Today's Content :-

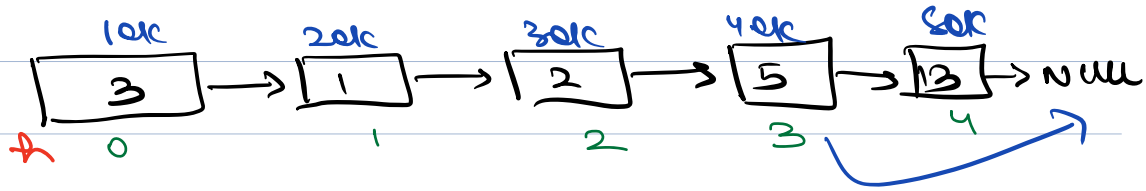
Trees → Basic Idea
Terminologies
Binary Tree

Traversal → Pre
Post
In

Size
Height.
Count nodes

→ Delete In BST → Doubt Session.

* Deleting in a LL.



head = 10ac

k = 0.

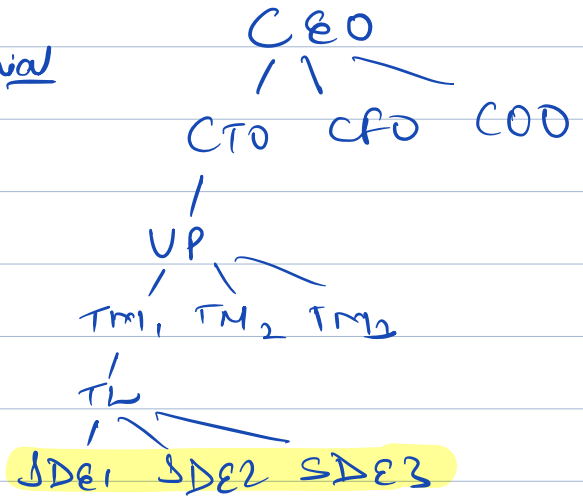
check if given index is valid or not.

```
Node deleteN(Node head, int k){
    if(k==0){
        h = h.next
        return h;
    }
    Node t = head;
    for(i=1; i<k; i++){
        t = t.next;
    }
    t.next = t.next.next
    return h
}
```

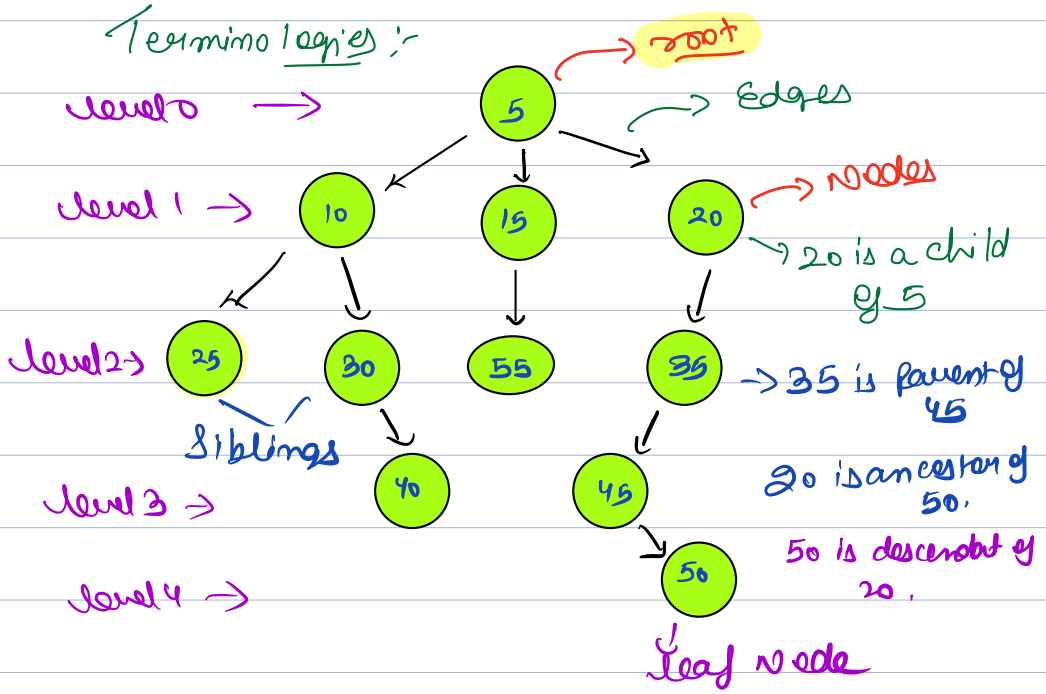
Trees → Intro :-

// arrays, ArrayList, LinkedList, Stack → Linear DS.

