Fakultas Informatika
School of Computing
Telkom University

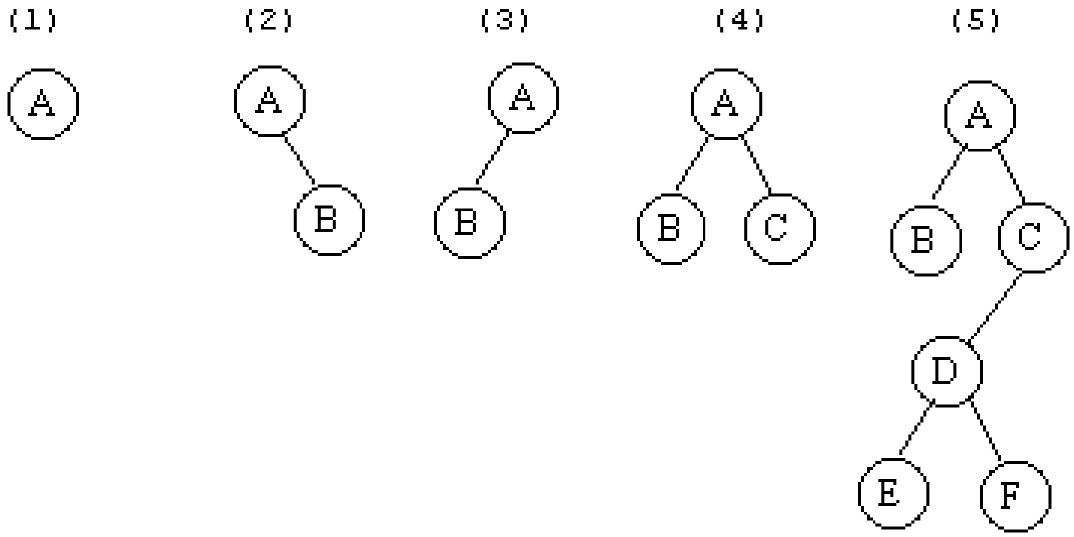
CSG2A3
ALGORITMA dan STRUKTUR DATA

Tree Data Structure

Binary Tree Data Structure

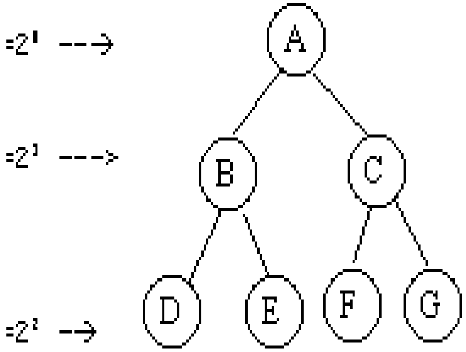
Binary Tree

- Tree Data Structure with maximum child (degree) of 2, which are referred to as the *left* child and the *right* child.



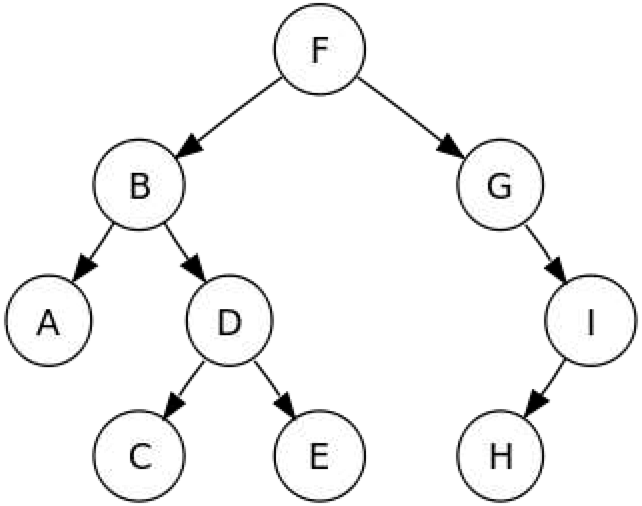
Properties of Binary Tree

- The number of nodes n a full binary tree is
  - At least : $n = 2(h+1) - 1$
  - At most : $n = 2^{h+1} - 1$
  - Where h is the height of the tree
- Maximum Node for each level = 2^n



