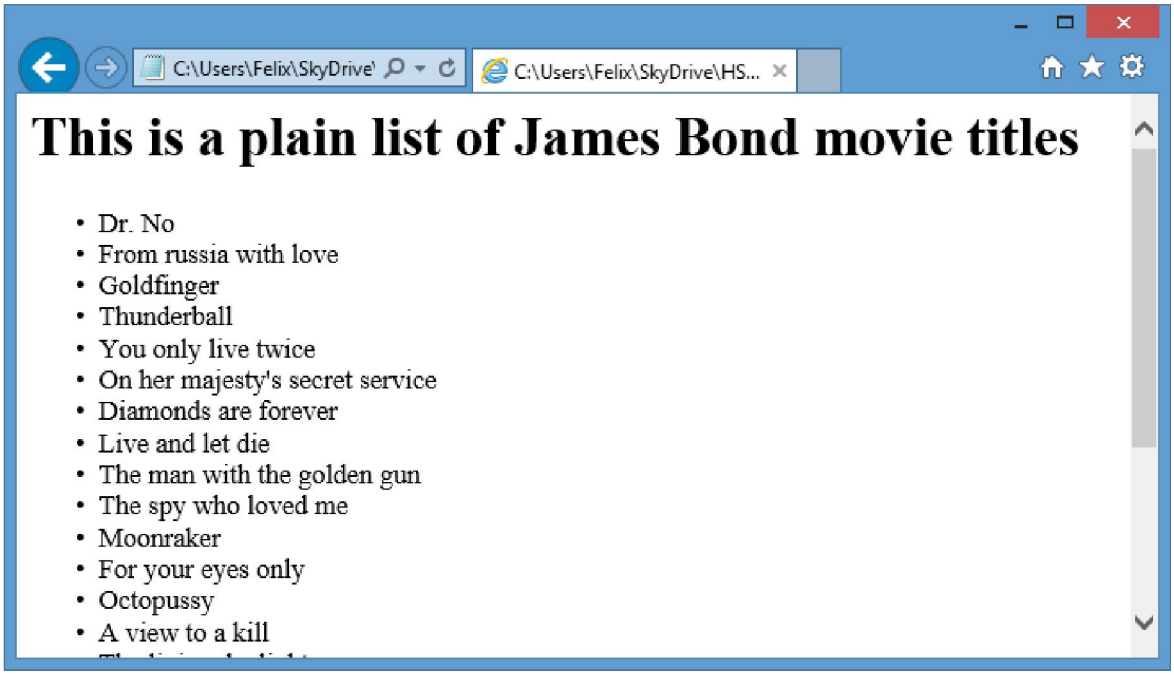
5 – XSLT | Experiment: XSLT in a Browser

1. Open the file bond_movie_list.xml and verify the presence of the following processing instruction:
   ```
   <?xml-stylesheet type="text/xsl" href="bond_movie_list.xsl"?>
   ```
   check

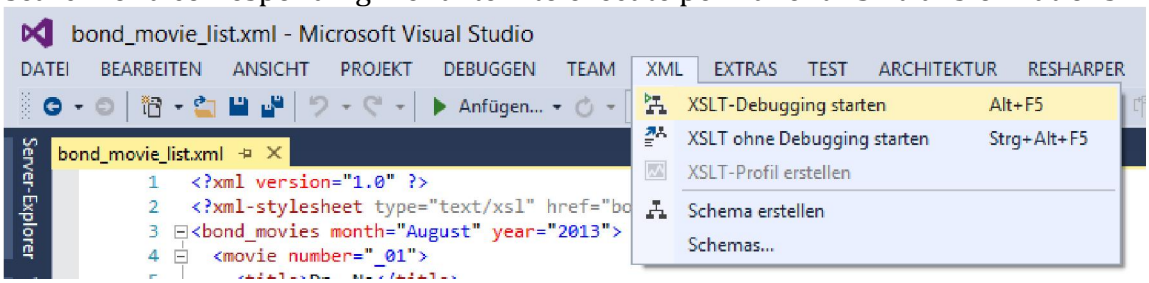
2. Drag-and-drop the XML file to a browser window. This executes the referenced XSL file, which is a transformation from XML to HTML. Your browser can directly interpret and display the result.
   Chrom: dafu…
   Firefox & IE: check



