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

program	::=			program
	|			nothing
	|	*top program*		a piece of the program
top	::=			toplevel construct
	|	**let** x *params* $= t$		let binding
	|	**let rec** x *params* $=$ *abs*		recursive let binding
	|	**type alias** $X = T$		type alias
	|	**type** X *variantArgs* $= V_1$ *tyList*$_1 \,..\, \mid V_n$ *tyList*$_n$		variants
	|	**exception** *Exc tyList*		exception
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
	|	λ*params* $\to t$	S	
	|	$t\ t'$		application
	|	$t[T]$		type application
	|	**let** x *params* $= t_1$ **in** t_2		let binding
	|	**let rec** x *params* $=$ *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 : *annot*		type annotation
	|	**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	== let _ : Unit = t in t'
	|	(t)	S	
	|	**failure** *exnval*	M	
	|	**TConstr** $V\ v_1 \,..\, v_n$	M	
v	::=			value
	|	**TConstr** $V\ v_1 \,..\, v_n$	M	type constructors
	|	$\lambda(x : T) \to t$		abstraction
	|	λ*valueParams* $\to v$	S	
	|	**let rec** x *params* $=$ *valAbs*$_1$ **in** *valAbs*$_2$		recursive let binding
valAbs	::=			
	|	$\lambda(x : T) \to t$		abstraction
abs	::=			lambda abstractions
	|	$\lambda(x : T) \to t$		abstraction

	$\lambda(X : K) \to abs$		type abstraction
	let $x\ params = t$ **in** abs		let binding
	let rec $x\ params = abs_1$ **in** abs_2		recursive let binding
	$abs : annot$		type annotation
eff	::=		effect
	$effelm_1, .., effelm_n$		
	$eff_1 \cup eff_2 \cup .. \cup eff_n$	M	
	$eff_1 \backslash [exn]$	M	
	(eff)	S	
effelm	::=		effects elements
	X		effect
	IO		IO effect
	Exn $[exn]$		exception
exn	::=		exceptions
	$Exc_1 \| .. \| Exc_n$		
K	::=		kinds
	$*$		star
	φ		the effect kind
	$K \to K'$		kind arrow
T	::=		type
	X		variable
	$[eff]$		effects
	Unit		Unit type (contained in the module opened by defau
	$T \to T'$	S	== T -[]-¿ T'
	$T - [eff] - > T'$		function
	$\lambda(X : K), T$		operator abstraction
	$\forall\,(X : K), T$		forall
	$\forall\, tyParams, T$	S	
	$T\ T'$		operator application
	(T)	S	
	$[X \mapsto T]\, T'$	M	
	$T_1 \to .. \to T_n \to T$	M	
	$T\ T_1 .. T_n$	M	
p	::=		pattern
	$V\ p_1 .. p_n$		variant
	x		wildcard variable
pe	::=		try pattern
	$Exc\ x_1 .. x_n$		Exception pattern
lambda	::=		lambda parameters

	$(x : T)$		value
	tyLambda		type
valueLambda ::=			
	$(x : T)$		value
tyLambda ::=			
	$(X : K)$		type
	X	S	
valueParams ::=			
	$valueLambda_1 \,..\, valueLambda_n$	S	
tyParams ::=			
	$tyLambda_1 \,..\, tyLambda_n$	S	
exnval ::=			runtime value of exceptions
	$Exc\ v_1 \,..\, v_n$		
annot ::=			
	T	S	
	$[[eff]]\,T$	S	
params ::=			
	$lambda_1 \,..\, lambda_n$	S	
	$lambda_1 \,..\, lambda_n : annot$	S	
Γ ::=			type environment
	$\emptyset$		empty
	$\Gamma, x : T$		vars
	$\Gamma, x_1 : T_1, \,.., x_n : T_n$	S	vars
	$\Gamma, V : T$	S	type constructors (contained in the above valu
	$\Gamma, X : K$		tvars
	$\Gamma, X_1 : K_1, \,.., X_n : K_n$	S	tvars
	$\Gamma, X : \{\, V_1\ tyList_1 \,..\, V_n\ tyList_n\}$		variants
	$\Gamma, Exc\ tyList$		exceptions
	$\Gamma_1 \cup \,..\, \cup \Gamma_n$	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	
terminals ::=			
	λ		
	$\backslash$		

$$
\begin{array}{lll}
 & | & \longrightarrow \\
 & | & \rightarrow \\
 & | & \vdash \\
 & | & \mapsto \\
 & | & \in \\
 & | & \equiv \\
 & | & \varphi \\
 & | & \forall \\
 & | & \cup \\
 & | & \rhd \\
 & | & \leftarrow \\
 & | & \ \& \\
 & | & \emptyset \\
 & | & \notin
\end{array}
$$

