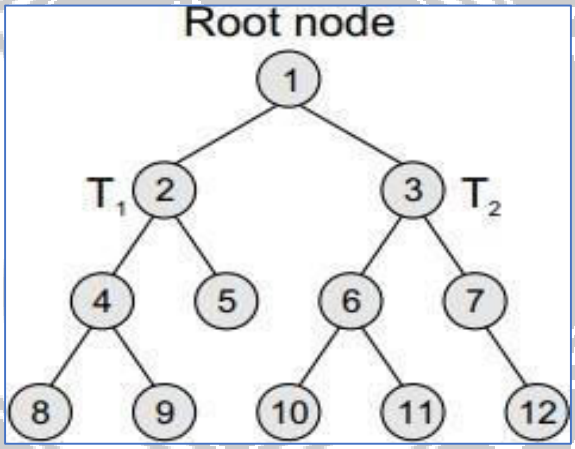
BINARY TREES

A binary tree is a data structure that is defined as a collection of elements called nodes. In a binary tree, the topmost element is called the root node, and each node has 0, 1, or at the most 2 children.

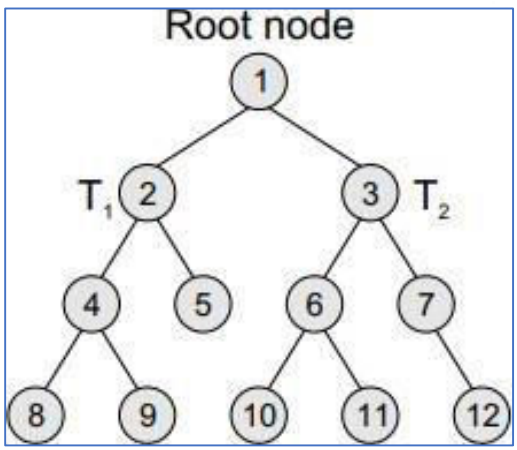
- Every node contains a data element, a left pointer which points to the left child, and a right pointer which points to the right child. The root element is pointed by a 'root' pointer. If root = NULL, then it means the tree is empty.
- In the figure, R is the root node and the two trees T1 and T2 are called the left and right sub-trees of R. T1 is said to be the left successor of R. Likewise, T2 is called the right successor of R.



TERMINOLOGY

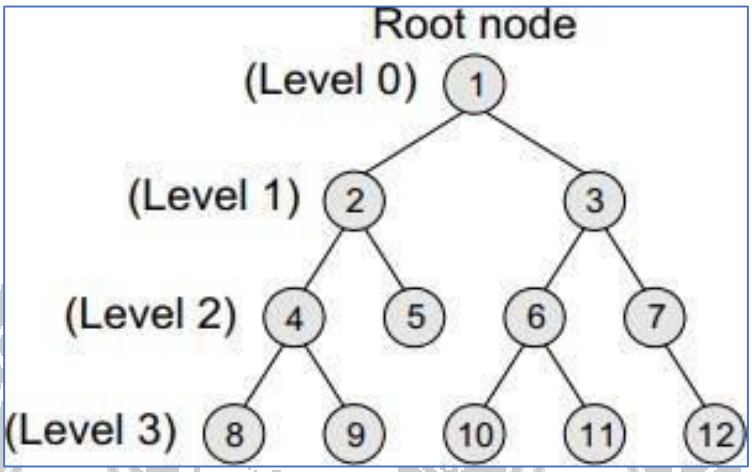
Parent

If N is any node in T that has left successor S1 and right successor S2, then N is called the parent of S 1 and S2. Correspondingly, S1 and S2 are called the left child and the right child of N. Every node other than the root node has a parent



Level number

Every node in the binary tree is assigned a level number. The root node is defined to be at level 0. The left and the right child of the root node have a level number 1. Similarly, every node is at one level higher than its parents. So all child nodes are defined to have level number as parent's level number + 1.



Degree of a node

Degree of a node is equal to the number of children that a node has The degree of a leaf node is zero.

- In-degree

  In-degree of a node is the number of edges arriving at that node.

- Out-degree

  Out-degree of a node is the number of edges leaving that node.

It is equal to the number of children that a node has. The degree of a leaf node is zero. For example, in above tree, degree of node 4 is 2, degree of node 5 is zero and degree of node 7 is 1.

Path

A sequence of consecutive edges. For example, in above Fig., the path from the root node to the node 8 is given as: 1, 2, 4, and 8.

