```
        *       DEFINE( ≠NEXTITEM(NULL)≠,≠NEXTITEM≠)
                DEFINE( ≠NEWSTATE(OLD,TKN)≠,≠NEWSTATE≠)
        *       DEFINE( ≠NEXTTKN(NULL)≠,≠NEXTTKN≠)
        *       DEFINE( ≠NEXTFNC(NULL)≠,≠NEXTFNC≠)
        *       DEFINE( ≠EXTEND(KEY)≠,≠EXTEND≠)
                DEFINE( ≠EXTENDSS(NULL)FLAG,STATE,TKN,NSTATE,XSTATE≠ )
                DEFINE( ≠ASSEM(LOC,OP,RAND,COMM),A,B≠ )
                DEFINE( ≠UNSAVE(NULL)≠ )
                DEFINE( ≠GET(NULL)≠ )
        *
        *       DEFINE( ≠DUMPFNC(NULL)≠,≠DUMPFNC≠)
        *       DEFINE( ≠DUMPTKN(NULL)≠,≠DUMPTKN≠)
        *       DEFINE( ≠DUMPSTATET(NULL)≠,≠DUMPSTATET≠)
        *       DEFINE( ≠DUMPRULES(NULL)≠,≠DUMPRULES≠)
        *       DEFINE( ≠DUMPNONTERM(NULL)≠,≠DMPNONTERM≠)
                DEFINE( ≠PRINTSS(SS,FLAG)NM,S≠ )
                DEFINE( ≠PRINTRULES(NULL)LINE≠ )
                DEFINE( ≠PRINTS(S)NM,LN,T≠ )
                DEFINE( ≠PRINTTKNS(NULL)≠ )
        *
        *
0               DATA( ≠LISTEL(VALUE,NEXT)≠ )
        *
1               F1 = ≠F1,≠
2               F2 = ≠F2,≠
3               F3 = ≠F3,≠
4               T1 = ≠T1,≠
5               T2 = ≠T2,≠
6               T3 = ≠T3,≠
7               T4 = ≠T4,≠
8               T5 = ≠T5,≠
9               T6 = ≠T6,≠
10              T7 = ≠T7,≠
11              T8 = ≠T8,≠
12              T9 = ≠T9,≠
13              T10 = ≠T10,≠
14              T11 = ≠T11,≠
15              S2 = ≠S2,≠
16              S4 = ≠S4,≠
17              S5 = ≠S5,≠
18              R4 = ≠R4,≠
19              R5 = ≠R5,≠
20              R6 = ≠R6,≠
21              R7 = ≠R7,≠
22              R8 = ≠R9,≠
23              SS1 = ≠SS1,≠
24              SS2 = ≠SS2,≠
25              SS4 = ≠SS4,≠
26              SS5 = ≠SS5,≠
        *
27              DETACH( ≠INPUT≠ )
28              INPUT( ≠INPUT≠, ≠GRAMMAR≠, 80 )
29              OUTPUT( ≠ASSEMLINE≠, ≠SCMMOD≠, ≠≠ )
30              INPUT(≠UNSAVELINE≠, ≠SAVEF≠, 80 )
```

```
41                      REWIND( *SAVEF* )
                *
                *
42                      SYNERR = ****** SYNTAX ERROR* *
                *
                *
                *
43                      :(GO)
                *
44              NEXTTKN ITEM = NEXTITEM()
45                      IDENT( ITEM , *.* ) :S(FRETURN)
46                      NEXTTKN = $( T1 ITEM )
47                      IDENT( NEXTTKN , *# ) :F(RETURN)S(FAIL1)
                *
48              NEXTFNC ITEM = NEXTITEM()
49                      IDENT( ITEM , *.* ) :S(FRETURN)
50                      NEXTFNC = $( F1 ITEM )
51                      IDENT( NEXTFNC , *# ) :F(RETURN)S(FAIL2)
                *
52              NEXTITEM LINE SPAN(* *) BREAK(* *) . NEXTITEM = *# :S(RETURN)
53              NEXTITEM1 LINE = INPUT
54                      OUTPUT = LINE
55                      LINE POS(0) *#*     :F(NEXTITEM)S(NEXTITEM1)
                *
56              NEWSTATE NEWSTATE = $( S4 OLD *.* TKN )
57                      IDENT( $(S4 OLD *.* TKN) , *# ) :F(RETURN)
58                      STATECNT = STATECNT + 1
59                      $(S2 STATECNT ) = $( S2 OLD) *.* TKN
60                      $( S4 OLD *.* TKN ) = STATECNT
61                      NEWSTATE = STATECNT  :(RETURN)
                *
                *
                *
                *                   EXTEND A LIST BY TRANSITIVITY THROUGH NON TERMINALS
                *
62              EXTEND  TKN = TERMTKNCNT + 1
63              EXTEND1 GT( TKN, TKNCNT )    :S(EXTEND2)
64                      $(T6 TKN) = *X*
65                      TKN = TKN + 1        :(EXTEND1)
                *
66              EXTEND2 FLAG = *OFF*
67                      TKNHD = TERMTKNCNT + 1
                *
68              EXTEND3 GT(TKNHD, TKNCNT )   :S(EXTEND9)
69                      CHKTKN = $(T6 TKNHD )
70                      $(T6 TKNHD) = $(KEY TKNHD)
71                      TKN = $(KEY TKNHD)
                *
72              EXTEND4 IDENT( TKN, CHKTKN)  :S(EXTEND8)
73                      LF( TKN, TERMTKNCNT)          :S(EXTEND7)
