땅울림

06

트리순회
TreeTraversal

자료구조 스터디 6회차

- 트리 순회 설명
  - 전위 순회
  - 중위 순회
  - 후위 순회
- 문제 유형들

1

트리 순회 설명

트리 개념

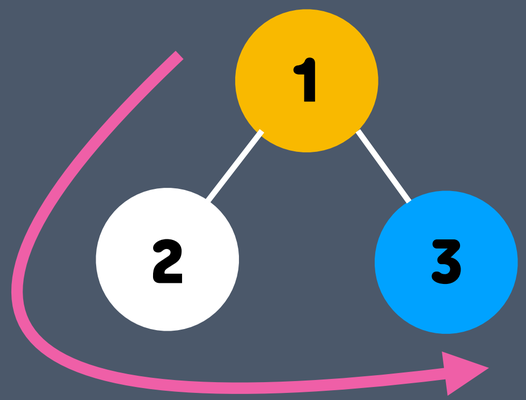
트리를 **탐색**하는 방법

트리는 노드와 간선으로 이루어진 **계층적**인 자료구조
-> 어떤 순서로 데이터에 접근할까

루트 노드를 기준으로 **자식 노드**들을 어떤 순서로 방문할지 !

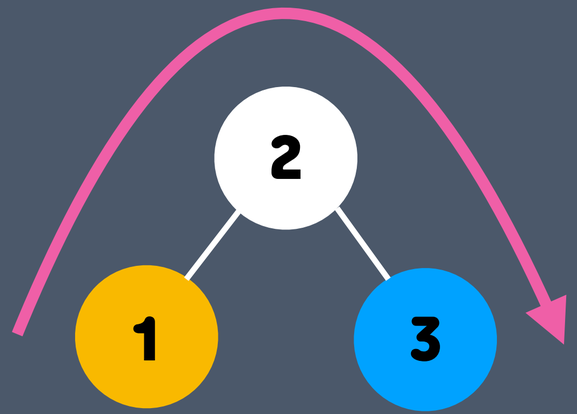
전위 순회
PreOrderTraversal

루트 제일 **처음**



중 - 왼 - 오

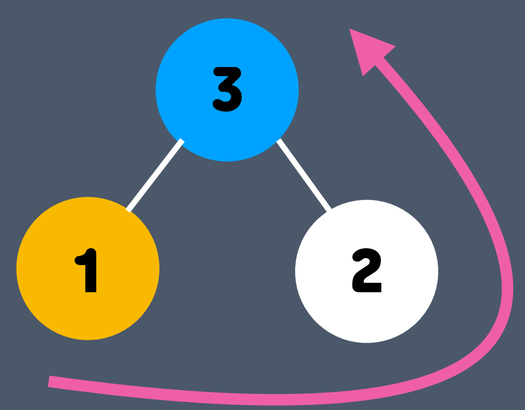
중위 순회
InOrderTraversal



왼 - 중 - 오

후위 순회
PostOrderTraversal

