破解算法面试
“Hack” the Algorithm Interview

课程版本 v5.1　　讲师 令狐冲

一共 9 节课
每节 2 小时
中间休息 5 分钟
今天是第 1 节

扫描二维码关注微信小程序/公众号
获取第一手求职资料

版权声明

九章的所有课程均受法律保护，不允许录像与传播录像

一经发现，将被追究法律责任和赔偿经济损失

讲师&助教介绍

讲师：令狐冲
算法竞赛国家队，多年算法教学经验
曾在2家北美顶尖IT企业就职，并担任面试官
国内TOP 1名校毕业
国内外顶级Offer 10+个
刷题数超过3000题

助教：
均获得过算法竞赛金奖
刷题数均超过1000题

Python 3
人工智能时代的语言

九章算法的三门算法主课程全面采用 Python 3 作为主要讲解语言

《Python 入门与算法基础班》- %50 Easy + 50% Medium

《九章算法班》- 80% Medium + 20% Hard

《九章算法强化班》- 50% Medium + 50% Hard

G家常见面试题

开场热身：最长回文子串
Longest Palindromic Substring

G

http://www.lintcode.com/problem/longest-palindromic-substring/

我现在是你的面试官，你可以问我任何问题

打开 www.collabedit.com 开始敲代码

写完的同学可以通过 Zoom 的 QA 功能分享你的代码链接给我

我会挑选一些同学的代码进行点评

写完再解释还是一边写一边解释？

该如何与面试官进行“有效”沟通？

常见错误:我知道有个算法叫 Manacher's Algorithm

该算法可以在 O(n) 的时间内求得最长回文子串，n=字符串长度

这是该问题的最优算法，却绝对不是面试官想让实现的算法

为什么？

因为他自己也不会

面试中常见误区

以为一定要最优解才能过

以为算法想出来了就能过

以为代码写出来了就能过

大家来找茬 I

九章算法

```python
class Solution:
    """
    @param s: input string
    @return: the longest palindromic substring
    """
    def longestPalindrome(self, s):
        start, longest = 0, 0
        for i in range(len(s)):
            for j in range(i, len(s)):
                if j - i + 1 > longest and self.is_palindrome(s, i, j):
                    longest = j - i + 1
                    start = i

        return s[start:start + longest]

    def is_palindrome(self, s, i, j):
        while i < j and s[i] == s[j]:
            i += 1
            j -= 1
        return i >= j
```

大家来找茬 II


```python
class Solution:
    """
    @param s: input string
    @return: the longest palindromic substring
    """
    def longestPalindrome(self, s):
        if not s:
            return ""

        for length in range(len(s), 0, -1):
            for i in range(len(s) - length + 1):
                l, r = i, i + length - 1
                while l < r and s[l] == s[r]:
                    l += 1
                    r -= 1
                if l >= r:
                    return s[i:i + length]
        return ""
```

大家来找茬 III

```python
class Solution:
    """
    @param s: input string
    @return: the longest palindromic substring
    """
    def longestPalindrome(self, s):
        if not s:
            return ""

        start, longest = 0, 0
        for middle in range(len(s)):
            # odd
            left, right = middle, middle
            while left >= 0 and right < len(s) and s[left] == s[right]:
                left -= 1
                right += 1
            if longest < right - left - 1:
                longest = right - left - 1
                start = left + 1

            # even
            left, right = middle, middle + 1
            while left >= 0 and right < len(s) and s[left] == s[right]:
                left -= 1
                right += 1
            if longest < right - left - 1:
                longest = right - left - 1
                start = left + 1

        return s[start:start + longest]
```

基于中心点枚举法 Enumeration 的最佳实践


```python
class Solution:
    """
    @param s: input string
    @return: the longest palindromic substring
    """
    def longestPalindrome(self, s):
        if not s:
            return ""

        self.start, self.longest = 0, 0
        for middle in range(len(s)):
            self.find_longest_palindrome_from(s, middle, middle)
            self.find_longest_palindrome_from(s, middle, middle + 1)

        return s[self.start:self.start + self.longest]


    def find_longest_palindrome_from(self, s, left, right):
        while left >= 0 and right < len(s) and s[left] == s[right]:
            left -= 1
            right += 1
        if self.longest < right - left - 1:
            self.longest = right - left - 1
            self.start = left + 1
```

