Nesta aula iremos resolver exercícios envolvendo a extracção e a transformação de informação proveniente de documentos HTML, com base na tecnologia *XQuery*. O software usado nas aulas de laboratório oferece, como complemento às funções da biblioteca XPath 2.0, algumas **funções de extensão relacionadas com o processamento de documentos HTML**. Entre estas funções de extensão encontram-se as seguintes:

- Função `java:gti.htmldoc()`, que utiliza uma API Java de nome *TagSoup* para converter um documento HTML com erros de formatação num documento XML bem formado (i.e., converter o HTML num documento XHTML). Esta função aceita como parâmetro um URI para um documento HTML e retorna um documento XML.
- Função `java:gti.httppost()`, que aceita como entrada um URL para um documento HTML que se encontre online num servidor Web, uma lista de parâmetros a serem enviados ao servidor Web, e uma lista de valores para cada um dos parâmetros da lista anterior, usando o método HTTP POST por forma a aceder ao servidor Web e retornar um documento XML com o conteúdo correspondente.
- Função `java:gti.textdoc()`, que aceita como parâmetro um URI para um documento e retorna a cadeia de caracteres correspondente ao seu conteúdo textual. Caso a entrada fornecida à função seja um documento XML ou HTML, a função irá retornar apenas o conteúdo textual, removendo instruções de marcação (i.e., as *tags*).

A Figura 1 exemplifica a utilização destas funções, mostrando como se pode aceder ao título da página Web do Instituto Superior Técnico.

```
declare namespace gti = "java:gti";
declare namespace x = "http://www.w3.org/1999/xhtml";

let $mydoc := gti:htmldoc("http://www.ist.utl.pt")
return $mydoc//x:title
```

Figura 1 – Um programa XQuery que exemplifica como aceder ao título de uma página HTML.

Nesta aula de laboratório irá também ser usada a ferramenta *RoadRunner*, a qual permite a extracção automática de informação relevante desde documentos HTML, com base num método de aprendizagem automática não supervisionado. Esta ferramenta encontra-se também já integrada na bancada de software usada nas aulas de laboratório:

- Função `java:gti.rrdata()`, que aceita como entrada uma lista de documentos XHTML e devolve como resultado um documento XML com a informação extraída.

Mais informações sobre a ferramenta *RoadRunner* podem ser obtidas desde o site Web do projecto que esteve na sua origem : http://www.dia.uniroma3.it/db/roadRunner/.

Exercício 1

Escreva um programa XQuery que aceda ao site Web de apostas desportivas que se encontra online no URL http://www.expekt.com/odds/eventsodds.jsp e faça a extracção de todas as apostas que lá estão definidas sob a forma de uma tabela HTML.

O programa XQuery deverá produzir como resultado um documento XML de acordo com o formato que se exemplifica na Figura 2.

```
<aposta competicao="Spa. ACB Basket"
        evento="Gran Canaria - Barcelona"
        hora="20:30">
   <valor_ganhar_1>3.12</valor_ganhar_1>
   <valor_empatar> </valor_empatar>
   <valor_ganhar_2>1.35</valor_ganhar_2>
</aposta>
```

Figura 2 – Documento XML com informação sobre apostas desportivas.

Exercício 2

Utilize a ferramenta *RoadRunner* por forma a extrair das páginas Web que se listam abaixo, as quais apresentam informação sobre teses de mestrado realizadas no DEI-IST, a informação sobre as dissertações.

- **https://fenix.ist.utl.pt/publico/department/theses.do?method=showThesisDetails&selectedDepartmentUnitID=61176&thesisID=37816&contentContextPath_PATH=/departamentos/dei/lateral/dissertacoes&_request_checksum_=a550a7bfd0fee674e997644f1b99da1759a4815a**
- **https://fenix.ist.utl.pt/publico/department/theses.do?method=showThesisDetails&selectedDepartmentUnitID=61176&thesisID=35216&contentContextPath_PATH=/departamentos/dei/lateral/dissertacoes&_request_checksum_=a299bedf407810b013a4915e99f309bf3a973c55**
- **https://fenix.ist.utl.pt/publico/department/theses.do?method=showThesisDetails&selectedDepartmentUnitID=61176&thesisID=41614&contentContextPath_PATH=/departamentos/dei/lateral/dissertacoes&_request_checksum_=61b0da30c228f539e05acb7b5a01945a0e8d8678**
- **https://fenix.ist.utl.pt/publico/department/theses.do?method=showThesisDetails&selectedDepartmentUnitID=61176&thesisID=37433&contentContextPath_PATH=/departamentos/dei/lateral/dissertacoes&_request_checksum_=084121c3dad76de9930ea43ff9bba0f3f4d61c3a**

Crie ainda um programa XQuery que, operando sobre os documentos XML produzidos como resultado pelo *RoadRunner,* apresente os dados seguindo o formato da Figura 3.

```
<tese>
   <aluno>Manuel</aluno>
   <titulo>Uso de XQuery em aplicações na Web</titulo>
   <palavras_chave>XML, X-Query, Web<palavras_chave>
</tese>
```

Figura 3 – Documento XML com informação sobre as dissertações.

Exercício 3

O autor do conhecido motor de XQuery *Saxon*, de nome Michael Kay, mantém um blog sobre a sua actividade neste projecto. O blog está disponível em http://saxonica.blogharbor.com/.

3.1 - Escreva um programa XQuery que aceda a todas as entradas no blog do Michael Kay e verifique quais as entradas que mencionam a expressão "XQuery Update" ou o termo "performance" no texto.

