2024-1
기초 스터디

8. 스택, 큐, 덱

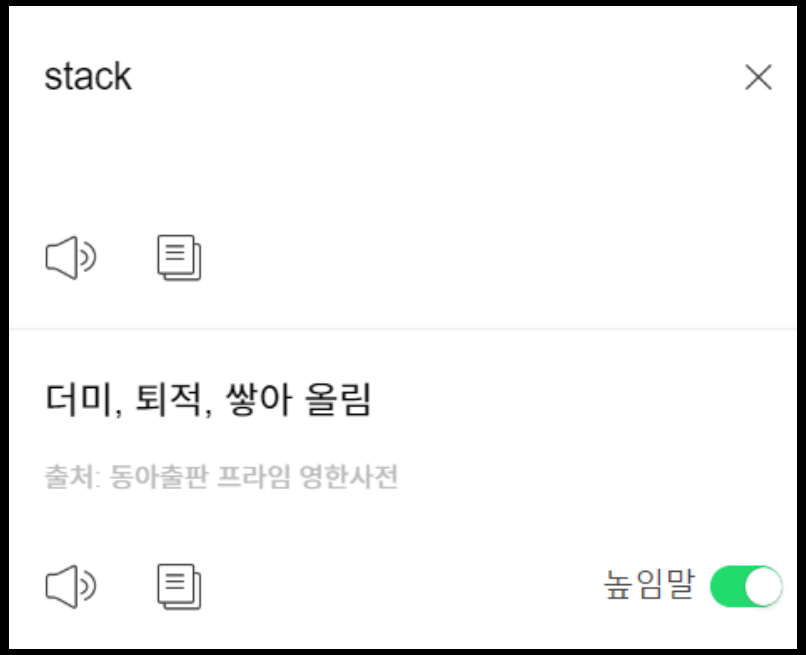
목차

자료 구조

- 자료(데이터)를 **어떤 구조에 맞춰 저장**하는 틀
- 데이터를 담고 있는 틀이기 때문에,
  데이터를 **넣는 동작**과 **빼는 동작**이 정의된다.

