树桩 (wood)

先看怎么求 $f(1,n)$。假设求出了从格子 $2i-1$ 发射出的普通豌豆子弹最终会变成什么类型的豌豆子弹，然后统计出其中有 x 个位置发射出的普通豌豆子弹最终会变成火焰豌豆子弹/寒冰豌豆子弹，剩下 $n-x$ 个位置发射出的普通豌豆子弹最终还是普通豌豆子弹。那么最优方案肯定是，将 $a_1,\ldots,a_n$ 中每秒射出豌豆最多的 x 个豌豆射手种在那 x 个位置，将剩下的 $n-x$ 个豌豆射手种在剩下的 $n-x$ 个位置。假设 $a_1 \ge \cdots \ge a_n$，那么 $f(1,n)=2\sum_{i=1}^{x} a_i+\sum_{i=x+1}^{n} a_i$。

做法一

枚举 l,r。从 r 往 l 扫描并统计出 x。时间复杂度 $O(n^3)$。

做法二

做法一中，我们可以先固定 r，然后将 l 从 r 往 1 扫描。过程中边移动 l，边统计 x，边将 $f(l,r)$ 计入答案。时间复杂度 $O(n^2)$。

做法三

更低的复杂度不允许我们一一枚举每对 l,r。我们需要对 x 到底是什么做进一步的分析，这里还是先考虑 $l=1,r=n$ 的情况。

注意到，如果有连续两个树桩类型一样，比如都是火炬树桩，那么经过这两个树桩后的豌豆子弹一定是火焰豌豆子弹。

那么找到最靠右的两个连续的类型相同的树桩 $p,p+1$，不妨设为火炬树桩。那么 $p+2,p+3,\ldots,n$ 号树桩依次为寒冰树桩、火炬树桩、……。可以发现：

- 如果 $p+2,p+3,\ldots,n$ 中共有奇数个树桩。那么从树桩 $p+1$ 左侧发射的普通豌豆子弹最终都会变为普通豌豆子弹，从 $p+1,p+2$ 之间发射的普通豌豆子弹最终会变为寒冰豌豆子弹，从 $p+2,p+3$ 之间发射的普通豌豆子弹最终会变为普通豌豆子弹……

  即，$x=\frac{n-p}{2}$。
- 如果 $p+2,p+3,\ldots,n$ 中共有偶数个树桩。那么从树桩 $p+1$ 左侧发射的普通豌豆子弹最终都会变为火焰豌豆子弹，从 $p+1,p+2$ 之间发射的普通豌豆子弹最终会变为普通豌豆子弹，从 $p+2,p+3$ 之间发射的普通豌豆子弹最终会变为火焰豌豆子弹……

  即，$x=\frac{n+p+1}{2}$。

当 r 固定时，$p,p+1$ 这两个树桩的位置与 l 的选择无关。而 x 随着 l 变化要么一直不变、要么一直加 1。无论哪种情况，都可以快速计算所有 l 的答案和。

时间复杂度 $O(n)$，可以通过此题。

大蒜 (garlic)

做法一

对每个 i 暴力计算答案，时间复杂度 $O(n^2m)$，可以获得 20 分。

做法二

设 f_{i,j,i_0} 表示从 (i,j) 出发最终是否能到达第 i_0 行，然后进行 DP，时间复杂度 $O(n^2m)$，可以获得 20 分。

注意到若 (i,j) 处没有大蒜，那么 $f_{i,j,*}=f_{i,j-1,*}$，所以在有大蒜时才需要计算新的 f。时间复杂度 $O(n(m+\text{大蒜数量}))$，可以额外获得 20 分。

用 `bitset` 优化最后一维，时间复杂度 $O(\frac{n^2m}{w})$，可以额外获得 20 分。

做法三

考虑从哪些行出发最终能到达第 i_0 行。发现，如果从第 $j-1$ 列出发最终能到 i_0 行的行是区间 $[l,r]$，那么：

- 若 $l=r$。
  - 若 (l,j) 处没有大蒜。那么 (l,j) 最后能到 i_0 行。而若 $(l,j-1)$ 有大蒜，$(l,j-1)$ 最终也能到 i_0 行。对 $(l,j+1)$ 同理。

