# GLOSSARY OF NOTATION

Boolean

| | | |
|---|---|---|
| *Bool* | = {*false*, *true*} | 37 |
| ¬ | negation | 37 |
| ∧ | and | 37 |
| ∨ | or | 37 |
| ⊃ | implies | 37 |
| ≡ | equivalent to | 37 |
| ∀ | for all | 37 |
| ∃ | there exists | 38 |
| ∃! | there exists exactly one | 38 |
| Δ | unique description: $(\Delta x)(P(x))$: the unique $x$ such that $P(x)$; if non-existing or not unique, then undefined | |

Arithmetic

| | | |
|---|---|---|
| *Int* | = {... ,-2,-1,0,1,2, ...} | 39,76 |
| *Nat0* | = {0,1,2, ...} | 39,76 |
| *Nat* | = {1,2,3, ...} | 39,76 |

with the usual operators: +, -, *, ×, /, **, <, ≤, =, ≠, ≥, >, etc.; / is integer division, ** exponentiation

Quotation Values

| | | |
|---|---|---|
| *Quot* | Set of enumerated specification specific elementary objects, e.g. LABEL, AND, NULL, ... | 43,76 |
| = | equal to | |
| ≠ | different from | |

Token Values

| | | |
|---|---|---|
| *Token* | Set of specification specific elementary objects whose representation is not exposed. | 43 |
| = | equal to | |
| ≠ | different from | |

## Sets

39,79

| | | | |
|---|---|---|---|
| *-set* | Set forming operator; defines all finite subsets of given set. | | |
| $\{a_1, a_2, \ldots, a_n\}$ | Explicit enumeration | | |
| $\{a \mid P(a)\}$ | Implicit formation | | |
| $\{a \in Set \mid P(a)\}$ | Implicit formation | | |
| $\in$ | membership | | |
| $\neg\in$ | non-membership | | |
| $\cup$ | union | | |
| $\cap$ | intersection | | |
| $-$ | difference | (sometimes: $\backslash$ ) | |
| $\subset$ | proper inclusion | | |
| $\subseteq$ | inclusion | | |
| $=$ | equal to | | |
| $\neq$ | different from | | |
| *card* | cardinality | | |
| *union* | distributed union | | |

## Tuples (Lists)

41,80

| | | |
|---|---|---|
| * <br> + | Tuple (or list) forming operator; defines all finite tuples whose elements are from the given set, * generates empty tuple, + does not. | |
| $\langle a_1, a_2, \ldots a_n \rangle$ | Explicit enumeration | |
| $\langle f(i) \mid P(i) \rangle$ | Implicit formation - where order defined | |

| | | | |
|---|---|---|---|
| *hd* | head | (sometimes: *h* ) | 42 |
| *tl* | tail | (sometimes: *t* ) | 42 |
| [ ] | index | (rarely: ( ) ) | |
| *len* | length | (sometimes: *l* ) | 42 |
| *inds* | index set, indices | (sometimes: *ind* ) | 41 |
| *elems* | elements | | 41 |
| ^ | concatenation | | 42 |
| $=$ | equal to | | |
| $\neq$ | different from | | |
| *conc* | distributed concatenation | | 42 |

Maps

| | | |
|---|---|---|
| | | 40,80 |
| $\overrightarrow{m}$ | Map forming operator; defines all finite maps between given sets, $\overleftrightarrow{m}$ generates only one-to-one maps. | 40 |
| $\overleftrightarrow{m}$ | | 40 |
| $[a_1, a_2, \ldots, a_n]$ | Explicit enumeration | 40 |
| $[d \mapsto f(d) \mid P(d)]$ | Implicit formation | 40 |
| $(\ )$ | application | |
| $\circ$ | composition | 32 |
| $\cup$ | merge | 41 |
| $+$ | override extend (sometimes: $\dagger$ ) | 41 |
| $\backslash$ | remove (with) | 41 |
| $\mid$ | restrict to | 41 |
| *dom* | domain | |
| *rng* | co-domain, range | 41 |
| $=$ | equal to | |
| $\neq$ | different from | |
| *merge* | distributed merge | 41 |

Trees

| | | |
|---|---|---|
| :: | Tree forming operator; defines *mk-A(t)* trees, where *A* is the given left operand identifier, and *t* is any object denoted by the right operand domain expression. | 44,78 |
| $\times$ | Tree forming domain operator; defines cartesian product, un-named trees. | 66,67 |
| *mk-* | Named tree constructor function name prefix. $mk\text{-}A(b_1,\ldots,b_n)$ constructs *A* named trees; assumes: $A: B_1\times\ldots\times B_n$ with $b_i \in B_i$, $(c_1,\ldots,c_m)$ consructs anonymous trees; implies $(C_1\times\ldots\times C_m)$, with $c_i \in C_i$. | 44,78 |
| *s-* | Selector function name prefixes; $s\text{-}B_j$, $s\text{-}C_k$ selects $B_j$, respectively $C_k$ objects. | 44,78 |
| $=$ | equal to | |
| $\neq$ | different from | |

