手把手带你搞定 10 道大厂算法面试题

覃超



互联网大厂面试的形态

• 头条面经解读

https://www.bilibili.com/video/BV14z411b7sT

leetcode-cn.com 论坛
 leetcode.com
 https://www.lpoint3acres.com/bbs/ (一亩三分地)



基础知识

语言 / 多并发 / 数据库 + 缓存 / 其他
Google -> 好的源: github、掘金、知乎、程序园、csdn

• 以 java 为例

https://github.com/Snailclimb/JavaGuide

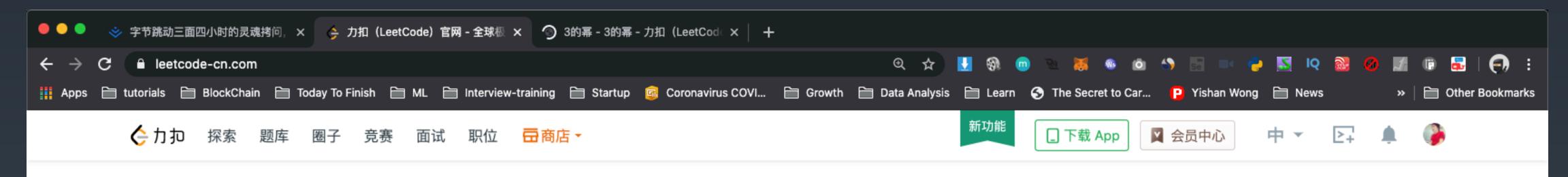
https://github.com/CyC2018/CS-Notes

https://juejin.im/entry/58f1d35744d904006cf14b17



算法"难题"的分析







3天后

参与竞赛: 第192场周赛



5分钟前

白白 在题目 26. 删除排序数组中的重复项 新增了题解 无

解题思路 双指针移动,对比元素解题。 当遇到重复元素时,左指针不动,右指针继续移动。关键在于理清遇到不相等元素时,左指针移动且其对应元素值的问题。(建议图解) 代码



6分钟前

y100k 在题目 面试题 01.01. 判定字符是否唯一 新增了题解 一行代码 关键是 Set 的特性

解题思路 根据题目描述 所有字符是否全都不同 也就是,是否有重复的字符 去重首选 Set 判断一下 new Set(astr)的size 和原来的字符串长度 是否相等就行了 关键是 Set 用法 https://developer.mozilla.org/zh-CN/docs/Mozilla/Add-ons/We



6分钟前

题目 93. 复原IP地址 的题解 回溯算法(画图分析剪枝条件) 有新的评论

回溯算法事实上就是在一个树形问题上做深度优先遍历,因此首先需要把问题转换为树形问题。 这里请大家一定要拿起纸和笔,模拟一下如何通过指定的字符串 s 生成 IP 地址的过程,把树形图 画出来(这一点很重要)。下面这张图我没有画完(如果画完,枝叶太多),请读者尽量不看我 画的这张图,自己动手尝试一下这个问题的树形图应该怎么画。



7分钟前

zhangyi1357 赞了题目 837. 新21点 的题解 还有比这更简单的题解吗?填格子游戏开始

解题思路 老实说,一开始没懂题目意思,后面才知道是求爱丽丝的胜率。 规则是这样: 她可以从牌面为[1,W]的牌中选择任意一张,这张牌是可以无限重复的,也就是说无论她取多少次,每次取到2(假如2在[1,W]范围内)的概率都是1/W; 如果她手上牌的总额小于K,她就会抽牌,大于等于K时,就停止抽牌; 停止抽牌后,她的牌面



第 192 场力扣周赛

06/07 周日上午 10:30 ~ 12:00





毎日1题 6 月热题来袋!

每天 0 点题库首页置顶 坚持刷题,持续优秀,很**回**!

力扣竞赛

参与竞赛,便有机会赢取 奖品



报名参赛



力扣圈子

极客云集、技术讨论、分 享经验的交流圈

打开圈子

使用力扣积分

在力扣商店使用你的力扣 积分



了大学



70 天彻底攻克算法,斩获大厂 Offer

6 月 22 日将永久涨价至 ¥1999 起



题目一

• https://leetcode-cn.com/problems/regular-expression-matching/



特点

- 没思路
- ●思维复杂度貌似非常高
- 一般是 分治 或者 动态规划 (DC ~ DP)

如何思考?



