

COSE213: Data Structure

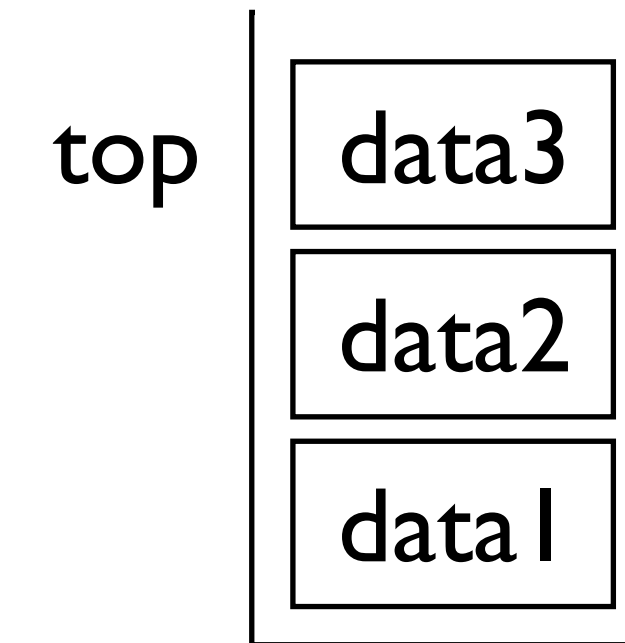
Lecture 7 - 트리 (Tree)

Minseok Jeon

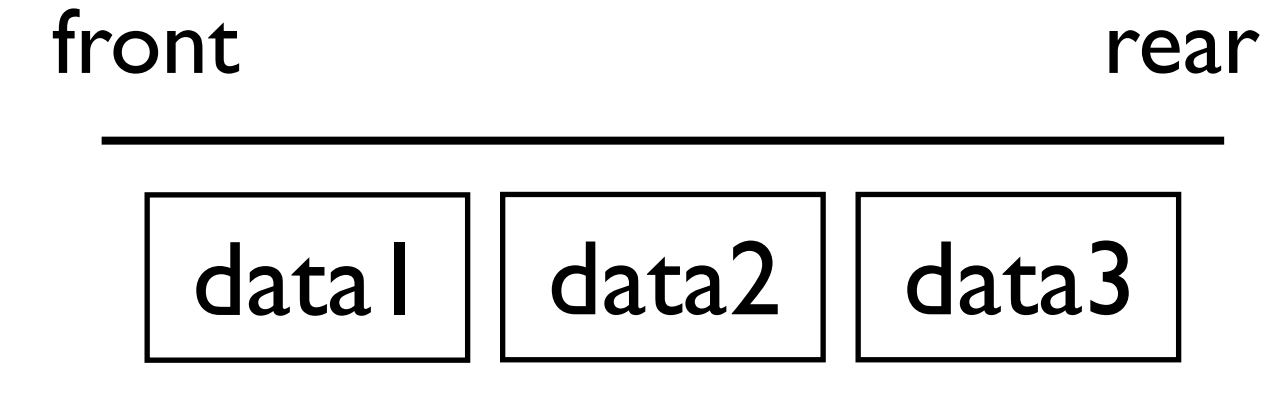
2024 Fall

리뷰: 선형 (linear) 자료구조

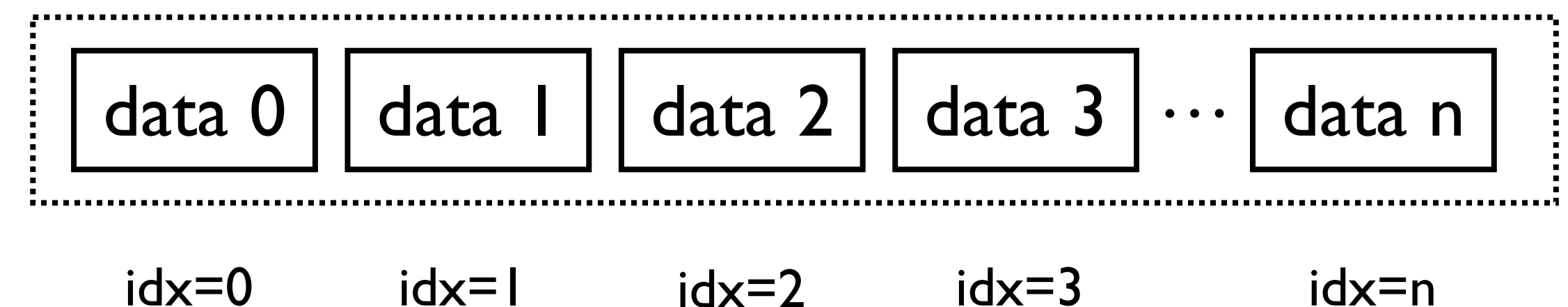
- 스택(Stack) 자료구조: 후입선출(LIFO) 원칙을 따르는 자료구조



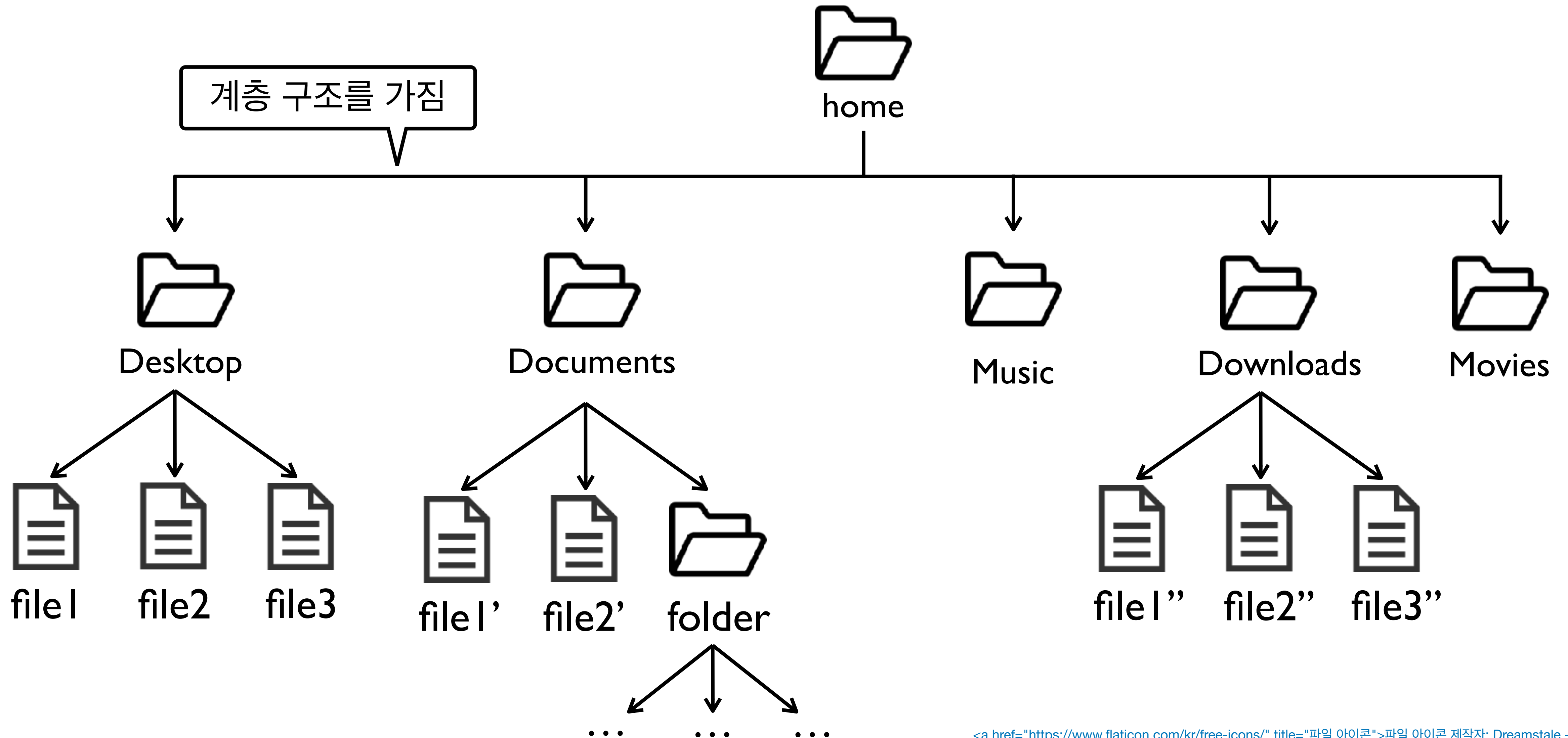
- 큐(Queue) 자료구조: 선입선출 원칙을 따르는 자료구조



- 리스트(list) 자료구조: 데이터들이 순차적으로 나열되어 있는 선형 자료구조



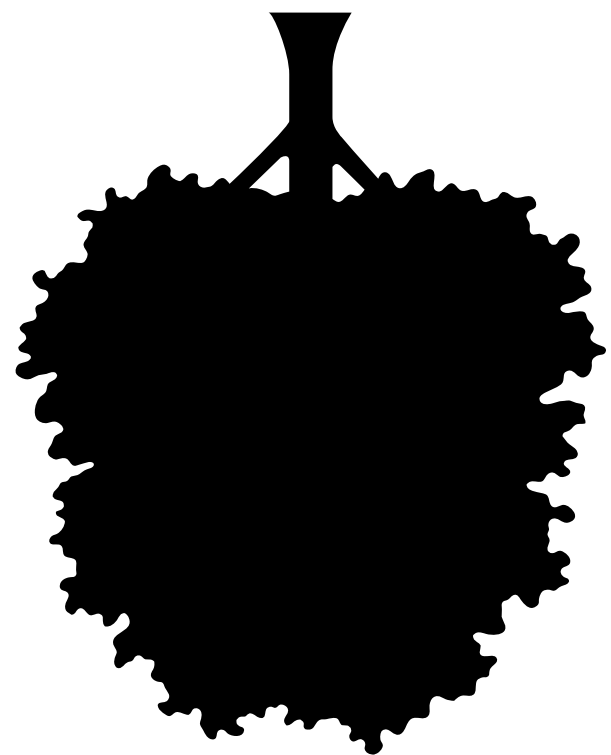
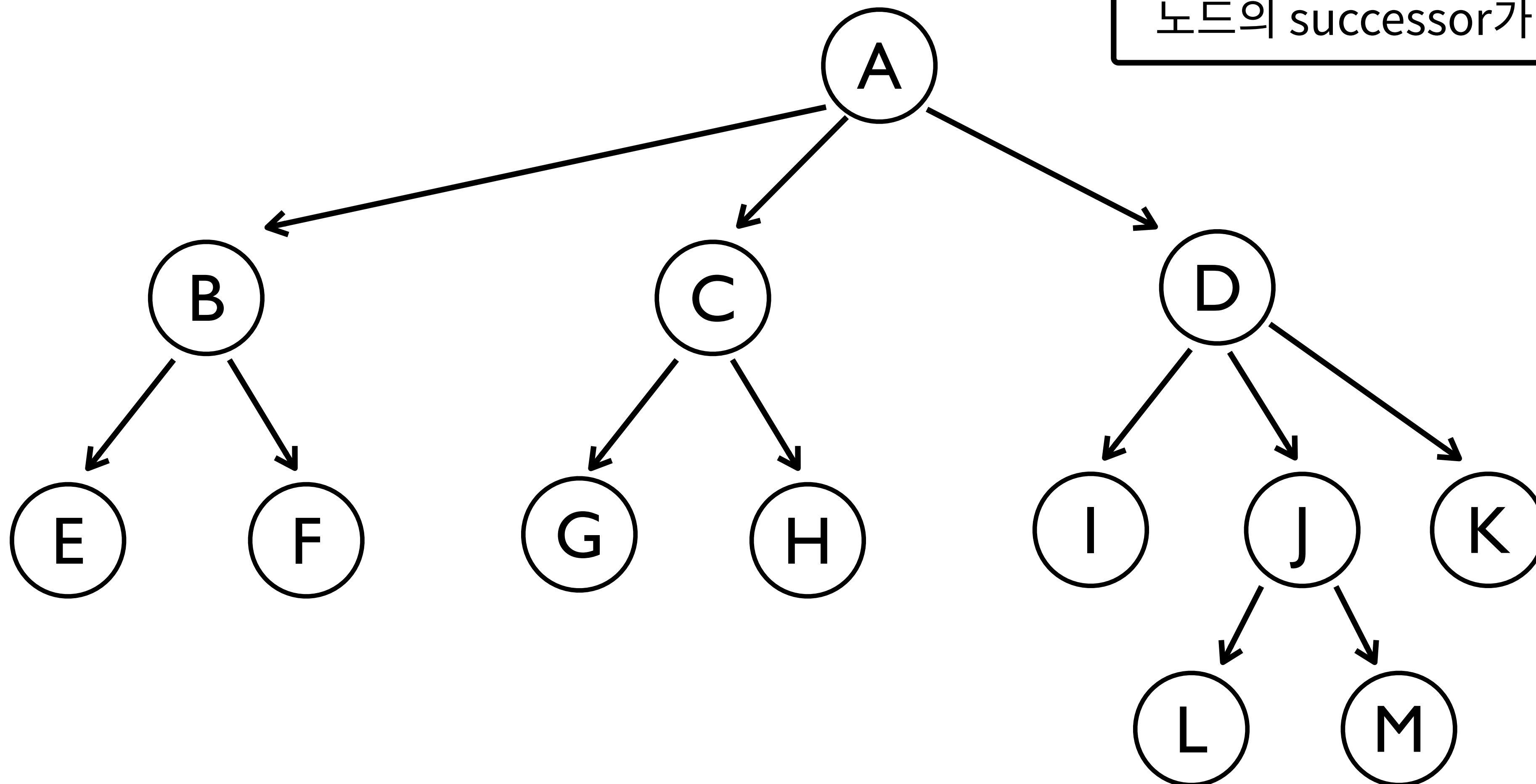
문제: 디렉터리(Directory) 구조 표현



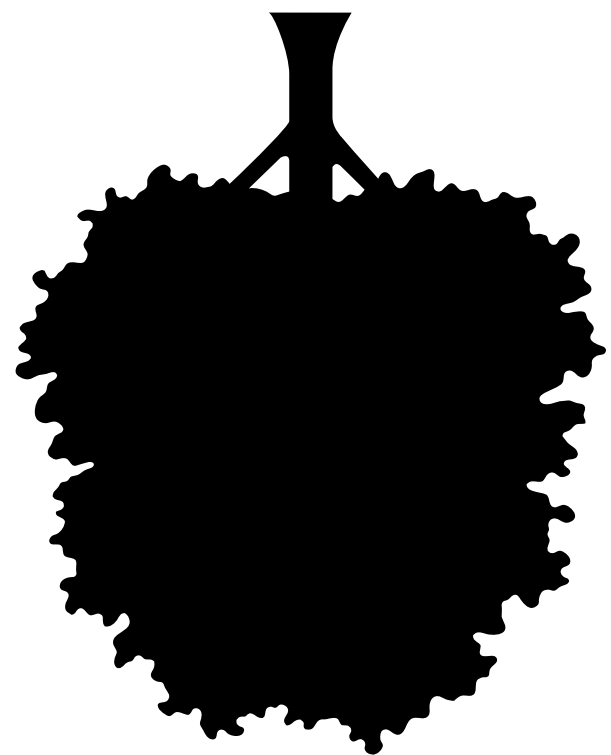
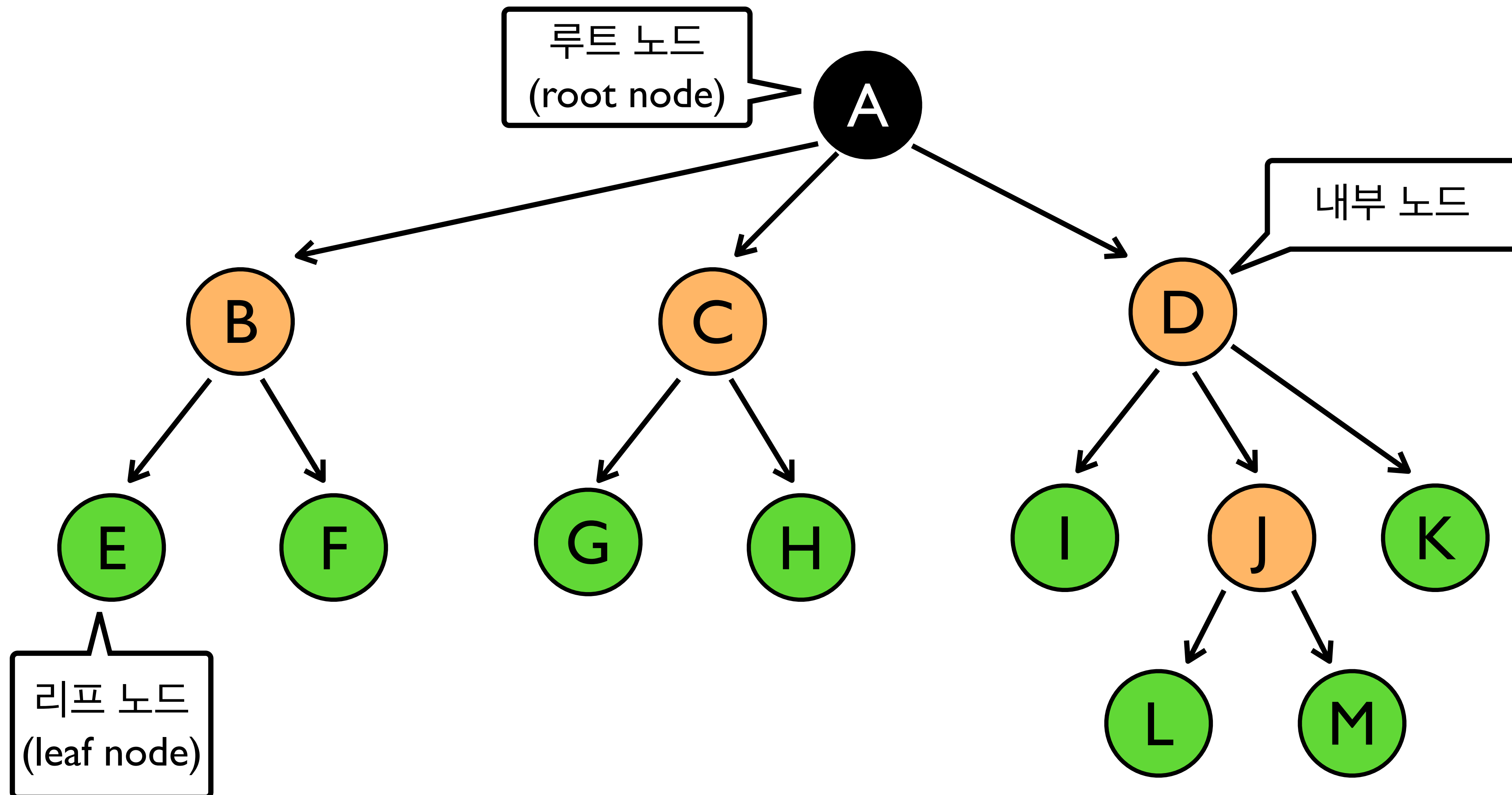
적합한 자료구조: 트리 (Tree)

- 트리는 계층적 자료를 표현하는데 이용하는 비 선형 (non-linear) 자료구조

노드의 successor가 두개 이상일 수 있음

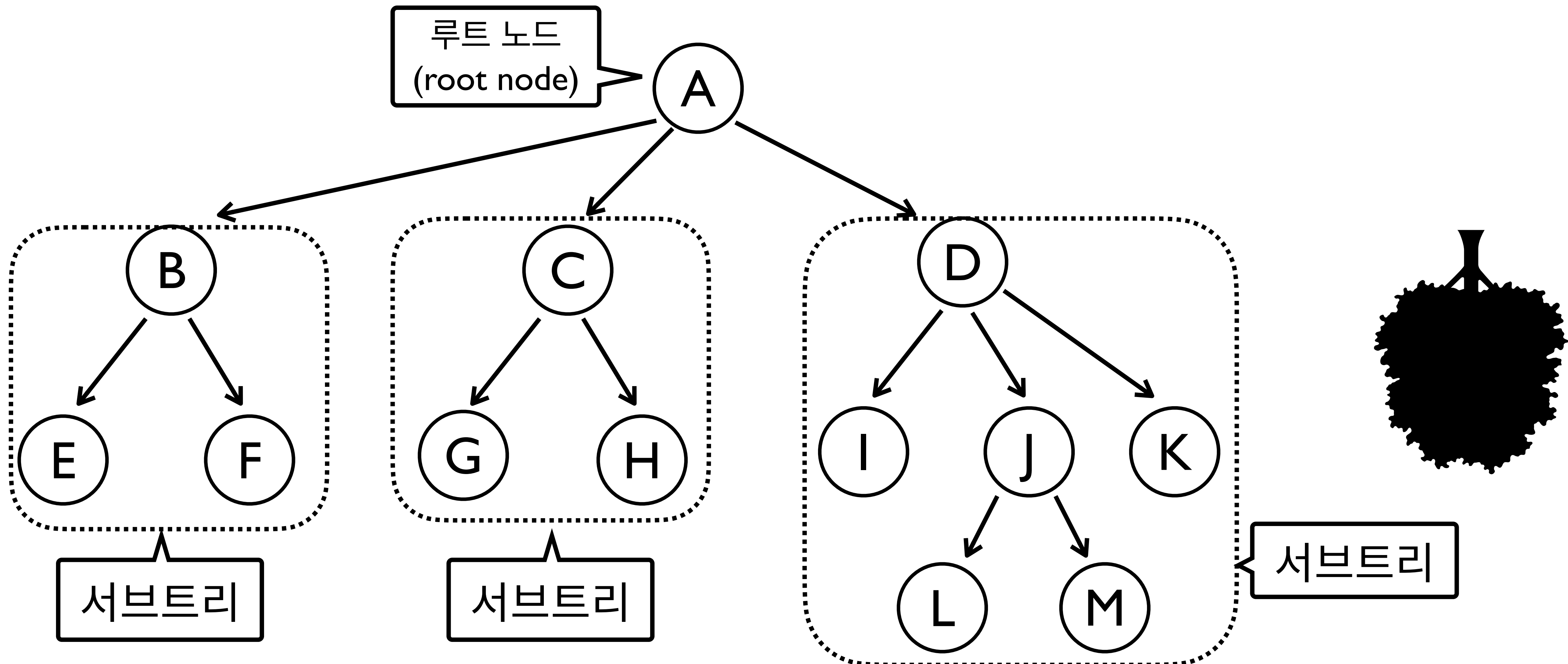


적합한 자료구조: 트리 (Tree)

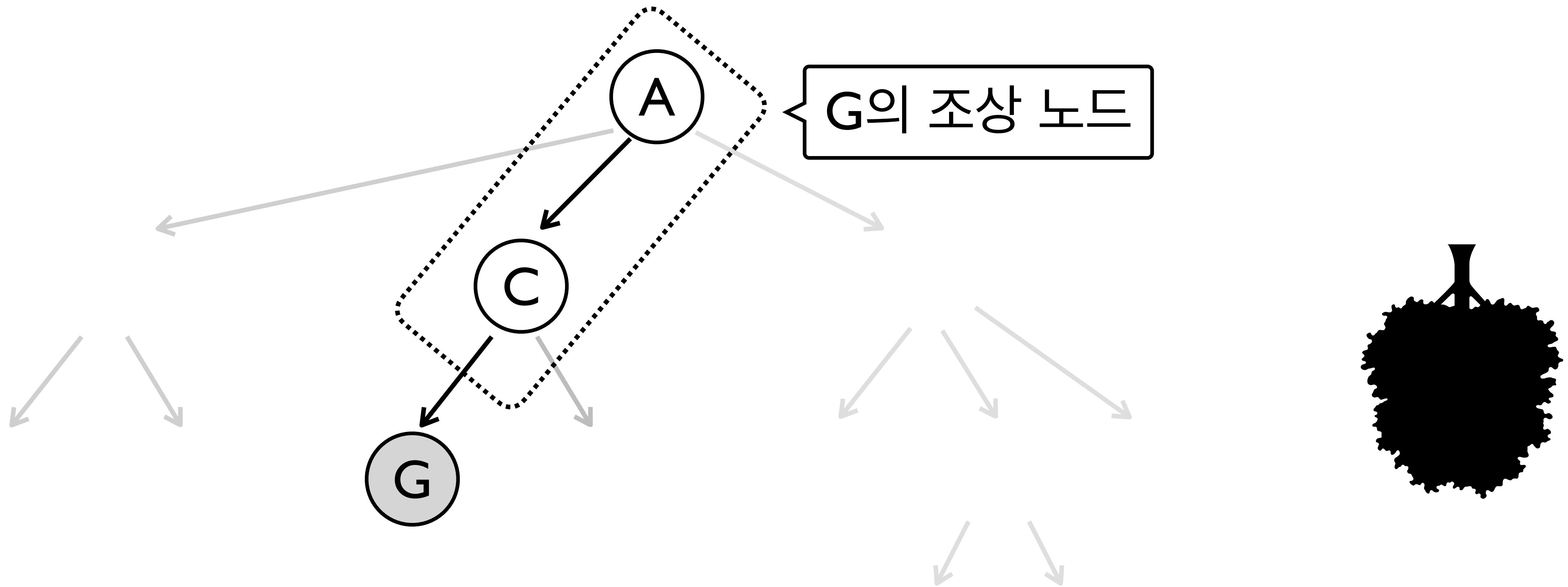


적합한 자료구조: 트리 (Tree)

