Chương 4 : Biến đổi tài liệu XML với XSLT

I. Mở đầu về XSLT

Khái niệm về chương trình XSLT :

- Một loại tài liệu XML đặc biệt bao gồm các thẻ xử lý cho phép biến đổi một tài liệu XML thành một tài liệu văn bản bất kỳ
- Một loại chương trình thông dịch đặc biệt với
  + Dữ liệu nguồn : Tài liệu XML
  + Kết xuất : Tài liệu dạng văn bản

Tài liệu XML ---- > Chương trình XSLT ---- > Tài liệu văn bản

Các ứng dụng chính :

XSLT có 2 ứng dụng chính hiện nay:

1. Thực hiện biến đổi từ tập tin XML vào trang Web với ngôn ngữ HTML
2. Thực hiện biến đổi từ tập tin XML vào tập tin XML khác

- **Xml --- > Html:**

Chp phép thể hiện nội dung tập tin Xml trên trang Web

Ví dụ :

Tập tin Xml Don_thuc.xml

<DON_THUC He_so="4" So_mu="6" />

thông qua xử lý của chương trình Xuat_don_thuc.Xslt sẽ được thể hiện trên trang Web

4x6

- **Xml -- > Xml:**

Cho phép tạo tập tin Xml mới từ tập tin Xml hiện có để có thể trích rút thông tin, tái cấu trúc các thẻ, v.v...

Ví dụ :

Tập tin Xml Don_thuc.xml

<DON_THUC He_so="4" So_mu="6" />

thông qua xử lý của một chương trình Xslt sẽ tạo ra tập tin Don_thuc_1.xml như sau

```
<DON_THUC>
        <He_so> 4 </He_so>
        <So_mu> 6 </So_mu>
</DON_THUC>
```

1) Cấu trúc chương trình XSLT

Cấu trúc chương trình XSLT đơn giản

```
<?xml version="1.0"  encoding="UTF-8" ?>
<xsl:stylesheet version="1.0"  xmlns:xsl=http://www.w3.org/1999/XSL/Transform>
```

```
<xsl:template match="/" >
    Các lệnh (thẻ) xử lý cho phép trích rút thông tin từ Tập tin Xml nguồn và kết xuất vào tập tin
    kết quả
</xsl:template>
</xsl:stylesheet>
```

Ví dụ:

Chương trình sau cho phép biến đổi tập tin Nguoi_dung.xml

```
<NGUOI_DUNG  Ho_ten="Trần Văn Long" />
```

để tạo tập tin văn bản với nội dung

Xin chào Trần Văn Long. Đây là chương trình XSLT đầu tiên của tôi

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output method="text"/>
  <xsl:template match="/" >
   Xin chào <xsl:value-of select="/NGUOI_DUNG/@Ho_ten"/>
   . Đây là chương trình XSLT đầu tiên của tôi
  </xsl:template>
</xsl:stylesheet>
```

Ví dụ 2 : Chương trình sau cho phép biến đổi tập tin xml

```
<GOC>
   <SO Gia_tri="5" />
   <SO Gia_tri="7" />
 </GOC>
```

để tạo tập tin văn bản với nội dung

5+7 =12

```
<?xmlversion="1.0"encoding="UTF-8" ?>
<xsl:stylesheetversion="1.0"xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:outputmethod ="text"/>
 <xsl:template match="/" >
   <xsl:value-of select="/GOC/SO[1]/@Gia_tri"/> +
   <xsl:value-of select="/GOC/SO[2]/@Gia_tri"/> =
   <xsl:value-of select="/GOC/SO[1]/@Gia_tri + /GOC/SO[2]/@Gia_tri" />
 </xsl:template>
</xsl:stylesheet>
```

2) Cho thực hiện chương trình XSLT

Quá trình thực hiện bao gồm 3 bước:

- Bước 1 : Chuẩn bị dữ liệu nguồn là tập tin XML
- Bước 2 : Soạn thảo chương trình XSLT
- Bước 3 : Cho thực hiện chương trình

- **Bước 1** : Dữ liệu nguồn có thể được chuẩn bị thông qua một trong các cách sau
  - Cách 1 : Sử dụng trình soạn thảo văn bản bất kỳ (vì tài liệu XML chỉ là một văn bản)
  - Cách 2 : Sử dụng trình soạn thảo XML Editor
- **Bước 2** : Chương trình XSLT có thể được chuẩn bị thông qua một trong các cách sau
  - Cách 1 : Sử dụng trình soạn thảo văn bản bất kỳ (vì tài liệu XML chỉ là một văn bản)
  - Cách 2 : Sử dụng trình soạn thảo XML Editor
  - Cách 3 : Sử dụng trình soạn thảo chương trình XSLT (XSLT Editor)
- **Bước 3 :** Tùy theo mục tiêu của việc thực hiện có thể tiến hành một trong 3 cách sau

Cách 1 : Sử dụng môi trường lập trình

Cho thực hiện trực tiếp bên trong môi trường lập trình

Cách này thích hợp cho việc học tập và thử nghiệm chương trình XSLT

Cách 2 : Sử dụng trình duyệt Web

Cho thực hiện trực tiếp với sự hổ trợ của trình duyệt Web

Cách này cho phép ứng dụng trực tiếp XSLT trong việc thể hiện thông tin trên Web

Cách 3 : Tự viết chương trình

Cho thực hiện thông qua việc viết một ứng dụng trong ngôn ngữ lập trình khác(ví dụ C#). Ứng dụng này sẽ

- Nạp/đọc chương trình XSLT vào bộ nhớ
- Chuẩn bị dữ liệu nguồn (nếu cần thiết)
- Cho thực hiện
- Xử lý kết xuất được tạo ra (nếu cần thiết)

Cách này thích hợp khi cần "nhúng" chương trình XSLT vào một ứng dụng để có thể thực hiện nhanh, dễ bảo trì, chuẩn một số xử lý biến đổi nào đó liên quan tài liệu XML

a) Sử dụng môi trường lập trình

Với môi trường lập trình Visual Studio.NET

Bước 1 : Tạo tập tin Xml nguồn

Chọn Project - Add New Item với loại tập tin là Xml

=== > Cửa sổ cho phép soạn thảo tập tin Xml

Bước 2 : Tạo chương trính XSLT

Chọn Project - Add New Item với loại tập tin Xslt

=== > Cửa sổ cho phép soạn thảo chương trình XSLT

Bước 3 : Cho thực hiện

3.1 Bước 3.1 : Chọn của sổ Properties để xác định tập tin Xml nguồn và tập tin kết xuất

3.2 Bước 3.2 : Quay về cửa sổ soạn thảo chương trình XSLT (Click và cửa sổ) và sau đó chọn chức năng Xml ---> Debug XSLT

Lưu ý : Bước 3.1 Chỉ cần thực hiện một lần nếu không thay đổi tập tin nguồn

Có thể đánh dấu điểm ngắt bên trong chương trình XSLT tương tự khi Debug ứng dụng với ngôn ngữ lập trình khác

b) Sử dụng trình duyệt Web

Bước 1 : Tạo tập tin Xml nguồn với chỉ thị yêu cầu thực hiện chương trình XSL

<?xml-stylesheet type="text/xsl" href=Chuỗi đường dẫn đến tập tin chương trình XSLT ?>

Ví dụ :

<?xmlversion="1.0"encoding="utf-8" ?>

<?xml-stylesheettype="text/xsl" href="Xuat_loi_chao.xslt" ?>

<NGUOI_DUNGHo_ten="Trần văn Long" />

Bước 2 : Tạo chương trình XSLT

Chọn Project - Add New Item với loại tập tin Xslt

=== > Cửa sổ cho phép soạn thảo chương trình XSLT

Bước 3 : Cho thực hiện

Mở trình duyệt Web và sau đó chọn URL là đường dẫn đến tập tin Xml

c) Tự viết chương trình

Bước 1 : Tạo tập tin Xml nguồn

Bước 2 : Tạo chương trình XSLT Chọn Project - Add New Item với loại tập tin Xslt

=== > Cửa sổ cho phép soạn thảo chương trình XSLT

Bước 3 : Cho thực hiện

....

Khai báo đối tượng Bo_thuc_hien

đọc tập tin chương trình XSL vào Bo_thuc_hien

Yêu cầu Bo_thuc_hien thực hiện chương trình XSLT với dữ liệu nguồn và kết xuất

....

Ví dụ : với Visual Studio.NET 2008 C#, đoạn chương trình sau sẽ cho thực hiện chương trình Xuat_loi_chao.Xslt

- Dữ liệu nguồn là tập tin Nguoi_dung.xml
- Kết xuất là tập tin văn bản Loi_chao.txt

(Tất cả các tập tin đều đặt trong thư mục của Project)

