Inorder Tree Traversal without recursion and without stack!

Using Morris Traversal, we can traverse the tree without using stack and recursion. The idea of Morris Traversal is based on Threaded Binary Tree. In this traversal, we first create links to Inorder successor and print the data using these links, and finally revert the changes to restore original tree.

1. Initialize current as root
2. While current is not NULL
If the current does not have left child
a) Print current's data
b) Go to the right, i.e., current = current->right
Else
a) Find rightmost node in current left subtree OR
node whose right child == current.
If we found right child == current
a) Update the right child as NULL of that node whose right child is current
b) Print current's data
c) Go to the right, i.e. current = current->right
Else
a) Make current as the right child of that rightmost
node we found; and
b) Go to this left child, i.e., current = current->left

```
def inorderMorris(root):
    curr = root
    while curr!=None:
        if curr.left==None:
            print(curr.val,end=" ")
            curr = curr.right
        else:
            temp = curr.left
            while temp.right!=None and temp.right!=curr:
                temp = temp.right
            if temp.right is None:
                temp.right = curr
                #print(curr.val,end=" ") =====> This is PreOrder
traversal.
                curr = curr.left
            else:
```

```
                temp.right=None
                print(curr.val,end=" ")  =====> Inorder Traversal
                curr = curr.right
```