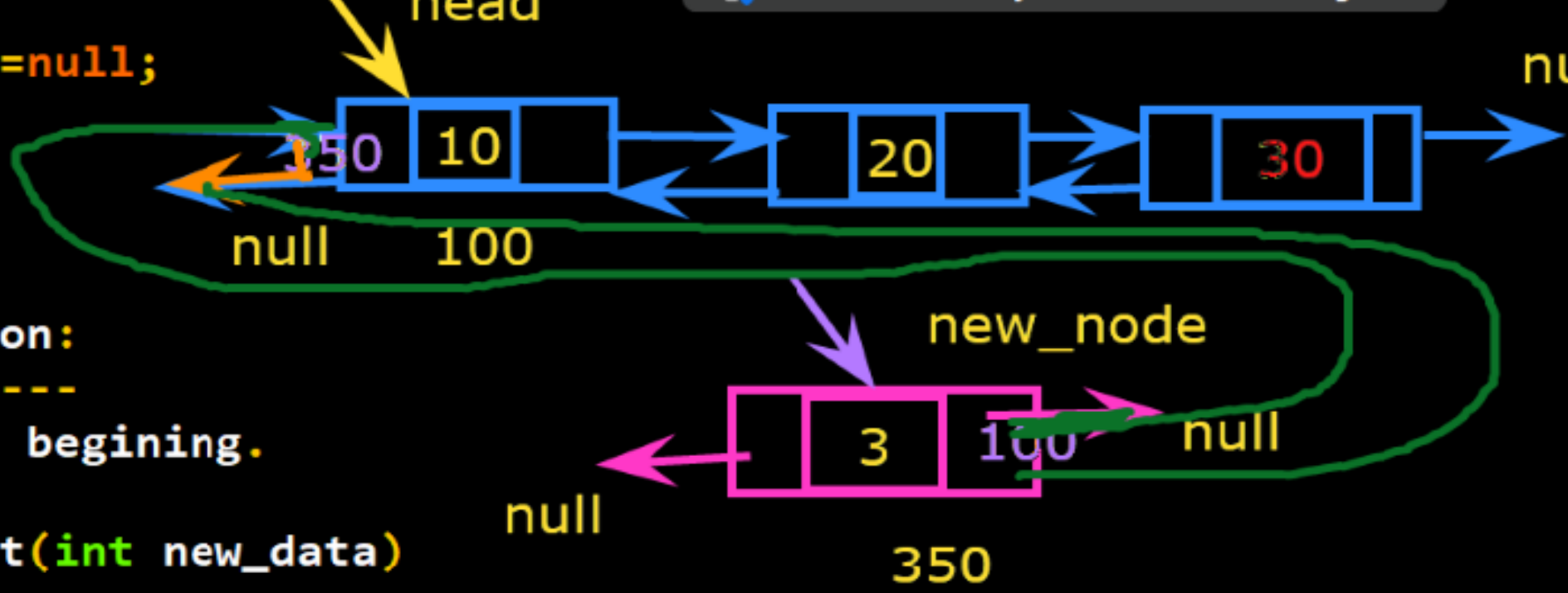
Sep22 : Day 8

Kiran Waghmare

CDAC Mumbai

Doubly Linked List:

```
        data = d;
        prev=next=null;
    }

}
```



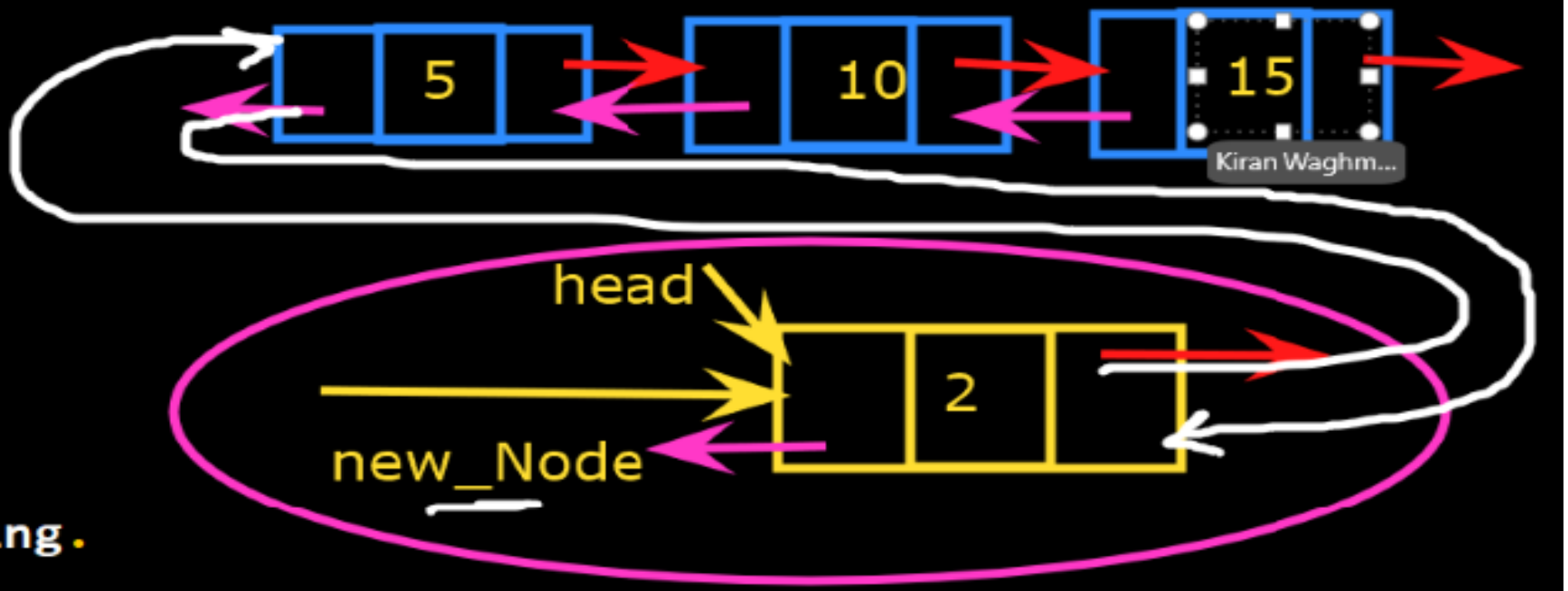
Insertion Operation:

---

case 1: Insert at begining.

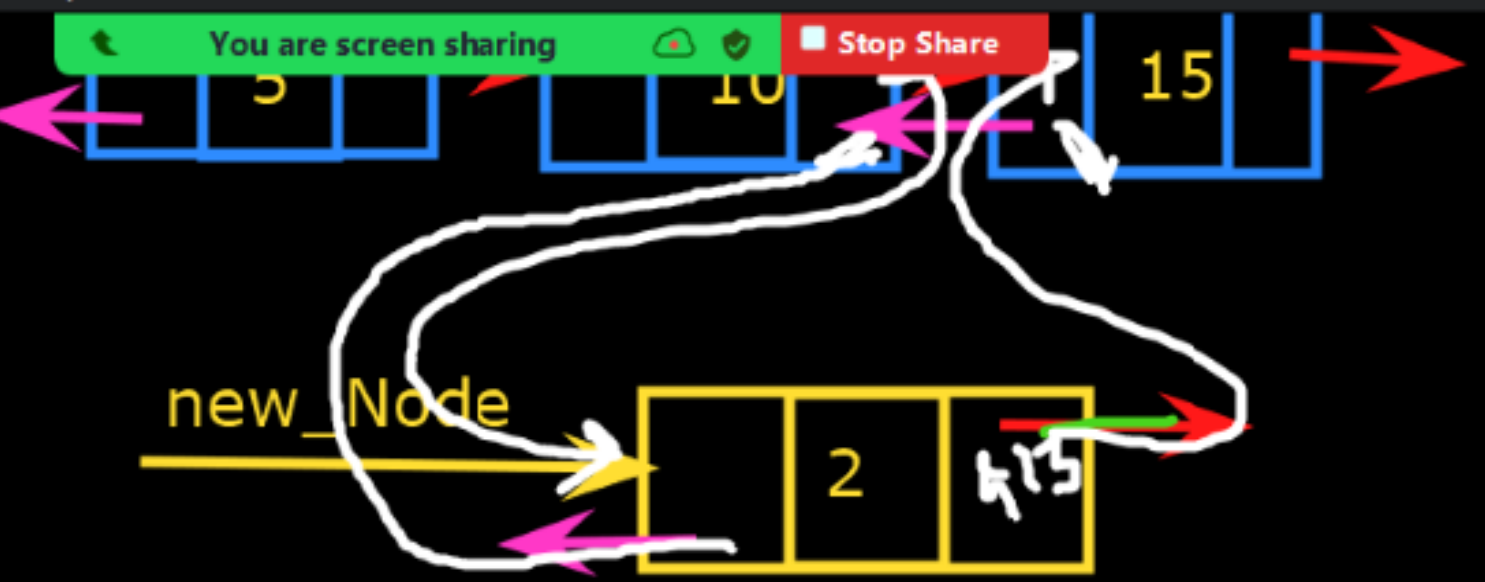
```
static void insert(int new_data)
{
    Node new_Node = new Node(new_data);
    new_Node.next = head;
    new_Node.prev = null;
    if(head != null)
        head.prev = new_node;
    head = new_Node;

}
```

