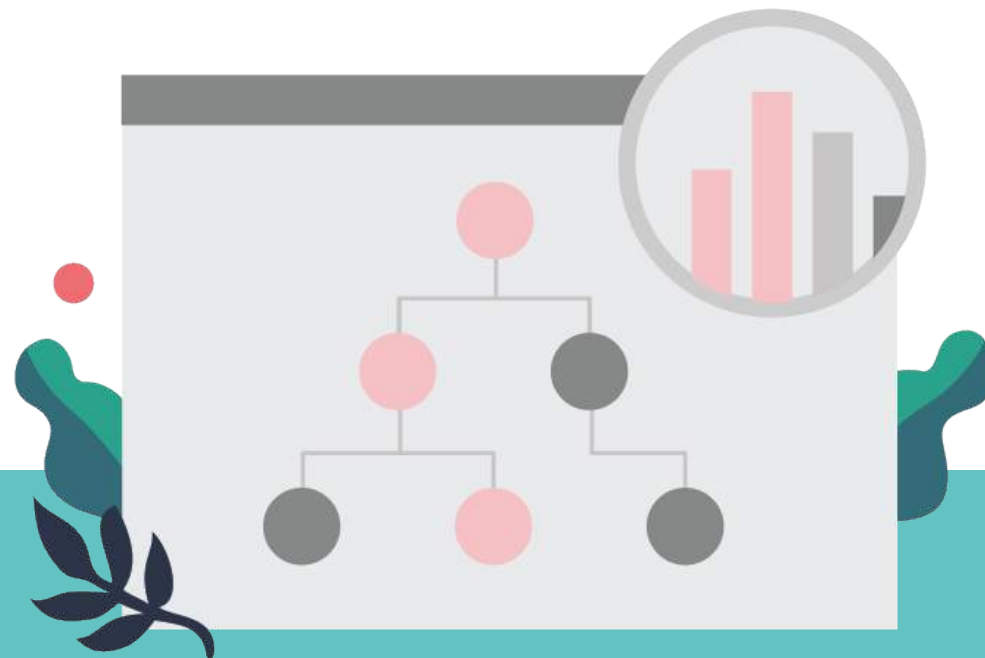


/* 데이터 구조 및 알고리즘 */

신현규 강사, 화/목 20:00

스택, 큐



/* elice */

목차

01 대표적인 자료구조

02 스택

03 큐

지난시간 요약

이번 과정의 목표

목적 달성을 위한 연산 횟수를 줄이는 자료구조 디자인

연산 횟수를 줄이면 장땡인가 ?

반드시 그런 것은 아니지만, 적어도 이번 과정에서는 Yes

리스트와 링크드 리스트

리스트

링크드 리스트

장점	i번째 원소의 값 접근이 빠름 연속된 값 읽기가 빠름	원소의 삽입 / 삭제가 빠름
단점	원소의 삽입 / 삭제가 느림	i번째 원소의 값 접근이 느림 연속된 값 읽기가 느림

주문 처리 시스템

리스트

링크드 리스트

addOrder

removeOrder

getOrder

주문 처리 시스템

리스트

링크드 리스트

addOrder

끝에 하나 추가

removeOrder myList.remove(orderId)

getOrder

몇 번째인지 반환

주문 처리 시스템

리스트

링크드 리스트

addOrder

끝에 하나 추가

끝에 하나 추가

removeOrder

myList.remove(orderId)

따라가면서 찾아 지운다 ?

getOrder

몇 번째인지 반환

몇 번째인지 반환

링크드 리스트의 removeOrder

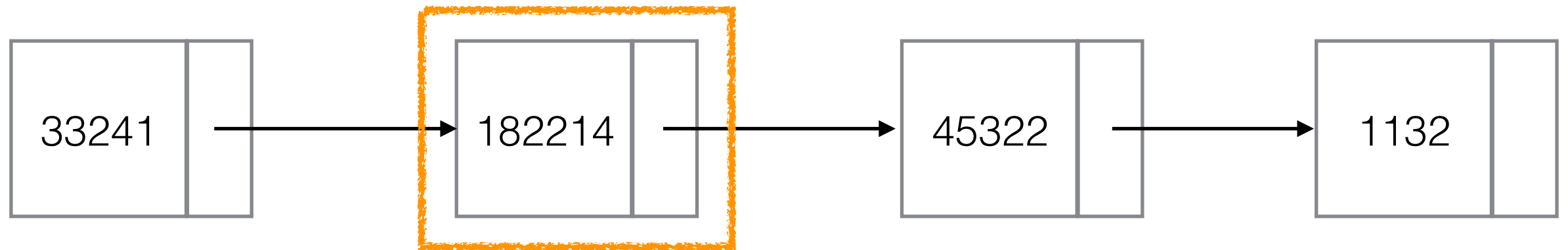
Dictionary를 사용하면 order ID로 원소를 바로 알 수 있음



order ID = 182214

링크드 리스트의 removeOrder

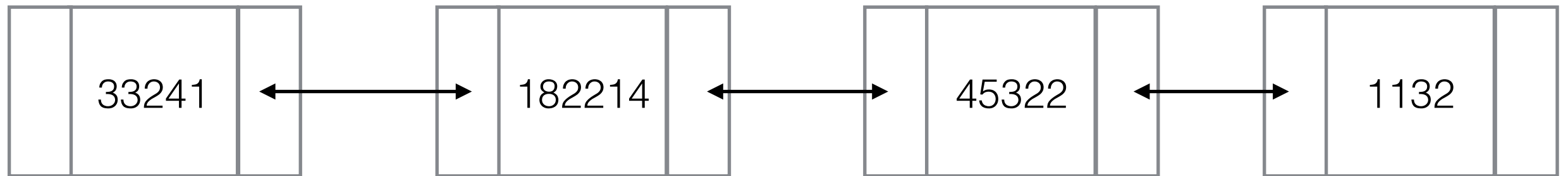
Dictionary를 사용하면 order ID로 원소를 바로 알 수 있음



order ID = 182214

링크드 리스트의 removeOrder

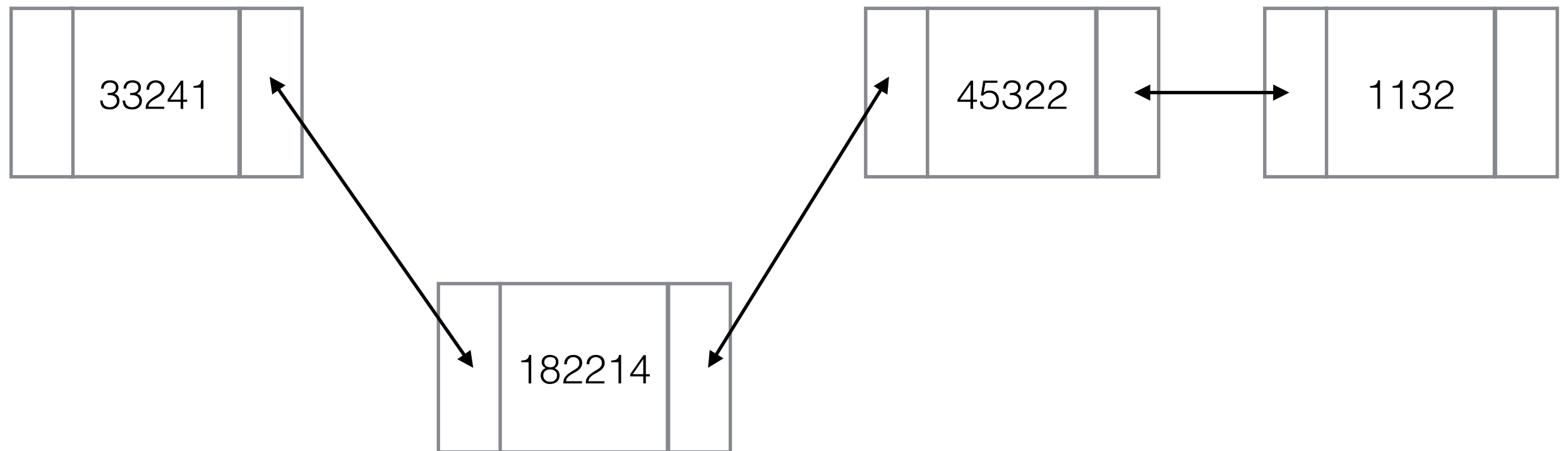
Dictionary를 사용하면 order ID로 원소를 바로 알 수 있음



