算法与数据结构体系课程

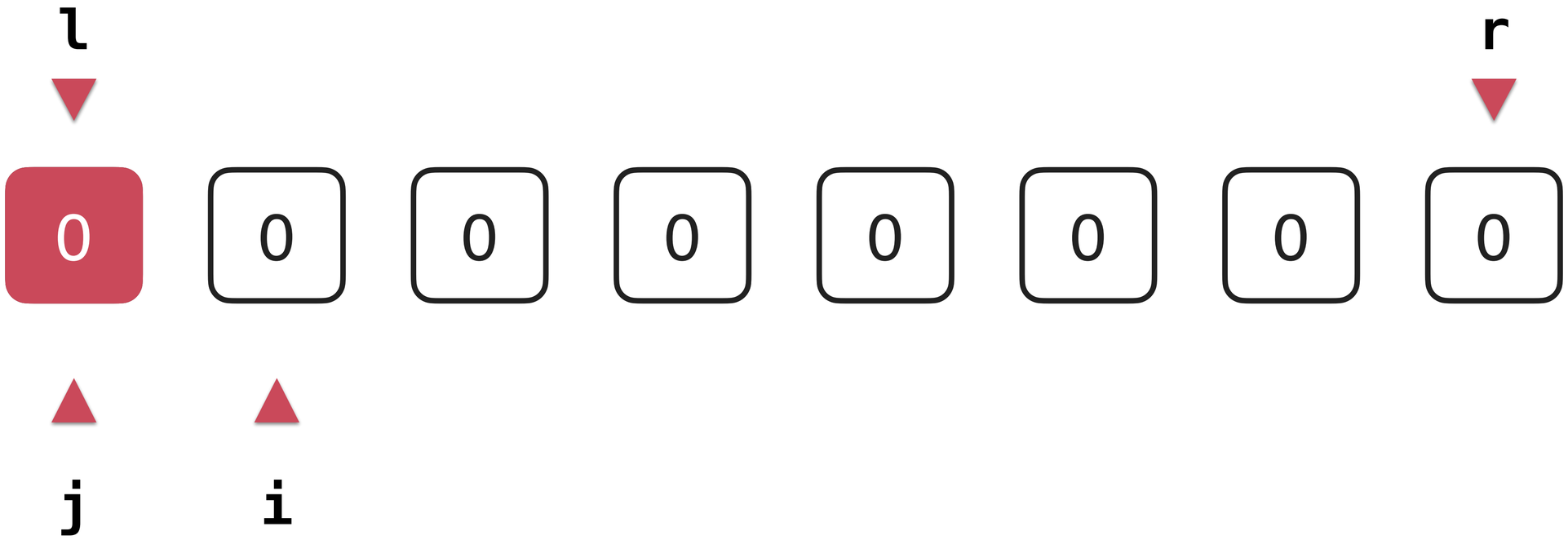
liuyubobobo

更多关于快速排序法

快速排序法还有问题

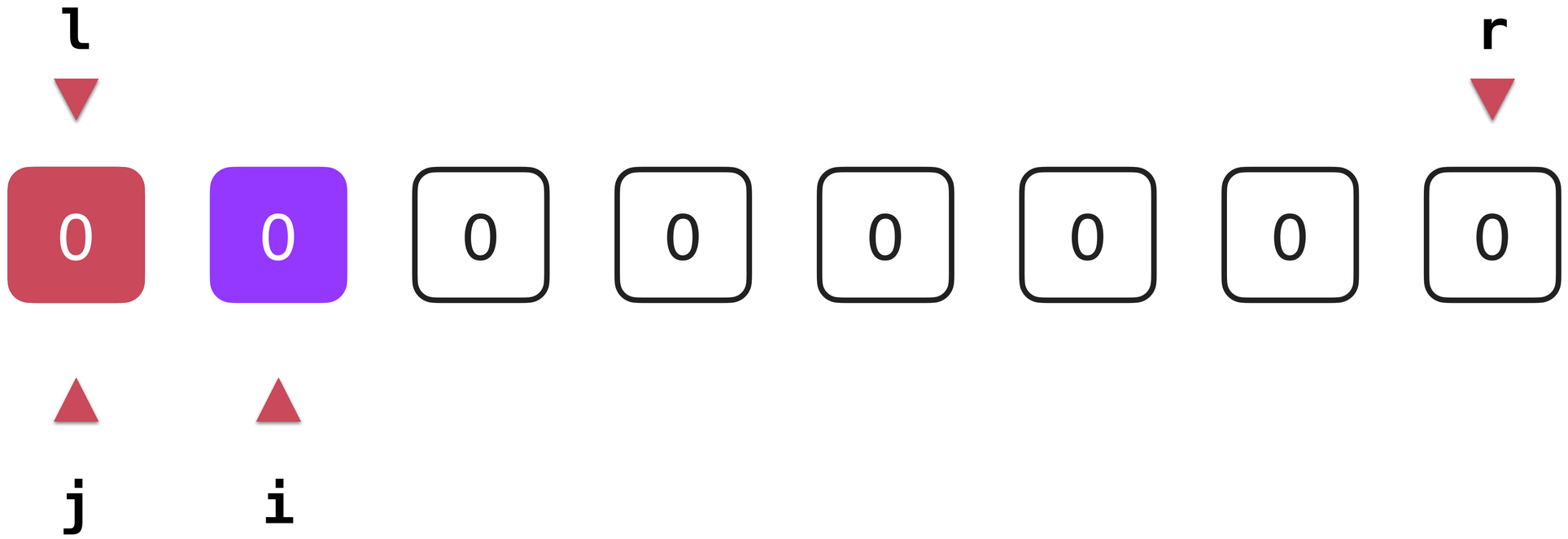
实践：展示快速排序法的问题

快速排序法还有问题



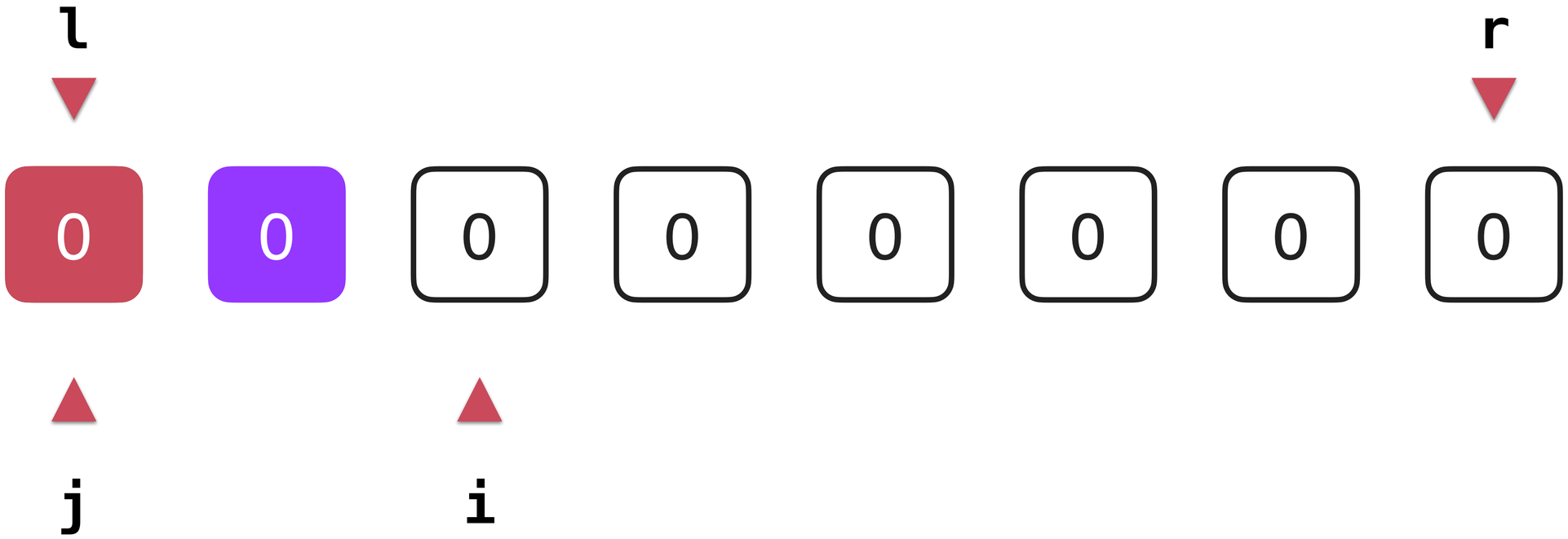
arr[l+1…j] < v　　arr[j+1…i-1] >= v

快速排序法还有问题



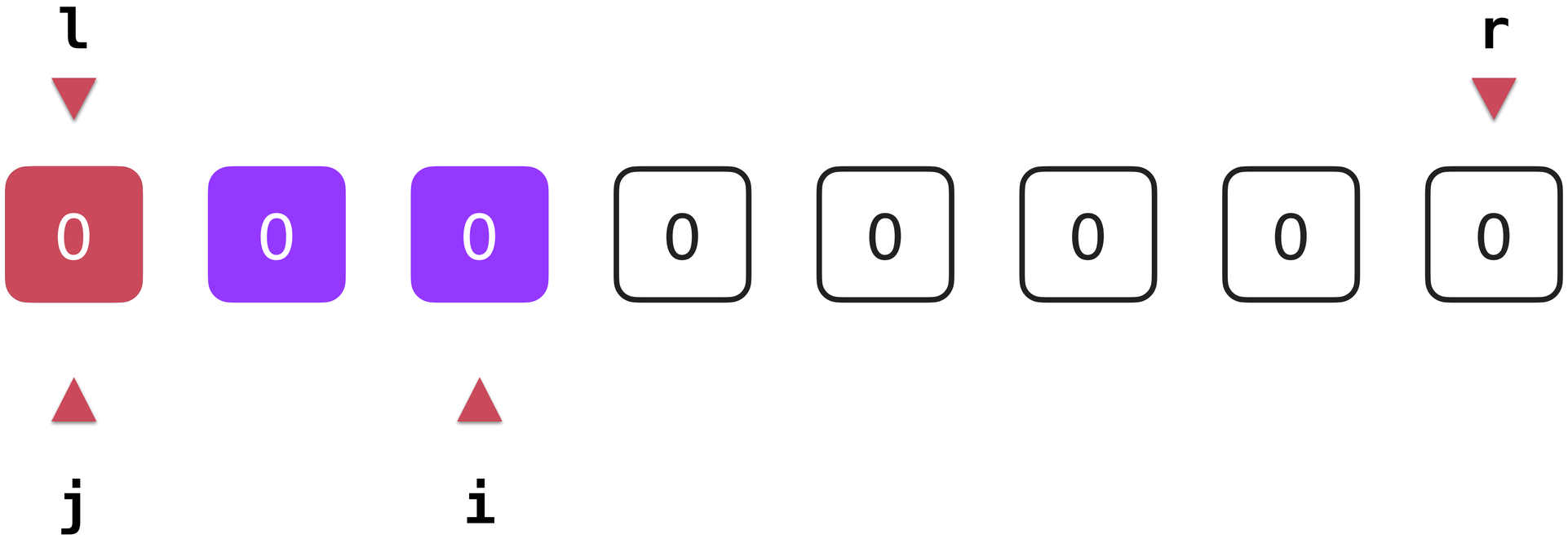
arr[l+1…j] < v　　arr[j+1…i-1] >= v

快速排序法还有问题



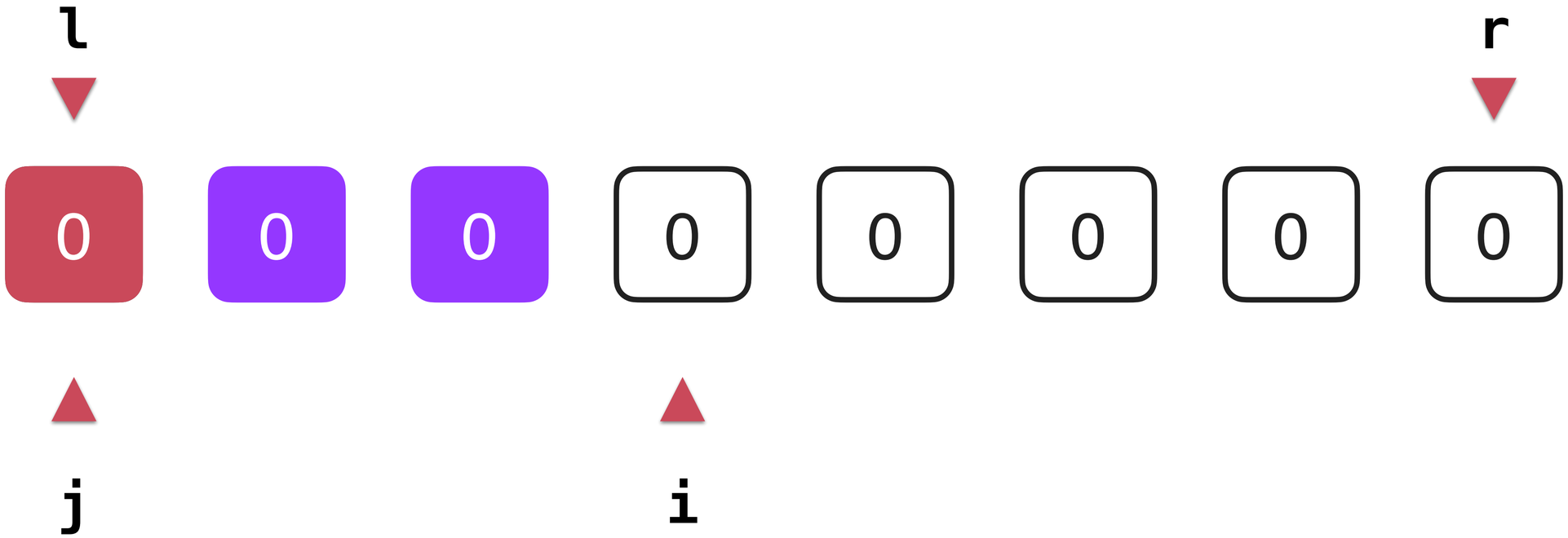
arr[l+1…j] < v　　arr[j+1…i-1] >= v

快速排序法还有问题



arr[l+1…j] < v　　arr[j+1…i-1] >= v

快速排序法还有问题

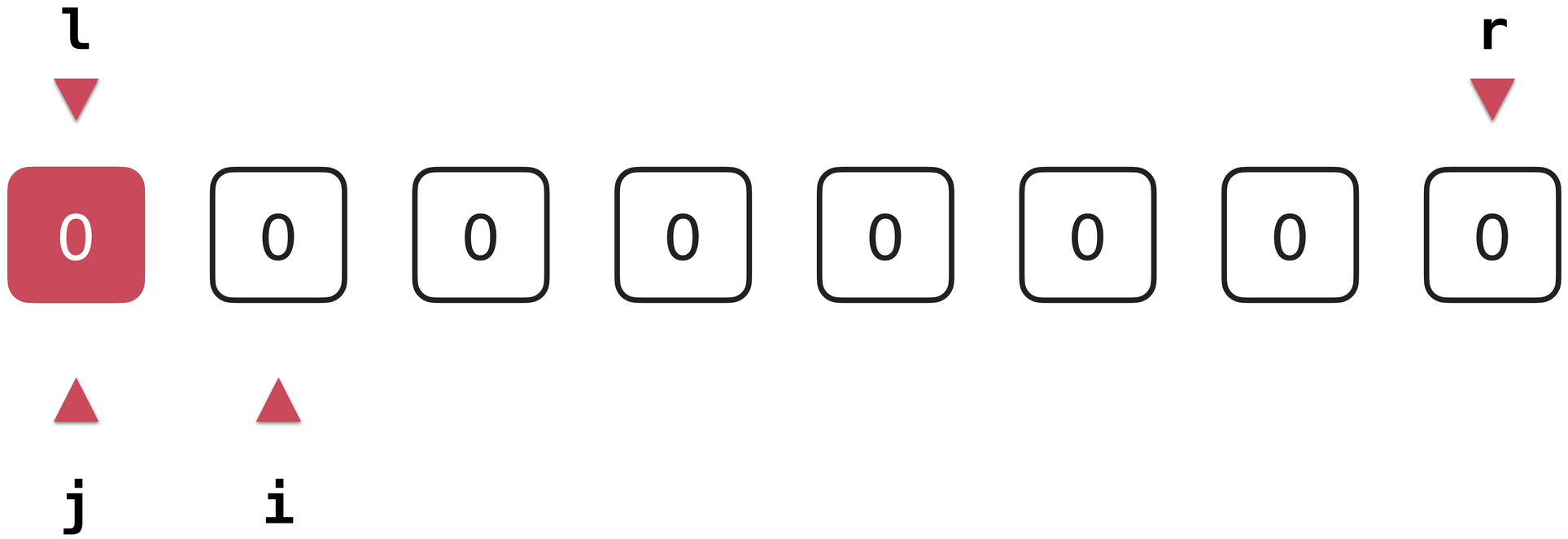


arr[l+1…j] < v　　arr[j+1…i-1] >= v

快速排序法还有问题

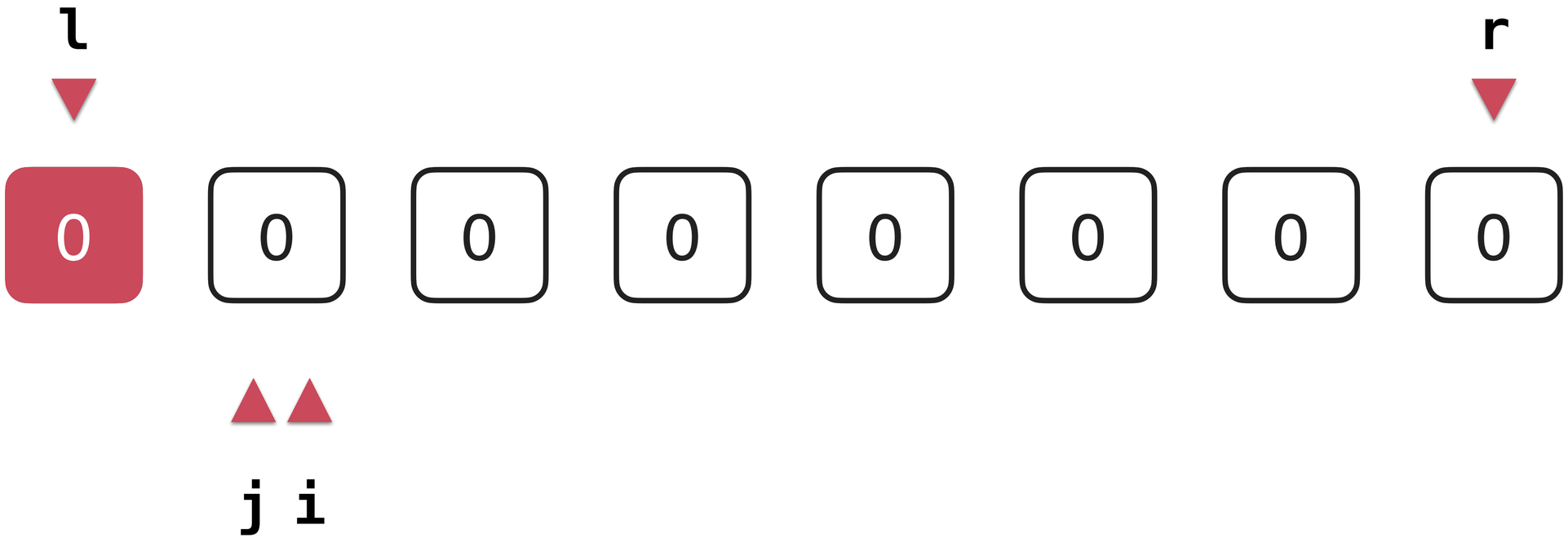
arr[l+1…j] < v　　arr[j+1…i-1] >= v

快速排序法还有问题



