Sequence：

将 a 序列从小到大排序，考虑比 a_i 小的数 a_j 加上去的情况，只会使 a_i 位数不变或加一，二分临界点即可。

Lock：

答案是这样的居民子集个数 x:重要度的和不足 m，但加入任何一个新居民都将导致重要度的和大于等于 m。

必要性:由于上面的集合重要度都不够,他们都至少缺一把锁。若不足 x 把锁，这些子集中必有两个 x，y 缺同一把锁 l。把这两个子集 x 和 y 并起来，仍然缺 l 这把锁，无法开门，但现在子集的重要度已经达到 n 了，与题目要求矛盾。

充分性:一共 x 把锁，每把锁上面各写一个这种居民的子集(互不相同)。一个居民持有所有上面的子集不包括自己的锁的钥匙，这样满足要求。

注意到如果所有人加起来重要度都不够,则需要一把锁,无人有钥匙。

Square：

对于一对 a，b 答案为 $\dfrac{ab}{\gcd(a,b)^2}$

$$\prod_{i=1}^{n}\prod_{j=1}^{n}\frac{ij}{\gcd(i,j)^2}=\frac{\prod_{i=1}^{n}i^n n!}{\prod_{i=1}^{n}\prod_{j=1}^{n}\gcd(i,j)^2}=\frac{n!^{2n}}{\prod_{i=1}^{n}\prod_{j=1}^{n}\gcd(i,j)^2}$$

于是我们只需要求 $\prod_{i=1}^{n}\prod_{j=1}^{n}\gcd(i,j)^2$

算法一：反演

$$\prod_{i=1}^{n}\prod_{j=1}^{n}\gcd(i,j)^2=\prod_{d=1}^{n}d^{\sum_{i=1}^{n}\sum_{j=1}^{n}[\gcd(i,j)==d]}=\prod_{d=1}^{n}d^{\sum_{i=1}^{\lfloor\frac{n}{d}\rfloor}\sum_{j=1}^{\lfloor\frac{n}{d}\rfloor}[\gcd(i,j)==1]}=\prod_{d=1}^{n}d^{\sum_{i=1}^{\lfloor\frac{n}{d}\rfloor}\sum_{j=1}^{\lfloor\frac{n}{d}\rfloor}\sum_{k|\gcd(i,j)}\mu(k)}=\prod_{d=1}^{n}d^{\sum_{k=1}^{\lfloor\frac{n}{d}\rfloor}\mu(k)\left\lfloor\frac{n}{dk}\right\rfloor^2}$$

不妨设 $f(x)=\sum_{i=1}^{x}\mu(i)\left\lfloor\frac{x}{i}\right\rfloor^2$，我们预处理出 f 数组后可以数论分块得到答案。

考虑预处理 f 数组，不妨枚举 i，然后枚举 i 的倍数 ki，考虑它在[(k-1)i,ki)中的贡献，于是只需要在[(k-1)i,ki)中区间加上 $\mu(i)(k-1)^2$，这个过程可以直接差分后前缀和实现。

时间复杂度 $\mathrm{O}(n\log n+t\log n\sqrt{n})$，期望得分 50。

算法二：

$\prod_{d=1}^{n}d^{\sum_{i=1}^{\lfloor\frac{n}{d}\rfloor}\sum_{j=1}^{\lfloor\frac{n}{d}\rfloor}[\gcd(i,j)==1]}$，考虑 $\sum_{i=1}^{n}\sum_{j=1}^{n}\left[\gcd(i,j)==1\right]$，不妨设 i>j，对于特定的 i，符合条件

的 j 的个数为 $\varphi(j)$，所以 $\sum_{i=1}^{n}\sum_{j=1}^{n}[\gcd(i,j)==1]=2*(\sum_{i=1}^{n}\varphi(i))-1$，所以原式为 $\prod_{d=1}^{n} d^{sum(\frac{n}{d})*2-1}$，sum 表示 φ 的前缀和，线性预处理出 sum 函数后数论分块即可。

时间复杂度 $\mathrm{O}(n+t\log n\sqrt{n})$，期望得分 70。

算法三：

我们考虑计算 $\prod_{i=1}^{n}\prod_{j=1}^{n}\frac{ij}{\gcd(i,j)^2}$ 不妨设 i>j，设 $f(x)=\prod_{i=1}^{x}\prod_{j=1}^{x}\frac{ij}{\gcd(i,j)^2}$，考虑由 $f(x-1)$ 转移到 $f(x)$，我们需要加入 $(\prod_{i=1}^{n}\frac{i*n}{\gcd(i,n)})^2=(\frac{n!*n^n}{\prod_{i=1}^{n}\gcd(i,n)})^2$，$\gcd(i,n)$ 只会是 n 的约数，对于数 x 只有 $\varphi(\frac{n}{x})$，这可以预处理出来，就能 O（1）回答询问。

时间复杂度 $\mathrm{O}(n\log n+t)$，期望得分 70。

算法四：

我们单独考虑每个素数对答案的贡献，对于一个素数 x，它在 kx 处会对答案有（2k-1）的贡献，求一遍前缀积即为答案。

时间复杂度 $\mathrm{O}(n\log\log n+t)$，期望得分 100。