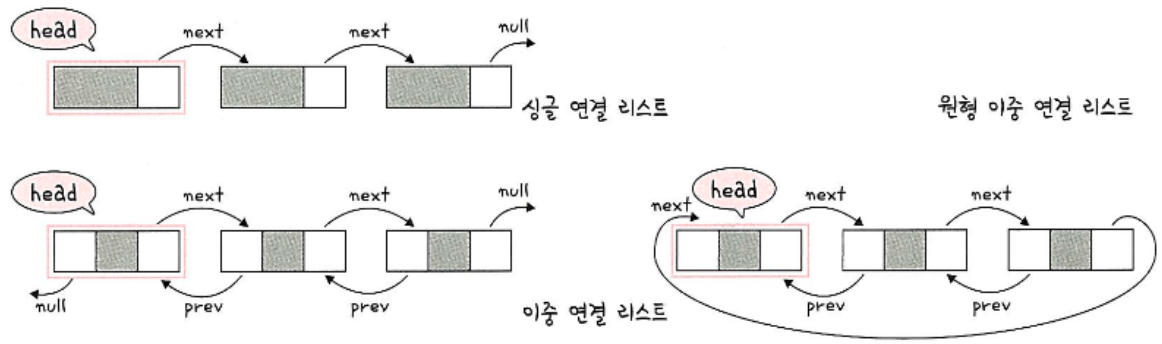
5.2 선형 자료 구조

5.2.1 연결 리스트

- 연결 리스트는 데이터를 감싼 노드를 포인터로 연결해서 공간적인 효율성을 극대화시킨 자료구조이다.
- 삽입과 삭제가 O(1)이 걸리며 탐색에는 O(n)이 걸린다.

▼ 그림 5-3 연결 리스트



- prev 포인터와 next 포인터로 앞과 뒤의 노드를 연결시킨 것이 연결 리스트이며, 연결 리스트는 싱글 연결 리스트, 이중 연결 리스트, 원형 이중 연결 리스트가 있다.
- 참고로 맨 앞에 있는 노드를 헤드 라고 한다.

  - **싱글 연결 리스트**: next 포인터만 가집니다.
  - **이중 연결 리스트**: next 포인터와 prev 포인터를 가집니다.
  - **원형 이중 연결 리스트**: 이중 연결 리스트와 같지만 마지막 노드의 next 포인터가 헤드 노드를 가리키는 것을 말합니다.

5.2.2 배열

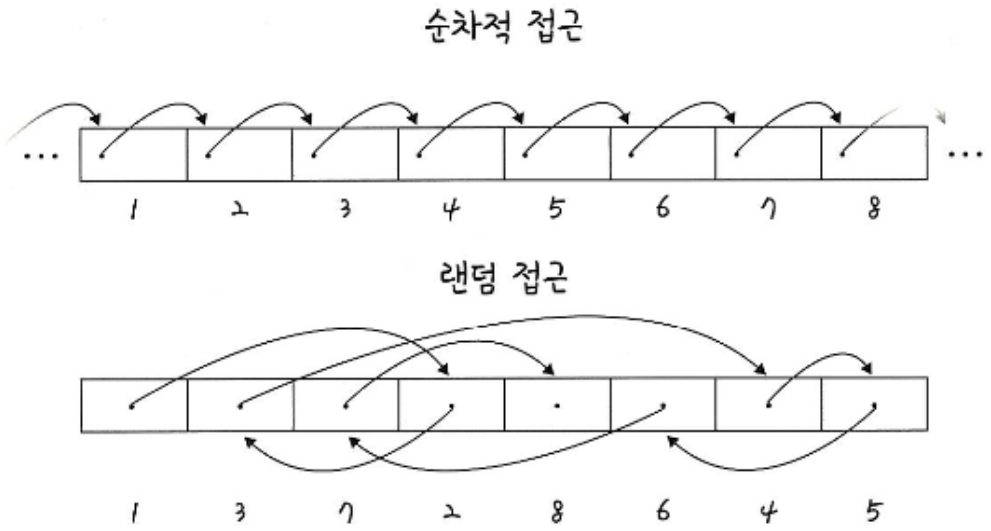
- 배열은 같은 타입의 변수들로 이루어져 있고, 크기가 정해져 있으며, 인접한 메모리 위치에 있는 데이터를 모아놓은 집합이다.
- 중복을 허용하고 순서가 있다.
- 탐색에 O(1)이 되어 랜덤 접근이 가능하다.
- 삽입과 삭제는 O(n)이 걸린다. 따라서 추가와 삭제를 많이 하는 것은 연결 리스트, 탐색을 많이 하는 것은 배열로 하는 것이 좋다.

