Exercise 1- XML example to Hand Up

- Write down a "well formed" XML snippet, using elements and/or attributes, describing:
  - Your name (distinguishing first, middle, surname)
  - Student ID
  - Favourite music groups
  - County
  - Expected date of graduation

- **Well formed-** XML Declaration required, Exactly one root element, Empty elements are written in one of two ways:Closing tag or Special start tag, For non-empty elements, closing tags are required, Attribute values must always be quoted, Start tag must match closing tag (name & case),Correct nesting of elements

Sample XML to show Syntax

```
<?xml version="1.0"?>
<!DOCTYPE catalog SYSTEM "books.dtd">
<catalog>
   <book id='bk101' type='softback'>
      <author>Gambardella, Matthew</author>
      <title>XML Developer's Guide</title>
      <genre>Computer</genre>
<price>44.95</price>
      <publish_date>2000-10-
01</publish_date>
      <description>An in-depth look at
creating applications with XML.
</description>
</book>
<book id='bk102' type='hardback'>
      <author nationality='irish'>Jenkins,
Fred</author>
      <title>XML Technology Guide</title>
          <price>50.00</price>
      <publish_date>2000-10-
01</publish_date>
      <description>An in-depth look at
using XML technologies.</description>
          <stocked_by>Easons</stocked_by>
          <stocked_by>Amazon</stocked_by>
   </book >
</catalog>
```

Solution Exercise 1

```
<?xml version="1.0" ?>

<student_info>
    <student_name student-id="1234 >
        <firstname> Declan </firstname>
        <surname>O'Sullivan</surname>
    </student_name>
    <fav_music>
            <band> U2 </band>
            <band> beatles </band>
    </fav_music>
</student_info>
```

Exercise 2- Suggest a DTD

```
<?xml version="1.0"?>
<!DOCTYPE catalog SYSTEM "books.dtd">
<catalog>
   <book id='bk101' type='softback'>
      <author>Gambardella, Matthew</author>
      <title>XML Developer's Guide</title>
      <genre>Computer</genre>
<price>44.95</price>
      <publish_date>2000-10-01</publish_date>
      <description>An in-depth look at creating
applications with XML.
</description>
</book>
<book id='bk102' type='hardback'>
      <author nationality='irish'>Jenkins,
Fred</author>
      <title>XML Technology Guide</title>
         <price>50.00</price>
      <publish_date>2000-10-01</publish_date>
      <description>An in-depth look at using XML
technologies.</description>
         <stocked_by>Easons</stocked_by>
         <stocked_by>Amazon</stocked_by>
   </book >
</catalog>
```

EXAMPLE DTD to show SYNTAX

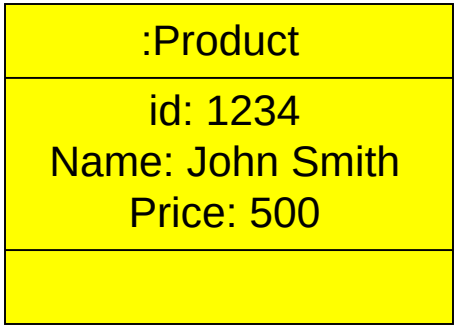
```
<!DOCTYPE NEWSPAPER [

<!ELEMENT NEWSPAPER (ARTICLE+)>
<!ELEMENT ARTICLE
    (HEADLINE,BYLINE+,LEAD?,BODY,NOTES*)>
<!ELEMENT HEADLINE (#PCDATA)>
<!ELEMENT BYLINE (#PCDATA)>
<!ELEMENT LEAD (#PCDATA)>
<!ELEMENT BODY (#PCDATA)>
<!ELEMENT NOTES (#PCDATA)>

<!ATTLIST ARTICLE AUTHOR CDATA #REQUIRED>
<!ATTLIST ARTICLE EDITOR CDATA #IMPLIED>
<!ATTLIST ARTICLE DATE CDATA #IMPLIED>
<!ATTLIST ARTICLE EDITION CDATA #IMPLIED>

<!ENTITY NEWSPAPER "Trinity Times">
<!ENTITY PUBLISHER "Trinity Press">
<!ENTITY COPYRIGHT "Copyright 1998 TCD Press">

]>
```

