树上杂题

南开信竞教练组 董又铭

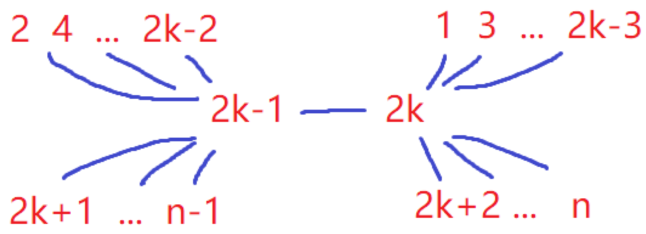
[A]热身题

- n个点的完全图，n是偶数
- 把n(n-1)/2条边分成n/2组，每组n-1条边都是一棵树
- 输出方案

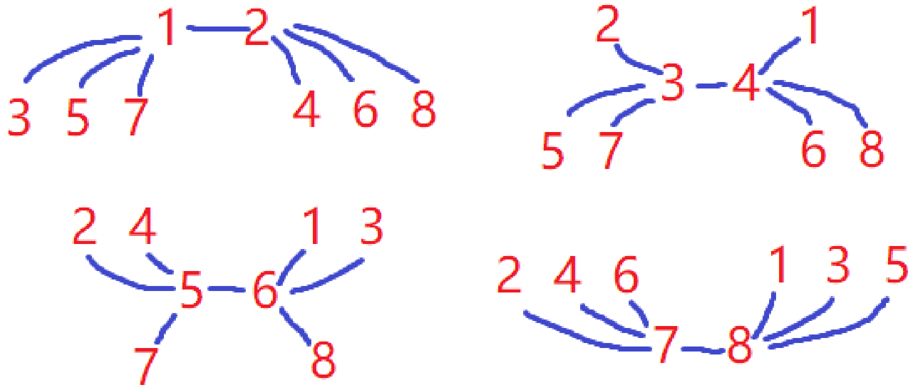
- $n \le 1000$

[A]热身题

- 方法1：每棵树如右图所示

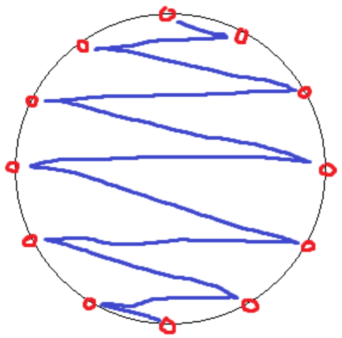


- 例如n=8的每棵树：

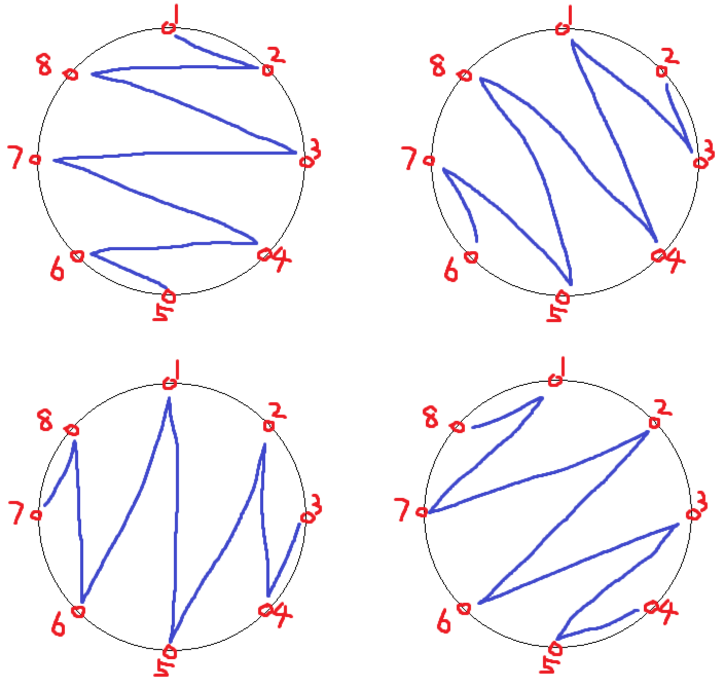


[A]热身题

- 方法2：将点排成一圈，按右图每次旋转1格



- 例如n=8的每棵树：
- （每棵树都是一条链）



[B] Trees of Tranquillity

- 给定两棵数T1,T2，节点数都是n，编号都是1~n。
- 找一个尽量大的编号集合，使集合中任意两个编号u,v满足：
  - 在T1上<u>是</u>祖先-后代关系
  - 在T2上<u>不是</u>祖先-后代关系
- 求集合的最大大小
- n ≤ 300 000

