

STREAM PIPELINE



```
graph TD; A[Creación] --> B[personas.stream()]; B --> C[Operaciones intermedias]; C --> D[.map(it -> it.getNombre())]; D --> E[Operación terminal]; E --> F[.collect(Collectors.toList());]
```

The code snippets are mapped to the stages of the Stream Pipeline as follows:

- Creación** points to `personas.stream()`
- Operaciones intermedias** points to `.map(it -> it.getNombre())`
- Operación terminal** points to `.collect(Collectors.toList());`

Stream pipelines

Creación

Collection stream()

```
List<Persona> personas = Arrays.asList(juan, antonia);  
Stream<Persona> s1 = personas.stream();
```

Arrays.asStream()

```
Persona[] personas = {juan, antonia};  
Stream<Persona> s2 = Arrays.stream(personas);
```

Utilidades de stream

```
Stream<Persona> s3 = Stream.of(juan, antonia);
```

otras APIs

```
Stream<String> s4 = Files.lines("fichero.txt");
```


Operaciones intermedias principales

- Convertir
- Filtrar
- Ordenar

- 
- Convertir
 - Filtrar
 - Ordenar

```
Stream<String> nombres = personas.map(it -> it.getNombre());
```

String

Persona

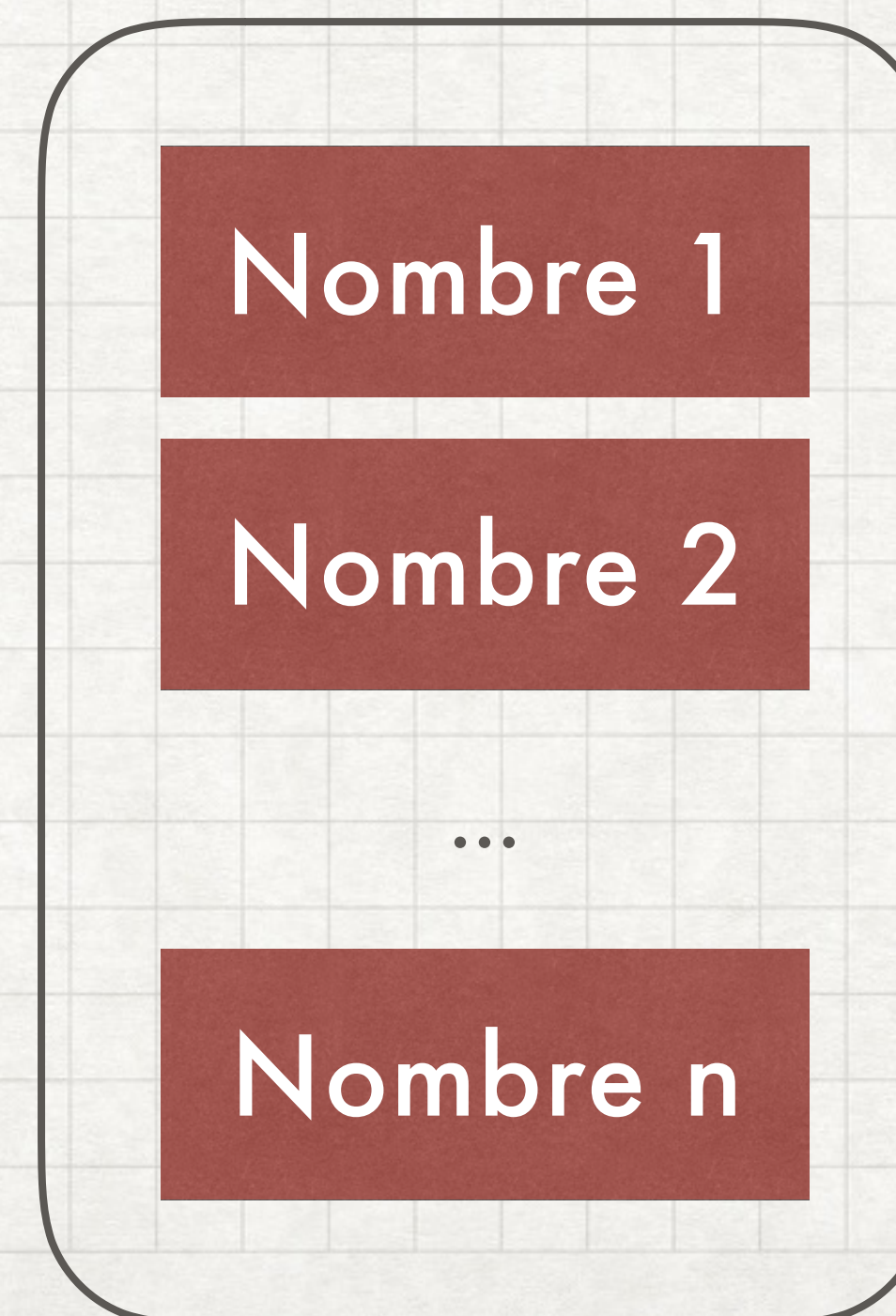
$f : \text{Persona} \rightarrow \text{String}$

