<http://bottlecaps.de/rr/ui>

BNF Comentada

**Identifiquei algumas situações:**

1. A declaração de variáveis como arranjos tem algumas diferenças de quando usamos elas.

inteiro: n := 1024

inteiro: x

x := 512

inteiro: A[10], B[n], C[x+n], G[getTamanho()]

1.1. Podemos ter uma lista de variáveis sendo declaradas como do mesmo tipo:

inteiro: i, A[10]

i = 2

A[i] := 50

A[3] := 5

INDICE := NUM\_INTEIRO | ID

var ::= var VIRGULA ID {: :}

| ID ECOLCHETES INDICE DCOLCHETES {: :}

| ID ECOLCHETES INDICE DCOLCHETES ECOLCHETES INDICE DCOLCHETES {: :}

| ID {: :}

;

Um outro problema: Temos que diferenciar declaração de arranjos do uso de arranjos.

a := A[i] + A[i+1]

No uso de A em expressões, o índice do arranjo pode ser uma expressão -> ... -> fator -> numero | ID (numero -> NUM\_INTEIRO

Podemos tentar melhorar a BNF com base nos problemas que vocês encontraram.

**Mudanças que estou propondo (tenho que incluir as que vocês detectaram):**

programa ::= lista\_declaracoes

lista\_declaracoes ::= lista\_declaracoes declaracao | declaracao

declaracao ::= declaracao\_variaveis | declaracao\_funcao

declaracao\_variaveis ::= tipo : lista\_de\_variaveis

lista\_de\_variaveis ::= lista\_de\_variaveis , variavel | variavel

declaracao\_variavel ::= ID | ID[indice] | ID[indice][indice]

indice ::= ID | NUM\_INTEIRO

tipo :: = INTEIRO | FLUTUANTE

...

variavel ::= ID | ID[expressao] | ID[expressao][expressao] // para cobrirmos: A[i\*n+j]

fator ::= ( expressao ) | variavel | chamada\_funcao | numero // para cobrirmos: a := b + sum(x,y)

programa ::= lista\_declaracoes

lista\_declaracoes ::= lista\_declaracoes declaracao | declaracao

declaracao ::= declaracao\_variaveis | declaracao\_funcao

declaracao\_variaveis ::= tipo : lista\_de\_variaveis

lista\_de\_variaveis ::= lista\_de\_variaveis , var | var

var ::= ID | ID indice

indice ::= indice “[“ expressao “]” | “[“ expressao “]”

tipo ::= INTEIRO **|** FLUTUANTE

declaracao\_funcao ::= funcao\_tipada | funcao\_sem\_tipo

declaracao\_funcao ::= tipo ID "**(**" lista\_parametros "**)**" lista\_statements FIM

funcao\_sem\_tipo ::= ID "**(**" lista\_parametros "**)**" lista\_statements FIM

lista\_parametros ::= lista\_parametros "," parametro

| parametro

| vazio

parametro ::= tipo : ID

| parametro **[ ]bb**

lista\_statements ::= lista\_statements statement

| vazio

<<\*AQUI\*>>

statement ::= expressao

| declaracao\_variavel

| declaracao\_selecao

| declaracao\_iteracao

| declaracao\_atribuicao

| declaracao\_leitura

| declaracao\_escrita

| declaracao\_retorno

| erro

declaracao\_selecao ::= SE expressao ENTAO lista\_statements FIM

| SE expressao ENTAO lista\_statements SENAO lista\_statements FIM

declaracao\_iteracao ::= REPITA lista\_statements ATE expressao

declaracao\_atribuicao ::= var ":=" expressao

declaracao\_leitura ::= LEIA "**(**" ID "**)**"

declaracao\_escrita ::= ESCREVA "**(**" expressao "**)**"

declaracao\_retorno ::= RETORNA "**(**" expressao "**)**"

expressao ::= expressao\_simples

| chamada\_funcao

expressao\_simples ::= unario expressao\_soma operador\_relacional expressao\_soma

| expressao\_soma

unario ::= operador\_soma expressao

operador\_relacional ::= "<" **|** ">" **|** "=" **|** "<>" **|** "<=" **|** ">="

expressao\_soma ::= expressao\_soma operador\_soma termo

| termo

operador\_soma ::= "**+**" **|** "**-**"

termo ::= termo operador\_multiplicacao fator

| fator

operador\_multiplicacao ::= "**\***" **|** "**/**"

fator ::= "**(**" expressao "**)**"

| numero

| ID

fator ::= "**(**" expressao "**)**"

| var

| chamada\_funcao

| numero

| ID

numero ::= NUM\_INTEIRO **|** NUM\_PONTO\_FLUTUANTE **|** NUM\_NOTACAO\_CIENTIFICA

chamadafuncao ::= ID "**(**" lista\_argumentos "**)**"

lista\_argumentos ::= lista\_argumentos "," expressao

| expressao

| vazio