Nov23_PSP_3Apr

sudhakar venkatachalam
Vijay V A
Manjunatha I
Mayur Hadawale
Harshil Dabhoya
Sai Sharath
Yash Malviya
Rajeev
Kevin Theodore E
Shaurya Srivastava
Suraj Devraye
kameswarreddy Yeddula
Vigneshwaran K
ALLEN GEOSHAN M
manikandan m
Robin Dhiman
Pranadarth S
MD JASHIMUDDIN
Sarat Patel
Nitendra Rajput
SIJU SAMSON
Chandu
Pradeep Kumar Chandra
Mohammed Arshad
Pushkar Deshpande

Batch PSP

55.1 → 58%

Upcoming Contest

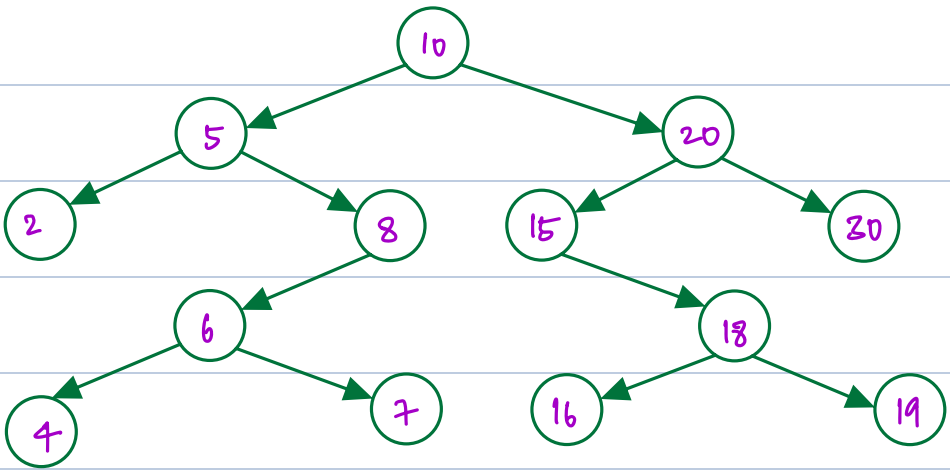
Two Pointer

Linked List

Stacks & Queues

pass → Dhiraj will personally call

Quick Recap

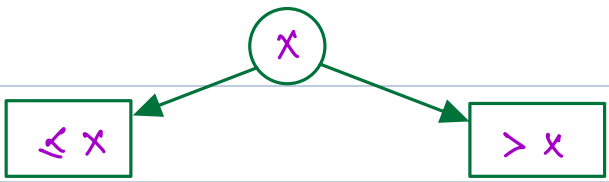


Binary Tree

1. Hierarchical data structure
2. Atmost two children (left and a right)

Binary search tree

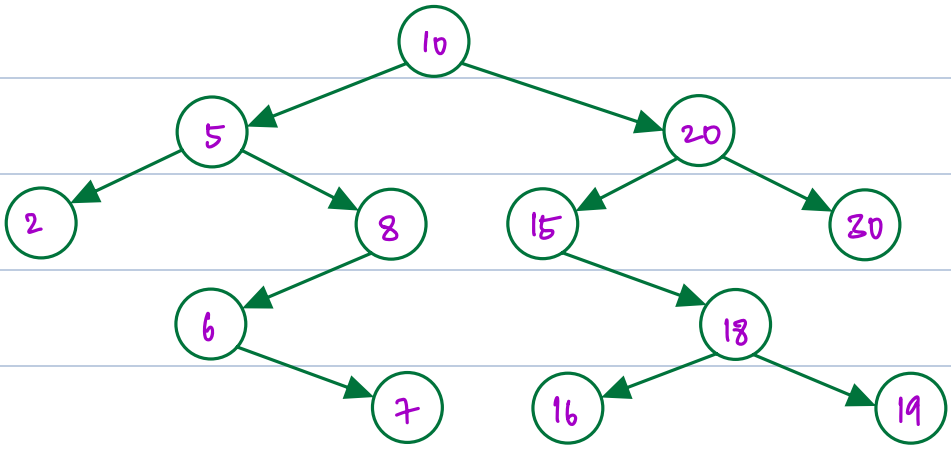
$\forall$ nodes x.



