$\mathit{termvar}$, x	term variable
$\mathit{variant}$, V	variant
typvar, X	type variable
exc, Exc	exception
effect, E	effect
n	
m	

$\mathit{program}$	$::=$			program
	|			nothing
	|	$\mathit{top}\ \mathit{program}$		a piece of the program
top	$::=$			toplevel construct
	|	$\textbf{let}\, x\ \mathit{params} = t$		let binding
	|	$\textbf{let rec}\, x\ \mathit{params} = \mathit{abs}$		recursive let binding
	|	$\textbf{type alias}\, X = T$		type alias
	|	$\textbf{type}\, X\ \mathit{variantArgs} = V_1\ \mathit{tyList}_1 \mid .. \mid V_n\ \mathit{tyList}_n$		variants
	|	$\textbf{exception}\, \mathit{Exc}\ \mathit{tyList}$		exception
$\mathit{variantArgs}$	$::=$			
	|	$(X_1 : K_1) .. (X_n : K_n)$		
tyList	$::=$			
	|	$T_1 .. T_n$		
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
	|	$\lambda \mathit{params} \to t$	S	
	|	$t\ t'$		application
	|	$t[T]$		type application
	|	$t[[\mathit{eff}]]$		effect application
	|	$\textbf{let}\, x\ \mathit{params} = t_1\ \textbf{in}\ t_2$		let binding
	|	$\textbf{let rec}\, x\ \mathit{params} = \mathit{abs}\ \textbf{in}\ t$		recursive let binding
	|	$\textbf{match}\, t\ \textbf{with}\, p_1 \to t_1 \mid .. \mid p_n \to t_n\ \textbf{end}$		pattern matching
	|	$t : \mathit{annot}$		type annotation
	|	$\textbf{fail}\, [T] \mathit{Exc}\ t_1 .. t_n$		fail
	|	$\textbf{try}\, t\ \textbf{with}\, \mathit{pe}_1 \to t_1 \mid .. \mid \mathit{pe}_n \to t_n\ \textbf{end}$		try
	|	$t; t'$	S	== let _ : Unit = t in t'
	|	(t)	S	
	|	$\textbf{failure}\, \mathit{exnval}$	M	
	|	$\textbf{TConstr}\, V\ v_1 .. v_n$	M	
v	$::=$			value
	|	$\textbf{TConstr}\, V\ v_1 .. v_n$	M	type constructors
	|	$\lambda(x : T) \to t$		abstraction
	|	$\lambda \mathit{valueParams} \to v$	S	
	|	$\textbf{let rec}\, x\ \mathit{params} = \mathit{valAbs}_1\ \textbf{in}\ \mathit{valAbs}_2$		recursive let binding
valAbs	$::=$			
	|	$\lambda(x : T) \to t$		abstraction

abs	$::=$			lambda abstractions
	$\vert$	$\lambda(x : T) \to t$		abstraction
	$\vert$	$\lambda(X : K) \to abs$		type abstraction
	$\vert$	$\lambda(E : \varphi) \to abs$		effect abstraction
	$\vert$	$\textbf{let}\, x\ params = t\, \textbf{in}\, abs$		let binding
	$\vert$	$\textbf{let rec}\, x\ params = abs_1\, \textbf{in}\, abs_2$		recursive let binding
	$\vert$	$abs : annot$		type annotation
eff	$::=$			effect
	$\vert$	$effelm_1, .. , effelm_n$		
	$\vert$	$eff_1 \cup eff_2 \cup .. \cup eff_n$	M	
	$\vert$	$eff_1 \backslash [exn]$	M	
	$\vert$	(eff)	S	
$effelm$	$::=$			effects elements
	$\vert$	E		effect
	$\vert$	$\textbf{IO}$		IO effect
	$\vert$	$\textbf{Exn}\, [exn]$		exception
exn	$::=$			exceptions
	$\vert$	$Exc_1 \vert .. \vert Exc_n$		
K	$::=$			kinds
	$\vert$	$*$		star
	$\vert$	$K \to K'$		kind arrow
T	$::=$			type
	$\vert$	X		variable
	$\vert$	$\textbf{Unit}$		Unit type (contained in the module opened by defaul
	$\vert$	$T \to T'$	S	== T -[]-¡¿ T’
	$\vert$	$T - [eff] - > T'$		function
	$\vert$	$\lambda(X : K), T$		operator abstraction
	$\vert$	$\forall\, (X : K), T$		forall
	$\vert$	$\forall\, (E : \varphi), T$		effect forall
	$\vert$	$\forall\, tyParams, T$	S	
	$\vert$	$T\ T'$		operator application
	$\vert$	(T)	S	
	$\vert$	$[X \mapsto T]\, T'$	M	
	$\vert$	$[E \mapsto eff]\, T$	M	
	$\vert$	$T_1 \to .. \to T_n \to T$	M	
	$\vert$	$T\ T_1 .. T_n$	M	
p	$::=$			pattern
	$\vert$	$V\ p_1 .. p_n$		variant
	$\vert$	x		wildcard variable
pe	$::=$			try pattern

