```
start : program
      ;

program : program unit
        | unit
        ;

unit : var_declaration
     | func_definition
     ;

func_definition : type_specifier ID LPAREN parameter_list RPAREN compound_statement
                | type_specifier ID LPAREN RPAREN compound_statement
                ;

parameter_list : parameter_list COMMA type_specifier ID
               | parameter_list COMMA type_specifier
               | type_specifier ID
               | type_specifier
               ;

compound_statement  : LCURL statements RCURL
                    | LCURL RCURL
                    ;

var_declaration : type_specifier declaration_list SEMICOLON
                ;

type_specifier  : INT
                | FLOAT
                | VOID
                ;

declaration_list : declaration_list COMMA ID
                 | declaration_list COMMA ID LTHIRD CONST_INT RTHIRD
                 | ID
                 | ID LTHIRD CONST_INT RTHIRD
                 ;

statements : statement
           | statements statement
           ;
```

```
statement : var_declaration
            | expression_statement
            | compound_statement
            | FOR LPAREN expression_statement expression_statement expression RPAREN
statement
          | IF LPAREN expression RPAREN statement
          | IF LPAREN expression RPAREN statement ELSE statement
          | WHILE LPAREN expression RPAREN statement
          | PRINTLN LPAREN ID RPAREN SEMICOLON
          | RETURN expression SEMICOLON
          ;

expression_statement : SEMICOLON
                       | expression SEMICOLON
                       ;

variable : ID
         | ID LTHIRD expression RTHIRD
         ;

expression : logic_expression
             | variable ASSIGNOP logic_expression
             ;

logic_expression : rel_expression
                   | rel_expression LOGICOP rel_expression
                   ;

rel_expression : simple_expression
                 | simple_expression RELOP simple_expression
                 ;

simple_expression : term
                    | simple_expression ADDOP term
                    ;

term : unary_expression
     | term MULOP unary_expression
     ;

unary_expression : ADDOP unary_expression
                   | NOT unary_expression
                   | factor
                   ;
```

```
factor : variable
        | ID LPAREN argument_list RPAREN
        | LPAREN expression RPAREN
        | CONST_INT
        | CONST_FLOAT
        | variable INCOP
        | variable DECOP
        ;

argument_list : arguments
                |
                ;

arguments : arguments COMMA logic_expression
            | logic_expression
            ;
```