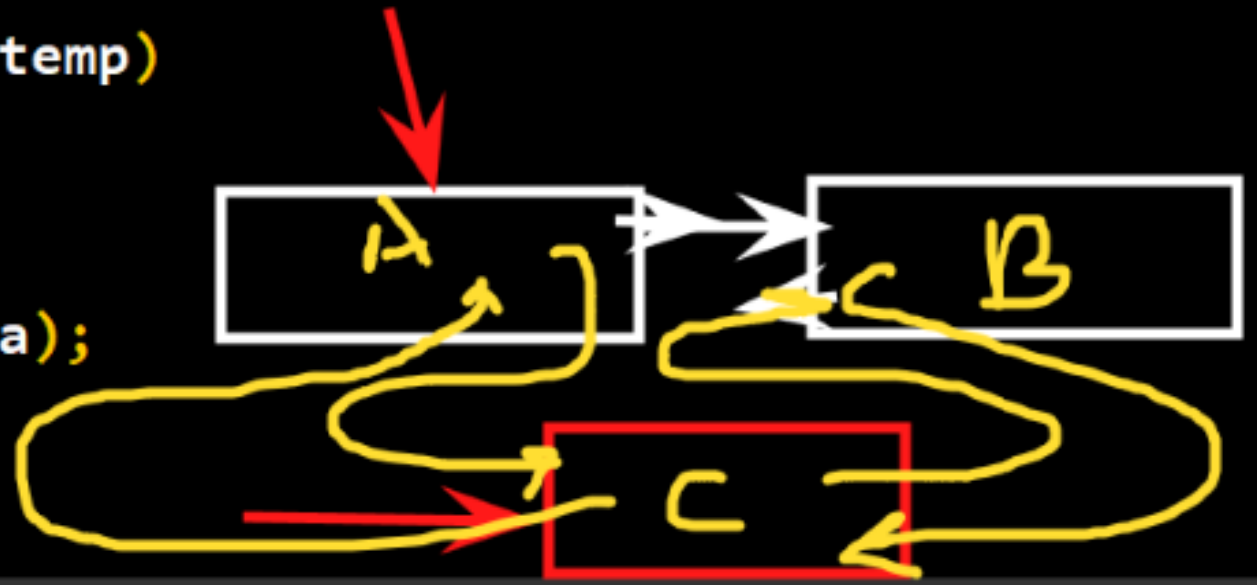
```
    Node(int d)
    {
        data = d;
        prev=next=null;
    }

}
```

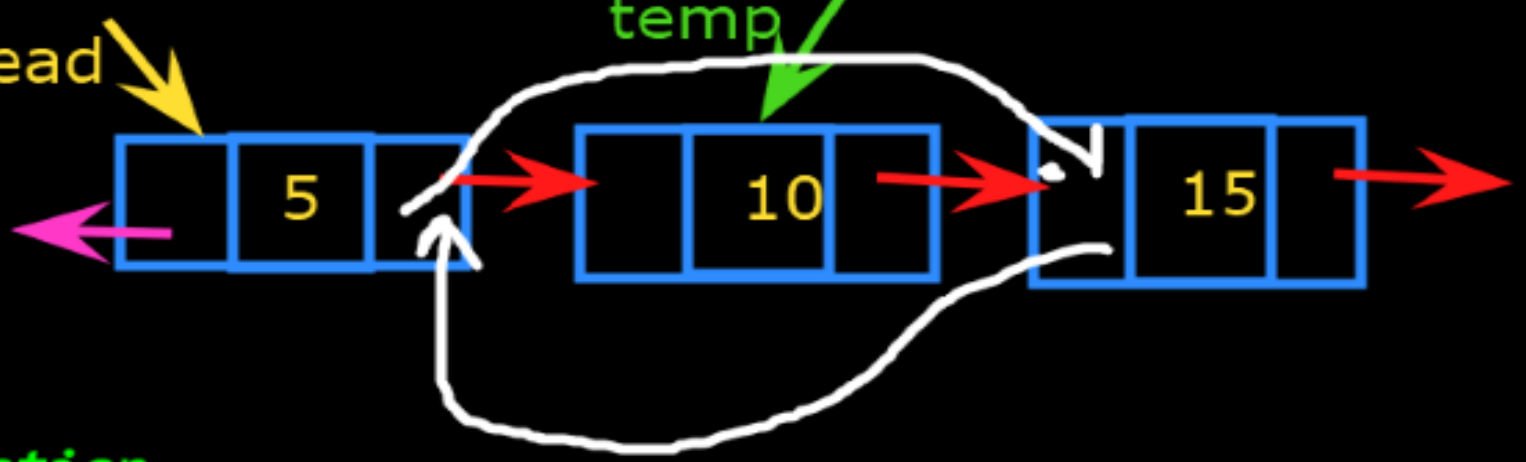



```
Insertion Operation:
---------------------
case 1: Insert at begining.

static void insert(int new_data)
{
    Node new_Node = new Node(new_data);
    new_Node.next = head;
    new_Node.prev = null;
    if(head != null)
        head.prev = new_node;
    head = new_Node;

}
```

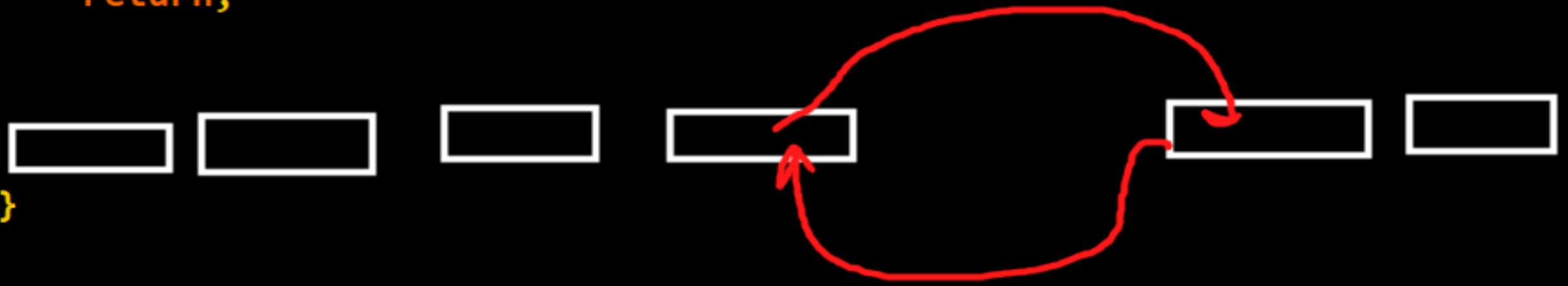
```
    }
    while(temp.next != null)
    {
        temp=temp.next;
    }
    temp.next = new_Node;
    new_Node.prev = temp;

}


void insertAfter(int new_data, Node temp)
{
    if(head==null)
        return;
    Node new_Node = new Node(new_data);
    new_Node.next = temp.next;
    temp.next = new_Node;
    new_Node.prev = temp;
new_Node.next.prev = new_Node;
```

