

Chapter 15

java.util 1 :

java.util (group) (subsystem) 2 가 , 2 1.4 가 java.util

가 , java.util

가

(pseudorandom)

, java.util

가

java.util

AbstractCollection(Java 2)	EventObject	PropertyResourceBundle
AbstractList(Java 2)	GregorianCalendar	Random
AbstractMap(Java 2)	HashMap(Java 2)	ResourceBundle
AbstractSequentialList(Java 2)	HashSet(Java 2)	SimpleTimeZone
AbstractSet(Java 2)	Hashtable	Stack
ArrayList(Java 2)	IdentityHashMap(Java 2, v1.4)	StringTokenizer
Arrays(Java 2)	LinkedHashMap(Java 2, v1.4)	Timer(Java 2, v1.3)
BitSet	LinkedHashSet(Java 2, v1.4)	TimerTask(Java 2, v1.3)
Calendar	LinkedList(Java 2)	TimeZone
Collections(Java 2)	ListResourceBundle	Treemap(Java 2)
Currency(Java 2, v1.4)	Locale	TreeSet(Java 2)
Date	Observable	Vector
Dictionary	Properties	WeakHashMap(Java 2)
EventListenerProxy(Java 2, v1.4)	PropertyPermission(Java 2)	

java.util

2 가

HashSet, TreeSet) (LinkedList,
가
가
Collections 가 가
가
가 가 Iterator Iterator
, iterator Iterator Iterator가
Iterator , set
list
가 , Map Map
map
, Map (collection-view) Map
Map
, java.util
가, (deprecation)
가
가 C++ C++
(Standard Template Library)
. C++가 , 가

컬렉션 인터페이스

Collection 인터페이스는 Collection 클래스를 구현하는 클래스를 정의하는 인터페이스이다. Collection 인터페이스는 Collection 클래스를 상속하는 클래스를 정의하는 인터페이스이다. Collection 인터페이스는 Collection 클래스를 상속하는 클래스를 정의하는 인터페이스이다.

Collection	가	.
List	()	Collection .
Set		Collection .
SortedSet	Set	.

가 , Comparator, Iterator, ListIterator, RandomAccess .
 (Comparator) , Iterator
 ListIterator . RandomAccess
 , .
 가
 가 (modifiable)
 가 (unmodifiable) . 가
 Exception 가 . UnsupportedOperation-
 가 .

Collection

Collection 가 가 .


```

int size()

Object[] toArray()

Object[] toArray
(Object array[] )
array 가
array 가
array 가
array 가
가
null
가 array
, ArrayStoreException
    
```

[15-1] Collection

```

add() 가 . add() Object
가 . Object
, (primitive)
int, char, double
, 14
(wrapper) . addAll()
가

remove()
, removeAll() . retainAll()

, clear()

contains() 가

containsAll() . isEmpty()

size()
    
```

toArray()

가

가

equals()

equals()

가

, equals()

가

가

(iterator)

iterator()

List

Collection

List

. 0

(index)

Collection

가

, List

15-2

UnsupportedOperationException

가

가

가

ClassCastException

가

Collection

add()

addAll()

, List

add(int, Object)

addAll(int, Collection)

가

가

, Collection

add(Object)

addAll(Collection)

가 List

get()

set()

, indexOf()

lastIndexOf()

<code>void add(int index, Object obj)</code>	<code>index</code>	<code>obj</code>
<code>(overwrite)</code>	<code>.</code>	<code>,</code>
<code>boolean addAll(int index, Collection c)</code>	<code>index</code>	<code>c</code>
<code>(overwrite)</code>	<code>.</code>	<code>,</code>
<code>가</code>	<code>true</code>	<code>,</code>
<code>false</code>	<code>.</code>	<code>.</code>
<code>Object get(int index)</code>	<code>.</code>	<code>.</code>
<code>int indexOf(Object obj)</code>	<code>obj</code>	<code>가</code>
<code>.</code>	<code>obj가</code>	<code>, -1</code>
<code>.</code>	<code>.</code>	<code>.</code>
<code>int lastIndexOf(Object obj)</code>	<code>obj</code>	<code>가</code>
<code>.</code>	<code>obj가</code>	<code>, -1</code>
<code>.</code>	<code>.</code>	<code>.</code>
<code>ListIterator listIterator()</code>	<code>.</code>	<code>.</code>
<code>ListIterator listIterator(int index)</code>	<code>.</code>	<code>.</code>
<code>Object remove(int index)</code>	<code>index</code>	<code>가</code>
<code>.</code>	<code>.</code>	<code>.</code>
<code>1</code>	<code>.</code>	<code>.</code>
<code>Object set(int index, Object obj)</code>	<code>index</code>	<code>obj</code>
<code>.</code>	<code>.</code>	<code>.</code>
<code>List subList(int start, int end)</code>	<code>start</code>	<code>end-1</code>
<code>.</code>	<code>.</code>	<code>.</code>

[15-2] List

(sublist)

, subList()

, subList()

Set

Set
가
가
. Set
, add()
Collection
false
. Set

SortedSet

SortedSet
. Set
가
, SortedSet 15-3
NoSuchElementException 가
ClassCastException 가 null null
, NullPointerException 가
SortedSet
first()
, last()
subSet()
(subset)
, headSet()
, tailSet()

```
Comparator comparator()  
    (comparator)  
    가  
    null  
Object first()  
SortedSet headSet(Object end)  
    end  
    SortedSet  
    가  
Object last()  
SortedSet subSet(Object start, Object end)  
    start end-1  
    SortedSet
```

SortedSet tailSet(Object start)	가	start	가	start
				SortedSet

[15-3] SortedSet

Collection 클래스

가

가

AbstractCollection	Collection	
AbstractList	AbstractCollection	List
AbstractSequentialList		AbstractList
LinkedList	AbstractSequentialList	
ArrayList	AbstractList	
AbstractSet	AbstractCollection	Set
HashSet		AbstractSet
LinkedHashSet	가	HashSet
TreeSet		AbstractSet