Sibling

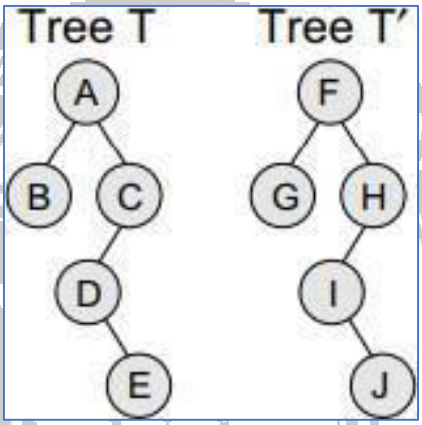
All nodes that are at the same level and share the same parent are called siblings (brothers). For example, nodes 2 and 3; nodes 4 and 5; nodes 6 and 7; nodes 8 and 9; and nodes 10 and 11 are siblings.

Leaf node

A node that has no children is called a leaf node or a terminal node. The leaf nodes in the tree are: 8, 9, 5, 10, 11, and 12.

Similar binary trees

Two binary trees T and T' are said to be similar if both these trees have the same structure. Figure shows two similar binary trees.



Edge

It is the line connecting a node N to any of its successors. A binary tree of n nodes has exactly n – 1 edges because every node except the root node is connected to its parent via an edge.

Depth

The depth of a node N is given as the length of the path from the root R to the node N. The depth of the root node is zero.

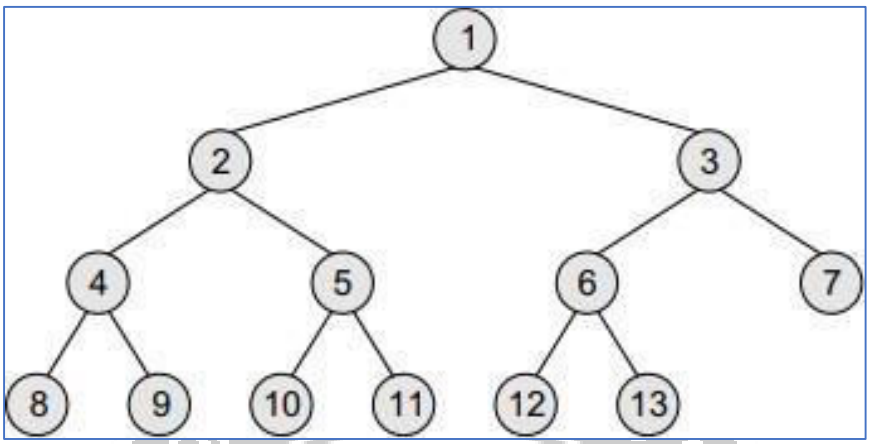
Height of a tree

It is the total number of nodes on the path from the root node to the deepest node in the tree. A tree with only a root node has a height of 1.

COMPLETE BINARY TREES

- A complete binary tree is a binary tree that satisfies two properties. First, in a complete binary tree, every level, except possibly the last, is completely filled. Second, all nodes appear as far left as possible.
- In a complete binary tree Tn , there are exactly n nodes and level r of T can have at most 2r nodes.

Figure shows a complete binary tree.



Above tree has exactly 13 nodes.

- The formula can be given as—if K is a parent node, then its left child can be calculated as 2× K and its right child can be calculated as 2 × K + 1.
- For example, the children of the node 4 are 8 (2 × 4) and 9 (2 × 4 + 1). Similarly, the parent of the node K can be calculated as | K/2 |. Given the node 4, its parent can be calculated as | 4/2 | = 2.
- The height of a tree Tn having exactly n nodes is given as:

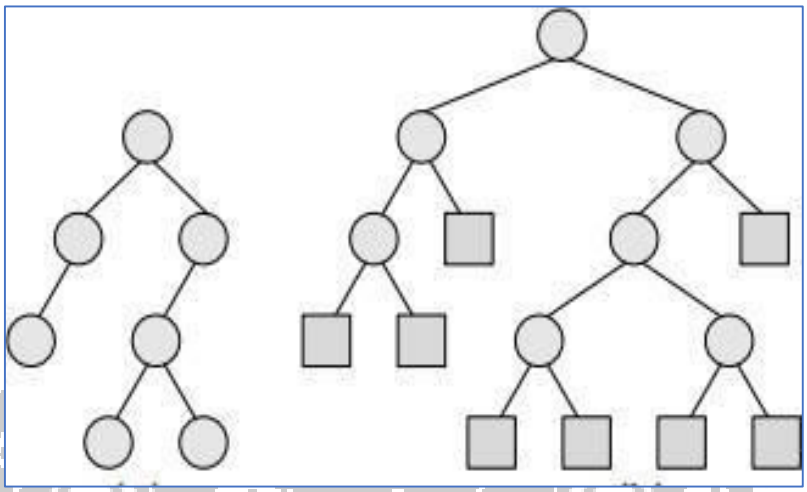
$$Hn = | \log_2 (n + 1) |$$

NOTE: In Fig. 9.7, level 0 has 20 = 1 node, level 1 has 21 = 2 nodes, level 2 has 22 = 4 nodes, level 3 has 6 nodes which is less than the maximum of 23 = 8 nodes.

