SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

삼성 청년 SW 아카데미

APS 기본

SAMSUNG SW ACADEMY FOR YOUTH

APS(Algorithm Problem Solving)

스택 (Stack) & 큐 (Queue)

- 원형큐
- 우선순위 큐
- 연습문제

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

원형 큐
(Circular Queue)

원형 큐의 구조

초기 공백 상태

- front = rear = 0

Index의 순환

- front와 rear의 위치가 배열의 마지막 인덱스인 n-1를 가리킨 후, 그 다음에는 논리적 순환을 이루어 배열의 처음 인덱스인 0으로 이동해야 함
- 이를 위해 나머지 연산자 mod를 사용함

원형 큐의 구조

front 변수

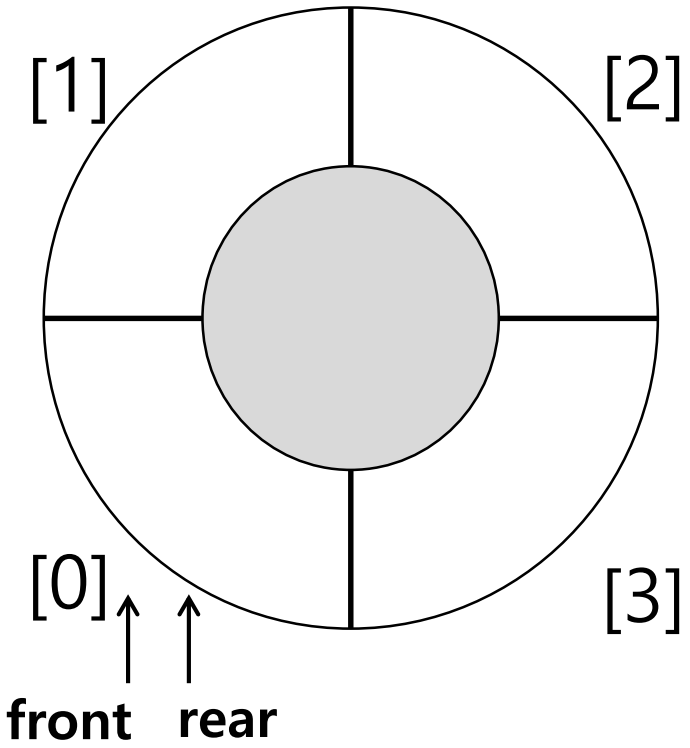
- 공백 상태와 포화 상태 구분을 쉽게 하기 위해 front가 있는 자리는 사용하지 않고 항상 빈자리로 둠

삽입 위치 및 삭제 위치

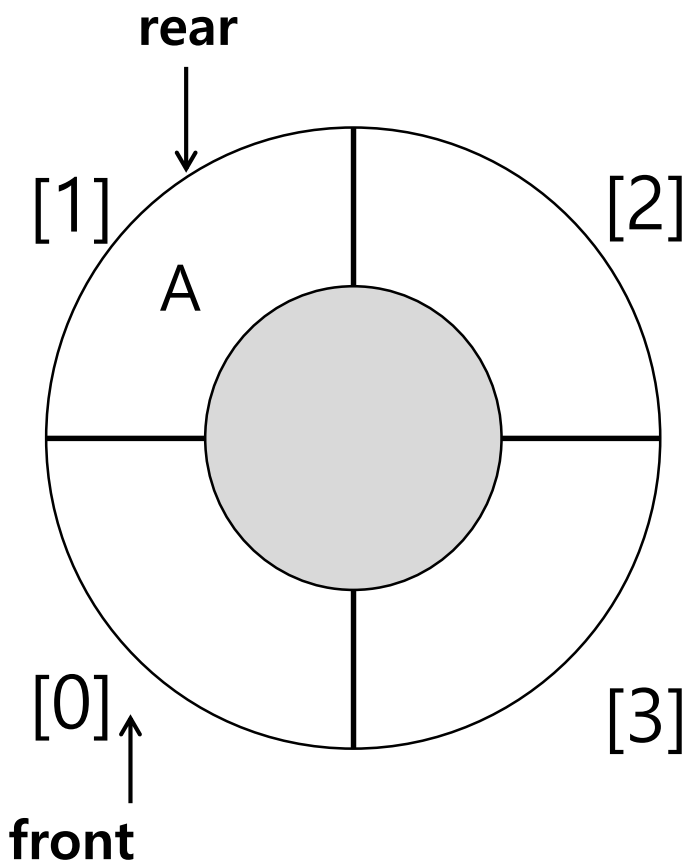
	삽입 위치	삭제 위치
선형큐	rear = rear + 1	front = front + 1
원형큐	rear = (rear + 1) mod n	front = (front + 1) mod n