LST has all data $\leq x$

RST has data $> x$

Q1 → Given a Binary Search Tree and a positive Integer K, find Kth smallest element in BST



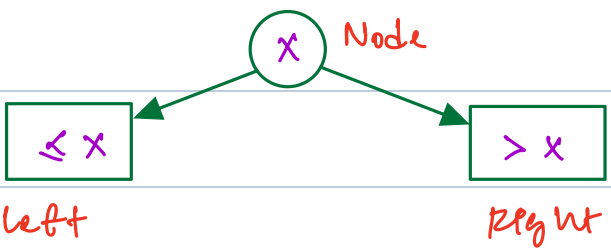
example

K = 4 ⟶ 7

K = 6 ⟶ 10

Observation

∀ nodes x.



Inorder traveral is sorted

left ≤ Node < Right

Solution

Do Inorder traversal and store Kth smallest element

pseudo code

```
int counter = 0;   int ans = -1;
void Inorder (root, k) {
    if (root == null) return;
    Inorder ( root.left, k)
    counter+=1;
    if (counter ==k)  ans = root.data;
    Inorder (root.right, k);
}
```

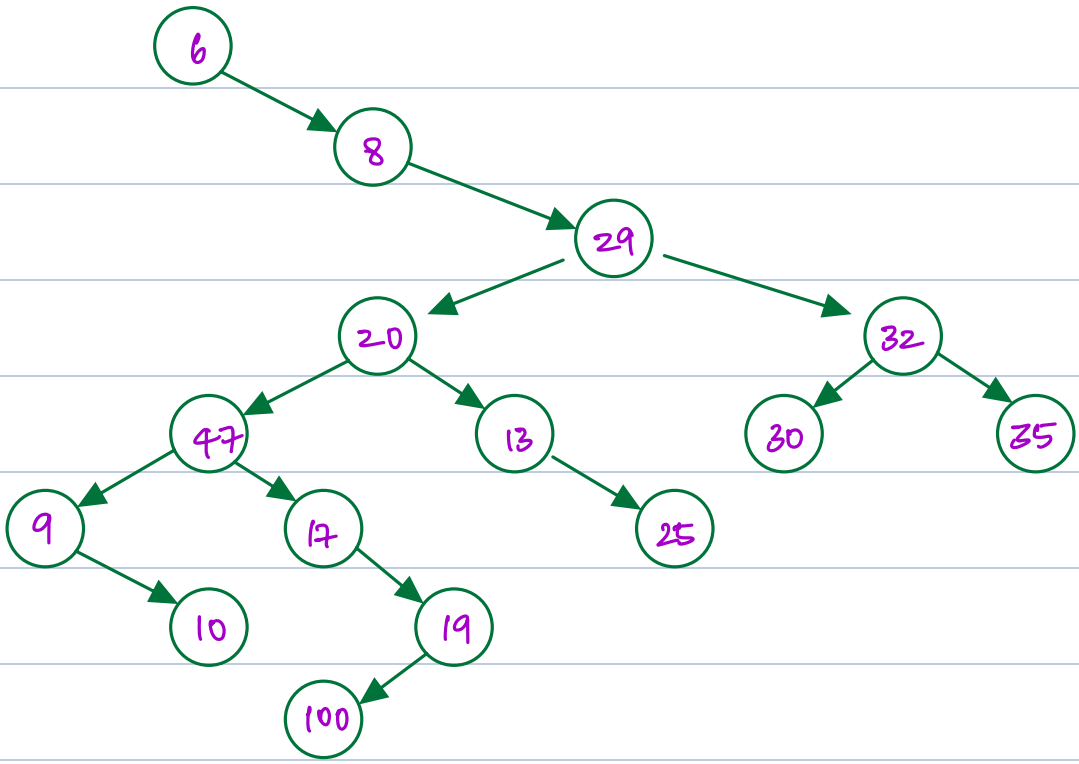
T.C = O(n) S.C = O(h)

Q2 → Morris Inorder Traversal

Without using any space, output the Inorder traversal

Recursion → we cannot use, stack space

Iteration → we cannot use stacks.



