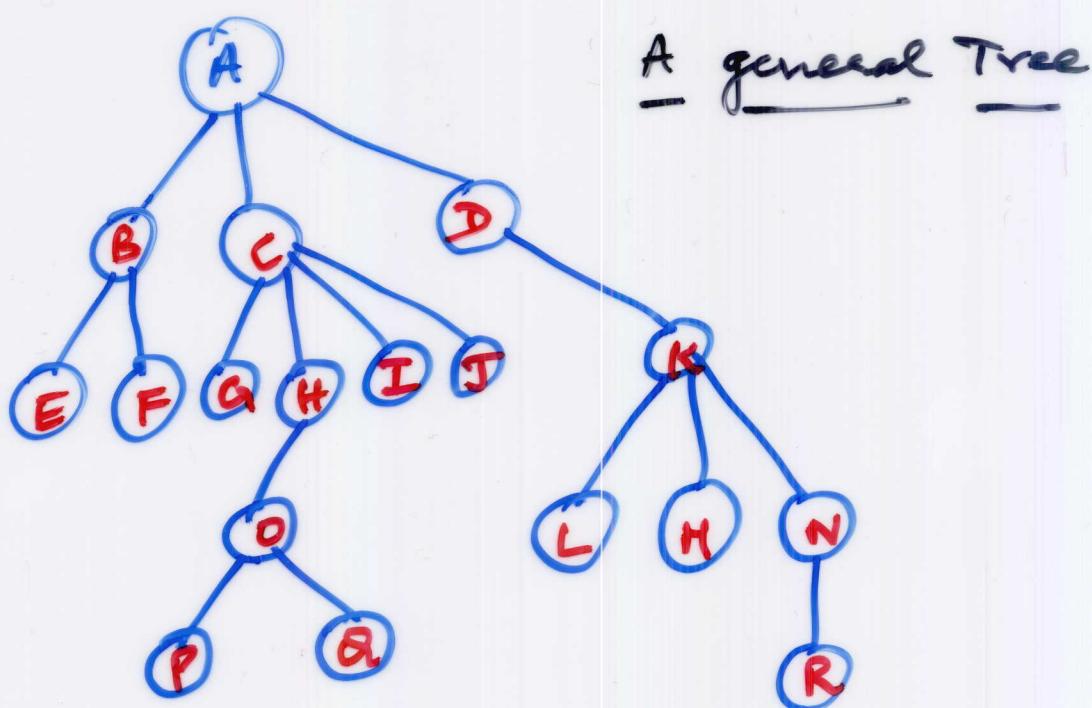


## General Trees :-

- It has any no. of nodes.
- Children of a node are called Siblings of each other.

Ex:



Difference between a Binary tree  
and a general tree:-

1. A binary tree may be empty.  
But there should be at least one node in a general tree.

2. Every node in a BT has atmost two sons.

Every node in a GT <sup>may have</sup> more than two sons.

3. children of a node in a BT are called left / right child.

But in a GT, children cannot be distinguished.

Memory representation of a GT:-

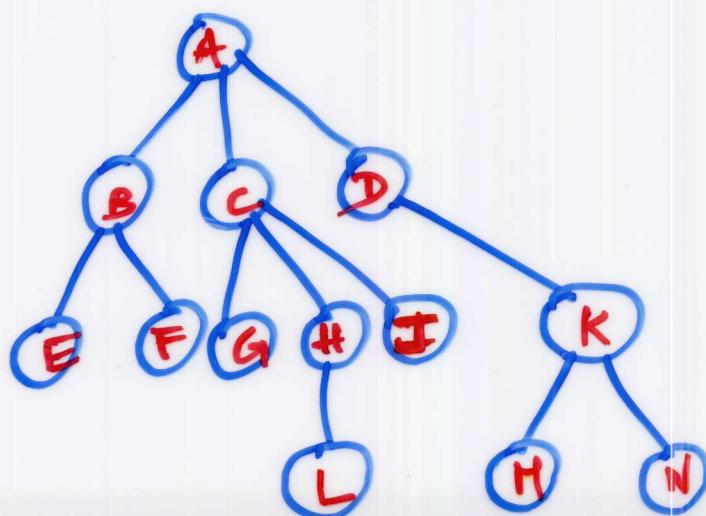
- Linked lists / arrays
- three parallel arrays INFO, CHILD and SIBL
- a pointer variable ROOT.

- Each node  $N$  of GT T has a location  $K$  in the array, such that

$\text{INFO}(k)$  - - - data at node  $N$   
 $\text{CHILD}(k)) \} - - - \text{location of the first}$   
 $\text{child of } N$   
 $= \text{NULL} \dots \text{no children}$

$SIBL(K) =$  location of the next  
 sibling of  $N$   
 $= \text{null}$ , if  $N$  is the last  
 child of its parent.