3. This is a quick way to test transformations into browser languages such as HTML, XHTML, SVG, MathML, ...
   ☺

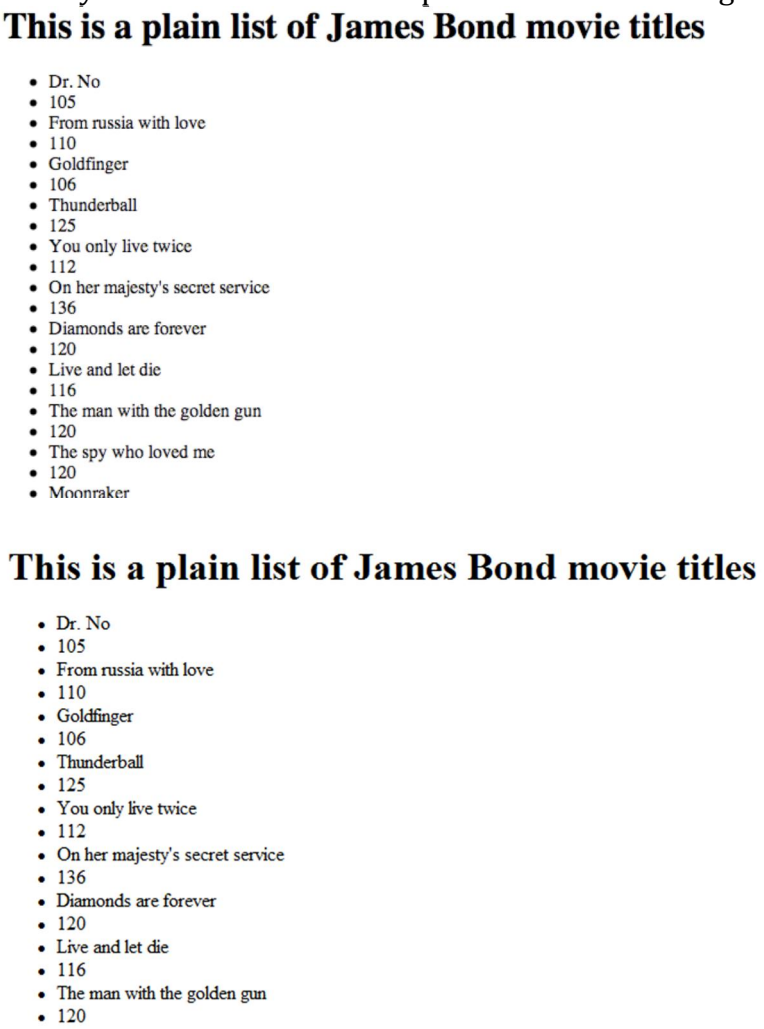
Experiment: Persistent Transformations

1. Open the file bond_movie_list.xml in your favorite XML development environment and search for a corresponding menu item to execute permanent XSL transformations.



Exercise

1. Modify the XSL file such that it produces the following output.



This is a plain list of James Bond movie titles

- Dr. No
- 105
- From russia with love
- 110
- Goldfinger
- 106
- Thunderball
- 125
- You only live twice
- 112
- On her majesty's secret service
- 136
- Diamonds are forever
- 120
- Live and let die
- 116
- The man with the golden gun
- 120
- The spy who loved me
- 120
- Moonraker

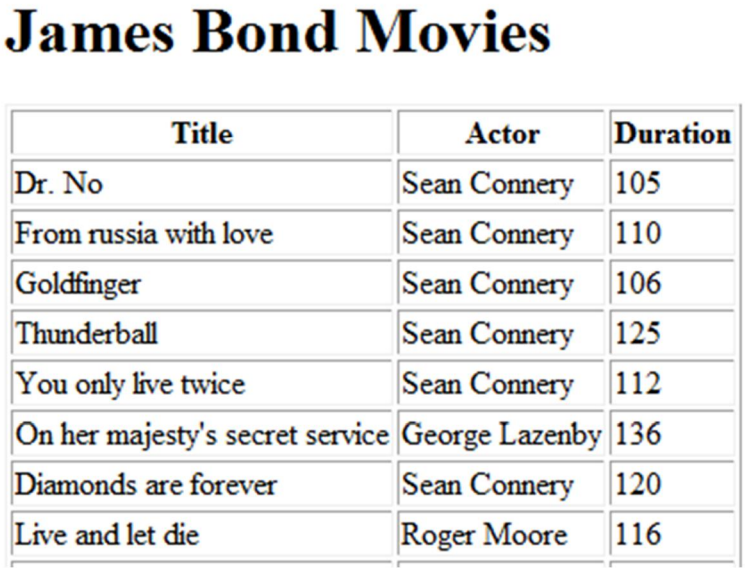
This is a plain list of James Bond movie titles

- Dr. No
- 105
- From russia with love
- 110
- Goldfinger
- 106
- Thunderball
- 125
- You only live twice
- 112
- On her majesty's secret service
- 136
- Diamonds are forever
- 120
- Live and let die
- 116
- The man with the golden gun
- 120

```
<xsl:template match="movie">
  <li>
    <xsl:value-of select="title"/>
  </li>

  <li>
    <xsl:value-of select="duration"/>
  </li>
</xsl:template>
```

Exercise

1. Write a stylesheet to transform the XML file bond_movies.xml into a HTML file similar to the one displayed on the right.