Inorder traversal

6	8	9	10	47	17	100	19	20	13	25	29	30	32	35
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Observation

① If node.left == null, print data and go right

② If node.left != null

Find the right most node on LST of node and point its right reference to node

while finding right most node if we come across curr node then remove the reference

a) Print curr.data

b) traverse to curr.right.

pseudo code

```
Node Morris Inorder (root) {
    If (root == null) return Null;
    Node curr = root;
    while (curr != null) {
        if (curr.left == null) {
            print (curr.data);
            curr = curr.right;
        }
        else { // Find right most node in LST
            temp = curr.left;
            while (temp.right != null &&
                   temp.right != curr)
                temp = temp.right;
            // create link
            If (temp.right == null) {
                temp.right = curr
                curr = curr.left
            }
```

// delete the link

else {

temp.right = null;

print (curr.data);

curr = curr.right;

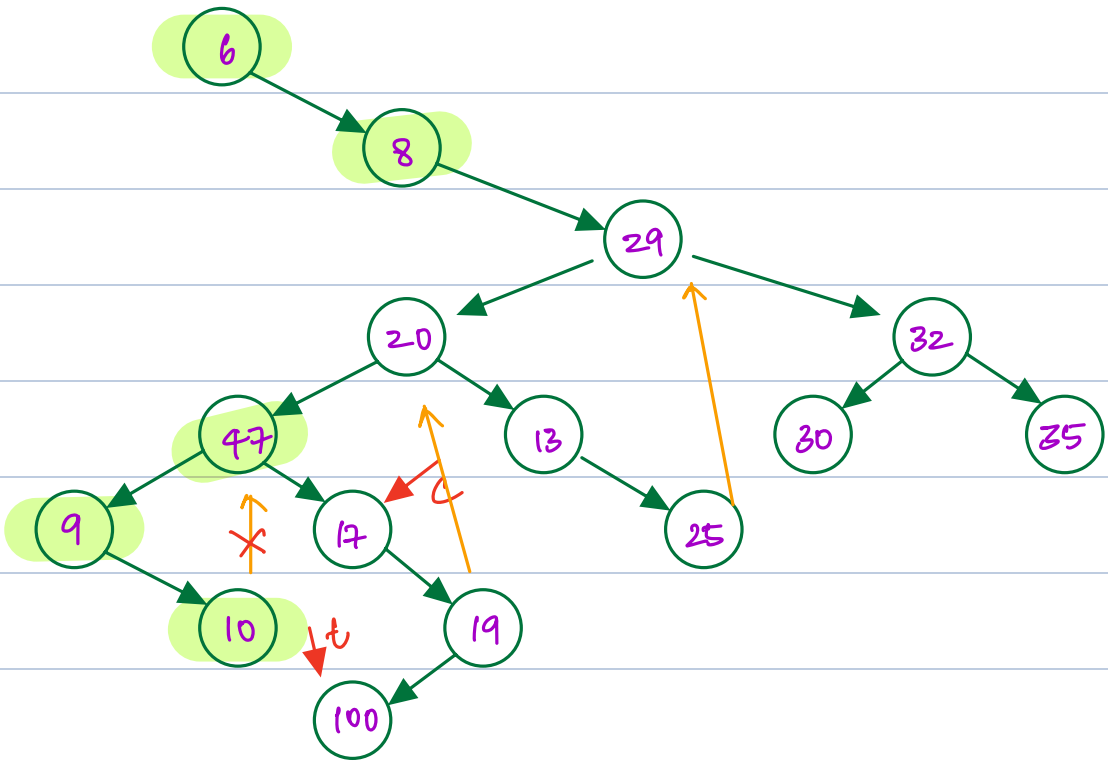
}

}

}

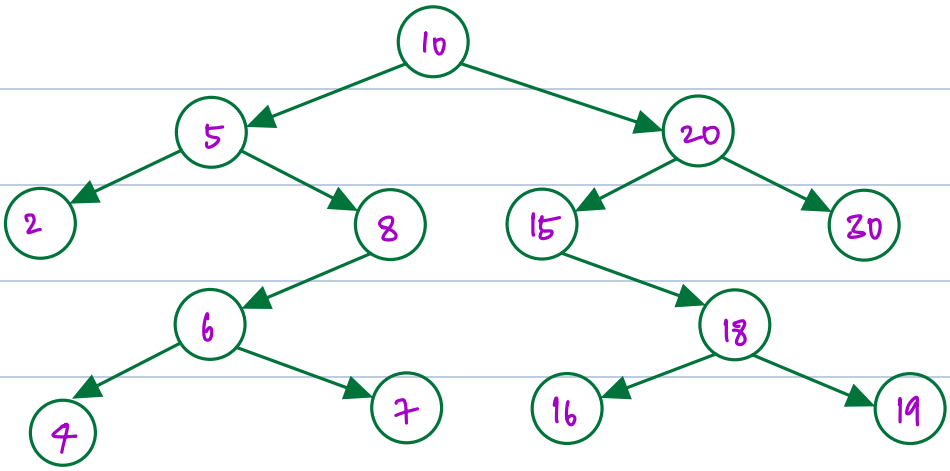
}

T.C = O(n) SC = O(1)



