L-33 Tree

Monday, April 25, 2022 9:51 AM

Properties of Tree

(1) There is one and only path between every pair of Vertices in a Tree.

(2) If In a graph G, There is One & - - - - - - - - - - - -. Then G is a Tree graph.

(3) A Tree with n-Vertices has (n-1) Edges.

No. of Edges in a Tree = n-1 where n = No. of Vertices.

Minimally Connected graph A Connected graph G is said to be minimally Connected if removal of Any Edge from G, It become disconnected Graph.

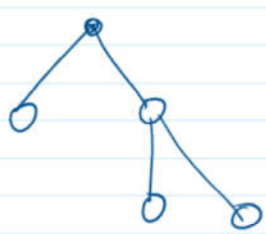
Note ! A Tree is Minimally Connected graph.

(2) A graph is a Tree iff it is Minimally Connected.

(4) In a non-Trivial Tree, There are atleast Two pendent Vertices.

$d(a) = d(b) = 1$

Both a & b are Pendent Vertices



Labeled Tree ↦ **Definition :** A Tree is said to be labeled in which every vertex of Tree has assigned a unique label.

Labeled tree is usually used to construct expression Tree. Any algebric expression can be represented with the help of labeled binary tree. For this root of tree is labeled with the central operator of main expression. The two offsprings of root are labeled with central operator of expression for left and right arguments respectively. If either argument is a constant or variable (instead of expression), this is used to label the corresponding offspring vertex. This process continues until expression is exhausted.

Root = Central Operator of Main Expression.

Example 5. Construct the tree of algebraic expression.

(*i*) $3 - (x + (6 \times (4 \div (2-3))))$

$3 - (x + (6 \times (4 \div (2-3))))$

$$3 - (x + (6 \times (4 \div (2-3))))$$

Left Argument

Central operator

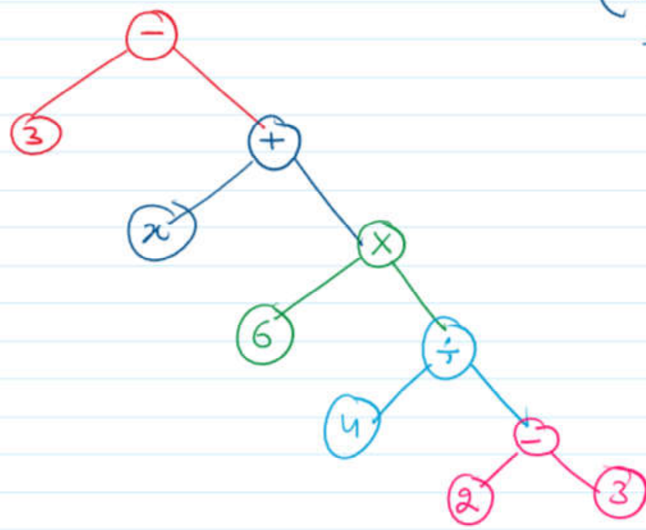
Right Argument

$(x+(6\times(4\div(2-3))))$

$6\times(\quad)$

$4\div(\quad)$

$(2-3)$



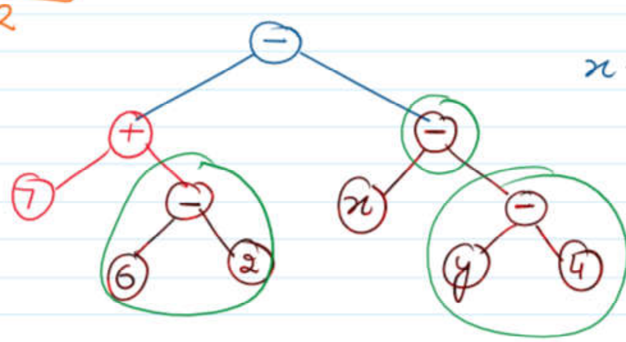
$(7 + (6-2)) - (x - (y-4))$

L

R

$7+(6-2)$

$x-(y-4)$

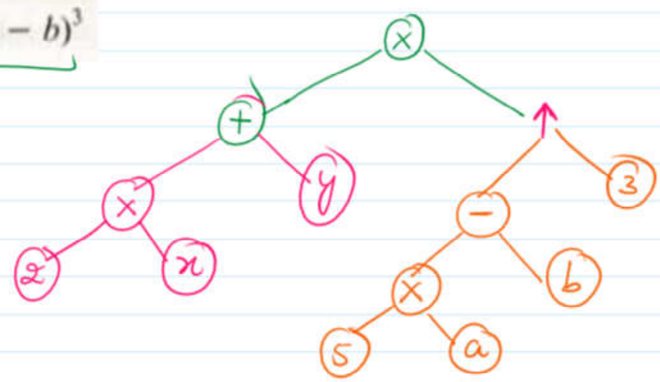


$$[7+(6-2)] - [x-(y-4)]$$

Ex

$((2 \times x) + y) \times ((5 \times a) - b)^3$

$(2\times x)+y$



Traversal of Binary Tree :→ To Visit each Vertex of Tree Exactly Once.

Preorder : (1) Process the root R.

(2) Traverse the left subtree of R in preorder.

Preorder : (1) Process the root R.

(2) Traverse the left subtree of R in preorder.

(3) Traverse the right subtree of R in preorder.

Inorder : (1) Traverse the left subtree of R in inorder.

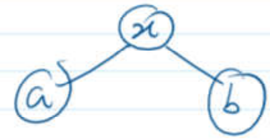
(2) Process the root R.

(3) Traverse the right subtree of R in inorder.

Postorder : (1) Traverse the left subtree of R in postorder.

(2) Traverse the right subtree of R in postorder

(3) Process the root R.



Preorder $x a b$

Inorder $a x b$

Post order $a b x$