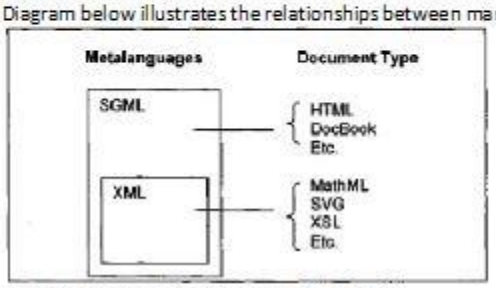
OCT 2010

PART A (60 MARKS)

QUESTION 1

Diagram below illustrates the relationships between markup languages.



a) Define the term "metalanguage". (2 marks)

-

b) Describe the concept of "markup". (2 marks)

-

c) Describe the relationship between HTML and XML. (2 marks)

-

d) State TWO differences between HTML and XML. (4 marks)

-

e) State a different between valid and well-formed. Write ONE (1) example for each of them. (3 marks)

-

QUESTION 2

Content models are used to indicate the structure and order in which child elements can appear within their parent element.

a) Describe the meaning of three types of model group listed below:

i) xsd : sequence

- meaning that the elements must show up in the order they are declared

ii) xsd:all

- meaning that the elements can show up in any order

iii) xsd : choice

- meaning that only one of the child elements may show up

(6 marks)

b) For each type of the content models above, write a valid XML code with XML Schema. (9 marks)

i)

```
<xs:element name="elementName">
	<xs:complexType>
		<xs:sequence>
			<xs:element name="child1" type="xs:string"/>
			<xs:element name="child2" type="xs:string /> 
			<xs:element name="child3" type="xs:string" />
		</xs:sequence>
	</xs:complexType>
</xs:element>
```

ii)

```
<xs:element name="elementName">
	<xs:complexType>
		<xs:all>
			<xs:element name="child1" type="xs:string"/>
			<xs:element name="child2" type="xs:string /> 
			<xs:element name="child3" type="xs:string" />
		</xs:all>
	</xs:complexType>
</xs:element>
```

iii)

```
<xs:element name="elementName">
	<xs:complexType>
		<xs:choice>
			<xs:element name="child1" type="xs:string"/>
			<xs:element name="child2" type="xs:string /> 
			<xs:element name="child3" type="xs:string" />
		</xs:choice>
	</xs:complexType>
</xs:element>
```

QUESTION 3

The definition for document type, my-doc.dtd contains the following markup declarations.
By considering the DTD below:

```
<!ENTITY % LanguageCode "NMTOKEN">
<!ENTITY % my-attr "xml:lang %LanguageCode; #IMPLIED">
<!ENTITY % my-fontstyle "big | small">
<!ENTITY % my-phrase "em | cite">
<!ENTITY % my-inline "%my-fontstyle; | %my-phrase;">
<!ENTITY % my-Style "(#PCDATA  |%my-inline;)*">
<!ELEMENT my-Para %my-Style;>
<!ATTLIST  my-Para %my-attr; color (red | black) "black">
<!ELEMENT em %my-Style;>       <!-- strong emphasis -->
<!ELEMENT cite %my-Style;>     <!-- citation -->
<!ELEMENT big %my-Style;>      <!-- bigger font -->
<!ELEMENT small %my-Style;>    <!-- smaller font -->
```

a) Replace and write the text of entity my-style with the equivalent text in DTD. (3 marks)

- (#PCDATA | big | small | em | cite)*

b) Write a well-formed and valid example for element my-Para. In addition, the element my-Para is required to have TWO children elements and TWO attributes. (5 marks)

- my-Para (#PCDATA | big | small | em | cite) is of mixed type

<my-para xml :lang="en" color="black"> Dari Anas RA
 Nabi SAW bersabda>
<cite> tidak sempurna keimanan seseorang dari kalian, sebelum ia mencintai saudaranya (sesame muslim) sebagaimana I mencintai dirinya sendiri.</cite>
</mypara>

QUESTION 4

The Document Object Model (DOM) is a language-independent alternative. It views XML documents as a tree-structure and all the elements, text and attributes are known as nodes. Given below are an XML document and a segment of JavaScript code used to load the XML file.

XML Document

```
<?xml version="1.0" ?>
<school>
   <teacher id="001" sex="F" age="19">Suriati Mokhtar</teacher>
   <teacher id="002" sex="F" age="24">Thanjit Kaur</teacher>
   <teacher id="003" sex="M" age="21">Khor Siok Luan</teacher>
   <subject>
      <enrollment code_id="E2000">50</enrollment>
      <enrollment code_id="C2001">40</enrollment>
      <enrollment code_id="B2002">20</enrollment>
   </subject>
</school>
```

A segment of Java Script code

```
    :
var xmlDoc = new ActiveXObject("Microsoft.XMLDOM");
function loadXML(xmlFile)
{
  xmlDoc.async="false";
  xmlDoc.onreadystatechange=verify;
  xmlDoc.load(xmlFile);
  xmlObj=xmlDoc.documentElement;
}
    :
```

