2 n). eval and searching in tree

Binary Tree

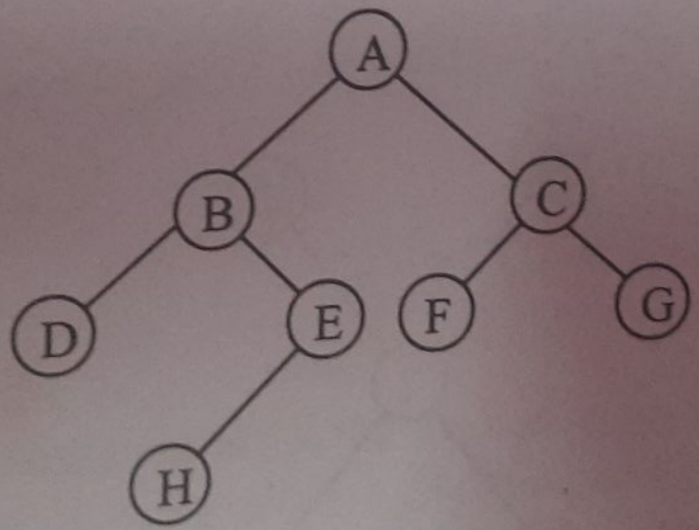
In linked list, every node has two fields, first represents the data and second has information of next node. But in tree every node has three fields, first represents data, while second and third keep the information of left and right child of node. As in real life every tree starts with root, similarly here also the beginning node is called root. A binary tree can be defined as-

A binary tree is a finite set of nodes.

(1) It is either empty or

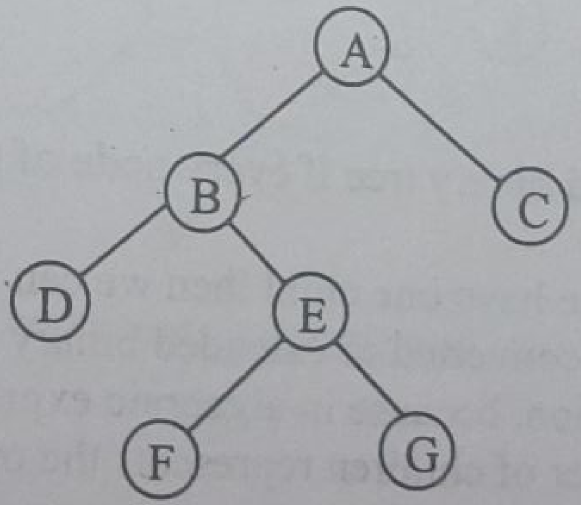
(2) It consists a node called root with two disjoint binary trees called left subtree and right subtree.

There are several ways to represent tree structure. We will use graph to represent tree in which every node will be represented by circle and a line from one node to other is called edge.



Strictly Binary Tree

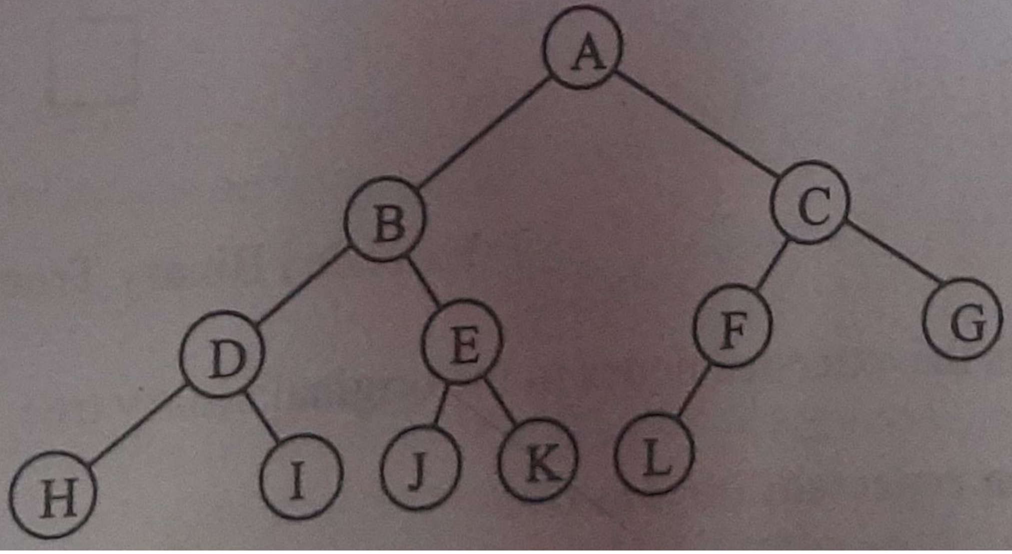
If every internal node (non terminal node) has its non empty left and right children then it is called strictly binary tree.



