# a Practical Theory of Programming

## 2018-11-7 edition


## Eric C.R. Hehner

Department of Computer Science
University of Toronto
Toronto ON M5S 2E4 Canada



The first edition of this book was published by
Springer-Verlag Publishers, New York, 1993
ISBN 0-387-94106-1 QA76.6.H428


The current edition is available free at
**www.cs.utoronto.ca/~hehner/aPToP**

An on-line course based on this book is at
**www.cs.utoronto.ca/~hehner/FMSD**

The author's website is
**www.cs.utoronto.ca/~hehner**

# 11.4 Laws

### 11.4.0 Binary

Let $a$ , $b$ , $c$ , $d$ , and $e$ be binary.

Binary Laws

$\top$

$\neg\bot$

Law of Excluded Middle (Tertium non Datur)

$a \lor \neg a$

Law of Noncontradiction

$\neg(a \land \neg a)$

Base Laws

$\neg(a \land \bot)$

$a \lor \top$

$a \Rightarrow \top$

$\bot \Rightarrow a$

Identity Laws

$\top \land a = a$

$\bot \lor a = a$

$\top \Rightarrow a = a$

$\top = a = a$

Idempotent Laws

$a \land a = a$

$a \lor a = a$

Reflexive Laws

$a \Rightarrow a$

$a = a$

Laws of Indirect Proof

$\neg a \Rightarrow \bot = a$ (Reductio ad Absurdum)

$\neg a \Rightarrow a = a$

Law of Specialization

$a \land b \Rightarrow a$

Associative Laws

$a \land (b \land c) = (a \land b) \land c$

$a \lor (b \lor c) = (a \lor b) \lor c$

$a = (b = c) = (a = b) = c$

$a \neq (b \neq c) = (a \neq b) \neq c$

$a = (b \neq c) = (a = b) \neq c$

Mirror Law

$a \Leftarrow b = b \Rightarrow a$

Law of Double Negation

$\neg\neg a = a$

Duality Laws (deMorgan)

$\neg(a \land b) = \neg a \lor \neg b$

$\neg(a \lor b) = \neg a \land \neg b$

Laws of Exclusion

$a \Rightarrow \neg b = b \Rightarrow \neg a$

$a = \neg b = a \neq b = \neg a = b$

Laws of Inclusion

$a \Rightarrow b = \neg a \lor b$ (Material Implication)

$a \Rightarrow b = (a \land b = a)$

$a \Rightarrow b = (a \lor b = b)$

Absorption Laws

$a \land (a \lor b) = a$

$a \lor (a \land b) = a$

Laws of Direct Proof

$(a \Rightarrow b) \land a \Rightarrow b$ (Modus Ponens)

$(a \Rightarrow b) \land \neg b \Rightarrow \neg a$ (Modus Tollens)

$(a \lor b) \land \neg a \Rightarrow b$(Disjunctive Syllogism)

Transitive Laws

$(a \land b) \land (b \land c) \Rightarrow (a \land c)$

$(a \Rightarrow b) \land (b \Rightarrow c) \Rightarrow (a \Rightarrow c)$

$(a = b) \land (b = c) \Rightarrow (a = c)$

$(a \Rightarrow b) \land (b = c) \Rightarrow (a \Rightarrow c)$

$(a = b) \land (b \Rightarrow c) \Rightarrow (a \Rightarrow c)$

Distributive Laws (Factoring)

$a \land (b \land c) = (a \land b) \land (a \land c)$

$a \land (b \lor c) = (a \land b) \lor (a \land c)$

$a \lor (b \land c) = (a \lor b) \land (a \lor c)$

$a \lor (b \lor c) = (a \lor b) \lor (a \lor c)$

$a \lor (b \Rightarrow c) = (a \lor b) \Rightarrow (a \lor c)$

$a \lor (b = c) = (a \lor b) = (a \lor c)$

$a \Rightarrow (b \land c) = (a \Rightarrow b) \land (a \Rightarrow c)$

$a \Rightarrow (b \lor c) = (a \Rightarrow b) \lor (a \Rightarrow c)$

$a \Rightarrow (b \Rightarrow c) = (a \Rightarrow b) \Rightarrow (a \Rightarrow c)$

$a \Rightarrow (b = c) = (a \Rightarrow b) = (a \Rightarrow c)$

Symmetry Laws (Commutative Laws)

$a \wedge b \;=\; b \wedge a$
$a \vee b \;=\; b \vee a$
$a = b \;=\; b = a$
$a \neq b \;=\; b \neq a$

Antisymmetry Law (Double Implication)

$(a \Rightarrow b) \wedge (b \Rightarrow a) \;=\; a = b$

Laws of Discharge

$a \wedge (a \Rightarrow b) \;=\; a \wedge b$
$a \Rightarrow (a \wedge b) \;=\; a \Rightarrow b$

Antimonotonic Law

$a \Rightarrow b \;\Longrightarrow\; (b \Rightarrow c) \Rightarrow (a \Rightarrow c)$

Monotonic Laws

$a \Rightarrow b \;\Longrightarrow\; c \wedge a \Rightarrow c \wedge b$
$a \Rightarrow b \;\Longrightarrow\; c \vee a \Rightarrow c \vee b$
$a \Rightarrow b \;\Longrightarrow\; (c \Rightarrow a) \Rightarrow (c \Rightarrow b)$

Law of Generalization

$a \Rightarrow a \vee b$

Antidistributive Laws

$a \wedge b \Rightarrow c \;=\; (a \Rightarrow c) \vee (b \Rightarrow c)$
$a \vee b \Rightarrow c \;=\; (a \Rightarrow c) \wedge (b \Rightarrow c)$

Laws of Portation

$a \wedge b \Rightarrow c \;=\; a \Rightarrow (b \Rightarrow c)$
$a \wedge b \Rightarrow c \;=\; a \Rightarrow \neg b \vee c$

Laws of Conflation

$(a \Rightarrow b) \wedge (c \Rightarrow d) \;\Longrightarrow\; a \wedge c \Rightarrow b \wedge d$
$(a \Rightarrow b) \wedge (c \Rightarrow d) \;\Longrightarrow\; a \vee c \Rightarrow b \vee d$

Contrapositive Law

$a \Rightarrow b \;=\; \neg b \Rightarrow \neg a$

Laws of Equality and Difference

$a{=}b \;=\; (a \wedge b) \vee (\neg a \wedge \neg b)$
$a{\neq}b \;=\; (a \wedge \neg b) \vee (\neg a \wedge b)$

Law of Resolution

$a \wedge c \;\Longrightarrow\; (a \vee b) \wedge (\neg b \vee c) \;=\; (a \wedge \neg b) \vee (b \wedge c) \;\Longrightarrow\; a \vee c$

Case Creation Laws

$a$ = **if** $b$ **then** $b \Rightarrow a$ **else** $\neg b \Rightarrow a$ **fi**
$a$ = **if** $b$ **then** $b \wedge a$ **else** $\neg b \wedge a$ **fi**
$a$ = **if** $b$ **then** $b{=}a$ **else** $b{\neq}a$ **fi**