원형 큐의 연산 과정

1) create Queue

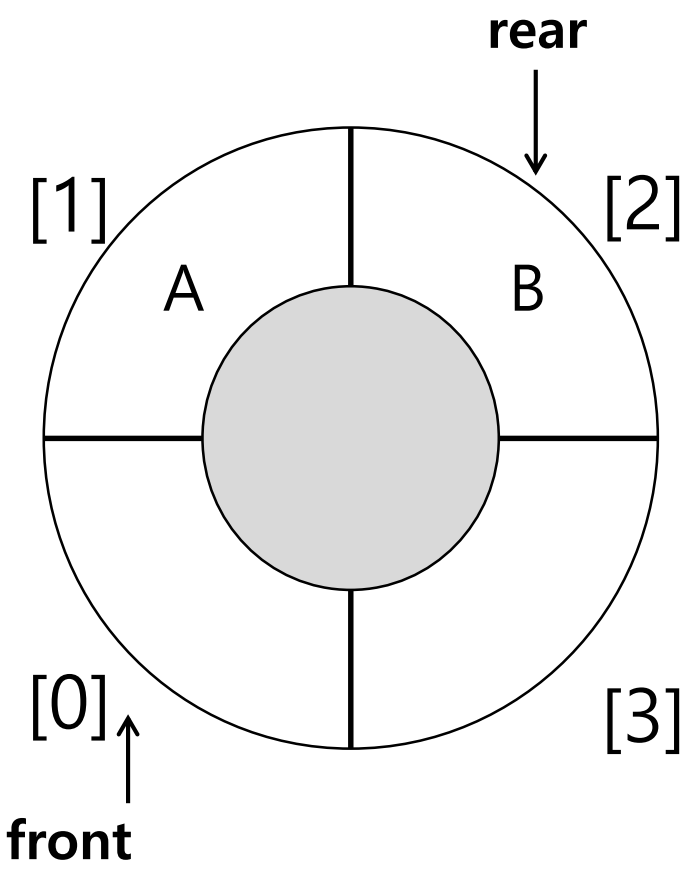


2) enQueue(A);

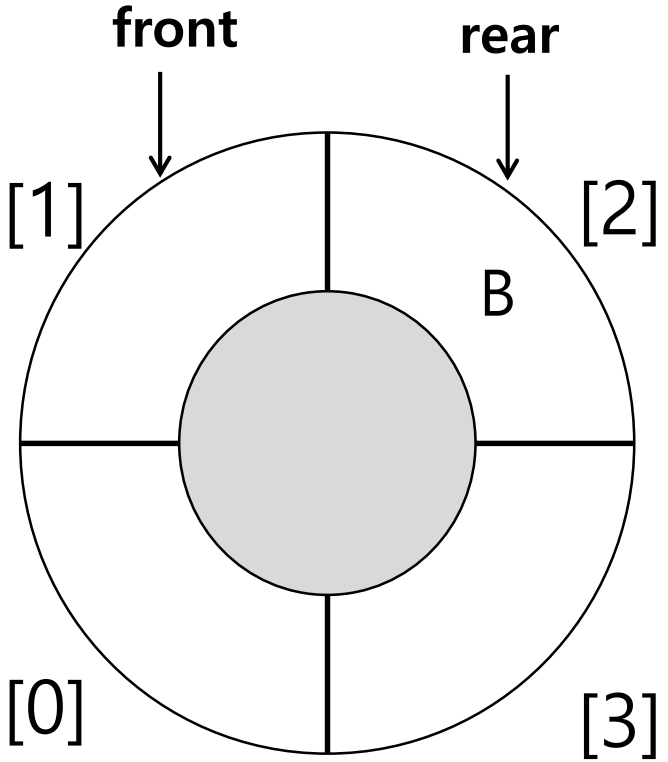


원형 큐의 연산 과정

3) enQueue(B);

