#### Universidade Federal de Alagoas

#### Instituto de Computação

Compiladores - 2017.2

# Hapais: gramática e analisador sintático

Autor 1
Dayvson SALES

Autor 2 Warley VITAL

# Introdução

O analisador sintático a ser utilizado será o Preditivo Tabular. Nos próximos capítulos serão apresentados as gramáticas e a tabela gerada, além das saídas dos exemplos de acordo com a especificação.

## Gramática Livre de Contexto

```
1. S = 'defmod', 'id', 'do', Global DeclFuncoes
   'endmod'
2. Global = DeclVar
3. DeclFuncoes = DeclFuncoes Funcao
4. DeclFuncoes = Funcao
5. Funcao = 'def' TipoFuncao 'id' '('
   Parametro ')' 'do' Instrucao 'end'
6. Return = 'return' Exp ';'
7. TipoFuncao = Tipo '[' ']'
8. TipoFuncao = Tipo
9. Tipo = 'int'
10. Tipo = 'void'
11. Tipo = 'str'
12. Tipo = 'real'
13. Tipo = 'bool'
14. Tipo = 'char'
15. TipoFixo = Tipo
16. TipoFixo = Tipo '[' ']'
17. TipoFixo = Tipo '[' CteInt ']'
18. Parametro = 'id' ':' TipoFuncao
19. Parametro = Parametro ',' 'id' ':'
   TipoFuncao
20. Parametro = epsilon
21. Instrucao = Instrucao Comando
22. Instrucao = Comando
23. Instrucao = epsilon
24. Comando = Rep
25. Comando = DoUntil
26. Comando = Until
27. Comando = When
28. Comando = TipoFixo 'id' AtrOuNao ';'
29. AtrOuNao = '=' Exp
30. AtrOuNao = ',
31. Comando = 'id' RArr AtrOuFunc ';'
32. AtrOuFunc = '=' Exp
33. AtrOuFunc = '(' ParFunc ')'
34. Comando = Return
35. When = 'when' '(' Exp ')' 'do' Instrucao
   Otherwise 'end'
```

```
36. Otherwise = 'otherwise' Instrucao
37. Otherwise = ''
38. Until = 'until', '(' Exp ')', 'do'
   Instrucao 'end'
39. DoUntil = 'do' Instrucao 'till' '(' Exp
   ')' 'end'
40. Rep = 'rep' '(' VarControl ',' Exp ',' Id
    '=' Exp')' 'do' Instrucao 'end'
41. VarControl = 'id' ':' Tipo '=' Exp
42. Atrib = Id '=' Exp ';'
43. Id = 'id' Arr
44. DeclVar = DeclVar LDeclVar
45. DeclVar = LDeclVar
46. LDeclVar = TipoFixo Atrib ';'
47. LDeclVar = TipoFixo ';'
48. DeclVar = epsilon
49. Arr = '[' Exp']'
50. Arr = epsilon
51. RArr = '[' LRArr ']'
52. RArr = epsilon
53. LRArr = Exp
54. LRArr = epsilon
55. ParFunc = ParFunc ',' Exp
56. ParFunc = Exp
57. ParFunc = epsilon
# ordem: - ~, * /, + -, $, < > <= >=, ==
   !=, &&, ||
58. Exp = Exp '||' ExpBoolAnd
59. Exp = ExpBoolAnd
60. ExpBoolAnd = ExpBoolAnd '&&' Exprdois
61. ExprBooldAnd = Exprdois
62. ExpRdois = ExpRdois 'opr2' ExpRum
63. ExpRdois = ExpRum
64. ExpRum = ExpRum 'opr1' ExpConcat
65. ExpRum = ExpConcat
66. ExpConcat = ExpConcat '$' Expa
```

- 67. ExpConcat = Expa
- 68. Expa = Expa 'opa' Expm
- 69. Expa = Expm
- 70. Expm = Expm 'opm' Expu 71. Expm = Expu
- 72. Expu = Unario Fa
  73. Expu = Fa
- 74. Unario = ',~'
- 75. Unario = '-'

- 76. Fa = '(' Exp ')'
- 77. Fa = 'id'
- 78. Fa = 'id' '(' ParFunc ')'
- 79. Fa = 'id' RArr
- 80. Fa = Cte
- 81. Cte = 'cteI'

- 82. Cte = 'cteR' 83. Cte = 'cteStr' 84. Cte = 'cteChar' 85. Cte = CteBool
- 86. CteBool = 'true' 87. CteBool = 'false'

## Gramática LL(1)

```
1. S = 'defmod' 'id' 'do' Global DeclFuncoes
     'endmod'
2. Global = RDeclVar
3. DeclFuncoes = Funcao DeclFuncoes
4. DeclFuncoes = epsilon
5. Funcao = 'def' TipoFuncao 'id' '('
    Parametro ')' 'do' Instrucao 'end'
6. Return = 'return' Exp';'
7. TipoFuncao = Tipo FArr
8. FArr = '[' RFArr ']'
9. RFArr = Exp
10. RFArr = epsilon
11. FArr = epsilon
12. Tipo = 'int'
13. Tipo = 'void'
14. Tipo = 'str'
15. Tipo = 'real'
16. Tipo = 'bool'
17. Tipo = 'char'
18. TipoFixo = Tipo FArr
19. Parametro = 'id' ':' TipoFixo Parametro
20. Parametro = ',' 'id' ':' TipoFixo
21. Parametro = epsilon
22. Instrucao = Comando RInstrucao
23. RInstrucao = epsilon
24. Comando = Rep Comando
25. Comando = DoUntil Comando
26. Comando = Until Comando
27. Comando = When Comando
28. Comando = TipoFixo 'id' ComandoA ';'
    Comando
29. Comando = 'id' RArr ComandoX Comando
30. ComandoX = = Exp ';'
31. ComandoX = '(' ParFunc ')' ';'
32. ComandoA = Exp
33. ComandoA = epsilon
34. Comando = Return
35. Comando = epsilon
36. Atrib = Id = Exp ';'
37. Atrib = epsilon
38. Id = 'id' Arr
39. RDeclVar = DeclVar RDeclVar
40. RDeclVar = epsilon
41. DeclVar = TipoFixo Atrib ';'
42. Arr = '[' Exp ']'
43. Arr = epsilon
44. RArr = '[' LRArr ']'
```

```
45. RArr = epsilon
46. LRArr = Exp
47. LRArr = epsilon
48. When = 'when' '(' Exp ')' 'do' Otherwise
49. Otherwise = Instrucao OtherwiseR
50. OtherwiseR = 'otherwise' Instrucao
51. OtherwiseR = epsilon
52. Until = 'until', '(' Exp ')', 'do'
   Instrucao 'end'
53. DoUntil = 'do' Instrucao 'till' '(' Exp
   ')' 'end'
54. Rep = 'rep' '(' VarControl ',' Exp ','
   Id = Exp ')' 'do' Instrucao 'end'
55. VarControl = 'id' ':' Tipo = Exp
    ParFunc = RParFunc
57. ParFunc = epsilon
58. RParFunc = Exp TRParFunc
59. TRParFunc = ',' Exp TParF
    TRParFunc = ',' Exp TParFunc
60.
    TRParFunc = epsilon
     TParFunc = epsilon
62. Exp = ExpBoolAnd RExp
63. RExp = '||' ExpBoolAnd RExp
64.
     RExp = epsilon
65.
    ExpBoolAnd = ExpRDois RExpBoolAnd
    RExpBoolAnd = '&&' ExpRDois RExpBoolAnd
66.
67.
    RExpBoolAnd = epsilon
68. ExpRDois = ExpRUm RExpRDois
69. RExpRDois = 'opr2' ExpRUm
70. RExpRDois = epsilon
71. ExpRUm = ExpConcat RExpRUm
72. RExpRUm = 'opr1' ExpConcat RExpRUm
73. RExpRUm = epsilon
74. ExpConcat = Expa RExpConcat
75. RExpConcat = '$' Expa RExpConcat
76.
    RExpConcat = epsilon
77.
    Expa = Expm RExpa
78.
    RExpa = 'opa' Expm RExpa
79.
    RExpa = epsilon
    Expm = ExpU RExpm
    RExpm = 'opm' ExpU RExpm
    RExpm = epsilon
83. ExpU = Unario Fa
84. ExpU = Fa
85. Unario = '~'
86. Unario = '-'
87. Fa = '(' Exp ')'
```

```
88. Fa = 'id' RFa
```

- 89. RFa = '(' ParFunc ')'
- 90. RFa = RArr
- 91. Fa = Cte
- 92. Cte = 'cteI'
- 93. Cte = 'cteR'

- 94. Cte = 'cteStr'
- 95. Cte = 'cteChar'
- 96. Cte = CteBool
- 97. CteBool = 'true' 98. CteBool = 'false'

# Tabela do Analisador Preditivo

Em anexo a este documento está a tabela.

## Saída dos exemplos

#### 5.1 Hello World

```
Start:
                                                              Instrucao = Comando RInstrucao
     defmod AloMundo do
                                                              Comando = 'id' RArr ComandoX
   1
      def void main() do
                                                                 Comando
                print("Alo mundo!");
                                                                  [003, 005] (0025,
                                                                                            id) {
                                                                     print}
   4
        end
                                                              RArr = epsilon
   5
      endmod
                                                              ComandoX = '(' ParFunc ')' ';'
                                                                  [003, 010] (0027,
   1 defmod AloMundo do
                                                                      {(}
          S = 'defmod', 'id', 'do', Global
                                                              ParFunc = RParFunc
              DeclFuncoes 'endmod'
                                                              RParFunc = Exp TRParFunc
              [001, 001] (0001,
                                     defmod) {
                                                              Exp = ExpBoolAnd RExp
                  defmod}
                                                              ExpBoolAnd = ExpRDois RExpBoolAnd
                                                              ExpRDois = ExpRUm RExpRDois
              [001, 008] (0000,
                                         id) {
                  AloMundo}
                                                              ExpRUm = ExpConcat RExpRUm
              [001, 017] (0003,
                                         do) {
                                                              ExpConcat = Expa RExpConcat
                  do}
                                                              Expa = Expm RExpa
                                                             Expm = ExpU RExpm
        def void main() do
                                                             ExpU = Fa
                                                             Fa = Cte
                                                              Cte = 'cteStr'
          Global = RDeclVar
          RDeclVar = epsilon
                                                                 [003, 011] (0019,
                                                                                         cteStr) {
          DeclFuncoes = Funcao DeclFuncoes
                                                                     Alo mundo!}
          Funcao = 'def' TipoFuncao 'id' '('
                                                              RExpm = epsilon
              Parametro ')' 'do' Instrucao '
                                                              RExpa = epsilon
                                                              RExpConcat = epsilon
              [002, 003] (0002,
                                                              RExpRUm = epsilon
                  def }
                                                              RExpRDois = epsilon
          TipoFuncao = Tipo FArr
                                                              RExpBoolAnd = epsilon
          Tipo = 'void'
                                                             RExp = epsilon
              [002, 007] (0011,
                                       void) {
                                                              TRParFunc = epsilon
                                                                  [003, 023] (0028,
                                                                                              ))
                  void}
          FArr = epsilon
                                                                     {)}
                                                                  [003, 024] (0023,
              [002, 012] (0000,
                                         id) {
                                                                                              ;)
                                                                      {;}
                  main}
              [002, 016] (0027,
                                          ()
                  {(}
          Parametro = epsilon
              [002, 017] (0028,
                                          ))
                                                              Comando = epsilon
                  {)}
                                                              RInstrucao = epsilon
              [002, 019] (0003,
                                         do) {
                                                                  [004, 003] (0004,
                                                                                           end) {
                  do}
                                                                      end }
                print("Alo mundo!");
   3
                                                         endmod
```