Case Absorption Laws

**if** $a$ **then** $b$ **else** $c$ **fi** = **if** $a$ **then** $a \wedge b$ **else** $c$ **fi**
**if** $a$ **then** $b$ **else** $c$ **fi** = **if** $a$ **then** $a \Rightarrow b$ **else** $c$ **fi**
**if** $a$ **then** $b$ **else** $c$ **fi** = **if** $a$ **then** $a{=}b$ **else** $c$ **fi**
**if** $a$ **then** $b$ **else** $c$ **fi** = **if** $a$ **then** $b$ **else** $\neg a \wedge c$ **fi**
**if** $a$ **then** $b$ **else** $c$ **fi** = **if** $a$ **then** $b$ **else** $a \vee c$ **fi**
**if** $a$ **then** $b$ **else** $c$ **fi** = **if** $a$ **then** $b$ **else** $a{\neq}c$ **fi**

Case Analysis Laws

**if** $a$ **then** $b$ **else** $c$ **fi** $=$ $(a \wedge b) \vee (\neg a \wedge c)$
**if** $a$ **then** $b$ **else** $c$ **fi** $=$ $(a \Rightarrow b) \wedge (\neg a \Rightarrow c)$

One Case Laws

**if** $a$ **then** $\top$ **else** $b$ **fi** $=$ $a \vee b$
**if** $a$ **then** $\bot$ **else** $b$ **fi** $=$ $\neg a \wedge b$
**if** $a$ **then** $b$ **else** $\top$ **fi** $=$ $a \Rightarrow b$
**if** $a$ **then** $b$ **else** $\bot$ **fi** $=$ $a \wedge b$

Case Distributive Laws (Case Factoring)

$\neg$ **if** $a$ **then** $b$ **else** $c$ **fi** $=$ **if** $a$ **then** $\neg b$ **else** $\neg c$ **fi**
**if** $a$ **then** $b$ **else** $c$ **fi** $\wedge d$ $=$ **if** $a$ **then** $b \wedge d$ **else** $c \wedge d$ **fi**
and similarly replacing $\wedge$ by any of $\vee \;=\; \neq \;\Rightarrow\; \Leftarrow$
**if** $a$ **then** $b \wedge c$ **else** $d \wedge e$ **fi** $=$ **if** $a$ **then** $b$ **else** $d$ **fi** $\wedge$ **if** $a$ **then** $c$ **else** $e$ **fi**
and similarly replacing $\wedge$ by any of $\vee \;=\; \neq \;\Rightarrow\; \Leftarrow$

End of Binary

### 11.4.1 Generic

The operators $=$ $\ne$ **if then else fi** apply to every type of expression (but the first operand of **if then else fi** must be binary), with the laws

| | | | | |
|---|---|---|---|---|
| $x = x$ | reflexivity | **if** $\top$ **then** $x$ **else** $y$ **fi** $= x$ | | case base |
| $x{=}y \;=\; y{=}x$ | symmetry | **if** $\bot$ **then** $x$ **else** $y$ **fi** $= y$ | | case base |
| $x{=}y \wedge y{=}z \;\Rightarrow\; x{=}z$ | transitivity | **if** $a$ **then** $x$ **else** $x$ **fi** $= x$ | | case idempotent |
| $x{=}y \;\Rightarrow\; f\,x = f\,y$ | transparency | **if** $a$ **then** $x$ **else** $y$ **fi** $=$ **if** $\neg a$ **then** $y$ **else** $x$ **fi** | | |
| $x{\ne}y \;=\; \neg(x{=}y)$ | unequality | | | case reversal |

The operators $<$ $\le$ $>$ $\ge$ apply to numbers, characters, strings, and lists, with the laws

| | |
|---|---|
| $x{\le}x$ | reflexivity |
| $\neg\, x{<}x$ | irreflexivity |
| $\neg(x{<}y \wedge x{>}y)$ | exclusivity |
| $\neg(x{<}y \wedge x{=}y)$ | exclusivity |
| $x{\le}y \wedge y{\le}x = x{=}y$ | antisymmetry |
| $x{\le}y \wedge y{\le}z \Rightarrow x{\le}z$ | transitivity |
| $x{<}y \wedge y{<}z \Rightarrow x{<}z$ | transitivity |
| $x{\le}y \;=\; x{<}y \vee x{=}y$ | inclusivity |
| $x{>}y \;=\; y{<}x$ | mirror |
| $x{\ge}y \;=\; y{\le}x$ | mirror |
| $x{<}y \vee x{=}y \vee x{>}y$ | totality, trichotomy |

End of Generic

### 11.4.2 Numbers

Let $d$ be a sequence of (zero or more) digits, and let $x$ , $y$ , and $z$ be numbers.

| | |
|---|---|
| $d0{+}1 = d1$ | counting |
| $d1{+}1 = d2$ | counting |
| $d2{+}1 = d3$ | counting |
| $d3{+}1 = d4$ | counting |
| $d4{+}1 = d5$ | counting |
| $d5{+}1 = d6$ | counting |
| $d6{+}1 = d7$ | counting |
| $d7{+}1 = d8$ | counting |
| $d8{+}1 = d9$ | counting |
| $d9{+}1 = (d{+}1)0$ | counting (see Exercise 32) |
| $x{+}0 = x$ | identity |
| $x{+}y = y{+}x$ | symmetry |
| $x{+}(y{+}z) \;=\; (x{+}y){+}z$ | associativity |
| $-\infty{<}x{<}\infty \Rightarrow (x{+}y = x{+}z \;=\; y{=}z)$ | cancellation |
| $-\infty{<}x \Rightarrow\; \infty{+}x = \infty$ | absorption |
| $x{<}\infty \;\Rightarrow\; -\infty + x = -\infty$ | absorption |
| $-x \;=\; 0 - x$ | negation |
| $- -x = x$ | self-inverse |
| $-(x{+}y) = -x + -y$ | distributivity |
| $-(x{-}y) = y{-}x$ | antisymmetry |
| $-(x{\times}y) = -x \times y$ | semi-distributivity |
| $-(x/y) = -x / y$ | semi-distributivity |
| $x{-}0 = x$ | identity |

| | |
|---|---|
| $x-y \ = \ x+-y$ | subtraction |
| $x+(y-z) \ = \ (x+y)-z$ | associativity |
| $-\infty<x<\infty \ \Rightarrow \ (x-y=x-z \ = \ y=z)$ | cancellation |
| $-\infty<x<\infty \ \Rightarrow \ x-x=0$ | inverse |
| $x<\infty \ \Rightarrow \ \infty-x=\infty$ | absorption |
| $-\infty<x \ \Rightarrow \ -\infty-x=-\infty$ | absorption |
| $-\infty<x<\infty \ \Rightarrow \ x\times 0=0$ | base |
| $x\times 1=x$ | identity |
| $x\times y=y\times x$ | symmetry |
| $x\times(y+z)=x\times y+x\times z$ | distributivity |
| $x\times(y\times z)=(x\times y)\times z$ | associativity |
| $-\infty<x<\infty \land x\neq 0 \ \Rightarrow \ (x\times y=x\times z \ = \ y=z)$ | cancellation |
| $0<x \ \Rightarrow \ x\times\infty=\infty$ | absorption |
| $0<x \ \Rightarrow \ x\times-\infty \ = \ -\infty$ | absorption |
| $x/1=x$ | identity |
| $-\infty<x<\infty \land x\neq 0 \ \Rightarrow \ x/x=1$ | inverse |
| $x\times(y/z)=(x\times y)/z=x/(z/y)$ | multiplication-division |
| $y\neq 0 \ \Rightarrow \ (x/y)/z=x/(y\times z)$ | multiplication-division |
| $-\infty<x<\infty \ \Rightarrow \ x/\infty=0=x/-\infty$ | annihilation |
| $-\infty<x<\infty \ \Rightarrow \ x^0=1$ | base |
| $x^1=x$ | identity |
| $x^{y+z}=x^y\times x^z$ | exponents |
| $x^{y\times z}=(x^y)^z$ | exponents |
| $-\infty<0<1<\infty$ | direction |
| $x<y \ = \ -y<-x$ | reflection |
| $-\infty<x<\infty \ \Rightarrow \ (x+y<x+z \ = \ y<z)$ | cancellation, translation |
| $0<x<\infty \ \Rightarrow \ (x\times y<x\times z \ = \ y<z)$ | cancellation, scale |
| $x<y \lor x=y \lor x>y$ | trichotomy |
| $-\infty \le x \le \infty$ | extremes |

