ident, x, y, y_p, y_f, _, *abbrev*, r	subscripts: p for pointers, f for functions
n, i, j	index variables
impl_const	implementation-defined constant
mem_int	memory integer value
member	C struct/union member name
	Ott-hack, ignore (annotations)
nat	OCaml arbitrary-width natural number
mem_ptr	abstract pointer value
mem_val	abstract memory value
	Ott-hack, ignore (locations)
mem_iv_c	OCaml type for memory constraints on integer values
UB_name	undefined behaviour
string	OCaml string
	Ott-hack, ignore (OCaml type variable TY)
	Ott-hack, ignore (Symbol.prefix)
mem_order, _	OCaml type for memory order
linux_mem_order	OCaml type for Linux memory order
logical_val	logical values (to be specified)
	Ott-hack, ignore (OCaml type variable bt)

$Sctypes\_t,\ \tau$	::=			C type
	|	$\tau*$		pointer to type τ
tag	::=			OCaml type for struct/union tag
	|	$ident$		
$\beta,\ \_$	::=			base types
	|	`unit`		unit
	|	`bool`		boolean
	|	`integer`		integer
	|	`real`		rational numbers?
	|	`loc`		location
	|	`array` β		array
	|	`list` β		list
	|	$\overline{\beta_i}^{\,i}$		tuple
	|	`struct` tag		struct
	|	`set` β		set
	|	`opt` (β)		option
	|	$\beta \to \beta'$		parameter types
	|	β_τ	M	of a C type
$binop$	::=			binary operators
	|	`+`		addition
	|	`-`		subtraction
	|	`*`		multiplication
	|	`/`		division
	|	`rem_t`		modulus
	|	`rem_f`		remainder
	|	`^`		exponentiation
	|	`=`		equality, defined both for integer and C types

	|	`>`	greater than
	|	`<`	less than
	|	`>=`	greater than or equal to
	|	`<=`	less than or equal to
	|	`/\`	conjucntion
	|	`\/`	disjunction
$binop_{arith}$	::=		arithmentic binary operators
	|	`+`	
	|	`-`	
	|	`*`	
	|	`/`	
	|	`rem_t`	
	|	`rem_f`	
	|	`^`	
$binop_{rel}$	::=		relational binary operators
	|	`=`	
	|	`>`	
	|	`<`	
	|	`>=`	
	|	`<=`	
$binop_{bool}$	::=		boolean binary operators
	|	`/\`	
	|	`\/`	
object_value	::=		C object values (inhabitants of object types), which can be read/stored
	|	*mem_int*	integer value
	|	*mem_ptr*	pointer value

	$\mid$	$\mathtt{array}\,(\,\overline{\mathit{loaded\_value}_i}^{\,i}\,)$	C array value
	$\mid$	$(\,\mathtt{struct}\,\mathit{ident})\{\,\overline{.\,\mathit{member}_i{:}\tau_i = \mathit{mem\_val}_i}^{\,i}\,\}$	C struct value
	$\mid$	$(\,\mathtt{union}\,\mathit{ident})\{\,.\,\mathit{member} = \mathit{mem\_val}\}$	C union value
loaded_value	::=		potentially unspecified C object values
	$\mid$	$\mathtt{specified}\,\mathit{object\_value}$	specified loaded value
value	::=		Core values
	$\mid$	*object_value*	C object value
	$\mid$	*loaded_value*	loaded C object value
	$\mid$	`Unit`	unit
	$\mid$	`True`	boolean true
	$\mid$	`False`	boolean false
	$\mid$	$\beta[\,\overline{\mathit{value}_i}^{\,i}\,]$	list
	$\mid$	$(\,\overline{\mathit{value}_i}^{\,i}\,)$	tuple
ctor_val	::=		data constructors
	$\mid$	$\mathtt{Nil}\,\beta$	empty list
	$\mid$	`Cons`	list cons
	$\mid$	`Tuple`	tuple
	$\mid$	`Array`	C array
	$\mid$	`Specified`	non-unspecified loaded value
ctor_expr	::=		data constructors
	$\mid$	`Ivmax`	max integer value
	$\mid$	`Ivmin`	min integer value
	$\mid$	`Ivsizeof`	sizeof value
	$\mid$	`Ivalignof`	alignof value
	$\mid$	`IvCOMPL`	bitwise complement
	$\mid$	`IvAND`	bitwise AND

	|	`IvOR`	bitwise OR
	|	`IvXOR`	bitwise XOR
	|	`Fvfromint`	cast integer to floating value
	|	`Ivfromfloat`	cast floating to integer value
name	::=		
	|	*ident*	Core identifier
	|	*impl_const*	implementation-defined constant
pval	::=		pure values
	|	*ident*	Core identifier
	|	*impl_const*	implementation-defined constant
	|	*value*	Core values
	|	$\texttt{constrained}\,(\overline{mem\_iv\_c_i, pval_i}^{\,i}\,)$	constrained value
	|	$\texttt{error}\,(string, pval)$	impl-defined static error
	|	$ctor\_val(\overline{pval_i}^{\,i}\,)$	data constructor application
	|	$(\,\texttt{struct}\,ident)\{\overline{.\,member_i = pval_i}^{\,i}\,\}$	C struct expression
	|	$(\,\texttt{union}\,ident)\{\,.\,member = pval\}$	C union expression
pexpr	::=		pure expressions
	|	*pval*	pure values
	|	$ctor\_expr(\overline{pval_i}^{\,i}\,)$	data constructor application
	|	$\texttt{array\_shift}\,(pval_1, \tau, pval_2)$	pointer array shift
	|	$\texttt{member\_shift}\,(pval, ident, member)$	pointer struct/union member shift
	|	$\texttt{not}\,(pval)$	boolean not
	|	$pval_1\ binop\ pval_2$	binary operations
	|	$\texttt{memberof}\,(ident, member, pval)$	C struct/union member access
	|	$name(\overline{pval_i}^{\,i}\,)$	pure function call
	|	$\texttt{assert\_undef}\,(pval,\ UB\_name)$	
	|	$\texttt{bool\_to\_integer}\,(pval)$	

	$\vert$	$\texttt{conv\_int}\,(\tau, \mathit{pval})$		
	$\vert$	$\texttt{wrapI}\,(\tau, \mathit{pval})$		
tpval	$::=$			top-level pure values
	$\vert$	$\texttt{undef}\ \mathit{UB\_name}$		undefined behaviour
	$\vert$	$\texttt{done}\,\mathit{pval}$		pure done
$\mathit{ident\_opt\_\beta}$	$::=$			type annotated optional identifier
	$\vert$	$\_{:}\beta$		
	$\vert$	$\mathit{ident}{:}\beta$		
$\mathit{pattern}$	$::=$			
	$\vert$	$\mathit{ident\_opt\_\beta}$		
	$\vert$	$\mathit{ctor\_val}(\,\overline{\mathit{pattern}_i}^{\,i}\,)$		
$\mathit{ident\_or\_pattern}$	$::=$			
	$\vert$	ident		
	$\vert$	$\mathit{pattern}$		
tpexpr	$::=$			top-level pure expressions
	$\vert$	tpval		top-level pure values
	$\vert$	$\texttt{case}\,\mathit{pval}\,\texttt{of}\,\overline{\mid \mathit{pattern}_i \Rightarrow \mathit{tpexpr}_i}^{\,i}\,\texttt{end}$		pattern matching
	$\vert$	$\texttt{let}\,\mathit{ident\_or\_pattern} = \mathit{pexpr}\,\texttt{in}\,\mathit{tpexpr}$		pure let
	$\vert$	$\texttt{if}\,\mathit{pval}\,\texttt{then}\,\mathit{tpexpr}_1\,\texttt{else}\,\mathit{tpexpr}_2$		pure if
	$\vert$	$[\mathcal{C}/\mathcal{C}']\mathit{tpexpr}$	M	simul-sub all vars in $\mathcal{C}$ for all vars in $\mathcal{C}'$ in tpexpr
$\mathit{m\_kill\_kind}$	$::=$			
	$\vert$	$\texttt{dynamic}$		
	$\vert$	$\texttt{static}\,\tau$		

