Equal

30%：

每次询问从两个端点开始 BFS 算出每个点到两个端点的距离。

然后暴力即可。

复杂度 O(NM)。

另外 10%：

全部输出 N 即可。

另外 30%：

由于树高有限，所以暴力找到 A 和 B 的中点，然后答案就为“N-A 来的子树大小-B 来的子树大小”。

100%：

对于 A 的深度=B 的深度，中点为 A 和 B 的 LCA。

如果不等于，可以先用 LCA 求出 A、B 的距离，再在树上倍增求出 AB 的中点。

注意特判 AB 中点不存在的情况。

Tree

30%：暴力枚举删除哪些边，然后验证即可。

50%：DP[i][j][k]表示当前子树根节点为 i，与子树中 j 个点相连，是否已经安排了 k 只企鹅。

复杂度 $O(N^3)$。

70%：

考虑 DP。

DP[i][j]表示 i 与子树中 j 个点相连，最多已经安排了多少企鹅。

复杂度 $O(N^2)$。

100%：

一条边可以用两只企鹅站，这样的一条点对，越多越好。

如果是 ans 对点，ans*2≥k，那么只需要 (k+1)/2 条边。

否则，需要 ans + (k-ans*2) 条边。

现在问题就转为求这样的点对有多少。

dp[i][0]表示以 i 为根的子树中能够两两配对的最大点数，不包含节点 i。

dp[i][1]表示以 i 为根的子树中能够两两配对的最大点数，包含节点 i。

转移方程：

dp[u][0]=Σv 是 u 的儿子 dp[v][1];

dp[u][1]=max(dp[u][1],dp[u][0]-dp[v][1]+dp[v][0]+2)。

最后 max(dp[1][0],dp[1][1])就是 ans 了。

Tea

Task1：

就是一个集合，支持加入、删除、撤销和求 mex,维护一个桶就可以暴力了。

Task2：

用数据结构比如线段树、堆之类的来维护。

Task3：

如果我们能够维护出当前被删的最小值，以及在不删除情况下的答案，就可以求出mex。离线处理，知道每一个元素的出现时间区间，按照权值从小到大去覆盖，找未被覆盖的位置用并查集。

Task4：

因为加入互不相同，每次只撤销最早被删除的，维护删除的最小值可以使用单调队列。红茶的编号只需保留到 ，超过的不可能对答案有贡献。