	$\|$	$\mathit{Exc}\ x_1\ ..\ x_n$		Exception pattern
lambda	$::=$			lambda parameters
	$\|$	$(x : T)$		value
	$\|$	$\mathit{tyLambda}$		type
$\mathit{valueLambda}$	$::=$			
	$\|$	$(x : T)$		value
$\mathit{tyLambda}$	$::=$			
	$\|$	$(X : K)$		type
	$\|$	X	S	
	$\|$	$(E : \varphi)$		effect
$\mathit{valueParams}$	$::=$			
	$\|$	$\mathit{valueLambda}_1\ ..\ \mathit{valueLambda}_n$	S	
$\mathit{tyParams}$	$::=$			
	$\|$	$\mathit{tyLambda}_1\ ..\ \mathit{tyLambda}_n$	S	
exnval	$::=$			runtime value of exceptions
	$\|$	$\mathit{Exc}\ v_1\ ..\ v_n$		
annot	$::=$			
	$\|$	T	S	
	$\|$	$[\mathit{eff}]\,T$	S	
params	$::=$			
	$\|$	$\mathit{lambda}_1\ ..\ \mathit{lambda}_n$	S	
	$\|$	$\mathit{lambda}_1\ ..\ \mathit{lambda}_n : \mathit{annot}$	S	
Γ	$::=$			type environment
	$\|$	$\emptyset$		empty
	$\|$	$\Gamma, x_1 : T_1, .., x_n : T_n$		vars
	$\|$	$\Gamma, V : T$	S	type constructors (contained in the above valu
	$\|$	$\Gamma, X_1 : K_1, .., X_n : K_n$		tvars
	$\|$	$\Gamma, X : \{\, V_1\ \mathit{tyList}_1\ ..\ V_n\ \mathit{tyList}_n \}$		variants
	$\|$	$\Gamma, \mathit{Exc}\ \mathit{tyList}$		exceptions
	$\|$	Γ, E		effects
	$\|$	$\Gamma_1 \cup .. \cup \Gamma_n$	M	
Δ	$::=$			runtime environment
	$\|$	$\emptyset$		empty
	$\|$	$\Delta, \{x_1 \leftarrow v_1\ ..\ x_n \leftarrow v_n\}$		vars
	$\|$	$\Delta, \{\, V_1 \leftarrow v_1\ ..\ V_n \leftarrow v_n\}$		variant vars
	$\|$	$\Delta_1 \cup .. \cup \Delta_n$	M	