스택

- 데이터를 아래에서 위로 **쌓아올린 형태**의 자료구조



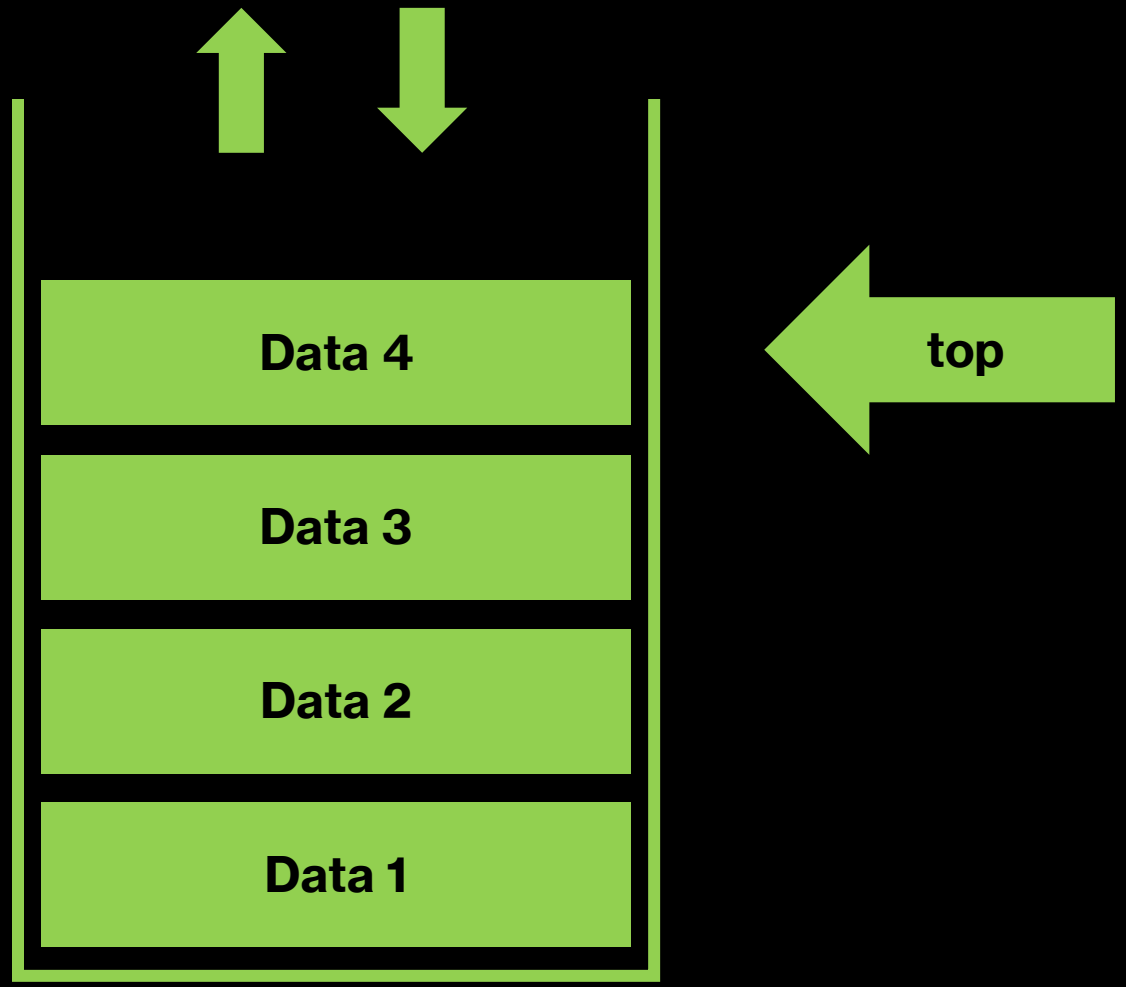
스택

- 데이터를 저장할 때 **아래부터 쌓으면서 저장**한다.
  → push
  데이터를 뺄 때는 **제일 위에 있는 데이터를 뺀다.**
  → pop

- 마지막에 들어간 데이터가 먼저 나오는 형태
  → (Last In First Out == **LIFO**)

스택 – push

스택 - pop



스택 - 구현

- **리스트**를 스택처럼 쓸 수 있다.
- 리스트의 pop() 연산이 **O(1)** 시간에 수행되기 때문
  (pop 메서드에 아무런 인자를 넣지 않았을 때 입니다!)

스택 - 구현

- 데이터 추가 (push) : **append()**
- 데이터 삭제 (pop) : **pop()**

```
1  stack = []
2  stack.append(1)
3  stack.append(2)
4  stack.append(3)
5
6  while stack:
7      x = stack.pop()
8      print(x)
```

```
3
2
1
```

스택 - 구현

- https://www.acmicpc.net/problem/28278

스택 2 성공

정수를 저장하는 스택을 구현한 다음, 입력으로 주어지는 명령을 처리하는 프로그램을 작성하시오.

스택 - 구현

- 한번 풀어봅시다

스택 - 구현

- 같이 풀어봅시다

스택 - 구현

- 정답 코드

```python
import sys
input = sys.stdin.readline

stack = []
n = int(input())
for _ in range(n):
    cmd = list(map(int, input().split()))
    if len(cmd) == 2:
        stack.append(cmd[1])
    else:
        if cmd[0] == 2:
            print(stack.pop() if stack else -1)
        elif cmd[0] == 3:
            print(len(stack))
        elif cmd[0] == 4:
            print(int(not bool(stack)))
        elif cmd[0] == 5:
            print(stack[-1] if stack else -1)
```

스택 - 활용

- https://www.acmicpc.net/problem/9012

괄호 성공 다국어

괄호 문자열(Parenthesis String, PS)은 두 개의 괄호 기호인 '(' 와 ')' 만으로 구성되어 있는 문자열이다.

한 쌍의 괄호 기호로 된 "( )" 문자열은 기본 VPS 이라고 부른다.

만일 x 가 VPS 라면 이것을 하나의 괄호에 넣은 새로운 문자열 "(x)"도 VPS

두 VPS x 와 y를 접합(concatenation)시킨 새로운 문자열 xy도 VPS 가 된다.

스택 - 활용

- 괄호 문자열의 정의를 보면 **재귀적**으로 이루어져 있음을 알 수 있다.

- **열린 괄호**를 함수의 **호출**,
  닫는 괄호를 함수의 **종료**로 보면,
  VPS는 올바른 함수의 호출 관계를 나타낸다!
  → 즉, **콜 스택**을 나타낸다!

