α, β, γ	cursor symbols			
d	data/content symbols			
formula	::=			
	|	*judgement*		
	|	α fresh		
s	::=			atomic symbol, a single unit of information
	|	α		cursor
	|	d		data/content
Z	::=			Symbol zipper
	|	$\langle S_1 \parallel \alpha \parallel S_2 \rangle$		Consists of symbols to left (S_1) and right (S_2) of active cursor α
	|	rev(Z)	M	The symbol zipper Z in reverse order (flipped left and right)
dir	::=			Zipper direction
	|	L		
	|	R		
c	::=			Command
	|	ins d *dir*		Insert d to direction *dir*
	|	rem *dir*		Remove next data symbol in direction *dir*
	|	move *dir*		Move the cursor over the data symbol to direction *dir*
	|	repl d *dir*		Replace next data symbol in direction *dir* with d
	|	mk α		Make a passive cursor α at the position of the active cursor.
	|	switch α		Switch active cursor to cursor α
	|	jmp α		Jump active cursor to position of cursor α
	|	join α		Join active cursor to the identity and position of cursor α.
C	::=			Command sequence
	|	ϵ		
	|	$c :: C$		
	|	$C :: c$		
	|	rev(C)	M	Command sequence C, in reverse order.
Z_{C}	::=			Command zipper
	|	$\langle C_1 \parallel C_2 \rangle$		Consists of command history C_1 and undo buffer C_2.
a	::=			Action
	|	cmd c		Perform command c
	|	undo		Undo previous command action.
	|	redo		Undo previous undo action, redoing undone command action.
A	::=			Action sequence
	|	ϵ		
	|	$A :: a$		
$\bar{\alpha}$	::=			Sequence of cursors
	|	$\alpha\, \bar{\alpha}$		
	|	ϵ		

T	::=		Unfocused symbol tree.
	|	ϵ	Empty tree
	|	s	Symbol
	|	$\langle\!\langle \bar{\alpha}_1\ T_1 \parallel \bar{\alpha}_2\ T_2 \rangle\!\rangle$	Binary tree; $\bar{\alpha}_1$ are cursors in T_1; $\bar{\alpha}_2$ are cursors in T_2
S	::=		Symbol sequence
	|	ϵ	
	|	$s :: S$	
	|	$S :: s$	
	|	$\mathsf{seq}(T)$	Sequence of tree T
J	::=		
	|	$Z_1 \leftrightarrow Z_2$	Zipper Z_1 refocuses to Z_2 in zero or more steps.
	|	$Z_1 \vdash c \longrightarrow Z_2$	Under zipper Z_1, performing command c yields zipper Z_2
	|	$Z_1 \vdash C \Downarrow Z_2$	Under zipper Z_1, performing command sequence C yields zipper
	|	$A \Downarrow Z_{\mathsf{C}}$	Performing action sequence A yields command zipper Z_{C}
	|	$A \Downarrow Z$	Performing action sequence A yields symbol zipper Z
Jdet	::=		
	|	$T_1; T_2 \Downarrow_{dir}^{\mathsf{next}} S$	From he *dir*-most position; the symbol sequence T_1 followed by
judgement	::=		
	|	J	
	|	*Jdet*	
user_syntax	::=		
	|	α	
	|	d	
	|	*formula*	
	|	s	
	|	Z	
	|	*dir*	
	|	c	
	|	C	
	|	Z_{C}	
	|	a	
	|	A	
	|	$\bar{\alpha}$	
	|	T	
	|	S	

$\boxed{Z_1 \leftrightarrow Z_2}$ Zipper Z_1 refocuses to Z_2 in zero or more steps.

$$\frac{}{Z \leftrightarrow Z}\ \textsc{Mv\_stop}$$

$$\frac{\langle S_1 \parallel \alpha \parallel s :: S_2\rangle \leftrightarrow Z}{\langle S_1 :: s \parallel \alpha \parallel S_2\rangle \leftrightarrow Z}\ \textsc{Mv\_left}$$

$$\frac{\langle S_1 :: s \parallel \alpha \parallel S_2\rangle \leftrightarrow Z}{\langle S_1 \parallel \alpha \parallel s :: S_2\rangle \leftrightarrow Z}\ \textsc{Mv\_right}$$

