```
/**
 * Definition for a binary tree node.
 * public class TreeNode {
 *     int val;
 *     TreeNode left;
 *     TreeNode right;
 *     TreeNode(int x) { val = x; }
 * }
 */
class Solution {
    public boolean isSubtree(TreeNode s, TreeNode t) {
        if (s == null && t == null) {
            return true;
        }
        if (s == null || t == null) {
            return false;
        }
        return isSame(s, t) || isSubtree(s.left, t) || isSubtree(s.right, t);
    }

    private boolean isSame(TreeNode s1, TreeNode s2) {
        if (s1 == null && s2 == null) {
            return true;
        }
        if (s1 == null || s2 == null) {
            return false;
        }
        return s1.val == s2.val && isSame(s1.left, s2.left) && isSame(s1.right, s2.right);
    }
}
```

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.