



2020 학년도 1 학기

컴퓨터 정보과

자료구조(Data Structures)

담당교수 : 김주현

제 3 주차 / 제 3 차시

자바 리뷰 추천사이트

<http://tcpschool.com/java/intro>

<https://wikidocs.net/book/31>

<https://opentutorials.org/module/516>

<https://e-koreatech.step.or.kr/page/lms/?m1=home%25>

tcpschool 자바

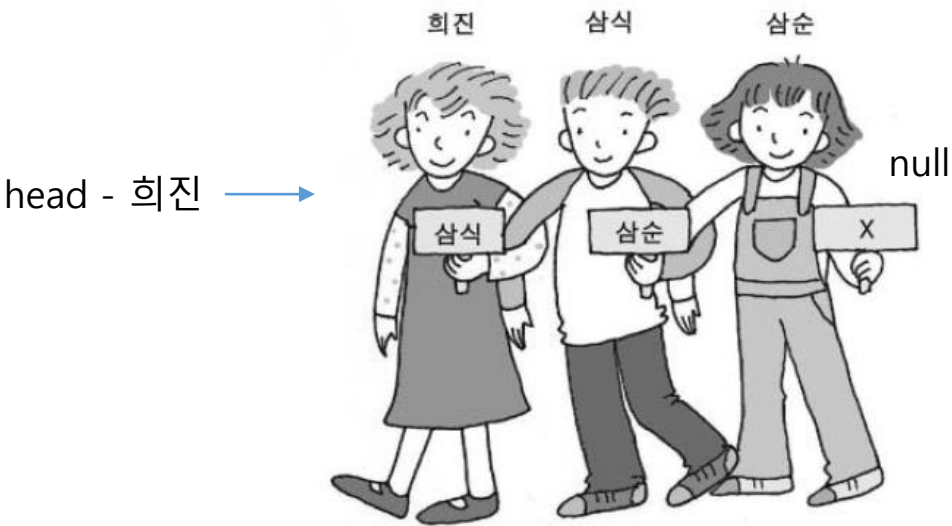
점프투자바

생활코딩 자바

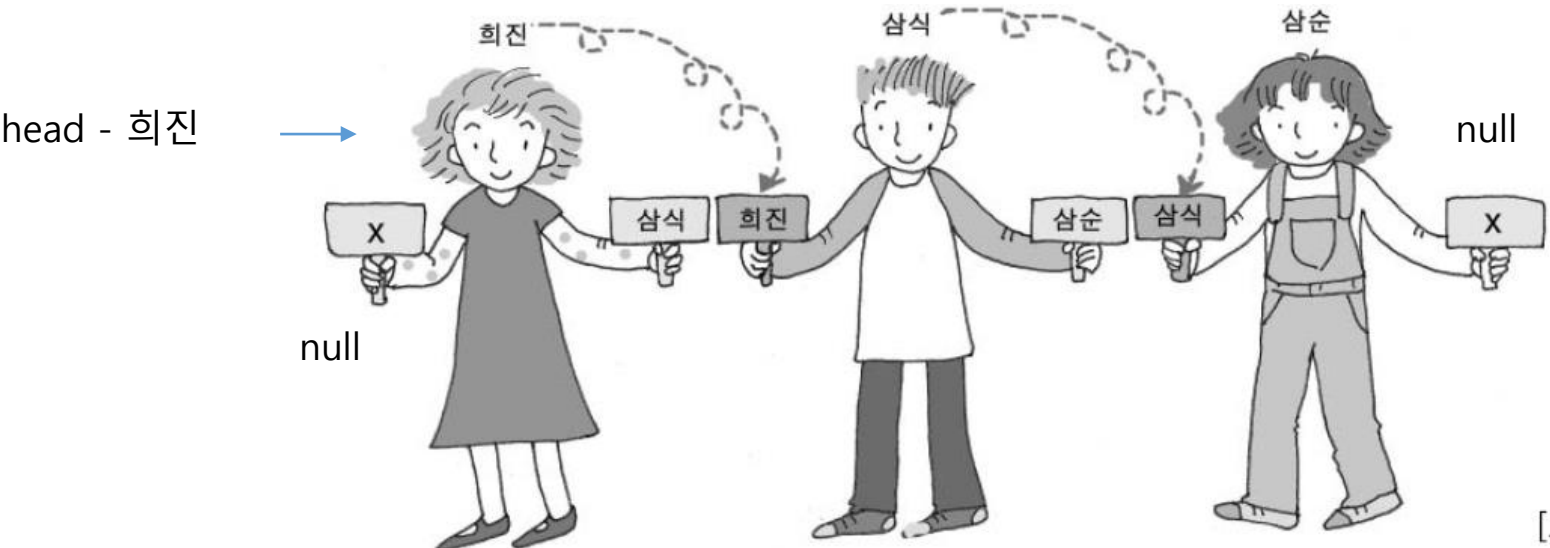
한국기술교육대학교

잘 이해가 되지 않는 부분은 구글이나 유튜브에서 검색을 해 보세요!
단톡방을 매일 확인하고 있으니 단톡방도 활용하길 바랍니다.

