**UNIVERSITY OF PRETORIA**

**Department of Information Science**

**IMY 210**
**Advanced Markup Languages**

**Examination**

| | | |
|---|---|---|
| ***Examiner:*** | | 13 June 2022 |
| *Internal:* | Mr YL Wong | *Marks:* 100 |
| *External:* | Mr ID Bosman | *Time:* 9 hours |

---

***General Instructions:***

***Use your own discretion when modifying the files and directory structure, remember to upload the correct files.***

***Upload the files in compressed "zip" format.***

***Only upload the file(s) indicated.***

***There will be two upload spots available on ClickUP, ensure that all your files are zipped correctly and are uploaded to both slots.***

***You will be provided multiple upload slots but only your final upload will be marked.***

***When uploading to the exam slot, make sure you included only the indicated files in the correct folder structure.***

***Remember to add your full name and student number to every file you upload as a comment. If your comment violates the integrity of your documents, it will be marked as not well-formed.***

***The upload slots will be available from 08:00 13 July until 17:00 13 July.***
***No late submissions will be accepted or marked.***

***General Warnings:***

***If your validation files do not validate the provided files, you will not receive more than 50% for that section of the exam.***

***If your files are not well-formed, you will not receive more than 50% for that section of the exam.***

***Upload checklist:***

- ***I have named the document according to the naming convention provided by this specification.***
- ***I have added a comment to the document that does not affect the integrity of the document.***
- ***I have ensured that the attached files are all well-formed.***
- ***I have ensured that the attached files work with the provided XML documents.***
- ***I have ensured that the zip file about to be uploaded consists only of the files required for the question.***
- ***If uploading the final section, I have ensured that the folder structure is correct.***

***Declaration:***

**By taking this exam you are consenting to the following integrity and plagiarism declaration:**

***"The University of Pretoria commits itself to producing academic work of integrity. I affirm that I am aware of and have read the Rules and Policies of the University, more specifically the Disciplinary Procedure and the Tests and Examinations Rules, which prohibit any unethical, dishonest, or improper conduct during tests, assignments, examinations and/or any other forms of assessment.***

***I am aware that no student or any other person may assist or attempt to assist another student, or obtain help, or attempt to obtain help from another student or any other person during tests, assessments, assignments, examinations and/or any other forms of assessment."***

---

## Section A: XML [5%]

**Provided files:**

- xml.txt

**Task:**

Convert the provided text file into a working XML file by renaming and resolving the errors found in the code.

Write the cause of the errors you identified and the associated line number as a comment at **the end of the file** (within the root element).

Indicate whether the error is grammatical or semantic.

E.g. `<!-- Line 3, Semantic, explaining the error -->`

**Warning:**

- Remember that your resulting XML file needs to be well-formed, otherwise, your submission will only be marked until 50%.
- Marks will only be rewarded for uniquely identified errors.
- You will be penalised if your final file's format is not the same as the file required.

**File to submit:**

- xml.xml

**[End of section A]**

## Section B: DTD [15%]

**Provided files:**

- dtd-xml_1.xml
- dtd-xml_2.xml
- dtd-xml_3.xml
- dtd-jpg.jpeg

**Task:**

Create a DTD schema (**dtd-xml.dtd**) that can be used to validate all three provided XML files.

**You will be evaluated on the following criteria**:

- Applications of every form of qualifiers.

- Sequential and choice-based ordering of elements.
- Applications of nested elements.
- Applications of empty elements.
- Use of optional and required attributes.
- Use of an ID valued attribute and other attribute types.
- An example of an entity.
- Use of notation definition.
- Correct logical representation of element rules.

**Warning**:

- The inability to validate **any of the three** XML files will limit the final mark of this section to 50%.
- The **ANY** keyword is prohibited from being used; any instance of the keyword **ANY** will not be marked.
- All rules must be declared within the external DTD (dtd-xml.dtd).
- Appropriateness of application will serve as a marking criterion for certain rules.
    - E.g., providing a default value for an attribute that does not require a default value will not yield all the marks for those criteria.
- You will be penalised if your final file's formats is not the same as the file required.

**File to submit:**

- dtd-xml.dtd

**[End of section B]**

## Section C: XML Schema and Namespaces [25%]

**Provided files:**

- xsd-xml_1.xml
- xsd-xml_2.xml
- xsd-xml_3.xml
- xsd-xml_4.xml

**Tasks:**

Create three XML Schema files (**validation.xsd, ns_sw.xsd, ns_fw.xsd, ns_f.xsd**) that can be used to validate all four provided XML files.

The provided XML files should give you enough details on creating your schema files.