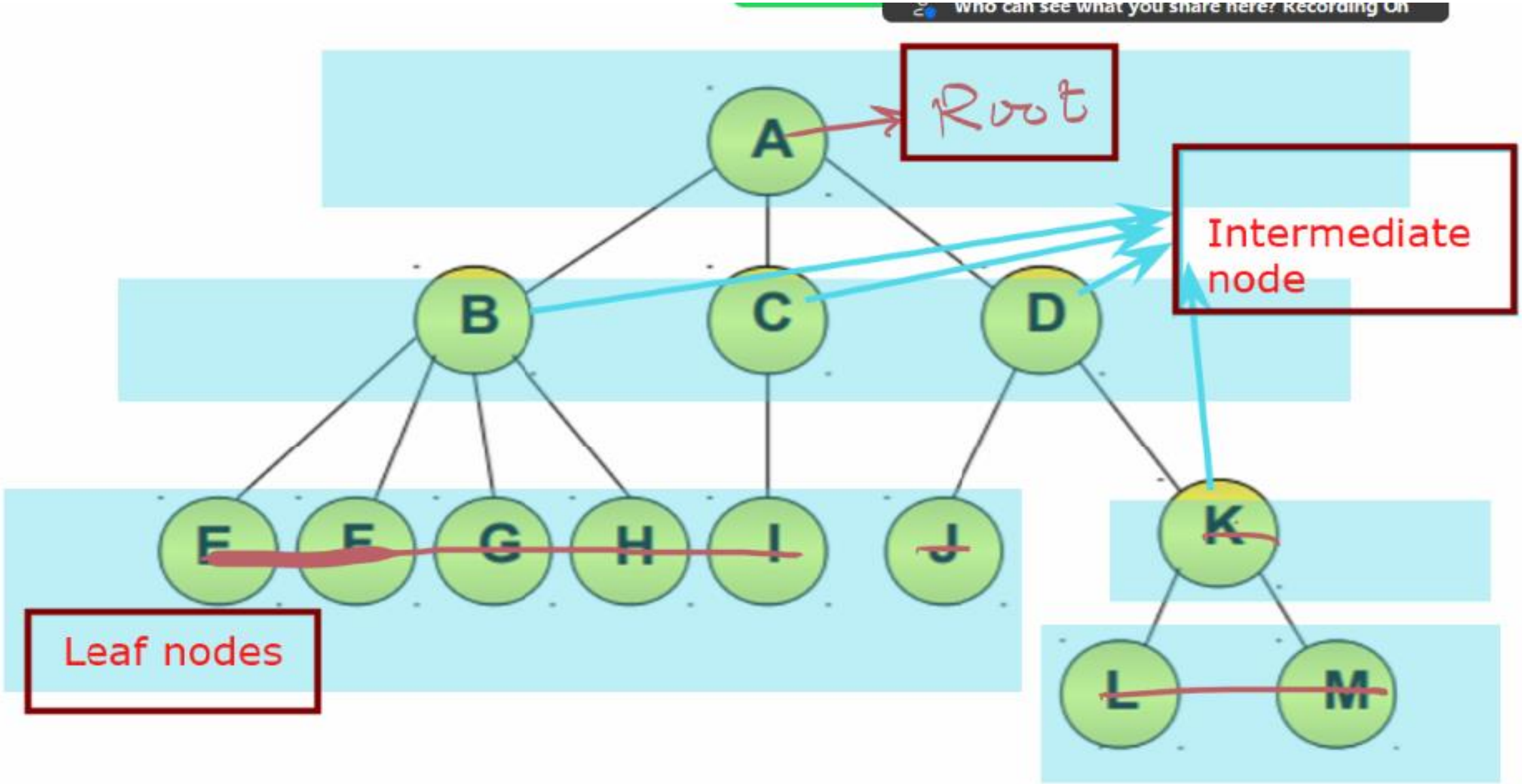


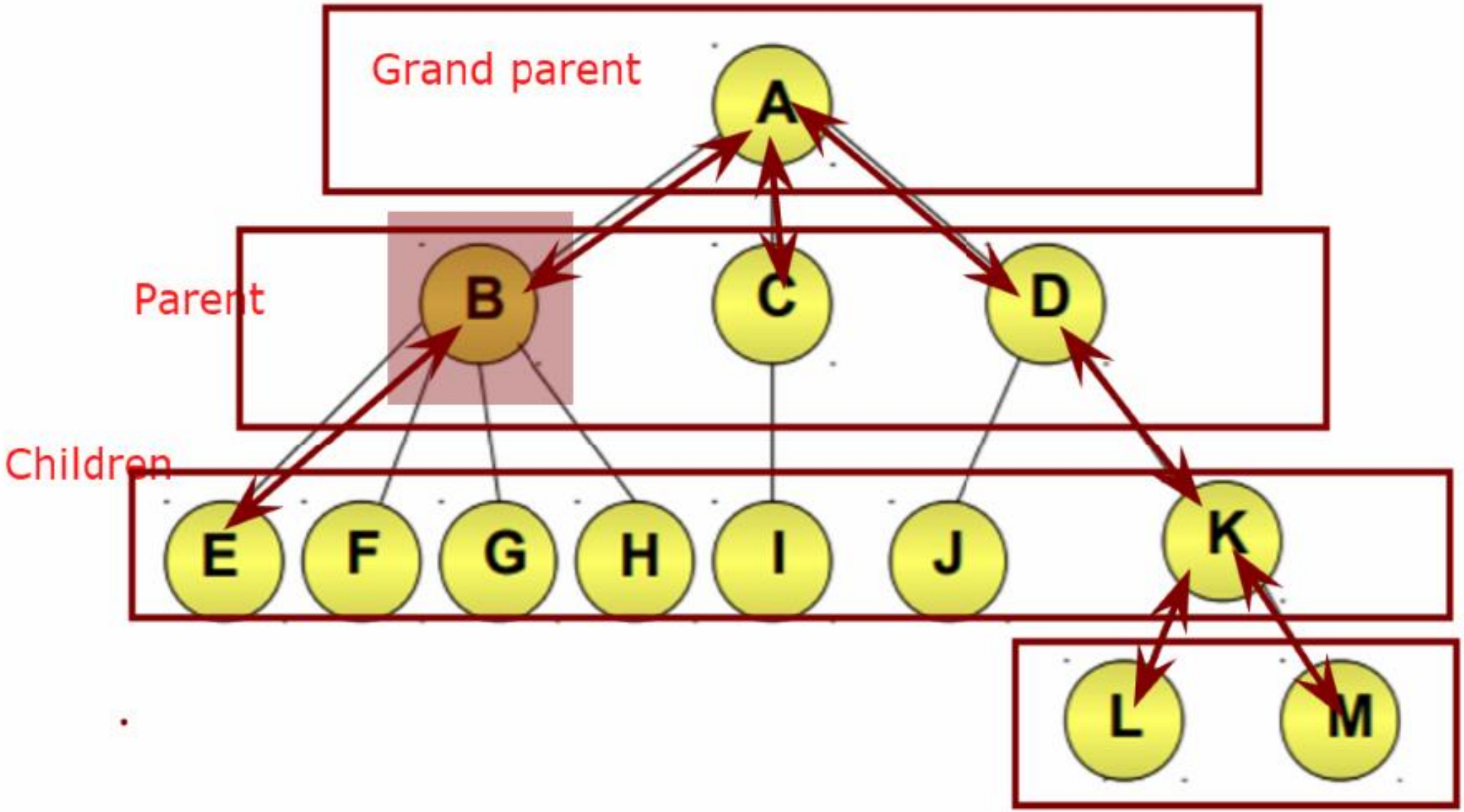





```
    //case 1: head deletion
    if(head == temp)
    {
        head = temp.next;
        //head.prev = null;
    }
    //case 2 and case 3 : deletion
    //Last node in the lis ???
    if(temp.next != null)
        temp.next.prev = temp.prev;
    if(temp.prev != null)
        temp.prev.next = temp.next;

    return;
}
```




Who can see what you share here? Recording On
A
Root
B
C
D
Intermediate node
E
F
G
H
I
J
K
L
M
Leaf nodes


