n, i, j, k	Index variables for meta-lists
num	Numeric literals
nat	Internal literal numbers
hex	Bit vector literal, specified by C-style hex number
bin	Bit vector literal, specified by C-style binary number
$string$	String literals
$regexp$	Regular expresions, as a string literal
x, y, z	Variables
ix	Variables

l	::=		Source locations
	|		
$x^l,\ y^l,\ z^l,\ name$	::=		Location-annotated names
	|	$x\ l$	
	|	$(ix)l$	Remove infix status
	|	$name\_t \to x^l$	M Extract x from a name_t
ix^l	::=		Location-annotated infix names
	|	$ix\ l$	
	|	$`x`l$	Add infix status
α	::=		Type variables
	|	$'x$	
α^l	::=		Location-annotated type variables
	|	$\alpha\ l$	
N	::=		numeric variables
	|	$''x$	
N^l	::=		Location-annotated numeric variables
	|	$N\ l$	
id	::=		Long identifers
	|	$x_1^l \ldots x_n^l.x^l\ l$	
tnv	::=		Union of type variables and Nexp type variables, without lo...
	|	α	
	|	N	
$tnvar^l$	::=		Union of type variables and Nexp type variables, with locati...
	|	α^l	
	|	N^l	
$tnvs$	::=		Type variable lists
	|	$tnv_1 \mathbin{..} tnv_n$	
$tnvars^l$	::=		Type variable lists
	|	$tnvar_1^l \mathbin{..} tnvar_n^l$	
$Nexp\_aux$	::=		Numerical expressions for specifying vector lengths and inde...
	|	N	
	|	num	
	|	$Nexp_1 * Nexp_2$	
	|	$Nexp_1 + Nexp_2$	
	|	$(Nexp)$	

Nexp	::=		Location-annotated vector lengths
	|	*Nexp_aux l*	
Nexp_constraint_aux	::=		Whether a vector is bounded or fixed size
	|	$Nexp = Nexp'$	
	|	$Nexp \geq Nexp'$	
Nexp_constraint	::=		Location-annotated Nexp range
	|	*Nexp_constraint_aux l*	
typ_aux	::=		Types
	|	_	Unspecified type
	|	α^l	Type variables
	|	$typ_1 \rightarrow typ_2$	Function types
	|	$typ_1 * .... * typ_n$	Tuple types
	|	*Nexp*	As a typ to permit applications over Nexps, other
	|	$id\ typ_1\ ..\ typ_n$	Type applications
	|	(typ)	
typ	::=		Location-annotated types
	|	*typ_aux l*	
lit_aux	::=		Literal constants
	|	**true**	
	|	**false**	
	|	*num*	
	|	*hex*	hex and bin are constant bit vectors, entered as 0
	|	*bin*	
	|	*string*	
	|	()	
	|	**bitzero**	bitzero and bitone are constant bits, if commonly
	|	**bitone**	
lit	::=		
	|	*lit_aux l*	Location-annotated literal constants
;?	::=		Optional semi-colons
	|		
	|	;	
pat_aux	::=		Patterns
	|	_	Wildcards
	|	$(pat\ \mathbf{as}\ x^l)$	Named patterns
	|	$(pat : typ)$	Typed patterns
	|	$id\ pat_1\ ..\ pat_n$	Single variable and constructor patterns
	|	$\langle|fpat_1; ...; fpat_n\ ;^?|\rangle$	Record patterns
	|	$[|pat_1; ..; pat_n\ ;^?|]$	Vector patterns

	$[	pat_1 .. pat_n	]$	Concatenated vector patterns
	$(pat_1, ..., pat_n)$	Tuple patterns
	$[pat_1; ..; pat_n ;^?]$	List patterns
	(pat)	
	$pat_1 :: pat_2$	Cons patterns
	$x^l + num$	constant addition patterns
	lit	Literal constant patterns

| pat | $::=$ | | Location-annotated patterns |
| | $pat\_aux\ l$ | |

| $fpat$ | $::=$ | | Field patterns |
| | $id = pat\ l$ | |

$	^?$	$::=$		Optional bars
	$	$	

exp_aux	$::=$		Expressions
	id		Identifiers
	N		Nexp var, has type num
	fun $psexp$		Curried functions
	function $|^?\ pexp_1|...|pexp_n$ **end**		Functions with pattern matching
	$exp_1\ exp_2$		Function applications
	$exp_1\ ix^l\ exp_2$		Infix applications
	$\langle|fexps|\rangle$		Records
	$\langle|exp\ \textbf{with}\ fexps|\rangle$		Functional update for records
	$exp.id$		Field projection for records
	$[|exp_1; ..; exp_n ;^?|]$		Vector instantiation
	$exp.(Nexp)$		Vector access
	$exp.(Nexp_1..Nexp_2)$		Subvector extraction
	match exp **with** $|^?\ pexp_1|...|pexp_n\ l$ **end**		Pattern matching expressions
	$(exp : typ)$		Type-annotated expressions
	let $letbind$ **in** exp		Let expressions
	$(exp_1, ..., exp_n)$		Tuples
	$[exp_1; ..; exp_n ;^?]$		Lists
	(exp)		
	begin exp **end**		Alternate syntax for (exp)
	if exp_1 **then** exp_2 **else** exp_3		Conditionals
	$exp_1 :: exp_2$		Cons expressions
	lit		Literal constants
	$\{exp_1|exp_2\}$		Set comprehensions
	$\{exp_1|\ \textbf{forall}\ qbind_1 .. qbind_n|exp_2\}$		Set comprehensions with explicit bin
	$\{exp_1; ..; exp_n ;^?\}$		Sets
	$q\ qbind_1 ... qbind_n.exp$		Logical quantifications
	$[exp_1|\ \textbf{forall}\ qbind_1 .. qbind_n|exp_2]$		List comprehensions (all binders mus
	do $id\ pat_1 <-exp_1; .. pat_n <-exp_n;$ **in** exp **end**		Do notation for monads

4

exp	$::=$		Location-annotated expressions
	$\|$	$exp\_aux\ l$	
q	$::=$		Quantifiers
	$\|$	**forall**	
	$\|$	**exists**	
$qbind$	$::=$		Bindings for quantifiers
	$\|$	x^l	
	$\|$	$(pat\ \textbf{IN}\ exp)$	Restricted quantifications over sets
	$\|$	$(pat\ \textbf{MEM}\ exp)$	Restricted quantifications over lists
$fexp$	$::=$		Field-expressions
	$\|$	$id = exp\ l$	
$fexps$	$::=$		Field-expression lists
	$\|$	$fexp_1;\ ...;fexp_n\ ;^?\ l$	
$pexp$	$::=$		Pattern matches
	$\|$	$pat \rightarrow exp\ l$	
$psexp$	$::=$		Multi-pattern matches
	$\|$	$pat_1\ ...\ pat_n \rightarrow exp\ l$	
$tannot^?$	$::=$		Optional type annotations
	$\|$		
	$\|$	$: typ$	
$funcl\_aux$	$::=$		Function clauses
	$\|$	$x^l\ pat_1\ ...\ pat_n\ tannot^? = exp$	
$letbind\_aux$	$::=$		Let bindings
	$\|$	$pat\ tannot^? = exp$	Value bindings
	$\|$	$funcl\_aux$	Function bindings
$letbind$	$::=$		Location-annotated let bindings
	$\|$	$letbind\_aux\ l$	
$funcl$	$::=$		Location-annotated function clauses
	$\|$	$funcl\_aux\ l$	
$name\_t$	$::=$		Name or name with type for inductively defined relation
	$\|$	x^l	
	$\|$	$(x^l : typ)$	
$name\_ts$	$::=$		Names with optional types for inductively defined relati
	$\|$	$name\_t_0\ ..\ name\_t_n$	

$rule\_aux$	::=		Inductively defined relat
	$\mid$	$x^l :$ **forall** $name\_t_1 \mathbin{..} name\_t_i.exp \Longrightarrow x_1^l\, exp_1 \mathbin{..} exp_n$	
$rule$	::=		Location-annotated indu
	$\mid$	$rule\_aux\, l$	
$witness^?$	::=		Optional witness type na
	$\mid$		
	$\mid$	**witness** $x^l;$	
$check^?$	::=		Option check name decl
	$\mid$		
	$\mid$	**check** $x^l;$	
$functions^?$	::=		Optional names and typ
	$\mid$		
	$\mid$	$x^l; typ$	
	$\mid$	$x^l : typ; functions^?$	
$indreln\_name\_aux$	::=		Name for inductively def
	$\mid$	$[x^l : typschm\ witness^?\ check^?\ functions^?]$	
$indreln\_name$	::=		Location-annotated nam
	$\mid$	$indreln\_name\_aux\, l$	
$typs$	::=		Type lists
	$\mid$	$typ_1 * ... * typ_n$	
$ctor\_def$	::=		Datatype definition clau
	$\mid$	x^l **of** $typs$	
	$\mid$	x^l	S Constant constructors
$texp$	::=		Type definition bodies
	$\mid$	typ	Type abbreviations
	$\mid$	$\langle| x_1^l : typ_1; ...; x_n^l : typ_n ;^? |\rangle$	Record types
	$\mid$	$|^?\ ctor\_def_1| ... |ctor\_def_n$	Variant types
$name^?$	::=		Optional name specificat
	$\mid$		
	$\mid$	$[name = regexp]$	
td	::=		Type definitions
	$\mid$	$x^l\, tnvars^l\, name^? = texp$	
	$\mid$	$x^l\, tnvars^l\, name^?$	Definitions of opaque
c	::=		Typeclass constraints
	$\mid$	$id\, tnvar^l$	

cs	::=		Typeclass and length constraint l
		$c_1, .., c_i \Rightarrow$	Must have > 0 constraints
		$Nexp\_constraint_1, .., Nexp\_constraint_i \Rightarrow$	Must have > 0 constraints
		$c_1, .., c_i; Nexp\_constraint_1, .., Nexp\_constraint_n \Rightarrow$	Must have > 0 of both form of

c_pre	::=		Type and instance scheme prefixe
		forall $tnvar_1^l .. tnvar_n^l.cs$	Must have > 0 type variables

typschm	::=		Type schemes
		$c\_pre\ typ$	

instschm	::=		Instance schemes
		$c\_pre(id\ typ)$	

target	::=		Backend target names
		hol	
		isabelle	
		ocaml	
		coq	
		tex	
		html	

τ	::=		Backend target name lists
		$\{target_1; ..; target_n\}$	

$\tau^?$	::=		Optional targets
		τ	

lemma_typ	::=		Types of Lemmata
		assert	
		lemma	
		theorem	

lemma_decl	::=		Lemmata and Tests
		$lemma\_typ\ \tau^?\ x^l : (exp)$	
		$lemma\_typ\ \tau^?(exp)$	

val_def	::=		Value definitions
		let $\tau^?$ *letbind*	Non-recursive value definitions
		let rec $\tau^?\ funcl_1$ **and** ... **and** $funcl_n$	Recursive function definitions
		let inline $\tau^?$ *letbind*	Function definitions to be inlin

val_spec	::=		Value type specifications
		val $x^l : typschm$	

def_aux ::=   Top-level d

        | **type** td_1 **and** ... **and** td_n Type def

        | val_def Value de

        | $lemma\_decl$ Lemmat

        | **rename** $\tau^?$ $id = x^l$ Rename

        | **module** $x^l =$ **struct** $defs$ **end**   Module

        | **module** $x^l = id$ Module

        | **open** id Opening

        | **indreln** $\tau^?$ $indreln\_name_1$ **and** ... **and** $indreln\_name_i$ $rule_1$ **and** ... **and** $rule_n$ Inductiv

        | val_spec Top-level

        | **class** $(x^l\ tnvar^l)$ **val** $x_1^l : typ_1\ l_1$ ... **val** $x_n^l : typ_n\ l_n$ **end**   Typeclas

        | **instance** $instschm\ val\_def_1\ l_1$ ... $val\_def_n\ l_n$ **end**   Typeclas

def ::=   Location-a

        | $def\_aux\ l$

$;;^?$ ::=   Optional d

        |

        | $;;$

$defs$ ::=   Definition s

        | $def_1\ ;;_1^?\ ..\ def_n\ ;;_n^?$

p ::=   Unique pat

        | $x_1 ... x_n.x$

        | **_list**

        | **_bool**

        | **_num**

        | **_set**

        | **_string**

        | **_unit**

        | **_bit**

        | **_vector**

σ ::=   Type varia

        | $\{tnv_1 \mapsto t_1\ ..\ tnv_n \mapsto t_n\}$

$t,\ u$ ::=   Internal ty

        | α

        | $t_1 \rightarrow t_2$

        | $t_1 * .... * t_n$

        | $p\ t\_args$

        | ne

        | $\sigma(t)$ M   Multiple

        | $\sigma(tnv)$ M   Single va

        | **curry** (t_multi, t) M   Curried,

ne	::=			internal numeric expressions
	|	N		
	|	nat		
	|	$ne_1 * ne_2$		
	|	$ne_1 + ne_2$		
	|	$(-ne)$		
	|	**normalize** (ne)	M	
	|	$ne_1 + ... + ne_n$	M	
	|	**bitlength** (bin)	M	
	|	**bitlength** (hex)	M	
	|	**length** $(pat_1 ... pat_n)$	M	
	|	**length** $(exp_1 ... exp_n)$	M	

t_args	::=			Lists of types
	|	$t_1 .. t_n$		
	|	$\sigma(t\_args)$	M	Multiple substitutions

t_multi	::=			Lists of types
	|	$(t_1 * .. * t_n)$		
	|	$\sigma(t\_multi)$	M	Multiple substitutions

nec	::=			Numeric expression constraints
	|	$ne\langle nec$		
	|	$ne = nec$		
	|	$ne <= nec$		
	|	ne		

| $names$ | ::= | | | Sets of names |
| | \| | $\{x_1, .., x_n\}$ | | |

| $\mathcal{C}$ | ::= | | | Typeclass constraint lists |
| | \| | $(p_1 \; tnv_1) .. (p_n \; tnv_n)$ | | |

env_tag	::=			Tags for the (non-constructor) value descriptie
	|	**method**		Bound to a method
	|	**val**		Specified with val
	|	**let**		Defined with let or indreln

v_desc	::=			Value descriptions
	|	$\langle$ **forall** $tnvs.t\_multi \to p, (x \textbf{ of } names)\rangle$		Constructors
	|	$\langle$ **forall** $tnvs.\mathcal{C} \Rightarrow t, env\_tag\rangle$		Values

| f_desc | ::= | | | |
| | \| | $\langle$ **forall** $tnvs.p \to t, (x \textbf{ of } names)\rangle$ | | Fields |

