Mini-I Grammar

PROGRAM:
Program -> **Program function** | empty

FUNCTION:
Function -> “function” “identifier” “;” “beginparams” **dec_loop** “endparams” “beginlocals” **dec_loop** “endlocals” “beginbody” **statement_loop** “endbody”

dec_loop -> dec_loop **Declaration** “;” | empty

Statement_loop -> **statement** “;” | **statement_loop statement** “;”

DECLARATION:
Declaration -> **id_loop** “:” **assignment**
Id_loop -> “identifier” | **id_loop** “,” “identifier”
assignment -> “integer” | “array” “[“ “number” “]” “of” “integer” |
“array” “[“ “number” “]” “[“ “number” “]” “of” “integer”

STATEMENT:
Statement -> A|B|C|D|E|F|G|H|I
A -> Var “:=” **Expression**
B -> “if” **Bool-Exp** “then” **Statement_loop** “endif” | “if” **Bool-Exp** “then” **Statement_loop** “else” **Statement_loop** “endif”
C -> “while” **Bool-Exp** “beginloop” **Statement_loop** “endloop”
D -> “do” “beginloop” **Statement_loop** “endloop” “while” **Bool-Exp**
E -> “for” **Var** “:=” “number” “;” **Bool-Exp** “;” **Var** “:=” **Expression** “beginloop” **Statement_loop** “endloop”
F -> “read” **Var_Loop**
Var_Loop -> Var “,” **Var_Loop** | **Var**
G -> “write” **Var_Loop**
H -> “continue”
I -> “return” **Expression**

Bool-Expr:
Bool-Expr -> Relation-And-Expr B’
B’ -> B’ “or” **Relation-And-Expr |** Empty

Relation-And-Expr:
Relation-And-Expr -> Relation-Expr RAE’
RAE’ -> RAE’ “and” **Relation-Expr |** Empty

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

COMP:

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

EXPRESSION:

Expression -> Multiplicative-Expr Expression’

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

Multiplicative-Expr:

Multiplicative-Expr -> Term Multi’

Multi’ -> Multi’ “*” **Term** | **Multi’**“/” **Term | Multi’**“%” **Term |** empty

Term:

Term -> “-” **Term’ | Term’ |** “identifier” “(“ **Expression_loop** “)”

Term’ -> Var | “number” | “(“ **Expression** “)”

Expression_loop -> Expression “,” **Expression_loop** | **Expression |** empty

Var:

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