Follow up: 不能枚举中心点

基于第一种方法，如何进行优化？

基于动态规划 Dynamic Programming 的最佳实践


```python
class Solution:
    """
    @param s: input string
    @return: the longest palindromic substring
    """
    def longestPalindrome(self, s):
        if not s:
            return ""

        n = len(s)
        is_palindrome = [[False] * n for _ in range(n)]
        for i in range(n):
            is_palindrome[i][i] = True
        for i in range(1, n):
            is_palindrome[i][i - 1] = True

        start, longest = 0, 1
        for length in range(2, n + 1):
            for i in range(n - length + 1):
                j = i + length - 1
                is_palindrome[i][j] = is_palindrome[i + 1][j - 1] and s[i] == s[j]
                if is_palindrome[i][j] and length > longest:
                    longest = length
                    start = i

        return s[start:start + longest]
```

从最长回文子串中我们可以得到的面试考点

- 面试不一定会要求你用最优复杂度的算法来解决问题
  - 因此单纯只刷LC之类的OJ，容易让你产生一定要用最优解来解决这样的误区
- 代码真的不是写出来就可以过
  - 代码质量（Coding Quality）很重要
  - 好的代码质量包括：
    - Bug Free
    - 好的代码风格（Coding Style），包括变量名命名规范有意义，合理的使用空格，善用空行
    - 容易让人读懂的逻辑。要把复杂的事情用简单的方式，别把简单的事情写复杂了。
    - 没有冗余代码
    - 有边界检测和异常处理

独孤九剑 —— 总决式

最容易出卖你的，就是你的Coding Style
工程师的代码长什么样比脸长什么样重要

Longest Palindromic Substring 的全部算法及时间复杂度

Manacher's Algorithm - O(n)

后缀数组 Suffix Array - O(nlogn)

动态规划 Dynamic Programming - O(n^2)

枚举法 Enumeration - O(n^2)

参考代码:

http://www.jiuzhang.com/solution/longest-palindromic-substring/

今后所有课上讲的题目的参考答案都可以在这里查询到

从 Longest Palindromic Substring 看面试的评分体系

- **Strong Hire**
- 使用 O(n) 或者 O(nlogn) 的算法实现出来 (Manacher's Algorithm or Suffix Array)，并且代码质量合格，无 Bug 或者 有很小的bug但是能自己发现并解决，无需太多提示
- **Hire**
- 能够分别使用枚举法和动态规划实现时间复杂度 O(n^2) 的算法。并且代码质量优秀，无Bug，无重复代码，无需面试官给提示
- **Weak Hire**
- 只使用了其中一种 O(n^2) 的算法实现出来，代码质量还不错，可以有一些小 Bug，面试官可以给一些小提示
- **No Hire**
- 只能想出一种 O(n^2) 的算法，但是 Bug 太多，或者需要很多提示
- **Strong No Hire**
- 连一种 O(n^2) 的算法都想不到

面试评分和 Offer 的关系

有 >= 1 个 Strong No Hire => No offer

有 >= 2 个 No hire => No offer

有 1 个 No Hire + 1 个 Weak Hire => No Offer

有 1 个 No Hire，其他都是 Hire => Offer or 加面（取决于公司招人多不多，门槛高不高）

有 1 个 Weak Hire => Offer or 加面

特殊情况：

一个 Strong Hire + 一个 Strong No Hire => 开个会一起讨论一下，通常结果是加面或者No Offer。

Implement strStr

http://www.lintcode.com/problem/strstr/

在一个字符串中查询另外一个字符串第一次出现的位置

常见错误:
我知道一个算法叫做KMP

A同学: 论坛上有人说考到了KMP呢！你骗人！

真问我比O(n^2)更好的算法怎么办？

这个概率只有1%

可以学习一个比KMP算法更简单的算法: Rabin-Karp

http://www.jiuzhang.com/video/rabin-karp

(后续学员可见)

休息五分钟

课件获取：关注下方微信公众号，回复“算法5”

