```
   let EndsExp[] = valof
   §E
       switchon ChType into
       §   case SECTKET:
        5      case REPEAT:
               case REPEATUNTIL:
               case REPEATWHILE:
               case DO:
               case OR:
       10      case IFSO:
               case IFNOT:
               case TO:
               case BY:
               case SEMICOLON:
       15      case INTO:
               case LET:
               case AND:
               case ENDPROG:
               case GLOBAL:
       20      case MANIFEST:
               case TABLE:
               case STATIC: resultis true
               default: resultis false
   $E
       25
   and Command[] be
   §C
       §R  switchon ChType into
           §S  case SECTBRA:
       30              Block[]
                       break
               case CASE:
                       ScanExpression[]
                       unless Ch = COLON do
       35              §   Error[113, HARD, Ch, Send, CCOLON]
                           loop
                       $
                       Send[CCOLON]
                       Read[]
       40              loop
               case DEFAULT:
                       Scan[]
                       unless Ch = COLON do
                       §   Error[113, HARD, Ch, Send, CCOLON]
       45                  loop
                       $
                       Send[CCOLON]
                       Read[]
                       loop
       50          case IF:
                   case UNLESS:
                   case WHILE:
                   case UNTIL:
                           ScanExpression[]
       55                  unless Ch = DO do
                           §   Error[130, HARD, Ch, DeleteCommand]
                               break
                           $
                           endcase
       60          case TEST:
                           ScanExpression[]
```

```
                    unless Ch = DO ∨ Ch = IFSO ∨ Ch = IFNOT do
                    §   Error[144, HARD, Ch, DeleteCommand]
                        break
                    §
                  § let A = TESTIFS - Ch
                    ScanCommand[]
                    unless Ch = A do
                    §   Error[145, HARD, Ch, DeleteCommand]
                        Send[A]
                        Send[SEMICOLON]
                        break
                    §
                    endcase
                  §
              case GOTO:
              case RESULTIS:
                    ScanExpression[]
                    unless EndsExp[] do
                    §   Error[102, HARD, Ch, DeleteCommand]
                        break
                    §
                    break
              case REPEAT:
              case REPEATUNTIL:
              case REPEATWHILE:
                    break
              case BREAK:
              case LOOP:
              case ENDCASE:
              case RETURN:
                    Scan[]
                    break
              case FOR:
                    NewLevel[LOCAL]
                  § let N = ScanName[]
                    if N = NOTNAME do
                    §   Error[201, HARD, Ch, DelComRestLevel]
                        break
                    §
                    AddDef[N, SIMPLE, VALDF]
                    unless Ch = EQ do
                    §   Error[122, HARD, Ch, DelComRestLevel]
                        break
                    §
                    Read[]
                    Expression[]
                    unless Ch = TO do
                    §   Error[132, HARD, Ch, DelComRestLevel]
                        break
                    §
                    ScanExpression[]
                    if Ch = BY do
                        ScanExpression[]
                    unless Ch = DO do
                    §   Error[130, HARD, Ch, DelComRestLevel]
                        break
                    §
                    ScanCommand[]
                    Send[ENDFOR]
                    RestoreLevel[]
                    break
                  §
              case NAME:
              case NUMBER:
              case STRING:
```

```
            case CHARACTER:
            case TRUE:
            case FALSE:
130         case RV:
            case RBRA:
            case VALOF:
                    §A  let n = 1
                        Expression[]
135                     if Ch = COLON endcase
                        unless Ch = ASS ∨ Ch = COMMA do
                        §   unless EndsExp[] do
                                Error[102, HARD, Ch, DeleteCommand]
                            break
140                     §
                        while Ch = COMMA do
                        §   ScanExpression[]
                            n := n+1
                        §
145                     unless Ch = ASS do
                        §   Error[112, HARD, Ch, DeleteCommand]
                            break
                        §
                        §   ScanExpression[]
150                         n := n-1
                        § repeatwhile Ch = COMMA
                        unless n = 0 do
                            Error[302, HARD, Ch, DeleteCommand]
                        unless EndsExp[] do
155                         Error[102, HARD, Ch, DeleteCommand]
                        break
                    §A
            case SWITCHON:
                    ScanExpression[]
160                 unless Ch = INTO do
                    §   Error[131, HARD, Ch, DeleteCommand]
                        break
                    §
