CS1003 Week 3 Exercise 2: XML

As with all lab exercises, this exercise is not assessed. It is intended solely to help you understand the module material. This sheet is intended to include problems for both exercise classes this week. There is probably more here than you will have time for during the lab hours; you are encouraged to complete it in your own time if you don't finish it. You will need to look at online resources, especially the documentation of key Java library classes.

The aim of this exercise is to practice writing Java programs that read or write semi-structured data using various libraries.

This exercise builds on Exercise 1 and you should complete that before attempting this one. You will be working with the same `hip2.dat` data set.

1. Write a new program (or extend the one from before) to write the data in XML format. The head element of the document should be a `<star-catalog>` which should contain a large number of `<star>` elements, each with attributes `"StarID", "right-ascension", "declination", "parallax"` and `"magnitude".`

   You will need to look at the DOM functionality for creating XML documents from scratch. Look at the documentation for the `DocumentBuilder` class (method `newDocument`); the `Document` class (method `createElement`) the `Element` class (method `setAttribute`) and the `Node` class (`appendChild`).

   You can copy the code from the `XMLPrune` example to output the `Document` to a file.

2. The XML schema `stars.xsd` provided specifies an XML format consisting of a star-catalog element with many star elements within it. Your outputs should be valid with respect to this schema. You can check this online using a validator such as http://www.corefiling.com/opensource/schemaValidate.html. Adjust your programs as necessary to make the XML valid. Once again, you may wish to choose a shorter file to test first.

Optional Extensions:

- The distance to a star in parsecs (one parsec is about 3 light-years) is 1000/(parallax in milliarcseconds). Modify your programs to compute the distances of the stars as they read them and include that instead of the parallax in the output. As a further step, if you are interested in astronomy, compute the absolute magnitudes as well.
- Both the `JsonGenerator` and the `Transformer` classes that we use for output have capabilities to automatically format the output to make it more human-readable. Look up pretty-printing for Json and automatic indenting for XML. Alternatively, you can insert text nodes containing whitespace in the DOM tree.

Submission

You are required to submit a zip file containing your exercise attempt to the appropriate slot in the Exercises category on MMS. Note that if you have done this work on the lab servers as suggested, you will need to upload your submission to your local machine before submitting to MMS.

To recursively zip the contents of a folder on Linux you can do this:

```
cd ~/Documents/CS1003/Exercises/W03Exercise2          (if you are not already there)
zip -r W03Exercise2.zip *
```

This will create a file called W03Exercise2.zip containing the entire contents of the current directory (files that match *).

Suppose that you have done that operation from this directory: `~/Documents/CS1003/Exercises` on the lab machine called `pc3-002-l`, you can copy the zip file from my local machine using scp as follows:

```
scp pc3-002-l.cs.st-andrews.ac.uk:~/Documents/CS1003/Exercises/W03Exercise2.zip .
```

and you will have a copy of the zip file in the working directory from which you executed the scp command. I can then submit the file to MMS using the web interface.

You could also use FileZilla instead of scp to download the file to your local machine (but you should really master scp).