termvar, x	term variable
variant, V	variant
typvar, X	type variable
exc, *Exc*	exception
effect, E	effect
n	
m	

program	::=			
	|			
	|	*top program*		
	|	$[x \mapsto t]$*program*	M	
top	::=			
	|	**let** $x = t$		
	|	**let rec** $x : T = abs$		
	|	**type alias** $X = T$		
	|	**type** X *variantArgs* $= V_1$ *tyList*$_1 \,..\, \mid V_n$ *tyList*$_n$		
	|	**exception** *Exc tyList*		
t	::=			term
	|	x		variable
	|	V		type constructors
	|	$\lambda(x : T) \to t$		abstraction
	|	$\lambda(X : K) \to t$		type abstraction
	|	$\lambda(E : \varphi) \to t$		effect abstraction
	|	$t\ t'$		application
	|	$t[T]$		type application
	|	$t[[\mathit{eff}]]$		effect application
	|	**let** $x = t_1$ **in** t_2		let binding
	|	**let rec** $x : T = abs$ **in** t		recursive let binding
	|	**match** t **with** $p_1 \to t_1 \mid .. \mid p_n \to t_n$ **end**		pattern matching
	|	$t : [\mathit{eff}]\, T$		type annotation
	|	$t : T$	S	type annotation = (t : [] T)
	|	**fail** $[T]$ *Exc* $t_1 \,..\, t_n$		fail
	|	**try** t **with** $pe_1 \to t_1 \mid .. \mid pe_n \to t_n$ **end**		try
	|	$t; t'$	S	sequence == let (_ : Unit) = t in t'
	|	(t)	S	
	|	$[x_1 \mapsto t_1, \,.., x_n \mapsto t_n]t$	M	
	|	**failure** *exnval*	M	
	|	**TConstr** $V\ v_1 \,..\, v_n$	M	
v	::=			value
	|	**TConstr** $V\ v_1 \,..\, v_n$		type constructors
	|	$\lambda(x : T) \to t$		abstraction
abs	::=			lambda abstractions
	|	$\lambda(x : T) \to t$		abstraction
	|	$\lambda(X : K) \to abs$		type abstraction
	|	$\lambda(E : \varphi) \to abs$		effect abstraction
	|	$abs : [\mathit{eff}]\, T$		type annotation
	|	$abs : T$	S	type annotation = (abs : [] T)
p	::=			pattern
	|	$V\ p_1 \,..\, p_n$		variant

	$\mid$	x	wildcard variable
pe	::=		try pattern
	$\mid$	$Exc\, x_1 \,..\, x_n$	Exception pattern
$exnval$	::=		runtime value of exceptions
	$\mid$	$Exc\, v_1 \,..\, v_n$	
$effelm$	::=		effects elements
	$\mid$	E	effect
	$\mid$	**IO**	IO effect
	$\mid$	**Exn** $[exn]$	exception
exn	::=		exceptions
	$\mid$	$Exc_1 \mid .. \mid Exc_n$	
eff	::=		effect
	$\mid$	$effelm_1, .., effelm_n$	
	$\mid$	$eff_1 \cup eff_2 \cup .. \cup eff_n$	M
	$\mid$	$eff_1 \backslash [exn]$	M
	$\mid$	(eff)	S
K	::=		kinds
	$\mid$	$*$	star
	$\mid$	$K \to K'$	kind arrow
T	::=		type
	$\mid$	X	variable
	$\mid$	**Unit**	Unit type (contained in the module opened by defau
	$\mid$	$T \to T'$	S pure function == T -[]-¿ T'
	$\mid$	$T - [eff] -> T'$	function
	$\mid$	$\lambda(X : K), T$	operator abstraction
	$\mid$	$\forall\,(X : K), T$	forall
	$\mid$	$\forall\,(E : \varphi), T$	effect forall
	$\mid$	$T\ T'$	operator application
	$\mid$	(T)	S
	$\mid$	$[X \mapsto T]\, T'$	M
	$\mid$	$[E \mapsto eff]\, T$	M
	$\mid$	$T_1 \to .. \to T_n \to T$	M
	$\mid$	$T\ T_1 \,..\, T_n$	M
Γ	::=		type environment
	$\mid$	**empty**	empty
	$\mid$	$\Gamma, x_1 : T_1, .., x_n : T_n$	vars
	$\mid$	$\Gamma, V : T$	S type constructors (contained in the above values envi
	$\mid$	$\Gamma, X_1 : K_1, .., X_n : K_n$	tvars
	$\mid$	$\Gamma, X : \{ V_1\ tyList_1 \,..\, V_n\ tyList_n \}$	variants

