n, i, j, k	Index variables for meta-lists
num	Numeric literals
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
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
	|	$x_1^l .. x_n^l . x^l\ l$	
tnv	::=		Union of type variables and Nexp type variables, without loca
	|	α	
	|	N	
$tnvar^l$	::=		Union of type variables and Nexp type variables, with location
	|	α^l	
	|	N^l	
$tnvs$	::=		Type variable lists
	|	$tnv_1 .. tnv_n$	
$tnvars^l$	::=		Type variable lists
	|	$tnvar_1^l .. tnvar_n^l$	
$Nexp\_aux$	::=		Numerical expressions for specifying vector lengths and indexe
	|	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 C
	|	*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
	$\langle\|exp$ **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
	$\{exp_1 \|$ **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
	$[exp_1 |$ **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

exp	::=		Location-annotated expressions
	|	*exp_aux l*	
pre_exp	::=		Pre-condition for persistent testing, setting up test genera
	|	*exp*	
	|	*test_spec*	
	|	$pre\_exp\ ix^l\ pre\_exp'$	
post_exp	::=		Post-condition, will be extended further if we want explic
	|	**result**	
	|	*exp*	
	|	*test_spec*	
	|	$post\_exp\ ix^l\ post\_exp'$	
test_exp	::=		Hoare-logic-based specification of a unit test
	|	$\{pre\_exp\}exp\{post\_exp\}$	exp is expected to be a pre-fix or infix call to one funct
	|	$\{pre\_exp\}exp$	exp is expected to be a relation, but could be an ordina
tbind	::=		
	|	x^l	Introduce a name, type to be inferred. Variable to be u
	|	$x^l : typ$	
	|	$x^l\{= exp\}$	Introduce a name and default value to use for unit test
tbinds	::=		
	|	*tbind*	
	|	*tbind*, *tbinds*	
test_spec_aux	::=		Quantified specification of a unit test
	|	**forall** *tbinds*.*test_exp*	
	|	*test_exp*	
test_spec	::=		
	|	*test_spec_aux l*	
q	::=		Quantifiers
	|	**forall**	
	|	**exists**	
qbind	::=		Bindings for quantifiers
	|	x^l	
	|	(*pat* **IN** *exp*)	Restricted quantifications over sets
	|	(*pat* **MEM** *exp*)	Restricted quantifications over lists
fexp	::=		Field-expressions
	|	*id* = *exp l*	
fexps	::=		Field-expression lists

	|	$fexp_1 ; \ldots ; fexp_n \,;^? \, l$	
$pexp$	::=		Pattern matches
	|	$pat \rightarrow exp \, l$	
$psexp$	::=		Multi-pattern matches
	|	$pat_1 \ldots pat_n \rightarrow exp \, l$	
$tannot^?$	::=		Optional type annotations
	|		
	|	$: typ$	
$funcl\_aux$	::=		Function clauses
	|	$x^l \; pat_1 \ldots pat_n \; tannot^? = exp$	
$letbind\_aux$	::=		Let bindings
	|	$pat \; tannot^? = exp$	Value bindings
	|	$funcl\_aux$	Function bindings
$letbind$	::=		Location-annotated let bindings
	|	$letbind\_aux \; l$	
$funcl$	::=		Location-annotated function clauses
	|	$funcl\_aux \; l$	
$id^?$	::=		Optional name for inductively defined relati
	|		
	|	$x^l :$	
$rule\_aux$	::=		Inductively defined relation clauses
	|	$id^? \; \textbf{forall} \; x_1^l \, .. \, x_n^l . exp \Longrightarrow x^l \; exp_1 \, .. \, exp_i$	
$rule$	::=		Location-annotated inductively defined relat
	|	$rule\_aux \; l$	
$typs$	::=		Type lists
	|	$typ_1 * \ldots * typ_n$	
$ctor\_def$	::=		Datatype definition clauses
	|	$x^l \; \textbf{of} \; typs$	
	|	x^l	S Constant constructors
$texp$	::=		Type definition bodies
	|	typ	Type abbreviations
	|	$\langle \| x_1^l : typ_1 ; \ldots ; x_n^l : typ_n \,;^? \| \rangle$	Record types
	|	$\|^? \; ctor\_def_1 \| \ldots \| ctor\_def_n$	Variant types

$name^?$	::=		Optional name specification for va
	|		
	|	$[name = regexp]$	
td	::=		Type definitions
	|	$x^l\ tnvars^l\ name^? = texp$	
	|	$x^l\ tnvars^l\ name^?$	Definitions of opaque types
c	::=		Typeclass constraints
	|	$id\ tnvar^l$	
cs	::=		Typeclass and length constraint li
	|		
	|	$c_1, .., c_i \Rightarrow$	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 prefixes
	|		
	|	$\mathbf{forall}\ tnvar^l_1 .. tnvar^l_n.cs$	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*)	
test_specs	::=		
	|	$test\_spec_0$ ... $test\_spec_1$	
forall_tests	::=		Forall and preconds that hold f
	|	**forall** *tbinds*.*pre_exp*	
	|	**forall** *tbinds*.*pat*	
	|		
test_decl	::=		Test declaration
	|	**test** x^l **begin** *forall_tests test_specs* **end**	The name identifies the decl
val_def	::=		Value definitions
	|	**let** $\tau^?$ *letbind*	Non-recursive value definitio
	|	**let rec** $\tau^?$ $funcl_1$ **and** ... **and** $funcl_n$	Recursive function definitions
	|	**let inline** $\tau^?$ *letbind*	Function definitions to be inl
val_spec	::=		Value type specifications
	|	**val** x^l : *typschm*	
def_aux	::=		Top-level definitions
	|	**type** td_1 **and** ... **and** td_n	Type definitions
	|	*val_def*	Value definitions
	|	*lemma_decl*	Lemmata
	|	*test_decl*	Test declarations
	|	**rename** $\tau^?$ $id = x^l$	Rename constant or type
	|	**module** x^l = **struct** *defs* **end**	Module definitions
	|	**module** $x^l = id$	Module renamings
	|	**open** *id*	Opening modules
	|	**indreln** $\tau^?$ $rule_1$ **and** ... **and** $rule_n$	Inductively defined relations
	|	*val_spec*	Top-level type constraints
	|	**class** (x^l $tnvar^l$) **val** x_1^l : typ_1 l_1 ... **val** x_n^l : typ_n l_n **end**	Typeclass definitions
	|	**instance** *instschm* val_def_1 l_1 ... val_def_n l_n **end**	Typeclass instantiations
def	::=		Location-annotated definitions
	|	*def_aux* *l*	
;;$^?$	::=		Optional double-semi-colon
	|		
	|	;;	
defs	::=		Definition sequences
	|	def_1 ;;$_1^?$ .. def_n ;;$_n^?$	

p	::=			Unique paths
	|	$x_1 .. x_n.x$		
	|	**__list**		
	|	**__bool**		
	|	**__num**		
	|	**__set**		
	|	**__string**		
	|	**__unit**		
	|	**__bit**		
	|	**__vector**		
σ	::=			Type variable substitutions
	|	$\{tnv_1 \mapsto t_1 .. tnv_n \mapsto t_n\}$		
$t,\ u$	::=			Internal types
	|	α		
	|	$t_1 \rightarrow t_2$		
	|	$t_1 * .... * t_n$		
	|	$p\ t\_args$		
	|	ne		
	|	$\sigma(t)$	M	Multiple substitutions
	|	$\sigma(tnv)$	M	Single variable substitution
	|	**curry** (t_multi, t)	M	Curried, multiple argument functions
ne	::=			internal numeric expressions
	|	N		
	|	num		
	|	$num * ne$		
	|	$ne_1 + ne_2$		
	|	$(-ne)$		
	|	**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 descripti
	|	**method**	Bound to a method
	|	**val**	Specified with val
	|	**let**	Defined with let or indreln
v_desc	::=		Value descriptions
	|	$\langle$ **forall** $tnvs.t\_multi \rightarrow p, (x$ **of** $names)\rangle$	Constructors
	|	$\langle$ **forall** $tnvs.\mathcal{C} \Rightarrow t, env\_tag\rangle$	Values
f_desc	::=		
	|	$\langle$ **forall** $tnvs.p \rightarrow t, (x$ **of** $names)\rangle$	Fields
$\Sigma^{\mathcal{C}}$	::=		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

