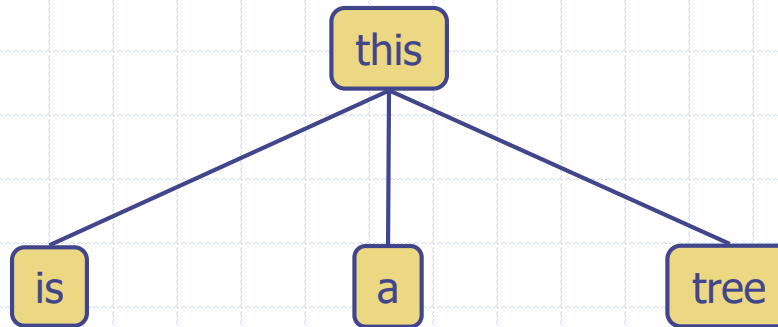


Trees



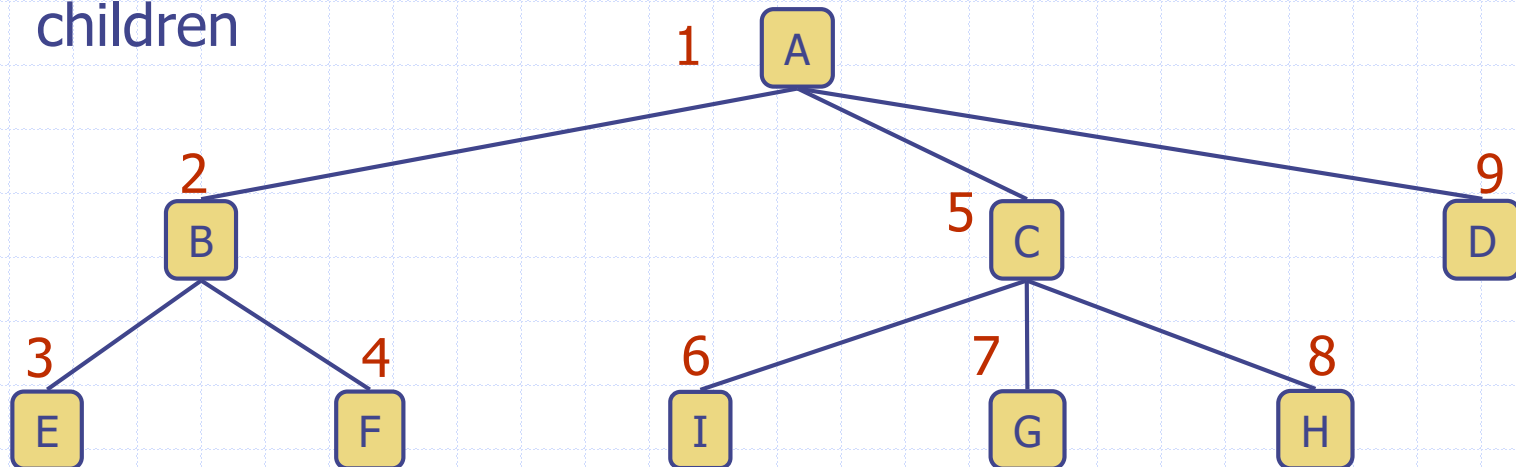
Traversal of Trees

- ◆ A traversal of a tree T is a systematic way of visiting all the nodes of T
- ◆ Traversing a tree involves visiting the root and traversing its subtrees
- ◆ There are the following traversal methods:
 - Preorder Traversal
 - Postorder Traversal
 - Inorder Traversal (of a binary tree)

Preorder Traversal

- ◆ In a preorder traversal, a node is visited before its descendants
- ◆ If a tree is ordered, then the subtrees are traversed according to the order of the children

```
Algorithm preOrder(v)  
    visit(v)  
    for each child w of v  
        preorder (w)
```

