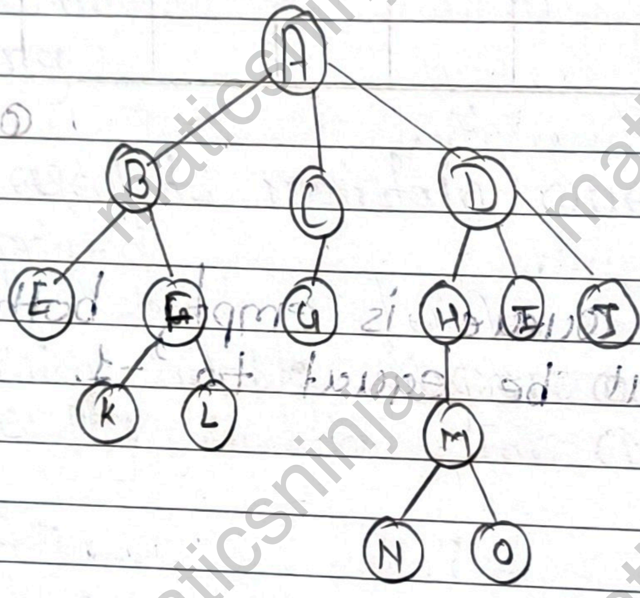
6. Tree

Defination:
A tree is the collection of elements called node. one of which is distinguished as Root Say are.



Root :
It is a Special node in a tree Structure and the entire tree is refrerence to it.
This node does not have parent. In above tree Root is A.

Parent :
It is an immediate predecer of a node
In above figure A is parent of B, C, D.

Child:
All imediate node Successer are child node.
B, C, D are children of A.

PAGE NO.:

4] Siblings:
Nodes with the same parents are Siblings
H, I and J are Siblings as they are children of the same Parent D.

5] Path
It is number of Successive edges from Source node to destination node. The path length from node D to node N in is 3. Node D is connected to N through three edges D-H, H-M, M-N.

6] Degree of Node
Degree of Node is a Number of children of a node in above fig A and D are nodes of degree 3,
- B, F, and M are nodes of degree 2,
- C and H are nodes of degree 1
- E, K, L, N, O, I, J are node of degree 0.
A node of degree no is also known as the leaf node.

7] Leaf node
A node of Degree zero Called leaf node It is called terminal node. It has no children

In this the E, K, L, G, N, O, I, J are leaf Node.

PAGE NO.:

8] level

The level of a node in a binary tree is defined as

1. The root of the tree has level 0.
2. The level of any other node in the tree is one more than the level of its father.

Nodes	Level
A	0
B, C, D	1
E, F, G, H, I, J	2
K, L, M	3
N, O	4

9] Depth

The depth of a binary tree is the maximum level of any node in the tree.

The depth of the tree in fig is 4.

10] Degree of a tree:

The degree of a tree is the maximum degree of a node in the tree.

Ex. The node D and node A have maximum degree and it is 3.

[1] Depth / Height of a tree :
Height of a tree is the distance of the root node from its farthest descendant.

The distance of the root node A from its farthest descendant N is 4.

[2] Ancestor / descendant node :
All the nodes on the path leadining to root node are ancestors of a node.

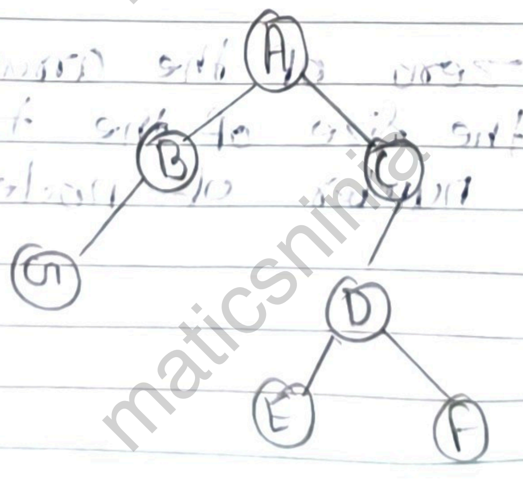
In fig, ancestors of the node N are (M, H, D, A)

all the nodes in the Subtress of a node are its descendants.

