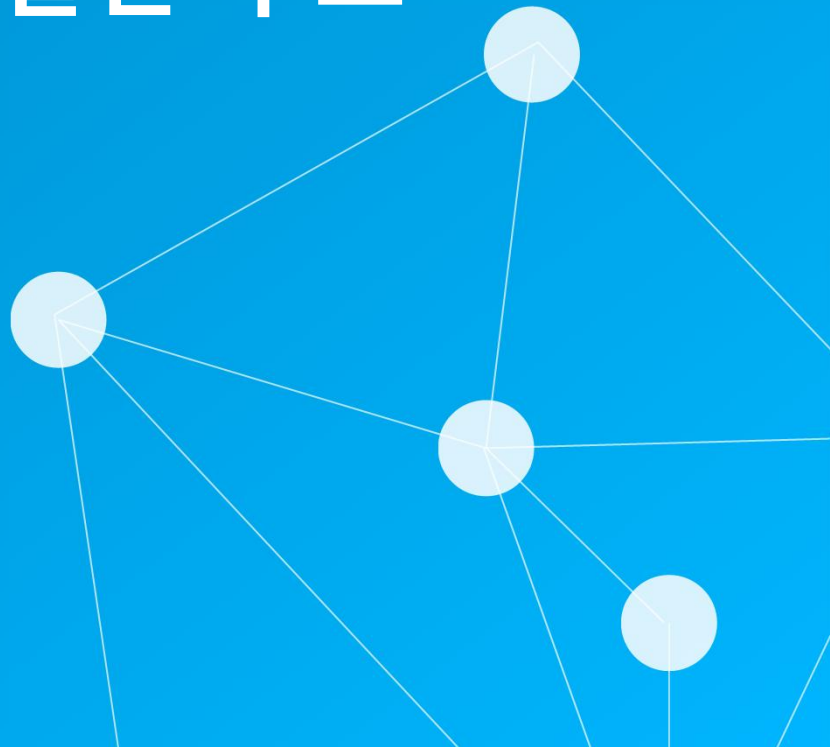
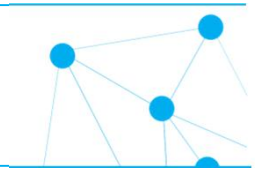
파이썬
자료구조

CHAPTER

연결된 구조

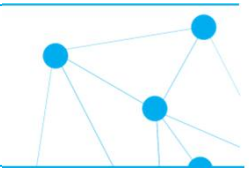


6.1 연결된 구조란?

