T1 树上期望，排序+前缀和

由于原图是个森林，所以若(ru)果(guo)超时空链路对每个点能切断的路径>=1，就带便能切断任意路径。否则不能切断任何路径。

设 dp[i]为第 i 个点及其子树的最大期望。

枚举每个节点作为根，

枚举每个节点，在有 cnt 棵子树被保留的情况下每个被保留节点对父节点的期望贡献。

dp[i]×p[i]+dp[i]×(1-p[i])/cnt

变形一下

dp[i]×(p[i]+(1-p[i])/cnt)

发现在 cnt 相同的情况下，dp[i]的相对大小和其对答案的贡献的相对大小是一样的，所以对 dp[i]从大到小排序，每次选最大的 cnt 个求和，乘上对应的概率系数，所有的方案取 max 即为 father 的答案，这里用前缀和优化，就可以在排序后 O(n)计算。

每个根的复杂度为 O（nlogn）,再算上枚举根整体复杂度为 O（n^2logn）

T2 分块+线段树

有的同学可能会想到树套树，但这是个区查区改，~~（原题是区间 Max，但发现我不会做）~~所以我写出来之后发现不可做，后来在岳神的指点下，改成了区间求和，并写出了分块+线段树的高端操作。

对于每行（或列）维护一个区间修改区间查询的 线段树 或 非递归线段树 或 树状数组。

同时每$\sqrt{n}$行（或列）维护一个标记线段树，对于其 L~R=Sum 表示从该线段树所管辖的 Y1~Y2

这$\sqrt{n}$行(或列)每行(或列)的 L~R 都增加了 Sum。

每$\sqrt{n}$行(或列)维护一个总和线段树，其 L~R 表示该线段树所管辖的 Y1~Y2 的这$\sqrt{n}$行(或列)所有行(或列)的 L~R 的总和。

具体实现要想清楚。

T3 等差数列+二次函数判断+贪心

数据读入

把兵种排个序，对于每个敌人二分查找以下就能知道这是几号兵种。

对于每个不扣血单位一定会在扣血单位消灭后打，且一定是每个都打到只剩一枪的血量，然后以回血量递增的顺序消灭。

对于扣血单位可以证明每个单位一定是一次性打死，每个兵种也是，但由于打击顺序无法贪心，注意到能造成伤害的兵种<=20 所以需要状压 Dp，表示在当前状态下最多剩下多少血，对于 0 血或负血的状态不能用来更新其他 dp，初始状态为所有灭掉一个兵种的 dp 值为 HP，拥有全部兵种的 dp 值为 HP+HPadd。

然后从全部兵种开始枚举状态，然后枚举打哪个兵种，杀死单个敌人的时间为 Time=Atk/Hpi 上取整，Numi 表示该敌人被杀死之前的同兵种敌人数量，其他兵种（含加血）的攻击力为 OtherSum.

对于杀死单个敌人的伤害为(Time-1)*Atki+(Numi-1)*Time*Atki+Time*OtherSum。

发现杀死一个兵种的总伤害可以用等差数列求和公式化简（等差数列求和+另一个部分），这样就可以 O（1）算出最终的血量。

到此为止已经可以得到绝大部分分数了，但如果打一个兵种之前是总伤害扣血，打完后是加

血，可能最终血量>0，但过程中出现<=0 的情况，所以我们需要进行判断。
可以通过构造一个二次函数求最值来判断是否出现上述情况。
我们设 Time 表示打死一个需要的时间，Num’(0<=Num’<=Num)为打到恰好还剩 Num’(std 中的意义是 Num’-1,所以式子有所不同)个单位。
则在打这个兵种时的伤害函数为
OtherSum * Time*(Num-Num’)+(Num*Time+(Num’+1)*Time)*(Num-Num’)/2*atk
-(Num-Num’)*atk
最后的-(Num-Num’)*atk 是因为敌人后开火，所以敌人会在死前少打一枪。
对这个函数进行丧心病狂的整理之后就变成了
y=Ax^2+Bx+C
的形式。这样我们又拿到了一个测试点的分数。
注意到该函数是指打死某些兵种后受到的的总伤害，但其实可能打死这个人后是加血，之前是减血，此时在打到某一个敌人只剩 1 枪的血量时受到的伤害最大。
所以如果 Time>1 则要少打一枪（Time==1 时就是上面的函数）。
上面的函数 - OtherSum - Num’*atk 就得到了新函数。
然后二次函数求最值（如果坐标不是整数，则按其左右的整点计算）判断是否会死亡。
这样就得到了满分做法。