In fig the descendants of node D are (H, I, J, M, N, O)

• Binary Tree

A tree is binary if each node of the tree can have maximum of too children.

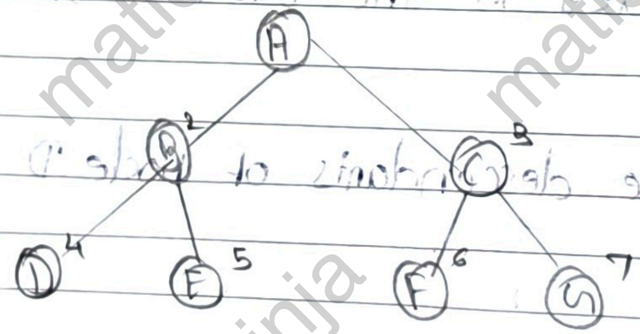


Node A has two children B&C. Similarly, nodes B and C each have one child namely G & D respectivly

⇒ Representation of a Binary Tree using an Array.

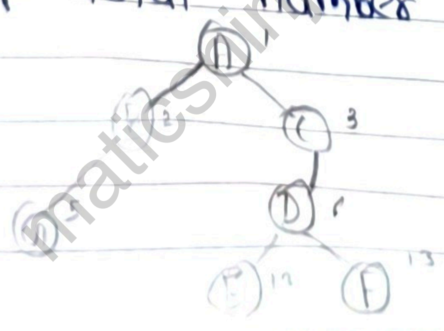
- In order to represent a tree in a single one dimensional array the node are numbered sequentially level by level left to right.

- Even empty nodes are numbered



0	1	2	3	4	5	6	7
7	A	B	C	D	E	F	G

Location number zero of the array can be used to store the size of the tree in the terms of total number of nodes.



Numbering of nodes with some non-existing children

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
13	A	B	C	\0	G	D	\0	\0	\0	\0	\0	E	F	\0	\0

Array representation of the tree.

⇒ LINK Representation of Binary Tree.

Linked representation of a binary tree is more efficient than array representation. A node of binary tree consist of three fields are shown below.

Data
Address of the left child.
Address of the right child.

Left	DATA	Right.

- The left hold the address of left node.
- The right hold the address of right node.