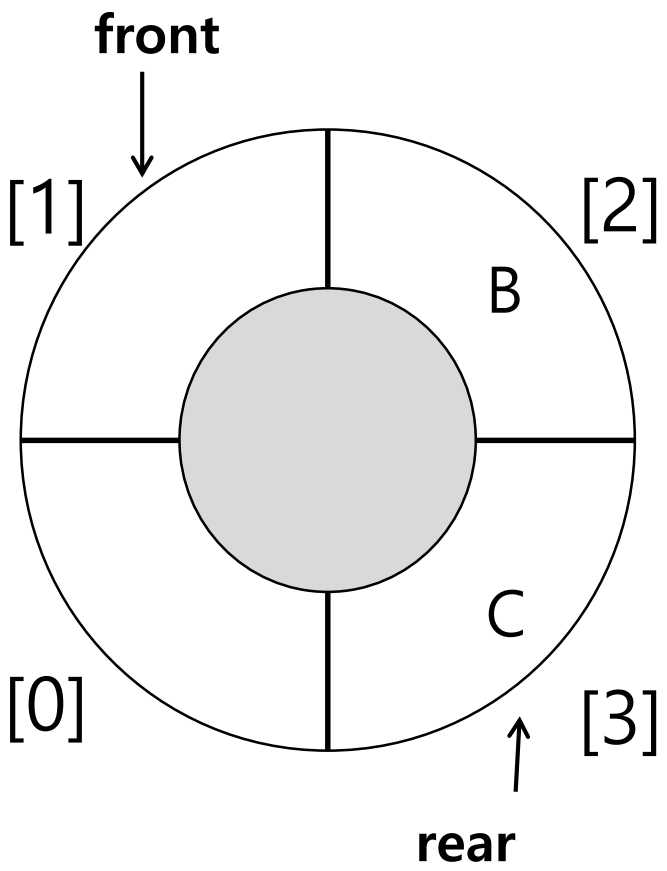


4) deQueue();

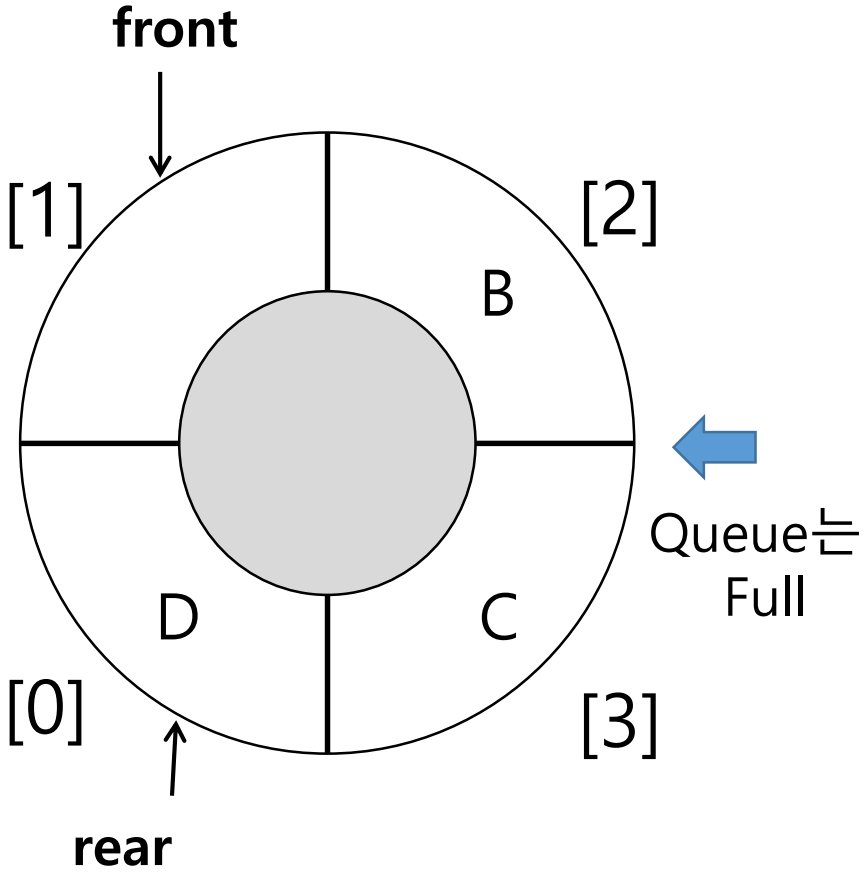


원형 큐의 연산 과정

5) enQueue(C);



6) enQueue(D);