#### 5.2 Fibonacci

```
Start:
   1 defmod Fibonacci do
        def void fib(limite : int) do
          int[limite+1] arr;
          arr[0] = 0;
   5
          arr[1] = 1;
   6
          int i = 0;
   7
   8
        when(limite <= 0) do
   9
               print("0");
  10
          end
  11
  12
          until(i < limite) do
  13
            when (i > 2) do
  14
              arr[i] = arr[i-1] + arr[i-2];
  15
  16
  17
            print(arr[i]);
  18
            when(i != limite-1) do
  19
            print(", ");
  20
  21
            end
  22
  23
           i = i + 1;
  24
         end
  25
        end
  26
  27
  28
        def void main() do
  29
          int limite = read();
  30
          fib(limite);
  31
        end
  32
  33
     endmod
     defmod Fibonacci do
          S = 'defmod', 'id', 'do', Global
              DeclFuncoes 'endmod'
              [001, 001] (0001,
                                     defmod) {
                  defmod}
              [001, 008] (0000,
                                         id) {
                  Fibonacci}
               [001, 018] (0003,
                                         do) {
                  do }
        def void fib(limite : int) do
          Global = RDeclVar
          RDeclVar = epsilon
          DeclFuncoes = Funcao DeclFuncoes
          Funcao = 'def' TipoFuncao 'id' '('
              Parametro ')' 'do' Instrucao '
              [002, 003] (0002,
                                        def) {
                  def}
```

```
TipoFuncao = Tipo FArr
Tipo = 'void'
    [002, 007] (0011,
                            void) {
       void}
FArr = epsilon
    [002, 012] (0000,
                             id) {
       fib}
    [002, 015] (0027,
                                ()
       {(}
Parametro = 'id' ':' TipoFixo
   Parametro
    [002, 016] (0000,
                              id) {
       limite}
    [002, 023] (0022,
                               :)
       {:}
TipoFixo = Tipo FArr
Tipo = 'int'
    [002, 025] (0012,
                            int) {
       int}
FArr = epsilon
Parametro = epsilon
    [002, 028] (0028,
                              ))
       {)}
    [002, 030] (0003,
                             do) {
       do}
int[limite+1] arr;
Instrucao = Comando RInstrucao
Comando = TipoFixo 'id' ComandoA
   ';' Comando
TipoFixo = Tipo FArr
Tipo = 'int'
    [003, 005] (0012,
                            int) {
       int}
FArr = '[' RFArr ']'
    [003, 008] (0029,
                                [)
RFArr = Exp
Exp = ExpBoolAnd RExp
ExpBoolAnd = ExpRDois RExpBoolAnd
ExpRDois = ExpRUm RExpRDois
ExpRUm = ExpConcat RExpRUm
ExpConcat = Expa RExpConcat
Expa = Expm RExpa
Expm = ExpU RExpm
ExpU = Fa
Fa = 'id' RFa
    [003, 009] (0000,
                             id) {
       limite}
RFa = RArr
RArr = epsilon
RExpm = epsilon
RExpa = 'opa' Expm RExpa
   [003, 015] (0034,
                             opa)
       {+}
```

	Expm = ExpU RExpm	Fa = Cte
	ExpU = Fa	Cte = 'cteI'
	Fa = Cte	[004, 014] (0017, cteI)
	Cte = 'cteI'	{0}
	[003, 016] (0017, cteI)	RExpm = epsilon
	{1}	RExpa = epsilon
	RExpm = epsilon	RExpConcat = epsilon
	RExpa = epsilon	RExpRUm = epsilon
	RExpConcat = epsilon	RExpRDois = epsilon
	RExpRUm = epsilon	RExpBoolAnd = epsilon
	RExpRDois = epsilon	RExp = epsilon
	RExpBoolAnd = epsilon	[004, 015] (0023, ;)
	RExp = epsilon	{;}
	[003, 017] (0030, ])	
	{]}	5 arr[1] = 1;
	[003, 019] (0000, id) {	
	arr}	Comando = 'id' RArr ComandoX
	ComandoA = epsilon	Comando
	[003, 022] (0023, ;)	[005, 005] (0000, id)
	{;}	arr}
	(,)	RArr = '[' LRArr ']'
1	arr[0] = 0;	
4	arrioj - 0;	
		{[}
	Comando = 'id' RArr ComandoX	LRArr = Exp
	Comando	Exp = ExpBoolAnd RExp
	[004, 005] (0000, id) {	ExpBoolAnd = ExpRDois RExpBoolAnd
	arr}	ExpRDois = ExpRUm RExpRDois
	RArr = '[' LRArr ']'	ExpRUm = ExpConcat RExpRUm
	[004, 008] (0029, [)	ExpConcat = Expa RExpConcat
	{[}	Expa = Expm RExpa
	LRArr = Exp	Expm = ExpU RExpm
	Exp = ExpBoolAnd RExp	ExpU = Fa
	ExpBoolAnd = ExpRDois RExpBoolAnd	Fa = Cte
	ExpRDois = ExpRUm RExpRDois	Cte = 'cteI'
		[005, 009] (0017, cteI)
	ExpRUm = ExpConcat RExpRUm	
	ExpConcat = Expa RExpConcat	{1}
	Expa = Expm RExpa	RExpm = epsilon
	Expm = ExpU RExpm	RExpa = epsilon
	ExpU = Fa	RExpConcat = epsilon
	Fa = Cte	RExpRUm = epsilon
	Cte = 'cteI'	RExpRDois = epsilon
	[004, 009] (0017, cteI)	RExpBoolAnd = epsilon
	{0}	RExp = epsilon
	RExpm = epsilon	[005, 010] (0030, ])
	RExpa = epsilon	{1}
	RExpConcat = epsilon	ComandoX = '=' Exp';'
	RExpRUm = epsilon	[005, 012] (0036, =)
	RExpRDois = epsilon	{=}
	•	
	RExpBoolAnd = epsilon	Exp = ExpBoolAnd RExp
	RExp = epsilon	ExpBoolAnd = ExpRDois RExpBoolAnd
	[004, 010] (0030, ])	ExpRDois = ExpRUm RExpRDois
	{]}	ExpRUm = ExpConcat RExpRUm
	ComandoX = '=' Exp';'	ExpConcat = Expa RExpConcat
	[004, 012] (0036, =)	Expa = Expm RExpa
	{=}	Expm = ExpU RExpm
	Exp = ExpBoolAnd RExp	ExpU = Fa
	ExpBoolAnd = ExpRDois RExpBoolAnd	Fa = Cte
	ExpRDois = ExpRUm RExpRDois	Cte = 'cteI'
	ExpRUm = ExpConcat RExpRUm	[005, 014] (0017, cteI)
	ExpConcat = Expa RExpConcat	{1}
	Expa = Expm RExpa	RExpm = epsilon
	Expm = ExpU RExpm	RExpm - epsilon
	ExpU = Fa	RExpConcat = epsilon

```
RExpRUm = epsilon
                                                          RFa = RArr
       RExpRDois = epsilon
                                                          RArr = epsilon
       RExpBoolAnd = epsilon
                                                          RExpm = epsilon
       RExp = epsilon
                                                          RExpa = epsilon
           [005, 015] (0023,
                                                          RExpConcat = epsilon
                                      ;)
                                                          RExpRUm = 'opr1' ExpConcat RExpRUm
               {;}
                                                              [008, 013] (0037,
                                                                                       opr1)
       int i = 0;
                                                                  {<=}
                                                          ExpConcat = Expa RExpConcat
                                                          Expa = Expm RExpa
       Comando = TipoFixo 'id' ComandoA
           ';' Comando
                                                          Expm = ExpU RExpm
       TipoFixo = Tipo FArr
                                                          ExpU = Fa
       Tipo = 'int'
                                                          Fa = Cte
                                                          Cte = 'cteI'
           [006, 005] (0012,
                                    int) {
                                                              [008, 016] (0017,
                                                                                       cteI)
              int}
       FArr = epsilon
                                                                 {0}
                                                          RExpm = epsilon
           [006, 009] (0000,
                                     id) {
                                                          RExpa = epsilon
              i}
       ComandoA = '=' Exp
                                                          RExpConcat = epsilon
           [006, 011] (0036,
                                                          RExpRUm = epsilon
                                       =)
               {=}
                                                          RExpRDois = epsilon
       Exp = ExpBoolAnd RExp
                                                          RExpBoolAnd = epsilon
       ExpBoolAnd = ExpRDois RExpBoolAnd
                                                          RExp = epsilon
       ExpRDois = ExpRUm RExpRDois
                                                              [008, 017] (0028,
                                                                                         ))
       ExpRUm = ExpConcat RExpRUm
                                                                  {)}
                                                              [008, 019] (0003,
       ExpConcat = Expa RExpConcat
                                                                                         do) {
       Expa = Expm RExpa
                                                                  do}
       Expm = ExpU RExpm
       ExpU = Fa
                                                                print("0");
       Fa = Cte
       Cte = 'cteI'
                                                          Otherwise = Instrucao OtherwiseR
           [006, 013] (0017,
                                                          Instrucao = Comando RInstrucao
                                    cteI)
                                                          Comando = 'id' RArr ComandoX
               {0}
       RExpm = epsilon
                                                              Comando
       RExpa = epsilon
                                                              [009, 006] (0025,
                                                                                         id) {
       RExpConcat = epsilon
                                                                 print}
                                                          RArr = epsilon
       RExpRUm = epsilon
       RExpRDois = epsilon
                                                          ComandoX = '(' ParFunc ')' ';'
       RExpBoolAnd = epsilon
                                                              [009, 011] (0027,
       RExp = epsilon
                                                                  {(}
           [006, 014] (0023,
                                                          ParFunc = RParFunc
                                       ;)
               {;}
                                                          RParFunc = Exp TRParFunc
                                                          Exp = ExpBoolAnd RExp
8
     when(limite <= 0) do
                                                          ExpBoolAnd = ExpRDois RExpBoolAnd
                                                          ExpRDois = ExpRUm RExpRDois
       Comando = When Comando
                                                          ExpRUm = ExpConcat RExpRUm
       When = 'when' '(' Exp ')' 'do'
                                                          ExpConcat = Expa RExpConcat
          Otherwise 'end'
                                                          Expa = Expm RExpa
           [008, 002] (0008,
                                    when) {
                                                          Expm = ExpU RExpm
              when}
                                                          ExpU = Fa
           [008, 006] (0027,
                                                          Fa = Cte
                                       ()
               {(}
                                                          Cte = 'cteStr'
       Exp = ExpBoolAnd RExp
                                                              [009, 012] (0019,
                                                                                     cteStr)
       ExpBoolAnd = ExpRDois RExpBoolAnd
                                                                  {0}
       ExpRDois = ExpRUm RExpRDois
                                                          RExpm = epsilon
                                                          RExpa = epsilon
       ExpRUm = ExpConcat RExpRUm
       ExpConcat = Expa RExpConcat
                                                          RExpConcat = epsilon
       Expa = Expm RExpa
                                                          RExpRUm = epsilon
       Expm = ExpU RExpm
                                                          RExpRDois = epsilon
       ExpU = Fa
                                                          RExpBoolAnd = epsilon
       Fa = 'id' RFa
                                                          RExp = epsilon
           [008, 007] (0000,
                                     id) {
                                                          TRParFunc = epsilon
               limite}
```

```
[009, 015] (0028,
                                       ))
               {)}
                                                           Instrucao = Comando RInstrucao
            [009, 016] (0023,
                                                           Comando = When Comando
                                       ;)
               {;}
                                                           When = 'when' '(' Exp')' 'do'
                                                              Otherwise 'end'
10
                                                               [013, 007] (0008,
                                                                                       when) {
        end
                                                                  when}
        Comando = epsilon
                                                               [013, 012] (0027,
                                                                                           ()
        RInstrucao = epsilon
                                                                  {(}
        OtherwiseR = epsilon
                                                           Exp = ExpBoolAnd RExp
                                                           ExpBoolAnd = ExpRDois RExpBoolAnd
            [010, 005] (0004,
                                     end) {
                                                           ExpRDois = ExpRUm RExpRDois
                end }
                                                           ExpRUm = ExpConcat RExpRUm
12
        until(i < limite) do</pre>
                                                           ExpConcat = Expa RExpConcat
                                                           Expa = Expm RExpa
        Comando = Until Comando
                                                           Expm = ExpU RExpm
        Until = 'until', '(' Exp ')', 'do'
                                                           ExpU = Fa
           Instrucao 'end'
                                                           Fa = 'id' RFa
            [012, 005] (0006,
                                                               [013, 013] (0000,
                                   until) {
                                                                                        id) {
               until}
                                                                  i}
            [012, 010] (0027,
                                        ()
                                                           RFa = RArr
               {(}
                                                           RArr = epsilon
        Exp = ExpBoolAnd RExp
                                                           RExpm = epsilon
        ExpBoolAnd = ExpRDois RExpBoolAnd
                                                           RExpa = epsilon
        ExpRDois = ExpRUm RExpRDois
                                                           RExpConcat = epsilon
        ExpRUm = ExpConcat RExpRUm
                                                           RExpRUm = 'opr1', ExpConcat RExpRUm
        ExpConcat = Expa RExpConcat
                                                               [013, 015] (0037,
                                                                                       opr1)
        Expa = Expm RExpa
                                                                  {>}
                                                           ExpConcat = Expa RExpConcat
        Expm = ExpU RExpm
                                                           Expa = Expm RExpa
        ExpU = Fa
        Fa = 'id' RFa
                                                           Expm = ExpU RExpm
            [012, 011] (0000,
                                     id) {
                                                           ExpU = Fa
                                                           Fa = Cte
               i}
        RFa = RArr
                                                           Cte = 'cteI'
        RArr = epsilon
                                                               [013, 017] (0017,
                                                                                       cteI)
        RExpm = epsilon
                                                                  {2}
        RExpa = epsilon
                                                           RExpm = epsilon
                                                           RExpa = epsilon
        RExpConcat = epsilon
        RExpRUm = 'opr1', ExpConcat RExpRUm
                                                           RExpConcat = epsilon
            [012, 013] (0037,
                                    opr1)
                                                           RExpRUm = epsilon
                                                           RExpRDois = epsilon
               {<}
        ExpConcat = Expa RExpConcat
                                                           RExpBoolAnd = epsilon
        Expa = Expm RExpa
                                                           RExp = epsilon
        Expm = ExpU RExpm
                                                               [013, 018] (0028,
                                                                                         ))
        ExpU = Fa
                                                                  {)}
        Fa = 'id' RFa
                                                                                         do) {
                                                               [013, 020] (0003,
           [012, 015] (0000,
                                      id) {
                                                                  dol
               limite}
        RFa = RArr
                                                  14
                                                               arr[i] = arr[i-1] + arr[i-2];
        RArr = epsilon
        RExpm = epsilon
                                                           Otherwise = Instrucao OtherwiseR
        RExpa = epsilon
                                                           Instrucao = Comando RInstrucao
                                                           Comando = 'id' RArr ComandoX
        RExpConcat = epsilon
        RExpRUm = epsilon
                                                              Comando
        RExpRDois = epsilon
                                                               [014, 009] (0000,
                                                                                         id) {
        RExpBoolAnd = epsilon
                                                                  arr}
                                                           RArr = '[' LRArr ']'
        RExp = epsilon
                                                              [014, 012] (0029,
            [012, 021] (0028,
                                      ))
                                                                                           [)
               {)}
                                                                  {[}
            [012, 023] (0003,
                                      do) {
                                                           LRArr = Exp
               do}
                                                           Exp = ExpBoolAnd RExp
                                                           ExpBoolAnd = ExpRDois RExpBoolAnd
```