	$\Gamma, \mathit{Exc}\ \mathit{tyList}$		exceptions
	Γ, E		effects
	$\Gamma_1 \cup .. \cup \Gamma_n$	M	

tyList ::=

	$T_1 .. T_n$

variantArgs ::=

	$(X_1 : K_1) .. (X_n : K_n)$

terminals ::=

	λ
	$\backslash$
	$\longrightarrow$
	$\rightarrow$
	$\vdash$
	$\mapsto$
	$\in$
	$\equiv$
	φ
	$\forall$
	$\cup$
	$\rhd$
	$\leftarrow$
	&

formula ::=

	judgement
	$\mathit{formula}_1 \quad .. \quad \mathit{formula}_n$
	$\mathbf{not}\,(\mathit{formula})$
	$T \equiv T'$
	$x : T \in \Gamma$
	$X : K \in \Gamma$
	$V : T \in \Gamma$
	$T : \{\mathit{Variant}\} \in \Gamma$
	$E \in \Gamma$
	$\mathit{Exc}\ \mathit{tyList} \in \Gamma$
	$\mathbf{set}\,(\mathit{eff}) = \mathbf{set}\,(\mathit{eff}')$
	$\mathbf{set}\,(\mathit{exn}) = \mathbf{set}\,(\mathit{exn}')$
	$V \in \mathit{Variant} \rhd T_1 .. T_n$
	$\mathit{RetVar} = \mathit{RetVar}'$
	$\Gamma = \Gamma'$

substs ::=

	$x_1 \leftarrow v_1, .., x_n \leftarrow v_n$	
	$\mathit{substs}_1 \cup .. \cup \mathit{substs}_n$	M

patterns	::=		
	|	$p_1 .. p_n$	
VArgs	::=		
	|	$T_1 .. T_n$	
Variant	::=		
	|	$V_1\ VArgs_1 .. V_n\ VArgs_n$	
	|	**empty**	
	|	$Variant \setminus V$	M
RetVar	::=		
	|	$X\ X_1 .. X_n$	
JProgram	::=		
	|	$\Gamma \vdash program \rhd \Gamma'$	Program typing
JTopType	::=		
	|	$\Gamma \vdash top \rhd \Gamma'$	Toplevel typing
JTypeDecl	::=		
	|	$RetVar\ \&\ \Gamma \vdash V\ tyList \rhd \Gamma'$	Type declaration
Jtype	::=		
	|	$\Gamma \vdash t : [eff]\,T$	Typing
Jkind	::=		
	|	$\Gamma \vdash T : K$	Kinding
JEff	::=		
	|	$\Gamma \vdash eff$	Effects typing
JEffElm	::=		
	|	$\Gamma \vdash effelm$	Effects elements typing
JPatternsTyping	::=		
	|	$Variant\ \&\ \Gamma \vdash patterns : T \rhd \Gamma_1 .. \Gamma_n$	Patterns matching typing
JPatternTyping	::=		
	|	$Variant\ \&\ \Gamma \vdash p : T \rhd \Gamma'$	Pattern matching typing
JExnPatternTyping	::=		
	|	$\Gamma \vdash pe \rhd Exc\ \&\ \Gamma'$	Exception pattern matching typin
Jequiv	::=		
	|	$T \equiv T'$	Type equivalence

JEffEquiv	::=		
	|	$\mathit{eff} \equiv \mathit{eff}'$	Effects equivalence
JEffElmEquiv	::=		
	|	$\mathit{effelm} \equiv \mathit{effelm}'$	Effect element equivalence
JTopOp	::=		
	|	$\mathit{program} \longrightarrow \mathit{program}'$	Toplevel evaluation
Jop	::=		
	|	$t \longrightarrow t'$	Evaluation
JExnMatches	::=		
	|	$\mathit{exnval}\ \mathbf{matches}\ \mathit{pe} \rhd \{x_1 \leftarrow v_1, \ldots, x_n \leftarrow v_n\}$	Exception pattern matching with s
JMatches	::=		
	|	$v\ \mathbf{matches}\ p \rhd \{\mathit{substs}\}$	Pattern matching with substitutio
judgement	::=		
	|	*JProgram*	
	|	*JTopType*	
	|	*JTypeDecl*	
	|	*Jtype*	
	|	*Jkind*	
	|	*JEff*	
	|	*JEffElm*	
	|	*JPatternsTyping*	
	|	*JPatternTyping*	
	|	*JExnPatternTyping*	
	|	*Jequiv*	
	|	*JEffEquiv*	
	|	*JEffElmEquiv*	
	|	*JTopOp*	
	|	*Jop*	
	|	*JExnMatches*	
	|	*JMatches*	
user_syntax	::=		
	|	*termvar*	
	|	*variant*	
	|	*typvar*	
	|	*exc*	
	|	*effect*	
	|	*n*	
	|	*m*	
	|	*program*	
	|	*top*	

| t
| v
| abs
| p
| pe
| $exnval$
| $effelm$
| exn
| eff
| K
| T
| Γ
| $tyList$
| $variantArgs$
| $terminals$
| $formula$
| $substs$
| $patterns$
| $VArgs$
| $Variant$
| $RetVar$

