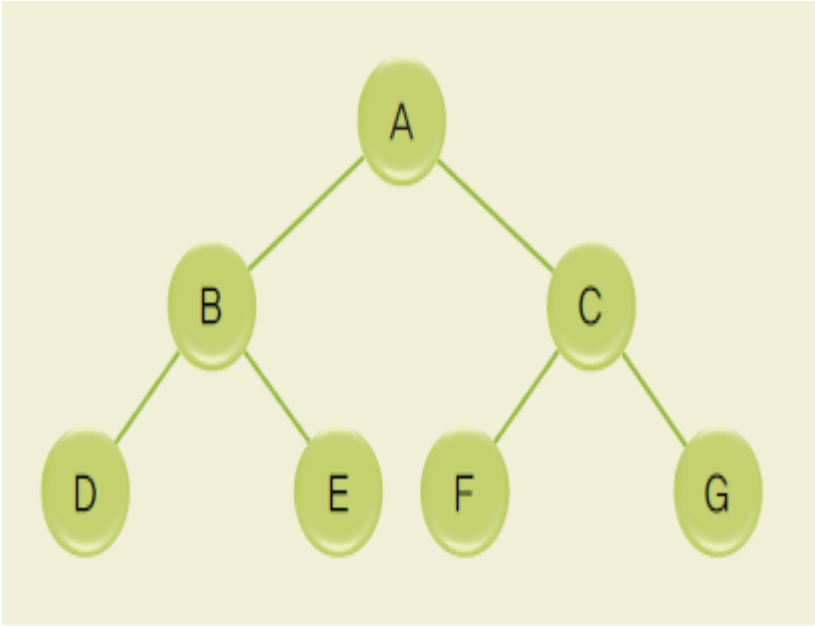
tree?

