Nombre:…… Legajo: ……

Primer Parcial de
Diseño y Procesamiento de Documentos XML

Ejercicio 1

Dado el siguiente XML Schema:

tweet.xsd

```
<?xml version="1.0"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:simpleType name="coordX">
        <xsd:restriction base="xsd:decimal">
            <xsd:minInclusive value ="-180"/>
            <xsd:maxInclusive value ="180"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="coordY">
        <xsd:restriction base="xsd:decimal">
            <xsd:minInclusive value ="-90"/>
            <xsd:maxInclusive value ="90"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="cientocuarenta">
        <xsd:restriction base="xsd:string">
            <xsd:maxLength value ="140"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:complexType name="TwType">
        <xsd:sequence>
            <xsd:element name="usuario" type="xsd:string"/>
            <xsd:element name="fecha" type="xsd:dateTime"/>
            <xsd:element name="coordenadas" minOccurs="0">
                <xsd:complexType>
                    <xsd:attribute name="x" type="coordX" use="required"/>
                    <xsd:attribute name="y" type="coordY" use="required"/>
                  </xsd:complexType>
            </xsd:element>
            <xsd:element name="pais" type="xsd:string"/>
            <xsd:element name="texto" type="cientocuarenta"/>
        </xsd:sequence>
        <xsd:attribute name="identif" type="xsd:ID"/>
    </xsd:complexType>
    <xsd:element name="tweets">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="tweet" type="TwType" minOccurs="0"
maxOccurs="unbounded"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
</xsd:schema>
```

Nota: en el esquema anterior se utiliza el tipo **dateTime**. Este tipo de datos predefinido sirve para indicar día y hora (en el mismo dato), con el siguiente formato: AAAA-MM-DDThh:mm:ss. La letra **T** separa la fecha de la hora.

Nombre:…… Legajo: ……

Indicar cuáles de los siguientes documentos XML validan con este XSD. En el caso de que no validen, indicar **TODOS** los problemas encontrados. Justificar.

1.1)

tweet01.xml

```
<?xml version="1.0"?>
 <tweets xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation= "tweet.xsd">
        <tweet>
                <usuario>@DOLLIIRIGOYEN</usuario>
                 <fecha>2014-12-30T01:00:00</fecha>
                 <pais>Argentina</pais>
                 <texto>Felices Fiestas!!!! Por un 2015 con mucha Paz y Amor. http://fb.me/2GPHXQIuz</texto>
        </tweet>
        <tweet>
                <usuario>@CalaCocinero</usuario>
                <fecha>2015-01-14T04:37:03</fecha>
                 <coordenadas x="-58.3971" y="-34.5806"/>
                 <texto>Hoy tocó matambre a la pizza y mollejas... Y bué... es lo que hay...</texto>
        </tweet>
</tweets>
```

Rta:

1.2)

tweet02.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
 <tweets xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation= "tweet.xsd"/>
```

Rta:

Nombre:…… Legajo: ……

1.3)
tweet03.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<tweets xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation= "tweet.xsd">
    <tweet/>
</tweets>
```

Rta:

1.4)
tweet04.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
 <tweets xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
                xsi:noNamespaceSchemaLocation= "tweet.xsd">
        <tweet identif="123">
                <usuario>@DOLLIIRIGOYEN</usuario>
                <fecha>2014-12-30T01:00:00</fecha>
                <pais>Argentina</pais>
                <texto>Felices Fiestas!!!! Por un 2015 con mucha Paz y Amor.
http://fb.me/2GPHXQIuz</texto>
        </tweet>
        <tweet identif="A123">
                <usuario>@CalaCocinero</usuario>
                <fecha>2015-01-16T03:12:20</fecha>
                <coordenadas x="-58.3949" y="-134.5925"/>
                <pais>Argentina</pais>
                <texto>Ni la lluvia nos para ... </texto>
        </tweet>
</tweets>
```

Rta:

Nombre:…… Legajo: ……

1.5)
tweet05.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
 <tweets xmlns:xsi = http://www.w3.org/2001/XMLSchema-instance
      xsi:noNamespaceSchemaLocation= "tweet.xsd">
      <tweet>
           <usuario>CalaCocinero</usuario>
           <fecha>2015-01-16T03:12:20</fecha>
           <coordenadas/>
           <pais>Argentina</pais>
           <texto>Ni la lluvia nos para ...</texto>
      </tweet>
</tweets>
```