$\boxed{Z_1 \vdash c \longrightarrow Z_2}$ Under zipper Z_1, performing command c yields zipper Z_2

$$\frac{}{\langle S_1 \parallel \alpha \parallel S_2\rangle \vdash \mathsf{ins}\ d\ \mathsf{L} \longrightarrow \langle S_1 :: d \parallel \alpha \parallel S_2\rangle}\ \textsc{Ec\_insertL1}$$

$$\frac{\langle S_1 \parallel \alpha \parallel S_2\rangle \vdash \mathsf{ins}\ d\ \mathsf{L} \longrightarrow \langle S_1' \parallel \alpha \parallel S_2'\rangle}{\langle S_1 :: \beta \parallel \alpha \parallel S_2\rangle \vdash \mathsf{ins}\ d\ \mathsf{L} \longrightarrow \langle S_1' :: \beta \parallel \alpha \parallel S_2'\rangle}\ \textsc{Ec\_insertL2}$$

$$\frac{}{\langle S_1 :: d \parallel \alpha \parallel S_2\rangle \vdash \mathsf{rem}\ \mathsf{L} \longrightarrow \langle S_1 \parallel \alpha \parallel S_2\rangle}\ \textsc{Ec\_removeL1}$$

$$\frac{\langle S_1 \parallel \alpha \parallel S_2\rangle \vdash \mathsf{rem}\ \mathsf{L} \longrightarrow \langle S_1' \parallel \alpha \parallel S_2'\rangle}{\langle S_1 :: \beta \parallel \alpha \parallel S_2\rangle \vdash \mathsf{rem}\ \mathsf{L} \longrightarrow \langle S_1' :: \beta \parallel \alpha \parallel S_2'\rangle}\ \textsc{Ec\_removeL2}$$

$$\frac{}{\langle S_1 :: d \parallel \alpha \parallel S_2\rangle \vdash \mathsf{move}\ \mathsf{L} \longrightarrow \langle S_1 \parallel \alpha \parallel d :: S_2\rangle}\ \textsc{Ec\_moveL1}$$

$$\frac{\langle S_1 \parallel \alpha \parallel \beta :: S_2\rangle \vdash \mathsf{move}\ \mathsf{L} \longrightarrow Z}{\langle S_1 :: \beta \parallel \alpha \parallel S_2\rangle \vdash \mathsf{move}\ \mathsf{L} \longrightarrow Z}\ \textsc{Ec\_moveL2}$$

$$\frac{\begin{array}{c}Z_1 \vdash \mathsf{rem}\ \mathsf{L} \longrightarrow Z_2\\ Z_2 \vdash \mathsf{ins}\ d\ \mathsf{L} \longrightarrow Z_3\end{array}}{Z_1 \vdash \mathsf{repl}\ d\ \mathsf{L} \longrightarrow Z_3}\ \textsc{Ec\_replaceL}$$

$$\frac{\mathsf{rev}(Z) \vdash \mathsf{ins}\ d\ \mathsf{L} \longrightarrow \mathsf{rev}(Z')}{Z \vdash \mathsf{ins}\ d\ \mathsf{R} \longrightarrow Z'}\ \textsc{Ec\_insertR}$$

$$\frac{\mathsf{rev}(Z) \vdash \mathsf{rem}\ \mathsf{L} \longrightarrow \mathsf{rev}(Z')}{Z \vdash \mathsf{rem}\ \mathsf{R} \longrightarrow Z'}\ \textsc{Ec\_removeR}$$

$$\frac{\mathsf{rev}(Z) \vdash \mathsf{move}\ \mathsf{L} \longrightarrow \mathsf{rev}(Z')}{Z \vdash \mathsf{move}\ \mathsf{R} \longrightarrow Z'}\ \textsc{Ec\_moveR}$$

$$\frac{\mathsf{rev}(Z) \vdash \mathsf{repl}\ d\ \mathsf{L} \longrightarrow \mathsf{rev}(Z')}{Z \vdash \mathsf{repl}\ d\ \mathsf{R} \longrightarrow Z'}\ \textsc{Ec\_replaceR}$$

$$\frac{\begin{array}{c}\gamma\ \mathsf{fresh}\\ \langle S_1 :: \alpha \parallel \gamma \parallel S_2\rangle \leftrightarrow \langle S_1' :: \beta \parallel \gamma \parallel S_2'\rangle\end{array}}{\langle S_1 \parallel \alpha \parallel S_2\rangle \vdash \mathsf{switch}\ \beta \longrightarrow \langle S_1' \parallel \beta \parallel S_2'\rangle}\ \textsc{Ec\_switchto}$$

$$\frac{\langle S_1 \parallel \alpha \parallel S_2\rangle \leftrightarrow \langle S_1' :: \beta \parallel \alpha \parallel S_2'\rangle}{\langle S_1 \parallel \alpha \parallel S_2\rangle \vdash \mathsf{jmp}\ \beta \longrightarrow \langle S_1' :: \beta \parallel \alpha \parallel S_2'\rangle}\ \textsc{Ec\_jumpto}$$