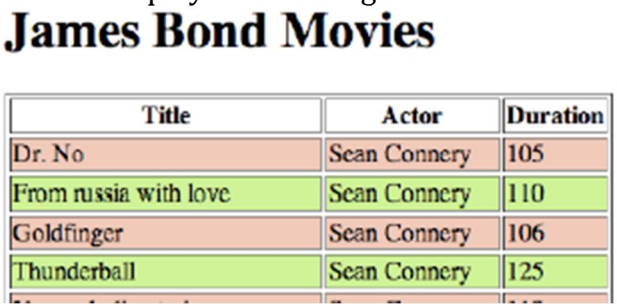
James Bond Movies

Title	Actor	Duration
Dr. No	Sean Connery	105
From russia with love	Sean Connery	110
Goldfinger	Sean Connery	106
Thunderball	Sean Connery	125
You only live twice	Sean Connery	112
On her majesty's secret service	George Lazenby	136
Diamonds are forever	Sean Connery	120
Live and let die	Roger Moore	116

```
<?xml version="1.0" ?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
      <html>
          <body>
              <h1>James Bond Movies</h1>
            <table border="1">
              <tr>
                <th>Title</th>
                <th>Actor</th>
                <th>Duration</th>
              </tr>
              <xsl:apply-templates />
            </table>
          </body>
      </html>
  </xsl:template>

  <xsl:template match="movie">
    <tr>
      <td><xsl:value-of select="title"/></td>
      <td>
        <xsl:value-of select="bond"/>
      </td>
      <td>
        <xsl:value-of select="duration"/>
      </td>
    </tr>
  </xsl:template>
</xsl:stylesheet>
```

Exercise

2. Write a stylesheet to transform the XML file bond_movies.xml into a HTML file similar to the one displayed on the right.



James Bond Movies

Title	Actor	Duration
Dr. No	Sean Connery	105
From russia with love	Sean Connery	110
Goldfinger	Sean Connery	106
Thunderball	Sean Connery	125

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

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template match="/">
  <html>
    <h1>James Bond Movies</h1>
    <table border="1">
              <th>Title</th>
        <th>Actor</th>
        <th>Duration</th>
        <xsl:apply-templates select="bond_movies/movie" />
    </table>
  </html>
</xsl:template>

<xsl:template match="movie">
<xsl:if test="position() mod 2 = 0">
 <tr style="background:#BEF781">
    <td><xsl:value-of select="title"/></td>
    <td><xsl:value-of select="bond"/></td>
    <td><xsl:value-of select="duration"/></td>
  </tr>
</xsl:if>
<xsl:if test="position() mod 2= 1">
 <tr style="background:#F5BCA9">
    <td><xsl:value-of select="title"/></td>
    <td><xsl:value-of select="bond"/></td>
    <td><xsl:value-of select="duration"/></td>
  </tr>
</xsl:if>
</xsl:template>