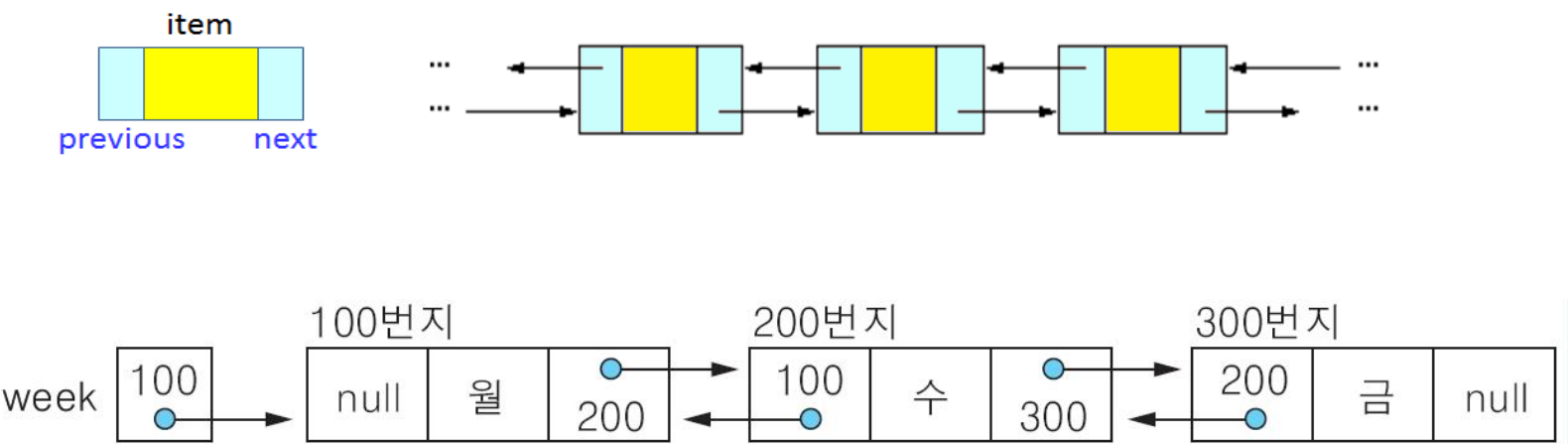
단순 연결 리스트



이중 연결 리스트



이중 연결 리스트(doubly linked list)



이중 연결 리스트의 노드를 위한 Dnode 클래스

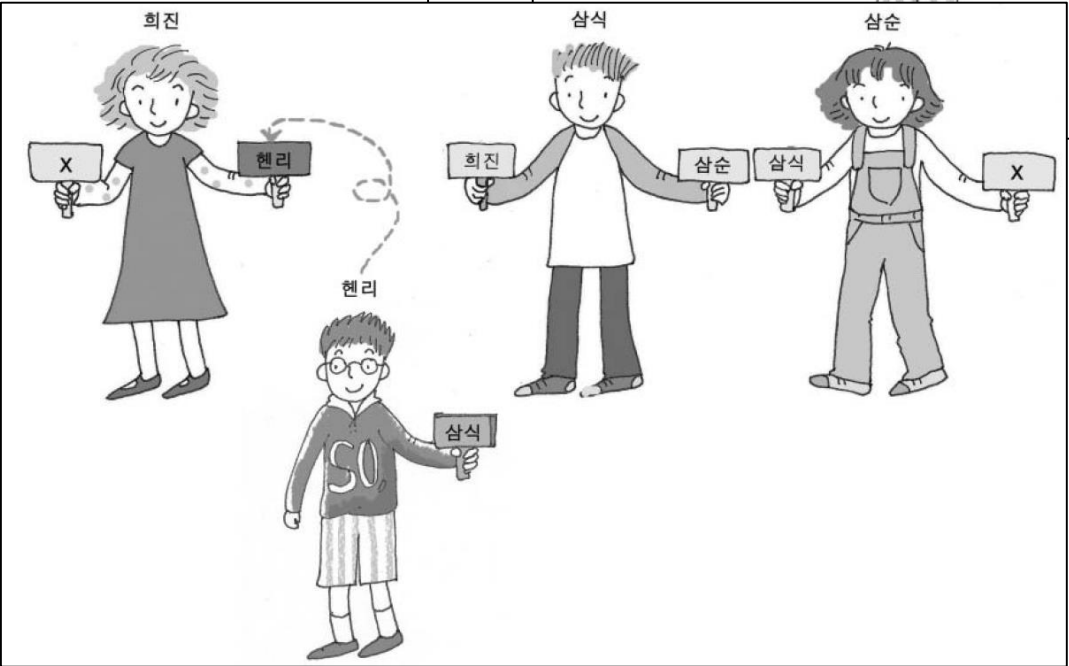
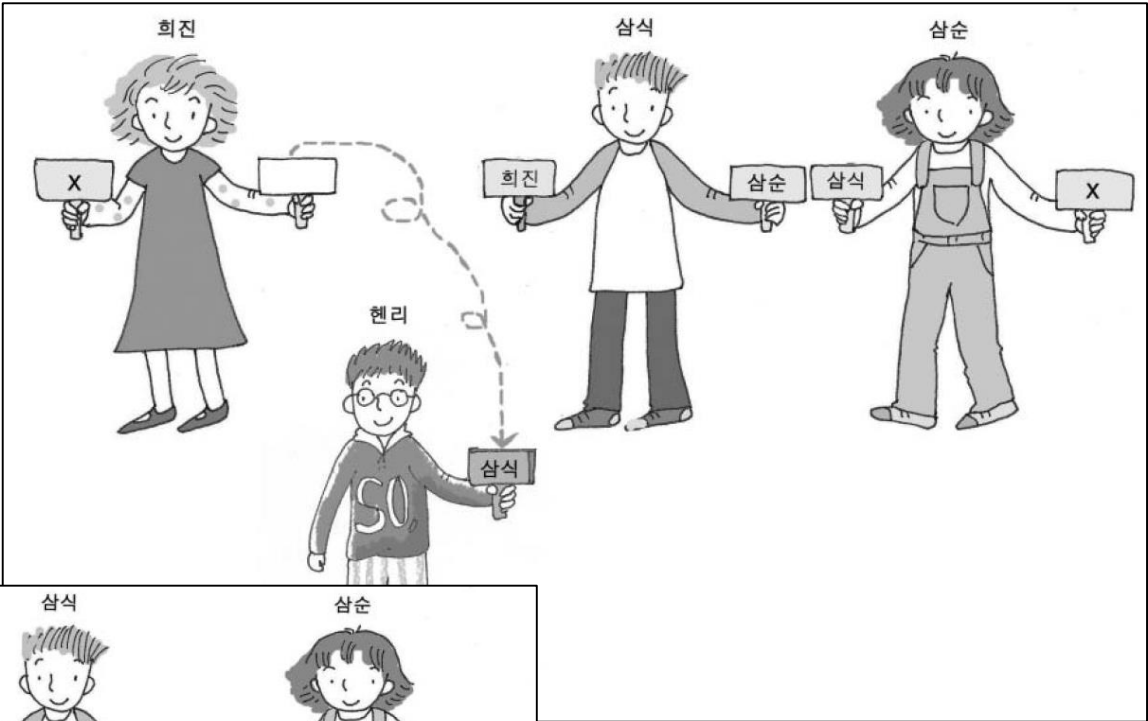
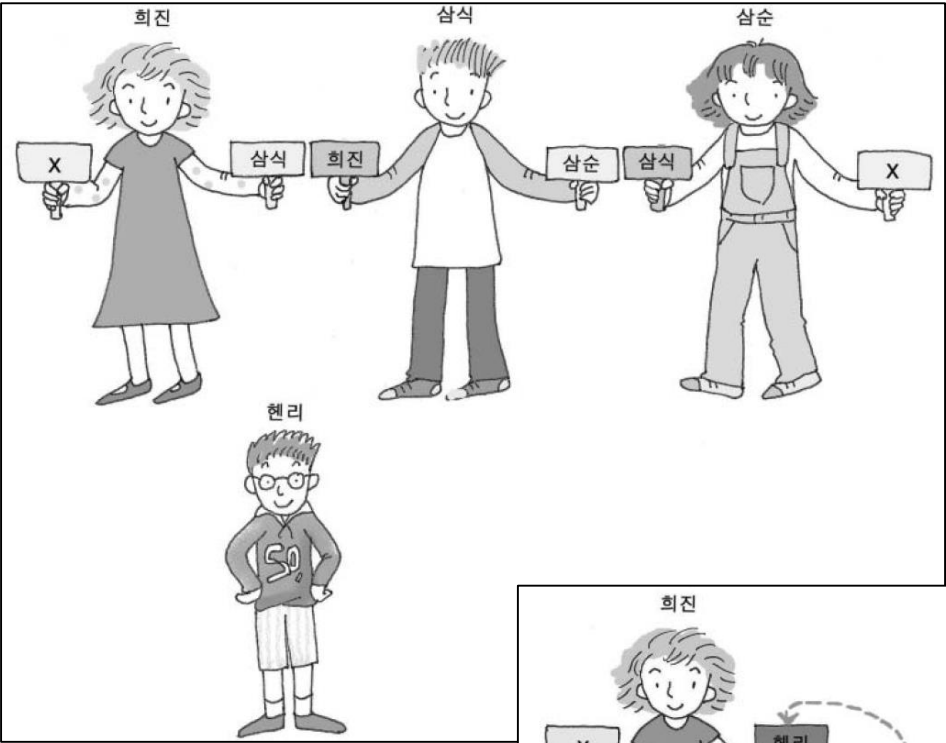
```
01 public class DNode <E> {
02     private E      item;
03     private DNode  previous;
04     private DNode  next;
05     public DNode(E newItem, DNode p, DNode q){ // 노드 생성자
06         item      = newItem;
07         previous  = p;
08         next      = q;
09     }
10     // get 메소드와 set 메소드
11     public E      getItem()      { return item;}
12     public DNode  getPrevious() { return previous;}
13     public DNode  getNext()      { return next;}
14     public void   setItem(E newItem) { item      = newItem;}
15     public void   setPrevious(DNode p) { previous = p;}
16     public void   setNext(DNode q)   { next      = q;}
17 }
```



