Gruppe A	PRÜFUNG AUS "SEMISTRUKTURIERTE DATEN" **184.705**		27. 06. 2012
Kennnr.	Matrikelnr.	Familienname	Vorname

Arbeitszeit: 100 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 XML-Schema Datei **test.xsd**:

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  <xsd:element name="X">
    <xsd:complexType mixed="true">
      <xsd:choice minOccurs="1" maxOccurs="2">
        <xsd:sequence>
          <xsd:element name="Y" minOccurs="0" type="xsd:int"/>
        </xsd:sequence>
        <xsd:choice>
          <xsd:element name="Z" maxOccurs="3" type="xsd:int"/>
          <xsd:element name="X" minOccurs="0" type="xsd:int"/>
        </xsd:choice>
      </xsd:choice>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

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

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

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

1. `<X>abc<Z>6</Z><Z>2</Z></X>` gültig ⊗ ungültig ◯
2. `<X><Y>123</Y>456<Y>789</Y></X>` gültig ⊗ ungültig ◯
3. `<X><X><Y>123</Y></X></X>` gültig ◯ ungültig ⊗
4. `<X><Z>1</Z><X>1</X></X>` gültig ⊗ ungültig ◯
5. `<X><Y><Y>123</Y></Y></X>` gültig ◯ ungültig ⊗
6. `<X><Y>1</Y><Y>2</Y><Y>3</Y></X>` gültig ◯ ungültig ⊗

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

Aufgabe 2: (12)

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

1. Wohlgeformte XML-Dokumente sind immer auch gültig. wahr ◯ falsch ⊗
2. DOM sieht Attribute als Kinder von Elementen an. wahr ◯ falsch ⊗
3. In XPath beginnt ein absoluter Pfad immer vom aktuellen Context Node aus wahr ◯ falsch ⊗
4. SAX erlaubt es, ein XML Dokument in beliebiger Reihenfolge zu durchlaufen. wahr ◯ falsch ⊗
5. Die DTD Elementdeklaration `(A*|B*|C*)` ist gleichwertig mit `(A|B|C)*`. wahr ◯ falsch ⊗
6. Bei XSLT wird pro Knoten maximal ein Template angewendet. wahr ⊗ falsch ◯
7. Attribute werden bei einem SAX Parser als eigene Events getriggert. wahr ◯ falsch ⊗
8. Der "parent" eines SAX-Filter kann selbst auch ein SAX-Filter sein. wahr ⊗ falsch ◯

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

Die folgenden Aufgaben 3 – 7 beziehen sich auf das XML-Dokument em.xml, das Sie auf der letzten Seite dieser Prüfungsangabe finden.

Aufgabe 3: (12)

Vervollständigen Sie das DTD Dokument **em.dtd**, sodass XML-Dokumente in der Gestalt von **em.xml** (siehe Anhang) bezüglich dieser DTD gültig sind. Berücksichtigen Sie beim Erstellen der DTD folgende Punkte:

- Das Element `gruppenphase` ist das Wurzelelement und besteht aus mindestens einem `gruppe`-Element, gefolgt von beliebig vielen `spiel`-Elementen, gefolgt von wiederum beliebig vielen `bericht`-Elementen.
- Das Element `gruppe` besteht aus exakt 4 `team`-Elementen; das Element `bericht` ist gemischt und darf `land`-Elemente enthalten.
- Das `land`-Attribut des `team`-Elements stellt einen Primärschlüssel dar. Versuchen Sie die entsprechenden Fremdschlüssel zu finden und in der DTD abzubilden.
- Kümmern Sie sich um eine entsprechende Behandlung des Namespaces und der Präfixes.
- Das Attribut des `gruppenphase` Elements muss, wenn im Instanzdokument angegeben, den Wert "em" haben; alle anderen Attribute sollen verpflichtend sein.
- Sollten bei bestimmten Elementen oder Attributen keine näheren Angaben bezüglich des genauen Typs vorgegeben sein, wählen Sie selbst einen sinnvollen Typ aus.

Datei **em.dtd**:

```
<!ELEMENT em:gruppenphase (em:gruppe+, em:spiel*, em:bericht*)>
<!ATTLIST em:gruppenphase xmlns:em CDATA #FIXED "em">
<!ELEMENT em:gruppe (em:team,em:team,em:team,em:team)>
<!ATTLIST em:gruppe name CDATA #REQUIRED>
<!ELEMENT em:team (#PCDATA)>
<!ATTLIST em:team land ID #REQUIRED>
<!ELEMENT em:spiel EMPTY>
<!ATTLIST em:spiel team1 IDREF #REQUIRED team2 IDREF #REQUIRED
                   tore1 CDATA #REQUIRED tore2 CDATA #REQUIRED>
<!ELEMENT em:bericht (#PCDATA | em:land)*>
<!ELEMENT em:land EMPTY>
<!ATTLIST em:land ref IDREF #REQUIRED>
```

Aufgabe 4: (6)

Kreuzen Sie an, ob die folgenden Aussagen wahr oder falsch sind. Die Aussagen beziehen sich auf eine DTD vom vorigen Beispiel, bzw. ein entsprechendes XML Schema Dokument, die XML-Dokumente in der Gestalt von **em.xml** spezifizieren.