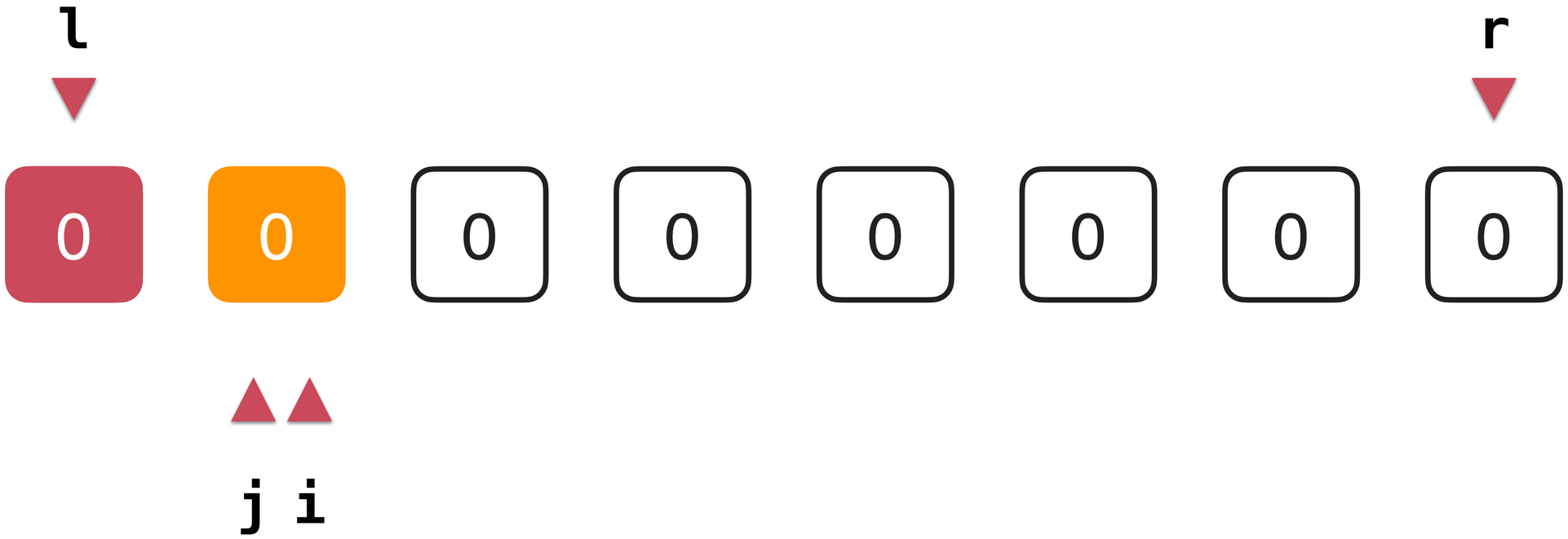
arr[l+1…j] <= v　　arr[j+1…i-1] > v

快速排序法还有问题



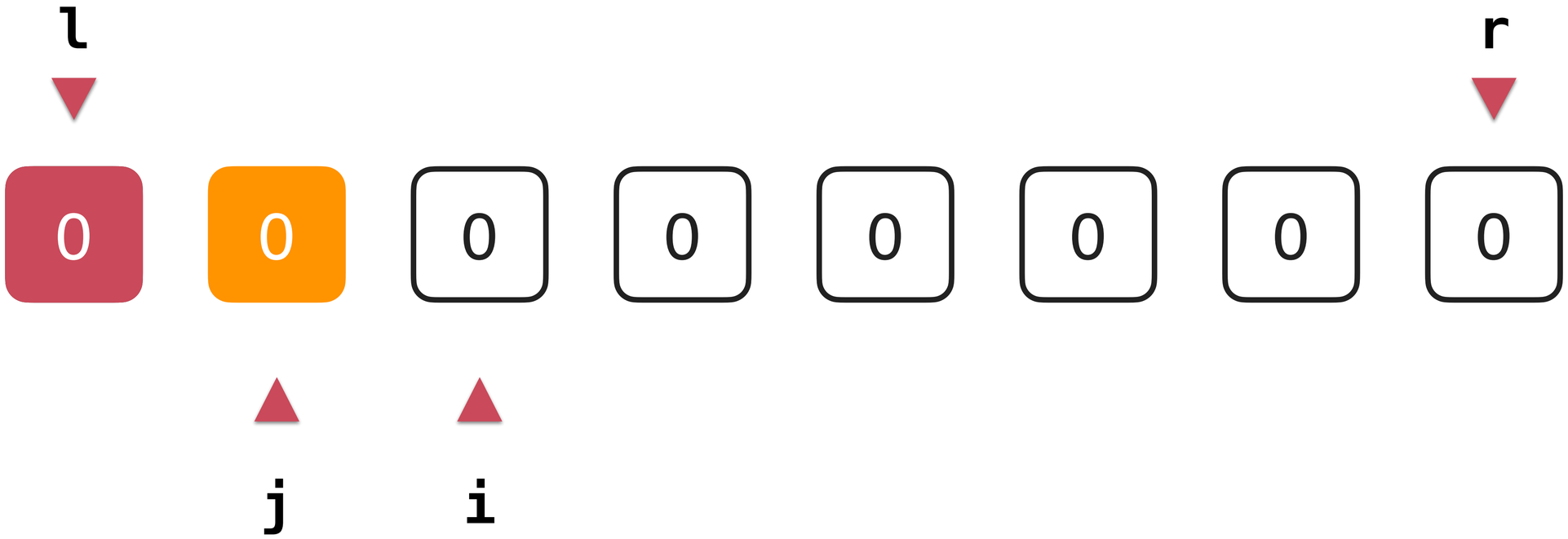
arr[l+1…j] <= v　　arr[j+1…i-1] > v

快速排序法还有问题



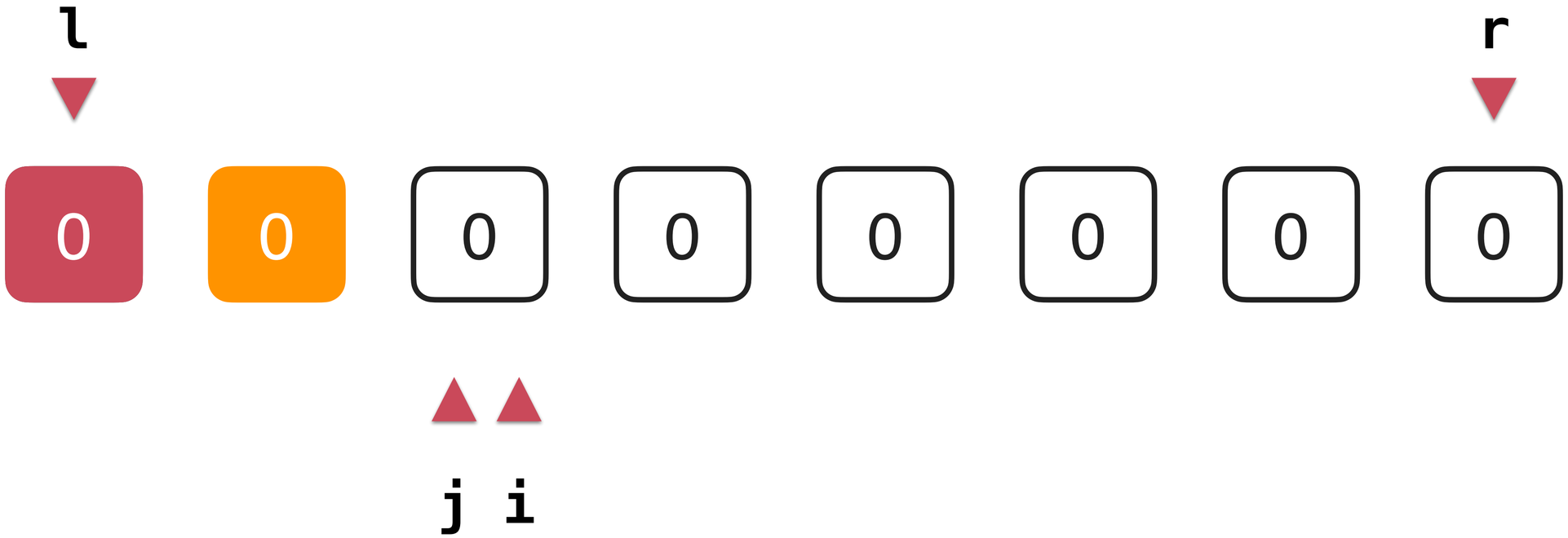
arr[l+1…j] <= v　　arr[j+1…i-1] > v

快速排序法还有问题



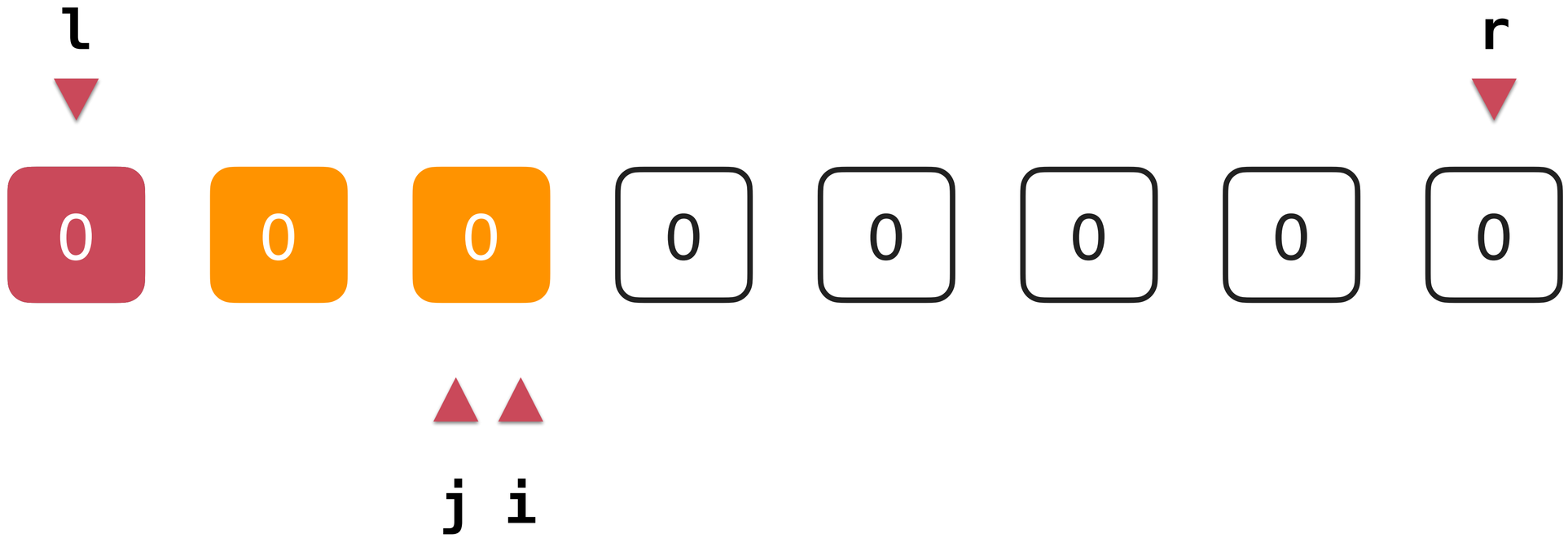
arr[l+1…j] <= v　　arr[j+1…i-1] > v

快速排序法还有问题



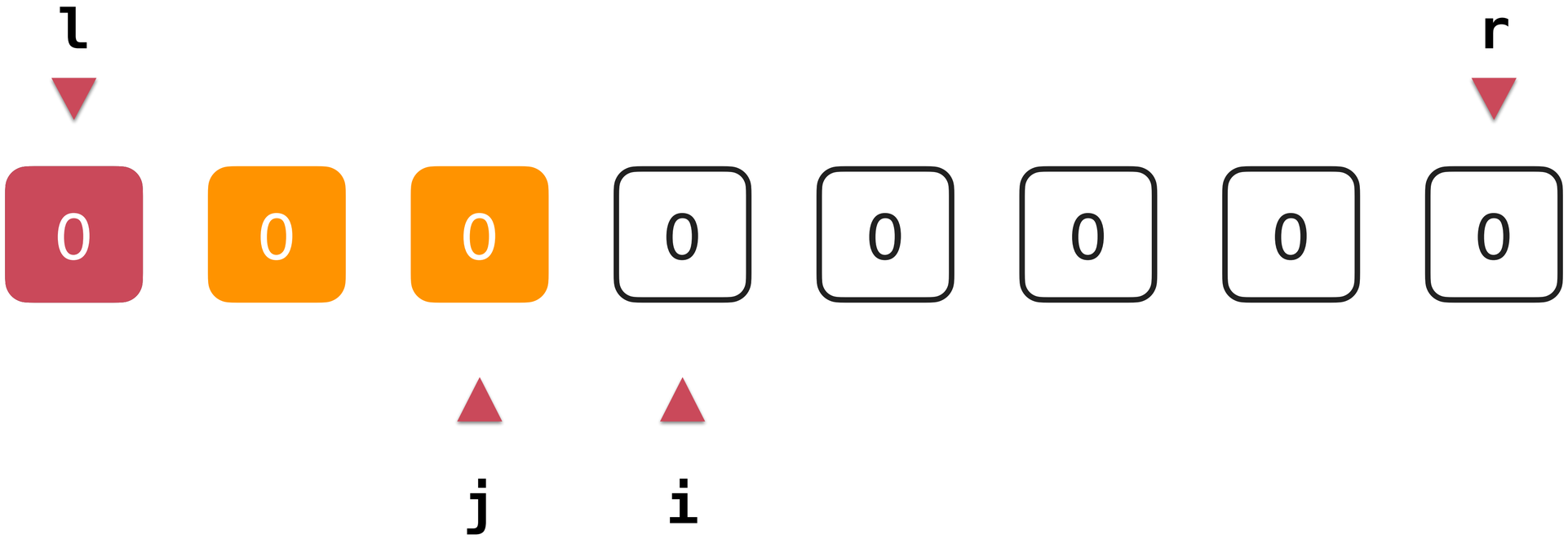
arr[l+1…j] <= v　　arr[j+1…i-1] > v

快速排序法还有问题



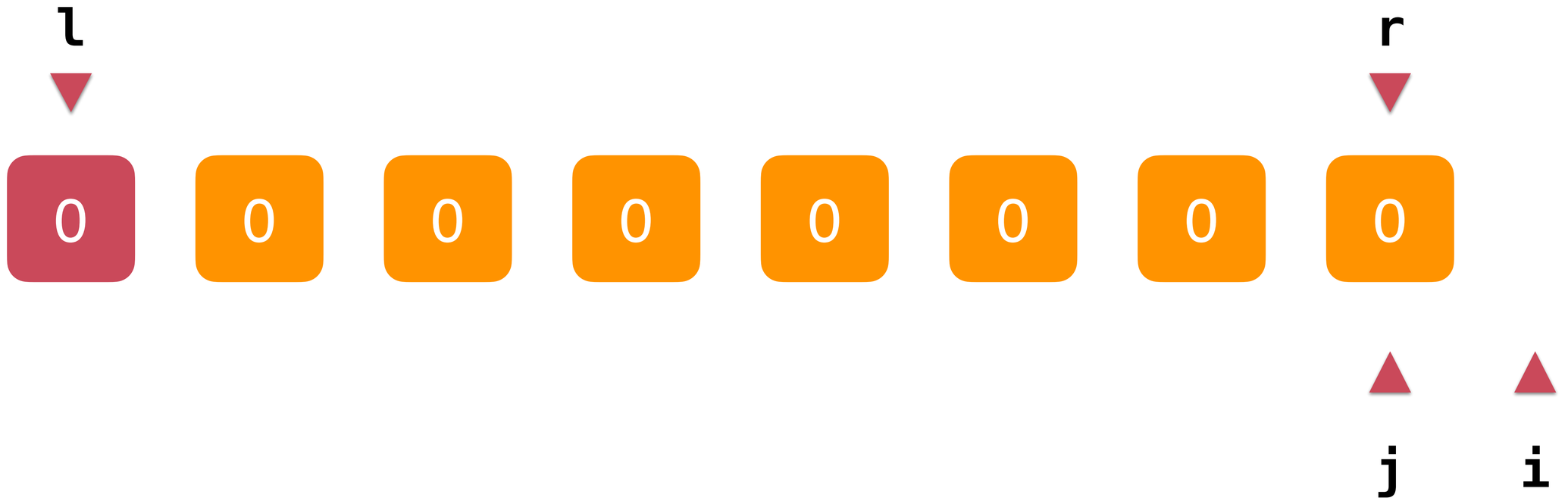
arr[l+1…j] <= v　　arr[j+1…i-1] > v

快速排序法还有问题



arr[l+1…j] <= v　　arr[j+1…i-1] > v

快速排序法还有问题



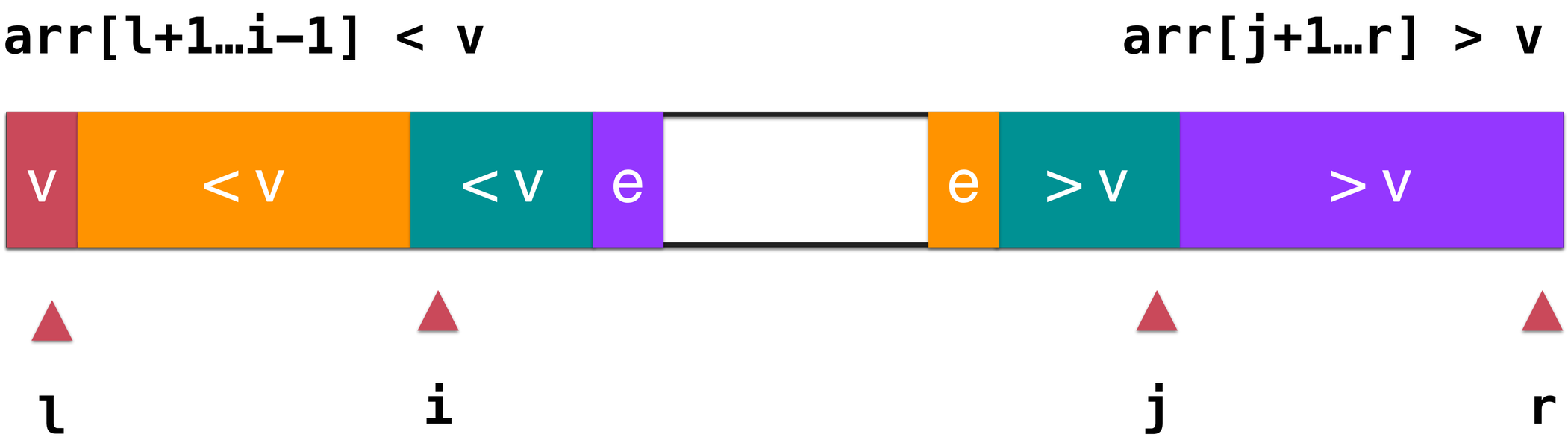
arr[l+1…j] <= v　　arr[j+1…i-1] > v