- 来源 CF1528C

[B] Trees of Tranquillity

- T1上的限制条件，提示要在T1上DFS
  - DFS时维护当前到根的链上，最多选多少个节点可以满足T2的限制
- T2上的限制条件，提示在T2上贪心选节点
  - 祖先后代关系的两个点，选择后代
  - 非祖先后代关系的两个点，都选

[B] Trees of Tranquillity

- T1上的限制条件，提示要在T1上DFS
  - DFS时维护当前到根的链上，最多选多少个节点可以满足T2的限制
- T2上的限制条件，提示在T2上贪心选节点
  - 祖先后代关系的两个点，选择后代
  - 非祖先后代关系的两个点，都选

- DFS时如何维护T2的贪心？
  - 把已选节点在T2上的DFS序编号放入容器
  - 查询是否和已选节点冲突，即查前驱后继
  - 用set即可
- 时间复杂度O(nlogn)

[C] Region Separation

- 给定一棵无根树，节点权值a[i]
- 将树按如下规则划分区域：
  - 区域共分为k个等级，k任意
  - 等级相同的每个区域没有交集，且节点权值之和相等。
  - 1级区域只有一个，即整棵树
  - 若i<k，则任何一个i级区域都被划分为至少两个i+1级区域
- 求方案数mod 10^9+7
- $n \le 10^6 \quad 1 \le a[i] \le 10^9$。

- 来源 CF1034C

[C] Region Separation

- 划分出x个同级区域？
  - x是n的约数，也是$S=\sum a[i]$的约数
  - 将子树内的点权求和得到数组s[i]
  - x可行，当且仅当s[i]是S/x倍数的节点i恰有x个

[C] Region Separation

- 划分出x个同级区域？
  - x是n的约数，也是S=∑a[i]的约数
  - 将子树内的点权求和得到数组s[i]
  - x可行，当且仅当s[i]是S/x倍数的节点i恰有x个
- 如何对x=1~n分别计算：s[i]是S/x倍数的i有几个？
  - 即A/x是gcd(S,s[i])的倍数
  - 先f[S/gcd]++，再枚举倍数求和

[C] Region Separation

- 划分出x个同级区域？
  - x是n的约数，也是S=∑a[i]的约数
  - 将子树内的点权求和得到数组s[i]
  - x可行，当且仅当s[i]是S/x倍数的节点i恰有x个
- 如何对x=1~n分别计算：s[i]是S/x倍数的i有几个？
  - 即A/x是gcd(S,s[i])的倍数
  - 先f[S/gcd]++，再枚举倍数求和
- 总方案数？
  - 找出所有f[x]==x的x
  - 枚举倍数，DP计算方案数
- 时间复杂度O(n(logn+loga))。

