〈자료구조 실습〉 - 큐

※ 입출력에 대한 안내

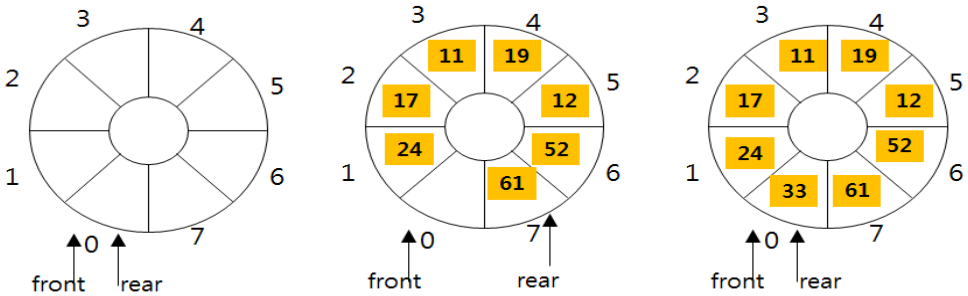
- 특별한 언급이 없으면 문제의 조건에 맞지 않는 입력은 입력되지 않는다고 가정하라.
- 특별한 언급이 없으면, 각 줄의 맨 앞과 맨 뒤에는 공백을 출력하지 않는다.
- 출력 예시에서 □는 각 줄의 맨 앞과 맨 뒤에 출력되는 공백을 의미한다.
- 입출력 예시에서 ↦ 이 후는 각 입력과 출력에 대한 설명이다.

[문제 1-큐] 배열로 구성된 **원형 큐**를 위한 삽입, 삭제 프로그램을 작성하시오.

○ **주요 전략** : 본 문제의 원형 큐에서는 **포화 상태와 공백 상태를 구분**하기 위해 한 자리를 비워둠.

- front, rear, 배열의 초기 값은 **0**
- 삽입 시 rear 값을 하나 증가시킨 후 데이터를 큐에 삽입 (출력 예시 1 참고)
- 삭제 시 front 값을 하나 증가시킨 후 front가 가리키는 데이터를 삭제
- front = rear면 공백 상태로 정의하고, front가 rear보다 하나 앞에 있으면 포화 상태로 정의함

※ **주의** : 주교재가 제시하는 전략에서는 front와 rear가 각각 큐의 맨 앞과 맨 뒤의 원소 위치를 직접 가리키는 방식으로 정의되어 있으나 위 전략은 front가 맨 앞 원소 위치보다 한 셀 앞 위치를 가리키는 방식으로 정의되었다. 주교재의 방식으로 변경하여 작성해도 무방하지만, **초기 상태에서 맨 처음 삽입되는 위치는 0번이 아니고, 1번이 되어야 함** (그렇지 않으면 본 문제의 입출력 예시와 다른 결과가 나올 수 있음).



공백상태 포화상태 오류상태

○ **입출력 형식:**

1) 첫 번째 라인 : 양의 정수 **q** (원형 큐의 크기)

※ **q** 값에는 제한이 없다. 또한, **동적 할당**을 사용할 것.

2) 두 번째 라인 : 양의 정수 **n** (연산의 개수)

3) 세 번째 이후 라인: **n**개의 연산이 차례로 입력됨.

※ 연산의 종류는 **I** (삽입), **D** (삭제), **P** (출력)

- **I 10** : 원형 큐에 원소 **10**을 삽입 (큐 원소는 양의 정수).
- **D** : 원형 큐에서 원소를 삭제한 후 해당 배열 원소 값을 **0**으로 치환.
- **P** : **배열 원소 전체**를 차례로 화면에 출력 (입출력 예시 참조).

※ **overflow 발생 시** (즉, 포화 상태에서 삽입을 시도한 경우),
화면에 overflow와 배열 값들을 모두 출력하고 프로그램 종료.

※ **underflow 발생 시** (즉, 공백 상태에서 삭제를 시도한 경우),
화면에 underflow를 출력하고 프로그램 종료.