NOTE

(*legacy*) 가 , *Vector*, *Stack*, *Hashtable*

ArrayList

ArrayList AbstractList List ArrayList

가 가 가 가

ArrayList (variable-length) ArrayList

ArrayList 가 가 가

NOTE

Vector

ArrayList 가

- ArrayList()
- ArrayList(Collection *c*)
- ArrayList(int *capacity*)

가 가 가

ArrayList *c* (*capacity*)

ArrayList 가

Contents of al: [C, A2, A, E, B, D, F]

Size of al after deletions: 5

Contents of al: [C, A2, E, B, D]

a1 . , a1
AbstractCollection toString() 가
(routine)
, toString()
ArrayList 가 , ensure-
Capacity() ArrayList 가
가 ,
ensureCapacity()
(signature)
void ensureCapacity(int cap)
, cap
가 ArrayList trim-
ToSize() trim-
void trimToSize()
·ArrayList Array
ArrayList , 가
, toArray()

, ArrayList array

```
// ArrayList
import java.util.*;

class ArrayListToArray {
    public static void main(String args[]) {
        //
        ArrayList al = new ArrayList();

        //
        al.add(new Integer(1));
        al.add(new Integer(2));
        al.add(new Integer(3));
        al.add(new Integer(4));

        System.out.println("Contents of al: " + al);

        //
        Object ia[] = al.toArray();
        int sum = 0;

        //
        for(int i=0; i<ia.length; i++)
            sum += ((Integer) ia[i]).intValue();

        System.out.println("Sum is: " + sum);
    }
}
```

Contents of al: [1, 2, 3, 4]
Sum is: 10

, Integer
Object
Object
toArray()
Integer

LinkedList

LinkedList AbstractSequentialList List

가

LinkedList()
LinkedList(Collection *c*)

c

LinkedList 가

가

addFirst() 가
addLast()

void addFirst(Object *obj*)
void addLast(Object *obj*)

, *obj* 가

, getFirst() 가

, getLast()

Object getFirst()
Object getLast()

, removeFirst()

, removeLast()

```
Object removeFirst()
Object removeLast()
```

LinkedList

```
// LinkedList
import java.util.*;

class LinkedListDemo {
    public static void main(String args[]) {
        //
        LinkedList ll = new LinkedList();

        //          가
        ll.add("F");
        ll.add("B");
        ll.add("D");
        ll.add("E");
        ll.add("C");
        ll.addLast("Z");
        ll.addFirst("A");

        ll.add(1, "A2");

        System.out.println("Original contents of ll: " + ll);

        //
        ll.remove("F");
        ll.remove(2);

        System.out.println("Contents of ll after deletion: " + ll);

        //
        ll.removeFirst();
        ll.removeLast();
        System.out.println("ll after deleting first and last: " + ll);

        //          가
        Object val = ll.get(2);
        ll.set(2, (String) val + " Changed");
    }
}
```

```
System.out.println("ll after change: " + ll);
    }
}
```

Original contents of ll: [A, A2, F, B, D, E, C, Z]
Contents of ll after deletion: [A, A2, D, E, C, Z]
ll after deleting first and last: [A2, D, E, C]
ll after change: [A2, D, E Changed, C]

```
LinkedList List, add(Object)
addLast() 가
, add(1, A2), add()
add(int, Object)
ll 가 get() set()
, get()
set()
```

HashSet

```
HashSet AbstractSet Set HashSet
(hashing)
가
, add(), contains(), remove(), size()
HashSet()
```

```

HashSet(Collection c)
HashSet(int capacity)
HashSet(int capacity, float fillRatio)

```

```

// HashSet
import java.util.*;

class HashSetDemo {
    public static void main(String args[]) {
        //
        HashSet hs = new HashSet();

        //
        hs.add("B");
        hs.add("A");
        hs.add("D");
        hs.add("E");
        hs.add("C");
        hs.add("F");

        System.out.println(hs);
    }
}

```

[A, F, E, D, C, B]

LinkedHashSet

2 1.4 LinkedHashSet 가 가 . HashSet
가 . LinkedHashSet 가
가 . , LinkedHashSet ,
. LinkedHashSet toString()
. LinkedHashSet
, HashSet LinkedHashSet .

[B, A, D, E, C, F]

TreeSet

TreeSet Set .
. TreeSet 가 .
가 .

TreeSet()
TreeSet(Collection *c*)
TreeSet(Comparator *comp*)
TreeSet(SortedSet *ss*)

. *c* . *comp*
(comparator, .)
. *ss* .

TreeSet

```
// TreeSet
import java.util.*;

class TreeSetDemo {
    public static void main(String args[]) {
        //
        TreeSet ts = new TreeSet();

        //
        ts.add("C");
        ts.add("A");
        ts.add("B");
        ts.add("E");
        ts.add("F");
        ts.add("D");

        System.out.println(ts);
    }
}
```

[A, B, C, D, E, F]

, TreeSet

반복자를 통해 컬렉션 접근

가 Iterator ListIterator
 iterator . Iterator
 가 . Iterator ListIterator (bi-
 directional traversal) . Iterator 15-4
 . ListIterator 15-5

iterator()

boolean hasNext()	가	true	.
	false	.	.
Object next()		.	가
	NoSuchElementException	.	.
void remove()		next()	가
	remove()		IllegalState-
	Exception	.	.

