题目大意

- 给定一颗 n 个点的有根树，初始所有点均为白色。
- 你可以执行不超过 k 次操作，每次操作为选定一个点，把它到根简单路径上的所有点涂成黑色。
- 求你最终最多能涂黑多少点。对 $k=1\sim n$ 分别求解。

记对于有标号有根树 T，上述问题在 $k=i$ 时的答案为 $ans(T,i)$。

给定 n,mod，对所有 $1\le k\le n,1\le m\le n$，计算有多少不同的 n 个点以 1 为根的有标号树 T 满足 $ans(T,k)=m$。答案对 mod 取模。

两颗有标号以 1 为根的树被认为是不同的，当且仅当它们的边集不同。

输入格式:

一行两个整数 n,mod。

输出格式

输出 n 行每行 n 个整数，第 k 行的第 m 个整数表示满足 $ans(T,k)=m$ 的不同的 n 个点以 1 为根的有标号树 T 的数量对 mod 取模的结果。

数据范围

Subtask	$n\le$	分值
1	5	1
2	10	9
3	20	10
4	32	15
5	40	5
6	50	15
7	65	5
8	80	5
9	120	15
10	300	20

对于所有数据：$1\le n\le 300$，$10^8\le mod\le 1.05\times 10^9$，保证 mod 是质数。

解题过程

$ans(T,k)$ 实际上就是对 T 进行长链剖分后，其前 k 长链的链长之和（如果不足 k 条是 n）。

n 个点的地位是相同的，下面我们考虑可以以任何点为根的情况，最后把答案除以 n 即可。

考虑以下计数 n 个点有标号有根树的方式:

- 记 $d_u=\max\limits_{v\in subtree(u)} dis(u,v)$，其中 $dis(u,v)$ 表示 $u\to v$ 简单路径的点数。
- 对于所有点 u，其所有孩子 v 一定满足 $d_v<d_u$。且对于所有点 u 满足 $d_u>1$，其一定存在一个孩子 v 满足 $d_v=d_u-1$。
- 上面两条限制就描述了给每个点认一个父亲需要满足的限制。
- 假设我们现在已经知道 $d_1\sim d_n$ 构成的可重集 S，考虑计算有多少有标号有根树会生成 S，记 s_i 为 S 中值 i 的出现次数。
- 先把 $1\sim n$ 的标号分配给 $d_u=1,2,3...n$ 的每个部分，这里的方案数是 $\binom{n}{s_1,s_2\dots s_n}$。
- 对于最大的 i 满足 $s_i>0$，容易知道一定有 $s_i=1$ 且这对应根节点，记这样最大的 i 为 up。
- 对于 $1\le i<up$，分别考虑 s_i 个 $d_u=i$ 的点，要么认 $d_u=i+1$ 的某个点为父亲，这样就满足了那个点 “至少要有一个孩子的 d 恰好比自己小 1” 的限制；或者随便认一个 $d_u>i+1$ 的点为父亲。
- 预处理所有 $cof_{a,b,c}$ 表示有 a 个 $d_u=i$ 的点，b 个 $d_u=i+1$ 的点，c 个 $d_u>i+1$ 的点，有多少种给 a 个 $d_u=i$ 的点认一个点作为父亲使得所有 b 个 $d_u=i+1$ 个点都至少有一个 $d_u=i$ 的孩子的方案数，以较高复杂度做这一步预处理并不难。
- 当 s 序列确定时，对应的有标号有根树数量就是 $[s_{up}=1]\binom{n}{s_1,s_2\dots s_n}\prod\limits_{i=1}^{up-1} cof_{s_i,s_{i+1},\sum\limits_{k=i+2}^{up} s_k}$。

可以发现对应方案数不为 0 的 s 序列其实一定有 s 不增。而 $\sum\limits_{i=1}^{n} s_i=n$ 且 $\forall 2\le i\le n, s_i\ge s_{i-1}$ 的 s 序列种数其实就是拆分数 $p(n)$，不过由于我们还给出了 $s_{up}=1$ 的限制所以会稍少一些。

于是我们就得到了一种 $O(p(n)n)$ 求 n^{n-1} 的算法，可喜可贺。

当然这个算法的作用不仅于此，因为我们直接枚举了 S 集合，而长链剖分后前 k 长链的链长之和其实还等于 $\sum\limits_{i=1}^{n}\min(s_i,k)$，这样我们就在 $O(p(n)n)$ 的复杂度内解决了原问题。

