Exercice 1 :

Soient différentes définitions du même élément conteneur. On suppose que les éléments contenuA et contenuB sont définis comme des balises vides.

```
1. <!ELEMENT conteneur (contenuA | contenuB)*>
2. <!ELEMENT conteneur (contenuA* | contenuB*)>
3. <!ELEMENT conteneur (contenuA+, contenuB*)>
4. <!ELEMENT conteneur (contenuA, contenuB)*>
5. <!ELEMENT conteneur (contenuA, contenuB+)*>
6. <!ELEMENT conteneur (contenuA | contenuB+)*>
7. <!ELEMENT conteneur (contenuA | contenuB+)>
8. <!ELEMENT conteneur (contenuA? | contenuB+)>
```

Pour chaque élément conteneur donné ci-dessus, donner les numéros des définitions pour lesquelles il est valide.

<-- exemple A -->	<-- exemple B -->	<-- exemple C -->	<-- exemple D -->
< conteneur >	< conteneur >	< conteneur >	< conteneur >
<contenuA/>	<contenuA/>	<contenuA/>	<contenuB/>
<contenuA/>	<contenuA/>	<contenuB/>	<contenuB/>
<contenuB/>	<contenuA/>	<contenuA/>	<contenuA/>
<contenuA/>	<contenuB/>	<contenuB/>	<contenuA/>
<contenuB/>	<contenuB/>	<contenuA/>	< /conteneur >
< /conteneur >	< /conteneur >	<contenuB/>	
		< /conteneur >	

Exercice 2 :

Le document XML ci-dessous présente un annuaire des équipes de la ligue de football. Proposez une DTD afin que le document XML donné lui soit conforme.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE ligue SYSTEM "ligue.dtd">
<ligue>
    <equipe nom="PSG">
        <entraineur>Thomas Tuchel</entraineur>
        <joueur poste= "attaquant">Edinson Cavani</joueur >
        <joueur poste= "gardien">Salvatore Sirigu</joueur >
        <joueur poste= "defenseur">Thiago Silva</joueur >
        <ville>Paris</ville>
    </equipe>
   <equipe nom="OM">
        <entraineur>Rudi Garcia</entraineur>
        <joueur poste= "gardien"> Steve Mandanda</ joueur >
       <joueur poste= "attaquant">Florian Thauvin</ joueur >
        <ville>Marseille</ville>
    </equipe>
</ligue >
```

Utilisez un validateur en ligne pour vérifier votre proposition : http://www.xmlvalidation.com/
ou http://www.utilities-online.info/xsdvalidation

Exercice 3 :

Proposez un schéma XSD correspondant à la DTD présentée ci-dessous

```
<?xml version="1.0"?>
<!DOCTYPE bibliotheque
[
<!ELEMENT bibliotheque (livre+)>
<!ELEMENT livre (titre, auteur, ISBN)>
<!ELEMENT titre (#PCDATA)>
<!ELEMENT auteur (#PCDATA)>
<!ELEMENT ISBN (#PCDATA)>
<! ATTLIST livre catégorie(système|modelisation|formalisation)>
]>
<bibliotheque>
<livre catégorie= "système">
<titre> Real-time systems</titre>
<auteur>Giorgio Buttazzo</auteur>
<ISBN>72626252</ISBN >
</livre>
<livre catégorie= "modelisation">
<titre>DSL Engineering</titre>
<auteur>Markus Voelter</auteur>
<ISBN>89756767</ISBN >
</livre>
<livre catégorie= "formalisation">
<titre>Langages formels</titre>
<auteur>Olivier Carton</auteur>
<ISBN>5363652</ISBN>
</livre>
</bibliotheque>
```

Exercice 4 :

```
<?xml version="1.0" encoding="UTF-8"?>
<banque>
    <client>
        <!-- identité du client -->
        <identite>
          <nom> NomDeFamille</nom>
          <prenom> PrenomDeLaPersone</prenom>
        </identite>
        <comptes>
            <livretA>
                <montant>3500</montant>
            </livretA>
            <courant>
                <montant>6400</montant>
            </courant>
        </comptes>
    </client>
</banque>
```

Ce document XML représente une banque et ses clients. Pour chaque client, on connaît son identité, le montant de son livret A ainsi que le montant de son compte courant.

A- Vérifiez si le document est conforme au schéma suivant

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:element name="banque">
        <xsd:complexType >
            <xsd:sequence>
                <xsd:element name="client">
                    <xsd:complexType>
                        <xsd:sequence>
                            <xsd:element name="identite">
                                <xsd:complexType>
                                    <xsd:sequence>
                                        <xsd:element name="nom" type="xsd:string" />
                                        <xsd:element name="prenom" type="xsd:string"/>
                                    </xsd:sequence>
                                </xsd:complexType>
                            </xsd:element>
                            <xsd:element name="comptes">
                                <xsd:complexType>
                                    <xsd:sequence>
                                        <xsd:element name="livretA">
                                            <xsd:complexType>
                                                <xsd:sequence>
                                                    <xsd:element name="montant"
type="xsd:double" />
                                                </xsd:sequence>
                                            </xsd:complexType>
                                        </xsd:element>
                                        <xsd:element name="courant">
                                            <xsd:complexType>
                                                <xsd:sequence>
                                                    <xsd:element name="montant"
type="xsd:double" />
                                                </xsd:sequence>
                                            </xsd:complexType>
                                        </xsd:element>
                                    </xsd:sequence>
                                </xsd:complexType>
                            </xsd:element>
                        </xsd:sequence>
                    </xsd:complexType>
                </xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
</xsd:schema>
```

