

Gruppe A	PRÜFUNG AUS "SEMISTRUKTURIERTE DATEN" 181.135		22. 10. 2009
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 As (A*)>
<!ELEMENT A (B,C?)>
<!ELEMENT B (#PCDATA|D|E)*>
<!ELEMENT C (#PCDATA)>
<!ATTLIST C attr CDATA #REQUIRED>
<!ELEMENT D EMPTY>
<!ELEMENT E (#PCDATA)>
```

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

- Gehen Sie davon aus, dass allen folgenden Dateien die Zeile
`<!DOCTYPE As SYSTEM "test.dtd">`
 vorangestellt ist.
- 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. <code><As><A>abc</As></code> | gültig <input type="radio"/> | ungültig <input type="radio"/> |
| 2. <code><As><A><C attr="abc">def</C></As></code> | gültig <input type="radio"/> | ungültig <input type="radio"/> |
| 3. <code><As><A>abc<C attr="def">ghj</C>ikl</As></code> | gültig <input type="radio"/> | ungültig <input type="radio"/> |
| 4. <code><As><A>abc<A><D/>def</As></code> | gültig <input type="radio"/> | ungültig <input type="radio"/> |
| 5. <code><As><A>abc<E>def</E>ghj</As></code> | gültig <input type="radio"/> | ungültig <input type="radio"/> |
| 6. <code><As></As></code> | 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:

(10)

Nehmen Sie an, dass folgender Java Code (links) auf das angegebene XML-Dokument (rechts) ausgeführt wird. Zeichnen Sie den resultierenden DOM-Baum *nach* der Ausführung des Codes auf.

Bemerkung: Die Einrückungen im XML Dokument (Whitespaces) dienen nur der Darstellung und sollen *nicht* als Textknoten im Baum aufscheinen. `documentNode` bezieht sich auf das XML Dokument, `rootNode` bezieht sich auf das Wurzelement.

```
class Exam {
    public static void main() {
        Document documentNode; // DOCUMENT Node
        Node rootNode; // Wurzelement (root)
        // Gegeben: Dokument lesen etc...
        rootNode = documentNode.getDocumentElement();
        secret(rootNode);
        // Aufgabe: DOM Baum zeichnen
    }
    private static void secret(Node n) {
        NodeList nl = n.getChildNodes();
        for(int i=0; i < 3; i++) {
            Node child = nl.item(0); // ACHTUNG! 0, nicht "i"
            n.removeChild(child);
        }
    }
}
```

```
<?xml version="1.0"?>
<!--Author: ssd@dbai.tuwien.ac.at-->
<?exam?>
<dbai>
  <office>1</office>
  <office status="public">2</office>
  <office>
    <suboffice>a</suboffice>
    <suboffice>b</suboffice>
    <suboffice>c</suboffice>
  </office>
  <office>3</office>
  <office>4</office>
</dbai>
```



Aufgabe 3:

(6)

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