루트 제일 **마지막**



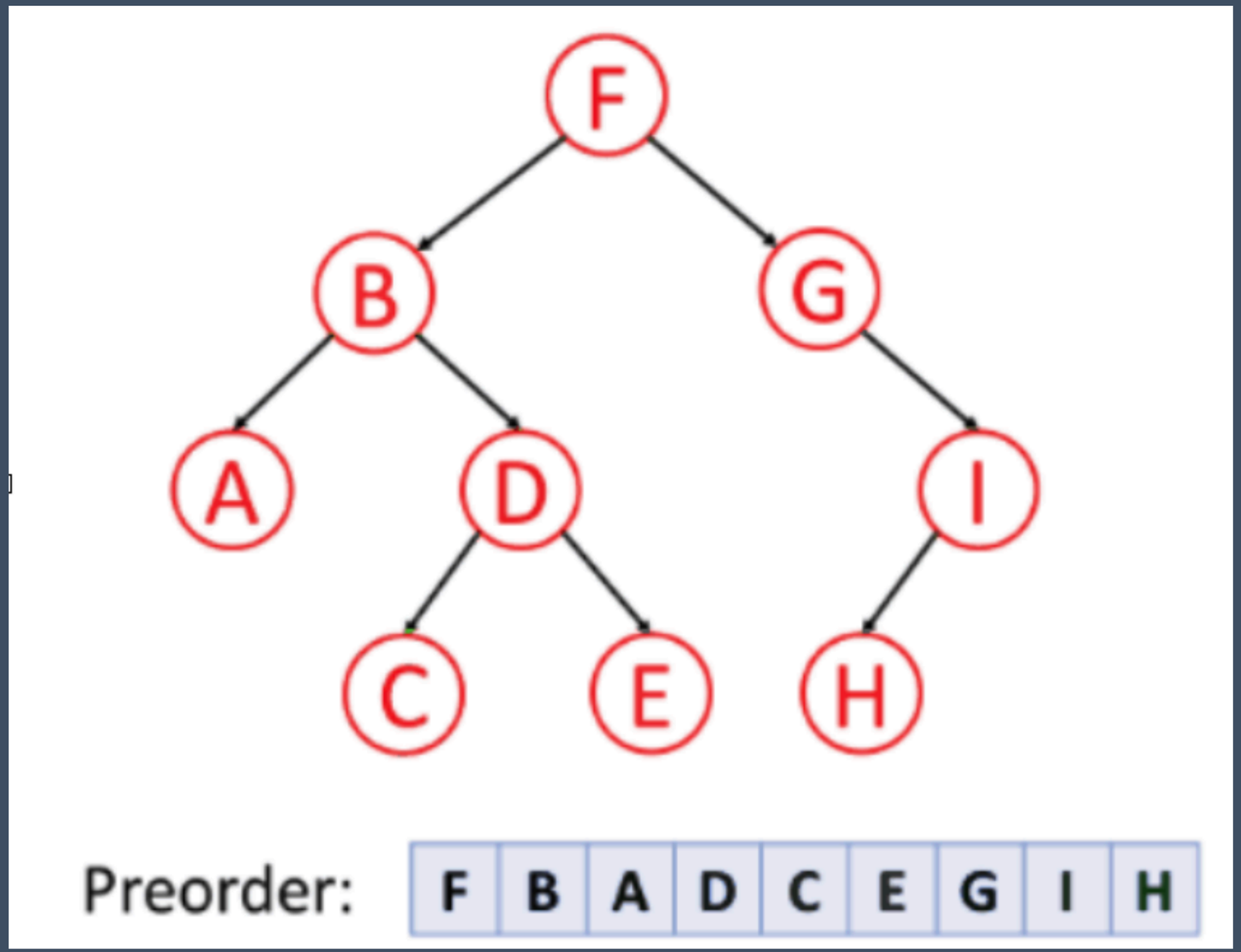
왼 - 오 - 중

1-1

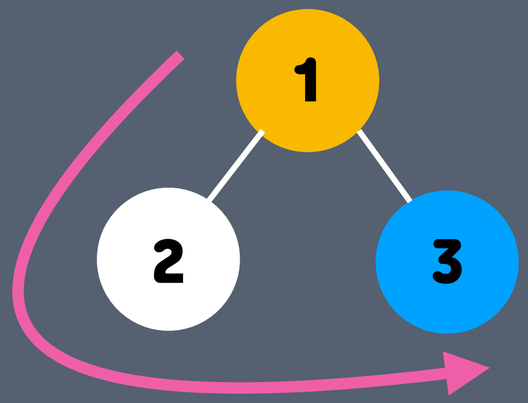
전위 순회

트리 개념

전위 순회 PreOrderTraversal

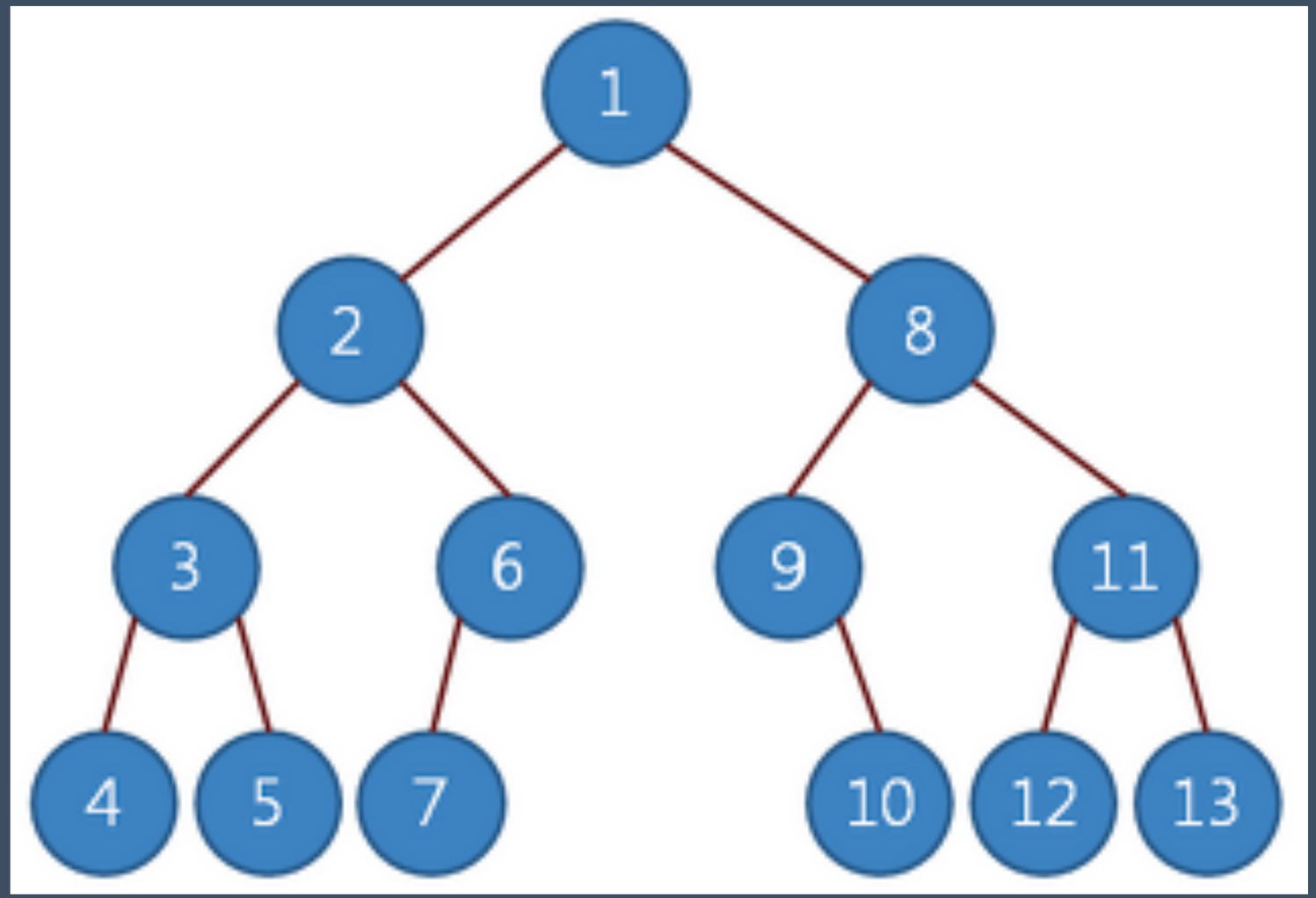


루트 제일 처음

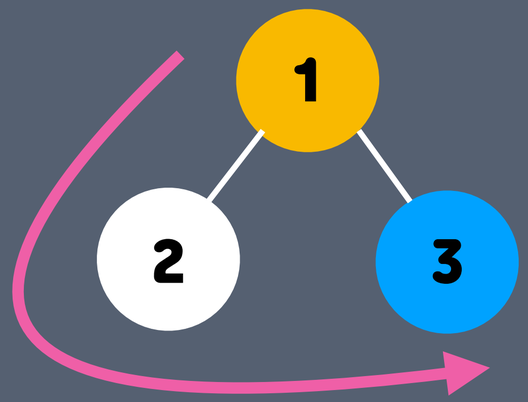


중 - 왼 - 오

전위 순회 PreOrderTraversal



루트 제일 처음



중 - 왼 - 오

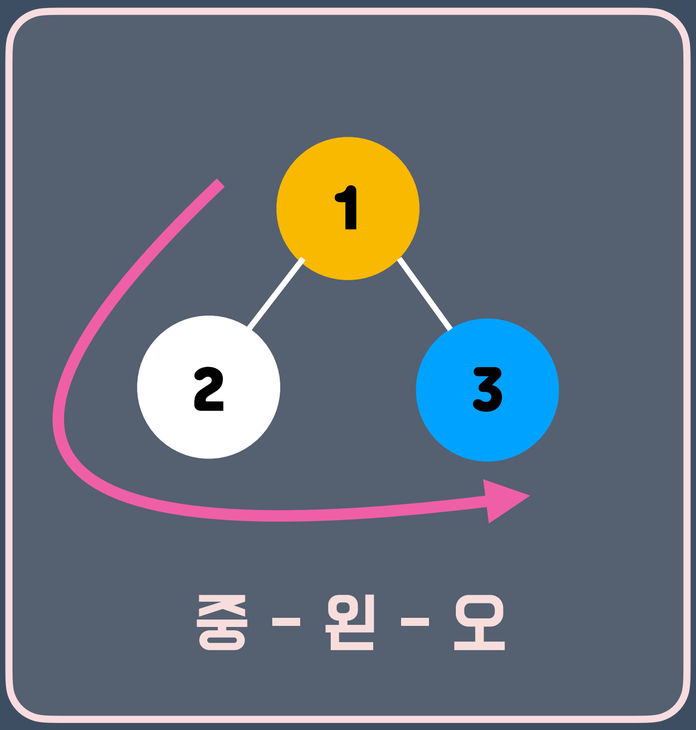
전위 순회 PreOrderTraversal

이진 트리인 경우

preOrderTraversal(Node*)

```
void preOrderTraversal(Node* node) {
