Laboratory: Pairs and Pair Structures

Summary: In this laboratory, you will further ground your understanding of what happens “behind the scenes” when Scheme deals with lists and other pair-based structures.

Contents:

- **Exercises**
  - Exercise 0: Preparation
  - Exercise 1: Some Pictures
  - Exercise 2: Some Pairs
  - Exercise 3: More Pictures
  - Exercise 4: Are They Pairs?
  - Exercise 5: Is It A List?
- **If You Have Extra Time**
  - Extra 1: Finding the Last element
  - Extra 2: Rewriting listp?
- **Notes**

Exercises

Exercise 0: Preparation

Make sure you have some blank pieces of paper (lined is okay) and something with which to write.

Exercise 1: Some Pictures

Draw box-and-pointer diagrams for each of the following lists:

- ((x) y z)
- (x (y z))
- ((a) b (c () )

Be prepared to share your pictures with me.

Exercise 2: Some Pairs

Enter each of the following expressions into Scheme. In each case, explain why Scheme does or does not use the dot notation when displaying the value.

- (cons 'a "Walker")
- (cons 'a null)
- (cons 'a "null")
Exercise 3: More Pictures

Draw a box-and-pointer representation of the value of the last two expressions in the previous exercise.

Exercise 4: Are They Pairs?

What do you think that pair? will return for each of the following? How about list?. Attempt to confirm each answer experimentally and explain any that you found particularly tricky.

   • (cons 'a 'b)
   • (cons 'a (cons 'b 'c))
   • (cons 'a null)
   • null
   • (list 'a 'b 'c)
   • (list 'a)
   • (list)

Exercise 5: Is It A List?

You may recall that I told you that many kinds of data are defined recursively. For example, a list is either (1) null or (2) cons of anything and a list.

Using that recursive definition of lists, write a procedure, (listp? val), that determines whether or not val is a list.

You may not use list? in your definition of listp?.

If You Have Extra Time

If you were able to complete the primary exercises with time to spare, you might want to consider the following problems:

Extra 1: Finding the Last element

Write a procedure, (last pairthing) that finds the “last” value in a list-like pairs structure. If the pair structure is actually a list, return the last element of the list. Otherwise, follow the cdrs until you find the last pair, and return the cdr of that pair.

In solving this problem you should only step through the list once.
Extra 2: Rewriting `listp`?

Write `listp` without using `if` or `cond`.

Notes

Copyright © 2007 Samuel A. Rebelsky. This work is licensed under a Creative Commons Attribution-NonCommercial 2.5 License. To view a copy of this license, visit [http://creativecommons.org/licenses/by-nc/2.5/](http://creativecommons.org/licenses/by-nc/2.5/) or send a letter to Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.