<<Collection>> 인터페이스

- 구조도 (책 참조)

- Set 인터페이스 특징 : 같은 데이터의 중복이 허용되지 않는다. 데이터 순서가 없다

- List 인터페이스 특징 : 같은 데이터 중복 허용, 데이터가 들어오는 순서대로 저장된다.

- Map 인터페이스 특징 : Key/Value 쌍으로 저장 , Key중복허용 안됨, Value 중복허용, 저장순서 없다. 검색속도가 빠르다.

Vector<E> 클래스의 주요 메소드

벡터는 가변 개수의배열이 필요할때 쓰임 – 스레드 간에 동기화 지원

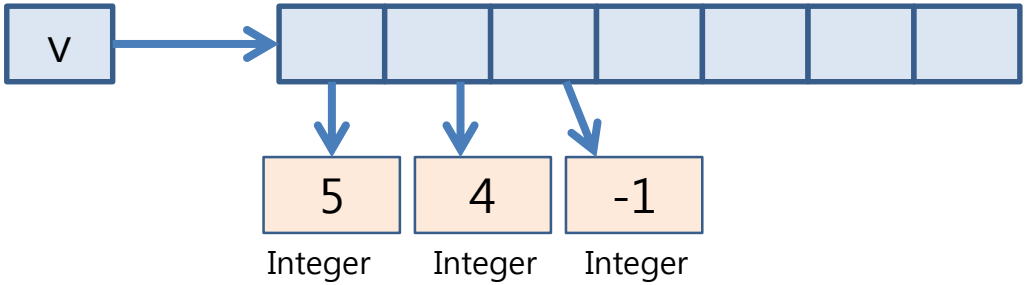
메소드	설명
boolean add(E e)	벡터의 맨뒤에 e추가
void add(int index, E element)	Index위치에 e 추가
boolean addAll(Collection<? extends E> c)	콜렉션 c의 모든 요소를 벡터 맨뒤에 추가
int capacity()	벡터의 현재 용량 추가
void clear()	벡터의 모든 요소 삭제
boolean contains(Object o)	벡터에 객체 o를 포함하고 있으면 true 리턴
E elementAt(int index)	Index의 요소 리턴
E get(int index)	인덱스이 요소 리턴
int indexOf(Object o)	객체 o와 같은 첫번째 인덱스 리턴 , 없으면 -1 리턴
boolean isEmpty()	벡터가 비어 있으면 true리턴
E remove(int index)	인덱스의 요소 삭제
boolean remove(Object o)	객체 o와 같은 첫번째 요소를 벡터에서 삭제
void removeAllElements()	벡터의 모든 요소를 삭제하고 크기를 0으로 만듦
int size()	벡터에 있는 요소의 개수를 리턴
Object[] toArray()	벡터에 있는 모든 요소를 포함하는 배열을 리턴

벡터생성

```
Vector<Integer> v = new Vector<Integer>(7)
```

요소삽입

```
v.add(5);
v.add(new Integer(4));
v.add(-1);
```



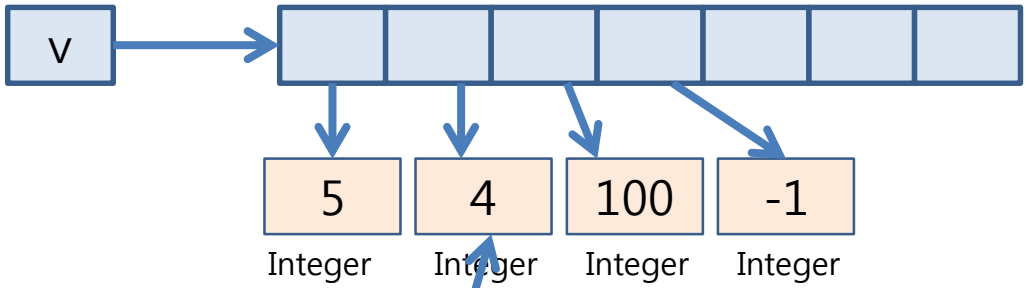
요소 개수

```
int n = v.size();
int c = v.capacity();
```

n=3
c=7

요소 중간 삽입

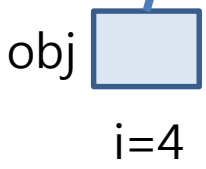
```
v.add(2, 100);
```




```
v.add(5, 100); //error v.size()보다 큰 곳에 삽입  안됨
```