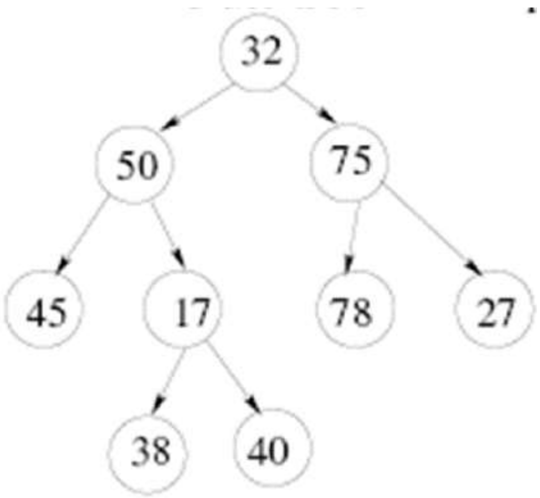
Types of Binary Tree

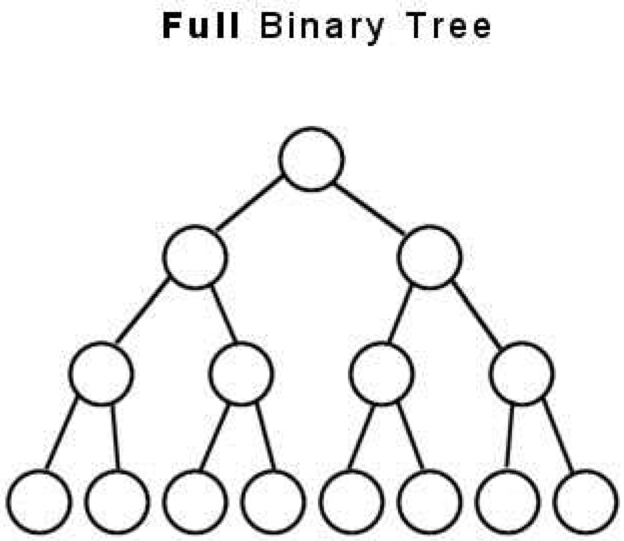
- Full Binary Tree
- Complete Binary Tree
- Skewed Binary Tree



Full Binary Tree

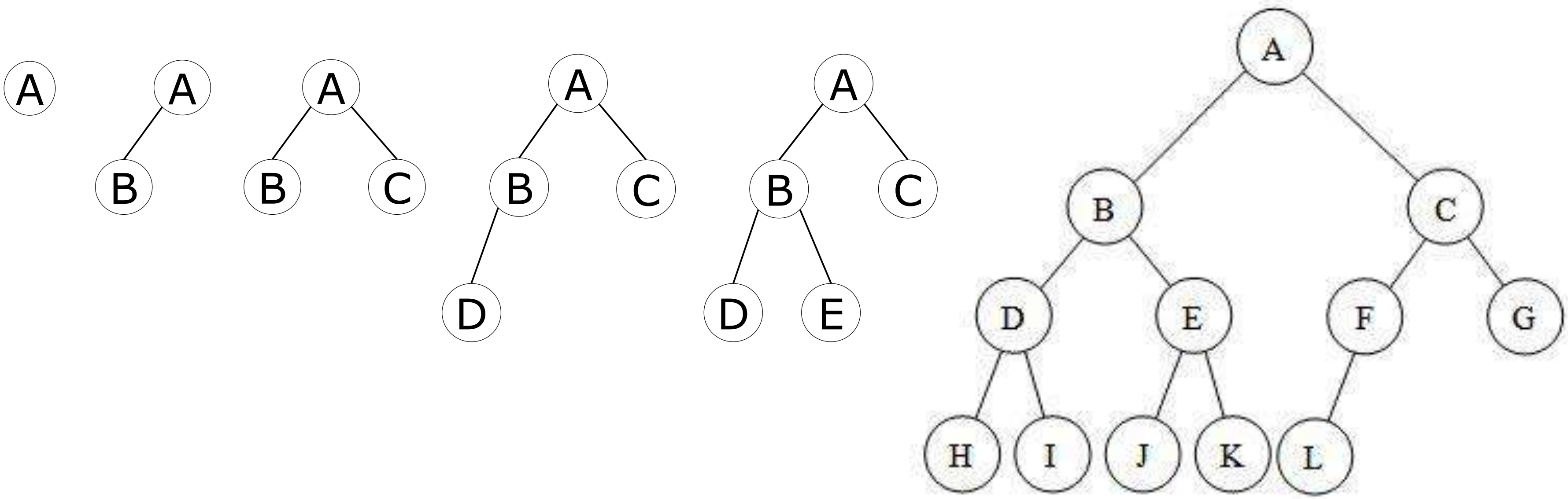
- A tree in which every node other than the leaves has two children
  - sometimes called proper binary tree or 2-tree





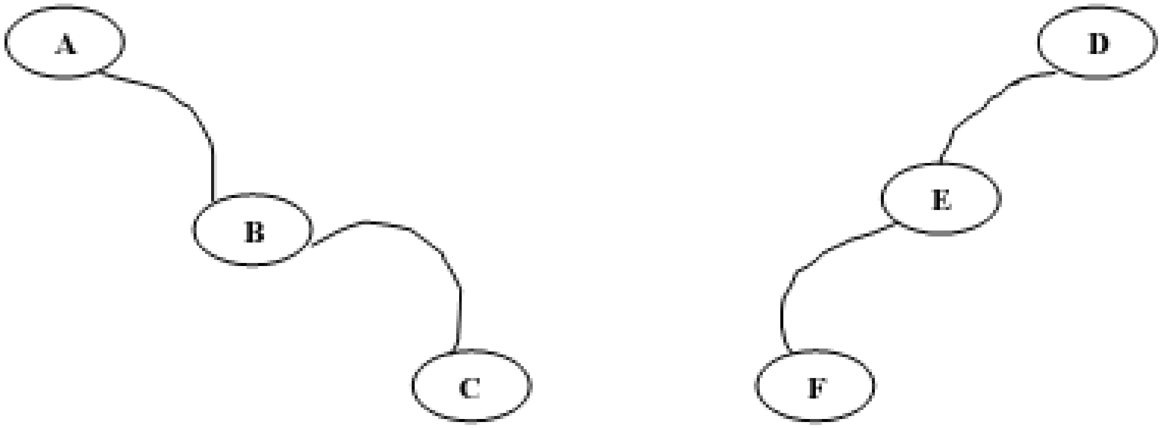
Complete Binary Tree

- a binary tree in which every level, except possibly the last, is completely filled, and all nodes are as far left as possible



