


1. 순차적 자료구조의 종류

🕒 작성일	@August 31, 2021 6:08 PM
👤 작성자	 김현빈

1. 배열 & 리스트

- 배열과 리스트는 각각의 원소들이 연속적으로 이어져 있다. 여기서 가장 큰 특징 Index로 임의의 원소를 접근한다는 것이다.

만약 $A = [3, 2, -1, 5, 7]$ 이라면 우리는 $A[0]$, $A[1]$ ~, $A[4]$ 와 같이 $[]$ 안에 인덱스를 사용하여 위치를 지정한다

- 우리는 연산자가 주어진다 $\rightarrow [] A[2] = -1$
- 우리는 삽입 연산자(append, insert)가 주어진다.
- 삭제 연산자(pop, remove)

Cf)

append와 pop 연산자는 마지막 자리의 수를 제거하거나 추가하기 때문에 배열의 이동이 필요 없다. 따라서 이들은 $O(1)$ 수행 시간이 걸린다.

반면, insert와 remove는 여러 요소가 연속적으로 나열되어 있는 원소 중 하나를 빼거나 더하기 때문에 다른 원소들이 옆자리로 이동해야 된다. 따라서 이들은 $O(n)$ 수행 시간이 걸린다.

2. Stack, Queue, Dequeue

- 제한된 접근(삽입, 삭제)만 허용한다.

배열과 리스트는 인덱스만 알면 자유롭게 삽입과 삭제를 할 수 있었다. 하지만 이와 다르게 Stack, Queue, Dequeue는 삽입과 삭제를 제한한다.

STACK: LIFO(Last In First Out)

Stack은 가장 늦게 들어간 것이 가장 먼저 나온다. 이는 우리가 책을 차곡차곡 쌓아 올린 것과 같다.

우리는 책을 쌓아 올리는 행위(연산)를 Push라고 하고, 책을 빼낼 때(삭제)는 pop 연산을 사용한다.

Queue: FIFO (First In First Out)

Queue는 선착순과 같다. 이는 가장 먼저 들어간 것이 가장 먼저 나온다.

Deque: Stack + Queue

이는 Stack과 Queue의 모든 삽입과 삭제를 허용한다.

3. Linked List(연결 리스트)

이는 단지 값들이 순차적으로 연결되어 있는 것이다. 배열과 리스트는 연속된 메모리 공간에 차례대로 나열되어 있다. 하지만 연결 리스트는 서로 연속되지 않고 독립적으로 띄엄띄엄 있는 메모리 공간에 원소들이 저장되어 있다. 따라서 각 원소들은 다음 원소의 위치를 알기 위해서 메모리 주소(링크)를 가지고 있다. 예를 들어 첫번째 원소가 3이라면, 3번 자리의 메모리에 내용물 3과 그 다음 원소 2인 값의 메모리 주소를 같이 가지고 있다. 따라서 x번째 원소를 알고 싶다면 처음부터 x까지 링크를 따라 가야된다는 단점이 있다. 하지만 이는 리스트와는 다른 장점이 있다.