June - 12, BST

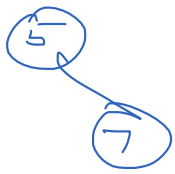
12 June 2020 20:09

1. Level Order Traversal //Done
2. Delete a Node in BST //Done
3. Lowest common ancestor(LCA) in Binary Tree //Done
4. LCA in BST //Done
5. Left view of a BT

HW:
Practice all questions from https://thecodingsimplified.com/binary-tree
&& https://thecodingsimplified.com/bst

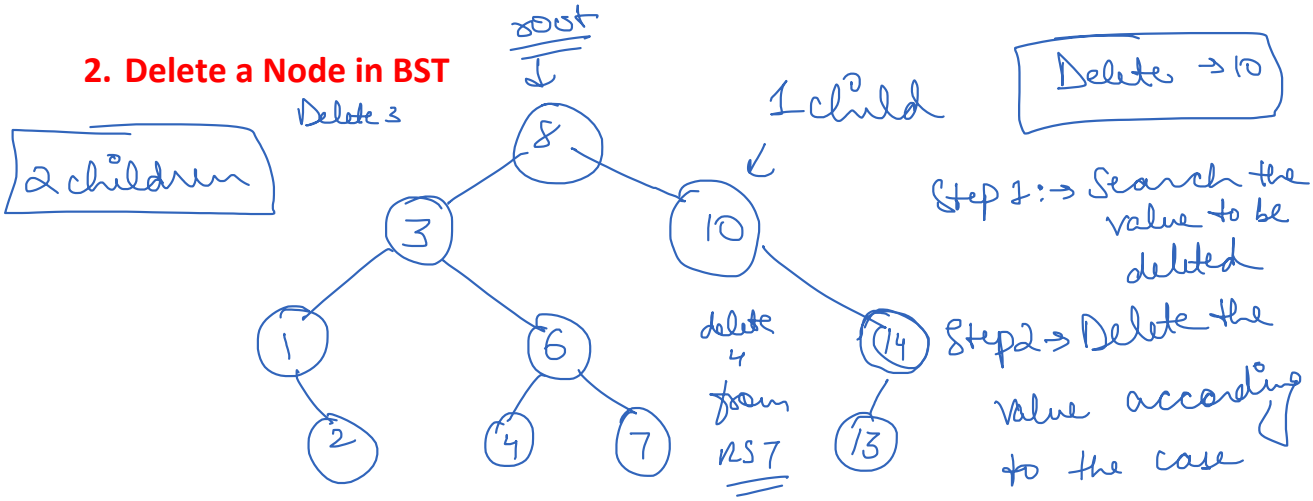
And do the views like, Right view, top view, bottom view.

1. BFS Iterative



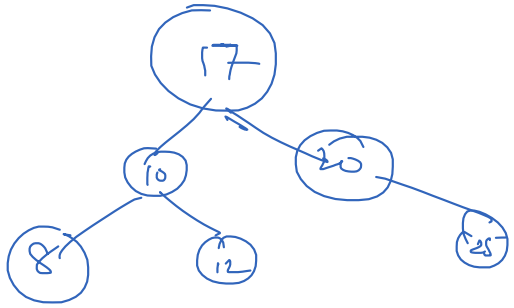
Take a Queue
Put root into the queue
Till queue is not empty ->
{
Print the first item in queue
Put the children of first item in queue
Remove the first item
}