| xs | ::= | | | |
| | \| | $x_1 .. x_n$ | | |

$\Sigma^{\mathcal{C}}$	::=			Typeclass constraints
	|	$\{(p_1\ t_1),\ ..,(p_n\ t_n)\}$		
	|	$\Sigma^{\mathcal{C}}_1 \cup\ ..\ \cup \Sigma^{\mathcal{C}}_n$	M	
$\Sigma^{\mathcal{N}}$	::=			Nexp constraint lists
	|	$\{nec_1,\ ..,nec_n\}$		
	|	$\Sigma^{\mathcal{N}}_1 \cup\ ..\ \cup \Sigma^{\mathcal{N}}_n$	M	
E	::=			Environments
	|	$\langle E^{\mathrm{M}}, E^{\mathrm{P}}, E^{\mathrm{F}}, E^{\mathrm{X}}\rangle$		
	|	$E_1 \uplus E_2$	M	
	|	ϵ	M	
E^{X}	::=			Value environments
	|	$\{x_1 \mapsto v\_desc_1,\ ..,x_n \mapsto v\_desc_n\}$		
	|	$E^{\mathrm{X}}_1 \uplus\ ..\ \uplus E^{\mathrm{X}}_n$	M	
E^{F}	::=			Field environments
	|	$\{x_1 \mapsto f\_desc_1,\ ..,x_n \mapsto f\_desc_n\}$		
	|	$E^{\mathrm{F}}_1 \uplus\ ..\ \uplus E^{\mathrm{F}}_n$	M	
E^{M}	::=			Module environments
	|	$\{x_1 \mapsto E_1,\ ..,x_n \mapsto E_n\}$		
E^{P}	::=			Path environments
	|	$\{x_1 \mapsto p_1,\ ..,x_n \mapsto p_n\}$		
	|	$E^{\mathrm{P}}_1 \uplus\ ..\ \uplus E^{\mathrm{P}}_n$	M	
E^{L}	::=			Lexical bindings
	|	$\{x_1 \mapsto t_1,\ ..,x_n \mapsto t_n\}$		
	|	$\{x^l_1 \mapsto t_1,\ ..,x^l_n \mapsto t_n\}$		
	|	$E^{\mathrm{L}}_1 \uplus\ ..\ \uplus E^{\mathrm{L}}_n$	M	
tc_abbrev	::=			Type abbreviations
	|	$.t$		
	|			
tc_def	::=			Type and class constructor definitions
	|	$tnvs\ tc\_abbrev$		Type constructors
Δ	::=			Type constructor definitions
	|	$\{p_1 \mapsto tc\_def_1,\ ..,p_n \mapsto tc\_def_n\}$		
	|	$\Delta_1 \uplus \Delta_2$	M	
δ	::=			Typeclass definitions
	|	$\{p_1 \mapsto xs_1,\ ..,p_n \mapsto xs_n\}$		
	|	$\delta_1 \uplus \delta_2$	M	

inst	::=		A typeclass instance, t must not contain nested types
	|	$\mathcal{C} \Rightarrow (p\ t)$	

I	::=		Global instances
	|	$\{inst_1, .., inst_n\}$	
	|	$I_1 \cup I_2$	M

D	::=		Global type definition store
	|	$\langle \Delta, \delta, I \rangle$	
	|	$D_1 \uplus D_2$	M
	|	ϵ	M

terminals	::=		
	|	$\geq$	`>=`
	|	$\rightarrow$	`->`
	|	$\Longrightarrow$	`==>`
	|	$\langle\|$	`<|`
	|	$\|\rangle$	`|>`
	|	$\cap$	
	|	$\cup$	
	|	$\uplus$	
	|	$\notin$	
	|	$\subset$	
	|	$\neq$	
	|	$\emptyset$	
	|	$\langle$	
	|	$\rangle$	
	|	$\vdash$	
	|	,	
	|	$\mapsto$	
	|	$\triangleright$	
	|	$\rightsquigarrow$	
	|	$\Rightarrow$	
	|	$-$	
	|	ϵ	

formula	::=		
	|	*judgement*	
	|	$formula_1 \quad .. \quad formula_n$	
	|	$E^{\mathrm{M}}(x) \triangleright E$	Module lookup
	|	$E^{\mathrm{P}}(x) \triangleright p$	Path lookup
	|	$E^{\mathrm{F}}(x) \triangleright f\_desc$	Field lookup
	|	$E^{\mathrm{X}}(x) \triangleright v\_desc$	Value lookup
	|	$E^{\mathrm{L}}(x) \triangleright t$	Lexical binding lookup
	|	$\Delta(p) \triangleright tc\_def$	Type constructor lookup
	|	$\delta(p) \triangleright xs$	Type constructor lookup
	|	$\mathbf{dom}\,(E_1^{\mathrm{M}}) \cap \mathbf{dom}\,(E_2^{\mathrm{M}}) = \emptyset$	

$$| \quad \mathbf{dom}\,(E_1^{\mathrm{X}}) \cap \mathbf{dom}\,(E_2^{\mathrm{X}}) = \emptyset$$
$$| \quad \mathbf{dom}\,(E_1^{\mathrm{F}}) \cap \mathbf{dom}\,(E_2^{\mathrm{F}}) = \emptyset$$
$$| \quad \mathbf{dom}\,(E_1^{\mathrm{P}}) \cap \mathbf{dom}\,(E_2^{\mathrm{P}}) = \emptyset$$

$\|$	$\mathbf{disjoint\,doms}\,(E_1^{\mathrm{L}}, \ldots, E_n^{\mathrm{L}})$	Pairwise disjoint domains
$\|$	$\mathbf{disjoint\,doms}\,(E_1^{\mathrm{X}}, \ldots, E_n^{\mathrm{X}})$	Pairwise disjoint domains
$\|$	$\mathbf{compatible\,overlap}\,(x_1 \mapsto t_1, \ldots, x_n \mapsto t_n)$	$(x_i = x_j) \Longrightarrow (t_i = t_j)$
$\|$	$\mathbf{duplicates}\,(tnvs) = \emptyset$	
$\|$	$\mathbf{duplicates}\,(x_1, \ldots, x_n) = \emptyset$	
$\|$	$x \notin \mathbf{dom}\,(E^{\mathrm{L}})$	
$\|$	$x \notin \mathbf{dom}\,(E^{\mathrm{X}})$	
$\|$	$x \notin \mathbf{dom}\,(E^{\mathrm{F}})$	
$\|$	$p \notin \mathbf{dom}\,(\delta)$	
$\|$	$p \notin \mathbf{dom}\,(\Delta)$	
$\|$	$\mathbf{FV}\,(t) \subset tnvs$	Free type variables
$\|$	$\mathbf{FV}\,(t\_multi) \subset tnvs$	Free type variables
$\|$	$\mathbf{FV}\,(\mathcal{C}) \subset tnvs$	Free type variables
$\|$	$inst\ \mathbf{IN}\ I$	
$\|$	$(p\ t) \notin I$	
$\|$	$E_1^{\mathrm{L}} = E_2^{\mathrm{L}}$	
$\|$	$E_1^{\mathrm{X}} = E_2^{\mathrm{X}}$	
$\|$	$E_1^{\mathrm{F}} = E_2^{\mathrm{F}}$	
$\|$	$E_1 = E_2$	
$\|$	$\Delta_1 = \Delta_2$	
$\|$	$\delta_1 = \delta_2$	
$\|$	$I_1 = I_2$	
$\|$	$names_1 = names_2$	
$\|$	$t_1 = t_2$	
$\|$	$\sigma_1 = \sigma_2$	
$\|$	$p_1 = p_2$	
$\|$	$xs_1 = xs_2$	
$\|$	$tnvs_1 = tnvs_2$	

$convert\_tnvars$	$::=$	
$\|$	$tnvars^l \rightsquigarrow tnvs$	
$\|$	$tnvar^l \rightsquigarrow tnv$	

$look\_m$	$::=$	
$\|$	$E_1(x_1^l \mathinner{..} x_n^l) \rhd E_2$	Name path lookup

$look\_m\_id$	$::=$	
$\|$	$E_1(id) \rhd E_2$	Module identifier lookup

$look\_tc$	$::=$	
$\|$	$E(id) \rhd p$	Path identifier lookup

$check\_t$	$::=$	
$\|$	$\Delta \vdash t\,\mathbf{ok}$	Well-formed types

| | $\Delta, tnv \vdash t \, \mathbf{ok}$ | Well-formed type/Nexps ma |

| teq | ::= | |
| | $\Delta \vdash t_1 = t_2$ | Type equality |

$convert\_typ$	::=	
	$\Delta, E \vdash typ \rightsquigarrow t$	Convert source types to inte
	$\vdash Nexp \rightsquigarrow ne$	Convert and normalize num

| $convert\_typs$ | ::= | |
| | $\Delta, E \vdash typs \rightsquigarrow t\_multi$ | |

| $check\_lit$ | ::= | |
| | $\vdash lit : t$ | Typing literal constants |

| $inst\_field$ | ::= | |
| | $\Delta, E \vdash \mathbf{field} \, id : p \, t\_args \rightarrow t \rhd (x \, \mathbf{of} \, names)$ | Field typing (also returns c |

| $inst\_ctor$ | ::= | |
| | $\Delta, E \vdash \mathbf{ctor} \, id : t\_multi \rightarrow p \, t\_args \rhd (x \, \mathbf{of} \, names)$ | Data constructor typing (al |

| $inst\_val$ | ::= | |
| | $\Delta, E \vdash \mathbf{val} \, id : t \rhd \Sigma^{\mathcal{C}}$ | Typing top-level bindings, c |

| not_ctor | ::= | |
| | $E, E^{\mathrm{L}} \vdash x \, \mathbf{not \, ctor}$ | v is not bound to a data co |

| $not\_shadowed$ | ::= | |
| | $E^{\mathrm{L}} \vdash id \, \mathbf{not \, shadowed}$ | id is not lexically shadowed |

$check\_pat$	::=	
	$\Delta, E, E_1^{\mathrm{L}} \vdash pat : t \rhd E_2^{\mathrm{L}}$	Typing patterns, building t
	$\Delta, E, E_1^{\mathrm{L}} \vdash pat\_aux : t \rhd E_2^{\mathrm{L}}$	Typing patterns, building t

| id_field | ::= | |
| | $E \vdash id \, \mathbf{field}$ | Check that the identifier is |

| id_value | ::= | |
| | $E \vdash id \, \mathbf{value}$ | Check that the identifier is |

$check\_exp$	::=	
	$\Delta, E, E^{\mathrm{L}} \vdash exp : t \rhd \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}$	Typing expressions, collecti
	$\Delta, E, E^{\mathrm{L}} \vdash exp\_aux : t \rhd \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}$	Typing expressions, collecti
	$\Delta, E, E_1^{\mathrm{L}} \vdash qbind_1 \, .. \, qbind_n \rhd E_2^{\mathrm{L}}, \Sigma^{\mathcal{C}}$	Build the environment for q
	$\Delta, E, E_1^{\mathrm{L}} \vdash \mathbf{list} \, qbind_1 \, .. \, qbind_n \rhd E_2^{\mathrm{L}}, \Sigma^{\mathcal{C}}$	Build the environment for q
	$\Delta, E, E^{\mathrm{L}} \vdash funcl \rhd \{x \mapsto t\}, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}$	Build the environment for a
	$\Delta, E, E_1^{\mathrm{L}} \vdash letbind \rhd E_2^{\mathrm{L}}, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}$	Build the environment for a

check_rule	::=	
	|	$\Delta, E, E^{\mathrm{L}} \vdash rule \triangleright \{x \mapsto t\}, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}$	Build the environment for an indu
check_texp_tc	::=	
	|	$xs, \Delta_1, E \vdash \mathbf{tc}\ td \triangleright \Delta_2, E^{\mathrm{P}}$	Extract the type constructor inforn
check_texps_tc	::=	
	|	$xs, \Delta_1, E \vdash \mathbf{tc}\ td_1 \mathinner{..}\ td_i \triangleright \Delta_2, E^{\mathrm{P}}$	Extract the type constructor inforn
check_texp	::=	
	|	$\Delta, E \vdash tnvs\ p = texp \triangleright \langle E^{\mathrm{F}}, E^{\mathrm{X}} \rangle$	Check a type definition, with its pa
check_texps	::=	
	|	$xs, \Delta, E \vdash td_1 \mathinner{..}\ td_n \triangleright \langle E^{\mathrm{F}}, E^{\mathrm{X}} \rangle$	
convert_class	::=	
	|	$\delta, E \vdash id \rightsquigarrow p$	Lookup a type class
solve_class_constraint	::=	
	|	$I \vdash (p\ t)\ \mathbf{IN}\ \mathcal{C}$	Solve class constraint
solve_class_constraints	::=	
	|	$I \vdash \Sigma^{\mathcal{C}} \triangleright \mathcal{C}$	Solve class constraints
check_val_def	::=	
	|	$\Delta, I, E \vdash val\_def \triangleright E^{\mathrm{X}}$	Check a value definition
check_t_instance	::=	
	|	$\Delta, (\alpha_1, \mathinner{..}, \alpha_n) \vdash t\ \mathbf{instance}$	Check that t be a typeclass instanc
check_defs	::=	
	|	$\overline{z_j}^{\,j}, D_1, E_1 \vdash def \triangleright D_2, E_2$	Check a definition
	|	$\overline{z_j}^{\,j}, D_1, E_1 \vdash defs \triangleright D_2, E_2$	Check definitions, given module pa
judgement	::=	
	|	*convert_tnvars*	
	|	*look_m*	
	|	*look_m_id*	
	|	*look_tc*	
	|	*check_t*	
	|	*teq*	
	|	*convert_typ*	
	|	*convert_typs*	
	|	*check_lit*	
	|	*inst_field*	
	|	*inst_ctor*	
	|	*inst_val*	

| | *not_ctor*
| | *not_shadowed*
| | *check_pat*
| | *id_field*
| | *id_value*
| | *check_exp*
| | *check_rule*
| | *check_texp_tc*
| | *check_texps_tc*
| | *check_texp*
| | *check_texps*
| | *convert_class*
| | *solve_class_constraint*
| | *solve_class_constraints*
| | *check_val_def*
| | *check_t_instance*
| | *check_defs*

user_syntax   ::=
| | n
| | *num*
| | *nat*
| | *hex*
| | *bin*
| | *string*
| | *regexp*
| | x
| | *ix*
| | l
| | x^l
| | ix^l
| | α
| | α^l
| | N
| | N^l
| | *id*
| | *tnv*
| | $tnvar^l$
| | *tnvs*
| | $tnvars^l$
| | *Nexp_aux*
| | *Nexp*
| | *Nexp_constraint_aux*
| | *Nexp_constraint*
| | *typ_aux*
| | *typ*

| lit_aux
| lit
| $;^?$
| pat_aux
| pat
| $fpat$
| $|^?$
| exp_aux
| exp
| q
| $qbind$
| $fexp$
| $fexps$
| $pexp$
| $psexp$
| $tannot^?$
| $funcl\_aux$
| $letbind\_aux$
| $letbind$
| $funcl$
| $name\_t$
| $name\_ts$
| $rule\_aux$
| $rule$
| $witness^?$
| $check^?$
| $functions^?$
| $indreln\_name\_aux$
| $indreln\_name$
| $typs$
| $ctor\_def$
| $texp$
| $name^?$
| td
| c
| cs
| c_pre
| $typschm$
| $instschm$
| $target$
| τ
| $\tau^?$
| $lemma\_typ$
| $lemma\_decl$
| val_def
| val_spec

| def_aux
| def
| ;;?
| defs
| p
| σ
| t
| ne
| t_args
| t_multi
| nec
| names
| C
| env_tag
| v_desc
| f_desc
| xs
| Σ^C
| Σ^N
| E
| E^X
| E^F
| E^M
| E^P
| E^L
| tc_abbrev
| tc_def
| Δ
| δ
| inst
| I
| D
| terminals
| formula

$$\boxed{tnvars^l \rightsquigarrow tnvs}$$

$$\frac{tnvar_1^l \rightsquigarrow tnv_1 \quad .. \quad tnvar_n^l \rightsquigarrow tnv_n}{tnvar_1^l .. tnvar_n^l \rightsquigarrow tnv_1 .. tnv_n} \quad \text{CONVERT\_TNVARS\_NONE}$$

$$\boxed{tnvar^l \rightsquigarrow tnv}$$

$$\frac{}{\alpha\, l \rightsquigarrow \alpha} \quad \text{CONVERT\_TNVAR\_A}$$

$$\frac{}{N\, l \rightsquigarrow N} \quad \text{CONVERT\_TNVAR\_N}$$

$$\boxed{E_1(x_1^l .. x_n^l) \triangleright E_2} \quad \text{Name path lookup}$$

$$\frac{}{E(\,) \triangleright E} \quad \text{LOOK\_M\_NONE}$$

$$\frac{\begin{array}{c} E^{\mathrm{M}}(x) \triangleright E_1 \\ E_1(\overline{y_i^l}^{\,i}) \triangleright E_2 \end{array}}{\langle E^{\mathrm{M}}, E^{\mathrm{P}}, E^{\mathrm{F}}, E^{\mathrm{X}} \rangle (x \; l \; \overline{y_i^l}^{\,i}) \triangleright E_2} \quad \text{LOOK\_M\_SOME}$$

