Parsed Rules

```
 (1) program → prolog FUNC func_list_pre MAIN '(' ')' '{' EOL body '}' EOL func_list EOF
 (2) prolog → PACKAGE MAIN EOL
 (3) func_list_pre → ε
 (4) func_list_pre → FUNC_ID '(' param_list ')' func_def_type EOL FUNC func_list_pre
 (5) func_list → ε
 (6) func_list → func_def func_list
 (7) func_def → FUNC FUNC_ID '(' param_list ')' func_def_type EOL
 (8) param_list → ε
 (9) param_list → VAR_ID type next_param
(10) next_param → ε
(11) next_param → ',' VAR_ID type next_param
(12) func_def_type → '{' EOL body '}'
(13) func_def_type → '(' func_def_ret
(14) func_def_ret → ')' '{' EOL body '}'
(15) func_def_ret → ret_list ')' '{' EOL body '}'
(16) ret_list → type next_ret
(17) next_ret → ε
(18) next_ret → ',' type next_ret
(19) body → ε
(20) body → command EOL body
(21) command → VAR_ID var_
(22) command → func_call
(23) command → if
(24) command → cycle
(25) command → return
(26) var_ → var_def
(27) var_ → var_move
(28) var_def → ':=' EXPR
(29) var_move → next_id '=' expr_list
(30) next_id → ε
(31) next_id → ',' VAR_ID next_id
(32) if → IF COND '{' EOL body '}' if_cont
(33) if_cont → ε
(34) if_cont → ELSE else_
(35) else_ → '{' EOL body '}'
(36) else_ → if
(37) cycle → FOR for_def ';' COND ';' for_move '{' EOL body '}'
(38) for_def → ε
(39) for_def → VAR_ID var_def
(40) for_move → ε
(41) for_move → VAR_ID var_move
(42) return → RETURN expr_list
(43) func_call → FUNC_ID '(' func_args ')'
(44) func_args → ε
(45) func_args → expr_list
(46) expr_list → EXPR next_expr
(47) expr_list → func_call
(48) next_expr → ε
(49) next_expr → ',' EXPR next_expr
(50) type → INT
(51) type → FLOAT64
(52) type → STRING
```