

第七届广西大学生程序设计大赛 题解

The 7th Guangxi Collegiate Programming Contest Tutorial

By Colin & His Friends

Hangzhou Dianzi University

2024 年 5 月 19 日

Overview

Difficulty	Easiest											Hardest	
Idea	L	B	E	J	F	D	K	H	M	A	C	I	G
Coding	L	B	E	J	F	A	H	D	M	K	I	C	G
Summary	L	B	E	J	F	D	H	K	A	M	C	I	G
On Site	L	B	E	F	J	D	H	A	K	M	C	I	G

L. Loong

Description

输出第 x 年之后的下一个龙年是哪年。

■ $1 \leq x \leq 9999$

Solution

直接预处理，复杂度 $O(n)$ 。

公式判断， $ans = \lfloor \frac{x+4}{12} \rfloor \times 12 + 8$ ，复杂度 $O(1)$ 。

两种方法均可通过。

B. Black Friday

Description

n 件物品，每件物品有一个原价和促销价。

求折扣符合给定要求的物品的促销价之和。

■ $1 \leq n \leq 10^4$ 。

Solution

循环判断即可。

分数作比较可以转成两侧同乘分母，就可以避免精度的问题。

复杂度 $O(n)$ 。

E. Epsilon Estuary

Description

有 n 只史莱姆，第 i 只血量是 x_i 。

每次史莱姆受到攻击会裂成两个，受到攻击后的血量是 x 的话，会裂成 $\lfloor \frac{x}{2} \rfloor$ 和 $\lceil \frac{x}{2} \rceil$ 。

现在你可以使用法术，每次对全场所有史莱姆造成 y 点攻击。

问清场最少需要使用多少次法术。

■ $1 \leq n \leq 10^5, 1 \leq x_i, y \leq 10^{18}$

Solution

首先注意到只用关注血量最高的史莱姆，因为每次受伤后其他裂出来的血量肯定不超过血量最高的裂出来的。

同理对于血量最高的怪，每次也只需要管他裂出来血量较大的一半。

接下来模拟即可，因为每次血量至少折半，单组复杂度 $O(n + \log \max x_i)$ 。

F. Farewell GXCPC!

Description

有 n 个题，每个题有一个难度 d_i 和一个有趣程度 f_i 。

Colin 要从中选一些组成一场比赛，满足：

1. 所有选中题目难度之和 $\in [L, R]$
2. 所有选中题目有趣程度最大。

现在 Colin 会进行 q 组询问，每次给出一对 L, R ，求此时可能的选中题目有趣程度之和的最大值。

$$\blacksquare 1 \leq n \leq 5000, 1 \leq d_i, f_i \leq 10^5, 1 \leq L_i \leq R_i \leq 10^5, 1 \leq q \leq 10^6$$