74                      SUBTKN = $(KEY TKN)
                *
75              EXTEND5 IDENT(SUBTKN, *X* ) :S(EXTEND7)
76                      IDENT( $(KEY TKNHD *.* SUBTKN), *# )     :F(EXTEND6)
```

```
77                $(KEY TKNHD ≠.≠ SUBTKN) = $(KEY TKNHD)
78                $(KEY TKNHD) = SUBTKN
79                FLAG = ≠ON≠
                  *
80        EXTEND6 SUBTKN = $(KEY TKN ≠.≠ SUBTKN)          :(EXTEND5)
                  *
81        EXTEND7 TKN = $(KEY TKNHD ≠.≠ TKN )    :(EXTEND4)
                  *
82        EXTEND8 TKNHD = TKNHD + 1    :(EXTEND3)
                  *
83        EXTEND9 IDENT( FLAG, ≠OFF≠ )          :S(RETURN)F(EXTEND2)
                  *
                  *
                  *                   EXTEND STATE SET REPRESENTED IN ARRAY SET
                  *                   BY LAMBDA MOVES
                  *
                  *                   SETS SET AND SETX EMPTY AT END
                  *                        (ASSUMS SO AT START FOR SETX )
                  *
                  *                   RETURNS NUMBER OF EXTENDED SET
                  *
84        EXTENDSS  FLAG = ≠OFF≠
85                  STATE = 1
                  *
86        EXTENDSS1 GT( STATE, STATECNT) :S(EXTENDSS5)
87                  IDENT( SET[STATE], ≠≠ )   :S(EXTENDSS4)
88                  IDENT( SETX[STATE], ≠1≠ ) :S(EXTENDSS4)
89                  SETX[STATE] = 1
90                  TKN = TERMTKNCNT + 1
                  *
91        EXTENDSS2 GT(TKN, TKNCNT)       :S(EXTENDSS4)
92                  NSTATE = $(S4 STATE ≠.≠ TKN )
93                  IDENT( NSTATE, ≠≠ )     :S(EXTENDSS3)
94                  XSTATE = $(S4 0 ≠.≠ TKN )
95                  IDENT( SET[XSTATE], 1 )  :S(EXTENDSS3)
96                  SET[XSTATE] = 1
97                  FLAG = ≠ON≠
                  *
98        EXTENDSS3 TKN = TKN + 1         :(EXTENDSS2)
                  *
99        EXTENDSS4 STATE = STATE + 1    :(EXTENDSS1)
                  *
100       EXTENDSS5 IDENT( FLAG, ≠OFF≠ )          :F(EXTENDS5)
101                 STATE = 1
102                 SSTATE = ≠≠
                  *
103       EXTENDSS6 GT( STATE , STATECNT)  :S(EXTENDSS8)
104                 IDENT( SET[STATE], ≠≠ )        :S(EXTENDSS7)
105                 SSTATE = SSTATE ≠,≠ STATE
                  *
106       EXTENDSS7 SET[STATE] = ≠≠
107                 SETX[STATE] = ≠≠
108                 STATE = STATE + 1   :(EXTENDSS6)
                  *
```

```
109         EXTENDSSR EXTENDSS = $( SS1  STATE)
110                   IDENT( EXTENDSS, ## )    :F(RETURN)
111                   SSCNT = SSCNT + 1
112                   EXTENDSS = SSCNT
113                   $( SS1 STATE) = SSCNT
114                   $(SS2 SSCNT ) = STATE     :(RETURN)
                *
                *
                *                    OUTPUT AN ASSEMBLY LINE
                *
                *
115         ASSEM     A = LOC #                 #
116                   A TAB(10) . B  = ##
117                   A = R OP #                  #
118                   A  TAB(20) . B  = ##
119                   IDENT( COMM, ## )    :S(ASSEM1)
120                   A = B RAND #                                  #
121                   A  TAB(50) . B  = ##
122                   ASSEMLINE = B COMM   :(RETURN)
                *
123         ASSEM1    ASSEMLINE = B RAND   :(RETURN)
                *
                *
                *                    READ IN INFO FROM PASS 1
                *
124         UNSAVE    TERMTRNCNT = GET()
125                   TKNCNT = GET()
126                   RULECNT = GET()
127                   FNCCNT = GET()
128                   STATECNT = GET()
129         UNSAVE1   ITEM1 = GET()
130                   IDENT( ITEM1 ,#,,,,,,# )  :S(UNSAVE2)
131                   $(ITEM1) = GET()    :(UNSAVE1)
132         UNSAVE2   OUT( #ITEMS IN = # ITEMCNT )   :(RETURN)
                *
                *
133         GET       GET = TRIM( UNSAVELINE )