- 배열은 인덱스에 해당하는 원소를 빠르게 접근해야 하거나, 간단하게 데이터를 쌓고 싶을때 사용한다.

랜덤접근과 순차적 접근

- 직접 접근이라고 하는 랜덤 접근은 동일한 시간에 배열과 같은 순차적인 데이터가 있을 때 임의의 인덱스에 해당하는 데이터에 접근할 수 있는 기능.
- 이는 데이터를 저장된 순서대로 검색해야 하는 순차적 접근과는 반대이다.

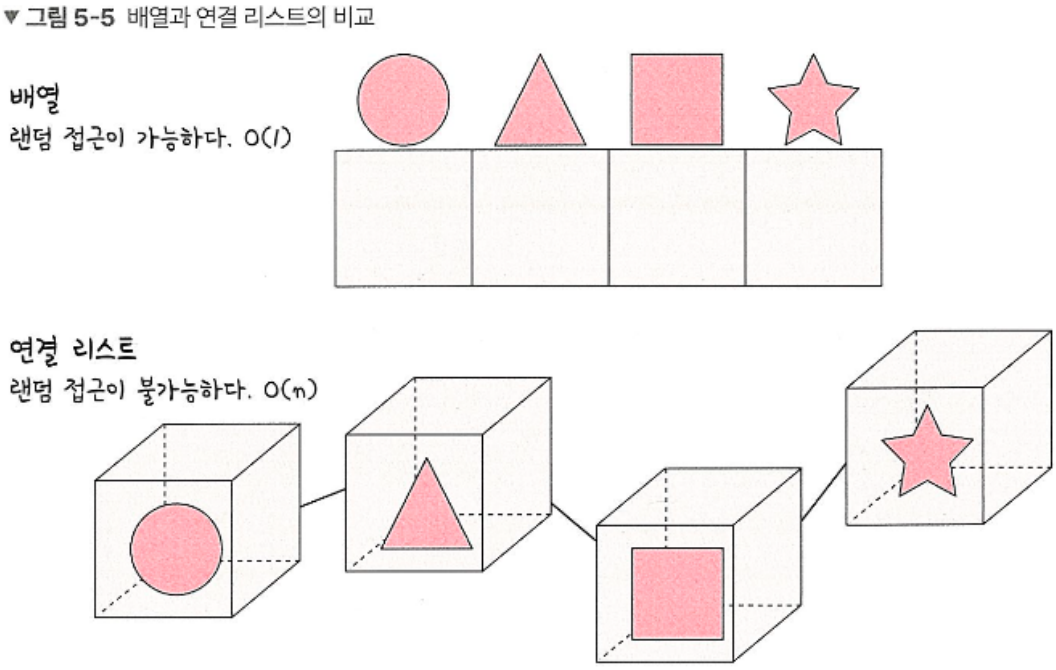
▼ **그림 5-4** 랜덤 접근과 순차적 접근



배열과 연결 리스트 비교

- 배열은 상자를 순서대로 나열한 데이터 구조이며, 몇번째 상자인지만 알면 해당 상자의 요소를 끄집어낼 수 있다.
- 연결 리스트는 상자를 선으로 연결한 형태의 데이터 구조이며 상자 안의 요소를 알기 위해서는 하나씩 상자 내부를 확인해봐야 한다는 점이 다르다.

