1、video

暴力求个阶乘和逆元就好了

2、Chance分析

假设已经知道第i个袋子中取到一个第一个数字为1的球的概率为pi

dp[i][j] 表示前i个袋子里取到了j个第一个数字为1的球的概率;

dp[i][j] = dp[i - 1][j - 1] \* pi + dp[i - 1][j] \* (1 - pi);

每个袋子里有多少个球是开头数字为1的随便算一算就可以算出来;

3、plutotree

30% 把图建建出来floyd

60% 把floyd换成spfa或者dijkstra

100%

树形dp

首先答案可能有三种情况：

1.不走叶子到根的边， 那就是一个简单的树上的问题， 两点之间有且仅有一条路径，树上倍增一下就可以把两个问题都解决， 或者闲着蛋疼的可以用树剖从u走到叶子，从叶子到根，从根到v （这种情况要把u、v调换再做一次）

2.从u走到叶子，从叶子到根，从根到另一个叶子节点，再从叶子节点到v

对于第二种、三种情况， 其实关键在于求出每个点到离他最近的叶子的路径，

3这样的路径有两种可能，一种是向下下走到叶子， 一种是先向上（有可能经过根也可能不经过）走，最终到达叶子

首先树形dp出每个节点向下到离它最近的叶子节点的距离，

并同时记录这个路径上的最大值，设为down[i], (down[i]为二元组，<路径权值和， 路径上最大权值>) down[i] 可以从 i号节点的所有儿子转移过来， 转移方程很显然

有了down[i], 就可以dp出我们需要的每个点到最近的叶子节点的路径了，将这个记为dp[i], dp[i]同样是一个二元组，和down[i]一样。 首先dp[1]我们是知道的，就是从根直接走到某一个叶子节点， 其余的dp[i]初值都是down[i], dp[i]可以从i号节点的父亲转移过来，所以这是一个自上而下的dp

考虑到可能爆栈的情况， 建议使用bfs处理树形dp

具体状态如何转移可以参考std