S -> {(Declaracao|Bloco_de_comandos)} EOF

Declaracao -> (int|float) {13} id [<-[-]valor {2}] {1} {,id {1} [<-[-]valor {2}] } ;

| (string|char) {14} id {1} [<-valor {2}] {,id {1} [<-valor {2}] } ;

| const id {23} = [- {24}] {21} valor ;

Bloco_de_comandos -> Comando | “{“ {Bloco_de_comandos} “}”

Comando -> id{3} ["[" Expressao{4} "]"] <- [-]{24} expressão{5} ;

| While expressão{6} bloco_de_comandos

| if expressão{6} Bloco_de_comandos [else Bloco_de_comandos]

| ;

| readln “(“ id{3} “)” ;

| (write|writeln) “(“ expressao{,expressão} “)”;

Expressao -> F {25} [([!]= | <[=] | >[=]) {22} F1{7}]

F -> E {26}{ (- | + | ”|””|”) {22} E1{8} }

E -> D {27}{ (* | / | div | mod | &&) {22} D1{9}}

D -> {!}C{10}

C -> B{15} | { (int | float) {16} ”(“ [-]Expressao{11} ”)” }+

B -> A{17} | {“(“ Expressao{18} “)”}+

A -> id{19} ["["expressao{12}"]"] | valor{20}

{1}

registro = pesquisa(id.lex)

Se registro != null então

    ERRO

Senão

    Inserir(reg.lex, registro)

{2}

se valor.tipo != id.tipo então

    ERRO

{3}

Registro = pesquisa(id.lex)

Se registro == null então

ERRO

senão se registro.classe == classe-const então

ERRO

Senão

Id.tipo = registro.tipo

{4}

se id.tipo != string ou expressão.tipo != inteiro então

ERRO

{5}

Se (negativo && reg.tipo != inteiro && reg.tipo != real) || (se expressão.tipo != id.tipo && id.tipo != real && expressão.tipo != inteiro) então

ERRO

{6}

se expressão.tipo != logica então

ERRO

{7}

se (F.tipo == string ou F1.tipo == string) então

se operador != = entao

ERRO

Senão

Expressao.tipo = logica

Senão se F.tipo == “” ou F1.tipo == “” então

ERRO

Senao // tipos inteiros, reais ou caracteres

Se F.tipo == caractere ou F1.tipo == caractere entao

Se F.tipo != F1.tipo então

ERRO

Senao

Expressao.tipo = logica

` Senao

Expressao.tipo = logica

{8}

se operador == || então

se (E.tipo != logica ou E1.tipo != logica) então

ERRO

senao

F.tipo = logica

Senão se (E.tipo != inteiro e E.tipo != real) ou (E1.tipo != inteiro e E1.tipo != real) então

ERRO

Senao se E.tipo == real ou E1.tipo == real então

F.tipo = real

Senao

F.tipo = inteiro

{9}

se operador == && então

se (E.tipo != logica ou D1.tipo != logica) então

ERRO

Senão

E.tipo = logica

Senão se operador == div ou operador == mod então

se (E.tipo != inteiro ou D1.tipo != inteiro) então

ERRO

Senão

E.tipo = inteiro

Senão se (E.tipo != inteiro e E.tipo != real) ou (D1.tipo != inteiro ou D1.tipo != real) então

ERRO

Senao se E.tipo == real ou D1.tipo == real entao

E.tipo = real

Senao

E.tipo = inteiro

{10}

se operador == ! então

    se C.tipo != logica então

        ERRO

    Senão

        D.tipo = logica

Senão

    D.tipo = C.tipo

{11}

se expressão != inteiro e expressão != real então

    ERRO

{12}

se A.tipo != string ou expressao.tipo != inteiro então

    ERRO

Senão A.tipo = caractere

{13}

Se reg.lex == int então

    id.tipo = inteiro

senão

    id.tipo = real

{14}

Se reg.lex == char então

    id.tipo = caractere

Senão

    id.tipo = string

{15}

C.tipo = B.tipo

{16}

Se reg.lex == int então

    C.tipo = inteiro

Senão

    C.tipo = real

{17}

B.tipo = A.tipo

{18}

B.tipo = expressão.tipo

{19}

Registro = pesquisa(id.lex)

Se registro == null então

    ERRO

Senão

    A.tipo = registro.tipo

{20}

A.tipo = valor.tipo

{21}

Inserir(reg.lex, id, classe-const, valor.tipo)

{22}

Operador = reg.token

{23}

Registro = pesquisa(id.lex)

If registro == null

    ERRO

Negativo = false

{24}

Negativo = true

{25}

Expressao.tipo = F.tipo

{26}

F.tipo = E.tipo

{27}

E.tipo = D.tipo