A - 限速（speed）

n 个点 m 条边的无向图，选出来一棵生成树。

每次可以调整一条边的边权 -1 或 $+1$ 。

求最小修改次数使得生成树上边权最大值 $= k$ 。

A - 限速（speed）- 第 1 档部分分

> 保证所有道路的 $s_i \le k$。

总有一棵生成树包含边权最大的边，因此答案就是 $k - \max s_i$ 。

A - 限速（speed）- 第 2 档部分分

> 保证所有道路的 $s_i \geq k$。

选择的生成树一定是最小生成树。

A - 限速（speed）- 满分

> $2 \le n \le 2 \times 10^5, n-1 \le m \le \min(2 \times 10^5, \frac{n(n-1)}{2}), 1 \le k \le 10^9$

首先找出来最小生成树，分两种情况考虑：

- 如果最小生成树上的最大边权 $\ge k$ ，那么最终选中的树一定就是最小生成树。
- 否则，你可以选择最接近 k 的那条边。

复杂度 $O(m \log m)$ 。

B - 酒鬼 （drunkard）

一条街道上有 n 个路灯排成一排，编号 $1, 2, \ldots, n$ 。小 Z 喝醉了，初始在时刻 0 时站在 1 号路灯旁。

小 Z 从某个时间 t_0 ($t_0 \geq 0$) 开始游走。在时间 t_0 之后，小 Z 每隔一个单位时间就会移动到相邻的路灯旁。如果小 Z 在时间 t 在 i 号路灯旁，则他在时间 $t+1$ 必然移动到 $i-1$ 号路灯或 $i+1$ 号路灯旁。特殊的，小 Z 不会移动到街道边界之外，即他始终只停留在路灯 1 和 n 之间（包含边界）。例如，小 Z 在时间 t_0+1 必然在 2 号路灯旁。

你得知了一些路人提供的信息，每条信息都可以用整数 p_i 和 q_i 表示，代表在 q_i 时刻某位路人看到到小 Z 在路灯 p_i 旁边。你想找到小 Z 开始到处走动的时间 t_0 。在收到信息的同时，你还想根据当前收到的信息推断 t_0 可能的最大值和最小值。

路人提供的信息也有可能不一致。一旦你发现提供的信息有矛盾，只需停止计算并报告信息不一致。

B - 酒鬼（drunkard）- 第 1 档部分分

> $m=2,\ p_i\ge 2,\ op_1=$ **clue**，$op_2=$ **min**，所有线索一致

直接枚举可能的最小值即可。

B - 酒鬼 （drunkard） - 第 2 档部分分

> $m = 2,\ p_i \geq 2,\ op_1 =$ **clue**， $op_2 =$ **max**，所有线索一致

直接取 $q_i - (p_i - 1)$ 即可。

B - 酒鬼 （drunkard） - 第 3 档部分分

> $1 \le m \le 1000$，只存在 **clue** 和 **min** 操作，所有线索一致

注意到答案只可能是 0 或 1 ，模拟判断一遍是否合法。

B - 酒鬼 （drunkard） - 第 4 档部分分

> $1 \le m \le 1000,\ p_i \ge 2$

对于最小值，只可能是 $0, 1$，直接每次 $O(m)$ 检验即可。

对于最大值，每次加入新元素后计算出该条信息对应的可能的最大值，检验其是否符合其他信息即可。注意处理冲突的情况，例如两条信息间隔时间不够移动对应的距离。

复杂度 $O(m^2)$ 。

B - 酒鬼 （drunkard） - 第 5 档部分分

> $p_i \geq 2$

如果已知的信息 $p_i \neq 1$ 时我们就可以确定酒鬼移动了。

此时首先要检查所有这样的信息 $p_i + q_i$ 的奇偶性一致，并且相邻时刻的信息对应的移动距离不能超过他们的时间差，这可以通过 set 维护 pair 支持插入。

注意最早的信息关于起点的距离也有限制。

B - 酒鬼（drunkard）- 满分做法

对于 $p_i = 1$ 的情况，不用保证 $p_i + q_i$ 奇偶性一样，但如果奇偶性不一样，则要保证时刻小于最晚移动时间，因此在处理 **min** 询问的时候要谨慎。

其余部分和前一档做法一致。复杂度 $O(m \log m)$ 。

C - 自驾游 (trip)

比特国有 n 个城市，编号 $1, 2, \cdots, n$，由 m 条**单向**的公路连接。

第 i 条公路从第 a_i 个城市出发，前往第 b_i 个城市，路程是 d_i 米。

小 Z 决定从 S 号城市出发，最终到达 T 号城市（保证 $S \neq T$），并且你的车子初始速度 v 为 1 米每秒。

全国共有 p $(0 \leq p \leq n)$ 个维修站，第 i 个位于第 x_i 个城市，小 Z 可以停留在这里花 c_i 秒的时间将车升级，使得车速翻倍。小 Z 可以在某个维修站将车升级多次，但每次升级都会花 c_i 秒的时间。例如，假设小 Z 当前的车速为 v，并且恰好在第 x_i 个城市，那么他可以在第 i 个维修站花 $3c_i$ 的时间将车升级三次，使得车速变为 $2 \times 2 \times 2 \times v = 8v$。

C - 自驾游 (trip)

假设小 Z 的车速是 v 米每秒，那么安全通过一条长度为 d 米的公路所需的时间为 $\lceil \frac{d}{v} \rceil$ 秒（$\lceil x \rceil$ 代表最小的不小于 x 的整数）。然而，有些公路是平整的，而另一些则是坑坑洼洼的。每当小 Z 通过一条坑坑洼洼的路，车的所有升级都会损耗，通过这条路之后车速会立即变回 1 米每秒（但在通过这条路的过程中车速不会改变）。

现在小 Z 想知道，他到达 T 号城市最少需要花多少秒?

