HE2B ESI

Prénom : .....
Nom : .....
Numéro : .....
Groupe : .....
Maître-assistant : .....

Haute École Bruxelles-Brabant
École Supérieure d'Informatique
Bachelor en Informatique

NVS
2019 – 2020
Mai 2020

WEBR4 – Bloc 2 (D21)

XML : Travail

Question :	XML	DTD	XSD	XPath	XSL	**Total**
Points :	2	12	22	10	14	60
Résultat :						

L'évaluation de la partie XML de l'unité d'enseignement WEBR4 consiste en la réalisation d'un travail. Il s'agit de produire des fichiers informatique, puis de participer à une défense orale de ces productions.

Il n'y a pas deux parties indépendantes : la production des fichiers informatiques et la défense orale. Ces deux parties forment une seule entité : l'évaluation de la partie XML de l'unité d'enseignement WEBR4 en fin de 2e quadrimestre de l'année académique 2019 – 2020.

Seuls les étudiants qui ont produit les fichiers informatiques demandés dans la suite de cet énoncé sont invités à présenter la partie orale.

La partie orale sert à expliquer voire développer les réponses données dans les fichiers informatiques [1].

On peut lire dans la suite de l'énoncé, qu'un certain nombre de points est associé à chaque question du travail. Ces points ne sont pas octroyés suite à la seule remise des fichiers informatiques. Ils sont déterminés *après la défense orale*. La défense orale constitue donc le pivot de l'évaluation.

Pour le dire sans détour, un étudiant incapable d'expliquer une réponse présente dans un des fichiers informatiques qu'il a produit comme solution au questionnaire ne reçoit pas les points associés à cette question, que la réponse dans le fichier informatique soit correcte ou non.

Le but à atteindre pour les étudiants ne s'arrête pas à la mise à disposition de l'enseignant des fichiers informatiques répondant aux questions posées, mais *consiste à surtout être capable d'expliquer, justifier, critiquer et éventuellement modifier, l'ensemble des réponses données dans les fichiers informatiques produits.*

Questions / Thématiques Il y a 5 questions principales, chacune associée à une thématique relative à XML :

1. XML : il s'agit de vérifier et obtenir un fichier XML bien formé ;
2. DTD : il s'agit de produire une DTD au regard de laquelle le fichier XML précédemment produit est valide ;
3. XSD : il s'agit de produire un schéma sur base de contraintes données dans l'énoncé et de corriger le fichier XML ;
4. XPath : il s'agit de produire des expressions XPath ;
5. XSL : il s'agit de produire une feuille de style pour la production d'un fichier HTML à partir du fichier XML.

1. C'est pourquoi la participation à la défense orale est limitée aux étudiants qui ont produit des réponses dans des fichiers informatiques.

Q1 XML (2 points)

Le répertoire `01_xml` contient le fichier `addressbook_ko.xml`. Ce fichier n'est pas bien formé. Dupliquez le fichier `addressbook_ko.xml` dans un nouveau fichier dont le nom est `addressbook.xml`. Dans le fichier `addressbook.xml` nouvellement créé, corrigez les problèmes du fichier `addressbook_ko.xml`. Voici comment procéder. Pour chacun des deux éléments posant problème, il vous est demandé de :

— le mettre en commentaire ;
— ajouter juste après cet élément mis en commentaire un nouvel élément. Ce nouvel élément est obtenu à partir de l'élément problématique (mis en commentaire) en effectuant le moins de modifications possibles sur celui-ci. Il n'y a, à chaque fois, qu'un seul caractère à modifier dans l'élément mal formé.

> Pour répondre à la **Q1**, vous devez produire le fichier XML `addressbook.xml` dans le répertoire `01_xml`.

Q2 DTD (12 points)

Le répertoire `02_dtd` contient une DTD produite automatiquement à partir du fichier `addressbook.xml` obtenu en fin de résolution de la **Q1**. Cette DTD est nommée `addressbook_nb.dtd`. Elle n'est pas optimale. Il vous est demandé de rendre cette DTD la plus restrictive possible, au regard du fichier XML `addressbook.xml` de fin de **Q1**. Pour vous aider et pour trancher en cas de doute, l'annexe B (p. 7) contient une série de contraintes s'appliquant aux données et au contenu du fichier XML `addressbook.xml`.