解决方案

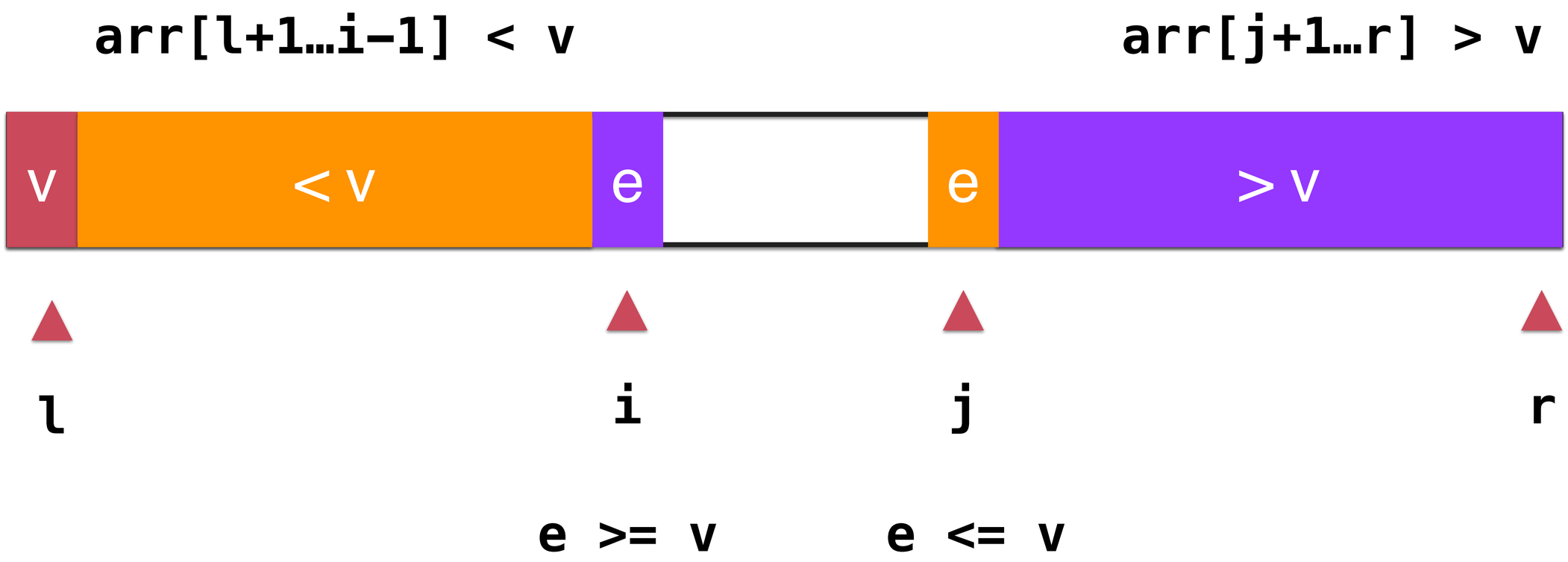
双路快速排序法

双路快速排序法

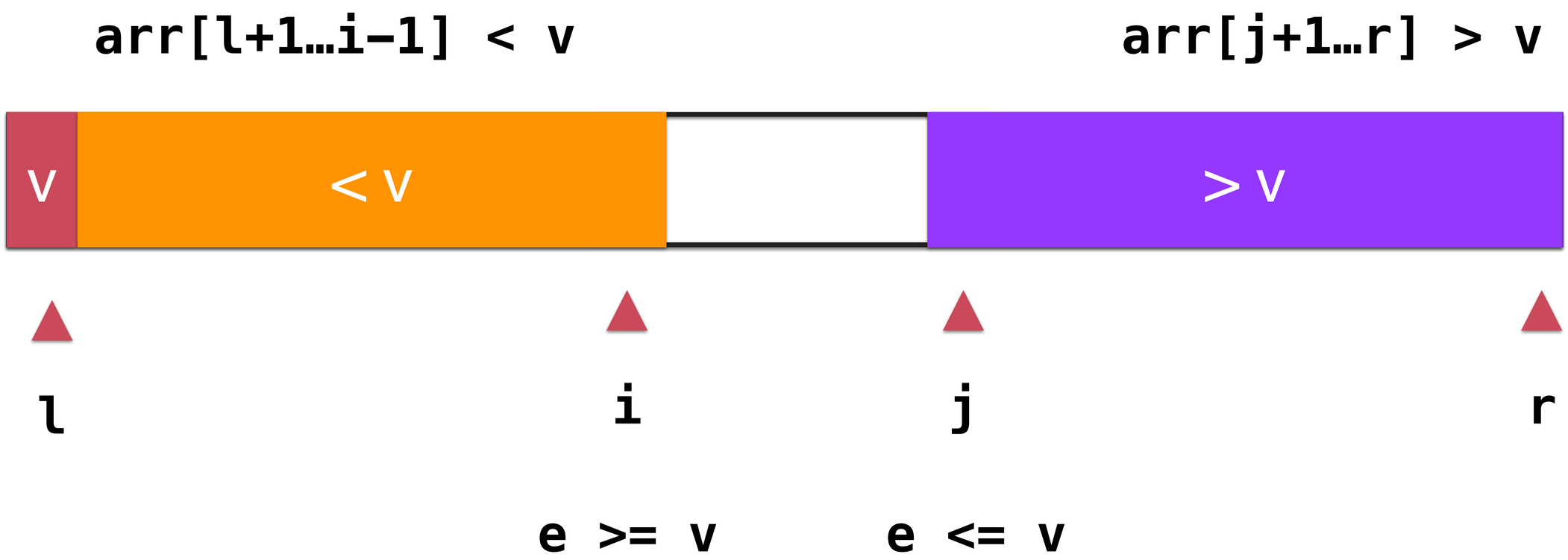
Partition2



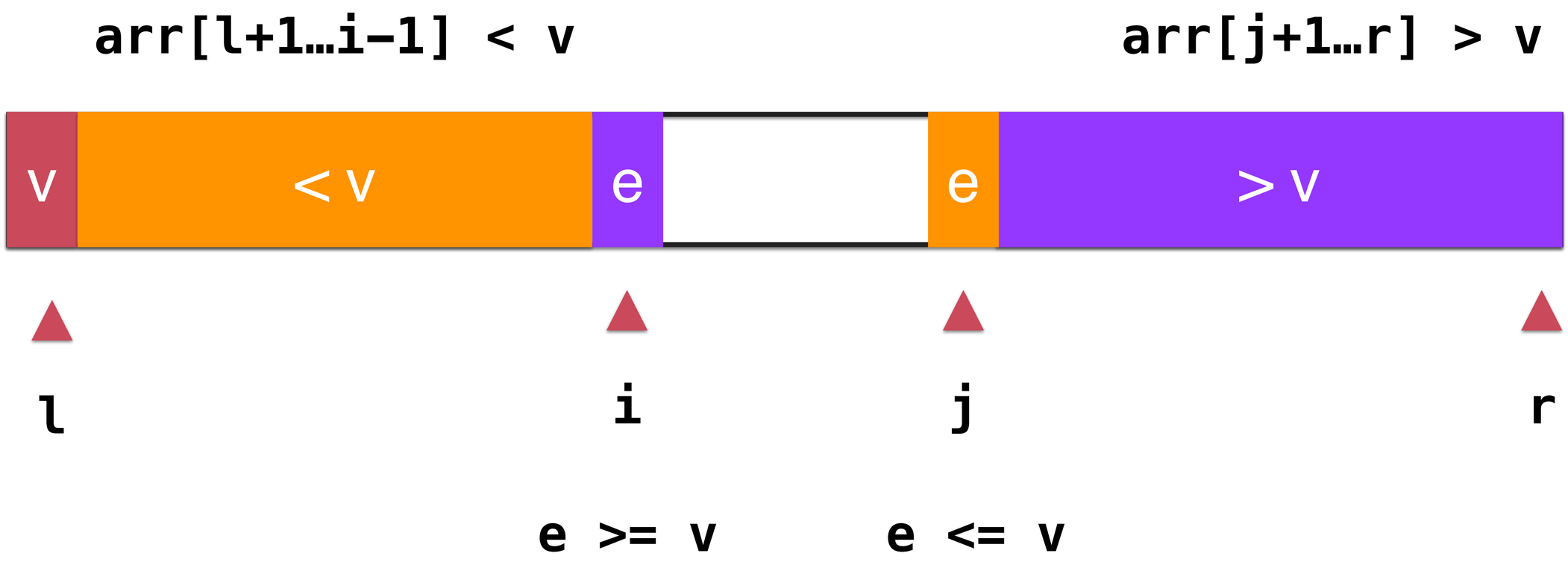
Partition2



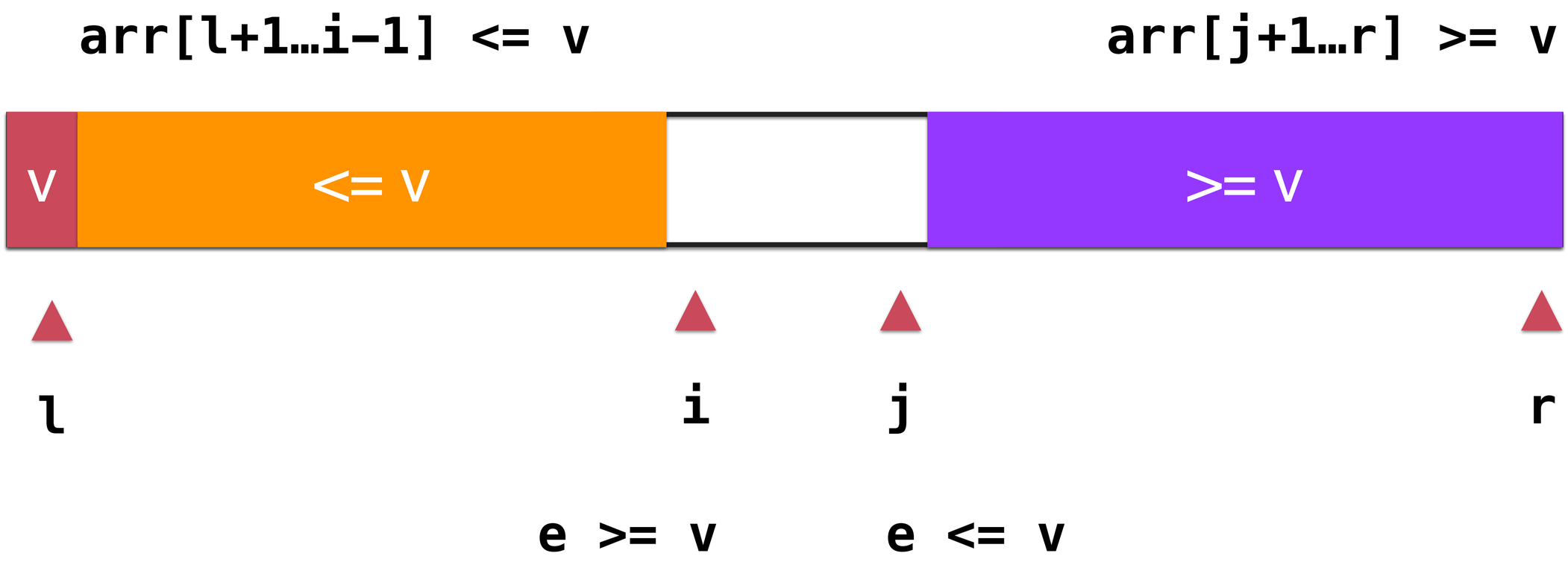
Partition2



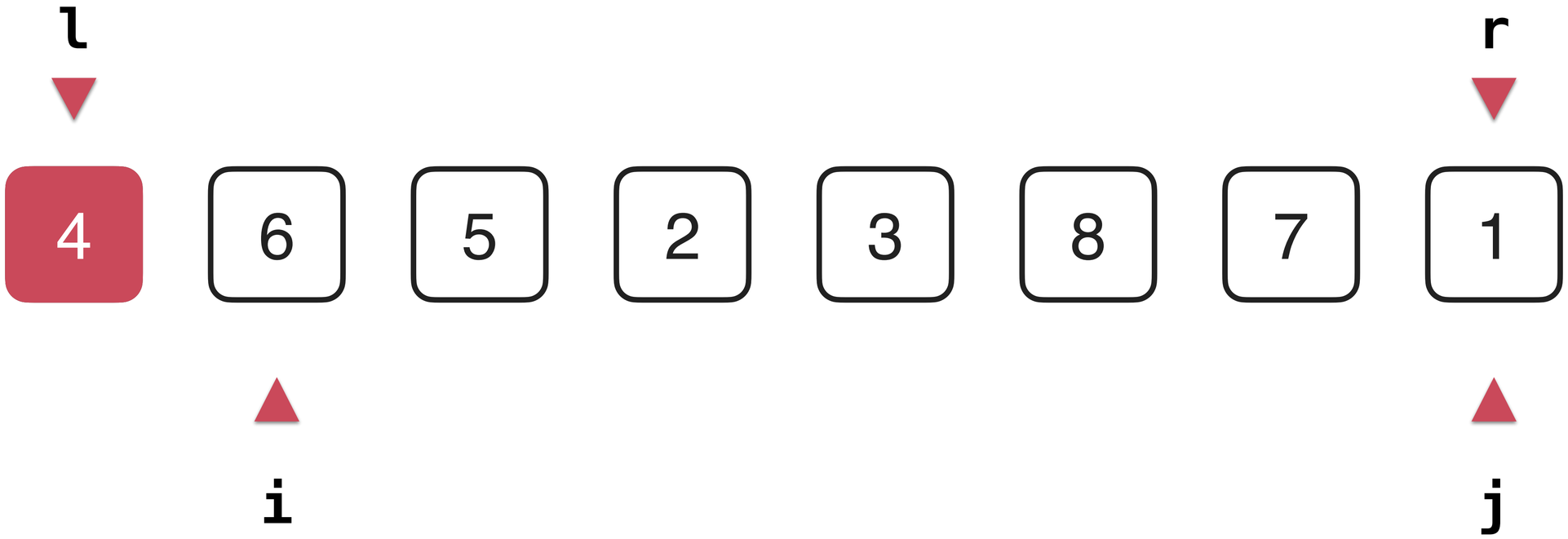
Partition2



Partition2



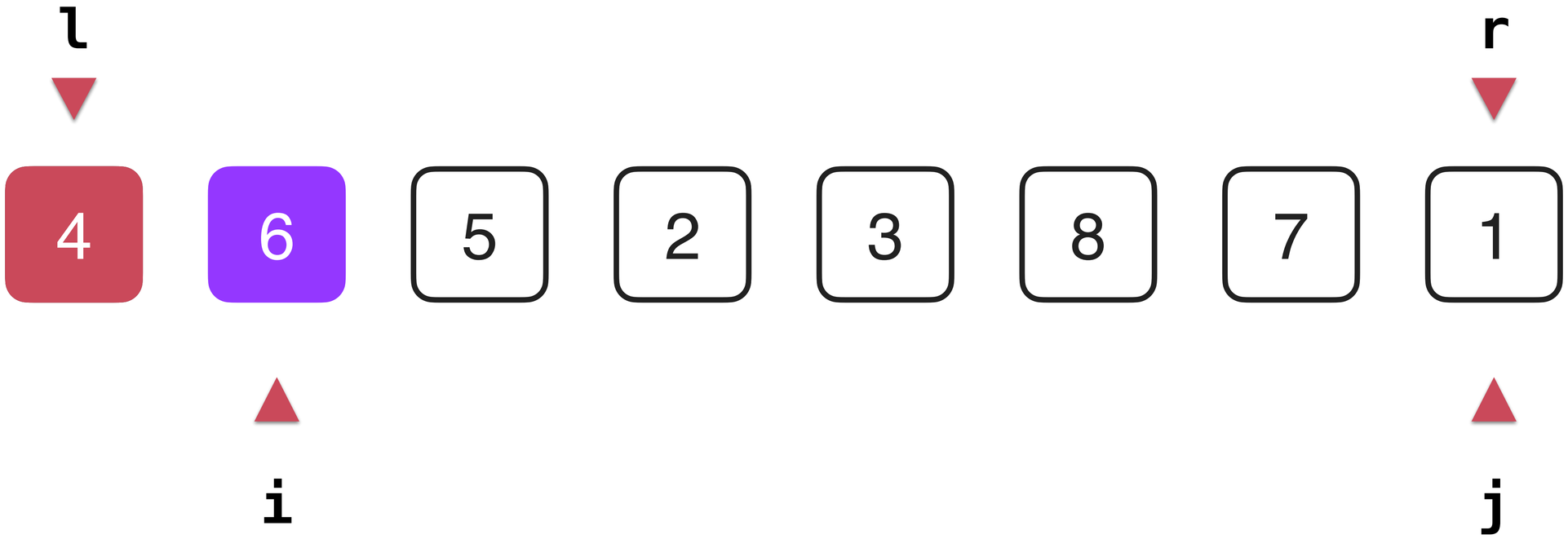
Partition2



`arr[l+1…i-1] <= v`

`arr[j+1…r] >= v`

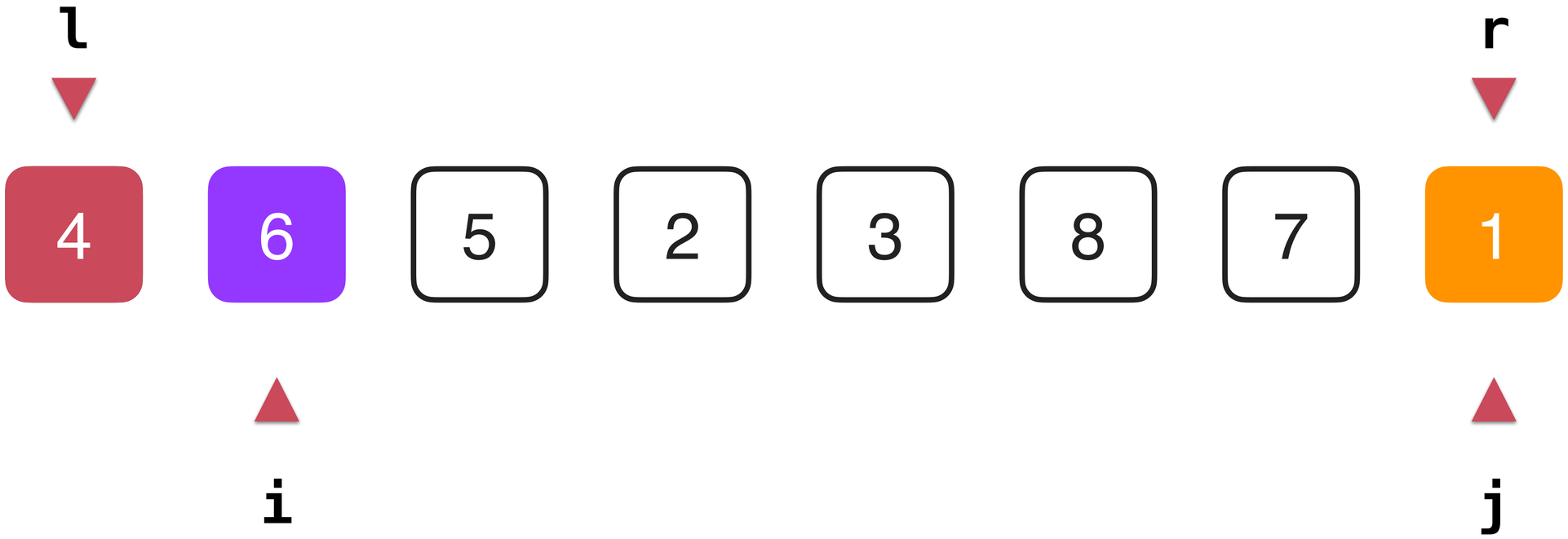
Partition2



`arr[l+1…i-1] <= v`

`arr[j+1…r] >= v`

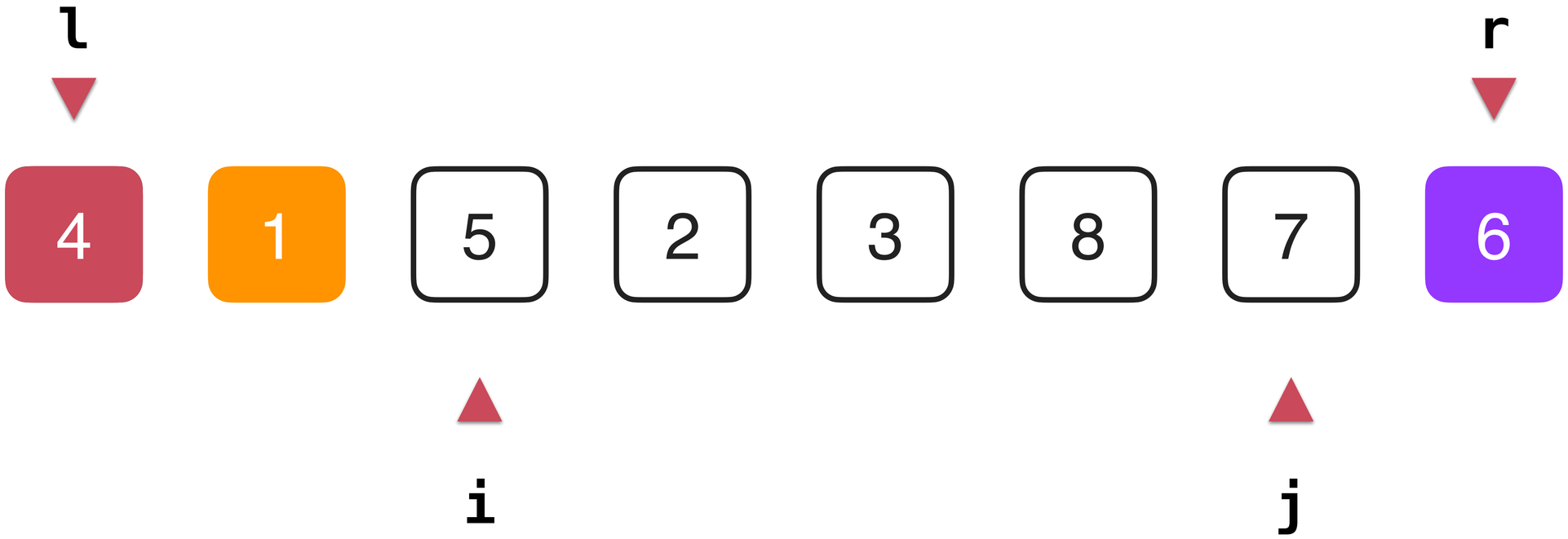
Partition2



`arr[l+1…i-1] <= v`

`arr[j+1…r] >= v`

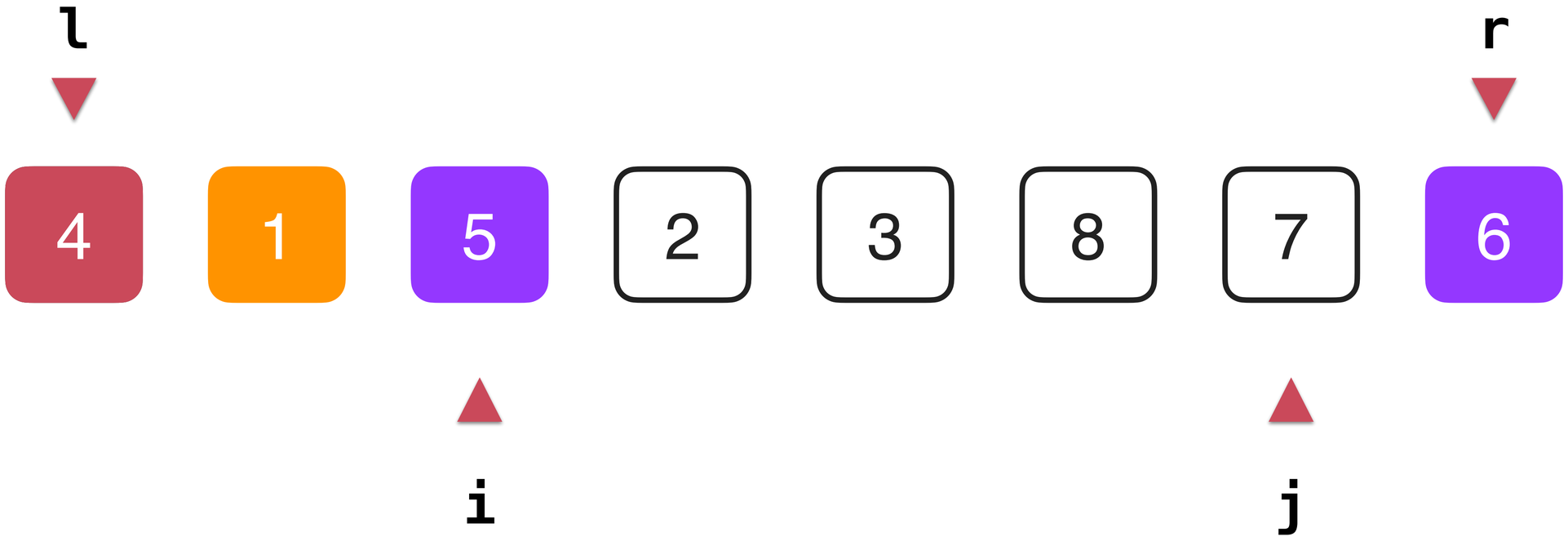
Partition2



