Universidade Católica de Pernambuco
Centro de Ciências e Tecnologia
Ciência da Computação
Linguagens Formais
Prof.: Eduardo Oliveira

Lista 3

1. Diga se a gramática G2 dada gera as palavras abaixo. Construa uma derivação e uma árvore de derivação para as palavras que a gramática puder gerar.

G2 = ({E, N}, {+, x, 0, 1}, P2, E)

P2:
E → N
| E + E
| E * E
| (E)
N → 0
| 1
| 0N
| 1N

   a. ((100*101)+(10+101))

   b. (1*1+1)*((10+101)+11)

2. Considere a gramática livre de contexto em notação BNF abaixo. Assuma que o não-terminal inicial é <assign>. Os seguintes exemplos conseguem demonstrar ambigüidade (DEE e DED)?

```
<assign> ::= <id> = <expr>

<id> ::= A | B | C

<expr> ::= <expr> “+” <expr>
         | <expr> “*” <expr>
         | <expr>
         | <id>
```

   a. A = (B+C) * (A+C)

   b. B = (B*B) + (A*C)

   c. C = B+B * A+C

3. Considere a gramática livre de contexto em notação BNF abaixo. Assuma que o não-terminal inicial é `<program>`.

```
G = ({program, stm_list, stmt, var, expression},
     {begin, end, ;, =, +, -, A, B, C}, P, program)

P = { <program>      →  begin <stmt_list> end

      <stmt_list>    →  <stmt>
                     |  <stmt> ; <stmt_list>

      <stmt>         →  <var> = <expression>

      <var>          →  A   |   B   |   C

      <expression>   →  <var> + <var>
                     |  <var> - <var>
                     |  <var>   }
```

   a. Dê um exemplo de **palavra** que possa ser gerada pela gramática acima e que tenha ao menos uma expressão.

4. Considere a gramática livre de contexto em notação BNF abaixo. Assuma que o não-terminal inicial é `<programa>`.

```
<programa> ::= <lista_decl>

<lista_decl> ::= <declaracao> <lista_decl>
               | <declaracao>

<declaracao> ::= <tipo> <nome> “;”
               | “func” <nome> “()” “{” <comando> “}”

<tipo> ::= “int” | “char”

<comando> ::= <nome> “=” <expr> “;”
            | “return” <expr> “;”

<expr> ::= <expr> “+” <expr>
         | <expr> “*” <expr>
         | <nome>

<nome> ::= “aux” | “temp” | “x”
```

   a. Dê um exemplo de **palavra** que possa ser gerada pela gramática acima e que tenha ao menos um comando.
   b. Mostre uma **árvore de derivação** para a palavra.
   c. Prove que a gramática acima é **ambígua**.

5. Crie uma gramática livre de contexto em notação BNF para gerar uma seqüência de expressões numéricas separadas por ponto-e-vírgula. Considere que os números tenham um só dígito. A gramática deve ser ambígua. Exemplo de palavras que ela deve gerar: “1+2*3”, “1+2; 3; 4+3+2”, etc.

6. Construa uma gramática livre de contexto, usando a notação BNF, para especificar a sintaxe da linguagem de programação descrita informalmente abaixo:
    a. Um **programa** é composto de um ou mais **procedimentos**.
    b. Cada **procedimento** é definido assim:
        - Inicia com a palavra “`proc`”
        - Depois vem um **identificador**
        - Depois vem um **bloco**
    c. Um **identificador** pode ser qualquer palavra formada apenas por **letras** (uma ou mais).
    d. Assuma que as **letras** podem ir apenas de ‘*a*’ a ‘*d*’.
    e. Um **bloco**
        - Inicia com “`[[`“
        - Depois vem uma seqüência de **comandos**, possivelmente vazia
        - Termina com “`]]`”
    f. Um **comando** é definido assim:
        - Ou a palavra “`var`” seguida de um **identificador** seguido de “`;`” (declaração de variável).
        - Ou um **identificador** seguido de “`<-`” seguido de uma **expressão** seguida de “`;`” (atribuição).
        - Ou um **identificador** seguido da palavra “`();`” (chamada de procedimento).
    g. Uma **expressão** é definida assim:
        - Uma **expressão** seguida de “`*`” seguida de outra **expressão**.
        - Ou “-“ seguido de uma **expressão**
        - Ou simplesmente a palavra “41”.
        - Ou simplesmente um **identificador**.

7. Dê um exemplo de **palavra** que possa ser gerada pela gramática acima e que tenha ao menos um comando. Mostre uma **árvore de derivação** para a palavra.