MASSACHUSETTS INSTITUTE OF TECHNOLOGY
Department of Electrical Engineering and Computer Science
6.001 Structure and Interpretation of Computer Programs
Spring, 2007

Recitation 2, February 9

Lambda Problems Dr. Kimberle Koile

Part 1

For each of the following expressions or sequences of expressions, state the value returned as a result of evaluating the final expression in each set, or indicate that the evaluation results in an error. If the result is an error, state in general terms what kind of error occurs.

1. ((lambda (a b) (a b))
   (lambda (c) (* 2 c))
   10)

2. (lambda (m n)
   (if m (* 2 n) (/ 2 n)))

3. ((lambda (x y z) (x y z)) < 10 20)

4. ((lambda (x y z) (z y x)) < 10 20)

5. (define three (lambda (a) (* a a a)))
   (three (three 2))

6. (define (double x) (* 2 x))
   (define (check y)
   (cond ((< (double y) 10) "foo")
   ((> (double y) 6) “bar”)
   (else “baz”)))
   (check 5)

*When will “baz” be returned?

Part 2

Write the following procedures.

1. *circle-area*, which takes the radius of a circle as its argument and returns the area of the circle. Assume pi is already defined as below.

(define pi 3.14159)

2. *sign*, which takes a number as its argument and returns -1 if number is negative, 1 if it is positive, 0 if it is 0.

3. *max*, which takes two numbers as arguments and returns the larger one.

Looking ahead a few lectures:

4. *average*, which takes a list of arguments and returns the average of the list's elements.
e.g., (average (list 2 4 6)) => 4

Assume that these procedures have been defined:

length, which takes a list as an argument and returns the number of elements in the list
e.g., (length (list 1 2 3)) => 3

apply, which takes as arguments a procedure and a list, and calls the procedure with the list's elements as arguments e.g., (apply + (list 1 2 3)) => (+ 1 2 3) => 6