`arr[l+1…i-1] <= v`

`arr[j+1…r] >= v`

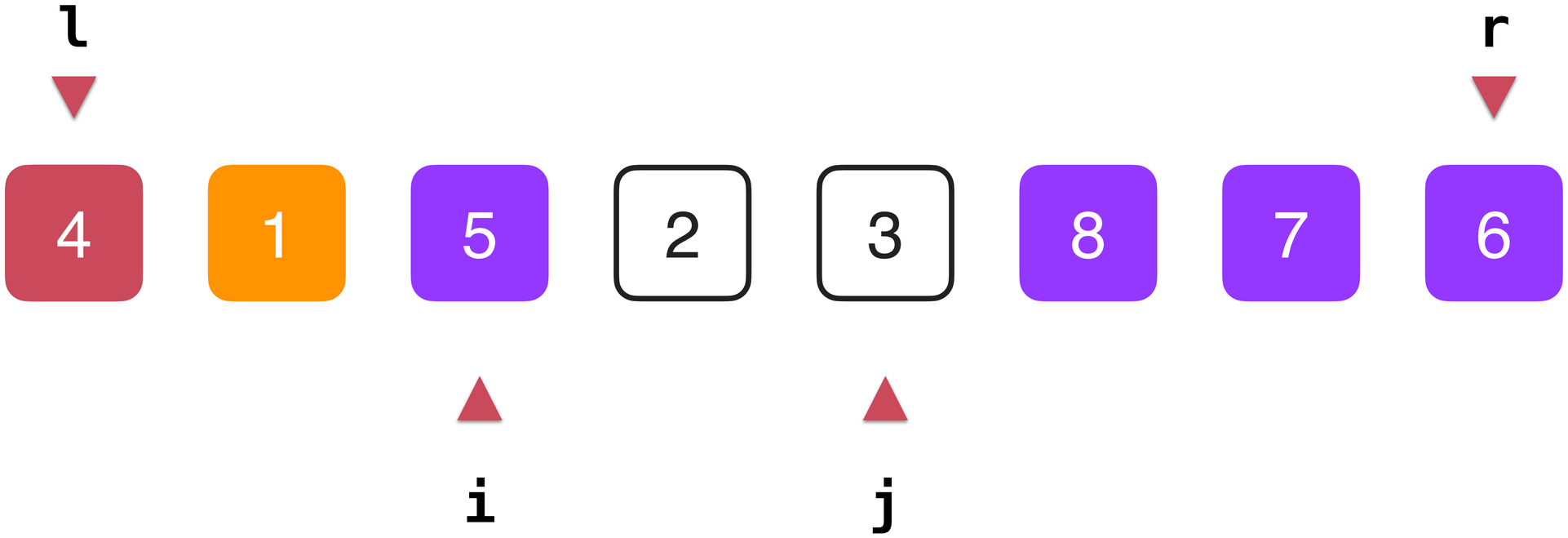
Partition2



`arr[l+1…i-1] <= v`

`arr[j+1…r] >= v`

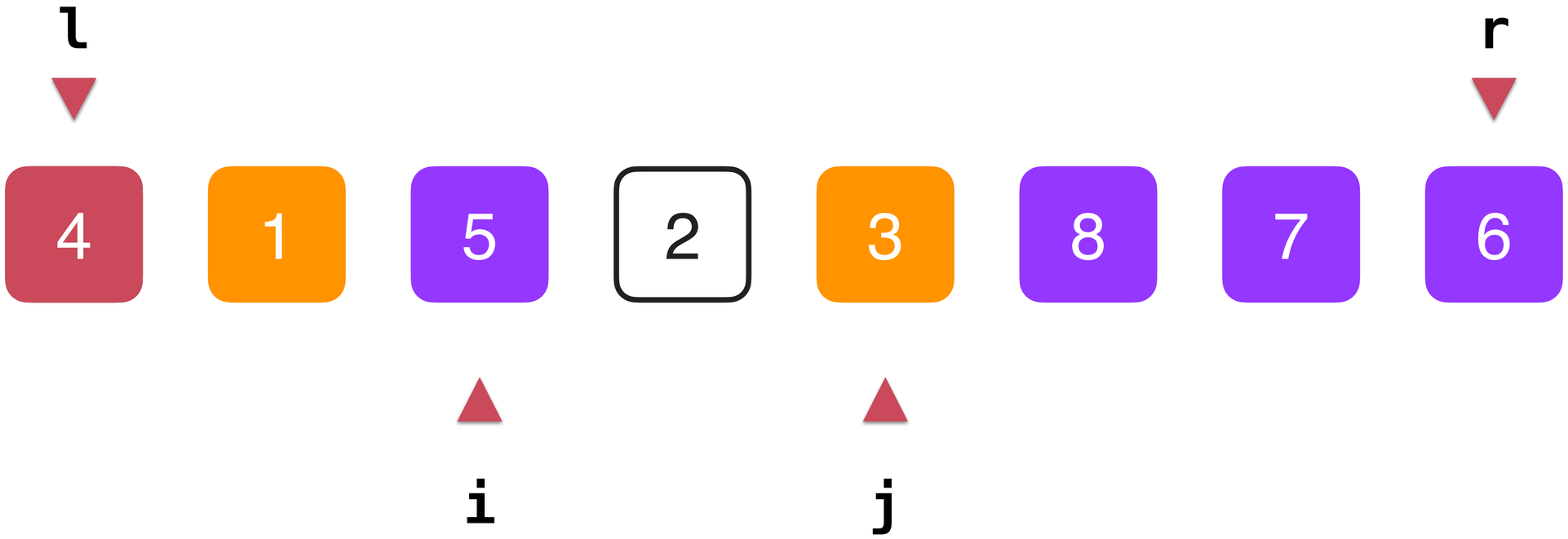
Partition2



`arr[l+1…i-1] <= v`

`arr[j+1…r] >= v`

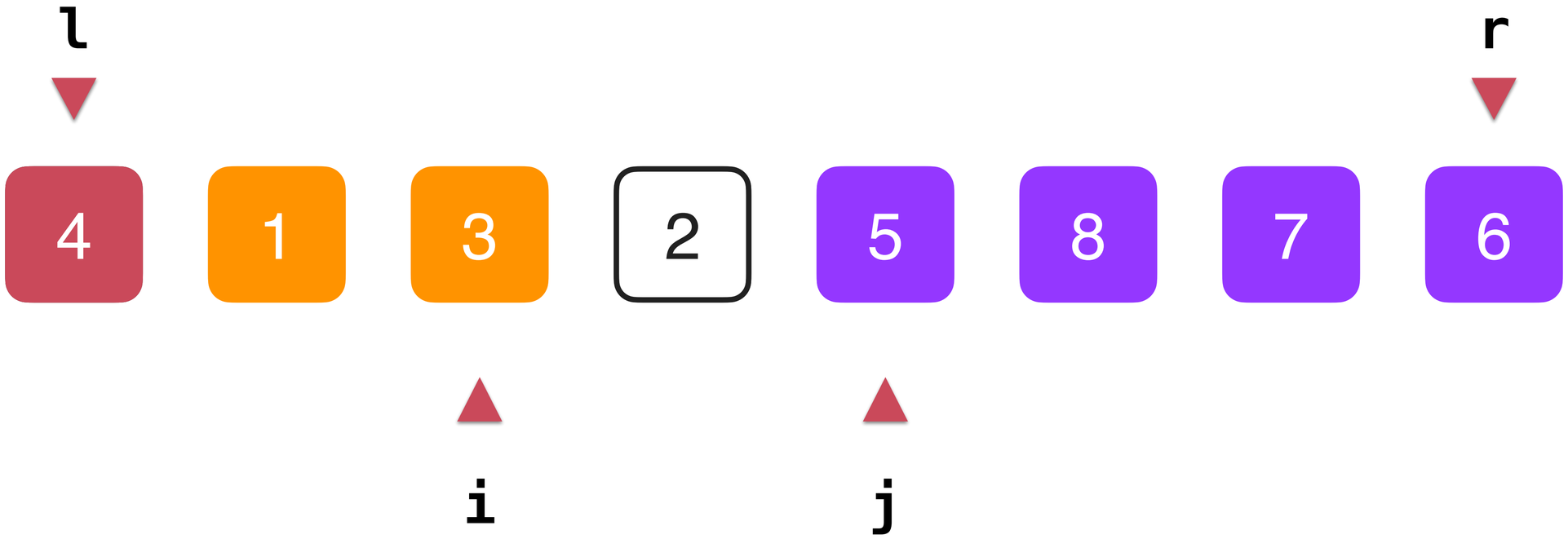
Partition2



`arr[l+1…i-1] <= v`

`arr[j+1…r] >= v`

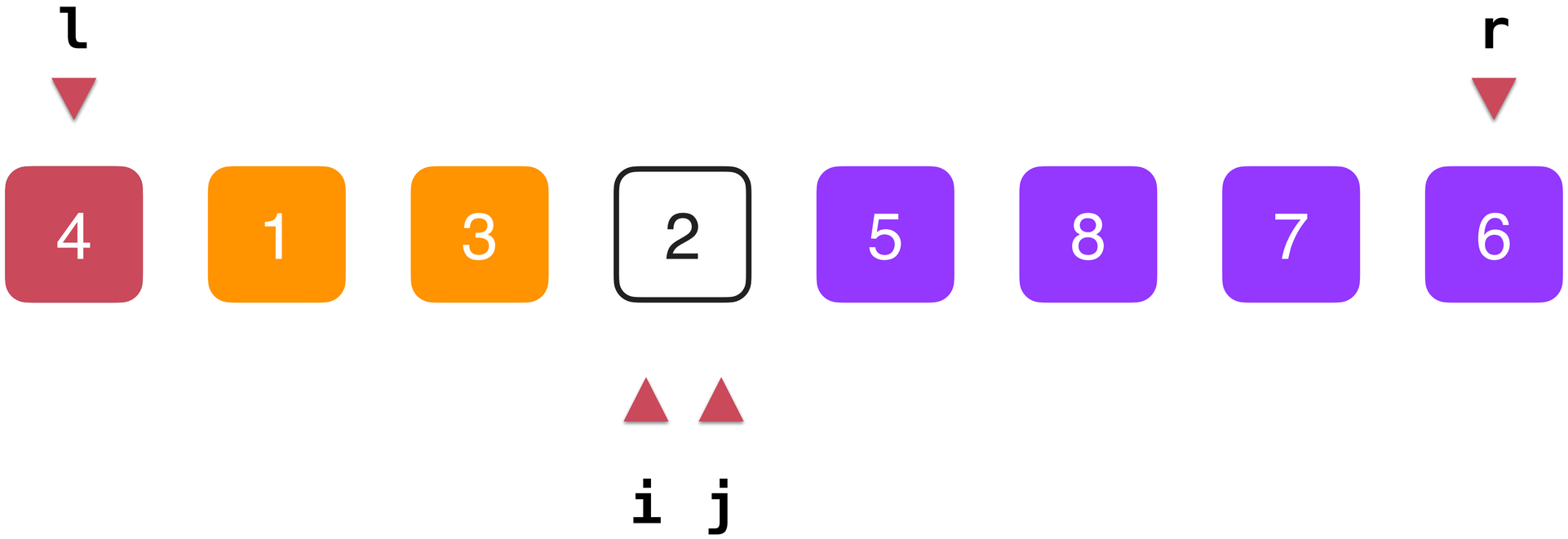
Partition2



`arr[l+1…i-1] <= v`

`arr[j+1…r] >= v`

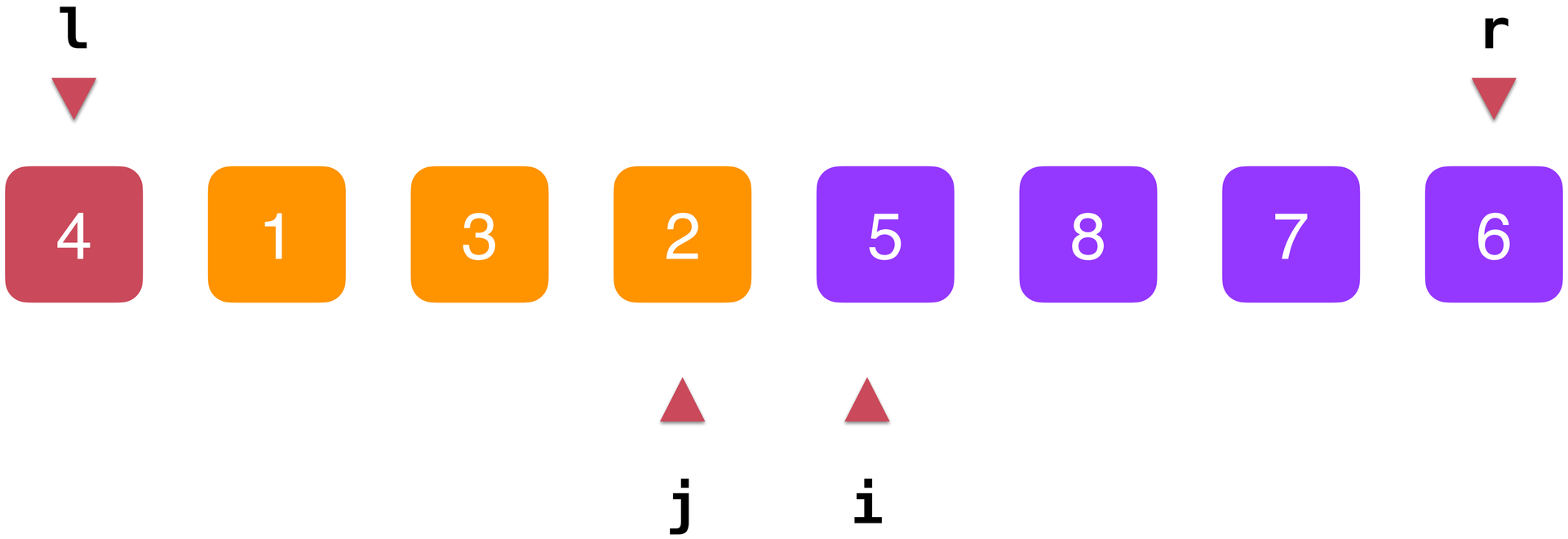
Partition2



`arr[l+1…i-1] <= v`

`arr[j+1…r] >= v`

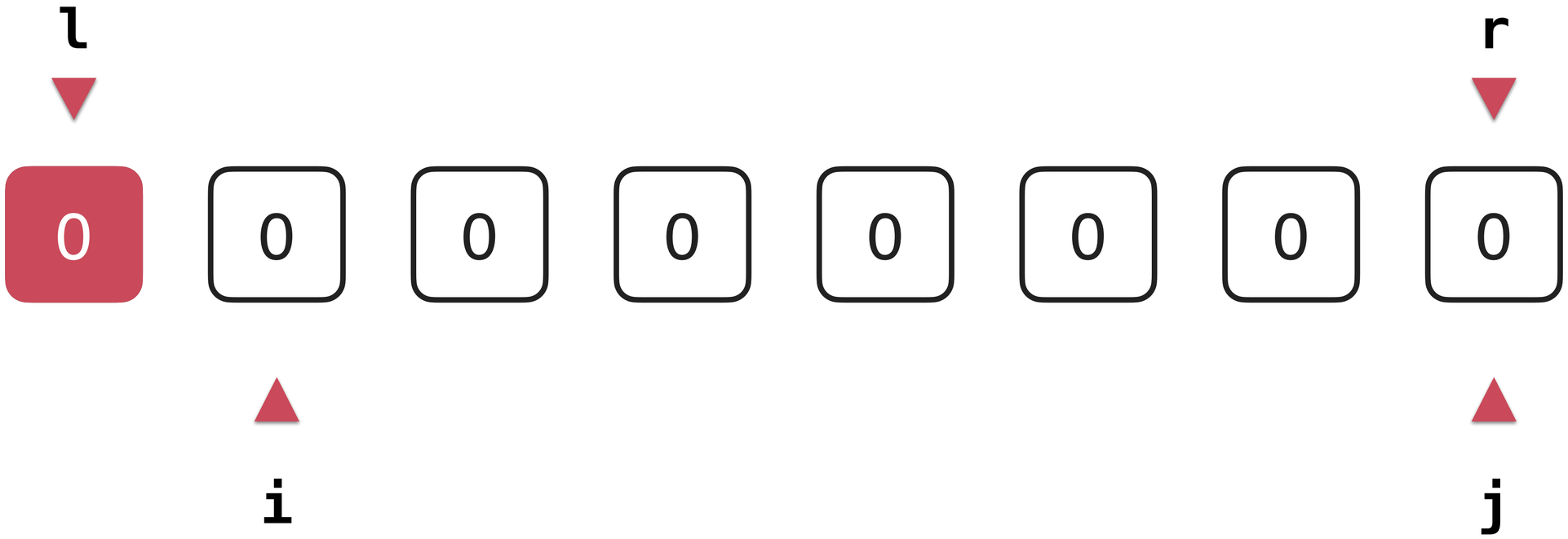
Partition2



`arr[l+1…i-1] <= v`

`arr[j+1…r] >= v`

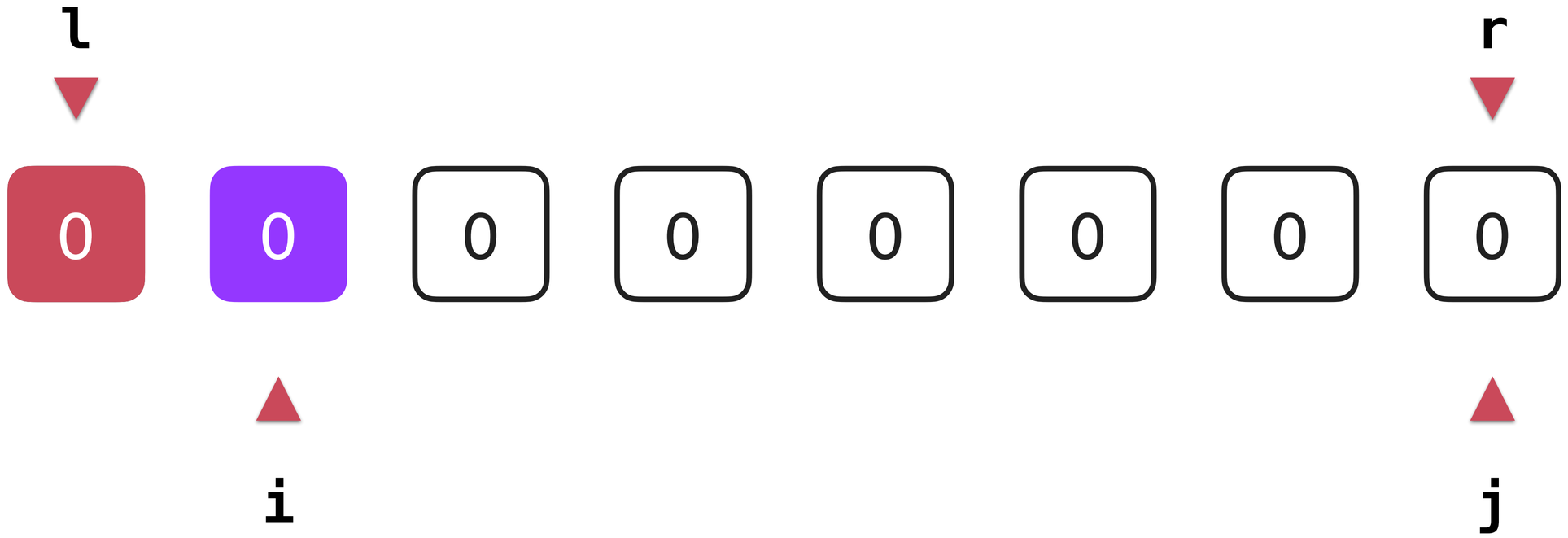
