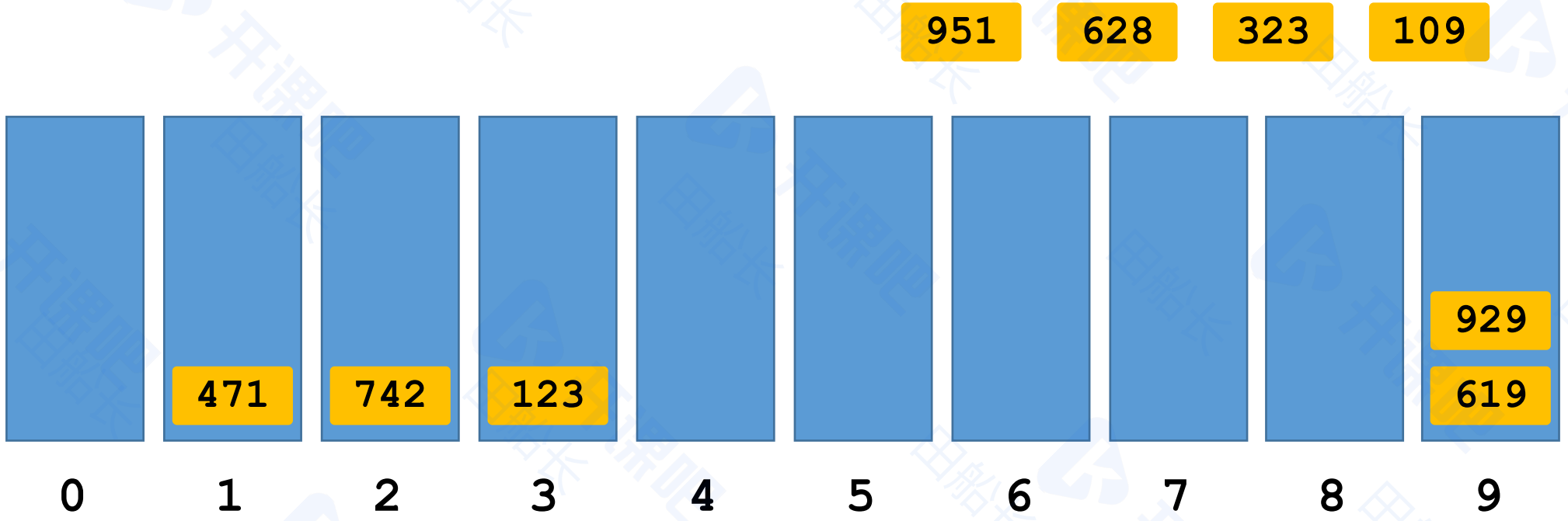
第一趟排序

以个位组织进桶中



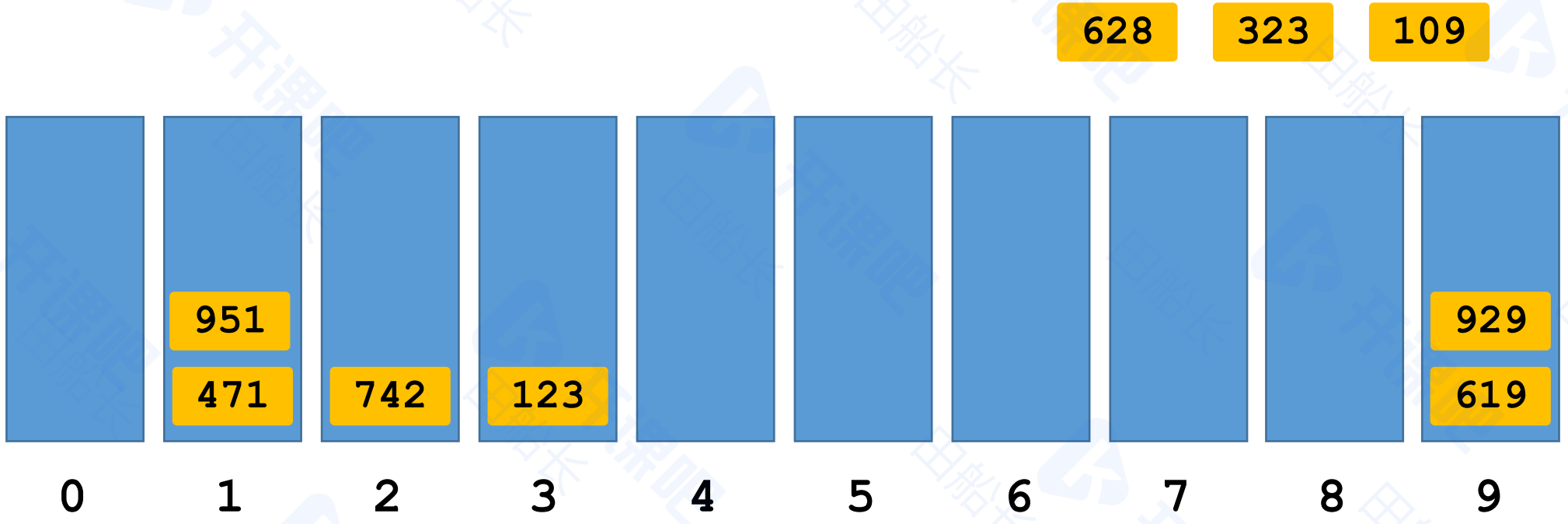
第一趟排序

以个位组织进桶中



第一趟排序

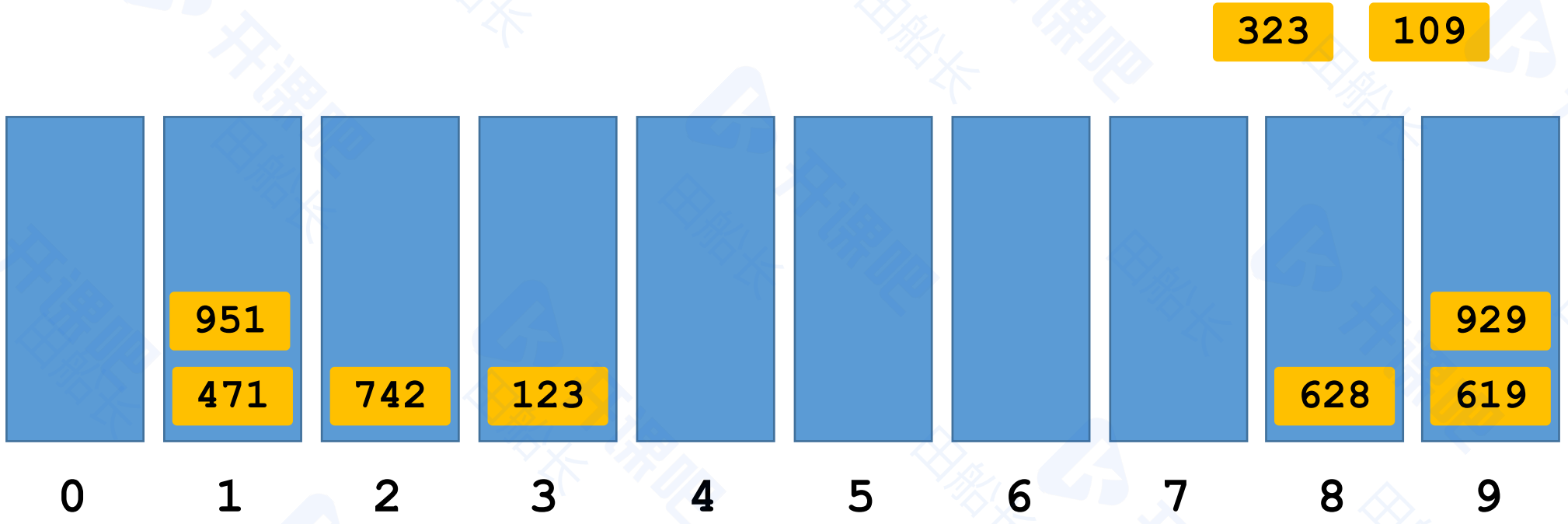
以个位组织进桶中



开课吧

第一趟排序

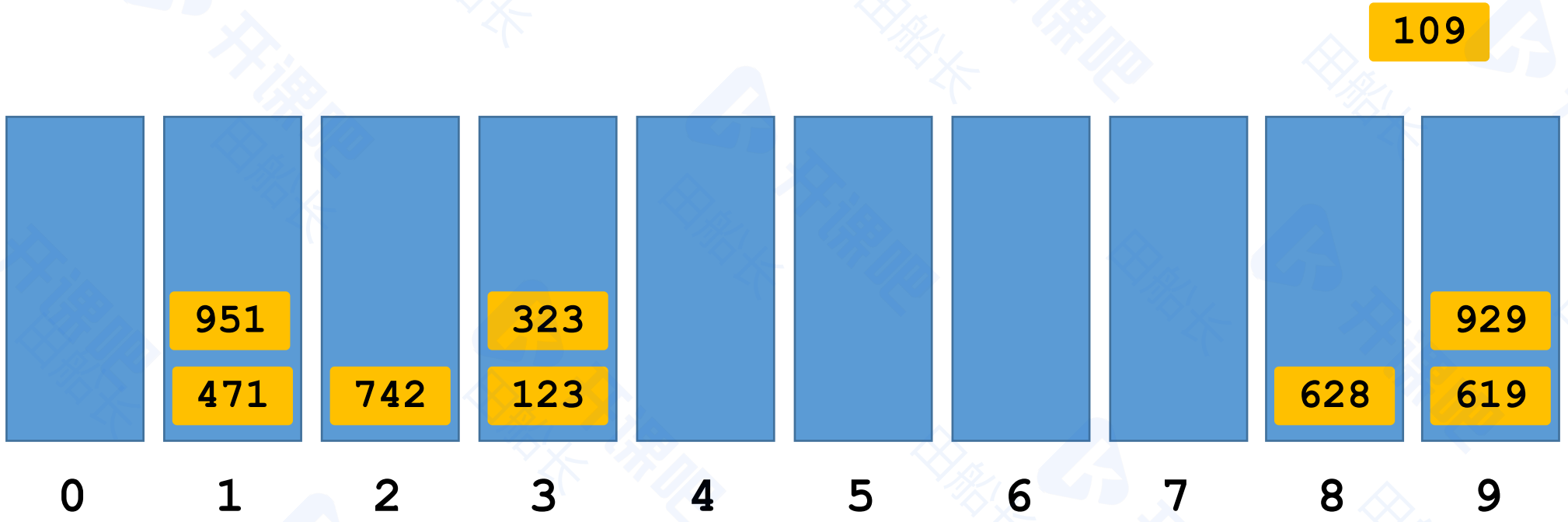
以个位组织进桶中



开课吧

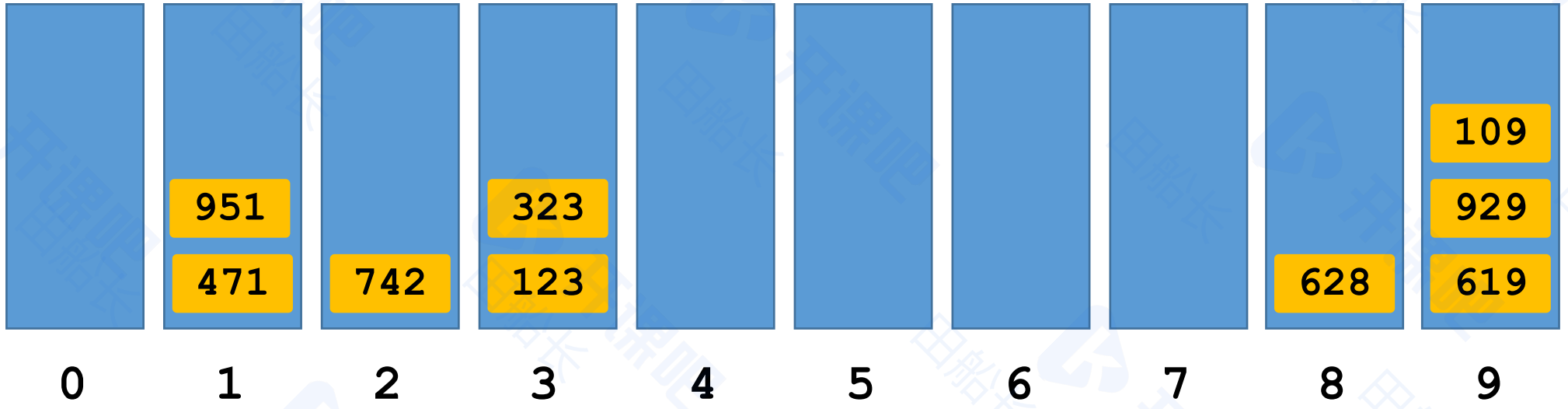
第一趟排序

以个位组织进桶中



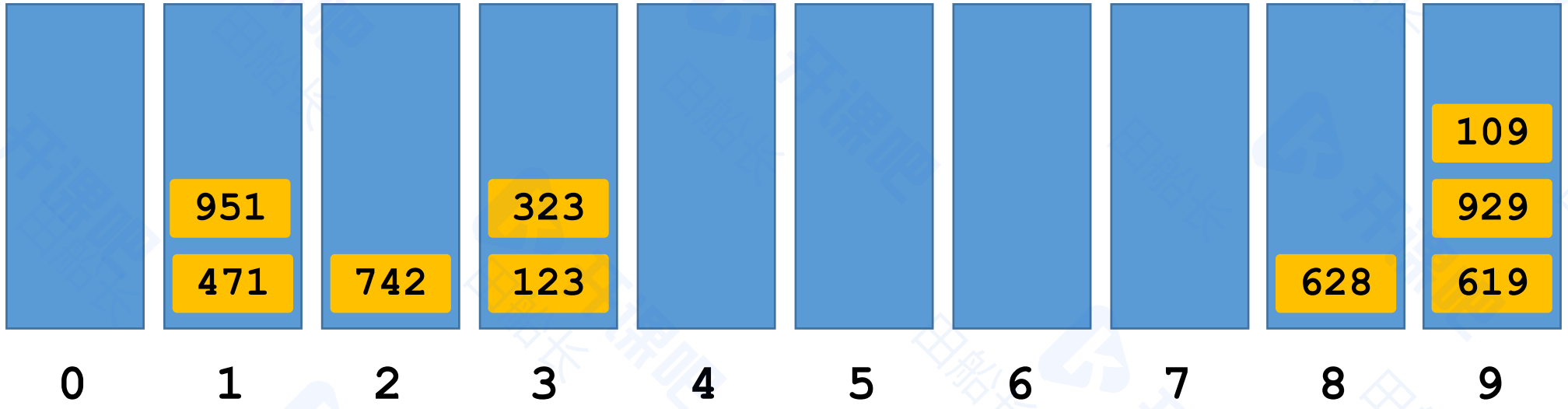
第一趟排序

以个位组织进桶中



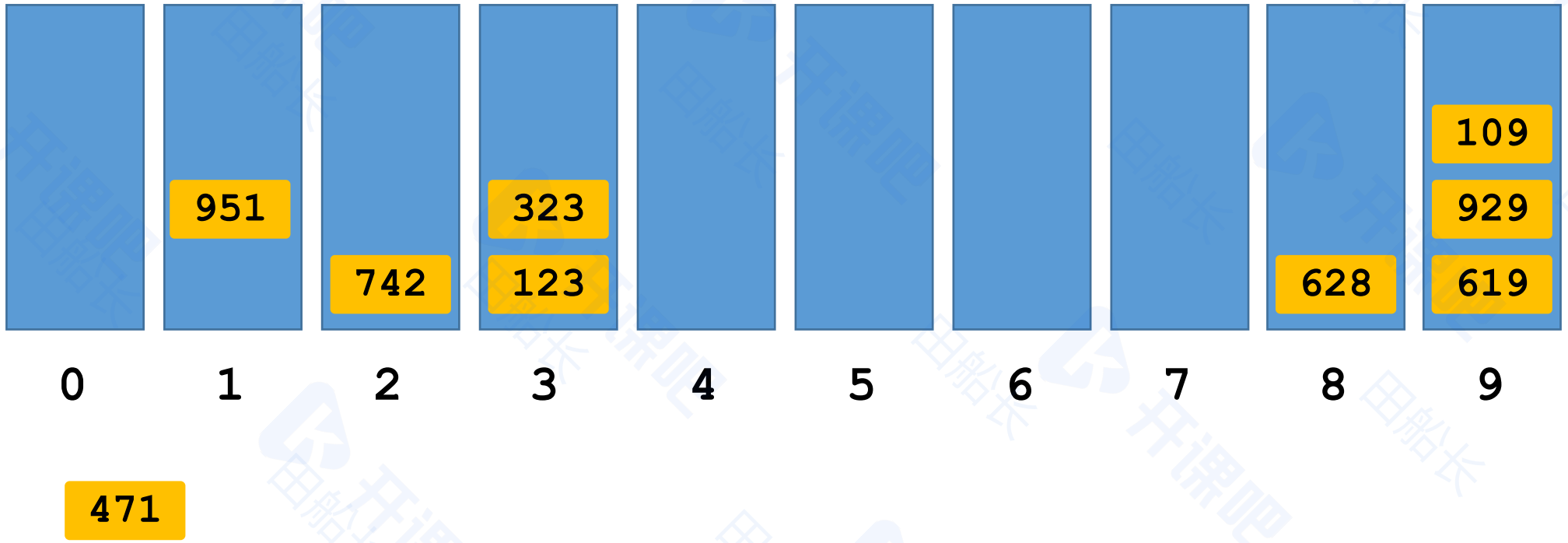
第一趟排序

按照先进先出的原则
将所有元素拿出



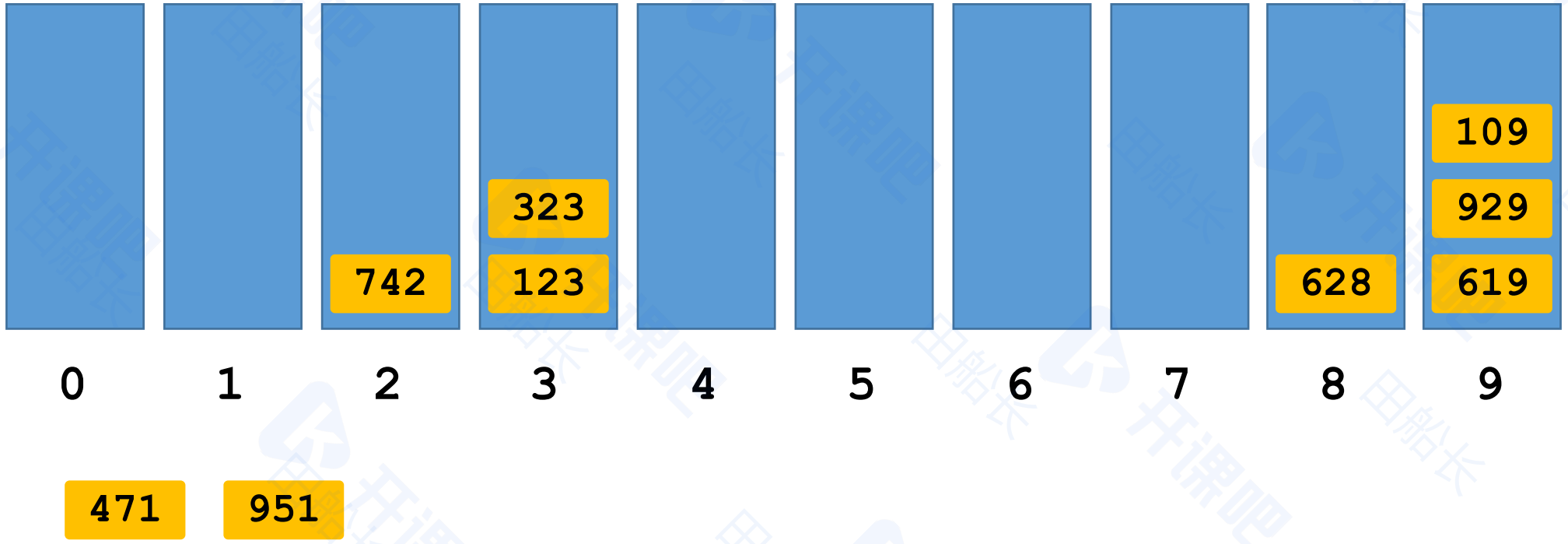
第一趟排序

按照先进先出的原则
将所有元素拿出



第一趟排序