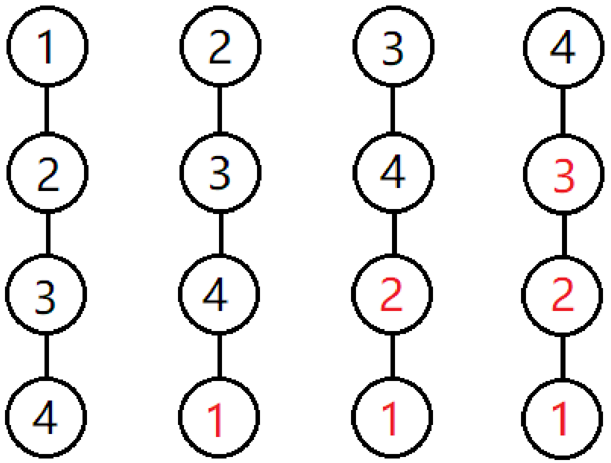
[D] Tree Calendar

- 给定一棵有根树，节点1是根。
- Touko进行了这样的操作：
  1. DFS这棵树，将DFS序编号作为节点权值a[i]。
  2. 在所有u是v父亲且a[u]<a[v]的节点对中，找到二元组(a[u],a[v])字典序最小的一对，交换swap(a[u],a[v])
  3. 反复执行第2条操作若干次，然后将此时的a[1~n]告诉你
- 根据Touko给你的a[1~n]数组判断：
  - 这个a[1~n]是否有可能是假的？
  - 如果是真的，求出第2条操作执行了多少次。
  - 如果是真的，求出初始DFS序。
- n ≤ 300 000
- 来源 CF1508E

[D] Tree Calendar

- 不知道儿子的顺序，所以不知道初始DFS序
- 手动观察第2条操作的过程：

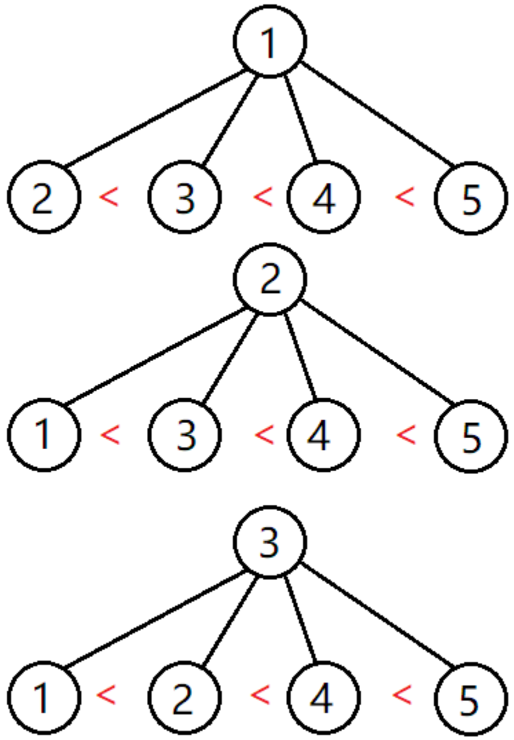
1. 是依次将1,2,3...通过若干次swap“沉底”



[D] Tree Calendar

- 不知道儿子的顺序，所以不知道初始DFS序
- 手动观察第2条操作的过程：

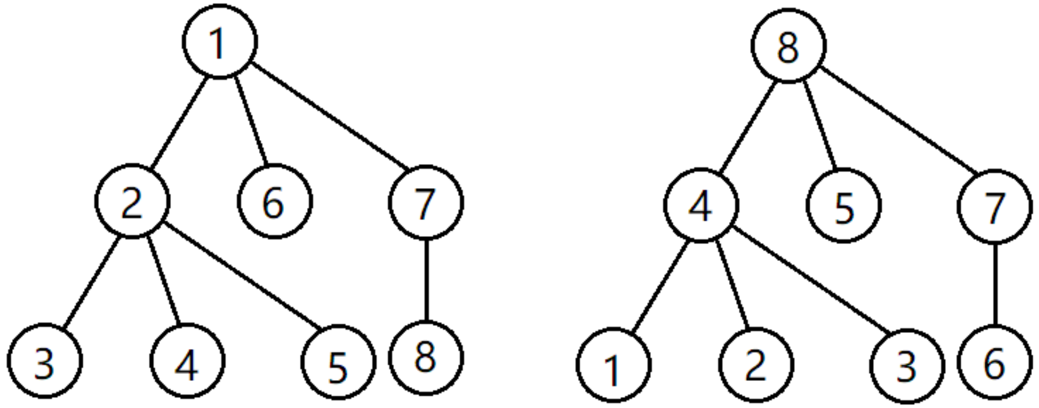
1. 是依次将1,2,3...通过若干次swap“沉底”
2. 同一节点每个儿子的大小关系不会变



