

北京理工大学第十届程序 设计新生赛正赛题解

(可能是近年来最简单的一场新生赛)

出题组合影~ (cf id)

- BFZD: D题出题人; (灰名都能来出题? 后台真的硬)
- BIT_Falfa: K题出题人;
- Delayi: L题出题人;
- fried-chicken: A、J题出题人;
- mlcd: I题出题人;
- sleep__: C、F、G题出题人;
- strawberry: B、H题出题人;
- xzm2022: E题出题人;

- 题解整理: fried-chicken

A.lzd的人生经验

- 惯例的阅读理解送分题~
- 只要认真读题，计算并比较能不能在 m 秒跑完即可；
- 注意根据2中的提示，本题需要使用long long数据类型；
- 相信本题的提示对于整场比赛的许多其他题也会有帮助~

B.Candies

- 首先想到分别对每包只有 2 颗的糖果和每包只有 3 颗的糖果分别讨论，计算是否能均分为三等份。
- 观察样例二发现，还可以让其中一个人拿 2 包 3 颗的糖果，剩下 2 个人拿 3 包 2 颗的糖果。这种情况下等价于将 2 包 3 颗糖果以及 3 包 2 颗的糖果打包成 1 包 6 颗的糖果。不妨考虑更一般的情况，枚举每包有 3 颗的糖果中拿出 x ($x \% 2 = 0$) 包打包成 1 包 6 颗的糖果，那么问题转化成：
 - - 每包 2 颗的糖果要能三等分
 - - 每包 3 颗的糖果要能三等分
 - - 每包 6 颗的糖果要能三等分
- 时间复杂度： $O(N)$

B.Candies

- 暴力解法 by 验题人:
- 由于我们知道糖果的总数, 记为 sum , 因此若均分则一定每人拿 $sum/3$ 个 (当然 sum 得是3的倍数) ;
- 那么首先, 暴力处理出所有使 $2x + 3y = sum/3$ 的解 (x, y) 并存起来, 假设有 L 个这样的解;
- 然后, 写一个这样的三重循环:
- `for(i in [1,L]) for(j in [1,L]) for(k in [1,L])`
- 来暴力验证三个人按照某种方案拿 $sum/3$ 时是否可以满足条件即可;
- 时间复杂度: $O(TL^3)$

C. 异或

- 首先考虑如何判断有没有一段区间异或为0。我们可以计算前缀异或和，如果前缀异或中有两个值相同，则说明这两个值对应的区间异或和为0。
- 因此，我们只要构造出这个前缀异或和序列，保证其两两不同就可以了。
- 接下来说明构造出长为n的序列的充要条件是 $x \geq k$ ，之中k为n的二进制最高位对应的数字（即 $k=1000\dots$ ），并说明如何构造出结果：
- 前缀异或和中最大值的二进制最高位必然大于等于k。而显然有一些数在这一位上为0，因此原序列至少有一个数该位为1。要令这样的数只有一个，且就为k，只需要令前缀异或和中最高位是0的数排在一起，最高位是1的排在一起，并让01交界的位置异或和为k即可。例如：011, 010, 001, 101, 100
- 直接输出 $\text{lowbit}(1), \text{lowbit}(2) \dots \text{lowbit}(n)$ 也是一种解，这样构造出的前缀异或和就是格雷码。

D. 饱了没红包

- 考虑满减红包是满 x 元减 y 元，那么对于所有花费大于 x 元的订单，一定是优先使用所有可以用的红包中节省金额最大的红包。
- 所以将红包与订单一起按**订单价格和红包满减需求金额**从小到大排序。
- 我们遍历所有的订单，在遍历订单过程中，用堆维护所有当前可以使用的红包（即满减需求金额小于等于当前订单价格的红包），对当前订单，从堆中选取红包金额最大的红包使用，订单花费减去红包金额就是当前订单的花费、并从堆中删除这一红包。

E. 技能加点

- 经过无比艰难的理解题意后，可以看出本题里，当你一次运动使用了 e 的能量后，净赚的GST为（字丑求不吐槽qwq）：

$$E^{0.49} - \frac{\text{round}(e \cdot (9.18 R^{-0.33} - 1.69))}{e} \cdot X$$

- 之中round为四舍五入；
- 又因为经过无比艰难的理解题意后，发现这里的 e 要求为0.2的倍数，所以记 $e = 0.2z$ ， z 取任意正整数；式子变为：

$$\text{记 } e = 0.2z$$
$$E^{0.49} - \frac{5 \cdot \text{round}(z \cdot (1.836 R^{-0.33} - 0.338))}{z} \cdot X$$