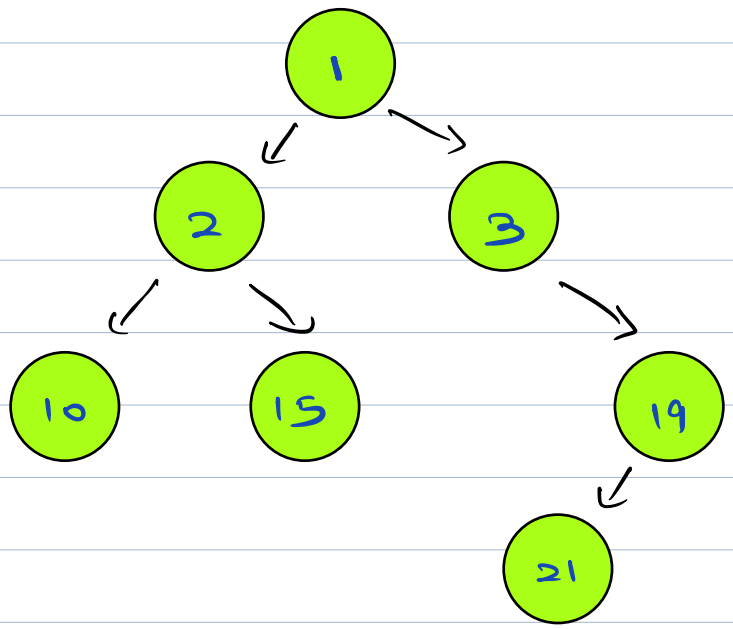
// Hierarchical



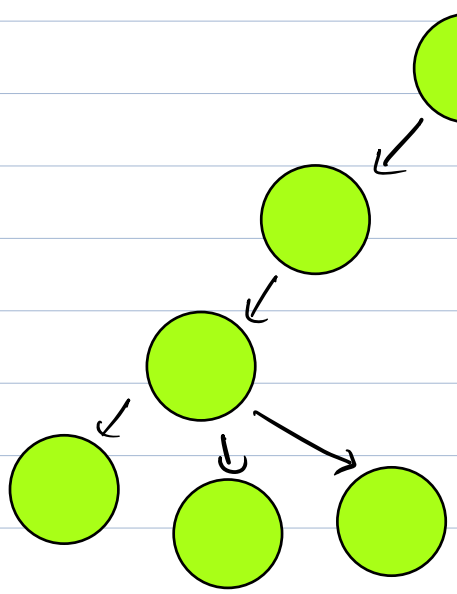
Terminologies :-



Binary Tree :- → for every node, no. of child nodes <= 2.

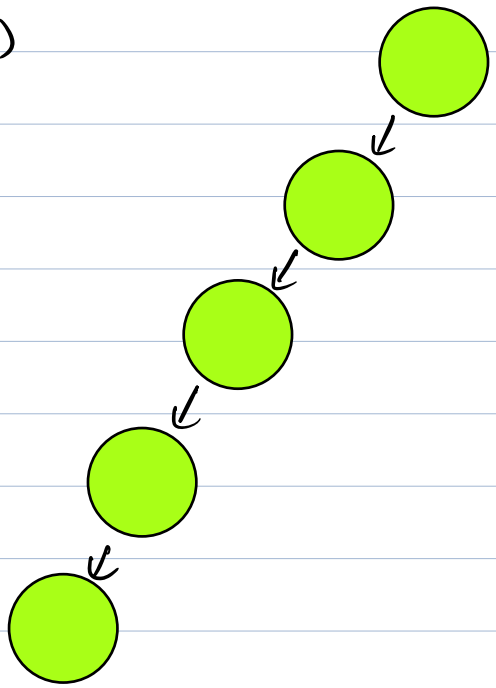


eg 2)



→ NO ✗

e.g 3)



→ B.T.

