XML Data Management **Fall-Winter 2011/12**

Exercises **Werner Nutt**

9. XQuery

Hint: Run your XQuery queries using Kernow. You can download Kernow from

```
http://kernowforsaxon.sourceforge.net/
```

Kernow is a Java package and can be run under Linux, Windows and Mac OS. Kernow is a graphical frontend to the Saxon XQuery/XSLT processor.
You can develop your queries in the XQuery Sandbox. To tell Kernow that the output should be properly indented, type as the first line:

```
declare option saxon:output "indent=yes";
```

The version of Kernow that you download allows you to run 100 queries. Then it will ask you to enter a code in order to proceed. I have the code and will send it to you if you ask me by email.

1. Recipes

The following are queries over the `recipes.xml` document. Formulate your queries in such a way that they return the correct result for all possible recipe documents that satisfy `recipes.dtd`.

1. Return a list, containing for every recipe the recipe's title element and an element with the number of calories
2. Return a similar list, ordered according to calories.
3. Return a similar list, alphabetically ordered according to title.
4. Return a similar list, ordered according to the fat content.
5. Return a similar list, with title as attribute and calories as content.
6. Return a list, containing for every recipe the top level ingredients, dropping the lower level ingredients

2. Countries

The following are queries over the `countries.xml` document. Formulate your queries in such a way that they return the correct result for all possible recipe documents that satisfy `countries.dtd`.

1. Return a list of <`country`> elements, containing elements <`name`> and <`population`>, ordered alphabetically by population.

2. Return a similar list, ordered by population in descending order.

3. Return a similar list, again ordered by population in descending order, with an additional attribute `rank`, which indicates the position of the country on the list.

   Try two approaches, one using the construct

   ```
   for $var at $pos in <expression>
   ```

   and another one without. Note that you may have to use the casting functions `number` and `string`.

4. Return a list of city elements, containing the name of the city, such that each city has an attribute population and another attribute country. The cities are returned according to their population, in descending order.

5. Restructure the document by listing countries according to population, cities within each country according to population, and languages within each country according to percentage. Try to reuse as much as possible existing elements and attributes instead of creating new ones. (Creation of nodes is usually a costly operation in XQuery engines.)

6. Return a list of language elements, alphabetically sorted, where each language element containts a list of country elements, such that the language is spoken in the country, together with the number of speakers of the language in that country. You may again need a casting function (`xs:int` casts an argument as an integer, whenever that is possible; `xs` is the default prefix for the XML Schema namespace).