	$\mid$	$\{x_1 \mapsto p_1, .., x_n \mapsto p_n\}$	
	$\mid$	$E_1^{\mathrm{P}} \uplus .. \uplus E_n^{\mathrm{P}}$	M
E^{L}	::=		Lexical bindings
	$\mid$	$\{x_1 \mapsto t_1, .., x_n \mapsto t_n\}$	
	$\mid$	$E_1^{\mathrm{L}} \uplus .. \uplus E_n^{\mathrm{L}}$	M
tc_abbrev	::=		Type abbreviations
	$\mid$	$.t$	
	$\mid$		
tc_def	::=		Type and class constructor definitions
	$\mid$	*tnvs tc_abbrev*	Type constructors
Δ	::=		Type constructor definitions
	$\mid$	$\{p_1 \mapsto tc\_def_1, .., p_n \mapsto tc\_def_n\}$	
	$\mid$	$\Delta_1 \uplus \Delta_2$	M
δ	::=		Typeclass definitions
	$\mid$	$\{p_1 \mapsto xs_1, .., p_n \mapsto xs_n\}$	
	$\mid$	$\delta_1 \uplus \delta_2$	M
inst	::=		A typeclass instance, t must not contain nested typ
	$\mid$	$\mathcal{C} \Rightarrow (p\ t)$	
I	::=		Global instances
	$\mid$	$\{inst_1, .., inst_n\}$	
	$\mid$	$I_1 \cup I_2$	M
D	::=		Global type definition store
	$\mid$	$\langle \Delta, \delta, I \rangle$	
	$\mid$	$D_1 \uplus D_2$	M
	$\mid$	ϵ	M
xs	::=		
	$\mid$	$x_1 .. x_n$	
terminals	::=		
	$\mid$	$\geq$	>=
	$\mid$	$\rightarrow$	->
	$\mid$	$\Longrightarrow$	==>
	$\mid$	$\langle\mid$	<|
	$\mid$	$\mid\rangle$	|>
	$\mid$	$\cap$	
	$\mid$	$\cup$	
	$\mid$	$\uplus$	
	$\mid$	$\notin$	

	$\subset$
	$\neq$
	$\emptyset$
	$\langle$
	$\rangle$
	$\vdash$
	,
	$\mapsto$
	$\triangleright$
	$\rightsquigarrow$
	$\Rightarrow$
	$\_$
	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
	$\mathbf{dom}\,(E_1^{\mathrm{M}}) \cap \mathbf{dom}\,(E_2^{\mathrm{M}}) = \emptyset$	
	$\mathbf{dom}\,(E_1^{\mathrm{X}}) \cap \mathbf{dom}\,(E_2^{\mathrm{X}}) = \emptyset$	
	$\mathbf{dom}\,(E_1^{\mathrm{F}}) \cap \mathbf{dom}\,(E_2^{\mathrm{F}}) = \emptyset$	
	$\mathbf{dom}\,(E_1^{\mathrm{P}}) \cap \mathbf{dom}\,(E_2^{\mathrm{P}}) = \emptyset$	
	$\mathbf{disjoint\,doms}\,(E_1^{\mathrm{L}}, \ldots., E_n^{\mathrm{L}})$	Pairwise disjoint domains
	$\mathbf{disjoint\,doms}\,(E_1^{\mathrm{X}}, \ldots., E_n^{\mathrm{X}})$	Pairwise disjoint domains
	$\mathbf{compatible\,overlap}\,(x_1 \mapsto t_1, .., x_n \mapsto t_n)$	$(x_i = x_j) \Longrightarrow (t_i = t_j)$
	$\mathbf{duplicates}\,(tnvs) = \emptyset$	
	$\mathbf{duplicates}\,(x_1, .., x_n) = \emptyset$	
	$x \notin \mathbf{dom}\,(E^{\mathrm{L}})$	
	$x \notin \mathbf{dom}\,(E^{\mathrm{X}})$	
	$x \notin \mathbf{dom}\,(E^{\mathrm{F}})$	
	$p \notin \mathbf{dom}\,(\delta)$	
	$p \notin \mathbf{dom}\,(\Delta)$	
	$\mathbf{FV}\,(t) \subset tnvs$	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 ::=		
	$E_1(x_1^l \,..\, x_n^l) \triangleright E_2$	Name path lookup
look_m_id ::=		
	$E_1(id) \triangleright E_2$	Module identifier lookup
look_tc ::=		
	$E(id) \triangleright p$	Path identifier lookup
check_t ::=		
	$\Delta \vdash t\,\mathbf{ok}$	Well-formed types
	$\Delta, tnv \vdash t\,\mathbf{ok}$	Well-formed type/Nexps m
teq ::=		
	$\Delta \vdash t_1 = t_2$	Type equality
convert_typ ::=		
	$\Delta, E \vdash typ \rightsquigarrow t$	Convert source types to in
	$\vdash Nexp \rightsquigarrow ne$	Convert and normalize nu
convert_typs ::=		
	$\Delta, E \vdash typs \rightsquigarrow t\_multi$	
check_lit ::=		
	$\vdash lit : t$	Typing literal constants
inst_field ::=		
	$\Delta, E \vdash \mathbf{field}\ id : p\ t\_args \rightarrow t \triangleright (x\,\mathbf{of}\ names)$	Field typing (also returns
inst_ctor ::=		
	$\Delta, E \vdash \mathbf{ctor}\ id : t\_multi \rightarrow p\ t\_args \triangleright (x\,\mathbf{of}\ names)$	Data constructor typing (a

