A

做法 1

由于 $\binom{i}{j}$ 可以看作关于 i 的 j 次多项式，所以 $f(i)$ 可以一定能表示成 $\sum_{j=0}^{m} a_j \binom{i}{j}$ 的形式。

考虑

$$
\begin{aligned}
Ans &= \sum_{i=0}^{n} \binom{n}{i} \sum_{j=0}^{\min\{m,i\}} a_j \binom{i}{j} \\
&= \sum_{j=0}^{m} a_j \sum_{i=j}^{n} \binom{n}{i} \binom{i}{j} \\
&= \sum_{j=0}^{m} a_j \sum_{i=j}^{n} \binom{n}{j} \binom{n-j}{i-j} \\
&= \sum_{j=0}^{m} a_j \binom{n}{j} \sum_{i=0}^{n-j} \binom{n-j}{i} \\
&= \sum_{j=0}^{m} a_j \binom{n}{j} 2^{n-j}
\end{aligned}
$$

直接对每个 j 算出 $\binom{i}{j}$ 对应的多项式，然后从高次到低次依次确定 a_j 的取值，复杂度约为 $O(m^2)$，可以得到 90 分。

做法 2

设 $f(x) = \sum_{j=0}^{m} b_j x^j$，我们推一波式子：

$$
\begin{aligned}
& \sum_{i=0}^{n} \binom{n}{i} f(i) \\
&= \sum_{k=0}^{m} b_k \sum_{i=0}^{n} \binom{n}{i} i^k \\
&= \sum_{k=0}^{m} b_k \sum_{i=0}^{n} \binom{n}{i} \sum_{j=0}^{\min\{i,k\}} \begin{Bmatrix} k \\ j \end{Bmatrix} \binom{i}{j} j! \\
&= \sum_{k=0}^{m} b_k \sum_{j=0}^{k} \begin{Bmatrix} k \\ j \end{Bmatrix} j! \sum_{i=j}^{n} \binom{n}{i} \binom{i}{j} \\
&= \sum_{k=0}^{m} b_k \sum_{j=0}^{k} \begin{Bmatrix} k \\ j \end{Bmatrix} j! \sum_{i=j}^{n} \binom{n}{j} \binom{n-j}{i-j} \\
&= \sum_{k=0}^{m} b_k \sum_{j=0}^{k} \begin{Bmatrix} k \\ j \end{Bmatrix} j! \binom{n}{j} 2^{n-j} \\
&= \sum_{j=0}^{m} \binom{n}{j} 2^{n-j} j! \sum_{k=j}^{m} \begin{Bmatrix} k \\ j \end{Bmatrix} b_k \\
&= \sum_{j=0}^{m} \binom{n}{j} 2^{n-j} j! \sum_{k=0}^{m} \begin{Bmatrix} k \\ j \end{Bmatrix} b_k \\
&= \sum_{j=0}^{m} \binom{n}{j} 2^{n-j} j! \sum_{k=0}^{m} b_k \sum_{t=0}^{j} (-1)^{j-t} \binom{j}{t} t^k \frac{1}{j!} \\
&= \sum_{j=0}^{m} \binom{n}{j} 2^{n-j} \sum_{t=0}^{j} (-1)^{j-t} \binom{j}{t} \sum_{k=0}^{m} b_k t^k
\end{aligned}
$$

可以先做一次多项式多点求值，对每个 t 算出 $\sum_{k=0}^{m} b_k t^k$；剩下的部分是个卷积，一次 NTT 即可得到答案。

B

这个题改编自[Chef and Elephant Tree]。

做法应该很多，这里介绍 std 的实现。

不妨考虑每条边的贡献；问题转化成对于每条边，统计能让这条边在虚树上的选叶子方案数。

假设现在要统计的是 x 到 f_x 的边，设点 x 的子树内总共有 cnt_x 个编号在 $[L,R]$ 中的叶子，设 tot 为 $[L,R]$ 中的叶子总数量。