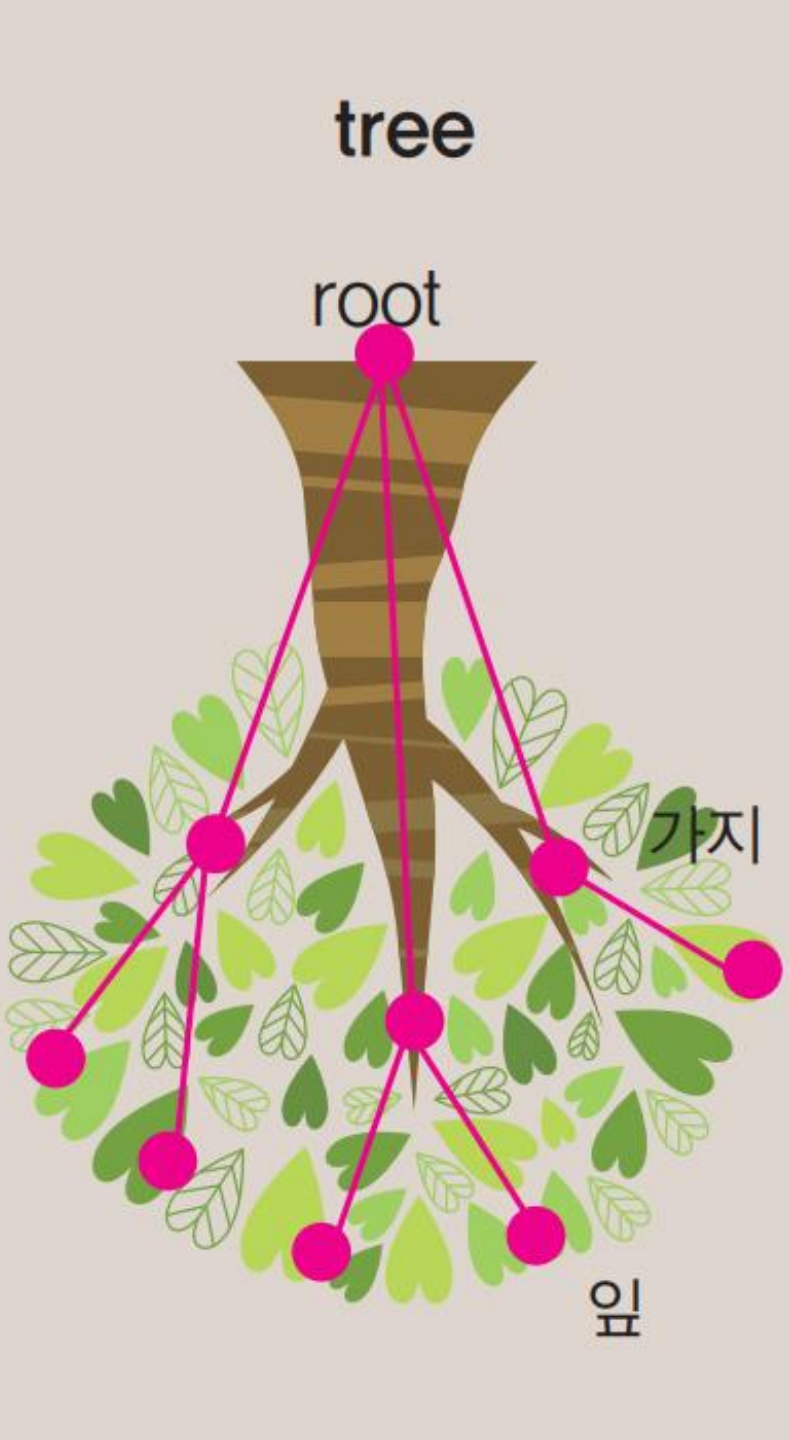
root

트리

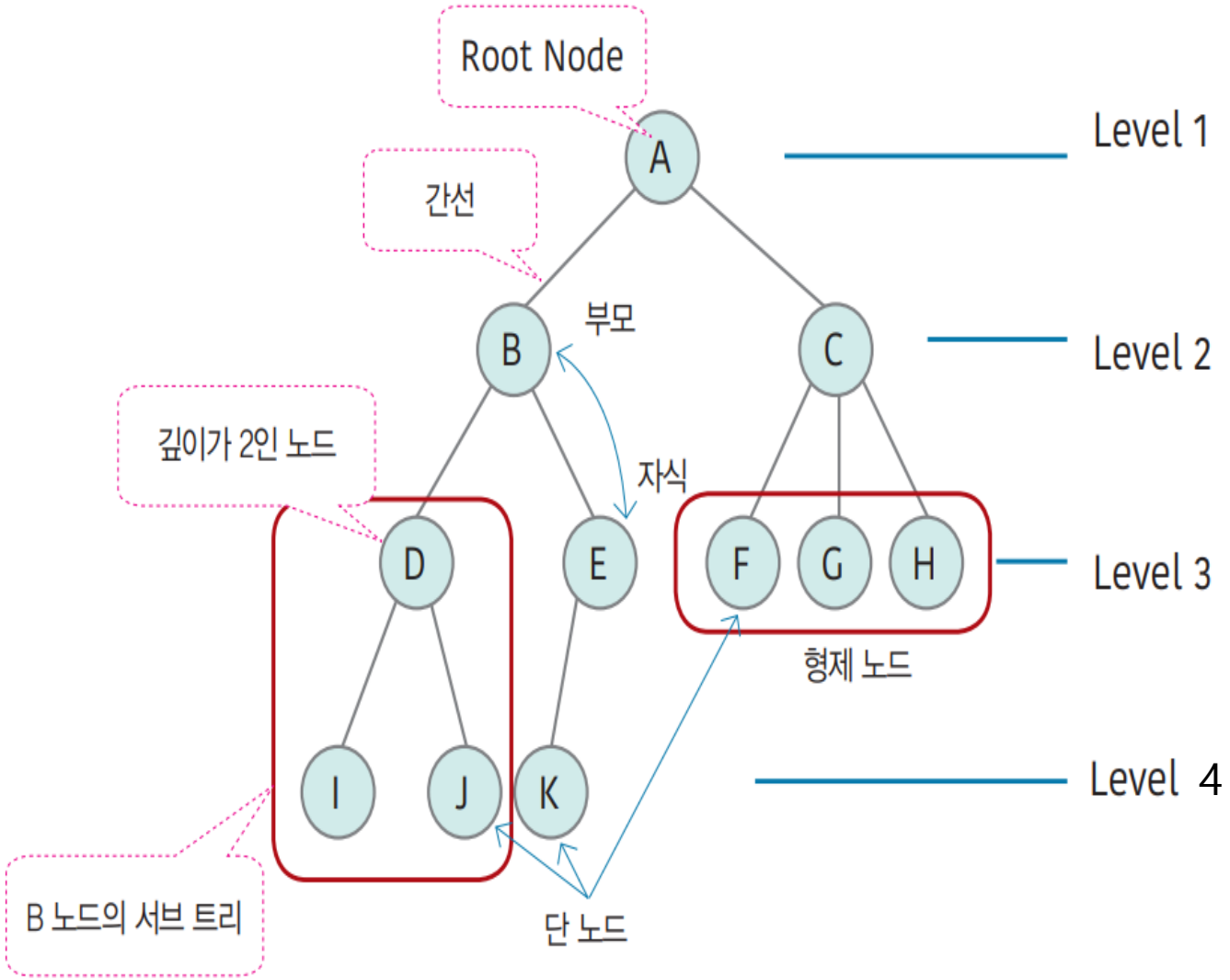
◘ 트리 개념

- 각각의 자료들을 **계층적**으로 서로 연결
- 주로 자료들 간의 포함 관계나 상 · 하위 관계를 표현할 때 사용.



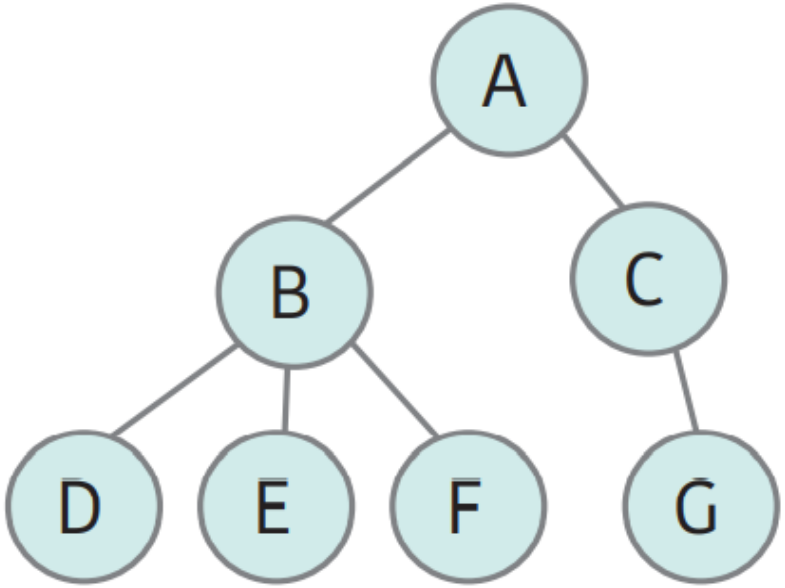


◘ 트리 용어



- 노드(node)
- 간선(edge)
- 근노드(root node)
- 단노드(leaf node)
- 부모노드(parent node)
- 자식노드(child node)
- 서브 트리(sub tree)
- 형제노드(sibling node)
- 차수(degree)
- 깊이(depth)
- 레벨(level)
- 높이(height)

