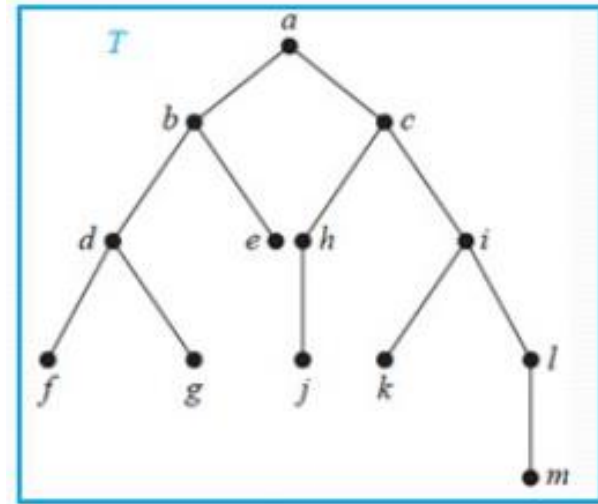# CHAPTER 11
# SECTION 11.3
# TREE TRAVERSAL

# ORDERED ROOT TREE APPLICATIONS

- Used to store information

- Procedures for visiting each vertex of an ordered rooted tree to access data



- Used to represent various types of expressions (arithmetic expressions involving numbers, variables, and operations)

# UNIVERSAL ADDRESS SYSTEMS

To order the vertices of an ordered rooted tree we must first label all the vertices. We do this recursively:

1. Label the root with the integer 0. Then label its k children (at level 1) from left to right

with 1, 2, 3,…, k.

2. For each vertex v at level n with label A, label its $k_v$ children, as they are drawn from left to right, with A.1, A.2,…, A. $k_v$
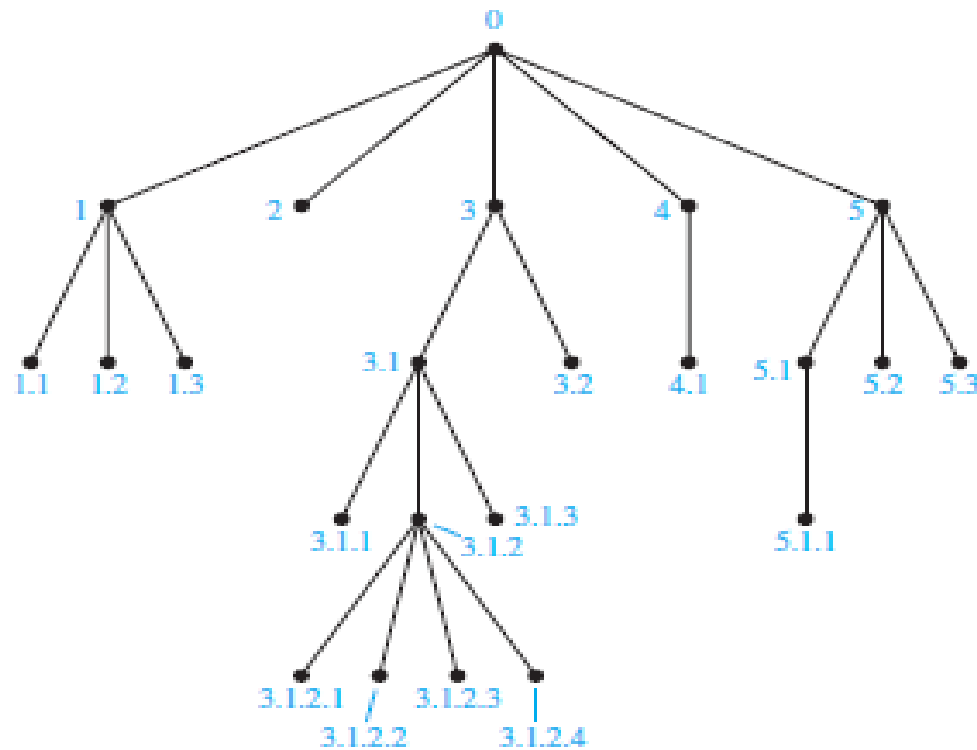
# UNIVERSAL ADDRESS SYSTEMS



**FIGURE 1** The universal address system of an ordered rooted tree.

# TRAVERSAL ALGORITHMS

A traversal algorithm is the procedure of systematically visiting each vertex of an ordered rooted tree.