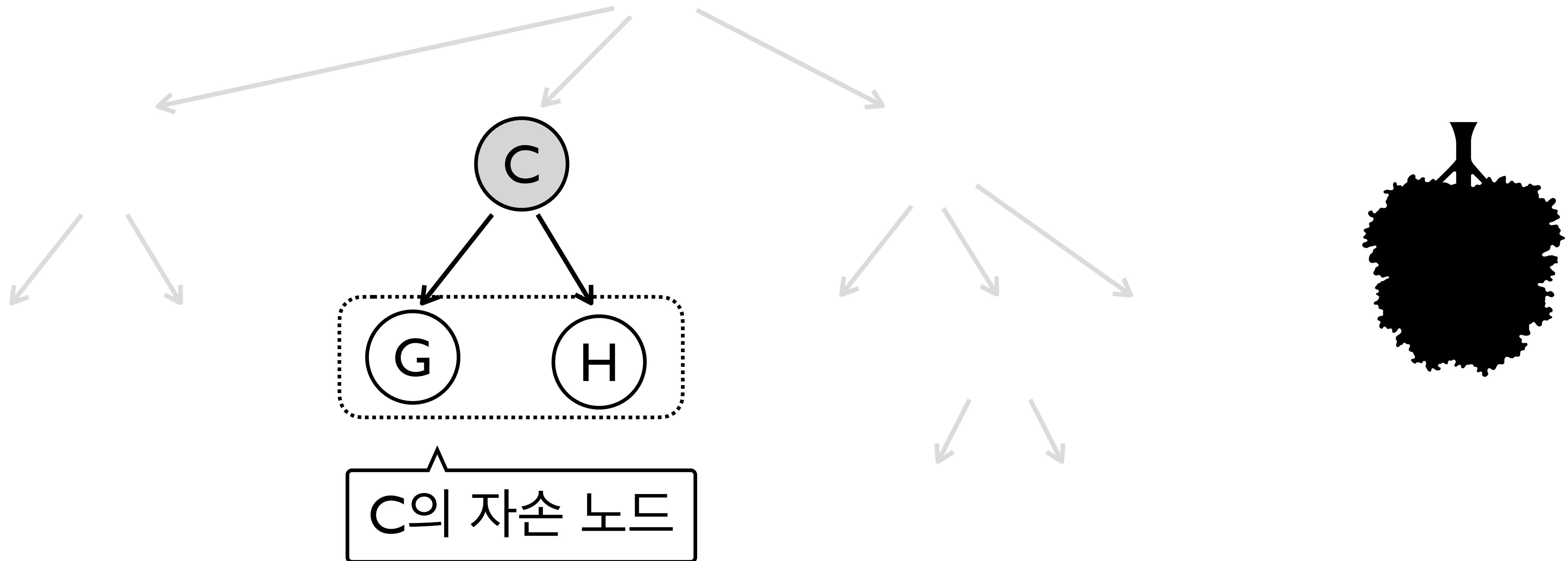
- 트리의 각 노드는 0개 이상의 서브트리를 가짐



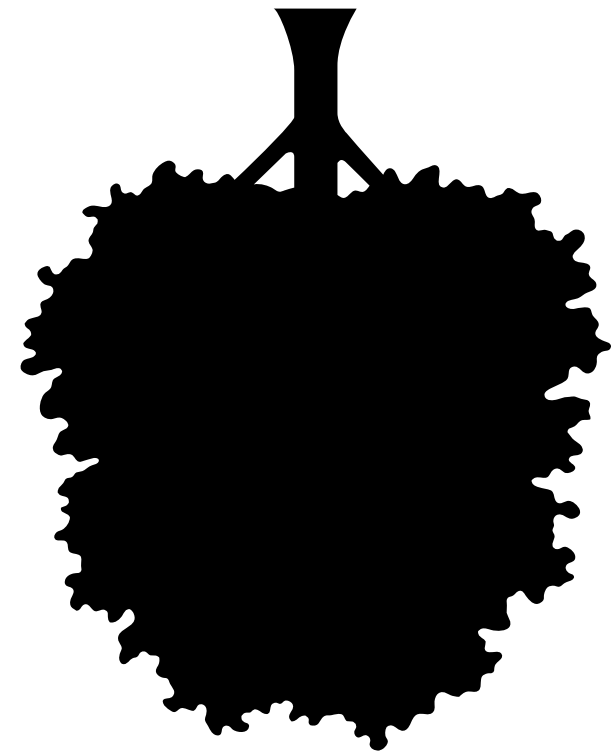
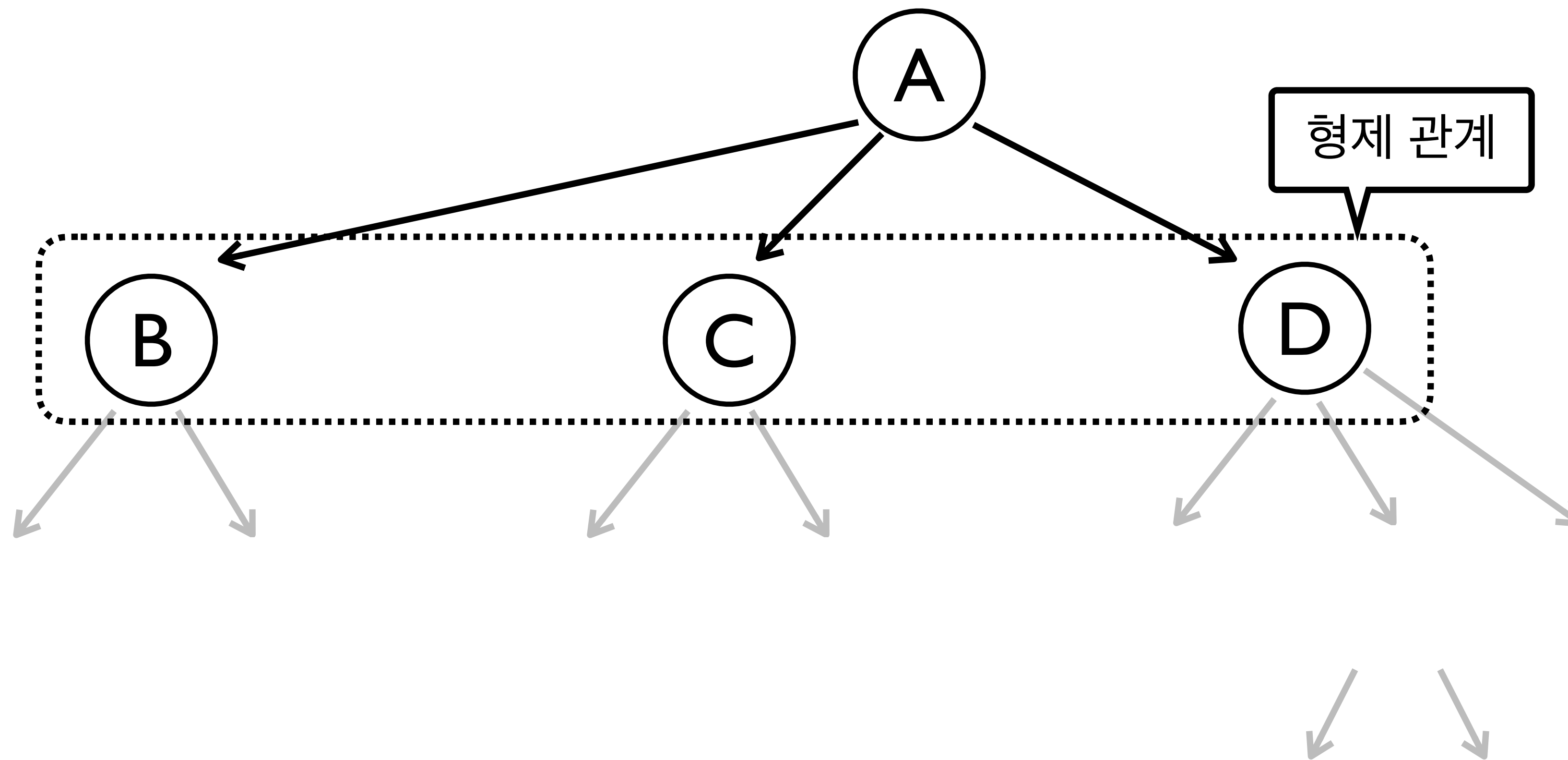
적합한 자료구조: 트리 (Tree)



적합한 자료구조: 트리 (Tree)

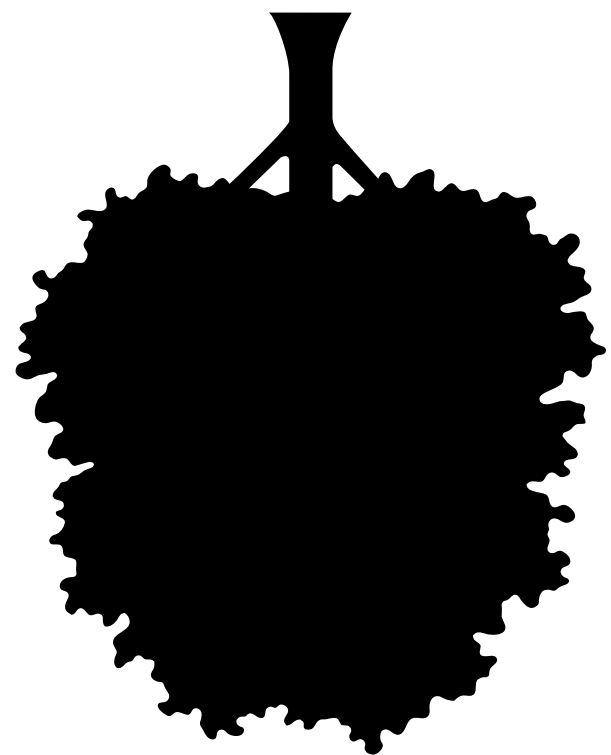
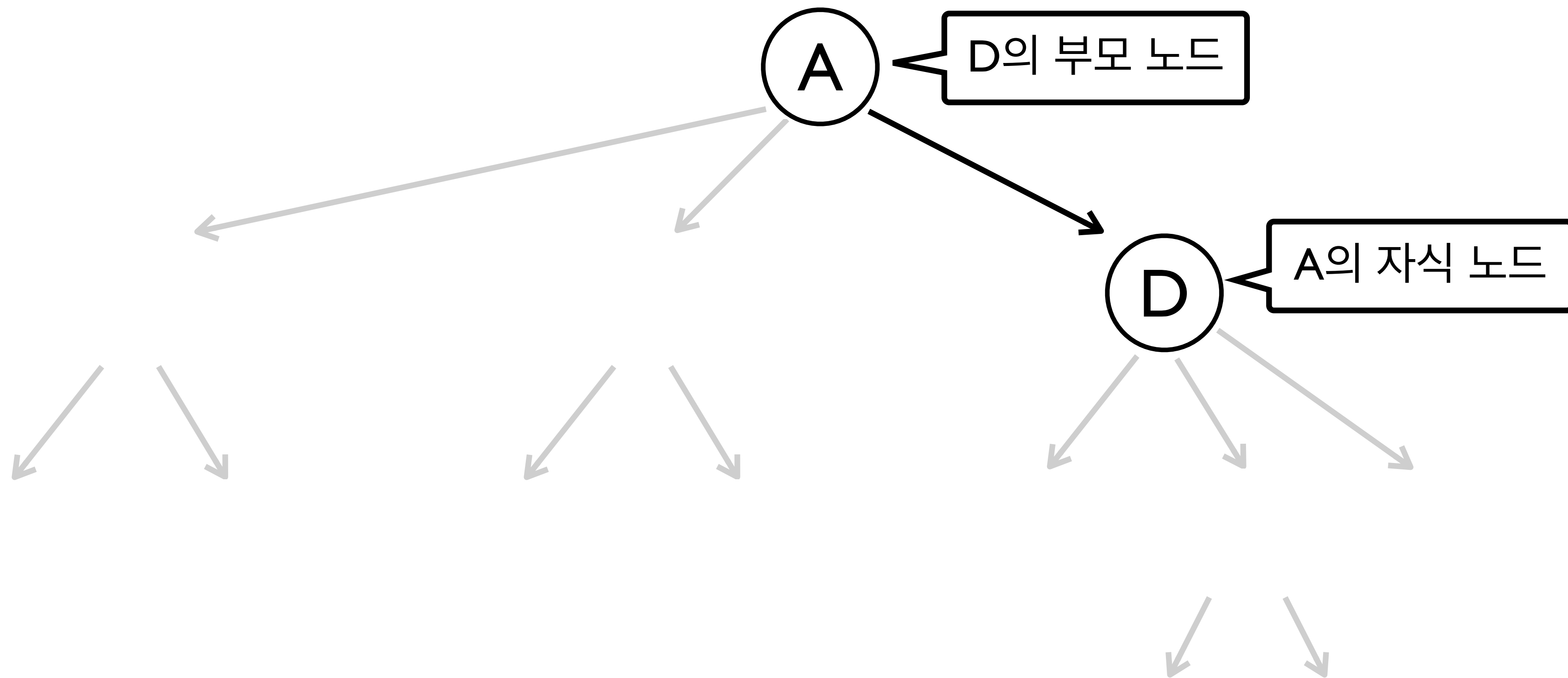


적합한 자료구조: 트리 (Tree)



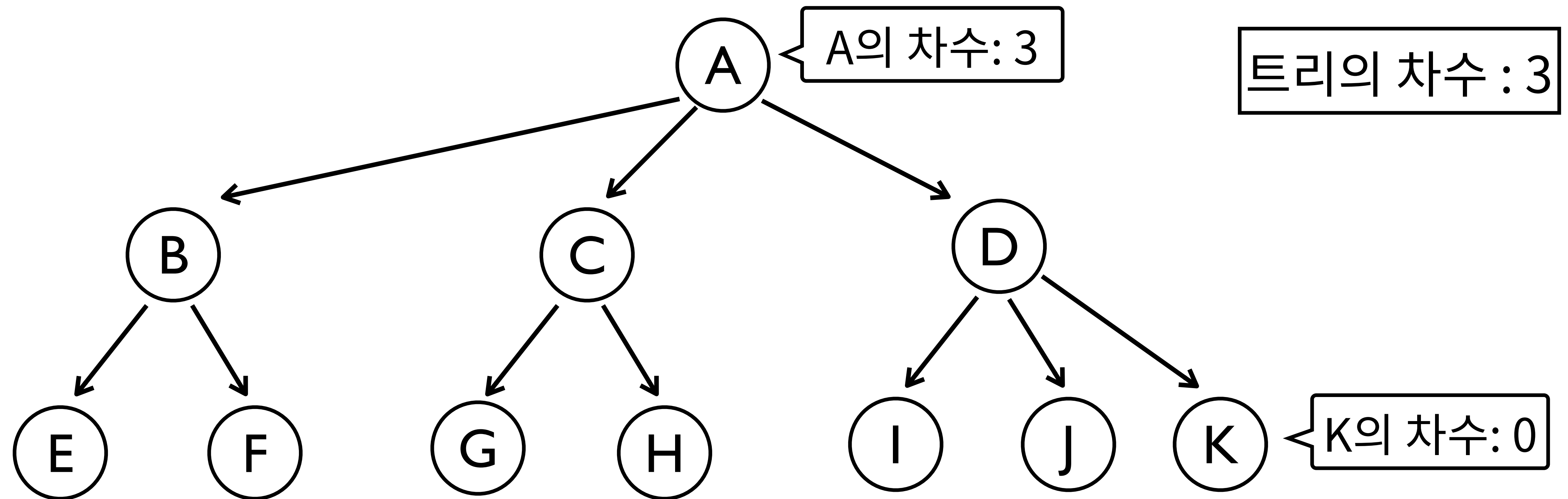
적합한 자료구조: 트리 (Tree)

- 루트 노드를 제외한 트리의 모든 노드들은 단 한개의 부모 노드만을 가짐



적합한 자료구조: 트리 (Tree)

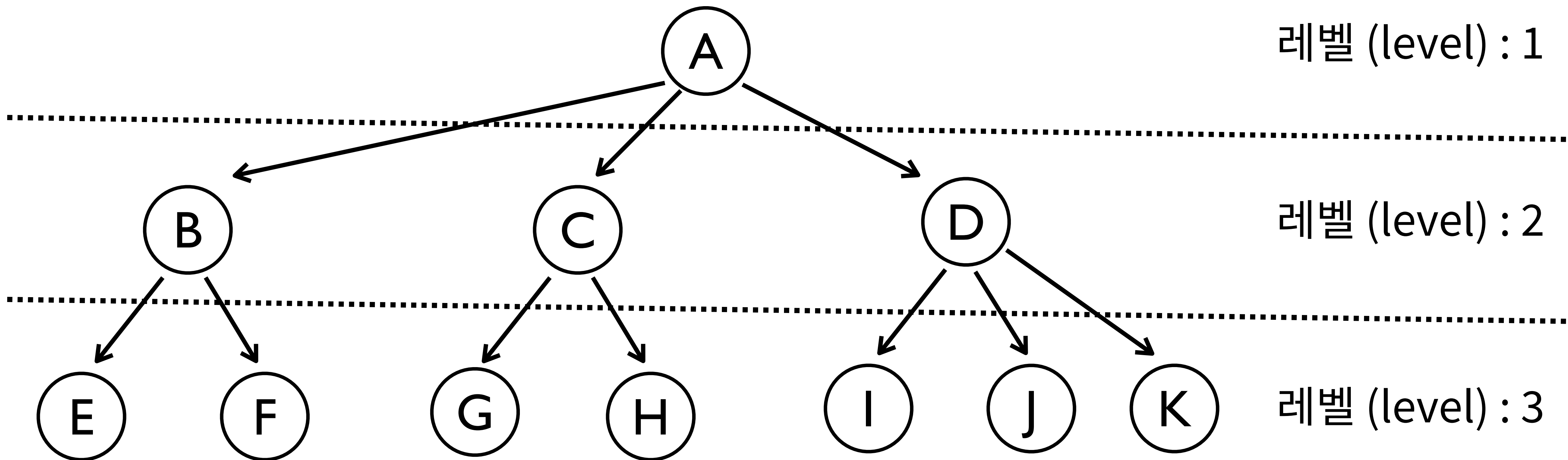
- 노드의 차수 (degree): 노드가 가진 자식 노드의 개수
- 트리의 차수: 가지고 있는 노드의 차수 중에서 가장 큰 차수

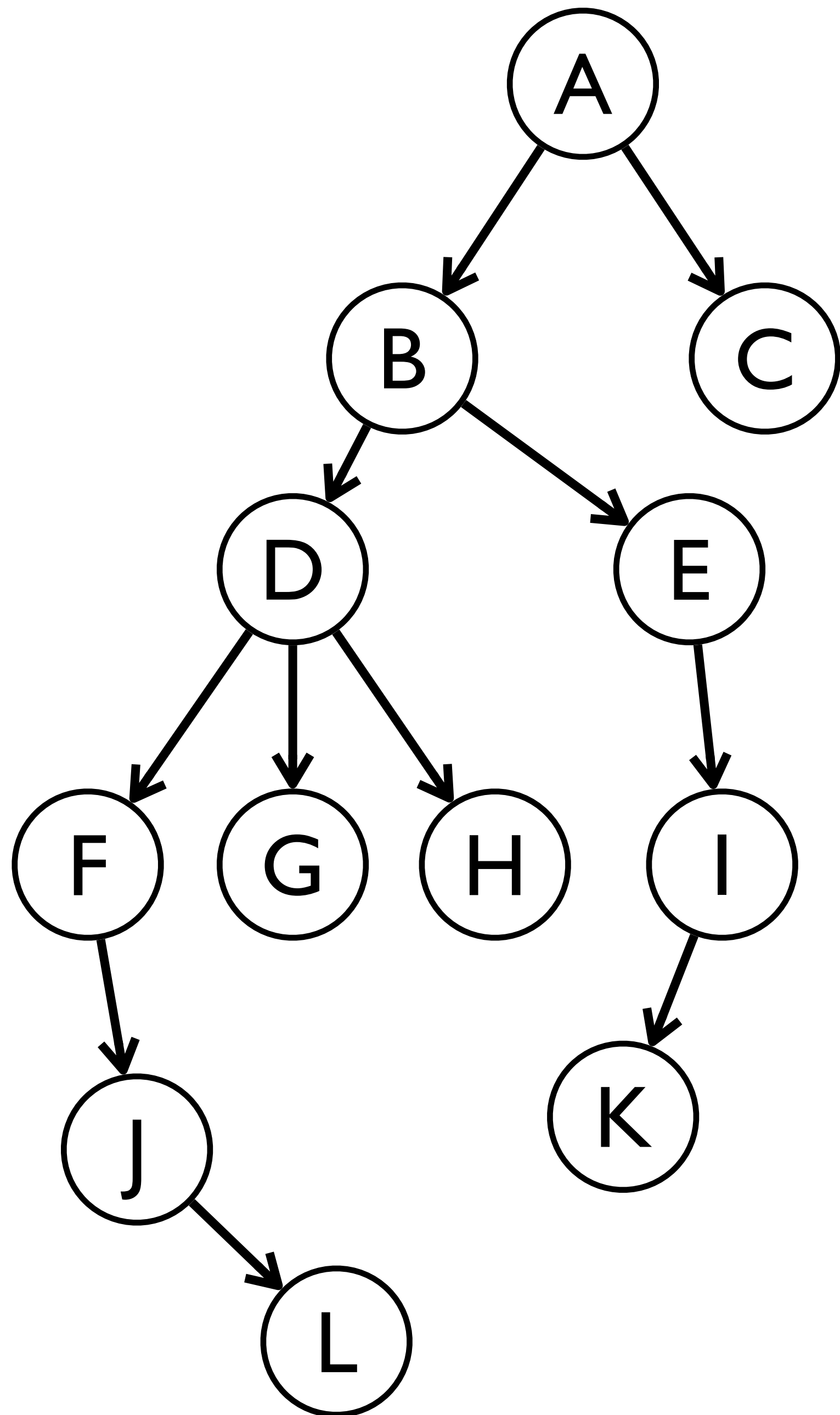


적합한 자료구조: 트리 (Tree)

- 레벨 (level) : 트리의 각 층에 번호를 매김 (루트의 레벨은 1)
- 트리의 높이 (height) : 트리가 가지고 있는 최대 레벨

트리의 높이 : 3





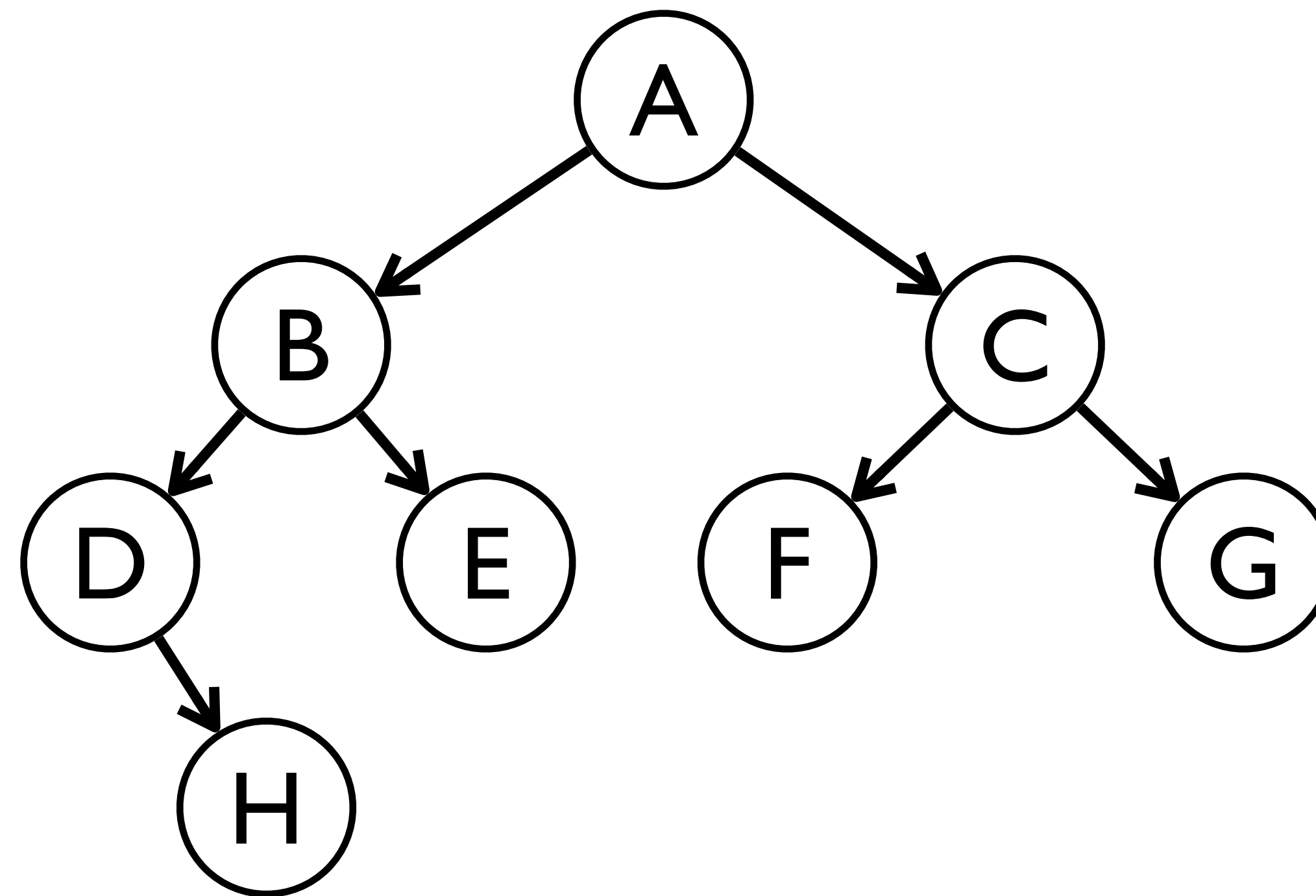
- 루트 노드:
- 리프 노드:
- 내부 노드:
- 노드 J의 조상 노드:
- 노드 D의 자손 노드:
- 노드 F의 형제 노드:
- 트리의 높이:
- 노드 D의 차수:
- 트리의 차수:
- 레벨 4에 위치한 노드:

트리 (Tree)

- 트리는 아래와 같이 재귀적(recursive)으로 정의됨

- 트리는 아래의 조건을 만족하는 노드의 집합
 - (1) 공집합 이거나
 - (2) 루트 노드가 하나 존재하고 루트 노드는 0개 이상의 (하위) 서브트리와 연결되어 있음. 서브트리 또한 트리의 정의를 만족함

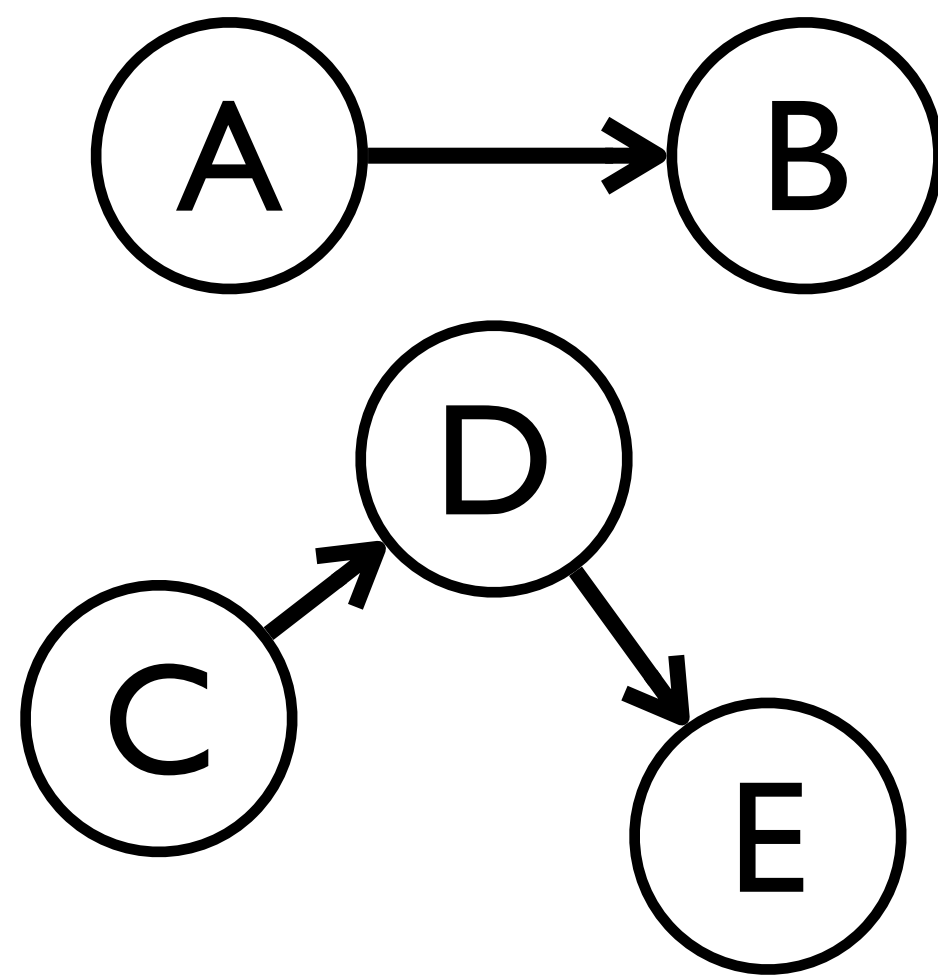
- 트리는 아래의 조건을 만족하는 노드의 집합
 - (1) 공집합 이거나
 - (2) 루트 노드가 하나 존재하고 루트 노드는 0개 이상의 (하위) 서브트리와 연결되어 있음. 서브트리 또한 트리의 정의를 만족함



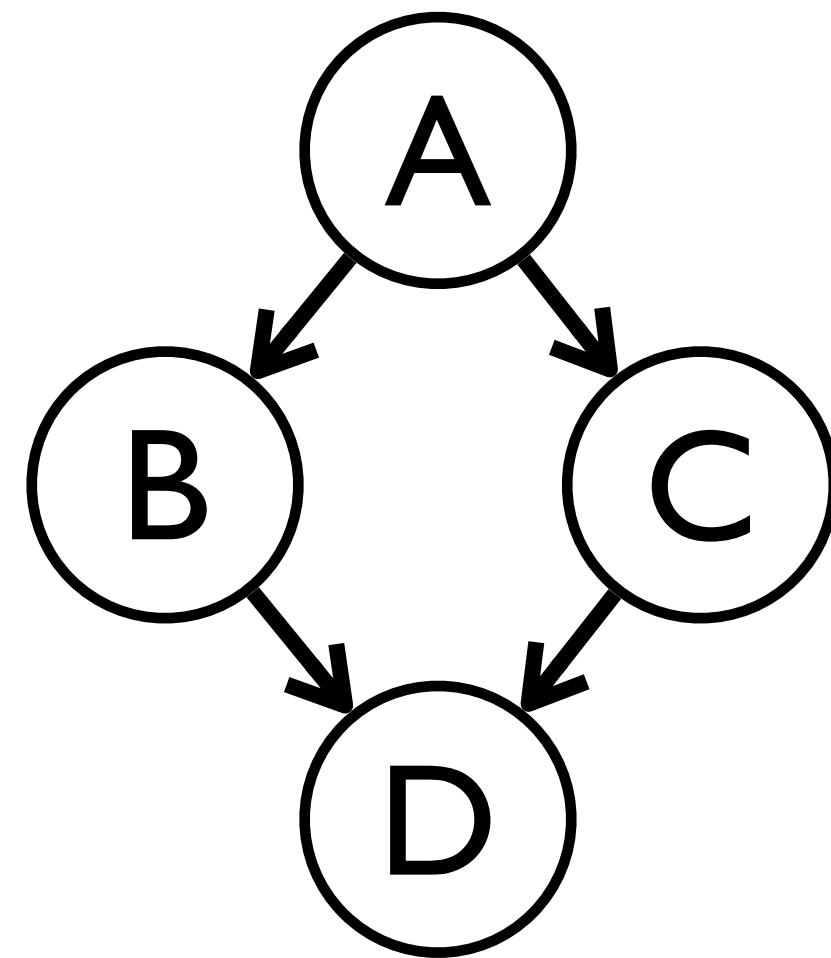
트리 예시

- 트리는 아래의 조건을 만족하는 노드의 집합
 - (1) 공집합 이거나
 - (2) 루트 노드가 하나 존재하고 루트 노드는 0개 이상의 (하위) 서브트리와 연결되어 있음. 서브트리 또한 트리의 정의를 만족함

