자료 구조

(스택, 큐)

https://youtu.be/Xh_2IQ0xQ4w

송경주

목차

스택 (stack)

- 스택이란?

데이터를 쌓는 방식. (아래서 위로 책을 쌓듯이 쌓임)

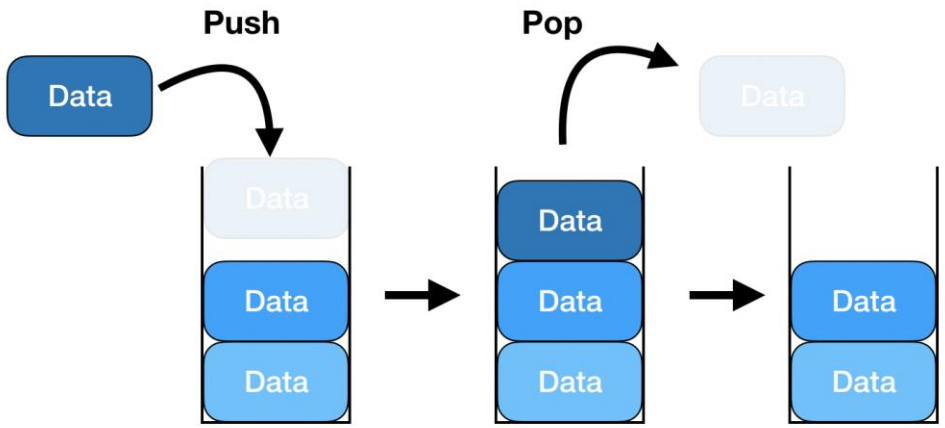
최근 값이 가장 먼저 나간다. (LIFO : Last-In First-Out)

- 스택 구현
  1. 배열
  2. 연결리스트

스택의 삽입과 삭제

삽입 연산 : push
삭제 연산 : pop
반환 연산 : peek

입력 순서 : A -> B -> C -> D -> E
출력 순서 : E -> D -> C -> B -> A



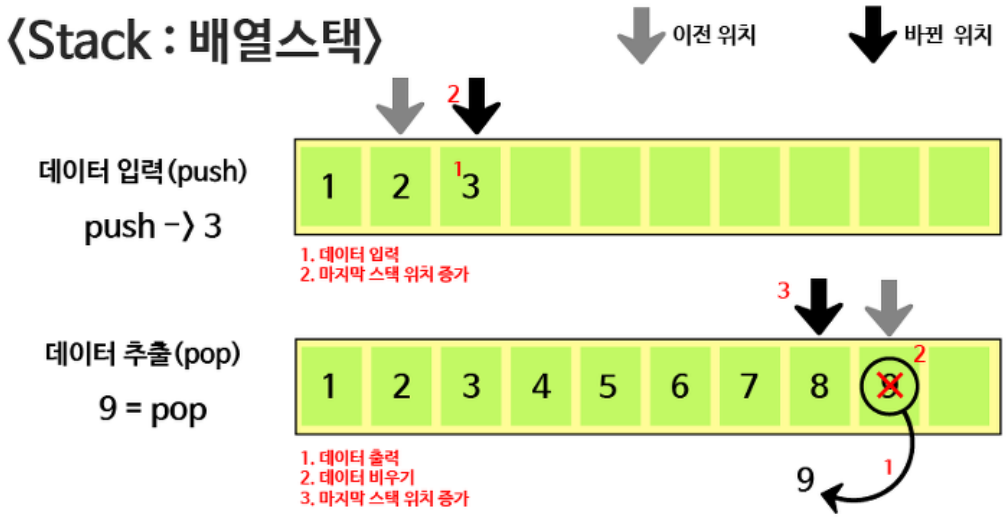
배열로 구현한 스택

-장점 : 구현이 쉽다.

-단점 : 크기가 고정된다

-스택은 마지막에 들어온 요소 (가장 최근에 들어온 요소)를 알아야 하므로 **top변수** 를 사용한다.

