Examen XML (Sujet C)

— Master d'Ingénierie Informatique —

Mars 2011, durée 2h.

Les réponses apportées aux questions doivent être justifiées avec clarté et concision. Les documents sont interdits à l'exception d'une feuille de memento (4 pages A4). L'examen est composé de quatre exercices indépendants.

► **Exercice 1** Écrire un schéma qui valide les documents semblables au document ci-dessous. L'élément `list` contient des éléments `item`. Chacun de ces éléments `item` doit, soit contenir un entier, soit être vide.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<list>
  <item>123</item>
  <item/>
  <item>789</item>
</list>
```

► **Exercice 2** On considère des documents similaires au document suivant ou un élément `list` contient des éléments `item` avec un entier comme contenu.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<list>
  <item>2</item>
  <item>3</item>
  <item>5</item>
  <item>7</item>
</list>
```

- Donner un schématron qui vérifie que les valeurs des éléments `item` sont croissantes comme sur l'exemple ci-dessus.
- Modifier le schématron pour que, comme à l'exercice 1, certains éléments `item` puissent avoir un contenu vide. Le schématron doit alors ignorer ces éléments et vérifier que les valeurs des autres éléments `item` restants sont croissantes.

► **Exercice 3** On considère des documents XML très inspirés des documents Gedcom comme le document ci-dessous.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<gedcom>
<indi id="I01">
  <name>John /Doe/</name>
  <sex>M</sex>
  <fams idref="F1"/>
</indi>
<indi id="I02">
  <name>Rosanna /Housewife/</name>
  <sex>F</sex>
  <fams idref="F1"/>
</indi>
<indi id="I03">
  <name>Joe /Doe/</name>
  <indi id="I04">
    <name>Frank /Doe/</name>
    <sex>M</sex>
    <famc idref="F1"/>
  </indi>
  <fam id="F1">
    <husb idref="I01"/>
    <wife idref="I02"/>
    <chil idref="I03"/>
    <chil idref="I04"/>
  </fam>
</gedcom>
```

```

<sex>M</sex>
<famc idref="F1"/>
</indi>

```

Écrire une feuille de style XSLT qui prend en entrée un document de cette forme et qui le recopie en ajoutant les éléments suivants :

- Pour chaque élément `indi`, des enfants `fath` et `moth` contenant un attribut de nom `idref` dont la valeur est l'identifiant du père et de la mère ;
- Pour chaque élément `indi`, un enfant `chil` contenant un attribut de nom `idref` dont la valeur est la liste des identifiants des enfants (séparés par des espaces) ;
- Faire en sorte que ces éléments `fath`, `moth` et `chil` soient ajoutés uniquement lorsque les informations sont présentes dans le document.

► **Exercice 4** On considère le document suivant inspiré de DocBook.

```

<?xml version="1.0" encoding="iso-8859-1"?>
<book xml:id="book" xml:lang="fr">
  <title>Livre en français</title>
  <chapter xml:id="chap1">
    <title xml:id="d1e8">Chapitre en Français</title>
    <p xml:id="p1">Paragraphe en Français</p>
    <p xml:id="p2" xml:lang="en">Paragraph in English</p>
  </chapter>
  <chapter xml:id="chap2" xml:lang="en">
    <title>Chapter in English</title>
    <p xml:id="p3" xml:lang="fr">Paragraphe en Français</p>
    <p xml:id="p4">Paragraph in English</p>
  </chapter>
</book>

```

Donner le résultat de l'application de la feuille de style XSLT suivante sur le document précédent.

```

<?xml version="1.0" encoding="iso-8859-1"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="xml" encoding="iso-8859-1" indent="yes"/>
  <xsl:template match="/*">
    <xsl:copy>
      <xsl:apply-templates/>
    </xsl:copy>
  </xsl:template>
  <xsl:template match="p">
    <xsl:copy>
      <xsl:if test="ancestor-or-self::*[@xml:lang]">
        <xsl:attribute name="xml:lang">
          <xsl:value-of select="ancestor-or-self::*[@xml:lang][1]/@xml:lang"/>
        </xsl:attribute>
      </xsl:if>
      <xsl:apply-templates/>
    </xsl:copy>
  </xsl:template>
</xsl:stylesheet>

```