order ID = 182214

링크드 리스트의 removeOrder

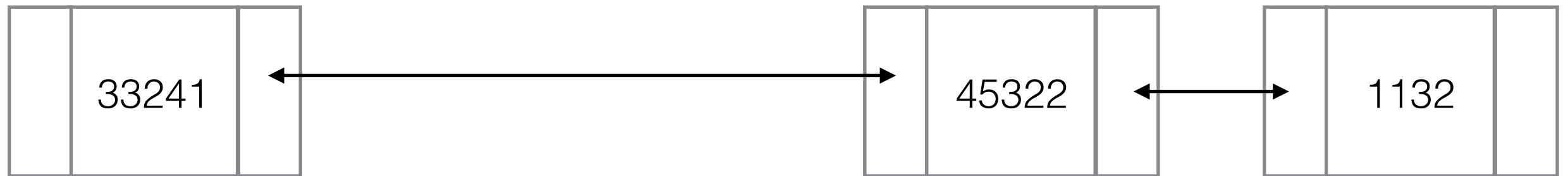
Dictionary를 사용하면 order ID로 원소를 바로 알 수 있음



order ID = 182214

링크드 리스트의 removeOrder

Dictionary를 사용하면 order ID로 원소를 바로 알 수 있음



order ID = 182214

성능 비교

리스트

```
--- 주문 조치가 매우 많을 경우 테스트 --- Testc
Testcase 12: accept (5 points, 445.613 ms),
Testcase 13: accept (5 points, 601.689 ms),
Testcase 14: accept (5 points, 693.773 ms),
Testcase 15: accept (5 points, 934.603 ms),
--- 주문 조치가 별로 없을 경우 테스트 --- Testc
Testcase 17: accept (5 points, 1249.445 ms),
Testcase 18: accept (5 points, 2187.972 ms),
Testcase 19: accept (5 points, 3384.084 ms),
Testcase 20: accept (5 points, 4902.619 ms),
```

링크드 리스트

```
--- 주문 조치가 매우 많을 경우 테스트 --- Testc
Testcase 12: accept (5 points, 782.010 ms),
Testcase 13: accept (5 points, 989.530 ms),
Testcase 14: accept (5 points, 1180.183 ms),
Testcase 15: accept (5 points, 1525.306 ms),
--- 주문 조치가 별로 없을 경우 테스트 --- Testc
Testcase 17: accept (5 points, 202.781 ms),
Testcase 18: accept (5 points, 306.003 ms),
Testcase 19: accept (5 points, 483.448 ms),
Testcase 20: accept (5 points, 660.719 ms),
```

이 문제에서의 메시지

알고리즘이 같아도 데이터에 따라 성능이 다르다

기업이 고객의 데이터를 분석하는 주된 이유

중요한 것

답안을 보는 것은 편법이 절대 아님

대부분의 경우에는 조교의 답이 내 답보다 깔끔하다

다른 사람의 코드를 보는 것은
코딩 실력 향상의 **확실한** 지름길

주차별 커리큘럼

1주차 과정 소개, 배열, 연결리스트, 클래스

2주차 스택, 큐, 해싱

3주차 시간복잡도

4주차 트리, 트리순회, 재귀호출

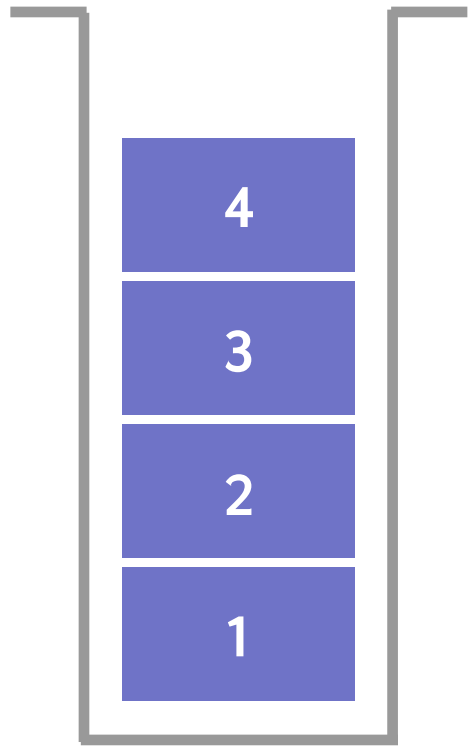
5주차 힙

6주차 그래프 소개, DFS

7주차 그래프 심화, BFS

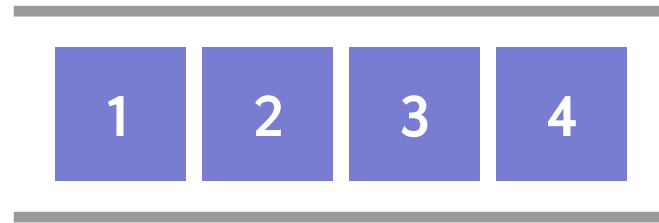
8주차 강의 요약, 알고리즘 과정 소개

대표적인 자료구조



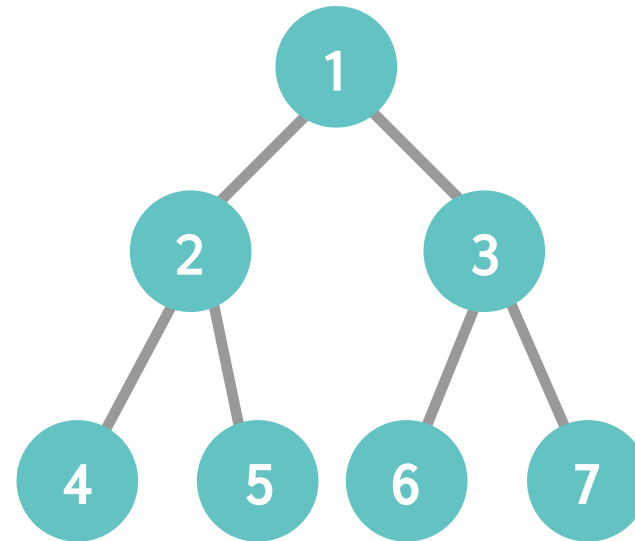
스택 (Stack)

Last In First Out

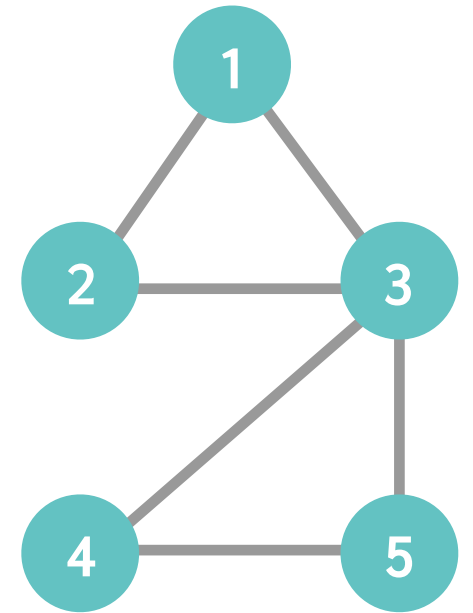


큐 (Queue)

First In First Out

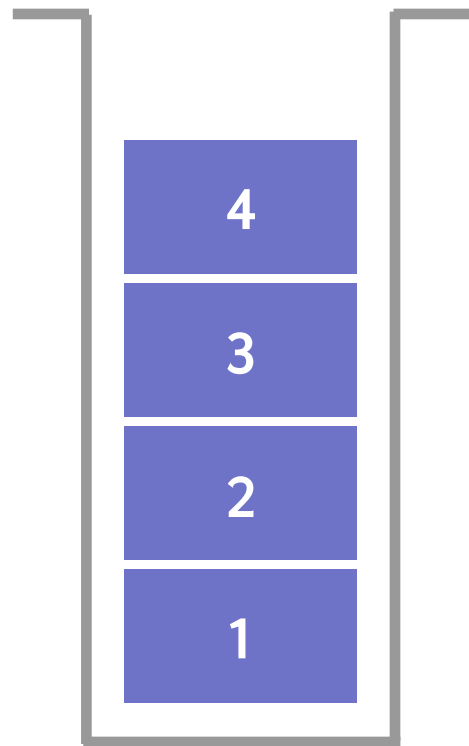


트리 (Tree)