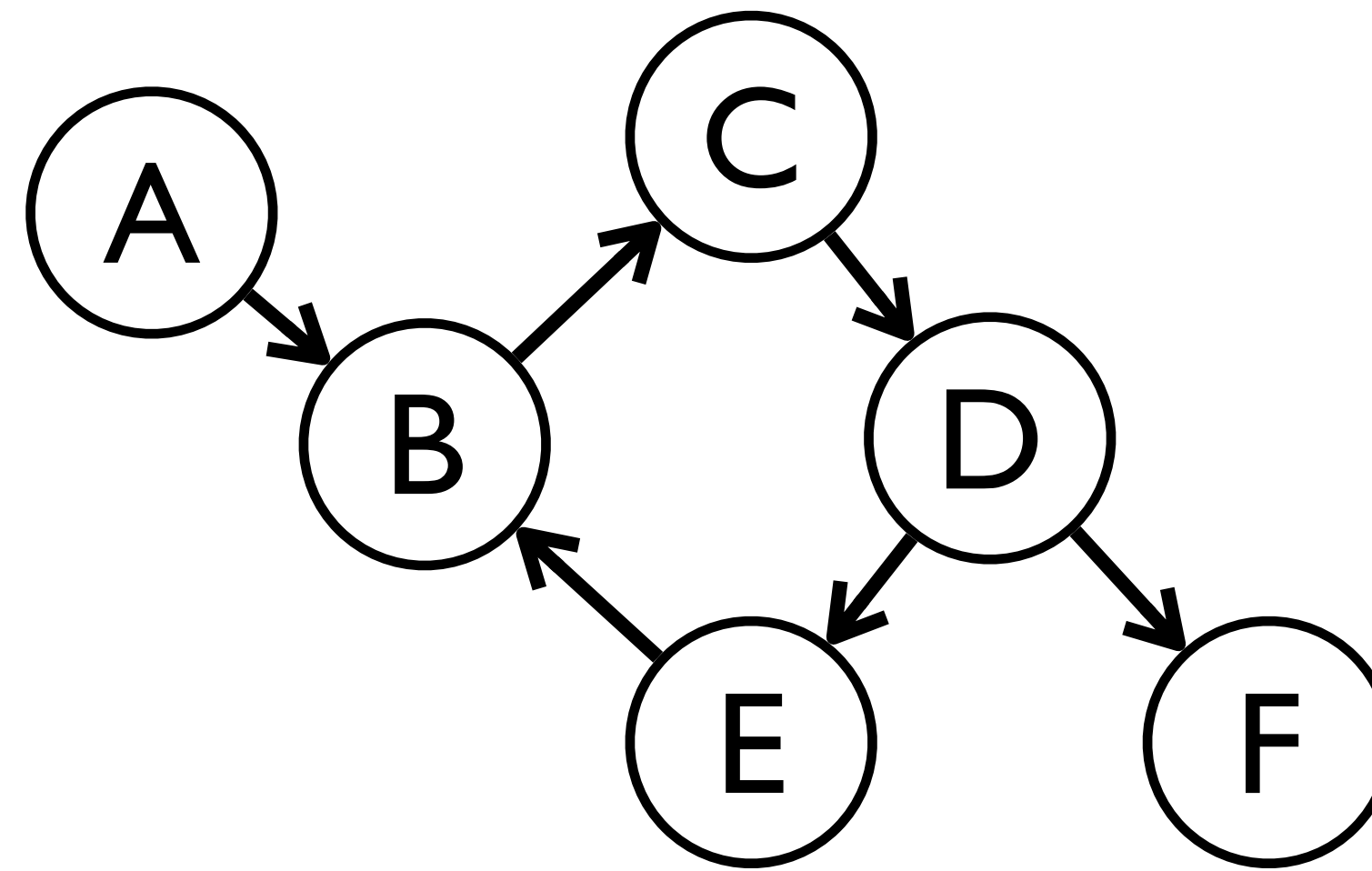
- 트리가 아닌 경우



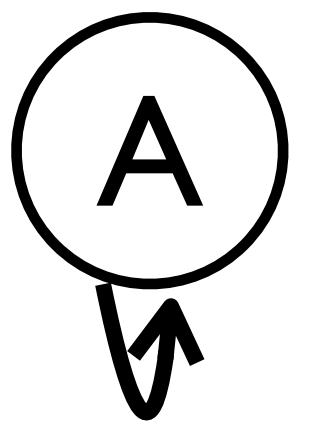
(a)



(b)



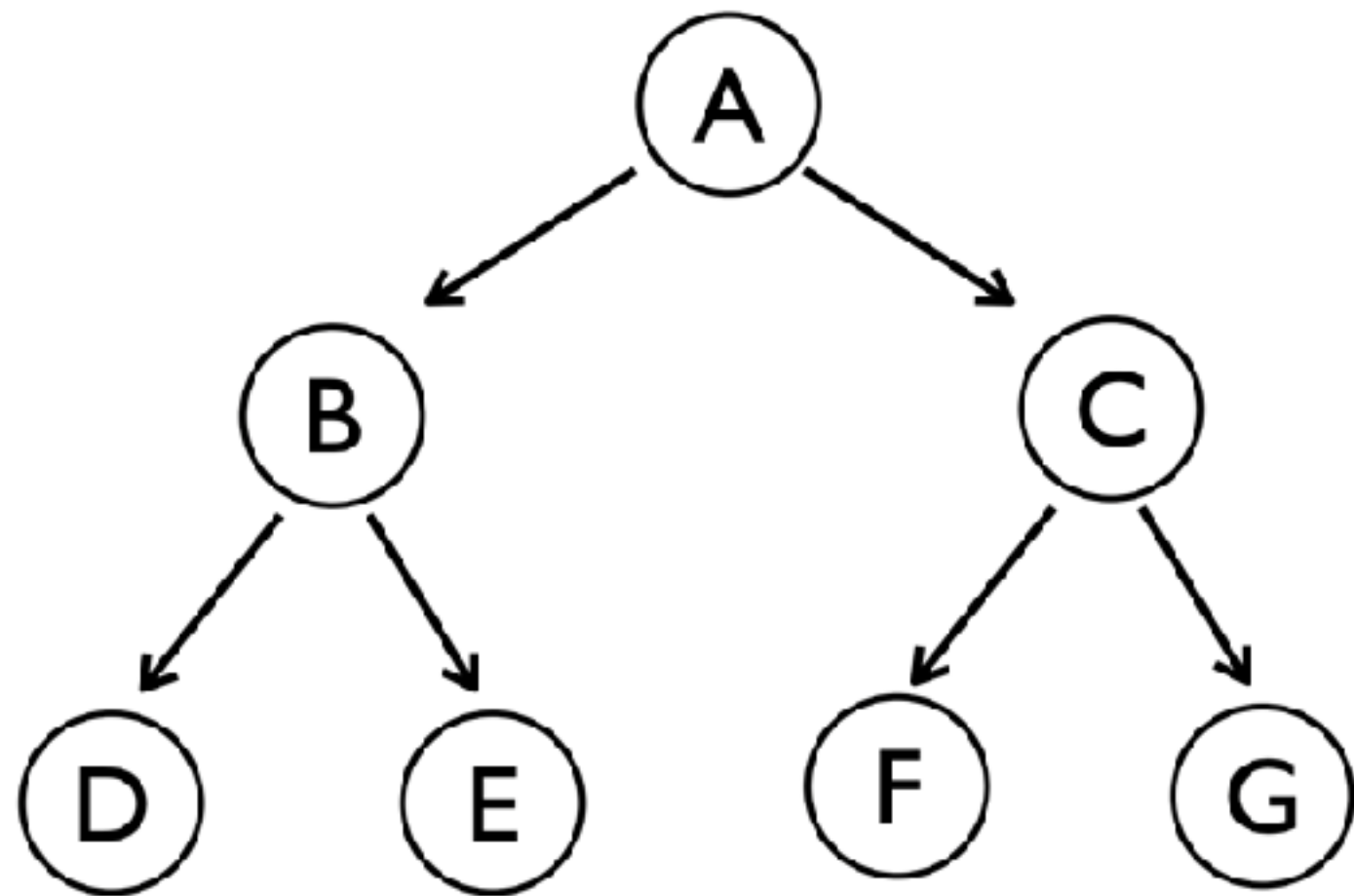
(c)



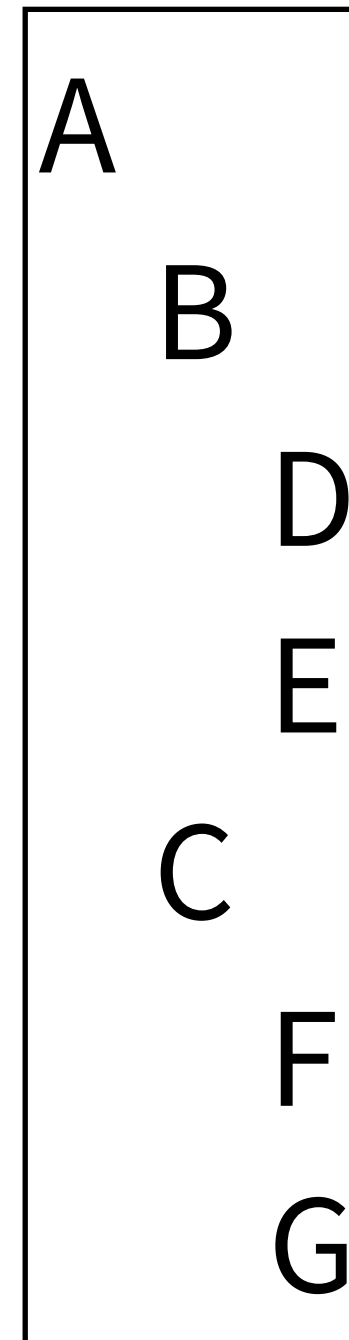
(c)

트리의 표현

(1) 노드를 이용한 표현

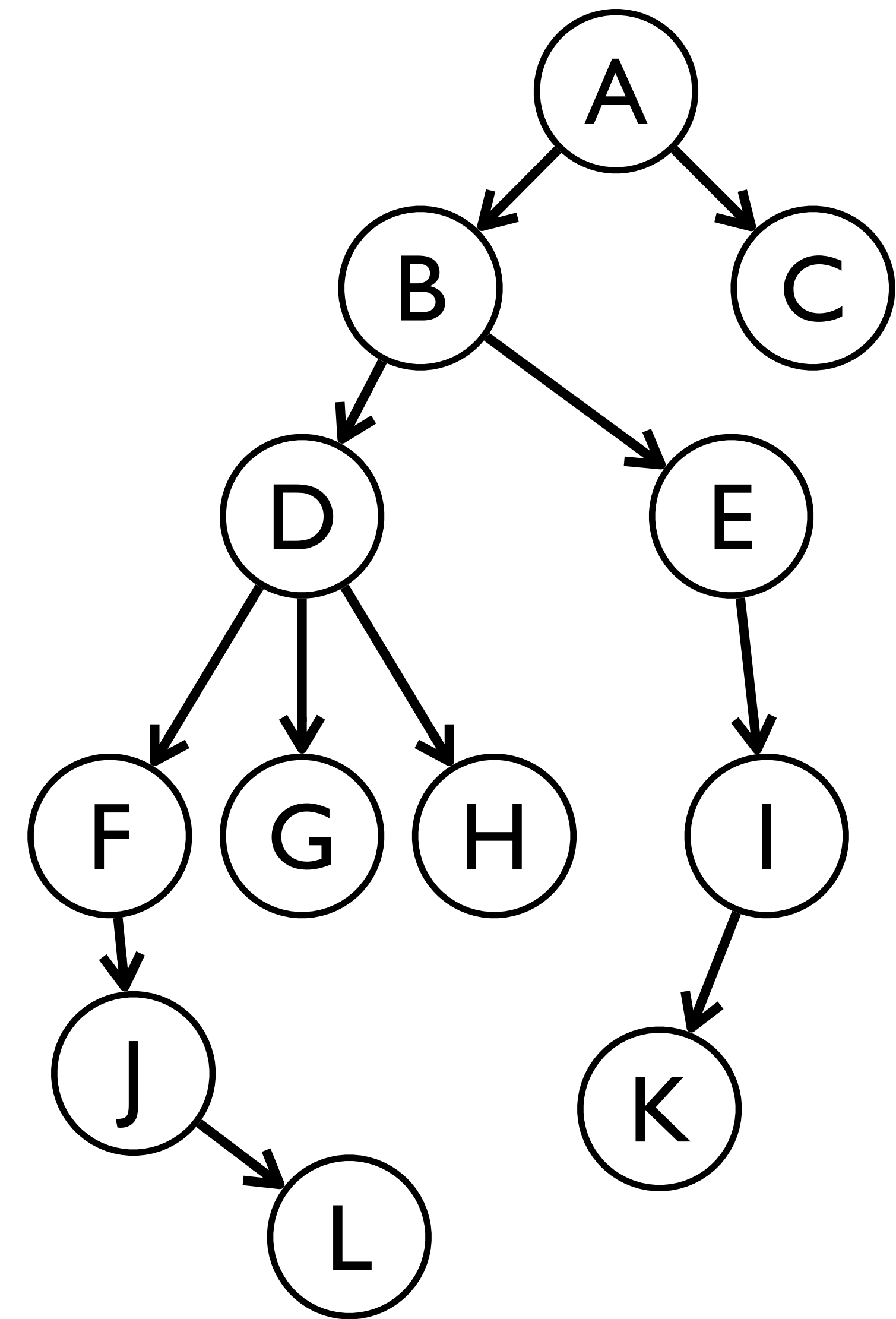


(2) 들여쓰기를 이용한 표현



(3) 괄호를 이용한 표현

$A(B(D E) C(F G))$



(1) 노드를 이용한 표현

(2) 들여쓰기를 이용한 표현

(3) 괄호를 이용한 표현

이진트리 (Binary Tree)

- 이진트리(binary tree)는 모든 노드가 2개의 서브트리를 갖는 트리
 - 모든 노드의 차수는 2 이하임 (최대 2개까지의 자식노드를 가질 수 있음)
- 이진트리(binary tree)의 정의는 아래와 같음

- (1) 공집합(빈 트리)이거나
- (2) 루트의 왼쪽 서브트리, 오른쪽 서브트리로 구성된 노드들의 유한집합.
- (3) 이진트리의 서브트리들은 모두 이진트리여야 함.

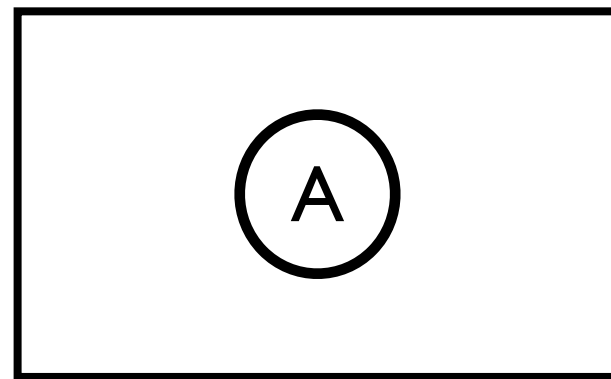
이진트리 (Binary Tree)

Definition

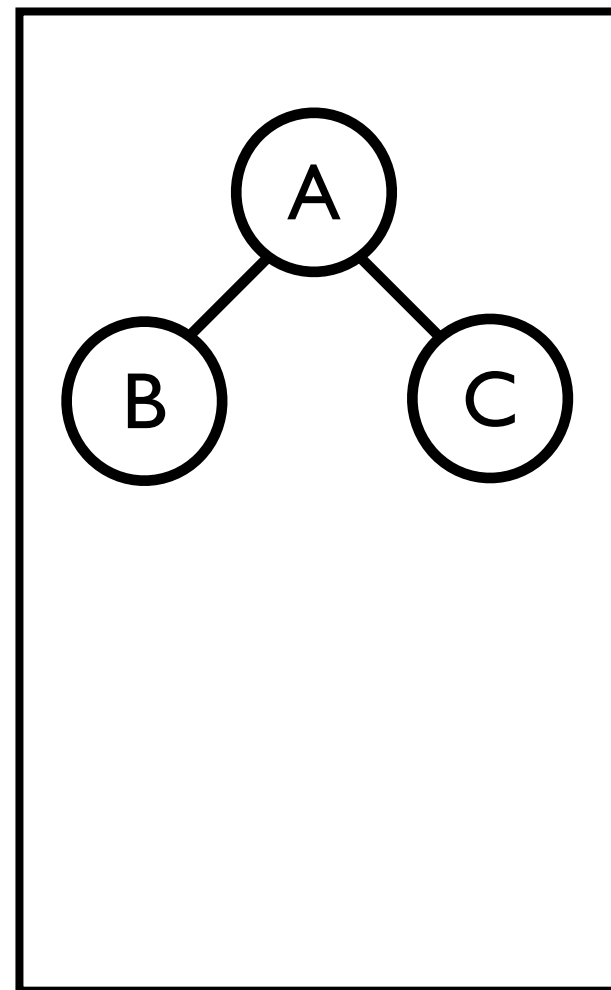
- (1) 공집합(빈 트리)이거나
- (2) 루트의 왼쪽 서브트리, 오른쪽 서브트리로 구성된 노드들의 유한집합.
- (3) 이진트리의 서브트리들은 모두 이진트리여야 함.



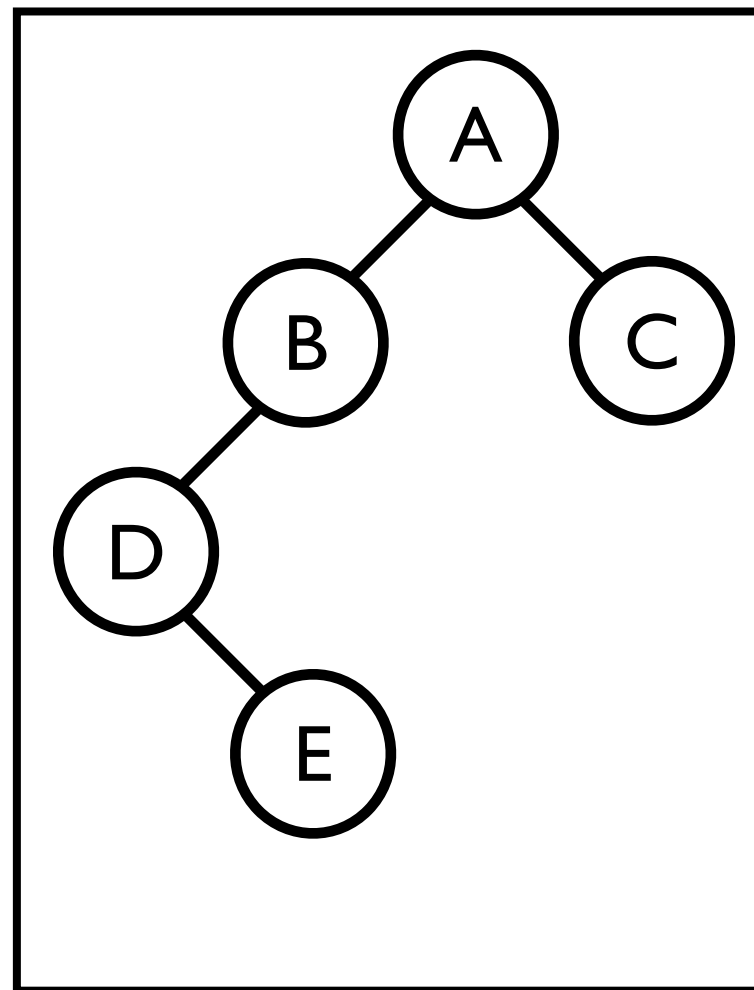
(a)



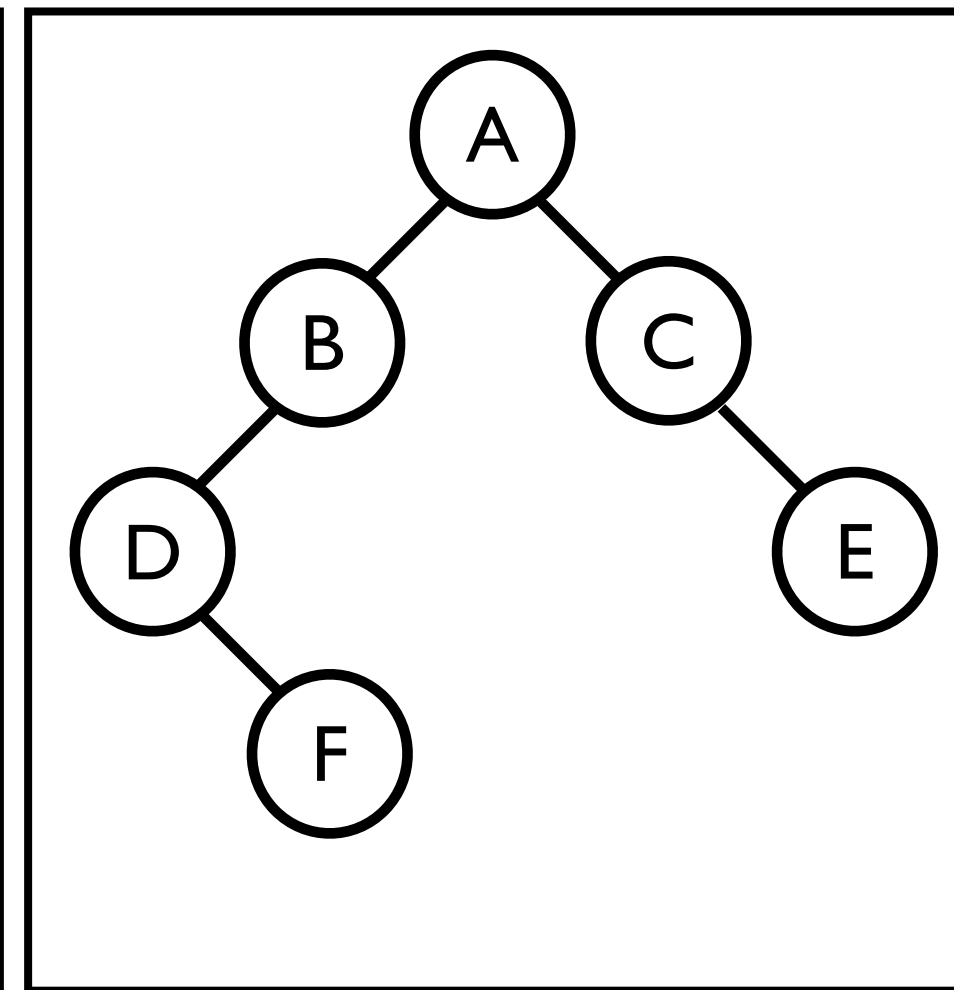
(b)



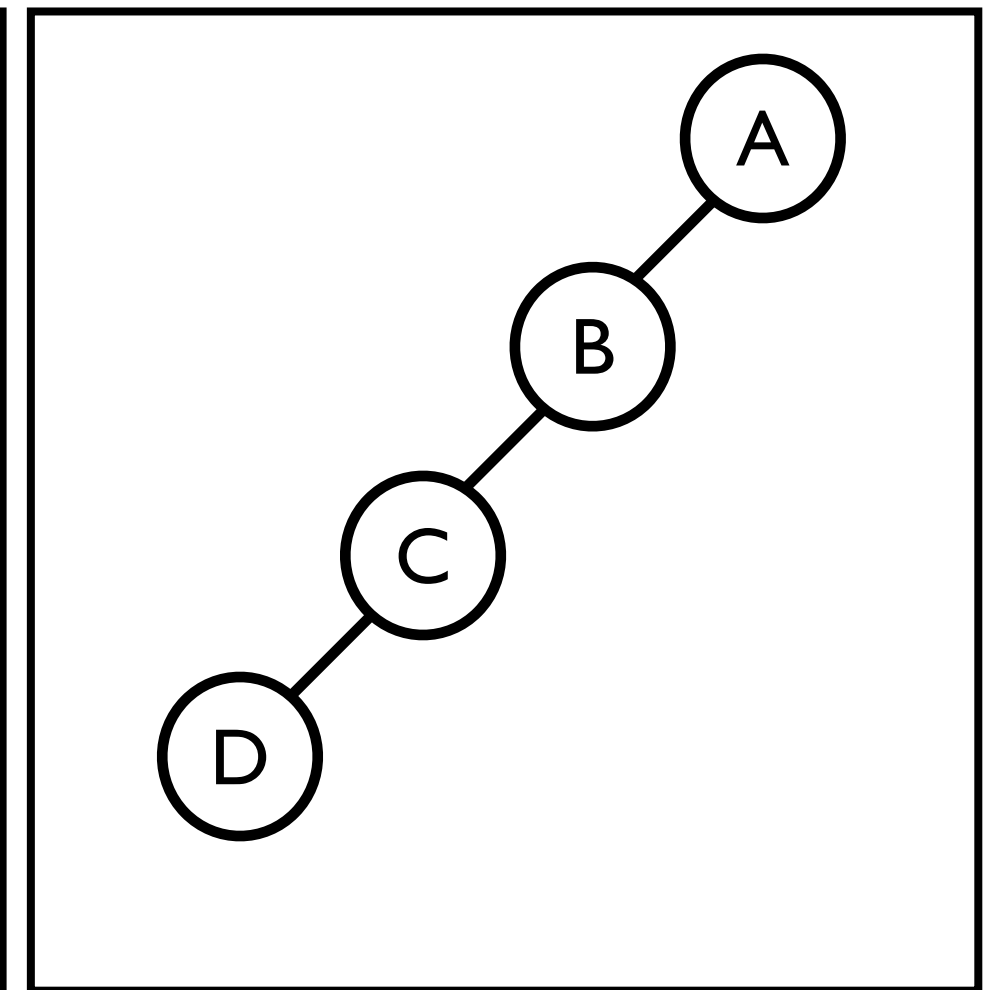
(c)



(d)



(e)



(f)

이진트리 (Binary Tree)

- 이진트리의 성질

(1) n 개의 노드를 가진 트리는 $n-1$ 개의 간선을 가짐

- 루트를 제외한 트리의 모든 노드가 하나의 부모 모드를 가지기 때문

(2) 높이가 h 인 이진트리는 h 개 이상, $2^h - 1$ 개 이하의 노드를 가짐

- 한 레벨에는 적어도 하나의 노드는 존재해야 함
- 하나의 노드는 최대 2개의 자식을 가질 수 있으므로 레벨 i 에서 최대 2^{i-1} 의 노드를 가짐

- 높이가 h 인 이진트리의 최대 노드의 개수는 $\sum_{i=1}^h 2^{i-1} = 2^h - 1$

이진트리 (Binary Tree)

- 이진트리의 성질

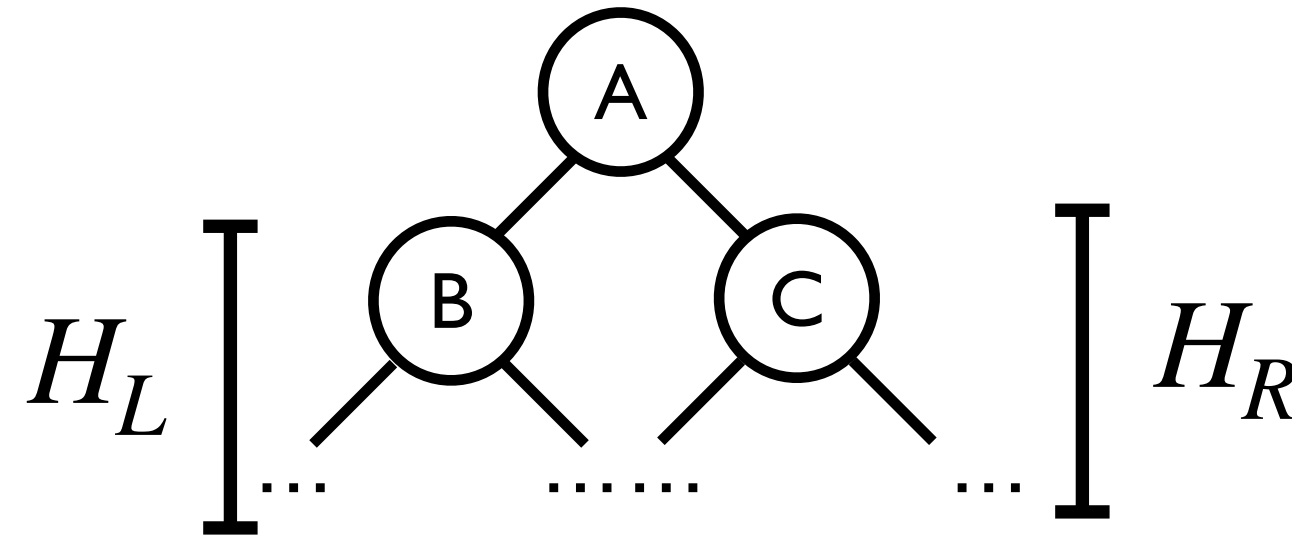
(3) n 개의 노드를 가지는 이진트리의 높이는 $\lceil \log_2(n + 1) \rceil$ 이상이고 n 이하이다.