Skewed Tree

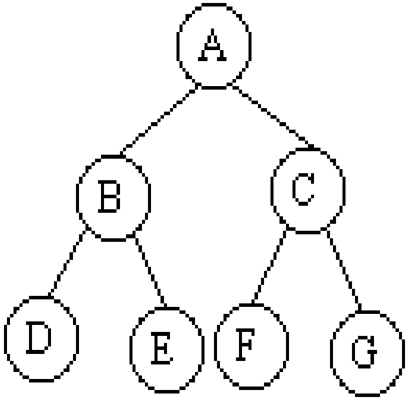
- Binary Tree with an unbalanced branch between left and right branch

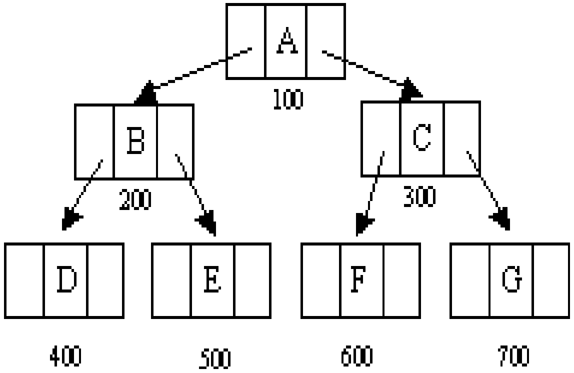


ADT Binary Tree

- Array Representation
- Linked list representation

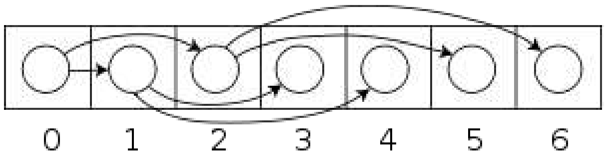
id	value
1	A
2	B
3	C
4	D
5	E
6	F
7	G

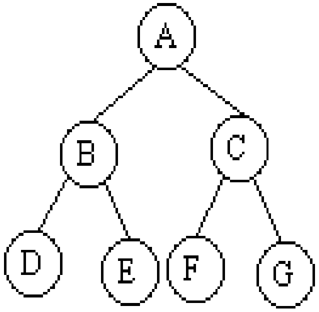




Array Representation

- if a node has an index *i*, its children are found at indices :
  - Left child : *2i + 1*
  - Right child : *2i + 2*
- while its parent (if any) is found at index $\left\lfloor \frac{i-1}{2} \right\rfloor$
  - (assuming the root has index zero)



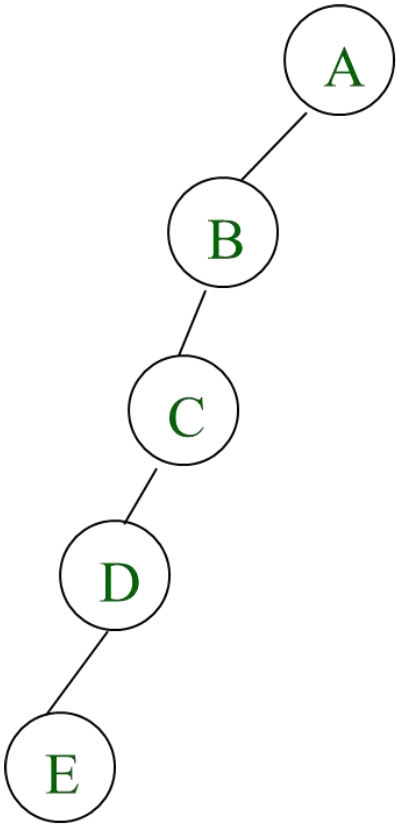


Array Representation

- Problem :

1	2	3	4	5	6	7	8	...	16
A	B	-	C	-	-	-	D	...	E

  - Waste space
  - Insertion / deletion problem



- Array Representation is good for Complete Binary Tree types
  - Binary Heap Tree

Linked List Representation

```
Type infotype : integer
Type address : pointer to Node

Type Node <
     info  : infotype
     left : address
     right : address
>
```

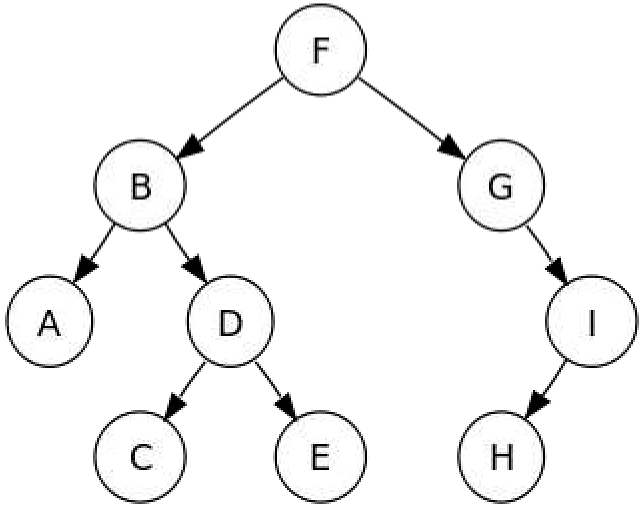
left	Info	right

```
Type BinTree :  address

Dictionary
     root : BinTree
```