134                   ITEMCNT = ITEMCNT + 1          :(RETURN)
                *
                *
                *
                *
                *
                *
                *
                *
                *
                *
                *
                *
                *
                *                    PRINT OUT A STATE SET
                *
                *
135         PRINTSS   NM = $(SS2 SS) #,#
136         PRINTSS1  NM #,# BREAK(#,#) . S = ##   :F(RETURN)
```

```
137                 PRINTS(S)
138                 IDENT( FLAG, ## )     :S(PRINTSS1)
139                 OUT(OUTPUT )          :(PRINTSS1)
                *
                *
                *
                *                   PRINT OUT A STATE
                *
140         PRINTS  NM = $(S2 S) #.#
141                 LN = #                #
142         PRINTS1 NM #.# BREAK(#.#) . T  = ##    :F(PRINTS2)
143                 LN = LN # # $(T2 T)    :(PRINTS1)
144         PRINTS2 OUTPUT = LN   :(RETURN)
                *
                *
                *                   PRINT OUT RULE TABLE
                *
145         PRINTRULES              OUTPUT = #
146                 OUTPUT = ##
147                 OUTPUT = #        RULE INFO#
148                 OUTPUT = ##
149                 I = 1
150         PRINTRULES1   GT(I, RULECNT)    :S(RETURN)
151                 LINE = #  # I #     # $(T2 $(R4 I)) #    # $(F2 $(R5 I))
152                 LINE = LINE #     # $(R6 I)
153                 OUTPUT = LINE
154                 I = I + 1     :(PRINTRULES1)
                *
                *
                *
                *                   PRINT OUT TOKEN INFO
                *                   AND PLACE IN ASSEMBLY ALSO
                *
155         PRINTTKNS OUTPUT = ##
156                 OUTPUT = ##
157                 OUTPUT = #     TOKEN DEFINITIONS#
158                 OUTPUT = ##
159                 I = 1
160         PRINTTKNS1  GT( I, TKNCNT )    :S(RETURN)
161                 TEMP = $(T2 I) #                    #
162                 TEMP  TAB(24) . TEMP1  = ##
163                 OUTPUT = TEMP1  $(T3 I)
164                 ASSEM( #XXX#, #SET#, $(T3 I), $(T2 I) )
165                 I = I + 1     :(PRINTTKNS1)
                *
                *
                *
                *
166         GO      UNSAVE()
                *
                *
                *
167                 MVESS = ARRAY( #1:# TKNCNT )
168                 MVER  = ARRAY( #1:# TKNCNT )
```

```
169                 SET   = ARRAY( *1:* STATECNT )
170                 SETX  = ARRAY( *1:* STATECNT )
171                 SETZ  = ARRAY( *1:* STATECNT )
            *
            *
            *
            *                   GENERATE STATE SETS AND TRANSITION TABLE
            *
            *
            *
172         SS0     TOKEN = GET()
173                 IDENT( TOKEN, *....* )          :S(SS0A)
174                 EXTNAME = GET()
175                 IDENT( EXTNAME, *....* )        :S(FAILB)
176                 STATE = NEWSTATE( 0, TOKEN)
177                 SET[STATE] = 1
178                 SS = EXTENDSS()
179                 ASSEM( **, *ENTRY*, EXTNAME )
180                 $(SS5 SS) = EXTNAME             :(SS0)
            *
181         SS0A    OUTPUT = **
            *
182                 OUTPUT = **
183                 OUTPUT = **
184                 OUTPUT = *      STATE SETS AND TRANSITION TABLE*
            *
185                 SS = 1
            *
186         SS1     GT( SS, SSCNT)          :S(SSEND)
187                 OUTPUT = **
188                 OUTPUT = **
189                 SSNAME = $(SS2 SS) *.*
190                 SCNT = 0
            *
191                 T = 1
192         SS1A    GT(T, TKNCNT)           :S(SS1B)
193                 MVESS[T] = **
194                 MVER[T] = **
195                 T = T + 1               :(SS1A)
196         SS1B    TYPESTKN = **
197                 SKEY = **
            *
198         SS2     SSNAME *.* BREAK(*.*) . S  =  *,*  :F(SS3)
