Program:

Program → Function | Program | ε

Function:

Function → "function" "identifier" ";" "beginparams" Dec "endparams" "beginlocals" Dec "endlocals" "beginbody" Statements "endbody"

Dec → Declaration ";" Dec | ε

Declaration:

Declaration → A

A → "identifier" B

B → "identifier" "," A | ":" C

C → "integer" | "array" "[" "number" "]" "of" "integer"

Statement:

Statement → A|B|C|D|E|F|G|H

A → Var ":=" Expression

B → "if" Bool-Exp "then" Statement "endif" | "if" Bool-Exp "then" Statement "else" Statement "endif"

C → "while" Bool-Exp "beginloop" Statement "endloop"

D → "do" "beginloop" Statement "endloop" "while" Bool-Exp

E → "read" Var E'

E' → "," Var E' | ε

F → "write" Var E'

G → "continue"

H → "return" Expression

Bool-Expr:

Bool-Expr → A

A → Relation-Expr B

B → Relation-Expr "or" B | ε

Relation-And-Expr:

Relation-And-Expr → A

A → Relation-Expr B

B → Relation-Expr "and" A | ε

Relation-Expr:

Relation-Expr → A | "not" A

A → B | C | D | E

B → Expression Comp Expression
C → "true"
D → "false"
E → "(" Bool-Expr ")"

Comp:
Comp → A
A → "==" | "<>" | "<" | ">" | "<=" | ">="

Expression:
Expression → A
A → Multiplicative-Expression B
B → "+" Multiplicative-Expression B | "-" Multiplicative-Expression B | ε

Multiplicative-Expr:
Multiplicative-Expr → A
A → Term B
B → "*" Term B | " /"Term B | " %" Term B | ε

Term:
Term → AB | C
A → "-" | ε
B → Var | number | "("Expression")"
C → "identifier" "(" D ")"
D → Expression | Expression "," D | ε

Var:
Var → A | B
A → "identifier"
B → "identifier" "[" Expression "]"