Rta:

Ejercicio 2

2.1) Sabiendo que el nombre de usuario en twitter debe comenzar con @ y luego contener hasta 15 caracteres que pueden ser:

- Letra minúscula
- Letra mayúscula
- Número
- Guión bajo

Se pide escribir el tipo **userT,** para tweet.xsd del **ejercicio 1)**, que responda a las reglas antes mencionadas.

Es decir, si en tweet.xsd del ejercicio 1) tuviéramos ahora

```
...
 <xsd:complexType name="TwType">
      <xsd:sequence>
           <xsd:element name="usuario" type="userT" />
           <xsd:element name="fecha" type="xsd:dateTime"/>
           <xsd:element name="coordenadas" minOccurs="0">
                <xsd:complexType>
                     <xsd:attribute name="x" type="coordX" use="required"/>
                     <xsd:attribute name="y" type="coordY" use="required"/>
                </xsd:complexType>
           </xsd:element>
           <xsd:element name="pais" type="xsd:string"/>
           <xsd:element name="texto" type="cientocuarenta"/>
      </xsd:sequence>
      <xsd:attribute name="identif" type="xsd:ID"/>
</xsd:complexType>
...
```

Nombre:…… Legajo: ……

podríamos validar, por ejemplo, los siguientes nombres de usuario

@DOLLIIRIGOYEN
@CalaCocinero
@
@123hoLLLA
@_

pero no validaríamos, por ejemplo:

Cal@aCocinero
@buenosdiasmisamigos
@est0!#

Rta:

Nombre:…… Legajo: ……

2.2) Usando los datos del **ejercicio 1)** (tweet.xsd), crear el tipo **twC,** como extensión del tipo TwType, que agregue a este último, dos elementos:

- **Fav**
- **Rt**

Ambos elementos son de tipo **entero** e indican respectivamente, la cantidad de favoritos y retweets que recibió un mensaje. Dichos elementos son optativos y en caso de estar presentes en el .xml, deberán aparecer exactamente una vez.

Es decir, si en **tweet.xsd** cambiamos la definición del elemento **tweet** para que su tipo sea **twC**:

```
.
.
 <xsd:element name="tweets">
      <xsd:complexType>
           <xsd:sequence>
                <xsd:element name="tweet" type="twC" minOccurs="0"  maxOccurs="unbounded"/>
           </xsd:sequence>
      </xsd:complexType>
</xsd:element>
.
.
```

se debe validar, por ejemplo, el siguiente xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<tweets xmlns:xsi = http://www.w3.org/2001/XMLSchema-instance
           xsi:noNamespaceSchemaLocation= "tweet2.2.xsd">
      <tweet>
           <usuario>@DOLLIIRIGOYEN</usuario>
           <fecha>2014-12-30T01:00:00</fecha>
           <pais>Argentina</pais>
           <texto>Felices Fiestas!!!! Por un 2015 con mucha Paz y Amor. http://fb.me/2GPHXQIuz</texto>
           <Fav>3</Fav>
           <Rt>12</Rt>
      </tweet>
</tweets>
```

Rta:

Nombre:…… Legajo: ……

Ejercicio 3

Se tiene un XML que representa la lista de pasajeros de un vuelo de avión:

flight.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<flight code="LA4554">
    <departure>2010-06-01</departure>
    <passengers>
        <person id="ARG1">
            <name>Gustavo Yoshizaki</name>
            <seat>
                <number>6</number>
                <position>F</ position >
            </seat>
            <telephone>15-5555-5555</telephone>
        </person>
        <person id="ARG2">
            <name>Silvia Gomez</name>
            <seat>
                <number>30</number>
                <position>A</position>
            </seat>
            <telephone>15-5555-4444</telephone>
        </person>
            .
            .
            .
        <person id="ARG3">
            <name>Baños Gabriel</name>
            <seat>
                <number>17</number>
                <position>B</position>
            </seat>
            <telephone>5555-3333</telephone>
            <specialFood>Only vegetables</specialFood>
        </person>
    </passengers>
</flight>
```

