```
/*program*/
program->functions


/*function*/
functions -> function functions| epsilon
function ->                                                                    ↵
  "function"identidier";""beginparams"declaration"endparams""beginlocals""endlocal ↵
  s""beginbody"statement"endbody"

/*identidier*/
identifiers -> ident
            |ident "," identifiers

/*declararion*/
declarations->declaration";"declarations | epsilon
declaration->identifiers":""array""[""number""]""of""integer" |                ↵
  identifiers":""integer"

/*statement*/
statements->statement";"statements |e
statement->var":="expression
| "if"bool-expr"then" statement"else"statement"endif"
| "while"bool-expr"beginloop"statement"endloop"
| "do""beginloop"statement"endloop""while"bool-expr
| "read"v
| "write"v
| "break"
| "return"expresstion
v->var | var","v

/*bool-expr*/
bool-expr->relation-and-expr| relation-and-expr"or"bool-expr

/*relation-and-expr*/
relation-and-expr -> relation-expr|relation-expr"or"relation-and-expr

/*relation-expr*/
relation-expr -> "not"r|r
r-> expresstion comp expression
    |true
    |false
    |"("bool-expr")"

/*comp*/
comp -> "=="
        |"<>"
        |"<"
        |">"
        |"<="
        |">="

```

```
/*expression*/
expression -> multiplicative-expr|multiplicative-expr"+"expression|multiplicative-
  expr"-"expression

/*multiplicative-expr*/
multiplicative-expr -> term|term"*"multiplicative-expr|term"/"multiplicative-
  expr|"%"multiplicative-expr

/*term*/
term -> "-"t
        |t
        |"identifier""("expression")"
t -> var
    |"number"
    |"("expression")"

/*var*/
var -> "identifier"
       |"identifier""["expression"]"
```