Linked List 의 구현

Python에서의 Linked List 구현 예

Single Linked List

단순 연결 리스트

```
class SList:
    class Node:
        def __init__(self, item, link): # 노드 생성자
            self.item = item            # 저장 항목
            self.next = link            # 다음 노드 레퍼런스

    def __init__(self):             # 단순연결리스트 생성자
        self.head = None            # head
        self.size = 0               # 항목 수

    def size(self): return self.size
    def is_empty(self): return self.size == 0
```

```
def insert_front(self, item):
    if self.is_empty():                # 첫 노드로 삽입
        self.head = self.Node(item, None)
         # head가 새 노드 참조
    else:
        self.head = self.Node(item, self.head)
    self.size += 1

 def insert_after(self, item, p):        # p 다음에 삽입
    p.next = SList.Node(item, p.next)
    self.size += 1
```

```
    def delete_front(self):
        if self.is_empty():     # 첫 노드 삭제
            raise EmptyError('Underflow')
        else:
            self.head = self.head.next
            self.size -= 1

    def delete_after(self, p):      # p 다음 노드 삭제
        if self.is_empty():         # 비어있는 경우
            raise EmptyError('Underflow')
        t = p.next
        p.next = t.next             # p 다음 노드를 건너뛰어 연결
        self.size -= 1
```

```
def search(self, target):        # target 탐색
        p = self.head            # head 로부터 순차 탐색
        for k in range(self.size):
            if target == p.item: return k     # 탐색 성공
            p = p.next
        return None                    # 탐색 실패

    def print_list(self): # 연결리스트 출력
        p = self.head
        while p:
            if p.next != None:   print(p.item, ' -> ', end='')
            else:                print(p.item)
            p = p.next          # 노드들을 순차 탐색
```

```
s = SList()
s.insert_front('orange')
s.insert_front('apple')
s.insert_after('cherry', s.head.next)
s.insert_front('pear')
s.print_list()
print('cherry는  %d번째' % s.search('cherr
print('kiwi는', s.search('kiwi'))
print('배 다음 노드 삭제 후:₩t₩t', end='')
s.delete_after(s.head)
s.print_list()
```

```
print('첫 노드 삭제 후:₩t₩t', end='')
s.delete_front()
s.print_list()
print('첫 노드로 망고,딸기 삽입 후:₩t',
end='')
s.insert_front('mango')
s.insert_front('strawberry')
s.print_list()
s.delete_after(s.head.next.next)
print('오렌지 다음 노드 삭제 후:₩t', end='')
s.print_list()
```

Double Linked List

이중 연결 리스트

```
class DList:
    class Node:
        def __init__(self, item, prev, link):          # 노드 생성자
            self.item = item
            self.prev = prev                            # 앞 노드 레퍼런스
            self.next = link                            # 뒤 노드 레퍼런스

    def __init__(self):                                 # 이중연결리스트 생성자
        self.head = self.Node(None, None, None)
        self.tail = self.Node(None, self.head, None)
        self.head.next = self.tail
        self.size = 0                                   # 항목 수
```

```
def insert_before(self, p, item): # p 앞에 삽입
        t = p.prev
        n = self.Node(item, t, p)       # 새 노드 생성하여 n 이 참조
        p.prev = n                      # 새 노드와 앞 노드 연결
        t.next = n                      # 새 노드와 뒤 노드 연결
        self.size += 1

def insert_after(self, p, item): # p 다음에 삽입
        t = p.next
        n = self.Node(item, p, t)
        t.prev = n
        p.next = n
        self.size += 1
```

```
def delete(self, x):          # x가 참조하는 노드 삭제
        f = x.prev
        r = x.next
        f.next = r            # x 를 건너뛰고 x의 앞뒤 노드를 직접 연결
        r.prev = f
        self.size -= 1
        return x.item

 def size(self):              return self.size

 def is_empty(self):          return self.size == 0
```

```
def print_list(self):            # 리스트 출력
    if self.is_empty():
        print('리스트 비어있음')
    else:
        p = self.head.next
        while p != self.tail:
            if p.next != self.tail:
                print(p.item, ' <=> ', end='')
            else:
                print(p.item)

            p = p.next              # 노드들을 차례대로 방문
```

```
s = DList()
s.insert_after(s.head,'apple')
s.insert_before(s.tail, 'orange')
s.insert_before(s.tail,'cherry')
s.insert_after(s.head.next,'pear')
s.print_list()
print('마지막 노드 삭제 후:₩t', end='')
s.delete(s.tail.prev)
s.print_list()
print('맨 끝에 포도 삽입 후:₩t', end='')
s.insert_before(s.tail,'grape')
s.print_list()
```

```
print('첫 노드 삭제 후:₩t', end='')
s.delete(s.head.next)
s.print_list()
print('첫 노드 삭제 후:₩t', end='')
s.delete(s.head.next)
s.print_list()
print('첫 노드 삭제 후:₩t', end='')
s.delete(s.head.next)
s.print_list()
print('첫 노드 삭제 후:₩t', end='')
s.delete(s.head.next)
s.print_list()
```