Tree Traversal

Section 10.3

Tree Traversal

- Process of "walking" on an ordered rooted tree
  - In a systematic fashion
- Has several strategies

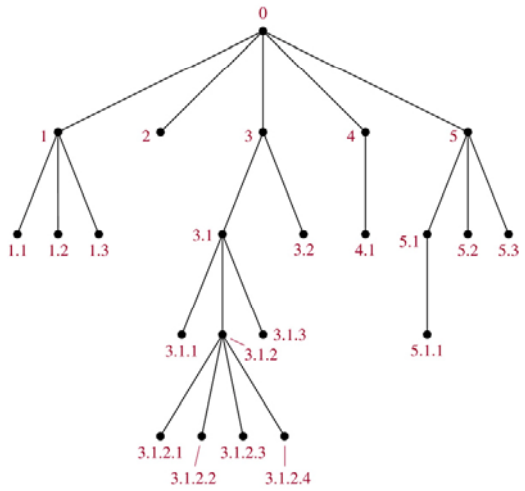
Why ordered tree?

- We require that because we want to "uniquely" identify each vertex
- Ordered tree has its vertices aligned properly
  - i.e., we have clear idea on what is the arrangement of the children

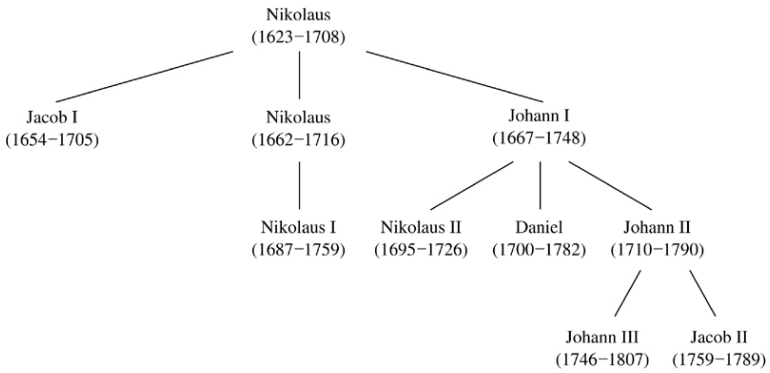
Universal Address System

- A way to uniquely identify vertices in a tree
- We label every vertex using the following rule
  - Label the root with the integer 0.
    - Then label its k children (at level 1) from left to right with $1,2,3,\ldots,k$
  - For each vertex v at level n with label A
    - We label its k_v children as they are drawn from left to right as $A.1, A.2, A.3, \ldots, A.k_v$

Example



Try it on the family tree



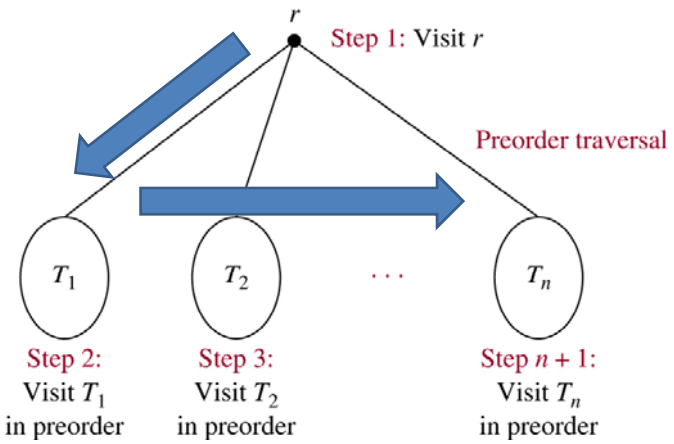
Back to traversal

- We wish to **"list"** vertices in the tree
  - in some particular order
- Several strategies
  - Preorder
  - Inorder
  - postorder

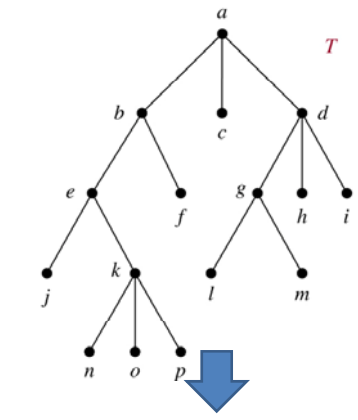
Preorder Traversal

- List the root
- then visit each subtree
  - Using the same method
    - (list root of the subtree then visit subtree of the subtree)
    - ...

