Solution

卡西欧质数(casio)

今日签到。显然是使用筛法的题目。

- 考虑埃式筛的过程，每次取一个质数后标记其倍数为合数，那么我们取质数的时候只要对不超过 P 的质数标记其倍数即可。

  后续将这些 卡西欧 质数取出来做二分就可以解决问题。

  时间复杂度为 $O(n \log \log n + T \log n)$。期望得分 60。

- 考虑使用线性筛求出卡西欧质数。注意到卡西欧质数的定义是最小的质因数大于等于 P。

  线性筛的本质过程其实就是使用最小的质因子筛出所有合数。因此我们可以使用线性筛晒出所有质数的同时记录所有合数的最小质因子。 随后就可以取出所有卡西欧质数，询问使用二分处理即可。

  时间复杂度为 $O(n + T \log n)$。期望得分 100。

推车(car)

> Codeforces 581E Kojiro and Furrari

很显然我们不会往左走，因为最开始的时候，推车是满的，并且装的就是优先级最高的推车人员需要的零食。因此我们只会往右走。

首先将零食店按照坐标从小到大排序。

我们假设不存在 SS 和 SN 的零食店，在路上只使用 SM 零食，每到一个 MS 零食店就补充满的情况下，从第 i 家 SM 零食店到 t 还缺多少零食。记作 $f_{2,i}$，从右往左递推：

$$f_{2,i} = f_{2,i+1} + \max(0, x_{i+1} - x_i - s)$$

其中 x_i 表示第 i 家 SM 零食店的坐标。

接着我们假设不存在 SS 零食店，在路上只使用 SM 和 SN 零食，每到一个 SM 或 SN 零食店就补充满的情况下，从第 i 家零食店到 t 还缺少多少零食，记作 $f_{1,i}$，同样的从右往左递推，方程相同。

最后考虑所有的零食店，答案记作 $f_{0,i}$，同样递推。

接着我们考虑每个询问，首先判断无解，我们先二分找到第一家坐标大于起点的零食店，判断两者距离是否不大于 m 且这个零食店到 t 是否需要额外零食，这一步我们已经预处理出来了。

然后考虑需要多少 SS 的零食，就是假设不存在 SS 零食的时候这个零食店到 t 还需要的零食数量，这个我们也预处理出来了。

最后考虑需要的 SN 的零食数量，答案是不存在 SS 和 SN 时这个零食店到 t 还需要的零食数量减去需要的 SS 的数量。这样我们就解决了问题。

优秀的拆分呢 (excellent)

需要明确的一点是，不考虑优秀如何，仅考虑长度因素，对于一个字符串 S，其拆分可能的量级是 $O(n)$ 的，即枚举 $|A|$ 的长度之后， $|B|$也就确定了。

做法一

暴力枚举子串及其拆分位置，枚举复杂度为 $O(n^3)$，暴力 $O(n)$ 判断前后两部分是否相等，时间复杂度为 $O(n^4)$。 常数为 $\frac{1}{4}$，能不能过 $n=200$ 看姿势水平了。

做法二

想要稳定的过掉 $n=200$ 并不困难，不难发现做法一中存在很多重复的部分，本质上，我们只是判断 S 内的子串 T 后的 $|T|$ 个字符能否和 T 匹配上而已，可以简单的记 $f_{i,j}$表示$S_{i...i+j-1}$是否与 $S_{i+j...i+2j-1}$匹配。

直接暴力预处理，时间复杂度为 $O(n^3)$。期望得分 40

做法三

做法二中复杂度 $O(n^3)$ 的部分有两个，一个是枚举子串及其拆分，一个是求 $f_{i,j}$，在全 `?` 与无 `?` 的两种数据中，$f_{i,j}$ 都可以 $O(n^2)$ 求出(哈希)。

枚举子串及其拆分的部分也可以换种思路，不妨考虑枚举字串开头以及 AA。我们记 g_i 表示 $f_{i,j}$ 中可以匹配的位置数量。那么我直接枚举字符串开头和 $|A|$ 之后，直接使用 g 就可以求出多个子串的一种拆分了。时间复杂度为 $O(n^2)$。

结合做法 2 可以得到 65 分。

做法四

做法三的瓶颈在于求出 f 和 g 时的判断匹配过程。固定左端点时匹配的变化很大。不过终归可以接受 $O(n^2)$ 的时间以及空间复杂度，因此考虑记录 $h_{i,j}$ 表示从 i,j 开始的前缀的最大匹配长度。使用 h 就可以简单的求出 f 和 g 了。而求 $h_{i,j}$ 就可以递推了，使用 $h_{i+1,j+1}$ 可以轻松求出 $h_{i,j}$。

