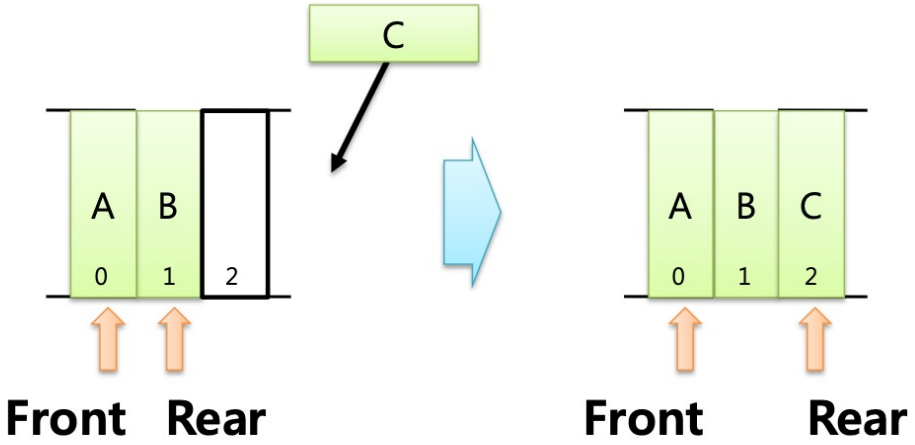
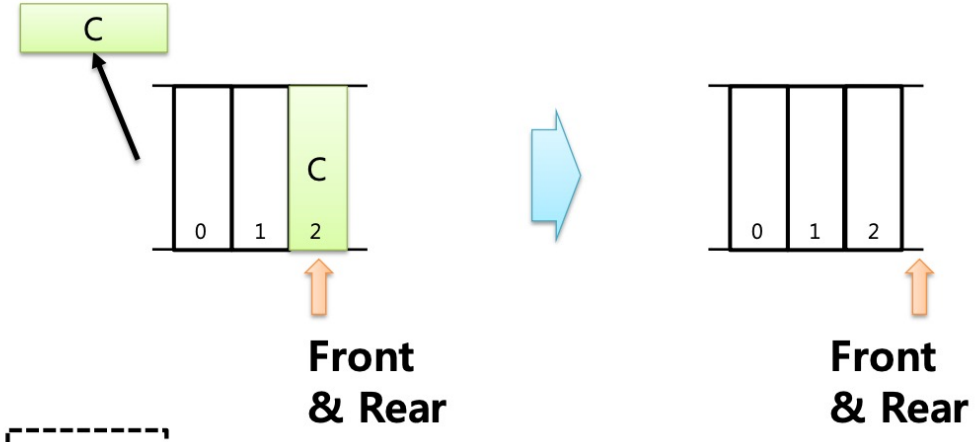
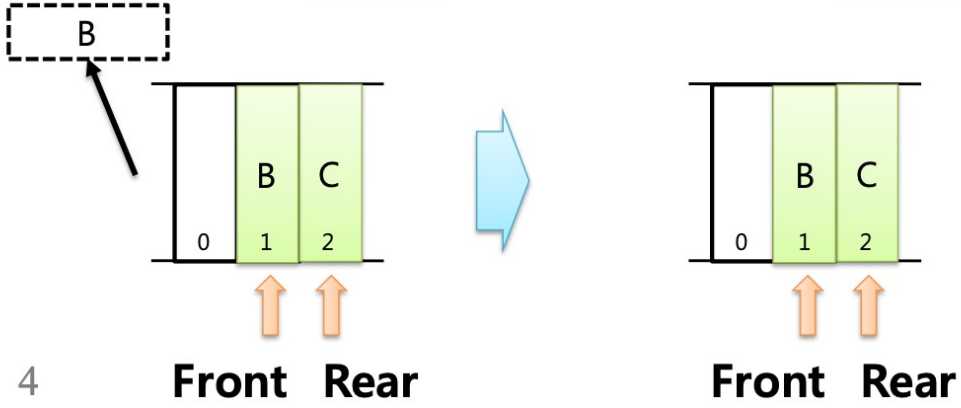
- 3가지 연산
  - 인큐(Enqueue, En-Queue)
    - 넘침(Overflow) 현상



