제8장 스레드 트리

1. 스레드 트리

○ 이진 트리의 노드 순회: 전위 순회, 중위 순회, 후위 순회

○ 이진 트리의 노드를 순회할 때. 방문하지 않고 지나쳐 온 노드들은 스택에 저장하여 관리해야 하는 번거로움이 발생함

○ 스레드 트리: '스레드'라는 포인터를 추가하여 트리 순회를 편리하게 한 것

- 스레드: 순회 방법에 따른 방문순서를 유지하는 포인터

2. 스레드 트리 구현

[스레드의 구현 방법 1]

○ 포인터 필드의 추가: 스레드를 저장하는 포인터를 추가하는 것

- 왼쪽 스레드 포인터, 왼쪽 자식 포인터, 데이터, 오른쪽 자식 포인터, 오른쪽 스레드 포인터 필드로 노드 구조를 정의함

○ 오른쪽 스레드: 정해진 순회 순서에 따른 그 노드의 후속 노드를 가리키고

○ 왼쪽 스레드: 그 노드의 선행 노드를 가리킴

[추가된 포인터 필드를 이용한 중위 순회 연산과정]

① 순회할 트리의 첫번째 노드를 가리키는 포인터 firstin을 매개변수로 하는 함수의 이름을 inorder() 로 합니다.

② 노드를 가리킬 수 있는 (TNode) 타입의 포인터 p를 생성하고 첫번째 노드를 가리키도록 합니다.

③ 포인터 p가 가리키는 데이터(info)를 출력하고 p를 p의 오른쪽 스레드 값으로 바꾸어 (p → right_thread) 중위 순회에 따른 다음 노드를 가리키도록 수정합니다.

④ 이 과정을 p가 null이 될 때까지 반복합니다.

```
void inorder(struct TNode *firstin) {
    struct TNode *p ;
    p = firstin ;
    while(p ! = NULL) {
        printf("%d", p->info) ;
        p = p->right_thread ;
    }
}
```

○ 스택을 운영하지 않고도 쉽게 트리에 속한 모든 노드를 순회 할 수 있음

○ 하지만 스레드를 위해 추가 기억장소를 사용한다는 부담이 생김

[스레드의 구현 방법 2]

○ 노드의 빈 포인터 필드 활용: 기존 이진 트리의 노드 구조를 그대로 사용하면서, 노드에 있는 사용하지 않는 포인터를 활용하는 방법

○ 추가 기억장소를 사용하지 않아도 되는 장점이 있음

○ 노드의 빈 포인터 필드의 활용

- 트리의 노드가 n개일 때, 전체 포인터는 2n개
- 루트 노드를 제외한 각 노드는 모두 진입차수가 1
- 루트 노드를 제외한 전체 노드 즉, 누군가로부터 가리켜져야 할 노드는 n-1개 (즉, null이 아닌 포인터 개수: n-1)
- 2n - (n-1) = n+1 개의 null 포인터가 노드에 존재함
- n+1개의 null 포인터를 스레드로 활용함

○ 이진 트리의 중위 순회

- 어떤 노드 X에 대해 오른쪽 포인터가 null이면 이 포인터를 X노드의 다음으로 순회되는 노드를 가리키도록 하고,
  - 왼쪽 포인터가 null이면 X노드의 바로 직전에 순회된 노드를 가리키게 함

○ 전위 순회: 각 노드의 왼쪽 포인터 필드를 스레드로 사용하는 제한을 가정함

- 어떤 노드의 왼쪽 포인터가 실제 왼쪽 자식을 가리키면 그대로 두고,
- 왼쪽 포인터가 null이면, 오른쪽 포인터가 다음으로 순회하는 노드를 가리키도록 지정함
- 각 노드에 대해 포인터가 스레드로 사용 중인지 아니면 서브트리에 대한 포인터인지를 구분하기 위해 태그 필드가 필요함(삽입/삭제 연산에서 반드시 필요함)

3. 스레드 트리 순회, 삽입, 삭제