End of Numbers

### 11.4.3 Bunches

Let $x$ and $y$ be elements (binary values, numbers, characters, sets, strings and lists of elements).

| | |
|---|---|
| $x\text{: } y \ = \ x=y$ | elementary law |
| $x\text{: } A,B \ = \ x\text{: } A \ \lor \ x\text{: } B$ | compound law |
| $A,A=A$ | idempotence |
| $A,B=B,A$ | symmetry |
| $A,(B,C)=(A,B),C$ | associativity |
| $A‘A=A$ | idempotence |
| $A‘B=B‘A$ | symmetry |
| $A‘(B‘C)=(A‘B)‘C$ | associativity |
| $A,B\text{: } C \ = \ A\text{: } C \ \land \ B\text{: } C$ | antidistributivity |
| $A\text{: } B‘C \ = \ A\text{: } B \ \land \ A\text{: } C$ | distributivity |
| $A\text{: } A,B$ | generalization |
| $A‘B\text{: } A$ | specialization |
| $A\text{: } A$ | reflexivity |
| $A\text{: } B \ \land \ B\text{: } A \ = \ A=B$ | antisymmetry |
| $A\text{: } B \ \land \ B\text{: } C \ \Rightarrow \ A\text{: } C$ | transitivity |
| $¢\ null=0$ | size |

| | |
|---|---|
| $¢x = 1$ | size |
| $¢(A, B) + ¢(A‘B) = ¢A + ¢B$ | size |
| $\neg\ x: A \;\Rightarrow\; ¢(A‘x) = 0$ | size |
| $A: B \;\Rightarrow\; ¢A \le ¢B$ | size |
| $A,(A‘B) \;=\; A$ | absorption |
| $A‘(A,B) \;=\; A$ | absorption |
| $A: B \;\;=\;\; A,B = B \;\;=\;\; A = A‘B$ | inclusion |
| $A,(B,C) \;=\; (A,B),(A,C)$ | distributivity |
| $A,(B‘C) \;=\; (A,B)‘(A,C)$ | distributivity |
| $A‘(B,C) \;=\; (A‘B), (A‘C)$ | distributivity |
| $A‘(B‘C) \;=\; (A‘B)‘(A‘C)$ | distributivity |
| $A: B \;\wedge\; C: D \;\Rightarrow\; A,C: B,D$ | conflation, monotonicity |
| $A: B \;\wedge\; C: D \;\Rightarrow\; A‘C: B‘D$ | conflation, monotonicity |
| $null: A$ | induction |
| $A, null = A$ | identity |
| $A \text{ ‘ } null = null$ | base |
| $¢A = 0 \;\;=\;\; A = null$ | size |
| $x: int \;\wedge\; y: xint \;\wedge\; x{\le}y \;\Rightarrow\; (i: x,..y \;\;=\;\; i: int \;\wedge\; x{\le}i{<}y)$ | |
| $x: int \;\wedge\; y: xint \;\wedge\; x{\le}y \;\Rightarrow\; ¢(x,..y) \;=\; y{-}x$ | |
| $-null \;=\; null$ | distribution |
| $-(A, B) \;=\; -A, -B$ | distribution |
| $A{+}null \;=\; null{+}A \;=\; null$ | distribution |
| $(A, B){+}(C, D) \;=\; A{+}C, A{+}D, B{+}C, B{+}D$ | distribution |

and similarly for many other operators (see the final page of the book)

End of Bunches

### 11.4.4 Sets

$\{\sim A\} \;=\; A$

$\sim\{A\} \;=\; A$

$\{A\} \;\ne\; A$

$A \in \{B\} \;\;=\;\; A: B$

$\{A\} \subseteq \{B\} \;\;=\;\; A: B$

$\{A\}: \text{ϟ}B \;\;=\;\; A: B$

$\$\{A\} = ¢A$

$\{A\} \cup \{B\} \;=\; \{A, B\}$

$\{A\} \cap \{B\} \;=\; \{A \text{ ‘ } B\}$

$\{A\} = \{B\} \;\;=\;\; A = B$

$\{A\} \ne \{B\} \;\;=\;\; A \ne B$

End of Sets

### 11.4.5 Strings

Let $S$, $T$, and $U$ be strings; let $i$ and $j$ be items (binary values, numbers, characters, sets, lists, functions); let $n$ be extended natural; let $x$, $y$, and $z$ be integers.

$nil; S \;=\; S; nil \;=\; S$

$S; (T; U) \;=\; (S; T); U$

$\leftrightarrow nil \;=\; 0$

$\leftrightarrow i \;=\; 1$

$\leftrightarrow(S; T) \;=\; \leftrightarrow S + \leftrightarrow T$

$S_{nil} \;=\; nil$

$\leftrightarrow S{<}\infty \;\Rightarrow\; (S; i; T)_{\leftrightarrow S} \;=\; i$

$S_{T;\, U} \;=\; S_T; S_U$

$S_{(T_U)} \;=\; (S_T)_U$

$S_{\{A\}} \;=\; \{S_A\}$

$\leftrightarrow S{<}\infty \;\Rightarrow\; nil \;\le\; S \;<\; S; i; T$

$\leftrightarrow S{<}\infty \;\Rightarrow\; (i{<}j \;\Rightarrow\; S; i; T \;<\; S; j; U)$

$\leftrightarrow S{<}\infty \;\Rightarrow\; (i{=}j \;\;=\;\; S; i; T \;=\; S; j; T)$

$0{*}S \;=\; nil$

$(n{+}1){*}S \;=\; n{*}S; S$

$\leftrightarrow S{<}\infty \;\Rightarrow\; S;i;T \triangleleft \leftrightarrow S \triangleright j \;=\; S;j;T$

$x;..x \;=\; nil$

$x;..x{+}1 \;=\; x$

$(x;..y) \;;\; (y;..z) \;=\; x;..z$

$\leftrightarrow(x;..y) \;=\; y{-}x$

End of Strings

### 11.4.6 Lists

Let $S$ and $T$ be strings; let $i$ and $j$ be items (binary values, numbers, characters, sets, lists, functions); let $L$, $M$, and $N$ be lists.

