

스택

큐

조

트리

, 문자열

자료구조 1

최백준 choi@startlink.io

Stack

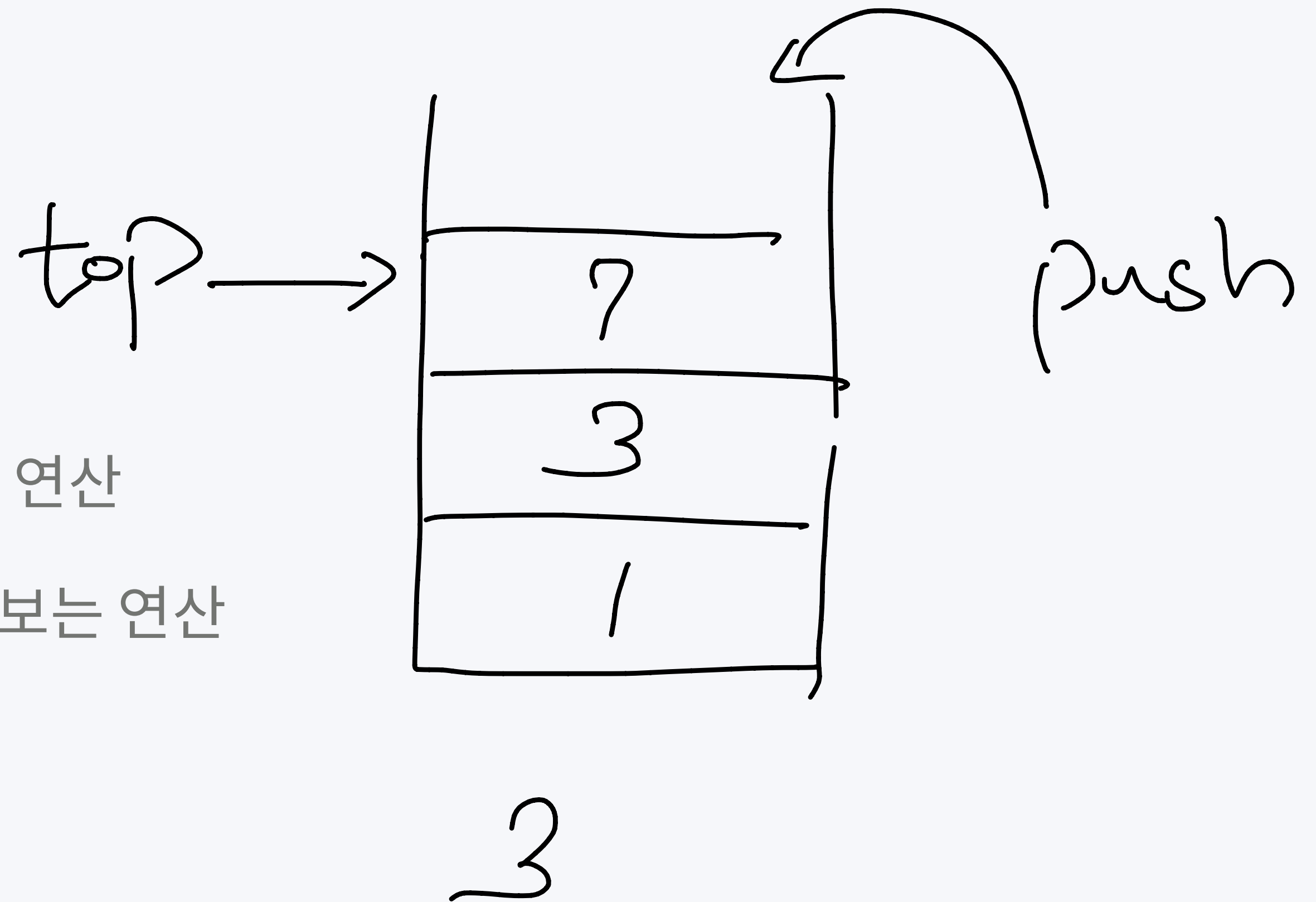
스택

스택

Stack

- 한쪽 끝에서만 자료를 넣고 뺄 수 있는 자료구조
- 마지막으로 넣은 것이 가장 먼저 나오기 때문에 Last In Frist Out (LIFO) 라고도 한다.

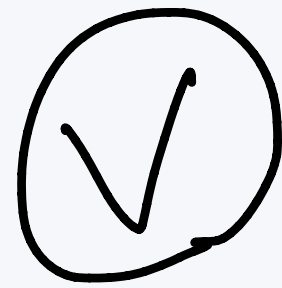
- push: 스택에 자료를 넣는 연산
- pop: 스택에서 자료를 빼는 연산
- top: 스택의 가장 위에 있는 자료를 보는 연산
- empty: 스택이 비어있는지 아닌지를 알아보는 연산
- size: 스택에 저장되어있는 자료의 개수를 알아보는 연산



0

스택

Stack



- 스택은 C++의 경우에는 STL의 stack
- Java의 경우에는 java.util.Stack을 사용하는 것이 좋다.

push

$\text{Stack}[\text{size}] = V;$

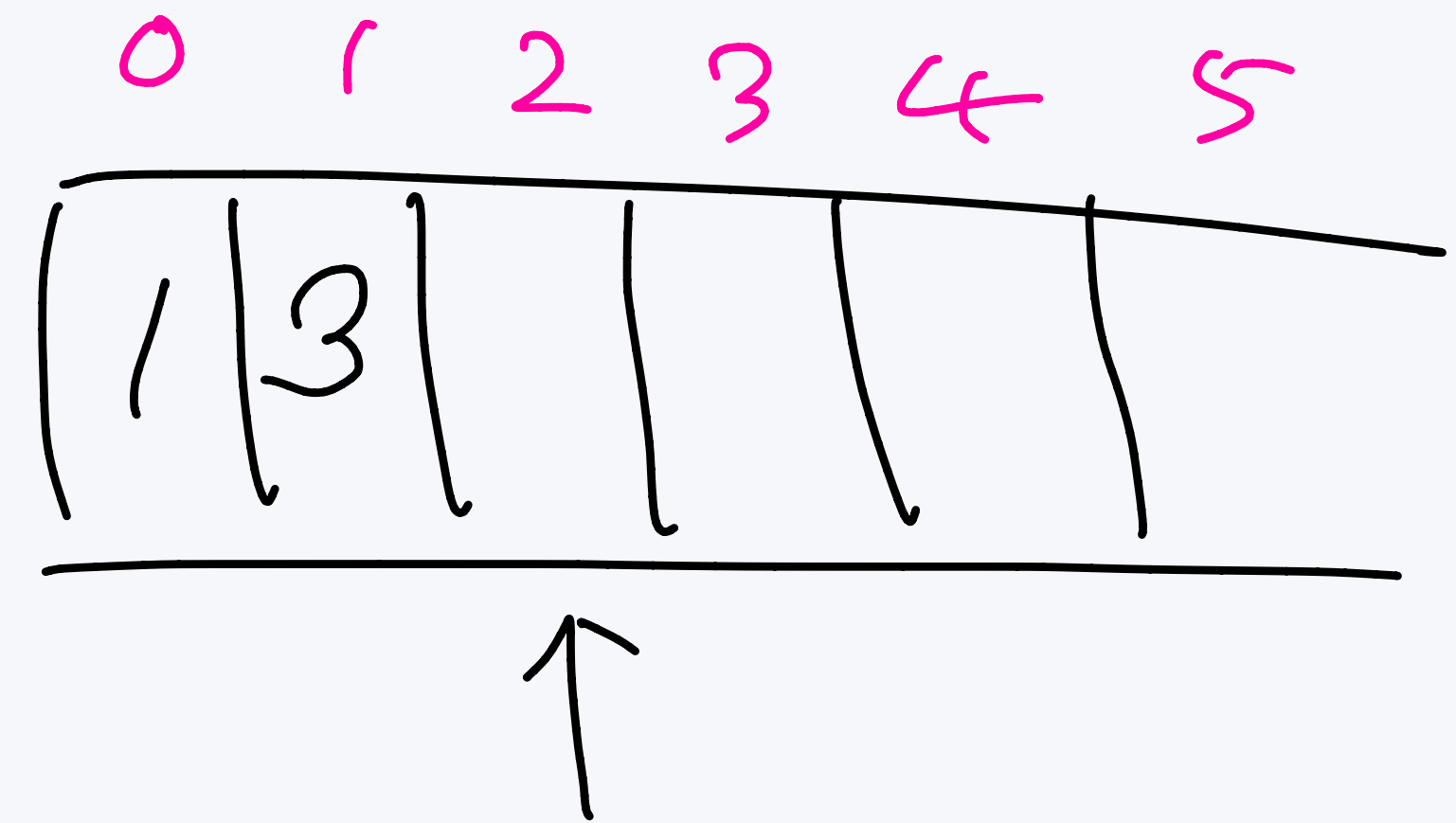
$\text{size} += 1;$

pop :

$\text{Stack}[\text{size} - 1] = 0;$

$\text{size} -= 1;$

Stack:



$\text{size} = 2$

스택

Stack

5

- 스택: (비어있음)
- push 1
- 스택: 1
- push 2
- 스택: 1 2
- top
- 가장 위에 있는 자료는 2
- size
- 스택에 저장되어 있는 자료의 개수는 2

스택

Stack

6

- 스택: 1 2
- empty
- 스택은 비어있지 않음
- pop
- 스택: 1
- pop
- 스택: (비어있음)
- pop
- 비어있기 때문에 연산을 무시함

스택

<https://www.acmicpc.net/problem/10828>

- 스택을 구현하는 문제
- C++ (STL): <https://gist.github.com/Baekjoon/d323faa77a5398acb888>
- C++ (구현): <https://gist.github.com/Baekjoon/436f71851cf017361030>
- Java: <https://gist.github.com/Baekjoon/a71527205b70dc6bd6c5>
- Python: <https://gist.github.com/Baekjoon/f8184d773a8643c3fb47>

괄호

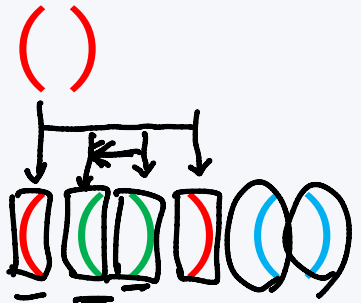


<https://www.acmicpc.net/problem/9012>

- 괄호 문자열이 주어졌을 때, 올바른 괄호 문자열인지 아닌지를 알아보는 문제
- 괄호 문자열: (와)로만 이루어진 문자열
- 올바른 괄호 문자열: 괄호의 쌍이 올바른 문제


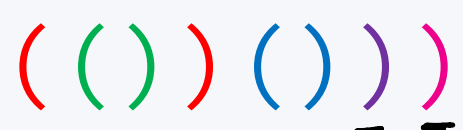

괄호

<https://www.acmicpc.net/problem/9012>

올바른 괄호 문자열의 예시

-  $O(N^2)$
-  ~~단 $O(N)$~~ $O(1)$
-  여 $O(1)$

올바른 괄호 문자열이 아닌 예

-  $O(N)$ $O(1)$
-  $O(N)$ $O(1)$
-  $O(N)$ $O(1)$

다른 괄호

① 왼쪽

② 아직 짝이 맞지 않음

③ 가장 오른쪽에 있는 괄호

Stack, ✓

top

괄호

<https://www.acmicpc.net/problem/9012>

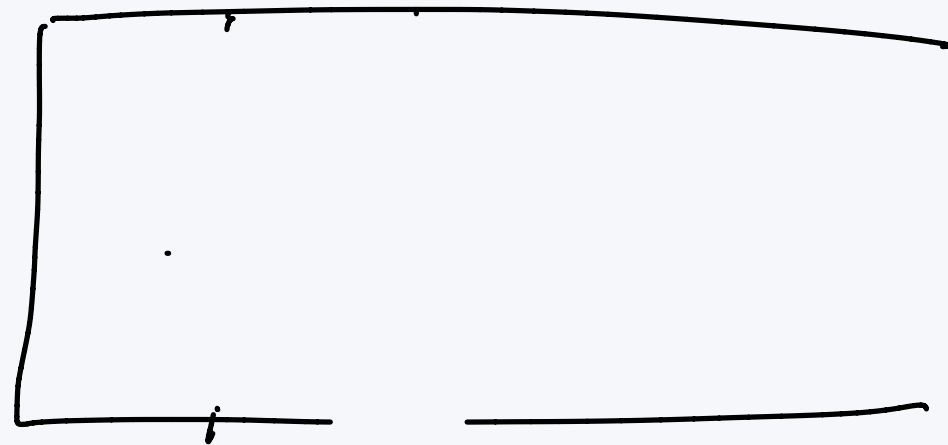
- 스택을 사용해서 올바른 괄호 문자열인지 아닌지를 알 수 있다.
- (가 나오면 스택에 (를 넣고
-)가 나오면 스택에서 하나를 빼서 (인지 확인한다.
- 또는 하나를 뺄 수 있는지를 확인한다.