Solution

令 $m = \max_{1 \leq i \leq q} R_i$ 。

首先注意到这是一个背包问题，可以用 $O(nm)$ 的经典 01 背包处理出难度和恰好为 i 的最大有趣程度和。

接下来就是在这个 dp 数组上多次询问区间最大值。使用 ST 表/线段树等支持区间 RMQ 查询的数据结构即可。

若使用 ST 表时间复杂度为 $O(nm + m \log m + q)$ 。

若使用线段树，时间复杂度为 $O(nm + q \log m)$ 。

J. Judge System with Random

Description

某场比赛一个题，评测系统会随机返回正确/错误。

一个人如果至少有一个提交的返回结果是正确的，罚时计算方式是，第一次正确的提交时间 + （此前提交次数） $\times 20$ 。

一个人如果至少有一个提交的返回结果是正确的，并且罚时是所有的这些人里最小的，那么他就是冠军。如果有多个最小的这些人会共享冠军。

Colin 和 Eva 分别进行了 n 次提交和 m 次提交，并给出相应的提交时间。

问所有的 2^{n+m} 种可能性中，有多少种 Colin 会成为冠军。

■ $1 \leq n, m \leq 10^5$

Solution

枚举 Colin 的第一个 accepted 的提交 i ，那么如果 Colin 想要成为冠军，可以发现 Eva 一定要有一个前缀必须全部是 rejected。

此时找到允许 accepted 的最靠前的 Eva 的提交 j ，那么就可以向答案累计 $2^{n-i} \times 2^{m-j+1}$ 。

注意到随着 i 变大， j 也会变大。因此可以双指针处理，复杂度 $O(n + m)$ 。

当然对于每个 i 去二分找到对应的 j 也可以通过。

D. Dune

Description

有 n 只沙虫在数轴上向正方向移动，第 i 只初始的位置描述为 $[l_i, r_i]$ ，以每秒 v_i 的速度向正方向移动。

你的位置在 x ，你可以选任意一个非负实数时刻召唤沙虫，你会召唤出这个时刻跨越 x 的所有沙虫中最短的一条。

问你有可能召唤出的最长的沙虫有多长。保证至少有一只沙虫经过你的位置。

■ $1 \leq n \leq 10^5, 0 \leq x \leq 10^9, -10^9 \leq l_i \leq r_i \leq 10^9, 1 \leq v_i \leq 10^9$

Solution 1

求出每个沙虫经过 x 的时间区间 $[t_{l,i}, t_{r,i}]$ 。

对时间进行扫描线，同时维护一个 multiset，每次遇到一个开始时刻就把这个虫子的长度放入 multiset，遇到结束时刻把虫子的长度移出 multiset。

答案就是任意时刻 multiset 中的最小值的最大值。

我们只需要判断每个有插入/删除操作的时刻，以及每次插入时刻 $+\epsilon$ 。

[—————]

[——] * [——]

具体实现可以让每个时刻先执行插入再执行删除，然后插入都做完判断一次，删除都做完再判断一次。复杂度 $O(n \log n)$ 。

Solution 2

求出每个沙虫经过 x 的时间区间 $[t_{l,i}, t_{r,i}]$ 。

二分答案，每次相当于判断一个沙虫的区间是否会被比他小的区间的并覆盖。

具体实现可以用 `set` 维护区间并。同样要注意上述间隙的问题。

复杂度 $O(n \log^2 n)$ 。

H. Hashing Collision

Description

给定一个字符串 S 和一组哈希参数。

定义两个子串发生了哈希冲突，当且仅当这两个串不同，但他们的哈希值相同。

问有多少个子串对发生了哈希冲突。

- $1 \leq |S| \leq 3000, 0 \leq S_i \leq 10^9$

Solution

答案就是哈希值相同的子串对数 - 相同的子串对数。

前者可以直接模拟求出，例如存储所有的哈希值然后排序统计。

后者可以执行 n 次 KMP 求出，也可以借用更高级的后缀自动机等工具。

也可以尝试使用双哈希求出后者，不保证一定正确，有可能会被生日攻击影响（字符集很大，所有子串有可能均不相同）。

复杂度 $O(n^2 \log n)$ 。

A. Ascending Random Number Sequence

Description

给定 n ，考虑随机生成一个上升数列的过程：

先从 $\{1, 2, \dots, n\}$ 中等概率随机选择一个数字 k 作为参数。

再从 $\{1, 2, \dots, n\}$ 中等概率随机选择一个数字 x 作为数列第一项。

接下来循环执行：如果当前数列中最大值已经 $\geq k$ 则退出；否则假设最大值为 y ，从 $\{y+1, y+2, \dots, n\}$ 中等概率随机选出一个数字，作为数列的新的一项。

求数列的期望长度。

■ $1 \leq n \leq 10^7$

Solution

推导的方法很多，下面是一种方法：

假设固定 k ，定义 $f[i]$ 表示从 i 开始到结束的期望长度。

则对于 $i \geq k$, $f[i] = 1$; 对于 $i < k$, $f[i] = 1 + \frac{1}{n-i} \sum_{j=i+1}^n f[j]$; 答案为 $\frac{1}{n} \sum_{i=1}^n f[i]$ 。