$[S] \neq S$
$\sim[S] = S$
$[\sim L] = L$
$[S]\, T = S_T$
$[S]+[T] = [S;\, T]$
$[S] = [T] \;=\; S = T$
$[S] < [T] \;=\; S < T$
$nil \rightarrow i \mid L = i$
$n \rightarrow i \mid [S] = [S \triangleleft n \triangleright i]$
$(S;T) \rightarrow i \mid L = S \rightarrow (T \rightarrow i \mid L@S) \mid L$

$\#[S] = \leftrightarrow S$
$S_{[T]} = [S_T]$
$[S]\,[T] = [S_T]$
$L\,\{A\} = \{L\,A\}$
$L\,[S] = [L\,S]$
$(L\,M)\,N = L\,(M\,N)$
$L@nil = L$
$L@i = L\,i$
$L@(S;\,T) = L@S@T$

End of Lists

### 11.4.7 Functions

Renaming Law — if $v$ and $w$ do not appear in $D$ and $w$ does not appear in $b$
$\langle v\colon D \rightarrow b\rangle = \langle w\colon D \rightarrow \langle v\colon D \rightarrow b\rangle w\rangle$

Application Law: if element $x\colon D$
$\langle v\colon D \rightarrow b\rangle x =$ (substitute $x$ for $v$ in $b$)

Domain Law
$\Box\langle v\colon D \rightarrow b\rangle = D$

Law of Extension
$f = \langle v\colon \Box f \rightarrow f\,v\rangle$

Function Inclusion Law
$f\colon g \;=\; \Box g\colon \Box f \land \forall x\colon \Box g\cdot fx\colon gx$

Function Equality Law
$f = g \;=\; \Box f = \Box g \land \forall x\colon \Box f\cdot fx = gx$

Laws of Functional Union
$\Box(f, g) = \Box f\ ‘\ \Box g$
$(f, g)\, x = f\,x, g\,x$

Distributive Laws
$f\,null = null$
$f\,(A, B) = f\,A, f\,B$
$f\,(\S g) = \S y\colon f\,(\Box g)\cdot \exists x\colon \Box g\cdot fx{=}y \land gx$
$f$ **if** $b$ **then** $x$ **else** $y$ **fi** = **if** $b$ **then** $f\,x$ **else** $f\,y$ **fi**
**if** $b$ **then** $f$ **else** $g$ **fi** $x$ = **if** $b$ **then** $f\,x$ **else** $g\,x$ **fi**

Function Composition Laws: If $\neg\, f\colon \Box g$
$\Box(g\,f) = \S x\colon \Box f\cdot fx\colon \Box g$
$(g\,f)\,x = g\,(f\,x)$
$f\,(g\,h) = (f\,g)\,h$

Laws of Functional Intersection
$\Box(f\ ‘\ g) = \Box f, \Box g$
$(f\ ‘\ g)\,x = (f \mid g)\,x\ ‘\ (g \mid f)\,x$

Laws of Selective Union
$\Box(f \mid g) = \Box f, \Box g$
$(f \mid g)\,x =$ **if** $x\colon \Box f$ **then** $f\,x$ **else** $g\,x$ **fi**
$f \mid f = f$
$f \mid (g \mid h) = (f \mid g) \mid h$
$(g \mid h)\,f = g\,f \mid h\,f$

Arrow Laws
$f\colon null \rightarrow A$
$A \rightarrow B\colon\ (A‘C) \rightarrow (B,D)$
$f\colon A \rightarrow B \;=\; A\colon \Box f \land \forall a\colon A\cdot fa\colon B$

End of Functions

### 11.4.8 Quantifiers

Let $x$ be an element, let $a$, $b$ and $c$ be binary, let $n$ and $m$ be numeric, let $f$ and $g$ be functions, and let $p$ be a predicate.

$\forall v\colon null\cdot b \;=\; \top$

$\forall v\colon x\cdot b \;=\; \langle v\colon x \to b\rangle\, x$

$\forall v\colon A,B\cdot b \;=\; (\forall v\colon A\cdot b) \wedge (\forall v\colon B\cdot b)$

$\forall v\colon (\S v\colon D\cdot b)\cdot c \;=\; \forall v\colon D\cdot b \Rightarrow c$

$\exists v\colon null\cdot b \;=\; \bot$

$\exists v\colon x\cdot b \;=\; \langle v\colon x \to b\rangle\, x$

$\exists v\colon A,B\cdot b \;=\; (\exists v\colon A\cdot b) \vee (\exists v\colon B\cdot b)$

$\exists v\colon (\S v\colon D\cdot b)\cdot c \;=\; \exists v\colon D\cdot b \wedge c$

$\Sigma v\colon null\cdot n \;=\; 0$

$\Sigma v\colon x\cdot n \;=\; \langle v\colon x \to n\rangle\, x$

$(\Sigma v\colon A,B\cdot n) + (\Sigma v\colon A‘B\cdot n) \;=\; (\Sigma v\colon A\cdot n) + (\Sigma v\colon B\cdot n)$

$\Sigma v\colon (\S v\colon D\cdot b)\cdot n \;=\; \Sigma v\colon D\cdot$ **if** $b$ **then** $n$ **else** $0$ **fi**

$\Pi v\colon null\cdot n \;=\; 1$

$\Pi v\colon x\cdot n \;=\; \langle v\colon x \to n\rangle\, x$

$(\Pi v\colon A,B\cdot n) \times (\Pi v\colon A‘B\cdot n) \;=\; (\Pi v\colon A\cdot n) \times (\Pi v\colon B\cdot n)$

$\Pi v\colon (\S v\colon D\cdot b)\cdot n \;=\; \Pi v\colon D\cdot$ **if** $b$ **then** $n$ **else** $1$ **fi**

$MIN\ v\colon null\cdot n \;=\; \infty$

$MIN\ v\colon x\cdot n \;=\; \langle v\colon x \to n\rangle\, x$

$MIN\ v\colon A,B\cdot n \;=\; min\ (MIN\ v\colon A\cdot n)\ (MIN\ v\colon B\cdot n)$

$MIN\ v\colon (\S v\colon D\cdot b)\cdot n \;=\; MIN\ v\colon D\cdot$ **if** $b$ **then** $n$ **else** $\infty$ **fi**

$MAX\ v\colon null\cdot n \;=\; -\infty$

$MAX\ v\colon x\cdot n \;=\; \langle v\colon x \to n\rangle\, x$

$MAX\ v\colon A,B\cdot n \;=\; max\ (MAX\ v\colon A\cdot n)\ (MAX\ v\colon B\cdot n)$

$MAX\ v\colon (\S v\colon D\cdot b)\cdot n \;=\; MAX\ v\colon D\cdot$ **if** $b$ **then** $n$ **else** $-\infty$ **fi**

$\S v\colon null\cdot b \;=\; null$

$\S v\colon x\cdot b \;=\;$ **if** $\langle v\colon x \to b\rangle\, x$ **then** $x$ **else** $null$ **fi**

$\S v\colon A,B\cdot b \;=\; (\S v\colon A\cdot b), (\S v\colon B\cdot b)$

$\S v\colon A‘B\cdot b \;=\; (\S v\colon A\cdot b)\ ‘\ (\S v\colon B\cdot b)$

$\S v\colon (\S v\colon D\cdot b)\cdot c \;=\; \S v\colon D\cdot b \wedge c$

Change of Variable Laws — if $d$ does not appear in $b$

$\forall r\colon fD\cdot b \;=\; \forall d\colon D\cdot \langle r\colon fD \to b\rangle\, (fd)$

$\exists r\colon fD\cdot b \;=\; \exists d\colon D\cdot \langle r\colon fD \to b\rangle\, (fd)$

$MIN\ r\colon fD\cdot n \;=\; MIN\ d\colon D\cdot \langle r\colon fD \to n\rangle\, (fd)$

$MAX\ r\colon fD\cdot n \;=\; MAX\ d\colon D\cdot \langle r\colon fD \to n\rangle\, (fd)$