$$\frac{\langle S_1 \parallel \alpha \parallel S_2\rangle \leftrightarrow \langle S_1' :: \beta \parallel \alpha \parallel S_2'\rangle}{\langle S_1 \parallel \alpha \parallel S_2\rangle \vdash \mathsf{join}\ \beta \longrightarrow \langle S_1' \parallel \beta \parallel S_2'\rangle}\ \textsc{Ec\_join}$$

$$\frac{}{\langle S_1 \parallel \alpha \parallel S_2\rangle \vdash \mathsf{mk}\ \beta \longrightarrow \langle S_1 :: \beta \parallel \alpha \parallel S_2\rangle}\ \textsc{Ec\_mk}$$

$\boxed{Z_1 \vdash C \Downarrow Z_2}$ Under zipper Z_1, performing command sequence C yields zipper Z_2

$$\frac{}{Z \vdash \epsilon \Downarrow Z}\ \textsc{EC\_nil}$$

$$\frac{\begin{array}{c}Z_1 \vdash c \longrightarrow Z_2\\ Z_2 \vdash C \Downarrow Z_3\end{array}}{Z_1 \vdash c :: C \Downarrow Z_3}\ \textsc{EC\_cons}$$

$\boxed{A \Downarrow Z_{\mathsf{C}}}$ Performing action sequence A yields command zipper Z_{C}

$$\frac{}{\epsilon \Downarrow \langle \epsilon \parallel \epsilon\rangle}\ \textsc{EAC\_nil}$$

$$\frac{A \Downarrow \langle C_1 :: c \parallel C_2 \rangle}{A :: \mathsf{undo} \Downarrow \langle C_1 \parallel c :: C_2 \rangle}\ \textsc{eac\_undo}$$

$$\frac{A \Downarrow \langle C_1 \parallel c :: C_2 \rangle}{A :: \mathsf{redo} \Downarrow \langle C_1 :: c \parallel C_2 \rangle}\ \textsc{eac\_redo}$$

$$\frac{A \Downarrow \langle C_1 \parallel C_2 \rangle}{A :: \mathsf{cmd}\ c \Downarrow \langle C_1 :: c \parallel \epsilon \rangle}\ \textsc{eac\_cmd}$$

$\boxed{A \Downarrow Z}$ Performing action sequence A yields symbol zipper Z

$$\frac{\begin{array}{l}\alpha\ \mathsf{fresh} \\ A \Downarrow \langle C_1 \parallel C_2 \rangle \\ \langle \epsilon \parallel \alpha \parallel \epsilon \rangle \vdash \mathsf{rev}(C_1) \Downarrow Z\end{array}}{A \Downarrow Z}\ \textsc{Zofa}$$

$\boxed{T_1; T_2 \Downarrow_{dir}^{\mathsf{next}} S}$ From he *dir*-most position; the symbol sequence T_1 followed by T_2 is S.

$$\frac{}{\epsilon; \epsilon \Downarrow_{dir}^{\mathsf{next}} \epsilon}\ \textsc{next2\_nil1}$$

$$\frac{T; \epsilon \Downarrow_{dir}^{\mathsf{next}} S}{\epsilon; T \Downarrow_{dir}^{\mathsf{next}} S}\ \textsc{next2\_nil2}$$

$$\frac{}{s; T \Downarrow_{dir}^{\mathsf{next}} s :: \mathsf{seq}(T)}\ \textsc{next2\_sym}$$

$$\frac{T_1; \langle\!\langle \epsilon\ T_2 \parallel \epsilon\ T_3 \rangle\!\rangle \Downarrow_{\mathsf{L}}^{\mathsf{next}} S}{\langle\!\langle \bar{\alpha}_1\ T_1 \parallel \bar{\alpha}_2\ T_2 \rangle\!\rangle ; T_3 \Downarrow_{\mathsf{L}}^{\mathsf{next}} S}\ \textsc{next2\_binL}$$

$$\frac{T_2; \langle\!\langle \epsilon\ T_3 \parallel \epsilon\ T_1 \rangle\!\rangle \Downarrow_{\mathsf{R}}^{\mathsf{next}} S}{\langle\!\langle \bar{\alpha}_1\ T_1 \parallel \bar{\alpha}_2\ T_2 \rangle\!\rangle ; T_3 \Downarrow_{\mathsf{R}}^{\mathsf{next}} S}\ \textsc{next2\_binR}$$

```
Definition rules:         30 good    0 bad
Definition rule clauses: 56 good    0 bad
```