Grand parent
A
Parent
B
C
D
Children
E
F
G
H
I
J
K
L
M

Degree of a node

A: 3

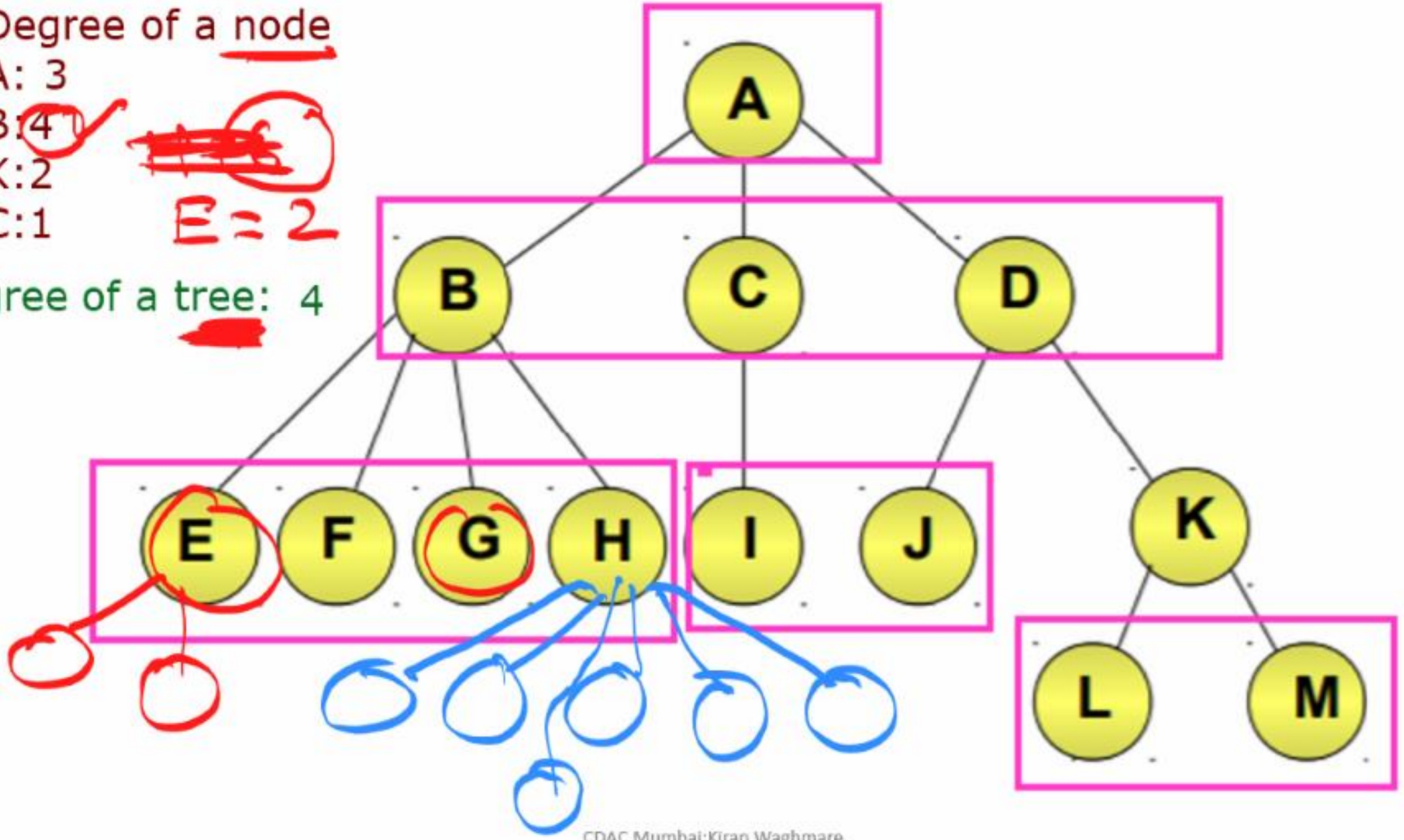
B: 4

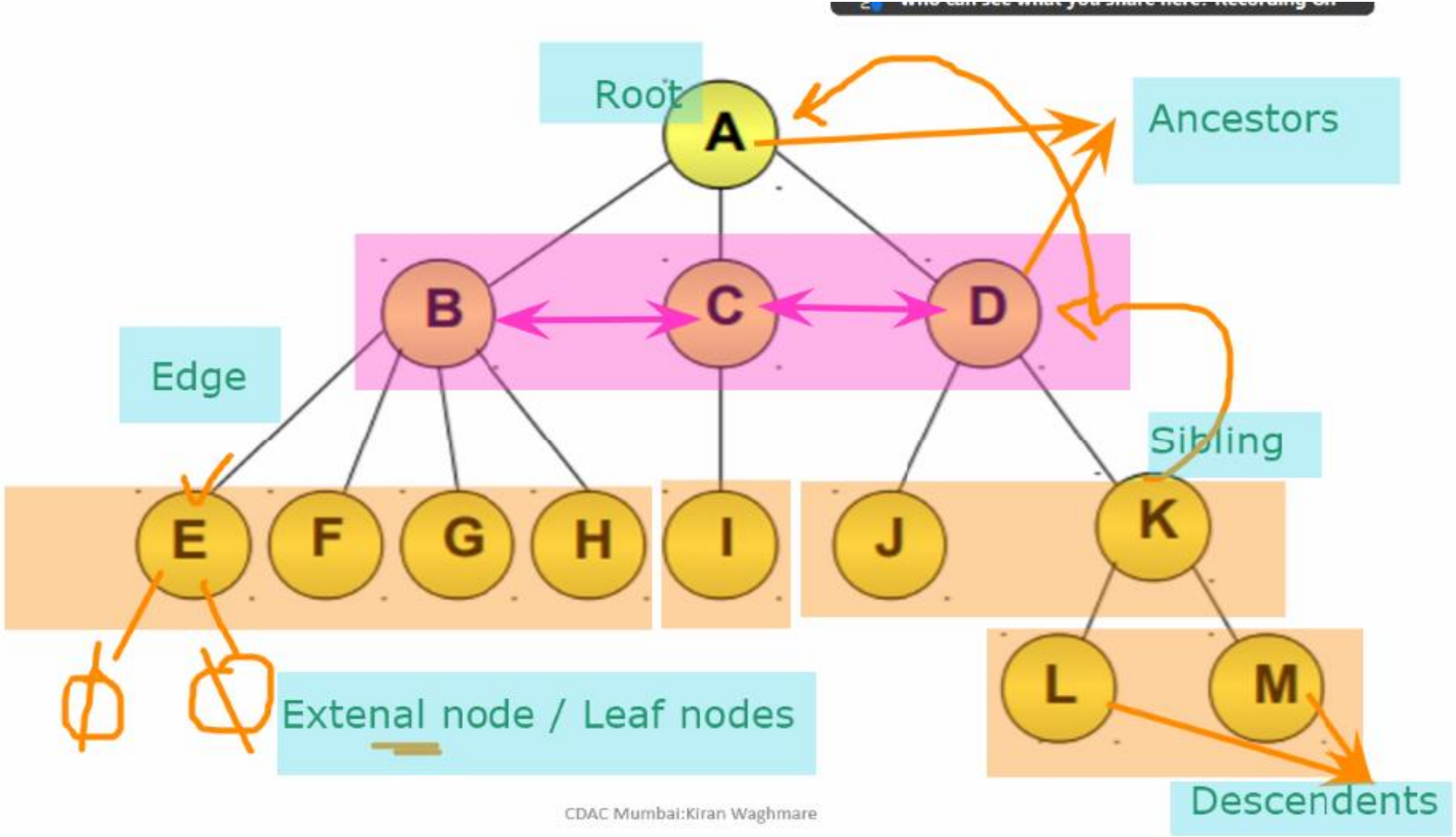
K: 2

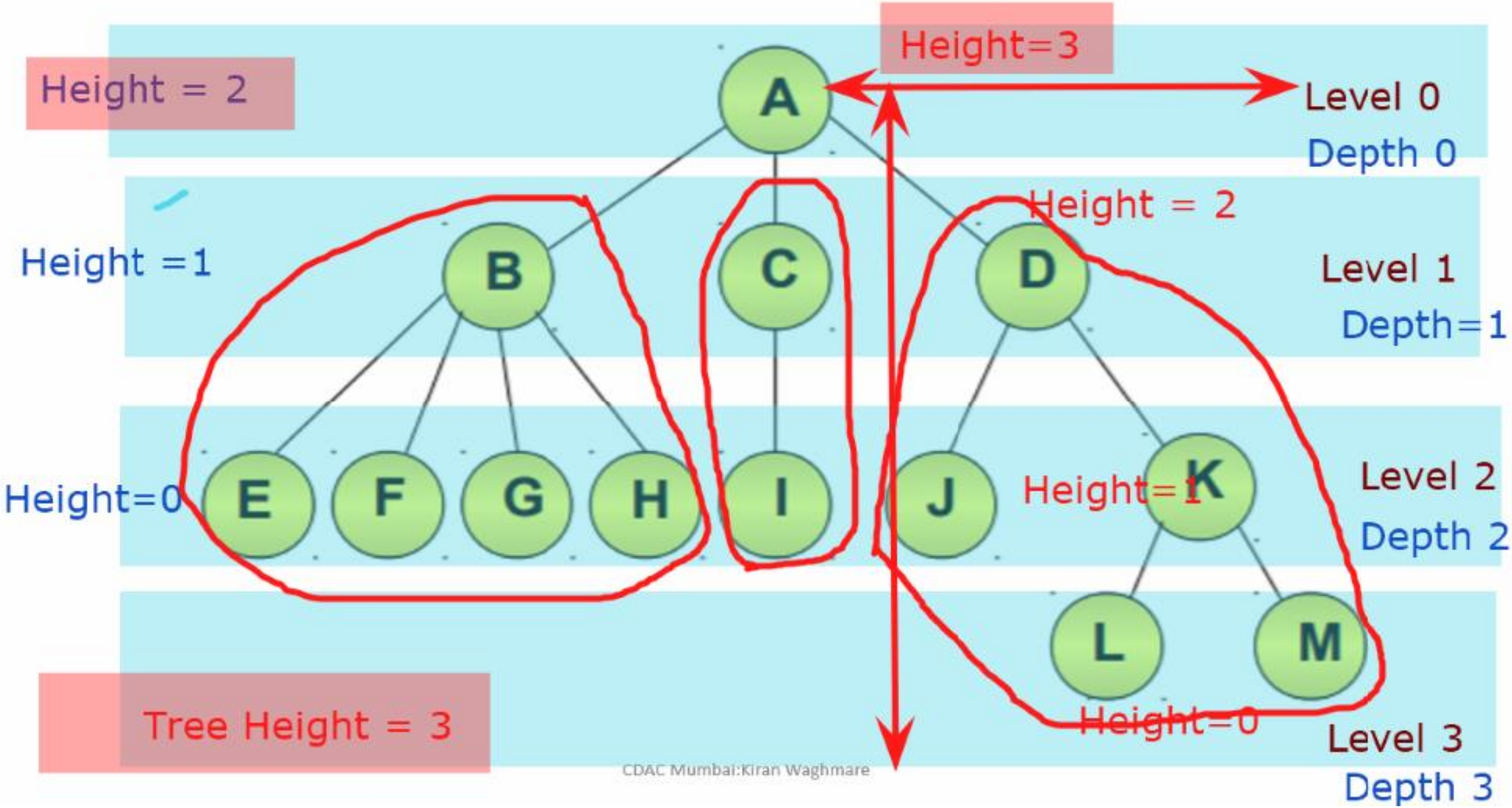