- 한 레벨에는 적어도 하나의 노드는 존재해야 하므로 높이는 n 을 넘을 수 없음
- 높이가 h 인 이진트리가 가질 수 최대 노드의 개수는 $2^h - 1$ 임 (i.e., $n \leq 2^h - 1$)

이진트리 (Binary Tree)

- 균형(Balance)

- 이진 트리의 왼쪽 서브트리와 오른쪽 서브트리의 높이를 각각 H_L 과 H_R 라 하자



- 이 때 균형 인수 (balance factor) B 는 아래와 같이 정의됨

$$B = H_L - H_R$$

- B 의 값이 -1, 0, 1이면 균형을 이룬 트리라 함 (balanced tree)

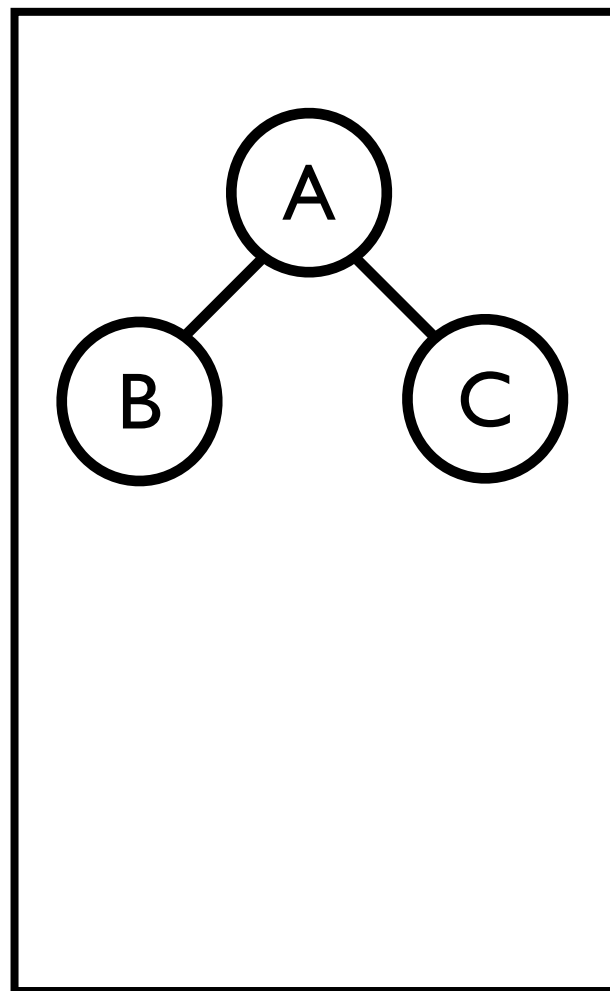
$B = 0 :$

$B < 0 :$

$B > 0 :$

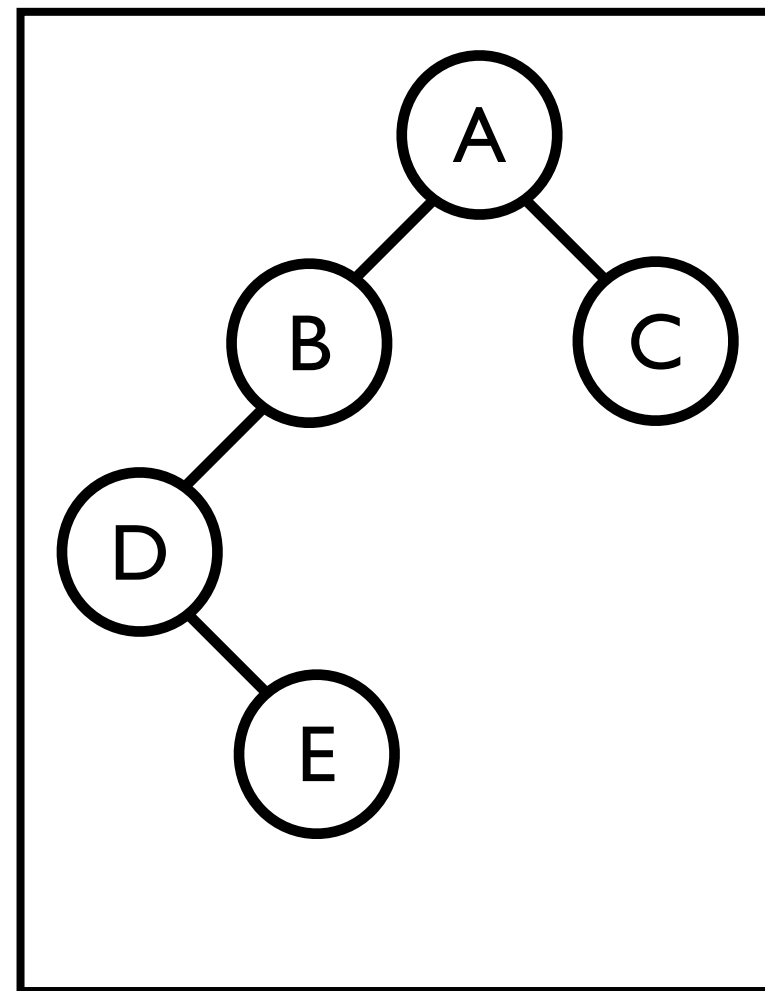
이진트리 (Binary Tree)

- 균형(Balance)



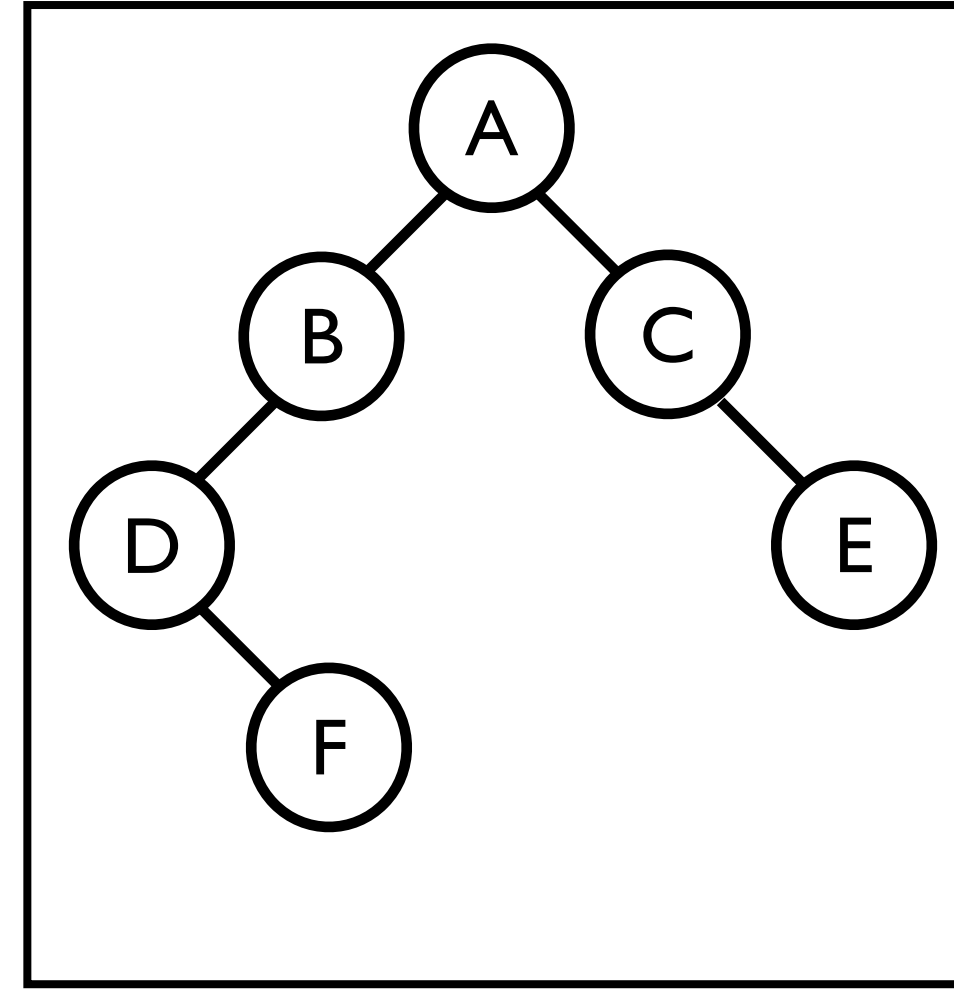
(a)

$H_L :$
 $H_R :$
 $B :$



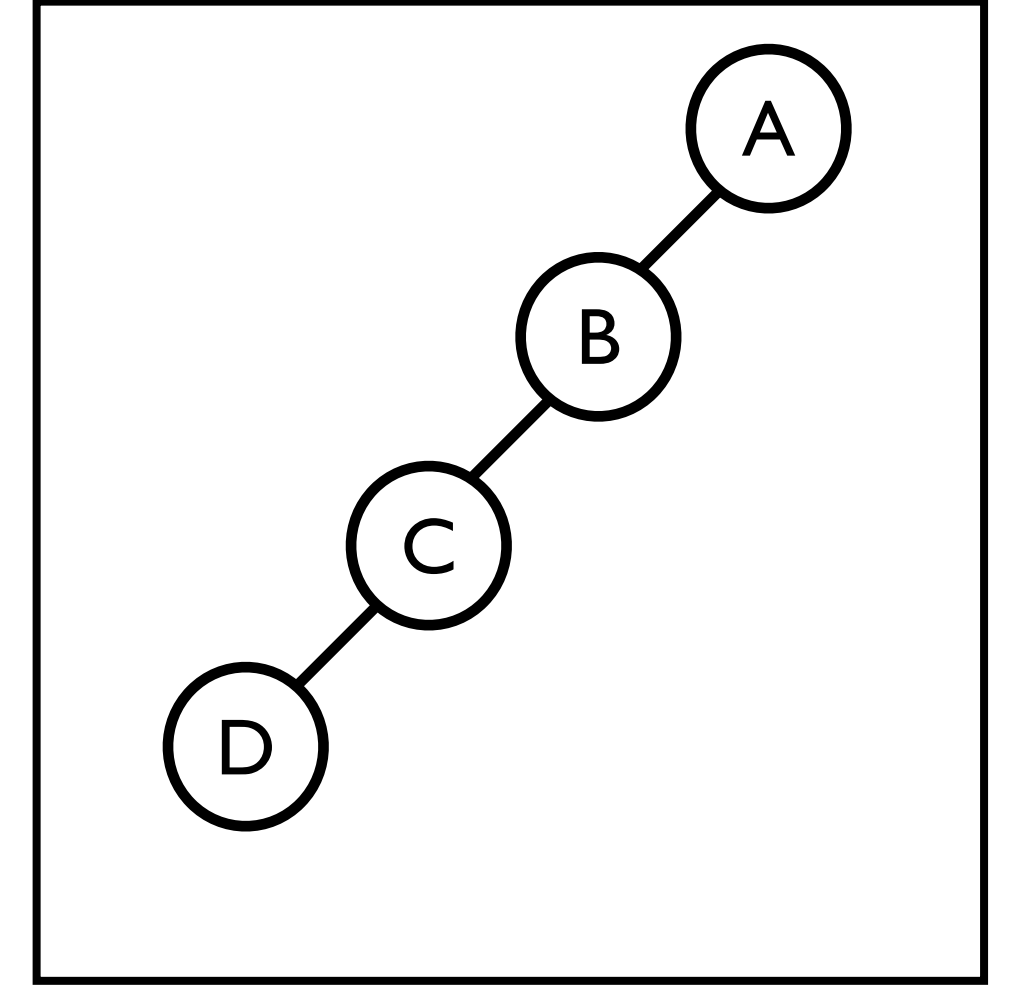
(b)

$H_L :$
 $H_R :$
 $B :$



(c)

$H_L :$
 $H_R :$
 $B :$

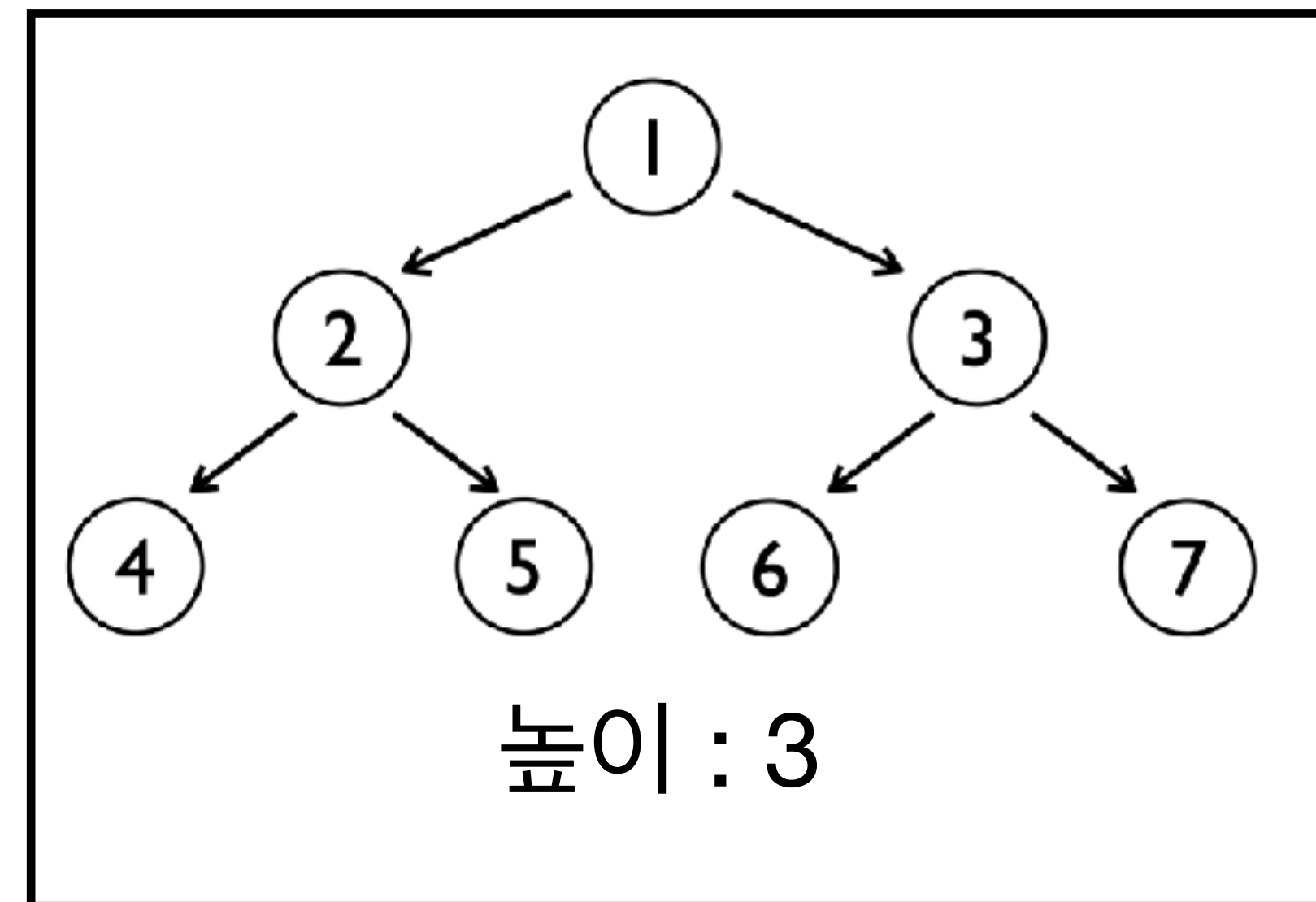
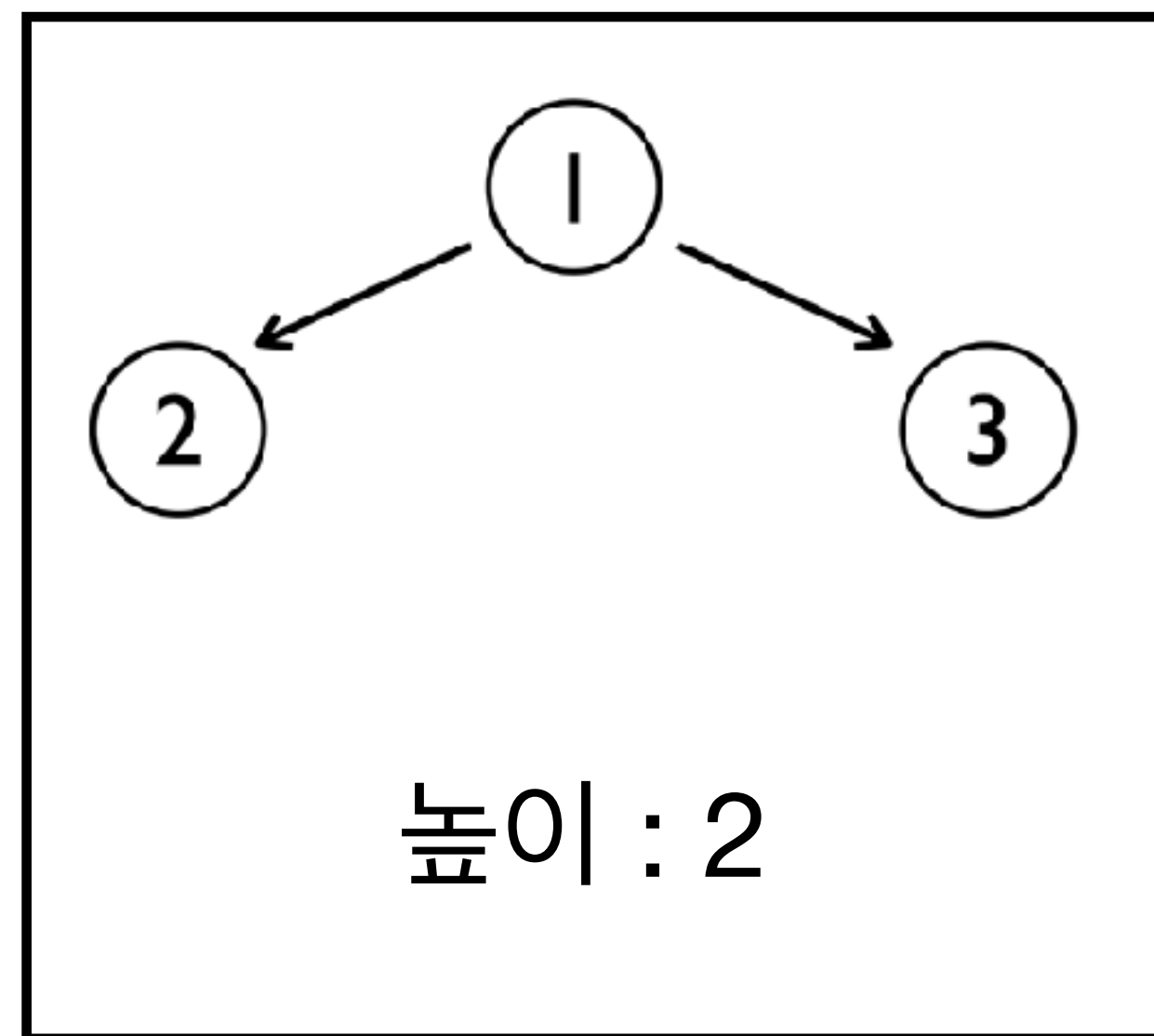
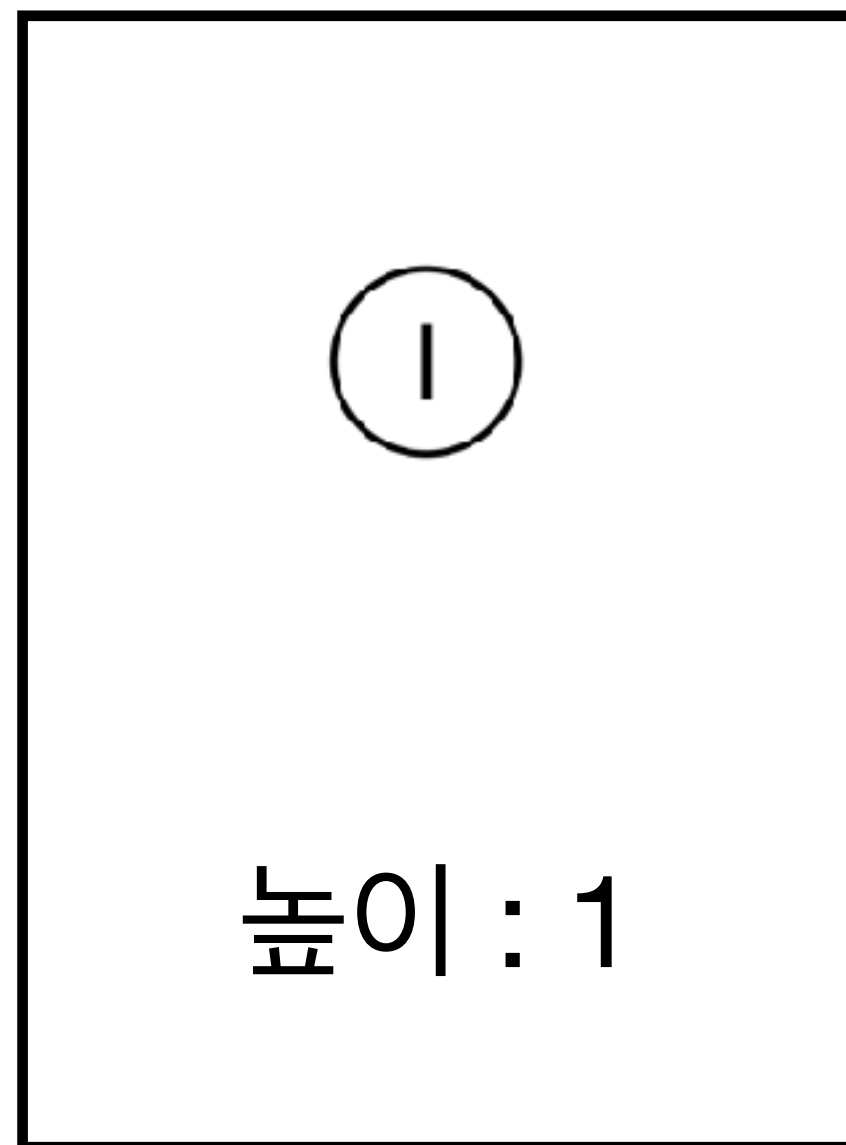


(d)

$H_L :$
 $H_R :$
 $B :$

포화 이진트리 (Full Binary Tree)

- 트리의 각 레벨에 노드가 꽉 차있는 이진트리



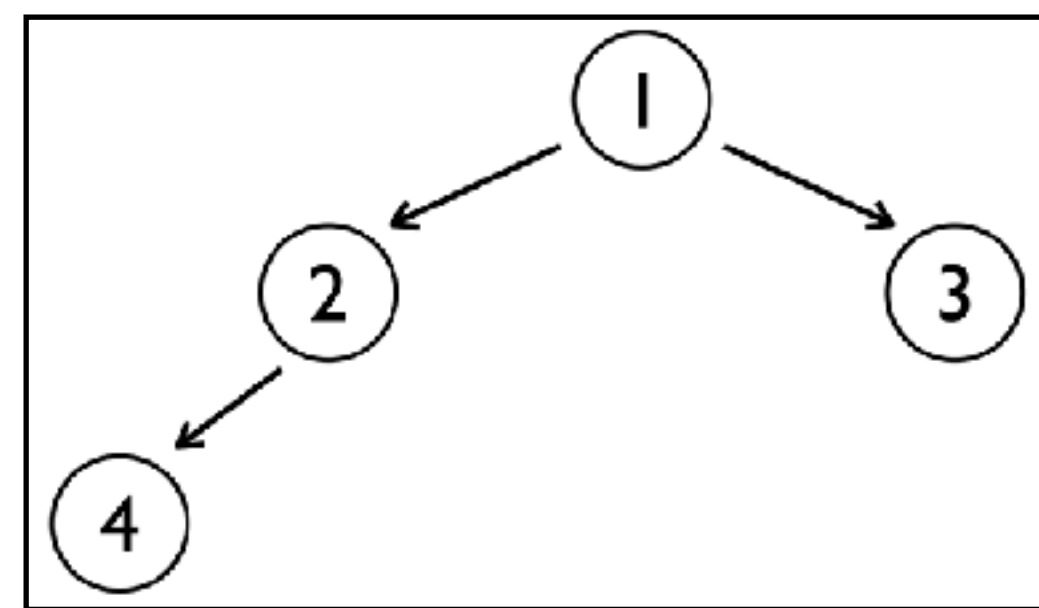
...

포화 이진트리 (Full Binary Tree)

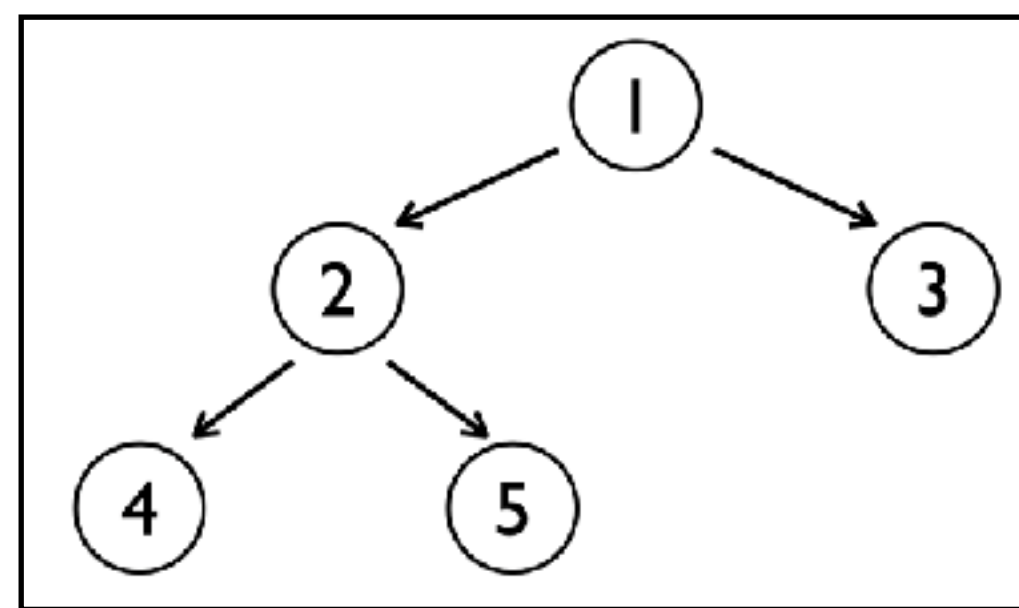
- 트리의 각 레벨에 노드가 꼭 차있는 이진트리
- 포화 이진트리의 특징
 - 모든 내부 노드들이 두개의 자식 노드들을 가짐
 - 모든 리프 노드들은 같은 레벨에 위치함
 - n 개의 노드가 있을 때, 포화 이진트리의 높이 h 는 $\log_2(n + 1) - 1$ 임
 - 높이가 h 일 경우 노드의 개수는 항상 $2^h - 1$ 임

완전 이진트리 (Complete Binary Tree)

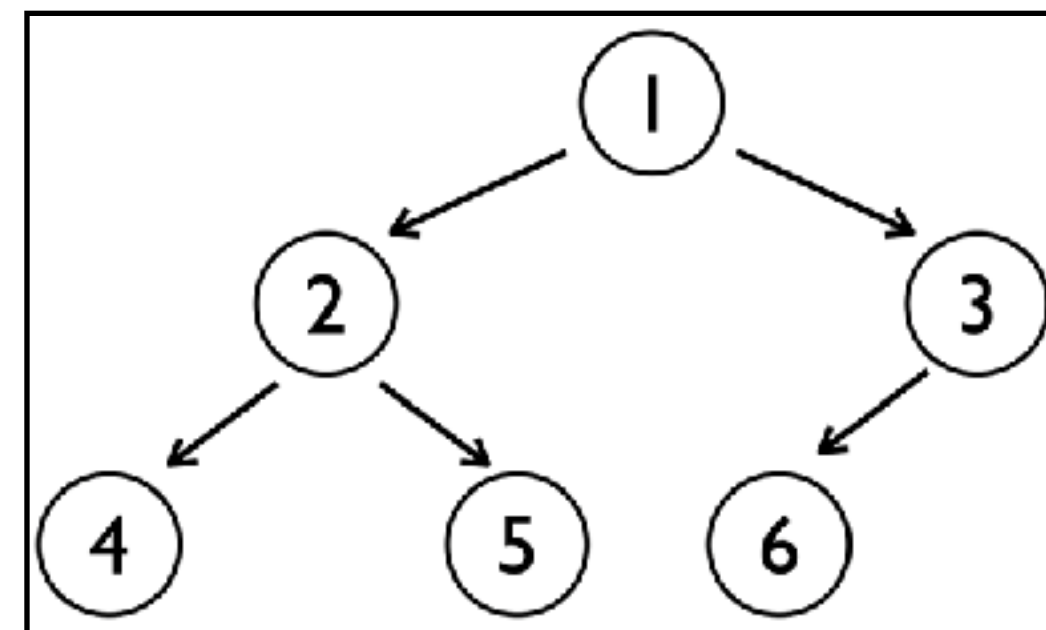
- (1) 높이가 k 인 트리에서 레벨 1부터 $k - 1$ 까지는 노드가 모두 채워져 있고
- (2) 마지막 레벨 k 에서는 왼쪽부터 오른쪽으로 노드가 순서대로 채워져 있는 이진트리



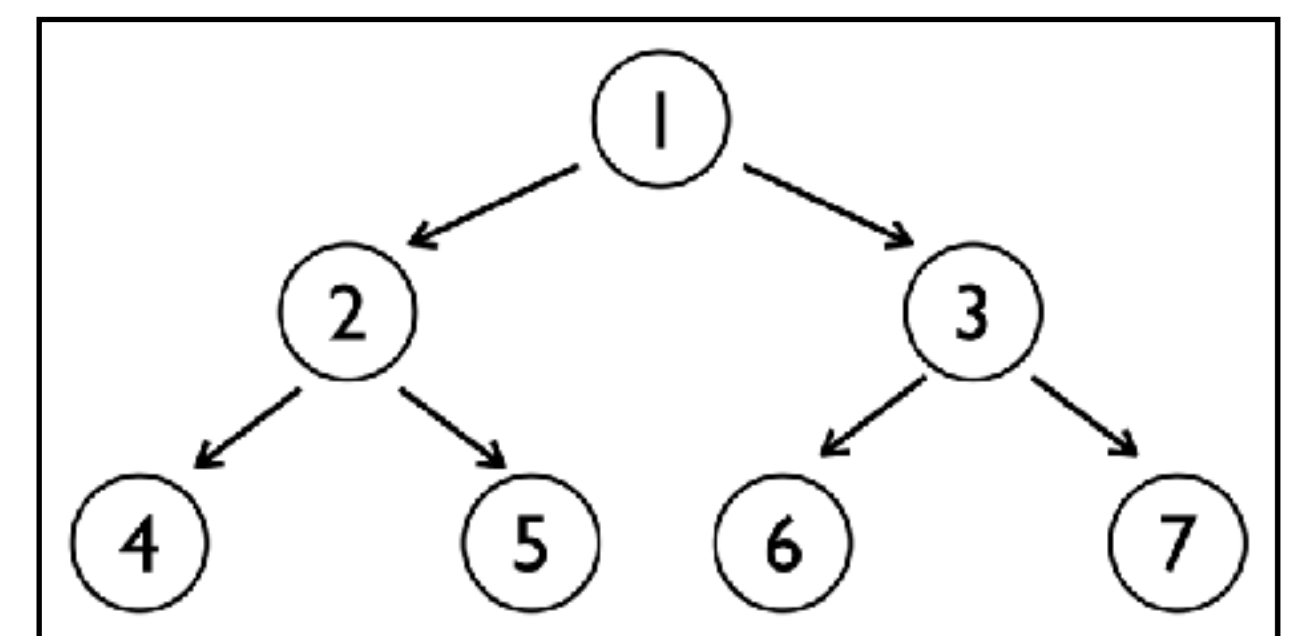
(a)



