

两根指针

九章算法强化班 第4章



扫描二维码关注微信/微博
获取最新面试题及权威解答

微信: [ninechapter](#)

微博: <http://www.weibo.com/ninechapter>

知乎: <http://zhuoanlan.zhihu.com/jiuzhang>

官网: <http://www.jiuzhang.com>

1. 一个数组, 从两边往中间移动(对撞型)
2. 一个数组, 同时向前移动(前向型)
3. 两个数组(并行型)

1. 对撞型或者相会型

Two sum 类和 Partition 类



Two sum II

<http://www.lintcode.com/zh-cn/problem/two-sum-ii/>

<http://www.jiuzhang.com/solutions/two-sum-ii/>

Triangle Count

<http://www.lintcode.com/en/problem/triangle-count/>

<http://www.jiuzhang.com/solutions/triangle-count/>

(4,3,6,7,8,9)

- 这一类通过对撞型指针优化算2法, 根本上其实要证明就是不用扫描多余状态

```
1  if(A[i] + A[j] > sum)
2      j--;
3      do something
4  else if(A[i] + [j] < sum)
5      i++;
6      do something
7  else
8      do something
9      i++ or j--
```

灌水 类型题目



Trapping Rain Water

<http://www.lintcode.com/en/problem/trapping-rain-water/>

<http://www.jiuzhang.com/solutions/trapping-rain-water/>

(3, 0, 1, 4, 0, 1, 2)

Container With Most Water

<http://www.lintcode.com/en/problem/container-with-most-water/>

<http://www.jiuzhang.com/solutions/container-with-most-water/>

[2,1,4,6,2,3]

Two Sum类题目总结思路

- Two sum

灌水

```

1  if(A[i] + A[j] > sum)
2      j--;
3      do something
4  else if(A[i] + A[j] < sum)
5      i++;
6      do something
7  else
8      do something
9      i++ or j--
    
```

```

1  if (A[i] > A[j])
2      j--;
3  else if (A[i] < A[j])
4      i++;
5  else
6      i++; or j --;
    
```

- 这一类通过**对撞型**指针优化算法, 根本上其实要证明就是**不用扫描多余状态**

```

1  if(考虑A[i]和A[j]满足某个条件)
2      j--; // 不用考虑[i+1, j-1] 和 j 组成的pair
3      do something
4  else if(考虑 A[i]和A[j]不满足某个条件)
5      i++; // 不用考虑 i 和 [i+1, j-1] 组成的pair
6      do something
7  else
8      do something
9      i++ or j--
    
```

Partition 类

Quick select

<http://www.lintcode.com/en/problem/kth-largest-element/>
<http://www.jiuzhang.com/solutions/kth-largest-element/>

- Partition 模板
- 问题？
- [5,5,5,3,5,5,5]

```
public int partition(int[] nums, int l, int r) {  
    // 初始化左右指针和pivot  
    int left = l, right = r;  
    int pivot = nums[left];  
  
    // 进行partition  
    while (left < right) {  
        while (left < right && nums[right] >= pivot) {  
            right--;  
        }  
        nums[left] = nums[right];  
        while (left < right && nums[left] <= pivot) {  
            left++;  
        }  
        nums[right] = nums[left];  
    }  
  
    // 返还pivot点到数组里面  
    nums[left] = pivot;  
    return left;  
}
```

Nuts & Bolts Problem

<http://www.lintcode.com/en/problem/nuts-bolts-problem/>

<http://www.jiuzhang.com/solutions/nuts-bolts-problem/>

2 Sum 类 (通过判断条件优化算法)

3 Sum Closest

4 Sum

3 Sum

k sum

Two sum II

Triangle Count

Trapping Rain Water

Container With Most Water

Partition 类

Partition-array

Sort Colors

Partition Array by Odd and Even

Sort Letters by Case

Valid Palindrome

quick sort/ quick select/ nuts bolts problem/wiggle sort II

Break

休息5分钟

2. 前向型或者追击型

窗口类 和 快慢类

窗口类



Minimum Size Subarray Sum

<http://www.lintcode.com/en/problem/minimum-size-subarray-sum/>

<http://www.jiuzhang.com/solutions/minimum-size-subarray-sum/>

通过两层for循环改进算法

-----不同于sliding window

```
for (i = 0; i < n; i++)  
    while(j < n){  
        if(满足条件)  
            j++;  
            更新j状态  
        else(不满足条件)  
            break;  
    }  
    更新i状态  
}
```

与sliding windows 区别 ?

Longest Substring Without Repeating Characters

<http://www.lintcode.com/en/problem/longest-substring-without-repeating-characters/>

<http://www.jiuzhang.com/solutions/longest-substring-without-repeating-characters/>

1. 前向型指针
2. Hash或者set记录上次访问

Minimum Window Substring

<http://lintcode.com/en/problem/minimum-window-substring/>

<http://www.jiuzhang.com/solutions/minimum-window-substring/>

[ABCDZDEF, ACD]

Longest Substring with At Most K (two) Distinct Characters

<http://www.lintcode.com/en/problem/longest-substring-with-at-most-k-distinct-characters/>

<http://www.jiuzhang.com/solutions/longest-substring-with-at-most-k-distinct-characters/>

- 优化类型：
 - 优化思想通过两层for循环而来
 - 外层指针依然是依次遍历
 - 内层指针证明是否需要回退

通过两层for循环改进算法

```
for (i = 0; i < n; i++)  
    while(j < n){  
        if(满足条件)  
            j++;  
            更新j状态  
        else(不满足条件)  
            break;  
    }  
    更新i状态  
}
```


- 窗口类
 - Remove Nth Node From End of List
 - minimum-size-subarray-sum
 - Minimum Window Substring
 - Longest Substring with At Most K Distinct Characters
 - Longest Substring Without Repeating Characters
- 快慢类
 - Find the Middle of Linked List
 - Linked List Cycle I, II

两个数组两个指针

两个数组各找一个元素， 使得和等于target

1. 找一种
2. 找全部种类

The Smallest Difference

<http://www.lintcode.com/en/problem/the-smallest-difference/>

<http://www.jiuzhang.com/solutions/the-smallest-difference/>

其他的题目

<http://www.lintcode.com/en/problem/merge-two-sorted-lists/>

- Triangle Count
 - Two Sum的变种, 灵活运用
- Nuts & Bolts Problem
 - 怎么样想不到快速排序还能这样考。
- Minimum Size Subarray Sum
 - 前向指针题目的经典入门题

- 两个指针
 - 对撞型 (2 sum 类 和 partition 类)
 - 前向型 (窗口类, 快慢类)
 - 两个数组, 两个指针 (并行)
- 模板
 - 2 Sum类模板
 - Partition 类模板
 - 窗口类模板



Thank You