bool, _	::=		OCaml booleans
	|	`true`	
	|	`false`	
int, _	::=		OCaml fixed-width integer
	|	i	literal integer
mem_action	::=		memory actions
	|	`create` $(pval, \tau)$	
	|	`create_readonly` $(pval_1, \tau, pval_2)$	
	|	`alloc` $(pval_1, pval_2)$	
	|	`kill` $(m\_kill\_kind, pval)$	
	|	`store` $(bool, \tau, pval_1, pval_2, mem\_order)$	true means store is locking
	|	`load` $(\tau, pval, mem\_order)$	
	|	`rmw` $(\tau, pval_1, pval_2, pval_3, mem\_order_1, mem\_order_2)$	
	|	`fence` (mem_order)	
	|	`cmp_exch_strong` $(\tau, pval_1, pval_2, pval_3, mem\_order_1, mem\_order_2)$	
	|	`cmp_exch_weak` $(\tau, pval_1, pval_2, pval_3, mem\_order_1, mem\_order_2)$	
	|	`linux_fence` $(linux\_mem\_order)$	
	|	`linux_load` $(\tau, pval, linux\_mem\_order)$	
	|	`linux_store` $(\tau, pval_1, pval_2, linux\_mem\_order)$	
	|	`linux_rmw` $(\tau, pval_1, pval_2, linux\_mem\_order)$	
polarity	::=		polarities for memory actions
	|	`Pos`	sequenced by `let weak` and `let strong`
	|	`Neg`	only sequenced by `let strong`
pol_mem_action	::=		memory actions with polarity
	|	*polarity mem_action*	

mem_op	::=		operations involving the memory state
	|	$pval_1 \equiv pval_2$	pointer equality comparison
	|	$pval_1 \neq pval_2$	pointer inequality comparison
	|	$pval_1 < pval_2$	pointer less-than comparison
	|	$pval_1 > pval_2$	pointer greater-than comparison
	|	$pval_1 \leq pval_2$	pointer less-than comparison
	|	$pval_1 \geq pval_2$	pointer greater-than comparison
	|	$pval_1 -_{\tau} pval_2$	pointer subtraction
	|	$\texttt{intFromPtr}\,(\tau_1, \tau_2, pval)$	cast of pointer value to integer value
	|	$\texttt{ptrFromInt}\,(\tau_1, \tau_2, pval)$	cast of integer value to pointer value
	|	$\texttt{ptrValidForDeref}\,(\tau, pval)$	dereferencing validity predicate
	|	$\texttt{ptrWellAligned}\,(\tau, pval)$	
	|	$\texttt{ptrArrayShift}\,(pval_1, \tau, pval_2)$	
	|	$\texttt{memcpy}\,(pval_1, pval_2, pval_3)$	
	|	$\texttt{memcmp}\,(pval_1, pval_2, pval_3)$	
	|	$\texttt{realloc}\,(pval_1, pval_2, pval_3)$	
	|	$\texttt{va\_start}\,(pval_1, pval_2)$	
	|	$\texttt{va\_copy}\,(pval)$	
	|	$\texttt{va\_arg}\,(pval, \tau)$	
	|	$\texttt{va\_end}\,(pval)$	
$spine\_elem$	::=		spine element
	|	$pval$	pure value
	|	$logical\_val$	logical variable
	|	res_term	resource valuel
$tval$	::=		(effectful) top-level values
	|	$\texttt{done}\ \overline{spine\_elem_i}^{\,i}$	end of top-level expression
	|	$\texttt{undef}\ UB\_name$	undefined behaviour

$bool\_op$	::=			
	$\mid$	$\neg\, term$		
	$\mid$	$term_1 = term_2$		
	$\mid$	$\bigwedge(\, \overline{term_i}^{\,i}\,)$		
	$\mid$	$\bigvee(\, \overline{term_i}^{\,i}\,)$		
	$\mid$	$term_1\ binop_{bool}\ term_2$	M	
	$\mid$	$\texttt{if}\ term_1\ \texttt{then}\ term_2\ \texttt{else}\ term_3$		
$arith\_op$	::=			
	$\mid$	$term_1 + term_2$		
	$\mid$	$term_1 - term_2$		
	$\mid$	$term_1 \times term_2$		
	$\mid$	$term_1 / term_2$		
	$\mid$	$term_1\ \texttt{rem\_t}\ term_2$		
	$\mid$	$term_1\ \texttt{rem\_f}\ term_2$		
	$\mid$	$term_1\ \texttt{\^{}}\ term_2$		
	$\mid$	$term_1\ binop_{arith}\ term_2$	M	
cmp_op	::=			
	$\mid$	$term_1 < term_2$		less than
	$\mid$	$term_1 \leq term_2$		less than or equal
	$\mid$	$term_1\ binop_{rel}\ term_2$	M	
$list\_op$	::=			
	$\mid$	$\texttt{nil}$		
	$\mid$	$\texttt{tl}\ term$		
	$\mid$	$term^{(int)}$		
$tuple\_op$	::=			
	$\mid$	$(\, \overline{term_i}^{\,i}\,)$		

$$
\begin{array}{lll}
 & | & term^{(int)} \\
\\
pointer\_op & ::= & \\
 & | & mem\_ptr \\
 & | & term_1 +_{\mathrm{ptr}} term_2 \\
\\
option\_op & ::= & \\
 & | & \texttt{none}\,\beta \\
 & | & \texttt{some}\,term \\
\\
array\_op & ::= & \\
 & | & term_1[term_2] \\
\\
param\_op & ::= & \\
 & | & ident{:}\beta.\,term \\
 & | & term(term_1, .., term_n) \\
\\
struct\_op & ::= & \\
 & | & term.member \\
\\
ct\_pred & ::= & \\
 & | & \texttt{representable}\,(\tau, term) \\
 & | & \texttt{alignedI}\,(term_1, term_2) \\
\\
term,\ \_ & ::= & \\
 & | & lit \\
 & | & arith\_op \\
 & | & bool\_op \\
 & | & cmp\_op \\
 & | & tuple\_op
\end{array}
$$

	$\mid$	$struct\_op$		
	$\mid$	$pointer\_op$		
	$\mid$	$list\_op$		
	$\mid$	$array\_op$		
	$\mid$	ct_pred		
	$\mid$	$option\_op$		
	$\mid$	$param\_op$		
	$\mid$	$(term)$	S	parentheses
	$\mid$	$[term_1/ident]term_2$	M	substitute $term_1$ for $ident$ in $term_2$
	$\mid$	$pval$	M	only the ones which can be embeded into the SMT value grammar, so no array literals
$terms$	$::=$			non-empty list of terms
	$\mid$	$[term_1, \ldots, term_n]$		
$predicate\_name$	$::=$			names of predicates
	$\mid$	$Sctypes\_t$		C type
	$\mid$	$string$		arbitrary
$init,$	$::=$			initialisation status
	$\mid$	$\checkmark$		initialised
	$\mid$	$\times$		uninitalised
$predicate$	$::=$			arbitrary predicate
	$\mid$	$terms_1 \, \mathbb{Q}\overset{init}{\mapsto}_{predicate\_name} \, terms_2$		
$resource$	$::=$			resources
	$\mid$	$\texttt{emp}$		empty heap
	$\mid$	$predicate$		heap predicate
	$\mid$	$term$		logical term
	$\mid$	$resource_1 \star resource_2$		seperating conjunction

	$\vert$	$\exists\, ident{:}\beta.\ resource$		existential
	$\vert$	$resource_1 \wedge resource_2$		logical conjuction
	$\vert$	$[pval/ident]resource$	M	substitute $pval$ for $ident$ in $resource$
res_term	$::=$			resource terms
	$\vert$	$\texttt{emp}$		empty heap
	$\vert$	$ident$		variable
	$\vert$	$\langle res\_term_1, res\_term_2\rangle$		seperating-conjunction pair
	$\vert$	$\texttt{pack}\,(pval, res\_term_2)$		packing for existentials
	$\vert$	(res_term_1, res_term_2)		logical-conjunction pair
$ret\_pattern$	$::=$			return pattern
	$\vert$	$\texttt{comp}\, ident$		computational variable
	$\vert$	$\texttt{log}\, ident$		logical variable
	$\vert$	$\texttt{res}\, ident$		resource variable
seq_expr	$::=$			sequential (effectful) expressions
	$\vert$	$pval$		pure values
	$\vert$	$\texttt{ccall}\,(\tau, pval, \overline{pval_i}^{\,i})$		C function call
	$\vert$	$\texttt{pcall}\,(name, \overline{pval_i}^{\,i})$		procedure call
seq_texpr	$::=$			sequential top-level (effectful) expressions
	$\vert$	$tval$		(effectful) top-level values
	$\vert$	$\texttt{run}\, ident\, pval_1, .., pval_n$		run from label
	$\vert$	$\texttt{nd}\,(pval_1, .., pval_n)$		nondeterministic choice
	$\vert$	$\texttt{let}\, ret\_pattern = seq\_expr\, \texttt{in}\, texpr$		bind return patterns
	$\vert$	$\texttt{letC}\, ident\_or\_pattern = seq\_expr\, \texttt{in}\, texpr$		bind computational patterns
	$\vert$	$\texttt{case}\, pval\, \texttt{with}\, \overline{\vert\ pattern_i \Rightarrow texpr_i}^{\,i}\, \texttt{end}$		pattern matching
	$\vert$	$\texttt{if}\, pval\, \texttt{then}\, texpr_1\, \texttt{else}\, texpr_2$		conditional
	$\vert$	$\texttt{bound}\,[int](is\_texpr)$		limit scope of indet seq behaviour, absent at runtime

