Processamento de Linguagens e Compiladores
LCC (3ºano) + MiEFis (4ºano)

Trabalho Prático nº 1 (FLex)

Ano lectivo 20/21

1 Objectivos e Organização

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

- aumentar a experiência de uso do ambiente Linux 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 com base no conceito de regras de produção *Condição-Ação*;
- utilizar o Flex para gerar *filtros de texto em C*.

Para o efeito, esta folha contém 7 enunciados, dos quais deverá resolver um escolhido em função do número do grupo (NGr) usando a fórmula $exe = (NGr \ \% \ 7) \ + \ 1$.
Neste TP que se pretende que seja resolvido rapidamente (2 semanas), aprecia-se a imaginação/criatividade dos grupos ao incluir outros processamentos!

Deve entregar a sua solução **até Domingo dia 15 de Novembro**. O ficheiro com o relatório e a solução deve ter o nome '`plc20TP1GrNN`' — *em breve serão dadas indicações precisas sobre a forma de submissão.*
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.
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.

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.

Para resolver os Enunciados 1 a 4, considere o seguinte contexto:
Para se fazer um estudo sócio-linguístico da forma e conteúdo dos comentários que uma dada notícia de Jornal suscitou, os dados relevantes à análise pretendida devem ser retirados automaticamente de cada ficheiro HTML extraído da versão online desse Jornal, devendo depois ser transformados no formato JSON a seguir mostrado.

```
"commentThread": [
  {
    "id": "STRING",
    "user": "STRING",
    "date": "STRING",
    "timestamp": NA,
    "commentText": "STRING",
    "likes": NUMBER,
    "hasReplies": TRUE/FALSE,
    "numberOfReplies": NUMBER

    "replies": [ ]
  },.......
]
```

Construa então um filtro de texto, recorrendo ao gerador FLex, que realize o processamento explicado, tendo em consideração que as respostas[1] que surjam a um dado comentário devem ser aninhadas, na forma de uma lista, dentro do campo `"replies"` do dito comentário, seguindo evidentemente o mesmo formato apresentado[2].

1 - Transformador Publico2NetLang

Concretize o enunciado genérico acima, considerando como base o ficheiro `http://www4.di.uminho.pt/~prh/Publico/Publico_extraction_portuguese_comments_110.html` que contém os comentários a uma notícia publicada no jornal O Público, o qual deve analisar com todo o cuidado. A ferramenta a desenvolver tem de funcionar bem para outros ficheiros recolhidos do mesmo jornal (na diretoria acima encontra mais exemplos para testar).

2 - Transformador Sol2NetLang

Concretize o enunciado genérico acima, considerando como base o ficheiro `http://www4.di.uminho.pt/~prh/Sol/Sol_extraction_portuguese_comments_145.html` que contém os comentários a uma notícia publicada no jornal Sol, o qual deve analisar com com todo o cuidado. A ferramenta a desenvolver tem de funcionar bem para outros ficheiros recolhidos do mesmo jornal (na diretoria acima encontra mais exemplos para testar).

3 - Transformador DailyExpress2NetLang

Concretize o enunciado genérico acima, considerando como base o ficheiro `http://www4.di.uminho.pt/~prh/DailyExpress/DailyExpress_extraction_english_comments_14.html` que contém os comentários a uma notícia publicada no jornal DailyExpress, o qual deve analisar com com todo o cuidado. A ferramenta a desenvolver tem de funcionar bem para outros ficheiros recolhidos do mesmo jornal (na diretoria acima encontra mais exemplos para testar).

4 - Transformador Metro2NetLang

Concretize o enunciado genérico acima, considerando como base o ficheiro `http://www4.di.uminho.pt/~prh/Metro/Metro_extraction_english_comments_102.html` que contém os comentários a uma notícia publicada no jornal Metro, o qual deve analisar com com todo o cuidado. A ferramenta a desenvolver tem de funcionar bem para outros ficheiros recolhidos do mesmo jornal (na diretoria acima encontra mais exemplos para testar).

[1] Que também são comentários.
[2] Note que deve ser um processo recursivo.

5 - Filtro para gramáticas

Considere o seguinte extracto yacc contido num ficheiro de nome '`f.y`'.

```
....
%union ....
%token ID MKLISTA NULA
%type ....
%%
  lista : MKLISTA '(' ids ')' { ... ação semântica em C }
        | NULA
        ;
  ids : ID ',' ids   { ignorar as constantes char (ex: '!') das ações semânticas }
      | ID           { if ( ... == 'a') { ... } else { ...} }
      ;
%%
... codigo C...
```

a) **Escreva um filtro flex** `extraiGIC` que extraia de '`f.y`' apenas a gramática pura, para ajudar na documentação:

```
lista : MKLISTA '(' ids ')'
      | NULA
      ;
ids : ID ',' ids
    | ID
    ;
```

b) Depois de escrever uma gramática, é preciso construir o analisador léxico associado, sendo fácil esquecer algum dos símbolos terminais.