    if(node == nullptr) return;

    cout << node->value << " "; // 1. 값 출력 (자신)
    preOrderTraversal(node->left); // 2. 왼쪽 자식
    preOrderTraversal(node->right); // 3. 오른쪽 자식
}
```



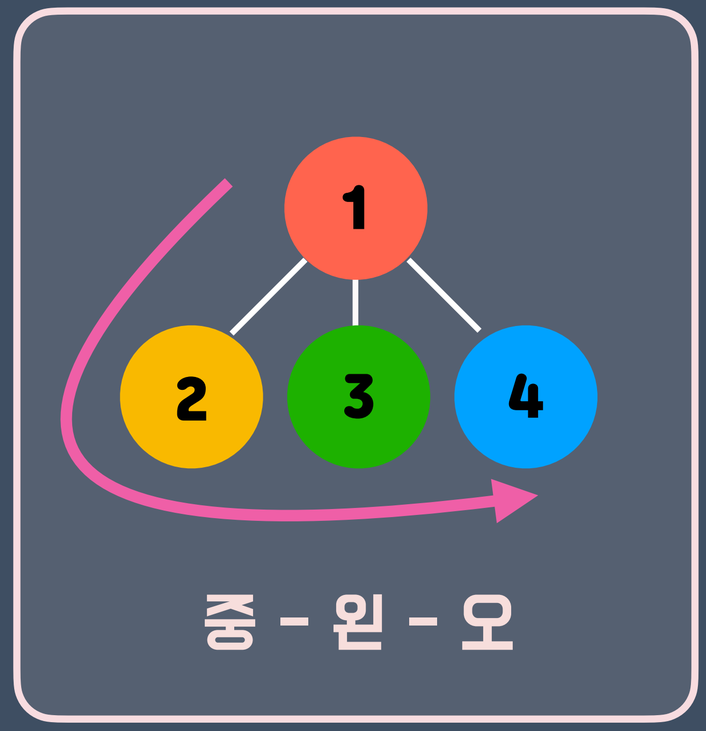
전위 순회 PreOrderTraversal

preOrderTraversal(Node*)

```
void preOrderTraversal(Node* node) {
    // 1. 출력
    cout << node->value << " ";

    // 2. 재귀 (자식 노드들을 전위 순회)
    for(Node* child : node->childList) {
        preOrderTraversal(child);
    }
}
```

