Listas: Interface List<E>

Métodos

- boolean **add**(E elemento): añade elemento
- void **add**(int posicion, E elemento): añade elemento en posición
- void **clear**(): elimina todos los elementos
- boolean **contains**(E elemento): comprueba si está un elemento
- boolean **equals**(Object x): compara
- E **get**(int posicion): devuelve el elemento de una posición
- int **indexOf**(E elemento): devuelve la posición de un elemento
- boolean **isEmpty**(): si está vacío
- Iterator<E> iterator()
- E **remove**(int posicion): elimina de una posición y devuelve elemento
- boolean **remove**(E elemento): elimina elemento
- E **set**(int posicion, E elemento): Reemplaza elemento de una posición
- int **size**(): número de elementos

Implementaciones

- ArrayList<E>
- LinkedList<E>
- Vector<E>

Métodos específicos LinkedList

- E **getFirst**(): devuelve el primer elemento
- E **getLast**(): devuelve el último elemento
- E **removeFirst**(): borra y devuelve el primero
- E **removeLast**(): borra y devuelve el último
- void **addFirst**(E elemento): añade al principio
- void **addLast**(E element): añade al final

Ejemplo

```
List<String> l = new ArrayList<String>() ;
l.add("hola");
l.add("adios");
l.add("ciao");
l.add("bye");
```

```
l.add(2, "bye");
System.out.println(l.size()); // Devuelve 5
System.out.println(l.get(0)); // Devuelve hola
System.out.println(l.get(1)); // Devuelve adios
System.out.println(l.get(2)); // Devuelve bye
System.out.println(l.get(3)); // Devuelve ciao
System.out.println(l.get(4)); // Devuelve bye
    for (String v: l)
System.out.print(v);          // Imprime: holaadiosbyeciaobye
for (int i=0; i<l.size(); i++) {
    String v = l.get(i);
    System.out.print(v);
}                             // Imprime: holaadiosbyeciaobye
Iterator<String> it = l.iterator();
while(it.hasNext()) {
    String v = it.next();
    System.out.print(v);
}                             // Imprime: holaadiosbyeciaobye
```

Conjuntos: Interface Set<E>

Métodos

- boolean **add**(E elemento)
- void **clear**()
- boolean **contains**(E elemento)
- boolean **equals**(Object x)
- boolean **isEmpty**()
- Iterator<E> iterator()
- boolean **remove**(E elemento)
- int **size**()

Implementaciones

- HashSet<E>
- LinkedHashSet<E>
- TreeSet<E>

Ejemplo

```
Set<String> s = new HashSet<String>() ;
s.add("hola");
s.add("adios");
s.add("ciao");
```

```
s.add("bye");
s.add("bye");
s.add("ciao");
System.out.println(l.size()); // Devuelve 4
for (String v: s)
    System.out.print(v);            // Imprime: holaadiosbyeciao
// (puede ser en otro orden)
Iterator<String> it = s.iterator();
while(it.hasNext()) {
    String v = it.next();
    System.out.print(v);
    procesa (dato);
}
```

Interface Map<K,V>

Métodos

- void **clear**()
- boolean **containsKey**(Object clave)
- boolean **containsValue**(Object valor)
- boolean **equals**(Object x)
- V **get**(Object clave)
- boolean **isEmpty**()
- Set<K> **keySet**()
- V **put**(K clave, V value)
- V **remove**(Object clave)
- int **size**()

Implementaciones

- HashMap<K,V>
- LinkedHashMap<K,V>
- TreeMap<K,V>
- Hashtable<K,V>

Ejemplo

```
Map<String,String> m = new HashMap <String,String>();
m.put("Pedro", "becario");
m.put("Pablo", "aprendiz");
m.put("Maria", "jefe");
```

```
m.put("Jorge", "desempleado");
m.put("Pablo", "empleado");
System.out.println(m.size()); // Devuelve 4
for (String k: m.keySet()) {
    String v = m.get(k);
    System.out.println(k + " = " + v);
}                              // Imprime: Pedro = becario
                               // Pablo = empleado
                               // Maria = jefe
                               // Jorge = desempleado
                               // (el orden puede variar)
for (Iterator<String> it=m.keySet().iterator(); it.hasNext();)
{
    String k = it.next();
    String v = m.get(k);
    System.out.println(k + " = " + v);
}
```

Iteradores

Clase Iterator<E>

Métodos

- boolean **hasNext**()
- E **next**()
- void **remove**()

Ejemplo:

Ver ejemplos en listas y conjuntos.