Partition2



`arr[l+1…i-1] <= v`

`arr[j+1…r] >= v`

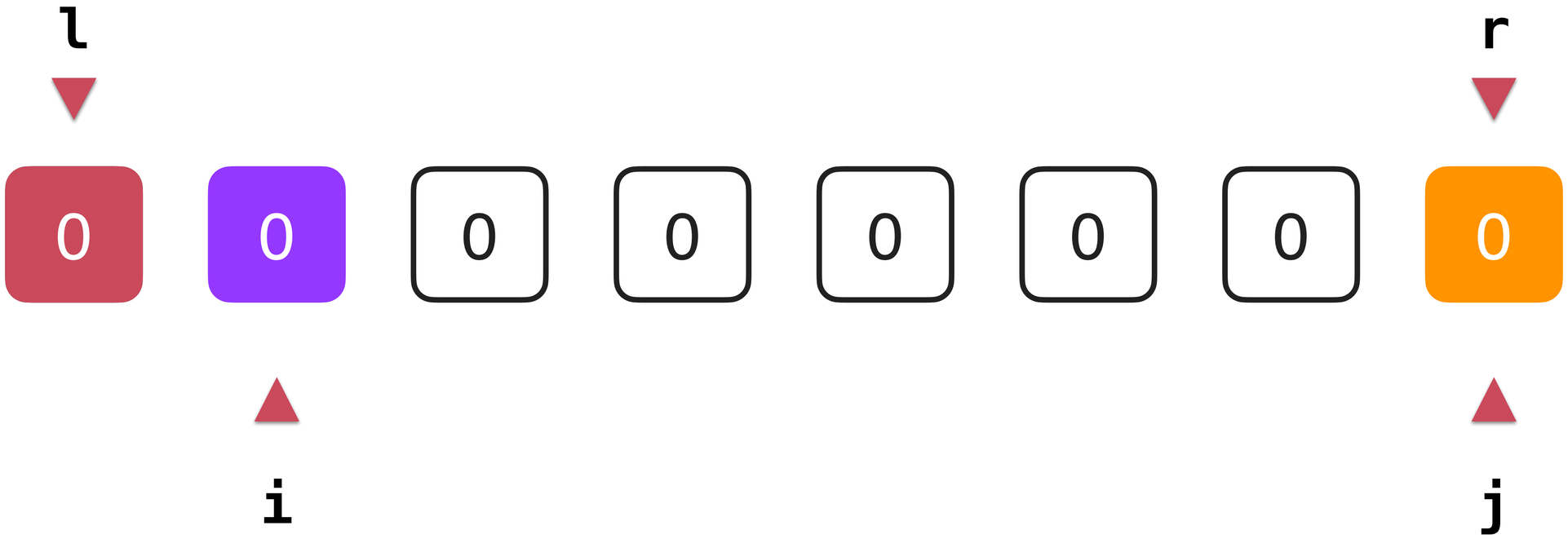
Partition2



`arr[l+1…i-1] <= v`

`arr[j+1…r] >= v`

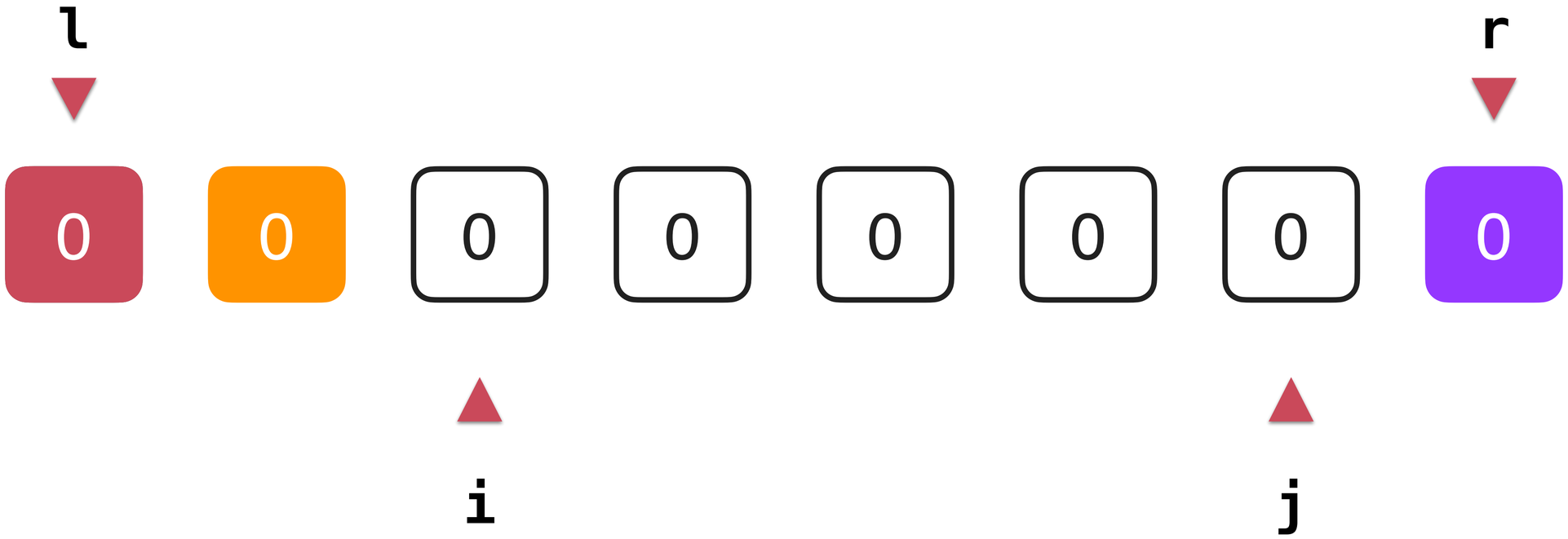
Partition2



`arr[l+1…i-1] <= v`

`arr[j+1…r] >= v`

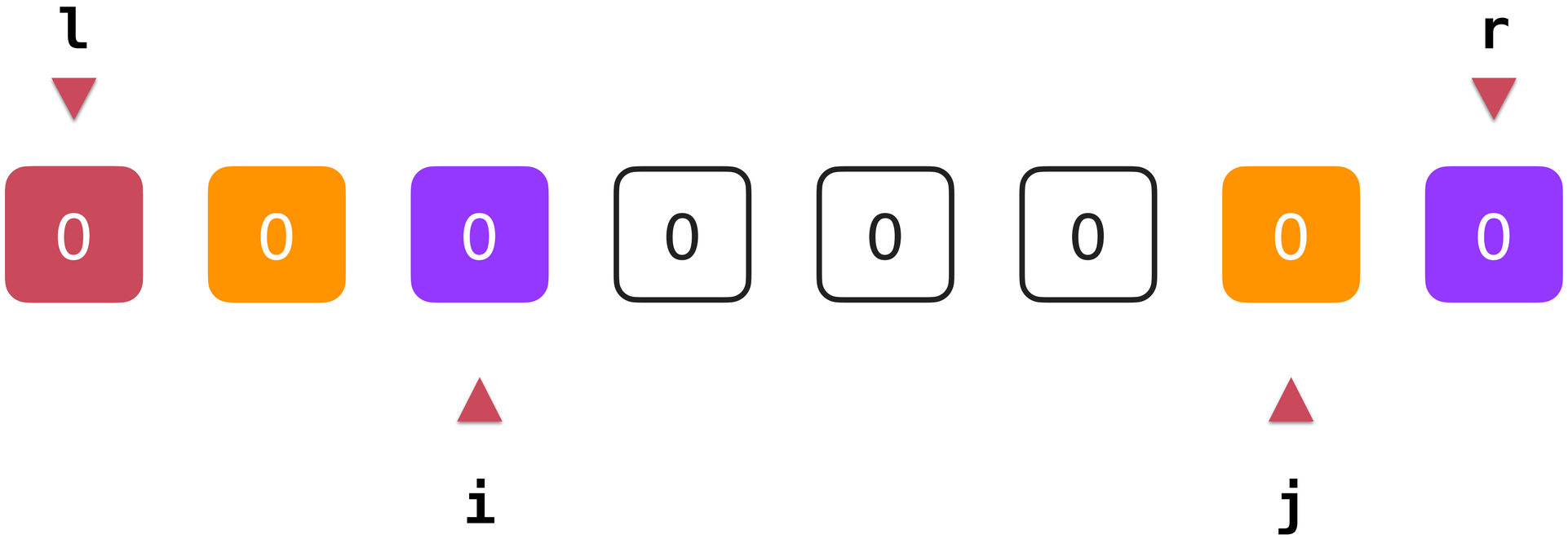
Partition2



`arr[l+1…i-1] <= v`

`arr[j+1…r] >= v`

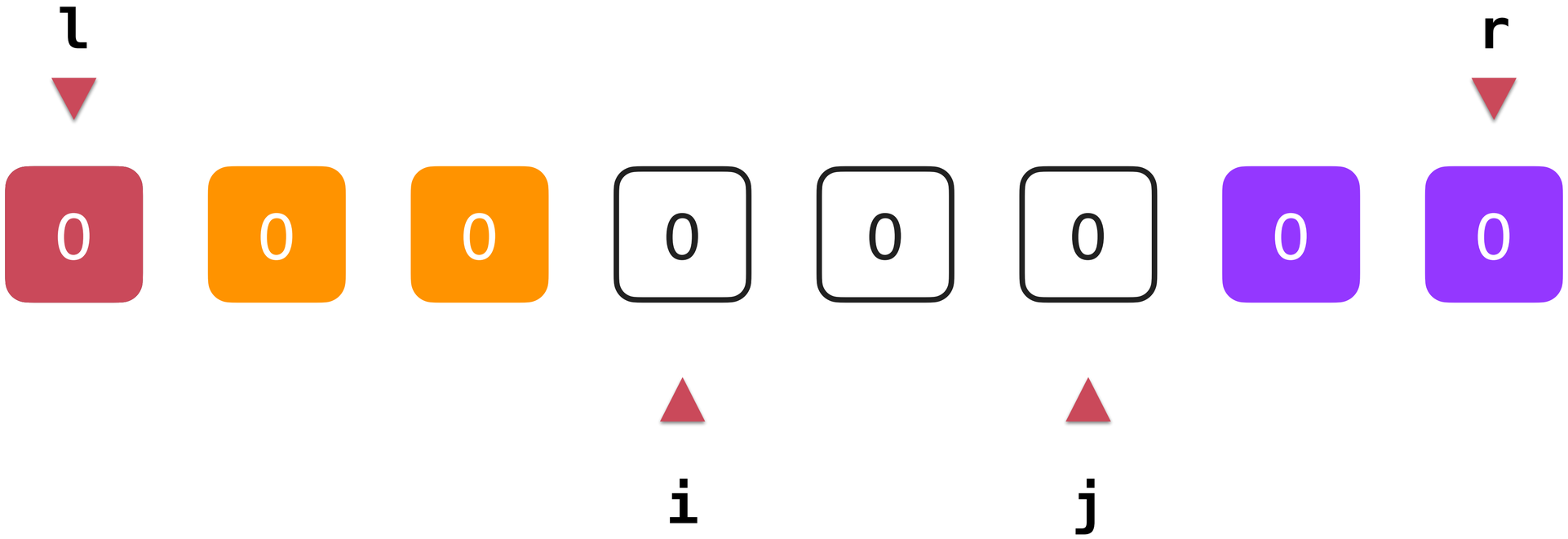
Partition2



`arr[l+1…i-1] <= v`

`arr[j+1…r] >= v`

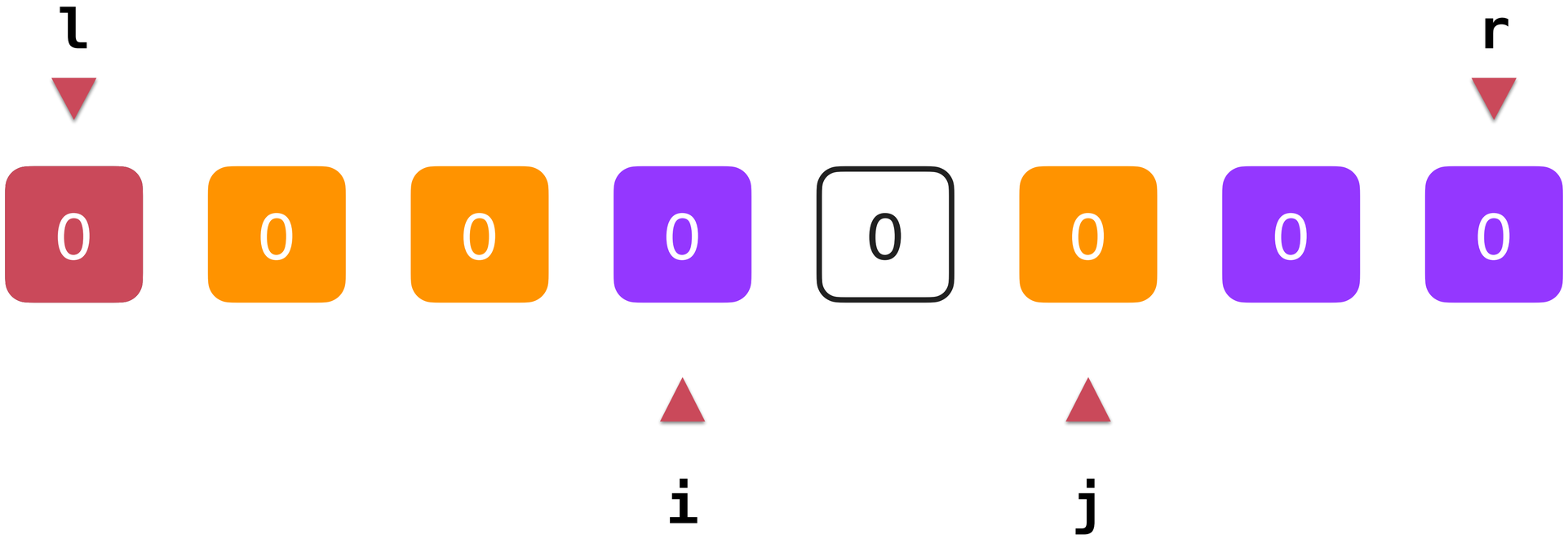
Partition2



`arr[l+1…i-1] <= v`

`arr[j+1…r] >= v`

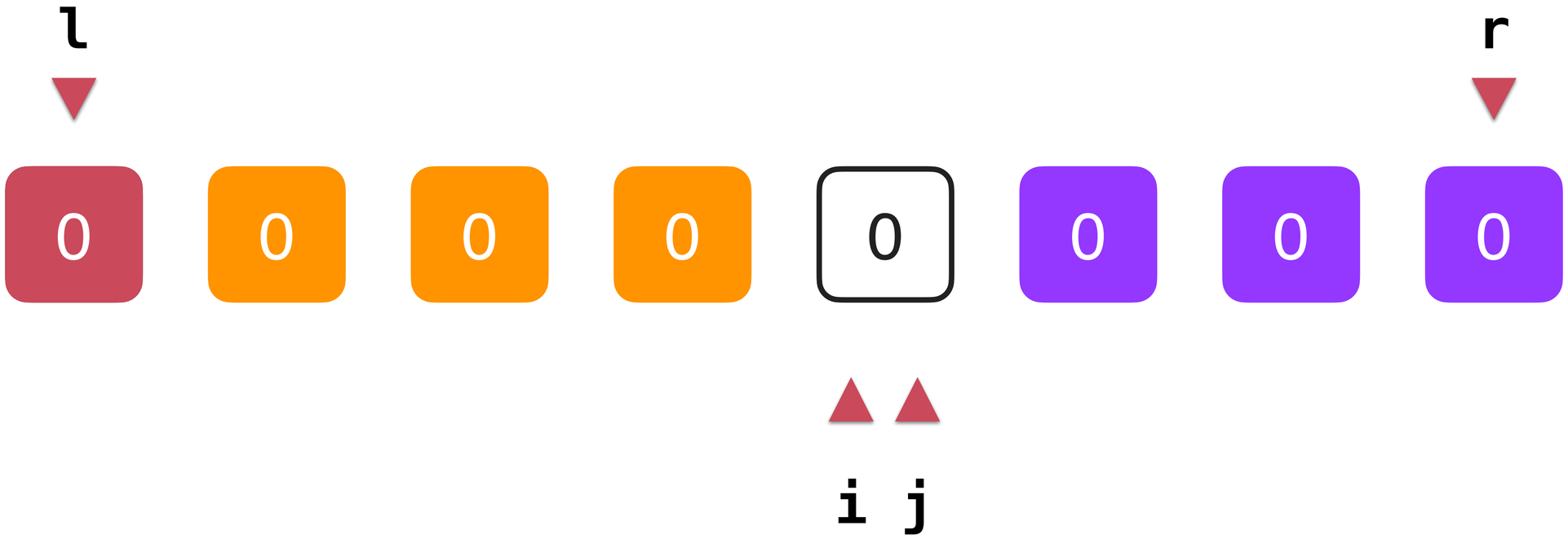
Partition2



`arr[l+1…i-1] <= v`

`arr[j+1…r] >= v`

