UNIVERSITY of LIMERICK

OLLSCOIL LUIMNIGH

COLLEGE *of* INFORMATICS *and* ELECTRONICS
Department of Computer Science and Information Systems

End-of-Semester Exam

Academic Year:	2007/2008	**Semester:** Autumn
Module Title:	Document Architectures	**Module Code:** CS4146
Exam Duration:	2½ Hours	**Total Marks:** 75 (75% of the final grade)
Lecturer:	Dr. N. S. Nikolov	

Instructions to Candidates:

Please write all answers in the answer booklet.
State clearly any assumptions you make.

Choose to answer either Q2.1 or Q2.2. All other questions should be answered.

QUESTIONS

Q1. Multiple-choice and short-answer questions **(15 marks)**

(N.B. Each multiple-choice question has **exactly one** correct answer.)

Q.1.1. Which of the following is a binary document format? **(1 mark)**

Q1.2. XHTML is an ____________ application. **(1 mark)**

Q1.3. Describe briefly how structure can be separated from presentation in HTML documents. **(3 marks)**

Q1.4. If used in a DTD, B* refers to element B that occurs in an XML document **(1 marks)**

In the following three questions consider the XML file:

```
<?xml version="1.0" encoding="utf-8"?>
<Class>
    <Student>
        <Name>Michelle</Name>
        <Course>Computer Systems</Course>
    </Student>
    <Student>
        <Name>Norah</Name>
        <Course>Software Localisation</Course>
    </Student>
    <Student>
        <Name>Peter</Name>
        <Course>Software Engineering</Course>
    </Student>
</Class>
```

Q1.5. What will be selected by the following XPath expression? **(1 mark)**

`/Class/Student[3]/*[2]`

Q1.6. What will be selected by the following XPath expression? **(1 mark)**

`/Class/Student[2]/*[2]/ancestor::*[2]`

Q1.7. What will be selected by the following XPath expression? **(1 mark)**

`/Class/Student[2]/ancestor::*[1]/child::*[2]`

Hint: The first child element of an element has index 1.

Q1.8. What are the four requirements for an XML document to be well formed? **(3 marks)**
Can an XML document be well formed but not valid? Why?

Q1.9. Compare briefly bitmaps vs. vector images. **(3 marks)**

Q2. Document Type Definition and XML Schema (20 marks)

Choose to answer either Q2.1 or Q2.2.

Consider the following two XML documents.

tvshow01.xml

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

<tvshow year="1999" rating="PG" season="1">
    <title>Futurama</title>
    <genrelist>
       <genre name="Animation"/>
       <genre name="SciFi"/>
       <genre name="Comedy"/>
    </genrelist>
    <creator>
       <firstname>Matt</firstname>
       <surname>Groening</surname>
    </creator>
    <cast>
       <castmember role="Philip J. Fry">
          <firstname>Billy</firstname>
          <surname>West</surname>
       </castmember>
       <castmember role="Turanga Leela">
          <firstname>Katey</firstname>
          <surname>Sagal</surname>
       </castmember>
    </cast>
</tvshow>
```

tvshow02.xml

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

<tvshow year="1999" rating="PG" season="1">
    <title>Futurama</title>
    <genrelist>
       <genre name="Cartoon"/>
    </genrelist>
    <creator>
       <firstname>Matt</firstname>
       <surname>Groening</surname>
    </creator>
    <cast>
       <castmember role="Philip J. Fry">
          <firstname>Billy</firstname>
          <surname>West</surname>
       </castmember>
    </cast>
</tvshow>
```

Q2.1. Write a DTD such that tvshow01.xml conforms to it (i.e. it is valid), and tvshow02.xml does NOT conform to it (i.e. it is invalid).

Hint: Use any of the DTD declarations:

```
<!ELEMENT name (type)>
<!ATTLIST element_name
          attribute_name type default_value
          attribute_name type default_value
          …
>
```

Q2.2. Write an XML Schema such that tvshow01.xml conforms to it (i.e. it is valid), and tvshow02.xml does NOT conform to it (i.e. it is invalid).