괄호

<https://www.acmicpc.net/problem/9012>

- $((()()))$ → 여는 괄호가 없다.

• 스택:



) U-9L2C1

스택이 비어있을 때

False

괄호

<https://www.acmicpc.net/problem/9012>

- (()) ()
- 스택: (
- (()) ()
- 스택: ((
- (()) ()
- 스택: (
- (()) ()
- 스택:

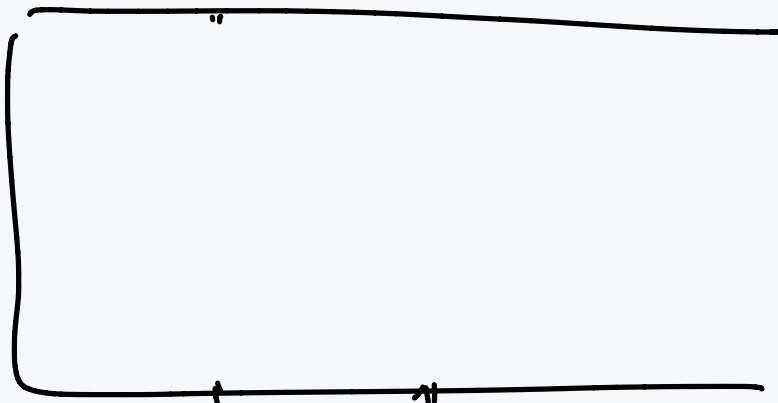
괄호

<https://www.acmicpc.net/problem/9012>

- (()) ()
- 스택: (
- (()) ()
- 스택:
- (()) ()
- 스택: ???
- 스택이 비어있는데)가 나타났으므로 올바른 괄호 문자열이 아니다

괄호

<https://www.acmicpc.net/problem/9012>

- ((()((()))) 
- 스택:
- ((()((())))
- 스택: (
- ((()((())))
- 스택: ((
- ((()((())))
- 스택: (

올바른 괄호 문자열

괄호

<https://www.acmicpc.net/problem/9012>

- (() (()))
- 스택: ((
- (() ()) ((()))
- 스택: (
- (() ()) ((()))
- 스택:
- (() ()) ((()))
- 스택: (

괄호

<https://www.acmicpc.net/problem/9012>

- (() ()) ((()))
- 스택: ((
- (() ()) ((()))
- 스택: (((
- (() ()) ((()))
- 스택: ((
- (() ()) ((()))
- 스택: (

괄호

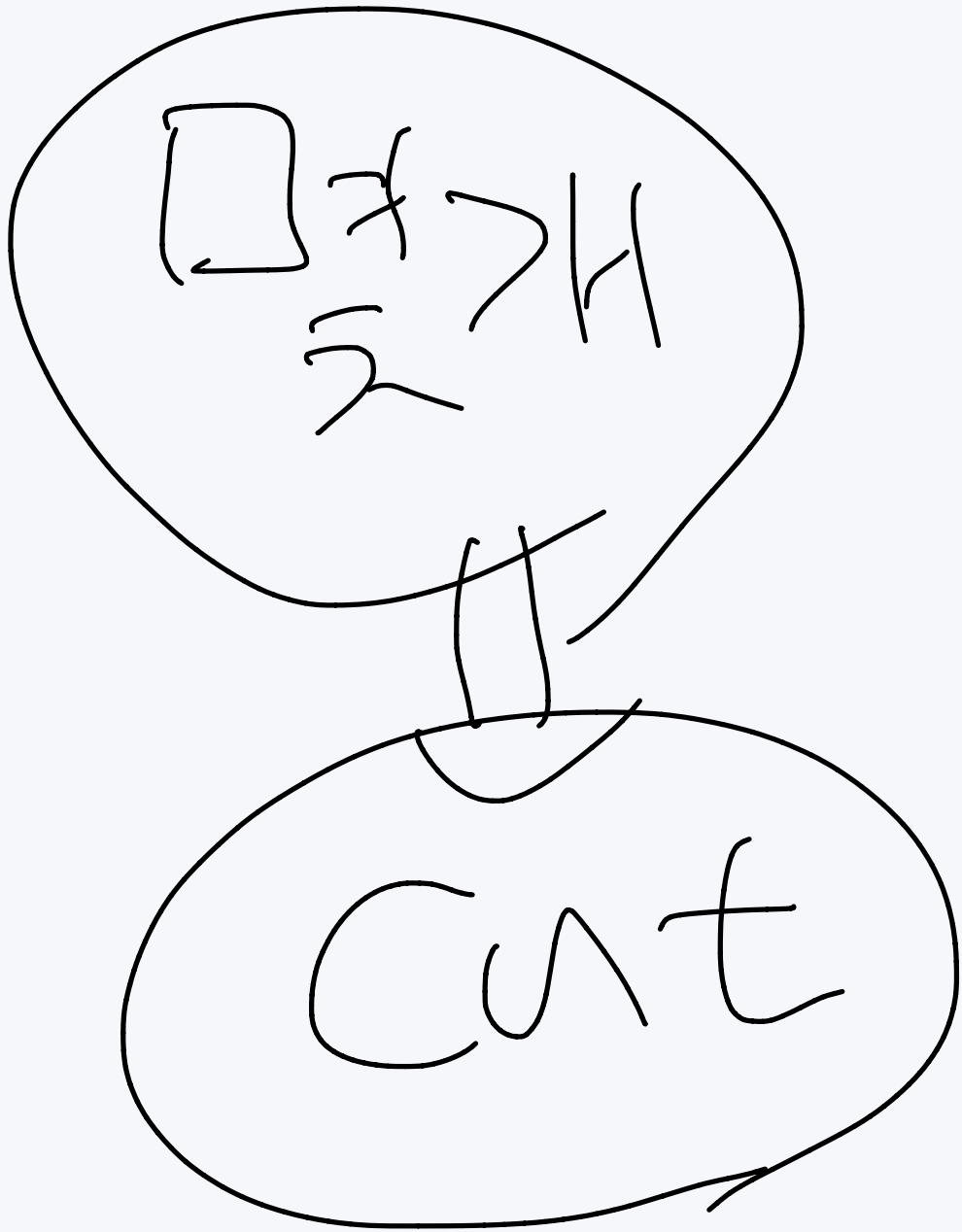
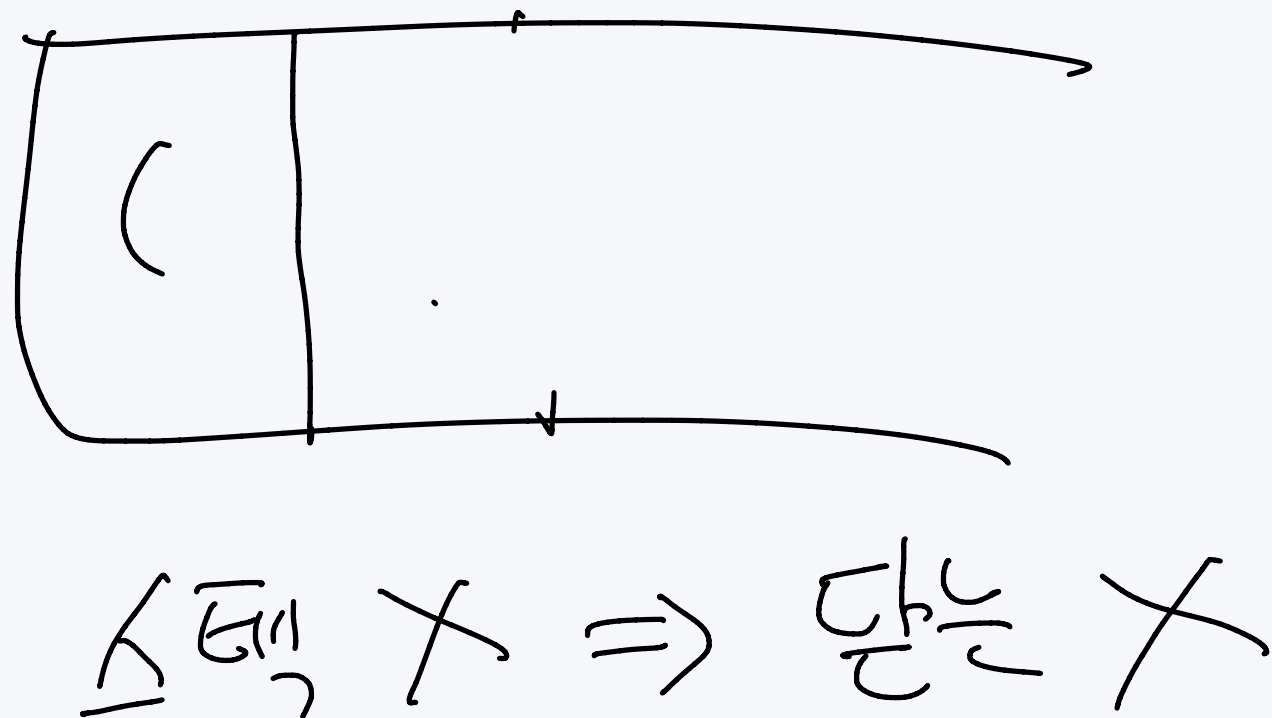
<https://www.acmicpc.net/problem/9012>