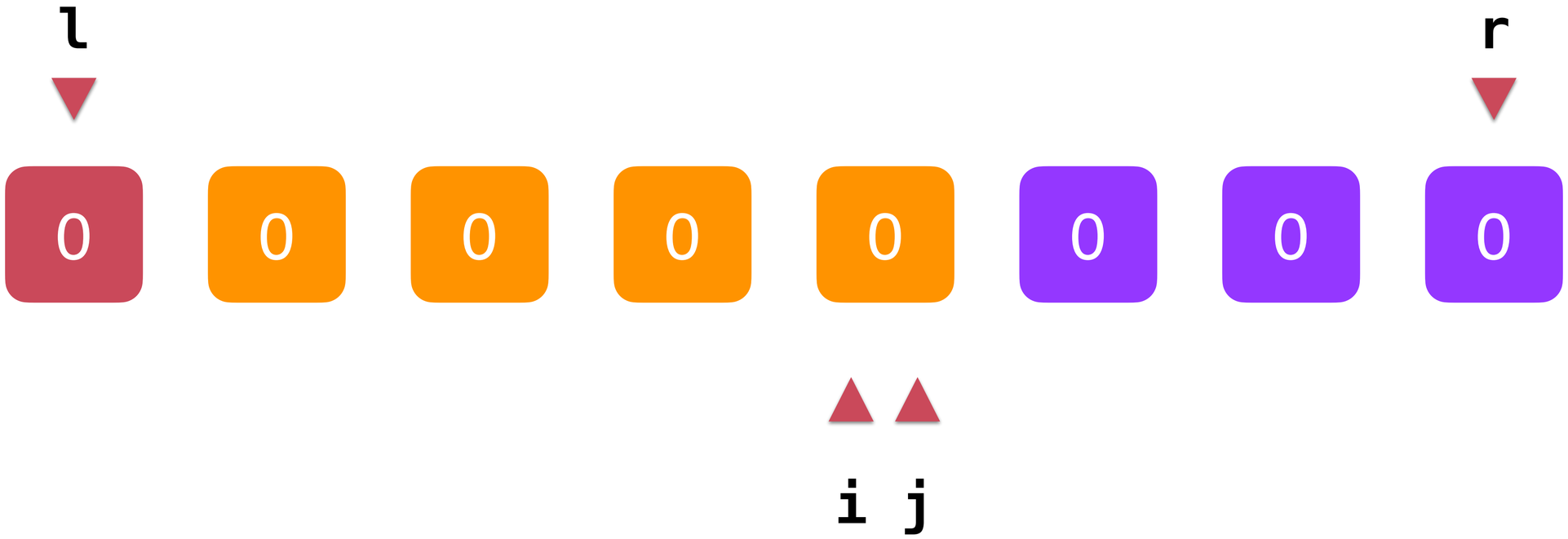
Partition2



arr[l+1…i-1] <= v

arr[j+1…r] >= v

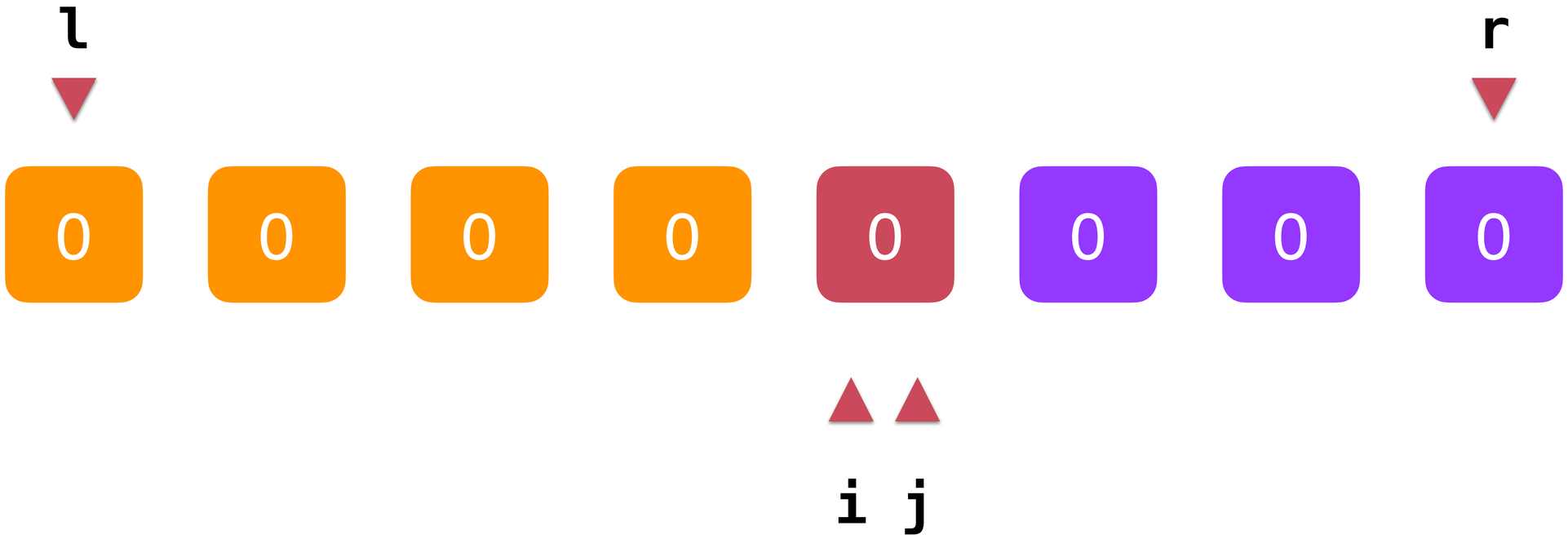
Partition2



`arr[l+1…i-1] <= v`

`arr[j+1…r] >= v`

Partition2



`arr[l+1…i-1] <= v`

`arr[j+1…r] >= v`

实现双路快速排序法

实践：双路快速排序法

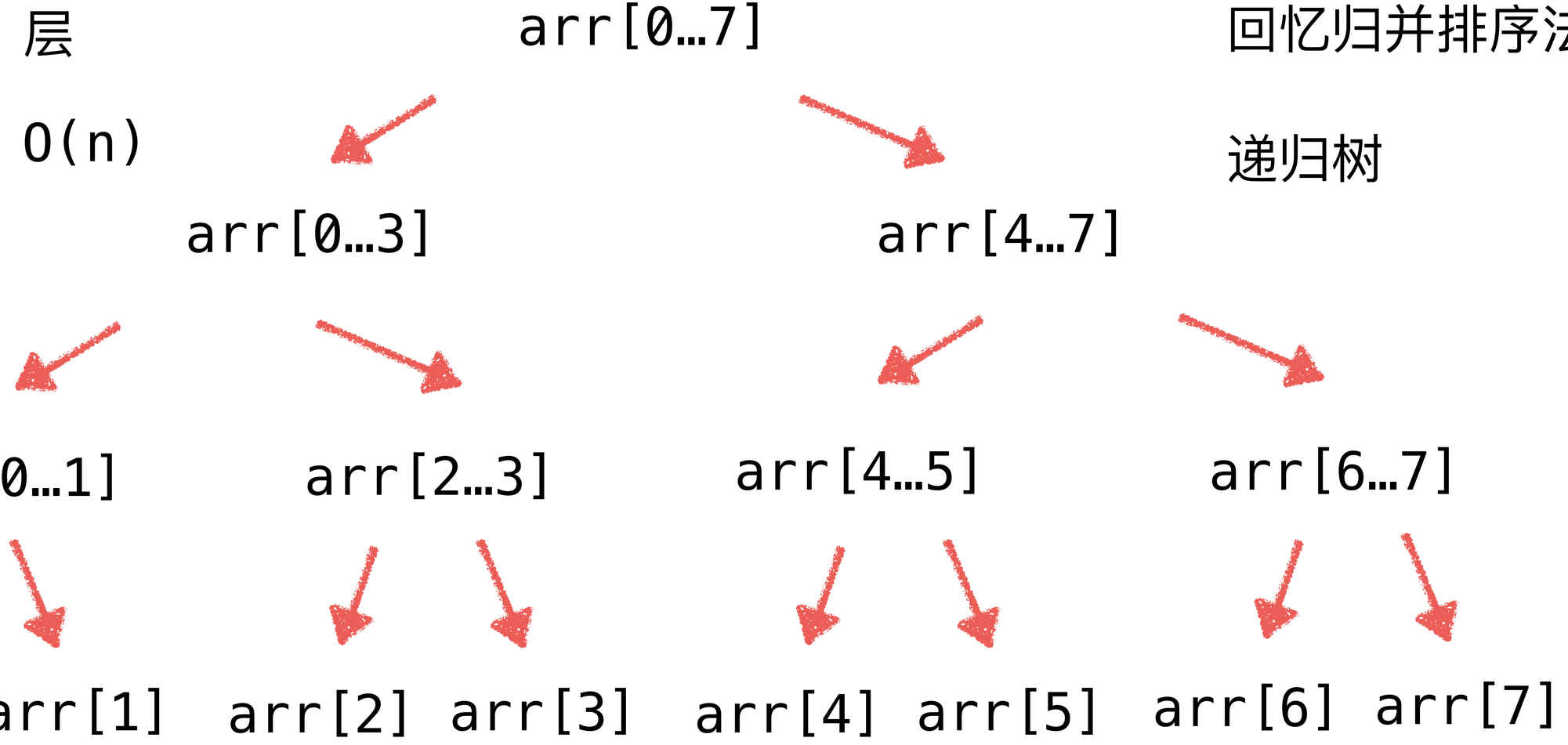
快速排序的复杂度分析

快速排序的复杂度分析

一共 logn 层

每层总共是 O(n)

O(nlogn)

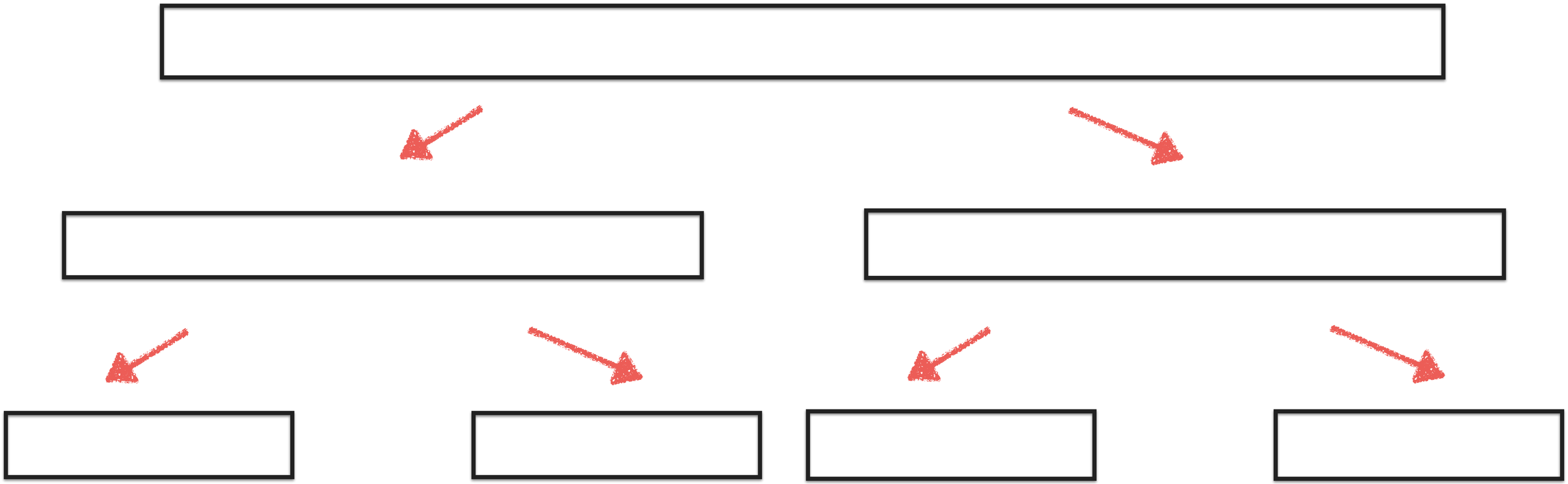


回忆归并排序法

递归树

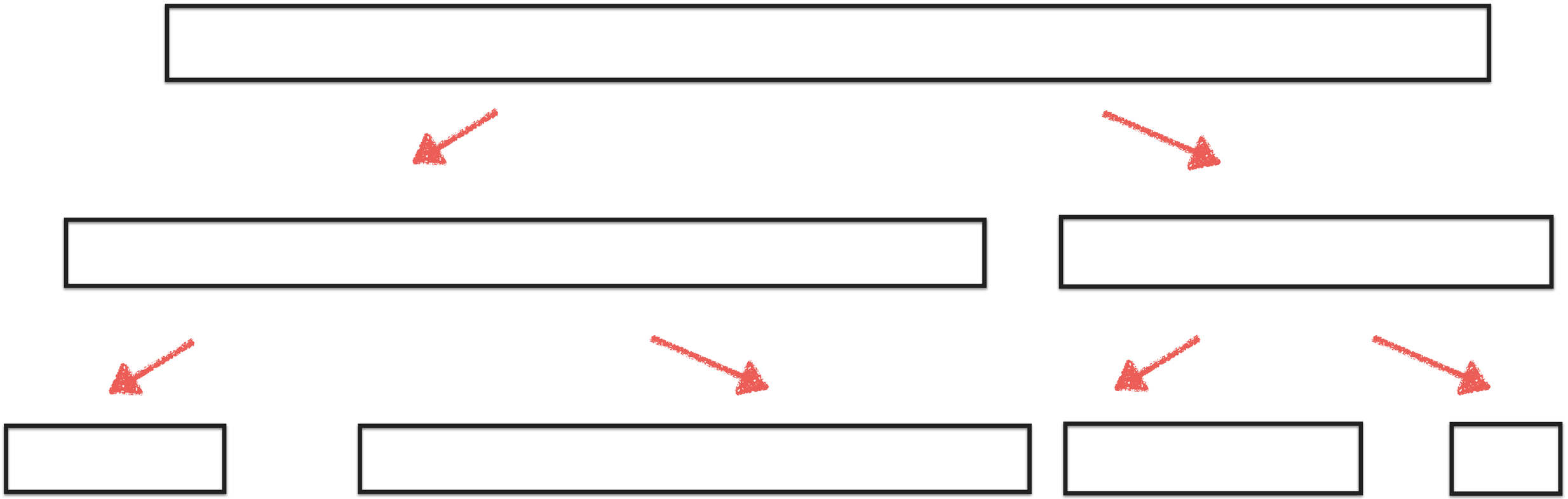
快速排序的复杂度分析

归并排序法



快速排序的复杂度分析

快速排序法



快速排序的复杂度分析

快速排序法

最坏复杂度　　O(n^2)

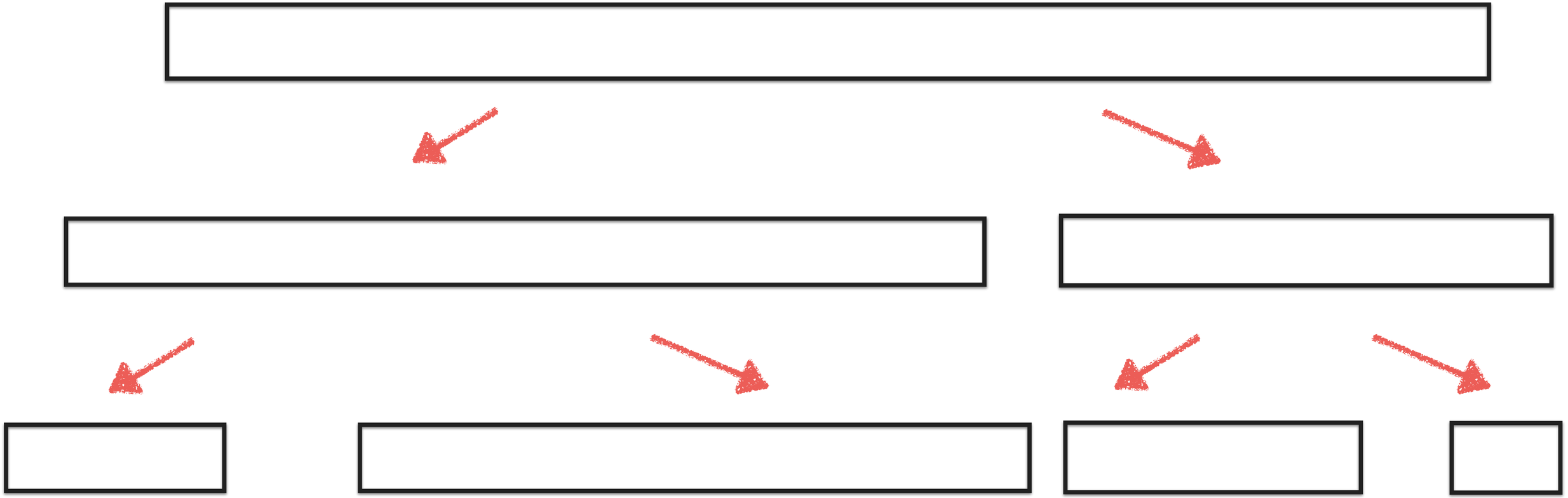
概率非常低

是一个随机算法

使用期望

快速排序的复杂度分析

数学期望的角度看：平分　　层数的期望值：O(logn)　　复杂度期望值：O(nlogn)