考虑从大到小枚举 i ，维护 $f[i]$ 的后缀和 sum ，注意到递推形式为初始化 $sum = n - k + 1$ ，然后枚举 i 从 $k-1$ 到 1 ，执行 $sum = 1 + \frac{n-i+1}{n-i} sum$ 。

注意到这本质上是若干一次函数的复合。因此维护前缀一次函数的复合，每次枚举 k 即可 $O(1)$ 求出对应的结果。

因为数据范围比较大，需要线性求逆元。

K. Kingdom Rush: Vengeance

Description

有 n 个范围伤害塔，第 i 个塔在每秒结束的时候会对区间 $[L_i, R_i]$ 里的所有怪造成 w_i 的伤害。

有 m 个怪，第 i 个怪在第 $0.5s$ 时生成在 $x_i + 0.5$ 处，血量为 h_i ，且以每秒 1 的速度向数轴正方向移动。

求每个怪能在场上存活多久，或者场上的塔不足以干掉它。

■ $1 \leq n, m \leq 10^5, 1 \leq L_i \leq R_i \leq 10^9, 1 \leq w_i \leq 10^4, 0 \leq x_i \leq 10^9, h_i \leq 10^{18}$

Solution 1

根据所有的 L_i, R_i 可以将数轴分成 $2n$ 段，每段内的单位时间伤害是一致的。

可以先定位到 x 所在的段，把这一段模拟走完。

然后二分找到怪物能活到的最远的段，然后最后一段再模拟找到击杀的精确位置。

可以用 vector 维护线段，预处理前缀和，然后二分。

复杂度 $O(n \log n)$ 。

Solution 2

动态开点线段树，支持区间加法查询区间和。

询问可以线段树上二分，复杂度 $O(n \log n)$ ，实现起来细节比较多。

也可以线段树 + 二分，复杂度 $O(n \log^2 n)$ ，常数不是很大也可以通过。

M. Matching on a Tree

Description

给定一棵有根树，保证除了根以外，任意一层都有偶数个点。

两个点可以匹配当且仅当他们深度相同。两个点匹配的代价是他们在树上的距离。

求最小代价最大匹配。

■ $1 \leq n \leq 10^6$

Solution 1

对于同一深度，最优策略是子树内能匹配就匹配，剩余至多一个点继续向上。

对于每条边，需要跨过它匹配的点数至少是子树内同层点数的奇偶性，上述贪心恰好达到这个下界。

考虑同时处理所有深度。令 $f_u[d] \in \{0, 1\}$ 表示 u 的子树内，距离 u 为 d 的点尽量匹配后，剩余的未匹配点数。初始化 $f_u[0] = 1$ ，其余状态为 0，然后从下往上合并。合并儿子 v 时，如果 $f_u[d+1] = f_v[d] = 1$ ，就将这两个点匹配，它们的 LCA 是 u ，因此答案增加 $2(d+1)$ 。随后令

$$f_u[d+1] \leftarrow f_u[d+1] \oplus f_v[d].$$

其中 $\oplus$ 表示异或，即两个点配成一对后不再向上保留。

Solution 1

长链剖分优化。设 h_u 为 u 向下的最长路径上的点数， f_u 的长度就是 h_u 。

每个非叶节点选择 h_v 最大的儿子 s ，沿这些边划分长链。

注意到最长儿子的数组可以直接继承： $f_u[d+1] = f_s[d]$, $0 \leq d < h_s$ 。

让同一条长链上的数组共享存储、错开一位，就不需要复制这些状态。只需令 $f_u[0] = 1$ ，再枚举其余儿子 v 的 h_v 个状态进行合并。

每个非最长儿子都是一条长链的链首，其 h_v 恰好是这条链的长度。所有长链互不相交，因此这些 h_v 的总和不超过 n ，合并的总复杂度为 $O(n)$ 。共享后的数组也只需要 $O(n)$ 空间。