扫描二维码关注微信小程序/公众号
获取第一手求职资料

上课须知

- **课程错过不补课，也不提供任何视频**
  - 你才会把在两个小时内集中精力，全神贯注
  - 你才会把学习放在第一位，而不是先 LoL 一把，先逛个街，先和朋友吃个饭
  - 你才会获得最佳的课程体验
  - 良苦用心希望同学们理解
- **不允许建私群（包括QQ群，微信群）**
  - 在QQ群中拉人私下组群的将被踢群并不再提供QQ答疑服务
- LintCode 需要单独先注册一个账户，不要使用九章的账号密码去登陆
- LintCode 阶梯训练必须先完成上一节课的作业，才能做下一节课的作业
- 课程各类服务的有效期为一年
  - LintCode阶梯训练访问权限
  - QQ群答疑
  - QA答疑
  - 课件
  - 知识点小视频

新学员问题

- 新学员必读常见问题解答
  - http://www.jiuzhang.com/qa/3/
- 第一节课错过了怎么办？
  - 报名下一期的《九章算法班》第一节课免费试听即可
- 学员QQ群是什么？怎么加？
  - 请登录官网在我的课程中查看QQ群号
- 九章的账户绑定到LintCode之后可以解除绑定么？
  - 不可以
  - 因此不要把你的九章账户给别人使用
    - 一些老学员的 LintCode 账号绑定了其他人的九章账户是因为你以前把账号共享给了其他人
    - 你可以申请新的 LintCode 账户和你现在的账户进行绑定

到底如何”破解”算法？

上面我们讲了面试的软技巧，学会了如何与面试官沟通，如何通过短时间的训练，让你的代码看上去不像一个初学者

下面我们来讲，你到底怎样才不至于在面试的时候什么都写不出来

面试算法 vs 算法

如果你还在看算法导论？赶紧扔掉

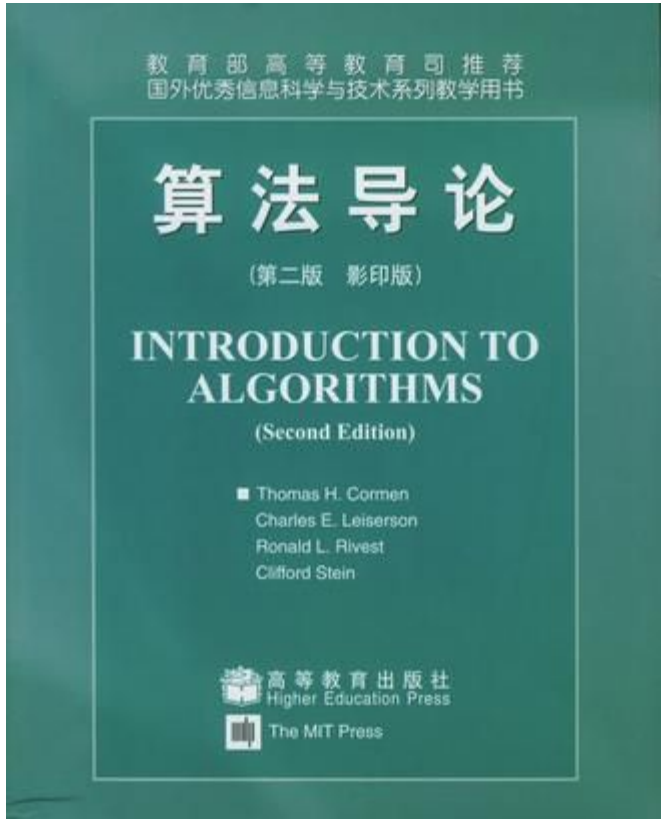
——我宁可你看的是《Cracking The Coding Interview》

也请不要去看普林斯顿的算法公开课

——很多内容面试依然不考，或考得很少

为什么？

——面试算法 != 算法



算法面试最“虚”的部分

不知道的算法那么多

你根本不知道可能考到什么样的问题

完全没有“底”

于是你只能去看面经押题，去LC“买题”

如果让你给算法面试“划考点”

请列举你觉得会考的知识点(算法与数据结构)