</xsl:stylesheet>
```

Exercise: Sorted XHTML Tables

1. Produce an XHTML page with the following tables:
   1. Movies with Sean Connery in alphabetical order by title
   2. Movies with Roger Moore in numerical order by year
   3. Movies with Timothy Dalton in numerical order by duration
   Movies with Pierce Brosnan in alphabetical order by name of Bond girl in descending order
   5. Movies with Daniel Craig in alphabetical order by name of Bond girl in ascending order

   The table for the second last could look like as follows:

Movies with Pierce Brosnan as main character in alphabetical order (by name of Bond girl in descending order):

Title	Bond Actor	Bond Girl	Producer	Year	Length
Tomorrow never dies	Pierce Brosnan	Michelle Yeoh	Roger Spottiswoode	1997	114 min
Goldeneye	Pierce Brosnan	Izabella Scorupco	Martin Campbell	1995	125 min
Die another day	Pierce Brosnan	Halle Berry	Lee Tamahori	2002	133 min
The world is not enough	Pierce Brosnan	Denise Richards	Michael Apted	1999	122 min

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

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns="http://www.w3.org/1999/xhtml">
<xsl:output doctype-public="-//W3C//DTD XHTML 1.0 Strict//EN" doctype-
system="http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"/>

<xsl:template match="bond_movies">
   <html>
      <head>
         <title>James Bond Movie Catalogue</title>
      </head>
      <body>

         <!-- Title and subtitles  -->
         <h1>James Bond Movie Catalog</h1>
         <div>
            <i>This is a list of all James Bond movies between 1962 and
               <xsl:value-of select="@year" />.</i>
            <br /><br />
            <b>Movies with Sean Connery as main character in alphabetical orde
               (by title):</b>
            <br/><br/>
         </div>

         <table border="1" cellpadding="10'">
            <tr style="background-color:#990066">
               <th>Title</th>
               <th>Bond Actor</th>
               <th>Bond Girl</th>
               <th>Producer</th>
               <th>Year</th>
               <th>Length</th>
            </tr>
            <!-- Add only movies from Sean Connery (sorted by title) -->
            <xsl:apply-templates select="movie[starts-with(bond/text(), 'Sean')]">
            <xsl:sort select="title" data-type="text"/>
            </xsl:apply-templates>
         </table>
```

```
<div>
   <br/><br/>
   <b>Movies with Roger Moore as main character in numerical order (by year):</b>
   <br/><br/>
</div>
<table border="1" cellpadding="10'">
   <tr style="background-color:#6666FF">
      <th>Title</th>
      <th>Bond Actor</th>
      <th>Bond Girl</th>
      <th>Producer</th>
      <th>Year</th>
      <th>Length</th>
   </tr>
   <!-- Add only movies from Roger Moore (sorted by year) -->
   <xsl:apply-templates select="movie[starts-with(bond/text(), 'Roger')]">
      <xsl:sort select="Year" data-type="number"/>
   </xsl:apply-templates>
</table>

<div>
   <br/><br/>
   <b>Movies with Timothy Dalton as main character in numerical order
      (by duration):</b>
   <br/><br/>
</div>
<table border="1" cellpadding="10'">
   <tr style="background-color:#9acd32">
      <th>Title</th>
      <th>Bond Actor</th>
      <th>Bond Girl</th>
      <th>Producer</th>
      <th>Year</th>
      <th>Length</th>
   </tr>
   <!-- Add only movies from Timothy Dalton (sorted by duration) -->
   <xsl:apply-templates select="movie[starts-with(bond/text(), 'Tim')]">
      <xsl:sort select="duration" data-type="number"/>
   </xsl:apply-templates>
</table>

<div>
   <br/><br/>
   <b>Movies with Pierce Brosnan as main character in alphabetical order
      (by name of Bond girl in descending order):</b>
   <br/><br/>
</div>
<table border="1" cellpadding="10'">
   <tr style="background-color:#CC66FF">
      <th>Title</th>
      <th>Bond Actor</th>
      <th>Bond Girl</th>
      <th>Producer</th>
      <th>Year</th>
      <th>Length</th>
   </tr>
   <xsl:apply-templates select="movie[starts-with(bond/text(), 'Pierce')]">
      <xsl:sort select="bond_girl" data-type="text" order="descending"/>
   </xsl:apply-templates>
</table>
```

```
         <div>
            <br/><br/>
            <b>Movies with Daniel Craig as main character in alphabetical order
               (by name of Bond girl in ascending order):</b>
            <br/><br/>
         </div>
         <table border="1" cellpadding="10'">
            <tr style="background-color:#F2F5A9">
               <th>Title</th>
               <th>Bond Actor</th>
               <th>Bond Girl</th>
               <th>Producer</th>
               <th>Year</th>
               <th>Length</th>
            </tr>
            <!-- Add only movies from Daniel Craig (sorted by name of Bond girl) -->
            <xsl:apply-templates select="movie[starts-with(bond/text(), 'Daniel')]">
               <xsl:sort select="bond_girl" data-type="text" order="ascending"/>
            </xsl:apply-templates>
         </table>

      </body>
   </html>
</xsl:template>

<xsl:template match="movie">
               <tr>
                  <td align="left">
                     <xsl:value-of select="title" />
                  </td>
                  <td align="left">
                     <xsl:value-of select="bond" />
                  </td>
                  <td align="left">
                     <xsl:value-of select="bond_girl" />
                  </td>
                  <td align="left">
                     <xsl:value-of select="regie" />
                  </td>
                  <td align="left">
                     <xsl:value-of select="year" />
                  </td>
                  <td align="center">
                     <xsl:choose>
                      <xsl:when test="duration">
                             <xsl:value-of select="duration" /> min
                      </xsl:when>
                      <xsl:otherwise>
                             <i>unknown</i>
                      </xsl:otherwise>
                     </xsl:choose>
                  </td>
               </tr>
</xsl:template>

</xsl:stylesheet>
```

Control Questions

1. When do you choose XSLT technology, i.e. what can it do for you ?

   Eine XML-Datei in ein anderes Format umwandeln, z.B. auch wieder XML oder XHTML, SVG/SMIL, FO (PDF, PNG, etc)

2. What constraints exist with respect to input and output language ?

   Input muss XML sein (weil die Matches mit XPath definiert sind), Output kann (muss aber nicht) XML sein.

3. What happens if I drag an XML document with an XSLT processing instruction to a browser window ?

   Der XSLT Processor wird ausgeführt, das XML wird zusammen mit dem XSLT ausgeführt ☺

   Der Browser lädt das XML und XSLT, er lädt das XML ins Ram (InMemory-Darstellung) und parst es mit den Templates. Sofern er keines findet nimmt er das interne default-template.

4. How does conflict resolution and built-in templates work ?

   Import: import hat die kleinere Priorität

   Template: dasjenige welches am besten zutrifft, falls zwei gleich sind wird das zuletzt definierte verwendet.

   Built-In Template kommen zum Einsatz wenn sonst keines gefunden wird

```
<xsl:template match="/">
   <xsl:apply-templates select="*">
</xsl:template>
```

5. Explain the template context.

   Der Context ist dasjenige Element worauf das Template aufgerufen wird.

   Der Context ist eine Liste von Knoten, welches aktuell vom Template bearbeitet wird.

   Innerhalb vom Context kann nicht „zurück“ gesprungen werden, es kann nun nach „vorne“ gehen.

6. Compare pull and push processing of XML elements.

   Template: Push

   For-Each: Pull

   Using <for-each> is called pull-processing, using templates is called push-processing.

   We can push the elements to a template or pull them out right now for processing.

