数据恢复

题目来源：codeforces gym 292796 J
正如题目所说，因为时间来不及，所以这题是搬的。

算法 1（图是一条链）

显然选择方法只有一种，因此直接模拟即可。

算法 2（$n \le 20$）

不清楚搜索算法可不可以通过，但是没有任何剪枝应该是不行的。
状压 dp 是可以通过的。
但是不知道神奇的贪心能不能通过，因为数据可能不是很强。

算法 3（菊花图）

第一次一定选 1 号点，然后剩下的 $n-1$ 个点就没有依赖关系了。

不难证明，当所有点按照 $\frac{a}{b}$ 从小到大排序后选取最优。

算法 4（全部子任务）

考虑拓展算法 3 的做法，我们记 $v_i = \frac{a_i}{b_i}$，那么我们贪心的选择 v_i 最小的。

此时分两种情况，一种是 i 的依赖已经被恢复了，那么直接恢复 i 一定是最优的。
而如果 i 的依赖没有被恢复，那么在 i 的依赖被恢复后直接恢复 i 一定是最优的。
因此我们可以把 i 和 f_i 进行一个捆绑，也就是把 i 和 f_i 合并成一个新的点，这个点按照 $f_i - i$ 的顺序包含了这两个点。而新的这个点的 a 和 b 是原来两个点的值的和。因此新点的 v 也可以计算了。
如果把这个新的点代替原来两个点，重新建树，那么新问题可原问题是等价的。
因此只需要不断重复这个操作，直到所有点都合并成一个点即可。
找依赖关系可以使用并查集维护，找最小值可以用堆维护。
时间复杂度：$O(n \log n)$

注： CF 官方题解中所谓的 $O(n)$ 并没有考虑堆的效率，所以时间复杂度还是 $O(n \log n)$ 的。

路哥

题目即问有多少种断边方案使得包含 1 号结点的联通块大小（权值）刚好为 k，最终答案只需要除以所有断边方案（即 2^{n-1}）即可。

20pts

枚举所有的断边方案，再 dfs 一遍判断该方案是否可行，时间复杂度 $O(2^n * n)$。

40pts

设立状态 $f_{i,j}$ 表示结点 i 的子树中包含 i 的连通块的权值为 j 的方案数依次进行树上背包，可以直接枚举儿子结点枚举权值转移，时间复杂度为 $O(n^3)$。

40pts

在树上背包的前提下优化。观察每个点的转移为每个儿子的所有权值方案和当前所有权值方案的卷积，可以使用多项式优化，时间复杂度 $O(n^2 \log n)$。

100pts

在树上背包的前提下使用常见树上背包优化技巧，即设立状态 $f_{i,j}$ 表示在 dfs 到 i 点时权值为 j 的方案总数，在每次递归的时候将该点的方案下传给儿子，再在每次回溯的时候将儿子的贡献计算到父亲上，再考虑不选连向这个儿子的边（这个儿子的子树中的边随便选）的所有方案即可，时间复杂度为 $O(n^2)$。

质数

- 给定一个复数序列，支持区间乘，区间赋值，区间询问素数个数。

算法一

- 我们假定如果 x 是素数，那么 -x 也是素数，询问则是问正素数个数。
- 容易发现的是，除了 1 外的非素数在赋值前都不会变成素数了。
- 使用平衡树维护序列中的连续段，遇到区间乘法就搜索，如果子树中都不是素数或 1 就跳过打标记。
- 这里可能会遇到问题，一个数被乘好几次还是素数或一怎么办？
- 容易发现乘的数必须是 0, 1, -1，这些全部打标记解决。
- 维护的信息只有子树正素数个数，负素数个数，是否存在素数，或 1。容易解决。
- 容易发现会被统计进入答案的素数不会超过值域。

算法二

- 现在是复数了，可能会产生奇异的状况。
- 例如：$(2 + i)(2 - i) = 5$，他变成了素数。
- 对复数稍微有点了解的同学会知道，复数的乘法就是模长相乘，辐角相加，也就是说这里模长是素数平方，他就可能是个素数。
- 使用类似刚刚的做法，如果一个数的模长平方因子个数超过 3，则它必定不是素数。而乘 0, -1, 1, i, -i 外的数，模长必定增加。
- 维护子树中是否存在因子个数不超过 3 的数，以及 辐角为 0°,90°,180°,270° 的模长是素数的数的个数，容易打标记解决。

4 小 j 的组合

我们先考虑存在哈密顿圈才停止怎么做，记这个时候答案是 g'。事实上，题目中的操作可看做将 v 的经过次数 $+1$。我们设最后每个点要求的经过次数为 e_i，那么 $g' = \sum_i e_i - 1$. 显然 $e_i \geq d_i$，其中 d_i 是 i 的度数，那么 $g' \geq \sum_i d_i - 1 = n - 2$. 另一方面若 $e = d$，取树的 Euler tour representation 即满足要求。故 $g' = n - 2$.

那么原题做法就很明显了，g 就是 n 减去树直径的顶点数，证明留作习题。

Bonus：每次选一个点 $\rightarrow$ 每次选原树上的一条路径