

Informationssysteme - Übungsblätter

Übungsskriptum

Dipl.-Ing. Paul Panhofer BSc.^{1*}

1 ZID, TU Wien, Taubstummengasse 11, 1040, Wien, Austria

Abstract: Die Verbesserung der LEISTUNGSFÄHIGKEIT von **Rechnersystemen** und die Fortschritte in der NETZWERK-TECHNOLOGIE haben in den letzten Jahren zu einer verstärkten **Dezentralisierung** von Diensten und Daten in Anwendungssystemen geführt. Zusammen mit dieser Entwicklung haben die Anstrengungen bei der STANDARDISIERUNG von **Kommunikationsdiensten** dazu beigetragen, daß Systeme verschiedener Hersteller miteinander verbunden werden können. Die Verbindungen werden hier zur *Datenübertragung* und zum *elektronischen Nachrichtenaustausch* benutzt.

Durch die Verbindung von leistungsstarken ARBEITSPLATZRECHNERN mittels NETZWERKEN konnte eine **Client/Server-Architektur** geschaffen werden. Manchesmal ist die Aufteilung des Anwendungssystems in Clients und Server für den Benutzer zu erkennen. Gegenwärtig versucht man verschiedene bestehende verteilte Anwendungssysteme zu einem einzigen verteilten Anwendungssysteme zusammenzufassen, um den Informationsaustausch in großen Unternehmen und Organisationen effizienter zu gestalten.

Als Ergebnis erhält man hochintegrierte Anwendungssysteme, die mit dem Begriff **Enterprise computing** bezeichnet werden

MSC: paul.panhofer@gmail.com

Keywords:

Contents

*E-mail: paul.panhofer@tuwien.ac.at

Übungsblatt: XPath Kurzform ▼



Beispielbeschreibung ▼

- **Kompetenz:** SEMISTRUKTURIERTE DATEN, XML
- **Schwerpunkt:** XPATH
- **Datum:** WOCHE 09.09 - 13.09

1.Beispiel) XPath Ausdrücke - euro.txt ■

Geben Sie für die folgende Aufgaben die entsprechenden XPath Lokalisierungspfade an. Verwenden Sie die `euro.xml` Datei als Ausgangspunkt.

► Aufgabenstellung: XPath Lokalisierungspfade ▼

```

1  -- -----
2  -- XPath Kurzform: euro.xml
3  -- -----
4  -- Adressieren Sie die Nachnamen aller Abwehr-
5  -- spieler der franzoesischen Mannschaft.
6
7  -- Adressieren Sie die Namen der Klubs in denen
8  -- die spanischen Stuermer spielen.
9
10 -- Adressieren Sie die Namen der Teams die
11 -- ins Semifinale eingezogen sind.
12
13 -- Adressieren Sie fuer jedes Team den
14 -- ersten Mittelfeldspieler.
15
16 -- Geben Sie an wieviele Spieler der
17 -- Weltmeisterschaft fuer Real Madrid spielen.
18
19 -- Adressieren Sie alle Teams in der Gruppen
20 -- phase, die die Gruppenphase uerberstanden
21 -- haben.
22 -- Hinweis: Verwenden Sie den . Operator um
23 -- auf den gegenwaertigen Knoten zuzugreifen
24
25 -- Geben Sie fuer das portugisische Team die
26 -- Anzahl der Spiele nach der Gruppenphase an.
27 -- Hinweis: Verwenden Sie den * Operator um
28 -- auf alle Kindelemente eines Knotens
29 -- zuzugreifen.
30
31 -- Geben Sie die Namen alle Gegner des
32 -- portugisischen Teams in der Gruppenphase
33 -- aus.
```



2.Beispiel) XPath Ausdrücke - travel.txt ■

Geben Sie für die folgenden Aufgaben die entsprechenden XPath Lokalisierungspfade an. Verwenden Sie die `travel.xml` Datei als Ausgangspunkt.