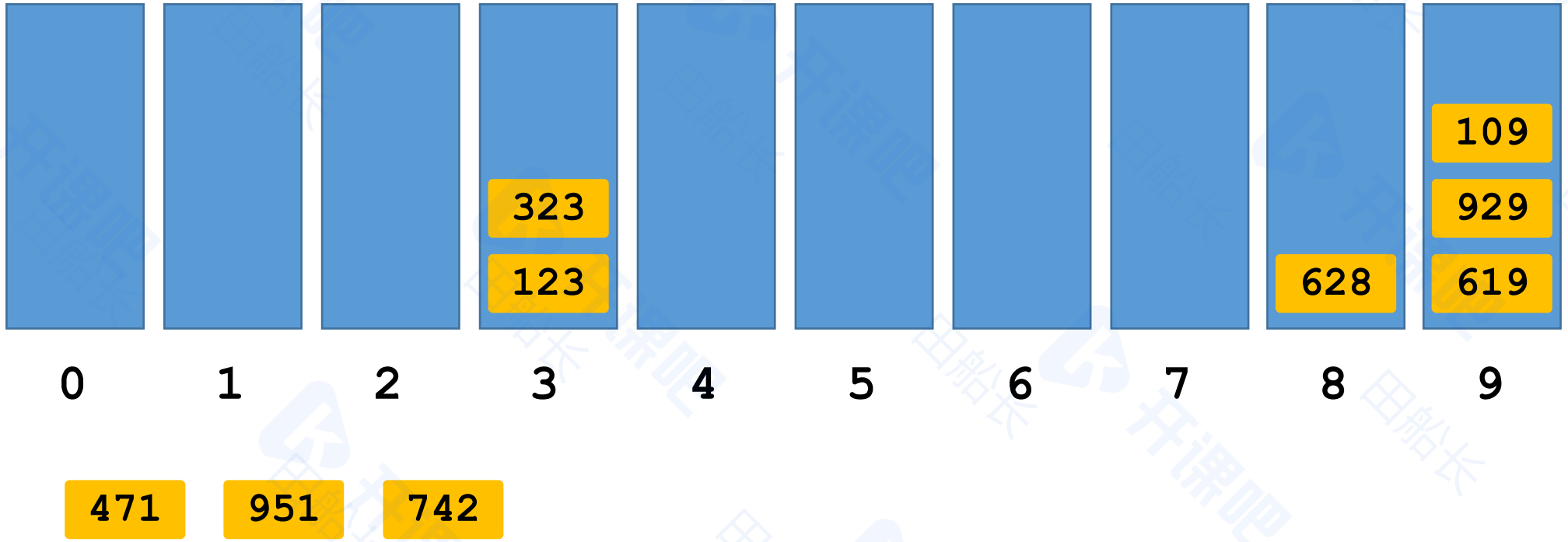
按照先进先出的原则
将所有元素拿出



开课吧

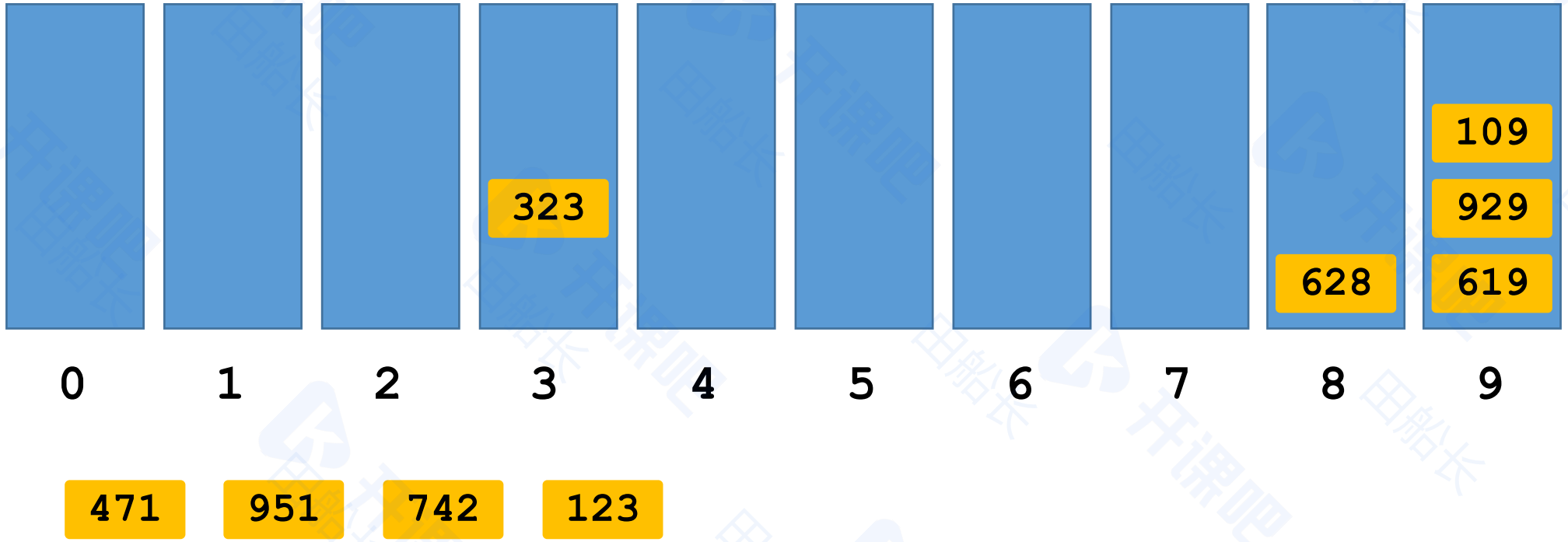
第一趟排序

按照先进先出的原则
将所有元素拿出



第一趟排序

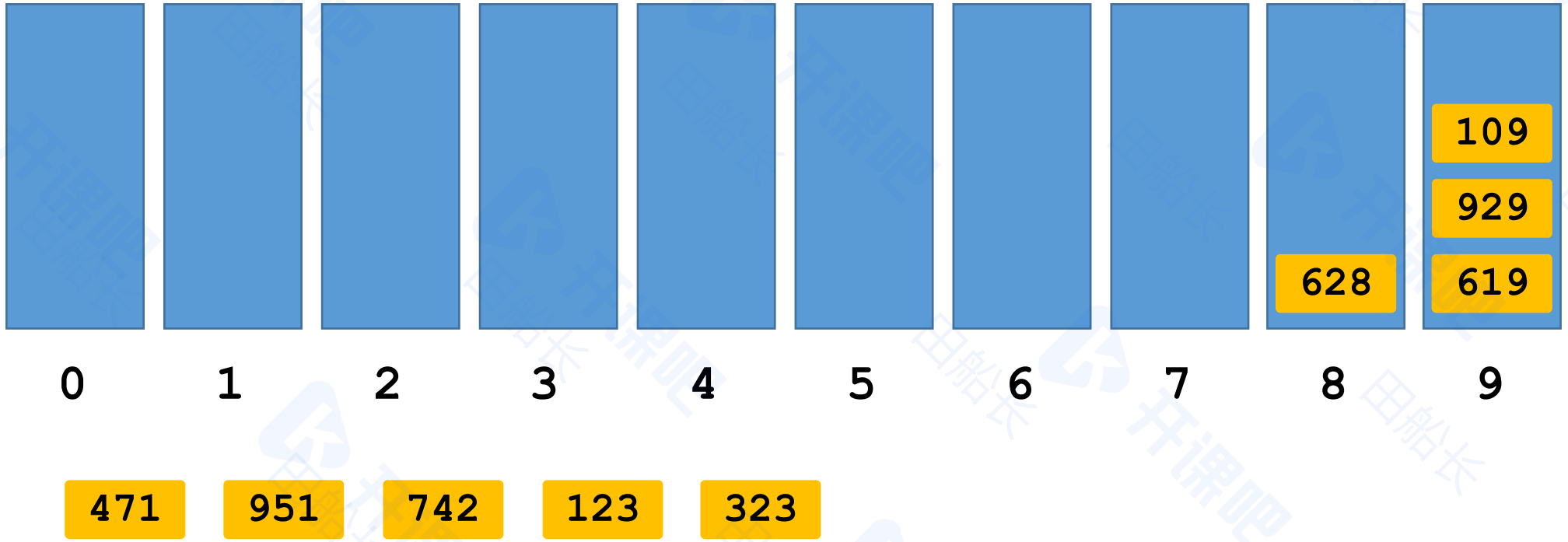
按照先进先出的原则
将所有元素拿出



开课吧

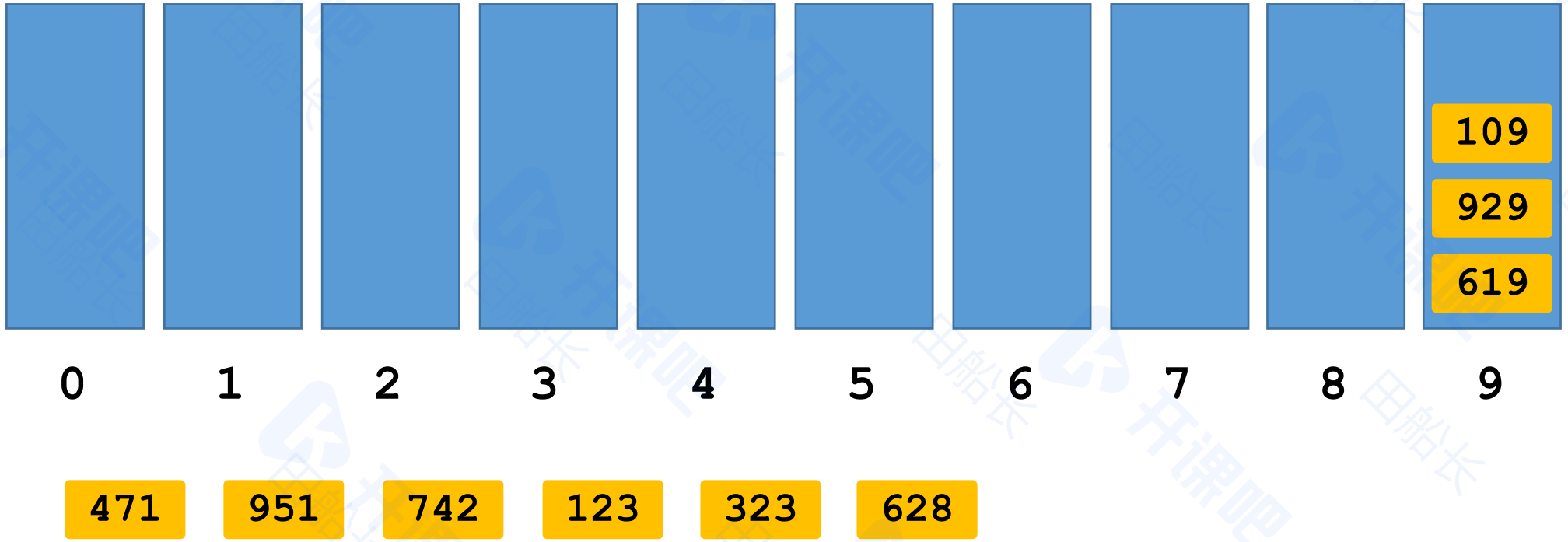
第一趟排序

按照先进先出的原则
将所有元素拿出



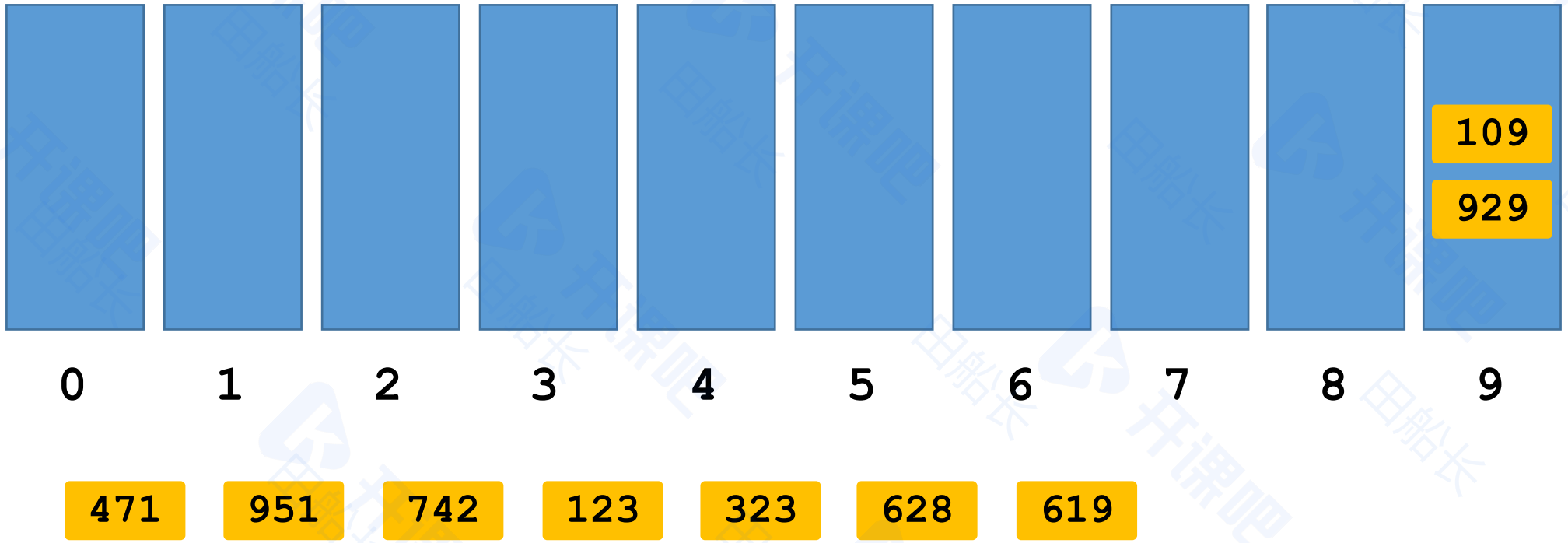
第一趟排序

按照先进先出的原则
将所有元素拿出



第一趟排序

按照先进先出的原则
将所有元素拿出



开课吧

第一趟排序

按照先进先出的原则
将所有元素拿出

471 951 742 123 323 628 619 929

第一趟排序

按照先进先出的原则
将所有元素拿出

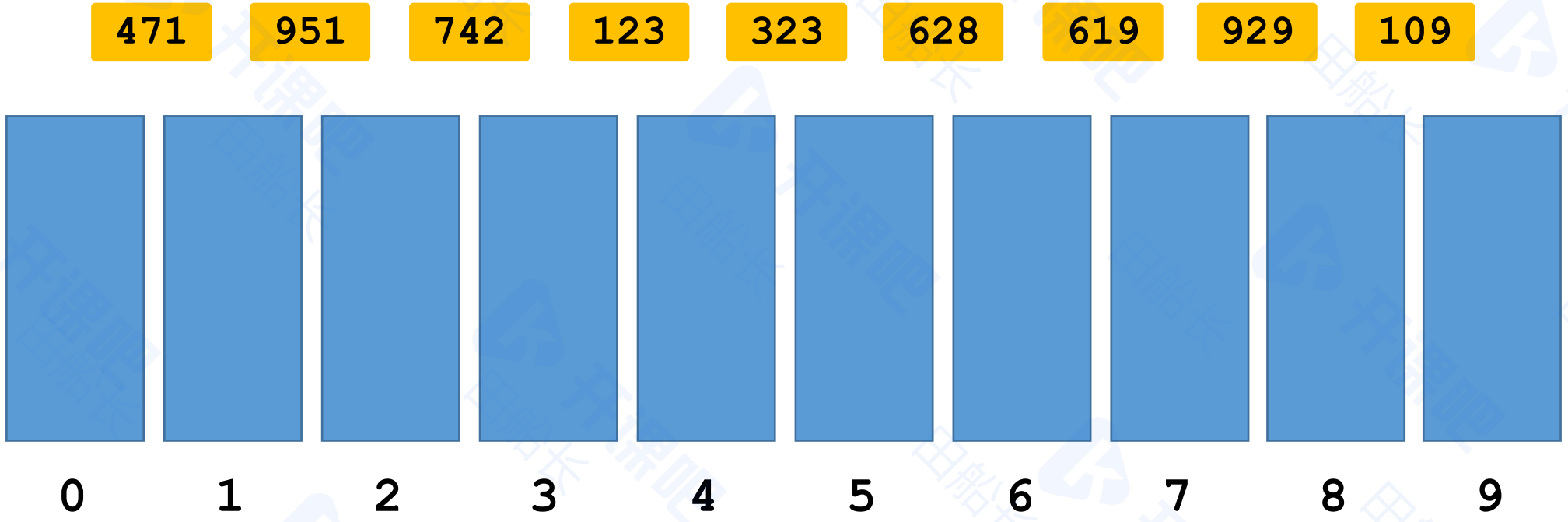
第一趟排序结束

所有元素已按照个位排好序

开课吧