E. 技能加点

- 于是题意为，E、R可以通过加点来操作，z可以任选正整数，使该式子取到最大值；

记 $c=0.2z$

$$E^{0.49} - \frac{\text{round}(z \cdot (1.836R - 0.338))}{z} \cdot X$$

- 可以看出，E越大可以使这个式子越大；R越大可以使上式圈一部分更小，而圈一部分更小就意味着在保持round后变为1的前提下，我们可以选择一个更大的z，这是我们所希望的（因为z在分母上）；
- 因此，最终的做法是枚举E加了多少点，剩余的点全都加到R上，并求出能保证round里为1的最大的z；枚举的这些情况中取一个最大值，就是本题的答案；

F. 落石

- 直接暴力让每块石头一格一格落下去是会超时的，我们考虑如何让石头一步到胃。
- 容易发现把石头切成一行一行的不影响结果（即把一块 $h*w$ 的石头视为 h 块 $1*w$ 的石头，便于处理），因此我们可以从下往上逐行处理，记录每一列 x 当前最高点的位置 $p[x]$ 。
- 当处理到一块位于 $[l, r]$ 的石头时只需要对 $p[l], p[l+1] \dots p[r]$ 取最高点记为 q ， q 即为当前这块石头的最终高度。
- 记录答案后更新 $p[l], p[l+1] \dots p[r]$ 为 $q+1$ 即可。
- 复杂度 $O(n*m)$ 。

G. 三角力量

- 出完之后发现有原题（洛谷P1989 无向图三元环计数）
- 坏消息是有原题，好消息是不用写题解了（x
- 还是魔改了一下，加上了重边。做法是类似的，见原题题解。只需要记录重边数量，计数的时候乘一下就可以了。

H. 还是分糖果

- 碎碎念：出题人最 naive 的想法是考查大家的哈希技巧，赛场上发现大家做法五花八门，感觉很欣慰！
- 这里介绍三种比较好写的做法（预处理、哈希、莫队）。
- 做法一：预处理
- 考虑为序列 A 和 B 的每个元素维护一个最早出现的位置，那么每次查询可以转化为判断前缀 $A[1 \cdots x]$ 中所有元素的出现位置是否均在前缀 $B[1 \cdots y]$ 中出现。可以为 A 和 B 每个前缀维护当前元素最早出现位置的最大值，那么每次询问只需要判断 A 中最大值是否小于 y 且 B 中最大值是否小于 x 即可。
- 时间复杂度： $O(N \log N)$

H. 还是分糖果

■ 做法二：哈希

- 由于询问是一个前缀，且重复出现的元素只考虑一次，因此直接对序列 A 和 B 的前缀中出现的元素种类进行哈希（需要特判重复元素），对于每次询问判断 A 和 B 前缀的哈希值是否相同即可。
- 哈希方法多种多样，保险起见还可以做双重（多重）哈希，推荐大家学习一手 Zobrist Hash（https://en.wikipedia.org/wiki/Zobrist_hashing）
- 时间复杂度： $O(n)$

■ 做法三：莫队

- 对询问离线的查询完按右端点分块，使用黑科技莫队算法（<https://oi-wiki.org/misc/mo-algo/>）计算即可，每次用 `std::set` 分别维护当前前缀 A 和 B 的集合内元素。
- ps. 验题时候有人用了这种做法，出题人表示很惊讶，但讨论后认为有能力写出莫队算法的选手也值得更高的排名，所以也没卡，快夸夸我这个良心的出题人！
- 时间复杂度： $O(n\sqrt{n})$