B- Proposez des améliorations pour que le schéma soit plus lisible (en séparant les types et en utilisant le référencement <xsd:element ref="mon_nom"/>)

Exercice 5:

Soit le schéma DTD suivant :

```
<!ELEMENT cv (nom, prenom, age?, rubrique+)>
<!ELEMENT nom (#PCDATA)>
<!ELEMENT prenom (#PCDATA)>
<!ELEMENT age (#PCDATA)>
<!ELEMENT rubrique (titre, contenu)>
<!ELEMENT contenu  (#PCDATA)>
<!ELEMENT titre (#PCDATA)>
```

A- Écrire un document XML valide par rapport à ce schéma avec au moins un élément age et deux rubriques.

Soit le document suivant le fichier HTML cible :

```
<html>
    <head>
        <title>CV de Foo BAR</title>
    </head>
    <body>
        <p><i>Foo</i></p>
        <p><i>BAR</i></p>
        <p><i>30 ans</i></p>
        <p><b>Compétences</b></p>
        <p>Athlète, Chanteur, guitariste</p>
        <p><b>Langues étrangères</b></p>
        <p>Espagnol, lu, parlé, écrit</p>
    </body>
</html>
```

B- Compléter le programme de transformation XSLT suivant pour obtenir le fichier HTML cible.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">
<xsl:template match="cv">
<html><head>
<title>CV de <xsl:value-of select="TODO1"/> <xsl:value-of select=" TODO2"/>
</title>
</head>
<body><xsl:apply-templates/></body>
</html>
</xsl:template>
<xsl:template match="nom">
<p><i> TODO3</i></p>
</xsl:template>
 TODO4
<xsl:template match="rubrique">TODO5</xsl:template>
<xsl:template match="titre">TODO6</xsl:template>
    TODO7
</xsl:stylesheet>
```

Exercice 6 :

Soit le schéma XML ci-dessous.

```
<papier type="scientifique">
    <titre>Les systèmes temps réel</titre>
    <sousTitre>De la modélisation à l'analyse</sousTitre>
    <auteur>E. Grolleau</auteur>
    <auteur>P. Richard</auteur>
    <resume>Nous proposons dans cet article d'aborder ...</resume>
    <abstract>In this paper we define...</abstract>
    <motsCles>
        <terme>Ingénierie des modèles</terme>
        <terme>XML</terme>
        <terme>Document</terme>
    </motsCles>
    <keywords>
        <word>model engineering</word>
        <word>XML</word>
        <word>Document</word>
    </keywords>
    <publication date="2019-07-05"/>
    <version maj="1" min="0"/>
    <ressource location="http://ensma.fr/lias.pdf"/>
</papier>
```

A- Sachant que le nœud courant (contexte) est un des éléments ***terme,*** donner quatre expressions XPath permettant de renvoyer l'élément **titre** du document XML.

Exercice 7 :

Écrire le programme XSLT permettant de transformer le fichier XML en résultat HTML.

```
<!--fichier.xml-->
<doc>
    <para>le premier paragraphe</para>
    <para>le deuxième paragraphe</para>
    <para>le dernier paragraphe</para>
</doc>
```

```
<html>
    <body>
        <p><i> le premier paragraphe </i></p>
        <p> le deuxième paragraphe </p>
        <p> le dernier paragraphe </p>
    </body>
</html>
```

Exercice 8 :

En se basant sur le fichier XML de l'exercice 6, compléter le fichier XSLT transf.xsl afin de générer, pour chaque élément terme, une instruction SQL d'insertion dans une table relationnelle de schéma : tMotsCles (terme, titre, url) (où terme est le terme sélectionné, titre est le titre du document et url est l'adresse de la ressource associée).

Pour rappel, la syntaxe d'insertion de données dans une table relationnelle en SQL :

INSERT INTO <Nom de la table> (<Liste ordonnée des colonnes>) VALUES (<Liste ordonnée des valeurs à affecter>).

```
<?xml version="1.0" encoding="UTF-8"?>
<!--transf.xsl-->
<xsl:stylesheet xmlns:xsl=http://www.w3.org/1999/XSL/Transform version="1.0">
<xsl:output method="text"/>
<xsl:template match="TODO1">
<xsl:apply-templates select="TODO2"/>
</xsl:template>
<xsl:template match=" TODO3">
TODO4 TODO5 tMotsCles (terme, titre, url) TODO6 (
'<xsl:value-of select=" TODO7"/>',
'<xsl:value-of select="// TODO8"/>',
'<xsl:value-of select="// TODO9"/>'
);
</xsl:template>
</xsl:stylesheet>
```