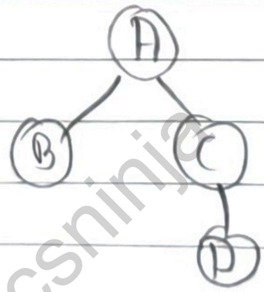


Fig. A Binary Tree

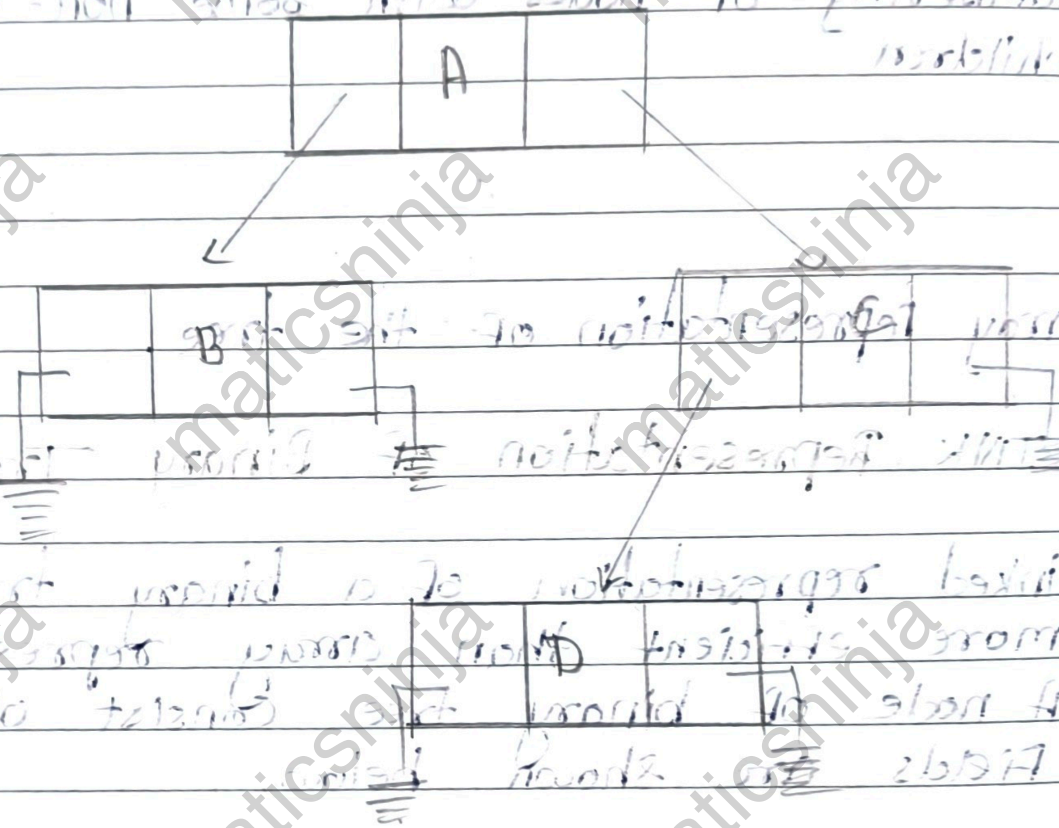
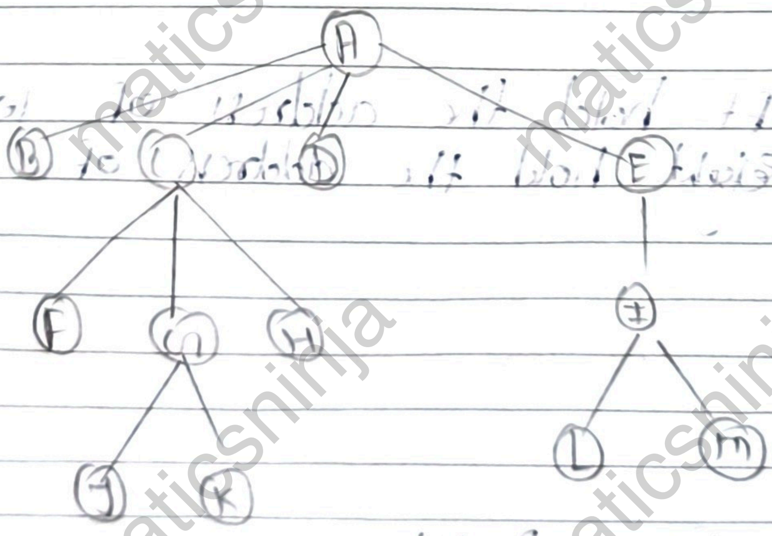


Fig. Link representation of binary tree.

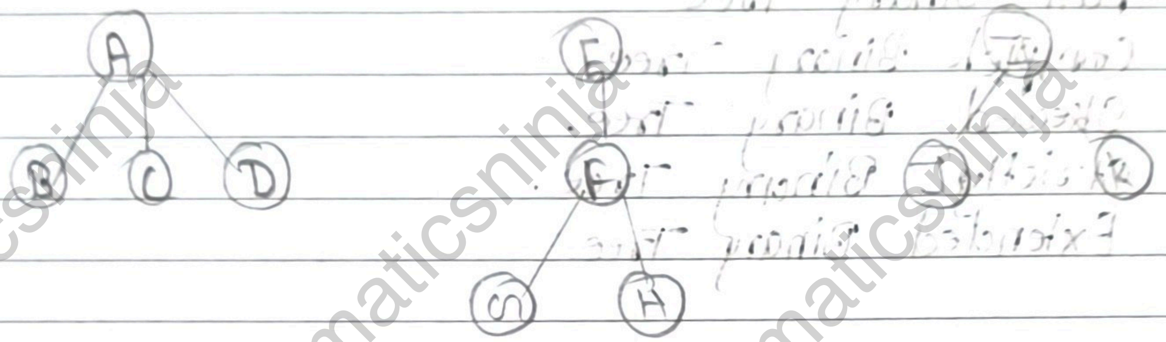
⇒ General Tree.