   - *Templates are more flexibel and easier to maintain.*
   - *Templates can be shared between stylesheets (see later).*
   - *Templates can be overwritten (see later).*
   - *Templates enable modular code.*

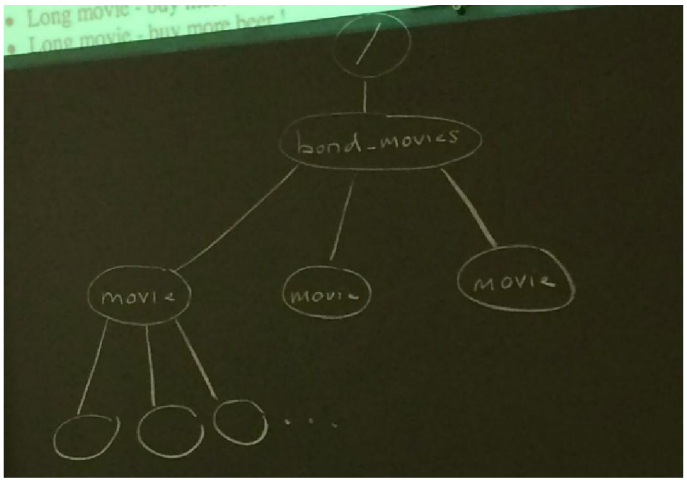
Exam Preparation Exercise 1a

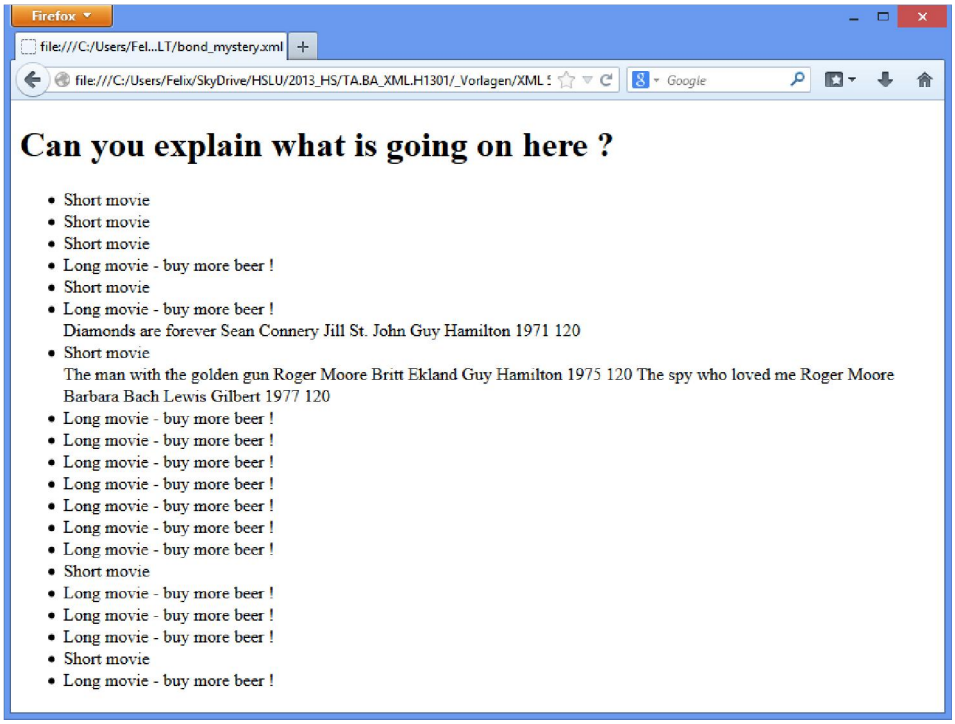
1. Make a prediction of the output of the following style sheet and verify.

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <html>
      <body>
        <h1>Can you explain what is going on here ?</h1>
        <ul><xsl:apply-templates /></ul>
      </body>
    </html>
  </xsl:template>

  <xsl:template match="movie[duration &gt; 120]">
    <li><xsl:text>Long movie - buy more beer !</xsl:text></li>
  </xsl:template>

  <xsl:template match="movie[duration &lt; 120]">
    <li><xsl:text>Short movie</xsl:text></li>
  </xsl:template>