Cada vuelo es identificado por un código, y posee la fecha de salida y el listado de pasajeros. La fecha tiene el formato YYYY-MM-DD y debe ser una fecha válida.

En el caso de que un vuelo no complete su cupo mínimo de 15 pasajeros, el mismo se cancela y no se guarda registro. Es decir, siempre un vuelo debe tener al menos 15 pasajeros.

Los pasajeros se identifican por el atributo id.

El teléfono permite ingresar tanto número de teléfonos fijos como números de teléfonos celulares. Los teléfonos fijos están formados por 8 dígitos, separados en dos grupos de 4 por un guión. Los teléfonos celulares están formados por 10 dígitos, separados en 3 grupos por guiones. El primer grupo esta formado por 2 números y los restantes grupos por 4 números.

Cada pasajero tiene un solo número de teléfono asociado. Es posible que los valores se repitan (por ejemplo para las familias que viajan juntas y solo indican el número de la residencia)

Los asientos (seat) válidos están formados por un número y una letra, donde los números deben estar en el intervalo [1-45] y las letras en el intervalo [A-F]. Los asientos no se pueden repetir para un mismo vuelo.

Solamente en los casos donde se debe indicar una comida especial para el pasajero, aparece dicha información con la descripción exacta.

Nombre:…… Legajo: ……

3.1) Indicar todas las validaciones que NO se pueden realizar con un DTD (no hay que reescribirlo).

Rta:

3.2) Escribir un xsd que valide **flight.xml**. Si existen validaciones que no son posibles de efectuar indicar cuáles y por qué.

```
<?xml version="1.0"?>
 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">




</xsd:schema>
```

Nombre:…… Legajo: ……

Ejercicio 4

Utilizando el documento **flight.xml**, los aeropuertos deciden unificar en un solo archivo los datos de todos los vuelos de salida. Para esto, crean un nuevo archivo **flights.xml**, de la siguiente forma:

flights.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<flights>
    <flight code="LA4554">
        <departure>2010-06-01</departure>
        <passengers>
            <person id="ARG1">
                <name>Gustavo Yoshizaki</name>
                <seat>
                    <number>6</number>
                    <position>F</position>
                </seat>
                <telephone>15-5555-5555</telephone>
            </person>
            ...
        </passengers>
    </flight>
    ...
    <flight code="LA4554">
    ...
    </flight>
    ...
    <flight code="LA4554">
    ...
    </flight>
    ...
    <flight code="LA4554">
    ...
    </flight>
    ...
</flights>
```

Obviamente, debido a que el atributo **id** de una persona es de tipo ID, no se podrían ingresar la misma persona para dos vuelos diferentes.

4.1) Suponiendo que se elimina la restricción de tipo ID sobre el campo **id**, permitiendo ingresar a la misma persona en dos vuelos distintos, repitiendo todos sus datos personales. ¿A qué tipo de anomalía está expuesto el XML? Ejemplificar.

Rta:

4.2) ¿Se puede evitar este tipo de anomalías mediante validaciones? Si la respuesta es afirmativa indicar cuál sería la solución.

Rta:

Nombre:…… Legajo: ……

4.3) Proponer un diseño de XML, que no presente anomalías de redundancia y actualización. Realizarlo utilizando la representación gráfica de documentos XML vista en clase.

Rta:

Nombre:…… Legajo: ……

Ejercicio 5

Dado el siguiente archivo **cuentas.xml**