is_expr	::=		indet seq (effectful) expressions
	|	`memop` (*mem_op*)	pointer op involving memory
	|	*pol_mem_action*	memory action
	|	`unseq` ($texpr_1$, .. , $texpr_n$)	unsequenced expressions
is_texpr	::=		indet seq top-level (effectful) expressions
	|	`let weak` *pattern* = *is_expr* `in` `mu_texpr_aux`	weak sequencing
	|	`let strong` *ident_or_pattern* = *is_expr* `in` `mu_texpr_aux`	strong sequencing
texpr	::=		top-level (effectful) expressions
	|	*seq_texpr*	sequential (effectful) expressions
	|	*is_texpr*	indet seq (effectful) expressions
terminals	::=		
	|	λ	
	|	$\longrightarrow$	
	|	$\rightarrow$	
	|	$\leadsto$	
	|	$\Rightarrow$	
	|	$\Leftarrow$	
	|	$\vdash$	
	|	$\in$	
	|	Π	
	|	$\forall$	
	|	$\multimap$	
	|	$\supset$	
	|	Σ	
	|	$\exists$	
	|	$\star$	
	|	$\times$	

	$\mid$	$\wedge$		
	$\mid$	$\bigwedge$		
	$\mid$	$\neg$		
	$\mid$	$=$		
	$\mid$	$\neq$		
	$\mid$	$\leq$		
	$\mid$	$\geq$		
	$\mid$	$\&$		
	$\mid$	$.$		
	$\mid$	$\mid$		
	$\mid$	$+_{\mathrm{ptr}}$		
	$\mid$	$\mapsto$		
	$\mid$	$*$		
	$\mid$	$::$		
	$\mid$	$\checkmark$		
	$\mid$	$:$		
	$\mid$	$.$		
	$\mid$	$.$		
	$\mid$	$\gg$		
	$\mid$	$::$		
	$\mid$	$\hat{}$		
	$\mid$	$\bigvee$		
	$\mid$	$\equiv$		
	$\mid$	$\langle$		
	$\mid$	$\rangle$		
z	$::=$			OCaml arbitrary-width integer
	$\mid$	i	M	literal integer
	$\mid$	$\mathrm{to\_int}(\mathit{mem\_int})$	M	
	$\mid$	$\mathrm{size\_of}(\tau)$	M	size of a C type

	$\|$	$\text{offset\_of}_{tag}(member)$	M	offset of a struct member
	$\|$	`ptr_size`	M	size of a pointer
	$\|$	max_int_{τ}	M	maximum value of int of type τ
	$\|$	min_int_{τ}	M	minimum value of int of type τ
$\mathbb{Q}$	$::=$			OCaml type for rational numbers
	$\|$	$\frac{int_1}{int_2}$		
lit	$::=$			
	$\|$	$ident$		
	$\|$	`unit`		
	$\|$	$bool$		
	$\|$	z		
	$\|$	$\mathbb{Q}$		
arg	$::=$			argument/function types
	$\|$	$\Pi\, ident{:}\beta.\ arg$		
	$\|$	$\forall\, ident{:}\beta.\ arg$		
	$\|$	$resource \multimap arg$		
	$\|$	$term \supset arg$		
	$\|$	ret		
	$\|$	$[spine\_elem/ident]arg$	M	
$pure\_arg$	$::=$			pure argument/function types
	$\|$	$\Pi\, ident{:}\beta.\ pure\_arg$		
	$\|$	$term \supset pure\_arg$		
	$\|$	$pure\_ret$		
	$\|$	$[spine\_elem/ident]pure\_arg$	M	
$ret,\ \_$	$::=$			return types

	$\mid$	$\Sigma\, ident{:}\beta.\ ret$		
	$\mid$	$\exists\, ident{:}\beta.\ ret$		
	$\mid$	$resource \star ret$		
	$\mid$	$term \wedge ret$		
	$\mid$	$\mathtt{I}$		
	$\mid$	$[spine\_elem/ident]ret$	M	
$pure\_ret$	$::=$			pure return types
	$\mid$	$\Sigma\, ident{:}\beta.\ pure\_ret$		
	$\mid$	$term \wedge pure\_ret$		
	$\mid$	$\mathtt{I}$		
	$\mid$	$[spine\_elem/ident]pure\_ret$	M	
$\mathcal{C}$	$::=$			computational var env
	$\mid$	$\cdot$		
	$\mid$	$\mathcal{C}, ident{:}\beta$		
	$\mid$	$\overline{\mathcal{C}_i}^{\,i}$		
	$\mid$	$\mathrm{fresh}(\mathcal{C})$	M	identical context except with fresh variable names
$\mathcal{L}$	$::=$			logical var env
	$\mid$	$\cdot$		
	$\mid$	$\mathcal{L}, ident{:}\beta$		
Φ	$::=$			constraints env
	$\mid$	$\cdot$		
	$\mid$	$\Phi, term$		
	$\mid$	$\overline{\Phi_i}^{\,i}$		
	$\mid$	$[\mathcal{C}/\mathcal{C}']\Phi$	M	
$\mathcal{R}$	$::=$			resources env

	$\mid$	$\cdot$
	$\mid$	$\mathcal{R}, \mathit{ident}{:}\mathit{resource}$
	$\mid$	$\mathcal{R}_1, \mathcal{R}_2$
formula	$::=$	
	$\mid$	$\mathit{judgement}$
	$\mid$	$\mathit{abbrev} \equiv \mathit{term}$
	$\mid$	$\mathtt{smt}\,(\Phi \Rightarrow \mathit{term})$
	$\mid$	$\mathtt{smt}\,(\Phi \Rightarrow \mathit{resource}_1 = \mathit{resource}_2)$
	$\mid$	$\mathit{ident}{:}\beta \in \mathcal{C}$
	$\mid$	$\mathit{ident}{:}\,\mathtt{struct}\,\mathit{tag}\ \&\ \overline{\mathit{member}_i{:}\tau_i}^{\,i} \in \mathtt{Globals}$
	$\mid$	$\overline{\mathcal{C}_i; \mathcal{L}_i; \Phi_i \vdash \mathit{mem\_val}_i \Rightarrow \mathtt{mem}\,\beta_i}^{\,i}$ — dependent on memory object model
	$\mid$	$\overline{\mathcal{C}_i; \mathcal{L}_i; \Phi_i \vdash \mathit{pval}_i \Rightarrow \beta_i}^{\,i}$
	$\mid$	$\mathcal{C} \vdash \mathit{name} \Rightarrow \mathit{pure\_arg}$
	$\mid$	$\overline{\mathit{term}_i\ \mathtt{as}\ \mathit{pattern}_i{:}\beta_i \leadsto \mathcal{C}_i; \Phi_i}^{\,i}$
	$\mid$	$\overline{\mathcal{C}_i; \mathcal{L}_i; \Phi_i \vdash \mathit{tpexpr}_i \Leftarrow y_i{:}\beta_i.\ \mathit{term}_i}^{\,i}$
	$\mid$	$\mathcal{L} \vdash \mathit{logical\_val}{:}\beta$
object_value_jtype	$::=$	
	$\mid$	$\mathcal{C}; \mathcal{L}; \Phi \vdash \mathit{object\_value} \Rightarrow \mathtt{obj}\,\beta$
pval_jtype	$::=$	
	$\mid$	$\mathcal{C}; \mathcal{L}; \Phi \vdash \mathit{pval} \Rightarrow \beta$
resource_jtype	$::=$	
	$\mid$	$\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \mathit{res\_term} \Leftarrow \mathit{resource}$
spine_jtype	$::=$	
	$\mid$	$\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \overline{\mathit{spine\_elem}_i}^{\,i} :: \mathit{arg} \gg \mathit{ret}$

