Modified Grammar - Group 33

Team Members:

- Akshit Khanna (2017A7PS0023P)
- Swadesh Vaibhav (2017A7PS0030P)
- Aryan Mehra (2017A7PS0077P)
- Vipin Baswan (2017A7PS0429P)

Steps taken:

- Complete the grammar
- Remove ambiguities
- Remove Left-recursion
- Remove Left-factoring

Assumptions:

- Unary + and - can only be at the beginning of expression (i.e. they should be the first character in the expression)
- The type checking for condition in **while** (i.e. iterative statement) will be done during semantic analysis

---

The final grammar is as follows:

1. <program> ---- <moduleDeclarations> <otherModules> <driverModule> <otherModules>
2. <moduleDeclarations> ---- <moduleDeclaration><moduleDeclarations> | ***ε***
3. <moduleDeclaration> ---- **DECLARE MODULE ID SEMICOL**
4. <otherModules> ---- <module> <otherModules> | ***ε***
5. <driverModule> ---- **DRIVERDEF DRIVER PROGRAM DRIVERENDDEF** <moduleDef>
6. <module> ---- **DEF MODULE ID ENDDEF TAKES INPUT SQBO** <input_plist> **SQBC SEMICOL** <ret><moduleDef>
7. <ret> ---- **RETURNS SQBO** <output_plist> **SQBC SEMICOL** | ***ε***
8. <input_plist> ---- **ID COLON** <dataType> <input_plist2>
9. <input_plist2> ---- **COMMA ID COLON** <dataType> <input_plist2> | ***ε***
10. <output_plist> ---- **ID COLON** <type> <output_plist2>
11. <output_plist2> ---- **COMMA ID COLON** <type> <output_plist2> | ***ε***
12. <dataType> ---- **INTEGER | REAL | BOOLEAN | ARRAY SQBO** <range1> **SQBC OF** <type>
13. <type> ---- **INTEGER | REAL | BOOLEAN**
14. <moduleDef> ---- **START** <statements> **END**
15. <statements> ---- <statement> <statements> | ***ε***
16. <statement> ---- <ioStmt>|<simpleStmt>|<declareStmt>|<conditionalStmt>|<iterativeStmt>
17. <ioStmt> ---- **GET_VALUE BO ID BC SEMICOL** | **PRINT BO** <varAndBool> **BC SEMICOL**

18. <varAndBool> ---- <var> | <boolConst>
19. <var> ---- **ID** <whichId> | **NUM** | **RNUM**
20. <whichId> ---- **SQBO** <index> **SQBC** | ***ε***
21. <simpleStmt> ---- <assignmentStmt> | <moduleReuseStmt>
22. <assignmentStmt> ---- **ID** <whichStmt>
23. <whichStmt> ---- <lvalueIDStmt> | <lvalueARRStmt>
24. <lvalueIDStmt> ---- **ASSIGNOP** <expression> **SEMICOL**
25. <lvalueARRStmt> ---- **SQBO** <index> **SQBC ASSIGNOP** <expression> **SEMICOL**
26. <index> ---- **NUM** | **ID**
27. <moduleReuseStmt> ---- <optional> **USE MODULE ID WITH PARAMETERS** <idList>**SEMICOL**
28. <optional> ---- **SQBO** <idList> **SQBC ASSIGNOP** | ***ε***
29. <idList> ---- **ID** <idList2>
30. <idList2> ---- **COMMA ID** <idList2> | ***ε***
31. <expression> ---- <expression2> | <unaryExprArithmetic>
32. <expression2> ---- <logicalExpr> <expression3> | <boolConst> <expression3>
33. <expression3> ---- <logicalOp> <expression2> | ***ε***
34. <logicalExpr> ---- <arithmeticExpr> <logicalExpr2>
35. <logicalExpr2> ---- <relationalOp> <arithmeticExpr> | ***ε***
36. <arithmeticExpr> ---- <term> <arithmeticExpr2>
37. <arithmeticExpr2> ---- <op1> <term> <arithmeticExpr2> | ***ε***
38. <term> ---- <factor> <term2>
39. <term2> ---- <op2> <factor> | ***ε***
40. <factor> ---- **BO** <expression2> **BC** | <var>
41. <unaryExprArithmetic> ---- <op1> <myOptions>
42. <myOptions> ---- <var> | **BO** <arithmeticExprBoolnt> **BC**
43. <arithmeticExprBoolnt> ---- <termBoolnt> <arithmeticExpr2Boolnt>
44. <arithmeticExpr2Boolnt> ---- <op1> <termBoolnt> <arithmeticExpr2Boolnt> | ***ε***
45. <termBoolnt> ---- <factorBoolnt> <term2Boolnt>
46. <term2Boolnt> ---- <op2> <factorBoolnt> | ***ε***
47. <factorBoolnt> ---- **BO** <arithmeticExprBoolnt> **BC** | <var>
48. <op1> ---- **PLUS** | **MINUS**
49. <op2> ---- **MUL** | **DIV**
50. <relationalOp> ---- **LT** | **LE** | **GT** | **GE** | **EQ** | **NE**
51. <logicalOp> ---- **AND** | **OR**
52. <boolConst> ---- **TRUE** | **FALSE**
53. <declareStmt> ---- **DECLARE** <idList> **COLON** <dataType> **SEMICOL**
54. <conditionalStmt> ---- **SWITCH BO ID BC START** <caseStmt><default> **END**
55. <caseStmt> ---- **CASE** <value> **COLON** <statements> **BREAK SEMICOL** <caseStmts>
56. <caseStmts> ---- **CASE** <value> **COLON** <statements> **BREAK SEMICOL** <caseStmts> | ***ε***
57. <value> ---- **NUM** | **TRUE** | **FALSE**
58. <default> ---- **DEFAULT COLON** <statements> **BREAK SEMICOL** | ***ε***
59. <iterativeStmt> ---- **FOR BO ID IN** <range2> **BC START** <statements> **END** | **WHILE BO** <expression> **BC START** <statements> **END**
60. <range1> ---- **NUM RANGEOP ID** | **ID RANGEOP** <index>
61. <range2> ---- **NUM RANGEOP NUM**