考虑进一步优化，接下来我们再介绍如何以 $O(n^3)$ 的复杂度计算 n^{n-1}：

- 设 $dp_{i,j,k}$ 为确定了 $s[i,n]$ 且 $s_i=j,\sum\limits_{p=i}^{n} s_p=k$ 的所有 $s[i,n]$ 的权值和，显然 $dp_{i,j,k}\ne 0$ 时必有 $ij\le n$。转移考虑枚举 s_{i-1}，这样 cof 的那一项转移系数也已知了。i 确定时转移复杂度为 $O(n(\frac{n}{i})^2)$，故总复杂度为 $O(n^3)$。

当 $k<s_1$ 时总能找到唯一的 i 使得 $s_i\le k, s_{i-1}>k$，考虑在这里计算贡献，那么 $m=ans(T,k)$ 显然为 $\sum\limits_{j=i}^{n} s_j+k(i-1)$，使用上面的 DP 可以发现需要的信息都已经被记录了，但还需要继续确定 $s[1,i-1]$ 得到那一部分的权值。使用转置原理，或者说倒过来做一遍 DP，计算所有 $g_{i,j,k}$ 为假设已知 $s_i=j,\sum\limits_{p=i}^{n} s_p=k$ 所有不同的 $s[1,i)$ 的权值和（这里被计入的权值显然与 dp 不同），把两部分 DP 的方案数乘起来即可。需要注意的是 $k\ge s_1$ 时这并不成立，但这只会对答案的最后一列产生影响，而我们知道答案应该每一行之和都是 n^{n-1}，因此可以拿 n^{n-1} 减去这一行前面的元素获得最后一列的值。

显然总时间复杂度仍为 $O(n^3)$。

还有一个可能的困难是如何 $O(n^3)$ 预处理所有 $cof_{a,b,c}$，可以发现实际上 $\frac{cof_{a,b,c}}{a!}=[x^a](e^x-1)^b(e^x)^c$，设 $f_{b,c}(x)=(e^x-1)^b(e^x)^c$，那么 $f'_{b,c}(x)=b(e^x-1)^{b-1}(e^x)^{c+1}+c(e^x-1)^b(e^x)^c$，所以 $(n+1)[x^{n+1}]f_{b,c}(x)=[x^n]f'_{b,c}(x)=[x^n](b(e^x-1)^{b-1}(e^x)^{c+1}+c(e^x-1)^b(e^x)^c)=[x^n]bf_{b-1,c+1}(x)+cf_{b,c}(x)$

据此即可 $O(n^3)$ 递推所有 $cof_{a,b,c}$。

部分分设计

0. 真的枚举 n^{n-2} 种有标号有根树然后 $poly(n)$ 算对应 $ans(T,k)$。可以跑到 $n=10$ 左右，获得 10 分。

1. 经典再现，显然两棵树只要同构其所有 $ans(T,k)$ 相同，爆搜无标号有根树然后 $poly(n)$ 算其对应的标号方案数以及所有 k 权值。可以跑到 $n=20$ 左右，获得 20 分。
2. 首先你注意到题面说的其实是长剖，直接把一颗树长剖出来的集合作为状态的话状态数只有 $\sum_{i=1}^{n} p(i)$ 种，然后考虑树可以由根拼上根的各个儿子构造，于是不难得到一个复杂度为 $\sum_{i+j\le n} p(i)p(j)$ 的做法，如果实现的不好可能还要乘 $poly(n)$。可以跑到 $n=40$ 左右，获得 40 分。
3. 上面介绍的 $O(p(n)n)$ 做法，可以跑到 $n=65$ 左右，获得 60 分。
4. 高复杂度的多项式复杂度做法，这个可能的指数就有很多了，如果指数太高可能甚至不如 3。根据不同的指数可以得到不同的分数。
5. 给上面的任意做法多跑一会然后打表，但是不知道模数是什么，所以可能需要交真实值的表或者套数据。但 n 较大的情况很难压进代码长度限制。

杂谈

最后正解 $O(n^3)$ 空间的版本是比较易于实现的，但如果再开大 n 这个做法很容易 MLE，但感觉区分一下空间复杂度也比较无趣，因此选择了比较小的 $n\le 300$ 的限制。

验题时发现存在 $O(n^4)$ 做法进行大力卡常也可以通过，考虑到以和为贵，尽量不要让写正解的人难受的道理决定不管。

出题人不太会在不使用 FFT 的前提下低于 $O(n^3)$ 求本题即使是最后一列的答案，如果有会的同学欢迎来交流。

参考资料

无。