원형 큐의 구현

초기 공백 큐 생성

- 크기 n인 1차원 배열 생성
- front와 rear를 0으로 초기화

공백상태 및 포화상태 검사 : isEmpty(), isFull()

- 공백상태 : front = rear
- 포화상태 : 삽입할 rear의 다음 위치 = 현재 front
  - (rear + 1) mod n = front

```
isEmpty() {
    if(front == rear) return true;
    else return false;
}
isFull() {
    if ((rear+1) mod n == front) return true;
    else return false;
}
```

원형 큐의 구현

삽입 : enQueue(item)

- 마지막 원소 뒤에 새로운 원소를 삽입하기 위해

  1) rear 값을 조정하여 새로운 원소를 삽입할 자리를 마련함 : rear ← (rear + 1) mod n;

  2) 그 인덱스에 해당하는 배열원소 cQ[rear]에 item을 저장

```
enQueue(item) {
    if (isFull()) print("Queue_Full")
    else {
        rear ← (rear + 1) mod n;
        cQ[rear] ← item;
    }
}
```

삭제 : deQueue(), delete()

- 가장 앞에 있는 원소를 삭제하기 위해

  1) front 값을 조정하여 삭제할 자리를 준비함

  2) 새로운 front 원소를 리턴 함으로써 삭제와 동일한 기능함

```
deQueue() {
    if (isEmpty()) print("Queue_Empty");
    else {
        front ← (front + 1) mod n;
        return cQ[front];
    }
}
```

원형 큐의 구현

삭제 : deQueue(), delete()

- 가장 앞에 있는 원소를 삭제하기 위해

  1) front 값을 조정하여 삭제할 자리를 준비함

  2) 새로운 front 원소를 리턴 함으로써 삭제와 동일한 기능함

```
delete() {
    if (isEmpty()) print("Queue_Empty");
    else front ← (front + 1) mod n;
}
```

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

우선순위 큐
(Priority Queue)

우선순위 큐(Priority Queue)

우선순위 큐의 특성

- 우선순위를 가진 항목들을 저장하는 큐
- FIFO 순서가 아니라 우선순위가 높은 순서대로 먼저 나가게 된다.

우선순위 큐의 적용 분야

- 시뮬레이션 시스템
- 네트워크 트래픽 제어
- 운영체제의 테스크 스케줄링

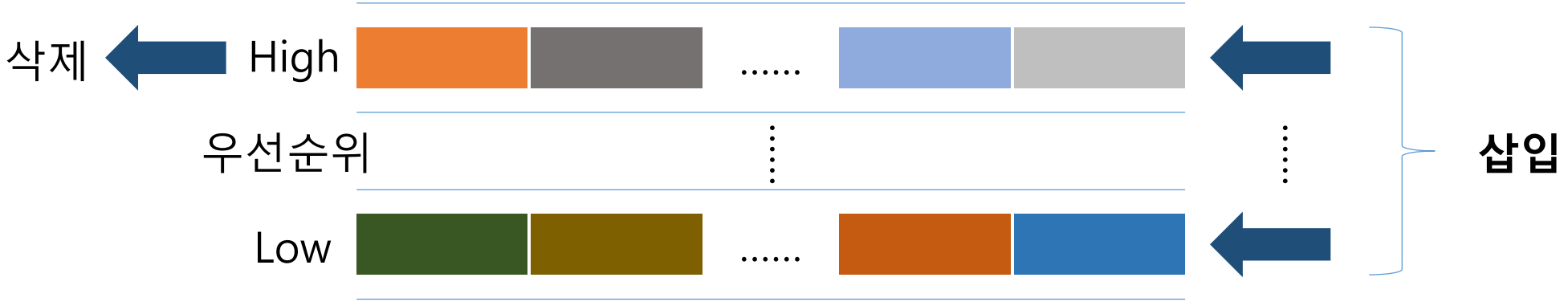
우선순위 큐(Priority Queue)

우선순위 큐의 구현

- 배열을 이용한 우선순위 큐
- 리스트를 이용한 우선순위 큐

우선순위 큐의 기본 연산

- 삽입 : enQueue
- 삭제 : deQueue



배열을 이용한 우선순위 큐

배열을 이용하여 우선순위 큐 구현

- 배열을 이용하여 자료 저장
- 원소를 삽입하는 과정에서 우선순위를 비교하여 적절한 위치에 삽입하는 구조
- 가장 앞에 최고 우선순위의 원소가 위치하게 됨