C - 自驾游 (trip) - 第 1 档部分分

> 对于 10% 的数据，保证 $n = 2, m = 1$。

判断是否连通，然后枚举翻倍次数即可。

C - 自驾游 (trip) - 第 2 档部分分

> 对于另外 20% 的数据，保证所有的 $d_i = 1$ 。

所有边长度都是 1，由于通过时间为上取整没必要提高车速。

相当于图没有边权求最短路，BFS 即可，复杂度 $O(n+m)$ 。

C - 自驾游 (trip) - 第 3 档部分分

> 对于另外 20% 的数据，保证 $p=0$ 。

没有维修站，车速恒定为 1 ，使用 Dijkstra 算法求解最短路即可。

复杂度 $O(m\log n)$ 。

C - 自驾游 (trip) - 第 4 档部分分

> 对于另外 20% 的数据，保证 $d_i \le 100$ 。

注意到车速超过 100 后通过每条路的时间就不会有变化了。

建立分层图，将图复制 100 层，第 x 层的第 u 个节点代表到达城市 u ，车速为 x 。

通过一条好的路：$(u,x) \to (v,x)$，边权为 $\lceil \frac{d_i}{x} \rceil$

通过一条不好的路：$(u,x) \to (v,1)$，边权为 $\lceil \frac{d_i}{x} \rceil$

维修站升级车速：$(u,x) \to (u,2x)$，边权为 c_i

然后在新的图上运行 Dijkstra 即可，复杂度 $O(\max d_i \times m \log n)$ 。

C - 自驾游 (trip) - 满分做法

注意到车速只会变成二倍和回到 1 ，因此车速一直是 2 的幂次。

建立分层图，将图复制 20 层，第 x 层的第 u 个节点代表到达城市 u ，车速为 2^x 。

通过一条好的路：$(u, x) \to (v, x)$，边权为 $\lceil \frac{d_i}{2^x} \rceil$

通过一条不好的路：$(u, x) \to (v, 0)$，边权为 $\lceil \frac{d_i}{2^x} \rceil$

维修站升级车速：$(u, x) \to (u, x+1)$，边权为 c_i

然后在新的图上运行 Dijkstra 即可，复杂度 $O(\log d_i \times m \log n)$ 。

D - 团队选拔（selection）

小 Z 是算法竞赛社团的指导老师，他指导了 N 个队员，编号为 $1, 2, \cdots, N$，编号为 i 的队员能力值为 a_i 。

现在小 Z 要选一些队伍去参加比赛，每个队伍由若干个编号连续的队员组成，也就是说一个队伍可以被描述成一个区间$[l, r]$，由编号为$l, l+1, \cdots, r$ 的队员组成。每位队员最多被选入一支队伍中。

选拔方案可以用一个区间的序列$[l_1, r_1], [l_2, r_2], \cdots, [l_k, r_k]$ 来描述，代表着该方案选拔了k 只队伍，第i 只队伍由编号在$[l_i, r_i]$ 的队员组成。形式化地，要求：

- $1 \le k \le N$
- $\forall i \in \{1, 2, \cdots, k\},\ 1 \le l_i \le r_i \le N$
- $\forall i \in \{1, 2, \cdots, k-1\},\ r_i < l_{i+1}$

D - 团队选拔（selection）

我们称两个**选拔方案**不同，当且仅当两个方案对应的区间的序列不同。

根据过去的经验，一个队伍的综合能力值为这个队伍中所有队员能力值的最大公约数 (gcd)。小 Z 希望他选出的所有队伍综合能力值相同，因此他定义一个**选拔方案**是**好的**，当且仅当

$$\gcd_{i\in[l_1,r_1]}\{a_i\} = \gcd_{i\in[l_2,r_2]}\{a_i\} = \cdots = \gcd_{i\in[l_k,r_k]}\{a_i\}$$

现在小 Z 想知道，对于每名选手，有多少个**好的选拔方案**，会将其选入在某一队伍中。

D - 团队选拔（selection） - 第 1 档部分分

> $n \le 10$。

可以使用搜索的方法，求出所有的不相交的区间的划分。求出每一个选出的区间的 $\gcd$，判断是否相等。如果相等，令对应位置的方案数加一即可。

根据不同的搜索方式时间复杂度有所不同。一种相对简单的搜索实现是根据上一个位置是否被处在一个区间中，枚举当前位置 断开/延续上一个区间/成为一个新区间的开头，时间复杂度为 $O(3^n)$。

