

# Languages-beta: SIMPLE-2-Expressions \*

The PPlanCompS Project

SIMPLE-2-Expressions.cbs | PLAIN | PRETTY

---

Language "SIMPLE"

## 2 Expressions

Syntax  $Exp : exp ::=$

- |  $( \text{exp} )$
- | value
- | lexp
- |  $lexp = \text{exp}$
- |  $++ \text{lexp}$
- |  $- \text{exp}$
- |  $\text{exp} ( \text{exp}^? )$
- |  $\text{sizeof} ( \text{exp} )$
- |  $\text{read} ( )$
- |  $\text{exp} + \text{exp}$
- |  $\text{exp} - \text{exp}$
- |  $\text{exp} * \text{exp}$
- |  $\text{exp} / \text{exp}$
- |  $\text{exp} \% \text{exp}$
- |  $\text{exp} < \text{exp}$
- |  $\text{exp} <= \text{exp}$
- |  $\text{exp} > \text{exp}$
- |  $\text{exp} >= \text{exp}$
- |  $\text{exp} == \text{exp}$
- |  $\text{exp} != \text{exp}$
- |  $! \text{exp}$
- |  $\text{exp} \&\& \text{exp}$
- |  $\text{exp} || \text{exp}$

Rule  $\llbracket ( \text{Exp} ) \rrbracket : exp = \llbracket \text{Exp} \rrbracket$

---

\*Suggestions for improvement: [plancomps@gmail.com](mailto:plancomps@gmail.com).  
Reports of issues: <https://github.com/plancomps/CBS-beta/issues>.

Semantics  $\text{rval}[\_ : \text{exp}] : \Rightarrow \text{values}$

Rule  $\text{rval}[V] = \text{val}[V]$

Rule  $\text{rval}[LE\text{xp}] = \text{assigned}(\text{lval}[LE\text{xp}])$

Rule  $\text{rval}[LE\text{xp} '=' Exp] =$   
     $\text{give}(\text{rval}[Exp],$   
         $\text{sequential}(\text{assign}(\text{lval}[LE\text{xp}], \text{given}),$   
             $\text{given}))$

Rule  $\text{rval}['++' LE\text{xp}] =$   
     $\text{give}(\text{lval}[LE\text{xp}],$   
         $\text{sequential}(\text{assign}(\text{given}, \text{integer-add}(\text{assigned}(\text{given}), 1)),$   
             $\text{assigned}(\text{given})))$

Rule  $\text{rval}['-' Exp] = \text{integer-negate}(\text{rval}[Exp])$

Rule  $\text{rval}[Exp '(' Exprs? ')'] = \text{apply}(\text{rval}[Exp], \text{tuple}(\text{rvals}[Exprs?]))$

Rule  $\text{rval}['sizeof' '(' Expr ')'] = \text{length}(\text{vector-elements}(\text{rval}[Exp]))$

Rule  $\text{rval}['\text{read}' '(' ')'] = \text{read}$

Rule  $\text{rval}[Exp_1 '+' Exp_2] = \text{integer-add}(\text{rval}[Exp_1], \text{rval}[Exp_2])$

Rule  $\text{rval}[Exp_1 '-' Exp_2] = \text{integer-subtract}(\text{rval}[Exp_1], \text{rval}[Exp_2])$

Rule  $\text{rval}[Exp_1 '*' Exp_2] = \text{integer-multiply}(\text{rval}[Exp_1], \text{rval}[Exp_2])$

Rule  $\text{rval}[Exp_1 '/' Exp_2] = \text{checked integer-divide}(\text{rval}[Exp_1], \text{rval}[Exp_2])$

Rule  $\text{rval}[Exp_1 \% Exp_2] = \text{checked integer-modulo}(\text{rval}[Exp_1], \text{rval}[Exp_2])$

Rule  $\text{rval}[Exp_1 '<' Exp_2] = \text{is-less}(\text{rval}[Exp_1], \text{rval}[Exp_2])$

Rule  $\text{rval}[Exp_1 '<=' Exp_2] = \text{is-less-or-equal}(\text{rval}[Exp_1], \text{rval}[Exp_2])$

Rule  $\text{rval}[Exp_1 '>' Exp_2] = \text{is-greater}(\text{rval}[Exp_1], \text{rval}[Exp_2])$

Rule  $\text{rval}[Exp_1 '>=' Exp_2] = \text{is-greater-or-equal}(\text{rval}[Exp_1], \text{rval}[Exp_2])$

Rule  $\text{rval}[Exp_1 '==' Exp_2] = \text{is-equal}(\text{rval}[Exp_1], \text{rval}[Exp_2])$

Rule  $\text{rval}[Exp_1 '!=' Exp_2] = \text{not}(\text{is-equal}(\text{rval}[Exp_1], \text{rval}[Exp_2]))$

Rule  $\text{rval}['!' Exp] = \text{not}(\text{rval}[Exp])$

Rule  $\text{rval}[Exp_1 '&\&' Exp_2] = \text{if-else}(\text{rval}[Exp_1], \text{rval}[Exp_2], \text{false})$

Rule  $\text{rval}[Exp_1 '||' Exp_2] = \text{if-else}(\text{rval}[Exp_1], \text{true}, \text{rval}[Exp_2])$

Syntax  $Exprs : \text{exps} ::= \text{exp} (' , ' \text{exps})?$

Semantics  $\text{rvals}[\_ : \text{exps?}] : (\Rightarrow \text{values})^*$

Rule  $\text{rvals}[] = ()$

Rule  $\text{rvals}[Exp] = \text{rval}[Exp]$

Rule  $\text{rvals}[Exp ' , ' Exprs] = \text{rval}[Exp], \text{rvals}[Exprs]$

Syntax  $LE\text{xp} : \text{lexp} ::= \text{id} \mid \text{lexp} '[' \text{exps} ']'$

Rule  $[[LE\text{xp} '[' Exp ' , ' Exprs ']' ] : \text{lexp} =$   
     $[[LE\text{xp} '[' Exp ']' ] '[' Exprs ']' ]$

Semantics  $\text{lval}[\_ : \text{lexp}] : \Rightarrow \text{variables}$

Rule  $\text{lval}[Id] = \text{bound}(\text{id}[Id])$

Rule  $\text{lval}[LE\text{xp} '[' Exp ']' ] =$   
     $\text{checked index}(\text{integer-add}(1, \text{rval}[Exp]), \text{vector-elements}(\text{rval}[LE\text{xp}]))$