199                 SCNT = SCNT + 1
200                 SETZ[SCNT] = S          :(SS2)
            *
201         SS3     TKN = 1
            *
202         SS4     GT( TKN, TKNCNT )       :S(SS9)
203                 I = 1
204                 TFLAG = *OFF*
            *
205         SS5     GT( I, SCNT)            :S(SS7)
206                 S = SETZ[I]
```

```
207                K = $(S4 S ≠,≠ TKN )
208                IDENT( K, ≠≠)          :S(S6)
209                SET[K] = 1
210                TFLAG = ≠ON≠
                *
211        SS6     I = I + 1 :(SS5)
                *
212        SS7     IDENT( TFLAG, ≠OFF≠ )          :S(SS8)
213                MVESS[TKN] = EXTENDS()
                *
214        SS8     TKN = TKN + 1          :(SS4)
                *
                *
215        SS9     I = 1
                *
216        SS10    GT(I, SCNT)            :S(S13)
217                 S= SETZ[I]
218                R = $(S= S)
219                IDENT( R, ≠≠)          :S(S12)
220                NTKN = $(R4 R)
221                T = $(T)() NTKN)
                *
222        SS11    IDENT( T, ≠X≠)         :S(S12)
223                IDENT( MVESS[T], ≠≠ )          :S(SS11A)
                *
224                ASSEM( ≠≠, ≠ERR≠, ≠≠, ≠RULE AND STATE MOVE CONFLICT≠ )
225                OUTPUT = ≠≠
226                OUTPUT = ≠≠
227                OUTPUT = ≠******** ERROR, RULE AND STATE MOVE CONFLICT≠
228                OUT( OUTPUT )
229                OUTPUT = ≠STATE SET = ≠ $S ≠     ≠ $(SS2 S)
230                OUT( OUTPUT )
231                PRINTS(SS, ≠YES≠ )
232                OUTPUT = ≠TOKEN = ≠ $(T2 T)
233                OUT( OUTPUT )
234                OUTPUT = ≠≠
235                OUTPUT = ≠≠
                *
236        SS11A   IDENT( MVER[T], ≠≠) :S(SS11B)
237                ASSEM( ≠≠, ≠ERR≠, ≠≠, ≠2 RULES WITH SAME TOKEN≠ )
238                OUTPUT = ≠≠
239                OUTPUT = ≠≠
240                OUTPUT = ≠******** ERROR, 2 RULES WITH SAME TOKEN≠
241                OUT( OUTPUT )
242                OUTPUT = ≠STATE SET = ≠ $S ≠     ≠ $(SS2 S)
243                OUT( OUTPUT )
244                PRINTS( SS, ≠YES≠ )
245                OUTPUT = ≠TOKEN = ≠ $(T2 T)
246                OUT( OUTPUT )
247                OUTPUT = ≠≠
248                OUTPUT = ≠≠
                *
249        SS11B   MVER[T] = R
250                T = $(T)() NTKN ≠,≠ T )        :(SS11)
```

```
                *
                *
251             SS12    I = I + 1 :(SS10)
                *
252             SS13    T = 1
253             SS13A   GT( T, TERMTKNCNT)  :S(SS14)
254                     IDENT( MVESS[T], ##)        :F(SS13B)
255                     IDENT( MVER[T], ##) :S(SS13D)
256             SS13B   IDENT( TYPESTKN, ##)        :S(SS13C)
257                     IDENT( TYPESTKN, $(T4 T)) :S(SS13D)
258                     ASSEM( ##, #ERR#, ##, #TOKEN TYPES CONFLICT# )
259                     OUTPUT = ##
260                     OUTPUT = ##
261                     OUTPUT = ######## ERROR, TOKEN TYPES CONFLICT#
262                     OUT( OUTPUT )
263                     OUTPUT = #STATE SET = # SS #    # $(SS2 SS)
264                     OUT( OUTPUT )
265                     PRINTSS( SS, #YES# )
266                     OUTPUT = #KINDS OF GETTKN ARE # TYPESTKN # AND # $(T4 T)
267                     OUT( OUTPUT )
268                     OUTPUT = ##
269                     OUTPUT = ##
270                             :(SS13C1)
271             SS13C   TYPESTKN = $(T4 T)
                *
272             SS13C1  IDENT( SKEY, ## )   :S(SS13C2)
273                     IDENT( SKEY, $(T11 T) )         :S(SS13D)
274                     ASSEM( ##, #ERR#, ##, #TOKEN SUB KEYS CONFLICT# )
275                     OUTPUT = ##
276                     OUTPUT = ##
277                     OUTPUT = ###### ERROR, TOKEN SUBKEYS CONFLICT#
278                     OUT( OUTPUT )
279                     OUTPUT = #STATE SET = # SS #    # $(SS2 SS)