자식 많은 트리인 경우



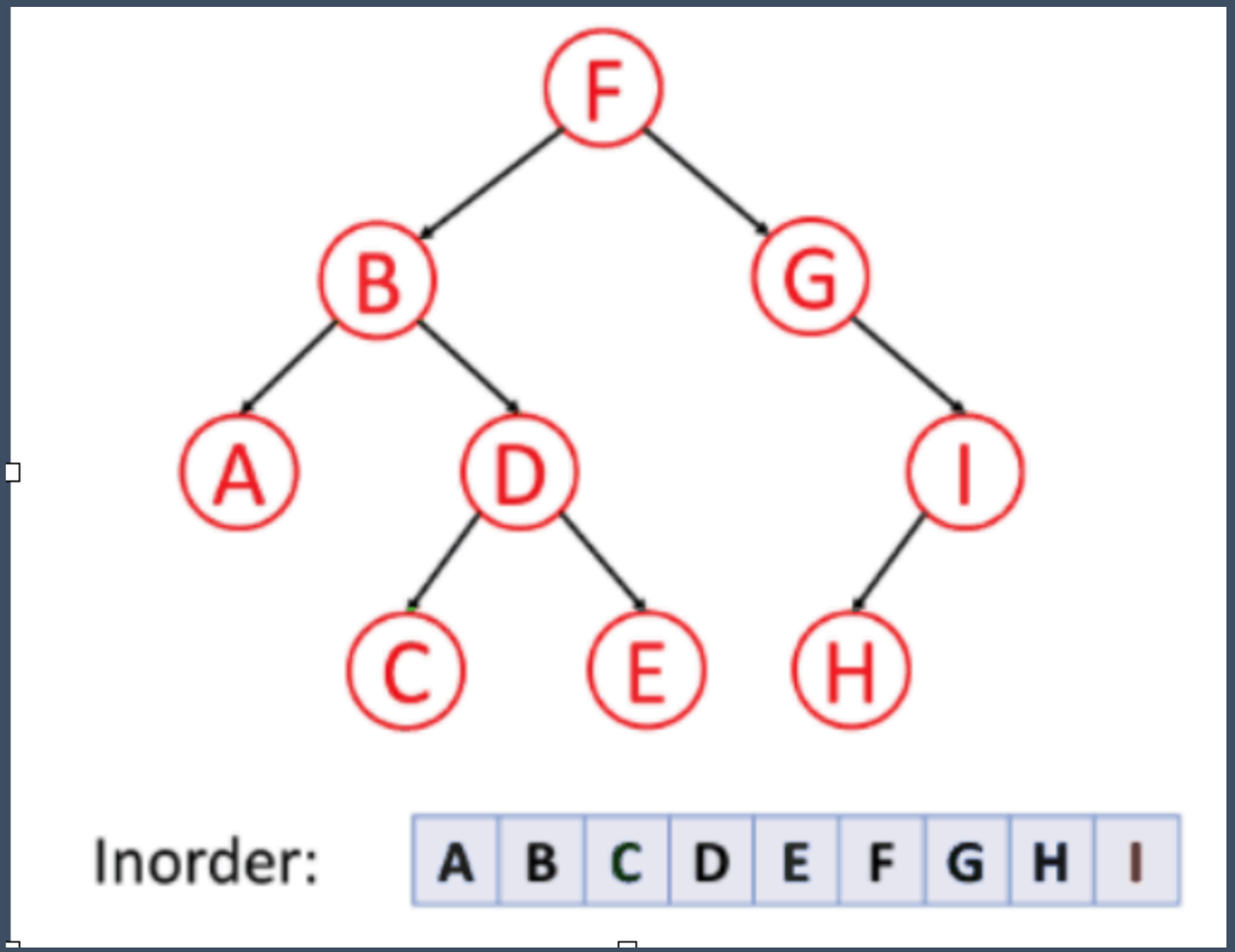
선 출력 후 재귀

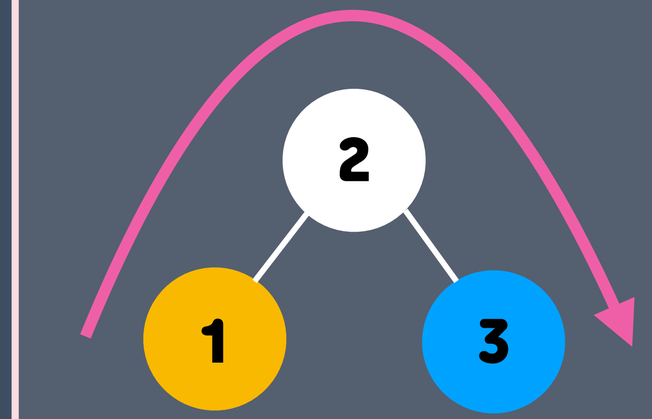
1-2

중위 순회

트리 개념

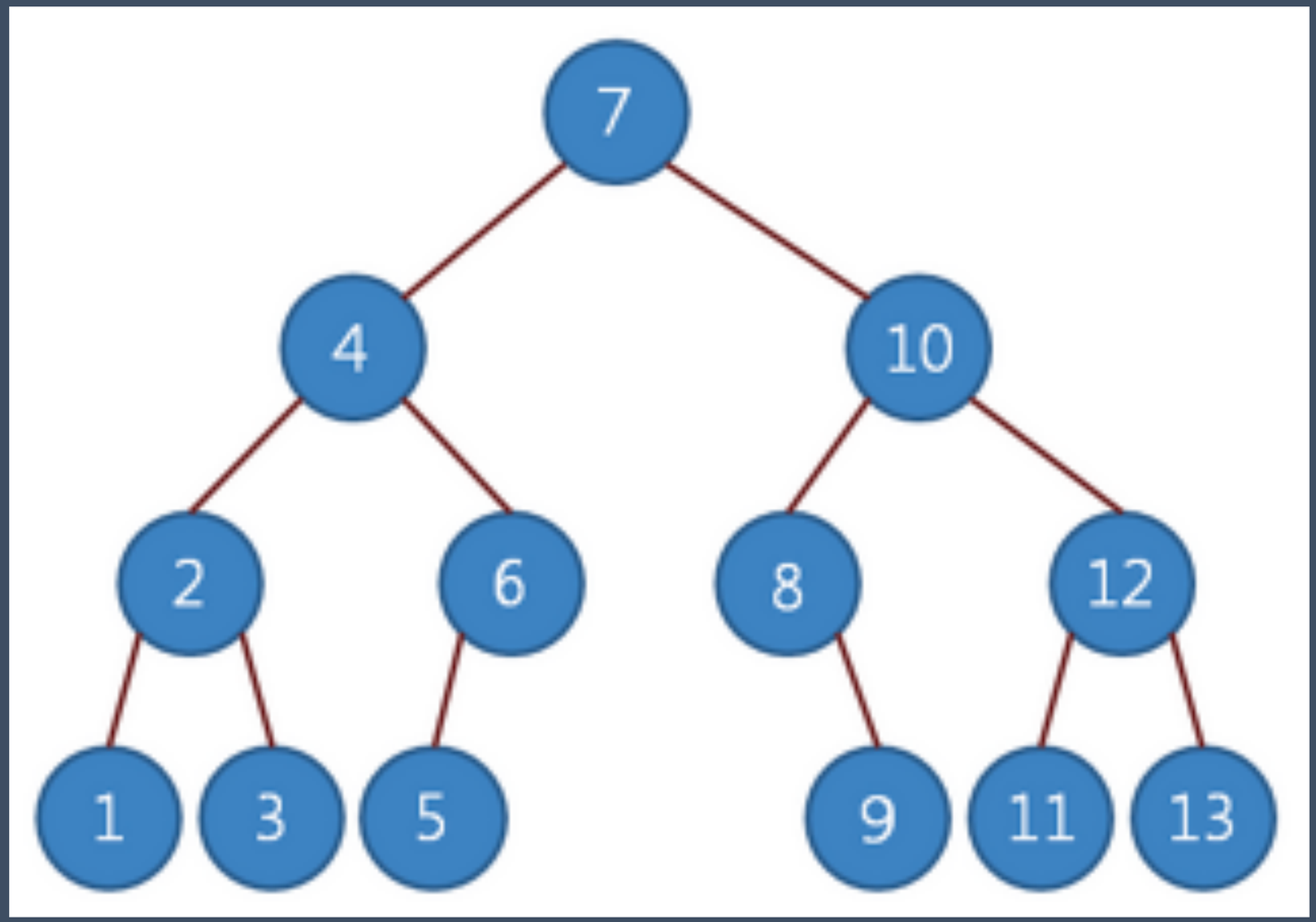
중위 순회 InOrderTraversal

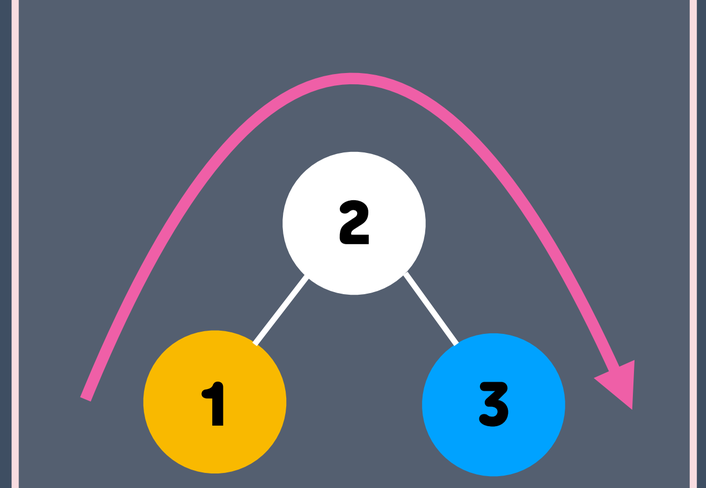


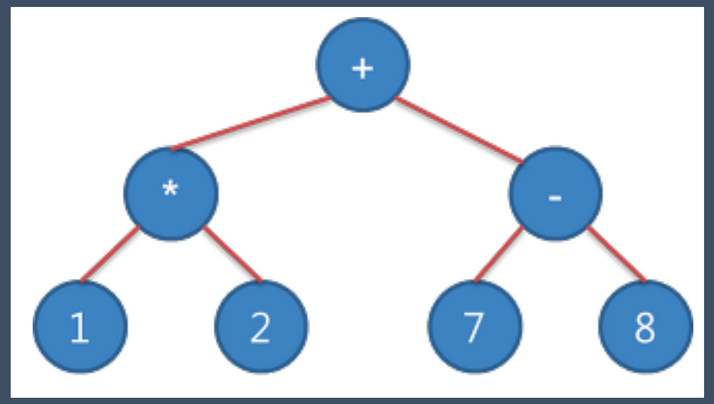


왼 - 중 - 오

중위 순회 InOrderTraversal







응용 : 중위 표기식