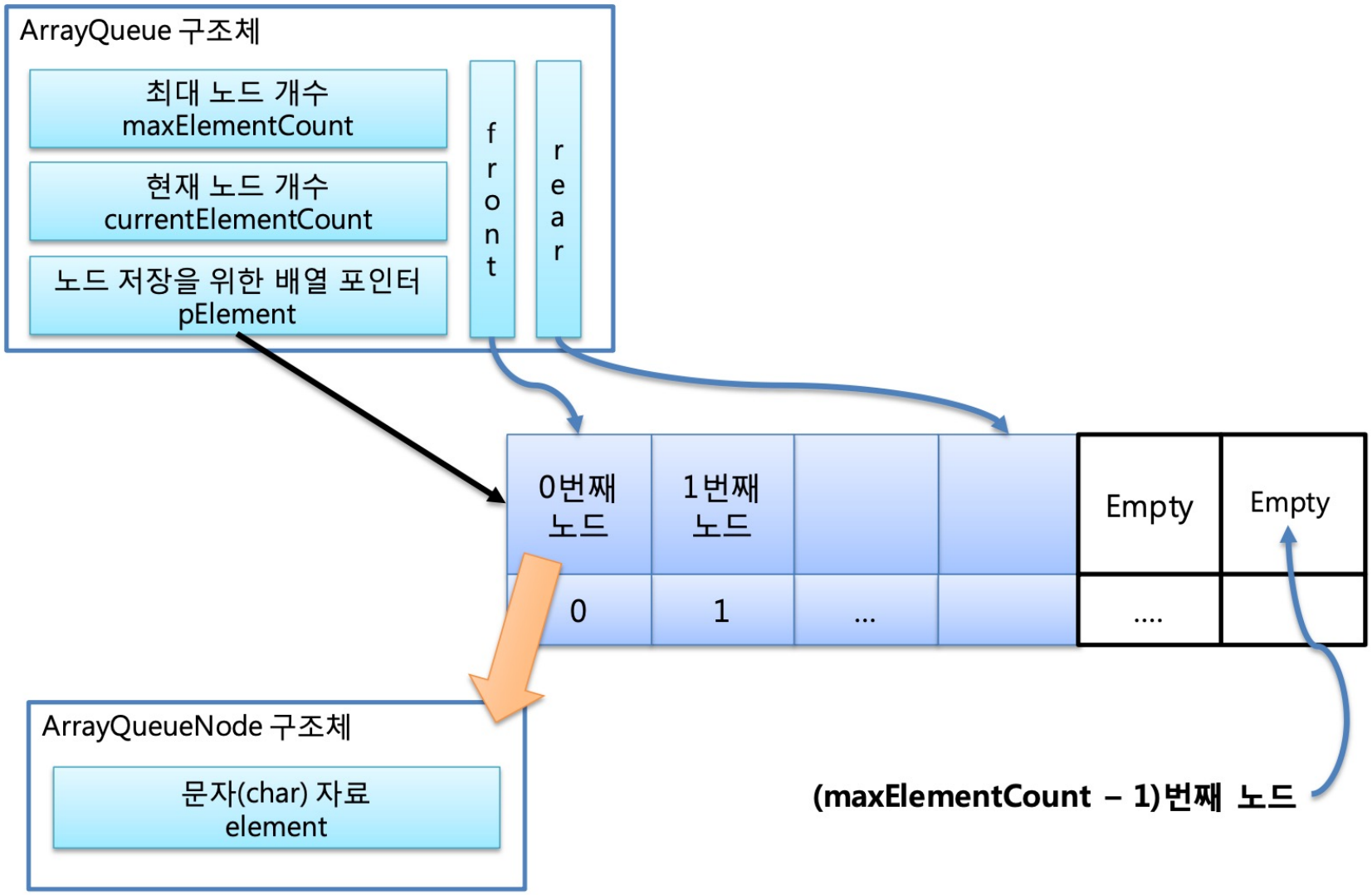
  - 디큐(Dequeue, De-Queue)
    - 부족(Underflow) 현상



