Processamento de Linguagens I
LESI + LMCC (3º ano)

Exame de 1ª Época – 2ª Chamada

Data: 02 de Julho de 2002
Hora: 09:30

Dispõe de 2:00 horas para realizar este exame

Questão 1: escrita de uma gramática

Para apoio aos analistas informáticos de uma *software-house* (um pouco desactualizada), pretende-se desenvolver uma linguagem que os ajude a descrever o modelo de dados do sistema de informação (SI) que estão a analisar.
Para isso a linguagem deve começar por permitir identificar o SI e a versão (número e data) em causa. Depois, devem ser descritas as entidades; cada uma, além do nome, é definida por um conjunto de atributos (cada qual com seu nome e tipo). Então, descrevem-se as relações (binárias) que ligam entre si essas entidades; para cada relação, e além do seu nome, tem de se especificar as duas entidades relacionadas e respectiva cardinalidade (1 ou N). Opcionalmente poderá ser definido um conjunto de atributos para essa relação.
A linguagem possibilitará, ainda, a listagem das operações (zero ou mais) requeridas sobre as entidades identificadas. Cada operação é identificada por um nome e pela sua assinatura, i.é, entidades que fazem parte do domínio e a entidade resultante (co-domínio).
Responda, então, às seguintes alíneas:

a) Construa uma gramática independente de contexto (GIC) que defina a linguagem pretendida, tendo em atenção que a gramática deve ser LL(1).

b) Mostre que alterações faria à gramática caso não fosse forçado a garantir a condição LL(1), mas em vez disso lhe solicitassem que diminuísse o número de símbolos não-terminais e de produções.

Questão 2: Gramáticas e Linguagens

Analise atentamente a seguinte GIC, cujo Símbolo Inicial é `Tabuleiro` e em que todos os símbolos escritos em minúsculas são pseudo-terminais, as palavras-reservadas são escritas em maiúsculas e os sinais estão entre apostrofes.

```
Tabuleiro -->  Cabec Entradas
Cabec     -->  &
Cabec     -->  TAB id
Entradas  -->  Entrada LstEnts
LstEnts   -->  '.'
           |   ';' Entrada LstEnt
Entrada   -->  '<' Lin ',' Col '>' Valor
Valor     -->  id
Valor     -->  int
Valor     -->  VAZIO
Lin       -->  LIN  int
Col       -->  COL  int
```

Responda, então, às alíneas seguintes:

a) Explique por palavras suas qual é a linguagem gerada por essa GIC.

b) Dê um exemplo de uma frase válida dessa linguagem, apresentando a respectiva árvore de derivação.

c) Prove que a gramática apresentada é LL(1).

d) Recorrendo à linguagem de programação C, escreva as rotinas dum *Parser Recursivo Descendente* correspondentes ao reconhecimento dos símbolos não-terminais `Cabec` e `Valor` e do símbolos terminais (um só procedimento para todos).

Questão 3: filtros de texto

Usando o gerador de programas flex descreva um **filtro de texto** para resolver cada um dos seguintes problemas:

a) Pretende-se um filtro que copie para um ficheiro de saída todos os números do texto com probabilidade de serem um número de telefone nacional (segundo a actual norma portuguesa, 9 digitos começados por 2, 9, 8 ou 7), ou internacional (começados por "00" ou por "+" e seguidos de pelo menos 6 dígitos) e para outro ficheiro de saída os números candidatos a serem preços em euros (qualquer valor maior ou igual a 0 na parte inteira, e 2 dígitos obrigatórios na parte decimal).

b) Pretende-se um processador de texto que conte os *nomes próprios* e os *acrónimos/siglas* que ocorrem (provavelmente) num dado texto. Para efeitos de contagem, consideram-se *nomes próprios* as palavras começadas por uma letra maiúscula e continuadas por letras minúsculas e consideram-se *acrónimos/siglas* as palavras formadas só por letras maiúsculas. Para identificar possíveis erros de classificação, confundindo nomes próprios com palavras capitalizadas devido a convenções ortográficas, pretende-se que o seu processador conte também todas as palavras começadas por maiúscula no início de frases, isto é, após os sinais de pontuação (ponto final, ponto de interrogação, ou ponto de exclamação).

Questão 4: Parser Recursivo Descendente

Considere a seguinte gramática:

```
SExp  -->  S  '.'
S     -->  pal
S     -->  num
S     -->  '('  L  ')'
L     -->  S CL
CL    -->  &
CL    -->  L
```

a) Verifique se a gramática é LR(0), construindo as respectivas tabelas de parsing *bottom-up*.

b) Especifique um analisador sintáctico para as frases da linguagem gerada por essa gramática, usando o yacc.

c) Acrescente à especificação yacc anterior acções semânticas para contar o número de palavras encontradas e calcular o somatório dos números durante o reconhecimento das frases dessa linguagem, devendo escrever esses valores no fim do parsing.