

<i>function</i>	$\Rightarrow$	$\begin{aligned} & \backslash\textbf{begin} \{ \textbf{eqcode} \} \{ \textbf{id} \} \\ & \{ [id [ , id ]^* ] \} \\ & \{ [ext\_type [ , ext\_type ]^* ] \} \{ ext\_type \} \\ & instr\_list \\ & \backslash\textbf{end} \{ \textbf{eqcode} \} \end{aligned}$
<i>indexes</i>	$\Rightarrow$	$( [upper] [lower] \mid lower\ upper )$
<i>idx</i>	$\Rightarrow$	$\textbf{id} \ indexes$
<i>numx</i>	$\Rightarrow$	$( \textbf{num} \mid divide ) \ indexes$
<i>idx\_numx</i>	$\Rightarrow$	$( idx \mid numx )$
<i>upper</i>	$\Rightarrow$	$\begin{aligned} & \wedge \{ ( [ ( linear \mid expr ) ] \mid expr ) \} \\ & \mid \\ & \wedge ( \textbf{id} \mid \textbf{num} ) \end{aligned}$
<i>linear</i>	$\Rightarrow$	$( \backslash\textbf{iter} [ ( + \mid - ) \textbf{num} ] \mid \textbf{num} )$
<i>lower</i>	$\Rightarrow$	$\begin{aligned} & - \{ expr [ , expr ]^* \} \\ & \mid \\ & - ( \textbf{id} \mid \textbf{num} ) \end{aligned}$
<i>type</i>	$\Rightarrow$	$\backslash\textbf{type} \{ ( \textbf{Z} \mid \textbf{R} \mid \textbf{N} \mid \textbf{B} ) \}$
<i>ext\_type</i>	$\Rightarrow$	$\begin{aligned} & type [ \wedge ( \{ seexpr \} \mid \textbf{num} \mid \textbf{id} ) \\ & [ - ( \{ seexpr [ , seexpr ]^* \} ) ] \mid \textbf{id} \mid \textbf{num} ) \end{aligned}$
<i>instr\_list</i>	$\Rightarrow$	$[instr \ \backslash\textbf{lend} ]^*$
<i>instr</i>	$\Rightarrow$	$\begin{aligned} & assign \\ & \mid \\ & declare \\ & \mid \\ & index\_loop \\ & \mid \\ & comment \\ & \mid \\ & if\_cond \\ & \mid \\ & return \end{aligned}$
<i>if\_cond</i>	$\Rightarrow$	$\begin{aligned} & \backslash\textbf{qif} \{ cond\_block \} \\ & instr\_list \\ & [ \backslash\textbf{qelseif} \{ cond\_block \} \\ & instr\_list ]^* [ \backslash\textbf{qelse} \\ & instr\_list ]^* \backslash\textbf{qendif} \end{aligned}$
<i>cond\_block</i>	$\Rightarrow$	$\begin{aligned} & expr [comp\ expr ]^+ \\ & [set\_op\ expr [comp\ expr ]^+ ]^* \end{aligned}$
<i>assign</i>	$\Rightarrow$	$idx [ , idx ]^* \backslash\textbf{gets} \ expr [ , expr ]^*$
<i>declare</i>	$\Rightarrow$	$idx [ , idx ]^* \backslash\textbf{in} \ ext\_type [ , ext\_type ]^*$

<i>boolop</i>	$\Rightarrow$	$\backslash\text{land}$   $\backslash\text{lor}$   $\backslash\text{oplus}$
<i>binop</i>	$\Rightarrow$	$+$   $-$   $\backslash\text{cdot}$   $\backslash\text{ll}$   $\backslash\text{gg}$   $\backslash\text{mod}$
<i>divide</i>	$\Rightarrow$	$( \backslash\text{frac} \mid \backslash\text{dfrac} ) \{ \text{expr} \} \{ \text{expr} \}$
<i>function_call</i>	$\Rightarrow$	$\backslash\text{call} \{ \text{id} \} \{ [ \text{expr} [ , \text{expr} ]^* ] \}$
<i>sexpr</i>	$\Rightarrow$	$( \backslash\text{lnot} \mid - ) \text{sexpr\_op} [ ( \text{binop} \mid \text{boolop} ) \text{sexpr\_op} ]^*$   $( \text{expr} )$   $\{ \text{expr} \}$
<i>sexpr_op</i>	$\Rightarrow$	$( \text{idx\_numx} \mid \text{function\_call} \mid \text{matrix} )$
<i>filter</i>	$\Rightarrow$	$\backslash\text{filter} \{ \text{id} \wedge \{ [ \text{id} ] \} \}$ $[ , \text{id} \wedge \{ [ \text{id} ] \} ]^*$   $\text{generator} \}$
<i>genarray</i>	$\Rightarrow$	$\backslash\text{genar} \backslash\text{limits} \wedge \{ \text{expr} \} ( \text{expr} )$
<i>matrix</i>	$\Rightarrow$	$\backslash\text{begin} \{ \text{tmatrix} \}$ $[ \text{expr} [ \& \text{expr} ]^* \backslash\text{lend} ] +$ $\backslash\text{end} \{ \text{tmatrix} \}$
<i>expr</i>	$\Rightarrow$	$( \text{sexpr} \mid \text{filter} \mid \text{genarray} ) \text{indexes}$
<i>index_loop</i>	$\Rightarrow$	$\text{idx} \mid \text{generator} \backslash\text{gets} ( \text{expr} \mid \text{index\_loop\_cases} )$
<i>index_loop_cases</i>	$\Rightarrow$	$\backslash\text{begin} \{ \text{cases} \}$ $[ \text{expr} \& \text{generator} ]^+$ $[ \text{expr} \& \backslash\text{otherwise} ] +$ $\backslash\text{end} \{ \text{cases} \}$

<i>print</i>	$\Rightarrow$	<code>\print { expr }</code>
<i>return</i>	$\Rightarrow$	<code>\return { expr }</code>
<i>generator</i>	$\Rightarrow$	<code>\forall id [ , id ]*</code>
		<code>id [ , id ]* : cond_block</code>
<i>comp</i>	$\Rightarrow$	<code>&lt;</code>
		<code>&gt;</code>
		<code>\leq</code>
		<code>\geq</code>
		<code>[ \not ] =</code>
<i>set_op</i>	$\Rightarrow$	<code>( \cup   \cap )</code>