Preorder traversal

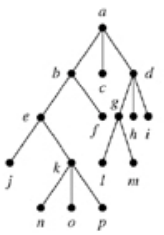


Example

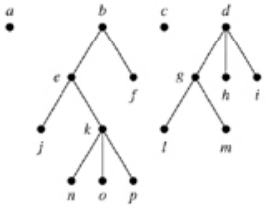


a b e j k n o p f c d g l m h i

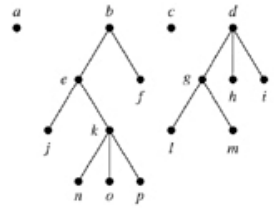
Step

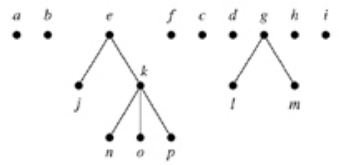


Preorder traversal: Visit root, visit subtrees left to right

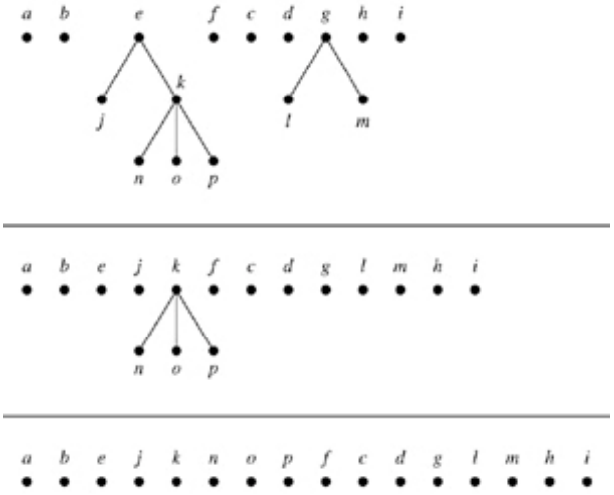


Step





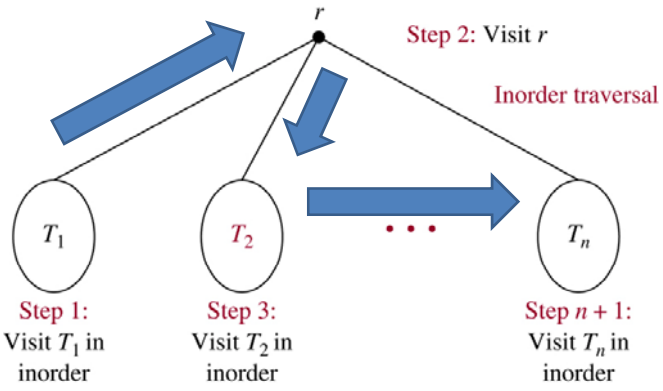
Step



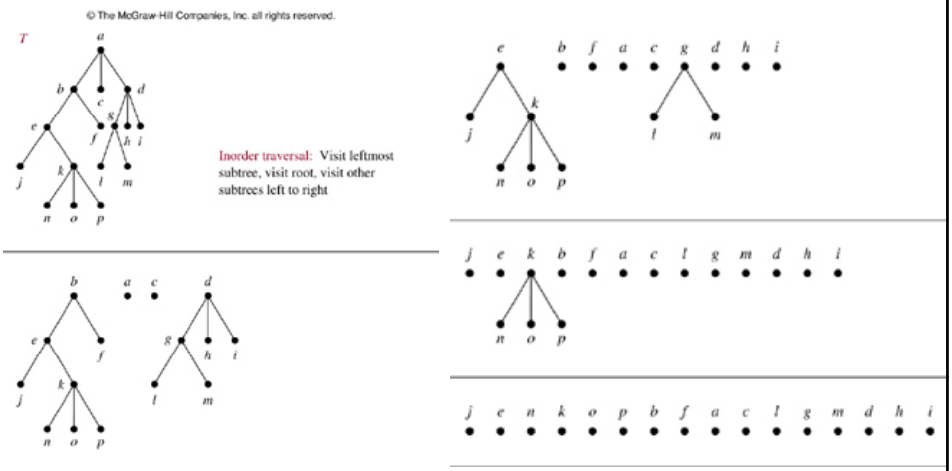
Preorder Traversal

- Visit the left subtree
- Then list the root
- Then visit the rest subtrees

Inorder Traversal