```
using System.Xml;
using System.Xml.Xsl;
class Program
static void Main()
{
          string Duong_dan_Xml = "../../Nguoi_dung.xml";
          string Duong_dan_Xslt= "../../Xuat_loi_chao.xslt" ;
          string Duong_dan_Kq = "../../Loi_chao.txt";
          XslCompiledTransform Thuc_hien = new XslCompiledTransform(true);
          Thuc_hien.Load(Duong_dan_Xslt);
          Thuc_hien.Transform(Duong_dan_Xml, Duong_dan_Kq);
}
```

3) Các ví dụ minh họa

Mục tiêu :

Minh họa trực quan một số chương trình XSLT.

Các chương trình này sẽ được diễn giải chi tiết về sau trong các mục khác

=== > Chưa yêu cầu hiểu ý nghĩa các lệnh

=== > Sử dụng để rèn luyện cách cho thực hiện chương trình XSLT

*** Xuất cây trường - khối - lớp**

Với tập tin Truong.xml có nội dung như sau

```
<?xml version="1.0" encoding="utf-8" ?>
<TRUONG  Ten="Trường cấp 3 XXX">
 <KHOI Ten="Khối 10" >
   <LOP Ten="Lớp 10A" />
   <LOP Ten="Lớp 10B" />
   <LOP Ten="Lớp 10C" />
   <LOP Ten="Lớp 10D" />
 </KHOI>
 <KHOI Ten="Khối 11" >
   <LOP Ten="Lớp 11A" />
   <LOP Ten="Lớp 11B" />
   <LOP Ten="Lớp 11C" />
 </KHOI>
 <KHOI Ten="Khối 12" >
   <LOP Ten="Lớp 12A" />
   <LOP Ten="Lớp 12B" />
   <LOP Ten="Lớp 12C" />
 </KHOI>
</TRUONG>
```

Chương trình Xuat_truong.xslt sau sẽ kết xuất (dạng Html) các thông tin về trường (bao gồm thông tin khối, lớp)

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet version="1.0"  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output method ="html" />
 <xsl:template match="/" >
   <xsl:apply-templates select="TRUONG" />
    Danh sách các khối lớp <br />
   <xsl:apply-templates select="KHOI" />
 </xsl:template>
 <xsl:template match="KHOI">
   <xsl:value-of select="@Ten"/>
```

```
    <br />
    <xsl:apply-templates select="LOP" />
 </xsl:template>
   <xsl:template match="LOP">
     <xsl:value-of select="@Ten"/>
     <br />
 </xsl:template>
</xsl:stylesheet>
```

Ghi chú :

Thuộc tính select trong thẻ xsl:apply-templates có thể được bỏ qua và khi đó sẽ được hiểu là select="*" (cho lượng giá là các nút con của nút ngữ cảnh)

=== > Một trong các cách đơn giản tổ chức chương trình Xslt là tổ chức chương trình theo các loại thẻ có trong tập tin Xml và gọi thực hiện (so khớp) không cần tham số

Gọi thực hiện : <xsl:apply-templates />

Khai báo hàm/mẩu so khớp :

```
<xsl:template  match="tên loại thẻ" >
      Các thẻ xử lý
</xsl:template>
```

```
<xsl:stylesheet  version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output method ="html" />
 <xsl:template  match="/" >
    <xsl:apply-templates  />
 </xsl:template>
 <xsl:template  match="TRUONG">
    <xsl:value-of  select="@Ten"/>
    <br />
    Danh sách các khối lớp <br />
    <xsl:apply-templates  />
 </xsl:template>
 <xsl:template  match="KHOI">
    <xsl:value-of  select="@Ten"/>
    <br />
    <xsl:apply-templates  />
 </xsl:template>
 <xsl:templatematch="LOP">
    <xsl:value-of  select="@Ten"/>
    <br />
 </xsl:template>
</xsl:stylesheet>
```

*** Xuất danh sách chọn**

Chương trình Xslt sau sẽ xuất danh sách chọn các đơn vị từ tập tin Cong_ty.xml

```
<?xml version="1.0"encoding="UTF-8" ?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output  method ="html" />
 <xsl:template  match="/" >
   <html>
        <body>
            <xsl:apply-templates />
        </body>
   </html>
 </xsl:template>
 <xsl:template  match="CONG_TY" >
   Danh sách đơn vị :
       <select>
          <xsl:apply-templates />
       </select>
 </xsl:template>
 <xsl:template  match="DON_VI" >
       <option>
         <xsl:value-of  select ="@Ten"/>
       </option>
 </xsl:template>
</xsl:stylesheet>
```

