```
program ->
        function-block

function-block ->
        function function-block

function ->
        FUNCTION identifier SEMICOLON BEGIN_PARAMS declaration-block-
optional END_PARAMS BEGIN_LOCALS declaration-block-optional END_LOCALS
BEGIN_BODY statement-block-optional END_BODY

declaration-block-optional ->
        epsilon | declaration-block

declaration-block ->
        declaration SEMICOLON | declaration SEMICOLON declaration-
block

statement-block-optional ->
        epsilon | statement-block

statement-block ->
        statement SEMICOLON | statement SEMICOLON statement-block

declaration ->
        identifier-block COLON declaration-type

declaration-type ->
        INTEGER | ARRAY L_SQUARE_BRACKET NUMBER R_SQUARE_BRACKET OF
INTEGER

identifier-block ->
        identifier | identifier COMMA identifier-block

identifier ->
        IDENT

statement ->
        var ASSIGN expression
      | IF bool-expr THEN statement-block ENDIF
      | IF bool-expr THEN statement-block ELSE statement-block ENDIF
      | WHILE bool-expr BEGINLOOP statement-block ENDLOOP
      | DO BEGINLOOP statement-block ENDLOOP WHILE bool-expr
      | READ var-block
      | WRITE var-block
      | BREAK
      | RETURN expression

bool-expr ->
        relation-and-expr
      | relation-and-expr OR bool-expr

relation-and-expr ->
        relation-expr
      | relation-expr AND relation-and-expr

relation-expr ->
        relation-expr-body
      | NOT relation-expr-body

relation-expr-body ->
        expression comp expression
      | TRUE
      | FALSE
      | L_PAREN bool-expr R_PAREN

comp ->
        EQ
      | NEQ
      | LT
      | GT
      | LTE
      | GTE

expression ->
        multiplicative-expr
      | multiplicative-expr ADD expression
      | multiplicative-expr SUB expression

multiplicative-expr ->
        term
      | term MULT multiplicative-expr
      | term DIV multiplicative-expr
      | term MOD multiplicative-expr

term ->
        term-body
      | MINUS term-body
      | IDENT L_PAREN expression-block R_PAREN

term-body ->
        NUMBER
      | var
      | L_PAREN expression R_PAREN

expression-block ->
        expression
      | expression COMMA expression-block

var-block ->
        var
      | var COMMA var-block

var ->
        identifier
      | identifier L_SQUARE_BRACKET expression R_SQUARE_BRACKET
```