实外 CCF CSP2023-S模拟赛

（提高组：第5场）

时间：8:00 ~ 12:00

题目名称	生成树	序列	魔力	时空结构
题目类型	传统型	传统型	传统型	传统型
目录	tree	sequence	stardust	spacetime
可执行文件名	tree	sequence	stardust	spacetime
输入文件名	tree.in	sequence.in	stardust.in	spacetime.in
输出文件名	tree.out	sequence.out	stardust.out	spacetime.out
每个测试点时限	1.0 秒	1.0 秒	2.0 秒	3.0 秒
内存限制	512 MB	256 MB	512MB	512MB
子任务数目	10	10	20	10
测试点是否等分	是	是	是	是

提交源程序文件名

对于 C++ 语言	tree.cpp	sequence.cpp	stardust.cpp	spacetime.cpp
对于 C 语言	tree.c	sequence.c	stardust.c	spacetime.c
对于 Pascal 语言	tree.pas	sequence.pas	stardust.pas	spacetime.pas

编译选项

对于 C++ 语言	-O2
对于 C 语言	-O2
对于 Pascal 语言	-O2

注意事项（请仔细阅读）

1. 文件名（程序名和输入输出文件名）必须使用英文小写。
2. C/C++ 中函数 main() 的返回值类型必须是 int，程序正常结束时的返回值必须是 0。
3. 提交的程序代码文件的放置位置请参考各省的具体要求。
4. 因违反以上三点而出现的错误或问题，申述时一律不予受理。
5. 若无特殊说明，结果的比较方式为全文比较（过滤行末空格及文末回车）。
6. 程序可使用的栈空间内存限制与题目的内存限制一致。
7. 全国统一评测时采用的机器配置为：Inter(R) Core(TM) i7-8700K CPU @3.70GHz，内存 32GB。上述时限以此配置为准。
8. 只提供 Linux 格式附加样例文件。
9. 评测在当前最新公布的 NOI Linux 下进行，各语言的编译器版本以此为准。

生成树

(tree.cpp/.c/.pas)

题目描述

有一个 n 个点的完全图，每两点之间有边，i, j 之间的边权值为 $\gcd(i, j)$。

求这张图的最大生成树和最小生成树的边权和。

输入描述 (tree.in)

一行一个整数 n。

输出描述 (tree.out)

一行两个整数 x, y，分别表示最大生成树和最小生成树的边权和。

样例数据

Sample #1

Input

```
5
```

Output

```
5 4
```

Explanation

略

数据范围&约定

对于 30% 的数据，$n \leq 100$；

对于 100% 的数据，$n \leq 5 \times 10^5$。

序列

(sequence.cpp/.c/.pas)

题目描述

有 n 个人站成一个圈，每个人有 a_i 个球，下面要进行一次操作：

每个人把一些它的球交给它左边的人，所有人同时进行。

完了过后每个人有 b_i 个球，记这个序列为 B。

对于每种 B，它的价值为 $\prod b_i$。

对于所有可能的 B，你要计算它们的价值和，对 10^9+7 取模。

输入格式 (sequence.in)

第一行 n，接下来一行 n 个数表示 a_i。

输出格式 (sequence.out)

一行一个数表示答案。

样例数据

Sample #1

Input

```
3
1 1 1
```

Output

```
1
```

Sample #2

Input

```
3
2 1 1
```

Output

```
6
```

数据范围&约定

测试点 1，$n \leq 5, a_i \leq 3$。
测试点 2,3，$n \leq 10, a_i \leq 3$。
测试点 4，$n \leq 100, a_i \leq 100$。
测试点 5，$n \leq 100, a_i \leq 1000$。
测试点 6，$n \leq 100$。
测试点 7,8，$n \leq 2000$。
测试点 9,10，无特殊限制。
对于所有测试点，$n \leq 10^6, 0 \leq a_i \leq 10^9$。

魔力

(stardust.cpp/.c/.pas)

题目描述

圣诞节到了。

在 β 世界线上，冈伦、真由理和桶子准备装饰一棵圣诞树。

在 α 世界线上，红莉栖也准备装饰一棵圣诞树。

圣诞树可以看作是一棵有 n 个节点的树，以 1 号节点为根节点，且每个节点有一个权值 w_i，定义每个节点到根节点的距离为 d_i。

出于特别的原因，他们想要装饰的这棵圣诞树拥有跨越世界线的力量，因此在一条世界线中被装饰的圣诞树在另一条世界线中也会被装饰。

在发现圣诞树的神奇力量后，α 世界线的三人准备将圣诞树的每个节点**依次**交给红莉栖装饰。

由于个人喜好的差异，四人分别不喜欢一些情况，若这些情况出现在了所在组需要装饰的部分，就会产生一些不满值（具体定义见下文）：

冈伦不喜欢点对 (i,j)，当且仅当 i 是 j 的祖先节点且 $w_i > w_j$；

红莉栖不喜欢点对 (i,j)，当且仅当 i 是 j 的祖先节点且 $w_i < w_j$；

桶子不喜欢点对 (i,j)，当且仅当 $i < j$ 且 i 和 j 不成祖先关系，即 i 不为 j 的祖先节点且 j 不为 i 的祖先节点；

真由理不喜欢离树根太远的节点。

设红莉栖装饰的点集为 B，冈伦、真由理和桶子装饰的点集为 A，则需要满足 $B \cup A = \{1,2,\ldots,n\}$ 且 $B \cap A = \emptyset$，此时产生的不满值为：