[입력과 출력하는 위치가 같으므로 변수는 1개]



연결리스트로 구현한 스택

-장점 : 크기 가변적 변경 가능.

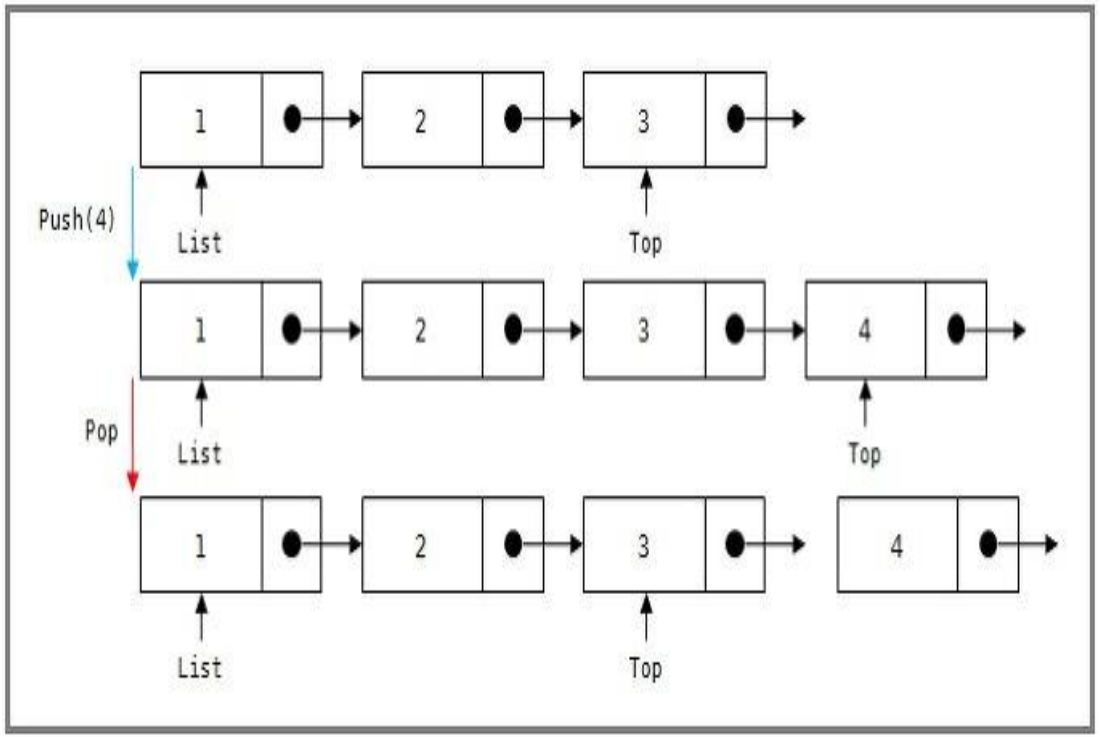
-단점 : 구현이 복잡, 메모리 공간 차지

-삽입과 삭제가 일어나는 곳: top

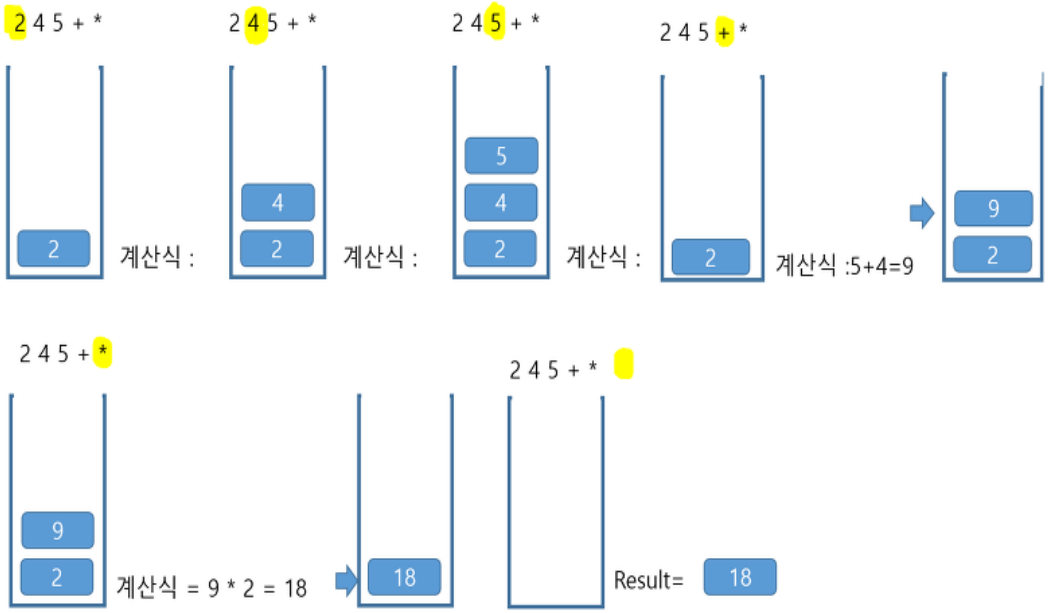
-스택은 마지막에 들어온 요소 (가장 최근에 들어온 요소)를 알아야 하므로 **top변수** 를 사용한다.

[입력과 출력하는 위치가 같으므로 변수는 1개]

※ 연결리스트에서의 스택에서는 top변수가 다음 노드를 가리키는 포인터로 선언된다.



스택의 활용



1) 2는 피연산자 이므로 스택에 push한다.

2) 4는 피연산자 이므로 스택에 push한다.