$$
\begin{array}{lcl}
\mathit{terminals} & ::= & \\
 & | & \lambda \\
 & | & \backslash \\
 & | & \longrightarrow \\
 & | & \rightarrow \\
 & | & \vdash \\
 & | & \mapsto \\
 & | & \in \\
 & | & \equiv \\
 & | & \varphi \\
 & | & \forall \\
 & | & \cup \\
 & | & \triangleright \\
 & | & \leftarrow \\
 & | & \ \& \\
 & | & \emptyset \\
 & & \\
\mathit{formula} & ::= & \\
 & | & \mathit{judgement} \\
 & | & \mathit{formula}_1 \quad .. \quad \mathit{formula}_n \\
 & | & \textbf{not}\,(\mathit{formula}) \\
 & | & T \equiv T' \\
 & | & x \,:\, T \;\in\; \Gamma \\
 & | & X \,:\, K \;\in\; \Gamma \\
 & | & V \,:\, T \;\in\; \Gamma \\
 & | & T : \{\mathit{Variant}\} \;\in\; \Gamma \\
 & | & E \;\in\; \Gamma \\
 & | & \mathit{Exc}\ \mathit{tyList} \;\in\; \Gamma \\
 & | & \textbf{set}\,(\mathit{eff}) = \textbf{set}\,(\mathit{eff}') \\
 & | & \textbf{set}\,(\mathit{exn}) = \textbf{set}\,(\mathit{exn}') \\
 & | & V \;\in\; \mathit{Variant} \triangleright T_1 \mathrel{..} T_n \\
 & | & \mathit{RetVar} = \mathit{RetVar}' \\
 & | & \Gamma = \Gamma' \\
 & | & \{x \leftarrow v\} \;\in\; \Delta \\
 & | & \{V \leftarrow v\} \;\in\; \Delta \\
 & | & (\mathit{formula})\ \textbf{after applications} \\
 & & \\
\mathit{patterns} & ::= & \\
 & | & p_1 \mathrel{..} p_n \\
 & & \\
\mathit{VArgs} & ::= & \\
 & | & T_1 \mathrel{..} T_n \\
 & & \\
\mathit{Variant} & ::= & \\
 & | & V_1\ \mathit{VArgs}_1 \mathrel{..} V_n\ \mathit{VArgs}_n \\
 & | & \mathit{Variant} \backslash V \qquad\qquad\qquad\qquad \mathsf{M}
\end{array}
$$

RetVar	$::=$		
	$\mid$	$X\ X_1\,..\,X_n$	
JProgram	$::=$		
	$\mid$	$\Gamma \vdash \mathit{program} \triangleright \Gamma'$	Program typing
JTopType	$::=$		
	$\mid$	$\Gamma \vdash \mathit{top} \triangleright \Gamma'$	Toplevel typing
JTypeDecl	$::=$		
	$\mid$	$\mathit{RetVar}\ \&\ \Gamma \vdash\ V\ \mathit{tyList} \triangleright \Gamma'$	Type declaration
Jtype	$::=$		
	$\mid$	$\Gamma \vdash t : [\mathit{eff}]\, T$	Typing
Jkind	$::=$		
	$\mid$	$\Gamma \vdash\ T : K$	Kinding
JEff	$::=$		
	$\mid$	$\Gamma \vdash \mathit{eff}$	Effects typing
JEffElm	$::=$		
	$\mid$	$\Gamma \vdash \mathit{effelm}$	Effects elements typing
JPatternsTyping	$::=$		
	$\mid$	$\mathit{Variant}\ \&\ \Gamma \vdash \mathit{patterns} : T \triangleright \Gamma_1\,..\,\Gamma_n$	Patterns matching typing
JPatternTyping	$::=$		
	$\mid$	$\mathit{Variant}\ \&\ \Gamma \vdash p : T \triangleright \Gamma'$	Pattern matching typing
JExnPatternTyping	$::=$		
	$\mid$	$\Gamma \vdash \mathit{pe} \triangleright \mathit{Exc}\ \&\ \Gamma'$	Exception pattern matching typing
Jequiv	$::=$		
	$\mid$	$T \equiv T'$	Type equivalence
JEffEquiv	$::=$		
	$\mid$	$\mathit{eff} \equiv \mathit{eff}'$	Effects equivalence
JEffElmEquiv	$::=$		
	$\mid$	$\mathit{effelm} \equiv \mathit{effelm}'$	Effect element equivalence
JTopOp	$::=$		
	$\mid$	$\Delta \vdash \mathit{program} \longrightarrow \Delta' \vdash \mathit{program}'$	Toplevel evaluation
JVarCreationOp	$::=$		
	$\mid$	$V\ \mathit{tyList} \triangleright v$	Variants creation

Jop	::=		
	|	$\Delta \vdash t \longrightarrow \Delta' \vdash t'$	Evaluation
JExnMatches	::=		
	|	$\mathit{exnval}\,\mathbf{matches}\,\mathit{pe} \triangleright \Delta$	Exception pattern matching with substitution creation
JMatches	::=		
	|	$v\,\mathbf{matches}\,p \triangleright \Delta$	Pattern matching with substitution creation
judgement	::=		
	|	*JProgram*	
	|	*JTopType*	
	|	*JTypeDecl*	
	|	*Jtype*	
	|	*Jkind*	
	|	*JEff*	
	|	*JEffElm*	
	|	*JPatternsTyping*	
	|	*JPatternTyping*	
	|	*JExnPatternTyping*	
	|	*Jequiv*	
	|	*JEffEquiv*	
	|	*JEffElmEquiv*	
	|	*JTopOp*	
	|	*JVarCreationOp*	
	|	*Jop*	
	|	*JExnMatches*	
	|	*JMatches*	
user_syntax	::=		
	|	*termvar*	
	|	*variant*	
	|	*typvar*	
	|	*exc*	
	|	*effect*	
	|	*n*	
	|	*m*	
	|	*program*	
	|	*top*	
	|	*variantArgs*	
	|	*tyList*	
	|	*t*	
	|	*v*	
	|	*valAbs*	
	|	*abs*	
	|	*eff*	
	|	*effelm*	