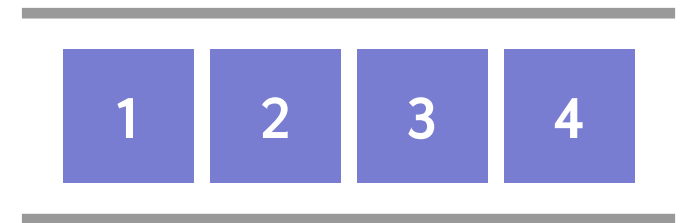
그래프 (Graph)

스택, 큐



Stack

Last In First Out

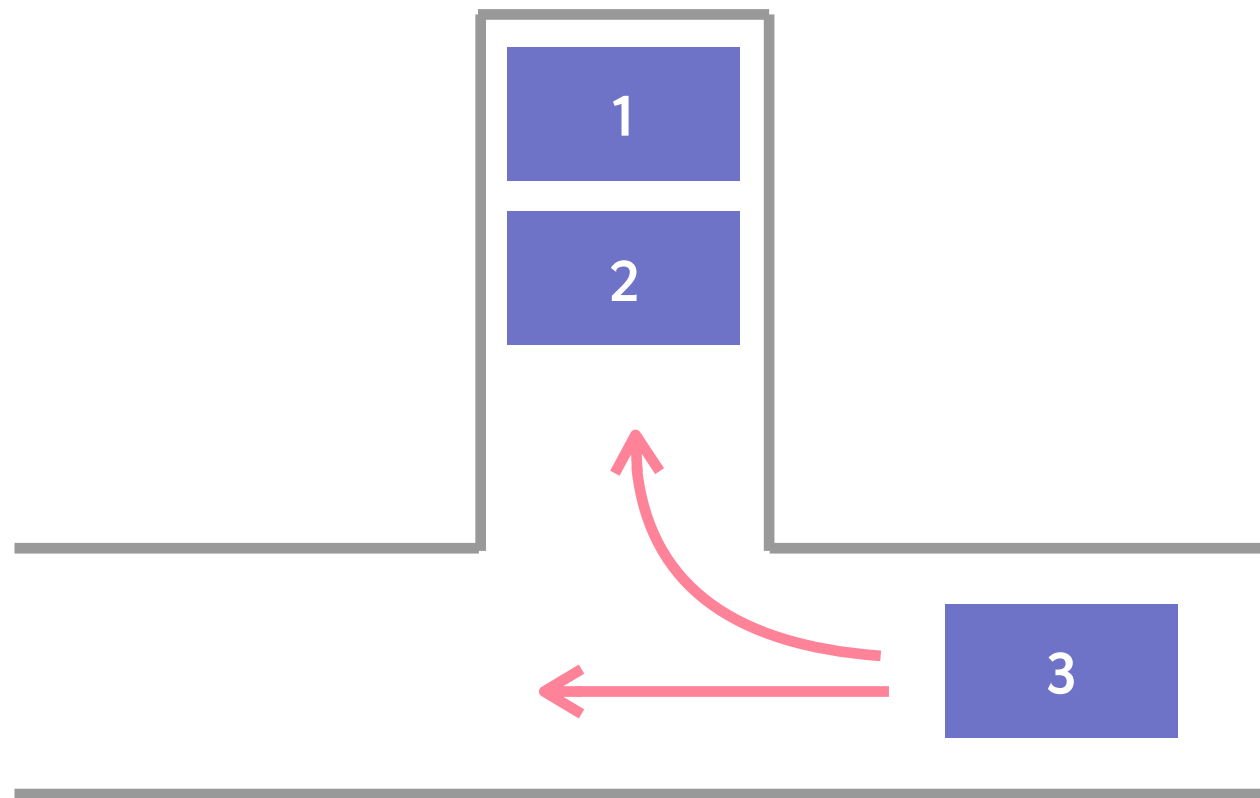


Queue

First In First Out

스택, 큐 끝

그럼 애는 왜 유명하지 않은가 ?



스태큐 : 신현규(26, 강사) 씨가 발명

그럼 애는 왜 유명하지 않은가 ?

어디다가 써야할지 모르겠으니까

그럼 애는 왜 유명하지 않은가 ?

어디다가 써야할지 모르겠으니까

그러면 **스택, 큐**는 어디다가 쓰는가 ?

흔히 하는 착각

특정 자료구조가 뭔지 아는 것은 중요하지 않음

이 자료구조를 **내 의도에 맞게**

쓸 줄 아는 능력이 중요

디자인을 많이 해봐야 실력이 향상됨

[문제 1] 스택 구현하기



```
/* elice */
```


[문제 2] 올바른 괄호인지 판단하기

괄호쌍이 주어질 때, 올바른 괄호인지 판단

입력의 예

(())()()

((()))

()

출력의 예

YES

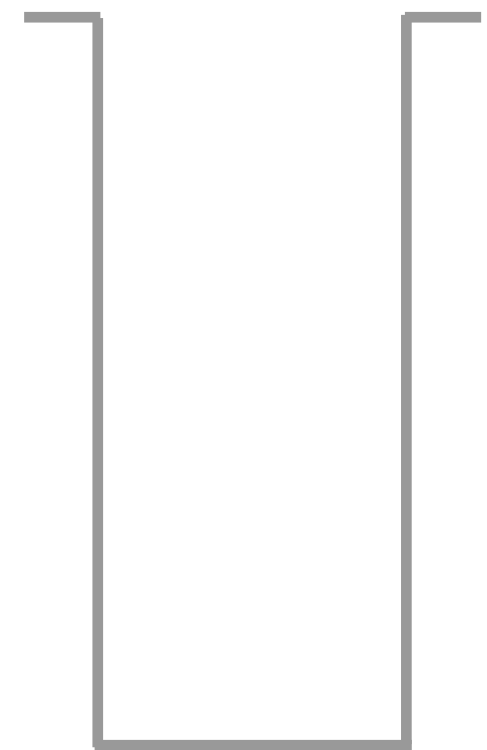
YES

NO

[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

((()) ())



Stack

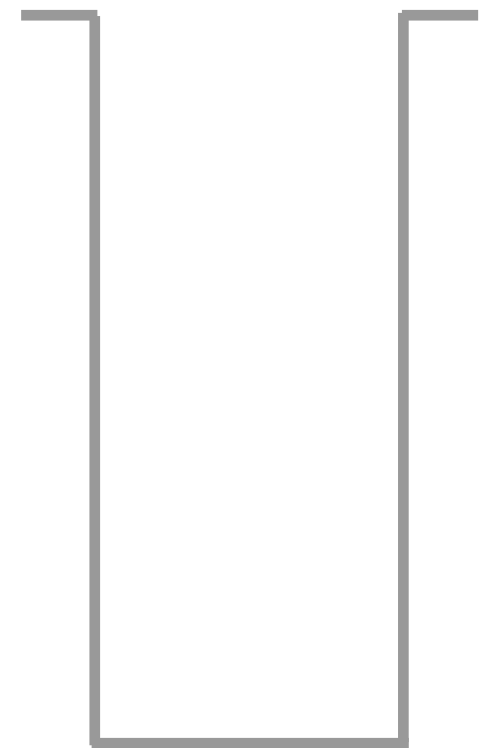
Last In First Out

[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용



((()) ())



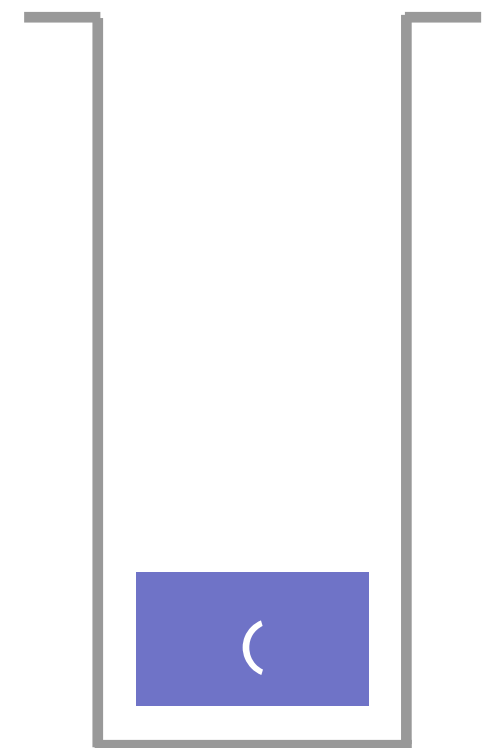
Stack

Last In First Out

[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

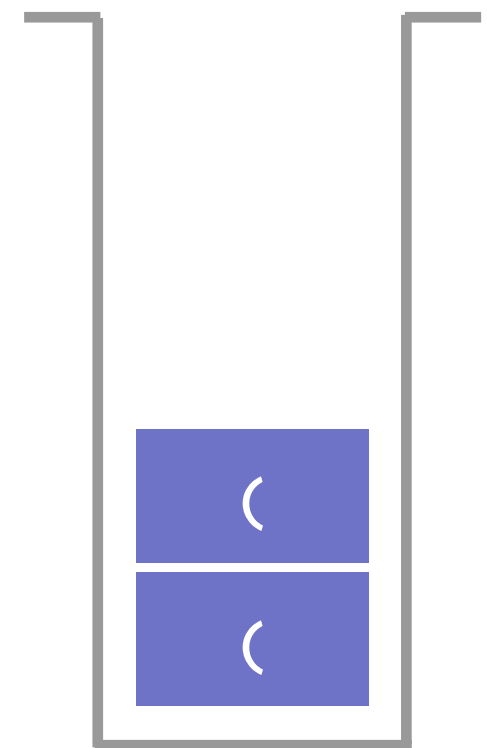

((()) ())