In a general Tree, number of children per node is not limited to two



⇒ Forest.

A Forest is defined as a set of trees.



H/W

Q. Compare General tree and Binary tree

General tree	Binary tree
① Any number of children for a node	① Maximum of two children per node
⑪ May not be possible to store address of children directly in a node	⑪ It is possible to store address of children in a node
⑪ Tree representation is difficult	⑪ Tree representation is easy.
⑫ Difficult to write algorithms for general tree	⑫ There are standard algorithms.

- Types of Binary Tree

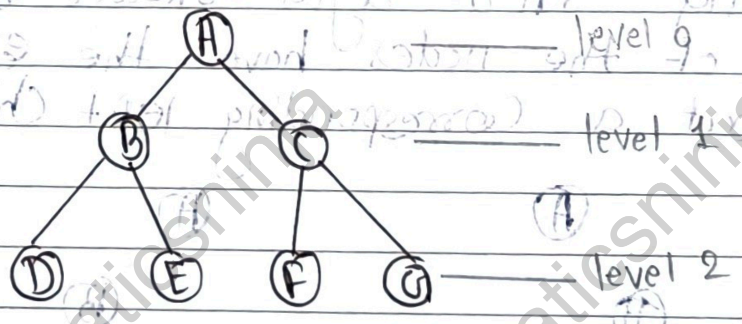
1] Full Binary Tree
2] Complete Binary Tree
3] Skewed Binary Tree.
4] Strictly Binary Tree.
5] Extended Binary Tree.

1] Full Binary Tree.

A Binary Tree is said to be full Binary Tree if each of its node has either Two children or no children at all.

Every level is completely filled up.

number of node at any level i in a full binary tree is given by 2^i



$2^i = 2^0 = 1$

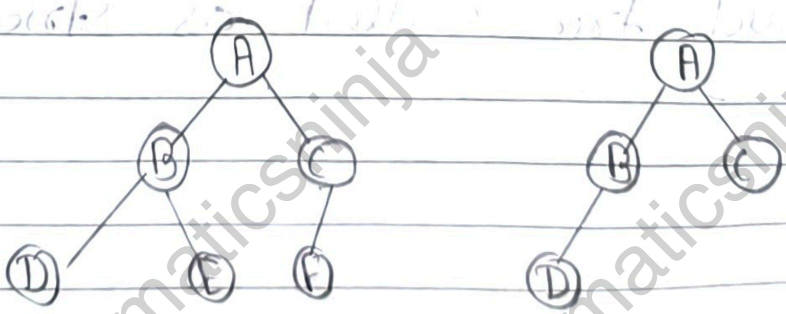
$2^1 = 2$

$2^2 = 4$

2] Complete Binary tree.

A complete Binary tree is defined as Binary tree has a binary tree where all leaf node are on level n or n-1.

levels are filled from left to right.

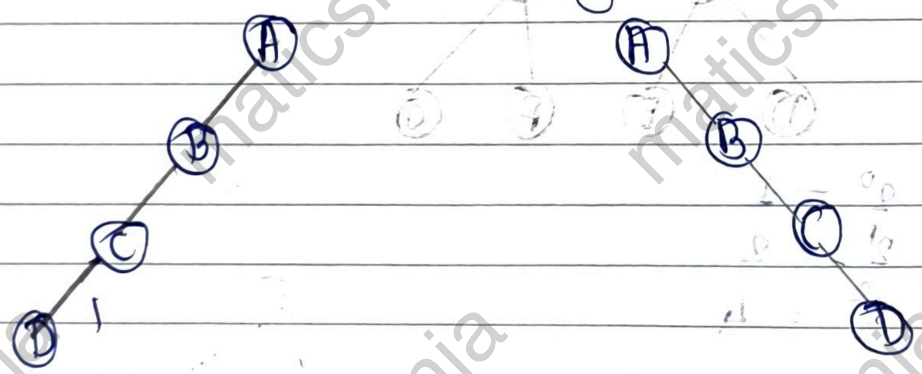


3] Skewed Binary tree.