$\boxed{\Gamma \vdash program \rhd \Gamma'}$ Program typing

$$\frac{}{\Gamma \vdash \rhd \Gamma} \quad \textsc{Prog\_Empty}$$

$$\frac{\Gamma \vdash top \rhd \Gamma' \quad \Gamma' \vdash program \rhd \Gamma''}{\Gamma \vdash top\ program \rhd \Gamma''} \quad \textsc{Prog\_Program}$$

$\boxed{\Gamma \vdash top \rhd \Gamma'}$ Toplevel typing

$$\frac{\Gamma \vdash t : [\,]\,T}{\Gamma \vdash \mathbf{let}\, x = t \rhd \Gamma, x : T} \quad \textsc{Top\_Let}$$

$$\frac{\Gamma, x : T \vdash abs : [\,]\,T}{\Gamma \vdash \mathbf{let\ rec}\, x : T = abs \rhd \Gamma, x : T} \quad \textsc{Top\_LetRec}$$

$$\frac{\Gamma \vdash T : K}{\Gamma \vdash \mathbf{type\ alias}\, X = T \rhd \Gamma, X : K} \quad \textsc{Top\_TypeAlias}$$

$$\frac{\begin{array}{l} RetVar = X\ X_1\ ..\ X_n \\ \Gamma' = \Gamma, X : K, X_1 : K_1, .., X_n : K_n \\ RetVar\ \&\ \Gamma' \vdash V_1\ tyList_1 \rhd \Gamma_1 \quad .. \quad RetVar\ \&\ \Gamma' \vdash V_n\ tyList_n \rhd \Gamma_n \end{array}}{\Gamma \vdash \mathbf{type}\, X\ (X_1 : K_1)(X_n : K_n) = V_1\ tyList_1 | \,..\, | V_n\ tyList_n \rhd \Gamma \cup \Gamma_1 \cup .. \cup \Gamma_n, X : \{\, V_1\ tyList_1\ ..\ V_n\ tyList_n\}} \quad \textsc{T}$$

$$\frac{\Gamma \vdash T_1 : K_1 \quad .. \quad \Gamma \vdash T_n : K_n}{\Gamma \vdash \mathbf{exception}\, Exc\ T_1\ ..\ T_n \rhd \Gamma, Exc\ T_1\ ..\ T_n} \quad \textsc{Top\_Exception}$$

$\boxed{RetVar\ \&\ \Gamma \vdash V\ tyList \rhd \Gamma'}$ Type declaration

$$\frac{\Gamma \vdash T_1 : K_1 \quad .. \quad \Gamma \vdash T_n : K_n}{X\ X_1\ ..\ X_n\ \&\ \Gamma \vdash V\ T_1\ ..\ T_n \rhd \mathbf{empty}, V : T_1 \to .. \to T_n \to X\ X_1\ ..\ X_n} \quad \textsc{TyDecl\_Decl}$$

$\boxed{\Gamma \vdash t : [eff]\,T}$ Typing

$$\frac{x : T \in \Gamma}{\Gamma \vdash x : [\,]\,T} \quad \text{T\_Var}$$

$$\frac{V : T \in \Gamma}{\Gamma \vdash V : [\,]\,T} \quad \text{T\_Variant}$$

$$\frac{\begin{array}{c}\Gamma, x_1 : T_1 \vdash t : [\mathit{eff}]\,T \\ \Gamma \vdash T_1 : * \end{array}}{\Gamma \vdash \lambda(x_1 : T_1) \to t : [\,]\,T_1 - [\mathit{eff}] -> T} \quad \text{T\_Abs}$$

$$\frac{\begin{array}{c}\Gamma \vdash t : [\mathit{eff}_1]\,T_1 - [\mathit{eff}_2] -> T_2 \\ \Gamma \vdash t' : [\mathit{eff}_3]\,T_1 \end{array}}{\Gamma \vdash t\ t' : [\mathit{eff}_1 \cup \mathit{eff}_2 \cup \mathit{eff}_3]\,T_2} \quad \text{T\_App}$$