$$
\begin{array}{lll}
\mathit{pexpr\_jtype} & ::= & \\
 & | & \mathcal{C}; \mathcal{L}; \Phi \vdash \mathit{pexpr} \Rightarrow \mathit{ident}{:}\beta.\, \mathit{term} \\[1em]
\mathit{pattern\_jtype} & ::= & \\
 & | & \mathit{term}\ \mathsf{as}\ \mathit{pattern}{:}\beta \leadsto \mathcal{C}; \Phi \\
 & | & \mathit{term}\ \mathsf{as}\ \mathit{ident\_or\_pattern}{:}\beta \leadsto \mathcal{C}; \Phi \\
 & | & \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \overline{\mathit{ret\_pattern}_i}^{\,i} {:}\mathit{ret} \leadsto \mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}' \\[1em]
\mathit{tpval\_jtype} & ::= & \\
 & | & \mathcal{C}; \mathcal{L}; \Phi \vdash \mathit{tpval} \Leftarrow \mathit{ident}{:}\beta.\, \mathit{term} \\[1em]
\mathit{tpexpr\_jtype} & ::= & \\
 & | & \mathcal{C}; \mathcal{L}; \Phi \vdash \mathit{tpexpr} \Leftarrow \mathit{ident}{:}\beta.\, \mathit{term} \\[1em]
\mathit{action\_jtype} & ::= & \\
 & | & \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \mathit{mem\_action} \Rightarrow \mathit{ret} \\
 & | & \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \mathit{tval} \Leftarrow \mathit{ret} \\
 & | & \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \mathit{seq\_expr} \Rightarrow \mathit{ret} \\
 & | & \mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \mathit{is\_expr} \Rightarrow \mathit{ret} \\[1em]
\mathit{judgement} & ::= & \\
 & | & \mathit{object\_value\_jtype} \\
 & | & \mathit{pval\_jtype} \\
 & | & \mathit{resource\_jtype} \\
 & | & \mathit{spine\_jtype} \\
 & | & \mathit{pexpr\_jtype} \\
 & | & \mathit{pattern\_jtype} \\
 & | & \mathit{tpval\_jtype} \\
 & | & \mathit{tpexpr\_jtype} \\
 & | & \mathit{action\_jtype}
\end{array}
$$

$$
\begin{array}{lcl}
\mathit{user\_syntax} & ::= & \\
& | & \mathit{ident} \\
& | & n \\
& | & \mathit{impl\_const} \\
& | & \mathit{mem\_int} \\
& | & \mathit{member} \\
& | & \\
& | & \mathit{nat} \\
& | & \mathit{mem\_ptr} \\
& | & \mathit{mem\_val} \\
& | & \\
& | & \mathit{mem\_iv\_c} \\
& | & \mathit{UB\_name} \\
& | & \mathit{string} \\
& | & \\
& | & \\
& | & \mathit{mem\_order} \\
& | & \mathit{linux\_mem\_order} \\
& | & \mathit{logical\_val} \\
& | & \\
& | & \mathit{Sctypes\_t} \\
& | & \mathit{tag} \\
& | & \beta \\
& | & \mathit{binop} \\
& | & \mathit{binop}_{\mathit{arith}} \\
& | & \mathit{binop}_{\mathit{rel}} \\
& | & \mathit{binop}_{\mathit{bool}} \\
& | & \mathit{ident} \\
& | & \tau
\end{array}
$$

$$
\begin{array}{ll}
| & \mathit{ident} \\
| & \mathit{object\_value} \\
| & \mathit{loaded\_value} \\
| & \beta \\
| & \mathit{value} \\
| & \mathit{ctor\_val} \\
| & \mathit{ctor\_expr} \\
| & \tau \\
| & \mathit{name} \\
| & \mathit{pval} \\
| & \mathit{pval} \\
| & \mathit{pexpr} \\
| & \mathit{pexpr} \\
| & \mathit{tpval} \\
| & \mathit{tpval} \\
| & \mathit{ident\_opt\_}\beta \\
| & \mathit{pattern} \\
| & \mathit{pattern} \\
| & \mathit{ident\_or\_pattern} \\
| & \mathit{tpexpr} \\
| & \mathit{tpexpr} \\
| & \mathit{m\_kill\_kind} \\
| & \mathit{bool} \\
| & \mathit{int} \\
| & \mathit{mem\_action} \\
| & \mathit{mem\_action} \\
| & \mathit{polarity} \\
| & \mathit{pol\_mem\_action} \\
| & \mathit{mem\_op}
\end{array}
$$

$$
\begin{array}{ll}
| & \mathit{spine\_elem} \\
| & \mathit{tval} \\
| & \mathit{tval} \\
| & \mathit{bool\_op} \\
| & \mathit{arith\_op} \\
| & \mathit{cmp\_op} \\
| & \mathit{list\_op} \\
| & \mathit{tuple\_op} \\
| & \mathit{pointer\_op} \\
| & \beta \\
| & \mathit{option\_op} \\
| & \mathit{array\_op} \\
| & \mathit{param\_op} \\
| & \mathit{struct\_op} \\
| & \mathit{ct\_pred} \\
| & \mathit{term} \\
| & \mathit{term} \\
| & \mathit{term} \\
| & \mathit{terms} \\
| & \mathit{predicate\_name} \\
| & \mathit{init} \\
| & \mathit{predicate} \\
| & \mathit{resource} \\
| & \mathit{res\_term} \\
| & \mathit{ret\_pattern} \\
| & \mathit{seq\_expr} \\
| & \mathit{seq\_expr} \\
| & \mathit{seq\_texpr} \\
| & \mathit{seq\_texpr}
\end{array}
$$

$$
\begin{array}{ll}
| & \mathit{is\_expr} \\
| & \mathit{is\_expr} \\
| & \mathit{is\_texpr} \\
| & \mathit{is\_texpr} \\
| & \mathit{texpr} \\
| & \mathit{terminals} \\
| & z \\
| & \mathbb{Q} \\
| & \mathit{lit} \\
| & \mathit{arg} \\
| & \mathit{pure\_arg} \\
| & \mathit{ret} \\
| & \mathit{pure\_ret} \\
| & \mathcal{C} \\
| & \mathcal{L} \\
| & \Phi \\
| & \mathcal{R} \\
| & \mathit{formula}
\end{array}
$$

$\boxed{\mathcal{C}; \mathcal{L}; \Phi \vdash \mathit{object\_value} \Rightarrow \texttt{obj}\, \beta}$

$$\frac{}{\mathcal{C}; \mathcal{L}; \Phi \vdash \mathit{mem\_int} \Rightarrow \texttt{obj}\, \texttt{integer}} \quad \textsc{Pval\_Obj\_Int}$$

$$\frac{}{\mathcal{C}; \mathcal{L}; \Phi \vdash \mathit{mem\_ptr} \Rightarrow \texttt{obj}\, \texttt{loc}} \quad \textsc{Pval\_Obj\_Ptr}$$

$$\frac{\overline{\mathcal{C}; \mathcal{L}; \Phi \vdash \mathit{loaded\_value}_i \Rightarrow \beta}^{\,i}}{\mathcal{C}; \mathcal{L}; \Phi \vdash \texttt{array}\, (\, \overline{\mathit{loaded\_value}_i}^{\,i}\,) \Rightarrow \texttt{obj}\, \texttt{array}\, \beta} \quad \textsc{Pval\_Obj\_Arr}$$

$$\frac{\begin{array}{c} \mathit{ident}\colon \mathtt{struct}\, \mathit{tag}\ \&\ \overline{\mathit{member}_i{:}\tau_i}^{\,i}\ \in\ \mathtt{Globals} \\ \overline{\mathcal{C};\mathcal{L};\Phi \vdash \mathit{mem\_val}_i \Rightarrow\ \mathtt{mem}\,\beta_i}^{\,i} \end{array}}{\mathcal{C};\mathcal{L};\Phi \vdash (\,\mathtt{struct}\,\mathit{tag})\{\,\overline{.\,\mathit{member}_i{:}\tau_i = \mathit{mem\_val}_i}^{\,i}\,\} \Rightarrow\ \mathtt{obj}\,\mathtt{struct}\,\mathit{tag}} \quad \textsc{Pval\_Obj\_Struct}$$