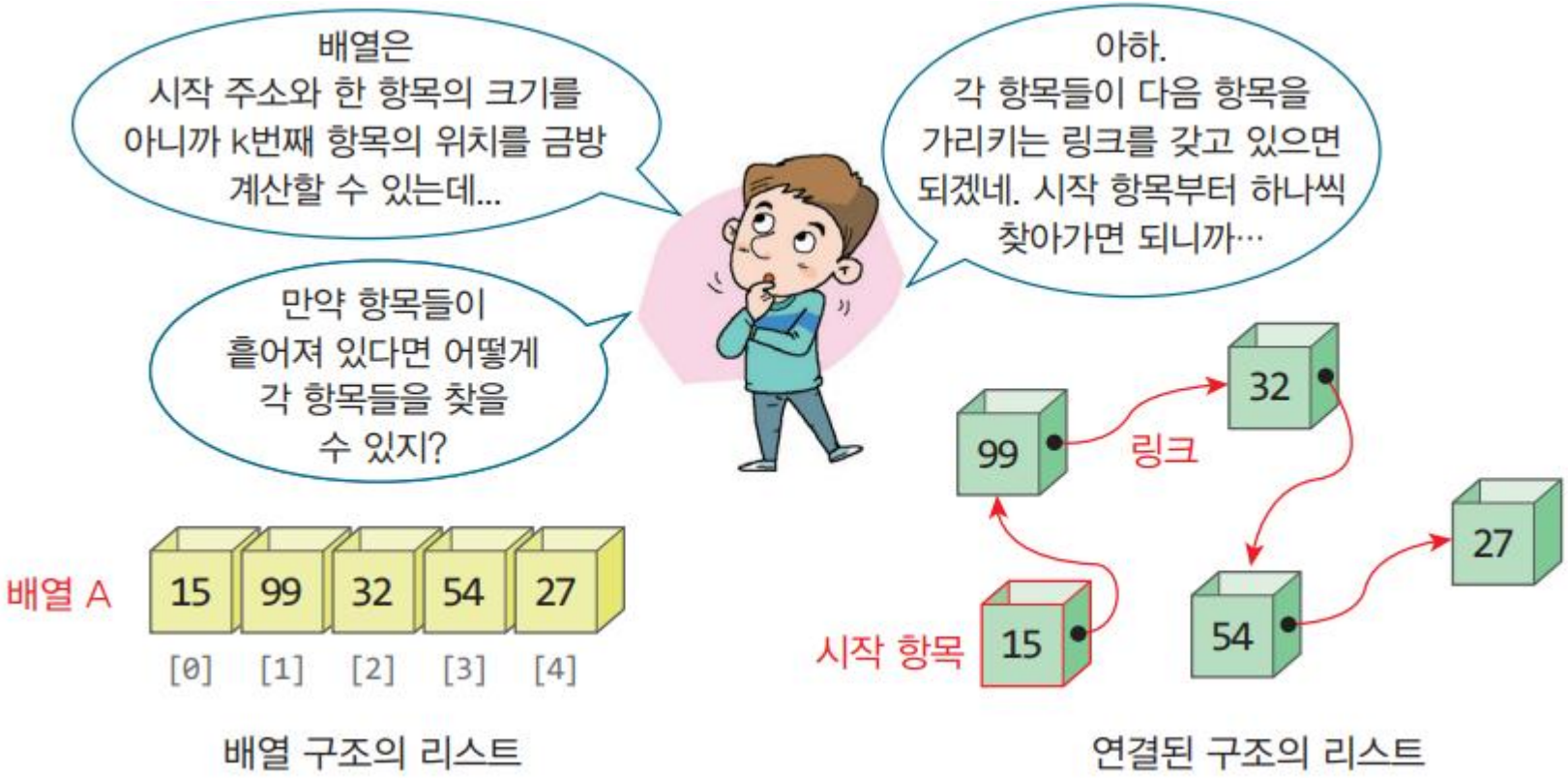


- 연결된 구조는 흩어진 데이터를 링크로 연결해서 관리한다.
- 연결된 구조의 특징
- 연결리스트의 구조
- 연결리스트의 종류

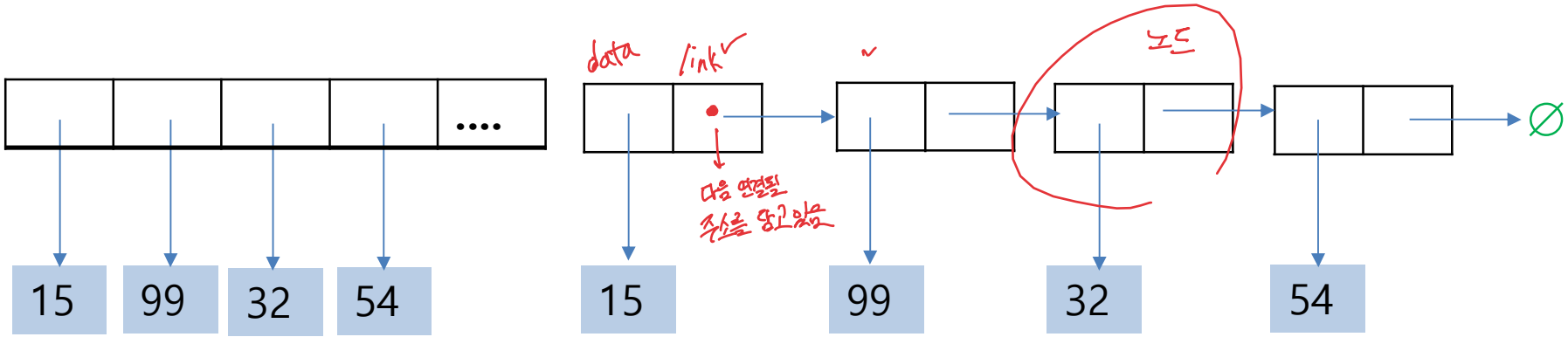
연결된 구조란?