$|\quad \mathit{exn}$
$|\quad K$
$|\quad T$
$|\quad p$
$|\quad \mathit{pe}$
$|\quad \mathit{lambda}$
$|\quad \mathit{valueLambda}$
$|\quad \mathit{tyLambda}$
$|\quad \mathit{valueParams}$
$|\quad \mathit{tyParams}$
$|\quad \mathit{exnval}$
$|\quad \mathit{annot}$
$|\quad \mathit{params}$
$|\quad \Gamma$
$|\quad \Delta$
$|\quad \mathit{terminals}$
$|\quad \mathit{formula}$
$|\quad \mathit{patterns}$
$|\quad \mathit{VArgs}$
$|\quad \mathit{Variant}$
$|\quad \mathit{RetVar}$

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

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

$$\frac{\begin{array}{l}\Gamma \vdash \mathit{top} \rhd \Gamma' \\ \Gamma' \vdash \mathit{program} \rhd \Gamma''\end{array}}{\Gamma \vdash \mathit{top}\ \mathit{program} \rhd \Gamma''} \quad \textsc{Prog\_Program}$$

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

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

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

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

$$\frac{\begin{array}{l}\mathit{RetVar} = X\ X_1 \,..\, X_n \\ \Gamma' = \Gamma, X : K, X_1 : K_1, \,..\,, X_n : K_n \\ \mathit{RetVar}\ \&\ \Gamma' \vdash V_1\ \mathit{tyList}_1 \rhd \Gamma_1 \quad .. \quad \mathit{RetVar}\ \&\ \Gamma' \vdash V_n\ \mathit{tyList}_n \rhd \Gamma_n\end{array}}{\Gamma \vdash \textbf{type}\, X\ (X_1 : K_1)(X_n : K_n) = V_1\ \mathit{tyList}_1 | \,..\, | V_n\ \mathit{tyList}_n \rhd \Gamma \cup \Gamma_1 \cup \,..\, \cup \Gamma_n, X : \{V_1\ \mathit{tyList}_1 \,..\, V_n\ \mathit{tyList}_n\}} \quad \text{T}$$

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

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

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

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

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

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

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

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

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

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

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

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

$$\frac{\begin{array}{l}\Gamma \vdash t : [\mathit{eff}] X \\ X \equiv X' \\ \Gamma \vdash X' : *\end{array}}{\Gamma \vdash t : [\mathit{eff}] X'}\quad \text{T\_E}\textsc{q}$$

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

$$\frac{\begin{array}{l}\Gamma, x : T_1 \vdash \mathit{abs} : [\,]\, T_1 \\ \Gamma, x : T_1 \vdash t_2 : [\mathit{eff}]\, T_2\end{array}}{\Gamma \vdash \mathbf{let\, rec}\, x\ = \mathit{abs} : T_1\, \mathbf{in}\, t_2 : [\mathit{eff}]\, T_2}\quad \text{T\_L}\textsc{et}\text{R}\textsc{ec}$$

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

$$\frac{\begin{array}{l}\Gamma \vdash t : [\mathit{eff}]\, T \\ \Gamma \vdash T : * \\ \Gamma \vdash \mathit{eff}\end{array}}{\Gamma \vdash (t : [\mathit{eff}]\, T) : [\mathit{eff}]\, T}\quad \text{T\_A}\textsc{nnot}$$

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

$$\frac{\begin{array}{c}T_1 \equiv T_2\\ T_2 \equiv T_3\end{array}}{T_1 \equiv T_3}\quad \textsc{Q\_Trans}$$

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

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

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

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

$$\frac{\begin{array}{c}T_{11} \equiv T_{21}\\ T_{12} \equiv T_{22}\end{array}}{T_{11}\ T_{12} \equiv T_{21}\ T_{22}}\quad \textsc{Q\_App}$$