- Tree traversals are defined recursively
- Three types of algorithms for tree traversals are-

1. Pre-order Traversal
2. In-order Traversal
3. Post-order Traversal

# TRAVERSAL ALGORITHMS (DEFINITION)

1. Pre-order Traversal
2. In-order Traversal
3. Post-order Traversal

**Definition 1** Let $T$ be an ordered rooted tree with root $r$. If $T$ consists only of $r$, then $r$ is the *preorder traversal* of $T$. Otherwise, suppose that $T_1, T_2, \ldots, T_n$ are the subtrees at $r$ from left to right in $T$. The *preorder traversal* begins by visiting $r$. It continues by traversing $T_1$ in preorder, then $T_2$ in preorder, and so on, until $T_n$ is traversed in preorder.

# PRE-ORDER TRAVERSAL

**Rules:** Visit – Step 1: Root

Step 2: Left

Step 3: Right

***Example:*** In which order does a preorder traversal visit the vertices in the ordered rooted tree T shown in Figure 3?
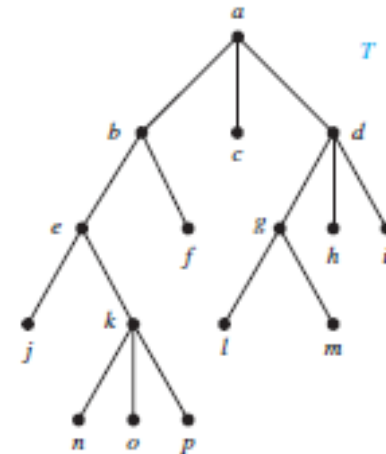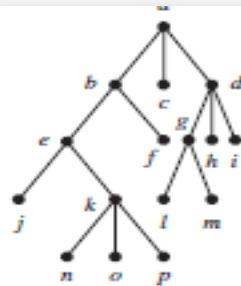
**FIGURE 3** The ordered rooted tree *T*.

# PRE-ORDER TRAVERSAL

**Solution:**

The Pre-order traversal
of T is ----------
a, b, e, j, k, n, o, p, f , c,
d, g, l, m, h, i.

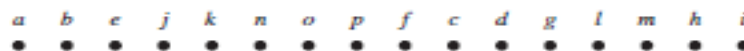

Preorder traversal: Visit root,
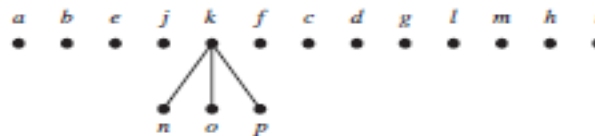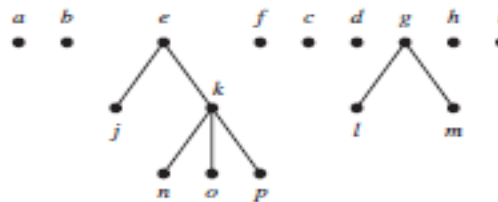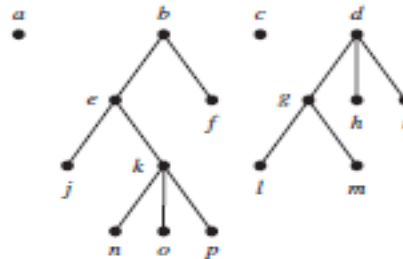visit subtrees left to right

FIGURE 4    The preorder traversal of *T*.

# IN-ORDER TRAVERSAL

**Rules:** Visit – Step 1: Left

Step 2: Root

Step 3: Right

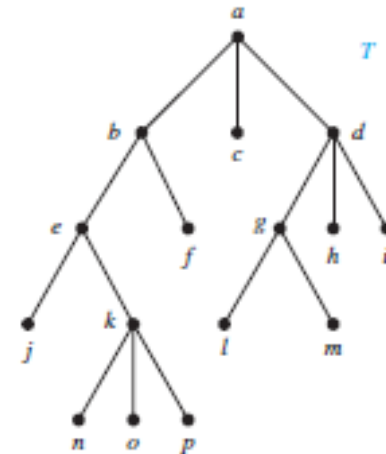***Example:*** In which order does a inorder traversal visit the vertices in the ordered rooted tree T shown in Figure 3?



**FIGURE 3** The ordered rooted tree *T*.

# IN-ORDER TRAVERSAL

**Solution:**

The in-order listing of the ordered rooted tree is------------

j, e, n, k, o, p, b, f , a, c, l, g, m, d, h, i.