inst_val	::=		
	|	$\Delta, E \vdash \mathbf{val}\ id : t \triangleright \Sigma^{\mathcal{C}}$	Typing top-level bindings, colle
not_ctor	::=		
	|	$E, E^{\mathrm{L}} \vdash x\ \mathbf{not\ ctor}$	v is not bound to a data const
not_shadowed	::=		
	|	$E^{\mathrm{L}} \vdash id\ \mathbf{not\ shadowed}$	id is not lexically shadowed
check_pat	::=		
	|	$\Delta, E, E_1^{\mathrm{L}} \vdash pat : t \triangleright E_2^{\mathrm{L}}$	Typing patterns, building their
	|	$\Delta, E, E_1^{\mathrm{L}} \vdash pat\_aux : t \triangleright E_2^{\mathrm{L}}$	Typing patterns, building their
id_field	::=		
	|	$E \vdash id\ \mathbf{field}$	Check that the identifier is a p
id_value	::=		
	|	$E \vdash id\ \mathbf{value}$	Check that the identifier is a p
check_exp	::=		
	|	$\Delta, E, E^{\mathrm{L}} \vdash exp : t \triangleright \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}$	Typing expressions, collecting
	|	$\Delta, E, E^{\mathrm{L}} \vdash exp\_aux : t \triangleright \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}$	Typing expressions, collecting
	|	$\Delta, E, E_1^{\mathrm{L}} \vdash qbind_1 \,..\, qbind_n \triangleright E_2^{\mathrm{L}}, \Sigma^{\mathcal{C}}$	Build the environment for quar
	|	$\Delta, E, E_1^{\mathrm{L}} \vdash \mathbf{list}\ qbind_1 \,..\, qbind_n \triangleright E_2^{\mathrm{L}}, \Sigma^{\mathcal{C}}$	Build the environment for quar
	|	$\Delta, E, E^{\mathrm{L}} \vdash funcl \triangleright \{x \mapsto t\}, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}$	Build the environment for a fu
	|	$\Delta, E, E_1^{\mathrm{L}} \vdash letbind \triangleright E_2^{\mathrm{L}}, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}$	Build the environment for a let
check_rule	::=		
	|	$\Delta, E, E^{\mathrm{L}} \vdash rule \triangleright \{x \mapsto t\}, \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}$	Build the environment for an i
check_texp_tc	::=		
	|	$xs, \Delta_1, E \vdash \mathbf{tc}\ td \triangleright \Delta_2, E^{\mathrm{P}}$	Extract the type constructor in
check_texps_tc	::=		
	|	$xs, \Delta_1, E \vdash \mathbf{tc}\ td_1 \,..\, td_i \triangleright \Delta_2, E^{\mathrm{P}}$	Extract the type constructor in
check_texp	::=		
	|	$\Delta, E \vdash tnvs\ p = texp \triangleright \langle E^{\mathrm{F}}, E^{\mathrm{X}} \rangle$	Check a type definition, with i
check_texps	::=		
	|	$xs, \Delta, E \vdash td_1 \,..\, td_n \triangleright \langle E^{\mathrm{F}}, E^{\mathrm{X}} \rangle$	
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	::=		
	|	$\Delta, I, E \vdash \mathit{val\_def} \triangleright E^{\mathrm{X}}$	Check a value definition
check_t_instance	::=		
	|	$\Delta, (\alpha_1, .., \alpha_n) \vdash t\,\mathbf{instance}$	Check that t be a typeclass instance
check_defs	::=		
	|	$\overline{z_j}^{\,j}, D_1, E_1 \vdash \mathit{def} \triangleright D_2, E_2$	Check a definition
	|	$\overline{z_j}^{\,j}, D_1, E_1 \vdash \mathit{defs} \triangleright D_2, E_2$	Check definitions, given module path, defi
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*	

| 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
| pre_exp
| $post\_exp$
| $test\_exp$
| $tbind$
| $tbinds$
| $test\_spec\_aux$
| $test\_spec$
| q
| $qbind$
| $fexp$
| $fexps$
| $pexp$
| $psexp$

| $tannot^{?}$
| $funcl\_aux$
| $letbind\_aux$
| $letbind$
| $funcl$
| $id^{?}$
| $rule\_aux$
| $rule$
| $typs$
| $ctor\_def$
| $texp$
| $name^{?}$
| td
| c
| cs
| c_pre
| $typschm$
| $instschm$
| $target$
| τ
| $\tau^{?}$
| $lemma\_typ$
| $lemma\_decl$
| $test\_specs$
| $forall\_tests$
| $test\_decl$
| val_def
| val_spec
| def_aux
| def
| $;;^{?}$
| $defs$
| p
| σ
| t
| ne
| t_args
| t_multi
| nec
| $names$
| $\mathcal{C}$
| env_tag
| v_desc
| f_desc
| $\Sigma^{\mathcal{C}}$
| $\Sigma^{\mathcal{N}}$

| E
| E^{X}
| E^{F}
| E^{M}
| E^{P}
| E^{L}
| tc_abbrev
| tc_def
| Δ
| δ
| $inst$
| I
| D
| xs
| $terminals$
| $formula$

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

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

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

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

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

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

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

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

$\boxed{E_1(id) \triangleright E_2}$ 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

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

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

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

$$\frac{\Delta \vdash t_1\ \mathbf{ok} \quad \Delta \vdash t_2\ \mathbf{ok}}{\Delta \vdash t_1 \to t_2\ \mathbf{ok}}\ \textsc{check\_t\_fn}$$

$$\frac{\Delta \vdash t_1\ \mathbf{ok} \quad .... \quad \Delta \vdash t_n\ \mathbf{ok}}{\Delta \vdash t_1 * .... * t_n\ \mathbf{ok}}\ \textsc{check\_t\_tup}$$

$$\frac{\Delta(p) \triangleright tnv_1 .. tnv_n\ tc\_abbrev \quad \Delta, tnv_1 \vdash t_1\ \mathbf{ok} \quad .. \quad \Delta, tnv_n \vdash t_n\ \mathbf{ok}}{\Delta \vdash p\ t_1 .. t_n\ \mathbf{ok}}\ \textsc{check\_t\_app}$$

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

$$\frac{\Delta \vdash t\ \mathbf{ok}}{\Delta, \alpha \vdash t\ \mathbf{ok}}\ \textsc{check\_tlen\_t}$$

$$\frac{}{\Delta, N \vdash ne\ \mathbf{ok}}\ \textsc{check\_tlen\_len}$$

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

$$\frac{\Delta \vdash t\ \mathbf{ok}}{\Delta \vdash t = t}\ \textsc{teq\_refl}$$

$$\frac{\Delta \vdash t_2 = t_1}{\Delta \vdash t_1 = t_2}\ \textsc{teq\_sym}$$

$$\frac{\Delta \vdash t_1 = t_2 \quad \Delta \vdash t_2 = t_3}{\Delta \vdash t_1 = t_3}\ \textsc{teq\_trans}$$

$$\frac{\Delta \vdash t_1 = t_3 \quad \Delta \vdash t_2 = t_4}{\Delta \vdash t_1 \to t_2 = t_3 \to t_4}\ \textsc{teq\_arrow}$$

$$\frac{\Delta \vdash t_1 = u_1 \quad .... \quad \Delta \vdash t_n = u_n}{\Delta \vdash t_1 * .... * t_n = u_1 * .... * u_n}\ \textsc{teq\_tup}$$

$$\frac{\Delta(p) \triangleright \alpha_1 .. \alpha_n \quad \Delta \vdash t_1 = u_1 \quad .. \quad \Delta \vdash t_n = u_n}{\Delta \vdash p\ t_1 .. t_n = p\ u_1 .. u_n}\ \textsc{teq\_app}$$

$$\frac{\Delta(p) \triangleright \alpha_1 .. \alpha_n\ .u}{\Delta \vdash p\ t_1 .. t_n = \{\alpha_1 \mapsto t_1 .. \alpha_n \mapsto t_n\}(u)}\ \textsc{teq\_expand}$$

$$\frac{ne = \mathbf{normalize}\,(ne')}{\Delta \vdash ne = ne'}\ \textsc{teq\_Nexp}$$

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

$$\frac{}{\Delta, E \vdash \alpha\ l'\ l \rightsquigarrow \alpha}\ \textsc{convert\_typ\_var}$$

$$\frac{\Delta, E \vdash typ_1 \rightsquigarrow t_1 \quad \Delta, E \vdash typ_2 \rightsquigarrow t_2}{\Delta, E \vdash typ_1 \to typ_2\ l \rightsquigarrow t_1 \to t_2}\ \textsc{convert\_typ\_fn}$$

$$\frac{\Delta, E \vdash typ_1 \rightsquigarrow t_1 \quad .... \quad \Delta, E \vdash typ_n \rightsquigarrow t_n}{\Delta, E \vdash typ_1 * .... * typ_n\ l \rightsquigarrow t_1 * .... * t_n}\ \textsc{convert\_typ\_tup}$$

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

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

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

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

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

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

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

$$\frac{}{\vdash num * N\ \rightsquigarrow num * N}\ \text{CONVERT\_NEXP\_MULT}$$

$$\frac{\vdash Nexp_1 \rightsquigarrow ne_1 \quad \vdash Nexp_2 \rightsquigarrow ne_2}{\vdash Nexp_1 + Nexp_2\ \rightsquigarrow ne_1 + ne_2}\ \text{CONVERT\_NEXP\_ADD}$$