[15-4] Iterator

void add(Object obj)		next()		obj
boolean hasNext()	가	true	.	.
	false	.	.	.
boolean hasPrevious()	가	true	.	.
	false	.	.	.
Object next()		.	가	.
	NoSuchElementException	.	.	.
int nextIndex()		.	가	.
Object previous()		.	가	.
	NoSuchElementException	.	.	.
int previousIndex()		.	가	.
	, -1	.	.	.
void remove()	가	remove()	가	next() previous() Illegal-
	StateException	.	.	.
void set(Object obj)	obj	.	obj next() previ-	ous()



[15-5] ListIterator

1. iterator()

2. hasNext() hasNext() 가 true

3. next()

List, ListIterator

ListIterator Iterator

Iterator ListIterator 가

ArrayList

ListIterator List

가

```
//
import java.util.*;

class IteratorDemo {
    public static void main(String args[]) {
        //
        ArrayList al = new ArrayList();

        // 가
        al.add("C");
        al.add("A");
        al.add("E");
        al.add("B");
        al.add("D");
        al.add("F");

        // al
        System.out.print("Original contents of al: ");
        Iterator itr = al.iterator();
```

```

while(itr.hasNext()) {
    Object element = itr.next();
    System.out.print(element + " ");
}
System.out.println();

//
ListIterator litr = al.listIterator();
while(litr.hasNext()) {
    Object element = litr.next();
    litr.set(element + "+");
}

System.out.print("Modified contents of al: ");
itr = al.iterator();
while(itr.hasNext()) {
    Object element = itr.next();
    System.out.print(element + " ");
}
System.out.println();

//
System.out.print("Modified list backwards: ");
while(litr.hasPrevious()) {
    Object element = litr.previous();
    System.out.print(element + " ");
}
System.out.println();
}
}

```

Original contents of al: C A E B D F

Modified contents of al: C+ A+ E+ B+ D+ F+

Modified list backwards: F+ D+ B+ E+ A+ C+

가 . 가 , litr
가 .(litr.hasNext()) false
.) ,
litr . 가

컬렉션의 사용자 정의 클래스 저장

```
import java.util.*;
import java.io.*;

class Address {
    String name;
    String street;
    String city;
    String state;
    String code;

    Address(String n, String s, String c,
            String st, String cd) {
        name = n;
        street = s;
        city = c;
        state = st;
        code = cd;
    }

    public String toString() {
        return name + "\n" + street + "\n" +
            city + " " + state + " " + code;
    }
}

class MailList {
    public static void main(String args[]) {
        LinkedList ml = new LinkedList();
    }
}
```

```
//
import java.util.*;

class Address {
    private String name;
    private String street;
    private String city;
    private String state;
    private String code;

    Address(String n, String s, String c,
            String st, String cd) {
        name = n;
        street = s;
        city = c;
        state = st;
        code = cd;
    }

    public String toString() {
        return name + "\n" + street + "\n" +
            city + " " + state + " " + code;
    }
}

class MailList {
    public static void main(String args[]) {
        LinkedList ml = new LinkedList();
    }
}
```

```
//          가  
ml.add(new Address("J.W. West", "11 Oak Ave",  
                  "Urbana", "IL", "61801"));  
ml.add(new Address("Ralph Baker", "1142 Maple Lane",  
                  "Mahomet", "IL", "61853"));  
ml.add(new Address("Tom Carlton", "867 Elm St",  
                  "Champaign", "IL", "61820"));  
  
Iterator itr = ml.iterator();  
while(itr.hasNext()) {  
    Object element = itr.next();  
    System.out.println(element + "\n");  
}  
System.out.println();  
}  
}
```

J.W. West
11 Oak Ave
Urbana IL 61801

Ralph Baker
1142 Maple Lane
Mahomet IL 61853

Tom Carlton
867 Elm St
Champaign IL 61820

RandomAccess 인터페이스

2 1.4 RandomAccess 가 가 .
.
.
RandomAccess
instanceof .(가
RandomAccess .) ArrayList Vector
RandomAccess .

Map 사용

(key) , / 가 ,
.
.
.
Map

Map	(map)	.
Map.Entry	(/)	. Map
SortedMap	Map	.

-Map

Map
 가
 , Map
 . Map
 15-6
 NoSuchElementException
 ElementException 가
 ClassCastException
 , NullPointerException
 UnsupportedOperationException

```

void clear() /
boolean containsKey(Object k) , false true
boolean containsValue(Object v) , false true
Set entrySet() Map.Entry Set Set
boolean equals(Object obj) obj가 Map , false true
Object get(Object k) k
int hashCode()
boolean isEmpty() true
Set keySet() Set
Object put(Object k, Object v) k v 가
null
void putAll(Map m) m
Object remove(Object k) 가 k
int size() /
Collection values()
    
```

[15-6] Map

```

get() put() 가
put()
get()
Collection
- , entrySet()
Set
, keySet()
values()

```

-SortedMap

```

SortedMap Map 가
SortedMap 15-7
NoSuchElementException 가
ClassCastException
NullPointerException
( , )
headMap(), tailMap(), subMap()
, firstKey()
, lastKey()

```

Comparator comparator()	(natural ordering)
Object firstKey()	가
SortedMap headMap(Object end)	가
Object lastKey()	
SortedMap subMap(Object start, Object end)	가

SortedMap tailMap(Object start)	start	가
---------------------------------	-------	---

[15-7] SortedMap

·Map.Entry

Map.Entry	Set	Map
entrySet()	15-8	
Map.Entry		

boolean equals(Object obj)	obj가	가	Map.
	Entry true		
Object getKey()			
Object getValue()			
int hashCode()			
Object setValue(Object v)	v	v가	
		ClassCastException	
	v 가	IllegalArgumentException	
	v가		
	NullPointerException		
	UnsupportedOperationException		

[15-8] Map.Entry

(Map)

AbstractMap	Map
-------------	-----

HashMap		AbstractMap	.
TreeMap		AbstractMap	.
WeakHashMap	(weak key)	AbstractMap	.
LinkedHashMap	-	HashMap	.
IdentityHashMap	AbstractMap	(document)	가

AbstractMap 가 가 가 WeakHashMap (garbage-collected)

·HashMap

HashMap Map
get() put()

가

HashMap()
HashMap(Map *m*)
HashMap(int *capacity*)
HashMap(int *capacity*, float *fillRatio*)

m
capacity

HashSet

HashMap Map AbstractMap
가 가 가 가 가 가

HashMap

```
import java.util.*;

class HashMapDemo {
    public static void main(String args[])
    {
        //
        HashMap hm = new HashMap();

        //
        hm.put("John Doe", new Double(3434.34));
        hm.put("Tom Smith", new Double(123.22));
        hm.put("Jane Baker", new Double(1378.00));
        hm.put("Todd Hall", new Double(99.22));
        hm.put("Ralph Smith", new Double(-19.08));

        //      가
        Set set = hm.entrySet();

        //
        Iterator i = set.iterator();

        //
        while(i.hasNext()) {
            Map.Entry me = (Map.Entry)i.next();
            System.out.print(me.getKey() + ": ");
            System.out.println(me.getValue());
        }
        System.out.println();

        // John Doe      1000
        double balance = ((Double)hm.get("John Doe")).doubleValue();
        hm.put("John Doe", new Double(balance + 1000));
        System.out.println("John Doe's new balance: " +
            hm.get("John Doe"));
    }
}
```

