

13장. 자료구조로 사용되는 자바 클래스들



학습 목표

- 자료구조란?
- 자료구조 클래스의 사용 방법

01. 자료 구조란?

자료구조란?

- 자료구조(data structure)

데이터를 효율적으로 사용할 수 있도록 구조를 만들어서 저장해둔 것

- 자료구조의 종류

- 리스트(list) : 배열 리스트(array list), 연결 리스트(linked list)로 세분됨
- 스택(stack)
- 큐(queue)
- 해쉬 테이블(hashtable)
- 집합(set) * 엄밀히 말하면 자료구조가 아님

01. 자료구조란?

자료구조 클래스

- JDK 라이브러리의 자료구조 클래스들

자료구조	클래스 이름
리스트	ArrayList LinkedList (Vector)
스택	LinkedList (Stack)
큐	LinkedList
해쉬 테이블	HashMap (Hashtable)
집합	HashSet

- 위 클래스들은 모두 java.util 패키지에 속함
- Vector, Stack, Hashtable 클래스는 사용이 권장되지 않음. 구버전부터 있던 클래스

02. 자료구조 클래스의 사용 방법

자료구조 클래스의 사용 방법

- 자료구조 클래스의 API 규격서

java.util (Java 2 Platform SE 5.0) - Microsoft Internet Explorer

주소(D) http://java.sun.com/j2se/1.5.0/docs/api/index.html

java.util

Interfaces

- Collection
- Comparator
- Enumeration
- EventListener
- Formattable
- Iterator
- List
- ListIterator
- Map
- Map.Entry
- Observer
- Queue
- RandomAccess
- Set
- SortedMap
- SortedSet

Classes

- AbstractSequentialList
- AbstractSet
- ArrayList
- Arrays
- BitSet
- Calendar
- Collections
- Currency
- Date

ArrayList<E>

Resizable-array implementation of the List interface.

Arrays

This class contains various methods for manipulating arrays (such as sorting and searching).

BitSet

This class implements a vector of bits that grows as needed.

Calendar

The Calendar class is an abstract class that provides methods for converting between a specific instant in time and a set of [calendar fields](#) such as YEAR, MONTH, DAY_OF_MONTH, HOUR, and so on, and for manipulating the calendar fields, such as getting the date of the next week.

Collections

This class consists exclusively of static methods that operate on or return collections.

Currency

Represents a currency.

Date

The class Date represents a specific instant in time, with millisecond precision.

Dictionary<K, V>

The Dictionary class is the abstract parent of any class, such as Hashtable, which maps keys to values.

EnumMap<K extends Enum<K>, V>

A specialized [Map](#) implementation for use with enum type keys.

EnumSet<E extends Enum<E>>

A specialized [Set](#) implementation for use with enum types.

EventListenerProxy

An abstract wrapper class for an EventListener class which associates a set of additional parameters with the listener.

EventObject

The root class from which all event state objects shall be derived.

FormattableFlags

FormattableFlags are passed to the [Formattable.formatTo\(\)](#) method and modify the output format for [Formattables](#).

Formatter

An interpreter for printf-style format strings.

GregorianCalendar

GregorianCalendar is a concrete subclass of Calendar and provides the standard calendar system used by most of the world.

HashMap<K, V>

Hash table based implementation of the Map interface.

HashSet<E>

Set implementation, backed by a hash table (actually a HashMap).

Hashtable<K, V>

Which maps keys to values.

IdentityHashMap<K, V>

This class implements the Map interface with a hash table in place of object-equality when comparing keys.

LinkedHashMap<K, V>

Hash table and linked list implementation of the Map interface, with predictable iteration order.

LinkedHashSet<E>

Hash table and linked list implementation of the Set interface, with predictable iteration order.

LinkedList<E>

Linked list implementation of the List interface.

해시 테이블을 구현하는 클래스

집합을 구현하는 클래스

리스트를 구현하는 클래스

02. 자료구조 클래스의 사용 방법

자료구조 클래스의 사용 방법

- 자료구조 클래스의 사용 방법

```
ArrayList<String> list = new ArrayList<String>();
```

String 객체를 담을 수 있는 ArrayList 객체를 생성합니다.
ArrayList 객체를 대입할 변수도 이렇게 선언해야 합니다.

타입 파라미터(type parameter)

02. 자료구조 클래스의 사용 방법

자료구조 클래스의 사용 방법

- 타입 파라미터에 의해 저장 데이터의 타입이 제한됨

```
list.add("포도");
```

String 타입의 데이터를
추가하는 것은 가능합니다.

```
list.add(new Integer(52));
```

Integer 타입의 데이터를
추가하는 것은 불가능합니다.

02. 자료구조 클래스의 사용 방법

리스트로 사용할 수 있는 클래스

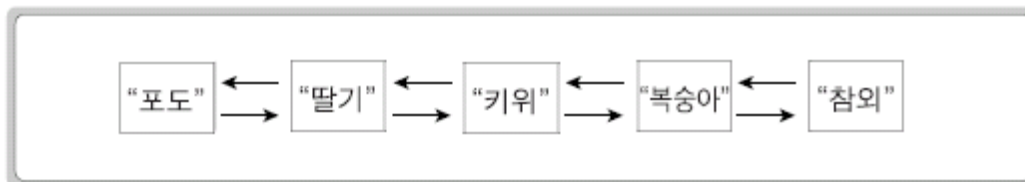
- 리스트(list) : 데이터를 일렬로 늘어놓은 자료구조
- 리스트로 사용할 수 있는 클래스 : ArrayList 클래스와 LinkedList 클래스

ArrayList 객체



ArrayList 클래스는 내부에 있는 배열에 데이터를 저장합니다.

LinkedList 객체



LinkedList 클래스는 인접 데이터가 서로 가리키는 식으로 데이터를 저장합니다.

02. 자료구조 클래스의 사용 방법

리스트 : ArrayList 클래스

- 사용 방법

- 1) 먼저 리스트에 저장할 데이터의 타입을 정해야 합니다.
- 2) 그 타입은 타입 파라미터로 삼아서 ArrayList 객체를 생성합니다.
- 3) ArrayList 객체에 데이터를 저장합니다.

[주의] 레퍼런스 타입만
타입 파라미터가 될 수 있음

02. 자료구조 클래스의 사용 방법

리스트 : ArrayList 클래스

- 리스트 생성 방법

```
ArrayList<String> list = new ArrayList<String>();
```

타입 파라미터

타입 파라미터

ArrayList 객체 안에 String 객체 10개를
담을 수 있는 배열이 생성됩니다.

ArrayList 객체



02. 자료구조 클래스의 사용 방법

리스트 : ArrayList 클래스

- 데이터 추가 방법

```
list.add("포도");  
list.add("딸기");  
list.add("복숭아");
```

타입 파라미터와 맞는
데이터 값을 넘겨주어야 합니다.

add 메소드 호출 순서대로
데이터가 저장됩니다.

ArrayList 객체

"포도"	"딸기"	"복숭아"							
------	------	-------	--	--	--	--	--	--	--

02. 자료구조 클래스의 사용 방법

리스트 : ArrayList 클래스

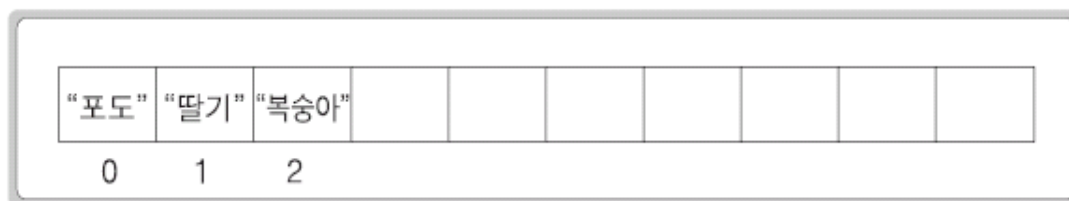
- 데이터를 가져오는 방법

```
String str = list.get(2);
```



인덱스 2 위치에 있는
"복숭아"를 리턴합니다.

ArrayList 객체



02. 자료구조 클래스의 사용 방법

리스트 : ArrayList 클래스

- 데이터의 수를 가져오는 방법

```
int num = list.size();
```



리스트에 있는 데이터의 수
3을 리턴합니다.

