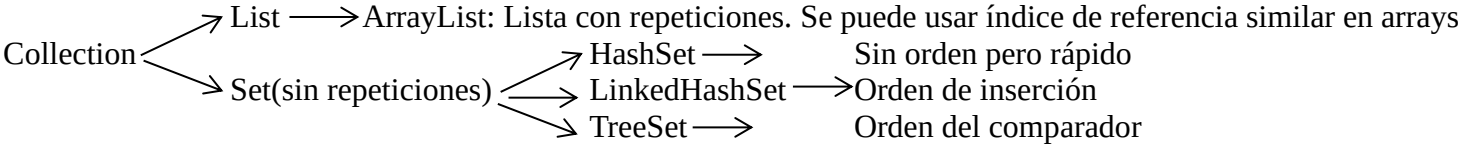
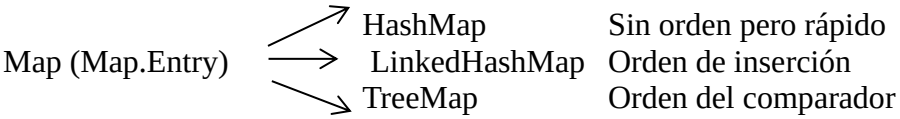
# Colecciones

import java.util.*;




Collection (add(objeto), remove(objeto), size( ), isEmpty(objeto), contains(elemento), clear( ), bol addAll(collection), bol removeAll(Collection), Iterator itereator(), obj[] toArray())

Iterator (elem next(), bol hasNext(), remove()→ del último elemento)

List (add(posicion, objeto), get(indice), set(indice,objeto), remove(indice), indexOf(objeto), lastIndexOf(objeto), sublist(indice_menor, indice_mayor), listIterator())

ArrayList → Lista simple

LinkedList → Lista doblemente enlazada (getFirst(), addFirst(obj), removeFirst(obj), getLast(), addLast(obj), removeLast(obj))

Set → colección sin duplicados. Tienen que definir los métodos equals y hashCode. (if obj instanceof Tipo)

HashSet → elemento en desorden

LinkedHashSet → elementos en el orden de inserción

TreeSet → arbol elementos ordenados. Debe implementar Comparable ( método compareTo). first(), last(),

Map<key, valor> → como un doble array, la clave indica elemento a coger. get(key), put(key, valor), containsKey(key), containsValue(valor), keyset() → lista claves, values() → lista de valores, entrySet → devuelve Map.Entry<k,v>, size(), clear()

Map.Entry<k,v> → objeto clave-valor getKey(), getValue(), setValue(v)

HashMap → elementos en desorden

LinkedHashMap → elementos en el orden de inserción

TreeMap → arbol elementos ordenados. Debe implementar Comparable

Collections → hacer diversas operaciones sobre elementos de tipo collections. Collections.sort(elemento) → debe implementar Comparable, Collections.reverse(elemento)

**Definición**

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

**Recorrido**

```
for (Map.Entry <k, v> elemento : objMapa.entrySet()) //recorrido map con un for each
        System.out.println("Clave: " + elemento.getKey() + ", valor: " + elemento.getValue());
```