()

Todd Hall: 99.22
 Ralph Smith: -19.08
 John Doe: 3434.34
 Jane Baker: 1378.0
 Tom Smith: 123.22

John Doe's current balance: 4434.34

```

        (mapping) 가 .
        entrySet()
        Map.Entry getKey() getValue() . John
Doe . put()
        , John Doe 가
        John Doe
    .TreeMap
    TreeMap Map . TreeMap
    / (retrieval) 가
    TreeMap
    TreeMap()
    TreeMap(Comparator comp)
    TreeMap(Map m)
    TreeMap(SortedMap sm)
    Comparator comp
    m
    sm sm
    TreeMap SortedMap AbstractMap
    가
    TreeMap
    
```

```

import java.util.*;

class TreeMapDemo {
    public static void main(String args[]) {

        //
        TreeMap tm = new TreeMap();

        //
        tm.put("John Doe", new Double(3434.34));
        tm.put("Tom Smith", new Double(123.22));
        tm.put("Jane Baker", new Double(1378.00));
        tm.put("Todd Hall", new Double(99.22));
        tm.put("Ralph Smith", new Double(-19.08));

        //
        Set set = tm.entrySet();

        //
        Iterator i = set.iterator();

        //
        while(i.hasNext()) {
            Map.Entry me = (Map.Entry)i.next();
            System.out.print(me.getKey() + ": ");
            System.out.println(me.getValue());
        }
        System.out.println();

        // John Doe          100
        double balance = ((Double)tm.get("John Doe")).doubleValue();
        tm.put("John Doe", new Double(balance + 1000));
        System.out.println("John Doe's new balance: " + tm.get("John Doe"));
    }
}

```

Jane Baker: 1378.0
 John Doe: 3434.34
 Ralph Smith: -19.08
 Todd Hall: 99.22

Tom Smith: 123.22

John Doe's current balance: 4434.34

```

    TreeMap (last name)
      (first name)

```

·LinkedHashMap

```

    2 1.4 LinkedHashMap 가 가 HashMap
    . LinkedHashMap ,
      가 , LinkedHashMap
    ,
      LinkedHashMap

```

LinkedHashMap

```

    LinkedHashMap()
    LinkedHashMap(Map m)
    LinkedHashMap(int capacity)
    LinkedHashMap(int capacity, float fillRatio)
    LinkedHashMap(int capacity, float fillRatio, boolean Order)

```

```

    LinkedHashMap m
    LinkedHashMap
      가 HashMap

```

```

    가 Order가 true ,
      가 Order가 false , 가

```

```

    LinkedHashMap HashMap 가
      removeEldestEntry()

```

```

    protected boolean removeEldestEntry(Map.Entry e)

```

```

    put() putAll() 가 가 e
    , false ,

```



```

boolean equals(Object obj)

obj . obj 가 Comparator
, true . , false
. equals()
.
.
compare
.
.

```

```

//
import java.util.*;

//
class MyComp implements Comparator {
    public int compare(Object a, Object b) {
        String aStr, bStr;

        aStr = (String) a;
        bStr = (String) b;

        //
        return bStr.compareTo(aStr);
    }

    // equals 가
}

class CompDemo {
    public static void main(String args[]) {
        //
        TreeSet ts = new TreeSet(new MyComp());

        // 가
        ts.add("C");
        ts.add("A");
        ts.add("B");
    }
}

```

```

ts.add("E");
ts.add("F");
ts.add("D");

//
Iterator i = ts.iterator();

//
while(i.hasNext()) {
    Object element = i.next();
    System.out.print(element + " ");
}
System.out.println();
}
}

```

F E D C B A

Comparator	compare()	MyComp
.(, equals()	
.) compare()	, String	compareTo()
.	, aStr	bStr
가		가 compareTo()
,		
.		가 (first name)
	(last name)	
(last name)		(last name)

TreeMap

```

//
import java.util.*;

//
class TComp implements Comparator {
    public int compare(Object a, Object b) {
        int i, j, k;
    }
}

```

```
String aStr, bStr;

aStr = (String) a;
bStr = (String) b;

//
i = aStr.lastIndexOf(' ');
j = bStr.lastIndexOf(' ');

k = aStr.substring(i).compareTo(bStr.substring(j));
if(k==0) //
    return aStr.compareTo(bStr);
else
    return k;
}

// 가
}

class TreeMapDemo2 {
    public static void main(String args[]) {
        //
        TreeMap tm = new TreeMap(new TComp());

        //
        tm.put("John Doe", new Double(3434.34));
        tm.put("Tom Smith", new Double(123.22));
        tm.put("Jane Baker", new Double(1378.00));
        tm.put("Todd Hall", new Double(99.22));
        tm.put("Ralph Smith", new Double(-19.08));

        //
        Set set = tm.entrySet();

        //
        Iterator itr = set.iterator();

        //
        while(itr.hasNext()) {
            Map.Entry me = (Map.Entry)itr.next();
            System.out.print(me.getKey() + ": ");
            System.out.println(me.getValue());
        }
    }
}
```

```

System.out.println();

// John Doe          1000
double balance = ((Double)tm.get("John Doe")).doubleValue();
tm.put("John Doe", new Double(balance + 1000));
System.out.println("John Doe's new balance: " +
    tm.get("John Doe"));
}
}

```

가 (last name)

Jane Baker: 1378.0
 John Doe: 3434.34
 Todd Hall: 99.22
 Ralph Smith: -19.08
 Tom Smith: 123.22