则这条边的贡献为

$$\sum_{i=1}^{k-1} cnt_x^i(tot-cnt_x)^{k-i}$$

把这个式子用二项式定理展开成 $\sum_{i=1}^{k} c_i cnt_x^i$ 的形式。考虑 cnt_x^i 的组合意义，相当于是从 x 的子树内的叶子中选出 i 个的方案数。我们可以先枚举这被选出的 i 个叶子 $l_1,l_2,\cdots l_i$，然后统计子树包含了它们的 x 的个数。注意到满足条件的 x 就是选出的点的公共祖先，所以满足条件的 x 的数量就是选出的点的最近公共祖先的深度。

$$\sum_x cnt_x^i = \sum_{l_1\in[L,R]}\sum_{l_2\in[L,R]}\cdots\sum_{l_i\in[L,R]} dep\left(lca\left(l_1,l_2,\cdots l_i\right)\right)$$

这道题中树的性质相当于是说，存在一种 dfs 序，满足每个点在 dfs 序中出现的位置与它的编号相同。

考虑一种与欧拉序求 lca 类似的转化：考虑叶子结点序列（既是按照编号从小到大，也是按照 dfs 序从小到大）$s_1,s_2,\cdots s_m$，令 val_i 为从 s_{i-1} 走到 s_i 的路径上，点的深度的最小值。则 $dep\left(lca\left(s_x,s_y\right)\right)=\min_{x<i\le y}\{val_i\}(x<y)$，$dep\left(lca\left(l_1,l_2,\cdots l_i\right)\right)=\min_{\min\{l_j\}<s_x\le\max\{l_j\}}\{val_x\}$。

令 L' 为最小的满足 $s_{L'}\ge L$ 的位置，R' 为最大的满足 $s_{R'}\le R$ 的位置。

枚举 $\min\{l_j\}$ 和 $\max\{l_j\}$ 以及它们在 $\{l_1,l_2,\cdots l_i\}$ 中的出现次数，得到：

$$\sum_x cnt_x^i=\sum_{l=L'}^{R'}\sum_{r=l+1}^{R'}\sum_{x=1}^{i-1}\sum_{y=1}^{i-x}\binom{i}{x+y}\binom{x+y}{x}(r-l-1)^{i-x-y}\min_{l+1\le j\le r}\{val_j\}$$

至此我们得到了一个优秀的多项式复杂度算法。

考虑优化，我们的瓶颈主要在于 $\sum_{l=L'}^{R'}\sum_{r=l+1}^{R'}$ 这个枚举。只要能快速求出 $g_t=\sum_{l=L'}^{R'}\sum_{r=l+1}^{R'}(r-l-1)^t\min_{l+1\le j\le r}\{val_j\}$，就能在 $O(k^2)$ 的时间里得到答案。

用二项式定理拆开 $(r-l-1)^t$ 拆开：

$$\begin{aligned}&\sum_{k=0}^{t}\binom{t}{k}\sum_{l=L'}^{R'}\sum_{r=l+1}^{R'}\min_{l+1\le j\le r}\{val_j\}\cdot(r-1)^k(-l)^{t-k}\\=&\sum_{k=0}^{t}\binom{t}{k}(-1)^{t-k}\sum_{l=L'}^{R'}\sum_{r=l+1}^{R'}\min_{l+1\le j\le r}\{val_j\}\cdot(r-1)^k l^{t-k}\end{aligned}$$