$$\frac{\Gamma, X : K \vdash t : [\mathit{eff}]\,T}{\Gamma \vdash \lambda(X : K) \to t : [\mathit{eff}] \forall\,(X : K),\, T} \quad \text{T\_TAbs}$$

$$\frac{\Gamma, E \vdash t : [\mathit{eff}]\,T}{\Gamma \vdash \lambda(E : \varphi) \to t : [\mathit{eff}] \forall\,(E : \varphi),\, T} \quad \text{T\_EAbs}$$

$$\frac{\begin{array}{c}\Gamma \vdash t : [\mathit{eff}] \forall\,(X : K),\, T_2 \\ \Gamma \vdash T_1 : K \end{array}}{\Gamma \vdash t[T_1] : [\mathit{eff}][X \mapsto T_1]\,T_2} \quad \text{T\_TApp}$$

$$\frac{\begin{array}{c}\Gamma \vdash t : [\mathit{eff}] \forall\,(E : \varphi),\, T \\ \Gamma \vdash \mathit{eff}' \end{array}}{\Gamma \vdash t[[\mathit{eff}']] : [\mathit{eff}][E \mapsto \mathit{eff}']\,T} \quad \text{T\_EApp}$$

$$\frac{\begin{array}{c}\Gamma \vdash t : [\mathit{eff}]\,X \\ X \equiv X' \\ \Gamma \vdash X' : * \end{array}}{\Gamma \vdash t : [\mathit{eff}]\,X'} \quad \text{T\_Eq}$$

$$\frac{\Gamma, x : T_1 \vdash t_2 : [\mathit{eff}_2]\,T_2}{\Gamma \vdash \mathbf{let}\, x = (t_1 : [\mathit{eff}_1]\,T_1)\, \mathbf{in}\, t_2 : [\mathit{eff}_1 \cup \mathit{eff}_2]\,T_2} \quad \text{T\_Let}$$

$$\frac{\begin{array}{c}\Gamma, x : T_1 \vdash \mathit{abs} : [\,]\,T_1 \\ \Gamma, x : T_1 \vdash t_2 : [\mathit{eff}]\,T_2 \end{array}}{\Gamma \vdash \mathbf{let\ rec}\, x : T_1 = \mathit{abs}\, \mathbf{in}\, t_2 : [\mathit{eff}]\,T_2} \quad \text{T\_LetRec}$$

$$\frac{\begin{array}{l}\Gamma \cup \Gamma_1 \vdash t_1 : [\mathit{eff}_1]\,T_2 \quad .. \quad \Gamma \cup \Gamma_n \vdash t_n : [\mathit{eff}_n]\,T_2 \\ T_1 : \{\,\mathit{Variant}\} \in \Gamma \\ \mathit{Variant}\ \&\ \Gamma \vdash p_1 \,..\, p_n : T_1 \rhd \Gamma_1 \,..\, \Gamma_n \\ \Gamma \vdash t : [\mathit{eff}]\,T_1 \end{array}}{\Gamma \vdash \mathbf{match}\, t\, \mathbf{with}\, p_1 \to t_1 | \,..\, | p_n \to t_n\, \mathbf{end} : [\mathit{eff} \cup \mathit{eff}_1 \cup \,..\, \cup \mathit{eff}_n]\,T_2} \quad \text{T\_Match}$$

$$\frac{\begin{array}{c}\Gamma \vdash t : [\mathit{eff}]\,T \\ \Gamma \vdash T : * \\ \Gamma \vdash \mathit{eff} \end{array}}{\Gamma \vdash (t : [\mathit{eff}]\,T) : [\mathit{eff}]\,T} \quad \text{T\_Annot}$$

$$\frac{\begin{array}{l}\Gamma \vdash t_1 : [\mathit{eff}_1]\,T_1 \quad .. \quad \Gamma \vdash t_n : [\mathit{eff}_n]\,T_n \\ \Gamma \vdash T : * \\ \mathit{Exc}\ T_1 \,..\, T_n \in \Gamma \end{array}}{\Gamma \vdash \mathbf{fail}\,[T]\mathit{Exc}\, t_1 \,..\, t_n : [\mathbf{Exn}\,[\mathit{Exc}] \cup \mathit{eff}_1 \cup \,..\, \cup \mathit{eff}_n]\,T} \quad \text{T\_Fail}$$