*** Sắp xếp kết quả thi đấu Olympic**

Với tập tin Xml Ket_qua_Olympic.xml

```
<?xml version="1.0" encoding="utf-8" ?>
<KET_QUA>
  <QUOC_GIA Ten="AAA" So_vang="10" So_bac="7" So_dong="2" />
  <QUOC_GIA Ten="XXX" So_vang="6" So_bac="0" So_dong="12" />
  <QUOC_GIA Ten="BBB" So_vang="10" So_bac="8" So_dong="13" />
  <QUOC_GIA Ten="DDD" So_vang="4" So_bac="17" So_dong="0" />
  <QUOC_GIA Ten="MMM" So_vang="6" So_bac="1" So_dong="0" />
  <QUOC_GIA Ten="KKK" So_vang="6" So_bac="0" So_dong="2" />
  <QUOC_GIA Ten="LLL" So_vang="10" So_bac="4" So_dong="23" />
  <QUOC_GIA Ten="PPP" So_vang="3" So_bac="27" So_dong="100" />
</KET_QUA>
```

Đoạn chương trình XSL sau sắp xếp các quốc gia giảm dần theo thứ tự ưu tiên

- Ưu tiên 1 : Số huy chương vàng
- Ưu tiên 2 : Số huy chương bạc
- Ưu tiên 3 : Số huy chương đồng

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet  version="1.0"  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output  method ="xml"  indent="yes" />
```

```
<xsl:template match="/" >
  <xsl:apply-templates />
</xsl:template>
<xsl:template  match="KET_QUA"  >
  <xsl:copy>
    <xsl:apply-templates  select ="QUOC_GIA">
      <xsl:sort order="descending" data-type="number" select="@So_vang" />
      <xsl:sort order="descending" data-type="number" select="@So_bac" />
      <xsl:sort order="descending" data-type="number" select="@So_dong" />
    </xsl:apply-templates>
  </xsl:copy>
</xsl:template>
<xsl:template match="QUOC_GIA" >
  <!--<xsl:copy-of  select="."/>-->
  <xsl:copy >
    <xsl:copy-of  select ="@*"/>
  </xsl:copy>
  </xsl:template>
</xsl:stylesheet>
```

II. Các thao tác cơ bản

Mục tiêu : Trình bày các kỹ thuật xử lý cơ bản khi xây dựng chương trình XSLT

Nội dung :

1. Trích rút thông tin và kết xuất với thẻ xử lý xsl:value-of , xsl:variable
2. Xử lý rẽ nhánh với xsl:if , xsl:choose
3. Xử lý lặp với xsl:for-each. Xử lý so khớp với xsl:apply-templates , xsl:template

1) Trích rút và kết xuất thông tin

Vấn đề : Cần trích một số thông tin trong tập tin Xml nguồn và đưa vào tập tin kết xuất

Hướng giải quyết :

Cách 1 : Trích rút thông tin từ tập tin Xml và sau đó kết xuất trực tiếp với thẻ xử lý xsl:value -of

Cách 2 : Trích rút thông tin vào biến với thẻ xử lý xsl:variable và sau đó sử dụng biến này trong thẻ xử lý xsl:value-of

a) Thẻ xsl:value-of

Ý nghĩa :

Cho phép trích rút thông tin từ tập tin Xml hay từ giá trị của biến và sau đó đưa vào tập tin kết quả

Cú pháp :

Nếu trích rút thông tin từ tập tin Xml nguồn

<xsl:value-of select="Biểu thức Xpath" />

Nếu trích rút thông tin từ biến

<xsl:value-of select="$Ten_bien" />

b) Thẻ xsl:variable

Ý nghĩa :

Cho phép trích rút thông tin từ tập tin Xml và đưa vào một biến (đúng ra là hằng vì nội dung biến này không thể thay đổi được)

Cú pháp :

<xsl:variable name="Ten_bien" select="Biểu thức Xpath" />

Ví dụ : Chương trình tính tổng 2 số nguyên có thể thực hiện theo 2 cách

Cách 1 : Trích rút thông tin trực tiếp

```
<?xml  version="1.0"encoding="UTF-8" ?>
<xsl:stylesheet  version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output  method ="text"/>
 <xsl:template  match="/" >
    <xsl:value-of select="/GOC/SO[1]/@Gia_tri"/> +
    <xsl:value-of select="/GOC/SO[2]/@Gia_tri"/> =
    <xsl:value-of select="/GOC/SO[1]/@Gia_tri + /GOC/SO[2]/@Gia_tri" />
 </xsl:template>
 </xsl:template>
</xsl:stylesheet>
```

Cách 2 : Thông qua các biến

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet  version="1.0"  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output  method ="text"/>
 <xsl:template match="/" >
    <xsl:variable name="So_1" select="/GOC/SO[1]/@Gia_tri" />
    <xsl:variable name="So_2" select="/GOC/SO[2]/@Gia_tri" />
    <xsl:value-of select="$So_1" /> +
    <xsl:value-of select="$So_2" /> =
    <xsl:value-ofselect="$So_1 + $So_2"/>
 </xsl:template>
</xsl:stylesheet
```

> **Lưu ý :**

- Chỉ số các thẻ của tập tin Xml bắt đầu từ 1
- Biểu thức bên trong thuộc tính select có thể
  + Một biểu thức Xpath duy nhất
  + Một biến duy nhất
  + Biểu thức số học với thành phần là biểu thức Xpath hay biến

điều này cho phép thực hiện một số xử lý trên thông tin nguồn trước khi kết xuất, tuy nhiên các xử lý này khá giới hạn vì Xslt được thiết kế và sử dụng vào mục tiêu chính là biến đổi

2) Rẻ nhánh

Vấn đề :

Cần rẽ nhánh xử lý kết xuất tùy vào điều kiện của tập tin Xml nguồn

Hướng giải quyết :

Cách 1 : Sử dụng thẻ xử lý xsl:if . Cách này cho phép chỉ kết xuất trong trường hợp một điều kiện nào đó được thỏa (và nếu không thỏa thì không kết xuất gì cả)

Cách 2 : Sử dụng thẻ xử lý xsl:choose . Cách này cho phép kết xuất tùy theo nhiều điều kiện với các trường hợp khác nhau

a) Thẻ xsl:if

Ý nghĩa :

Cho phép chỉ thực hiện một số thẻ xử lý khi điều kiện được thỏa

Cú pháp :

```
<xsl:if  test="Biểu thức logic "  >
   Các thẻ xử lý
</xsl:if>
```

Ghi chú : Biểu thức logic bao gồm các biểu thức tính toán (trên chuỗi Xpath hay giá trị biến) cùng với các phép toán quan hệ >, >= , <,<= , = , != và các phép toán logic not , and , or

b) Thẻ xsl:choose

Ý nghĩa :

