연습문제

6.1 다음 중 연결된 구조의 리스트에 대한 설명이 아닌 것은?

① 항목들이 메모리에 연속적으로 저장되어 있지 않다.
② 링크를 사용해 다음 항목을 찾는다.
③ 시작 항목을 알면 k번째 항목의 위치를 바로 계산할 수 있다.
④ 용량이 고정되어 있지 않다.

6.2 다음 중 원형 연결 리스트의 특징으로 가장 적절한 것은?

① 마지막 노드의 링크가 첫 노드를 가리킨다.
② 반드시 두 개의 링크를 가진 노드를 사용해야 한다.
③ 선행 노드를 바로(O(1)) 찾아갈 수 있다.
④ NULL (또는 None) 링크가 존재한다.

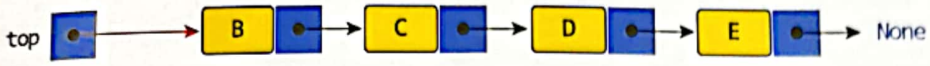
6.3 n개의 자료로 구성된 리스트를 단순 연결 구조로 표현하려고 한다. 다음 중 시간 복잡도 가장 낮은 연산은?

① 리스트의 길이 출력
② 어떤 노드의 다음 노드를 리스트에서 삭제
③ 어떤 노드의 바로 앞에 새로운 노드를 추가
④ 마지막 노드의 데이터를 출력

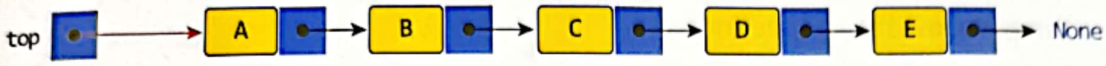
6.4 연결된 구조로 구현한 리스트에 대한 설명으로 가장 거리가 먼 것은?

① 노드의 삽입과 삭제가 쉽다.
② 노드들을 링크로 연결되어 있어 탐색이 빠르다.
③ 링크를 위한 추가 공간이 필요하다
④ 연결 중간에 링크가 끊어지면 그 다음 노드를 찾기 어렵다.

6.5 다음은 연결된 스택을 표현한 것이다. 이에 대한 설명으로 옳지 않은 것은?



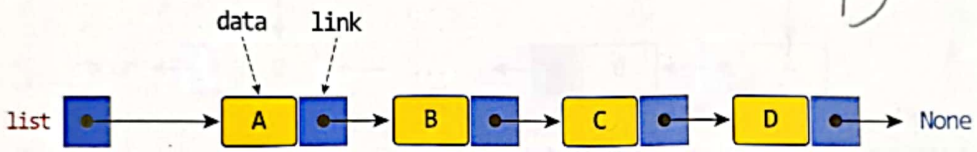
(a) push(A) 연산 수행 전 스택



(b) push(A) 연산 수행 후 스택

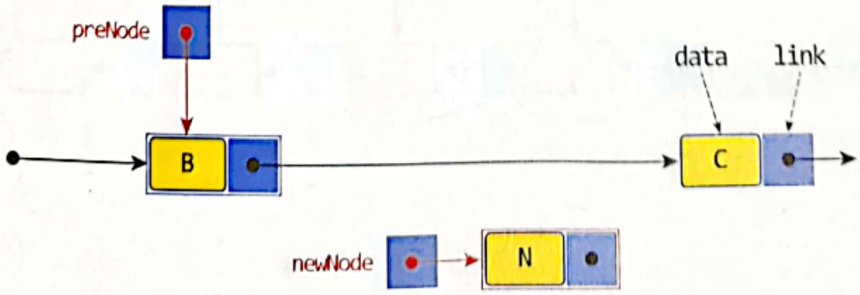
① 스택에 가장 최근에 입력된 자료는 top이 가리킨다.
② 스택에 입력된 자료 중 E가 가장 오래된 자료이다.
③ (b)에서 자료 C를 가져오려면 pop 연산이 2회 필요하다.
④ (a)에서 자료가 입력된 순서는 E, D, C, B이다.

6.6 다음과 같은 연결 리스트에 아래와 같은 코드를 실행한다고 하자. 실행이 끝난 후에 포인터 p가 가리키는 노드는 어떤 노드인가?




```
p = list
while p.link != None :
    ...
    p = p.link
```

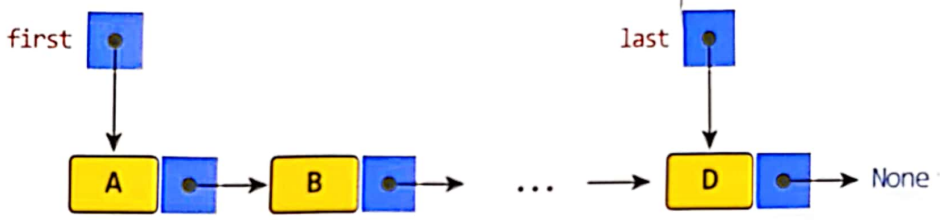
6.7 연결 리스트(linked list)의 "preNode" 노드와 그 다음 노드 사이에 새로운 "newNode" 노드를 삽입하려고 한다. 빈 칸에 들어갈 문장으로 옳은 것은?



```
...
newNode = Node(N, None)
[                                        ]
preNode.link = newNode
```

① newNode.link = preNode
② newNode.link = preNode.link
③ newNode.link.link = preNode
④ newNode = preNode.link

6.8 덱은 삽입과 삭제가 양끝에서 임의로 수행되는 자료구조이다. 다음 그림과 같이 단순 연결 리스트로 덱을 구현한다고 할 때 O(1) 시간 내에 수행할 수 없는 연산은? (단, first와 last는 각각 덱의 첫 번째 원소와 마지막 원소를 가리킨다.



① addFirst 연산: 덱의 첫 번째 원소로 삽입
② addLast 연산: 덱의 마지막 원소로 삽입
③ deleteFirst 연산: 덱의 첫 번째 원소를 삭제
④ deleteLast 연산: 덱의 마지막 원소를 삭제

6.9 다음은 연결 리스트를 역순으로 변화하는 함수이다. 빈칸에 들어가야 하는 코드를 적으시