CSE 341 — Scheme Metacircular Interpreter Discussion Questions Answers

Suppose we have read in the following definitions into the interpreter:

```
(define (double n)
  (* 2 n))

(define (map f s)
  (if (null? s)
      '()
      (cons (f (car s)) (map f (cdr s)))))

(define (multiplyall c s)
  (map (lambda (b) (* b c)) s))
```

1. Suppose we evaluate `(double 6)`. What is the result? What is the environment used when evaluating the body of `double`?

   Result: 12

   Environment used when evaluating the body of `double`: a list consisting of two frames. The first frame just contains the variable `n` and value `6`. The second frame is the global environment, and has bindings for `double`, `map`, `multiplyall`, `cons`, `car`, etc. (Sorry - too hard to draw a picture!)

2. Suppose we evaluate `(map double '(1 2 3))`. What is the result? What is the environment used when evaluating the body of `double`?

   Result: (2 4 6)

   Environment used when evaluating the body of `double`: pretty much the same – a list consisting of two frames. We do this 3 times. The first time we evaluate, the first frame just contains the variable `n` and value `1`. The second frame is the global environment, and has bindings for `double`, `map`, `multiplyall`, `cons`, `car`, etc. The second time we evaluate, it's the same except that n is 2; and it's 3 for the third time.

3. Suppose we evaluate `(map (lambda (n) (+ n 1)) '(1 2 3))`. What is the result? What is the environment used when evaluating the body of the lambda?

   Result: (2 3 4)

   Environment used when evaluating the body of the lambda: again, about the same! It's a list consisting of two frames. We do this 3 times. The first time we evaluate, the first frame just contains the variable `n` and value `1`. The second frame is the global environment, and has bindings for `double`, `map`, `multiplyall`, `cons`, `car`, etc. The second time we evaluate, it's the same except that n is 2; and it's 3 for the third time.

4. Suppose we evaluate `(multiplyall 10 '(1 2 3))`. What is the result? What is the environment used when evaluating the body of the lambda?

   Result: (10 20 30)

   Environment used when evaluating the body of the lambda: finally something different! We evaluate it 3 times. Each time it's a list consisting of three frames. The first time we evaluate, the first frame contains the variable `b` and value `1`. The second frame consists of two variables, `c` and `s`, which are bound to 10 and (1 2 3) respectively. The third frame is the global environment, and has bindings for `double`, `map`, `multiplyall`, `cons`, `car`, etc. Notice that the second frame is the environment in which the lambda was closed, and we need it to get the value for `c` to evaluate the body of the lambda.

5. Suppose we evaluate the following expressions. What is the result? How is this converted to lambdas in the interpreter? What is the environment used when evaluating the body of the innermost let?

   ```
   (define z 5)
   (let ((x 10))
     (let ((x 30)
           (y x))
       (+ x y z)))
   ```

Result: 45

This is converted to lambdas, in two steps. First we get:

```
((lambda (x) (let ((x 30) (y x)) (+ x y z))) 10)
```

Then when we evaluate the body of the lambda, the inner let gets rewritten as:

```
((lambda (x y) (+ x y z)) 30 x)
```

Environment used when evaluating the body of the innermost let: It's a list of 3 frames. The first frame contains the variables `x` and `y` with values 30 and 10. The second frame consists of one variable, `x`, bound to 10. The third frame is the global environment, and has bindings for `z` and all the other global variables.