SNS COLLEGE OF TECHNOLOGY

Coimbatore-35

An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A+' Grade

Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF MCA

UNIT II - TREES

19CAT602 – DATA STRUCTURES & ALGORITHMS

I YEAR I SEM

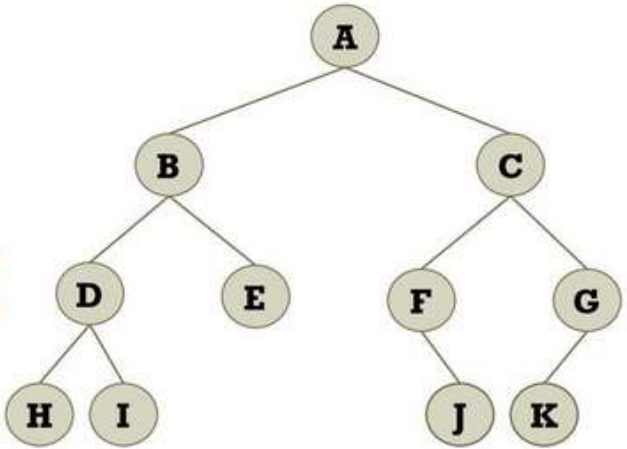
TOPIC 8 - Binary tree Traversals

Binary tree Traversal

- Traversal
  - The process of visiting each node in a tree

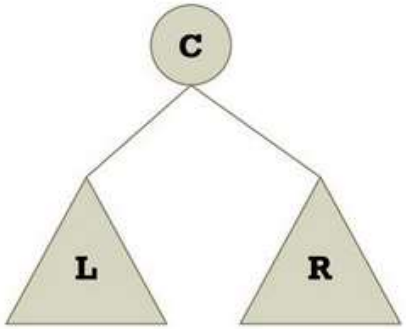
- Why the traversal necessary?
  - Checking whether insertions/deletions work well.
  - Searching a specific node.

- How to visit all nodes once?



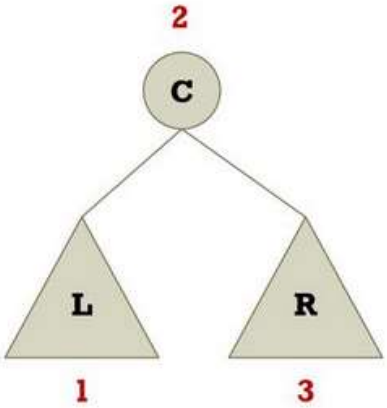
Traversal of Binary Tree

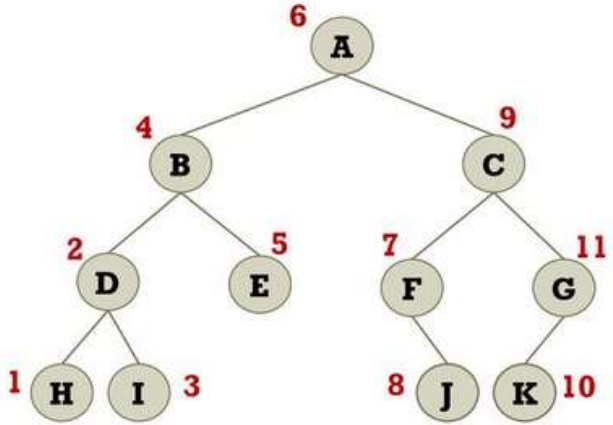
- Traversal methods
  - **Inorder traversal: LCR**
    - Visiting a left subtree, a root node, and a right subtree
  - **Preorder traversal: CLR**
    - Visiting the root node node before subtrees
  - **Postorder traversal: LRC**
    - Visiting subtrees before visiting the root node
  - **Level order traversal**