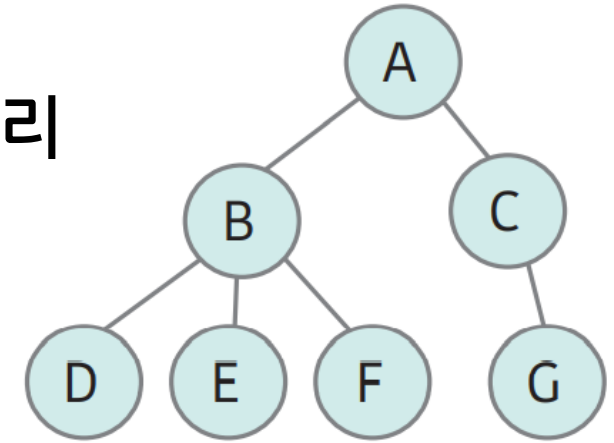
QUIZ



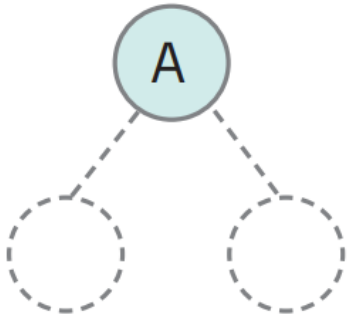
- 노드를 연결하는 선은?
- ( )노드는 부모가 없는 노드로 가장 위에 있는 노드를 말한다.
- 왼쪽 트리의 높이는 ?
- B의 차수는 ?
- C의 형제 노드는 ?
- B의 자식 노드는 ?
- E의 부모 노드는 ?
- 2레벨에 있는 노드는?

□ 이진 트리

vs 일반 트리

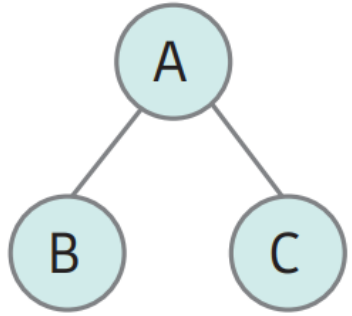


루트를 포함한 모든 노드가 **최대 2개**의 자식 노드를 가질 수 있는 자료 구조이다.



