MASSACHVSETTS INSTITVTE OF TECHNOLOGY
Department of Electrical Engineering and Computer Science
6.001—Structure and Interpretation of Computer Programs
Fall 2007

Recitation 5
Data Structures and Abstractions

Scheme

New procedures

1. `(cons a b)` - Makes a cons-cell (pair) from a and b
2. `(car c)` - extracts the value of the first part of the pair
3. `(cdr c)` - extracts the value of the second part of the pair
4. (`c`$\frac{a}{d}\frac{a}{d}\frac{a}{d}\frac{a}{d}$`r c`) - shortcuts. `(cadr x)` is the same as `(car (cdr x))`
5. `(list a b c ...)` - builds a list of the arguments to the procedure
6. `(define nil '())` - the special object `'()`, called the empty list, denotes the end of a list. We often write this as `nil` instead of `'()`.
7. `(null? a)` - returns `#t` if `a` is the empty list (`nil` or `'()`), and `#f` otherwise.

Problems

1. Draw box-and-pointer diagrams for the values of the following expressions. Also give the printed representation.

   (a) `(cons 1 2)`

   (b) `(cons 1 (cons 3 (cons 5 '())))`

   (c) `(cons (cons (cons 3 2) (cons 1 0)) '())`

   (d) `(cons 0 (list 1 2))`

(e) `(list (cons 1 2) (list 4 5) 3)`

2. Write expressions whose values will print out like the following.

   (a) `(1 2 3)`

   (b) `(1 2 .  3)`

   (c) `((1 2) (3 4) (5 6))`

3. Create a data abstraction for points in a plane. It should have a constructor, (`make-point` *x y*), which returns a point, and two selectors (`point-x` *pt*) and (`point-y` *pt*), which return the x and y coordinates.

4. Now, extend the point abstraction to handle line segments, with a constructor (`make-line-segment` *pt*1 *pt*2), and selectors `line-segment-start` and `line-segment-end`.

5. Write a procedure (`intersection` *seg*1 *seg*2) that returns a point where two line segments intersect if they do, and returns #f if they do not intersect. Be sure to honor the abstractions defined.