T1

不难发现在 S 后面加个 #，T 后面加个 $，答案就是从 S,T 里分别选一个字符使得他们不一样的方案数。复杂度 $O(n)$。

T2

我们先考虑 a_i 全部相等的时候如何处理。~~熟练的选手可以直接看出是个多项式快速幂~~。

考虑倍增 dp，即 $g(i,j)$ 表示当前 dp 了前 2^i 种糖果，占了 j 个位置的方案数。转移显然：

$$g(i,j)=\sum_{k=0}^{j} g(i-1,k)g(i-1,j-k)\binom{j}{k}$$

其中，$g(0,0\cdots a_i)=1$。

有了这个倍增的 dp 数组，我们就可以合并出 n 种糖果的答案了。这部分复杂度 $O(m^2\log n)$。

如果 a_i 不全相等，注意到 $\geq m$ 的 a_i 都可以直接变成 m，因此对于总共 m 种 a_i 分别做一遍上面的倍增即可，最后再把所有 dp 数组背包合并起来。

总复杂度 $O(m^3\log n)$。

T3

考虑一个性质，令 $h(i)$ 表示 S 长度为 i 的后缀和 T 的编辑距离，则 $i-h(i)$ 单调递增，并且有 $|i-h(i)|\leq |T|$。

首先证明单调递增，每次 i 增加 1 的时候，$h(i)$ 至多增加 1，这个很显然，因此单调递增。

其次 $-|T|\leq i-h(i)\leq |T|$，这个也很显然。

于是我们就可以进行分段函数 dp 了。即 $f(i,j,k)$ 表示考虑了 S 长度为 i 的前缀，T 长度为 j 的前缀，最大的 x 使得 $x-h(x)\leq k$。

转移方程仔细思考一下即可，也就是魔改原来编辑距离的 dp。记 g 为暴力编辑距离的 dp：

$g(i,j)=\min\{g(i-1,j)+1,g(i,j-1)+1,g(i-1,j-1)+[S_i\neq T_j]\}$。

于是 $f(i,j,k)=\min\{f(i-1,j,k)+1,f(i,j-1,k-1),f(i-1,j-1,k-[S_i=T_j])+1\}$。

初始值需要注意一下，$f(i,j,-|T|-1\cdots-j-1)=-1,f(0,0,-|T|-1\cdots-1)=-1,f(0,0,0\cdots|T|)=0$，剩下全是 $+\infty$。

这个 dp 的复杂度为 $O(|S||T|^2)$，查询 $[l,r]$ 的话直接找最小的 k 使得 $f(r,|T|,k)\ge r-l+1$ 即可，答案为 $r-l+1-k$，因此单次询问的复杂度可以做到 $O(|T|)$ 或 $O(\log|T|)$。

T4

显然是按位确定的套路，我们考虑确定了前 $i-1$ 个点的值，来确定第 i 个点的值。

首先若前 $i-1$ 个点构成的前缀已经严格比给定的前缀小了，那么当前显然就没有任何限制了，否则当前数字会有一个上界。

假设我们暴力枚举当前数字是 k，于是问题转化为了钦定前 i 个点的数字，求此时合法的方案数。

不难发现现在就是若干个限制，每条限制都是限制子树内的所有数字要大于某个数。

设 s_i 表示被它限制的数的数量（注意不是子树大小），t_i 表示限制，我们把所有数字按照 t_i 从大到小排序，那么方案数显然是：

$$\prod\binom{n-t_i-s_1-\cdots-s_{i-1}-i+1}{s_i}$$

于是我们就获得了一个 $O(n^3)$ 的做法。

优化成 $O(n^2)$ 也十分简单，考虑枚举当前数字的那部分，直接 two pointers 扫描更新乘积即可。