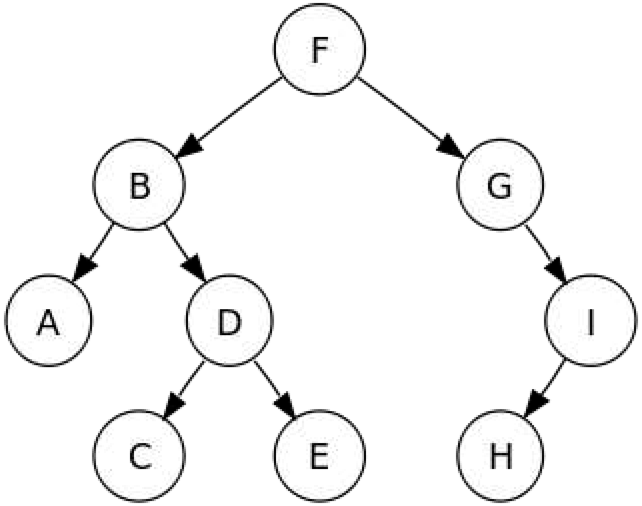
Traversal on Binary Tree

- DFS traversal
  - Pre-order
  - In-order
  - Post-order
- BFS traversal
  - Level-order



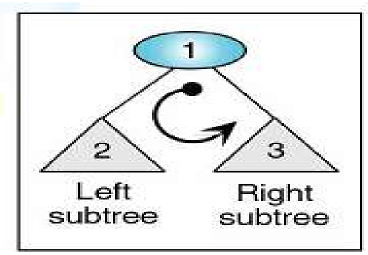
Pre-order Traversal

- Deep First Search
- Root → Left → Right
  - Prefix notation
- Result :
  - FBADCEGIH

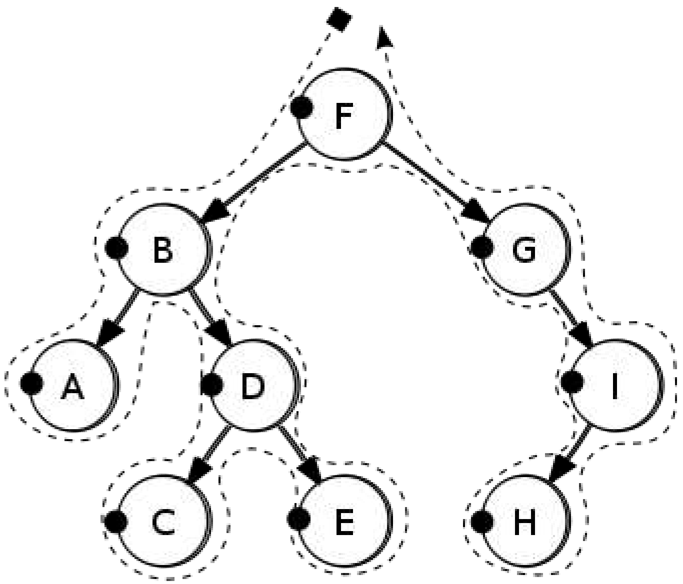


Pre-order Traversal

```
Procedure preOrder( i/o root : tree )
Algorithm
    if ( root != null ) then
        output( info( root ) )
        preOrder( left( root ) )
        preOrder( right( root ) )
```