到**目前为止**，下面哪些算法和数据结构，**不在**面试考察范围内？

九章算法

最短路算法
Dijkstra / Floyd / SPFA

拓扑排序算法
Topological Sorting

Morris 算法
O(1)额外空间前序遍历

贪心法
Greedy

Manacher 算法
求最长回文子串

KMP算法
strstr / indexOf

最小生成树算法
Minimum Spanning Tree

模拟法
Simulation

外排序 / 多路归并算法
External Sorting

网络流算法
Network Flow

希尔排序
Shell Sort

动态规划
Dynamic Programming

线段树
Segment Tree

平衡排序二叉树
如 Red-black Tree

字典树
Trie

并查集
Union Find

树状数组
Binary Index Tree

堆
Heap

KD树
KD-Tree

B树/B+树
B-Tree / B+ Tree

越红考得越多，灰色不考或者出现概率低于千分之一

最短路算法
Dijkstra / Floyd / SPFA

拓扑排序算法
Topological Sorting

Morris 算法
O(1)额外空间前序遍历

贪心法
Greedy

Manacher 算法
求最长回文子串

KMP算法
strstr / indexOf

最小生成树算法
Minimum Spanning Tree

模拟法
Simulation

外排序 / 多路归并算法
External Sorting

网络流算法
Network Flow

希尔排序
Shell Sort

动态规划
Dynamic Programming

线段树
Segment Tree

平衡排序二叉树
如 Red-black Tree

字典树
Trie

并查集
Union Find

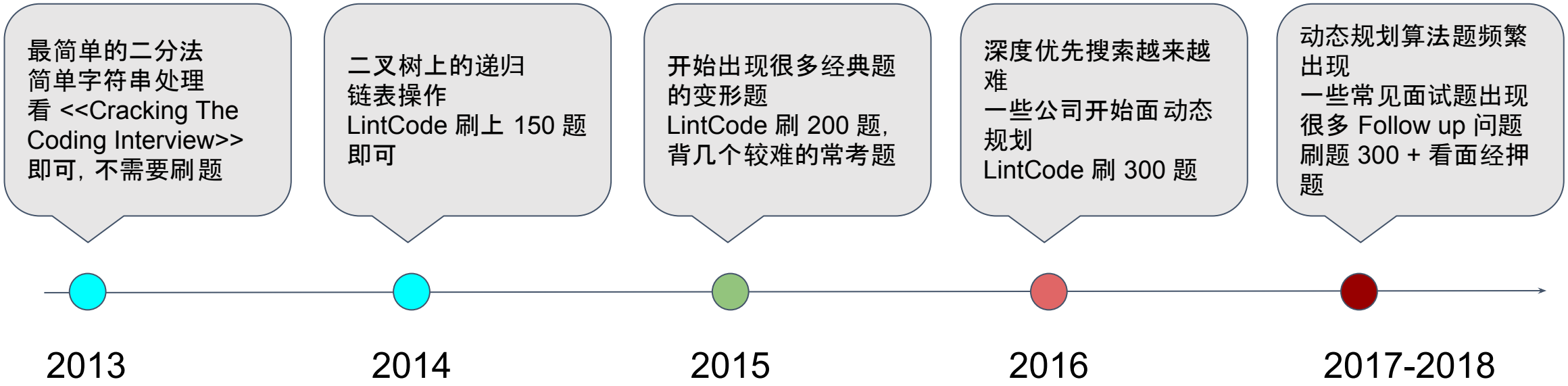
树状数组
Binary Index Tree

堆
Heap

KD树
KD-Tree

B树/B+树
B-Tree / B+ Tree

2013-2017 的面试难度变化



名词中英文对照：

动态规划 - Dynamic Programming
链表 - Linked List
递归 - Recursion
二叉树 - Binary Tree
二分法 - Binary Search
深度优先搜索 -Depth First Search (DFS)

面试常见知识点的考察频率，学习难度

九章算法

算法/数据结构	大公司考察频率	其他公司考察频率	难度	熟练掌握所需做题数	性价比	包含在哪些九章课程中，及学习建议
字符串，模拟法	高	高	低	20~50	中	可自学，练习为主，反复做同一个题，不断简化
二分法	高	高	中	10~20	高	九章算法班
二叉树，链表	高	高	低	30~50	高	九章算法班
递归，深度优先搜索	高	高	高	20~40	中	九章算法班，九章算法强化班
宽度优先搜索	高	高	中	5~10	超高	九章算法班
堆（优先队列）	低	低	中	5~10	中	九章算法班
哈希表	高	高	中	10~30	高	九章算法班
树状数组	低	无	中	2~3	中	九章算法班
两根指针	高	高	中	10~20	高	九章算法班
动态规划	中	低	高	40~60	低	九章算法强化班（部分），动态规划专题班（全部）
字典树	中	低	低	2~5	高	九章算法强化班
并查集	低	无	低	2~5	高	九章算法强化班