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

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

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

$$\frac{}{\vdash \mathbf{true}\ l : \_\_\mathbf{bool}}\ \text{CHECK\_LIT\_TRUE}$$

$$\frac{}{\vdash \mathbf{false}\ l : \_\_\mathbf{bool}}\ \text{CHECK\_LIT\_FALSE}$$

$$\frac{}{\vdash num\ l : \_\_\mathbf{num}}\ \text{CHECK\_LIT\_NUM}$$

$$\frac{num = \mathbf{bitlength}\ (hex)}{\vdash hex\ l : \_\_\mathbf{vector}\ num\ \_\_\mathbf{bit}}\ \text{CHECK\_LIT\_HEX}$$

$$\frac{num = \mathbf{bitlength}\ (bin)}{\vdash bin\ l : \_\_\mathbf{vector}\ num\ \_\_\mathbf{bit}}\ \text{CHECK\_LIT\_BIN}$$

$$\frac{}{\vdash string\ l : \_\_\mathbf{string}}\ \text{CHECK\_LIT\_STRING}$$

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

$$\frac{}{\vdash \mathbf{bitzero}\ l : \_\_\mathbf{bit}}\ \text{CHECK\_LIT\_BITZERO}$$

$$\frac{}{\vdash \mathbf{bitone}\ l : \_\_\mathbf{bit}}\ \text{CHECK\_LIT\_BITONE}$$

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

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

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

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

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

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

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

$$\frac{E^{\mathrm{L}}(x) \triangleright t}{E, E^{\mathrm{L}} \vdash x\ \mathbf{not\ ctor}}\ \textsc{not\_ctor\_val}$$

$$\frac{x \notin \mathbf{dom}\,(E^{\mathrm{X}})}{\langle E^{\mathrm{M}}, E^{\mathrm{P}}, E^{\mathrm{F}}, E^{\mathrm{X}} \rangle, E^{\mathrm{L}} \vdash x\ \mathbf{not\ ctor}}\ \textsc{not\_ctor\_unbound}$$

$$\frac{E^{\mathrm{X}}(x) \triangleright \langle \mathbf{forall}\ tnv_1\ ..\ tnv_n.(p_1\ tnv_1')\ ..\ (p_i\ tnv_i') \Rightarrow t, env\_tag \rangle}{\langle E^{\mathrm{M}}, E^{\mathrm{P}}, E^{\mathrm{F}}, E^{\mathrm{X}} \rangle, E^{\mathrm{L}} \vdash x\ \mathbf{not\ ctor}}\ \textsc{not\_ctor\_bound}$$

$\boxed{E^{\mathrm{L}} \vdash id\ \mathbf{not\ shadowed}}$ id is not lexically shadowed

$$\frac{x \notin \mathbf{dom}\,(E^{\mathrm{L}})}{E^{\mathrm{L}} \vdash x\ l_1\ l_2\ \mathbf{not\ shadowed}}\ \textsc{not\_shadowed\_sing}$$

$$\frac{}{E^{\mathrm{L}} \vdash x_1^l\ ..\ x_n^l.y^l.z^l\ l\ \mathbf{not\ shadowed}}\ \textsc{not\_shadowed\_multi}$$

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

$$\frac{\Delta, E, E_1^{\mathrm{L}} \vdash pat\_aux : t \triangleright E_2^{\mathrm{L}}}{\Delta, E, E_1^{\mathrm{L}} \vdash pat\_aux\ l : t \triangleright E_2^{\mathrm{L}}}\ \textsc{check\_pat\_all}$$

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

$$\frac{\Delta \vdash t\ \mathbf{ok}}{\Delta, E, E^{\mathrm{L}} \vdash \_ : t \triangleright \{\}}\ \textsc{check\_pat\_aux\_wild}$$

$$\frac{\begin{array}{l} \Delta, E, E_1^{\mathrm{L}} \vdash pat : t \triangleright E_2^{\mathrm{L}} \\ x \notin \mathbf{dom}\,(E_2^{\mathrm{L}}) \end{array}}{\Delta, E, E_1^{\mathrm{L}} \vdash (pat\ \mathbf{as}\ x\ l) : t \triangleright E_2^{\mathrm{L}} \uplus \{x \mapsto t\}}\ \textsc{check\_pat\_aux\_as}$$

$$\frac{\Delta, E, E_1^{\text{L}} \vdash pat : t \triangleright E_2^{\text{L}} \quad \Delta, E \vdash typ \rightsquigarrow t}{\Delta, E, E_1^{\text{L}} \vdash (pat : typ) : t \triangleright E_2^{\text{L}}}\ \text{CHECK\_PAT\_AUX\_TYP}$$

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

$$\frac{\begin{array}{l} \Delta, E, E^{\mathrm{L}} \vdash exp_1 : t_1 \to t_2 \triangleright \Sigma^{\mathcal{C}}{}_1, \Sigma^{\mathcal{N}}{}_1 \\ \Delta, E, E^{\mathrm{L}} \vdash exp_2 : t_1 \triangleright \Sigma^{\mathcal{C}}{}_2, \Sigma^{\mathcal{N}}{}_2 \end{array}}{\Delta, E, E^{\mathrm{L}} \vdash exp_1\ exp_2 : t_2 \triangleright \Sigma^{\mathcal{C}}{}_1 \cup \Sigma^{\mathcal{C}}{}_2, \Sigma^{\mathcal{N}}{}_1 \cup \Sigma^{\mathcal{N}}{}_2}\ \text{CHECK\_EXP\_AUX\_APP}$$

$$\frac{\begin{array}{l}\Delta, E, E^{\mathrm{L}} \vdash (ix) : t_1 \to t_2 \to t_3 \triangleright {\Sigma^{\mathcal{C}}}_1, {\Sigma^{\mathcal{N}}}_1 \\ \Delta, E, E^{\mathrm{L}} \vdash exp_1 : t_1 \triangleright {\Sigma^{\mathcal{C}}}_2, {\Sigma^{\mathcal{N}}}_2 \\ \Delta, E, E^{\mathrm{L}} \vdash exp_2 : t_2 \triangleright {\Sigma^{\mathcal{C}}}_3, {\Sigma^{\mathcal{N}}}_3\end{array}}{\Delta, E, E^{\mathrm{L}} \vdash exp_1\ ix\ l\ exp_2 : t_3 \triangleright {\Sigma^{\mathcal{C}}}_1 \cup {\Sigma^{\mathcal{C}}}_2 \cup {\Sigma^{\mathcal{C}}}_3, {\Sigma^{\mathcal{N}}}_1 \cup {\Sigma^{\mathcal{N}}}_2 \cup {\Sigma^{\mathcal{N}}}_3}\ \text{CHECK\_EXP\_AUX\_INFIX\_APP1}$$

$$\frac{\begin{array}{l}\Delta, E, E^{\mathrm{L}} \vdash\ x\ : t_1 \to t_2 \to t_3 \triangleright {\Sigma^{\mathcal{C}}}_1, {\Sigma^{\mathcal{N}}}_1 \\ \Delta, E, E^{\mathrm{L}} \vdash exp_1 : t_1 \triangleright {\Sigma^{\mathcal{C}}}_2, {\Sigma^{\mathcal{N}}}_2 \\ \Delta, E, E^{\mathrm{L}} \vdash exp_2 : t_2 \triangleright {\Sigma^{\mathcal{C}}}_3, {\Sigma^{\mathcal{N}}}_3\end{array}}{\Delta, E, E^{\mathrm{L}} \vdash exp_1\ `x`l\ exp_2 : t_3 \triangleright {\Sigma^{\mathcal{C}}}_1 \cup {\Sigma^{\mathcal{C}}}_2 \cup {\Sigma^{\mathcal{C}}}_3, {\Sigma^{\mathcal{N}}}_1 \cup {\Sigma^{\mathcal{N}}}_2 \cup {\Sigma^{\mathcal{N}}}_3}\ \text{CHECK\_EXP\_AUX\_INFIX\_APP2}$$