[D] Tree Calendar

- 不知道儿子的顺序，所以不知道初始DFS序
- 手动观察第2条操作的过程：

1. 是依次将1,2,3...通过若干次swap“沉底”
2. 同一节点每个儿子的大小关系不会变
3. 最终a[1~n]会变成DFS的后续遍历



[D] Tree Calendar

- 不知道儿子的顺序，所以不知道初始DFS序
- 手动观察第2条操作的过程：

1. 是依次将1,2,3...通过若干次swap“沉底”
2. 同一节点每个儿子的大小关系不会变
3. 最终a[1~n]会变成DFS的后续遍历

- 利用性质2得到儿子顺序，即得到初始DFS序b[1~n]、后续遍历c[1~n]
- 对比a[1~n]和c[1~n]，得到已经“沉底”的节点有几个
- 一步步模拟“正在沉底”的节点，检查a[1~n]真假同时算出总步数
- 时间复杂度O(n)

[E] 救灾

- 节点编号1~n，节点i的父亲是$\varphi(i)$（欧拉函数）
- 每个节点都需要喝水，可以挖井取水，或者到最近的井取水。
- 挖一口井的费用为c[0]，到距离为d的井取水费用为c[d]
- 求最小总费用。
- $n \le 200\ 000$ $c[0], c[d] \le 10^9$

- 来源 NKOJ3030 nodgd命题

[E] 救灾

- 树的形态不是直接读入，必有蹊跷！
- 观察发现，n≤200 000时，树高≤20，似乎是O(logn)
  - 为什么呢？$\varphi(n\text{是奇数}) = \text{偶数}$，$\varphi(n\text{是偶数}) \le n/2$。

[E] 救灾

- 树的形态不是直接读入，必有蹊跷！
- 观察发现，n≤200 000时，树高≤20，似乎是O(logn)
  - 为什么呢？$\varphi(n\text{是奇数}) = \text{偶数}$，$\varphi(n\text{是偶数}) \leq n/2$。

- 考虑树形DP，状态需要二维：
  - f1[u][d]和f2[u][d]表示u子树内总费用
  - f1[u][d] 表示u子树<u>内</u>有一口与u距离为d的井，节点u去这里打水
  - f2[u][d] 表示u子树<u>外</u>有一口与u距离为d的井，节点u去这里打水
- 状态转移时f1要考虑井位于哪个子树，略
- 时间复杂度O(nlogn)

[F] 抢购与促销

- 给定一棵有根树
- 每个节点有一个商品，售价p[i]，必须按DFS序依次购买所有商品
- 由于“买一送一”促销，只需要支付第1,3,5,7,...件商品的费用
- 合理规划路线，使总费用最小。求最小总费用。
- $n \le 500\ 000 \quad 0 \le p[i] \le 10^9$

- 来源 nodgd命题
- 加强版：无根树，对每个节点当根分别计算答案

[F] 抢购与促销

- 树形DP：
  - 根据u在DFS序上的奇偶性，u子树有两种的策略
  - f0[u]和f1[u]分别表示u节点免费/付费的情况下，u子树的最小费用
- 如何调整儿子v1,v2,...的顺序，才能使费用最小呢？

[F] 抢购与促销

- 树形DP：
  - 根据u在DFS序上的奇偶性，u子树有两种的策略
  - f0[u]和f1[u]分别表示u节点免费/付费的情况下，u子树的最小费用
- 如何调整儿子v1,v2,...的顺序，才能使费用最小呢？
  - 设儿子v子树大小是s[v]，并按s[v]的奇偶性分成两组
  - 如果所有s[v]都是偶数，则顺序不影响总费用
  - 如果至少有一个奇数