Solution Exercise 2

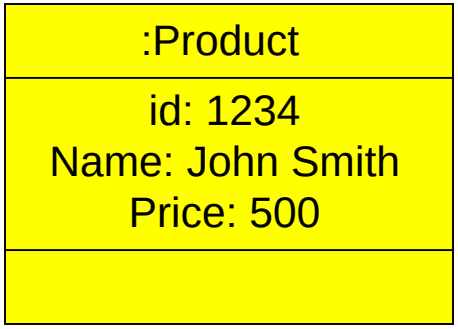
1. DTD

```
2. <!DOCTYPE catalog [
 3. <!ELEMENT catalog    (book+) >
 4. <!ELEMENT book       (author, title, genre?,
    price, publish_date, description,stocked_by*) >
 5. <!ATTLIST book   id ID #REQUIRED >
 6. <!ATTLIST book   type (hardback|softback)
    #REQUIRED >
 7.
 8. <!ELEMENT author          (#PCDATA)   >
 9. <!ATTLIST author nationality CDATA #IMPLIED >
 10.    <!ELEMENT title          (#PCDATA)   >
 11.    <!ELEMENT genre          (#PCDATA)   >
 12.    <!ELEMENT price          (#PCDATA)   >
 13.    <!ELEMENT publish_date   (#PCDATA)   >
 14.    <!ELEMENT description    (#PCDATA)   >
 15. <!ELEMENT stocked_by    (#PCDATA)   >
 16. ]>
```

Exercise 3- Convert UML 2 XML

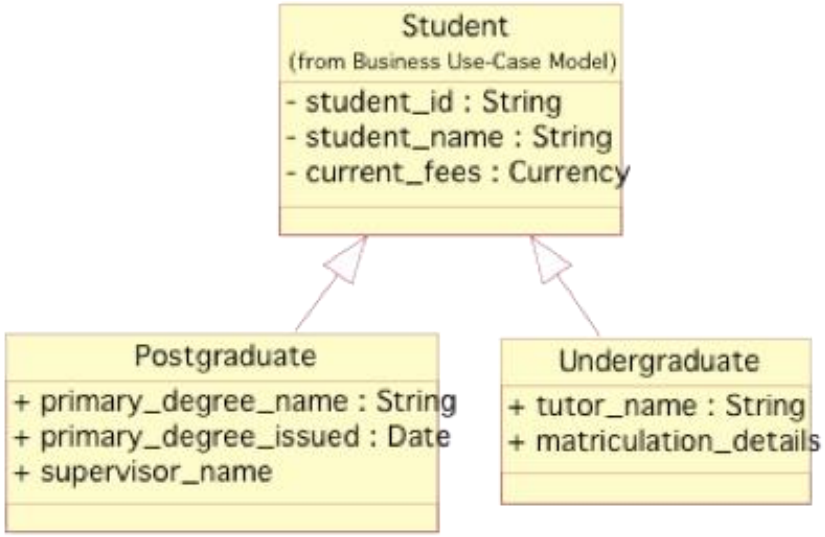


Solution Exercise 3




```
<Product>
   <Product. id> 1234 </Product.id>
   <Product.name> Lens </Product.name>
   <product.price> 500 </Product.price>
</Product>
```