► Aufgabenstellung: XPath Lokalisierungspfade ▼

```

1  -- -----
2  -- XPath Kurzform: travel.xml
3  -- -----
4  -- Adressieren Sie die Namen aller Sommercamps
5  -- die bereits gebucht worden sind.
6
7  -- Adressieren Sie die Namen aller Camps
8  -- die bereits gebucht worden sind.
9  -- Hinweis: Verwenden Sie den * Operator um
10 -- auf alle Kindelemente eines Knotens
11 -- zuzugreifen.
12
13 -- Adressieren Sie das alle Sportcamps fuer
14 -- die ein Buchungsdatum bekannt ist.
15
16 -- Adressieren Sie das erste Sportcamps fuer
17 -- das ein Buchungsdatum bekannt ist.
18
19 -- Adressieren Sie alle Sportcamps fuer die
20 -- kein Buchungsdatum bekannt ist.
```



Übungsblatt: XPath Standardform ▾



Beispielbeschreibung ▾

- **Kompetenz:** SEMISTRUKTURIERTE DATEN, XML
- **Schwerpunkt:** XPATH
- **Datum:** WOCHE 16.09 - 20.09

1.Beispiel) XPath Ausdrücke - travel.txt ■

Geben Sie für die folgenden XPath Lokalisierungspfade die entsprechenden XPath Lokalisierungspfade in Standardform an. Verwenden Sie die `travel.xml` Datei als Ausgangspunkt.

▸ Aufgabenstellung: XPath Lokalisierungspfade ▾

```

1  -----
2  -- XPath Standardform: travel.xml
3  -----
4  /travelAgency/sportcamps/summercamp
5
6  //summercamp/@id
7
8  /travelAgency/.
9
10 //summercamp/*
11
12 //sportcamps/*[@id = //booking[@date]/@campid]
13
14 //customer[@id = //booking/@customerid]/text()
```



2.Beispiel) XPath Ausdrücke - sport.txt ■

Geben Sie für die folgenden Aufgaben die entsprechenden XPath Lokalisierungspfade an. Verwenden Sie die `sport.xml` Datei als Ausgangspunkt.

Hinweis: Die `sport.xml` Datei beschreibt einen Katalog von Sportarten.

- **Sportarten** sind Elemente die keine anderen Elementen enthalten.
- **Katalogeinträge** sind keine Sportarten. Sie sind daran zu erkennen dass Sie andere Elemente enthalten.
- 2 Sportarten befinden sich im selben Katalog wenn sie sich zumindestens einen Katalogeintrag teilen - `<sports>` ausgenommen.

▸ Aufgabenstellung: XPath Lokalisierungspfade ▾

```

1  -----
2  -- XPath Standardform: sport.xml
3  -----
4  -- Adressieren Sie alle Sportart
5
6  -- Adressieren Sie alle Katalogeintraege denen
7  -- die Sportart <super-g> zugeordnet ist.
8
9  -- Wieviele Sportarten sind im Winterspiele
10 -- Katalog <snow-sports> zu finden?
11
12 -- Wieviele Katalogeintraege befinden sich im
13 -- gesamten Katalog
14
15 -- Adressieren Sie alle Sommersportarten
16 -- die keine Ballspiele sind.
17
18 -- Adressieren Sie alle Sportarten die sich nicht
19 -- im selben Katalog wie <ski-run> befinden.
```



3.Beispiel) XPath Ausdrücke - result.txt ■

Geben Sie für die folgenden Aufgaben die entsprechenden XPath Lokalisierungspfade an. Verwenden Sie die `result.xml` Datei als Ausgangspunkt.

▸ Aufgabenstellung: XPath Lokalisierungspfade ▾

```

1  -----
2  -- XPath Standardform: result.xml
3  -----
4  -- Adressieren Sie fuer jede Gruppe das team
5  -- das am weitesten gekommen ist - die meisten
6  -- Runden hinter sich hat.
```



Aufgabenblatt: XSLT Stylesheets ▾



Aufgabenbeschreibung ▾

SCHREIBEN EINFACHER XSLT PROGRAMME, DEKLARATIVE PROGRAMMIERUNG

1.Beispiel) XSLT Stylesheet - team.xml ■

Schreiben Sie ein XSLT Stylesheet, das die **euro.xml** Datei transformiert. Die gewünschte Output Datei wird im Beispiel angegeben.