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

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

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

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

$$\frac{\begin{array}{c}\mathit{effelm}_1 \equiv \mathit{effelm}_2\\ \mathit{effelm}_2 \equiv \mathit{effelm}_3\end{array}}{\mathit{effelm}_1 \equiv \mathit{effelm}_3}\quad \textsc{EffElmEq\_Trans}$$

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

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

$$\frac{\Delta \vdash t \longrightarrow \Delta \vdash t'}{\Delta \vdash \textbf{let}\, x \;= t\ \textit{program} \longrightarrow \Delta \vdash \textbf{let}\, x \;= t'\ \textit{program}} \quad \textsc{TopE\_Let1}$$

$$\frac{}{\Delta \vdash \textbf{let}\, x \;= v\ \textit{program} \longrightarrow \Delta, \{x \leftarrow v\} \vdash \textit{program}} \quad \textsc{TopE\_Let2}$$

$$\frac{\Delta \vdash \textit{abs} \longrightarrow \Delta \vdash \textit{abs}'}{\Delta \vdash \textbf{let}\,\textbf{rec}\, x \;= \textit{abs}\ \textit{program} \longrightarrow \Delta \vdash \textbf{let}\,\textbf{rec}\, x \;= \textit{abs}'\ \textit{program}} \quad \textsc{TopE\_LetRec1}$$

$$\frac{}{\Delta \vdash \textbf{let}\,\textbf{rec}\, x \;= \textit{valAbs}\ \textit{program} \longrightarrow \Delta, \{x \leftarrow \textbf{let}\,\textbf{rec}\, x \;= \textit{valAbs}\ \textbf{in}\ \textit{valAbs}\} \vdash \textit{program}} \quad \textsc{TopE\_LetRec2}$$

$$\frac{}{\Delta \vdash \textbf{type}\,\textbf{alias}\ X = T\ \textit{program} \longrightarrow \Delta \vdash \textit{program}} \quad \textsc{TopE\_TypeAlias}$$

$$\frac{V_1\ \textit{tyList}_1 \rhd v_1 \quad .. \quad V_n\ \textit{tyList}_n \rhd v_n}{\Delta \vdash \textbf{type}\ X\ \textit{variantArgs} = V_1\ \textit{tyList}_1 \,|\, .. \,|\, V_n\ \textit{tyList}_n\ \textit{program} \longrightarrow \Delta, \{V_1 \leftarrow v_1 .. V_n \leftarrow v_n\} \vdash \textit{program}} \quad \textsc{TopE\_T}$$

$$\frac{}{\Delta \vdash \textbf{exception}\ \textit{Exc}\ \textit{tyList}\ \textit{program} \longrightarrow \Delta \vdash \textit{program}} \quad \textsc{TopE\_Exception}$$

$\boxed{V\ \textit{tyList} \rhd v}$ Variants creation

$$\frac{(\{x_1 \leftarrow v_1\} \in \Delta)\ \textbf{after applications} \quad .. \quad (\{x_n \leftarrow v_n\} \in \Delta)\ \textbf{after applications}}{V\ T_1 .. T_n \rhd \lambda(x_1 : T_1) .. (x_n : T_n) \to \textbf{TConstr}\ V\ v_1 .. v_n} \quad \textsc{VarCreation\_Create}$$

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

$$\frac{\{x \leftarrow v\} \in \Delta}{\Delta \vdash x \longrightarrow \Delta \vdash v} \quad \textsc{E\_Var}$$

$$\frac{\{V \leftarrow v\} \in \Delta}{\Delta \vdash V \longrightarrow \Delta \vdash v} \quad \textsc{E\_Variant}$$

$$\frac{}{\Delta \vdash (\textbf{failure}\ \textit{exnval})\ t \longrightarrow \Delta \vdash \textbf{failure}\ \textit{exnval}} \quad \textsc{E\_App1Failure}$$

$$\frac{}{\Delta \vdash v\ (\textbf{failure}\ \textit{exnval}) \longrightarrow \Delta \vdash \textbf{failure}\ \textit{exnval}} \quad \textsc{E\_App2Failure}$$