[F] 抢购与促销

- 树形DP：
  - 根据u在DFS序上的奇偶性，u子树有两种的策略
  - f0[u]和f1[u]分别表示u节点免费/付费的情况下，u子树的最小费用
- 如何调整儿子v1,v2,...的顺序，才能使费用最小呢？
  - 设儿子v子树大小是s[v]，并按s[v]的奇偶性分成两组
  - 如果所有s[v]都是偶数，则顺序不影响总费用
  - 如果至少有一个奇数
    - 则偶数的s[v]都可以插入到合适位置，费用min(f0[v],f1[v])
    - 只考虑奇数s[v]的子树，它们要均匀分成两组，一组费用f0另一组费用f1
    - 先假设都选min(f0,f1)，再改正几个费用增量最小的
- 时间复杂度O(nlogn)，也可以做到O(n)
- 加强版：换根DP套路

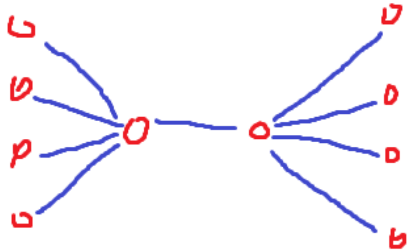
[G] ThREE

- 给定一棵n个节点的无根树
- 请你给每个节点赋一个权值p[i]，满足：
  - 权值构成1~n的排列
  - 任意距离为3的两个节点i,j，权值之和p[i]+p[j]、权值乘积p[i]*p[j]至少有一个是3的倍数
- 判断无解或构造一组方案。
- n ≤ 200 000

- 来源 Atcoder hitachi2020 C

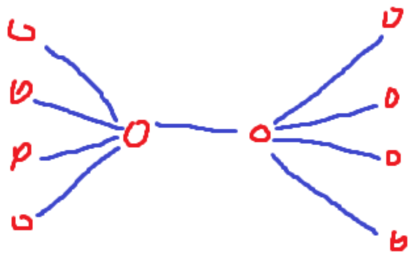
[G] ThREE

- 距离为3的点对很多，最坏情况如右图所示，几乎是二染色
- 大胆猜测：一定有解，且距离为奇数的点对都满足题意



[G] ThREE

- 距离为3的点对很多，最坏情况如右图所示，几乎是二染色
- 大胆猜测：一定有解，且距离为奇数的点对都满足题意



- 二染色后分别有m和n-m个点，不妨m<=n/2
- 若m <= n/3
  - 将m个节点都设为3k即可
- 否则n/3 < m <= n/2
  - 将m个节点设为尽量3k+1不够的用3k填补
  - 将n-m个节点设为尽量3k+2不够的用3k填补
- 时间复杂度O(n)

[H] 永远的LCT

- 一棵有根树，LCT规定：
  - 每个节点的重儿子可以随时变化
  - 一开始每个节点都没有重儿子。
- 一次access(u)操作：
  - 设从根到u的节点依次是v1,v2,...,vk（v1=根,vk=u）
  - 如果v[i] (i<k) 的重儿子不是v[i+1]就切换成v[i+1]
- 你可以进行m次access，每次任选节点。
- 设重儿子切换总次数最多是f(m)，当m趋于无穷时f(m)/m会趋于常数，而且是个有理数，求这个数mod 998,244,353的值。

- 来源 HDU6841 数据范围 n<=5000
- 来源 NKOJ7958 数据范围加强到 n<=300 000

[H] 永远的LCT

- 树形DP，f[u]表示u子树的答案
- 节点u的重儿子，有两种情况：
  - 在f最大和次大两个儿子之间反复横跳　　f[u] = (f[v1] + f[v2]) / 2 + 1
  - 一直是f最大的儿子　　f[u] = f[v1]

[H] 永远的LCT

- 树形DP，f[u]表示u子树的答案
- 节点u的重儿子，有两种情况：
  - 在f最大和次大两个儿子之间反复横跳　　f[u] = (f[v1] + f[v2]) / 2 + 1
  - 一直是f最大的儿子　　f[u] = f[v1]
- 题目要求取模，又要比大小，所以要同时计算f的浮点数值和模意义数值
  - 注意到状态转移中“/2”的次数可以是O(树高)次，即使long double精度也不够
  - 先无脑冲，时间复杂度O(n)。HDU原题可以AC，NKOJ上会WA。

