2022河南萌新联赛第（一）场 题解

非常抱歉前期没有及时看到问题，回答问题。

A. Alice and Bob

我们可以对 n 进行质因数分解，那么其实就是把 n 分成了一些石子，对于每个素因数p 都看作一堆石子，那么每次操作转化为从某堆石子拿走一些石子，那么就转化为 $Anti-Nim$ 博弈。

根据 $Anti-Nim$ 游戏结论，当且仅当所有堆石子数等于 1 并且所有堆石子数量异或和为 0 (有偶数堆)或者存在一堆石子数大于 1 并且所有堆石子数量异或和大于 0 先手必胜；反之必败。所以先分解质因数，然后使用$Anti-Nim$ 博弈。

时间复杂度为 $O(\sqrt{n})$

参考代码：std1 std2

B. 打对子

对于 Alice 和 Bob 两个人，分别判断每一个字母数量的奇偶性，若为奇数，则该牌打对子将无法打完。

时间复杂度：$O(n)$

参考代码：std

C. 割竿榄

首先判断下割m次能否把n棵竿榄都割到。

情况一 如果不能都割到或者（即3 * m < n），那么只需找到长度为3*m的一段土地，使得权值最大，然后m次全部割掉即可。

情况二 如果能够全部割到（即3 * m >= n），那么我们可以发现有两种情况，一种是一些土地割的次数均匀（即n % 3 == 0），这时为了割的竿榄最多，我们先不割竿榄（割掉的长度为0），让竿榄生长，当剩余的割竿榄的次数刚好把所有竿榄割一次的时候再割（全部割掉）。另一种情况就是割的次数不均匀（即n % 3 ！ = 0）例如土地长度为4或5时，割两次的话会有重合的部分，这是我们需要考虑是否全割掉，当长度为4时，第一次割需要保留1单位，第二次把后面部分全部割掉。当长度为5时，两次都全部割掉。可以发现长度为4或5时，采用相应的最优策略计算结果一致（长度为4情况是有一块土地留了1单位没割掉，长度为5情况是有1单位的土地不能生长了，都是跟理想情况下相比少了1单位）。

代码实现思路

首先默认为全部割掉，先把全部割掉的部分记录，再计算生长的部分。

可以先计算前缀和，然后根据两种情况分别计算初始的竿榄。然后可以利用循环来计算每一段生长的部分。

时间复杂度：$O(n)$

参考代码：std

D. 纪念品领取

方法一：模拟

对于每次的抽签，我们可以直接将其往后放，即对于第 i 次抽签对应的人，我们就将其位置信息更新为 $n+i$，最后通过排序，寻找位置最小的五个人按序号大小输出即可。

时间复杂度: $O(n \log n)$

参考代码：std

方法二：模拟

我们也可以使用倒着进行考虑，对于放到最后，倒着考虑就是那些被抽到人的倒着的顺序。
然后记录一下，就可以处理出结果了。

时间复杂度: $O(n)$

参考代码：std

E. 聚会

我们考虑增量的方式，当前表示的和是 $[1,x]$，还没有选择的里面最小值是 a_i。

> 1. $a_i > x+1$ 时，$x+1$ 表示不出来，所以答案就是 $x+1$ 。
> 2. $a_i <= x+1$ 时，那么表示的和就变为 $[1, x+a_i]$ ，对于小于 x 的原先就可以表示，对于大于 x 的，我们可以使用 a_i 和原先的进行组合。

所以我们进行排序，依次进行增量。

时间复杂度：$O(n \log n)$

参考代码：std

F. 买车

在每一次车没有电的时候，我们考虑从前面经过的所有店铺里面能走的最远的一个店铺买车，这样可以保证使用最少的次数到达目的地。

提供一个贪心正确性的简单证明：假设我们当前选择的最优店铺为 x ，最远店铺为 y $(x \leq y)$ 。那么我们下次所进行选择的集合 S_x 是最远店铺选择集合 S_y 的一个子集。那么对于每一次的选择，我们都有一个更优的选择去有可能减少选择次数，即 $ans_x \geq ans_y$ 。这显然与我们所作的假设不符。