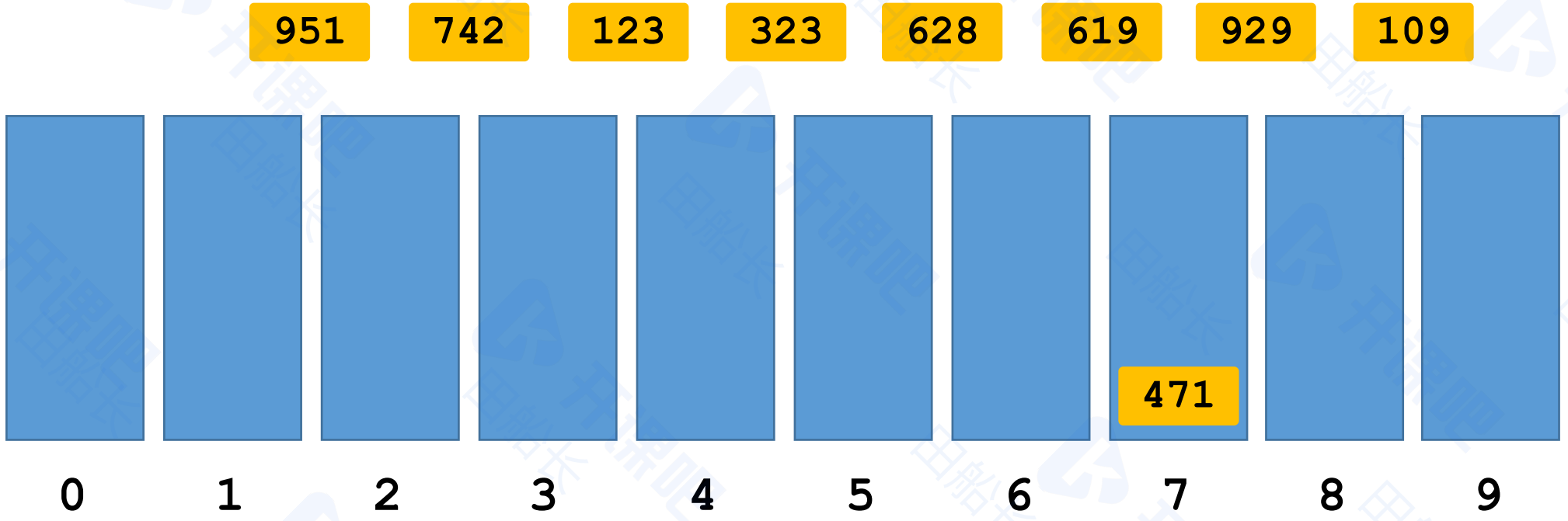
第二趟排序

以十位组织进桶中



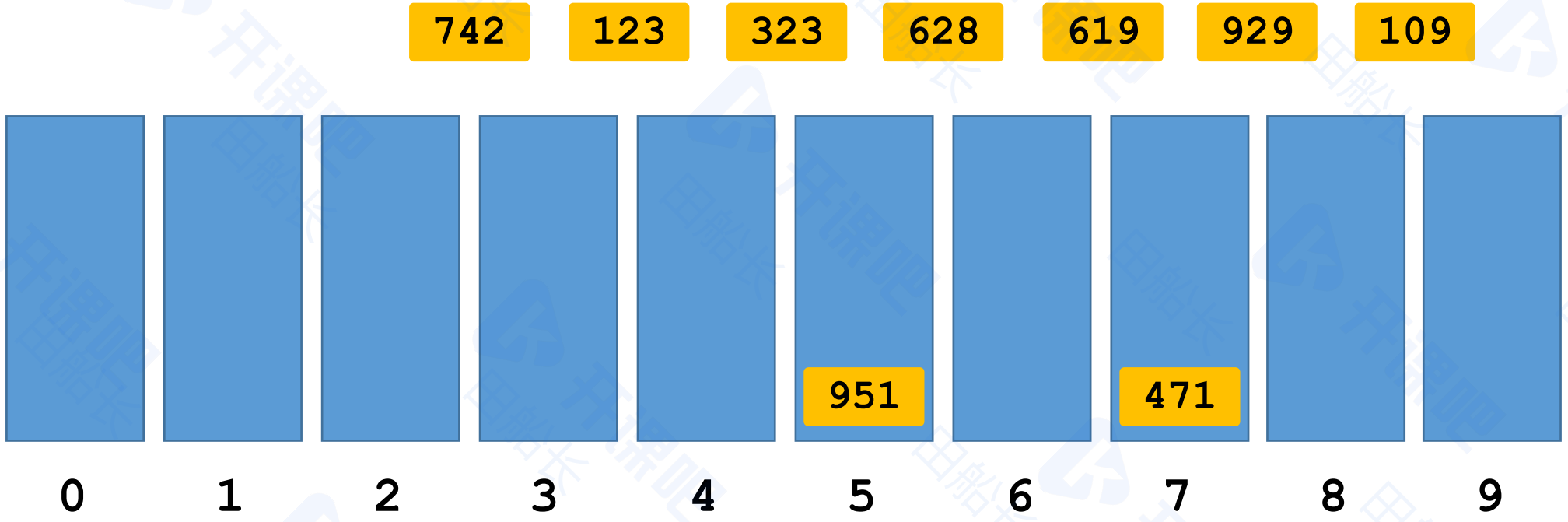
第二趟排序

以十位组织进桶中



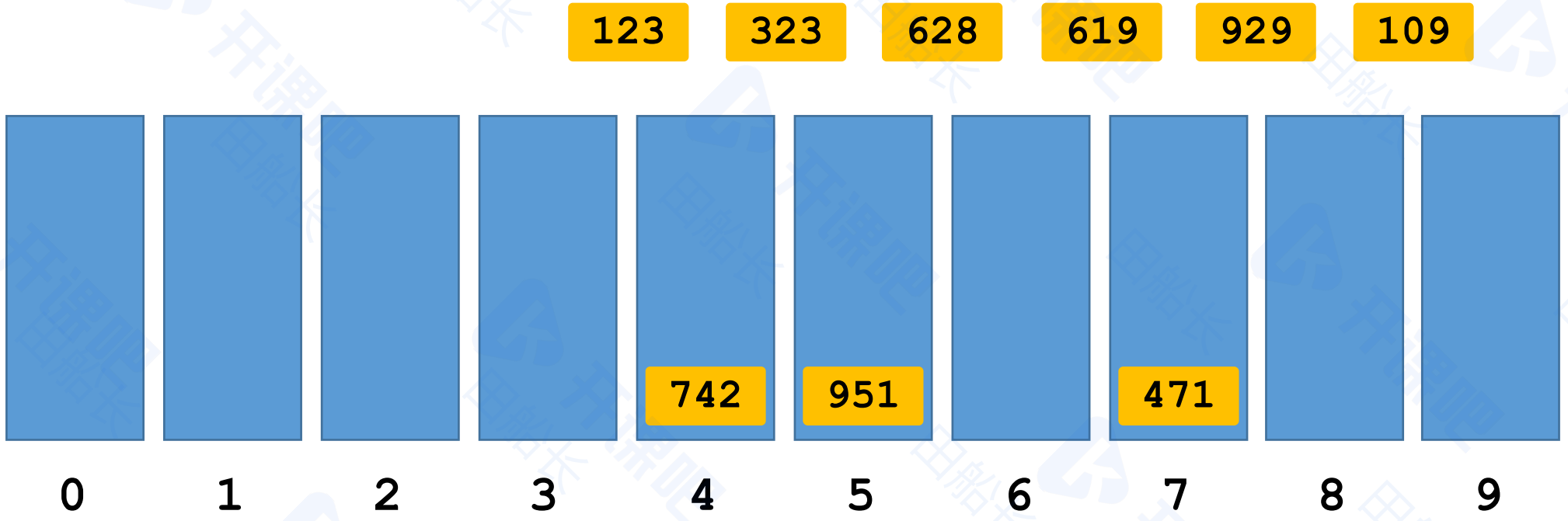
第二趟排序

以十位组织进桶中



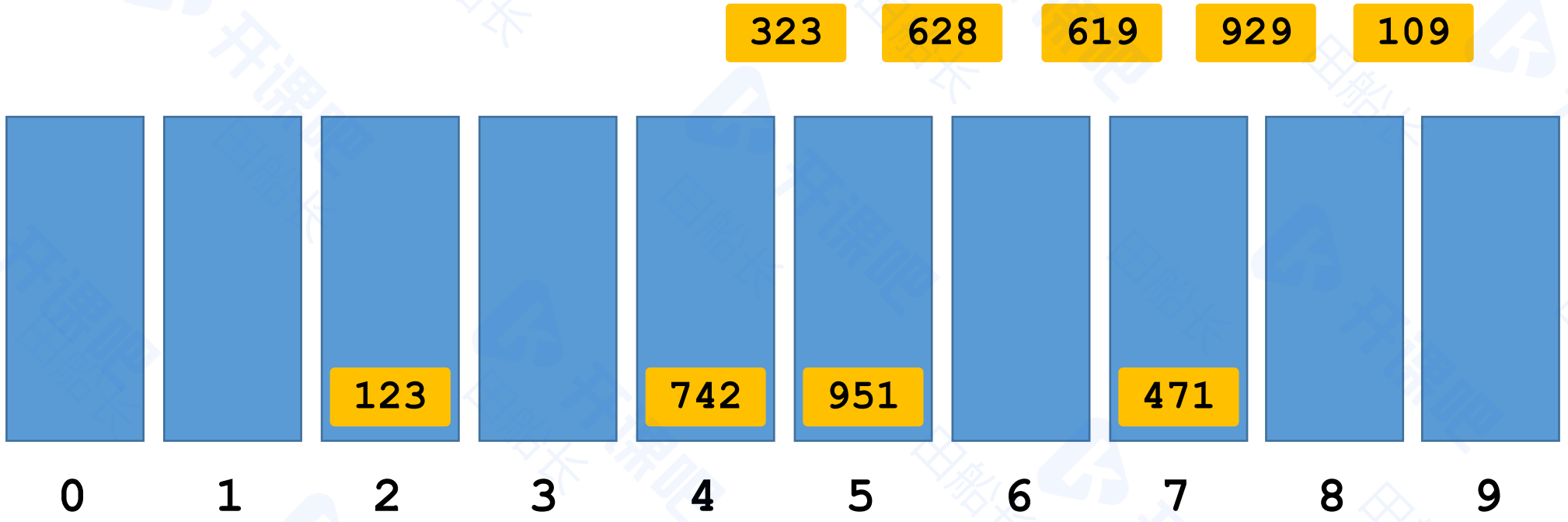
第二趟排序

以十位组织进桶中



第二趟排序

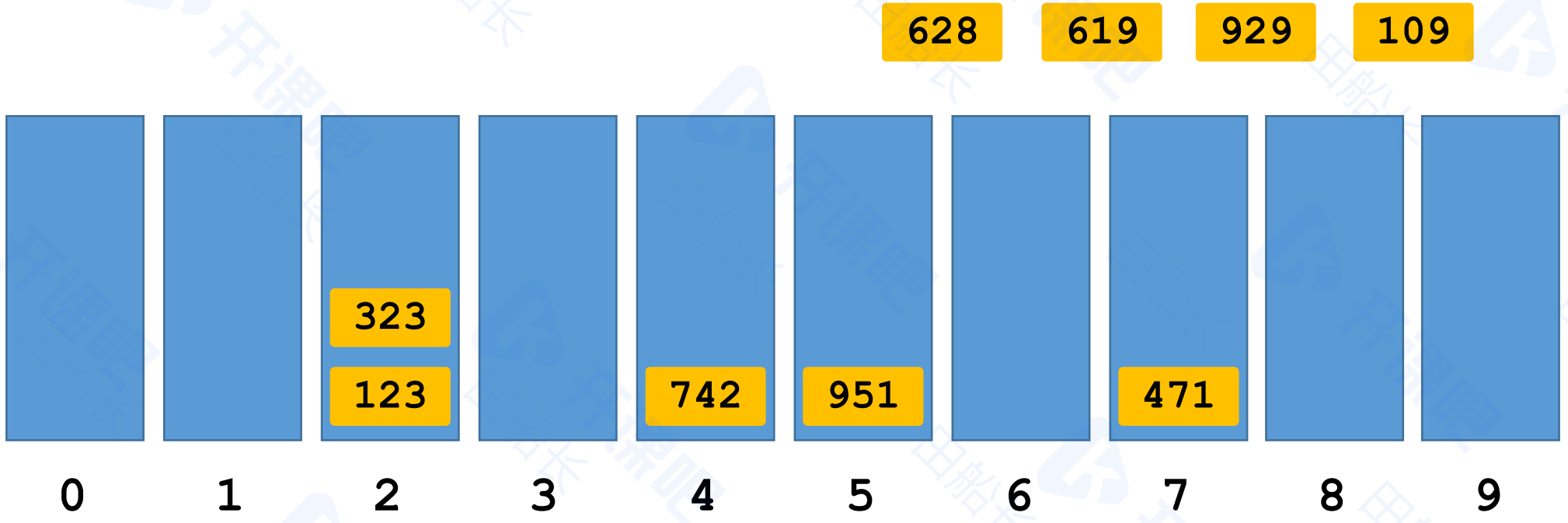
以十位组织进桶中



开课吧

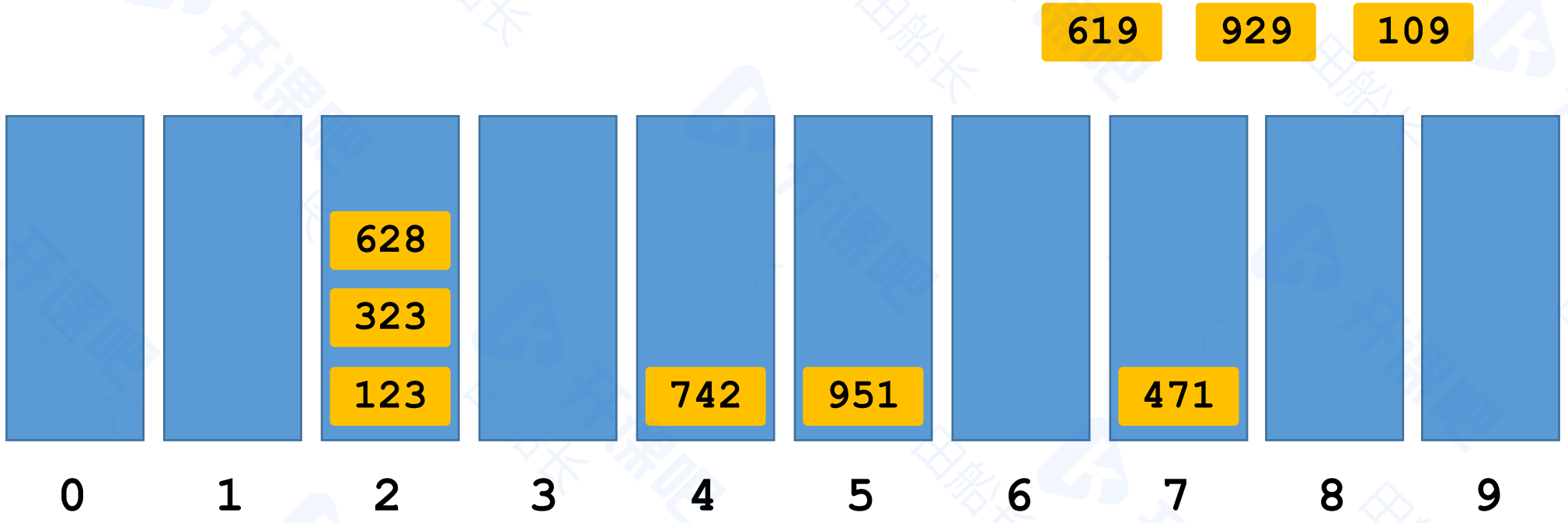
第二趟排序

以十位组织进桶中



第二趟排序

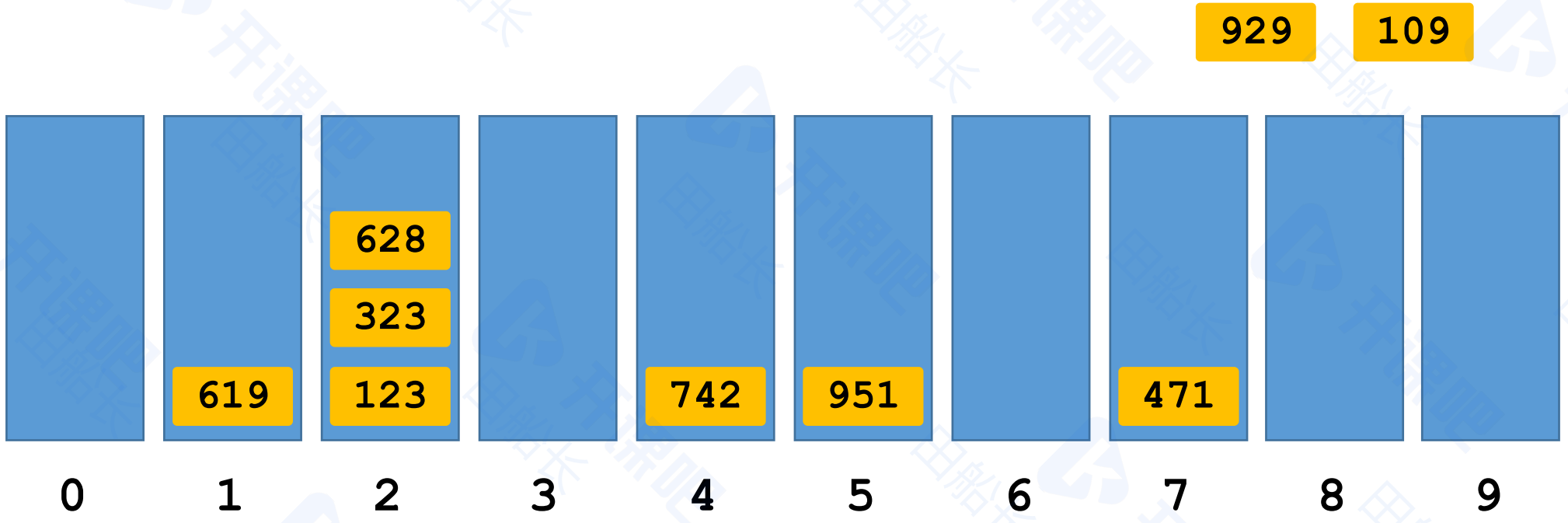
以十位组织进桶中



开课吧

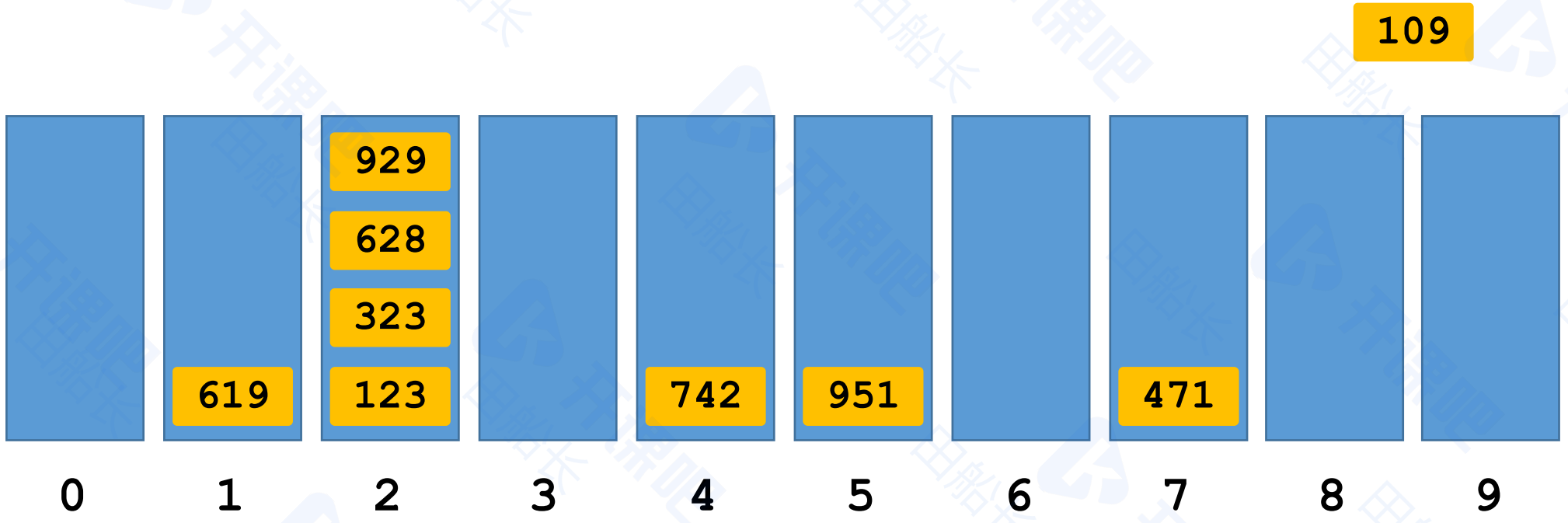
第二趟排序

以十位组织进桶中



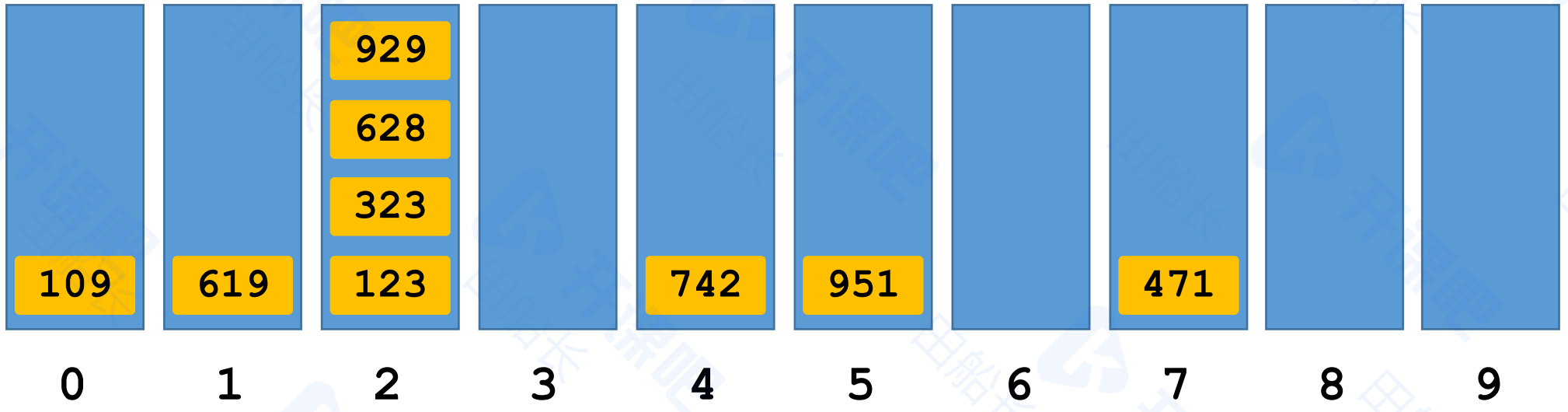