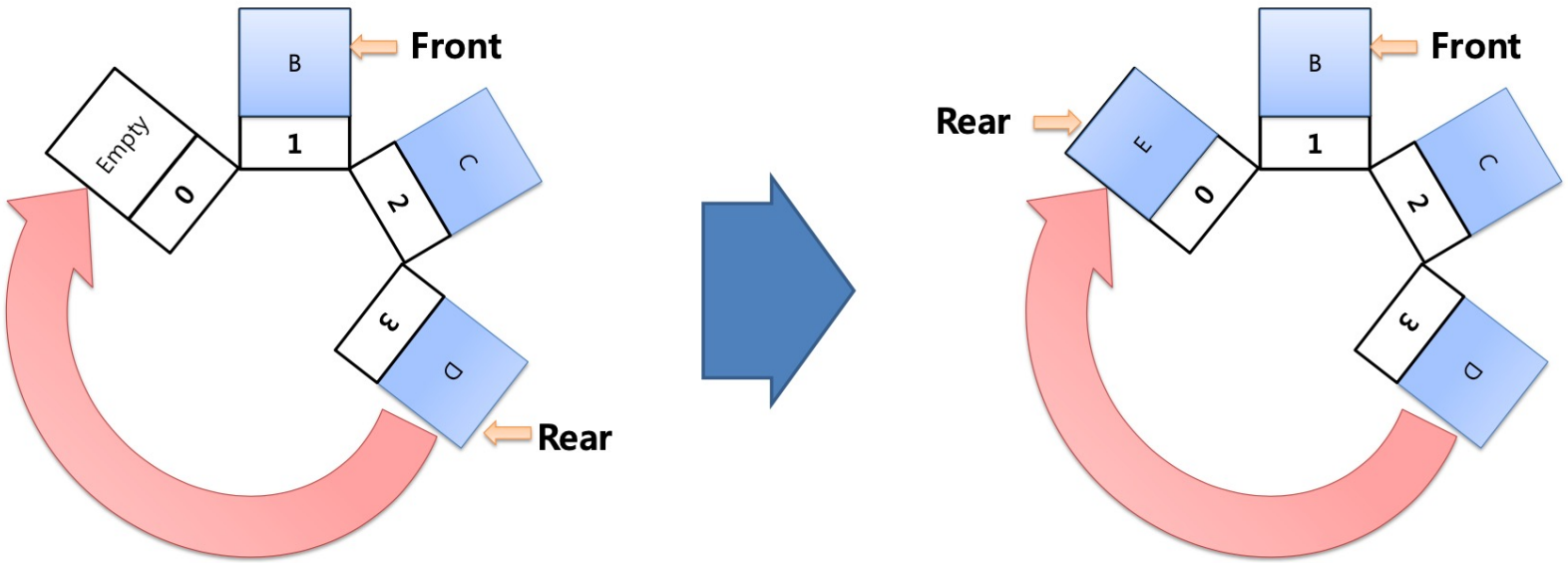
  - 피크(Peek)