[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

((()) ())



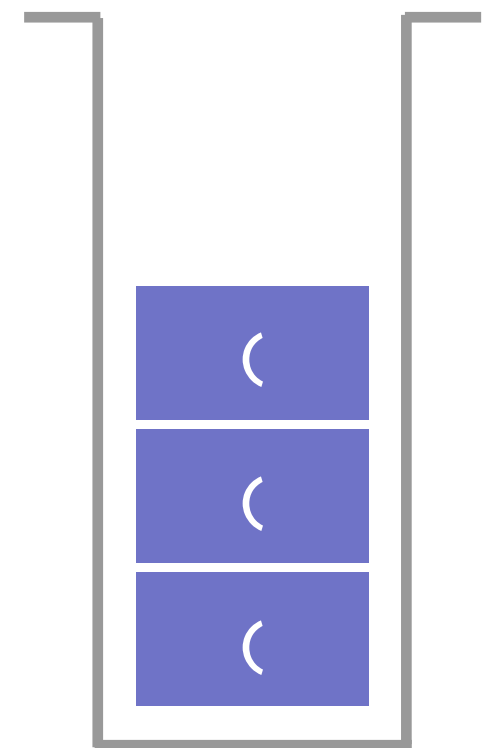

Stack

Last In First Out

[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

((()) ())



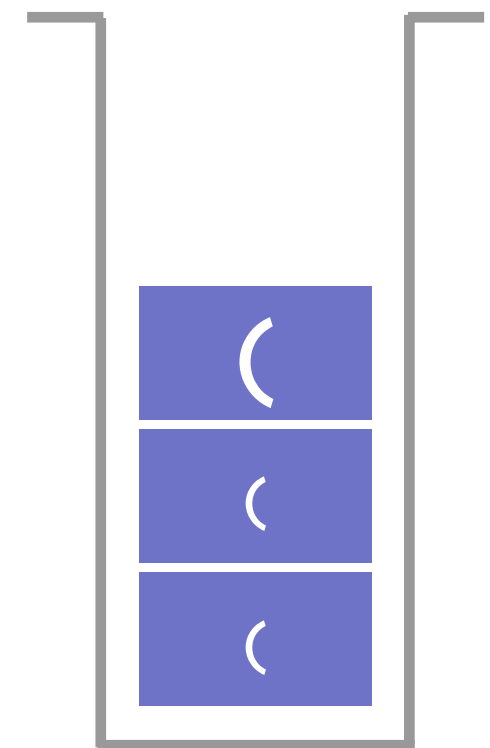

Stack

Last In First Out

[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

((()) ())



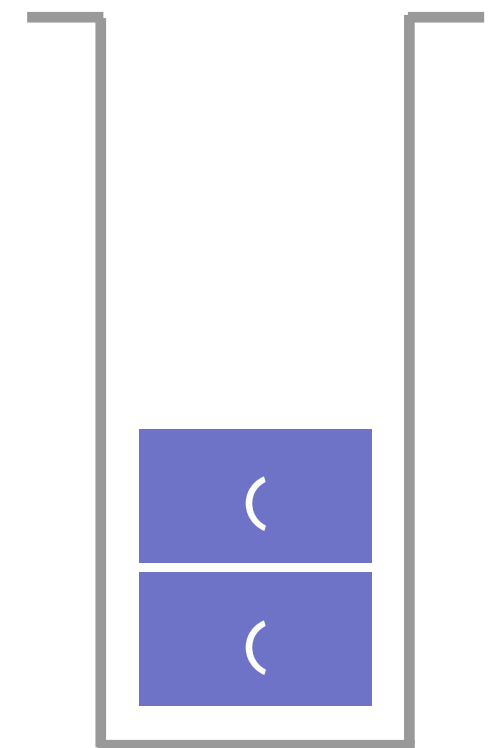

Stack

Last In First Out

[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

((()) ())



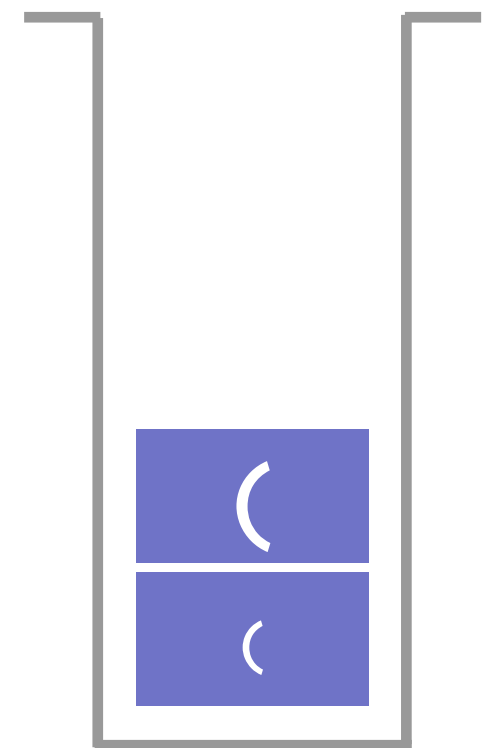

Stack

Last In First Out

[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

((()) ())



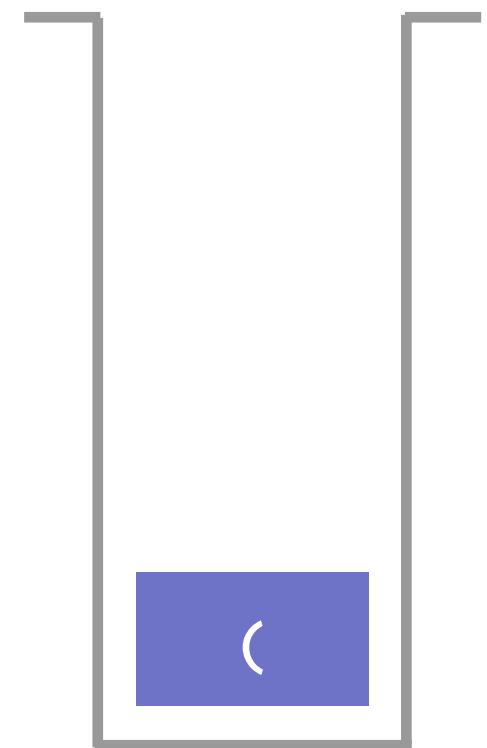
Stack

Last In First Out

[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

((()) ())



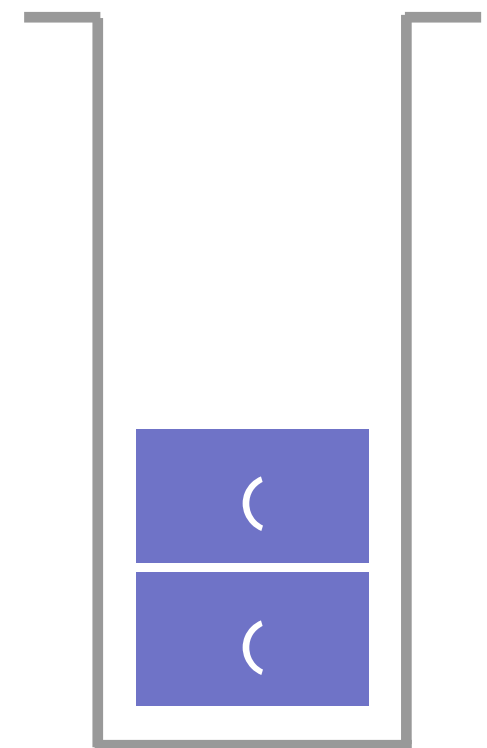

Stack

Last In First Out

[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

((()) ())