* Node Structure of a Binary Tree :-

```
class Node {
    int data
    Node next;
}
```

LL

```
class Node {
    int data;
    Node left;
    Node right;
}
```

T

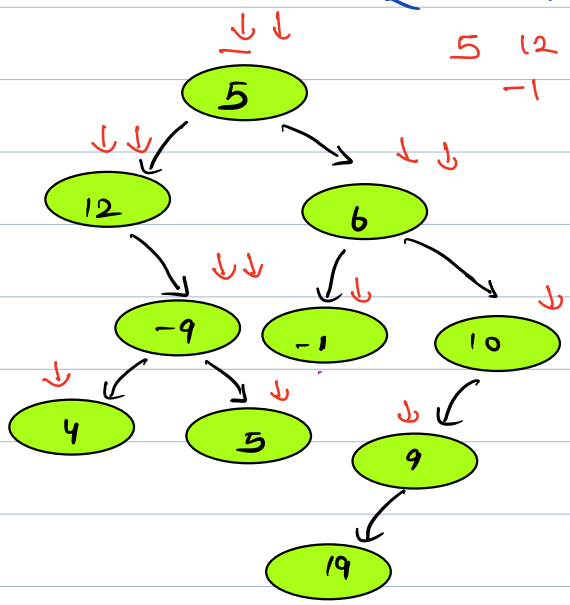
B.T. Node Struct

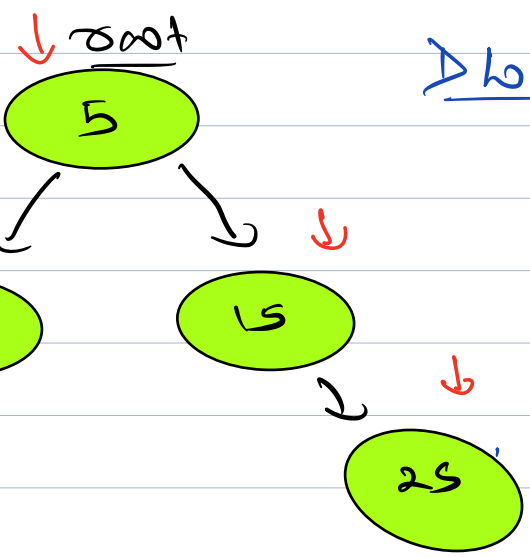
Tree Traversals :-

→ Pre Order
→ Post Order
→ In Order

1) PreOrder :- ① ② ③ D L R

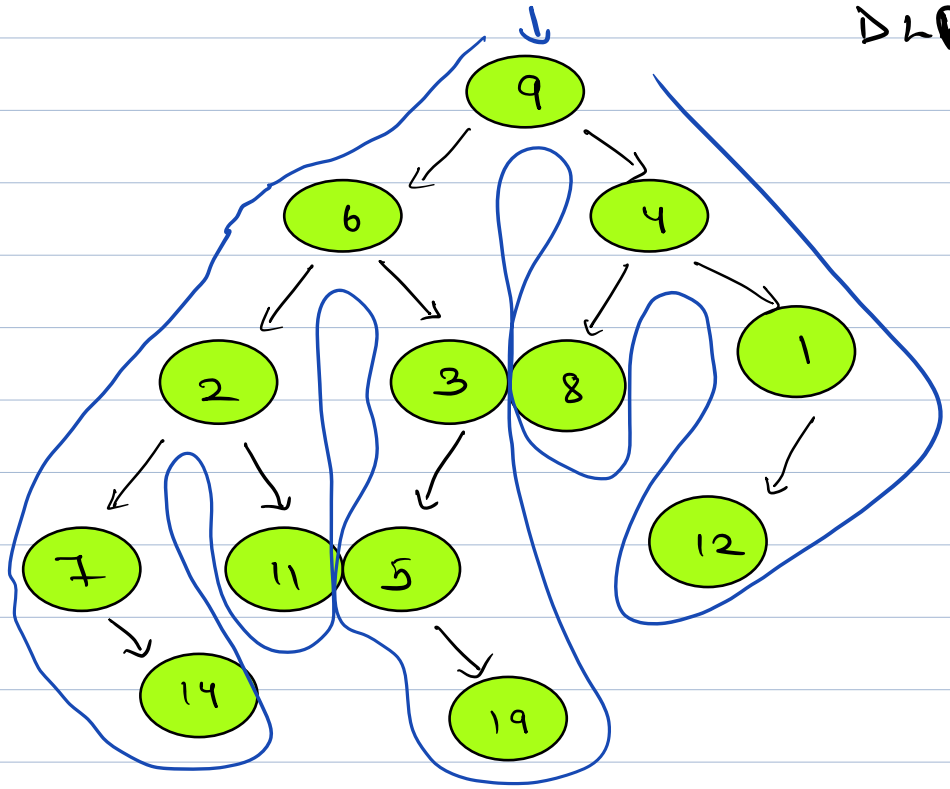