$\boxed{E_1(id) \triangleright E_2}$ Module identifier lookup

$$\frac{E_1(\overline{y_i^l}^{\,i} \; x \; l_1) \triangleright E_2}{E_1(\overline{y_i^l.}^{\,i} \; x \; l_1 \; l_2) \triangleright E_2} \quad \text{LOOK\_M\_ID\_ALL}$$

$\boxed{E(id) \triangleright p}$ Path identifier lookup

$$\frac{\begin{array}{c} E(\overline{y_i^l}^{\,i}) \triangleright \langle E^{\mathrm{M}}, E^{\mathrm{P}}, E^{\mathrm{F}}, E^{\mathrm{X}} \rangle \\ E^{\mathrm{P}}(x) \triangleright p \end{array}}{E(\overline{y_i^l.}^{\,i} \; x \; l_1 \; l_2) \triangleright p} \quad \text{LOOK\_TC\_ALL}$$

$\boxed{\Delta \vdash t \, \mathbf{ok}}$ Well-formed types

$$\frac{}{\Delta \vdash \alpha \, \mathbf{ok}} \quad \text{CHECK\_T\_VAR}$$

$$\frac{\begin{array}{c} \Delta \vdash t_1 \, \mathbf{ok} \\ \Delta \vdash t_2 \, \mathbf{ok} \end{array}}{\Delta \vdash t_1 \to t_2 \, \mathbf{ok}} \quad \text{CHECK\_T\_FN}$$

$$\frac{\Delta \vdash t_1 \, \mathbf{ok} \quad .... \quad \Delta \vdash t_n \, \mathbf{ok}}{\Delta \vdash t_1 * .... * t_n \, \mathbf{ok}} \quad \text{CHECK\_T\_TUP}$$

$$\frac{\begin{array}{c} \Delta(p) \triangleright tnv_1 .. tnv_n \; tc\_abbrev \\ \Delta, tnv_1 \vdash t_1 \, \mathbf{ok} \quad .. \quad \Delta, tnv_n \vdash t_n \, \mathbf{ok} \end{array}}{\Delta \vdash p \; t_1 .. t_n \, \mathbf{ok}} \quad \text{CHECK\_T\_APP}$$

$\boxed{\Delta, tnv \vdash t \, \mathbf{ok}}$ Well-formed type/Nexps matching the application type variable

$$\frac{\Delta \vdash t \, \mathbf{ok}}{\Delta, \alpha \vdash t \, \mathbf{ok}} \quad \text{CHECK\_TLEN\_T}$$

$$\frac{}{\Delta, N \vdash ne \, \mathbf{ok}} \quad \text{CHECK\_TLEN\_LEN}$$

$\boxed{\Delta \vdash t_1 = t_2}$ Type equality

$$\frac{\Delta \vdash t \, \mathbf{ok}}{\Delta \vdash t = t} \quad \text{TEQ\_REFL}$$

$$\frac{\Delta \vdash t_2 = t_1}{\Delta \vdash t_1 = t_2} \quad \text{TEQ\_SYM}$$

$$\frac{\begin{array}{c} \Delta \vdash t_1 = t_2 \\ \Delta \vdash t_2 = t_3 \end{array}}{\Delta \vdash t_1 = t_3} \quad \text{TEQ\_TRANS}$$

$$\frac{\begin{array}{c} \Delta \vdash t_1 = t_3 \\ \Delta \vdash t_2 = t_4 \end{array}}{\Delta \vdash t_1 \to t_2 = t_3 \to t_4} \quad \text{TEQ\_ARROW}$$

$$\frac{\Delta \vdash t_1 = u_1 \quad .... \quad \Delta \vdash t_n = u_n}{\Delta \vdash t_1 * .... * t_n = u_1 * .... * u_n} \quad \text{TEQ\_TUP}$$

$$\frac{\begin{array}{c}\Delta(p) \triangleright \alpha_1 .. \alpha_n \\ \Delta \vdash t_1 = u_1 \quad .. \quad \Delta \vdash t_n = u_n\end{array}}{\Delta \vdash p\ t_1 .. t_n = p\ u_1 .. u_n} \quad \text{TEQ\_APP}$$

$$\frac{\Delta(p) \triangleright \alpha_1 .. \alpha_n . u}{\Delta \vdash p\ t_1 .. t_n = \{\alpha_1 \mapsto t_1 .. \alpha_n \mapsto t_n\}(u)} \quad \text{TEQ\_EXPAND}$$

$$\frac{ne = \mathbf{normalize}\,(ne')}{\Delta \vdash ne = ne'} \quad \text{TEQ\_NEXP}$$

$\boxed{\Delta, E \vdash typ \rightsquigarrow t}$ Convert source types to internal types

$$\frac{}{\Delta, E \vdash \alpha\ l'\ l \rightsquigarrow \alpha} \quad \text{CONVERT\_TYP\_VAR}$$

$$\frac{\begin{array}{c}\Delta, E \vdash typ_1 \rightsquigarrow t_1 \\ \Delta, E \vdash typ_2 \rightsquigarrow t_2\end{array}}{\Delta, E \vdash typ_1 \rightarrow typ_2\ l \rightsquigarrow t_1 \rightarrow t_2} \quad \text{CONVERT\_TYP\_FN}$$

$$\frac{\Delta, E \vdash typ_1 \rightsquigarrow t_1 \quad .... \quad \Delta, E \vdash typ_n \rightsquigarrow t_n}{\Delta, E \vdash typ_1 * .... * typ_n\ l \rightsquigarrow t_1 * .... * t_n} \quad \text{CONVERT\_TYP\_TUP}$$

$$\frac{\begin{array}{c}\Delta, E \vdash typ_1 \rightsquigarrow t_1 \quad .. \quad \Delta, E \vdash typ_n \rightsquigarrow t_n \\ E(id) \triangleright p \\ \Delta(p) \triangleright \alpha_1 .. \alpha_n\ tc\_abbrev\end{array}}{\Delta, E \vdash id\ typ_1 .. typ_n\ l \rightsquigarrow p\ t_1 .. t_n} \quad \text{CONVERT\_TYP\_APP}$$

$$\frac{\vdash Nexp \rightsquigarrow ne}{\Delta, E \vdash Nexp \rightsquigarrow ne} \quad \text{CONVERT\_TYP\_NEXP}$$

$$\frac{\Delta, E \vdash typ \rightsquigarrow t}{\Delta, E \vdash (typ)\ l \rightsquigarrow t} \quad \text{CONVERT\_TYP\_PAREN}$$

$$\frac{\begin{array}{c}\Delta, E \vdash typ \rightsquigarrow t_1 \\ \Delta \vdash t_1 = t_2\end{array}}{\Delta, E \vdash typ \rightsquigarrow t_2} \quad \text{CONVERT\_TYP\_EQ}$$

$\boxed{\vdash Nexp \rightsquigarrow ne}$ Convert and normalize numeric expressions

$$\frac{}{\vdash N\ l \rightsquigarrow N} \quad \text{CONVERT\_NEXP\_VAR}$$

$$\frac{}{\vdash num\ l \rightsquigarrow nat} \quad \text{CONVERT\_NEXP\_NUM}$$

$$\frac{\begin{array}{c}\vdash Nexp_1 \rightsquigarrow ne_1 \\ \vdash Nexp_2 \rightsquigarrow ne_2\end{array}}{\vdash Nexp_1 * Nexp_2\ l \rightsquigarrow ne_1 * ne_2} \quad \text{CONVERT\_NEXP\_MULT}$$

$$\frac{\begin{array}{c}\vdash Nexp_1 \rightsquigarrow ne_1 \\ \vdash Nexp_2 \rightsquigarrow ne_2\end{array}}{\vdash Nexp_1 + Nexp_2\ l \rightsquigarrow ne_1 + ne_2} \quad \text{CONVERT\_NEXP\_ADD}$$

$\boxed{\Delta, E \vdash typs \rightsquigarrow t\_multi}$

$$\frac{\Delta, E \vdash typ_1 \rightsquigarrow t_1 \quad .. \quad \Delta, E \vdash typ_n \rightsquigarrow t_n}{\Delta, E \vdash typ_1 * .. * typ_n \rightsquigarrow (t_1 * .. * t_n)} \quad \text{CONVERT\_TYPS\_ALL}$$

$\boxed{\vdash lit : t}$ Typing literal constants

$$\frac{}{\vdash \textbf{true}\, l : \_\_\textbf{bool}} \quad \text{CHECK\_LIT\_TRUE}$$

$$\frac{}{\vdash \textbf{false}\, l : \_\_\textbf{bool}} \quad \text{CHECK\_LIT\_FALSE}$$

$$\frac{}{\vdash num\, l : \_\_\textbf{num}} \quad \text{CHECK\_LIT\_NUM}$$

$$\frac{nat = \textbf{bitlength}\,(hex)}{\vdash hex\, l : \_\_\textbf{vector}\, nat\, \_\_\textbf{bit}} \quad \text{CHECK\_LIT\_HEX}$$

$$\frac{nat = \textbf{bitlength}\,(bin)}{\vdash bin\, l : \_\_\textbf{vector}\, nat\, \_\_\textbf{bit}} \quad \text{CHECK\_LIT\_BIN}$$

$$\frac{}{\vdash string\, l : \_\_\textbf{string}} \quad \text{CHECK\_LIT\_STRING}$$

$$\frac{}{\vdash ()\, l : \_\_\textbf{unit}} \quad \text{CHECK\_LIT\_UNIT}$$

$$\frac{}{\vdash \textbf{bitzero}\, l : \_\_\textbf{bit}} \quad \text{CHECK\_LIT\_BITZERO}$$

$$\frac{}{\vdash \textbf{bitone}\, l : \_\_\textbf{bit}} \quad \text{CHECK\_LIT\_BITONE}$$

$\boxed{\Delta, E \vdash \textbf{field}\, id : p\, t\_args \to t \,\triangleright\, (x\, \textbf{of}\, names)}$ Field typing (also returns canonical field names)

$$\frac{\begin{array}{l} E(\overline{x_i^l}^{\,i}) \,\triangleright\, \langle E^{\mathrm{M}}, E^{\mathrm{P}}, E^{\mathrm{F}}, E^{\mathrm{X}}\rangle \\ E^{\mathrm{F}}(y) \,\triangleright\, \langle\, \textbf{forall}\, tnv_1 \,..\, tnv_n.p \to t, (z\, \textbf{of}\, names)\rangle \\ \Delta \vdash t_1\, \textbf{ok} \quad .. \quad \Delta \vdash t_n\, \textbf{ok} \end{array}}{\Delta, E \vdash \textbf{field}\, \overline{x_i^l.}^{\,i}\, y\, l_1\, l_2 : p\, t_1 \,..\, t_n \to \{tnv_1 \mapsto t_1 \,..\, tnv_n \mapsto t_n\}(t) \,\triangleright\, (z\, \textbf{of}\, names)} \quad \text{INST\_FIELD\_ALL}$$

$\boxed{\Delta, E \vdash \textbf{ctor}\, id : t\_multi \to p\, t\_args \,\triangleright\, (x\, \textbf{of}\, names)}$ Data constructor typing (also returns canonical constru

$$\frac{\begin{array}{l} E(\overline{x_i^l}^{\,i}) \,\triangleright\, \langle E^{\mathrm{M}}, E^{\mathrm{P}}, E^{\mathrm{F}}, E^{\mathrm{X}}\rangle \\ E^{\mathrm{X}}(y) \,\triangleright\, \langle\, \textbf{forall}\, tnv_1 \,..\, tnv_n.t\_multi \to p, (z\, \textbf{of}\, names)\rangle \\ \Delta \vdash t_1\, \textbf{ok} \quad .. \quad \Delta \vdash t_n\, \textbf{ok} \end{array}}{\Delta, E \vdash \textbf{ctor}\, \overline{x_i^l.}^{\,i}\, y\, l_1\, l_2 : \{tnv_1 \mapsto t_1 \,..\, tnv_n \mapsto t_n\}(t\_multi) \to p\, t_1 \,..\, t_n \,\triangleright\, (z\, \textbf{of}\, names)} \quad \text{INST\_CTOR\_ALL}$$

$\boxed{\Delta, E \vdash \textbf{val}\, id : t \,\triangleright\, \Sigma^{\mathcal{C}}}$ Typing top-level bindings, collecting typeclass constraints

$$\frac{\begin{array}{l} E(\overline{x_i^l}^{\,i}) \,\triangleright\, \langle E^{\mathrm{M}}, E^{\mathrm{P}}, E^{\mathrm{F}}, E^{\mathrm{X}}\rangle \\ E^{\mathrm{X}}(y) \,\triangleright\, \langle\, \textbf{forall}\, tnv_1 \,..\, tnv_n.(p_1\, tnv_1') \,..\, (p_i\, tnv_i') \Rightarrow t, env\_tag\rangle \\ \Delta \vdash t_1\, \textbf{ok} \quad .. \quad \Delta \vdash t_n\, \textbf{ok} \\ \sigma = \{tnv_1 \mapsto t_1 \,..\, tnv_n \mapsto t_n\} \end{array}}{\Delta, E \vdash \textbf{val}\, \overline{x_i^l.}^{\,i}\, y\, l_1\, l_2 : \sigma(t) \,\triangleright\, \{(p_1\, \sigma(tnv_1')), \,..\,, (p_i\, \sigma(tnv_i'))\}} \quad \text{INST\_VAL\_ALL}$$

