FH Aargau: Vorlesung XML Sommersemester 2004

Test XSL Tranformations
28.6.2004, 13^{15} - 14^{15}

Name: _____

Bitte schreiben Sie *sorgfältig* und *gut lesbar*! Danke!
Für diesen Test dürfen Sie Ihre *schriftlichen Unterlagen* sowie Ihren Laptop, aber nur mit ausgestecktem Netzwerkkabel!

Da XSLT Code durch die XML-Syntax schnell unübersichtlich wird, können Sie in diesem Test eine Pseudo-Notation verwenden, in der Sie sich sie End Tags sparen und Schachtelungen statt dessen durch Klammern markieren. Sie können auch Start Tags vereinfachen (kein Namespace Prefix, keine spitzen Klammern), aber die Syntax der Attribute sollte erhalten bleiben. Anweisungen werden durch Semikolon getrennt. Hier ein kleines Beispiel:

```
<xsl:for-each select="test">
 <xsl:value-of select="child"/>
 <xsl:text>  </xsl:text>
</xsl:for-each>
```

```
for-each select="test" {
 value-of select="child" ;
 text { "  " } ;
}
```

Sollte Ihnen der Pseudocode nicht gefallen, so können Sie natürlich auch kompletten XSLT Code schreiben, bei reinen Syntaxfehlern gibt es keine Abzüge in der Bewertung!

1) In einem XML-Dokument seien Personendaten gespeichert. Die Struktur ist sehr simpel, wobei darauf zu achten ist, dass jede Person beliebig viele Vornamen haben kann, jedoch nur einen Nachnamen. Es können beliebig viele Personen aufgelistet werden. Das Dokument hat den folgenden Aufbau:

```
<personen>
       <person>
              <vorname>Erik</vorname>
              <vorname>Thomas</vorname>
              <nachname>Wilde</nachname>
       </person>
       <person> .... </person>
</personen>
```

   In dieser Aufgabe geht es um die Verwendung von Keys in XSLT, und um ihre Anwendung durch die Kombination verschiedener Keys.
   a) Definieren sie Keys, mit denen Personen *nach ihren Vornamen* und *nach ihrem Nachnamen* gesucht werden können.
   b) Benutzen Sie die in a) definierten Keys, um alle Personen zu selektieren, die die Vornamen "Erik" *und* "Thomas" *und* den Nachnamen "Wilde" haben.
   c) Selektieren Sie ebenfalls mit den Keys aus Teil a) alle die Personen, die mit Nachnamen "Wilde", aber mit Vornamen *nicht* "Erik" heissen.
   d) Schreiben Sie XSLT Code (also keine XML Schema Identity Constraints), der überprüft, dass *niemand zwei gleiche Vornamen* trägt. Der Code muss nicht auf Effizienz optimiert sein. Der Code muss auch nicht unbedingt Keys benutzen.

2) Lösen Sie die Aufgaben 1b) und 1c) durch XSLT *ohne die Verwendung von Keys*.

3) Vereinfachen Sie den folgenden XSLT Code:

```
<xsl:for-each select="//table">
	<xsl:if test="thead">
		<xsl:for-each select="thead/tr">
			<xsl:for-each select="th | td">
				<xsl:if test="@class = 'centered'">
					<dosomething/>
				</xsl:if>
			</xsl:for-each>
		</xsl:for-each>
	</xsl:if>
</xsl:for-each>
```

Es soll nur noch eine `<xsl:for-each>` Schleife geben im vereinfachten Code:

```
<xsl:for-each select=" … ">
	<dosomething/>
</xsl:for-each>
```

Sie müssen also alle im Ausgangscode geschachtelten Bedingungen und Schleifen in *einem einzigen XPath* zusammenfassen, der funktional *identisch* sein muss zum Ausgangscode. Es gibt verschiedene Arten, diese Aufgabe zu lösen!

4) Schreiben Sie ein allgemeingültiges XSLT Programm, das ein beliebiges XML Dokument als Eingabe nimmt und nahezu unverändert wieder ausgibt, mit der Ausnahme, dass *Mixed Content* speziell behandelt wird. Mixed Content soll so verändert werden, dass jeder *Mixed Content Text Node* in ein neues Element verpackt wird, so dass das Dokument nach der Transformation frei ist von Mixed Content. Als Beispiel wird das Dokument

```
<document test="value">
	<content test="value">KEIN mixed content!</content>
	<content>mixed content! <content>KEIN mixed content!</content>
	</content>
</document>
```

durch die Transformation (die unterstrichenen Teile sind wichtig) zu

```
<document test="value">
	<content test="value">KEIN mixed content!</content>
	<content>
		<TEXTNODE>mixed content! </TEXTNODE>
		<content>KEIN mixed content!</content>
	</content>
</document>
```

Der wichtigste Aspekt bei dieser Aufgabe ist, Mixed Content Text Nodes durch einen XPath zu selektieren, andere Lösungen sind deutlich aufwendiger! Damit nicht alle Whitespace Text Nodes als Mixed Content behandelt werden, fügen Sie am besten `<xsl:strip-space elements="*"/>` in Ihr XSLT Programm ein.