</xsl:stylesheet>
```

Apply-template: ohne match → Context bleibt bei „/“

Nur Template für duration > 120 und duration < 120; jedoch keines für duration = 120 → weiter gehen nach unten (Titel, Bond, etc.) dafür gibt es kein Template → Default-Template





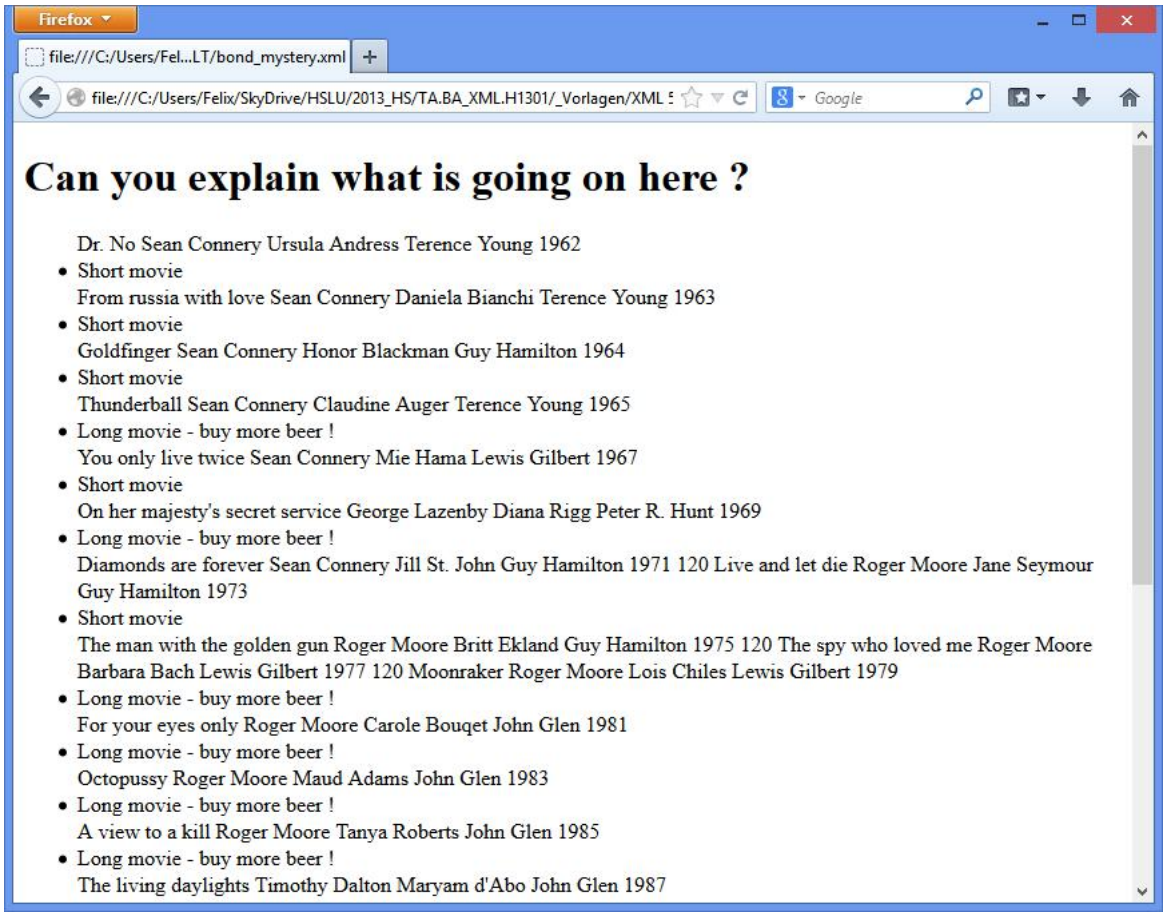
Can you explain what is going on here ?

- Short movie
- Short movie
- Short movie
- Long movie - buy more beer !
- Short movie
- Long movie - buy more beer !
  Diamonds are forever Sean Connery Jill St. John Guy Hamilton 1971 120
- Short movie
  The man with the golden gun Roger Moore Britt Ekland Guy Hamilton 1975 120 The spy who loved me Roger Moore Barbara Bach Lewis Gilbert 1977 120
- Long movie - buy more beer !
- Long movie - buy more beer !
- Long movie - buy more beer !
- Long movie - buy more beer !
- Long movie - buy more beer !
- Long movie - buy more beer !
- Long movie - buy more beer !
- Short movie
- Long movie - buy more beer !
- Long movie - buy more beer !
- Long movie - buy more beer !
- Short movie
- Long movie - buy more beer !

Exam Preparation Exercise 1b

1. Make a prediction of the output of the following style sheet and verify.

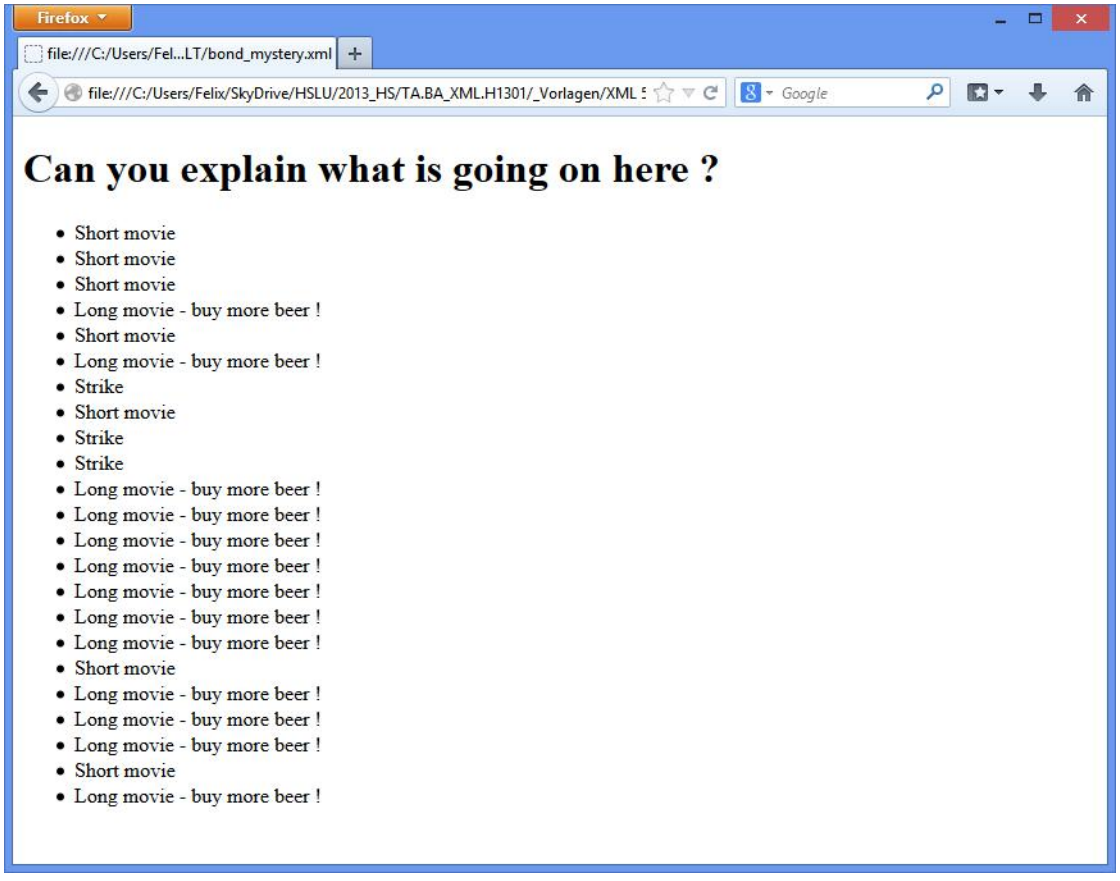
```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <html>
      <body>
        <h1>Can you explain what is going on here ?</h1>
        <ul><xsl:apply-templates /></ul>
      </body>
    </html>
  </xsl:template>

