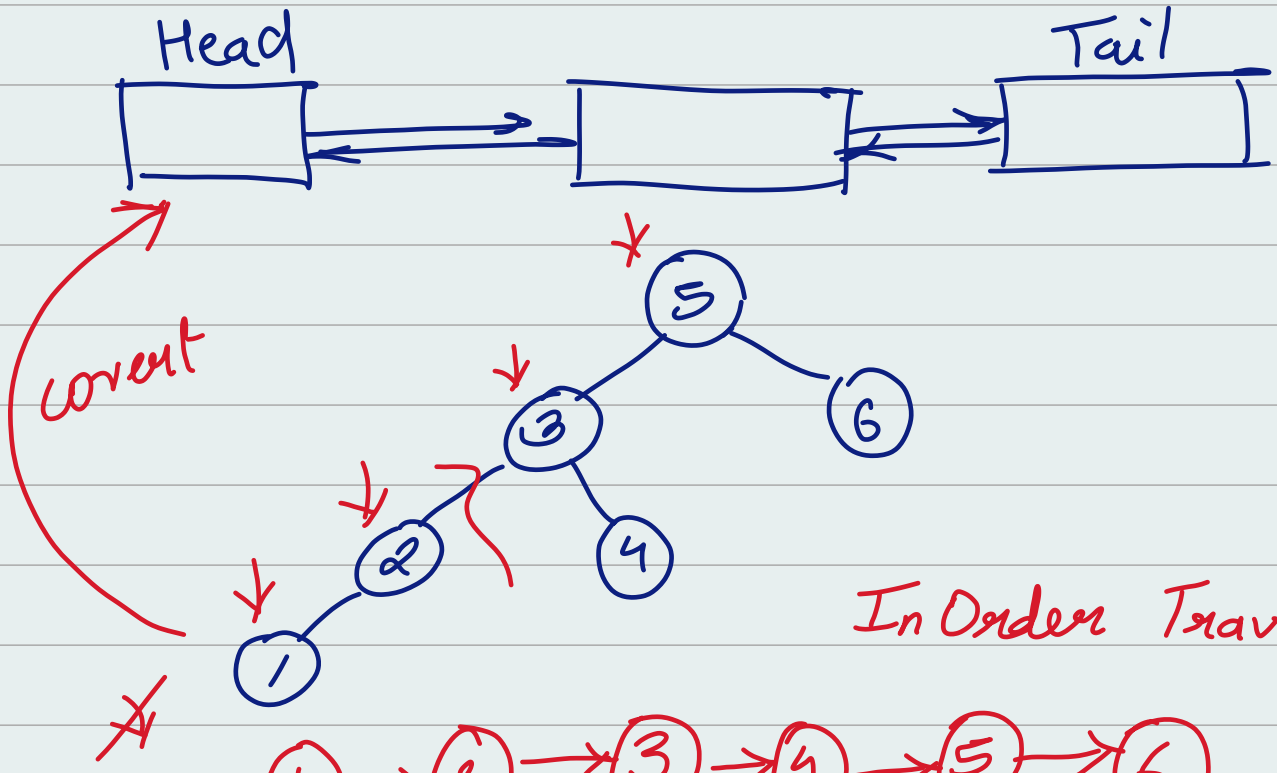
Convert a Binary Tree to Doubly Linked List

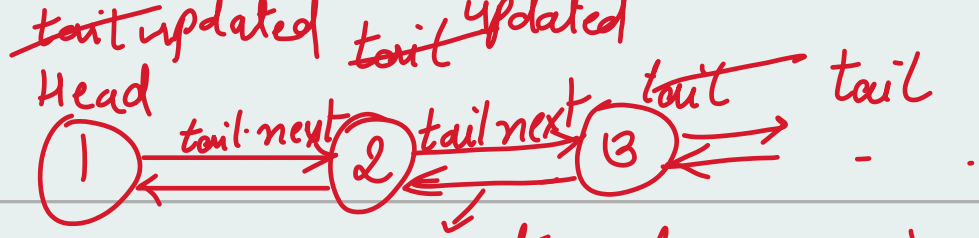
→ InOrder traversals gives sorted order in Binary Search Tree