$\boxed{E, E^{\mathrm{L}} \vdash x\, \textbf{not}\, \textbf{ctor}}$ v is not bound to a data constructor

$$\frac{E^{\mathrm{L}}(x) \,\triangleright\, t}{E, E^{\mathrm{L}} \vdash x\, \textbf{not}\, \textbf{ctor}} \quad \text{NOT\_CTOR\_VAL}$$

$$\frac{x \notin \textbf{dom}\,(E^{\mathrm{X}})}{\langle E^{\mathrm{M}}, E^{\mathrm{P}}, E^{\mathrm{F}}, E^{\mathrm{X}}\rangle, E^{\mathrm{L}} \vdash x\, \textbf{not}\, \textbf{ctor}} \quad \text{NOT\_CTOR\_UNBOUND}$$

$$\frac{E^{\mathrm{X}}(x) \,\triangleright\, \langle\, \textbf{forall}\, tnv_1 \,..\, tnv_n.(p_1\, tnv_1') \,..\, (p_i\, tnv_i') \Rightarrow t, env\_tag\rangle}{\langle E^{\mathrm{M}}, E^{\mathrm{P}}, E^{\mathrm{F}}, E^{\mathrm{X}}\rangle, E^{\mathrm{L}} \vdash x\, \textbf{not}\, \textbf{ctor}} \quad \text{NOT\_CTOR\_BOUND}$$

$$\boxed{E^{\mathrm{L}} \vdash id \ \textbf{not shadowed}} \qquad id \text{ is not lexically shadowed}$$

$$\frac{x \notin \textbf{dom}\,(E^{\mathrm{L}})}{E^{\mathrm{L}} \vdash \ x \ l_1 \ l_2 \ \textbf{not shadowed}} \quad \text{NOT\_SHADOWED\_SING}$$

$$\frac{}{E^{\mathrm{L}} \vdash x_1^l ... x_n^l . y^l . z^l \ l \ \textbf{not shadowed}} \quad \text{NOT\_SHADOWED\_MULTI}$$

$$\boxed{\Delta, E, E_1^{\mathrm{L}} \vdash pat : t \vartriangleright E_2^{\mathrm{L}}} \qquad \text{Typing patterns, building their binding environment}$$

$$\frac{\Delta, E, E_1^{\mathrm{L}} \vdash pat\_aux : t \vartriangleright E_2^{\mathrm{L}}}{\Delta, E, E_1^{\mathrm{L}} \vdash pat\_aux \ l : t \vartriangleright E_2^{\mathrm{L}}} \quad \text{CHECK\_PAT\_ALL}$$

$$\boxed{\Delta, E, E_1^{\mathrm{L}} \vdash pat\_aux : t \vartriangleright E_2^{\mathrm{L}}} \qquad \text{Typing patterns, building their binding environment}$$

$$\frac{\Delta \vdash t \ \textbf{ok}}{\Delta, E, E^{\mathrm{L}} \vdash \_ : t \vartriangleright \{\,\}} \quad \text{CHECK\_PAT\_AUX\_WILD}$$

$$\frac{\begin{array}{c}\Delta, E, E_1^{\mathrm{L}} \vdash pat : t \vartriangleright E_2^{\mathrm{L}} \\ x \notin \textbf{dom}\,(E_2^{\mathrm{L}})\end{array}}{\Delta, E, E_1^{\mathrm{L}} \vdash (pat \ \textbf{as} \ x \ l) : t \vartriangleright E_2^{\mathrm{L}} \uplus \{x \mapsto t\}} \quad \text{CHECK\_PAT\_AUX\_AS}$$

$$\frac{\begin{array}{c}\Delta, E, E_1^{\mathrm{L}} \vdash pat : t \vartriangleright E_2^{\mathrm{L}} \\ \Delta, E \vdash typ \leadsto t\end{array}}{\Delta, E, E_1^{\mathrm{L}} \vdash (pat : typ) : t \vartriangleright E_2^{\mathrm{L}}} \quad \text{CHECK\_PAT\_AUX\_TYP}$$

$$\frac{\begin{array}{c}\Delta, E \vdash \textbf{ctor} \ id : (t_1 * .. * t_n) \to p \ t\_args \vartriangleright (x \ \textbf{of} \ names) \\ E^{\mathrm{L}} \vdash id \ \textbf{not shadowed} \\ \Delta, E, E^{\mathrm{L}} \vdash pat_1 : t_1 \vartriangleright E_1^{\mathrm{L}} \quad .. \quad \Delta, E, E^{\mathrm{L}} \vdash pat_n : t_n \vartriangleright E_n^{\mathrm{L}} \\ \textbf{disjoint doms}\,(E_1^{\mathrm{L}}, .., E_n^{\mathrm{L}})\end{array}}{\Delta, E, E^{\mathrm{L}} \vdash id \ pat_1 .. \ pat_n : p \ t\_args \vartriangleright E_1^{\mathrm{L}} \uplus .. \uplus E_n^{\mathrm{L}}} \quad \text{CHECK\_PAT\_AUX\_IDENT\_CONSTR}$$

$$\frac{\begin{array}{c}\Delta \vdash t \ \textbf{ok} \\ E, E^{\mathrm{L}} \vdash x \ \textbf{not ctor}\end{array}}{\Delta, E, E^{\mathrm{L}} \vdash \ x \ l_1 \ l_2 \ : t \vartriangleright \{x \mapsto t\}} \quad \text{CHECK\_PAT\_AUX\_VAR}$$

$$\frac{\begin{array}{c}\overline{\Delta, E \vdash \textbf{field} \ id_i : p \ t\_args \to t_i \vartriangleright (x_i \ \textbf{of} \ names)}^{\,i} \\ \overline{\Delta, E, E^{\mathrm{L}} \vdash pat_i : t_i \vartriangleright E_i^{\mathrm{L}}}^{\,i} \\ \textbf{disjoint doms}\,(\overline{E_i^{\mathrm{L}}}^{\,i}\,) \\ \textbf{duplicates}\,(\overline{x_i}^{\,i}\,) = \emptyset\end{array}}{\Delta, E, E^{\mathrm{L}} \vdash \langle|\ \overline{id_i = pat_i \ l_i}^{\,i}\ ;^?|\rangle : p \ t\_args \vartriangleright \ \uplus \overline{E_i^{\mathrm{L}}}^{\,i}} \quad \text{CHECK\_PAT\_AUX\_RECORD}$$

$$\frac{\begin{array}{c}\Delta, E, E^{\mathrm{L}} \vdash pat_1 : t \vartriangleright E_1^{\mathrm{L}} \quad ... \quad \Delta, E, E^{\mathrm{L}} \vdash pat_n : t \vartriangleright E_n^{\mathrm{L}} \\ \textbf{disjoint doms}\,(E_1^{\mathrm{L}}, ..., E_n^{\mathrm{L}}) \\ \textbf{length}\,(pat_1 ... pat_n) = nat\end{array}}{\Delta, E, E^{\mathrm{L}} \vdash [|pat_1; ...; pat_n\,;^?|] : \_\_\textbf{vector} \ nat \ t \vartriangleright E_1^{\mathrm{L}} \uplus ... \uplus E_n^{\mathrm{L}}} \quad \text{CHECK\_PAT\_AUX\_VECTOR}$$

$$\frac{\begin{array}{c}\Delta, E, E^{\mathrm{L}} \vdash pat_1 : \_\_\textbf{vector} \ ne_1 \ t \vartriangleright E_1^{\mathrm{L}} \quad ... \quad \Delta, E, E^{\mathrm{L}} \vdash pat_n : \_\_\textbf{vector} \ ne_n \ t \vartriangleright E_n^{\mathrm{L}} \\ \textbf{disjoint doms}\,(E_1^{\mathrm{L}}, ..., E_n^{\mathrm{L}}) \\ ne' = ne_1 + ... + ne_n\end{array}}{\Delta, E, E^{\mathrm{L}} \vdash [|pat_1 ... pat_n|] : \_\_\textbf{vector} \ ne' \ t \vartriangleright E_1^{\mathrm{L}} \uplus ... \uplus E_n^{\mathrm{L}}} \quad \text{CHECK\_PAT\_AUX\_VECTOR(}$$

$$\frac{\begin{array}{c}\Delta, E, E^{\mathrm{L}} \vdash pat_1 : t_1 \vartriangleright E_1^{\mathrm{L}} \quad .... \quad \Delta, E, E^{\mathrm{L}} \vdash pat_n : t_n \vartriangleright E_n^{\mathrm{L}} \\ \textbf{disjoint doms}\,(E_1^{\mathrm{L}}, ..., E_n^{\mathrm{L}})\end{array}}{\Delta, E, E^{\mathrm{L}} \vdash (pat_1, ..., pat_n) : t_1 * .... * t_n \vartriangleright E_1^{\mathrm{L}} \uplus .... \uplus E_n^{\mathrm{L}}} \quad \text{CHECK\_PAT\_AUX\_TUP}$$

$$\frac{\begin{array}{l} \Delta \vdash t \, \mathbf{ok} \\ \Delta, E, E^{\mathrm{L}} \vdash pat_1 : t \rhd E_1^{\mathrm{L}} \quad .. \quad \Delta, E, E^{\mathrm{L}} \vdash pat_n : t \rhd E_n^{\mathrm{L}} \\ \mathbf{disjoint\,doms}\,(E_1^{\mathrm{L}}, .., E_n^{\mathrm{L}}) \end{array}}{\Delta, E, E^{\mathrm{L}} \vdash [pat_1; ..; pat_n\,;^?] : \_\mathbf{list}\, t \rhd E_1^{\mathrm{L}} \uplus .. \uplus E_n^{\mathrm{L}}} \quad \text{CHECK\_PAT\_AUX\_LIST}$$

$$\frac{\Delta, E, E_1^{\mathrm{L}} \vdash pat : t \rhd E_2^{\mathrm{L}}}{\Delta, E, E_1^{\mathrm{L}} \vdash (pat) : t \rhd E_2^{\mathrm{L}}} \quad \text{CHECK\_PAT\_AUX\_PAREN}$$

$$\frac{\begin{array}{l} \Delta, E, E_1^{\mathrm{L}} \vdash pat_1 : t \rhd E_2^{\mathrm{L}} \\ \Delta, E, E_1^{\mathrm{L}} \vdash pat_2 : \_\mathbf{list}\, t \rhd E_3^{\mathrm{L}} \\ \mathbf{disjoint\,doms}\,(E_2^{\mathrm{L}}, E_3^{\mathrm{L}}) \end{array}}{\Delta, E, E_1^{\mathrm{L}} \vdash pat_1 :: pat_2 : \_\mathbf{list}\, t \rhd E_2^{\mathrm{L}} \uplus E_3^{\mathrm{L}}} \quad \text{CHECK\_PAT\_AUX\_CONS}$$

$$\frac{\vdash lit : t}{\Delta, E, E^{\mathrm{L}} \vdash lit : t \rhd \{\,\}} \quad \text{CHECK\_PAT\_AUX\_LIT}$$

$$\frac{E, E^{\mathrm{L}} \vdash x \, \mathbf{not\,ctor}}{\Delta, E, E^{\mathrm{L}} \vdash x \, l + num : \_\mathbf{num} \rhd \{x \mapsto \_\mathbf{num}\,\}} \quad \text{CHECK\_PAT\_AUX\_NUM\_ADD}$$

$\boxed{E \vdash id \, \mathbf{field}}$ Check that the identifier is a permissible field identifier

$$\frac{E^{\mathrm{F}}(x) \rhd f\_desc}{\langle E^{\mathrm{M}}, E^{\mathrm{P}}, E^{\mathrm{F}}, E^{\mathrm{X}} \rangle \vdash x \, l_1 \, l_2 \, \mathbf{field}} \quad \text{ID\_FIELD\_EMPTY}$$

$$\frac{\begin{array}{l} E^{\mathrm{M}}(x) \rhd E \\ x \notin \mathbf{dom}\,(E^{\mathrm{F}}) \\ E \vdash \overline{y_i^l.}^{\,i} \, z^l \, l_2 \, \mathbf{field} \end{array}}{\langle E^{\mathrm{M}}, E^{\mathrm{P}}, E^{\mathrm{F}}, E^{\mathrm{X}} \rangle \vdash x \, l_1. \, \overline{y_i^l.}^{\,i} \, z^l \, l_2 \, \mathbf{field}} \quad \text{ID\_FIELD\_CONS}$$

$\boxed{E \vdash id \, \mathbf{value}}$ Check that the identifier is a permissible value identifier

$$\frac{E^{\mathrm{X}}(x) \rhd v\_desc}{\langle E^{\mathrm{M}}, E^{\mathrm{P}}, E^{\mathrm{F}}, E^{\mathrm{X}} \rangle \vdash x \, l_1 \, l_2 \, \mathbf{value}} \quad \text{ID\_VALUE\_EMPTY}$$

$$\frac{\begin{array}{l} E^{\mathrm{M}}(x) \rhd E \\ x \notin \mathbf{dom}\,(E^{\mathrm{X}}) \\ E \vdash \overline{y_i^l.}^{\,i} \, z^l \, l_2 \, \mathbf{value} \end{array}}{\langle E^{\mathrm{M}}, E^{\mathrm{P}}, E^{\mathrm{F}}, E^{\mathrm{X}} \rangle \vdash x \, l_1. \, \overline{y_i^l.}^{\,i} \, z^l \, l_2 \, \mathbf{value}} \quad \text{ID\_VALUE\_CONS}$$

$\boxed{\Delta, E, E^{\mathrm{L}} \vdash exp : t \rhd \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}}$ Typing expressions, collecting typeclass and index constraints

$$\frac{\Delta, E, E^{\mathrm{L}} \vdash exp\_aux : t \rhd \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}}{\Delta, E, E^{\mathrm{L}} \vdash exp\_aux \, l : t \rhd \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}} \quad \text{CHECK\_EXP\_ALL}$$

