LEETCODE 235

By Priyansh

9) The problem is Asking to find the nearmost Ox lowest common Ancestor of two nodes of a binary Search trees-

```
class Solution {

'public:

TreeNode* lowestCommonAncestor(TreeNode* root, TreeNode* p, TreeNode* q) {

if(root-vual == p-vual || root > val == q-vual) return root;

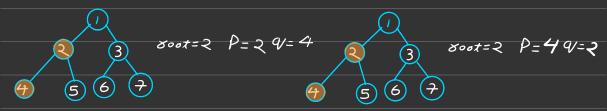
else if(root-vual > p-vual && root-vual < q-vual)return root;

else if(root-vual > q-vual && root-vual < p-vual)return root;

else if(root-vual > q-vual && root-vual > p-vual) return lowestCommonAncestor(root->left,p,q);

else return lowestCommonAncestor(root->right,p,q);
}

3.1
```



```
class Solution {
    public:
        TreeMode* lowestCommonAncestor(TreeMode* root, TreeMode* p, TreeMode* q) {
            if(root->val == p->val || root -> val == q->val) return root;
            else if(root->val > p->val && root->val < q->val) return root;
            else if(root->val > q->val && root->val < p->val) return root;
            else if(root->val > q->val && root->val > p->val) return lowestCommonAncestor(root->left,p,q);
            else return lowestCommonAncestor(root->right,p,q);
            }
        }
};

Made with Goodnotes
```

```
TreeNode* lowestCommonAncestor(TreeNode* root, TreeNode* p, TreeNode* q) {
     else if(root->val > p->val && root->val < q->val)return root;
     else if(root->val > q->val && root->val > p->val) return lowestCommonAncestor(root->left,p,q);
     else return lowestCommonAncestor(root->right,p,q);
                                  -> Root = Lowest common Ancis+exoco
     TreeNode* lowestCommonAncestor(TreeNode* root, TreeNode* p, TreeNode* q) {
       else if(root->val > p->val && root->val < q->val)return root;
       else if(root->val > q->val && root->val< p->val)return root;
       else return lowestCommonAncestor(root->right,p,q);
                                    → Rooz
lowes+
                                                   Called left from oot
Ansistor
       else if(root->val > p->val && root->val < q->val)return root;
       else if(root->val > q->val && root->val< p->val)return root;
       else if(root->val > q->val && root->val > p->val) return lowestCommonAncestor(root->left,p,q);
                                     > Root — ) called bight
                        ર
                                                  lowes+
                                                   common
                                                  Ansistor
```

NEW Code Same fasks-(In order to remove some unnecessary check list)...

```
class Solution {
    public:
        TreeNode* lowestCommonAncestor(TreeNode* root, TreeNode* p, TreeNode* q) {
            if(root->val > q->val && root->val > p->val) return lowestCommonAncestor(root->left,p,q);
            else if(root->val < p->val && root->val < q->val) return lowestCommonAncestor(root->right,p,q);
            else return root;
        }
    };
```