- `((()())((())))`
- 스택:
- 모든 과정이 끝났고, 스택이 비어있다!
- 올바른 괄호 문자열

괄호

<https://www.acmicpc.net/problem/9012>

- ((0)
- 스택:
- ((
- 스택: (
- ((
- 스택: ((
- ((
- 스택: (



• 끝났는데, 스택이 비어있지 않기 때문에 올바른 괄호 문자열이 아니다

괄호

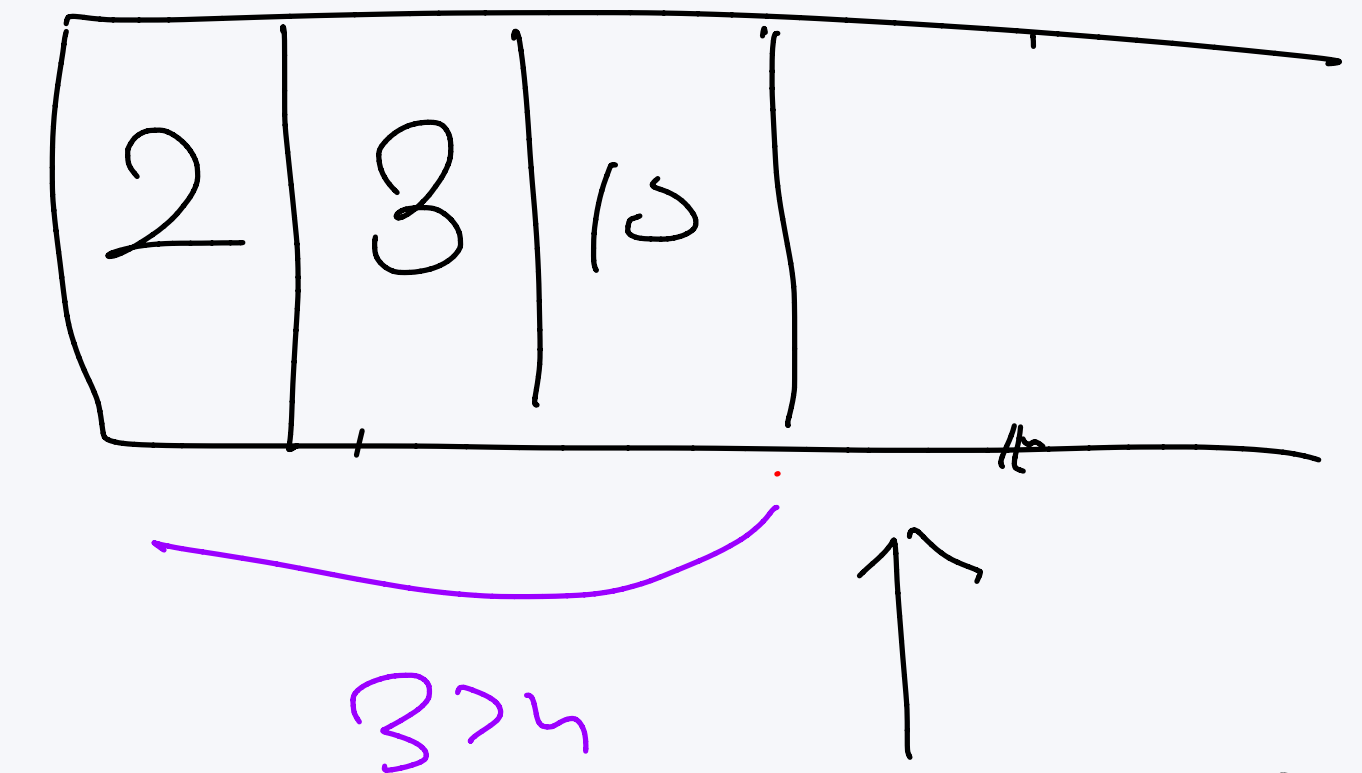
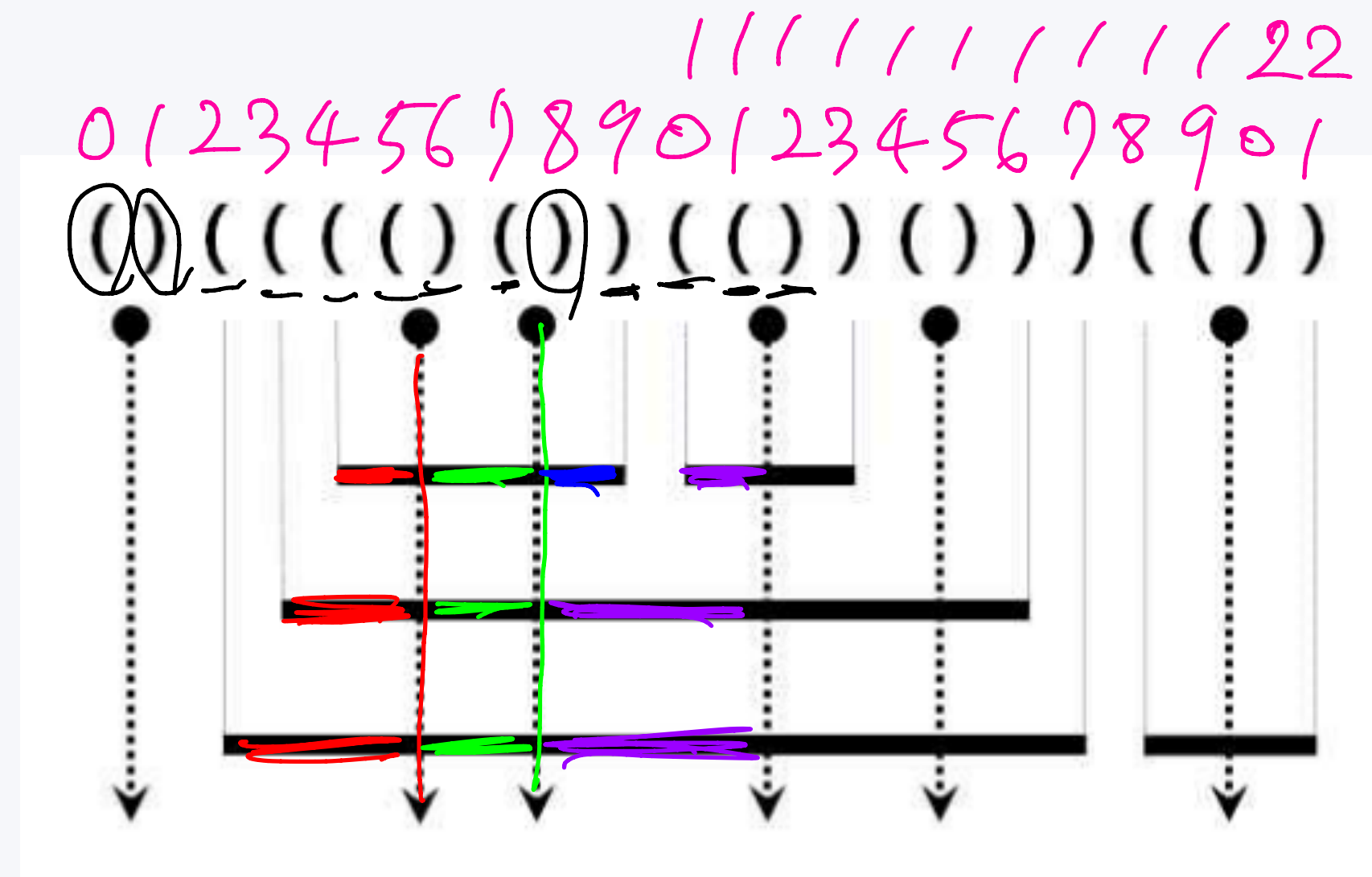
<https://www.acmicpc.net/problem/9012>

- C++: <https://gist.github.com/Baekjoon/d2c9ecf50f11197f947e>
- Java: <https://gist.github.com/Baekjoon/c5052cc44c9ccc15733b>
- Python: <https://gist.github.com/Baekjoon/834101d1ca6ab4be7d4b>

- 레이저는 여는 괄호와 닫는 괄호의 인접한 쌍 $()$ 으로 표현된다. 또한, 모든 $()$ 는 반드시 레이저를 표현한다.

$$3 + 3 + 1 + 3$$

- 쇠막대기의 왼쪽 끝은 여는 괄호 ‘ (’ 로, 오른쪽 끝은 닫힌 괄호 ‘) ’ 로 표현된다



쇠막대기

<https://www.acmicpc.net/problem/10799>

- 올바른 괄호 문자열과 비슷하게 풀 수 있다
- ()가 나올 때 마다 스택에 들어있는 (의 개수를 세어준다
- 그런데,)가 나왔을 때, 이것이 레이저인지 쇠막대기인지 구분을 해줘야 한다.
- 레이저는 항상 ()와 같이 붙어진 상태로 나온다
- 스택에 (의 인덱스를 넣어서 인덱스가 1차이나는지 확인해야 한다

최막대기

<https://www.acmicpc.net/problem/10799>

- C/C++: <https://gist.github.com/Baekjoon/cc379d17334b3896cfd3>
- Java: <https://gist.github.com/Baekjoon/52814ae195373c8384b9>
- Python: <https://gist.github.com/Baekjoon/ab86743a1e2ed62352f1>

에디터

(N)

N^2

36003

23

<https://www.acmicpc.net/problem/1406>

abc | xyz

• 커서: 문장의 맨 앞, 맨 뒤, 문장 중간 임의의 곳에 위치할 수 있다.