InOrder Traversal

1 → 2 → 3 → 4 → 5 → 6

while printing DLL, check if the head is equal to null, yes currently list is empty. so assign (1) as head and tail



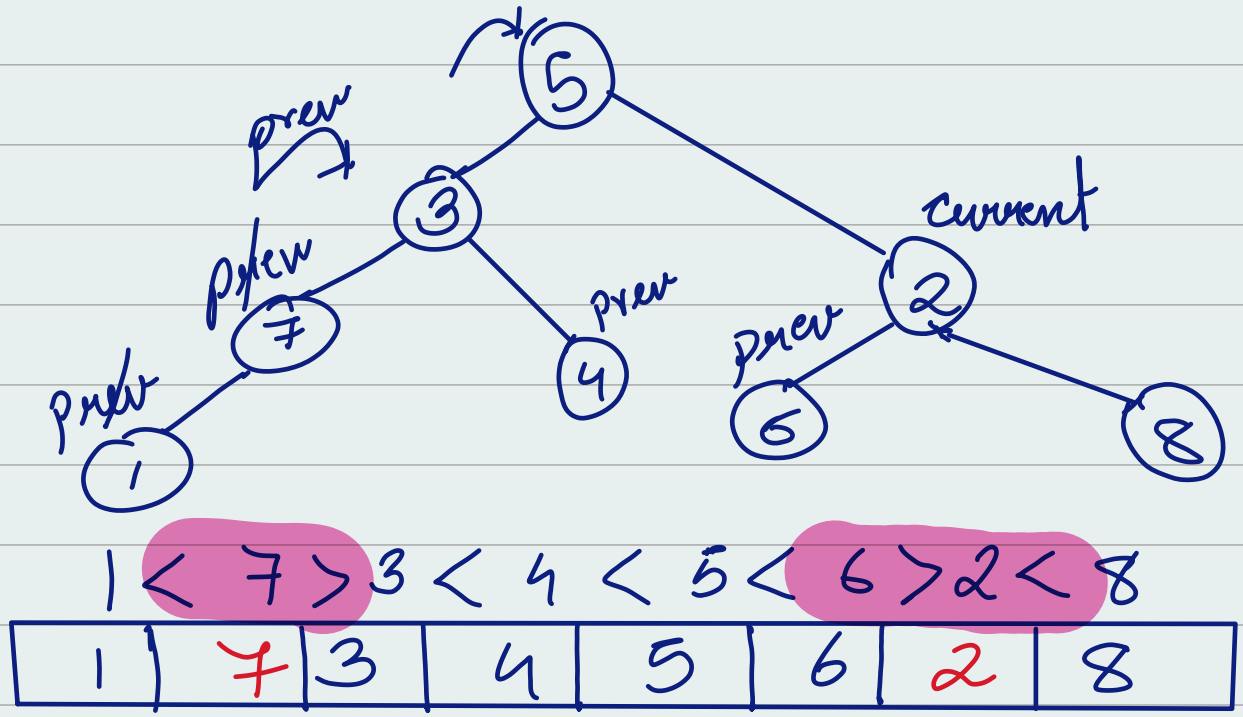
current node previous is the previous tail.

Code :-

```
public class BTtoDLL {

    LLNode head;

    LLNode tail;

    public TreeNode convert(TreeNode root){
        if(root == null){
            return null;
        }
        helper(root);
        return root;
    }

    private void helper(TreeNode node){
        if(node == null){
            return;
```

```
}
// call the inorder traversal
helper (node left);
LLNode newNode = new LLNode
                 (node.value);

if (head == null){
    head= newNode;   ① head
    tail = newNode,  ① tail
}
 else{
     tail.next = newNode;
      // tail.next =2
  newNode prev = tail // 2.prev=1
     tail = newNode; // update th
                        tail.
  }
  helper (node right);

}
// Class for linkedList
// make a class for Binary Tree
```

8) Correct Binary Tree thas has two nodes swapped.



1 < 7 > 3 < 4 < 5 < 6 > 2 < 8

1	7	3	4	5	6	2	8

Node first = ∅
Node second = ∅

(current < prev)
if (first != null)
first = prev,
second = current