Cardinality Law

$¢A \;=\; \Sigma\,(A \to 1)$

Identity Laws

$\forall v\cdot \top$

$\neg \exists v\cdot \bot$

Bunch-Element Conversion Laws

$V\colon W \;=\; \forall v\colon V\cdot \exists w\colon W\cdot v{=}w$

$fV\colon gW \;=\; \forall v\colon V\cdot \exists w\colon W\cdot fv{=}gw$

Idempotent Laws — if $D{\neq}null$ and $v$ does not appear in $b$

$\forall v\colon D\cdot b \;=\; b$

$\exists v\colon D\cdot b \;=\; b$

Absorption Laws — if $x\colon D$

$\langle v\colon D \to b\rangle\, x \wedge \exists v\colon D\cdot b \;=\; \langle v\colon D \to b\rangle\, x$

$\langle v\colon D \to b\rangle\, x \vee \forall v\colon D\cdot b \;=\; \langle v\colon D \to b\rangle\, x$

$\langle v\colon D \to b\rangle\, x \wedge \forall v\colon D\cdot b \;=\; \forall v\colon D\cdot b$

$\langle v\colon D \to b\rangle\, x \vee \exists v\colon D\cdot b \;=\; \exists v\colon D\cdot b$

Distributive Laws — if $D{\neq}null$ and $v$ does not appear in $a$

$a \wedge \forall v\colon D\cdot b \;=\; \forall v\colon D\cdot a \wedge b$

$a \wedge \exists v\colon D\cdot b \;=\; \exists v\colon D\cdot a \wedge b$

$a \vee \forall v\colon D\cdot b \;=\; \forall v\colon D\cdot a \vee b$

$a \vee \exists v\colon D\cdot b \;=\; \exists v\colon D\cdot a \vee b$

$a \Rightarrow \forall v\colon D\cdot b \;=\; \forall v\colon D\cdot a \Rightarrow b$

$a \Rightarrow \exists v\colon D\cdot b \;=\; \exists v\colon D\cdot a \Rightarrow b$

Antidistributive Laws — if $D{\neq}null$ and $v$ does not appear in $a$

$a \Leftarrow \exists v\colon D\cdot b \;=\; \forall v\colon D\cdot a \Leftarrow b$

$a \Leftarrow \forall v\colon D\cdot b \;=\; \exists v\colon D\cdot a \Leftarrow b$

Specialization Law — if $x: D$

$\forall v: D\cdot b \Longrightarrow \langle v: D\to b\rangle\, x$

Generalization Law — if $x: D$

$\langle v: D\to b\rangle\, x \Longrightarrow \exists v: D\cdot b$

One-Point Laws — if $x: D$ and $v$ does not appear in $x$

$\forall v: D\cdot v{=}x \Rightarrow b = \langle v: D\to b\rangle\, x$

$\exists v: D\cdot v{=}x \wedge b = \langle v: D\to b\rangle\, x$

Splitting Laws — for any fixed domain

$\forall v\cdot a \wedge b = (\forall v\cdot a) \wedge (\forall v\cdot b)$

$\exists v\cdot a \wedge b \Longrightarrow (\exists v\cdot a) \wedge (\exists v\cdot b)$

$\forall v\cdot a \vee b \Longleftarrow (\forall v\cdot a) \vee (\forall v\cdot b)$

$\exists v\cdot a \vee b = (\exists v\cdot a) \vee (\exists v\cdot b)$

$\forall v\cdot a \Rightarrow b \Longrightarrow (\forall v\cdot a) \Rightarrow (\forall v\cdot b)$

$\forall v\cdot a \Rightarrow b \Longrightarrow (\exists v\cdot a) \Rightarrow (\exists v\cdot b)$

$\forall v\cdot a = b \Longrightarrow (\forall v\cdot a) = (\forall v\cdot b)$

$\forall v\cdot a = b \Longrightarrow (\exists v\cdot a) = (\exists v\cdot b)$

Duality Laws

$\neg \forall v\cdot b = \exists v\cdot \neg b$ (deMorgan)

$\neg \exists v\cdot b = \forall v\cdot \neg b$ (deMorgan)

$-\, MAX\ v\cdot n = MIN\ v\cdot -n$

$-\, MIN\ v\cdot n = MAX\ v\cdot -n$

Commutative Laws

$\forall v\cdot \forall w\cdot b = \forall w\cdot \forall v\cdot b$

$\exists v\cdot \exists w\cdot b = \exists w\cdot \exists v\cdot b$

Solution Laws

$\S v: D\cdot \top = D$

$(\S v: D\cdot b):\ D$

$\S v: D\cdot \bot = null$

$(\S v\cdot b): (\S v\cdot c) = \forall v\cdot b \Rightarrow c$

$(\S v\cdot b), (\S v\cdot c) = \S v\cdot b \vee c$

$(\S v\cdot b)\ \text{‘}\ (\S v\cdot c) = \S v\cdot b \wedge c$

$x: \S p = x: \Box p \wedge px$

$\forall f = (\S f){=}(\Box f)$

$\exists f = (\S f){\neq}null$

Semicommutative Laws (Skolem)

$\exists v\cdot \forall w\cdot b \Longrightarrow \forall w\cdot \exists v\cdot b$

$\forall x\cdot \exists y\cdot pxy = \exists f\cdot \forall x\cdot px(fx)$

Domain Change Laws

$A: B \Longrightarrow (\forall v: A\cdot b) \Leftarrow (\forall v: B\cdot b)$

$A: B \Longrightarrow (\exists v: A\cdot b) \Rightarrow (\exists v: B\cdot b)$

$\forall v: A\cdot v: B \Rightarrow p = \forall v: A\text{‘}B\cdot p$

$\exists v: A\cdot v: B \wedge p = \exists v: A\text{‘}B\cdot p$

Bounding Laws

if $v$ does not appear in $n$

$n > (MAX\ v: D\cdot m) \Rightarrow (\forall v: D\cdot n{>}m)$

$n < (MIN\ v: D\cdot m) \Rightarrow (\forall v: D\cdot n{<}m)$

$n \ge (MAX\ v: D\cdot m) = (\forall v: D\cdot n{\ge}m)$

$n \le (MIN\ v: D\cdot m) = (\forall v: D\cdot n{\le}m)$

$n \ge (MIN\ v: D\cdot m) \Leftarrow (\exists v: D\cdot n{\ge}m)$

$n \le (MAX\ v: D\cdot m) \Leftarrow (\exists v: D\cdot n{\le}m)$

$n > (MIN\ v: D\cdot m) = (\exists v: D\cdot n{>}m)$

$n < (MAX\ v: D\cdot m) = (\exists v: D\cdot n{<}m)$

Extreme Laws

$(MIN\ n: int\cdot n) = (MIN\ n: real\cdot n) = -\infty$

$(MAX\ n: int\cdot n) = (MAX\ n: real\cdot n) = \infty$

Connection Laws (Galois)

