572. Subtree of Another Tree

Easy 2445 116 Add to List Share

Given two **non-empty** binary trees **s** and **t**, check whether tree **t** has exactly the same structure and node values with a subtree of **s**. A subtree of **s** is a tree consists of a node in **s** and all of this node's descendants. The tree **s** could also be considered as a subtree of itself.

Example 1:
Given tree s:

```
     3
    / \
   4   5
  / \
 1   2
```

Given tree t:

```
   4
  / \
 1   2
```

Return **true**, because t has the same structure and node values with a subtree of s.

Example 2:
Given tree s:

```
     3
    / \
   4   5
  / \
```

```java
/**
 * Definition for a binary tree node.
 * public class TreeNode {
 *     int val;
 *     TreeNode left;
 *     TreeNode right;
 *     TreeNode() {}
 *     TreeNode(int val) { this.val = val; }
 *     TreeNode(int val, TreeNode left, TreeNode right) {
 *         this.val = val;
 *         this.left = left;
 *         this.right = right;
 *     }
 * }
 */
class Solution {
    public boolean isSubtree(TreeNode source, TreeNode trans)
{

        if(source == null || trans == null) return false;
        boolean machingWithParent = isMatched(source, trans);
        boolean machingWithLeft = isSubtree(source.left,
trans);
        boolean machingWithRight = isSubtree(source.right,
trans);

        return (machingWithParent || machingWithLeft ||
machingWithRight);
    }

    private boolean isMatched(TreeNode source, TreeNode trans)
{
        if ((source == null && trans == null))
            return true;
        if (source == null || trans == null)
            return false;
        return (source.val == trans.val) &&
```

Your previous code was restored from your local storage. Reset to default