• L: 커서를 왼쪽으로 한 칸 옮김 $O(1)$ $O(N)$

• D: 커서를 오른쪽으로 한 칸 옮김 $O(1)$

• B: 커서 왼쪽에 있는 문자를 삭제함 $O(N)$ $O(1)$

• P \$: \$라는 문자를 커서 오른쪽에 추가함. 커서는 \$에 위치함 $O(N)$

↙ L

ab | cxyz

↓ B

acd | xyz

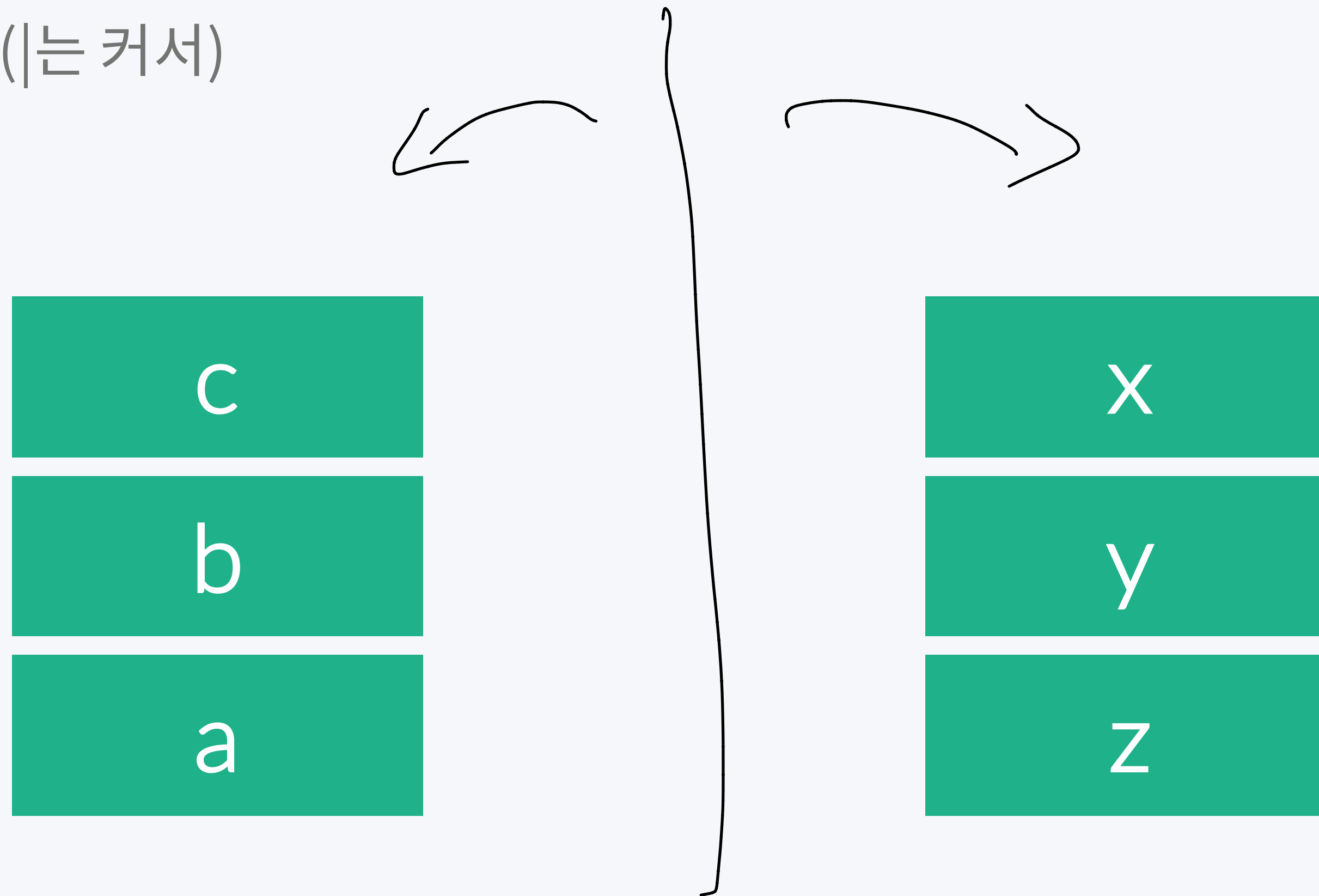
acd | xyz
↙ P
↓

$O(1)$
← D
acd | xyz

에디터

<https://www.acmicpc.net/problem/1406>

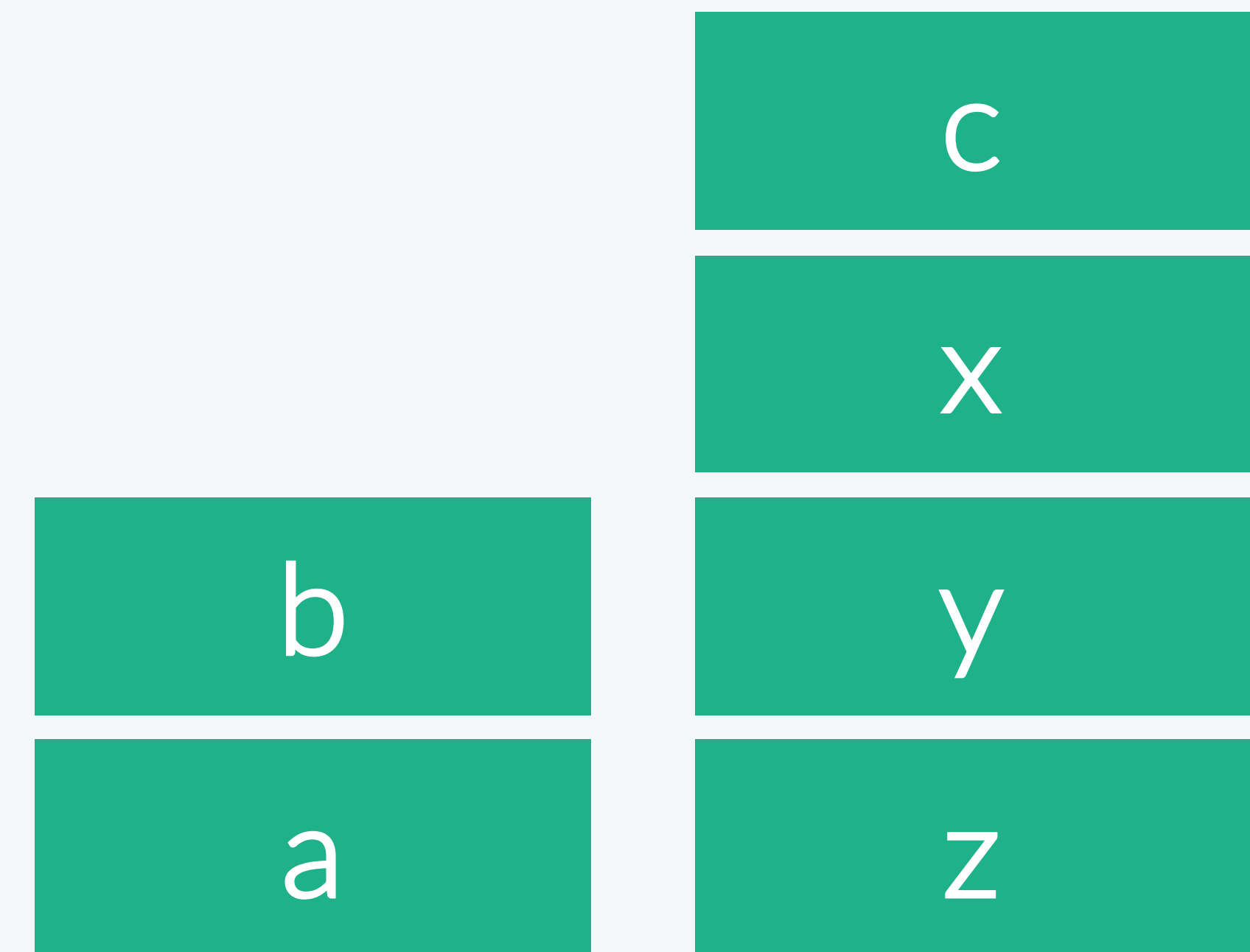
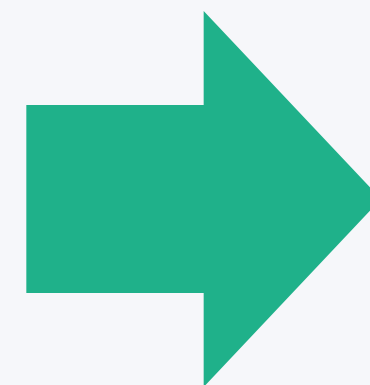
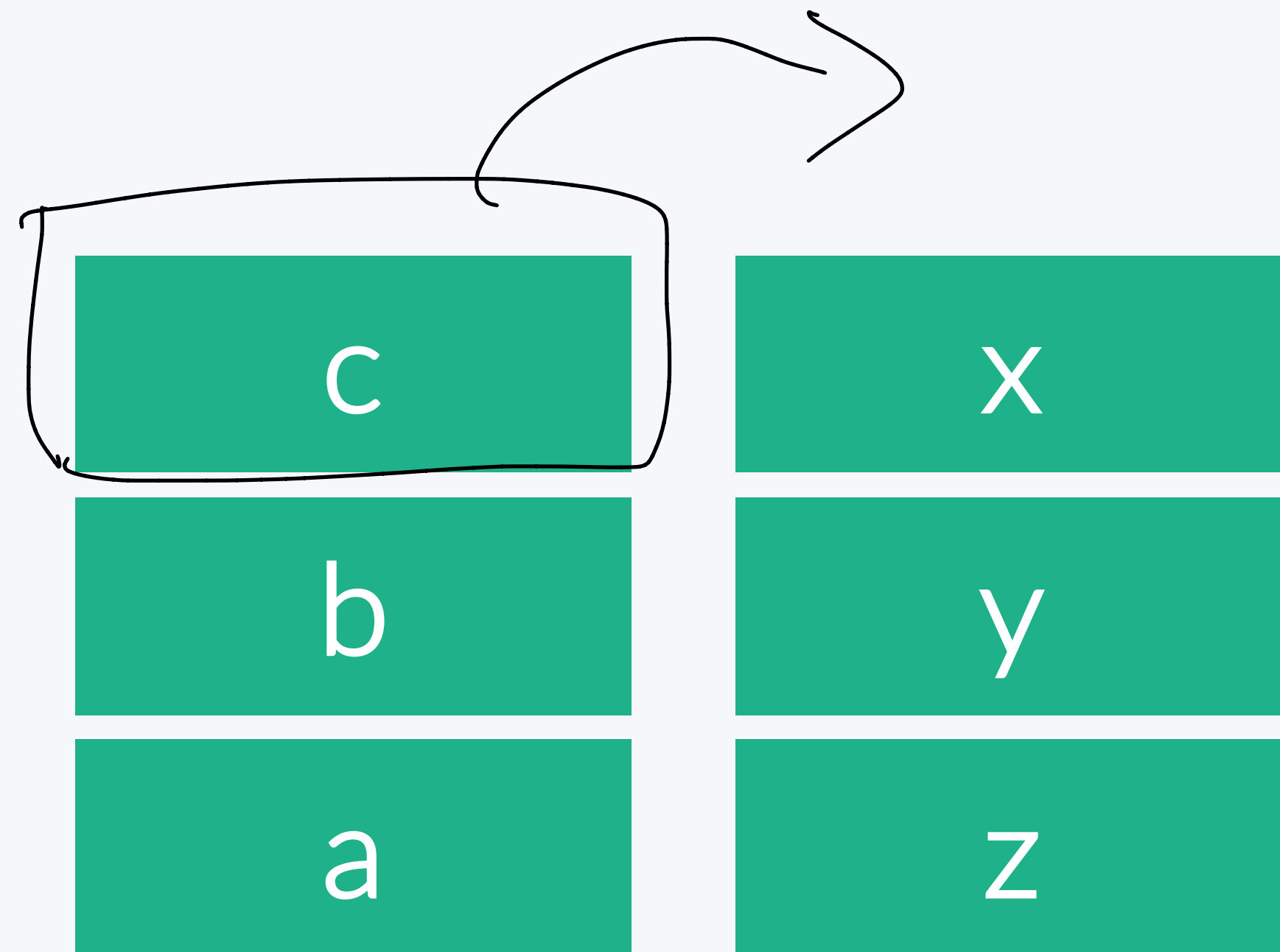
- 커서를 기준으로 커서의 왼쪽 스택(left)와 오른쪽 스택(right)로 나눠서 문제를 풀 수 있다.
- 예: abc | xyz (|는 커서)



