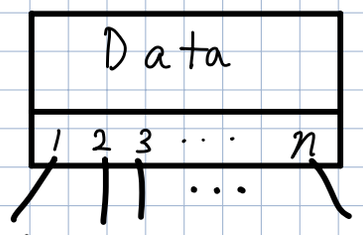
Title : _____ Name : _____

Date : _____

Tree 구현 방법

① N-link node



각 n개의 링크 포인터가 자식 노드를 가리키는 구조

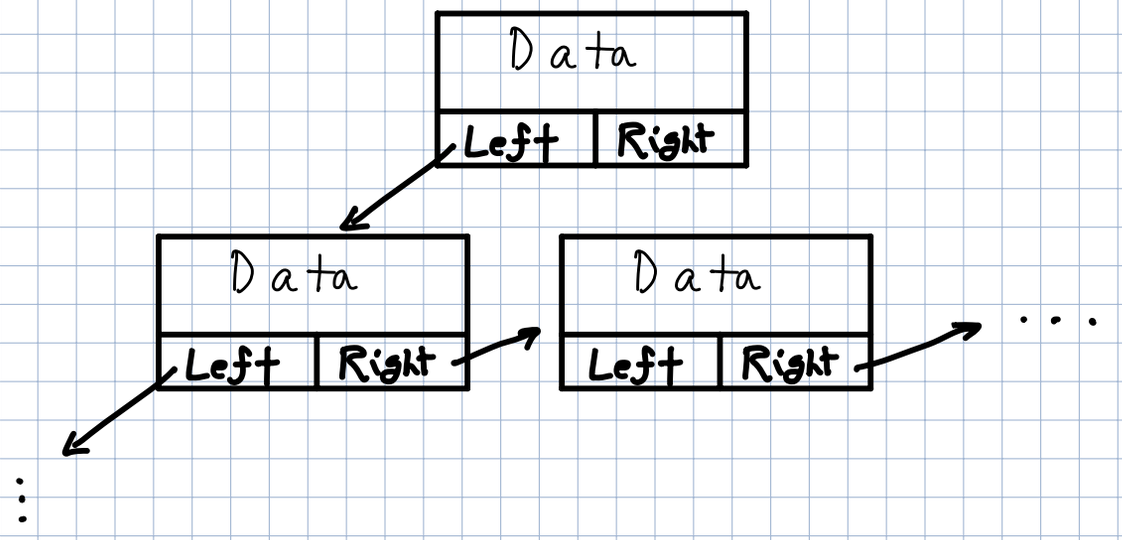
단점) 자식이 너무 많으면? 유연하게 링크 추가 불가능

자식이 너무 적으면? 기존에 생성된 링크만큼 낭비 발생

② Left Child Right Sibling

왼쪽 링크는 자식 노드를 가리키고

오른쪽 링크는 본인과 레벨이 같은 형제 노드를 가리킴



이 방법이 일반적인 트리 구현 시 사용된다.

Title : _____ Name : _____

Date : _____

Tree Node 구현

```
typedef struct tagLCRSNode {
        struct tagLCRSNode LeftChild;
        struct tagLCRS Node RightSibiling;

        ElementType Data;
} LCRSNode;
```

이 노드를 생성하려면,

```
LCRSNode * LCRS_CREATE (ElementType newData) {
    ① 노드 동적 할당
    LCRSNode * NewNode =
        (LCRSNode*)malloc (sizeof(LCRSNode));

    NewNode → LeftChild = NULL;
    NewNode → RightSibiling = NULL;
    NewNode → Data = newData;
}
```

자식노드 추가하는 조건

① 부모노드의 LeftChild가 NULL → 바로 추가

② NULL이 아니면 자식노드로 이동해서 자식노드의 RightSibiling 포인터를 끝까지 따라가서 맨 오른쪽에 노드 추가

Title : _____ Name : _____

Date : _____

이진 트리

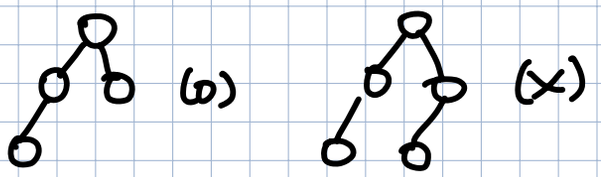
- 최대 자식노드의 개수가 2개인 트리 (Degree)

* 포화 이진트리 (Full Binary Tree)

→ Leaf를 제외한 모든 Node의 Degree = 2

* 완전 이진트리 (Complete Binary Tree)

→ Leaf Node가 왼쪽부터 순서대로 채워진 트리



* 높이 균형 트리 (Height Balanced Tree)

⇒ Root를 기준으로 왼쪽 Subtree의 Height과 오른쪽 Subtree의 Height의 차이가 2 이상 나지 않는 이진트리

$$|\text{Left Sub Height} - \text{Right Sub Height}| < 2$$

* 완전높이 균형 트리

$$|\text{Left Sub Height} - \text{Right Sub Height}| = 0$$

Title : Name :

Date :

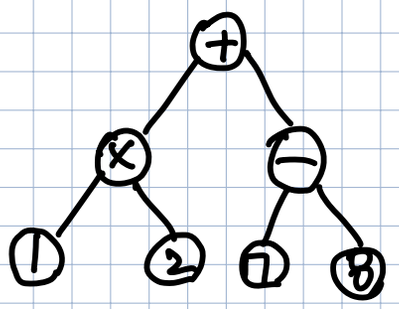
순회방법

전위 : Root → L → R

중위 : L → Root → R

후위 : L → R → Root

예를 들어 중위 표기식 (1×2)+(7-8) 을 Binary Tree로 나타낸다면



전위순회
⇒ + × 1 2 - 7 8

후위순회
⇒ 1 2 × 7 8 - +

이진트리 구현

2-link 형태로 표현 (Simple Binary Tree = SBT)

```
typedef struct tagSBTNode {
    ElementType Data;

    struct tagSBTNode * Left;
    struct tagSBTNode * Right;
} SBTNode;
```

Title : Name :

Date :

노드 생성함수

```
SBTNode *  Create (ElementType NewData){
    SBTNode * NewNode = (SBTNode*) malloc(sizeof(SBTNode));
    NewNode -> Left, Right =NULL;
    NewNode -> Data = NewData;
    return NewNode;
}
```

전위순회 함수 → 재귀로 구현

```
void  Pre(SBTNode* Node) {
   if (Node == NULL){ return } // 종료조건
   printf( Node -> Data)

   Pre (Node -> Left) → Left 탐색
   Pre( Node -> Right) → Right 탐색
```

*중위, 후위 순회는 위 코드를 잘 바꿔주면 된다. Easy

트리 소멸 방법

메모리 누수방지를 위해 반드시 후위순회를 통해 삭제