[1] Explain the differences between linear and non-linear data structures!

-> In linear data structure, data elements are sequentially connected and each element is traversable through a single run.

-> In non-linear data structure, data elements are hierarchically connected and are present at various levels.

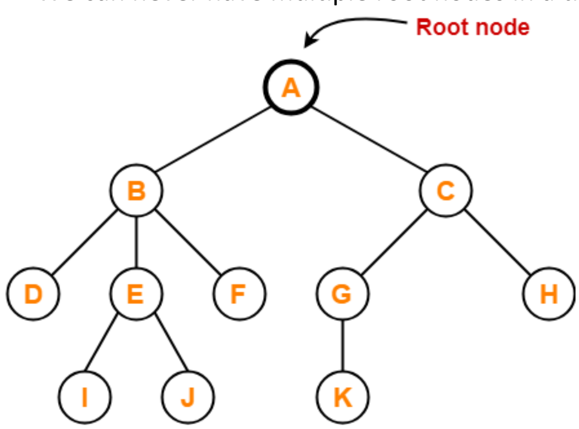
[2] Describe the following terminology in a tree: base root, key, edge, siblings, parent, child, and leaf!

1. Root

-> The first node from where the tree originates is called as a root node.

-> In any tree, there must be only one root node.

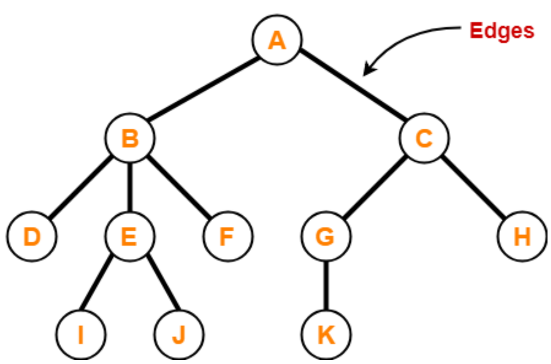
-> We can never have multiple root nodes in a tree data structure.



2. Edge

-> The connecting link between any two nodes is called as an edge.

-> In a tree with n number of nodes, there are exactly (n-1) number of edges

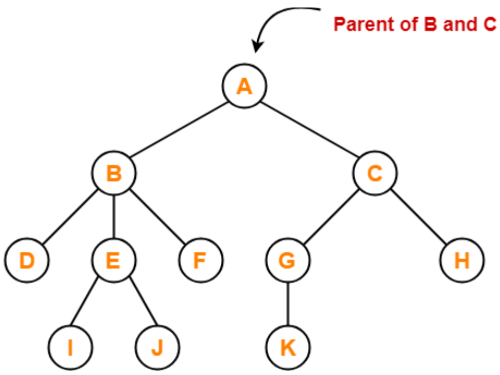


3. Parent

-> The node which has a branch from it to any other node is called as a parent node.

-> In other words, the node which has one or more children is called as a parent node.

-> In a tree, a parent node can have any number of child nodes.

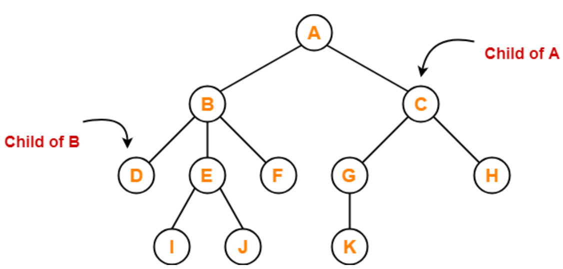


- ⇨ Node A is the parent of nodes B and C
- ⇨ Node B is the parent of nodes D, E and F
- ⇨ Node C is the parent of nodes G and H
- ⇨ Node E is the parent of nodes I and J
- ⇨ Node G is the parent of node K

4. Child

-> The node which is a descendant of some node is called as a child node.

-> All the nodes except root node are child nodes.