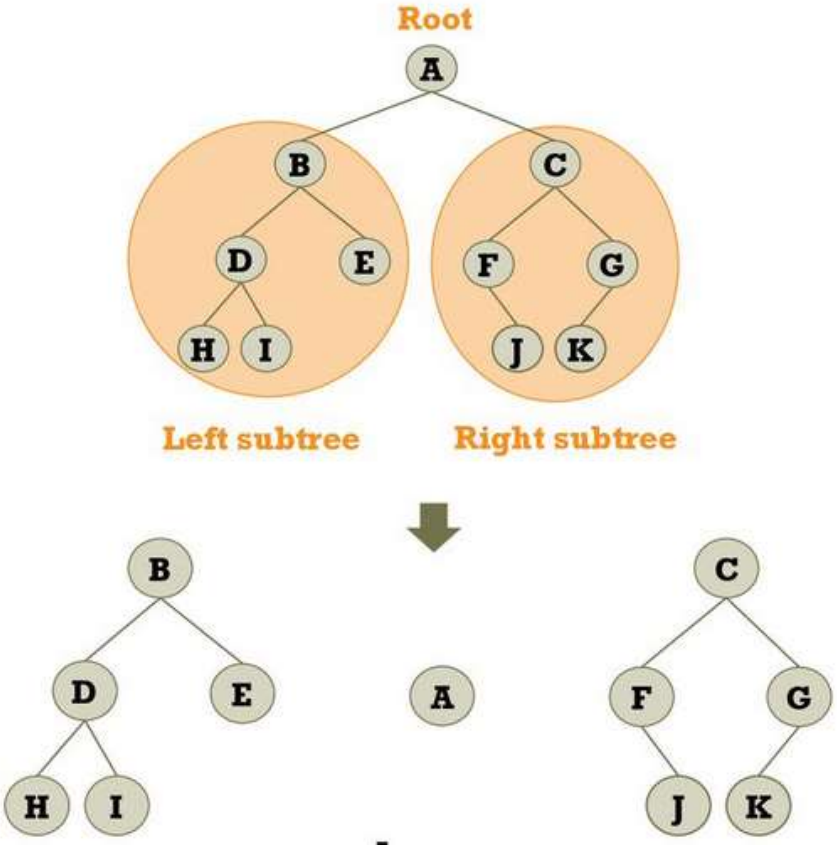
Inorder Traversal (LCR)

- Algorithm: LCR
  - Step 1: Visiting a left subtree
  - Step 2: Visiting the root node
  - Step 3: Visiting a right subtree

