远恋

只需求出树的个数减去没有边的点个数，后者容易计算。

众所周知树的点数等于边数减一，于是总点数减总边数即为树的个数。

记得用 map 处理重边，复杂度 $O(q\log n)$，如果用哈希表就是 $O(q)$。

心加心

考虑用建图描述本问题，用点 k 表示模 p 等于 k 的数，用边 $x\to y$ 表示状态 x 在后面接一个数能变为状态 y，边长为数字的长度，我们即求从源点开始到每个点的最短路，可以 dijkstra 解决。

直接用 dijkstra 是 $O((p+m)\log p)$，但本题 m 是 $O(pn)$ 级别，于是复杂度到达 $O(np\log p)$。

采用不使用堆优化的 dijkstra 即可，复杂度 $O(np)$。

依存症

先考虑如何求权值，差分，那么操作相当于 flip 至多两个位，我们要把每个位置变成 1。记 w 为差分后为 0 的位置数量，权值即为 $\lceil\frac{w}{2}\rceil$，求和只需考虑求 w 之和与 $[w \bmod 2=1]$ 之和。

前者：

$$f(l,r)=\sum_{i=l}^{r}\sum_{j=i+1}^{r}[s_i=s_j]2^{r-l-(j-i)}$$

$$f(l,r)=f(1,r)-f(1,l-1)-\sum_{i=1}^{l-1}\sum_{j=i+1}^{r}[s_i=s_j]2^{r-l-(j-i)}$$

容易推式子做到 $O(n+q)$。

后者考虑这个值事实上只与子序列的元素个数，以及开头结尾是否相等有关，分开头结尾距离等于 2 和大于 2 的情况，前者满足当且仅当两者值相等，后者就是在 $[i+1,j-1]$ 中选元素个数为奇数/偶数的子序列数量，就是 2^{r-l-2}，同样容易 $O(n+q)$。

左脑右脑

假设度数序列不变应该如何处理，此时代价固定，只需判定是否有解。

假设最小的最大匹配为 L，最大的最大匹配为 R，我们断言能构造出所有 $[L,R]$ 内的最大匹配大小——构造很简单，任取两个度数非零的非匹配点 a,b，取出其连向的任意两个匹配点 u,v，拆掉 $(a,u),(b,v)$ 连上 $(a,b),(u,v)$ 即可从最小匹配向最大匹配依次构造。

R 好求，左右两部非零点贪心连一下即可，重点在如何求 L。

使用 Hall 定理，最大匹配大小为 $|X|+\min_{S\subseteq X}(|N(S)|-|S|)$（$X$ 为左部点集合），也就是找到一个 X 的子集满足 $|X|+|N(S)|-|S|$ 最小。我们枚举 $N(S)$，只需找到连边都在其内部的最大 $|S|$，那么其只要求 S 度数和小于等于枚举的集合，可以 dp 记录左右部点集合大小，子集和进行判定。

如果度数序列不固定，我们考虑对度数序列进行 dp 的过程中记录上面的子集信息，设计一个 dp 求出数组 $g_{i,j,k}$ 表示左部钦定了 i 个点，钦定度数和为 j，有 c 个非零度数点，然后枚举两边信息 $O(n^6)$ 合并即可。