- 연결된 구조는 흩어진 데이터를 링크로 연결해서 관리



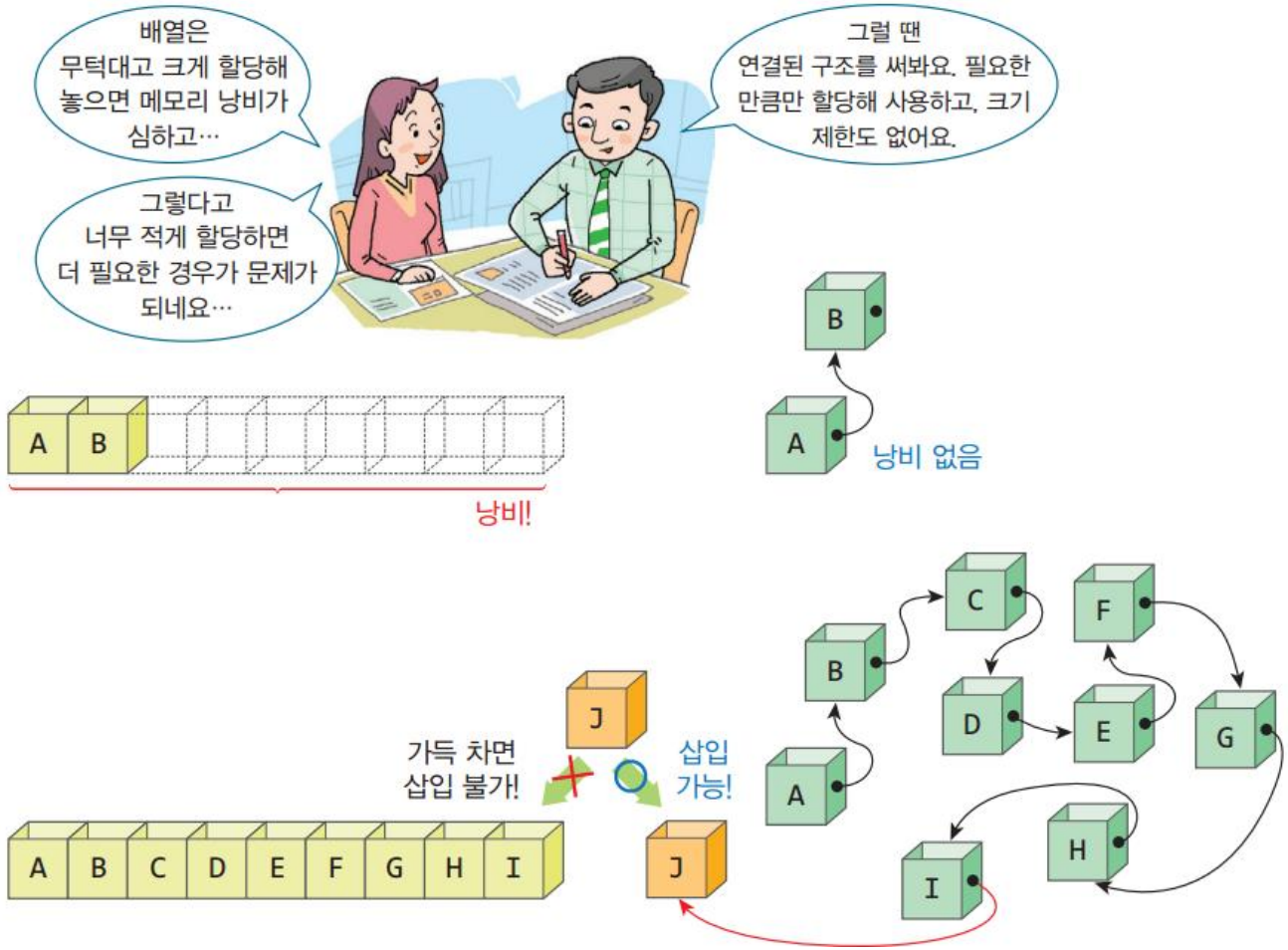
연결된 구조의 예



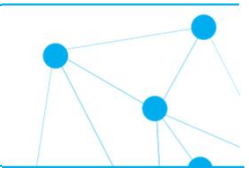
배열에서는 남는 메모리가 있는데
이와 같은 구조는 내용이 있는 것들만
연결되므로 메모리를 효율적으로 사용

연결된 구조의 특징

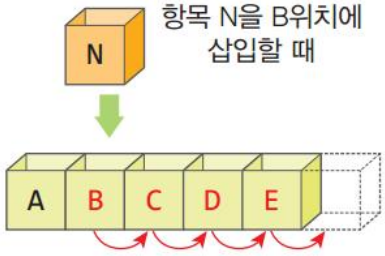
- 용량이 고정되지 않음.