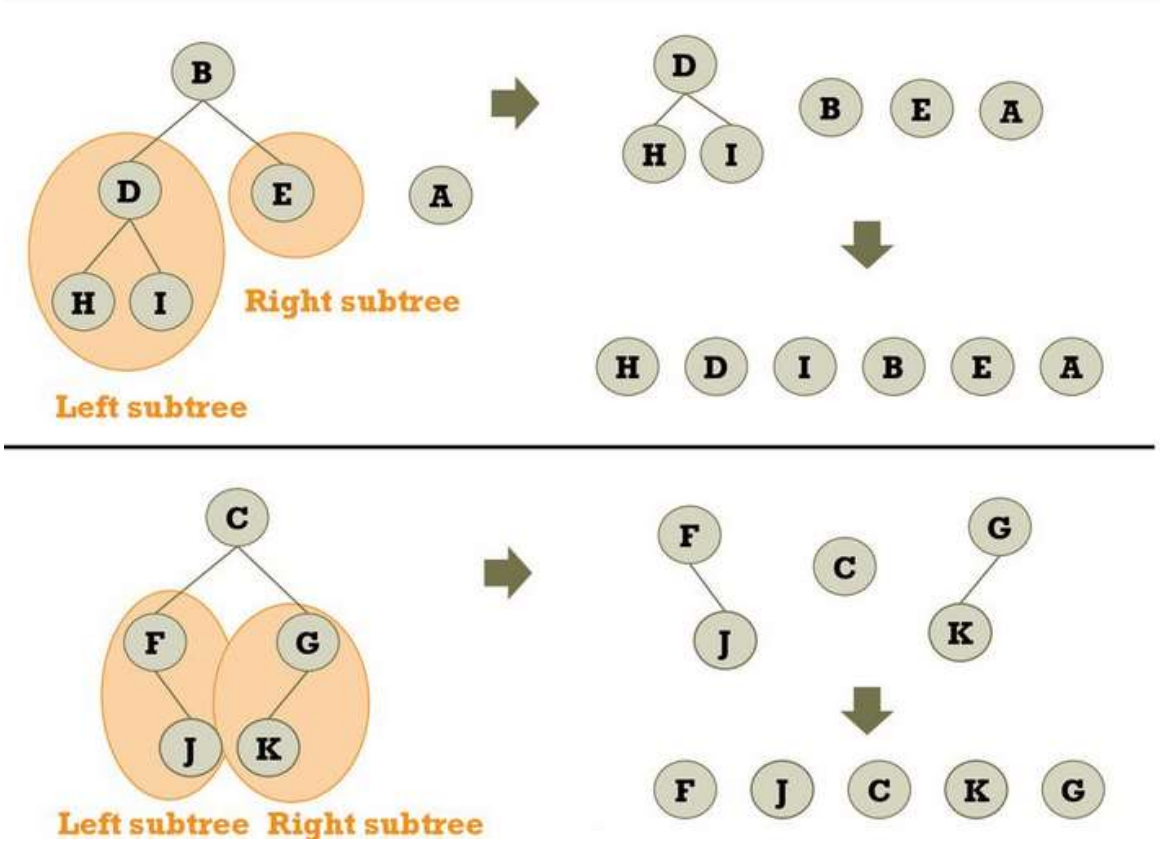




Inorder Traversal (LCR)



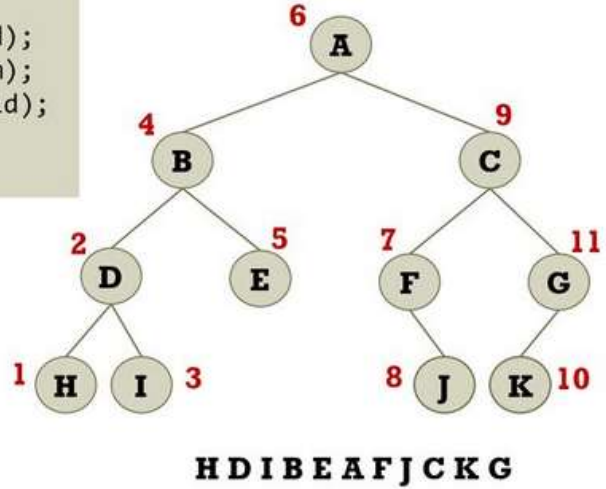
Inorder Traversal (LCR)



Inorder Traversal (LCR)

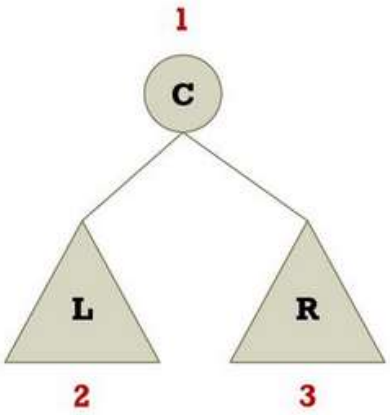
- Algorithm

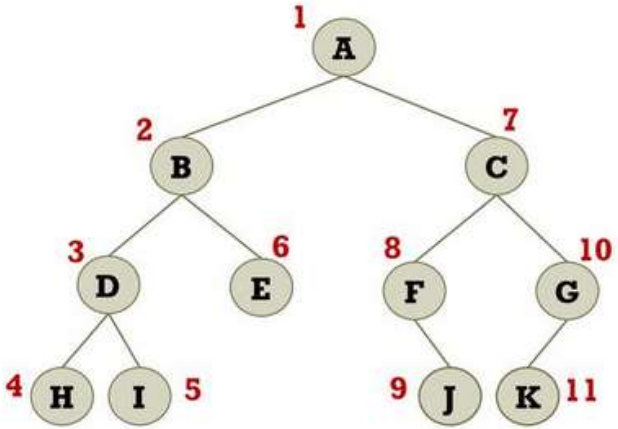
```
void Inorder(BTreeNode* root)
{
    if (root != NULL)
    {
        Inorder(root->left_child);
        printf("%d ", root->item);
        Inorder(root->right_child);
    }
}
```