Il est impératif de lire l'annexe B avant de répondre à la **Q2** !

Ne travaillez pas directement dans `addressbook_nb.dtd`. Faites plutôt comme à la **Q1**. Dupliquez le fichier `addressbook_nb.dtd` dans un nouveau fichier dont le nom est `addressbook.dtd`. Dans ce nouveau fichier, mettez en commentaire chaque déclaration DTD qui peut être remplacée par une déclaration davantage restrictive. Ajouter cette nouvelle déclaration restrictive juste après celle mise en commentaire et qu'elle remplace. Lors de la production du fichier `addressbook.dtd`, vous *devez* prendre en compte toutes les contraintes listées dans l'annexe B, pour autant que cela soit possible dans une DTD.

Dans la but de tester votre fichier `addressbook.dtd`, vous pouvez copier dans le répertoire `02_dtd` le fichier `addressbook.xml` obtenu après avoir répondu à la **Q1**. Si vous le voulez, mais ce n'est toujours pas obligatoire, vous pouvez également lier le fichier XML `addressbook.xml` à la DTD `addressbook.dtd`. Le fichier `addressbook.xml` doit bien évidemment être valide pour la DTD `addressbook.dtd` restrictive.

> Pour répondre à la **Q2**, vous devez produire le fichier `addressbook.dtd` dans le répertoire `02_dtd`.

Q3 XSD (22 points)

Copiez dans le répertoire `03_xsd` le fichier `addressbook.xml` obtenu après avoir

répondu à la **Q1**. Le fichier XML utilisé pour la **Q3** est donc, comme pour la **Q2**, le fichier XML *bien formé* (`addressbook.xml`) obtenu après avoir corrigé celui fourni (`addressbook_ko.xml`).

(a) (17 points) Le répertoire `03_xsd` contient le fichier de schéma de nom `addressbook_todo.xsd`. Dupliquez ce fichier dans un nouveau fichier de nom `addressbook.xsd`. Dans ce fichier nouvellement créé, tout comme dans le fichier original fourni `addressbook_todo.xsd`, se trouvent plusieurs (15) commentaires de la forme : `<!-- TODO: compléter ici -->`.

Il vous est demandé de compléter le fichier `addressbook.xsd` à la suite de chacun des commentaires `<!-- TODO: compléter ici -->` de sorte que le fichier XML `addressbook.xml` soit valide au regard du fichier schéma `addressbook.xsd` ainsi corrigé.

Il est strictement interdit de modifier le fichier `addressbook.xsd` ailleurs qu'aux endroits marqués `<!-- TODO: compléter ici -->` !

Comme pour la **Q2**, vous devez, pour cette **Q3**, impérativement prendre en compte les contraintes énumérées dans l'annexe B (p. 7), à deux exceptions près.

Ces exceptions sont les suivantes. Vous ne devez, provisoirement, pas vous occuper du fait que :

— l'attribut `id` des éléments `<person>` est un identifiant unique au sein de l'élément `<persons>` ;

— les valeurs attendues pour les éléments `<person_id>` sont celles de l'attribut `id` des éléments `<person>`.

Vous pouvez lier le fichier XML `addressbook.xml` au fichier de schéma `addressbook.xsd`, mais ce n'est pas obligatoire.

Lorsque vous avez terminé de répondre à cette partie de la **Q3**, le fichier `addressbook.xml` doit être valide au regard du schéma `addressbook.xsd`.

(b) (5 points) Occupez-vous maintenant des contraintes de clé et de référence de clé laissées en suspens au point précédent.

Complétez le fichier `addressbook.xsd` au(x) bon(s) endroit(s) de sorte que :

— l'attribut `id` des éléments `<person>` soit un identifiant unique au sein de l'élément `<persons>` ;

— les valeurs attendues pour les éléments `<person_id>` soient celles de l'attribut `id` des éléments `<person>`.

Si ces contraintes sont bien ajoutées au schéma, vous devez alors réaliser que le fichier `addressbook.xml` n'est plus valide au regard du schéma `addressbook.xsd` complet !

Modifiez maintenant le fichier `addressbook.xml`, en y mettant en commentaire le plus petit nombre d'éléments nécessaires pour qu'il reste bien formé et qu'il soit valide pour le schéma `addressbook.xsd` complet.