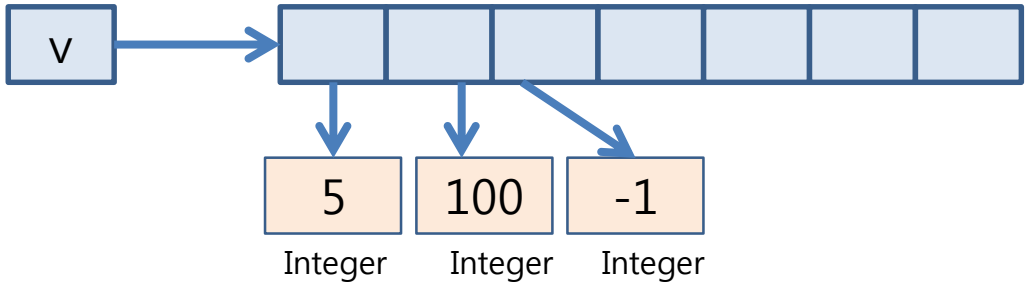
요소 얻어내기

```
Integer obj = v.get(1);
int i = obj.intValue();
```



요소 삭제

```
v.remove(1);
```




```
v.remove(4); // 오류 size()의 인덱스 보다 크다
```

마지막 요소

```
int last = v.lastElement();  // last = -1
```

모든 요소 삭제

```
v.removeAllElements();
```

```
import java.util.Vector;

public class VectorEx{

  public static void main(String[] args) {

     Vector<Integer> v = new Vector<Integer>();

     v.add(5);
     v.add(4);
     v.add(-1);
     v.add(2,100);
     System.out.println("벡터 내의 모든 요소 수: "+v.size());
     System.out.println("벡터 현재 용량: "+v.capacity());

     for(int i=0; i<v.size();i++) {
            int n = v.get(i);
             System.out.println(n);
     }
```

```
        int sum=0;

        for(int i=0; i<v.size() ;i++) {
            int n = v.elementAt(i);
            sum+=n;
        }

        System.out.println("벡터의 있는 정수의 합 : " + sum);


    } //main ()


} //class VectorEx
```

<예제>

(x,y) 한 점을 추상화한 Point클래스를 만들고 Point 클래스의 객체를 저장하는 벡터를 만들어 내용을 출력하는 클래스를 만들어라 (위치를 나타내는 포인트들을 저장하여 선을 만들 수 있다)

```
import java.util.Vector;

class Point {

  private int x, y;

  public Point(int x, int y) {
      this.x = x;
      this.y= y;
  }

  public String toString() {
      return " ("+x+", "+y+") ";
  }
```

```
public class PointVectorEx{

  public static void main(String[] args) {

    Vector<무엇을 넣을까요> v = new Vector<무엇을..>();

    v.add(어떻게 넣을까요);
    v.add(어떻게 넣을까요);
    v.add(어떻게 넣을까요);

    for(int i=0; i<v.size();i++) {
          무엇을 넣을까요= v.get(i);
          System.out.println(p); //p.toString()수행
    }
```

```
public class PointVectorEx{

  public static void main(String[] args) {

     Vector<Point> v = new Vector<Point>();

     v.add(new Point(3,2));
     v.add(new Point(-5, 20));
     v.add(new Point(30, -8));

     for(int i=0; i<v.size();i++) {
            Point p= v.get(i);
            System.out.println(p); //p.toString()수행
     }
```

