Two elements of a binary search tree (BST) are swapped by mistake.

Recover the tree without changing its structure.

**Note:**  
A solution using O(*n*) space is pretty straight forward. Could you devise a constant space solution?

空间复杂度O(n)思路：

1. 中序遍历二叉树，把节点和节点值分别放在列表中
2. 把节点的值列表排序，再重新组成一个二叉搜索树，这样组成的二叉搜索树不一定是原来的搜索树，且时间复杂度高。

空间时间复杂度O(1)思路：

添加三个额外的变量 prev 、first、second

中序遍历时将first赋为第一个乱序的节点，second赋为第二个乱序的节点，最后再将first和second的值交换。Prev用来记录遍历的上一个节点，用来与本节点比较。

例如：

4

　　　　　　　/     \

　　            2        6

                /   \    /   \

               1    5  3    7

遍历到4时发现4比前一个节点值5小，因此4赋给first，5赋给second ，接着遍历到七时，此时first已经有值说明是第二个乱序，将7赋给second

最后再交换first和second的值即可

注意每引用一次全局变量一定要self