Here every internal node A, B, E has two non empty left and right child hence it is strictly binary tree.

Complete Binary Tree

We have already seen binary tree where each node has only left and right child but only two children maximum are allowed. So we can say at any level maximum number of nodes will be 2^l only. A complete binary tree is a binary tree where all the nodes have both children except last node. Let us take a complete binary tree.



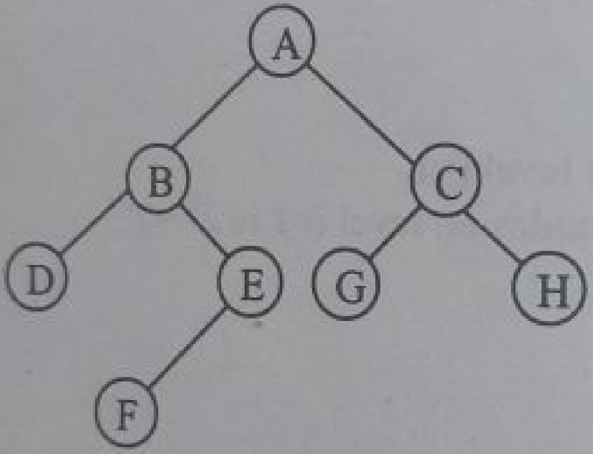
Representation of Binary Tree

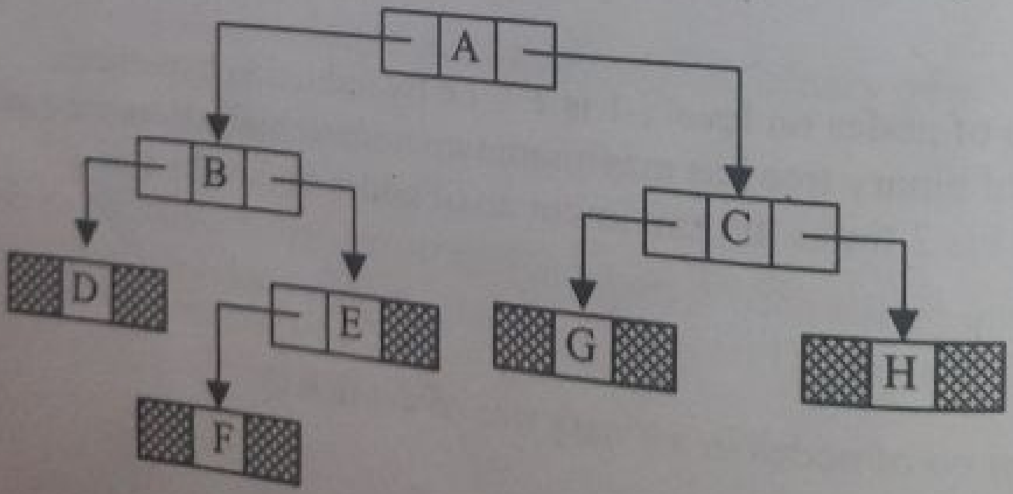
As list is implemented in two ways, first in linked list and second through array. Similarly binary tree can also be implemented in two ways.

1.Linked representation
2.Array representation

Linked Representation

As in the linked list we take the structure for tree. In which we take three members. First member for data (this can be whole record), second member for left child and third member is for right child. Second and third members are structure pointers which point to the same structure as for tree node.





First we should declare structure for tree node.

```
struct node {
            char data;
            node * lchild;
            node  *rchild;
           }
```

Here first member data is for information field of node, second member is for left child of node which points to the structure itself, it contains the address of left child. If node has no left child then it should be NULL.Third member is for right child of node which also points to the structure itself, it contains the address of right child. If node has no right child it should be NULL.