$$\frac{\begin{array}{l}\Gamma_1 \vdash t_1 : [\mathit{eff}_1]\,T \quad .. \quad \Gamma_n \vdash t_n : [\mathit{eff}_n]\,T \\ \Gamma \vdash \mathit{pe}_1 \rhd \mathit{Exc}_1 \;\&\; \Gamma_1 \quad .. \quad \Gamma \vdash \mathit{pe}_n \rhd \mathit{Exc}_n \;\&\; \Gamma_n \\ \Gamma \vdash t : [\mathit{eff}]\,T\end{array}}{\Gamma \vdash \mathbf{try}\; t\; \mathbf{with}\; \mathit{pe}_1 \to t_1 | \,..\, | \mathit{pe}_n \to t_n\; \mathbf{end} : [(\mathit{eff} \setminus [\mathit{Exc}_1 | \,..\, | \mathit{Exc}_n]) \cup \mathit{eff}_1 \cup \,..\, \cup \mathit{eff}_n]\,T}\ \textsc{T\_Try}$$

$\boxed{\Gamma \vdash T : K}$ Kinding

$$\frac{X : K \in \Gamma}{\Gamma \vdash X : K}\ \textsc{K\_TVar}$$

$$\frac{\Gamma, X : K_1 \vdash T : K_2}{\Gamma \vdash \lambda(X : K_1), T : K_1 \to K_2}\ \textsc{K\_Abs}$$

$$\frac{\begin{array}{l}\Gamma \vdash T_1 : K_{11} \to K_{12} \\ \Gamma \vdash T_2 : K_{11}\end{array}}{\Gamma \vdash T_1\; T_2 : K_{12}}\ \textsc{K\_App}$$

$$\frac{\begin{array}{c}\Gamma \vdash T_1 : * \\ \Gamma \vdash \mathit{eff} \\ \Gamma \vdash T_2 : *\end{array}}{\Gamma \vdash T_1 - [\mathit{eff}] -> T_2 : *}\ \textsc{K\_Arrow}$$

$$\frac{\Gamma, X : K_1 \vdash T_2 : *}{\Gamma \vdash \forall\,(X : K_1), T_2 : *}\ \textsc{K\_All}$$

$$\frac{\Gamma, E \vdash T : *}{\Gamma \vdash \forall\,(E : \varphi), T : *}\ \textsc{K\_EAll}$$

$\boxed{\Gamma \vdash \mathit{eff}}$ Effects typing

$$\frac{\Gamma \vdash \mathit{effelm}_1 \quad .. \quad \Gamma \vdash \mathit{effelm}_n}{\Gamma \vdash \mathit{effelm}_1, .., \mathit{effelm}_n}\ \textsc{Eff\_Eff}$$

$\boxed{\Gamma \vdash \mathit{effelm}}$ Effects elements typing

$$\frac{E \in \Gamma}{\Gamma \vdash E}\ \textsc{EffElm\_Eff}$$

$$\frac{}{\Gamma \vdash \mathbf{IO}}\ \textsc{EffElm\_IO}$$

$$\frac{\mathit{Exc}_1\; \mathit{tyList}_1 \in \Gamma \quad .. \quad \mathit{Exc}_n\; \mathit{tyList}_n \in \Gamma}{\Gamma \vdash \mathbf{Exn}\,[\mathit{Exc}_1 | \,..\, | \mathit{Exc}_n]}\ \textsc{EffElm\_Exn}$$

$\boxed{\mathit{Variant}\ \&\ \Gamma \vdash \mathit{patterns} : T \rhd \Gamma_1 \,..\, \Gamma_n}$ Patterns matching typing

$$\frac{\mathit{Variant}\ \&\ \Gamma \vdash p_1 : T \rhd \Gamma_1 \quad .. \quad \mathit{Variant}\ \&\ \Gamma \vdash p_n : T \rhd \Gamma_n}{\mathit{Variant}\ \&\ \Gamma \vdash p_1 \,..\, p_n : T \rhd \Gamma_1 \,..\, \Gamma_n}\ \textsc{PsTy\_Patterns}$$

$\boxed{\mathit{Variant}\ \&\ \Gamma \vdash p : T \rhd \Gamma'}$ Pattern matching typing

$$\frac{\begin{array}{l}V \in \mathit{Variant} \rhd T_1 \,..\, T_n \\ T_1 : \{\mathit{Variant}_1\} \in \Gamma \quad .. \quad T_n : \{\mathit{Variant}_n\} \in \Gamma \\ \mathit{Variant}_1\ \&\ \Gamma \vdash p_1 : T_1 \rhd \Gamma_1 \quad .. \quad \mathit{Variant}_n\ \&\ \Gamma \vdash p_n : T_n \rhd \Gamma_n\end{array}}{\mathit{Variant}\ \&\ \Gamma \vdash V\; p_1 \,..\, p_n : T \rhd \Gamma_1 \cup \,..\, \cup \Gamma_n}\ \textsc{PTy\_Variant}$$

$$\frac{}{\mathit{Variant}\ \&\ \Gamma \vdash x : T \rhd \mathbf{empty}, x : T}\ \textsc{PTy\_Wildcard}$$