Inorder traversal: Visit leftmost subtree, visit root, visit other subtrees left to right
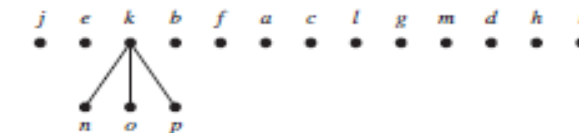
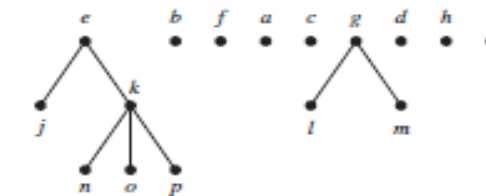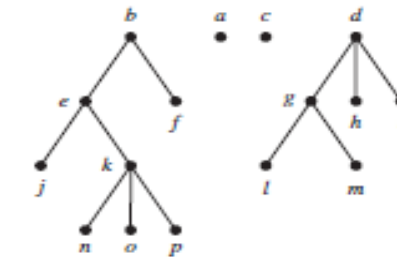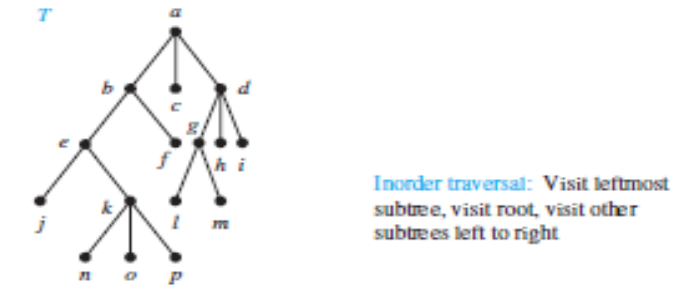**FIGURE 6**   The inorder traversal of *T*.

# POST-ORDER TRAVERSAL

**Rules:** Visit – Step 1: Left

                     Step 2: Right

                     Step 3: Root

*Example:* In which order does a postorder traversal visit the vertices in the ordered rooted tree T shown in Figure 3?
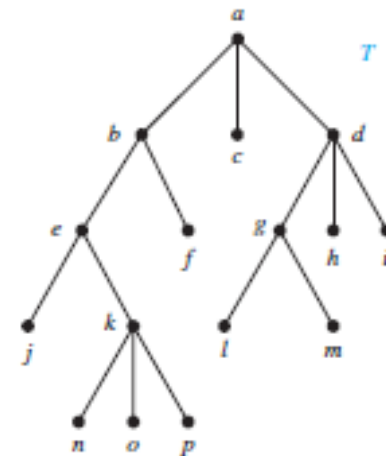


**FIGURE 3** The ordered rooted tree *T*.

# POST-ORDER TRAVERSAL

**Solution:**

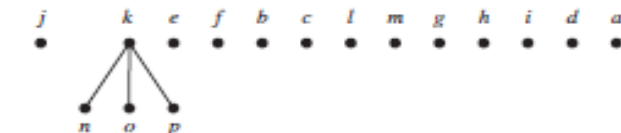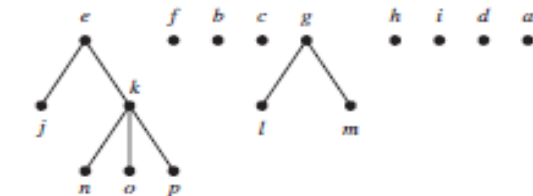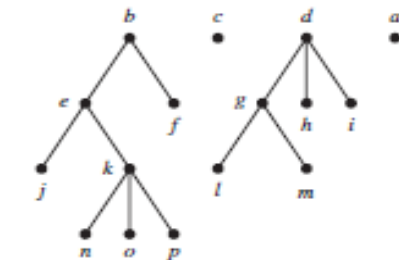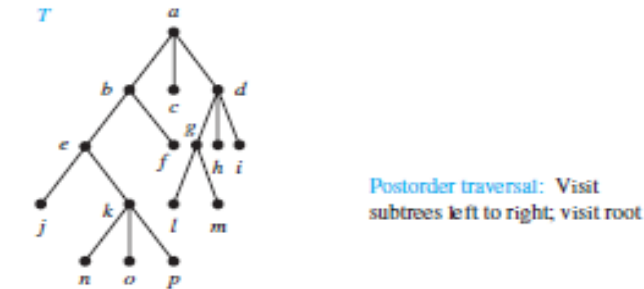The post-order traversal of T is j, n, o, p, k, e, f , b, c, l, m, g, h, i, d, a.



