Topic

Unit No.

* Construct a Binary Tree from Preorder and Inorder Traversal

Example

Preorder - 1, 2, 4, 8, 9, 10, 11, 5, 3, 6, 7

Inorder - 8, 4, 10, 9, 11, 2, 5, 1, 6, 3, 7

⇒ As we know,

PreOrder ⇒ Root, Left child, Right child (DLR)

& InOrder ⇒ Left Root Right [LDR]

∴ Step 1 from given Preorder Traversal '1' is root node & Inorder Traversal gives left & Right child of '1' i.e.

8, 4, 10, 9, 11, 2, 5 (1) 6, 3, 7

Left Right

Step 2 ⇒ Step (1) will be repeated for Newly Left & Right child with the help of given preorder & Inorder traversal.

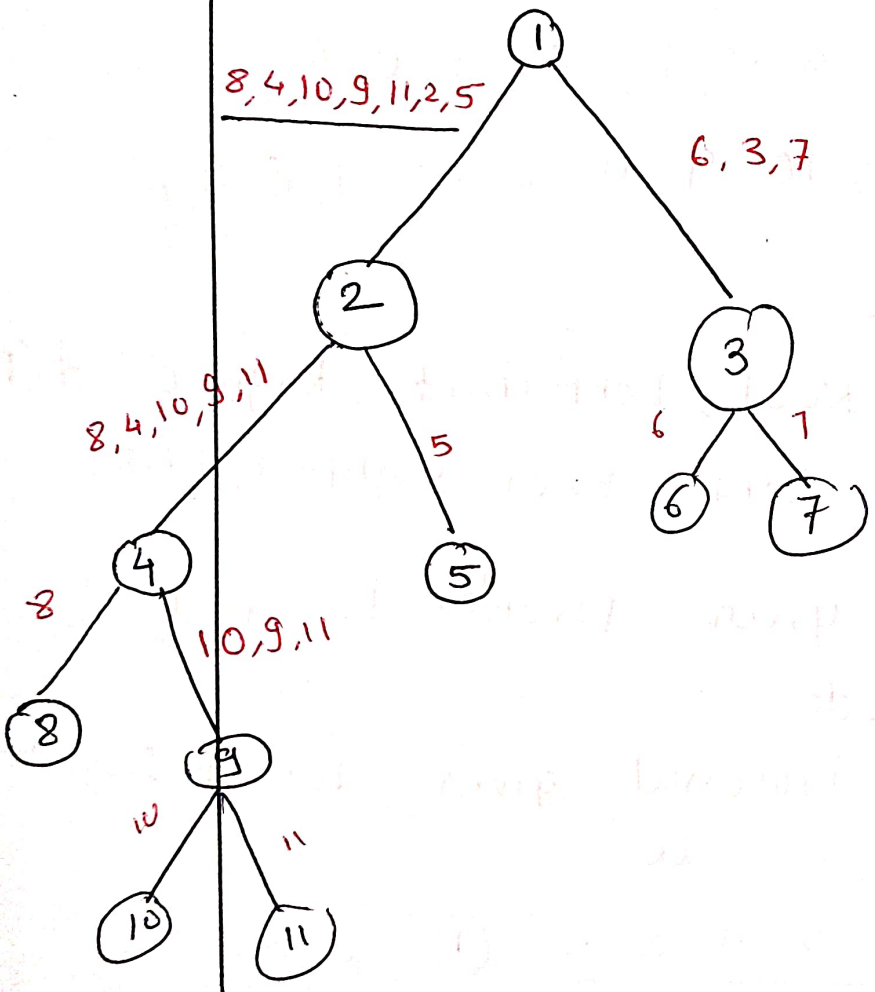
Topic

Unit No.

Preorder ⇒ 1, 2, 4, 8, 9, 10, 11, 5, 3, 6, 7

Inorder ⇒ 8, 4, 10, 9, 11, 2, 5, 1, 6, 3, 7

L R



Topic

Unit No.