John Doe's new balance: 4434.34

Tcomp (last name) 가
 (last name)
 (last name), (first name)
 (last name) (last name) (first name)
 Tom Smith Ralph Smith가

컬렉션 알고리즘

Collections (static method)
 15-9
 ClassCastException 가
 UnsupportedOperationException

synchronizedList() synchronizedSet() 가
 ()
 . , .
 synchronized .
 unmodifiable 가
 (가)

static int binarySearch(List list, Object value, Comparator c)	c value 1	list value value	. list value
static int binarySearch(List list, Object value)	list value 1	value value	. value
static void copy(List list1, List list2)	list2	list1	.
static Enumeration enumeration(Collection c)	c Enumeration	.(.)	.
static void fill(List list, Object obj)	list	obj	.
static int indexOfSubList(List list, List subList)	subList가 1 (2 1.4 가 .)	list .	.
static int lastIndexOfSubList(List list, List subList)	subList가 1 (2 1.4 가 .)	list .	.
static ArrayList list(Enumeration enum)	enum 2 1.4 가 .)	ArrayList .	.(
static Object max(Collection c, Comparator comp)	comp	c 가	.
static Object max(Collection c)	.	c 가	.

static Object min(Collection c, Comparator comp)	comp	c	가	.
static Object min(Collection c)	.	c	가	.
static List nCopies(int num, Object obj)	.	num	0	obj num
static boolean replaceAll(List list, Object old, Object new)	list	가	old new	.
	, false	.	true	.
	.	.	(2 1.4	가
static void reverse(List list)	list	.	.	.
static Comparator reverseOrder()	가	.	()
static void rotate(List list, int n)	list	.	n	.
	.	.	, n	.
	2 1.4	가	.	(
static void shuffle(List list, Random r)	.	.	r	list
	.	.	.	(i.e., randomizes)
static void shuffle(List list)	list	.	.	(i.e., randomizes)
static Set singleton(Object obj)	obj	.	.	.
	.	.	가	.
static List singletonList(Object obj)	obj	.	.	.
	.	.	가	.
	(2 1.3	가	.	.
static Map singletonMap(Object k, Object v)	/	k/v	.	.
	.	.	/	가
	.	.	(2 1.3	가
static void sort(List list, Comparator comp)	comp	.	.	list
static void sort(List list)	.	.	.	list
static void swap(List list, int idx1, int idx2)	list	idx1	idx2	.
	.	.	(2 1.4	가
static Collection synchronizedCollection(Collection c)	c	.	.	.

static List synchronizedList(List list)	list	.
static Map synchronizedMap(Map m)	m	.
static Set synchronizedSet (Set s)	s	.
static SortedMap synchronizedSortedMap (SortedMap sm)	sm	.
static SortedSet synchronizedSortedSet (SortedSet ss)	ss	.
static Collection unmodifiableCollection (Collection c)	c	.
static List unmodifiableList(List list)	list	.
static Map unmodifiableMap(Map m)	m	.
static Set unmodifiableSet (Set s)	s	.
static SortedMap unmodifiableSortedMap (SortedMap sm)	sm	.
static SortedSet unmodifiableSortedSet (SortedSet ss)	ss	.

[15-9] Collections

Collections EMPTY_SET, EMPTY_LIST, EMPTY_MAP (static
variable) . EMPTY_MAP 2 1.3

가 .

가 Comparator . reverseOrder() Integer

shuffle()

```
//
import java.util.*;

class AlgorithmsDemo {
    public static void main(String args[]) {

        //
        LinkedList ll = new LinkedList();
        ll.add(new Integer(-8));
        ll.add(new Integer(20));
        ll.add(new Integer(-20));
        ll.add(new Integer(8));

        //
        Comparator r = Collections.reverseOrder();

        //
        Collections.sort(ll, r);

        //
        Iterator li = ll.iterator();

        System.out.print("List sorted in reverse: ");
        while(li.hasNext())
            System.out.print(li.next() + " ");
        System.out.println();

        Collections.shuffle(ll);

        //
        li = ll.iterator();
        System.out.print("List shuffled: ");
        while(li.hasNext())
            System.out.print(li.next() + " ");
        System.out.println();

        System.out.println("Minimum: " + Collections.min(ll));
        System.out.println("Maximum: " + Collections.max(ll));
    }
}
```

```
List sorted in reverse: 20 8 -8 -20
List shuffled: 20 -20 8 -8
Minimum: -20
Maximum: 20
```

```
가          min() max()   가
          가
```

배열

Arrays

```
. Arrays          가          . Arrays
```

asList()

List

```
,
가
```

```
static List asList(Object[ ] array)
```

array

```
. binarySearch()
```

```
가
```

```
static int binarySearch(byte[ ] array, byte value)
static int binarySearch(char[ ] array, char value)
static int binarySearch(double[ ] array, double value)
static int binarySearch(float[ ] array, float value)
static int binarySearch(int[ ] array, int value)
static int binarySearch(long[ ] array, long value)
static int binarySearch(short[ ] array, short value)
static int binarySearch(Object[ ] array, Object value)
static int binarySearch(Object[ ] array, Object value, Comparator c)
```

```
, array
```

```
value
```

```
. 가
```

```

    array가 ( , Double StringBuffer)
value가 array ClassCastException
    , Comparator c가 array
    , value가 array , 가
    . equals() true
    , false . equals() 가

```

```

static boolean equals(boolean array1[ ], boolean array2[ ])
static boolean equals(byte array1[ ], byte array2[ ])
static boolean equals(char array1[ ], char array2[ ])
static boolean equals(double array1[ ], double array2[ ])
static boolean equals(float array1[ ], float array2[ ])
static boolean equals(int array1[ ], int array2[ ])
static boolean equals(long array1[ ], long array2[ ])
static boolean equals(short array1[ ], short array2[ ])
static boolean equals(Object array1[ ], Object array2[ ])