(b)



(c)

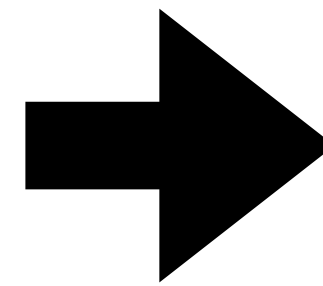
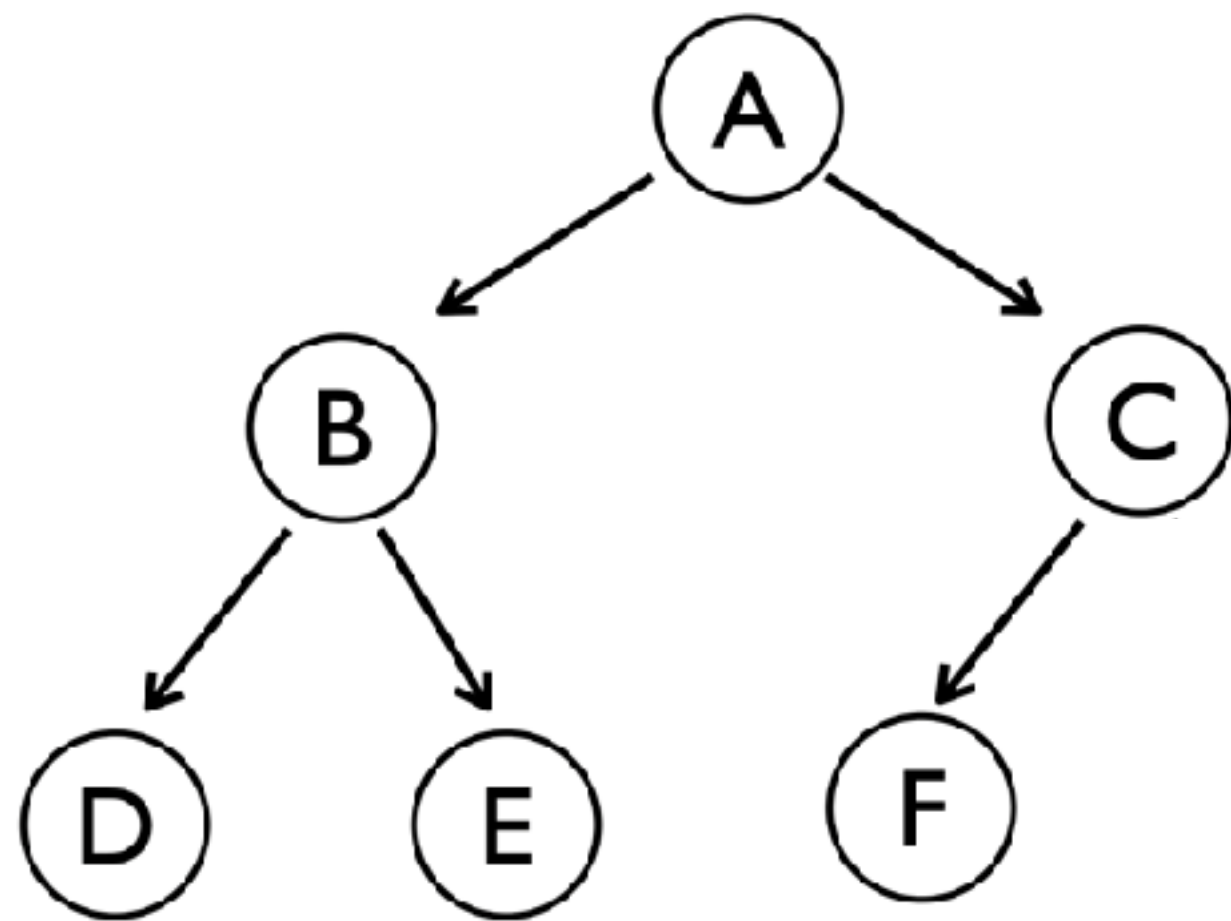


(d)

높이가 3인 완전 이진트리들

이진트리의 구현

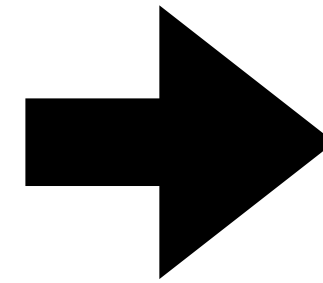
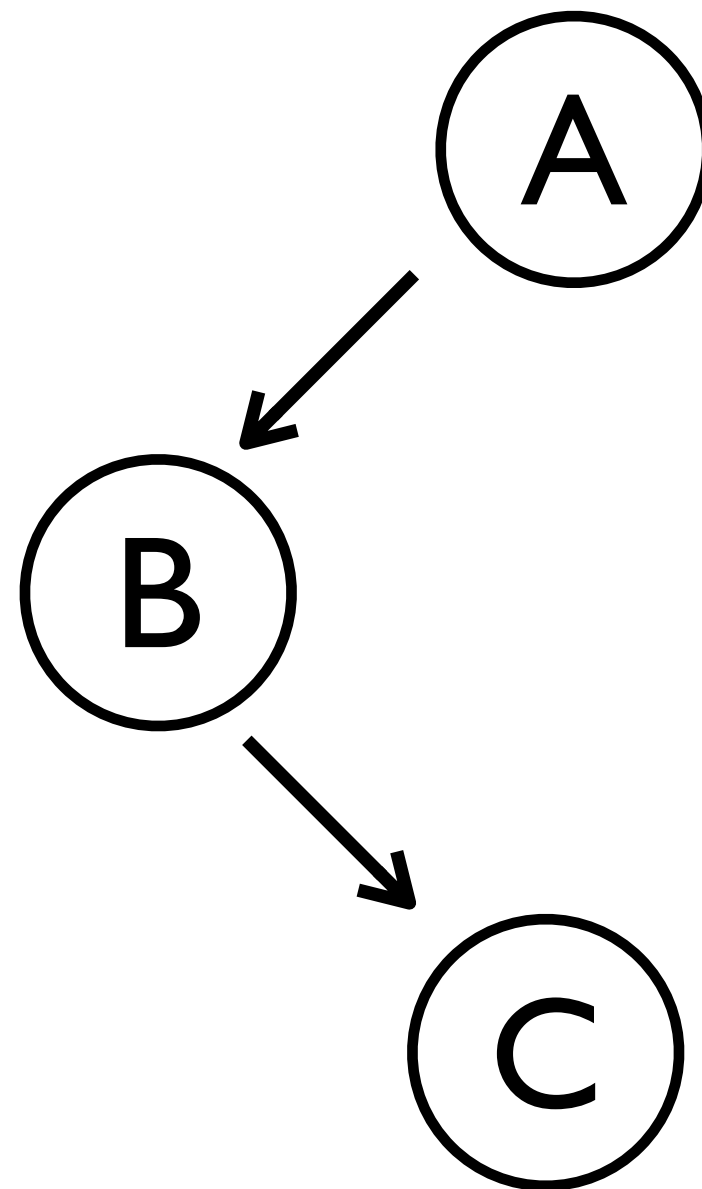
- 배열(Array) 표현법
 - 이진트리가 완전 이진트리라고 가정하고 높이가 k 이면 $2^k - 1$ 개의 연속적인 공간으로 표현



0	
1	A
2	B
3	C
4	D
5	E
6	F
7	
8	

이진트리의 구현

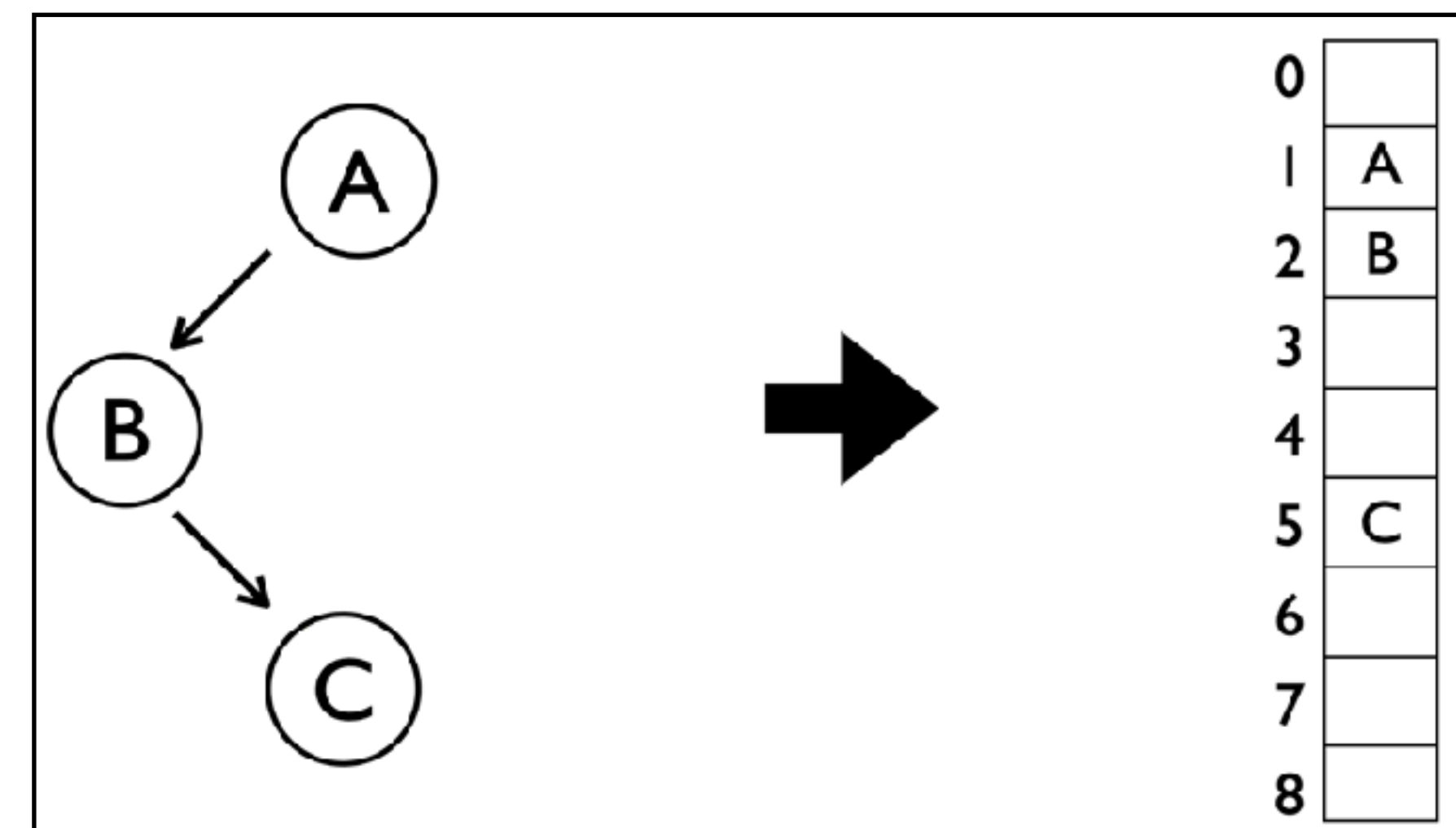
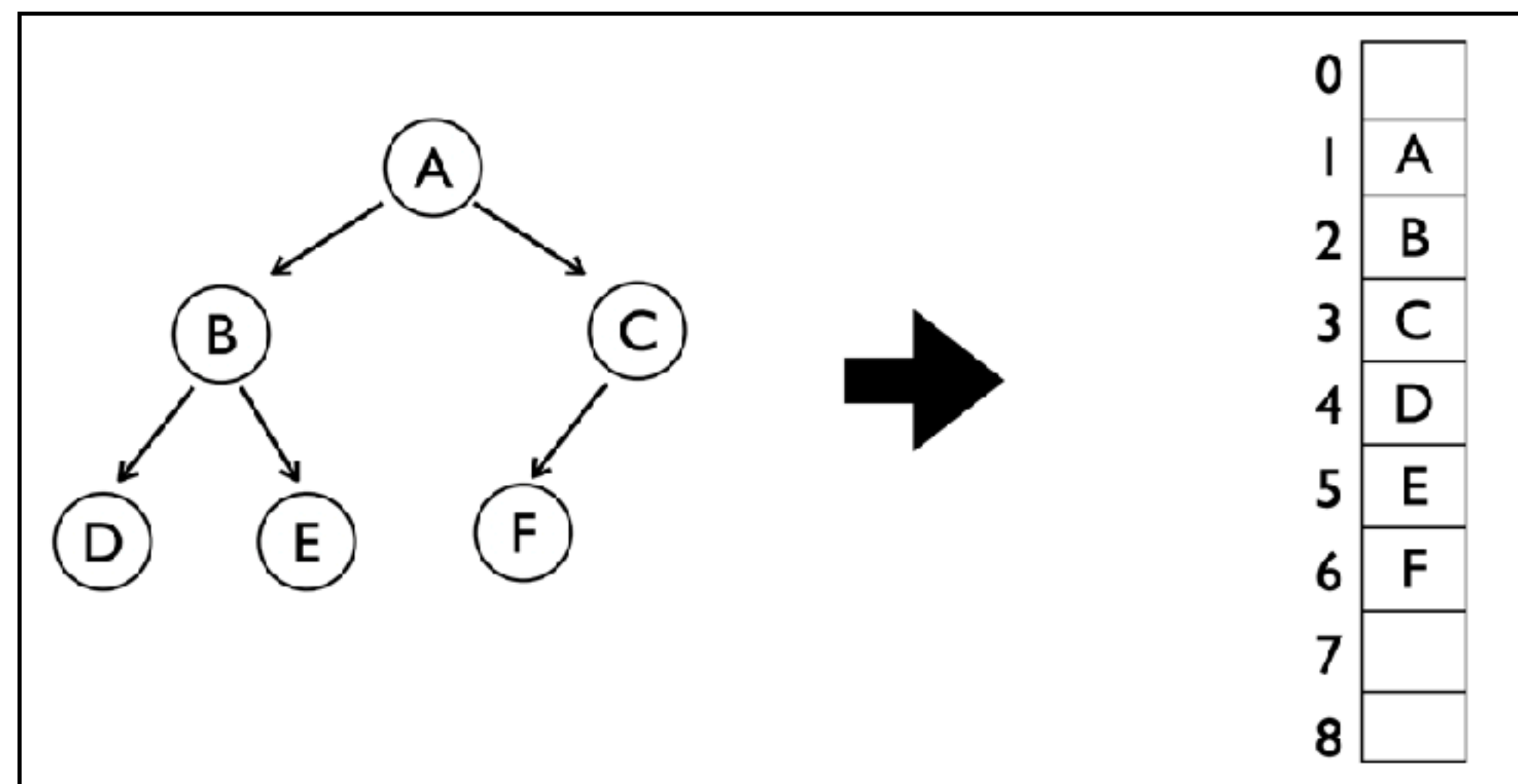
- 배열(Array) 표현법
 - 이진트리가 완전 이진트리라고 가정하고 높이가 k 이면 $2^k - 1$ 개의 연속적인 공간으로 표현
 - 완전 이진트리가 아닌 일반 이진트리도 배열로 표현할 수 있음



0	
1	A
2	B
3	
4	
5	C
6	
7	
8	

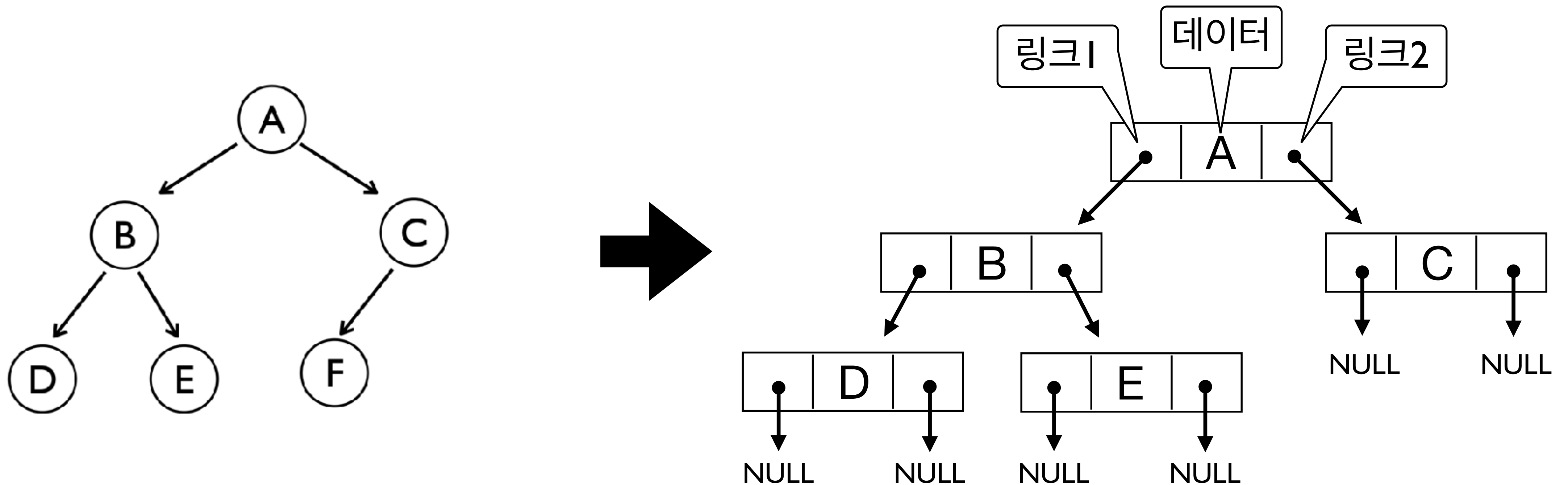
이진트리의 구현

- 배열(Array) 표현법
- 부모와 자식의 인덱스 사이에는 다음과 같은 공식이 성립함
 - 노드의 인덱스가 i 일 때 부모 노드의 인덱스 = $i/2$
 - 인덱스가 i 인 노드의 왼쪽 자식 노드의 인덱스 = $2i$
 - 인덱스가 i 인 노드의 오른쪽 자식 노드의 인덱스 = $2i + 1$



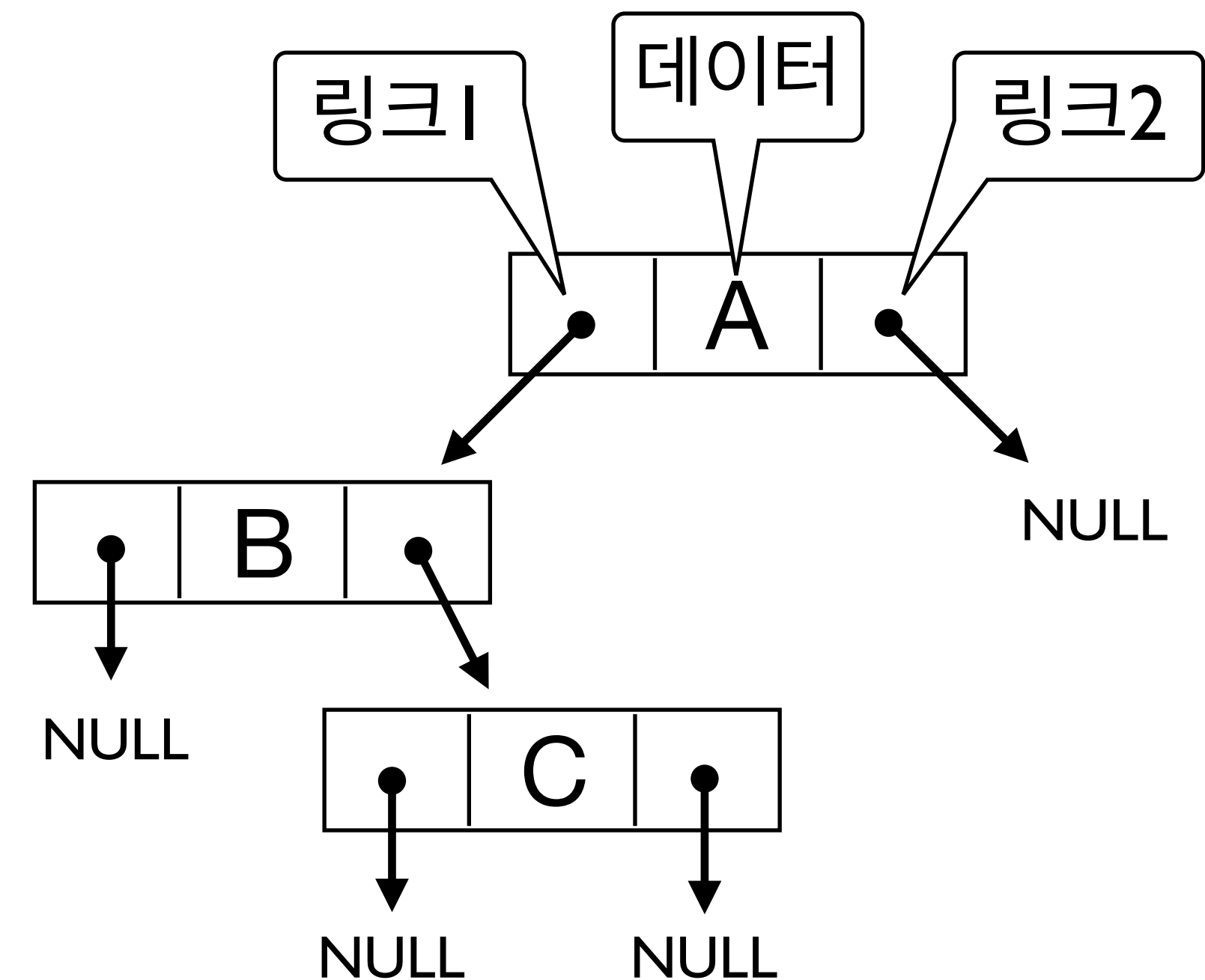
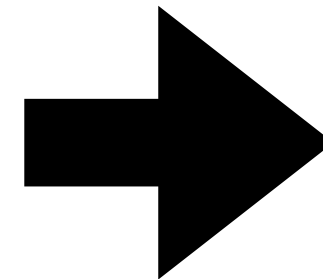
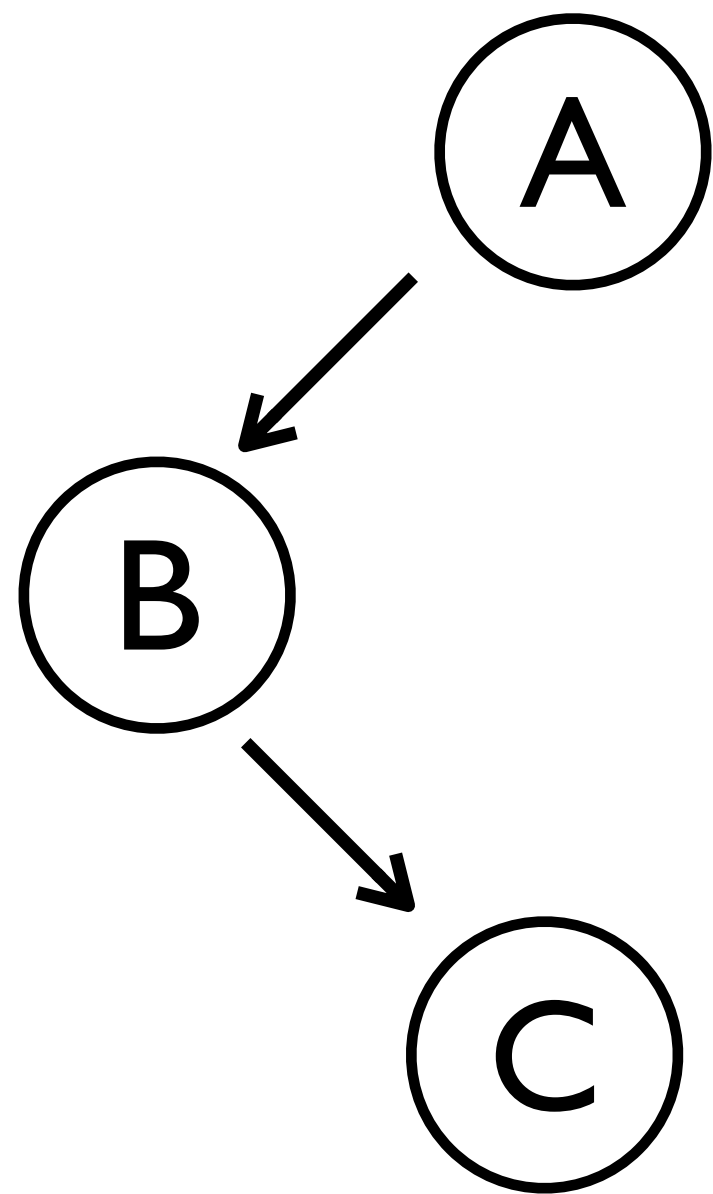
이진트리의 구현

