SECUENCIA DE EJERCICIOS CON EL PROBADOR DE PATRONES

Nos vendrá bien conocer algunos elementos sintácticos a la hora de generar el árbol de resultados una vez aplicado el patrón correspondiente. Así sería la sintaxis en modo abreviado

/	Se sitúa en elemento raíz y el árbol de salida que genera coincide con el del documento original
../predecesor/elemento/sucesor/..	Para establecer caminos, como en MS-DOS.
elemento[i]	Cuando se ha encontrado más de un nodo que encaja con el patrón, se pueden recorrer como una tabla.
elemento[position() *condición***]**	Busca seleccionar nodos valiéndose del índice como si fuese un tabla.
elemento[end()]	Si hay más de nodo, coge el último.
*	Función parecida al del MS-DOS.
elemento[hijo]	Operador filtro.
@atributo	Operador distinción de atributo.
@*	Todos los atributos.
./	El contexto para iniciar la búsqueda es el nodo actual.
//elemento	Operador descendente recursivo, desde el nodo raíz, con lo cual está buscando todas las apariciones de ese elemento.
.//elemento	Operador descendente recursivo, desde el nodo actual.
elemento|elemento	Operador unión, se utiliza para hacer búsquedas combinadas.
any	Operador de selección cualquiera.
all	Operador de selección todos.
()	Operador agrupamiento de expresiones.
not()	Operador negación.

A esto añadimos los operadores aritméticos y de comparación:

Operador	Descripción	Ejemplo	Valor de retorno
|	Computa dos conjuntos de nodos	//edicion | //isbn	Nodos con todos los elementos edicion e isbn
+	Adición o suma	8+4	12
-	Substracción o resta	8-4	4
*	Multiplicación	8*4	32
div	División	8 div 4	2
mod	Módulo (resto de la división)	5 mod 2	1
=	Igual a	altura = 1.80	True si es 1.80; False si es 1.90
!=	No igual a	altura != 1.80	True si es 1.90; False si es 1.80
<	Menor que	altura < 1.80	True si es 1.70; False si es 1.90
<=	Menor o igual que	altura <= 1.80	True si es 1.80; False si es 1.90
>	Mayor que	altura > 1.80	True si es 1.90; False si es 1.80
>=	Mayor o igual que	altura >= 1.80	True si es 1.90; False si es 1.70
or	O	altura = 1.80 or altura = 1.70	True si es 1.80; False si es 1.50
and	Y	altura > 1.80 and altura < 2.10	True si es 1.90; False si es 1.70

Conectaremos con el probador de patrones que se encuentra en el aula virtual, y que es accesible en la dirección web (https://www.freeformatter.com/xpath-tester.html#ad-output), para comenzar el trabajo.

Partiendo del documento XML que puedes encontrar aquí debajo o en el mismo fichero comprimido donde obtuviste éste con el nombre *XML1.xml*, copia-pégalo en la ventana **Option 1: Copy-paste your XML here**. A partir de ese momento ya podrás teclear en el cuadro **XPath expression** los correspondientes patrones (dándole al botón EVALUATE XPath) que produzcan las salidas requeridas de los ejemplos planteados:

```
<?xml version="1.0"?>
<listaCompras>
    <cliente>
        <nombre>Ricardo Tapia</nombre>
        <dni>50.988.654</dni>
    </cliente>
    <producto>
        <nombre>Ternera</nombre>
        <cantidad unidad="Kg.">3</cantidad>
        <distribuidor NIF="Q45872267I">
            <nombre>Districarne</nombre>
        </distribuidor>
    </producto> <producto>
        <nombre>Jamón Cocido</nombre>
        <cantidad unidad="gr.">500</cantidad>
    </producto>
    <producto CAD="01/06/18">
        <nombre>Judías Verdes</nombre>
        <cantidad unidad="Kg.">2</cantidad>
    </producto>
    <producto CAD="02/10/18" PROC="Noruega">
        <nombre>Salmón Ahumado</nombre>
        <cantidad unidad="gr.">250</cantidad>
    </producto>
    <producto PROC="Hungría">
        <nombre>Tokaji</nombre>
        <cantidad unidad="cl.">500</cantidad>
    </producto>
</listaCompras>
```

/listaCompras/cliente : devuelve los elementos incluidos en las etiquetas **cliente** que están dentro de **listaCompras**.

```
<cliente>
  <nombre>Ricardo Tapia</nombre>
  <dni>50.988.654</dni>
</cliente>
```

/listaCompras/producto/nombre : devuelve los elementos incluidos en las etiquetas **nombre** que están en **producto** que están dentro de **listaCompras**.

```
<nombre>Ternera</nombre>
<nombre>Jamón Cocido</nombre>
<nombre>Judías Verdes</nombre>
<nombre>Salmón Ahumado</nombre>
<nombre>Tokaji</nombre>
```

/listaCompras/producto[nombre="Judías Verdes"] : devuelve los elementos incluidos en las etiquetas **producto** cuyo **nombre** coincide con *Judias Verdes*, que están dentro de **listaCompras**.

```
<producto CAD="01/06/18">
  <nombre>Judías Verdes</nombre>
  <cantidad unidad="Kg.">2</cantidad>
</producto>
```