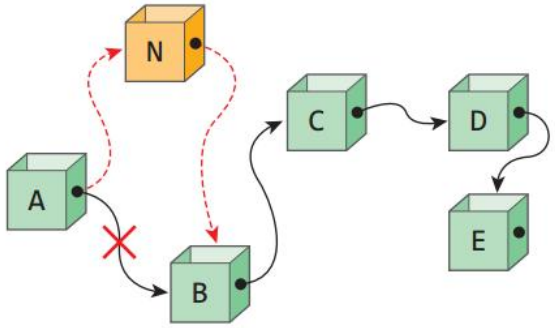


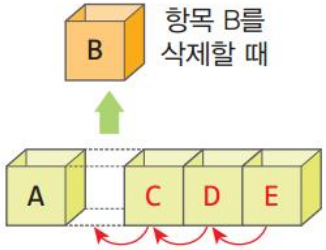
연결된 구조의 특징

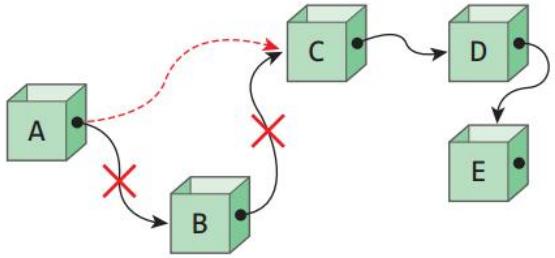


- 중간에 자료를 삽입하거나 삭제하는 것이 용이



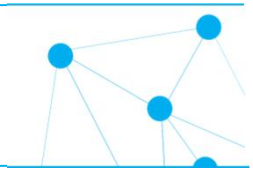




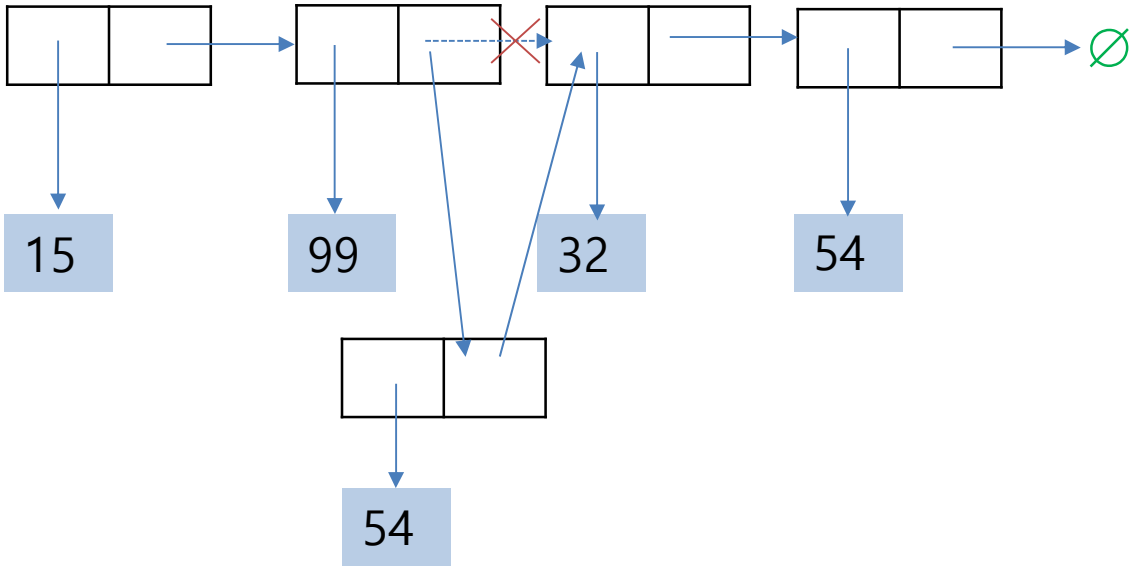


- n번째 항목에 접근하는데 $O(n)$의 시간이 걸림.

연결된 구조의 특징

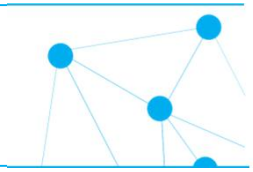


- 중간에 자료를 삽입하거나 삭제하는 것이 용이