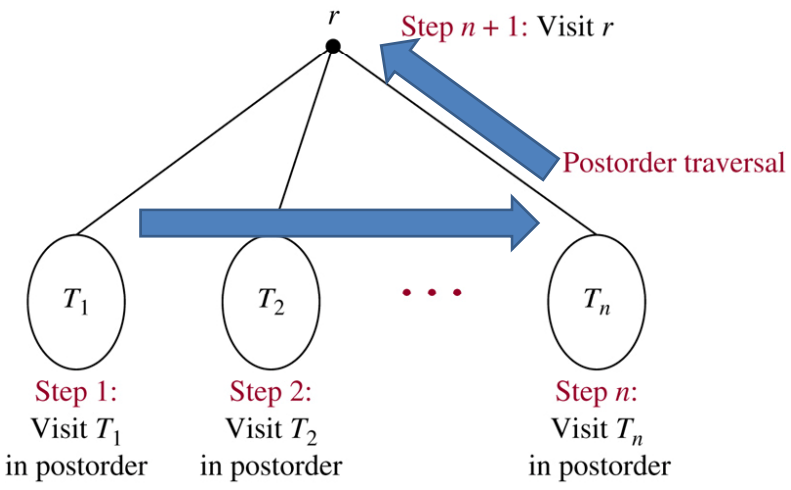
Example



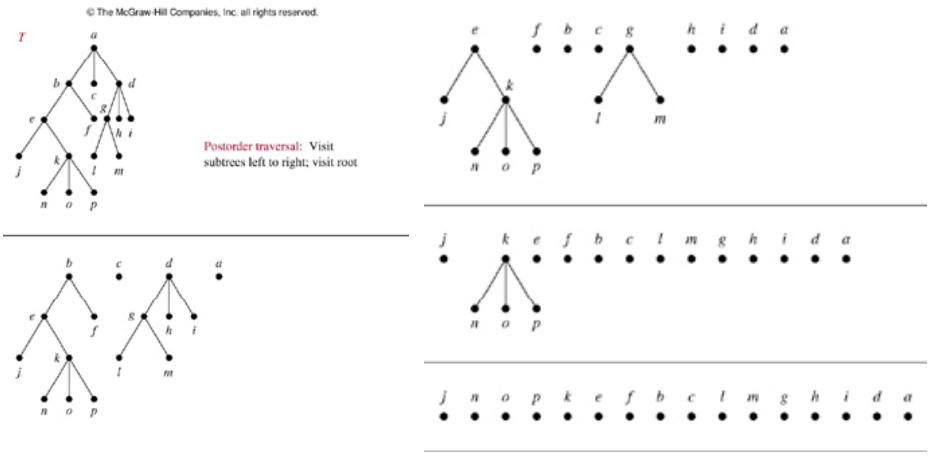
Postorder Traversal

- Visit each subtree
- Then list the root

Postorder Traversal

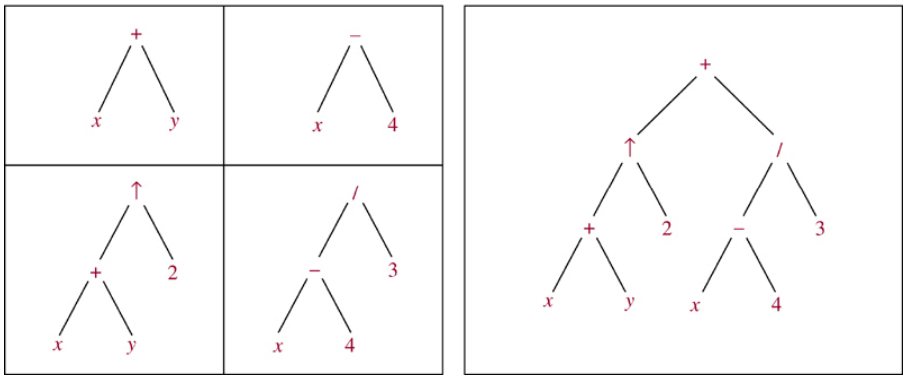


Example



Expression Tree

- Using a tree to represent mathematical expression

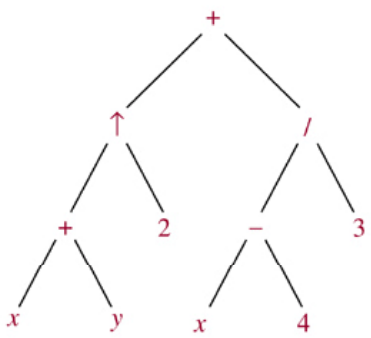


Infix, Prefix and Postfix notation

- By traversing an expression tree using inorder, preorder or postorder
  - We get infix, prefix or postfix notation of the expression

- Infix : an operator is in the middle