Extended Binary Trees

- A binary tree T is said to be an extended binary tree (or a 2-tree) if each node in the tree has either no child or exactly two children.
- In an extended binary tree, nodes having two children are called internal nodes and nodes having no children are called external nodes.
- In Given Fig., the internal nodes are represented using circles and the external nodes are represented using squares.

- To convert a binary tree into an extended tree, every empty sub-tree is replaced by a new node. The original nodes in the tree are the internal nodes, and the new nodes added are called the external nodes. Below Figure shows how an ordinary binary tree is converted into an extended binary tree.



Representation of Binary Trees in the Memory

In the computer's memory, a binary tree can be maintained either by using a linked representation or by using a sequential representation.

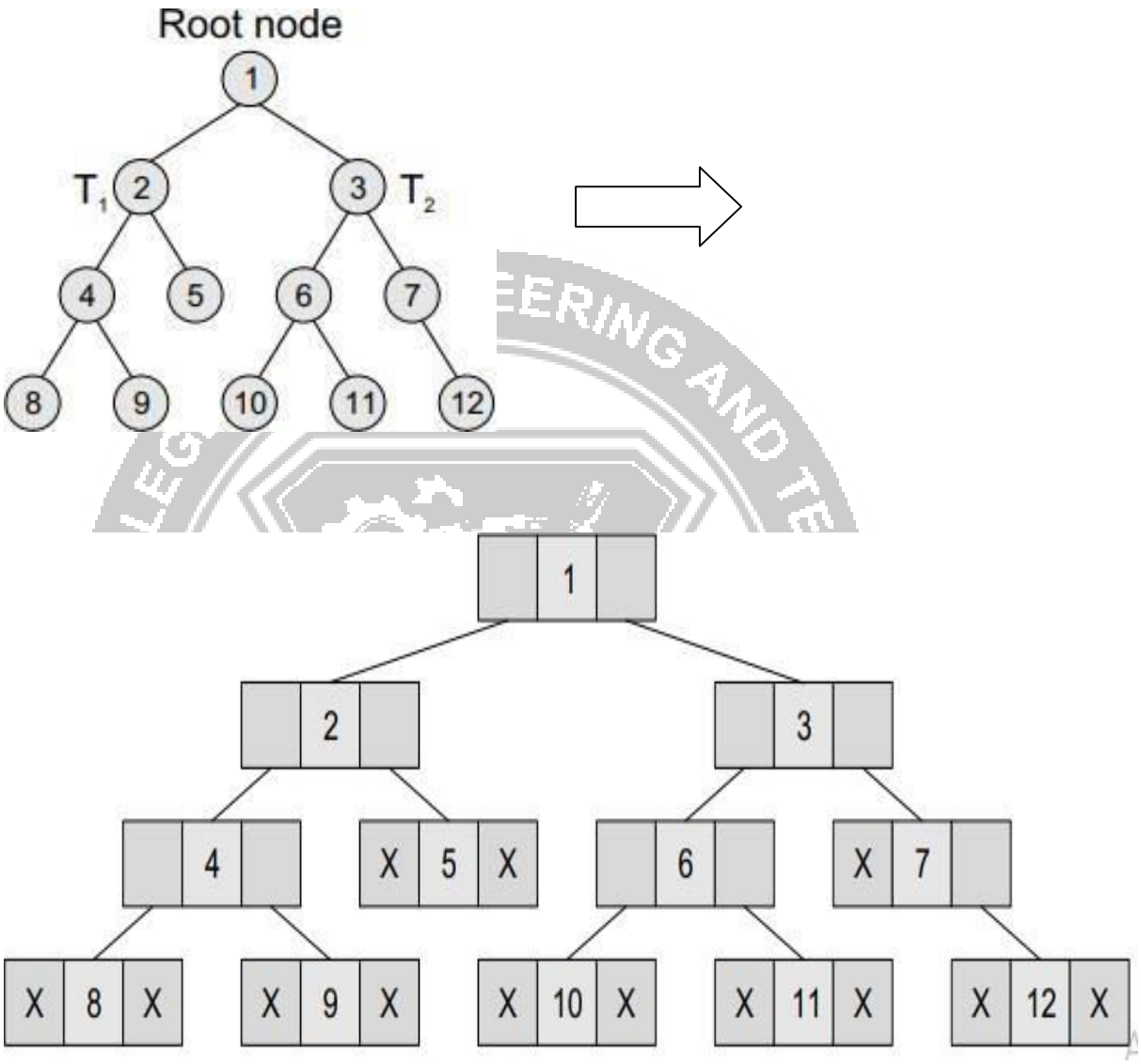
1. **Linked representation of binary trees**