→ A skewed Binary tree could be skewed to the left to right.

In left skewed Binary tree most of the nodes have the left child without corresponding right child

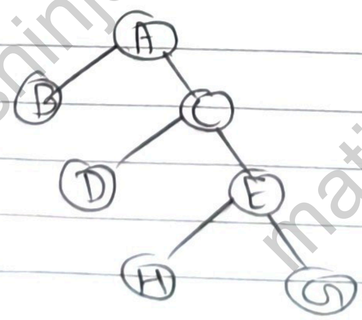
Similarly In a right skewed binary tree most of the nodes have the right child without a corresponding left child



4] Strictly Binary tree.

If every non-terminal node in a binary Tree consist of non-empty left subtree and right subtree

Then such tree called as strictly Binary tree.



5] Extended Binary tree.

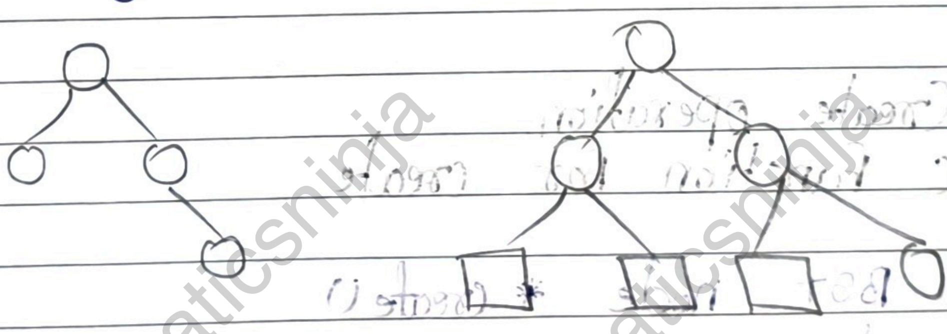
In an extended tree each empty Subtree is replaced by a failure node.

A failure node is represented by ☐ Square.

nodes with Two children are called internal node.

nodes with 0 zero children are called external node

Any Binary tree Can be converted into Extended binary tree By replacing each empty Subtree by failure node.



Binary tree   Extended Binary tree

⇒ Traversing of Binary tree

To traverse a Binary tree There are three Method

preorder Traversal
Inorder Traversal
postorder Traversal

1] Preorder Traversal

Step 1: Visit the root
Step 2: Visit the left Subtree
Step 3 : Visit the Right Subtree

2] Inorder Traversal

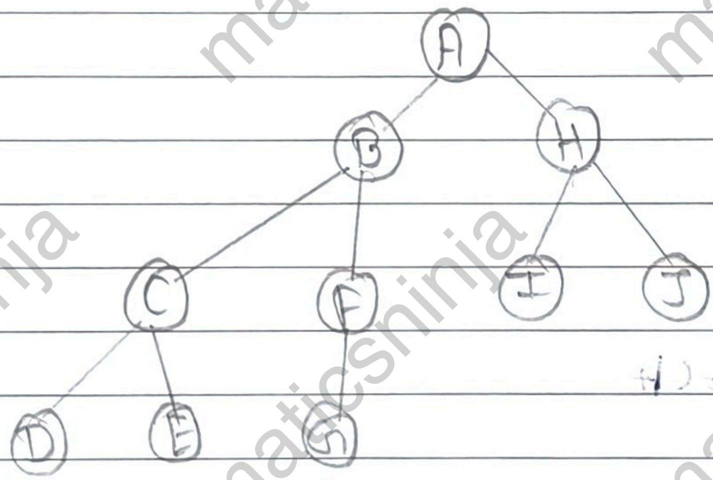
Step 1: Visit the left subtree
Step 2: Visit the root.
Step 3: Visit the Right subtree.

PAGE NO.:

3] Postorder traversal

Step 1 : Visit the left subtree
Step 2 : Visit the right subtree
Step 3 : Visit the root.