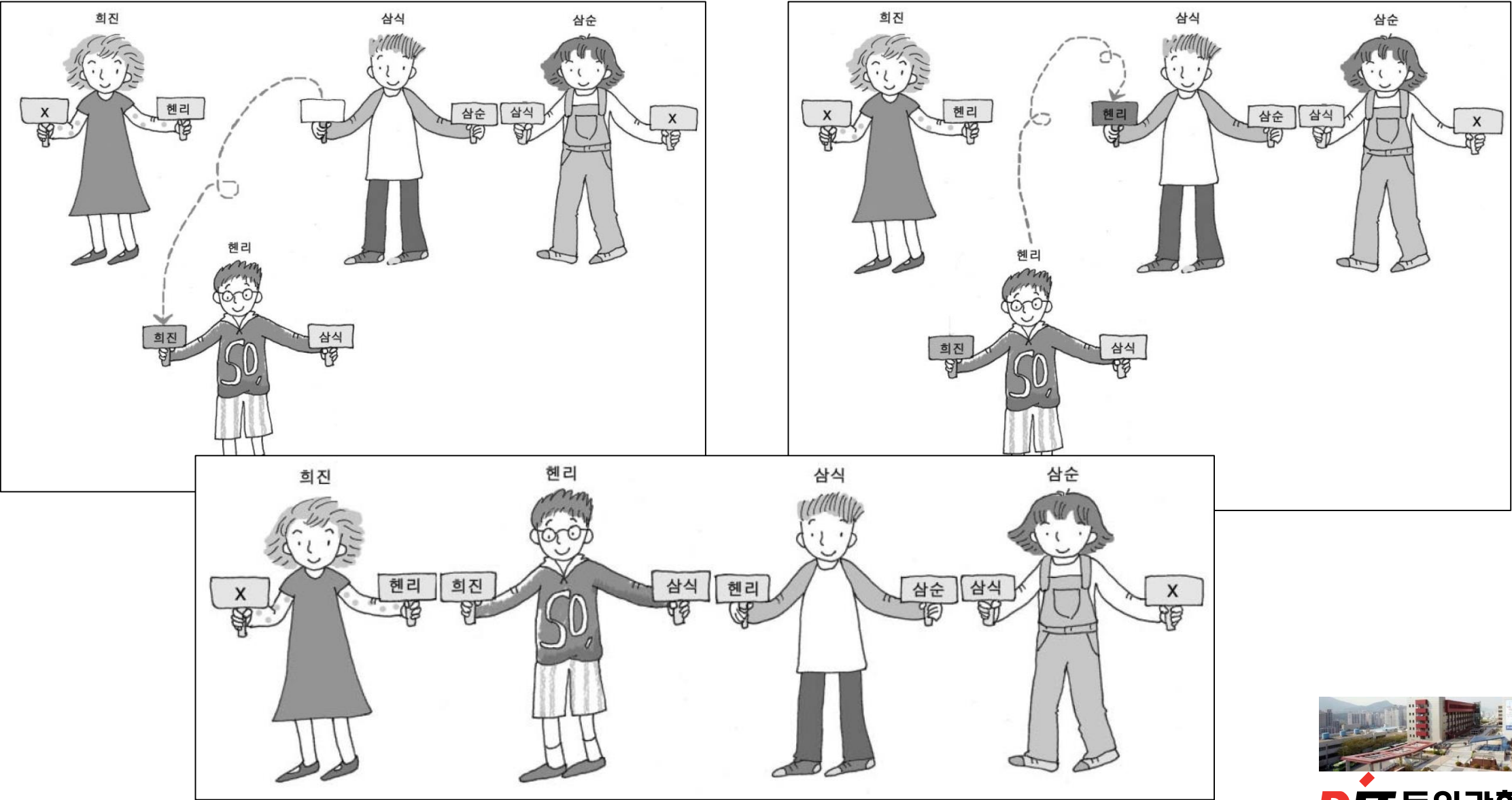
이중 연결 리스트를 위한 DList 클래스

```
01 import java.util.NoSuchElementException;
02 public class DList <E> {
03     protected DNode head, tail;
04     protected int size;
05     public DList(){ //생성자
06         head = new DNode (null, null, null);
07         tail = new DNode (null, head, null); // tail의 이전 노드를 head로 만든다.
08         head.setNext(tail); //head의 다음 노드를 tail로 만든다.
09         size = 0;
10     }
    // 삽입, 삭제 연산을 위한 메소드 선언
}
```