ExpRDois = ExpRUm RExpRDois

13

when (i > 2) do

	1		
ExpRUm = ExpConcat RExpRUm		RExpConcat = epsilon	
ExpConcat = Expa RExpConcat		RExpRUm = epsilon	
Expa = Expm RExpa		RExpRDois = epsilon	
Expm = ExpU RExpm		RExpBoolAnd = epsilon	
ExpU = Fa		RExp = epsilon	
_			
Fa = 'id' RFa		[014, 025] (0030,	,
[014, 013] (0000, id)	1	{]}	
i}		RExpm = epsilon	
RFa = RArr		RExpa = 'opa' Expm RExpa	
RArr = epsilon		[014, 027] (0034, opa)	)
RExpm = epsilon		{+}	
RExpa = epsilon		Expm = ExpU RExpm	
RExpConcat = epsilon		ExpU = Fa	
RExpRUm = epsilon		Fa = 'id' RFa	
RExpRDois = epsilon			) {
RExpBoolAnd = epsilon		arr}	, .
RExp = epsilon		RFa = RArr	
[014, 014] (0030, ])		RArr = '[' LRArr ']'	
{]}		[014, 032] (0029,	)
ComandoX = '=' Exp ';'		{[}	
[014, 016] (0036, =)		LRArr = Exp	
{=}		Exp = ExpBoolAnd RExp	
Exp = ExpBoolAnd RExp		ExpBoolAnd = ExpRDois RExpBoolAnd	d
ExpBoolAnd = ExpRDois RExpBoolAnd		ExpRDois = ExpRUm RExpRDois	
ExpRDois = ExpRUm RExpRDois		ExpRUm = ExpConcat RExpRUm	
ExpRUm = ExpConcat RExpRUm		ExpConcat = Expa RExpConcat	
ExpConcat = Expa RExpConcat		Expa = Expm RExpa	
Expa = Expm RExpa		Expm = ExpU RExpm	
Expm = ExpU RExpm		ExpU = Fa	
ExpU = Fa		Fa = 'id' RFa	
Fa = 'id' RFa		[014, 033] (0000, id)	) {
[014, 018] (0000, id)	{	i}	
arr}		RFa = RArr	
RFa = RArr		RArr = epsilon	
RArr = '[' LRArr ']'		RExpm = epsilon	
[014, 021] (0029, [)		RExpa = 'opa' Expm RExpa	
{[}		[014, 034] (0034, opa	)
LRArr = Exp		{-}	,
Exp = ExpBoolAnd RExp		Expm = ExpU RExpm	
ExpBoolAnd = ExpRDois RExpBoolAnd		ExpU = Fa	
ExpRDois = ExpRUm RExpRDois		Fa = Cte	
ExpRUm = ExpConcat RExpRUm		Cte = 'cteI'	
ExpConcat = Expa RExpConcat		[014, 035] (0017, ctel	)
Expa = Expm RExpa		{2}	
Expm = ExpU RExpm		RExpm = epsilon	
ExpU = Fa		RExpa = epsilon	
Fa = 'id' RFa		RExpConcat = epsilon	
[014, 022] (0000, id)	{	RExpRUm = epsilon	
i}	-	RExpRDois = epsilon	
RFa = RArr		RExpBoolAnd = epsilon	
RArr = epsilon		RExp = epsilon	
RExpm = epsilon		[014, 036] (0030,	`
			,
RExpa = 'opa' Expm RExpa		{]}	
[014, 023] (0034, opa)		RExpm = epsilon	
{-}		RExpa = epsilon	
Expm = ExpU RExpm		RExpConcat = epsilon	
ExpU = Fa		RExpRUm = epsilon	
Fa = Cte		RExpRDois = epsilon	
<pre>Cte = 'cteI'</pre>		RExpBoolAnd = epsilon	
[014, 024] (0017, cteI)		RExp = epsilon	
{1}		[014, 037] (0023,	)
RExpm = epsilon		{;}	
RExpa = epsilon			