- 링크 표현법
 - 링크 표현법은 연결 리스트와 유사함
 - 이진 트리의 노드는 데이터와 두개의 링크를 가짐



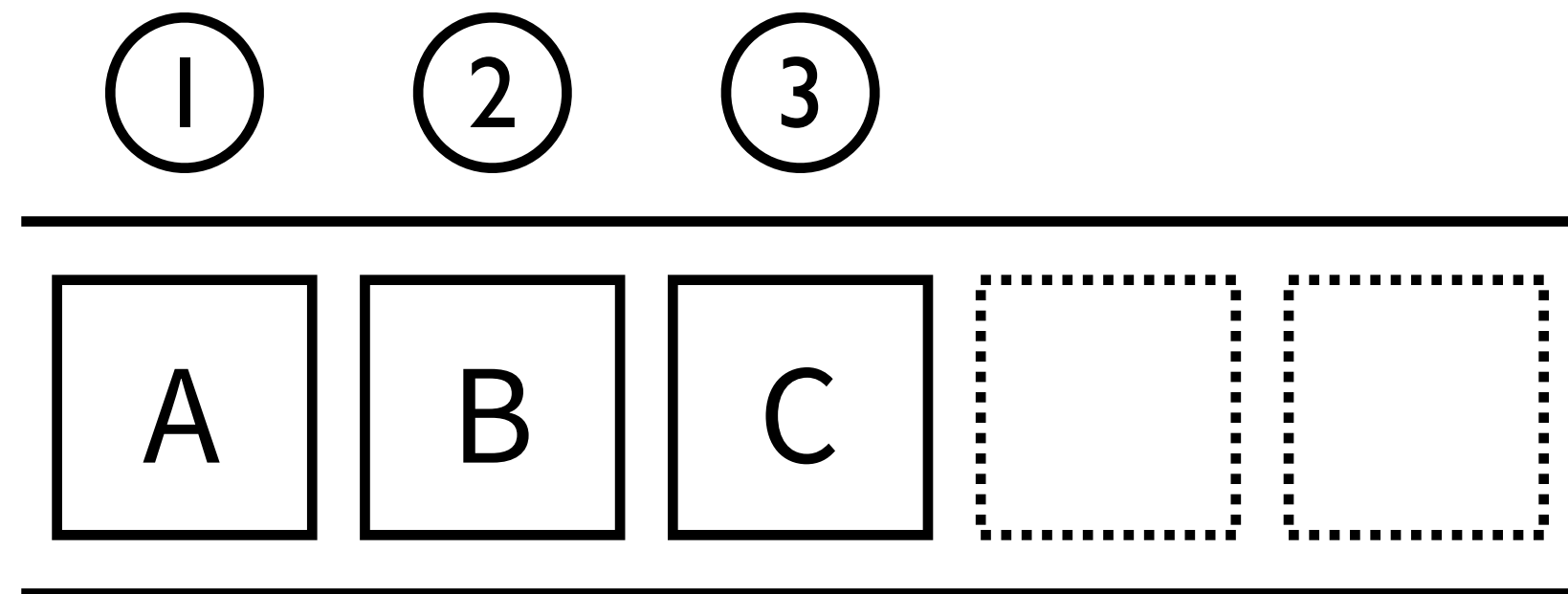
이진트리의 구현

- 링크 표현법
 - 링크 표현법은 연결 리스트와 유사함
 - 이진 트리의 노드는 데이터와 두개의 링크를 가짐

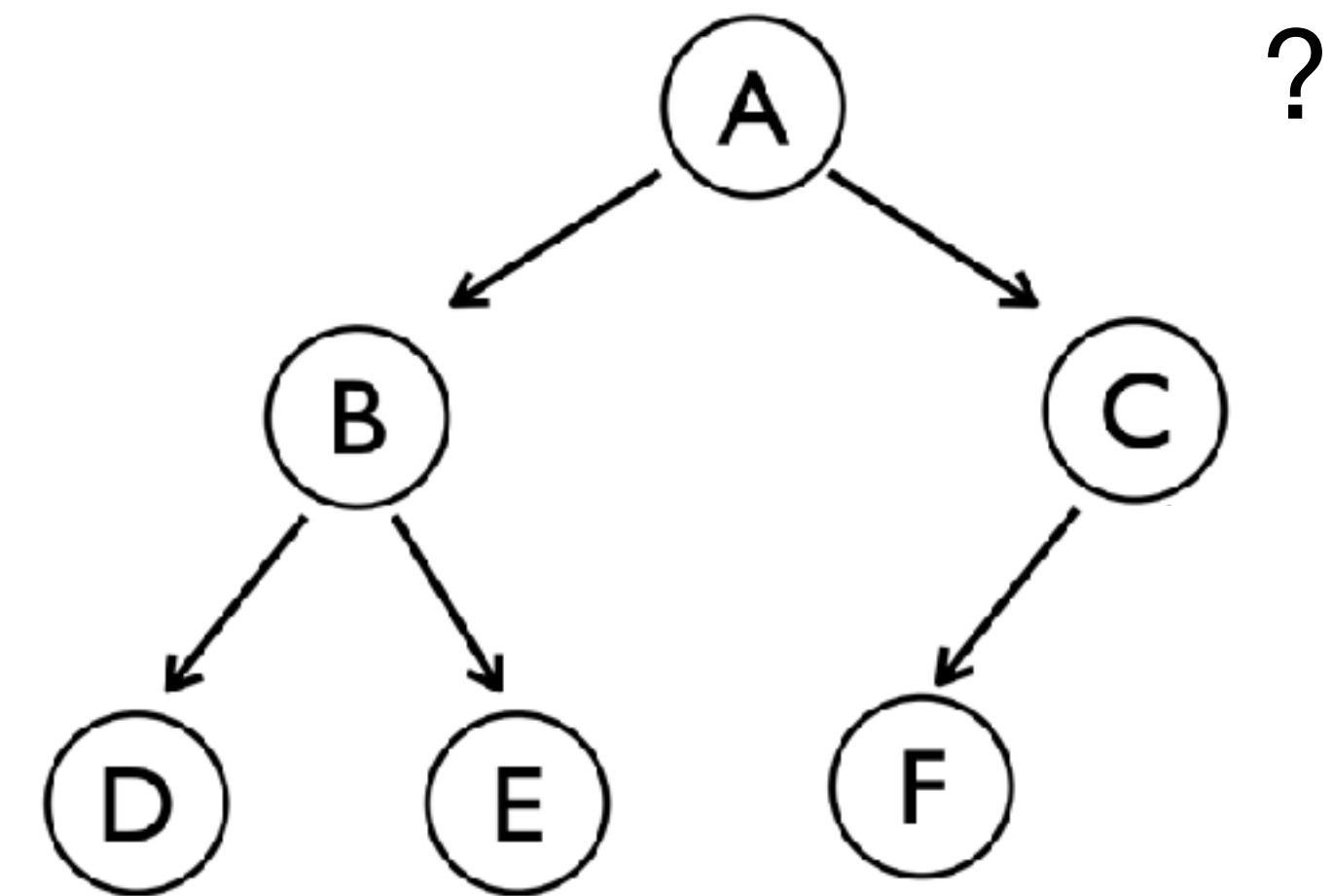


이진트리의 순회

- 트리를 순회(traversal)한다는 것은 모든 노드를 한 번씩 방문하여 가지고 있는 데이터를 목적에 맞게 처리하는 것을 의미함



큐(선형 자료구조)에서의 순회

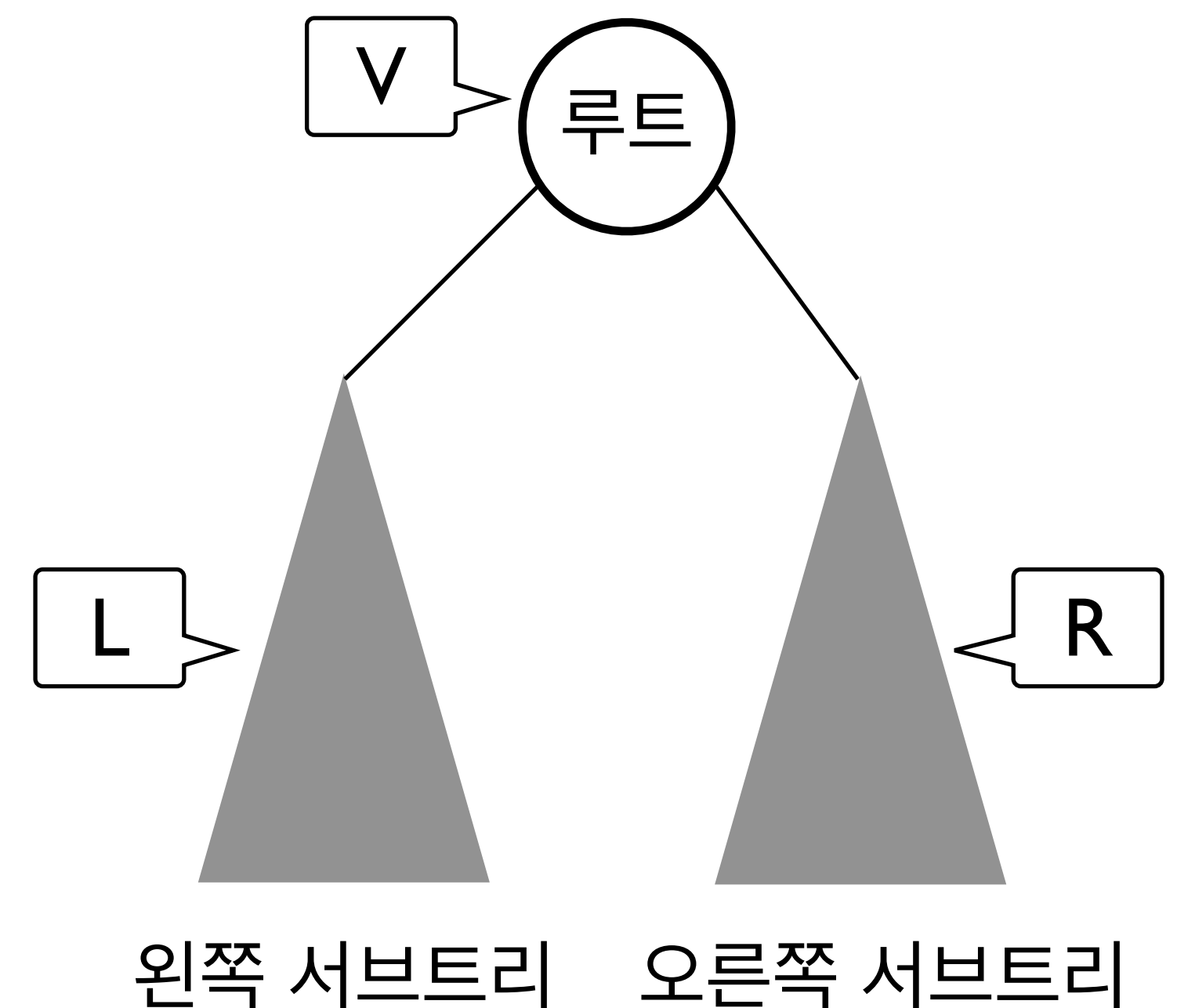


(b) 이진트리(비선형 자료구조)에서의 순회

이진트리의 순회

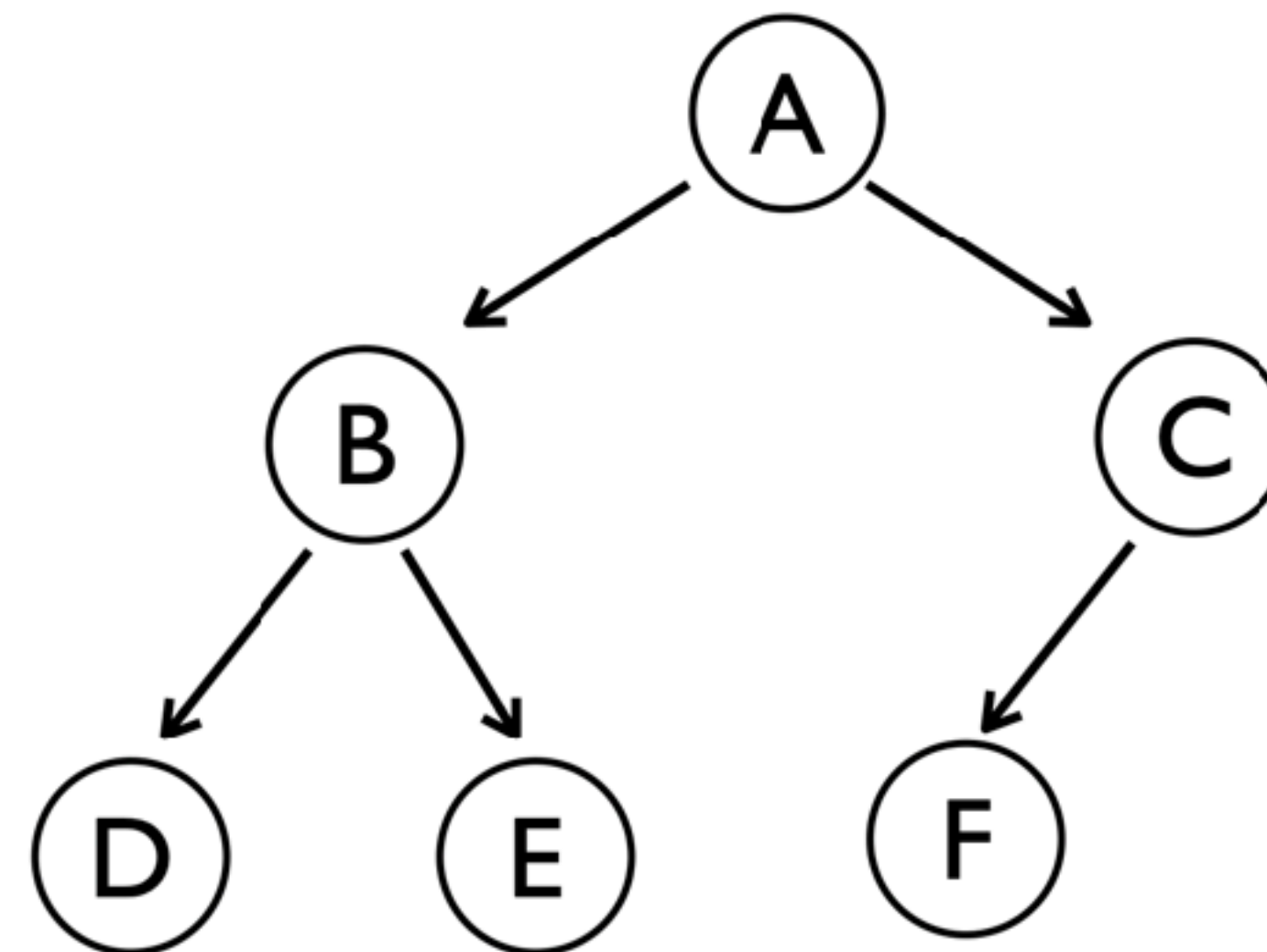
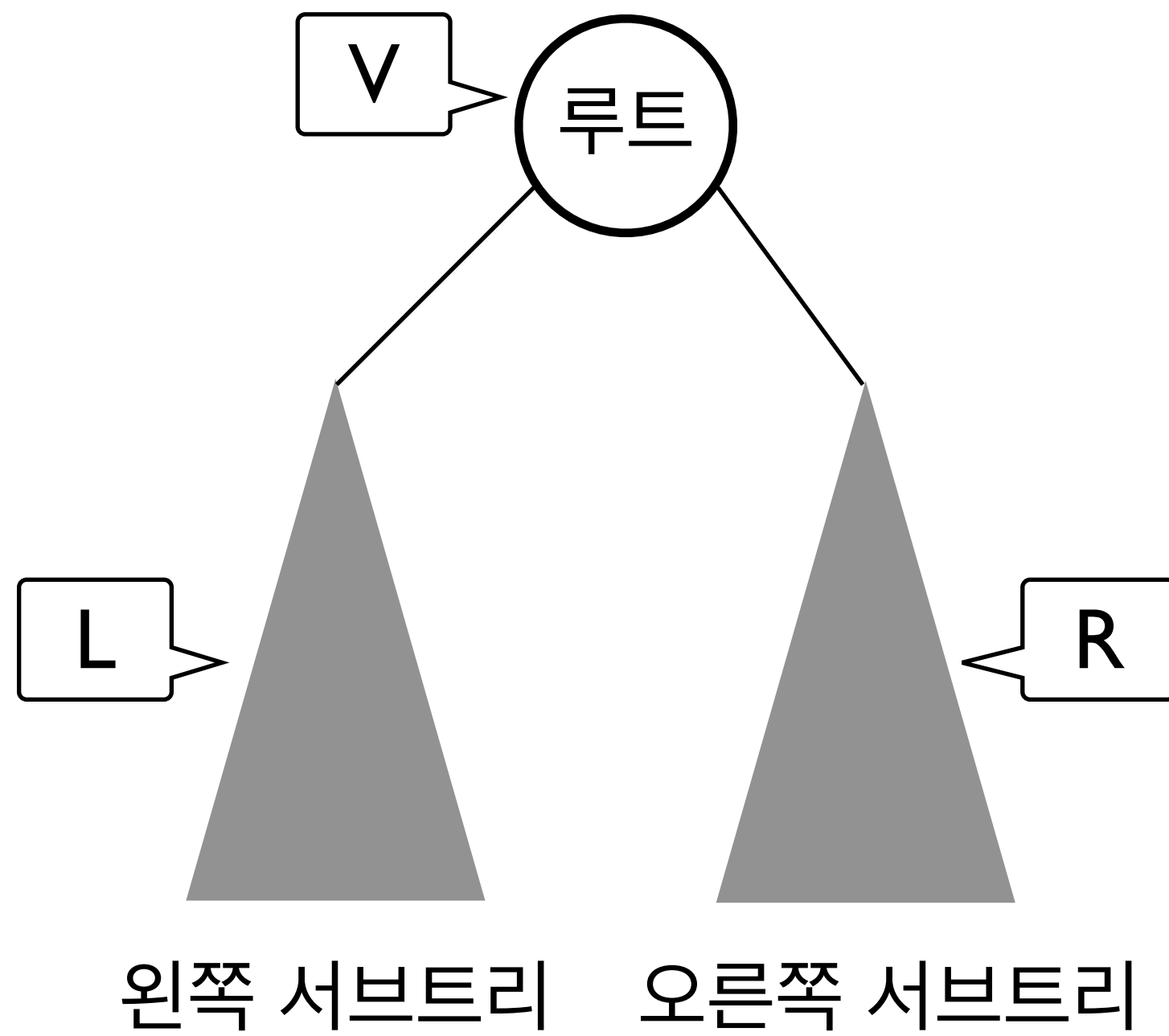
- 이진트리를 순회하는 표준적인 방법에는 전위, 중위, 후위의 3가지 방법이 있음
 - 루트와 왼쪽 서브트리, 오른쪽 서브트리를 각각 어떤 순서로 방문하느냐에 따라 구분함
 - 만약 루트를 방문하는 작업을 V, 왼쪽 서브트리 방문을 L, 오른쪽 서브트리 방문을 R이라고 했을 때

- 전위순회 (preorder traversal) : VLR 순으로 처리
- 중위순회 (inorder traversal) : LVR 순으로 처리
- 후위순회 (postorder traversal) : LRV 순으로 처리



이진트리의 순회

- 전위순회 (preorder traversal) : VLR 순으로 처리

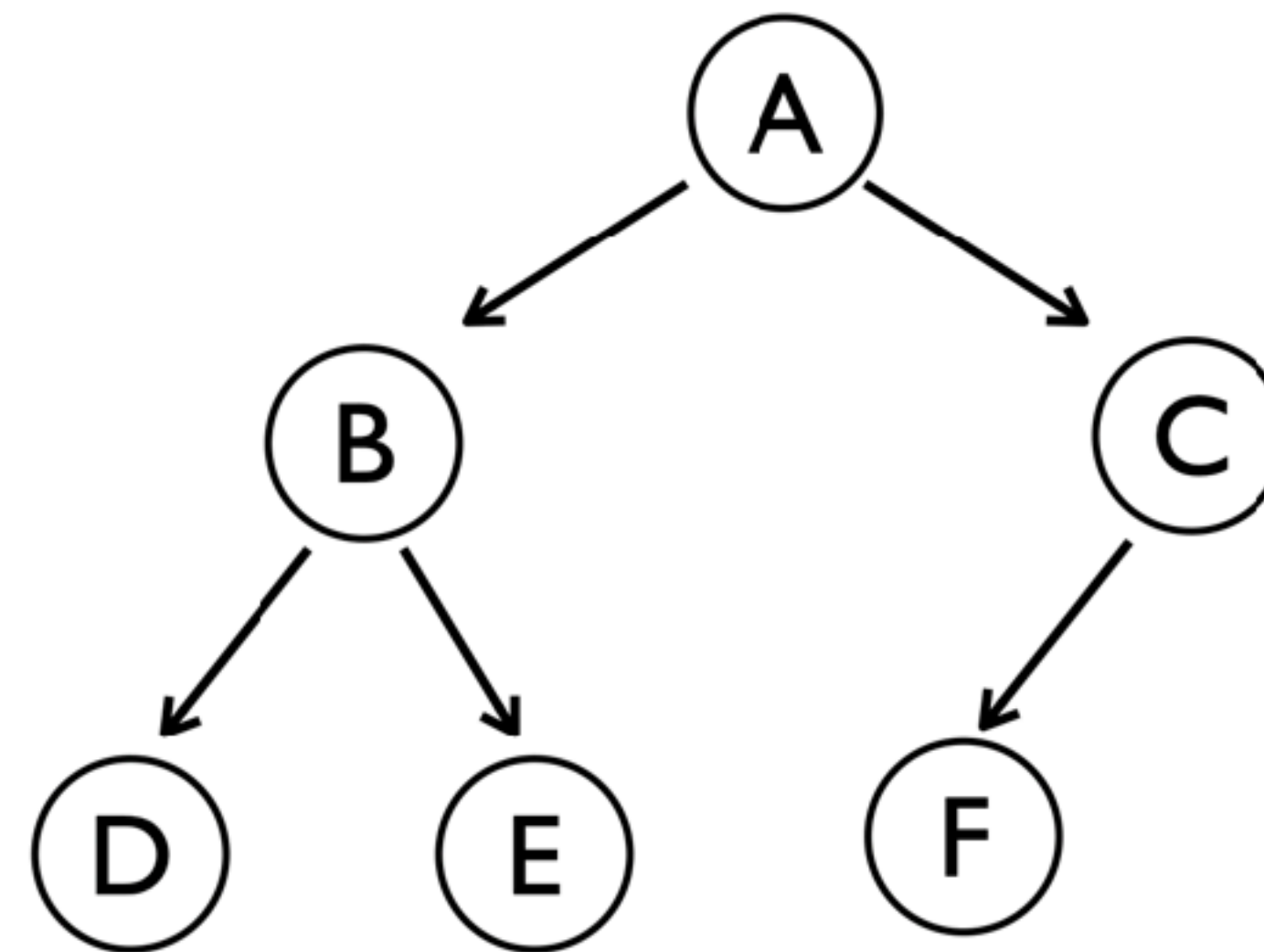


이진트리의 순회

- 전위순회 (preorder traversal) : VLR 순으로 처리

```
procedure preorderTraversal(node)
  if node is not null then
    process(node)
    preorderTraversal(node.left)
    preorderTraversal(node.right)
  end if
end procedure
```

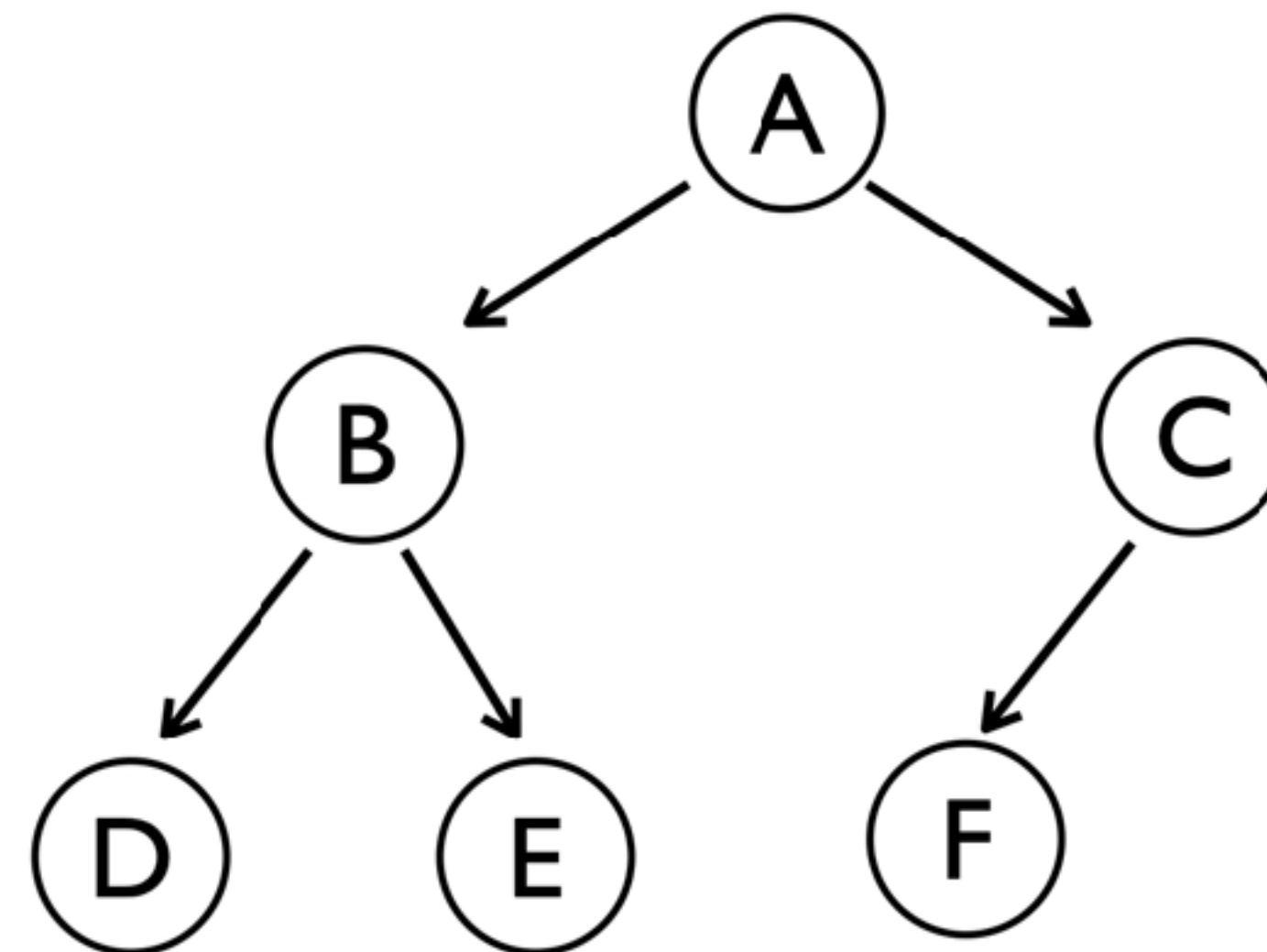
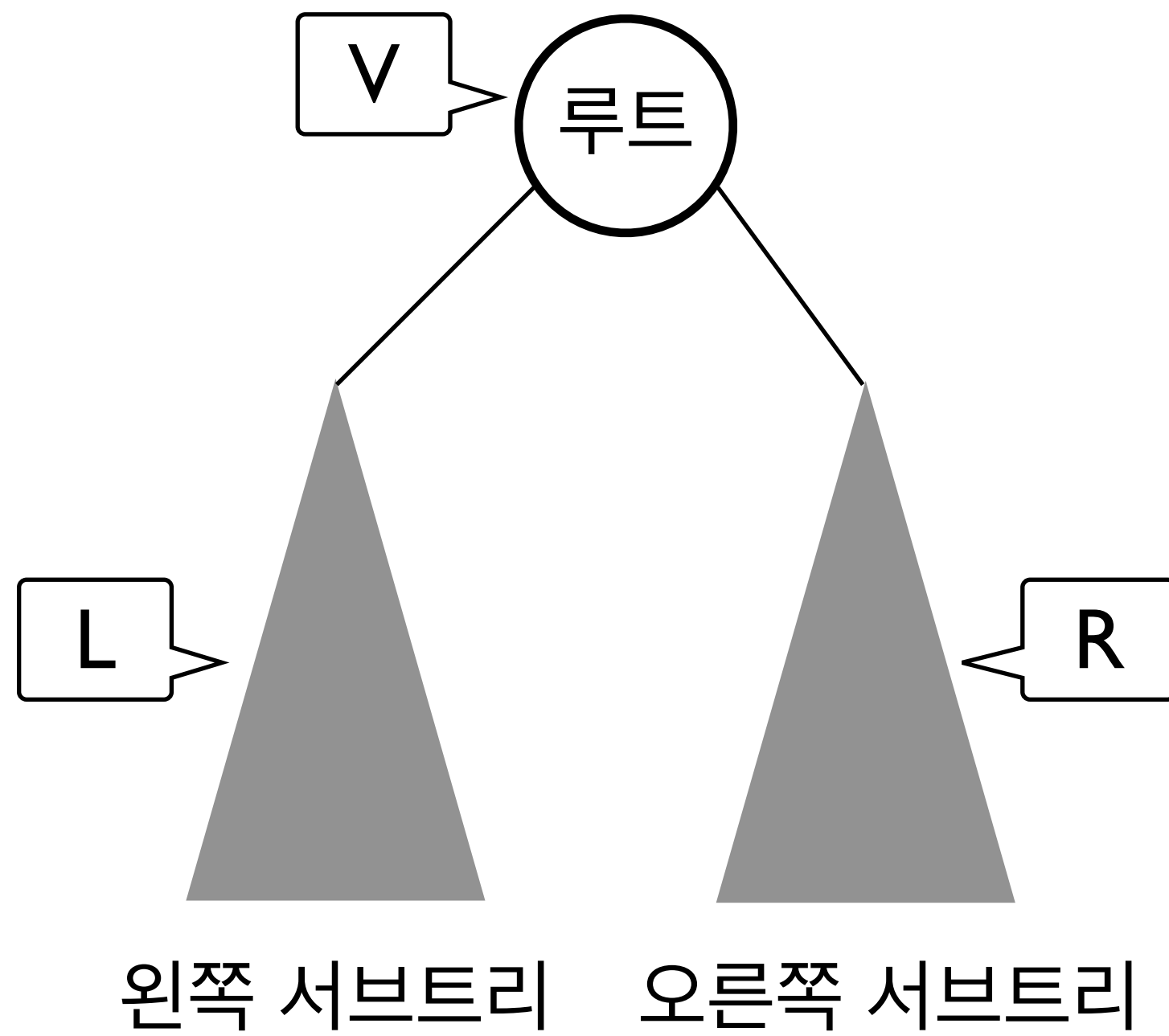
슈도 코드



A -> B -> D -> E -> C -> F

이진트리의 순회

- 중위순회 (inorder traversal) : LVR 순으로 처리

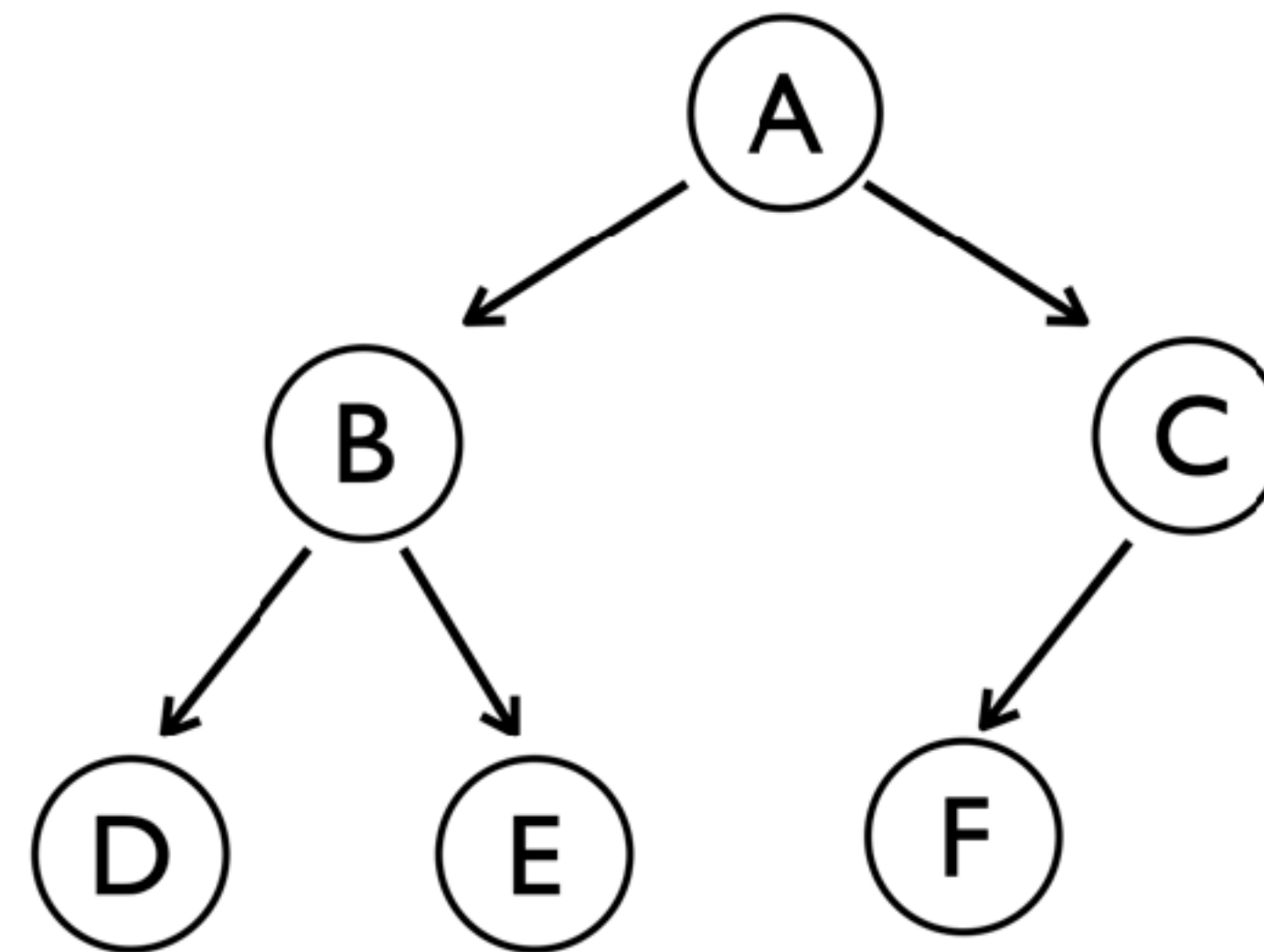


이진트리의 순회

- 중위순회 (inorder traversal) : LVR 순으로 처리

```
procedure inorderTraversal(node)
  if node is not null then
    inorderTraversal(node.left)
    process(node)
    inorderTraversal(node.right)
  end if
end procedure
```

