```
function      ⇒  \begin { eqcode } { id }
                   { [id [ , id ]* ] }
                   { [ext_type [ , ext_type ]* ] } { ext_type }
                  instr_list
                   \end { eqcode }
indexes       ⇒  ( [upper ] [lower ] | lower upper )
idx           ⇒  id indexes
numx          ⇒  num
              |  divide
idx_numx      ⇒  ( idx | numx )
upper         ⇒  ˆ { ( [ linear ] | linear ) }
              |  ˆ ( id | num )
linear        ⇒  id [( + | - ) num ]
              |  num
lower         ⇒  _ { sexpr [ , sexpr ]* }
              |  _ ( id | num )
type          ⇒  \type { ( Z | R | N | B ) }
ext_type      ⇒  type [ ˆ ( { sexpr } | num | id )
                   [ _ ( { sexpr [ , sexpr ]* } ] ] | id | num )
instr_list    ⇒  [instr \lend ]*
instr         ⇒  assign
              |  declare
              |  with_loop
              |  return
assign        ⇒  idx \gets expr
declare       ⇒  idx \in ext_type
```

```
boolop           ⇒   \land
                 |   \lor
                 |   \oplus
binop            ⇒   +
                 |   -
                 |   \cdot
                 |   \ll
                 |   \gg
                 |   \mod
divide           ⇒  ( \frac | \dfrac ) { expr } { expr }
function_call    ⇒   \call { id } { [idx_numx [ , idx_numx ]* ] }
sexpr            ⇒  ( \lnot | - ) sexpr_op [( binop | boolop ) sexpr_op ]*
                 |   ( sexpr )
                 |   { sexpr }
sexpr_op         ⇒  ( idx_numx | function_call )
filter           ⇒   \filter { id ^ { [ id ] }
                      [ , id ^ { [ id ] } ]*
                       | generator }
genarray         ⇒   \genar \limits ^ { sexpr } ( sexpr )
vector           ⇒   \begin { tvector }
                      [sexpr \lend ]+
                       \end { tvector }
matrix           ⇒   \begin { tmatrix } { id }
                      [sexpr [ & sexpr ]* \lend ]+
                       \end { tmatrix }
expr             ⇒  ( sexpr | filter | genarray | vector | matrix ) indexes
with_loop        ⇒  idx | generator \gets ( expr | with_loop_cases )
with_loop_cases  ⇒   \begin { cases }
                      [expr & generator ]+
                      [expr & \otherwise ]+
                       \end { cases }
```

return ⇒ **\return** { *expr* }

generator ⇒ **\forall** ***id*** [**,** ***id***]*
| ***id*** [**,** ***id***]* **:** *sexpr* [*comp sexpr*]+
[*set_op sexpr* [*comp sexpr*]+]*

comp ⇒ **<**
| **>**
| **\leq**
| **\geq**
| [**\not**] **=**

set_op ⇒ (**\land** | **\lor**)