Linked List

연결 리스트

목차

1. 연결 자료구조 방식

(1). 연결 자료구조 방식을 사용하는 이유

- 순차 자료구조 방식에서는 데이터의 논리적인 순서와 물리적인 순서가 일치해야 한다. 이 때문에 데이터의 위치를 찾아 접근하기 쉽다. 하지만 삽입 연산이나 삭제 연산 후, 연속적인 물리 주소를 유지하기 위해 데이터들을 이동시켜야 한다. 이러한 작업이 많아지면 오버헤드가 증가해 성능상의 문제가 발생할 수 있다. 또한, 순차 자료구조 방식이 사용하는 배열은 메모리 사용에 있어서 비효율적이다.

- 이러한 순차 자료구조 방식에서의 연산 시간에 대한 문제와 저장 공간에 대한 문제를 개선한 것이 연결 자료구조 방식.

- 연결 자료구조는 각 데이터에 저장된 다음 주소를 참조하여 연결되는 방식이기 때문에 물리적인 순서를 맞추기 위한 *오버헤드가 발생하지 않음.

  * 오버헤드: 어떤 처리를 하기 위해 들어가는 간접적인 처리 시간 · 메모리 등

- **배열과 연결 자료구조 비교 테이블**

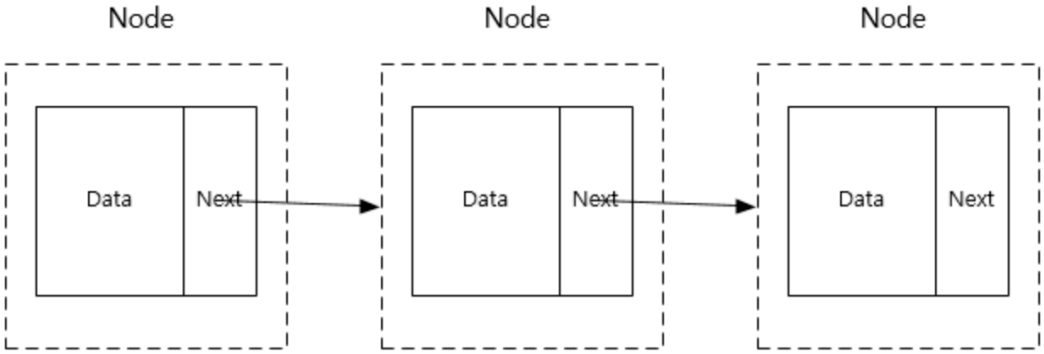
	배열	연결 리스트
장점	원소에 대한 접근이 빠르다	동적으로 메모리 사용가능 메모리 사용 효율적 데이터 재구성이 용이 대용량 데이터 처리 적합
단점	메모리 사용이 비효율적 배열 내에서 데이터 이동과 재구성이 힘들다	특정 위치의 데이터를 검색할 때 느리다 메모리를 추가적으로 사용해야 한다

(2). 노드

- 노드 = 데이터 필드 + 링크 필드
  데이터 필드 = 원소의 값을 저장, 하나 이상일 수 있다.
  링크 필드 = 다음 노드의 주소를 저장

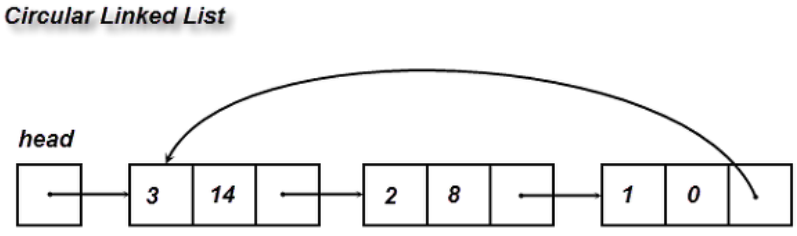
(3). 단순 연결 리스트

- 노드의 첫 번째 링크 필드가 두 번째 노드를 가리키고, 두 번째 노드의 링크 필드가 세 번째 노드를 가리킨다. 마지막 노드의 링크 필드에는 Null 값이 저장되어있어 다음 노드를 가리키지 않는다.