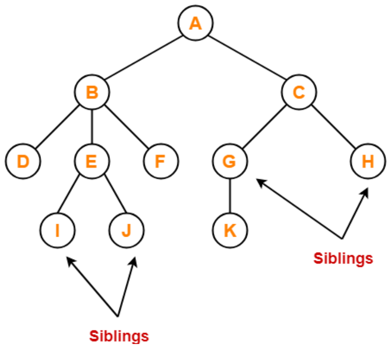


- ⇨ Nodes B and C are the children of node A
- ⇨ Nodes D, E and F are the children of node B
- ⇨ Nodes G and H are the children of node C
- ⇨ Nodes I and J are the children of node E
- ⇨ Node K is the child of node G

5. Siblings

-> Nodes which belong to the same parent are called as siblings.

-> In other words, nodes with the same parent are sibling nodes.

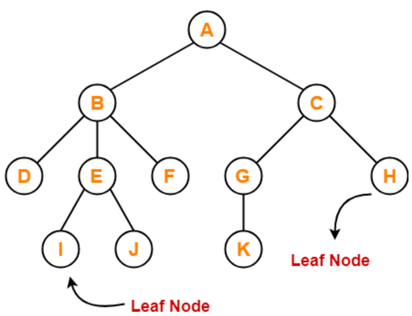


- ⇨ Nodes B and C are siblings
- ⇨ Nodes D, E and F are siblings
- ⇨ Nodes G and H are siblings
- ⇨ Nodes I and J are siblings

6. Leaf Node

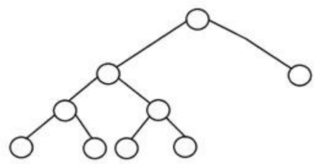
-> The node which does not have any child is called as a leaf node.

-> Leaf nodes are also called as external nodes or terminal nodes.

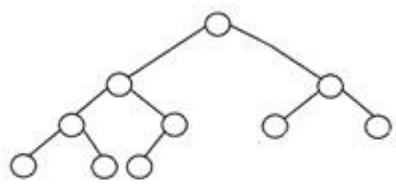


[3] Explain the following types of binary trees: full, complete, and perfect!

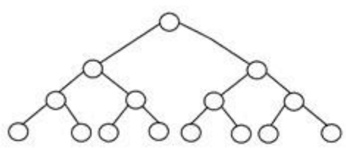
-> Full Binary Tree A Binary Tree is a full binary tree if every node has 0 or 2 children.



-> Complete Binary Tree: A Binary Tree is a complete Binary Tree if all the levels are completely filled except possibly the last level and the last level has all keys as left as possible



-> Perfect Binary Tree A Binary tree is a Perfect Binary Tree in which all the internal nodes have two children and all leaf nodes are at the same level.



[4] What makes a tree balanced?

A node in a tree is height-balanced if the heights of its subtrees differ by no more than 1. (That is, if the subtrees have heights h1 and h2, then $|h1 - h2| \leq 1$.)

[5] Explain the four properties of a binary tree!

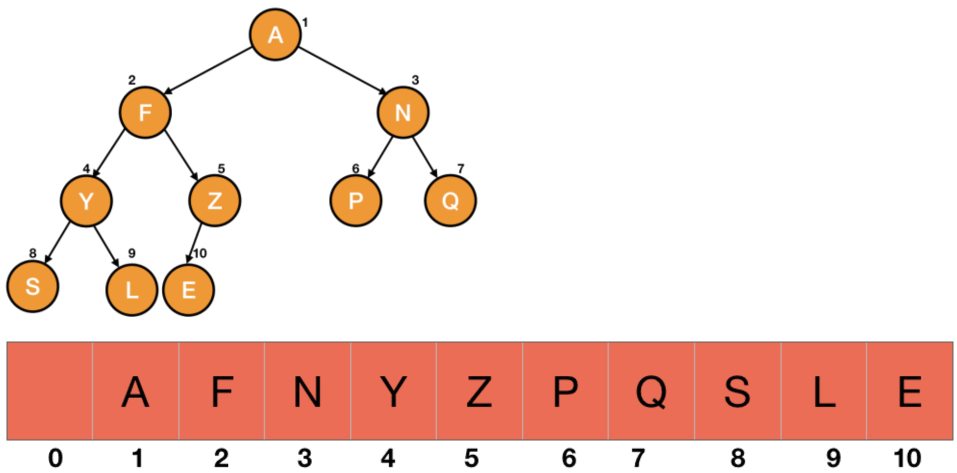
-> Maximum number of nodes present in binary tree of height h is 2h−12h−1 . Here height is the max number of nodes on root to leaf path. Here we are considering height of a tree with one node is 1.