문제점

- 배열을 사용하므로, 삽입이나 삭제 연산이 일어날 때 원소의 재배치가 발생함
- 이에 소요되는 시간이나 메모리 낭비가 큼

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

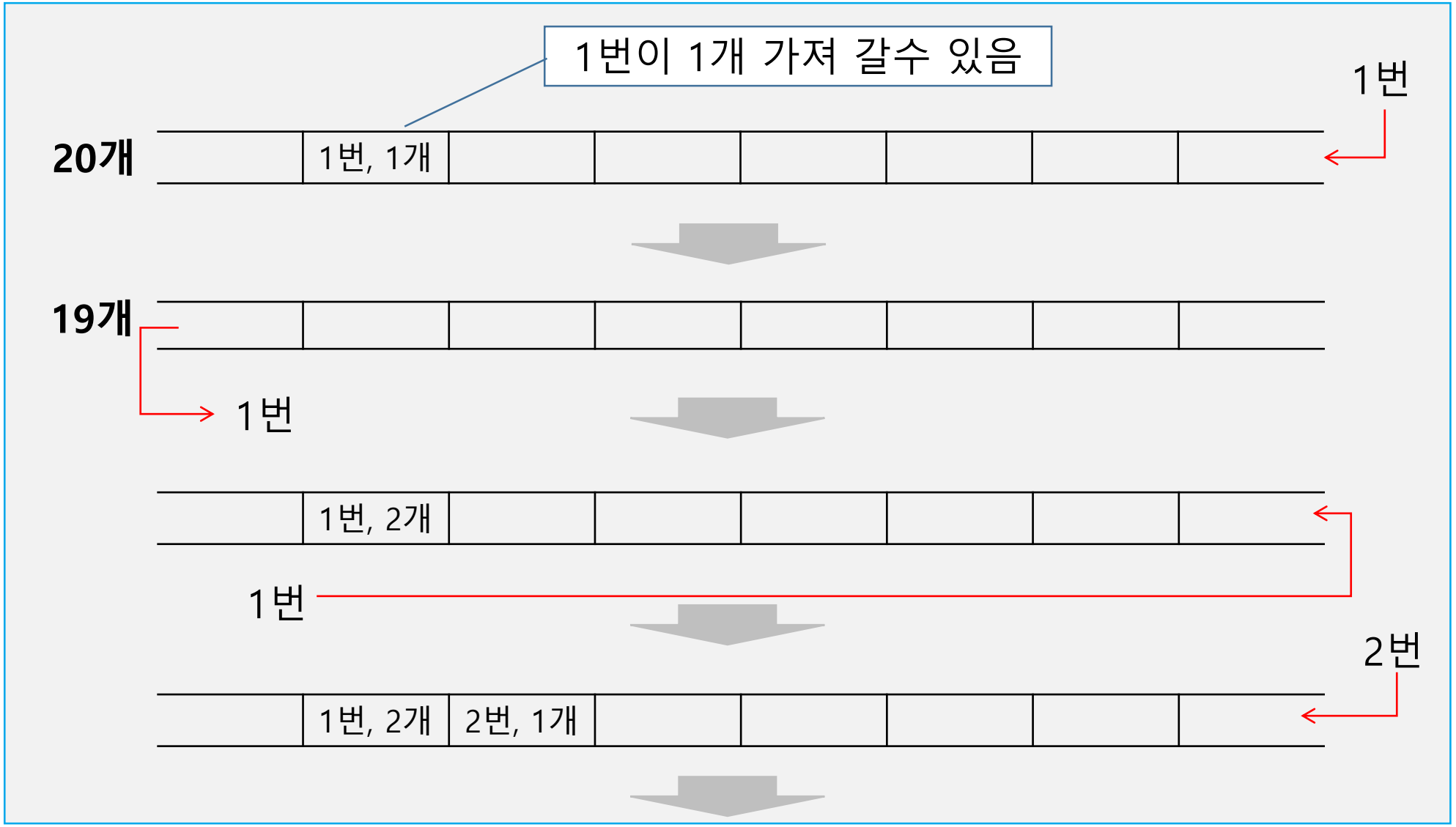
연습문제

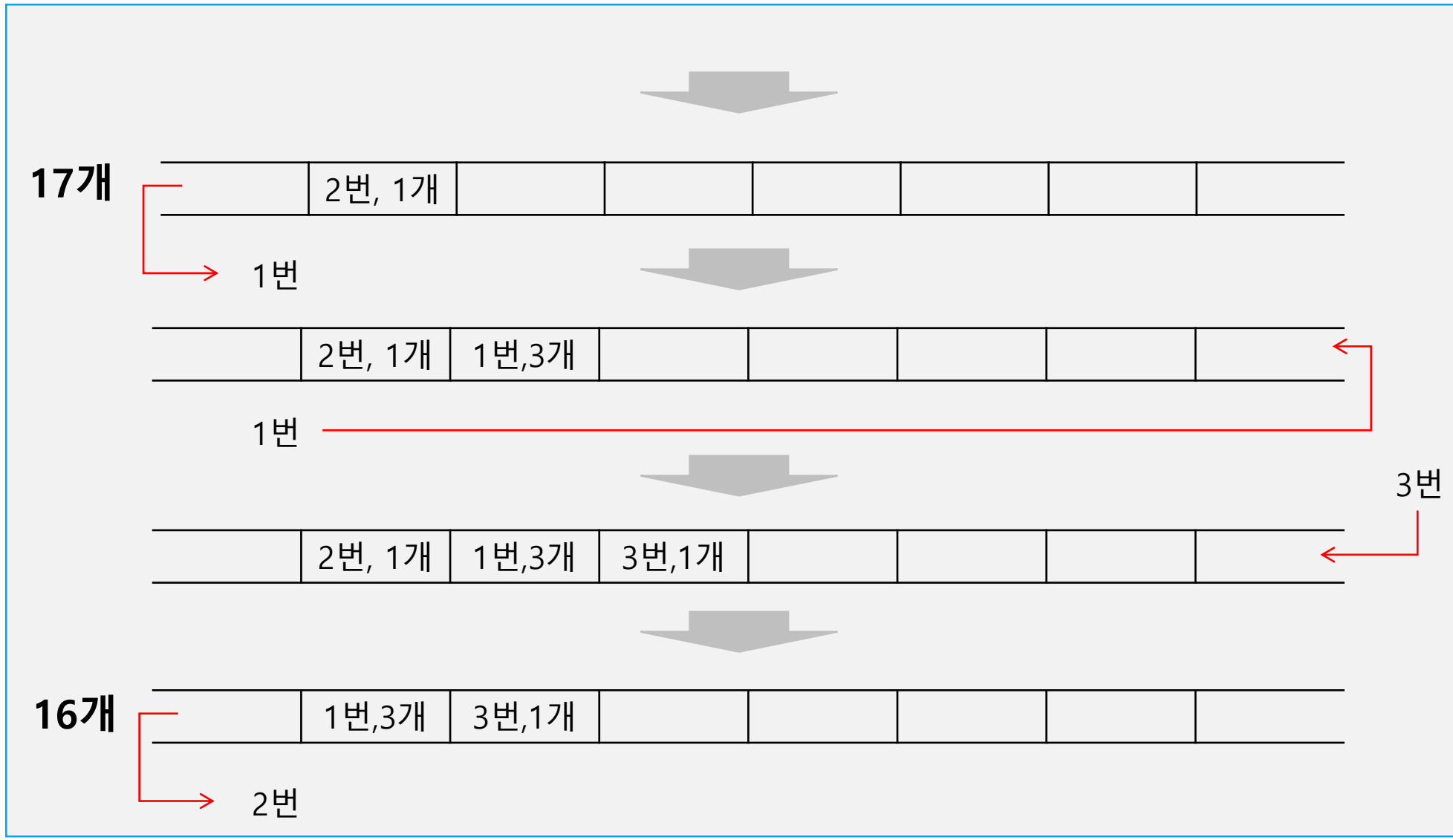