배열로 구현하기

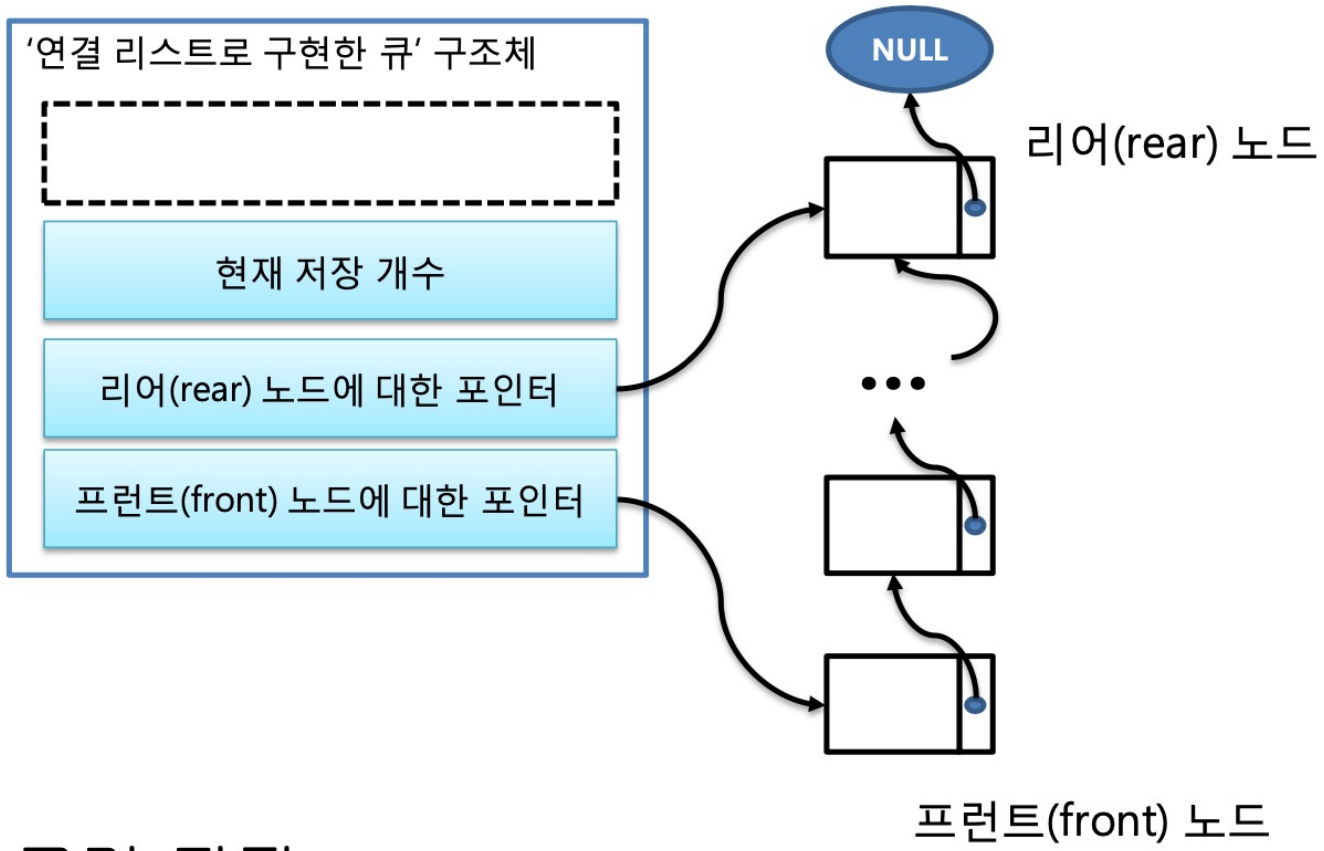

ArrayQueue 구조체
최대 노드 개수
maxElementCount
현재 노드 개수
currentElementCount
노드 저장을 위한 배열 포인터
pElement
front
rear
0번째 노드
1번째 노드
Empty
Empty
0
1
...
....
ArrayQueueNode 구조체
문자(char) 자료
element
(maxElementCount – 1)번째 노드