第二趟排序

以十位组织进桶中



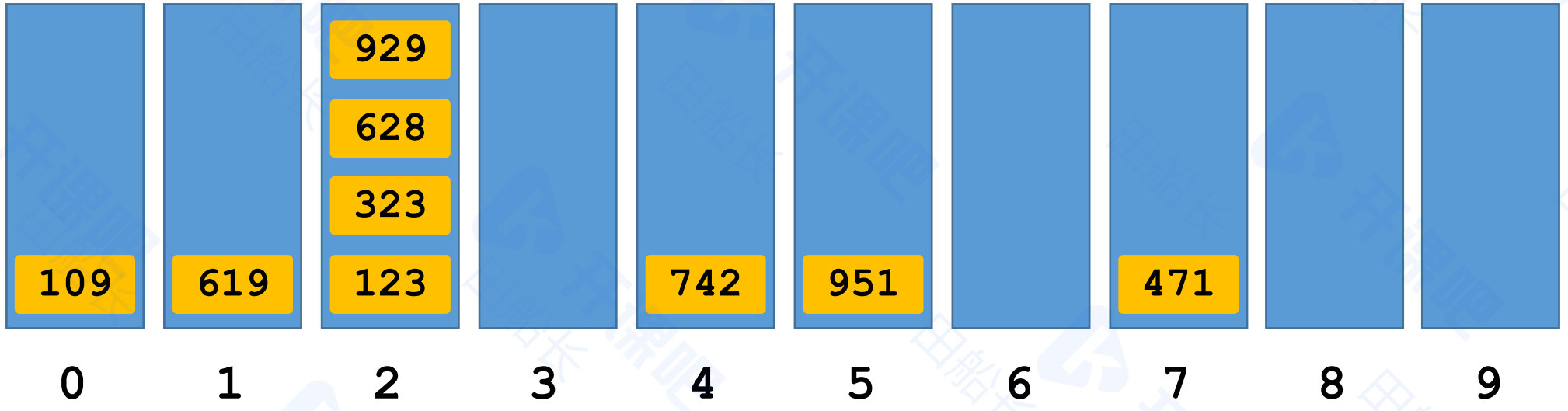
第二趟排序

以十位组织进桶中



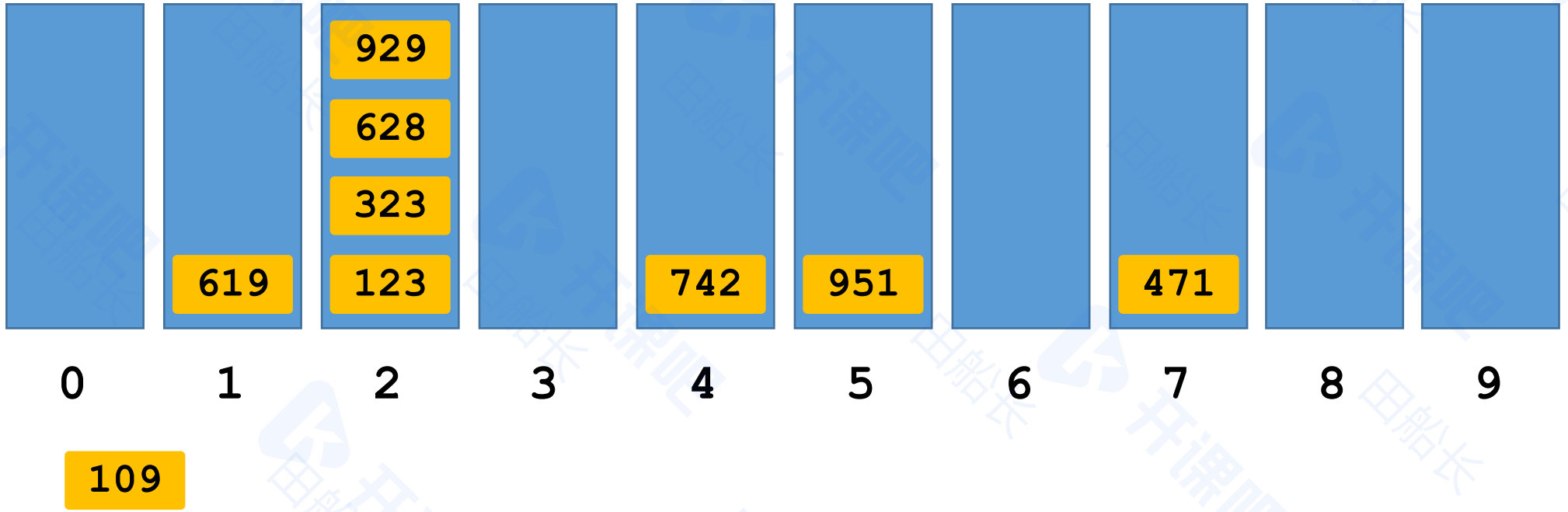
第二趟排序

按照先进先出的原则
将所有元素拿出



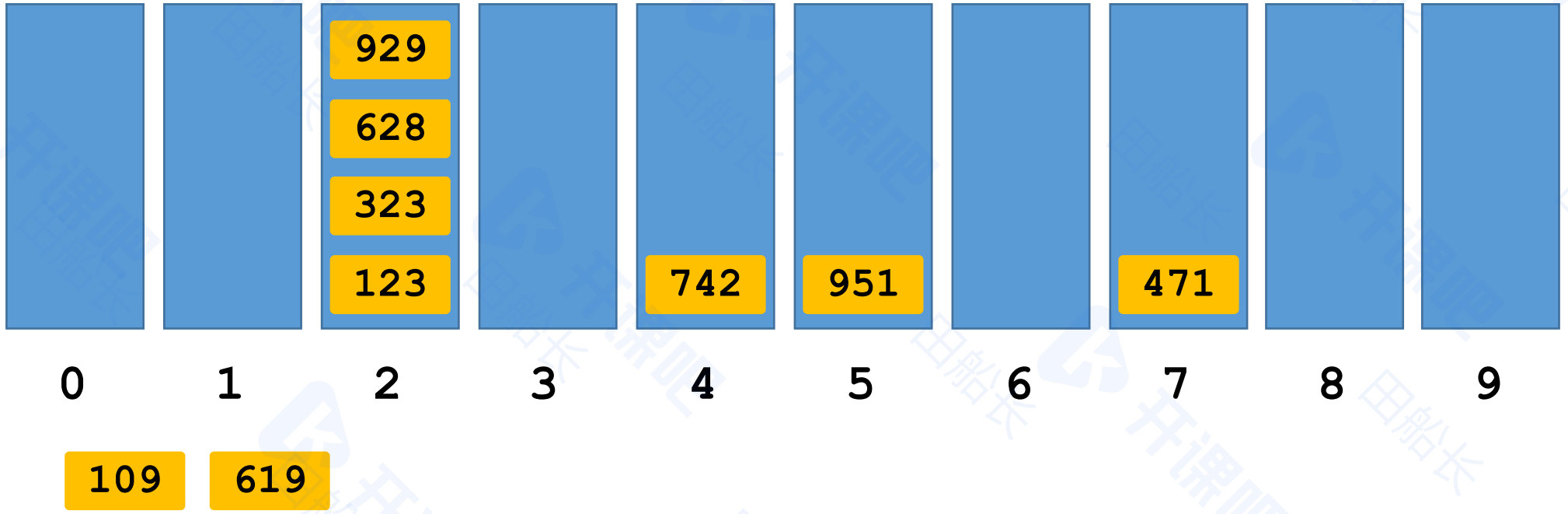
第二趟排序

按照先进先出的原则
将所有元素拿出



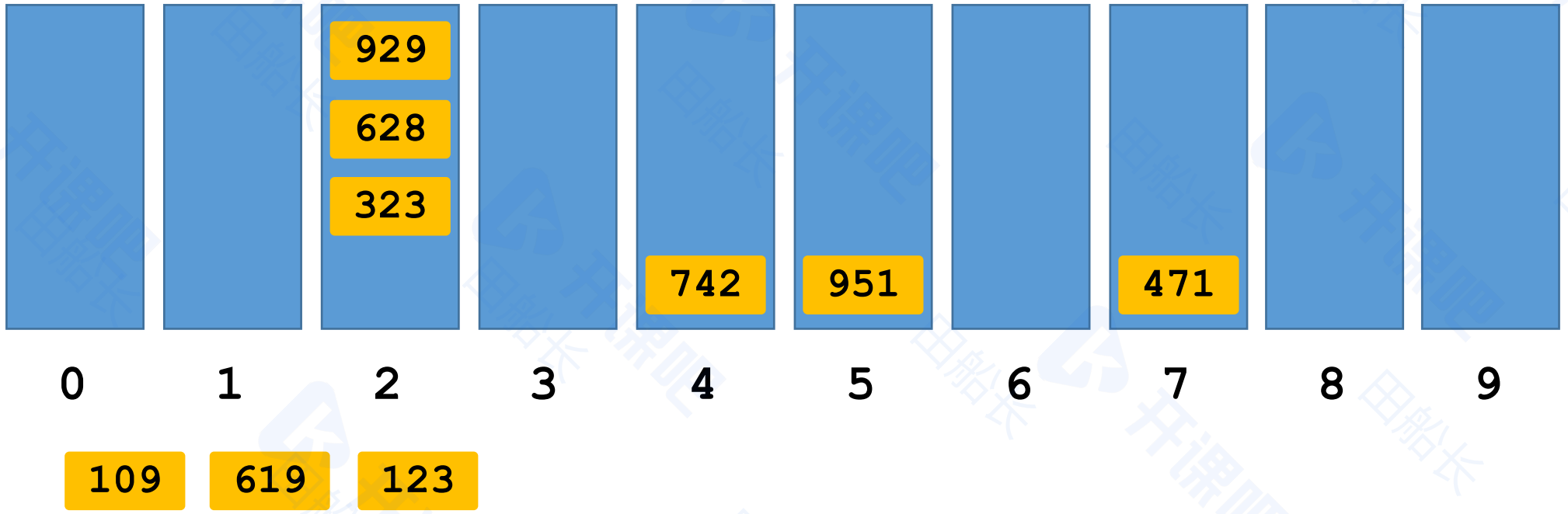
第二趟排序

按照先进先出的原则
将所有元素拿出



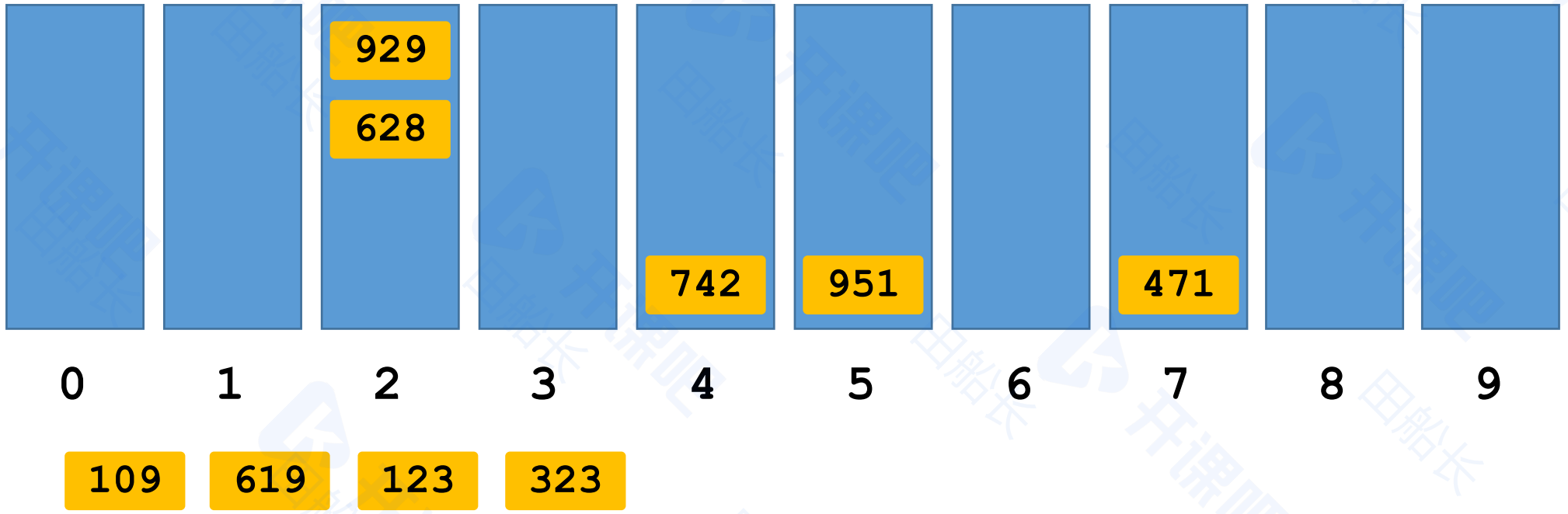
第二趟排序

按照先进先出的原则
将所有元素拿出



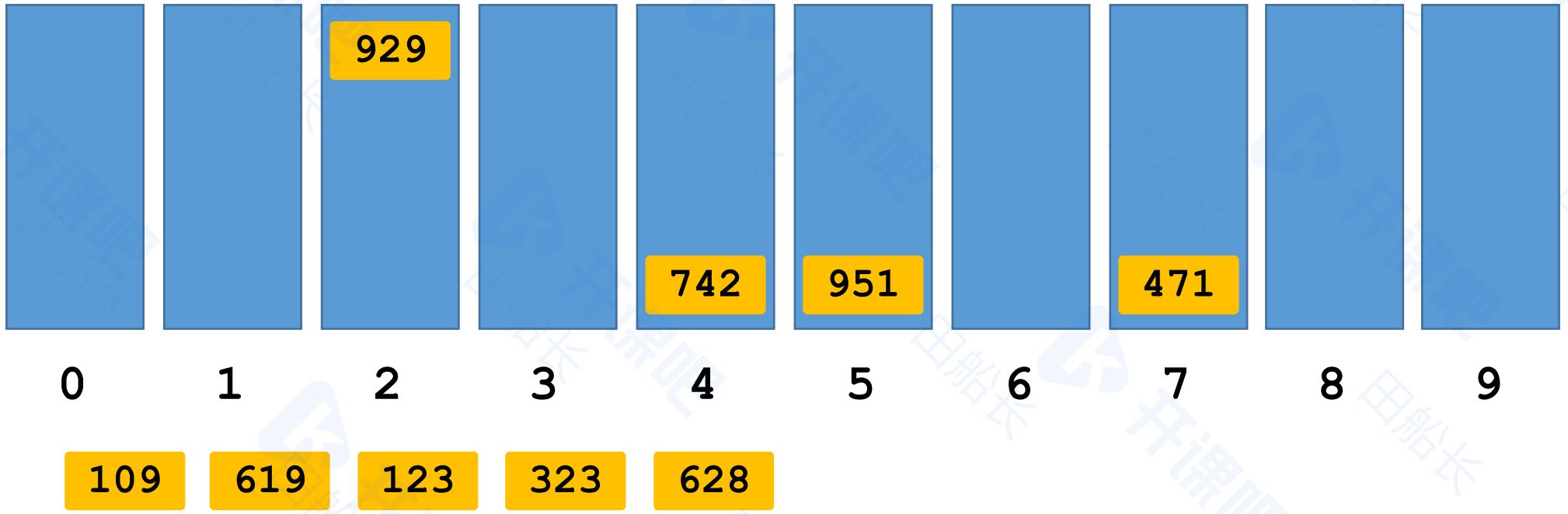
第二趟排序

按照先进先出的原则
将所有元素拿出



第二趟排序

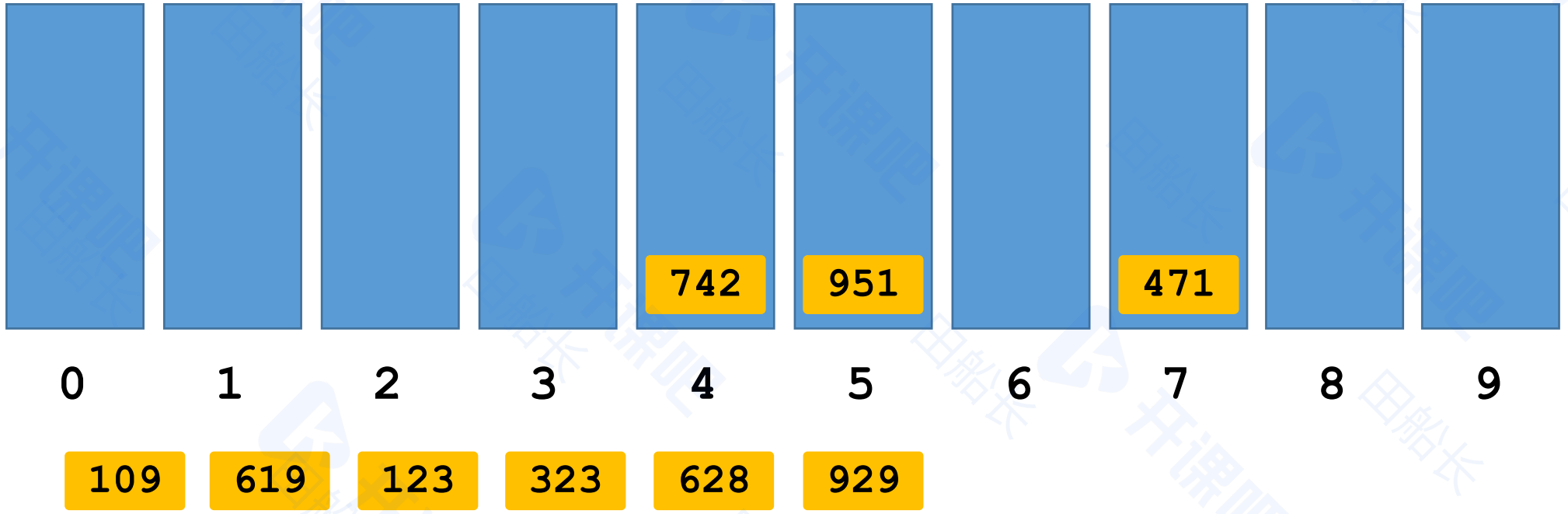
按照先进先出的原则
将所有元素拿出



开课吧

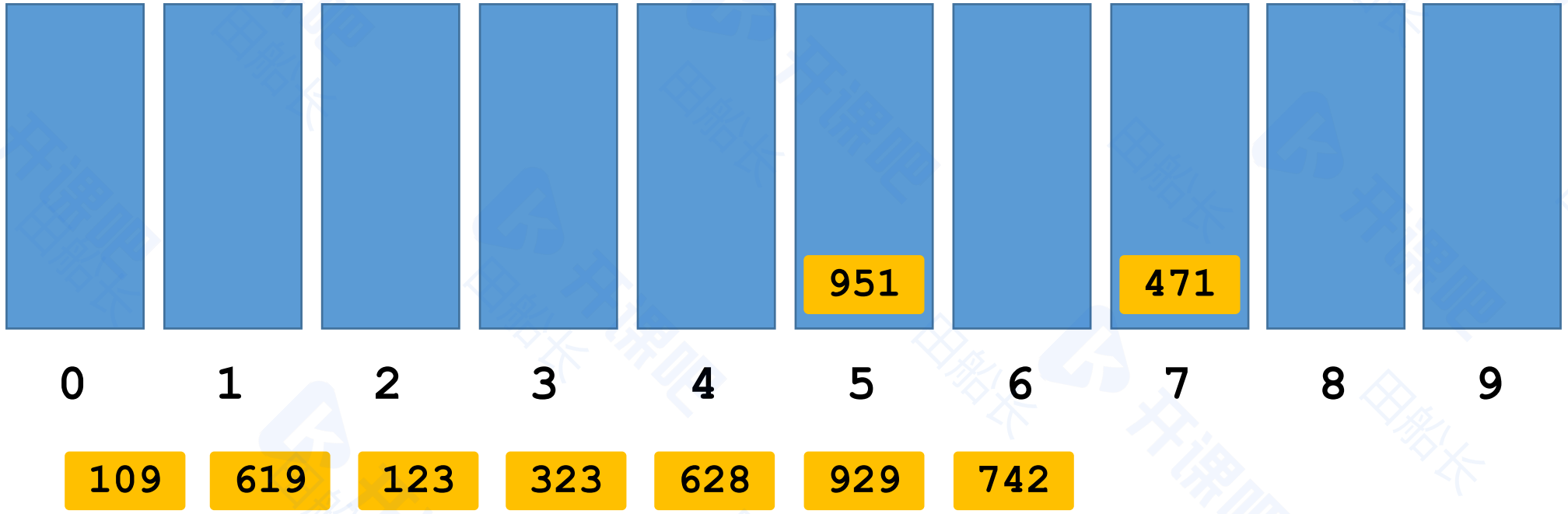