   In the linked representation of a binary tree, every node will have three parts: the data element, a pointer to the left node, and a pointer to the right node.

```
struct node
{
struct node *left;
int data;
struct node *right;
};
```

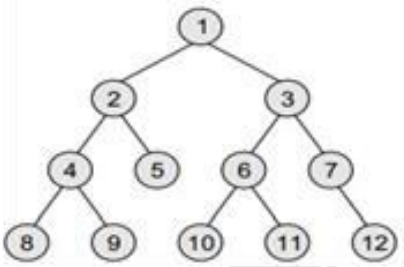
- Every binary tree has a pointer ROOT, which points to the root element (topmost element) of the tree.
- If ROOT = NULL, then the tree is empty.

Consider the binary tree given. The schematic diagram of the linked representation of the binary tree is shown in Fig.



In Fig, the left position is used to point to the left child of the node or to store the address of the left child of the node. The middle position is used to store the data. Finally, the right position is used to point to the right child of the node or to store the address of the right child of the node. Empty sub-trees are represented using X (meaning NULL).

Look at the tree given. Note how this tree is represented in the main memory using a linked list



ROOT 3

15 AVAIL

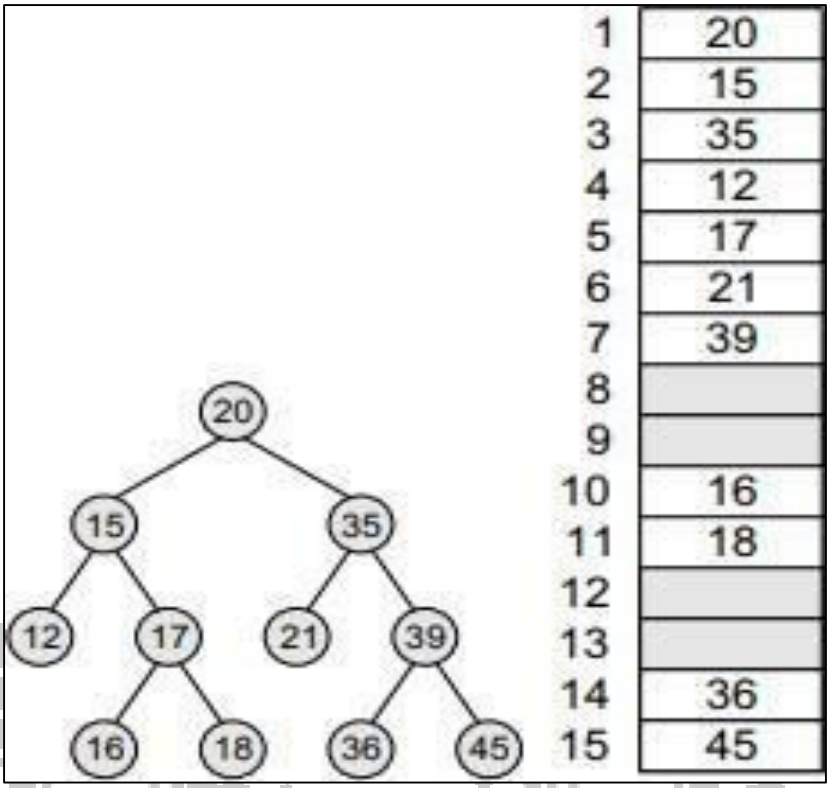
	LEFT	DATA	RIGHT
1	–1	8	–1
2	–1	10	–1
3	5	1	8
4			
5	9	2	14
6			
7			
8	20	3	11
9	1	4	12
10			
11	–1	7	18
12	–1	9	–1
13			
14	–1	5	–1
15			
16	–1	11	–1
17			
18	–1	12	–1
19			
20	2	6	16

2. Sequential representation of binary trees

Sequential representation of trees is done using single or one-dimensional arrays. Though it is the simplest technique for memory representation, it is inefficient as it requires a lot of memory space.

A sequential binary tree follows the following rules:

- A one-dimensional array, called TREE, is used to store the elements of tree.
- The root of the tree will be stored in the first location. That is, TREE [1] will store the data of the root element.
- The children of a node stored in location K will be stored in locations (2 × K) and (2× K+1).
- The maximum size of the array TREE is given as (2h–1), where h is the height of the tree.
- An empty tree or sub-tree is specified using NULL. If TREE [1] = NULL, then the tree is empty.



This shows a binary tree and its corresponding sequential representation. The tree has 11 nodes and its height is 4