$\boxed{\Gamma \vdash \mathit{pe} \rhd \mathit{Exc}\ \&\ \Gamma'}$ Exception pattern matching typing

$$\frac{Exc\ T_1\ ..\ T_n\ \in\ \Gamma}{\Gamma \vdash Exc\ x_1\ ..\ x_n \rhd Exc\ \&\ \Gamma, x_1 : T_1, .., x_n : T_n}\quad \textsc{PETy\_Exc}$$

$\boxed{T \equiv T'}$ Type equivalence

$$\frac{}{T \equiv T}\quad \textsc{Q\_Refl}$$

$$\frac{T \equiv T'}{T' \equiv T}\quad \textsc{Q\_Symm}$$

$$\frac{T_1 \equiv T_2 \quad T_2 \equiv T_3}{T_1 \equiv T_3}\quad \textsc{Q\_Trans}$$

$$\frac{T_{11} \equiv T_{21} \quad \mathit{eff}_1 \equiv \mathit{eff}_2 \quad T_{12} \equiv T_{22}}{T_{11} - [\mathit{eff}_1] -> T_{12} \equiv T_{21} - [\mathit{eff}_2] -> T_{22}}\quad \textsc{Q\_Arrow}$$

$$\frac{T_1 \equiv T_2}{\forall\,(X : K), T_1 \equiv \forall\,(X : K), T_2}\quad \textsc{Q\_All}$$

$$\frac{T_1 \equiv T_2}{\forall\,(E : \varphi), T_1 \equiv \forall\,(E : \varphi), T_2}\quad \textsc{Q\_EAll}$$

$$\frac{T_1 \equiv T_2}{\lambda(X : K), T_1 \equiv \lambda(X : K), T_2}\quad \textsc{Q\_Abs}$$

$$\frac{T_{11} \equiv T_{21} \quad T_{12} \equiv T_{22}}{T_{11}\ T_{12} \equiv T_{21}\ T_{22}}\quad \textsc{Q\_App}$$

$$\frac{}{(\lambda(X : K), T_{11})\ T_{12} \equiv [X \mapsto T_{12}]\,T_{11}}\quad \textsc{Q\_AppAbs}$$

$\boxed{\mathit{eff} \equiv \mathit{eff}'}$ Effects equivalence

$$\frac{}{\mathit{eff} \equiv \mathit{eff}}\quad \textsc{EffEq\_Refl}$$

$$\frac{\mathbf{set}\,(\mathit{effelm}_1, .., \mathit{effelm}_n) = \mathbf{set}\,(\mathit{effelm}'_1, .., \mathit{effelm}'_n)}{\mathit{effelm}_1, .., \mathit{effelm}_n \equiv \mathit{effelm}'_1, .., \mathit{effelm}'_n}\quad \textsc{EffEq\_Eq}$$

$\boxed{\mathit{effelm} \equiv \mathit{effelm}'}$ Effect element equivalence

$$\frac{}{\mathit{effelm} \equiv \mathit{effelm}}\quad \textsc{EffElmEq\_Refl}$$

$$\frac{\mathit{effelm} \equiv \mathit{effelm}'}{\mathit{effelm}' \equiv \mathit{effelm}}\quad \textsc{EffElmEq\_Symm}$$

$$\frac{\mathit{effelm}_1 \equiv \mathit{effelm}_2 \quad \mathit{effelm}_2 \equiv \mathit{effelm}_3}{\mathit{effelm}_1 \equiv \mathit{effelm}_3}\quad \textsc{EffElmEq\_Trans}$$

$$\frac{\mathbf{set}\,(Exc_1|\,..\,|Exc_n) = \mathbf{set}\,(Exc'_1|\,..\,|Exc'_n)}{\mathbf{Exn}\,[Exc_1|\,..\,|Exc_n] \equiv \mathbf{Exn}\,[Exc'_1|\,..\,|Exc'_n]}\quad \textsc{EffElmEq\_ExnEq}$$