Q1. Perform Inorder, Preorder and postorder traversal b



Preorder :

ABCDEFGHIJ

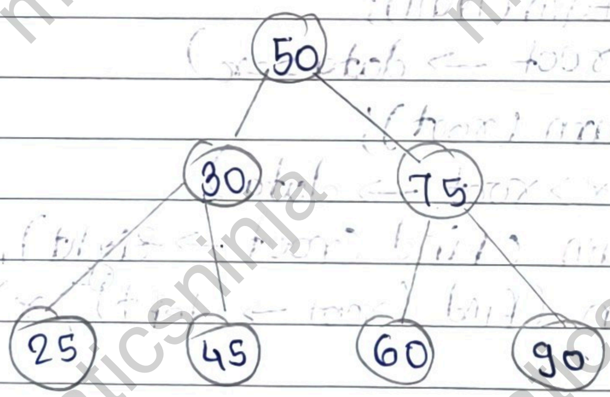
Inorder :
DECBGFAIHJ

postorder :
DECGFBIJHA

- Binary Search tree.

A Binary Search tree is a Binary tree which is either empty or In which each node Contain a key that Satisfy the following conditions.

1] All Keys are destenct.
11] For every node x in the tree the Values of all the keys in its left Subtree are Smaller than the key value in x.
111] For every node x in the tree the Values are all the keys in its right subtree are larger than the key value in x.



⟹ operations on Binary Search tree

1] Initialize
2] Find
3] make empty
4] Insert
5] Delete
6] Create

7] Find mean min.
8] Find max.

1] Initialize

```
BST * node initialize()
{
  return(Null);
}
```

2] C function for find()

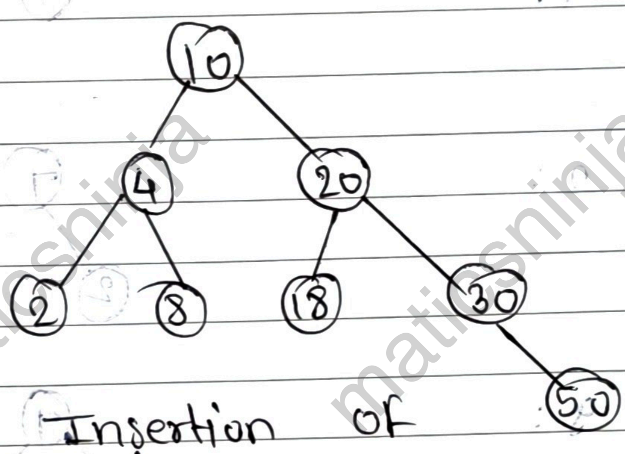
```
BST node *find (BST node * root, int x)
{
    if (root == Null)
        return(Null);
    if (root -> data == x)
        return (root);
    if (x > root -> data)
        return (find (root -> right), x);
    return (find (root -> left), x);
}
```

3] C function to ~~reverse~~ release memory.

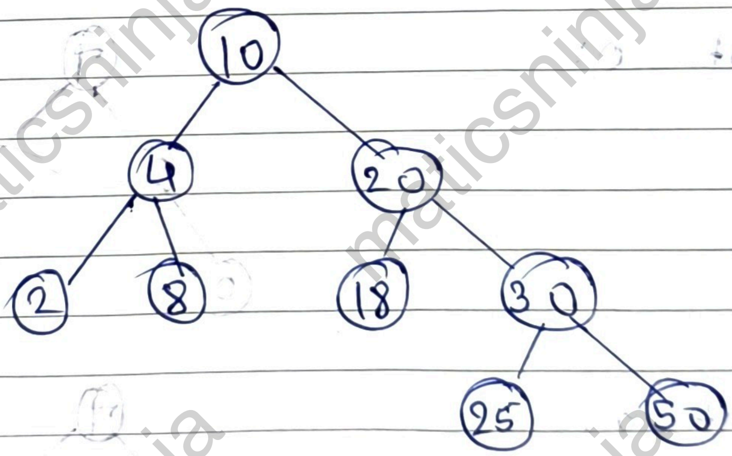
```
BST node * makeempty (BST node * root)
{
if (root != Null)
{
```

makeempty (root → left);
makeempty (root → right);
Free (root);
}
return (Null);
}

4] Insertion operation.



Before Insertion of element 25.



BST After Insertion.