- n번째 항목에 접근하는데 $O(n)$의 시간이 걸림.

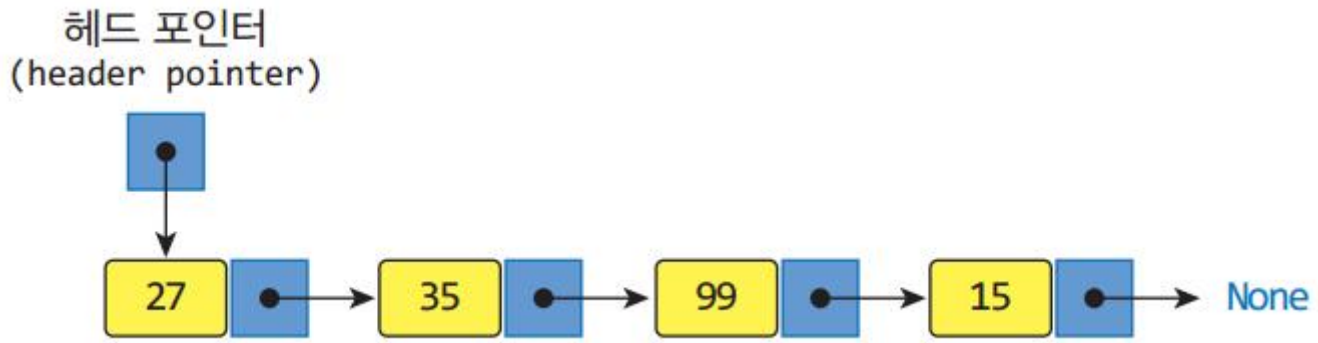
연결 리스트의 구조



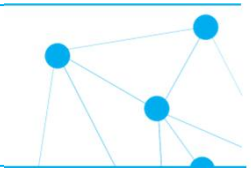
- 노드 (node)
  - 데이터 필드(data field)
  - 하나 이상의 링크 필드(link field)

```
__init__(self, element)
self.data = element
self.link = None
```

- 헤드 포인터 (head pointer) : 연결리스트의 첫번째 노드를 가리킴
- **앞으로, 노드의 data는 편의상 자료를 저장되어 있는 것으로 표기함 (참조 대신)**