State the output return from the DOM reference code below.

a) xmlObj .childNodes(3).hasChildNodes()
   - return TRUE

b) xmlObj .childNodes(2).hasChildNodes()
   - Return FALSE

c) xmlObj.tagName
   - Return school

d) xmlObj .childNodes(0).tagName
   - Return Teacher

e) xmlObj .childNodes(3) .childNodes(0).tagName
   - Return Enrollment

f) xmlObj childNodes(0) .firstChild.text
   - Return Suriati Mokhtar

g) xmlObj .childNodes(3) .childNodes(1).firstChild.text
   - Return 40

h) xmlObj .childNodes(1).getAttribute("age")
   - Return 24

QUESTION 5

Given below is an XML document.

```
<?xml version='1.0'?>

<color>
   <x> green </x>
   <y>
      <x> yellow </x>
      <x> blue </x>
   </y>
   <z>
      <x> black </x>
      <x> red </x>
   </z>
   <x> purple </x>

   <x>
      <y> orange </y>
      <y> grey </y>
   </x>
   <z>
      <y> cyan </y>
      <y> brown </y>
   </z>
</color>
```

Write the output extracted from the following XPath expression:

a) x | y/ x

- Green yellow blue purple orange gray cyan brown

b) // x

- Green yellow blue black red purple orange grey cyan brown

c) // y

- Yellow blue orange grey cyan brown

d) y[1] / x[2]

- blue

e). x[3] / y[2]

- grey

f) (x/y) [1]

- orange

g) x / y[1]

- orange cyan

h) x / y [position n () = 1]

- orange cyan

QUESTION 6

Given below are the DTD and HTML document. By considering the DTD below, write an XSLT coding to transform any XML document that has the structure described by the DTD to produce the following HTML document. The data in the HTML document is sorted by name/first.

```
<!ELEMENT cards (card)+>
<!ELEMENT card (name, title, email, phone)>
<!ELEMENT name (first, last)>
<!ELEMENT first (#PCDATA)>
<!ELEMENT last (#PCDATA)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT email (#PCDATA)>
<!ELEMENT phone (#PCDATA)>
<!ATTLIST card type CDATA #IMPLIED>
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
<title>Business card</title>
 <body>
   <h1>
      <span style="color:#ff0000">
        <i> text data of element first </i>
      </span>

      <b> text data of element last </b>
   </h1>
   <h3><i>text data of element title</i></h3>
   <p>email: <a href="mailto: text data of element email ">
      <tt>text data of element email </tt></a>
   </p>
   <p>phone: text data of element phone</p>
 </body>
....
....
....
</html>
```

(8 marks)

```
xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/xsl/transfom" version="1.0"
xmlns="http://www.w3.org/1999/xhtml">
        <html xml ns="http://www.w3.org/1999/xhtml">
                <title>Business card</title>
                <body>
                        <xsl:apply-templates>
                                <xsl:sort select="name/first" order="ascending"/>
                        </xsl:apply-templates>
                </body>
        </html>

        <xsl:template match="name">
        <hl>
                <span style="color:#ff0000">
                        <i><xsl:value-of select="first"/></i>
                </span>
                <b><xsl:value-of select="last"/></b>
        </hl>
        </xsl:template>

        <xsl:template match="title">
        <h3xi><xsl:value-of select="text()"/></ix/h3>
        <p>email: <a href="mailto:{text()}">
                <tt><xsl:value-of select="text()"/></ttx/a>
        </p>
        </xsl:template>

        <xsl:template match="phone">
                <p><xsl:value-of select="text()"/></p>
        </xsl:template>
</xsl:stylesheet>
```

PART B (40 MARKS)

QUESTION 1

Given below is a list of integer number written in XML document. Write an XSLT Stylesheet code to transform the data in XML into a sorted list of data in ascending order with alternating font color red and blue.

```
<nombor>
   <integer> 15 </integer>
   <integer> 12 </integer>
   <integer> 17 </integer>
   <integer> 25 </integer>
   <integer> 10 </integer>
   <integer> 17 </integer>
   <integer> 23 </integer>
<nombor>
```

(10 marks)

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/xsl/transform" version="1.0"
xmlns="http://www.w3.org/1999/xhtml">
        <xsl:template match="nombor">
                <xsl:copy>
                        <xsl:apply-templates>
                                <xsl:sort select="." data-type="number"/>
                        </xsl:apply-templates>
                </xsl:copy>
        </xsl:template>

        <xsl:template match="integer">
                <xsl:copy of select="."/>
        </xsl:template>
</xsl:stylesheet>
```

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/xsl/transform" version="1.0"
xmlns="http://www.w3.org/1999/xhtml">
        <xsl:template match="nombor">
                <xsl:copy>
                        <xsl:apply-templates/>
                </xsl:copy>
        </xsl:template>

        <xsl:template match="integer[position() mod 2 =0]">
                <li><font color="blue"> <xsl:copy of select="text()"/></font></li>
        </xsl:template>

        <xsl:template match="integer[position() mod 2 =1]">
                <li><font color="red"><xsl:copy of select="text()"/></font></li>
        </xsl:template>
</xsl:stylesheet>
```

