链滴

2021 年 5 月滴滴算法岗：三面拿下 offer，面试题分享

作者：julyedu

原文链接：https://ld246.com/article/1622538804046

来源网站：链滴

<h2 id="添加微信-julyedufu77-回复---11---领取最新升级版-名企AI面试100题-电子书----">添加信：julyedufu77，回复 “ 11 ” ，领取最新升级版《名企 AI 面试 100 题》电子书！！ **</h2>
<h2 id="面试题">面试题</h2>
问题 1：岛屿个数（dfs）
问题 2：有向图最短路径（引导我用动态规划解决）
问题 3：问我对 NP 和 P 问题的看法
问题 4：哥德巴赫猜想、四色猜想
问题 5：无序数组第 k 大的元素，要求给出最低时间复杂度的方法以及推导时间复杂度
<h2 id="问题1-岛屿个数-dfs-">问题 1：岛屿个数（dfs）</h2>
题干： 给你一个由 '1'（陆地）和 '0'（水）组成的的二维网格，请你计算格中岛屿的数量。岛屿总是被水包围，并且每座岛屿只能由水平方向和/或竖直方向上相邻的陆地连形成。此外，你可以假设该网格的四条边均被水包围。

样例： grid = [["1","1","1"], ["0","1","0"], ["1","0","0"], ["1","0","1"]]
解题思路-深度优先： 题干要求的是岛屿数量 num, 可以按照行列遍历所元素，第一次遇到 “1” 时 岛屿数量 +1，然后按照深度优先遍历后面连续的 “1” ，并将 “1” 赋值 “0” ；然后继续遍历，再次遇到 “1” 时岛屿数量 +1，然后再将其链接的 “1” 改为 “0” ；重复该程直至遍历所有元素；
参考代码如下：

<h2 id="问题2-有向图最短路径-引导我用动态规划解决-">问题 2：有向图最短路径（引导我用动态划解决）</h2>
每次找到离源点最近的一个点，以该点为中心，更新源点到其他源点的最短路径，贪心的思想。
参考代码：

<h2 id="问题3-问我对NP和P问题的看法">问题 3：问我对 NP 和 P 问题的看法</h2>
解释 1：
最简单的解释：
P：算起来很快的问题
NP：算起来不一定快，但对于任何答案我们都可以快速的验证这个答案对不对
NP-hard：比所有的 NP 问题都难的问题
NP-complete：是 NP hard 的问题， 并且是 NP 问题
解释 2：
通俗的讲，P 就是能在多项式时间内解决的问题，NP 就是能在多项式时间验证答案正确与否的题。所以 P 是否等于 NP 实质上就是在问，如果对于一个问题我能在多项式时间内验证其答案的正确，那么我是否能在多项式时间内解决它。再说说 NP-hardness 和 NP-completenes. 这里涉及一个念，不妨称为问题之间的归约。
可以认为各个问题的难度是不同的，表现形式为，如果我可以把问题 A 中的一个实例转化为问题 中的一个实例，然后通过解决问题 B 间接解决问题 A，那么就认为 B 比 A 更难。通过对归约过程做限制可以得到不同类型的归约。复杂度理论里经常用到的规约叫 polynomial-time Karp' reduction 其要求是转化问题的过程必须是多项式时间内可计算的。到这为止 NP-hardness 和 NP-completenes 就很好理解了。
称问题 L 是 NP-hard，如果任意一个 NP 的问题都可以多项式规约到 L。如果一个 NP-hard 的题 L 本身就是 NP 的，则称 L 是 NP-complete。这个定义可以推广到所有复杂度类。所以 complenes 的直观解释就是，我能解决这个问题就相当于具备了用相同级别的计算资源解决这个复杂度类里所问题的能力。

<h2 id="问题4-科学计数法哥德巴赫猜想-四色猜想">问题 4：科学计数法哥德巴赫猜想、四色猜想<h2>
验证“哥德巴赫猜想”，即：任意一个大于 2 的偶数均可表示成两个素数之和数学领域著名的“德巴赫猜想”：
任何一个大于 2 的偶数总能表示为两个素数之和；将一个偶数用两个素数之和表示的方法，等于一横线上，蓝线和红线的交点数。数值验证：
1938 年，尼尔斯·皮平（Nils Pipping）验证了所有小于 10^5 的偶数。
1964 年，M·L·斯坦恩和 P·R·斯坦恩验证了小于 10^7 的偶数。
1989 年，A·格兰维尔将验证范围扩大到 10^10。
1993 年，Matti K. Sinisalo 验证了 10^11 以内的偶数。
2000 年，Jörg Richstein 验证了 10^14 内的偶数。
截至 2014 年，数学家已经验证了 10^18 以内的偶数，在所有的验证中，没有发现偶数哥德巴猜想的反例。
下面用 Python 进行验证：
思路： 对于偶数 M，遍历 0~M，判断 x ~[0, M]是否是素数，若是素数判断 M-x 是否是素数，若是则找到一个要求的答案：
假设 M=100

<h2 id="问题5-序列无序数组第k大的元素-要求给出最低时间复杂度的方法以及推导时间复杂度">题 5：序列无序数组第 k 大的元素，要求给出最低时间复杂度的方法以及推导时间复杂度</h2>
思路： 快排中 partition
思想： 对于某个索引 j，nums[j] 经过 partition 操作： nums[left] 到 nus[j - 1] 中的所有元素都不大于 nums[j]； nums[j + 1] 到 nums[right] 中的所有元素都不小于 nums[j 即 nums[j]元素的位置是排序好了的，我们判断索引 j 是否满足条件，若满足则直接返回结果，若不足我只需要遍历左序列或者右序列，直至满足要求；无需对整个序列进行排序，所以减少了计算量；
参考代码：

复杂度分析：时间复杂度：O(N)，这里 N 是数组的长度空间复杂度：O(1)，原地排序没有借助外的辅助空间。
<h2 id="添加微信-julyedufu77-回复---11---领取最新升级版-名企AI面试100题-电子书--">添加微：julyedufu77，回复 “ 11 ” ，领取最新升级版《名企 AI 面试 100 题》电子书！！</h2>