Construção de Compiladores 1 - 2018.1 - Profs. Mário César San Felice (e Helena Caseli, Murilo Naldi, Daniel Lucrédio)
Tópico 03 - Introdução à Análise Sintática - Lista de Exercícios

1. Qual é a função da etapa de análise sintática no processo de tradução de um compilador?
2. Faça a análise sintática do seguinte ditado: "A fé move montanhas" (pesquise um pouco de gramática da língua portuguesa)
3. Qual a importância da teoria da computação (linguagens e autômatos) para a análise sintática?
4. Porque utiliza-se gramáticas livres de contexto em compiladores, para análise sintática, e não outras classes de gramáticas?
5. Qual a diferença entre uma árvore de análise sintática e uma árvore de sintaxe abstrata?
6. Defina uma gramática não ambígua para expressões aritméticas onde os operandos são constantes numéricas e variáveis. Os operadores são os operadores aritméticos comuns: *,/,+,-, além do exponencial ^. A precedência e associatividade dos operadores, porém, é o contrário do usual, ou seja: + e – tem precedência sobre * e / que tem precedência sobre ^. Todos os operadores são associativos à direita, com exceção do exponencial "^" que é associativo à esquerda.
7. Modifique a gramática do exercício anterior de forma que expoentes só possam envolver constantes numéricas, e nunca variáveis
8. Considere a seguinte gramática

```
E : E op E | (E) | NUM
```

Modifique essa gramática para que ela passe a utilizar uma notação pré-fixada, ou seja, ao invés de escrever 2 + 3 * 4, escreveríamos + 2 * 3 4. A gramática resultante precisa de parêntesis? A gramática resultante é ambígua? Justifique suas respostas.

9. Considere as seguintes regras gramaticais:

```
comando : comando ';' listaComandos
comando : comando ';' 'if' '(' expr ')' 'then' comando
expr : expr '+' termo | expr '-' termo | termo
termo : termo '*' fator | termo '/' fator | fator
fator : fator '.' VAR | VAR
comando : 'while' '(' expr ')' comando 'endwhile'
listacomandos : '{' comando '}'
```

Elimine toda recursividade à esquerda

10. Considere as seguintes regras gramaticais

```
comandoCondicao
  :  'SE' expressaoRelacional 'ENTAO' comando 'FIMSE'
  |  'SE' expressaoRelacional 'ENTAO' comando 'SENAO' comando 'FIMSE'
  ;
```

Fatore à esquerda, caso necessário.

11. Considere as seguintes regras gramaticais

```
lexp : atomo | lista
atomo : numero | identificador
lista : ( lexpseq )
lexpseq : lexpseq lexp | lexp
```

Remova a recursividade à esquerda e modifique as regras de forma a empregar EBNF

12. Desenhe os diagramas sintáticos para as regras da gramática dos exercícios 9, 10 e 11 (com e sem EBNF)
13. Dada a gramática a seguir

```
S : ( L ) | a
L : L , S | S
```

a) Elimine a recursividade à esquerda.
b) Desenhe os grafos sintáticos correspondentes.
c) Construa os procedimentos recursivos para os grafos sintáticos construídos na letra b) bem como o programa principal, utilizando pseudo-código.

14. Construa os procedimentos recursivos e o programa principal, usando pseudo-código, da gramática do exercício 9 (já sem recursividade à esquerda)
15. Construa os procedimentos recursivos e o programa principal, usando pseudo-código, da gramática do exercício 10 (já fatorada à esquerda)
16. Construa os procedimentos recursivos e o programa principal, usando pseudo-código, da gramática do exercício 11 (já sem recursividade à esquerda e usando EBNF)