**You will be evaluated on the following criteria**:

- Applications of named custom simple and complex types.
- Ability to use include and import functions.
- Use of occurrences.
- At least two different types of model groups.
- Applications of facets for restrictions and extensions of data types.
- Use of list and union elements.
- Error detection.
- Demonstration of the applications of namespaces.
- Data-agnostic capability of your schema file.

**Warning:**

- The inability to validate **any of the four** provided XML files will limit the final mark of this section to 50%.
- If you are unable to demonstrate appropriate usage of certain criteria, you will not receive all the marks associated with said criteria.

    - In instances where you use a custom name type for an element that does need to be defined in such a manner, you will not receive all the marks associated with custom name types.
- You will be penalised if your final files' formats are not the same as the files required.

**Files to submit:**

- Four XSD files named:
    - **validation.xsd**
    - **ns_sw.xsd**
    - **ns_fw.xsd**
    - **ns_f.xsd**

**[End of section C]**

## Section D: XSL [25%]

**Provided files:**

- xsl-xml.xml
- demo.pdf
- images folder (images downloaded separately)

**Suggestion:** Shorten the base XML file when you are working on your XSLT. Once you have finished the section, render the result with the original XML.

**Tasks:**

Create a single XSLT (**xsl.xsl**) file that will be used to transform the provided XML into an XSL-FO format and will be further used to generate a PDF file.

The final format of the PDF should look similar to the demo.pdf provided.

**You will be evaluated on the following criteria**:

- Applications of templates.
- Use of literal elements.
- Use of logic conditions.
- Sorting entries alphanumerically.
- Page formatting.
- Styling and formatting of content.
- Appropriate use of functions.
- Similarity to the given results.
- Data-agnostic capability of your XSLT file.

**Notes:**

- You can use the table elements to help you organise your content; the format is similar to that of HTML
    - <fo:table> equivalent to HTML <table>
    - <fo:table-body> used to contains <fo:table-row> and <fo:table-cell>
    - <fo:table-row> equivalent to HTML <tr>
    - <fo:table-cell> equivalent to HTML <td>

- Here are a few attributes to be used for basic styling; the format is the same as that of CSS.
    - border-style
    - border-width
    - border-color
    - font-weight

- font-size
- color
- margin
- padding

- You will not be penalised if the colour, margin or header spacing you used is not exactly the same as demo.pdf. However, the content needs to all appear on one page.
  (Page dimension: width-210mm, height-148mm)

**Warning:**

- The inability to generate a pdf from the provided XML will limit the final mark of this section to 50%.
- When referencing an image, keep in mind all images will be located in the folder structure provided.
- You will be penalised if your final file's format is not the same as the file required.

**File to submit:**

- **xsl.xsl**

**[End of section D]**

## Section E: XQuery [15%]

**Provided files:**

- query-xml-one.xml
- query-xml-two.xml
- query-result.html

**Tasks:**

Create a single XQuery file that will provide the exact result as query-result.html when executed along with the provided XML file.

**You will be evaluated on the following criteria**:

- Using an HTML tag within the result.
- Extracting data from the XML document.
- Using different documents in the same query.
- Applying sorts and groups.
- Generation of the output.
- Showing an example of longhand expression.

**Warning:**

- If the submitted XQuery file does not execute/compile, this section of the paper will only be marked until 50%.
- You will be penalised if your final file's format is not the same as the file required.

**File to submit:**

- **query.xq**

**[End of section E]**

## Section F: JSON [5%]

**Provided files:**

- json-xml.xml

**Tasks:**

Transform the data in the provided XML file to an appropriate JSON counterpart.

**You will be evaluated on the following criteria**:

- Demonstration of JSON's object-orientated format.
- Demonstration of JSON's key-value format.
- Demonstration of the application of four different data types.

**Warning:**

- If your final JSON file is not well-formed, the final mark of this section will be limited to 50%.
- If you are unable to demonstrate appropriate usage of a data type, you will not receive all the marks associated with the said data type.
- You will be penalised if your final file's format is not the same as the file required.

**File to submit:**

- **json-xml.json**

**[End of section F]**

## Section G: Vue [10%]

**Provided files:**

- index.html

**Tasks:**

Complete the Vue declaration for the provided HTML file (index.html) that would result in a simple system that allows users to append tasks to an existing list.

The list will not only store a task but also the current time of when the task has been added.

**You will be evaluated on the following criteria**:

- Transversing and adding of data into a data array
- The use of VUE keyword(s)
- Declaration and usage of JS functions

**Warning:**

- Keep in mind that your final solution will be marked based on functionality
- You will be penalised if your final file's format is not the same as the file required

**File to submit:**

- **index.html**

**[End of section G]**