동아대학교

데이터 구조

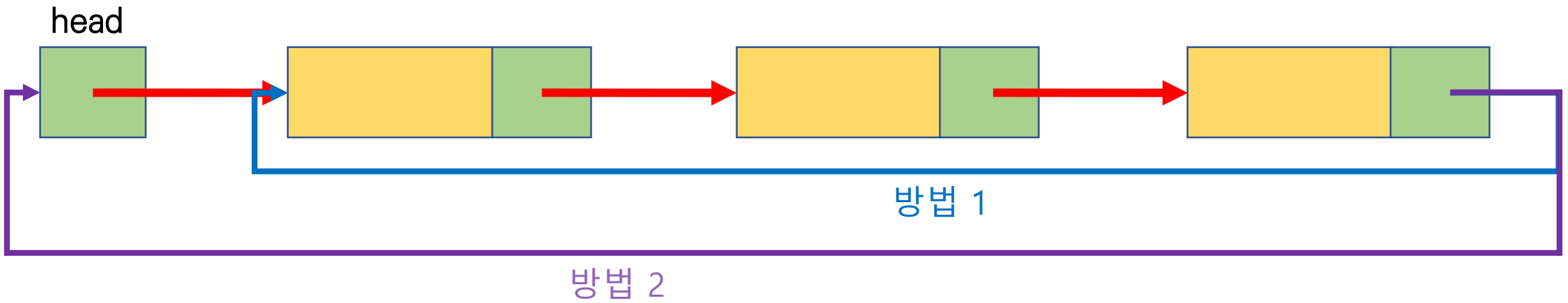
5주차: 원형 연결리스트

박 경 훈

원형연결리스트 필요 메소드

1. HEAD 생성
2. 마지막 위치에 데이터 추가
3. 원하는 위치에 데이터 추가
4. 데이터 전체 순차 출력
5. 데이터 삭제
6. 데이터 수정
7. 원하는 위치 검색

원형연결리스트 구조



- 원형 연결리스트는 단순연결리스트를 바탕으로 구성된 구조이며, 마지막 위치가 곧 시작 위치가 된다.
- 원형리스트를 코딩할 때는 마지막 노드가 head 주소를 가르킬 것인지 첫번째 노드를 가르킬 것인지 결정하여 코드

Q. 원형 연결리스트는 처음과 마지막의 위치가 같다. 어떤 부분을 조심해야 할까?

Q. 파이썬 에서는 다른 언어에서는 존재하는 Do ~ while 구문이 없다. 해결방안은?

※. Do ~ While 문이란?

원형연결리스트 주요코드

방법 1

```
def node_append(self, data):
    if self.head is None:
        return 99, "Error : Head Node is not Create..."

    current = self.head
    last = self.head

    while True:
        current = current.next
        if current == None:
            newNode = self.Node(data)
            self.head.next = newNode
            newNode.next = newNode
            return 100, "Successfully, appended data!!"

        if current.next == last.next:
            break;

    newNode = self.Node(data)
    newNode.next = current.next
    current.next = newNode

    return 100, "Successfully, appended data!!"
```

방법 2

```
def node_append(self, data):
    if self.head is None:
        return 99, "Error : Head Node is not Create..."

    current = self.head
    last = self.head

    while True:
        current = current.next
        if current == None:
            newNode = self.Node(data)
            self.head.next = newNode
            newNode.next = self.head
            return 100, "Successfully, appended data!!"


        if current.next == last:
            #print("append: " & current.data)
            break;

    newNode = self.Node(data)
    current.next = newNode
    newNode.next = self.head

    return 100, "Successfully, appended data!!"
```

원형연결리스트 주요코드

원형리스트는 처음과 마지막이 같으므로 데이터 추가, 삭제 코드 작성시

두 개 위치 변수(head, last)을 두고,

그 중 head 를 변경하여 Node 를 이동하고,

head와 last 를 비교하여 마지막 위치 즉, Node를 전부 순회했는지 여부를 판단한다.

원형연결리스트

- 장점 : 하나의 노드에서 모든 노드를 접근할 수 있다.

Try

- 본 자료 4Page를 참조하여 원형리스트 삽입, 삭제, 수정 코드를 스스로 작성해 보세요.
- 이중노드 구조를 이용하여 원형리스트 구현 해 보세요.