15	end	RExpBoolAnd = epsilon RExp = epsilon
	Comando = epsilon	TRParFunc = epsilon
	RInstrucao = epsilon	[017, 019] (0028, ))
	OtherwiseR = epsilon	{)}
	_	
	[015, 007] (0004, end) { end}	[017, 020] (0023, ;) {;}
17	<pre>print(arr[i]);</pre>	19 when(i != limite-1) do
	Comando = 'id' RArr ComandoX	Comando = When Comando
	Comando	When = 'when' '(' Exp ')' 'do'
	[017, 007] (0025, id) {	Otherwise 'end'
	<pre>print}</pre>	[019, 007] (0008, when) {
	RArr = epsilon	when}
	<pre>ComandoX = '(' ParFunc ')' ';'</pre>	[019, 011] (0027, ()
	[017, 012] (0027, ()	{(}
	{(}	Exp = ExpBoolAnd RExp
	ParFunc = RParFunc	ExpBoolAnd = ExpRDois RExpBoolAnd
	RParFunc = Exp TRParFunc	ExpRDois = ExpRUm RExpRDois
	Exp = ExpBoolAnd RExp	ExpRUm = ExpConcat RExpRUm
	ExpBoolAnd = ExpRDois RExpBoolAnd	ExpConcat = Expa RExpConcat
	ExpRDois = ExpRUm RExpRDois	Expa = Expm RExpa
	<pre>ExpRUm = ExpConcat RExpRUm</pre>	Expm = ExpU RExpm
	ExpConcat = Expa RExpConcat	ExpU = Fa
	Expa = Expm RExpa	Fa = 'id' RFa
	Expm = ExpU RExpm	[019, 012] (0000, id) {
	ExpU = Fa	i}
	Fa = 'id' RFa	RFa = RArr
	[017, 013] (0000, id) {	RArr = epsilon
	arr}	RExpm = epsilon
	RFa = RArr	RExpa = epsilon
	RArr = '[' LRArr ']'	RExpConcat = epsilon
	[017, 016] (0029, [)	RExpRUm = epsilon
	{[}	RExpRDois = 'opr2' ExpRUm
	LRArr = Exp	[019, 014] (0038, opr2)
	Exp = ExpBoolAnd RExp	{!=}
	ExpBoolAnd = ExpRDois RExpBoolAnd	ExpRUm = ExpConcat RExpRUm
	ExpRDois = ExpRUm RExpRDois	ExpConcat = Expa RExpConcat
	ExpRUm = ExpConcat RExpRUm	Expa = Expm RExpa
	ExpConcat = Expa RExpConcat	Expm = ExpU RExpm ExpU = Fa
	Expa = Expm RExpa Expm = ExpU RExpm	Fa = 'id' RFa
	ExpU = Fa	
	Fa = 'id' RFa	[019, 017] (0000, id) {
	[017, 017] (0000, id) {	RFa = RArr
	i}	RArr = epsilon
	RFa = RArr	RExpm = epsilon
	RArr = epsilon	RExpa = 'opa' Expm RExpa
	RExpm = epsilon	[019, 023] (0034, opa)
	RExpa = epsilon	{-}
	RExpConcat = epsilon	Expm = ExpU RExpm
	RExpRUm = epsilon	
	RExpRDois = epsilon	Fa = Cte
	RExpBoolAnd = epsilon	Cte = 'cteI'
	RExp = epsilon	[019, 024] (0017, cteI)
	[017, 018] (0030, ])	{1}
	{]}	RExpm = epsilon
	RExpm = epsilon	RExpa = epsilon
	RExpa = epsilon	RExpConcat = epsilon
	RExpConcat = epsilon	RExpRUm = epsilon
	RExpRUm = epsilon	RExpBoolAnd = epsilon
	RExpRDois = epsilon	RExp = epsilon

```
[019, 025] (0028,
                                       ))
                                                           ExpBoolAnd = ExpRDois RExpBoolAnd
                                                           ExpRDois = ExpRUm RExpRDois
                {)}
            [019, 027] (0003,
                                       do) {
                                                           ExpRUm = ExpConcat RExpRUm
                                                           ExpConcat = Expa RExpConcat
                do}
                                                           Expa = Expm RExpa
20
                                                           Expm = ExpU RExpm
            print(", ");
                                                           ExpU = Fa
                                                           Fa = 'id' RFa
[023, 011] (0000,
        Otherwise = Instrucao OtherwiseR
        Instrucao = Comando RInstrucao
                                                                                         id) {
        Comando = 'id' RArr ComandoX
           Comando
                                                           RFa = RArr
            [020, 009] (0025,
                                     id) {
                                                           RArr = epsilon
                                                           RExpm = epsilon
               print}
                                                           RExpa = 'opa' Expm RExpa
        RArr = epsilon
        ComandoX = '(' ParFunc ')' ';'
[020, 014] (0027,
                                                               [023, 013] (0034,
                                                                                          opa)
                                        ()
                                                                  {+}
               {(}
                                                           Expm = ExpU RExpm
        ParFunc = RParFunc
                                                           ExpU = Fa
        RParFunc = Exp TRParFunc
                                                           Fa = Cte
        Exp = ExpBoolAnd RExp
                                                           Cte = 'cteI'
        ExpBoolAnd = ExpRDois RExpBoolAnd
                                                               [023, 015] (0017,
        ExpRDois = ExpRUm RExpRDois
                                                                   {1}
        ExpRUm = ExpConcat RExpRUm
                                                           RExpm = epsilon
        ExpConcat = Expa RExpConcat
                                                           RExpa = epsilon
                                                           RExpConcat = epsilon
        Expa = Expm RExpa
                                                           RExpRUm = epsilon
        Expm = ExpU RExpm
                                                           RExpRDois = epsilon
        ExpU = Fa
        Fa = Cte
                                                           RExpBoolAnd = epsilon
        Cte = 'cteStr'
                                                           RExp = epsilon
            [020, 015] (0019, cteStr)
                                                                [023, 016] (0023,
                                                                                            ;)
               {,}
        RExpm = epsilon
        RExpa = epsilon
                                                   24
                                                           end
        RExpConcat = epsilon
        RExpRUm = epsilon
                                                           Comando = epsilon
        RExpRDois = epsilon
                                                           RInstrucao = epsilon
        RExpBoolAnd = epsilon
                                                                [024, 005] (0004,
                                                                                         end) {
        RExp = epsilon
                                                                   end}
        TRParFunc = epsilon
            [020, 019] (0028,
                                       ))
                                                   25
                                                         end
                {)}
            [020, 020] (0023,
                                                           Comando = epsilon
                                        ;)
                {;}
                                                           RInstrucao = epsilon
                                                                [025, 003] (0004,
                                                                                        end) {
21
          end
                                                                   end }
        Comando = epsilon
                                                   28
                                                         def void main() do
        RInstrucao = epsilon
        OtherwiseR = epsilon
                                                           DeclFuncoes = Funcao DeclFuncoes
            [021, 007] (0004,
                                                           Funcao = 'def' TipoFuncao 'id' '('
                                     end) {
                                                               Parametro ')' 'do' Instrucao '
                end}
                                                                [028, 003] (0002,
23
          i = i + 1;
                                                                                          def) {
                                                                   def }
        Comando = 'id' RArr ComandoX
                                                           TipoFuncao = Tipo FArr
                                                           Tipo = 'void'
           Comando
            [023, 007] (0000,
                                                               [028, 007] (0011,
                                       id) {
                                                                                        void) {
               i}
                                                                   void}
        RArr = epsilon
                                                           FArr = epsilon
        ComandoX = '=' Exp ';'
                                                               [028, 012] (0000,
                                                                                         id) {
            [023, 009] (0036,
                                        =)
                                                                   main}
               {=}
                                                                [028, 016] (0027,
                                                                                           ()
        Exp = ExpBoolAnd RExp
                                                                   {(}
```

```
Parametro = epsilon
                                                              Comando = 'id' RArr ComandoX
              [028, 017] (0028,
                                          ))
                                                                 Comando
                                                                  [030, 005] (0000,
                  {)}
                                                                                           id) {
               [028, 019] (0003,
                                         do) {
                                                                     fib}
                  do}
                                                              RArr = epsilon
                                                              ComandoX = '(' ParFunc ')' ';'
  29
          int limite = read();
                                                                  [030, 008] (0027,
                                                                                              ()
                                                                     {(}
          Instrucao = Comando RInstrucao
                                                              ParFunc = RParFunc
          Comando = TipoFixo 'id' ComandoA
                                                              RParFunc = Exp TRParFunc
              ';' Comando
                                                              Exp = ExpBoolAnd RExp
          TipoFixo = Tipo FArr
                                                              ExpBoolAnd = ExpRDois RExpBoolAnd
          Tipo = 'int'
                                                              ExpRDois = ExpRUm RExpRDois
                                                              ExpRUm = ExpConcat RExpRUm
              [029, 005] (0012,
                                       int) {
                                                              ExpConcat = Expa RExpConcat
                  int}
          FArr = epsilon
                                                              Expa = Expm RExpa
              [029, 009] (0000,
                                                              Expm = ExpU RExpm
                                         id) {
                                                              ExpU = Fa
                 limite}
          ComandoA = '=' Exp
                                                              Fa = 'id' RFa
              [029, 016] (0036,
                                                                  [030, 009] (0000,
                                                                                            id) {
                                          =)
                  {=}
                                                                     limite}
          Exp = ExpBoolAnd RExp
                                                              RFa = RArr
          ExpBoolAnd = ExpRDois RExpBoolAnd
                                                              RArr = epsilon
          ExpRDois = ExpRUm RExpRDois
                                                              RExpm = epsilon
                                                              RExpa = epsilon
          ExpRUm = ExpConcat RExpRUm
                                                              RExpConcat = epsilon
          ExpConcat = Expa RExpConcat
          Expa = Expm RExpa
                                                              RExpRUm = epsilon
                                                              RExpRDois = epsilon
          Expm = ExpU RExpm
                                                              RExpBoolAnd = epsilon
          ExpU = Fa
          Fa = 'id' RFa
                                                              RExp = epsilon
              [029, 018] (0024,
                                                              TRParFunc = epsilon
                                         id) {
                                                                  [030, 015] (0028,
                 read}
                                                                                              ))
          RFa = '(' ParFunc ')'
[029, 022] (0027,
                                                                      {)}
                                                                  [030, 016] (0023,
                                           ()
                                                                                              ;)
                  {(}
                                                                      {;}
          ParFunc = epsilon
              [029, 023] (0028,
                                          ))
                                                     31
                                                            end
                  {)}
          RExpm = epsilon
                                                              Comando = epsilon
          RExpa = epsilon
                                                              RInstrucao = epsilon
          RExpConcat = epsilon
                                                                  [031, 003] (0004,
                                                                                            end) {
          RExpRUm = epsilon
                                                                      end}
          RExpRDois = epsilon
          RExpBoolAnd = epsilon
                                                     33 endmod
          RExp = epsilon
              [029, 024] (0023,
                                                              DeclFuncoes = epsilon
                                          ;)
                  {;}
                                                                  [033, 001] (0005,
                                                                                         endmod) {
                                                                      endmod}
  30
          fib(limite);
                                                   Aceito!
                                                    End:
5.3 ShellSort
```

```
Start:
                                                          comando para os tokens
     defmod Shellsort do
                                                      6
                                                             int h = 1;
   1
   2
                                                      7
   3
        def int[] shellsort(valores : int[])
                                                      8
                                                              until(h < n) do
                                                      9
      do
                                                              h = h * 3 + 1;
                                                     10
   5
          int n = len(valores); #-- retorna
                                                     11
      tamanho do array valores, aqui nao eh
                                                     12
                                                             h = h / 3;
```

```
13
                                                            RDeclVar = epsilon
14
        int c;
                                                            DeclFuncoes = Funcao DeclFuncoes
                                                            Funcao = 'def' TipoFuncao 'id' '('
15
        int j;
16
                                                                Parametro ')' 'do' Instrucao '
17
        until(n > 2) do
                                                                end'
18
                                                                 [003, 003] (0002,
                                                                                           def) {
19
          rep (i : int = h, i < n, i = i +
                                                                    def}
   1) do
                                                             TipoFuncao = Tipo FArr
20
            c = valores[i];
                                                             Tipo = 'int'
21
            j = i;
                                                                 [003, 007] (0012,
                                                                                           int) {
22
                                                                    int}
                                                            FArr = '[' RFArr ']'
[003, 010] (0029,
23
            until(j >= h && valores[j - h]
   > c) do
                                                                                             L)
                     valores[j] = valores[j
24
                                                                   {[}
                                                            RFArr = epsilon
   - h];
25
                     j = j - h;
                                                                 [003, 011] (0030,
                                                                                             ])
26
            end
                                                                    {]}
                                                                 [003, 013] (0000,
27
                                                                                            id) {
28
           valores[j] = c;
                                                                    shellsort}
                                                                 [003, 022] (0027,
                                                                                             ()
29
          end
30
                                                                    {(}
31
         h = h / 2;
                                                            Parametro = 'id' ':' TipoFixo
32
                                                                Parametro
33
        end
                                                                 [003, 023] (0000,
                                                                                            id) {
34
                                                                     valores}
35
                                                                 [003, 031] (0022,
        return valores;
                                                                                            :)
36
      end
                                                                    {:}
37
                                                            TipoFixo = Tipo FArr
38
      def void main() do
                                                            Tipo = 'int'
                                                                 [003, 033] (0012,
39
                                                                                           int) {
        int[] valores;
40
                                                                    int}
                                                            FArr = '[' RFArr ']'
41
                                                                 [003, 036] (0029,
42
        until(read() != EOF) do
                                                                                              [)
          valores[] = lastValueRead();
43
                                                                    {[}
                                                            RFArr = epsilon
44
        end
45
                                                                 [003, 037] (0030,
                                                                                             ])
        int[] arr = shellsort(valores);
46
                                                                    {]}
47
                                                            Parametro = epsilon
                                                                 [003, 038] (0028,
48
        rep(i : int = 0, i < len(valores),</pre>
                                                                                            ))
   i = i + 1) do
                                                                     {(}
                                                                 [003, 040] (0003,
49
          print(arr[i]);
                                                                                            do) {
          print("\n");
                                                                     do}
51
        end
52
                                                     5
                                                             int n = len(valores); #-- retorna
53
      end
                                                         tamanho do array valores, aqui nao eh
                                                         comando para os tokens
54
55
   endmod
                                                             Instrucao = Comando RInstrucao
1 defmod Shellsort do
                                                             Comando = TipoFixo 'id' ComandoA
                                                                ';' Comando
        S = 'defmod', 'id', 'do', Global
                                                            TipoFixo = Tipo FArr
            DeclFuncoes 'endmod'
                                                             Tipo = 'int'
            [001, 001] (0001,
                                   defmod) {
                                                                [005, 005] (0012,
                                                                                          int) {
                defmod}
                                                                    int}
             [001, 008] (0000,
                                                            FArr = epsilon
                                       id) {
                                                                [005, 009] (0000,
                Shellsort}
                                                                                           id) {
             [001, 018] (0003,
                                        do) {
                                                                   n}
                                                            ComandoA = '=' Exp
                do}
                                                                [005, 011] (0036,
                                                                                             =)
     def int[] shellsort(valores : int[])
                                                                    {=}
    do
                                                            Exp = ExpBoolAnd RExp
                                                            ExpBoolAnd = ExpRDois RExpBoolAnd
        Global = RDeclVar
                                                             ExpRDois = ExpRUm RExpRDois
```

```
ExpRUm = ExpConcat RExpRUm
                                                   ExpRUm = ExpConcat RExpRUm
ExpConcat = Expa RExpConcat
                                                   ExpConcat = Expa RExpConcat
Expa = Expm RExpa
                                                   Expa = Expm RExpa
Expm = ExpU RExpm
                                                   Expm = ExpU RExpm
ExpU = Fa
                                                   ExpU = Fa
                                                   Fa = Cte
Fa = 'id' RFa
   [005, 013] (0000,
                             id) {
                                                   Cte = 'cteI'
       len}
                                                       [006, 013] (0017,
                                                                                cteI)
RFa = '(' ParFunc ')'
                                                          {1}
    [005, 016] (0027,
                                ()
                                                   RExpm = epsilon
       {(}
                                                   RExpa = epsilon
ParFunc = RParFunc
                                                   RExpConcat = epsilon
RParFunc = Exp TRParFunc
                                                   RExpRUm = epsilon
                                                   RExpRDois = epsilon
Exp = ExpBoolAnd RExp
                                                   RExpBoolAnd = epsilon
ExpBoolAnd = ExpRDois RExpBoolAnd
ExpRDois = ExpRUm RExpRDois
                                                   RExp = epsilon
ExpRUm = ExpConcat RExpRUm
                                                       [006, 014] (0023,
                                                                                  ;)
ExpConcat = Expa RExpConcat
                                                           {;}
Expa = Expm RExpa
Expm = ExpU RExpm
                                           8
                                                   until(h < n) do
ExpU = Fa
Fa = 'id' RFa
                                                   Comando = Until Comando
   [005, 017] (0000,
                              id) {
                                                   Until = 'until' '(' Exp ')' 'do'
       valores}
                                                      Instrucao 'end'
RFa = RArr
                                                       [008, 005] (0006,
                                                                              until) {
RArr = epsilon
                                                           until}
                                                       [008, 010] (0027,
RExpm = epsilon
                                                                                   ()
RExpa = epsilon
                                                          {(}
RExpConcat = epsilon
                                                   Exp = ExpBoolAnd RExp
RExpRUm = epsilon
                                                   ExpBoolAnd = ExpRDois RExpBoolAnd
RExpRDois = epsilon
                                                   ExpRDois = ExpRUm RExpRDois
RExpBoolAnd = epsilon
                                                   ExpRUm = ExpConcat RExpRUm
RExp = epsilon
                                                   ExpConcat = Expa RExpConcat
                                                   Expa = Expm RExpa
TRParFunc = epsilon
    [005, 024] (0028,
                                                   Expm = ExpU RExpm
                               ))
       {)}
                                                   ExpU = Fa
                                                   Fa = 'id' RFa
RExpm = epsilon
RExpa = epsilon
                                                       [008, 011] (0000, id) {
RExpConcat = epsilon
RExpRUm = epsilon
                                                   RFa = RArr
RExpRDois = epsilon
                                                   RArr = epsilon
RExpBoolAnd = epsilon
                                                   RExpm = epsilon
RExp = epsilon
                                                   RExpa = epsilon
    [005, 025] (0023,
                               ;)
                                                   RExpConcat = epsilon
                                                   RExpRUm = 'opr1' ExpConcat RExpRUm
       {;}
                                                       [008, 013] (0037,
                                                                                opr1)
int h = 1;
                                                          {<}
                                                   ExpConcat = Expa RExpConcat
Comando = TipoFixo 'id' ComandoA
                                                   Expa = Expm RExpa
   ';' Comando
                                                   Expm = ExpU RExpm
TipoFixo = Tipo FArr
                                                   ExpU = Fa
Tipo = 'int'
                                                   Fa = 'id' RFa
                                                       [008, 015] (0000,
   [006, 005] (0012,
                                                                                 id) {
                             int) {
       int}
                                                          n}
FArr = epsilon
                                                   RFa = RArr
   [006, 009] (0000,
                             id) {
                                                   RArr = epsilon
                                                   RExpm = epsilon
       h}
ComandoA = '=' Exp
                                                   RExpa = epsilon
    [006, 011] (0036,
                                =)
                                                   RExpConcat = epsilon
       {=}
                                                   RExpRUm = epsilon
Exp = ExpBoolAnd RExp
                                                   RExpRDois = epsilon
                                                   RExpBoolAnd = epsilon
ExpBoolAnd = ExpRDois RExpBoolAnd
                                                   RExp = epsilon
ExpRDois = ExpRUm RExpRDois
```

```
[008, 016] (0028,
                                       ))
                {)}
                                                   12
                                                           h = h / 3;
            [008, 018] (0003,
                                       do) {
                                                            Comando = 'id' RArr ComandoX
                do}
                                                               {\tt Comando}
9
          h = h * 3 + 1;
                                                                [012, 005] (0000,
                                                                                           id) {
                                                                   h}
        Instrucao = Comando RInstrucao
                                                            RArr = epsilon
        Comando = 'id' RArr ComandoX
                                                            ComandoX = '=' Exp ';'
                                                                [012, 007] (0036,
           Comando
                                                                                            =)
            [009, 007] (0000,
                                       id) {
                                                                    {=}
               h}
                                                            Exp = ExpBoolAnd RExp
                                                            ExpBoolAnd = ExpRDois RExpBoolAnd
        RArr = epsilon
        ComandoX = '=' Exp ';'
                                                            ExpRDois = ExpRUm RExpRDois
            [009, 009] (0036,
                                                            ExpRUm = ExpConcat RExpRUm
                                        =)
               {=}
                                                            ExpConcat = Expa RExpConcat
        Exp = ExpBoolAnd RExp
                                                            Expa = Expm RExpa
                                                           Expm = ExpU RExpm
        ExpBoolAnd = ExpRDois RExpBoolAnd
        ExpRDois = ExpRUm RExpRDois
                                                            ExpU = Fa
        ExpRUm = ExpConcat RExpRUm
                                                            Fa = 'id' RFa
        ExpConcat = Expa RExpConcat
                                                                [012, 009] (0000,
                                                                                          id) {
        Expa = Expm RExpa
                                                                   h}
        Expm = ExpU RExpm
                                                            RFa = RArr
        ExpU = Fa
                                                            RArr = epsilon
                                                            RExpm = 'opm' ExpU RExpm
        Fa = 'id' RFa
            [009, 011] (0000,
                                      id) {
                                                                [012, 011] (0035,
                                                                                          opm)
               h}
                                                                   {/}
                                                           ExpU = Fa
        RFa = RArr
        RArr = epsilon
                                                           Fa = Cte
        RExpm = 'opm' ExpU RExpm
                                                            Cte = 'cteI'
            [009, 013] (0035,
                                                                [012, 013] (0017,
                                      opm)
                                                                                         cteI)
                {*}
                                                                  {3}
                                                            RExpm = epsilon
        ExpU = Fa
                                                            RExpa = epsilon
        Fa = Cte
        Cte = 'cteI'
                                                            RExpConcat = epsilon
            [009, 015] (0017,
                                                            RExpRUm = epsilon
                                     cteI)
                                                           RExpRDois = epsilon
               {3}
                                                            RExpBoolAnd = epsilon
        RExpm = epsilon
        RExpa = 'opa' Expm RExpa
                                                            RExp = epsilon
            [009, 017] (0034,
                                                                [012, 014] (0023,
                                                                                          ;)
                                      opa)
               {+}
                                                                    {;}
        Expm = ExpU RExpm
        ExpU = Fa
                                                   14
                                                            int c;
        Fa = Cte
        Cte = 'cteI'
                                                            Comando = TipoFixo 'id' ComandoA
            [009, 019] (0017,
                                                               ';' Comando
                                     cteI)
               {1}
                                                            TipoFixo = Tipo FArr
        RExpm = epsilon
                                                            Tipo = 'int'
        RExpa = epsilon
                                                                [014, 005] (0012,
                                                                                          int) {
        RExpConcat = epsilon
                                                                   int}
                                                            FArr = epsilon
        RExpRUm = epsilon
        RExpRDois = epsilon
                                                                [014, 009] (0000,
                                                                                           id) {
        RExpBoolAnd = epsilon
                                                                   c}
                                                            ComandoA = epsilon
[014, 010] (0023,
        RExp = epsilon
           [009, 020] (0023,
                                        ;)
                                                                                            ;)
                {;}
                                                                   {;}
10
                                                   15
                                                           int j;
        end
        Comando = epsilon
                                                            Comando = TipoFixo 'id' ComandoA
        RInstrucao = epsilon
                                                               ';' Comando
                                                            TipoFixo = Tipo FArr
            [010, 005] (0004,
                                     end) {
                end}
                                                            Tipo = 'int'
```

```
[015, 005] (0012,
                                  int) {
                                                          Rep = 'rep' '(' VarControl ',' Exp
                                                             ',' Id '=' Exp ')' 'do'
               int}
                                                              Instrucao 'end'
        FArr = epsilon
            [015, 009] (0000,
                                                              [019, 007] (0007,
                                     id) {
                                                                                       rep) {
                                                                 rep}
              j}
        ComandoA = epsilon
                                                              [019, 011] (0027,
                                                                                          ()
            [015, 010] (0023,
                                      ;)
                                                                 {(}
                                                          VarControl = 'id' ':' Tipo '=' Exp
                                                              [019, 012] (0000,
                                                                                        id) {
17
        until(n > 2) do
                                                                 i}
                                                              [019, 014] (0022,
                                                                                          :)
        Comando = Until Comando
                                                                 {:}
        Until = 'until' '(' Exp ')' 'do'
                                                          Tipo = 'int'
           Instrucao 'end'
                                                              [019, 016] (0012,
                                                                                       int) {
            [017, 005] (0006,
                                  until) {
                                                                 int}
                                                              [019, 020] (0036,
                                                                                          =)
               until}
            [017, 010] (0027,
                                                                 {=}
                                       ()
               {(}
                                                          Exp = ExpBoolAnd RExp
                                                          ExpBoolAnd = ExpRDois RExpBoolAnd
        Exp = ExpBoolAnd RExp
        ExpBoolAnd = ExpRDois RExpBoolAnd
                                                          ExpRDois = ExpRUm RExpRDois
        ExpRDois = ExpRUm RExpRDois
                                                          ExpRUm = ExpConcat RExpRUm
        ExpRUm = ExpConcat RExpRUm
                                                          ExpConcat = Expa RExpConcat
        ExpConcat = Expa RExpConcat
                                                          Expa = Expm RExpa
        Expa = Expm RExpa
                                                          Expm = ExpU RExpm
        Expm = ExpU RExpm
                                                          ExpU = Fa
        ExpU = Fa
                                                          Fa = 'id' RFa
                                                              [019, 022] (0000,
        Fa = 'id' RFa
                                                                                    id) {
            [017, 011] (0000,
                                     id) {
                                                                 h}
                                                          RFa = RArr
               n}
                                                          RArr = epsilon
        RFa = RArr
        RArr = epsilon
                                                          RExpm = epsilon
        RExpm = epsilon
                                                          RExpa = epsilon
        RExpa = epsilon
                                                          RExpConcat = epsilon
        RExpConcat = epsilon
                                                          RExpRUm = epsilon
        RExpRUm = 'opr1' ExpConcat RExpRUm
                                                          RExpRDois = epsilon
            [017, 013] (0037,
                                                          RExpBoolAnd = epsilon
                                    opr1)
               {>}
                                                          RExp = epsilon
        ExpConcat = Expa RExpConcat
                                                              [019, 023] (0021,
                                                                                          ,)
        Expa = Expm RExpa
                                                                  {,}
        Expm = ExpU RExpm
                                                          Exp = ExpBoolAnd RExp
        ExpU = Fa
                                                          ExpBoolAnd = ExpRDois RExpBoolAnd
        Fa = Cte
                                                          ExpRDois = ExpRUm RExpRDois
        Cte = 'cteI'
                                                          ExpRUm = ExpConcat RExpRUm
           [017, 015] (0017,
                                  cteI)
                                                          ExpConcat = Expa RExpConcat
               {2}
                                                          Expa = Expm RExpa
        RExpm = epsilon
                                                          Expm = ExpU RExpm
        RExpa = epsilon
                                                          ExpU = Fa
        RExpConcat = epsilon
                                                          Fa = 'id' RFa
        RExpRUm = epsilon
                                                              [019, 025] (0000,
                                                                                       id) {
        RExpRDois = epsilon
                                                                 i}
        RExpBoolAnd = epsilon
                                                          RFa = RArr
        RExp = epsilon
                                                          RArr = epsilon
           [017, 016] (0028,
                                       ))
                                                          RExpm = epsilon
                                                          RExpa = epsilon
               {)}
            [017, 018] (0003,
                                      do) {
                                                          RExpConcat = epsilon
                                                          RExpRUm = 'opr1' ExpConcat RExpRUm
               do}
                                                              [019, 027] (0037,
                                                                                      opr1)
         rep (i : int = h, i < n, i = i +
19
                                                                 {<}
   1) do
                                                          ExpConcat = Expa RExpConcat
                                                          Expa = Expm RExpa
        Instrucao = Comando RInstrucao
                                                          Expm = ExpU RExpm
        Comando = Rep Comando
                                                          ExpU = Fa
```

