# Data Structure

**Spring 2019**

**M 16:00-18:00 W 11:00-13:00**

**http://smart.hallym.ac.kr**

**Instructor:** **Jin Kim**
**010-6267-8189(033-248-2318)**
**jinkim@hallym.ac.kr**

**Office Hours:**

# Lab(linked list-basic)

**Spring 2019**

**http://smart.hallym.ac.kr**

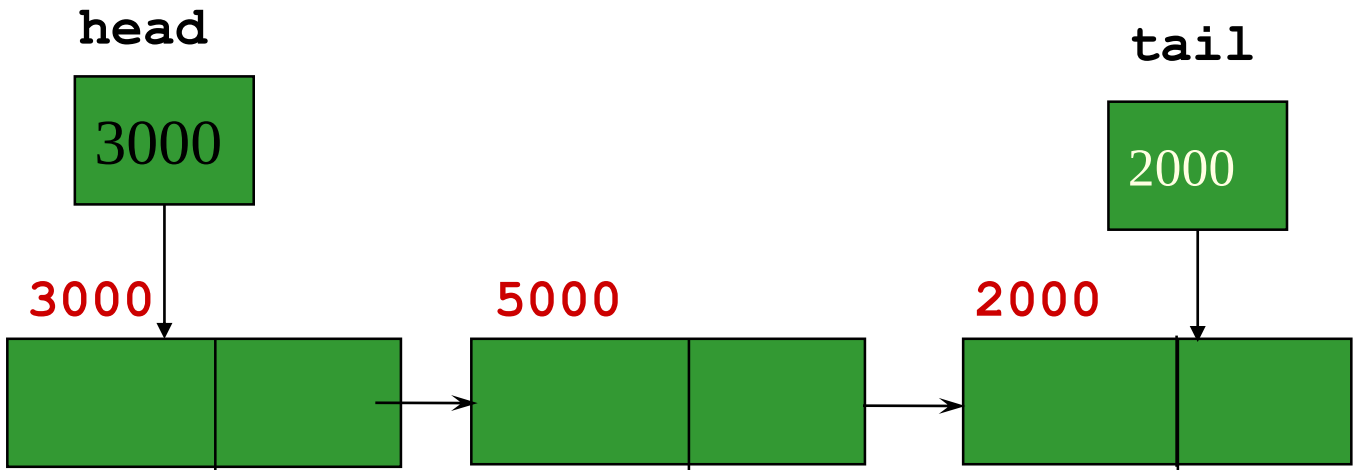
**Instructor: Jin Kim**

**010-6267-8189(033-248-2318)**

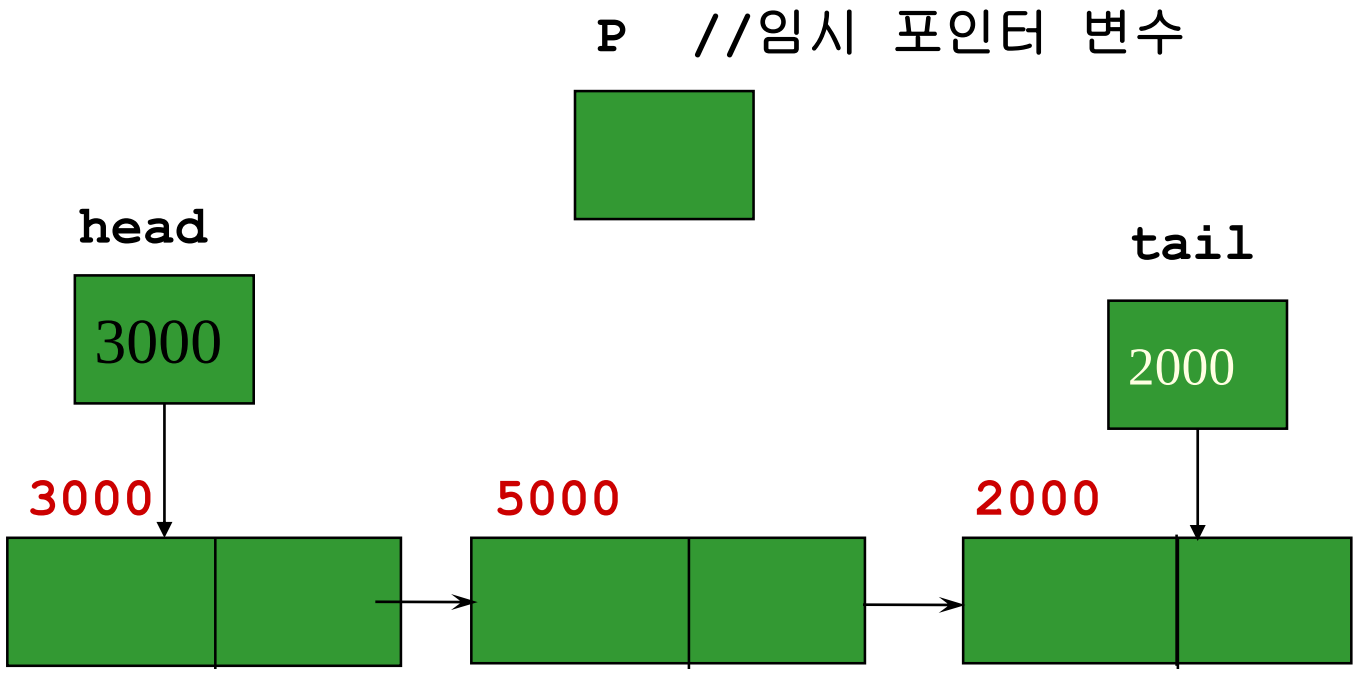
**jinkim@hallym.ac.kr**

**Office Hours:**

# LinkedList



# LinkedList



# 모든 노드를 한번씩 방문

Traversing a linked list  //리스트의 노드방문코드

```
// 리스트의 마지막 노드를 찾음+마지막노드의 데이터를 접근할 수 없다

p ← L;      // p는 임시 순회 포인터 변수
while p.link ≠ null do
    p ← p.link;
```