时间复杂度：$O(n \log n)$

参考代码：std

G. 热身小游戏

方法一：线段树

将 q 次操作看成一个长度为 q 的序列，初始值都是 1 。

对于第 i 次操作，如果是操作 1 则将下标为 i 的位置修改为 a，对于操作 2则进行区间修改为 1，对于操作 3 就是区间求乘积。

时间复杂度为$O(n \log n)$

参考代码：std

方法二：并查集

对于一个已经删除的一些点来说，我们可以进行使用在线段上往后跳，就是删除过的点我们不重复删除，所以可以使用并查集快速向后跳。
如果同时使用路径压缩和按秩合并时间复杂度为 反阿克曼函数，我们这里为了简单就只使用路径压缩了。

时间复杂度 $O(n \log n)$

参考代码：std

H. 兴奋值

十分抱歉，没能出足够强的数据，让一些人使用了一些小优化过了。
赛后已加强，希望使用正确的做法解决这道题。

方法一: 二分 + st表 / 线段树

这里我们发现对于一次询问来说，询问中的点 $i-l+1$ 是随着 i 单调递增的，那么我们使用二分答案，对于答案 x 来说，我们只需要求出 $[l+x-1,r]$ 中 a_i的最大值，然后判断 a_i 最大值是否大于等于x来判断是否成立。

这里如果使用st表时间复杂度是 $O(n\log n)$ ，线段来维护区间最大值时间复杂度为 $O(n\log^2 n)$ 都可以过。

时间复杂度：$O(n\log n)$

参考代码：std

方法二：离线询问 + 线段树

我们考虑一个位置来说，这个元素在查询的时候占主导地位的是谁，是 $a[i]$ 还是 $i-l+1$。
如果 a[i] 是最小值(主导)的话 $a[i]\le i-l+1\ \Rightarrow\ l\le i+1-a[i]$
就是说当询问的左边界 $l\le i+1-a[i]$ 的时候，最小值(主导)的都是 $a[i]$。
所以我们可以离线进行处理，对每个询问按照 l 进行排序，当处理到这个 l 的时候，那么将这个位置更换主导进行修改，在线段树中从主导 $a[i]$ 替换成 $i-l+1$。

时间复杂度：$O(n\log n)$

参考代码：std

I. 巡逻机器人

通过思考分析可以发现，这题中两个机器人相遇后会反向巡逻的条件，其实是可以忽略的，因此这题对于机器人的状态，逐个分析即可。忽略相遇反向，看作一直在环上走。

方法一: 枚举

那么其实就是找到每一个门口最晚被机器人巡逻到的时间。
对于每个门口被巡逻到的时间，其实就是在其左边离的最近的一个向右(R)的机器人、在其右边离的最近的一个向左(L)的机器人 这两个机器人中到达该门口时间短的那个时间点。我们可以对此进行维护查询，找到所需时间最大的门口的值即为答案。

时间复杂度：$O(n)$

参考代码：std

方法二：二分 + 差分

对最终的答案进行二分，那么二分的验证就是判断是否巡逻完所有门口，这里可以借助差分对机器人巡逻门口进行统计，从而统计所有门口是否已经巡逻。

时间复杂度：$O(n \log n)$

参考代码：std

J. 樱果运输

可以看做是一个二维费用的背包问题，两个约束条件为卡车数量和资金数。

最后控制资金数循环判断卡车数即可。

时间复杂度：$O(n * n * y)$

参考代码：std

K. 糟糕的一天

签到题。题目要对于每一天判断当前后面有没有大于当前的，有就统计。
我们倒着进行处理即可。

时间复杂度：$O(n)$

参考代码：std