```

```

    , array1 array2

```

```

fill()
    . fill() 가 가

```

```

static void fill(boolean array[ ], boolean value)
static void fill(byte array[ ], byte value)
static void fill(char array[ ], char value)
static void fill(double array[ ], double value)
static void fill(float array[ ], float value)
static void fill(int array[ ], int value)
static void fill(long array[ ], long value)
static void fill(short array[ ], short value)
static void fill(Object array[ ], Object value)

```

```

value array . fill()

```

```

static void fill(boolean array[ ], int start, int end, boolean value)
static void fill(byte array[ ], int start, int end, byte value)

```

```

static void fill(char array[ ], int start, int end, char value)
static void fill(double array[ ], int start, int end, double value)
static void fill(float array[ ], int start, int end, float value)
static void fill(int array[ ], int start, int end, int value)
static void fill(long array[ ], int start, int end, long value)
static void fill(short array[ ], int start, int end, short value)
static void fill(Object array[ ], int start, int end, Object value)

```

```

    , value array start      end-1
      start가 end          IllegalArgumentException
    , start end가 (bound)   ArrayIndexOutOfBoundsException
    . sort()
sort()      가      가

```

```

static void sort(byte array[ ])
static void sort(char array[ ])
static void sort(double array[ ])
static void sort(float array[ ])
static void sort(int array[ ])
static void sort(long array[ ])
static void sort(short array[ ])
static void sort(Object array[ ])
static void sort(Object array[ ], Comparator c)

```

```

    , array      , c array
      Comparator      , Object
                          ClassCastException
sort()

```

```

static void sort(byte array[ ], int start, int end)
static void sort(char array[ ], int start, int end)
static void sort(double array[ ], int start, int end)
static void sort(float array[ ], int start, int end)
static void sort(int array[ ], int start, int end)
static void sort(long array[ ], int start, int end)
static void sort(short array[ ], int start, int end)
static void sort(Object array[ ], int start, int end)

```

```

static void sort(Object array[], int start, int end, Comparator c)
    , start          end-1          array
    , c array
Comparator
    start > end          IllegalArgumentException
    start < end-1      , ArrayIndexOutOfBoundsException

```

Arrays

```

//
import java.util.*;

class ArraysDemo {
    public static void main(String args[]) {

        //
        int array[] = new int[10];
        for(int i = 0; i < 10; i++)
            array[i] = -3 * i;

        //
        System.out.print("Original contents: ");
        display(array);
        Arrays.sort(array);
        System.out.print("Sorted: ");
        display(array);

        //
        Arrays.fill(array, 2, 6, -1);
        System.out.print("After fill(): ");
        display(array);

        //
        Arrays.sort(array);
        System.out.print("After sorting again: ");
        display(array);

        // -9

```

```

System.out.print("The value -9 is at location ");
int index =
    Arrays.binarySearch(array, -9);
System.out.println(index);
}

static void display(int array[]) {
    for(int i = 0; i < array.length; i++)
        System.out.print(array[i] + " ");
    System.out.println("");
}
}

```

Original contents: 0 -3 -6 -9 -12 -15 -18 -21 -24 -27
 Sorted: -27 -24 -21 -18 -15 -12 -9 -6 -3 0
 After fill(): -27 -24 -1 -1 -1 -1 -9 -6 -3 0
 After sorting again: -27 -24 -9 -6 -3 -1 -1 -1 -1 0
 The value -9 is at location 2

기존(legacy) 클래스와 인터페이스

, java.util
 (ad hoc)
 가 . 2 가 ,
 가 . ,
 가 ,
 가 ,
 가 ,
 가 ,
 가 ,
 Collections

java.util

Dictionary

Hashtable

Properties

Stack

Vector

Enumeration

Enum-

eration

Enumeration

Enumeration

()

Iterator

, Enumeration

(Vector Properties)

API

Enumeration

boolean hasMoreElements()

Object nextElement()

, hasMoreElements()

가

true

, 가

false

. nextElement()

enum-eration

Object

. , nextElement()

enumeration

enumeration

(cast)

(Vector)

Vector

. Vector

ArrayList

가

가 . Vector

,

가

2

, Vector가 ArrayList

List

가

. Vector

Vector()

Vector(int size)

Vector(int *size*, int *incr*)
 Vector(Collection *c*)

10 가 . size
 가 . size
incr 가 가 . 가 가
 . 2 가 . *c*
 가 ,
 가 ,
 가 ,
 가 가 ,
 가 .

Vector protected .

```
int capacityIncrement;
int elementCount;
Object elementData[ ];
```

가 capacityIncrement .
 elementCount . elementData .

List 가 , Vector 15-10

```
void addElement(Object element 가 .
    element)
int capacity()
Object clone()
boolean contains(Object element가 true
    element) , false .
```

void copyInto(Object array[])		array
Object elementAt(int index)	index	.
Enumeration elements()		enumeration .
void ensureCapacity(int size)	size	.
Object firstElement()		.
int indexOf(Object element)	element가 가	, -1 .
int indexOf(Object element, int start)	element가 start 가	, -1 .
void insertElementAt(Object element, int index)	index	element 가 .
boolean isEmpty()	가	true , true .
Object lastElement()		.
int lastIndexOf(Object element)	element가 가	, -1 .
int lastIndexOf(Object element, int start)	start element가 가	, -1 .
void removeAllElements()	가 0	가 ,
boolean removeElement (Object element)	element 가	가 , true false .
void removeElementAt(int index)	index	.
void setElementAt(Object element, int index)	index	element .
void setSize(int size)	가 가 가	size . 가 , 가 , null

```

int size()
String toString()
void trimToSize()

```

[15-10] Vector

```

Vector List, ArrayList
, addElement()
, elementAt()
, firstElement()
가, lastElement()
indexOf()
lastIndexOf()
, removeElement() removeElementAt()

Vector, Enumeration

```

```

//
import java.util.*;

class VectorDemo {
    public static void main(String args[]) {

        // 3 가 2
        Vector v = new Vector(3, 2);

        System.out.println("Initial size: " + v.size());
        System.out.println("Initial capacity: " + v.capacity());
        v.addElement(new Integer(1));
        v.addElement(new Integer(2));
        v.addElement(new Integer(3));
        v.addElement(new Integer(4));
    }
}