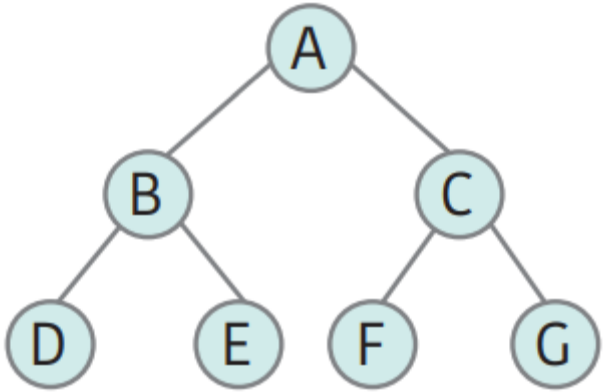
(a) 차수가 0인 경우

(b) 차수가 1인 경우

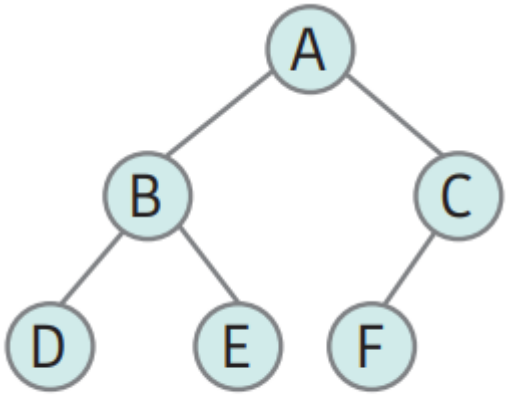


(c) 차수가 2인 경우

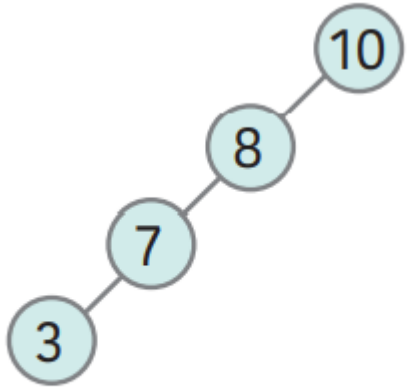
□ 이진 트리의 종류



(a) 포화 이진트리

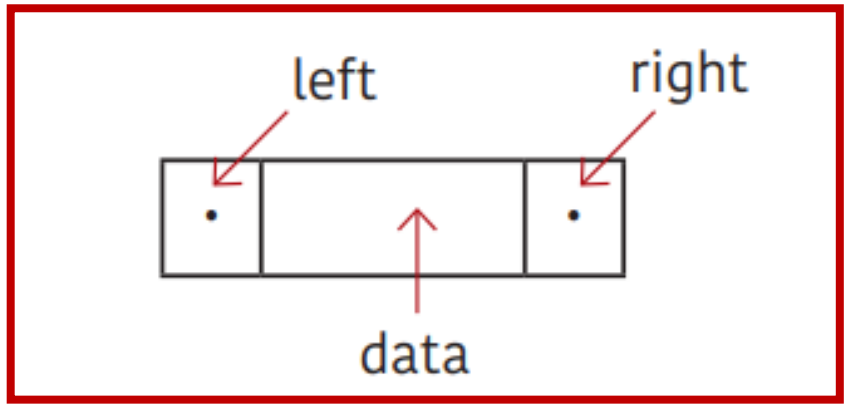


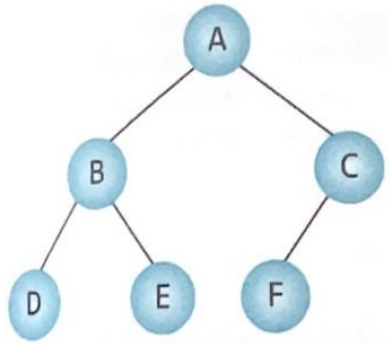
(b) 완전 이진트리