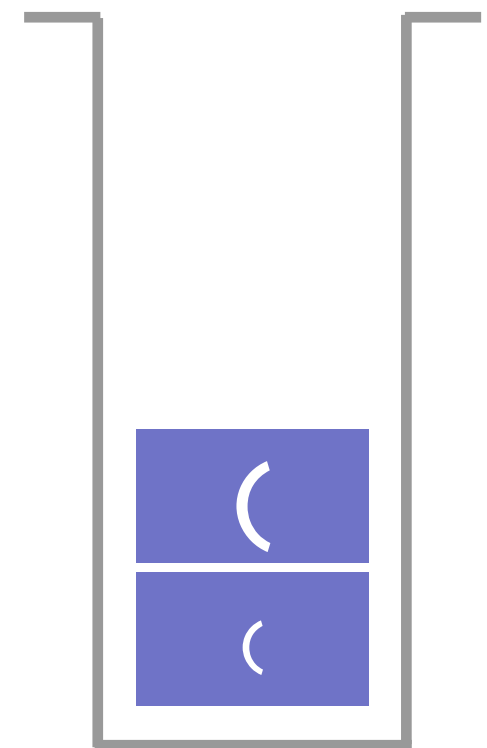

Stack

Last In First Out

[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

((()) ())



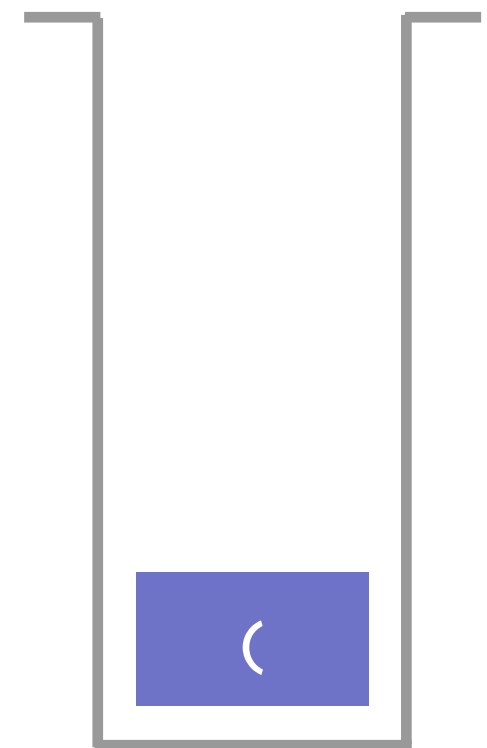

Stack

Last In First Out

[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

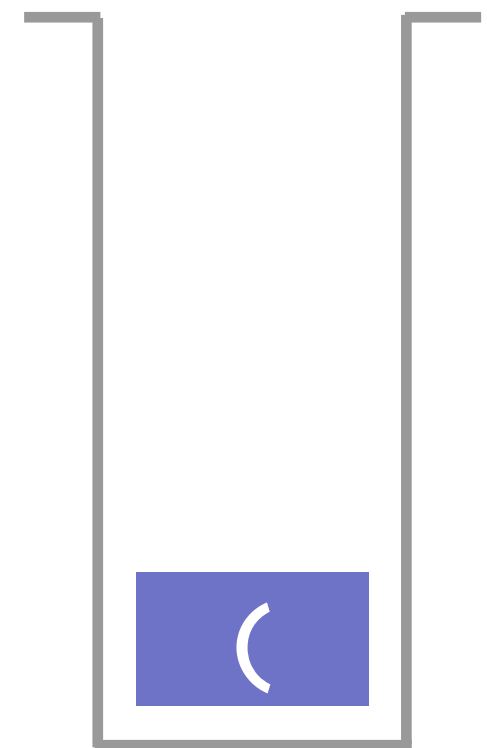

((()) ())



[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

((()) ())



Stack

Last In First Out

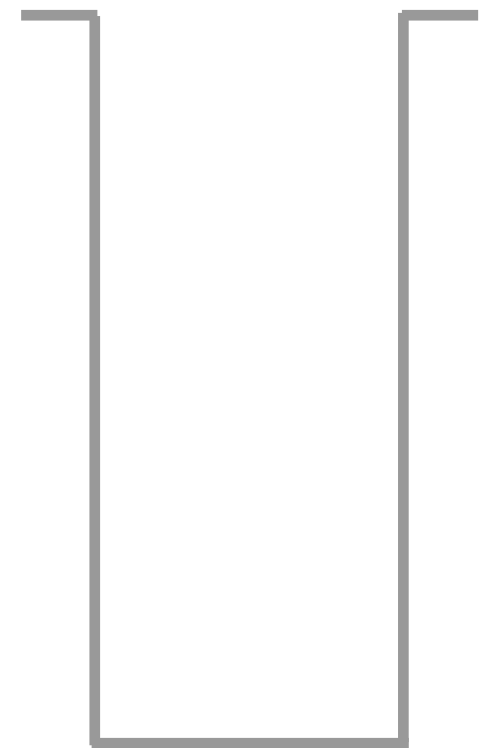
[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

YES



((()) ())



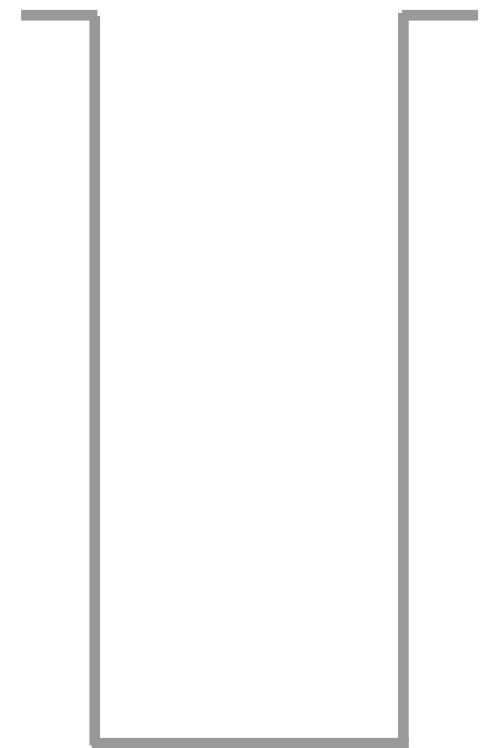
Stack

Last In First Out

[문제 2] 올바른 괄호인지 판단하기

올바르지 않은 경우에는 어떻게 되는가?

((())



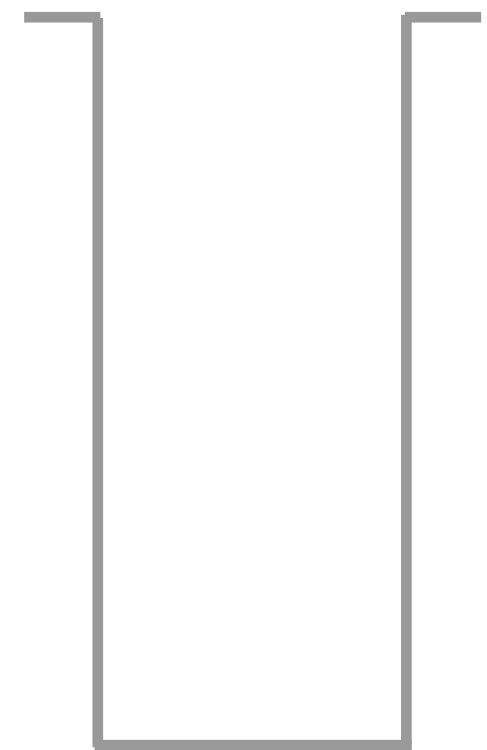
Stack

Last In First Out

[문제 2] 올바른 괄호인지 판단하기

올바르지 않은 경우에는 어떻게 되는가?