Function<Persona, Nombre>

Function<? super Persona, ? extends String>

- 
- Convertir
 - Filtrar
 - Ordenar

```
Stream<String> nombres = personas.map(it -> it.getNombre());
```



- Mismo número
- Mismo orden
- Distinto tipo

- Convertir
- ➔ • Filtrar
- Ordenar

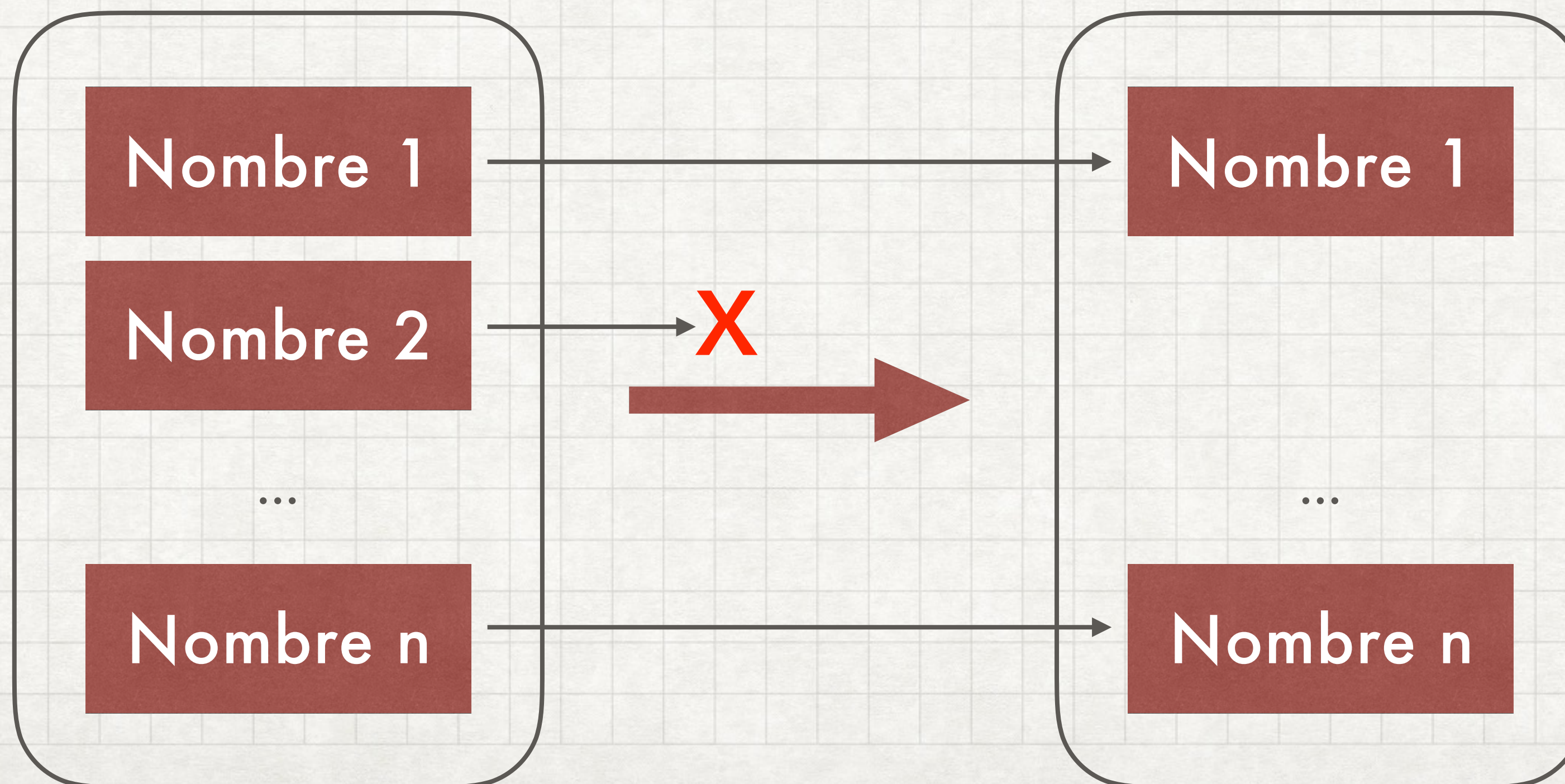
```
Stream<String> nombresA = nombres.filter(it -> it.startsWith("A"));
```

