APR 2010

PART A (60 MARKS)

QUESTION 1

XML can be designed and used in various kinds of application. MathML is one of the examples of language based on XML. In order to create an XML, an author needs to be aware of the syntax rules to make an XML document well-formed.

a) List FOUR (4) main syntax rules of XML that meet the well-formedness constraint given by the XML 1.0 specification. (2 marks)

- XML Elements must have a Closing Tag
- XML Tags are Case sensitive
- XML Elements Must be properly Nested
- XML document must have a Root Element
- XML Attribute must be Quoted

b) Describe what is meant by this statement: "XML is a metalanguage". (2 marks)

- Xml can be used to describe and generate other language markup

c) List TWO (2) XML-based languages listed in W3C Recommendation. (1 mark)

-

d) Based on the answer in c), list the domain or field of each the language. (1 mark)

-

QUESTION 2

a) DTD is abbreviation for Document Type Definitions. Define the purpose of DTD. (2 marks)

- Defined the structure of an XML document with a list of legal elements.

b) Write a DTD statement based on the following condition:

i) Declare an element named bag that contains either element **vegetable** or element **fruit**, but not both.

```
<!ELEMENT bag (vegetable| fruit)>
```

ii) Declare an element named bag that contains the element **vegetable**, or the **element fruit**, or both of them in the sequence vegetable, fruit.

```
<!ELEMENT bag (vegetable|fruit | (vegetable, fruit))>
```

iii) Declare an element named bag that contains two elements **of vegetable** or two elements of **fruit** in the sequence. (6 marks)

```
<!ELEMENT bag (vegetable* | Fruit*) >
```

QUESTION 3

Write XML Schema Definition (XSD) codes to define the following elements/attributes:

a) Simple type element named ClassName. The values for ClassName start with a number, followed by six letters of the alphabet in capital letters. (2 marks)

```
<xs:element name="ClassName">
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:patent value="[0-9][A-Z][A-Z][A-Z][A-Z][A-Z][A-Z]"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
```

b) Simple type element named Passwd. Passwd consists of eight characters which contain any combination of numbers, uppercase letters and/or lowercase letters. (2 marks)

```
<xs:element name="Passwod">
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:pattern value="[a-zA-Z0-9] {8}"/>
        </xs:restriction>
    </xs:simpleType>
</xs:element>
```

c) Attribute named state . The values of stat e may only contain Selangor, Perak or Johor. (2 marks)

```
<xs:simpleType name="state">
    <xs:restriction base="xs:string">
        <xs:enumeration value="Selangor"/>
        <xs:enumeration value="Perak"/>
        <xs:enumeration value="Johor"/>
    </xs:restriction>
</xs:simpleType>
```

d) Complex type element named **Order**. **Order** contains mixed content. The child elements of **Order** are **Orderid , Date**, and **Quantity** . **Orderid** contains positive integers. **Date** contains **date**, while **Quantity** contains numbers. All the child elements must appear in the following sequence: **Orderid , Date** and **Quantity .** (3 marks)

```
<xs:element name=Order>
    <xs:complexType>
        <xs:sequence>
            <xs:element name="Orderid" type="xs:string"/>
            <xs:element name="Date" type="xs:date"/>
            <xs:element name="Quantity" type="xs:postitiveInteger"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
```

QUESTION 4

Given the following XML document which consists of data about lecturers of FSKM and their Special Interest Group (SIG):

```
<!-- FSKM.xml -->
<FSKM>
  <CTN_lec>
        <ctn sig='security'>
              <name>Kamariah</name>
        </ctn>
        <ctn sig='network'>
              <name>Jamil</name>
        </ctn>
  </CTN_lec>

  <CS_lec>
        <cs sig="CSR">
              <name>Hamzah</name>
         </cs>
        <cs sig="IP">
              <name>Rosnah</name>
         </cs>
        <cs sig="NM">
              <name>Mariam</name>
         </cs>
  </CS_lec>
</FSKM>
```

a) Write an XPath expression to select all CS lecturers (cs) with name Rosnah.

- /FSKM/CS_lec//cs[name="Rosnah"]

b) Write an XPath expression to select the text of the name for the second cs element.

- /FSKM/CS_lec/cs/name[2]

c) Write an XPath expression to select the SI G (sig) of all CTN_lec.

- /FSKM//CSTN_lec/ctn/sig

d) Write an XPath expression to select the SI G for CS_lec named Mariam.