- Prefix: an operator is at the front
- Postfix: an operator is at the back

Example: Infix

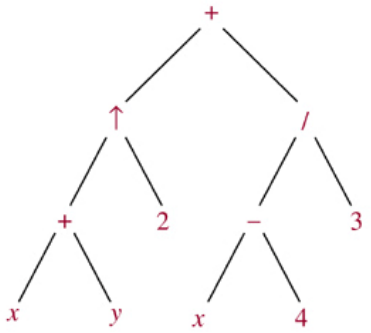


x + y ↑ 2 + x – 4 / 3

(((x + y) ↑ 2) + ((x – 4) / 3))

Prefix notation

- Operator is in front of the operands

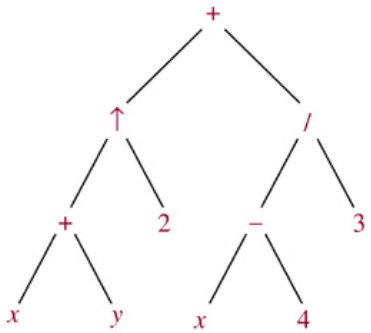


+ ↑ + x y 2 / - x 4 3

No parenthesis is needed!!!

Postfix notation

- Operator is in front of the operands



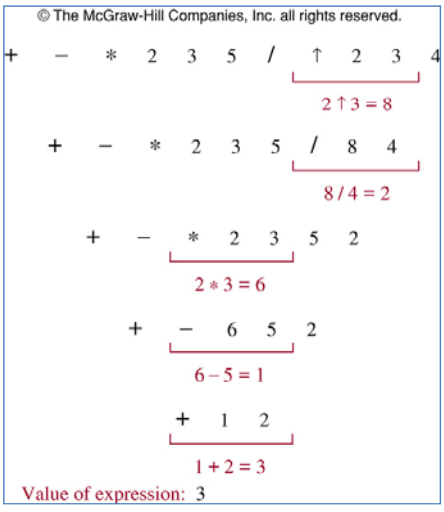
x y + 2 ↑ x 4 – 3 / +

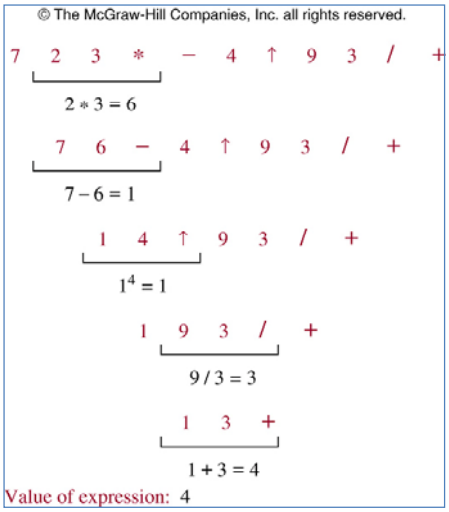
No parenthesis is needed!!!