Fa = 'id' RFa

```
[019, 029] (0000,
                             id) {
                                                       [020, 011] (0036,
                                                                                   =)
       n}
                                                           {=}
RFa = RArr
                                                   Exp = ExpBoolAnd RExp
RArr = epsilon
                                                   ExpBoolAnd = ExpRDois RExpBoolAnd
RExpm = epsilon
                                                   ExpRDois = ExpRUm RExpRDois
RExpa = epsilon
                                                   ExpRUm = ExpConcat RExpRUm
RExpConcat = epsilon
                                                   ExpConcat = Expa RExpConcat
RExpRUm = epsilon
                                                   Expa = Expm RExpa
RExpRDois = epsilon
                                                   Expm = ExpU RExpm
RExpBoolAnd = epsilon
                                                   ExpU = Fa
                                                   Fa = 'id' RFa
[020, 013] (0000,
RExp = epsilon
   [019, 030] (0021,
                               ,)
                                                                                 id) {
       {,}
                                                          valores}
Id = 'id' Arr
                                                   RFa = RArr
   [019, 032] (0000,
                                                   RArr = '[' LRArr ']'
                              id) {
                                                      [020, 020] (0029,
                                                                                   [)
       i}
Arr = epsilon
                                                          {[}
    [019, 034] (0036,
                                                   LRArr = Exp
                               =)
       {=}
                                                   Exp = ExpBoolAnd RExp
Exp = ExpBoolAnd RExp
                                                   ExpBoolAnd = ExpRDois RExpBoolAnd
ExpBoolAnd = ExpRDois RExpBoolAnd
                                                   ExpRDois = ExpRUm RExpRDois
ExpRDois = ExpRUm RExpRDois
                                                   ExpRUm = ExpConcat RExpRUm
ExpRUm = ExpConcat RExpRUm
                                                   ExpConcat = Expa RExpConcat
ExpConcat = Expa RExpConcat
                                                   Expa = Expm RExpa
Expa = Expm RExpa
                                                   Expm = ExpU RExpm
Expm = ExpU RExpm
                                                   ExpU = Fa
                                                   Fa = 'id' RFa
ExpU = Fa
Fa = 'id' RFa
                                                       [020, 021] (0000,
                                                                              id) {
    [019, 036] (0000,
                              id) {
                                                          i}
       i}
                                                   RFa = RArr
                                                   RArr = epsilon
RFa = RArr
RArr = epsilon
                                                   RExpm = epsilon
                                                   RExpa = epsilon
RExpm = epsilon
RExpa = 'opa' Expm RExpa
                                                   RExpConcat = epsilon
    [019, 038] (0034,
                                                   RExpRUm = epsilon
                              opa)
       {+}
                                                   RExpRDois = epsilon
                                                   RExpBoolAnd = epsilon
Expm = ExpU RExpm
ExpU = Fa
                                                   RExp = epsilon
Fa = Cte
                                                       [020, 022] (0030,
                                                                                   ])
Cte = 'cteI'
                                                           {]}
                                                   RExpm = epsilon
   [019, 040] (0017,
                            cteI)
       {1}
                                                   RExpa = epsilon
RExpm = epsilon
                                                   RExpConcat = epsilon
RExpa = epsilon
                                                   RExpRUm = epsilon
RExpConcat = epsilon
                                                   RExpRDois = epsilon
                                                   RExpBoolAnd = epsilon
RExpRUm = epsilon
RExpRDois = epsilon
                                                   RExp = epsilon
RExpBoolAnd = epsilon
                                                      [020, 023] (0023,
                                                                                   ;)
RExp = epsilon
                                                           {;}
    [019, 041] (0028,
                               ))
       {)}
                                           21
                                                       j = i;
    [019, 043] (0003,
                              do) {
       do}
                                                   Comando = 'id' RArr ComandoX
                                                      Comando
                                                       [021, 009] (0000,
                                                                                  id) {
    c = valores[i];
                                                          j}
                                                   RArr = epsilon
Instrucao = Comando RInstrucao
Comando = 'id' RArr ComandoX
                                                   ComandoX = '=' Exp ';'
   Comando
                                                       [021, 011] (0036,
                                                                                   =)
    [020, 009] (0000,
                             id) {
                                                          {=}
       c}
                                                   Exp = ExpBoolAnd RExp
RArr = epsilon
                                                   ExpBoolAnd = ExpRDois RExpBoolAnd
ComandoX = '=' Exp ';'
                                                   ExpRDois = ExpRUm RExpRDois
```