快速排序的复杂度分析

更严谨的数学推导：参考《算法导论》

快速排序的复杂度分析

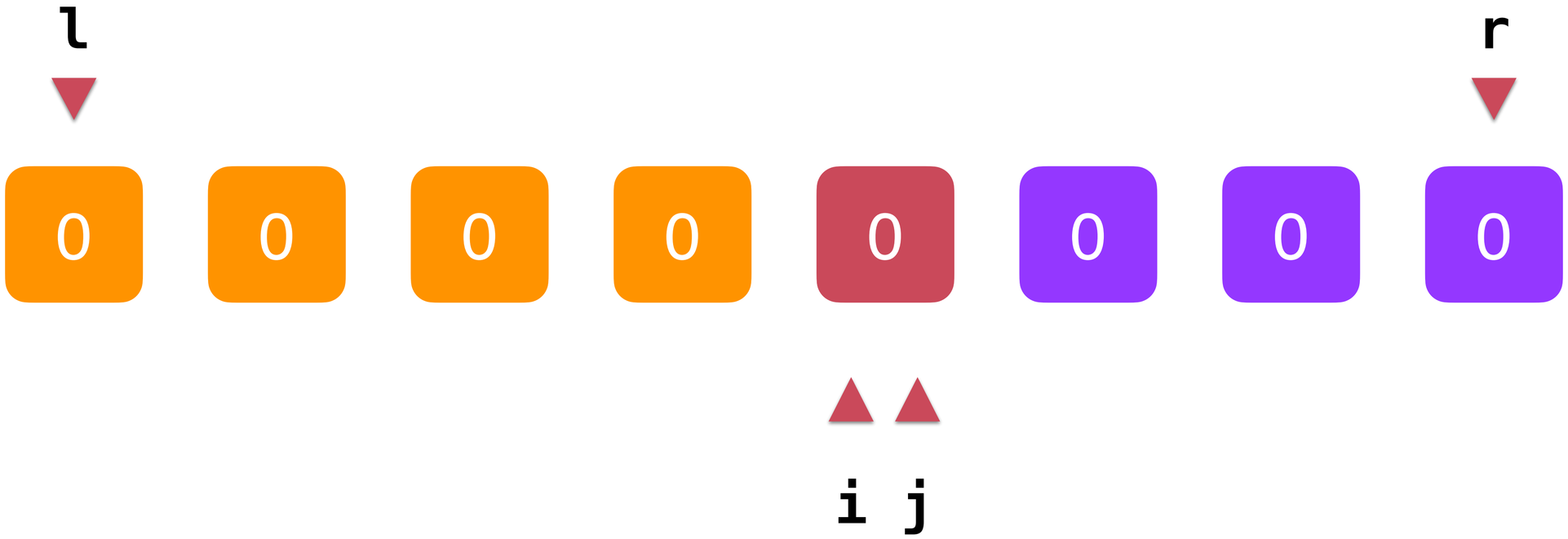
普通算法：看最差　　能找到一组数据 100% 恶化

随机算法：看期望　　没有一组数据能 100% 恶化

多次调用？尝试均摊分析

三路快速排序算法

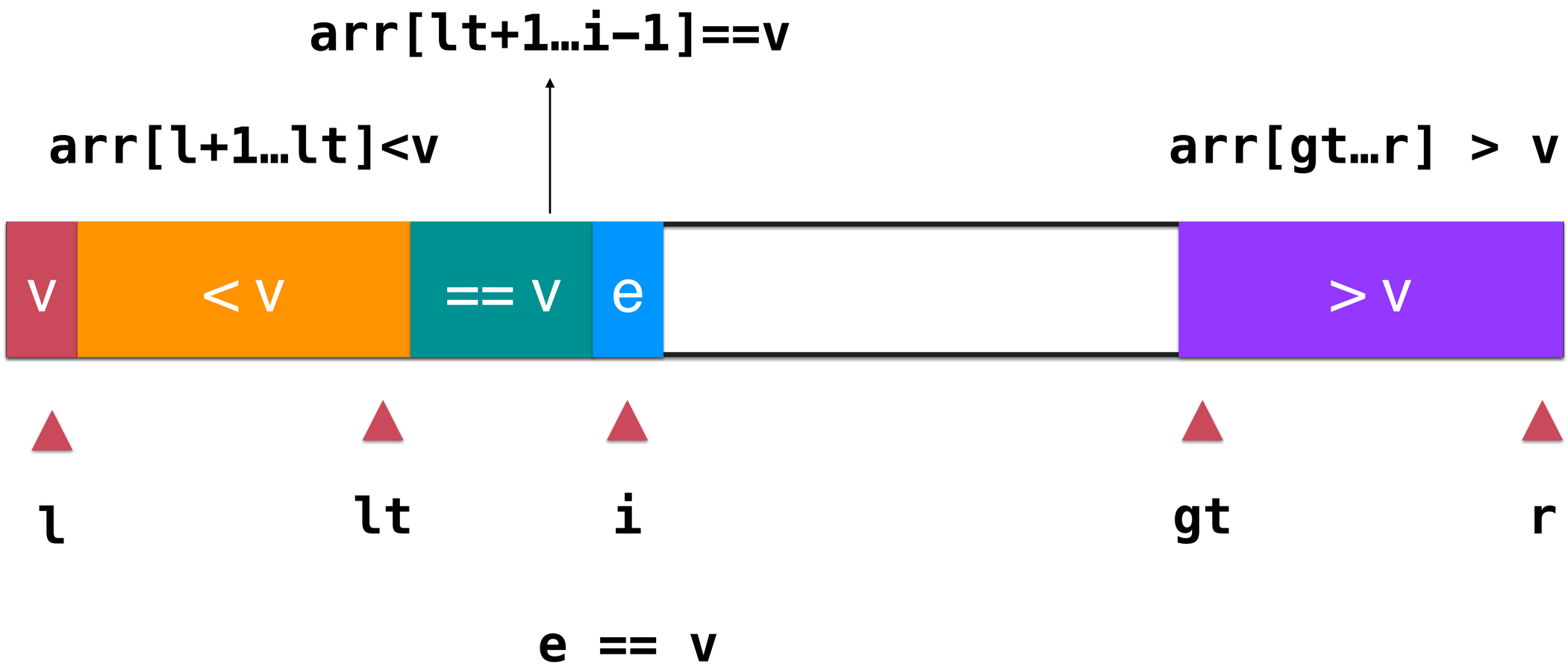
Partition2



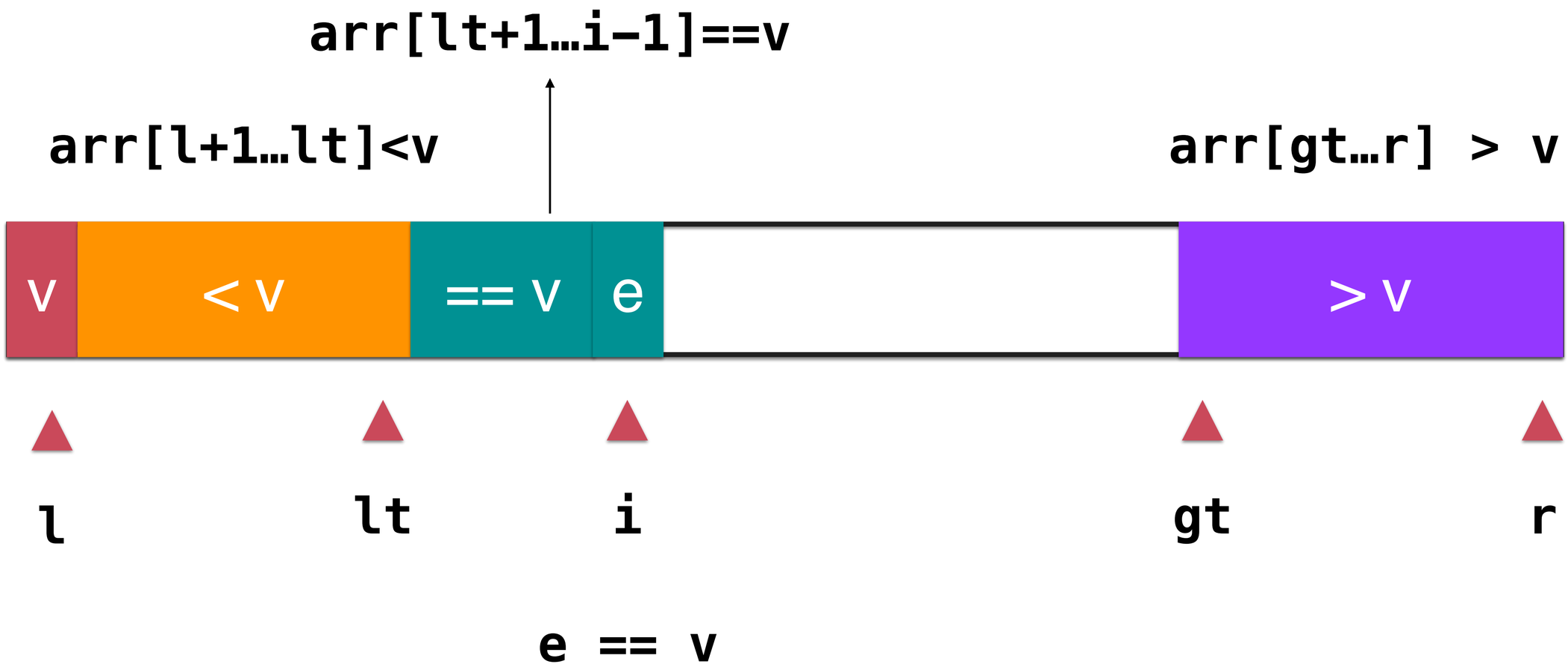
`arr[l+1…i-1] <= v`

`arr[j+1…r] >= v`

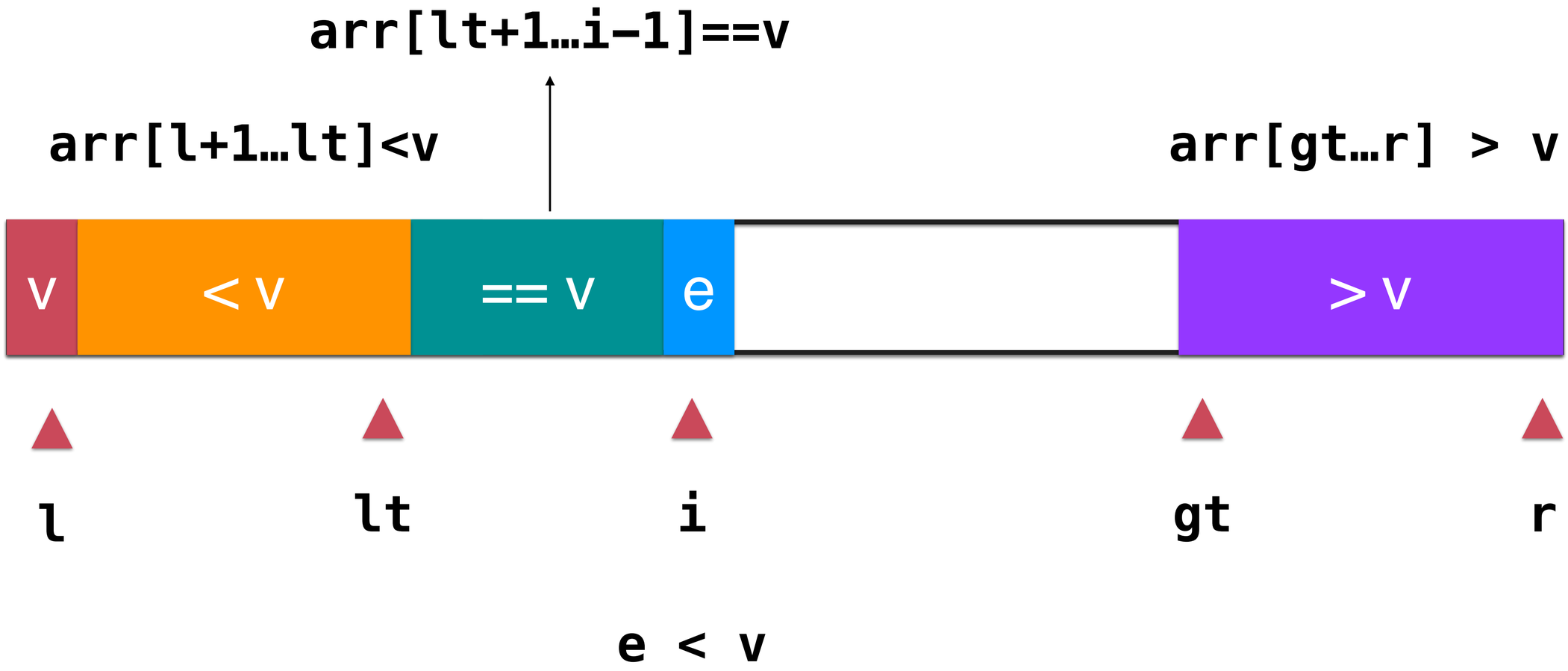
Quick Sort 3 Ways



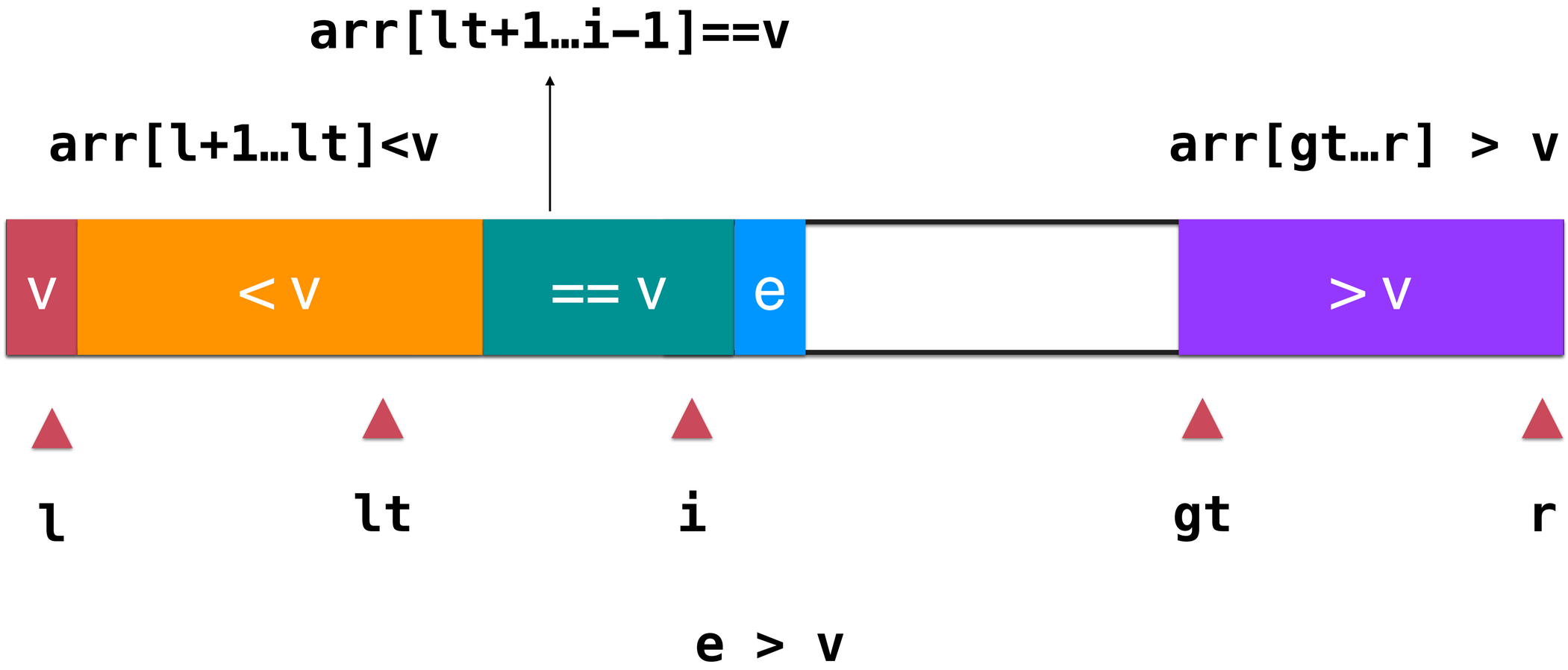
Quick Sort 3 Ways



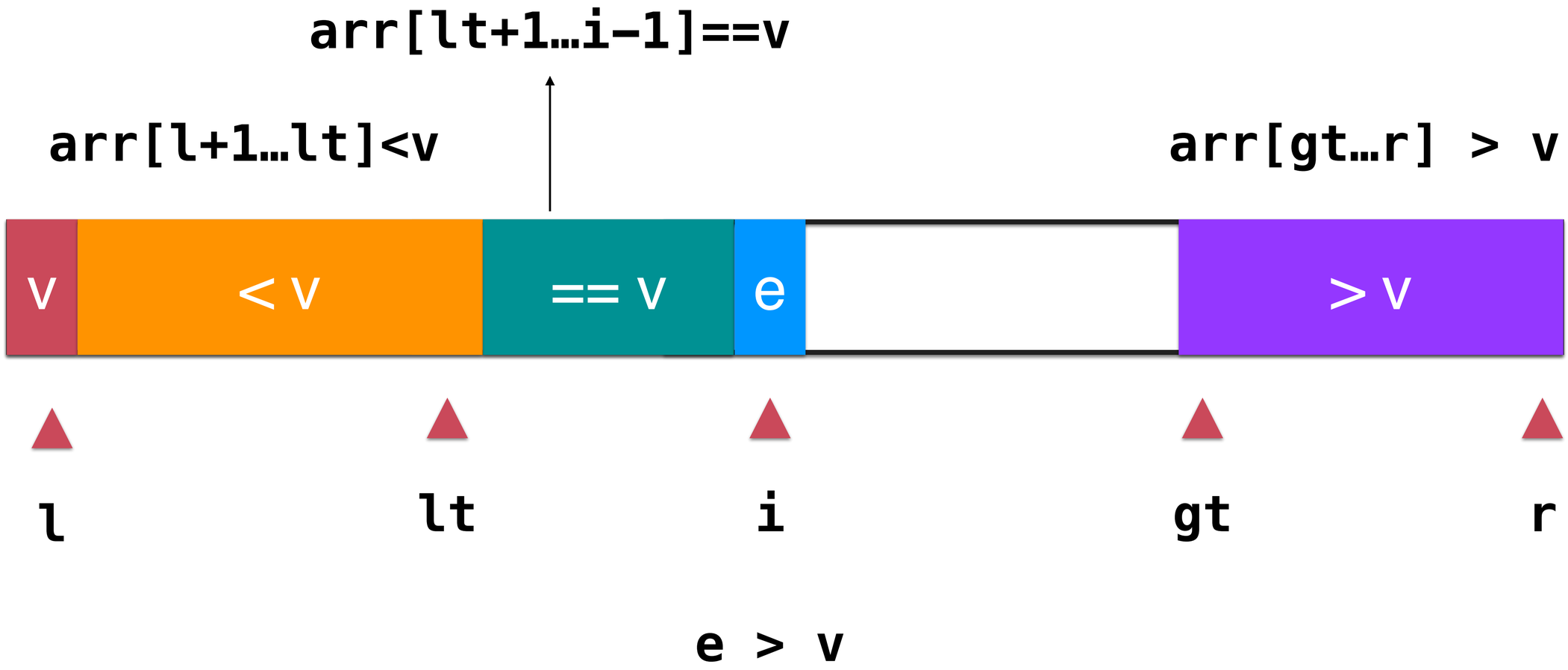
Quick Sort 3 Ways



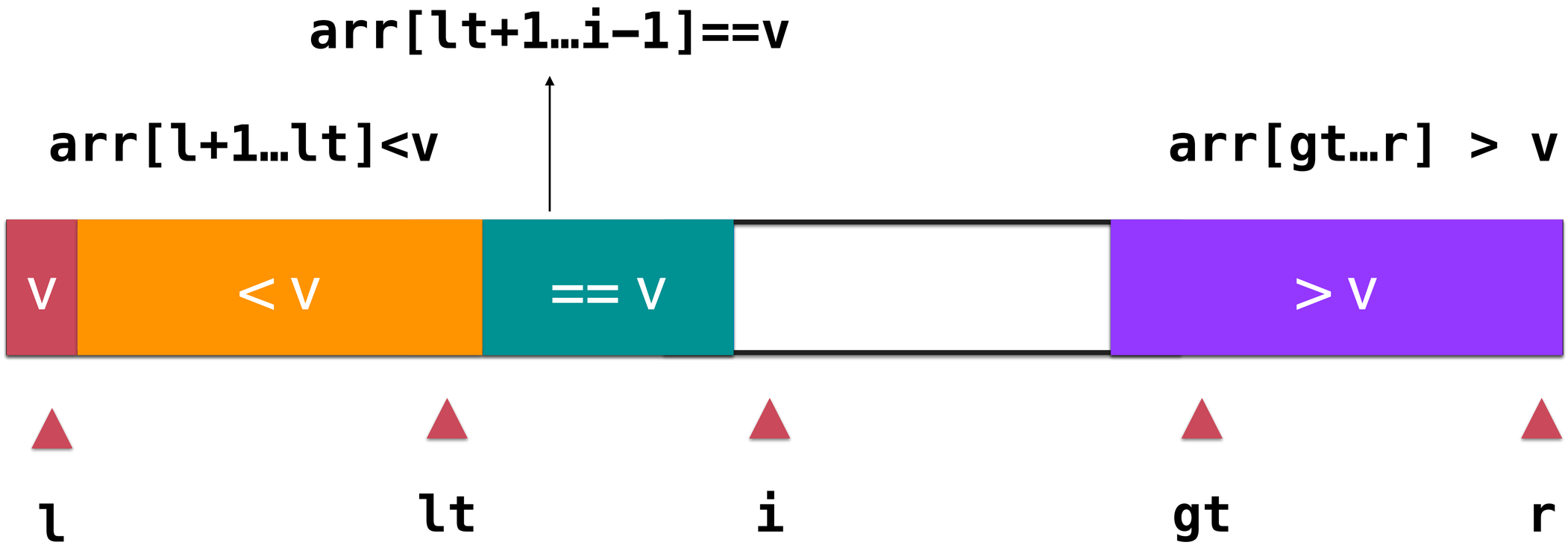
Quick Sort 3 Ways



Quick Sort 3 Ways

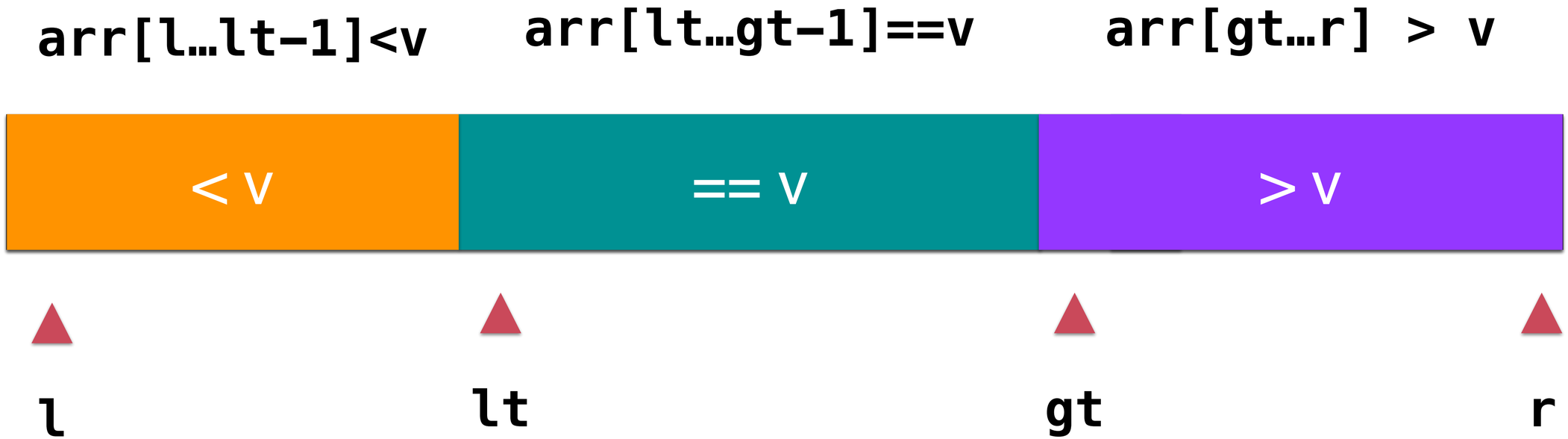


Quick Sort 3 Ways

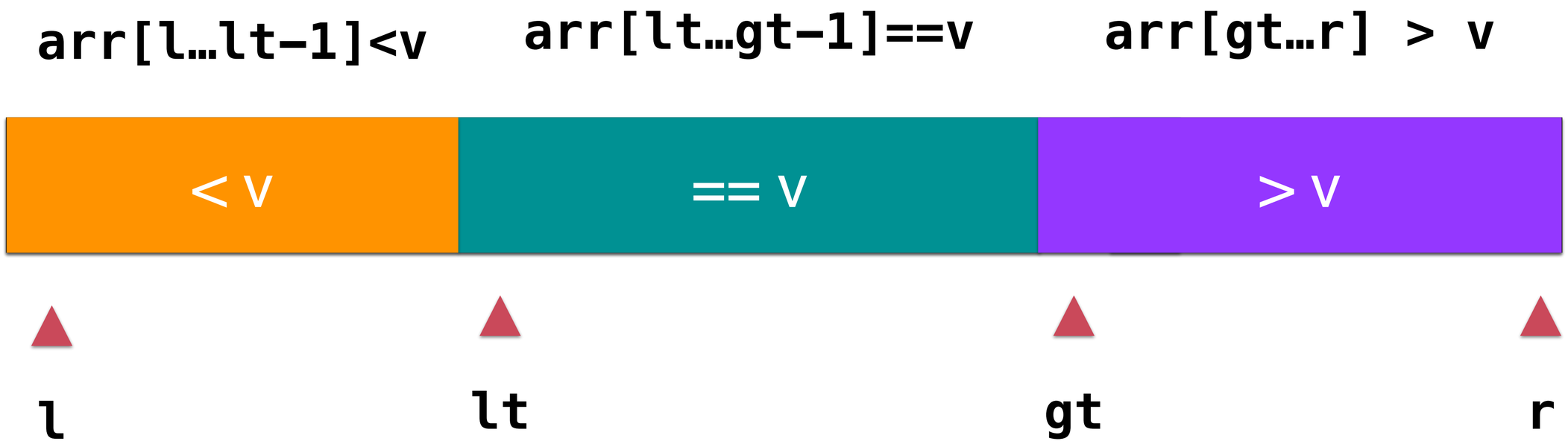


Quick Sort 3 Ways

Quick Sort 3 Ways



Quick Sort 3 Ways



所有元素都相同的数组：O(n)

实现三路快速排序算法

实践：实现三路快速排序算法

作业： Sort Colors

作业解析：Sort Colors

Select K

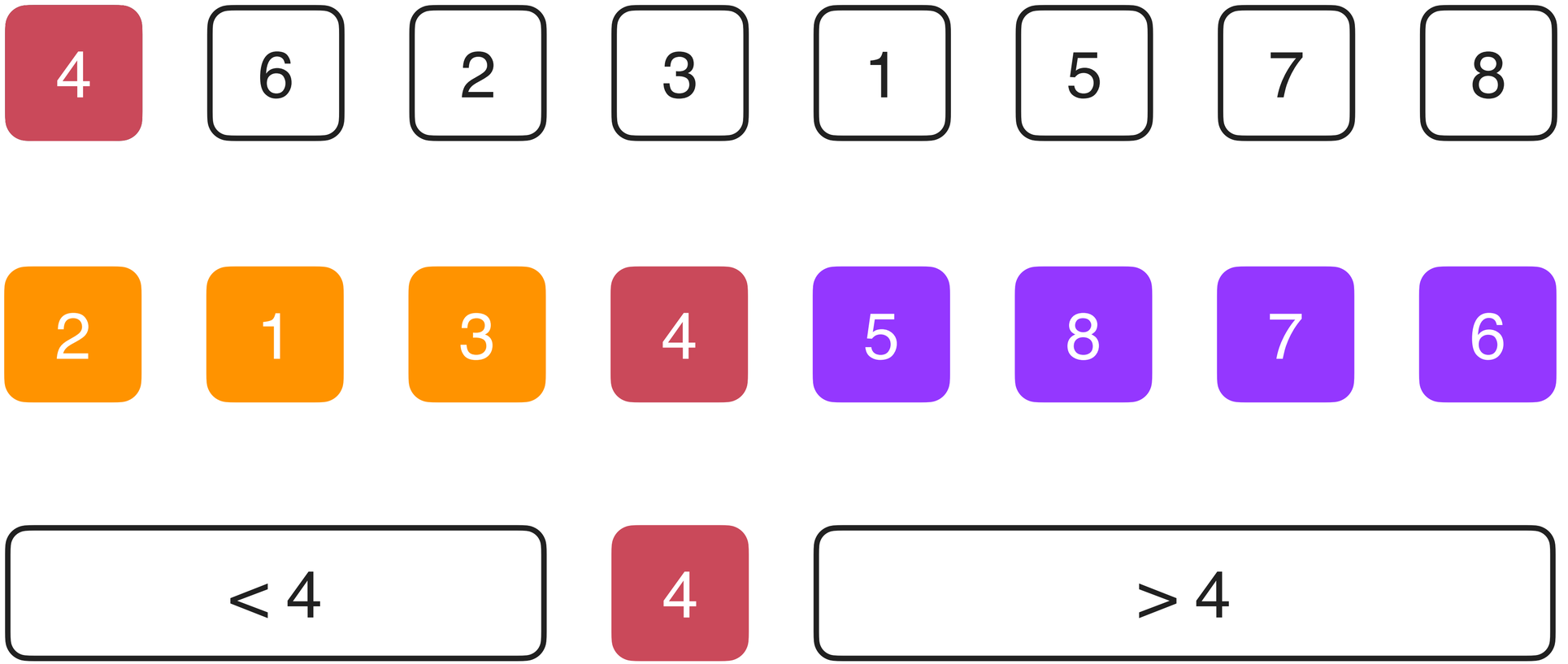
Select K

给出一个无序数组，找出数组的第 K 小的元素

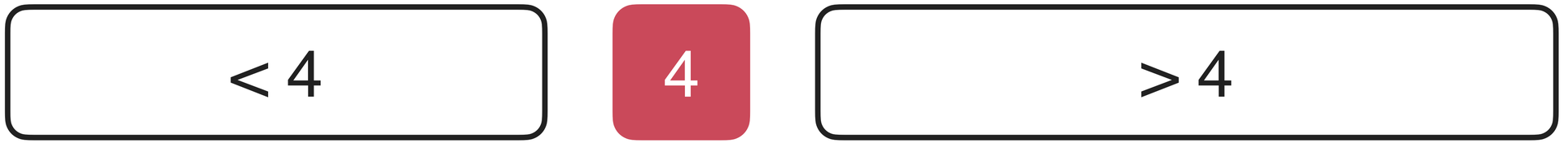
排序；arr[k - 1]　　O(nlogn)

使用快速排序算法的思路，可以在 O(n) 时间完成