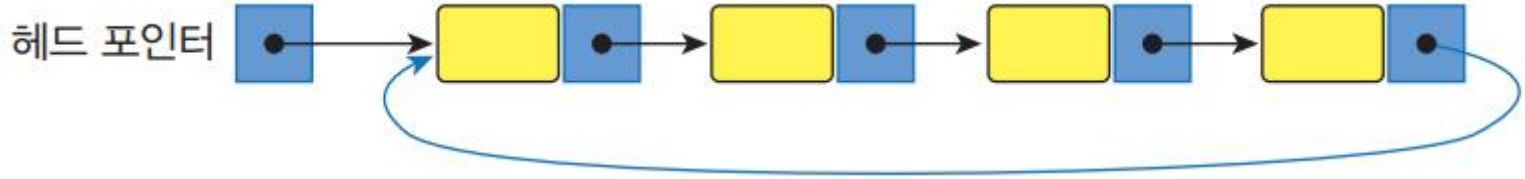


연결 리스트의 종류

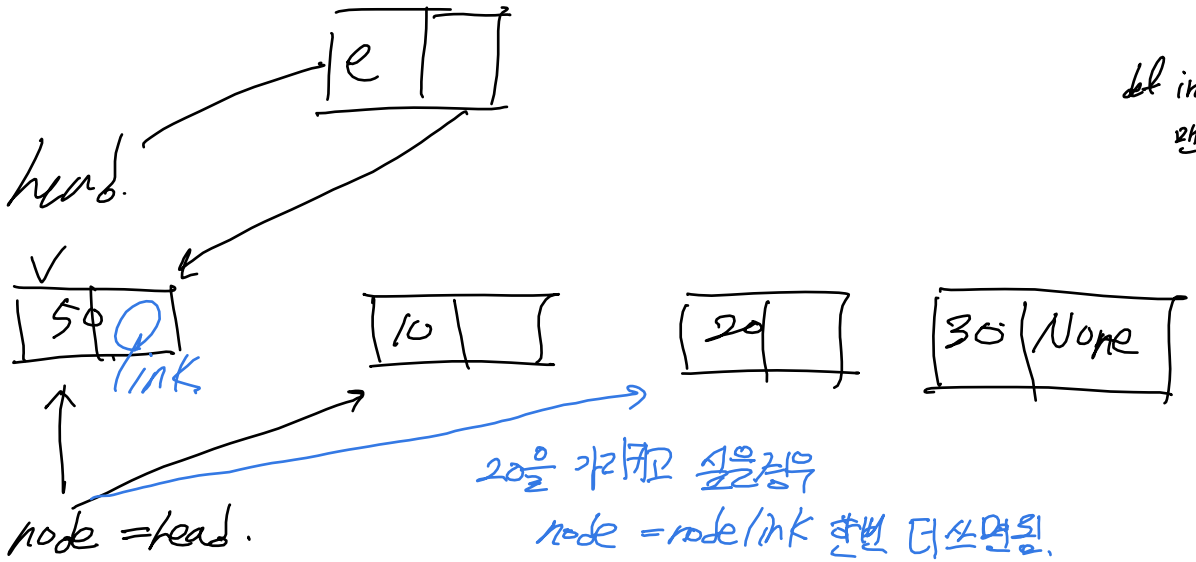


- 단순 연결 리스트(singly linked list)

- 원형 연결 리스트(circular linked list)



- 이중 연결 리스트(doubly linked list)



let insert_front

맨앞에 있는 노드 반환

20을 가리고 싶을경우

node = node.link 한번 더 쓰면됨.

node = node.link

하지만 .node가 몇번일지를 코드가 결정할수 없음.

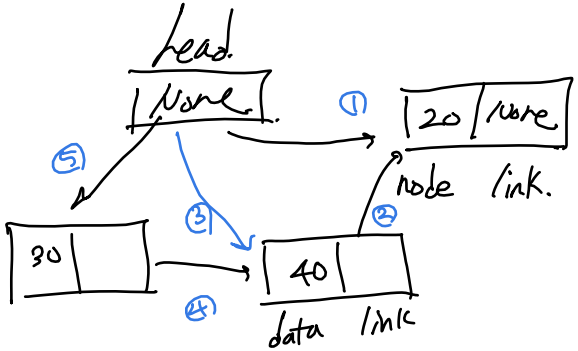
while문을 사용해서 리스트의 값을 출력하는 구조 만들기

Node가 마지막까지 None을 만날때까지

data값 출력

```
while node != None:
    print(node.data, end=' ')
    node = node.link
print()
```

⇒ Node의 값이 None 일 경우 None 출력이 가능해짐.