Uma possível estratégia de concretização passa por desenvolver um programa que aceda ao conteúdo da página de entrada do blog e siga recursivamente os links com a âncora textual "Previous", por forma a encontrar os URLs de todos os *posts*.

3.2 - Escreva um programa XQuery que aceda às dez primeiras entradas no blog e que verifique se as entradas têm uma conotação positiva, negativa ou neutra. No URL https://dspace.ist.utl.pt/bitstream/2295/366636/1/words-sentiment.xml encontra-se um documento XML que contem uma lista de palavras classificadas de acordo com a sua polaridade, i.e. palavras que tipicamente têm uma conotação positiva, neutra ou negativa.

Para cada uma das entradas no blog, deve contar o número de ocorrências de palavras positivas e negativas. Com base nestas contagens, o programa XQuery deve produzir um documento XML que indique, para cada título de entrada no blog, uma pontuação obtida pela soma do valor +1 para cada palavra positiva, e -1 para cada palavra negativa.

Exercício 4

Em http://saxonica.blogharbor.com/blog/index.xml encontra-se um *feed* XML no formato RSS com informação correspondente às entradas efectuadas no blog do Michael Kay.

Para cada um dos seguintes casos, escreva **expressões XQuery** que, utilizado as funções XPath para processamento de expressões regulares ou as funções de extensão introduzidas nas aulas de laboratório para a pesquisa de elementos textuais com base em palavras chave, permitam aceder à informação pretendida.

4.1 – Encontrar os títulos de todas as entradas que obedeçam aos seguintes critérios:

1. Contêm a palavra chave "performance" na descrição da entrada.
2. Não contêm a palavra "tinytree" na descrição da entrada.
3. Têm a expressão "compile-time" no título da entrada.
4. Foram efectuadas numa hora do dia que corresponde a um número ímpar.

A resolução de cada uma das alíneas deve considerar a ordenação dos resultados de acordo com o grau de relevância para com os critérios de pesquisa.

4.2 – Extrair os URL que se encontram mencionados no texto descritivo das entradas que contêm as palavras "published" ou "downloaded". Deverá ter o cuidado de eliminar potenciais URL em duplicado.

Exercício 5

No URL http://cdiac.ornl.gov/ftp/trends/co2/sposio.co2 encontra-se um documento de texto contendo informação sobre a concentração atmosférica de CO^2 na zona do Pólo Sul.

Escreva um programa XQuery que aceda ao documento de texto e represente a informação sob a forma de um documento XML, seguindo o exemplo da Figura 4.

```
<Measurements>
 <Measurement>
  <Year>1940</Year>
  <Jan>-99.99</Jan>
  <Aug>-99.99</Aug>
  <Annual>-99.99</Annual>
  <Annual-Fitted>-99.99</Annual-Fitted>
 </Measurement>
</Measurements>
```

Figura 4 – Documento XML representando informação sobre concentrações de CO^2

Exercício 6

Escreva um **programa XQuery** que aceda ao site Yahoo! Shopping (i.e., converta o HTML para um documento bem formado para posteriormente ser processado com expressões XPath e XQuery) e construa um novo documento XML com uma lista de máquinas fotográficas digitais da marca Nikon.

O URL do site Yahoo! Shopping é http://shopping.yahoo.com. Para aceder à informação sobre as máquinas Nikon deve ser usado o seguinte URL:

http://shopping.yahoo.com/s:Digital%20Cameras:4168-Brand=Nikon:browsename=Nikon%20Digital%20Cameras

Tenha em atenção que a listagem completa das máquinas Nikon é apresentada através de uma sequencia de páginas HTML, em que cada uma contem apenas 15 itens. Na parte inferior cada uma das páginas encontra-se um índice que permite aceder às restantes, por exemplo através dos links "*Previous*" e "*Next*".

O programa XQuery desenvolvido neste exercício deverá **recolher até 10 páginas da listagem de produtos (um máximo de 150 produtos)**. Uma estratégia de concretização é, por exemplo, seguir um máximo de 10 links com a âncora textual "Next".

O programa XQuery desenvolvido neste exercício deverá ainda produzir como resultado um documento XML segundo o formato que se exemplifica na Figura 5.

```
<produtos>
   <produto>
      <nome>Camera Name</nome>
      <desc>Nikon – Camera Name - Compact - 7.1 Megapixels...</desc>
      <preco>$169 - $200</preco>
   </produto>
  <!-- restantes máquinas fotográficas -->
</produtos>
```

Figura 5 – Documento XML com informação sobre máquinas fotográficas.

Exercício 7

Em http://www.indexgeo.com.au/data/capital/capital.csv encontra-se um ficheiro de texto com valores separados por vírgulas, contendo informação sobre as capitais de estado Australianas e as coordenadas de latitude e longitude correspondentes.

Escreva um **programa XQuery** para aceder ao documento online e converter os resultados para o formato KML usado pelo Google Earth. O formato de saída encontra-se exemplificado na Figura 6.

```
<kml xmlns="http://www.opengis.net/kml/2.2">
<Response>
 <name>places in australia</name>
 <Status><code>200</code><request>geocode</request></Status>
 <Placemark id="p1">
  <address>Sydney NSW, Australia</address>
  <AddressDetails Accuracy="4">
   <Country>
    <CountryNameCode>AU</CountryNameCode>
    <CountryName>Australia</CountryName>
   </Country>
  </AddressDetails>
  <Point>
   <coordinates>151.207114,-33.867139,0</coordinates>
  </Point>
 </Placemark>
</Response>
</kml>
```

Figura 6 – Documento KML de exemplo.