3) 5는 피연산자 이므로 스택에 push한다.

4) +는 연산자. 스택에서 pop()을 2번 수행해 각각 값을 변수에 담고 더한다. 이후 스택에 결과 값을 push한다.

5) *는 연산자. 스택에서 pop()을 2번 수행해 각각 값을 변수에 담고 곱한다. 이후 스택에 결과값을 push한다.

6) 수식이 끝났으므로 스택에 마지막 남은 값을 pop한다. 그 값이 최종 결과 값이다.

2 4 5 + * ➡ 2 (5+4) * ➡ 2 9 * ➡ (9 * 2) ➡ 18

큐 (queue)

- 큐란?

데이터를 줄세우는 방식. (데이터가 들어오는 곳과 나가는 곳이 다르다)

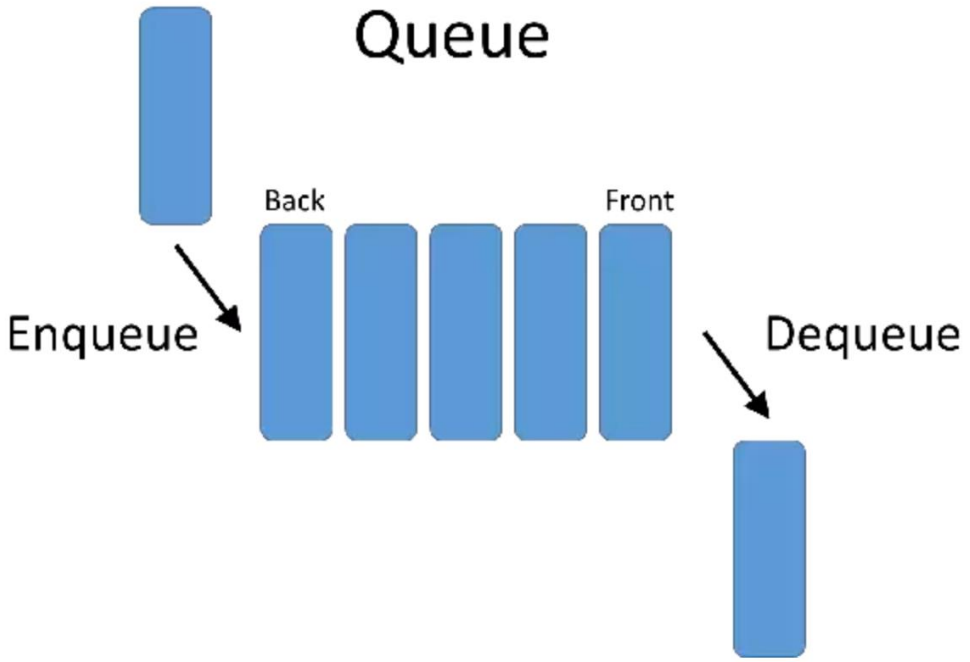
가장 먼저 들어온 데이터가 가장 먼저 나간다. (FIFO : Frist-In First-Out)