스택 - 활용

- **열린 괄호**는 데이터를 스택에 넣으라는 뜻 **(push)**
- **닫힌 괄호**는 데이터를 스택에서 빼라는 뜻 **(pop)**

스택 – 활용

- 한번 풀어봅시다

스택 – 활용

- 같이 풀어봅시다

스택 – 활용

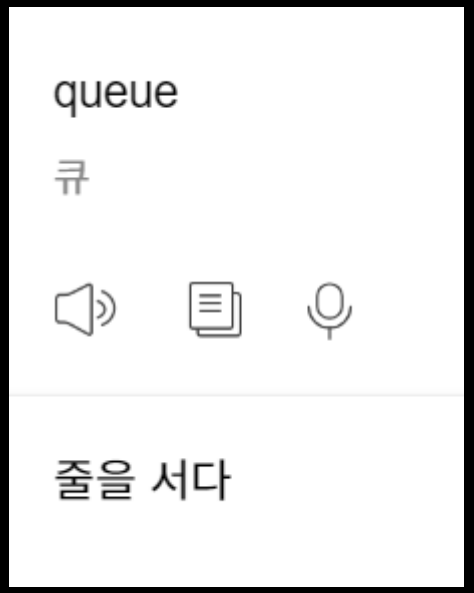
- 정답 코드

```python
import sys
input = sys.stdin.readline

T = int(input())
for _ in range(T):
    s = input().rstrip()
    stack = []
    found = False
    for c in s:
        if c == '(':
            stack.append(1)
        else:
            if stack:
                stack.pop()
            else:
                print("NO")
                found = True
                break
    if not found:
        if stack:
            print("NO")
        else:
            print("YES")
```

