$$(a \wedge b)\Big|_{((a\ (p \wedge q))(b\ \ a))}$$

$$\equiv ((p \wedge q) \wedge a)$$

$$\cancel{(p \wedge \neg p)\Big|_{((p\ t)(p\ nil))}}$$

Conjecture
Theorem
Axiom
Corollary
Lemma

Sat.

$$(a \wedge b)\Big|_{((a\ t)(b\ t))}$$

Falsifiable

$$(a \wedge b)\Big|_{((a\ nil)(b\ t))}$$

Instantiation

$$(p \oplus q) \wedge \neg (p \oplus q)$$

$$\equiv \{\, p \wedge \neg p \equiv nil \,\Big|_{((p\ \ (p \oplus q)))}$$

nil

Conjecture

$$(p \oplus q) \wedge \neg (p \oplus q) \equiv nil$$

$$\boxed{\text{(listp z)} \Rightarrow \text{(equal (len (cons x z))} \\ \text{(len (cons y z)))}}$$

$P \Rightarrow Q$

Proof

C.I. (listp z)

Prove $\Rightarrow$

Assume P

Prove Q

Use $\Rightarrow$

Show P is true

conclude Q

Modus Ponens

```
(defunc alen (x)
  :input-contract t
  :output-contract (natp (alen x))
  (if (atom x)
      0
      (+ 1 (alen (rest x)))))
```

]] or } or whatever ≡ magic parens

conjecture 1:

(alen x) = (alen (list x))

0 1

Counter: x = 4

Conjecture 2

(alen (cons x z)) = (alen (cons y z))

Lemma 1

(alen (cons x z)) ≡ (+ 1 (alen z))

Lemma 1

LHS

(alen (cons x z))

≡ { Def alen |((x (cons x z))) }

(if (atom (cons x z))
    0
    (+ 1 (alen (rest (cons x z)))))

≡ { Def atom |((x (cons x z))) }

(if (not (consp (cons x z)))
    0
    (+ 1 (alen (rest (cons x z)))))

≡ { Def not |((a (consp (cons x z)))) }

(if (if (consp (cons x z)) nil t) 0 (+1 (alen (rest (cons x z)))))

≡ { if axiom, consp axiom }

(if nil 0 (+1 (alen (rest (cons x z)))))

≡ { if axiom }

(+1 (alen (rest (cons x z))))

≡ { first-rest axioms }

$\boxed{\text{(+ 1 (alen z))}}$ ⟸ RHS

$\boxed{\text{LHS = RHS Q.E.D.}}$

RHS (alen (cons y z))

≡ {

LHS of conjecture

(alen (cons x z))

≡ { Lemma 1 | ((x x)) }

(+1 (alen z))

RHS

(alen (cons y z))

≡ { Lemma 1 | ((x y)) }

(+ 1 (alen z))

LHS = RHS Q.E.D.

Reflexivity

$$X \equiv X$$

Symmetry of Equality

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

Transitivity of Equality

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

if axioms

$$(x = nil) \Rightarrow (\text{if } x\ y\ z) \equiv z$$

$$(x \neq nil) \Rightarrow (\text{if } x\ y\ z) \equiv y$$

t

consp axiom

$$(\text{consp } (\text{cons } x\ y)) \equiv t$$

first-rest axioms

$$(\text{first } (\text{cons } x\ y)) \equiv x$$

$$(\text{rest } (\text{cons } x\ y)) \equiv y$$

```
(defunc f (x1 x2 ... xn)
  :input-contract (and (R1 x1) (R2 x2) ... (Rn xn))
  :output-contract (R (f x1 x2 ... xn))
  <body>)
```

$IC \Rightarrow OC$ contract theorem eg $(\text{and } (R_1\ x_1) \ldots (R_n\ x_n)) \Rightarrow (R\ (f\ x_1 \ldots x_n))$

$IC \Rightarrow ((f\ x_1\ x_2 \ldots x_n) = \langle \text{body} \rangle)$

Equality Axiom Schema for Functions

$x_1 = y_1 \wedge x_2 = y_2 \ldots x_n = y_n \Rightarrow$
$(f\ x_1 \ldots x_n) = (f\ y_1 \ldots y_n)$

$(\text{equal } l1\ l2) \equiv (l1 = l2) \Longleftarrow$ Shortcut!

