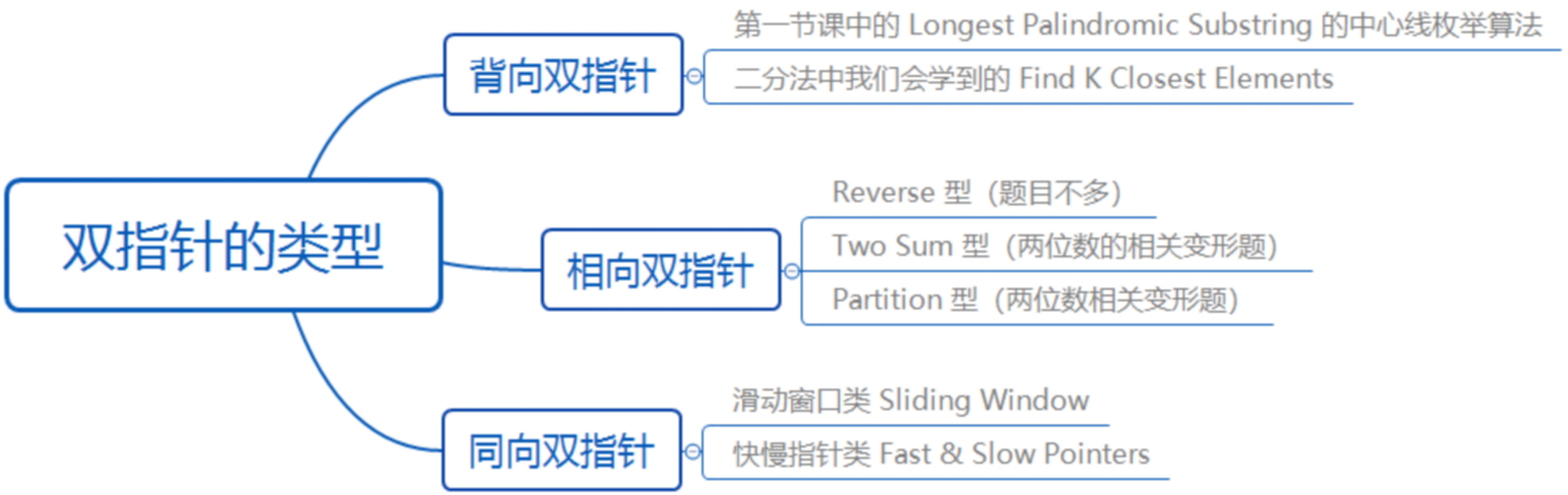
高频算法之王——
双指针算法之相向双指针

主讲人 令狐冲
课程版本 v7.0

版权声明

双指针的类型


双指针的类型
背向双指针
第一节课中的 Longest Palindromic Substring 的中心线枚举算法
二分法中我们会学到的 Find K Closest Elements
相向双指针
Reverse 型（题目不多）
Two Sum 型（两位数的相关变形题）
Partition 型（两位数相关变形题）
同向双指针
滑动窗口类 Sliding Window
快慢指针类 Fast & Slow Pointers

相向双指针

两根指针一头一尾，向中间靠拢直到相遇

时间复杂度 O(n)

Two Sum 类

先修内容中我们已经讲解了双指针的经典题 Two Sum
接下来我们来看这类问题可能的变化

Two Sum - Data Structure Design

http://www.lintcode.com/problem/two-sum-data-structure-design/

http://www.jiuzhang.com/solutions/two-sum-data-structure-design/

Two Sum - Unique pairs

https://www.lintcode.com/problem/two-sum-unique-pairs/

https://www.jiuzhang.com/solutions/two-sum-unique-pairs/

问：是否可以先去重？

3Sum

https://www.lintcode.com/problem/3sum/

https://www.jiuzhang.com/solutions/3sum/

统计所有的和为 0 的三元组 (Triples)