  <xsl:template match="movie/duration[text() &gt; 120]">
    <li><xsl:text>Long movie - buy more beer !</xsl:text></li>
  </xsl:template>

  <xsl:template match="movie/duration[text() &lt; 120]">
    <li><xsl:text>Short movie</xsl:text></li>
  </xsl:template>

</xsl:stylesheet>
```



Exam Preparation Exercise 1b

1. Make a prediction of the output of the following style sheet and verify.

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <html>
      <body>
        <h1>Can you explain what is going on here ?</h1>
        <ul><xsl:apply-templates /></ul>
      </body>
    </html>
  </xsl:template>

  <xsl:template match="movie[duration &gt; 120]">
    <li><xsl:text>Long movie - buy more beer !</xsl:text></li>
  </xsl:template>

  <xsl:template match="movie[duration &lt; 120]">
    <li><xsl:text>Short movie</xsl:text></li>
  </xsl:template>

  <xsl:template match="movie[duration = 120]">
    <li><xsl:text>Strike</xsl:text></li>
  </xsl:template>

</xsl:stylesheet>
```



Exam Preparation Exercise 2

1. Produce a valid XHTML page similar to the one on the right. Countries are sorted after their population. The last row contains the sum of all column entries. The trend is (economically) positive is there are more younger than older people. The absolute number of young and old people is calculated from the % data in the XML document.

Statistics of Population

	Population	Population < 15 in %	Population < 15 abs	Population > 64 in %	Population > 64 abs	Trend
Iceland	276000	23.5 %	64860	11.6 %	32016	+
Luxembourg	429000	18.8 %	80652	14.3 %	61347	+
Ireland	3735000	22.2 %	829170	11.3 %	422055	+
Norway	4445000	19.5 %	866775	15.5 %	688975	+
Finland	5160000	18.4 %	949440	14.7 %	758520	+
Denmark	5314000	18.2 %	967148	14.9 %	791786	+
Switzerland	7124000	17.5 %	1246700	15.2 %	1082848	+
Austria	8083000	17.0 %	1374110	15.5 %	1252865	+
Sweden	8854000	18.6 %	1646844	17.4 %	1540596	+
Portugal	9980000	16.9 %	1686620	15.2 %	1516960	+
Belgium	10214000	17.7 %	1807878	16.6 %	1695524	+
Greece	10522000	15.4 %	1620388	16.9 %	1778218	-
Netherlands	15760000	18.5 %	2915600	13.5 %	2127600	+
Spain	39394000	15.3 %	6027282	16.4 %	6460616	-
Italy	57613000	14.5 %	8353885	17.7 %	10197501	-
France	58973000	19.0 %	11204870	15.8 %	9317734	+
United Kingdom	59623000	19.0 %	11328370	15.6 %	9301188	+
Germany	82037000	15.8 %	12961846	15.9 %	13043883	-
Total:	387536000		65932438		62070232	+

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

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output doctype-public="-//W3C//DTD XHTML 1.0 Strict//EN"
              doctype-system="http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"/>

  <xsl:template match="european_countries">
    <html>
      <head>
        <title>European Countries</title>
      </head>
      <body>
        <h1 style="color:#9900CC">Statistics of Population</h1>
        <table border="1" cellpadding="5" >
          <tr style="background-color:#CCCCCC">
            <th></th>
            <th align="center">Population</th>
            <th align="center">
              Population<br />&lt; 15 in %
            </th>
            <th align="center">
              Population<br />&lt; 15 abs
            </th>
            <th align="center">
              Population<br />&gt; 64 in %
            </th>
            <th align="center">
              Population<br />&gt; 64 abs
            </th>
            <th align="center">Trend</th>
          </tr>