第二趟排序

按照先进先出的原则
将所有元素拿出



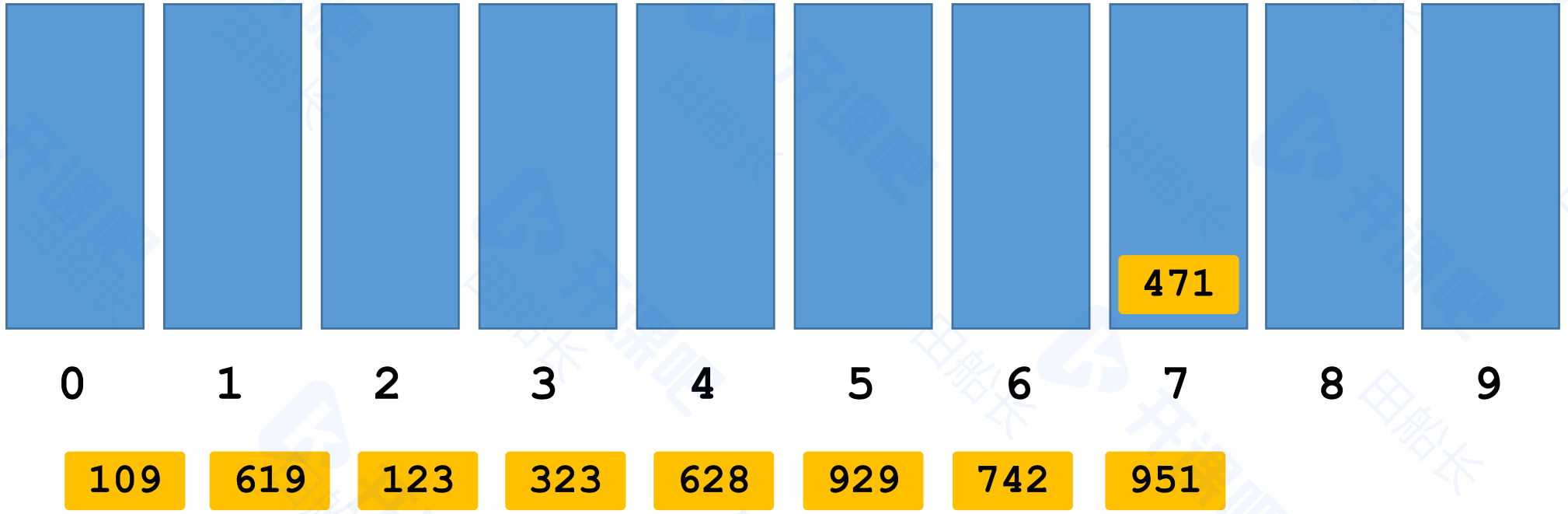
第二趟排序

按照先进先出的原则
将所有元素拿出



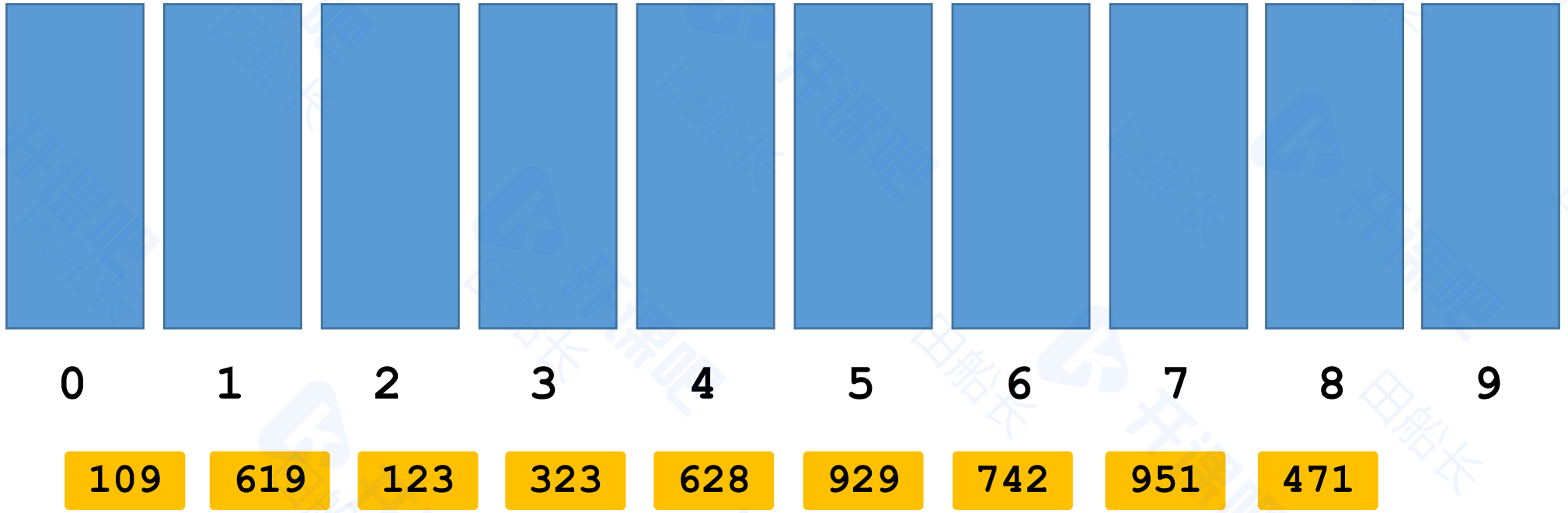
第二趟排序

按照先进先出的原则
将所有元素拿出



第二趟排序

按照先进先出的原则
将所有元素拿出



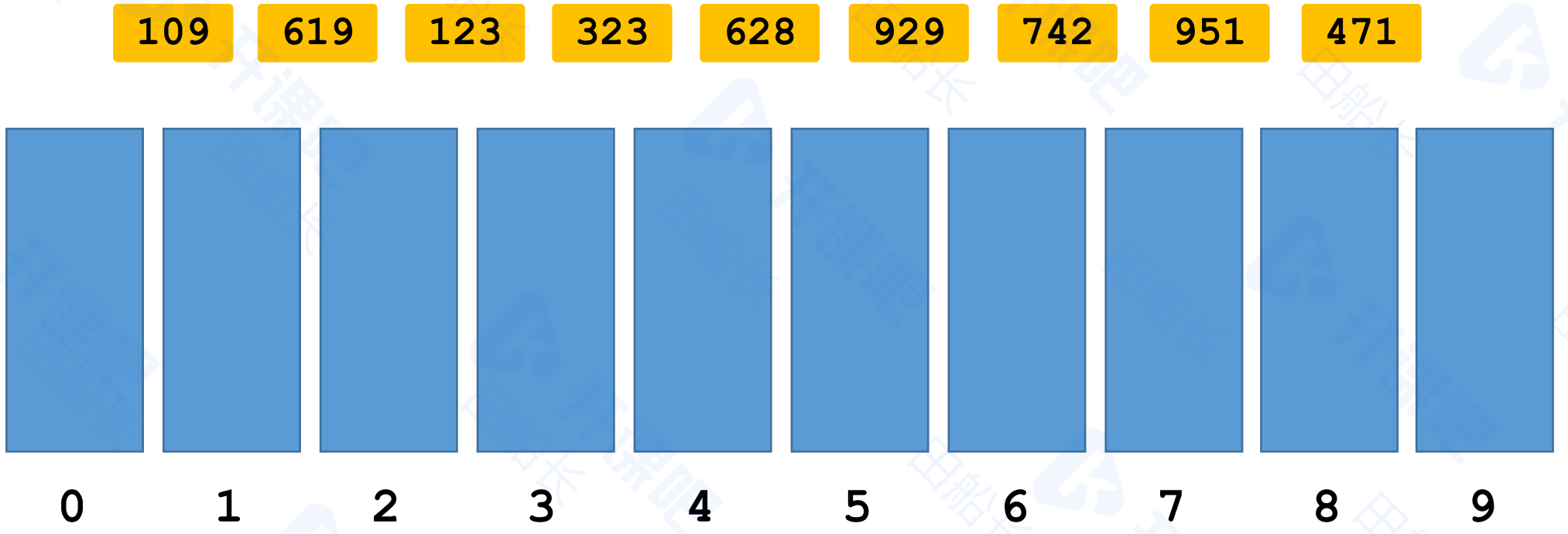
第二趟排序结束

所有元素已按照十位排好序
十位相同的元素按照个位排序

0	1	2	3	4	5	6	7	8	9
109	619	123	323	628	929	742	951	471	

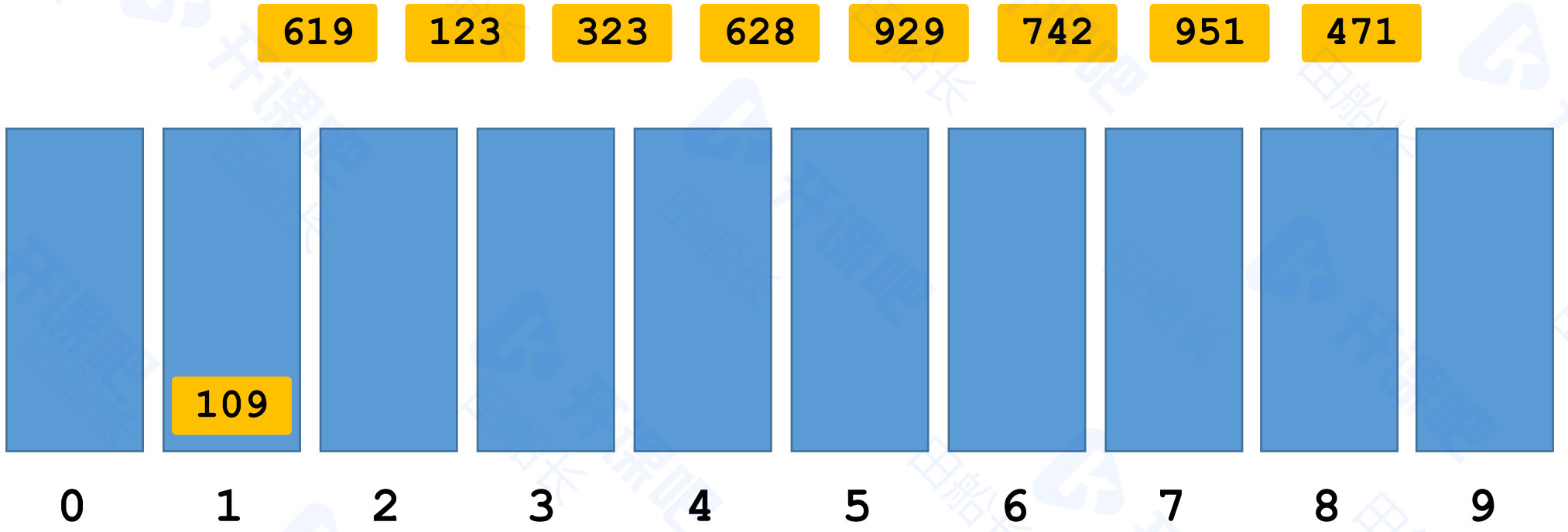
第三趟排序

以百位组织进桶中



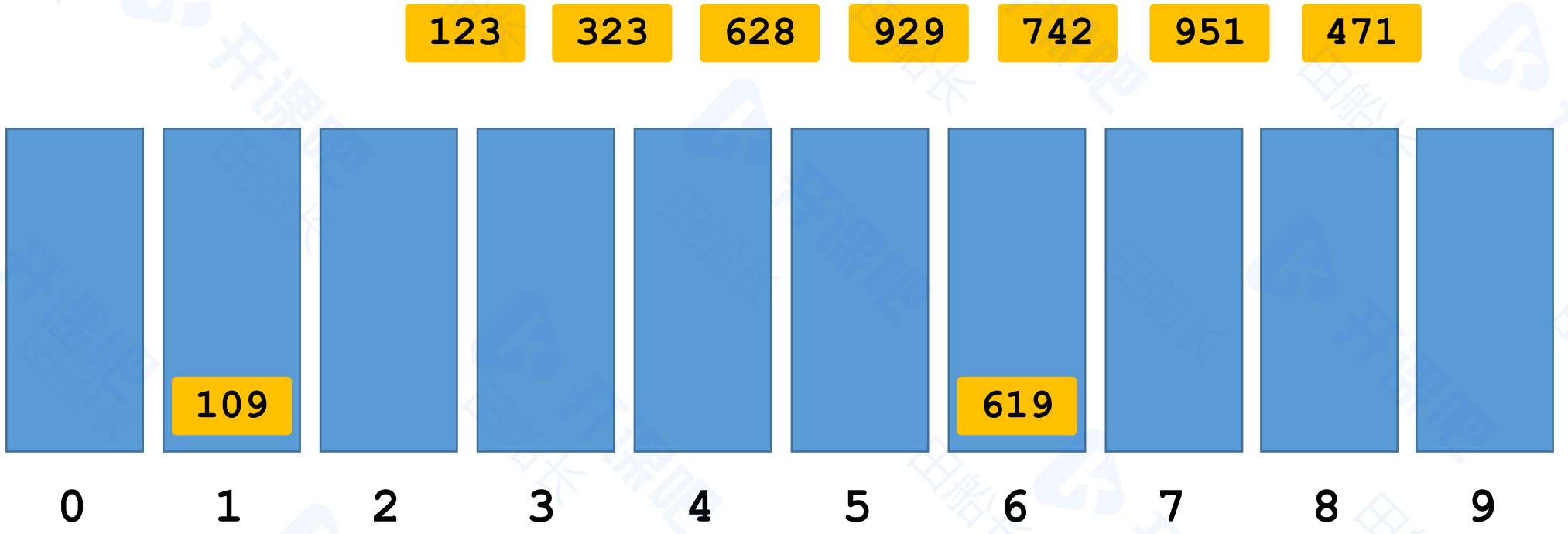
第三趟排序

以百位组织进桶中



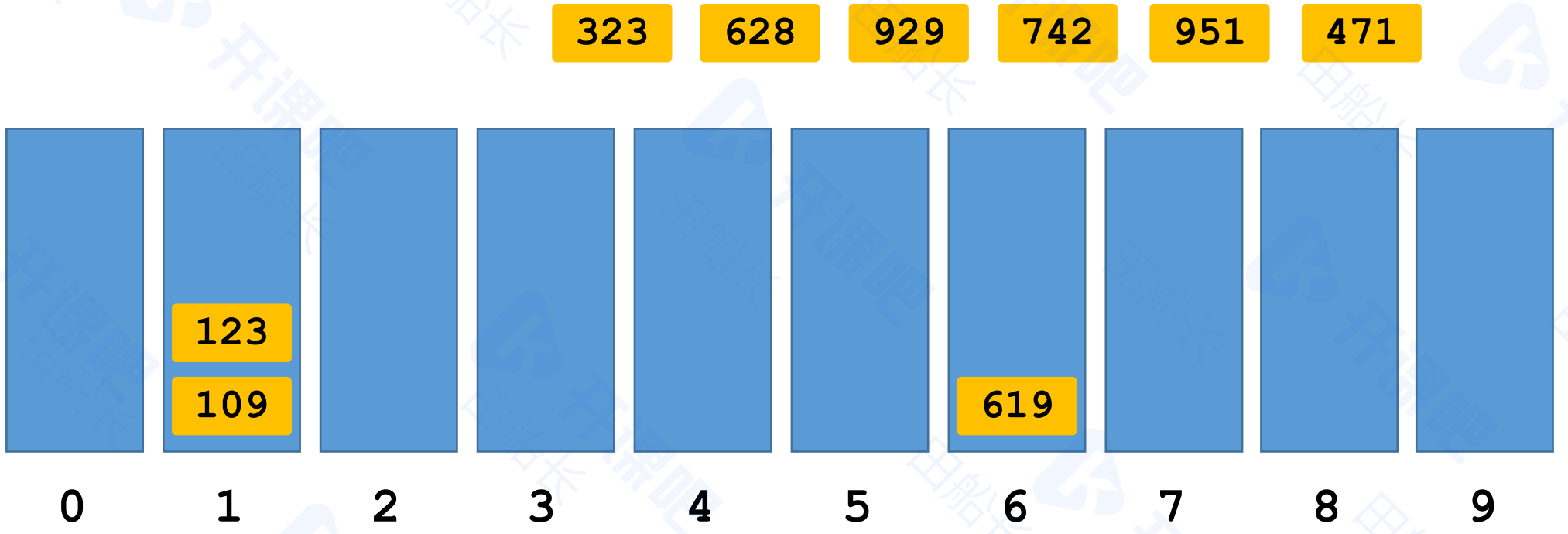
第三趟排序

以百位组织进桶中