I. Jump

- 本题是一道比较板的二分图博弈题目，因为比较板，赛前我还很愉快的预测过会有多少人过这道题（？
- 所以预测结果算是挺准的（？

二分图博弈普及度感觉蛮低的，应该算难题，不用担心被光速ak了

冠军崽崽没跑

俺觉得沈帝不会

俺觉得zzlkq会

19-信科-去睡！

至少y fz队友会

去年ccpc长春出了个挺板的二分图博弈，不妨假设年轻选手有不少会的

王马老吉 2022/5/19 0:07:30

俺觉得zzlkq会

哎，感觉不一定了

2020 China Collegiate Programming Contest Changchun Onsite

Final standings

Double click (or ctrl+click) each entry to view its submission history

#	Who	=	Penalty	A	B	C	D	E	F	G	H	I	J	K	L
1	USA1: ksun48, ecnerwala, scott_wu	11	1029	00:00	+5		+3	+3	+	+	+	+	+	+1	+1
2	* _KING_: HIR180, yutaka1999, maroonk	11	1479	00:09	+1	-1	+	+	+	+	+	+	+	+4	+4
3 (9)	Radewoosh	8	755	00:07			+	+	+	+	+	+	+	+5	+5
4 (11)	IEThargy: mango_jessi, YaoBig	8	1082	00:08	+4		+1	+	+	+	+	+	+	+1	+1
5 (36)	XHSJCE: NeuraXmy, Metacau, Pecco	6	821	00:09			+	+	+	+	+	+	+	+1	+1
6 (59)	Farhod_Farmon	5	611	00:10			+	+	+	+	+	+	+	+1	+1
7 (61)	HCMUS-Iluminati(NGXQuang, NOThuc_PVAn), Greymane Silverfang, xuanquang1999, ZoonK	5	626	00:06			+	+	+	+	+	+	+	+3	+1
8 (64)	So.Proficient: vandoor, Durant_Lee, ShqLiao	5	673	00:10			+	+	+	+	+	+	+	+2	+2
9 (70)	Lai Q Tago: _Viole_, cqvzzx, mico	5	703	00:08			+	+	+	+	+	+	+	+1	+1
10 (103)	ddlddd: fallaves01, smzzl, CMG tmk	4	416	00:07			+	+	+	+	+	+	+	-3	-5
11 (157)	Ars.vou.a.student?: sahdsg, konghonglee, heyuhhh	4	916	00:14	+1		+	+	+	+	+	+	+		
12 (206)	LSP steaming bun: Pedestrian1, soxsz, lmsh	3	288	00:08			+	+	+	+	+	+	+	-1	
13 (327)	Triwater: 1427314831a, Ryker0923, et1 tay	2	79	00:22			+	+	+	+	+	+	+		
14 (330)	foreverlasting	2	80	00:19			+	+	+	+	+	+	+		

H是那个二分图博弈，我们观察到一些年轻人并没有过

我们再看看补题情况

好，发现上述年轻人里，只有zc补了这题

I. Jump

- 下证棋子从起点出发，建图为二分图：

- $a = 0$ 时，显然其为二分图

- $a \neq 0$ 时：

$\gcd(a, b) \neq 1$ 时，可以同时将 a, b 除以 $\gcd$ ，可知该行为对二分图与否没有影响；

$\gcd(a, b) = 1$ 时，此时有两种情况：

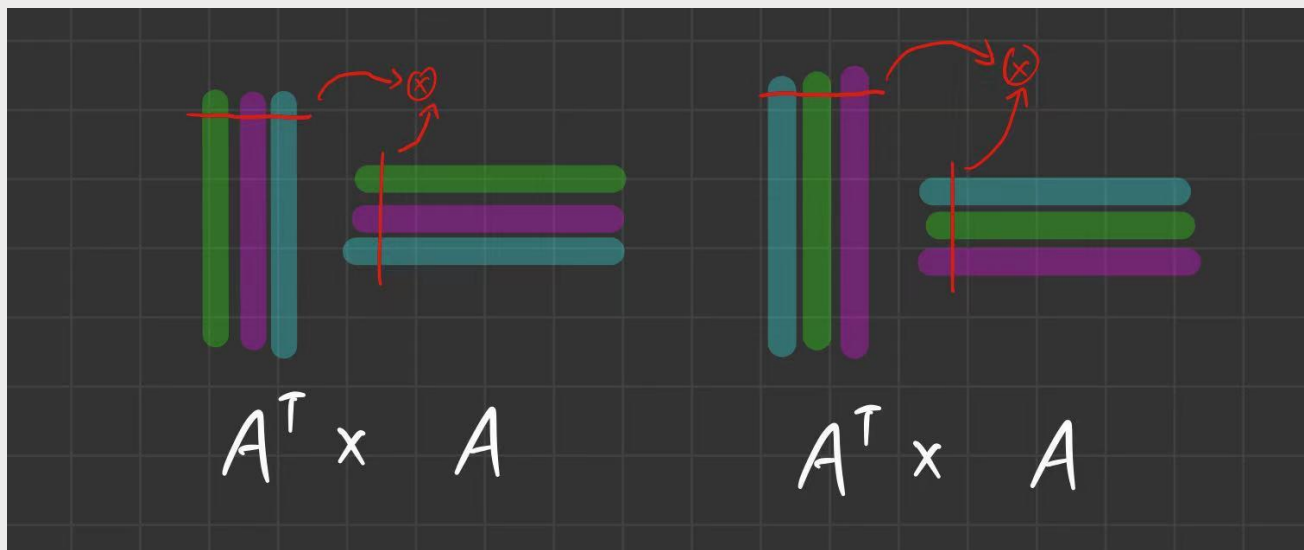
- 1、 a, b 一个奇数，一个偶数。此时每走一步会导致行列之和的奇偶变化，根据行列之和黑白染色，每走一步颜色变化，得到的为二分图。
- 2、 a, b 均为奇数。此时每走一步必然导致行数（或列数）的奇偶性变化。根据行数（或列数）黑白染色，每走一步颜色变化，得到的为二分图。

