BNF

<program> ::= <statement>

<statement> ::= <decla_statement> <statement> | <if_stm> <statement>

| <for_clause> <statement> | <assignment_statement> <statement>

| <write_stmt> <statement> | <arithmetic_expression> <statement>

| vacio

<decla_statement> ::= <simple_variable> ;

<simple_variable> ::= VAR <identifier_list>

<identifier_list> ::= ID <identifier_list2>

<identifier_list2> ::= , ID <identifier_list2> | [NUM | ID] <identifier_list2> | vacio

<assignment_statement> ::= <identifier_list> = <assignment_statement2> ;

<assignment_statement2> ::= <arithmetic_expression> | ID | NUM | LITERAL

<arithmetic_expression> ::= <termino> <arithmetic_expression2>

<arithmetic_expression2> ::= <addop> <termino> <arithmetic_expression2> |vacio

<addop> ::= +|-

<termino> ::= <signo> <factor> <termino2>

<termino2> ::= <mulop> <signo> <factor> <termino2> |vacio

<signo> ::= -| vacio

<mulop> ::= * | /

<factor> ::= (<arithmetic_expression>) | NUM

<for_clause> ::= FOR ID = <for_list> TO <for_list> STEP <for_list> DO <statement> END FOR ;

<for_list> ::= <arithmetic_expression> | ID | NUM

<if_stm> ::= IF <condition> THEN <statement> <if_blocks> END IF ;

<if_blocks> ::= ELSE <if_blocks2> | vacio

<if_blocks2> ::= <statement> | <if_stm>

<relational_op> ::= == | <> | < | > | >= | <=

<condition> ::= <expresion> <relational_op expresion>

<expresion> ::= <arithmetic_expression> | ID | NUM | LITERAL

<write_stmt> ::= WRITE (<expresion>) ; | WRITELN(<expresion>);