ArrayList 객체

“포도”	“딸기”	“복숭아”							
------	------	-------	--	--	--	--	--	--	--

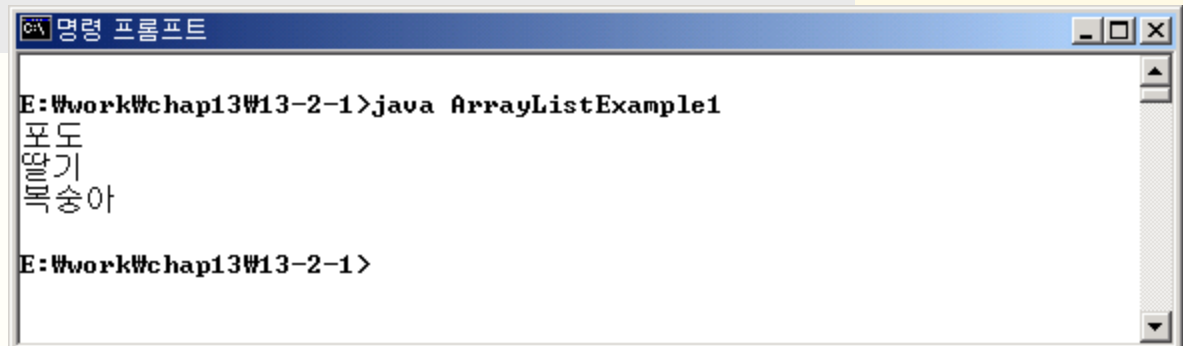
02. 자료구조 클래스의 사용 방법

리스트 : ArrayList 클래스

[예제 13-1] ArrayList 클래스의 사용 예

```
1  import java.util.*;
2  class ArrayListExample1 {
3      public static void main(String args[]) {
4          ArrayList<String> list = new ArrayList<String>();  ----- ArrayList 객체를 생성합니다.
5          list.add("포도");
6          list.add("딸기");
7          list.add("복숭아"); }  리스트에 3개의 데이터를 추가합니다.
8          int num = list.size();
9          for (int cnt = 0; cnt < num; cnt++) {
10             String str = list.get(cnt);
11             System.out.println(str);
12         }
13     }
14 }
```

리스트에 있는 데이터의 수만큼
루프를 돌면서 데이터를 읽어와서
출력합니다.

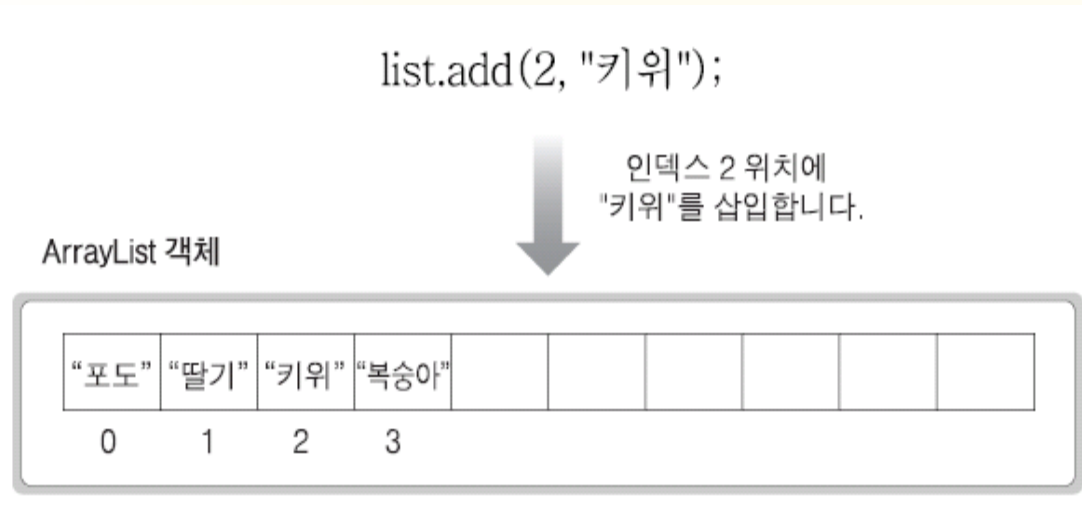


```
명령 프롬프트
E:\work\chap13\13-2-1>java ArrayListExample1
포도
딸기
복숭아
E:\work\chap13\13-2-1>
```

02. 자료구조 클래스의 사용 방법

리스트 : ArrayList 클래스

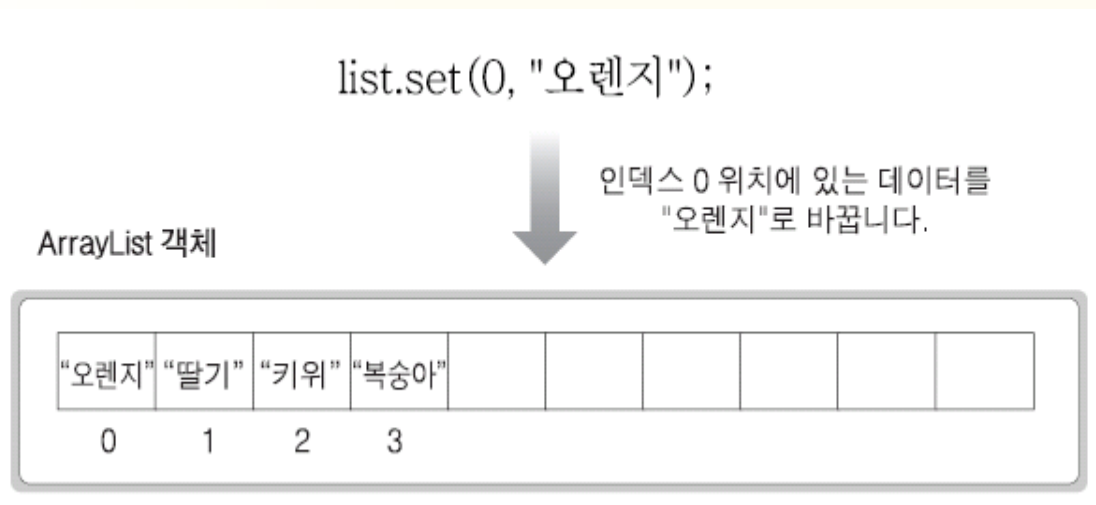
- 데이터를 중간에 삽입하는 방법



02. 자료구조 클래스의 사용 방법

리스트 : ArrayList 클래스

- 기존 데이터를 교체하는 방법.



02. 자료구조 클래스의 사용 방법

리스트 : ArrayList 클래스

- 데이터를 삭제하는 방법 (1)



02. 자료구조 클래스의 사용 방법

리스트 : ArrayList 클래스

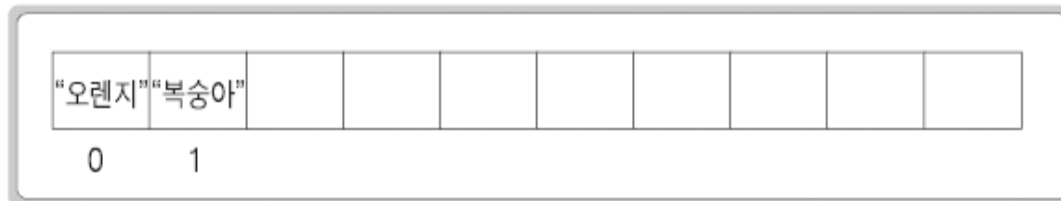
- 데이터를 삭제하는 방법 (2)

```
list.remove("키위");
```



리스트에서 "키위"라는
데이터를 찾아서 삭제합니다.

LinkedList 객체



02. 자료구조 클래스의 사용 방법

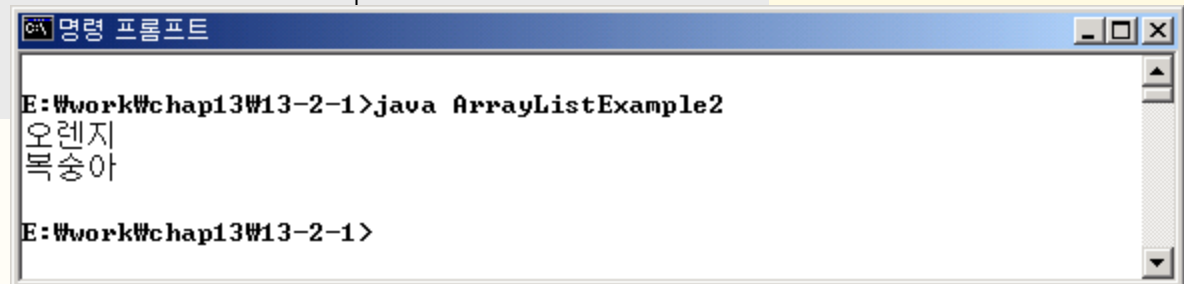
리스트 : ArrayList 클래스

[예제 13-2] ArrayList에 데이터를 삽입/수정/삭제하는 예

```
1  import java.util.*;
2  class ArrayListExample2 {
3      public static void main(String args[]) {
4          ArrayList<String> list = new ArrayList<String>();
5          list.add("포도");
6          list.add("딸기");
7          list.add("복숭아");
8          list.add(2, "키위");
9          list.set(0, "오렌지");
10         list.remove(1);
11         list.remove("키위");
12         int num = list.size();
13         for (int cnt = 0; cnt < num; cnt++) {
14             String str = list.get(cnt);
15             System.out.println(str);
16         }
17     }
18 }
```

add, set, remove 메소드를 이용하여
리스트에 데이터를 삽입/수정/삭제합니다.

리스트의 데이터를 순서대로
가져와서 출력합니다.



```
E:\work\chap13\13-2-1>java ArrayListExample2
오렌지
복숭아
키위

E:\work\chap13\13-2-1>
```

02. 자료구조 클래스의 사용 방법

리스트 : ArrayList 클래스

- 데이터를 검색하는 방법

```
int index = list.indexOf("사과");
```

ArrayList 객체



첫번째 "사과"의
위치 1을 리턴합니다.