큐

- 데이터를 옆으로 나열한 **대기줄 형식**의 자료구조



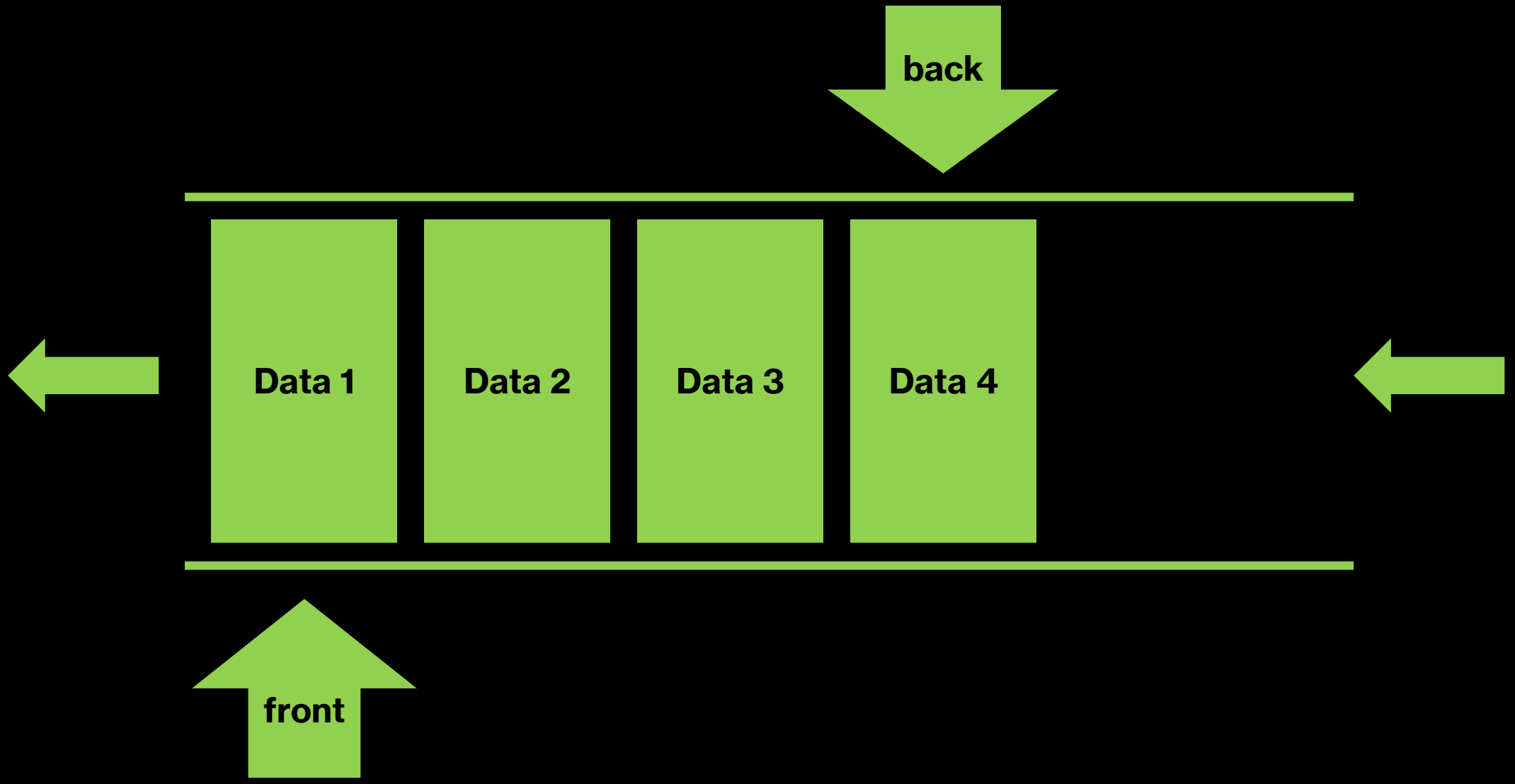
큐

- 데이터를 저장할 때 **앞에서부터 저장**한다.
  → push
  데이터를 뺄 때는 **제일 앞에 있는 데이터를 뺀다.**
  → pop

- 먼저 들어간 데이터가 먼저 나오는 형태
  → (First In First Out == **FIFO**)

큐 – push

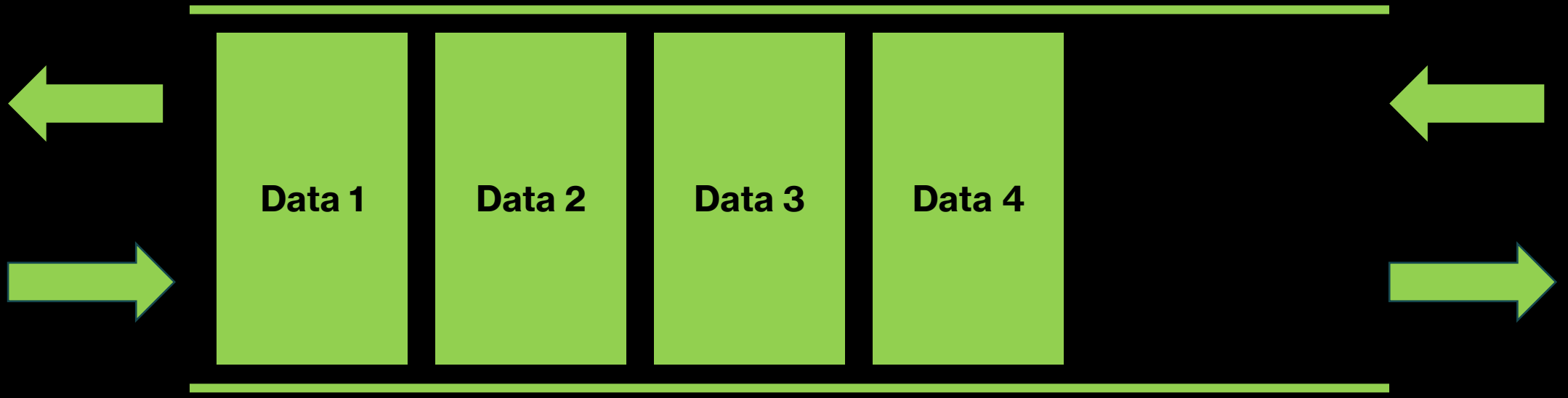
큐 – pop



덱 (Deque)

- **D**ouble **E**nded **Que**ue
- 말 그대로, 끝 점이 양쪽에 있는 큐
- 왼쪽에서도 뺄 수 있고, 오른쪽에서도 뺄 수 있다.

덱 (Deque)



덱 (Deque)

- Q. 스택도 덱으로 구현할 수 있을 것 같은데요..?

- A. 가능합니다! 덱으로 구현해도 괜찮아요!
  구현 방법은 자유입니다!

큐 & 덱 – 구현

- 리스트는 큐, 덱처럼 쓸 수 없다!
- 리스트의 **pop(0) 연산이 O(N)** 시간에 수행되기 때문
  → 별도 모듈이 필요하다!