▼ 그림 5-5 배열과 연결 리스트의 비교

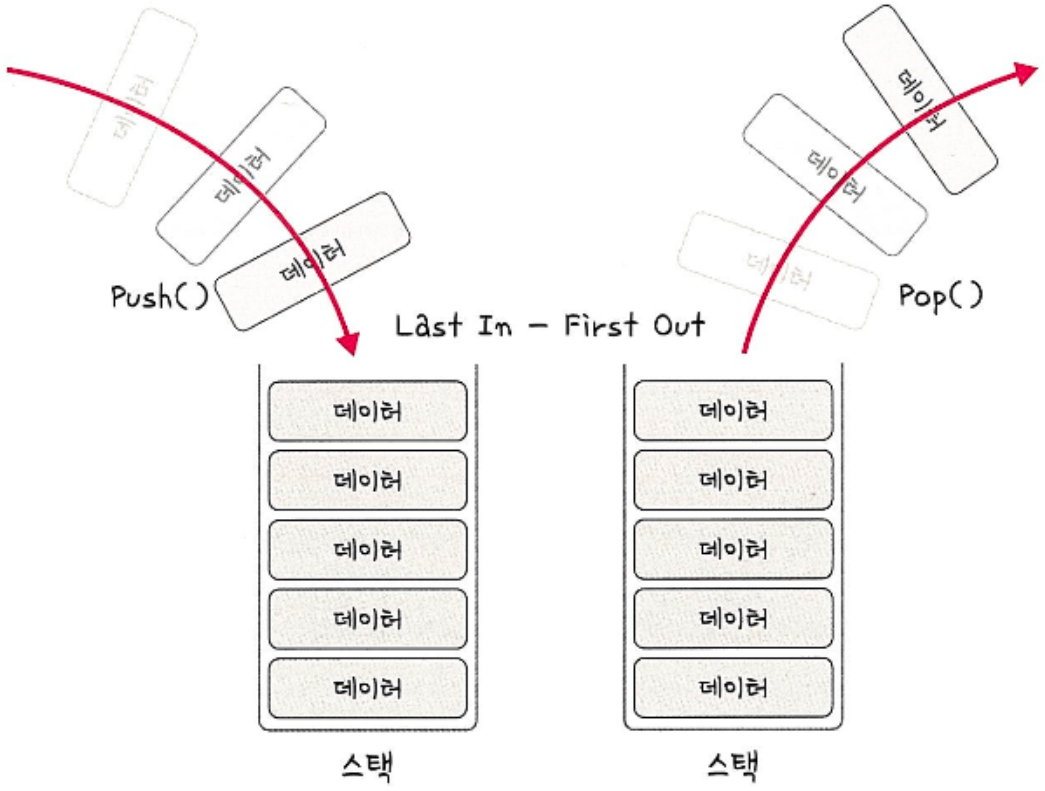


- 탐색은 배열이 빠르고, 연결 리스트는 느리다.
- 배열의 경우 그저 상자 위에 있는 요소를 탐색하면 되는 반면에, 연결 리스트는 상자를 열어야하고 주어진 선을 기반으로 순차적으로 열어야 한다.

5.2.4 스택

- 스택은 가장 마지막으로 들어간 데이터가 가장 첫번째로 나오는 성질 LIFO을 가진 자료 구조이다.
- 재귀적인 함수, 알고리즘에 사용되며 웹 브라우저 방문 기록등에 쓰인다.
- 삽입 및 삭제에 O(1) 탐색에 O(n)이 걸린다.
-

▼ 그림 5-7 스택



5.2.5 큐

- 큐는 먼저 집어넣은 데이터가 먼저 나오는 성질 FIFO을 지닌 자료 구조이며, 나중에 집어 넣은 데이터가 먼저 나오는 스택과는 반대되는 개념을 가졌다.
- 삽입 및 삭제에 O(1), 탐색에 O(n)이 걸린다.
- CPU작업을 기다리는 프로세스, 스레드 행렬 또는 네트워크 접속을 기다리는 행렬, 너비 우선 탐색, 캐시 등에 사용된다.

▼그림 5-8 큐