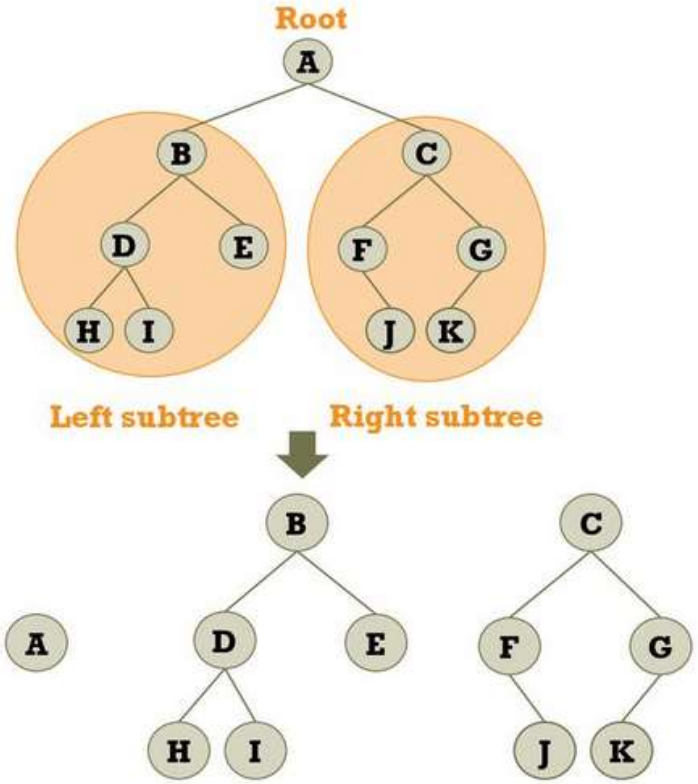
Preorder Traversal (CLR)

- Algorithm: CLR
  - Step 1: Visiting the root node
  - Step 2: Visiting a left subtree
  - Step 3: Visiting a right subtree

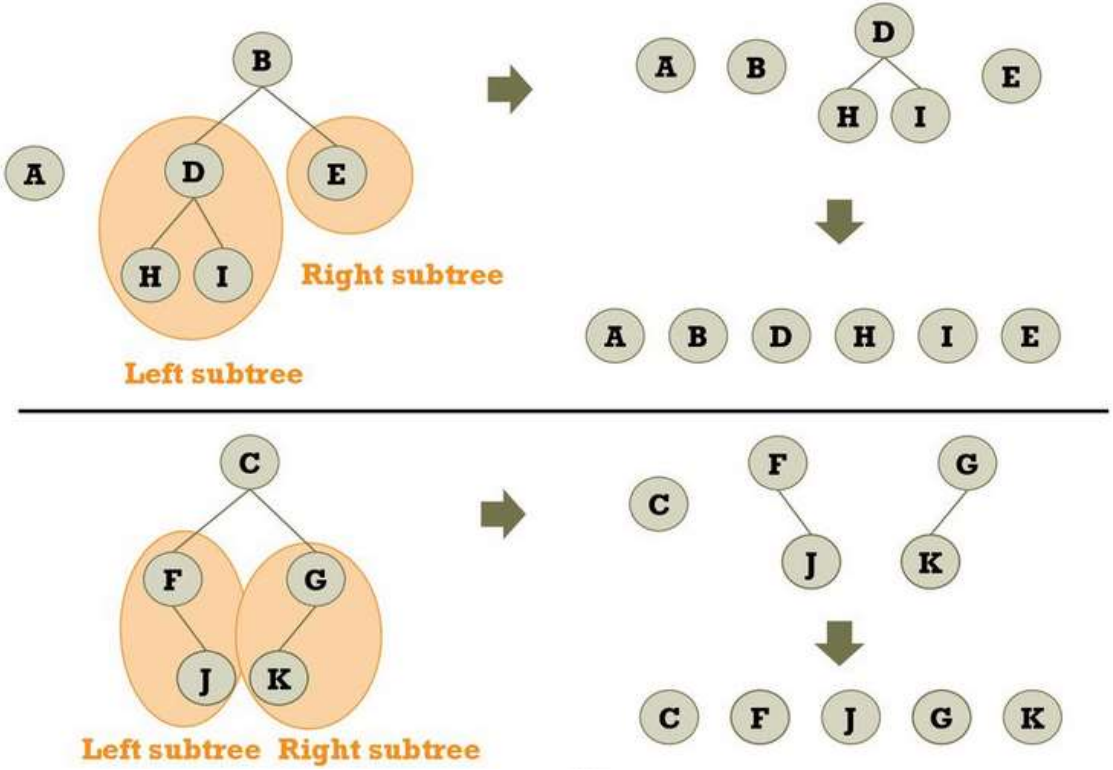




Preorder Traversal (CLR)