(1*2)+(7*8)

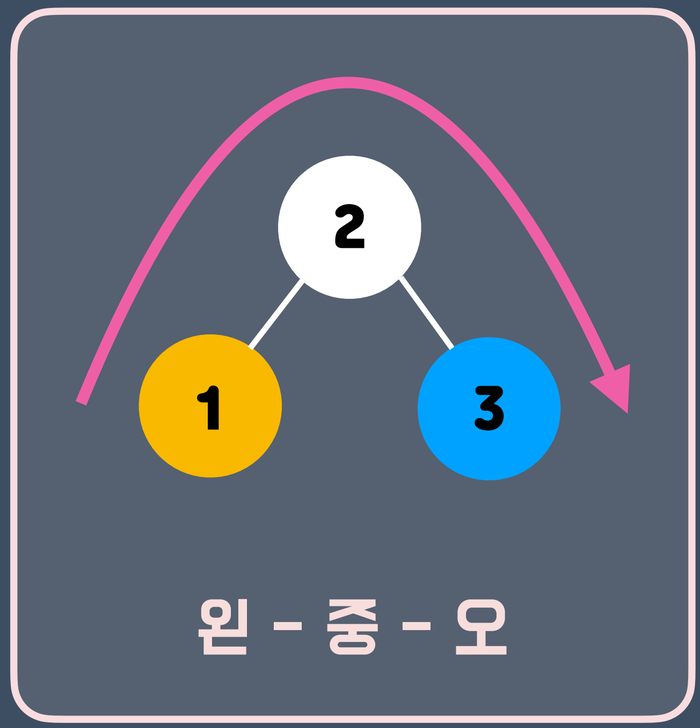
중위 순회 InOrderTraversal

이진 트리인 경우

inOrderTraversal(Node*)

```
void inOrderTraversal(Node* node) {
    if(node == nullptr) return;

    inOrderTraversal(node->left); // 1. 왼쪽 자식
    cout << node->value << " "; // 2. 값 출력 (자신)
    inOrderTraversal(node->right); // 3. 오른쪽 자식
}
```