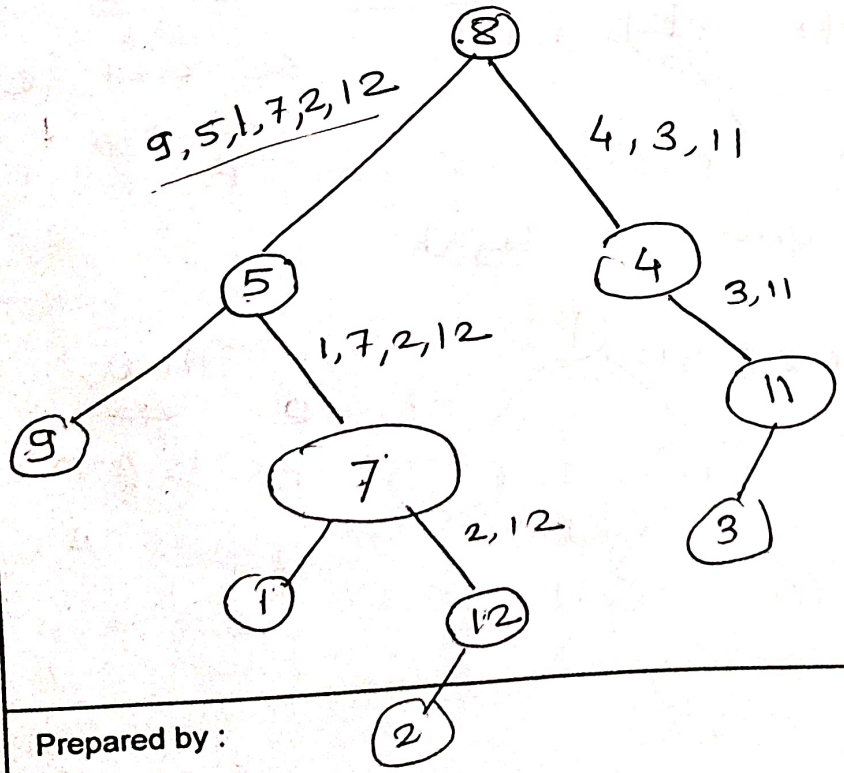
Construct Binary Tree from postorder and Inorder Traversal

Postorder - 9, 1, 2, 12, 7, 5, 3, 11, 4, 8
(Left, Right, Root)

Inorder - 9, 5, 1, 7, 2, 12, 8, 4, 3, 11
(Left, Root, Right)

→ Step-I →
As we know in postorder traversal Root node is last then "8" is root node of the new tree

9, 5, 1, 7, 2, 12 | 8 | 4, 3, 11
Left child | Root | Right child



Topic

Unit No.

* Construct Binary Tree from Preorder and Postorder Traversal (Unique possible Full BT)

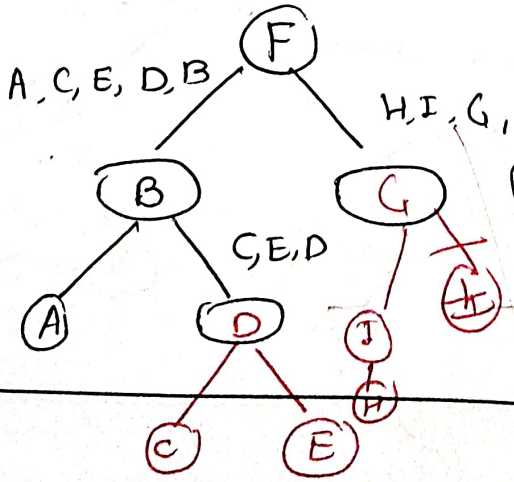
(Only with these two it is not possible to construct unique Binary Tree).

PreOrder - F B A D C E G I H (Root, L, R)

PostOrder - A C E D B H I G F (L, R, Root)

⇒ As in postorder Last is Root i.e. F

Therefore, Preorder First is root is root. & Acc. to for given. & check it's successor (Next) of F is preorder is B as it's Left child so all elements in postorder of 'B' is left & after B Right.



Now,

Pre B A D C E
post → A C E D B } skewed

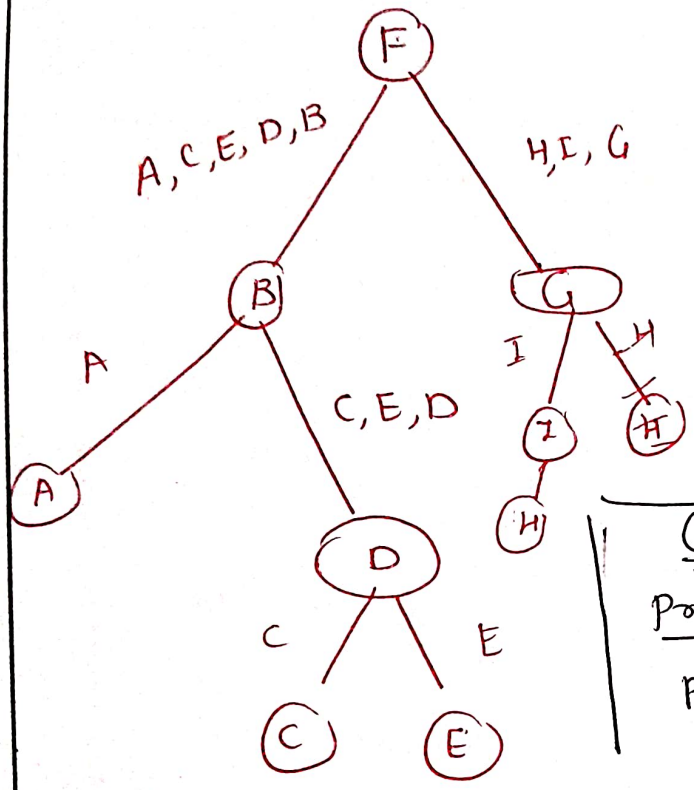
Pre → B C E
post - C E B } skewed

Prof. G I H
post - H I G

Topic

Unit No.

