EISTI – 2008-2009 : Examen de Structuration de l'Information XML

Conditions d'examen

- Durée 1H30
- Tous documents électroniques déposés sur votre ordinateur.
- Accès interdit à Internet
- Rendu de l'examen sur clé USB (l'administration vous fournira la clé USB à la fin de l'examen)

Exercice 1 : Bases du langage XSL

On vous donne ci-dessous un fichier XML qui décrit de façon simplifiée le schéma d'une base de données. Ce fichier a été emprunté à Mr Daniel Lucazeau sur le site sam-mag.com.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<database name="infos">
   <tables>
      <table name="codeDepartement">
         <descRecord>
            <field name="IDDep" type="string"/>
            <field name="NOM" type="string"/>
         </descRecord>
         <records>
            <record>
               <field name="IDDep">01</field>
               <field name="NOM">Ain</field>
            </record>
         </records>
      </table>
      <table name="codeRegion">
         <descRecord>
            <field name="IDRegion" type="int"/>
            <field name="nom" type="string"/>
          </descRecord>
          <records>
            <record>
               <field name="IDRegion">1</field>
               <field name="nom">Alsace</field>
            </record>
         </records>
      </table>
   </tables>
</database>
```

Ecrire une feuille XSL qui permet d'afficher dans un fichier HTML, les noms des différentes tables ainsi que les noms des colonnes. L'affichage se fait dans un tableau comme suit :

Table	CodeDepartement
Colonne	IDDep
Colonne	NOM
Table	codeRegion
Colonne	IDRegion
Colonne	NOM

Le tableau ci-dessus est celui que vous devez obtenir ave l'exemple. Votre code XSL doit bien sûr s'appliquer à tout arbre XML ayant la même structure que l'exemple.

EISTI – 2008-2009 : Examen de Structuration de l'Information XML

Réponse : Barême 5 points

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!-- entête commune à tous les fichiers XMl -->

<!-- Début de la feuille XSL -->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="html" indent="yes" encoding="utf-8" />

<!-- le patron du noeud database -->
<xsl:template match="database">
<html>
	<head>
	</head>
	<body>
	<table border="1px">
		<xsl:for-each select="tables/table">
			<tr>
				<td>Table</td>
				<td><xsl:value-of select="@name"/></td>
			</tr>
			<xsl:for-each select="descRecord/field">
				<tr>
					<td>Colonne</td>
					<td><xsl:value-of select="@name"/></td>
				</tr>
			</xsl:for-each>
		</xsl:for-each>
	</table>
	</body>
</html>
</xsl:template>
<!-- fin de la feuille xsl-->
</xsl:stylesheet>
```

Exercice 2 : Navigation et filtre avec XSL

On reprend le fichier XML de l'exercice ci-dessus.

Ecrire la feuille XSL qui permet d'afficher dans un fichier texte le contenu des lignes (record) des tables non vides. Pour chaque table la présentation se fait comme suit :

Nom de la table
Nom du champs1 : valeur du champ1 ⋯ Nom du champsn : valeur du champsn
….

Réponse : Barême 5 points

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!-- entête commune à tous les fichiers XMl -->

<!-- Début de la feuille XSL -->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="text" indent="yes" encoding="utf-8" />

<!-- le patron du noeud database -->
<xsl:template match="database">
	<xsl:for-each select="tables/table[records]">
		<xsl:value-of select="@name"/>
<xsl:text>
</xsl:text>
		<xsl:for-each select="records/record">
			<xsl:for-each select="field">
				<xsl:value-of select="@name"/><xsl:text> : </xsl:text><xsl:value-of select="."/>
				<xsl:if test="not(position() = last())">
					<xsl:text> -- </xsl:text>
				</xsl:if>
			</xsl:for-each>
<xsl:text>
</xsl:text>

		</xsl:for-each>
	</xsl:for-each>
</xsl:template>


<!-- fin de la feuille xsl-->
</xsl:stylesheet>
```

Exercice 3 : XSD et DTD

1. XSD :
   Thème cinéma
   Donner l'extrait de schéma XSD qui permet de représenter un film qui est constitué :
   - d'un titre : attribut texte
   - d'une année : attribut entier entre 1895 et 2050

- d'au moins un réalisateur
- éventuellement d'acteurs

Les réalisateurs et les acteurs sont constitués :

- d'un nom : attribut texte
- d'un prénom : attribut texte

Réponse : Barême 3 points

```
<xsd:element name="film" type="typeFilm/>

   <xsd:simpleType name="typeAnnee">
      <xsd:restriction base="xsd:integer">
        <xsd:minInclusive value="1895"/>
        <xsd:maxInclusive value="2050"/>
      </xsd:restriction>
   </xsd:simpleType >