时间复杂度为 $O(n^2)$。期望得分 100。

机器人 （robot)

> JOISC2022 Day1T1 监狱

结论：如果存在合法方案，则一定存在一组合法方案时对于每一个机器人 i，直接从 p_i 移动到 q_i。

证明：首先要明确，一个机器人 x 阻挡机器人 y 至多一次。在此前提下，我们钦点一个人如果走到一半要停下来等另一个人走，那么我们选择让另一个人先走完再走。既然有解，这种阻挡关系不存在环，因此一定可以找到这种阻挡关系的终点。

接下来我们只考虑给一个顺序，每次让一个机器人从 p_i 移动到 q_i。

显然对于机器人 i ，如果存在一个机器人 j ,使得 p_j 在路径 $p_i \to q_i$ 上，则 j 应该在 i 前移动，即连一条 $j \to i$ 的边。如果存在一个机器人 k 使得 q_k 在路径 $p_i \to q_i$ 上，则 i 应该在 k 之前移动。即连一条 $i \to k$ 的边。

然后直接拓扑排序，如果出现环则说明不存在合法方案，这样我们能做到 $O(n^2)$ 的复杂度。

复杂度瓶颈在于建图。需要注意的是，由于我们是有向图，树上原本的无向边并不好直接使用。

考虑把 u,v之间的边通过树上的点中转，对于每一个点，我们复制一份，形成 x_i, x_i'。考虑如下建图:

对于第 i 个人，设 $p_i \to q_i$ 的路径中，除 p_i 外走到的第一个点是 p，除 t_i 外到达的最后一个点是 q。

- 对于 $u \in path(p_i, q)$,连边 $i \to u$;
- 对于 $u \in path(p, q_i)$，连边 $u' \to i$;
- 连边 $q_i \to i, i \to p_i'$

这样，如果 p_i 位于 $path(p_j, q_j)$ 上，可以通过 $i \to p_i' \to j$从 i 到达 j ;如果 q_i 位于 $path(p_j, q_j)$ 上，可以通过 $j \to q_i \to i$ 从 j 到达 i 。并且这样建图不会造成多余影响。

最后一个问题：一个点向一条链连边，可以树剖线段树建图，边数为 $O(n \log^2 n)$。倍增可以做到 $O((n+m)\log n)$ 的复杂度，不过 std 好像常数有点过大了。实际上两种做法时间几乎不会有区别，树剖建图的边仍然是 $O(m \log n)$级别的，只不过找到点需要两个 $\log$的时间。

实际表现结果来看，树剖跑的快很多。

动态树 （lct）

可以 LCT ，但是我选择线段树分治。

首先按照套路，我们把每条边按出现/消失的时间挂在线段树区间上处理。

gcd 等于 1 有些难受，考虑莫比乌斯反演。

设 $g(n)$ 表示 gcd 恰好为 n 的方案数，$f(n)$ 表示 gcd 为 n 或 n 的倍数的方案数。

那么 $f(n)=\sum\limits_{n|d}g(d)$，反演一下 $g(n)=\sum\limits_{n|d}\mu(\frac{d}{n})f(d)$。

要求的是 $g(1)$ 也就是 $\sum\limits_{i=1}^{m}\mu(i)f(i)$。

对于 $f(i)$ 相当于把树中边为 i 的倍数的拎出来，答案是 $\sum C^2_{\text{连通块大小}}$，很好整。

我们对值域中每一个值 i 开一个并查集，处理各自的 $f(i)$。

可以发现 $\mu(i)=0$ 的那些 i 毫无用处，直接不管。

这样，对于我们要添进去的一条边权为 v 的边，质因数分解 $v=p_1^{k_1}p_2^{k_2}\dots p_r^{k_r}$，只要枚举那些 μ 值不等于 0 的因数，也就是该因数只有选或不选质数 p_i 两种情况，而 $r\le 6$，所以最多只有 64 种可能需处理的因数，在这些因数对应的并查集里连接这条边。

但是并查集直接开满空间会炸。

一种做法时把询问分成 B 份，一份一份搞，适当地调整 B 的取值使得时空复杂度平衡。

第二种做法是把每个 $\mu\neq 0$ 的 i 单独处理，每个 i 把时间离散化后跑线段树分治。

第三种做法，事实上只有 $64(n+q)$ 种不同的边，$128(n+q)$ 种不同的点，所以我们动态开点，做一个哈希表（map又慢空间又炸，别懒了，手写吧）。

写个挂链哈希表，空间就可以开小一点，但空间越小效率越低，所以开一个比较平衡的大小。