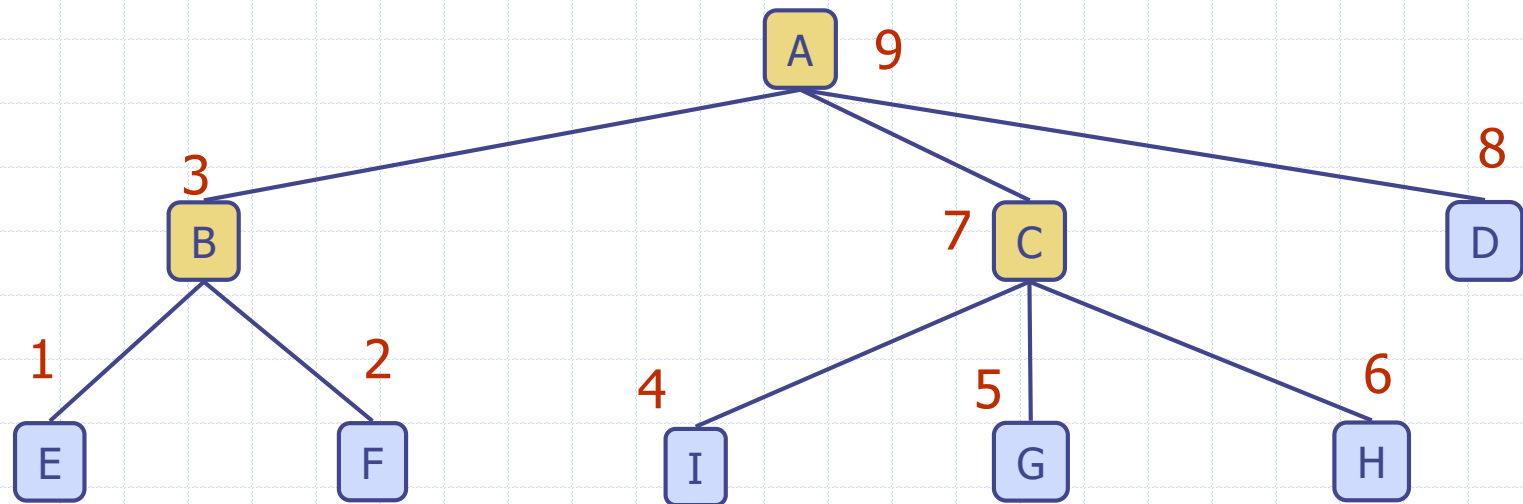


Preorder: ABEFCIGHD

Postorder Traversal

- ◆ In a postorder traversal, a node is visited after its descendants

Algorithm *postOrder*(v)
 for each child w of v
 postOrder (w)
 visit(v)



Postorder: *EFBIGHCDA*

Inorder Traversal

- ◆ In an inorder traversal a node is visited after its left subtree and before its right subtree

Algorithm *inOrder*(*v*)

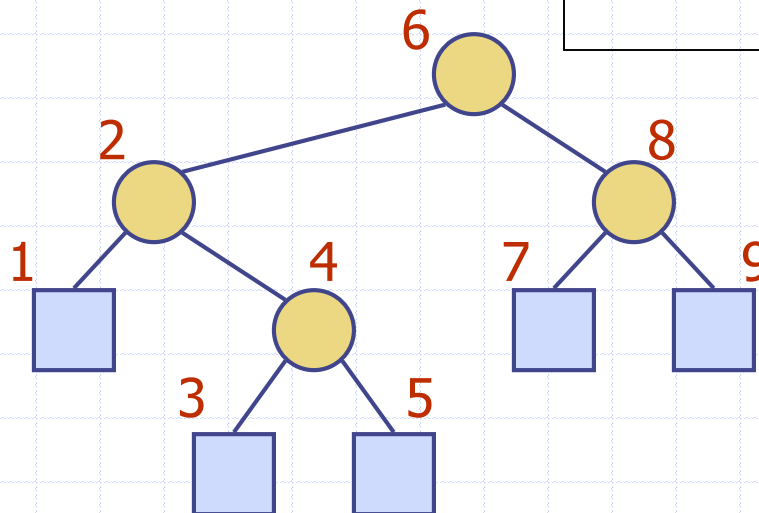
if *isInternal* (*v*)

inOrder (*leftChild* (*v*))

visit(*v*)

if *isInternal* (*v*)

