2023年牛客寒假集训营第5场

花絮

A：一开始觉得来点典题，给萌新学点二分和前缀和，所以思考了这样一道题。

B：某一天智乃说想出一个情侣玩的博弈游戏，一方只想着输，另一方只想着赢的游戏，但是我构造不出来完美的题，但是灵机一动想到了一方必输的情况，且还可以偶尔平局让对方开心的这题，本题本来还有一个hard版是个高级的 7 次对不同东西前缀和的DP，但是由于可以被分类讨论之后逐个oeis掉，就不想出了。

C：本来想在愚人节出的，当时以为十分的傻逼，就没有出，但是后来定睛一看，好像不那么傻逼就又拿回来了。

D：idea来源于炸鸡块君的某次校赛，想到了和女朋友打游戏的快乐（tong ku）经历，改编了一些，算上一个比较经典的数据结构题。

E：去年寒假比完想到的一个思维题，但是苦于没有位置放，本来想丢暑假多校的，但是多校已经丢了一个构造了就没丢出去，十分美妙的构造题，爱死了。

F：看到一道典题随意改编了一下，没想到兰子说他不会，所以就出出来了，没想到居然还有一个坑，当时样例给的太强，突然意识到欸，他能wa人，所以就出出来了，告诫大家写典题不要一眼，多思考一下。

G：本人是极度狂热的格雷码爱好者，经常拿格雷码玩一玩，所以出了这样的一道题，算是和E一起出的，但是没地方丢，就丢回来了。

H：起初是因为，我出校赛的时候，不小心在第二签随意丢了一道等差序列求和公式的题，没想到全部爆零了，然后痛定思痛想怎么简单一点，所以思考了这样的一道题，但是思考了没位置丢，刚好这缺签到就拿过来了。

I：当时bot搭载了修仙系统，玩的人有点多，十分感谢湖南农业大学黎**同学在聊天的时候提了一嘴，然后想到的idea。私密马赛，idea侠。

J：本来是兰子和嘤嘤讨论的red题，但是当时他们讨论了半天都不会，我拿过来画一画之后觉得挺简单的，但是他们说十分的毒瘤，我就拿过来当压轴的，我相信大家的强大的构造技巧，一定不会说他难的，确信。

K：实在是想不到好的签到题了，然后枚举了一下幼儿园时候的游戏，发现欸，这游戏不错可以出成题，随后出成了一个最短路题，但是又害怕思维难度会不会太高了，所以先出一道签到来证明一下结论。

L：本来是一道最短路的，但是后面发现这玩意好像还可以直接DP，正好原先的压轴DP被我删了，所以改成了DP题。

A:小沙の好客

代码：https://ac.nowcoder.com/acm/contest/view-submission?submissionId=60724555

解题思路

选择最多 k 个不大于 x 元的商品，那么考虑贪心，我们一定是在不大于 x 的范围内选择 k 个最大的商品，如果没有 k 个就全部选完。

那么我们就需要先对商品排序，排序之后就可以二分找到那一部分是不大于 x 的商品，随后选择最大的 k 个商品。

由于是多次询问，当 k 较大时，我们使用循环一个一个取时间复杂度最坏达到 $O(n^2)$ ，这将会导致超时，但是我们可以发现我们取的是一段区间以内的值之和，对于区间问题我们可以采用前缀和求解。

所以我们可以先对所有商品进行一次排序，之后建一个前缀和数组进行区间的查询，随后每次询问采用二分查找的方式找到最大的合法商品。

tag：前缀和，二分

B:小沙の博弈

代码：https://ac.nowcoder.com/acm/contest/view-submission?submissionId=60724563

解题思路

考虑需要使自己的字典序比对方小，那么每次都尽可能的少选，由于每次必须选择一个，那么双方都会一直选择一个直至石子被取完。

当石子数目为偶数时，双方的取石子的次数相同，得到的字典序也相同，所以平手；

当石子数目为奇数时，先手会比后手多取一次，所以得到的字典序大于后手，所以后手获胜。

tag：贪心

C:小沙の不懂

代码1：https://ac.nowcoder.com/acm/contest/view-submission?submissionId=60724570

代码2：https://ac.nowcoder.com/acm/contest/view-submission?submissionId=60724576

解题思路

本题主要考查，前导 0 带来的大小影响。

思路1

我们可以一开始将两个字符串长度提升至相同长度，短的字符串前导数字默认为 0 。随后将字符串从前往后第一次出现的每一个关键数对进行储存起来，这样的数对最多不超过 $(m+1)^2$ 个。随后枚举全排列，进行判断字符串谁大谁小。当任意情况均为一方大时，那么他们存在恒定大于的关系，否则无法判断。时间复杂度为 $O((m+1)^2 \times m!)$ ，这里的 m 为字符集大小。