对这门课的错误期望

老师把300题一题一题给我讲一遍，带着我都写一遍

—— 臣妾做不到，9节课18个课时，每节课只能”精讲2个题+粗讲7-8个题“。如有这个需求 Youtube 上一大堆的视频等着你

把 Facebook, Google, Amazon, Microsoft 等公司的所有明天就会面的题目给我讲一遍，让我面试能够押到题

—— 押题你可以自己来，网上随时能搜到很多面经，LC高级账号买一个也能解决你的需求

我上完课一定得拿到 Offer，否则这个课就是辣鸡

—— 我能带你学会一个知识，但无法保证你拿到 Offer。面试是 5 分运气，5 分实力。带着尽人事知天命的心态去准备面试，才能更从容的应对。

老师会把那些我平时练习的时候不会做的很难的题讲给我听

—— 你只有 1% 的几率碰到这样的题，性价比不高的东西我不会放在课上来讲。课上讲述的是那些最高频，性价比最高的**知识点**。这些高频知识点的题目不一定很难。

老师会把所有要考的知识点都讲一遍

—— 因为课时限制，这门课只涵盖 70% 的最高频的知识点。

——《九章算法班》+《九章算法强化班》会覆盖 99% 的面试中的算法与数据结构知识点

—— 有 1% 的知识点虽然有的公司考过，但是很低频。

《九章算法班》能带给你什么？

节约时间

你自己需要**三个月**才能准备下来的东西

我用**一个月**带你准备好

其他算法视频 vs 九章算法直播课

九章算法

	其他算法视频	九章算法直播课
形式	录制，滋长**惰性**，学习的时候无人可以问问题。虽然可以反复观看，但是不懂的东西直接问人是更节约时间的。反复观看只是浪费时间，不懂的还是不懂。	直播，定时定量学习，没有再来一次的机会会更加珍惜和集中精力。课程配备**直播助教实时答疑**
氛围	一个人在战斗，很难坚持下去	你不是一个人在战斗，学员QQ群里一起学习，学习积极性更高
课后	看不懂最多只能反复看视频，课后遇到新问题无人可以帮忙解决	课上没有掌握的知识，平时学习遇到的问题，都可以在QQ群，问答板块问老师，问助教
内容	陈题，没有面试官角度的分析，没有面试技巧编程技巧的讲解，没有题目在面试中评价标准的分析，通常是**对单个题如何解决**的讲解，通常只讲一个解法，知识没有连贯性，跳跃极大	**永远是最新内容**，从面试官角度分析，讲算法的同时讲解面试技巧和编程技巧，通常是由知识点带动题目讲解，学会**如何解决一类问题**而不是一个问题。知识点连贯性强，学习流程更加科学化
师资	作者一般只刷过 200-300 题	3000+ 的刷题经验，算法竞赛国家集训队员，ACM竞赛金牌
题库	不配套题库，或需要对题库进行额外付费	课上所涉及的题目，包括相关练习题约200道题，**无需对题库重复付费**，一年之内可以随便刷
评测	无，你根本不知道自己学得好不好	平时作业 + 期末考试，检验自己学习的水平，更有底气去面试 优秀学员还可以获得 FLAG 等企业的**内推机会**
价格	**免费的东西是最贵的**，因为你浪费了时间，不付钱你也不会珍惜	便宜的价格，不便宜的质量

我们卖的不是视频，而是**服务**

即便你搞来了九章往期的盗版视频(或者你正在观看盗版视频)

