자료 구조 Lab 005 :

Lab18005.zip : LabTest.java, lab005.java, lab.in, lab.out, lab005.pdf

---

제출

lab005.java 를 학번.java 로 변경하여 이 파일 한 개만 제출할 것.

다음은 Linked List의 사용에 관한 문제이다.
이번 연습은 Chain에서 원소의 **삽입은 임의의 위치에**서 일어나고, 원소의 **삭제는 항상 처음에서** 일어나는 기능을 제공하는 Chain이라는 클래스를 구현하는 것이다.
수행 예는 다음과 같다.

```
kmucs@localhost: ~/dbox/classes181/ds/lab18/lab18005
kmucs@localhost:~/dbox/classes181/ds/lab18/lab18005$ java LabTest
Chain >
ins 0 1
List (1) : 1

Chain >
ins 0 2
List (2) : 2 1

Chain >
ins 0 3
List (3) : 3 2 1

Chain >
delf
        Output : 3
List (2) : 2 1

Chain >
ins 4 7
Cannot Insert into position 4
List (2) : 2 1

Chain >
```

사용자가 사용하는 명령어의 syntax는 다음과 같다. LabTest.java의 main() 함수에 정의되어 있다.

- ins : ins theIndex item

  정수인 theIndex의 위치에 정수인 item 값을 가지는 노드를 리스트에 추가한다.

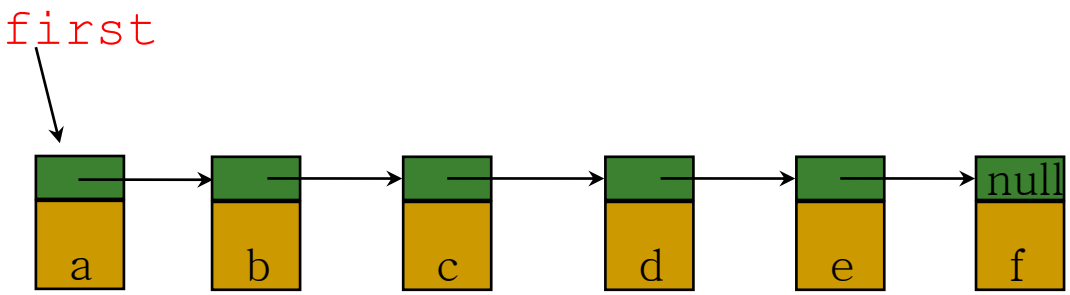
- delf : delf

  리스트의 맨 앞에 있는 노드를 제거한다.
- quit : quit

  프로그램을 종료한다.

명령이 실행되고 나면 리스트의 내용이 출력되는데, **괄호 안의 숫자는 리스트의 원소의 수**이다. 대부분의 알고리즘은 ppt 파일에 있기 때문에 참조하면 된다.

구현할 Chain 클래스는 원소들을 아래와 같은 구조로 유지한다.



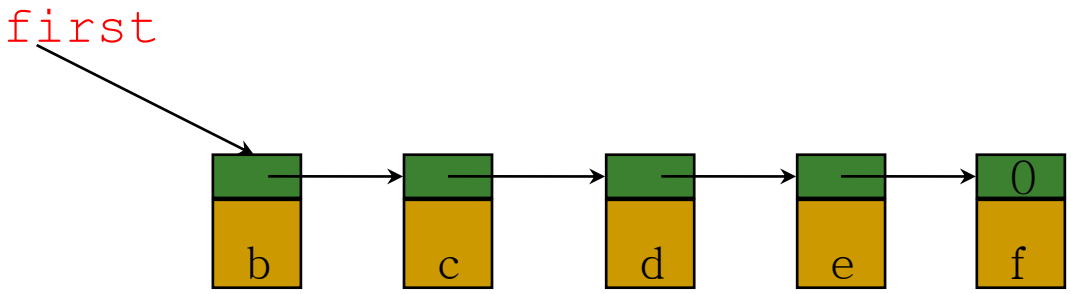
구현이 필요한 부분은 다음 세 함수이다. 구체적인 수행내용도 함께 설명한다.

- final int Size();

  이 함수는 현재 리스트에 들어 있는 원소의 수를 return하는 함수이다. 위 그림에서는 6을 리턴한다.
- boolean Insert(int theIndex, T theElement);

  이 함수는 theIndex가 가리키는 위치에 새로운 노드를 하나 추가 하고, 그의 데이터 필드에 theElement 를 입력하는 함수이다. 인덱스는 0부터 시작한다. 예를 들면 theIndex가 3이라면 위 그림에서 c와 d 사이에 새 노드가 추가되어야 한다. 삽입이 성공하면 true를 return 한다. 하지만 만약 theIndex가 리스트의 길이보다 크다면 (예 theIndex=6) 삽입을 할 수가 없기 때문에 이렇게 삽입이 실패하면 false를 return 한다.
- T DeleteFront();

  리스트의 맨 앞에 있는 노드를 삭제하는 함수이다. 삭제한 후에는 그 노드가 가지는 data의 값을 return 한다. 만약 리스트가 empty 인 경우에는 null을 return한다. 위 그림의 예에서는 DeleteFront() 가 불렸을 때 다음과 같이 변경되어야 한다.



프로그램 테스트

컴파일

```
$ javac lab005.java LabTest.java
```

실행

```
$ java LabTest
```

주어진 input으로 실행

```
$ java LabTest < lab.in
```

주어진 output과 비교

```
$ java LabTest < lab.in > abc
$ diff abc lab.out
```