思路2

由于当字符串长度不等时，一定存在长的大于短的，那么我们仅需要去寻找当长字符串尽可能小时，是否小于短字符串即可判断，这里可以根据自己的思路自由发挥。

tag：贪心 或 全排列

D:小沙の赌气

代码1：https://ac.nowcoder.com/acm/contest/view-submission?submissionId=60724578

代码2：https://ac.nowcoder.com/acm/contest/view-submission?submissionId=60724583

思路1

使用STL中的set进行进行模拟推关卡操作，比较考验选手的代码能力。

思路2

使用离散化并查集操作，模拟推关卡的操作，由于每个关卡的关键点最多为推动 1 次，加上排序以及离散化的时间，所以时间复杂度为 $O(n \times log_2 n)$ 。

tag：数据结构

E:小沙の印章

代码：https://ac.nowcoder.com/acm/contest/view-submission?submissionId=60724586

解题思路

出题人也不知道怎么想到的，提出这个idea之后，画了半小时就画出来了。

方案可以看一眼代码十分的美观且美妙。

这里直接证明为什么可行：

首先对于最小次数一定为 $n-3$ 。

对于由于每次会选择一对交换位置，那么他们交换之后一定也相邻，所以不会增加新的印花，但有可能另一侧会有新的印花产生，所以对于交换的数而言，他每轮最多产生 1 个印花，由于一开始拥有了自己与相邻的 2 个印花，所以他还需要至少 $n-3$ 轮才能收集其其他印花。

还有另一种情况，对于自己本身不交换，使相邻的两个，与他们各自相邻的数交换，这样会使得该人本轮印花数增加 2 个，但下一轮无法增加，因为下一轮交换过来的一定也是已有的。所以均匀下来每轮产生的印花数为 1 ，该点最少需要 $n-4$轮才能收集所有印花，但是由于该方案在 $n-4$ 轮收集全的条件为第一轮就不进行交换，所以只要第一轮存在交换的就无法满足所以人均在 $n-4$ 轮收集完所以印花，所以从获取信息的数目来看，最少需要 $n-3$ 轮才能使得每个人全部收集完所有印花。

我们假设保证 1 不动，并删除 1 号点，那么有一条 $2 \to n$ 链，我们规定链从左到右放置，小号在左。

思考他们的移动轨迹，即奇数次奇数位向右移动，偶数位向左移动，偶数次奇数位向左移动，偶数位向右移动。该运动轨迹，会使初始奇数位先向右移动，直至最右位，停止移动一个轮次之后向左移动，每轮次移动一个单位。

则对于每个数，他们均会向左或向右移动直至链首尾，等价于与 1 相邻一次，由于每轮次移动一个单位，由于有他们移动到首尾的最长距离为 $n-3$ （例如数字 3 他到数字 n 的距离为 $n-3$，而数字 2 以及处于首尾所以为 0 ，数字 n 同理）。所以数字 1，与每个数字相邻一次的最小轮次是 $n-3$ 轮。

对于其他数字的数字对：

当两数为不同方位时，相向而行时，他们的间距一定为奇数，且他们均会到达链首尾，所以他们一定会相邻。

相互远离时，这部分可能有点难想，但是我们可以预先推测出他们在 $n-3$ 轮后所除位置，随后进行倒推，情况与上诉情况相同，所以一定会相邻。

当两数为同一方向时，他们均会到达链首尾，由于到达先到达链首尾的数会暂停一轮次的移动，所以他们之间的距离一定会转变成一个奇数，且由于后到达的数也一定会在 $n-3$ 轮内到达，所以他们一定会相邻。