280                     OUT(OUTPUT)
281                     PRINTSS( SS, #YES# )
282                     OUTPUT = #KIND OF GETTKN IS # TYPESTKN
283                     OUT(OUTPUT)
284                     OUTPUT = #SUBKEY TYPES ARE # SKEY # AND # $(T11 T)
285                     OUT(OUTPUT)
286                     OUT(OUTPUT)
287                     OUTPUT = ##
288                     OUTPUT = ##
289                             :(SS13D)
                *
290             SS13C2  SKEY = $(T11 T)
291             SS13D   T = T + 1 :(SS13A)
                *
292             SS14    OUTLINE = SS #  # $(SS2 SS) #    # TYPESTKN #  # $(SS5 SS)
293                     OUTPUT = OUTLINE  #  # SKEY
294                     OUTPUT = ##
295                     PRINTSS(SS)
296                     ASSEM( #S,# SS, #VFD#, #30/# SKEY #,30/# TYPESTKN )
297                     OUTPUT = ##
298                      T = 1
```

```
299       SS14A     GT( T, TKNCNT )          :S(SS15)
300                 IDENT( MVESS[T], ##)  :S(SS14B)
301                 OUTPUT = #              STATE  # $(T2 T) #  # MVESS[T]
302                 RAND = #12/0,18/S.# MVESS[T] #,18/# $(T3 T) #,12/0#
303                 ASSEM( #,# , #VFD#, RAND )      :(SS14C)
304       SS14B     IDENT( MVER[T], ##)   :S(SS14C)
305                 OUTPUT = #              RULE   # $(T2 T) #  # MVER[T]
306                 RAND = #12/0,18/R.# MVER[T] #,18/# $(T3 T) #,12/1#
307                 ASSEM( #,# , #VFD#, RAND )
          *
308       SS14C     T = T + 1    :(SS14A)
          *
309       SS15      OUTPUT = #              FAIL#
310                 ASSEM( ##, #DATA#, 5 )
311                 IDENT( $(SS5 SS), ## )      :S(SS15B)
312                 ASSEM( $(SS5 SS), #EQU#, #S.# S$ )
          *
313       SS15B     SS = SS + 1              :(SS1)
          *
314       SSEND
          *
315                 ASSEM( #FL#, #BSS#, #0# )
316                 ASSEM( ## , #END # , ## )
          *
317                 :(END)
          *
          *
318       FAIL1     OUT( $YNERR  ITEM # NOT A TOKEN# ) :(FAIL)
319       FAIL2     OUT( $YNERR  ITEM # NOT A FUNCTION# )  :(FAIL)
320       FAIL4     OUT( $YNERR  # PERIOD EXPECTED# ) :(FAIL)
321       FAIL5     OUT( $YNERR  ## #I# EXPECTED ## )  :(FAIL)
322       FAIL6     OUT( $YNERR  ## #)# EXPECTED ## )  :(FAIL)
323       FAIL7     OUT( $YNERR  # PERIOD NOT EXPECTED# ) :(FAIL)
324       FAIL8     OUTPUT = #.... NOT EXPECTED ON $AVEF#    :(FAIL)
          *
325       FAIL      OUTPUT = SYNERR
326                 ASSEM( ##, #ERR#, ##, SYNERR )
          *
          *
          *
          *
          *
327       END

SUCCESSFUL COMPILATION


      STATE SETS AND TRANSITION TABLE


1   .1.5.7.18.27.29.35.39.50.52   GETTKN  S.PARAM  0

          PARAM
          EXP
          LOC.EXP
          LOC.TERM
          LOC.PRIM
```

```
            INTEGER.EXP
            INTEGER.TERM
            INTEGER.PRIM

            STATE  NMBSGN  2
            STATE  $  3
            STATE  (  4
            STATE  +  5
            STATE  -  6
            STATE  ↑  7
            STATE  INTEGER  8
            STATE  IDENTIFIER  9
            STATE  WORD.EXP  10
            STATE  WORD.PART  11
            STATE  INTEGER.EXP  12
            STATE  INTEGER.TERM  13
            STATE  INTEGER.PRIM  14
            STATE  LOC.EXP  15
            STATE  LOC.TERM  16
            STATE  LOC.PRIM  17
            STATE  EXP  18
            FAIL

2   .12.39.50.52   GETTKN     0

            LOC.EXP NMBSGN
            INTEGER.EXP
            INTEGER.TERM
            INTEGER.PRIM

            STATE  (  4
            STATE  +  5
            STATE  -  6
            STATE  ↑  7
            STATE  INTEGER  8
            STATE  IDENTIFIER  9
            STATE  INTEGER.EXP  19
            STATE  INTEGER.TERM  13
            STATE  INTEGER.PRIM  14
            FAIL

3   .14    GETKYWD     KEYWRL

            LOC.EXP $

            STATE  REG  20
            FAIL