-> In a binary tree with n nodes, minimum possible height or minimum number of levels are log2⌈n+1⌉log2⌈n+1⌉ . If we consider that the height of leaf node is considered as 0, then the formula will be log2⌈n+1⌉−1log2⌈n+1⌉−1

-> A binary tree with ‘L’ leaves has at least log2L+1log2L+1 number of levels

-> If a binary tree has 0 or 2 children, then number of leaf nodes are always one more than nodes with two children

[6] Explain the intuition of implementing a binary tree using an array!

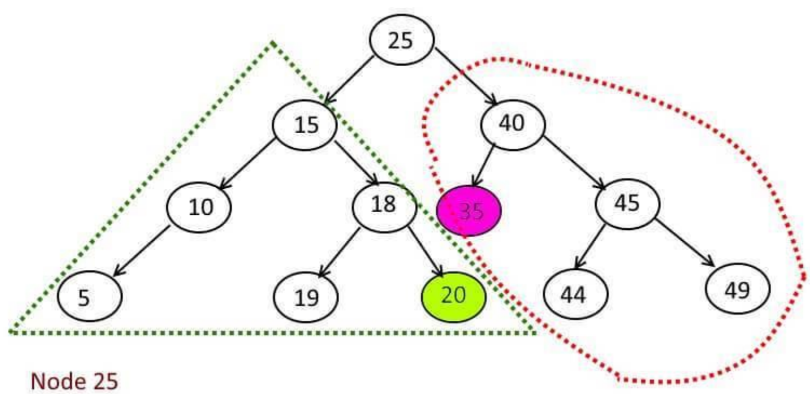


Array Representation of Binary Tree

get the parent, the right child and the left child using the properties of a complete binary tree. for a node i, the parent is i/2, the left child is 2i and the right child is 2i+1.

[7] Explain the differences between inorder successor and inorder predecessor!

Predecessor(the node lies behind of given node) and Successor (the node lies ahead of given node).



Node 25

Predecessor of node 25 will the right mode element in the left subtree.

which is 20

Successor of node 25 will the left most element in the right subtree

which is 35

[8] Draw the following binary search tree step by step (14 pictures):

- Insert 80, 30, 60, 50, 75
- Delete 60, 30, 75
- Insert 65, 30, 35
- Delete 80, 65, 35

```
Insert(80)          60            insert(50)             60
---------->                       ---------->           /
                                                       50

Insert(75)          60            insert(30)             60
---------->        /   \          ---------->          /    \
                 50     75                           50       75
                                                    /
                                                   30

Insert(80)          60            Delete(30)             60
---------->        /    \         ---------->          /    \
                 50      75                          50       75
                /          \                                    \
              30            80                                   80

delete(75)          60            Delete(60)             50
---------->        /    \         ---------->              \
                 50      80                                 80
```

```
insert(30)          50
---------->       /     \
                30       80
```

```
insert(65)            50
---------->         /       \
                 30         80
                           /
                         65
```

```
insert(35)          50
---------->       /      \
                30        80
                  \       /
                   35   65
```

```
delete(35)            50
---------->         /     \
                  30       80
                          /
                        65
```

```
delete(65)          50
---------->       /      \
                30        80
```

```
delete(80)            50
---------->          /
                   30
```