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

Recitation 16b, April 18

Higher Order Procedures Practice (Mike Leventon) **Dr. Kimberle Koile**

1. Write a function *swap* that takes a function f, and returns a function that takes two arguments, and returns f with the variables swapped: (f x y) == ((swap f) y x) For example, ((swap -) 4 5) 1.

2. Composing functions

Now try to write the function *compose* that takes two functions, f and g, and returns a function, that takes one argument, and composes f and g on that argument.

Example: composing double and cube means take the double of the cube of x
((compose double cube) 3) => (double (cube 3)) => 54

3. Using compose, define the function f^3/2 which takes a number x and computes x $^{3/2}$.

4. Repeated Composition of Functions

We saw how to compose two functions to produce another function. For example, we can define the following:

```
   (define fourth-power (compose square square))
   (define eight-power (compose square (compose square square)))
... and so on ...
```

Write a recursive procedure called *repeat* that takes a function f and an integer n, and composes f, n times. For example:

```
  (define fourth-power (repeat square 2))
  (define eight-power (repeat square 3))
... and so on ...
```

5. Iterative Repeat: Write a version that creates the repeat procedure iteratively by calling compose. (Note: The procedure created will run as a recursive procedure.)

6. Iterative Repeated 2: Write a version of *repeat* that creates a procedure that will run as an iterative procedure. (Hint: Do not use compose.)