C: 1

E = 2

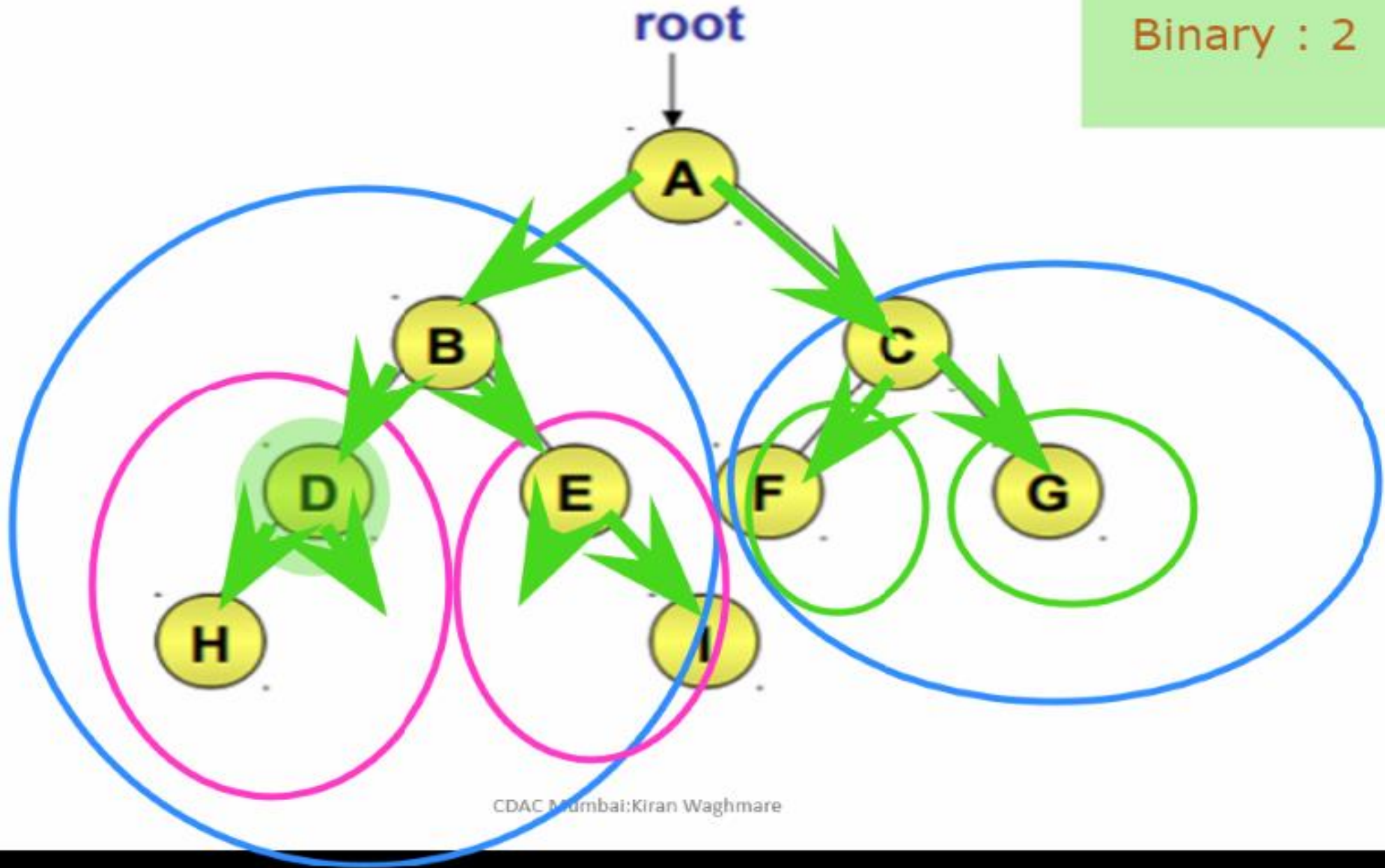
Degree of a tree: 4




Root
A
Ancestors
B
C
D
Edge
Sibling
E
F
G
H
I
J
K
L
M
Extenal node / Leaf nodes
Descendents
CDAC Mumbai:Kiran Waghmare

