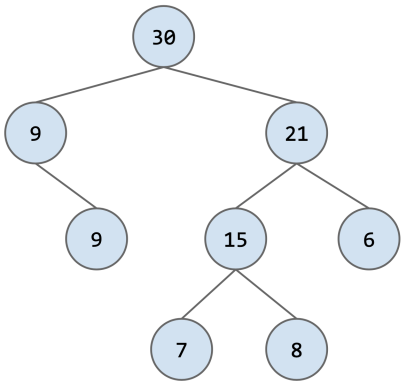
Binary Trees

For each of the following question, use the following `Tree` class for reference.

```
1  public class Tree {
2      public Tree(Tree left, int value, Tree right) {
3          _left = left;
4          _value = value;
5          _right = right;
6      }
7      public Tree(int value) {
8          this(null, value, null);
9      }
10     public int value() {
11         return _value;
12     }
13     public Tree leftChild() {
14         return _left;
15     }
16     public Tree rightChild() {
17         return _right;
18     }
19     private int _value;
20     private Tree _left, _right;
21 }
```

(a) Given a binary tree, check if it is a sum tree or not. In a sum tree, the value at *each* non-leaf node is equal to the sum of its children. For example, the following binary tree is a sum tree:




```
1  public boolean isSumTree(Tree t) {
2      ____________________________________________________________
3      ____________________________________________________________
4      ____________________________________________________________
5      ____________________________________________________________
6      ____________________________________________________________
7      ____________________________________________________________
8      ____________________________________________________________
9      ____________________________________________________________
10 }
```

(b) Given a binary tree with distinct elements, an input list, and an empty output list, add all the elements in the input list that appear in the tree to the output list. The elements in the output list should be ordered in the same order that would be returned from an inorder traversal.

For example, for the tree in Q.2(a), assuming either of the duplicate 9s have been removed, if the input list is [15, 9, 8, 30, 6], then after the operation the output list should be [9, 30, 15, 8, 6].

```
1  public static void sortRelative(Tree t,
2                                  List<Integer> inputList,
3                                  List<Integer> outputList) {
4      ____________________________________________________________
5      ____________________________________________________________
6      ____________________________________________________________
7      ____________________________________________________________
8      ____________________________________________________________
9      ____________________________________________________________
10     ____________________________________________________________
11     ____________________________________________________________
12 }
```