ExpRUm = ExpConcat RExpRUm	RExpBoolAnd = '&&' ExpRDois
ExpConcat = Expa RExpConcat	RExpBoolAnd
Expa = Expm RExpa	[023, 021] (0039, &&)
Expm = ExpU RExpm	{&&}
ExpU = Fa	ExpRDois = ExpRUm RExpRDois
Fa = 'id' RFa	ExpRUm = ExpConcat RExpRUm
[021, 013] (0000, id) {	ExpConcat = Expa RExpConcat
i}	Expa = Expm RExpa
RFa = RArr	Expm = ExpU RExpm
RArr = epsilon	ExpU = Fa
RExpm = epsilon	Fa = 'id' RFa
RExpa = epsilon	[023, 024] (0000, id) {
RExpConcat = epsilon	valores}
RExpRUm = epsilon	RFa = RArr
RExpRDois = epsilon	RArr = '[' LRArr ']'
RExpBoolAnd = epsilon	[023, 031] (0029, [)
RExp = epsilon	{[}
[021, 014] (0023, ;)	LRArr = Exp
{;}	Exp = ExpBoolAnd RExp
	ExpBoolAnd = ExpRDois RExpBoolAnd
23	ExpRDois = ExpRUm RExpRDois
> c) do	
> c) do	ExpRUm = ExpConcat RExpRUm
	ExpConcat = Expa RExpConcat
Comando = Until Comando	Expa = Expm RExpa
Until = 'until' '(' Exp ')' 'do'	Expm = ExpU RExpm
Instrucao 'end'	ExpU = Fa
[023, 009] (0006, until) {	Fa = 'id' RFa
until}	[023, 032] (0000, id) {
[023, 014] (0027, ()	j}
{(}	RFa = RArr
Exp = ExpBoolAnd RExp	RArr = epsilon
ExpBoolAnd = ExpRDois RExpBoolAnd	RExpm = epsilon
ExpRDois = ExpRUm RExpRDois	RExpa = 'opa' Expm RExpa
ExpRUm = ExpConcat RExpRUm	[023, 034] (0034, opa)
ExpConcat = Expa RExpConcat	{-}
Expa = Expm RExpa	Expm = ExpU RExpm
Expm = ExpU RExpm	ExpU = Fa
ExpU = Fa	Fa = 'id' RFa
-	
Fa = 'id' RFa	
[023, 015] (0000, id) {	h}
j}	RFa = RArr
RFa = RArr	RArr = epsilon
RArr = epsilon	RExpm = epsilon
RExpm = epsilon	RExpa = epsilon
RExpa = epsilon	RExpConcat = epsilon
RExpConcat = epsilon	RExpRUm = epsilon
RExpRUm = 'opr1' ExpConcat RExpRUm	RExpRDois = epsilon
[023, 016] (0037, opr1)	RExpBoolAnd = epsilon
{>=}	RExp = epsilon
ExpConcat = Expa RExpConcat	[023, 037] (0030, ])
Expa = Expm RExpa	{]}
Expm = ExpU RExpm	RExpm = epsilon
ExpU = Fa	RExpa = epsilon
Fa = 'id' RFa	RExpConcat = epsilon
[023, 019] (0000, id) {	RExpRUm = 'opr1' ExpConcat RExpRUm
h}	[023, 039] (0037, opr1)
RFa = RArr	{>}
RArr = epsilon	ExpConcat = Expa RExpConcat
RExpm = epsilon	Expa = Expm RExpa
RExpa = epsilon	Expm = ExpU RExpm
RExpConcat = epsilon	ExpU = Fa
RExpRUm = epsilon	Fa = 'id' RFa
RExpRDois = epsilon	

```
[023, 041] (0000,
                                     id) {
                                                               [024, 030] (0000,
                                                                                        id) {
                                                                   valores}
               c}
        RFa = RArr
                                                           RFa = RArr
        RArr = epsilon
                                                           RArr = '[' LRArr ']'
        RExpm = epsilon
                                                               [024, 037] (0029,
                                                                                           [)
        RExpa = epsilon
                                                                  {[}
        RExpConcat = epsilon
                                                           LRArr = Exp
        RExpRUm = epsilon
                                                           Exp = ExpBoolAnd RExp
        RExpRDois = epsilon
                                                           ExpBoolAnd = ExpRDois RExpBoolAnd
        RExpBoolAnd = epsilon
                                                           ExpRDois = ExpRUm RExpRDois
        RExp = epsilon
                                                           ExpRUm = ExpConcat RExpRUm
            [023, 042] (0028,
                                       ))
                                                           ExpConcat = Expa RExpConcat
                                                           Expa = Expm RExpa
               {)}
                                                           Expm = ExpU RExpm
            [023, 044] (0003,
                                      do) {
                                                           ExpU = Fa
               dol
                                                           Fa = 'id' RFa
24
                    valores[j] = valores[j
                                                               [024, 038] (0000,
                                                                                        id) {
   - h];
                                                                  j}
                                                           RFa = RArr
                                                           RArr = epsilon
        Instrucao = Comando RInstrucao
        Comando = 'id' RArr ComandoX
                                                           RExpm = epsilon
           Comando
                                                           RExpa = 'opa' Expm RExpa
            [024, 017] (0000,
                                      id) {
                                                              [024, 040] (0034,
                                                                                         opa)
               valores}
                                                                { -}
        RArr = '[' LRArr ']'
                                                           Expm = ExpU RExpm
            [024, 024] (0029,
                                        [)
                                                           ExpU = Fa
                                                           Fa = 'id' RFa
               {[}
                                                               [024, 042] (0000,
        LRArr = Exp
                                                                                  id) {
        Exp = ExpBoolAnd RExp
                                                                  h}
        ExpBoolAnd = ExpRDois RExpBoolAnd
                                                           RFa = RArr
        ExpRDois = ExpRUm RExpRDois
                                                           RArr = epsilon
        ExpRUm = ExpConcat RExpRUm
                                                           RExpm = epsilon
                                                           RExpa = epsilon
        ExpConcat = Expa RExpConcat
        Expa = Expm RExpa
                                                           RExpConcat = epsilon
        Expm = ExpU RExpm
                                                           RExpRUm = epsilon
        ExpU = Fa
                                                           RExpRDois = epsilon
        Fa = 'id' RFa
                                                           RExpBoolAnd = epsilon
                                   id) {
            [024, 025] (0000,
                                                           RExp = epsilon
                                                               [024, 043] (0030,
                                                                                           ])
               j}
        RFa = RArr
                                                                  {]}
                                                           RExpm = epsilon
        RArr = epsilon
        RExpm = epsilon
                                                           RExpa = epsilon
        RExpa = epsilon
                                                           RExpConcat = epsilon
        RExpConcat = epsilon
                                                           RExpRUm = epsilon
                                                           RExpRDois = epsilon
        RExpRUm = epsilon
        RExpRDois = epsilon
                                                           RExpBoolAnd = epsilon
        RExpBoolAnd = epsilon
                                                           RExp = epsilon
        RExp = epsilon
                                                               [024, 044] (0023,
                                                                                           ;)
            [024, 026] (0030,
                                       ])
                                                                   {;}
               {]}
        ComandoX = '=' Exp';'
[024, 028] (0036,
                                                   25
                                                                       j = j - h;
                                                           Comando = 'id' RArr ComandoX
               {=}
        Exp = ExpBoolAnd RExp
                                                              Comando
        ExpBoolAnd = ExpRDois RExpBoolAnd
                                                               [025, 017] (0000,
                                                                                          id) {
        ExpRDois = ExpRUm RExpRDois
                                                                  j}
                                                           RArr = epsilon
        ExpRUm = ExpConcat RExpRUm
                                                           ComandoX = '=' Exp ';'
        ExpConcat = Expa RExpConcat
        Expa = Expm RExpa
                                                               [025, 019] (0036,
                                                                                           =)
        Expm = ExpU RExpm
                                                                  {=}
        ExpU = Fa
                                                           Exp = ExpBoolAnd RExp
        Fa = 'id' RFa
                                                           ExpBoolAnd = ExpRDois RExpBoolAnd
```