1. Falls sich im Instanzdokument, z.B. **em.xml**, der Präfix ändert (z.B. von "em:…" auf "wm:…") muss auch die DTD geändert werden. wahr ⊗ falsch ◯
2. Falls sich im Instanzdokument, z.B. **em.xml**, der Präfix ändert (z.B. von "em:…" auf "wm:…"), muss auch das XML Schema geändert werden. wahr ◯ falsch ⊗
3. In einem XML Schema für **em.xml** muss das Attribut `xmlns:em` explizit definiert werden. wahr ◯ falsch ⊗
4. In einem XML Schema für **em.xml** kann man auf einen Target-Namespace verzichten. wahr ◯ falsch ⊗

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

Aufgabe 5: (10)

Betrachten Sie die folgenden XPath-Abfragen angewandt auf das Dokument **em.xml** (siehe Anhang).

- Falls der angegebene XPath Ausdruck keine Knoten selektiert, notieren Sie im entsprechenden Feld "leere Ausgabe".
- Falls als Ergebnis `em:team` Elemente selektiert werden, geben Sie jeweils das Attribut `land` an.
- Falls als Ergebnis `em:spiel` Elemente selektiert werden, geben Sie jeweils die Attribute `team1` und `team2` durch ein `-` getrennt an.
- Falls als Ergebnis mehrere Elemente selektiert werden, trennen Sie die jeweiligen Ausgaben durch Leerzeichen.

Betrachten Sie dazu folgendes Beispiel:

```
//em:spiel[1] | //em:spiel[2]
```

```
ESP-ITA   IRL-CRO
```

Geben Sie nun die entsprechende Ausgaben der folgenden XPath-Abfragen an.

```
//em:team[2]
```

```
ITA SWE
```

```
(//em:team)[2]
```

```
ITA
```

```
//em:spiel[@tore1=@tore2]
```

```
ESP-ITA   FRA-ENG   ITA-CRO
```

```
//em:spiel[@team1=//em:land/@ref]
```

```
FRA-ENG   SWE-ENG   SWE-FRA   ENG-UKR
```

```
//em:team[@land=(//em:spiel[@tore1>2]/@team1 |
                 //em:spiel[@tore2>2]/@team2)]
```

```
ESP CRO ENG
```

Aufgabe 6: (9)

Erstellen Sie ein XSLT-Stylesheet **em.xsl**, das angewandt auf Dokumente der Gestalt **em.xml** eine Ausgabe folgender Form liefert:

```
<reports>
  <report>Spain vs. Italy: 2 goals</report>
  <report>Republic of Ireland vs. Croatia: 4 goals</report>
  ...
</reports>
```

Im Detail soll also:

- Als Dokument-Element ein Element `reports` erzeugt werden (im leeren Namespace).
- Für jedes Spiel (`em:spiel`) soll ein `report` Element erzeugt werden. Darin soll enthalten sein:
  - Der vollständige Name beider Teams (Inhalt der entsprechenden `em:team` Elemente).
  - Die Anzahl der Tore in diesem Spiel (Summe der Attribute `tore1` und `tore2`).

Vervollständigen Sie hier das XSLT-Stylesheet **em.xsl**. Die Verwendung von Kontrollstrukturen wie `xsl:for-each` ist für die Lösung grundsätzlich erlaubt, aber nicht erforderlich! Sie brauchen sich nicht um Whitespaces etc. zu kümmern (inbesondere sind die Zwischentexte wie z.B. " vs. " bzw. " goals" nicht unbedingt erforderlich – so kann Schreibaufwand gespart werden!)

Datei **em.xsl**:

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:em="em">

  <xsl:template match="/">
    <reports>
      <xsl:apply-templates select="//em:spiel"/>
    </reports>
  </xsl:template>

  <xsl:template match="em:spiel">
    <report>
      <xsl:value-of select="//em:team[@land=current()/@team1]"/>
      <xsl:text> vs. </xsl:text>
      <xsl:value-of select="//em:team[@land=current()/@team2]"/>
      <xsl:text>: </xsl:text>
      <xsl:value-of select="@tore1 + @tore2"/>
      <xsl:text> goals</xsl:text>
    </report>
  </xsl:template>

</xsl:stylesheet>
```

Aufgabe 7: (9)

Betrachten Sie folgende-XQuery Abfrage **em.xq**:

```
declare namespace em = "em";

for $g in //em:gruppe
return element {string($g/@name)} {
   for $t in $g/em:team
   let $s := //em:spiel[@team1=$t/@land or @team2=$t/@land]
   return element {string($t/@land)} {
      sum(($s/@tore1, $s/@tore2))
   }
}
```

Geben Sie nun die Ausgabe von **em.xq** angewandt auf **em.xml** an.

Die exakte Behandlung von Whitespaces ist für dieses Beispiel nicht relevant.