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

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

$$\frac{\begin{array}{l}\Delta, E, E^{\mathrm{L}} \vdash exp_1 : t \triangleright {\Sigma^{\mathcal{C}}}_1, {\Sigma^{\mathcal{N}}}_1 \quad ... \quad \Delta, E, E^{\mathrm{L}} \vdash exp_n : t \triangleright {\Sigma^{\mathcal{C}}}_n, {\Sigma^{\mathcal{N}}}_n \\ \mathbf{length}\,(exp_1\ ...\ exp_n) = num\end{array}}{\Delta, E, E^{\mathrm{L}} \vdash [|exp_1; \ ...\ ; exp_n\ |] : \_\_\mathbf{vector}\ num\ t \triangleright {\Sigma^{\mathcal{C}}}_1 \cup ... \cup {\Sigma^{\mathcal{C}}}_n, {\Sigma^{\mathcal{N}}}_1 \cup ... \cup {\Sigma^{\mathcal{N}}}_n}\ \text{CHECK\_EXP\_AUX\_VECTOR}$$

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

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

$$\frac{\begin{array}{l}E \vdash id\ \mathbf{field} \\ \Delta, E \vdash \mathbf{field}\ id : p\ t\_args \to t \triangleright (x\ \mathbf{of}\ names) \\ \Delta, E, E^{\mathrm{L}} \vdash exp : p\ t\_args \triangleright {\Sigma^{\mathcal{C}}}, {\Sigma^{\mathcal{N}}}\end{array}}{\Delta, E, E^{\mathrm{L}} \vdash exp.id : t \triangleright {\Sigma^{\mathcal{C}}}, {\Sigma^{\mathcal{N}}}}\ \text{CHECK\_EXP\_AUX\_FIELD}$$

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

$$\frac{\begin{array}{l}\Delta, E, E^{\mathrm{L}} \vdash exp : t \triangleright {\Sigma^{\mathcal{C}}}, {\Sigma^{\mathcal{N}}} \\ \Delta, E \vdash typ \rightsquigarrow t\end{array}}{\Delta, E, E^{\mathrm{L}} \vdash (exp : typ) : t \triangleright {\Sigma^{\mathcal{C}}}, {\Sigma^{\mathcal{N}}}}\ \text{CHECK\_EXP\_AUX\_TYPED}$$

$$\frac{\begin{array}{l}\Delta, E, E_1^{\mathrm{L}} \vdash letbind \triangleright E_2^{\mathrm{L}}, {\Sigma^{\mathcal{C}}}_1, {\Sigma^{\mathcal{N}}}_1 \\ \Delta, E, E_1^{\mathrm{L}} \uplus E_2^{\mathrm{L}} \vdash exp : t \triangleright {\Sigma^{\mathcal{C}}}_2, {\Sigma^{\mathcal{N}}}_2\end{array}}{\Delta, E, E^{\mathrm{L}} \vdash \mathbf{let}\ letbind\ \mathbf{in}\ exp : t \triangleright {\Sigma^{\mathcal{C}}}_1 \cup {\Sigma^{\mathcal{C}}}_2, {\Sigma^{\mathcal{N}}}_1 \cup {\Sigma^{\mathcal{N}}}_2}\ \text{CHECK\_EXP\_AUX\_LET}$$

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

$$\frac{\begin{array}{l}\Delta \vdash t\ \mathbf{ok} \\ \Delta, E, E^{\mathrm{L}} \vdash exp_1 : t \triangleright {\Sigma^{\mathcal{C}}}_1, {\Sigma^{\mathcal{N}}}_1 \quad .. \quad \Delta, E, E^{\mathrm{L}} \vdash exp_n : t \triangleright {\Sigma^{\mathcal{C}}}_n, {\Sigma^{\mathcal{N}}}_n\end{array}}{\Delta, E, E^{\mathrm{L}} \vdash [exp_1; \ldots; exp_n ;^?] : \_\_\mathbf{list}\ t \triangleright {\Sigma^{\mathcal{C}}}_1 \cup .. \cup {\Sigma^{\mathcal{C}}}_n, {\Sigma^{\mathcal{N}}}_1 \cup .. \cup {\Sigma^{\mathcal{N}}}_n} \text{ CHECK\_EXP\_AUX\_LIST}$$

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

$$\frac{\Delta, E, E^{\mathrm{L}} \vdash exp : t \triangleright \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}}{\Delta, E, E^{\mathrm{L}} \vdash \mathbf{begin}\ exp\ \mathbf{end} : t \triangleright \Sigma^{\mathcal{C}}, \Sigma^{\mathcal{N}}} \text{ CHECK\_EXP\_AUX\_BEGIN}$$

$$\frac{\begin{array}{l}\Delta, E, E^{\mathrm{L}} \vdash exp_1 : \_\_\mathbf{bool} \triangleright {\Sigma^{\mathcal{C}}}_1, {\Sigma^{\mathcal{N}}}_1 \\ \Delta, E, E^{\mathrm{L}} \vdash exp_2 : t \triangleright {\Sigma^{\mathcal{C}}}_2, {\Sigma^{\mathcal{N}}}_2 \\ \Delta, E, E^{\mathrm{L}} \vdash exp_3 : t \triangleright {\Sigma^{\mathcal{C}}}_3, {\Sigma^{\mathcal{N}}}_3\end{array}}{\Delta, E, E^{\mathrm{L}} \vdash \mathbf{if}\ exp_1\ \mathbf{then}\ exp_2\ \mathbf{else}\ exp_3 : t \triangleright {\Sigma^{\mathcal{C}}}_1 \cup {\Sigma^{\mathcal{C}}}_2 \cup {\Sigma^{\mathcal{C}}}_3, {\Sigma^{\mathcal{N}}}_1 \cup {\Sigma^{\mathcal{N}}}_2 \cup {\Sigma^{\mathcal{N}}}_3} \text{ CHECK\_EXP\_AUX\_IF}$$

$$\frac{\begin{array}{l}\Delta, E, E^{\mathrm{L}} \vdash exp_1 : t \triangleright {\Sigma^{\mathcal{C}}}_1, {\Sigma^{\mathcal{N}}}_1 \\ \Delta, E, E^{\mathrm{L}} \vdash exp_2 : \_\_\mathbf{list}\ t \triangleright {\Sigma^{\mathcal{C}}}_2, {\Sigma^{\mathcal{N}}}_2\end{array}}{\Delta, E, E^{\mathrm{L}} \vdash exp_1 :: exp_2 : \_\_\mathbf{list}\ t \triangleright {\Sigma^{\mathcal{C}}}_1 \cup {\Sigma^{\mathcal{C}}}_2, {\Sigma^{\mathcal{N}}}_1 \cup {\Sigma^{\mathcal{N}}}_2} \text{ CHECK\_EXP\_AUX\_CONS}$$

$$\frac{\vdash lit : t}{\Delta, E, E^{\mathrm{L}} \vdash lit : t \triangleright \{\,\}, \{\,\}} \text{ CHECK\_EXP\_AUX\_LIT}$$

