ISY00246 – Client/Server Systems

Assignment 2 (S2 2015)

This assignment is due about on Friday 18th September 2015. You should submit your source code (Java, XML, etc) for the all parts of the assignment via e-mail to your tutor (or by other means approved by your tutor). Note that marks will be deducted for poorly structured or uncommented code. All source code files submitted must include title comments that at least identify the author of the code and the assignment part. The separate parts of the assignment are to be submitted in separate subdirectories (e.g. Part1, Part2 etc.) – submissions that ignore this instruction and leave all assignment files in one directory may be penalised.

Please note that this assignment will be due several weeks after the suggested study timetable has covered the last of required materials. Do not leave this assignment to the last minute – do it while concepts are fresh in your mind. If you require an extension you must apply to your tutor before the due date to be considered.

Part 1 – XML Schema (7 marks)

The following is an example of an XML document type. It represents the members of a sports team.

```
<?xml version="1.0"?>
<!-- the A team details -->
<team>
  <teamDetail id="theATeam">
     <leader>
        <name>Barry Wilks</name>
        <phone>9999 3002</phone>
        <email>barwil@mail.com</email>
     </leader>
     <fullname>The A Team</fullname>
     <players no="3"/>
  </teamDetail>
  <member name="Jo Player" position="left half">
    <mailAddress>Long Road, Upper Woop Woop West, NSW</mailAddress>
    <altPosition>left wing</altPosition>
    <feesPaid>100.00</feesPaid>
  </member>
  <member name="Peter Player" position="goal keeper">
    <mailAddress>Short Road, Woop Woop, NSW</mailAddress>
    <feesPaid>0.00</feesPaid>
  </member>
  <member name="Barry Wilks" position="back">
    <mailAddress>23 Jetty Street, Coffs Jetty, NSW</mailAddress>
    <altPosition>left right out</altPosition>
    <feesPaid>100.00</feesPaid>
  </member>
</team>
```

Elements have the following restrictions:

a. `<teamDetail>` elements must appear once and only once. The `id` attribute is compulsory and can be any alphanumeric text between 4 and 10 characters long. The `<fullname>` element can be any length and contain any printable characters.

b. `<name>` and `<phone>` elements in the `<leader>` element are compulsory and occur only once. The `<email>` element is optional (0 or 1 occurrence). Each of these elements can be any non-empty string.

c. the `<players>` elements in the `<teamDetail>` element is compulsory and the `no` attribute must have a number between 2 and 20 (inclusive).

d. `<member>` elements occur between two and twenty times (inclusive). The `name` and `position` attributes are compulsory and can contain any text.

e. `<mailAddress>` and `<feesPaid>` elements are compulsory and the `<altPosition>` element is optional.

f. the `<feesPaid>` element can only have values 0.0, 10.0, 50.0 and 100.0. No other values are allowed.

Write and test an XML Schema for these XML documents. One mark will be deducted for each element incorrectly specified (and one mark is for a well laid out Schema definition).

Part 2 – Processing XML in Java (7 marks)

Write a program that will read XML documents specified in Part A. Your program should print a report in a format as close as possible to that below:

```
Team details for "The A Team" (code theATeam):

Players:
Jo Player, left half (Fees paid: $100.00)
    Long Road, Upper Woop Woop West, NSW
    Alt Position: left wing
Peter Player, goal keeper (Fees paid: $0.00)
    Short Road, Woop Woop, NSW
Barry Wilks, back (Fees paid: $100.00)
    23 Jetty Street, Coffs Jetty, NSW
    Alt position: left right out

Total of 3 team member(s).
Team contact Barry Wilks 9999 3002, barwil@mail.com
```

Note that you do not have to get Part A working to do this. Just test your program by giving a correct document to process. Two marks are allocated for a well structured program with correct package and class inclusions. Remaining marks are allocated based on how much of the document you correctly process (e.g. one mark for interpreting the root element). Note that the player's details are indented a certain number of columns. You also need to count the number of `<member>` elements to produce the second last line of the report. Your program will be tested with a larger and more diverse XML file using the same format (per schema).

Part 3 – A Servlet for a simple stock market database (6 marks)

Write a web application in NetBeans based on a Browser client with a simple home page (HTML or JSP) and a single Servlet to process requests. The application will maintain a simple server in-memory database of stock information (1 mark). Data for each stock will consist of:

Code – a 1 to 8 character code for the company

Full name – a string containing the full company name

Price – current price in cents

Web page – a URL to the company's web page

Your application should contain implement the following use-cases:

Add a stock (1 mark)

Delete a stock (1 mark) – include a "Do you really want to delete XXX" screen.

Search for a stock by code and display details *so that they may be updated* (1 mark)

List all stocks (1 mark)

Your application should implement useful and easy to use control structures (1 mark) using HTML links. For example, you should be able to click on the URL for a company's web page to go there. Remember that the servlet can generate many different pages, each with its own format and these pages should contain the links or other controls necessary to make the application easy to use. Think through the above use-cases and how/why the user would use them. Remember a user may want to do more than one use-case in one session so links to the main menu/options area should be available. Your assignment marker will certainly do this while testing your application.

Note that there is opportunity for very fancy applications here. Make sure you have addressed each of the requirements before you try to improve the appearance of your application.