► Aufgabenstellung: Output Datei ▾

```

1  <!-- ----- -->
2  <!--           XSLT Output: team.xml           -->
3  <!-- ----- -->
4  <?xml version="1.0" encoding="UTF-8"?>
5  <teams>
6
7      <team id="FRA">
8          <player>Laurent Koscielny</player>
9          <player>Djibril Sidibe</player>
10         <player>Antoine Griezmann</player>
11         <player>Olivier Giroud</player>
12         <player>Nabil Fekir</player>
13         ...
14     </team>
15
16     <team id="SPA">
17         <player>Jordi Alba</player>
18         <player>Dani Carvajal</player>
19         <player>Sergio Ramos</player>
20         <player>Alberto Moreno</player>
21         ...
22     </team>
23
24     <team id="GER">
25         <player>Jerome Boateng</player>
26         <player>Marcel Halstenberg</player>
27         <player>Mats Hummels</player>
28         ...
29     </team>
30
31     ...
32
33     <team id="IRL">
34         <player>Daren Randolph</player>
35         <player>Keiren Westwood</player>
36         <player>Alex Pearce</player>
37         <player>Jonny Hayes</player>
38         <player>Harry Arter</player>
39         ...
40     </team>
41
42 </teams>

```

2.Beispiel) XSLT Stylesheet - positions.xml ■

Schreiben Sie ein XSLT Stylesheet, das die **euro.xml** Datei transformiert. Die gewünschte Output Datei wird im Beispiel angegeben.

► Aufgabenstellung: Output Datei ▾

```

1  <!-- ----- -->
2  <!--           XSLT Output: positions.xml           -->
3  <!-- ----- -->
4  <?xml version="1.0" encoding="UTF-8"?>
5  <positions>
6
7      <position id="defense">
8          <team code="FRA">
9              <player>Laurent Koscielny</player>
10             <player>Djibril Sidibe</player>
11             </team>
12          <team code="SPA">
13              <player>Jordi Alba</player>
14              <player>Dani Carvajal</player>
15              ...
16          </team>
17          ...
18      </position>
19      <position id="playground">
20          <team code="FRA"/>
21          <team code="SPA">
22              <player>Marco Asensio</player>
23              <player>Andres Iniesta</player>
24              <player>David Silva</player>
25              <player>Javi Martinez</player>
26              <player>Ander Herrera</player>
27          </team>
28          <team code="GER">
29              <player>Julian Brandt</player>
30              <player>Emre Can</player>
31              <player>Sami Khedira</player>
32              <player>Toni Kroos</player>
33              <player></player>
34          </team>
35          ...
36      </position>
37      <position id="attack">
38          <team code="FRA">
39              <player>Antoine Griezmann</player>
40              <player>Olivier Giroud</player>
41              <player>Nabil Fekir</player>
42              <player>Florian Thauvin</player>
43              <player>Alexnadre Lacazette</player>
44          </team>
45          <team code="SPA">
46              <player>David Villa</player>
47              <player>Diego Costa</player>
48              ...
49          </team>
50          ...
51      </position>
52
53 </positions>

```

3.Beispiel) XSLT Stylesheets - clubs.xsl

Schreiben Sie ein XSLT Stylesheet, das die `euro.xml` Datei transformiert. Die gewünschte Output Datei wird im Beispiel angegeben.