你也远远达不到九章直播课的学习效果

你可以在这个班中学会

哪些知识点是高频的，多花一些时间，哪些不是，少花一些时间，哪些根本不考，别花时间

如何判断一个题目该用什么样的算法和数据结构去解

该怎样才能写出一个不出、少出 Bug 的代码

哪些代码模板是我面试之前必须充分练习的

在老师 3000+ 刷题经验的讲解中，更深入的理解算法知识点的精髓

刷题刷到什么程度去面试才够？

你永远没有觉得自己准备好的那一天！

怎样 Debug 才能高效？

“抓虫神功”有三重境界，你练到了哪一层：

第一重：断点调试

第二重：分段输出

第三重：用肉眼看

当你的代码有 Bug 的时候，不要直接贴给老师、助教或者其他同学帮你 Debug，如果别人给你指出错误在哪儿你自己什么都学不到！

拿到Offer的四大法宝

除了押题，你还能干嘛？

1. 别做难题

不要花时间攻关难题，多做 LintCode 上 Medium 难度的题

不要把时间浪费在那些基本不会考你又很心虚的内容

比如KMP，红黑树，AVL，ACM竞赛题

2. 是面试不是考试

和面试官愉快交流，一起合作解决面试问题

证明自己牛逼，但别去证明面试官傻逼

3. 理解而不是单纯的背诵

在课程中主要学习的是思维方式和分析技巧

而不是某个题的解法

4. 反复练习

把时间花在如何做到 BUG FREE 和提高编程速度上

代码超过50行的题，可以反复写几遍

天下武功，唯快不破！

大纲和上课时间

http://www.jiuzhang.com/course/1/

注意时区，特别是周几上课，**错过不补**

v5.0 版本的课程更新

相比于 v4.0

基础知识先修

重点知识直播

补充知识拓展

形式：文字+小视频

先修内容通常需要 1 小时，务必在课前完成

否则难以跟上课程进度

补充知识有空的时候看，内容也很重要

不能不看

课程形式

1. 增加配套的九章微课堂随课教程。
2. 从原来的纯上课直播，变为了三个部分：基础知识先修，重点知识直播，相关知识补充。先修知识和补充知识被放在了配套的随课教程中。
3. 课前学生可以通过随课教程中的视频和文字自学基础知识，保证课程能够跟上节奏。
4. 一些课上较难的重点内容也会被以文字或视频的形式放在教程中供学生课后复习。
5. 一些在课上讲不完的重要内容和相关知识，也会放在随课教程中供学生课后自学。

期末考试

考试中排名靠前的 10 名学员将获得一次内推机会

可以内推 FLAG 等企业

1. 所有章节的题目，都根据最新的面经进行了一次换血。部分章节题目改动较大。
2. 对于二分法做了一个扩充，同时随着目前面试难度提高，有些O(N)算法已经不能获得Offer，针对O(LogN)级别的算法也将做一个讲解
3. 增加了一节深度优先搜索的课程，将有三节课来讲解深度优先搜索，分为基于树的DFS，基于图的DFS，基于组合的DFS，基于排列的DFS。
4. 新增一个数据结构 - 树状数组(Binary Indexed Tree)的讲解。树状数组是近期Facebook、Google面试中出现过的数据结构，热门新增，与时俱进
5. 将链表的相关问题挪到了课后补充学习资料中，增加了区间和矩阵的相关内容。
6. 动态规划部分(原来包含一节动态规划的入门课)移入九章算法强化班和动态规划专题班

依旧保持的课程特色

1. 课程内容永远最实时，最结合当前的面试形式。老师会在课上讲解很多面试的技巧，不仅仅是面试题。
2. 课堂上在线直播助教实时答疑，确保你能够及时扫清障碍。
3. 课后有任何问题都能够在QQ群或者官网问答板块找到老师和助教来为你解答。
4. 通过老师 3000+ 的刷题经验，带领大家更深入的理解各种面试所考察的知识点。
5. 原本自己学习需要 3 个月时间才能准备下来的内容，跟班学习可以在 1 个月的时间内学下来。
6. 配套的 LintCode 练习题，边学边练，保证学习效果。
7. 直播学习，比视频教学更加高效，学习效果更好。
8. 依旧低廉的课程价格，平均每小时不到 10$。

这门课适合谁？

只要你有求职需求，只要你面试会面算法

无论是的主语言是 Java 还是 C++, Python

只要你先节约求职准备时间，提高求职准备效率

你可以获得哪些学员权限？

- LintCode专属阶梯训练题
  - 180 道精选题
  - 50+ 私有练习题
- 九章问答提问有专人回答
  - 助教老师100%回答
- 九章问答课程与内推板块浏览权限
  - 最新最热面试题面经实时分享
  - 让九章老学员帮你内推各大公司
- 九章课程QQ群
  - 与同学们实时交流学习问题
  - 随时 @老师 @助教 答疑解惑
  - 认识更多志同道合的朋友，一起打鸡血
  - 学员线下活动（自行组织）

