Gruppe A

Bitte tragen Sie **sofort** und **leserlich** Namen, Studienkennzahl und Matrikelnummer ein und legen Sie Ihren Studentenausweis bereit.

PRÜFUNG AUS "SEMISTRUKTURIERTE DATEN" **181.135**			17. 06. 2008
Kennnr.	Matrikelnr.	Familienname	Vorname

Arbeitszeit: 120 Minuten. Aufgaben sind auf den Angabeblättern zu lösen; Zusatzblätter werden nicht gewertet.

Aufgabe 1: (12)

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

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

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

- Gehen Sie davon aus, dass allen folgenden Dateien die Zeile
  `<!DOCTYPE A 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. `<A><B K="A1">abc</B><C>cde<D>fgh</D></C></A>` gültig ◯ ungültig ⊗
2. `<A><B K="A1"><C>abc</C><C>fgh</C></B></A>` gültig ⊗ ungültig ◯
3. `<A><B K="A1"><C>abc</C></B><C L="A2">def</C></A>` gültig ◯ ungültig ⊗
4. `<A><C><E/><E></E></C></A>` gültig ⊗ ungültig ◯
5. `<A><C><E><E/></E></C></A>` gültig ◯ ungültig ⊗
6. `<A><B K="A1"><C><D>abc</D></C><B K="A2"><C>def</C></B></B></A>` gültig ⊗ ungültig ◯
7. `<A><B K="A1"><C><D>abc</D></C></B><B K="A2"><C>def</C></B></A>` gültig ⊗ ungültig ◯
8. `<A><C></C></A>` 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: (15)

Geben Sie auf den nächsten beiden Seiten eine XML Schema Definition **giro.xsd** an, sodass das XML-Dokument **giro.xml** (siehe Anhang) bezüglich dieses Schemas gültig ist.

Die Datei **giro.xml** enthält Information zu den Teams die beim Radrennen Giro d'Italia starten. Berücksichtigen Sie dabei:

- Im Wurzelelement "giro" sollen beliebig viele (aber mindestens ein) "team"-Element vorkommen. Nach den "team"-Elementen, folgt ein "bemerkung"-Element.
- Das "name"-Attribut des Team-Elements ist verpflichtend und vom Typ String.
- Ein Team besteht aus genau neun Fahrern. Der Name des Fahrers ist als Inhalt eines "fahrer"-Elements angegeben und kann ein beliebiger String sein.
- Für jeden Fahrer kann optional ein "typ"-Attribut angegeben werden. Stellen Sie sicher, dass nur folgende Attributwerte zulässig sind: "Kapitaen", "Sprinter" und "Bergfahrer".
- Das "bemerkung"-Element hat gemischten Inhalt und kann beliebig viele (auch keine) Subelemente "name", "team", sowie maximal ein Subelement "korrektur" beinhalten. Die Reihenfolge des Auftretens der einzelnen Subelemente soll beliebig sein. "name", "team" und "korrektur" sollen einfache Elemente mit Stringinhalt sein.

Sie können sich (müssen aber nicht) neue Typen explizit definieren.