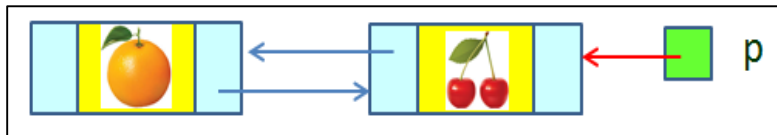
기차놀이에 헨리 끼워주기



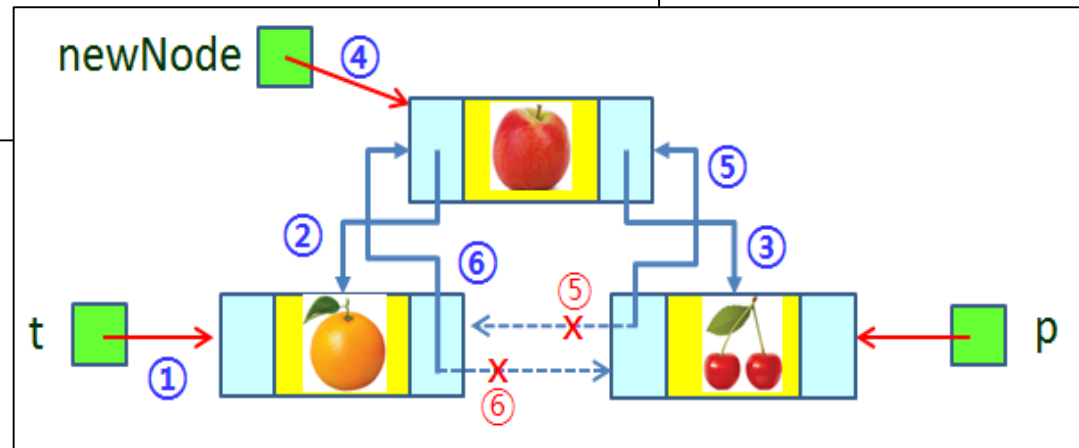
기차놀이에 헨리 끼워주기




```
01 public void insertBefore(DNode p, E newItem){ // p가 가리키는 노드 앞에 삽입
02     ① DNode t = p.getPrevious();
03     DNode newNode = new DNode(newItem, t, p);
04     ⑤ p.setPrevious(newNode);      🍎      ② ③
05     ⑥ t.setNext(newNode);
06     size++;
07 }
```

