Tutorial 9: Lexical Lessons and Family Fun

CS 135 Fall 2007

November 7-9, 2007

Today's tutorial really has two parts. First, we get some practice with the semantics of `local` and lexical scoping rules. Then, we make use of `local` as well as the ideas of functional abstraction to write some pairs of functions on family trees. This comes from lecture module 9 and the the first half of 10 in the course notes. You do not need `lambda` for this tutorial.

1 Local!

For each of the following examples, follow through the full semantic substitutions step by step until no more substitutions are possible or an error occurs. If an error occurs, describe it and say why it happened.

1.
```
(define (f x) (add1 x))
(f (local ((define (f x) (sub1 x)))
     (f 5)))
```

2.
```
(define a 2)
(define (f x)
  (local (define (a 5))
    (sqr a)))
(f -10)
```

3.
```
(define (f x)
  (local ((define (g y)
            (cond [(empty? y) 5]
                  [else (* (first x)
                           (f (rest y)))]))))
```

```
    (g (rest x))))
(f (list 2 3 4 5 6))
(f (list 1 2))
```

4.
```
(define a 2)
(local ((define b 4)
        (define a (add1 b)))
  (+ a b))
(+ a b)
```

2 Functional Abstraction using Family Trees

Recall that a family tree (ft) is either `empty` or

```
(make-child f m n d e)
```

where `f` and `m` are family trees, `n` and `e` are symbols (for name, eye color), and `d` is a number (for year of birth).

Each of the following exercises gives two functions to write on family trees. For each one, abstract out as much as possible of the structure that the two functions share into a single helper function, and then use this helper function to write the two functions required. Use `local` effectively so that there are exactly three top-level definitions for each exercise.

2.1 Matching name or eye color

- `has-name?` consumes a family tree and a symbol for a name and produces true iff the given name is the name of any ancestor in the given tree.

- `has-eyes?` consumes a family tree and a symbol for a color and produces true iff someone in the given tree has the given eye color.

2.2 Averages and Ratios

- `avg-age` consumes a nonempty family tree and produces the average age of all ancestors as of December 31, 2007 (subtract each birth year from 2007 to get each age).

- `pct-eyes` consumes a nonempty family tree and a symbol for a color and produces the percentage of ancestors in the given tree with the given eye color.

2.3 Least and Most

- `rarest-eyes` consumes a nonempty family tree and produces the name of one ancestor with the rarest eye color in the given tree.
- Define the 'gap' of a child to be the largest difference in age between the child and one of his/her parents. The function `biggest-gap` consumes a nonempty family tree and produces the name of an ancestor with the largest gap between the age of one of the ancestor's parents.