► Aufgabenstellung: Output Datei ▼

```

1  <!-- ----- -->
2  <!--      XSLT Output: clubs.xml      -->
3  <!-- ----- -->
4  <?xml version="1.0" encoding="UTF-8"?>
5  <clubs>
6
7  <club id="Arsenal London" playercount="1">
8    <player code="FRA">Laurent Koscielny</player>
9  </club>
10
11 <club id="AS Monaco" playercount="1">
12   <player code="FRA">Djibril Sidibe</player>
13   <player code="PLA">Kamil Glik</player>
14 </club>
15
16 <club id="Atletico Madrid" playercount="1">
17   <player code="FRA">Antoine Griezmann</player>
18 </club>
19
20 <club id="Arsenal" playercount="1">
21   <player code="FRA">Olivier Giroud</player>
22 </club>
23
24 <club id="Lyon" playercount="2">
25   <player code="FRA">Nabil Fekir</player>
26   <player code="FRA">Alexandre
27     Lacazette</player>
28 </club>
29
30 <club id="Marseille" playercount="1">
31   <player code="FRA">Florian Thauvin</player>
32 </club>
33
34 <club id="FC Barcelona" playercount="4">
35   <player code="SPA">Jordi Alba</player>
36   <player code="SPA">Andres Iniesta</player>
37   <player code="SPA">David Villa</player>
38   <player code="GER">Jerome Boateng</player>
39 </club>
40
41 <club id="Real Madrid" playercount="6">
42   <player code="SPA">Dani Carvajal</player>
43   <player code="SPA">Sergio Ramos</player>
44   <player code="SPA">Marco Asensio</player>
45   <player code="SPA">Luacas Vazquez</player>
46   <player code="GER">Toni Kroos</player>
47   <player code="CRO">Luka Modric</player>
48 </club>
49 ...
50
51 </clubs>

```



Übungsblatt: XSLT Stylesheets



Beispielbeschreibung

- **Kompetenz:** SEMISTRUKTURIERTE DATEN, XSLT
- **Schwerpunkt:** XSLT STYLESHEETS
- **Datum:** WOCHE 23.09 - 27.09

1.Beispiel) XSLT Stylesheet - result.xml

Schreiben Sie ein XSLT Stylesheet, das die `euro.xml` Datei transformiert. Die gewünschte Output Datei wird im Beispiel angegeben.

► Aufgabenstellung: Output Datei ▼

```

1 <!-- ----- -->
2 <!--           XSLT Output: result.xml      -->
3 <!-- ----- -->
4 <?xml version="1.0" encoding="UTF-8"?>
5 <result>
6
7   <group id="1">
8     <team code="FRA" round="5"/>
9     <team code="SPA" round="2"/>
10    <team code="GER" round="4"/>
11    <team code="GBS" round="2"/>
12    <team code="POR" round="5"/>
13    <team code="BEL" round="3"/>
14  </group>
15
16  <group id="2">
17    <team ...>
18    ...
19
20    ...
21  </group>
22
23  <group id="3">
24    <team ...>
25    ...
26
27    ...
28  </group>
29
30  <group id="4">
31    <team code="TUR" round="1"/>
32    <team code="IRL" round="2"/>
33    <team code="ISL" round="3"/>
34    <team code="WAL" round="4"/>
35    <team code="ALB" round="1"/>
36    <team code="NIR" round="2"/>
37  </group>
38
39 </result>

```

2.Beispiel) XSLT Stylesheet - stats.xml

Schreiben Sie ein XSLT Stylesheet, das die `euro.xml` Datei transformiert. Die gewünschte Output Datei wird im Beispiel angegeben.

► Aufgabenstellung: Output Datei ▼

```

1 <!-- ----- -->
2 <!--           XSLT Output: stats.xml      -->
3 <!-- ----- -->
4 <?xml version="1.0" encoding="UTF-8"?>
5 <stats>
6
7   <team group="1" code="FRA">
8     <name>France</name>
9     <!-- Geben Sie alle Gegner der Gruppen- -->
10    <!-- phase an. Hat die Mannschaft die . -->
11    <!-- Gruppenphase ueberstanden sollen . -->
12    <!-- die Gegner der spaeteren Spiele . -->
13    <!-- ebenfalls angegeben werden.      . -->
14    <opponents>
15      <team code="SPA"/>
16      <team code="POR"/>
17      <team code="BEL"/>
18      <team code="IRL"/>
19      ...
20      <team code="GER"/>
21    </opponents>
22  </team>
23
24  <team group="1" code="SPA">
25    ...
26  </team>
27
28  <team group="1" code="GBS">
29    ...
30  </team>
31
32  <team group="1" code="GBS">
33    ...
34  </team>
35
36  ...
37  <team group="1" code="POR">
38    <name>Portugal</name>
39    <opponents>
40      <team code="FRA"/>
41      <team code="GBS"/>
42      <team code="BEL"/>
43      <team code="CRO"/>
44      ...
45      <team code="WAL"/>
46    </opponents>
47  </team>
48  ...
49  <team group="4" code="NIR">
50    ...
51  </team>
52
53 </stats>