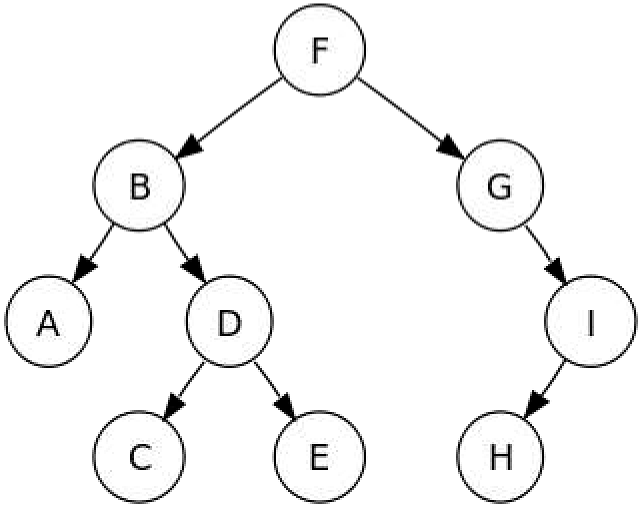


(a) Preorder traversal



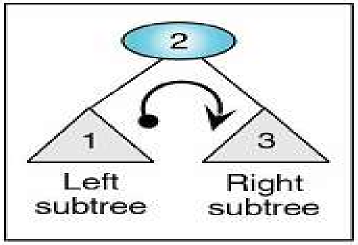
In-order Traversal

- Left → Root → Right
  - Infix notation
- Result :
  - ABCDEFGHI

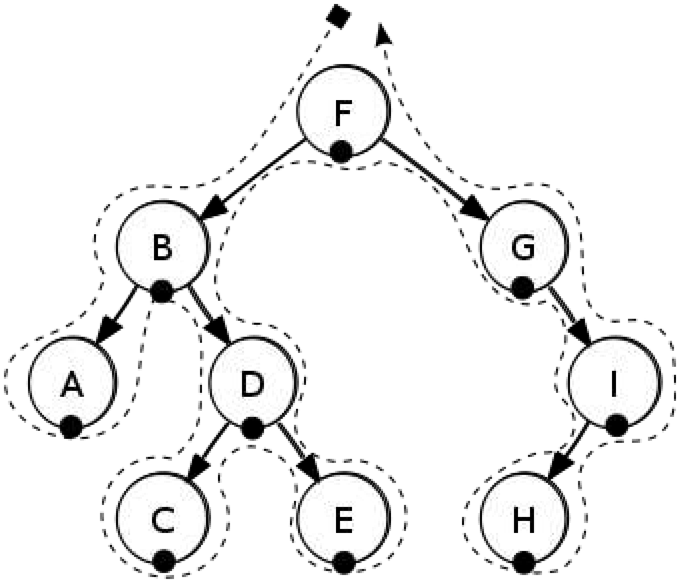


In-order Traversal

```
Procedure inOrder( i/o root : tree )
Algorithm
    if ( root != null ) then
        inOrder( left( root ) )
        output( info( root ) )
        inOrder( right( root ) )
```

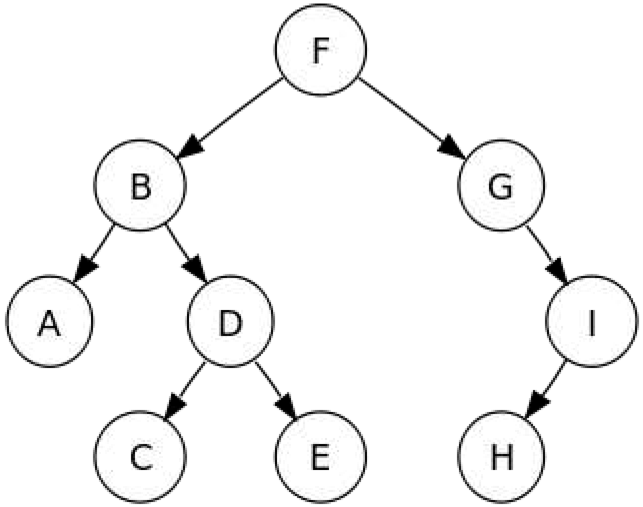


(b) Inorder traversal



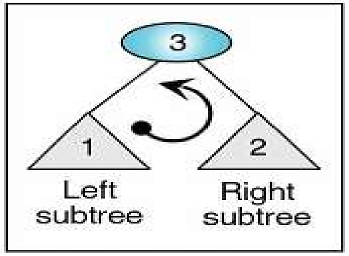
Post-order Traversal

- Left → right → Root
  - Postfix notation
- Result :
  - ACEDBHIGF

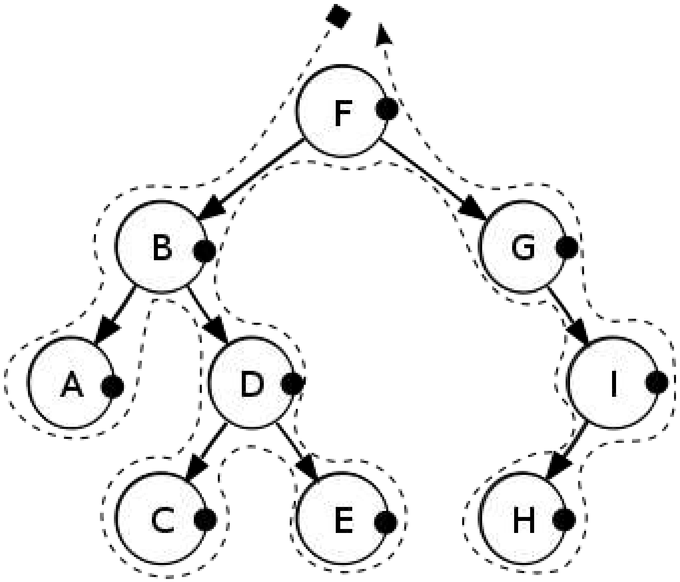


Post-order Traversal

```
Procedure postOrder( i/o root : tree )
Algorithm
    if ( root != null ) then
        postOrder( left( root ) )
        postOrder( right( root ) )
        output( info( root ) )
```