5 12 -9 4 5 6
-1 10 9 19





D L R

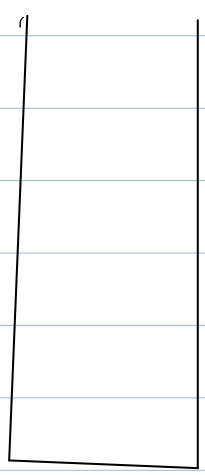
5 10 20 15 25



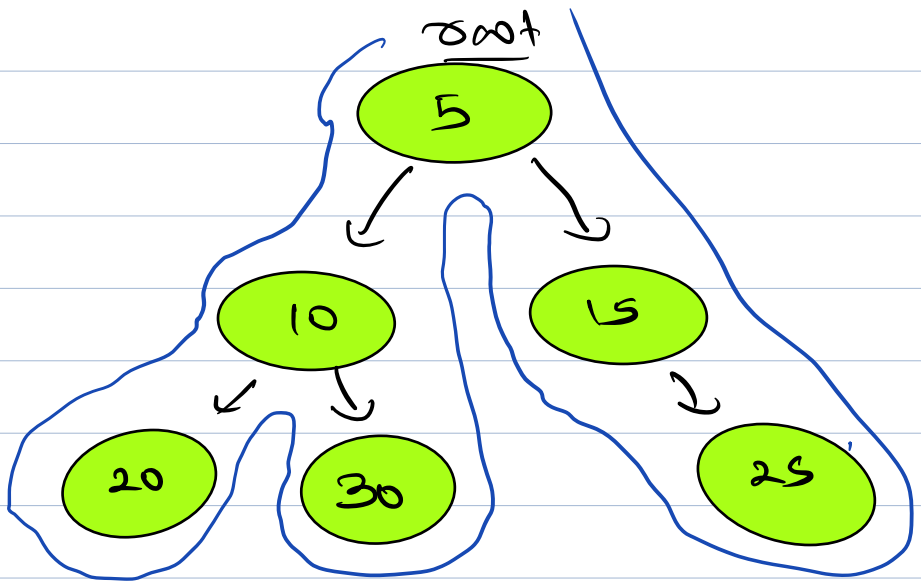
DLR

9 6 2 7 14 11 3 5 19 4 8
1 12

```
void Preorder(Node root) {
    (1)   if (root == null) {
              return;
          }
    (2)   print(root.data);
    (3)   Preorder(root.left);
    (4)   Preorder(root.right);
}
```