$$\sum_{i\in A} d_i + \sum_{i\in A}\sum_{j\in A}[\text{冈伦不喜欢点对 } (i,j) \text{ 或桶子不喜欢点对 } (i,j)] + \sum_{i\in B}\sum_{j\in B}[\text{红莉栖不喜欢点对 } (i,j)]$$

其中 $[P]$ 的定义为若 P 为真则值为 1，否则为 0。

一开始，A 中包含了所有节点，即 $A = \{1,2,\ldots,n\}$，然后，冈伦三人组会**一个一个地**将 A 中的节点转移给 B，转移的顺序如下：

- 首先选出转移后不满值最小的节点；
- 若转移后不满值最小的节点有多个，则比较如下事项：假设该次转移了该节点，计算下一次、下下次、下下下次的转移在最优情况下的不满值，设节点 i 转移后的不满值序列为 $\{u_{i1}, u_{i2}, u_{i3}, \ldots\}$，则选择这个不满值序列字典序最小的节点；
- 若这样的节点仍有多个，容易发现选择哪个不会影响后续不满值，任选一个即可。

换句话说，转移的顺序等价为，对于一个长度为 n 的排列，依次将这 n 个元素从集合 A 转移到集合 B，设每次转移后不满值序列为 $\{u_1, u_2, \ldots, u_n\}$，则冈伦会选择所有 $n!$ 个排列中不满值序列字典序最小的一个进行转移，若有多个排列不满值序列字典序均为最小，则随意选一个。

冈伦想要知道，若按这个转移序列操作，初始不满值是多少，每次转移后产生的不满值是多少。

输入格式 (stardust.in)

第一行输入一个整数 n，表示圣诞树的节点个数。

接下来一行 n 个整数，第 i 个整数表示第 i 个节点的权值 w_i。

接下来 $n-1$ 行一行两个整数 u, v，表示一条树边。

输出格式 (stardust.out)