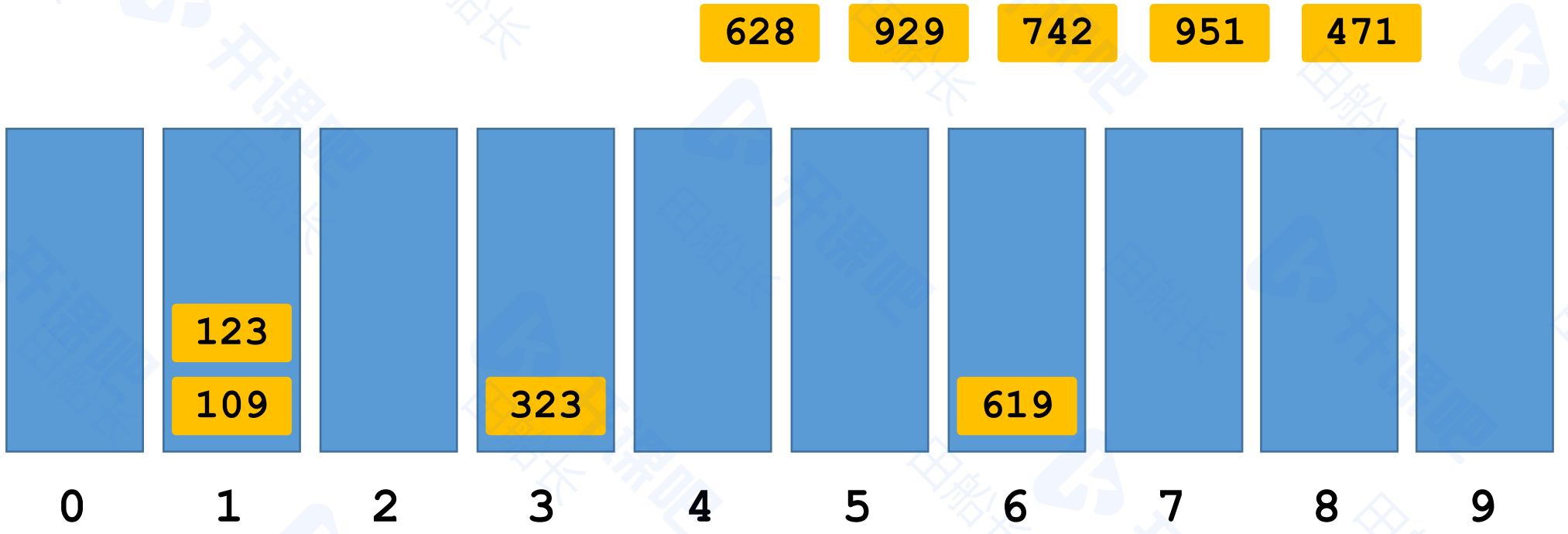
第三趟排序

以百位组织进桶中



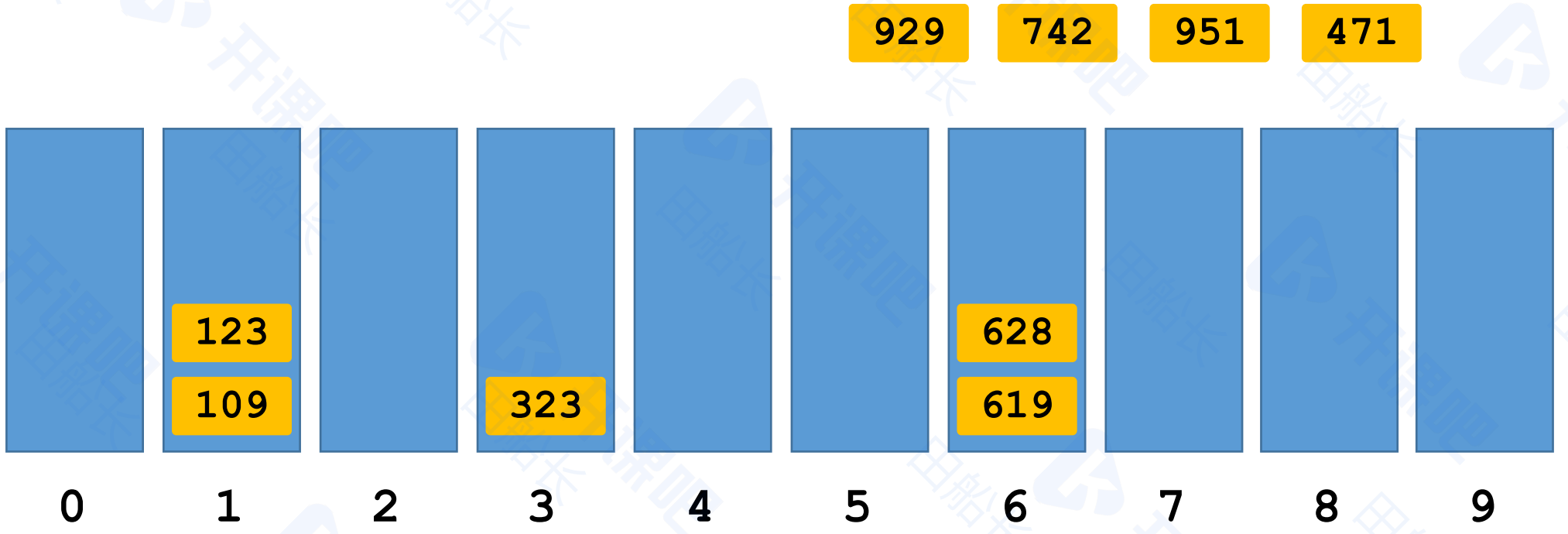
第三趟排序

以百位组织进桶中



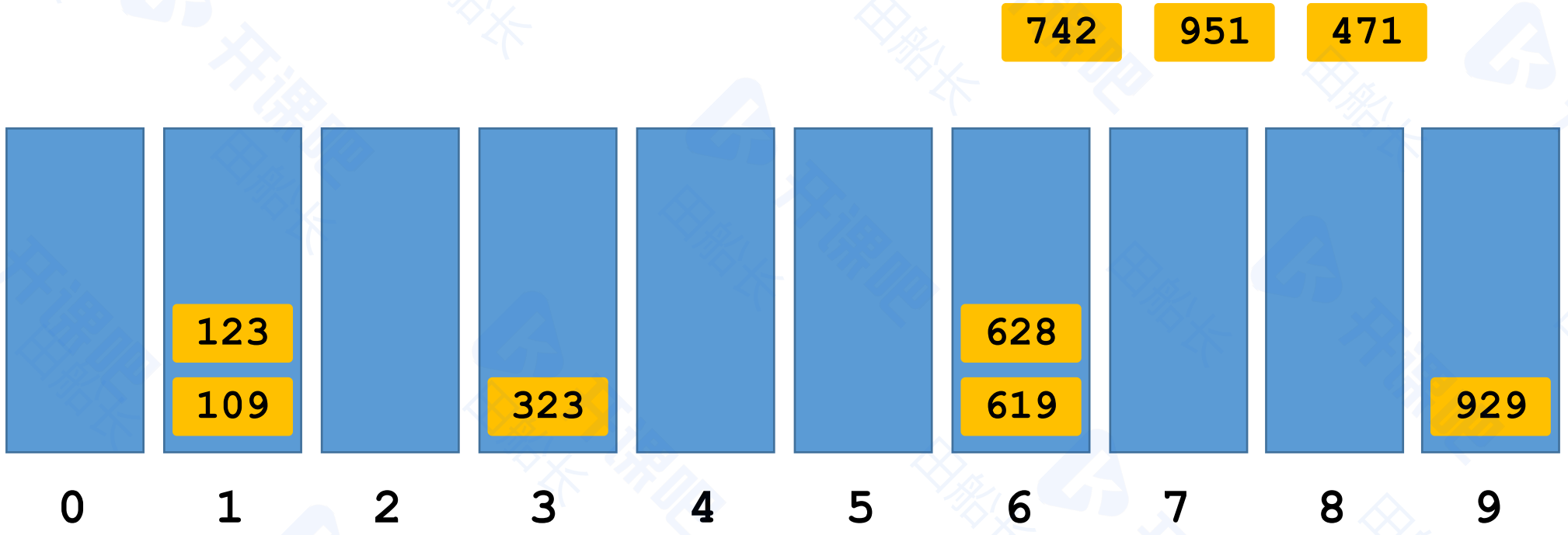
第三趟排序

以百位组织进桶中



第三趟排序

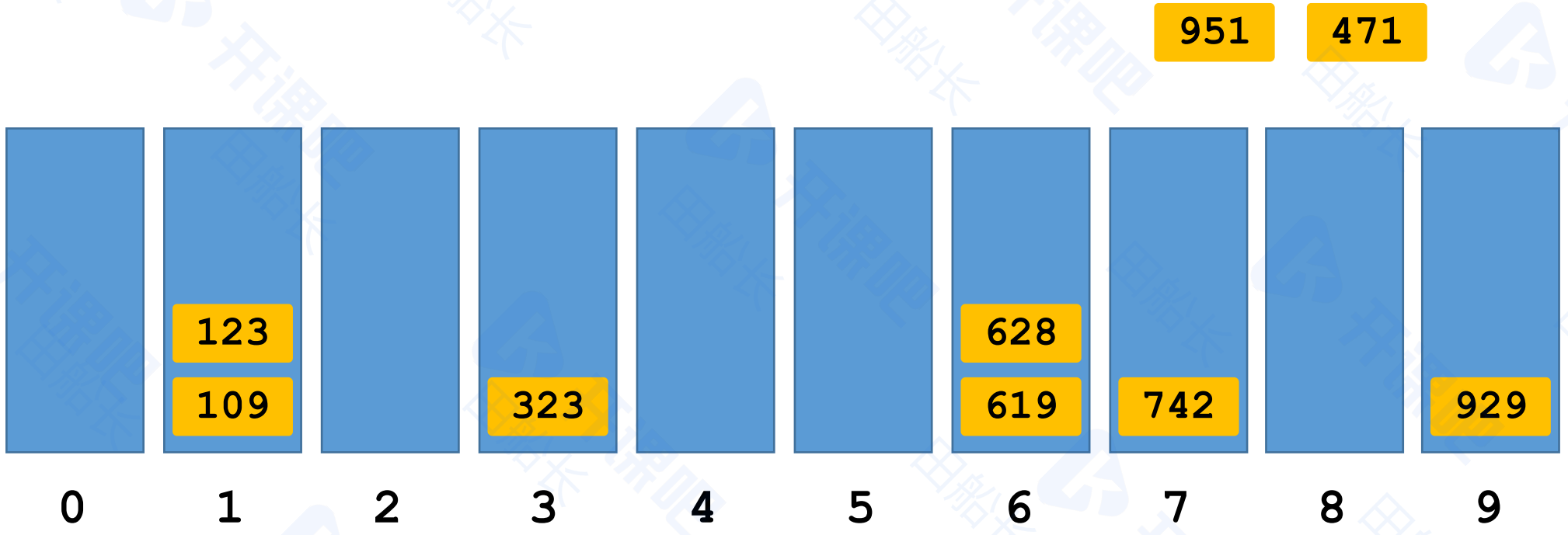
以百位组织进桶中



开课吧

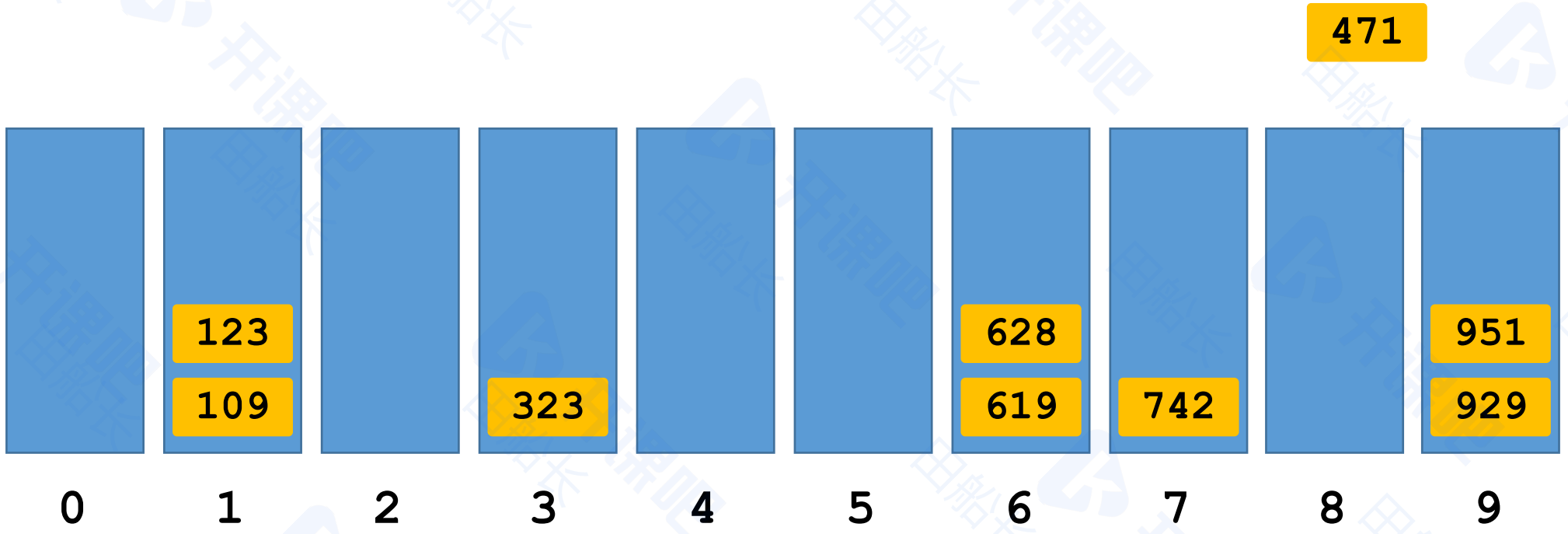
第三趟排序

以百位组织进桶中



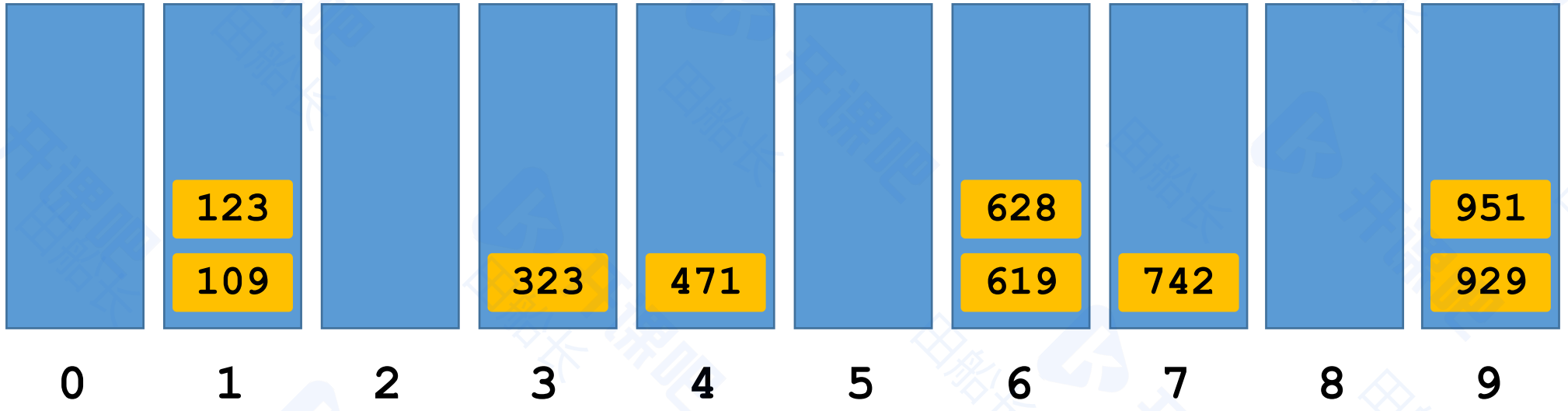
第三趟排序

以百位组织进桶中



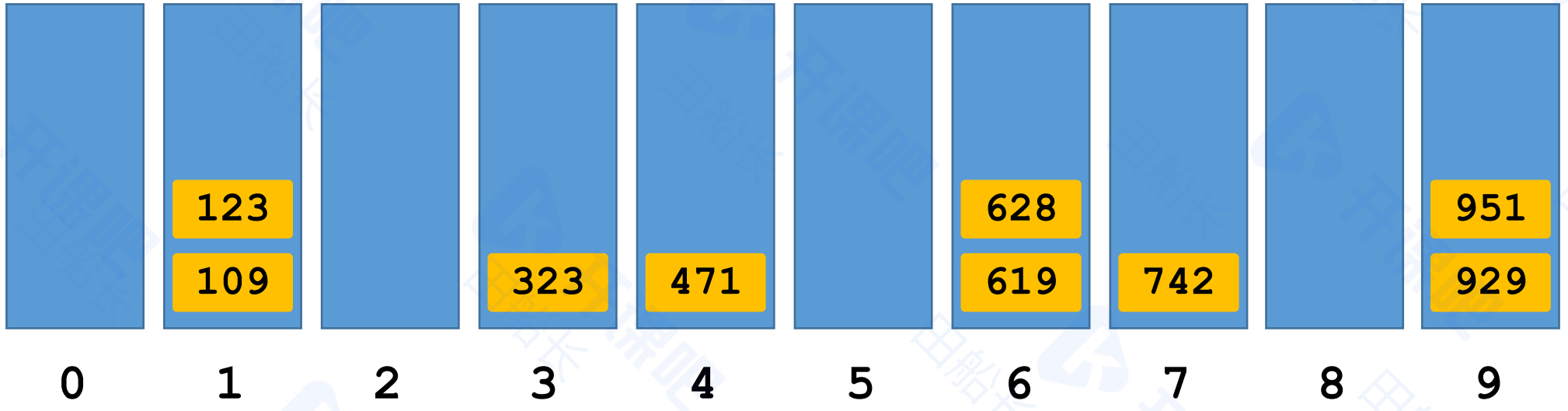
第三趟排序

以百位组织进桶中



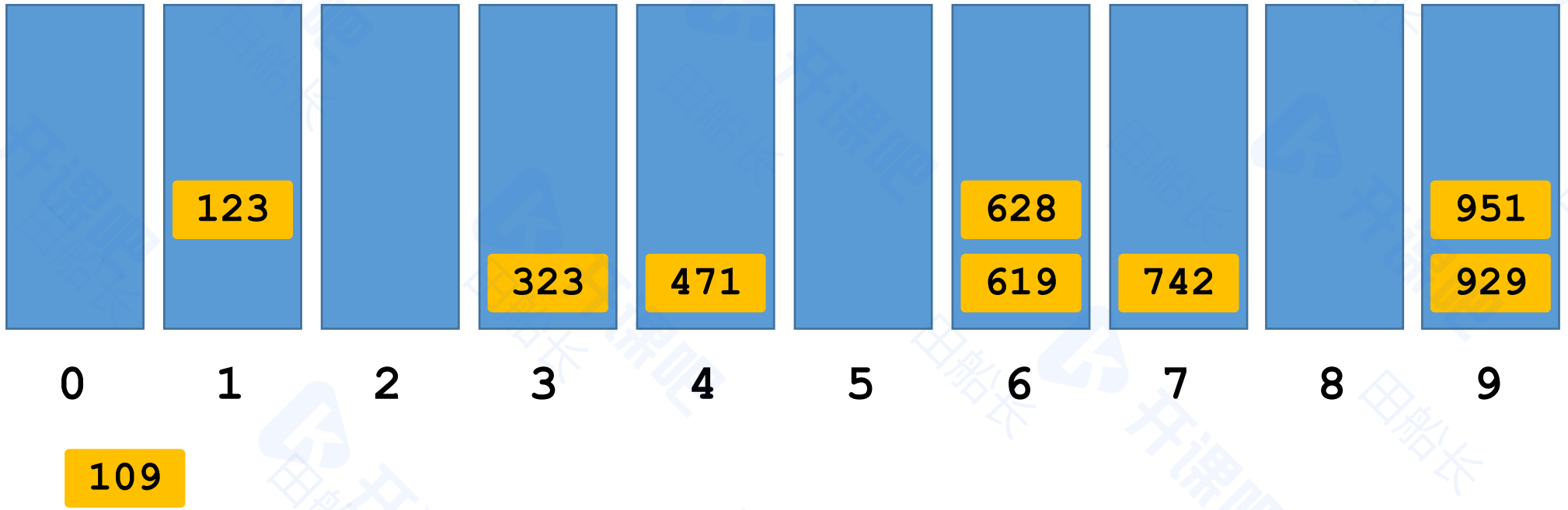
第三趟排序

按照先进先出的原则
将所有元素拿出



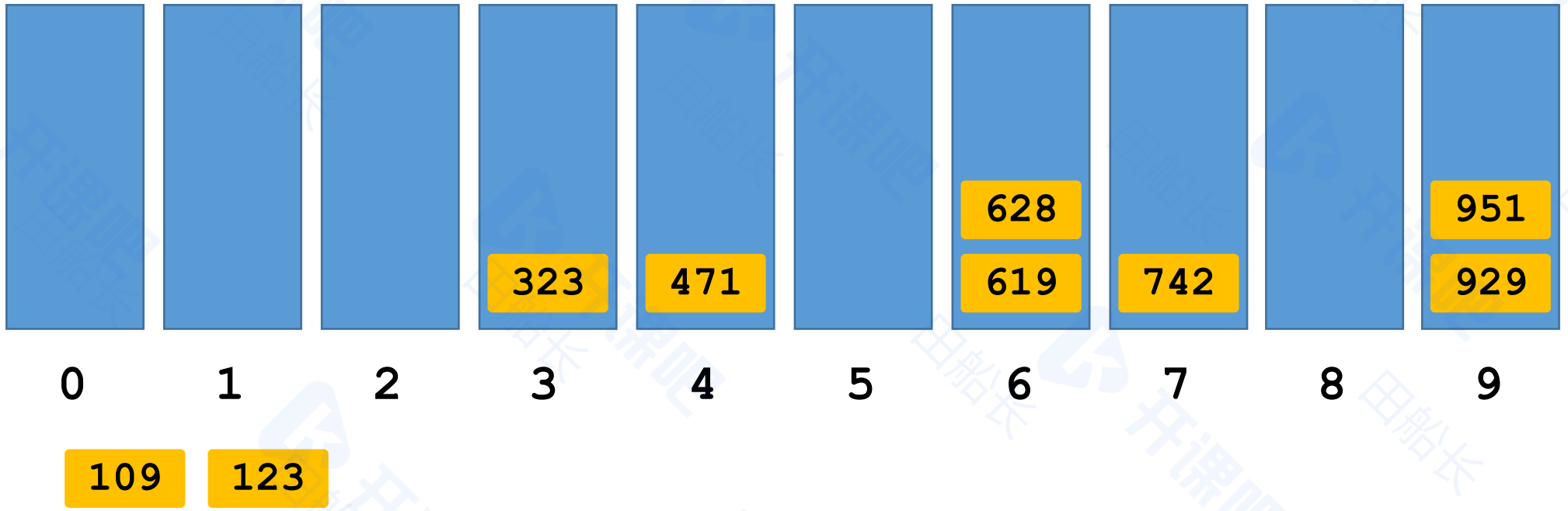
