湖南省队集训 Day5

长郡中学

2021 年 7 月 7 日

题目名称	下落的数字	序排速快	树
题目类型	传统型	传统型	传统型
目录	`fall`	`tros`	`tree`
可执行文件名	`fall`	`tros`	`tree`
输入文件名	`fall.in`	`tros.in`	`tree.in`
输出文件名	`fall.out`	`tros.out`	`tree.out`
是否捆绑测试	是	是	是
测试点数目	5	6	10
每个测试点是否等分	否	否	是
每个测试点时限	2.0 秒	2.0 秒	1.0 秒
内存限制	512 MB	512 MB	512 MB

提交源程序文件名

对于 C++ 语言	`fall.cpp`	`tros.cpp`	`tree.cpp`

编译选项

对于 C++ 语言	`-lm -std=c++11 -O2`

注意事项

1. 文件名（包括程序名和输入输出文件名）必须使用英文小写。
2. C++ 中函数 `main()` 的返回值类型必须是 `int`，值必须为 0。
3. 若无特殊说明，输入文件中同一行内的多个整数、浮点数、字符串等均使用一个空格进行分隔。
4. 若无特殊说明，结果比较方式为忽略行末空格、文末回车后的全文比较。
5. 在终端下可使用命令 `ulimit -s unlimited` 将栈空间限制放大，但你使用的栈空间大小不应超过题目限制。
6. 如果有其它问题请询问出题人。

下落的数字（fall）

题目描述

一棵以 1 为根的树，每个点有一个权值，一个数字从根节点出发，不断按照如下策略行动：

- 在当前点的所有儿子中，选择权值大于等于它且权值最小的儿子，走到那个儿子上。
- 如果不存在这样的儿子，它将停在当前节点。

问这个数字最终停在哪个节点。
m 次操作，每次操作为修改一个点的权值，或者查询一个数字 c 从根节点出发最终到达的点的编号。

输入格式

第一行两个数 n, m，为树的节点数和操作数。
接下来一行 n 个整数，第 i 个数表示 i 号点的权值 w_i。
接下来 $n-1$ 行，每行两个正整数 x, y 描述树上的一条边 (x, y)。
接下来 m 行每行描述一个操作，有如下两种格式：

- `1 a b` 表示把 a 节点的权值改成 b。
- `2 c` 表示查询数字 c 从根节点出发最终到达的点的编号。

注意 1 号点的权值并无意义，你不需要关心它以及对它的修改（尽管这样的修改可能存在）。
数据保证任意时刻所有点的权值互不相同。

输出格式

对于每个查询操作，输出一行一个整数，表示数字 c 最终到达的节点的编号。

样例 1

fall1.in	fall1.ans
5 5 6 7 8 4 2 1 2 1 3 3 4 3 5 2 8 2 4 1 3 5 2 1 2 3	3 2 5 4

样例 2, 3, 4

见选手目录下的 `fall/fall*.in` 与 `fall/fall*.ans`。

测试点约束

对于所有数据，保证 $1 \le n, m \le 2 \times 10^5$，$1 \le w_i, b, c \le 10^9$。

子任务编号	$n, m \le$	特殊性质	分值
1	5000		20
2	2×10^5	树为一条链	10
3	2×10^5	存在一个点的度数为 $n-1$	10
4	2×10^5	没有修改操作	20
5	2×10^5		40

序排速快（tros）

题目描述

定义一次泡冒为：

```
1: function BUBBLESORT(A)
2:     for i = 1 → LENGTH(A) − 1 do
3:         if A_i > A_{i+1} then
4:             SWAP(A_i, A_{i+1})
5:         end if
6:     end for
7: end function
```

即, 从该数组的第一个位置开始, 设当前进行到的位置为 i , 若 $A_i > A_{i+1}$ 则交换。

定义一次序排速快为：

```
1: function QUICKSORT(A)
2:     if LENGTH(A) = 1 then
3:         return
4:     end if
5:     while no partition points exist in A do
6:         cnt ← cnt + LENGTH(A)
7:         BUBBLESORT(A)
8:     end while
9:     divide A at all partition points; do quicksort at each piece
10: end function
```

即, 对于当前递归到的区间 $[l, r]$, 定义一个位置 i 为分割点, 当且仅当 $\forall x \in [l, i], y \in [i + 1, r], A_x < A_y$。

序排速快的过程是这样的, 对于当前的区间 $[l, r]$, 若该区间长度为 1 , 直接返回；

否则, 若该区间中存在分割点, 则找出所有分割点, 递归两两相邻分割点中的区间。

即, 若当前递归区间为 $[l, r]$, 分割点为 $p_1, p_2, \cdots, p_k, \forall p_i \in [l, r]$ 则递归 $[l, p_1], [p_1 + 1, p_2], \ldots, [p_{k-1} + 1, p_k], [p_k + 1, r]$。