- 큐의 구현
  1. 배열
  2. 연결리스트

큐의 삽입과 삭제

삽입 연산 : enqueue
삭제 연산 : dequeue
반환 연산 : peek

입력 순서 : A -> B -> C -> D -> E
출력 순서 : A -> B -> C -> D -> E



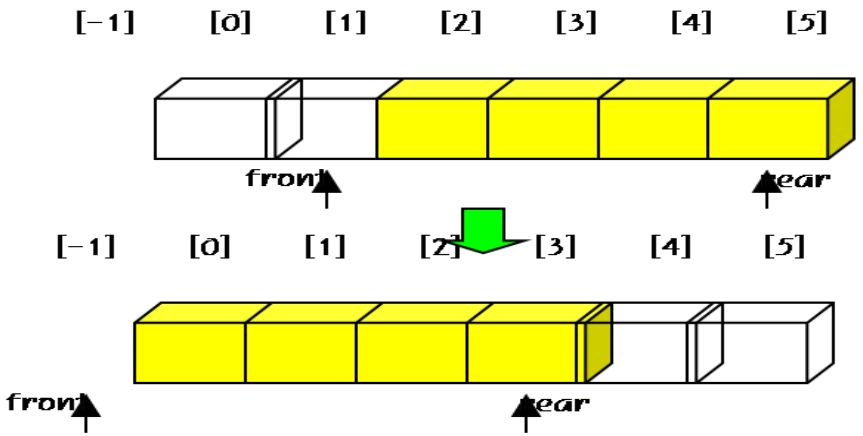
배열로 구현한 큐 (선형 큐)

-장점 : 구현이 쉽다.

-단점 : 크기가 고정된다,

값을 사용하다 보면 앞부분이 비어도 사용 불가

→주기적으로 이동시켜줘야 한다 (원형 큐로 문제 해결)



-삽입이 일어나는 곳 : 후단(rear)

-삭제가 일어나는 곳: 전단(front)

[삽입의 위치와 삭제의 위치를 알아야 하므로 포인터는 2개]

배열로 구현한 큐 (원형 큐)

-장점 : 구현이 쉽다.

-단점 : 크기가 고정된다,

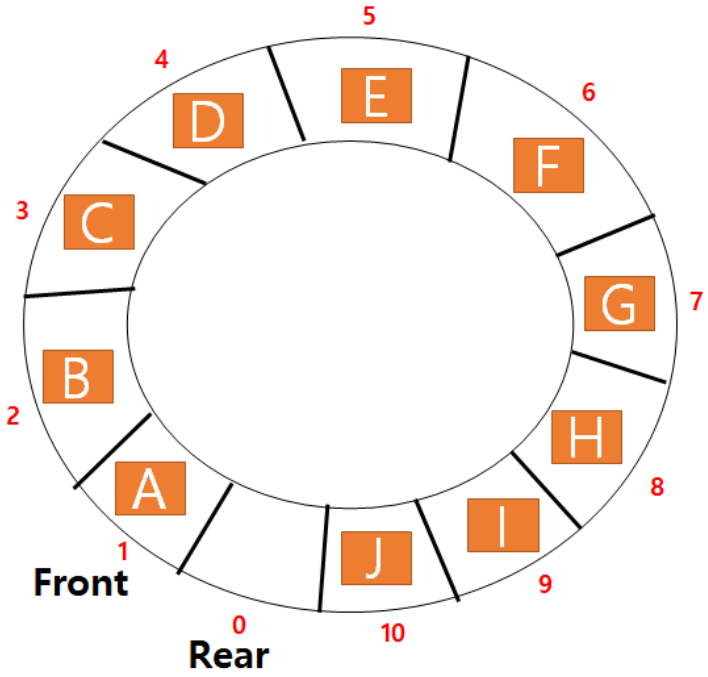
값을 사용하다 보면 앞부분이 비어도 사용 불가

→주기적으로 이동시켜줘야 한다 (원형 큐로 문제 해결)