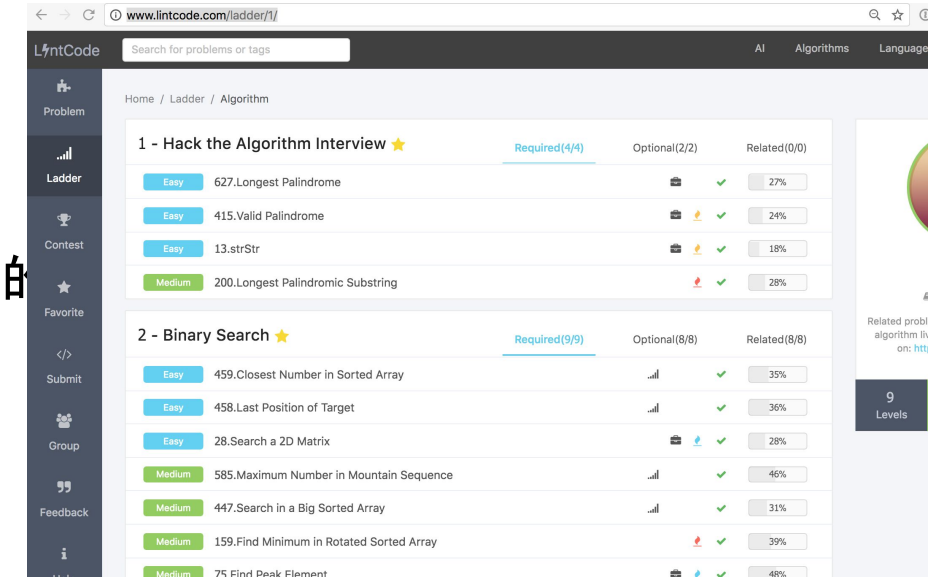
课程配套的 LintCode 阶梯训练

访问地址：

http://www.lintcode.com/ladder/1/

玩法：

解决上一节课的阶梯中的必做题（Required）才能看到下一节课的



Required - 通常是课上讲过的题

Optional - 其他近期高频的该类算法问题，但课上没讲

Related - 其他与该类算法或者某些题目相关的值得一练的题

你需要先注册一个 LintCode 账号，九章账号无法直接登录

有效期 1 年（自开课之日起算）

其他课程 http://www.jiuzhang.com/course/

九章算法

九章算法班

硅谷求职必上，FLAG敲门砖，成为Offer收割机。

8 分钟 后开始直播

系统设计班

怎样设计Facebook？理解Google的三驾马车！

2 周 后开始直播

Android 项目实战

硅谷工程师教你从零开始学习Android 编程！

1 周 后开始直播

Big Data 项目实战

硅谷工程师教你从零开始学习Big Data！

等待下次开课

九章算法强化班

寒冬期，算法面试越来越难，你需要充电！

8 小时，38 分钟 后截止报名

Java入门与基础算法班

不会Java？转专业找CS？算法零基础？

等待下次开课

算法面试高频题班

Facebook, Google, Linkedin, Amazon 高频题短期冲刺班

1 周，6 天 后开始直播

动态规划专题班

全网唯一的 DP 专题课，更加系统的掌握面试中最难的算法！

1 周，1 天 后开始直播

《硅谷算法求职集训营》课程介绍

http://www.jiuzhang.com/camp/15/

学习周期更长，内容更细致，节奏更慢
个性化服务更多，详询课程顾问 →

付款方式？

九章官网登陆账户 →
我的课程 →
找到对应报名信息点击付款链接

支付宝/微信付款 优惠 2-5%

优惠码的获得？

关注微信“九章算法”

点击右下角“课程优惠”按照提示操作

版权声明

九章的所有课程均受法律保护，不允许录像与传播录像

一经发现，将被追究法律责任和赔偿经济损失

课后拓展阅读

必读：

Google Coding Style: https://google.github.io/styleguide/javaguide.html

Rabin Karp: http://www.jiuzhang.com/video/rabin-karp

函数式编程：https://www.zhihu.com/question/28292740

选读：

Manacher's Algorithm:

https://segmentfault.com/a/1190000003914228

https://www.geeksforgeeks.org/manachers-algorithm-linear-time-longest-palindromic-substring-part-1/

Q & A

常见问题 http://www.jiuzhang.com/qa/3/

扫描二维码关注微信小程序/公众号
获取第一手求职资料