$n{\le}m = \forall k\cdot k{\le}n \Rightarrow k{\le}m$

$n{\le}m = \forall k\cdot k{<}n \Rightarrow k{<}m$

$n{\le}m = \forall k\cdot m{\le}k \Rightarrow n{\le}k$

$n{\le}m = \forall k\cdot m{<}k \Rightarrow n{<}k$

Distributive Laws — if $D{\neq}null$ and $v$ does not appear in $n$

$max\ n\ (MAX\ v: D\cdot m) = (MAX\ v: D\cdot max\ n\ m)$

$max\ n\ (MIN\ v: D\cdot m) = (MIN\ v: D\cdot max\ n\ m)$

$min\ n\ (MAX\ v: D\cdot m) = (MAX\ v: D\cdot min\ n\ m)$

$min\ n\ (MIN\ v: D\cdot m) = (MIN\ v: D\cdot min\ n\ m)$

$n + (MAX\ v: D\cdot m) = (MAX\ v: D\cdot n{+}m)$

$n + (MIN\ v: D\cdot m) = (MIN\ v: D\cdot n{+}m)$

$n - (MAX\ v: D\cdot m) = (MIN\ v: D\cdot n{-}m)$

$n - (MIN\ v: D\cdot m) = (MAX\ v: D\cdot n{-}m)$

$(MAX\ v: D\cdot m) - n = (MAX\ v: D\cdot m{-}n)$

$(MIN\ v: D\cdot m) - n = (MIN\ v: D\cdot m{-}n)$

$n{\ge}0 \Rightarrow n \times (MAX\ v: D\cdot m) = (MAX\ v: D\cdot n{\times}m)$

$n{\ge}0 \Rightarrow n \times (MIN\ v: D\cdot m) = (MIN\ v: D\cdot n{\times}m)$

$n{\le}0 \Rightarrow n \times (MAX\ v: D\cdot m) = (MIN\ v: D\cdot n{\times}m)$

$n{\le}0 \Rightarrow n \times (MIN\ v: D\cdot m) = (MAX\ v: D\cdot n{\times}m)$

$n \times (\Sigma v: D\cdot m) = (\Sigma v: D\cdot n{\times}m)$

$(\Pi v: D\cdot m)^n = (\Pi v: D\cdot m^n)$

End of Quantifiers

### 11.4.9 Limits

$(MAX\ m\cdot\ MIN\ n\cdot\ f(m+n))\ \le\ (LIM\ f)\ \le\ (MIN\ m\cdot\ MAX\ n\cdot\ f(m+n))$

$\exists m\cdot\ \forall n\cdot\ p(m+n)\ \Longrightarrow\ LIM\ p\ \Longrightarrow\ \forall m\cdot\ \exists n\cdot\ p(m+n)$

$(LIM\ n\cdot\ n) = \infty$

End of Limits

### 11.4.10 Specifications and Programs

For specifications $P$ , $Q$ , $R$ , and $S$ , and binary $b$ ,

$ok\ =\ x'=x\ \land\ y'=y\ \land\ ...$

$x{:=}\ e\ =\ x'=e\ \land\ y'=y\ \land\ ...$

$P.\ Q\ =\ \exists x'', y'', ...\cdot\ \langle x', y', ...\rightarrow P\rangle\ x''\ y''\ ...\ \land\ \langle x, y, ...\rightarrow Q\rangle\ x''\ y''\ ...$

$P\|Q\ =\ \exists t_P, t_Q\cdot\ \langle t'\rightarrow P\rangle t_P\ \land\ \langle t'\rightarrow Q\rangle t_Q\ \land\ t' = max\ t_P\ t_Q$

$\textbf{if}\ b\ \textbf{then}\ P\ \textbf{else}\ Q\ \textbf{fi}\ =\ b \land P \lor \lnot b \land Q$

$\textbf{var}\ x{:}\ T\cdot\ P\ =\ \exists x, x'{:}\ T\cdot\ P$

$\textbf{frame}\ x\cdot\ P\ =\ P\ \land\ y'=y\ \land\ ...$

$\textbf{while}\ b\ \textbf{do}\ P\ \textbf{od}\ =\ t'\ge t \land \textbf{if}\ b\ \textbf{then}\ P.\ \ t{:=}\ t{+}1.\ \ \textbf{while}\ b\ \textbf{do}\ P\ \textbf{od}\ \textbf{else}\ ok\ \textbf{fi}$

$\forall\sigma, \sigma'\cdot\ \textbf{if}\ b\ \textbf{then}\ P.\ \ W\ \textbf{else}\ ok\ \textbf{fi}\ \Leftarrow\ W\ \Longrightarrow\ \forall\sigma, \sigma'\cdot\ \textbf{while}\ b\ \textbf{do}\ P\ \textbf{od}\ \Leftarrow\ W$

$\qquad (Fmn\ \Leftarrow\ m{=}n \land ok) \land (Fik\ \Leftarrow\ m{\le}i{<}j{<}k{\le}n \land (Fij.\ Fjk))$

$\Longrightarrow\quad Fmn\ \Leftarrow\ \textbf{for}\ i{:=}\ m;..n\ \textbf{do}\ m{\le}i{<}n \Rightarrow Fi(i{+}1)\ \textbf{od}$

$Im{\Rightarrow}I'n\ \Longleftarrow\ \textbf{for}\ i{:=}\ m;..n\ \textbf{do}\ m{\le}i{<}n \land Ii \Rightarrow I'(i{+}1)\ \textbf{od}$

$\textbf{wait until}\ w\ =\ t{:=}\ max\ t\ w$

$\textbf{assert}\ b\ =\ \textbf{if}\ b\ \textbf{then}\ ok\ \textbf{else}\ print$ “error”. $\textbf{wait until}\ \infty\ \textbf{fi}$

$\textbf{ensure}\ b\ =\ b \land ok$

$P.\ (P\ \textbf{result}\ e){=}e$ but do not double-prime or substitute in $(P\ \textbf{result}\ e)$

$c?\ =\ r{:=}\ r{+}1$

$c\ =\ \mathbb{M}c_{\ rc-1}$

$c!\ e\ =\ \mathbb{M}c_{\ wc} = e\ \land\ \mathbb{T}c_{\ wc} = t\ \land\ (wc{:=}\ wc{+}1)$

$\surd c\ =\ \mathbb{T}c_{\ rc} + \text{(transit time)} \le t$

$\textbf{ivar}\ x{:}\ T\cdot\ S\ =\ \exists x{:}\ time{\rightarrow}T\cdot\ S$

$\textbf{chan}\ c{:}\ T\cdot\ P\ =\ \exists\mathbb{M}c{:}\ \infty{*}T\cdot\ \exists\mathbb{T}c{:}\ \infty{*}xreal\cdot\ \textbf{var}\ rc\ , wc{:}\ xnat := 0\cdot\ P$