**FIGURE 8** The postorder traversal of *T*.

# PREFIX, IN-FIX & POSTFIX NOTATION

**<u>Rules:</u>**

- The representation of an arithmetic expression involving the operators + (addition), − (subtraction), ∗(multiplication), / (division), and ↑ (exponentiation).

- parentheses is used to indicate the order of the operations.

- The internal vertices represent operations, and the leaves represent the variables or numbers.

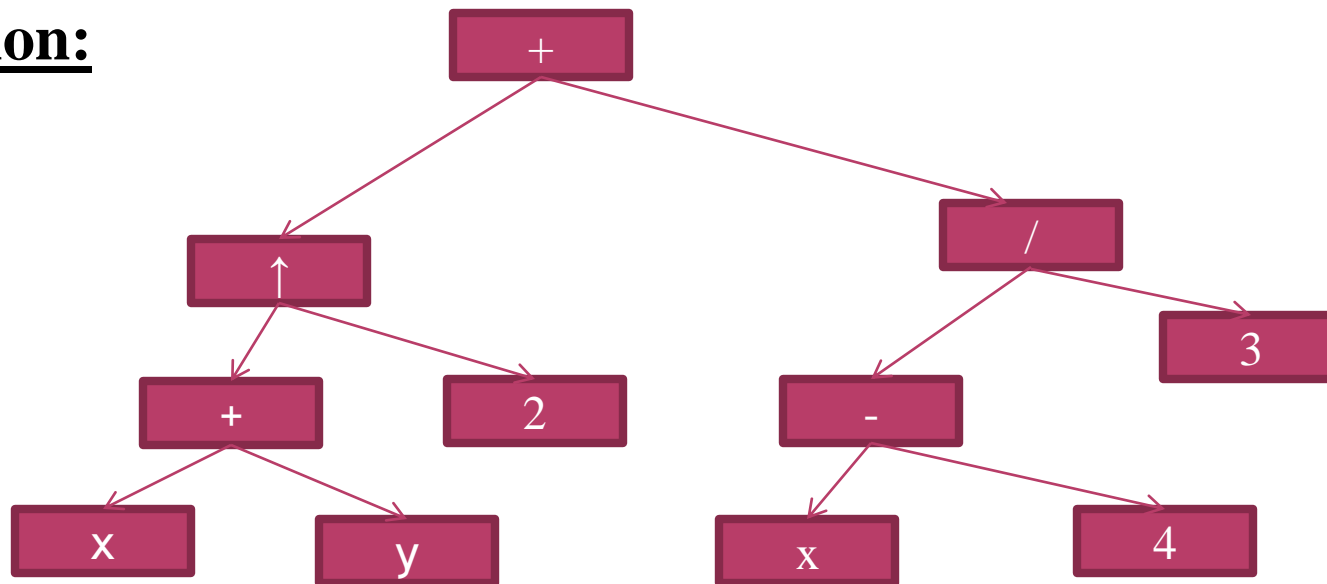- Each operation operates on its left and right subtrees (in that order).