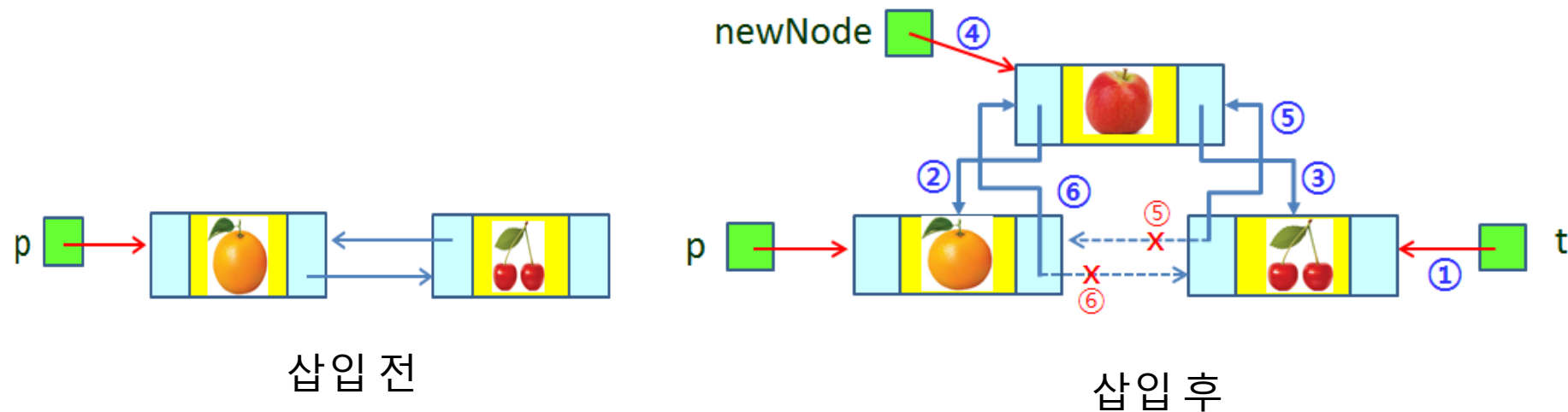


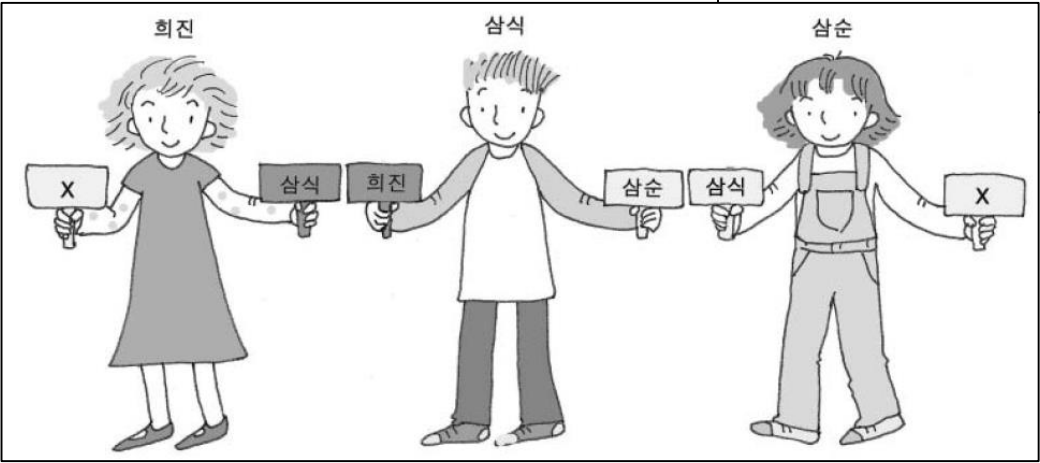
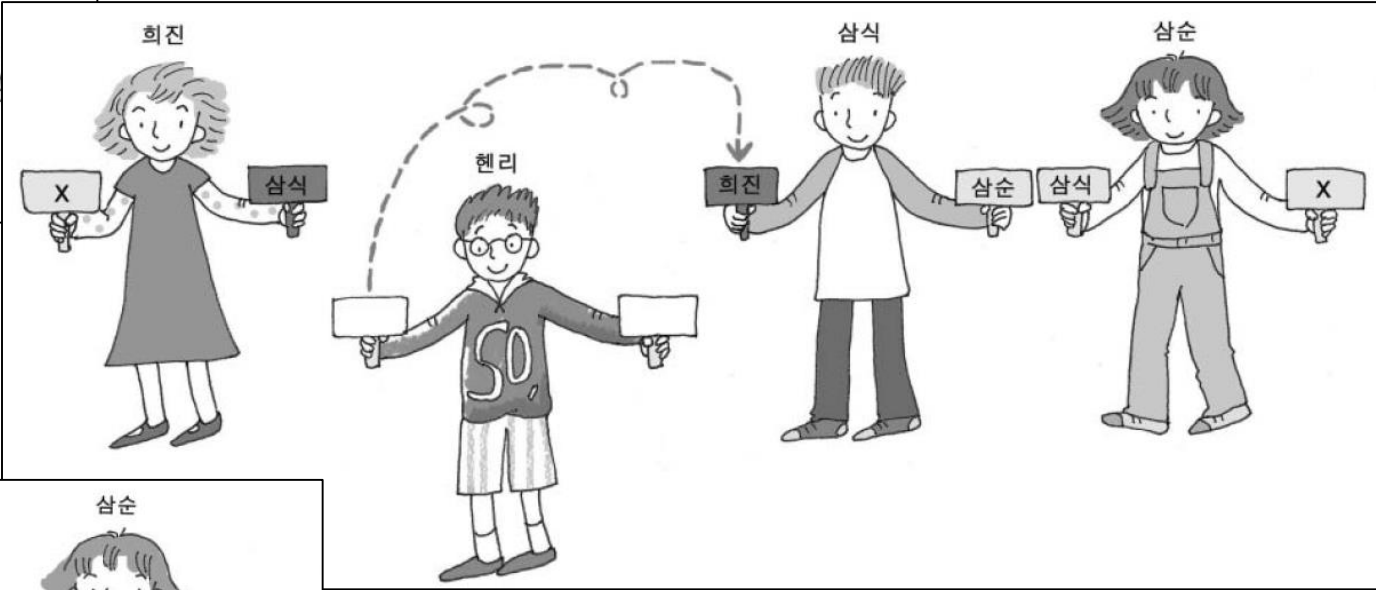
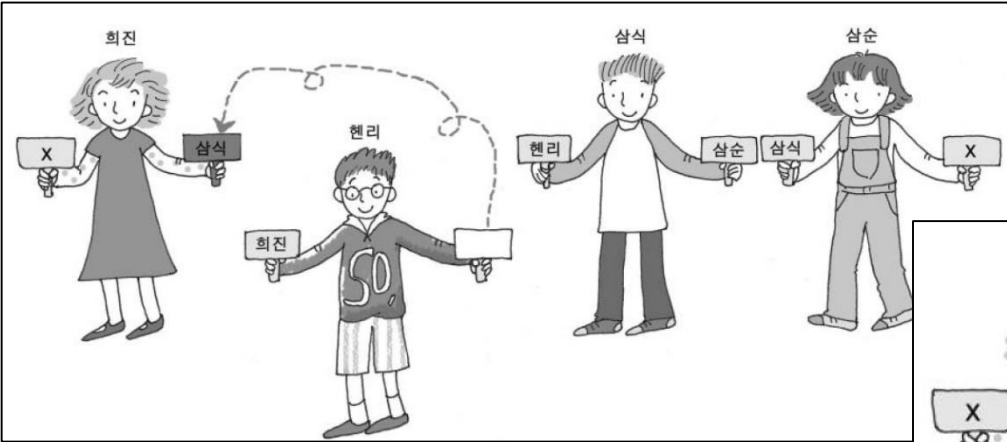
삽입 전



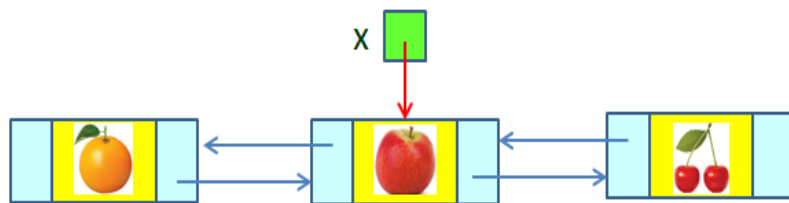
삽입 후

```
01 public void insertAfter(DNode p, E newItem){ // p가 가리키는 노드 뒤에 삽입
02     ① DNode t = p.getNext();
03     DNode newNode = new DNode(newItem, p, t);
04     ⑤ t.setPrevious(newNode);
05     ⑥ p.setNext(newNode);
06     size++;
07 }
```

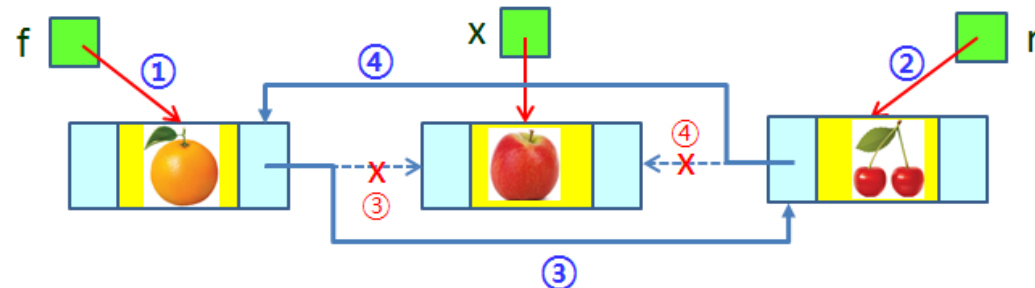




```
01 public void delete(DNode x) { // x가 가리키는 노드 삭제
02     if (x == null) throw new NoSuchElementException();
03     ① DNode f = x.getPrevious();
04     ② DNode r = x.getNext();
05     ③ f.setNext(r);
06     ④ r.setPrevious(f);
07     size--;
08 }
```



