Grammar for MINI-L Language

Program:

Program - > **Function Program** | ε

Function:

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

Function -> “function” “identifier” “;” “beginparams” (**Declaration** “;”) * “endparams” “beginlocals” (**Declaration** “;”)* “endlocals” “beginbody” (**Statement** “;”) * “endbody”

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

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

Declaration:

Declaration -> **Identifiers** “:” **Arrays** “integer”

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

Arrays -> “array” “[” “number” “]” “of” | ε

Statement:

Statement -> A|B|C|D|E|F|G|H|I

A-> **Var** “:=” **Expression**

B-> “if” **Bool-Exp** “then” **States ElseStates**“endif”

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

ElseStates -> “else” **States** | ε

C-> “while” **Bool-Exp** “beginloop” **States** “endloop”

D-> “do” “beginloop” **States** “endloop” “while” **Bool-Exp**

E-> “read” **Vars**

G-> “write” **Vars**

H-> “continue”

I-> “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” **Re-Ex | Re-Ex**
Re-Ex -> **Expressions |** “true” | “false” | “(” **Bool-Expr**“)”
Expressions -> “Expression” “Comp” “Expression”

Comp:

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

Expression:

Expression -> Multiplicative-Expr Expre
Expre->
“+”**Multiplicative-Expr Expre** | “-” **Multiplicative-Expr Expre**| ε

Multiplicative-Expr:

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

Term:

Term -> Pos-term | “-” **Pos-term** | **ide**
ide -> “identifier” “(” **Ex** “)”
Ex -> Expression “,” **Ex** | ε

Pos-term -> Var | “number” | “(” **Expression**“)”

Var:

Vars -> Var “,” Vars | Var
Var -> “identifier” | “identifier” “[” **Expression** “]”