Datei **giro.xsd**:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:element name="giro">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="team" maxOccurs="unbounded">
     <xs:complexType>
      <xs:sequence>
       <xs:element name="fahrer" minOccurs="9" maxOccurs="9">
        <xs:complexType>
         <xs:simpleContent>
          <xs:extension base="xs:string">
           <xs:attribute name="typ">
            <xs:simpleType>
             <xs:restriction base="xs:string">
              <xs:enumeration value="Kapitaen"/>
              <xs:enumeration value="Sprinter"/>
              <xs:enumeration value="Bergfahrer"/>
             </xs:restriction>
            </xs:simpleType>
           </xs:attribute>
          </xs:extension>
         </xs:simpleContent>
        </xs:complexType>
       </xs:element>
      </xs:sequence>
      <xs:attribute name="name" type="xs:string" use="required"/>
     </xs:complexType>
    </xs:element>
    <xs:element name="bemerkung">
     <xs:complexType mixed="true">
      <xs:sequence>
       <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element name="name" type = "xs:string" />
        <xs:element name="team" type = "xs:string" />
       </xs:choice>
       <xs:element name="korrektur" type = "xs:string" minOccurs="0" />
        <xs:choice minOccurs="0" maxOccurs="unbounded">
         <xs:element name="name" type = "xs:string" />
         <xs:element name="team" type = "xs:string" />
        </xs:choice>
       </xs:sequence>
      </xs:complexType>
     </xs:element>
    </xs:sequence>
   </xs:complexType>
  </xs:element>
</xs:schema>
```

Datei **giro.xsd** (Fortsetzung).

Aufgabe 3: (12)

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

Anmerkung: Pro Teilaufgabe sind jeweils 4 Punkte erreichbar.

Datei **query1.xsl**:

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

<xsl:template match="/">
   <teams>
      <xsl:apply-templates select="//team/@*"/>
   </teams>
</xsl:template>


</xsl:stylesheet>
```

Vervollständigen Sie hier den Output von **query1.xsl** angewandt auf **giro.xml**:

```
<?xml version="1.0" encoding="UTF-8"?>

<teams>CofidisBarloworld</teams>
```

Datei **query2.xsl**:

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

<xsl:template match="/">
        <fahrer><xsl:apply-templates/></fahrer>
</xsl:template>

<xsl:template match="bemerkung/name">
   <xsl:value-of select="."/>
</xsl:template>

<xsl:template match="text()"/>

</xsl:stylesheet>
```

Vervollständigen Sie hier den Output von **query2.xsl** angewandt auf **giro.xml**:

```
<?xml version="1.0" encoding="UTF-8"?>

<fahrer>Alberto Contador</fahrer>
```

Fortsetzung Beispiel 3.

Datei **query3.xsl**:

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


<xsl:template match="/">
   <fahrer>
   <xsl:for-each select="giro/team">
     <xsl:for-each select="fahrer">
       <xsl:if test="position()=last()">
         <xsl:value-of select="."/>
         <xsl:text>: </xsl:text>
         <xsl:value-of select="@typ"/>
        </xsl:if>
      </xsl:for-each>
      <xsl:element name="br"/>
   </xsl:for-each>
   </fahrer>
</xsl:template>

</xsl:stylesheet>
```

Vervollständigen Sie hier den Output von **query3.xsl** angewandt auf **giro.xml**:

```
<?xml version="1.0" encoding="UTF-8"?>

<fahrer>Nick Nuyens: Sprinter<br/>Geraint Thomas: Bergfahrer<br/></fahrer>
```

Aufgabe 4: (7)

Geben Sie die Eventnamen plus relevanten Inhalt (z.B. Text), wie sie von einem SAX Parser gefeuert werden, für das folgende XML Dokument an. Gehen Sie davon aus, dass jeweils nur ein characterEvent pro Textknoten gefeuert wird (also der gesamte Textinhalt auf einmal übergeben wird). Sie können die Events auch abkürzen (zB characterEvent = cE). Beispiel: "characterEvent beispieltext" bzw. "cE beispieltext"

```
<dok><wort lang="de"><!--not widely known-->tantieme</wort><wort lang="en">royality</wort></dok>
```

startDocument	
startElement	dok
startElement	wort [lang: de]
characterEvent	tantieme
endElement	wort
startElement	wort [lang: en]
characterEvent	royality
endElement	wort
endElement	dok
endDocument	

Aufgabe 5: (9)
Vervollständigen Sie die folgende Java Methode sodass für alle Elemente die Anzahl der Attribute eines übergebenen XML Dokuments auf die Konsole ausgegeben wird (zB "elementname: anzahl").

- Sie brauchen **keine Fehlerbehandlung** implementieren.