$\boxed{\Delta, E, E^{\mathrm{L}} \vdash exp\_aux : t \rhd \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}}$ Typing expressions, collecting typeclass and index constraints

$$\frac{E^{\mathrm{L}}(x) \rhd t}{\Delta, E, E^{\mathrm{L}} \vdash x \, l_1 \, l_2 : t \rhd \{\,\}, \{\,\}} \quad \text{CHECK\_EXP\_AUX\_VAR}$$

$$\frac{}{\Delta, E, E^{\mathrm{L}} \vdash N : \_\mathbf{num} \rhd \{\,\}, \{\,\}} \quad \text{CHECK\_EXP\_AUX\_NVAR}$$

$$\frac{\begin{array}{l} E^{\mathrm{L}} \vdash id \, \mathbf{not\,shadowed} \\ E \vdash id \, \mathbf{value} \\ \Delta, E \vdash \mathbf{ctor}\, id : t\_multi \to p \, t\_args \rhd (x \, \mathbf{of}\, names) \end{array}}{\Delta, E, E^{\mathrm{L}} \vdash id : \mathbf{curry}\,(t\_multi, p \, t\_args) \rhd \{\,\}, \{\,\}} \quad \text{CHECK\_EXP\_AUX\_CTOR}$$

$$\frac{\begin{array}{c} E^{\mathrm{L}} \vdash id \ \textbf{not shadowed} \\ E \vdash id \ \textbf{value} \\ \Delta, E \vdash \ \textbf{val} \ id : t \, \triangleright \, \Sigma^{\mathcal{C}} \end{array}}{\Delta, E, E^{\mathrm{L}} \vdash id : t \, \triangleright \, \Sigma^{\mathcal{C}}, \{\,\}} \quad \text{CHECK\_EXP\_AUX\_VAL}$$

$$\frac{\begin{array}{c} \Delta, E, E^{\mathrm{L}} \vdash pat_1 : t_1 \, \triangleright \, E_1^{\mathrm{L}} \quad ... \quad \Delta, E, E^{\mathrm{L}} \vdash pat_n : t_n \, \triangleright \, E_n^{\mathrm{L}} \\ \Delta, E, E^{\mathrm{L}} \uplus E_1^{\mathrm{L}} \uplus ... \uplus E_n^{\mathrm{L}} \vdash exp : u \, \triangleright \, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}} \\ \textbf{disjoint doms} \, (E_1^{\mathrm{L}}, ..., E_n^{\mathrm{L}}) \end{array}}{\Delta, E, E^{\mathrm{L}} \vdash \textbf{fun} \ pat_1 ... pat_n \rightarrow exp \ l : \textbf{curry} \, ((t_1 * ... * t_n), u) \, \triangleright \, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}} \quad \text{CHECK\_EXP\_AUX\_FN}$$

$$\frac{\begin{array}{c} \overline{\Delta, E, E^{\mathrm{L}} \vdash pat_i : t \, \triangleright \, E_i^{\mathrm{L}}}^{\ i} \\ \overline{\Delta, E, E^{\mathrm{L}} \uplus E_i^{\mathrm{L}} \vdash exp_i : u \, \triangleright \, \Sigma^{\mathcal{C}}_{\ i}, \Sigma^{\mathcal{N}}_{\ i}}^{\ i} \end{array}}{\Delta, E, E^{\mathrm{L}} \vdash \textbf{function} \ |^? \overline{pat_i \rightarrow exp_i \ l_i}^{\ i} \ \textbf{end} : t \rightarrow u \, \triangleright \, \overline{\Sigma^{\mathcal{C}}_{\ i}}^{\ i}, \overline{\Sigma^{\mathcal{N}}_{\ i}}^{\ i}} \quad \text{CHECK\_EXP\_AUX\_FUNCTION}$$

$$\frac{\begin{array}{c} \Delta, E, E^{\mathrm{L}} \vdash exp_1 : t_1 \rightarrow t_2 \, \triangleright \, \Sigma^{\mathcal{C}}_{\ 1}, \Sigma^{\mathcal{N}}_{\ 1} \\ \Delta, E, E^{\mathrm{L}} \vdash exp_2 : t_1 \, \triangleright \, \Sigma^{\mathcal{C}}_{\ 2}, \Sigma^{\mathcal{N}}_{\ 2} \end{array}}{\Delta, E, E^{\mathrm{L}} \vdash exp_1 \ exp_2 : t_2 \, \triangleright \, \Sigma^{\mathcal{C}}_{\ 1} \cup \Sigma^{\mathcal{C}}_{\ 2}, \Sigma^{\mathcal{N}}_{\ 1} \cup \Sigma^{\mathcal{N}}_{\ 2}} \quad \text{CHECK\_EXP\_AUX\_APP}$$

$$\frac{\begin{array}{c} \Delta, E, E^{\mathrm{L}} \vdash (ix) : t_1 \rightarrow t_2 \rightarrow t_3 \, \triangleright \, \Sigma^{\mathcal{C}}_{\ 1}, \Sigma^{\mathcal{N}}_{\ 1} \\ \Delta, E, E^{\mathrm{L}} \vdash exp_1 : t_1 \, \triangleright \, \Sigma^{\mathcal{C}}_{\ 2}, \Sigma^{\mathcal{N}}_{\ 2} \\ \Delta, E, E^{\mathrm{L}} \vdash exp_2 : t_2 \, \triangleright \, \Sigma^{\mathcal{C}}_{\ 3}, \Sigma^{\mathcal{N}}_{\ 3} \end{array}}{\Delta, E, E^{\mathrm{L}} \vdash exp_1 \ ix \ l \ exp_2 : t_3 \, \triangleright \, \Sigma^{\mathcal{C}}_{\ 1} \cup \Sigma^{\mathcal{C}}_{\ 2} \cup \Sigma^{\mathcal{C}}_{\ 3}, \Sigma^{\mathcal{N}}_{\ 1} \cup \Sigma^{\mathcal{N}}_{\ 2} \cup \Sigma^{\mathcal{N}}_{\ 3}} \quad \text{CHECK\_EXP\_AUX\_INFIX\_APP}1$$

$$\frac{\begin{array}{c} \Delta, E, E^{\mathrm{L}} \vdash x \ : t_1 \rightarrow t_2 \rightarrow t_3 \, \triangleright \, \Sigma^{\mathcal{C}}_{\ 1}, \Sigma^{\mathcal{N}}_{\ 1} \\ \Delta, E, E^{\mathrm{L}} \vdash exp_1 : t_1 \, \triangleright \, \Sigma^{\mathcal{C}}_{\ 2}, \Sigma^{\mathcal{N}}_{\ 2} \\ \Delta, E, E^{\mathrm{L}} \vdash exp_2 : t_2 \, \triangleright \, \Sigma^{\mathcal{C}}_{\ 3}, \Sigma^{\mathcal{N}}_{\ 3} \end{array}}{\Delta, E, E^{\mathrm{L}} \vdash exp_1 \ `x` l \ exp_2 : t_3 \, \triangleright \, \Sigma^{\mathcal{C}}_{\ 1} \cup \Sigma^{\mathcal{C}}_{\ 2} \cup \Sigma^{\mathcal{C}}_{\ 3}, \Sigma^{\mathcal{N}}_{\ 1} \cup \Sigma^{\mathcal{N}}_{\ 2} \cup \Sigma^{\mathcal{N}}_{\ 3}} \quad \text{CHECK\_EXP\_AUX\_INFIX\_APP}2$$

$$\frac{\begin{array}{c} \overline{\Delta, E \vdash \ \textbf{field} \ id_i : p \ t\_args \rightarrow t_i \, \triangleright \, (x_i \ \textbf{of} \ names)}^{\ i} \\ \overline{\Delta, E, E^{\mathrm{L}} \vdash exp_i : t_i \, \triangleright \, \Sigma^{\mathcal{C}}_{\ i}, \Sigma^{\mathcal{N}}_{\ i}}^{\ i} \\ \textbf{duplicates} \, (\overline{x_i}^{\ i}) = \emptyset \\ names = \{\, \overline{x_i}^{\ i} \,\} \end{array}}{\Delta, E, E^{\mathrm{L}} \vdash \langle | \overline{id_i = exp_i \ l_i}^{\ i} ;^? l | \rangle : p \ t\_args \, \triangleright \, \overline{\Sigma^{\mathcal{C}}_{\ i}}^{\ i}, \overline{\Sigma^{\mathcal{N}}_{\ i}}^{\ i}} \quad \text{CHECK\_EXP\_AUX\_RECORD}$$

$$\frac{\begin{array}{c} \overline{\Delta, E \vdash \ \textbf{field} \ id_i : p \ t\_args \rightarrow t_i \, \triangleright \, (x_i \ \textbf{of} \ names)}^{\ i} \\ \overline{\Delta, E, E^{\mathrm{L}} \vdash exp_i : t_i \, \triangleright \, \Sigma^{\mathcal{C}}_{\ i}, \Sigma^{\mathcal{N}}_{\ i}}^{\ i} \\ \textbf{duplicates} \, (\overline{x_i}^{\ i}) = \emptyset \\ \Delta, E, E^{\mathrm{L}} \vdash exp : p \ t\_args \, \triangleright \, \Sigma^{\mathcal{C}'}, \Sigma^{\mathcal{N}'} \end{array}}{\Delta, E, E^{\mathrm{L}} \vdash \langle | exp \ \textbf{with} \ \overline{id_i = exp_i \ l_i}^{\ i} ;^? l | \rangle : p \ t\_args \, \triangleright \, \Sigma^{\mathcal{C}'} \cup \overline{\Sigma^{\mathcal{C}}_{\ i}}^{\ i}, \Sigma^{\mathcal{N}'} \cup \overline{\Sigma^{\mathcal{N}}_{\ i}}^{\ i}} \quad \text{CHECK\_EXP\_AUX\_RECUP}$$

$$\frac{\begin{array}{c} \Delta, E, E^{\mathrm{L}} \vdash exp_1 : t \, \triangleright \, \Sigma^{\mathcal{C}}_{\ 1}, \Sigma^{\mathcal{N}}_{\ 1} \quad ... \quad \Delta, E, E^{\mathrm{L}} \vdash exp_n : t \, \triangleright \, \Sigma^{\mathcal{C}}_{\ n}, \Sigma^{\mathcal{N}}_{\ n} \\ \textbf{length} \, (exp_1 ... exp_n) = nat \end{array}}{\Delta, E, E^{\mathrm{L}} \vdash [| exp_1; ...; exp_n ;^? |] : \_\_\textbf{vector} \ nat \ t \, \triangleright \, \Sigma^{\mathcal{C}}_{\ 1} \cup ... \cup \Sigma^{\mathcal{C}}_{\ n}, \Sigma^{\mathcal{N}}_{\ 1} \cup ... \cup \Sigma^{\mathcal{N}}_{\ n}} \quad \text{CHECK\_EXP\_AUX\_VECTOR}$$

$$\frac{\begin{array}{c} \Delta, E, E^{\mathrm{L}} \vdash exp : \_\_\textbf{vector} \ ne' \ t \, \triangleright \, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}} \\ \vdash Nexp \rightsquigarrow ne \end{array}}{\Delta, E, E^{\mathrm{L}} \vdash exp.(Nexp) : t \, \triangleright \, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}} \cup \{ne\langle ne'\}} \quad \text{CHECK\_EXP\_AUX\_VECTORGET}$$

$$\frac{\begin{array}{c} \Delta, E, E^{\mathrm{L}} \vdash exp : \_\_\textbf{vector} \ ne' \ t \, \triangleright \, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}} \\ \vdash Nexp_1 \rightsquigarrow ne_1 \\ \vdash Nexp_2 \rightsquigarrow ne_2 \\ ne = ne_2 + (-ne_1) \end{array}}{\Delta, E, E^{\mathrm{L}} \vdash exp.(Nexp_1..Nexp_2) : \_\_\textbf{vector} \ ne \ t \, \triangleright \, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}} \cup \{ne_1\langle ne_2\langle ne'\}} \quad \text{CHECK\_EXP\_AUX\_VECTORSUB}$$

$$E \vdash id \, \textbf{field}$$
$$\Delta, E \vdash \textbf{field} \, id : p \, t\_args \to t \rhd (x \, \textbf{of} \, names)$$
$$\dfrac{\Delta, E, E^{\text{L}} \vdash exp : p \, t\_args \rhd \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}}{\Delta, E, E^{\text{L}} \vdash exp.id : t \rhd \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}} \quad \text{CHECK\_EXP\_AUX\_FIELD}$$

$$\dfrac{\begin{array}{c} \overline{\Delta, E, E^{\text{L}} \vdash pat_i : t \rhd E_i^{\text{L}}}^{\,i} \\ \overline{\Delta, E, E^{\text{L}} \uplus E_i^{\text{L}} \vdash exp_i : u \rhd \Sigma^{\mathcal{C}}_i, \Sigma^{\mathcal{N}}_i}^{\,i} \\ \Delta, E, E^{\text{L}} \vdash exp : t \rhd \Sigma^{\mathcal{C}'}, \Sigma^{\mathcal{N}'} \end{array}}{\Delta, E, E^{\text{L}} \vdash \textbf{match} \, exp \, \textbf{with} \, |^? \, \overline{pat_i \to exp_i \, l_i}^{\,i} \, l \, \textbf{end} : u \rhd \Sigma^{\mathcal{C}'} \cup \overline{\Sigma^{\mathcal{C}}_i}^{\,i}, \Sigma^{\mathcal{N}'} \cup \overline{\Sigma^{\mathcal{N}}_i}^{\,i}} \quad \text{CHECK\_EXP\_AUX\_CASE}$$

$$\dfrac{\begin{array}{c} \Delta, E, E^{\text{L}} \vdash exp : t \rhd \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}} \\ \Delta, E \vdash typ \rightsquigarrow t \end{array}}{\Delta, E, E^{\text{L}} \vdash (exp : typ) : t \rhd \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}} \quad \text{CHECK\_EXP\_AUX\_TYPED}$$

$$\dfrac{\begin{array}{c} \Delta, E, E_1^{\text{L}} \vdash letbind \rhd E_2^{\text{L}}, \Sigma^{\mathcal{C}}_1, \Sigma^{\mathcal{N}}_1 \\ \Delta, E, E_1^{\text{L}} \uplus E_2^{\text{L}} \vdash exp : t \rhd \Sigma^{\mathcal{C}}_2, \Sigma^{\mathcal{N}}_2 \end{array}}{\Delta, E, E_1^{\text{L}} \vdash \textbf{let} \, letbind \, \textbf{in} \, exp : t \rhd \Sigma^{\mathcal{C}}_1 \cup \Sigma^{\mathcal{C}}_2, \Sigma^{\mathcal{N}}_1 \cup \Sigma^{\mathcal{N}}_2} \quad \text{CHECK\_EXP\_AUX\_LET}$$

$$\dfrac{\Delta, E, E^{\text{L}} \vdash exp_1 : t_1 \rhd \Sigma^{\mathcal{C}}_1, \Sigma^{\mathcal{N}}_1 \quad .... \quad \Delta, E, E^{\text{L}} \vdash exp_n : t_n \rhd \Sigma^{\mathcal{C}}_n, \Sigma^{\mathcal{N}}_n}{\Delta, E, E^{\text{L}} \vdash (exp_1, ..., exp_n) : t_1 * .... * t_n \rhd \Sigma^{\mathcal{C}}_1 \cup .... \cup \Sigma^{\mathcal{C}}_n, \Sigma^{\mathcal{N}}_1 \cup .... \cup \Sigma^{\mathcal{N}}_n} \quad \text{CHECK\_EXP\_AUX\_TUP}$$