Break 10:20 pm → 10:26 pm

Q → Given a value, find path from root to that node (Binary Tree)



example

K = 7	10 → 5 → 8 → 6 → 7
K = 18	10 → 20 → 15 → 18

Observation

what traversal can we use?

level order traversal ✗

preorder → Inserting data

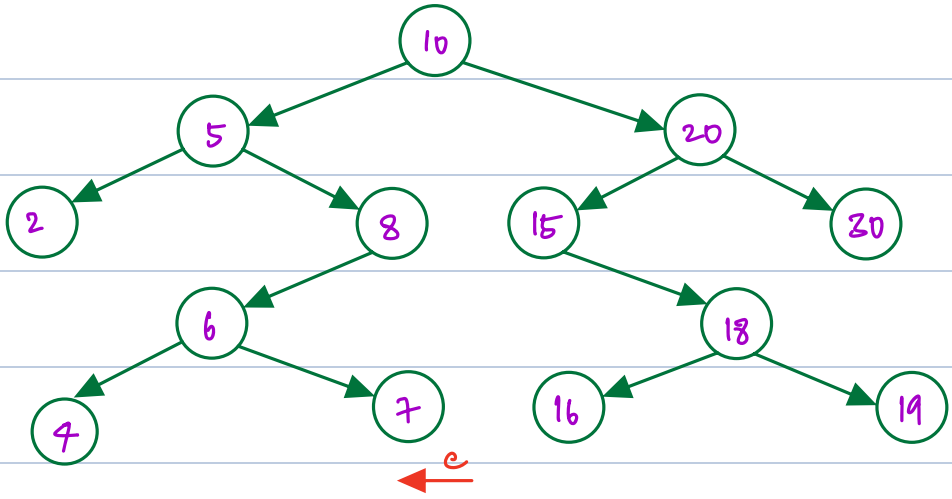
Inorder

postorder → removing data

Dry Run

Path →

10	5	8	6	7		



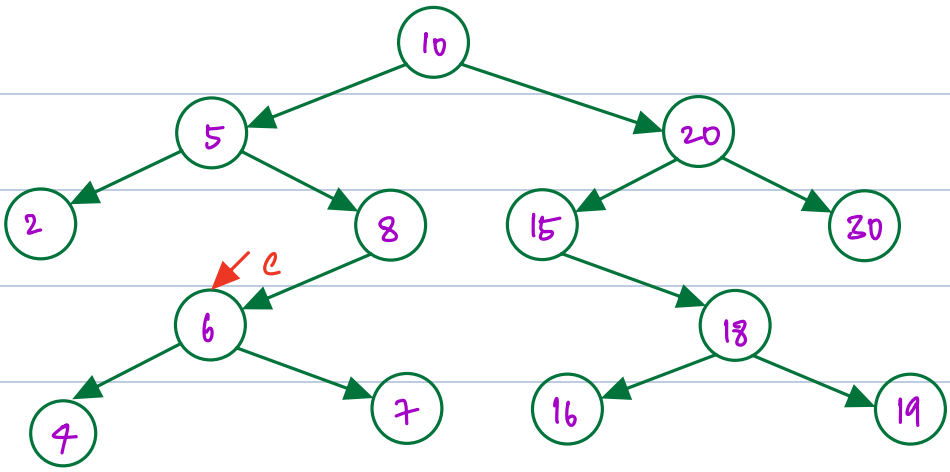
K = 7

pseudo code

```
path = [ ]
boolean  travel ( root, K) {
    if (root == null)  return false
     path. append (root)
     if  (root. data == K)  return true
     boolean  f_left = travel (root.left, K)
      if  (f_left)  return true;
     boolean  f_right = travel (root. right, K)
```

```
        if (f.right) return true
        path.removeLast();
        return false;
    }
```