```

```

System.out.println("Capacity after four additions: " + v.capacity());
v.addElement(new Double(5.45));
System.out.println("Current capacity: " + v.capacity());
v.addElement(new Double(6.08));
v.addElement(new Integer(7));

System.out.println("Current capacity: " + v.capacity());
v.addElement(new Float(9.4));
v.addElement(new Integer(10));

System.out.println("Current capacity: " + v.capacity());
v.addElement(new Integer(11));
v.addElement(new Integer(12));

System.out.println("First element: " + (Integer)v.firstElement());
System.out.println("Last element: " + (Integer)v.lastElement());

if(v.contains(new Integer(3)))
    System.out.println("Vector contains 3.");

//
Enumeration vEnum = v.elements();

System.out.println("\nElements in vector:");
while(vEnum.hasMoreElements())
    System.out.print(vEnum.nextElement() + " ");
System.out.println();
}
}

```

```

Initial size: 0
Initial capacity: 3
Capacity after four additions: 5
Current capacity: 5
Current capacity: 7
Current capacity: 9
First element: 1
Last element: 12
Vector contains 3.

```

Elements in vector:
 1 2 3 4 5.45 6.08 7 9.4 10 11 12

2, Vector 가 .
 enumeration () ,
 enumeration .

```
//
Iterator vltr = v.iterator();

System.out.println("\nElements in vector:");
while(vltr.hasNext())
    System.out.print(vltr.next() + " ");
System.out.println();
```

enumeration ,
 enumeration , enumeration
 enumeration

(Stack)

Stack (last-in, first-out) Vector .
 Stack . Stack 15-11
 , Vector ,
 가 . , push() .
 , pop() .
 pop() EmptyStackException .
 , peek() .
 , empty() true . search() 가
 , 가
 pop() 가 .

```

boolean empty()                true ,
                               false .
Object peek()                  .
Object pop()                    , .
Object push(Object element)    element . element가 .
int search(Object element)     element . ,
                               (offset)가 . -1

```

[15-11] Stack

, Integer

```

//
import java.util.*;

class StackDemo {
    static void showpush(Stack st, int a) {
        st.push(new Integer(a));
        System.out.println("push(" + a + ")");
        System.out.println("stack: " + st);
    }

    static void showpop(Stack st) {
        System.out.print("pop -> ");
        Integer a = (Integer) st.pop();
        System.out.println(a);
        System.out.println("stack: " + st);
    }

    public static void main(String args[]) {
        Stack st = new Stack();

        System.out.println("stack: " + st);
        showpush(st, 42);
        showpush(st, 66);
    }
}

```

```

showpush(st, 99);
showpop(st);
showpop(st);
showpop(st);
showpop(st);
try {
    showpop(st);
} catch (EmptyStackException e) {
    System.out.println("empty stack");
}
}
}

```

(underflow)

EmptyStackException

가

```

stack: [ ]
push(42)
stack: [42]
push(66)
stack: [42, 66]
push(99)
stack: [42, 66, 99]
pop -> 99
stack: [42, 66]
pop -> 66
stack: [42]
pop -> 42
stack: [ ]
pop -> empty stack

```

(Dictionary)

Dictionary /

Map

(abstract)

, Dictionary

가

dictionary /

2

dictionary Map

Dictionary

Dictionary

15-12

가

```

put()
Enumeration . keys() elements()
.size() dictionary /
.isEmpty() dictionary가 true /
remove()

```

Enumeration elements()	dictionary	enumeration	.
Object get(Object key)	key 가	dictionary key가 , 가	.
boolean isEmpty()	dictionary가	true , false	.
Enumeration keys()	dictionary	enumeration	.
Object put(Object key, Object value)	dictionary dictionary key가 key가 key		.
Object remove(Object key)	dictionary key가		.
int size()	dictionary		.

[15-12] Dictionary



Dictionary

Map

```

Hashtable java.util Dictionary . , 2
Map Hashtable . , Hash- table
.Hashtable HashMap ,

```

HashMap, Hashtable / Hashtable
 가
 가
 Object hashCode() equals()
 hashCode()
 equals()
 hashCode() Hashtable 가
 String . String hashCode() equals()

Hashtable

Hashtable()
 Hashtable(int *size*)
 Hashtable(int *size*, float *fillRatio*)
 Hashtable(Map *m*)

size 가
size *fillRatio*
 가 0.0 1.0
 가
 , 0.75가
m
m 2 0.75가
 2 가 Hashtable Map
 가 , Hashtable 15-13

void clear()	(reset)
Object clone()	
boolean contains(Object value)	value
	true false

boolean containsKey(Object key)	key	가	true	false
boolean containsValue(Object value)	value	(Map 2 가 .)	true	false
Enumeration elements()		enumeration		
Object get(Object key)	key	가 key가 , 가		
boolean isEmpty()			true false	
Enumeration keys()		enumeration		
Object put(Object key, Object value)	가	가		
void rehash()				
Object remove(Object key)		가 , 가		
int size()				
String toString()				

[15-13] Hashtable

Hashtable

```
//
import java.util.*;
class HTDemo {
    public static void main(String args[]) {
        Hashtable balance = new Hashtable();
        Enumeration names;
```

```

String str;
double bal;

balance.put("John Doe", new Double(3434.34));
balance.put("Tom Smith", new Double(123.22));
balance.put("Jane Baker", new Double(1378.00));
balance.put("Todd Hall", new Double(99.22));
balance.put("Ralph Smith", new Double(-19.08));

//
names = balance.keys();
while(names.hasMoreElements()) {
    str = (String) names.nextElement();
    System.out.println(str + ": " + balance.get(str));
}

System.out.println();

// John Doe          1000
bal = ((Double)balance.get("John Doe")).doubleValue();
balance.put("John Doe", new Double(bal+1000));
System.out.println("John Doe's new balance: " + balance.get("John Doe"));
}
}

```

```

Todd Hall: 99.22
Ralph Smith: -19.08
John Doe: 3434.34
Jane Baker: 1378.0
Tom Smith: 123.22

