3. Document Type Definitions

This sheet contains exercises about DTDs.

1. A DTD for Recipes

Consider again the document `recipes.xml` on the course website. Write a DTD `recipes.dtd` that captures this document as tightly as possible. This means,

- the document should be valid with respect to the DTD;
- if in any place you constrain the DTD further, then the document does not satisfy the DTD any more.

Constraining the DTD further means, for instance, turning an expression `element*` into `element+`, `element?` or `element`. Other examples of constraining the DTD further are to drop an alternative in an "OR" expression, or to make an optional attribute mandatory.
Connect your document to the DTD. Try both approaches of making it an internal and an external DTD.

2. How a DTD Can Help the DOM Parser

Load the the document `recipes.xml` into Java with the DOM parser.
Write a method that counts the number of *arbitrary* children of the top level element and use it to count the number of children of the element `recipes` in the current document. How many children does it report?
Connect the document to the DTD. In your Java code, make sure that the flag `IgnoringElementContentWhitespace` of the DocumentBuilderFactory that creates your DocumentBuilder has been set to `true`. How many children of `recipes` does Java report now?

3. A DTD for a Student List

Suppose you need to define a structure for XML documents that list students in the Faculty of Computer Science. Consider the following requirements.

- A `studentlist` consists of one or more student entries.
- A student has a matriculation number by which they can be uniquely identified. A student entry consists of the student's name, the study programme in which the student is enrolled, and the courses the student has taken so far. For each student, we need to record as well the year when the student enrolled.
- A student's name consists of the first name, the middle name, which is optional, and the last name.
- Study programmes are "BSc in CS and Engineering", "BSc in Applied CS", "Master in CS", and "PhD in CS".
- Each course entry contains the name of the course. Courses are undergraduate (Bachelor) courses, postgraduate (Master) courses, or both. A course can also be mandatory or optional.
- A course entry contains the title of the course. We want to record as well whether the student has already received a course work mark for the course and, if so, the mark. Moreover, we want to record whether the student has already taken an exam for the course and, if so, what was the outcome of the exam.

Tasks:

1. Write a DTD that will define the following structure for documents of type `studentlist`.
2. Give an instance of a `studentlist` document that is *valid* with respect to your DTD.
3. Give an instance of a `studentlist` that is *invalid* with respect to your DTD. Make sure that the instance contains only element and attribute names introduced in the DTD. Explain why it is invalid.