Solution

A

方法一

$n \leq 5$ 时可以手玩。

方法二

考虑状压 dp，设 $dp[msk]$ 表示当前剩余的数的状态是 msk 时先手的胜负情况，转移考虑先手第一步选啥即可。

时间复杂度 $O(2^n n)$，期望得分 70 分。

方法三

考虑如果初始集合为空则后手胜。

如果初始集合非空的话，则 1 一定在其中。如果先手第一步选除去 1 之外的某个数能够胜利的话则选这个数，否则就选 1，将几乎同样的局面丢给后手，依然能获得胜利。故先手必胜。

实际上就只要判断 m 是不是 0 即可，期望得分 100 分。

B

方法一

设 f_{msk} 为假设当前墙面状态为 msk 时到最终形态的期望代价，可以列出 2^{nm} 个方程，然后高斯消元解一下方程即可。

时间复杂度 $O(2^{3nm})$，期望得分 60 分左右。

方法二

考虑如何减少状态的数量，实际上就是把所有状态分成若干个等价类，使得每个等价类里面的每个元素的每种方式转移到的点都在同一个等价类里。如果一个位置右下角的某个位置跟最终的情况不同，那么这个位置之后一定会被修改，于是它当前的值就不重要了。考虑 $p_{i,j}$ 为 (i,j) 右下角的所有位置是不是已经变成了最终的样子，那么所有 $p_{i,j}$ 为 1 的状态一定是成一个阶梯堆在右下角，所以总共的状态数为 $\binom{n+m}{n}$。考虑对于一次修改 (x,y,c)，对于所有不在 (x,y) 左上角的点 (i,j)，$p_{i,j}$ 不变，对于所有在 (x,y) 左上角的点 (i,j)，需要先判断 (i,j) 到 (x,y) 这个矩形的元素是否均为 c，然后在判断 $p_{x+1,j}$ 和 $p_{i,y+1}$ 是否均为 1 即可。总之它可以唯一转移到另一个状态，所以按照这种方式计算出转移方程，高斯消元解一下即可。

时间复杂度 $O(\binom{n+m}{n}^3)$，期望得分 100 分。

C

方法一

枚举 r 个位置然后暴力模拟。

时间复杂度 $O(\binom{nm}{r} * nmr)$，期望得分 20 分。

方法二

考虑 $E(X)=\sum_{i=1} P(X\geq i)$，所以可以对于每个 i 计算 i 步之后区分不出它的概率，可以考虑对于每一步，维护每个起点收到的信息的等价类，然后能区分当且仅当选中的 r 个初始起点在不同的等价类中，这个的概率可以用 dp 算出。只要能在多项式时间复杂度内维护等价类和计算概率，应该就能过 subtask2，期望得分 40 分。

方法三

可以考虑用哈希等方式维护等价类，一共 nm 步，每步维护的时间复杂度为 $O(nm)$，总时间复杂度 $O(n^2m^2)$。但是有树的地方少，所以可以动态构建等价类，即对于每一步，将当前这一步有树的起点从其所在的等价类中单独剥离出来，这样总时间复杂度就是 $O(nmk)$ 了。

考虑朴素的 dp 是这样的：设 $dp[i][j]$ 为考虑前 i 个等价类，选了 j 个数，使得它们在不同等价类里的方案数。这样每次计算的复杂度为 $O(nmr)$，总时间复杂度为 $O(n^2m^2r)$。考虑优化，设当前等价类的大小分别为 $a_1, a_2 \dots a_p$，那么方案数即为 $[x^r]\prod_{i=1}^{p}(1+a_ix)$。故可以实时维护后面的多项式，考虑等价类只会做 $O(nm)$ 次分裂，每次一个大小为 a 的等价类分裂成 b 和 c 时，对这个多项式进行的操作就是除以 $(1+ax)$ 然后乘上 $(1+bx)(1+cx)$，这些都可以在 $O(r)$ 的时间内完成。故总时间复杂度为 $O(nmr)$。

总时间复杂度 $O(nmk+nmr)$，期望得分 100 分。

D

方法一

设 $dp[msk][j]$ 表示当前还剩 msk 的盒子没开，当前已开的盒子下的最大价值的宝物价值为 j，在最优策略下的期望收益。转移即考虑下一步开什么箱子，然后枚举箱子里的状态即可。

时间复杂度 $O(2^n(\sum_i k_i)^2)$，期望得分 20 分。

方法二

设第 i 个盒子里的宝物价值为 v_i，打开这个盒子的代价为 c_i，假设当前打开的盒子最大价值为 v，你如果下一步打开第 i 个盒子，最大的价值就会变为 $max(v,v_i)$，如果期望收益，也就是 $E(max(v,v_i)-v)$，小于 c_i，那么我们现在不会打开这个盒子，并且随着 v 越来越大，之后的期望收益也会越来越小，所以我们之后永远都不会打开这个盒子了。故设 σ_i 为一个 threshold，使得当 $v>\sigma_i$ 时就不会开第 i 个盒子了。设 x^+ 表示 $max(x,0)$，则 σ_i 满足 $E((v_i-\sigma_i)^+)=c_i$。σ_i 直接扫一遍就可以算出。通过之前的分析可以得出，终止开盒的条件就是当前的最大价值 v 大于所有未被开的盒的 σ_i 即终止。那么可以猜测最优的开盒的顺序应该是按照 σ_i 从大往小开，直到 v 大于剩余的所有 σ 为止，事实上也可通过归纳加上反证的方式证明这个策略是最优的，证明细节这里略去。