I. Jump

- 二分图博弈有以下结论：起始点在该图的所有最大匹配中，则先手必胜，反之后手必胜。
- 解决时，可以先不加起点跑一遍匈牙利算法，再加入起点跑一遍匈牙利算法。若结果相同，则后手胜，反之先手胜。
也可采用dinic，只需在跑完第一次dinic的图的基础上跑第二次dinic，看有没有增广路即可。
- std采用dinic，460ms。另有验题人的匈牙利做法，4800ms。为保证大家的良好体验，时限开到5.5s。出题人和验题人都自带大常数，建议卡不过去的好好反思自己的代码习惯。

J.runaway problem

- 虚假的零时困境续作：零时困境 II；
- 真正的零时困境续作：本题
- 真正来自零时困境原班人马（？）的全新坑人题目；
- 注意到交换两行的操作**完全不会改变**式子 $A^T * A$ 的值；
- 所以直接判断 $A^T * A = B$ 最开始是否成立即可；（wa的大多数矩阵乘和转置没处理好）
- 原理（抽象版）：
- 即A交换行后，发现仍是对应颜色相乘再相加



J.runaway problem

- 题面里还玩了一些梗：
- runaway有轻易获得的意思，表示这题十分简单；（梗来自《弥留之国的爱丽丝》）

3 轻易的；迅速的；难以控制的
happening very easily or quickly, and not able to be controlled
a runaway winner/victory 轻易获胜的人；轻而易举的胜利

- 在题面第一段又臭又长的背景里，其实直接给出了这题的答案；

Recently, lzd has been learning two-view correspondences and geometry using order-aware network, which is a neural network that infers the probabilities of correspondences being inliers and regresses the relative pose encoded by the essential matrix. The order-aware network contains three novel operations: differentiable pooling layer(a permutation-invariant methods to cluster nodes in a learnable manner and capture the local context), order-aware differentiable unpooling layer(to upsample the coarsened feature maps and build a hierarchical architecture), and order-aware filtering block(to capture the global context with spatial connections). Since the order-aware network achieved the state-of-the-art(SOTA) performance, lzd considers it a good methods and want to share a problem, which is relevant to one of the operation mentioned above. for you to solve.

- 即后面提到本题出自上述三种操作之一，而操作一就是本题的操作，注意到括号中就写着该操作是permutation-invariant的；

K. 零时困境 II

- 考虑到只要生成了一个1，最终答案就是1；
- 观察随机数生成代码，可以认为这个随机数生成器的概率基本上是均等的，所以不生成出任何1的概率为 $\left(1 - \frac{1}{k}\right)^n$ ，由于 $n > 100*k$ ，所以这个概率小于 $\frac{1}{e^{100}}$ ，直接输出1即可；
- 其实很多选手都没有用正确的方式读取输入的unsigned long long，但由于输出1和输入也没什么关系，所以也都可以正常通过此题；

L. 核酸排队

- 因为每检测10个人就需要花费3min 去整理核算物资，所以显然收集核酸样本数目越少的队列越好，所以令

$$mn = \min_{i \in \{1 \dots n\}} S[i]$$

- 则答案为：

$$m + \lfloor \frac{m + mn}{10} \rfloor \times 3$$