Pour répondre à la **Q3**, vous devez produire le fichier `addressbook.xsd` et fournir une version corrigée du fichier `addressbook.xml`. Ces deux fichiers doivent se trouver dans le répertoire `03_xsd`.

Q4 XPath (10 points)

Le dossier `04_xpath` contient le fichier texte `xpath.txt`. Celui-ci contient une série de questions. Il vous y est demandé de produire des expressions XPath.

Répondez à ces questions directement dans le fichier `xpath.txt`, aux emplacements indiqués.

Si vous le désirez, dans le but de tester vos requêtes avec `xmllint` ou un autre outil, vous pouvez copiez dans le répertoire `04_xpath` le fichier `addressbook.xml` obtenu après avoir répondu à la **Q3**. Il est alors fortement conseillé de modifier ce fichier XML de sorte à ne plus placer ses éléments dans l'espace de nommage `https://esi-bru.be/WEBR4`.

Pour répondre à la **Q4**, vous devez compléter le fichier `xpath.txt` dans le répertoire `04_xpath`.

Q5 XSL (14 points)

Le répertoire `05_xsl` contient la feuille de style `addressbook_incomplete.xsl` et le fichier `addressbook_html.pdf`. Ce dernier est le résultat de l'impression d'une page HTML dans un fichier PDF.

Pour commencer, dupliquez le fichier `addressbook_incomplete.xsl` dans une nouvelle feuille de style de nom `addressbook.xsl`. Ensuite, complétez, modifiez, supprimez comme et où vous le voulez le contenu de `addressbook.xsl` de sorte que lorsqu'on applique cette feuille de style au fichier `addressbook.xml` obtenu en fin de résolution de la **Q3** [2], on obtienne une page HTML de rendu identique au fichier PDF `addressbook_html.pdf`.

Il est évident que le contenu de la page HTML est « calculé » à partir de celui du fichier XML de départ. Si les données du fichier XML changent, mais pas sa structure, il n'est pas nécessaire de modifier la feuille de style et les données dans la page HTML produite changent également, mais pas sa structure.

Quelques clarifications :

— les personnes dont les relations sont données en tableaux sont affichées dans l'ordre du document XML. Par exemple, on a « Raymond Loiselle » en premier car c'est la première `<person>` du document XML ;

— dans les titres `<h2>` du document HTML, les informations suivantes sont données, dans cet ordre :

1. le premier prénom de la personne. Par exemple : « Raymond » et non « Claude » ;

2. C'est-à-dire le fichier XML bien formé lors de la résolution de la **Q1** duquel on a retiré, en le mettant en commentaire, un élément à référence de clé erronée détecté lors de la résolution de la **Q3**.

2. le nom de famille de la personne. Par exemple : « Loiselle » ;
3. une † entre parenthèses si la personne est décédée. Par exemple, cette marque n'apparaît pas pour Raymond Loiselle mais bien pour Juliane Diederich, décédée le 2 mai 2005 ;
4. le sexe de la personne, entre crochets :
   M : s'il s'agit d'un homme ;
   F : s'il s'agit d'une femme ;
   O : s'il ne s'agit ni d'un homme, ni d'une femme.
   Par exemple : [M] pour Raymond Loiselle ;

— attention à bien prendre en compte les personnes qui n'ont pas de relation connue, comme les 9e et 12e ;
— dans le tableau des relations d'une personne :
  — les relations sont triées en majeur dans l'ordre décroissant des degrés (*level*) et en mineur [3] dans l'ordre alphabétique du nom de famille de la personne en relation. Par exemple, pour la 2e personne, on a trois relations de degré 2. Elles sont donc classées dans l'ordre alphabétique de leur noms de famille : « Loiselle » d'abord, puis « Sanatana Parra » et enfin « Wielens » ;
  — dans la 2e colonne, les mêmes informations que celles des titres `<h2>` sont données sur les personnes, sauf le sexe qui n'est ici pas renseigné ;
  — les lignes de relation de niveau inconnu (−1) ont un fond grisé.

En vue de tester votre feuille de style XSL `addressbook.xsl`, vous pouvez copiez dans `05_xsl` le fichier `addressbook.xml` obtenu après avoir répondu à la **Q3**.

Pour répondre à la **Q5**, vous devez produire le fichier `addressbook.xsl` dans le répertoire `05_xsl`.