$$
\begin{array}{lll}
\mathit{formula} & ::= & \\
 & | & \mathit{judgement} \\
 & | & \mathit{formula}_1 \ \ .. \ \ \mathit{formula}_n \\
 & | & \mathbf{not}\,(\mathit{formula}) \\
 & | & x : T \in \Gamma \\
 & | & X : K \in \Gamma \\
 & | & V : T \in \Gamma \\
 & | & T : \{\mathit{Variant}\} \in \Gamma \\
 & | & \mathit{Exc}\ \mathit{tyList} \in \Gamma \\
 & | & \mathbf{set}\,(\mathit{eff}) = \mathbf{set}\,(\mathit{eff}') \\
 & | & \mathbf{set}\,(\mathit{exn}) = \mathbf{set}\,(\mathit{exn}') \\
 & | & V \in \mathit{Variant} \rhd T_1 .. T_n \\
 & | & \mathit{RetVar} = \mathit{RetVar}' \\
 & | & \Gamma = \Gamma' \\
 & | & \{x \leftarrow v\} \in \Delta \\
 & | & \{V \leftarrow v\} \in \Delta \\
 & | & (\mathit{formula})\ \mathbf{after\ applications} \\
 & | & \mathit{effelm} \notin \mathit{eff}
\end{array}
$$

$$
\begin{array}{lll}
\mathit{patterns} & ::= & \\
 & | & p_1 .. p_n
\end{array}
$$

$$
\begin{array}{lll}
\mathit{VArgs} & ::= & \\
 & | & T_1 .. T_n
\end{array}
$$

$$
\begin{array}{lll}
\mathit{Variant} & ::= & \\
 & | & V_1\ \mathit{VArgs}_1 .. V_n\ \mathit{VArgs}_n \\
 & | & \mathit{Variant} \backslash V \qquad\qquad \mathsf{M}
\end{array}
$$

$$
\begin{array}{lll}
\mathit{RetVar} & ::= & \\
 & | & X\ X_1 .. X_n
\end{array}
$$

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

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

| *exn*
| *K*
| *T*
| *p*
| *pe*
| *lambda*
| *valueLambda*
| *tyLambda*
| *valueParams*
| *tyParams*
| *exnval*
| *annot*
| *params*
| Γ
| Δ
| *terminals*
| *formula*
| *patterns*
| *VArgs*
| *Variant*
| *RetVar*

$\boxed{\Gamma \vdash \mathbf{ok}}$ Typing environment validity

$$\frac{}{\emptyset \vdash \mathbf{ok}} \quad \textsc{GammaValid\_Empty}$$

$$\frac{\Gamma \vdash \mathbf{ok}}{\Gamma, x : T \vdash \mathbf{ok}} \quad \textsc{GammaValid\_Vars}$$

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

$$\frac{\begin{array}{l}\Gamma, X : K \vdash t : [[\mathit{eff}]]\,T \\ \mathbf{IO}\ \notin\ \mathit{eff}\end{array}}{\Gamma \vdash \lambda(X : K) \to t : [[\mathit{eff}]] \forall\,(X : K), T}\ \textsc{T\_TAbs}$$

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

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

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

$$\frac{\begin{array}{l}\Gamma, x : T_1 \vdash \mathit{abs} : [[\mathit{eff}_1]]\,T_1 \\ \Gamma, x : T_1 \vdash t_2 : [[\mathit{eff}_2]]\,T_2\end{array}}{\Gamma \vdash \mathbf{let\,rec}\ x\ = \mathit{abs} : [[\mathit{eff}_1]]\,T_1\ \mathbf{in}\ t_2 : [[\mathit{eff}_1 \cup \mathit{eff}_2]]\,T_2}\ \textsc{T\_LetRec}$$

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

$$\frac{\begin{array}{l}\Gamma \vdash t : [[\mathit{eff}]]\,T \\ \Gamma \vdash T : * \\ \Gamma \vdash \mathit{eff} : \varphi\end{array}}{\Gamma \vdash (t : [[\mathit{eff}]]\,T) : [[\mathit{eff}]]\,T}\ \textsc{T\_Annot}$$

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

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

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

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

$$\frac{\Gamma \vdash \mathit{eff} : \varphi}{\Gamma \vdash [\mathit{eff}] : \varphi}\ \text{K\_Eff}$$

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

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

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

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

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

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

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

$$\frac{X : \varphi \in \Gamma}{\Gamma \vdash X : \varphi}\ \text{EffElm\_Eff}$$

$$\frac{}{\Gamma \vdash \mathbf{IO} : \varphi}\ \text{EffElm\_IO}$$

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

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

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

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

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

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

$\boxed{\Gamma \vdash \mathit{pe} \rhd \mathit{Exc}\ \&\ \Gamma'}$ 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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

$$\frac{\Delta \vdash t \longrightarrow \Delta \vdash t'}{\Delta \vdash \mathbf{let}\,x\ = t\ \mathit{program} \longrightarrow \Delta \vdash \mathbf{let}\,x\ = t'\ \mathit{program}}\quad \textsc{TopE\_Let1}$$

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

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

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

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

$$\frac{V_1\ tyList_1 \rhd v_1 \quad .. \quad V_n\ tyList_n \rhd v_n}{\Delta \vdash \textbf{type}\ X\ variantArgs = V_1\ tyList_1 | .. | V_n\ tyList_n\ program \longrightarrow \Delta, \{V_1 \leftarrow v_1\ ..\ V_n \leftarrow v_n\} \vdash program}\ \textsc{TopE\_Ty}$$

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

```
Definition rules:        88 good    0 bad
Definition rule clauses: 180 good    0 bad
```