CS 61A Structure and Interpretation of Computer Programs
Spring 2017

Quiz 8 Solutions

1. (1.5 points) Scheme Primer (Conceptual)

(a) Describe all interpretations of Scheme parentheses that you can think of (in other words, say you see some parentheses... what could their meaning be?).

Parentheses either denote procedure calls or special forms. Importantly, note that every set of parentheses counts; you can never leave them out and you can never add more.

(b) Do you enjoy counting parentheses? Circle one: Yes

(c) What is a symbol in Scheme?

Symbols are like code itself – specifically symbols are immutable, interned strings. You can think of them as variable names; in this way symbols will come in handy where interpreters are concerned!

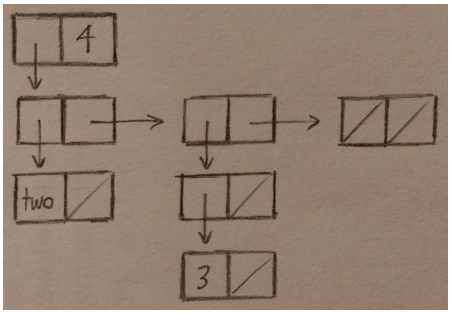
2. (2 points) WWSP?

```
scm> '((list 2 3))
((list 2 3))

scm> (list '(2 3))
((2 3))

scm> (define x (+))
x
scm> (define y +)
y
scm> (x 3 4)
Error:  cannot call:  0

scm> (y 3 4)
7
```

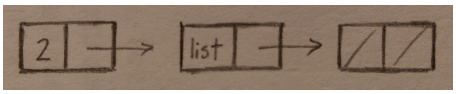
3. (2.5 points) Box and Pointers

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

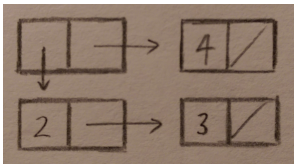
```
scm> '(2 . 3 4)
Error; you can only have a single element after a dot!

scm> (cons (list '(two) '((3)) nil) 4)
```



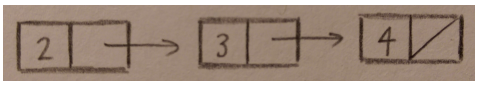
```
scm> (cons 2 '(list nil))
```




```
scm> (list (append '(2) '(3) nil) 4)
```




```
scm> '(2 . (3 . (4)))
```



4. (4 points) Last One

Write a function `take` that takes in a list `s` and a positive number `n`, and returns a list `t` such that `(car t)` is the first `n` elements of `s` and `(cdr t)` is the remaining elements of `s`. If `n` is greater than the length of `s`, `(car t)` should be `s` and `(cdr t)` should be `nil`.

```
(define (take s n)
  (cond ((= n 0) (cons nil s))
        ((null?  s) (cons s nil))
        (else (let ((rec (take (cdr s) (- n 1))))
              (cons (cons (car s) (car rec)) (cdr rec))))
  )
)
```

Example usage:

```
scm> (define a (take '(1 2 3) 2))
scm> (car a)
(1 2)
scm> (cdr a)
(3)
scm> (define b (take '(1 2 3) 4)) ; n > (length s)
scm> (car b)
(1 2 3)
scm> (cdr b)
()
```