所以其他数对也均会相邻。

tag：构造，人脑枚举

F:小沙の串串

代码1：https://ac.nowcoder.com/acm/contest/view-submission?submissionId=60724589

代码2：https://ac.nowcoder.com/acm/contest/view-submission?submissionId=60724597

解题思路

思路1

算比较经典的典题了，贪心的考虑，由于每次操作可以使得更大的数字往前移动，所以可以采用单调栈来维护一个单调递减的数字串。

关键点来了，如果对整个串维护完之后，如果存在多余的操作没有使用，则可以选择最后几个最小的数字将其放在后面，使得前面曾被移动的到后面的数字更加靠前。随后将所有被移动后的字符排序之后，接在最后面即可，因为我们的操作其实是不分先后顺序的，我们可以先将需要删除的最大的字符进行删除，使得最后他尽可能的靠前。

思路2

采用子序列自动机的方式，使得每次贪心的选取最大的一个字符，随后将未选择字符进行一个排序接在已选择的后方即可。

tag：数据结构维护贪心，字符串

G:小沙の编码

代码1：https://ac.nowcoder.com/acm/contest/view-submission?submissionId=60724603

代码2：https://ac.nowcoder.com/acm/contest/view-submission?submissionId=60724607

解题思路

怎么赛上全是网络流的，早知道加大数据范围了，不过感觉也卡不太死的样子。

考虑格雷码性质，对于相邻的每个点他们仅有一位不同，由此延申，距离为 2 的点有两位不同，或者相等，由于每个数字仅出现一次，所以不可能出现相等情况，所以一定会出现两位不同。当知道距离为 2 的点的值之后，我们可以将将该点的值抽象成边，他连接着两个合法的解。

例如：

$0\ x\ 5$，他们中间的值只可能为$1\ or\ 4$，所以我们对节点 1 与节点 4 连上一条边。

由于我们最后的构造方案等价于，每一条边会选择两侧中的一个端点，所以如果我们需要合法匹配，那么对于每一个匹配联通块，我们均需要保证该联通块是一颗基环树，因为对于一棵树而言，他是 n 个点， $n-1$ 条边无法构造全匹配的情况，所以他一定是一颗基环树。

知道他是许多棵基环树之后就可以采用匹配了，这里有验题人采用匈牙利匹配，但是由于出题人卡不死所以放过去了，理论上来说时间复杂度为$O(\left\lfloor \frac{n^2}{4} \right\rfloor)$，但是在采用动态清空之后，他跑的飞快。

考虑正解，在已经有了基环树森林之后，贪心的匹配，对于每一颗基环树的叶子，他仅有一条边，所以使用拓扑排序可以将每个点以及他的唯一边进行匹配。最后会剩下来许多的环，对于环的分配我们可以采用拆环的方案。

由于最后一个点无法根据左右两边推测出他的值，所以我们可以采用并查集维护基环树的方式，最后选择一对可能出现的边进行构造基环树。

tag：基环树，贪心，匈牙利

H:小沙の店铺

代码：https://ac.nowcoder.com/acm/contest/view-submission?submissionId=60724613

解题思路

模拟这个操作，有几个容易出现问题的点：

1，没开long long

2，没有记录曾出售过的数目

3，以为是实时涨价，导致计算错误

实现方案很多，可以随便点开一个代码看。

tag：模拟

I:小沙の金银阁

代码：https://ac.nowcoder.com/acm/contest/view-submission?submissionId=60724628

解题思路

仔细品读题目，可以将整个题面翻译一下：

不会亏灵石 指 前缀已经使用的灵石数不大于当前论次所压灵石数

前缀相同的情况瞎，第 X 轮获胜时总灵石数越多越优指，前缀和尽可能最大。

当能够翻译出以下两点就有许多方法解决了，由于 $m \leq 10^{15}$ 所以，只要当 $n > 1 + log_2 m$ 时，就会导致无解，此时 n 很小，可以采用暴力转移的方式进行移动，也有采用二分的。

这里题解给出一个贪心解：我们假设 x 轮的前缀和为 s_x，则有前 $x-1$ 轮前缀和为 $\left\lfloor \frac{s_x}{2} \right\rfloor$，因为这样能保证前缀尽可能的大，且前缀已经使用的灵石数不大于当前论次所压灵石数，均符合题意，所以该解正确。

由于每轮灵石数均为正整数，所以记得检查是否出现所压灵石数为 0 的情况。

tag：贪心，前缀和

J:小沙の最短路

代码：https://ac.nowcoder.com/acm/contest/view-submission?submissionId=60724643

解题思路

考虑构建最长的最短路，那么有，尽可能没有分叉路径，即起点到终点只有一条路径，其次基于迪杰斯特拉的更新想法，如果存在一个点，起点到他的距离较大，需要保证该点的可达点中，仅只有一个点未被更新，也代表这他的可达点中存在以被更新点，也就是需要保证以被更新点无法到达他，这暗指需要尽可能的通过单向边去构建整张图。