D - 团队选拔（selection） - 第 2 档部分分

> 所有选手的能力值都是 1。

那么序列中任一区间的 $\gcd$ 都为 1，不需要关心 $\gcd$ 相等的限制。

设 f_i 表示 $1\dots i$ 的方案数，且满足存在一个区间以 i 结尾。那么 $f_i=\sum_{j=0}^{i-1} f_j$ 。类似的，g_i 表示 $i\dots n$ 的方案数，且满足存在一个区间以 i 开头，$g_i=\sum_{j=i+1}^{n+1} g_j$。

统计答案时，求问题的补，即求有多少种方案不包含一名选手。那么对于选手 i，不包含其的方案应该是 $\sum_{j=0}^{i-1} f_j \times \sum_{j=i+1}^{n+1} g_j$。

使用前缀和优化上述过程，时间复杂度为 $O(n)$。

D - 团队选拔（selection） - 第 3 档部分分

> $n \le 100$。

设 $f_{i,j}$ 表示 $1 \dots j$ 的方案数，且满足 $i \dots j$ 是一支队伍。

那么有 $f_{i,j} = \sum_{x \le y < i} f_{x,y} \times [\gcd(a_x \dots a_y) = \gcd(a_i \dots a_j)]$。

类似的，$g_{i,j}$ 表示 $i \dots n$ 的方案数，且满足 $i \dots j$ 是一支队伍。求出 f, g 后可以使用第 2 档部分分中的求有多少种方案不包含一名选手的方法统计答案。

时间复杂度为 $O(n^4)$。

D - 团队选拔（selection） - 第 4 档部分分

> $n \le 1000$。

一个关于 gcd 的重要结论是对于 $a_1, a_2, \dots, a_i$ ，集合 $\{\gcd(a_j \dots a_i) | 1 \le j \le i\}$ 的大小为 $O(\log a_i)$ 。因为后缀的 gcd 发生一次变化至少会减半，所以变化次数是 $\log a_i$ 级别。那么对于整个序列来说，所有的区间的 gcd 共有 $O(n \log a)$ 种。

于是可以改进第 3 档部分分中的算法，设 $f_{i,w}$ 表示 $1 \dots i$ 的方案数，且最后一段区间以 i 为结尾，gcd 为 w 。尽管 w 可能很大，但根据上面的结论，只有 $\log a_i$ 种 w 是有意义的。

枚举一个 j ，表示最后选择的区间为 j , \$w = \gcd(a_j\dots a_i) \$ 。那么 $f_{i,w} \leftarrow \sum_{k=0}^{j-1} f_{k,w}$ 。采用相似的方法可以计得到 g ，并使用第 3 档部分分中的方法统计答案。具体实现时由于 w 较大，可以先将所有可能的 gcd 离散化。

时间复杂度为 $O(n^2 \log a)$。

D - 团队选拔（selection） - 第 5 档部分分

> 所有的 a_i 都是 2 的整数次幂。

设 $f_{i,w}$ 表示 $1\dots i$ 的方案数，且最后一段区间以 i 为结尾，$\gcd$ 为 2^w。

精细化实现一个单调栈可以求出所有满足 $\gcd(a_j\dots a_i)=2^w$ 的 j 在一个区间 $[l_w,r_w]$ 中。

那么有 $f_{i,w}=\sum_{j=l_w}^{r_w}\sum_{k=0}^{j-1}f_{k,w}=\sum_{k=0}^{r_w-1}f_{k,w}\times(r_w-\max(k,l_w-1))$。对于每一个 w 维护一个前缀和即可。

时间复杂度 $O(n\log^2 a)$。

D - 团队选拔（selection） - 满分做法

采取类似第 5 档部分分中的单调栈可以在 $O(n\log a)$ 的时间内求出所有形如 (l,r,x,v) 的四元组，表示 $\forall i\in[l,r], \gcd(a_i\dots a_x)=v$。这些四元组能表示所有区间的 gcd。

对于每一种 gcd 分别求解，考虑使得所有选出的区间的 gcd 为 w，设 pre_i 表示从 $1\dots i$ 中选出若干 gcd 为 w 的区间的方案数，beh_i表示从 $i\dots n$ 选出若干 gcd 为 w 区间的方案数。那么对于不包含点 i 的方案数就是 $(pre_{i-1}+1)\times(beh_{i+1}+1)-1$

。

D - 团队选拔（selection） - 满分做法

考虑求出 pre，对于所有 $v = w$ 的四元组，如果不存在一个四元组 (l, r, x, v) 满足 $x = i$，那么 $pre_i = pre_{i-1}$；否则选一段以 i 结尾的区间，那么 $pre_i = pre_{i-1} + \sum_{j=l}^{r} 1 + pre_{j-1}$ 。第一类转移可以直接使用线段树来区间覆盖，第二类转移求一个区间和即可。复杂度是和四元组个数相关的。同理，beh 也可以这样求出。

相同的 pre 和 beh 会构成一段段区间，区间个数即为四元组个数；可以将区间排序后区间覆盖，用差分实现区间加即可。复杂度是$O(n \log^2 a)$。