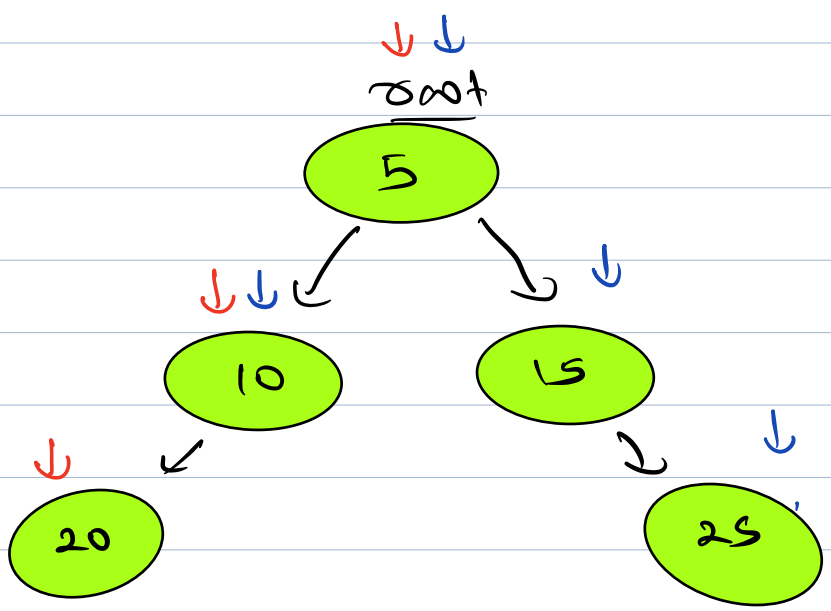


3



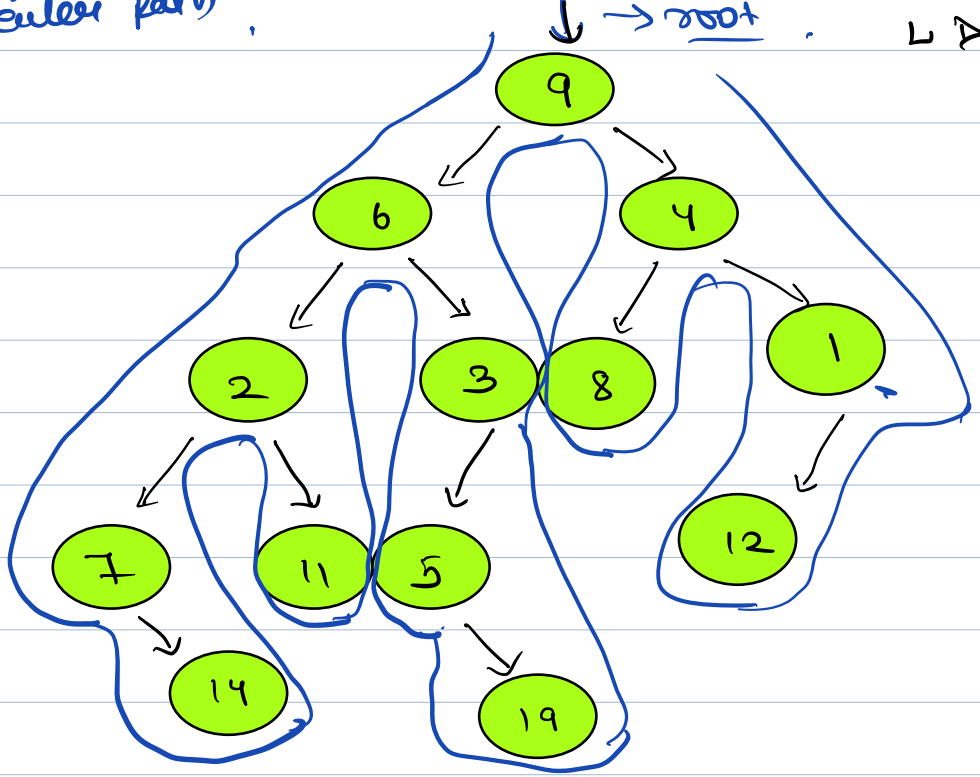
5 10 20 30 15 25

② Inorder → L D R.