总时间复杂度和空间复杂度均为 $O(n)$ 。

Solution 2

考虑直接 DFS 模拟前述的贪心策略，返回值表示子树内剩下多少个未匹配的点。如果当前子树内剩下的未匹配点超过 1 个，就尽可能多的组合。

因为要分别处理不同深度的所有节点，为了降低 DFS 的复杂度，可以对每一层的节点建立虚树后进行 DFS，这样总复杂度 $O(n \log n)$ 。

C. Cyclic Isomorphism

Description

给定 n 个字符串 S_i ，给定一个字符串 Q 。

总共 M 次询问，每次询问一个 x, y ($x < y$)。

问有多少 $Q[x, r]$ ($r \geq y - 1$)，可以拆分成 ABC ，其中 B 为某个 S_i 的循环同构， $A = C$ 且 C 的起点要在 y 上。

$$\blacksquare \sum |S_i| \leq 5 \times 10^5, |Q| \leq 10^6, m \leq 10^6$$

Solution

首先考虑对 S_i 倍长，然后将所有的 $S_i + S_i$ 建立广义后缀自动机，建立出 fail 树，可以在 fail 树上通过倍增找到每一种循环同构表示对应所在的节点 p ，将其长度记录在 p 上。

考虑枚举 A, C 长度，这一过程其实可以通过二分 + hash 或者 sa 求出 lcp 来加速这个过程。之后就可以得到一段合法的 A, C 串长区间，接下来的问题变成只需要考虑在一个长度区间内合法的 B 即可。同样的我们需要快速定位 $Q[y-1]$ 在 fail 树上的对应节点。由于查询可以离线，可以按 y 排序，在广义后缀自动机上线性地跑匹配。这部分也可以把 Q 直接加到广义 sam 中，但是会造成不必要的空间开销，可能会 mle。

最后的问题是在 fail 树上求一个点到某个祖先的路径权值和，注意 fail 树上每个点可能会挂很多合法的长度，在求前缀和的时候需要扣除掉一些部分。

时间复杂度为 $O(N \log N + Q + M \log N)$ 。

I. Ice Skating

Description

给一个 $n \times m$ 的方格图，其中 k 个格子（是格子，不是格点）上有障碍物。

每次选一个运动方向（只能是向下或者向右），会沿该方向一直移动，直到被障碍物阻挡或即将出界时停止（会停在有障碍物或出界的前一个格子），该过程视为一次移动，不可中途停止。

共 q 次操作，每次操作是询问或者修改。

询问：给定一个起点坐标 (x, y) ，问最少移动几次，可以离开方格边界或陷入无法移动的局面。

修改：给定一个没有障碍物的坐标 (x, y) ，向该位置添加障碍物。

■ $1 \leq n, m, k, q \leq 10^5$

Solution

一共有 $n \times m$ 个格子，但实际上需要考虑的，只有边界上的格子和障碍物左边或上边的格子，共 $O(n + m + q + k)$ 个格子，不妨把这些格子称为关键点。每次移动后，一定会停在一个关键点上，且关键点上最多只有一个方向可以移动，停在另一个关键点上，因此所有关键点可以构成一个森林。记录每个点的深度，就可以快速完成询问。

添加一个障碍物，会导致添加和删除一些关键点，并相应的改变一些连边。考虑维护每次修改操作所引起的森林形态变化，用 lct。

但如果在某一行或列集中了大量障碍物，而在下一行或列，从右往左添加大量障碍物，则可能会导致每次修改操作都要大规模的修改树形态，复杂度会退化为 $O(q(n + m) \log(n + m + q))$ 。

Solution

添加障碍物的操作可能导致大规模的树形变化，考虑删除一个障碍物。删除一个障碍物同样会导致添加和删除一些关键点。可能会添加的关键点只有一个，即障碍物本身的位置，可能会被删除的关键点有两个，即障碍物本身位置的左边和上边。由于设置关键点是不影响结果的情况下减少计算量，因此添加关键点的操作是必要的，而删除关键点的操作是不必要的，可以通过仅修改点权的方式消除本该删除的关键点对答案的影响。考虑把每个关键点裂成两个点，一个点代表从上方到达该点的路径，一个点代表从左边到达该点的路径，当这个关键点被删除时，从上方到达该点的路径继续向下延伸找到新的父亲，从左侧到达该点的路径继续向右延伸找到新的父亲，这样可以让森林形态的变化变得很小。