F B A D C E G I H



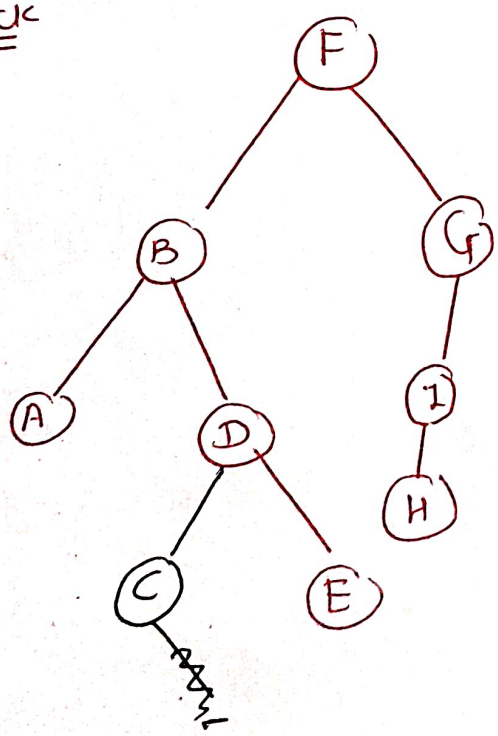
Given

preorder F B A D C E G I H

postorder A C E D B H I G F

only left

Trick



preorder → Right side

& postorder → Left side of currently added or Crear)

Topic

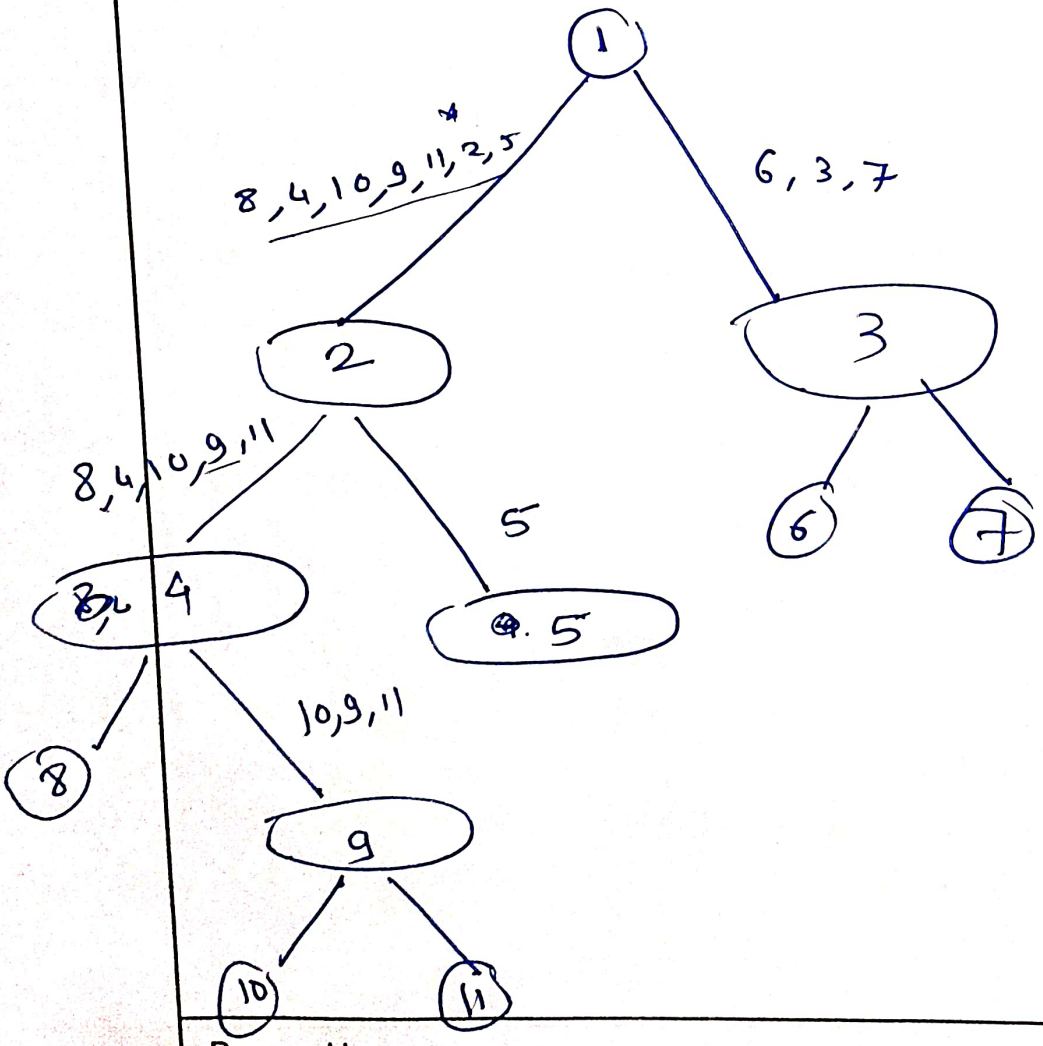
Unit No.