一行 $n+1$ 个整数，第一个整数表示初始不满值，接下来 n 个整数表示按上述排列转移后每次转移后的不满值。

样例数据

Sample #1

Input

```
4
2 4 2 3
1 2
1 3
2 4
```

Output

```
7 3 1 0 2
```

Explanation

开始转移前，真由理产生的不满值为 `4`，点对 $(2,3), (2,4), (3,4)$ 分别产生了 `1` 的不满值，故总不满值为 `7`。

此时，若转移 `1` 号节点，对不满值无影响，仍为 `7`；

若转移 `2` 号节点，不满值减少 `3`，为 `4`；

若转移 `3` 号节点，不满值减少 `3`，为 `4`；

若转移 `4` 号节点，不满值减少 `4`，为 `3`。

此时转移 4 号节点最优，故转移 4 号节点至 B 集合，第一次转移后不满值为 3。

第二次转移前，真由理产生的不满值为 2，点对 $(2,3)$ 产生了 1 的不满值。

此时，若转移 1 号节点，红莉栖将对点对 $(1,4)$ 产生不满值，不满值变为 4；

若转移 2 号节点，不满值减少 1，为 2；

若转移 3 号节点，不满值减少 2，为 1。

此时转移 3 号节点最优，故转移 3 号节点至 B 集合，第二次转移后不满值为 1。

第三次转移前，真由理产生的不满值为 1，没有点对产生不满值。

此时，若转移 1 号节点，红莉栖将对点对 $(1,4)$ 产生不满值，不满值变为 2；

若转移 2 号节点，不满值减少 1，变为 0。

此时转移 2 号节点最优，故转移 2 号节点至 B 集合，第三次转移后不满值为 0。

第四次转移前，没有不满值。

转移 1 至 B 集合，此时红莉栖将对点对 $(1,2),(1,4)$ 产生不满值，不满值变为 2。

Sample #2

Input

```
7
1 0 4 8 5 9 6
1 2
1 3
2 4
2 5
3 6
6 7
```

Output

```
23 16 11 7 4 3 4 9
```

Explanation

这里有一个很精彩的解释，但是我写不下。

Sample #3

见下发文件中 ex_stardust3.in/out，该组数据满足约定 1 的限制。

Sample #4

见下发文件中 ex_stardust4.in/out，该组数据满足约定 2 的限制。

Sample #5

见下发文件中 ex_stardust5.in/out，该组数据满足约定 3 的限制。

Sample #6

见下发文件中 `ex_stardust6.in/out`。

数据范围&约定

对于 20% 的数据，有 $n \leq 10$；

对于 40% 的数据，有 $n \leq 100$；

对于另外 10% 的数据，满足约定 `1`：树成一条链；

对于另外 10% 的数据，满足约定 `2`：所有节点权值相同；

对于另外 10% 的数据，满足约定 `3`：所有节点权值互不相同；

对于 100% 的数据，有 $1 \leq n \leq 5 \times 10^5, 0 \leq w_i \leq 5 \times 10^5$，保证输入的所有边构成一棵树。

时空结构

(spacetime.cpp/.c/.pas)

题目描述

时空结构可以看作是一棵以 1 号节点为根节点的大小为 n 的树，每个节点都有一个初始的干扰值 v_i。

真帆拟定了一个访问序列 P，这个访问序列是一个 1 到 n 的排列。

在第 i 时刻，真帆会访问 P_i 号时空节点，由于访问时空节点会对时空结构产生一定的干扰，因此第 P_i 号节点及其子树中所有节点的干扰值都会永久性地增加 v_{P_i}。

她想要知道，若随机拟定一个访问序列，最终所有时空节点干扰值的和的期望是多少。

设期望为 E，为了方便你的输出，你只需要输出 $E \times n!$ 即可，不难发现这是一个整数。

由于结果可能很大，所以你的输出应对 998244353 取模。

输入格式 (spacetime.in)

第一行一个正整数 n ($n \leq 10^5$)，表示时空节点的个数。

接下来 $n-1$ 行一行两个正整数 x, y，表示一条树边。

接下来一行 n 个非负整数，第 i 个表示 v_i (即第 i 个时空节点一开始的不稳定度)，有 $0 \leq v_i < 998244353$。

输出格式 (spacetime.out)

一行一个整数 ans，表示 $E \times n!$ 对 998244353 取模后的结果。

样例数据

Sample #1

Input

```
2
1 2
1 7
```

Output

```
35
```

Explanation

一开始两个节点的不稳定度分别为 `1 7`

若排列为 `1 2`，则第一个时刻后两个节点的不稳定度变为 `2 8`，第二个时刻后变为 `2 16`，这种情况的不稳定度之和为 `18`

若排列为 `2 1`，则第一个时刻后两个节点的不稳定度变为 `1 14`，第二个时刻后变为 `2 15`，这种情况的不稳定度之和为 `17`

最终的期望 $E=\frac{17+18}{2}$，$ans=E\times n!=\frac{17+18}{2}\times 2!=35$

Sample #2

Input

```
3
3 1
2 1
9 2 5
```

Output

```
354
```

Explanation

一开始三个节点的不稳定度为 `9 2 5`

若排列为 `1 2 3`，那么不稳定度的变化为：`9 2 5`，`18 11 14`，`18 22 14`，`18 22 28`，和为 `68`

若排列为 `1 3 2`，最终不稳定度为 `18 22 28`，和为 `68`

若排列为 `2 1 3`，最终不稳定度为 `18 13 28`，和为 `59`

若排列为 `2 3 1`，最终不稳定度为 `18 13 19`，和为 `50`

若排列为 `3 1 2`，最终不稳定度为 `18 22 19`，和为 `59`

若排列为 `3 2 1`，最终不稳定度为 `18 13 19`，和为 `50`

则 $E=\frac{(68+59+50)\times 2}{6}$，$E\times n!=\frac{(68+59+50)\times 2}{6}\times 3!=354$

Sample #3

Input

```
13
1 2
1 3
```

```
1 4
2 5
2 6
2 7
3 8
3 9
3 10
4 11
4 12
4 13
11 5 13 15 14 15 6 18 9 5 14 4 19
```

Output

```
602440893
```

Hint

本组数据满足任意一点到 1 号节点的最短路径只需要经过至多两条边。

Sample #4

见选手下发文件 `ex_spacetime4.in/out`，该组数据不具有特殊性质。

数据范围&约定

对于 30% 的数据，满足 $1 \leq n \leq 10$。

对于另外 20% 的数据，数据具有特殊性质，满足树是一张以 1 为根的菊花图。

对于另外 20% 的数据，数据具有特殊性质，满足树上任意一个点到达 1 号点需要经过的边数不超过 2 。

对于 100% 的数据，$1 \leq n \leq 10^5$，满足对于没有特殊性质的测试点，树为随机生成，具体而言，生成树的方法是先随机生成一个 `Prufer` 序列，再根据这个 `Prufer` 序列生成一棵树。

对于所有数据点，有 $1 \leq x, y \leq n, 0 \leq v_i < 998244353$，且保证给出的所有边构成一棵树。