CS37 – Fall 2009

Midterm – Take-home part

There are 40 points on this part of the midterm exam. You may use the textbook, drscheme help, and any material I have given you in clabs or from my pub directory. Do not discuss this with any person other than Charles Kelemen. Except for the material mentioned in the sentence above, what you email in should be your work alone.

1. Object Oriented Programming (20 points)

Write, debug, and test Scheme code to model a basic object which will represent a fraction. **Your object must be able to accept messages using the `ask` function.**

The fraction object will have two data fields (representing the numerator and the denominator). The object will support three methods, called using the messages `num`, `den`, and `times`. The num and den methods take no parameters and return the numerator and denominator, respectively. The times method takes one parameter (another fraction) and returns a new fraction equal to multiplying the two fractions together. Do not worry about reducing the fraction. The only error checking you need to do is to ensure that the message passed to the object is valid.

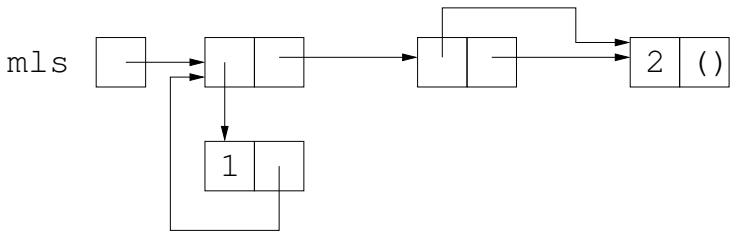
Below is the definition of the ask function and then some example uses which your code should support.

```
(define ask
  (lambda (obj msg . args)
    (apply (obj msg) args)))


(define f (make-fraction 3 5))
(ask f 'num)  ; 3
(ask f 'den)  ; 5
(define g (make-fraction 4 7))
(define h (ask f 'times g))   ; f and g are unchanged
(ask h 'num)  ; 12
(ask h 'den)  ; 35
```

2. Mutation (20 points)

(a) (5 points) Define the variable `mls` as shown in the box-and-pointer diagram below:



(b) (15 points) Write the function `member*` which takes two parameters `item` and `lst` and returns true if the `item` is in the `lst`; otherwise, returns false. The list `lst` can be both a deep list and can have cycles. The list `mls` from the previous question is an example of an acceptable input list.

```
(member* 1 mls) ; #t
(member* 2 mls) ; #t
(member* 7 mls) ; #f
```