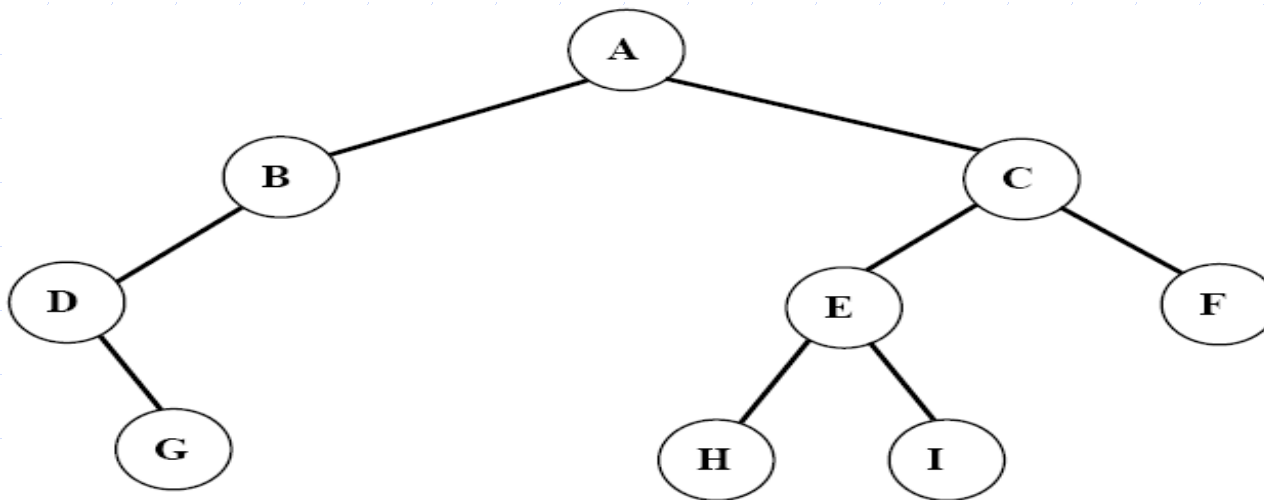
inOrder (*rightChild* (*v*))



Inorder Traversal

Traversing a binary tree in *inorder*

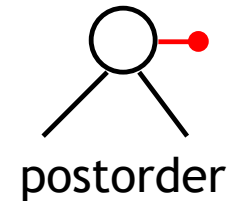
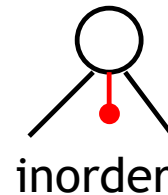
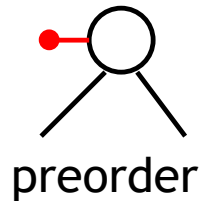
1. Traverse the ***left subtree*** in inorder.
2. Visit the ***root***.
3. Traverse the ***right subtree*** in inorder.



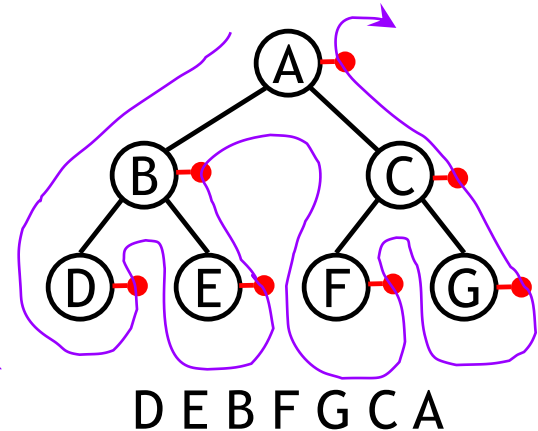
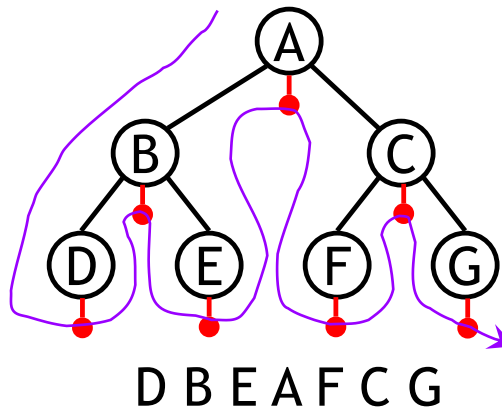
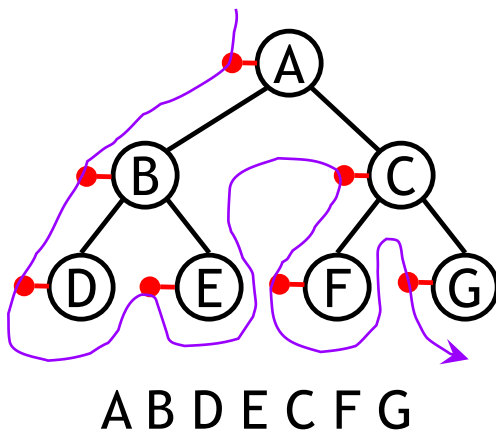
Inorder: DGBAHEICF