Tương tự như thẻ xsl:if nhưng cho phép sử dụng nhiều điều kiện khác nhau

Cú pháp :

```
<xsl:choose>
  <xsl:when test="Biểu thức logic 1 " >
     Các thẻ xử lý khi biểu thức logic 1 thỏa
  </xsl:when>
  <xsl:when test="Biểu thức logic 2 " >
     Các thẻ xử lý khi biểu thức logic 2 thỏa
  </xsl:when>
     <xsl:otherwise >
     Các thẻ xử lý khi tất cả các biểu thức logic trên đều không thỏa
  </xsl:when>
</xsl:choose>
```

Ghi chú :

Thẻ xử lý trên có tác dụng tương tự cấu trúc if (
điều kiện 1)

```
  {  ....
  }
```

else if (điều kiện 2)

```
  { ...     }
 ..... else
  { ....
  }
```

Ví dụ :

Chương trình xác định số nguyên lớn nhất có thể thực hiện theo 2 cách

Cách 1 : Sử dụng xsl:if

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output method ="text"    />
 <xsl:template match="/" >
```

```
    <xsl:variable name="So_1" select="/GOC/SO[1]/@Gia_tri" />
    <xsl:variable name="So_2" select="/GOC/SO[2]/@Gia_tri" /> Số lớn
    nhất trong 2 số
    <xsl:value-of select="$So_1"/> và
    <xsl:value-of select="$So_2"/> là
    <xsl:if test="$So_1> $So_2" >
      <xsl:value-of select ="$So_1"/>
    </xsl:if>
    <xsl:if test="$So_1 &lt;=$So_2" >
       <xsl:value-of select ="$So_2"/>
    </xsl:if>
 </xsl:stylesheet>
```

Cách 2 : Sử dụng xsl:choose

```
<?xml  version="1.0"  encoding="UTF-8" ?>
<xsl:stylesheet  version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output  method ="text"    />
 <xsl:template match="/" >
    <xsl:variable name="So_1" select="/GOC/SO[1]/@Gia_tri" />
    <xsl:variable name="So_2" select="/GOC/SO[2]/@Gia_tri" />
     Số lớn nhất trong 2 số
    <xsl:value-of select="$So_1"/> và
    <xsl:value-of select="$So_2"/> là
   <xsl:choose>
     <xsl:when test="$So_1 > $So_2">
       <xsl:value-of  select ="$So_1"/>
     </xsl:when>
     <xsl:otherwise>
      <xsl:value-of select ="$So_2"/>
     </xsl:otherwise>
    </xsl:choose>
  </xsl:template>
 </xsl:stylesheet>
```

3) Vòng lặp

Vấn đề :

Cần lặp lại các xử lý kết xuất trên một tập hợp các nút của tài liệu Xml nguồn

Hướng giải quyết :

Sử dụng thẻ xsl:for-each với biểu thức Xpath tương ứng tập hợp nút cần lặp

***Thẻ xsl:for-each**

Ý nghĩa :

Cho phép lặp lại việc thức hiện các thẻ xử lý trên tập hợp các nút là kết quả của một chuỗi truy vấn Xpath

Cú pháp :

```
<xsl:for-each  select="Biểu thức Xpath "  >
      Các thẻ xử lý
</xsl:for-each>
```

Ghi chú : Các thẻ xử lý bên trong vòng lặp có thể sử dụng biểu thức Xpath theo cách định vị tương đối từ nút ngữ cảnh (nút hiện hành) thay cho sử dụng định vị tuyệt đối

Ví dụ :

Với tập tin nguồn Cong_ty.xml :

```
<?xml version="1.0" encoding="utf-8" ?>
<CONG_TY Ten="Công ty X">
 <DON_VI Ten="đơn vị A" />
 <DON_VI Ten="đơn vị B" />
 <DON_VI Ten="đơn vị C" />
 <DON_VI Ten="đơn vị D" />
</CONG_TY>
```

Chương trình Xuat_cong_ty.xslt sau sẽ kết xuất thông tin về công ty cùng với các đơn vị (theo dạng kết xuất Html)

```
Công ty X
Danh sách các đơn vị
đơn vị A
đơn vị B
đơn vị C
đơn vị D
```

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet version="1.0"  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output method ="html" />
 <xsl:template match="/" >
   <xsl:value-of select="/CONG_TY/@Ten"/>
   <br/>
   Danh sách các đơn vị <br />
   <xsl:for-each  select="/CONG_TY/DON_VI" >
     <xsl:value-of  select="@Ten"/>
    <br />
   </xsl:for-each>
 </xsl:template>
```

</xsl:stylesheet>

4) Hàm

Vấn đề :

Với tập tin Xml có cấu trúc phức tạp hay xử lý kết xuất phức tạp. Việc tổ chức chương trình Xslt chỉ với một thẻ xsl:template duy nhất (tương tự hàm Main duy nhất trong C#)

=== > Chương trình khó viết

=== > Chương trình khó đọc

====> Chương trình khó bảo trì

====> Các đoạn lệnh không tái sử dụng được

Hướng giải quyết :

Tổ chức chương trình thành các phần nhỏ với thẻ xử lý xsl:template (tương tự các hàm tự định nghĩa. Mỗi phần như thế có tên gọi là tập mẫu và đóng vai trò tương tự như hàm trong ngôn ngữ lập trình khác

*Thẻ xsl:template

Ý nghĩa : Cho phép tổ chức chương trình Xslt với các thành phần nhỏ

== > Dễ viết

==> Dễ đọc

== > Dễ bảo trì

===> Tái sử dụng

Cú pháp khai báo

```
<xsl:template match="Biểu thức Xpath">
    Các thẻ xử lý
</xsl:template>
```

Cú pháp "gọi thực hiện"

```
<xsl:apply-templates  select="Biểu thức Xpath" />
```

Cơ chế "gọi thực hiện" (cơ chế so khớp)

Quá trình "gọi thực hiện" (so khớp) của thẻ xử lý xsl:apply-templates như sau

Bước 1 : Lượng giá biểu thức Xpath của thẻ xử lý xsl:apply-templates

Bước 2 : Tìm khai báo xsl:template có thuộc tính match so khớp đúng

Bước 3 : "Gọi thực hiện " nhiều lần các thẻ xử lý bên trong, mỗi lần với một nút ngữ cảnh thuộc danh sách ước lượng của bước 1

Ví dụ 1 : Chương trình xuất thông tin về công ty có thể viết lại như sau

```
<?xmlversion="1.0"encoding="UTF-8" ?>
<xsl:stylesheetversion="1.0"xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:outputmethod ="html" />
 <xsl:template match="/" >
   <xsl:value-of select="/CONG_TY/@Ten"/>
   <br/>
   Danh sách các đơn vị <br />
```

```
   <xsl:apply-templates select="/CONG_TY/DON_VI" />
 </xsl:template>
 <xsl:template match="DON_VI" >
     <xsl:value-of select="@Ten"/>
   <br />
 </xsl:template>
</xsl:stylesheet>
```

