SOL

opt

本题是一道签到题，注意到枚举 $\gcd = x$，然后直接将是 x 倍数的数中 v 最大的两个分别作为 i, j，取最优即可，可以证明，这样求出来的就是答案。

那么直接狄利克雷后缀最大值即可。具体的，我们开一个 a 的桶，存对应的最大与次大的 v，然后枚举质数 p，按 x 从大到小的顺序，将 $x \times p$ 的贡献加到 x 上。容易证明复杂度为 $O(p \log \log p)$。

需要注意，本题需要 $\text{unsigned long long}$ 进行存储。

math

我们显然不关心 x，所以有两个测试点 x 非常大，真去读入估计直接就没了。

由期望线性性，所求即为每个位置是暴击的概率之和，考虑位置 i 暴击的概率为 $\int_0^1 dx \prod_{j=1}^{i-1} \frac{j}{i}x$ 即第 j 个位置实际伤害小于 $\frac{x}{i}$ 的概率之积，那么就等于 $\frac{(i-1)!}{i^i}$ 。

那么分子很容易处理。

考虑如何快速筛出 i^i 。对于前 8 个测试点，直接快速幂即可。接下来注意到 p 最大只有 $4e7$ ，考虑找到 p 的原根 g，那么 $i^i = g^{i \times \log_g i}$ ，而 $\log_g i$ 扫一遍就出来了，所以可以 $O(p)$ 求得答案。

需要注意的是，如果只是朴素实现可能由于常数原因无法通过本题，可以使用一些奇技淫巧来加速。

事实上本题还有一个对任意 p 的优化算法，复杂度为 $O(n \log \log n)$，但由于其常数较大，无法与暴力拉开较大差距。（该算法 1s 时，暴力似乎只需 1.5s ）。

ds

注意到事实上我们进行了 $64 \times q \times len$ 次询问，一次一次做即使 $O(1)$ 也必死无疑，但问的是连续一段，又恰好是 64 个，这不禁令我们想到 bitset。有强制在线的事实也要求我们必须对 q 次询问，每次先快速处理出某个点子树内的 bitset 情况，然后在这个 bitset 上 $O(len)$ 回答每个询问。

我们的第一想法毫无疑问是套路地将子树询问转化到 dfn 区间，然后直接上莫队，但可惜有强制在线。那么改进莫队，令 B 为块长，在序列上预处理出 $[k \times B, l \times B]$ 的情况，然后快速回答每个询问。注意到该算法时间复杂度为 $O(\frac{n^3}{wB^2})$，但空间复杂度高到 $O(\frac{n^3}{8B^2})$ ，导致 B 只能开到 5000 上下。 另一个想法是类似猫树，不过空间复杂度仍然 $O(\frac{n^2}{8B} \log \frac{n}{B})$，且询问常数翻倍，仍然无法通过（不会有人真能卡过去罢）。

正解其实很简单，我们还将问题放在树上考虑，我们处理 $O(B)$ 个点子树内的 bitset，使得无论查询哪个点，都可以找到一个在查询点子树内，且与之只差 $O(\frac{n}{B})$ 个点的 bitset 。具体来讲，如果当前点子树内已经被预处理的最大子树大小和当前点子树大小之差超过 $O(\frac{2n}{B})$ ，那么就预处理当前子树。可以证明其正确性。std 将 B 开到 800 ，每次询问只需额外找最多 500 个点的情况。

本题数据不强，可能有其他做法可以通过。