```
<?xml version="1.0"?>
<info>
    <clientes>
        <cliente id="CL0001">
            <nombre>Marcelo</nombre>
            <apellido>Gonzalez</apellido>
            <ingresos_anuales>200000</ingresos_anuales>
        </cliente>
        <cliente id="CL0002">
            <nombre>Alicia</nombre>
            <apellido>Hertz</apellido>
            <ingresos_anuales>250000</ingresos_anuales>
        </cliente>
        <cliente id="CL0003">
            <nombre>Marina</nombre>
            <apellido>Pierini</apellido>
            <ingresos_anuales>130000</ingresos_anuales>
        </cliente>
    </clientes>
    <cuentas>
        <cuenta id="CTA001" cliente="CL0003">
            <tipo>caja de ahorro</tipo>
            <saldo_inicial>2000</saldo_inicial>
        </cuenta>
        <cuenta id="CTA002" cliente="CL0002">
            <tipo>cuenta corriente</tipo>
            <saldo_inicial>40000</saldo_inicial>
        </cuenta>
        <cuenta id="CTA003" cliente="CL0003">
            <tipo>cuenta corriente</tipo>
            <saldo_inicial>3000</saldo_inicial>
        </cuenta>
    </cuentas>
    <transacciones>
        <transaccion id="T0034" cuenta="CTA002">
            <tipo>extraccion</tipo>
            <fecha>2014/08/01</fecha>
            <monto>2000</monto>
        </transaccion>
        <transaccion id="T0045" cuenta="CTA001">
            <tipo>deposito</tipo>
            <fecha>2014/08/01</fecha>
            <monto>5000</monto>
        </transaccion>
        <transaccion id="T0061" cuenta="CTA003">
            <tipo>deposito</tipo>
            <fecha>2014/08/03</fecha>
            <monto>2500</monto>
        </transaccion>
        <transaccion id="T0089" cuenta="CTA001">
            <tipo>extraccion</tipo>
            <fecha>2014/08/04</fecha>
            <monto>700</monto>
        </transaccion>
        <transaccion id="T0102" cuenta="CTA003">
            <tipo>extraccion</tipo>
            <fecha>2014/08/15</fecha>
            <monto>2000</monto>
        </transaccion>
        <transaccion id="T0133" cuenta="CTA001">
            <tipo>deposito</tipo>
            <fecha>2014/08/22</fecha>
            <monto>1400</monto>
        </transaccion>
    </transacciones>
</info>
```

Nombre:…… Legajo: ……

5.1)

Escribir la expresión Xpath 1.0 genérica que permita obtener el listado de los **nombres** de aquellos clientes que posean cuenta corriente. Usando **cuentas.xml** el resultado esperado es:

```
<nombre>Alicia</nombre>
<nombre>Marina</nombre>
```

Rta:

5.2)

Escribir la expresión XPath 1.0 genérica para obtener los **id**s de las cuentas que realizaron depósitos el 2014/08/01. Usando **cuentas.xml** obtendríamos:

```
CTA001
```

Rta:

5.3)

¿Cuál es el resultado obtenido al ejecutar la siguiente consulta XPath 1.0 sobre el documento **cuentas.xml?**

```
count(//tipo)
```

Rta:

5.4)

¿Cuál es el resultado obtenido al ejecutar la siguiente consulta XPath 1.0 sobre **cuentas.xml**?