## Abstract Syntax

| | | |
|---|---|---|
| | | 42,78 |
| $A = E$ | Domain equations; = gives the name $A$ to the set of objects denoted by $E$, | 43,78 |
| $A :: E$ | :: gives the name $A$ to the set of tree *mk-A(e)* tree objects, where $e$ is any object in domain $E$. | 43,78 |
| *-set* | -- see under Sets above | 39 |
| $^*$, $^+$ | -- see under Tuples above | 41 |
| $\overset{m}{\rightarrow}$, $\overset{m}{\leftrightarrow}$ | -- see under Maps above | 40 |
| $\times$ | -- see under Trees above | 66,77 |
| $\rightarrow$ | (total) functions | 28-32,78-9 |
| $\overset{\sim}{\rightarrow}$ | partial functions | 28-32,78-9 |
| $\underline{\cup}$ | Map domain merging | |
| [ ] | Optional domain forming operator; defines domain of given set union {<u>*nil*</u>}. | 43 |
| \| | Non-discriminated union forming domain operator. | 43,66,77 |
| *is-* | Domain membership test predicate name prefix, *is-A(o)* corresponds to: $o \in A$. | |

## Function Definitions

| | | |
|---|---|---|
| | | 78 |
| $\lambda x.e$ | Function from $x$ domain to domain of $e$ values. | 29-30 |
| $f(a) \underline{\underline{\Delta}} B(a)$ | *f*: function name, $a$ argument(s), $B(a)$ is any clause: stament or expression (sometimes $\underline{\Delta}$ or just = is used). | 28 |
| <u>*type*</u>: $A \rightarrow B$ }<br>$f$: $A \rightarrow B$ }<br><u>*type*</u>: $f$: $A \rightarrow B$ } | three synonymous type expressions | |
| $\Rightarrow$ | used only in type expressions; defines state usage: $A \Rightarrow B$ is thus equal to $A \rightarrow (\Sigma \rightarrow (\Sigma \times B))$ | 114 |

Applicative Combinators

| | | |
|---|---|---|
| *let* $id=e$ *in* $b$ | Block expression; defines all free occurrences of $id$ in $B$ as bound to $e$. Non-recursive *lets* correspond to: $(\lambda id.b)(e)$. | |
| $f(a)$ | Function application | |

Imperative Combinators

| | | |
|---|---|---|
| *dcl* v:=$e$ *type* $D$ | declaration of assignable variable: v | |
| *c* | contents operator; applied to a variable ('v'), *c* v defines its contents. | 113 |
| v := e | assignment | 113 |
| ; | statement composition | 33,92,107 |
| *def* $id$:$e$; $s$ | imperative *let* clause | 34,94 |
| *while* $e$ *do* $s$ | while loop | |
| *for* $i=m$ *to* $n$ *do* $S(i)$ | iterative loop; steps in ordered sequence from static lower bound $m$ to static upper bound $n$. | |
| *for all* $e \in Set$ *do* $S(e)$ | iterative loop; steps in arbitrary sequence with $e$ ranging over static set $Set$. | |
| $return(v)$ | raises pure value to "imperative value": ( $\lambda v.\lambda \sigma.(\sigma, v)$ ) | 34,94 |

Structured Combinators

| | | |
|---|---|---|
| *if* $t$ *then* $c$ *else* $a$ | If-then-else clause | |
| $b_1 \rightarrow c_1, \ldots b_n \rightarrow c_n$ | $n$-way if-then-else clause | |

| | | |
|---|---|---|
| *cases* $e_0$: $e_1 \rightarrow c_1, \ldots, e_n \rightarrow c_n$ | $n$-way cases selector clause | |

## Exit Combinators

| | | |
|---|---|---|
| *trap* *id* *with* $E(id)$ *in* $B$ | Non-recursive exit stopper | |
| *always* $E(id)$ *in* $B$ | Non-recursive exit filter | 37,108 |
| *tixe* [ $a \mapsto b$ \| $P(a,b)$ ] *in* $B$ | Recursive exit stopper | 36,107 |
| *exit* | exit causer -- no value passing | 36 |
| *exit*(*e*) | exit causer -- with value passing | 36,107 |

## Overloaded Symbols (for references, see above)

| | | |
|---|---|---|
| +, + | integer addition | -- map extension |
| \| | domain union | -- map restriction |
| - | integer subtraction | -- set difference |
| * | integer multiplication | -- tuple domain former |
| × | integer multiplication | -- cartesian domain former |
| ∪ | set union | -- map merge |
| → | function domain former | -- conditional clause delimiter |
| [ ] | map object delimiter | -- optional domain former |
| *[ ]* | tuple index operator | -- syntactic argument delimiter |
| = | equality between any object pair | |
| ≠ | in-equality between any object pair | |