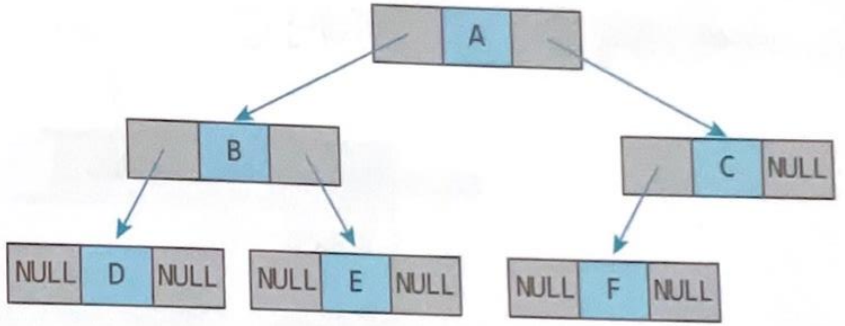


(c) 편향 이진트리

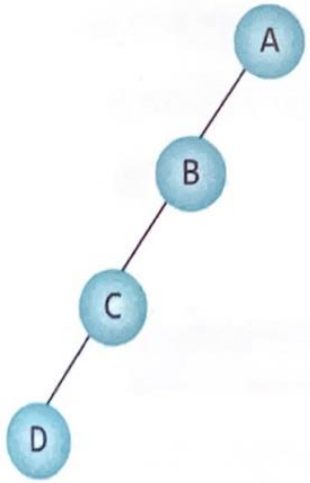
□ 이진 트리의 구현

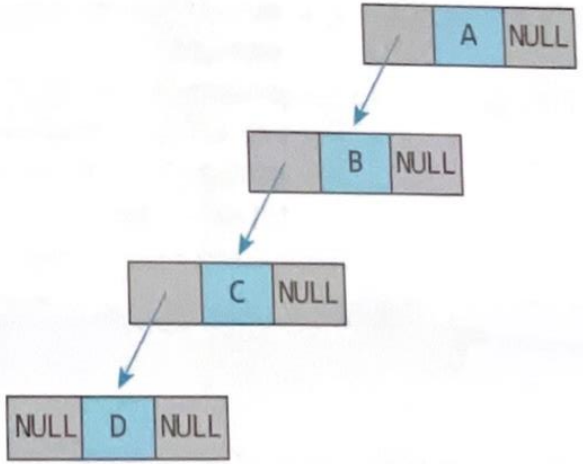






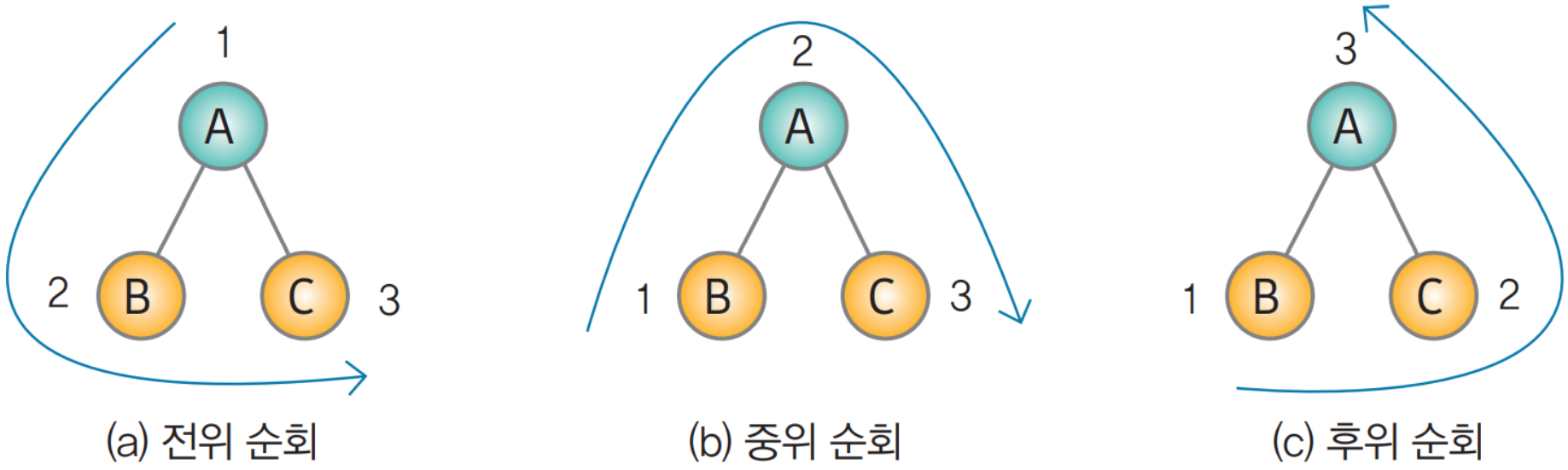
이중연결리스트를 이용하여 구현
- 왼쪽 링크 필드는
왼쪽 자식 노드를 가리키는 포인터 변수
- 오른쪽 링크 필드는
오른쪽 자식 노드를 가리키는 포인터 변수
- 데이터 필드에는 데이터 값