큐 & 덱 – 구현

- 보통 큐와 덱을 구현할 때는 **deque(덱)**을 이용한다.

```
1  from collections import deque
```

큐 - 구현

- 데이터 추가 (push) : **append()**
- 데이터 삭제 (pop) : **popleft()**

```
1  from collections import deque
2
3  queue = deque() # 비어있는 queue 생성
4  queue.append(1)
5  queue.append(2)
6  queue.append(3)
7
8  while queue:
9      print(queue.popleft())
```

```
>>> %
1
2
3
```

덱 - 구현

- 데이터 추가 (push) : **append(), appendleft()**
- 데이터 삭제 (pop) : **popleft(), pop()**

```python
from collections import deque

d = deque([1, 2, 3]) # 1, 2, 3으로 초기화된 deque

print(d.popleft())  # [2, 3]
print(d.pop())      # [2]
d.appendleft(1)     # [1, 2]
d.append(4)         # [1, 2, 4]
print(d)
```

```
>>> %Run -c $EDIT

1
3
deque([1, 2, 4])
```

큐 & 덱 – 구현

- Q. 스택도 덱으로 구현할 수 있을 것 같은데요..?

- A. 가능합니다! 덱으로 구현해도 괜찮아요.
  구현 방법은 자유입니다!
  (저는 모듈 import 하는 것도 불편하고, 모듈을 사용하면 오버헤드가 생길 것 같아서 내장 리스트를 쓰려고 하는 편입니다.)

큐 – 구현

- https://www.acmicpc.net/problem/18258

큐 2 성공

정수를 저장하는 큐를 구현한 다음, 입력으로 주어지는 명령을 처리하는 프로그램을 작성하시오.

큐 – 구현

- 한번 풀어봅시다

큐 – 구현

- 같이 풀어봅시다

큐 – 구현

- 정답 코드

```python
from collections import deque
import sys
input = sys.stdin.readline

n = int(input())
q = deque()
for _ in range(n):
    cmd = input().split()
    if len(cmd) == 2:
        q.append(cmd[1])
    elif cmd[0] == 'pop':
        print(q.popleft() if q else -1)
    elif cmd[0] == 'size':
        print(len(q))
    elif cmd[0] == 'empty':
        print(1 if not q else 0)
    elif cmd[0] == 'front':
        print(-1 if not q else q[0])
    elif cmd[0] == 'back':
        print(-1 if not q else q[-1])
```