Tipp: Navigieren Sie **rekursiv** mithilfe der Methoden aus den DOM-Folien durch den Baum.

```
void printNumberOfAttributes(Node node) throws Exception {

  if(node.getNodeType() == Node.ELEMENT_NODE) {
    System.out.println(node.getNodeName() + ": " + node.getAttributes().getLength());
  }
  NodeList nl = node.getChildNodes();
  for(int i=0; i < nl.getLength(); ++i) {
    Node n = nl.item(i);
    printNumberOfAttributes(n);
  }
}

}
```

Aufgabe 6: (8)
Schreiben Sie XPath-Anfragen um für XML-Dokumente wie **giro.xml** (siehe Anhang) folgende Informationen zu selektieren:

1. Die Namen der Teams, in denen zumindest ein "Bergfahrer" und ein "Sprinter" fahren.
   `/giro/team[fahrer[@typ='Bergfahrer'] and fahrer[@typ='Sprinter']]/@name`

2. Die Namen jener Fahrer in den Teamlisten fuer die kein Attribut "typ" angegeben ist.
   `//fahrer[not (@typ)]`

3. Den Teil aus dem "bemerkung"-Element, der nach dem "korrektur"-Element steht (also im Fall von **giro.xml** soll als Stringwert "Favorit auf den Gesamtsieg: Alberto Contador vom Team Astana" Resultat der Abfrage sein).
   `//korrektur/following-sibling::node()`

4. Alle Fahrer der Teams die innerhalb des "bemerkung"-Elements mittels eines "team"-Elements erwähnt werden.
   `//fahrer[../@name=//bemerkung/team]`

Aufgabe 7: (12)
Kreuzen Sie an, ob die folgenden Aussagen wahr oder falsch sind.

1. Namespace-Attribute wie z.B. `xmlns` müssen in DTDs extra spezifiziert werden	wahr ⊗	falsch ◯
2. Wenn der Defaultnamespace nicht angegeben wird, ist in einer XML-Schema Datei der Defaultnamespace immer gleich dem Targetnamespace.	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 ist nicht immer ein wohlgeformtes XML-Dokument	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)

XML-Dokument **giro.xml**:

```
<?xml version="1.0" encoding="UTF-8"?>
<giro
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation="giro.xsd">
   <team name="Cofidis">
      <fahrer typ="Kapitaen"  >Mickael Buffaz</fahrer>
      <fahrer                 >Kevin de Weert</fahrer>
      <fahrer typ="Sprinter"  >Bingen Fernandez</fahrer>
      <fahrer typ="Sprinter"  >Nicolas Hartmann</fahrer>
      <fahrer                 >Steve Zampieri</fahrer>
      <fahrer                 >Yann Huguet</fahrer>
      <fahrer typ="Sprinter"  >Damien Monier</fahrer>
      <fahrer                 >Rik Verbrugghe</fahrer>
      <fahrer typ="Sprinter"  >Nick Nuyens</fahrer>
   </team>
   <team name="Barloworld">
      <fahrer typ="Kapitaen"  >Francesco Bellotti</fahrer>
      <fahrer                 >Patrick Calcagni</fahrer>
      <fahrer typ="Sprinter"  >Felix Rafael Cardenas</fahrer>
      <fahrer typ="Sprinter"  >Steve Cummings</fahrer>
      <fahrer typ="Sprinter"  >Enrico Gasparotto</fahrer>
      <fahrer                 >Christian Pfannberger</fahrer>
      <fahrer                 >Carlo Scognamiglio</fahrer>
      <fahrer                 >Mauricio Soler</fahrer>
      <fahrer typ="Bergfahrer">Geraint Thomas</fahrer>
   </team>

   <!-- hier koennen weitere Teams folgen -->

   <bemerkung>
      Die Startliste umfasst 22 Teams. Darunter auch <team>Barloworld</team>...
      <korrektur>
         Korrekturen seit der ersten Version der Startliste: wegen kurzfristiger Verletzungen etc.:
         Morabito fuer Murawjew (Astana),
         de Weert und Zampieri fuer Duclos-Lassalle und Heijboer (Cofidis).
      </korrektur>
      Favorit auf den Gesamtsieg: <name>Alberto Contador</name> vom Team <team>Astana</team>.
   </bemerkung>
</giro>
```

Gesamtpunkte: 75