QUESTION 2

XML document below contains data related to car manufacturing.

```
<CARS>
   <CATEGORY> Accessories </CATEGORY>
   <COMPONENT>
      <ITEM_NAME> Break Pad</ITEM_NAME>
      <MODEL> BP123</MODEL>
      <MANUFACTURER>Cheer AutoCar </MANUFACTURER>
      <PRICE>2000</PRICE>
   </COMPONENT>
   <COMPONENT>
      <ITEM_NAME> Gear Box </ITEM_NAME>
      <MODEL> GB190</MODEL>
      <MANUFACTURER>Cleanz CarKit </MANUFACTURER>
      <PRICE>3200</PRICE>
   </COMPONENT>
</CARS>
```

Write an internal DTD to validate the XML document above by adding the requirement listed below.

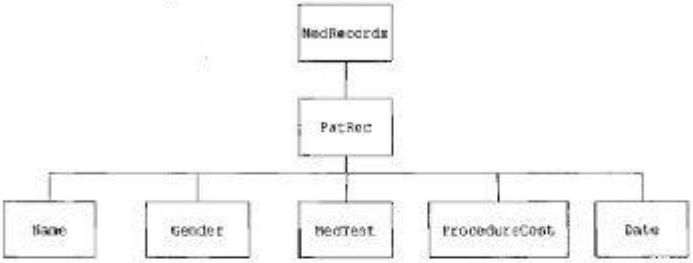
- an optional text string attribute "id " for the element CATE GORY.
- an attribute named "discount " for element PRIC E which can have one of four possible values: 10, 20, 30 and 40.
- a "code" attribute is required for element MODEL with the value must contain no white space.
- an element ITEM_NAM E has an optional attribute named "product", which refer to the "id " item of the name of CATEGORY.
- The element COMPONENT has an optional attribute named "instock " that can have the value "yes" or "no". The default value is "yes".

(15 marks)

```
<!DOCTYPE CARS[

<!ELEMENT CARS (CATEGORY, COMPONENT+)>

<!ELEMENT CATEGORY (#PCDATA)>
<!ELEMENT COMPNENT (ITEM_NAME, MODEL, MANUFACTURE, PRICE)>
<!ELEMENT ITEM_NAME(#PCDATA)>
<!ELEMENT MODEL(#PCDATA)>
<!ELEMENT MANUFACTURE(#PCDATA)>
<!ELEMENT PRICE(#PCDATA)>

<!ATTLIST CATEGORY id ID #IMPLIED>
<!ATTLIST PRICE discount ("10"|"20"|"30"|"40") #REQUIRED>
<!ATTLIST MODEL code NMTOKEN #REQUIRED>
<!ATTLIST ITEM_NAME product IDREF #IMPLIED>
<!ATTLIST COMPONENT inStock ("yes"|"no") "yes">

]>
```

QUESTION 3

Given the hierarchy structure of an XML data as below:



MedRecords	Root element for reports about patient's data and their medical tests.
PatRec	The record delimiter tag for each patient's set of data.
Name	The patient's name.
Gender	Gender of the patient either male or female
MedTest	The type of medical test performed on the patient.
ProcedureCost	The cost of the medical procedure
Date	The date when the medical test was performed.

a) Write an XML Schema Definition (XSD) to define the structure above based on the following restrictions:

- PatRec element can occur one or more
- Gender only acceptable value is male OR female
- ProcedureCost has a value of at least RM15.00 but cannot exceed the value of RM2,500.00
- MedTest element only acceptable value is xRay OR blood OR urine" (10 marks)

b) Write any possible XML document based on the XML Schema Definition created above. (5 marks)

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLschema">
	<xs:element name="MedRecords">
		<xs:complexType>
			<xs:sequence>
				<xs:element ref="PatRec" maxOccurs="unbounded"/>
			</xs:sequence>
		</xs:complexType>
	</xs:element>

	<xs:element name="PatRec">
		<xs:complexType>
			<xs:sequence>
				<xs:element name="Name" type="xs:string"/>
				<xs:element name="Gemder"/>
					<xs:simpleType>
						<xs:restriction base="xs:string">
							<xs:pattern value="male|female"/>
						</xs:restriction>
					</xs:simpleType>
				</xs:element>
				<xs:element name="MedTest">
					<xs:simpleType>
						<xs:restriction base="xs:string">
							<xs:pattern value="xRay|bload|uraine"/>
						</xs:restriction>
					</xs:simpleType>
				</xs:element>
				<xs:element name="ProcedureCost" type="MedProcedureCost"/>
				<xs:element name="Date" type="xs:date"/>
			</xs:sequence>
		</xs:complextype>
	</xs:element>
	<xs:simpleType name="MedProceduteCost">
		<xs:restriction base="xs:decimal"/>
			<xs:minInclusive value="15.00"/>
			<xs:maxInclusive value="2500.00"/>
		</xs:restriction>
	</xs:simpleType>
</xs:schema>
```