

Gruppe A	PRÜFUNG AUS "SEMISTRUKTURIERTE DATEN" 181.135		23. 06. 2010
Kennnr.	Matrikelnr.	Familienname	Vorname

Arbeitszeit: 120 Minuten. Aufgaben sind auf den Angabeblättern zu lösen; Zusatzblätter werden nicht gewertet. Bitte tragen Sie Namen, Studienkennzahl und Matrikelnummer ein. Halten Sie Ihren Studentenausweis bereit.

### Aufgabe 1:

(9)

Betrachten Sie die folgende DTD **test.dtd**:

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT A (B*,C?,D?)>
<!ELEMENT B (#PCDATA|D|E)*>
<!ELEMENT C (E?,D,D?)>
<!ELEMENT D EMPTY>
<!ATTLIST D P ID #REQUIRED>
<!ELEMENT E (#PCDATA)>
<!ATTLIST E Q IDREF #IMPLIED>
```

Betrachten Sie weiters die sechs verschiedenen XML-Dateien, die unten angeführt sind. Hinweise:

- Gehen Sie davon aus, dass allen folgenden Dateien die Zeilen  

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE A SYSTEM "test.dtd">
```

vorangestellt sind.
- Sie können auch davon ausgehen, dass alle folgenden XML-Dateien wohlgeformt sind. Es geht also lediglich darum, ihre Gültigkeit bezüglich **test.dtd** zu entscheiden.

Kreuzen Sie an, welche der folgenden XML-Dateien gültig bezüglich **test.dtd** sind.

- |  |                              |                                |
|--|------------------------------|--------------------------------|
| 1. <A><B>xyz</B><C>xyz</C></A>             | gültig <input type="radio"/> | ungültig <input type="radio"/> |
| 2. <A><C><D P="s1"/></C></A>               | gültig <input type="radio"/> | ungültig <input type="radio"/> |
| 3. <A><B>xyz<D P="s1"/>xyz</B></A>         | gültig <input type="radio"/> | ungültig <input type="radio"/> |
| 4. <A><C><E Q="s1"></E><D P="s1"/></C></A> | gültig <input type="radio"/> | ungültig <input type="radio"/> |
| 5. <A><C>xyz<D P="s1"/></C></A>            | gültig <input type="radio"/> | ungültig <input type="radio"/> |
| 6. <A><B><E Q="s1"></E></B></A>            | gültig <input type="radio"/> | ungültig <input type="radio"/> |

(Pro korrekter Antwort 1.5 Punkte, **pro inkorrektter Antwort -1.5 Punkte**, pro nicht beantworteter Frage 0 Punkte, für die gesamte Aufgabe mindestens 0 Punkte)

## Aufgabe 2:

(6)

Betrachten Sie die folgende XML- Datei **ns.xml**:

```
<?xml version="1.0" encoding="UTF-8"?>
<A xmlns="uri1">
  <ns1:B xmlns="uri2" xmlns:ns1="uri1">
    <D attr1="X"/>
  </ns1:B>
  <C xmlns="uri2">
    <E attr2="Y"/>
  </C>
</A>
```

Kreuzen Sie an, ob die folgenden Aussagen für die Datei **ns.xml** wahr oder falsch sind.

1. Element E und Attribut **attr2** liegen im selben Namespace. wahr  falsch
2. Element A und Element B liegen im selben Namespace. wahr  falsch
3. Elemente A und E liegen im selben Namespace. wahr  falsch
4. Element D liegt im leeren Namespace. wahr  falsch

(Pro korrekter Antwort 1.5 Punkte, **pro inkorrektter Antwort -1.5 Punkte**, pro nicht beantworteter Frage 0 Punkte, für die gesamte Aufgabe mindestens 0 Punkte)

## Aufgabe 3:

(12)

Kreuzen Sie an, ob die folgenden Aussagen wahr oder falsch sind.

1. Um die Wohlgeformtheit eines XML-Dokuments zu überprüfen wird eine DTD oder ein XML-Schema benötigt. wahr  falsch
2. In einem wohlgeformten XML-Dokument dürfen zwei Attribute mit demselben qualifizierten Namen in einem Elementstarttag auftreten. wahr  falsch
3. Die DTD Elementdeklaration  $(A^*|B^*|C^*)$  ist gleichwertig mit  $(A|B|C)^*$ . wahr  falsch
4. Rekursive Definitionen wie z.B.  $\langle !ELEMENT B (B?,C) \rangle$  sind in DTDs verboten. wahr  falsch
5. Im DOM-Modell ist die Wurzel gleich dem Wuzelement des Dokuments. wahr  falsch
6. SAX: Textevents können auch direkt hintereinander auftreten. wahr  falsch
7. Die Reihenfolge der Templates in einem XSLT 1.0 Stylesheet hat standardmäßig keine Auswirkung auf den Ablauf des XSLT-Stylesheets. wahr  falsch
8. In XQuery sind selbst definierte rekursive Funktionen erlaubt. wahr  falsch

