Processamento de Linguagens
LEI (3ºano)

Trabalho Prático nº 1 (FLex)

Ano lectivo 14/15

1 Objectivos e Organização

Este trabalho prático tem como principais **objectivos**:

- aumentar a experiência de uso do ambiente Linux, da linguagem imperativa C (para codificação das estruturas de dados e respectivos algoritmos de manipulação), e de algumas ferramentas de apoio à programação;
- aumentar a capacidade de escrever *Expressões Regulares (ER)* para descrição de *padrões de frases*;
- desenvolver, a partir de ERs, sistemática e automaticamente *Processadores de Linguagens Regulares*, que filtrem ou transformem textos;
- utilizar *geradores de filtros de texto*, como o Flex

Para o efeito, esta folha contém vários enunciados, dos quais deverá resolver pelo menos um, devendo entregar (eletronicamente no Bb) o relatório (PDF) de desenvolvimento do projeto, explicando o problema e a sua solução, **até Domingo dia 29 de Março**.
O **relatório** a elaborar, deve ser claro e, além do respectivo enunciado, da descrição do problema, das decisões que lideraram o desenho da solução e sua implementação (incluir a especificação Flex), deverá conter exemplos de utilização (textos fontes diversos e respectivo resultado produzido). Como é de tradição, o relatório será escrito em LaTeX.
O programa desenvolvido será apresentado aos membros da equipa docente, totalmente pronto e a funcionar (acompanhado do respectivo relatório de desenvolvimento) e será defendido por todos os elementos do grupo, em data a marcar.

2 Enunciados

Para sistematizar o trabalho que se pede em cada uma das propostas seguintes, considere que deve, em qualquer um dos casos, realizar a seguinte lista de tarefas:

1. Especificar os padrões de frases que quer encontrar no texto-fonte, através de ERs.
2. Identificar as acções semânticas a realizar como reacção ao reconhecimento de cada um desses padrões.
3. Identificar as Estruturas de Dados globais que possa eventualmente precisar para armazenar temporariamente a informação que vai extraindo do texto-fonte ou que vai construindo à medida que o processamento avança.
4. Desenvolver um Filtro de Texto para fazer o reconhecimento dos padrões identificados e proceder à transformação pretendida, com recurso ao Gerador Flex.

2.1 Museu da Pessoa — tratamento de fotografias

Do material documental recolhido durante as entrevistas feitas para construir o Museu da Pessoa[1] fazem parte ficheiros de texto que descrevem as fotos disponibilizadas pelos entrevistados.
Essas descrições estão anotadas em XML para ser fácil de ler e processar, tal como se exemplifica a seguir.

```
  <fotos>
  <foto ficheiro="022-F-01.jpg">
 <quando data="1961-01-15"/>
 <quem>Ana de Lourdes de Oliveira Chaminé; António Oliveira
Machado</quem>
 <facto> Os noivos cortam o bolo de casamento</facto>
  </foto>
  <foto ficheiro="022-F-02.jpg">
 <onde>Casa Machado, Afurada, Vila Nova de Gaia</onde>
 <quando data="2000-09-12"/>
 <quem>Ana de Lourdes de Oliveira Chaminé; António Oliveira
Machado</quem>
 <facto>António Machado e a sua esposa, dona Ana atrás do
balcão da taberna Casa Machado.</facto>
  </foto>
  </fotos>
```

Após analisar cuidadosamente as amostras de descrições acima, desenvolva com o Flex um Filtro de Texto que gere um album em HTML em que se mostrem as fotografias mencionadas por ordem cronológica (da mais antiga para a mais recente), usando o *facto* como cabeçalho ou título de cada foto. Apresente no início uma espécie de índice com o nome de todas as pessoas retratadas.

2.2 Processamento de Entidades Nomeadas (Enamex)

Uma etapa comum no processamento de textos em lingua natural é o reconhecimento de entidades nomeadas, ou mencionadas[2]. Nessa tarefa identificam-se as palavras que podem ser substantivos próprios (nomes) e procura-se associar-lhes um tipo de entidade mencionada: Pessoa, Localidade, ou Organização. Para facilitar o seu posterior processamento, normalmente com vista à extração de informação, essas entidades são *marcadas*, ou *anotadas*, no texto inicial.
Existe uma dialeto XML chamado Enamex precisamente para esse efeito. As marcas mais usadas são:

```
<ENAMEX TYPE="PERSON">Francisco de Vilela Barbosa</ENAMEX>
<ENAMEX TYPE="LOCATION" SUBTYPE="COUNTRY">Portugal</ENAMEX>
<ENAMEX TYPE="LOCATION" SUBTYPE="CITY">Rio de Janeiro</ENAMEX>
<ENAMEX TYPE="ORGANIZATION">Universidade do Minho</ENAMEX>
<ENAMEX TYPE="?">Marquês de Pombal</ENAMEX>
<ENAMEX TYPE="?">Novembro</ENAMEX>
```

A seguir mostram-se 2 textos (retirados de biografias do espólio do Museu virtual da Emigração) anotados desta forma

```
  <ENAMEX TYPE="PERSON"> Bento de Castro Abreu </ENAMEX>, que
emigrou para o <ENAMEX TYPE="LOCATION" SUBTYPE="CITY"> Rio de Janeiro </ENAMEX> em
<TIMEX TYPE="DATE"> 1/2/1895 </TIMEX>,
com 26 anos de idade, é referido no seu passaporte como proprietário, sobrinho de
outro <ENAMEX TYPE="?"> Brasileiro </ENAMEX>
<ENAMEX TYPE="PERSON"> Fernando de Castro Abreu e Magalhães </ENAMEX>,
que apoiou financeiramente a construção da casa do
<ENAMEX TYPE="LOCATION"> Santo Novo </ENAMEX>.
```

[1] Um museu virtual que relata a vida dos cidadãos anónimos que tem o seu percurso de vida carregado de episódios interessantes, ora bons ora maus, e que constroem os organizações e as sociedades.
[2] Conhecida, em inglês técnico, como NER – Named-Entities Recognition.

e

```
<ENAMEX TYPE="?">Foi Visconde e Marquês de Paranaguá</ENAMEX> <ENAMEX TYPE="PERSON">Francisco
Vilela Barbosa</ENAMEX> que nasceu
no <ENAMEX TYPE="LOCATION" SUBTYPE="COUNTRY">Brasil</ENAMEX>, em
<ENAMEX TYPE="LOCATION" SUBTYPE="CITY">Parati</ENAMEX>
em <ENAMEX TYPE="?">Novembro</ENAMEX> de <NUMEX>1769</NUMEX> e ali
morreu em <TIMEX TYPE="DATE">Setembro de 1846</TIMEX>, filho de
<ENAMEX TYPE="PERSON">Francisco Vilela Barbosa</ENAMEX>,
natural de <ENAMEX TYPE="LOCATION" SUBTYPE="CITY">Braga</ENAMEX>
(<ENAMEX TYPE="LOCATION" SUBTYPE="Country">Portugal</ENAMEX>) e de
<ENAMEX TYPE="PERSON">D. Ana Maria da Conceição</ENAMEX>.

Formado em matemática pela <ENAMEX TYPE="ORGANIZATION">Universidade
de Coimbra</ENAMEX>, em <Numex>1789</NuMEX>
assentou praça na armada e quando 2º tenente prestou relevantes
serviços no cerco de <ENAMEX TYPE="LOCATION" SUBTYPE="CITY">Tunis</ENAMEX>
e na perseguição aos pirata argelinos do
<ENAMEX TYPE="?">Mediterrâneo</ENAMEX>.

Nomeado lente da <ENAMEX TYPE="ORGANIZATION">Academia da Marinha</ENAMEX>,
passou em <Numex>1801</NuMEX> para o
<ENAMEX TYPE="ORGANIZATION">Real Corpo de Engenheiros</ENAMEX>
e em <Numex>1810</NuMEX> foi promovido a <ENAMEX TYPE="?">Major</ENAMEX>
e reformado mais tarde no
posto de <ENAMEX TYPE="?">Brigadeiro</Enamex>.
```

Neste contexto, desenvolva um Filtro de Texto com o Flex para ler um documento anotado no sistema Enamex e:

a) Listar todas as Pessoas identificadas, sem repetições;

b) Listar os Países e Cidades marcadas;

c) Listar as Organizações.

Apresente os resultados em páginas HTML.

2.3 Análise de Código — processamento de comentários

Como sabe todas as linguagens de programação permitem o uso de comentários para ajudar a tornar o código-fonte dos programas mais compreensível.
Por exemplo, a linguagem Java permite o uso de *comentários in-line* começados por "//"(até ao fim de linha), de *comentários de bloco* que permitem abranger várias linhas e que começam em "/*"e vão até "*/", e ainda *comentários de documentação* que começam com "/**"e acabam como os anteriores mas dentro do bloco as linhas começam por um "*"e podem ter texto livre ou podem ter comandos para criar documentação técnica, tais como:

```
@author  name-text
@param  parameter-name description
@return  description
@see "string"
@see <a href="URL#value">label</a>
@version  version-text
```

Neste contexto, desenvolva um Filtro de Texto com o Flex para ler um programa-fonte em Java e:

a) contar o número de comentários de cada tipo;

b) mostrar os comentários in-line mostrando toda a linha onde aparecem;

c) mostrar os comentários de bloco indicando a linha onde começam e acabam e a lingua em que estão escritos (PT, EN ou "?")

d) mostrar os comentários de documentação de um determinado autor ou de uma determinada versão indicando a 1ª linha de código a seguir ao fim do comentário.

Apresente os resultados em páginas HTML para dar um aspeto legível e atraente aos elementos a analisar.

2.4 Análise de Código — processamento de identificadores

Na sequência do exercício anterior, um outro tipo de análise que se faz com frequência tem a ver com os identificadores (de variáveis, funções/métodos, classes) que se usam num programa. Esses identificadores, se bem escolhidos e usados pelo programador, dão informação preciosa quanto ao significado do programa. Por isso mesmo, as comunidades de programadores costumam estabelecer uma lista de boas práticas que incluem o uso de identificadores compostos, recorrendo ao *underscore* ou ao *camelCase* para separar as partes; por exemplo, "`ins_alunos`"ou "`totVendas`".

Neste contexto, desenvolva um Filtro de Texto com o Flex para ler um programa-fonte em Java e:

a) retirar todos os identificadores de variáveis, métodos e classes;

b) contar o número total de identificadores desses três grupos e o número de identificadores de cada tipo (*underscore* ou *camelCase*) para se poder concluir qual a boa prática que predomina (se alguma);

c) mostrar todos os cabeçalhos completos de classe;

d) mostrar os tipos dos parâmetros de cada método.

Apresente os resultados em páginas HTML para dar um aspeto legível e atraente aos elementos a analisar.

2.5 Processamento de ficheiros com Canções

No seu entusiasmo de colecionador de tudo o que seja artístico e literário, o prof. José João Almeida, arquiva em ficheiros de texto letras de canções portuguesas com alguma meta-informação inicial sobre cada uma.
Vejam-se os 2 exemplos abaixo que ilustram a forma como essa informação é armazenada nos ficheiros de texto.

```
title: Canta, canta amigo canta
from: jj
author: António Macedo

Canta canta amigo canta
vem cantar a nossa canção
tu sozinho não és nada
juntos temos o mundo na mão

Erguer a voz e cantar
À força de quem é novo
viver sempre a esperar
fraqueza de quem é povo
Viver em casa de tábuas
à  espera dum novo dia
enquanto a terra engole
a tua antiga alegria

Canta canta amigo canta
```

```
...
##---------------------------------
title: E alegre se fez triste
lyrics: Manuel Alegre
music: José Niza
singer: Adriano Correia de Oliveira
From: Fernando Pais

Aquela clara madrugada que
Viu lágrimas correrem no teu rosto
E alegre se fez triste como se
chovesse de repente em pleno Agosto

Ela só viu meus dedos nos teus dedos
Meu nome no teu nome e demorados
Viu nossos olhos juntos nos segredos
Que em silêncio dissemos separados

A clara madrugada em que parti
Só ela viu teu rosto olhando a estrada
Por onde o automóvel se afastava

E viu que a pátria estava toda em ti
E ouviu dizer adeus essa palavra
Que fez tão triste a clara madrugada
```

Neste contexto, desenvolva um Filtro de Texto com o Flex para ler uma ou mais canções, separadas por uma linha de hífens iniciada por "`##`", e gerar, para cada uma, um documento (da classe *article*) em LaTeX cujo título seja o indicado no descritor title e o autor seja, ou o indicado no descritor author, ou ambos os indicados como autores da lírica e da música.
Na secção única, sem numeração e designada por "Letra") deve aparecer o poema devidamente formatado e no fim de tudo, encostado è direita, deve aparecer o cantor, quando se souber.

2.6 Processamento do Dicionauro

O projecto dicionauro, criado pelo prof José João Almeida, pretende ser um dicionário infantil. Para o efeito as várias entradas foram criadas num editor normal devidamente anotadas nun sistema de marcação muito simples e ligeiro, tendo armazenadas em diferentes ficheiros conforme o tema. As entradas (cada uma caraterizada por vários campos de informação) estão separadas umas das outras por uma linha em branco.
Cada campo corresponde a 1 linha de texto e o seu descritor (ou identificador) é representado por uma sigla colocada no início da linha e separada da informação por 1 espaço.
Para perceber melhor, atende nos exemplos abaixo—onde se define *fruta*, *maça* e *pêra*—retirados do ficheiro "`alimento_fruta.tx`
Note que a 1ª linha tem sempre a palavra-chave em Português e por baixo a sua categoria gramatical (`catgra`) e exemplos de uso (`exuso`); a definição dessa palavra (descritor `Def`) aparece quase no fim.

```
PT fruta
-catgra nf
-exuso a banana é uma fruta.
EN fruit
AUDEN n
FR fruit
AUDFR n
ES fruta
AUDES n
DE frucht
```

```
AUDDE n
CN ??
NU 10015
DOM alimento
BT
PART semente
PART casca
PART polpa
POF árvore
UP fazer salada, sumo, doce, batido, compota, licor, bolo, gelado e tarte
Def algo como a banana, a maçã ou o morango, que cresce a partir de uma árvore ou planta
GI 2
IM 3
ID manuel

PT maçã
-catgra nf
-exuso A mãe está a preparar uma tarte de maçã.
EN apple
FR pomme
ES manzana
DE apfel
CN ??
NU 205
DOM alimento
BT
POF macieira
Def fruto redondo e duro que pode ter casca vermelha, verde claro ou amarela
IMG Maça.jpg
GI 4
IM 4
ID manuel

PT pêra
-catgra nf
-exuso gosto muito de sumo de pêra.
EN pear
FR poire
ES pera
DE birne
CN ?
NU 206
DOM alimento
BT
POF pereira
Def fruto doce e sumarento que tem uma base arredondada ficando mais fina no em cima
IMG Pêra.jpg
GI 4
IM 4
ID manuel
```

Neste contexto, desenvolva um Filtro de Texto com o Flex para ler um todas as entradas do dicionário contidos num destes ficheiros do dicionauro e:

a) criar uma página HTML[3] com cada palavra-chave em Português (PT) associada às respetivas traduções para inglês (EN), francês (FR), espanhol (ES) e alemão (DE);

b) criar uma página HTML[4] com cada palavra-chave em Português (PT) associada à respetiva definição e aos exemplos de uso;

c) criar uma hierarquia de termos referentes ao domínio identificado pelo descritor `DOM`, sendo que uma palavra-chave é uma subclasse da palavra identificada no descritor `POF` (*Part-Of*) e é também uma superclasse das palavras identificadas no descritores `PART`;

d) produza uma lista com todos as entradas que sejam substantivos femininos (categoria gramatical denotada por "`nf`").

2.7 Processamento de Ontologias em OWL

O uso de ontologias em informática, para representar rigorosamente o conhecimento de determinado domínio, é cada vez mais comum e importante. Basicamente, uma ontologia é formada por um conjunto de *conceitos* (também chamados *classes*) que pertencem a esse domínio e por um conjunto de relações (hierárquicas, ou não-hierárquicas) entre esses conceitos.
Para o efeito foi definido há alguns anos um dialeto XML, chamado OWL[5], que permite escrever ontologias em formato legível pelo humano e processável pelo computador. A dita linguagem de anotação é muito completa e complexa, porém neste contexto apenas nos interessa considerar as anotações que indicam as relações entre as classes de modo a poder desenhar-se o grafo que descreve a ontologia em causa.
Para perceber o problema analise com cuidado o fragmento de uma ontologia escrita em OWL que se mostra abaixo e que descreve duas relações, `receives` e `owns`, a primeira que liga a classe `Laundry` com a classe `Order` e a segunda que liga a classe `Client` com a classe já referida `Order`.

```
<ObjectPropertyDomain>
    <ObjectProperty IRI="#receives"/>
    <Class IRI="#Laundry"/>
</ObjectPropertyDomain>
<ObjectPropertyRange>
    <Annotation>
        <AnnotationProperty abbreviatedIRI="owl:backwardCompatibleWith"/>
        <IRI>#Laundry</IRI>
    </Annotation>
    <ObjectProperty IRI="#receives"/>
    <Class IRI="#Order"/>
</ObjectPropertyRange>

<ObjectPropertyDomain>
    <ObjectProperty IRI="#owns"/>
    <Class IRI="#Client"/>
</ObjectPropertyDomain>
<ObjectPropertyRange>
    <Annotation>
        <AnnotationProperty abbreviatedIRI="owl:backwardCompatibleWith"/>
        <IRI>#Client</IRI>
    </Annotation>
    <ObjectProperty IRI="#owns"/>
    <Class IRI="#Order"/>
</ObjectPropertyRange>
```

[3]Um Dicionário de Traduções.
[4]Um Dicionário de Significados.
[5]do inglês *Ontology Web Language*.

por sua vez a declaração a seguir exemplificada diz que a classe `Type` tem uma propriedade `material` cujo valor (a instanciar posteriormente) será do tipo `string`.

```
<DataPropertyDomain>
    <DataProperty IRI="#material"/>
    <Class IRI="#Type"/>
</DataPropertyDomain>
<DataPropertyRange>
    <DataProperty IRI="#material"/>
    <Datatype abbreviatedIRI="xsd:string"/>
</DataPropertyRange>
```

por fim, mostra-se abaixo um outro exemplo em que se definem ligações hierárquicas entre duas classes (ou conceitos), em que se diz que `Emigracao` é uma subclasse de `Evento` e que `Emigrante` é uma subclasse de `Pessoa`.

```
<SubClassOf>
    <Class IRI="Emigracao"/>
    <Class IRI="Evento"/>
</SubClassOf>
<SubClassOf>
    <Class IRI="Emigrante"/>
    <Class IRI="Pessoa"/>
</SubClassOf>
```

Neste contexto, desenvolva um Filtro de Texto com o Flex para ler uma ontologia descrita em OWL e para desenhar um grafo que ligue os conceitos entre si e os ligue também ao tipo das suas propriedades, recorrendo para isso á linguagem Dotty e ao sistema de construção de grafos Dot.

2.8 Validação de Referências em LaTeX

Como sabe, uma das facilidades do sistema de anotação para estruturação/formatação de documentos LaTeX é a referência (usando o comando '\ref{etiqueta}') a qualquer ambiente dentro do documento (capítulo, secção, figura, equação, definição, etc.) através da adição de etiquetas (através do comando '\label{etiqueta}') a cada um desses ambientes. Como se percebe do que acaba de ser dito, é fundamental que as referências indiquem etiquetas que foram definidas algures no texto e é também importante que as etiquetas definidas sejam referidas pelo menos uma vez no texto.
Neste contexto, desenvolva um Filtro de Texto com o Flex para ler um programa documento escrito em LaTeX e:

a) indique todas as referências que usem etiquetas não definidas;

b) indique todas as etiquetas que foram definidas e nunca referidas, identificando o respetivo contexto;

c) indique todas as figuras, tabelas e listagens que não estejam associadas a etiquetas.