          <!-- Add table entries  -->
          <xsl:apply-templates select="country">
            <xsl:sort select="population" data-type="number"/>
          </xsl:apply-templates>
```

```
        <!-- Last table row - calculate sum of columns  -->
        <tr>
          <td align="center" style="background-color:#CCCCCC">
            <b>Total:</b>
          </td>
          <td align="right" style="color:blue">
            <xsl:value-of select="sum(country/population)" />
          </td>
          <td></td>

          <xsl:variable name="sum_u15">
            <xsl:call-template name="calculate_sum_u15">
              <xsl:with-param name="index" select="'1'"/>
              <xsl:with-param name="runningSum" select="'0'" />
              <xsl:with-param name="base" select="country" />
            </xsl:call-template>
          </xsl:variable>

          <td align="right" style="color:blue">
            <xsl:value-of select="$sum_u15"/>
          </td>
          <td></td>

          <xsl:variable name="sum_o64">
            <xsl:call-template name="calculate_sum_o64">
              <xsl:with-param name="index" select="'1'"/>
              <xsl:with-param name="runningSum2" select="'0'" />
              <xsl:with-param name="base" select="country" />
            </xsl:call-template>
          </xsl:variable>

          <td align="right" style="color:blue">
            <xsl:value-of select="$sum_o64"/>
          </td>

          <xsl:choose>
            <xsl:when test="$sum_u15 &gt; $sum_o64">
              <td align="center" style="color:green">
                <b>+</b>
              </td>
            </xsl:when>
            <xsl:otherwise>
              <td align="center" style="color:red">
                <b>-</b>
              </td>
            </xsl:otherwise>
          </xsl:choose>

        </tr>
      </table>
    </body>
  </html>
</xsl:template>

<!-- Template to add row to first table  -->
<xsl:template match="country">
  <tr>
    <td align="center" style="background-color:#CCCCCC">
      <b>
        <xsl:value-of select="name"></xsl:value-of>
      </b>
    </td>
    <td align="right">
      <xsl:value-of select="population"></xsl:value-of>
    </td>
    <td align="right">
      <xsl:value-of select="format-number(population_under_15, '00.0')"></xsl:value-of> %
    </td>
    <td align="right">
      <xsl:value-of select="round(population * population_under_15 div 100)"></xsl:value-of>
    </td>
    <td align="right">
      <xsl:value-of select="format-number(population_over_64, '00.0')"></xsl:value-of> %
    </td>
    <td align="right">
      <xsl:value-of select="round(population * population_over_64 div 100)"></xsl:value-of>
    </td>
```

```
      <xsl:choose>
        <xsl:when test="population_under_15 &gt; population_over_64">
          <td align="center" style="color:green">
            <b>+</b>
          </td>
        </xsl:when>
        <xsl:otherwise>
          <td align="center" style="color:red">
            <b>-</b>
          </td>
        </xsl:otherwise>
      </xsl:choose>
    </tr>
  </xsl:template>

  <!-- Templates to calculate sum of columns  -->
  <xsl:template name="calculate_sum_u15">
    <xsl:param name="index" />
    <xsl:param name="runningSum" />
    <xsl:param name="base" />
    <xsl:variable name="currentItem">
      <xsl:value-of select="round($base[$index]/population_under_15
                                  * $base[$index]/population div 100)" />
    </xsl:variable>
    <xsl:variable name="remainingItems">
      <xsl:choose>
        <xsl:when test="$index = count($base)">
          <xsl:text>0</xsl:text>
        </xsl:when>
        <xsl:otherwise>
          <xsl:call-template name="calculate_sum_u15">
            <xsl:with-param name="index" select="$index + 1" />
            <xsl:with-param name="runningSum" select="$runningSum + $currentItem"/>
            <xsl:with-param name="base" select="$base" />
          </xsl:call-template>
        </xsl:otherwise>
      </xsl:choose>
    </xsl:variable>
    <xsl:value-of select="$currentItem + $remainingItems" />
  </xsl:template>

  <xsl:template name="calculate_sum_o64">
    <xsl:param name="index" />
    <xsl:param name="runningSum2" />
    <xsl:param name="base" />
    <xsl:variable name="currentItem">
      <xsl:value-of select="round($base[$index]/population_over_64
                                  * $base[$index]/population div 100)" />
    </xsl:variable>
    <xsl:variable name="remainingItems">
      <xsl:choose>
        <xsl:when test="$index = count($base)">
          <xsl:text>0</xsl:text>
        </xsl:when>
        <xsl:otherwise>
          <xsl:call-template name="calculate_sum_o64">
            <xsl:with-param name="index" select="$index + 1" />
            <xsl:with-param name="runningSum2" select="$runningSum2 + $currentItem"/>
            <xsl:with-param name="base" select="$base" />
          </xsl:call-template>
        </xsl:otherwise>
      </xsl:choose>
    </xsl:variable>
    <xsl:value-of select="$currentItem + $remainingItems" />
  </xsl:template>

</xsl:stylesheet>
```