3. C'est-à-dire pour celles de même rang pour le critère majeur, donc ici pour les relations de même degré.

A Sources

Voici les sources qui ont été utilisées pour produire les données du fichier XML `addressbook_ko.xml` :

- Fake Name Generator : `https://fr.fakenamegenerator.com/`
- ISO 3166-1 (country codes) : `https://en.wikipedia.org/wiki/ISO_3166-1`
- List of country calling codes : `https://en.wikipedia.org/wiki/List_of_country_calling_codes`
- `xml:lang` : `https://www.w3.org/International/questions/qa-when-xmllang.en`
- IETF language tag : `https://en.wikipedia.org/wiki/IETF_language_tag`
- List of ISO 639-1 codes (language codes) : `https://en.wikipedia.org/wiki/List_of_ISO_639-1_codes`
- `xsd:date` : `https://www.w3.org/TR/2004/REC-xmlschema-2-20041028/datatypes.html#date`
- generatedata : `https://www.generatedata.com/`
- Telephone numbers in Belgium : `https://en.wikipedia.org/wiki/Telephone_numbers_in_Belgium`
- Telephone numbers in Germany : `https://en.wikipedia.org/wiki/Telephone_numbers_in_Germany`
- Telephone numbers in France : `https://en.wikipedia.org/wiki/Telephone_numbers_in_France`

B Contraintes

Les contraintes explicitées ci-dessous s'appliquent au fichier `addressbook.xml` et à ses données. Ce fichier est le fichier XML bien formé obtenu après avoir corrigé deux fautes de frappe (**Q1**) dans le fichier `addressbook_ko.xml` fourni dans `01_xml` et après y avoir mis un élément en commentaire lors de la résolution de la **Q3**.

Il est indispensable d'utiliser les contraintes explicitées ci-dessous pour répondre aux **Q2** et **Q3** !

B.1 Contraintes générales

Dans le fichier `addressbook.xml` :

- *tous les éléments sont obligatoires*, sauf ceux dont on dit ci-après que leur nombre est totalement arbitraire ;
- *l'ordre (l'emplacement) de chaque élément est fixé* ;
- *tous les attributs sont obligatoires* ;
- les *types des valeurs des éléments et des attributs sont des chaînes de caractères*, sauf s'il est mentionné le contraire dans les descriptions d'éléments et d'attributs ci-après.

B.2 `addressbook`

La racine **<addressbook>** possède exactement deux éléments enfants : d'abord **<persons>**, puis **<countries>**.

B.3 `persons`

Le nombre d'éléments **<person>**, enfants de l'élément **<persons>**, est totalement arbitraire.

B.3.1 `person`

Les enfants de l'élément **<person>** sont : **<identity>**, **<addresses>** et enfin **<relations>**, dans cet ordre.

Chaque élément **<person>** est unique au sein de l'élément **<persons>**. C'est l'attribut `id` de chaque **<person>** qui contrôle cette unicité.

`id` L'attribut `id` des éléments **<person>** est un identifiant unique au sein de l'élément **<persons>**.

Les valeurs de cet attribut sont construites sur le modèle : `p`n où n est un entier strictement positif. Par exemple : `p23`.

B.3.2 `identity`

Les éléments enfants de l'élément **<identity>** sont : **<lastname>**, **<firstnames>**, **<sex>**, **<birth>** et **<death>**, dans cet ordre.

B.3.3 `firstnames`

L'élément **<firstnames>** possède au moins un enfant **<firstname>**.

B.3.4 `sex`

Les seules valeurs autorisées pour l'élément **<sex>** sont : `MALE`, `FEMALE` ou `OTHER`.

B.3.5 `birth`

Le type des valeurs autorisées pour l'élément **<birth>** est une `xsd:date` (`https://www.w3.org/TR/2004/REC-xmlschema-2-20041028/datatypes.html#date`).

C'est donc une date sous la forme `YYYY-MM-DD`, où `YYYY` représente l'année, `MM` le mois, `DD` le jour et où la plus petite valeur de mois et de jour est `01`. Par exemple : `2020-03-16` pour le 16 mars 2020.

B.3.6 `death`

Les valeurs autorisées pour l'élément obligatoire **<death>** sont une `xsd:date` (voir **<birth>** à la section B.3.5) ou la valeur `NULL`. Cette dernière sert à indiquer l'absence de valeur pour **<death>** [4], c'est-à-dire que la personne n'est pas décédée.

