June 10 - Trees

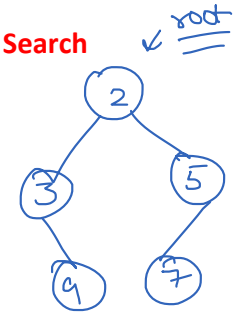
10 June 2020 20:13

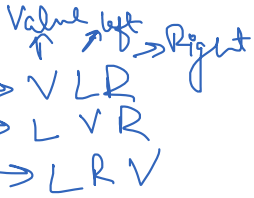
1. Traversals
- DFS - Preorder, InOrder, PostOrder //Done
- BFS - LevelOrder (With & Without Recursion) // at the end
2. Check if two trees are identical //Done
3. Check if two trees are mirror to each other //Done
4. Convert a tree into its mirror tree //Done
5. Check if two trees have the same structure //Done
6. Check if two trees are IsoMorphic //Done
7. Get the height of a tree iteratively without using recursion

H.W.

1. Find the level of a given node.

1. DFS: Depth First Search





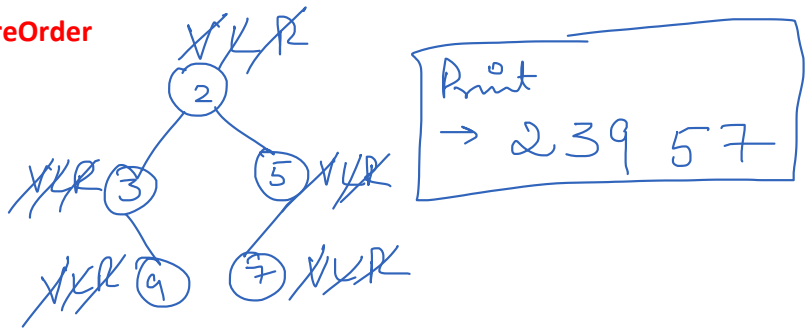
3 ways:

