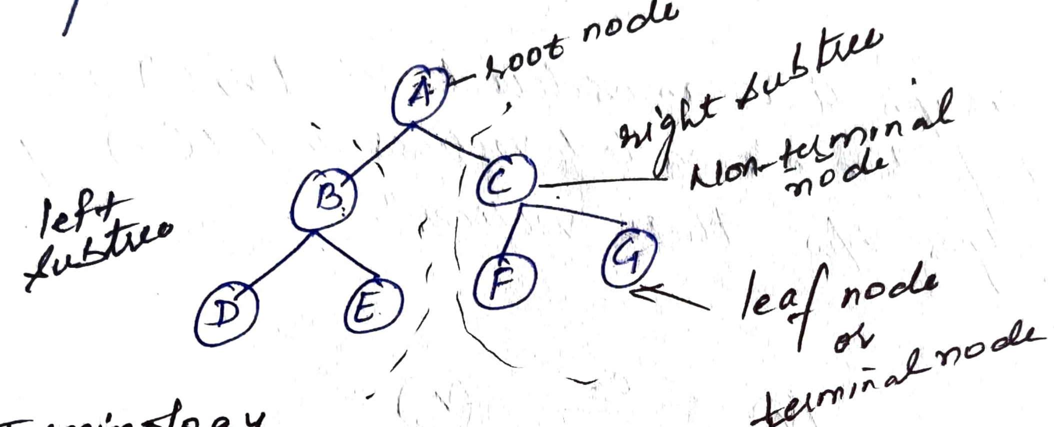
TREE [An ideal data structure for representing hierarchical data]. (1)

A (general) tree T is a finite non-empty set of elements is called the root, and the remaining elements, if any, are partitioned into trees, which are called the subtrees of T.



Tree Terminology

<1> **Level of an Element:**

The root of the tree is at level 1; its children, if any are at level 2 etc.

<2> **Degree of an Element:**

The degree of an element is the number of children it has. The degree of a leaf is zero.

<3> **Degree of a Tree:**

The degree of a tree is the maximum of its element degrees. The Degree of 2.

(4) **Height / Depth of a tree:**

If the root is at level 1, then the maximum level number of the tree is known as its height or depth.

→ BINARY TREE

Binary tree is a special type of tree in which every node or vertex has either no child, or one child node or two child nodes. A binary tree is an important class of data structure in which a node can have atmost two children.

Child node in a binary tree on the left is termed as 'left child' and node in the right is termed as the 'right child' node.

A binary tree may also be defined as follows:

(1) A binary tree is either an empty tree

(2) Or a binary tree consists of a node called the root node, a left subtree and a right subtree, or both of which will act like a binary tree once again.

Properties of Binary tree

(1) The maximum number of nodes at level 'l' of a binary tree is 2^{l-1}.

Here level is number of nodes on path from root to the node (including root and node). Level of root is 1.

This can be proved by induction.

For a root, $l=1$, number of nodes $= 2^{1-1} = 1$

Assume that maximum number of nodes on level l is

2^{l-1}.

Since in Binary tree every node has atmost 2 children. next level would have twice nodes ie $2 * 2^{l-1}$.

A binary
h add atmo

(2) In Binary tree, number of leaf node or external node is always one more than ~~number of internal~~ nodes. with two children.

$. L = T + 1$

Let n_0 be the number of leaf nodes

n_1 be the number of nodes with degree 1

n_2 be the number of nodes with degree 2

n = total number of nodes

ie $n = n_0 + n_1 + n_2$ ——— (1)

we know that the number of branches of binary tree is $n-1$. It can also be observed that branches may emerge from the node that has degree one or the node that has degree two

Hence $2n_2 + n_1$

So, $n - 1 = 2n_2 + n_1$

$n = 2n_2 + n_1 + 1$ ← (2)

from eqn (1) & (2) we get

$n_0 + \cancel{n_1} + n_2 = 2n_2 + \cancel{n_1} + 1$

$$\boxed{n_0 = n_2 + 1}$$

Maximum

h' is 2^h

Soln

Since

leve

A binary tree of height h , $h \geq 0$ has at least h and atmost 2^h-1 elements in it.

Or

Maximium number of nodes in a binary tree of height 'h' is 2^h-1.

Soln

Since, there must be at least one element at each level, the number of elements is atleast h.

As each element can have atmost two children, the number of element at level i is at most 2^{i-1}, $i>0$. for $h=0$ the total number of elements is 0, which is equals 2^0-1. for $h>0$, the number of elements cannot exceed.

$$\sum_{i=1}^{h} 2^{i-1} = 2^{1-1}+2^{2-1}+2^{3-1}+\cdots+2^{h-1}$$

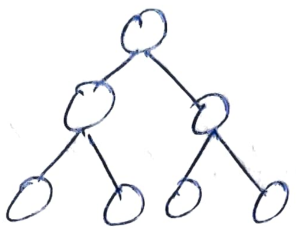
$$= 2^0+2^1+2^2+\cdots+2^{h-1}$$

$$= 1+2^1+2^2+\cdots+2^{h-1}$$

$$= \frac{1(2^h-1)}{(2-1)} = 2^h-1 \qquad \left\{ S_n = \frac{a(r^n-1)}{(r-1)} \right.$$

$\{\therefore$ If height of a leaf is considered as 0. In this convention, the above formula becomes $2^{h+1}-1\}$

