GRAMMAR

1. **<program>** → <moduleDeclarations><otherModules><driverModule><otherModules>

2. **<moduleDeclarations>** → <moduleDeclaration><moduleDeclarations> | ε

3. **<moduleDeclaration>** → DECLARE MODULE ID SEMICOL

4. **<otherModules>** → <module><otherModules> | ε

5. **<module>** → DEF MODULE ID ENDDEF TAKES INPUT SQBO <input_plist> SQBC SEMICOL <ret> <moduleDef>

6. **<ret>** → RETURNS SQBO <output_plist> SQBC SEMICOL | ε

7. **<input_plist>** → ID COLON <dataType> <input_plist_dash>

8. **<input_plist_dash>** → COMMA ID COLON <dataType><input_plist_dash> | ε

9. **<output_plist>** → ID COLON <type> <output_plist_dash>

10. **<output_plist_dash>** → COMMA ID COLON <type><output_plist_dash> | ε

11. **<dataType>** → INTEGER | BOOLEAN | REAL | ARRAY SQBO <range_array> SQBC OF <type>

12. **<range_array>** → <index> RANGEOP <index>

13. **<type>** → INTEGER | REAL | BOOLEAN

14. **<driverModule>** → DRIVERDEF DRIVER PROGRAM DRIVERENDDEF <moduleDef>

15. **<moduleDef>** → START <statements> END

16. **<statements>** → <statement><statements> | ε

17. **<statement>** → <ioStmt> | <simpleStmt> | <declareStmt> | <conditionalStmt> | <iterativeStmt>

18. **<declareStmt>** → DECLARE <idList> COLON <dataType> SEMICOL

19. **<idList>** → ID <idList_dash>

20. **<idList_dash>** → COMMA ID <idList_dash> | ε

21. **<ioStmt>** → GET_VALUE BO ID BC SEMICOL | PRINT BO <var> BC SEMICOL

22. <**var**> → ID <whichId> | NUM | RNUM | TRUE | FALSE

23. <**whichId**> → SQBO <index> SQBC | ε

24. <**simpleStmt**> → <assignmentStmt> | <moduleReuseStmt>

25. <**assignmentStmt**> → ID <whichStmt>

26. <**whichStmt**> → <lvalueIDStmt> | <lvalueARRStmt>

27. <**lvalueIDStmt**> → ASSIGNOP <expression> SEMICOL

28. <**lvalueARRStmt**> → SQBO <index> SQBC ASSIGNOP <expression> SEMICOL

29. <**index**> → NUM | ID

30. <**moduleReuseStmt**> → <optional> USE MODULE ID WITH PARAMETERS <idList> SEMICOL

31. <**optional**> → SQBO <idList> SQBC ASSIGNOP | ε

32. <**expression**> → <arithmeticOrBooleanExpression> | <op_plus_minus> <unaryOrExpr>

33.<**unaryOrExpr**> → BO <arithmeticExpr> BC | ID | NUM | RNUM

34. <**arithmeticOrBooleanExpression**> → <arithmetic_bool> <bool>

35. <**arithmetic_bool**> → <arithmeticExpr> <arithmeticExpre_dash>

36. <**bool**> → <logicalOp> <arithmetic_bool> <bool> | ε

37. <**arithmeticExpre_dash**> → <relationalOp> <arithmeticExpr> <arithmeticExpre_dash> | ε

38. <**arithmeticExpr**> → <term> <arithmeticExpr_recur>

39. <**arithmeticExpr_recur**> → <op_plus_minus> <term> <arithmeticExpr_recur> | ε

40. <**term**> → <factor> <term_dash>

41. <**term_dash**> → <op_mul_div> <factor> <term_dash> | ε

42. <**factor**> → BO <arithmeticExpr> BC | <var>

43. <**op_plus_minus**> → PLUS | MINUS

44. <**op_mul_div**> → MUL | DIV

45. <**logicalOp**> → AND | OR

46. <**relationalOp**> → GT | LT | GE | LE | EQ | NE

47 **<conditionalStmt>** → SWITCH BO ID BC START <caseStmts> <default> END

48. **<caseStmts>** → CASE <value> COLON <statements> BREAK SEMICOL <caseStmt>

49. **<caseStmt>** → CASE <value> COLON <statements> BREAK SEMICOL <caseStmt> | ε

50. **<value>** → NUM | TRUE | FALSE

51. **<default>** → DEFAULT COLON <statements> BREAK SEMICOL | ε

52. **<iterativeStmt>** → FOR BO ID IN <range> BC START <statements> END | WHILE BO <arithmeticOrBooleanExpression> BC START <statements> END

53. **<range>** → NUM RANGEOP NUM

ASSUMPTIONS

1. The print statement can print ID, constant, true, false and any indices of an array.
2. We do not allow empty statements, i.e. statements having only a semicolon.
3. Negated expressions such as **-(a+b)** etc. are allowed, where a,b are identifiers.
4. We have used <range_array> for RANGEOP operation of array where the limits can be integers or identifiers and <range> for inside the **for loop** where the limits can be only integers.