T.C = O(n) S.C = O(H)

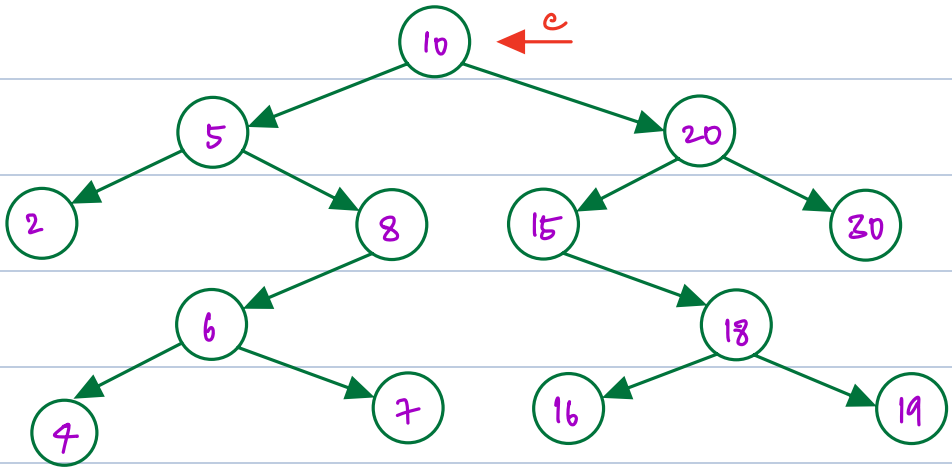


Q → Find the lowest Common Ancestor of two nodes

Ancestors

all nodes from root node to current node

Is ancestor of current node



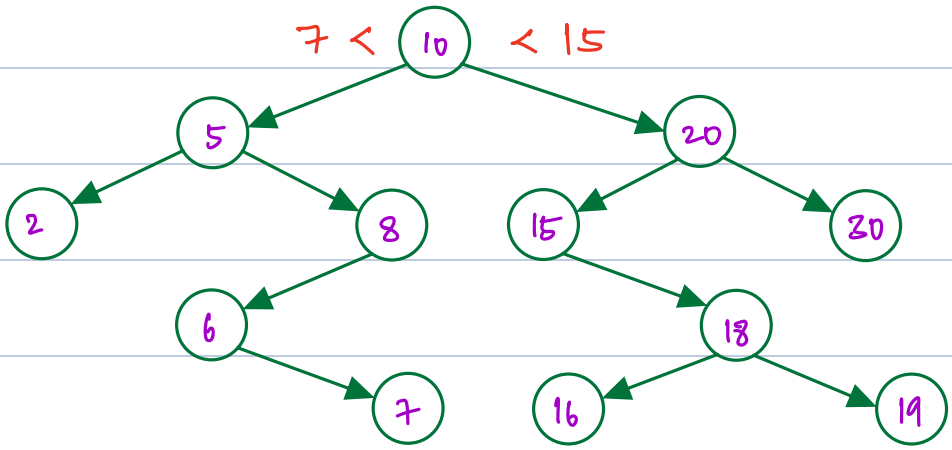
ancestors of 4 : [10, 5, 8, 6, 4]

ancestors of 2 : [10, 5, 2]

pseudo code

```
curr = root;
path_a = travel ( curr, node_a.data);
curr = root;
path_b = travel (curr, node_b.data);
```

Q → Find LCA of Binary search Trees



LCA (7, 15) = 10

LCA (2, 7) = 5

LCA (16, 30) = 20

pseudo code

```
curr = root ;    LCA = -1 ;
while ( curr != null) {
    if (curr.data > x && curr.data > y)
        curr = curr.left ;
    else if (curr.data < x && curr.data < y)
        curr = curr.right ;
    else
        return curr.data
}
```