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

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

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

$$\frac{\begin{array}{l}\Delta, E, E_1^{\mathrm{L}} \vdash \overline{qbind_i}^{\,i} \triangleright E_2^{\mathrm{L}}, {\Sigma^{\mathcal{C}}}_1 \\ \Delta, E, E_1^{\mathrm{L}} \uplus E_2^{\mathrm{L}} \vdash exp : \_\_\mathbf{bool} \triangleright {\Sigma^{\mathcal{C}}}_2, {\Sigma^{\mathcal{N}}}_2\end{array}}{\Delta, E, E_1^{\mathrm{L}} \vdash q\ \overline{qbind_i}^{\,i}.exp : \_\_\mathbf{bool} \triangleright {\Sigma^{\mathcal{C}}}_1 \cup {\Sigma^{\mathcal{C}}}_2, {\Sigma^{\mathcal{N}}}_2} \text{ CHECK\_EXP\_AUX\_QUANT}$$

$$\frac{\begin{array}{l}\Delta, E, E_1^{\mathrm{L}} \vdash \mathbf{list}\ \overline{qbind_i}^{\,i} \triangleright E_2^{\mathrm{L}}, {\Sigma^{\mathcal{C}}}_1 \\ \Delta, E, E_1^{\mathrm{L}} \uplus E_2^{\mathrm{L}} \vdash exp_1 : t \triangleright {\Sigma^{\mathcal{C}}}_2, {\Sigma^{\mathcal{N}}}_2 \\ \Delta, E, E_1^{\mathrm{L}} \uplus E_2^{\mathrm{L}} \vdash exp_2 : \_\_\mathbf{bool} \triangleright {\Sigma^{\mathcal{C}}}_3, {\Sigma^{\mathcal{N}}}_3\end{array}}{\Delta, E, E_1^{\mathrm{L}} \vdash [exp_1 |\, \mathbf{forall}\, \overline{qbind_i}^{\,i} \,| exp_2] : \_\_\mathbf{list}\ t \triangleright {\Sigma^{\mathcal{C}}}_1 \cup {\Sigma^{\mathcal{C}}}_2 \cup {\Sigma^{\mathcal{C}}}_3, {\Sigma^{\mathcal{N}}}_2 \cup {\Sigma^{\mathcal{N}}}_3} \text{ CHECK\_EXP\_AUX\_LIST\_COMP\_}$$

$\boxed{\Delta, E, E_1^{\mathrm{L}} \vdash qbind_1 .. qbind_n \triangleright E_2^{\mathrm{L}}, \Sigma^{\mathcal{C}}}$ Build the environment for quantifier bindings, collecting typeclass cons

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

$$\frac{\begin{array}{l}\Delta \vdash t\ \mathbf{ok} \\ \Delta, E, E_1^{\mathrm{L}} \uplus \{x \mapsto t\} \vdash \overline{qbind_i}^{\,i} \triangleright E_2^{\mathrm{L}}, \Sigma^{\mathcal{C}}{}_1 \\ \mathbf{disjoint\ doms}\,(\{x \mapsto t\}, E_2^{\mathrm{L}})\end{array}}{\Delta, E, E_1^{\mathrm{L}} \vdash x\ l\ \overline{qbind_i}^{\,i} \triangleright \{x \mapsto t\} \uplus E_2^{\mathrm{L}}, \Sigma^{\mathcal{C}}{}_1}\ \text{CHECK\_LISTQUANT\_BINDING\_VAR}$$

$$\frac{\begin{array}{l}\Delta, E, E_1^{\mathrm{L}} \vdash pat : t \triangleright E_3^{\mathrm{L}} \\ \Delta, E, E_1^{\mathrm{L}} \vdash exp : \_\_\mathbf{set}\ t \triangleright \Sigma^{\mathcal{C}}{}_1, \Sigma^{\mathcal{N}}{}_1 \\ \Delta, E, E_1^{\mathrm{L}} \uplus E_3^{\mathrm{L}} \vdash \overline{qbind_i}^{\,i} \triangleright E_2^{\mathrm{L}}, \Sigma^{\mathcal{C}}{}_2 \\ \mathbf{disjoint\ doms}\,(E_3^{\mathrm{L}}, E_2^{\mathrm{L}})\end{array}}{\Delta, E, E_1^{\mathrm{L}} \vdash (pat\ \mathbf{IN}\ exp)\ \overline{qbind_i}^{\,i} \triangleright E_2^{\mathrm{L}} \uplus E_3^{\mathrm{L}}, \Sigma^{\mathcal{C}}{}_1 \cup \Sigma^{\mathcal{C}}{}_2}\ \text{CHECK\_LISTQUANT\_BINDING\_RESTR}$$

$$\frac{\begin{array}{l}\Delta, E, E_1^{\mathrm{L}} \vdash pat : t \triangleright E_3^{\mathrm{L}} \\ \Delta, E, E_1^{\mathrm{L}} \vdash exp : \_\_\mathbf{list}\ t \triangleright \Sigma^{\mathcal{C}}{}_1, \Sigma^{\mathcal{N}}{}_1 \\ \Delta, E, E_1^{\mathrm{L}} \uplus E_3^{\mathrm{L}} \vdash \overline{qbind_i}^{\,i} \triangleright E_2^{\mathrm{L}}, \Sigma^{\mathcal{C}}{}_2 \\ \mathbf{disjoint\ doms}\,(E_3^{\mathrm{L}}, E_2^{\mathrm{L}})\end{array}}{\Delta, E, E_1^{\mathrm{L}} \vdash (pat\ \mathbf{MEM}\ exp)\ \overline{qbind_i}^{\,i} \triangleright E_2^{\mathrm{L}} \uplus E_3^{\mathrm{L}}, \Sigma^{\mathcal{C}}{}_1 \cup \Sigma^{\mathcal{C}}{}_2}\ \text{CHECK\_LISTQUANT\_BINDING\_LIST\_RESTR}$$

$\boxed{\Delta, E, E_1^{\mathrm{L}} \vdash \mathbf{list}\ qbind_1\ ..\ qbind_n \triangleright E_2^{\mathrm{L}}, \Sigma^{\mathcal{C}}}$ Build the environment for quantifier bindings, collecting typeclass

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

$$\frac{\begin{array}{l}\Delta, E, E_1^{\mathrm{L}} \vdash pat : t \triangleright E_3^{\mathrm{L}} \\ \Delta, E, E_1^{\mathrm{L}} \vdash exp : \_\_\mathbf{list}\ t \triangleright \Sigma^{\mathcal{C}}{}_1, \Sigma^{\mathcal{N}}{}_1 \\ \Delta, E, E_1^{\mathrm{L}} \uplus E_3^{\mathrm{L}} \vdash \overline{qbind_i}^{\,i} \triangleright E_2^{\mathrm{L}}, \Sigma^{\mathcal{C}}{}_2 \\ \mathbf{disjoint\ doms}\,(E_3^{\mathrm{L}}, E_2^{\mathrm{L}})\end{array}}{\Delta, E, E_1^{\mathrm{L}} \vdash \mathbf{list}\,(pat\ \mathbf{MEM}\ exp)\ \overline{qbind_i}^{\,i} \triangleright E_2^{\mathrm{L}} \uplus E_3^{\mathrm{L}}, \Sigma^{\mathcal{C}}{}_1 \cup \Sigma^{\mathcal{C}}{}_2}\ \text{CHECK\_QUANT\_BINDING\_RESTR}$$