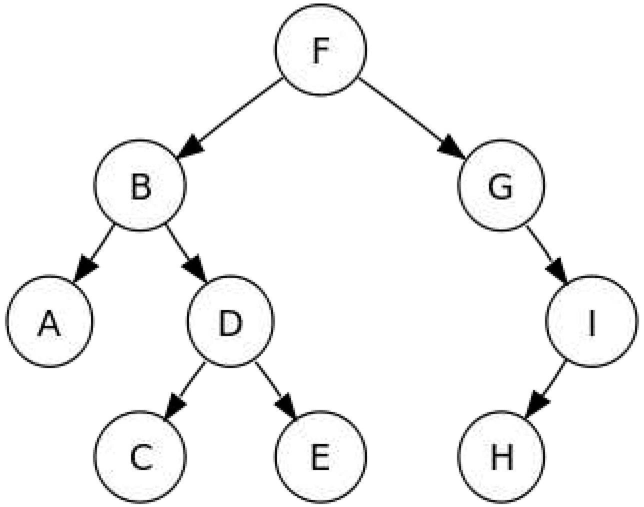


(c) Postorder traversal



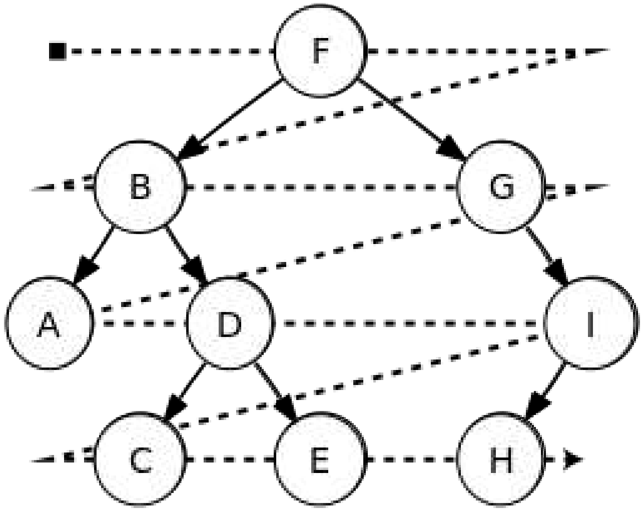
Level-order Traversal

- Breadth First Search
- recursively at each node
- Result :
  - FBGADICEH

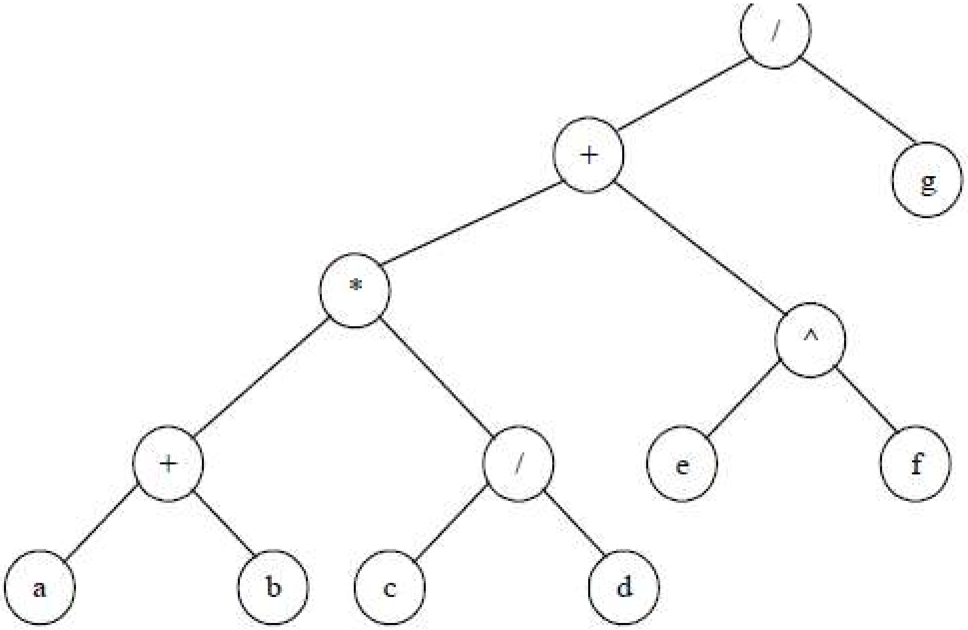


Level-order Traversal

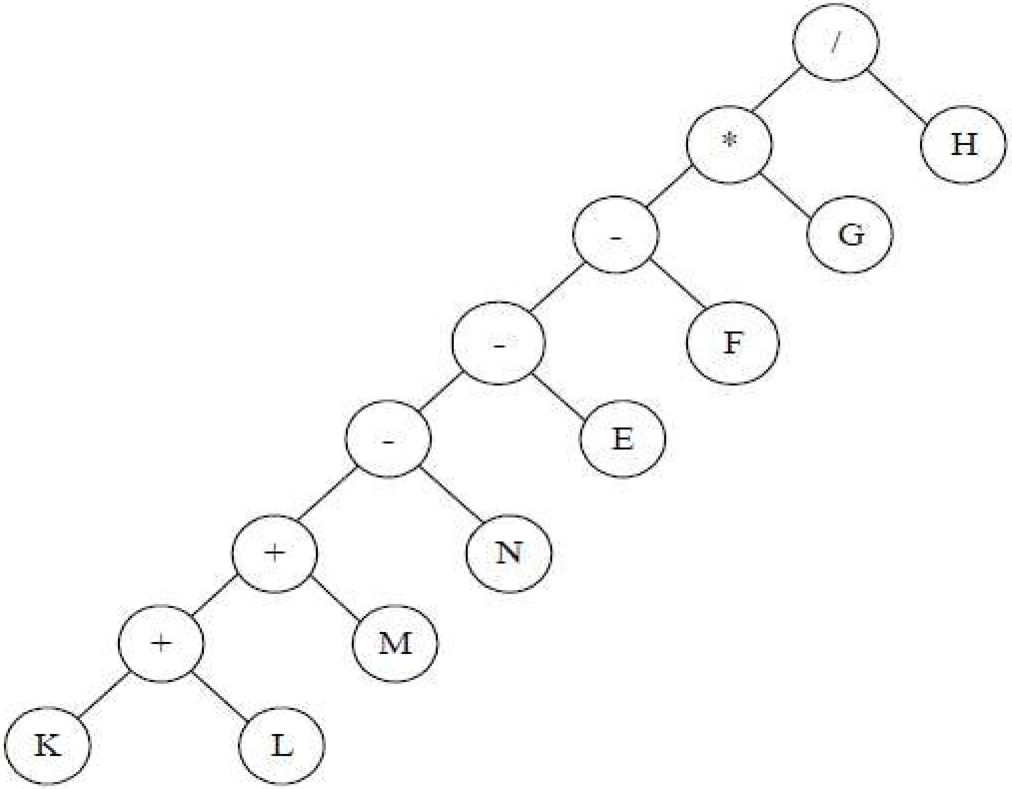
```
Procedure levelOrder( root : tree )
Dictionary
    Q : Queue
Algorithm
    enqueue( Q, root )
    while ( not isEmpty(Q) )
        n ← dequeue( Q )
        output( n )
        enqueue( child ( n ) )
```