$\boxed{\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval} \Rightarrow \beta}$

$$\frac{x{:}\beta\ \in\ \mathcal{C}}{\mathcal{C};\mathcal{L};\Phi \vdash x \Rightarrow \beta} \quad \textsc{Pval\_Var}$$

$$\frac{\mathcal{C};\mathcal{L};\Phi \vdash \mathit{object\_value} \Rightarrow\ \mathtt{obj}\,\beta}{\mathcal{C};\mathcal{L};\Phi \vdash \mathit{object\_value} \Rightarrow \beta} \quad \textsc{Pval\_Obj}$$

$$\frac{\mathcal{C};\mathcal{L};\Phi \vdash \mathit{object\_value} \Rightarrow\ \mathtt{obj}\,\beta}{\mathcal{C};\mathcal{L};\Phi \vdash \mathtt{specified}\,\mathit{object\_value} \Rightarrow \beta} \quad \textsc{Pval\_Loaded}$$

$$\frac{}{\mathcal{C};\mathcal{L};\Phi \vdash \mathtt{Unit} \Rightarrow \mathtt{unit}} \quad \textsc{Pval\_Unit}$$

$$\frac{}{\mathcal{C};\mathcal{L};\Phi \vdash \mathtt{True} \Rightarrow \mathtt{bool}} \quad \textsc{Pval\_True}$$

$$\frac{}{\mathcal{C};\mathcal{L};\Phi \vdash \mathtt{False} \Rightarrow \mathtt{bool}} \quad \textsc{Pval\_False}$$

$$\frac{\overline{\mathcal{C};\mathcal{L};\Phi \vdash \mathit{value}_i \Rightarrow \beta}^{\,i}}{\mathcal{C};\mathcal{L};\Phi \vdash \beta[\,\overline{\mathit{value}_i}^{\,i}\,] \Rightarrow \mathtt{list}\,\beta} \quad \textsc{Pval\_List}$$

$$\frac{\overline{\mathcal{C};\mathcal{L};\Phi \vdash \mathit{value}_i \Rightarrow \beta_i}^{\,i}}{\mathcal{C};\mathcal{L};\Phi \vdash (\,\overline{\mathit{value}_i}^{\,i}\,) \Rightarrow \overline{\beta_i}^{\,i}} \quad \textsc{Pval\_Tuple}$$

$$\frac{\texttt{smt}\,(\Phi \Rightarrow \texttt{false})}{\mathcal{C};\mathcal{L};\Phi \vdash \texttt{error}\,(\mathit{string},\mathit{pval}) \Rightarrow \beta} \quad \textsc{Pval\_Error}$$

$$\frac{}{\mathcal{C};\mathcal{L};\Phi \vdash \texttt{Nil}\,\beta(\,) \Rightarrow \texttt{list}\,\beta} \quad \textsc{Pval\_Ctor\_Nil}$$

$$\frac{\begin{array}{c}\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval}_1 \Rightarrow \beta\\ \mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval}_2 \Rightarrow \texttt{list}\,\beta\end{array}}{\mathcal{C};\mathcal{L};\Phi \vdash \texttt{Cons}(\mathit{pval}_1,\mathit{pval}_2) \Rightarrow \texttt{list}\,\beta} \quad \textsc{Pval\_Ctor\_Cons}$$

$$\frac{\overline{\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval}_i \Rightarrow \beta_i}^{\,i}}{\mathcal{C};\mathcal{L};\Phi \vdash \texttt{Tuple}(\,\overline{\mathit{pval}_i}^{\,i}\,) \Rightarrow \overline{\beta_i}^{\,i}} \quad \textsc{Pval\_Ctor\_Tuple}$$

$$\frac{\overline{\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval}_i \Rightarrow \beta}^{\,i}}{\mathcal{C};\mathcal{L};\Phi \vdash \texttt{Array}(\,\overline{\mathit{pval}_i}^{\,i}\,) \Rightarrow \texttt{array}\,\beta} \quad \textsc{Pval\_Ctor\_Array}$$

$$\frac{\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval} \Rightarrow \beta}{\mathcal{C};\mathcal{L};\Phi \vdash \texttt{Specified}(\mathit{pval}) \Rightarrow \beta} \quad \textsc{Pval\_Ctor\_Specified}$$

$$\frac{\overline{\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval}_i \Rightarrow \beta_i}^{\,i}}{\mathcal{C};\mathcal{L};\Phi \vdash (\,\texttt{struct}\,\mathit{tag})\{\,\overline{.\,\mathit{member}_i = \mathit{pval}_i}^{\,i}\,\} \Rightarrow \texttt{struct}\,\mathit{tag}} \quad \textsc{Pval\_Struct}$$

$\boxed{\mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \mathit{res\_term} \Leftarrow \mathit{resource}}$

$$\frac{}{\mathcal{C};\mathcal{L};\Phi;\cdot \vdash \texttt{emp} \Leftarrow \texttt{emp}} \quad \textsc{Resource\_Emp}$$

$$\frac{\texttt{smt}\,(\Phi \Rightarrow \mathit{resource} = \mathit{resource}')}{\mathcal{C};\mathcal{L};\Phi;\cdot,r{:}\mathit{resource} \vdash r \Leftarrow \mathit{resource}'} \quad \textsc{Resource\_Var}$$

$$\frac{\begin{array}{c}\mathcal{C};\mathcal{L};\Phi;\mathcal{R}_1 \vdash \mathit{res\_term}_1 \Leftarrow \mathit{resource}_1 \\ \mathcal{C};\mathcal{L};\Phi;\mathcal{R}_2 \vdash \mathit{res\_term}_2 \Leftarrow \mathit{resource}_2\end{array}}{\mathcal{C};\mathcal{L};\Phi;\mathcal{R}_1,\mathcal{R}_2 \vdash \langle \mathit{res\_term}_1, \mathit{res\_term}_2 \rangle \Leftarrow \mathit{resource}_1 \star \mathit{resource}_2} \quad \textsc{Resource\_SepConj}$$

$$\frac{\begin{array}{c}\mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \mathit{res\_term}_1 \Leftarrow \mathit{resource}_1 \\ \mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \mathit{res\_term}_2 \Leftarrow \mathit{resource}_2\end{array}}{\mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash (\mathit{res\_term}_1, \mathit{res\_term}_2) \Leftarrow \mathit{resource}_1 \wedge \mathit{resource}_2} \quad \textsc{Resource\_Conj}$$

$$\frac{\begin{array}{l}\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval} \Rightarrow \beta \\ \mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \mathit{res\_term}_2 \Leftarrow [\mathit{pval}/y]\mathit{resource}\end{array}}{\mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \mathtt{pack}\,(\mathit{pval}, \mathit{res\_term}_2) \Leftarrow \exists\, y{:}\beta.\ \mathit{resource}} \quad \textsc{Resource\_Pack}$$

$$\boxed{\mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \overline{\mathit{spine\_elem}_i}^{\,i} :: \mathit{arg} \gg \mathit{ret}}$$

$$\frac{}{\mathcal{C};\mathcal{L};\Phi;\cdot \vdash\ :: \mathit{ret} \gg \mathit{ret}} \quad \textsc{Spine\_Empty}$$

$$\frac{\begin{array}{l}\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval} \Rightarrow \beta \\ \mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \overline{\mathit{spine\_elem}_i}^{\,i} :: [\mathit{pval}/x]\mathit{arg} \gg \mathit{ret}\end{array}}{\mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \mathit{pval},\ \overline{\mathit{spine\_elem}_i}^{\,i} :: \Pi\, x{:}\beta.\ \mathit{arg} \gg \mathit{ret}} \quad \textsc{Spine\_Computational}$$

$$\frac{\begin{array}{l}\mathcal{L} \vdash \mathit{logical\_val}{:}\beta \\ \mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \overline{\mathit{spine\_elem}_i}^{\,i} :: [\mathit{logical\_val}/x]\mathit{arg} \gg \mathit{ret}\end{array}}{\mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \mathit{logical\_val},\ \overline{\mathit{spine\_elem}_i}^{\,i} :: \forall\, x{:}\beta.\ \mathit{arg} \gg \mathit{ret}} \quad \textsc{Spine\_Logical}$$

$$\frac{\begin{array}{l}\mathcal{C};\mathcal{L};\Phi;\mathcal{R}_1 \vdash \mathit{res\_term} \Leftarrow \mathit{resource} \\ \mathtt{smt}\,(\Phi \Rightarrow \mathit{resource} = \mathit{resource}') \\ \mathcal{C};\mathcal{L};\Phi;\mathcal{R}_2 \vdash \overline{\mathit{spine\_elem}_i}^{\,i} :: \mathit{arg} \gg \mathit{ret}\end{array}}{\mathcal{C};\mathcal{L};\Phi;\mathcal{R}_1,\mathcal{R}_2 \vdash \mathit{res\_term},\ \overline{\mathit{spine\_elem}_i}^{\,i} :: \mathit{resource}' \multimap \mathit{arg} \gg \mathit{ret}} \quad \textsc{Spine\_Resource}$$

$$\dfrac{\begin{array}{l}\texttt{smt}\,(\Phi \Rightarrow \mathit{term}) \\ \mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \overline{\mathit{spine\_elem}_i}^{\,i} :: \mathit{arg} \gg \mathit{ret}\end{array}}{\mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \overline{\mathit{spine\_elem}_i}^{\,i} :: \mathit{term} \supset \mathit{arg} \gg \mathit{ret}} \quad \textsc{Spine\_Constraint}$$