# PREFIX, IN-FIX & POSTFIX NOTATION

**Example:** What is the prefix form for

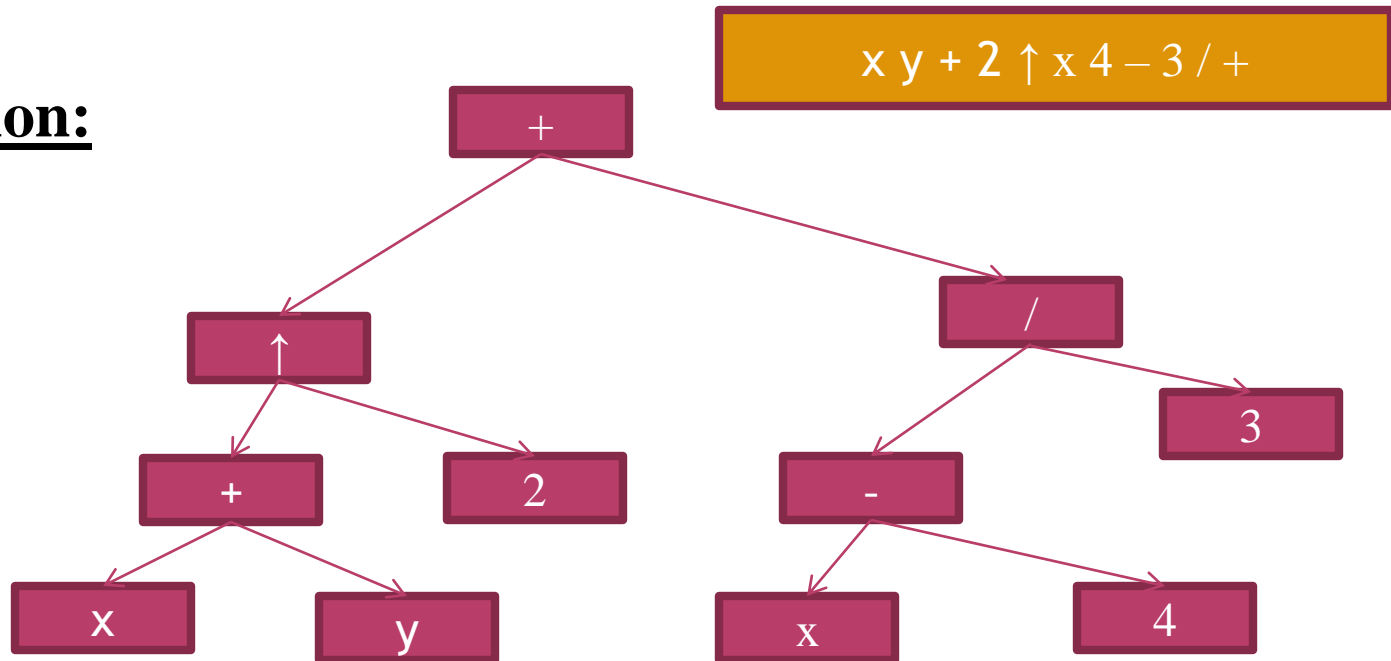$((x + y) \uparrow 2) + ((x - 4)/3)$?

+↑x y 2 /-x 4 3

**Solution:**

# PREFIX, IN-FIX & POSTFIX NOTATION

**Example:** What is the postfix form for

$((x + y) \uparrow 2) + ((x - 4)/3)$?
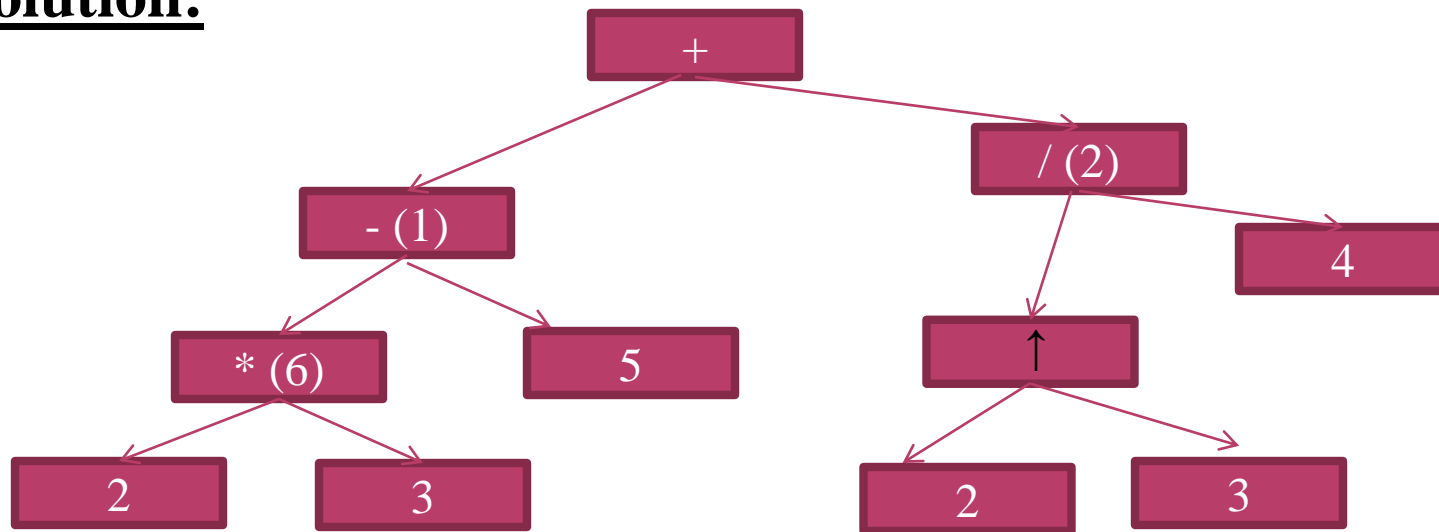
**Solution:**

x y + 2 ↑ x 4 − 3 / +

# PREFIX, IN-FIX & POSTFIX NOTATION

**Example:** What is the value of the prefix expression

$+ - * \, 2 \, 3 \, 5 / \uparrow 2 \, 3 \, 4$? (PREORDER- Root +L +R)