$$\dfrac{\begin{array}{c} \Delta \vdash t \, \textbf{ok} \\ \Delta, E, E^{\text{L}} \vdash exp_1 : t \rhd \Sigma^{\mathcal{C}}_1, \Sigma^{\mathcal{N}}_1 \quad .. \quad \Delta, E, E^{\text{L}} \vdash exp_n : t \rhd \Sigma^{\mathcal{C}}_n, \Sigma^{\mathcal{N}}_n \end{array}}{\Delta, E, E^{\text{L}} \vdash [exp_1; ..; exp_n ;^?] : \_\textbf{list} \, t \rhd \Sigma^{\mathcal{C}}_1 \cup .. \cup \Sigma^{\mathcal{C}}_n, \Sigma^{\mathcal{N}}_1 \cup .. \cup \Sigma^{\mathcal{N}}_n} \quad \text{CHECK\_EXP\_AUX\_LIST}$$

$$\dfrac{\Delta, E, E^{\text{L}} \vdash exp : t \rhd \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}}{\Delta, E, E^{\text{L}} \vdash (exp) : t \rhd \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}} \quad \text{CHECK\_EXP\_AUX\_PAREN}$$

$$\dfrac{\Delta, E, E^{\text{L}} \vdash exp : t \rhd \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}}{\Delta, E, E^{\text{L}} \vdash \textbf{begin} \, exp \, \textbf{end} : t \rhd \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}} \quad \text{CHECK\_EXP\_AUX\_BEGIN}$$

$$\dfrac{\begin{array}{c} \Delta, E, E^{\text{L}} \vdash exp_1 : \_\textbf{bool} \rhd \Sigma^{\mathcal{C}}_1, \Sigma^{\mathcal{N}}_1 \\ \Delta, E, E^{\text{L}} \vdash exp_2 : t \rhd \Sigma^{\mathcal{C}}_2, \Sigma^{\mathcal{N}}_2 \\ \Delta, E, E^{\text{L}} \vdash exp_3 : t \rhd \Sigma^{\mathcal{C}}_3, \Sigma^{\mathcal{N}}_3 \end{array}}{\Delta, E, E^{\text{L}} \vdash \textbf{if} \, exp_1 \, \textbf{then} \, exp_2 \, \textbf{else} \, exp_3 : t \rhd \Sigma^{\mathcal{C}}_1 \cup \Sigma^{\mathcal{C}}_2 \cup \Sigma^{\mathcal{C}}_3, \Sigma^{\mathcal{N}}_1 \cup \Sigma^{\mathcal{N}}_2 \cup \Sigma^{\mathcal{N}}_3} \quad \text{CHECK\_EXP\_AUX\_IF}$$

$$\dfrac{\begin{array}{c} \Delta, E, E^{\text{L}} \vdash exp_1 : t \rhd \Sigma^{\mathcal{C}}_1, \Sigma^{\mathcal{N}}_1 \\ \Delta, E, E^{\text{L}} \vdash exp_2 : \_\textbf{list} \, t \rhd \Sigma^{\mathcal{C}}_2, \Sigma^{\mathcal{N}}_2 \end{array}}{\Delta, E, E^{\text{L}} \vdash exp_1 :: exp_2 : \_\textbf{list} \, t \rhd \Sigma^{\mathcal{C}}_1 \cup \Sigma^{\mathcal{C}}_2, \Sigma^{\mathcal{N}}_1 \cup \Sigma^{\mathcal{N}}_2} \quad \text{CHECK\_EXP\_AUX\_CONS}$$

$$\dfrac{\vdash lit : t}{\Delta, E, E^{\text{L}} \vdash lit : t \rhd \{\,\}, \{\,\}} \quad \text{CHECK\_EXP\_AUX\_LIT}$$

$$\dfrac{\begin{array}{c} \overline{\Delta \vdash t_i \, \textbf{ok}}^{\,i} \\ \Delta, E, E^{\text{L}} \uplus \{\, \overline{x_i \mapsto t_i}^{\,i} \,\} \vdash exp_1 : t \rhd \Sigma^{\mathcal{C}}_1, \Sigma^{\mathcal{N}}_1 \\ \Delta, E, E^{\text{L}} \uplus \{\, \overline{x_i \mapsto t_i}^{\,i} \,\} \vdash exp_2 : \_\textbf{bool} \rhd \Sigma^{\mathcal{C}}_2, \Sigma^{\mathcal{N}}_2 \\ \textbf{disjoint doms} \, (E^{\text{L}}, \{\, \overline{x_i \mapsto t_i}^{\,i} \,\}) \\ E = \langle E^{\text{M}}, E^{\text{P}}, E^{\text{F}}, E^{\text{X}} \rangle \\ \overline{x_i \notin \textbf{dom} \, (E^{\text{X}})}^{\,i} \end{array}}{\Delta, E, E^{\text{L}} \vdash \{ exp_1 | exp_2 \} : \_\textbf{set} \, t \rhd \Sigma^{\mathcal{C}}_1 \cup \Sigma^{\mathcal{C}}_2, \Sigma^{\mathcal{N}}_1 \cup \Sigma^{\mathcal{N}}_2} \quad \text{CHECK\_EXP\_AUX\_SET\_COMP}$$

$$\dfrac{\begin{array}{c} \Delta, E, E_1^{\text{L}} \vdash \overline{qbind_i}^{\,i} \rhd E_2^{\text{L}}, \Sigma^{\mathcal{C}}_1 \\ \Delta, E, E_1^{\text{L}} \uplus E_2^{\text{L}} \vdash exp_1 : t \rhd \Sigma^{\mathcal{C}}_2, \Sigma^{\mathcal{N}}_2 \\ \Delta, E, E_1^{\text{L}} \uplus E_2^{\text{L}} \vdash exp_2 : \_\textbf{bool} \rhd \Sigma^{\mathcal{C}}_3, \Sigma^{\mathcal{N}}_3 \end{array}}{\Delta, E, E_1^{\text{L}} \vdash \{ exp_1 | \textbf{forall} \, \overline{qbind_i}^{\,i} | exp_2 \} : \_\textbf{set} \, t \rhd \Sigma^{\mathcal{C}}_1 \cup \Sigma^{\mathcal{C}}_2 \cup \Sigma^{\mathcal{C}}_3, \Sigma^{\mathcal{N}}_2 \cup \Sigma^{\mathcal{N}}_3} \quad \text{CHECK\_EXP\_AUX\_SET\_COMP\_}$$

$$\frac{\begin{array}{c}\Delta \vdash t\,\mathbf{ok}\\ \Delta, E, E^{\mathrm{L}} \vdash exp_1 : t \vartriangleright \Sigma^{\mathcal{C}}{}_1, \Sigma^{\mathcal{N}}{}_1 \quad .. \quad \Delta, E, E^{\mathrm{L}} \vdash exp_n : t \vartriangleright \Sigma^{\mathcal{C}}{}_n, \Sigma^{\mathcal{N}}{}_n\end{array}}{\Delta, E, E^{\mathrm{L}} \vdash \{exp_1; ..; exp_n;^?\} : \_\mathbf{set}\, t \vartriangleright \Sigma^{\mathcal{C}}{}_1 \cup .. \cup \Sigma^{\mathcal{C}}{}_n, \Sigma^{\mathcal{N}}{}_1 \cup .. \cup \Sigma^{\mathcal{N}}{}_n} \quad \text{CHECK\_EXP\_AUX\_SET}$$

$$\frac{\begin{array}{c}\Delta, E, E^{\mathrm{L}}_1 \vdash \overline{qbind_i}^{\ i} \vartriangleright E^{\mathrm{L}}_2, \Sigma^{\mathcal{C}}{}_1\\ \Delta, E, E^{\mathrm{L}}_1 \uplus E^{\mathrm{L}}_2 \vdash exp : \_\mathbf{bool} \vartriangleright \Sigma^{\mathcal{C}}{}_2, \Sigma^{\mathcal{N}}{}_2\end{array}}{\Delta, E, E^{\mathrm{L}}_1 \vdash q\, \overline{qbind_i}^{\ i} .exp : \_\mathbf{bool} \vartriangleright \Sigma^{\mathcal{C}}{}_1 \cup \Sigma^{\mathcal{C}}{}_2, \Sigma^{\mathcal{N}}{}_2} \quad \text{CHECK\_EXP\_AUX\_QUANT}$$

$$\frac{\begin{array}{c}\Delta, E, E^{\mathrm{L}}_1 \vdash \mathbf{list}\, \overline{qbind_i}^{\ i} \vartriangleright E^{\mathrm{L}}_2, \Sigma^{\mathcal{C}}{}_1\\ \Delta, E, E^{\mathrm{L}}_1 \uplus E^{\mathrm{L}}_2 \vdash exp_1 : t \vartriangleright \Sigma^{\mathcal{C}}{}_2, \Sigma^{\mathcal{N}}{}_2\\ \Delta, E, E^{\mathrm{L}}_1 \uplus E^{\mathrm{L}}_2 \vdash exp_2 : \_\mathbf{bool} \vartriangleright \Sigma^{\mathcal{C}}{}_3, \Sigma^{\mathcal{N}}{}_3\end{array}}{\Delta, E, E^{\mathrm{L}}_1 \vdash [exp_1|\,\mathbf{forall}\, \overline{qbind_i}^{\ i}|exp_2] : \_\mathbf{list}\, t \vartriangleright \Sigma^{\mathcal{C}}{}_1 \cup \Sigma^{\mathcal{C}}{}_2 \cup \Sigma^{\mathcal{C}}{}_3, \Sigma^{\mathcal{N}}{}_2 \cup \Sigma^{\mathcal{N}}{}_3} \quad \text{CHECK\_EXP\_AUX\_LIST\_COMP\_}$$

$\boxed{\Delta, E, E^{\mathrm{L}}_1 \vdash qbind_1 .. qbind_n \vartriangleright E^{\mathrm{L}}_2, \Sigma^{\mathcal{C}}}$ Build the environment for quantifier bindings, collecting typeclass cons

$$\frac{}{\Delta, E, E^{\mathrm{L}} \vdash \vartriangleright \{\,\}, \{\,\}} \quad \text{CHECK\_LISTQUANT\_BINDING\_EMPTY}$$

$$\frac{\begin{array}{c}\Delta \vdash t\,\mathbf{ok}\\ \Delta, E, E^{\mathrm{L}}_1 \uplus \{x \mapsto t\} \vdash \overline{qbind_i}^{\ i} \vartriangleright E^{\mathrm{L}}_2, \Sigma^{\mathcal{C}}{}_1\\ \mathbf{disjoint\,doms}\,(\{x \mapsto t\}, E^{\mathrm{L}}_2)\end{array}}{\Delta, E, E^{\mathrm{L}}_1 \vdash x\, l\, \overline{qbind_i}^{\ i} \vartriangleright \{x \mapsto t\} \uplus E^{\mathrm{L}}_2, \Sigma^{\mathcal{C}}{}_1} \quad \text{CHECK\_LISTQUANT\_BINDING\_VAR}$$

$$\frac{\begin{array}{c}\Delta, E, E^{\mathrm{L}}_1 \vdash pat : t \vartriangleright E^{\mathrm{L}}_3\\ \Delta, E, E^{\mathrm{L}}_1 \vdash exp : \_\mathbf{set}\, t \vartriangleright \Sigma^{\mathcal{C}}{}_1, \Sigma^{\mathcal{N}}{}_1\\ \Delta, E, E^{\mathrm{L}}_1 \uplus E^{\mathrm{L}}_3 \vdash \overline{qbind_i}^{\ i} \vartriangleright E^{\mathrm{L}}_2, \Sigma^{\mathcal{C}}{}_2\\ \mathbf{disjoint\,doms}\,(E^{\mathrm{L}}_3, E^{\mathrm{L}}_2)\end{array}}{\Delta, E, E^{\mathrm{L}}_1 \vdash (pat\,\mathbf{IN}\, exp)\, \overline{qbind_i}^{\ i} \vartriangleright E^{\mathrm{L}}_2 \uplus E^{\mathrm{L}}_3, \Sigma^{\mathcal{C}}{}_1 \cup \Sigma^{\mathcal{C}}{}_2} \quad \text{CHECK\_LISTQUANT\_BINDING\_RESTR}$$

$$\frac{\begin{array}{c}\Delta, E, E^{\mathrm{L}}_1 \vdash pat : t \vartriangleright E^{\mathrm{L}}_3\\ \Delta, E, E^{\mathrm{L}}_1 \vdash exp : \_\mathbf{list}\, t \vartriangleright \Sigma^{\mathcal{C}}{}_1, \Sigma^{\mathcal{N}}{}_1\\ \Delta, E, E^{\mathrm{L}}_1 \uplus E^{\mathrm{L}}_3 \vdash \overline{qbind_i}^{\ i} \vartriangleright E^{\mathrm{L}}_2, \Sigma^{\mathcal{C}}{}_2\\ \mathbf{disjoint\,doms}\,(E^{\mathrm{L}}_3, E^{\mathrm{L}}_2)\end{array}}{\Delta, E, E^{\mathrm{L}}_1 \vdash (pat\,\mathbf{MEM}\, exp)\, \overline{qbind_i}^{\ i} \vartriangleright E^{\mathrm{L}}_2 \uplus E^{\mathrm{L}}_3, \Sigma^{\mathcal{C}}{}_1 \cup \Sigma^{\mathcal{C}}{}_2} \quad \text{CHECK\_LISTQUANT\_BINDING\_LIST\_RESTR}$$

$\boxed{\Delta, E, E^{\mathrm{L}}_1 \vdash \mathbf{list}\, qbind_1 .. qbind_n \vartriangleright E^{\mathrm{L}}_2, \Sigma^{\mathcal{C}}}$ Build the environment for quantifier bindings, collecting typeclass

$$\frac{}{\Delta, E, E^{\mathrm{L}} \vdash \mathbf{list} \vartriangleright \{\,\}, \{\,\}} \quad \text{CHECK\_QUANT\_BINDING\_EMPTY}$$

$$\frac{\begin{array}{c}\Delta, E, E^{\mathrm{L}}_1 \vdash pat : t \vartriangleright E^{\mathrm{L}}_3\\ \Delta, E, E^{\mathrm{L}}_1 \vdash exp : \_\mathbf{list}\, t \vartriangleright \Sigma^{\mathcal{C}}{}_1, \Sigma^{\mathcal{N}}{}_1\\ \Delta, E, E^{\mathrm{L}}_1 \uplus E^{\mathrm{L}}_3 \vdash \overline{qbind_i}^{\ i} \vartriangleright E^{\mathrm{L}}_2, \Sigma^{\mathcal{C}}{}_2\\ \mathbf{disjoint\,doms}\,(E^{\mathrm{L}}_3, E^{\mathrm{L}}_2)\end{array}}{\Delta, E, E^{\mathrm{L}}_1 \vdash \mathbf{list}\, (pat\,\mathbf{MEM}\, exp)\, \overline{qbind_i}^{\ i} \vartriangleright E^{\mathrm{L}}_2 \uplus E^{\mathrm{L}}_3, \Sigma^{\mathcal{C}}{}_1 \cup \Sigma^{\mathcal{C}}{}_2} \quad \text{CHECK\_QUANT\_BINDING\_RESTR}$$