PreOrder → V L R

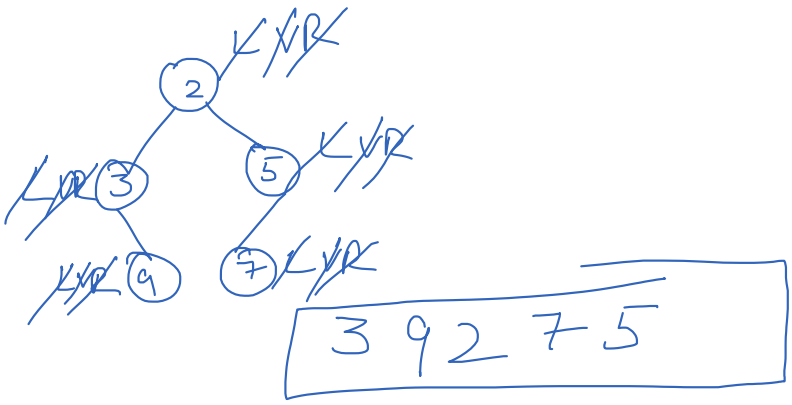
InOrder → L V R

PostOrder → L R V

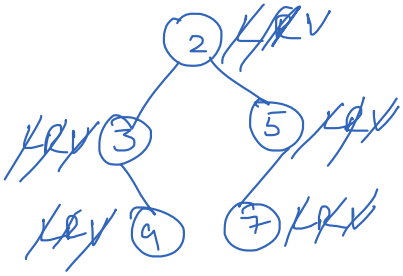
PreOrder

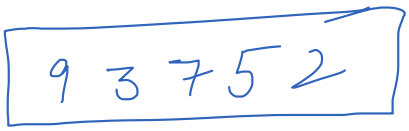


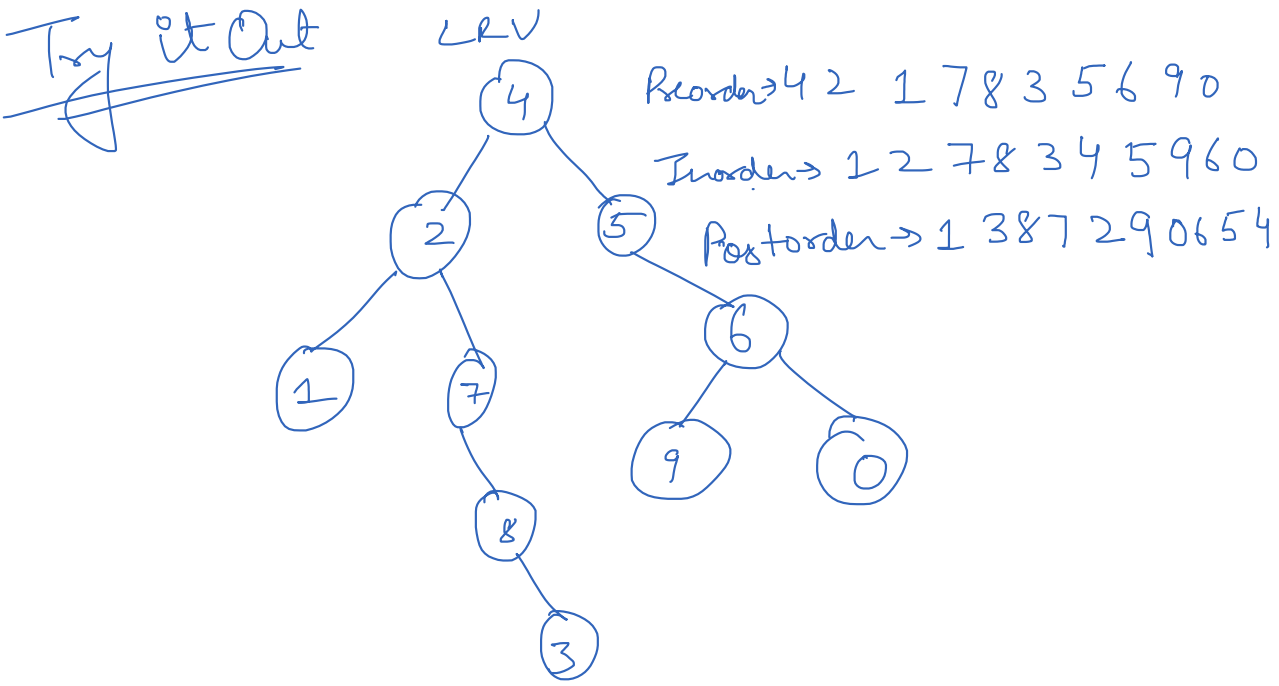
InOrder



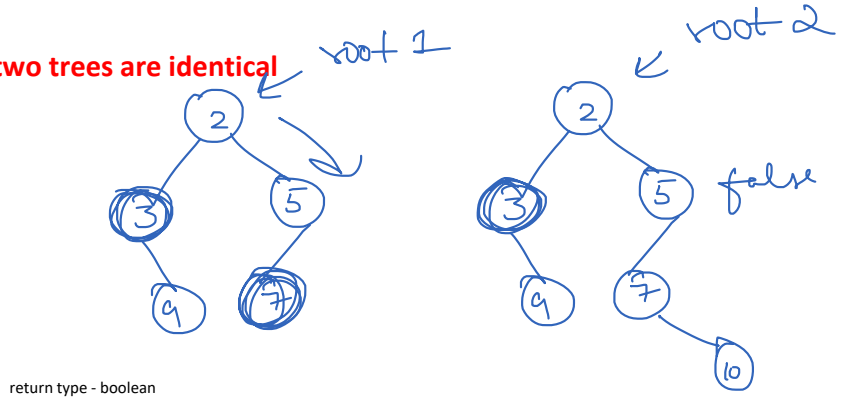
PostOrder







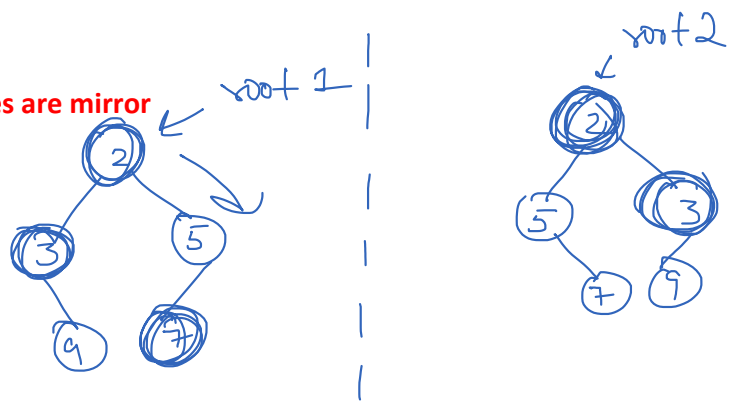
2. Check if two trees are identical



return type - boolean

If both the nodes are empty then they are identical
If both the nodes are not empty then they are not identical