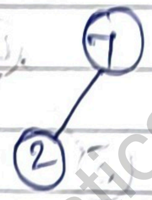
Q. Creation of Binary Search tree.

Insert (7, 2, 9, 0, 5, 6, 8, 1) into Binary Search with By repeted application of insert operation.

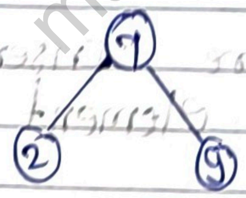
→

1] Insert 7:

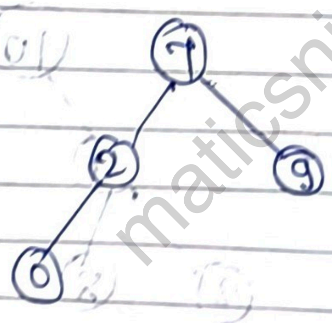
2] Insert 2:



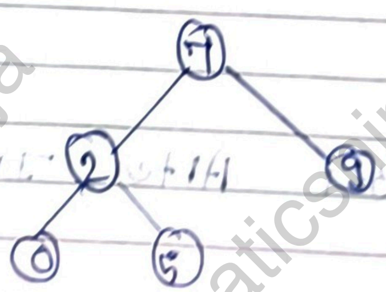
3] Insert 9:



4] Insert 0:

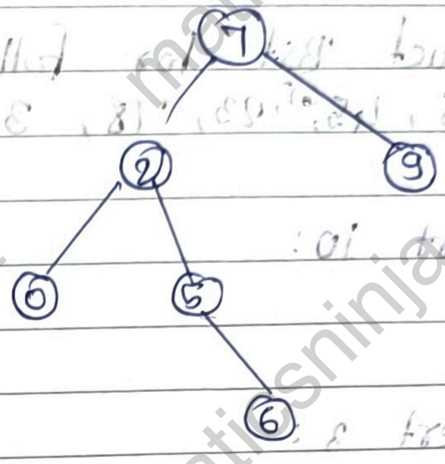


5] Insert 5:

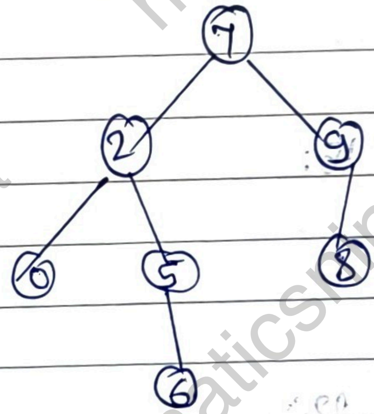


PAGE NO.:

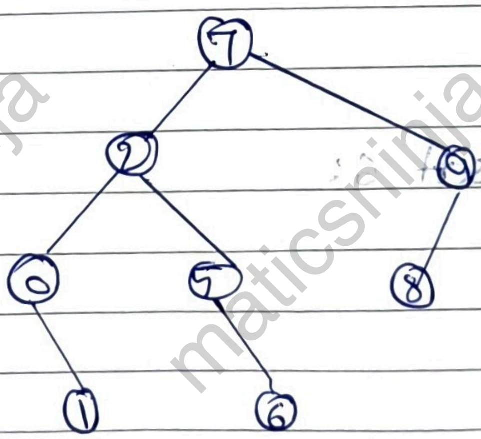
6] Insert 6 :



7] Insert 8 :



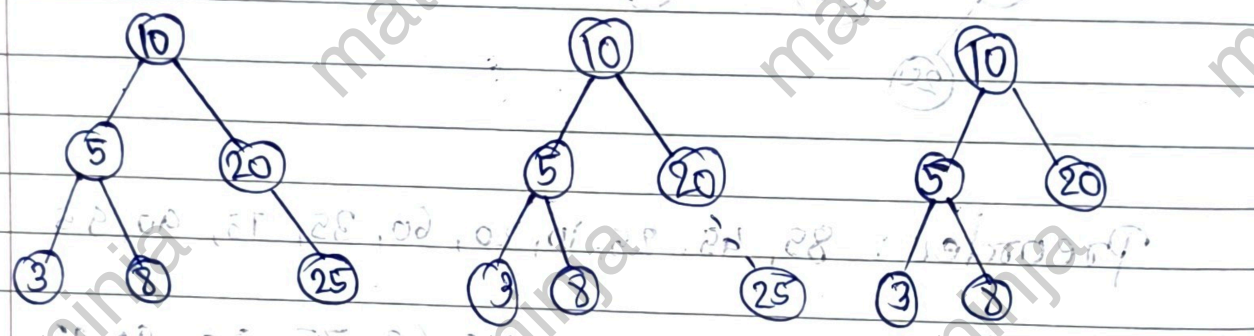
8] Insert 1 :



