Antonio Carlos de Oliveira Bezerra
Gabriel Antônio

Gramática BNF

Declarações

<booleano> ::= "True" | "False"

<tipo> ::= "Integer" | "Boolean"

Números e identificadores

<letra > ::= "a" | "b" | "c" | … | "x" | "y" | "z" | "A" | "B" | "C" | … | "X" | "Y" | "Z"

<digito> ::= "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"

<número> ::= <digito> {<digito>}*

<identificador> ::= <letra > (<letra > | <número>)*

Início

<início do programa> ::= begin{ <bloco> }end

Variáveis

<declaração de variável> ::= <tipo> <identificador> = <atribuição>;

<atribuição> ::= <chamada de função> | <booleano> | <número> | <identificador> |
<chamada de operador>

Parâmetros

<params> ::= <tipo> <identificador> (, <params> | <empty>)
<chamada de parâmetros > ::= <identificador> (, <chamada de parâmetros > |<empty>)*

Função

<declaração de função> ::= dfunc <tipo> <identificador> (<params>) { <bloco4> };

<chamada de função> ::= cfunc <identificador> (<chamada de parâmetros >);

Procedimento

<declaração de procedimento> ::= dproc <identificador> (<params>) { <bloco> };

<chamada de procedimento> ::= cproc <identificador> (<chamada de parâmetros >);

Operador

<chamada de operador> ::= <número> <operador aritmético > <chamada de operador2>
| <identificador> <operador aritmético > <chamada de operador2>

<chamada de operador2> ::= <número> <chamada de operador4> |
<identificador> <chamada de operador4>

<chamada de operador3> ::= <chamada de operador2> | <número>

<chamada de operador4> ::= (<operador aritmético > <chamada de operador3>)* | <empty>

Expressão

<operador booleano> ::= == | != | > | < | >= | <=

<operador aritmético > ::= + | - | * | /

<expressão > ::= <identificador> <expressão 2> | <número> <expressão

2> <expressão 2> ::= <operador booleano> <expressão 3>

<expressão 3> ::= <identificador> | <número>

Condicional

<chamada do if> ::= if(<expressão >){<bloco3>} <else_part>

<else_part> ::= else { <bloco3> } | <empty>

<chamada do if2> ::= if(<expressão >){<bloco2>} <else_part2>

<else_part2> ::= else { <bloco2> } | <empty>

Laço

<chamada do while> ::= while(<expressão >){<bloco2>}

Desvio incondicional

<incondicional> ::= continue; | break;

Retorno

<chamada de retorno> ::= return <chamada de retorno2> ;

<chamada de retorno2> ::= <chamada de função> | <identificador> | <número> | <booleano>

Impressão

<chamada de impressão > ::= println (<parâmetros de impressão >) ;

<parâmetros de impressão > ::= <identificador> | <chamada de função> | <chamada de operador> | <booleano> | <número>

Blocos

1- bloco geral

<bloco> ::= <declaração de variável> | <declaração de função> | <declaração de procedimento> | <chamada de função> |<chamada de procedimento> | <chamada de impressão > | <chamada do if> | <chamada do while> | <identificador> |<empty>

2- bloco do while

<bloco2> ::= <declaração de variável> | <chamada de procedimento> | <chamada de função> | <identificador> | <chamada do if2> | <chamada do while> | <incondicional> | <chamada de impressão >

3- bloco do if/else

<bloco3> ::= <declaração de variável> | <chamada de procedimento> | <chamada de função> | <identificador> | <chamada do if> | <chamada do while> | <chamada de impressão >

4- bloco do return(função)

<bloco4>::= <declaração de variável> | <chamada de função> |<chamada de procedimento> | <chamada de impressão > | <chamada do if> | <chamada do while> | <identificador>|<chamada de retorno> |<empty>