Triangle Count

https://www.lintcode.com/problem/triangle-count/

https://www.jiuzhang.com/solutions/triangle-count/

Two Sum 计数问题

统计所有和 <= target 的配对数

http://www.lintcode.com/problem/two-sum-less-than-or-equal-to-target/

http://www.jiuzhang.com/solutions/two-sum-less-than-or-equal-to-target/

统计所有和 >= target 的配对数

http://www.lintcode.com/en/problem/two-sum-greater-than-target/

http://www.jiuzhang.com/solutions/two-sum-greater-than-target/

Two Sum - Closest to Target

https://www.lintcode.com/problem/two-sum-closest-to-target/

https://www.jiuzhang.com/solutions/two-sum-closest-to-target/

3Sum Closest

http://www.lintcode.com/problem/3sum-closest/

http://www.jiuzhang.com/solutions/3sum-closest/

4 Sum

https://www.lintcode.com/problem/4sum/description

https://www.jiuzhang.com/solutions/4sum

在数组中求 a + b + c + d = target 的所有四元组

4 Sum II

https://www.lintcode.com/problem/4sum-ii/description

https://www.jiuzhang.com/solutions/4sum-ii

在4个数组中，分别取4个数，使得和为 target

求满足条的四元组个数

k数之和

https://www.lintcode.com/problem/k-sum/description 求方案总数

https://www.lintcode.com/problem/k-sum-ii/description 求具体方案

敬请期待在动态规划和深度优先搜索中对这两个问题的讲解

休息一会儿

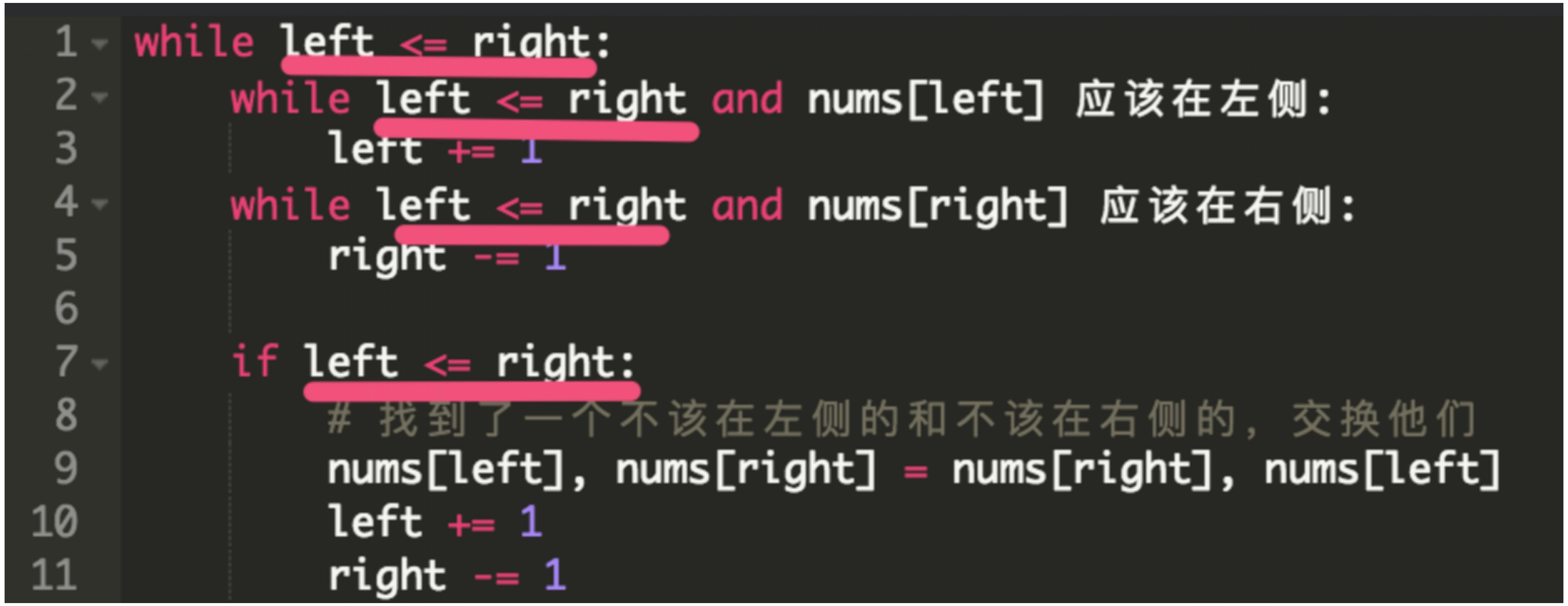
Take a break

Partition Array

https://www.lintcode.com/problem/partition-array/

https://www.jiuzhang.com/solutions/partition-array/

Partition 模板



```
while left <= right:
    while left <= right and nums[left] 应该在左侧:
        left += 1
    while left <= right and nums[right] 应该在右侧:
        right -= 1

    if left <= right:
        # 找到了一个不该在左侧的和不该在右侧的，交换他们
        nums[left], nums[right] = nums[right], nums[left]
        left += 1
        right -= 1
```

独孤九剑 —— 破鞭式

时间复杂度与最内层循环主体的执行次数有关
与有多少重循环无关

Interleaving positive and negative numbers

http://www.lintcode.com/problem/interleaving-positive-and-negative-numbers/

http://www.jiuzhang.com/solutions/interleaving-positive-and-negative-numbers/

将一个数组中的元素正负交替排列

数据确保正负数个数相差不超过1

do it in-place

Related Questions

- **Partition Array by Odd and Even**
- http://www.lintcode.com/problem/partition-array-by-odd-and-even/
- http://www.jiuzhang.com/solutions/partition-array-by-odd-and-even/

- **Sort Letters by Case**
- http://www.lintcode.com/problem/sort-letters-by-case/
- http://www.jiuzhang.com/solutions/sort-letters-by-case/

Sort Colors

http://www.lintcode.com/problem/sort-colors/

http://www.jiuzhang.com/solutions/sort-colors/

如果你不会 3-part partition 的算法

是否可以用 2-part partition 解决?

排序 Rainbow Sort

https://www.lintcode.com/problem/sort-colors-ii/

https://www.jiuzhang.com/solutions/sort-colors-ii/

问：猜一猜最优的时间复杂度？

其他有趣的排序

烙饼排序 Pancake Sort（有可能会考哦）

https://en.wikipedia.org/wiki/Pancake_sorting

http://www.geeksforgeeks.org/pancake-sorting/

睡眠排序 Sleep Sort

https://rosettacode.org/wiki/Sorting_algorithms/Sleep_sort

面条排序 Spaghetti Sort

https://en.wikipedia.org/wiki/Spaghetti_sort

猴子排序 Bogo Sort

https://en.wikipedia.org/wiki/Bogosort

Move Zeroes

http://www.lintcode.com/problem/move-zeroes/

http://www.jiuzhang.com/solution/move-zeroes

将数组中非 0 的元素移动到数组的后半部分

确保数组的“修改”次数最少

两种问法

如果不需要维持原来数组中元素的相对顺序，最优算法是什么?

如果需要维持原来数组的相对顺序，最优算法是什么?

同向双指针

请在第10章和第11章的互动课中学习

Thank You

Q & A