找一个由点和到达方向组成的二元组的父亲，可以用 `set` 实现。同样用 `lct` 维护，时间复杂度 $O(q \log(n + m + q))$ 。

G. Grand XOR Counting Problem Challenge

Description

给定 n, k, m 和长度为 n 的整数序列 a , 满足 $0 \leq a_i < m$ 。

对于每个 $0 \leq x < m$, 求有多少个下标组

$$1 \leq i_1 < i_2 < \cdots < i_k \leq n$$

满足 $a_{i_1} \oplus a_{i_2} \oplus \cdots \oplus a_{i_k} = x$ 。

其中 $\oplus$ 表示异或操作, 答案对 998244353 取模。

- $1 \leq n \leq 10^6, 1 \leq k \leq n, 1 \leq m \leq 2^{20}$ 。

Solution

令 B 为不小于 m 的最小的 2 的幂，将频数数组补零至长度 B 。

记录 $f[v] = \sum_{i=1}^n [a_i = v]$ ，先对 f 做一次 FWT。

记 $\chi_s(v) = (-1)^{\langle s, v \rangle}$ ，其中 $\langle s, v \rangle$ 为二进制对应位乘积之和模 2，那么

$$b_s = \text{FWT}(f)_s = \sum_{i=1}^n \chi_s(a_i).$$

注意到 $\chi_s(a_i)$ 只可能是 1 或 -1 。若其中有 t_s 个 -1 ，有

$$b_s = n - 2t_s, \quad t_s = \frac{n - b_s}{2}.$$

Solution

考虑答案经过 FWT 后的第 s 项。由于 $\chi_s(u \oplus v) = \chi_s(u)\chi_s(v)$ ，它等于

$$g_s = \sum_{1 \leq i_1 < \dots < i_k \leq n} \prod_{j=1}^k \chi_s(a_{i_j}) = [z^k] \prod_{i=1}^n (1 + z\chi_s(a_i)).$$

每个因子表示一个下标选或不选，因此已经保证下标互不相同，每种方案只算一次。

根据 t_s 的定义，上式就是

$$g_s = [z^k] (1 - z)^{t_s} (1 + z)^{n - t_s}.$$

所以只需求出所有 $h_t = [z^k] (1 - z)^t (1 + z)^{n - t}$ ， $0 \leq t \leq n$

令 $g_s = h_{t_s}$ ，再做一次 IFWT 得到答案。

Solution

接下来考虑如何快速求出所有 h_t 。

把 h_t 看成 n 个符号中选出 k 个的乘积之和，其中 t 个符号是 -1 ，其余是 1 。分别将这 n 个位置中的一个符号翻转，再把得到的答案相加。

翻转一个 1 后得到 h_{t+1} ，翻转一个 -1 后得到 h_{t-1} ，所以总和为 $(n-t)h_{t+1} + th_{t-1}$ 。

另一方面，对于每个固定的 k 元子集，翻转选中的 k 个位置会使贡献变号，翻转其余 $n-k$ 个位置不会改变贡献。因此

$$(n-t)h_{t+1} + th_{t-1} = (n-2k)h_t, \quad 1 \leq t < n.$$

也就得到了一个三项递推。

Solution

初始时所有符号都是 1，因此 $h_0 = \binom{n}{k}$ 。

将上述翻转过程用于 $t = 0$ ，又有 $nh_1 = (n - 2k)h_0$ 。于是

$$h_1 = (n - 2k)h_0 \cdot n^{-1}, \quad h_{t+1} = ((n - 2k)h_t - th_{t-1})(n - t)^{-1}.$$

递推范围为 $1 \leq t < n$ ，所有分母都在 $[1, n]$ 内，逆元一定存在。线性预处理逆元和组合数所需的阶乘，即可在 $O(n)$ 时间内求出全部 h_t 。

最后令 $g_s = h_{t_s}$ ，对 g 做 IFWT，输出前 m 项即可。

总时间复杂度为 $O(n + B \log B)$ ，空间复杂度为 $O(n + B)$ 。

Thank you!