02. 자료구조 클래스의 사용 방법

리스트 : ArrayList 클래스

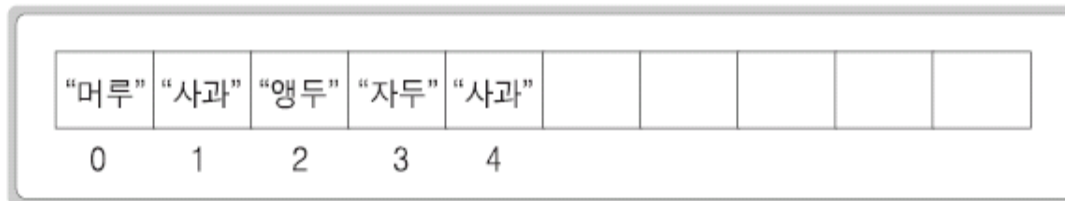
- 데이터를 뒤에서부터 검색하는 방법

```
int index = list.lastIndexOf("사과");
```



마지막 "사과"의
위치 4를 리턴합니다.

ArrayList 객체



02. 자료구조 클래스의 사용 방법

리스트 : ArrayList 클래스

[예제 13-3] ArrayList의 데이터를 검색하는 예

```
1  import java.util.*;
2  class ArrayListExample3 {
3      public static void main(String args[]) {
4          ArrayList<String> list = new ArrayList<String>();
5          list.add("머루");
6          list.add("사과");
7          list.add("앵두");
8          list.add("자두");
9          list.add("사과");
10         int index1 = list.indexOf("사과");
11         int index2 = list.lastIndexOf("사과");
12         System.out.println("첫번째 사과: " + index1);
13         System.out.println("마지막 사과: " + index2);
14     }
15 }
```

리스트에 있는 첫번째 “사과”와 마지막 “사과”의 인덱스를 가져와서 출력합니다.



```
명령 프롬프트
E:\work\chap13\13-2-1>java ArrayListExample3
첫번째 사과: 1
마지막 사과: 4
E:\work\chap13\13-2-1>
```

02. 자료구조 클래스의 사용 방법

리스트 : LinkedList 클래스

- 기본적인 사용 방법
 - * ArrayList 클래스와 동일합니다.

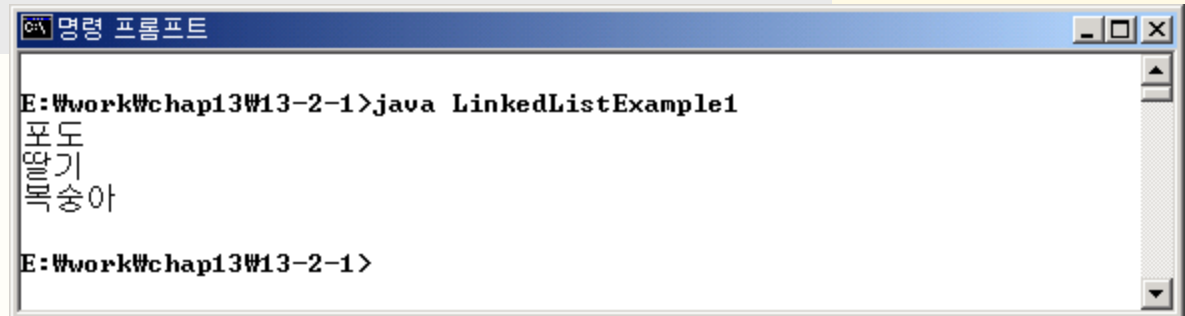
02. 자료구조 클래스의 사용 방법

리스트 : LinkedList 클래스

[예제 13-4] LinkedList의 사용 예

```
1  import java.util.*;
2  class LinkedListExample1 {
3      public static void main(String args[]) {
4          LinkedList<String> list = new LinkedList<String>(); ----- LinkedList 객체를 생성합니다.
5          list.add("포도");
6          list.add("딸기");
7          list.add("복숭아");
8          int num = list.size();
9          for (int cnt = 0; cnt < num; cnt++) {
10             String str = list.get(cnt);
11             System.out.println(str);
12         }
13     }
14 }
```

[예제 13-1]과 동일합니다.



```
명령 프롬프트
E:\work\chap13\13-2-1>java LinkedListExample1
포도
딸기
복숭아
E:\work\chap13\13-2-1>
```

02. 자료구조 클래스의 사용 방법

리스트 : LinkedList 클래스

- LinkedList 객체를 생성할 때 일어나는 일

```
LinkedList<String> list = new LinkedList<String>();
```

LinkedList 객체



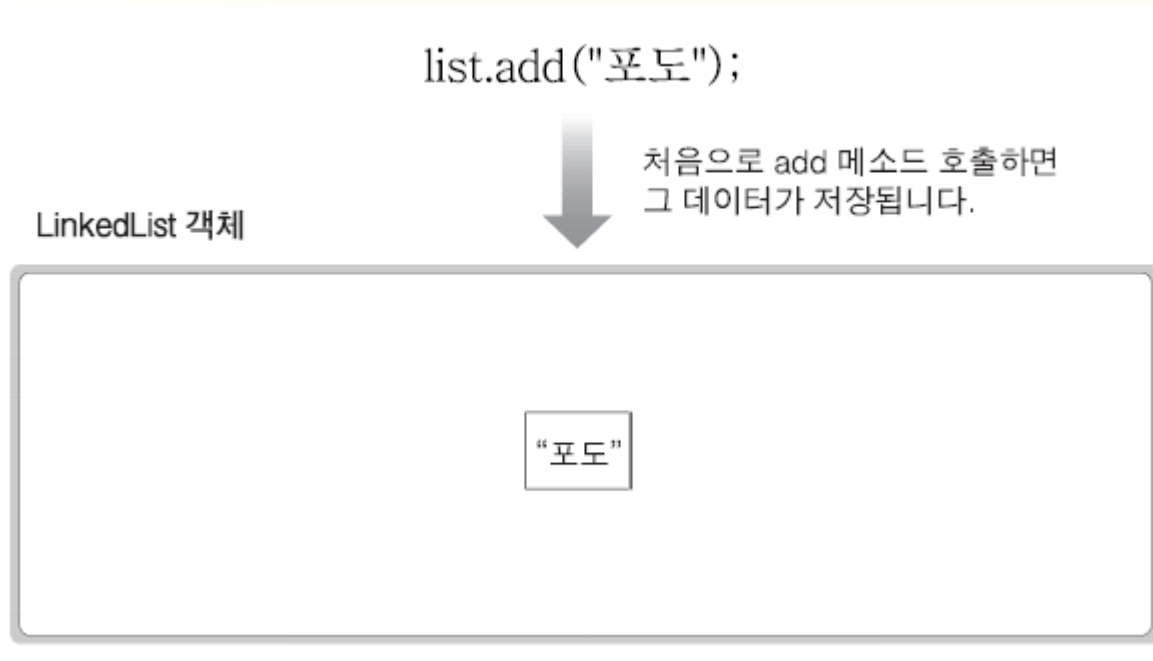
LinkedList 객체를 생성한다고 해서
데이터 저장 영역이 생기지는 않습니다.



02. 자료구조 클래스의 사용 방법

리스트 : LinkedList 클래스

- add 메소드를 처음으로 호출할 때 일어나는 일



02. 자료구조 클래스의 사용 방법

리스트 : LinkedList 클래스

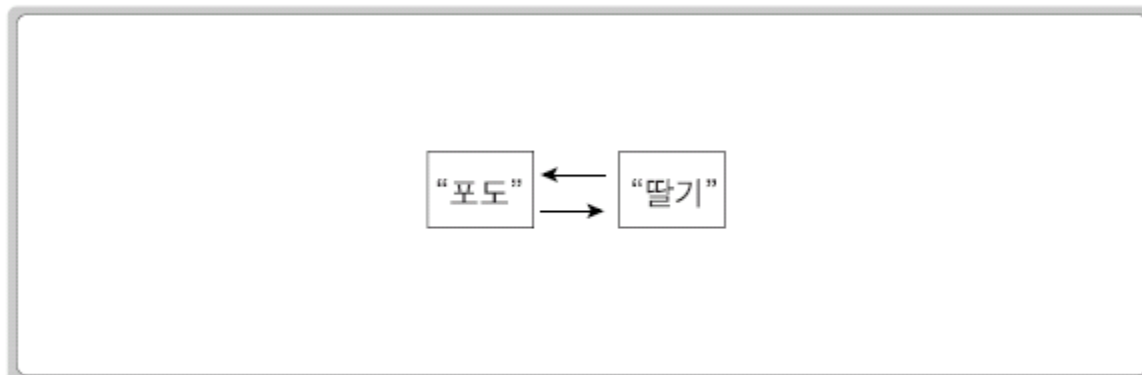
- add 메소드를 두번째로 호출할 때 일어나는 일

```
list.add("딸기");
```



add 메소드를 또 호출하면 마지막 데이터와 서로 가리키는 식으로 저장됩니다.

LinkedList 객체



02. 자료구조 클래스의 사용 방법

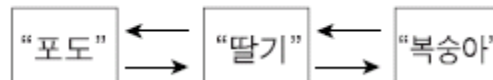
리스트 : LinkedList 클래스

- add 메소드를 세번째로 호출할 때 일어나는 일

```
list.add("복숭아");
```

add 메소드를 또 호출하면 마지막 데이터와 서로 가리키는 식으로 저장됩니다.