First and Follow Sets - Group 33

NT	First	Follow
<program>	{DECLARE, DEF, DRIVERDEF}	{EOF}
<moduleDeclaration>	{DECLARE}	{DRIVERDEF, DECLARE, DEF}
<moduleDeclarations>	{DECLARE, ε }	{DRIVERDEF, DEF}
<otherModules>	{DEF, ε }	{DRIVERDEF, EOF}
<module>	{DEF}	{DEF, DRIVERDEF, EOF}
<moduleDef>	{START}	{DEF, DRIVERDEF, EOF}
<ret>	{RETURNS, ε }	{START}
<input_plist>	{ID}	{SQBC}
<output_plist>	{ID}	{SQBC}
<dataType>	{INTEGER, REAL, BOOLEAN, ARRAY}	{COMMA, SEMICOL, SQBC}
<type>	{INTEGER, REAL, BOOLEAN}	{COMMA, SEMICOL, SQBC}
<input_plist2>	{COMMA, ε }	{SQBC}
<output_plist2>	{COMMA, ε }	{SQBC}
<range1>	{NUM, ID}	{SQBC}
<statement>	{GET_VALUE, PRINT, ID, SQBO, USE, DECLARE, SWITCH, FOR, WHILE}	{GET_VALUE, PRINT, ID, SQBO, USE, DECLARE, SWITCH, FOR, WHILE, END, BREAK}
<statements>	{GET_VALUE, PRINT, ID, SQBO, USE, DECLARE, SWITCH, FOR, WHILE, ε }	{END, BREAK}
<ioStmt>	{GET_VALUE, PRINT}	{GET_VALUE, PRINT, ID, SQBO, USE, DECLARE, SWITCH, FOR, WHILE, END, BREAK}
<simpleStmt>	{ID, SQBO, USE}	{GET_VALUE, PRINT, ID, SQBO, USE, DECLARE, SWITCH, FOR, WHILE, END, BREAK}
<declareStmt>	{DECLARE}	{GET_VALUE, PRINT, ID, SQBO, USE, DECLARE, SWITCH, FOR, WHILE, END, BREAK}

