Containere

java.util.*

Interface Collection

ArrayList

add(obj) / addAll(Collection)

size()

contains(obj)

se folosește equals pt. determinare

retainAll(Coll) - păstrează doar intersecția

remove(obj) -> (prima apariție)

clear()

obj[] = .toArray(obj[])

equals

ArrayList<Book> ---

Interface List

get (index)
set (index, Obj)
lastIndexOf (Obj)

Interface Collection

Iterator iterator() ⟵ boolean hasNext()
Obj next()
void remove()

Iterator it = bookList.iterator()

```
while ( it.hasNext() )
((Book) it.next()).setPages(10);
```

~~it = new Iterator()~~

- LinkedList (implements List)
  - ↗ eff for inserting / removing
  - ↘ eff for random access

+ multe alte tipuri de liste

Set ← HashSet, TreeSet

- HashSet — add (nevoie equals)

(verifică egalitatea de hashCode)

x.equals(y) → x.hashCode == y.hashCode

păstrează doar valorile unice

```
public int hashCode()
{
    return myValue;
}
```

- **LinkedHashSet** -> păstrează ordinea de inserție a elementelor
- **TreeSet** -> ordonate „natural"

string.compareTo(Str) $\begin{cases} -1 \\ 0 \\ 1 \end{cases}$
(ord. lexicografică)

```
MyClass implements Comparable<MyClass>
{
    public int compareTo(MyClass o);
}
```

- **HashMap<Obj, Obj>** implements Map

put(key, val) key value

put(key, val == null ? 1 : val + 1)

val = mp.get(key)

~~<int, int>~~
<Integer, ...>

- TreeMap -> sortate

Variable number of arguments

void add (MyStringElement... elem)

set.add(m1, m2, m3)

Clone (din clasa Object)

class implements Clonable
-> doar atunci este permisă metoda

interfață de marcaj

! atenție la Aliasing
-> se clonează valorile câmpurilor
(se copiază referințele)

Shallow Cloning

Deep Cloning

-> se copiază doar valorile (fiecare)

-> se clonează fiecare referință