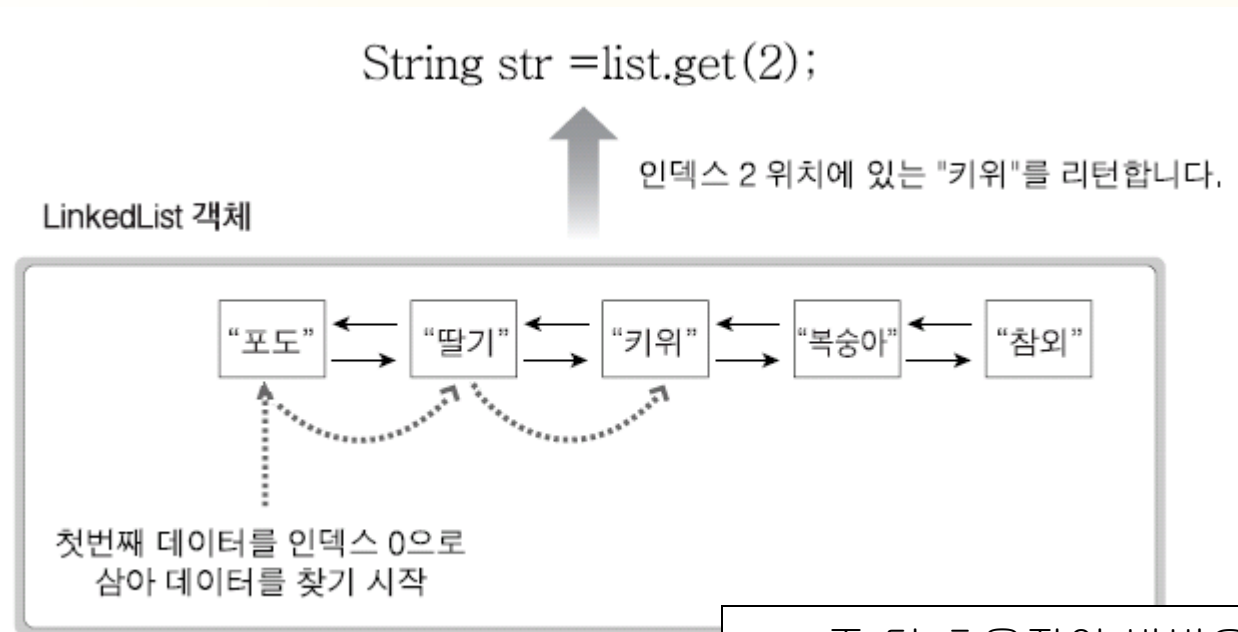
LinkedList 객체



02. 자료구조 클래스의 사용 방법

리스트 : LinkedList 클래스

- get 메소드를 호출할 때 일어나는 일



좀 더 효율적인 방법은
없을까요?


02. 자료구조 클래스의 사용 방법

리스트 : LinkedList 클래스

[예제 13-5] LinkedList에 데이터를 삽입/수정/삭제하는 예

```
1  import java.util.*;
2  class LinkedListExample2 {
3      public static void main(String args[]) {
4          LinkedList<String> list = new LinkedList<String>(); ----- LinkedList 객체를 생성합니다.
5          list.add("포도");
6          list.add("딸기");
7          list.add("복숭아");
8          list.add(2, "키위");
9          list.set(0, "오렌지");
10         list.remove(1);
11         list.remove("키위");
12         int num = list.size();
13         for (int cnt = 0; cnt < num; cnt++) {
14             String str = list.get(cnt);
15             System.out.println(str);
16         }
17     }
18 }
```

[예제 13-2]와 동일합니다.

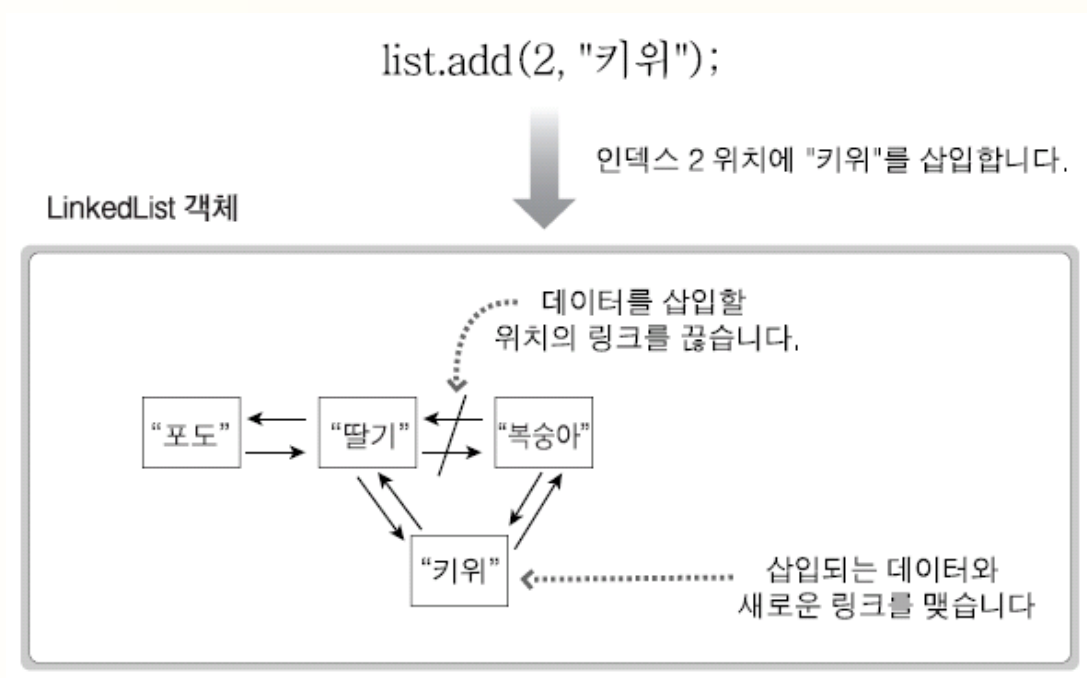


```
명령 프롬프트
E:\work\chap13\13-2-1>java LinkedListExample2
오렌지
복숭아
키위
E:\work\chap13\13-2-1>
```

02. 자료구조 클래스의 사용 방법

리스트 : LinkedList 클래스

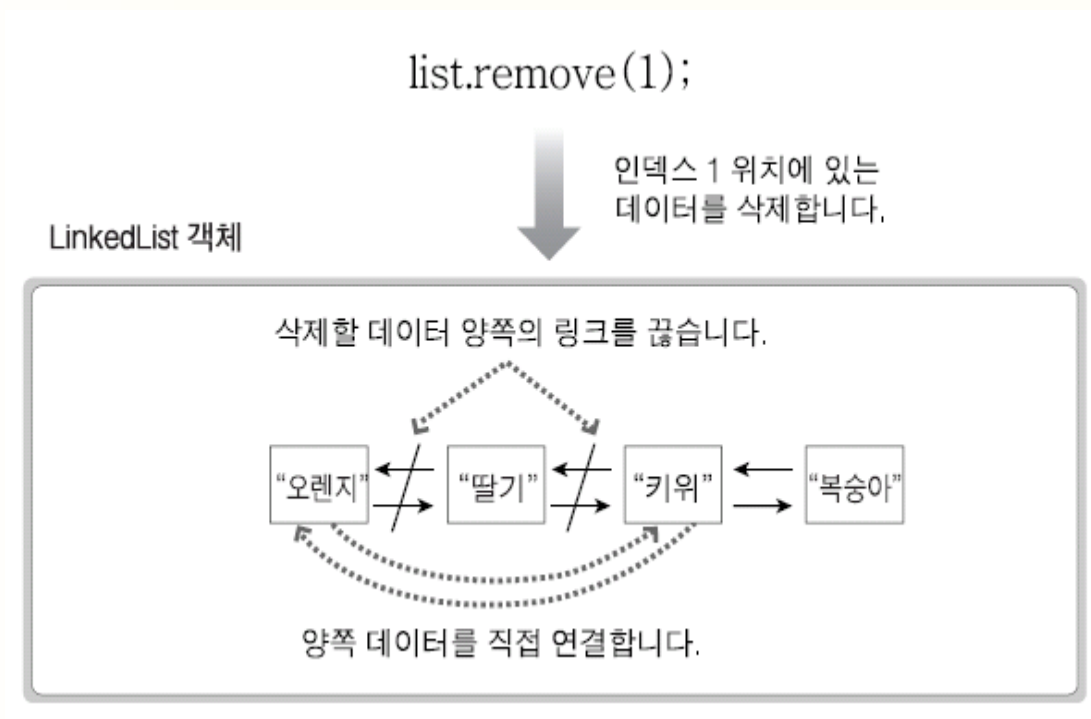
- 데이터를 중간에 삽입할 때 일어나는 일



02. 자료구조 클래스의 사용 방법

리스트 : LinkedList 클래스

- 데이터를 중간에서 삭제할 때 일어나는 일



02. 자료구조 클래스의 사용 방법

리스트 : LinkedList 클래스

- 데이터 순차 접근을 효율적으로 하는 방법

1) iterator 메소드를 호출합니다.

```
Iterator<String> iterator = list.iterator();
```

타입 파라미터

Iterator 객체를 리턴하는 메소드

2) Iterator 객체에 대해 next 메소드를 호출합니다.

```
String str = iterator.next();
```

Iterator를 통해 데이터를 읽어오는 메소드

next 메소드는 더 이상 데이터가 없으면
NoSuchElementException을 발생합니다.

