# Sing Alive

先转化为 $a^2+b^2=(d^2-c^2)+k$.

对于每个 $x$，记录 $f_x$ 表示 $x$ 可以表示成多少种 $a^2+b^2$，$g_x$ 表示 $x$ 可以表示成多少种 $d^2-c^2+k$。

然后就只需要求 $\sum f_x g_x$ 就行了。

复杂度 $O(n^2)$。

# Neo Aspect

称 $s_i=0$ 的元素为 $0$ 元素，反之则为 $1$ 元素。考虑设计 DP 状态 $f(i,j)$ 表示只考虑前 $i$ 个元素组成的排列中，$0$ 元素的最大值是多少。那么考虑插入 $i+1$ 时，如果 $i+1$ 是 $0$ 元素，那么要么 $p_{i+1}\le j$，这种情况的转移是 $j\times f_{i,j}\to f_{i+1,j+1}$；要么 $p_{i+1}>j$，这种情况的转移是 $\sum_{j<k} f_{i,j}\to f_{i+1,k}$。如果 $i+1$ 是 $1$ 元素，那么转移只能是 $(i+1-j)\times f_{i,j}\to f_{i+1,j}$。用前缀和优化，复杂度 $O(n^2)$.

# Oneness

$L=R=1$ 的时候就是有向基环森林。求最短路是容易的。

$L>1$ 的时候可以看作每次选择一个步长 $L\le x\le R$，然后在基环树上沿着边跳 $x$ 的距离。

那么跳 $k$ 步就可以走任意 $\in[kL,kR]$ 的长度。

那么现在就是要处理这样的一个问题：从 $s\to t$ 需要走 $d$ 步（如果 $t$ 不在环上）或者 $pw+q$ 步（如果 $t$ 在环上且环长为 $w$，注意这里 $q$ 可能会 $\ge w$，且 $p$ 可以是自选的任意非负整数）。

前者应该是弱于后者的。我们直接看 $pw+q$ 的情况怎么做。你当然可以直接类欧做。但是下面介绍一种不超纲的办法。设弱连通块大小为 $s$，那么在这个弱连通块的问询的答案应该 $\le s$，并且 $q$ 也应该 $\le s$。

所以我们对于每个这样的连通块，都做一遍下面的操作：令 $f_x$ 表示 $q=x$ 时的答案。枚举答案 $k$，然后更新走的总长度为 $[kL,kR]$ 时可以更新的 $f_x$。注意到如果可以同时更新 $f_{x-w}$ 和 $f_x$，那么我们只需要更新 $f_x$，然后最后再一起从后往前扫一遍并用 $f_{x+w}$ 更新 $f_x$。所以我们只需要选出极大的那些 $x$ 去更新就好了，这些 $x$ 会形成最多两个区间。这就是一个典型的，区间覆盖，然后最后再求出每个点的值。复杂度 $O(s\log s)$。

总复杂度 $O(n\log n)$。代码较复杂。

# Zeal of Proud

首先确定一个性质：假如在目前位置我们可以黏贴，那么我们没有必要先输入几个字符之后再去做黏贴。这样显然没有意义。以下的 DP 状态与转移全部是基于这样的前提限制的。

一种暴力 DP：构建 $s[1,i]$ 的最少步数是 $f_i$，然后转移时枚举上一次复制是哪里。

考虑把转移分步一下。我们可以设一个 $g_i$ 表示刚刚黏贴完得到 $s[1,i]$ 的最少步数。$f_i$ 的转移就是枚举最后一次黏贴的位置，用 $g_j+(i-j)$ 来更新 $f_i$。

现在考虑 $g_i$ 的转移。$g_i$ 的转移就是要找上一次复制的位置 $j$，用 $f_j$ 来更新 $g_i$。

我们发现假如我们复制了 $s[1,j]$，那么在下一次复制之前，我们会在哪些位置黏贴是固定的（一定是从左到右，如果能匹配上就黏贴）。所以对于 $s[1,j]$，其可能会粘贴的位置只有 $\le \frac{n}{j}$ 个，所以可能的 $f_j \to g_i$ 的更新只有 $O(n\log n)$ 种。

现在问题是如何找这样的位置。我们先跑一下 Z 函数（二分+哈希对于 $\le 10^5$ 也行），求出 $r_i$ 表示 $s[i,n]$ 和 $s$ 的 LCP，然后我们相当于只需要每次对于 $i,j$，找到最小的 $k>i$ 使得 $r_k \ge j$。到这里就已经可以 $O(n\log^2 n)$ 解决了。

但是这个问题可以做到更好。我们 $j$ 不断变小时，可行的满足 $r_k \ge j$ 的合法位置会不断变少。所以我们的操作其实是：找到 $i$ 之后的第一个还存在的位置；将一个位置删掉。使用并查集即可。链上并查集可以在线性时间内维护（但用普通并查集应该也能过）。所以总复杂度 $O(n\log n)$。