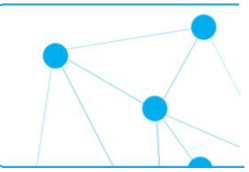
6.2 단순연결리스트 응용: Linked Stack

- 노드

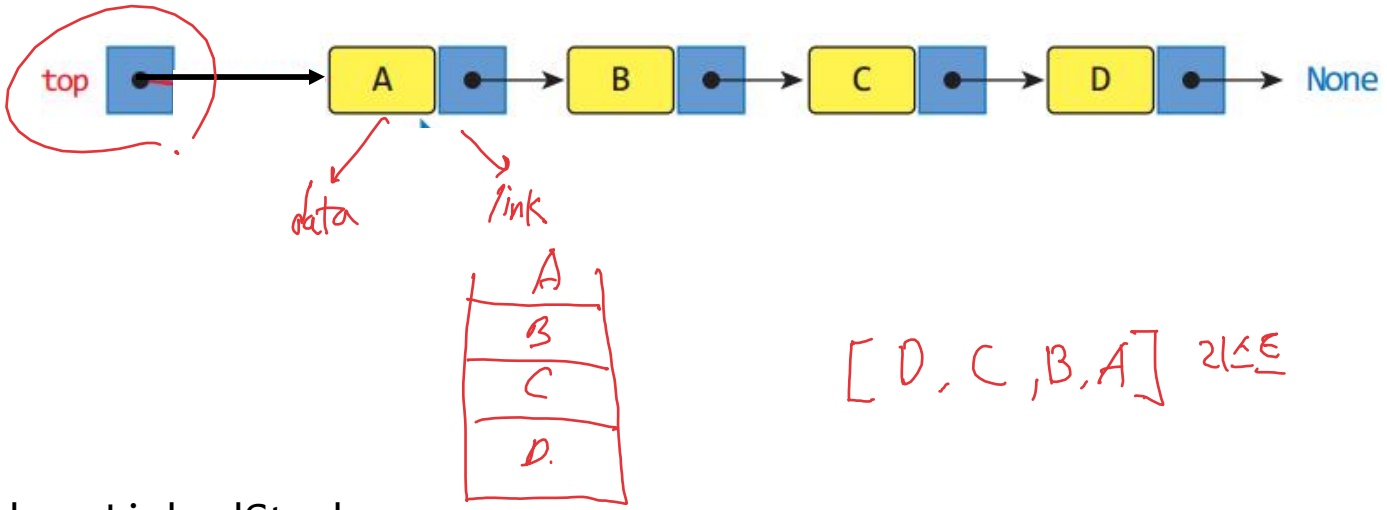
- 노드 클래스

```
class Node:
    def __init__(self,element):
        self.data = element
        self.link = None
```

