20-8-26

(1) ArrayList클래스 : 배열을 위한 컬렉션

```
package a20_8_26;

import java.util.ArrayList;
import java.util.Collections;

//ArrayList  배열
public class collection1 {

	public static void show(ArrayList<Integer> a) {
		for(int num : a) {
			System.out.println(num);
		}
	}
	public static void main(String[] args) {
		ArrayList<Integer> a=new ArrayList<Integer>();
		a.add(10);
		a.add(20);
		a.add(30);
		a.add(20);
		a.add(20);
		a.remove(0);    //0번째를 삭제
		a.remove(new Integer(20));   //20이라는 정수를 삭제
		//a.remove(new Integer(30));
		a.set(0, 100);
		a.set(1, 200);
		show(a);
		System.out.println("===========");
		Collections.sort(a);    //Collections클래스에서 지원되는 sort메소드
		show(a);
	}
}
```

(2) Vector 클래스: 배열을 위한 컬렉션

```
package a20_8_26;

import java.util.Vector;

//Vector 배열
public class collection2 {

	public static void show(Vector<String> vec) {
		System.out.println("==============");
		for(String name : vec) {
			System.out.println(name);
		}
	}

	public static void main(String[] args) {
		Vector<String> vec=new Vector<String>(2,2);
		//기본2개할당받고, 꽉차면 2씩증가
		vec.add("홍길동");
		System.out.println(vec.size()+","+vec.capacity());

		vec.add("이순자");
		System.out.println(vec.size()+","+vec.capacity());

		vec.add("이기자");
		System.out.println(vec.size()+","+vec.capacity());

		vec.addElement("최고야");       //추천
		show(vec);

		vec.remove(0);
		vec.removeElement("최고야");   //추천
		show(vec);
	}

}
```

(3) Stack 클래스 : LIFO 구조 (push(), pop())

```
package a20_8_26;

import java.util.Stack;
```

```
import java.util.Vector;

//스택 stack :LIFO(last in first out)
public class collection3 {

	public static void show(Stack<String> name) {
		System.out.println("==============");
		while(!name.isEmpty()) {
			System.out.println(name.pop());
		}
	}

	public static void main(String[] args) {
		Stack<String> name=new Stack<String>();
		Stack<Integer> jumsu=new Stack<Integer>();
		name.add("홍길동");//비추천
		name.push("이기자"); //추천
		name.push("최고야");
		show(name);
	}

}
```

(4) Iterator 인터페이스, Enumeration 인터페이스는 순차접근을 위한 집합

```
package a20_8_26;

import java.util.Enumeration;
import java.util.Iterator;
import java.util.Vector;

//순차접근을 위한 집합 Iterator(이터레이터)
//순차접근을 위한 집합 Enumeration(이뉴머레이션)
//Vector나 ArrayList, LinkedList 에서 사용할것
public class collection4 {

	public static void main(String[] args) {
		Vector<String> name=new Vector<String>();
		name.addElement("최명실");
		name.addElement("최수지");
		name.addElement("최진실");
		name.addElement("최순실");

		Iterator<String> item=name.iterator();  //item집합
		while(item.hasNext()) {
			System.out.println(item.next());
		}

		System.out.println("==============");

		Enumeration<String> item2=name.elements();
		while(item2.hasMoreElements()) {
			System.out.println(item2.nextElement());
		}

	}

}
```

(5) Map : key, value를 한쌍으로 구성한다.

```
package a20_8_26;

import java.util.Hashtable;
import java.util.Set;

//map(맵) : key,value를 한쌍으로 구성된 것 -->순차구조x
//맵 종류중에서 Hashtable 클래스 활용
public class collection5 {

	public static void main(String[] args) {
		Hashtable<String,String> ht=new Hashtable<String,String>();
		ht.put("홍길동", "대구,010-939-997,18");
		ht.put("이순자", "인천,010-939-997,18");
		ht.put("이기자", "부산,010-939-997,18");
		ht.put("허준", "서울,010-939-997,18");

		Set<String> name=ht.keySet();
		for(String item : name) {
			System.out.println(item +":" + ht.get(item));
```

```
                        }

                        System.out.println("================");
                        System.out.println(ht.get("허준"));   //허준의 값

            }

}
```