Exportation

$(\text{listp } x) \wedge (\text{listp } y) \wedge (\text{listp } z) \ \cancel{\Rightarrow}\ \wedge$

$((\text{endp } x) \Rightarrow (\text{app } (\text{app } x\ y)\ z) = (\text{app } x\ \overbrace{(\text{app } y\ z)}^{\text{"}y\text{"}})))$

C1. (listp x)
C2. (listp y)
C3. (listp z)
C4 (endp x)

} General Context

C5. $(x = \text{nil})$ {C1, C4, Def listp}

① $LHS = x$, $RHS = x$ } $LHS = RHS$

② $LHS = RHS$

$x \stackrel{?}{=} x$

③ $LHS = RHS$.

(app (app x y) z)

≡ {Def of app, if axiom, C4}

(app $\boxed{y}$ z)

≡ {if axiom | ((x (endp x)) (y (app y z)) (z (cons (first x) (app (rest x) (app y z))))), C4}

```
(if (endp x)
    (app y z)
    (cons (first x) (app (rest x) (app y z))))
```

≡ {Def app | ((y (app y z)))}

(app x (app y z))

Theorem 4.1.

(listp y) $\wedge$ (listp z) $\Rightarrow$ (app (cons x y) z) = (cons x (app y z))

```
(defunc app (x y)
  :ic (and (listp x) (listp y))
  :oc (listp (app x y))
  (if (endp x)
      y
    (cons (first x) (app (rest x) y))))
```

(conjecture $\varphi_{assoc\_app_2}$

(consp x) $\wedge$ [(app (app (rest x) y) z)

= (app (rest x) (app y z))]

Jerk move

$\Rightarrow$ (app (app x y) z) = (app x (app y z)))

$$(\text{listp } x) \land (\text{listp } y) \land (\text{listp } z) \land (\text{consp } x)$$
$$\land \underbrace{\left[(\text{listp}(\text{rest } x)) \land (\text{listp } y) \land (\text{listp } z) \Rightarrow (\text{app}\,(\text{app}\,(\text{rest } x)\, y)\, z) = (\text{app}\,(\text{rest } x)\,(\text{app } y\, z))\right]}_{\alpha}$$
$$\Rightarrow ((\text{app}\,(\text{app } x\, y)\, z) = (\text{app } x\,(\text{app } y\, z)))$$

$$A \Rightarrow (B \Rightarrow (C \Rightarrow D))$$
$$\equiv A \Rightarrow (B \land C \Rightarrow D)$$
$$\equiv A \land B \land C \Rightarrow D$$
$$\Rightarrow E)$$

C1. (listp x)
C2. (listp y)
C3. (listp z)
C4. (consp x)
C5 α

C6 (listp (rest x)) {C1, C4, Def listp}

C7. (app (app (rest x) y) z)
= (app (rest x) (app y z))
{C6, C2, C3, C5, MP}

Modus Ponens

$$[(A \Rightarrow B) \land A] \Rightarrow B$$

C8. (not (endp x))
{C4, C1, def endp}

(app (app x y) z)

≡ {Def app, if ~~axiom~~, C8}

Lazy

(app (cons (first x) (app (rest x) y)) z)

≡ {Theorem 4.1 ((y (app (rest x) y))) }

(cons (first x) (app (app (rest x) y) z))

C7!

≡ {C7}

(cons (first x) (app (rest x) (app y z)))

"y"

≡ {Def app, if ~~axiom~~, C8}

(app x (app y z))

QED.

Assoc of App

(listp x) ∧ (listp y) ∧ (listp z)

⇒ [(app (app x y) z)

= (app x (app y z))]

$\wedge$ (endp x) $\wedge$ (consp x) $\wedge$ [———] $\Rightarrow$ [(app (app x y) z) = (app x (app y z))]

(listp x) $\wedge$ (listp y) $\wedge$ (listp z)

C1. (listp x)
C2. (listp y)
C3. (listp z)
C4. (endp x)
C5. (consp x)
C6. [———]