에디터

<https://www.acmicpc.net/problem/1406>

- L: 커서를 왼쪽으로 한 칸 옮김
- `abc | xyz -> ab | cxyz`



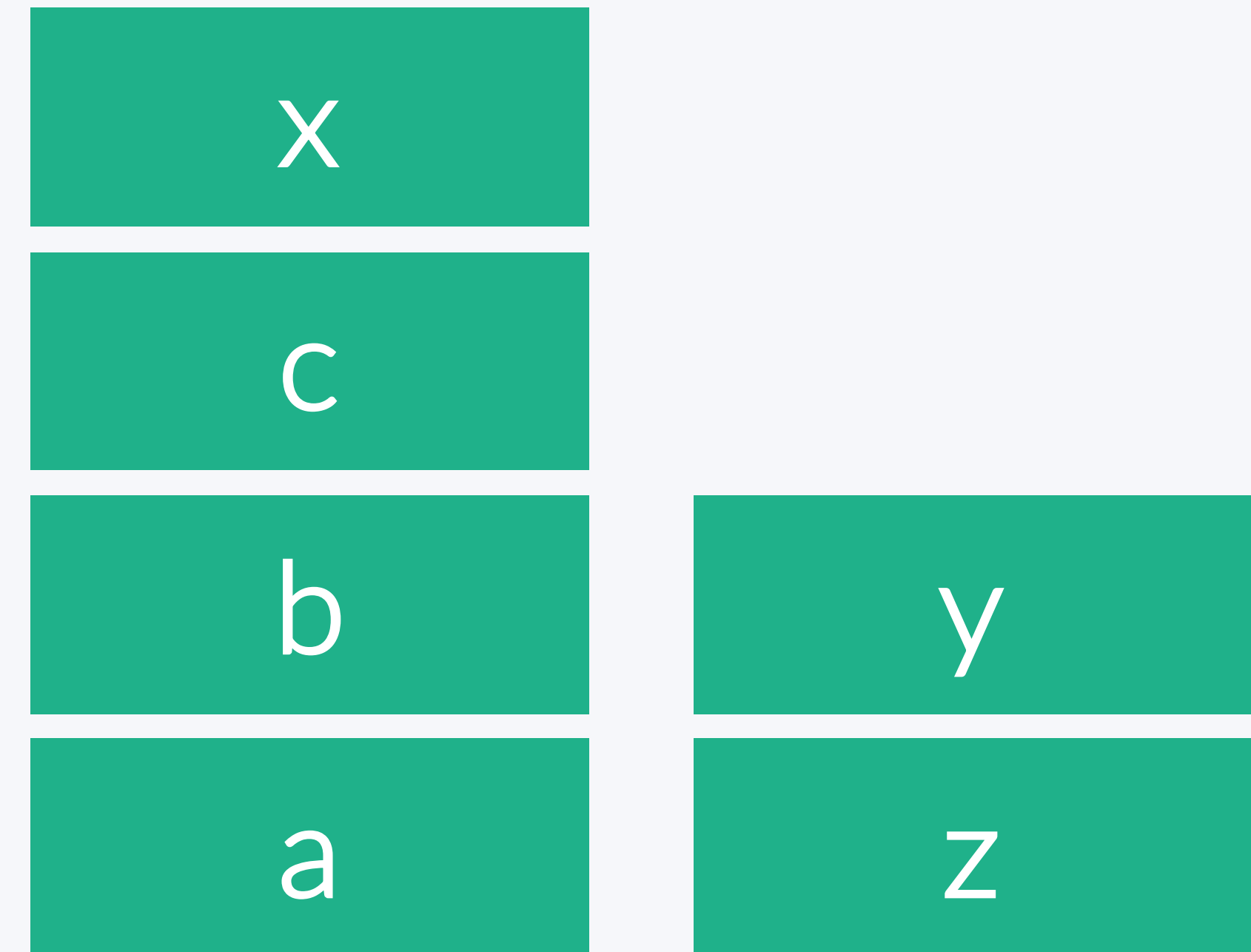
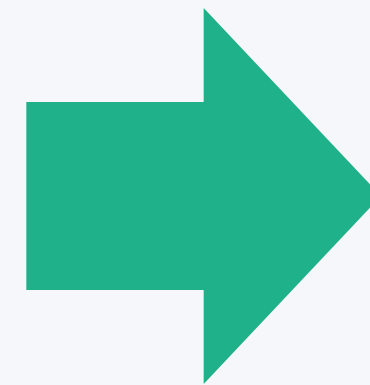
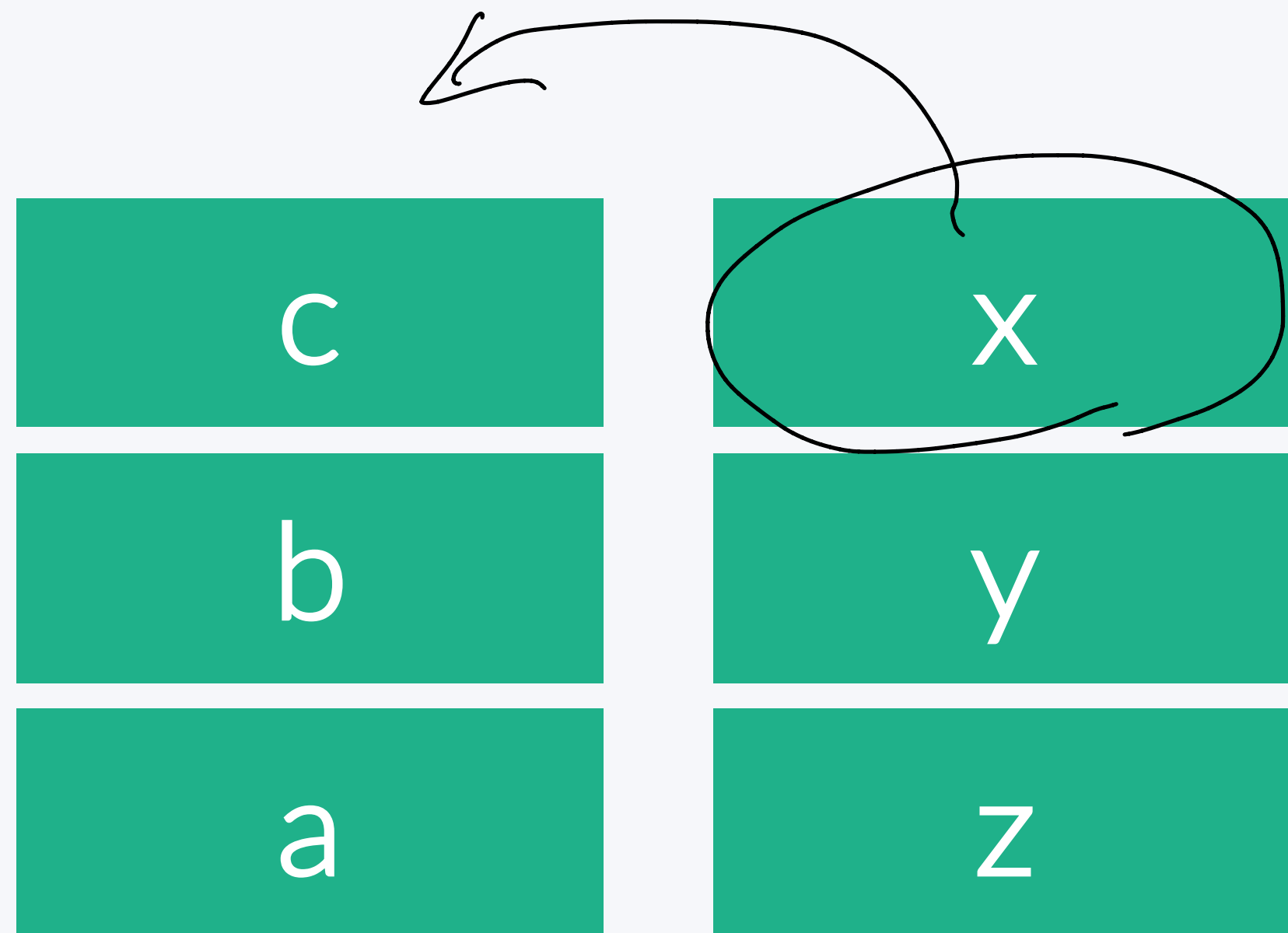
pop
push

O ()

에디터

<https://www.acmicpc.net/problem/1406>

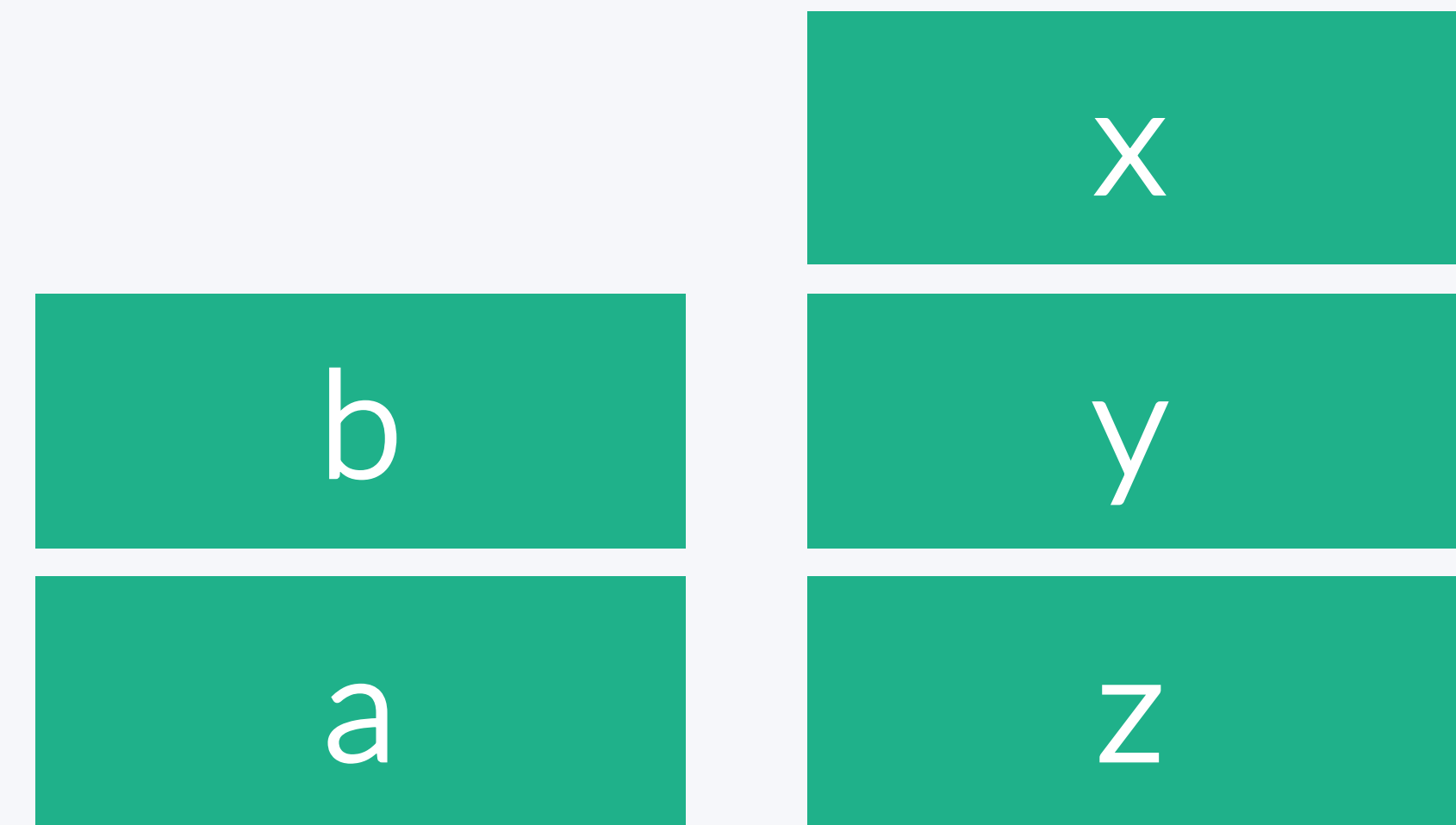
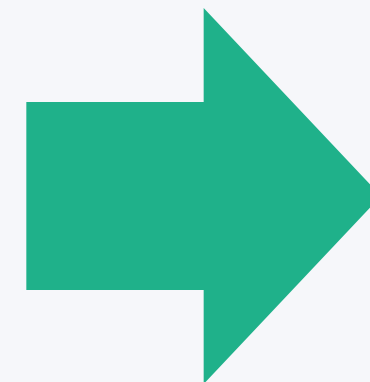
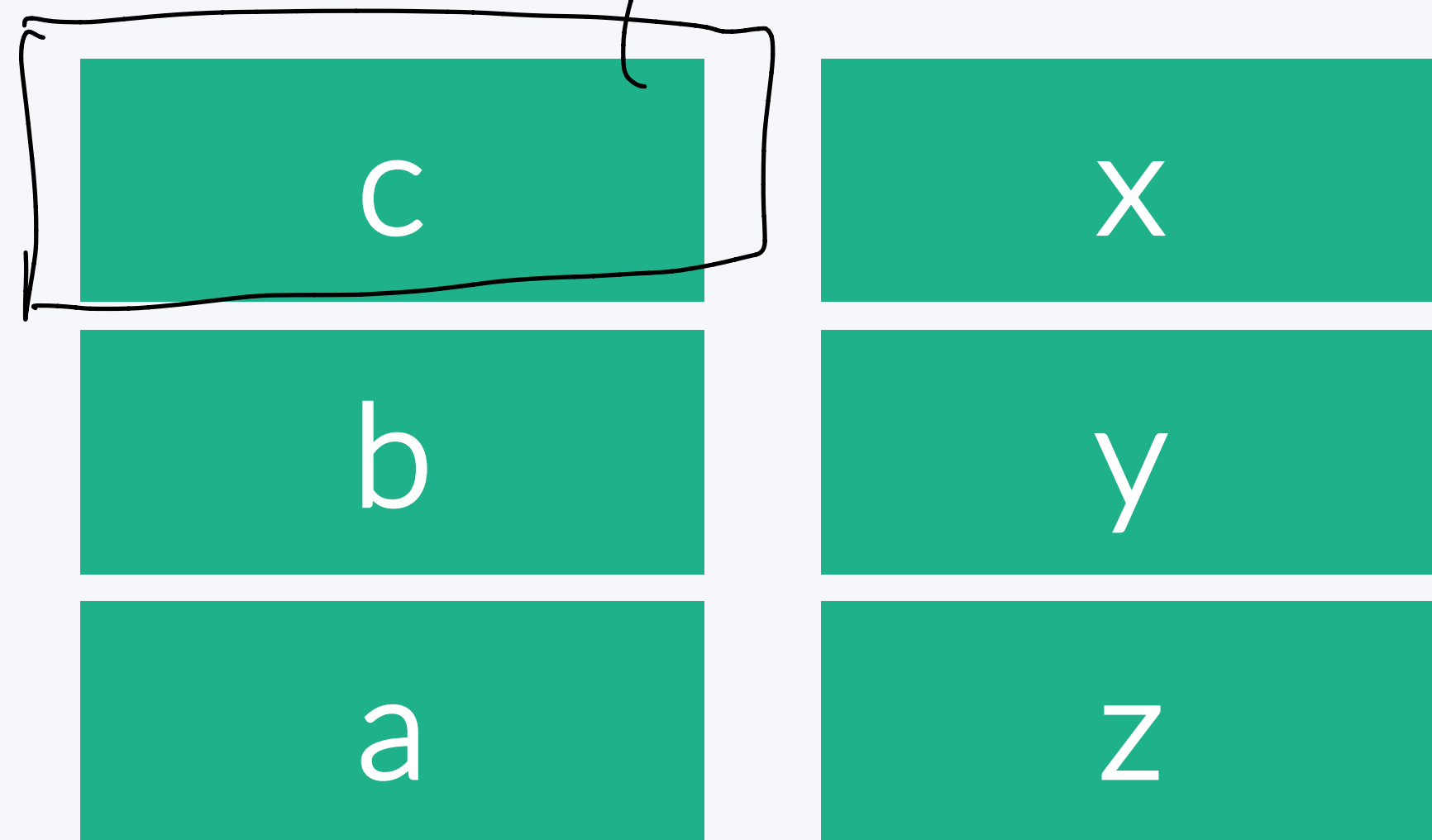
- D: 커서를 오른쪽으로 한 칸 옮김
- $abc \mid xyz \rightarrow abcx \mid yz$



