Flag-based Tree Encoding and Recursive Traversals X22095_en

Several old-fashioned encodings of binary trees can be found in the complete works of the great Professor Donald Ervin Knuth. One of them, on which the related problem X30150 is based, uses tree sizes instead of marking the empty trees at the leaves. For the present problem, instead, we keep even less information; together with each root in the preorder traversal, we keep just two binary flags: whether the left subtree is empty, and whether the right subtree is empty. Careful thinking will prove to you that this is sufficient to uniquely identify the tree and reconstruct it.
To reduce potential confusion with nonnegative integers stored at the nodes, we do not employ zeros in the binary flags: instead, a nonempty subtree is represented by 1, and an empty subtree is represented by -1. Thus, trees are received by their preorder traversal and every node is described by three integer values: the nonnegative value stored at the root, a value 1 or -1 according to whether the left subtree is empty, and a value 1 or -1 according to whether the right subtree is empty. For instance, the encoding of the tree that can be seen by clicking on the pdf link of problem P98436 (the same one employed to exemplify X30150) would be:
3 1 1 0 1 1 7 -1 1 4 -1 -1 2 -1 -1 5 1 1 4 -1 -1 7 1 -1 6 -1 1 1 -1 -1
If the size of the tree is n, then the whole encoding uses $3n$ values.
Write a program that reads a tree encoded thus and outputs its postorder and inorder.

Input

A flag-encoded nonempty binary tree: its preorder traversal where each node information is followed by the +1/-1 flags indicating whether the left and the right subtrees are empty.

Output

The postorder and inorder traversals of the given tree, in two separate lines.

Sample input 1

```
3 1 1 0 1 1 7 -1 1 4 -1 -1 2 -1 -1 5 1 1 4 -1 -1 7 1 -1 6 -1 1 1 -1 -1
```

Sample output 1

```
post: 7 4 2 0 3 6 5 1 7
in: 7 4 0 2 3 4 5 6 1 7
```

Sample input 2

```
11 -1 -1
```

Sample output 2

```
post: 11
in: 11
```

Observation

In case you are using the `pytokr` module, be aware that you should not need the iterator: the function that reads the next word suffices.

Problem information

Author : José Luis Balcázar
Generation : 2024-06-06 11:47:31