* Construct A Binary Tree from Preorder & Inorder

Preorder → 1, 2, 4, 8, 9, 10, 11, 5, 3, 6, 7

Inorder - 8, 4, 10, 9, 11, 2, 5, 1, 6, 3, 7

as '1' is root because Preorder



Topic Tree (BST) Unit No. 02

Binary Search Tree

~~then~~ - A BST is a BT which is either empty or in which each node contains a key that satisfies the following conditions:

(i) All keys (if any) in the left subtree of the root is ~~less than~~ precede the key in the root.

(ii) The key in the root precedes all keys (if any) in its right subtree.

(iii) The right and left subtrees are again search trees

In other words, BST is a BT in which all the keys having values less than root will be in the left subtree and those greater than root lie on right subtree.

Topic

Unit No.

Algorithm creating a BST

① Read the key value & store it in node say 'p'

② if (root == NULL i.e. if tree is currently empty then attach 'p' to root & goto step ④

i.e. root = p

③ q = root

if (p → info < q → info) then

if (q → left == NULL) then attach p to q → left and goto step ④

i.e. q → left = p

else

q = q → left and goto step ③

(i.e. if new element to be inserted is less than root element then move to left- once left- becomes NULL attach new node there)

else

if (q→ right == NULL) then

attach p to q→ right & goto step (4)

i.e. q→right = p

else

q = q→right

and goto step (3)

(if new element is greater than root value, move to right and connect the node there)

Step (4) Repeat step 1-3 for all the nodes

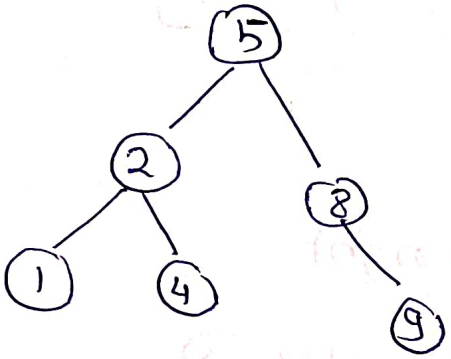
Step (5) stop

Topic

Unit No.

Create a BST for the input elements:

5, 2, 8, 4, 1, 9



Topic Unit No.

Delehion In BST

For Delehion of Node en BST we must consider three Cases,

① Leaf Node

These are the easiest, since what we do is delete the leaf and set the pointer of this nodes parent to NULL

(11) Nodes with a single child

These are also fairly simple, since we redirect pointer from nodes parent to the node's child and then delete the node.

(111) Nodes having both children

Here some arrangement is needed. what we can do is, replace the node being deleted by the rightmost node

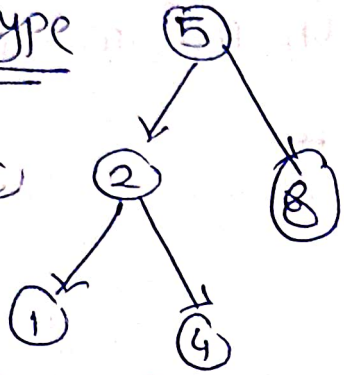
Topic

Unit No.

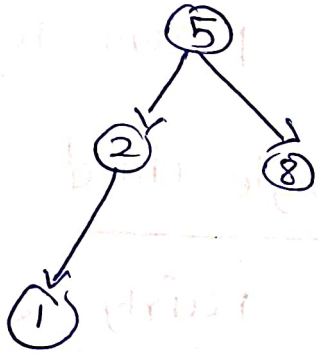
in its left subtree

Ist type

Example ①



Lets say 4 is deleted.



Example-②