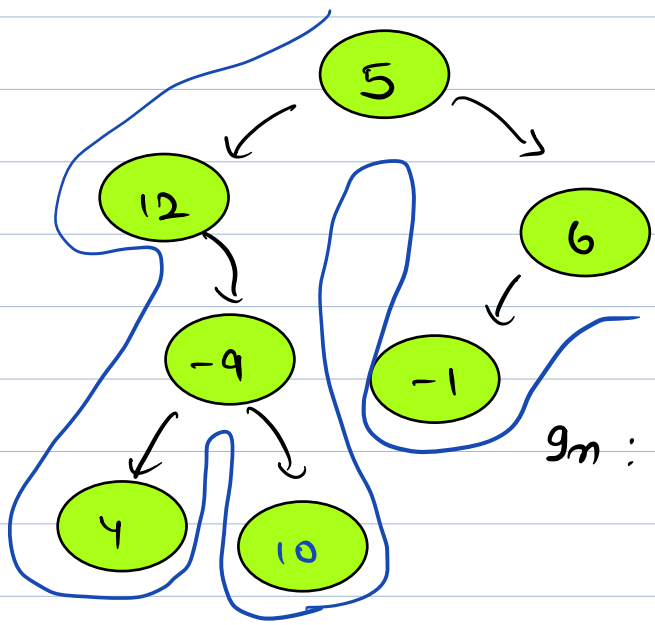


20 10 5 15 25

Euler path. ↓ → root . L D R



7, 14, 2, 11, 6, 5, 19, 3, 9, 8, 4, 12, 1



Pre :- 5, 12, -9, 4, 10, 6, -1

In : 12, 4, -9, 10, 5, -1, 6.

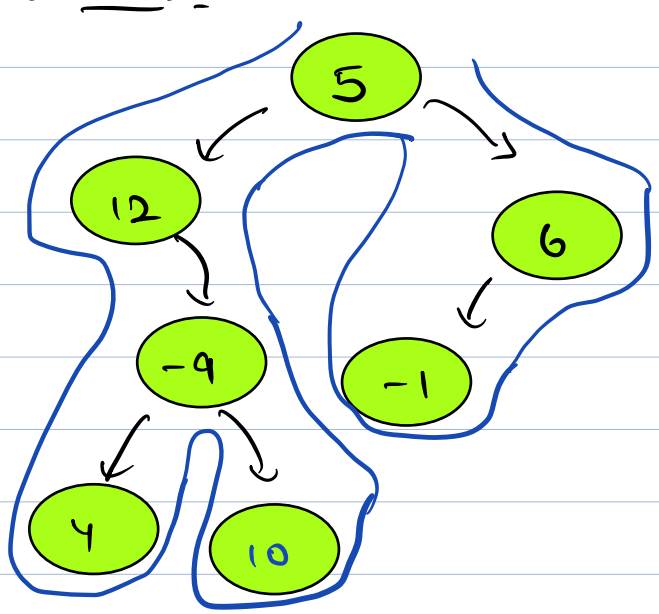
D L R

```
void Preorder(Node root) {
    (1) if (root == null) {
            return;
        }
    (2) print(root.data);
    (3) Preorder(root.left);
    (4) Preorder(root.right);
}
```

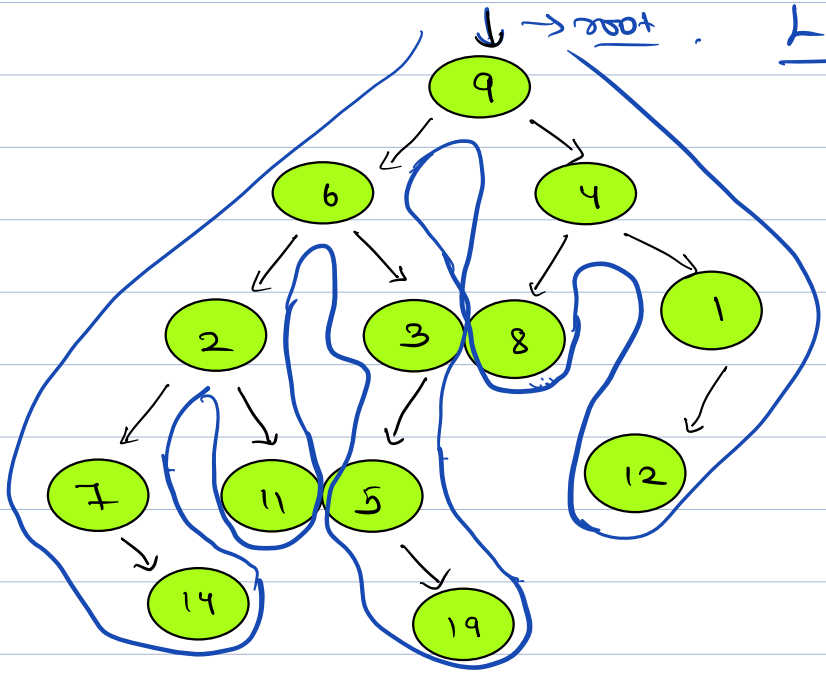
L D R

```
void Inorder(Node root) {
    if (root == null) {
        return
    }
    Inorder(root.left);
    Print(root.data);
    Inorder(root.right);
}
```