Tree traversals using “flags”

- The order in which the nodes are visited during a tree traversal can be easily determined by imagining there is a “flag” attached to each node, as follows:

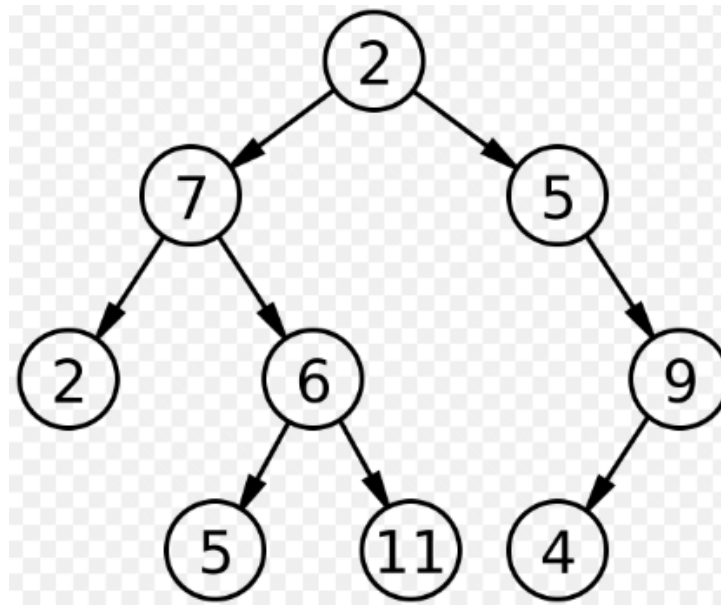


- To traverse the tree, collect the flags:



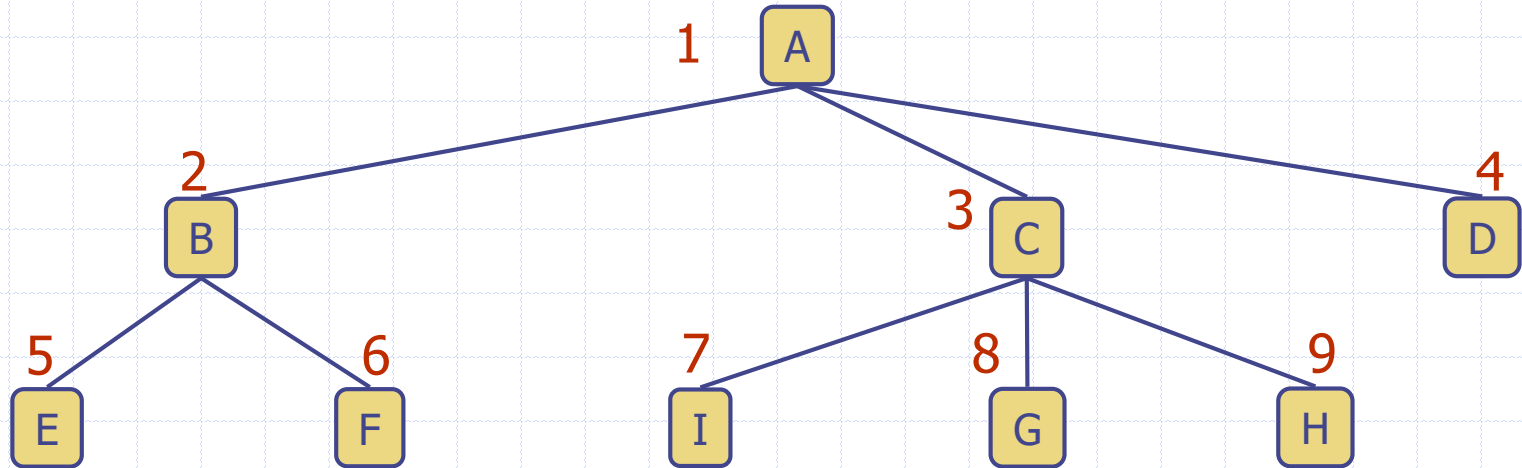
Exercise

- Perform pre-order, post-order and in-order types of traversal on the following tree