于是就可以改一下方法一里的 dp，按照 σ_i 从大到小排序，设 $dp[i][j]$ 表示已经开了前 i 个盒，当前最大价值为 j 时，按照前述的策略的期望收益，时间复杂度 $O((\sum_i k_i)^2)$，期望得分 50 分。

方法三

优化一下方法二，假设 σ_i 已经按照从大到小排序好了，枚举最终取的 v_i，如果 $v_i>\sigma_i$，则开了前 i 个盒就结束了，且 $v_1, v_2 \dots, v_{i-1}<\sigma_i$。若 $v_i<\sigma_i$，则后面还会再开几个盒子，一直开到第 j 个。则只需要求 $v_1 \dots v_{i-1}, v_{i+1} \dots v_j$ 均 $<v_i$ 的概率即可。于是我们要处理若干个形如这样的 query：求 $v_1 \dots v_i$ 均 $<x$ 的概率。这个可以按照 x 从小往大处理，及维护一个单点修改，前缀乘积的数据结构即可。

时间复杂度 $O((\sum_i k_i)logn)$，期望得分 100 分。

方法四

上述做法太过复杂，考虑直接从分析最终的收益值的角度出发得到一个更简单的做法。

设 A_i 表示你最终有没有选第 i 个箱子里的宝物，I_i 表示你有没有开第 i 个箱子。A_i, I_i 的取值范围均为 $0/1$。

一种策略即是一个 $v_1 \dots v_n$ 的取值到 $A_{1\dots n}, I_{1\dots n}$ 的一个映射。

考虑最终的收益为 $\sum_{i=1}^{n} A_i v_i - I_i c_i$。分析这个收益(将其变成只与 v_i 相关的式子)。

$$
\begin{aligned}
& E_{v_1\dots v_n}(\sum_{i=1}^{n} A_i v_i - I_i c_i) \\
=\ & E_{v_1\dots v_n}(\sum_{i=1}^{n} A_i v_i - I_i E_{v_i'}((v_i' - \sigma_i)^+)) \\
=\ & E_{v_1\dots v_n}(\sum_{i=1}^{n} A_i v_i) - E_{v_1\dots v_n}(\sum_{i=1}^{n} I_i E_{v_i'}((v_i' - \sigma_i)^+)) \\
=\ & E_{v_1\dots v_n}(\sum_{i=1}^{n} A_i v_i) - \sum_{i=1}^{n} E_{v_1\dots v_n}(I_i E_{v_i'}((v_i' - \sigma_i)^+)) \\
=\ & E_{v_1\dots v_n}(\sum_{i=1}^{n} A_i v_i) - \sum_{i=1}^{n} E_{v_1\dots v_{i-1}, v_{i+1}\dots v_n}(E_{v_i}(I_i E_{v_i'}((v_i' - \sigma_i)^+))) \\
=\ & E_{v_1\dots v_n}(\sum_{i=1}^{n} A_i v_i) - \sum_{i=1}^{n} E_{v_1\dots v_{i-1}, v_{i+1}\dots v_n}(I_i E_{v_i}(E_{v_i'}((v_i' - \sigma_i)^+))) \\
=\ & E_{v_1\dots v_n}(\sum_{i=1}^{n} A_i v_i) - \sum_{i=1}^{n} E_{v_1\dots v_{i-1}, v_{i+1}\dots v_n}(I_i E_{v_i}((v_i - \sigma_i)^+)) \\
=\ & E_{v_1\dots v_n}(\sum_{i=1}^{n} A_i v_i) - \sum_{i=1}^{n} E_{v_1\dots v_n}(I_i (v_i - \sigma_i)^+) \\
=\ & E_{v_1\dots v_n}(\sum_{i=1}^{n} A_i v_i - I_i (v_i - \sigma_i)^+) \\
\le\ & E_{v_1\dots v_n}(\sum_{i=1}^{n} A_i (v_i - (v_i - \sigma_i)^+)) \\
=\ & E_{v_1\dots v_n}(\sum_{i=1}^{n} A_i * min(v_i, \sigma_i)) \\
\le\ & E_{v_1\dots v_n}(max_{i=1}^{n}(min(v_i, \sigma_i)))
\end{aligned}
$$

这样我们就得出了期望收益的一个上界，考虑当两个不等号都取等时这个上界就能达到。分析一下两个不等号何时取等，第一个不等号当 $(I_i - A_i)(v_i - \sigma_i)^+ = 0$ 时取等，考虑如果我们按照之前说的那个策略，当我们开了一个盒子 i 但没取它时，$v_i < \sigma_i$，其它情况下 $I_i - A_i = 0$，故第一个不等号取等。考虑第二个不等号，它取等时当且仅当最终取了 $min(v_i, \sigma_i)$ 最大的那个盒子里的宝物，可以分析出这个策略下我们一定会取 $min(v_i, \sigma_i)$ 最大的那个盒子里的宝物，故两个不等号均取等，这个策略的期望收益达到最大值，为 $E_{v_1\dots v_n}(max_{i=1}^{n}(min(v_i, \sigma_i)))$。于是只要求这个式子即可。可以对于每一个 i，先求 $min(v_i, \sigma_i)$ 的分布，然后再从大往小枚举 max 的取值，实时维护每个位置 $<$ 当前取值的概率即可。

时间复杂度 $O((\sum_i k_i) logn)$，期望得分 100 分。