- /FSKM/CS_lec/cs@sig[name="Mariam"]

e) Write an XPath expression to select the third lecturer of FSKM (FSKM). (10 marks)

-

QUESTION 5

Document Object Model (DOM) represents an XML document as a tree or hierarchy of node objects. The node interface is the primary datatype for the entire DOM.

a) List FOUR (4) basic types of nodes used to represent XML document components. (4 marks)

- element
- attribute
- text
- comment
- document note

b) List FIVE (5) names and their description of the node properties used in W3C. (5 marks)

- nodeName – specifies the name of node
- nodeValue – specifies the value of a node
- nodeType – return the type of node.
- childNodes – returns a nodelist of child nodes for a node
- nextSibling – return the node immediately following a node
- fistChild – return the first child of a node
- lastChild – return the last child of node.

c) Given an XML document that contains data about student's information in their class.

```
<?xml version = '1.0' encoding='UTF-8'?>
<student_info>
<class type = '2Science1'>
   <year>2010</year>
      <attendees>
         <name group='A'>
            <firstName>Zulkifli</firstName>
            <lastName>Abdullah</lastName>
         </name>
         <name group='B'>
            <firstName>Teoh</firstName>
            <lastName>Ben Hock</lastName>
         </name>
         <name group='A'>
            <firstName>Vijay</firstName>
            <lastName>Jones</lastName>
         </name>
      </attendees>
</class>
</student_info>
```

Create a DOMDocument object to load in the XML document the name of the third student. Given below is a segment of JavaScript for you to fill in.

```
<HTML>
<HEAD>
<TITLE> Extracting Student Data</TITLE>
<SCRIPT LANGUAGE = "JavaScript">
   function readData()
   {

      ..create your DOM here..

   }
</SCRIPT>
</HEAD>

<BODY>
   <H1> Extracting Student Data </H1>
   <INPUT TYPE="BUTTON" VALUE="CLICK" ONCLICK="readData()">
</BODY>
</HTML>
```

(6 marks)

```
function loadXMLDoc(dname)
{
        if (window.XMLHttpRequest)
                {
                xhttp=new XMLHttpRequest();
                }
        else
                {
                xhttp=new ActiveXObject("Microsoft.XMLHTTP");
                }
        xhttp.open("GET",dname,false);
        xhttp.send();
        return xhttp.responseXML;
}

xmlDoc=loadXMLDoc("student.xml");
function readData()
{
        first=xmlDoc.getElementsByTagName("firstName")[2].childNodes[0].nodeValue;
        last=xmlDoc.getElementsByTagName("lastName")[2].childNodes[0].nodeValue;
        document.write(first + " " + last + "<br />");
}
```

QUESTION 6

Write the XSL code to produce the HTML output from the XML file.

a)

XML	HTML
<xslTutorial> <title>XSL</title> <author>John Smit</author> </xslTutorial>	<HTML> <HEAD></HEAD> <BODY> <H1>XSL</H1> <H2>John Smith</H2> </BODY> </HTML>

(4 marks)

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
	<xsl:template match="/">
		<html>
			<head></head>
			<body>
				<h1><xsl:value-of select="xslTutorial/title"/></h1>
				<h2><xsl:value-of select="xslTutorial/author"/></h2>
			</body>
		</html>
	</xsl:template>
</xsl:stylesheet>
```

b)

XML	HTML

<xslTutorial>
  <El id="a1" pos="start">
  <B2 id="B1"/>
  </El>
  <Bl id="a2">
  <B2 id="b3"/>
  <CC id="c1">
  </El>
</xslTutorial>
	<DIV style="color:purple">
	  El id=a1
	</DIV>
	<DIV style="color:purple">
	  Bl id=a2
	</DIV>

(4 marks)

c)

XML	HTML
<xslTutorial> <name>Bahar</name> <name>Devi</name> <name>Ahmad</name> <name>Chong</name> </xslTutorial>	<HTML> <HEAD> </HEAD> <BODY> <TABLE> <TR><TH>Ahmad</TH></TR> <TR><TH>Bahar</TH></TR> <TR><TH>Chong</TH></TR> <TR><TH>Devi</TH></TR> </TABLE> </BODY> </HTML>

(4 marks)

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
        <xsl:template match="/">
                <html>
                        <head></head>
                        <body>
                                <table border="2">
                                <xsl:for-each select="xslTutorial/name">
                                        <TR><TH><xsl:value-of select="."/>
                                        <xsl:element name="br"/></TH></TR>
                                </xsl:for-each>
                                </table>
                        </body>
                </html>
        </xsl:template>
</xsl:stylesheet>
```

