(perm (app x (cons e y)) (cons e (app x y)))

Using IS for listp

c1 (listp x)
c2 (listp y)
c3 (endp x)

(perm (app x (cons e y)) (cons e (app x y)))
= {Def app, c3}
(perm (cons e y) (cons e y))
= {perm identity a (listp x) => (perm x x) ... you can prove for practice}

t

Obligation 2

C1. (listp x)
C2. (listp y)
C3. ~(endp x)
C4. (listp (rest x)) ∧ (listp y) => (perm (app (rest x) (cons e y)) (cons e (app (rest x) y)))

C5 (listp (rest x)) {C1, C3, Def listp}
C6. (perm (app (rest x) (cons e y)) (cons e (app (rest x) y))
{C5, C4, C2, MP}

(perm (app x (cons e y)) (cons e (app x y))
= {Def app, C3}
(perm (cons (first x) (app (rest x) (cons e y))) (cons e (cons (first x) (app (rest x) y)

Case 1

C7. (e = (first x))

$\equiv$ {Def perm, C7}

(perm (app (rest x)(cons e y))(del (first x)(cons e (app x y))))

$\equiv$ {Def del, C7}

(perm (app (rest x)(cons e y))(~~cons e~~ (app x y)))

$\equiv$ {Def app, C7}

(perm (app (rest x)(cons e y))(cons e (app (rest x) y)))

$\equiv$ {C6}

t

Case 2 C7. (e ≠ (first x)) ← from last page

{Def perm, P7, C7, Def del}

(perm (app (rest x)(cons e y))(cons e (del (first x)(app x y))))

$\equiv$ {Def app, C4} ← def del

(perm (app (rest x)(cons e y))(cons e (app (rest x) y)))

$\equiv$ {C6}

t

Let's find a proof that actually needs IS for perm.

(lisp x) ∧ (lisp y) ⇒
~~(perm x y) ≡ (perm y x)~~

"Like a pro"

(perm ~~(app x y)~~ ~~(app y~~ x))
x (isort x))

≡

(perm x (insert (first x) (isort (rest x))))

≡ { Need theorem showing (in e (insert e x)) Lemma

also ~~can prove~~

Lemma (del e (insert e x)) ≡ x

(perm (rest x) (del (first x) (insert (first x) (isort (rest x)))))

≡ { the (del e (insert e x)) theorem }

(perm (rest x) (isort (rest x)))

≡ { Ind assumption }

t

Will use perm for I.S.

```
(defunc isort (l)
  :ic (lorp l)
  :oc (lorp (isort l))
  (cond ((endp l) l)
        (t (insert (first l) (isort (rest l))))))

(defunc insert (e l)
  :ic (and (rationalp e) (lorp l))
  :oc (lorp (insert e l))
  (cond ((endp l) (cons e l))
        ((< e (first l)) (cons e l))
        (t (cons (first l) (insert e
                                   (rest l))))))
```

OK. Now would need to prove proof obligations and ~~generalize to~~ do the formal proof

Plus prove the lemmata

Notes from today:

- HW9 due date set to Wednesday due to tail recursion
- HW10 still released Wednesday
- HW11 may be due later. Designed to help with exam
-