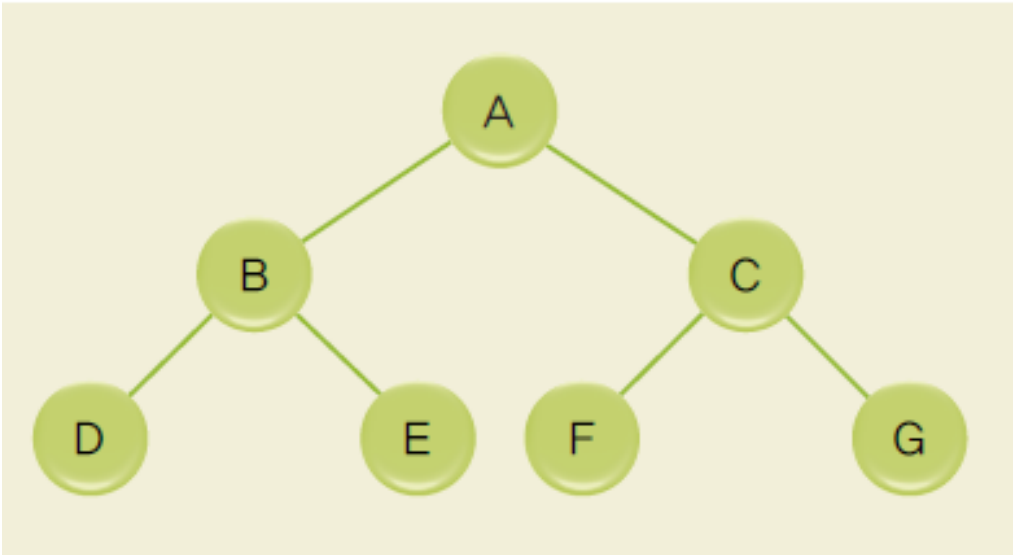




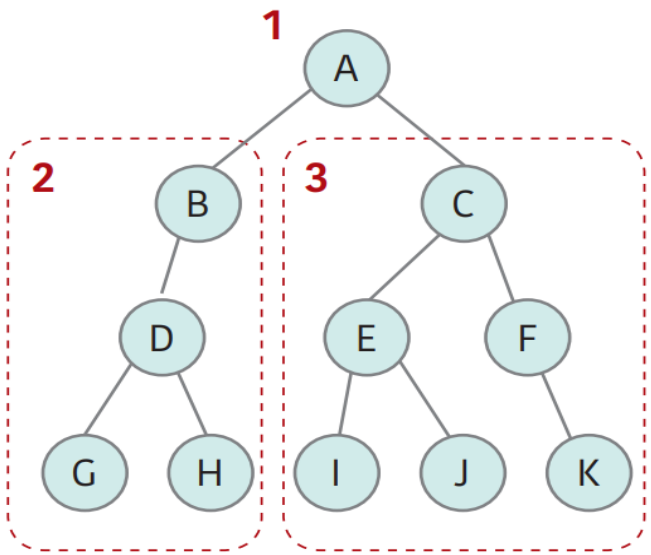
□ 이진 트리의 순회

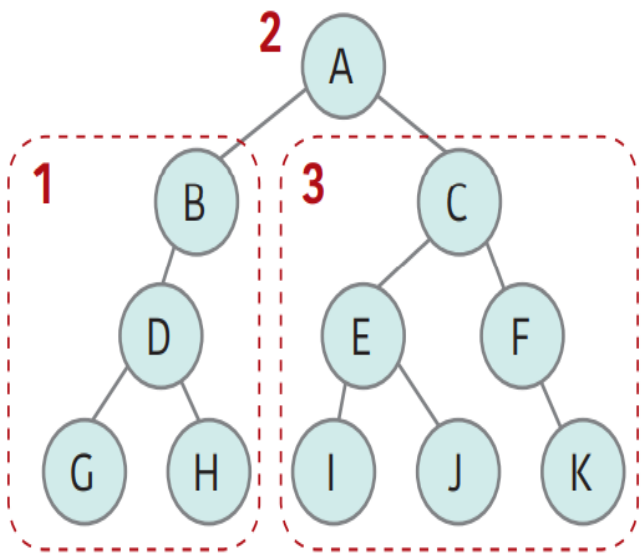
- 순회 : 모든 노드를 정해진 순서에 따라 한번씩 방문하는 것

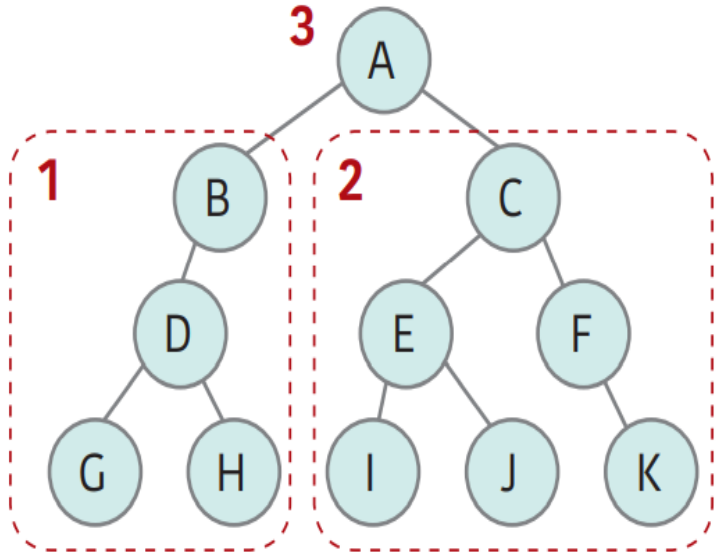




전위순회 순서 : A → B → D → E → C → F → G
중위순회 순서 : D → B → E → A → F → C → G
후위순회 순서 : D → E → B → F → G → C → A







A → B → D → G → H → C → E →I →J →F →K

G → D → H → B → A → I → E → J → C →F →K

G → H → D → B → I → J → E → K → F → C → A

실습 : 이진 트리의 순회 구현하기

재귀 함수를 사용하여 이진 트리를 순회하는 함수를 구현해보자.