$f : \text{String} \rightarrow \text{boolean}$
`Predicate<String>`

`Predicate<? super String>`

- Convertir
- ➔ • Filtrar
- Ordenar

```
Stream<String> nombresA = nombres.filter(it -> it.startsWith("A"));
```



- Distinto número
- Mismo orden
- Mismo tipo

- Convertir
- Filtrar
- ➔ • Ordenar

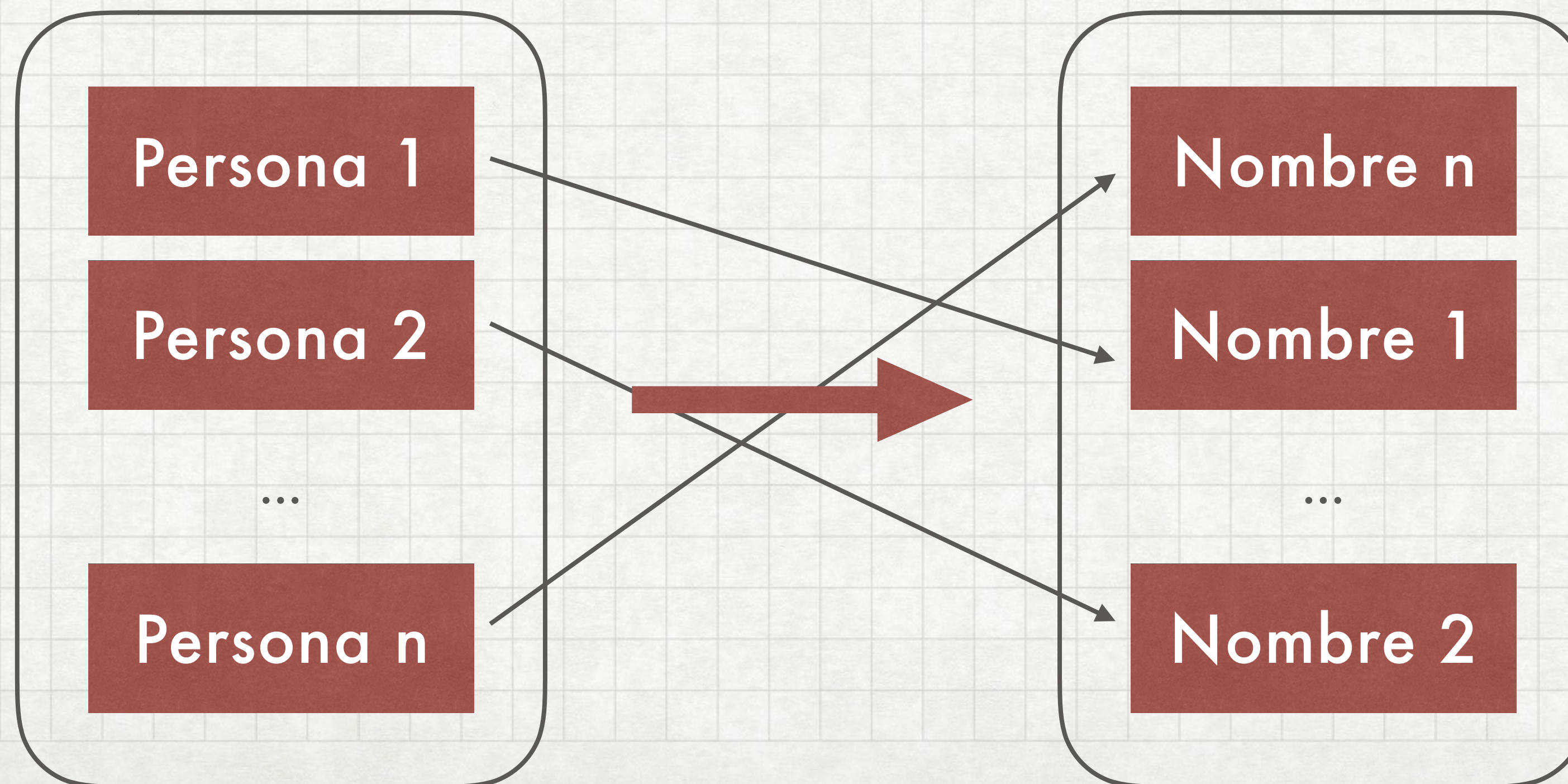
```
nombres.sorted((o1, o2) -> o2.length() - o1.length())
```