02. 자료구조 클래스의 사용 방법

리스트 : LinkedList 클래스

- NoSuchElementException의 발생을 막는 방법

```
while (iterator.hasNext()) {  
    String str = iterator.next();  
      
}  
}
```

hasNext 메소드의 리턴 값이 true일 동안
next 메소드를 반복 호출합니다.
next 메소드로 가져온
데이터를 처리하는 부분

02. 자료구조 클래스의 사용 방법

리스트 : LinkedList 클래스

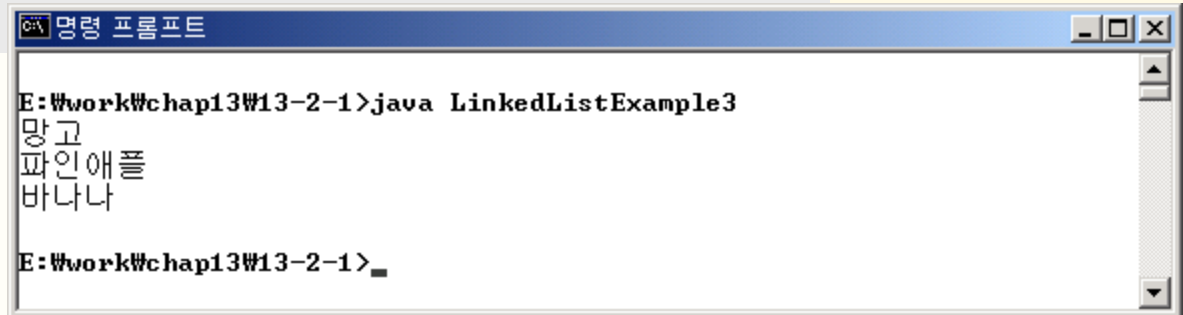
[예제 13-6] Iterator의 사용 예

```
1  import java.util.*;
2  class LinkedListExample3 {
3      public static void main(String args[]) {
4          LinkedList<String> list = new LinkedList<String>();
5          list.add("망고");
6          list.add("파인애플");
7          list.add("바나나");
8          Iterator<String> iterator = list.iterator();
9          while (iterator.hasNext()) {
10             String str = iterator.next();
11             System.out.println(str);
12         }
13     }
14 }
```

LinkedList 객체를 생성하여
3개의 데이터를 저장합니다.

iterator 메소드를 호출하여
Iterator 객체를 얻습니다.

Iterator 객체를 이용하여 리스트에 있는
데이터를 순서대로 가져와서 출력합니다.



```
명령 프롬프트
E:\work\chap13\13-2-1>java LinkedListExample3
망고
파인애플
바나나
E:\work\chap13\13-2-1>
```

02. 자료구조 클래스의 사용 방법

리스트 : LinkedList 클래스

- 향상된 for 문으로 리스트를 사용하는 방법

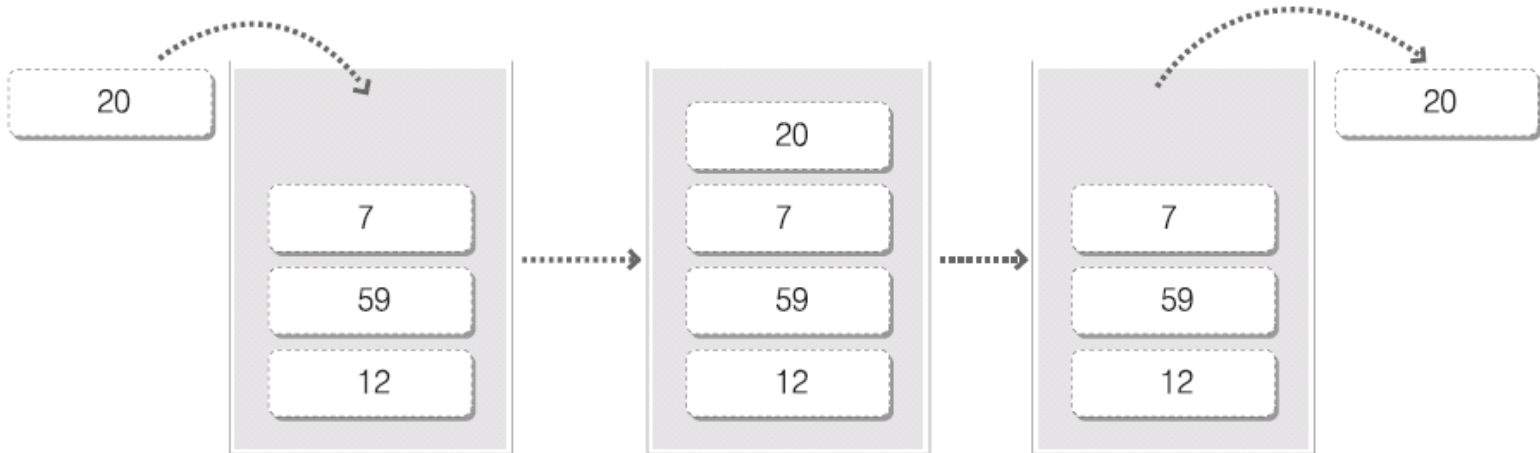


이런 형식의 for 문에서는 리스트로부터 Iterator 객체가 자동으로 얻어지고, 그 Iterator 객체를 이용하여 얻은 데이터가 str 변수에 자동으로 대입됩니다.

02. 자료구조 클래스의 사용 방법

스택

- 스택(stack) : 데이터를 넣은 순서의 역순으로만 꺼낼 수 있는 자료구조



- 스택으로 사용할 수 있는 클래스 : `LinkedList` 클래스

02. 자료구조 클래스의 사용 방법

스택 : LinkedList 클래스

- 스택 생성 방법

```
LinkedList<Integer> stack = new LinkedList<Integer>();
```

↑
타입 파라미터

↑
타입 파라미터

02. 자료구조 클래스의 사용 방법

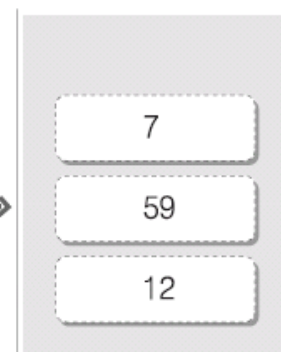
스택 : LinkedList 클래스

- 데이터를 넣는 방법

```
list.addLast(new Integer(12));  
list.addLast(new Integer(59));  
list.addLast(new Integer(7));
```

타입 파라미터에 해당하는
데이터를 넘겨주어야 합니다.

addLast 메소드는 데이터를
스택의 아래쪽부터 순서대로 저장합니다.



02. 자료구조 클래스의 사용 방법

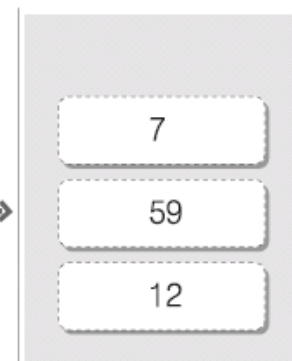
스택 : LinkedList 클래스

- 데이터를 꺼내는 방법

```
list.addLast(new Integer(12));  
list.addLast(new Integer(59));  
list.addLast(new Integer(7));
```

↑
타입 파라미터에 해당하는
데이터를 넘겨주어야 합니다.

addLast 메소드는 데이터를
스택의 아래쪽부터 순서대로 저장합니다.



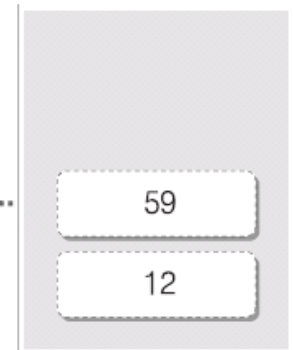
02. 자료구조 클래스의 사용 방법

스택 : LinkedList 클래스

- 데이터를 제거하지 않고 꺼내는 방법

Integer obj = getLast();

스택의 제일 위에 있는 59를 리턴합니다.



02. 자료구조 클래스의 사용 방법

스택 : LinkedList 클래스

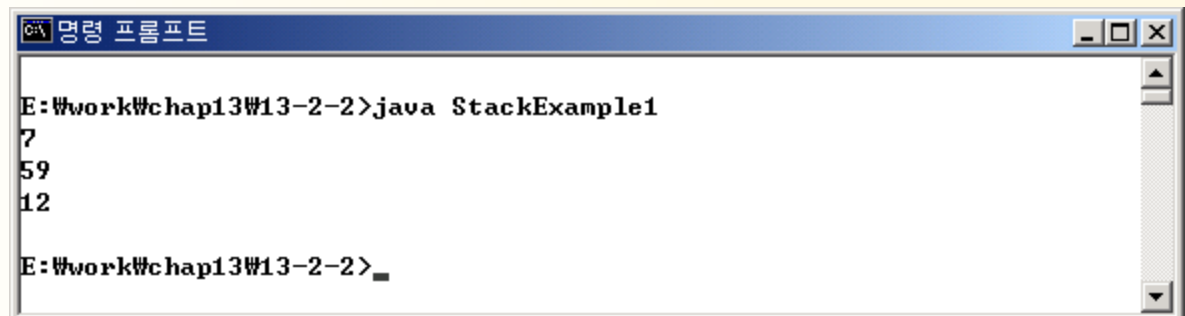
[예제 13-7] LinkedList 클래스를 스택으로 사용하는 예

```
1  import java.util.*;
2  class StackExample1 {
3      public static void main(String args[]) {
4          LinkedList<Integer> stack = new LinkedList<Integer>();
5          stack.addLast(new Integer(12));
6          stack.addLast(new Integer(59));
7          stack.addLast(new Integer(7));
8          while(!stack.isEmpty()) {
9              Integer num = stack.removeLast();
10             System.out.println(num);
11         }
12     }
13 }
```

스택으로 사용할 LinkedList 객체를 생성합니다.