```

3.Beispiel) XSLT Stylesheet - report.xsl

Schreiben Sie ein XSLT Stylesheet, das die `euro.xml` Datei transformiert. Die gewünschte Output Datei wird im Beispiel angegeben.

► Hint: Programmierung ▼

- Versuchen Sie mit so wenig `<xsl:template>` Elementen wie möglich auszukommen.
- Verwenden Sie das `<xsl:element>` Element!

► Aufgabenstellung: Output Datei ▼

```

1  <!-- ----- -->
2  <!--      XSLT Output: report.xml      -->
3  <!-- ----- -->
4  <?xml version="1.0" encoding="UTF-8"?>
5  <report>
6
7      <group>
8          <group-team code="FRA">
9              <name>France</name>
10             </group-team>
11         </group>
12
13         <last-sixteen>
14             <team-sixteen code="SUI">
15                 <name>Switzerland</name>
16             </team-sixteen>
17             ...
18
19         </last-sixteen>
20
21         <last-eight>
22             <team-eight code='PLA'>
23                 <name>.. </name>
24             </team-eight>
25             ...
26
27         </last-eight>
28
29         <last-four>
30             <team-four code="POR">
31                 <name>... </name>
32             </team-four>
33             ...
34
35         </last-four>
36
37         <last-two>
38             <finalist code="POR">
39                 <name> ... </name>
40             </finalist>
41             ...
42
43         </last-two>
44
45 </report>

```

□

Übungsblatt: XSLT Stylesheets



Beispielbeschreibung

- **Kompetenz:** SEMISTRUKTURIERTE DATEN, XSLT
- **Schwerpunkt:** XSLT STYLESHEETS
- **Datum:** WOCHE 30.09 - 04.10

1.Beispiel) XSLT Stylesheet - magazine.xml

Schreiben Sie ein XSLT Stylesheet, das die `report.xml` Datei transformiert. Die gewünschte Output Datei wird im Beispiel angegeben.

► Hint: Programmierung

- Für XSLT Suchmuster in Templaterregeln sind Sie für Knotenabfragen nicht auf Elementvergleiche beschränkt! Verwenden Sie falls sinnvoll auch Knotentypentests.
- Sie können mehrere Templaterregel mit demselben Suchmuster definieren. Definieren Sie in so einem Fall das `mode` Attribut.

► Aufgabenstellung: Output Datei

```

1  <!-- ----- -->
2  <!--      XSLT Output: magazine.xml      -->
3  <!-- ----- -->
4  <?xml version="1.0" encoding="UTF-8"?>
5  <magazine>
6    <report>
7      <head>
8        <authors>
9          <author>Arthur D. Dent</author>
10         <author>Ford Perfect</author>
11         <author>J.J. R. Tolkien</author>
12       </authors>
13       <pages>23</pages>
14       <url>...</url>
15     </head>
16     <body>
17       <authors>
18         This report is co-authored by
19         Arthur D. Dent and Ford Perfect,
20         sampled by J.J. R. Tolkien.
21       </authors>
22       <content>
23         Simcenter is a unified, scalable, open
24         ...
25       </content>
26     </body>
27   </report>
28 </magazine>
```

2.Beispiel) XSLT Stylesheet - familytree.xml

Schreiben Sie ein XSLT Stylesheet, das die `euro.xml` Datei transformiert. Die gewünschte Output Datei wird im Beispiel angegeben.