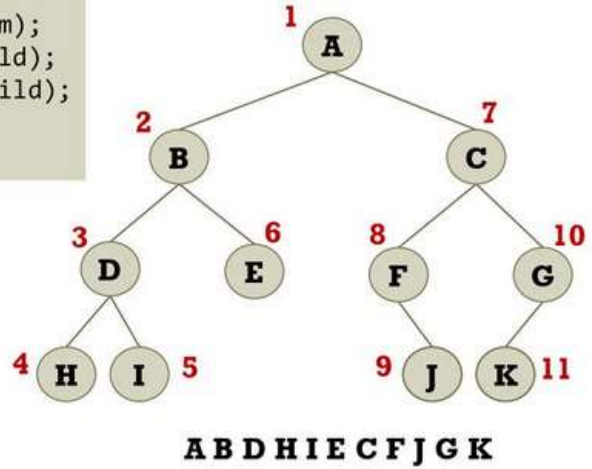
Preorder Traversal (CLR)



Preorder Traversal (CLR)

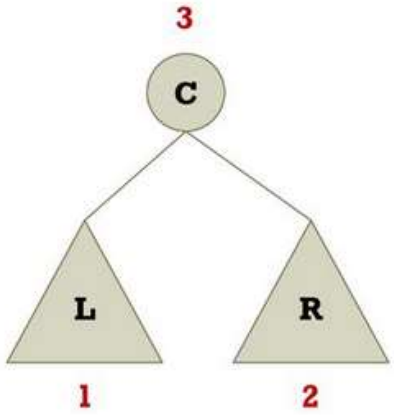
- Algorithm: CLR

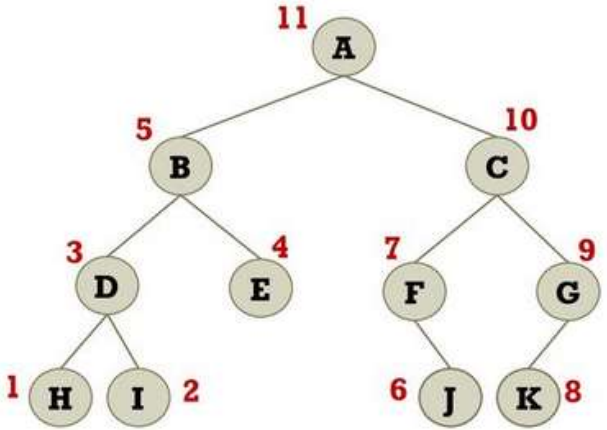
```
void Preorder(BTreeNode* root)
{
    if (root != NULL)
    {
        printf("%d ", root->item);
        Preorder(root->left_child);
        Preorder(root->right_child);
    }
}
```



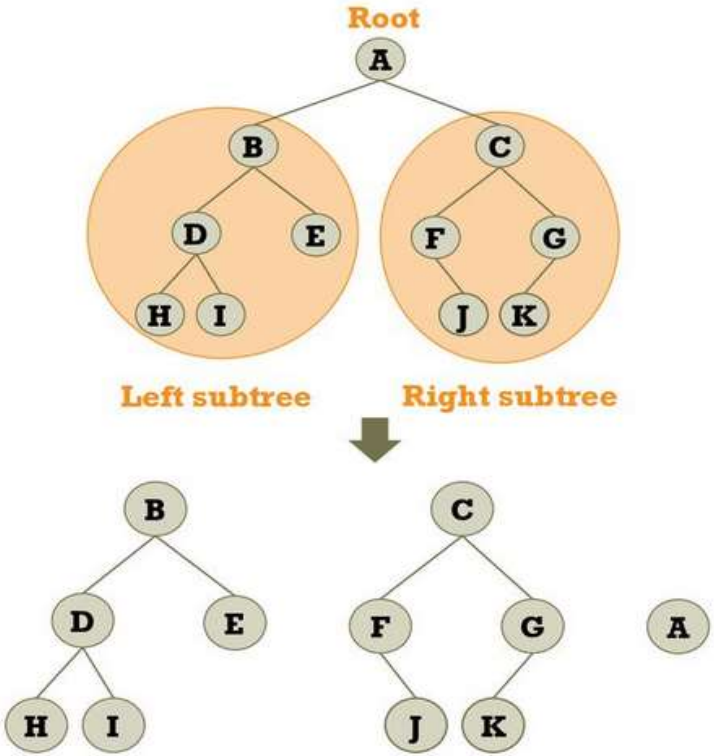
Postorder Traversal (LRC)

- Algorithm: LRC
  - Step 1: Visiting a left subtree
  - Step 2: Visiting a right subtree
  - Step 3: Visiting the root node