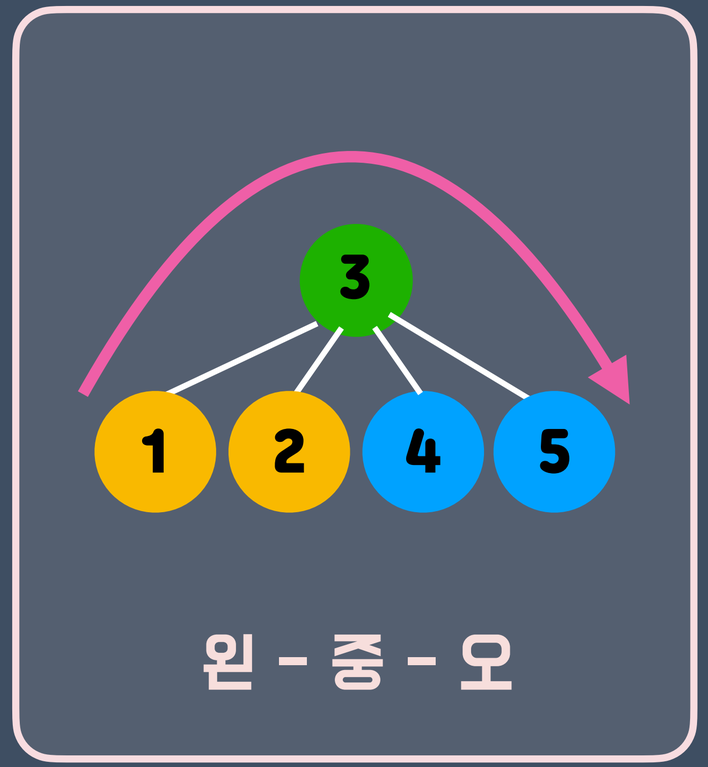


중위 순회 InOrderTraversal

자식 많은 트리인 경우

```
inOrderTraversal(Node*)

자식사이즈의 절반만큼(왼쪽 자식들이라고 기준) 순회 후 (1,2)
자신(중간 노드) 출력  (3)
남은 자식들 순회(오른쪽 자식들이라고 기준) (4,5)
```