$\boxed{\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pexpr} \Rightarrow \mathit{ident}{:}\beta.\,\mathit{term}}$

$$\dfrac{\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval} \Rightarrow \beta}{\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval} \Rightarrow y{:}\beta.\; y = \mathit{pval}} \quad \textsc{PExpr\_Val}$$

$$\dfrac{\begin{array}{l}\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval}_1 \Rightarrow \texttt{loc} \\ \mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval}_2 \Rightarrow \texttt{integer}\end{array}}{\mathcal{C};\mathcal{L};\Phi \vdash \texttt{array\_shift}\,(\mathit{pval}_1, \tau, \mathit{pval}_2) \Rightarrow y{:}\texttt{loc}.\; y = \mathit{pval}_1 +_{\mathrm{ptr}} (\mathit{pval}_2 \times \mathrm{size\_of}(\tau))} \quad \textsc{PExpr\_Array\_Shift}$$

$$\dfrac{\begin{array}{l}\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval} \Rightarrow \\ \_'{:}\,\texttt{struct}\,\mathit{tag}\ \&\,\overline{\mathit{member}_i{:}\tau_i}^{\,i} \in \texttt{Globals}\end{array}}{\mathcal{C};\mathcal{L};\Phi \vdash \texttt{member\_shift}\,(\mathit{pval}, \mathit{tag}, \mathit{member}_j) \Rightarrow y{:}\texttt{loc}.\; y = \mathit{pval} +_{\mathrm{ptr}} \mathrm{offset\_of}_{\mathit{tag}}(\mathit{member}_j)} \quad \textsc{PExpr\_Member\_Shift}$$

$$\dfrac{\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval} \Rightarrow \texttt{bool}}{\mathcal{C};\mathcal{L};\Phi \vdash \texttt{not}\,(\mathit{pval}) \Rightarrow y{:}\texttt{bool}.\; y = \neg\,\mathit{pval}} \quad \textsc{PExpr\_Not}$$

$$\dfrac{\begin{array}{l}\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval}_1 \Rightarrow \texttt{integer} \\ \mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval}_2 \Rightarrow \texttt{integer}\end{array}}{\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval}_1\ \mathit{binop}_{\mathit{arith}}\ \mathit{pval}_2 \Rightarrow y{:}\texttt{integer}.\; y = (\mathit{pval}_1\ \mathit{binop}_{\mathit{arith}}\ \mathit{pval}_2)} \quad \textsc{PExpr\_Arith\_Binop}$$

$$\dfrac{\begin{array}{l}\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval}_1 \Rightarrow \texttt{integer} \\ \mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval}_2 \Rightarrow \texttt{integer}\end{array}}{\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval}_1\ \mathit{binop}_{\mathit{rel}}\ \mathit{pval}_2 \Rightarrow y{:}\texttt{bool}.\; y = (\mathit{pval}_1\ \mathit{binop}_{\mathit{rel}}\ \mathit{pval}_2)} \quad \textsc{PExpr\_Rel\_Binop}$$

$$\frac{\begin{array}{c}\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval}_1 \Rightarrow \mathtt{bool}\\ \mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval}_2 \Rightarrow \mathtt{bool}\end{array}}{\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval}_1\ \mathit{binop}_{\mathit{bool}}\ \mathit{pval}_2 \Rightarrow y{:}\mathtt{bool}.\ y = (\mathit{pval}_1\ \mathit{binop}_{\mathit{bool}}\ \mathit{pval}_2)}\ \textsc{PExpr\_Bool\_Binop}$$

$$\frac{\begin{array}{l}\mathcal{C} \vdash \mathit{name} \Rightarrow \mathit{pure\_arg}\\ \mathcal{C};\mathcal{L};\Phi;\cdot \vdash \overline{\mathit{pval}_i}^{\,i} :: \mathit{pure\_arg} \gg \Sigma\, y'{:}\beta'.\ \mathit{term}' \wedge \mathtt{I}\end{array}}{\mathcal{C};\mathcal{L};\Phi \vdash \mathit{name}(\overline{\mathit{pval}_i}^{\,i}) \Rightarrow y'{:}\beta'.\ \mathit{term}'}\ \textsc{PExpr\_Call}$$

$$\frac{\begin{array}{l}\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval} \Rightarrow \mathtt{bool}\\ \mathtt{smt}\,(\Phi \Rightarrow \mathit{pval})\end{array}}{\mathcal{C};\mathcal{L};\Phi \vdash \mathtt{assert\_undef}\,(\mathit{pval},\ \mathit{UB\_name}) \Rightarrow y{:}\mathtt{unit}.\ y = \mathtt{unit}}\ \textsc{PExpr\_Assert\_Undef}$$

$$\frac{\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval} \Rightarrow \mathtt{bool}}{\mathcal{C};\mathcal{L};\Phi \vdash \mathtt{bool\_to\_integer}\,(\mathit{pval}) \Rightarrow y{:}\mathtt{integer}.\ y = \mathtt{if}\ \mathit{pval}\ \mathtt{then}\ 1\ \mathtt{else}\ 0}\ \textsc{PExpr\_Bool\_To\_Integer}$$

$$\frac{\begin{array}{l}\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval} \Rightarrow \mathtt{integer}\\ \mathit{abbrev}_1 \equiv \mathrm{max\_int}_\tau - \mathrm{min\_int}_\tau + 1\\ \mathit{abbrev}_2 \equiv \mathit{pval}\ \mathtt{rem\_f}\ \mathit{abbrev}_1\end{array}}{\mathcal{C};\mathcal{L};\Phi \vdash \mathtt{wrapI}\,(\tau, \mathit{pval}) \Rightarrow y'{:}\beta.\ y = \mathtt{if}\ \mathit{abbrev}_2 \leq \mathrm{max\_int}_\tau\ \mathtt{then}\ \mathit{abbrev}_2\ \mathtt{else}\ \mathit{abbrev}_2 - \mathit{abbrev}_1}\ \textsc{PExpr\_WrapI}$$

$$\boxed{\mathit{term}\ \mathtt{as}\ \mathit{pattern}{:}\beta \leadsto \mathcal{C};\Phi}$$

$$\frac{}{\mathit{term}\ \mathtt{as}\ \_{:}\beta{:}\beta \leadsto \cdot;\cdot}\ \textsc{Comp\_Pattern\_No\_Sym\_Annot}$$

$$\frac{}{\mathit{term}\ \mathtt{as}\ x{:}\beta{:}\beta \leadsto \cdot, x{:}\beta; \cdot, x = \mathit{term}}\ \textsc{Comp\_Pattern\_Sym\_Annot}$$

$$\frac{}{\mathit{term}\ \mathtt{as}\ \mathtt{Nil}\ \beta(\,){:}\mathtt{list}\ \beta \leadsto \cdot;\cdot}\ \textsc{Comp\_Pattern\_Nil}$$