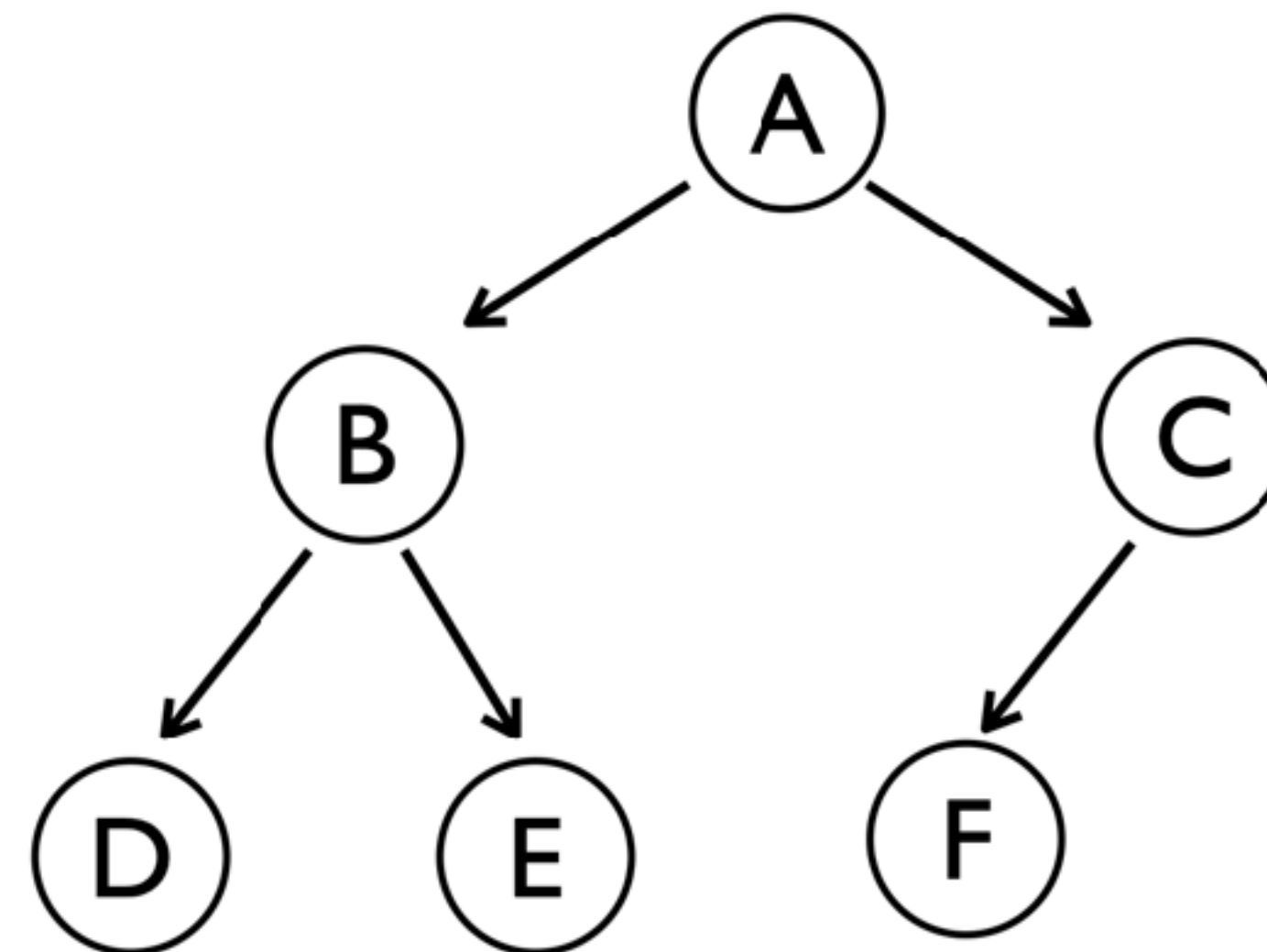
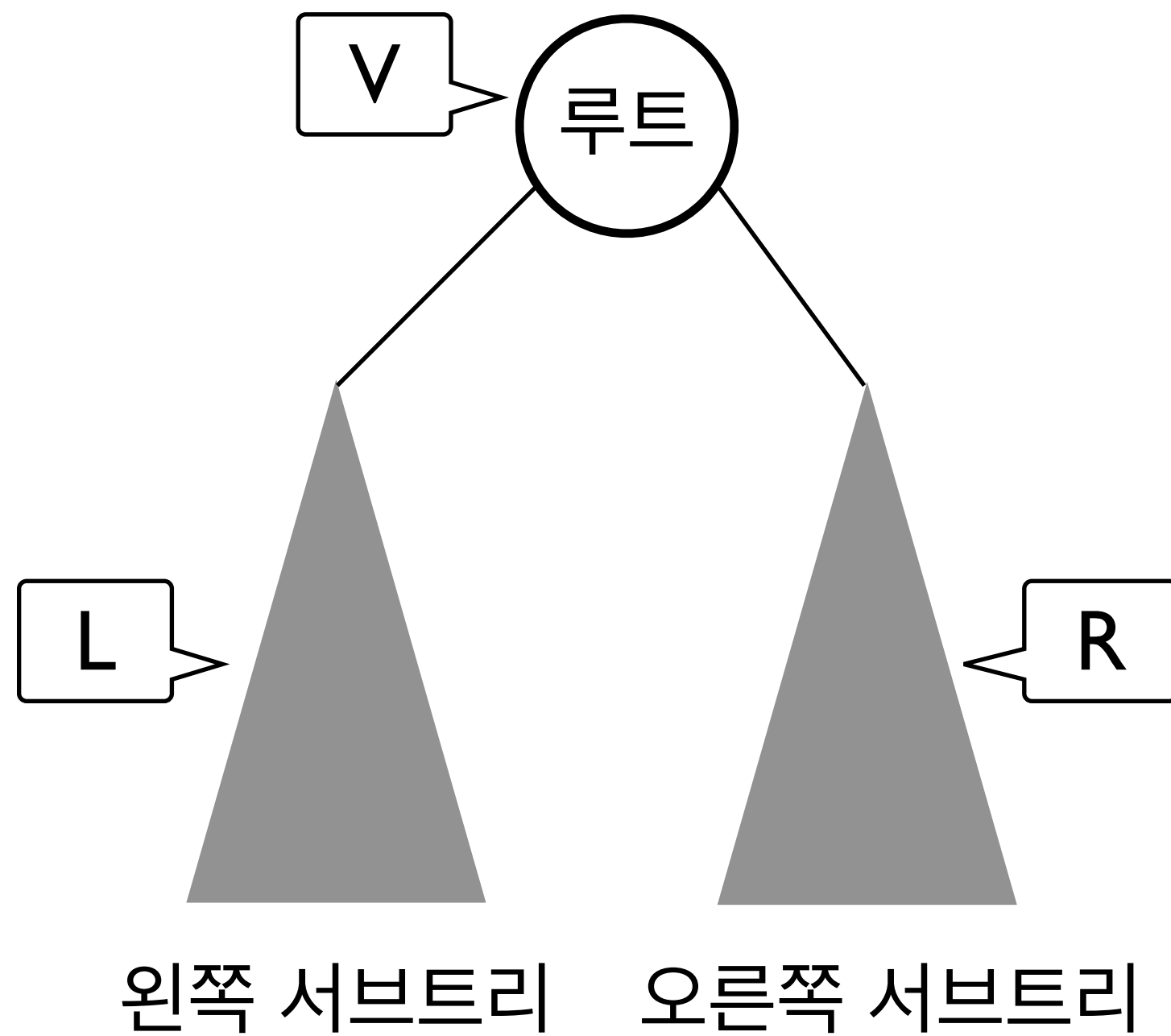
슈도 코드



D -> B -> E -> A -> F -> C

이진트리의 순회

- 후위순회 (postorder traversal) : LRV 순으로 처리

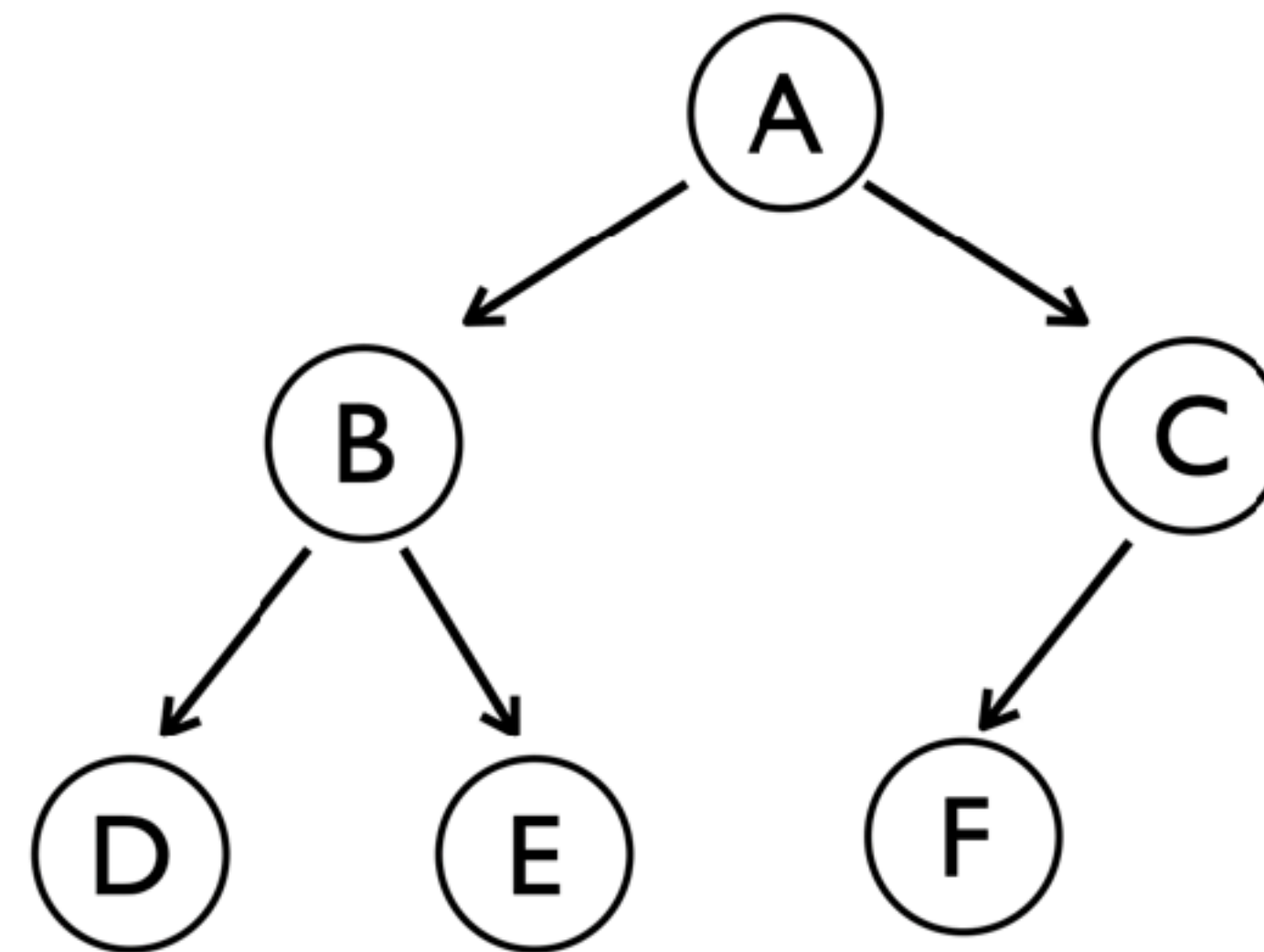


이진트리의 순회

- 후위순회 (postorder traversal) : LRV 순으로 처리

```
procedure postorderTraversal(node)
  if node is not null then
    postorderTraversal(node.left)
    postorderTraversal(node.right)
    process(node)
  end if
end procedure
```

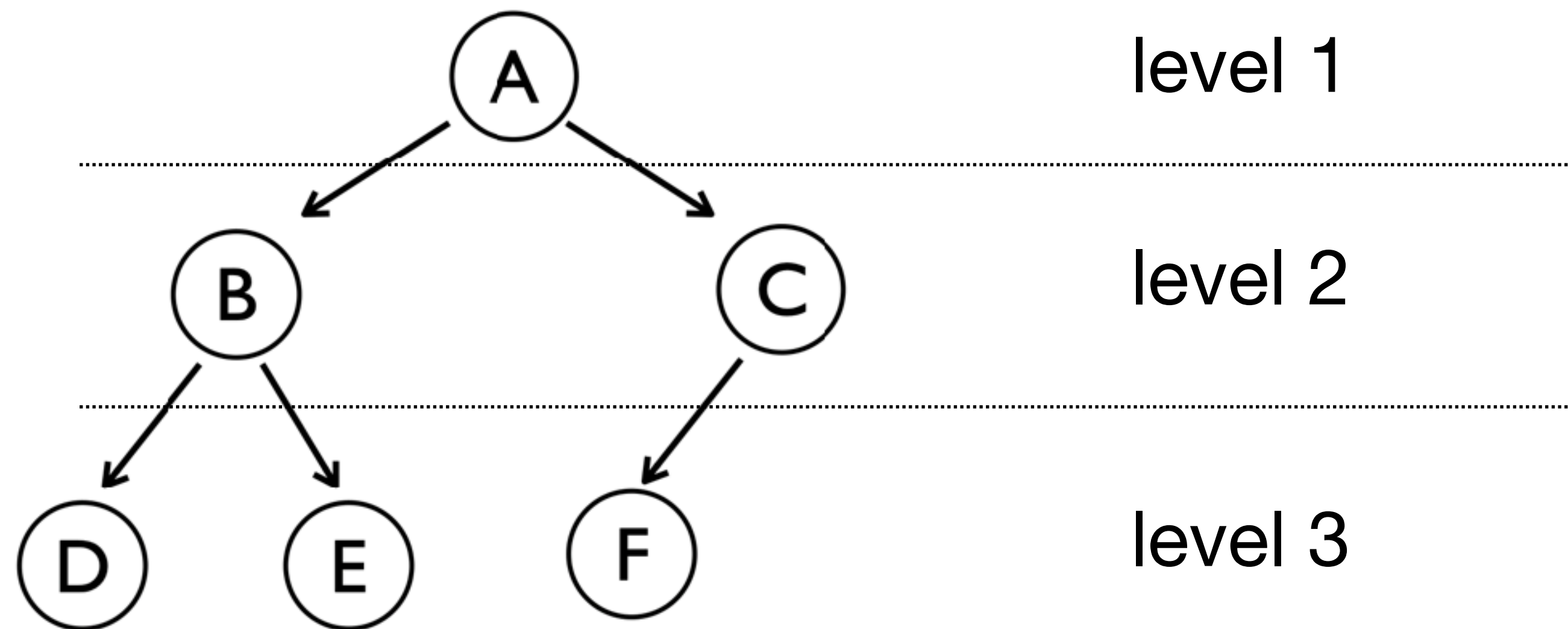
슈도 코드



D -> E -> B -> F -> C -> A

이진트리의 순회

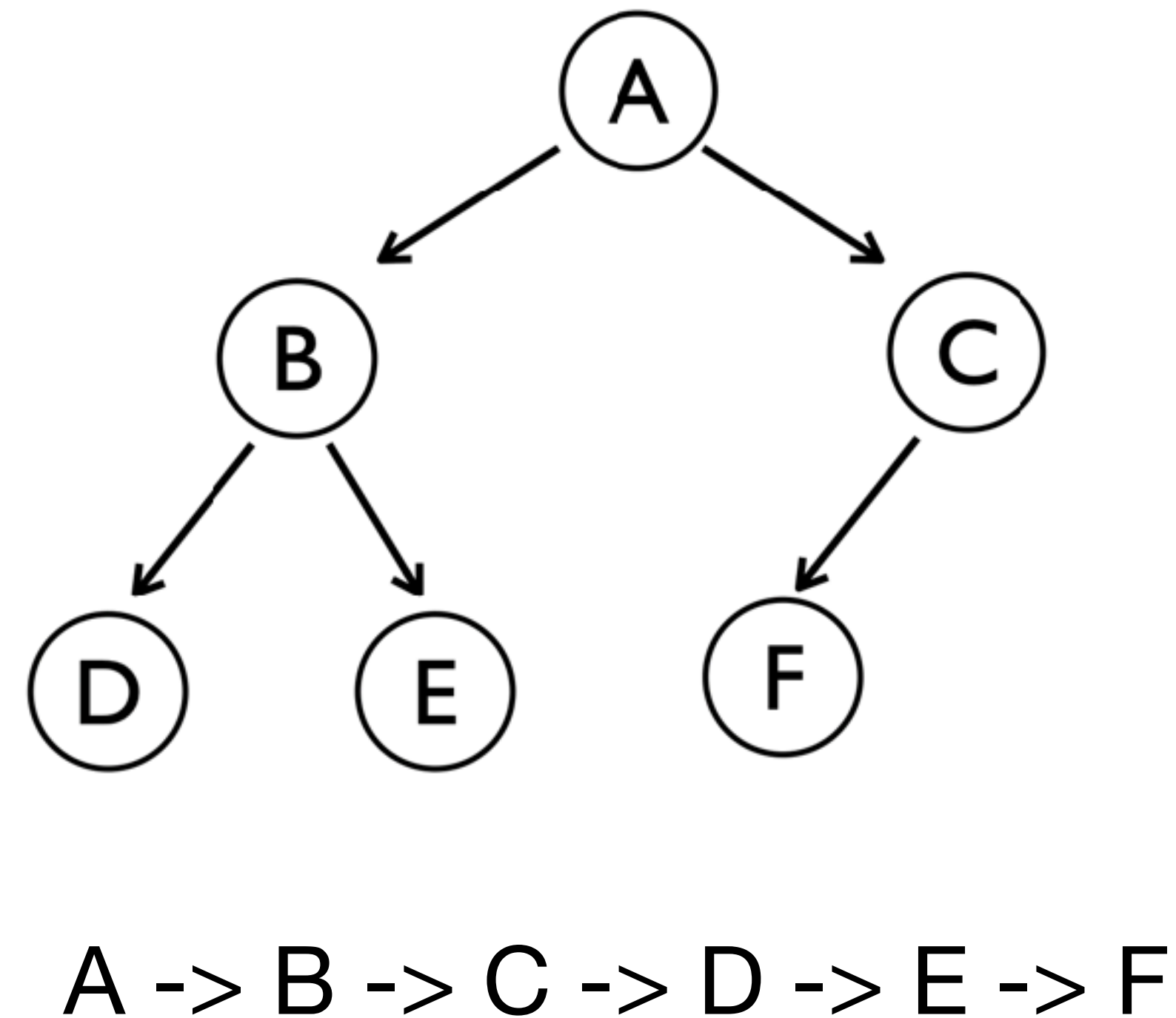
- 레벨 순회 (level order)는 각 노드를 레벨 순으로 순회하는 방법이다.
 - 동일한 레벨의 경우 좌에서 우로 방문한다.



이진트리의 순회

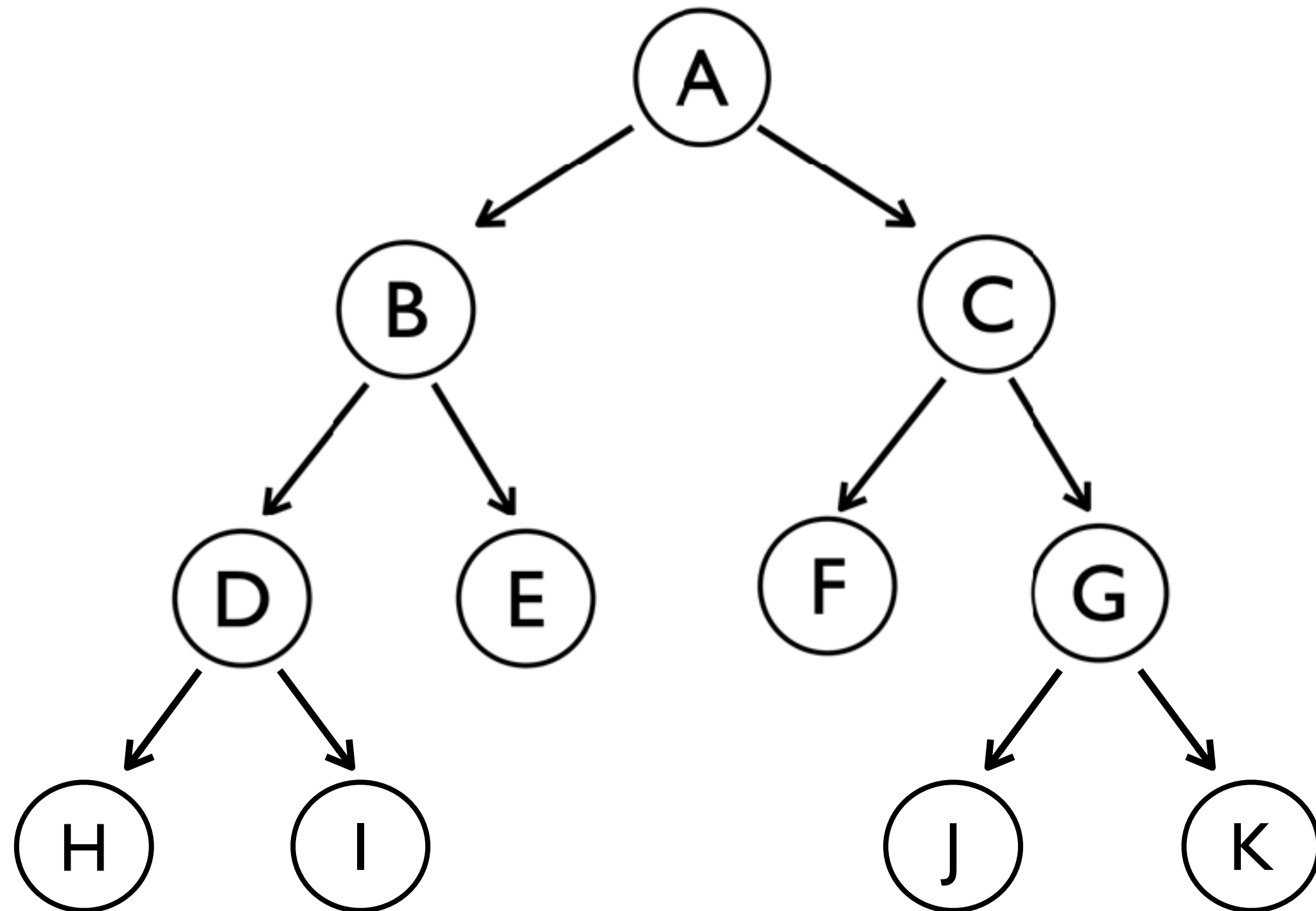
- 레벨 순회 (level order)는 각 노드를 레벨 순으로 순회하는 방법임
- 동일한 레벨의 경우 좌에서 우로 방문함

```
procedure levelOrderTraversal(root)
  queue ← createQueue()
  enqueue(queue, root)
  while isEmpty(queue) ≠ true do
    node ← dequeue(queue);
    if (node ≠ null) then
      process(node)
      enqueue(queue, node.left)
      enqueue(queue, node.right)
    end if
  end while
end procedure
```



이진트리의 순회

- Example: 아래 이진트리에서 전위순회, 중위순회, 후위순회 하였을 때 처리한 노드들이 처리된 순서는?