4   .7.18.27.29.35.39.50.52.55    GETTKN     0

            LOC.EXP
            LOC.TERM
            LOC.PRIM
            WORD.EXP
            WORD.PART
            INTEGER.EXP
            INTEGER.TERM
            INTEGER.PRIM
            INTEGER.PRIM (

            STATE  NMBSGN  2
            STATE  $  3
```

```
            STATE   +   5
            STATE   -   6
            STATE   ↑   7
            STATE   INTEGER   8
            STATE   IDENTIFIER   9
            STATE   WORD.EXP   10
            STATE   WORD.PART   11
            STATE   INTEGER.EXP   12
            STATE   INTEGER.TERM   13
            STATE   INTEGER.PRIM   14
            STATE   LOC.EXP   21
            STATE   LOC.TERM   16
            STATE   LOC.PRIM   17
            FAIL


5   .46.50.52   GETTKN   0

            INTEGER.EXP =
            INTEGER.TERM
            INTEGER.PRIM

            STATE   (   4
            STATE   ↑   7
            STATE   INTEGER   8
            STATE   IDENTIFIER   9
            STATE   INTEGER.TERM   22
            STATE   INTEGER.PRIM   14
            FAIL


6   .48.50.52   GETTKN   0

            INTEGER.EXP =
            INTEGER.TERM
            INTEGER.PRIM

            STATE   (   4
            STATE   ↑   7
            STATE   INTEGER   8
            STATE   IDENTIFIER   9
            STATE   INTEGER.TERM   23
            STATE   INTEGER.PRIM   14
            FAIL


7   .58   GETTKN   0

            INTEGER.PRIM ↑

            STATE   IDENTIFIER   24
            FAIL


8   .54   GETTKN   0

            INTEGER.PRIM INTEGER

            RULE   BLANK   24
            RULE   NMBSGN   24
            RULE   )   24
            RULE   +   24
            RULE   *   24
            RULE   -   24
```

```
          RULE    v   24
          RULE    CR  24
          FAIL

9   .53   GETTKN    0

          INTEGER.PRIM IDENTIFIER

          RULE    BLANK  23
          RULE    NMBSGN  23
          RULE    )   23
          RULE    +   23
          RULE    *   23
          RULE    =   23
          RULE    :   23
          RULE    >   23
          RULE    v   23
          PULE    CR  23
          FAIL

10  .28.32   GETTKN    0

          LOC.PRIM WORD.EXP
          WORD.EXP WORD.EXP

          RULE    BLANK  12
          RULE    NMBSGN  12
          RULE    )   12
          STATE   *   25
          RULE    :   12
          RULE    >   12
          RULE    CR  12
          FAIL

11  .31   GETTKN    0

          WORD.EXP WORD.PART

          RULE    BLANK  14
          RULE    NMBSGN  14
          RULE    )   14
          RULE    *   14
          RULE    :   14
          RULE    >   14
          RULE    CR  14
          FAIL

12  .30.36.41   GETTKN    0

          WORD.EXP INTEGER.EXP
          WORD.PART INTEGER.EXP
          INTEGER.EXP INTEGER.EXP

          RULE    BLANK  13
          RULE    NMBSGN  13
          RULE    )   13
          STATE   +   26
          RULE    .   13
          STATE   -   27
          RULE    :   13
          RULE    >   13
```

```
            RULE    CR  13
            FAIL

13  .40     GETTKN      0

            INTEGER.EXP INTEGER.TERM

            RULE    BLANK   17
            RULE    NMBSGN  17
            RULE    )   17
            RULE    +   17
            RULE    *   17
            RULE    -   17
            RULE    /   17
            RULE    >   17
            RULE    <   17
            RULE    CR  17
            FAIL

14  .51     GETTKN      0

            INTEGER.TERM INTEGER.PRIM

            RULE    BLANK   22
            RULE    NMBSGN  22
            RULE    )   22
            RULE    +   22
            RULE    *   22
            RULE    -   22
            RULE    /   22
            RULE    >   22
            RULE    <   22
            RULE    CR  22
            FAIL

15  .6.9    GETTKN      0

            EXP LOC.EXP
            LOC.EXP LOC.EXP

            RULE    BLANK   3
            STATE   NMBSGN  20
            RULE    CR  3
            FAIL

16  .8.22   GETTKN      0

            LOC.EXP LOC.TERM