$$\frac{\begin{array}{c} \mathit{term}^{(1)}\ \mathtt{as}\ \mathit{pattern}_1{:}\beta \rightsquigarrow \mathcal{C}_1; \Phi_1 \\ \mathtt{tl}\ \mathit{term}\ \mathtt{as}\ \mathit{pattern}_2{:}\mathtt{list}\ \beta \rightsquigarrow \mathcal{C}_2; \Phi_1 \end{array}}{\mathit{term}\ \mathtt{as}\ \mathtt{Cons}(\mathit{pattern}_1, \mathit{pattern}_2){:}\mathtt{list}\ \beta \rightsquigarrow \mathcal{C}_1, \mathcal{C}_2; \Phi_1, \Phi_2} \quad \textsc{Comp\_Pattern\_Cons}$$

$$\frac{\overline{\mathit{term}^{(i)}\ \mathtt{as}\ \mathit{pattern}_i{:}\beta_i \rightsquigarrow \mathcal{C}_i; \Phi_i}^{\,i}}{\mathit{term}\ \mathtt{as}\ \mathtt{Tuple}(\ \overline{\mathit{pattern}_i}^{\,i}\ ){:}\overline{\beta_i}^{\,i} \rightsquigarrow \overline{\mathcal{C}_i}^{\,i}; \overline{\Phi_i}^{\,i}} \quad \textsc{Comp\_Pattern\_Tuple}$$

$$\frac{\overline{\mathit{term}[i]\ \mathtt{as}\ \mathit{pattern}_i{:}\beta \rightsquigarrow \mathcal{C}_i; \Phi_i}^{\,i}}{\mathit{term}\ \mathtt{as}\ \mathtt{Array}(\ \overline{\mathit{pattern}_i}^{\,i}\ ){:}\mathtt{array}\ \beta \rightsquigarrow \overline{\mathcal{C}_i}^{\,i}; \overline{\Phi_i}^{\,i}} \quad \textsc{Comp\_Pattern\_Array}$$

$$\frac{\mathit{term}\ \mathtt{as}\ \mathit{pattern}{:}\beta \rightsquigarrow \mathcal{C}; \Phi}{\mathit{term}\ \mathtt{as}\ \mathtt{Specified}(\mathit{pattern}){:}\beta \rightsquigarrow \mathcal{C}; \Phi} \quad \textsc{Comp\_Pattern\_Specified}$$

$\boxed{\mathit{term}\ \mathtt{as}\ \mathit{ident\_or\_pattern}{:}\beta \rightsquigarrow \mathcal{C}; \Phi}$

$$\frac{}{\mathit{term}\ \mathtt{as}\ x{:}\beta \rightsquigarrow \cdot, x{:}\beta; \cdot, x = \mathit{term}} \quad \textsc{Sym\_Or\_PatternSym}$$

$$\frac{\mathit{term}\ \mathtt{as}\ \mathit{pattern}{:}\beta \rightsquigarrow \mathcal{C}; \Phi}{\mathit{term}\ \mathtt{as}\ \mathit{pattern}{:}\beta \rightsquigarrow \mathcal{C}; \Phi} \quad \textsc{Sym\_Or\_PatternPattern}$$

$\boxed{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash\ \overline{\mathit{ret\_pattern}_i}^{\,i}\ {:}\mathit{ret} \rightsquigarrow \mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}'}$

$$\frac{}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash\ {:}\mathtt{I} \rightsquigarrow \cdot; \cdot; \cdot; \cdot} \quad \textsc{Ret\_Pattern\_Empty}$$

$$\frac{\mathcal{C}, y{:}\beta; \mathcal{L}; \Phi; \mathcal{R} \vdash\ \overline{\mathit{ret\_pattern}_i}^{\,i}\ {:}\mathit{ret} \rightsquigarrow \mathcal{C}'; \mathcal{L}'; \Phi'; \mathcal{R}'}{\mathcal{C}; \mathcal{L}; \Phi; \mathcal{R} \vdash \mathtt{comp}\ y,\ \overline{\mathit{ret\_pattern}_i}^{\,i}\ {:}\Sigma\, y{:}\beta.\ \mathit{ret} \rightsquigarrow \mathcal{C}, y{:}\beta; \mathcal{L}'; \Phi'; \mathcal{R}'} \quad \textsc{Ret\_Pattern\_Computational}$$

$$\frac{\mathcal{C};\mathcal{L},y{:}\beta;\Phi;\mathcal{R}\vdash\overline{\mathit{ret\_pattern}_i}^{\,i}{:}\mathit{ret}\leadsto\mathcal{C}';\mathcal{L}';\Phi';\mathcal{R}'}{\mathcal{C};\mathcal{L};\Phi;\mathcal{R}\vdash\mathtt{log}\,y,\,\overline{\mathit{ret\_pattern}_i}^{\,i}{:}\exists\,y{:}\beta.\,\mathit{ret}\leadsto\mathcal{C};\mathcal{L}',y{:}\beta;\Phi';\mathcal{R}'}\quad\textsc{Ret\_Pattern\_Logical}$$

$$\frac{\mathcal{C};\mathcal{L};\Phi;\mathcal{R},y{:}\mathit{resource}\vdash\overline{\mathit{ret\_pattern}_i}^{\,i}{:}\mathit{ret}\leadsto\mathcal{C}';\mathcal{L}';\Phi';\mathcal{R}'}{\mathcal{C};\mathcal{L};\Phi;\mathcal{R}\vdash\mathtt{res}\,y,\,\overline{\mathit{ret\_pattern}_i}^{\,i}{:}\mathit{resource}\star\mathit{ret}\leadsto\mathcal{C};\mathcal{L}';\Phi';\mathcal{R}',y{:}\mathit{resource}}\quad\textsc{Ret\_Pattern\_Resource}$$

$$\frac{\mathcal{C};\mathcal{L};\Phi,\mathit{term};\mathcal{R}\vdash\overline{\mathit{ret\_pattern}_i}^{\,i}{:}\mathit{ret}\leadsto\mathcal{C}';\mathcal{L}';\Phi';\mathcal{R}'}{\mathcal{C};\mathcal{L};\Phi;\mathcal{R}\vdash\overline{\mathit{ret\_pattern}_i}^{\,i}{:}\mathit{term}\wedge\mathit{ret}\leadsto\mathcal{C};\mathcal{L}';\Phi',\mathit{term};\mathcal{R}'}\quad\textsc{Ret\_Pattern\_Constraint}$$

$\boxed{\mathcal{C};\mathcal{L};\Phi\vdash\mathit{tpval}\Leftarrow\mathit{ident}{:}\beta.\,\mathit{term}}$

$$\frac{\mathtt{smt}\,(\Phi\Rightarrow\mathtt{false})}{\mathcal{C};\mathcal{L};\Phi\vdash\mathtt{undef}\ \mathit{UB\_name}\Leftarrow y{:}\beta.\,\mathit{term}}\quad\textsc{TPVal\_Undef}$$

$$\frac{\begin{array}{c}\mathcal{C};\mathcal{L};\Phi\vdash\mathit{pval}\Rightarrow\beta\\ \mathtt{smt}\,(\Phi\Rightarrow\mathit{term})\end{array}}{\mathcal{C};\mathcal{L};\Phi\vdash\mathtt{done}\,\mathit{pval}\Leftarrow y{:}\beta.\,\mathit{term}}\quad\textsc{TPVal\_Done}$$

$\boxed{\mathcal{C};\mathcal{L};\Phi\vdash\mathit{tpexpr}\Leftarrow\mathit{ident}{:}\beta.\,\mathit{term}}$

$$\frac{\begin{array}{l}\mathcal{C};\mathcal{L};\Phi\vdash\mathit{pval}\Rightarrow\mathtt{bool}\\ \mathcal{C};\mathcal{L};\Phi,\mathit{pval}=\mathtt{true}\vdash\mathit{tpexpr}_1\Leftarrow y{:}\beta.\,\mathit{term}\\ \mathcal{C};\mathcal{L};\Phi,\mathit{pval}=\mathtt{false}\vdash\mathit{tpexpr}_2\Leftarrow y{:}\beta.\,\mathit{term}\end{array}}{\mathcal{C};\mathcal{L};\Phi\vdash\mathtt{if}\,\mathit{pval}\,\mathtt{then}\,\mathit{tpexpr}_1\,\mathtt{else}\,\mathit{tpexpr}_2\Leftarrow y{:}\beta.\,\mathit{term}}\quad\textsc{TPExpr\_If}$$

