

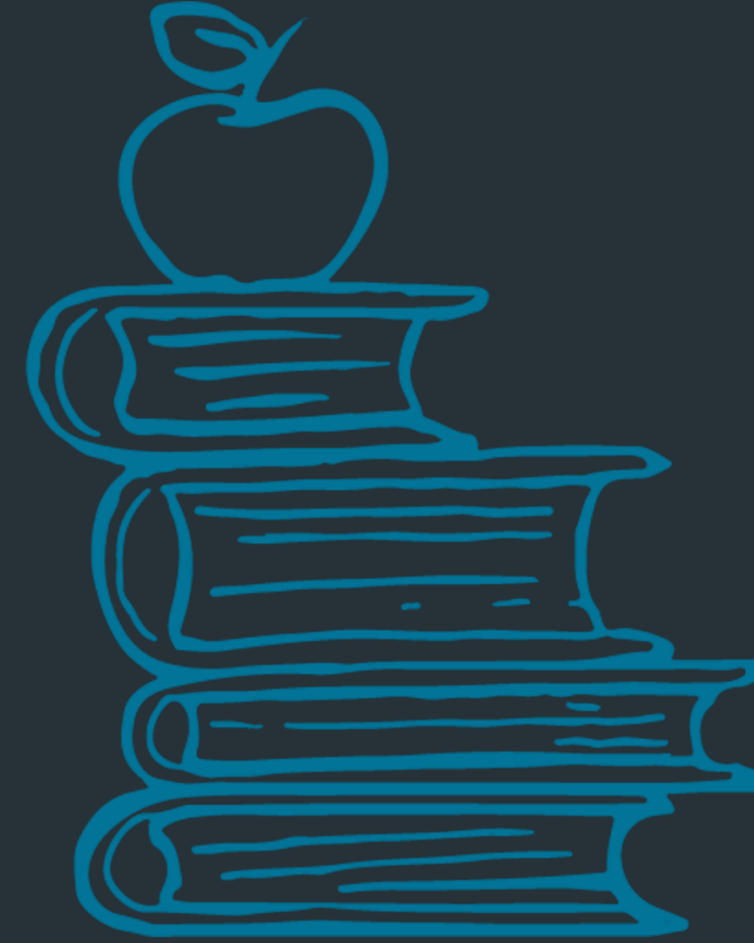
# 알튜비튜

## 스택, 큐, 덱

오늘은 STL에서 제공하는 container adaptor인 stack과 queue 그리고 sequence container인 deque에 대해 알아봅니다.  
가장 대표적이면서 가장 중요하기도 한 자료구조들 입니다.

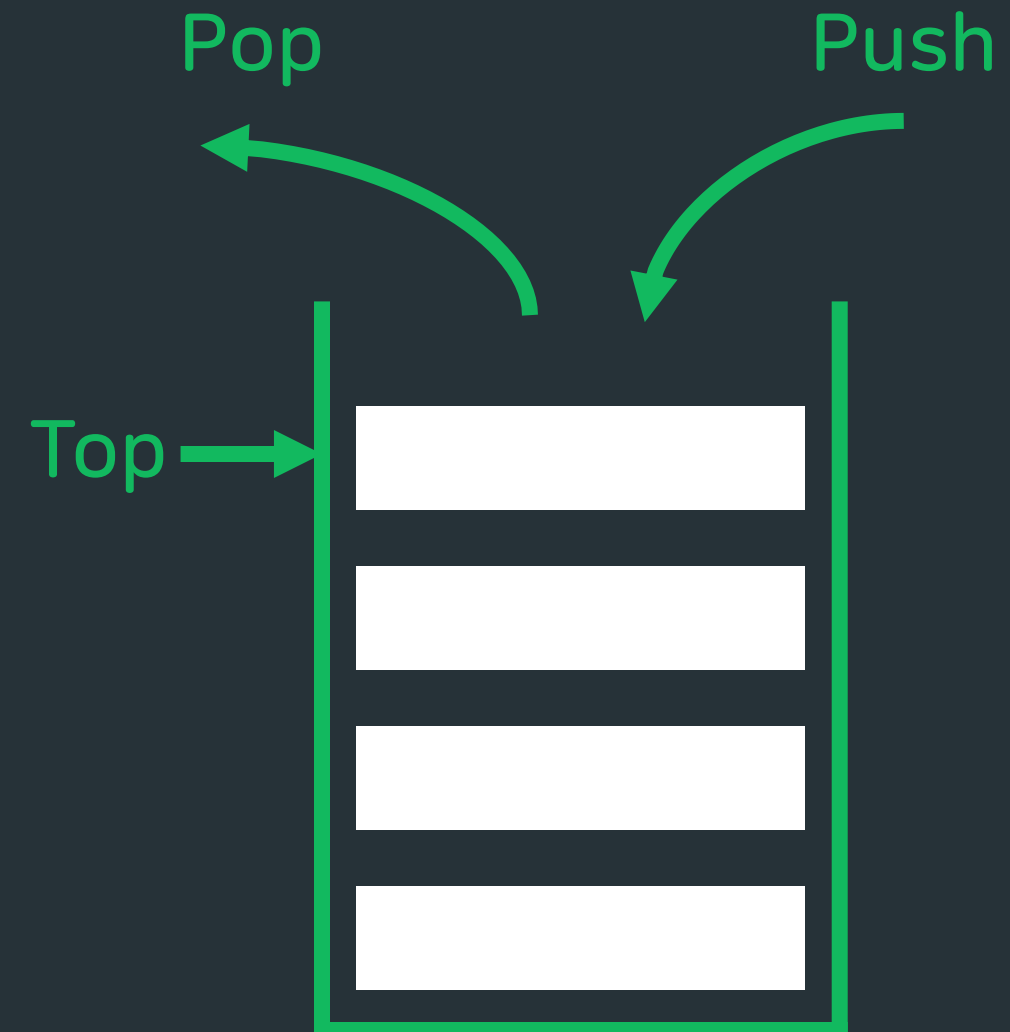
어디까지 먹어봤니?  
내가 모르는 프링글스의 세상!