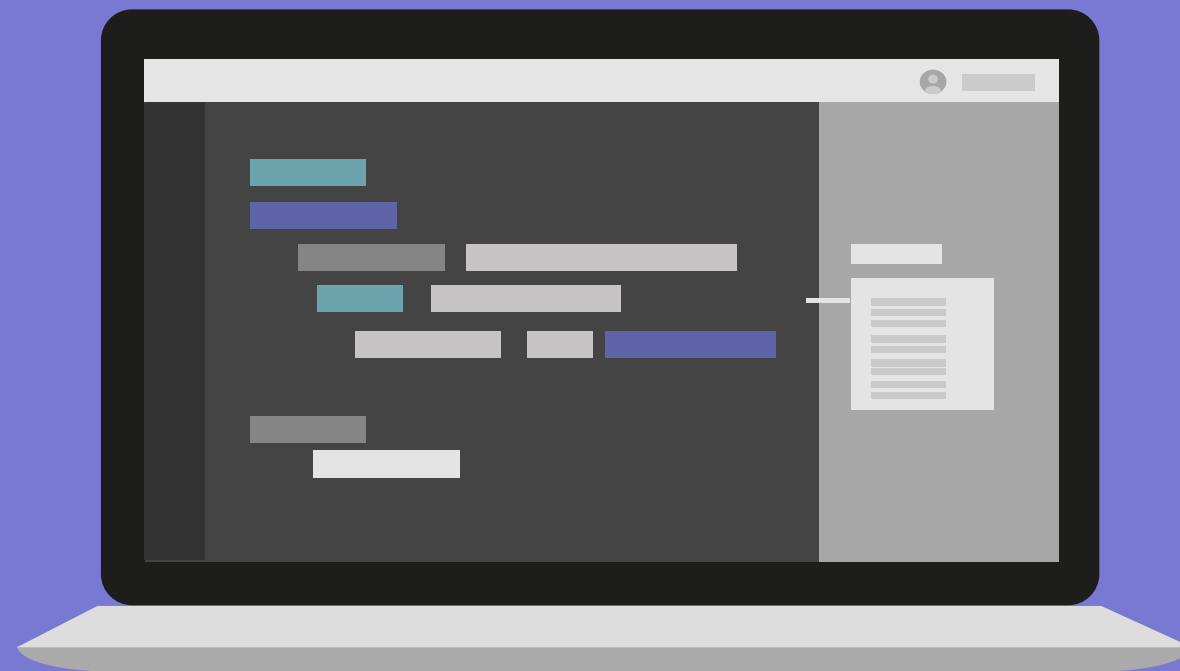
(() ()) ()



Stack

Last In First Out

[문제 2] 올바른 괄호인지 판단하기



```
/* elice */
```

그래서 스택은 언제 쓰나요 ?

스택은 자료구조

그래서 스택은 언제 쓰나요 ?

스택은 자료구조

무슨 자료를 저장하는가 ?

그래서 스택은 언제 쓰나요 ?

스택은 자료구조

무슨 자료를 저장하는가 ?

상태 (Status)

실생활의 예



실생활의 예



실생활의 예



실생활의 예



1. 미역
2. 국간장
3. 후추
4. 고기
5. 냄비

실생활의 예



1. 마역
2. 국간장
3. 후추
4. 고기
5. 냄비

실생활의 예



1. 마역
2. 국간장
3. 후추
4. 고기
5. 냄비

1. 문 열기
2. 서랍 열기
3. 카드 꺼내기
4. 집 나오기

실생활의 예



1. 마역
2. 국간장
3. 후추
4. 고기
5. 냄비

1. 문 열기
2. 서랍 열기
3. 카드 꺼내기
4. 집 나오기

실생활의 예



1. 마역
2. 국간장
3. 후추
4. 고기
5. 냄비



1. 열쇠 찾기



1. 문 열기
2. 서랍 열기
3. 카드 꺼내기
4. 집 나오기

실생활의 예



1. 마역
2. 국간장
3. 후추
4. 고기
5. 냄비



1. 열쇠 찾기



1. 문 열기
2. 서랍 열기
3. 카드 꺼내기
4. 집 나오기

실생활의 예



1. 마역
2. 국간장
3. 후추
4. 고기
5. 냄비



1. 열쇠 찾기



1. 문 열기
2. 서랍 열기
3. 카드 꺼내기
4. 집 나오기

실생활의 예



1. 마역
2. 국간장
3. 후추
4. 고기
5. 냄비



1. 열쇠 찾기



1. 문 열기
2. 셔랍 열기
3. 카드 꺼내기
4. 집 나오기

실생활의 예



1. 마역
2. 국간장
3. 후추
4. 고기
5. 냄비



1. 열쇠 찾기



1. 문 열기
2. 셔랍 열기
3. 카드 꺼내기
4. 짐 나오기

실생활의 예



1. 마~~역~~
2. 국~~간~~장
3. 후~~추~~
4. 고~~기~~
5. 냄~~바~~



1. 열~~쇠~~ 찾~~기~~



1. 문 ~~열~~기
2. 셔~~랍~~ 열~~기~~
3. 카~~드~~ 꺼~~내~~기
4. 집 ~~나~~오~~기~~

실생활의 예



1. ~~미역~~
2. ~~국간장~~
3. ~~후추~~
4. ~~고기~~
5. ~~냄바~~



1. ~~열쇠 찾기~~



1. ~~문 열기~~
2. ~~셔랍 열기~~
3. ~~카드 꺼내기~~
4. ~~집 나오기~~

실생활의 예 : 자료구조 관점



1. 미역
2. 국간장
3. 후추
4. 고기
5. 냄비



1. 열쇠 찾기



1. 문 열기
2. 서랍 열기
3. 카드 꺼내기
4. 집 나오기

실생활의 예 : 자료구조 관점



1. 미역
2. 국간장
3. 후추
4. 고기
5. 냄비



1. 열쇠 찾기



1. 문 열기
2. 서랍 열기
3. 카드 꺼내기
4. 집 나오기

미역국 끓이기

실생활의 예 : 자료구조 관점



1. 미역
2. 국간장
3. 후추
4. 고기
5. 냄비



1. 열쇠 찾기



1. 문 열기
2. 서랍 열기
3. 카드 꺼내기
4. 집 나오기

마트. (4)

미역국 끓이기

실생활의 예 : 자료구조 관점



1. 미역
2. 국간장
3. 후추
4. 고기
5. 냄비



1. 열쇠 찾기



1. 문 열기
2. 서랍 열기
3. 카드 꺼내기
4. 집 나오기

집. (2)

마트. (4)

미역국 끓이기

실생활의 예 : 자료구조 관점



1. 미역
2. 국간장
3. 후추
4. 고기
5. 냄비



1. 열쇠 찾기



1. 문 열기
2. 서랍 열기
3. 카드 꺼내기
4. 집 나오기

집. (2)

마트. (4)

미역국 끓이기

실생활의 예 : 자료구조 관점



1. 미역
2. 국간장
3. 후추
4. 고기
5. 냄비



1. 열쇠 찾기



1. 문 열기
2. 셔랍 열기
3. 카드 꺼내기
4. 집 나오기

마트. (4)

미역국 끓이기

실생활의 예 : 자료구조 관점



1. 미역
2. 국간장
3. 후추
4. 고기
5. 냄바



1. 열쇠 찾기



1. 문 열기
2. 셔랍 열기
3. 카드 꺼내기
4. 집 나오기

미역국 끓이기

실생활의 예 : 자료구조 관점



1. 미역
2. 국간장
3. 후추
4. 고기
5. 냄바



1. 열쇠 찾기



1. 문 열기
2. 셔랍 열기
3. 카드 꺼내기
4. 집 나오기

그래서 스택은 언제 쓰나요 ?

“상태”의 의존상태가 생길 때

A라는 일을 마치기 위해서 B라는 일을 먼저 끝내야 할 때

그래서 스택은 언제 쓰나요 ?