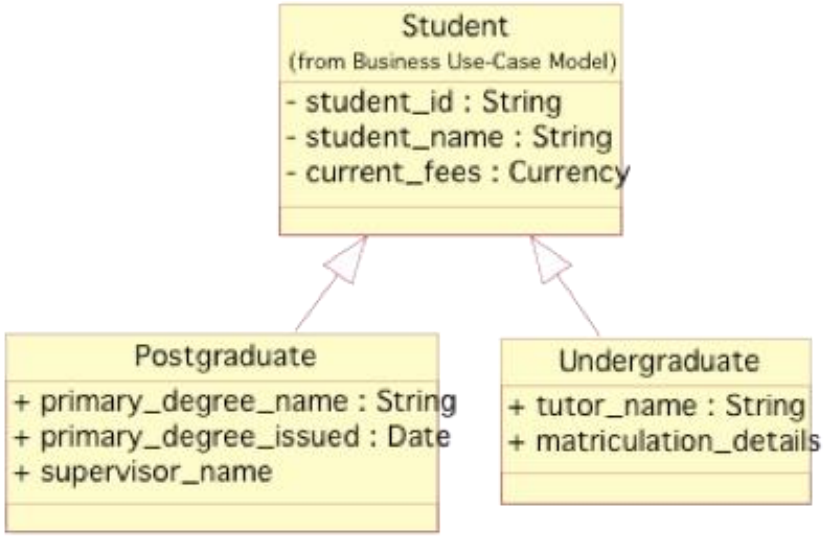
Exercise 4 - Convert UML 2 XML

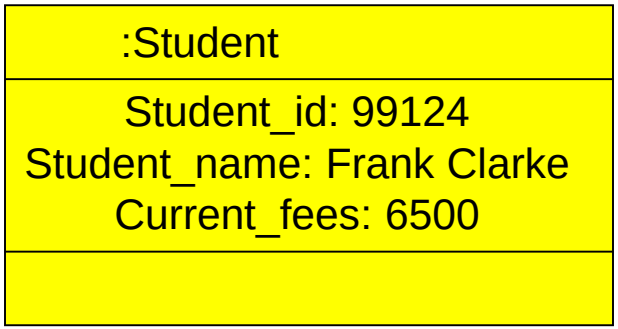


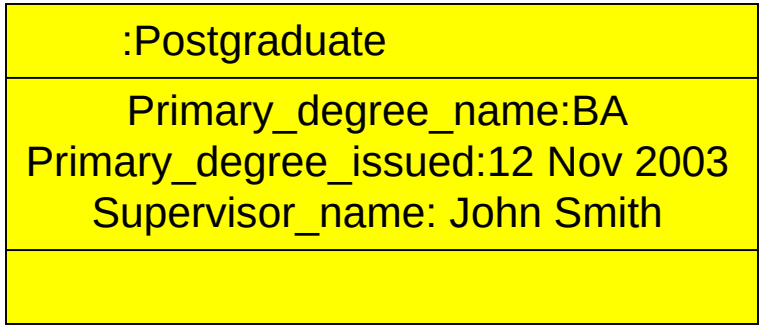
:Student
Student_id: 99124 Student_name: Frank Clarke Current_fees: 6500

:Postgraduate
Primary_degree_name:BA Primary_degree_issued:12 Nov 2003 Supervisor_name: John Smith

Solution Exercise 4







<Postgraduate>
<Student.student_id> 99124 </Student.student_id>
<Student.student_name> Frank Clarke </Student.student_name>
<Student.current_fees> 6500 </Student.current_fees>
<Postgraduate.primary_degree_name> BA </Postgraduate.primary_degree_name>
<Postgraduate.primary_degree_issued > 12 November 2003 </Postgraduate.primary_degree_issued>
<Postgraduate.supervisor_name > John Smith </Postgraduate.supervisor_name>
</Postgraduate>