Check root.data
Check LST's
Check RST's

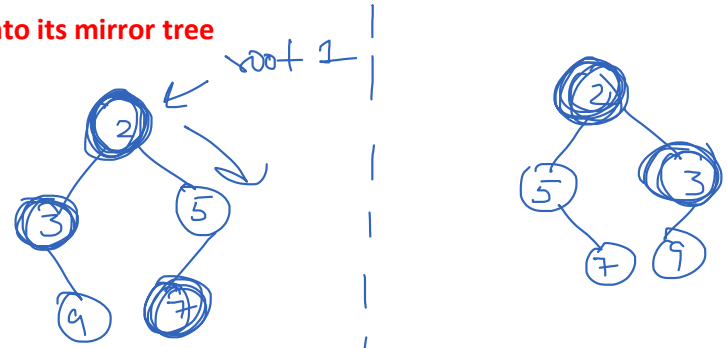
3. Check if two trees are mirror

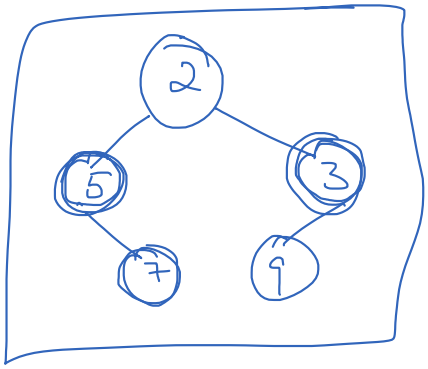


return type - boolean

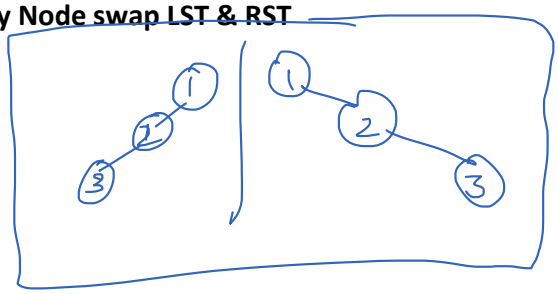
If both the nodes are empty then they are identical
If both the nodes are not empty then they are not identical
Check root.data of both trees
Check LST of 1st tree && RST of 2nd tree, they must be mirror
Check RST of 1st tree && LST of 2nd tree, they must be mirror