```
<?xml version="1.0" encoding="UTF-8"?>
<root xmlns="http://dbai.tuwien.ac.at" xmlns:tu="http://tuwien.ac.at">
  <first attr="wert"/>
  <tu:second xmlns:tu="http://tuwis.tuwien.ac.at"/>
</root>
```

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

1. Das Element **first** liegt im Namespace **http://dbai.tuwien.ac.at**. wahr falsch
2. Das Attribut **attr** liegt im leeren Namespace. wahr falsch
3. Das Element **tu:second** liegt im Namespace **http://tuwien.ac.at**. wahr falsch
4. Das Element **root** liegt im Namespace **http://dbai.tuwien.ac.at**. wahr falsch

Aufgabe 4:

(12)

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

1. `<e><text/>something<text/></e>` ist wohlgeformt. wahr falsch
2. Die DTD Elementdeklaration `(A?|B?|C?)+` ist gleichwertig mit `(A|B|C)*`. wahr falsch
3. In einer XML-Schema Datei muss der Defaultnamespace immer gleich dem Targetnamespace sein. wahr falsch
4. SAX verwendet die DOM API um ein XML Dokument zu lesen. wahr falsch
5. Der "parent" eines SAX-Filter kann selbst auch ein SAX-Filter sein. wahr falsch
6. Der XPath-Ausdruck `./a[@b=c]` ist die Kurzschreibweise des XPath-Ausdrucks `self::node()/child::a/attribute::b[c]` wahr falsch
7. Das Resultat eines XSLT-Stylesheet (angewandt auf ein wohlgeformtes XML-Dokument) ist immer ein wohlgeformtes XML-Dokument wahr falsch
8. Ein XSLT-Stylesheet muss nicht unbedingt ein wohlgeformtes XML-Dokument sein. wahr falsch

Geben Sie eine DTD **kino.dtd** an, sodass das XML-Dokument **kino.xml** (siehe Anhang) bezüglich dieser DTD gültig ist. Berücksichtigen Sie beim Erstellen der DTD folgende Punkte:

- Die Reihenfolge der Elemente soll wie in **kino.xml** angegeben fixiert werden.
- Das Wurzelement beinhaltet jeweils genau ein Element **standort**, **filme** und **spielplan**.
- Ein **saal** beinhaltet genau ein **name** Element.
- Ein **standort** beinhaltet ein oder mehrere **saal** Elemente.
- Ein **film** beinhaltet genau ein **name** und **regie** Element, jedoch zwei oder mehr **darsteller** Elemente.
- Alle weiteren Elemente können beliebig oft vorkommen.
- Alle Attribute sollen verpflichtend sein. Die Attribute **kennung** und **nr** sollen vom Typ ID sein, und **film** sowie **saal** sollen Referenzen auf die ID-Attribute sein.

Datei **kino.dtd**:

Aufgabe 6:

(12)

Betrachten Sie die folgenden drei XSLT-Stylesheets. Geben Sie jeweils den Output an, den das entsprechende Stylesheet angewandt auf **kino.xml** (siehe Anhang) liefert. Sie brauchen sich dabei nicht um Whitespaces etc. kümmern.

Anmerkung: Pro Teilaufgabe sind jeweils 4 Punkte erreichbar.

Nehmen Sie an dass jede Datei mit einem korrekten Header versehen ist. zB

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

Datei **query1.xsl**:

```
<xsl:output method="text" encoding="UTF-8" version="1.0" />
<xsl:template match="/kino/standort">
  <xsl:apply-templates/>
</xsl:template>
<xsl:template match="/kino/filme">
  <xsl:apply-templates select="./film/name"/>
</xsl:template>
<xsl:template match="/kino/spielplan">
  <!-- empty -->
</xsl:template>
<xsl:template match="/kino/spielplan/auffuehrung">
  <xsl:copy-of select="."/>
</xsl:template>
</xsl:stylesheet>
```

Geben Sie hier den Output von **query1.xsl** angewandt auf **kino.xml** an:

Datei **query2.xsl**:

```
<xsl:output method="text" encoding="UTF-8" version="1.0" />
<xsl:template match="/">
  <xsl:for-each select="/kino/filme/film">
    <xsl:variable name="curFilm" select="./@nr"/>
    <xsl:value-of select="./name"/>
    <xsl:for-each select="/kino/spielplan/auffuehrung[@film=$curFilm]">
      <xsl:value-of select="./@datum"/>
    </xsl:for-each>
  </xsl:for-each>
</xsl:template>
</xsl:stylesheet>
```

Geben Sie hier den Output von **query2.xsl** angewandt auf **kino.xml** an:

Datei **query3.xsl**:

```
<xsl:output method="xml" encoding="UTF-8" version="1.0" />
<xsl:template match="/">
  <xsl:for-each select="/kino/*[position()=1]/*">
    <xsl:if test="position()=last()">
      <xsl:copy-of select="."/>
    </xsl:if>
  </xsl:for-each>
</xsl:template>
</xsl:stylesheet>
```

Geben Sie hier den Output von **query3.xsl** angewandt auf **kino.xml** an:

Aufgabe 7:

(9)

Schreiben Sie einen **XMLFilter** der nur das **filme** Element **und dessen Subelemente** weiterreicht. Sie müssen also alle Events ignorieren außer Sie befinden sich innerhalb des **filme** Elements. Sie brauchen sich lediglich um Elemente und Attribute kümmern. Beispiel (Input → Output):