            LOC.TERM LOC.TERM

            RULE    BLANK   4
            RULE    NMBSGN  4
            RULE    )   4
            STATE   :   30
            RULE    CR  4
            FAIL

17  .19     GETTKN      0
```

```
        RULE    BLANK   8
        RULE    NMBSGN  8
        RULE    )   8
        RULE    :   8
        STATE   >   31
        RULE    CR  8
        FAIL

18  .2  GETTKN     0

        PARAM EXP

        STATE   BLANK   32
        STATE   CR  33
        FAIL

19  .13.41  GETTKN     0

        LOC.EXP NMBSGN INTEGER.EXP
        INTEGER.EXP INTEGER.EXP

        RULE    BLANK   4
        RULE    NMBSGN  6
        RULE    )   6
        STATE   +   26
        STATE   -   27
        RULE    CR  6
        FAIL

20  .15 GETTKN     0

        LOC.EXP $ REG

        STATE   NMBSGN  34
        FAIL

21  .9.56   GETTKN     0

        LOC.EXP LOC.EXP
        INTEGER.PRIM ( LOC.EXP

        STATE   NMBSGN  29
        STATE   )   35
        FAIL

22  .47 GETTKN     0

        INTEGER.EXP + INTEGER.TERM

        RULE    BLANK   20
        RULE    NMBSGN  20
        RULE    )   20
        RULE    +   20
        RULE    ,   20
        RULE    -   20
        RULE    :   20
        RULE    >   20
        RULE    V   20
        RULE    CR  20
        FAIL
```

```
23  .49   GETTKN    0

          INTEGER.EXP = INTEGER.TERM

          RULE   BLANK  21
          RULE   NMBSGN  21
          RULE   )  21
          RULE   +  21
          RULE   *  21
          RULE   =  21
          RULE   :  21
          RULE   >  21
          RULE   ^  21
          RULE   CR  21
          FAIL

24  .59   GETTKN    0

          INTEGER.PRIM = IDENTIFIER

          RULE   BLANK  26
          RULE   NMBSGN  26
          RULE   )  26
          RULE   +  26
          RULE   *  26
          RULE   =  26
          RULE   :  26
          RULE   >  26
          RULE   ^  26
          RULE   CR  26
          FAIL

25  .33.35.39.50.52   GETTKN    0

          WORD.EXP WORD.EXP ,
          WORD.PART
          INTEGER.EXP
          INTEGER.TERM
          INTEGER.PRIM

          STATE  (  4
          STATE  +  5
          STATE  -  6
          STATE  ^  7
          STATE  INTEGER  8
          STATE  IDENTIFIER  9
          STATE  WORD.PART  30
          STATE  INTEGER.EXP  37
          STATE  INTEGER.TERM  13
          STATE  INTEGER.PRIM  14
          FAIL

26  .42.50.52   GETTKN    0

          INTEGER.EXP INTEGER.EXP +
          INTEGER.TERM
          INTEGER.PRIM

          STATE  (  4
          STATE  ^  7
```

```
          STATE   IDENTIFIER  9
          STATE   INTEGER.TERM  38
          STATE   INTEGER.PRIM  14
          FAIL

27  .44.50.52   GETTKN     0

          INTEGER.EXP INTEGER.EXP =
          INTEGER.TERM
          INTEGER.PRIM

          STATE   (  4
          STATE   ↑  7
          STATE   INTEGER  8
          STATE   IDENTIFIER  9
          STATE   INTEGER.TERM  39
          STATE   INTEGER.PRIM  14
          FAIL

28  .37.39.50.52   GETTKN     0

          WORD.PART INTEGER.EXP ∨
          INTEGER.EXP
          INTEGER.TERM
          INTEGER.PRIM

          STATE   (  4
          STATE   +  5
          STATE   -  6
          STATE   ↑  7
          STATE   INTEGER  8
          STATE   IDENTIFIER  9
          STATE   INTEGER.EXP  40
          STATE   INTEGER.TERM  13
          STATE   INTEGER.PRIM  14
          FAIL

29  .10.39.50.52   GETTKN     0

          LOC.EXP LOC.EXP NMRSGN
          INTEGER.EXP
          INTEGER.TERM
          INTEGER.PRIM

          STATE   (  4
          STATE   +  5
          STATE   -  6
          STATE   ↑  7
          STATE   INTEGER  8
          STATE   IDENTIFIER  9
          STATE   INTEGER.EXP  41
          STATE   INTEGER.TERM  13
          STATE   INTEGER.PRIM  14
          FAIL

30  .23   GETTKN     0

          LOC.TERM LOC.TERM :

          STATE   IDENTIFIER  42
          FAIL
```

```
31  .20   GETTKN     A

          LOC.TERM LOC.PRIM >

          STATE  IDENTIFIER  43
          FAIL