ExpRDois = ExpRUm RExpRDois

```
ExpRUm = ExpConcat RExpRUm
                                                           RExpConcat = epsilon
        ExpConcat = Expa RExpConcat
                                                           RExpRUm = epsilon
        Expa = Expm RExpa
                                                           RExpRDois = epsilon
        Expm = ExpU RExpm
                                                           RExpBoolAnd = epsilon
        ExpU = Fa
                                                           RExp = epsilon
        Fa = 'id' RFa
                                                               [028, 018] (0030,
                                                                                           ])
           [025, 021] (0000,
                                     id) {
                                                                  {]}
               j}
                                                           ComandoX = '=' Exp';'
        RFa = RArr
                                                               [028, 020] (0036,
                                                                                           =)
        RArr = epsilon
                                                                   {=}
        RExpm = epsilon
                                                           Exp = ExpBoolAnd RExp
        RExpa = 'opa' Expm RExpa
                                                           ExpBoolAnd = ExpRDois RExpBoolAnd
           [025, 023] (0034,
                                                           ExpRDois = ExpRUm RExpRDois
                                      opa)
               {-}
                                                           ExpRUm = ExpConcat RExpRUm
        Expm = ExpU RExpm
                                                           ExpConcat = Expa RExpConcat
        ExpU = Fa
                                                           Expa = Expm RExpa
        Fa = 'id' RFa
                                                           Expm = ExpU RExpm
           [025, 025] (0000,
                                                           ExpU = Fa
                                      id) {
                                                           Fa = 'id' RFa
               h}
        RFa = RArr
                                                               [028, 022] (0000,
                                                                                        id) {
        RArr = epsilon
                                                                  c}
        RExpm = epsilon
                                                           RFa = RArr
        RExpa = epsilon
                                                           RArr = epsilon
        RExpConcat = epsilon
                                                           RExpm = epsilon
                                                           RExpa = epsilon
        RExpRUm = epsilon
        RExpRDois = epsilon
                                                           RExpConcat = epsilon
                                                           RExpRUm = epsilon
        RExpBoolAnd = epsilon
                                                           RExpRDois = epsilon
        RExp = epsilon
                                                           RExpBoolAnd = epsilon
            [025, 026] (0023,
                                       ;)
                                                           RExp = epsilon
               {;}
                                                               [028, 023] (0023,
                                                                                           ;)
26
            end
                                                   29
        Comando = epsilon
                                                             end
        RInstrucao = epsilon
            [026, 009] (0004,
                                      end) {
                                                           Comando = epsilon
                                                           RInstrucao = epsilon
               end}
                                                               [029, 007] (0004,
                                                                                       end) {
28
            valores[j] = c;
                                                                   end}
                                                             h = h / 2;
        Comando = 'id' RArr ComandoX
           Comando
                                                           Comando = 'id' RArr ComandoX
            [028, 009] (0000,
                                       id) {
               valores}
                                                              Comando
        RArr = '[' LRArr ']'
                                                               [031, 007] (0000,
                                                                                          id) {
            [028, 016] (0029,
                                        [)
                                                                  h}
                                                           RArr = epsilon
               {[}
        LRArr = Exp
                                                           ComandoX = '=' Exp';'
                                                               [031, 009] (0036,
        Exp = ExpBoolAnd RExp
                                                                                           =)
        ExpBoolAnd = ExpRDois RExpBoolAnd
                                                                  {=}
        ExpRDois = ExpRUm RExpRDois
                                                           Exp = ExpBoolAnd RExp
        ExpRUm = ExpConcat RExpRUm
                                                           ExpBoolAnd = ExpRDois RExpBoolAnd
        ExpConcat = Expa RExpConcat
                                                           ExpRDois = ExpRUm RExpRDois
        Expa = Expm RExpa
                                                           ExpRUm = ExpConcat RExpRUm
        Expm = ExpU RExpm
                                                           ExpConcat = Expa RExpConcat
        ExpU = Fa
                                                           Expa = Expm RExpa
        Fa = 'id' RFa
                                                           Expm = ExpU RExpm
            [028, 017] (0000,
                                                           ExpU = Fa
                                     id) {
                                                           Fa = 'id' RFa
               j}
        RFa = RArr
                                                               [031, 011] (0000,
                                                                                        id) {
        RArr = epsilon
                                                                  h}
        RExpm = epsilon
                                                           RFa = RArr
                                                           RArr = epsilon
        RExpa = epsilon
```

```
RExpm = 'opm' ExpU RExpm
                                                          Funcao = 'def' TipoFuncao 'id' '('
                                                              Parametro ')' 'do' Instrucao '
            [031, 013] (0035,
                                      opm)
               {/}
                                                              end'
        ExpU = Fa
                                                               [038, 003] (0002,
                                                                                        def) {
        Fa = Cte
                                                                  def }
        Cte = 'cteI'
                                                           TipoFuncao = Tipo FArr
           [031, 015] (0017,
                                                           Tipo = 'void'
                                    cteI)
               {2}
                                                              [038, 007] (0011,
                                                                                       void) {
        RExpm = epsilon
                                                                  void}
        RExpa = epsilon
                                                           FArr = epsilon
                                                               [038, 012] (0000,
        RExpConcat = epsilon
                                                                                         id) {
        RExpRUm = epsilon
                                                                  main}
        RExpRDois = epsilon
                                                               [038, 016] (0027,
                                                                                          ()
        RExpBoolAnd = epsilon
                                                                 {(}
                                                           Parametro = epsilon
        RExp = epsilon
                                                               [038, 017] (0028,
           [031, 016] (0023,
                                      ;)
                                                                                         ))
               {;}
                                                                  {)}
                                                               [038, 019] (0003,
                                                                                        do) {
33
                                                                  do}
        end
        Comando = epsilon
                                                  40
                                                           int[] valores;
        RInstrucao = epsilon
            [033, 005] (0004,
                                     end) {
                                                           Instrucao = Comando RInstrucao
               end}
                                                           Comando = TipoFixo 'id' ComandoA
                                                              ';' Comando
35
                                                           TipoFixo = Tipo FArr
        return valores;
                                                           Tipo = 'int'
        Comando = Return
                                                               [040, 005] (0012,
                                                                                       int) {
        Return = 'return' Exp ';'
                                                                  int}
                                                           FArr = '[' RFArr ']'
            [035, 005] (0009, return) {
                                                               [040, 008] (0029,
               return}
                                                                                           [)
        Exp = ExpBoolAnd RExp
                                                                  {[}
                                                           RFArr = epsilon
        ExpBoolAnd = ExpRDois RExpBoolAnd
        ExpRDois = ExpRUm RExpRDois
                                                               [040, 009] (0030,
                                                                                          ])
        ExpRUm = ExpConcat RExpRUm
                                                                  {]}
        ExpConcat = Expa RExpConcat
                                                               [040, 011] (0000,
                                                                                         id) {
        Expa = Expm RExpa
                                                                  valores}
        Expm = ExpU RExpm
                                                           ComandoA = epsilon
        ExpU = Fa
                                                               [040, 018] (0023,
                                                                                         ;)
        Fa = 'id' RFa
                                                                  {;}
           [035, 012] (0000,
                                      id) {
               valores}
                                                  42
                                                           until(read() != EOF) do
        RFa = RArr
        RArr = epsilon
                                                           Comando = Until Comando
                                                           Until = 'until' '(' Exp ')' 'do'
        RExpm = epsilon
        RExpa = epsilon
                                                              Instrucao 'end'
        RExpConcat = epsilon
                                                               [042, 005] (0006,
                                                                                     until) {
        RExpRUm = epsilon
                                                                  until}
        RExpRDois = epsilon
                                                               [042, 010] (0027,
                                                                                           ()
        RExpBoolAnd = epsilon
                                                                  {(}
        RExp = epsilon
                                                           Exp = ExpBoolAnd RExp
            [035, 019] (0023,
                                                           ExpBoolAnd = ExpRDois RExpBoolAnd
                                      ;)
               {;}
                                                           ExpRDois = ExpRUm RExpRDois
                                                           ExpRUm = ExpConcat RExpRUm
36
      end
                                                           ExpConcat = Expa RExpConcat
                                                           Expa = Expm RExpa
                                                          Expm = ExpU RExpm
        RInstrucao = epsilon
            [036, 003] (0004,
                                                          ExpU = Fa
                                     end) {
                                                          Fa = 'id' RFa
               end }
                                                               [042, 011] (0024,
                                                                                        id) {
38
      def void main() do
                                                                  read}
                                                          RFa = '(' ParFunc ')'
        DeclFuncoes = Funcao DeclFuncoes
```