```
<kino>
  <standort>
    <!-- ... -->
  </standort>
  <filme>
    <film nr="f123">
      <!-- ... -->
    </film>
  </filme>
  <spielplan>
    <!-- ... -->
  </spielplan>
</kino>
```

```
<filme>
  <film nr="f123">
    <!-- ... -->
  </film>
</filme>
```

Tipp: Die Klasse `FilmPassFilter` erweitert `XMLFilterImpl`, dh. per default werden alle Events durchgereicht. Überschreiben Sie die relevanten Methoden und geben Sie ein Event nur dann an die Superklasse weiter (mittels `super...`) wenn Sie es behalten wollen.

Verwenden Sie eine Variable die anzeigt ob sie innerhalb des **filme** Elements sind.

```
class FilmPassFilter extends XMLFilterImpl {
```

```
}
```

Aufgabe 8:

(8)

Schreiben Sie XPath-Anfragen für folgende Aufgabenstellungen. Zu jeder Abfrage ist ein Beispiel mit der erwarteten Ausgabe (bezgl. kino.xml) angegeben. Die Abfragen sollen auf allen Dokumenten, die gültig bezgl. kino.dtd sind, funktionieren.

1. Geben Sie die Namen aller bekannten Säle aus.

```
<name>Audi Max</name>  
<name>Saal 2</name>  
<name>Saal 3</name>  
<name>Saal 4</name>
```

2. Selektieren Sie jene auffuehrung-Elemente aus, die keinen ort-Subelement haben.

```
<auffuehrung datum="2009-10-23" film="f123"/>
```

3. Geben Sie aus, wie oft der Film "f123" am "2009-10-22" gespielt wird (zählen Sie die ort Elemente).

2

4. Geben Sie alle Regisseure aus, die im "Audi Max" (Sie dürfen die Kennung "am" direkt verwenden) gespielt werden.

```
<regie>George Lucas</regie>  
<regie>Peter Jackson</regie>
```


Sie können diese Seite abtrennen!

kino.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE kino SYSTEM "kino.dtd">
<kino>
  <standort>
    <saal kennung="am"> <name>Audi Max</name> </saal>
    <saal kennung="s2"> <name>Saal 2</name> </saal>
    <saal kennung="s3"> <name>Saal 3</name> </saal>
    <saal kennung="s4"> <name>Saal 4</name> </saal>
  </standort>
  <filme>
    <film nr="f123">
      <name>Krieg der Sterne IV</name>
      <regie>George Lucas</regie>
      <darsteller>Mark Hamill</darsteller>
      <darsteller>Harrison Ford</darsteller>
    </film>
    <film nr="f124">
      <name>Der Herr der Ringe - Die Gefährten</name>
      <regie>Peter Jackson</regie>
      <darsteller>Elijah Wood</darsteller>
      <darsteller>Ian McKellen</darsteller>
    </film>
    <film nr="f125">
      <name>Der Pate</name>
      <regie>Francis Ford Coppola</regie>
      <darsteller>Marlon Brando</darsteller>
      <darsteller>Al Pacino</darsteller>
    </film>
  </filme>
  <spielplan>
    <auffuehrung datum="2009-10-22" film="f123">
      <ort uhrzeit="22:00" saal="am"/>
      <ort uhrzeit="22:30" saal="s4"/>
    </auffuehrung>
    <auffuehrung datum="2009-10-22" film="f124">
      <ort uhrzeit="19:15" saal="am"/>
      <ort uhrzeit="21:00" saal="s2"/>
      <ort uhrzeit="20:45" saal="s3"/>
    </auffuehrung>
    <auffuehrung datum="2009-10-22" film="f125">
      <ort uhrzeit="20:15" saal="s3"/>
    </auffuehrung>
    <auffuehrung datum="2009-10-23" film="f123"/>
  </spielplan>
</kino>
```

Gesamtpunkte: 75