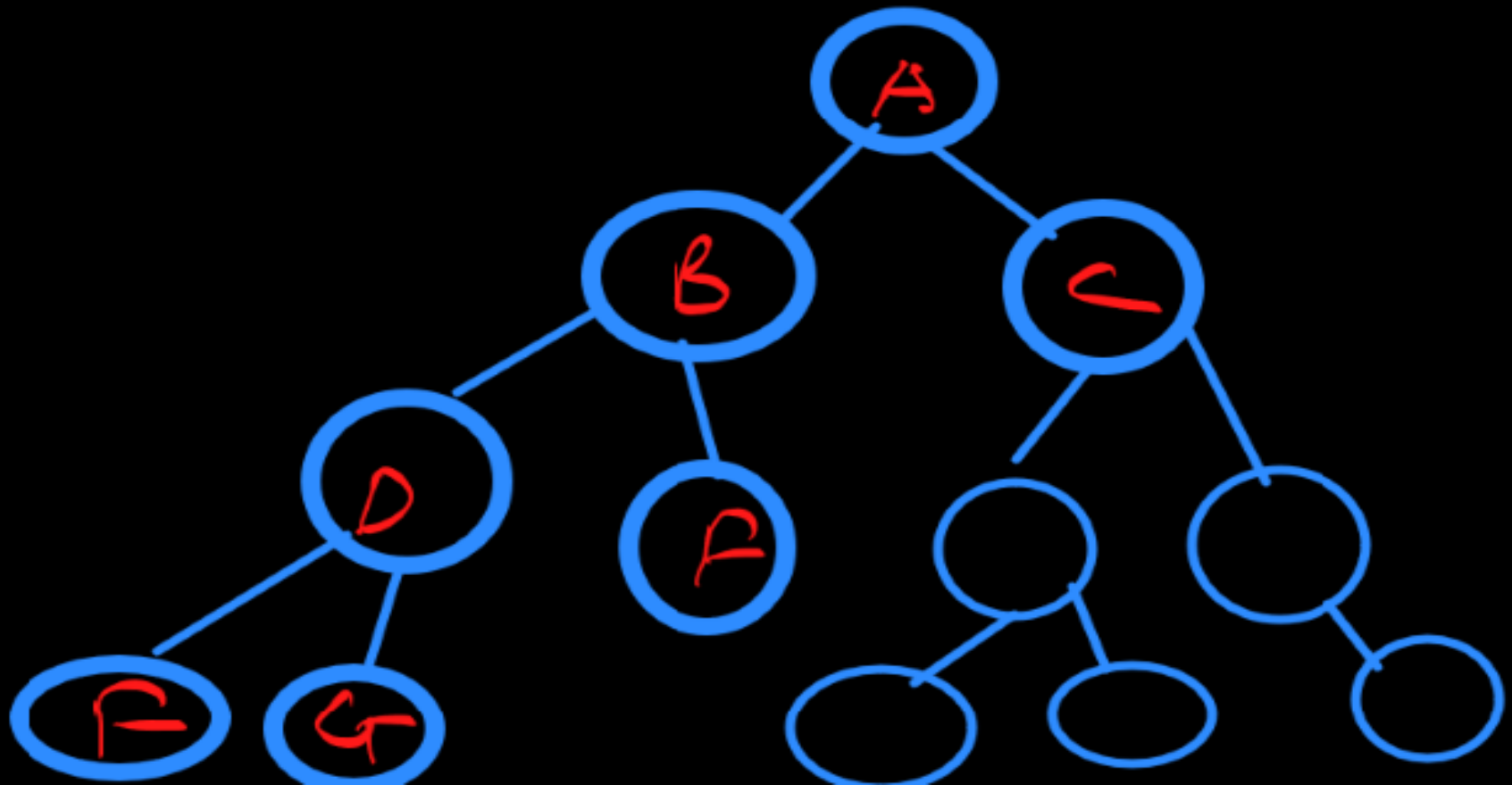


Binary : 2



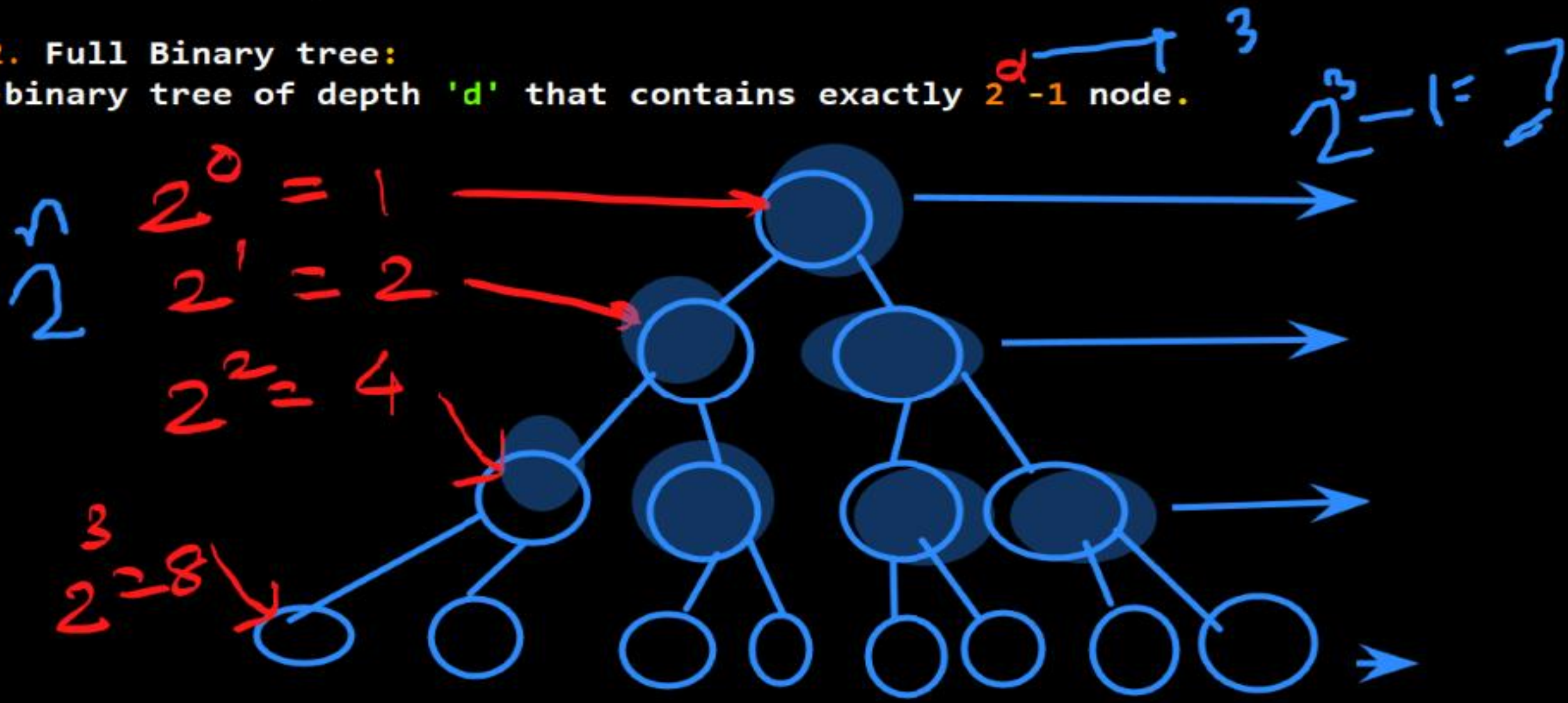
Types:
1. Strictly Binary tree
2. Full Binary tree
3. Complete binary tree.

1. Strictly Binary tree:
- every node, except for leaf nodes,
  has non-empty left and right children.
  -0-2:strictly



-0-2:strictly

2. Full Binary tree:
-binary tree of depth 'd' that contains exactly $2^d - 1$ node.

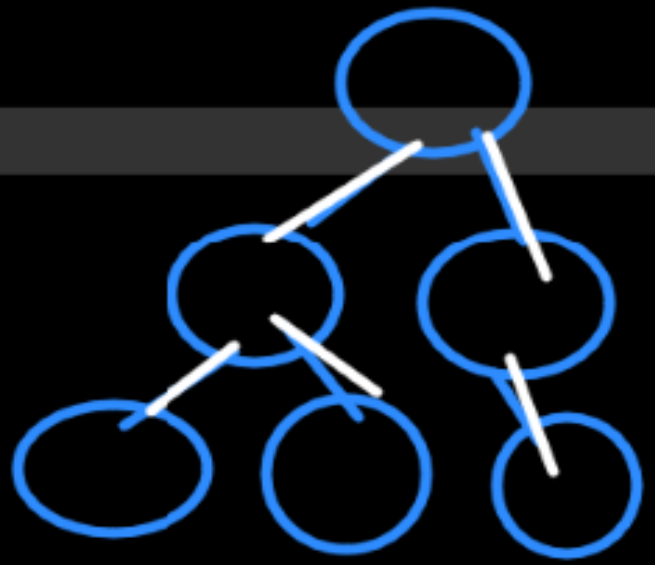


-0-2:strictly

2. Full Binary tree:
-A binary tree in which every node has 2 children except leaf nodes is known as full binary tree.
-binary tree of depth 'd' that contains exactly 2 -1 node.