- void PreorderTraverse(BTreeNode* bt)
- void InorderTraverse(BTreeNode* bt)
- void PostorderTraverse(BTreeNode* bt)

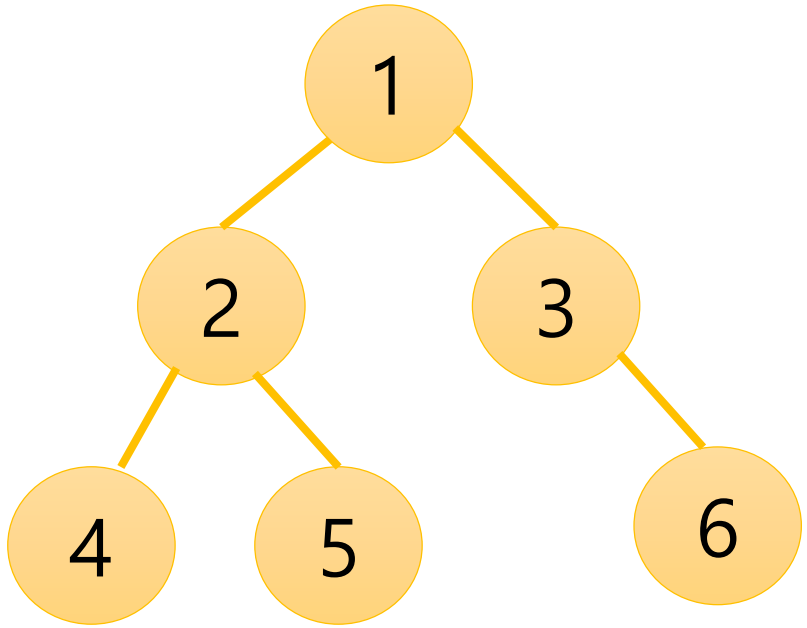
```
// 종료 조건

// 방문한 노드의 데이터를 출력

// 다음 노드 방문
```

실습 : 이진 트리의 순회 구현하기

```
SetData(bt1, 1);
SetData(bt2, 2);
SetData(bt3, 3);
SetData(bt4, 4);
SetData(bt5, 5);
SetData(bt6, 6);
MakeLeftSubTree(bt1, bt2);
MakeRightSubTree(bt1, bt3);
MakeLeftSubTree(bt2, bt4);
MakeRightSubTree(bt2, bt5);
MakeRightSubTree(bt3, bt6);
```




```
preorder  : 124536
inorder   : 425136
postorder : 452631
```