에디터

<https://www.acmicpc.net/problem/1406>

- B: 커서를 왼쪽에 있는 문자를 삭제함
- `abc | xyz -> ab | xyz`



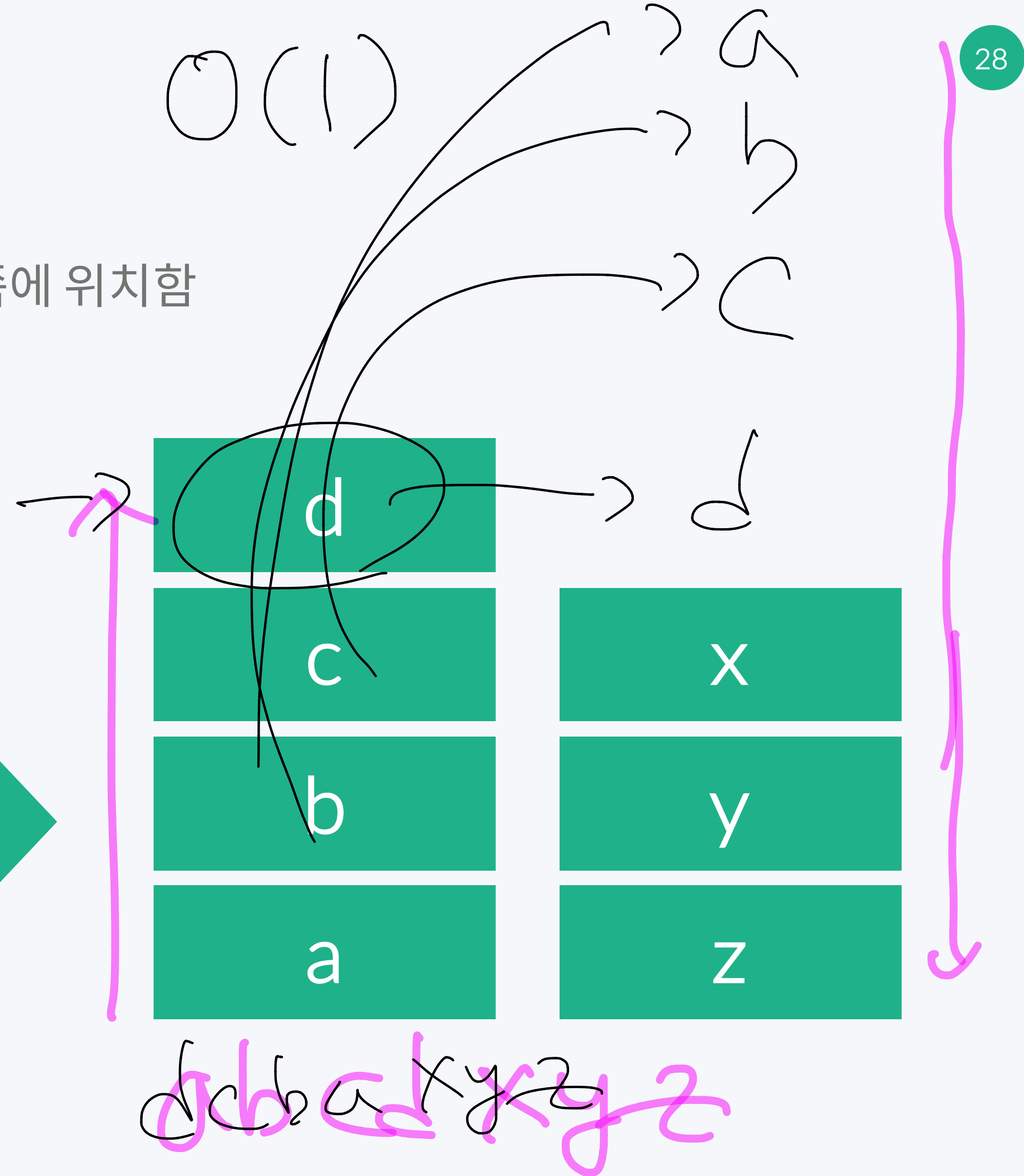
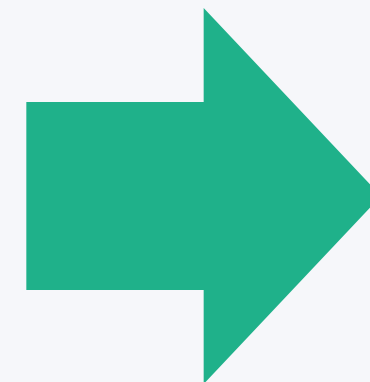
`O(1)`

에디터

<https://www.acmicpc.net/problem/1406>

- P \$: \$를 커서 오른쪽에 추가하고 커서는 \$의 오른쪽에 위치함
- abc | xyz -> abcd | xyz

c	x
b	y
a	z



에디터

<https://www.acmicpc.net/problem/1406>

- C/C++: <https://gist.github.com/Baekjoon/c52dd9a4987ea1a009fc>
- Java: <https://gist.github.com/Baekjoon/3f90237fcd175f9b1aea>

큐

큐

Queue

31

- 한쪽 끝에서만 자료를 넣고 다른 한쪽 끝에서만 뺄 수 있는 자료구조
- 먼저 넣은 것이 가장 먼저 나오기 때문에 First In First Out (FIFO) 라고도 한다.
- push: 큐에 자료를 넣는 연산
- pop: 큐에서 자료를 빼는 연산
- front: 큐의 가장 앞에 있는 자료를 보는 연산
- back: 큐의 가장 뒤에 있는 자료를 보는 연산
- empty: 큐가 비어있는지 아닌지를 알아보는 연산
- size: 큐에 저장되어있는 자료의 개수를 알아보는 연산



큐

Queue

줄
대기

- ~~스택~~은 C++의 경우에는 STL의 queue
- Java의 경우에는 java.util.Queue을 사용하는 것이 좋다.

push

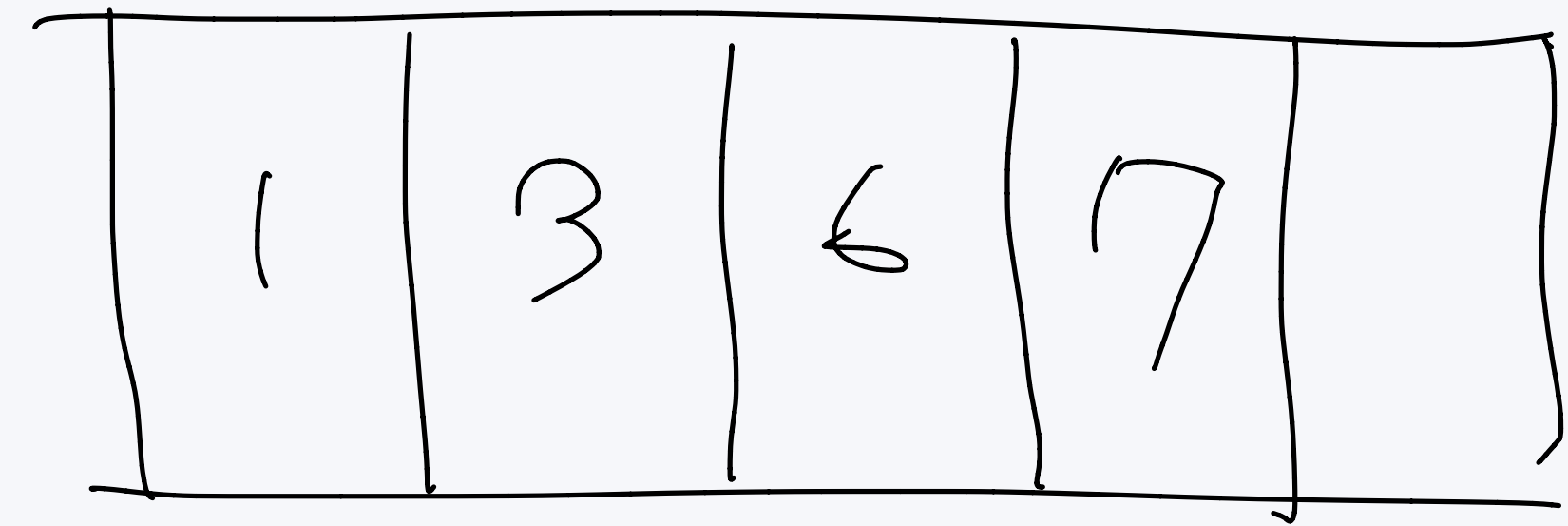
queue[end] = val

end += 1

pop

queue[begin] = 0

begin += 1



begin

end

Size: end - begin

Empty begin = end



Queue

- 큐: (비어있음)
- push 1
- 큐: 1
- push 2
- 큐: 1 2
- front
- 1
- back
- 2



Queue

- 큐: 1 2
- size
- 크기: 2
- pop
- 큐: 2
- pop
- 큐: (비어있음)

큐

<https://www.acmicpc.net/problem/10845>

- 큐를 구현하는 문제

큐

<https://www.acmicpc.net/problem/10845>

- 큐를 구현하는 문제
- C++ (STL): <https://gist.github.com/Baekjoon/b4c43e344304cc81f164>
- C++ (구현): <https://gist.github.com/Baekjoon/ad3f65c8bf538e228a0>
- Java: <https://gist.github.com/Baekjoon/35dbd617141f6c57cf57>

조세퍼스 문제

$$N=7, M=3$$

37

<https://www.acmicpc.net/problem/1158>

- 1번부터 N번까지 N명의 사람이 원을 이루면서 앉아있고, 양의 정수 $M(\leq N)$ 이 주어진다
- 이제 순서대로 M번째 사람을 제거한다
- 한 사람이 제거되면 남은 사람들로 이루어진 원을 따라 이 과정을 계속해 나간다
- 이 과정은 N명의 사람이 모두 제거될 때까지 계속된다
- 원에서 사람들이 제거되는 순서를 (N, M)-조세퍼스 순열이라고 한다

$$O(NM)$$

3, 6, 2, 7, 5, 1, 4

(7, 3)