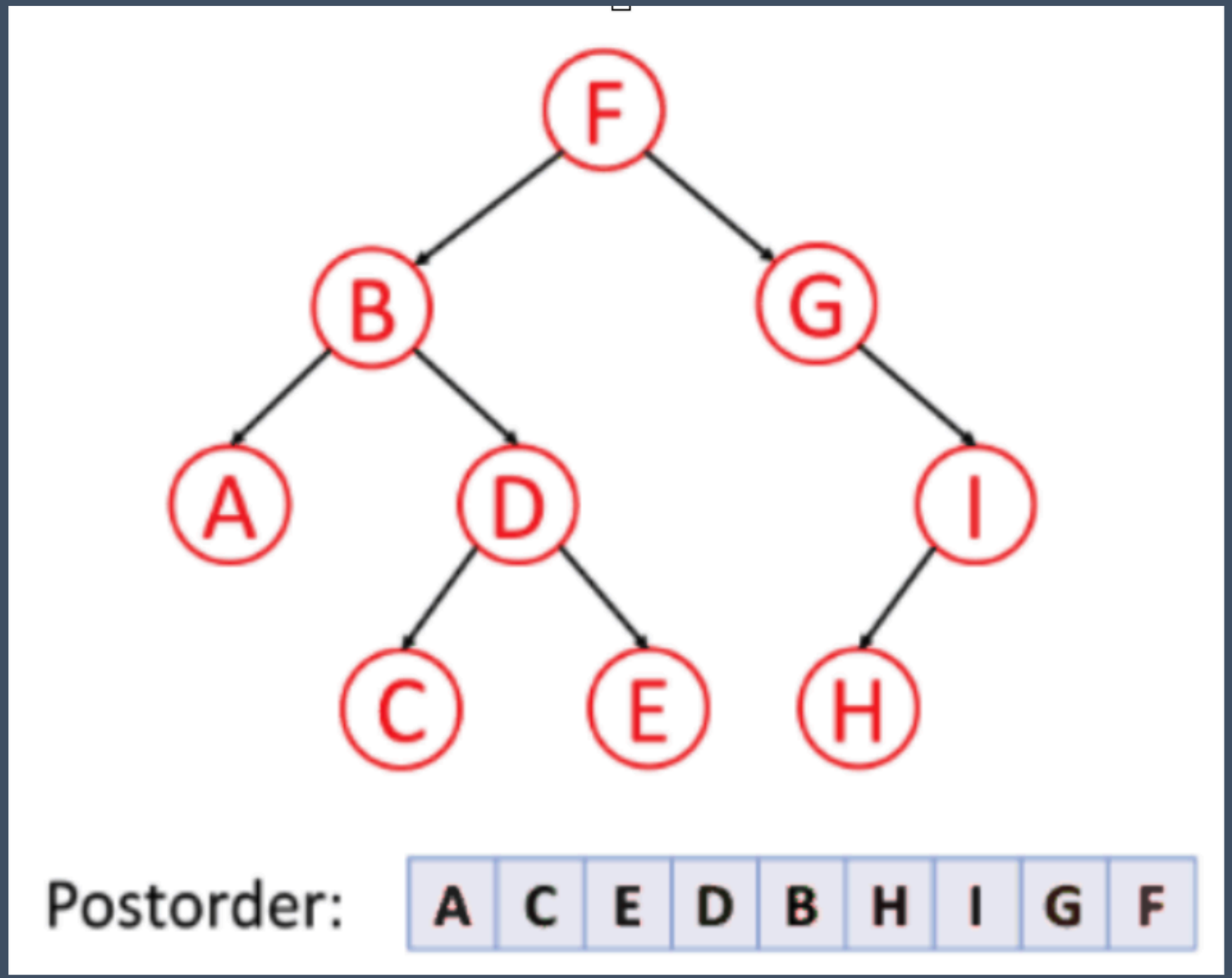


1-3

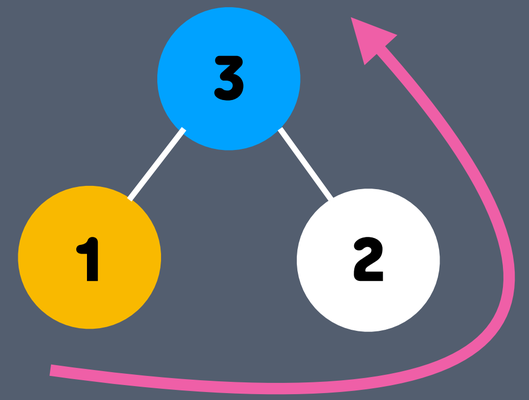
후위 순회

트리 개념

후위 순회 PostOrderTraversal

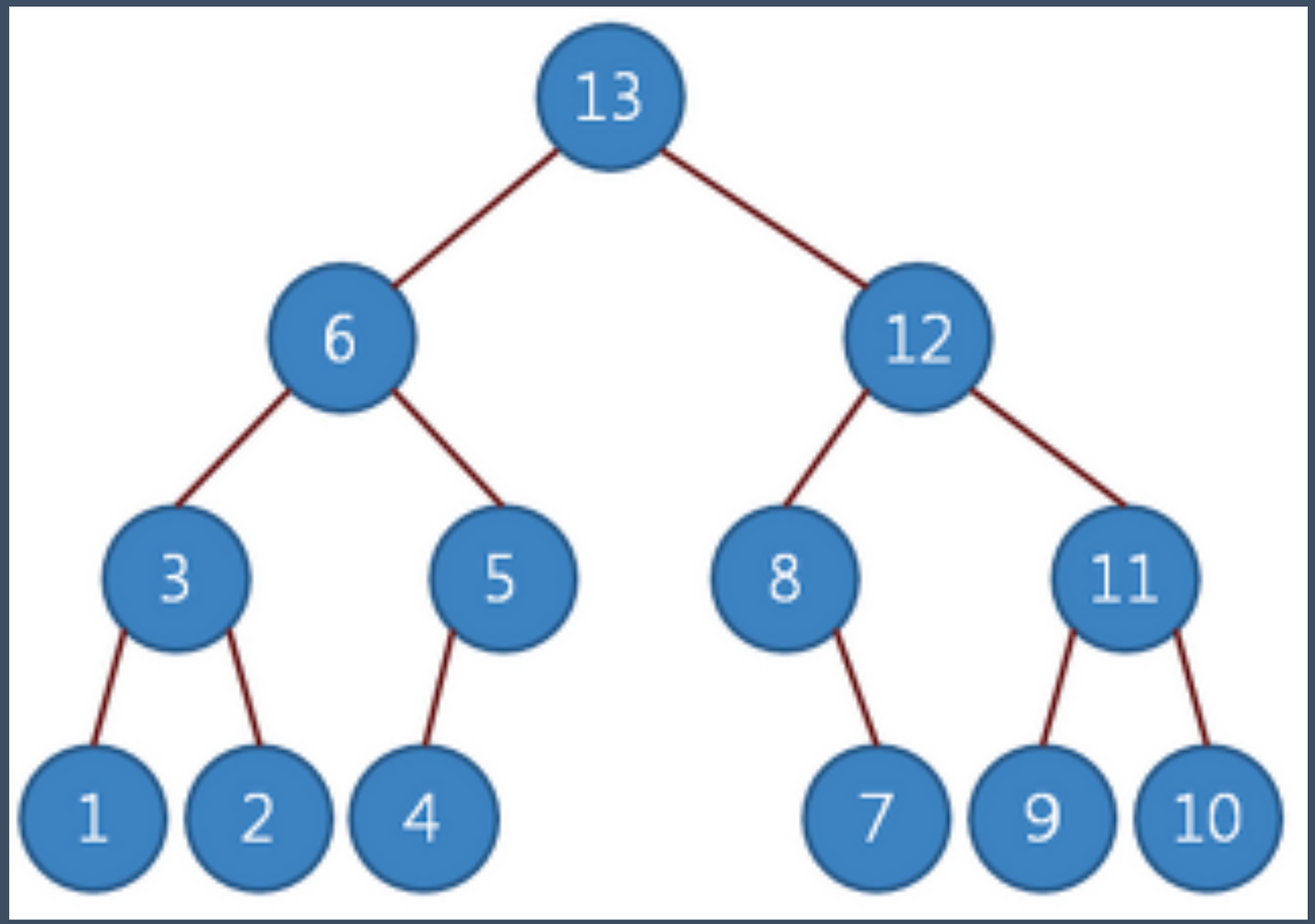


루트 제일 마지막

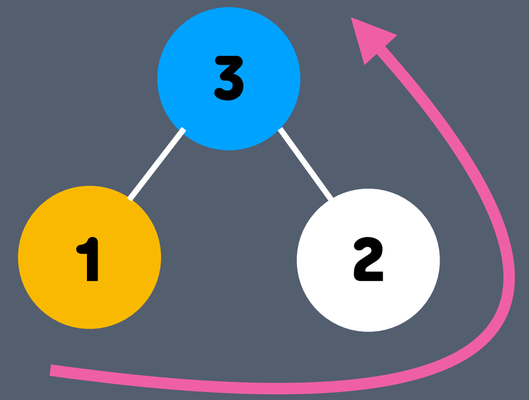


왼 - 오 - 중

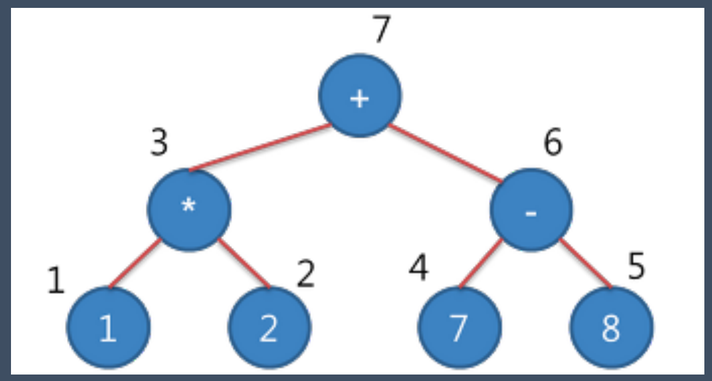
후위 순회 PostOrderTraversal



루트 제일 마지막



왼 - 오 - 중



응용 : 후위 표기식

12*78*+

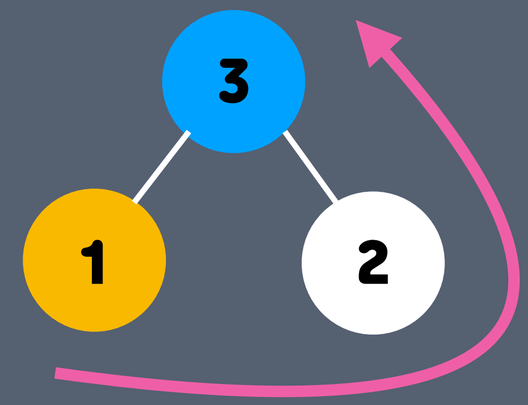
후위 순회 PostOrderTraversal

이진 트리인 경우

postOrderTraversal(Node*)

```
void postOrderTraversal(Node* node) {
    if(node == nullptr) return;

    postOrderTraversal(node->left); // 1. 왼쪽 자식
    postOrderTraversal(node->right); // 2. 오른쪽 자식
    cout << node->value << " "; // 3. 값 출력 (자신)
}
```



왼 - 오 - 중

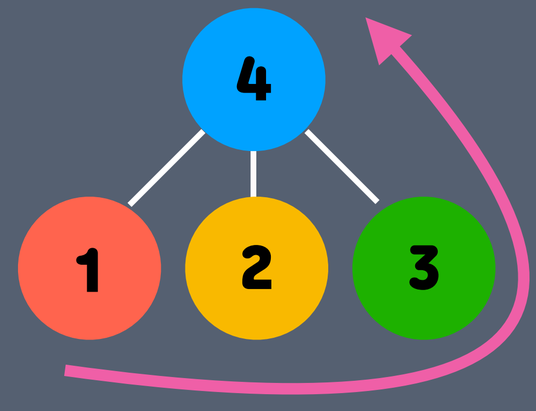
후위 순회 PostOrderTraversal

postOrderTraversal(Node*)

```cpp
void postOrderTraversal(Node* node) {
    // 1. 재귀 (자식 노드들을 후위 순회)
    for(Node* child : node->childList) {
        postOrderTraversal(child);
    }

    // 2. 출력
    cout << node->value << " ";
}
```

자식 많은 트리인 경우



왼 - 오 - 중

선 재귀 후 출력

이진트리 세가지 순회 비교

전위 순회 PreOrderTraversal

```
cout << node->value;
pre(node->left);
pre(node->right);
```

출력 왼 오

중위 순회 InOrderTraversal

```
in(node->left);
cout << node->value;