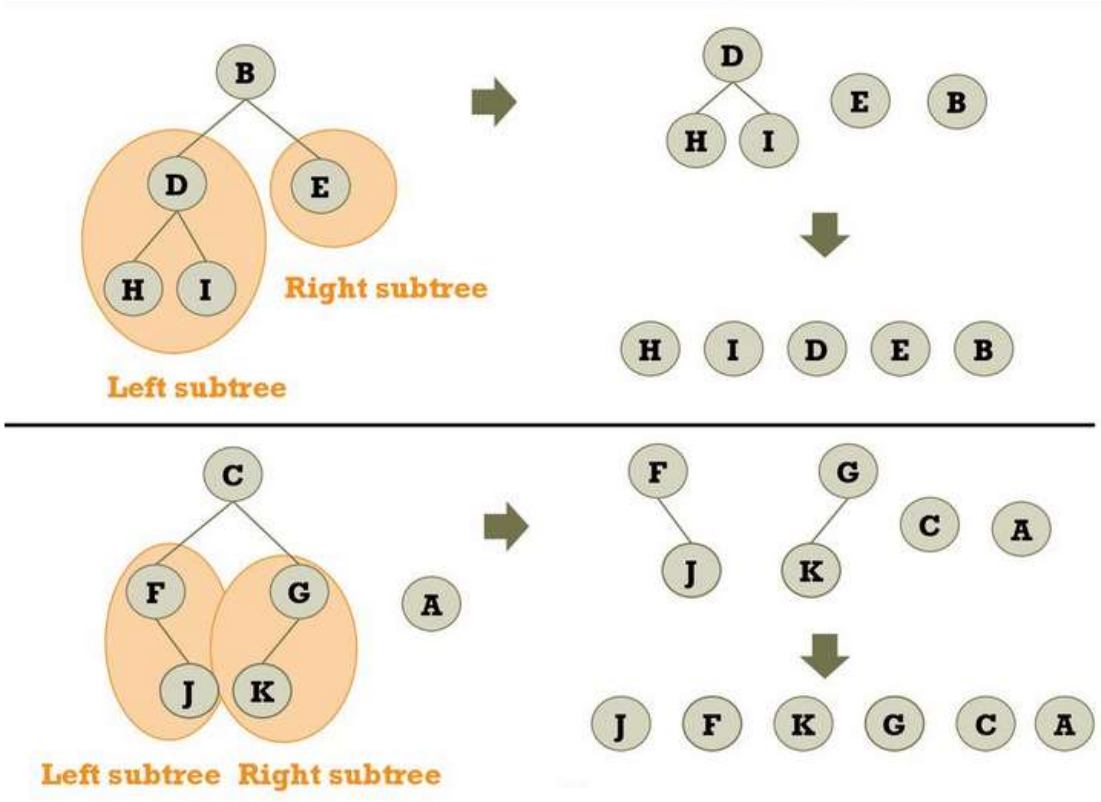




Postorder Traversal (LRC)



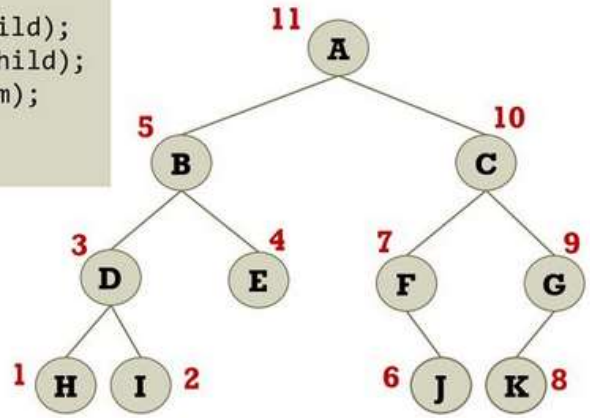
Postorder Traversal (LRC)



Postorder Traversal (LRC)

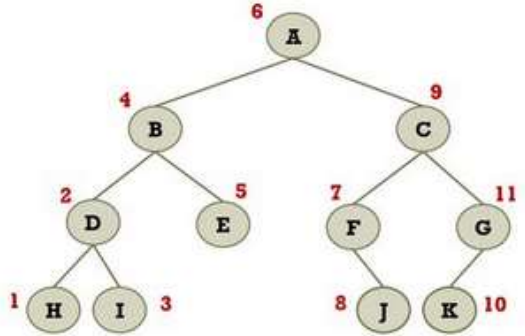
- Algorithm: LRC

```
void Postorder(BTreeNode* root)
{
    if (root != NULL)
    {
        Postorder(root->left_child);
        Postorder(root->right_child);
        printf("%d ", root->item);
    }
}
```