                    Scan[]
165                 Block[]
                    Send[ENDSWITCH]
                    break
            case OR:
            case IFSO:
170         case IFNOT:
            case SEMICOLON:
            case SECTKET:
            case ENDPROG:
            case LET:
175         case AND:
            case GLOBAL:
            case MANIFEST:
            case TABLE:
            case STATIC:
180                 return
            default:
                    Error[103, HARD, Ch, DeleteCommand]
                    break
        §S
185     Scan[]
    §R repeat

    switchon ChType into
    §S
190     default:return
```

```
            case REPEAT:
                    Scan[]
                    endcase
195
            case REPEATUNTIL:
            case REPEATWHILE:
                    ScanExpression[]
                    unless EndsExp[] do
200                 §   Error[102, HARD, Ch, DeleteCommand]
                        endcase
                    §
                    endcase
        $S repeat
205 $C

    and Expression[] be
    §E
        §R
210         switchon ChType into
            §S
                case VALOF:
                        NewLevel[VALBODY]
                        ScanCommand[]
215                     RestoreLevel[]
                        Send[ENDVALOF]
                        loop
                case RBRA:
                        ScanExpression[]
220                     unless Ch = RKET do
                            Error[115, HARD, Ch, ExpressionError]
                        endcase
                case VECAP:
                        ScanExpression[]
225                     unless Ch = SKET do
                            Error[116, HARD, Ch, ExpressionError]
                        endcase
                case SBRA:
                        ScanExpression[] repeatwhile Ch = COMMA
230                     unless Ch = SKET do
                            Error[116, HARD, Ch, ExpressionError]
                        endcase
                case COND:
                        ScanExpression[]
235                     unless Ch = COMMA do
                            Error[114, HARD, Ch, ExpressionError]
                        endcase

                case NAME:
240                   § let F = NameBlock[Ch]
                        until F = NILNAME do
                        §   if NameVal[F] = UNUSED do
                                UpdateNameVal[F, UNDEFINED]
                            F := PreviousNameBlock[F]
245                     §
                        endcase
                      §

                case SECTBRA:
250             case IF:
                case UNLESS:
                case WHILE:
                case UNTIL:
                case TEST:
255             case FOR:
                case LOOP:
```

```
                case BREAK:
                case RETURN:
                case RESULTIS:
260             case SWITCHON:
                case CASE:
                case DEFAULT:
                case ENDCASE:
                case BE:
265             case GOTO:
                case IS:
                case VEC:
                        Error[101, HARD, Ch, ExpressionError]
                case SECTKET:
270             case REPEAT:
                case REPEATUNTIL:
                case REPEATWHILE:
                case DO:
                case OR:
275             case IFSO:
                case IFNOT:
                case TO:
                case BY:
                case SEMICOLON:
280             case INTO:
                case LET:
                case AND:
                case ENDPROG:
                case GLOBAL:
285             case MANIFEST:
                case TABLE:
                case STATIC:
                case RKET:
                case SKET:
290             case COLON:
                case COMMA:
                case ASS:
                        return
                case ERROR:
295                     ExpError := true
                        return
                default:
                        endcase              || all other symbols
            $S
300         if ExpError return
            Scan[]
        $R repeat
    $E

305 and DeleteCommand[] be
    §DC
        ExpError := false
        Send[ERROR]
        until IsEndofCom[] do Read[]
310 $DC

    and IsEndofCom[] = valof
    §IEC
        switchon ChType into
315     §S
            case SEMICOLON:
            case SECTKET:
            case LET:
            case AND:
320         case GLOBAL:
            case MANIFEST:
```

```
            case STATIC:
            case TABLE:
            case ENDPROG: resultis true
325         case SECTBRA: DeleteBlock[]
            default:      resultis false
        $S
    $IEC

330 and DeleteBlock[] be
    §D
        let SectKet = FormSecKet[Ch]
        Read[]
        switchon ChType into
335     §S
            case SECTBRA: DeleteBlock[]
                          Read[]
                          endcase
            case SECTKET: if Ch = SectKet return
340                       unless SectKet = NullSecKet[] do
                            Error[107, SOFT, SectKet, NullProgram]
                          BackUp[SectKet]
                          return
            case ENDPROG: Error[5, HARD, Ch, BackUp, SectKet]
345                       endcase
            default:      Read[]
        $S
        repeat
    $D
350
    and DelComRestLevel[] be
    §DCRL
        DeleteCommand[]
        RestoreLevel[]
355 $DCRL

    and ExpressionError[] be
    §E
        BackUp[ERROR]
360     ExpError := true
    $E

****
```