Sol - flower

40%

从 L 向 R 枚举，检查是不是 k 的倍数。

60%

答案是 $\frac{R-L}{k}+1$。

100%

将 L 调整至不小于 L 的第一个 k 的倍数 L'，显然从 L 到 R 的答案和 L' 到 R 的答案没有差别。

将 R 调整至不大于 R 的最后一个 k 的倍数 R'，显然从 L' 到 R 的答案和从 L' 到 R' 的答案没有差别。

然后使用 60% 的做法即可。

调整 L 时，首先看一下它是否已经是 k 的倍数了，如果不是的话，它应该是一个 $ak+b$，只需要加一个 $k-b$ 就可以了，而 b 实际上就是 $L\%k$。

调整 R 时，直接计算 $R'=R-R\%k$ 就可以了。

Sol - chess

30%

读入询问，去暴力跑一遍，复杂度 $O(Qn^2)$。

50%

使用二维前缀和，记 $S[x][y]$ 表示所有满足 $1 \le i \le x, 1 \le j \le y$ 的 a_{ij} 之和，那么当我们想要求出 $a \le i \le c, b \le j \le d$ 的 a_{ij} 之和时，我们只需要计算出 $S[c][d] - S[a-1][d] - S[c][b-1] + S[a-1][b-1]$ 即可。

而计算 $S[x][y]$ 则可以通过 $S[x][y] = a_{xy} + S[x-1][y] + S[x][y-1] - S[x-1][y-1]$。

这两个公式的具体内涵请自行体悟或在网上搜寻资料。

100%

其实和 50% 没有本质区别，我们做 10 个二维前缀和，第 i 个维护大于等于 $i-1$ 的值的个数就可以了。

extra

这题能做到 $a_{ij} \le 10^9$，但是出题组一致认为过于超纲。

本质上就是将询问离线化后动态地维护二维前缀和，这需要二维树状数组或者树套树的帮助。

对于 20% 的数据

直接暴力枚举 $P_1, P_2, \cdots, P_7$ 即可，复杂度 $O(n^7)$。

对于 50% 的数据

首先对 A_i 排序，枚举 P_2, P_3 ，可以发现金牌和铜牌的选取是互不关联的。

铜牌可以贪心的选取小于 P_3 最大的四个。

金牌可以贪心的选取大于 P_2 且小于 $P_2 + P_3$ 最大的一个，这个二分即可。

复杂度 $O(n^2 \log n)$。

对于 100% 的数据

观察到对 A_i 排序后，$P_3, \cdots, P_7$ 总是在 A 数组中连续选取，我们可以基于此对 50 分的做法进行一些优化。

首先对第二个条件移项得：$P_2 < P_4 + P_5 + P_6 + P_7 - P_3$。

我们从小到大枚举 P_2，若一组 $P_3, \cdots, P_7$ 不大于当前的 P_2 ，那么他将对于之后的所有 P_2 都不合法。

所以可以用一个栈来维护 $P_3, \cdots, P_7$ 这些五元组（由于他们是连续的，只需存储一个下标），每次把当前新增的一个五元组入栈，然后把不合法的弹栈，栈顶即为目前合法的最大值；注意到 P_1 的选取区间为 $[P_2, P_2 + P_3)$，取能取到的最大的 P_3 一定最优，二分 P_1 即可，该做法正确性显然。

复杂度为 $O(n \log n)$。

Sol - Tree

40%

由于编号超过 100 的节点都没有用，直接对前 100 个节点建树然后暴力算 lca，复杂度是 $O(Ta)$。

50%

我知道你在想什么，全输出 1 是过不了的。

观察发现，第二天结束时数据范围之外的编号就已经没有意义了，我们不关心。换句话说，可以认为子节点的生长只会发生两次。

在这一情况下，两个不同的点的 lca 只可能是 1 或者 1 的某个子节点，我们考虑什么时候会是第二种情况。

第一天会生长的新叶子编号是 $[2, 1145141919811]$，第二天时，叶子 1 长出了 $[1145141919812, 2290283839621]$，叶子 2 长出了 $[2290283839622, 3435425759431]$。

以此类推，可以认为叶子 x 的新叶子应该是在 $[2 + 1145141919810x, 1145141919811 + 1145141919810x]$ 这个范围内，所以对 a, b 检查一下是否同属某个范围就可以了。

70%

和正解基本上是一样的，但是最后那个预处理没写，多了一个 $\log a$。

话说出题人也不知道会不会有人被卡到这一档。

100%

首先，假设第 i 天开始时有 p 个叶子，那么这一天会长出 kp 个新叶子，最终会得到 $p(1+k)$ 个叶子。

那么根据数学归纳法，可以发现在第 i 天结束时有 $(1+k)^i$ 个叶子。

那么我们只需要找出最大的 x 使得 $(1+k)^x < a$，就可以得知 a 是在哪一天被长出来的。

考虑如果我们已知 a 是哪一天长出来的，我们可以直接求 $a - (1+k)^x$ 来得知它是第几片被长出来的，记为 a'，那么显然它的父亲就是 $\left\lfloor \frac{a'-1}{k} \right\rfloor + 1$。

然后我们会发现，每过一天，树高就会多一层，而 $a, b \le 10^{18}$ 保证所求节点是在相当早的天数时长出来的，这表明对答案有影响的节点的深度都非常低，实际上，这个值的量级在 $\log_3(a)$，大约也就 38。

这个性质允许我们使用暴力的方式计算 lca：把 a, b 的祖先都搞出来，对齐 1 的位置，然后看最后（或者最前，取决于你排列祖先的方式）一个相同的祖先是谁，这个就是答案。

注意一下求父亲的时候可能需要预处理一下 $1+k$ 的次幂，这个可以降低一个大约 $\log a$ 的复杂度。