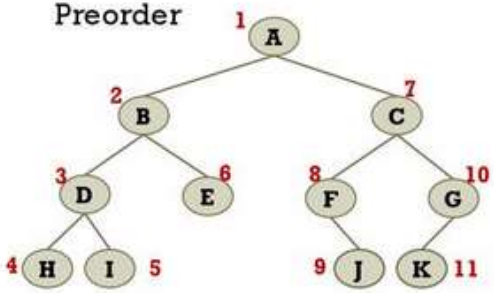


H I D E B J F K G C A

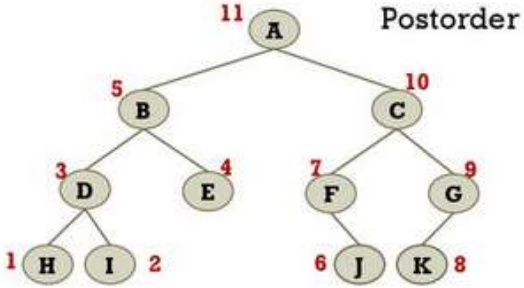
Inorder



Preorder

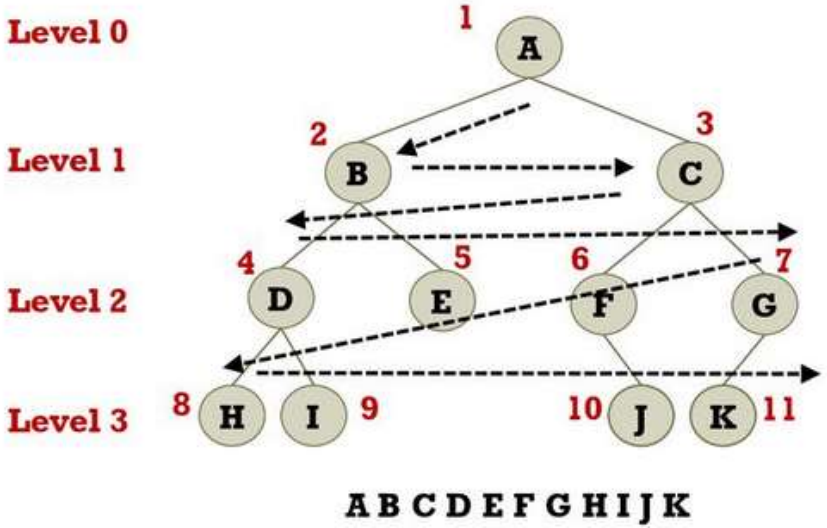


Postorder



Level Order Traversal

- Algorithm
  - For each level, visit nodes from the left to right direction.

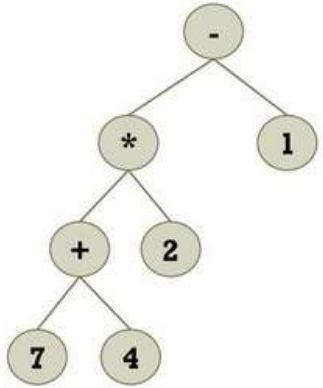


Binary Expression Tree

- Infix notation: X + Y
  - Operators are written in-between their operands.
  - Need extra information to make the order of evaluation of the operators clear.

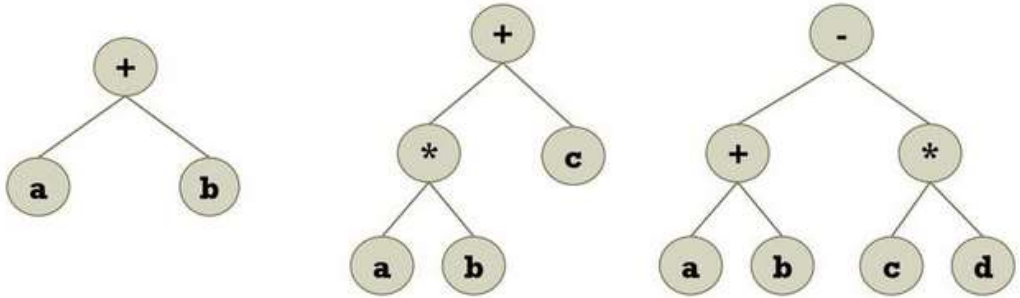
- Example: (7 + 4) * 2 – 1
  - The expression tree is easier to understand than infix notation.