B.3.7 `addresses`

Les éléments enfants de l'élément **<addresses>** sont : **<postal>**, **<emails>** et **<phones>**, dans cet ordre.

B.3.8 `postal`

Les éléments enfants de l'élément **<postal>** sont : **<country_id>**, **<postalcode>**, **<city>**, **<street>** et **<number>**, dans cet ordre.

B.3.9 `country_id`

Les valeurs attendues pour les éléments **<country_id>** sont celles qu'on trouve au sein des éléments **<alpha2code>**.

B.3.10 `emails`

Le nombre d'éléments **<email>**, enfants de l'élément **<emails>**, est totalement arbitraire.

B.3.11 `email`

La valeur de l'élément **<email>** est une adresse électronique.

use Les valeurs autorisées pour l'attribut `use` sont : `PRIVATE`, `PROFESSIONAL` ou `OTHER`.

B.3.12 `phones`

Le nombre d'éléments **<phone>**, enfants de l'élément **<phones>**, est totalement arbitraire.

B.3.13 `phone`

La valeur de l'élément **<phone>** est une chaîne de caractères constituée uniquement de chiffres de `0` à `9`, ces valeurs incluses.

use Les valeurs autorisées pour l'attribut `use` sont : `PRIVATE`, `PROFESSIONAL` ou `OTHER`.

4. Il est à remarquer que plutôt d'utiliser la valeur `NULL`, on aurait pu rendre l'élément facultatif ou utiliser `xsi:nil`.

type Les valeurs autorisées pour l'attribut `type` sont : `LANDLINE`, `CELLPHONE`, `IPPHONE` ou `OTHER`.

B.3.14 `relations`

Le nombre d'éléments **<relation>**, enfants de l'élément **<relations>**, est totalement arbitraire.

B.3.15 `relation`

Les éléments enfants de l'élément **<relation>** sont : **<person_id>** et **<level>**, dans cet ordre.

B.3.16 `person_id`

Les valeurs attendues pour les éléments **<person_id>** sont celles de l'attribut `id` des éléments **<person>**.

B.3.17 `level`

Les valeurs attendues pour les éléments **<level>** sont les valeurs entières comprises entre -1 et 4, ces valeurs incluses.

La signification de ces valeurs est :
-1 : inconnu (*unknown*) ;
0 : néant (*void*) ;
1 : faible (*low*) ;
2 : moyen (*medium*) ;
3 : élevé (*high*) ;
4 : très élevé (*higher*).

B.4 `countries`

Le nombre d'éléments **<country>**, enfants de l'élément **<countries>**, est totalement arbitraire.

B.4.1 `country`

Chaque élément **<country>** est unique au sein du document XML, c'est-à-dire de son élément racine **<addressbook>**. C'est le nœud textuel de l'élément `alpha2code` de chaque **<country>** qui contrôle cette unicité.

Les éléments enfants de l'élément **<country>** sont : **<alpha2code>**, **<names>** et **<callingcode>**, dans cet ordre.

B.4.2 `alpha2code`

Les valeurs des éléments **<alpha2code>** sont des codes de pays à deux lettres ISO 3166-1 (`https://fr.wikipedia.org/wiki/ISO_3166-1`) en majuscule.

Elles servent d'identifiant, unique au sein du document XML. Il est à remarquer qu'elles sont référencées par les éléments `<country_id>`.

B.4.3 `names`

Le nombre d'éléments `<name>`, enfants de l'élément `<names>`, est totalement arbitraire.

B.4.4 `name`

`xml:lang` L'attribut `xml:lang` est l'attribut standard `xml:lang`. On trouve des informations à son sujet ici : `https://www.w3.org/International/questions/qa-when-xmllang.en`, par exemple.

Les valeurs de cet attribut sont des codes de langues à deux lettres ISO 639 (`https://fr.wikipedia.org/wiki/Liste_des_codes_ISO_639-1`) en minuscule.

B.4.5 `callingcode`

Les valeurs des éléments `<callingcode>` sont des indicatifs téléphoniques internationaux (`https://fr.wikipedia.org/wiki/Liste_des_indicatifs_t%C3%A91%C3%A9phoniques_internationaux_par_indicatif`). Il s'agit donc d'entiers strictement positifs.