UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"

Faculdade de Ciências e Tecnologia

Departamento de Matemática e Computação

Bacharelado em Ciência da Computação

Disciplina: Compiladores. 2015-2

Especificação da Linguagem Pascal Simplificado - LALG

Sintaxe do PS

Programa e Bloco

1. <programa> ::=
   program <identificador> **;**
   <bloco>**.**
2. <bloco> ::=
   [<parte de declarações de variáveis>]
   [<parte de declarações de sub-rotinas>]
   <comando composto>

Declarações

3. <parte de declarações de variáveis> ::=
   <declaração de variáveis> {**;** <declaração de variáveis>}**;**
4. <declaração de variáveis>::= <tipo> <lista de identificadores>
5. <lista de identificadores> ::= <identificador> {**,** <identificador>}
6. <parte de declarações de subrotinas> ::= {declaração de procedimento> **;**}
7. <declaração de procedimento> ::=
   procedure <identificador> [<parâmetros formais>] **;** <bloco>
8. <parâmetros formais> ::=
   (<seção de parâmetros formais> { **;** <seção de parâmetros formais>} **)**
9. <seção de parâmetros formais> ::=
   [**var**] <lista de identificadores> **:** <identificador>

OBS: Todos os identificadores devem ser declarados antes de serem utilizados. Isto implica a impossibilidade de se utilizar recursão múltipla entre subrotinas quando elas não estão declaradas uma dentro da outra. São considerados **identificadores pré-declarados** os identificadores de tipo simples **int** e **boolean**, os identificadores de procedimentos **read** e **write** (usados respectivamente para leitura e impressão) e os identificadores de constantes **true** e **false**.

Comandos

10. <comando composto ::=
    begin <comando> { **;** <comando>} **end**
11. <comando> ::=
    <atribuição>
    | <chamada de procedimento>
    | <comando composto>
    | <comando condicional 1>
    | <comando repetitivo 1>
12. <atribuição>::= <variável> **:=** <expressão>
13. <chamada de procedimento> ::= <identificador> [**(** <lista de expressões> **)**]
14. <comando condicional 1> ::=
    if <expressão> **then** <comando>
    [**else** <comando>]
15. <comando repetitivo 1> ::=
    while <expressão> **do** <comando>

OBS: Os comandos de entrada (read) e saída (write) estão incluídos nas chamadas de procedimentos. Supomos que existe um arquivo padrão de entrada contendo apenas números lidos pelo comando read(v1,v2, ...vn), em que v1, v2, ... vn são variáveis inteiras. Os pormenores sobre o formato do arquivo de entrada serão ignorados. Analogamente, o comando write(e1,e2, ...en) imprimirá os valores das expressões inteiras e1, e2, ...em num arquivo padrão de saída.

Expressões

16. <expressão> ::= <expressão simples> [<relação> <expressão simples>]
17. <relação> ::= **=** | **<>** | **<** | **<=** | **>=** | **>**
18. <expressão simples> ::= [**+** | **-**] <termo> {(**+** | **-** | **or**) <termo>}
19. <termo> ::= <fator> {(***** | **div** | **and**) <fator> }
20. <fator> ::=
    <variavel>
    | <número>
    | **(** <expressão> **)**
    | **not** <fator>
21. <variável> ::= <identificador> | <identificador> **[** <expressão> **]**
22. <lista de expressões> ::= <expressão> {**,** <expressão>}

OBS: As expressões podem ser **inteiros** ou **booleanas** e a distinção deverá ser feita pelo compilador. Note que as constantes **true** e **false** são consideradas identificadores pré-declarados.

Números e Identificadores

23. <número> ::= <dígito> {<dígito>}
24. <dígito> ::= **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9**
25. <identificador> ::= <letra> {<letra> | <dígito>}
26. <letra> ::= **_** | **a** -**z** |**A** - **Z**

Obs 1: As produções 24 a 27 estão incluídas para tornar a gramática completa. Será mais conveniente, entretanto, tratar números e identificadores como símbolos terminais da gramática. Será conveniente também impor o número máximo de caracteres que pode entrar na formação de números e identificadores.

EBNF:

[α] = significa um item opcional

{α} = repetição da cadeia α zero ou mais vezes

α | β = α ou β devem ser escolhidos

Não terminais aparecem entre < e > e terminais aparecem em negrito.

Obs 2: o comentário de código na linguagem LALG inicia-se por **{** e finaliza-se com **}** para múltiplas linhas e **//** no caso de uma única linha.