4. Convert a tree into its mirror tree

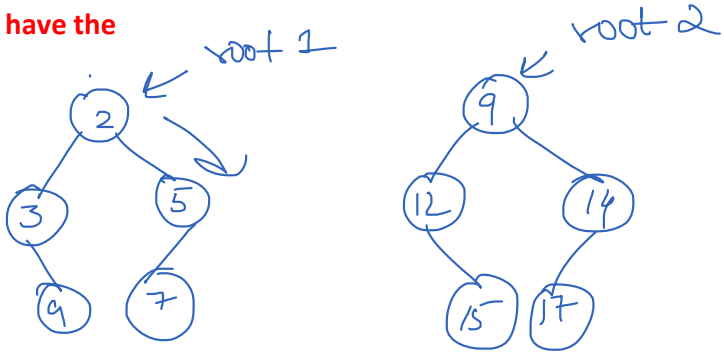




For every Node swap LST & RST



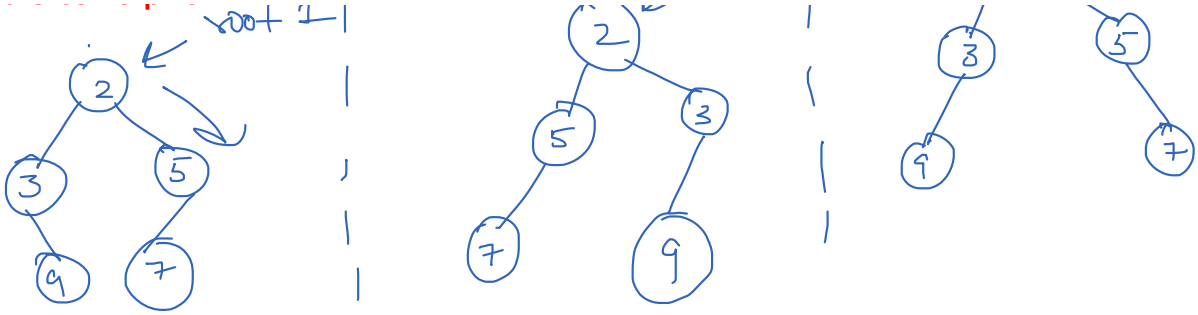
5. Check if two trees have the same structure



return type - boolean

If both the nodes are empty then they are identical
If both the nodes are not empty then they are not identical
Check LST's
Check RST's

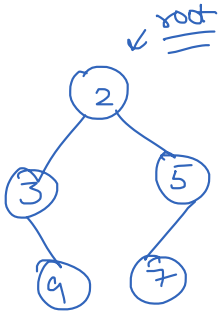
5. Check if two trees are IsoMorphic



IsoMorphic -> For every Node, either the children are identical or they can be mirror to each other.

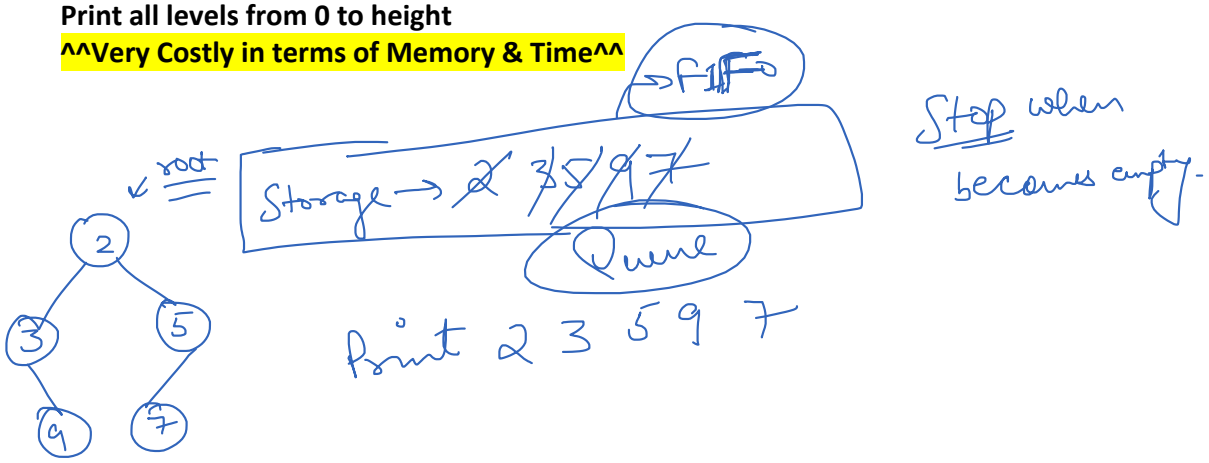
If both the nodes are empty then they are identical
If both the nodes are not empty then they are not identical
Check root.data of both trees
Either -> subtrees are identical in both the given trees
Or -> they are mirror

1. b) BFS: Breadth First Search - Level Order Traversal



BFS → 2 3 5 9 7

Find the height
Print all levels from 0 to height
^^Very Costly in terms of Memory & Time^^



Take a Queue
Put root into the queue
Till queue is not empty ->
{
Print the first item in queue
Put the children of first item in queue
Remove the first item

```
}
```