Revisit to 마이쮸

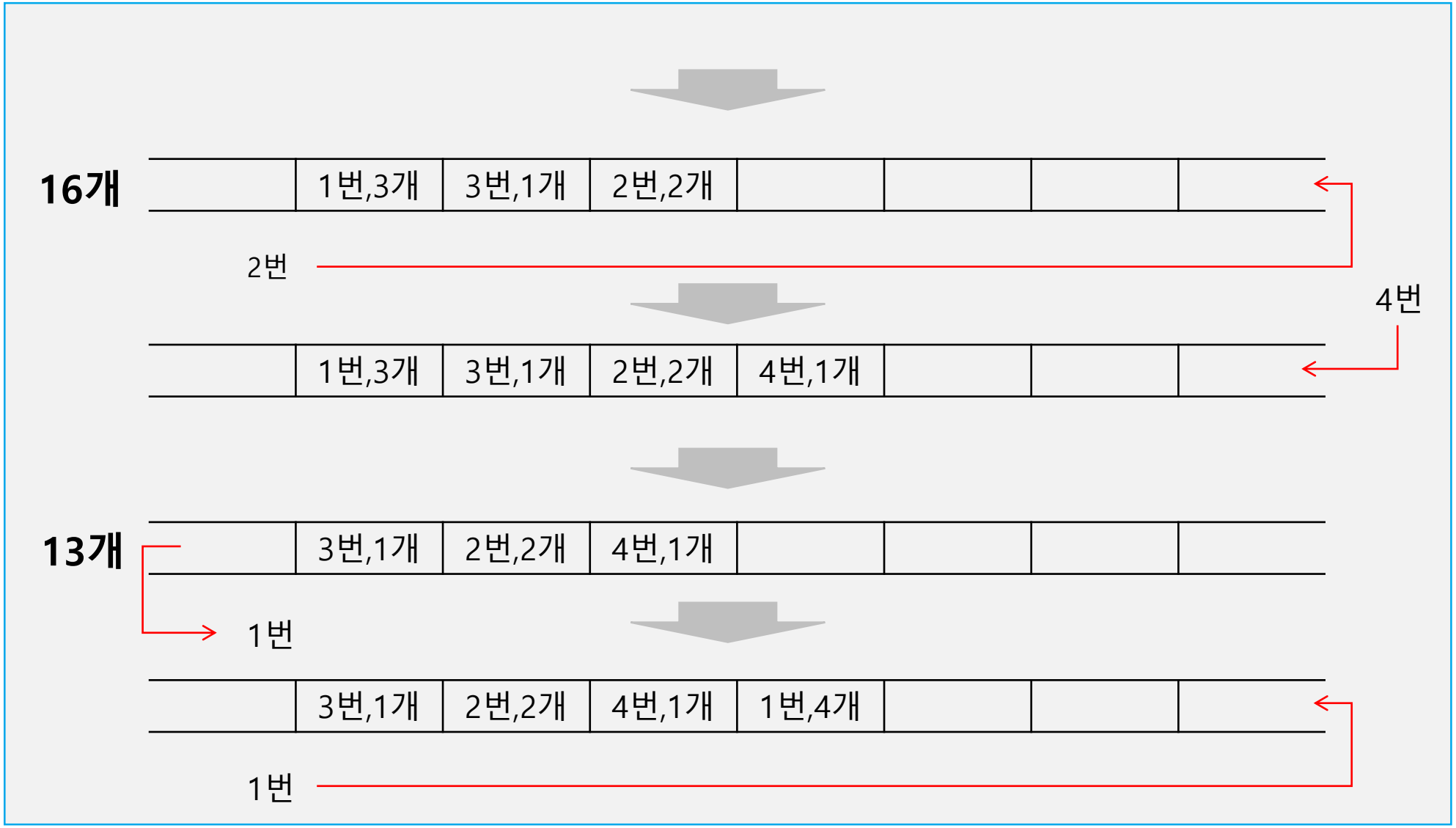
Queue 를 이용하여 마이쮸 나눠주기 시뮬레이션을 해 보자

- 1번이 줄을 선다.
- 1번이 한 개의 마이쮸를 받는다.
- 1번이 다시 줄을 선다.
- 새로 2번이 들어와 줄을 선다.
- 1번이 두 개의 마이쮸를 받는다.
- 1번 다시 줄을 선다.
- 새로 3번이 들어와 줄을 선다.
- 2번이 한 개의 마이쮸를 받는다.
- 2번이 다시 줄을 선다.
- 새로 4번이 들어와 줄을 선다.
- 1번이 세 개의 마이쮸를 받는다.
- 1번이 다시 줄을 선다
- 새로 5번이 들어와 줄을 선다.
- 3번이 한 개의 마이쮸를 받는다.
- ...
- 20개의 마이쮸가 있을 때 마지막 것을 누가 가져갈까?