Comparator<? super String>

```
nombres.sorted()
```


- Convertir
- Filtrar
- ➔ • Ordenar

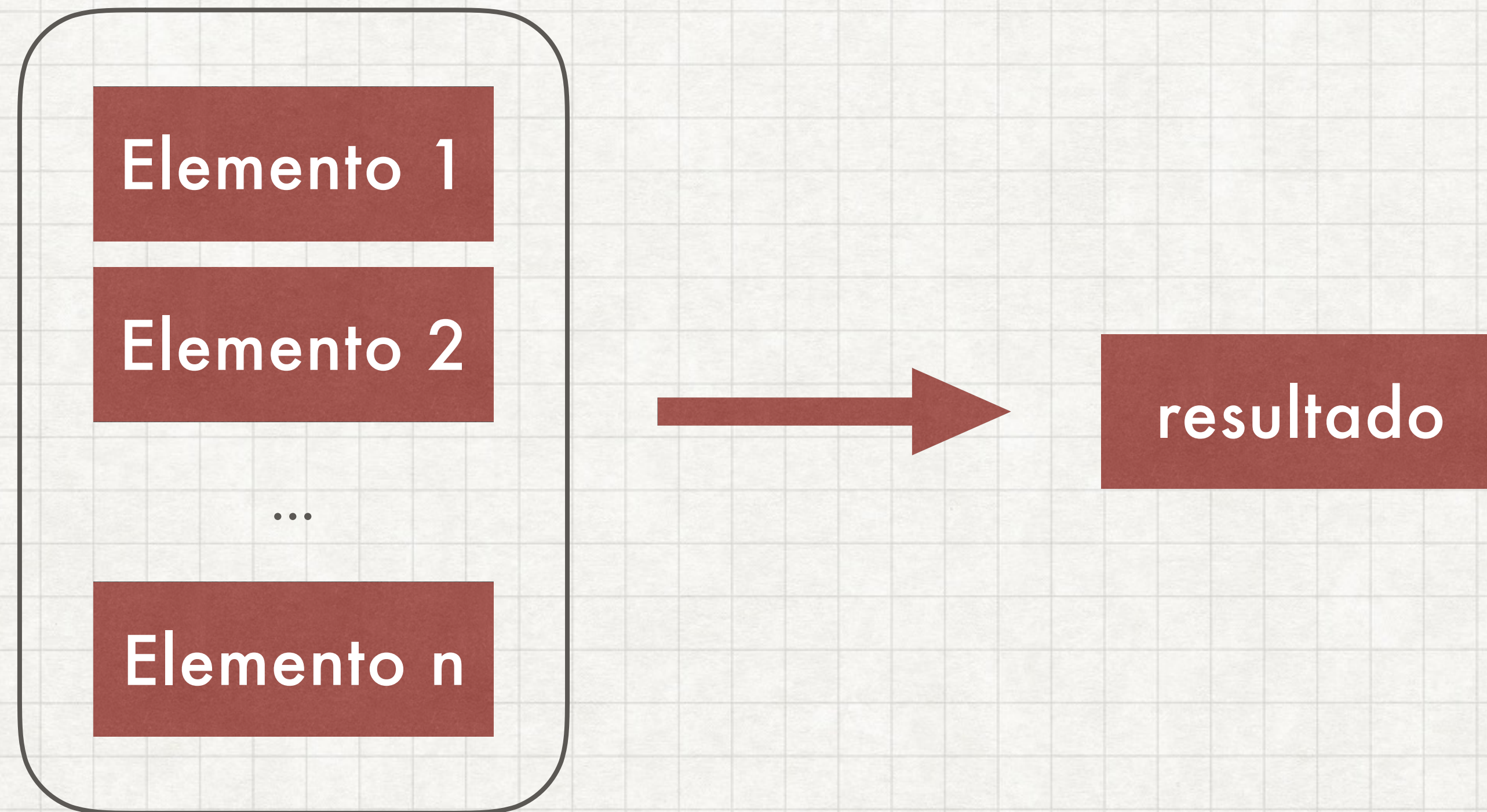
`nombres.sorted((o1, o2) -> o2.length() - o1.length())`