스택에 3개의 데이터를 추가합니다.

루프를 돌면서 스택의 데이터를 모두 가져와서 출력합니다.



```
명령 프롬프트
E:\work\chap13\13-2-2>java StackExample1
7
59
12
E:\work\chap13\13-2-2>
```

02. 자료구조 클래스의 사용 방법

큐

- 큐(queue) : 데이터를 넣은 순서와 같은 순서로만 꺼낼 수 있는 자료구조



- 큐로 사용할 수 있는 클래스 : LinkedList 클래스

02. 자료구조 클래스의 사용 방법

큐 : LinkedList 클래스

- 큐 생성 방법

```
LinkedList<String> queue = new LinkedList<String>();
```

↑
타입 파라미터

↑
타입 파라미터

02. 자료구조 클래스의 사용 방법

큐 : LinkedList 클래스

- 사용 방법

```
queue.offer("토끼");  
queue.offer("사슴");  
queue.offer("호랑이");
```

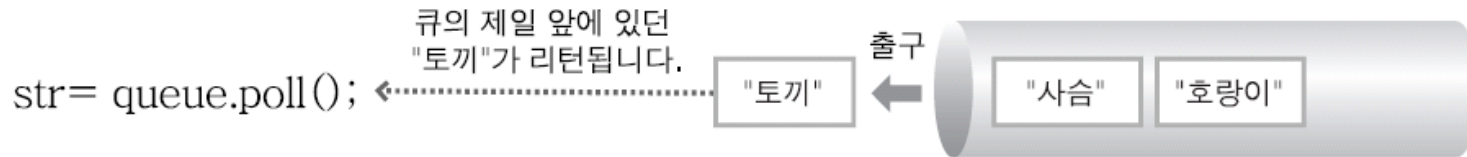
파라미터로 넘겨준
데이터를 큐에 저장합니다.



02. 자료구조 클래스의 사용 방법

큐 : LinkedList 클래스

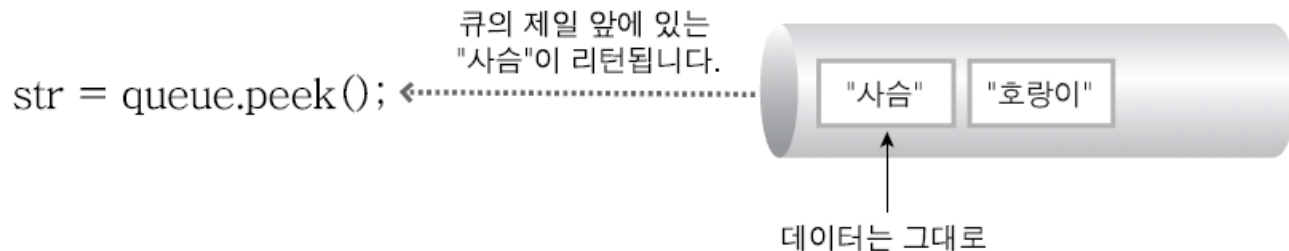
- 사용 방법



02. 자료구조 클래스의 사용 방법

큐 : LinkedList 클래스

- 사용 방법



02. 자료구조 클래스의 사용 방법

큐 : LinkedList 클래스

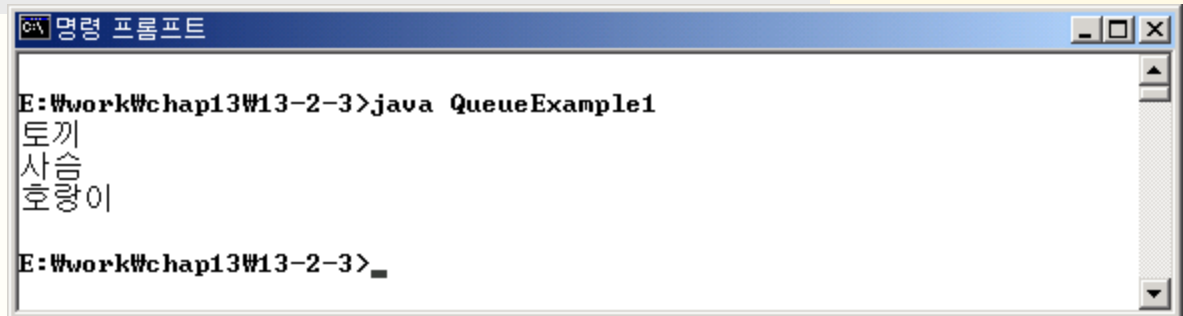
[예제 13-8] LinkedList 클래스를 큐로 사용하는 예

```
1  import java.util.*;
2  class QueueExample1 {
3      public static void main(String args[]) {
4          LinkedList<String> queue = new LinkedList<String>();
5          queue.offer("토끼");
6          queue.offer("사슴");
7          queue.offer("호랑이");
8          while(!queue.isEmpty()) {
9              String str = queue.poll();
10             System.out.println(str);
11         }
12     }
13 }
```

큐로 사용할 LinkedList 객체를 생성합니다.

큐에 3개의 데이터를 추가합니다.

루프를 돌면서 큐의 데이터를 모두 가져와서 출력합니다.



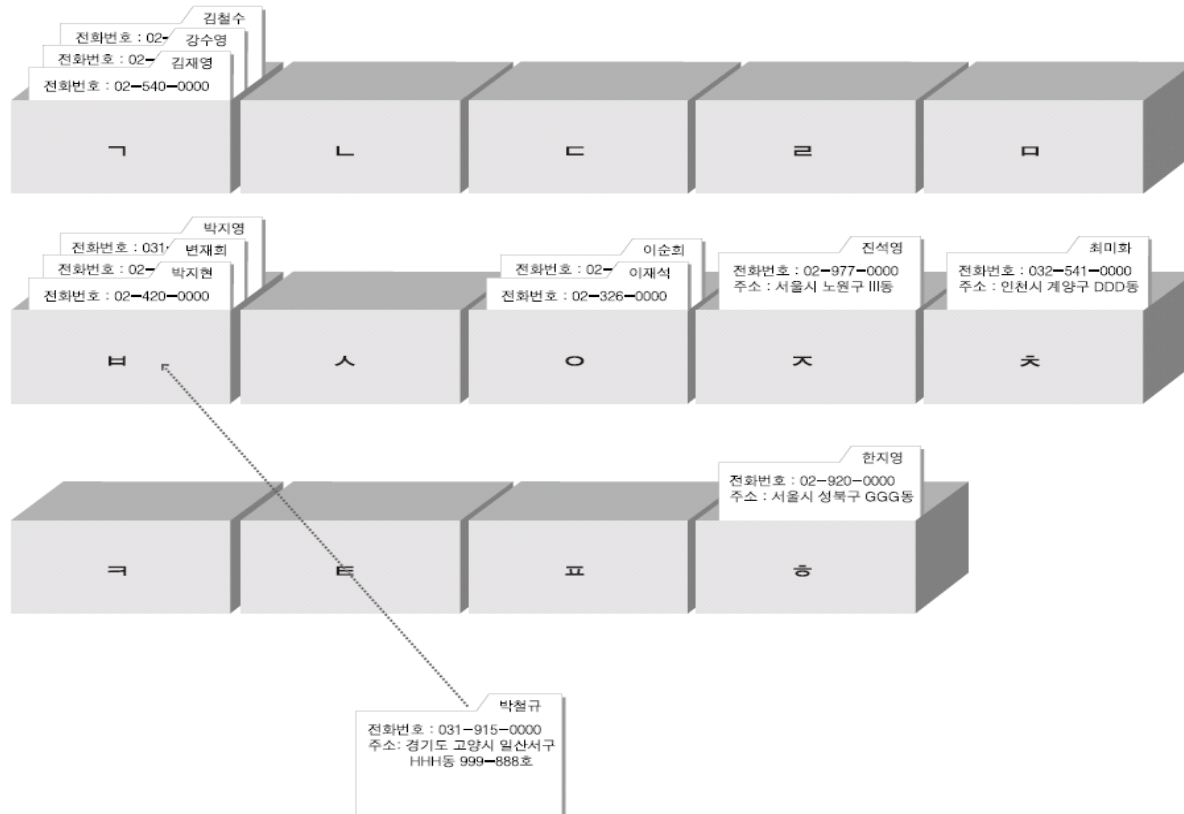
```
명령 프롬프트
E:\work\chap13\13-2-3>java QueueExample1
토끼
사슴
호랑이
E:\work\chap13\13-2-3>
```

02. 자료구조 클래스의 사용 방법

해쉬 테이블

- 해쉬 테이블(hash table)

여러 개의 통(bucket)을 만들어 두고 키 값을 이용하여 데이터를 넣을 통 번호를 계산하는 자료구조



키 값(이름)을 가지고 데이터를 넣을 통의 번호(첫번째 자음)를 계산해서 넣습니다.