(4) A binary tree with n nodes has exactly n-1



$n = 8$ (nodes)
no of branches or edges
$= 7$ $(n-1)$

(5) The height of a binary tree that contains n, $n \geq 0$ elements is atmost n and atleast $\lceil \log_2 (n+1) \rceil$

Sol^n

Since there must be atleast one element at each level, the height cannot exceed 'n'.

we know that a binary tree of height h can have no more than $2^h - 1$ elements

Thus,

$$n \leq 2^h - 1$$

$$n + 1 \leq 2^h$$

Taking log both sides,

$$\log_2 (n+1) \leq h \cdot \log_2 2$$

$$\log_2 (n+1) \leq h \cdot 1$$

$$h \geq \log_2 (n+1)$$

Since h is an integer, therefore we have

$$h \geq \lceil \log_2 (n+1) \rceil$$

Representation of a Binary tree

Binary tree can be represented in any of the following two ways:

(1) Using an array (Linear representation)
(2) Using a linked list (Link representation).

(1) Array Representation:

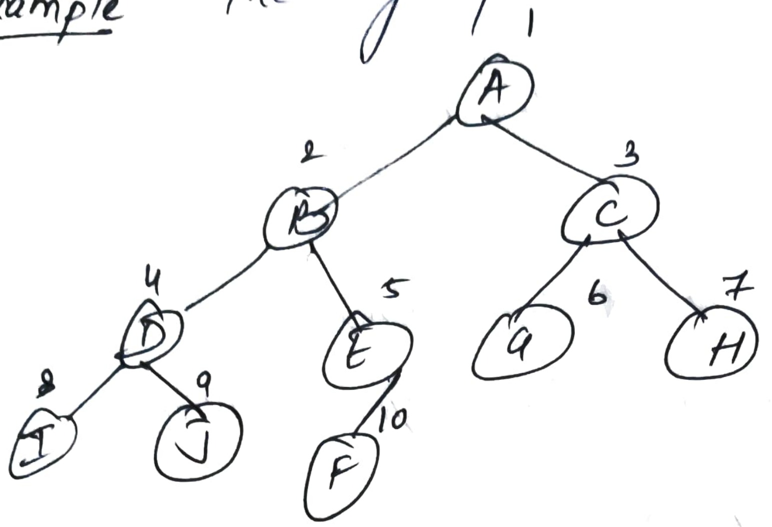
A binary tree can be stored represented using an array, which is called sequential representation. This type is representation is static ie a block of memory for an array is to be allocated before going to store the actual tree in it.

In this representation, the nodes of the tree are stored level by level, starting from the 0th level.

1. The root node is at location 1.
2. For any node with index $i, 1 < i \le n$.

(a) Parent (i) = $\lfloor i/2 \rfloor$
(b) left child(i) = $2i$
(c) Right child(i) = $2i+1$

example The array representation of the binary tree



1	2	3	4	5	6	7	8	9	10
A	B	C	D	E	G	H	I	J	F

Advantages

(1) Data are stored without any pointer to the successor or ancestor.

(2) Any node can be accessed from any other node by calculating the index.

(3) Programming languages which do not support dynamic memory allocation use only this type of representation.

(4) Simplicity.

Disadvantages

(1) Array representation is not suitable for normal binary tree but it is only ideal for complete binary tree.

tree:) most of the array entries are empty i.e. unnecessarily more memory is wasted.

(3) There is no way possible to enhance the tree structure.

(4) Additions and deletions of nodes are inefficient because of the data movements in the array.

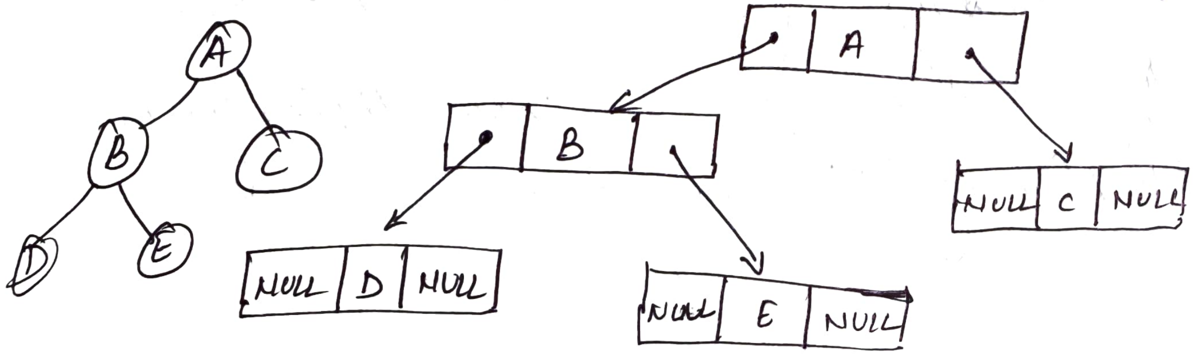
(2) Linked list Representation of a Binary tree:

The binary tree can be represented using dynamic memory allocation of a node in a linked list form. In a linked list allocation technique a node in a tree contains three fields.

1) Info (data)
2) Left link (left)
3) Right link (Right)

left	info	Right

when a node has no children, the corresponding pointer field are NULL.



declare structure of tree node

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

This type of representation is dynamic ie block of memory required for the tree need not be allocated before. They are allocated only on demand.

Advantages:

(1) Insertion and deletion involve no data movement.
(2) No wastage of memory.
(3) Enhancement of the tree is possible.

Disadvantages

(1) In this, pointer fields are involved which occupy more space than just data fields.

(2) programming languages which donot support dynamic memory allocation have difficulty in implementing the binary tree.