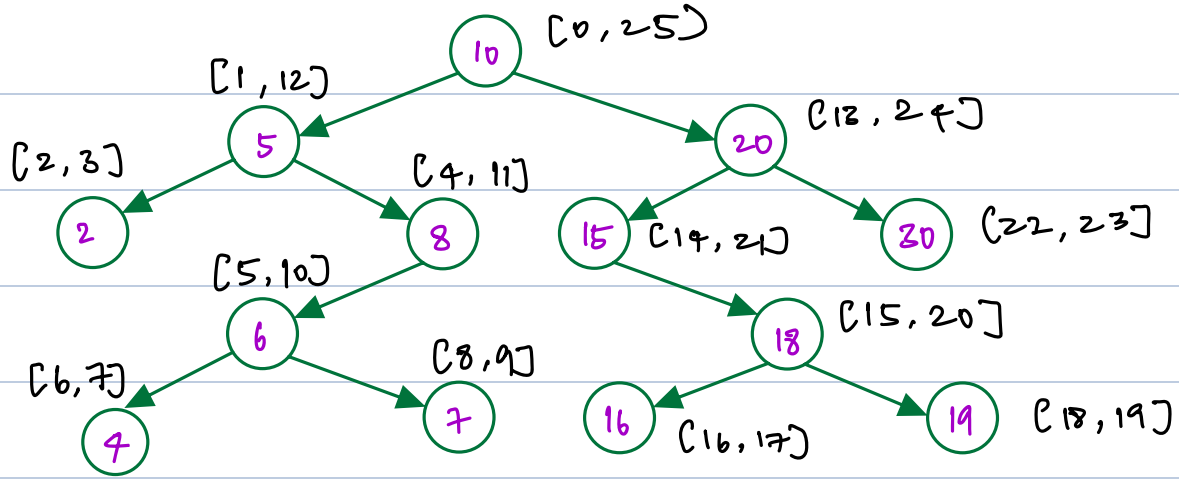
T.C = O(H) S.C = O(1)

Additional Problem

In Time / Out Time

↓ Start subtree ↓ end subtree

Binary tree



pseudo code

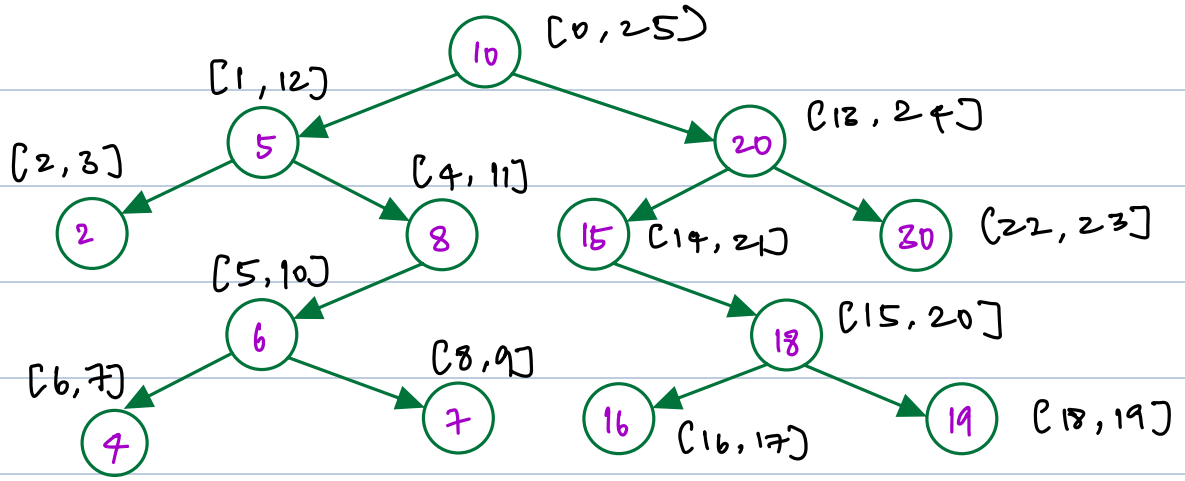
```
t=0;

void travel (root) {
    if (root == null) return
    in [root] = t;
    t += 1;
    travel (root.left);
    travel (root.right);
    out [root] = t;
    t += 1;
}
```

in / out

HashMap

< node : int >



LCA (7, 2)

example (5) as ancestor

Pn(5) < Pn(2) & Pn(7)

1 < 2 & 7

out(5) > out(2) & out(7

12 > 3 & 9

LCA (a, b)

```
curr = root;
while (curr != null) {
    If (curr.left is ancestor of a & b) {
        curr = curr.left;
    }
    else if (curr.right is ancestor of a & b){
        curr = curr.right;
    }
    else return curr;
}
```