Select K



Select K



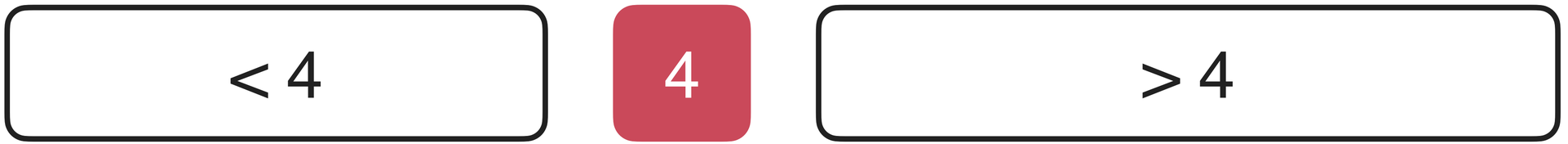
p

k == p?　　找到了！

k < p?　　到左边找

k > p?　　到右边找

Select K



k == p? **找到了！**

k < p? **到左边找**

k > p? **到右边找**

复杂度分析 期望

$$n + n/2 + n/4 + \dots + 1$$

$$=2n = O(n)$$

Select K

https://leetcode-cn.com/problems/zui-xiao-de-kge-shu-lcof/

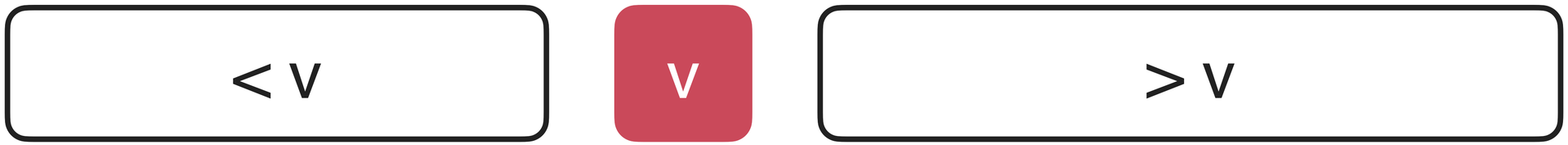
https://leetcode-cn.com/problems/kth-largest-element-in-an-array/

作业解析：Select K

快速排序算法总结

快速排序法总结

快速排序整体思想：



快速排序法总结

单路快速排序

arr[l+1…j] < v　　arr[j+1…i-1] > v

快速排序法总结

单路快速排序

完全有序的数据退化

引入随机化

快速排序法总结

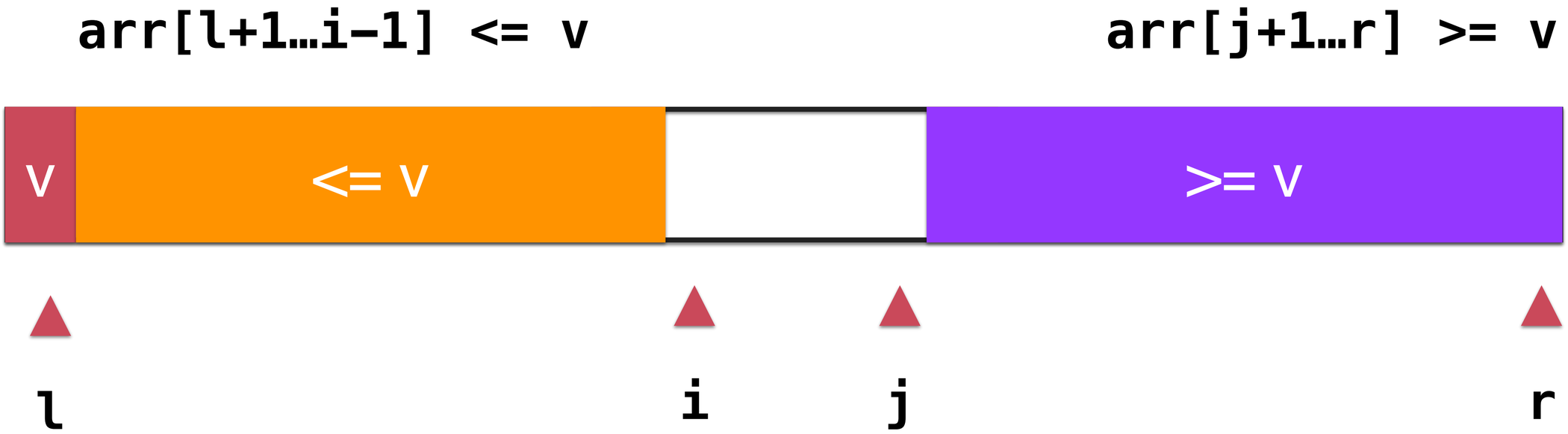
引入随机化

所有元素一样的数据退化

双路快速排序

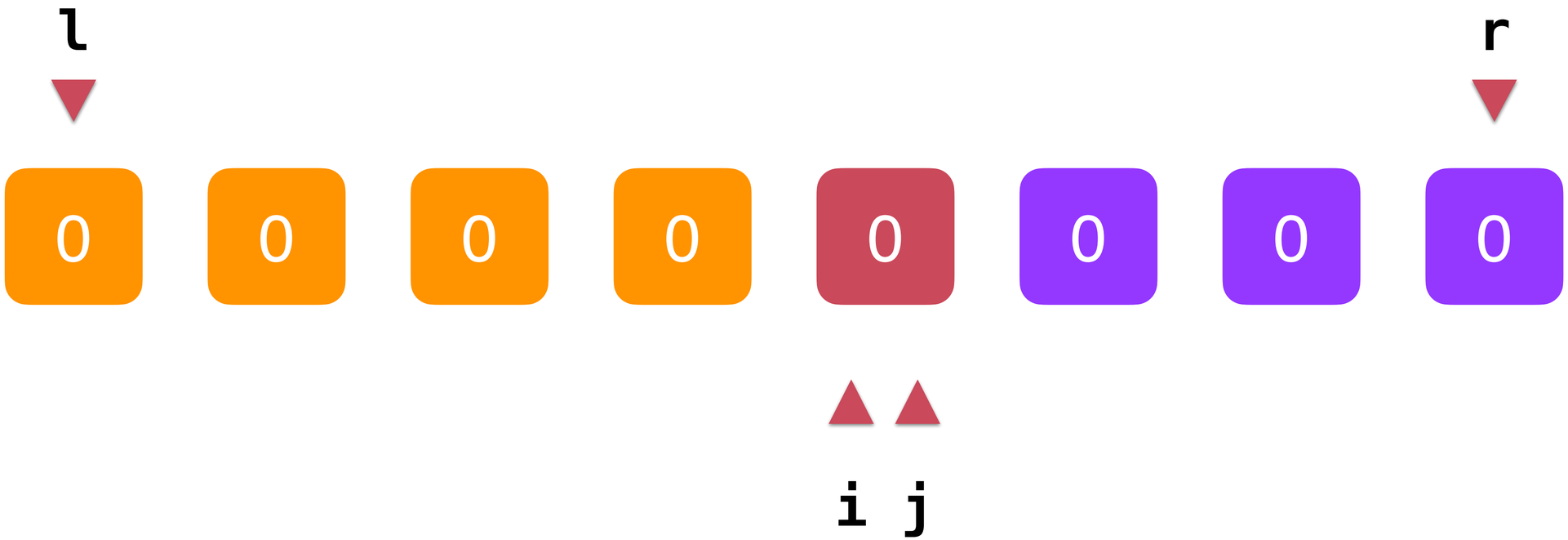
快速排序法总结

双路快速排序



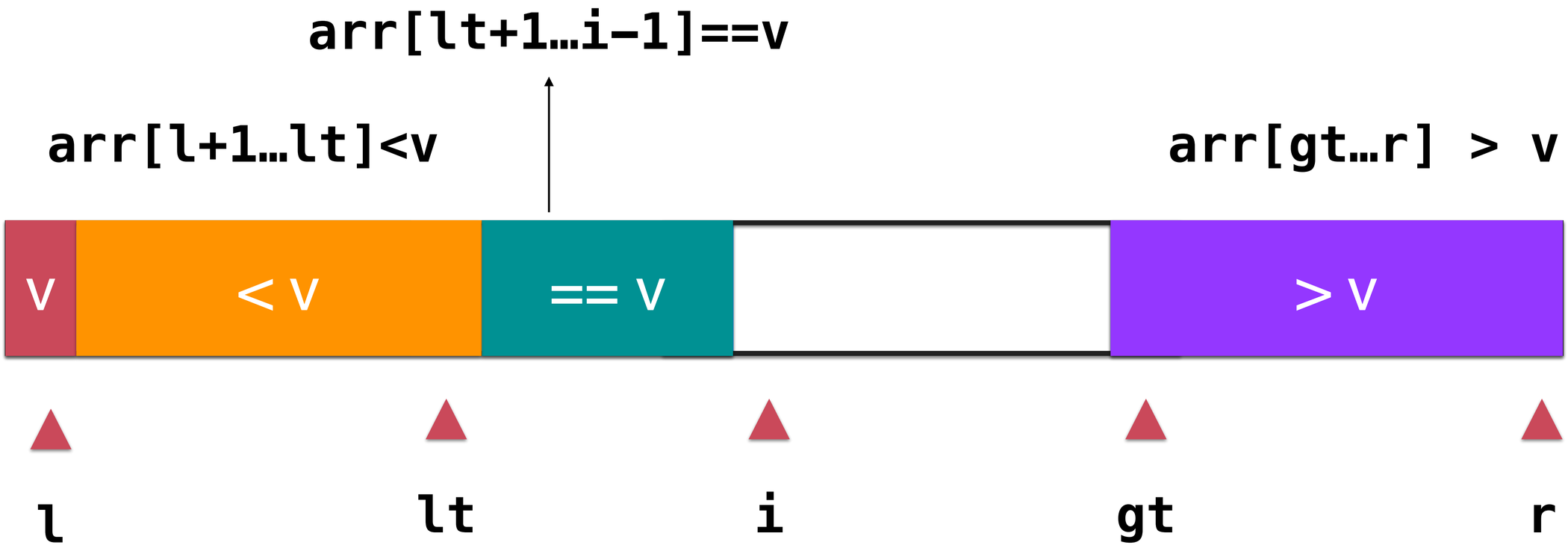
快速排序法总结

双路快速排序



快速排序法总结

三路快速排序　　所有元素都一样的数据：O(n)



快速排序法总结

单路快速排序

随机化

双路快速排序

三路快速排序

快速排序法总结

普通算法：看最差　　能找到一组数据 100% 恶化

随机算法：看期望　　没有一组数据能 100% 恶化

多次调用？ 尝试均摊分析

快速排序法总结

快速排序算法的思想:

排序　　SelectK

其他

欢迎大家关注我的个人公众号：是不是很酷

算法与数据结构体系课程

liuyubobobo