

## Repasando colecciones

Con el System Browser de Pharo, analice y compare el comportamiento de los mensajes enunciados abajo para cada una de estas clases: `OrderedCollection`, `SortedCollection`, `Array`, `Dictionary`, `Bag` y `Set`.

<code>#add:</code>	<code>#at:</code>	<code>#at:put:</code>
<code>#size</code>	<code>#do:</code>	<code>#detect:</code>
<code>#select:</code>	<code>#collect:</code>	<code>#reject:</code>
<code>#inject:into:</code>	<code>#includes:</code>	<code>#isEmpty</code>

Responda a las siguientes preguntas:

1. ¿Es posible que algunos mensajes no sean aplicables para algunas colecciones? Por ejemplo, ¿Se le puede enviar el mensaje `#add:` a un `Array`? ¿Y a un `Set`? ¿Se le puede enviar el mensaje `#at:` y `#at:put:` a un `Set`?
2. En respuesta al mensaje `#select:`, ¿qué retorna un `Array`? ¿Y un `Dictionary`? ¿Y una `SortedCollection`?
3. En respuesta al mensaje `#size`, ¿qué retorna un `Array` creado con `Array new:10`? ¿Qué retorna una `OrderedCollection` creada con `OrderedCollection new:10`?
4. ¿Como se averigua la posición de un elemento en un `Array`? ¿Y la del primer elemento que `□` cumple una condición? ¿Es posible hacerlo en un `Set`?
5. Indique la diferencia entre `#detect:` y `#detect:ifNone:`. ¿Para qué sirve el bloque que se `□` envía como parámetro en `#ifNone:`?
6. ¿Cómo crea una `SortedCollection` para contener instancias de `String` ordenadas por `□` tamaño? ¿Cómo crea una `SortedCollection` para contener instancias de `String` ordenadas alfabéticamente?
7. ¿Cómo consigue los elementos en un `Array` eliminado las repeticiones?
8. ¿Cuál es el problema con la siguiente expresión si `col` es un `Set` con elementos? ¿Y si fuera una `OrderedCollection`? `□`

a. `col do: [ :each | col remove: each]`

9. ¿Cuál es el efecto de enviar el mensaje `#add:` con `nil` como parámetro a una `OrderedCollection`? ¿Por qué?
10. ¿Qué diferencia hay entre los mensajes `#includes:` y `#contains:`?

### `#do:` , `#collect:` , `#select:` , `#detect:ifNone:` , `#sumNumbers:`

Conocer los iteradores de colecciones nos ahorra mucho tiempo y evita que reinventemos la rueda.

Asumiendo que "productos" es un variable que apunta a una colección de objetos que entienden los siguientes mensajes: `#marca`, `#modelo`, `#precio`, `#esNuevo` (que retorna verdadero o falso); escriba expresiones para:

- 1) Obtener una nueva colección en la que solo hay productos nuevos.
- 2) Obtener una colección con los nombres de todos los productos, ordenada alfabéticamente.
- 3) Obtener una colección con todos los productos que no son nuevos, ordenada por precio del producto.
- 4) Actualizar el precio de todos los productos para que sea un 10% más caro.
- 5) Obtener el precio total (suma) de todos los productos de la colección.
- 6) Encontrar un producto cuya marca sea: 'Samsung'.
- 7) Obtener la lista de las marcas de productos en la colección, pero sin repeticiones.
- 8) Siendo que "unProducto" es una variable que apunta a un producto, escriba una expresión para verificar si unProducto está dentro de la colección "productos".