02. 자료구조 클래스의 사용 방법

해쉬 테이블 : HashMap 클래스

- 해쉬 테이블로 사용할 수 있는 클래스 : HashMap 클래스
- 해쉬 테이블 생성 방법

```
HashMap<String, Integer> hashtable =
```

```
new HashMap<String, Integer>();
```

키의 타입

데이터의 타입

16 개의 통으로 구성된
해쉬 테이블을 생성합니다.


12	13	14	15
8	9	10	11
4	5	6	7
0	1	2	3

02. 자료구조 클래스의 사용 방법

해쉬 테이블 : HashMap 클래스

- 100 개의 통으로 구성된 해쉬 테이블 생성하기

```
HashMap<String , Integer> hashtable = new HashMap<String , Integer>(100);
```



100 개의 통으로 구성된
해쉬 테이블을 생성합니다.

02. 자료구조 클래스의 사용 방법

해쉬 테이블 : HashMap 클래스

- 데이터 넣기

```
hashtable.put("해리", new Integer(95));
```

키 값

데이터



키 값("해리")으로 통 번호(5)를
계산하여 그 통에 키 값과
데이터를 넣습니다.

해리
95

02. 자료구조 클래스의 사용 방법

해쉬 테이블 : HashMap 클래스

- 데이터 찾기

```
Integer num = hashtable.get("해리");
```

키 값

12	13	14	15
8	9	10	11
4	5	6	7
0	1	2	3

키 값으로 통 번호를 계산하고, 그 통 안에서
키 값과 일치하는 데이터를 찾아서 리턴합니다.

02. 자료구조 클래스의 사용 방법

해쉬 테이블 : HashMap 클래스

- 데이터 삭제하기

```
hashtable.remove("해리");
```

키 값

12	13	14	15
8	9	10	11
4	5	6	7
0	1	2	3

키 값으로 통 번호를 계산하고,
해당 통에서 키 값과 일치하는
데이터를 찾아서 삭제합니다.

해리
95

02. 자료구조 클래스의 사용 방법

해쉬 테이블 : HashMap 클래스

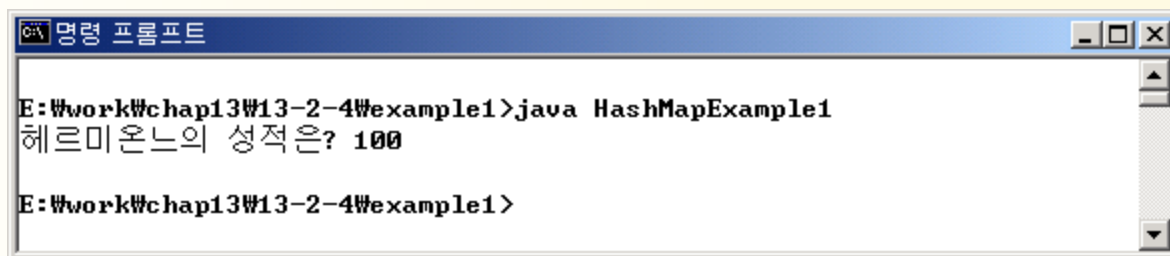
[예제 13-9] HashMap 클래스의 사용 예 (1)

```
1  import java.util.*;
2  class HashMapExample1 {
3      public static void main(String args[]) {
4          HashMap<String, Integer> hashtable = new HashMap<String, Integer>();
5          hashtable.put("해리", new Integer(95));
6          hashtable.put("헤르미온느", new Integer(100));
7          hashtable.put("론", new Integer(85));
8          hashtable.put("드레이코", new Integer(93));
9          hashtable.put("네빌", new Integer(70));
10         Integer num = hashtable.get("헤르미온느");
11         System.out.println("헤르미온느의 성적은? " + num);
12     }
13 }
```

----- 해쉬 테이블로 사용할
HashMap 객체를 생성합니다.

해쉬 테이블에 5개의 데이터를 추가합니다.

키 값으로 해쉬 테이블의 데이터를 찾아서
출력합니다.



```
명령 프롬프트
E:\work\chap13\13-2-4\example1>java HashMapExample1
헤르미온느의 성적은? 100
E:\work\chap13\13-2-4\example1>
```

02. 자료구조 클래스의 사용 방법

해쉬 테이블 : HashMap 클래스

- 키값을 가지고 해쉬 테이블의 통번호를 계산하는 공식은?

프로그래머가 알 필요가 없음

- 하지만 다음 사실은 꼭 알아두어야 함

해쉬 테이블 계산에는 hashCode 메소드가 사용됨

[예] “헤르미온느”라는 문자열이 키로 사용되면

그 문자열(String 객체)에 대해 hashCode 메소드가 호출됨

02. 자료구조 클래스의 사용 방법

해쉬 테이블 : hashCode 메소드

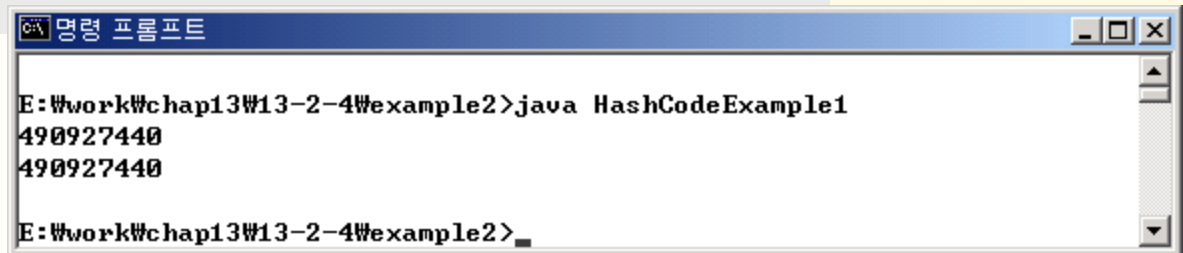
[예제 13-10] hashCode 메소드의 사용 예

```
1  class HashCodeExample1 {  
2      public static void main(String args[]) {  
3          String obj1 = new String("헤르미온느");  
4          String obj2 = new String("헤르미온느");  
5          int hash1 = obj1.hashCode();  
6          int hash2 = obj2.hashCode();  
7          System.out.println(hash1);  
8          System.out.println(hash2);  
9      }  
10 }
```

두 개의 String 객체를 생성합니다.

각각의 객체에 대해 hashCode 메소드를 호출합니다.

hashCode 메소드의 리턴 값을 출력합니다.



```
명령 프롬프트  
E:\work\chap13\13-2-4\example2>java HashCodeExample1  
490927440  
490927440  
E:\work\chap13\13-2-4\example2>
```


02. 자료구조 클래스의 사용 방법

해쉬 테이블 : hashCode 메소드

- 성과 이름을 모두 키로 사용하고 싶을 때는?
다음과 같이 직접 선언한 클래스를 사용하면 됩니다.

[예제 13-11] 사람의 이름을 표현하는 클래스

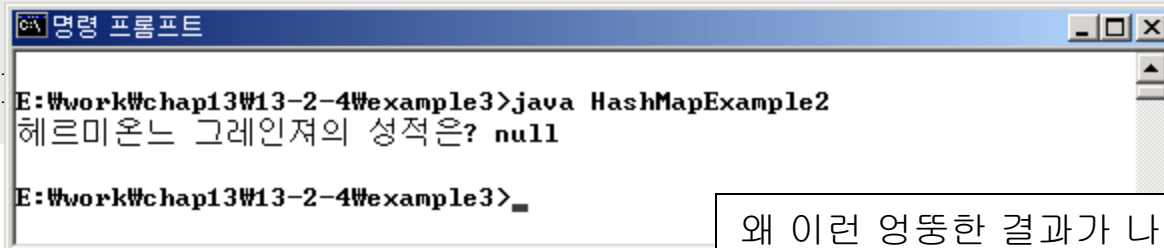
```
1    class Name {  
2        String firstName;        // 이름  
3        String lastName;        // 성  
4        Name(String firstName, String lastName) {  
5            this.firstName = firstName;  
6            this.lastName = lastName;  
7        }  
8    }
```

02. 자료구조 클래스의 사용 방법

해쉬 테이블 : hashCode 메소드

[예제 13-12] HashMap 클래스의 사용 예 (2)

```
1  import java.util.*;
2  class HashMapExample2 {
3      public static void main(String args[]) {
4          HashMap<Name, Integer> hashtable = new HashMap<Name, Integer>();
5          hashtable.put(new Name("해리", "포터"), new Integer(95));
6          hashtable.put(new Name("헤르미온느", "그레인저"), new Integer(100));
7          hashtable.put(new Name("론", "위즐리"), new Integer(85));
8          hashtable.put(new Name("드레이코", "말포이"), new Integer(93));
9          hashtable.put(new Name("네빌", "롱버텀"), new Integer(70));
10         Integer num = hashtable.get(new Name("헤르미온느", "그레인저"));
11         System.out.println("헤르미온느 그레인저의 성적은? " + num);
12
13
```



```
E:\work\chap13\13-2-4\example3>java HashMapExample2
헤르미온느 그레인저의 성적은? null

E:\work\chap13\13-2-4\example3>_
```

왜 이런 엉뚱한 결과가 나오는 걸까요?

