4. Unit: XML Processing with Java (II)

Exercise 4.1 (XML Digester: River Networks in Mondial) Use the XML Digester in a Java class that creates an internal data structure about seas, rivers and their tributaries (including lakes) and generates an output similar to that in Exercise 2.1.
Additionally, this output should show for every river the sum of the length of all rivers flowing into it (directly or indirectly).

Exercise 4.2 (XML Schema & JAXB: Arithmetic Trees)

Design an XML markup for representing *arithmetic term trees* over integers (cf. Exercise Sheet 3 (XSLT), Exercise 4 of the XML lecture) that is appropriate for use with JAXB. Define an XML Schema and create base classes. Extend these classes with the common functionality for arithmetic trees: (i) loading a tree, (ii) outputting the tree contents in as a term, (iii) outputting the tree in a graphical way (e.g., nested tables) in HTML, (iv) evaluating the term.

Exercise 4.3 (Independence in JAXB)

- Do the independence use case exercise using JAXB.
  (`mondial.xsd` is available on the Web site)
  Do as much as possible with JAXB. Where does a problem occur, and why? Solve it with a reasonable workaround.
- Why is it not a good idea to do it with the Digester?

Exercise 4.4 (Parsing of Sloppy HTML Pages: Analysis of XML Tools) Often, HTML pages are not strictly valid XHTML, and thus cannot be queried by XQuery (like wikipedia, country pages from `https://www.cia.gov/library/publications/the-world-factbook/` or from `https://www.citypopulation.de/`, soccer tables from `https://www.kicker.de/`). Usually, the problem are empty elements that consist only of opening tags instead of correct `<... />` syntax.

Consider the XML Parsing APIs DOM, SAX and StAX when parsing such input.

- (i) what happens?
- (ii) what does that "mean" wrt. the underlying metaphor of the DOM tree and of the stream in SAX/StAX?
- (iii) where does it *exactly* happen in the Java code (you can use the previous Mondial examples by just exchanging mondial.xml with a sloppy HTML input)?
- (iv) as a consequence, which of the APIs can be rather easily adapted to be able to handle sloppy HTML?

Demonstrate your solution by implementing some simple query against a wikipedia page (e.g., elevation of the Mont Blanc).

If you like, you can extend it to loop over all mountains from mondial.xml.

Exercise 4.5 (HTML Web Data Extraction: Wikipedia) Use the Wikipedia HTML pages for the following:

a) Parse the HTML page with an HTML parser (e.g. JSoup),

b) Write an Java method that is invoked with the name of a mountain (e.g. "Mont Blanc") that returns a small XML fragment with data about that mountain.

c) Extend your program to apply this to invoke the function for all mountains in Mondial that are located in Germany.

Exercise 4.6 (XML Schema vs DTD: Analysis)

XMLSchema is basically like DTDs, but with atomic datatypes and with an XML syntax and more engineering-style “commands” how to create structural descriptions.

Is there something else that cannot be described by DTDs, but with XML Schema?
Hints:

- not something in very detailed expressiveness, but some rather common structural aspect,
- for finding it, it is merely sufficient to look at the basic conctructs of DTDs and to see what they can do and what then cannot do.