► Aufgabenstellung: Output Datei

```

1  <!-- ----- -->
2  <!--      XSLT Output: familytree.xml      -->
3  <!-- ----- -->
4  <?xml version="1.0" encoding="UTF-8"?>
5  <familytree>
6    <person>
7      <name>John Washington</name>
8      <child>
9        <person>
10         <name>Lawrence Washington</name>
11         <child>
12           <person>
13             <name>August Washington</name>
14             <child>
15               <person>
16                 <name>
17                   George Washington
18                 </child>
19               </name>
20             </person>
21           </child>
22         </person>
23       <child>
24         </person>
25       </child>
26     </person>
27   <person>
28     <name>Anne Pope</name>
29     <child>
30       <person>
31         <name>Lawrence Washington</name>
32         <child>
33           <person>
34             <name>August Washington</name>
35             <child>
36               <person>
37                 <name>
38                   George Washington
39                 </child>
40               </name>
41             </person>
42           </child>
43         </person>
44       <child>
45         </person>
46       </child>
47     </person>
48 </familytree>
```



Aufgabenblatt: XML Schema ▼



Aufgabenbeschreibung ▼

FORMULIEREN VON XML SCHEMAS FÜR XML DATEIEN.

1.Beispiel) XML Schema - person.xsd

Schreiben Sie für die folgende XML Datei XML Schemas.

► Hint: komplexe Datentypen ▼

- Jedes der Elemente der **person.xml** Datei darf nur einmal vorkommen.
- Namen müssen mindesten 2 Zeichen haben. Ein Name kann nicht länger als 50 Zeichen sein.
- Schreiben Sie einen eigenen Datentypen für um Namen zu beschreiben.

► Aufgabenstellung: XML Datei ▼

```

1  <!-- ----- -->
2  <!--      XML Datei: person.xml      -->
3  <!-- ----- -->
4  <?xml version="1.0" encoding="UTF-8"?>
5  <person xmlns:xs=
6      "http://www.w3.org/2001/XMLSchema-instance"
7      xs:noNamespaceSchemaLocation="person.xsd">
8      <first-name>Albert</first-name>
9      <middle-name>F.</middle-name>
10     <last-name>Alfons</last-name>
11 </person>
12
13 <?xml version="1.0" encoding="UTF-8"?>
14 <xs:schema
15     xmlns:xs="http://www.w3.org/2001/XMLSchema"
16     ...
17 </xs:schema>

```



2.Beispiel) XML Schema - euro.xsd

Schreiben Sie für die **euro.xml** Datei ein XML Schema.

Übungsblatt: XML Schema



Beispielbeschreibung

- **Kompetenz:** SEMISTRUKTURIERTE DATEN, XML SCHEMA
- **Schwerpunkt:** XML SCHEMA
- **Datum:** WOCHE 07.10 - 11.10

1.Beispiel) XML Schema - history.xsd

Schreiben Sie für die `history.xml` Datei ein XML Schema.

► Hint: XML Struktur ▼

- Ein `<periods>` Element kann mehrere `<period>` Elemente enthalten.
- Ein `<period>` Element kann `<leaders>` und `<battles>` Elemente enthalten.
- Legen Sie für die Werte des `spec` Attributs einen eigenen Datentyp an. Der Datentyp soll nur die Werte AC,BC enthalten.

► Aufgabenstellung: history.xml ▼

```

1  <!-- ----- -->
2  <!--      XSLT Output: history.xml      -->
3  <!-- ----- -->
4  <?xml version="1.0" encoding="UTF-8"?>
5  <history>
6      <periods ordered="false">
7          <period>
8              <description>...</description>
9              <timeperiod spec="BC">
10                 <begin>500</begin>
11                 <end>0</end>
12             </timeperiod>
13             <leaders>
14                 <leader historical-figure="true">
15                     <name pseudonym="true">
16                         Alexander the Great
17                     </name>
18                     <origin>Macedonia</origin>
19                     <battles>
20                         <battle>
21                             ...
22                         </battle>
23                     </battles>
24                 </leader>
25             </leaders>
26         </period>
27     </periods>
28 </history>

```

2.Beispiel) XML Schema - report.xsd

Schreiben Sie für die `report.xml` Datei ein XML Schema.