Level Order Traversal

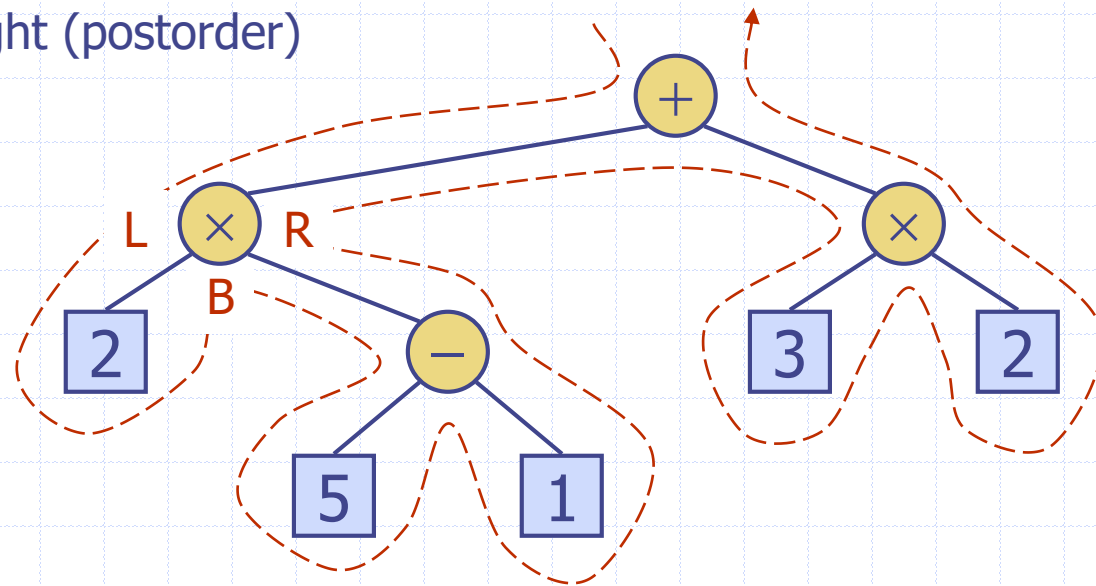
- ◆ In a level order traversal, every node on a level is visited before going to a lower level



Level order: *ABCDEFGH*

Euler Tour Traversal

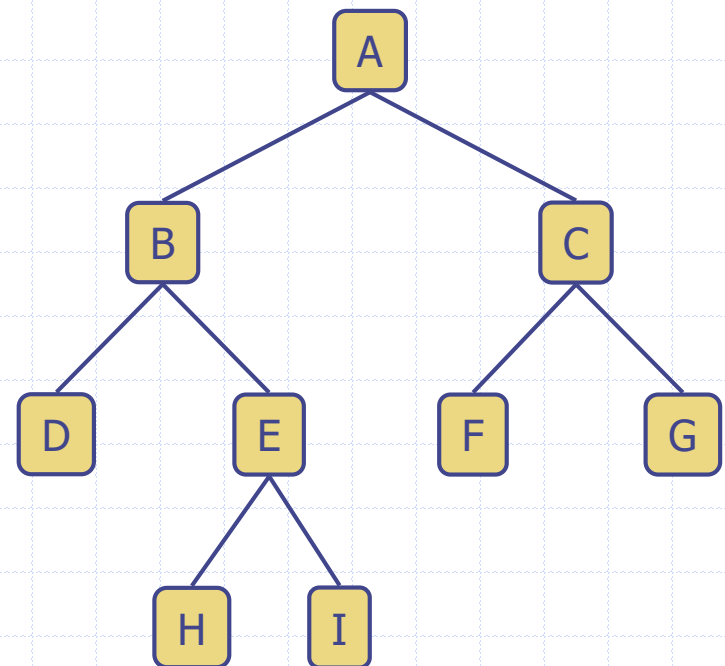
- ◆ Generic traversal of a binary tree
- ◆ Includes a special cases the preorder, postorder and inorder traversals
- ◆ Walk around the tree and visit each node three times:
 - on the left (preorder)
 - from below (inorder)
 - on the right (postorder)



