Xquery FLWOR: ejercicios 3

Partiendo del siguiente documento libros.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="COOKING">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="CHILDREN">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="WEB">
    <title lang="en">XQuery Kick Start</title>
    <author>James McGovern</author>
    <author>Per Bothner</author>
    <author>Kurt Cagle</author>
    <author>James Linn</author>
    <author>Vaidyanathan Nagarajan</author>
    <year>2003</year>
    <price>49.99</price>
  </book>
  <book category="WEB">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

- Mostrar la suma total de los precios de los libros con la etiqueta "total".

  ```
  <total>
        { sum(doc("libros.xml")/bookstore/book/price) }
  </total>
  ```

  Entre etiquetas total, calculamos la suma de toda la tupla de precios, como he visto en otros ejercicios, para hacer el cálculo de una función de agregación, no

necesitamos realizar un for, aunque también podría hacerse con uno y sumar el total de los precios de la tupla en el return.

- Mostrar cada uno de los precios de los libros, y al final una nueva etiqueta con la suma de los precios.

```
<libros>
	{
		for $libro in doc("libros.xml")/bookstore/book
		return $libro/price
	}
	<total>
	{
		sum(doc("libros.xml")/bookstore/book/price)
	}
	</total>
</libros>
```

Entre etiquetas “libros” hacemos dos consultas FLWOR, cada consulta debe ir entre llaves, la primera nos devuelve el precio de todos los libros, la segunda es como el ejercicio anterior, la suma de la tupla completa de precios.

- Mostrar el título y el número de autores que tiene cada libro en etiquetas diferentes.

```
for $libro in doc("libros.xml")/bookstore/book
return
	<libro>
		{ $libro/title }
		<autores> { count( $libro/author) } </autores>
	</libro>
```

A partir de un for recuperamos todas las tuplas de los libros.
En la cláusula return creamos dos etiquetas “libro” y dentro ponemos el títulos, y luego entre etiquetas “autores”, aplicamos la función “count” para utilizar la función de agregación que se encarga de contar los elementos “author”.

- Mostrar en la misma etiqueta el título y entre paréntesis el número de autores que tiene ese título.

```
for $libro in doc("libros.xml")/bookstore/book
return
	<libro>
		{ $libro/title/text() } ( { count($libro/author) } )
	</libro>
```

A partir de un for recuperamos todas las tuplas de los libros.

En la cláusula return, creamos la etiqueta “libro” y dentro enmarcamos el título del libro con la función “text” y entre paréntesis contamos los autores de cada libro con la función “count”.

- Mostrar los libros escritos en años que terminen en "3".

```
for $libro in doc("libros.xml")/bookstore/book
where ends-with($libro/year, "3")
return $libro
```

A partir de un for recuperamos todas las tuplas de los libros.
Filtramos las tuplas que no cumplan que la etiqueta “year” terminen en 3 con la función “ends-with”.
En la cláusula return recuperamos los libros filtrados.

- Mostrar los libros cuya categoría empiece por "C".

```
for $libro in doc("libros.xml")/bookstore/book
where starts-with($libro/@category, "C")
return $libro
```

A partir de un for recuperamos todas las tuplas de los libros.
Filtramos las tuplas que no cumplan que el atributo “category” empiece con “C” con la función “starts-with”.
En la cláusula return recuperamos los libros filtrados.

- Mostrar los libros que tengan una "X" mayúscula o minúscula en el título ordenados de manera descendente.

```
for $libro in doc("libros.xml")/bookstore/book
where contains($libro/title, "x") or contains($libro/title, "X")
order by $libro/title descending
return $libro
```

A partir de un for recuperamos todas las tuplas de los libros.
Filtramos las tuplas que contengan en el título una “c” minúscula o una “C” mayúscula.
Ordenamos la salida por el título en modo descendente.
En la cláusula return recuperamos los libros filtrados.

- Mostrar el título y el número de caracteres que tiene cada título, cada uno con su propia etiqueta.

```
for $libro in doc("libros.xml")/bookstore/book
return
      <libro>
```

```
            { $libro/title }
            <length> { string-length($libro/title) } </length>
      </libro>
```

A partir de un for recuperamos todas las tuplas de los libros.
En la cláusula return creamos la etiqueta “libro” y dentro de ella enmarcamos cada uno de los títulos y entre etiquetas “length” la cantidad de caracteres que tiene cada título utilizando la función “string-length”.

- Mostrar todos los años en los que se ha publicado un libro eliminando los repetidos. Etiquetarlos con "año".

```
for $año in distinct-values(doc("libros.xml")/bookstore/book/year)
return <año> { $año } </año>
```

A partir de un for recuperamos todas las tuplas de los años de los libros, con la excepción que en el momento indicar el nodo, utilizamos la función “distinct-values” que nos devuelve únicamente las tuplas sin repetir.
En la cláusula return creamos la etiqueta “año” y enmarcamos en ella cada tupla resultante.

- Mostrar todos los autores eliminando los que se repiten y ordenados por el número de caracteres que tiene cada autor.

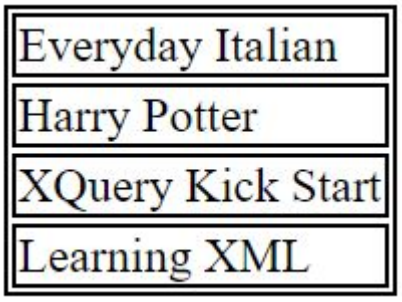
```
for $autor in distinct-values(doc("libros.xml")/bookstore/book/author)
order by string-length($autor)
return <autor> { $autor } </autor>
```

A partir de un for recuperamos todas las tuplas de los autores de cada libro, con la excepción que en el momento indicar el nodo, utilizamos la función “distinct-values” que nos devuelve únicamente las tuplas sin repetir.
Ordenamos por la extensión de la palabra del autor
En la cláusula return creamos la etiqueta “autor” y enmarcamos en ella cada tupla resultante, es decir, los autores de los libros sin repetir y ordenamos por su tamaño.

- Mostrar los títulos en una tabla de HTML.

```
<table>
      {
            for $libro in doc("libros.xml")/bookstore/book
            return
                  <tr>
                  <td> { $libro/title/text() } </td>
                  </tr>
      }
</table>
```

Creamos unas etiquetas de “table” que es la etiqueta en html para indicar que vamos a desarrollar una tabla.
Desarrollamos la función FLWOR dentro de ella.
A partir de un for recuperamos todas las tuplas de los libros.
Por cada título devuelto, en la cláusula return indicamos las etiquetas de columna y fila en HTML “tr” para fila y “td” para columna. Por lo que nuestra tabla estará compuesta por una fila por cada título y éste siempre irá en la primera columna.



Podríamos hacerlo al revés, para que cada libro sea una columna y todos en una fila:

```
<table>
	<tr>
	{
		for $libro in doc("libros.xml")/bookstore/book
		return
		<td> { $libro/title/text() } </td>
	}
	</tr>
</table>
```