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) indexes*

idx_numx ⇒ *(idx | numx)*

upper ⇒ ^ { *(* **[** *(linear | expr)* **]** *| expr)* }
| ^ *(* ***id*** *|* ***num*** *)*

linear ⇒ *(* **\iter** *[(* **+** *|* **-** *)* ***num*** *]|* ***num*** *)*

lower ⇒ _ { *expr [* **,** *expr]** }
| _ *(* **id** *|* **num** *)*

type ⇒ **\type** { *(* **Z** *|* **R** *|* **N** *|* **B** *)* }

arraytype ⇒ **\arraytype** { *(sexpr |* **\ldots** *) [* **,** *(sexpr |* **\ldots** *)]* }
{ *(* **Z** *|* **R** *|* **N** *|* **B** *)* }

functiontype ⇒ **(** *[ext_type]** **\to** *[ext_type]+* **)**

ext_type ⇒ *(type | arraytype | functiontype)*

instr_list ⇒ *[instr* **\lend** *]**

instr ⇒ *assign*
| *declare*
| *index_loop*
| *comment*
| *if_cond*
| *return*

if_cond ⇒ **\qif** { *cond_block* }
instr_list
[**\qelseif** { *cond_block* }
instr_list] [* **\qelse**
*instr_list]** **\qendif**

cond_block ⇒ *expr [comp expr]+*
*[set_op expr [comp expr]+]**

assign ⇒ *idx* [, *idx*]* \gets *expr* [, *expr*]*

declare ⇒ *idx* [, *idx*]* \in *ext_type* [, *ext_type*]*

boolop ⇒ \land
| \lor
| \oplus

binop ⇒ +
| -
| \cdot
| \ll
| \gg
| \mod

divide ⇒ (\frac | \dfrac) { *expr* } { *expr* }

function_call ⇒ \call { ***id*** } { [*expr* [, *expr*]*] }

sexpr ⇒ (\lnot | -) *sexpr_op* [(*binop* | *boolop*) *sexpr_op*]*
| (*expr*)
| { *expr* }

sexpr_op ⇒ (***idx_numx*** | *function_call* | *matrix*)

filter ⇒ \filter { ***id*** ^ { [***id***] }
[, ***id*** ^ { [***id***] }]*
| *generator* }

genarray ⇒ \genar \limits ^ { *expr* } (*expr*)

matrix ⇒ \begin { tmatrix }
[*expr* [& *expr*]* \lend]+
\end { tmatrix }

expr ⇒ (*sexpr* | *filter* | *genarray*) *indexes*

index_loop ⇒ *idx* | *generator* \gets (*expr* | *index_loop_cases*)

index_loop_cases ⇒ \begin { cases }
[*expr* & *generator*]+
[*expr* & \otherwise]+
\end { cases }

print ⇒ **\print** { *expr* }

return ⇒ **\return** { *expr* }

generator ⇒ **\forall** ***id*** [**,** ***id***]*

| ***id*** [**,** ***id***]* **:** *cond_block*

comp ⇒ **<**

| **>**

| **\leq**

| **\geq**

| [**\not**] **=**

set_op ⇒ (**\cup** | **\cap**)