in(node->right);
```

왼 출력 오

후위 순회 PostOrderTraversal

```
post(node->left);
post(node->right);
cout << node->value;
```

왼 오 출력

자식 많은 트리 순회 비교

전위 순회 PreOrderTraversal

```cpp
// 1. 출력
cout << node->value << " ";

// 2. 재귀 (자식 노드들을 전위 순회)
for(Node* child : node->childList)
    preOrderTraversal(child);
}
```

선 출력 후 재귀

후위 순회 PostOrderTraversal

```cpp
// 1. 재귀 (자식 노드들을 후위 순회)
for(Node* child : node->childList)
    postOrderTraversal(child);
}

// 2. 출력
cout << node->value << " ";
```

선 재귀 후 출력

2

트리 순회 문제 유형

문제 유형

트리 구현 + 특정 함수 구현 (순회 이용)

21년도 실습 문제 족보

https://github.com/Landvibe-DataStructure-2023Study/LimJumin/tree/main/21%20%EC%8B%A4%EC%8A%B5%20%EC%BD%94%EB%93%9C/%ED%8A%B8%EB%A6%AC%20%EC%88%9C%ED%9A%8C

P1	P2	P3	P4
전위순회하며 부모노드 값 출력	전위순회한 값과 depth가 주어질 때, 후위 순회한 값 출력	후위순회하며 부모 노드 값 출력	후위순회한 값과 depth가 주어질 때, 전위 순회한 값 출력
기본 문제	약간 어렵	기본 문제	어려운 문제

22년도(김영호 교수님) 족보

https://github.com/Landvibe-DataStructure-2023Study/LimJumin/tree/main/22%20%EA%B9%80%EC%98%81%ED%98%B8%20%EC%8B%A4%EC%8A%B5%20%EC%BD%94%EB%93%9C/%ED%8A%B8%EB%A6%AC%20%EC%88%9C%ED%9A%8C

P1	P2	P3	P4
전위순회하며 Depth 출력	폴더 개수 계산 (자신의 하위 리프 노드들 개수 계산)	후위 순회하며 depth 출력	폴더 용량 계산 (자신의 하위 자식들 용량 합)
기본 문제	약간 어렵	기본 문제	약간 어렵

트리 순회 문제 유형