PART B (40 MARKS)

QUESTION 1

Given the following XML and DTD documents:

```
<?xml version="1.0"?>
<biggest_loser>
   <session date="2009-11-01" type="jogging" heartrate="90"/>
   <session date="2009-11-03" type="swimming" heartrate="78"/>
   <session date="2009-11-05" type="push_up" heartrate="100"/>
   <session date="2009-11-07" type="jogging" heartrate="92"/>
      ...
      ...
</biggest_loser>
```

```
<!DOCTYPE biggest_loser[
   <!ELEMENT biggest_loser(session)*>
   <!ELEMENT session EMPTY>
   <!ATTLIST session
      date CDATA #REQUIRED
      type CDATA #REQUIRED
      heartrate CDATA #REQUIRED>
]>
```

a) Write an XSL Stylesheet to transform the given XML document to the following XML document:

```
<workout>
        <jogging heartrate="90" date="2009-11-01" />
        <swimming heartrate ="78" date="2009-ll-03" />
        <push_up heartrate ="100" date="2009-11-05" />
        <jogging heartrate="92" date="2009-ll-07" />
</workout>
```

Please take note that the values for the attributes are NOT constants. They are captured from the XML document. (10 marks)

b) Write an XSL Stylesheet to transform the given XML document to an HTML file that will display a table on a browser with the following information:

LOGBOOK		
Session Date	Type	Heart Rate

Sort the table by date in descending order. (8 marks)

QUESTION 3

Given an XML document, named person.xml as follow:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="person.xsl"?>

<PersonInfo>
      <Person>
            <name>Luqman Hakim Ridzuan</name>
            <address> 5 Jln Emas Tmn Perak II</address>
            <city>Melaka</city>
            <Icnumber>0D01G1085675</Icnumber>
            <email>luqman@yahoo.com</email>
      </Person>
      <Person>
            <name>Marina Ahmad</name>
            <address> Rt6.5 Kampung Jaya Kulim</address>
            <city>Kedah</city>
      </Person>
</PersonInfo>
```

In order to display the output, write an XSLT Stylesheet by using <xsl:apply-templates> element so that you can choose the specific elements to be formatted. Below is the guideline of data format:

- Element name - font: arial bold, text alignment: right
- Element address - font: arial 12, text alignment: right
- Element city - font: arial 12 bold, text alignment: right
- Element icnumber - font: vendana 10, text alignment: justify
- Element email - font: vendana 10 italic, text-alignment: justify

(10 marks)

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
      <xsl:template match="/">
            <html>
                  <head></head>
                  <body>
                        <h2>Person Info</h2>
                        <xsl:apply-templates/>
                  </body>
            </html>
      </xsl:template>
      <xsl:template match="PersonInfo/Person">
            <p>
                  <xsl:apply-templates select="name"/>
                  <xsl:apply-templates select="address"/>
                  <xsl:apply-templates select="city"/>
                  <xsl:apply-templates select="Icnumber"/>
                  <xsl:apply-templates select="email"/>
            </p>
      </xsl:template>
```

```
	<xsl:template match="name">
		Name: <span style="font-family:Arial; font-weight:bold; text-align:right;">
		<xsl:value-of select="."/></span>
		<br />
	</xsl:template>

	<xsl:template match="address">
		Address: <span style="font-family:Arial; font-size:12px; text-align:right;">
		<xsl:value-of select="."/></span>
		<br />
	</xsl:template>

	<xsl:template match="city">
		City: <span style="font-family:Arial; font-size:12px; font-weight:bold;  text-align:right;">
		<xsl:value-of select="."/></span>
		<br />
	</xsl:template>

	<xsl:template match="Icnumber">
		IC number: <span style="font-family:verdana; font-size:10px; font-weight:bold; text-
align:right;">
		<xsl:value-of select="."/></span>
		<br />
	</xsl:template>

	<xsl:template match="email">
		Email: <span style="font-family:Arial; font-size:12px; font-style:italic; text-align:right;">
		<xsl:value-of select="."/></span>
		<br />
	</xsl:template>
</xsl:stylesheet>
```