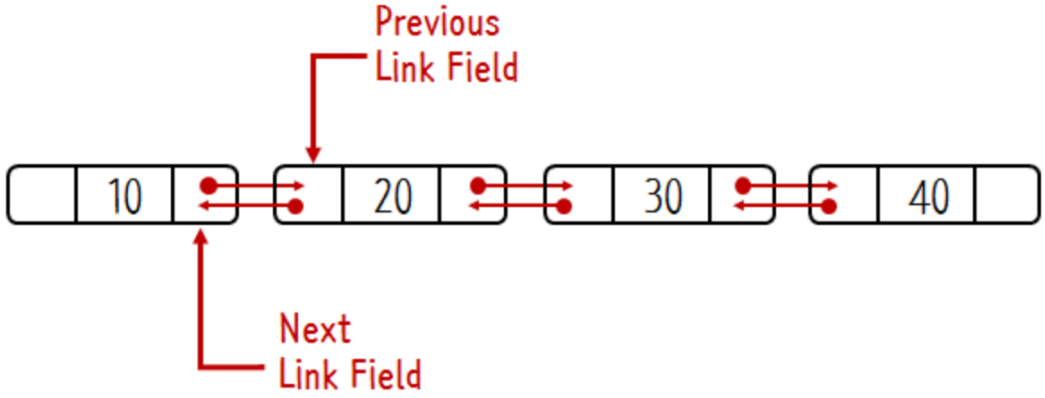
(4). 원형 연결 리스트

- 단순 연결 리스트는 임의의 노드에서부터 이전에 위치한 노드에 접근할 수 없고, 다시 head 부터 시작해야 한다. 이런 제한을 해결하기 위한 한 가지 구조가 원형 연결 리스트로 마지막 노드의 링크 필드가 첫 번째 노드를 가리킨다.



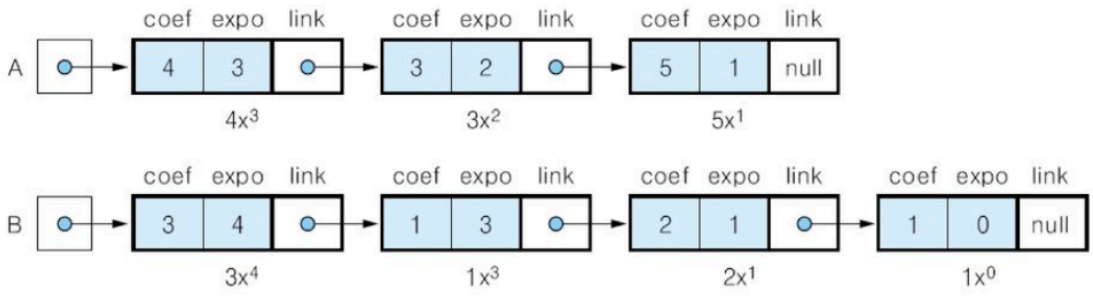
(5). 이중 연결 리스트

하나의 노드에 자신의 앞에 있는 링크 필드와 그 자신의 뒤에 있는 노드의 링크 필드를 연결시킨 구조로, 여러 노드가 링크 필드로 연결된 연결 리스트 구조이다. 데이터 필드의 양방향 검색이 가능하고 삽입, 삭제가 쉽다는 장점을 갖는다.



(6.) 다항식의 연결 자료구조 표현

- 단순 연결 리스트를 이용하여 다항식을 표현할 수 있다. 다항식에 있는 하나의 항은 하나의 노드로 표현하는데, 노드는 각 항에 대해서 계수와 지수를 저장하기 위해 데이터 필드 두 개와 다음 항에 대한 노드를 연결하는 링크 필드로 구성한다.



*리스트의 종류에 따른 연산 방법(삽입, 삭제등)은 스터디 도서 참고