Evaluating Prefix and Postfix

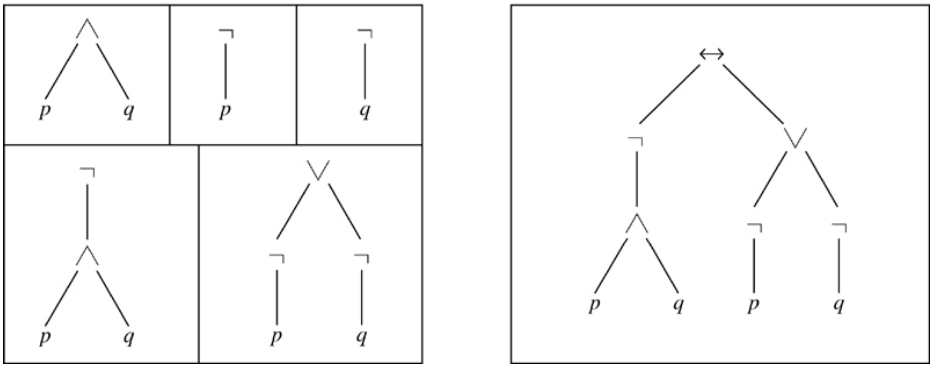
- Scan from left to right until we found something in this form
  - Let "a" be some operand
  - Let "*" be some operators
- Prefix: * a a
- Postfix: a a *
- Convert that

Evaluating Prefix and Postfix





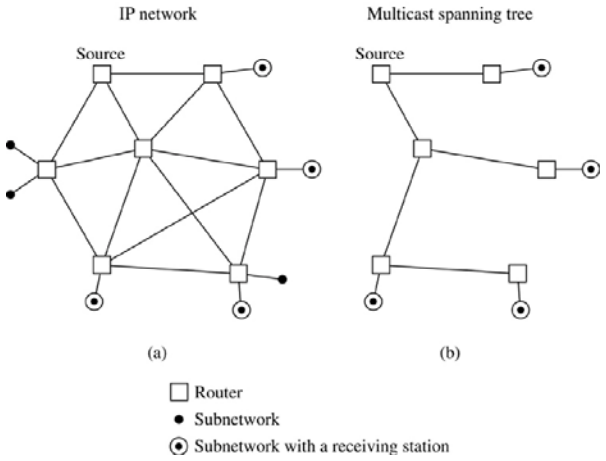
Expression Tree for Logic



Spanning Tree

Section 10.4

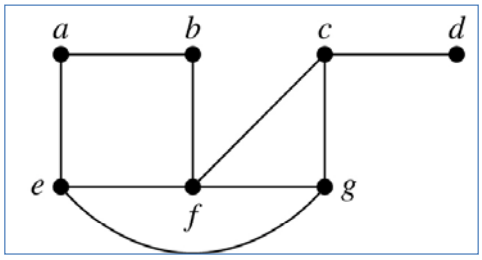
Spanning Tree



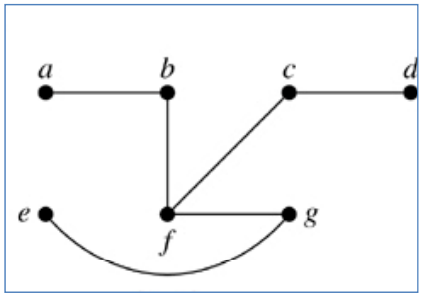
Spanning Tree Definition

- Let G be a simple graph. A spanning tree of G is a subgraph of G that is a tree containing every vertex of G

Example



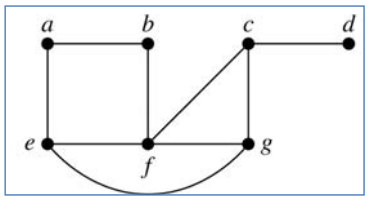
A simple graph G

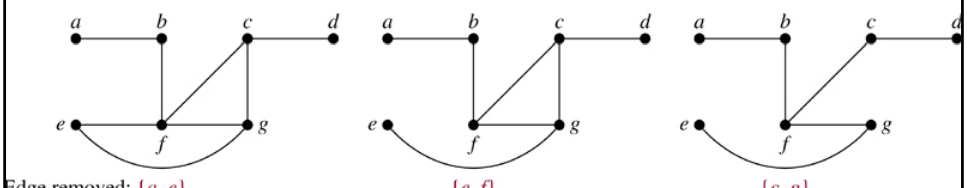


Spanning Tree of G

Constructing Spanning Tree

- By removing edges that form simple circuit





Edge removed: $\{a, e\}$ $\{e, f\}$ $\{c, g\}$

(a) (b) (c)

Theorem 1

- A simple graph is connected if and only if it contains spanning tree