- **Mismo número**
- **Distinto orden**
- **Mismo tipo**

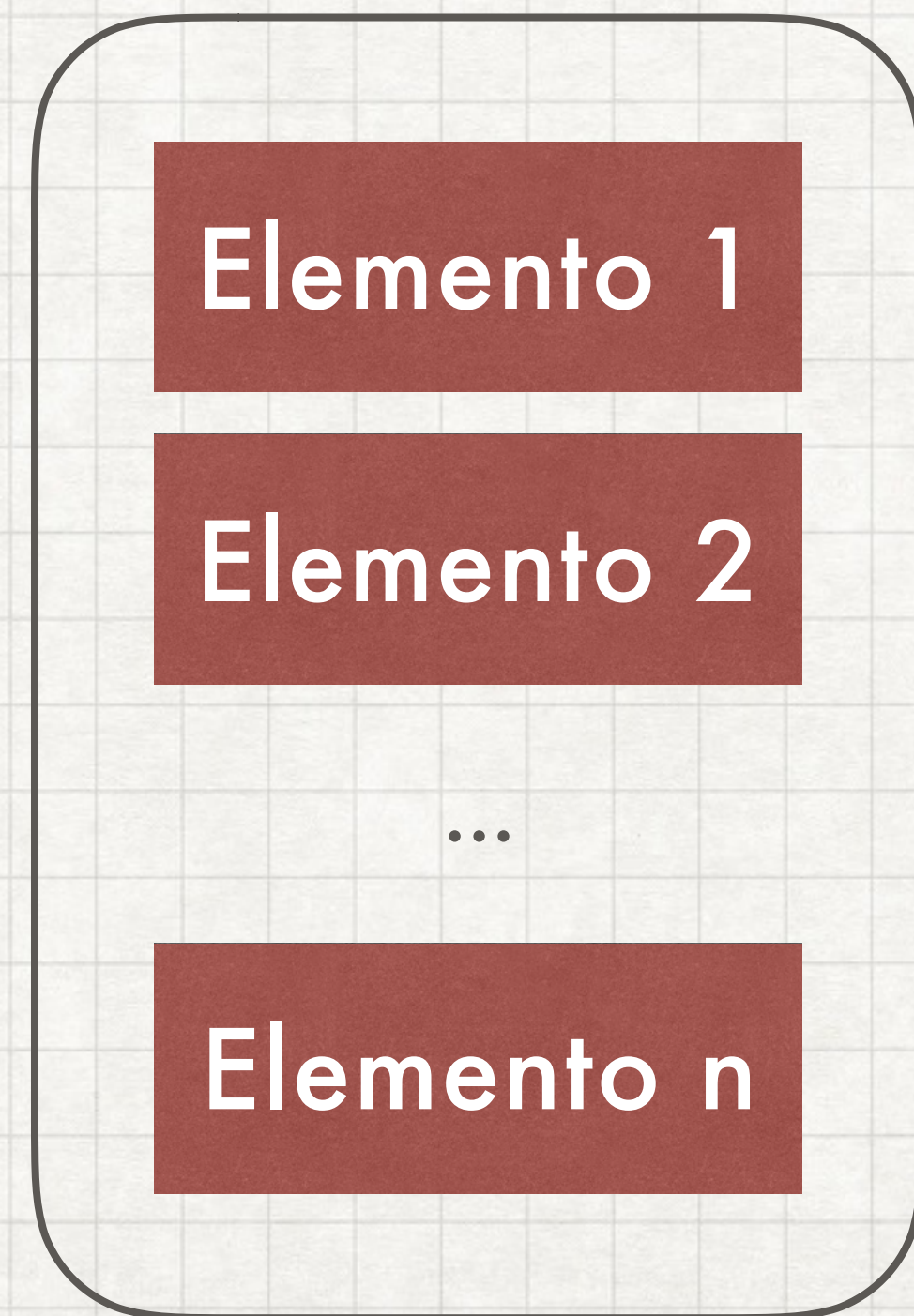
Operación terminal

reduce



reduce

"ingredientes"



Elemento
identidad

$f : a, b \rightarrow c$

ejemplo:

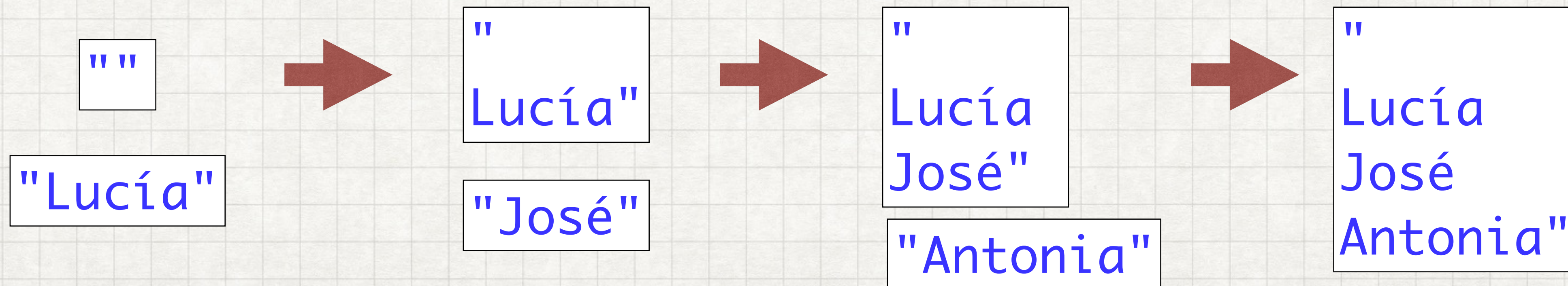
listado de elementos

""

$(a, b) \rightarrow a + "\backslash n" + b$

reduce

```
String todos = Stream.of("Lucía", "José", "Antonia")  
                    .reduce("", (a, b) -> a + "\n" + b);
```



stream pipeline

Creación

```
String todos = listaPersonas.stream()  
    .map(it -> it.getNombre())  
    .sorted( (o1, o2) -> o1.length() - o2.length())  
    .map( it -> "\"" + it + "\"")  
    .reduce("", (a, b) -> a + "\n" + b);
```

Operaciones intermedias

Operación terminal