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

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

$$\frac{}{\Delta \vdash (\lambda(x : T) \to t_{12})\ v_2 \longrightarrow \Delta, \{x \leftarrow v_2\} \vdash t_{12}} \quad \textsc{E\_AppAbs}$$

$$\frac{}{\Delta \vdash \lambda(X : K) \to t \longrightarrow \Delta \vdash t} \quad \textsc{E\_TAbs}$$

$$\frac{}{\Delta \vdash \lambda(E : \varphi) \to t \longrightarrow \Delta \vdash t} \quad \textsc{E\_EAbs}$$

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

$$\frac{}{\Delta \vdash t[[\textit{eff}]] \longrightarrow \Delta \vdash t} \quad \textsc{E\_EApp}$$

$$\frac{}{\Delta \vdash \textbf{let}\, x \;= \textbf{failure}\ \textit{exnval}\ \textbf{in}\ t_2 \longrightarrow \Delta \vdash \textbf{failure}\ \textit{exnval}} \quad \textsc{E\_LetFailure}$$

$$\frac{\Delta \vdash t_1 \longrightarrow \Delta \vdash t_1'}{\Delta \vdash \textbf{let}\, x \;= t_1\ \textbf{in}\ t_2 \longrightarrow \Delta \vdash \textbf{let}\, x \;= t_1'\ \textbf{in}\ t_2} \quad \textsc{E\_Let1}$$

$$\frac{}{\Delta \vdash \textbf{let}\, x\ = v\, \textbf{in}\, t \longrightarrow \Delta, \{x \leftarrow v\} \vdash t}\quad \textsc{E\_Let2}$$

$$\frac{\Delta \vdash \mathit{abs} \longrightarrow \Delta \vdash \mathit{abs}'}{\Delta \vdash \textbf{let}\,\textbf{rec}\, x\ = \mathit{abs}\, \textbf{in}\, t \longrightarrow \Delta \vdash \textbf{let}\,\textbf{rec}\, x\ = \mathit{abs}'\, \textbf{in}\, t}\quad \textsc{E\_LetRec1}$$

$$\frac{}{\Delta \vdash \textbf{let}\,\textbf{rec}\, x\ = \mathit{valAbs}\, \textbf{in}\, t \longrightarrow \Delta, \{x \leftarrow \textbf{let}\,\textbf{rec}\, x\ = \mathit{valAbs}\, \textbf{in}\, \mathit{valAbs}\} \vdash t}\quad \textsc{E\_LetRec2}$$

$$\frac{}{\Delta \vdash \textbf{match}\,\textbf{failure}\, \mathit{exnval}\, \textbf{with}\, p_1 \to t_1 | \mathinner{..} | p_n \to t_n\, \textbf{end} \longrightarrow \Delta \vdash \textbf{failure}\, \mathit{exnval}}\quad \textsc{E\_MatchFailure}$$

$$\frac{\Delta \vdash t \longrightarrow \Delta \vdash t'}{\Delta \vdash \textbf{match}\, t\, \textbf{with}\, p_1 \to t_1 | \mathinner{..} | p_n \to t_n\, \textbf{end} \longrightarrow \Delta \vdash \textbf{match}\, t'\, \textbf{with}\, p_1 \to t_1 | \mathinner{..} | p_n \to t_n\, \textbf{end}}\quad \textsc{E\_Match}$$

$$\frac{v\, \textbf{matches}\, p_1 \rhd \Delta'}{\Delta \vdash \textbf{match}\, v\, \textbf{with}\, p_1 \to t_1 | \mathinner{..} | p_n \to t_n\, \textbf{end} \longrightarrow \Delta \cup \Delta' \vdash t_1}\quad \textsc{E\_MatchFound}$$

$$\frac{\textbf{not}\, (v\, \textbf{matches}\, p_1 \rhd \Delta')}{\Delta \vdash \textbf{match}\, v\, \textbf{with}\, p_1 \to t_1 | p_2 \to t_2 | \mathinner{..} | p_n \to t_n\, \textbf{end} \longrightarrow \Delta \vdash \textbf{match}\, v\, \textbf{with}\, p_2 \to t_2 | \mathinner{..} | p_n \to t_n\, \textbf{end}}\quad \textsc{E\_MatchS}$$

$$\frac{\Delta \vdash t \longrightarrow \Delta \vdash t'}{\Delta \vdash (t : [\mathit{eff}]\, T) \longrightarrow \Delta \vdash t'}\quad \textsc{E\_Annot}$$

