FIRST AND FOLLOW SETS

Non-Terminal	First Set	Follow Set
<program>	{DECLARE, DEF, DRIVERDEF}	{$}
<moduleDeclarations>	{DECLARE, e}	{DEF, DRIVERDEF}
<moduleDeclaration>	{DECLARE}	{DECLARE, DEF, DRIVERDEF}
<otherModules>	{DEF, e}	{DRIVERDEF, $}
<module>	{DEF}	{DEF, DRIVERDEF, $}
<ret>	{RETURNS, e}	{START}
<input_plist>	{ID}	{SQBC}
<input_plist_dash>	{COMMA, e}	{SQBC}
<output_plist>	{ID}	{SQBC}
<output_plist_dash>	{COMMA, e}	{SQBC}
<dataType>	{INTEGER, BOOLEAN, REAL, ARRAY}	{COMMA, SQBC, SEMICOL}
<range_array>	{NUM, ID}	{SQBC}
<type>	{INTEGER, REAL, BOOLEAN}	{COMMA, SQBC, SEMICOL}
<driverModule>	{DRIVERDEF}	{DEF, $}
<moduleDef>	{START}	{DEF, DRIVERDEF,$}
<statements>	{GET_VALUE, PRINT, ID, SQBO, USE, DECLARE, SWITCH, FOR, WHILE, e}	{BREAK, END}
<statement>	{GET_VALUE, PRINT, ID, SQBO, USE, DECLARE, SWITCH, FOR, WHILE}	{GET_VALUE, PRINT, ID, USE, SQBO, DECLARE, SWITCH, FOR, WHILE, BREAK, END}

<declareStmt>	{DECLARE}	{GET_VALUE, USE, SQBO, DECLARE, SWITCH, FOR, WHILE, BREAK, END}
<idList>	{ID}	{COLON, SEMICOL, SQBC}
<idList_dash>	{COMMA, e}	{COLON, SEMICOL, SQBC}
<ioStmt>	{GET_VALUE, PRINT}	{GET_VALUE, PRINT, ID, USE, SQBO, DECLARE, SWITCH, FOR, WHILE, BREAK, END}
<var>	{ID, NUM, RNUM, TRUE, FALSE}	{BC, MUL, DIV, PLUS, MINUS, GT ,LT , GE , LE , EQ, NE,AND,OR, SEMICOL}
<whichId>	{SQBO,e}	{BC. MUL,DIV,PLUS, MINUS, GT ,LT , GE , LE , EQ , NE,AND,OR, SEMICOL}
<simpleStmt>	{ID, SQBO, USE}	{GET_VALUE, PRINT, ID, USE, SQBO, DECLARE, SWITCH,FOR,WHILE, BREAK,END}
<assignmentStmt>	{ID}	{GET_VALUE, PRINT, ID, USE, SQBO, DECLARE, SWITCH, FOR, WHILE, BREAK, END}
<whichStmt>	{ASSIGNOP, SQBO}	{GET_VALUE, PRINT, ID, USE, SQBO, DECLARE, SWITCH, FOR, WHILE, BREAK, END}
<lvalueIDStmt>	{ASSIGNOP}	{GET_VALUE, PRINT, ID USE, SQBO, DECLARE, SWITCH, FOR, WHILE, BREAK, END}
<lvalueARRStmt>	{SQBO}	{GET_VALUE, PRINT, ID, USE, SQBO, DECLARE, SWITCH, FOR, WHILE, BREAK, END}
<index>	{NUM, ID}	{SQBC, RANGEOP}

<moduleReuseStmt>	{SQBO, USE}	{GET_VALUE, PRINT, ID, USE, SQBO, DECLARE, SWITCH, FOR, WHILE, BREAK, END}
<optional>	{SQBO, e}	{USE}
<expression>	{BO, ID, NUM, RNUM, TRUE, FALSE, MINUS, PLUS}	{SEMICOL}
<arithmeticOrBooleanExpression>	{BO, ID, NUM, RNUM, TRUE, FALSE}	{SEMICOL, BC}
<unaryOrExpr>	{BO,ID,NUM,RNUM}	{SEMICOL}
<arithmetic_bool>	{BO, ID, NUM, RNUM, TRUE, FALSE}	{AND, OR, SEMICOL, BC}
<bool>	{AND, OR, e}	{SEMICOL, BC}
<arithmeticExpre_dash>	{GT, LT, GE, LE, EQ, NE, e}	{AND, OR, SEMICOL, BC}
<arithmeticExpr>	{BO, ID, NUM, RNUM, TRUE, FALSE}	{GT, LT, GE, LE, EQ, NE, AND, OR, BC, SEMICOL}
<arithmeticExpr_recur>	{PLUS, MINUS, e}	{GT, LT, GE, LE, EQ, NE, AND, OR, BC, SEMICOL}
<term>	{BO, ID, NUM, RNUM, TRUE, FALSE}	{PLUS, MINUS, GT, LT, GE , LE, EQ, NE, AND, OR, BC, SEMICOL}
<term_dash>	{MUL, DIV, e}	{PLUS, MINUS, GT, LT, GE , LE, EQ, NE, AND, OR, BC, SEMICOL}
<factor>	{BO, ID, NUM, RNUM, TRUE, FALSE}	{MUL, DIV, PLUS, MINUS, GT ,LT , GE , LE , EQ , NE, AND, OR, BC, SEMICOL}
<op_plus_minus>	{PLUS, MINUS}	{BO, ID, NUM, RNUM, TRUE, FALSE}
<op_mul_div>	{MUL, DIV}	{BO, ID, NUM, RNUM, TRUE, FALSE}

<logicalOp>	{AND, OR}	{BO, ID, NUM, RNUM, TRUE, FALSE}
<relationalOp>	{GT, LT , GE , LE , EQ , NE}	{BO, ID, NUM, RNUM, TRUE, FALSE}
<conditionalStmt>	{SWITCH}	{GET_VALUE, PRINT, ID, USE, SQBO, DECLARE, SWITCH, FOR, WHILE, BREAK, END}
<caseStmts>	{CASE}	{DEFAULT, END}
<caseStmt>	{CASE, e}	{DEFAULT, END}
<value>	{NUM, TRUE, FALSE}	{COLON}
<default>	{DEFAULT, e}	{END}
<iterativeStmt>	{FOR, WHILE}	{GET_VALUE, PRINT, ID, USE, SQBO, DECLARE, SWITCH, FOR, WHILE, BREAK, END}
<range>	{NUM}	{BC}