-삽입이 일어나는 곳 : 후단(rear) - 마지막 요소를 가리킴

-삭제가 일어나는 곳: 전단(front) – 첫번째 요소의 하나 앞을 가리킴

[삽입의 위치와 삭제의 위치를 알아야 하므로 변수는 2개]



배열로 구현한 큐 (원형 큐)

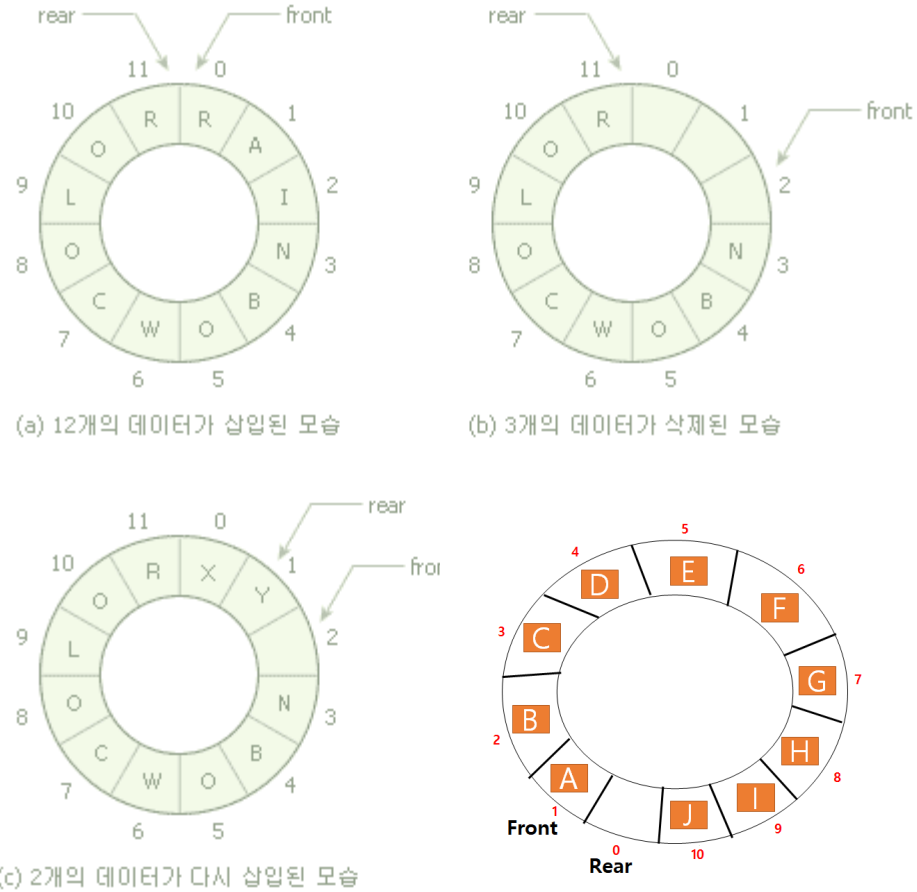
-삽입이 일어나는 곳 : 후단(rear) - 마지막 요소를 가리킴

-삭제가 일어나는 곳: 전단(front) – 첫번째 요소의 하나 앞을 가리킴

공백상태와 포화상태를 구분하기 위해!!

Front == rear 이면 공백상태 고, front가 rear 보다 하나 앞에 있으면 포화상태

(front가 첫번째 요소 하나 앞을 가리키면 rear과 front 사이에 공백이 생기며 이를 이용함.)