```
/info/clientes/cliente[ingresos_anuales<sum(//ingresos_anuales) div
count(//cliente)]/nombre
```

Rta:

Nombre:…… Legajo: ……

Ejercicio 6

Se tiene el siguiente XML Schema:

paypal.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <xs:element name="reportingEngineRequest">
     <xs:complexType>
         <xs:sequence>
            <xs:element name="authRequest">
                <xs:complexType>
                 <xs:sequence>
                     <xs:element name="user" type="xs:string"/>
                     <xs:element name="vendor" type="xs:string"/>
                     <xs:element name="partner" type="xs:string"/>
                     <xs:element name="password" type="xs:string"/>
                     <xs:element name="executor" type="xs:string" minOccurs="0"/>
                 </xs:sequence>
               </xs:complexType>
           </xs:element>
            <xs:choice>
               <xs:element name="runReportRequest">
                  <xs:complexType>
                     <xs:sequence>
                        <xs:choice>
                           <xs:element name="reportName" type="xs:string"/>
                           <xs:element name="templateName" type="xs:string"/>
                        </xs:choice>
                         <xs:element name="reportParam" type= "reportParamType" minOccurs="0"
maxOccurs="unbounded"/>
                         <xs:element name="notificationRequired"  type="xs:boolean" minOccurs="0"/>
                         <xs:element name="pageSize" type="xs:int" minOccurs="0"/>
                     </xs:sequence>
                  </xs:complexType>
               </xs:element>
                <xs:element name="getResultsRequest">
                  <xs:complexType>
                     <xs:sequence>
                        <xs:choice minOccurs="0">
                           <xs:element name="scheduleName" type="xs:string"/>
                           <xs:element name="templateName" type="xs:string"/>
                           <xs:element name="reportName" type="xs:string"/>
                           <xs:element name="searchName" type="xs:string"/>
                           <xs:element name="reportId" type="xs:string"/>
                        </xs:choice>
                         <xs:choice minOccurs="0">
                               <xs:sequence>
                                  <xs:element name="startDate" type="xs:date"/>
                                  <xs:element name="endDate" type="xs:date"/>
                               </xs:sequence>
                         </xs:choice>
                     </xs:sequence>
                  </xs:complexType>
               </xs:element>
            </xs:choice>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   <xs:complexType name="reportParamType">
          <xs:sequence>
             <xs:element name="paramName" type="xs:string"/>
             <xs:element name="paramValue" type="xs:string"/>
          </xs:sequence>
   </xs:complexType>
</xs:schema>
```

Indicar cuál/es de los siguientes documentos XML no validan con *paypal.xsd*. En el caso de que no validen, justificar indicando TODOS los problemas encontrados.
En caso de no marcar TODOS los errores o de marcar como error algo que no lo es, el subítem será considerado sin puntaje alguno.

a) paypal1.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<reportingEngineRequest xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="paypal.xsd">
  <authRequest>
    <user>User One</user>
    <vendor>The Vendor</vendor>
    <partner>Some Partner</partner>
    <password>SecretPassword</password>
  </authRequest>
  <runReportRequest>
    <reportName>DailyActivityReport</reportName>
    <reportParam>
      <paramName>report_date</paramName>
       <paramValue>2018-09-07</paramValue>
    </reportParam>
    <pageSize>50</pageSize>
  </runReportRequest>
</reportingEngineRequest>
```

Nombre:…… Legajo: ……

Rta:

b) paypal2.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<reportingEngineRequest xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="paypal.xsd">
  <authRequest>
    <user>User One</user>
    <password>SecretPassword</password>
    <vendor>The Vendor</vendor>
  </authRequest>
  <runreportrequest>
    <reportName>DailyActivityReport</reportName>
    <reportParam>
      <paramName>report_date</paramName>
       <paramValue>2018/09/07</paramValue>
    </reportParam>
    <pageSize>50.0</pageSize>
  </runReportRequest>
</reportingEngineRequest>
```

Rta:

c) paypal3.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<reportingEngineRequest xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="paypal.xsd">
  <runReportRequest>
    <reportName>DailyActivityReport</reportName>
  </runReportRequest>
  <getResultsRequest>
    <reportId>131a</reportId>
  </getResultsRequest>
</reportingEngineRequest>
```

Rta:

d) paypal4.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<reportingEngineRequest xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="paypal.xsd">
  <authRequest>
    <user>12345</user>
    <vendor>Vendor</vendor>
    <partner>Partner</partner>
    <password/>
  </authRequest>
  <getResultsRequest>
    <reportId>131</reportId>
  </getResultsRequest>
      </reportingEngineRequest>
```