(Pro korrekter Antwort 1.5 Punkte, **pro inkorrektter Antwort -1.5 Punkte**, pro nicht beantworteter Frage 0 Punkte, für die gesamte Aufgabe mindestens 0 Punkte)

Die folgenden Aufgaben 4 – 8 beziehen sich auf das XML-Dokument `wm.xml`, das Sie auf der letzten Seite dieser Prüfungsangabe finden.

**Aufgabe 4:**

(9)

Nehmen Sie an, dass das Dokument `wm.xml` (siehe Anhang) als DOM-Baum im Speicher ist.

Kreuzen Sie an, ob folgende Codeblöcke den Knoten

```
<land kuerzel="FRA" name="Frankreich" weltmeister="ja"/>
```

zurückgeben (5. Zeile von `wm.xml`). Der Aufruf erfolgt jeweils mit der Document-Node von `wm.xml` als Parameter.

```
1. public Node variante1(Node node) {  
    NodeList nl = ((Document)node).getElementsByTagName("land");  
    return nl.item(2);  
}
```

wahr  falsch

```
2. public Node variante2(Node node) {  
    if (! node.hasChildNodes()) return null;  
    NodeList list = node.getChildNodes();  
    for (int i=0; i < list.getLength(); i++) {  
        Node subnode = list.item(i);  
        if (subnode.getNodeType() == Node.ELEMENT_NODE) {  
            if (subnode.getNodeName().equals("land")) {  
                return subnode.getNextSibling().getNextSibling();  
            } else {  
                Node tmp = variante2(subnode);  
                if(tmp!=null) return tmp;  
            }  
        }  
    }  
    return null;  
}
```

wahr  falsch

```
3. public Node variante3(Node node) {  
    NodeList nl = ((Document)node).getElementsByTagName("teilnehmer");  
    Node teilnehmer = nl.item(0);  
    return teilnehmer.getChildNodes().item(2);  
}
```

wahr  falsch

(Pro korrekter Antwort 3 Punkte, **pro inkorrektter Antwort -3 Punkte**, pro nicht beantworteter Frage 0 Punkte, für die gesamte Aufgabe mindestens 0 Punkte)

Vervollständigen Sie das XML Schema **wm.xsd**, sodass XML-Dokumente in der Gestalt von **wm.xml** (siehe Anhang) bezüglich dieses XML Schemas gültig sind. Berücksichtigen Sie beim Erstellen des XML Schemas folgende Punkte:

- Sie brauchen nur den Typ **einteilungType** zu vervollständigen, der Rest des XML Schemas ist bereits gegeben.
- Das Element **einteilung** kann ein oder mehrere Elemente **gruppe** enthalten.
- Jedes Element **gruppe** enthält genau vier Elemente **land**, gefolgt von einem optionalen Element **vorschau**.
- Der Inhalt des **vorschau**-Elementes soll gemischt sein; es dürfen als Subelemente **spieler**- und **land**-Elemente in beliebiger Anzahl (auch gar nicht) auftreten.
- Alle Attribute sind verpflichtend und vom Typ **xs:string**.
- Die Definition von Schlüsselbeziehungen ist **nicht** notwendig.

Datei **wm.xsd**:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="wm">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="teilnehmer" type="teilnehmerType"/>
        <xs:element name="spieler" maxOccurs="unbounded" type="spielerType"/>
        <xs:element name="einteilung" type="einteilungType"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:complexType name="teilnehmerType">
    <!-- Nicht Teil der Prüfungsangabe -->
  </xs:complexType>

  <xs:complexType name="spielerType">
    <!-- Nicht Teil der Prüfungsangabe -->
  </xs:complexType>

  <xs:complexType name="einteilungType">
```

Datei **wm.xsd** (Fortsetzung):

```
</xs:complexType>  
</xs:schema>
```

**Aufgabe 6:**

Betrachten Sie die folgende XQuery-Anfrage **kaderliste.xqy**.

Datei **kaderliste.xqy**:

```
let $wm := doc('wm.xml')/wm return
<kaderliste>
{for $land in $wm//teilnehmer/land order by $land/@name return
  <nation name="{ $land/@name }">
    {for $spieler in $wm/spieler where $spieler/@land = $land/@kuerzel return
      <spieler trikotnr="{ $spieler/@trikotnr }">
        { $spieler/@name }
      </spieler> }
    </nation> }
</kaderliste>
```

Vervollständigen Sie hier das XSLT-Stylesheet **kaderliste.xsl**. Es soll für alle XML-Dokumente der Gestalt **wm.xml** (siehe Anhang) die gleichen Ergebnisse liefern wie **kaderliste.xqy**. Verwenden Sie dabei keine Kontrollstrukturen wie z.B. `xsl:for-each`. Sie brauchen sich nicht um Whitespaces etc. zu kümmern.

Datei **kaderliste.xsl**:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

```
</xsl:stylesheet>
```

## Aufgabe 7:

(6)

Schreiben Sie XPath-Anfragen für folgende Aufgabenstellungen. Die Abfragen sollen für alle XML-Dokumente der Gestalt **wm.xml** (siehe Anhang) funktionieren. Zu jeder Abfrage ist ein Beispiel mit der erwarteten Ausgabe (bezgl. **wm.xml**) angegeben.

1. Ausgabe aller Spieler mit der Trikotnummer 10.

```
<spieler id="0210" name="Lionel Messi" land="ARG" trikotnr="10"/>
<spieler id="0160" name="Diego Forlan" land="URU" trikotnr="10"/>
```

2. Ausgabe aller Länder, die noch nicht Weltmeister waren.

```
<land kuerzel="GRE" name="Griechenland"/>
<land kuerzel="MEX" name="Mexiko"/>
<land kuerzel="NGA" name="Nigeria"/>
<land kuerzel="RSA" name="Suedafrika"/>
<land kuerzel="KOR" name="Suedkorea"/>
```

3. Ausgabe aller Gruppen, in denen mehr als eine Weltmeisternation spielt.

```
<gruppe name="A">
  <!-- ... -->
</gruppe>
```

**Aufgabe 8:**

(9)

Nehmen Sie an, dass folgender XMLFilter auf das XML-Dokument **wm.xml** (siehe Anhang) ausgeführt wird. Schreiben Sie auf, welche SAX-Events die Applikation weitergereicht bekommt.

Eventnamen dürfen Sie abkürzen.

```
public class SSDFilter extends XMLFilterImpl {
    public void startElement(String uri, String localName, String qName,
        Attributes atts) throws SAXException {
        if("10".equals(atts.getValue("trikotnr"))) {
            super.startElement(uri,localName,qName,atts);
            super.endElement(uri,localName,qName);
        }
    }
    public void endElement(String uri, String localName, String qName)
        throws SAXException {
    }
    public void characters(char []ch, int start, int length) throws SAXException {
        //
    }
}
```

Eventname	Elementname, Attribute, Textinhalt etc...

Sie können diese Seite abtrennen!

Datei wm.xml:

```
<wm>
  <teilnehmer>
    <land kuerzel="ARG" name="Argentinien" weltmeister="ja"/>
    <land kuerzel="GRE" name="Griechenland"/>
    <land kuerzel="FRA" name="Frankreich" weltmeister="ja"/>
    <land kuerzel="MEX" name="Mexiko"/>
    <land kuerzel="NGA" name="Nigeria"/>
    <land kuerzel="RSA" name="Suedafrika"/>
    <land kuerzel="KOR" name="Suedkorea"/>
    <land kuerzel="URU" name="Uruguay" weltmeister="ja"/>
    <!-- .... -->
  </teilnehmer>
  <spieler id="0210" name="Lionel Messi" land="ARG" trikotnr="10"/>
  <spieler id="0160" name="Diego Forlan" land="URU" trikotnr="10"/>
  <spieler id="0301" name="Hugo Lloris" trikotnr="1" land="FRA"/>
  <spieler id="0312" name="Thierry Henry" trikotnr="12" land="FRA"/>
  <!-- .... -->
  <einteilung>
    <gruppe name="A">
      <land>RSA</land><land>MEX</land><land>FRA</land><land>URU</land>
      <vorschau>
        Nur durch das umstrittene Handstor von <spieler id="0312"/> konnte
        sich <land kuerzel="FRA"/> qualifizieren. Toptorjäger in dieser
        Gruppe ist <spieler id="0160"/>.
      </vorschau>
    </gruppe>
    <gruppe name="B">
      <land>KOR</land><land>NGA</land><land>ARG</land><land>GRE</land>
    </gruppe>
    <!-- .... -->
  </einteilung>
</wm>
```

Gesamtpunkte: 75