연결리스트로 구현한 큐 (연결된 큐, linked queue)

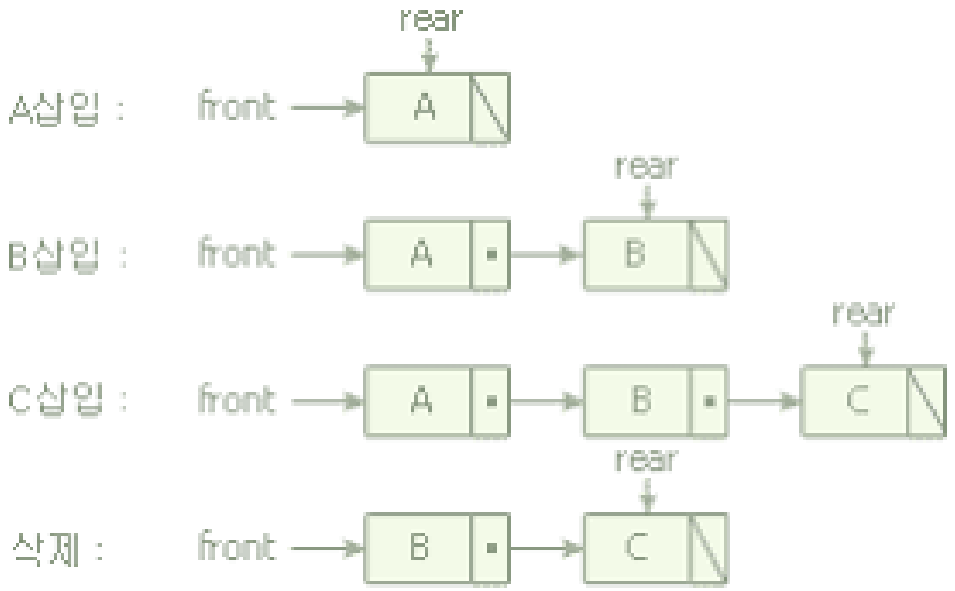
-장점 : 크기 가변적 변경 가능.

-단점 : 구현이 복잡, 메모리 공간 차지

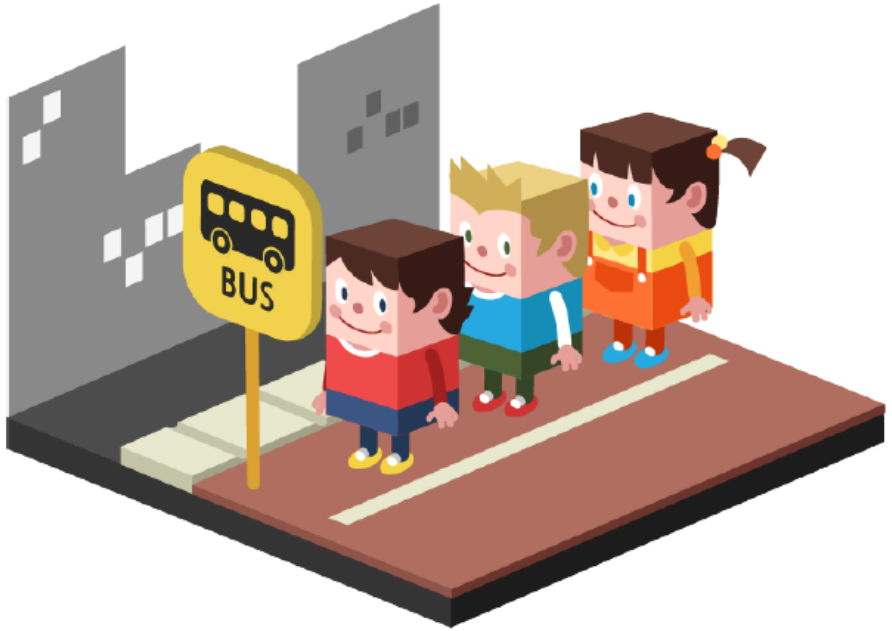
-삽입이 일어나는 곳 : 후단(rear)

-삭제가 일어나는 곳: 전단(front)

[삽입의 위치와 삭제의 위치를 알아야 하므로 변수는 2개]



큐의 활용



감사합니다 :-)