$\boxed{\Delta, E, E^{\mathrm{L}} \vdash funcl \vartriangleright \{x \mapsto t\}, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}}$ Build the environment for a function definition clause, collecting typecl

$$\frac{\begin{array}{c}\Delta, E, E^{\mathrm{L}} \vdash pat_1 : t_1 \vartriangleright E^{\mathrm{L}}_1 \quad ... \quad \Delta, E, E^{\mathrm{L}} \vdash pat_n : t_n \vartriangleright E^{\mathrm{L}}_n\\ \Delta, E, E^{\mathrm{L}} \uplus E^{\mathrm{L}}_1 \uplus ... \uplus E^{\mathrm{L}}_n \vdash exp : u \vartriangleright \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}\\ \mathbf{disjoint\,doms}\,(E^{\mathrm{L}}_1, ..., E^{\mathrm{L}}_n)\\ \Delta, E \vdash typ \rightsquigarrow u\end{array}}{\Delta, E, E^{\mathrm{L}} \vdash x\, l_1\, pat_1 ... pat_n : typ = exp\, l_2 \vartriangleright \{x \mapsto \mathbf{curry}\,((t_1 * ... * t_n), u)\}, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}} \quad \text{CHECK\_FUNCL\_ANNOT}$$

$$\dfrac{\begin{array}{l}\Delta, E, E^{\mathrm{L}} \vdash pat_1 : t_1 \rhd E_1^{\mathrm{L}} \quad ... \quad \Delta, E, E^{\mathrm{L}} \vdash pat_n : t_n \rhd E_n^{\mathrm{L}} \\ \Delta, E, E^{\mathrm{L}} \uplus E_1^{\mathrm{L}} \uplus ... \uplus E_n^{\mathrm{L}} \vdash exp : u \rhd \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}} \\ \mathbf{disjoint\ doms}\,(E_1^{\mathrm{L}}, ..., E_n^{\mathrm{L}})\end{array}}{\Delta, E, E^{\mathrm{L}} \vdash x\ l_1\ pat_1 ... pat_n = exp\ l_2 \rhd \{x \mapsto \mathbf{curry}\,((t_1 * ... * t_n), u)\}, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}} \quad \text{\footnotesize CHECK\_FUNCL\_NOANNOT}$$

$\boxed{\Delta, E, E_1^{\mathrm{L}} \vdash letbind \rhd E_2^{\mathrm{L}}, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}}$ Build the environment for a let binding, collecting typeclass and index con

$$\dfrac{\begin{array}{l}\Delta, E, E_1^{\mathrm{L}} \vdash pat : t \rhd E_2^{\mathrm{L}} \\ \Delta, E, E_1^{\mathrm{L}} \vdash exp : t \rhd \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}} \\ \Delta, E \vdash typ \rightsquigarrow t\end{array}}{\Delta, E, E_1^{\mathrm{L}} \vdash pat : typ = exp\ l \rhd E_2^{\mathrm{L}}, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}} \quad \text{\footnotesize CHECK\_LETBIND\_VAL\_ANNOT}$$

$$\dfrac{\begin{array}{l}\Delta, E, E_1^{\mathrm{L}} \vdash pat : t \rhd E_2^{\mathrm{L}} \\ \Delta, E, E_1^{\mathrm{L}} \vdash exp : t \rhd \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}\end{array}}{\Delta, E, E_1^{\mathrm{L}} \vdash pat = exp\ l \rhd E_2^{\mathrm{L}}, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}} \quad \text{\footnotesize CHECK\_LETBIND\_VAL\_NOANNOT}$$

$$\dfrac{\Delta, E, E_1^{\mathrm{L}} \vdash funcl\_aux\ l \rhd \{x \mapsto t\}, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}}{\Delta, E, E_1^{\mathrm{L}} \vdash funcl\_aux\ l \rhd \{x \mapsto t\}, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}} \quad \text{\footnotesize CHECK\_LETBIND\_FN}$$

$\boxed{\Delta, E, E^{\mathrm{L}} \vdash rule \rhd \{x \mapsto t\}, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}}$ Build the environment for an inductive relation clause, collecting typecl

$$\dfrac{\begin{array}{l}\overline{\Delta \vdash t_i\,\mathbf{ok}}^{\,i} \\ E_2^{\mathrm{L}} = \{\,\overline{name\_t_i \to x \mapsto t_i}^{\,i}\,\} \\ \Delta, E, E_1^{\mathrm{L}} \uplus E_2^{\mathrm{L}} \vdash exp' : \_\_\mathbf{bool} \rhd \Sigma^{\mathcal{C}'}, \Sigma^{\mathcal{N}'} \\ \Delta, E, E_1^{\mathrm{L}} \uplus E_2^{\mathrm{L}} \vdash exp_1 : u_1 \rhd \Sigma^{\mathcal{C}}{}_1, \Sigma^{\mathcal{N}}{}_1 \quad .. \quad \Delta, E, E_1^{\mathrm{L}} \uplus E_2^{\mathrm{L}} \vdash exp_n : u_n \rhd \Sigma^{\mathcal{C}}{}_n, \Sigma^{\mathcal{N}}\end{array}}{\Delta, E, E_1^{\mathrm{L}} \vdash x_1^l : \mathbf{forall}\,\overline{name\_t_i}^{\,i}.exp' \Longrightarrow x\ l\ exp_1 .. exp_n\ l' \rhd \{x \mapsto \mathbf{curry}\,((u_1 * .. * u_n), \_\_\mathbf{bool}\,)\}, \Sigma^{\mathcal{C}'} \cup \Sigma^{\mathcal{C}}{}_1 \cup .}$$

$\boxed{xs, \Delta_1, E \vdash \mathbf{tc}\ td \rhd \Delta_2, E^{\mathrm{P}}}$ Extract the type constructor information

$$\dfrac{\begin{array}{l}tnvars^l \rightsquigarrow tnvs \\ \Delta, E \vdash typ \rightsquigarrow t \\ \mathbf{duplicates}\,(tnvs) = \emptyset \\ \mathbf{FV}\,(t) \subset tnvs \\ \overline{y_i.}^{\,i}\ x \notin \mathbf{dom}\,(\Delta)\end{array}}{\overline{y_i}^{\,i}, \Delta, E \vdash \mathbf{tc}\ x\ l\ tnvars^l = typ \rhd \{\overline{y_i.}^{\,i}\ x \mapsto tnvs\,.t\}, \{x \mapsto \overline{y_i.}^{\,i}\ x\}} \quad \text{\footnotesize CHECK\_TEXP\_TC\_ABBREV}$$

$$\dfrac{\begin{array}{l}tnvars^l \rightsquigarrow tnvs \\ \mathbf{duplicates}\,(tnvs) = \emptyset \\ \overline{y_i.}^{\,i}\ x \notin \mathbf{dom}\,(\Delta)\end{array}}{\overline{y_i}^{\,i}, \Delta, E_1 \vdash \mathbf{tc}\ x\ l\ tnvars^l \rhd \{\overline{y_i.}^{\,i}\ x \mapsto tnvs\,\}, \{x \mapsto \overline{y_i.}^{\,i}\ x\}} \quad \text{\footnotesize CHECK\_TEXP\_TC\_ABSTRACT}$$

$$\dfrac{\begin{array}{l}tnvars^l \rightsquigarrow tnvs \\ \mathbf{duplicates}\,(tnvs) = \emptyset \\ \overline{y_i.}^{\,i}\ x \notin \mathbf{dom}\,(\Delta)\end{array}}{\overline{y_i}^{\,i}, \Delta_1, E \vdash \mathbf{tc}\ x\ l\ tnvars^l = \langle|x_1^l : typ_1; ...; x_j^l : typ_j\,;^?|\rangle \rhd \{\overline{y_i.}^{\,i}\ x \mapsto tnvs\,\}, \{x \mapsto \overline{y_i.}^{\,i}\ x\}} \quad \text{\footnotesize CHECK\_TEXP\_TC\_REC}$$

$$\dfrac{\begin{array}{l}tnvars^l \rightsquigarrow tnvs \\ \mathbf{duplicates}\,(tnvs) = \emptyset \\ \overline{y_i.}^{\,i}\ x \notin \mathbf{dom}\,(\Delta)\end{array}}{\overline{y_i}^{\,i}, \Delta_1, E \vdash \mathbf{tc}\ x\ l\ tnvars^l = |^? ctor\_def_1| ... |ctor\_def_j \rhd \{\overline{y_i.}^{\,i}\ x \mapsto tnvs\,\}, \{x \mapsto \overline{y_i.}^{\,i}\ x\}} \quad \text{\footnotesize CHECK\_TEXP\_TC\_VAR}$$

$\boxed{xs, \Delta_1, E \vdash \mathbf{tc}\ td_1 .. td_i \rhd \Delta_2, E^{\mathrm{P}}}$ Extract the type constructor information

$$\dfrac{}{xs, \Delta, E \vdash \mathbf{tc} \rhd \{\,\}, \{\,\}} \quad \text{\footnotesize CHECK\_TEXPS\_TC\_EMPTY}$$

$$xs, \Delta_1, E \vdash \textbf{tc}\; td \rhd \Delta_2, E_2^{\mathrm{P}}$$
$$xs, \Delta_1 \uplus \Delta_2, E \uplus \langle \{\,\}, E_2^{\mathrm{P}}, \{\,\}, \{\,\} \rangle \vdash \textbf{tc}\; \overline{td_i}^{\,i} \rhd \Delta_3, E_3^{\mathrm{P}}$$
$$\textbf{dom}\,(E_2^{\mathrm{P}}) \cap \textbf{dom}\,(E_3^{\mathrm{P}}) = \emptyset$$
$$\rule{6cm}{0.4pt}$$
$$xs, \Delta_1, E \vdash \textbf{tc}\; td\; \overline{td_i}^{\,i} \rhd \Delta_2 \uplus \Delta_3, E_2^{\mathrm{P}} \uplus E_3^{\mathrm{P}} \qquad \text{CHECK\_TEXPS\_TC\_ABBREV}$$

$\boxed{\Delta, E \vdash \textit{tnvs}\; p = \textit{texp} \rhd \langle E^{\mathrm{F}}, E^{\mathrm{X}} \rangle}$ Check a type definition, with its path already resolved

$$\frac{}{\Delta, E \vdash \textit{tnvs}\; p = \textit{typ} \rhd \langle \{\,\}, \{\,\} \rangle} \qquad \text{CHECK\_TEXP\_ABBREV}$$

$$\overline{\Delta, E \vdash \textit{typ}_i \rightsquigarrow t_i}^{\,i}$$
$$\textit{names} = \{\, \overline{x_i}^{\,i} \,\}$$
$$\textbf{duplicates}\,(\,\overline{x_i}^{\,i}\,) = \emptyset$$
$$\overline{\textbf{FV}\,(t_i) \subset \textit{tnvs}}^{\,i}$$
$$E^{\mathrm{F}} = \{\, \overline{x_i \mapsto \langle \textbf{forall}\; \textit{tnvs}.p \to t_i, (x_i\; \textbf{of}\; \textit{names}) \rangle}^{\,i} \,\}$$
$$\rule{7cm}{0.4pt}$$
$$\Delta, E \vdash \textit{tnvs}\; p = \langle |\; \overline{x_i^l : \textit{typ}_i}^{\,i}\; ;^? | \rangle \rhd \langle E^{\mathrm{F}}, \{\,\} \rangle \qquad \text{CHECK\_TEXP\_REC}$$

$$\overline{\Delta, E \vdash \textit{typs}_i \rightsquigarrow t\_multi_i}^{\,i}$$
$$\textit{names} = \{\, \overline{x_i}^{\,i} \,\}$$
$$\textbf{duplicates}\,(\,\overline{x_i}^{\,i}\,) = \emptyset$$
$$\overline{\textbf{FV}\,(t\_multi_i) \subset \textit{tnvs}}^{\,i}$$
$$E^{\mathrm{X}} = \{\, \overline{x_i \mapsto \langle \textbf{forall}\; \textit{tnvs}.t\_multi_i \to p, (x_i\; \textbf{of}\; \textit{names}) \rangle}^{\,i} \,\}$$
$$\rule{7cm}{0.4pt}$$
$$\Delta, E \vdash \textit{tnvs}\; p = |^?\; \overline{x_i^l\; \textbf{of}\; \textit{typs}_i}^{\,i} \rhd \langle \{\,\}, E^{\mathrm{X}} \rangle \qquad \text{CHECK\_TEXP\_VAR}$$

$\boxed{xs, \Delta, E \vdash td_1 \,..\, td_n \rhd \langle E^{\mathrm{F}}, E^{\mathrm{X}} \rangle}$

$$\frac{}{\overline{y_i}^{\,i}, \Delta, E \vdash \rhd \langle \{\,\}, \{\,\} \rangle} \qquad \text{CHECK\_TEXPS\_EMPTY}$$

$$\textit{tnvars}^l \rightsquigarrow \textit{tnvs}$$
$$\Delta, E_1 \vdash \textit{tnvs}\; \overline{y_i.}^{\,i}\; x = \textit{texp} \rhd \langle E_1^{\mathrm{F}}, E_1^{\mathrm{X}} \rangle$$
$$\overline{y_i}^{\,i}, \Delta, E \vdash \overline{td_j}^{\,j} \rhd \langle E_2^{\mathrm{F}}, E_2^{\mathrm{X}} \rangle$$
$$\textbf{dom}\,(E_1^{\mathrm{X}}) \cap \textbf{dom}\,(E_2^{\mathrm{X}}) = \emptyset$$
$$\textbf{dom}\,(E_1^{\mathrm{F}}) \cap \textbf{dom}\,(E_2^{\mathrm{F}}) = \emptyset$$
$$\rule{8cm}{0.4pt}$$
$$\overline{y_i}^{\,i}, \Delta, E \vdash x\; l\; \textit{tnvars}^l = \textit{texp}\; \overline{td_j}^{\,j} \rhd \langle E_1^{\mathrm{F}} \uplus E_2^{\mathrm{F}}, E_1^{\mathrm{X}} \uplus E_2^{\mathrm{X}} \rangle \qquad \text{CHECK\_TEXPS\_CONS\_CONCRETE}$$

$$\frac{\overline{y_i}^{\,i}, \Delta, E \vdash \overline{td_j}^{\,j} \rhd \langle E^{\mathrm{F}}, E^{\mathrm{X}} \rangle}{\overline{y_i}^{\,i}, \Delta, E \vdash x\; l\; \textit{tnvars}^l\; \overline{td_j}^{\,j} \rhd \langle E^{\mathrm{F}}, E^{\mathrm{X}} \rangle} \qquad \text{CHECK\_TEXPS\_CONS\_ABSTRACT}$$

$\boxed{\delta, E \vdash id \rightsquigarrow p}$ Lookup a type class

$$\frac{E(id) \rhd p}{\delta, E \vdash id \rightsquigarrow p} \qquad \text{CONVERT\_CLASS\_ALL}$$
$$\frac{\delta(p) \rhd xs}{}$$