$$\frac{\Delta \vdash t \longrightarrow \Delta \vdash t'}{\Delta \vdash \textbf{fail}\, [T] \mathit{Exc}\, v_1 \mathinner{..} v_n\, t\, t_1 \mathinner{..} t_n \longrightarrow \Delta \vdash \textbf{fail}\, [T] \mathit{Exc}\, v_1 \mathinner{..} v_n\, t'\, t_1 \mathinner{..} t_n}\quad \textsc{E\_FailureArgs}$$

$$\frac{}{\Delta \vdash \textbf{fail}\, [T] \mathit{Exc}\, v_1 \mathinner{..} v_n \longrightarrow \Delta \vdash \textbf{failure}\, \mathit{Exc}\, v_1 \mathinner{..} v_n}\quad \textsc{E\_Failure}$$

$$\frac{\Delta \vdash t \longrightarrow \Delta \vdash t'}{\Delta \vdash \textbf{try}\, t\, \textbf{with}\, pe_1 \to t_1 | \mathinner{..} | pe_n \to t_n\, \textbf{end} \longrightarrow \Delta \vdash \textbf{try}\, t'\, \textbf{with}\, pe_1 \to t_1 | \mathinner{..} | pe_n \to t_n\, \textbf{end}}\quad \textsc{E\_Try}$$

$$\frac{}{\Delta \vdash \textbf{try}\, v\, \textbf{with}\, pe_1 \to t_1 | \mathinner{..} | pe_n \to t_n\, \textbf{end} \longrightarrow \Delta \vdash v}\quad \textsc{E\_TryNoFailure}$$

$$\frac{\textbf{not}\, (\mathit{exnval}\, \textbf{matches}\, pe_1 \rhd \Delta')}{\Delta \vdash \textbf{try}\,\textbf{failure}\, \mathit{exnval}\, \textbf{with}\, pe_1 \to t_1\, \textbf{end} \longrightarrow \Delta \vdash \textbf{failure}\, \mathit{exnval}}\quad \textsc{E\_TryNotFound}$$

$$\frac{\mathit{exnval}\, \textbf{matches}\, pe_1 \rhd \Delta'}{\Delta \vdash \textbf{try}\,\textbf{failure}\, \mathit{exnval}\, \textbf{with}\, pe_1 \to t_1 | \mathinner{..} | pe_n \to t_n\, \textbf{end} \longrightarrow \Delta \cup \Delta' \vdash t_1}\quad \textsc{E\_TryFound}$$

$$\frac{\textbf{not}\, (\mathit{exnval}\, \textbf{matches}\, pe_1 \rhd \Delta')}{\Delta \vdash \textbf{try}\,\textbf{failure}\, \mathit{exnval}\, \textbf{with}\, pe_1 \to t_1 | pe_2 \to t_2 | \mathinner{..} | pe_n \to t_n\, \textbf{end} \longrightarrow \Delta \vdash \textbf{try}\,\textbf{failure}\, \mathit{exnval}\, \textbf{with}\, pe_2 \to t_2 | \mathinner{..} | pe_n}$$

$\boxed{\mathit{exnval}\, \textbf{matches}\, pe \rhd \Delta}$ Exception pattern matching with substitution creation

$$\frac{}{\mathit{Exc}\, v_1 \mathinner{..} v_n\, \textbf{matches}\, \mathit{Exc}\, x_1 \mathinner{..} x_n \rhd \emptyset, \{x_1 \leftarrow v_1 \mathinner{..} x_n \leftarrow v_n\}}\quad \textsc{ExnMatches\_Matches}$$

$\boxed{v\, \textbf{matches}\, p \rhd \Delta}$ Pattern matching with substitution creation

$$\frac{}{v\, \textbf{matches}\, x \rhd \emptyset, \{x \leftarrow v\}}\quad \textsc{Matches\_Any}$$

$$\frac{v_1\, \textbf{matches}\, p_1 \rhd \Delta_1 \quad \mathinner{..} \quad v_n\, \textbf{matches}\, p_n \rhd \Delta_n}{\textbf{TConstr}\, V\, v_1 \mathinner{..} v_n\, \textbf{matches}\, V\, p_1 \mathinner{..} p_n \rhd \Delta_1 \cup \mathinner{..} \cup \Delta_n}\quad \textsc{Matches\_Matches}$$

```
Definition rules:        91 good    0 bad
Definition rule clauses: 185 good    0 bad
```