	[042, 015] (0027,	()			ParFunc = epsilon		
	{(}	()			[043, 033] (0028,	))	
	ParFunc = epsilon				{)}		
		))			RExpm = epsilon		
	{)}				RExpa = epsilon		
	RExpm = epsilon				RExpConcat = epsilon		
	RExpa = epsilon				RExpRUm = epsilon		
	RExpConcat = epsilon				RExpRDois = epsilon		
	RExpRUm = epsilon				RExpBoolAnd = epsilon		
	RExpRDois = 'opr2' ExpRUm				RExp = epsilon		
	[042, 018] (0038, opi	(2)			[043, 034] (0023,	;)	
	{ ! = }				{;}		
	<pre>ExpRUm = ExpConcat RExpRUm</pre>						
	ExpConcat = Expa RExpConcat			44	end		
	Expa = Expm RExpa						
	Expm = ExpU RExpm				Comando = epsilon		
	ExpU = Fa				RInstrucao = epsilon		
	Fa = 'id' RFa				[044, 005] (0004, en	d)	{
	[042, 021] (0000,	id)	{		end}		
	EOF}						
	RFa = RArr			46	<pre>int[] arr = shellsort(valores);</pre>		
	RArr = epsilon						
	RExpm = epsilon				Comando = TipoFixo 'id' Comando.	A	
	RExpa = epsilon				';' Comando		
	RExpConcat = epsilon				TipoFixo = Tipo FArr		
	RExpRUm = epsilon				Tipo = 'int'		
	RExpBoolAnd = epsilon					t)	{
	RExp = epsilon				int}		
		))			FArr = '[' RFArr ']'		
	{)}					[]	
	[042, 026] (0003,	(of	{		{[}		
	do}	,	•		RFArr = epsilon		
	,				<del>-</del>	])	
43	<pre>valores[] = lastValueRead();</pre>				{]}		
						d)	{
	Instrucao = Comando RInstrucao				arr}	/	•
	Comando = 'id' RArr ComandoX				ComandoA = '=' Exp		
	Comando					=)	
		id)	{		{=}	,	
	valores}				Exp = ExpBoolAnd RExp		
	RArr = '[' LRArr ']'				ExpBoolAnd = ExpRDois RExpBoolA:	n d	
	[043, 014] (0029,	[)			ExpRDois = ExpRUm RExpRDois	ı. u	
	{[}	L			ExpRUm = ExpConcat RExpRUm		
	LRArr = epsilon				ExpConcat = Expa RExpConcat		
	[043, 015] (0030,	])			Expa = Expm RExpa		
	{]}	١,			Expm = ExpU RExpm		
	ComandoX = '=' Exp';'				ExpU = Fa		
	[043, 017] (0036,	=)			Fa = 'id' RFa		
	[045, 017] (0056, {=}	-)				d)	1
	Exp = ExpBoolAnd RExp				shellsort}	u)	ι
	ExpBoolAnd = ExpRDois RExpBoolA	ام ۱			RFa = '(' ParFunc ')'		
		ina				<b>(</b> )	
	ExpRDois = ExpRUm RExpRDois					()	
	ExpRUm = ExpConcat RExpRUm				{(}		
	ExpConcat = Expa RExpConcat				ParFunc = RParFunc		
	Expa = Expm RExpa				RParFunc = Exp TRParFunc		
	Expm = ExpU RExpm				Exp = ExpBoolAnd RExp	1	
	ExpU = Fa				ExpBoolAnd = ExpRDois RExpBoolA	пa	
	Fa = 'id' RFa		r		ExpRDois = ExpRUm RExpRDois		
	•	id)	1		ExpRUm = ExpConcat RExpRUm		
	lastValueRead}				ExpConcat = Expa RExpConcat		
	RFa = '(' ParFunc ')'	, .			Expa = Expm RExpa		
	[043, 032] (0027,	()			Expm = ExpU RExpm		
	{(}				ExpU = Fa		

```
Fa = 'id' RFa
                                                           RExpBoolAnd = epsilon
           [046, 027] (0000,
                                     id) {
                                                           RExp = epsilon
                                                               [048, 020] (0021,
               valores}
                                                                                            ,)
        RFa = RArr
                                                                   {,}
        RArr = epsilon
                                                           Exp = ExpBoolAnd RExp
                                                           ExpBoolAnd = ExpRDois RExpBoolAnd
        RExpm = epsilon
        RExpa = epsilon
                                                           ExpRDois = ExpRUm RExpRDois
        RExpConcat = epsilon
                                                           ExpRUm = ExpConcat RExpRUm
        RExpRUm = epsilon
                                                           ExpConcat = Expa RExpConcat
        RExpRDois = epsilon
                                                           Expa = Expm RExpa
        RExpBoolAnd = epsilon
                                                           Expm = ExpU RExpm
                                                           ExpU = Fa
        RExp = epsilon
                                                           Fa = 'id' RFa
        TRParFunc = epsilon
                                                               [048, 022] (0000,
           [046, 034] (0028,
                                       ))
                                                                                         id) {
               {)}
                                                                   i}
        RExpm = epsilon
                                                           RFa = RArr
        RExpa = epsilon
                                                           RArr = epsilon
        RExpConcat = epsilon
                                                           RExpm = epsilon
                                                           RExpa = epsilon
        RExpRUm = epsilon
        RExpRDois = epsilon
                                                           RExpConcat = epsilon
        RExpBoolAnd = epsilon
                                                           RExpRUm = 'opr1' ExpConcat RExpRUm
        RExp = epsilon
                                                                [048, 024] (0037,
            [046, 035] (0023,
                                        ;)
                                                                   {<}
                {;}
                                                           ExpConcat = Expa RExpConcat
                                                           Expa = Expm RExpa
        rep(i : int = 0, i < len(valores),</pre>
48
                                                           Expm = ExpU RExpm
   i = i + 1) do
                                                           ExpU = Fa
                                                           Fa = 'id' RFa
                                                               [048, 026] (0000,
        Comando = Rep Comando
                                                                                        id) {
        Rep = 'rep' '(' VarControl ',' Exp
                                                                   len}
            ',' Id '=' Exp ')' 'do'
                                                           RFa = '(' ParFunc ')'
[048, 029] (0027,
            Instrucao 'end'
                                                                                            ()
            [048, 005] (0007,
                                    rep) {
                                                           ParFunc = RParFunc
               rep}
                                                           RParFunc = Exp TRParFunc
            [048, 008] (0027,
                                        ()
                                                           Exp = ExpBoolAnd RExp
        VarControl = 'id' ':' Tipo '=' Exp
                                                           ExpBoolAnd = ExpRDois RExpBoolAnd
            [048, 009] (0000,
                                     id) {
                                                           ExpRDois = ExpRUm RExpRDois
                i}
                                                           ExpRUm = ExpConcat RExpRUm
            [048, 011] (0022,
                                                           ExpConcat = Expa RExpConcat
                                        :)
                                                           Expa = Expm RExpa
               {:}
        Tipo = 'int'
                                                           Expm = ExpU RExpm
            [048, 013] (0012,
                                     int) {
                                                           ExpU = Fa
               int}
                                                           Fa = 'id' RFa
            [048, 017] (0036,
                                                               [048, 030] (0000,
                                                                                         id) {
                                        =)
                {=}
                                                                   valores}
                                                           RFa = RArr
        Exp = ExpBoolAnd RExp
        ExpBoolAnd = ExpRDois RExpBoolAnd
                                                           RArr = epsilon
        ExpRDois = ExpRUm RExpRDois
                                                           RExpm = epsilon
                                                           RExpa = epsilon
        ExpRUm = ExpConcat RExpRUm
        ExpConcat = Expa RExpConcat
                                                           RExpConcat = epsilon
        Expa = Expm RExpa
                                                           RExpRUm = epsilon
        Expm = ExpU RExpm

ExpU = Fa
                                                           RExpRDois = epsilon
                                                           RExpBoolAnd = epsilon
        Fa = Cte
                                                           RExp = epsilon
        Cte = 'cteI'
                                                           TRParFunc = epsilon
                                                               [048, 037] (0028,
           [048, 019] (0017,
                                                                                           ))
                                     cteI)
               {0}
                                                                  {)}
        RExpm = epsilon
                                                           RExpm = epsilon
        RExpa = epsilon
                                                           RExpa = epsilon
        RExpConcat = epsilon
                                                           RExpConcat = epsilon
        RExpRUm = epsilon
                                                           RExpRUm = epsilon
        RExpRDois = epsilon
                                                           RExpRDois = epsilon
```

RExpBoolAnd = epsilon			Expa = Expm RExpa	
RExp = epsilon			Expm = ExpU RExpm	
[048, 038] (0021,	,)		ExpU = Fa	
{,}			Fa = 'id' RFa	
Id = 'id' Arr			[049, 013] (0000,	id) -
[048, 040] (0000,	id) {		arr}	
i}			RFa = RArr	
Arr = epsilon			RArr = '[' LRArr ']'	
[048, 042] (0036,	=)		[049, 016] (0029,	[)
{=}			{[}	
Exp = ExpBoolAnd RExp			LRArr = Exp	
ExpBoolAnd = ExpRDois RExp	BoolAnd		Exp = ExpBoolAnd RExp	
ExpRDois = ExpRUm RExpRDoi			ExpBoolAnd = ExpRDois REx	pBoolAnd
ExpRUm = ExpConcat RExpRUm			ExpRDois = ExpRUm RExpRDo	_
ExpConcat = Expa RExpConca			ExpRUm = ExpConcat RExpRU	
Expa = Expm RExpa			ExpConcat = Expa RExpConc	
Expm = ExpU RExpm			Expa = Expm RExpa	
ExpU = Fa			Expm = ExpU RExpm	
Fa = 'id' RFa			ExpU = Fa	
[048, 044] (0000,	id) {		Fa = 'id' RFa	
i}			[049, 017] (0000,	id) ·
RFa = RArr			i}	
RArr = epsilon			RFa = RArr	
RExpm = epsilon			RArr = epsilon	
RExpa = 'opa' Expm RExpa			RExpm = epsilon	
[048, 046] (0034,	opa)		RExpa = epsilon	
{+}	-1/		RExpConcat = epsilon	
Expm = ExpU RExpm			RExpRUm = epsilon	
ExpU = Fa			RExpRDois = epsilon	
Fa = Cte			RExpBoolAnd = epsilon	
Cte = 'cteI'			RExp = epsilon	
[048, 048] (0017,	ctel)		[049, 018] (0030,	])
{1}	CUEI		{]}	17
RExpm = epsilon			RExpm = epsilon	
RExpa = epsilon			RExpa = epsilon	
RExpConcat = epsilon			RExpConcat = epsilon	
RExpRUm = epsilon			RExpRUm = epsilon	
RExpRDois = epsilon			RExpRDois = epsilon	
RExpBoolAnd = epsilon			RExpBoolAnd = epsilon	
			RExp = epsilon	
RExp = epsilon [048, 049] (0028,	))			
{)}	, ,		TRParFunc = epsilon [049, 019] (0028,	))
	ع ا ا			))
[048, 051] (0003,	do) {		{)}	
do}			[049, 020] (0023,	;)
			{;}	
<pre>print(arr[i]);</pre>		F.0		
		50	<pre>print("\n");</pre>	
Instrucao = Comando RInstr				
Comando = 'id' RArr Comand	οХ		Comando = 'id' RArr Coman	Xob
Comando			Comando	
[049, 007] (0025,	id) {		[050, 007] (0025,	id) -
<pre>print}</pre>			<pre>print}</pre>	
RArr = epsilon			RArr = epsilon	
<pre>ComandoX = '(' ParFunc ')'</pre>	<b>'</b> ; '		<pre>ComandoX = '(' ParFunc')</pre>	, ,;,
[049, 012] (0027,	()		[050, 012] (0027,	()
{(}			{(}	
ParFunc = RParFunc			ParFunc = RParFunc	
RParFunc = Exp TRParFunc			RParFunc = Exp TRParFunc	
Exp = ExpBoolAnd RExp			Exp = ExpBoolAnd RExp	
ExpBoolAnd = ExpRDois RExp	BoolAnd		ExpBoolAnd = ExpRDois REx	pBoolAnd
ExpRDois = ExpRUm RExpRDoi			ExpRDois = ExpRUm RExpRDo	_
ExpRUm = ExpConcat RExpRUm			ExpRUm = ExpConcat RExpRU	
ExpConcat = Expa RExpConca			ExpConcat = Expa RExpConc	

```
Expa = Expm RExpa
Expm = ExpU RExpm
                                                           Comando = epsilon
ExpU = Fa
                                                           RInstrucao = epsilon
Fa = Cte
                                                               [051, 005] (0004,
                                                                                             end) {
Cte = 'cteStr'
                                                                   end}
    [050, 013] (0019,
                              cteStr)
        \{\n}
                                                 53
                                                        end
RExpm = epsilon
RExpa = epsilon
                                                           Comando = epsilon
                                                           RInstrucao = epsilon
RExpConcat = epsilon
RExpRUm = epsilon
RExpRDois = epsilon
RExpBoolAnd = epsilon
                                                                [053, 003] (0004,
                                                                                            end) {
                                                                    end}
RExp = epsilon
                                                 55 endmod
TRParFunc = epsilon [050, 017] (0028,
                                                           DeclFuncoes = epsilon
  [055, 001] (0005,
                                     ))
        {)}
                                                                                          endmod) {
     [050, 018] (0023,
                                     ;)
                                                                    endmod}
         {;}
                                               Aceito!
                                               End:
end
```