입력 예시 1

```
6          ↦ q = 6
10         ↦ n = 10
I 10       ↦ 삽입
I 20       ↦ 삽입
P          ↦ 화면출력
I 30       ↦ 삽입
I 40       ↦ 삽입
D          ↦ 삭제
P          ↦ 화면출력
I 50       ↦ 삽입
I 60       ↦ 삽입
I 70       ↦ 삽입
```

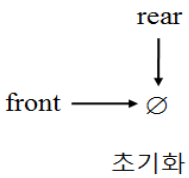
출력 예시 1

```
□0 10 20 0 0 0        ↦ 3번째 연산(P)에 의한 출력
□0 0 20 30 40 0       ↦ 7번째 연산(P)에 의한 출력
overflow□60 0 20 30 40 50
                  ↦ 10번째 연산(I 70)에서 overflow 발생
```

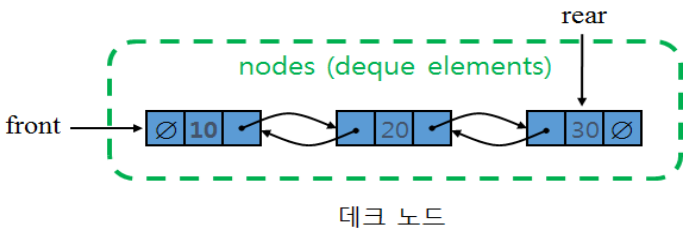
[문제 2-데크] 데크는 큐의 전단(front)과 후단(rear)에서 모두 삽입과 삭제가 가능한 자료구조다. **헤드 노드와 테일 노드가 없는 이중연결리스트**를 사용하여 아래에 정의된 데크 함수들을 구현하시오.

○ **초기 상태**

- **주의** : 연산 수행 도중 원소가 모두 삭제되어 데크가 비는 경우에도, 아래 초기 상태가 되어야 함.



초기화



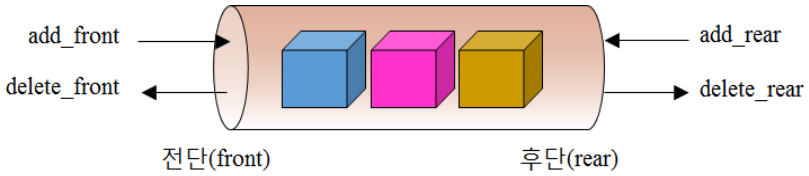
데크 노드

○ **데크 연산**

- **add_front**(deque, X) : deque의 앞에 원소 X를 추가 (주교재의 **push**와 동일).
- **add_rear**(deque, X) : deque의 뒤에 원소 X를 추가 (주교재의 **inject**와 동일).
- **delete_front**(deque) : deque의 앞에 있는 원소를 반환한 다음 삭제 (주교재의 **pop**과

동일).

- **delete_rear**(deque) : deque의 뒤에 있는 원소를 반환한 다음 삭제 (주교재의 **eject**와 동일).
- **print**(deque) : deque의 모든 원소들을 전단부터 후단까지 차례로 출력.



○ **입출력 형식:**

1) 첫 번째 라인 : 연산의 개수 **n**

2) 두 번째 이후 라인: **n**개의 연산이 한 줄에 하나씩 차례로 입력됨.

- 하나의 줄에는 연산의 종류, 추가인 경우 원소가 주어짐 (원소는 양의 정수로 표기).
- 연산의 종류: 다음의 연산 이름이 대문자로 주어짐.

AF (add_front), **AR** (add_rear), **DF** (delete_front), **DR** (delete_rear), **P** (print)

※ underflow 발생 시, 화면에 underflow를 출력하고 프로그램 종료.

입력 예시 1

7 ↦ 연산의 개수 AF 10 ↦ add_front(deque, 10) AF 20 ↦ add_front(deque, 20) AR 30 ↦ add_rear(deque, 30) P ↦ print(deque) DF ↦ delete_front(deque) DR ↦ delete_rear(deque) P ↦ print(deque)

출력 예시 1

□20 10 30 ↦ 4번째 연산(P)에 의한 출력 □10 ↦ 7번째 연산(P)에 의한 출력

입력 예시 2

15 ↦ 연산의 개수 AF 10 ↦ add_front(deque, 10) AF 20 ↦ add_front(deque, 20) AF 30 ↦ add_front(deque, 30) AR 40 ↦ add_rear(deque, 40) AR 50 ↦ add_rear(deque, 50) P ↦ print(deque) DF ↦ delete_front(deque) DF ↦ delete_front(deque) DR ↦ delete_rear(deque) P ↦ print(deque) DF ↦ delete_front(deque) DR ↦ delete_rear(deque) DR ↦ delete_rear(deque)

출력 예시 2

□30 20 10 40 50 ↦ 6번째 연산(P)에 의한 출력 □10 40 ↦ 10번째 연산(P)에 의한 출력 underflow ↦ 13번째 연산(DR)에서 underflow발생. 실행을 종료함