“상태”의 의존상태가 생길 때

A라는 일을 마치기 위해서 B라는 일을 먼저 끝내야 할 때

재귀호출

Computational Thinking

실생활의 예 : 자료구조 관점 (2)

실생활의 예 (2)



실생활의 예 (2)



실생활의 예 (2)



실생활의 예 (2)



1. 미역
2. 국간장
3. 후추
4. 고기
5. 냄비

실생활의 예 (2)



1. 마역
2. 국간장
3. 후추
4. 고기
5. 냄비

실생활의 예 (2)



1. 마역
2. 국간장
3. 후추
4. 고기
5. 냄비



1. 옷 찾기



실생활의 예 (2)



1. 마역
2. 국간장
3. 후추
4. 고기
5. 냄비



1. 옷 찾기



실생활의 예 (2)



1. 마역
2. 국간장
3. 후추
4. 고기
5. 냄비



1. 옷 찾기



1. 방세 입금하기

실생활의 예 (2)



1. 마역
2. 국간장
3. 후추
4. 고기
5. 냄바



1. 옷 찾기



1. 방세 입금하기

실생활의 예 (2)



1. 마역
2. 국간장
3. 후추
4. 고기
5. 냄바



1. 옷 찾기



1. 방세 입금하기

실생활의 예 (2)



1. 마역
2. 국간장
3. 후추
4. 고기
5. 냄바



1. 옷 찾기



1. 방세 입금하기

실생활의 예 (2)



1. 마역
2. 국간장
3. 후추
4. 고기
5. 냄바



1. 옷 찾기



1. 방세 입금하기

실생활의 예 (2)



1. 마역
2. 국간장
3. 후추
4. 고기
5. 냄바



1. 옷 찾기



1. 방세 입금하기

실생활의 예 (2) : 자료구조 관점



1. 미역
2. 국간장
3. 후추
4. 고기
5. 냄비

실생활의 예 (2) : 자료구조 관점



미역국
끓이기

1. 미역
2. 국간장
3. 후추
4. 고기
5. 냄비

실생활의 예 (2) : 자료구조 관점



미역국
끓이기

1. 미역
2. 국간장
3. 후추
4. 고기
5. 냄비

실생활의 예 (2) : 자료구조 관점



1. 미역
2. 국간장
3. 후추
4. 고기
5. 냄비



1. 옷 찾기



미역국
끓이기

옷
찾기

실생활의 예 (2) : 자료구조 관점



1. 미역
2. 국간장
3. 후추
4. 고기
5. 냄비



1. 옷 찾기



미역국
끓이기

옷
찾기

실생활의 예 (2) : 자료구조 관점



1. 마역
2. 국간장
3. 후추
4. 고기
5. 냄비



1. 옷 찾기



1. 월세 입금하기

미역국
끓이기

옷
찾기

월세
입금

실생활의 예 (2) : 자료구조 관점



1. 미역
2. 국간장
3. 후추
4. 고기
5. 냄바



1. 옷 찾기



1. 월세 입금하기

미역국
끓이기

옷
찾기

월세
입금

실생활의 예 (2) : 자료구조 관점



1. 마역
2. 국간장
3. 후추
4. 고기
5. 냄바



1. 옷 찾기



1. 월세 입금하기

옷
찾기

월세
입금

실생활의 예 (2) : 자료구조 관점



1. 마역
2. 국간장
3. 후추
4. 고기
5. 냄바



1. 옷 찾기



1. 월세 입금하기

옷
찾기

월세
입금

실생활의 예 (2) : 자료구조 관점



1. 미역
2. 국간장
3. 후추
4. 고기
5. 냄바



1. 옷 찾기



1. 월세 입금하기

월세
입금

실생활의 예 (2) : 자료구조 관점



1. 미역
2. 국간장
3. 후추
4. 고기
5. 냄바



1. 옷 찾기



1. 월세 입금하기

월세
입금

실생활의 예 (2) : 자료구조 관점



1. 미역
2. 국간장
3. 후추
4. 고기
5. 냄바



1. 옷 찾기



1. 월세 입금하기

그래서 큐는 언제 쓰나요 ?

“상태”의 의존상태가 없을 때

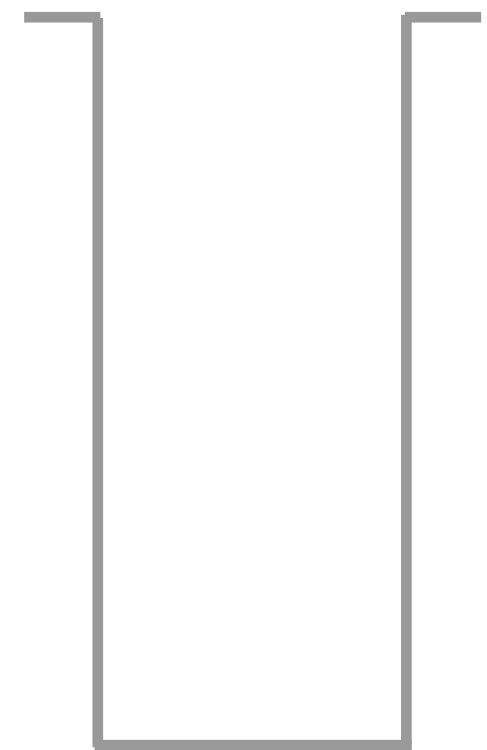
A와 B가 서로 관련이 없지만 모두 하긴 해야할 때

스케줄링, 병렬화

[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

((()) ())



Stack

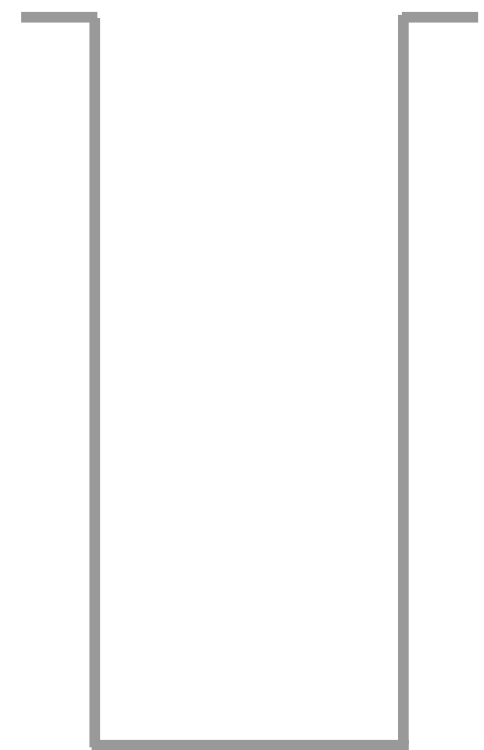
Last In First Out

[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용



((()) ())



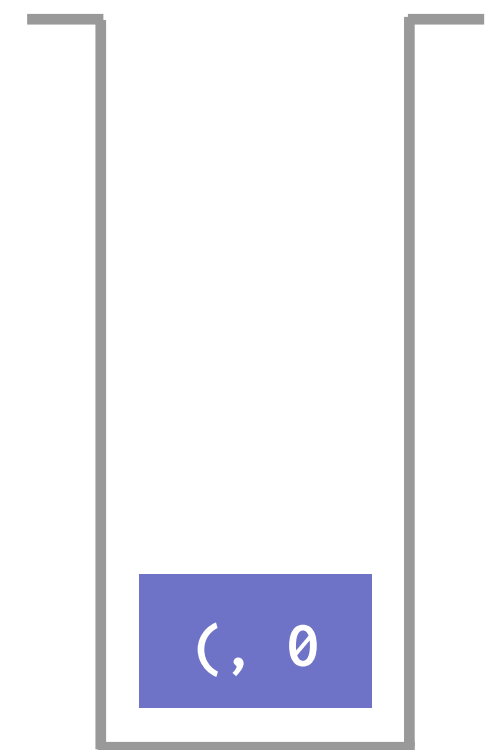
Stack

Last In First Out

[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

((()) ())



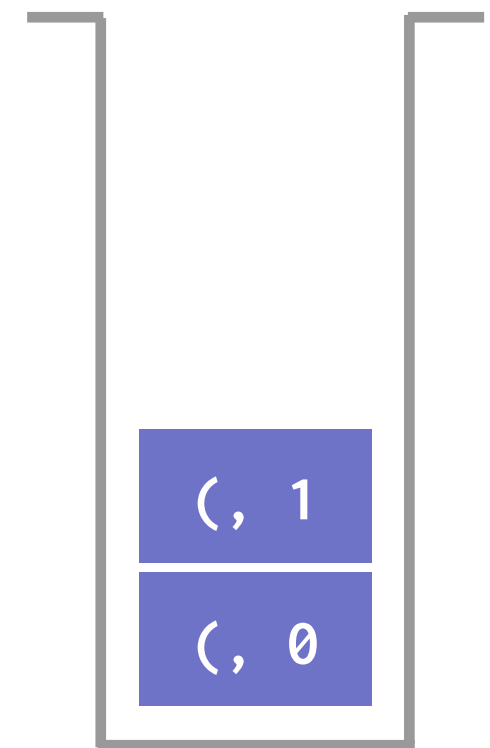

Stack

Last In First Out

[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

((()) ())