* Post Order :- L R D.



4 10 -9 12 -1 6 5



LRD ,,

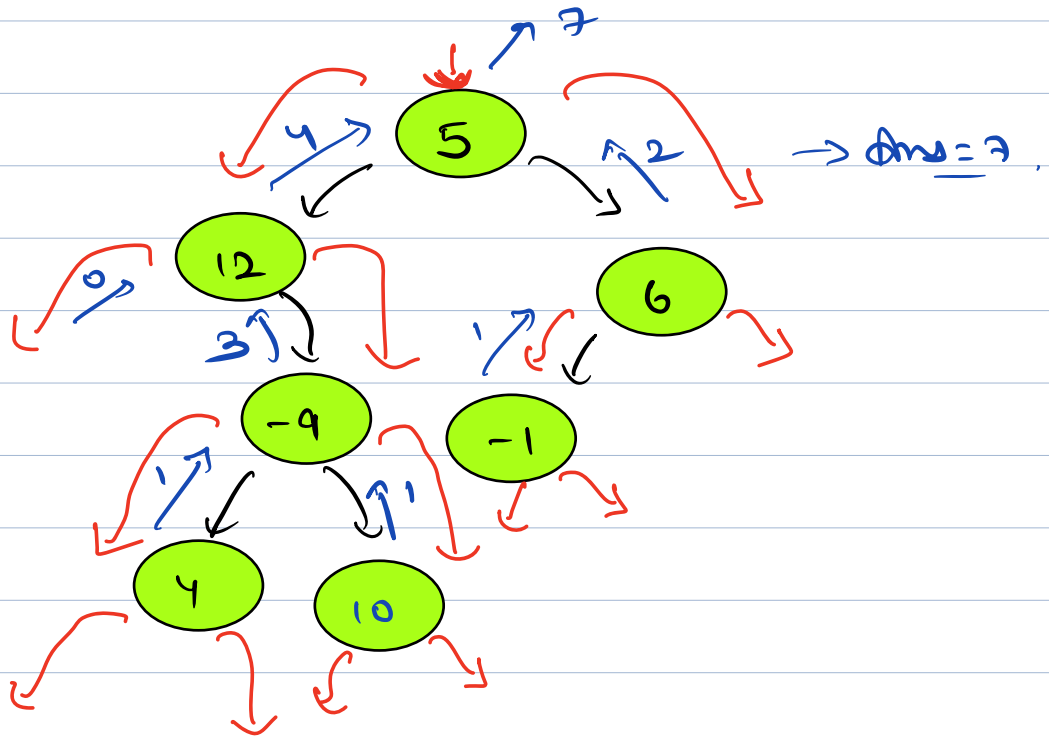
14 7 11 2 19 5 3 6 8 12 1 4 9

L D R

```
void PostOrder(Node root) {
    if (root == null) {
        return
    }
    PostOrder(root.left);
    PostOrder(root.right);
    Print(root.data);
}
```

L R D

* Calculate size of tree.