Para ajudar os mais esquecidos, **escreva um filtro flex** `lexgen` que, dado um texto contendo uma gramática em notação yacc, gere o esqueleto de texto de um analisador léxico para essa gramática.

Este filtro deverá procurar a lista dos símbolos terminais da gramática (tokens) – tanto os expressamente definidos como tokens (`%token ...`), como os terminais que aparecem entre apostrofes nas produções da gramática (exemplo ',').

A execução de `lexgen f.y` (recorde que '`f.y`' é a gramática yacc apresentada acima) deverá gerar algo como se esquematiza abaixo (Note que a palavra `FIXME` no template será para o utilizador depois substituir pelas expressões regulares que definem os terminais (tokens) se podem escrever na sua linguagem concreta):

```
%%
FIXME {return ID;}
FIXME {return MKLISTA;}
FIXME {return NULA;}
[(),] {return yytext[0];}
%%
```

6 - EnameXPro, processador de Enamex

O Reconhecimento de Entidades Nomeadas (em inglês *NER, Named Entities Recognition*) é uma atividade muito complexa que constitui ainda hoje um enorme desafio para os investigadores em PLN (Processamento de Lingua Natural). Existem inclusive concursos internacionais para verificar quem é capaz de desenvolver reconhecedores com maior precisão e acuidade (que marquem corretamente todos os casos de entidades cujo nome surge num dado texto de entrada, em análise).

O problema todo está em criar reconhecedores capazes de identificar os nomes num dado texto e indicar se se trata de uma *pessoa*, *local* (cidade ou país), ou *organização*.

Neste trabalho o que se lhe pede é para pós-processar um ficheiro já criado por um processador NER que anotou o texto de entrada com etiquetas (tags XML) da norma ENAMEX—veja o anexo `'exemplo-Enamex.xml'`—e responder às alíneas seguintes:

1. Criar uma página HTML com todos os nomes de pessoas encontrados (por ordem alfabética e sem repetições).
2. Tal como na alinea anterior, criar uma página com todos os locais identificados indicando se é uma cidade ou país.
3. Ajudar a resolver as indefinições, isto é, os casos em que o processador NER original identificou um nome (palavra começada por maiúscula) mas não conseguiu saber que tipo de entidade estava a ser nomeada (a maioria das vezes porque não era mesmo o nome de uma entidade).

7 - BibTeXPro, Um processador de BibTeX

BibTeX é uma ferramenta de formatação de citações bibliográficas em documentos LaTeX, criada com o objectivo de facilitar a separação da base de dados da bibliografia consultada da sua apresentação no fim do documento LaTeX em edição. BibTeX foi criada por Oren Patashnik e Leslie Lamport em 1985, tendo cada entrada nessa base de dados textual o aspecto que se ilustra a seguir

```
@InProceedings{CPBFH07e,
  author =    {Daniela da Cruz and Maria João Varanda Pereira
               and Mário Béron and Rúben Fonseca and Pedro Rangel Henriques},
  title =     {Comparing Generators for Language-based Tools},
  booktitle = {Proceedings of the 1.st Conference on Compiler
               Related Technologies and Applications, CoRTA'07
               --- Universidade da Beira Interior, Portugal},
  year =      {2007},
  editor =    {},
  month =     {Jul},
  note  =     {}
}
```

De modo a familiarizar-se com o formato do BibTeX poderá consultar o ficheiro `exemplo-utf8.bib` que se anexa e ainda a página oficial do formato referido (`http://www.bibtex.org/`), devendo para já saber que a primeira palavra (logo a seguir ao caracter "@") designa a categoria da referência (havendo em BibTeX pelo menos 14 diferentes).

As tarefas que deverá executar neste trabalho prático são:

a) Analise o documento BibTeX referido acima e faça a contagem das categorias (`phDThesis`, `Misc`, `InProceeding`, etc.), que ocorrem no documento. No final, deverá produzir um documento em formato HTML com o nome das categorias encontradas e respectivas contagens.

b) Complete o processador de modo a filtrar, para cada entrada de cada categoria, a respectiva chave (a 1ªpalavra a seguir à chaveta), autores e título. O resultado final deverá ser incluído no documento HTML gerado na alínea anterior.

c) Crie um índice de autores, que mapeie cada autor nos respectivos registos, de modo a que posteriormente uma ferramenta de procura do Linux possa fazer a pesquisa.

d) Construa um Grafo que mostre, para um dado autor (definido à partida) todos os autores que publicam normalmente com o autor em causa.
Recorrendo à linguagem Dot do GraphViz[3], gere um ficheiro com esse grafo de modo a que possa, posteriormente, usar uma das ferramentas que processam Dot [4] para desenhar o dito grafo de associações de autores.

[3] Disponível em `http://www.graphviz.org`
[4] Disponíveis em `http://www.graphviz.org/Resources.php` ou a ferramenta Web `http://www.webgraphviz.com/`