Grammar for MINI-L Language

Program:

Prog_start -> Functions

Functions - > **Function Functions** | ε

Function:

Function - > “function” “identifier” “;” “beginparams” **Declarations** “endparams” “beginparams” **Declarations** “endlocals” “beginbody” **Statements** “endbodys”

Declarations -> Declaration “;” **Declarations | Declaration** “;” | ε

Statements -> **Statement** “;” **Statements** | **Statement** “;”

Declaration:

Declaration -> **Identifiers** “:” **Array_id** “array” “[” “number” “]” “of” “integer” |

Identifiers “:” “integer”

;

Identifiers -> “identifier” | “identifier” “,” **Identifiers**

Statement:

Statement -> Var “:=” **Expression**

| if” **Bool-Exp** “then” **Statements** “else” **Statements** “endif”

| “if” **Bool-Exp** “then” **Statements** “endif”

| “while” **Bool-Exp** “beginloop” **Statements** “endloop”

| “do” “beginloop” **Statements** “endloop” “while” **Bool-Exp**

| “read” **Vars**

| “write” **Vars**

| “return” **Expression**

;

Bool-Expr:

Bool-Expr - >

Relation-And-Expr | Relation-And-Expr “or” **Bool-Expr**

Relation-And-Expr:

Relation-And-Expr->
Relation-Expr | Relation-Expr “and” **Relation-And-Expr**

Relation-Expr:

Relation-Expr -> “not”“Expression” “Comp” “Expression”
| “not”“true”
| “not”“false”
| “not”“(” **Bool-Expr**“)”
|“Expression” “Comp” “Expression”
|“true”
| “false”
| “(” **Bool-Expr**“)”

Comp:

Comp -> “==” | “<>” | “<” | “>” | “<=” | “>=”

Expression:

Expression -> Multiplicative-Expr
| **Expression** “+” **Multiplicative-Expr**
| **Expression** “-” **Multiplicative-Exp**

Multiplicative-Expr:

Multiplicative-Expr -> Multiplicative-Expr “%” **Term**
| **Multiplicative-Expr** “/” **Term**
| **Multiplicative-Expr** “*” **Term**
|**Term**

Term:

Term -> “-” Var
| “-” “number”

|"-" "(" **Expression**")"
| "identifier" "(" **Ex** ")"
;

Ex -> Expression "," **Ex| Expression** | ε

Var:

Vars -> Var "," Vars | **Var**
Var -> "identifier" | "identifier" "[" **Expression** "]"