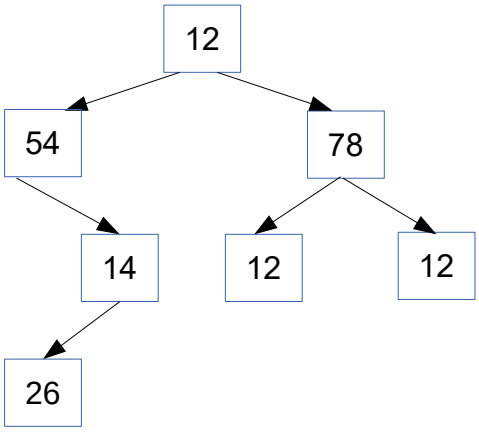
CSI 33 **Lecture 18** **Trees Intro** **In-class practice**

Given the tree



1. Give its array representation

2. Give its in-order traversal

3. Give its post-order traversal

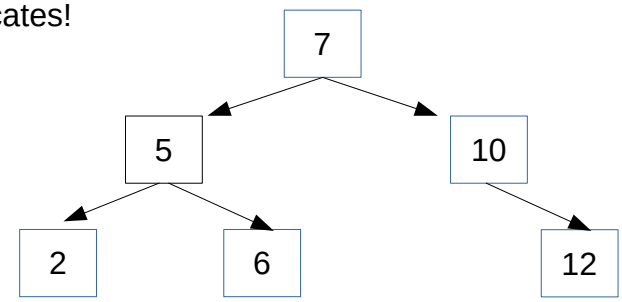
4. Give its pre-order traversal

5. Build/create this tree using class `TreeNode`

7. Consider a **Binary Search Tree** (**BST**): a binary tree where every node has the following property:

- Each value in the *left subtree* is less than the value at the node.
- Each value in the *right subtree* is greater than the value at the node.

Note that BSTs don't allow for duplicates!



Think of the implementation of such a tree:
1) We need to have a constructor/initializator (let's assume that we start with an empty tree!)
2) We need to decide on class attributes.
3) We would also like to be able to print the tree. Let's print it as an array. For this, we would use in-order traversal.
4) We should be able to insert the elements one by one! For insertion we could use this recursive strategy:

- if the root is empty, the value goes there
- if the root is not empty, compare the element with its value:
  - if the root's value is less than the value to be inserted, insert the value into the right subtree
  - if the root's value is greater than the value to be inserted, insert the value into the left subtree

If at any moment, the value to be inserted is equal to a node's value, report that we already have such a value in our tree

So your goal is to define a class BST with at least three methods: `__init__` , `__str__`, and `insert`.