예제
키보드로 부터 문자열 4개를 입력받아
ArrayList에 삽입하고 가장 긴 문자을 출력하라.

```
Import java.util.*;

Public class ArrayListEx {
    public static void main(String[] args) {
      // ArrayList a를 생성하라

       //Scanner 객체를 생성하라

     //키보드로 부터 문자열4번을 입력받고 ArrayList a에 저장하라

     //ArrayList에 들어 있는 모든 이름 출력

     //가장 긴 이름을 출력한다.
     //ArrayList 내의 첫번째 인덱스를 가장 긴 이름이라고 가정하고
     //다음번의 인덱스와 비교하여 긴 인덱스를 설정한다.

  }
}
```

```
Import java.util.*;

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

      ArrayList<String> a = new ArrayList<String>();
      Scanner scan = new Scanner(System.in);

      for(int i=0; i<4 ; i++) {
         System.out.print("문자를 입력하세요 :");
         String s = scan.next();
         a.add(s);
      }

      for(int i=0; i<a.size() ; i++) {
         String s = a.get(i);
          System.out.print(s+" ");
      }
```

```
    int lngIndx = 0;
    for(int i=1 ; i<a.size() ; i++) {
        if(a.get(lngIndx).length() < a.get(i).length()) // 이름 길이 비교
                    lngIndx = i ;
    }

   System.out.println("가장 긴 문자열은 : "+a.get(lngIndx));

  }
}
```

HashMap<K,V> 주요 메소드

메소드	설명
void clear()	HashMap의 모든 요소 삭제
boolean containsKey(Object key)	지정된 키(key)를 포함하고 있으면 true 리턴
boolean containsValue(Object value)	하나 이상의 키를 지정된 값에 매핑시킬 수 있으면 true 리턴
V get(Object key)	지정된 키에 매핑되는 값 리턴, 없으면 null 리턴
boolean isEmpty()	HashMap이 비어 있으면 true 리턴
Set<k> keySet()	HashMap에 있는 모든 키를 Set<k>로 리턴
V put(K key, V value)	키와 값을 매핑하여 HashMap에 저장
V remove(Object key)	지정된 키에 매핑된 값을 HashMap에서 삭제
int size()	HashMap에 포함된 요소의 개수 리턴

Iterator() , Enumeration()

```
import java.util.*;

public class IteratorEx {

    public static void main(String[] args) {
        // 정수 값만 다루는 제네릭 벡터 생성
        Vector<Integer> v = new Vector<Integer>();
        v.add(5); // 5 삽입
        v.add(4); // 4 삽입
        v.add(-1); // -1 삽입
        v.add(2, 100); // 4와 -1 사이에 정수 100 삽입

        // Iterator를 이용한 모든 정수 출력하기
        Iterator<Integer> it = v.iterator(); // Iterator 객체 얻기
        while(it.hasNext()) {
            int n = it.next();
            System.out.println(n);
        }
```

```
// Iterator를 이용하여 모든 정수 더하기
        int sum = 0;
        it = v.iterator(); // Iterator 객체 얻기
        while(it.hasNext()) {
            int n = it.next();
            sum += n;
        }
        System.out.println("벡터에 있는 정수 합 : " + sum);
    }
}
```

영어 단어와 한글 단어를 쌍으로 HashMap에 저장하고 영어 단어로 한글 단어를 검색하는 프로그램을 작성하라

```
import java.util.*;

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

        HashMap<String, String> dic = new HashMap<String, String>();

        dic.put("baby", "아기");
        dic.put("love", "사랑");
        dic.put("apple", "사과");

        // dic 컬렉션에 들어 있는 모든 (key, value) 쌍 출력
        Set<String> keys = dic.keySet(); // key 문자열을 가진 집합 Set 컬렉션 리턴
        Iterator<String> it = keys.iterator(); // key 문자열을 순서대로 접근할 수 있는
                                               // Iterator 리턴
        while(it.hasNext()) {
            String key = it.next();
            String value = dic.get(key);
            System.out.println("(" + key + "," + value + ")");
        }
```

```
 // 사용자로부터 영어 단어를 입력받고 한글 단어 검색
        Scanner scanner = new Scanner(System.in);
        for(int i=0; i<3; i++) {
            System.out.print("찾고 싶은 단어는?");
            String eng = scanner.next();
            System.out.println(dic.get(eng));
        }
    }
}
```

LinkedList<E>
- 요소들을 양방향으로 관리한다.
- Vector, ArrayList와 매우 유사

Collections 클래스 , Arrays 클래스
- static타입으로 객체를 생성할 필요는 없다
- sort(), reverse(), max(), min(), binarySearch() 메소드등이 있다.