* Deletion of

1] leaf node
2] node with one child
3] node with two children

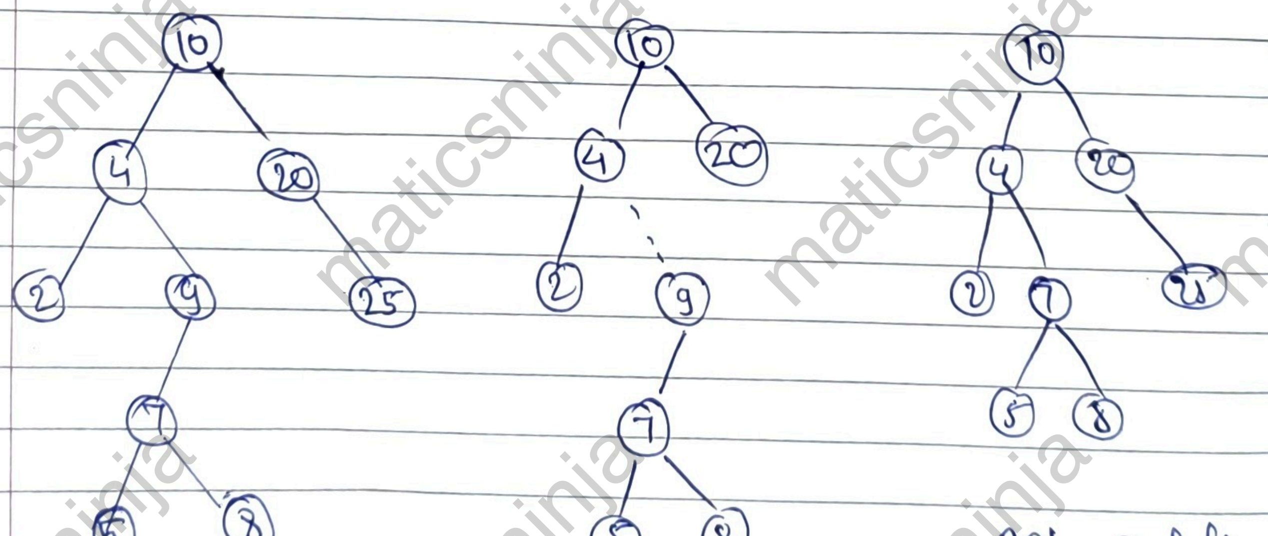
i]



Before Deletion   Deletion   After Deletion

Fig: Deletion of leaf node.

ii]



Before Deletion   Deletion.   After Deletion

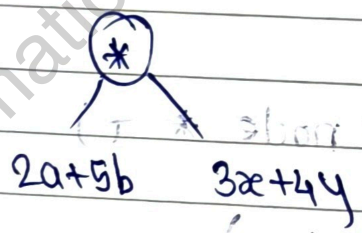
Fig - Node with one child.

⇒ Applications of Trees

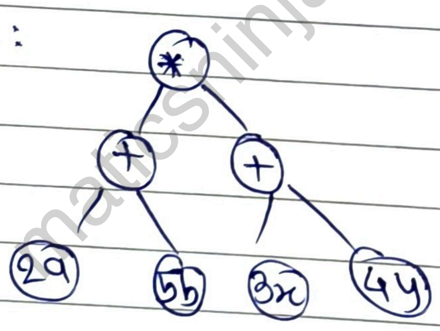
I) Expression Tree

$(2a+5b)(3x+4y)$

Step 1:



Step 2:



Step 3: