Processamento de Linguagens – LEI (3^{o}ano)
Trabalho Prático n^{o} 1 (FLex)

Ano lectivo 13/14

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/processadores de texto*, como o Flex

Para o efeito, esta folha contém vários enunciados, dos quais deverá resolver pelo menos um. 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 (3 alunos), em data a marcar.
O **relatório** a elaborar, deve ser claro e, além do respectivo enunciado, da descrição do problema, das decisões que lideraram o desenho e a implementação, deverá conter exemplos de utilização (textos fontes diversos e respectivo resultado produzido). Como é de tradição, o relatório será escrito em LaTeX.

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 Processador de Texto para fazer o reconhecimento dos padrões identificados e proceder à transformação pretendida, com recurso ao Gerador Flex.

2.1 Processador de ABC (formato textual para música)

Estude o formato ABC (`abcnotation.com`, `abcplus.sourceforge.net` – ABC Plus project) e instale algum dos seus processadores (Exemplo EasyABC, abcmidi ou abcm2ps)

Para uma documentação completa sobre o formato ABC, veja: `abcplus.sourceforge.net/abcplus_en-2012-03-30.zip`

Construa um preprocessador em Flex que "reconheça"um texto ABC e que faça pequenas alterações:

1. Conte quantos compassos tem a música;
2. Alterar o instrumento associado a uma voz;
3. Ponha o volume de uma voz a 100% e todas as outras a 20% (para ajudar a aprender essa voz);
4. Ponha o volume de uma voz a 20% e todas as outras a 100% (para ajudar a treinar essa voz);

Para detalhes e dúvidas → contactar `jj@di`

2.2 Pré-processador para LaTeX ou HTML

Desenvolver um documento em LaTeX ou mesmo em HTML é uma actividade inteligente e intelectualmente interessante enquanto estruturante das ideias e sistematizante dos processos.

Porém o acto de editar o respectivo documento é por vezes fastidioso devido ao peso das marcas (as *tags*) que tem de ser inseridas para anotar o texto com indicações de forma, conteúdo ou formato.

Por isso apareceram editores sensíveis ao contexto que sabendo que se está a escrever um documento LaTeX ou HTML nos facilitam a vida inserindo as ditas marcas, ou anotações.

Uma alternativa mais simples mas também muito frequente é permitir o uso de anotações mais leves e simples (até de preferência independentes do tipo de documento final) e depois recorrer ao pré-processamento para substituir essa notação ligeira, abreviada, pelas marcas finais correctas.

Este é o caso do conhecido PPP[1], desenvolvido há alguns anos por José Carlos Ramalho, ou mesmo do mais actual e não menos conhecido sistema Wiki para construção interactiva e via web de páginas HTML.

O que se lhe pede neste trabalho é que, depois de investigar os tais pré-processadores PPP, Wiki, ou outro análogo, e de estudar bem a linguagem HTML, especifique uma sua linguagem de anotação para abreviar a escrita de:

- **formatação**: *negrito*, *itálico*, *sublinhado*;
- vários **níveis de *títulos***;
- **listas de tópicos (items)** *não-numerados*, *numerados* ou tipo *entradas de um dicionário* (preparar o pré-processador para aceitar listas aninhadas);
- **inclusão de imagens** (permitir associar à imagem uma legenda e alguns atributos: centrada, largura máxima, ...);
- **inclusão e formatação de tabelas** (permitir associar à tabela alguns atributos: centrada, largura máxima, etc;
- todos os outros que achar necessário ou a sua imaginação vislumbrar (pode, por exemplo, recorrer a bibliotecas CSS para embelezar o resultado: *twitter bootstrap* ou *zurb foundation*).

Deve, depois e recorrendo à ferramenta Flex, criar um processador que transforme a sua notação em LaTeX ou HTML[2].

[1] Consultar detalhes no manual da linguagem em `http://www3.di.uminho.pt/~jcr/TUTORIAL/tutorial-ppp.html`.

[2] O mais interessante mesmo é que fosse possível escolher a saída final no inicio do próprio texto a pré-processar.

2.3 Processamento da Wikipedia

A Wikipedia é actualmente uma base de conhecimento online disponível em várias línguas (entre as quais, o português). Como tal, e por estar disponível num formato passível de transformação (XML) muitas são as ferramentas que trabalham sobre a Wikipedia para produzir diferentes resultados.

Neste trabalho pretende-se que desenvolva em Flex um filtro para estruturar, num site HTML, um conjunto de informações extraídas da Wikipedia.

Para isso poderá descarregar a versão completa da wikipedia (exemplo wikipedia PT) ou exportar uma ou mais páginas usando o Special Export disponível em `http://pt.wikipedia.org/wiki/Especial:Exportar` (ou `http://en.wikipedia.org/wiki/Special:Export` para a versão inglesa).

Variante1

Estude o seu conteúdo de forma a produzir os seguintes resultados para cada página existente no ficheiro XML obtido:

- Título;
- Autor da última revisão;
- Data da última revisão;
- Nº de links internos (e explicitar quais);
- Nº de links externos (e explicitar quais);
- Nº de secções (e explicitar quais).

Estude ainda a possibilidade de gerar uma página HTML para cada página existente no ficheiro com o respectivo conteúdo original, colocando como cabeçalho as informações referidas acima (escolha um *layout* do seu gosto para a página HTML).

Variante2

Construa dicionários bilingues através de:

Usando a wikipedia completa, extrair os termos verbetes (ponto de entrada) e a sua tradução para a língua destino pretendida; junte também outra informação que achar relevante (exemplo as categorias).

Construa um dicionário LaTeXcom essa informação.

Variante3

Escolha algumas Infoboxes e centrando-se só nelas, atravesse a wikipedia extrai-as e gere um conjunto de páginas HTML com a informação relevante que conseguir extrair.

Exemplo processar info-boxes de tipo geográficas – coordenadas:

```
{{geocoordenadas|41_43_07_N_8_18_34_W_type:adm2nd_region:PT-16|41° 43’ N, 8° 19’ W}}
```

ou Municípios

```
{{Info/Município de Portugal|
|município                = Terras de Bouro
|imagem_brasão            = TBR.png
|imagem_geral             = Pacos_do_concelho.jpg
|imagem_localização       = LocalTerrasDeBouro.svg
|região                   = [[Região Norte (Portugal)|Norte]]
```

```
|subregião                = [[Cávado (sub-região)|Cávado]]
|distrito                 = [[Distrito de Braga|Braga]]
|província                = [[Minho (província)|Minho]]
|área                     = 277.5
|população                = 7253
...
```

2.4 Processador de Ficheiros de Genalogia (gedcom)

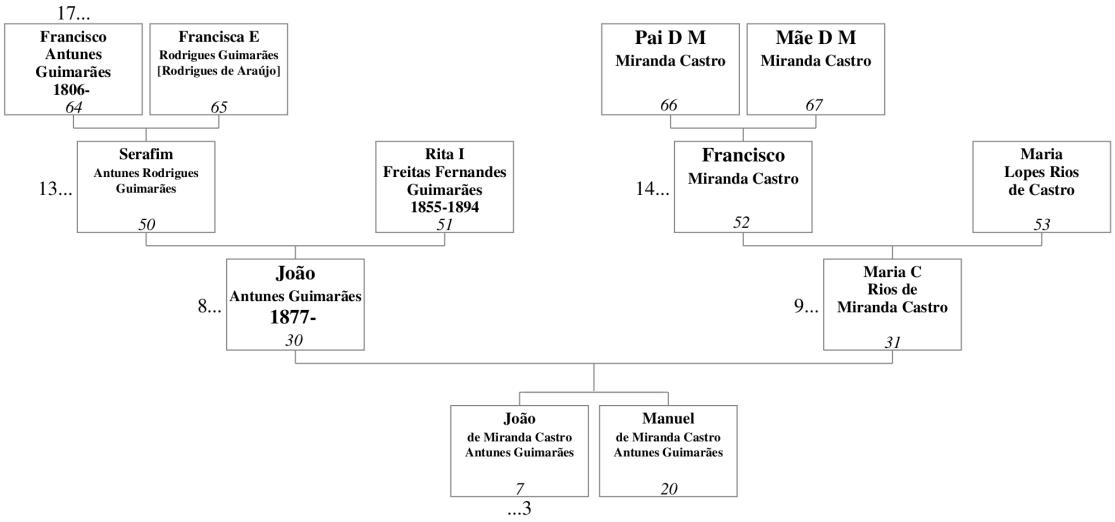
O formato GEDCOM é muito usado para intercambio de dados genealógicos.

```
0 @I1@ INDI
1 NAME Rui Jorge da Costa dos /Reis Borges/
2 GIVN Rui Jorge da Costa dos
2 SURN Reis Borges
1 SEX M
1 BIRT
2 DATE 27 APR 1944
2 PLAC S.Mamede Lisboa
1 FAMS @F2@
1 FAMS @F3@
1 FAMC @F1@
0 @F2@ FAM
1 HUSB @I1@
1 WIFE @I4@
1 CHIL @I7@
1 MARR
2 PLAC Angra do Heroísmo Terceira Açores
0 @F3@ FAM
1 HUSB @I1@
1 WIFE @I5@
1 MARR
2 PLAC Cascais
```

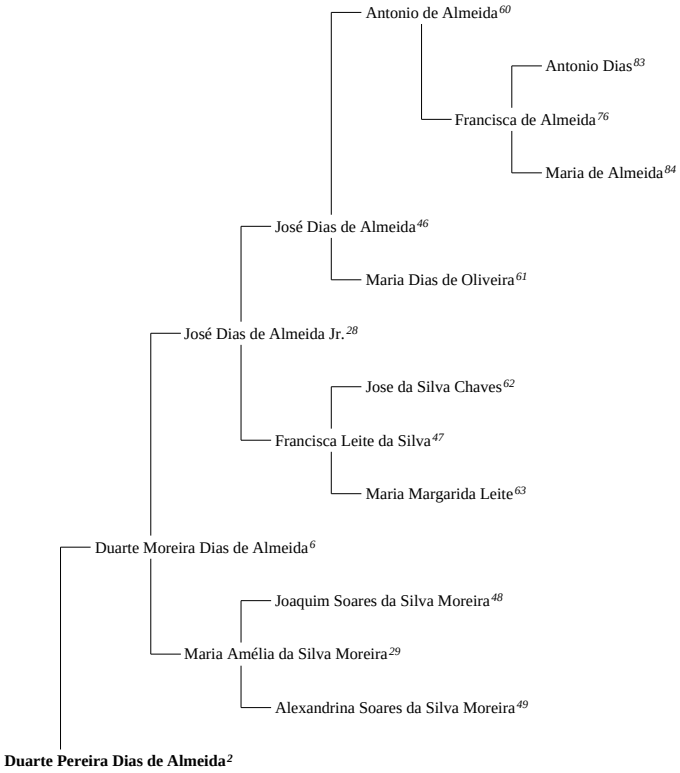
Variante 1 Processe um ficheiro GEDCOM e gere uma página (notação Wiki? Zim? HTML?) com a informação de cada pessoa.

Variante 2 Crie (em LaTeX?) diagramas de família (fihos+casal+pais+avós); ou crie um livro de família.

5. João Antunes Guimarães e Maria Cecília Rios de Miranda Castro



1. ASCENDÊNCIA PATERNA



2.5 Browser/Viewer Web para ontologias em SKOS

O *Simple Knowledge Organization System* (SKOS) é um vocabulário RDF (*Resource Description Framework*) para representar especificações de conhecimento semi-formais, tais como thesauri, taxonomias, sistemas de classificação ou listas finitas de termos, às vezes designadas *vocabulários controlados*. Como o SKOS é baseado em RDF, as especificações SKOS podem ser lidas e interpretadas por máquinas e podem ser trocadas entre aplicações de software.

O SKOS foi concebido com o objetivo de facilitar a migração de modelos organizacionais existentes para a Web Semântica. No entanto, pode ser usado para especificar novos modelos de conhecimento e partilhá-los na Web. Pode ser usado isoladamente ou combinado com outras linguagens mais formais como o OWL (*Ontology Web Language*). Pode também ser usado como ponte entre as linguagens de ontologias como o OWL e as pouco estruturadas ferramentas que suportam a Web social.

Para uma documentação mais abrangente os alunos deverão consultar os documentos oficiais do W3C: `http://www.w3.org/TR/skos-primer/`

Neste projeto, pretende-se que seja desenvolvido um processador que, recebendo um ficheiro SKOS contendo uma especificação de um modelo de conhecimento, crie um conjunto de páginas HTML que permitam uma navegação fácil no modelo.

Listam-se a seguir duas ontologias SKOS que os alunos deverão usar como casos de estudo:

Ontologia das localidades portuguesas : `http://www.di.uminho.pt/~jcr/XML/didac/xmldocs/SKOS/localidades.rdf`;

Ontologia informática da ACM : `http://www.di.uminho.pt/~jcr/XML/didac/xmldocs/SKOS/ACM-SKOSTaxonomy.xml`.

Para mais informações os alunos deverão contactar o docente: José Carlos Ramalho.