삭제 전



삭제 후

```
public class DNode <E> {  
    private E    item;  
    private DNode previous;  
    private DNode next;  
  
    public DNode(E newItem, DNode p, DNode q){ // 노드 생성자  
        item    = newItem;  
        previous = p;  
        next    = q;  
    }  
    // get 메소드와 set 메소드  
    public E    getItem()    { return item;}  
    public DNode getPrevious() { return previous;}  
    public DNode getNext()    { return next;}  
    public void setItem(E newItem) { item    = newItem;}  
    public void setPrevious(DNode p) { previous = p;}  
    public void setNext(DNode q)    { next    = q;}  
}
```

```
import java.util.NoSuchElementException;
public class DList <E> {
    protected DNode head, tail;
    protected int size;

    public DList(){ //생성자
        head = new DNode (null, null, null);
        tail = new DNode (null, head, null); // tail의 이전 노드를 head로 만든다.
        head.setNext(tail); //head의 다음 노드를 tail로 만든다.
        size = 0;
    }

    public int size() { return size; }
    public boolean isEmpty() { return size == 0; }

    public void insertBefore(DNode p, E newItem){ // p가 가리키는 노드 앞에 삽입
        DNode t = p.getPrevious();
        DNode newNode = new DNode(newItem, t, p);
        p.setPrevious(newNode);
        t.setNext(newNode);
        size++;
    }
}
```

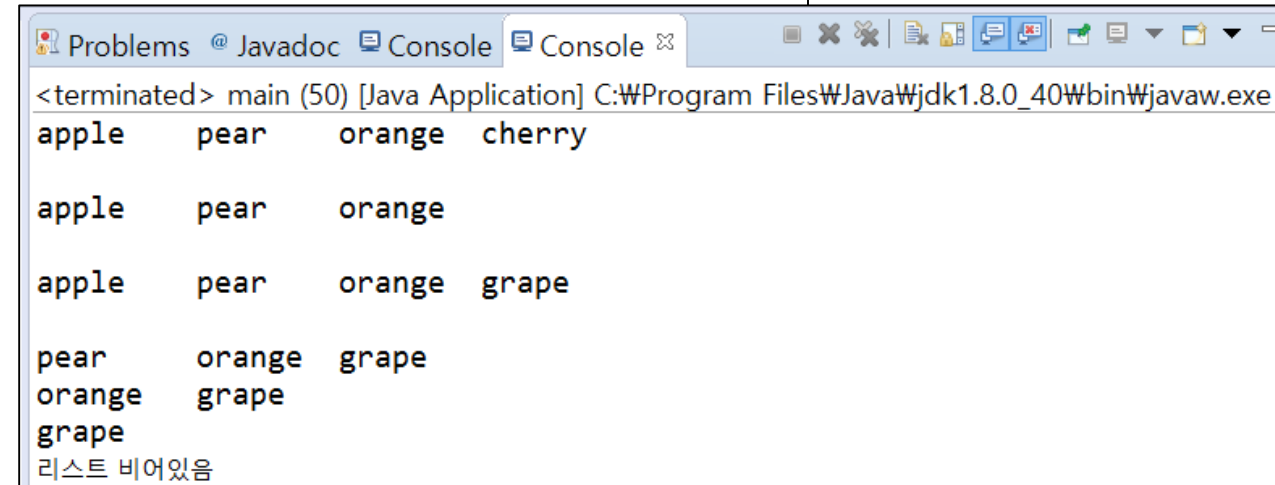
```
public void insertAfter(DNode p, E newItem){ // p가 가리키는 노드 뒤에 삽입
    DNode t = p.getNext();
    DNode newNode = new DNode(newItem, p, t);
    t.setPrevious(newNode);
    p.setNext(newNode);
    size++;
}

public void delete(DNode x) {           // x가 가리키는 노드 삭제
    if (x == null) throw new NoSuchElementException();
    DNode f = x.getPrevious();
    DNode r = x.getNext();
    f.setNext(r);
    r.setPrevious(f);
    x.setPrevious(null); // x의 previous를 null로 만든다.
    x.setNext(null);    // x의 next를 null로 만든다.
    size--;
}
```



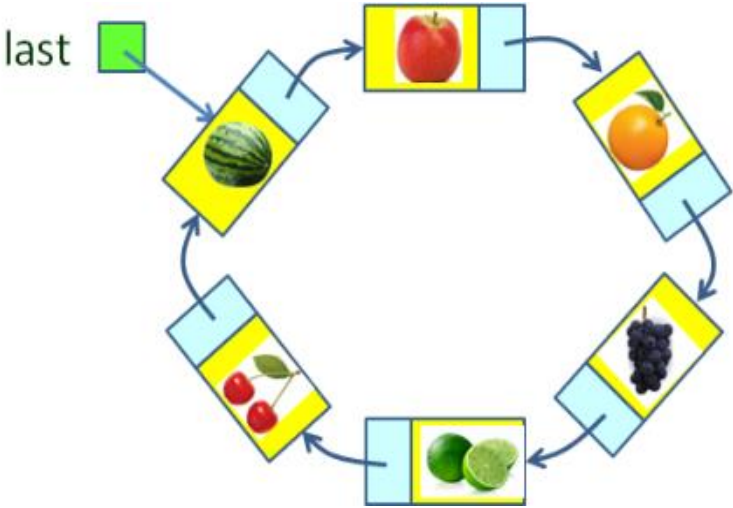
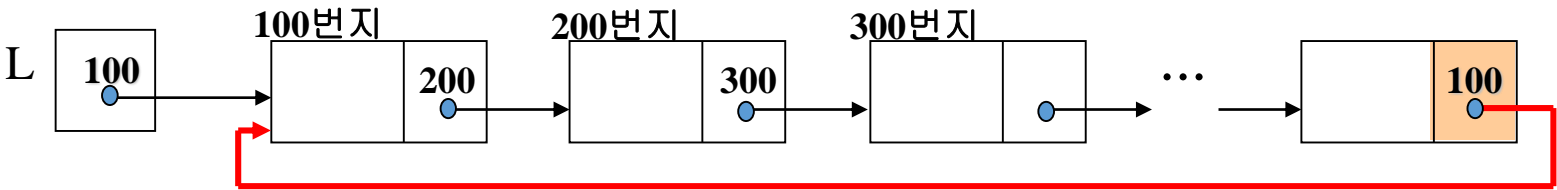
```
public void print(){ // 연결 리스트 노드들의 item들을 차례로 출력
    if (size > 0)
        for (DNode p = head.getNext(); p != tail; p = p.getNext())
            System.out.print(p.getItem()+"\t ");
    else
        System.out.println("리스트 비어있음");
    System.out.println();
}
```

```
public class main {  
    public static void main(String[] args) {  
        DList<String> s = new DList<String>(); // 이중 연결 리스트 객체 s 생성  
  
        s.insertAfter(s.head,"apple");  
        s.insertBefore(s.tail, "orange");  
        s.insertBefore(s.tail,"cherry");  
        s.insertAfter(s.head.getNext(),"pear");  
        s.print(); System.out.println();  
  
        s.delete(s.tail.getPrevious());  
        s.print(); System.out.println();  
  
        s.insertBefore(s.tail,"grape");  
        s.print(); System.out.println();  
        s.delete(s.head.getNext()); s.print();  
        s.delete(s.head.getNext());s.print();  
        s.delete(s.head.getNext()); s.print();  
        s.delete(s.head.getNext());s.print();  
    }  
}
```



```
<terminated> main (50) [Java Application] C:\Program Files\Java\jdk1.8.0_40\bin\javaw.exe  
apple    pear    orange  cherry  
apple    pear    orange  
apple    pear    orange  grape  
pear     orange  grape  
orange   grape  
grape  
리스트 비어있음
```

