9강. 양방향 연결리스트

연습문제

1. 양방향 연결리스트 클래스의 insert_front() 메서드의 알고리즘을 의사코드로 표현하시오.

정답 :

```
def insert_front(self, item):
        dnode = DNode(item)             # create a new node
        dnode.prev = None
        dnode.next = self.head  # The new node points to the previous first node
        if (self.head != None):
                self.head.prev = dnode
        self.head = dnode
        return
```

해설 : 먼저 추가할 item으로 새로운 노드를 만들고, 이 노드의 next 링크가 기존의 첫 번째 노드를 가리키도록 한다. (기존에 비어 있는 리스트였다면 None(NULL)을 가리키면 됨). 새로운 노드가 첫 번째 노드가 되므로 새로운 노드의 prev 링크는 None(NULL)을 가리킨다.

만약 기존에 비어있는 리스트가 아니었다면, 기존에 첫 번째 노드의 prev 링크는 새로 추가된 노드를 가리키도록 한다. 마지막으로 self.head(연결리스트에서 첫 번째 노드를 가리킬 변수)가 새로운 노드를 가리키도록 하면 된다.

2. 양방향 연결리스트가 변수 head를 통해 첫 번째 노드만 직접 접근할 수 있을 때, 연결리스트의 마지막에 새로운 노드를 추가하는 insert_back() 메서드의 최악의 시간복잡도를 구하시오.

정답 : O(N)

해설 : 처음 노드부터 링크를 순서대로 따라가며 마지막 노드를 찾고, 마지막 노드에 새로운 노드를 추가해야 한다. 따라서 스캔하는 연산의 반복 횟수가 노드의 개수에 비례한다. 따라서 시간복잡도는 O(N)이다.

3. 양방향 연결리스트가 변수 head를 통해 첫 번째 노드만 직접 접근할 수 있을 때, 연결리스트의 처음에 새로운 노드를 추가하는 insert_front() 메서드의 최악의 시간복잡도를 구하시오.

정답 : O(1)

해설 : 처음 노드를 head를 통해 직접 접근할 수 있으므로 노드의 개수에 상관없이 한번에 처음 노드를 발견할 수 있다. 따라서 시간복잡도는 O(1)이다.