Hint: Use the following general structure for the schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <!-- definition of simple types -->  ...
  <!-- definition of complex types -->  ...
  <xs:element name="root_element" type="root_element_type"/>
</xs:schema>
```

Q3. Consider the following XML document. **(20 marks)**

library.xml

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

<library>
    <section id="1" subject="Computer Science">
       <book language="English" copies="30">
           <title>C++ Primer</title>
           <author>
               <firstname>John</firstname>
               <surname>Murphy</surname>
           </author>
           <author>
               <firstname>Dan</firstname>
               <surname>North</surname>
           </author>
       </book>
       <book language="English" copies="40">
            <title>Problem Solving with Java</title>
            <author>
                <firstname>Nell</firstname>
                <surname>Dale</surname>
            </author>
       </book>
    </section>
    <section id="2" subject="Chemistry"/>
</library>
```

Let

```
xmlDoc = new ActiveXObject('Microsoft.XMLDOM');
xmlDoc.load('books.xml');
```

a. **(5 marks)** Draw the XML DOM tree for `library.xml`. For each node, specify its name, type and value.

b. **(5 marks)** What are the values of the variables **`a, b,`** and **`c`** after executing the following fragment of JavaScript code.

```
doc = xmlDoc.documentElement; //root
a = doc.firstChild.childNodes.item(0).firstChild.nodeValue;
b = doc.firstChild.childNodes.item(0).firstChild.firstChild.nodeValue;
c = doc.lastChild.getAttribute("subject");
```

c. **(5 marks)** Write a JavaScript statement that assigns the contents of the last **`surname`** element (i.e., the string "**`Dale`**") to a variable called **`sname`**.

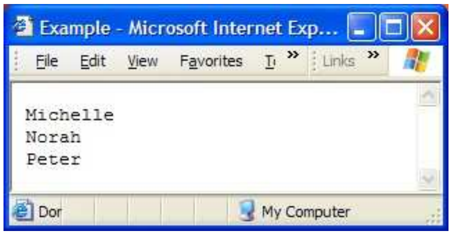
d. **(5 marks)** Write a JavaScript statement that assigns the value of the attribute **`copies`** of the first **`book`** element (i.e., "**`30`**") to a variable called **`ncopies`**.

Q4. Consider the following XML document. **(20 marks)**

class.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<Class>
      <Students>
            <Student id="03453345">
                  <Name>Michelle</Name>
                  <Grade module="CS8182" value="3"/>
                  <Grade module="CS8358" value="3"/>
                  <Grade module="CS8234" value="1"/>
                  <Grade module="CS8742" value="4.5"/>
            </Student>
            <Student id="03329032">
                  <Name>Norah</Name>
                  <Grade module="CS8182" value="8"/>
                  <Grade module="CS8358" value="6.25"/>
                  <Grade module="CS8234" value="5"/>
                  <Grade module="CS8742" value="4"/>
            </Student>
            <Student id="03353453">
                  <Name>Peter</Name>
                  <Grade module="CS8182" value="1"/>
                  <Grade module="CS8358" value="2"/>
                  <Grade module="CS8234" value="2"/>
                  <Grade module="CS8742" value="2"/>
            </Student>
      </Students>
</Class>
```

a. (5 marks) Write an XSLT stylesheet that transforms **`class.xml`** into an HTML document which contains only the names of the students. That is, when loaded in a browser the HTML document should look like:



```
Michelle
Norah
Peter
```

b. (15 marks) Write an XSLT stylesheet that transforms **`class.xml`** into an XML document which contains only the students whose second grade is greater than or equal to 3, and has the following format:

```
<?xml version="1.0" encoding="utf-8"?>
<Class>
    <Student name="Michelle">
        <Grades>3, 2, 1, 4.5</Grades>
    </Student>
    <Student name="Norah">
        <Grades>8, 6.25, 5, 4</Grades>
    </Student>
</Class>
```

Hint: Use the expression <xsl:if test="Grade[2] /@value &gt;= 3"> to check the value of the second grade.

End of Exam