$$\frac{\begin{array}{l}\mathcal{C};\mathcal{L};\Phi\vdash\mathit{pexpr}\Rightarrow y_1{:}\beta_1.\,\mathit{term}_1\\ y_1\,\mathtt{as}\,\mathit{ident\_or\_pattern}{:}\beta_1\leadsto\mathcal{C}_1;\Phi_1\\ \mathcal{C},\mathrm{fresh}(\mathcal{C}_1);\mathcal{L},y_1{:}\beta_1;\Phi,\mathit{term}_1,[\mathrm{fresh}(\mathcal{C}_1)/\mathcal{C}_1]\Phi_1\vdash[\mathrm{fresh}(\mathcal{C}_1)/\mathcal{C}_1]\mathit{tpexpr}\Leftarrow y_2{:}\beta_2.\,\mathit{term}_2\end{array}}{\mathcal{C};\mathcal{L};\Phi\vdash\mathtt{let}\,\mathit{ident\_or\_pattern}=\mathit{pexpr}\,\mathtt{in}\,\mathit{tpexpr}\Leftarrow y_2{:}\beta_2.\,\mathit{term}_2}\quad\textsc{TPExpr\_Let}$$

$$\frac{\begin{array}{l}\mathcal{C};\mathcal{L};\Phi \vdash pval \Rightarrow \beta_1 \\ \overline{y_1\ \mathtt{as}\ pattern_i{:}\beta_1 \rightsquigarrow \mathcal{C}_i;\Phi_i}^{\,i} \\ \overline{\mathcal{C},\mathrm{fresh}(\mathcal{C}_i);\mathcal{L},y_1{:}\beta_1;\Phi,y_1 = pval,[\mathrm{fresh}(\mathcal{C}_i)/\mathcal{C}_i]\Phi_i \vdash [\mathrm{fresh}(\mathcal{C}_i)/\mathcal{C}_i]tpexpr_i \Leftarrow y_2{:}\beta_2.\ term_2}^{\,i}\end{array}}{\mathcal{C};\mathcal{L};\Phi \vdash \mathtt{case}\ pval\ \mathtt{of}\ \overline{\mid pattern_i \Rightarrow tpexpr_i}^{\,i}\ \mathtt{end} \Leftarrow y_2{:}\beta_2.\ term_2}\ \textsc{TPExpr\_Case}$$

$\boxed{\mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash mem\_action \Rightarrow ret}$

$$\frac{\mathcal{C};\mathcal{L};\Phi \vdash pval \Rightarrow \mathtt{integer}}{\mathcal{C};\mathcal{L};\Phi;\cdot \vdash \mathtt{create}\,(pval,\tau) \Rightarrow \Sigma\, y_p{:}\mathtt{loc}.\ \exists\, y{:}\beta_\tau.\ \mathtt{representable}\,(\tau*,y_p) \wedge \mathtt{alignedI}\,(pval,y_p) \wedge [y_p]\ 1\overset{\times}{\mapsto}_\tau\ [y] \star \mathtt{I}}\ \textsc{Action\_Create}$$

$$\frac{\begin{array}{l}\mathcal{C};\mathcal{L};\Phi \vdash pval_1 \Rightarrow \mathtt{loc} \\ \mathcal{C};\mathcal{L};\Phi \vdash pval_2 \Rightarrow \beta_\tau \\ \mathtt{smt}\,(\Phi \Rightarrow \mathtt{representable}\,(\tau,pval_2)) \\ \mathtt{smt}\,(\Phi \Rightarrow pval_0 = pval_1)\end{array}}{\mathcal{C};\mathcal{L};\Phi;\cdot,r{:}[pval_0]\ 1{\mapsto_\tau}\ [\_] \vdash \mathtt{store}\,(\_,\tau,pval_1,pval_2,\_) \Rightarrow \Sigma\,\_{:}\mathtt{unit}.\ [pval_0]\ 1\overset{\checkmark}{\mapsto}_\tau\ [pval_2] \star \mathtt{I}}\ \textsc{Action\_Store}$$

$$\frac{\begin{array}{c}\mathcal{C};\mathcal{L};\Phi \vdash pval_1 \Rightarrow \mathtt{loc} \\ \mathtt{smt}\,(\Phi \Rightarrow pval_0 = pval_1)\end{array}}{\mathcal{C};\mathcal{L};\Phi;\cdot,r{:}[pval_0]\ 1{\mapsto_\tau}\ [\_] \vdash \mathtt{kill}\,(\mathtt{static}\,\tau,pval_1) \Rightarrow \Sigma\,\_{:}\mathtt{unit}.\ \mathtt{I}}\ \textsc{Action\_Kill\_Static}$$

$\boxed{\mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash tval \Leftarrow ret}$

$$\frac{}{\mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \mathtt{done}\ \Leftarrow \mathtt{I}}\ \textsc{TVal\_I}$$

$$\frac{\begin{array}{l}\mathcal{C};\mathcal{L};\Phi \vdash pval \Rightarrow \beta \\ \mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \mathtt{done}\ \overline{spine\_elem_i}^{\,i} \Leftarrow [pval/y]ret\end{array}}{\mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \mathtt{done}\ pval,\ \overline{spine\_elem_i}^{\,i} \Leftarrow \Sigma\, y{:}\beta.\ ret}\ \textsc{TVal\_Computational}$$

$$\frac{\begin{array}{l}\mathcal{L} \vdash \mathit{logical\_val}{:}\beta \\ \mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \mathtt{done}\,\overline{\mathit{spine\_elem}_i}^{\,i} \Leftarrow [\mathit{logical\_val}/y]\mathit{ret}\end{array}}{\mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \mathtt{done}\,\mathit{logical\_val},\ \overline{\mathit{spine\_elem}_i}^{\,i} \Leftarrow \exists\, y{:}\beta.\ \mathit{ret}} \quad \textsc{TVal\_Logical}$$

$$\frac{\begin{array}{l}\mathtt{smt}\,(\Phi \Rightarrow \mathit{term}) \\ \mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \mathtt{done}\,\overline{\mathit{spine\_elem}_i}^{\,i} \Leftarrow \mathit{ret}\end{array}}{\mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \mathtt{done}\,\overline{\mathit{spine\_elem}_i}^{\,i} \Leftarrow \mathit{term} \wedge \mathit{ret}} \quad \textsc{TVal\_Constraint}$$

$$\frac{\begin{array}{l}\mathcal{C};\mathcal{L};\Phi;\mathcal{R}_1 \vdash \mathit{res\_term} \Leftarrow \mathit{resource} \\ \mathcal{C};\mathcal{L};\Phi;\mathcal{R}_2 \vdash \mathtt{done}\,\overline{\mathit{spine\_elem}_i}^{\,i} \Leftarrow \mathit{ret}\end{array}}{\mathcal{C};\mathcal{L};\Phi;\mathcal{R}_1,\mathcal{R}_2 \vdash \mathtt{done}\,\mathit{res\_term},\ \overline{\mathit{spine\_elem}_i}^{\,i} \Leftarrow \mathit{resource} \star \mathit{ret}} \quad \textsc{TVal\_Resource}$$

$$\frac{\mathtt{smt}\,(\Phi \Rightarrow \mathtt{false})}{\mathcal{C};\mathcal{L};\Phi;\cdot \vdash \mathtt{undef}\ \mathit{UB\_name} \Leftarrow \mathit{ret}} \quad \textsc{TVal\_UB}$$

$\boxed{\mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \mathit{seq\_expr} \Rightarrow \mathit{ret}}$

$$\frac{\mathcal{C};\mathcal{L};\Phi \vdash \mathit{pval} \Rightarrow \beta}{\mathcal{C};\mathcal{L};\Phi;\cdot \vdash \mathit{pval} \Rightarrow \Sigma\, y{:}\beta.\ y = \mathit{pval} \wedge \mathtt{I}} \quad \textsc{Seq\_Expr\_Pure}$$

$\boxed{\mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \mathit{is\_expr} \Rightarrow \mathit{ret}}$

$$\frac{\mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \mathit{mem\_action} \Rightarrow \mathit{ret}}{\mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \mathtt{Pos}\,\mathit{mem\_action} \Rightarrow \mathit{ret}} \quad \textsc{IS\_Expr\_Action}$$

$$\frac{\mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \mathit{mem\_action} \Rightarrow \mathit{ret}}{\mathcal{C};\mathcal{L};\Phi;\mathcal{R} \vdash \mathtt{Neg}\,\mathit{mem\_action} \Rightarrow \mathit{ret}} \quad \textsc{IS\_Expr\_Neg\_Action}$$

```
Definition rules:         71 good    0 bad
Definition rule clauses: 163 good    0 bad
```