<conditionalStmt>	{SWITCH}	{GET_VALUE, PRINT, ID, SQBO, USE, DECLARE, SWITCH, FOR, WHILE, END, BREAK}
<iterativeStmt>	{FOR, WHILE}	{GET_VALUE, PRINT, ID, SQBO, USE, DECLARE, SWITCH, FOR, WHILE, END, BREAK}
<var>	{ID, NUM, RNUM}	{MUL, DIV, BO, PLUS, MINUS, LT, LE, GT, GE, EQ, NE, SEMICOL, BC}
<whichId>	{SQBO, ε }	{MUL, DIV, BO, PLUS, MINUS, LT, LE, GT, GE, EQ, NE, SEMICOL, BC}
<assignmentStmt>	{ID}	{GET_VALUE, PRINT, ID, SQBO, USE, DECLARE, SWITCH, FOR, WHILE, END, BREAK}
<whichStmt>	{ASSIGNOP, SQBO}	{GET_VALUE, PRINT, ID, SQBO, USE, DECLARE, SWITCH, FOR, WHILE, END, BREAK}
<moduleReuseStmt>	{SQBO, USE}	{GET_VALUE, PRINT, ID, SQBO, USE, DECLARE, SWITCH, FOR, WHILE, END, BREAK}
<lvalueIDStmt>	{ASSIGNOP}	{GET_VALUE, PRINT, ID, SQBO, USE, DECLARE, SWITCH, FOR, WHILE, END, BREAK}
<lvalueARRStmt>	{SQBO}	{GET_VALUE, PRINT, ID, SQBO, USE, DECLARE, SWITCH, FOR, WHILE, END, BREAK}
<index>	{NUM, ID}	{SQBC}
<expression>	{TRUE, FALSE, BO, ID, NUM, RNUM, PLUS, MINUS}	{SEMICOL, BC}
<optional>	{SQBO, ε }	{USE}
<idList>	{ID}	{SQBC, SEMICOL, COLON}
<idList2>	{COMMA, ε }	{SQBC, SEMICOL, COLON}
<logicalExpr>	{BO, ID, NUM, RNUM}	{AND, OR, BC, SEMICOL}
<expression2>	{BO, ID, NUM, RNUM, TRUE, FALSE}	{BC, SEMICOL}
<logicalOp>	{AND, OR}	{BO, ID, NUM, RNUM, TRUE, FALSE}
<arithmeticExpr>	{BO, ID, NUM, RNUM}	{AND, OR, SEMICOL, BC, LT, LE, GT, GE, EQ, NE}
<logicalExpr2>	{LT, LE, GT, GE, EQ, NE, ε }	{AND, OR, BC, SEMICOL}
<relationalOp>	{LT, LE, GT, GE, EQ, NE}	{BO, ID, NUM, RNUM}

<arithmeticExpr2>	{PLUS, MINUS, ε }	{AND, OR, SEMICOL, BC, LT, LE, GT, GE, EQ, NE}
<op1>	{PLUS, MINUS}	{BO, ID, NUM, RNUM}
<term>	{BO, ID, NUM, RNUM}	{PLUS, MINUS, AND, OR, SEMICOL, BC, LT, LE, GT, GE, EQ, NE}
<factor>	{BO, ID, NUM, RNUM}	{MUL, DIV, PLUS, MINUS, AND, OR, SEMICOL, BC, LT, LE, GT, GE, EQ, NE}
<term2>	{MUL, DIV, ε }	{PLUS, MINUS, AND, OR, SEMICOL, BC, LT, LE, GT, GE, EQ, NE}
<op2>	{MUL, DIV}	{BO, ID, NUM, RNUM}
<caseStmt>	{CASE}	{DEFAULT, END}
<default>	{DEFAULT, ε }	{END}
<value>	{NUM, TRUE, FALSE}	{COLON}
<caseStmts>	{CASE, ε }	{DEFAULT, END}
<range2>	{NUM}	{BC}
<varAndBool>	{ID, NUM, RNUM, TRUE, FALSE}	{BC}
<boolConst>	{TRUE, FALSE}	{AND, OR, BC, SEMICOL}
<unaryExprArithmetic>	{PLUS, MINUS}	{SEMICOL, BC}
<expression3>	{AND, OR, ε }	{BC, SEMICOL}
<driverModule>	{DRIVERDEF}	{DEF, EOF}
<arithmeticExpr2BoolInt>	{PLUS, MINUS, ε }	{BC}
<termBoolInt>	{BO, ID, NUM, RNUM}	{PLUS, MINUS, BC}
<factorBoolInt>	{BO, ID, NUM, RNUM}	{MUL, DIV, PLUS, BC}
<term2BoolInt>	{MUL, DIV, ε }	{PLUS, MINUS, BC}
<arithmeticExprBoolInt>	{BO, ID, NUM, RNUM}	{BC}
<myOptions>	{BO, ID, NUM, RNUM}	{BC, SEMICOL}