32  .3

          PARAM EXP BLANK

          FAIL


33  .4

          PARAM EXP CR

          FAIL


34  .16.39.50.52   GETTKN     0

          LOC.EXP $ REG NMBSGN
          INTEGER.EXP
          INTEGER.TERM
          INTEGER.PRIM

          STATE  (  4
          STATE  +  5
          STATE  -  6
          STATE  +  7
          STATE  INTEGER  8
          STATE  IDENTIFIER  9
          STATE  INTEGER.EXP  44
          STATE  INTEGER.TERM  13
          STATE  INTEGER.PRIM  14
          FAIL


35  .57   GETTKN     0

          INTEGER.PRIM ( LOC.EXP )

          RULE   BLANK  25
          RULE   NMBSGN  25
          RULE   )  25
          RULE   +  25
          RULE   ,  25
          RULE   -  25
          RULE   ;  25
          RULE   >  25
          RULE   V  25
          RULE   CR  25
          FAIL


36  .34   GETTKN     0

          WORD.EXP WORD.EXP . WORD.PART

          RULE   BLANK  15
          RULE   NMBSGN  15
          RULE   )  15
```

```
RULE    >   15
RULE    CR  15
FAIL
```

```
37  .36.41    GETTKN      0

          WORD.PART INTEGER.EXP
          INTEGER.EXP INTEGER.EXP

          STATE   +   26
          STATE   -   27
          STATE   V   28
          FAIL
```

```
38  .43     GETTKN      A

          INTEGER.EXP INTEGER.EXP + INTEGER.TERM

          RULE    BLANK   18
          RULE    NMBSGN  18
          RULE    )   18
          RULE    +   18
          RULE    *   18
          RULE    -   18
          RULE    :   18
          RULE    >   18
          RULE    V   18
          RULE    CR  18
          FAIL
```

```
39  .45     GETTKN      0

          INTEGER.EXP INTEGER.EXP - INTEGER.TERM

          RULE    BLANK   19
          RULE    NMBSGN  10
          RULE    )   19
          RULE    +   19
          RULE    *   19
          RULE    -   19
          RULE    :   19
          RULE    >   19
          RULE    V   19
          RULE    CR  19
          FAIL
```

```
40  .39.41    GETTKN      0

          WORD.PART INTEGER.EXP V INTEGER.EXP
          INTEGER.EXP INTEGER.EXP

          RULE    BLANK   16
          RULE    NMBSGN  16
          RULE    )   16
          STATE   +   26
          RULE    *   16
          STATE   -   27
          RULE    :   16
          RULE    >   16
          RULE    CR  16
```

```
41  .11.41   GETTKN     0

         LOC.EXP LOC.EXP NMBSGN INTEGER.EXP
         INTEGER.EXP INTEGER.EXP

         RULE   BLANK  E
         RULE   NMBSGN  5
         RULE   )  E
         STATE  +  26
         STATE  -  27
         RULE   CR  5
         FAIL


42  .24   GETTKN     0

         LOC.TERM LOC.TERM : IDENTIFIER

         RULE   BLANK  11
         RULE   NMBSGN  11
         RULE   )  11
         RULE   :  11
         STATE  :  45
         RULE   CR  11
         FAIL


43  .21   GETTKN     0

         LOC.TERM LOC.PRIM > IDENTIFIER

         RULE   BLANK  0
         RULE   NMBSGN  9
         RULE   )  0
         RULE   :  0
         RULE   CR  9
         FAIL


44  .17.41   GETTKN     0

         LOC.EXP $ REG NMBSGN INTEGER.EXP
         INTEGER.EXP INTEGER.EXP

         RULE   BLANK  7
         RULE   NMBSGN  7
         RULE   )  7
         STATE  +  26
         STATE  -  27
         RULE   CR  7
         FAIL


45  .25.27.29.35.39.50.55   GETTKN     0

         LOC.TERM LOC.TERM : IDENTIFIER ;
         LOC.PRIM
         WORD.EXP
         WORD.PART
         INTEGER.EXP
         INTEGER.TERM
         INTEGER.PRIM
```

```
          STATE   +  5
          STATE   -  6
          STATE   ↑  7
          STATE   INTEGER  8
          STATE   IDENTIFIER  9
          STATE   WORD.EXP  10
          STATE   WORD.PART  11
          STATE   INTEGER.EXP  12
          STATE   INTEGER.TERM  13
          STATE   INTEGER.PRIM  14
          STATE   LOC.PRIM  46
          FAIL

46  .25   GETTKN     8

          LOC.TERM LOC.TERM ; IDENTIFIER ; LOC.PRIM

          RULE    BLANK  10
          RULE    NMBSGN  10
          RULE    )  10
          RULE    ;  10
          RULE    CR  10
          FAIL
```