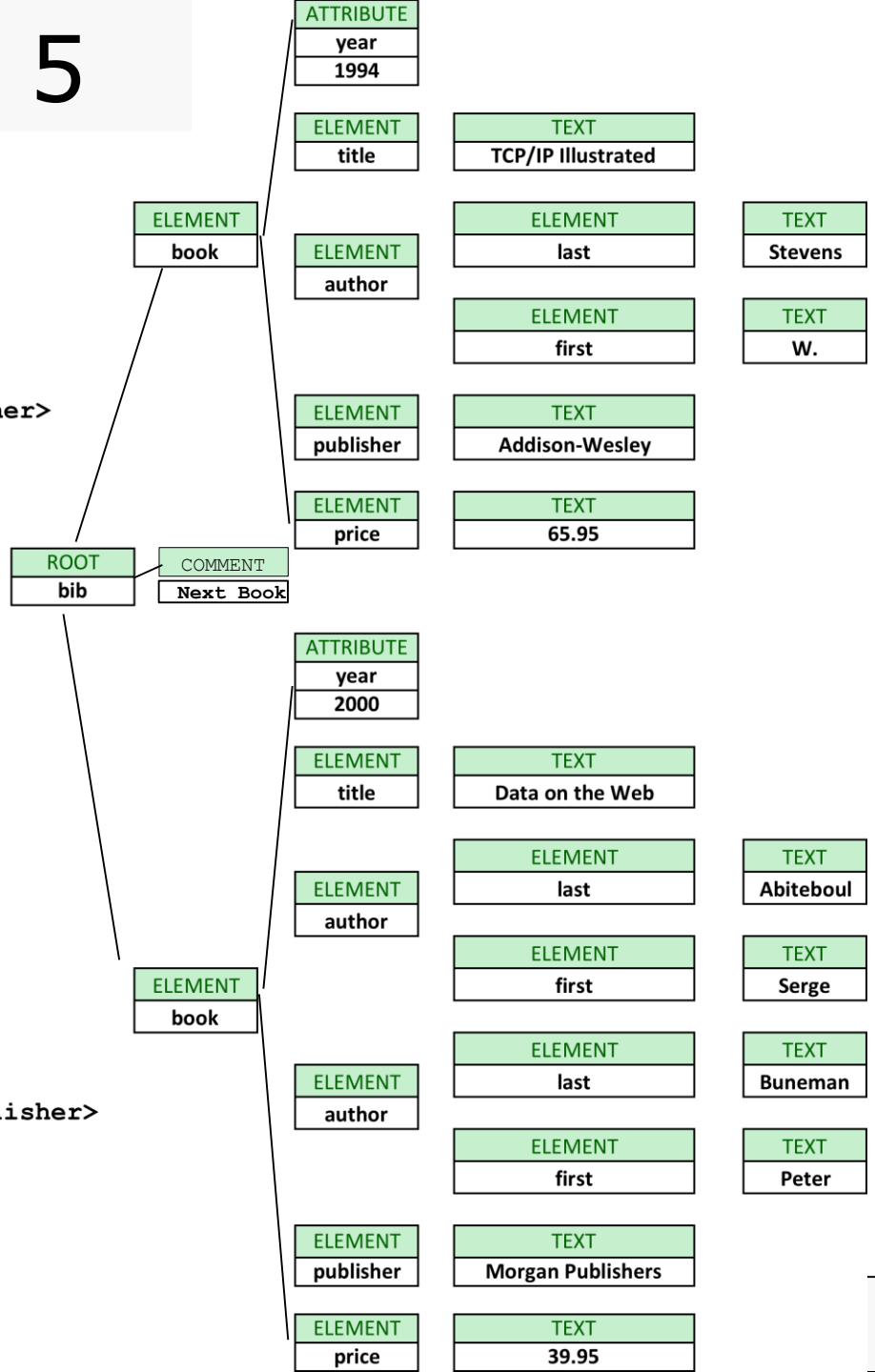
Exercise 5

- Create a XML Tree representation for the snippet of XML

```
<bib>
  <book year="1994">
    <title>TCP/IP Illustrated</title>
    <author>
      <last>Stevens</last>
      <first>W.</first>
    </author>
    <publisher>Addison-Wesley</publisher>
    <price>65.95</price>
</book>
<!-- Next Book --!>
  <book year="2000">
    <title>Data on the Web</title>
    <author>
      <last>Abiteboul</last>
      <first>Serge</first>
    </author>
    <author>
      <last>Buneman</last>
      <first>Peter</first>
    </author>
<publisher>Morgan Publishers</publisher>
    <price>39.95</price>
  </book>

</bib>
```

Solution Exercise 5

```
<bib>
  <book year="1994">
    <title>TCP/IP Illustrated</title>
    <author>
      <last>Stevens</last>
      <first>W.</first>
    </author>
    <publisher>Addison-Wesley</publisher>
    <price>65.95</price>
  </book>



   <!-- Next Book --!>



  <book year="2000">
    <title>Data on the Web</title>
    <author>
      <last>Abiteboul</last>
      <first>Serge</first>
    </author>
    <author>
      <last>Buneman</last>
      <first>Peter</first>
    </author>
    <publisher>Morgan Publishers</publisher>
    <price>39.95</price>
  </book>

</bib>
```