- - - - - - - - - - - - - - - - - - -

C7. (not (consp x)) {Def endp, C4}
C8. nil {C7, C5, PL}

nil $\Rightarrow$ X $\equiv$ true

QED

$\Phi_{in\,app}$:

$$(\text{not}\,(\text{in}\ a\ (\text{app}\ x\ y))) \Rightarrow (\text{not}\,(\text{in}\ a\ x))$$

$$\Downarrow$$

$$(\text{listp}\ x) \land (\text{listp}\ y) \ \cancel{\Rightarrow}\!\land\ (\text{in}\ a\ x) \Rightarrow (\text{in}\ a\ (\text{app}\ x\ y))$$

```
(defunc in (e X)
  :ic (listp X)
  :oc (booleanp (in e x))
(if (endp x)
    nil
    (or (equal e (first x))
        (in e (rest x)))))
```

$$\Phi_{in\_app2}: \Big[\big((\text{endp}\ x) \lor (\neg(\text{endp}\ x) \land (a = (\text{first}\ x)))$$
$$\lor\ (\neg(\text{endp}\ x) \land (a \neq (\text{first}\ x)) \land \Phi_{in\_app}\big|_{((x\ (\text{rest}\ x)))}\big)\Big]$$

$$\land\ (\text{listp}\ x) \land (\text{listp}\ y) \land (\text{in}\ a\ x) \Rightarrow (\text{in}\ a\ (\text{app}\ x\ y))$$

Inductive Assumption

Case 1: endp x

$(\text{endp } X) \wedge (\text{listp } X) \wedge (\text{listp } y) \wedge (\text{in } a\, X) \Rightarrow (\text{in } a\, (\text{app } X\, y))$

C1. (endp X)

C2. (listp X)

C3. (listp y)

C4. (in a X)

- - - - - - - - - -

C5. (not (in a X)) {C1, Def in, if axiom}

C6. nil {C5, C4, PL}

$\text{nil} \Rightarrow X \equiv \text{true}$

$\neg(\text{endp } x) \wedge (\text{listp } x) \wedge (\text{listp } y) \wedge (a = (\text{first } x)) \wedge (\text{in } a\, X) \Rightarrow (\text{in } a\, (\text{app } X\, y))$

C1. (listp X)

C2. (listp y)

C3. $\neg$(endp X)

C4. (a = (first X))

C5. (in a X)

(in a (app x y))

≡ {Def of app, C3}

"x"

(in a (cons (first x) (app (rest x) y)))

≡ {Def in, Def endp, cons axiom, if axiom}

(or (equal a (first (cons (first x) (app (rest x) y))))

(in a (rest (cons (first x) (app (rest x) y))))).

≡ {first-rest axioms}

(or (equal a (first x)) (in a (app (rest x) y)))

≡ {C4, PL}

true

Case 3:

$(\text{listp } x) \wedge (\text{listp } y) \wedge \neg(\text{endp } x) \wedge (a \neq (\text{first } x)) \wedge [(\text{listp } (\text{rest } x)) \wedge (\text{listp } y) \wedge (\text{in } a (\text{rest } x)) \Rightarrow (\text{in } a (\text{app} (\text{rest } x)\ y))] \wedge (\text{in } a\ x)$

α

$\Rightarrow (\text{in } a (\text{app } x\ y))$

$((A \Rightarrow B) \wedge A) \Rightarrow B$

C1. (listp x)

C2. (listp y)

C3. $\neg$(endp x)

C4. $(a \neq (\text{first } x))$

C5. α

C6. (in a x)

C7. (listp (rest x)) {C1, C3, Def listp, Def endp}

C8. (in a (rest x)) {C6, Def in, C3, PL, C4}

C9. (in a (app (rest x) y)) {C5, C8, C7, C2, MP}

(in a (app x y))

= {Def app, C3}

(in a (cons (first x) (app (rest x) y)))

"x"

≡ {def in, def endp, consp axiom, first-rest axioms}

(or (equal a (first x))
    (in a (app (rest x) y)))

≡ {PL, C4}

(in a (app (rest x) y))

≡ {C9}

true

$\phi_{in\text{-}app}$

$$\overset{A}{(\text{listp } x)} \wedge \overset{B}{(\text{listp } y)} \wedge \overset{C}{(\text{in } a\ x)} \Rightarrow \overset{G}{(\text{in } a\ (\text{app } x\ y))}$$

$\wedge$ true

$\equiv$ (endp x) $\vee$ $\neg$(endp x)

$\equiv$ $\underset{D}{\text{(endp x)}}$ $\vee$ ($\neg$(endp x) $\wedge$ (a = (first x))) E

$\vee$ ($\neg$(endp x) $\wedge$ (a $\neq$ (first x))) F

$$A \wedge B \wedge C \wedge (D \vee E \vee F) \Rightarrow G$$

$$\equiv (A \wedge B \wedge C \wedge D)$$
$$\vee (A \wedge B \wedge C \wedge E)$$
$$\vee (A \wedge B \wedge C \wedge F) \Rightarrow G$$

C1. (listp x)

C2. (listp y)

C3. (in a x)

- - - - - - - - - - - -

C4. nil $\{C1|_{((x\ 4))}\}$

$nil \Rightarrow \varphi \equiv true$

QED.

