

- 聊聊面试当中的Follow Up问题
- 做题的常见误区
- 九章强化算法班正确打开方式
- 聊聊如何在更好的准备面试
- 后续课程安排
- 每日一鸡

# 聊聊面试当中的FollowUp问题

以两个指针中“前向型”指针为例子  
讲解一个模板可以突破多个题目

# 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

```
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  
}
```

# 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

# 怎么解决不会做FollowUp问题

定期整理自己做过的题目

对自己提三个问题:

属于哪一类？

同类的题目有什么相似之处？

他们思考的思路是怎么样的？

# 课前预习

上课之前浏览一遍当前课需要讲的内容。

最好是自己思考一下每道题的解法，如果时间不够，可以浏览一下每个题目的题意。这个非常有助于上课理解。

# 课中笔记

笔记本+PPT

- 思维思考方式
- key关键点
- 一系列题目相应的总结

# 如何在更好的准备面试？

以第k大为例子讲解怎么样总结题型？  
题目做完了后，要去“诈尸”

# 求第K小元素FollowUp

矩阵或者多个数组

方向数组

## 寻找颜五渣

<http://www.lintcode.com/en/problem/kth-smallest-number-in-sorted-matrix/>

<http://www.jiuzhang.com/solutions/kth-smallest-number-in-sorted-matrix/>

是否需要遍历全部的元素呢？

总结:见到集合求Min/Max  
就要想到堆



# Kth Largest in N Arrays

马甲变换一

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



总结：见到数组先排序

# Kth Largest in N Arrays

- 用什么数据结构？
  - Answer:堆
- 方法：
  - 把N个数组先排序
  - 排序过后把每个数组最大的数字放入堆里面
  - 然后堆里面只维护前k个元素
  - 堆pop k次得到答案。
  - 时间复杂度 $N * \text{Len} * \text{Log}(\text{len}) + K * \text{log}N$  (len 是平均每个数组的长度)

# 寻找最佳情侣配对

马甲变换二

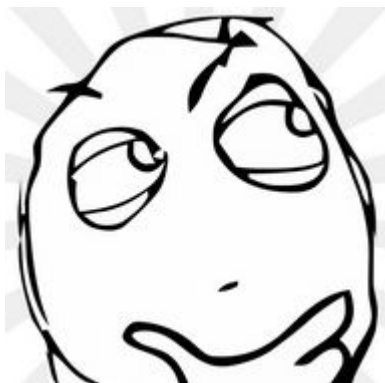
<http://www.lintcode.com/en/problem/kth-smallest-sum-in-two-sorted-arrays/>

<http://www.jiuzhang.com/solutions/kth-smallest-sum-in-two-sorted-arrays/>

要是给两个无序数组呢？

# Kth smallest product in two positive element arrays

马甲变换三



## 小结三道题

- 三道题相似点：
  - 求矩阵/数组的第k大
- 可以总结的规律
- 规律1
  - 见到需要维护一个集合的最小值/最大值的时候要想到用堆
- 规律2
  - 第k小的元素, Heap用来维护当前**候选集合**。
- 规律3
  - 见到数组要想到先排序

# 总结

## 怎么解决不会做follow up问题


1. 定期整理自己做过的题目, Note

2. 学会反思:

它属于哪一类? (归类)

和它同类的题目有什么相似之处? (集体特征)

这些题目思考的思路是怎么样的? (这个题属于哪一类)



分类算法?  
----套路

- 1.透析热门IT公司中的FollowUp面试题
- 2.数据结构(上)
- 3.数据结构(下)
- 4.两个指针 + 按值二分查找
- 5.动态规划(上)
- 6.动态规划(下)
- 7.如何解决困难的follow up 问题



## 2. 数据结构(上)

---

- 并查集
  - 并查集的基本原理
  - 并查集的相关运用
  - 并查集的拓展(带路径压缩)
  - 并查集的运用
- Trie 树
  - Trie 的基本结构
  - Trie 树的相关运用
  - Trie 和 DFS 结合考察
    - Boggle Game
    - Words Square
- 增加线段树小视频

## 3. 数据结构(下)

---

- 堆的深入理解和运用
  - 堆重要拓展:
  - Median 问题拓展
  - Sliding Windows问题总结
- 栈
  - 栈在表达式上面的运用
  - 单调栈的使用方法
- 双端队列Deque
  - Sliding Windows问题总结

## 4.二分法+扫描线

---

- 扫描线
  - 扫描线入门
  - 扫描线和堆结合拓展
- 二分法的运用
  - 普通二分拓展
    - Find Peak Element
  - 按值二分详细解析
    - Subarray sum ii
    - Wood Cut
    - Copy Books
    - Sqrt x II
    - Sqrt X
    - Maximum Average Subarray

## 5. 动态规划(上)

---

- 匹配类动态规划
  - Longest common string
  - Edit distance/ K Edit distance
  - Interleaving string
- 动态规划的空间优化
  - Robber house I/II

## 6. 动态规划(下)

---

- 记忆化搜索
  - 经典滑雪问题
  - 区间动态规划
  - 博弈类动态规划
- 背包类动态规划
  - BackPack I/II
  - K sum
  - Minimum Adjustment Cost

## 7. 如何解决困难的follow up 问题

---

- Iterator
  - Flatten Nested List Iterator
  - Zigzag Iterator
  - Binary Search Tree Iterator
  - Nested List Weight Sum / Nested List Weight Sum II
  - Flatten 2d Array Iterator + Delete
- Subarray Sum
  - Subarray Sum
  - Submatrix Sum
  - Subarray Sum Closest
  - Subarray Sum II
  - Build Post Office
- Wiggle Sort
  - Wiggle Sort I
  - Wiggle Sort II

- 题目难度
  - Medium 50% + Hard 50%
- 目标公司
  - FLAG + USPD
- 学习新的解题思路和较难的算法
  - Trie, 并查集, 单调栈, 动态规划优化
- 题目思路总结, 举一反三
  - 解决follow up思路和构思过程

# Debug 的基本步骤

<http://www.jiuzhang.com/qa/3815/>