第三趟排序

按照先进先出的原则
将所有元素拿出



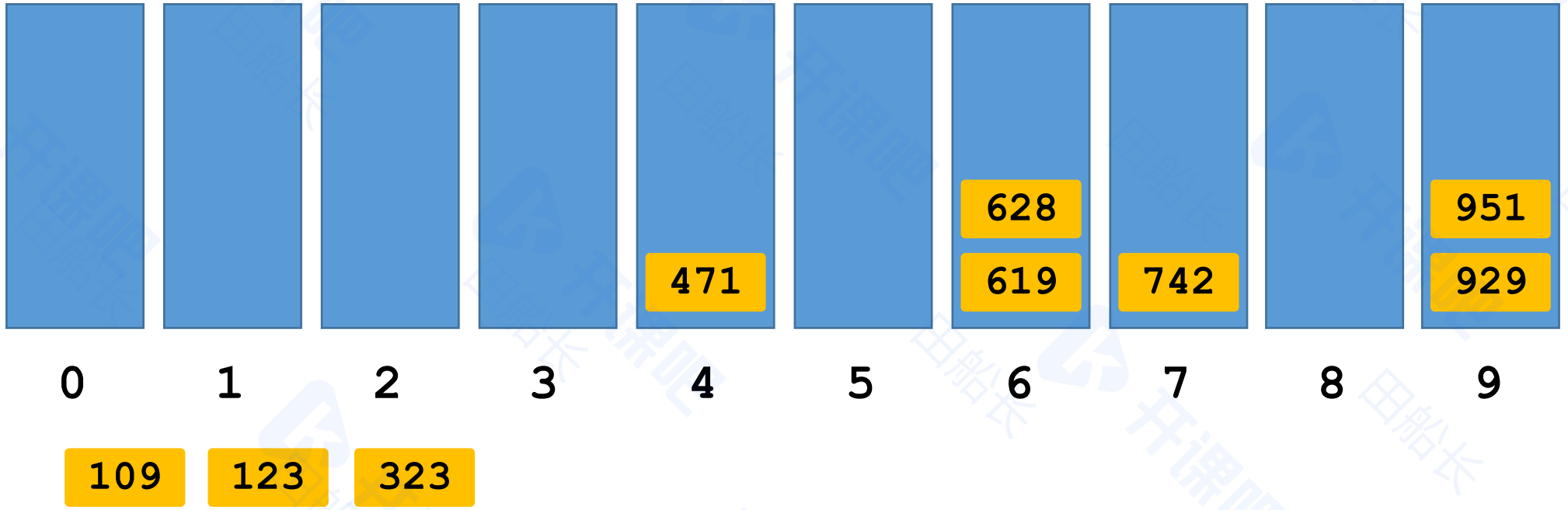
第三趟排序

按照先进先出的原则
将所有元素拿出



第三趟排序

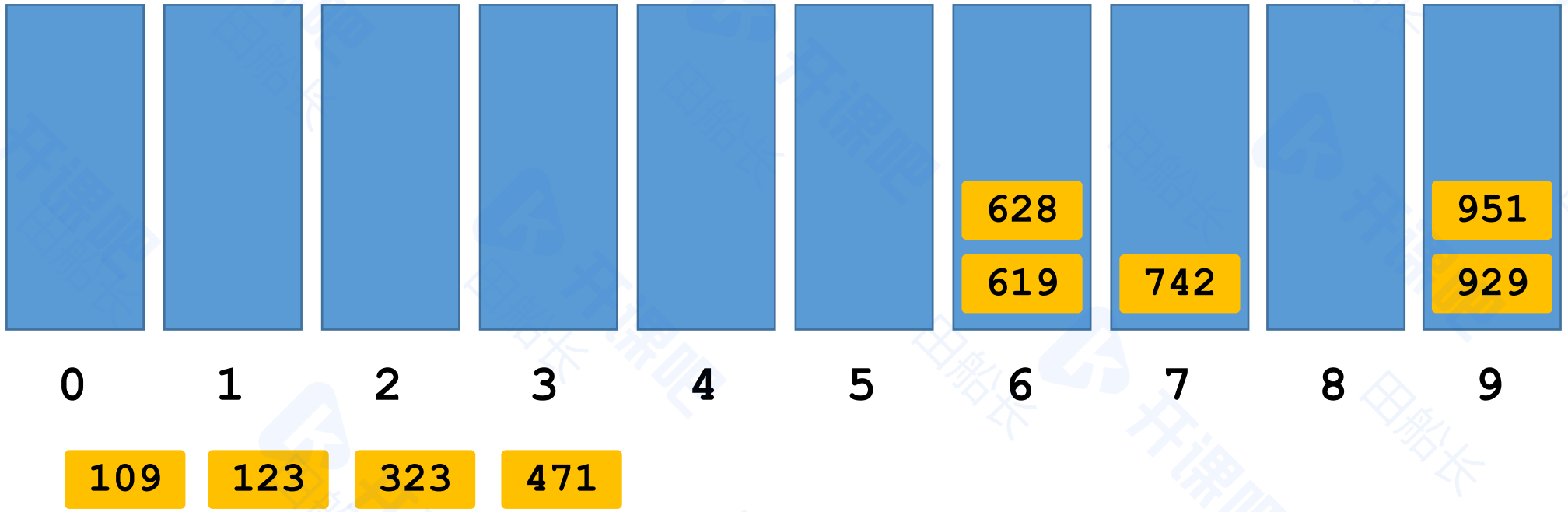
按照先进先出的原则
将所有元素拿出



开课吧

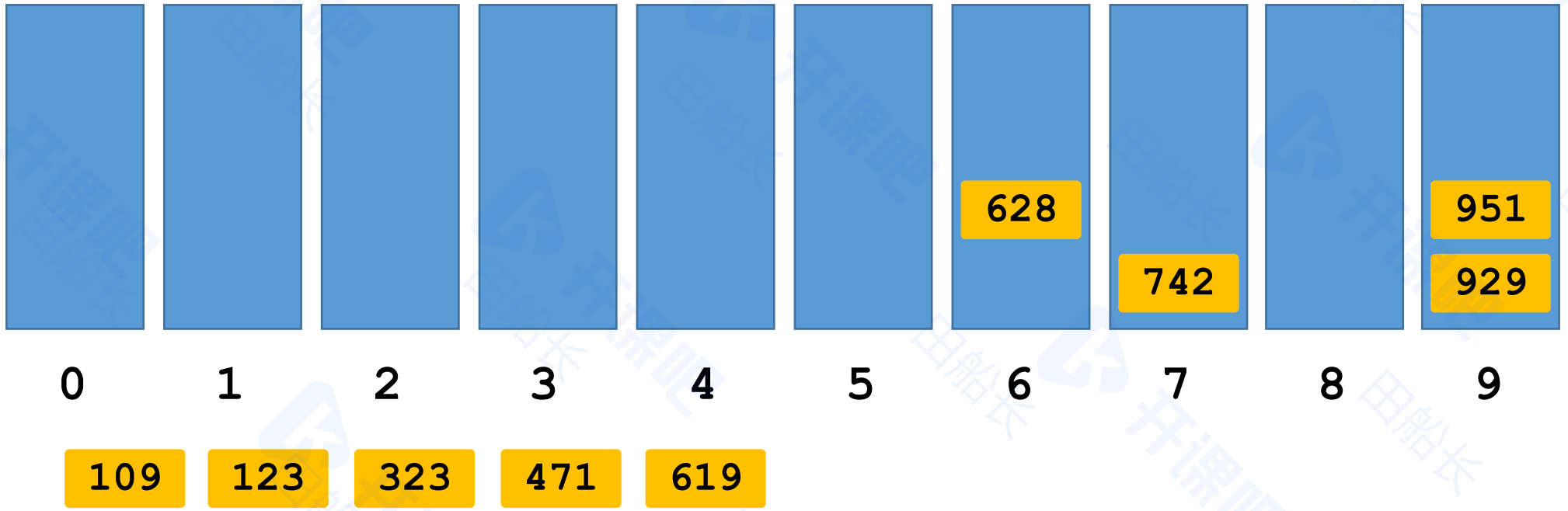
第三趟排序

按照先进先出的原则
将所有元素拿出



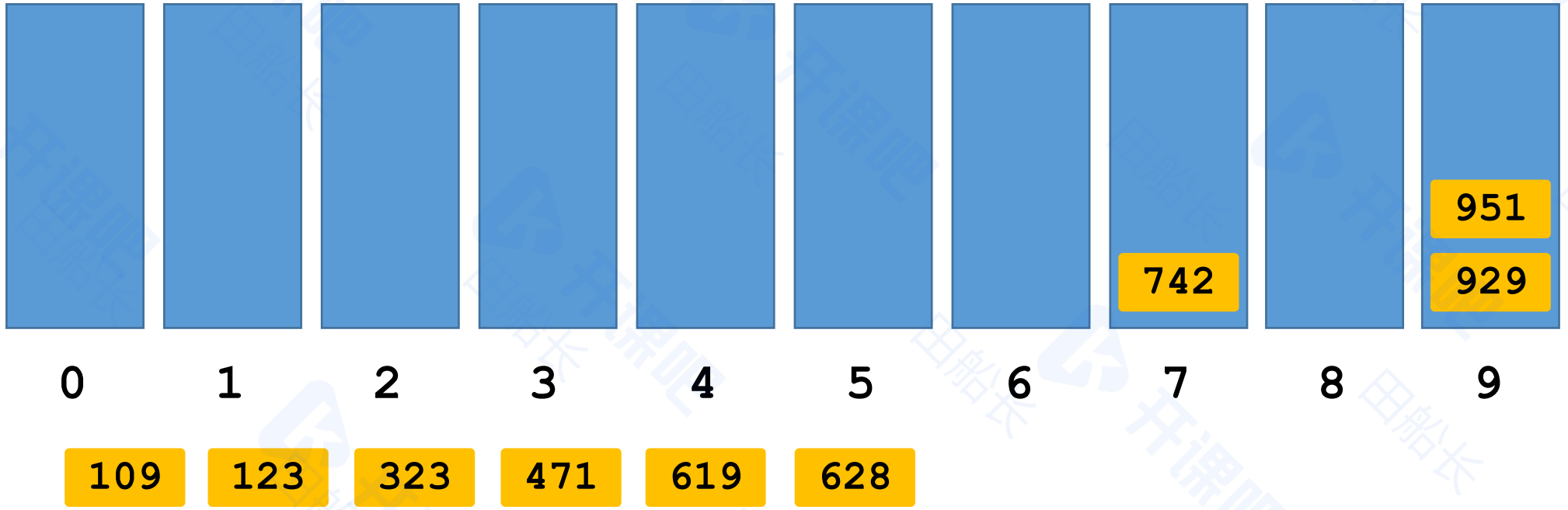
第三趟排序

按照先进先出的原则
将所有元素拿出



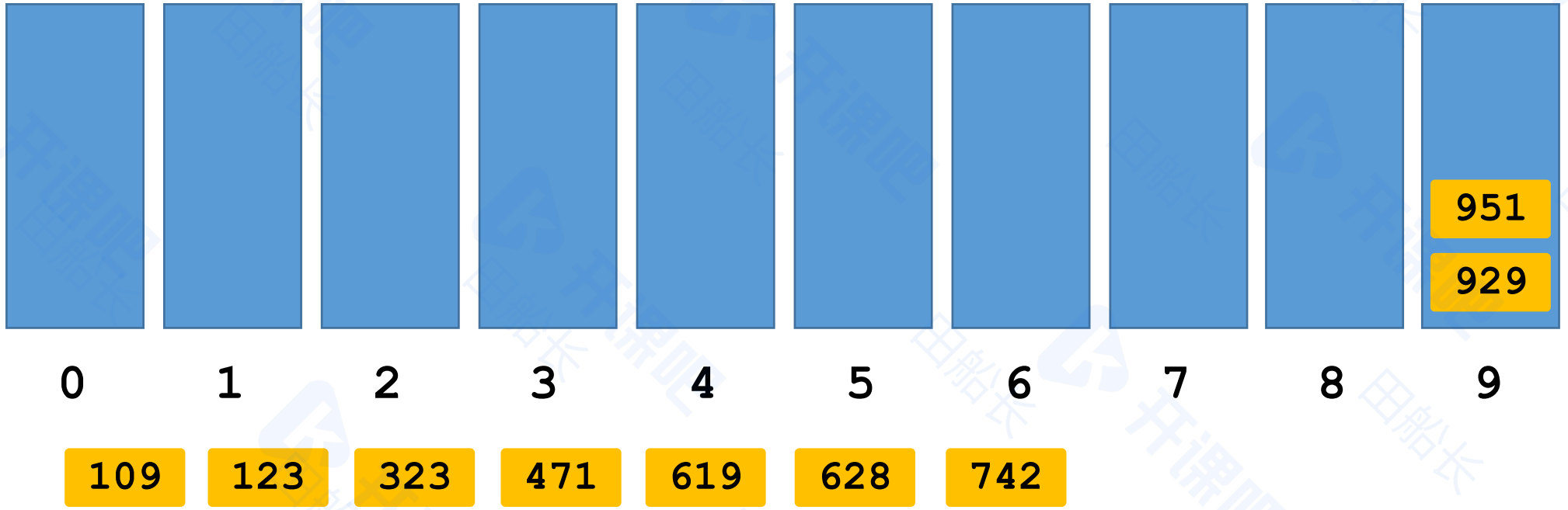
第三趟排序

按照先进先出的原则
将所有元素拿出



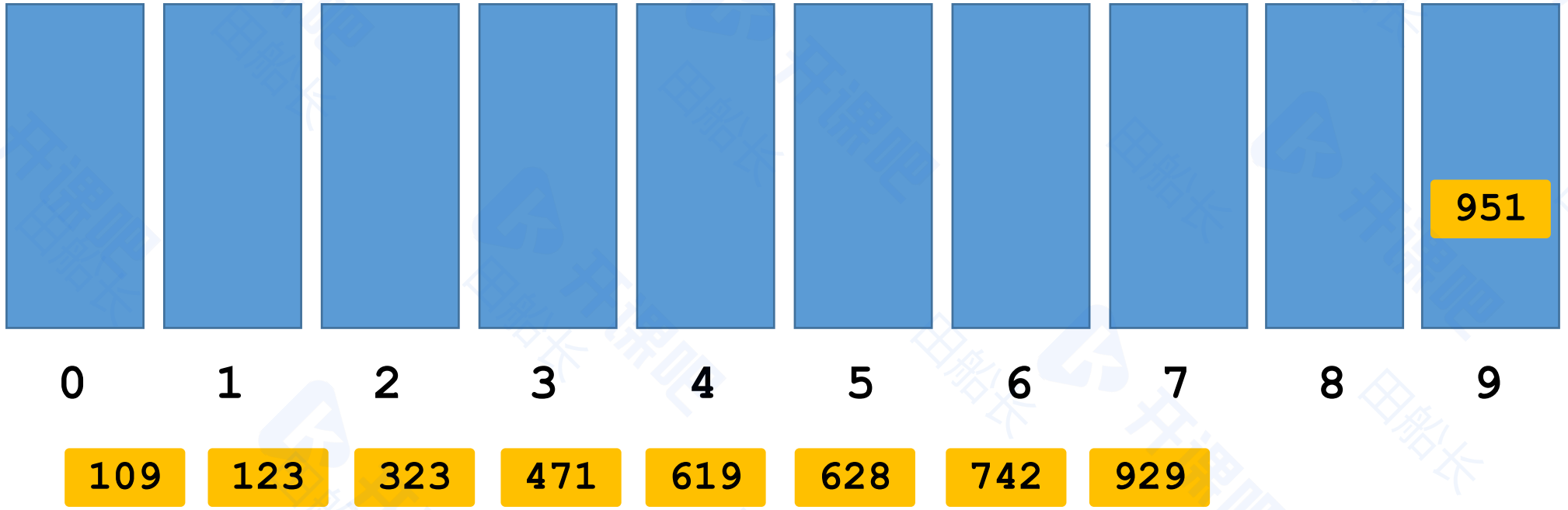
第三趟排序

按照先进先出的原则
将所有元素拿出



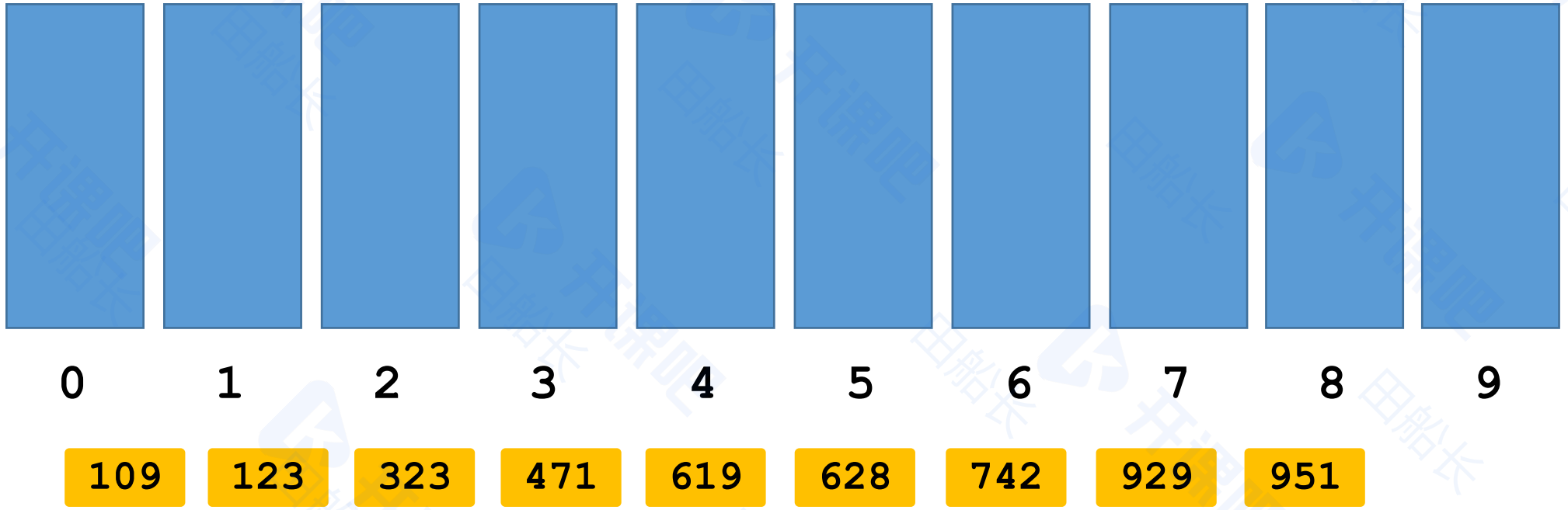
第三趟排序

按照先进先出的原则
将所有元素拿出