**Solution:**



```
+
├── - (1)
│   ├── * (6)
│   │   ├── 2
│   │   └── 3
│   └── 5
└── / (2)
    ├── ↑
    │   ├── 2
    │   └── 3
    └── 4
```
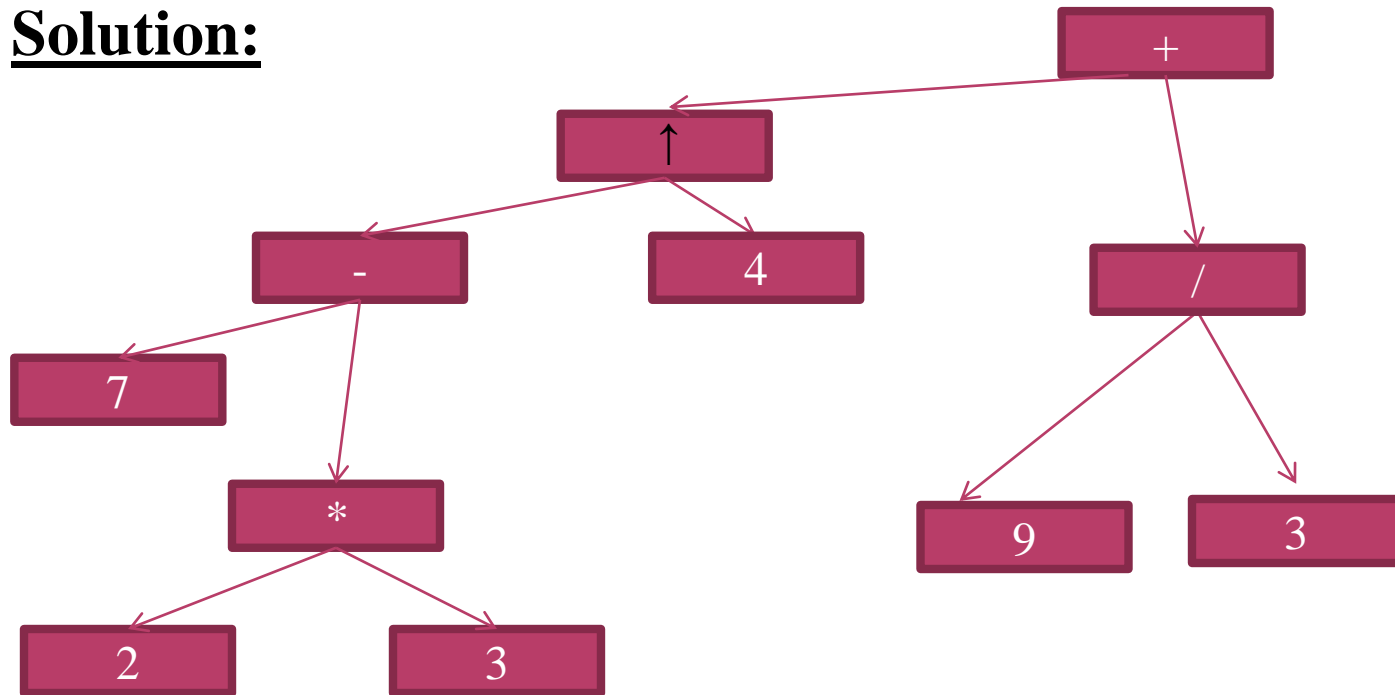
2^3=8   (8/4)=2    2*3=6

# PREFIX, IN-FIX & POSTFIX NOTATION

**Example:** What is the value of the postfix expression

7 2 3 ∗ −4 ↑ 9 3∕+? (PostORDER-  L +R+ Root )

**Solution:**



2*3=6 7-6=1 1^4=1 9/3=3 3+1=4