# 같은 내용의 코드

```
// 리스트의 마지막 노드를 찾음+마지막노드의 데이터를 접근할 수 있다

p ← L;      // p는 임시 순회 포인터 변수
while p ≠ null do {
        if (p.data==“???”){ // do something
        }
      p ← p.link;
}
```

```
// 리스트의 마지막 노드를 찾음+마지막노드의 데이터를 while 안에서 접근할 수 없다

p ← L;      // p는 임시 순회 포인터 변수
while p.link ≠ null do {
        if (p.data==“???”){ // do something
        }
     p ← p.link;
}
if (p.data==“???”){ // 마지막 노드의 데이터 접근
}
```

- addFirst() 메소드를 구현하라
- addLast() 메소드를 구현하라
- deleteFirst() 메소드를 구현하라
- deleteLast() 메소드를 구현하라
- getNumberOfNode() 메소드를 구현하라. 노드의 개수
- replaceNode(x, y) 메소드를 구현하라 // x를 y로 교체

# printList()

```
ListNode p;
 p = head;
 System.out.print("(");
 // 원소를 방문하는 방식 1
 while (p != null) {
                System.out.print(p.data + " ");
                p = p.link;
 }

 // 원소를 방문하는 방식2 테스트해보라
//               for (p = head; p != tail; p = p.link) {
//                      System.out.print(p.data + " ");
//               }
//               System.out.print(p.data + " ");
               System.out.println(")");
```

# addFirst()

```
ListNode newNode = new ListNode();
newNode.data = x;
newNode.link = null;
if (head == null) { // 첫번째 원소 생성시
      head = newNode;
      tail = newNode;
} else { // 원소가 이미 존재하면
    newNode.link = head; // 기존의 마지막 원소의 링크에 새로이 찍은 빵을 연결
    head = newNode;  // 새로운 원소가 마지막 원소로 추가됨
}
```

# addLastNode()

```
                //빵을 찍고
//              ListNode newNode = new ListNode();
//              newNode.data = x;
//              newNode.link = null;
                // 빵을 다른 생성자를 사용하여 이렇게 찍을수도 있다.
                ListNode newNode = new ListNode(x);
                if (head == null) { // 첫번째 원소 생성시 선두원소이자 마지막 원소이므로
                        head = newNode;
                        tail = newNode;
                } else { // 원소가 이미 존재하면
                        tail.link = newNode; // 기존의 마지막 원소의 링크에 새로이 찍은 빵을 연결
                        tail = newNode;  // 새로운 원소가 마지막 원소로 추가됨
                }
```

# deleteFirst()

```
String x = " ";
if (head == null) {
        return "No element";
}
x = head.data;
head = head.link;
if (head == null) { tail = null; }
return x;
```

# deleteLast()

```
String x;
ListNode p;
ListNode q;

if (head == null) {
        return "No element found";
}
p = head; q = null;
while (p.link != null) {
        q = p;
        p = p.link;

}
// q는 마지막 -1 번 원소, p는 마지막 원소를 가르킴
x = p.data;
tail = q;
q.link = null;
return x;
```

1. LinkedListMain.java

Zip all your programs(name.zip) and upload to smart.hallym.ac.kr

# 도전정신으로 해결해보라

- deleteNode(int key) : key값을 가진 노드를 제거하라.
- printMiddle() : 리스트의 원소들 가운데 중앙에 있는 값을 출력하라.
  - 힌트 : 두 개의 포인터를 사용. 하나는 매번 두개노드전진, 다른하나는 한번 전진. 첫번째가 마지막에 도달하면 두번째 원소가 중앙값.
- reverse() : 리스트를 뒤집어라.
- printNthFromLast() : 마지막에서부터 n번째 원소를 출력하라.
  - 힌트 : 두개의 포인터 사용. 첫번째포인터는 n만큼 먼저 전진. 이후 두 개의 포인터가 한번씩 전진. 첫번째 포인터가 마지막에 도달하면 두번째 포인터가 마지막에서 n만큼 떨어져 있음.

# Shift.java

- Write a function to rotate a linked list left by one; i.e., the old first element becomes the new last element. Do not use new operator. For example
- 즉 L5 **=("Kim","Lee","Park","Yoo")**를 만들어라. 원소들을 하나씩 오른쪽으로 이동시켜라. 마지막 원소는 첫번째 원소가 되도록 하라. L5는 단순 연결리스트이다.
- **마지막에서 두번째 노드의 링크부분을 null로 변경**
- **마지막노드를 첫번째 노드가 되도록 리스트 이름에 첫번째 노드의 주소입력**
- **기존의 마지막노드의 링크부분을 기존의 선두원소주소저장**

**L5=("Kim","Lee","Park","Yoo")-> L5=("Yoo","Kim","Lee","Park")**

```
p=L;
c=L.link;
while (c.link != null) {
    p = c;
    c = c.link;
}
c.link=L;  //마지막 노드의 null부분에 첫번째 원소주소저장
p.link=null; //마지막에서 두번째 노드 주소에 null저장
L=c; //L에 첫번째 원소의 주소인 C저장
```