~~1~~ ~~2~~ ~~3~~ ~~4~~ ~~5~~ ~~6~~ ~~7~~

조세퍼스 문제

<https://www.acmicpc.net/problem/1158>

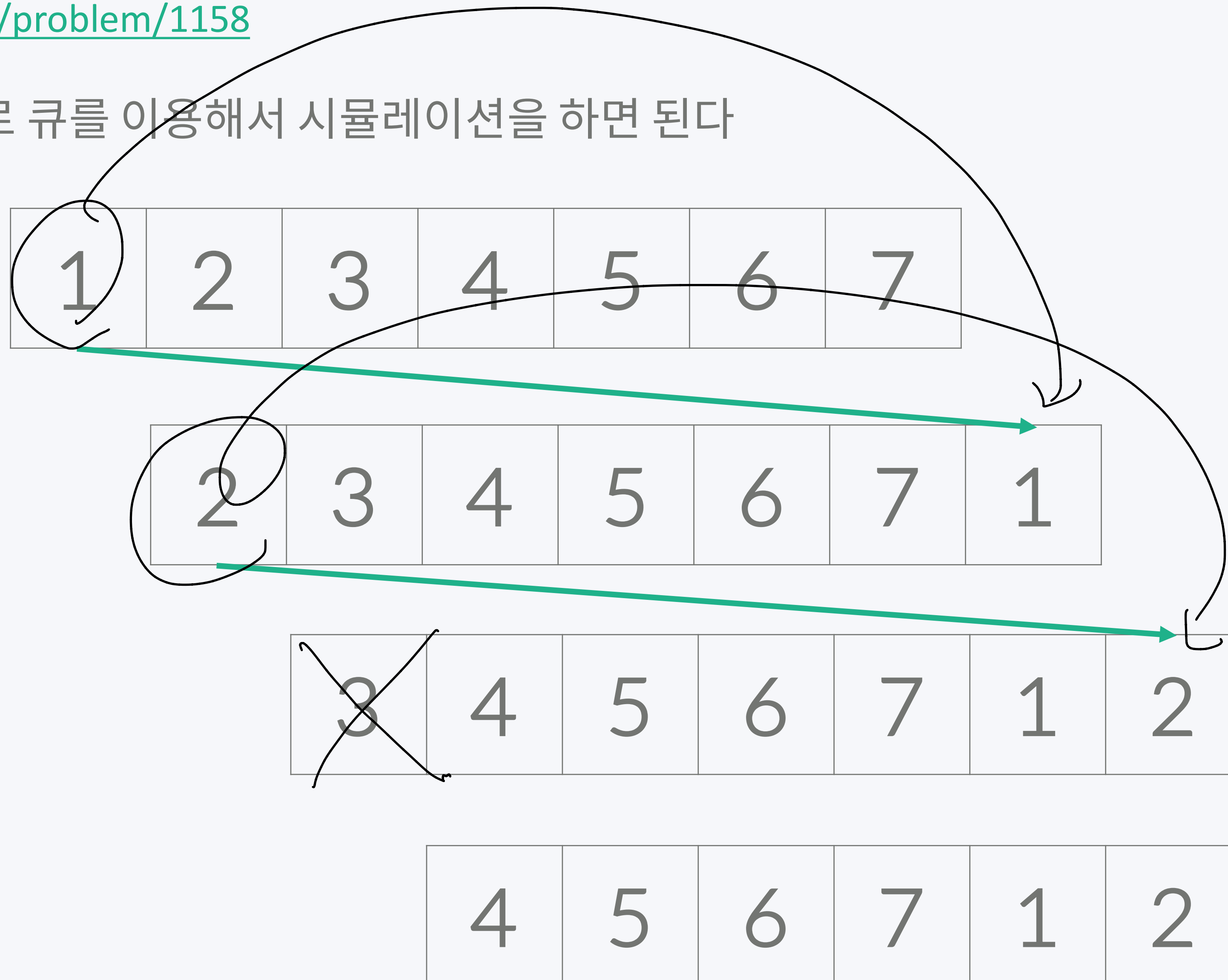
- 문제에 나와있는 대로 큐를 이용해서 시뮬레이션을 하면 된다
- $N = 7$
- $M = 3$

1	2	3	4	5	6	7
---	---	---	---	---	---	---

조세퍼스 문제

<https://www.acmicpc.net/problem/1158>

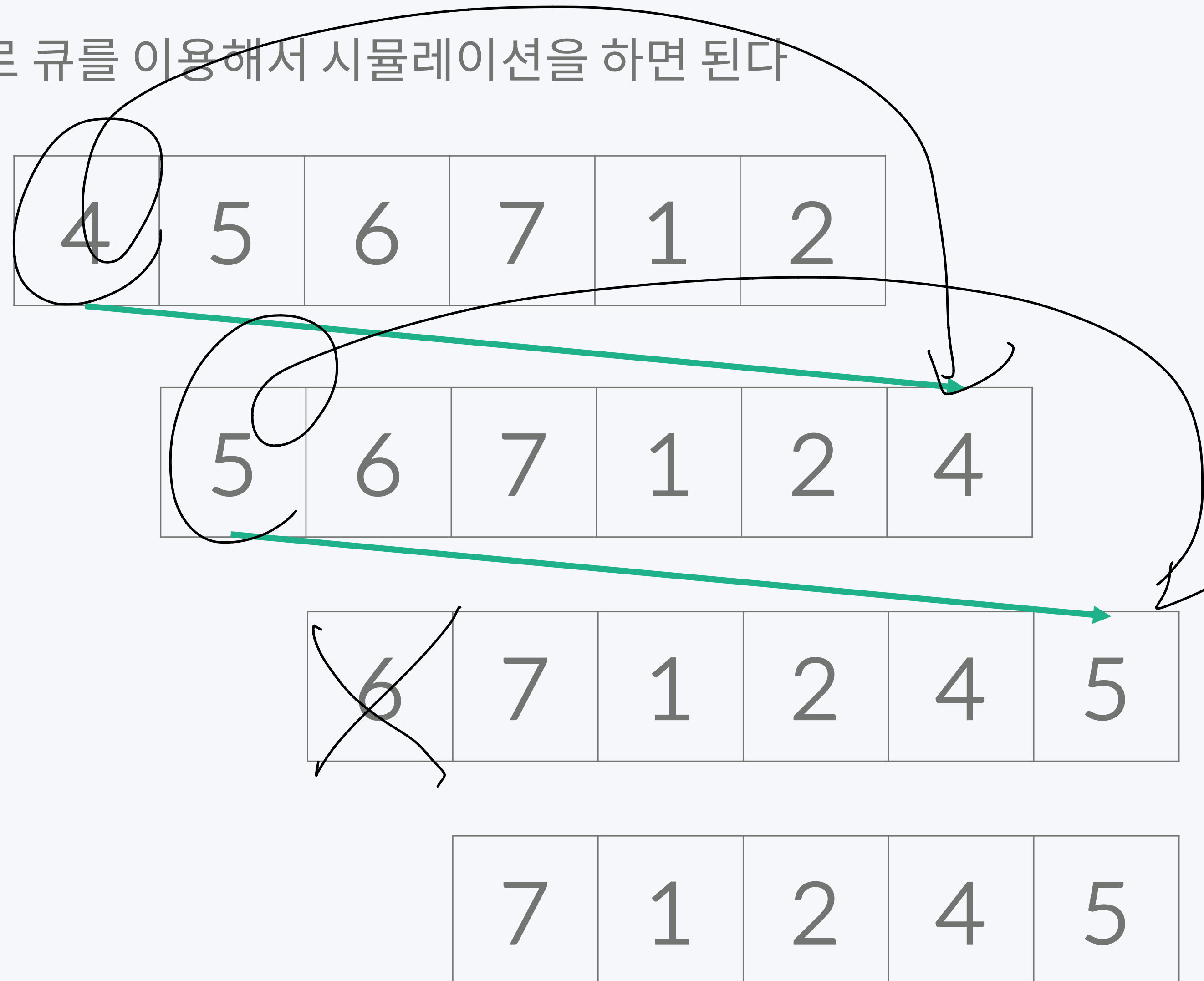
- 문제에 나와있는 대로 큐를 이용해서 시뮬레이션을 하면 된다
- $N = 7$
- $M = 3$



조세퍼스 문제

<https://www.acmicpc.net/problem/1158>

- 문제에 나와있는 대로 큐를 이용해서 시뮬레이션을 하면 된다
- $N = 7$
- $M = 3$



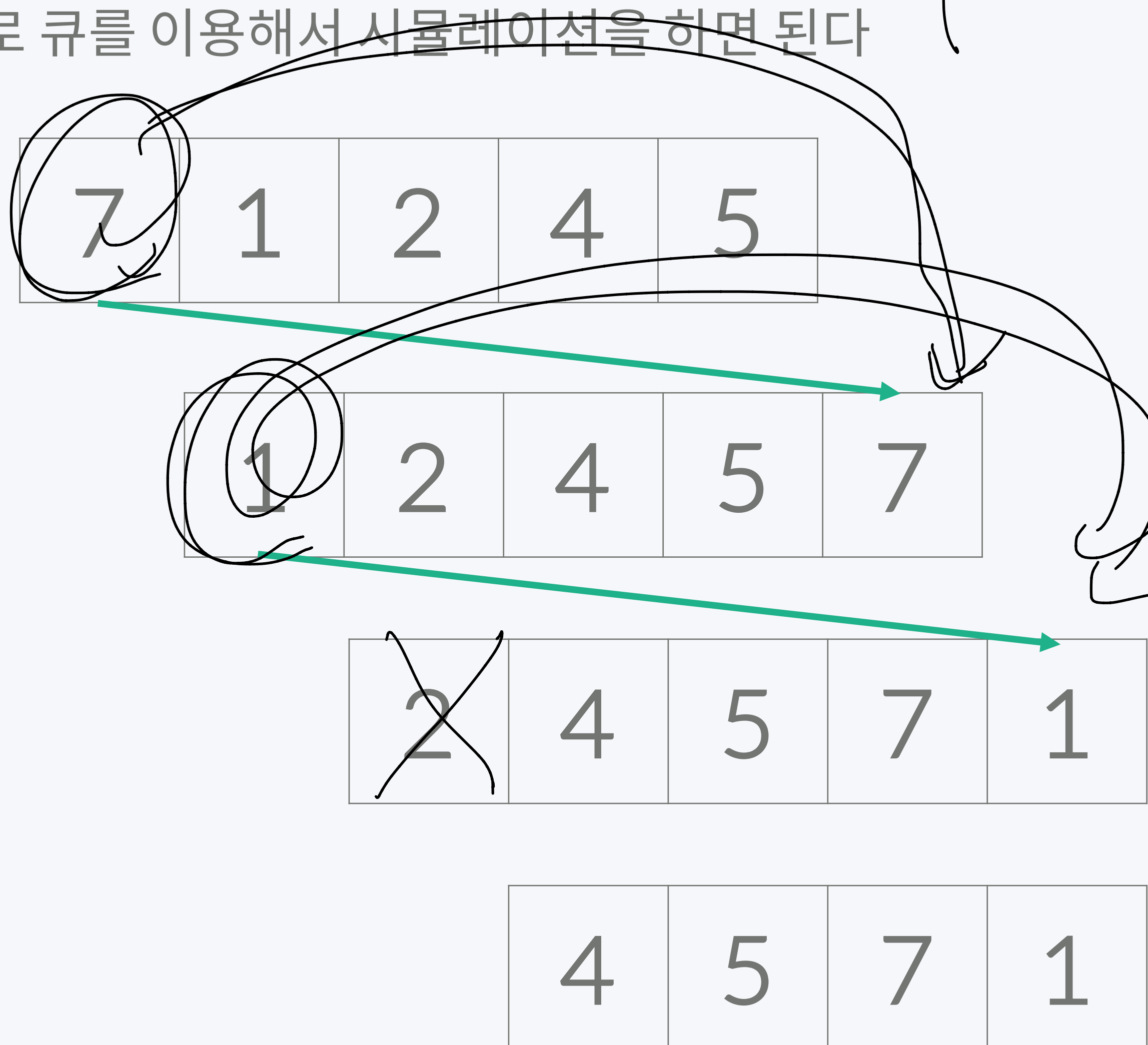
조세퍼스 문제

<https://www.acmicpc.net/problem/1158>

- 문제에 나와있는 대로 큐를 이용해서 시뮬레이션을 하면 된다

- $N = 7$

- $M = 3$



Pop $M-1 + 1$

Push $M-1$

(N, M)

조세퍼스 문제

<https://www.acmicpc.net/problem/1158>

- C/C++: <https://gist.github.com/Baekjoon/bc266ec5479e06de1150>
- Java: <https://gist.github.com/Baekjoon/da7ca551223efc6e21ab>

데

덱

Deque

- 양 끝에서만 자료를 넣고 양 끝에서 뺄 수 있는 자료구조
- Double-ended queue의 약자이다.
- push_front: 큐에 자료를 넣는 연산
- pop: 큐에서 자료를 빼는 연산
- front: 큐의 가장 앞에 있는 자료를 보는 연산
- back: 큐의 가장 뒤에 있는 자료를 보는 연산
- empty: 큐가 비어있는지 아닌지를 알아보는 연산
- size: 큐에 저장되어있는 자료의 개수를 알아보는 연산