   <xsd:complexeType name="typeInd">
      <xsd:attribute name="nom" type="xsd:string"/>
      <xsd:attribute name="prenom" type="xsd:string"/>
   </xsd:simpleType>

   <xsd:complexeType name="typeFilm">

   <xsd:attribute name="titre" type="xsd:string"/>
   <xsd:attribute name="annee" type="typeAnnee"/>

   <xsd:sequence>
          <xsd:element name="realisateur" type="typeInd" minOccurs="1"
            maxOccurs="unbounded"/>
          <xsd:element name="acteur" type="typeInd" minOccurs="0"
            maxOccurs="unbounded"/>
   </xsd:sequence>
   </xsl:complexeType>
```

2. DTD

On considère 3 possibilités de définitions de l'élément e :

1. <!ELEMENT e(a*|b*)*>
2. <!ELEMENT e(a*|b*)>
3. <!ELEMENT e(a,b)*>

Les éléments a et b sont définis comme

<!ELEMENT a EMPTY>
<!ELEMENT b EMPTY>

-

Pour chaque élément, cocher la (ou les) DTD(s) qui le valide(nt) :

1. <e><a/><b/><a/><b/><a/><b/><a/><b/></e> o 1 o 2 o 3
2. <e><a/><b/><a/><a/><b/><a/><b/><b/><a/><b/></e> o 1 o 2 o 3
3. <e><a/><a/><a/></e> o 1 o 2 o 3
4. <e/> o 1 o 2 o 3

Réponse : Barême 2 points

1. <e><a/><b/><a/><b/><a/><b/><a/><b/></e> • 1 o 2 • 3
2. <e><a/><b/><a/><a/><b/><a/><b/><b/><a/><b/></e> • 1 o 2 o 3
3. <e><a/><a/><a/></e> • 1 • 2 o 3
4. <e/> • 1 • 2 • 3

Exercice 4 : Récursivité avec XSL

Le problème des **tours de Hanoï** est un jeu de réflexion imaginé par le mathématicien français Édouard Lucas, et consistant à déplacer des disques de diamètres différents d'une tour de « départ » à une tour d'« arrivée » en passant par une tour « intermédiaire » et ceci en un minimum de coups, tout en respectant les règles suivantes :

- on ne peut déplacer plus d'un disque à la fois,
- on ne peut placer un disque que sur un autre disque plus grand que lui ou sur un emplacement vide.

Au départ tous les disques se situent sur la tour de départ. Le but consiste à passer tous les disques sur la tour d'arrivée en se servant de la tour intermédiaire.

L'algorithme est le suivant :

```
procedure Hanoi(tdepart Entier, tintermediaire Entier, tarrivee Entier, nbDisques Entier)
    Si nbDisques = 1 Alors
        Déplacer un disque de la tour tdepart vers la tour tarrivee
    Sinon
        Hanoi(tdepart, tarrivee, tintermediaire, nbDisques - 1)
        Déplacer un disque de la tour tdepart vers la tour tarrivee
        Hanoi(tintermediaire, tdepart, tarrivee, nbDisques - 1)
    Fin Si
```

Traduire cet algorithme en XSL avec un template nommé.

Réponse : Barême : 5 points

```
<xsl:template name="hanoi">
    <xsl:param name="tour1"/>
    <xsl:param name="tour2"/>
    <xsl:param name="tour3"/>
    <xsl:param name="nbDisques"/>
<xsl:choose>
    <xsl:when test="nbDisques = 1">
```

```
		Déplacer un disque de <xsl:value-of select="$tour1"/> vers la tour <xsl:value-of select="$tour3"/>
	</xsl:when>
	<xsl:otherwise>
		<xsl:call-template name="hanoi">
			<xsl:with-param name="tour1" select="$tour1"/>
			<xsl:with-param name="tour2" select="$tour3"/>
			<xsl:with-param name="tour3" select="$tour2"/>
			<xsl:with-param name="nbDisques" select="$nbDisques - 1"/>
		</xsl:call-template>
		Déplacer un disque de <xsl:value-of select="$tour1"/> vers la tour <xsl:value-of select="$tour3"/>
		<xsl:call-template name="hanoi">
			<xsl:with-param name="tour1" select="$tour2"/>
			<xsl:with-param name="tour2" select="$tour1"/>
			<xsl:with-param name="tour3" select="$tour3"/>
			<xsl:with-param name="nbDisques" select="$nbDisques -1"/>
		</xsl:call-template>
	</xsl:otherwise>
</xsl:choose>
</xsl:template>
```