| | |
|---|---|
| $ok.\ P\ =\ P.\ ok\ =\ P$ | identity |
| $P.\ (Q.\ R)\ =\ (P.\ Q).\ R$ | associativity |
| $P{\lor}Q.\ R{\lor}S\ =\ (P.\ R) \lor (P.\ S) \lor (Q.\ R) \lor (Q.\ S)$ | distributivity |
| $\textbf{if}\ b\ \textbf{then}\ P\ \textbf{else}\ Q\ \textbf{fi}.\ R\ =\ \textbf{if}\ b\ \textbf{then}\ P.\ R\ \textbf{else}\ Q.\ R\ \textbf{fi}$ | distributivity (unprimed $b$ ) |
| $P.\ \ \textbf{if}\ b\ \textbf{then}\ Q\ \textbf{else}\ R\ \textbf{fi}\ =\ \textbf{if}\ P.\ b\ \textbf{then}\ P.\ Q\ \textbf{else}\ P.\ R\ \textbf{fi}$ | distributivity (unprimed $b$ ) |
| $P \parallel Q\ =\ Q \parallel P$ | symmetry |
| $P \parallel (Q \parallel R)\ =\ (P \parallel Q) \parallel R$ | associativity |
| $P \parallel t'{=}t\ =\ P\ =\ t'{=}t \parallel P$ | identity |
| $P \parallel Q{\lor}R\ =\ (P \parallel Q) \lor (P \parallel R)$ | distributivity |
| $P \parallel \textbf{if}\ b\ \textbf{then}\ Q\ \textbf{else}\ R\ \textbf{fi}\ =\ \textbf{if}\ b\ \textbf{then}\ P \parallel Q\ \textbf{else}\ P \parallel R\ \textbf{fi}$ | distributivity |
| $\textbf{if}\ b\ \textbf{then}\ P\|Q\ \textbf{else}\ R\|S\ \textbf{fi}\ =\ \textbf{if}\ b\ \textbf{then}\ P\ \textbf{else}\ R\ \textbf{fi} \parallel \textbf{if}\ b\ \textbf{then}\ Q\ \textbf{else}\ S\ \textbf{fi}$ | distributivity |
| $x{:=}\ \textbf{if}\ b\ \textbf{then}\ e\ \textbf{else}\ f\ \textbf{fi}\ =\ \textbf{if}\ b\ \textbf{then}\ x{:=}\ e\ \textbf{else}\ x{:=}\ f\ \textbf{fi}$ | functional-imperative |

End of Specifications and Programs

### 11.4.11 Substitution

Let $x$ and $y$ be different boundary state variables, let $e$ and $f$ be expressions of the prestate, and let $P$ be a specification.

$x{:=}\,e.\,P \;=\;$ (for $x$ substitute $e$ in $P$)

$(x{:=}\,e \,\|\, y{:=}\,f).\; P \;=\;$ (for $x$ substitute $e$ and independently for $y$ substitute $f$ in $P$)

End of Substitution

### 11.4.12 Conditions

Let $P$ and $Q$ be any specifications, and let $C$ be a precondition, and let $C'$ be the corresponding postcondition (in other words, $C'$ is the same as $C$ but with primes on all the state variables).

$C \wedge (P.\,Q) \;\Longleftarrow\; C{\wedge}P.\,Q$

$C \Rightarrow (P.Q) \;\Longleftarrow\; C{\Rightarrow}P.\,Q$

$(P.Q) \wedge C' \;\Longleftarrow\; P.\,Q{\wedge}C'$

$(P.Q) \Leftarrow C' \;\Longleftarrow\; P.\,Q{\Leftarrow}C'$

$P.\,C{\wedge}Q \;\Longleftarrow\; P{\wedge}C'.\,Q$

$P.\,Q \;\Longleftarrow\; P{\wedge}C'.\; C{\Rightarrow}Q$

$C$ is a sufficient precondition for $P$ to be refined by $S$
if and only if $C{\Rightarrow}P$ is refined by $S$ .

$C'$ is a sufficient postcondition for $P$ to be refined by $S$
if and only if $C'{\Rightarrow}P$ is refined by $S$ .

End of Conditions

### 11.4.13 Refinement

Refinement by Steps (Stepwise Refinement) (monotonicity, transitivity)

If $A \Longleftarrow$ **if** $b$ **then** $C$ **else** $D$ **fi** and $C \Longleftarrow E$ and $D \Longleftarrow F$ are theorems,
then $A \Longleftarrow$ **if** $b$ **then** $E$ **else** $F$ **fi** is a theorem.

If $A \Longleftarrow B.C$ and $B \Longleftarrow D$ and $C \Longleftarrow E$ are theorems, then $A \Longleftarrow D.E$ is a theorem.

If $A \Longleftarrow B\|C$ and $B \Longleftarrow D$ and $C \Longleftarrow E$ are theorems, then $A \Longleftarrow D\|E$ is a theorem.

If $A \Longleftarrow B$ and $B \Longleftarrow C$ are theorems, then $A \Longleftarrow C$ is a theorem.

Refinement by Parts (monotonicity, conflation)

If $A \Longleftarrow$ **if** $b$ **then** $C$ **else** $D$ **fi** and $E \Longleftarrow$ **if** $b$ **then** $F$ **else** $G$ **fi** are theorems,
then $A{\wedge}E \Longleftarrow$ **if** $b$ **then** $C{\wedge}F$ **else** $D{\wedge}G$ **fi** is a theorem.

If $A \Longleftarrow B.C$ and $D \Longleftarrow E.F$ are theorems, then $A{\wedge}D \Longleftarrow B{\wedge}E.\,C{\wedge}F$ is a theorem.

If $A \Longleftarrow B\|C$ and $D \Longleftarrow E\|F$ are theorems, then $A{\wedge}D \Longleftarrow B{\wedge}E \,\|\, C{\wedge}F$ is a theorem.

If $A \Longleftarrow B$ and $C \Longleftarrow D$ are theorems, then $A{\wedge}C \Longleftarrow B{\wedge}D$ is a theorem.

Refinement by Cases

$P \Longleftarrow$ **if** $b$ **then** $Q$ **else** $R$ **fi** is a theorem if and only if
$P \Longleftarrow b \wedge Q$ and $P \Longleftarrow \neg b \wedge R$ are theorems.

End of Refinement

End of Laws

# 11.5 Names

| | |
|---|---|
| $abs$: $xreal \to \S r: xreal \cdot r \ge 0$ | $abs\ r\ =\ $ **if** $r \ge 0$ **then** $r$ **else** $-r$ **fi** |
| $bin$ (the binary values) | $bin\ =\ \top, \bot$ |
| $ceil$: $real \to int$ | $r \le ceil\ r < r+1$ |
| $char$ (the characters) | $char\ =\ ..., “a”, “A”, ...$ |
| $div$: $real \to (\S r: real \cdot r>0) \to int$ | $div\ x\ y\ =\ floor\ (x/y)$ |
| $divides$: $(nat+1) \to int \to bin$ | $divides\ n\ i\ =\ i/n: int$ |
| $entro$: $prob \to \S r: xreal \cdot r \ge 0$ | $entro\ p\ =\ p \times info\ p\ +\ (1-p) \times info\ (1-p)$ |
| $even$: $int \to bin$ | $even\ i\ =\ i/2: int$ |
| | $even\ =\ divides\ 2$ |
| $floor$: $real \to int$ | $floor\ r \le r < floor\ r + 1$ |
| $info$: $prob \to \S r: xreal \cdot r \ge 0$ | $info\ p\ =\ -\ log\ p$ |
| $int$ (the integers) | $int\ =\ nat, -nat$ |
| $LIM$ (limit quantifier) | see Laws |
| $log$: $(\S r: xreal \cdot r \ge 0) \to xreal$ | $log\ (2^x) = x$ |
| | $log\ (x \times y)\ =\ log\ x + log\ y$ |
| $max$: $xrat \to xrat \to xrat$ | $max\ x\ y\ =\ $ **if** $x \ge y$ **then** $x$ **else** $y$ **fi** |
| | $-\ max\ a\ b\ =\ min\ (-a)\ (-b)$ |
| $MAX$ (maximum quantifier) | see Laws |
| $min$: $xrat \to xrat \to xrat$ | $min\ x\ y\ =\ $ **if** $x \le y$ **then** $x$ **else** $y$ **fi** |
| | $-\ min\ a\ b\ =\ max\ (-a)\ (-b)$ |
| $MIN$ (minimum quantifier) | see Laws |
| $mod$: $real \to (\S r: real \cdot r>0) \to real$ | $0 \le mod\ a\ d < d$ |
| | $a\ =\ div\ a\ d \times d\ +\ mod\ a\ d$ |
| $nat$ (the naturals) | $0, nat+1: nat$ |
| | $0, B+1: B\ \Rightarrow\ nat: B$ |
| $nil$ (the empty string) | $\leftrightarrow nil\ =\ 0$ |
| | $nil; S\ =\ S\ =\ S; nil$ |
| | $nil \le S$ |
| $null$ (the empty bunch) | $¢null\ =\ 0$ |
| | $null, A\ =\ A\ =\ A, null$ |
| | $null: A$ |
| $odd$: $int \to bin$ | $odd\ i\ =\ \neg\ i/2: int$ |
| | $odd\ =\ \neg even$ |
| $ok$ (the empty program) | $ok\ =\ \sigma'=\sigma$ |
| | $ok.P\ =\ P\ =\ P.ok$ |
| $prob$ (probability) | $prob\ =\ \S r: real \cdot 0 \le r \le 1$ |
| $rand$ (random number) | $rand\ n: 0,..n$ |
| $rat$ (the rationals) | $rat\ =\ int/(nat+1)$ |
| $real$ (the reals) | $r: real\ =\ r: xreal\ \wedge\ -\infty<r<\infty$ |
| $suc$: $nat \to (nat+1)$ | $suc\ n = n+1$ |
| $xint$ (the extended integers) | $xint\ =\ -\infty, int, \infty$ |
| $xnat$ (the extended naturals) | $xnat\ =\ nat, \infty$ |
| $xrat$ (the extended rationals) | $xrat\ =\ -\infty, rat, \infty$ |
| $xreal$ (the extended reals) | $x: xreal\ =\ \exists f: nat \to rat \cdot x = LIM\ f$ |