```

```

John Doe's new balance: 4434.34

```

```

가
,
enumeration
가
, entrySet() keySet() Map

```

```

//
import java.util.*;

class HTDemo2 {
    public static void main(String args[]) {
        Hashtable balance = new Hashtable();
        String str;
        double bal;

        balance.put("John Doe", new Double(3434.34));
        balance.put("Tom Smith", new Double(123.22));
        balance.put("Jane Baker", new Double(1378.00));
        balance.put("Todd Hall", new Double(99.22));
        balance.put("Ralph Smith", new Double(-19.08));

        //
        Set set = balance.keySet(); //

        //
        Iterator itr = set.iterator();
        while(itr.hasNext()) {
            str = (String) itr.next();
            System.out.println(str + ": " + balance.get(str));
        }

        System.out.println();

        // John Doe          1000
        bal = ((Double)balance.get("John Doe")).doubleValue();
        balance.put("John Doe", new Double(bal+1000));
        System.out.println("John Doe's new balance: " + balance.get("John Doe"));
    }
}

```

Properties

Properties Hashtable

. Properties 가 String , String
 . Properties

```

        , Properties
System.getProperties() 가
Properties
    Properties defaults;
        Properties 가 . Properties
Properties()
Properties(Properties propDefault)
propDefault Properties 가
Properties Hashtable 가 , 15-14
        . Properties save() . save()
store()
    
```

String getProperty(String key)	key	. key가	가
String getProperty(String key, String defaultProperty)	Key	. key가	default-Property가
void list(PrintStream streamOut)	streamOut	.	
void list(PrintWriter streamOut)	streamOut	.	
void load(InputStream streamIn) throws IOException	streamIn	.	
Enumeration propertyNames()	enumeration	.	enumeration

```

Object setProperty(String key,      value key      . key
                    String value)  , key
                                   .(      , 2   가   .)

void store(OutputStream      streamOut      description
           streamOut, String description)
                                   .( 2   가   .)

```

[15-14] Properties

```

Properties      가      ,
               .      getProperty(name, default value)
               , getProperty()      .      name
               , default value가      . Properties
               Properties
               Properties      getProperty(foo)
foo      가      ,      Properties      foo
               .
               Properties      가
               가      .      Florida
               (      )      .

```

```

//
import java.util.*;

class PropDemo {
    public static void main(String args[]) {
        Properties capitals = new Properties();
        Set states;
        String str;

        capitals.put("Illinois", "Springfield");
        capitals.put("Missouri", "Jefferson City");
        capitals.put("Washington", "Olympia");
        capitals.put("California", "Sacramento");
        capitals.put("Indiana", "Indianapolis");
    }
}

```

```

// ( )
states = capitals.keySet(); // -
Iterator itr = states.iterator();

while(itr.hasNext()) {
    str = (String) itr.next();
    System.out.println("The capital of " +
        str + " is " +
        capitals.getProperty(str)
        + ".");
}

System.out.println();

//
str = capitals.getProperty("Florida", "Not Found");
System.out.println("The capital of Florida is " + str + ".");
}
}

```

The capital of Missouri is Jefferson City.
 The capital of Illinois is Springfield.
 The capital of Indiana is Indianapolis.
 The capital of California is Sacramento.
 The capital of Washington is Olympia.

The capital of Florida is Not Found.

```

Florida
    getProperty()
    ,
    , Properties
    , 가
    , 가
    Florida
    , Florida

```

```

//
import java.util.*;

class PropDemoDef {
    public static void main(String args[]) {
        Properties defList = new Properties();
        defList.put("Florida", "Tallahassee");
        defList.put("Wisconsin", "Madison");

        Properties capitals = new Properties(defList);
        Set states;
        String str;

        capitals.put("Illinois", "Springfield");
        capitals.put("Missouri", "Jefferson City");
        capitals.put("Washington", "Olympia");
        capitals.put("California", "Sacramento");
        capitals.put("Indiana", "Indianapolis");

        //
        states = capitals.keySet(); //
        Iterator itr = states.iterator();

        while(itr.hasNext()) {
            str = (String) itr.next();
            System.out.println("The capital of " + str + " is " +
                capitals.getProperty(str) + ".");
        }

        System.out.println();

        // Florida
        str = capitals.getProperty("Florida");
        System.out.println("The capital of Florida is " + str + ".");
    }
}

```

store()

load()

Properties

가

store() load()

Properties

가

```

        .          ,          Properties
        .
        ,
        .          .          ,          store()
load()          .          ,          phonebook.dat          가
        .          ,
        .          가          .          가가          ,
        가          .          가
        .

```

```

/*          */
import java.io.*;
import java.util.*;

class Phonebook {
    public static void main(String args[])
        throws IOException
    {
        Properties ht = new Properties();
        BufferedReader br =
            new BufferedReader(new InputStreamReader(System.in));
        String name, number;
        FileInputStream fin = null;
        boolean changed = false;

        // phonebook.dat
        try {
            fin = new FileInputStream("phonebook.dat");
        } catch(FileNotFoundException e) {
            //
        }

        /*          */
        try {
            if(fin != null) {
                ht.load(fin);
                fin.close();
            }
        }
    }
}

```

```

    } catch(IOException e) {
        System.out.println("Error reading file.");
    }

    //
    do {
        System.out.println("Enter new name" + " ('quit' to stop): ");
        name = br.readLine();
        if(name.equals("quit")) continue;

        System.out.println("Enter number: ");
        number = br.readLine();

        ht.put(name, number);
        changed = true;
    } while(!name.equals("quit"));

    //
    if(changed) {
        FileOutputStream fout = new FileOutputStream("phonebook.dat");

        ht.store(fout, "Telephone Book");
        fout.close();
    }

    //
    do {
        System.out.println("Enter name to find" + " ('quit' to quit): ");
        name = br.readLine();
        if(name.equals("quit")) continue;

        number = (String) ht.get(name);
        System.out.println(number);
    } while(!name.equals("quit"));
}
}

```

컬렉션 요약

가



가
,
가
,
(imagination)

.
. , TreeMap
. TreeSet
,
-