► Hint: XML Struktur ▼

- Das `<authors>` Element kann eine freie Abfolge von `<author>` Elementen enthalten.
- Ein `<author>` Element kann 2 mögliche Inhalte haben: 1.Variante - `<first-name>`, `<middle-name>`, `<last-name>`
2.Variante - `<name>`, `<is-alive>`
- Das `<middle-name>` ist immer optional.
- Namen - `<name>`, `<first-name>`, `<middle-name>`, `<last-name>` - müssen mindesten 3 Zeichen haben. Ein Name kann nicht länger als 40 Zeichen sein. Schreiben Sie einen eigenen Datentypen für um Namen zu beschreiben.
- `id` Attribute dürfen nur Werte wie `app1`, `app2`, ..., `appN` bzw. `author1`, `author2`, ..., `authorN` beinhalten.

► Aufgabenstellung: report.xml ▼

```

1  <!-- ----- -->
2  <!--      XSLT Output: report.xml      -->
3  <!-- ----- -->
4  <?xml version="1.0" encoding="UTF-8"?>
5  <report>
6      <authors>
7          <author id="author1">
8              <first-name>Arthur</first-name>
9              <middle-name>D.</middle-name>
10             <last-name>Dent</last-name>
11          </author>
12          <author id="author2">
13              <name>Ford Prefect</name>
14              <is-alive>false</is-alive>
15          </author>
16      </authors>
17      <content>
18          This report is co-authored <authorref
19              id="author1"/> and <authorref id="author2"/>
20          and organized in sections.
21          <section>This section is based on the
22              data</section>
23      </content>
24      <appendix id="app1" title="A">
25          some data
26      </appendix>
27      <appendix id="app2" title="B">
28          even more data
29      </appendix>
30  </report>

```

3.Beispiel) XML Schema - browser.xsd

Schreiben Sie für die **browser.xml** Datei ein XML Schema.

► Aufgabenstellung: browser.xml ▼

```

1  <!-- ----- -->
2  <!--           XSLT Output: browser.xml      -->
3  <!-- ----- -->
4  <?xml version="1.0" encoding="UTF-8"?>
5  <browser>
6      <tab id="t1">
7          <sublink url="http://www.examp.le"/>
8          <subpage id="s1">
9              <content>
10                 here we have bla.. bla..
11             </content>
12         </subpage>
13         <sublink url="http://www.examp.le"/>
14     </tab>
15     <tab>
16         <content>
17             here we go again
18         </content>
19         <subpage>
20             <content>alora, alora</content>
21             <subpage>
22                 <content></content>
23             </subpage>
24             <sublink url="http://www.examp.le"/>
25             <sublink url="http://www.examp.le"/>
26         </subpage>
27         <subpage id="4">
28             <content>hallo</content>
29         </subpage>
30     </tab>
31 </browser>

```



► Aufgabenstellung: search.xml ▼

```

1  <!-- ----- -->
2  <!--           XSLT Output: search.xml      -->
3  <!-- ----- -->
4  <?xml version="1.0" encoding="UTF-8"?>
5  <search>
6      <or>
7          <and>
8              <term>x + 2</term>
9              <or>
10                 <term>y - 1</term>
11             </or>
12             <not>
13                 <term>k + 3</term>
14             </not>
15         </and>
16     </or>
17 </search>

```



4.Beispiel) XML Schema - search.xsd

Schreiben Sie für die **search.xml** Datei ein XML Schema.

► Hint: XML Struktur ▼

- Das **<search>** Element beinhaltet immer ein Element. Mögliche Elemente sind: **<or>**, **<and>**, **<term>** und **<not>**.
- Die **<and>** und **<or>** Elemente können 2 der folgenden Elemente beinhalten: **<and>**, **<or>**, **<not>**, **<term>**.
- Das **<not>** Element kann eins von folgenden Elementen beinhalten: **<and>**, **<or>**, **<not>**, **<term>**.