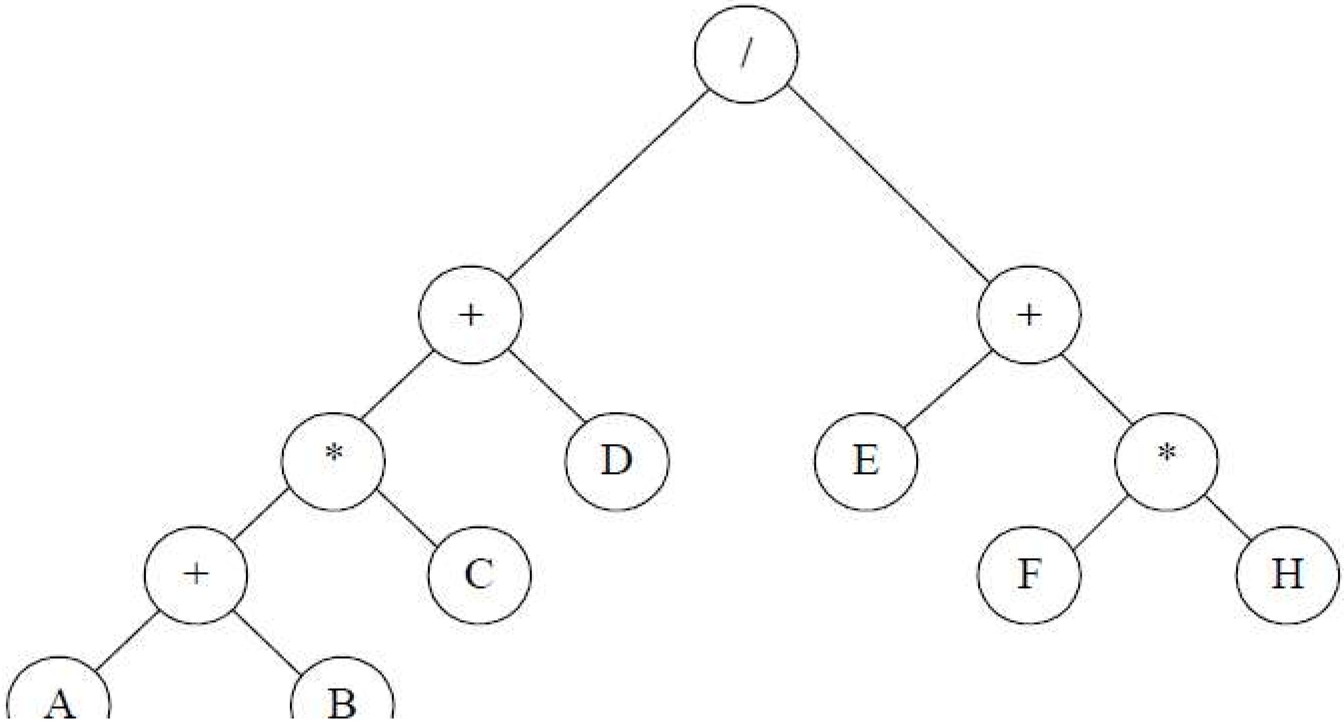
Exercise – write the traversal - 1



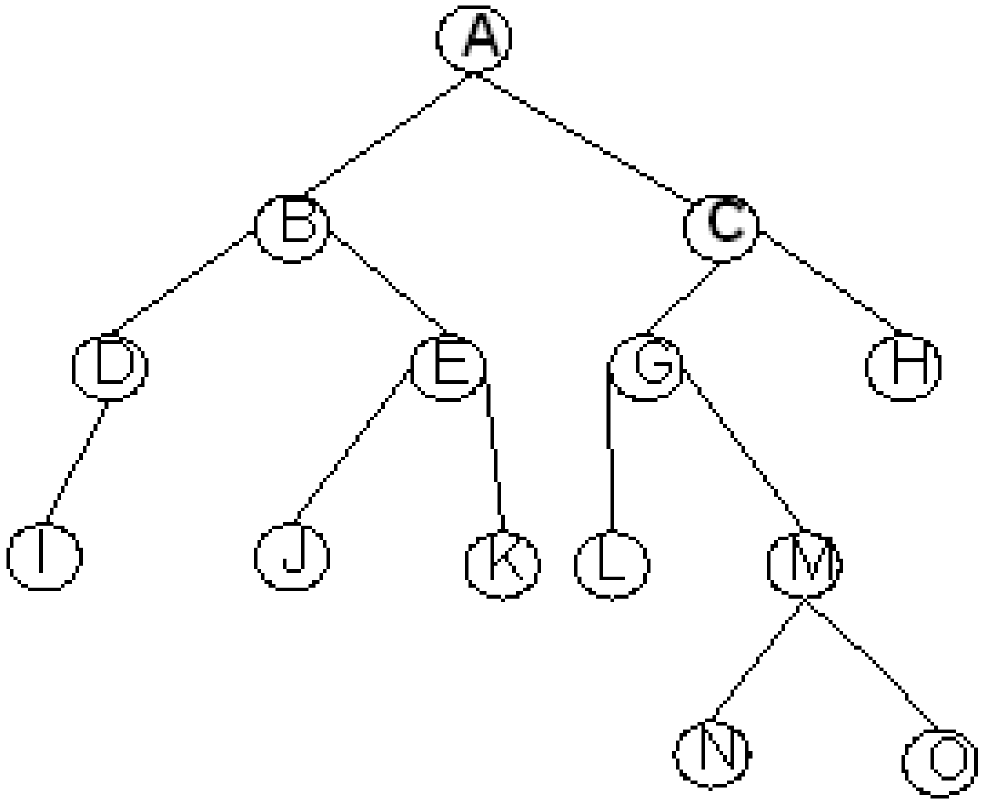
Exercise – write the traversal - 2



Exercise – write the traversal - 3



Exercise – write the traversal - 4

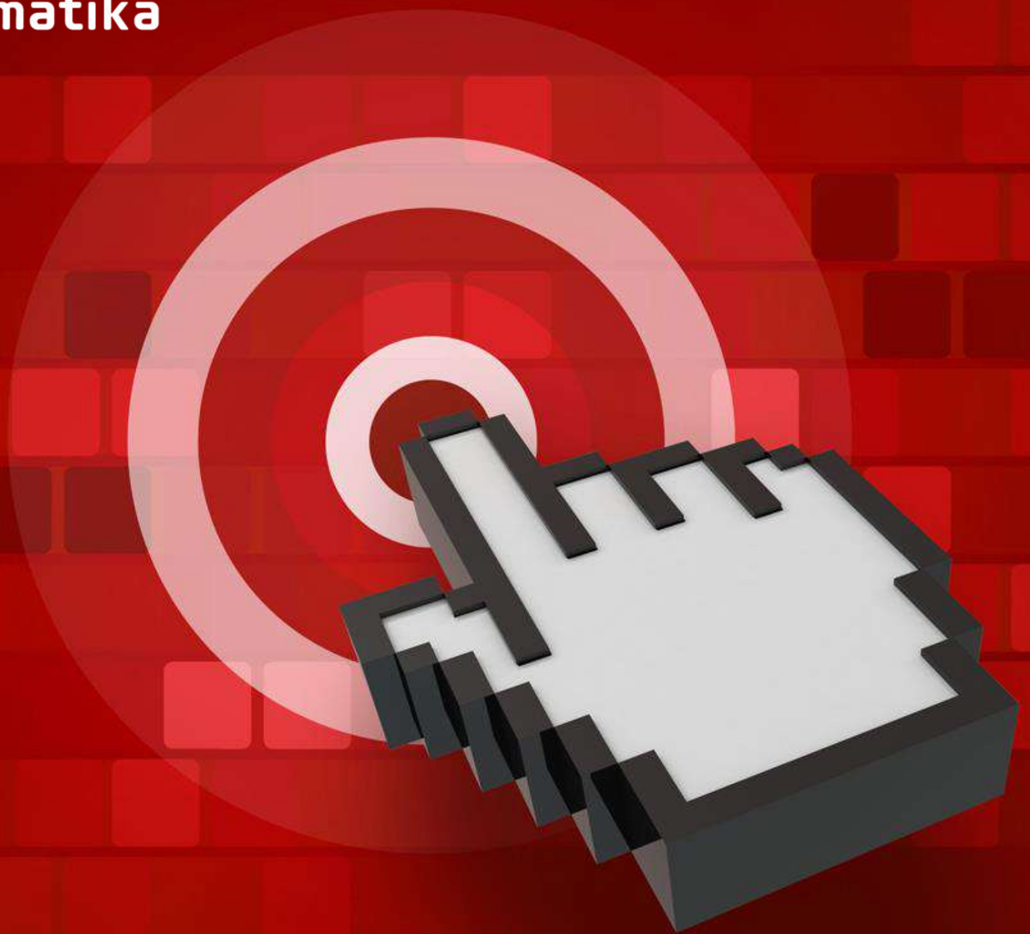


Exercise – Create the Tree

- Assume there is ONE tree, which if traversed by Inorder resulting : EACKFHDBG, and when traversed by Preorder resulting : FAEKCDHGB
- Draw the tree that satisfy the condition above

Question?

Fakultas Informatika
School of Computing
Telkom University



THANK YOU