Recover Two Misplaced Nodes of Binary Search Tree [LeetCode](https://leetcode.com/problems/recover-binary-search-tree/description/)

You are given the root of a binary search tree (BST), where the values of **exactly** two nodes of the tree were swapped by mistake. *Recover the tree without changing its structure*.

Example:

1

/ \

7 17

/ / \

5 15 19

/ /

1. 13

Output:

10

/ \

7 17

/ / \

5 15 19

/ /

1. 13

**Approach 1: Recover a binary search tree (BST) with two swapped nodes**

* **Function Purpose**:
  + The **recoverTree** function recovers a BST with two swapped nodes.
* **Explanation**:
  + It performs an in-order traversal of the BST and stores the values in a vector.
  + The vector is sorted, and the BST is updated with the sorted values.
* **Time Complexity**:
  + **O(N log N), where N is the number of nodes in the BST due to sorting.**
* **Space Complexity:**
  + **O(N), for the vector storing in-order traversal results and call stack space for recursion.**

**Approach 2: Recover a binary search tree (BST) with two swapped nodes using an optimized approach**

* **Function Purpose**:
  + The **recoverTreeOptimized** function recovers a BST with two swapped nodes using an optimized approach.
* **Explanation**:
  + It uses a helper function **recoverBSTHelper** to identify the misplaced nodes.
  + The function keeps track of the first and last misplaced nodes and then swaps their values.
* **Time Complexity**:
  + **O(N), where N is the number of nodes in the BST, as it traverses all nodes once.**
* **Space Complexity:**
  + **O(H), where H is the height of the BST, for the call stack space.**

**Conclusion**:

* Approach 2 is more efficient with a lower time complexity, making it the preferred approach.