프링글스

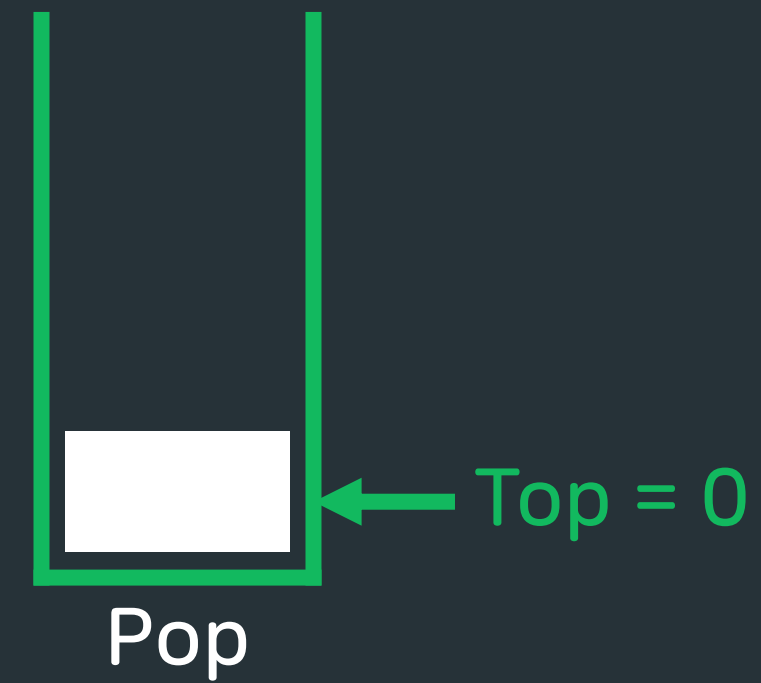
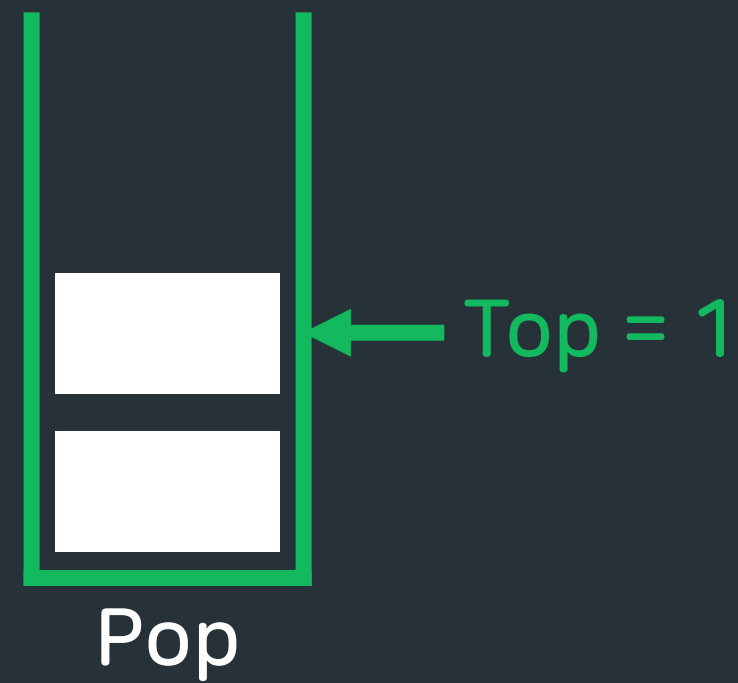
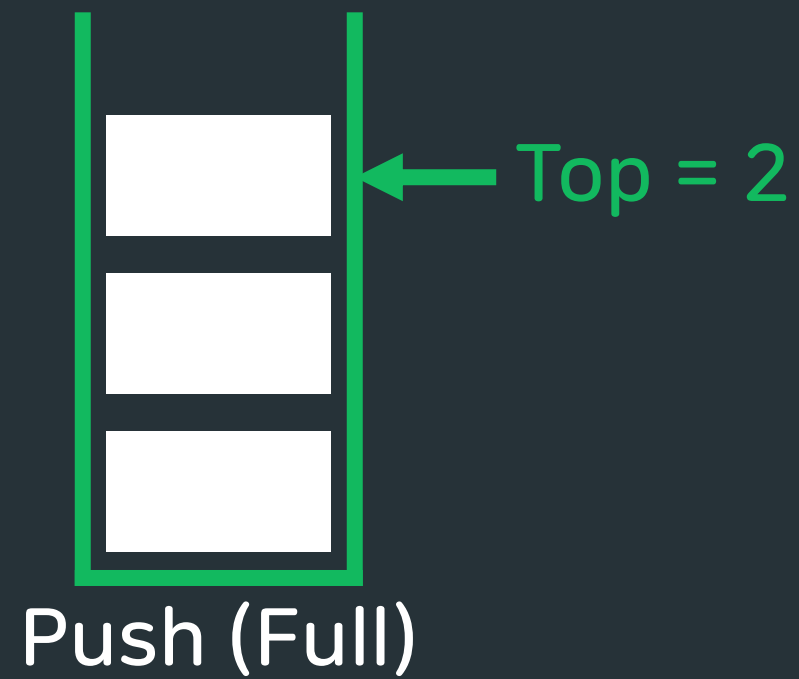
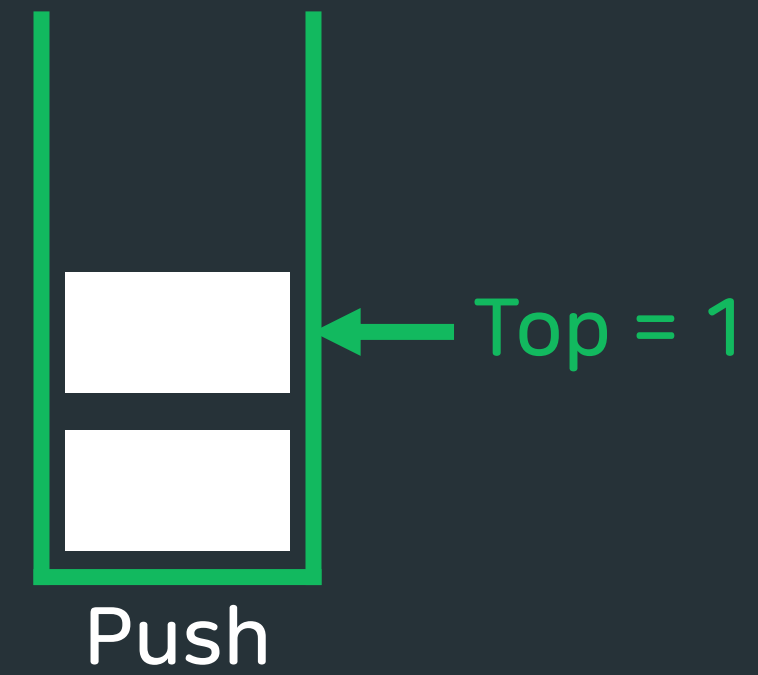


## Stack

- LIFO (Last In, First Out)
- 자료의 맨 끝 위치에서만 모든 연산이 이루어짐
- 모든 연산에 대한 시간 복잡도는  $O(1)$
- 연산이 이루어지는 위치를 *top*이라고 부름
- 삽입은 *push*, 삭제는 *pop*



# 배열로 크기 3의 스택 구현하기



## /<> 10828번 : 스택 - Silver 4

### 문제

- 다음의 명령을 처리하는 스택 프로그램 만들기
  1. push X : 정수 X를 스택에 **삽입**
  2. pop : 스택에서 가장 위에 있는 정수를 **빼고**, 출력. 스택이 비었다면 -1 출력
  3. size : 스택에 들어있는 정수의 **개수** 출력
  4. empty : 스택이 **비었으면 1, 아니라면 0**을 출력
  5. top : 스택의 가장 위에 있는 정수를 **출력**. 스택이 비었다면 -1 출력

### 제한 사항

- 명령의 수 N의 범위는  $1 \leq N \leq 10,000$
- 명령과 함께 주어지는 정수 k의 범위는  $1 \leq k \leq 100,000$





Search:  Go

Not logged in  
[register](#) [log in](#)

C++

Information  
Tutorials  
Reference  
Articles  
Forum

Reference

C library:

Containers:

<array>  
<deque>  
<forward\_list>  
<list>  
<map>  
<queue>  
<set>  
<stack>  
<unordered\_map>  
<unordered\_set>  
<vector>

Input/Output:  
Multi-threading:  
Other:

<stack>

stack

stack

stack::stack

member functions:

stack::emplace  
stack::empty  
stack::pop  
stack::push  
stack::size  
stack::swap

You were redirected to [cplusplus.com/stack](#) || See search results for: "stack"

class template  
**std::stack**

```
template <class T, class Container = deque<T> > class stack;
```

**LIFO stack**  
Stacks are a type of container adaptor, specifically designed to operate in a LIFO context (last-in first-out), where elements are inserted and extracted only from one end of the container.  
**stacks** are implemented as *container adaptors*, which are classes that use an encapsulated object of a specific container class as its *underlying container*, providing a specific set of member functions to access its elements. Elements are *pushed/popped* from the "back" of the specific container, which is known as the *top* of the stack.  
The underlying container may be any of the standard container class templates or some other specifically designed container class. The container shall support the following operations:

- empty
- size
- back
- push\_back
- pop\_back

The standard container classes `vector`, `deque` and `list` fulfill these requirements. By default, if no container class is specified for a particular stack class instantiation, the standard container `deque` is used.

