Homework1

Write a function that inverts the order of atoms

I tried to keep the original tree, but I got always dotted pairs. So I decided to just inverse the order of the atoms.

(defun rev-atx (s) ; function rev-atx: reverse-atoms
  (cond ((null s) '())
        ((atom s) (list s))
        (t (append (rev-atx (cdr s))
                    (rev-atx (car s))))
    )
)

Write a Lisp program that copies an arbitrary list data structure on all levels but atoms

This problem we solved in Class on Friday. I could trace it, but I did not find out how to write it into a file to be able to copy it.

(defun list-copyx (S)
  (cond ((atom S) S)
        (T (cons (list-copyx (car S)) (list-copyx (cdr S))))
    )
)

Write a test program for the following functions:

The required test program for the command I interpreted to play with the command and write code to show it. I tried to do so. Unfortunately I could not find the explanation for some of the required commands. I also had some warnings about the declaration of the variable a and b, which I could not yet eliminate.

The requested code for the program is the following:
Write a test of as many Lisp functions as you can do in one week.
You should invent tests for the following functions:
car, cdr, caddr, cons, list, set,setq, prog, progn, cond,
atom, number, greaterp, plus, minus, minusp, numberp, go,
nconc, reverse, nreverse, print, trace, traceset, debug.
I could not find the explanation of prog, progn, nconc,
traceset and debug.

(defun func-test-carx (S) ; function test: car
  (format t "You entered the following : ~S~%" S)
car
  (format t "The cmd (car subject) picks the element : ~S~%" (car S))
cdr
  (format t "The cmd (cdr subject) leaves the element : ~S~%" (cdr S))
caddr
  (format t "The cmd (caddr subject) leaves the element : ~S~%" (caddr S))
cons
  (format t "The cmd (cons 14 input) adds 14 to the input : ~S~%" (cons 14 S))
list
  (format t "The cmd (list 23 input) creates a list starting with 23 followed by input : ~S~%" (list 23 S))
set
  (format t "The cmd (set ‘a 11) defines the quoted variable ‘a’ with the value 11 : ~S~%" (set ‘a 11))
setq
  (format t "The cmd (setq b 55) defines the variable ‘b’ with the value 55 : ~S~%" (setq b 55))
cond
  (format t ": comes from (cond ((condition_1: (> a b)) (action_1: print '(a is bigger)))~% ((condition_2: (< a b)) (action_2: print '(a is smaller)))~%is a possibility to react on different possible statments~S~%"
    (cond ((> a b) (format t "a is bigger"))
          ((< a b) (format t "a is smaller"))
          (t (format t "they are the same")))))
atom
  (format t "The cmd (atom b) checks if b is an atom : ~S~%" (atom b))
  (format t "The cmd (atom input) checks if the input is an atom : ~S~%" (atom S))
number
  (format t "The cmd (number b) checks if b is a number : ~S~%" (number b))
  (format t "The cmd (number input) checks if the input is a number : ~S~%" (number input))
greaterp
  (format t "The cmd (> b a) checks if b is greater than a : ~S~%" (> b a))
plus
  (format t "The cmd (+ b a) adds the values : ~S~%" (+ b a))
minus
  (format t "The cmd (- b a) subtracts the values : ~S~%" (- b a))

Homework 1 1/10/2005