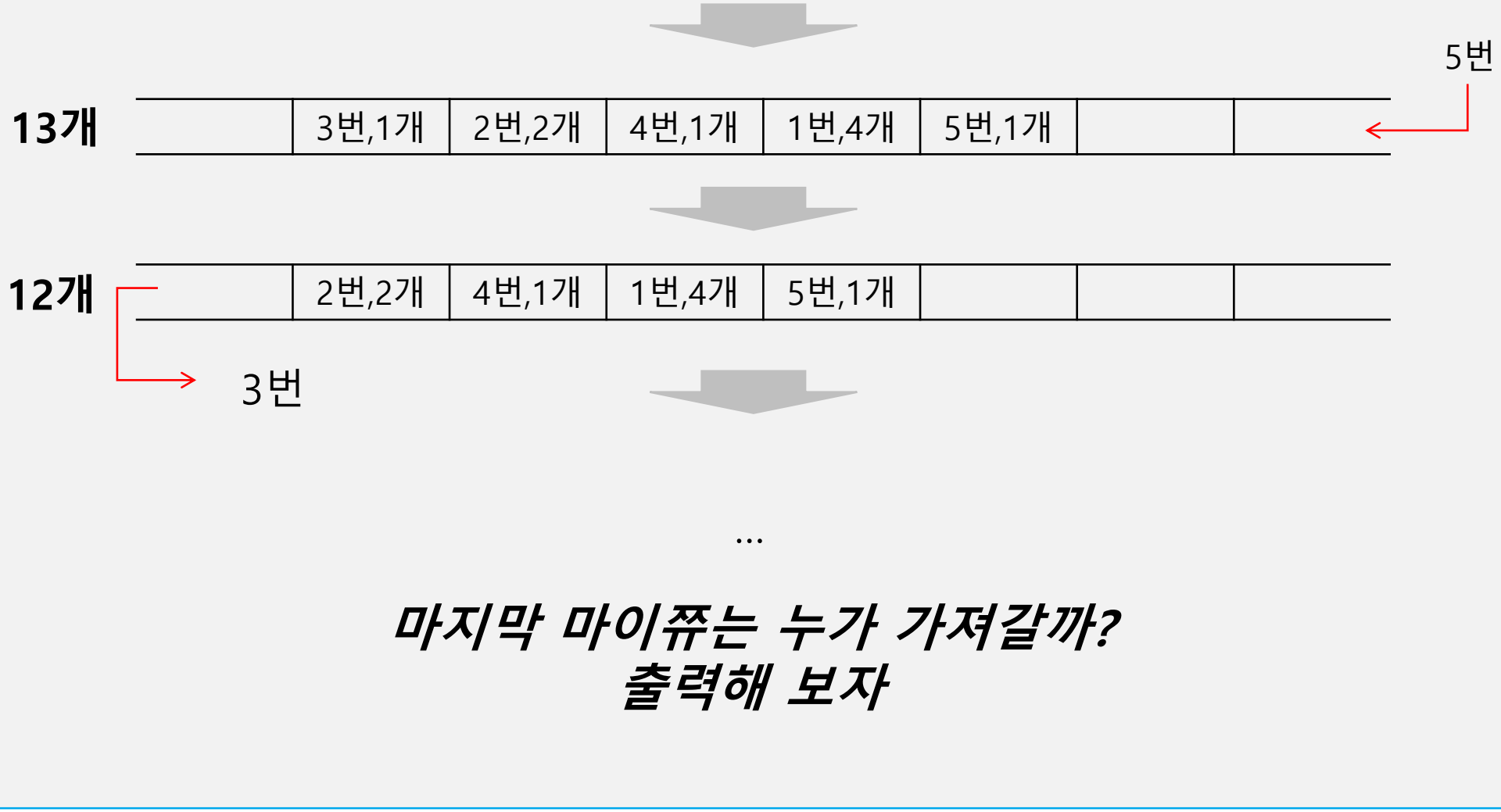
마이쮸




17개
2번, 1개
1번
2번, 1개
1번,3개
1번
3번
2번, 1개
1번,3개
3번,1개
16개
1번,3개
3번,1개
2번



마이쮸



마지막 마이쮸는 누가 가져갈까?
출력해 보자

<연습문제2>

- **마이쮸 시뮬레이션 구현**

- **엔터를 칠 때마다 다음 정보를 화면에 출력해 보자.**
  - 큐에 있는 사람 수
  - 현재 일인당 나눠주는 사탕의 수
  - 현재까지 나눠준 사탕의 수

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

다음 방송에서 만나요!

삼성 청년 SW 아카데미