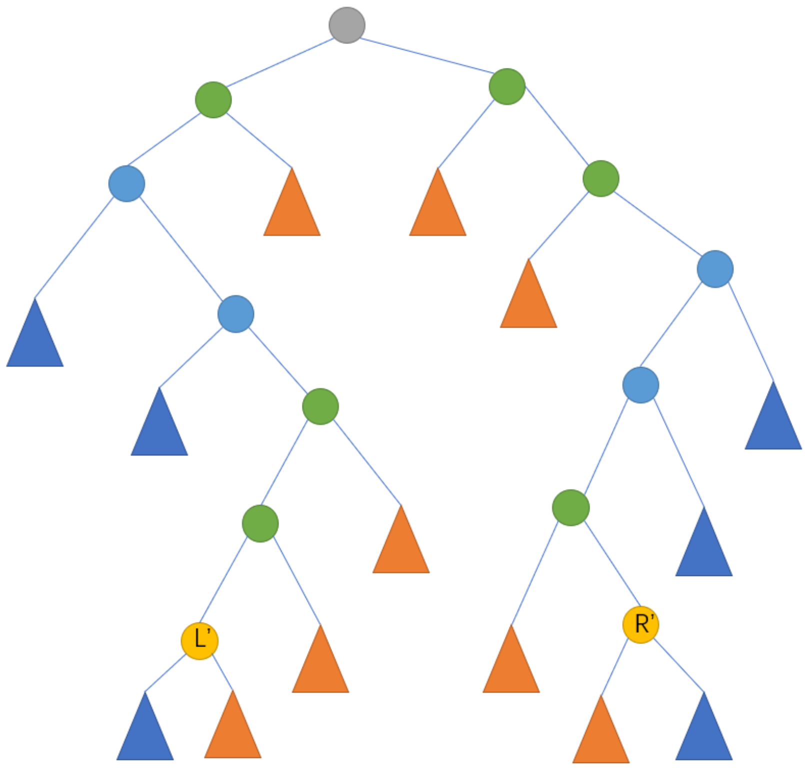
考虑每个 val_j 的贡献。设 pre_j 为 $\max_{i<j,val_i<val_j}\{i\}$，$suf_j$ 为 $\min_{i>j,val_i<val_j}\{i\}$，则原式等于

$$\sum_{k=0}^{t}\binom{t}{k}(-1)^{t-k}\sum_j val_j\sum_{l=\max\{L',pre_j+1\}-1}^{j-1}\sum_{r=j}^{\min\{R',suf_j-1\}}(r-1)^k l^{t-k}$$

设 $F(k,n)$ 为 $\sum_{i=0}^{n} i^k$，则原式等于

$$\sum_{k=0}^{t}\binom{t}{k}(-1)^{t-k}\sum_j val_j\left(F\left(t-k,j-1\right)-F\left(t-k,\max\{L'-2,pre_j-1\}\right)\right)\cdot\left(F\left(k,\min\{R'-1,suf_j-2\}\right)-F\left(k,j-2\right)\right)$$

考虑 val 的笛卡尔树（每个点的 val 不大于儿子的 val，且中序遍历为原序列）：



注意到：

- 橘色的子树内的点满足 $L' \le pre_j \le R'$
- 左边的绿色点和黄色点（即 L' ）一定满足 $pre_j \le L', suf_j \le R'$
- 右边的绿色点和黄色点（即 R' ）一定满足 $pre_j \ge L', suf_j \ge R'$
- 灰色点为 L', R' 的 LCA，它满足 $pre_j \le L', suf_j \ge R'$
- 绿色点、黄色点、灰色点和橘色子树中的点，就是区间 $[L', R']$ 中的点

我们对每种颜色的点分别统计它们的贡献即可。具体地，橘色的子树就是 L' 到 LCA 的路径上的右子树和 R' 到 LCA 路径上的左子树，左边的绿色点是"是父亲的左儿子"的点的父亲，右边的绿色点是右儿子的父亲，用树上前缀和分别维护它们的信息即可。

时间复杂度大约是 $O(n \log n + (n+q)k^2)$。具体实现细节可以参考标程。

C

~~薇可以请各位打比赛的 dalao 吃饭，但是请不要喷她 /kel~~

我只会近似算法。

设 $f(a)$ 为平面 $z = a$ 与题目中给出图形的交的面积，则要求的相当于

$$\int_{lz}^{rz} f(z)\,\mathrm{d}z$$

尽管 $f(z)$ 的表达式看上去几乎不可能求出来，但是我们可以考虑近似积分。我们只需要求出采样点的函数值即可，也就是求平面上的圆的面积并。

这是个经典问题。可以参考这个博客中的介绍。

感受一下 $f(z)$ 的图像应该是一个光滑的曲线，所以 Simpson's Rule 的效果应该会比 Riemann sum 好得多。而实际上在保证同样精度的情况下，Simpson's Rule 的效率的确远高于 Riemann sum。此外，如果将 z 根据平面切到的球的集合分段，对每一段分别计算，也能让计算次数除以掉一个常数。标程的实现就是对 z 分段然后对每一段做自适应辛普森积分。