算法

- If-else, switch —> branch
- for, while loop —> Iteration
- 递归 Recursion (Divide & Conquer, Backtrace)
- •搜索 Search: 深度优先搜索 Depth first search, 广度优先搜索 Breadth first search, A*, etc
- · 动态规划 Dynamic Programming
- 二分查找 Binary Search
- 贪心 Greedy
- 数学 Math , 几何 Geometry

注意: 在头脑中回忆上面每种算法的思想和代码模板



难题的分析

- https://leetcode-cn.com/problems/climbing-stairs/
- https://leetcode-cn.com/problems/coin-change
- https://leetcode-cn.com/problems/decode-ways/

· 寻找重复性 —> 分治以及DP的思路和想法



例题集合

- https://leetcode-cn.com/problems/all-possible-full-binary-trees/
- https://leetcode-cn.com/problems/partition-array-for-maximum-sum/
- https://leetcode-cn.com/problems/burst-balloons/
- 寻找重复性 —> 分治的思路和想法



例题集合

- https://leetcode-cn.com/problems/generate-parentheses/
- https://leetcode-cn.com/problems/combinations/ 等



数据结构

- —维:
 - 基础: 数组 array (string), 链表 linked list
 - 高级: 栈 stack, 队列 queue, 双端队列 deque, 集合 set, 映射 map (hash or map), etc
- · 二维:
 - 基础: 树 tree, 图 graph
 - 高级: 二叉搜索树 binary search tree (red-black tree, AVL), 堆 heap, 并查集 disjoint set, 字典树 Trie, etc
- 特殊:
 - 位运算 Bitwise, 布隆过滤器 BloomFilter
 - LRU Cache



时间复杂度

Data Structure	Time Complexity								Space Complexity
	Average				Worst				Worst
	Access	Search	Insertion	Deletion	Access	Search	Insertion	Deletion	
<u>Array</u>	Θ(1)	<mark>Θ(n)</mark>	<mark>Θ(n)</mark>	Θ(n)	0(1)	0(n)	0(n)	0(n)	<mark>0(n)</mark>
<u>Stack</u>	Θ(n)	<mark>Θ(n)</mark>	Θ(1)	Θ(1)	0(n)	0(n)	0(1)	0(1)	<mark>0(n)</mark>
<u>Queue</u>	<mark>Θ(n)</mark>	<mark>Θ(n)</mark>	Θ(1)	Θ(1)	0(n)	0(n)	0(1)	0(1)	<mark>0(n)</mark>
Singly-Linked List	<mark>Θ(n)</mark>	<mark>Θ(n)</mark>	Θ(1)	Θ(1)	0(n)	0(n)	0(1)	0(1)	<mark>0(n)</mark>
Doubly-Linked List	Θ(n)	<mark>Θ(n)</mark>	Θ(1)	Θ(1)	0(n)	0(n)	0(1)	0(1)	<mark>0(n)</mark>
<u>Skip List</u>	$\theta(\log(n))$	$\theta(\log(n))$	$\theta(\log(n))$	θ(log(n))	0(n)	0(n)	0(n)	0(n)	O(n log(n))
<u>Hash Table</u>	N/A	Θ(1)	Θ(1)	Θ(1)	N/A	0(n)	0(n)	0(n)	<mark>0(n)</mark>
Binary Search Tree	θ(log(n))	θ(log(n))	Θ(log(n))	θ(log(n))	0(n)	0(n)	0(n)	0(n)	<mark>0(n)</mark>
Cartesian Tree	N/A	θ(log(n))	Θ(log(n))	θ(log(n))	N/A	0(n)	0(n)	0(n)	<mark>0(n)</mark>
B-Tree	θ(log(n))	θ(log(n))	$\theta(\log(n))$	θ(log(n))	0(log(n))	0(log(n))	0(log(n))	0(log(n))	<mark>0(n)</mark>
Red-Black Tree	θ(log(n))	$\theta(\log(n))$	$\theta(\log(n))$	θ(log(n))	O(log(n))	0(log(n))	0(log(n))	0(log(n))	<mark>0(n)</mark>
Splay Tree	N/A	$\theta(\log(n))$	$\theta(\log(n))$	θ(log(n))	N/A	0(log(n))	0(log(n))	0(log(n))	<mark>0(n)</mark>
AVL Tree	θ(log(n))	θ(log(n))	Θ(log(n))	θ(log(n))	O(log(n))	0(log(n))	0(log(n))	0(log(n))	<mark>0(n)</mark>
KD Tree	θ(log(n))	θ(log(n))	Θ(log(n))	θ(log(n))	0(n)	0(n)	0(n)	0(n)	<mark>0(n)</mark>

算法

- If-else, switch —> branch
- for, while loop —> Iteration
- 递归 Recursion (Divide & Conquer, Backtrace)
- •搜索 Search: 深度优先搜索 Depth first search, 广度优先搜索 Breadth first search, A*, etc
- · 动态规划 Dynamic Programming
- 二分查找 Binary Search
- 贪心 Greedy
- 数学 Math , 几何 Geometry

注意: 在头脑中回忆上面每种算法的思想和代码模板





70 天彻底攻克算法,斩获大厂 Offer

6 月 22 日将永久涨价至 ¥1999 起



算法模板+脑图总结

• 图形化记忆和学习 (脑图) + 费曼学习法

• 动手练习(五毒神掌)

• 班级、微信群、战友、人脉资源



过遍数一>脑图记忆

- 书先是越读越厚,然后越读越薄
- 看山是山,看山不是山,看山還是山
- 内功浑厚: 重剑无锋、大巧不工





70 天彻底攻克算法,斩获大厂 Offer

6 月 22 日将永久涨价至 ¥1999 起



问题环节