$\boxed{\mathit{program} \longrightarrow \mathit{program}'}$ Toplevel evaluation

$$\frac{t \longrightarrow t'}{\textbf{let}\ x = t\ program \longrightarrow \textbf{let}\ x = t'\ program}\ \textsc{TopE\_Let1}$$

$$\frac{}{\textbf{let}\ x = v\ program \longrightarrow [x \mapsto v]program}\ \textsc{TopE\_Let2}$$

$$\frac{}{\textbf{let rec}\ x : T = abs\ program \longrightarrow [x \mapsto (\textbf{let rec}\ x : T = abs\ \textbf{in}\ abs)]program}\ \textsc{TopE\_LetRec}$$

$$\frac{}{\textbf{type alias}\ X = T\ program \longrightarrow program}\ \textsc{TopE\_TypeAlias}$$

$$\frac{}{\textbf{type}\ X\ variantArgs = V_1\ tyList_1 | \ldots | V_n\ tyList_n\ program \longrightarrow program}\ \textsc{TopE\_Type}$$

$$\frac{}{\textbf{exception}\ Exc\ tyList\ program \longrightarrow program}\ \textsc{TopE\_Exception}$$

$\boxed{t \longrightarrow t'}$ Evaluation

$$\frac{}{V \longrightarrow \textbf{TConstr}\ V}\ \textsc{E\_Variant}$$

$$\frac{}{(\textbf{failure}\ exnval)\ t \longrightarrow \textbf{failure}\ exnval}\ \textsc{E\_App1Failure}$$

$$\frac{}{v\ (\textbf{failure}\ exnval) \longrightarrow \textbf{failure}\ exnval}\ \textsc{E\_App2Failure}$$

$$\frac{t_1 \longrightarrow t_1'}{t_1\ t \longrightarrow t_1'\ t}\ \textsc{E\_App1}$$

$$\frac{t_1 \longrightarrow t_1'}{v\ t_1 \longrightarrow v\ t_1'}\ \textsc{E\_App2}$$

$$\frac{}{(\textbf{TConstr}\ V\ v_1\ ..\ v_n)\ v_m \longrightarrow \textbf{TConstr}\ V\ v_1\ ..\ v_n\ v_m}\ \textsc{E\_AppVariant}$$

$$\frac{}{(\lambda(x : T) \rightarrow t_{12})\ v_2 \longrightarrow [x \mapsto v_2]t_{12}}\ \textsc{E\_AppAbs}$$

$$\frac{}{t[T] \longrightarrow t}\ \textsc{E\_TApp}$$

$$\frac{}{t[[eff]] \longrightarrow t}\ \textsc{E\_EApp}$$

$$\frac{}{\textbf{let}\ x = \textbf{failure}\ exnval\ \textbf{in}\ t_2 \longrightarrow \textbf{failure}\ exnval}\ \textsc{E\_LetFailure}$$

$$\frac{t_1 \longrightarrow t_1'}{\textbf{let}\ x = t_1\ \textbf{in}\ t_2 \longrightarrow \textbf{let}\ x = t_1'\ \textbf{in}\ t_2}\ \textsc{E\_Let1}$$

$$\frac{}{\textbf{let}\ x = v\ \textbf{in}\ t \longrightarrow [x \mapsto v]t}\ \textsc{E\_Let2}$$

$$\frac{}{\textbf{let rec}\ x : T = abs\ \textbf{in}\ t \longrightarrow [x \mapsto (\textbf{let rec}\ x : T = abs\ \textbf{in}\ abs)]t}\ \textsc{E\_LetRec}$$

$$\frac{}{\textbf{match failure}\ exnval\ \textbf{with}\ p_1 \rightarrow t_1 | \ldots | p_n \rightarrow t_n\ \textbf{end} \longrightarrow \textbf{failure}\ exnval}\ \textsc{E\_MatchFailure}$$

$$\frac{t \longrightarrow t'}{\textbf{match}\ t\ \textbf{with}\ p_1 \rightarrow t_1 | \ldots | p_n \rightarrow t_n\ \textbf{end} \longrightarrow \textbf{match}\ t'\ \textbf{with}\ p_1 \rightarrow t_1 | \ldots | p_n \rightarrow t_n\ \textbf{end}}\ \textsc{E\_Match}$$

$$\frac{v\ \textbf{matches}\ p_1 \rhd \{x_1 \leftarrow v_1, \ldots, x_n \leftarrow v_n\}}{\textbf{match}\ v\ \textbf{with}\ p_1 \rightarrow t_1 | \ldots | p_n \rightarrow t_n\ \textbf{end} \longrightarrow [x_1 \mapsto v_1, \ldots, x_n \mapsto v_n]t_1}\ \textsc{E\_MatchFound}$$

$$\frac{\textbf{not}\,(v\ \textbf{matches}\ p_1 \rhd \{x_1 \leftarrow v_1,\ ..,\ x_n \leftarrow v_n\})}{\textbf{match}\ v\ \textbf{with}\ p_1 \to t_1 | p_2 \to t_2 | ..\ | p_n \to t_n\ \textbf{end} \longrightarrow \textbf{match}\ v\ \textbf{with}\ p_2 \to t_2 | ..\ | p_n \to t_n\ \textbf{end}}\ \text{E\_MatchStep}$$