원형 큐



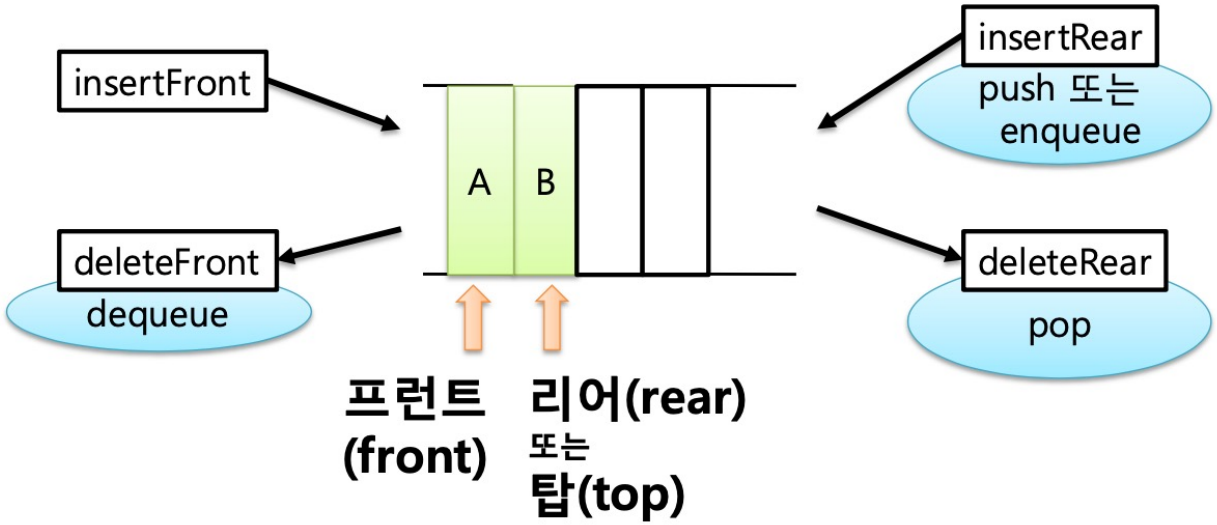
rear = (rear + 1) % maxElementCount

연결 리스트로 큐 구현하기



- 차이점
  - 큐의 크기 지정

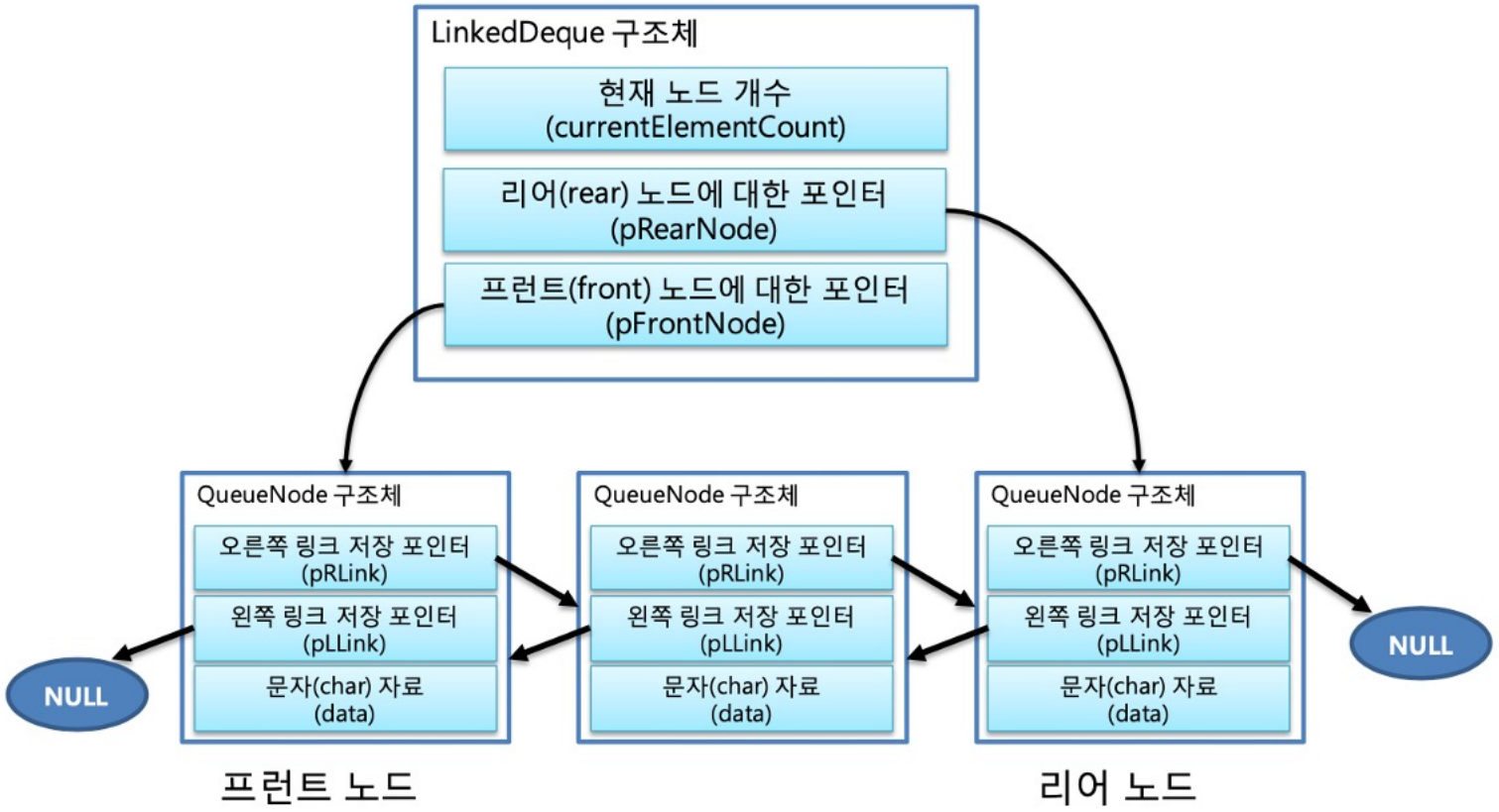
연결 리스트로 덱(deque) 구현하기



연산		덱	스택	큐
앞, 프런트(front)	추가	insertFront	없음	없음
	반환	deleteFront	없음	Dequeue
뒤, 리어(rear)	추가	insertRear	Push	Enqueue
	반환	deleteRear	Pop	없음