단순연결리스트 응용: 연결된 스택



- 연결된 스택 클래스



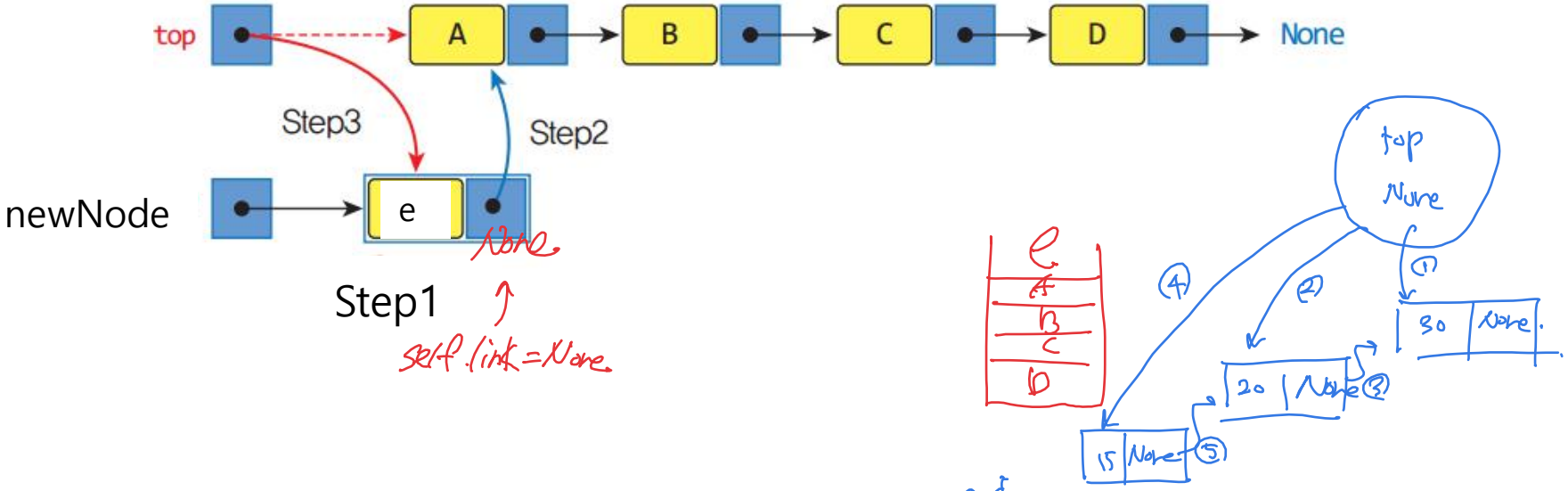

```
class LinkedStack:
    def __init__(self):
        self.top = None

    def isEmpty(self):
        return self.top == None
```

객체 생성 초기화부분.

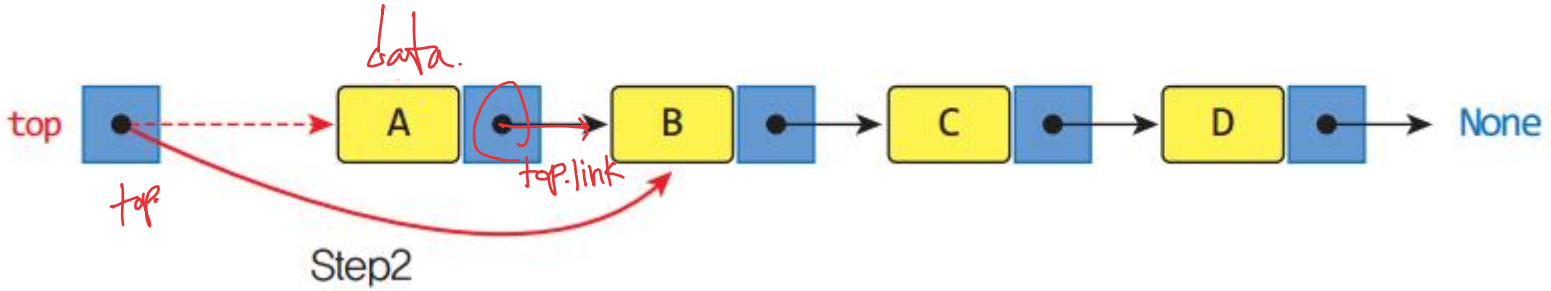
top은 연결된 리스트에 첫번째 노드를 가리킴.

삽입 연산




```
def push(self,e):
    newNode = Node(e)       # Step 1
    newNode.link = self.top # Step 2
    self.top = newNode      # Step 3
```

삭제 연산




```
def pop(self):
        if (self.empty()):
            print("Stack is empty")
            return
        e = self.top.data
        self.top = self.top.link
        return e
```

- 메모리 해제를 신경 쓸 필요 없음!