$$\frac{t \longrightarrow t'}{(t : [\mathit{eff}]\,T) \longrightarrow t'}\ \text{E\_Annot}$$

$$\frac{t \longrightarrow t'}{\textbf{fail}\,[T]\mathit{Exc}\ v_1\ ..\ v_n\ t\ t_1\ ..\ t_n \longrightarrow \textbf{fail}\,[T]\mathit{Exc}\ v_1\ ..\ v_n\ t'\ t_1\ ..\ t_n}\ \text{E\_FailureArgs}$$

$$\frac{}{\textbf{fail}\,[T]\mathit{Exc}\ v_1\ ..\ v_n \longrightarrow \textbf{failure}\,\mathit{Exc}\ v_1\ ..\ v_n}\ \text{E\_Failure}$$

$$\frac{t \longrightarrow t'}{\textbf{try}\ t\ \textbf{with}\ pe_1 \to t_1 | ..\ | pe_n \to t_n\ \textbf{end} \longrightarrow \textbf{try}\ t'\ \textbf{with}\ pe_1 \to t_1 | ..\ | pe_n \to t_n\ \textbf{end}}\ \text{E\_Try}$$

$$\frac{}{\textbf{try}\ v\ \textbf{with}\ pe_1 \to t_1 | ..\ | pe_n \to t_n\ \textbf{end} \longrightarrow v}\ \text{E\_TryNoFailure}$$

$$\frac{\textbf{not}\,(\mathit{exnval}\ \textbf{matches}\ pe_1 \rhd \{x_1 \leftarrow v_1,\ ..,\ x_n \leftarrow v_n\})}{\textbf{try failure}\ \mathit{exnval}\ \textbf{with}\ pe_1 \to t_1\ \textbf{end} \longrightarrow \textbf{failure}\ \mathit{exnval}}\ \text{E\_TryNotFound}$$

$$\frac{\mathit{exnval}\ \textbf{matches}\ pe_1 \rhd \{x_1 \leftarrow v_1,\ ..,\ x_n \leftarrow v_n\}}{\textbf{try failure}\ \mathit{exnval}\ \textbf{with}\ pe_1 \to t_1 | ..\ | pe_n \to t_n\ \textbf{end} \longrightarrow [x_1 \mapsto v_1,\ ..,\ x_n \mapsto v_n]t_1}\ \text{E\_TryFound}$$

$$\frac{\textbf{not}\,(\mathit{exnval}\ \textbf{matches}\ pe_1 \rhd \{x_1 \leftarrow v_1,\ ..,\ x_n \leftarrow v_n\})}{\textbf{try failure}\ \mathit{exnval}\ \textbf{with}\ pe_1 \to t_1 | pe_2 \to t_2 | ..\ | pe_n \to t_n\ \textbf{end} \longrightarrow \textbf{try failure}\ \mathit{exnval}\ \textbf{with}\ pe_2 \to t_2 | ..\ | pe_n \to t_n\ \textbf{end}}$$

$\boxed{\mathit{exnval}\ \textbf{matches}\ pe \rhd \{x_1 \leftarrow v_1,\ ..,\ x_n \leftarrow v_n\}}$ Exception pattern matching with substitution creation

$$\frac{}{\mathit{Exc}\ v_1\ ..\ v_n\ \textbf{matches}\ \mathit{Exc}\ x_1\ ..\ x_n \rhd \{x_1 \leftarrow v_1,\ ..,\ x_n \leftarrow v_n\}}\ \text{ExnMatches\_Matches}$$

$\boxed{v\ \textbf{matches}\ p \rhd \{\mathit{substs}\}}$ Pattern matching with substitution creation

$$\frac{}{v\ \textbf{matches}\ x \rhd \{x \leftarrow v\}}\ \text{Matches\_Any}$$

$$\frac{v_1\ \textbf{matches}\ p_1 \rhd \{\mathit{substs}_1\} \quad .. \quad v_n\ \textbf{matches}\ p_n \rhd \{\mathit{substs}_n\}}{\textbf{TConstr}\ V\ v_1\ ..\ v_n\ \textbf{matches}\ V\ p_1\ ..\ p_n \rhd \{\mathit{substs}_1 \cup .. \cup \mathit{substs}_n\}}\ \text{Matches\_Matches}$$

```
Definition rules:        86 good    0 bad
Definition rule clauses: 174 good    0 bad
```