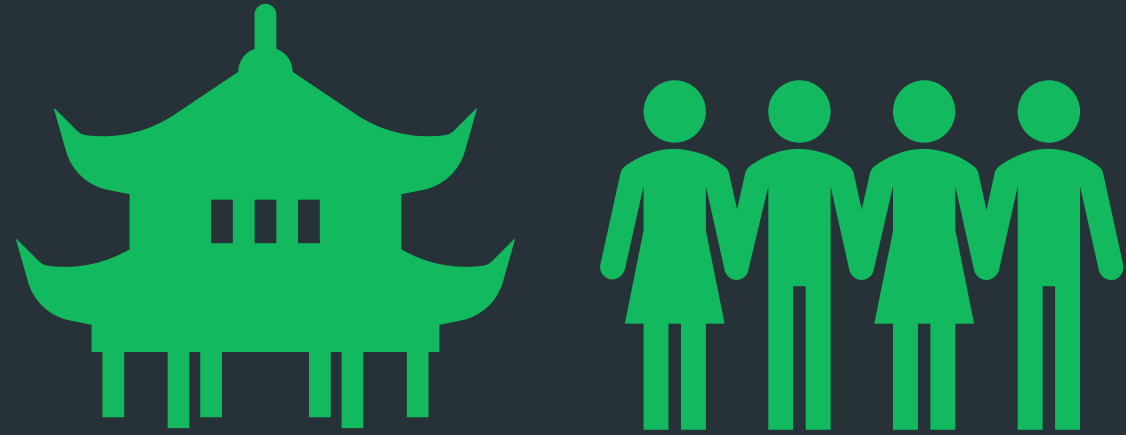
**Template parameters**

T

Type of the elements.  
Aliased as member type `stack::value_type`.

Container

Type of the internal *underlying container* object where the elements are stored.  
Its `value_type` shall be T.  
Aliased as member type `stack::container_type`.





**Magic pass**

**와일드 정글**

기존의 시시함은 가라! 이것이 리얼 어드벤처!

110cm 미만

현장대기  
45분

**<50 ~ 54세 연령층(1967.1.1. ~ 1971.12.31. 출생)>**

○ (예약기간)

53~54세(67~68년생) : 7.19.(월) 20시 ~ 7.20.(화) 18시

50~52세(69~71년생) : 7.20.(화) 20시 ~ 7.21.(수) 18시

50~54세(67~71년생) : 7.21.(수) 20시 ~ 7.24.(토) 18시

\* 예방접종센터 예약은 7.21.(수) 20시 ~ 7.24.(토) 18시

○ (접종기간) 8.16.(월) ~ 8.28.(토)

**<60 ~ 74세 고령층 중 사전예약 후 미접종자(1947.1.1. ~ 1961.12.31. 출생)>**

○ (예약기간) 7.14.(수) 20시 ~ 7.24.(토) 18시

○ (접종기간) 7.26.(월) ~

**서비스 접속대기 중입니다.**

예상대기시간 : 3시간 41분 19초

고객님 앞에 172442 명, 위에 7713 명의 대기자가 있습니다.  
현재 접속 사용자가 많아 대기 중이며, 잠시만 기다려시면  
서비스로 자동 접속 됩니다.

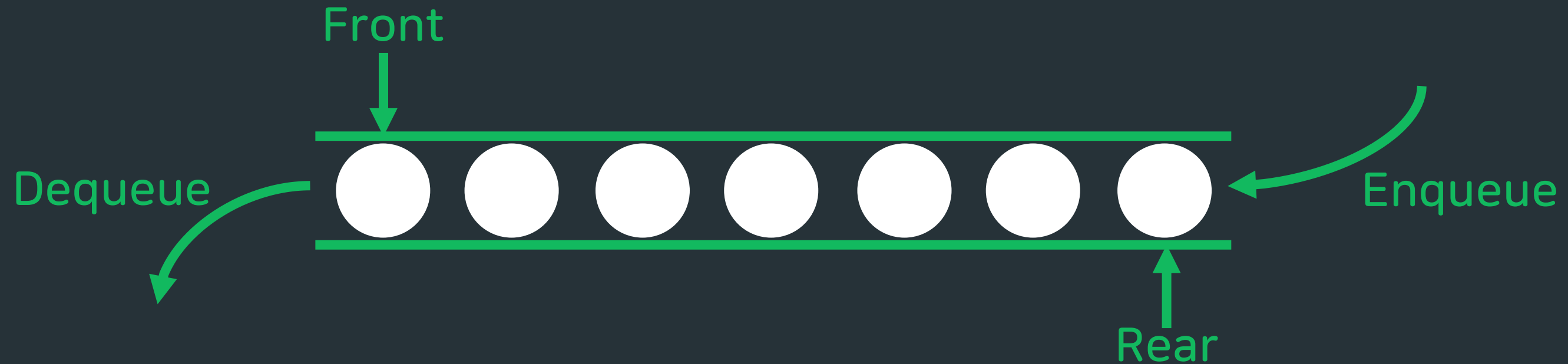
※ 재 접속하시면 대기시간이 더 길어집니다. [중지]

코로나19 예방접종 사전예약 시스템 | 개인정보처리방침 | 질병관리청 바로가기

(28159) 충북 청주시 흥덕구 오송읍 오송생명2로 187 오송보건의료행정타운내 질병관리청

COPYRIGHT © 2021 질병관리청 ALL RIGHTS RESERVED.

