[Data Structure] Summary6

Queue

**Queue Application:**

**1.Buffer**

**-** Queues can work as buffers that coordinate an interaction between two processes

- read & write can not be done at the same time -> using buffer lock()!

🔐lock(): function to avoid producer and consumer access the buffer simultaneously.

When buffer locked, others can not access to that buffer.

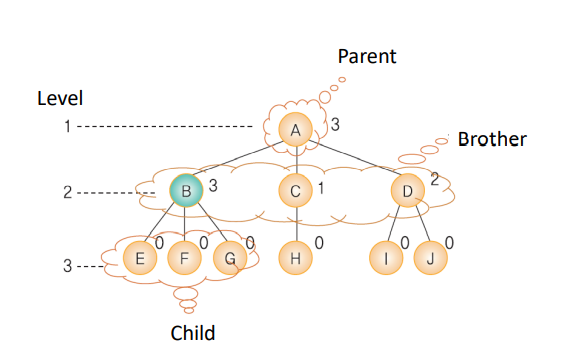
**2. Simulation**

**-** used to simulate and analyze system characteristics when FIFO.

-

Tree

Data structure representing a hierarchical structure



**Terminology**

• Node: Tree component

• Root: Node without parents

• Subtree: consists of one node and its descendants

• Terminal node(=leaf node): node without children (E, F, G, H, I, J)

• Non-terminal node: node with at least one child (A, B, C, D)

**Binary Tree**

**:** A tree that has at most two child node.

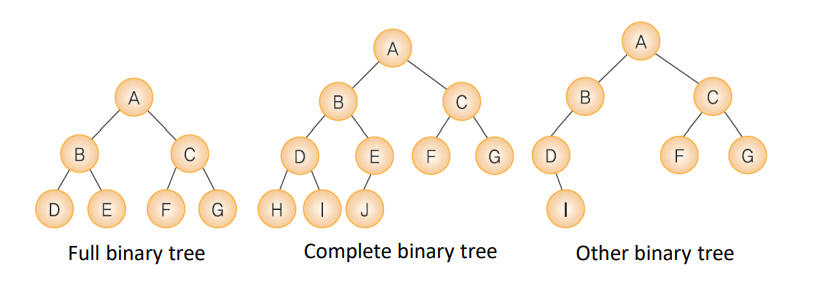
- left should be filled with before right

- The subtrees of the binary tree is also binary trees.

If the number of nodes is n, number of links is n-1

For a binary tree of height, h<=number of nodes<=-1 (=1+2+….)

For the binary tree with n nodes, <=height of binary tree<=n (n<=-1, n+1<=-1, log(n+1)<=h )

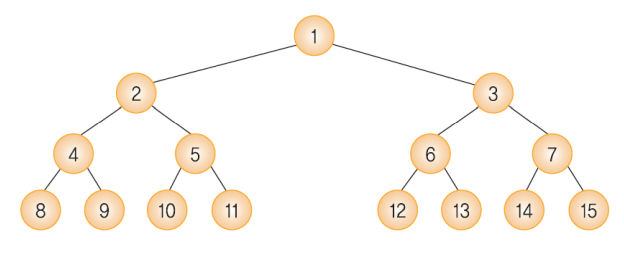
**Type**

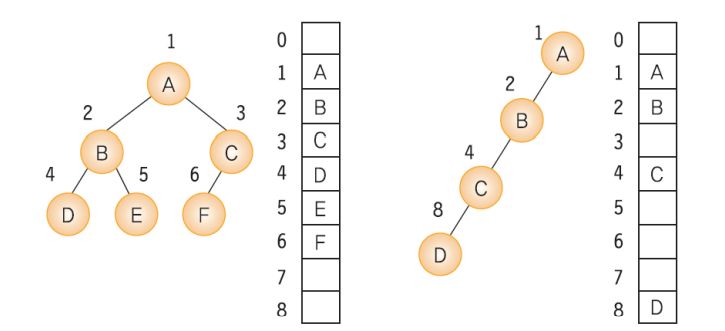
• Full binary tree : has full node

• Complete binary tree : full tree except for last level

• Other binary tree : unbalanced

**counting**



**Binary Tree using Array**

– Parent node of node i: i/2

– Left child node of node : 2i

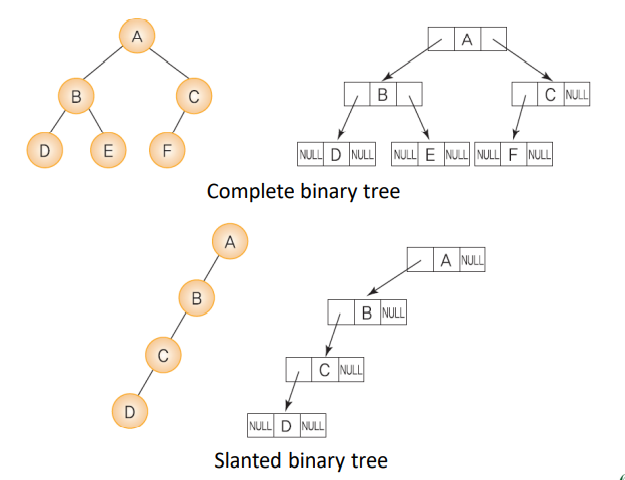
– Right child node of node :2i+1

+: easy to implement

-: wastes memory spaces, except for full or complete binary trees (sucha as second example)

**Binary Tree using Linked List**

:using doubly linked list, no waste of memory!

****

**Traversal of Binary Tree**

Preorder traversal: V -> L -> R ex) Output of structured documents

Inorder traversal: L -> V -> R ex) Formula tree

Postorder traversal: L -> R -> V ex) Calculation of directory size ex)

**Level Traversal**

: visiting each node in order of level using queue!