根据以上思路可以推理出几种图形，这里std采用螺旋形进行构造。

当存在一个中心点为起点，且外圈可以进入内圈，但内圈无法直接跑出外圈仅留一条单向通道缓慢延申即可。

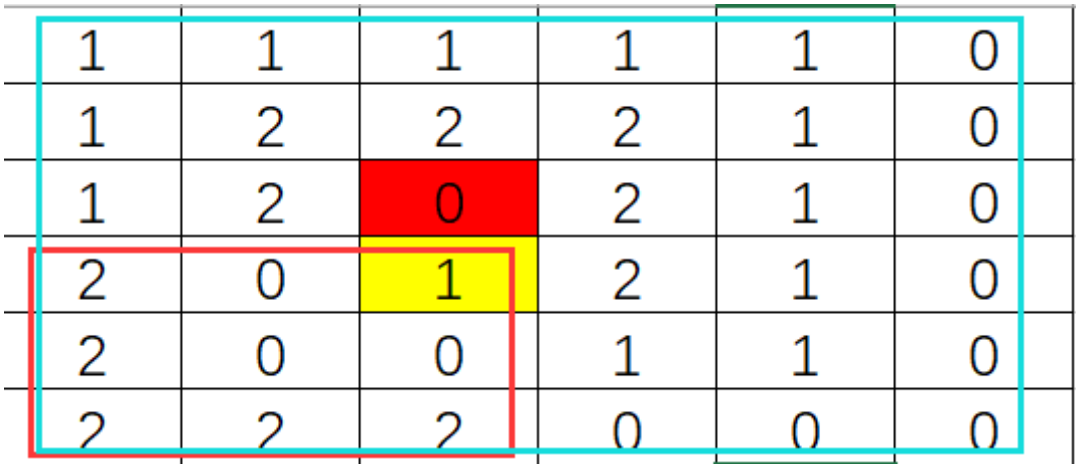
以边长为6的网格为例：

1	1	1	1	1	0
1	2	2	2	1	0
1	2	0	2	1	0
2	0	1	2	1	0
2	0	0	1	1	0
2	2	2	0	0	0

假设红色为起点，我们发现以该起点为圈，出现内圈无法到达外圈，但外圈可以进入内圈。由此给出解。

模拟该操作即可。

这里给出一个发现的规律：



我们发现底下一个方形，以黄色为中心块，可以按照距离进行分层，整体的方块除红色方框外，以红色为中心块，按照距离分层，所以利用该方法，本体题代码极为简洁。

tag：构造，最短路思想

k:小沙の抱团 easy

代码：https://ac.nowcoder.com/acm/contest/view-submission?submissionId=60724648

解题思路

我们发现每次操作其实等价于需要让 n 整除一个数 x，随后舍去其余数。那么我们想要使得余数尽可能的大，则需要使 x 尽可能的大，但是当 x 变大超过 $\lceil \frac{n}{2} \rceil$ 之后，会使得余数逐渐变小，所以我们考虑在 $\lceil \frac{n}{2} \rceil$ 附近选取，当 n 为整数时，如果选取 $\frac{n}{2}$ 那么会导致余数为 0，所以我们需要多选择 $\frac{n}{2}+1$，这样我们可以使得 n，减少 $\frac{n}{2}-1$。当 n 为奇数时，如果选取 $\lceil \frac{n}{2} \rceil$，会减少 $\lfloor \frac{n}{2} \rfloor$。此时他们均是最大的，同时可以发现，当 n越大时，他们的最大余数也非递减，所以可以考虑贪心策略。

tag：贪心

L:小沙の抱团 hard

代码：https://ac.nowcoder.com/acm/contest/view-submission?submissionId=60724743

解题思路

考虑DP转移，即对于每一条指令 x_i ，均会存在一条 $k \times x_i + (1..k-1) \rightarrow k \times x_i$ 且代价为b_i，由于每一条指向边均为端点大的指向小的，将大端点向小端点转移权值可得到最小代价，时间复杂度 $O(nm)$。

tag：动态规划，最短路。