OPEN

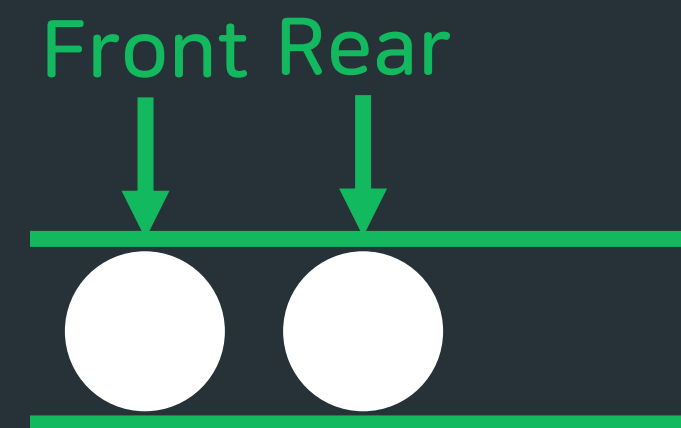
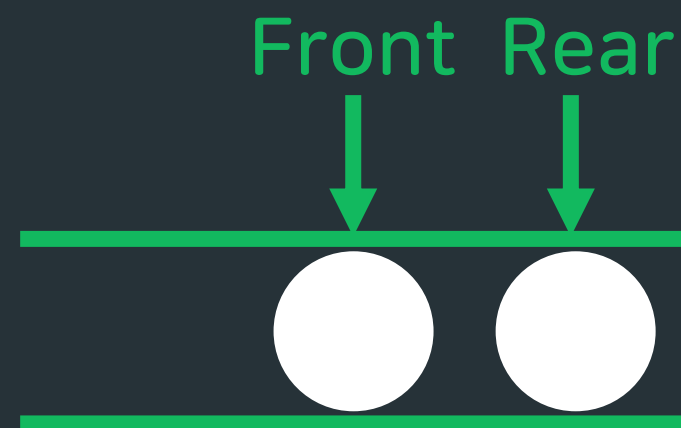
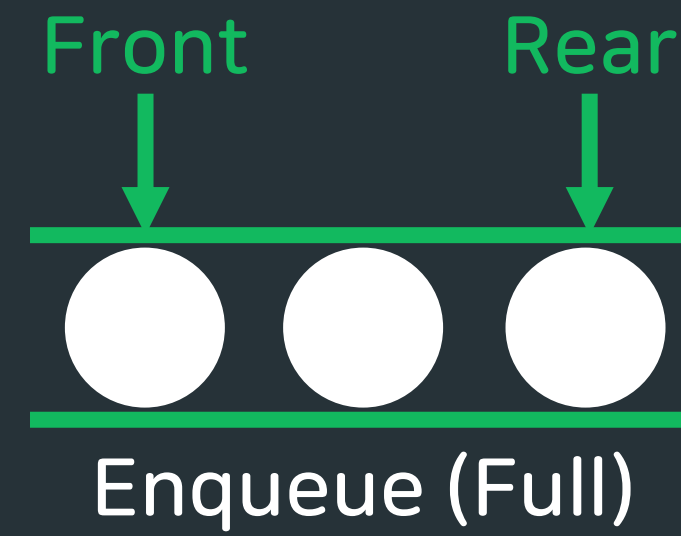
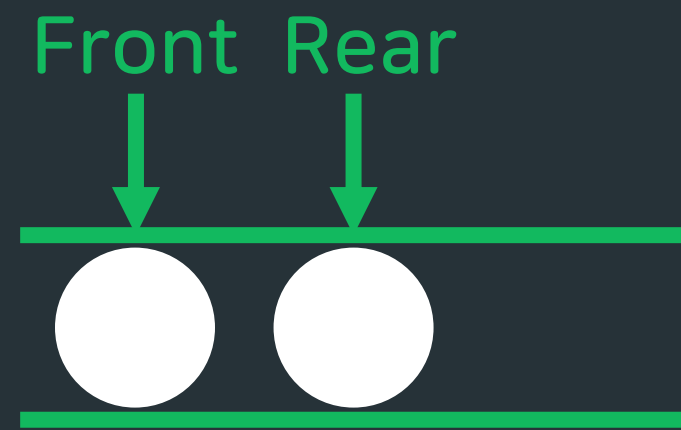


## Queue

- FIFO (First In, First Out)
- 자료의 왼쪽 끝 위치에서 삭제, 오른쪽 끝 위치에서 삽입 연산이 이루어짐
- 모든 연산에 대한 시간 복잡도는  $O(1)$
- 삭제가 이루어지는 위치를 *front*, 삽입이 이루어지는 위치를 *rear*라고 부름
- 삽입은 *enqueue*, 삭제는 *dequeue*

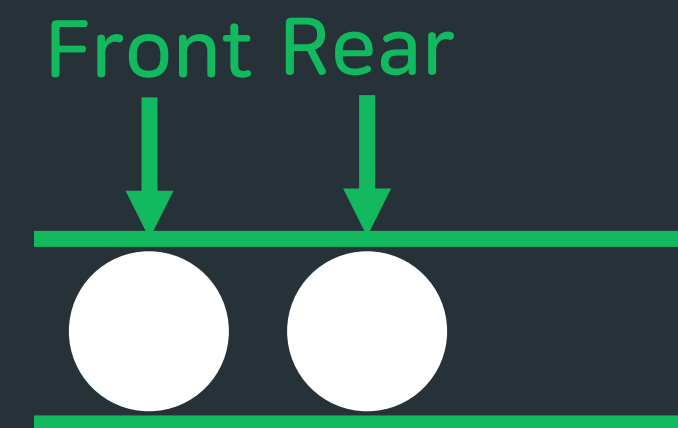
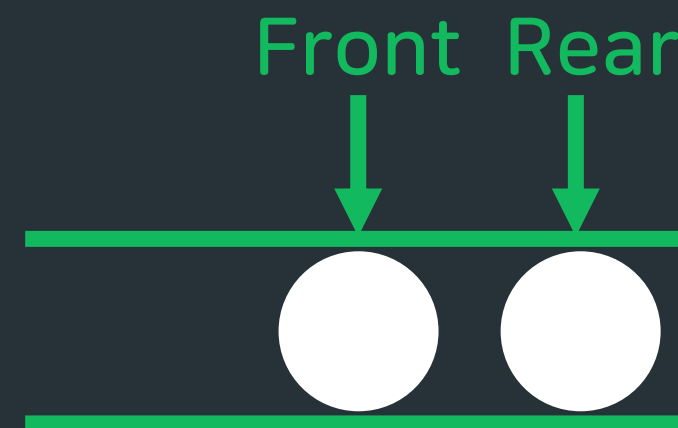
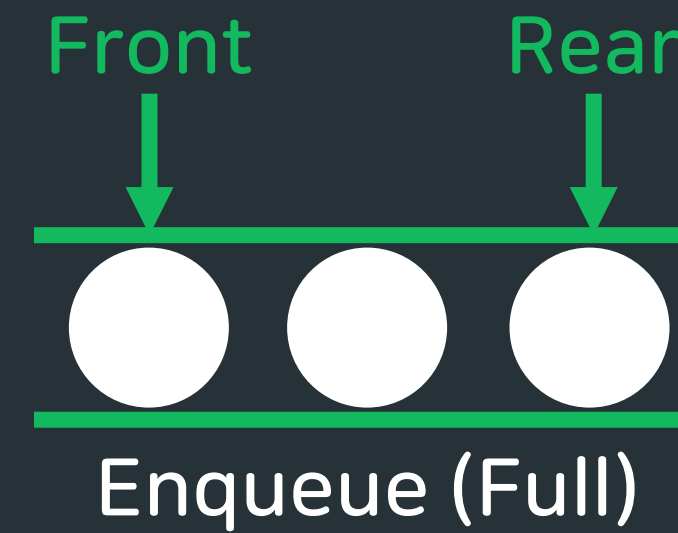
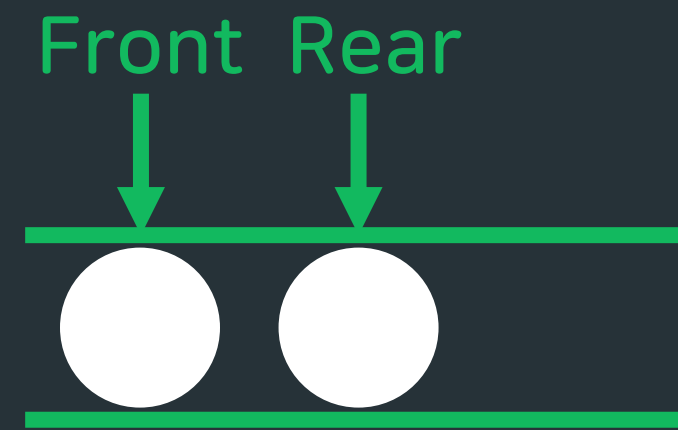


# 배열로 크기 3의 큐 구현하기?



Dequeue?

# 배열로 크기 3의 큐 구현하기?



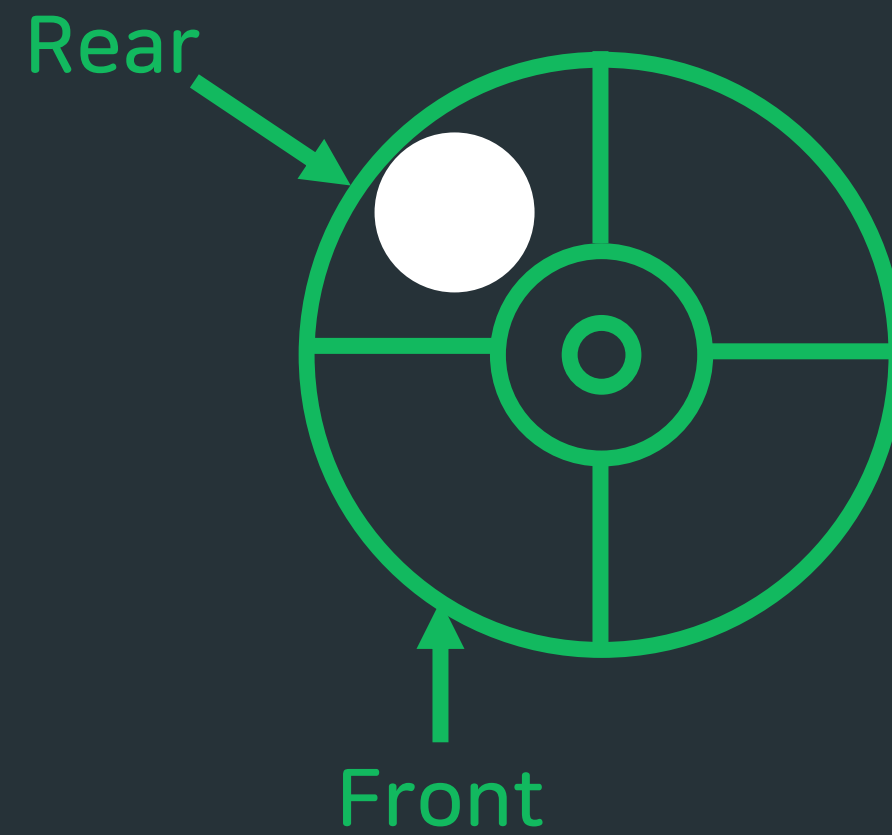
Dequeue?

Dequeue 연산마다 배열의 모든 원소를 한 칸씩 옮기는 건 비효율적이다!

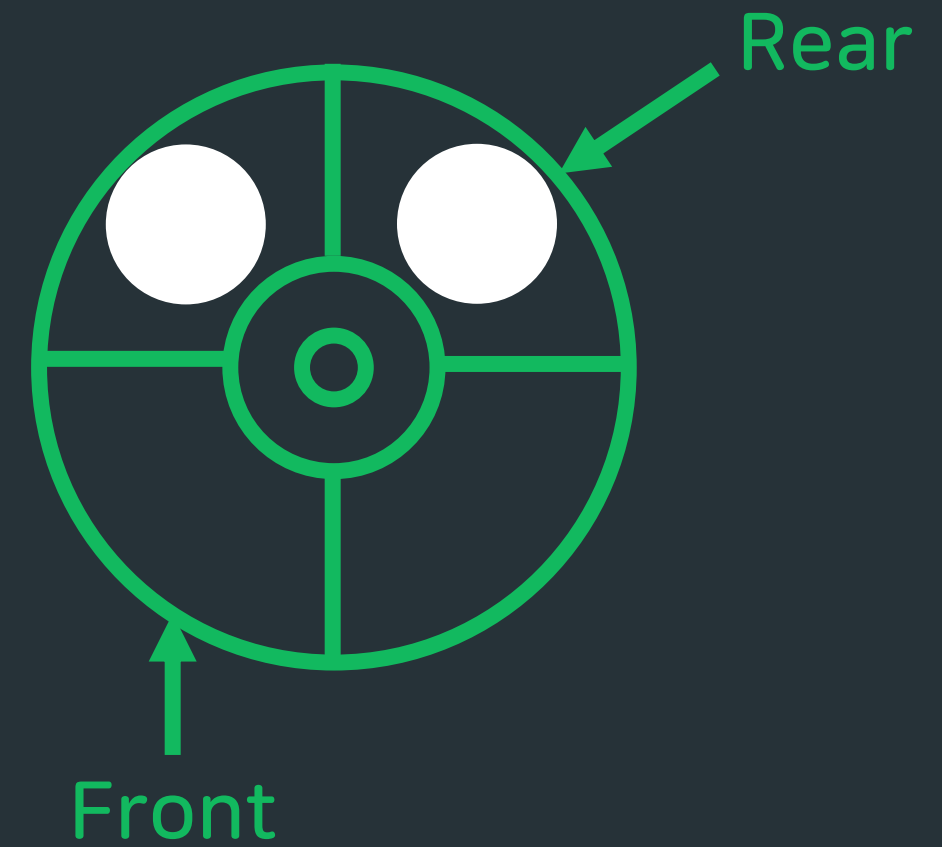
# 배열로 크기 3의 큐 구현하기



Empty

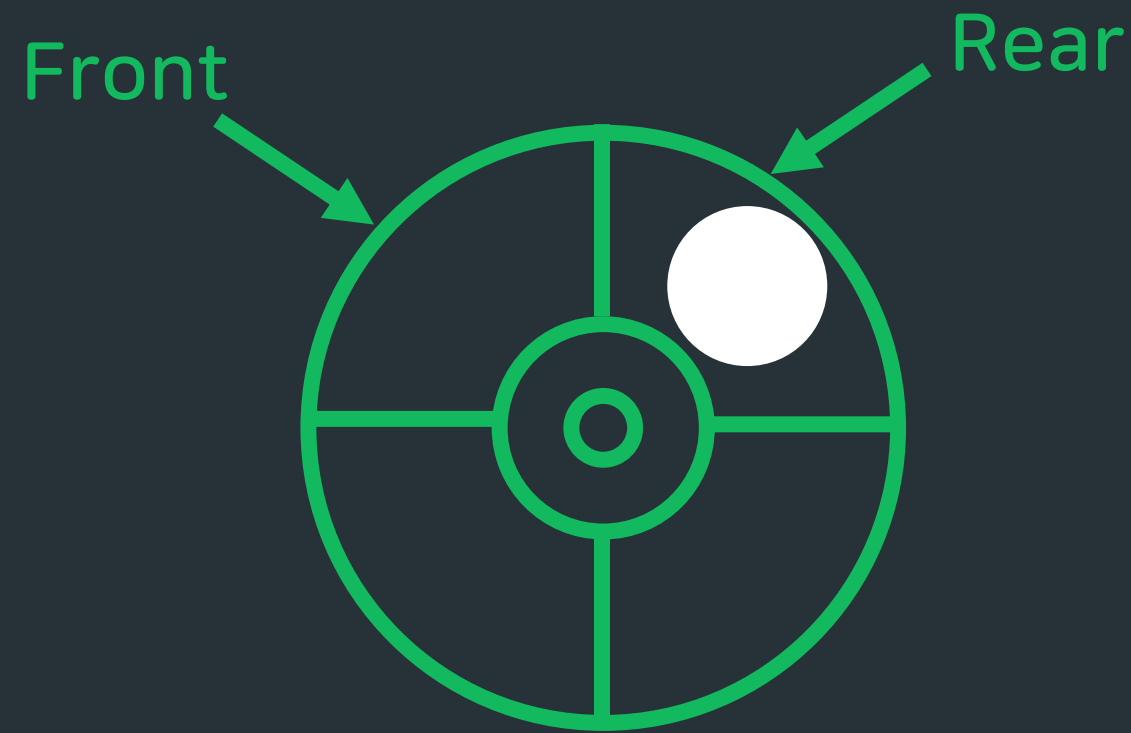


Enqueue

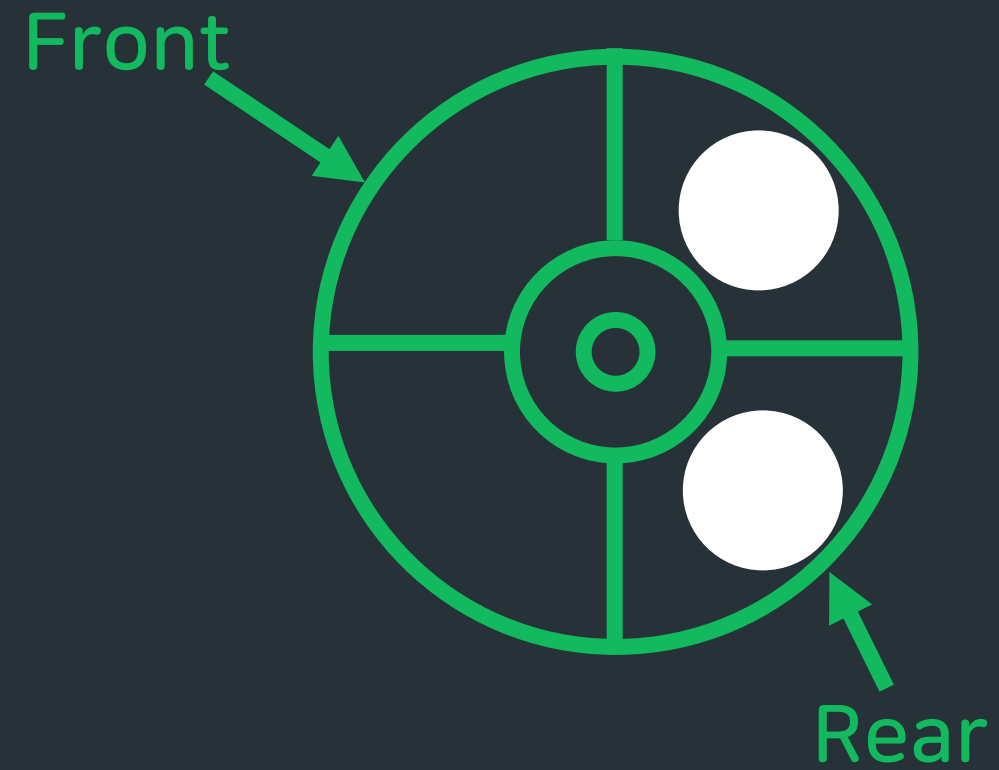


Enqueue

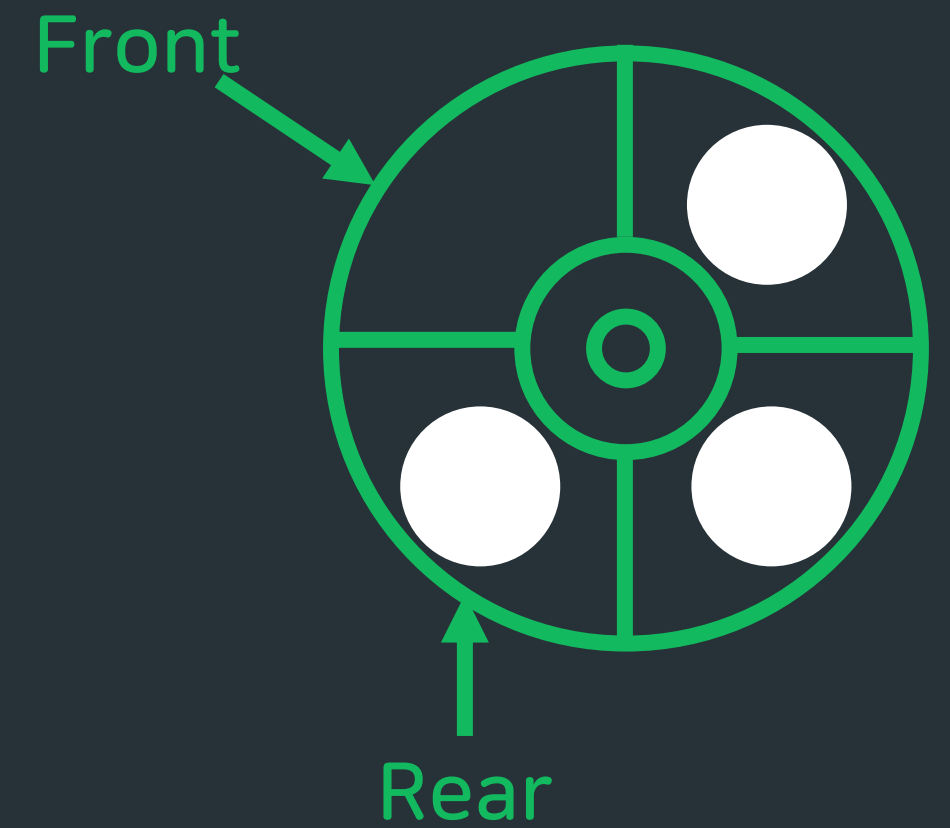
# 배열로 크기 3의 큐 구현하기



Dequeue



Enqueue



Enqueue (Full)

## /<> 10845번 : 큐 - Silver 4


### 문제

- 다음의 명령을 처리하는 큐 프로그램 만들기
  1. push X : 정수 X를 큐에 **삽입**
  2. pop : 큐에서 가장 앞에 있는 정수를 **빼고**, 출력. 큐가 비었다면 -1 출력
  3. size : 큐에 들어있는 정수의 **개수** 출력
  4. empty : 큐가 **비었으면 1, 아니라면 0**을 출력
  5. front : 큐의 가장 앞에 있는 정수를 **출력**. 큐가 비었다면 -1 출력
  6. back : 큐의 가장 뒤에 있는 정수를 **출력**. 큐가 비었다면 -1 출력

### 제한 사항

- 명령의 수 N의 범위는  $1 \leq N \leq 10,000$
- 명령과 함께 주어지는 정수 k의 범위는  $1 \leq k \leq 100,000$





Search:

[Reference](#)
[<queue>](#)
[queue](#)

[register](#)
[log in](#)

C++

[Information](#)
[Tutorials](#)
[Reference](#)
[Articles](#)
[Forum](#)

Reference

C library:

Containers:

[<array>](#)
[<deque>](#)
[<forward\\_list>](#)
[<list>](#)
[<map>](#)
[<queue>](#)
[<set>](#)
[<stack>](#)
[<unordered\\_map>](#)
[<unordered\\_set>](#)
[<vector>](#)

Input/Output:

Multi-threading:

Other:

<queue>

[priority\\_queue](#)
[queue](#)

queue

queue::queue

member functions:

[queue::back](#)
[queue::emplace](#)
[queue::empty](#)
[queue::front](#)
[queue::pop](#)
[queue::push](#)
[queue::size](#)

You were redirected to [cplusplus.com/queue](#) || See search results for: "queue"

class template

std::queue

<queue>

```
template <class T, class Container = deque<T> > class queue;
```

FIFO queue

queues are a type of container adaptor, specifically designed to operate in a FIFO context (first-in first-out), where elements are inserted into one end of the container and extracted from the other.

queues are implemented as *containers adaptors*, which are classes that use an encapsulated object of a specific container class as its *underlying container*, providing a specific set of member functions to access its elements. Elements are *pushed* into the "back" of the specific container and *popped* from its "front".

The underlying container may be one of the standard container class template or some other specifically designed container class. This underlying container shall support at least the following operations:

- empty
- size
- front
- back
- push\_back
- pop\_front

The standard container classes `deque` and `list` fulfill these requirements. By default, if no container class is specified for a particular queue class instantiation, the standard container `deque` is used.

Template parameters

T

Type of the elements.  
Aliased as member type `queue::value_type`.

Container

Type of the internal *underlying container* object where the elements are stored.  
Its `value_type` shall be T.  
Aliased as member type `queue::container_type`.



## Deque

- Double-Ended Queue
- Stack + Queue
- 자료의 양 끝에서 연산이 이루어짐
- 모든 연산에 대한 시간 복잡도는  $O(1)$

## /<> 4949번 : 균형잡힌 세상 - Silver 4

### 문제

- 문자열이 주어졌을 때, 괄호의 짝이 잘 맞는지 판단

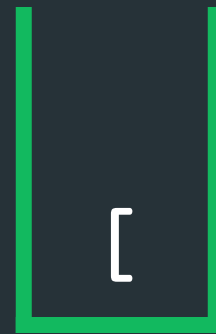
### 제한 사항

- 문자열의 길이는 100글자보다 작거나 같다

## Hint

1. 띄어쓰기까지 포함해서 입력을 받으려면 어떻게 해야 할까요?
2. ((([])))

# 괄호의 짝이 맞는 경우



[ first in ] ( first out ).



## 괄호의 짝이 맞는 경우

[ ← ']'와 짝

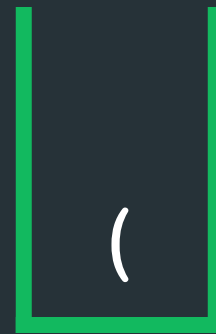
[ first in ] ( first out ).

# 괄호의 짝이 맞는 경우



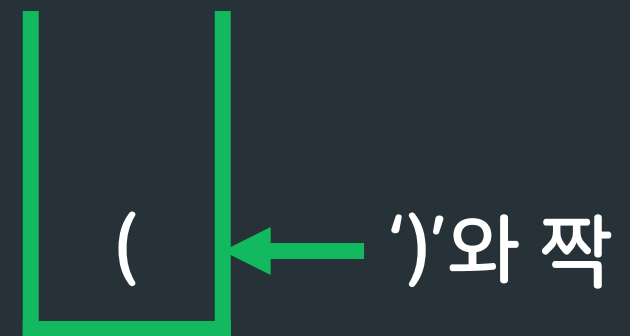
[ first in ] ( first out ).

# 괄호의 짝이 맞는 경우



[ first in ] ( first out ).

## 괄호의 짝이 맞는 경우



[ first in ] ( first out ).

# 괄호의 짝이 맞는 경우

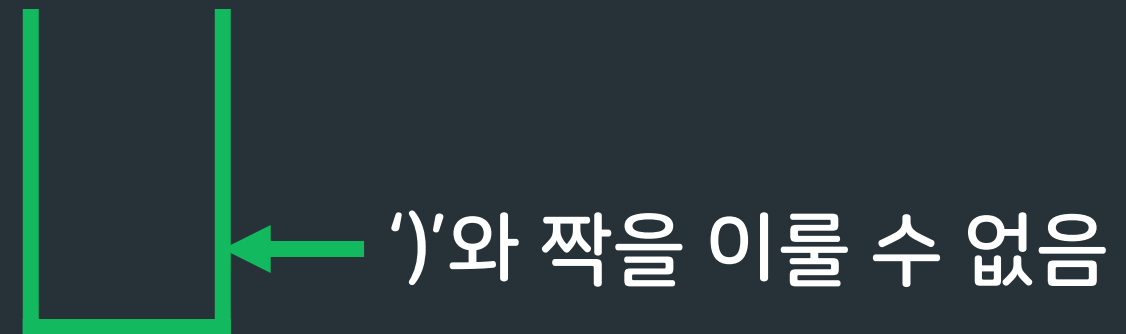


[ first in ] ( first out ).

Empty Stack



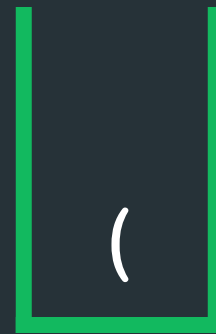
## 괄호의 짝이 맞지 않는 경우 1



A rope may form )( a trail in a maze

Empty Stack

## 괄호의 짝이 맞지 않는 경우 2



Help( I[m being held prisoner in a fortune cookie factory]).

## 괄호의 짝이 맞지 않는 경우 2

[  
(

Help( I[m being held prisoner in a fortune cookie factory]).

## 괄호의 짝이 맞지 않는 경우 2

[  
( ← ')'와 짝을 이룰 수 없음

Help( I[m being held prisoner in a fortune cookie factory]).

# 괄호의 짝이 맞지 않는 경우 3



(

( ) (



## 괄호의 짝이 맞지 않는 경우 3

( ← ')'와 짝

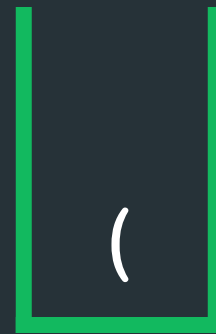
( ) (

# 괄호의 짝이 맞지 않는 경우 3



( ) (

## 괄호의 짝이 맞지 않는 경우 3



`() (`

Not Empty

## /<> 2164번 : 카드2 - Silver 4

### 문제

- N장의 카드에 대해 카드가 한 장 남을 때까지 다음을 반복한다.
  1. 제일 위에 있는 카드를 버린다.
  2. 제일 위에 있는 카드를 제일 아래에 있는 카드 밑으로 옮긴다.

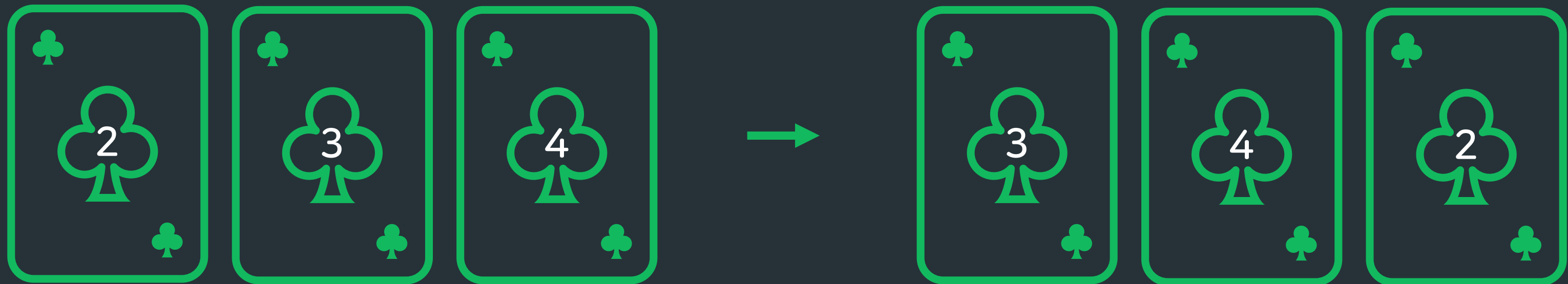
### 제한 사항

- N의 범위는  $1 \leq N \leq 500,000$

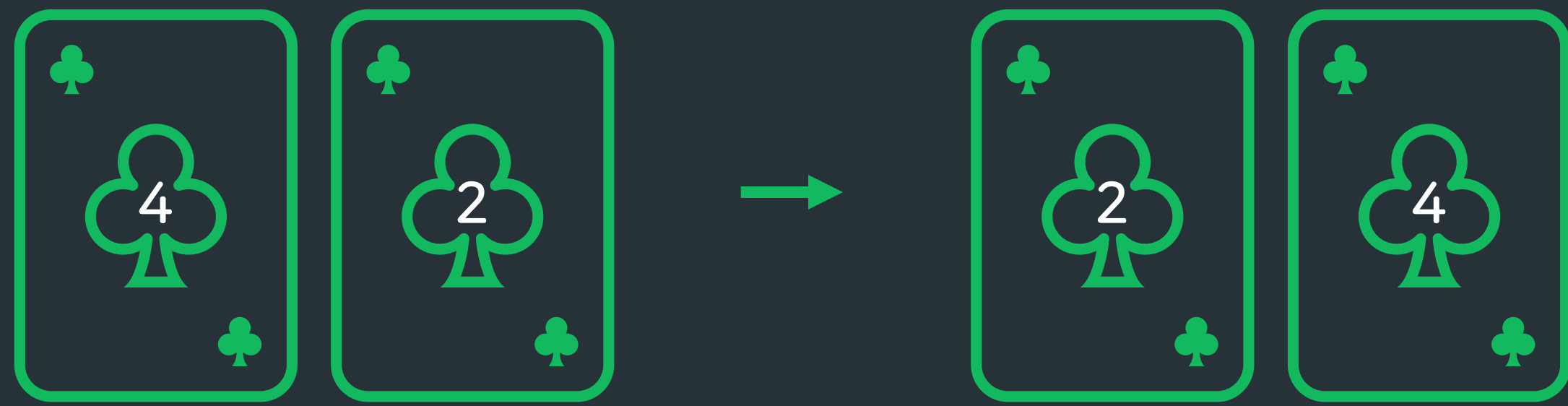
## Hint

1. 카드를 버리는 위치와 추가하는 위치가 다르네요?
2. 맨 앞에 있는 카드를 제거하고 새로운 카드는 맨 뒤에 추가하고...

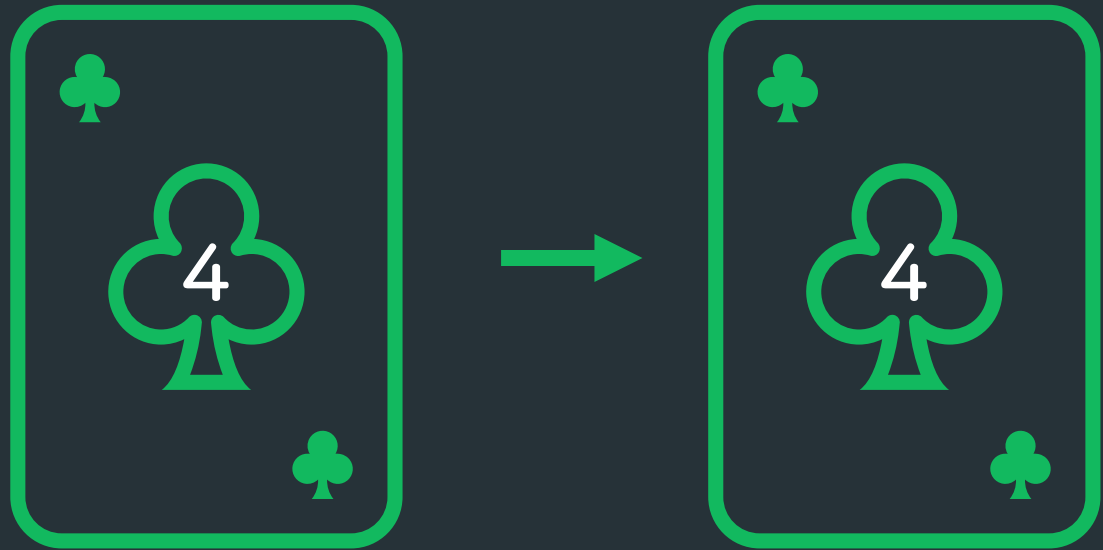
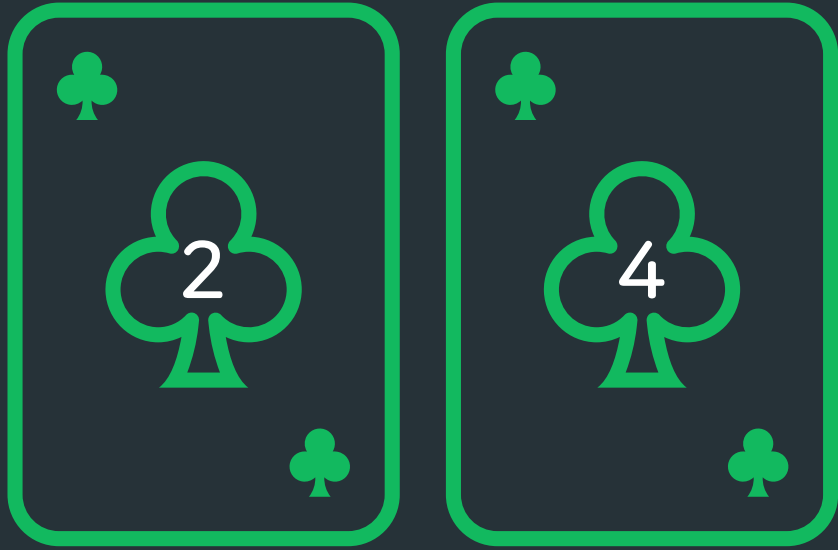
N = 4일 때



N = 4일 때



N = 4일 때





## 정리

- 스택, 큐, 덱 모두 연산에서의 시간 복잡도가  $O(1)$ 인 자료구조
- 효율성을 보는 문제에 사용되는 경우가 많음
- 순회는 벡터보다 불편함
- 무한 루프 (pop을 하지 않음), 런타임 에러 (empty 체크 안하고 조회 or 삭제 시도) 조심!!

## 이것도 알아보세요!

- 재귀 함수로 짠 알고리즘은 스택으로 구현할 수도 있는 경우가 많아요. 예습 할 겸 찾아보세요!

## 3문제 이상 선택

 2019 카카오 개발자 겨울 인턴십 : 크레인 인형뽑기 게임 - Level 1

/<> 1918번 : 후위 표기식 - Gold 3

/<> 1935번 : 후위 표기식2 - Silver 3

/<> 5430번 : AC - Gold 5

/<> 10866번 : 덱 - Silver 4

/<> 20920번 : 영단어 암기는 괴로워 - Silver 3