End of Names

## 11.6 Symbols

| | | |
|---|---|---|
| ⊤ | 3 | true |
| ⊥ | 3 | false |
| ¬ | 3 | not |
| ∧ | 3 | and |
| ∨ | 3 | or |
| ⇒ | 3 | implies |
| ⟹ | 3 | implies |
| ⇐ | 3 | follows from, is implied by |
| ⟸ | 3 | follows from, is implied by |
| = | 3 | equals, if and only if |
| ═ | 3 | equals, if and only if |
| ⧧ | 3 | differs from, is unequal to |
| < | 13 | less than |
| > | 13 | greater than |
| ≤ | 13 | less than or equal to |
| ≥ | 13 | greater than or equal to |
| + | 12 | plus |
| + | 20 | list catenation |
| – | 12 | minus |
| × | 12 | times, multiplication |
| / | 12 | divided by |
| , | 14 | bunch union |
| ,.. | 16 | union from (incl) to (excl) |
| ‘ | 14 | bunch intersection |
| ; | 17 | string catenation |
| ;.. | 19 | catenation from (incl) to (excl) |
| : | 14 | is in, are in, bunch inclusion |
| :: | 89 | includes |
| := | 36 | assignment |
| . | 36 | dep't (sequential) composition |
| · | 26 | quantifier abbreviation |
| ! | 133 | output |
| ? | 133 | input |

| | | |
|---|---|---|
| √ | 133 | input check |
| ( ) | 4 | parentheses for grouping |
| { } | 17 | set brackets |
| [ ] | 20 | list brackets |
| ⟨ ⟩ | 23 | function (scope) brackets |
| ϟ | 17 | power |
| ¢ | 14 | bunch size, cardinality |
| $ | 17 | set size, cardinality |
| ↔ | 18 | string size, length |
| # | 20 | list size, length |
| \| | 20,24 | selective union, otherwise |
| \|\| | 118 | indep't (parallel) composition |
| ~ | 17,20 | contents of a set or list |
| * | 18 | repetition of a string |
| □ | 23 | domain of a function |
| → | 23 | function arrow |
| ∈ | 17 | element of a set |
| ⊆ | 17 | subset |
| ∪ | 17 | set union |
| ∩ | 17 | set intersection |
| @ | 22 | index with a pointer |
| ∀ | 26 | for all, universal quantifier |
| ∃ | 26 | there exists, existential quantifier |
| Σ | 26 | sum of, summation quantifier |
| Π | 26 | product of, product quantifier |
| § | 28 | those, solution quantifier |
| ′ | 34 | $x'$ is final value of state var $x$ |
| “ ” | 13,19 | “hi” is a text or string of chars |
| $a^b$ | 12 | exponentiation |
| $a_b$ | 18 | string indexing |
| $a\ b$ | 20,31 | indexing,application,composition |
| ◃ ▹ | 18 | string modification |
| ∞ | 12 | infinity |

| | |
|---|---|
| **assert** | 77 |
| **chan** | 138 |
| **do od** | 71 |
| **ensure** | 77 |
| **exit when** | 71 |
| **for do od** | 74 |
| **frame** | 67 |
| **go to** | 75 |
| **if then else fi** | 4 |
| **ivar** | 126 |
| **or** | 77 |
| **result** | 78 |
| **var** | 66 |
| **wait until** | 76 |
| **while do od** | 69 |

End of Symbols

# 11.7 Precedence

| | |
|---|---|
| 0 | ⊤ ⊥ ( ) { } [ ] ⟨ ⟩ **if fi** **do od** number text name superscript subscript |
| 1 | @ juxtaposition |
| 2 | prefix– ¢ $ ↔ # * ~ ↯ □ → √ |
| 3 | × / ∩ |
| 4 | + infix– + ∪ |
| 5 | ; ;.. ‘ |
| 6 | , ,.. \| ◁ ▷ |
| 7 | = ⧧ < > ≤ ≥ : :: ∈ ⊆ |
| 8 | ¬ |
| 9 | ∧ |
| 10 | ∨ |
| 11 | ⇒ ⇐ |
| 12 | := ! ? |
| 13 | **exit when** **go to** **wait until** **assert** **ensure** **or** |
| 14 | . ‖ **result** |
| 15 | ∀· ∃· Σ· Π· §· *LIM*· *MAX*· *MIN*· **var**· **ivar**· **chan**· **frame**· |
| 16 | **=** **⟹** **⟸** |

Superscripting and subscripting serve to bracket all operations within them.

Juxtaposition associates from left to right, so *a b c* means the same as (*a b*) *c* . The infix operators @ / – associate from left to right. The infix operators * → associate from right to left. The infix operators × ∩ + + ∪ ; ‘ , | ∧ ∨ . ‖ are associative (they associate in both directions).

On levels 7, 11, and 16 the operators are continuing. For example, $a=b=c$ neither associates to the left nor associates to the right, but means the same as $a=b \land b=c$ . On any one of these levels, a mixture of continuing operators can be used. For example, $a \le b < c$ means the same as $a \le b \land b < c$ .

The operators **=** **⟹** **⟸** are identical to = ⇒ ⇐ except for precedence.

End of Precedence

# 11.8 Distribution

The operators in the following expressions distribute over bunch union in any operand:

[*A*] *A*@*B* *A B* –*A* $*A* ↔*A* #*A* ~*A*

$A^B$ $A_B$ *A*×*B* *A*/*B* *A*∩*B* *A*+*B* *A*–*B* *A*+*B* *A*∪*B* *A*;*B* *A*‘*B*

¬*A* *A*∧*B* *A*∨*B*

The operator in *A***B* distributes over bunch union in its left operand only.

End of Distribution

End of Reference

End of a Practical Theory of Programming