Grupo :

Bruno Albuquerque Brito : 41583469

Clauber Stipkovic : 31243045

Henrique Olivares : 41514181

Rabah Zeineddine : 41502779

MiniAlg é uma linguagem *case sensitive*.

**Exemplo de código correto**

(Precisa ser ajustado de acordo com a gramática)

**programa** correto;

**inteiro** a , b , c ;

**booleano** d , e , f ;

**procedimento** proc **( inteiro** a **)**

**inteiro** a , b , c ;

**booleano** d , e , f ;

a = 1 ;

**se (** a < 1 **)**

**entao**

a = 12 ;

**fimse**

**fimprocedimento**

a = 2 ;

b = 10 ;

c = 11 ;

a = b + c ;

d = **verdadeiro** ;

e = **falso** ;

f = **verdadeiro** ;

**escreva (** b **) ;**

**se (** d **) entao**

a = 20 ;

b = 10 **mul** c ;

c = a **div** b ;

**fimse**

**enquanto (** a > 1 **) faca**

**se (** b > 10 **) entao**

b = 2 ;

**senao**

a = a - 1 ;

**fimse**

**proc (** a , 123 , **verdadeiro ) ;**

/\* Este e um comentaria, sera descartado na analise. \*/

**fimenquanto**

a = ( 123 + a ) **div** 2 ;

**fimprograma**

Um programa MiniAlg deverá estar codificado em um único arquivo fonte, sem fazer referências a detalhes externos a ele.

**ESPECIFICAÇÃO LÉXICA**

* **Comentários**

Aparecem delimitados por “/\*” e “\*/”. Tudo que segue os símbolos“/\*” é ignorado pelo MiniAlg até que se encontre “\*/”.

/\* comentário \*/

* **Identificadores (variáveis)**

Podem ser criados somente com letras de **a** até **z** (maiúsculas ou minúsculas).

Value

companyNumber

12Val /\* inválido \*/

Val12 /\* inválido \*/

Ex$ /\* inválido \*/

company-name /\* inválido \*/

company\_name /\* inválido \*/

* **Tipos de dados**

Numerais : Inteiros (tipo inteiro)

Booleano : **falso** | **verdadeiro** (tipo booleano)

* **Operadores**

+ - div mul < > <> <= >= == =

* **Delimitadores**

**(** **)** , : ;

* **PalavrasReservadas**

**programa se escreva**

**procedimento fimse verdadeiro**

**fimprograma inteiro falso**

**fimprocedimento booleano faca**

**enquanto entao**

**fimenquanto senao**

**Sintaxe**

1. <programa>::=

**programa identificador** **;**

<bloco>

**fimprograma**

2. <bloco>:: =

[ <parte de declarações de variaveis> ]

[ <parte de declarações de funcoes> ]

<commando composto>

**Declarações**

3. <parte de declarações de variaveis>::=

<declaração de variaveis>

<resto de declarações de variaveis>

4. <declaração de variaveis>::=

<tipo> <lista de identificadores> **;**

5. <resto de declaração de variaveis>::=

<declaração de variaveis><resto de declaração de variaveis>

| **“vazio”**

6. <tipo> ::=

**Inteiro** | **booleano**

7. <lista de identificadores> ::=

**Identificador** <resto de identificadores>

8. <resto de identificadores>::=

**, identificador** <resto de identificadores>

| **“vazio”**

9. <parte de declarações de procedimentos>::=

<declaracao de procedimento> <resto de declaracoes de procedimentos>

10. <declaracao de procedimento>::=

**procedimento identificador ( [** <parametros formais> ] **)**

<bloco>

**fimprocedimento**

11. <parametros formais> ::=

<parametro formal> <resto de parametros formais>

12. <parametro formal>::=

<tipo> **identificador**

13. <resto de parametros formais> :: =

**,** <parametron formal> <resto de parametros formais>

**| “vazio”**

14. <resto de declaracoes de procedimentos>::=

<declaracao de procedimento><resto de declaracoes de procedimentos>

| **“vazio”**

**Comandos**

15. <comando composto>::=

<comando> <resto de comandos>

16. <resto de comandos>::=

<comando> <resto de comandos>

| **“vazio”**

17. <comando>::=

<atribuição>

| <chamada de procedimento>

| <comando condicional>

| <comando repetitivo>

| **escreva ( identificador ) ;**

18. <atribuição> ::=

<variável> **=** <expressão> ;

19. <chamada de procedimento>::=

**identificador (** [ <lista de parâmetros > ] **) ;**

20. <lista de parâmetros> :: =

[<parâmetro de procedimento>] <resto de parâmetros de procedimento >

21. <parâmetro de procedimento> :: =

**identificador** | <numero> | <bool>

22. <resto de parâmetros de procedimento >::=

**,** <parâmetro de procedimento> <resto de parâmetros de procedimento >

| **“vazio”**

23. <comando condicional>::=

**se (** <expressão> **) entao** <comando composto> [ **senao** <comando composto> ]

**fimse**

24. <comando repetitivo> ::=

**enquanto (** <expressâo> **) faca**

<comando composto>

**fimenquanto**

**Expressões**

25. <expressão> ::=

<expressão simples> <resto de expressão>

26. <resto de expressão> ::=

<relação> <expressão simples>

| **“vazio”**

27. <relação>::=

**==** | **<>** | **<** | **<=** | **>=** |  **>**

28. <expressão simples> ::=

[ **+** | **-** ] <termo> <resto de expressão simples>

29. <termo> ::=

<fator> <resto de termo>

30. <fator> ::=

<variável>

| <numero>

| <bool>

| **(** <expressão simples> **)**

31. <resto de termo> ::=

( **mul** | **div** ) <fator>

| **"vazio"**

32. <resto de expressão simples>::=

[ **+** | **-** ] <termo> | **“vazio”**

33. <variável> :: = **identificador**

**Números e Identificadores**

34. <bool>::=

**verdadeiro** | **falso**

35. <numero>::=

**+** | **-** **dígitos**

**EBNF:**

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

α | β = α ou β devem ser escolhidos

[α] = Opcional.

(α | β) = α ou β devem ser escolhidos

Não terminais aparecem entre **** e ****.