- 전위순회 (VLR):

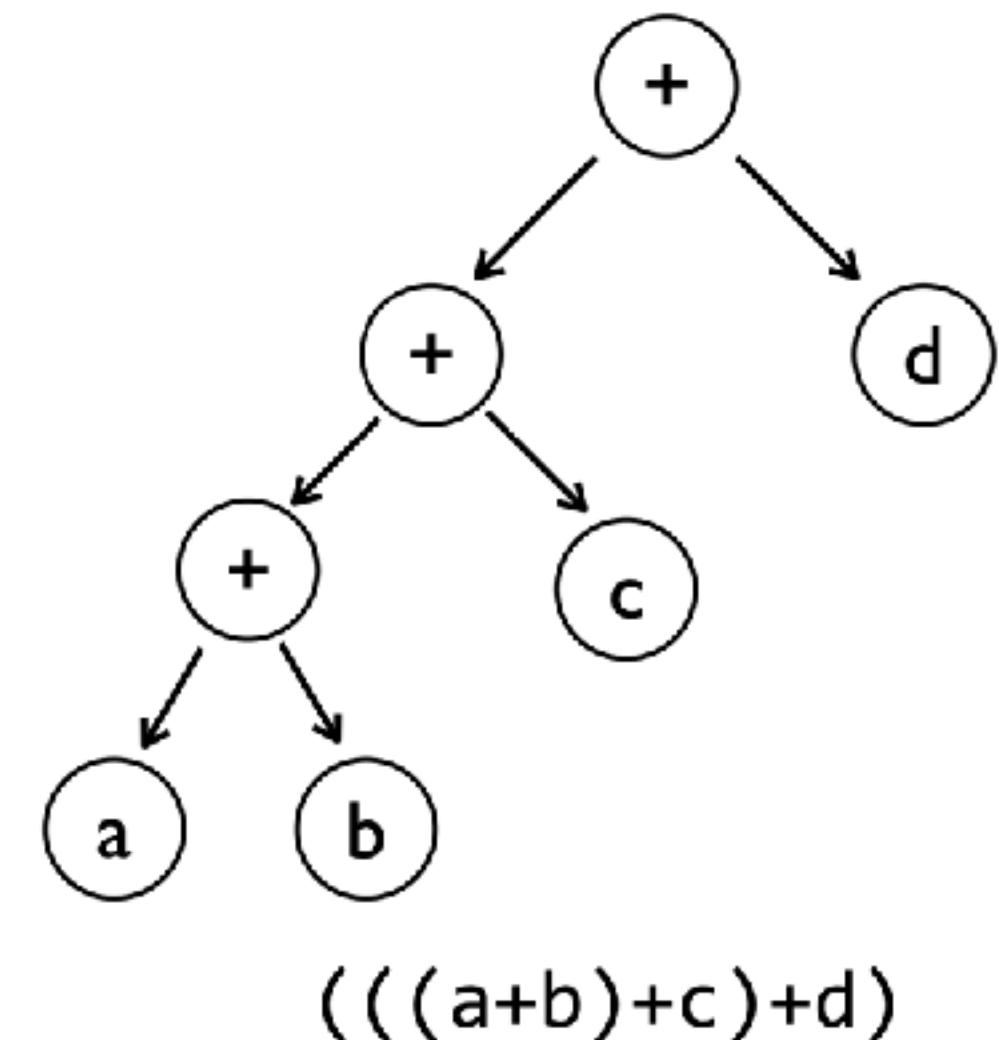
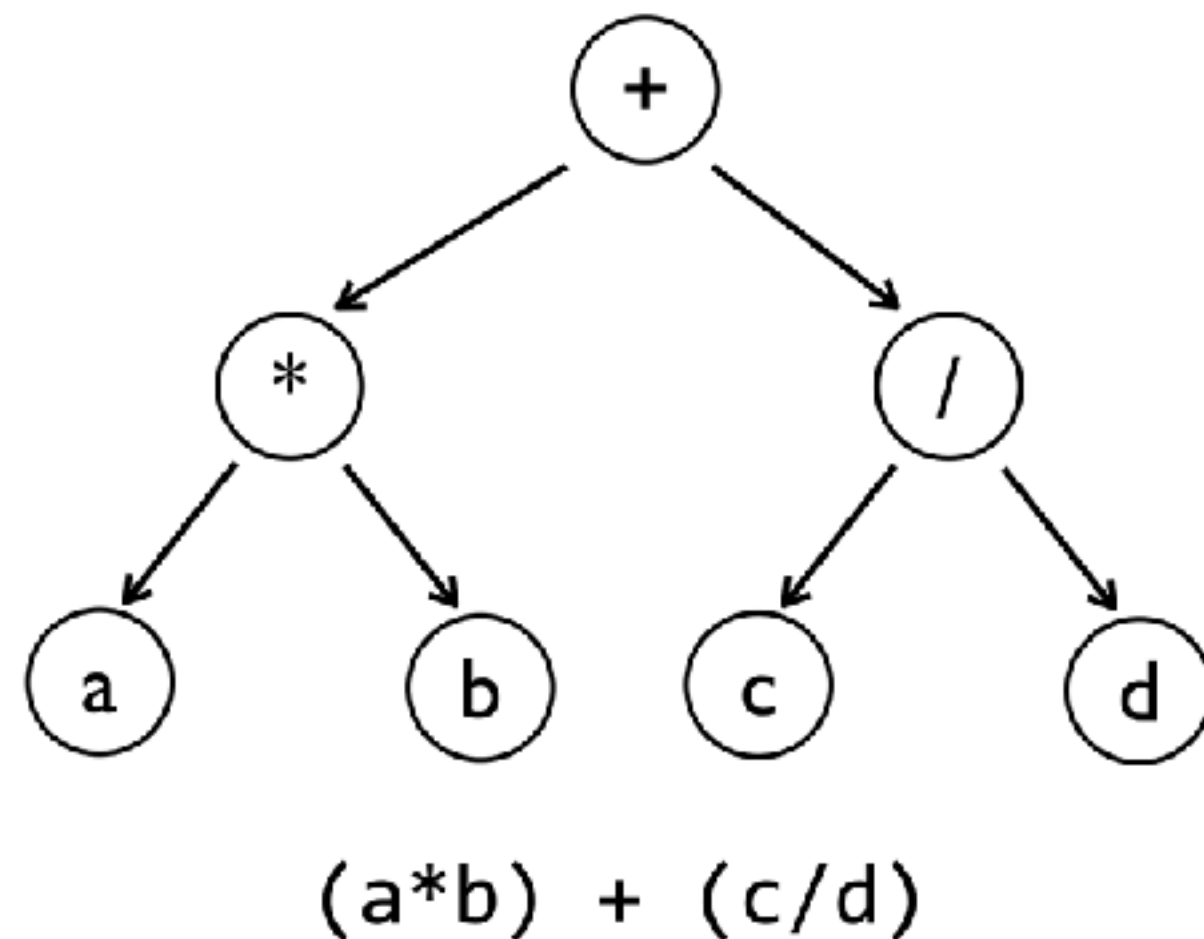
- 중위순회 (LVR):

- 후위순회 (LRV):

- 레벨순회:

이진트리의 응용: 수식 표현 (Expression Tree)

- 수식을 표현한 이진 트리(expression tree)에서 각 노드는 연산자 (+, -, *, /) 또는 피연산자임
 - 모든 리프 노드들은 피연산자임
 - 루트 노드와 모든 내부 노드들은 연산자임
 - 서브 트리들은 식의 부분(subexpression)들을 표현함

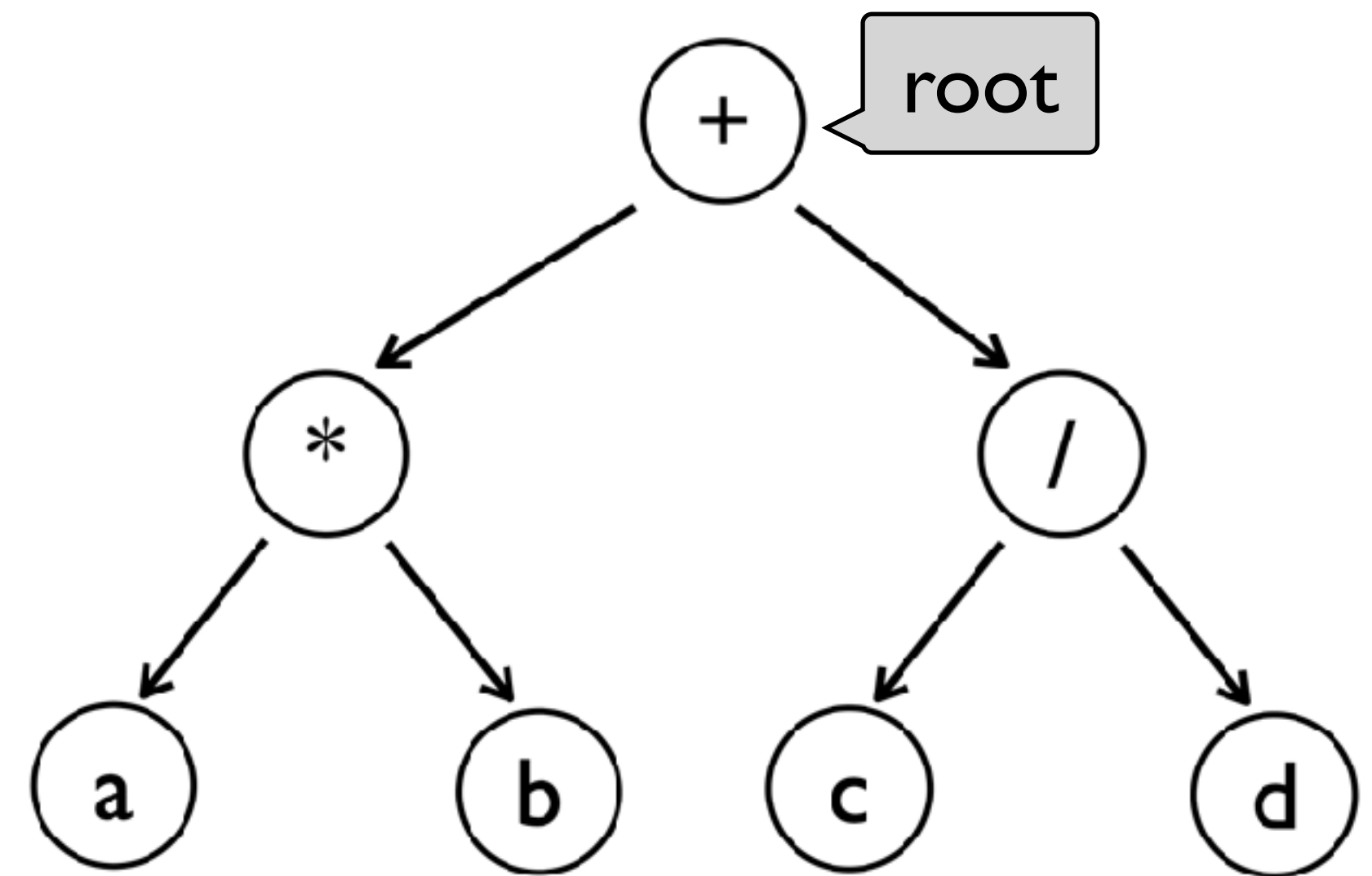


연습 문제

- 순회로 이진 트리가 표현하고 있는 수식 출력하기

```
procedure printExpression(root)
```

```
end procedure
```



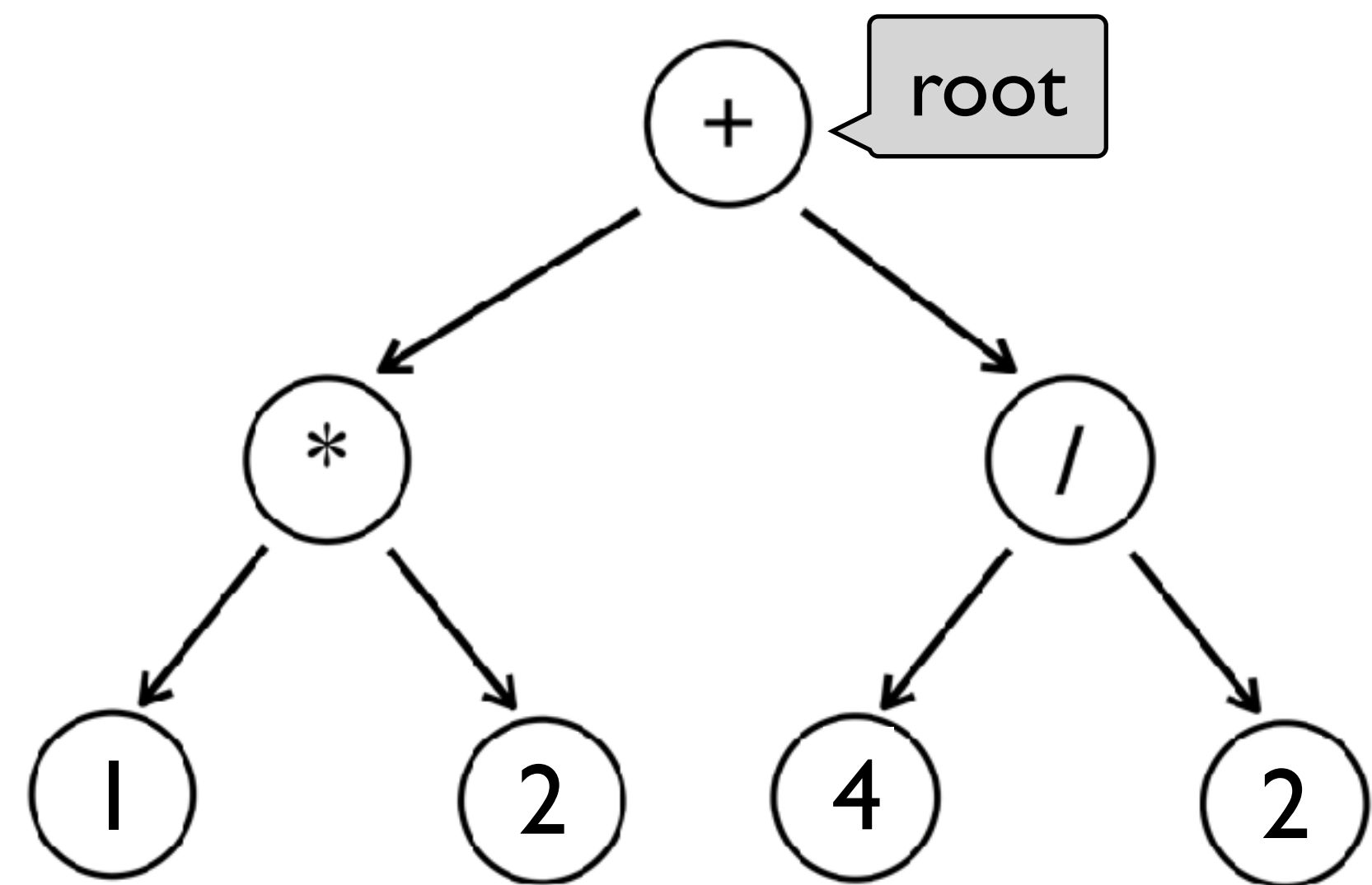
$((a * b) + (c / d))$

- 이진 트리가 표현하고 있는 수식의 값 계산하기

procedure eval(root)

end procedure

Example



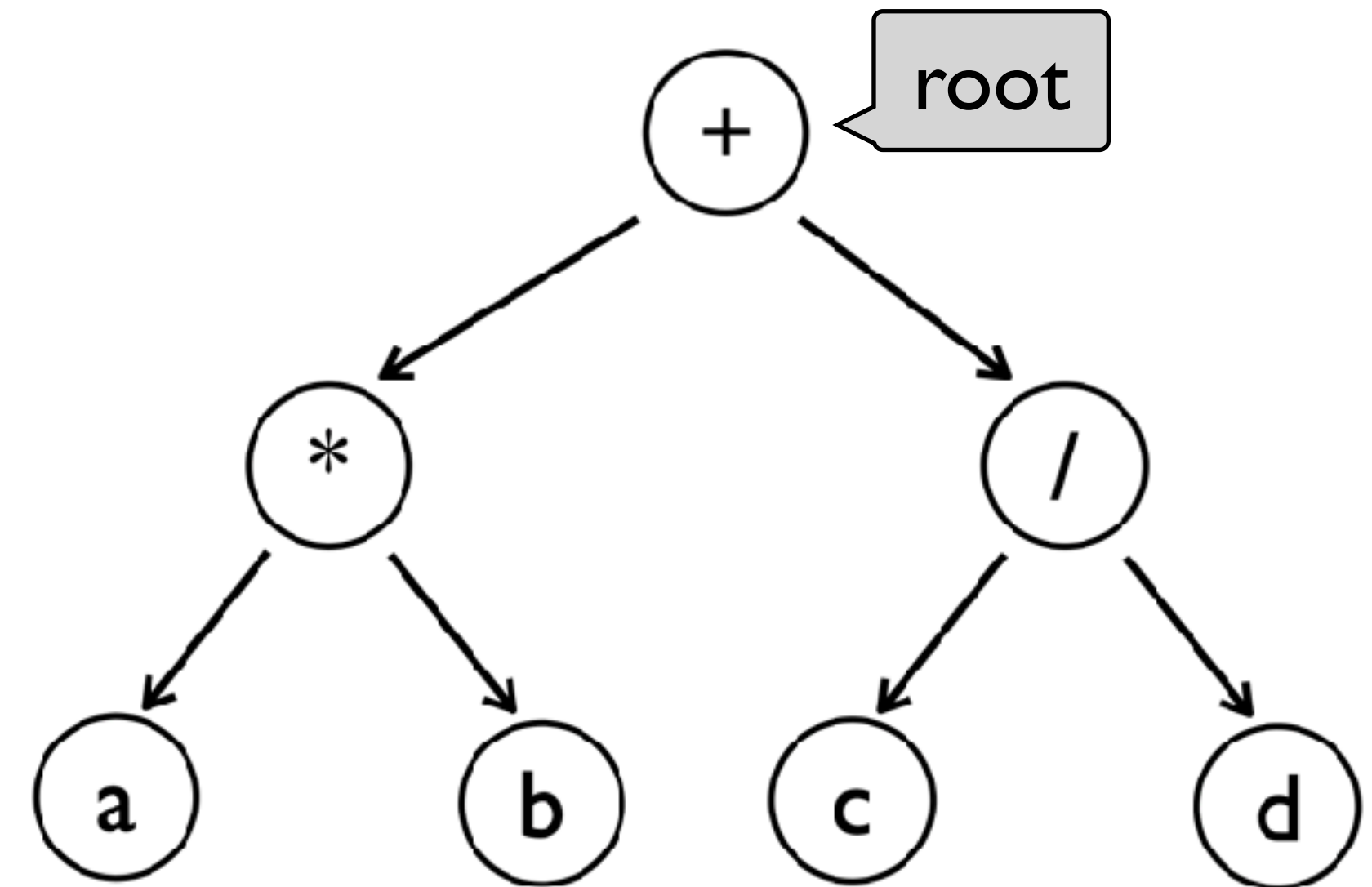
$((1 * 2) + (4 / 2))$

연습 문제

- 이진 트리의 높이를 구하는 알고리즘을 기술하시오

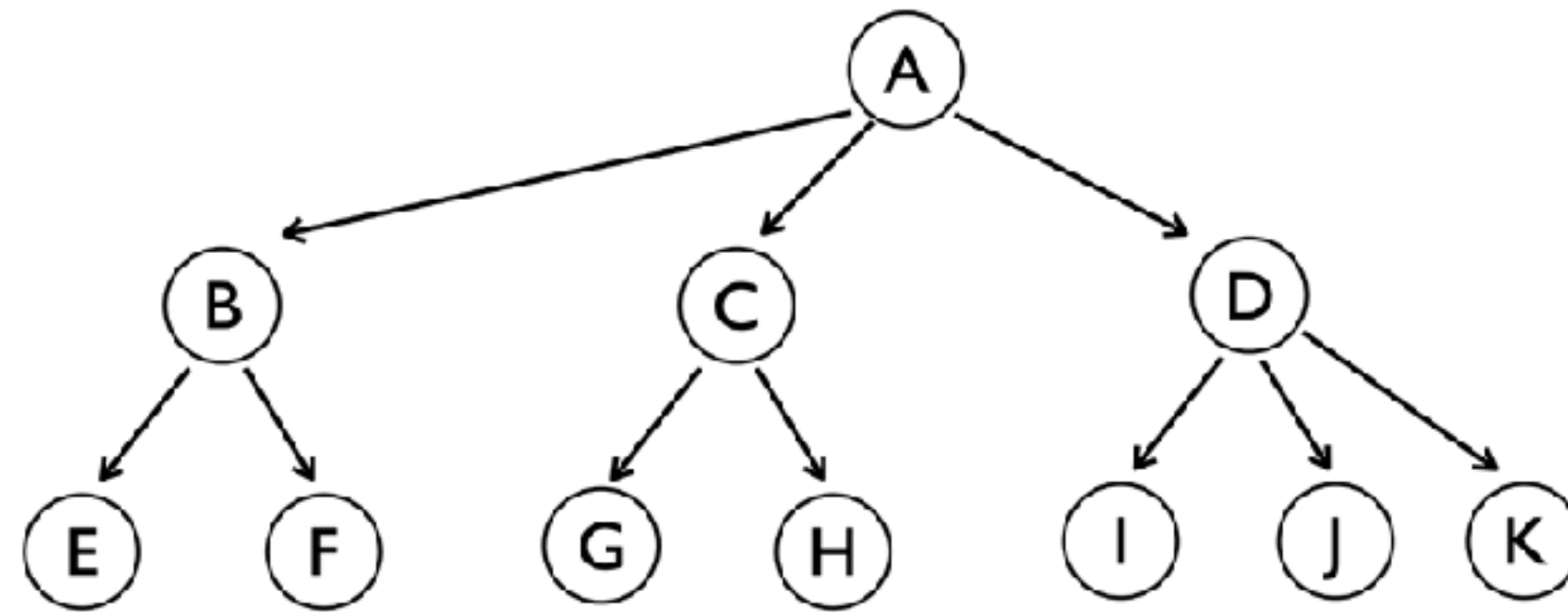
```
procedure height(root)
```

```
end procedure
```

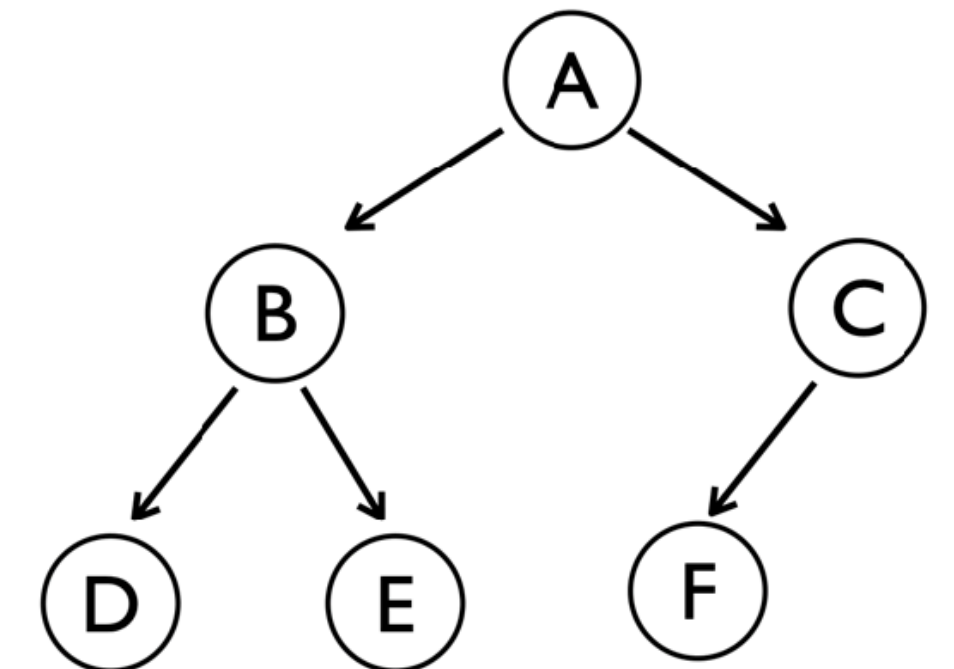


마무리

- 계층적 자료를 표현해야 하는 경우가 있음
 - 트리는 계층적 자료를 표현하는데 주로 사용하는 비 선형 (non-linear) 자료구조



- 이진트리(binary tree)는 모든 노드가 2개의 서브트리를 갖는 트리
 - 모든 노드의 차수는 2 이하임 (최대 2개까지의 자식노드를 가질 수 있음)

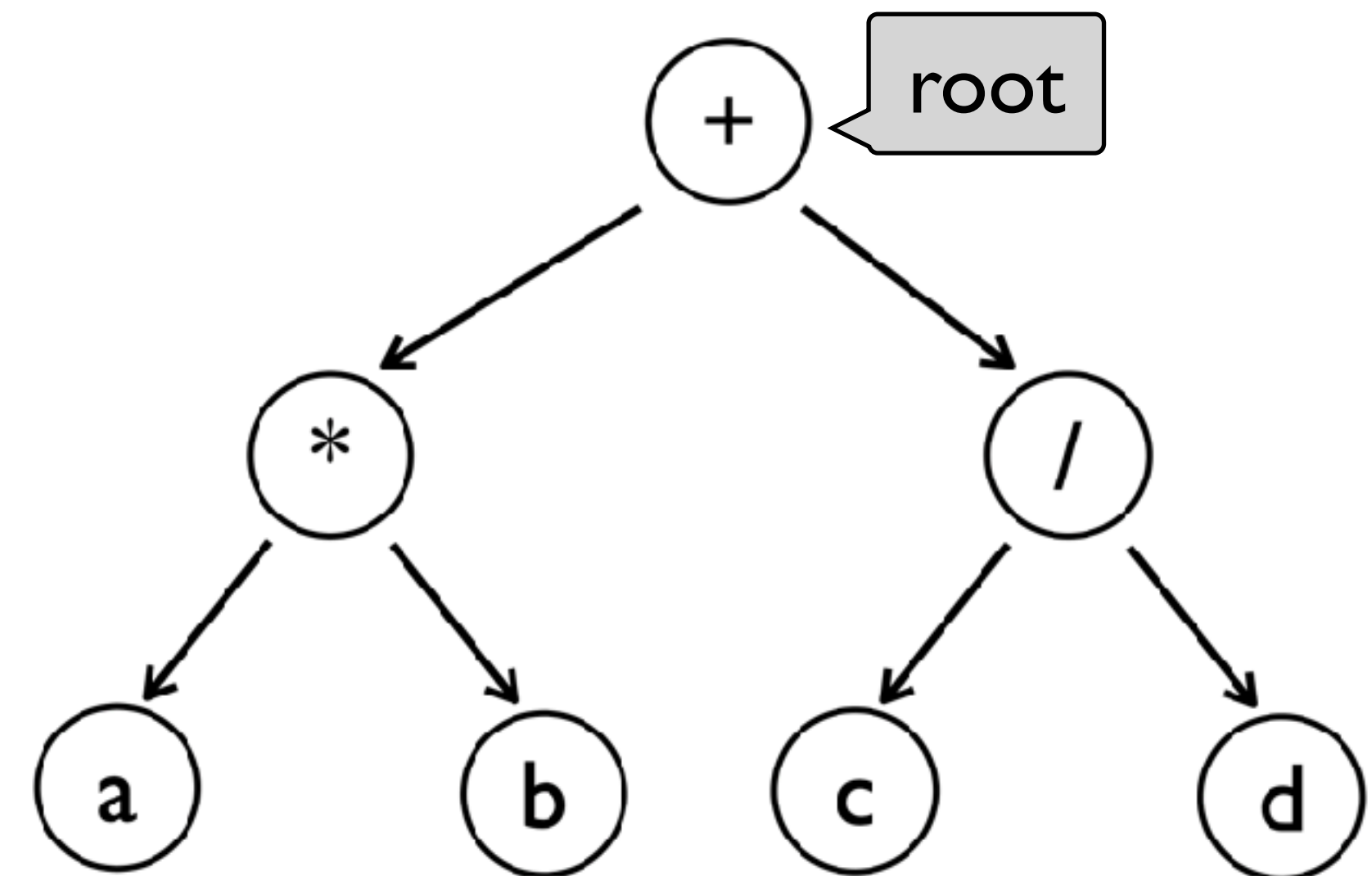


이진트리의 응용: 수식 표현 (Expression Tree)

- 순회로 이진 트리가 표현하고 있는 수식 출력하기

```
procedure printExpression(root)
  if root = NULL then
    return
  if root.left  $\neq$  NULL or root.right  $\neq$  NULL then
    print("(")
  end if
  printExpression(root.left)
  print(root.data)
  printExpression(root.right)
  if root.left  $\neq$  NULL or root.right  $\neq$  NULL then
    print("(")
  end if
end procedure
```

Example

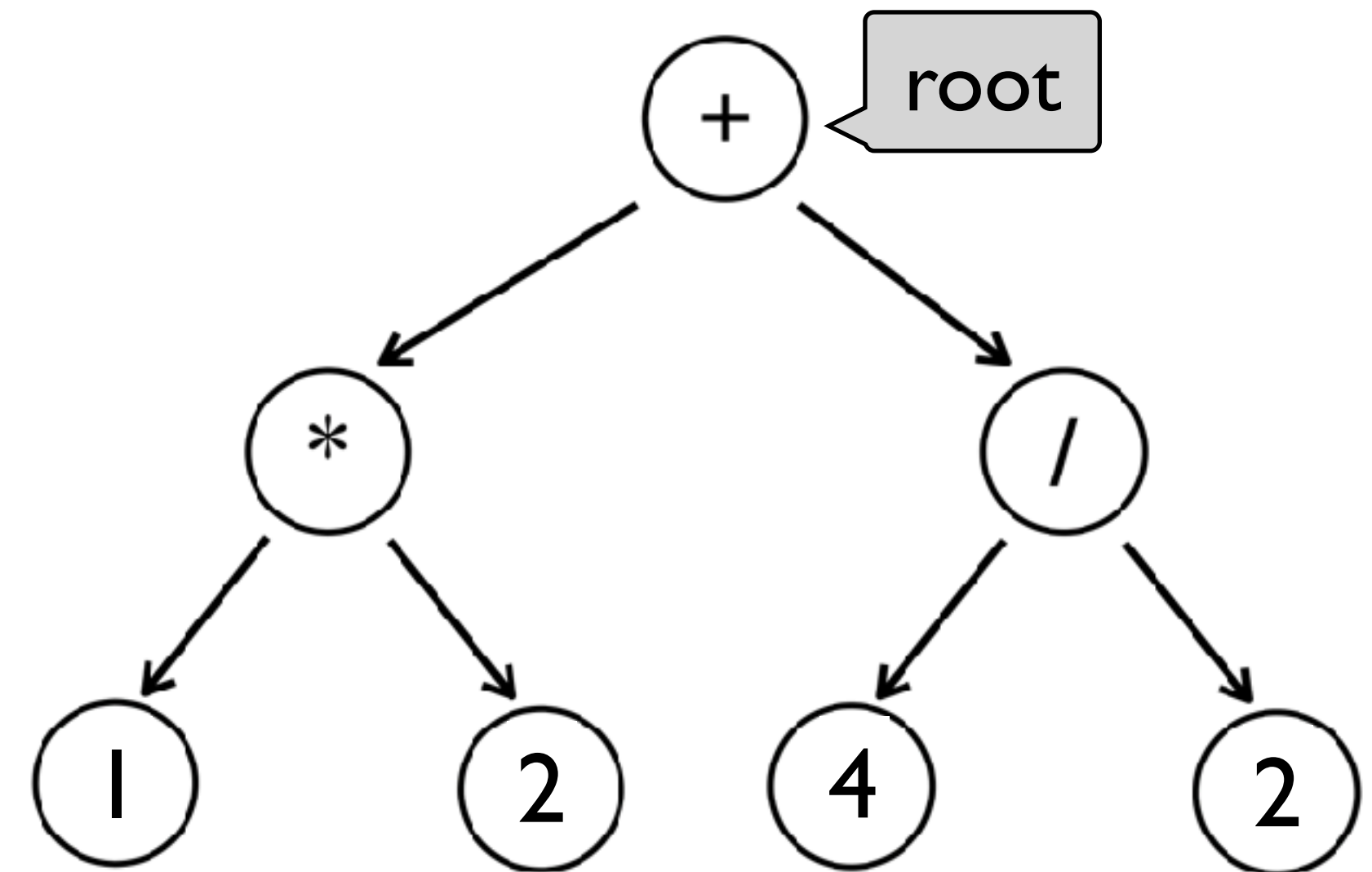


((a * b) + (c / d))

- 이진 트리가 표현하고 있는 수식의 값 계산하기

```
procedure eval(root)
  if root = NULL then
    return 0
  if root.left = NULL or root.right = NULL then
    return root.data
  end if
  leftValue ← eval(root.left)
  rightValue ← eval(root.right)
  if node.data = '+' then
    return leftValue + rightValue
  elif node.data = '-' then
    return leftValue - rightValue
  elif node.data = '*' then
    return leftValue * rightValue
  elif node.data = '/' then
    if rightValue = 0 then
      return error()
    else
      return leftValue/rightValue
    end if
  end if
end procedure
```

Example



((1 * 2) + (4 / 2))

Example

- 이진 트리의 높이를 구하는 알고리즘을 기술하시오

```
procedure height(root)
  if node = NULL then
    return 0
  leftHeight  $\leftarrow$  height(root.left)
  rightHeight  $\leftarrow$  height(root.right)
  if leftHeight > rightHeight then
    return leftHeight + 1
  else
    return rightHeight + 1
end procedure
```