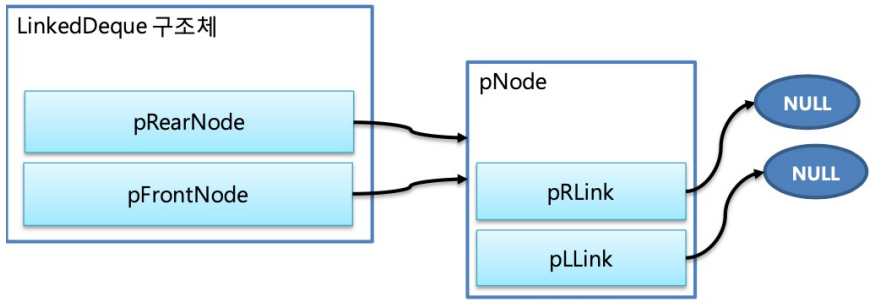
연결 리스트로 덱(deque) 구현하기

- 구조체



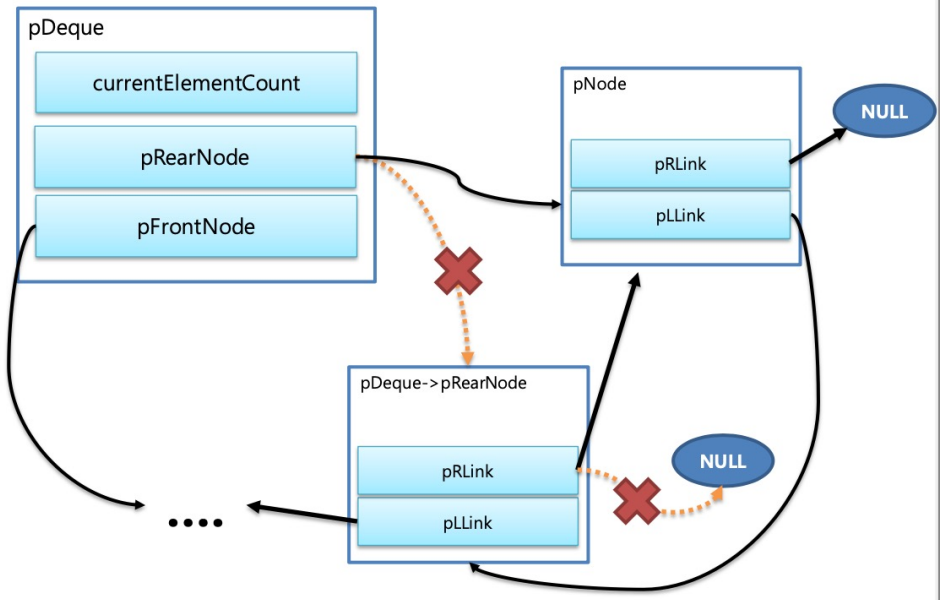
연결 리스트로 덱(deque) 구현하기

- 뒤 추가
  - 공백 상태일 때




```
pDeque->pFrontNode = pNode;
pDeque->pRearNode = pNode;
```

공백 상태가 아닐 때



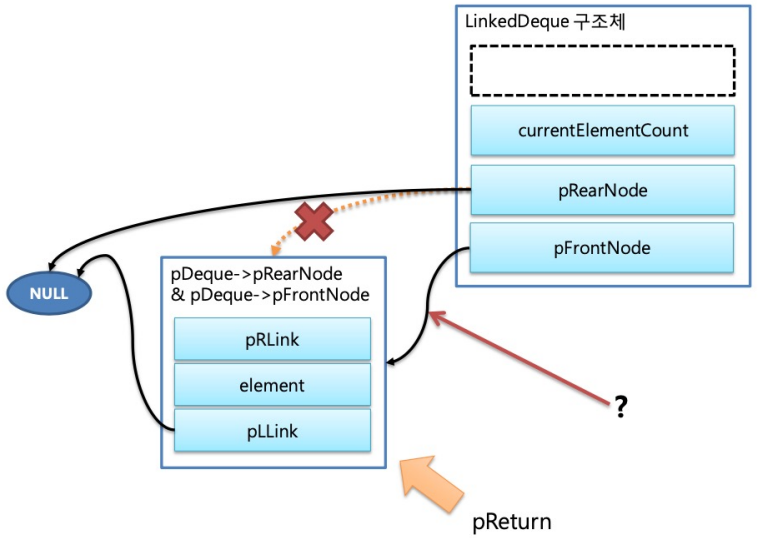

```
pDeque->pRearNode->pRLink = pNode;
pNode->pLLink = pDeque->pRearNode;
pDeque->pRearNode = pNode;
```

연결 리스트로 덱(deque) 구현하기

- 뒤 제거/반환

  - 공통

```
pReturn = pDeque->pRearNode;
pDeque->pRearNode = pDeque->pRearNode->pLLink;
pReturn->pLLink = NULL;
```



  - 노드가 2개 이상인 덱 (일반적인 경우)

```
pDeque->pRearNode->pRLink = NULL;
```

  - 노드가 1개인 큐

```
pDeque->pFrontNode = NULL;
```