Grammar

Start Symbol : <program>
<program> → <moduleDeclarations> <otherModules><driverModule><otherModules>
<moduleDeclarations> → <moduleDeclaration><moduleDeclarations> | ε
<moduleDeclaration> → DECLARE MODULE ID SEMICOL
<otherModules> → <module><otherModules>| ε
<driverModule> → DEF DRIVER PROGRAM ENDDEF <moduleDef>
<module> → DEF MODULE ID ENDDEF TAKES INPUT SQBO <input_plist> SQBC SEMICOL
<ret><moduleDef>
<input_plist> → ID COLON <dataType> <input_plist'>
<input_plist'> →COMMA ID COLON <dataType><input_plist'> | ε
<dataType> → INTEGER | REAL | BOOLEAN | ARRAY SQBO <range> SQBC OF <type>
<type>→ INTEGER | REAL | BOOLEAN
<ret> → RETURNS SQBO <output_plist> SQBC SEMICOL | ε
<output_plist> → ID COLON <type><output_plist'>
<output_plist'> → COMMA ID COLON <type><output_plist'> | ε
<moduleDef>→ START <statements> END
<statements> → <statement> <statements> | ε
<statement> → <ioStmt>|<simpleStmt>|<declareStmt>|<condionalStmt>|<iterativeStmt>
<ioStmt> → GET_VALUE BO ID <whichId> BC SEMICOL | PRINT BO <var> BC SEMICOL
<var> → ID <whichId> | NUM | RNUM
<whichId> → SQBO <index> SQBC | ε
<simpleStmt> → <assignmentStmt> | <moduleReuseStmt>
<assignmentStmt> → ID <whichStmt>
<whichStmt> → <lvalueIDStmt> | <lvalueARRStmt>
<lvalueIDStmt> → ASSIGNOP <expression> SEMICOL
<lvalueARRStmt> → SQBO <index> SQBC ASSIGNOP <expression> SEMICOL
<index> →NUM | ID
<moduleReuseStmt> → <optional> USE MODULE ID WITH PARAMETERS <idList>SEMICOL
<optional> → SQBO <idList> SQBC ASSIGNOP | ε
<idList> → ID ><idList'>
<idList'> →COMMA ID <idList'> | ε
<expression> → <arithmeticExpr> | <booleanExpr>
<arithmeticExpr> → <term><arithmeticExpr'>
<arithmeticExpr'> →PLUS <term> <arithmeticExpr'> | MINUS <term> <arithmeticExpr'> | ε
<term> →<factor> <term'>
<term'> →MUL <factor> <term'> | DIV <factor> <term'> | ε
<factor> → BO <arithmeticExpr> BC | <var>
<booleanExpr> → <arithmeticExpr> <relationalOp> <arithmeticExpr> <booleanExpr'> | BO
<booleanExpr> BC <booleanExpr'>
<booleanExpr'> →<logicalOp><booleanExpr><booleanExpr'> | ε
<relationalOp> → LT | LE | GT | GE | EQ | NE
<declareStmt> → DECLARE <idList> COLON <dataType> SEMICOL
<conditionalStmt> → SWITCH BO ID BC START <caseStmt><default> END
<caseStmt> →CASE <value> COLON <statements> BREAK SEMICOL <caseStmt> | ε
<value> → NUM | TRUE | FALSE
<default> → DEFAULT COLON <statements> BREAK SEMICOL | ε
<iterativeStmt> → FOR BO ID IN <range> BC START <statements> END | WHILE BO
<booleanExpr> BC START <statements> END
<range> →NUM RANGEOP NUM