/listaCompras/producto : devuelve los elementos incluidos en las etiquetas **producto** que están dentro de **listaCompras**.

```
<producto>
  <nombre>Ternera</nombre>
  <cantidad unidad="Kg.">3</cantidad>
  <distribuidor NIF="Q45872267I">
      <nombre>Distribuciones Cárnicas S.L.</nombre>
  </distribuidor>
</producto>
<producto>
  <nombre>Jamón Cocido</nombre>
  <cantidad unidad="gr.">500</cantidad>
</producto>
<producto CAD="01/06/18">
 <nombre>Judias Verdes</nombre>
 <cantidad unidad="Kg.">2</cantidad>
</producto>
<producto CAD="02/10/18" PROC=”Noruega”>
 <nombre>Salmón Ahumado</nombre>
 <cantidad unidad="gr.">250</cantidad>
</producto>
<producto PROC=”Hungría”>
 <nombre>Tokaji</nombre>
 <cantidad unidad="cl.">500</cantidad>
</producto>
```

/listaCompras/producto[@CAD] : devuelve los elementos incluidos en las etiquetas **producto** que tengan el atributo **CAD**, que están dentro de **listaCompras**.

```
<producto CAD="01/06/18">
  <nombre>Judias Verdes</nombre>
  <cantidad unidad="Kg.">2</cantidad>
</producto>
<producto CAD="02/10/18" PROC=”Noruega”>
 <nombre>Salmón Ahumado</nombre>
 <cantidad unidad="gr.">250</cantidad>
</producto>
```

/listaCompras/producto[0] : devuelve el primer elemento **producto** dentro de **listaCompras**. La limitación de este probador radica en que considera que el primer nodo se numera con el 1, el segundo nodo con el 2 y así sucesivamente. Eso sí, cuando se trabaja con el XML Copy Editor u otro que no sea un simulador, el primer nodo será considerado como el 0.

```
<producto>
  <nombre>Ternera</nombre>
  <cantidad unidad="Kg.">3</cantidad>
  <distribuidor NIF="Q45872267I">
    <nombre>Districarne</nombre>
 </distribuidor>
</producto>
```

/listaCompras/producto[position() < 2] : devuelve los dos primeros elementos **producto** dentro de **listaCompras**. La limitación indicada en el punto anterior sigue vigente para el patronaje de nodos a partir de una condición.

```
<producto>
 <nombre>Ternera</nombre>
<cantidad unidad="Kg.">3</cantidad>
<distribuidor NIF="Q45872267I">
  <nombre>Districarne</nombre>
 </distribuidor>
</producto>
<producto>
 <nombre>Jamón Cocido</nombre>
<cantidad unidad="gr.">500</cantidad>
</producto>
```

/listaCompras/producto[@CAD and @PROC] : devuelve aquellos elementos **producto** que tengan ambos atributos dentro de **listaCompras**.

```
<producto CAD="02/10/18" PROC=”Noruega”>
   <nombre>Salmón Ahumado</nombre>
   <cantidad unidad="gr.">250</cantidad>
</producto>
```

A continuación, y a partir del documento anterior, toca construir el patrón XPath a partir de los enunciados de los siguientes ejercicios:

1) La salida deseada al aplicar el patrón debe ser todo el documento.

2) Ahora visualizar los elementos **producto** con su contenido que existan en el documento.

3) Ahora visualizar los elementos **nombre** de **producto** que existan en el documento.

4) Ahora visualizar los elementos **nombre** de **cliente** que existan en el documento.

5) Prueba el siguiente patrón, ¿qué ha ocurrido?

```
.//
```

6) Ahora visualizar los elementos **nombre** de **distribuidor** que existan en el documento.

7) Ahora visualizar el primero de los elementos **producto** con su contenido que existan en el documento.

8) Ahora visualizar todos elementos **producto** con su contenido que existan en el documento, pero a partir del primero.

9) Ahora visualizar todos elementos **producto** con su contenido, que existan en el documento, pero a partir del tercero.

10) Ahora visualizar todos elementos **producto** con su contenido, que existan en el documento, pero entre el primero y el tercero.

11) Prueba el siguiente patrón, ¿qué ha ocurrido?

```
//nombre
```

12) Ahora visualizar todos elementos **producto** con su contenido, que tengan como elemento descendiente **distribuidor**.

13) Ahora visualizar todos elementos **producto** con su contenido, que tengan como **distribuidor** a Districarne.

14) Ahora visualizar todos elementos **producto** con su contenido, que contengan el atributo CAD.

15) Ahora visualizar todos elementos **producto** con su contenido, cuyo atributo CAD tenga el valor 01/12/00.

16) Ahora visualizar los elementos **cliente** y **producto** que existan en el documento.

17) Ahora visualizar todos elementos **producto** con su contenido, cuyo **nombre** sean Judías Verdes.

18) Los que se te vayan ocurriendo...

Nota: También puedes probar los ejemplos que propone el apartado del simulador (XPath Examples) aprovechando el documento XML expuesto.

Partiendo del documento XML denominado *XML2.xml*, teclea los patrones que produzcan las salidas deseadas. (Lo encontrarás en el aula virtual, ¡mu largo pa ponehlo!)

1) Visualizar los elementos **alumno** (con todo lo que contienen) que aprobaron en la convocatoria de Junio.

2) Ahora visualizar los elementos **alumno** (con todo lo que contienen) cuya nota final esté por encima de 7.

3) Ahora visualizar los elementos **alumno** (con todo lo que contienen) cuya nota final esté por encima de 7 y por debajo de 8.

4) Ahora visualizar los elementos **alumno** (con todo lo que contienen) cuya nota de teoría esté sea 5 y su nota práctica por encima de 8.

5) Ahora visualizar sólo el **nombre** de todos los elementos **alumno**.

6) Ahora visualizar sólo el **nombre** y **apellidos** de todos los elementos **alumno**.

7) Ahora visualizar sólo el **nombre**, **apellidos** y **nota-final** de todos los elementos **alumno**.

8) Ahora visualizar los elementos **alumno** (con todo lo que contienen) cuyo apellido sea García.

9) Ahora visualizar los tres primeros elementos **alumno** (con todo lo que contienen).