三



# Array representation of a GT:-

INFO

1	
2	A
3	B
4	C
5	
6	G
7	H
8	J
9	N
10	H
11	L
12	K
13	
14	F
15	E
16	D

CHILD

3
15
6
13
0
11
0
0
10
0
0
12

SIBL

0
4
16
7
8
0
0
9
0
0
0
0
14
0

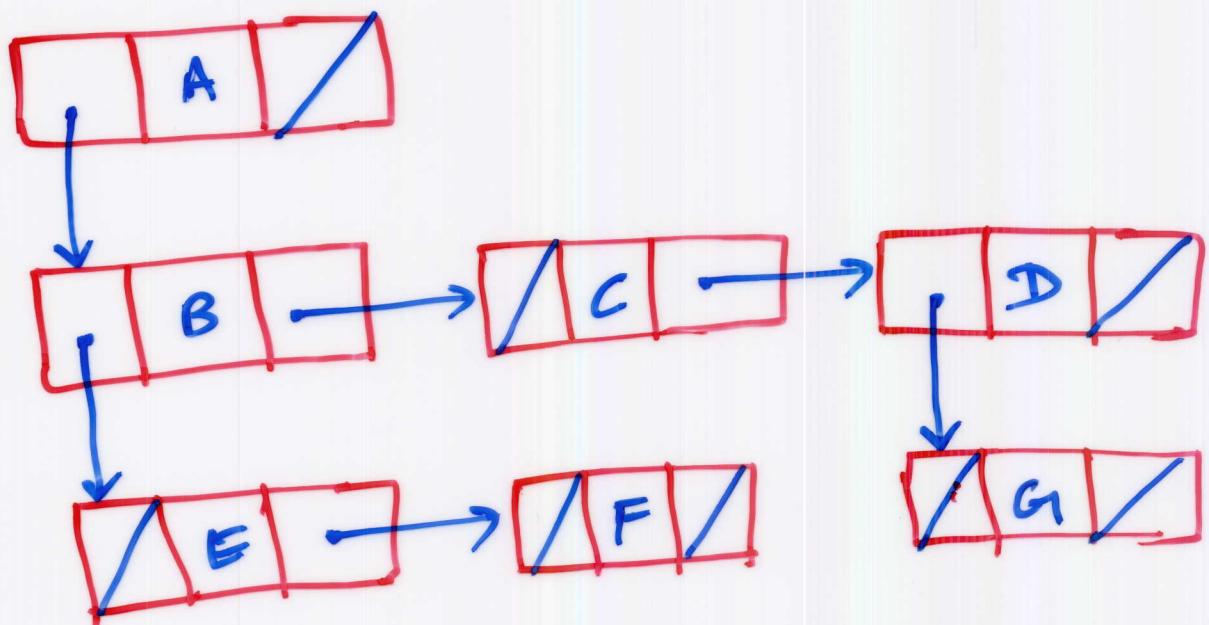
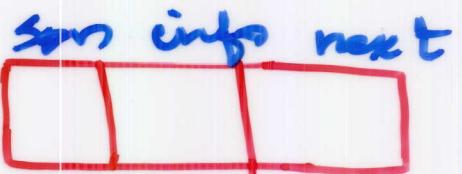
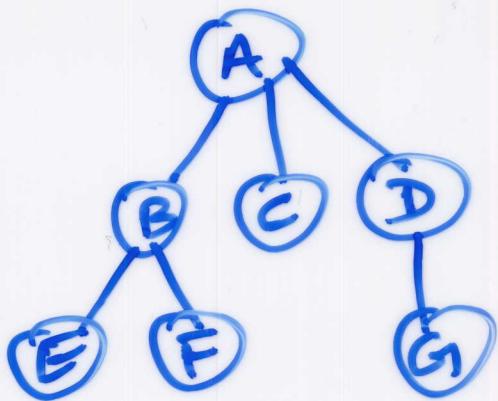
Linked repn:-

```
struct treenode{  
    int info;  
    struct treenode * father;  
    struct treenode * son;  
    struct treenode * next;  
};
```

∴ all traversals are from a node  
to its sons, father field may be  
omitted.

```
struct node{  
    int info;  
    struct node * son;  
    struct node * next;  
};
```

Ex:-



Traversals:-

In-order:-

1. Traverse son in in-order
2. Visit node
3. Traverse next in in-order

(b)

Ex:

```
void inorder(struct treenode *t)
{
    if (t)
    {
        inorder ( t->son );
        printf( " %c ", t->info );
        inorder ( t->next );
    }
}
```

Ex:

E F B C G D A

Pre-order :-

1. Visit node
2. Traverse son in pre-order.
3. Traverse next in pre-order.

Ex:

A B E F C D G

(7)

Post-order :-

1. Traverse sons in post-order
2. Traverse next in post-order
3. visit ~~the~~ node.

~~E F G D C B A~~

E F G D C B A