比赛链接：https://ac.nowcoder.com/acm/contest/326

出题人：teitoku

感谢：winterzz1 提供部分题的部分数据，和tokitsukaze的技术支持

验题人：calabash_boy，Heaven_，yuzuko，lizi_lzy，meopass_0v0，

花絮：

本场比赛中的 B 题和 C 题是被 tokitsukaze 强行要求换的，原计划是一道比较有新意的字符串和一道有坑点的博弈，但是难度可能比现在的 B 和 C 要大一些，就被强行下了，换成了现在这两道比较基础的题。

A、

时间复杂度 O(1)

签到题：要考虑题中定义二进制整数必须是 2 的 k 次方（k>0），因此二进制整数不可能是奇数，所以二进制半整数也不可能是奇数；而且最小应该是 4。

然后二进制整数也可能是二进制半整数。

综合下来就是满足，大于等于 4，并且不是奇数，并且二进制位里有 1 个或者 2 个 1。

B

入门动态规划：

时间复杂度 O(N*(A+B))

dp[5000] [2] [50]

第一个维度表示当前处理到第几个位置也就是 DP 的阶段，第二个维度表示重复的是元音还是辅音，第三个维度表示重复的次数。

对于每一个阶段，考虑 4 种转移。

上一个是元音当前状态也是元音的转移。

上一个是辅音当前状态是元音的转移。

上一个是元音当前状态是辅音的转移。

上一个是辅音当前状态也是辅音的转移。

转移方程：

dp[i][0][j]+=dp[i-1][0][j-1]*5;

dp[i][0][1]+=dp[i-1][1]*5

dp[i][1][1]+=dp[i-1][0][j]*21;

dp[i][1][j]+=dp[i-1][1][j-1]*21;;

C 题

时间复杂度预处理 O(233*233*233)

单次查询 O(1)

弗洛伊德处理 233*232 对之间的转换最短路。

先特判 A 与 B 是否相等，相等就是 0，如果 A 和 B 不相等并且 B 大于等于 233 的话，就是

无解。
如果 A 小于 233，B 小于 233，直接查询弗洛伊德表。
如果 A 大于等于 233 的话我们要先使用 F(X)和 G（X）将 A 变换一步，然后 A 的值就小于 233 了，然后查弗洛伊德表，两者里选取优的方案。
D 题
解法很多，比如 DP,贪心，二分。
题解里提供二分的解法。
时间复杂度 O(T*log(1e18)*2)
我们发现，最后交出来的答案的区间在小进制下一定连续的。
所以我们考虑，所以我们二分出答案区间在小进制下的左右端点，然后做差就是答案。
注意，特判两段区间没交集的时候请输出 0 而不是负数或者其他的结果。

E 题
后缀数组：
时间复杂度 O(SUM(M)*log(SUM(M))+ SUM(M)*N)
我们把歌曲当成字符串。
我们将所有的字符串，加上分割字符之后，跑后缀数组，然后 for 一遍 sa 数组，通过 height 数组，来维护一个 dist 数组，用来代表当前后缀所在的字符串和所有其他字符串的“满足 SA 小于当前 SA 的最近一个后缀”的 LCP，在过程中任意两串之间的关系取最大即可处理出所有不能同时选择的字符串。
然后二进制枚举选择的结果，判断里面是否存在矛盾关系。

F 题
强连通分量+并查集
时间复杂度 O(M)
由于情况非常的复杂，考虑一次性判断完所有情况，可能会写的非常的累。
于是我们可以分成三次检查，每次检查只检查一部分不合法的情况。
第一次强连通算法的检查：
强连通算法跑出来之后，如果存在一个强连通分量中有一些成员拥有照片和一些成员没有即为不合法，如果超过一个强连通分量里有人拥有照片也是不合法。
第二次并查集检查：
对于每一条两个端点都是拥有照片的人的边，我们合并这条边的两个端点。然后依此合并完，如果所有拥有照片的人无法合并到一个集合中，那么我们认为也是不合法。
第三次检查，DFS 检查
如果每一个人拥有照片之后，DFS 到的后继节点没有照片，那么也是不合法的。
如果通过三次检查我们就认为是可行的，不然就是有人在说谎。

上述三重检查，存在一些重复判断，两个判断之间会存在一些重复部分，其实完全可以写在一次 DFS 过程中，使得代码非常的优雅。但是那样写需要比较好的代码能力。但是拆成三次检查，就会降低代码编写的难度，但是会造成代码长度过长，而且也会比较臃肿，所以具体采取哪种写法，任凭选择。