2. Delete a Node in BST

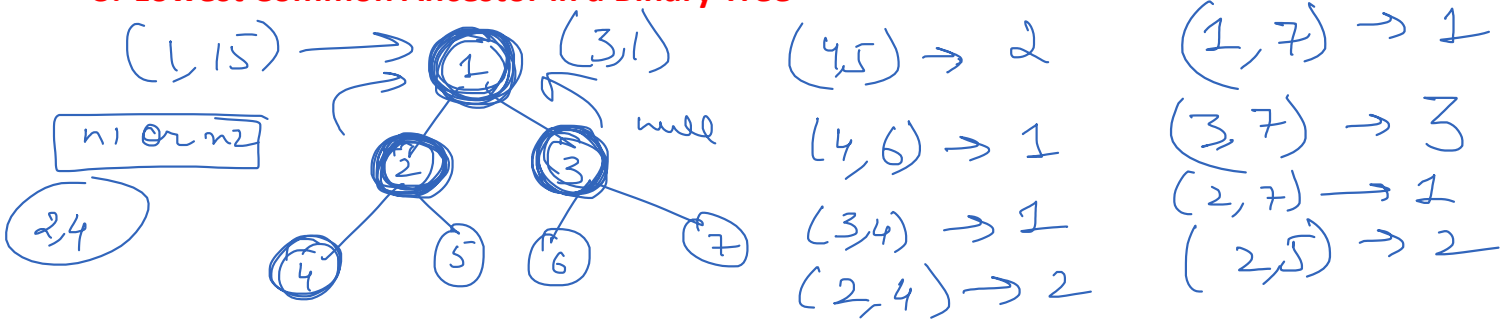



```
//data is the value to be deleted
Node deleteNode(Node root, int data)
{
    if(root==null) return null;
    if(data<root.data)
    {
        root.left = deleteNode(root.left,data);
    }
    else if(data>root.data)
    {
        root.right = deleteNode(root.right,data);
    }
    else{
        if(root.left == null) return root.right;        //1child case && No child case
        else if(root.right==null) return root.left;
        else{
            root.data = min(root.right);
            root.right = deleteNode(root.right,root.data);
        }

    }
}
```

```
    }
}
```



3. Lowest Common Ancestor in a Binary Tree



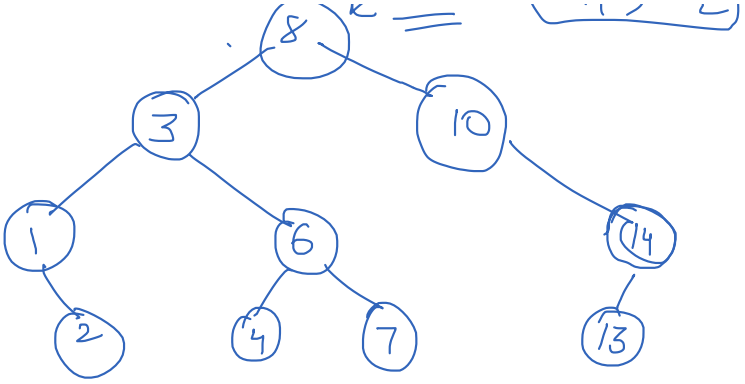

```
If(root == null) return null;
If(root.data == n1 || root.data==n2) return root;
If one node lies on left side and the other on right then that
particular node is the LCA.
If both are on left side then LCA will come from LST.
If both are on right side then LCA will come from RST.

Node findLCA(Node root, int n1,int n2)
    {
        if(root==null) return null;
        if(root.data==n1 || root.data==n2) return root;
        Node left_LCA = findLCA(root.left,n1,n2);
        Node right_LCA = findLCA(root.right,n1,n2);
        if(left_LCA!=null && right_LCA!=null)
        {
            return root; //current node is lca
        }
        if(left_LCA!=null) return left_LCA;
        return right_LCA;
    }
```

We are assuming that nodes with both values n1 & n2 exist

Will this code work for a BST? → Yes

BUT WE CAN DO BETTER

4. Lowest Common Ancestor in a BST



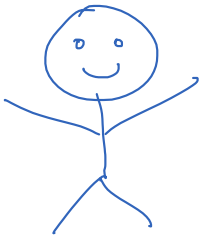
(2, 13) → 8

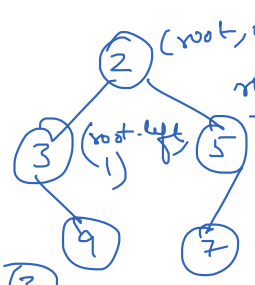
(3, 6) → 3

(3, 10) → 8

(10, 13) → 10

5. Left view of a Binary Tree





(levelPrint = 0)

3

[2, 3, 9] → First elements of each level.

```
void leftView (Node root, int level)
                              ↳ level of that particular Node
{ if (root == null) return;
  if (level == levelPrint)
  { SOP(root.data + " "); [2 3 9]
    levelPrint++;
  }
  leftView (root.left, level+1);
  leftView (root.right, level+1);
}
```