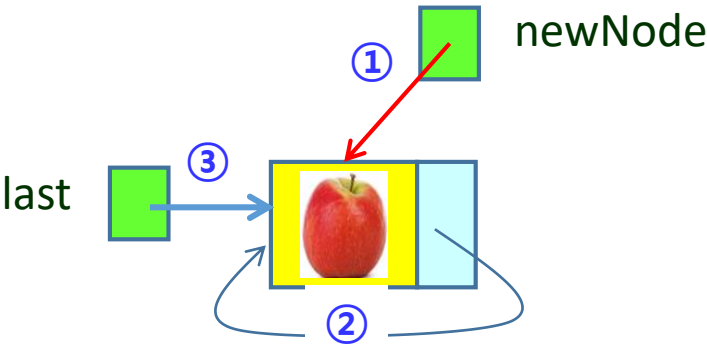
원형 연결 리스트



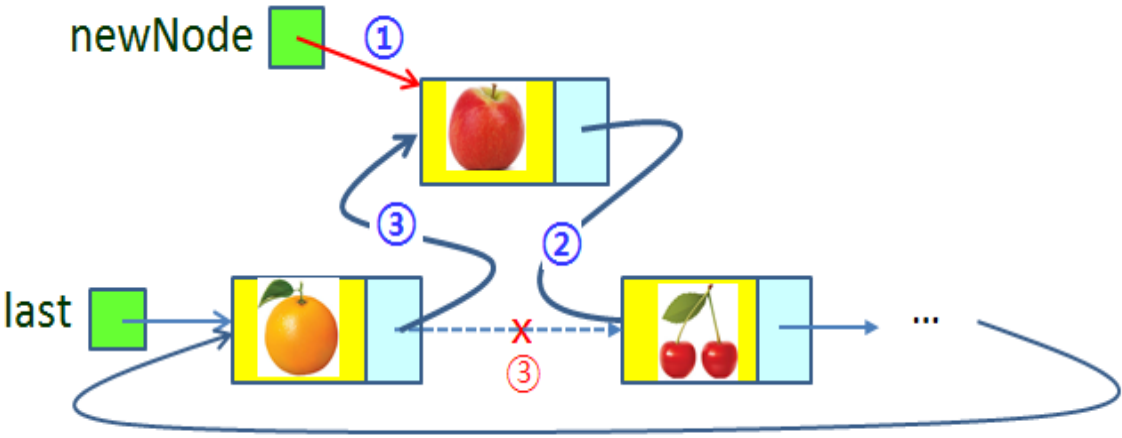
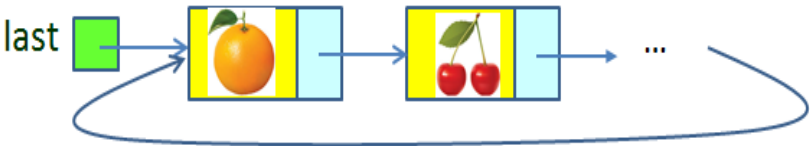
```
01 import java.util.NoSuchElementException;
02 public class CList <E> {
03     private Node last; // 리스트의 마지막 노드(항목)을 가리킨다.
04     private int size; // 리스트의 항목(노드) 수
05     public CList() { // 리스트 생성자
06         last = null;
07         size = 0;
08     }
    // 삽입, 삭제 연산을 위한 메소드 선언
}
```

```
01 public void insert(E newItem) { // last가 가리키는 노드의 다음에 새노드 삽입
02     ① Node newNode = new Node(newItem, null); // 새 노드 생성
03     if (last == null) { // 리스트가 empty일때
04         ② newNode.setNext(newNode);
05         ③ last = newNode;
06     }
07     else {
08         ② newNode.setNext(last.getNext()); // newNode의 다음 노드가 last가 가리키는 노드의 다음노드가 되도록
09         ③ last.setNext(newNode); // last가 가리키는 노드의 다음 노드가 newNode가 되도록
10     }
11     size++;
12 }
```