Ví dụ 2 : Với tập tin Truong.xml có nội dung như sau

```
<?xml  version="1.0" encoding="utf-8" ?>
<TRUONG Ten="Trường cấp 3 XXX">
 <KHOI  Ten="Khối 10" >
   <LOP Ten="Lớp 10A" />
   <LOP Ten="Lớp 10B" />
   <LOP Ten="Lớp 10C" />
   <LOP Ten="Lớp 10D" />
 </KHOI>
 <KHOI Ten="Khối 11" >
   <LOP Ten="Lớp 11A" />
   <LOP Ten="Lớp 11B" />
   <LOP Ten="Lớp 11C" />
 </KHOI>
 <KHOI Ten="Khối 12" >
   <LOP Ten="Lớp 12A" />
   <LOP Ten="Lớp 12B" />
   <LOP Ten="Lớp 12C" />
 </KHOI>
</TRUONG>
```

Chương trình Xuat_truong.xslt sau sẽ kết xuất (dạng Html) các thông tin về trường (bao gồm thông tin khối, lớp)

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output method ="html" />
 <xsl:template match="/" >
   <xsl:apply-templates  select="TRUONG" />
 </xsl:template>
 <xsl:template match="TRUONG">
   <xsl:value-of select="@Ten"/>
   <br />
```

```
   Danh sách các khối lớp <br />
   <xsl:apply-templates select="KHOI" />
 </xsl:template>
 <xsl:template match="KHOI">
   <xsl:value-of select="@Ten"/>
   <br />
   <xsl:apply-templates select="LOP" />
 </xsl:template>
  <xsl:template match="LOP">
    <xsl:value-of select="@Ten"/>
    <br />
 </xsl:template>
</xsl:stylesheet>
```

Ghi chú :

Thuộc tính select trong thẻ xsl:apply-templates có thể được bỏ qua và khi đó sẽ được hiểu là select="*" (cho lượng giá là các nút con của nút ngữ cảnh)

=== > Một trong các cách đơn giãn tổ chức chương trình Xslt là tổ chức chương trình theo các loại thẻ có trong tập tin Xml và gọi thực hiện (so khớp) không cần tham số

Gọi thực hiện :

```
<xsl:apply-templates />
```

Khai báo hàm/mẩu so khớp :

```
<xsl:template match="tên loại thẻ"
   > Các thẻ xử lý
</xsl:template>
```

```
<xsl:stylesheetversion="1.0"xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output method ="html" />
 <xsl:template match="/" >
   <xsl:apply-templates  />
 </xsl:template>
 <xsl:template match="TRUONG">
   <xsl:value-of select="@Ten"/>
   <br />
   Danh sách các khối lớp <br />
   <xsl:apply-templates  />
 </xsl:template>
 <xsl:templatematch="KHOI">
   <xsl:value-of select="@Ten"/>
   <br />
```

```
    <xsl:apply-templates  />
  </xsl:template>
  <xsl:templatematch="LOP">
    <xsl:value-ofselect="@Ten"/>
    <br />
  </xsl:template>
 </xsl:stylesheet>
```

III. Một số kỹ thuật xử lý

Mục tiêu : Trình bày 2 kỹ thuật cơ bản tương ứng 2 ứng dụng chính của XML

- Biến đối XML ---- > HTML
- Biến đổi XML ---- > XML

1) XML -- > HTML

Mục tiêu : Trình bày một số kỹ thuật cơ bản cho phép thể hiện nội dung tập tin Xml trên trang Web

Ví dụ :

với tập tin Xml Don_thuc.xml

```
<DON_THUC  He_so="4" So_mu="6" />
```

chương trình Xslt sau sẽ cho phép thể hiện đơn thức dưới dạng trình bày trên Web

Chương trình Xuat_don_thuc.xslt

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet  version="1.0"  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output  method ="html" />
 <xsl:template match="/" >
<html>
    <body>
        <xsl:apply-templates />
    </body>
</html>
 </xsl:template>
 <xsl:template match="DON_THUC" >
    P(x)=<xsl:value-of  select ="@He_so" /> x<sup><xsl:value-of  select ="@So_mu"/></sup>
 </xsl:template>
</xsl:stylesheet>
```

a) XML -- > Thẻ select

Vấn đề : Cần xuất danh sách chọn trên trang Web từ một danh sách các nút của tập tin Xml trong một ứng dụng

Web

Ví dụ :

Xuất danh sách các mặt hàng

Xuất danh sách các đơn vị Xuất

danh sách các khối

....

Hướng giải quyết :

Sử dụng thẻ select, option của ngôn ngữ Html

....

```
<select>
  <xsl:apply-templates select="Biểu thức Xpath tương ứng danh sách" />
</select>
....
<xsl:template match="Biểu thức Xpath tương ứng một phần tử trong danh sách" >
   <option>
      Thẻ xử lý kết xuất giá trị
     </option>
</xsl:template>
```

Ví dụ :

Chương trình Xslt sau sẽ xuất danh sách chọn các đơn vị từ tập tin Cong_ty.xml

```
<?xmlversion="1.0"encoding="UTF-8" ?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output method ="html" />
 <xsl:template match="/" >
   <html>
     <body>
       <xsl:apply-templates  />
     </body>
   </html>
 </xsl:template>
 <xsl:template match="CONG_TY" >
 Danh sách đơn vị :
   <select>
     <xsl:apply-templates  />
   </select>
 </xsl:template>
 <xsl:template match="DON_VI" >
   <option>
     <xsl:value-of  select ="@Ten"/></option>
 </xsl:template>
</xsl:stylesheet>
```

b) XML -- > Thẻ Table

Vấn đề :

Cần xuất danh sách dạng lưới trên trang Web từ một danh sách các nút của tập tin Xml trong một ứng dụng Web

Ví dụ :

Xuất danh sách các mặt hàng : tên , đơn giá

Xuất danh sách các nhân viên : Họ tên , Ngày sinh , Giới tính

Xuất danh sách các môn học : tên môn , Số tiết LT, Số tiết thực hành

....