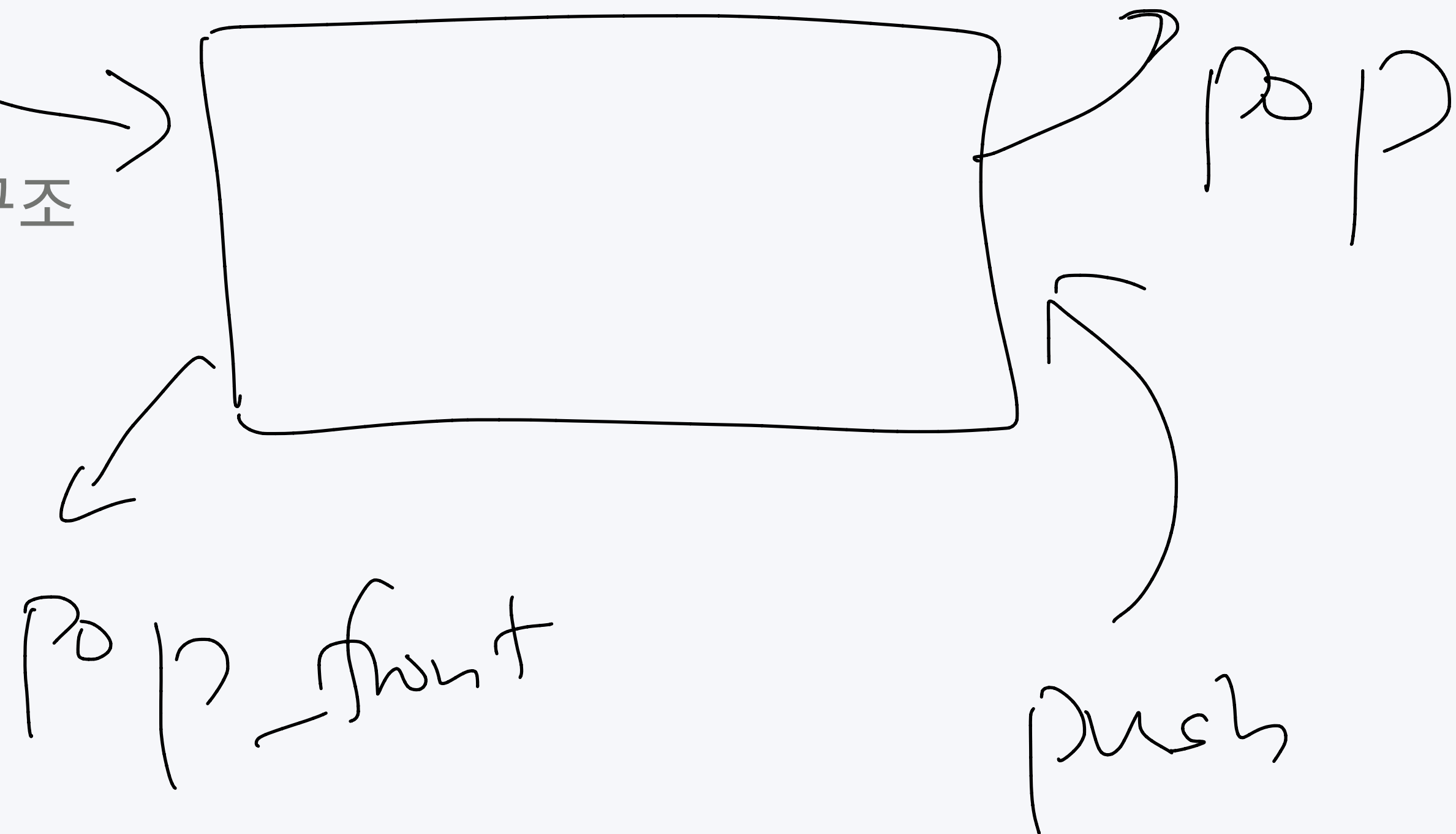
덱

Deque

- 양 끝에서만 자료를 넣고 양 끝에서 뺄 수 있는 자료구조
- Double-ended queue의 약자이다.
- push_front: 덱의 앞에 자료를 넣는 연산
- push_back: 덱의 뒤에 자료를 넣는 연산
- pop_front: 덱의 앞에서 자료를 빼는 연산
- pop_back: 덱의 두에서 자료를 빼는 연산
- front: 덱의 가장 앞에 있는 자료를 보는 연산
- back: 덱의 가장 뒤에 있는 자료를 보는 연산

push_front

45



덱

<https://www.acmicpc.net/problem/10866>

- 덱을 구현하는 문제



<https://www.acmicpc.net/problem/10866>

- C++: <https://gist.github.com/Baekjoon/d0dbc7a158134ba6980f>
- Java: <https://gist.github.com/Baekjoon/3e49b05d92bf0ec4d41997bc9bb0f238>

문자열

아스키코드

ASCII

- 문자 인코딩 방법
- 외울 필요는 없다.
- 대표적인 아스키 코드
- '0' => 48
- 'A' => 65
- 'a' => 97
- 0은 아스키 코드로는 NULL을 나타낸다.
- 숫자가 저장되어있는데, 출력만 글자로 해주는 것으로 이해하면 편하다.

0 NULL
48 → '0'
'A'

아스키코드

ASCII

```
printf("%c", 65); AO  
printf("%c", 48);
```

- 위의 코드의 실행 결과는 AO 이다.

알파벳 개수

<https://www.acmicpc.net/problem/10808>

- 알파벳 소문자로 이루어진 단어에서 각 알파벳이 몇 개인지 구하는 문제
- C++: <https://gist.github.com/Baekjoon/cbb9fb7ae3f97e9950df>
- Java: <https://gist.github.com/Baekjoon/bb71a2992aa6ab52a0db83fc28fbd16e>

알파벳 찾기

<https://www.acmicpc.net/problem/10809>

- 알파벳 소문자로 이루어진 단어에서 각 알파벳이 몇 번째에 처음 등장하는지 찾는 문제
- C++: <https://gist.github.com/Baekjoon/ab427bf6fd25746fe77a>
- Java: <https://gist.github.com/Baekjoon/24ab43e6a651a12bf8eec0290beabab4>

문자열 분석

<https://www.acmicpc.net/problem/10820>

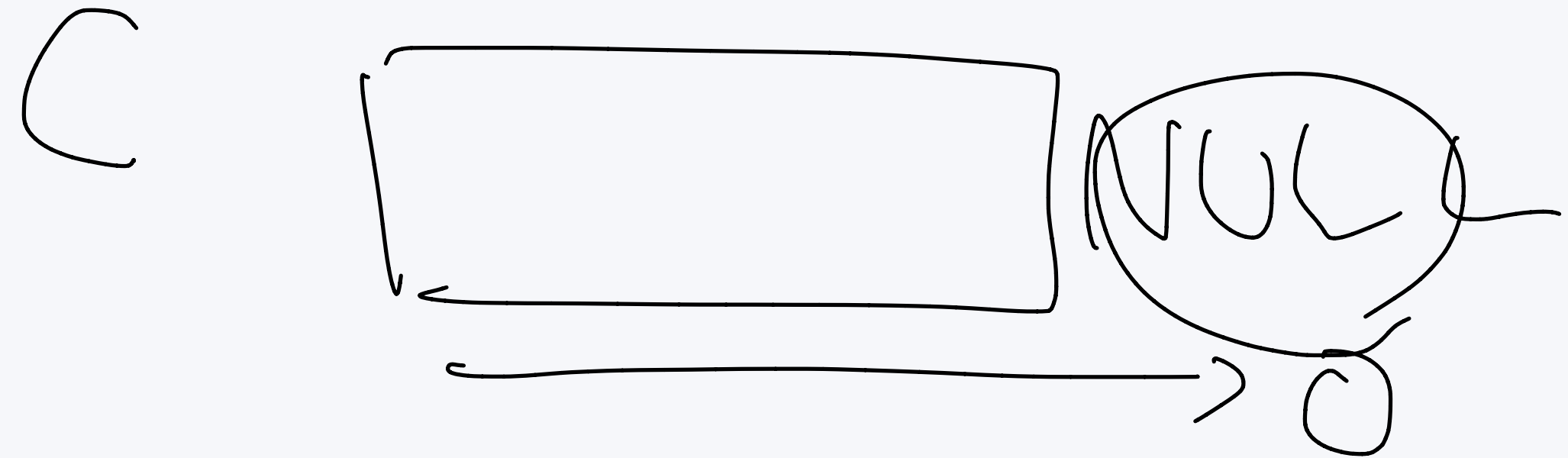
- 문자열 N개에 포함되어 있는 소문자, 대문자, 숫자, 공백의 개수를 세는 문제
- C++: <https://gist.github.com/Baekjoon/380a9ea81096b1b67230>
- Java: <https://gist.github.com/Baekjoon/fb2f6c0e2eaff1ea6909fbae8c9fc022>

단어 길이 재기

<https://www.acmicpc.net/problem/2743>

- 단어를 입력받고 길이를 재는 문제
- strlen이나 string의 length나 size를 이용하면 되지만 이런 것을 사용할 수 없는 경우에는
- 다음과 같이 길이를 잴 수 있다.

```
scanf("%s", s);  
int len = 0;  
for (int i = 0; s[i] != '\0'; i++) {  
    len += 1;  
}  
printf("%d\n", len);
```



O(1)

단어 길이 재기

<https://www.acmicpc.net/problem/2743>

- strlen 함수의 시간 복잡도는 $O(N)$ 이기 때문에, 다음과 같이 작성하면 $O(N^2)$ 코드이다.

```
for (int i=0; i<strlen(s); i++) {  
    // Do something  
}
```

$S.size()$

$O(N^2)$

- 아래와 같이 작성하는 것이 올바르다.

```
int len = strlen(s);  
for (int i=0; i<len; i++) {  
    // Do something  
}
```

$O(1)$

$O(N)$

단어 길이 재기

<https://www.acmicpc.net/problem/2743>

- C/C++: <https://gist.github.com/Baekjoon/21cf39dcebaa63691fe4>
- C/C++: <https://gist.github.com/Baekjoon/0eee47640b0712240b9c>
- C++: <https://gist.github.com/Baekjoon/5158790856b22ad55c2c>
- Java: <https://gist.github.com/Baekjoon/6ba7d385682a802ed5358e51a31b73cf>

ROT13

<https://www.acmicpc.net/problem/11655>

- ROT13으로 암호화하는 프로그램을 만드는 문제
- C++: <https://gist.github.com/Baekjoon/082fbecba925386573e6>
- Java: <https://gist.github.com/Baekjoon/529eda569f98d62d8e36d9e81d57c97e>

문자열 -> 정수

stoi, stol, stoll

- C++ string을 문자로 바꾸려면 stoi, stol, stoll 등등의 함수를 사용하면 된다.

- stoi: string -> int

- stol: string -> long

- stoll: string -> long long

- stof: string -> float

- stod: string -> double

- stold: string -> long double

- stoul: string -> unsigned long

- stoull: string -> unsigned long long

#include <string>

네 수

<https://www.acmicpc.net/problem/10824>

- 네 자연수 A, B, C, D가 주어진다. 이 때, A와 B를 붙인 수와 C와 D를 붙인 수의 합을 구하는 문제
- C++: <https://gist.github.com/Baekjoon/2d6a92f0db47234a7b4e>
- Java: <https://gist.github.com/Baekjoon/53307ff9b60d7f97b6846660e2d4ceb7>

정수 -> 문자열

to_string

- to_string 함수를 사용하면 된다.

정수 \Rightarrow 문자열

<String>

접미사 배열

<https://www.acmicpc.net/problem/11656>

- 접미사 배열은 문자열 S의 모든 접미사를 사전순으로 정렬해 놓은 배열이다.
- baekjoon의 접미사는 baekjoon, aekjoon, ekjoon, kjoon, joon, oon, on, n 으로 총 8가지가 있고, 이를 사전순으로 정렬하면, aekjoon, baekjoon, ekjoon, joon, kjoon, n, on, oon이 된다.
- 문자열 S가 주어졌을 때, 모든 접미사를 사전순으로 정렬한 다음 출력하는 프로그램을 작성하시오.

접미사 배열

<https://www.acmicpc.net/problem/11656>

- 문자열의 부분 문자열은 substr를 이용해서 구할 수 있다.
- C++: <https://gist.github.com/Baekjoon/f4fb6c40f6ed89b315c0>
- Java: <https://gist.github.com/Baekjoon/2a4e6ce84b26184c341defbddd2d6dc67>