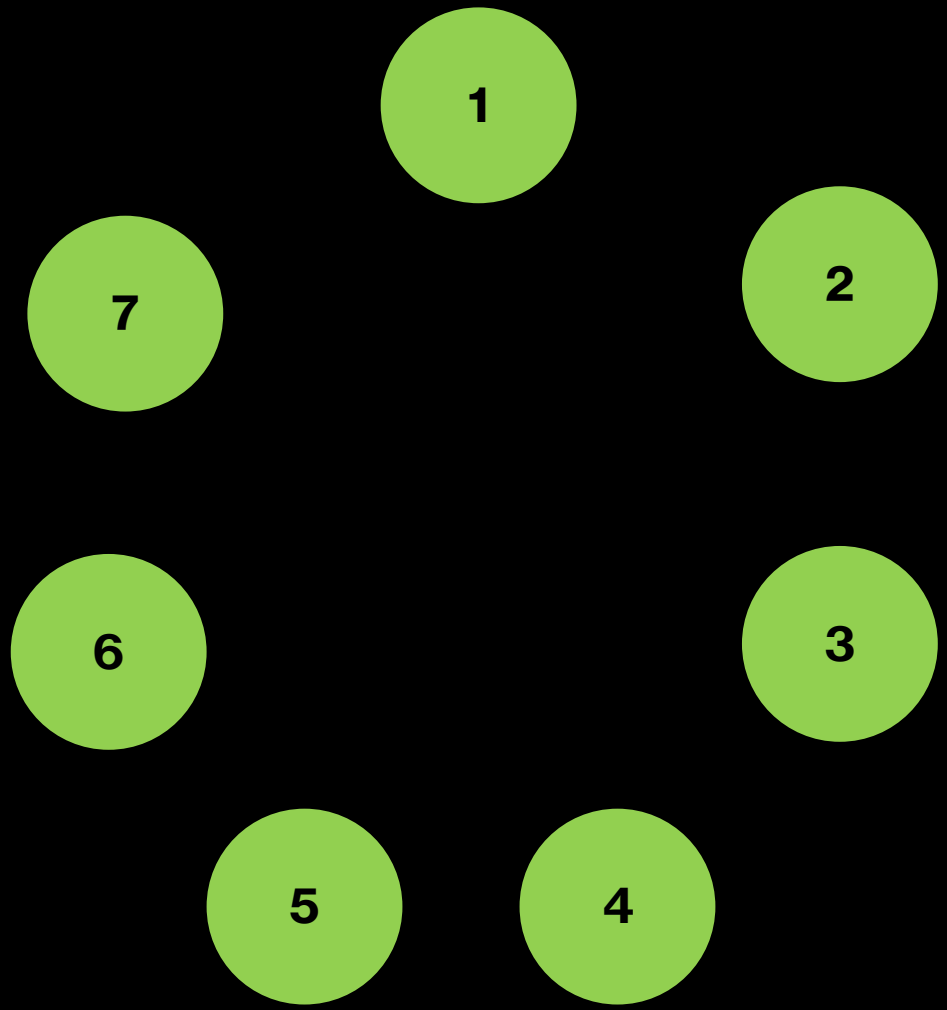
큐 & 덱 – 활용

- https://www.acmicpc.net/problem/11866

요세푸스 문제 0 성공

1번부터 N번까지 N명의 사람이 원을 이루면서 앉아있고, 양의 정수 K(≤ N)가 주어진다. 이제 순서대로 K번째 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, K)-요세푸스 순열이라고 한다. 예를 들어 (7, 3)-요세푸스 순열은 <3, 6, 2, 7, 5, 1, 4>이다.

큐 & 덱 – 활용



큐 & 덱 – 활용

- 같이 풀어봅시다

큐 & 덱 – 활용

- 정답 코드 (큐)

```python
from collections import deque
n, k = map(int, input().split())

answer = []
d = deque(i for i in range(1, n+1))
while d:
    for _ in range(k-1):
        d.append(d.popleft())
    answer.append(str(d.popleft()))

print(f"<{', '.join(answer)}>")
```

큐 & 덱 – 활용

- 정답 코드 (덱)

```python
from collections import deque
n, k = map(int, input().split())

answer = []
d = deque(i for i in range(1, n+1))
while d:
    d.rotate(-(k-1))
    answer.append(str(d.popleft()))

print(f"<{', '.join(answer)}>")
```

추가 연습 문제

- https://www.acmicpc.net/problem/28279

덱 2 성공

1. `1 X` : 정수 X를 덱의 앞에 넣는다. ($1 \le X \le 100,000$)
2. `2 X` : 정수 X를 덱의 뒤에 넣는다. ($1 \le X \le 100,000$)
3. `3` : 덱에 정수가 있다면 맨 앞의 정수를 빼고 출력한다. 없다면 `-1` 을 대신 출력한다.
4. `4` : 덱에 정수가 있다면 맨 뒤의 정수를 빼고 출력한다. 없다면 `-1` 을 대신 출력한다.
5. `5` : 덱에 들어있는 정수의 개수를 출력한다.
6. `6` : 덱이 비어있으면 `1` , 아니면 `0` 을 출력한다.
7. `7` : 덱에 정수가 있다면 맨 앞의 정수를 출력한다. 없다면 `-1` 을 대신 출력한다.
8. `8` : 덱에 정수가 있다면 맨 뒤의 정수를 출력한다. 없다면 `-1` 을 대신 출력한다.

추가 연습 문제

- 정답 코드

```python
from collections import deque
import sys

input = sys.stdin.readline
n = int(input())
d = deque()
for _ in range(n):
    cmd = input().split()
    if len(cmd) == 2:
        if cmd[0] == '1':
            d.appendleft(cmd[1])
        elif cmd[0] == '2':
            d.append(cmd[1])
    elif cmd[0] == '3':
        print(d.popleft() if d else -1)
    elif cmd[0] == '4':
        print(d.pop() if d else -1)
    elif cmd[0] == '5':
        print(len(d))
    elif cmd[0] == '6':
        print(1 if not d else 0)
    elif cmd[0] == '7':
        print(d[0] if d else -1)
    elif cmd[0] == '8':
        print(d[-1] if d else -1)
```

이번 주 연습 문제

- 10773 제로
- 4949 균형잡힌 세상
- 12789 도키도키 간식드리미
- 2164 카드2
- 2346 풍선 터뜨리기
- 24511 queuestack