[H] 永远的LCT

- 树形DP，f[u]表示u子树的答案
- 节点u的重儿子，有两种情况：
  - 在f最大和次大两个儿子之间反复横跳　　f[u] = (f[v1] + f[v2]) / 2 + 1
  - 一直是f最大的儿子　　f[u] = f[v1]
- 题目要求取模，又要比大小，所以要同时计算f的浮点数值和模意义数值
  - 注意到状态转移中“/2”的次数可以是O(树高)次，即使long double精度也不够
  - 先无脑冲，时间复杂度O(n)。HDU原题可以AC，NKOJ上会WA。
- 高精度浮点数
  - 用vector倒序存储小数部分二进制位，再用int存储整数部分
  - “/2”再“+1”只需push_back(1)
  - 加法时间复杂度等同于启发式合并
- 总时间复杂度O(nlogn)

[I] Joint Excavation

- 给定一个连通简单无向图。
- 你可以任选一条简单路径（节点不能重复经过），并将未经过的节点分成数量相等的两组，使这两组之间没有边。
- 输入保证有解，请构造一组方案。
- n,m <= 200 000

- 来源 NWERC2020 J NKOJ8170

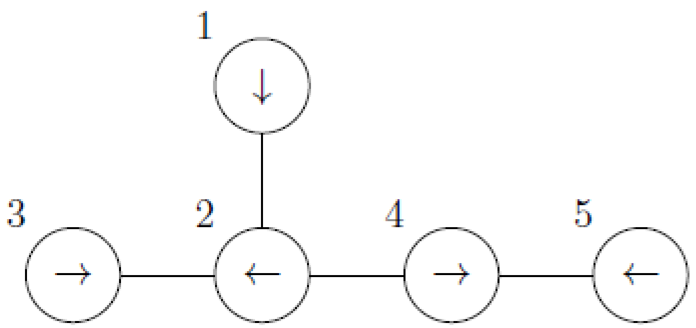
[I] Joint Excavation

- 考虑任选一个点开始DFS的过程
  - 路径：DFS栈中的节点
  - 集合1：已经DFS完返回的节点
  - 集合2：DFS还没去过的节点
- 注意到，集合1不断增大，集合2不断减小，且始终没有边
- 当集合1等于集合2时，停止DFS输出答案
- 时间复杂度O(n+m)

[J] Through Another Maze Darkly

- 一个树形迷宫，每个节点都给定了所有相邻节点按顺时针的顺序。
- 每个节点有一个激光指示器，指向某一个相邻的节点。
- 从节点1出发，按如下策略在迷宫中走k步，求到达的节点编号：
  - 先将激光指示器顺时针旋转到下一个，然后走到激光指示器指的节点。
- 共Q次询问，每次给定一个k。

- 来源 CCO2021 day1C NKOJ8224



[J] Through Another Maze Darkly

- 设节点1是根节点
- 观察行走过程发现：
  - 会多次回到根节点
  - 每次回根的间隔时间逐渐增大，直到每次可以遍历整棵树
  - 每当回根时，与根连通的一块区域指示器都指向父亲，且逐渐扩散
  - 能遍历整棵树后，一次遍历是欧拉环游序
  - 每次遍历的节点序列，都是下一次遍历的子序列

[J] Through Another Maze Darkly

- 设节点1是根节点
- 观察行走过程发现：
  - 会多次回到根节点
  - 每次回根的间隔时间逐渐增大，直到每次可以遍历整棵树
  - 每当回根时，与根连通的一块区域指示器都指向父亲，且逐渐扩散
  - 能遍历整棵树后，一次遍历是欧拉环游序
  - 每次遍历的节点序列，都是下一次遍历的子序列
- 先DFS算出，每个节点首次被访问是第几次回根之后。怎么查询？
  - 预处理每次回根的时间，二分查找得到当前是第几次回根之后。
  - 预处理每次遍历的节点序列，用可持久化线段树。
  - 线段树上二分得到走k步到达的节点
- 时间复杂度O((n+q)logn)