φ_{BAD} $x=1 \Rightarrow 0=1$

c1 $x=1$

$\overset{0}{\underset{1}{\equiv}} \{c1|_{((x\ 0))}\}$

QED

$nil \Rightarrow \varphi \equiv \underline{true}$

$(0=1) \Rightarrow \varphi \equiv true$

$\{\varphi_{BAD}|_{((x\ 1))}, MP\}$

$\equiv true \Rightarrow \varphi$

Thus $\varphi \equiv true$

ϕ_{del_in}: (listp x) $\wedge$ ~~(in a x)~~ $\Rightarrow$ (not (in a (dela a x)))

Case 1 (endp x)

Case 2 $\neg$(endp x) $\wedge$ (a = (first x)) $\wedge$ β

Case 3 $\neg$(endp x) $\wedge$ (a $\neq$ (first x))

$\wedge$ [(listp (rest x)) $\wedge$ ~~(in a (rest x))~~ $\Rightarrow$ (not (in a (del a (rest x))))] β

$\Rightarrow$ change function to make valid

$\Rightarrow$ make simpler

```
(defunc dela (a x)
  :ic (listp x)
  :oc (listp (dela a x))
  (cond ((endp x)   x)
        ((equal a (first x))
           (dela a (rest x)))
        (t  (cons (first x) (dela a (rest x))))))
```

```
(defunc in (a x)
  :ic (listp x)
  :oc (booleanp (in a x))
  (cond ((endp x) nil)
        ((equal a (first x)) t)
        (t (in a (rest x)))))
```

C1. (listp x)
C2. (in a x)
C3. (a = (first x))
C4. ¬(endp x)

(not (in a (del a x)))
≡ {Def del, C4, C3}
(not (in a (rest x)))

Case 1

C1. (listp x)
C2. (endp x)

(not (in a (dela a x)))
≡ {Def dela, C2}
(not (in a x))
≡ {Def in, C2}
(not nil)
≡ {P2}
true

Case 2

C1. (listp x)
C2. ¬(endp x)
C3. (a = (first x))
C4. B
C5. (listp (rest x)) {Def listp, C2, C1}
C6. (not (in a (dela a (rest x))))
{C4, C5, MP}

(not (in a (dela a x)))
≡ {Def dela, C3, C2}
(not (in a (del a (rest x))))
≡ {C6}
true

Case 3

C1. (listp x)

C2. ¬(endp x)

C3. $(a \neq (\text{first } x))$

C4. B

- - - - -

C5. (listp (rest x)) {C1, C2, Def endp}

C6. (not (in a (dela a (rest x)))) {C5, C4, MP}

Proof (not (in a (dela a x)))

≡ {Def dela, C3, C2}

(not (in a (cons $\underbrace{\text{(first x)}}_{\text{first}}$ $\underbrace{\text{(dela a (rest x))}}_{\text{rest}}$))))

≡ {Def in, consp axiom, Def endp, first-rest axioms}

(not (if ~~(equal a (first x))~~

t

(in a (dela a (rest x)))))

≡ {C3}

(not (in a (dela a (rest x))))

≡ {C6}

true

Q.E.D.

$$\sum_{i=0}^{n} i = 0+1+2+3+\dots+n$$
$$+\ n+(n-1)+(n-2)+\dots 1+0$$
$$\underbrace{n+n+n+\dots n}_{n+1}$$

$$\sum_{i=0}^{n} i = \frac{n*(n+1)}{2}$$

```
(defunc sum (n)
  :ic (natp n)
  :oc (natp (sum n))
  (if (equal n 0)
      0
      (+ n (sum (- n 1)))))
```

```
(and (implies (and (natp n) (n=0))
              (equal (sum n) (/ (* n (+ n 1)) 2)))

     (implies (and (natp n) (n>0)
                   [implies (natp (- n 1)) (equal (sum (- n 1)) (/ (* (- n 1) n) 2))])
              (equal (sum n) (/ (* n (+ n 1)) 2))))
```