Exercise 6:Design XPath queries

2. Get all the titles of books in the file (without using //)

3. Get just the text from the first name elements of author

4. Return only the book elements that has an editor

5. Return only the books that are published after 1998

6. Return the entire book element whose title is "Data on the Web"

7. Alter the last query to just return the second author

8. Return those books which are priced between 50 and 100 only

9. Return all those books that are NOT published by Addison-Wesley

```
<?xml version="1.0" ?>
  <?xml version="1.0" ?>
  <bib>
     <book year="1994">
        <title>TCP/IP Illustrated</title>
        <author><last>Stevens</last><first>W.</first></author>
        <publisher>Addison-Wesley</publisher>
        <price>65.95</price>
     </book>

     <book year="1992">
        <title>Advanced Programming in the Unix
environment</title>
        <author><last>Stevens</last><first>W.</first></author>
        <publisher>Addison-Wesley</publisher>
        <price>65.95</price>
     </book>

     <book year="2000">
        <title>Data on the Web</title>
<author><last>Abiteboul</last><first>Serge</first></author>
<author><last>Buneman</last><first>Peter</first></author>
        <author><last>Suciu</last><first>Dan</first></author>
        <publisher>Morgan Kaufmann Publishers</publisher>
        <price>39.95</price>
     </book>

     <book year="1999">
        <title>The Economics of Technology and Content for
Digital TV</title>
        <editor>
             <last>Gerbarg</last><first>Darcy</first>
              <affiliation>CITI</affiliation>
        </editor>
           <publisher>Kluwer Academic Publishers</publisher>
        <price>129.95</price>
     </book>
```

Exercise 6:Sample Solution

2. Get all the titles of books in the file (without using //)

```
/bib/book/title
```

3. Get just the text from the first name elements of author

```
//first/string()
```

4. Return only the book elements that has an editor

```
//book[editor]
```

5. Return only the books that are published after 1998

```
//book[@year>=1998]
```

6. Return the entire book element whose title is "Data on the Web”

```
 //book[title/string()="Data on the Web"]
```

7. Alter the last query to just return the second author

```
//book[title/string()="Data on the Web"]/author[2]
```

8. Return those books which are priced between 50 and 100 only

```
 //book[price>50][price<100]
```

9. Return all those books that are NOT published by Addison-Wesley

```
 //book[publisher!="Addison-Wesley"]
```

Exercise 7

Source

```
<database>
<person age='34'>
    <name>
        <title> Mr </title>
        <firstname> John </firstname>
        <firstname> Paul </firstname>
        <surname> Murphy </surname>
    </name>
    <hobby> Football </hobby>
    <hobby> Racing </hobby>
</person>

<person >
    <name>
        <firstname> Mary </firstname>
        <surname> Donnelly </surname>
    </name>
</person>
</database>
```

- Example syntax

```
let $c:=
   doc("data/tcd.xml")/assessments
   /course/mark
return
   <list_of_avg_course_marks>
   {$c}
   </list_of_avg_course_marks>

for $j in
        doc("data/tcd.xml")/assess
   ments/course/@name
return
   <one_of_courses_is>
   {$j}
   </one_of_courses_is>
```

Define a query which will return an element called "paul_hobbys" which contains the hobby elements for each of person elements who have "Paul" as a firstname

Solution Exercise 7

Source

```
<database>
<person age='34'>
    <name>
        <title> Mr </title>
        <firstname> John </firstname>
        <firstname> Paul </firstname>
        <surname> Murphy </surname>
    </name>
    <hobby> Football </hobby>
    <hobby> Racing </hobby>
</person>

<person >
    <name>
        <firstname> Mary </firstname>
        <surname> Donnelly </surname>
    </name>
</person>
</database>
```

XQuery

```
for $p in
    doc("persondb.xml")/database/person
where $p/name/firstname=" Paul "
return
<paul_hobbys>
{$p/hobby}
</paul_hobbys>
```

Result

```
<paul_hobbys>
   <hobby> Football </hobby>
   <hobby> Racing </hobby>
</paul_hobbys>
```