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

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

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

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

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

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

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

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

$$\frac{\begin{array}{l}\overline{\Delta \vdash t_i\ \mathbf{ok}}^{\,i} \\ E_2^{\mathrm{L}} = \{\, \overline{y_i \mapsto t_i}^{\,i} \,\} \\ \Delta, E, E_1^{\mathrm{L}} \uplus E_2^{\mathrm{L}} \vdash \mathit{exp}' : \_\_\mathbf{bool} \triangleright \Sigma^{\mathcal{C}'}, \Sigma^{\mathcal{N}'} \\ \Delta, E, E_1^{\mathrm{L}} \uplus E_2^{\mathrm{L}} \vdash \mathit{exp}_1 : u_1 \triangleright \Sigma^{\mathcal{C}}{}_1, \Sigma^{\mathcal{N}}{}_1 \quad .. \quad \Delta, E, E_1^{\mathrm{L}} \uplus E_2^{\mathrm{L}} \vdash \mathit{exp}_n : u_n \triangleright \Sigma^{\mathcal{C}}{}_n, \Sigma^{\mathcal{N}}{}_n\end{array}}{\Delta, E, E_1^{\mathrm{L}} \vdash \mathit{id}^{?}\ \mathbf{forall}\ \overline{y_i\ l_i}^{\,i}\ .\mathit{exp}' \Longrightarrow x\ l\ \mathit{exp}_1\ ..\ \mathit{exp}_n\ l' \triangleright \{x \mapsto \mathbf{curry}\,((u_1 * .. * u_n), \_\_\mathbf{bool}\,)\}, \Sigma^{\mathcal{C}'} \cup \Sigma^{\mathcal{C}}{}_1 \cup .. \cup \Sigma}$$

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

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

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

$$\frac{\begin{array}{l}\mathit{tnvars}^{\,l} \rightsquigarrow \mathit{tnvs} \\ \mathbf{duplicates}\,(\mathit{tnvs}) = \emptyset \\ \overline{y_i.}^{\,i}\ x \notin \mathbf{dom}\,(\Delta)\end{array}}{\overline{y_i}^{\,i}, \Delta_1, E \vdash \mathbf{tc}\ x\ l\ \mathit{tnvars}^{\,l} = \langle |x_1^l : \mathit{typ}_1; \ldots; x_j^l : \mathit{typ}_j\ ;^{?}| \rangle \triangleright \{\overline{y_i.}^{\,i}\ x \mapsto \mathit{tnvs}\,\}, \{x \mapsto \overline{y_i.}^{\,i}\ x\}}\ \text{CHECK\_TEXP\_TC\_REC}$$

$$\frac{\begin{array}{l}\mathit{tnvars}^{\,l} \rightsquigarrow \mathit{tnvs} \\ \mathbf{duplicates}\,(\mathit{tnvs}) = \emptyset \\ \overline{y_i.}^{\,i}\ x \notin \mathbf{dom}\,(\Delta)\end{array}}{\overline{y_i}^{\,i}, \Delta_1, E \vdash \mathbf{tc}\ x\ l\ \mathit{tnvars}^{\,l} = |^{?}\ \mathit{ctor\_def}_1 | \ldots | \mathit{ctor\_def}_j \triangleright \{\overline{y_i.}^{\,i}\ x \mapsto \mathit{tnvs}\,\}, \{x \mapsto \overline{y_i.}^{\,i}\ x\}}\ \text{CHECK\_TEXP\_TC\_VAR}$$

$\boxed{\mathit{xs}, \Delta_1, E \vdash \mathbf{tc}\ \mathit{td}_1\ ..\ \mathit{td}_i \triangleright \Delta_2, E^{\mathrm{P}}}$ Extract the type constructor information

$$\frac{}{\mathit{xs}, \Delta, E \vdash \mathbf{tc} \triangleright \{\,\}, \{\,\}}\ \text{CHECK\_TEXPS\_TC\_EMPTY}$$

$$\frac{\begin{array}{l}\mathit{xs}, \Delta_1, E \vdash \mathbf{tc}\ \mathit{td} \triangleright \Delta_2, E_2^{\mathrm{P}} \\ \mathit{xs}, \Delta_1 \uplus \Delta_2, E \uplus \langle \{\,\}, E_2^{\mathrm{P}}, \{\,\}, \{\,\} \rangle \vdash \mathbf{tc}\ \overline{\mathit{td}_i}^{\,i} \triangleright \Delta_3, E_3^{\mathrm{P}} \\ \mathbf{dom}\,(E_2^{\mathrm{P}}) \cap \mathbf{dom}\,(E_3^{\mathrm{P}}) = \emptyset\end{array}}{\mathit{xs}, \Delta_1, E \vdash \mathbf{tc}\ \mathit{td}\ \overline{\mathit{td}_i}^{\,i} \triangleright \Delta_2 \uplus \Delta_3, E_2^{\mathrm{P}} \uplus E_3^{\mathrm{P}}}\ \text{CHECK\_TEXPS\_TC\_ABBREV}$$

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

$$\frac{}{\Delta, E \vdash \mathit{tnvs}\ p = \mathit{typ} \triangleright \langle \{\,\}, \{\,\} \rangle}\ \text{CHECK\_TEXP\_ABBREV}$$

$$\frac{\begin{array}{l}\overline{\Delta, E \vdash \mathit{typ}_i \rightsquigarrow t_i}^{\,i} \\ \mathit{names} = \{\, \overline{x_i}^{\,i} \,\} \\ \mathbf{duplicates}\,(\,\overline{x_i}^{\,i}\,) = \emptyset \\ \overline{\mathbf{FV}\,(t_i) \subset \mathit{tnvs}}^{\,i} \\ E^{\mathrm{F}} = \{\, \overline{x_i \mapsto \langle\, \mathbf{forall}\ \mathit{tnvs}.p \to t_i, (x_i\ \mathbf{of}\ \mathit{names}) \rangle}^{\,i} \,\}\end{array}}{\Delta, E \vdash \mathit{tnvs}\ p = \langle |\, \overline{x_i^l : \mathit{typ}_i}^{\,i}\ ;^{?}| \rangle \triangleright \langle E^{\mathrm{F}}, \{\,\} \rangle}\ \text{CHECK\_TEXP\_REC}$$

$$\frac{\begin{array}{l}\overline{\Delta, E \vdash typs_i \rightsquigarrow t\_multi_i}^{\,i} \\ names = \{\, \overline{x_i}^{\,i} \,\} \\ \mathbf{duplicates}\,(\,\overline{x_i}^{\,i}\,) = \emptyset \\ \overline{\mathbf{FV}\,(t\_multi_i) \subset tnvs}^{\,i} \\ E^{\mathrm{X}} = \{\, \overline{x_i \mapsto \langle\, \mathbf{forall}\, tnvs.t\_multi_i \to p, (x_i\, \mathbf{of}\, names)\rangle}^{\,i} \,\}\end{array}}{\Delta, E \vdash tnvs\; p = |^{?}\, \overline{x_i^l\, \mathbf{of}\, typs_i}^{\,i} \triangleright \langle \{\,\}, E^{\mathrm{X}}\rangle}\ \textsc{check\_texp\_var}$$

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

$$\frac{}{\overline{y_i}^{\,i}, \Delta, E \vdash \triangleright \langle \{\,\}, \{\,\}\rangle}\ \textsc{check\_texps\_empty}$$

$$\frac{\begin{array}{l}tnvars^l \rightsquigarrow tnvs \\ \Delta, E_1 \vdash tnvs\, \overline{y_i.}^{\,i}\, x = texp \triangleright \langle E_1^{\mathrm{F}}, E_1^{\mathrm{X}}\rangle \\ \overline{y_i}^{\,i}, \Delta, E \vdash \overline{td_j}^{\,j} \triangleright \langle E_2^{\mathrm{F}}, E_2^{\mathrm{X}}\rangle \\ \mathbf{dom}\,(E_1^{\mathrm{X}}) \cap \mathbf{dom}\,(E_2^{\mathrm{X}}) = \emptyset \\ \mathbf{dom}\,(E_1^{\mathrm{F}}) \cap \mathbf{dom}\,(E_2^{\mathrm{F}}) = \emptyset\end{array}}{\overline{y_i}^{\,i}, \Delta, E \vdash x\; l\, tnvars^l\; = texp\, \overline{td_j}^{\,j} \triangleright \langle E_1^{\mathrm{F}} \uplus E_2^{\mathrm{F}}, E_1^{\mathrm{X}} \uplus E_2^{\mathrm{X}}\rangle}\ \textsc{check\_texps\_cons\_concrete}$$

$$\frac{\overline{y_i}^{\,i}, \Delta, E \vdash \overline{td_j}^{\,j} \triangleright \langle E^{\mathrm{F}}, E^{\mathrm{X}}\rangle}{\overline{y_i}^{\,i}, \Delta, E \vdash x\; l\, tnvars^l\; \overline{td_j}^{\,j} \triangleright \langle E^{\mathrm{F}}, E^{\mathrm{X}}\rangle}\ \textsc{check\_texps\_cons\_abstract}$$