    所以从第 j 列出发最终能到 i_0 行的行是 $[l-1,l+1]$ 的一个子区间。
  - 若 (l,j) 处有大蒜。那么 (l,j)最后不能到i_0行。此时(l,j-1)和(l,j+1)最终能不能到i_0$ 行同样取决于它们各自的位置上有没有大蒜。

    但根据题目的限制，这两个位置不可能同时有大蒜。所以从第 j 列出发最终能到 i_0 行的行只有至多一行。
- 若 $l<r$。对每个 $l\le i\le r$，(i,j) 最终一定能到 i_0 行。而 $(l-1,j)$ 和 $(r+1,j)$ 最终能不能到 i_0 行取决于它们的位置有没有大蒜。

所以从第 j 列出发最终能到 i_0 行的行仍然保持一个区间。那么我们按照上面的讨论，从最左边的列推到最右边的列，过程中维护这个区间即可。单组数据时间复杂度 $O(nm)$，可以通过此题。

猫尾草 (cattail)

先解决子问题：每次给定 x，所有尖刺都向 x 移动一步。

首先注意到，任意时刻包含尖刺的节点一定是树上的一个连通块，记为 S。

现在考虑让所有尖刺都向 x 移动一步。以 x 为根建树，把 S 的深度最小的节点作为根，那么每个尖刺都向 x 移动一步后，S 会删去其所有的叶子，并将其根的父亲加入 S。那么 $|S|$ 应该恰好减少 $S\text{的叶子数}-1$。如果将 S 看为无根的，$|S|$ 就会至少减少 $S\text{的叶子数}-2$。

又注意到，维护好 S 的所有叶子，其实也就维护好了 S 的形态。那么做法就呼之欲出了：将 $|S|$ 作为势能，维护 S 的叶子集合。每次操作暴力枚举每个叶子并将其 S 上的父节点更新为叶子，同时需要新增加一个叶子（S 往 x 路径上的第一个点）。

对于原问题，每次我们可以先求出目标点 x 和 S 的距离 d，然后新的叶子只可能是原来的叶子向 x 移动 d 步的结果。仍然用类似的势能分析即可。

时间复杂度 $O(n\log n)$，可以通过此题。

Bonus

对初始时的每个尖刺，求出最后它落在哪里。

头脑风暴 (brain)

在 时间-坐标 平面上，将第 i 个僵尸对应到从 $(t_i,0)$ 出发、斜率为 v_i 的射线，将第 i 个头脑风暴对应于线段 $[(T_i,x_i-r_i),(T_i,x_i+r_i)]$。那么题目就转化为，对每条射线求，是否存在一条线段被该射线穿过。由于所有射线都从 x 轴出发，这里可以把射线改为直线。

先考虑射线数量不超过 1000 时怎么做。考虑对 x 维度做扫描线。每次出现一个头脑风暴时，我们需要找到所有当前 y 坐标在某个 $[l, r]$ 内的直线（并将它们删除）。如果维护好了所有直线按 y 坐标排序后的序列，我们只需要在这个序列上二分即可。而这个序列也是可以维护的：注意到这个序列会改变当且仅当两条直线相交了，而且此时也只是序列中相邻两个元素交换一下位置。而 n 条直线间只有至多 n^2 个交点，那么暴力找出这些相交事件，并在扫描线过程中处理它们带来的序列上的交换即可。时间复杂度 $O((n^2 + m) \log n)$。

对 $n = 10^5$ 的情况，考虑将所有射线每 B 个分为一组，每组都按上面的做法来做。这样的时间复杂度为 $O(\frac{n}{B}(B^2 + m) \log B)$，视 n, m 同阶并取 $B = \sqrt{n}$，得到 $O(n\sqrt{n} \log n)$。