C1. (natp n)
C2. (n = 0)

(sum n)
≡ {Def sum, C2}
0
≡ {Arithmetic}
0 * 1 / 2
≡ {C2}
n * (n+1) / 2

Case 1

Case 2:

C1 (natp n)
C2. (n > 0)
C3. (natp (- n 1)) ⇒ (sum (- n 1)) = (n * (n-1) / 2)

C4. (natp (- n 1)) {C1, C2, Arith/Def natural numbers}
C5. (sum (- n 1) = (n * (n-1) / 2) {C3, C4, MP}
C6? (n ≠ 0)

(sum n)
≡ {Def sum, C2, Arithmetic}
(+ n (sum (- n 1))
≡ {C5}
(+ n (/ (* n (- n 1)) 2))

$\equiv$ {Arithmetic}

(/(+ (* 2 n) (* n (- n 1))) 2)

$\equiv$ {Arithmetic}

$$\frac{(n * (n+1))}{2}$$

QED

$$\frac{2n + (n * (n-1))}{2}$$

(app (app x (list a)) y) = (app x (cons a y)) (listp x) ∧ (listp y) ⇒ (app (app x (list a)) y) = (app x (cons a y))

C1. (listp x)

C2. (listp y)

(app (app x (list a)) y)

= { Assoc of app ((y (list a)) (z y)) }

(app x (app (list a) y))

= { Def list }

(app x (app (cons a nil) y))

= { Def app, consp axiom, Def endp, first-rest axioms }

(app x (cons a (app nil y)))

{ Def endp, def app }

(app x (cons a y))

Q.E.D.

ϕ_{even} $(natp\ b) \wedge (natp\ e) \Rightarrow (even\text{-}natp\ (exp\ b\ e)) = (even\text{-}natp\ b)$

$\phi_{even}\big|_{((b\ 2)(e\ 0))}$ · = nil or false ·

```
(defunc exp (b e)
  :ic (and (natp b)(natp e))
  :oc (natp (exp b e))
  (if (equal e 0)
      1
      (* b (exp b (- e 1)))))
```

```
(defunc no-dupesp (l)
  :ic (listp l)
  :oc (booleanp (no-dupesp l))
  (cond ((endp l) t)
        ((in (first l) (rest l)) nil)
        (t (no-dupesp (rest l)))))
```

(listp l) ∧ (no-dupesp l) ⇒ ((del a l) = (del-all a l))

Case 1

C1 (listp l)

C2. (no-dupesp l)

C3. (endp l)

LHS

(del a l)

= {C3, Def del}

l

RHS

(del-all a l)

= {C3, Def del-all}

l

LHS = RHS

$\varphi_{notin\text{-}delall}$: (listp l) $\wedge$ (not (in a l)) $\Rightarrow$ ((del-all a l) $\equiv$ l)

Case 2

C1. (listp l)

C2. (no-dupesp l)

C3. $\neg$(endp l)

C4. (a = (first l))

C5. (listp (rest l)) {C1, C3, Def listp}

C6. (not (in (first l) (rest l))) {C2, Def no-dupesp, C3, P2}

C7. (not (in a (rest l)) {C6, C4}

RHS

(del-all a l)

$\equiv$ {Def del-all, C4, C3}

(del-all a (rest l))

$\equiv$ {$\varphi_{notin\text{-}delall}|_{((l\ (rest\ l))}$, C7, C5}

MP if you want

(rest l)

LHS

(del a l)

$\equiv$ {Def del, C3, C4}

(rest l)

LHS = RHS

Case 3

C1. (listp l)

C2. (no-dupesp l)

C3. $\neg$(endp l)

C4. (a $\neq$ (first l))

C5. (listp (rest l)) $\wedge$ (no-dupesp (rest l)) $\Rightarrow$

(del-all a (rest l)) = (del a (rest l))

- - - - - - - - - -

C6. (listp (rest l)) {C1, C3, Def listp}

C7. (no-dupesp (rest l)) {Def no-dupesp, C2, C3, PL}

C8. (del-all a (rest l)) = (del a (rest l))

(del-all a l)

= {C4, C3}

(cons (first l) (del-all a (rest l)))

= {C8}

(cons (first l) (del a (rest l)))

= {Def del, C3, C4}

(del a l)

Q.E.D.