* 두 클래스를 활용한 예를 살펴보자

```
import java.util.*;

public class CollectionsEx {
   static void printList(LinkedList<String> l) {

            Iterator<String> iterator = l.iterator(); // Iterator 객체 리턴
            while (iterator.hasNext()) { // Iterator 객체에 요소가 있을 때까지 반복
            String e = iterator.next(); // 다음 요소 리턴
            String separator;
            if (iterator.hasNext())
            separator = "->"; // 마지막 요소가 아니면 → 출력
            else
            separator = "₩n"; // 마지막 요소이면 줄바꿈
            System.out.print(e+separator);
            }
   }
```

```
 public static void main(String[] args) {
        LinkedList<String> myList = new LinkedList<String>(); // 빈 리스트 생성
        myList.add("트랜스포머");
        myList.add("스타워즈");
        myList.add("매트릭스");
        myList.add(0,"터미네이터");
        myList.add(2,"아바타");

        Collections.sort(myList); // 요소 정렬
        printList(myList); // 정렬된 요소 출력

        Collections.reverse(myList); // 요소의 순서를 반대로
        printList(myList); // 요소 출력

        int index = Collections.binarySearch(myList, "아바타") + 1; // "아바타"의 인
        System.out.println("아바타는 " + index + "번째 요소입니다.");
        }
}
```

풀어봅시다.

HashMap을 이용하여 전화번호 관리 프로그램을 만들어 보자

PhoneBook클래스 작성 –전화번호 정보 표현
전화번호 정보 – 이름 , 주소 , 전화번호
프로그램 메뉴 – 삽입,삭제,찾기 , 전체보기, 종료

```
전화번호 프로그램을 시작합니다.
삽입(0), 삭제(1), 찾기(2), 전체보기(3), 종료(4) >>> 0
이름>>노광현
주소>>서울시 관악구
전화번호>>010777777
삽입(0), 삭제(1), 찾기(2), 전체보기(3), 종료(4) >>> 1
삭제번호>>0103333333
--전화번호가 등록되지 않았습니다.
삽입(0), 삭제(1), 찾기(2), 전체보기(3), 종료(4) >>> 2
찾는전화번호>>>0010777777
--이름 : 노광현
--주소 : 서울시 관악구
```

```
import java.util.HashMap;
import java.util.Scanner;

class PhoneBook{
    String name;
    String phoneNum;
    String addr;

    PhoneBook(String name, String num, String addr){
        this.name =name;
        phoneNum = num;
        this.addr = addr;
    }

    @Override
    public String toString(){
            return  name+ "    " +addr;
    }

}
```

```
class PhoneBooks {

   HashMap<String, PhoneBook> books = new HashMap<String, PhoneBook>();

  void insert(String num, String name, String addr){
      books.put(num, new PhoneBook(name,num,addr));
  }

  void delete(String num){
     if(books.containsKey(num)) books.remove(num);
     else System.out.println("전화번호부에 등록되지 않았습니다.");
  }

  void search(String num) {
     if(books.containsKey(num)) System.out.println(books.get(num));
    else System.out.println("전화번호부에 등록되지 않았습니다.");
   }

  void allPrint(){
     System.out.println("    전화번호부 내용    ");
     System.out.println(books);
   }
}
```

```
public class TestPhonBooks {

   public static void main(String[] args) {
      System.out.println("전화번호부 관리 프로그램입니다");
      Scanner in = new Scanner(System.in);
      int menu_num;

      PhoneBooks phonebooks = new PhoneBooks();

     do{
         System.out.print("삽입(0), 삭제(1), 찾기(2), 전체보기(3), 종료(4) >>> ");
         menu_num = in.nextInt();

           switch(menu_num) {
           case 0 :System.out.print("이름 >>");
                   String name =in.next();
                   System.out.print("주소 >>");
                   String addr = in.next();
                   System.out.print("전화번호 >>");
                   String num = in.next();

                   phonebooks.insert(name, addr, num);
                   break;
```

```
            case 1: System.out.print("삭제할  전화번호 >>");
                    String del_num = in.next();
                    phonebooks.delete(del_num);
                 break;
            case 2:System.out.print("전화번호 >>");
                     String serch_num = in.next();
                     phonebooks.search(serch_num);
                     break;
            case 3:phonebooks.allPrint();
                break;
            }

    }while(menu_num != 4);

    System.out.println("프로그램 종료합니다.");

    } //main

} //class
```