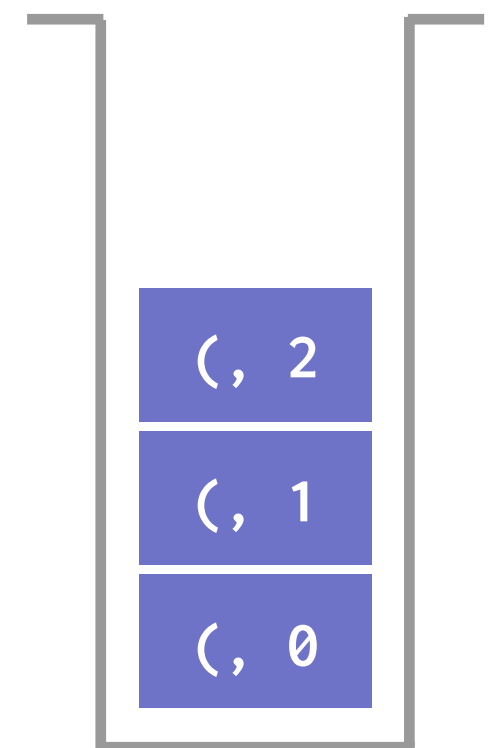

Stack

Last In First Out

[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

((()) ())



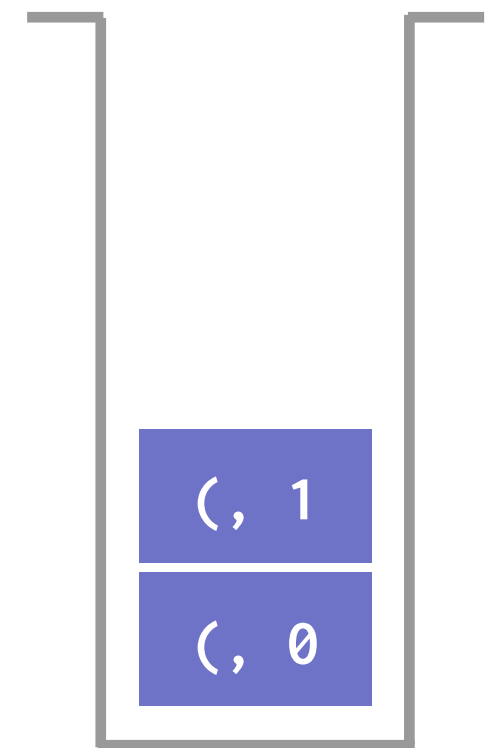

Stack

Last In First Out

[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

((()) ())



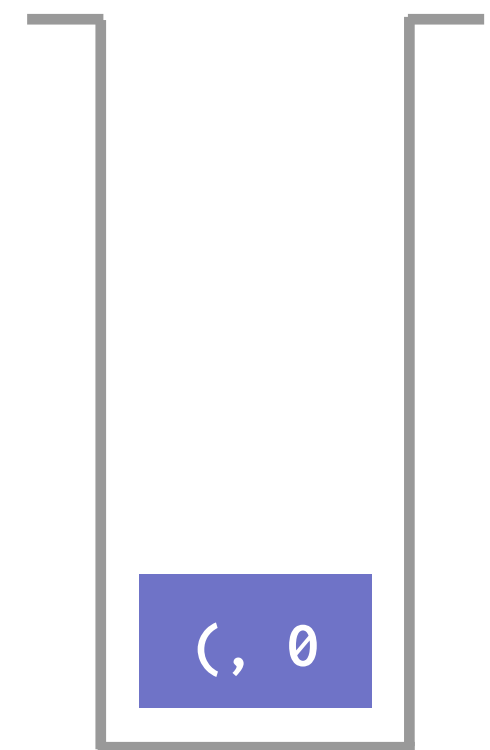
Stack

Last In First Out

[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

((()) ())



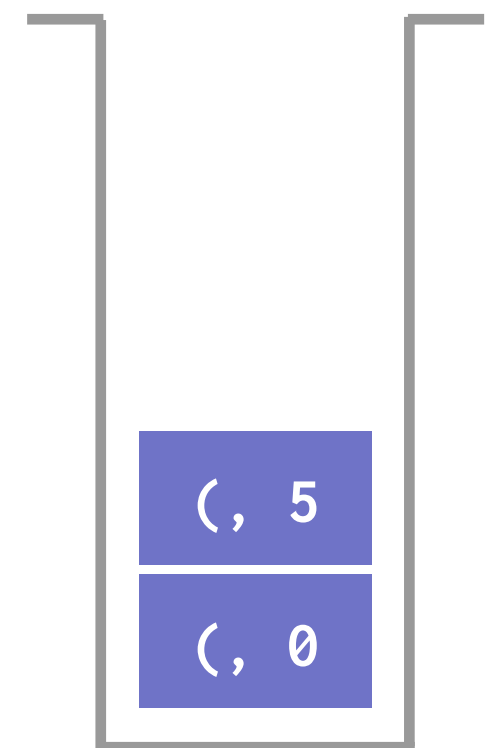
Stack

Last In First Out

[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

((()) ())



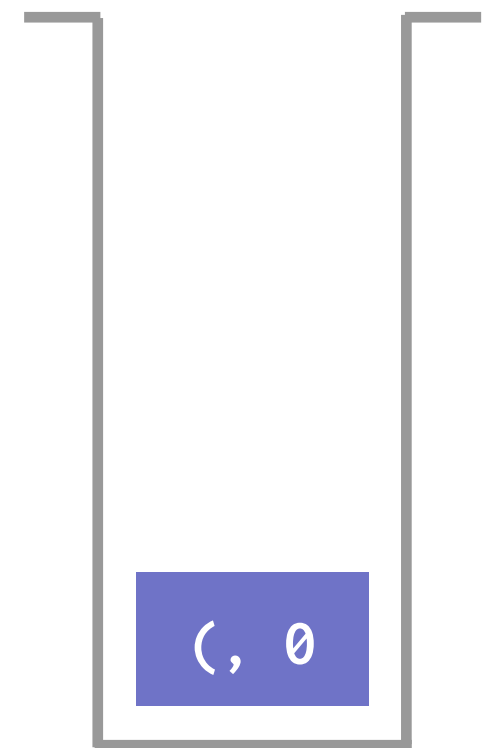
Stack

Last In First Out

[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

((()) ())



Stack

Last In First Out

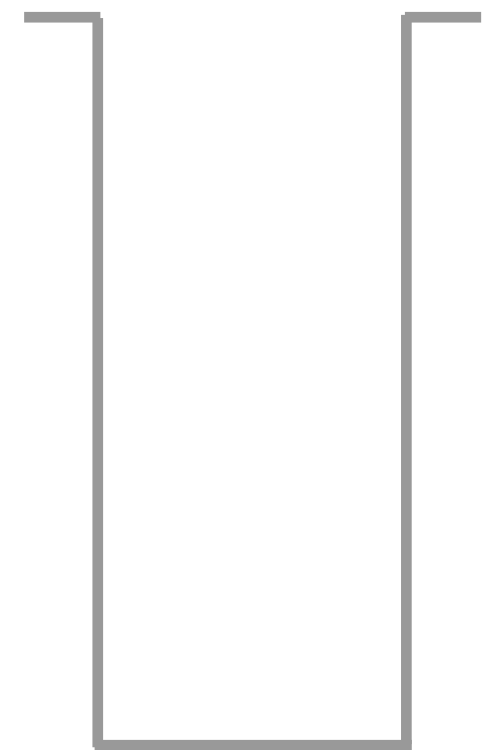
[문제 2] 올바른 괄호인지 판단하기

스택의 대표적인 활용

YES



((()) ())



Stack

Last In First Out

요약

선형 자료구조가 제일 어렵다

→ 목적이 모호하기 때문

요약

선형 자료구조가 제일 어렵다

→ 목적이 모호하기 때문

스택과 큐에는 많은 경우 “상태”를 저장한다

요약

선형 자료구조가 제일 어렵다

→ 목적이 모호하기 때문

스택과 큐에는 많은 경우 “상태”를 저장한다

스택과 큐는 그 용도가 다르다

→ 스택은 “상태”의 의존상태가 있을 때 (재귀호출)

→ 큐는 “상태”의 의존상태가 없을 때 (스케줄링, 병렬화)

감사합니다!

신현규

E-mail : hyungyu.sh@kaist.ac.kr

Kakao : yougatup

/* elice */

문의 및 연락처

academy.elice.io

contact@elice.io

facebook.com/elice.io

blog.naver.com/elicer