3. Complete binary tree.

4. InComplete binary tree.



Complete binary tree

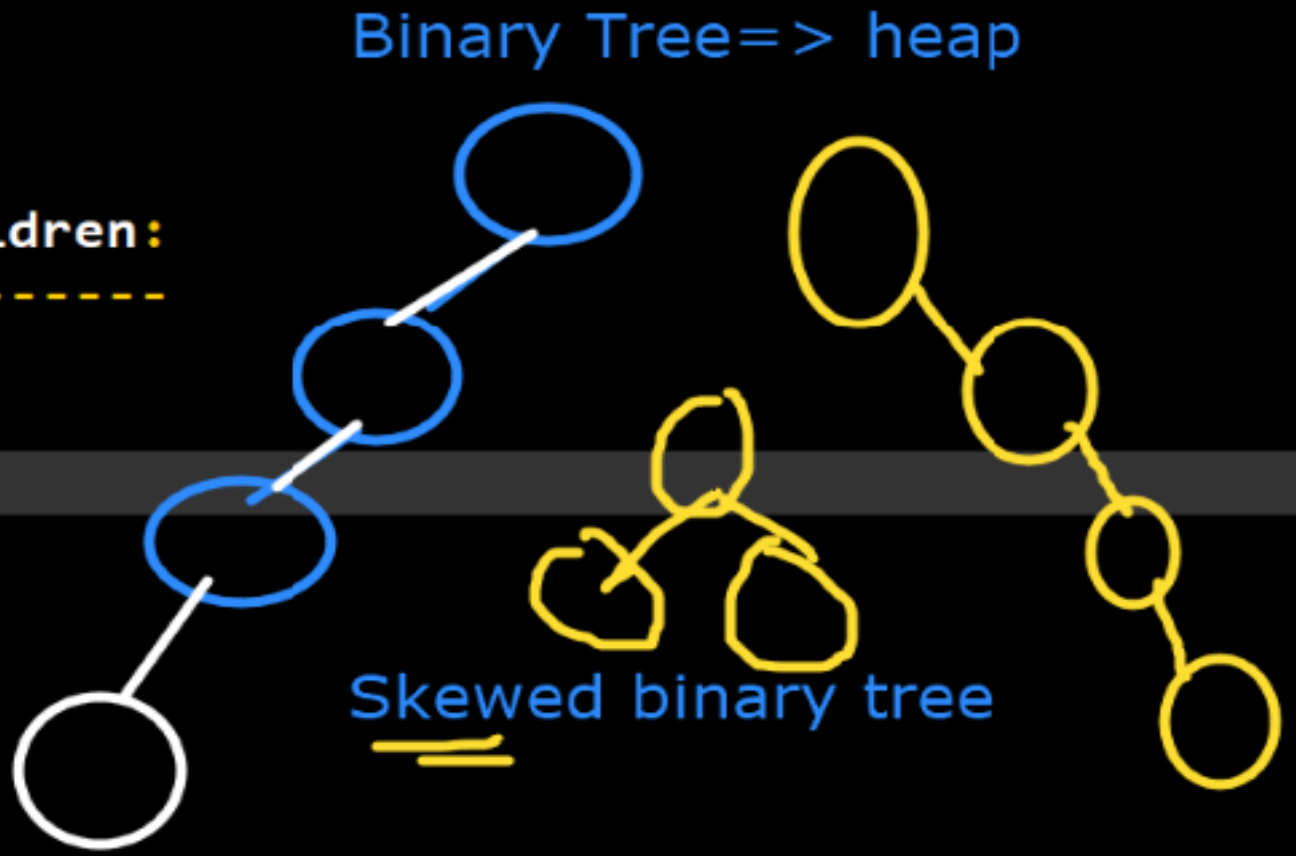
3. Complete binary tree.
-binary tree with n nodes and depth d whose nodes corresponds to the nodes numbereded to the nodes numbered from 0 to n-1 in the full binary tree of dep

4. InComplete binary tree.

Relationship between parent and children:

---

parent = i
Left child = 2*i
Right child = 2*i+1



3. Complete binary tree.

-binary tree with n nodes and depth d whose nodes corresponds to the nodes numbereded to the nodes numbered from 0 to n-1 in the full binary tree of dep

4. InComplete binary tree.

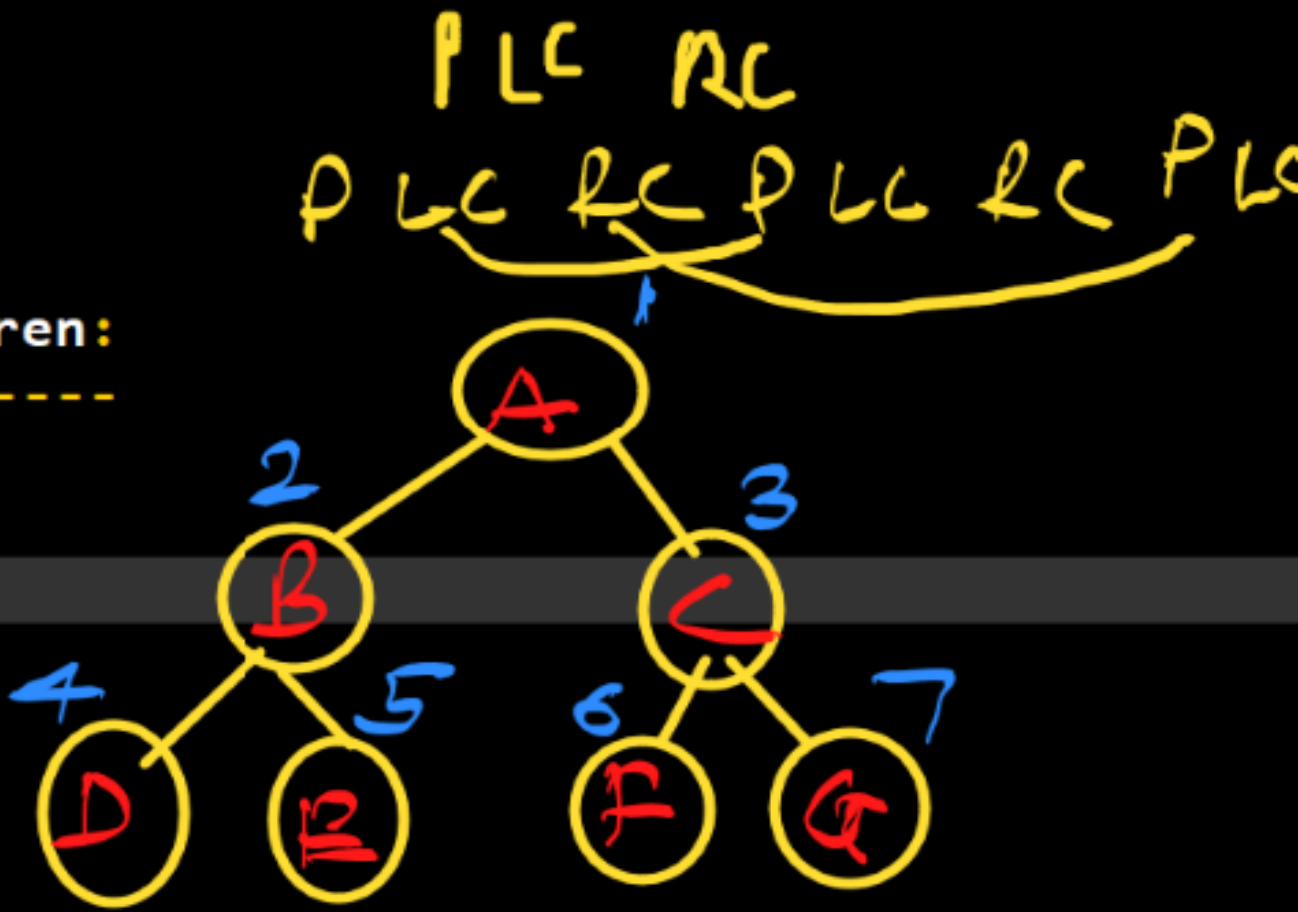
Relationship between parent and children:

---

parent = i=0

Left child = 2*i = 0

Right child = 2*i+1 =0 + 1



Relationship between parent and children:

---

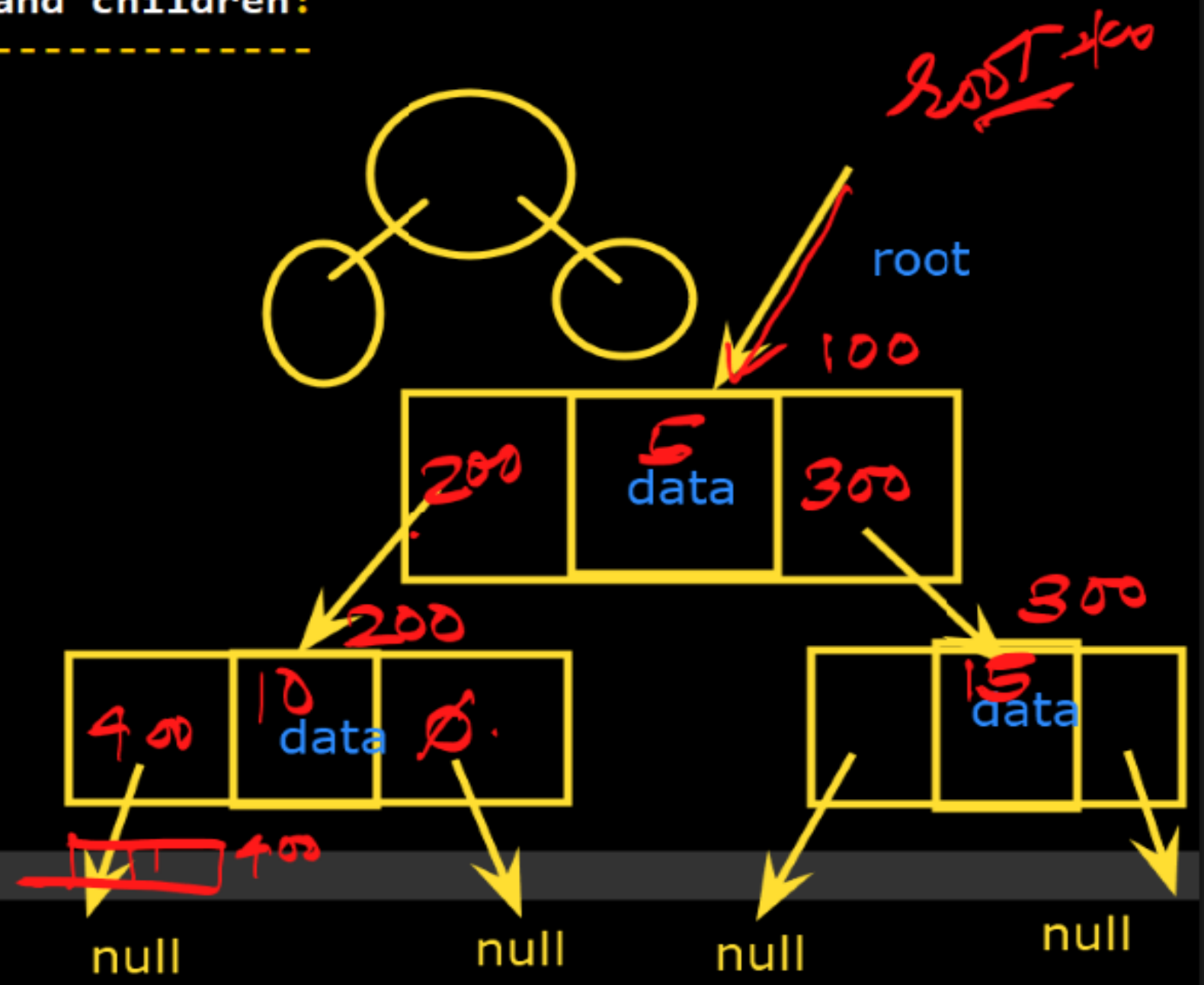
parent = i=0

Left child = 2*i = 0

Right child = 2*i+1 =0 + 1

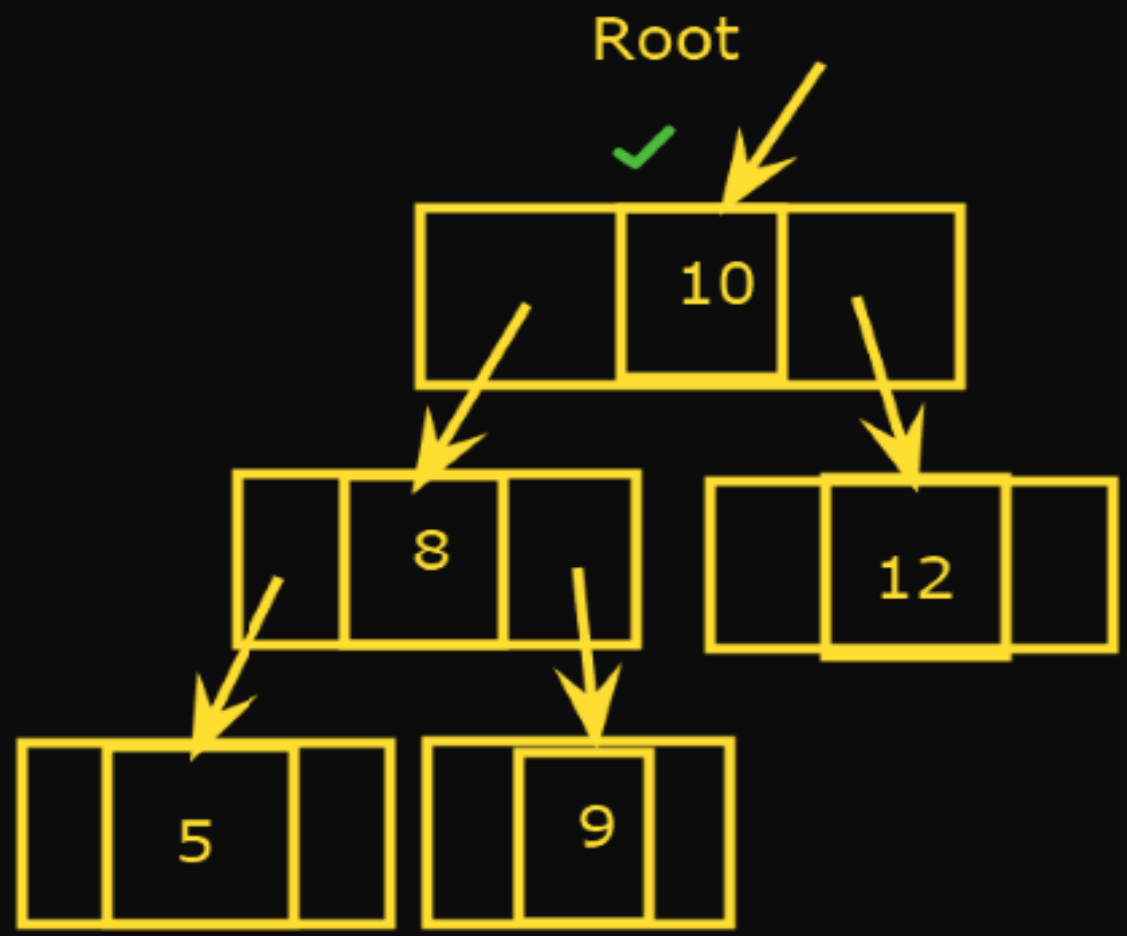
Node structure

```
class Node{

    int data;
    Node left, right;

    Node(int d)
    {
        data=d;
        left = right = null;
    }
}
```



```
}
void postO
{
    printf
}
    public
    {
        BTr
        roo
        roo
        roo
        roo
        roo

        Sys
        t1.
        Sys
        t1.
        Sys
        t1.
```

```
Microsoft Windows [Version 10.0.22000.1
(c) Microsoft Corporation. All rights reserved.

C:\Test>javac BTree.java

C:\Test>java BTree

C:\Test>javac BTree.java

C:\Test>java BTree
Preorder.....
10
8
5
9
12
Inorder.....
5
8
9
10
12
Postorder.....
5
9
8
12
10

C:\Test>
```



Root,LC, RC