若不存在分割点, 则执行多次泡冒, 每一次泡冒将 cnt 的值加上 $r - l + 1$, 直至出现分割点, 执行递归。

给出两个正整数 L, R , 对所有的 $n \in [L, R]$ 分别求出所有长度为 n 的排列的 cnt 值之和对 998244353 取模的结果。

输入格式

一行两个正整数 L, R, 意义同题目描述。

输出格式

为减少输出量，仅输出一个整数表示所有答案的异或和。

样例 1

tros1.in	tros1.ans
2 8	920329

样例 1 解释

当排列长度为 2 时存在两种排列 $1,2,2,1$。

对于第一个排列，第一次递归 $[1,2]$ 时，分割点 $1,2$ 都已出现，则递归 $[1,1],[2,2]$，可以发现 cnt 的值为 0。

对于第二个排列，第一次递归 $[1,2]$ 时，无分割点出现，则进行一次泡冒，排列变为 $1,2$，此时分割点 $1,2$ 均已出现，递归 $[1,1],[2,2]$, 可以发现 cnt 的值为 2。

故对于所有长度为 2 的排列，cnt 的和对 998244353 取模的结果为 2。

长度为 3 时存在 6 种排列，cnt 的值为 17 。

长度为 6 时，cnt 的值为 9648。

长度为 8 时，cnt 的值为 1000800。

样例 2, 3

见选手目录下的 `tros/tros*.in` 和 `tros/tros*.ans`。

测试点约束

对于 100% 的数据，$2 \le L, R \le 10^7$。

子任务编号	$R-L+1$	R	分值
1	≤ 10	≤ 8	10
2	≤ 10	≤ 500	10
3	≤ 10	≤ 5000	10
4	≤ 10	$\le 10^5$	20
5	≤ 10	$\le 10^6$	20
6	$\le 10^7$	$\le 10^7$	30

树（tree）

题目描述

给定一张 n 个点 m 条边的无向连通图和这张图的一棵生成树 T_1，**保证给定的图满足删掉任意一个点和它连接的所有边之后，这张图还是连通的**。

现在你想要 T_1 更换成图上的另外一棵生成树 T_2，你可以进行如下操作：

1. 选中 T_1 中的一个叶子 x。
2. 选择原图上一个的点 z $(z \neq x)$ 满足 z 和 x 在原图上相连，将 T_1 中的 x 和它的父亲之间的边断开，然后连接 x 和 z。

你想知道能否将 T_1 变成 T_2，如果能，请你构造一个方案。

输入格式

第一行两个正整数 n, m 表示图的点数和边数。

接下来 m 行四个整数 x, y, f_1, f_2 描述图上的一条边，f_1, f_2 分别表示这条边是否在 T_1, T_2 上。

输出格式

如果不能，输出 `No`。

否则第一行输出 `Yes`，第二行输出一个正整数 k 表示操作步数。

接下来 k 行，每行两个正整数 x, z，表示将 x 和其父亲断开，然后连接 x 和 z。

样例

tree.in	tree.ans
8 9 1 2 1 1 2 3 1 1 3 4 1 1 4 5 1 0 5 6 1 1 6 7 1 1 7 8 1 1 8 1 0 0 2 7 0 1	Yes 5 1 8 2 7 3 2 4 3 1 2

说明

本题使用 SPJ 进行评测。

下发文件中给出了 `checker.cpp`，使用方法是 `./checker <input-file> <output-file> <answer-file>`。

如果输出文件的 `Yes` 或 `No` 和答案文件给出的不一致，checker 会给出 `Wrong Answer [1]`。

如果在你的某次输出中出现了 $z = x$ 的情况，会给出 `Wrong Answer [2]`。

如果在某次输出中 x 不是叶子节点，会给出 `Wrong Answer [3]`。

如果 (x, z) 这条边没有在原图上出现过，会给出 `Wrong Answer [4]`。

如果最后操作完的树不是 T_2，会给出 `Wrong Answer [5]`。

否则会给出 `Accepted!`。

测试点约束

对于 100% 的数据，$3 \leq n \leq 100, m \leq \frac{n(n-1)}{2}$，保证图没有重边和自环。

子任务编号	$n \leq$	分值	子任务编号	$n \leq$	分值
1	3	10	6	9	10
2	4	10	7	11	10
3	5	10	8	13	10
4	6	10	9	15	10
5	7	10	10	100	10