리스트가 empty인 경우



리스트가 empty가 아닌 경우

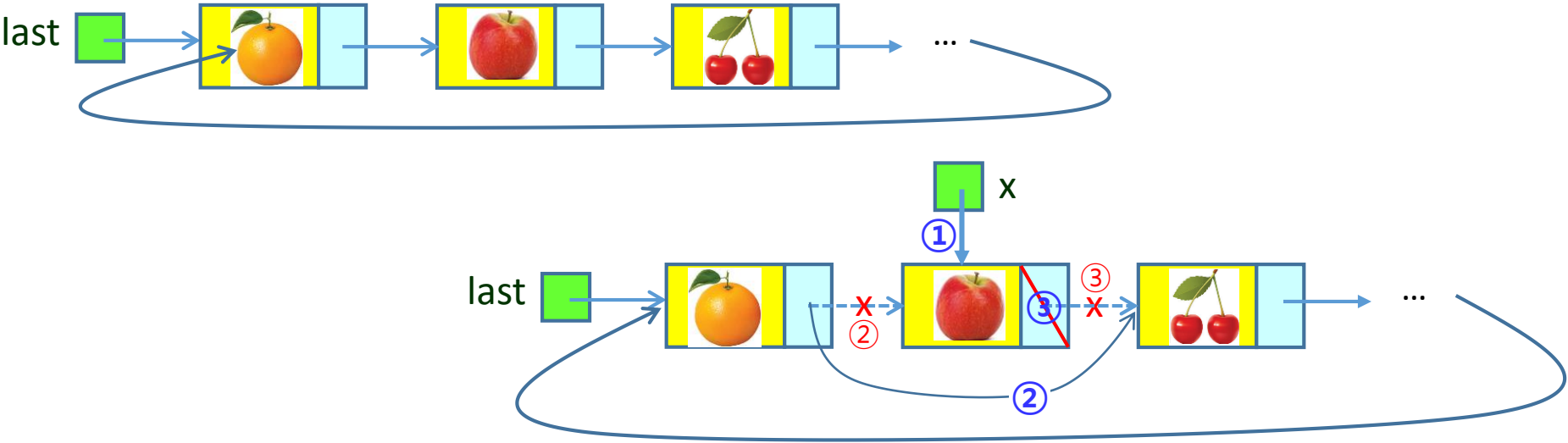


```
01 public Node delete() { // last가 가리키는 노드의 다음 노드를 제거
02     if (isEmpty()) throw new NoSuchElementException();
03     ① Node x = last.getNext(); // x가 리스트의 첫 노드를 가리킴
04     if (x == last) last = null; ② // 리스트에 1개의 노드만 있는 경우
05     else {
06         ② last.setNext(x.getNext()); // last가 가리키는 노드의 다음 노드가 x의 다음 노드가 되도록
07         ③ x.setNext(null); // x의 next를 null로 만든다.
08     }
09     size--;
10     return x;
11 }
```


삭제 후 리스트 empty인 경우



삭제 후 리스트 empty가 안되는 경우



```
public class Node <E>{  
    private E    item;  
    private Node next;  
  
    public Node(E newItem, Node p){ // 생성자  
        item = newItem;  
        next = p;  
    }  
    // get 메소드와 set 메소드  
    public E    getItem()    { return item;}  
    public Node getNext()    { return next;}  
    public void setItem(E newItem)    { item = newItem;}  
    public void setNext(Node newNext)    { next = newNext;}  
}
```

```
import java.util.NoSuchElementException;
public class CList <E> {
    private Node last; // 리스트의 마지막 노드(항목)을 가리킨다.
    private int size; // 리스트의 항목(노드) 수
    public CList() { // 리스트 생성자
        last = null;
        size = 0;
    }
    public int size() { return size;}
    public boolean isEmpty() { return size == 0;}
    public void insert(E newItem) { // last가 가리키는 노드의 다음에 새노드 삽입
        Node newNode = new Node(newItem, null); // 새 노드 생성
        if (last == null) { // 리스트가 empty일때
            newNode.setNext(newNode);
            last = newNode;
        }
        else {
            newNode.setNext(last.getNext()); // newNode의 다음 노드가
            // last가 가리키는 노드의 다음노드가 되도록
            last.setNext(newNode); // last가 가리키는 노드의 다음 노드가 newNode가 되도록
        }
        size++;
    }
}
```

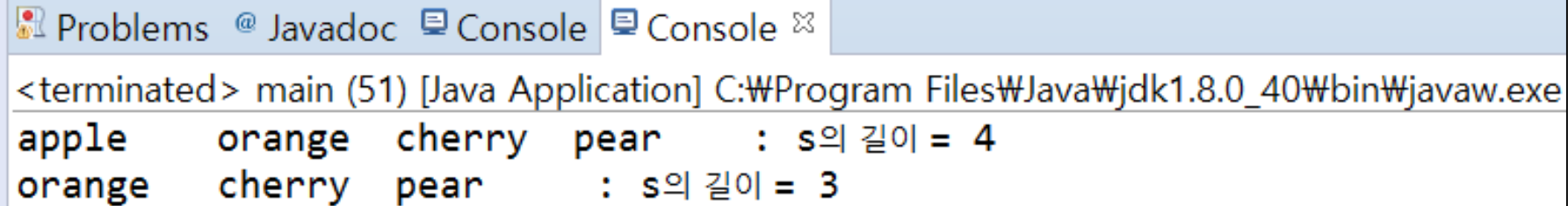


```
public Node delete() { // last가 가리키는 노드의 다음 노드를 제거
    if (isEmpty()) throw new NoSuchElementException();
    Node x = last.getNext();    // x가 리스트의 첫 노드를 가리킴
    if (x == last) last = null; // 리스트에 1개의 노드만 있는 경우
    else {
        last.setNext(x.getNext()); // last가 가리키는 노드의 다음 노드가 x의 다음 노드가 되도록
        x.setNext(null);           // x의 next를 null로 만든다.
    }
    size--;
    return x;
}

public void print(){ // 연결 리스트 노드들의 항목들을 차례로 출력
    if (size > 0){
        int i = 0;
        for (Node p = last.getNext(); i < size ; p = p.getNext(), i++)
            System.out.print(p.getItem()+"\t ");
    }
    else
        System.out.println("리스트 비어있음.");
}
}
```



```
public class main {  
    public static void main(String[] args) {  
        CList<String> s = new CList<String>(); // 연결 리스트 객체 s 생성  
  
        s.insert("pear");    s.insert("cherry");  
        s.insert("orange"); s.insert("apple");  
        s.print();  
        System.out.print(": s의 길이 = "+s.size()+"\n");  
  
        s.delete();  
        s.print();  
        System.out.print(" : s의 길이 = "+s.size());System.out.println();  
  
    }  
}
```



Problems @ Javadoc Console Console

<terminated> main (51) [Java Application] C:\Program Files\Java\jdk1.8.0_40\bin\javaw.exe

apple orange cherry pear : s의 길이 = 4

orange cherry pear : s의 길이 = 3

Report

이중 연결 리스트와 원형 연결 리스트를 테스트하고 그 소스를 Github에 업로드 하세요!

Reference

- 자바로 배우는 쉬운 자료구조, 이지영, 한빛아카데미
- 자바와 함께하는 자료구조의 이해, 양성봉, 생능출판

언제 어디서나 즐공, 열공, 진공하세요.

감사합니다