Nombre:…… Legajo: ……

Rta:

Ejercicio 7

El elemento *pageSize* de ***paypal.xsd*** es actualmente de tipo entero. Se quiere ahora restringir dicho tipo para que sea un número entero entre 0 y 1024 y además que tenga un atributo *unidad* (opcional) que indique si se mide en **KB** o **MB.** Se pide modificar lo necesario en ***paypal.xsd*** para que este cambio sea posible.
Luego de realizados los cambios el elemento pageSize validaría con

```
<pageSize unidad=”MB”>120</pageSize>
 <pageSize>0</pageSize>
```

Pero no debería validar con:

```
<pageSize>-1</pageSize>
 <pageSize unidad=”B”>0</pageSize>
```

Nombre:…… Legajo: ……

Ejercicio 8

Dados los documentos xml ***peli0.xml, peli1.xml***, escribir el documento DTD ***pelicula.dtd*** que valide a los 2 xml simultáneamente teniendo en cuenta las siguientes restricciones:

- Se debe tratar de que el DTD se ajuste lo máximo posible a los 2 xml, por lo que no se permite el uso del descriptor ANY.
- El elemento película está formado por los elementos *titulo*, *estreno*, *director* y *actores*. *titulo* y *estreno* deben aparecer obligatoriamente y en ese orden. Luego aparecen *director* y *actores* en cualquier orden, exactamente una vez cada uno.
- Los valores que puede tomar el atributo *moneda*, en cualquiera de sus apariciones, sólo serán: **Pesos** ó **USD**.

peli0.xml

```
<!DOCTYPE pelicula SYSTEM "pelicula.dtd">
<pelicula identif="peli001">
<titulo>Batman, La Pelicula</titulo>
<estreno>1966-07-30</estreno>
<director>
<nombre>Leslie H.Martinson</nombre>
<fechaNacimiento>1915-01-16</fechaNacimiento>
<salario moneda="USD">200000.2</salario>
</director>
<actores>
<actor>
<nombre>Adam West</nombre>
<fechaNacimiento>1928-09-19</fechaNacimiento>
<salario moneda="USD">100000</salario>                    <personaje>Batman</personaje>
</actor>
<actor>
<nombre>Burt Ward</nombre>
<fechaNacimiento>1945-07-06</fechaNacimiento>
<salario moneda="USD">50000.75</salario>
<personaje>Robin</personaje>
</actor>
</actores>
</pelicula>
```

peli1.xml

```
<!DOCTYPE pelicula SYSTEM "pelicula.dtd">
<pelicula identif="peli001">
<titulo>Iron Man</titulo>
<estreno>2008-04-30</estreno>
<actores>
<actor>
<nombre>Robert Downey Jr.</nombre>
<fechaNacimiento>1965-04-04</fechaNacimiento>
<salario moneda="USD">500000</salario>
<personaje>Ironman</personaje>
</actor>
</actores>
<director>
<nombre>Jon Favreau</nombre>
<fechaNacimiento>1966-10-19</fechaNacimiento>
<salario moneda="USD">115000000</salario>
</director>
</pelicula>
```

Nombre:…… Legajo: ……

Rta:

Ejercicio 9

Se tiene la siguiente colección formada por 3 documentos XML que no tiene todavía estructura DTD que los valide.

doc1.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<song>
    <band/>
    <album/>
    <year/>
    <track/>
    <author/>
    <author/>
</song>
```

doc2.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<song>
    <band/>
    <album/>
    <year/>
    <author/>
    <author/>
</song>
```

doc3.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<song>
    <band/>
    <album/>
    <year/>
    <author/>
    <author/>
    <author/>
</song>
```

Se pide escribir un DTD **docs.dtd** que valide los 3 documentos XML antes mencionados.

Rta:

Nombre:…………………………………………………………………………. Legajo: ………………………….