02. 자료구조 클래스의 사용 방법

해쉬 테이블 : hashCode 메소드

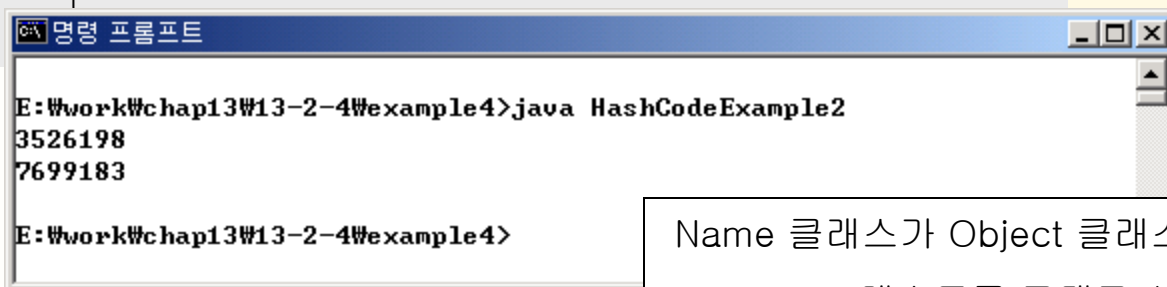
[예제 13-13] Name 클래스에 대해 hashCode 메소드를 호출하는 프로그램

```
1  class HashCodeExample2 {  
2      public static void main(String args[]) {  
3          Name obj1 = new Name("헤르미온느", "그레인저");  
4          Name obj2 = new Name("헤르미온느", "그레인저");  
5          int hash1 = obj1.hashCode();  
6          int hash2 = obj2.hashCode();  
7          System.out.println(hash1);  
8          System.out.println(hash2);  
9      }  
10 }
```

두 개의 Name 객체를 생성합니다.

각각의 객체에 대해 hashCode 메소드를 호출합니다.

hashCode 메소드의 리턴 값을 출력합니다.



```
E:\work\chap13\13-2-4\example4>java HashCodeExample2  
3526198  
7699183  
E:\work\chap13\13-2-4\example4>
```

Name 클래스가 Object 클래스로부터 상속받은 hashCode 메소드를 그대로 사용하고 있기 때문

02. 자료구조 클래스의 사용 방법

해쉬 테이블 : hashCode 메소드

- 직접 작성한 클래스를 해쉬 테이블의 키로 사용하려면?

hashCode 메소드를 오버라이드해야 합니다.

- hashCode 메소드의 오버라이드 방법 (1)

```
public int hashCode() {  
    return 1;  
}
```

상수를 리턴합니다.

한 통에만 데이터가 몰리게 됨

02. 자료구조 클래스의 사용 방법

해쉬 테이블 : hashCode 메소드

- hashCode 메소드의 오버라이드 방법 (2)

```
public int hashCode() {  
    return firstName.length() + lastName.length();  
}
```

필드 값을 기반으로 리턴 값을 계산하면
같은 값의 객체들은 같은 값을 리턴하게 됩니다.

통이 많아져도 몇몇 통에만
데이터가 물리게 됨

02. 자료구조 클래스의 사용 방법

해쉬 테이블 : hashCode 메소드

- hashCode 메소드의 오버라이드 방법

```
public int hashCode() {  
    return firstName.hashCode() + lastName.hashCode();  
}
```

String 타입 필드의 hashCode 메소드를 가지고 리턴 값을 계산하면 좀 더 고른 분포를 갖는 리턴 값을 만들 수 있습니다.

바람직한 방법

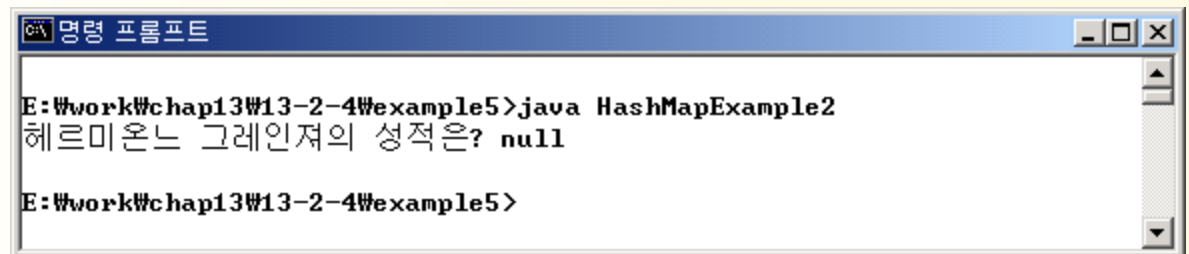
02. 자료구조 클래스의 사용 방법

해쉬 테이블 : hashCode 메소드

[예제 13-14] hashCode 메소드를 추가한 Name 클래스 - 미완성

```
1  class Name {  
2      String firstName;  
3      String lastName;  
4      Name(String firstName, String lastName) {  
5          this.firstName = firstName;  
6          this.lastName = lastName;  
7      }  
8      public int hashCode() {  
9          return firstName.hashCode() + lastName.hashCode();  
10     }  
11 }
```

} 추가된 hashCode 메소드



```
명령 프롬프트  
E:\work\chap13\13-2-4\example5>java HashMapExample2  
헤르미온느 그레인저의 성적은? null  
E:\work\chap13\13-2-4\example5>
```

02. 자료구조 클래스의 사용 방법

해쉬 테이블 : hashCode 메소드

[예제 13-15] equals 메소드를 추가한 Name 클래스 - 완성

```
1  class Name {
2      String firstName;
3      String lastName;
4      Name(String firstName, String lastName) {
5          this.firstName = firstName;
6          this.lastName = lastName;
7      }
8      public boolean equals(Object obj) {
9          if (!(obj instanceof Name))
10             return false;
11          Name name = (Name) obj;
12          if (firstName.equals(name.firstName) && lastName.equals(name.lastName))
13             return true;
14          else
15             return false;
16      }
17      public int hashCode() {
18          return firstName.hashCode() * 31 + lastName.hashCode();
19      }
20 }
```

추가된 equals 메소드



명령 프롬프트 창에서 실행된 명령과 출력 결과:

```
E:\work\chap13\13-2-4\example6>java HashMapExample2
헤르미온느 그레인저의 성적은? 100

E:\work\chap13\13-2-4\example6>
```


02. 자료구조 클래스의 사용 방법

집합

- 집합(Set) : 수학에서 말하는 집합처럼 데이터를 중복 저장하지 않음
- 집합으로 사용할 수 있는 클래스 : HashSet 클래스

02. 자료구조 클래스의 사용 방법

집합 : HashSet 클래스

- 집합 생성 방법

```
HashSet<String> set = new HashSet<String>();
```

↑
타입 파라미터

↑
타입 파라미터

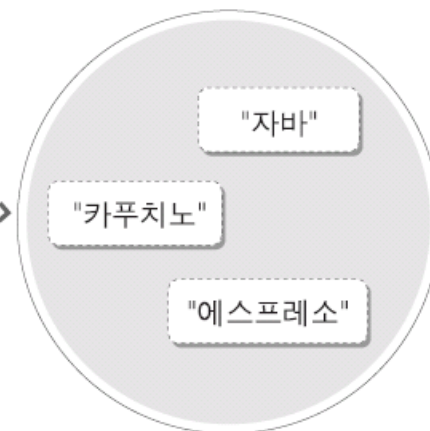
02. 자료구조 클래스의 사용 방법

집합: HashSet 클래스

- 데이터 추가 방법

```
set.add("자바");
```

이미 있는 데이터를 저장하면
집합에 변동이 일어나지 않습니다.



02. 자료구조 클래스의 사용 방법

집합 : HashSet 클래스

- 데이터의 수를 가져오는 방법

```
int num = set.size();
```

집합에 있는 데이터의 수
3을 리턴합니다.



02. 자료구조 클래스의 사용 방법

집합 : HashSet 클래스

- 모든 데이터를 읽어오는 방법

```
Iterator<String> iterator = set.iterator();  
while (iterator.hasNext()) {  
    String str = iterator.next();  


데이터 처리 부분

  
}
```

iterator 메서드를 호출하여
iterator 객체를 가져옵니다.

iterator 객체의 데이터를
순서대로 가져와서 처리합니다.


02. 자료구조 클래스의 사용 방법

집합 : HashSet 클래스

[예제 13-16] HashSet의 사용 예

```
1  import java.util.*;
2  class SetExample1 {
3      public static void main(String args[]) {
4          HashSet<String> set = new HashSet<String>();  ----- 집합으로 사용할 HashSet 객체를 생성합니다.
5          set.add("자바");
6          set.add("카푸치노");
7          set.add("에스프레소");
8          set.add("자바");
9          System.out.println("저장된 데이터의 수 = " + set.size());
10         Iterator<String> iterator = set.iterator();
11         while (iterator.hasNext()) {
12             String s = iterator.next();
13             System.out.println(s);
14         }
15     }
16 }
```

집합에 데이터를 저장합니다.



```
E:\work\chap13\13-2-5>java SetExample1
저장된 데이터의 수 = 3
에스프레소
카푸치노
자바
E:\work\chap13\13-2-5>
```