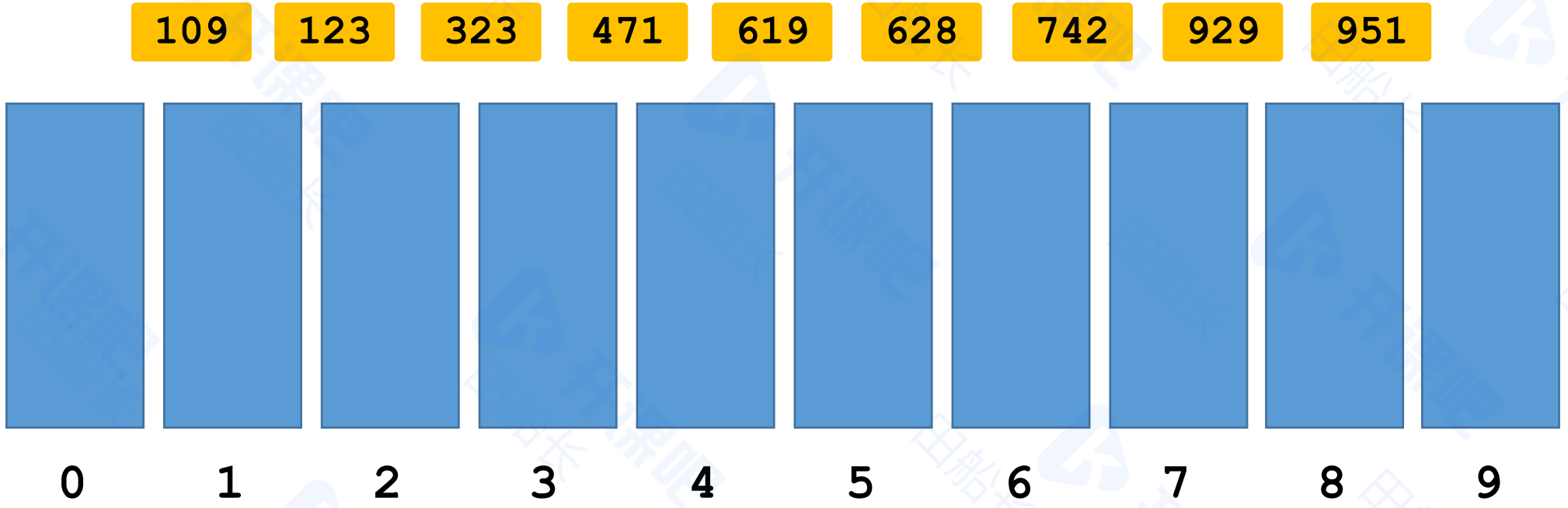
第三趟排序

按照先进先出的原则
将所有元素拿出



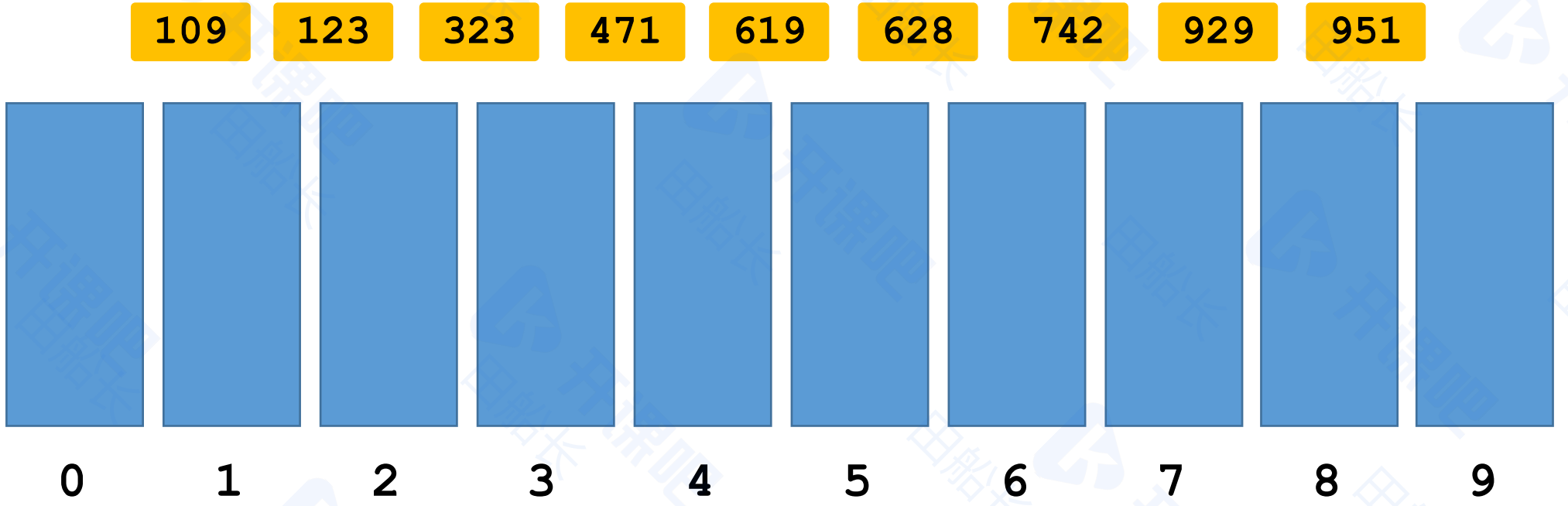
第三趟排序结束

所有元素已按照百位排好序
百位相同的元素按照十位排序
十位相同的元素按照个位排序



开课吧

基数排序结束
所有元素均有序



内部排序的对比

排序方法		插入排序	冒泡排序	选择排序	希尔排序	快速排序	堆排序	归并排序	基数排序
时间复杂度	最好	O(N)	O(N)	O(N^2)	O(N^k)* k∈[1,2]	O(NlogN)	O(NlogN)	O(NlogN)	O(kN)
	平均	O(N^2)	O(N^2)	O(N^2)		O(NlogN)	O(NlogN)	O(NlogN)	
	最坏	O(N^2)	O(N^2)	O(N^2)		O(N^2)	O(NlogN)	O(NlogN)	
空间复杂度		O(1)	O(1)	O(1)	O(1)	O(logN)*	O(1)	O(N)	O(m+N)
是否稳定		是	是	否	否	否	否	是	是

希尔排序的时间复杂度不定　　快速排序的空间占用取决于递归深度