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

$$\frac{\begin{array}{c}E(id) \triangleright p \\ \delta(p) \triangleright xs\end{array}}{\delta, E \vdash id \rightsquigarrow p}\ \textsc{convert\_class\_all}$$

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

$$\frac{}{I \vdash (p\; \alpha)\, \mathbf{IN}\, (p_1\; tnv_1) \,..\, (p_i\; tnv_i)(p\; \alpha)(p'_1\; tnv'_1) \,..\, (p'_j\; tnv'_j)}\ \textsc{solve\_class\_constraint\_immediate}$$

$$\frac{\begin{array}{l}(p_1\; tnv_1) \,..\, (p_n\; tnv_n) \Rightarrow (p\; t)\, \mathbf{IN}\, I \\ I \vdash (p_1\; \sigma(tnv_1))\, \mathbf{IN}\, \mathcal{C} \quad .. \quad I \vdash (p_n\; \sigma(tnv_n))\, \mathbf{IN}\, \mathcal{C}\end{array}}{I \vdash (p\; \sigma(t))\, \mathbf{IN}\, \mathcal{C}}\ \textsc{solve\_class\_constraint\_chain}$$

$\boxed{I \vdash \Sigma^{\mathcal{C}} \triangleright \mathcal{C}}$ Solve class constraints

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

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

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

$$\frac{\begin{array}{l}\overline{\Delta, E, E^{\mathrm{L}} \vdash \mathit{funcl}_i \triangleright \{x_i \mapsto t_i\}, \Sigma^{\mathcal{C}}{}_i, \Sigma^{\mathcal{N}}{}_i}^{\,i} \\ I \vdash \Sigma^{\mathcal{C}} \triangleright \mathcal{C} \\ \overline{\mathbf{FV}\,(t_i) \subset \mathit{tnvs}}^{\,i} \\ \mathbf{FV}\,(\mathcal{C}) \subset \mathit{tnvs} \\ \mathbf{compatible\ overlap}\,(\,\overline{x_i \mapsto t_i}^{\,i}\,) \\ E^{\mathrm{L}} = \{\,\overline{x_i \mapsto t_i}^{\,i}\,\}\end{array}}{\Delta, I, E \vdash \mathbf{let\,rec}\,\tau^{?}\,\overline{\mathit{funcl}_i}^{\,i} \triangleright \{\,\overline{x_i \mapsto \langle\,\mathbf{forall}\,\mathit{tnvs}.\mathcal{C} \Rightarrow t_i, \mathbf{let}\rangle}^{\,i}\,\}}\ \text{CHECK\_VAL\_DEF\_RECFUN}$$

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

$$\frac{}{\Delta, (\alpha) \vdash \alpha\,\mathbf{instance}}\ \text{CHECK\_T\_INSTANCE\_VAR}$$

$$\frac{}{\Delta, (\alpha_1, \ldots., \alpha_n) \vdash \alpha_1 * \ldots. * \alpha_n\,\mathbf{instance}}\ \text{CHECK\_T\_INSTANCE\_TUP}$$

$$\frac{}{\Delta, (\alpha_1, \alpha_2) \vdash \alpha_1 \to \alpha_n\,\mathbf{instance}}\ \text{CHECK\_T\_INSTANCE\_FN}$$

$$\frac{\Delta(p) \triangleright \alpha'_1 .. \alpha'_n}{\Delta, (\alpha_1, .., \alpha_n) \vdash p\,\alpha_1 .. \alpha_n\,\mathbf{instance}}\ \text{CHECK\_T\_INSTANCE\_TC}$$

$\boxed{\overline{z_j}^{\,j}, D_1, E_1 \vdash \mathit{def} \triangleright D_2, E_2}$ Check a definition

$$\frac{\begin{array}{c}\overline{z_j}^{\,j}, \Delta_1, E \vdash \mathbf{tc}\,\overline{td_i}^{\,i} \triangleright \Delta_2, E^{\mathrm{P}} \\ \overline{z_j}^{\,j}, \Delta_1 \uplus \Delta_2, E \uplus \langle\{\,\}, E^{\mathrm{P}}, \{\,\}, \{\,\}\rangle \vdash \overline{td_i}^{\,i} \triangleright \langle E^{\mathrm{F}}, E^{\mathrm{X}}\rangle\end{array}}{\overline{z_j}^{\,j}, \langle\Delta_1, \delta, I\rangle, E \vdash \mathbf{type}\,\overline{td_i}^{\,i}\,l \triangleright \langle\Delta_2, \{\,\}, \{\,\}\rangle, \langle\{\,\}, E^{\mathrm{P}}, E^{\mathrm{F}}, E^{\mathrm{X}}\rangle}\ \text{CHECK\_DEF\_TYPE}$$

$$\frac{\Delta, I, E \vdash \mathit{val\_def} \triangleright E^{\mathrm{X}}}{\overline{z_j}^{\,j}, \langle\Delta, \delta, I\rangle, E \vdash \mathit{val\_def}\,l \triangleright \epsilon, \langle\{\,\}, \{\,\}, \{\,\}, E^{\mathrm{X}}\rangle}\ \text{CHECK\_DEF\_VAL\_DEF}$$

$$\frac{\begin{array}{l}\overline{\Delta, E_1, E^{\mathrm{L}} \vdash \mathit{rule}_i \triangleright \{x_i \mapsto t_i\}, \Sigma^{\mathcal{C}}{}_i, \Sigma^{\mathcal{N}}{}_i}^{\,i} \\ I \vdash \overline{\Sigma^{\mathcal{C}}{}_i}^{\,i} \triangleright \mathcal{C} \\ \overline{\mathbf{FV}\,(t_i) \subset \mathit{tnvs}}^{\,i} \\ \mathbf{FV}\,(\mathcal{C}) \subset \mathit{tnvs} \\ \mathbf{compatible\ overlap}\,(\,\overline{x_i \mapsto t_i}^{\,i}\,) \\ E^{\mathrm{L}} = \{\,\overline{x_i \mapsto t_i}^{\,i}\,\} \\ E_2 = \langle\{\,\}, \{\,\}, \{\,\}, \{\,\overline{x_i \mapsto \langle\,\mathbf{forall}\,\mathit{tnvs}.\mathcal{C} \Rightarrow t_i, \mathbf{let}\rangle}^{\,i}\,\}\rangle\end{array}}{\overline{z_j}^{\,j}, \langle\Delta, \delta, I\rangle, E_1 \vdash \mathbf{indreln}\,\tau^{?}\,\overline{\mathit{rule}_i}^{\,i}\,l \triangleright \epsilon, E_2}\ \text{CHECK\_DEF\_INDRELN}$$

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

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

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

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

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

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

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

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

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

```
Definition rules:          145 good    0 bad
Definition rule clauses:   437 good    0 bad
```