Hướng giải quyết :

Sử dụng thẻ table , tr, td của ngôn ngữ Html

....

```
<table>
   <xsl:apply-templates select="Biểu thức Xpath tương ứng danh sách" />
</table>
....
<xsl:template match="Biểu thức Xpath tương ứng một phần tử trong danh sách" >
  <tr>
    <td> Thẻ xử lý kết xuất giá trị tại cột thứ 1</td>
    <td>Thẻ xử lý kết xuất giá trị tại cột thứ 2 </td>
    .....
</tr>
</xsl:template>
```

Ví dụ :

Chương trình Xslt sau sẽ xuất bảng đơn giá thuê phòng từ tập tin Bang_don_gia.xml

```
<?xml  version="1.0"  encoding="UTF-8" ?>
<xsl:stylesheet  version="1.0"xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output  method ="html" />
 <xsl:template  match="/" >
       <html>
         <body>
       <div align="center">
        Bảng đơn giá thuê phòng <br />
        <xsl:apply-templates  />
          </div>
         </body>
       </html>
 </xsl:template>
 <xsl:template match="BANG_DON_GIA" >
```

```
<table border="2">
 <tr>
   <td> Loại phòng</td>
  <td> đơn giá </td>
 </tr>
 <xsl:apply-templates />
   </table>
</xsl:template>
<xsl:template match="LOAI_PHONG" >
  <tr>
   <td><xsl:value-of select ="@Ten"/></td>
   <td> <xsl:value-of select ="@Don_gia"/> </td>
  </tr>
</xsl:template>
</xsl:stylesheet>
```

2) XML --- > XML

Mục tiêu : Trình bày một số kỹ thuật cơ bản cho phép tạo tài liệu Xml mới dựa trên một tài liệu Xml đã có

- Trích rút thông tin
- Tái cấu trúc

a) Tạo nút và thuộc tính

Vấn đề : Cần tạo thẻ mới X cùng với các thuộc tính trong tập tin xml kết xuất

Hướng giải quyết :

Cách 1 : Tạo lập trực tiếp thẻ mới X trong chương trình Xslt (tương tự như soạn thảo tập tin XML)

Cách 2 : Sử dụng các thẻ xử lý xsl:element , xsl:attribute

- **Thẻ xsl:element**

Ý nghĩa :

Cho phép tạo thẻ mới trong tập tin Xml kết xuất

Cú pháp :

```
<xsl:element  name="Ten_the" >
        Các thẻ xử lý tạo thuộc tính ( nếu có )
        Các thẻ khác
</xsl:element>
```

- **Thẻ xsl:attribute**

Ý nghĩa : Cho phép tạo thuộc tính của một thẻ trong tập tin Xml kết xuất

Cú pháp :

```
<xsl:element name="Ten_the" >
        <xsl:attribute name="ten_thuoc_tinh" >
                Thẻ xử lý kết xuất giá trị của thuộc tính
```

</xsl:attribute>

Các thẻ khác

</xsl:element>

Vi dụ :

Chương trình Xslt sau đây biến đổi tập tin Phieu_thu.xml với tập tin Xml kết xuất có các nút con tương ứng các thuộc tính

```
<?xmlversion="1.0"encoding="utf-8" ?>
<PHIEU_THU Khach_hang="Trần văn Long"  Ngay_thu="11/2/2007"  So_tien="40000" >
</PHIEU_THU>
```

Chương trình Xslt

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet version="1.0"  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output  method ="xml"  indent="yes" />
 <xsl:template match="/" >
   <xsl:apply-templates />
 </xsl:template>
 <xsl:template  match="PHIEU_THU" >
   <xsl:variable name="Ngay_thu" select ="@Ngay_thu" />
   <xsl:variable name="So_tien" select ="@So_tien" />
   <xsl:variable name ="Ten" select ="@Khach_hang" />
   <PHIEU_THU  Ngay_thu="{$Ngay_thu}"  So_tien="{$So_tien}">
     <KHACH_HANG  Ten="{$Ten}" />
   </PHIEU_THU>
 </xsl:template>
</xsl:stylesheet>
```

*** Sao chép nút**

Vấn đề : Cần tạo thẻ kết xuất trong tập tin xml kết xuất có cùng tên và các thuộc tính với thẻ trong tập tin nguồn

Hướng giải quyết :

Cách 1 : Sử dụng các thẻ xử lý xsl:element , xsl:attribute

Cách 2 : Sử dụng các thẻ xử lý xsl:copy , xsl:attribute

Thẻ xsl:copy

Ý nghĩa :

Cho phép sao chép thẻ từ tập tin xml nguồn (với nút ngữ cảnh tương ứng thẻ) sang tập tin xml kết xuất

Cú pháp :

<xsl:copy >

Các thẻ xử lý tạo thuộc tính (nếu có) Các thẻ khác

```
    </xsl:copy>
```

Kết hợp với xsl:attribute để sao chép thẻ - thuộc tính

```
    <xsl:copy >
        <xsl:for-each select="@*" >
            <xsl:attribute name="{name()}" >
                <xsl:value-of  select="." />
            </xsl:attribute>
        </xsl:for-each>
    </xsl:copy>
```

Ví dụ 1:

Chương trình sau trích danh sách các khối của tập tin truong.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet version="1.0"xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output  method ="xml" indent="yes" />
 <xsl:template match="/" >
      <xsl:apply-templates />
 </xsl:template>
 <xsl:template match="TRUONG" >
   <xsl:copy >
     <xsl:attribute name="Ten" >
       <xsl:value-of select="@Ten"/>
     </xsl:attribute>
     <xsl:apply-templates />
   </xsl:copy>
 </xsl:template>
 <xsl:template match="KHOI" >
   <xsl:copy >
     <xsl:attribute name="Ten" >
       <xsl:value-of select="@Ten"/>
     </xsl:attribute>
   </xsl:copy>
 </xsl:template>
</xsl:stylesheet>
```

Ví dụ 2 :Chương trình Xslt sau cho phép biến đổi tập tin Xml bất kỳ theo qui tắc : Tất cả thuộc tính sẽ biến thành thẻ con

```
<?xml  version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet version="1.0"xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output  method ="xml" indent="yes" />
```

```
<xsl:template match="/" >
  <xsl:apply-templates />
</xsl:template>
<xsl:template match="*" >
  <xsl:copy >
    <xsl:for-each select="@*" >
      <xsl:element  name="{name()}" >
       <xsl:value-of  select="."/>
      </xsl:element>
    </xsl:for-each>
    <xsl:apply-templates />
  </xsl:copy>
</xsl:template>
</xsl:stylesheet>
```

*** Sao chép nút - thuộc tính - nút con**

Vấn đề : Cần sao chép toàn bộ thẻ X , tất cả thuộc tính của X, tất các thẻ con mọi cấp của X trong tập tin xml nguồn vào tập tin Xml kết xuất

Hướng giải quyết :

Cách 1 : Sử dụng các thẻ xử lý xsl:copy , xsl:attribute

Cách 2 : Sử dụng thẻ xử lý xsl:copy-of

- **Thẻ xsl:copy-of**

Ý nghĩa :