```
<C>
  <ESP>7</ESP>
  <ITA>6</ITA>
  <IRL>10</IRL>
  <CRO>7</CRO>
</C>
<D>
  <UKR>6</UKR>
  <SWE>10</SWE>
  <FRA>6</FRA>
  <ENG>8</ENG>
</D>
```

Aufgabe 8: (8)

Vervollständigen Sie den folgenden SAX XMLFilter `TorDifferenzFilter`, der angewandt auf Dokumente der Gestalt **em.xml** jedem `em:spiel` ein Kindelement `tdif` hinzufügt. Inhalt dieses Elements `tdif` soll die Tordifferenz (`tore1` - `tore2`) sein. Der Rest des Dokuments soll unverändert bleiben.

Um Fehlerbehandlung müssen Sie sich nicht kümmern. Betrachten Sie dazu folgendes (ausschnittsweises) Beispiel:

```
...
<em:spiel team1="ESP" team2="ITA" tore1="1" tore2="1">
  <tdif>0</tdif>
</em:spiel>
<em:spiel team1="IRL" team2="CRO" tore1="1" tore2="3">
  <tdif>-2</tdif>
</em:spiel>
...
```

```
public static class TorDifferenzFilter extends XMLFilterImpl {

  public void startElement(String namespaceURI, String localName,
    String qName, Attributes atts) throws SAXException {

    super.startElement(namespaceURI, localName, qName, atts);

    if ("spiel".equals(localName)) {
      int tore1 = Integer.parseInt(atts.getValue("tore1"));
      int tore2 = Integer.parseInt(atts.getValue("tore2"));

      super.startElement("", "tdif", "tdif", new AttributesImpl());
      char[] rollTotalArray = Integer.toString(tore1-tore2).toCharArray();
      super.characters(rollTotalArray, 0, rollTotalArray.length);
      super.endElement("", "tdif", "tdif");
    }
  }
}
```

Sie können diese Seite abtrennen!

Datei **em.xml**:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE em:gruppenphase SYSTEM "em.dtd">

<em:gruppenphase xmlns:em="em">

  <em:gruppe name="C">
    <em:team land="ESP">Spain</em:team>
    <em:team land="ITA">Italy</em:team>
    <em:team land="IRL">Republic of Ireland</em:team>
    <em:team land="CRO">Croatia</em:team>
  </em:gruppe>
  <em:gruppe name="D">
    <em:team land="UKR">Ukraine</em:team>
    <em:team land="SWE">Sweden</em:team>
    <em:team land="FRA">France</em:team>
    <em:team land="ENG">England</em:team>
  </em:gruppe>

  <em:spiel team1="ESP" team2="ITA" tore1="1" tore2="1"/>
  <em:spiel team1="IRL" team2="CRO" tore1="1" tore2="3"/>
  <em:spiel team1="UKR" team2="SWE" tore1="2" tore2="1"/>
  <em:spiel team1="FRA" team2="ENG" tore1="1" tore2="1"/>
  <em:spiel team1="ESP" team2="IRL" tore1="4" tore2="0"/>
  <em:spiel team1="ITA" team2="CRO" tore1="1" tore2="1"/>
  <em:spiel team1="SWE" team2="ENG" tore1="2" tore2="3"/>
  <em:spiel team1="UKR" team2="FRA" tore1="0" tore2="2"/>
  <em:spiel team1="CRO" team2="ESP" tore1="0" tore2="1"/>
  <em:spiel team1="ITA" team2="IRL" tore1="2" tore2="0"/>
  <em:spiel team1="SWE" team2="FRA" tore1="2" tore2="0"/>
  <em:spiel team1="ENG" team2="UKR" tore1="1" tore2="0"/>

  <em:bericht>Beim Gruppenspiel <em:land ref="FRA"/> gegen
    <em:land ref="ENG"/> wurden 2 Tore erzielt.</em:bericht>
  <em:bericht>Beim Gruppenspiel <em:land ref="SWE"/> gegen
    <em:land ref="ENG"/> wurden 5 Tore erzielt.</em:bericht>

</em:gruppenphase>
```

Gesamtpunkte: 75