(7 + 4) * 2 - 1 ➡



Binary Expression Tree

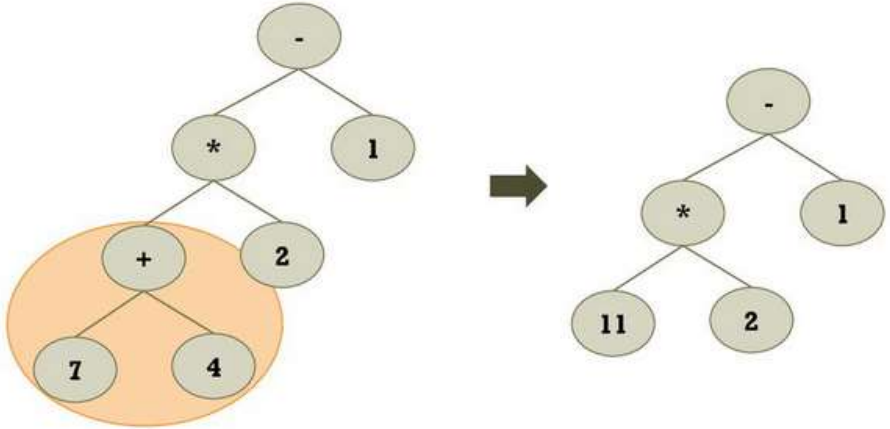
- Definition
  - Representing an expression as a tree.
    - Non-leaf node: operator, leaf node: operand



	a+b	a*b+c	a+b-c*d
Infix	a+b	a*b+c	a+b-c*d
Prefix	+ab	+*abc	-+ab*cd
Postfix	ab+	ab*c+	ab+cd*-

Calculating Expression Tree

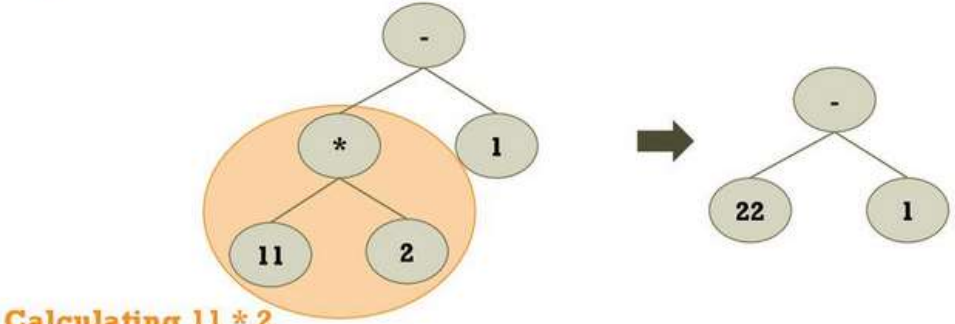
- Calculate 7 + 4 and update the node.



Calculating 7 + 4

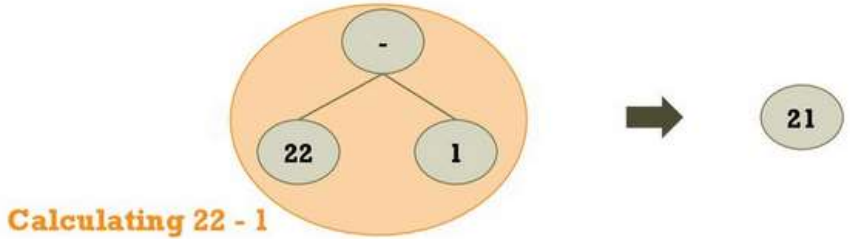
Calculating Expression Tree

- Calculate 11 * 2 and update the node.



Calculating 11 * 2

- Calculate 22 - 1 and update the node.



Calculating 22 - 1

Reference

1. Tanaenbaum A.S., Langram Y. Augestein M.J “Data Structures using C”, Pearson Education , 2008.
2. https://www.studytonight.com/data-structures
3. www.tutorialpoint.com
4. https://www.javatpoint.com/tree