Cho phép sao chép toàn bộ thẻ X , tất cả thuộc tính của X, tất các thẻ con mọi cấp của X trong tập tin xml nguồn vào tập tin Xml kết xuất

Cú pháp :

```
<xsl:copy-of select="Biểu thức Xpath" />
```

Vi dụ :

Cho tập tin xml Bang_phan_cong.xml

```
<?xml version="1.0" encoding="utf-8" ?>
<TRUONG Ten="Trường X" >
 <MON Ten="Toán">
   <GIAO_VIEN Ten="Trần văn Minh" >
     <LOP Ten="10A1" />
     <LOP Ten="11A1" />
     <LOP Ten="12A1"  />
   </GIAO_VIEN>
   <GIAO_VIEN Ten="Trần văn Lộc" >
     <LOP Ten="10B1"  />
```

```
    <LOP Ten="11B1"  />
    <LOP Ten="12B1"  />
  </GIAO_VIEN>
  <GIAO_VIEN Ten="Trần văn Hùng" >
    <LOP Ten="10B2"  />
    <LOP Ten="11A3"  />
    <LOP Ten="12B2"  />
  </GIAO_VIEN>
</MON>
<MON Ten="Lý">
  <GIAO_VIEN Ten="Lê văn Sơn" >
    <LOP Ten="10A1"  />
    <LOP Ten="11B1"  />
    <LOP Ten="12C1"  />
  </GIAO_VIEN>
  <GIAO_VIEN Ten="Nguyễn thị bé Nhỏ" >
    <LOP Ten="10B1"  />
    <LOP Ten="11B2"  />
    <LOP Ten="12B1"  />
  </GIAO_VIEN>
</MON>
<MON  Ten="Hóa">
  <GIAO_VIEN Ten="Ngô thị Bích" >
    <LOP Ten="10B1"  />
    <LOP Ten="11B1"  />
    <LOP Ten="12C1"  />
  </GIAO_VIEN>
  <GIAO_VIEN Ten="Lê văn Lớn" >
    <LOP Ten="10A1"  />
    <LOP Ten="12B1"  />
  </GIAO_VIEN>
  <GIAO_VIEN Ten="Đỗ thị Tuyết" >
    <LOP Ten="10B2"  />
    <LOP Ten="11A3"  />
    <LOP Ten="12B2" />
  </GIAO_VIEN>
</MON>
<MON Ten="Sinh">
```

```
   <GIAO_VIEN Ten="Nguyễn hùng Cường">
     <LOP Ten="10A1"  />
     <LOP Ten="11A1"  />
     <LOP Ten="12A1"  />
   </GIAO_VIEN>
   <GIAO_VIEN Ten="Lê văn Tùng" >
     <LOP Ten="11A2"  />
     <LOP Ten="12B1"  />
   </GIAO_VIEN>
   <GIAO_VIEN Ten="Nguyễn thị đẹp" >
     <LOP Ten="11B2"  />
     <LOP Ten="11A3"  />
     <LOP Ten="11B3" />
   </GIAO_VIEN>
 </MON>
</TRUONG>
```

Đoạn chương trình sau cho phép tái cấu trúc tập tin Bang_phan_cong.xml với thẻ MON chuyển thành thuộc tính Bo_mon của thẻ GIAO_VIEN

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output  method ="xml" indent="yes" />
 <xsl:template match="/" >
   <xsl:apply-templates />
 </xsl:template>
 <xsl:template match="TRUONG" >
   <xsl:copy >
     <xsl:attribute name="Ten" >
        <xsl:value-of select="@Ten"/>
     </xsl:attribute>
     <xsl:apply-templates />
   </xsl:copy>
 </xsl:template>
 <xsl:template match="GIAO_VIEN" >
   <xsl:copy >
     <xsl:attribute name="Ten" >
      <xsl:value-of select="@Ten"/>
     </xsl:attribute>
```

```
    <xsl:attribute name="Bo_mon" >
      <xsl:value-of  select="../@Ten"/>
    </xsl:attribute>
    <xsl:copy-of  select="LOP"/>
      </xsl:copy>
 </xsl:template>
</xsl:stylesheet>
```

*** Sắp thứ tự các nút**

Vấn đề : Cần sắp thứ tự danh sách các thẻ X của tập tin xml kết xuất

Hướng giải quyết :

Sử dụng thẻ xử lý xsl:sort kết hợp với xsl:apply-templates

- **Thẻ xsl:sort**

Ý nghĩa :

Cho phép sắp thứ tự danh sách các thẻ X của tập tin xml kết xuất

Cú pháp : Sắp tăng

```
<xsl:sort order="accending" select="Thuộc tính" />
```

Sắp giảm

```
<xsl:sort order="descending" select="Thuộc tính" />
```

Kết hợp với xsl:apply-templates để tiến hành sắp thứ tự các kết quả sau khi thực hiện so khớp các hàm/mẩu

```
<xsl:apply-templates select="Biểu thức Xpath" >
   <xsl:sort  order="...."  select="...." />
   <xslsort  order="...."  select="...." />
    .....
</xsl:apply-templates>
```

Ví dụ : Với tập tin Xml Ket_qua_Olympic.xml

```
<?xml version="1.0" encoding="utf-8" ?>
<KET_QUA>
  <QUOC_GIA Ten="AAA" So_vang="10" So_bac="7" So_dong="2" />
  <QUOC_GIA Ten="XXX" So_vang="6" So_bac="0" So_dong="12" />
  <QUOC_GIA Ten="BBB" So_vang="10" So_bac="8" So_dong="13" />
  <QUOC_GIA Ten="DDD" So_vang="4" So_bac="17" So_dong="0" />
  <QUOC_GIA Ten="MMM" So_vang="6" So_bac="1" So_dong="0" />
  <QUOC_GIA Ten="KKK" So_vang="6" So_bac="0" So_dong="2" />
  <QUOC_GIA Ten="LLL" So_vang="10" So_bac="4" So_dong="23" />
  <QUOC_GIA Ten="PPP" So_vang="3" So_bac="27" So_dong="100" />
</KET_QUA>
```

đoạn chương trình XSL sau sắp xếp các quốc gia giảm dần theo thứ tự ưu tiên

- Ưu tiên 1 : Số huy chương vàng

- Ưu tiên 2 : Số huy chương bạc
- Ưu tiên 3 : Số huy chương đồng