(Proper) Binary Tree

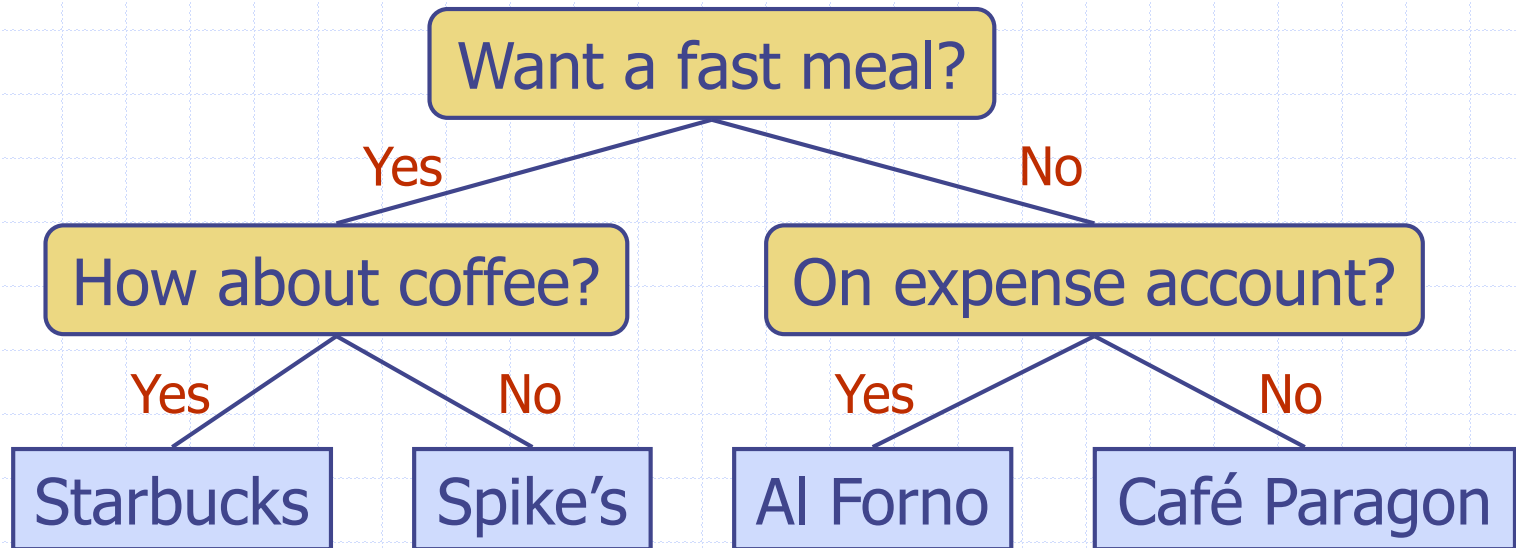
- ◆ A (proper) binary tree is a tree with the following properties:
 - Each internal node has two children
 - The children of a node are an ordered pair
- ◆ We call the children of an internal node left child and right child
- ◆ Alternative recursive definition: a (proper) binary tree is either
 - a tree consisting of a single node, or
 - a tree whose root has an ordered pair of children, each of which is a binary tree

- ◆ Applications:
 - arithmetic expressions
 - decision processes
 - searching



Decision Tree

- ◆ Binary tree associated with a decision process
 - internal nodes: questions with yes/no answer
 - external nodes: decisions
- ◆ Example: dining decision



Arithmetic Expression Tree

- ◆ Binary tree associated with an arithmetic expression
 - internal nodes: operators
 - external nodes: operands
- ◆ Example: arithmetic expression tree for the expression $(2 \times (a - 1) + (3 \times b))$

