T1

搜索剪枝，预处理出在分别不考虑魔力/回合数的情况下，从当前局面到最后需要最少的回合数/魔力，如果劣于最优解则剪枝。

T2

一个子树是bst当且仅当子树的中序遍历序列的点权单调不降，而每个子树的中序遍历序列是整个树中序遍历序列的一段区间。

求出中序遍历序列 $a_1...a_n$，那么设 u 的子树对应 $[l,r]$，那么 u 是bst当且仅当 $\forall i\in[l,r),a_i\le a_{i+1}$。

查询某个点是否合法只需要用树状数组维护区间不满足 $a_i\le a_{i+1}$ 的个数。

每次修改点权只会影响该点到根路径的结果，而且合法的点一定是连续的一段。

树上倍增即可。

T3

问题相当于是 k 维背包，每个物品在某些维中有体积。

考虑数位dp，从高位到低位，在每一位中依次考虑每个物品体积的当前二进制位是 0 还是 1。

$f_{i,j,a,b,c,d}$ 表示当前考虑到 2^i 位下第 j 个物品，当前剩余空间为 $a*2^i,b*2^i,c*2^i,d*2^i$ 所能得到的最大值。

通过分析可以发现，每一维只与 2^{k-1} 个物品相关，这个背包容量每维的容量不需要超过 15，因为否则剩余的所有子集加起来也不够。

进一步分析，以第一维相关的物品为例子，有 $(1000,1001,1010,1011,1100,1101,1110,1111)$ 八个物品与其相关。

而将其两两分组 $(1000,1111),(1001,1110),(1010,1101),(1100,1011)$，最优方案中只有至多一组中两个物品都选了。

于是背包容量可以降为 9，可以通过。

T4

首先考虑对于没有修改的情况计算答案. 对一个点 u, 考虑其所有儿子 $v_1,v_2,\ldots,v_k$. 由于同构保持父子关系, 于是按照同构关系将他们划分成等价类 $P_1,P_2,\ldots,P_m$ 后. 对点 u 的儿子进行置换的方案数为

$$f(u)=\prod_{i=1}^m|P_i|!,$$

计算一个点 x 对应的答案, 只需要对其子树中的每个点 u 计算 $f(u)$, 然后用乘法原理全部乘起来:

$$\mathrm{ans}_x=\prod_{u\in\mathrm{subtree}(x)}f(u)$$

接下来注意到同构的树大小相同, 于是新加入一个点的时候, 如果它的某个祖先节点 u 的同构关系改变, 那么 fa_u 还有一个子树大小和 u 相同的儿子, 于是 $\mathrm{siz}_{\mathrm{fa}_u}\ge 2\mathrm{siz}_u$, 因此发生变化的同构关系只有 $\mathcal{O}(\log)$ 个. 如果我们能找出这些位置, 用线段树维护答案就做完了.

接下来我们用树哈希来判断同构. 考虑如下的哈希函数:

$$h_u = G(\mathrm{dep}_u)\left(1 + \prod_{v \in \mathrm{son}_u} h_v\right)$$

其中 G 是一个随机函数. 这个哈希函数是容易用 ddp 维护的. 这样在新加入点的时候, 在这个点到根的路径上每次二分第一个 $\mathrm{siz}_{\mathrm{fa}_u} \geq 2\mathrm{siz}_{\mathrm{u}}$ 的位置并查询哈希函数即可. 总复杂度 $\mathcal{O}(n \log^2 n)$.

P.S. 一些更简单更容易维护的哈希函数被卡了, 比如:

$$h_u = \sum_{v \in \mathrm{subtree}(x)} \mathrm{dep}_v^k,$$

$$h_u = G(\mathrm{dep}_u)\left(1 + \sum_{v \in \mathrm{son}_u} h_v\right),$$

可以被第二个样例卡掉.