```
<?xml  version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet  version="1.0"  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
 <xsl:output   method ="xml" indent="yes" />
 <xsl:template match="/" >
   <xsl:apply-templates />
 </xsl:template>
 <xsl:template match="KET_QUA"  >
   <xsl:copy >
     <xsl:apply-templates  select ="QUOC_GIA">
       <xsl:sort  order="descending" data-type="number" select="@So_vang" />
       <xsl:sort  order="descending" data-type="number" select="@So_bac" />
       <xsl:sort  order="descending" data-type="number" select="@So_dong" />
     </xsl:apply-templates>
   </xsl:copy>
 </xsl:template>
 <xsl:templatematch="QUOC_GIA" >
   <!--<xsl:copy-of select="."/>-->
        <xsl:copy >
     <xsl:copy-of select ="@*"/>
        </xsl:copy>
 </xsl:template>
</xsl:stylesheet>
```

IV. Bài tập

1. XML --- > HTML

*** Tích 2 phân số**

Yêu cầu :

Viết chương trình XSLT cho phép tạo kết xuất từ dữ liệu nguồn

- Dữ liệu nguồn : Tập tin xml biểu diễn thông tin về 2 phân số
- Kết xuất : Trang Web thể hiện kết quả nhân 2 phân số

Ví dụ : Với phân số 4/7, 5/11

Kết xuất sẽ là

Kết quả tính tích 2 phân số 1/7 và 5/11

4/7*5/11=20/77

*** Phương trình đường thẳng**

Yêu cầu :

Viết chương trình XSLT cho phép tạo kết xuất từ dữ liệu nguồn

- Dữ liệu nguồn : Tập tin xml biểu diễn thông tin về các hệ số của phương trình đường thẳng trong mặt phẳng

- Kết xuất : Trang Web thể hiện kết quả là phương trình đường thẳng

Ví dụ :

Với giá trị các hệ số 2,3,4

Kết xuất sẽ là : Phương trình đường thẳng 2x + 3y + 4=0

Với giá trị các hệ số 7,-3

Kết xuất sẽ là : Phương trình đường thẳng 7x - 3y =0

*** Bảng xếp hạng Olympic**

Yêu cầu :

Viết chương trình XSLT cho phép tạo kết xuất từ dữ liệu nguồn

- Dữ liệu nguồn : Tập tin xml biểu diễn thông tin kết quả thi đấu Olympic các quốc gia

- Kết xuất :

a) Trang Web thể hiện bảng kết quả thi đấu

b) Trang Web cho phép cập nhật số huy chương vàng, bạc, đồng

*** Bảng điểm sinh viên:**

- Dữ liệu nguồn : Tập tin xml biểu diễn thông tin điểm thi của sinh viên

```
<?xml version="1.0" encoding="utf-8" ?>
<SV maso="T01"  hoten="Nguyen Thi Ngoc Trang" lop="10T1">
  <Hocphan maso="JV2" ten="Java 2" tinchi="3" hocky="1">7</Hocphan>
  <Hocphan maso="CS" ten="Co so du lieu" tinchi="3" hocky="1">8</Hocphan>
  <Hocphan maso="VB" ten="Lap trinh truc quan" tinchi="3" hocky="1">8</Hocphan>
  <Hocphan maso="WEB" ten="Lap trinh web" tinchi="2" hocky="1">3</Hocphan>
</SV>
<SV maso="T02" hoten="Tran Xuan Tinh" lop="10T1">
  <Hocphan maso="JV2" ten="Java 2" tinchi="3" hocky="1">4</Hocphan>
  <Hocphan maso="CS" ten="Co so du lieu" tinchi="3" hocky="1">9</Hocphan>
  <Hocphan maso="VB" ten="Lap trinh truc quan" tinchi="3" hocky="1">5</Hocphan>
  <Hocphan maso="WEB" ten="Lap trinh web" tinchi="2" hocky="1">10</Hocphan>
  <Hocphan maso="CT" ten="Tu tuong Ho Chi Minh" tinchi="2" hocky="1">7</Hocphan>
</SV>
<SV maso="T03" hoten="Pham Van Dung" lop="10T1">
  <Hocphan maso="JV2" ten="Java 2" tinchi="3" hocky="1">6</Hocphan>
  <Hocphan maso="CS" ten="Co so du lieu" tinchi="3" hocky="1">8</Hocphan>
  <Hocphan maso="VB" ten="Lap trinh truc quan" tinchi="3" hocky="1">4</Hocphan>
  <Hocphan maso="WEB" ten="Lap trinh web" tinchi="2" hocky="1">9</Hocphan>
  <Hocphan maso="DA" ten="Do an phan mem" tinchi="2" hocky="1">5</Hocphan>
  <Hocphan maso="MG" ten="Mang may tinh" tinchi="3" hocky="1">8</Hocphan>
</SV>
```

```
 <SV maso="T04" hoten="Luong Thanh Dung" lop="10T2">
   <Hocphan maso="MG" ten="Mang may tinh" tinchi="3" hocky="1">10</Hocphan>
 </SV>
</DCT>
```

Yêu cầu :

Viết chương trình XSLT kết xuất dữ liệu ra trang web thông tin kết quả điểm thi của sinh viên có mã số T03 theo mẫu sau:

Mã sinh viên:

Họ tên sinh viên:

Lớp:

Mã môn | Tên môn học | Số tín chỉ | Điểm

----- tương tự như trên, nếu có nhiều hơn 1 môn

2. XML - XML

*** Hồ sơ nhân viên**

Yêu cầu :

Viết chương trình XSLT cho phép tạo kết xuất từ dữ liệu nguồn

- Dữ liệu nguồn : Tập tin xml biểu diễn thông tin về hồ sơ nhân viên với
  + Thông tin bao gồm : Họ và tên, Giới tính , Ngày sinh, địa chỉ , đơn vị
  + Tất cả các thông tin đều biểu diễn dưới dạng thẻ con
- Kết xuất : Tập tin Xml

a) Tất cả các thông tin đều biểu diễn dạng thuộc tính

b) Tất cả các thông ngoại trừ đơn vị đều biểu diễn dạng thuộc tính

*** Trường - khối - lớp**

Yêu cầu :

Viết chương trình XSLT cho phép tạo kết xuất từ dữ liệu nguồn

- Dữ liệu nguồn : Tập tin xml biểu diễn thông tin tổ chức trường , các khối của trường, các lớp của khối
- Kết xuất :

a) Tập tin Xml chỉ bao gồm các lớp có sỉ số trên 30

b) Tập tin Xml chỉ bao gồm các khối có hơn 5 lớp

*** Bảng phân công giáo viên**

Yêu cầu :

Viết chương trình XSLT cho phép tạo kết xuất từ dữ liệu nguồn

- Dữ liệu nguồn : Tập tin xml biểu diễn thông tin bảng phân công các giáo viên của một trường
- Kết xuất :

a) Tập tin Xml chỉ bao gồm danh sách các bộ môn cùng với số lượng các giáo viên

b) Tập tin Xml chỉ bao gồm danh sách các giáo viên được phân công dạy trên 2 lớp