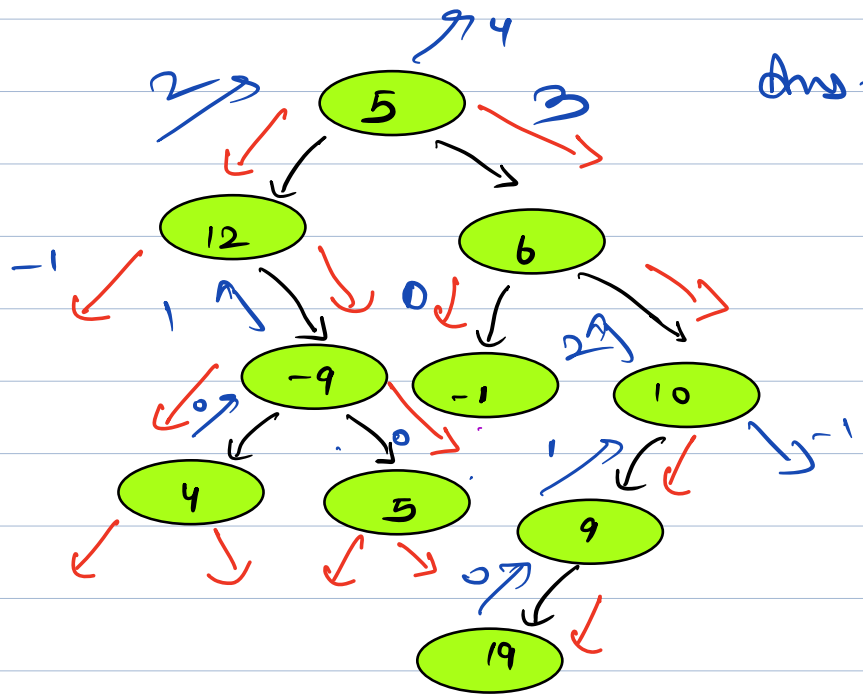



```
int size (Node root) {
    if (root == null) { return 0; }

    int lsize = size (root.left);
    int rsize = size (root.right);

    return lsize + rsize + 1;
}
```

Ques) Calculate height of tree :-

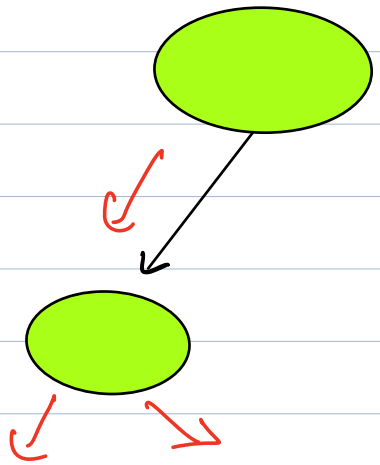


Ans = 4.

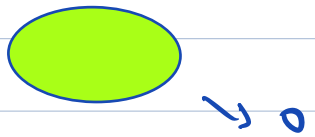
```
int height (Node root) {
    if (root == null) { return -1;

    int l = height (root.left)
    int r = height (root.right)
    return max (l, r) + 1

}
```



→ 1



→ 0

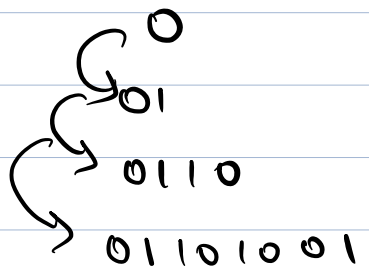
null

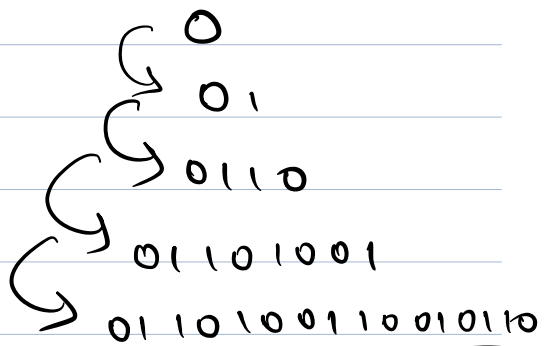
Arrays
Bit Manip
Hashing
Recursion

→ 70-80%.

↳ revised.

Attend class reg.
Revise Everyday
Solve atleast 2 HIVE.





$0 \to 01$

$1 \to 10$

Analog By

3rd