$\boxed{I \vdash (p\; t)\; \textbf{IN}\; \mathcal{C}}$ Solve class constraint

$$\frac{}{I \vdash (p\; \alpha)\; \textbf{IN}\; (p_1\; \textit{tnv}_1)\,..\,(p_i\; \textit{tnv}_i)(p\; \alpha)(p_1'\; \textit{tnv}_1')\,..\,(p_j'\; \textit{tnv}_j')} \qquad \text{SOLVE\_CLASS\_CONSTRAINT\_IMMEDIATE}$$

$$(p_1\; \textit{tnv}_1)\,..\,(p_n\; \textit{tnv}_n) \Rightarrow (p\; t)\; \textbf{IN}\; I$$
$$I \vdash (p_1\; \sigma(\textit{tnv}_1))\; \textbf{IN}\; \mathcal{C} \quad .. \quad I \vdash (p_n\; \sigma(\textit{tnv}_n))\; \textbf{IN}\; \mathcal{C}$$
$$\rule{7cm}{0.4pt}$$
$$I \vdash (p\; \sigma(t))\; \textbf{IN}\; \mathcal{C} \qquad \text{SOLVE\_CLASS\_CONSTRAINT\_CHAIN}$$

$$\boxed{I \vdash \Sigma^{\mathcal{C}} \rhd \mathcal{C}} \quad \text{Solve class constraints}$$

$$\frac{I \vdash (p_1\, t_1)\, \mathbf{IN}\, \mathcal{C} \quad .. \quad I \vdash (p_n\, t_n)\, \mathbf{IN}\, \mathcal{C}}{I \vdash \{(p_1\, t_1), .., (p_n\, t_n)\} \rhd \mathcal{C}} \quad \textsc{solve\_class\_constraints\_all}$$

$$\boxed{\Delta, I, E \vdash val\_def \rhd E^{\mathrm{X}}} \quad \text{Check a value definition}$$

$$\frac{\begin{array}{l} \Delta, E, \{\,\} \vdash letbind \rhd \{\,\overline{x_i \mapsto t_i}^{\,i}\,\}, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}} \\ I \vdash \Sigma^{\mathcal{C}} \rhd \mathcal{C} \\ \overline{\mathbf{FV}\,(t_i)\, \subset\, tnvs}^{\,i} \\ \mathbf{FV}\,(\mathcal{C})\, \subset\, tnvs \end{array}}{\Delta, I, E_1 \vdash \mathbf{let}\, \tau^?\, letbind \rhd \{\,\overline{x_i \mapsto \langle \mathbf{forall}\, tnvs.\mathcal{C} \Rightarrow t_i, \mathbf{let}\rangle}^{\,i}\,\}} \quad \textsc{check\_val\_def\_val}$$

$$\frac{\begin{array}{l} \overline{\Delta, E, E^{\mathrm{L}} \vdash funcl_i \rhd \{x_i \mapsto t_i\}, \Sigma^{\mathcal{C}}_{\,i}, \Sigma^{\mathcal{N}}_{\,i}}^{\,i} \\ I \vdash \Sigma^{\mathcal{C}} \rhd \mathcal{C} \\ \overline{\mathbf{FV}\,(t_i)\, \subset\, tnvs}^{\,i} \\ \mathbf{FV}\,(\mathcal{C})\, \subset\, tnvs \\ \mathbf{compatible\, overlap}\,(\,\overline{x_i \mapsto t_i}^{\,i}\,) \\ E^{\mathrm{L}} = \{\,\overline{x_i \mapsto t_i}^{\,i}\,\} \end{array}}{\Delta, I, E \vdash \mathbf{let\, rec}\, \tau^?\, \overline{funcl_i}^{\,i} \rhd \{\,\overline{x_i \mapsto \langle \mathbf{forall}\, tnvs.\mathcal{C} \Rightarrow t_i, \mathbf{let}\rangle}^{\,i}\,\}} \quad \textsc{check\_val\_def\_recfun}$$

$$\boxed{\Delta, (\alpha_1, .., \alpha_n) \vdash t\, \mathbf{instance}} \quad \text{Check that } t \text{ be a typeclass instance}$$

$$\frac{}{\Delta, (\alpha) \vdash \alpha\, \mathbf{instance}} \quad \textsc{check\_t\_instance\_var}$$

$$\frac{}{\Delta, (\alpha_1, ..., \alpha_n) \vdash \alpha_1 * .... * \alpha_n\, \mathbf{instance}} \quad \textsc{check\_t\_instance\_tup}$$

$$\frac{}{\Delta, (\alpha_1, \alpha_2) \vdash \alpha_1 \to \alpha_n\, \mathbf{instance}} \quad \textsc{check\_t\_instance\_fn}$$

$$\frac{\Delta(p) \rhd \alpha'_1 .. \alpha'_n}{\Delta, (\alpha_1, .., \alpha_n) \vdash p\, \alpha_1 .. \alpha_n\, \mathbf{instance}} \quad \textsc{check\_t\_instance\_tc}$$

$$\boxed{\overline{z_j}^{\,j}, D_1, E_1 \vdash def \rhd D_2, E_2} \quad \text{Check a definition}$$

$$\frac{\begin{array}{l} \overline{z_j}^{\,j}, \Delta_1, E \vdash \mathbf{tc}\, \overline{td_i}^{\,i} \rhd \Delta_2, E^{\mathrm{P}} \\ \overline{z_j}^{\,j}, \Delta_1 \uplus \Delta_2, E \uplus \langle \{\,\}, E^{\mathrm{P}}, \{\,\}, \{\,\}\rangle \vdash \overline{td_i}^{\,i} \rhd \langle E^{\mathrm{F}}, E^{\mathrm{X}}\rangle \end{array}}{\overline{z_j}^{\,j}, \langle \Delta_1, \delta, I\rangle, E \vdash \mathbf{type}\, \overline{td_i}^{\,i}\, l \rhd \langle \Delta_2, \{\,\}, \{\,\}\rangle, \langle \{\,\}, E^{\mathrm{P}}, E^{\mathrm{F}}, E^{\mathrm{X}}\rangle} \quad \textsc{check\_def\_type}$$

$$\frac{\Delta, I, E \vdash val\_def \rhd E^{\mathrm{X}}}{\overline{z_j}^{\,j}, \langle \Delta, \delta, I\rangle, E \vdash val\_def\, l \rhd \epsilon, \langle \{\,\}, \{\,\}, \{\,\}, E^{\mathrm{X}}\rangle} \quad \textsc{check\_def\_val\_def}$$

$$\begin{array}{l} \overline{\Delta, E_1, E^{\mathrm{L}} \vdash rule_i \rhd \{x_i \mapsto t_i\}, \Sigma^{\mathcal{C}}_{\,i}, \Sigma^{\mathcal{N}}_{\,i}}^{\,i} \\ I \vdash \overline{\Sigma^{\mathcal{C}}_{\,i}}^{\,i} \rhd \mathcal{C} \\ \overline{\mathbf{FV}\,(t_i)\, \subset\, tnvs}^{\,i} \\ \mathbf{FV}\,(\mathcal{C})\, \subset\, tnvs \\ \mathbf{compatible\, overlap}\,(\,\overline{x_i \mapsto t_i}^{\,i}\,) \\ E^{\mathrm{L}} = \{\,\overline{x_i \mapsto t_i}^{\,i}\,\} \\ E_2 = \langle \{\,\}, \{\,\}, \{\,\}, \{\,\overline{x_i \mapsto \langle \mathbf{forall}\, tnvs.\mathcal{C} \Rightarrow t_i, \mathbf{let}\rangle}^{\,i}\,\}\rangle \end{array}$$

<<no parses (char 59): </zj//j/>,<TD,TC,I>,E1 |- indreln targets_opt indreln_names*** </rulei

$$\frac{\overline{z_j}^j\,x, D_1, E_1 \vdash \mathit{defs} \rhd D_2, E_2}{\overline{z_j}^j\,, D_1, E_1 \vdash \mathbf{module}\,x\,l_1 = \mathbf{struct}\,\mathit{defs}\,\mathbf{end}\,l_2 \rhd D_2, \langle\{x \mapsto E_2\}, \{\,\}, \{\,\}, \{\,\}\rangle} \quad \text{CHECK\_DEF\_MODULE}$$

$$\frac{E_1(id) \rhd E_2}{\overline{z_j}^j\,, D, E_1 \vdash \mathbf{module}\,x\,l_1 = id\,l_2 \rhd \epsilon, \langle\{x \mapsto E_2\}, \{\,\}, \{\,\}, \{\,\}\rangle} \quad \text{CHECK\_DEF\_MODULE\_RENAME}$$

$$\frac{\begin{array}{l} \Delta, E \vdash \mathit{typ} \rightsquigarrow t \\ \mathbf{FV}\,(t) \subset \overline{\alpha_i}^i \\ \mathbf{FV}\,(\overline{\alpha_k'}^k) \subset \overline{\alpha_i}^i \\ \overline{\delta, E \vdash id_k \rightsquigarrow p_k}^k \\ E' = \langle\{\,\}, \{\,\}, \{\,\}, \{x \mapsto \langle\mathbf{forall}\,\overline{\alpha_i}^i.\overline{(p_k\,\alpha_k')}^k \Rightarrow t, \mathbf{val}\rangle\}\rangle \end{array}}{\overline{z_j}^j\,, \langle\Delta, \delta, I\rangle, E \vdash \mathbf{val}\,x\,l_1 : \mathbf{forall}\,\overline{\alpha_i\,l_i''}^i.\overline{id_k\,\alpha_k'\,l_k'}^k \Rightarrow \mathit{typ}\,l_2 \rhd \epsilon, E'} \quad \text{CHECK\_DEF\_SPEC}$$

$$\frac{\begin{array}{l} \overline{\Delta, E_1 \vdash \mathit{typ}_i \rightsquigarrow t_i}^i \\ \overline{\mathbf{FV}\,(t_i) \subset \alpha}^i \\ p = \overline{z_j}^j\,.\,x \\ E_2 = \langle\{\,\}, \{x \mapsto p\}, \{\,\}, \{\overline{y_i \mapsto \langle\mathbf{forall}\,\alpha.(p\,\alpha) \Rightarrow t_i, \mathbf{method}\rangle}^i\}\rangle \\ \delta_2 = \{p \mapsto \overline{y_i}^i\} \\ p \notin \mathbf{dom}\,(\delta_1) \end{array}}{\overline{z_j}^j\,, \langle\Delta, \delta_1, I\rangle, E_1 \vdash \mathbf{class}\,(x\,l\,\alpha\,l'')\,\overline{\mathbf{val}\,y_i\,l_i : \mathit{typ}_i\,l_i}^i\,\mathbf{end}\,l' \rhd \langle\{\,\}, \delta_2, \{\,\}\rangle, E_2} \quad \text{CHECK\_DEF\_CLASS}$$

$$\frac{\begin{array}{l} E = \langle E^{\mathrm{M}}, E^{\mathrm{P}}, E^{\mathrm{F}}, E^{\mathrm{X}}\rangle \\ \Delta, E \vdash \mathit{typ}' \rightsquigarrow t' \\ \Delta, (\overline{\alpha_i}^i) \vdash t'\,\mathbf{instance} \\ \mathit{tnvs} = \overline{\alpha_i}^i \\ \mathbf{duplicates}\,(\mathit{tnvs}) = \emptyset \\ \overline{\delta, E \vdash id_k \rightsquigarrow p_k}^k \\ \mathbf{FV}\,(\overline{\alpha_k'}^k) \subset \mathit{tnvs} \\ E(id) \rhd p \\ \delta(p) \rhd \overline{z_j}^j \\ I_2 = \{\overline{\Rightarrow (p_k\,\alpha_k')}^k\} \\ \overline{\Delta, I \cup I_2, E \vdash \mathit{val\_def}_n \rhd E_n^{\mathrm{X}}}^n \\ \mathbf{disjoint\,doms}\,(\overline{E_n^{\mathrm{X}}}^n) \\ \overline{E^{\mathrm{X}}(x_k) \rhd \langle\mathbf{forall}\,\alpha''.(p\,\alpha'') \Rightarrow t_k, \mathbf{method}\rangle}^k \\ \{\overline{x_k \mapsto \langle\mathbf{forall}\,\mathit{tnvs}.\Rightarrow \{\alpha'' \mapsto t'\}(t_k), \mathbf{let}\rangle}^k\} = \overline{E_n^{\mathrm{X}}}^n \\ \overline{x_k}^k = \overline{z_j}^j \\ I_3 = \{\overline{(p_k\,\alpha_k') \Rightarrow (p\,t')}^k\} \\ (p\,\{\overline{\alpha_i \mapsto \alpha_i'''}^i\}(t')) \notin I \end{array}}{\overline{z_j}^j\,, \langle\Delta, \delta, I\rangle, E \vdash \mathbf{instance\,forall}\,\overline{\alpha_i\,l_i'}^i.\overline{id_k\,\alpha_k'\,l_k''}^k \Rightarrow (id\,\mathit{typ}')\,\overline{\mathit{val\_def}_n\,l_n}^n\,\mathbf{end}\,l' \rhd \langle\{\,\}, \{\,\}, I_3\rangle, \epsilon} \quad \text{CHECK\_DEF\_}$$

$$\boxed{\overline{z_j}^j\,, D_1, E_1 \vdash \mathit{defs} \rhd D_2, E_2} \quad \text{Check definitions, given module path, definitions and environment}$$

$$\frac{}{\overline{z_j}^j\,, D, E \vdash \rhd \epsilon, \epsilon} \quad \text{CHECK\_DEFS\_EMPTY}$$

$$\frac{\begin{array}{l} \overline{z_j}^j\,, D_1, E_1 \vdash \mathit{def} \rhd D_2, E_2 \\ \overline{z_j}^j\,, D_1 \uplus D_2, E_1 \uplus E_2 \vdash \overline{\mathit{def}_i \,;;_i^?}^i \rhd D_3, E_3 \end{array}}{\overline{z_j}^j\,, D_1, E_1 \vdash \mathit{def} \,;;^?\,\overline{\mathit{def}_i \,;;_i^?}^i \rhd D_2 \uplus D_3, E_2 \uplus E_3} \quad \text{CHECK\_DEFS\_RELEVANT\_DEF}$$

$$
\frac{
\begin{array}{c}
E_1(id) \vartriangleright E_2 \\
\overline{z_j}^{\,j}, D_1, E_1 \uplus E_2 \vdash \overline{def_i \,;\,;_i^?}^{\,i} \vartriangleright D_3, E_3
\end{array}
}{
\overline{z_j}^{\,j}, D_1, E_1 \vdash \mathbf{open}\ id\ l \,;\,;^? \overline{def_i \,;\,;_i^?}^{\,i} \vartriangleright D_3, E_3
}\ \ \text{CHECK\_DEFS\_OPEN}
$$

```
Definition rules:        144 good     1 bad
Definition rule clauses: 438 good     1 bad
```